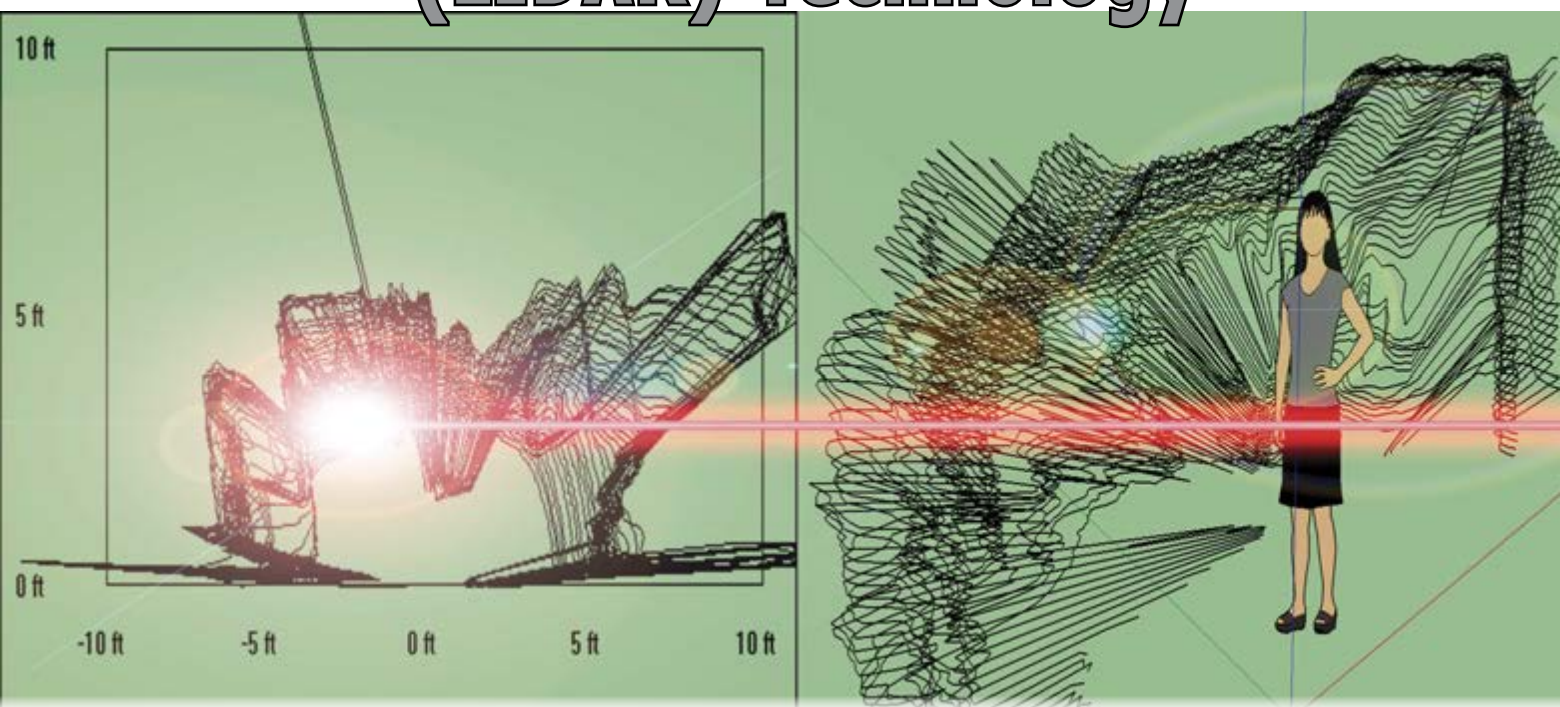




circuit cellar

# REMOTE SENSING WITH PULSED LIGHT

## Map Your Environment with Light Detection & Ranging (LIDAR) Technology



■ New Innovations in Wearable Tech ■ Embedded Serial Communications |

Acoustic Recording App (Part 2) | Security Network Image Processing |

■ Find & Eliminate Ground Loops | How to Use a PSoC's Programmable Logic |

IoT Design: Tips for Choosing a Wireless Carrier | Vintage Electronic Calculators

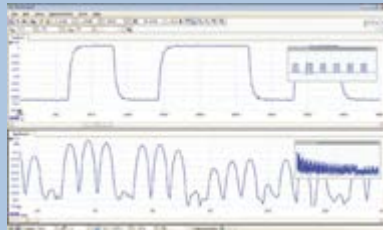
■ Custom Peripheral Cores for Digital Sensor Interfaces



# PC OSCILLOSCOPES

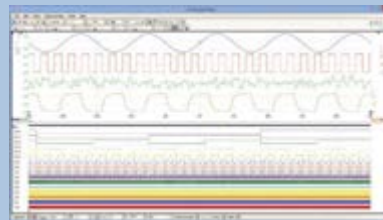


## Low cost



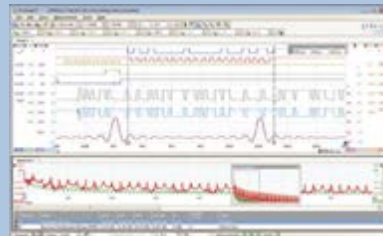
- 10 MHz to 200 MHz bandwidth
- 100 MS to 1 GS/s sampling
- 8 bit resolution (12 bit enhanced)
- 8 to 48 kS buffer memory
- USB powered
- Prices from \$131

## MSO



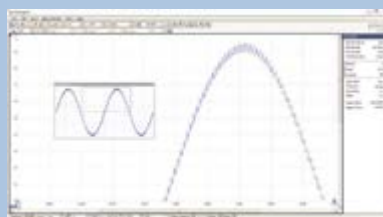
- 2 or 4 analog channels + 16 digital
- 50 to 200 MHz bandwidth
- 8 bit resolution (12 bit enhanced)
- 64 to 512 MS buffer memory
- USB or AC adaptor powered
- Prices from \$824

## Eight channels



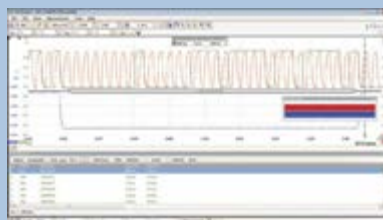
- 20 MHz bandwidth
- 80 MS/s sampling
- 12 bit resolution (16 bit enhanced)
- 256 MS buffer memory
- USB powered
- Just \$2302

## Flexible resolution



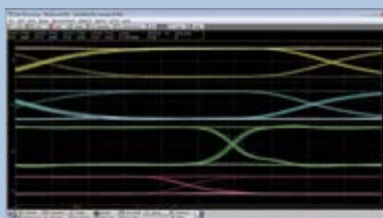
- 8, 12, 14, 15 & 16 bits all in one device
- 60 to 200 MHz bandwidth
- 250 MS/s to 1 GS/s sampling
- 8 to 512 MS buffer memory
- USB or AC adaptor powered
- Prices from \$1153

## 2 GS memory



- 250 MHz to 1 GHz bandwidth
- 5 GS/s sampling
- 8 bit resolution (12 bit enhanced)
- 256 MS to 2 GS buffer memory
- AC adaptor powered
- Prices from \$3292

## 20 GHz sampling



- DC to 20 GHz bandwidth
- 17.5 ps rise time
- 16 bit, 60 dB dynamic range
- AC adaptor powered
- Sig. gen, CDR, diff. TDR/TDT
- Prices from \$14,995

Full software included as standard with serial bus decoding and analysis (CAN, LIN, RS232, I2C, I2S, SPI, FlexRay), segmented memory, mask testing, spectrum analysis, and software development kit (SDK) all as standard, with free software updates. Five years warranty real time oscilloscopes, 2 years warranty sampling oscilloscopes.

[www.picotech.com/pco540](http://www.picotech.com/pco540)  
1-800-591-2796





## RTOS & Development Tools Complete Development Suite



Powerful Integrated Development Environment  
Proven RTOS and the Fastest JTAG/SWD Emulators



NEW



RTOS

SEGGER Embedded Studio

J-Link Debug Probes

# The Embedded Experts

Learn more: [www.segger.com](http://www.segger.com)

Contact: [info@segger-us.com](mailto:info@segger-us.com)

Issue 301 August 2015 | ISSN 1528-0608

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

Circuit Cellar, Inc.  
111 Founders Plaza, Suite 904  
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 904  
East Hartford, CT 06108, E-mail: newproducts@circuitcellar.com

#### HEAD OFFICE

Circuit Cellar, Inc. 111 Founders Plaza, Suite 904  
East Hartford, CT 06108  
Phone: 860.289.0800

#### COVER PHOTOGRAPHY

Chris Rakoczy, www.rakoczyphoto.com

#### COPYRIGHT NOTICE

Entire contents copyright © 2015 by Circuit Cellar, Inc. All rights reserved. Circuit Cellar is a registered trademark of Circuit Cellar, Inc. Reproduction of this publication in whole or in part without written consent from Circuit Cellar, Inc. is prohibited.

#### DISCLAIMER

Circuit Cellar® makes no warranties and assumes no responsibility or liability of any kind for errors in these programs or schematics or for the consequences of any such errors. Furthermore, because of possible variation in the quality and condition of materials and workmanship of reader-assembled projects, Circuit Cellar® disclaims any responsibility for the safe and proper function of reader-assembled projects based upon or from plans, descriptions, or information published by Circuit Cellar®.

The information provided by Circuit Cellar® is for educational purposes. Circuit Cellar® makes no claims or warrants that readers have a right to build things based upon these ideas under patent or other relevant intellectual property law in their jurisdiction, or that readers have a right to construct or operate any of the devices described herein under the relevant patent or other intellectual property law of the reader's jurisdiction. The reader assumes any risk of infringement liability for constructing or operating such devices.

© Circuit Cellar 2015 Printed in the United States

## 40 YEARS OF TECH IN 84 PAGES

Whether you're interested in old-school, 1970s-era technology (e.g., retro calculators) or cutting-edge embedded systems, we've got you covered. No matter your background or current profession, this issue is sure to fascinate and inspire you.

Interested in a low-speed serial communications protocol using the standard USART available on most embedded devices? On page 10, Chuck Baird explains how to configure one master and up to 31 slave devices on a single daisy-chained serial line.

Turn to page 18 for the second installment in the "Sound Ecology and Acoustic Health" article series. It addresses the topics of recording and playing back audio .3GPP files.

On page 30, Claudiu Chiculita wraps up his series on the process of building a microcontroller-based, 'Net-connected security system. This month he covers network functionality, as well as image processing and messaging.

A ground loop can cause harmful noise and interference in an electronic system. In "Ground Loop Blues," George Novacek shares some advice on finding and eliminating ground loops in analog AV systems (p. 44).

APSoC has a small amount of programmable logic for simple operations. On page 48, Colin O'Flynn explains how to use a PSoC when you need just a bit of glue logic.

On page 54, Bob Japenga continues his series on the Internet of Things. In addition to offering a few options for selecting a carrier for an embedded device, he covers OTA technologies, project budgeting, and more.

Do you love vintage electronics calculators? Robert Lacoste does. This month he shares several examples from his collection of retro calculators (p. 58).

In the second part of the series titled "Light Detection and Ranging," Jeff Bachiochi addresses laser sensor exploration (p. 66). He also looks into the communication via the I<sup>2</sup>C interface.

We wrap up the issue with an essay by Daniel Casner on the future of custom peripheral cores for digital sensor interfaces. After you read it, let us know what you think!

**C. J. Abate**

cabate@circuitcellar.com



Retro calculators



Embedded serial communications

## THE TEAM

### EDITOR-IN-CHIEF

C. J. Abate

### ART DIRECTOR

KC Prescott

### ADVERTISING COORDINATOR

Kim Hopkins

### PRESIDENT

Hugo Van haecke

### COLUMNISTS

Jeff Bachiochi (From the Bench), Ayse K. Coskun

(Green Computing), Bob Japenga (Embedded in Thin Slices), Robert Lacoste (The Darker Side), Ed Nisley (Above the Ground Plane), George Novacek (The Consummate Engineer), and Colin O'Flynn (Programmable Logic in Practice)

### FOUNDER

Steve Ciarcia

### PROJECT EDITORS

Chris Coulston, Ken Davidson, and David Tweed

### OFFICE ASSISTANT

Debbie Lavoie



## OUR NETWORK



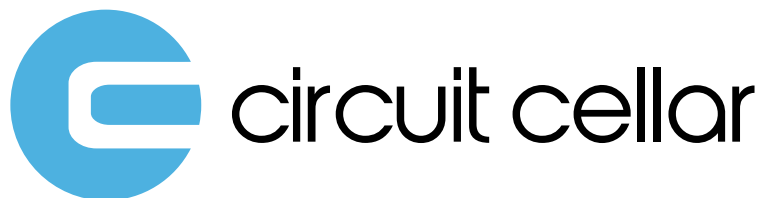
## SUPPORTING COMPANIES

Accutrace	7	MaxBotix, Inc.	79
All Electronics Corp.	79	Measurement Computing Corp.	29
Custom Computer Services	15, 79	Mouser Electronics	21
EMAC, Inc.	47	MyRO Electronic Control Devices, Inc.	79
Flash Memory Summit	39	NetBurner, Inc.	23, 33
Front Panel Express	9	NKC Electronics	9
Hot Chips Symposium	73	PCB West Conference & Exhibit	37
HuMANDATA, Ltd.	25	Pico Technology	C2
IAR Systems	17	RoboBusiness 2015	63
Imagineering, Inc.	C4	Saelig Co., Inc.	47
Ironwood Electronics	79	SEgger Microcontroller Systems	1
Jeffery Kerr, LLC	79	Technologic Systems	13
Lauterbach GmbH	57	Teledyne LeCroy Corp.	53

## NOT A SUPPORTING COMPANY YET?

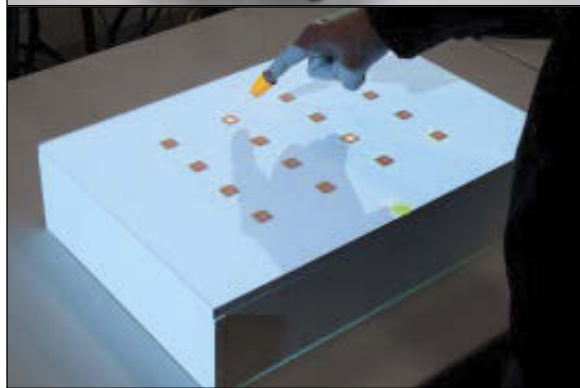
Contact Peter Wostrel (circuitcellar@smmarketing.us, Phone 978.281.7708, Fax 978.281.7706)  
to reserve your own space for the next edition of our members' magazine.

## CONTENTS

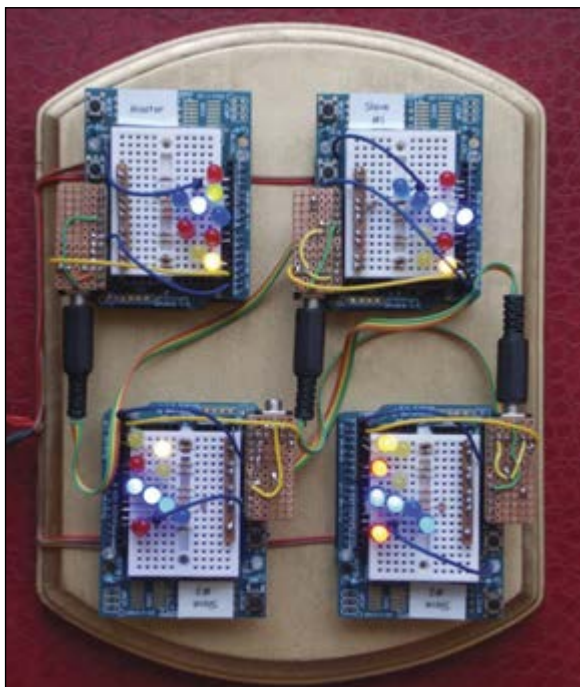


AUGUST 2015 • ISSUE 301

# EMBEDDED DEVELOPMENT



ADVANCES IN WEARABLES, VIRTUAL REALITY, &amp; MORE



EMBEDDED SERIAL COMMUNICATIONS

## INDUSTRY & ENTERPRISE

### 06 : PRODUCT NEWS

### 09 : CLIENT PROFILE

EMA Design Automation (Rochester, NY)

## FEATURES

### 10 : Simple Embedded Serial Communications

*By Chuck Baird*

A low-speed, low-volume serial communications protocol using the USART in most embedded devices

### 18 : Sound Ecology and Acoustic Health (Part 2)

Record and Play Audio .3GPP Files

*By Adrien Gaspard & Mike Smith*

How to record and play back .3GPP files

### 30 : 'Net-Connected Security Network (Part 2)

Image Processing, Messaging, and Movement Monitoring

*By Claudiu Chiculita & Liviu Ene*

Node hardware and an overview of the finished security network's functionality

## CC COMMUNITY

### 40 : QUESTIONS & ANSWERS

#### Innovations in Wearable Technology

An Interview with Bruce Thomas

A look at exciting new research at the University of South Australia's Wearable Computer Lab

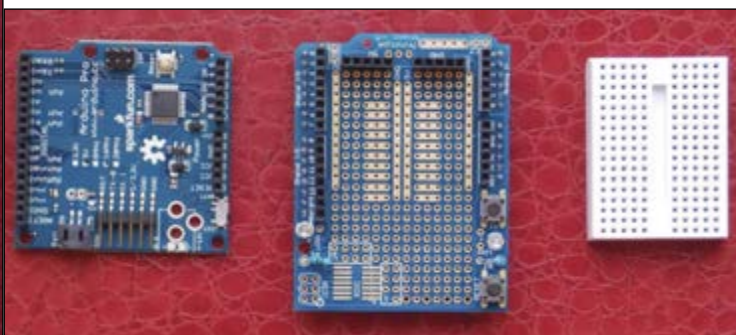
## COLUMNS

### 44 : THE CONSUMMATE ENGINEER

#### Ground Loop Blues

*By George Novacek*

Tips for locating and fixing ground loops in analog AV systems





## CONTENTS



SECURITY SYSTEM IMAGE PROCESSING AND MESSAGING



VINTAGE ELECTRONIC CALCULATORS

### 48 : PROGRAMMABLE LOGIC IN PRACTICE

#### Just Enough Programmable Logic

The PSoC Advantage

By Colin O'Flynn

Tips for using a PSoC's programmable logic for simple operations

### 54 : EMBEDDED IN THIN SLICES

#### The Internet of Things (Part 2)

Choosing a Wireless Carrier for Your Embedded System

By Bob Japenga

Essential topics such as OTA technologies, mobile virtual network operators, project budgeting, and more

### 58 : THE DARKER SIDE

#### Vintage Electronic Calculators

By Robert Lacoste

An overview of Robert's growing collection of vintage electronic calculators

### 66 : FROM THE BENCH

#### Light Detection and Ranging (Part 2)

Laser Sensor Exploration

By Jeff Bachiochi

Laser sensor exploration and communication via the I<sup>2</sup>C interface

## TESTS & CHALLENGES

### 77 : TEST YOUR EQ

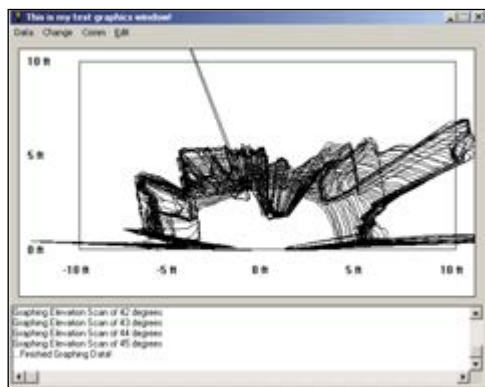
### 78 : CROSSWORD

## TECH THE FUTURE

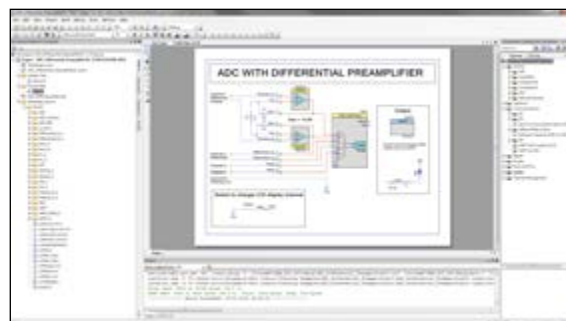
### 80 : Trends in Custom Peripheral Cores for Digital Sensor Interfaces

By Daniel Casner

Creating custom soft cores that use fewer resources and are easier to program than a state machine



LASER SENSOR DATA PLOTTING



USING A PSoC FOR SIMPLE OPERATIONS



@editor\_cc  
@circuitcellar



circuitcellar

## PRODUCT NEWS

### HIGH-RESOLUTION RESISTIVE SENSING SIGNAL CONDITIONER

Texas Instruments recently introduced the PGA900 high-resolution resistive sensing signal conditioner. The PGA900 enables the fast and precise 24-bit measurement of conditions such as pressure, flow, strain, or liquid levels. Its programmable core enables flexible linearization and temperature compensation for numerous resistive bridge sensing applications.

Key features and benefits include:

- Fast, precise sensor signal and temperature compensation: Integrates two 24-bit ADCs to provide high-resolution signal acquisition. Low-drift voltage reference of 10 ppm/°C, maximum, enables high accuracy across the -40°C to 150°C operating temperature range.
- Integrated 14-bit DAC: Enables highly linear analog outputs.
- User-programmable temperature and nonlinearity compensation algorithms: Integrated ARM Cortex-M0 core allows developers to use proprietary temperature and nonlinearity compensation algorithms to differentiate their end products.
- Simple calibration: One-wire interface allows communication, configuration and calibration through the power supply pin without using additional lines.
- Wide input voltage allows direct connection to the power supply: Integrated power management circuitry accepts input voltages ranging from 3.3 to 30 V to simplify the design and provide reliability.

With the PGA900 evaluation module (EVM), you can to quickly and easily evaluate the device's performance and integrated features. The PGA900EVM is available for \$249. You can download PGA900 example software and the user's guide, as well as the PSpice and TINA-TI Spice and TINA-TI models, at [www.ti.com](http://www.ti.com).

The PGA900 resistive sensing conditioner comes in a 6 mm × 6 mm very thin quad flat no-lead (VQFN) package. It costs \$4.50 in 1,000-unit quantities.

**Texas Instruments | [www.ti.com](http://www.ti.com)**

### Industry's highest resolution resistive sensing conditioner

- Superior analog functions with dual 24-bit ADCs
- Programmable core
- Complete one chip solution

**TEXAS INSTRUMENTS**

### HIGH-SIDE CURRENT/POWER SENSOR

Microchip Technology recently introduced the PAC1921, a high-side current sensor with both a digital output, as well as a configurable analog output that can present power, current or voltage over the single output pin. Simultaneously, all power related output values are also available over the 2-Wire digital bus, which is compatible with I2C. The PAC1921 is available in a 10-lead 3 × 3 mm VDFN package. It was designed with the 2-Wire bus to maximize data and diagnostic reporting, while having the analog

output to minimize data latency. The analog output can also be adjusted for use with 3-, 2-, 1.5-, or 1-V microcontroller inputs.

The PAC1921 is ideal for networking, power-distribution, power-supply, computing and industrial-automation applications that cannot allow for latency when performing high-speed power management. A 39-bit accumulation register and 128 times gain configuration make this device ideal for both heavy and light system-load power measurement, from 0 to 32 V. It has the ability to integrate more than two seconds of power-consumption data. Additionally, the PAC1921 has a READ/INT pin for host control of the measurement period; and this pin can be used to synchronize readings of multiple devices.

The PAC1921 is supported by Microchip's \$64.99 PAC1921 High-Side Power and Current Monitor Evaluation Board (ADM00592). The PAC1921 is available for sampling and volume production, in a 10-lead 3 × 3 mm VDFN package, starting at \$1.18 each in 5,000-unit quantities.

**Microchip Technology | [www.microchip.com](http://www.microchip.com)**



**PAC1921 High-Side Power and Current Monitor  
Evaluation Board  
(Part # ADM00592)**



PRINTED CIRCUIT BOARDS

THINK YOU CAN FIND PCB PRICES THAT BEAT OURS?

**WE DARE YOU.**

If you do, than we  
will match the price  
AND give you \$100  
towards your  
next order!

**THERE ARE NO GAMES INVOLVED IN OUR PRICING**



### Our Capabilities:

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

Take the Accutrace Challenge and see WHY OUR PRICING CANNOT BE BEATEN

**Accutrace inc.**

[www.PCB4u.com](http://www.PCB4u.com) [sales@PCB4u.com](mailto:sales@PCB4u.com)

## PRODUCT NEWS

### VIDEO DECODER WITH MIPI-CSI2 OUTPUT INTERFACE SUPPORTS NEXT-GENERATION SoCs

Intersil Corp. recently introduced the TW9992 analog video decoder, which features an integrated MIPI-CSI2 output interface that provides compatibility with the newest SoC processors. The decoder's MIPI-CSI2 interface simplifies design by making it easier to interface with SoCs, while also lowering the system's EMI profile. The TW9992 decoder takes both single-ended and differential composite video inputs from a vehicle's backup safety camera, and is the latest addition to Intersil's video decoder product family for automotive applications.

Designed with built-in diagnostics and superior video quality, the TW9992 addresses the biggest challenges faced by automotive video systems. For example, the decoder's Automatic Contrast Adjustment (ACA) image enhancement feature overcomes a major challenge for backup camera systems by adapting to rapidly changing lighting conditions. ACA is able to automatically boost up or reduce the brightness/contrast of an image for greater visibility and safety.

In addition, vehicle backup cameras typically employ differential twisted pair cables that require designers to use an operational amplifier (op amp) in front of the video decoder to convert the differential signal to single-ended. The TW9992 decoder eliminates the need for an external op amp by supporting direct differential CVBS inputs, thus reducing system cost and board space. The built-in short-to-battery and short-to-ground detection capability on each differential input channel further enhances video performance and automotive system reliability.

Features and specifications:

- NTSC/PAL 10-bit ADC analog video decoder with 4H adaptive comb filter
- MIPI-CSI2 output interface
- Software selectable analog input control allows for

combinations of single-ended or differential CVBS

- Advanced image enhancement features: automatic contrast adjustment, and programmable hue, brightness, saturation, contrast and sharpness
- Output voltage: 1.8 to 3.3 V with 3.3 V tolerance
- Low-power consumption: 100-mW typical
- Integrated short-to-battery and short-to-ground detection tests
- AEC-Q100 qualified

The automotive-grade TW9992 analog video decoder is available in a 32-pin wettable flank QFN package. It costs \$3 in 1,000-piece quantities.

Intersil | [www.intersil.com](http://www.intersil.com)



### 700-V HVICs INCREASE SYSTEM RELIABILITY, SHRINK BOARD SPACE

Infineon Technologies recently launched a family of rugged, reliable 700-V High-Voltage ICs (HVICs) optimized for solar, power supply, uninterruptible power supplies (UPS), welding, and industrial drive applications. The 700-V offering

enables designers of high-voltage power stages to simplify their designs while making them more robust.

The new IR7xxxS series of HVICs feature sink/source ratings from 60 to 2,300 mA and utilize PN junction technology. Available in half bridge and high- and low-side configurations, the new HVICs are optimized for 700-V MOSFETs and 650-V IGBTs and offer full driver capability with extremely fast switching speeds to reduce magnetics component count.

Other key features of the new devices include under-voltage lock-out protection for both channels, lower di/dt gate driver for better noise immunity. In addition, the HVICs are tolerant to negative transient voltage dv/dt, offer matched propagation delay for both channels and are 3.3- and 15-V input logic compatible.

The new IR7xxxS series is available in surface-mount (8-SOIC) packages in high volume. The lead-free devices are RoHS-compliant.

Infineon Technologies | [www.infineon.com](http://www.infineon.com)





## CLIENT PROFILE

# EMA Design Automation

Location: Rochester, NY (USA)

Web: [www.ema-eda.com](http://www.ema-eda.com)

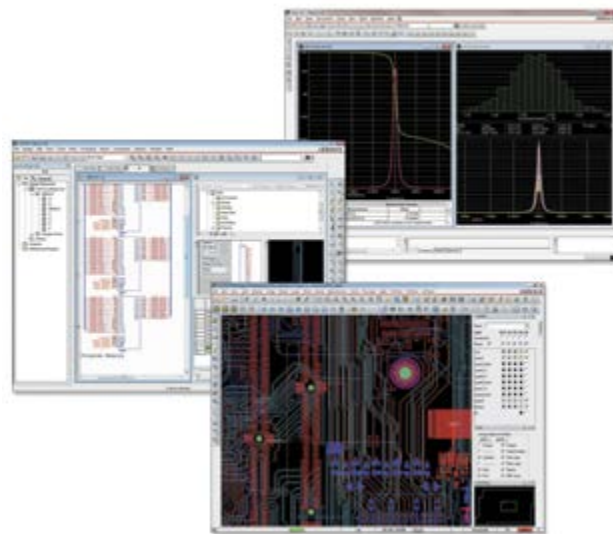
Email: [info@ema-eda.com](mailto:info@ema-eda.com)

## EMBEDDED PRODUCTS

EMA Design Automation provides PCB design software, training, and implementation services throughout North America. It is a Cadence channel partner representing the OrCAD, Allegro, and Sigrity product lines. OrCAD and Allegro provide the only fully scalable common platform PCB design solution that can grow with your design needs without requiring expensive retraining or translations.

## WHY SHOULD CC READERS BE INTERESTED?

OrCAD Lite (free) — OrCAD provides an unbeatable mix of value, capability, and performance that engineering teams across the world rely on to help meet their PCB design objectives, on-time and on-budget. OrCAD Lite provides free access to this production proven scalable PCB design technology for users of all types and budgets. Download OrCAD Lite today and experience the OrCAD advantage first-hand ([www.orcad.com/resources/orcad-lite-overview](http://www.orcad.com/resources/orcad-lite-overview)).



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.

## The Easiest Way to Design Custom Front Panels & Enclosures

Free  
Front Panel  
Designer



**You design it**  
to your specifications using  
our FREE CAD software,  
Front Panel Designer

**We machine it**  
and ship to you a  
professionally finished product,  
no minimum quantity required

- Cost effective prototypes and production runs with no setup charges
- Powder-coated and anodized finishes in various colors
- Select from aluminum, acrylic or provide your own material
- Standard lead time in 5 days or express manufacturing in 3 or 1 days

**FRONT PANEL  
EXPRESS**

[FrontPanelExpress.com](http://FrontPanelExpress.com)

## Embedded Systems Development Tools



NKCELECTRONICS

**Digilent FPGA Development Boards — from \$89**

Basys3 - Nexys4-DDR - Anvyl - CMOD S6

**Digilent Xilinx Zynq-7000 SoC Boards — from \$189**

Zybo - ZedBoard

**NI myRIO Add On Boards — from \$24.99**

Shield Adapter - NXT Sensor Adapter - Motor Adapter - Breadboards

**NI LabView Home Bundle (Windows) — \$49**

Personal License (non-commercial, non-industrial, non-academic)

**NI Physical Computing Kit for LabView — \$89**

NI LabView Home Bundle for Windows and chipKIT WF32 board

**Sparkfun Intel Edison Starter Pack — \$119.95**

Includes Intel Edison and Blocks (Base, Battery and GPIO)

**RS232 to TTL adapters — \$9.95**

3V to 5.5V - DTE w/male DE9 or DCE w/female DE9

**MDE8051 Trainer Kit with DS89C450 — \$59.99**

Includes MDE8051 Trainer board, power supply, serial cable and wires

**Arduino Starter Kit — \$99**

Genuine Arduino Starter Kit Made in Italy

**TI MSP430G2 LaunchPad bundle — \$16.45**

MSP-EXP430G2 with Breadboard and wires

**TI MSP430F5529 LaunchPad bundle — \$19.45**

MSP-EXP430F5529LP with Breadboard and wires

**Zeroplus PC Based Logic Analyzers — from \$139**

16 channels, 32k per channel, 100MHz sample rate, 58 Free Protocols

**Saleae Logic Analyzers — from \$219**

Logic 8, Logic Pro 8, Logic Pro 16

**Digilent JTAG HS3 Programming Cable — \$59**

JTAG Programming and Debugging Solution for Xilinx FPGA and SoC

**Digilent Analog Discovery — \$279**

Turn any PC into a powerful Electrical Engineering Workstation

**JYE Function Generator Kit — \$44.95**

200KHz waveform generator DIY Kit. All components included

[NKCElectronics.com/cc](http://NKCElectronics.com/cc)

Design  
Prototype

Debug  
Deploy

# Simple Embedded Serial Communications

Chuck presents a low-speed, low-volume serial communications protocol using the standard USART available on most embedded devices. With it, he can configure one master and up to 31 slave devices on a single daisy-chained serial line. Here he demonstrates a method for implementing this kind of inter-MCU communication.

*By Chuck Baird (United States)*

**I**n this article I describe a simple, low-speed, low-volume serial communications protocol using the standard USART available on most embedded devices. I can configure one master and up to 31 slave devices on a single daisy-chained serial line. There are numerous applications where multiple microcontrollers might be useful to offload tasks, and a need to communicate among them would probably be essential. I demonstrate one method for implementing this kind of inter-microcontroller communication.

I start with a brief description of the protocol, followed by an overview of the included demonstration software, which illustrates most of its features using a simple configuration of a master and three slaves. A complete description of the protocol, the demonstration source code (written in C), and more details on the design of the programs are provided in the downloadable files posted on the *Circuit Cellar* FTP site.

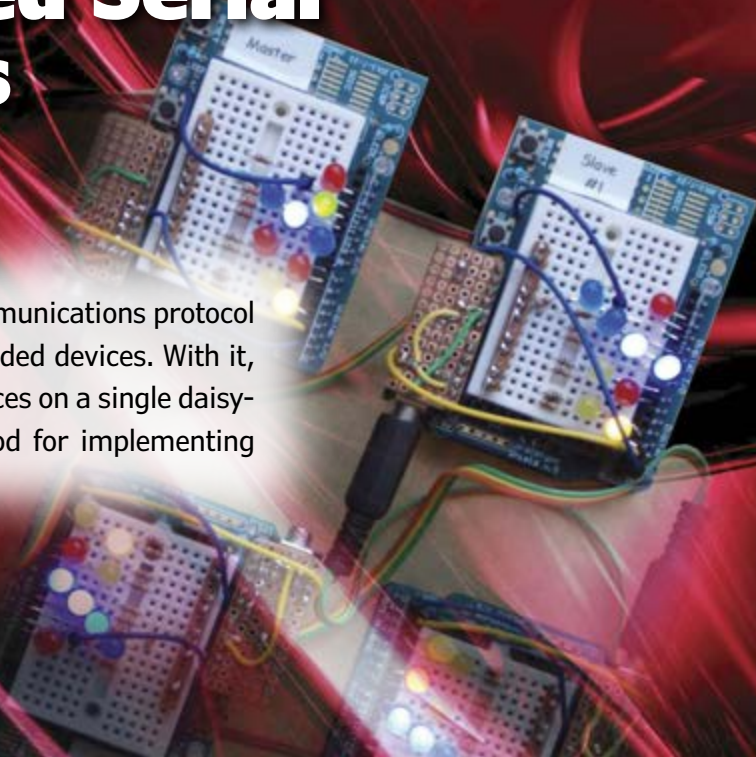
## PHILOSOPHY

The goal for designing this protocol was to enable multiple embedded devices to perform low data rate communication using no additional external active hardware. Typically, the devices would simply be wired together; but for our example, mini stereo jacks and plugs were used. The user simply plugs one slave into the master and then plugs the slaves together in a chain. The last slave's

jack is not used, and the physical order of the slave devices on the chain does not matter.

I used standard Arduino boards (SparkFun Arduino Pro 328 5 V/16 MHz clones) for both the master and slaves because they are robust, inexpensive, and easy to use. I also used Arduino prototyping shields with breadboards for the example configuration (see **Photo 1**). I didn't use the Arduino software and environment. The devices were programmed in their native AVR mode using Atmel Studio 6.2 and an Atmel STK500 programmer. The software was written in GCC C and will easily port to the Arduino environment. One advantage associated with using Arduino development boards is their ability to act either as Arduino or native AVR devices, and the easy switch between the two modes.

AVRs, and consequently Arduino boards, have more efficient and faster methods of serial communications than the USART, but the USART was used here for simplicity and portability. Arduino boards do not contain RS-232 or other line driver circuitry, so the signals between devices are at standard logic levels (3.3 or 5 V depending on the model). For this reason, all the wiring between the devices was kept short. Using logic levels makes the connections more sensitive to noise, distance, and destruction by random acts of nature than using something like RS-232 would. However, any or all of the interconnections





could use line drivers without modification to the software or protocol if noise or distance is an issue.

## DEVICE INTERCONNECTIONS

The master and slaves are connected using three wires: ground, serial transmit (Tx), and serial receive (Rx). Arduino boards vary in their operating voltages, so care must be taken to ensure that the various devices are compatible or at least tolerant of any differences. The serial lines are connected between devices so that there is a complete loop, each Tx feeding the next Rx.

For convenience, I used mini stereo jacks and plugs, which are wired as shown in **Figure 1**. With nothing plugged into the master's jack, Tx feeds Rx, and the master receives anything it sends. The protocol will detect this condition at start-up. When a slave is plugged into the master's jack, the master's Tx connects to the slave's Rx, and the slave's Tx is fed to the next slave's Rx. The last slave in the chain, the one with nothing plugged into its jack, routes its Tx back to the master's Rx to complete the loop (see **Photo 2**).

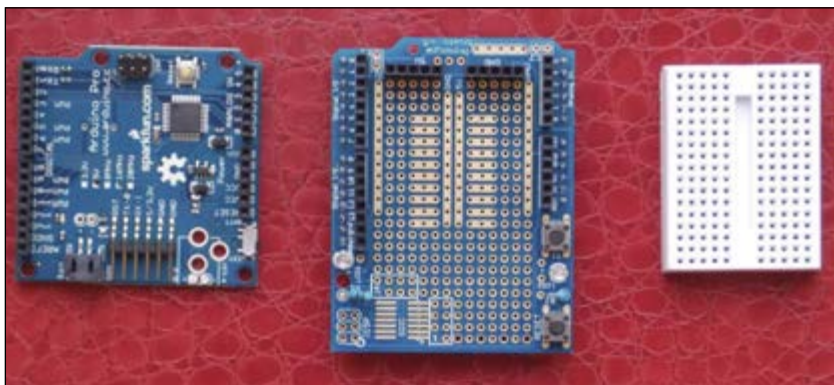
## PROTOCOL OVERVIEW

The protocol consists of four commands—INIT, RESET, DATA, and REQ—and two pseudo-commands—SYNC1 and SYNC2—used for synchronization. Five of the commands are single bytes, and the sixth (DATA) is 2 to 9 bytes long. REQ may be issued only by a slave and the other commands are only issued by the master. The master controls the serial “bus.”

For all but REQ the master transmits a command, and each slave in turn examines it with the option of modifying it. It then transmits it, so eventually each (possibly altered) command reaches the master in the order it was sent. The REQ command travels from the initiating slave to the master where it then dies.

The first command, INIT, is used by the master to make sure reliable communications are in place, initialize the slaves (if any), and assign them ID numbers. The synchronization pseudo-commands are used in conjunction with the INIT command to insure all slaves are powered up and have consistent baud rates and everyone's fun meter is pegged before proceeding.

Slave IDs are numerical, 1 to 31, and indicate the slave's position in the chain. The slave plugged into the master is assigned an ID of 1. An ID of 0 is used in some cases to indicate all slaves are being addressed. If there are more than 31 slaves in the chain, the excess slaves are disabled and simply



**PHOTO 1**

Arduino, prototyping shield, and breadboard

transmit anything they receive on their serial line.

The protocol does not care about the physical order of the slaves, and a robust application will support them plugged together in any order. An application should not depend on a specific slave having a specific ID.

The INIT command eventually returns to the master, and it will have been modified to contain the ID of the last slave in the chain. If this returned ID is 0, then there are no slaves plugged in. If 31, there is a possibility that one or more slaves may have been disabled, although the master has no way to determine this.

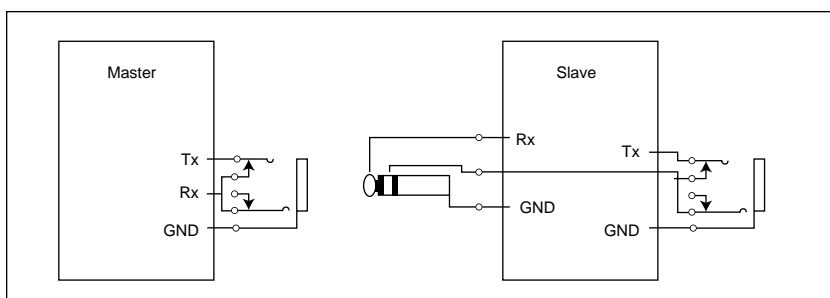
The RESET command is used by the master to do a “soft” reset of one or all slaves. What constitutes a soft reset is application dependent, but presumably it would clear any pending actions and return the slave to a known state. RESET does not alter any slave IDs.

The DATA command, issued by the master, sends a packet of data to a slave. Each packet contains the initial command byte and a descriptor byte, plus 0 to 7 optional bytes. The receiving slave alters the packet, typically removing any data, and returns it to the master with an acknowledge bit set. There are some defined packet types, but mostly the format and interpretation of the packet and its data are application dependent. The master and slave software must agree on what the data means.

The REQ command, issued only by a

**FIGURE 1**

Master/slave connections





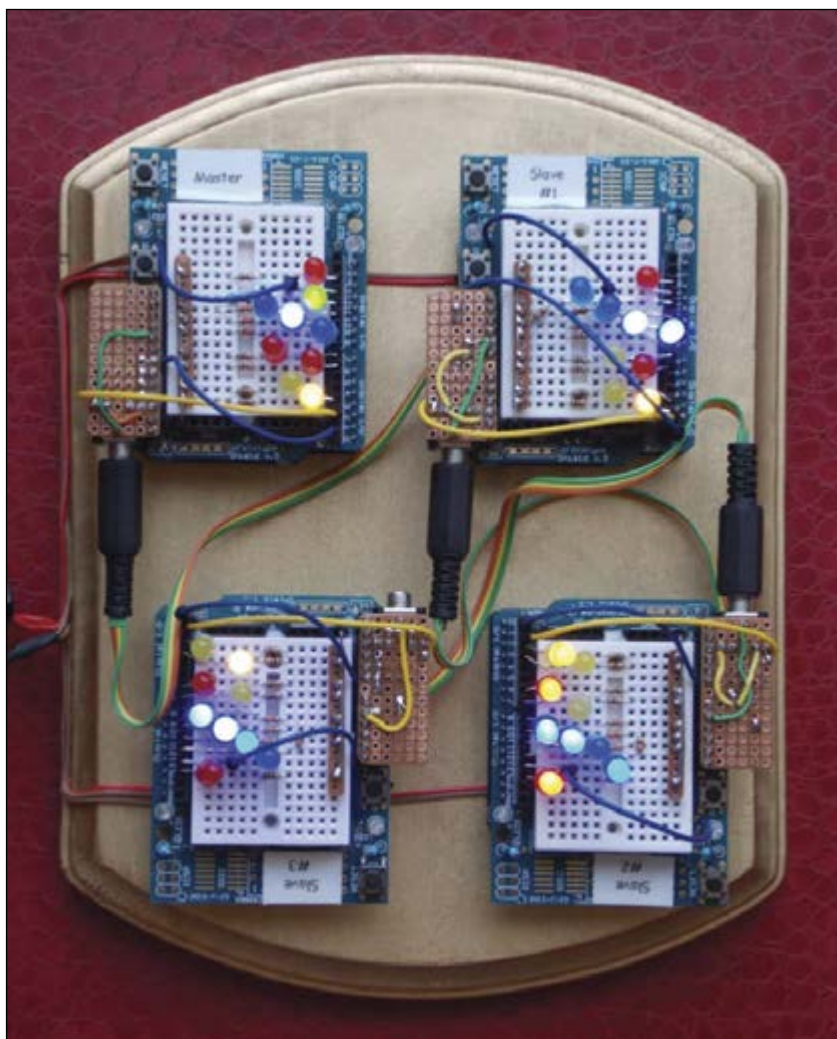


PHOTO 2

Master and three slaves

slave, requests that the master send it a data packet. When the packet arrives the slave inserts its data in the packet and returns it to the master. If the slave has more than 7 bytes to send, there is a mechanism to automatically request an additional packet from the master for the next 7 bytes. Thus, the master can send a slave unsolicited data (which it hopefully will know how to handle), and a slave can send the master data. While it is outside the protocol, with proper data packet interpretation one slave can obtain the ID of another and exchange packets by routing them through the master.

Obviously there is significant overhead in this arrangement, since every slave has to handle every byte on the chain. However, each segment of the chain is independent, so there can be multiple commands passed along simultaneously. While the protocol handling of each slave can be fairly dumb and is in fact driven by a simple finite state machine (FSM), the master has to juggle several balls at the same time. (Refer to the sidebar for additional information about FSMs.)

There are three time-critical situations, the first occurring during initialization when the master is trying to establish the chain. After a specified amount of time and/or a certain number of failed attempts the master should probably give up and somehow let the user know.

The second critical situation keeps slaves awake. Each slave, when it sees the start of a data packet, starts a timer to make sure the packet is fully received in a reasonable time. If not, the slave flags an error and resets itself out of the packet receive mode. This is necessary because commands are not recognized within packets to allow unrestricted data values. However, if something hangs or fails the slave needs to be able to snap out of its stupor and again recognize commands.

The third timeout situation occurs when the master does not receive a reply to a command. All commands eventually wind up back at the master, so a timely reply is always expected.

Specialized data packets allow the master to query each slave as to its identity and/or capabilities (application dependent) and error conditions. These packets and the slave finite state machine are described in detail in the accompanying protocol documentation file.

## MASTER & SLAVE DEVICES

The master and the slaves share some common code, but they are fundamentally different in their approach to handling the protocol. In addition, the master and each individual slave will have unique software depending on the device's purpose in the application. The master may be nothing more than a traffic cop, or it may have purposes of its own in addition to supporting the slave communications.

Although not required, each slave will probably be customized to at least identify itself to the master when queried. The format and nature of this identification is application specific. After initialization, the master typically asks each slave for identification (one or more bytes) and associates its identification with its ID. Thus, a slave may identify itself as, for example, a servo controller (appropriately encoded in the returned byte(s)), and the master, or other slaves with the master's help, may send data to the servo controller without knowing its specific ID in advance. Because of this method of addressing the physical order of the slaves on the chain is irrelevant.

As I already mentioned, slave to slave communication is not directly supported but the master may include code to handle it. Thus, it is possible to support slave requests like "send this data to all LCD slaves" and have the master transmit individual packets to

# SUPERIOR **EMBEDDED** SOLUTIONS



**DESIGN YOUR SOLUTION TODAY**  
**CALL 480-837-5200**

[www.embeddedARM.com](http://www.embeddedARM.com)

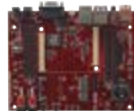


## TS-4900 Computer on Module

Industrial High Performance  
i.MX6 Module with Wireless  
Connectivity and Flash Storage

- 1 GHz Solo or Quad Core Freescale i.MX6 ARM CPU
- 512 MB, 1 GB, or 2 GB DDR3 RAM and 4 GB eMMC Flash Storage
- Wireless 802.11 b/g/n and Bluetooth 4.0 Soldered Module
- 4k LUT FPGA, 1x Gigabit Ethernet, 1x PCI Express Bus
- 1x microSD Socket, 1x SATA II, 1x USB Host, 1x USB OTG
- 70x DIO, 4x I2C, 1x I2S, 2x SPI, 2x CAN
- -40 °C to 85 °C Industrial Temperature Range
- Runs Linux, Android, QNX, Windows
- QT, OpenGL, DirectFB, GNU Tool Kit, and More

Starting at  
**\$89**  
Qty 100  
**\$122**  
Qty 1



Available w/ TS-8550  
PC/104 Development Kit



## TS-TPC-7990 Touch Panel PC

COMING  
SOON!

7" High End i.MX6 Mountable  
Panel PC with Dev Tools Such  
as Debian GNU and QTCreator

- 1 GHz Solo or Quad Core Freescale i.MX6 ARM CPU
- 7 Inch or 10 Inch Touch Panel PC
- Resistive and Capacitive Screens
- Linux, Android, QNX, and Windows
- QTCreator, GTK, DirectFB, GNU Tool Kit, and More
- Runs Yocto, Debian, Ubuntu Distributions

Starting at  
**\$299**  
Qty 100  
**\$342**  
Qty 1



Enclosed TPCs  
Also Available



## TS-7970 Single Board Computer

NEW!

Embedded Computer Version  
of the TS-4900 i.MX6 CoM with  
Dual Ethernet, Rugged Connector

- 1 GHz Solo or Quad Core Freescale i.MX6 ARM CPU
- 512 MB, 1 GB, or 2 GB DDR3 RAM and 4 GB eMMC Flash Storage
- Wireless 802.11 b/g/n and Bluetooth 4.0 Soldered Module
- 4k LUT FPGA, 2x Gigabit Ethernet, 1x PCI Express Bus
- 1x microSD Socket, 1x SATA II, 4x USB Host, 1x USB OTG
- Daughter card interface for cell modem and more
- -40 °C to 85 °C Industrial Temperature Range
- HDMI, LVDS, and Audio In/Out Connections
- Runs Linux, Android, QNX, Windows

Starting at  
**\$169**  
Qty 100  
**\$214**  
Qty 1



Also available in this form factor are the  
**TS-7670** and **TS-7680** with 454 MHz CPU



## TS-7250-V2 Single Board Computer

Extensible PC/104 Embedded  
System with Customizable  
Features and Industrial Temps

- 800 MHz or 1 GHz Marvell PXA166 ARM CPU
- 512 MB DDR3 RAM and 2 GB SLC eMMC Flash Storage
- PC/104 Connector with FPGA Driven Pins (8k or 17k LUT FPGA)
- 2x 10/100 Ethernet, 1x microSD Socket, 2x USB Host
- 75x DIO, 5x ACD, 3x RS232, 3x TTL UART, 1x RS485, 1x CAN
- -40 °C to 85 °C Industrial Temperature Range
- Preinstalled Debian Linux OS and Utilities

Starting at  
**\$165**  
Qty 100  
**\$199**  
Qty 1



Available with TS-ENC720 enclosure



We've never  
discontinued a  
product in 30 years



Embedded  
systems that are  
built to endure



Support every step  
of the way with  
open source vision



Unique embedded  
solutions add value  
for our customers

[www.embeddedARM.com](http://www.embeddedARM.com)

each slave that has identified itself as having an LCD.

Common code includes a timeout clock and switch debouncing functions. These are of such general use they are included as part of the core code. The clock allows multiple event timers to be running, with a callback to designated functions when timeouts occur. Event timers can be started, paused, resumed, cancelled, and repeated. The timer resolution as coded in the accompanying software is approximately one millisecond but may be easily changed.

The switch debouncing functions allow multiple switches, pushbuttons, etc. to be debounced, with an optional function callback when a switch changes state. The type of change (open to closed, closed to open, either) that triggers the callback is also specified. Debouncing is performed by reading each switch input once per timer interrupt (referred to as a tic; as coded, each millisecond). When the inputs are identical for a user specified number of timer tics the switch is considered stable. When the current stable state differs from the previous stable state, the callback function may be invoked if desired.

## DEMONSTRATION HARDWARE

For this article an almost trivial arrangement of one master and three slaves was wired on breadboards to demonstrate the protocol in action. Each slave uses identical hardware and runs identical software, contrary to the above discussion of customizing each slave to at least identify its features. More practical slave hardware and software, as well as more complex master operations, are briefly discussed later.

The prototyping shield boards used on both the master and slaves contain a switch and are provided with a small breadboard.

The switches were used for inputs, and LEDs with resistors were wired on the breadboards.

The master simply drives nine LEDs on which it shows the overall status. This includes the following:

- Initialization error/complete (two LEDs)
- Other error reported by slave (one LED)
- Last slave ID to which a command was sent (three LEDs)
- Tx/Rx toggles as bytes are sent or received by the master (two LEDs)
- Local 1-Hz heartbeat (one LED)

Each slave also drives nine LEDs, and their interpretation is:

- An LED that is toggled by the master in response to the slave's switch being pressed (called the User LED below)
- An LED that is controlled by the master (called the Master LED below).
- The slave's FSM's state (0000 to 1010). The accompanying documentation outlines the various states of the finite state machine. During reset, all state LEDs flash (four LEDs).
- Tx/Rx toggles as bytes are sent or received by this slave (two LEDs).
- Local 1-Hz heartbeat (one LED)

The slave's FSM will spend most of its time in a ready state with very brief excursions elsewhere, so at reasonable baud rates, the state changes would barely be discernible on the LEDs. For this reason, the display of the state changes is buffered and shown at a greatly reduced and visually perceptible rate of 300 ms each. As a consequence the state LEDs won't always be showing the actual real time state.

## FINITE STATE MACHINE

A finite state machine (FSM) is an abstract model that can be in one of a finite number of states. It will transition from its current state to a new state when triggered by an event or condition such as an interval of time passing or new input being received. The state the machine then enters depends on the current state and the nature of the triggering event or condition.

For example, consider the set of four traffic signals facing each direction at an intersection. Each has red, yellow, green, and possibly green left turn arrow lights. If it is a "smart" signal, there are sensors in the roadway to indicate when and where traffic is waiting. The signal will change its state (i.e., which lights are illuminated) appropriately as time passes or traffic conditions change. The sequence of lights is carefully controlled, and there are only a few different combinations of lights that will ever display. Such a scenario could be easily implemented as a finite state machine. An FSM can be defined by listing all of its states, and the triggering condition for each state transition.

## DEMONSTRATION SOFTWARE

The master and each slave have one LED dedicated to a 1-Hz heartbeat. This verifies that the microcontroller is alive and running and indirectly indicates its clock speed is correct. Not that different clock speeds aren't allowed, but each device's software should use appropriate parameters for its hardware so that its heartbeat LED flashes at 1 Hz.

At power-up, the master and slaves flash the remaining eight LEDs in sequence. This shows that all the LEDs are operational (on a breadboard, a miracle of its own), and that the software actually knows how they are physically wired. The sequenced flashing continues until the user pushes the master's switch to initiate initialization of the slaves. Initialization will succeed (green LED), fail



# PIC<sup>®</sup> MCU POWER TOOLS

- The First & Most Mature Compiler for PIC<sup>®</sup> MCUs
- Powerful, Professional & Easy to Use
- Over 90,000 Compilers Sold
- Built-in Functions for Rapid Development
- Optimized for the unique PIC<sup>®</sup> Architecture
- Easy Code Portability between devices

## Easy-to-Use Libraries & Built-in Functions:

- ✓ USB, TCP/IP, RS-232/485, MODBus, CANBus, LINBus
- ✓ SPI, I<sup>2</sup>C, SD Cards, ADC, DAC, Capacitive Touch
- ✓ Serial EEPROMs, Flash, Real-time Clocks, XTEA
- ✓ DCI, PSP, QEI, CRC, PID, Comparators, Watchdog
- ✓ PWM, CRCP, LCDs, RFID, Timers, Precision Delays
- ✓ Interrupts, Real Time Operating System (RTOS)
- ✓ 100+ Example Programs

## Power Wizards for Rapid Development:

- ✓ Both PIC Hardware and External Peripherals
- ✓ USB, TCP/IP, MODBUS, CANBUS, RS-232
- ✓ Bootloaders & Digital Filtering

## C-Aware IDE Features

- |                           |                                                    |
|---------------------------|----------------------------------------------------|
| - Project time accounting | - Source code formatter                            |
| - Source code history     | - Automatic documentation generator                |
| - File compare            | - Easy project packaging archiving & copy function |
| - Spell check components  | - Data file to C data converter                    |
| - Project notes           | - Includes file encryption support                 |
| - Flow chart drawing tool |                                                    |

## For your Insight:

- Graphical call tree
- Full memory allocation map
- Numerous code complexity metrics

## ID Explorer shows:

- Full relationship between functions, locals, globals, files & defines with full cross-referencing

## WARNING

Never use a Repurposed, Generic C Compiler for Serious PIC<sup>®</sup> Development!

## Unique CCS Power Features:

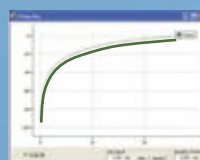
- ✓ Data Streaming uses an ICD to Collect Diagnostic Data, set Parameters, Perform Calibration & Production Testing
- ✓ Code Profiler for Real-time Timing & Execution Stats, verify Full Path Testing and Trace Call Paths
- ✓ Debugger Tightly Integrated with Compiler
- ✓ Context Sensitive Help and Interactive Helpers
- ✓ CCS Engineers Available for Hire on short notice
- ✓ Complete line of programmers/debuggers & Dev Kits

## NEW in 2015



Bluetooth<sup>®</sup> Library linking to tablets & Smartphones using a free generic app on the device. Control by a program on the PIC<sup>®</sup>

PIC24 Graphic Display/Touch Screen Library with PC Design Tool



dsPIC33 Audio Wizard with easy DSP Filter setup

Educational Classroom Resources



Download FREE 45 Day Demo: [www.ccsinfo.com/CCF815](http://www.ccsinfo.com/CCF815)



sales@ccsinfo.com  
(262) 522-6500 X35  
[www.ccsinfo.com](http://www.ccsinfo.com)



PIC<sup>®</sup> MCU is a registered trademark of Microchip Technology Inc.

## ABOUT THE AUTHOR

Chuck Baird started in programming by learning FORTRAN for a numerical analysis course in college. Today, he still enjoys bit fiddling and problem solving. Chuck has authored two books about programming Atmel AVR's. You may contact Chuck via private message at avrfreaks.net (username: zbaird).

softly (a brief period of flashing all the LEDs, then resumption of the LED flashing sequence so it can be tried again), or fail hard (flashing red LED for no slaves attached, or solid RED for some other failure).

After initialization, the master cycles through each slave repeatedly. It sends a data packet which turns the current slave's Master LED on, waits 800 ms, and then sends another packet to turn it off. After an 800-ms delay, it advances to the next slave. The effect is that Master LEDs on each slave will toggle in sequence. This demonstrates each slave receiving and handling unsolicited data from the master.

At any time the user may press a slave's switch, and the slave will request that the master toggle its Slave LED. This demonstrates the slave sending an REQ command, the master responding with a data packet, and the slave responding to the received data packet by toggling its LED.

After initialization, each push of the master's switch will send a RESET command to each slave in turn, then all of them, then repeat with slave #1. When a slave receives a RESET, it flashes all four of its blue LEDs a few times.

Pushing the master's RESET button should cause the slaves' LEDs to freeze (because there is no incoming data) and the master's LEDs to flash in sequence. Then pushing the master's switch should initialize all slaves and everything should be up and running normally. Cross your fingers.

## TO THE FUTURE, AND BEYOND

While this demonstration software does little more than show the protocol in action, it provides a framework for more interesting projects. The basic master and slave versions of the protocol and the various support functions are supplied, so all that is required is creating your specific application.

You will need to decide just what kinds of things the various microcontrollers need to talk about and the sorts of data they need to exchange. Also decide whether this protocol is fast enough for your application. As coded it uses a bit rate of 19.2 Kbps, or about 0.5 ms per character, so latency issues might become important. As mentioned, the data flows on


each segment independently, and most of the time a slave passes a character on as soon as it is fully received, resulting in a delay of 0.5 ms per slave. Of course, the addressed slave(s) may take a little longer processing the received command.

One desirable feature for applications would be the ability for slaves to exchange data. This could be accomplished by structuring the packet data to include an address (slave ID), perhaps as part of the first data byte. A slave would need to obtain the ID of its target from the master (using an agreed upon packet data interpretation), and then ask the master to relay the packet to the target slave (yet another agreed upon interpretation). Designating the first byte of each packet as routing and use information would be a reasonable approach to generalizing packet delivery. Another approach might be to utilize the unassigned packet type as a special packet containing routing information.

## DIG DEEPER

The value of this article and the serial protocol may lie more in education than in actually utilizing the specific code. An examination of both the code and the separate commentary file will reveal some methods for handling switch debouncing, virtual timers, callback functions, circular buffers, linked lists, and juggling a myriad of asynchronous events without tying yourself in knots. Admittedly, in places, the code looks complex; but hopefully the two commentary files and the comments within the code will provide you with enough guidance to successfully understand it.

C is a language that can be subjected to infinite obfuscation, and many programmers delight in making the simple needlessly complex. I have attempted to write straightforward, easily deciphered code. Often there are more elegant ways things could have been (and perhaps should have been) done, but hopefully I have erred on the side of comprehensibility. There are cases of global variables that should have limited scope, using variables where they are not needed, and so on, but fine tuning the code was not my goal. Additionally, the bottom line on many minor stylistic coding points is, the compiler's optimizer will do what is right anyway.

If you are interested in digging deeper, the first step in writing code for any microcontroller is to obtain a copy of its datasheet. It contains all the details about the device's memory layout, register usage, care and feeding of each peripheral unit, and so on. Atmel's ATmega328p datasheet is available as a free download (Google is your friend), although it is 600 pages long. All is revealed to he or she who seeks. 



circuitcellar.com/ccmaterials

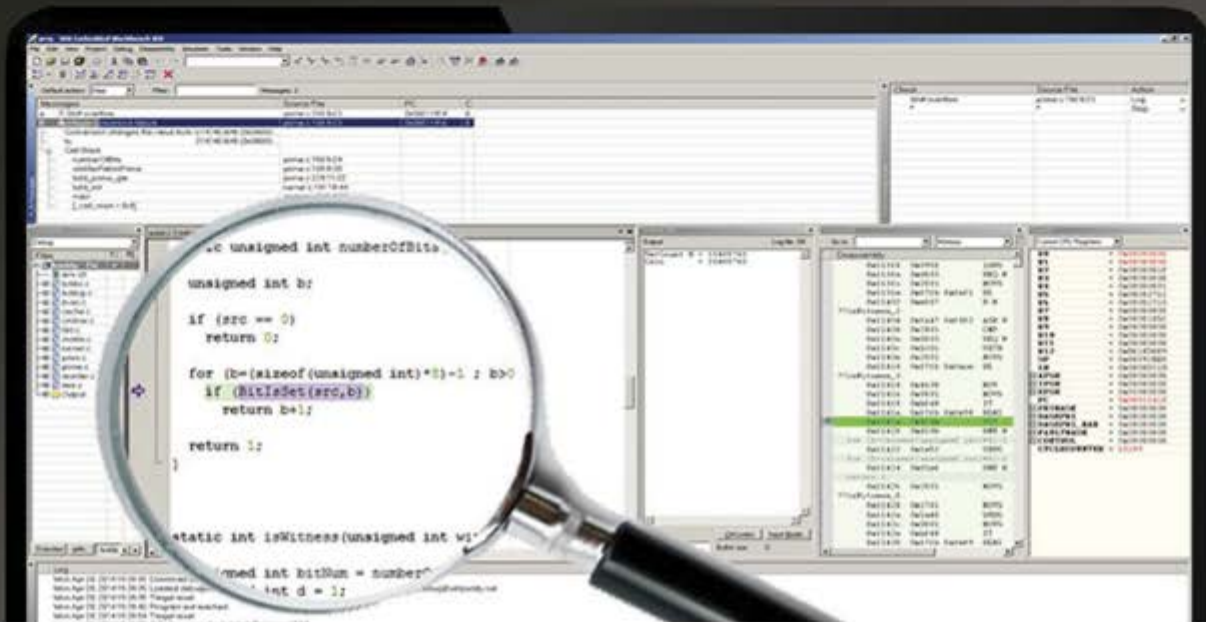
## SOURCES

ATmega328p Microcontroller  
Atmel Corp. | [www.atmel.com](http://www.atmel.com)

Arduino Pro 328 5 V/16 MHz  
SparkFun | [www.sparkfun.com](http://www.sparkfun.com)

# EXPLORE C-RUN FOR ARM

## Runtime Analysis Simplified



C-RUN is a high-performance runtime analysis add-on product, fully integrated with world-leading C/C++ compiler and debugger tool suite IAR Embedded Workbench.

C-RUN performs runtime analysis by monitoring application execution directly within the development environment. The tight integration with IAR Embedded Workbench improves development workflow and provides each developer with access to runtime analysis that is easy-to-use.

[www.iar.com/crun](http://www.iar.com/crun)





# Sound Ecology and Acoustic Health (Part 2)

## Record and Play Audio .3GPP Files

FEATURES

Last month, Adrien and Mike described how to start a basic Android app and their plans to add custom extensions for identification. This month, Adrien and Mike tackle the topics of recording and playing back audio .3GPP files.

*By Adrien Gaspard and Mike Smith (Canada)*

**I**n the first part of this article series, we light-heartily discussed a supposed backyard BBQ discussion between neighbors about urban noise nuisances. Unfortunately, noise nuisances are real in some of our local Calgary communities, and we are looking for some simple, inexpensive approaches to help people investigate and reduce the problem.

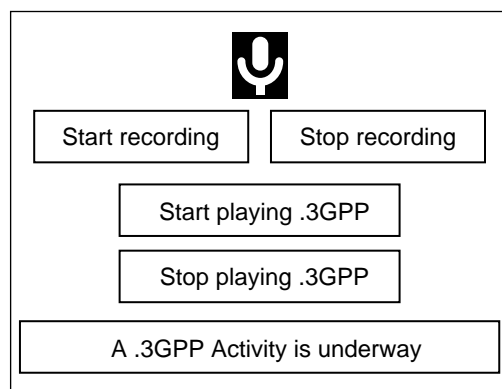
We demonstrated the first steps of our solution: the development of an Android project with basic code to generate a main screen with a button that generated a welcome screen when pressed. We called it WAT\_AN\_APP—meaning, we were able to develop it Without Any Teenager Assistance Being Necessary. In this article, we want

to extend our basic WAT\_AN\_APP project to recording and playing back audio .3GPP files. This will allow us to record any physical noises present that are less easily heard by others in your house or need more study as they are less noticeable during the day when hidden under traffic noise.

In this article, we want to take a more adult approach. We use a JEAC process that uses Just Enough Additional Code to make the new recording activity work.

### QUICK RECAP

**Listing 1** provides the key elements of the main activity java file. Applying the JEAC philosophy, we added enough code to pop up a screen with a welcome message and a button. Pressing the button activated the audio record and playback “AudioRecordPlayback” activity (Line 20). This activity used the layout described in the activity\_audio\_record\_playback.xml file to activate a TextView object to print a message “DUMMY NEW ACTIVITY SCREEN” (see **Listing 2**). Please note that the main activity’s layout file, activity\_main.xml, is identical to the code described in Listing 2 in first article of this series. In this article we are going to extent this dummy activity so that we can record and play back audio .3GPP files, the first step towards doing some real signal processing on audio signals.



**FIGURE 1**

Our planned .3GPP file control screen

## JEAC BUTTONS FOR .3GPP CONTROL

The new `activity_audio_record_playback.xml` layout (see **Listing 3**) shows how we can use a lot of our knowledge gained from the previous article to generate an audio menu screen (see **Figure 1**). A few new custom attributes help to lay out the four buttons controlling recording and playback. The `alignParentStart` command (line 434) makes the leading edge of the START RECORDING button match that of the STOP RECORDING button. The attributes `alignParentLeft` and `alignParentRight` place these buttons one next to another. We have used a new `layout_below` attribute (line 462) to put one button below another button. If you want to have more information concerning layout attributes, visit [developer.android.com/reference/android](http://developer.android.com/reference/android).

Since it just takes a few lines, we decided to add a microphone picture to the top of the screen. The new widget, `ImageView`, is used to load and display images from the "WAT\_AN\_APP\res\drawable-xxx" folders that store images with different resolutions. Refer to the "Quick Help Guide" sidebar in this article to learn how to add an "ic\_action\_mic.png" microphone picture in the "WAT\_AN\_APP\res\drawable-mdpi" folder.

As for the text and buttons, the `ImageView` widget has to be given an ID, "microphone" (line 421). We set the placement and source of the picture to display using Lines 422 and 423.

## ADD JEAC AUDIO ACTIVITY

Android offers a simple `MediaRecorder` class which provides a "blackbox" designed to capture, save and play back all types of media, including pictures, videos, and audio. We have followed two online examples to build an audio recorder/player: [developer.android.com/guide/topics/media/audio-capture.html](http://developer.android.com/guide/topics/media/audio-capture.html) and [tutorialspoint.com/android/android\\_audio\\_capture.htm](http://tutorialspoint.com/android/android_audio_capture.htm).

**Listing 4** and **Listing 5** show the `AudioRecordPlayback` activity. We start by defining our `MediaRecorder` `myRecorder`, the `MediaPlayer` `myPlayer`, the file name `outputFileName` for our recording and the four buttons (lines 316 to 319). We then code the `onCreate()` method to initialize our activity. Line 322 sets the user interface (UI) from the layout resource (see **Listing 3**) using `setContentView()`. Finally, we detail the four buttons we need to interact with the application and set the `start_recording` button as active (line 327).

Pressing the `start_recording` button will activate the `start_recording` public method (see **Listing 5**, line 340).

```
1. package com.wat_an_app; // MainActivity.java
2. ... // SAME AS Article 1, Listing 1, Lines 2 to 5

    // Cause display of MainActivity screen layout
10. public class MainActivity extends Activity{
11. ... // SAME AS Article 1, Listing 1, Lines 11 to 15

    // Display AudioRecordPlayback screen layout
20. public void AudioRecordPlayback(View v){
21. ... // SAME AS Article 1, Listing 1, Lines 21 to 24
```

### LISTING 1

Article 1's key code from `MainActivity.java` in the `WAT_AN_APP\src\` folder

```
    <!--Used by AudioRecordPlayback -->
400. <RelativeLayout
401. xmlns:android=http://schemas.android.com/apk/res/
    android
402. ... <!-- SAME AS Article 1, Listing 5, Lines 402
    to 405 -->

410.     <TextView
411.         android:id="@+id/audio_record_playback_text"
412.         ...<-- SAME AS Article 1 Listing 5, Lines 410
    to 417-->

499. </RelativeLayout>
```

### LISTING 2

Article 1's `AudioRecordPlayback` activity layout from `activity_audio_record.playback.xml` file in the `WAT_AN_APP\res\layout` folder

```
<!--Used by AudioRecordPlayback -->
400. <RelativeLayout
401. xmlns:android=http://schemas.android.com/apk/res/
    android
402. ... <!--SAME AS Article 1 Listing 4, Lines 402 to 405-->

410. <-- Delete Article 1 Listing 5 Lines 410 - 417 -->

    <!-- Small microphone image -->
420. <ImageView
421.     android:id="@+id/microphone"
422.     android:layout_marginTop="150dp"
423.     android:src="@drawable/ic_action_mic"
424.     tools:ignore="ContentDescription"
425.     android:layout_width="wrap_content"
426.     android:layout_height="wrap_content"
427.     android:layout_centerHorizontal="true"
428. />

    <!-- Start recording button -->
430. <Button
431.     android:id="@+id/button_start_rec"           (continued)
```

### LISTING 3

This updated `activity_audio_record_playback.xml` layout file (`WAT_AN_APP\res\layout` folder) generates the four upper .wav control buttons and lower toast screen shown in **Figure 1**.

```

432.    android:onClick="start_recording"
433.    android:text="@string/start_recording"
434.    android:layout_alignParentStart="true"
435.    android:layout_alignParentLeft="true"
436.    android:layout_width="wrap_content"
437.    android:layout_height="wrap_content"
438.    android:layout_centerHorizontal="true"
439.    android:layout_centerVertical="true"
440. />

    <!-- Stop recording button -->
450.    <Button
451.        android:id="@+id/button_stop_rec"
452.        android:onClick="stop_recording"
453.        android:text="@string/stop_recording"
454.        android:layout_alignParentRight="true"
455.        android:layout_alignParentEnd="true"
456.        <!-- COPY Lines 436 to 439 -->
457.    />

    <!-- Start playback button -->
460.    <Button
461.        android:id="@+id/button_start_playback"
462.        android:layout_below="@+id/button_stop_rec"
463.        android:onClick="start_playback"
464.        android:text="@string/start_playback"
465.        <!-- COPY Lines 436 to 439 -->
466.    />

    <!-- Stop playback button -->
470.    <Button
471.        android:id="@+id/button_stop_playback"
472.        android:layout_below="@+id/button_start_
473.        playback"
474.        android:onClick="stop_playback"
475.        android:text="@string/stop_playback"
476.    />

499. </RelativeLayout>

```

**LISTING 3 (CONTINUED)**

This updated `activity_audio_record_playback.xml` layout file (`WAT_AN_APP\res\layout` folder) generates the four upper .wav control buttons and lower toast screen shown in Figure 1.

This is a busy method which sets up the recording file path as well as the name of the recording `myrecording.3gpp` in line 341. The `MediaRecorder` is initialized (lines 342 to 346) and starts a recording (line 349). The method finishes with a flourish by disabling and ghosting the `start_recording` button, activating the `stop_recording` button, and issues a toast, Android message, on the screen to show the user that a recording has started (lines 352 to 354).

Listing 5 also shows the similar format of the other audio control methods: `stop_recording` (lines 360 to 366), `start_`

`playback` (lines 370 to 377), and `stop_playback` (lines 380 to 386). They each manipulate the `MediaPlayer` and enable the next method in the audio control stream before turning themselves off.

The fact that these methods turn off themselves reminded Mike of an early electronic toy he used to have in a much more simple time. When the switch on the top of the toy was turned on, the toy's box lid opened and a hand came out and pushed the switch to turn the toy off.

Once the recording has been stopped (see Listing 5, line 361), it is important to issue a `release MediaRecorder` command (line 362). This frees up the audio hardware and other system resources which are all shared across the different applications running on the phone.

We can start playing this recorded file by calling the `start_playback` method which reinitializes the `MediaPlayer` in a play-back mode (line 370). We identify the recorded file, stored in `outputFileName`, and then prepare the player to begin playing data. Pressing the stop playback button again releases the `MediaPlayer` resources back to the system and displays a "Stop Playing Back" message on the screen.

Just before you hit the compile button for the last time, remember we have been using a lot of strings. As we noted in the first part of this series, avoid the compiler warning messages by adding preset strings to the `strings.xml` file in the "WAT\_AN\_APP\res\values" folder (see Listing 6).

**FORGIVENESS & PERMISSIONS**

In the everyday world there is a saying, "Sometimes you get further ahead by asking for forgiveness rather than asking for permission." In our JEAC, world there is an equivalent saying, "Sometimes your project finishes faster if you set permissions to allow a few things rather than writing more code to allow everything."

For example, do you want to hunt ghosts or write the code needed to ensure your app can handle you switching from portrait to landscape modes? Currently, an event will be triggered that will restart the audio activity if you rotate the screen while recording or playing. That risks the `MediaRecorder` or `MediaPlayer` not being properly released and reinitialized and causing the activity to crash at that point. Solve this by disabling the auto-rotate permissions in your phone's settings menu.

Other potential issues can also be prevented, rather than requiring coding, by setting permissions in the `AndroidManifest.xml` file in the `WAT_AN_APP` project's root directory. For example, line 3011 in Listing 7



Authorized global distributor of the **NEWEST** electronic components.

[mouser.com/new](https://mouser.com/new)

You can't invent the future with products from the past.

Design with the **NEWEST PRODUCTS** ahead of their time.



Mouser and Mouser Electronics are registered trademarks of Mouser Electronics, Inc. Other products, logos, and company names mentioned herein, may be trademarks of their respective owners.



The Newest Products for Your Newest Designs®

**LISTING 4**

The prologue of the AudioRecordPlayback.java file (WAT\_AN\_APP\src\ folder) sets up the user interface. The onCreate() method enables the Start Recording button (line 327). The other methods in this class detailed in Listing 5.

```
// Replace existing code from Article 1 Listing 4 Lines 300 to 302
300. package com.wat_an_app;
301. import android.widget.Toast;
302. import android.os.Bundle;
303. import android.os.Environment;
304. import android.widget.Button;
305. import android.view.View;
306. import android.support.v7.app.ActionBarActivity;
307. import android.media.MediaPlayer;
308. import android.media.MediaRecorder;
309. import java.io.IOException;
310. import com.wat_an_app.R;

315. public class AudioRecordPlayback
                                extends ActionBarActivity {
316. private MediaRecorder myRecorder;
317. public MediaPlayer myPlayer = null;
318. private String outputFileName = null;
319. private Button button_start_recording,
    button_stop_recording, button_start_playback,
    button_stop_playback;

320. @Override
    protected void onCreate(Bundle savedInstanceState) {
321.     super.onCreate(savedInstanceState);
322.     setContentView(R.layout.activity_audio_record_playback);
323.     button_start_recording
        = (Button)findViewById (R.id.button_start_rec);
324.     button_stop_recording
        = (Button)findViewById (R.id.button_stop_rec);
325.     button_start_playback
        = (Button)findViewById (R.id.button_start_playback);
326.     button_stop_playback
        = (Button)findViewById(R.id.button_stop_playback);

327.     button_start_recording.setEnabled(true);
328.     button_stop_recording.setEnabled(false);
329.     button_start_playback.setEnabled(false);
330.     button_stop_playback.setEnabled(false);
331. }

    // public void start_recording(View view)
    //Listing 3B Lines 340 to 355

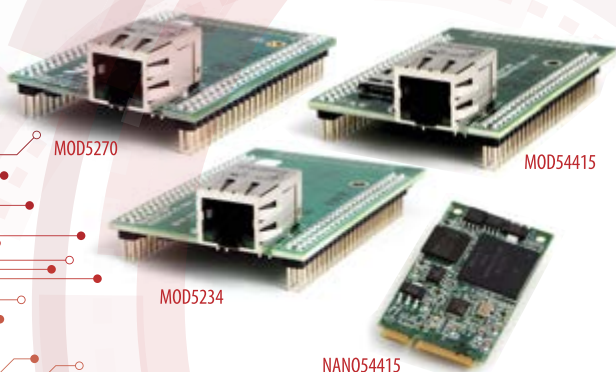
    // public void stop_recording(View view)
    //Listing 3B Lines 360 to 366

    // public void start_playback(View view) throws
    // IllegalArgumentException, SecurityException,
    // IllegalStateException, IOException
    //Listing 3B Lines 370 to 377

    // public void stop_playback(View view)
    //Listing 3B Lines 380 to 386

399. } // End class AudioRecordPlayBack
```

# Ethernet Core Modules with High-Performance Connectivity Options



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

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



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



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



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

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

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

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

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



```

340. public void start_recording(View view) {
341.     outputFileName =
        Environment.getExternalStorageDirectory().
            getAbsolutePath()+ "/myrecording.3gpp";
342.     myRecorder = new MediaRecorder();
343.     myRecorder.setAudioSource(MediaRecorder.
        AudioSource.MIC);
344.     myRecorder.setOutputFormat(MediaRecorder.
        OutputFormat.THREE_GPP);
345.     myRecorder.setAudioEncoder(MediaRecorder.
        OutputFormat.AMR_NB);
346.     myRecorder.setOutputFile(outputFileName);
347.     try {
348.         myRecorder.prepare();
349.         myRecorder.start();
350.     } catch (IllegalStateException e) {e.printStackTrace();}
351.     catch (IOException e) {e.printStackTrace();}
352.     button_start_recording.setEnabled(false);
353.     button_stop_recording.setEnabled(true);
354.     Toast.makeText(getApplicationContext(),
        "Start Recording", Toast.LENGTH_SHORT).show();
355. }

360. public void stop_recording(View view){
361.     myRecorder.stop();
362.     myRecorder.release();
363.     myRecorder = null;
364.     button_stop_recording.setEnabled(false);
365.     button_start_playback.setEnabled(true);
366. }

370. public void start_playback(View view)
    throws
        IllegalArgumentException, SecurityException,
        IllegalStateException, IOException{
371.     myPlayer = new MediaPlayer();
372.     myPlayer.setDataSource(outputFileName);
373.     myPlayer.prepare();
374.     myPlayer.start();
375.     button_start_playback.setEnabled(false);
376.     button_stop_playback.setEnabled(true);
377. }

380. public void stop_playback(View view){
381.     button_stop_playback.setEnabled(false);
382.     myPlayer.release();
383.     myPlayer = null;
384.     Toast.makeText(getApplicationContext(),
        "Stop Playing Back", Toast.LENGTH_SHORT)
        .show();
385.     button_start_recording.setEnabled(true);
386. }

399. }

```

**LISTING 5**

Details of the Recording and Playback methods from the AudioRecordPlayback.java file (WAT\_AN\_APP\src\ folder)

**SAVING COST=TIME with readily available  
FPGA boards**

## XILINX FPGA Board (Kintex-7 Series)

Kintex-7 FBG484 FPGA board

### XCM-022 series

Kintex-7 SW  
MRAM LED  
DDR3 I/O:100  
RocketIO SIF40



- FPGA : XC7K70T-1FBG484C  
XC7K160T-1FBG484C
- Board size : 86 x 54 [mm]

RoHS compliant

Kintex-7 FBG484 FPGA board

### XCM-112 series

Kintex-7 SW  
RocketIO LED  
DDR3 I/O:128



- FPGA : XC7K70T-1FBG484C  
XC7K160T-1FBG484C
- Board size : 43 x 54 [mm]

RoHS compliant

Kintex-7 USB-FPGA board

### EDX-008

Kintex-7 SW  
MRAM LED  
DDR3 I/O:100  
USB Config.  
USB Comm. HI-SPEED



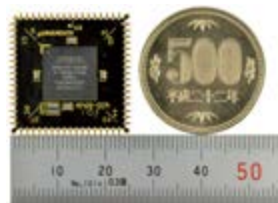
- FPGA : XC7K70T-1FBG484C  
XC7K160T-1FBG484C
- Board size : 86 x 54 [mm]

RoHS compliant

## PLCC68 Series Stamp size FPGA/CPLD Module

**Easy and Quickly Mountable on 68-pin IC socket**

- ☐ 50 I/Os(External clock inputs are available)
- ☐ Separated supply-inputs: Core, I/O drivers
- ☐ 3.3V single power supply operation
- ☐ JTAG signal
- ☐ Voltage converters for auxiliary power supply
- ☐ All PLCC68 series have common pin assignment
- ☐ Very small size (25.3 x 25.3 [mm])
- ☐ Mountable on IC socket



## ALTERA Series

Cyclone V PLCC68 FPGA Module

### AP68-06 Series

- FPGA : 5CEBA4U15C8N
- FRAM
- 3.3 V single power supply operation
- On-board oscillator, 50MHz
- Configuration Device
- 8 Layer PCB



RoHS compliant

Cyclone III PLCC68 FPGA Module

### AP68-03 Series

- FPGA : EP3C10U256C8N
- 3.3V single power supply operation
- On-board oscillator, 50MHz
- Configuration Device
- 8 Layer PCB
- Board size : 25.3 x 25.3[mm]

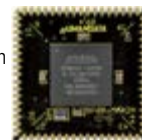


RoHS compliant

MAX II PLCC68 CPLD Module

### AP68-01 Series

- CPLD : EPM240F100C5N  
EPM570F100C5N
- 3.3V single power supply operation
- Board size : 25.3 x 25.3[mm]
- 4 Layer PCB



RoHS compliant

## XILINX Series

Spartan-6 PLCC68 FPGA Module

### XP68-03 Series

- FPGA : XC6SLX45-2CSG324C
- 3.3V single power supply operation
- on-board oscillator, 50MHz
- Configuration Device
- Board size : 25.3 x 25.3[mm]
- 6 Layer PCB



RoHS compliant

Spartan-3AN PLCC68 FPGA Module

### XP68-02 Series

- FPGA : XC3S200AN-4FTG256C
- 3.3V single power supply operation
- on-board oscillator, 50MHz
- Board size : 25.3 x 25.3[mm]
- 6 Layer PCB



RoHS compliant

Spartan-6 PLCC68 FPGA Module

### XP68-01 Series

- FPGA : XC6SLX16-2CSG225C
- 3.3V single power supply operation
- on-board oscillator, 50MHz
- Configuration Device
- Board size : 25.3 x 25.3[mm]
- 6 Layer PCB



RoHS compliant

We also have many other products  
All stocked items are ready to be  
shipped immediately

**Humandata Products are in stock@amazon!!**



**HuMANDATA LTD.**

TEL : +81-72-620-2002 (Japanese) FAX : +81-72-620-2003 (Japanese/English)  
E-Mail : s2@hdl.co.jp URL : <http://www2.hdl.co.jp/en/>



See all our products, A/D D/A conversion board,  
boards with USB chip from FTDI and accessories at :

**[www.hdl.co.jp/CC1508](http://www.hdl.co.jp/CC1508)**

```

200. <?xml version="1.0" encoding="utf-8"?>
201. <resources>
202. <!-- SAME AS Article 1 Listing 3, Lines 205 to
      214 -->

220. <!-- String required for the second part of the
      record and playback a sound -->
221. <string name="start_recording">
      Start recording</string>
222. <string name="stop_recording">
      Stop recording</string>
223. <string name="start_playback">
      Start playback .3GPP</string>
224. <string name="stop_playback">
      Stop playback .3GPP</string>
249. </resources>

```

**LISTING 6**

To avoid compiler warning messages, new preset string values must be set in strings.xml (WAT\_AN\_APP\res\values folder).

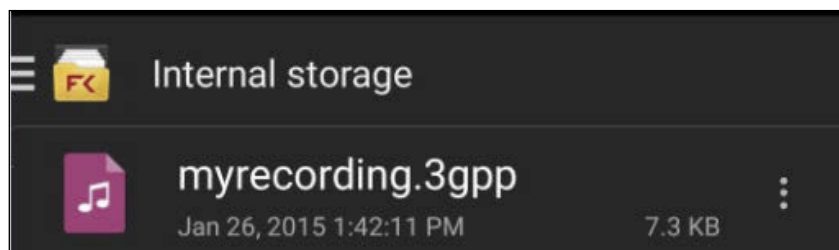
stops the app from going to sleep while we are recording because switching the sleeping screen off can cause the current audio record/playback operation to stop.

Currently, the app is like a well-trained dog sitting at your feet with ears actively waiting for your command to GO AND PLAY! So give our application the right to listen (record audio) and to remember what we have told it (read and write on external storage), lines 3008 to 3010. Please note that the minimum SDK version the application can run on has been set to "14" (line 3006), compared to "9" in the first article. Make sure that it has been set to "14" in your AndroidManifest.xml file as well, avoiding compatibility issue with the code we have implemented. On one of our phones, a Nexus 5 from Google, there is no physical slot for adding an external micro SD memory card. The recorded audio file is then automatically saved into the phone's internal storage memory (see **Figure 2**).

**CAN A (LOG)CAT SEE GHOSTS?**

In a much more far away time and civilization, cats were regarded as gods, supposedly for their unique perception of the spirit world. When developing this app, we started to appreciate that the LogCat tool has a unique perception of your Android system. LogCat can be used to view and filter logs from applications and portions of the Android system. A crash, error or warning details for an application are outputted in the LogCat window.

A system crash message is easily interpreted, but that is not so for the some warning and error messages. As they say on the TV, "For that there is stackoverflow.com," with proposed solutions from the wide Android programming community. Reducing coding time with a JEAC approach means that the code is not "commercial release grade." So when debugging you should expect to have to click through (ignore) some LogCat error messages. For example, if your phone runs on the Lollipop OS then LogCat complains that we

**FIGURE 2**

Recorded file physically present in the phone internal storage memory

**ABOUT THE AUTHORS**

Adrien Gaspard (gasp.adrien@gmail.com) earned a Masters of Engineering from CPE Lyon, France, in February 2015. He tackled his final practicum as an exchange student in Electrical and Computer Engineering at the University of Calgary. He undertook self-directed term projects directed towards the possible use of noise cancelling to solve the community noise problem in Calgary community of Ranchlands. Adrien intends to focus his career in the fields of embedded systems and wireless telecommunications.

Mike Smith (Mike.Smith@ucalgary.ca) has been contributing to *Circuit Cellar* since the 1980s. He is a professor of Computer Engineering at the University of Calgary, Canada. Mike's main interests are in developing new biomedical engineering algorithms and moving them onto multi-core and multiple-processor embedded systems in a systematic and reliable fashion. He is a recent convert to the application of agile methodologies in the embedded environment. Mike has been an Analog Devices University Ambassador since 2001.



# Verilog HDL

## With the right tools

such as this book,

**designing a microprocessor  
can be easy.**

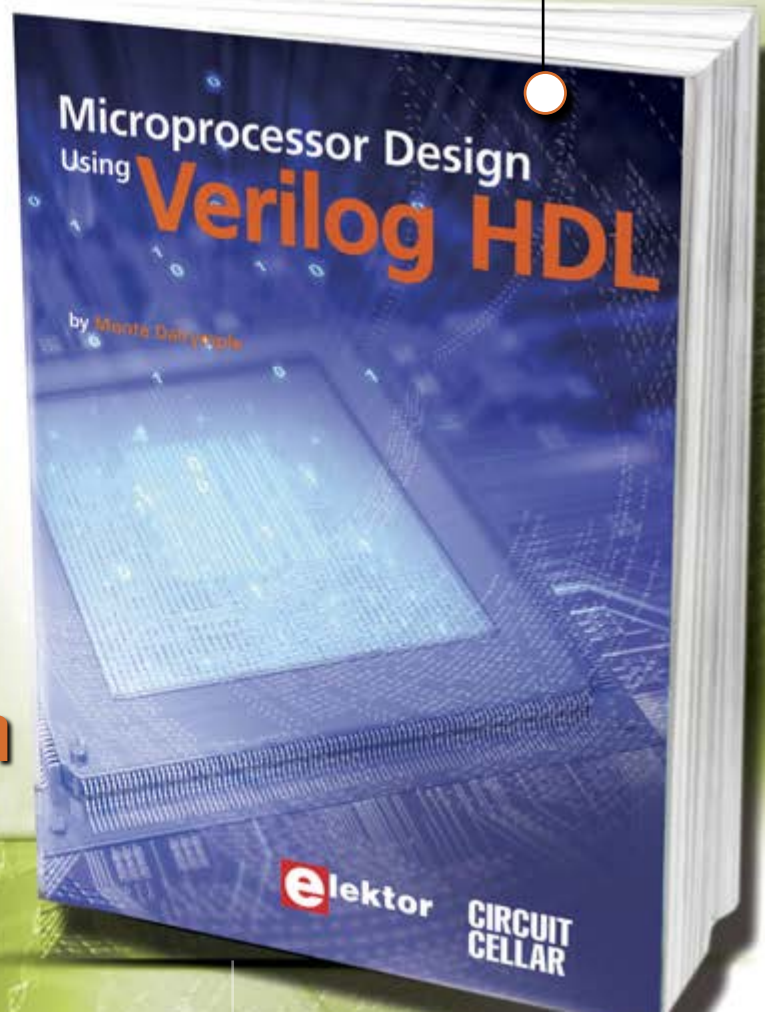
Okay, maybe not easy, but certainly less complicated. Monte Dalrymple has taken his years of experience designing embedded architecture and microprocessors and compiled his knowledge into one comprehensive guide to processor design in the real world.

Microprocessor Design Using Verilog HDL will provide you with information about:

- Verilog HDL Review
- Verilog Coding Style
- Design Work
- Microarchitecture
- Writing in Verilog
- Debugging, Verification, and Testing
- Post Simulation and more!

Monte demonstrates how Verilog hardware description language (HDL) enables you to depict, simulate, and synthesize an electronic design so you can reduce your workload and increase productivity.

**cc-webshop.com**



“should have subtitle controller already set” every time the MediaPlayer starts playing the recorded sound.

If you have fast enough reflexes it is possible to make LogCat issue an error message of the form “Get Occurred on inactive InputConnection.” From Stackoverflow I have learnt that InputConnection is the communication channel interface from an

input method back to the application. You can get other error messages over this channel if some part of the app like a Toast is taking too long to respond.

Stackoverflow.com offers many solutions for this sort of problem. However, we would rather click through and take the JEAC amendment: “Enough coding already. Let’s go ghost hunting.”

```

3000. <?xml version="1.0" encoding="utf-8"?>
3001. <manifest xmlns:android=
           "http://schemas.android.com/apk/res/android"
3002. package="com.wat_an_app"
3003. android:versionCode="1"
3004. android:versionName="1.0" >

           <!--Check lines 3005 and 3007 and update if necessary-->
3005. <uses-sdk
3006.   android:minSdkVersion="14"
3007.   android:targetSdkVersion="21" />

           <!--Manually add lines 3008 to 3011-->
3008. <uses-permission android:name=
           "android.permission.RECORD_AUDIO" />
3009. <uses-permission android:name=
           "android.permission.WRITE_EXTERNAL_STORAGE" />
3010. <uses-permission android:name=
           "android.permission.READ_EXTERNAL_STORAGE" />
3011. <uses-permission android:name=
           "android.permission.WAKE_LOCK" />

           <!--Automatically added -->
3012. <application
3013.   android:allowBackup="true"
3014.   android:icon="@drawable/ic_launcher"
3015.   android:label="@string/app_name"
3016.   android:theme="@style/AppTheme" >

3017. <activity
3018.   android:name=".MainActivity"
3019.   android:label="@string/app_name" >

3020.   <intent-filter>
3021.     <action android:name="android.intent.action.MAIN" />
3022.     <category
3023.       android:name="android.intent.category.LAUNCHER" />
3024.   </intent-filter>

3025. </activity>

3026. <activity
3027.   android:name=".AudioRecordPlayback"
3028.   android:label="@string/title_activity_audio_record_
3029.   playback" >
3030. </activity>

3030. </application>


3031. </manifest>

```

#### LISTING 7

AndroidManifest.xml from the WAT\_ AN\_APP project's root directory

## READY FOR SOME 4H TIME

We can now record a noise in neighborhood and play the sound in a quieter environment where we can examine it in more detail. We are also ready to put in some 4H time—Happy Hunting Haunting Hours. Watch out for our next article in which we give up our qualitative ghost hunting approach and go in for real boasting rights. With a quantitative ghost hunting app, we will be able to prove that there are more ghosts in our neighborhood than anywhere else in the world! 

## QUICK HELP GUIDE

If you want some additional coding help, call Android Busters. This is who we looked up and called (online) when we needed a tutorial about how to create an audio recorder/playback, as well as when we needed help to bust the hard, and sometimes simple, Android problems that we found “haunting” us.

- Tutorial at Android Developer (Audio Capture): [developer.android.com/guide/topics/media/audio-capture.html](http://developer.android.com/guide/topics/media/audio-capture.html)
- Tutorial at Tutorials Point (Audio Record/Playback): [http://tutorialspoint.com/android/android\\_audio\\_capture.htm](http://tutorialspoint.com/android/android_audio_capture.htm)
- Android Community help at <http://stackoverflow.com/questions/tagged/android>

There are a whole load of icons to facilitate our app design and implementation at <https://www.google.com/design/icons/index.html>. Scroll down the web page to the AV section where there is a mic icon. You will be able to download the mic icon as part of a `ic_mic_black_24dp.zip` file. After unzipping, copy the `ic_mic_black_24dp.png` file in directory `android/drawable-mdpi` into the “`WAT_AN_APP\res\drawable-mdpi\`” folder in Eclipse and rename as `ic_action_mic.png`. This allows us to define the mic icon source using “`src="@drawable/ic_action_mic`” in the `ImageView` tag Line 423 in Listing 3.

If you want to be able boast to your neighbours that you have captured a ghost sound, then you will need to play your captured .3GPP file more than just one once. To do that, add `button_start_playback.setEnabled(true);` after Line 381 in Listing 5.

## MEASUREMENT COMPUTING

# High-Speed Conditioned Measurements *with Channel-to-Channel Isolation*

Easy to Use • Easy to Integrate • Easy to Support

## SC-1608 Series Highlights

- Signal conditioning for **Thermocouple**, **RTD**, **Strain Gage**, **Frequency**, **Direct Voltage**, and **Current**
- USB or Ethernet interface available
- Up to 500 kS/s sampling
- 16-bit resolution
- Channel-to-channel isolation
- Isolated analog output available
- Isolated digital input/output



From Only  
**\$999**

The new **SC-1608 Series** from Measurement Computing, uses per-channel analog and digital conditioning modules paired with a USB or Ethernet high-speed DAQ device for a flexible and cost-effective solution for connecting directly to sensors.

[mccdaq.com/SC1608](http://mccdaq.com/SC1608)

 **MEASUREMENT  
COMPUTING™**

Contact us  
**1.800.234.4232**

©2015 Measurement Computing Corporation, 10 Commerce Way, Norton, MA 02766 • [info@mccdaq.com](mailto:info@mccdaq.com)



# 'Net-Connected Security Network (Part 2)

FEATURES

## Image Processing, Messaging, and Movement Monitoring

Last month, Claudiu and Liviu presented their security system's hardware and a node's basic firmware. Here they explain how the network functions and cover the topics of image processing inside each node, movement detection, messaging, and monitoring.

By Claudiu Chiculita & Liviu Ene (Romania)



**PHOTO 1**

Bottom and top view of the first hardware version

In the first part of this article series, we introduced our compact security system, which we power through passive power over Ethernet (PoE) and includes a servomotor for rotation and a video camera (see **Photo 1**). In this article, we'll explain how each device in the system independently uses the video camera to acquire and process images in order to detect movement. We also detail how multiple devices connected in a network can cooperate by exchanging messages to ensure a better view from multiple angles (see **Figure 1**).

We use a COMedia C328 serial camera, which is easy to connect to a microcontroller's UART, as well as to a PC via an FTDI USB-to-UART 3.3-V adapter. All camera acquisition images and processing routines were developed first on the PC and then included in the microcontroller project, so many source files are platform independent. From the camera, JPEG images are captured at 640 × 480 resolution (usually under 20 KB) to be stored on an SD card or transmitted to the PC application for display. Raw, uncompressed images are also captured at the minimum

available resolution (80 × 60) and take 4,800 bytes in size for grayscale content.

### NETPBM FORMAT

While implementing image processing routines, it is necessary to constantly examine the input and the processed image. Even if development was done first on a PC, it is useful to have an image format that's easy to examine and modify. Here comes the NetPBM format to the rescue! The format defines three types of files (with different extension and headers) for black and white, grayscale, and color images, each of which has an ASCII or binary representation. The ASCII format is great for manually constructing and examining images, and the grayscale binary is perfect for working with images that are stored with 8 bits per pixel, as is the case for the basic raw image returned by the camera.

**Figure 2a** shows an example of a 3 × 3 image with a darker cross on a light background. In **Figure 2b** and **Figure 2c**, the content of the .pgm file is listed for ASCII and binary modes. (The gray levels were chosen so they can be easily displayed: chr(122)='z' and chr(33)='!'.)

Saving an image from the C328 camera is just a matter of first writing the header "P5 80 60 255"—where "P5" represents the binary Portable GrayMap type, "80 60" represents the resolution, and "255" is the maxim pixel value—followed by the raw data, and saving with the "pgm" extension. That is all it takes to be able to see the raw format of the C328 camera with an image viewer like IrfanView.

## IMAGE PROCESSING

In general image processing requires significant resources (in terms of computing power and also memory). In order to do this inside a microcontroller with limited resources, it requires working with smaller resolution images and also doing algorithms that are not very complex. In this case, the steps taken for the image processing are as follows.

First, a frame is acquired in raw mode, 8-bit grayscale, 1 byte per pixel, and stored in a RAM buffer. At a resolution of  $60 \times 80$ , the image size is 4,700 bytes. A table showing the different stages of image processing is posted on the *Circuit Cellar* FTP site.

Next, the previous image is retrieved. Initially, it was planned to store the previous frame externally in the WIZnet 5500's RAM, but because there is still space in Microchip Technology PIC32MX250F128 microcontroller's RAM, that feature is not used. Changes between the current and previous image are detected by subtracting them. The resulting matrix will have values close to zero (black) in places where nothing changed and higher values (light gray) where there was movement.

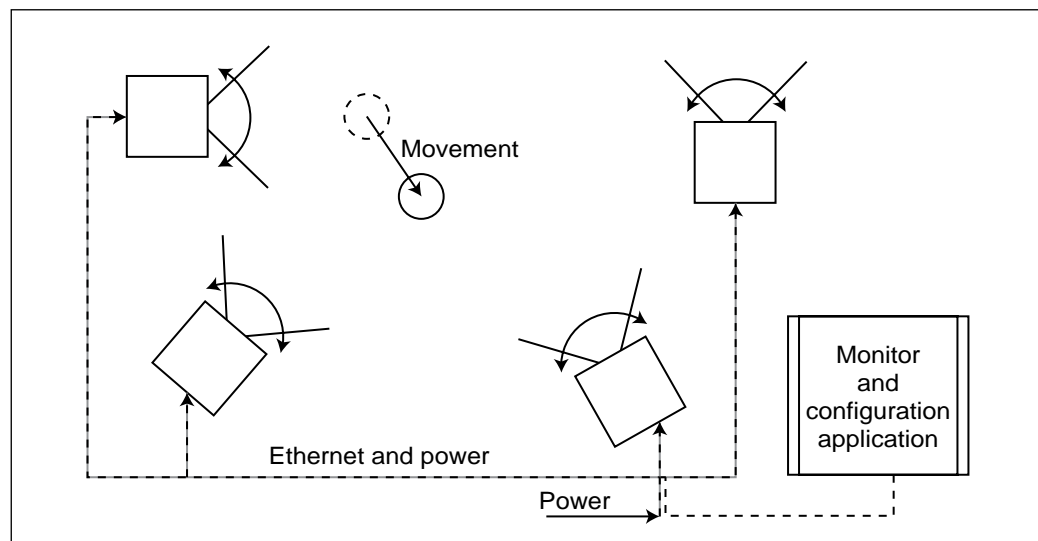
The next step is image segmentation. From an image with levels of gray that represent the amount of change, a binary image (only black and white) is obtained by comparing each value with a segmentation threshold.



**PHOTO 1**

The system mounted for testing

In most cases, even after segmentation, there will be multiple zones where there are differences, which are usually caused by noise or camera vibration. If the noise is very small (one pixel in size), a routine of single pixel removal can be applied that will clear the image from single, unconnected pixels. But noise, or maybe small object movements, can appear as a small area. These small islands of noise can be removed by applying a morphological operation called erosion. This was applied with a  $3 \times 3$  cross-structured element. For each pixel all its four-connected neighbors will be examined. The effect of the erosion is that a width of one pixel will be removed from the margin from all areas in

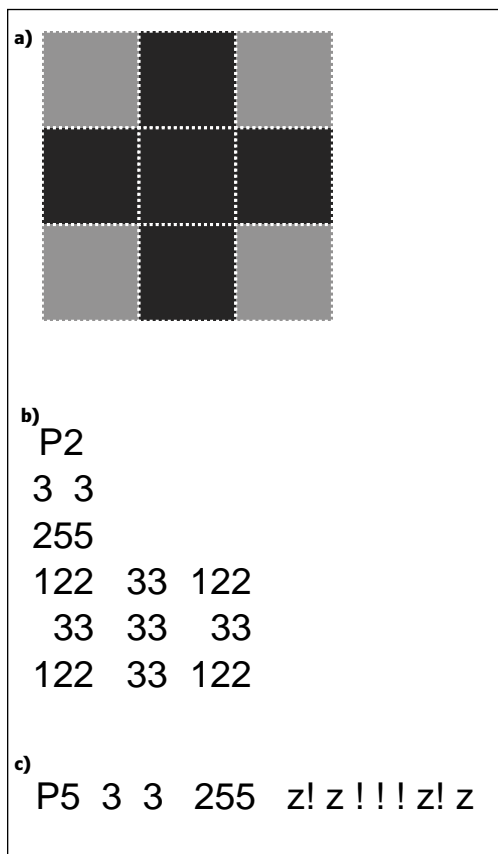


**FIGURE 1**

Overview of the security network

**FIGURE 2**

This is an image in NetPBM format (a), along with ASCII format (b) and binary format (c).



the image. This way a small island of noise will be removed completely. The erosion also affects the objects of interest, making them smaller. To repair this, the dilation operation is performed that adds back a “border” of one pixel to existing objects.

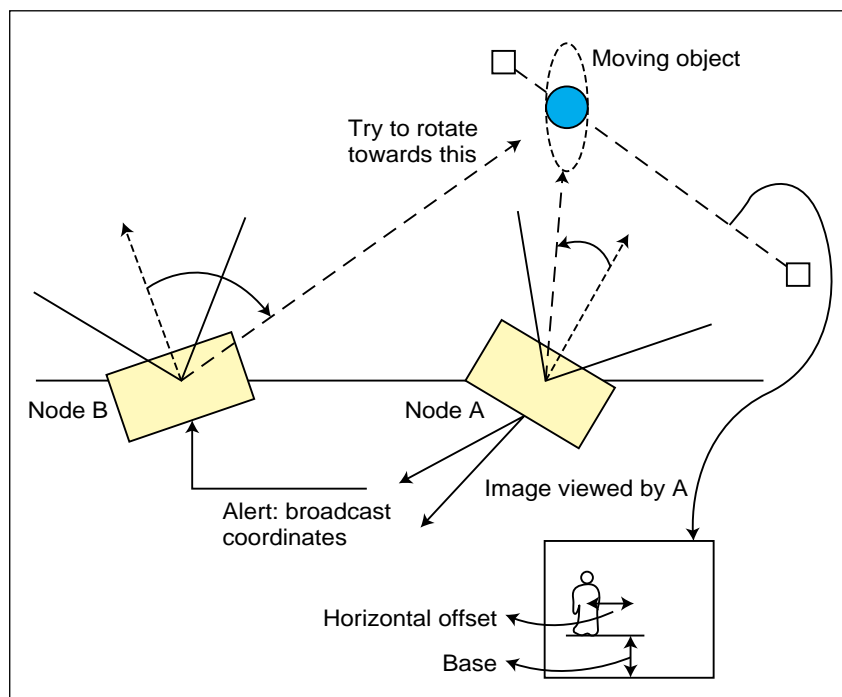
The next step is blob detection, which determines the characteristics (dimensions, area, center of mass, perimeter, and bounding box) of all objects present in the image. (The objects here are the areas where movement was detected.) This task can be easily accomplished on a PC by using a recursive algorithm (depth-first search) because it isn’t a problem having thousands saved stack frames in RAM. Because in a microcontroller’s RAM is limited, we implemented the breadth-first search method by maintaining in the program a custom queue where unprocessed pixels are progressively queued and then dequeued for processing, thus maintaining a very small memory footprint. At the end of it, for each object in the image, we have the area (the number of pixels), the center of mass (the average of all its pixel coordinates), the bounding box (the limits on the x- and y-axis), and perimeter. Based on the position of the object in the image, the angle relative to the camera is computed and movement can be initiated. Because the node knows its absolute position in space, using the Base and the Horizontal offset measurements from the image, a rough approximation of the position in space of the object can be found and also transmitted to the peers (see **Figure 3**).

When an object enters or leaves the camera’s field of view, a single large blob will be observed (the place where it entered or the place where it left). While an object moves through the camera’s field of view (depending on the movement speed, shape, color and texture of the object), two large blobs can be observed (the location from where it started and where it arrived).

When the PIC32 is running at only 8 MHz, the image processing routines take less than 200 ms. The blob detection routine takes a variable amount of time (depending on the number of objects and size), while the duration of the rest of the routines remains constant (approximately 100 ms).

## THE NODE

The execution steps performed inside each node are presented in **Figure 4**. At start-up, one of the UARTs is configured to communicate with the C328 camera and a secondary UART is set as a debug console. The W5500 chip is initialized using the SPI. A unique 48-bit MAC is obtained from a Microchip Technology 25AA02E48T SPI EEPROM with Node Identity. For the SD card access the MDD File System



**FIGURE 3**

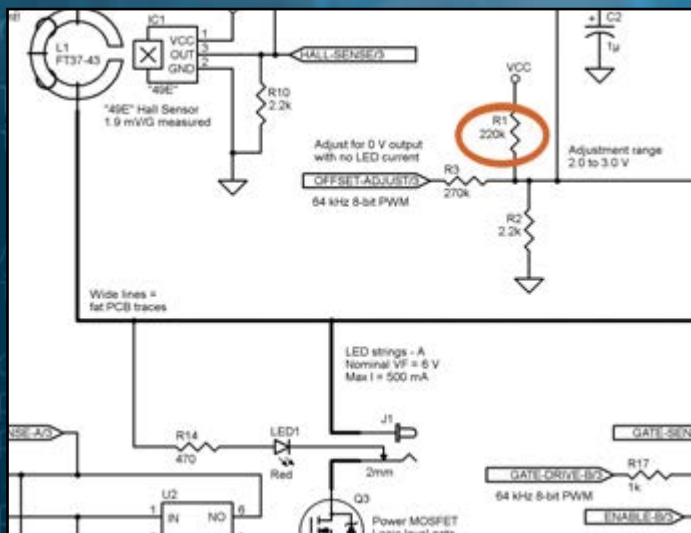
Node A detects movement, estimates the position of the object, and generates an alarm. Node B tries to rotate to capture the event.



# MONTHLY ENGINEERING CHALLENGE

Each month, you're challenged to find an error in a schematic or in code that's presented on the challenge webpage. Locate the error for a chance to win prizes and recognition in Circuit Cellar magazine!

Prizes such as a NetBurner MOD54415 LC Development kit or a Circuit Cellar subscription will be announced each month.



```

1  #include <stdio.h>
2
3  int main()
4  {
5      int first, second, sum;
6      int *p, *q;
7
8      printf("Enter two integers to add: \n");
9      scanf("%d%d", &first, &second);
10
11     p = &first;
12     q = &second;
13
14     sum = p + q;
15
16     printf("Sum of entered numbers = %d\n", sum);
17
18     return 0;
19 }

```

**Participate:** [circuitcellar.com/engineering-challenge-netburner](http://circuitcellar.com/engineering-challenge-netburner)

**Launch:** 1st of each month

**Deadline:** 20th of each month

No purchase necessary to enter or win. Void where prohibited by law. Registration required. Prizes subject to change based on availability. Review these terms before submitting each Entry. More info: [circuitcellar.com/engineering-challenge-netburner-terms](http://circuitcellar.com/engineering-challenge-netburner-terms)

libraries provided by Microchip are used. In order to correctly timestamp the locally saved files, each node keeps its own time using a timer that is initialized by performing an NTP request to an Internet server.

The main loop begins with a check for messages received from the network (by interrogating the W5500 chip). Only packets from within the WIZnet network are accepted (see **Figure 5**) and an action is performed

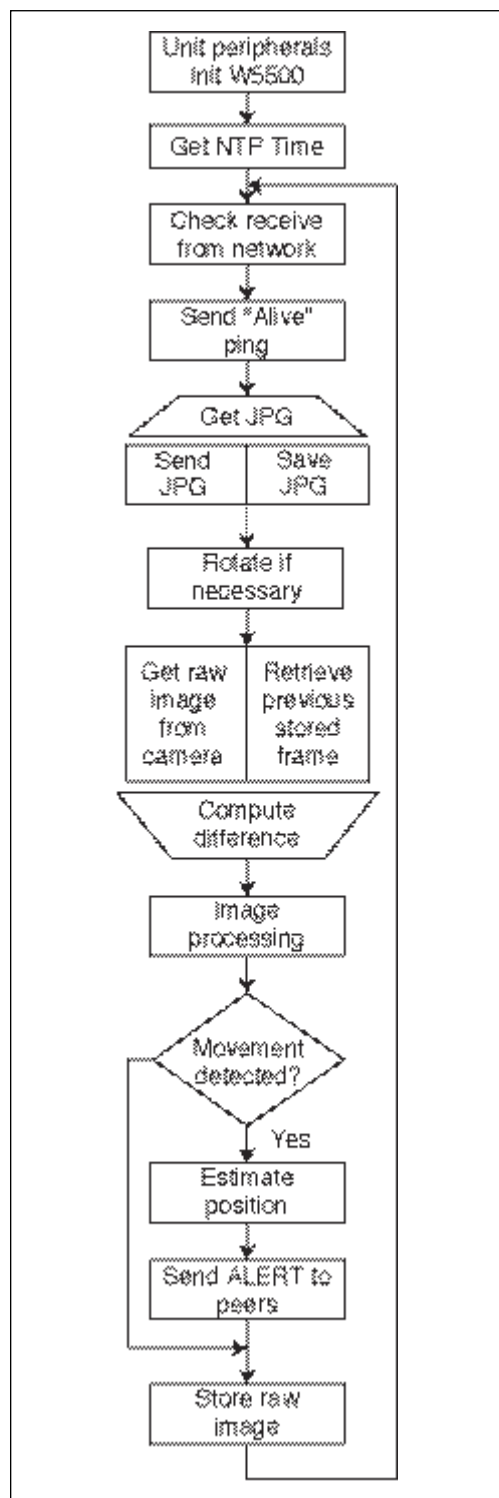
depending on the type of message received (see **Table 3**). A “ping” message is broadcasted so all the peers know the node is alive. Each node keeps a ping counter for all its peers, and if one of them stops sending, it assumes it has been tampered with and generates an alarm with the node’s position. A JPEG picture is taken from the camera, and, depending on the settings, it’s saved to the SD card and also sent to the PC monitor. The JPEG packets arriving from the camera are stripped of their header and sent to the server, as they have a convenient size of 506 bytes (i.e., less than the maximum size accepted by W5500 chip). If a movement command was queued (either from the user or from the object detection), it initiates a rotation (giving a command to the servo) to a new position. Then a raw image is captured (optionally saved to SD card as pgm) and image processing is performed as described above. If a node detects an event after image processing or from the PIR sensor, it broadcasts an alert message to all the peers containing an approximation of the location of the detected event (based on his position and the position of the object detected in the image).

When a node detects movement in an image (e.g., node A in **Figure 3**), it computes a rough estimation of the coordinates of the detected object in 2-D space. The horizontal offset of the detected blob in the image is used to compute the angle relative to the camera position. The base of the blob in the image is used to get a rough estimate of the distance of the blob from the camera. (It is assumed the object sits on the floor.) Each node knows its absolute coordinates and orientation and by computing the above offsets the approximate location where movement took place is determined and these 2-D coordinates are broadcast to all nodes in an alert message.

If a node receives an alert message from a peer, it checks to see if the coordinates included in the message are in its view angle and if it can rotate to get the event in his sight. If so, it initiates a rotation to that position.

## DATA COMMUNICATION

Our security network system features several types of communication: messages exchanged between nodes (using UDP broadcast); messages exchanged between a node and the monitor application; and queries launched by the node to an Internet server. All messages exchanged in the network use UDP packets with the header structure presented in **Figure 5**. The header length is a round number, 32 bytes. The unused locations are available for future use. The first 3 bytes are constant “WSN” to identify packets from the security network. Seq.No.



**FIGURE 4**  
Firmware execution flowchart

# Have you stopped by the shop lately?

*Circuit Cellar* is always adding new items to help with your design projects. Stop by often for the latest deals.

books  
audio  
back issues

contest cds

archive cds



[www.cc-webshop.com](http://www.cc-webshop.com)





0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
W	S	N	Type	*	Seq. No.	Frag. No.	Frag. Max	Alarm	Alarm	Alarm coord. X		Alarm coord. Y		*	*

**FIGURE 5**  
Structure of the header for the UDP messages

is a sequence number that is used for the logical frames. Frag.No. is the number of the current fragment used when a larger frame is spitted into smaller packets. This is necessary because the W5500 lacks IP fragmentation. JPEG images are transmitted in packets of 506 bytes, as they arrive from camera. Max.frag. is the number of packets into which the logical frame was split. Alarm fields are used to specify if an alarm was detected, its type, and its estimated 2-D coordinates. Type identifies the kind of frame, as listed in **Table 1**.

Due to the small size of the jpeg images (less than 64 KB), we initially intended to transmit an image in a single UDP packet, allowing the sending and receiving parties to use a simpler logic for image management. But this wasn't possible because the W5500 does not support IP fragmentation and it cannot split a large UDP packet by itself. In conclusion, this fragmentation has to be performed at the application level. So, images are now transmitted in smaller packets and have fields in the header that contains the fragment number, which allows reconstruction at the receiver. The transmission in smaller packets (although requiring more logic at the receiving side) has the advantage of using less memory in the microcontroller because the JPEG image arrives from the camera in packets and it's no longer required to buffer

the entire JPEG image.

**PC APPLICATION**

The nodes in the network can process and store images, as well as send messages to peers with alarms. But in order to estimate the absolute location in space where movement was detected, they need to know their position and orientation in space. This is where a PC application is needed to enable the user to input the coordinates of each node. The user will input a 2-D map of the surveyed area and place each camera in the right location (corresponding to the real one). This is done by adjusting the 2-D coordinates and the mounting camera angle (when the servo is in the middle position). The application interface is presented in **Photo 2**. Only two nodes were used for testing (as being the only available).

The second purpose of the application is for monitoring all the nodes. This takes place on the top-left side where two camera controls were instantiated. Each camera view works independently, taking care only of the associated camera; it receives packets (dispatched from the main window) containing pieces of JPEG images sent by the specific camera, it combines the pieces into one image based on the fragment number information from the header, saves it to disk with a unique name and displays it. If the

J	JPEG sent from camera to PC monitor application
P	Ping; each node broadcast this message once a second in order to let the others know it's alive
!	Alarm broadcast; the alarm fields are filled with type of event, 2-D location, range
L	Localization information about nodes in the network; this is sent by the PC application to the nodes, and contains IPs and the absolute position and orientation in 2-D space
M	Move command issued by the PC monitor application to manually rotate the camera
T	Command to get and sync the local time with Master time
	Debug commands:
j	Request to get a JPEG image
b	Request to get a raw (.pgm) image
1	Command to turn the Node ON
0	Command to turn the Node OFF
R	Reset the node

**TABLE 1**  
Types of packets vehiculated in the network

# PCB WEST 2015

## Conference & Exhibition

**CONFERENCE: September 15 - 17**  
**EXHIBITION: Wednesday, September 16**  
**SANTA CLARA CONVENTION CENTER, CA**

### PCB WEST OFFERS:

- A comprehensive three-day conference, with several affordable packages
- More than 65 presentations in all, the largest PCB event in the Silicon Valley
- Targeted conference sessions for all levels of experience and training, from novice designer and engineer to seasoned pro
- Covering topics such as RF/microwave/PI/SI, printed electronics/flex circuits and next-gen components
- Exhibit hall featuring the industry's leading suppliers and services in a one-day exhibition
- 10+ Free technical sessions PLUS networking events - all on the exhibit floor - lunch, afternoon breaks and an evening reception
- and more...

**Register for the conference by August 14th and save up to \$100**

**[www.pcbwest.com](http://www.pcbwest.com)**

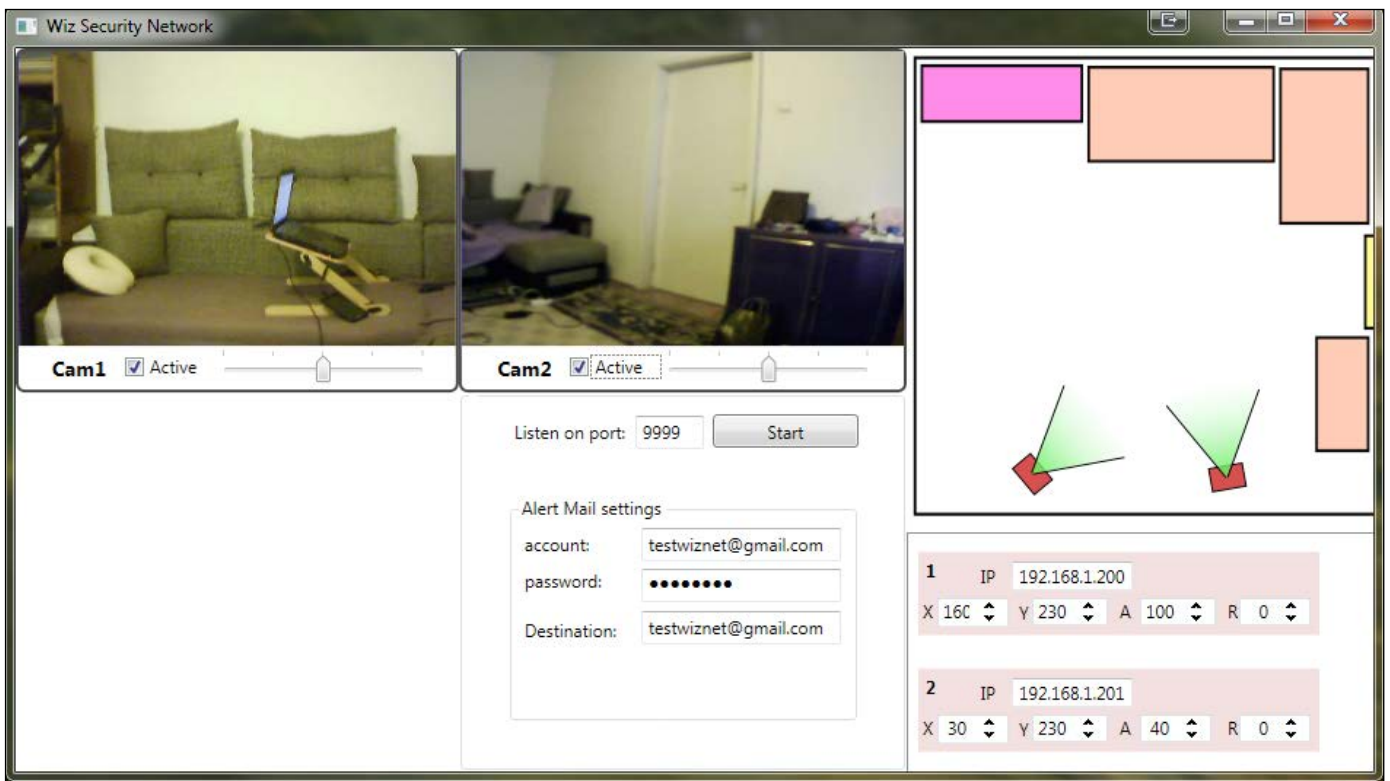



PHOTO 2

Monitoring PC application

image contains information about an active alert will notify the main window. Using the

slider below the image, the user can manually rotate the camera.

The PC application was written in C# with WPF components using Visual Studio IDE. When the application is started, it begins listening for messages on port 9999. It then sends the layout of the network (the position information for each node from the right side) to all nodes on the network. When a jpeg packet is received, it is forwarded (based on its IP) on the corresponding camera view for reassembly and display. When an alarm flag is detected in a packet, the application it will collect a number of images from all incoming streams and send an e-mail with the images at attachments, so a remote user can have several images from all cameras in a single e-mail.

The realized system is not a ready-to-deploy network, but it shows how to construct a network of embedded devices when you have limited resources. The nodes are capable of independently processing information from a video camera, and they can reorient themselves for a better view. In addition, the nodes are capable of exchanging information for improved performance. It also shows that simple image processing techniques can be successfully implemented in medium-sized microcontrollers. To improve performance, the system will require a camera with a faster interface, slightly better resolution, improved image-processing algorithms, and encrypted communications. 

## ABOUT THE AUTHORS

Claudiu Chiculita has an MS in Computer Science, an MS in Telecommunications, and a PhD in Control Systems. His interests include researching, teaching, and developing embedded systems and software. You may reach him at [claudiu.chiculita@gmail.com](mailto:claudiu.chiculita@gmail.com).

Liviu Ene has a BE in Applied Electronics. He is currently an MTech student in Advanced Informatics Technologies. Liviu's main area of interest is embedded systems. You may reach him at [ene.liviu09@yahoo.com](mailto:ene.liviu09@yahoo.com).



[circuitcellar.com/ccmaterials](http://circuitcellar.com/ccmaterials)

IrfanView graphic viewer  
IrfanView | [www.irfanview.com](http://www.irfanview.com)

PIC32MX250F128 Microcontroller  
Microchip Technology, Inc. | [www.microchip.com](http://www.microchip.com)

L5973D 2.5-A Regulator  
STMicroelectronics | [www.st.com](http://www.st.com)

W5500 Network chip  
WIZnet | [www.wiznet.co.kr](http://www.wiznet.co.kr)

## SOURCES

C328 JPEG Camera Module  
COMedia Ltd. | [www.comedia.com.hk](http://www.comedia.com.hk)





# The #1 Flash Memory Conference!

## Featuring Flash in Enterprise Storage

### Keynotes

**Micron • NetApp**  
**Oracle • PMC**  
**Samsung • SanDisk**  
**Seagate • SK-Hynix**  
**Toshiba • Toyota**  
**Kaminario • Tegile**

*“Solid state drives  
will be the biggest  
change in storage, a  
total game-changer,  
and flash will be the  
dominant type of  
SSD for the  
foreseeable future.”*

Joe Tucci, EMC

### Features

**Major Exhibitors**  
**Top Industry Keynotes**  
**Leading Experts**  
**SSDs and Controllers**  
**New Technologies**  
**Enterprise Storage**  
**Pre-Conference Seminars**  
**VC Forum**  
**Market Research Session**

Are you struggling with crucial Solid State Drive (SSD) decisions? Can SSDs resolve your application bottlenecks? How can you maximize SSD performance? Flash Memory Summit will explore new frontiers in enterprise storage and help you make the right choices.

### Highlights

Three days packed full of seminars, forums, keynotes, and sessions:

- Plenary on 3-D Flash
- Forums on NVMe, Architectures, Enterprise SSDs, Enterprise Storage Design, PCIe SSDs, and Application Performance
- Sessions on Enterprise SSD Buying Trends and Flash Storage Performance Measurements
- New Technologies Sessions: RRAM, MRAM, and Life Beyond Flash
- Annual Update: Flash Technology, Enterprise Flash Storage, Interfaces, and New Technologies
- Beer, Pizza and Chat with the Experts Session

Don't miss the 10th Annual Flash Memory Summit and Exhibition! Hear from industry leaders, learn about the latest technologies, and talk to key vendors. Nothing else comes close!

**Register online today**

**[www.FlashMemorySummit.com](http://www.FlashMemorySummit.com)**

*Circuit Cellar readers: Enter priority code  
SPGP for \$100 discount!*

**FREE PARKING**



**August 11-13 2015**

Santa Clara Convention Center



produced by Conference ConCepts, Inc

## QUESTIONS & ANSWERS



# Innovations in Wearable Technology

## An Interview with Bruce Thomas (Director, Wearable Computer Lab, The University of South Australia)

New developments in wearable technology, virtual reality, and augment reality research are poised to change everything from healthcare to gaming. We recently asked Professor Bruce Thomas to tell us about the research in these areas taking place at the Wearable Computer lab at the University of South Australia.

**CIRCUIT CELLAR:** How did you become interested in electronics and computing? Did you start at a young age?

**BRUCE:** I started programming in high school in 1972. I have always been interested in electronics, but with the latest Maker-style community, I am able to follow and learn this skill.

**CIRCUIT CELLAR:** You studied physics as an undergraduate at George Washington University. What led you to transition to computer science for your graduate studies?

**BRUCE:** I have always been interested in computer science. I studied to become physics teacher. After I taught for two years, I went back and got my Master's in computer science

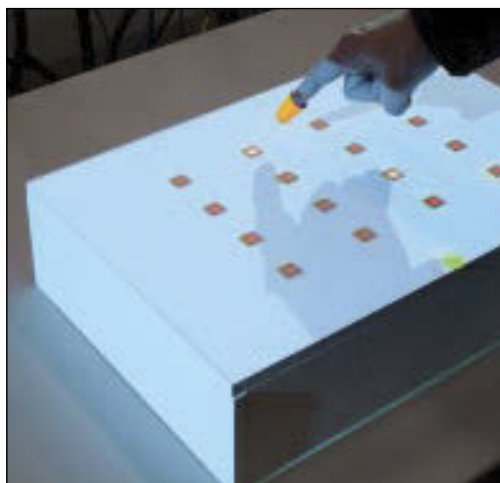
at the University of Virginia. I guess I wanted a new challenge.

**CIRCUIT CELLAR:** In 1997, Carnegie Mellon, Georgia Tech, and MIT hosted the first IEEE International Symposium on Wearable Computers. How has research in the area changed since the late 1990s?

**BRUCE:** I remember the 1997 conference well. We presented one of our first wearable computer papers there. Some differences are: People do not build their own wearable computers anymore. Back then you had to hack your own system. Today you buy a smartphone and you have all the computing power you need. We now use Mac minis, and this replaces about five or six components when we started. The research now is very focused on contextual aware computing. In 1997 the focus was on all disciplines of computer science. There is less work on garment integrated systems today. A large advantage is there are many computing and wearable platforms to start with.

**CIRCUIT CELLAR:** How did the Wearable Computer Lab at the University of South Australia come into being in 1998?

**BRUCE:** Someone in the Australian Defense Science and Technology Organization (a defense lab) showed me two Phoenix 486 belt mounted wearable computers with Private Eye head-mounted displays (HMDs). He told me this was the next big thing for defense, and he asked how I can help him out with some research. He wanted to use the new Sony



Interactive prototyping with special augmented reality (SAR). Here a user is interacting with projected buttons. (Image used with permission from Bruce H. Thomas, Wearable Computer Lab, University of South Australia)

## QUESTIONS & ANSWERS



The ARQuake system is an interactive outdoor augmented reality collaboration system that enables users to walk around in the real world while playing the computer game Quake. (Image used with permission from Bruce H. Thomas, Wearable Computer Lab, University of South Australia)



Glasstron see-through displays. The first thing I thought of was, "This would be great for outdoor augmented reality."

**CIRCUIT CELLAR:** You supervised Wayne Piekarski's Tinminth augmented reality backpack project, which started in the late 1990s. Can you give us a brief overview of the project and how it evolved from 1998 to 2006?

**BRUCE:** The project went from a fixed frame for a backpack used in bush walking loaded with many devices to a belt-mounted system with just the Mac mini modified and a helmet (with many sensors and HMD). I wanted to port this

onto a smartphone, but we never got there.

The project had many different software architectures. The system was a research platform to support outdoor augmented reality user interaction research. The system was very flexible and robust.

**CIRCUIT CELLAR:** Which of the Lab's current projects most interests you at this time?

**BRUCE:** We are interested in portable haptic devices. Dr. Ross Smith is leading the research effort into this. This involves layered jamming placed in a mitten to provide mobile haptic sensations to the user. This was presented at the International Symposium on Wearable



## QUESTIONS & ANSWERS

Wearable Jamming Mitten for virtual environment haptics developed by Tim Simon with Dr. Ross Smith and Professor Bruce H. Thomas (Image used with permission from Bruce H. Thomas, Wearable Computer Lab, University of South Australia)



Computing in 2014. We are continuing this work.

**CIRCUIT CELLAR:** Can you tell us about your most current project or projects at the Lab?

BRUCE: We have moved into large-scale projector-based augmented reality. We are looking into tools to support designers of large spaces, but command and control rooms. We are about to start a seven year project working with Jumbo Vision International with the Innovative Manufacturing CRC.

**CIRCUIT CELLAR:** Tell us about the current crop of students working at the Lab. Which area of research (e.g., virtual reality) is popular among the students?

BRUCE: There are many topics. Two are looking into the use of projector-based augmented reality to support remote collaboration. One is looking into the application cognitive psychology into improving augmented

reality presentations. One is investigating visualization techniques for big data. One is working on the haptic mitten. One is looking at situated analytics, the application of AR to visual analytics.

**CIRCUIT CELLAR:** We assume many companies are interested in the research you do. How much does input from industry affect what you focus on in the lab?

BRUCE: The research is driven by real-world problems and open research questions. I would say about 50% of the research is driven by industry and 50% is blue-sky research.

**CIRCUIT CELLAR:** In an April 2015 TechTimes.com article, "Augmented Reality vs. Virtual Reality," Vamien McKalin writes: "It is clear that the way things are right now, AR has the upper hand against VR, and that might not be changing anytime soon." Do you agree?

BRUCE: It all comes down to who wins the head-mounted display wars. I think both will be winners in the near future. They solve different problems and have different uses. I can see a VR HMD attached to many games controllers, and I can see people using AR displays in their workplace.


**CIRCUIT CELLAR:** When you think about the short term (5 to 10 years), in which area do you think wearable technology will make the biggest impact: consumer, healthcare, military, or enterprise?

BRUCE: In 5 to 10 years, I think the biggest impact will be in healthcare. Wearables will go beyond fitness monitoring to health monitoring. This will enable better monitor of health issues and provide doctors much more diagnostics to help people.

A second area is extending the ability for elderly people to stay in their homes. Better health monitoring and activity monitoring will allow people to safely stay at home longer.

**CIRCUIT CELLAR:** What is the "next big thing" in wearables? Is there a specific area or technology that you think will be a game changer?

BRUCE: If we can make batteries a tenth the size they are now, this will be a major breakthrough. When you talk to people about battery life it is either one day or one week. Anything in between does not matter. This can be done by greater capacity or different recharge methods or better electronics.

The sci-fi answer is contact displays. Those would be cool. 

This is a virtual reality simulation system that supports research relating to chronic neck pain therapies developed by Dr. Markus Broecker and Dr. Ross Smith. (Image used with permission from Bruce H. Thomas, Wearable Computer Lab, University of South Australia)



# When it comes to robotics, the future is now!



From home control systems to animatronic toys to unmanned rovers, it's an exciting time to

be a roboticist. *Advanced Control Robotics* simplifies the theory and best practices of advanced robot technologies, making it ideal reading for beginners and experts alike. You'll gain superior knowledge of embedded design theory by way of handy code samples, essential schematics, and valuable design tips.

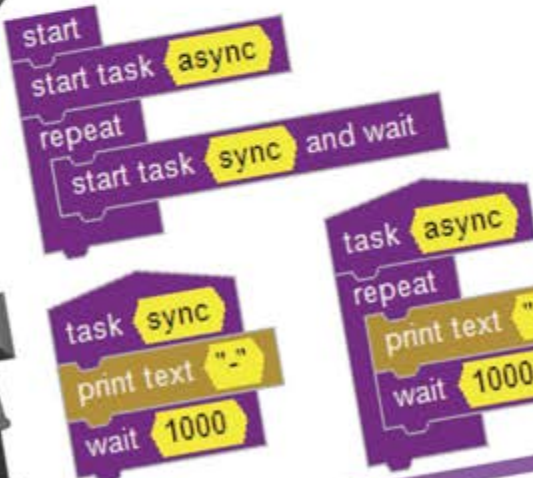
*With this book, you'll learn about:*

- Communication Technologies
- Control Robotics
- Embedded Technology
- Programming Language
- Visual Debugging... and more

## ADVANCED CONTROL ROBOTICS

HANNO SANDER

AD



HANNO SANDER

Get it today at [cc-webshop.com](http://cc-webshop.com).



## THE CONSUMMATE ENGINEER

# Ground Loop Blues

COLUMNS

A ground loop is an annoyance that can cause noise and interference in your electronic systems. In this article, George explains the cause of ground loop interference and provides tips for finding and eliminating ground loops in analog AV systems.

*By George Novacek (Canada)*

Everything had been fine with my home entertainment center—comprising a TV, surround-sound amplifier, an AM/FM tuner, a ROKU, and a CD/DVD/BlueRay player—until I connected my desktop PC, which stores many of my music and video files on one of its hard drives. With the PC connected, the speakers put out a low level, annoying, 60-Hz hum—a clear indication of a ground loop. All my audio and video (AV) devices are fairly new, quality, brand-name products equipped with two-prong power cords, so even though the PC has a three-prong plug, there should not be multiple signal returns causing the ground loop. This article describes an approach to eliminating ground loops in analog AV systems.

## GROUND LOOPS

By definition, ground loops bring about unwanted currents flowing through two or more signal return paths. Thus induction coils are formed, usually of one turn only. These loops pick up interference signals from the environment. Because every conductor has a finite impedance, a voltage potential— $V_i = I_g(R_1 + R_2)$ —develops between the two connected signal return points. This voltage is the source of the interference: a hum, hiss noise that high-frequency signals pick

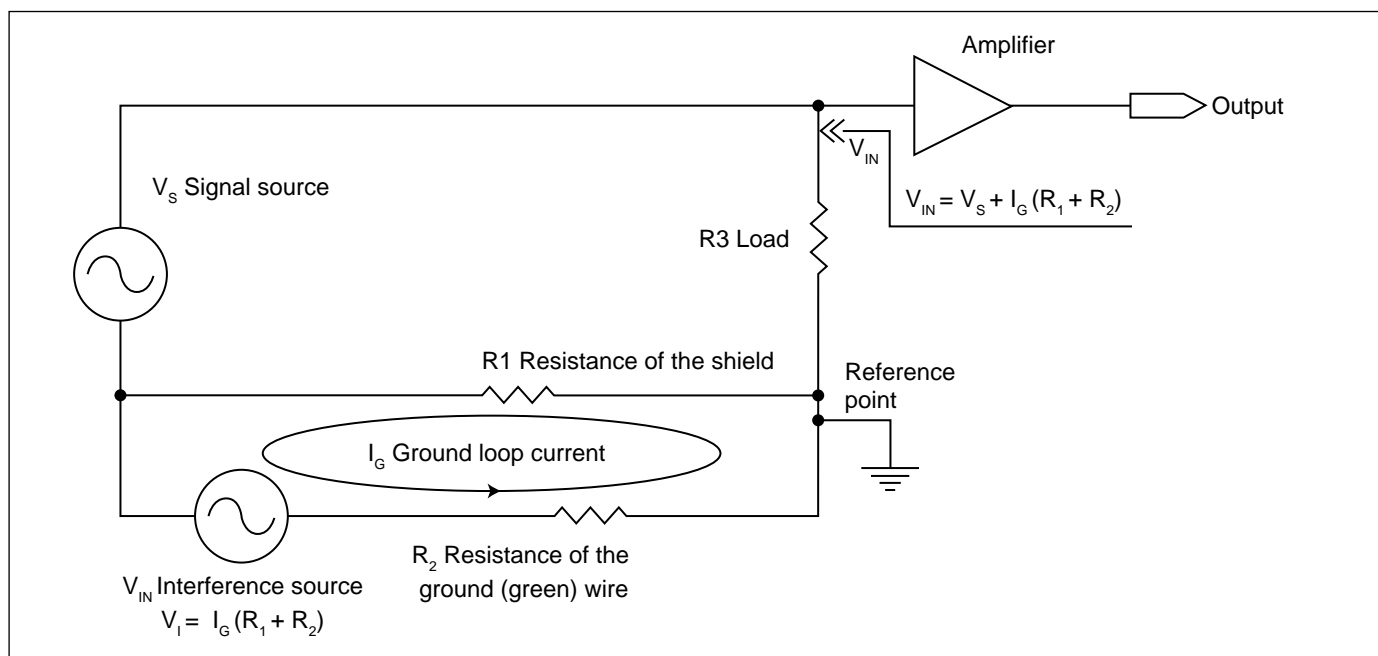
up (e.g., a local AM station), and so forth. A simplified example is illustrated in **Figure 1**.

An audio signal source  $V_s$  in Figure 1—an audio card inside the PC, for example—is connected to an amplifier via a shielded cable. The shield is grounded at both ends to the chassis of both devices. Three-prong power plugs connect the chassis of both AV components to the house power distribution ground wire. Let's consider the amplifier ground to be the reference point. (It doesn't matter which point in the loop we pick.) The loop, comprising the cable shield and the power distribution ground wire, picks up all kinds of signals causing loop current  $I_g$  to flow and as a result interference voltage  $V_i$  to be generated.

$V_i$  is added to the signal from the audio card. The  $I_g$  current induced into the loop comes from many potential sources. It can be induced in the ground wire by the current flowing in the 120-VAC hot and its return neutral wires, acting like a transformer. There can be leakages, induction by magnetic fields, capacitive coupling, or an electromagnetic interference (EMI) induction into the loop. Once  $V_i$  is added to the signal it is generally impossible to filter it out.

Much of electrical equipment requires the third power prong for safety. This is





**FIGURE 1**  
Cause of the ground loop interference.

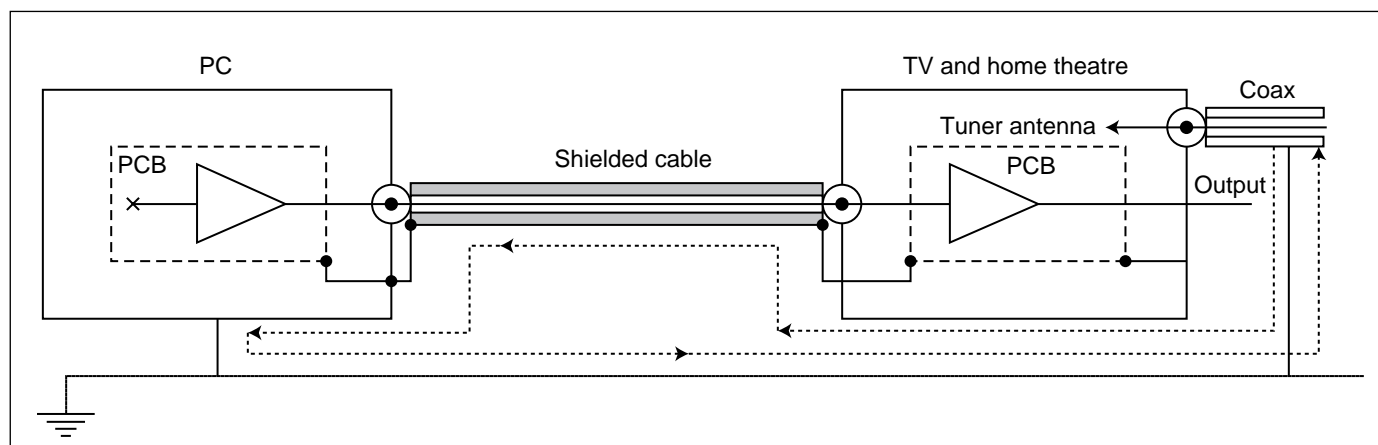
connected to the chassis and at the electrical distribution panel to the neutral (white wire) and the local ground—usually a metal stake buried in the earth. The earth ground is there to dissipate lightning strikes but has no effect on the ground loops we are discussing.

The ground wire's primary purpose is safety plus transient and lightning diversion to ground. Under normal circumstances no current should flow through this wire. Should an internal fault in an appliance connect either the neutral (white) or the hot (black or red) wire to the chassis, the green wire shunts the chassis to the ground. Ground fault interrupters (GFI) compare the current through the hot wire to the return through the neutral. If not identical, the GFI disconnects.

Manufacturers of audio equipment know that grounding sensitive equipment at

different places along the ground wire results in multiple returns causing ground loops. These facilitate the interference noise to enter the system. From the perspective of electrical safety, the small currents induced in the ground loop can be ignored. Unfortunately, they are large enough to play havoc with sensitive electronics. The simplest solution to the dilemma is to avoid creating ground loops by not grounding the AV equipment. Thus the two-prong plugs have been used on such equipment. To satisfy the safety requirements, the equipment is designed with double insulation, meaning that even in case of an internal fault, a person cannot come into contact with a live metallic part by touching anywhere on the surface of the equipment.

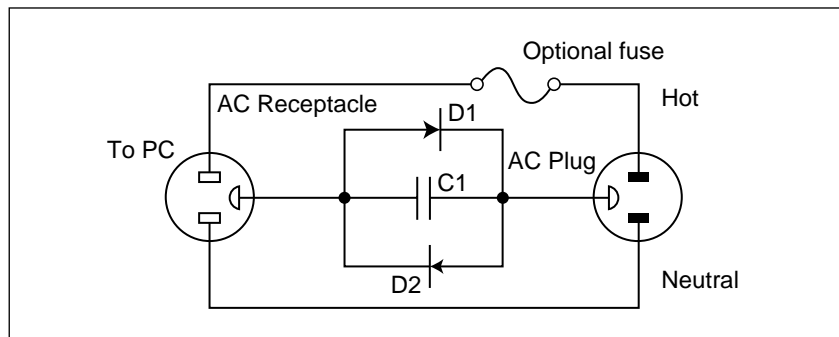
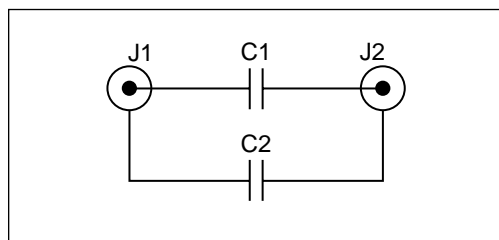
My PC, like most desktops, has a three-prong plug. **Figure 2** shows the arrangement.



**FIGURE 2**  
Ground loop in my entertainment system

**FIGURE 3**

Ground isolator for CATV coax

**FIGURE 4**

Ground isolator for three-prong powered appliances

**ABOUT THE AUTHOR**

George Novacek is a professional engineer with a degree in Cybernetics and Closed-Loop Control. Now retired, he was most recently president of a multinational manufacturer for embedded control systems for aerospace applications. George wrote 26 feature articles for

*Circuit Cellar* between 1999 and 2004. Contact him at [gnovacek@nexicom.net](mailto:gnovacek@nexicom.net) with "Circuit Cellar" in the subject line.



[circuitcellar.com/ccmaterials](http://circuitcellar.com/ccmaterials)

**RESOURCES**

B. Whitlock, "Understanding, Finding, & Eliminating Ground Loops," Jensen Transformers, 2008, [http://web.mit.edu/~jhawk/tmp/p/EST016\\_Ground\\_Loops\\_handout.pdf](http://web.mit.edu/~jhawk/tmp/p/EST016_Ground_Loops_handout.pdf).

The PC is grounded through its power cord. Unfortunately, the cable TV (CATV) introduces a second ground connection through its coax connector. I measured the resistance between the coax shield as it entered the house and the house power distribution ground wire. The resistance was 340 mΩ, indicating a hard connection between the coax shield and the house ground, the cause of the ground loop. I was unable to establish where that connection was made, but it wasn't through the earth.

There can be multiple ground loops around a computer system if you have hard-wired peripherals with three-prong plugs, such as some printers, scanners and so forth. Digital circuits are much less sensitive to ground loops than the analog ones, but it is a good idea to minimize potential loops by connecting all your peripherals, other than wireless, into a single power bar.

Ground loops may also be created when long shielded cables are used to interface the PC and the home theatre box. Two shielded cables needed for stereo represent two signal returns creating a ground loop of their own. And then there are video cables. Another loop. Fortunately, connectors on the back of the PC and AV equipment are very close to each other, which means a minimal potential difference between them at low frequencies. Stereo cables keep the loop small. To minimize all the loops' areas for interference pick-up, I have bundled the interface cables very close to each other with plastic wire ties. In severe situations re-routing the cables or the use of a metal conduit or wireless interfaces may be needed to kill the interference.

**FIXES**

Having disconnected the CATV cable from the TV, the hum went away. As well, temporarily replacing the PC with a laptop, which is not grounded, also fixed the problem. So how else can we fix those offending multiple returns?

The obvious answer is to break the loop. I strongly suggest you don't disconnect the PC from the ground by using a two-prong plug adapter or just cutting the ground prong off. It will render your system unsafe.

What you need is a ground isolator. Jensen Transformers, for example, sell isolators such as VRD-IFF or PC-2XR to break the ground connection, but you can build one for a small fraction of the purchase price. **Figure 3** and **Figure 4** show you how.

To break the ground loop caused by the CATV, you can make a little gizmo shown in Figure 3. J1 and J2 are widely available cable TV female connectors. C1 and C2 capacitors placed between them should be about 0.01 μF each. The assembly does not require a printed


circuit board. You might place it in a tiny box or just solder everything together, wrap it with electrical tape, and put it somewhere out of the way. Remember that the capacitors' working voltage must be at least double the power distribution voltage. That is 250 V in North America and more than 500 V elsewhere in the world.

Figure 4 shows how to break ground for appliances, such as a PC, with three-prong plugs. You can build this circuit into a computer or another appliance, but I find it better to build it as an independent break-out box. The diodes provide open loop for signals up to about  $1.3 V_{pp}$ . A hum is usually of a substantially lower amplitude. C1, 0.01  $\mu F$ , provides bypass for high-frequency EMI to ground. The loop would be closed for voltages higher than  $1.3 V_{pp}$ , such as the ones due to isolation fault of the hot wire to the chassis. For 120 VAC distribution, D1, D2, and C1 should be rated for 250 V at a minimum. In a circuit branch with a 15-A breaker or fuse, the diodes need to be rated for a minimum of 20 A so that the breaker opens up before the diodes blow. If the appliance takes only a fraction of the rated fuse current, say 2 A, you could use 5-A diodes and include an optional

fuse rated for 2 A. For countries with 230-VAC power, the components must be rated accordingly.

You can also break the ground loop by using a power isolation transformer between the power line and the PC, or quality signal transformers on the signal lines. The downside of this is that good isolation and signal transformers are costly and not widely available. Equipment powered from wall warts—and especially those with optically coupled inputs and outputs, common today—is inherently ground loop impervious.

## TRIAL & ERROR

This article describes an approach to eliminating ground loops in analog AV systems. While you need to understand how ground loops occur, finding them and eliminating their effects may turn out to be a matter of frustrating trial and error. 

## The Lowest Prices on the Best Scopes







**Passport-Size PC Scopes** \$129+  
 Great scopes for field use with laptops. Up to 200MHz bandwidth with 1GSa/s, high speed data streaming to 1MSa/s, built-in 1GSa/s AWG/function gen. PS2200A series

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

**50MHz Scope** \$399  
 50MHz, 4-ch scope at 2-ch price! Up to 1GSa/s rate and huge 12Mpts memory! Innovative "UltraVision" technology for real time wfm recording. FREE carry case! DS1054Z

**60MHz Scope** \$349  
 Best selling 60MHz, 2-ch scope with 500MSa/s rate plus huge 10MSa memory! 8-in color TFT-LCD. Includes FREE carry case and 3 year warranty! SDS5062V

**200MHz - 1GHz Scopes** \$1,500+  
 2/4 channel, 8 or 12 bit with long memory, powerful debug capabilities (Teledyne LeCroy) WaveAce 2000 Series


• Free Technical Support  
 • Excellent Customer Service



## Low Power ARM Module

### SoM-A5D36

- Atmel ARM Cortex A5 536Mhz Processor
- 4GB of eMMC Flash
- 512 MB of LP DDR2 RAM
- 16MB of Serial Data Flash
- 22 GPIO (3.3V) Lines
- 6x Serial Ports
- 24-bit LCD Controller
- Up to 720P Video
- Touch Controller
- External Address/Data Bus
- Internal Real time clock/calendar
- 4 PWM Channels, 5 Timer/Counters
- 10/100/1000 BaseT Ethernet
- 2x USB 2.0 High Speed Host ports
- 1x USB 2.0 High Speed Host/Device port
- 6 channels of 12 bit A/D (0 to 3.3V)
- 200 pin SODIMM form factor (2.66" x 2.375")




### Industrial Temperature

Designed and manufactured in the USA, the SoM-A5D36 is a System on Module (SoM) based on the Atmel ARM Cortex A5 ATSAMA5D36 processor. This low power, wide temperature ARM 536 MHz SoM utilizes 4GB of eMMC Flash, 16MB of serial data flash, and up to 512MB of LP DDR2 RAM. Like other modules in EMAC's SoM product line, the SoM-A5D36 is designed to plug into a custom or off-the-shelf carrier board containing all the connectors and any additional I/O components that may be required. Qty 1 pricing is \$155. Please contact EMAC for OEM & Distributor Pricing.

[http://www.emacinc.com/products/system\\_on\\_module/SoM-A5D36](http://www.emacinc.com/products/system_on_module/SoM-A5D36)

Since 1985  
 OVER  
 30  
 YEARS OF  
 SINGLE BOARD  
 SOLUTIONS



Phone: ( 618) 529-4525 • Fax: (618) 457-0110 • [www.emacinc.com](http://www.emacinc.com)



## PROGRAMMABLE LOGIC IN PRACTICE

# Just Enough Programmable Logic

## The PSoC Advantage

```
/* Setup hardware modules */
UART_Start();
CRC_TX_Start();
```

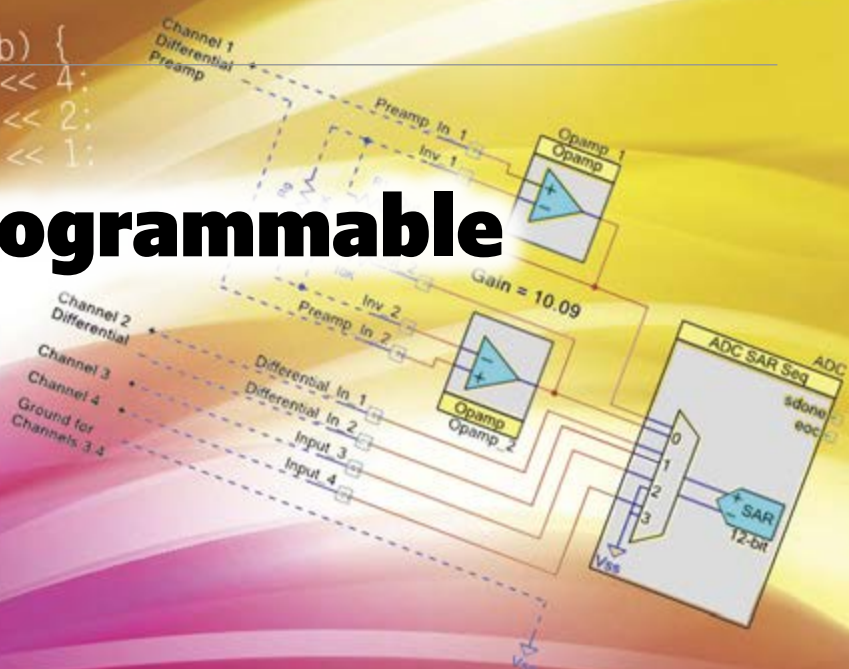
```
/* Write DE, AD, BE, EF */
UART_PutChar(0xDE);
UART_PutChar(0xAD);
UART_PutChar(0xBE);
UART_PutChar(0xEF);
```

```
/* Wait until transmit complete */
while(((uint8)~UART_ReadTxStatus
```

```
/* Read CRC from hardware module */
crc = CRC_TX_ReadCRC();
```

```
/* Swap bit order */
crc = reverse(crc);
```

```
/* Append to packet */
```



Sometimes you need a microcontroller with just a bit of glue logic or even a customized peripheral. While you could implement a soft-core processor in an FPGA and use the FPGA fabric for your custom logic, this probably involves far more programmable logic than you need. As another option is a PSoC device, which has a small amount of programmable logic for such simple operations.

By Colin O'Flynn (Canada)

Several of my previous columns have discussed the combination of a microcontroller and FPGA, either through a soft-core processor implemented in the FPGA or through physically separate chips. But all of these columns have assumed the FPGA side was from one of the large vendors of programmable logic (such as Xilinx, Altera, Microsemi, or Lattice Semiconductor).

Occasionally, however, manufacturers besides the regular programmable logic giants have products to achieve a similar goal of combining programmable logic fabric with a microcontroller core. These products aren't always a clear win. Developing the synthesis tools for the programmable logic aspect is no small feat—they may just license an external tool, for example—and it makes it hard for a company only dabbling in programmable logic to offer cost or feature-competitive devices, compared to the companies that are dedicated to programmable logic.

For example, Atmel, maker of the wildly popular AVR microcontroller, has a product called FPSLIC, which marries an AVR core with a FPGA fabric. This seems like a great

combination in theory. Unfortunately, the FPSLIC is considerably more expensive than the cost of a separate AVR microcontroller and similarly sized FPGA device from Xilinx or Altera. The FPSLIC instead targets the close interaction possible with the on-die integration, such as dedicated interrupts from the FPGA fabric going into the AVR. Personally, in such a situation, I'd be more likely to turn to a soft-core processor, such as I covered in my August 2014 column in *Circuit Cellar*. Using a soft-core processor also gives me an easy upgrade path to much larger FPGAs should I require it, whereas the Atmel FPSLIC part only come in a few smaller sizes.

But this column is about what I consider a device interesting enough to tilt me away from the soft-core option. This device is the PSoC series from Cypress Semiconductors. What makes this part interesting is the ability to easily configure the programmable logic aspect, while still being able to modify the underlying hardware design if required. Before jumping into the programmable logic aspect I need to give you a brief overview of the PSoC device, and after that we can look

at the specifics of the programmable logic system, before finally wrapping up with a simple example.

## INTRO TO PSOC ARCHITECTURE

It's worth first noting there are several families of the PSoC device. I'll be using the PSoC 4 here (which has an ARM Cortex-M0 microcontroller core), but most of what I talk about applies to the PSoC 5LP (with an ARM Cortex-M3 core) and the PSoC 3 (with an 8051 core). Even with that, I leave the caveat that specific features further vary within each family, so be sure to check detailed datasheet information. This article will generally refer to PSoC 4 devices, with some features of more advanced families also mentioned.

The word "programmable" normally refers to digital logic. In the case of the PSoC system, the programmable aspect extends to include programmable analog blocks. Depending on the specific device or family this could include several op-amps, DACs, ADCs, comparators, mixers, and S&H circuits.

This isn't just the normal programmable amplification or differential inputs on the front of an ADC that many microcontrollers have. Instead there is a complete analog matrix allowing arbitrary connection of these blocks, even for example routing the op-amp connections to external pins, allowing you to simply use the op-amp block as a replacement for a discrete op-amp in your circuit.

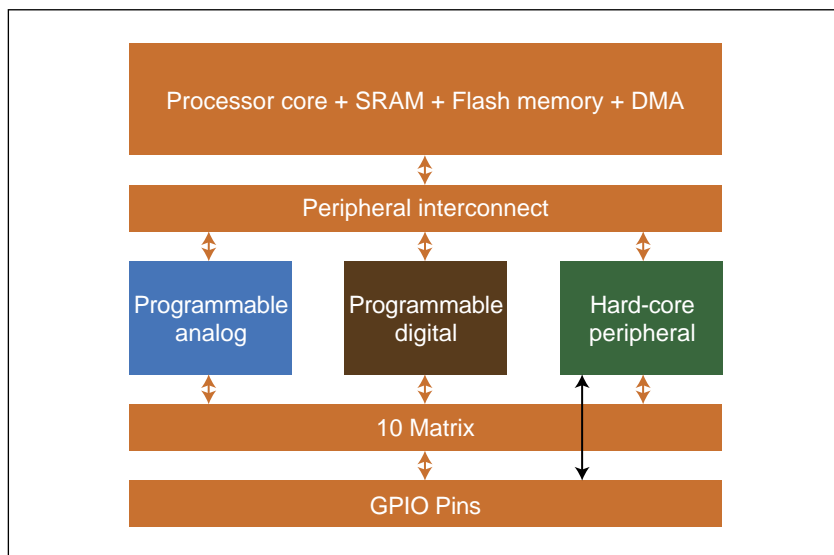
There are also programmable digital blocks, which can be used for a combination of standard peripherals or to implement your own custom logic. I'll discuss the programmable logic blocks in more detail later, so for now you can imagine these digital blocks as the mythical solution to all your problems (unfortunately this dream of solving all your problems be crushed in a few short paragraphs, but they can still solve some problems).

**Figure 1** shows a block diagram of the PSoC device, showing some of the programmable blocks, hard-core peripherals, processor core, and connection busses.

Practically, the PSoC has the advantage of fairly low device cost and physically simple packages. To give you a specific example, a PSoC 4200 device such as the CY8C4245AXI-473 costs \$2.32 (quantity 100, Digi-Key) and is in a TQFP-44 form factor. This means you can use a simple two-layer PCB for example in many applications, and are using very mature technology for the assembly process.

## DIGITAL PROGRAMMABLE LOGIC

While you might assume the PSoC device simply has some FPGA fabric sprinkled around

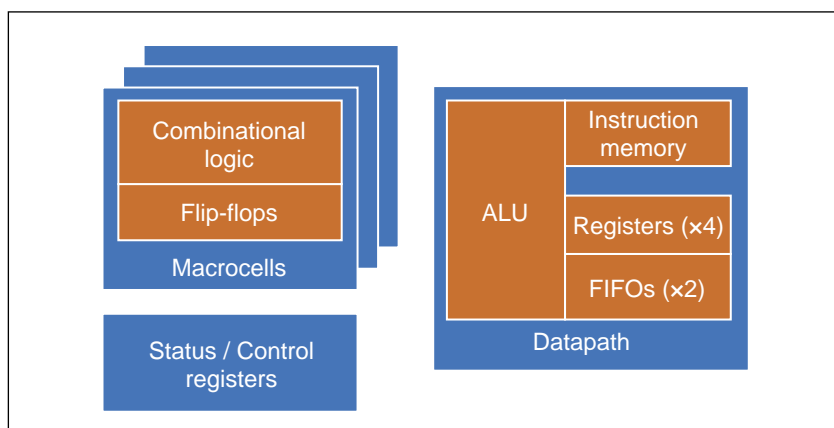


**FIGURE 1**

This simplified block diagram shows some features of the PSoC 4 device. The I/O matrix on the output gives flexible routing, although certain hard-core peripherals are limited to specific I/O pins (as in normal microcontrollers).

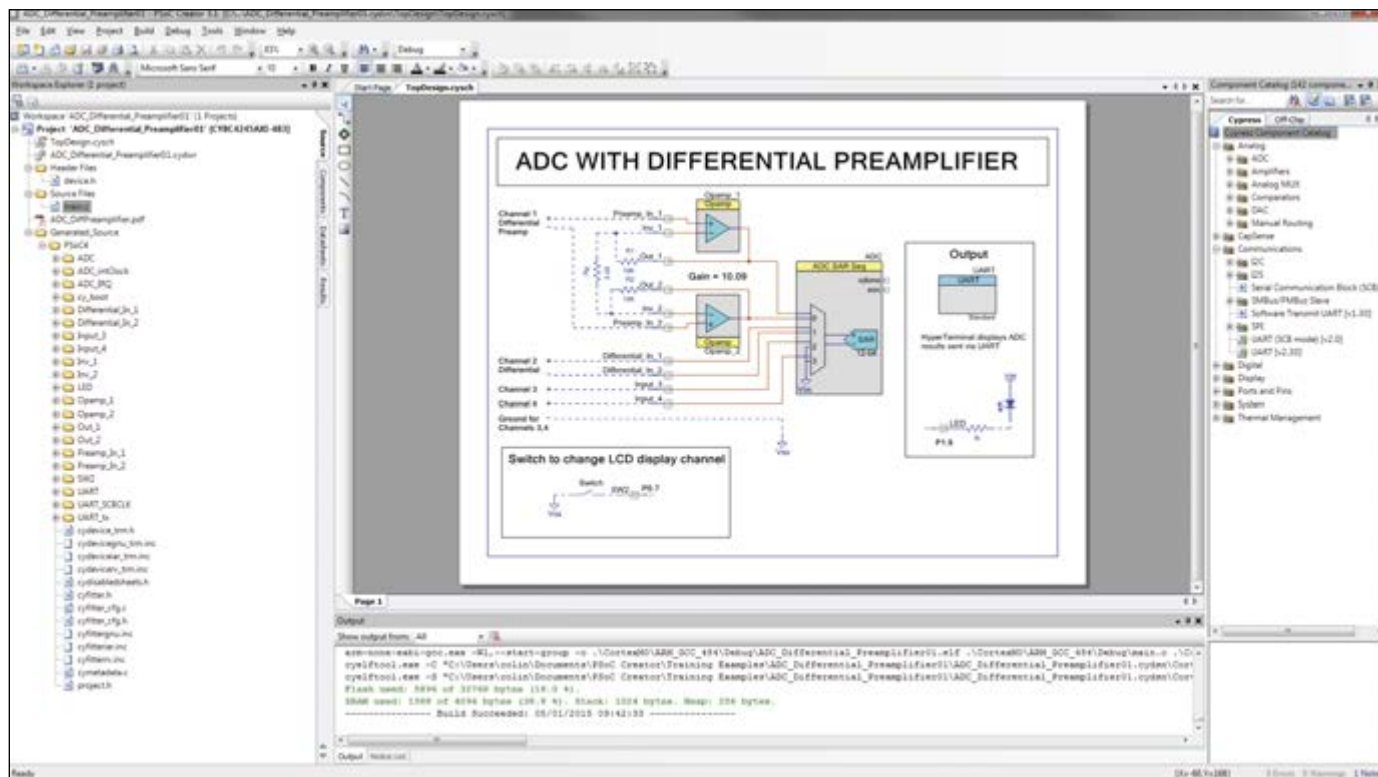
the processor core, the design of the PSoC system is more complicated. This was done to optimize the PSoC system for certain common applications, but comes with the downside that simply porting even a simple Verilog design will quickly exhaust the programmable logic fabric in the device.

If we wish to delight Cypress's marketing folks and stick with their naming scheme, we can refer to what they call universal digital blocks (UDBs). The PSoC 4200 series devices contain four of these UDB blocks, as shown in **Figure 2**. Each UDB contains eight "macrocells," where the macrocell is similar



**FIGURE 2**

The PSoC device has between 4–24 Universal Digital Blocks (UDBs), a summary of the included features shown here. The "macrocell" is similar to a regular CPLD, the "status/control registers" used to simplify setting peripheral options, and the "datapath" is effectively an 8-bit ALU with support components.



#### PHOTO 1

This shows an example project, with both analog and digital blocks being placed on the schematic-based workspace. Clicking on a block brings up a graphical configuration utility. Note the schematic also includes 'external' components such as resistors, LEDs, and switches. They are used purely for reference and are not implemented on the PSoC system.

to those found in a CPLD. This would be a total of 32 macrocells across the device, about the same as you would find in one of the smallest CPLDs from Xilinx for example. Within each macrocell you have a number of logic elements along with a single flip-flop – which at first might suggest you have only 32 bits of storage across the programmable logic, but this is where the UDB introduces a unique feature to differentiate it from standard CPLD devices.

The most prevalent is what Cypress calls the "datapath" block, which is effectively a simple 8-bit arithmetic logic unit (ALU). This ALU can be programmed to perform a few common functions such as shifts, comparisons, additions, or even CRCs. The

ALU is wrapped with several registers allowing you to operate the ALU as an accumulator, or to have a small 4-byte FIFO on the input and output of the ALU.

These blocks can be interconnected to use as a 16-, 24-, or 32-bit datapath block. On the PSoC 4 device there is only four UDB blocks, meaning you could only have a single 32-bit datapath block, for example. But the PSoC 5LP device has up to 24 UDB blocks, giving you much more room to implement programmable logic functions, especially if you want to deal with wider data-paths. The PSoC 5LP also adds some additional resources such as a digital filter block, giving you even more flexibility in your programmable logic design.

You should start to get the feeling that the UDB isn't designed as a replacement for a FPGA-based design. There is simply not enough programmable resources to implement hardware designs you have pushed into a FPGA. But this is truly a strength of the system—it gives you a low-cost option for when you need just a little bit of programmable logic, and avoids trying to oversell itself as an FPGA replacement.

Finally, the UDB provides a simple interface to the microcontroller. Most of the UDB registers and memory can be read/written through the processor core – for example you can read or write to the registers that are part of the datapath block (i.e., surrounding the ALU).

#### ABOUT THE AUTHOR

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





Each UDB also provides several dedicated 8-bit registers that are similar to “standard” peripheral registers a microcontroller expects—that is, the control, status, and interrupt registers. This simple interface matches well with the limited resources in the Cypress PSoC programmable logic system. Let’s see how these can be added to our processor core.

## VISUAL PROGRAMMING

The design software (called PSoC Creator) focuses on a graphical configuration of the hardware blocks. These blocks can be hard-core peripherals with fixed functions, or soft-core peripherals implemented in the UDB blocks. Building the design causes the system to generate Verilog code for the UDB blocks and interconnect, and then synthesizes this design. Once a hardware design is generated, the API is also built for your specific system and added to your software project file.

**Photo 1** gives you an overview of the GUI, showing the hardware design process for configuration of hard-core, soft-core, and analog blocks. This same GUI also includes the code editor, compiler, and debugger.

The tools make it almost seamless as to whether you are placing a soft-core block

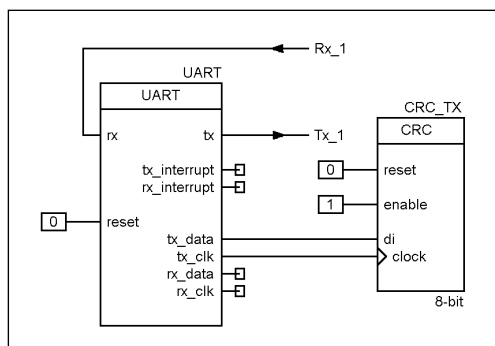


FIGURE 3

(i.e., one using the UDB resources) or a hard-core peripheral. The blocks can be configured through the GUI to avoid hunting down specific register settings, however a full API is still provided. This means you don’t have to worry about the GUI tools only containing certain use-cases or otherwise getting in the way of your standard register-based configuration.

To better understand the limitations of the programmable logic subsystem, let’s look at a simple design. I wanted to avoid some of the more obvious programmable logic extensions (e.g., adding extra PWM channels) and instead make a little more advanced peripheral. In this case I’ll try adding automatic CRC generation to a UART peripheral.

# Circuit Cellar 2014 Digital Archive

With this digital subscription, you have access to all 12 issues of Circuit Cellar 2014 from any computer or tablet at anytime. Readers can explore project ideas, bookmark pages, and make annotations throughout each issue.

## Circuit Cellar 2014 CD

CD includes 12 issues of Circuit Cellar in PDF format along with related article code.

Order yours today

**cc-webshop.com**



```

CRC_v2_40 CRC_TX (
    .di(Net_6553),
    .clock(Net_6538),
    .reset(Net_6555),
    .enable(Net_6556));
defparam CRC_TX.Resolution = 8;
defparam CRC_TX.TimeMultiplexing = 0;

assign Net_6556 = 1'h1;
assign Net_6090 = 1'h0;

```

**LISTING 1**

The resulting Verilog code for the CRC block connection is shown here. The net naming scheme is automatically generated by the tool.

**FUN WITH AUTO-CRC**

Having automatic CRC calculation would simplify the implementation of a serial protocol requiring either a CRC check on receiving data, or sending a CRC on outgoing packets. While the hard-block peripheral doesn't provide this function, I can use soft-core peripherals to give me this custom version of the peripheral. Since CRC has many options regarding the specific implementation (e.g., polynomial, bit ordering, initial value, and final processing), it is well suited to a soft-core peripheral defined at implementation time.

The schematic design is shown in **Figure 3**. Here I'm using a soft-core UART block, as one of the features of this soft-core block is the ability to mirror the transmitted or received data stream to another block. We are using this mirror to feed our CRC block with the outgoing data stream.

You'll notice there is only a single CRC block, rather than a CRC on both transmit and receive. The smaller PSoC 4 device was unable to fit two soft-core 8-bit CRC blocks along with the full-duplex soft-core UART, as this required more programmable resources than available in the device. Upgrading to the PSoC 5LP would have moved from four to 24 UDB blocks, so if you plan on doing more than a few simple tasks in the programmable logic

the PSoC 4 will likely be too small for you.

Note I could have switched the soft-core UART to be half-duplex, where it can still transmit and receive, but requires a software API call to switch between transmit and receive mode. This half-duplex UART takes fewer resources, and I could then fit two 8-bit CRC blocks alongside the soft-core UART.

A brief snippet of the automatically generated Verilog source is given in **Listing 1**. You can see in this case the CRC block has automatically named nets connecting it to either the fixed values or the UART block (the UART block is not shown). Refer to Figure 3 for the schematic representation of this design.

Building the hardware design generates the software framework, which I then added appropriate calls to drive the UART and CRC block. The resulting code to drive this is shown in **Listing 2**. The only trick in this is I had to mirror the bit order of the resulting CRC before transmitting. Your gut instinct here would be to modify the underlying Verilog to reverse the bit order of the output register to eliminate this step. In this case the output register is actually part of the hard-core datapath block (i.e., ALU)—meaning, reversing the bit order requires either adding an additional control register or trying to use the datapath block to perform the reversing operation.

Using the soft-core blocks from Cypress makes utilizing the programmable architecture simple. While expanding or modifying them isn't too difficult, you will need a bit of a shift in thought-process to use the UDB blocks including the datapath to their best potential.

**THE PSoC DRAWER**

Compared to even a small FPGA, the PSoC series has a minute amount of programmable logic. But the sub-\$3 price of the PSoC 4 makes it clear this isn't designed for solving the same problems that might force you to use a soft-core processor in an FPGA.

While the larger resources of the PSoC 5LP make it possible to achieve slightly more complex peripherals, the specialized device architecture means you cannot simply port an existing Verilog design onto the programmable logic system.

This leaves the PSoC well suited to tasks requiring minor tweaks to peripherals—think nonstandard length SPI registers, special protocol interfaces, special calculators such as CRC or other frame check sequences, or just standard glue logic. The simple graphical configuration tool along with the ability to reach out and modify the underlying Verilog configuration leaves me feeling more confident of the long-term usability of this device.



circuitcellar.com/ccmaterials

**SOURCES**

PSoC 4200

Cypress Semiconductor Corp. | [www.cypress.com](http://www.cypress.com)

```
#include <project.h>

unsigned char reverse(unsigned char b) {
    b = (b & 0xF0) >> 4 | (b & 0x0F) << 4;
    b = (b & 0xCC) >> 2 | (b & 0x33) << 2;
    b = (b & 0xAA) >> 1 | (b & 0x55) << 1;
    return b;
}

int main()
{
    uint8_t crc;

    /* Setup hardware modules */
    UART_Start();
    CRC_TX_Start();

    /* Write DE, AD, BE, EF */
    UART_PutChar(0xDE);
    UART_PutChar(0xAD);
    UART_PutChar(0xBE);
    UART_PutChar(0xEF);

    /* Wait until transmit complete (meaning CRC calculation should
    also be done) */
    while(((uint8_t)~UART_ReadTxStatus() & UART_TX_STS_COMPLETE) !=
    0u);

    /* Read CRC from hardware module */
    crc = CRC_TX_ReadCRC();


    /* Swap bit order */
    crc = reverse(crc);

    /* Append to packet */
    UART_PutChar(crc);
}
```

**LISTING 2**

This shows some example C code for driving the UART and CRC blocks, those blocks having been automatically generated by the Cypress tools.

Hopefully, this article has given you a brief overview of an interesting processor architecture. As a final note, it is worth mentioning that Cypress has released a \$10 development board (part number CY8KIT-059), which is fitted with one of the larger PSoC 5LP devices. Now you can keep a PSoC in your drawer for the next time you need just a little bit of programmable hardware (digital or analog).

Be sure to check out [ProgrammableLogicInPractice.com](http://ProgrammableLogicInPractice.com) for the full project files. I've also posted more snippets of the automatically generated Verilog code to give you a better idea of what the output of these tools look like. 

## Introducing The Future of USB



### USB Type-C™ Protocol Analyzer

- **USB 2.0**
- **Power Delivery 2.0**

The Mercury™ T2C is the industry's first affordable USB and Power Delivery protocol analyzer. Featuring the new "Type-C" connectors, the Mercury™ T2C includes all the cables to interface with both legacy and Type-C devices.

- **Power Delivery 2.0**  
Capture and debug PD messages (BMC) and Type-C link states.
- **USB 2.0 Capture and Decode**  
Automatically decodes upper-level protocol events.
- **Real Time Triggering**  
Isolate specific USB and PD protocol events with hardware-based triggering.

	Mercury T2C Power Delivery 2.0	Mercury T2C Standard USB 2.0
Type-C connectors and cables	✓	✓
USB 2.0 / 1.1	□	✓
Power Delivery 2.0	✓	□
Recording Memory	256 MB	256 MB
Event Triggering	✓	✓
List Price (US)	\$995	\$995

□ may be added by upgrading

[teledynelecroy.com](http://teledynelecroy.com)

800-909-7211 or 408-653-1262  
psgsales@teledynelecroy.com

 **TELEDYNE LECROY**  
Everywhere you look™



## EMBEDDED IN THIN SLICES

# The Internet of Things (Part 2)

## Choosing a Wireless Carrier for Your Embedded System

COLUMNS

Bob started this article series by examining a few different ways to connect an embedded system wirelessly to the Internet. This month he covers a few options for selecting a carrier for an embedded device and covers topics such as: OTA technologies, mobile virtual network operators, data plans, and project budgeting.

*By Bob Japenga (US)*

Several years ago, I was somewhat startled when I saw Dan Hesse, who was then the CEO and president of Sprint, on television talking about how Sprint was positioning itself to serve the machine to machine (M2M) community in the coming years. Sprint was going to be the place to go if you were a M2M developer. As embedded systems designers, it is not very often that we are the target of prime time television advertisements. I suspect Dan was not really targeting us. There are too few of us. He was targeting investors. But the point was made. The personal phone market had saturated in the US and M2M wireless connectivity with the advent of the Internet of Things (IoT) offered an almost unlimited growth potential in the US and worldwide. And the wireless carriers are critical to making that happen.

I began this article series by discussing some of the ways our company has connected our embedded systems wirelessly to the Internet. The current buzz in our industry is the IoT. Because of our lack of experience in satellite M2M, I will not be discussing satellite options. This may be an applicable wireless technology for some of your products, but we

must take things in thin slices. This month, I want to talk about one of the critical first steps in designing your embedded IoT system: choosing the wireless carrier and the wireless technology that works best for you.

### TERMINOLOGY

Before we start, let's define some terms:

*Carrier:* Okay, so most of us know about T-Mobile, AT&T, Sprint, Verizon, and maybe even Vodafone. These are wireless carriers.

*Wireless or Over-the-Air (OTA) Technology:* Here we get into some marketing jargon, but OTA technologies include GSM, LTE, EDGE, and so on. They are marketed as 2G, 3G, and 4G. It does make me wonder what the Fifth-Generation OTA technology will do.

*Radio Access Technology (RAT):* Okay, I haven't used this term, yet but many confuse the wireless or OTA technology with RAT. Here are the most prevalent RATs: time-division multiple access (TDMA); frequency-division multiple access (FDMA); code-division multiple access (CDMA); and orthogonal frequency



division multiple access (OFDMA). All of these RATs are separate ways to put multiple calls on the smallest available radio frequency spectrum. FDMA uses different frequency bands for each call. TDMA provides different time slots to the different calls. CDMA uses statistical spread-spectrum techniques to put each call in a particular frequency. OFDMA is not easily defined in a single sentence. Suffice it to say, it assigns subsets of subcarriers to each call. Remember that the goal is to put the most data through the smallest frequency band.

## GEE-WIZ

If you listen to the marketing pitch for wireless service providers, you hear them talking about 3G and 4G. Even after several years and several successful deployments of wireless systems, my head sometimes spins when all of the various wireless technologies are described. **Table 1** is my attempt at a concise table for OTA technologies.

The classification is important to us as developers because we need to choose a carrier and the OTA technology (sometimes called “air interface”) that fits our product specifications including recurring costs, nonrecurring costs, life of the product, and target location. Let me see if I can tease out how this can get complicated real fast.

## TARGET LOCATION

Where is your design going to be used? If it is only in the US, then you have a wide number of options. Does the product move around? If you want your target to be used anywhere in

the world, then your options just reduced. You can: pick an OTA technology that is available worldwide (UMTS); design in a module that supports multiple OTA technologies; or design your system to support separate modules for each of the OTA technologies that you want to support. **Table 2** provides a summary of these options.

Fortunately, most of the module manufacturers create pin-for-pin compatible parts which are, for the most part, software compatible. If you take this option, you must certify the two different OTA technologies. Modules that support multiple OTA technologies are much more expensive and thus increase your recurring costs. But you only certify once. Certification costs are very large and certifying two different OTA technology modules can be time consuming and expensive (non-recurring costs). Also, most modules are only certified for three years so that nonrecurring cost becomes a recurring cost before you know it.

We like to design it once, certify it and then ship it for 10 years. This time limitation on certification is a serious problem for all embedded system designers like us. We hope this will change.

## PRODUCT LIFE

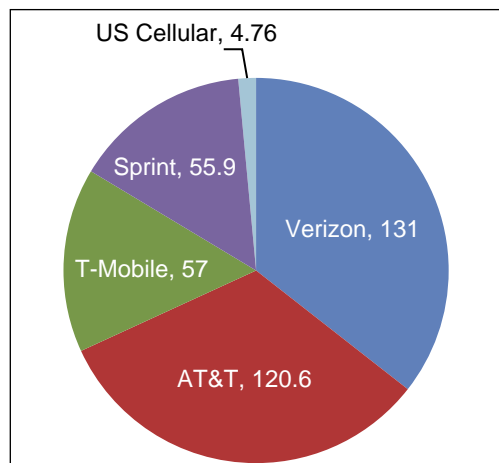
This is where the expected life of the product comes into play. M2M is very different from cell phone users who only keep their phones for one to two years. Two of our IoT devices have less than a five-year life because of their unique medical usage. For

**Table 1**  
OTA technology summary

OTA Technology	OTA spelled out	Marketing Classification	RAT	US Carriers
GSM	Global System for Mobile Communications	2G	TDMA	AT&T,T-Mobile
GSM GPRS	GSM General Packet Radio Service	2G+ (2.5G)	TDMA	AT&T,T-Mobile
GSM EDGE	Enhanced Data rate for GSM evolution	2G+ (2.75G)	TDMA	AT&T,T-Mobile
UMTS	Universal Mobile Telecommunication System	3G	W-CDMA	AT&T,T-Mobile
1xRTT	Radio Transmission Technology	3G	CDMA (CDMA2000)	Verizon, Sprint, US Cellular
EV-DO	Evolution-data optimized	3G	CDMA	Verizon, Sprint, US Cellular
LTE	Long Term Evolution	4G	OFDMA	Verizon, Sprint, US Cellular, T-Mobile
LTE-Advanced	Long Term Evolution Advanced	4G	OFDMA	AT&T

**Figure 1**

Millions of subscribers



such units, choosing an OTA technology is not constrained. But one of our IoT devices has a targeted 20-year life and that dramatically complicates your decision.

Here is the problem. In 2012, AT&T announced that they were sun setting their 2G network starting January 1, 2017. When we create a design with a long life, we need to choose the technology with the highest probability of surviving for that length of time. But that is not trivial. Think of the developers of OnStar, which was based on an early cell technology. Even though the US federal government forced the carriers to extend the life of that technology to 2008, the carriers eventually shut down the system used on millions of cars. If you choose a 3G OTA technology today, will it be available in 15 years? Only time will tell. I think that the M2M explosion will provide the revenue stream for the carriers to maintain these long after the phone users have stopped using 3G. But remember, I am an engineer not a financial guy.

### CAN YOU HEAR ME NOW?

Once you have some idea as to the OTA technologies that are acceptable to your price point and location, how do you choose a carrier? First, in the US, you need to understand the three types of carriers that

you can go with.

**Cell Providers:** The first type is those providers that actually build cell towers and receive and transmit RF to the M2M device. Of course, in the US, the big five are AT&T, Sprint, T-Mobile, Verizon, and U.S. Cellular. **Figure 1** shows their market shares by subscribers in the US. There are a host of very small providers with very local coverage that may work fine for one of your designs if your target is local, but we have no experience with them.

**Mobile Virtual Network Operator (MVNO):** We live in a day of many "virtuals." So it should not surprise you that there are scores of MVNOs. These are companies that lease wireless data services from the big five. Each is dependent upon the wireless carriers for everything. By choosing an MVNO, you have more leverage in negotiating your data plan because they are smaller. This is less of an issue now since the major players want your M2M business.

**Hybrid Carriers:** There is one carrier, Aeris, which is somewhat in a class by itself. In the US, it has its own CDMA network (3G 1xRTT and EV-DO), which is specifically designed for M2M applications. In the rest of the world, they are an MVNO leasing a GSM network (2G and 3G) from the local provider. This is a giant advantage for companies that don't want to negotiate data plans with all of the carriers worldwide. Each country can have its own separate wireless carrier. They have done the leg work for you. Of course the data plan costs will reflect this extra work, but we have found that for most small companies that we work with, the effort of negotiating a data plan with each of the GSM cell providers worldwide and obtaining the necessary certification on each network is a significant block to worldwide deployment. We have found that Aeris makes that simpler and the cost penalty is not that significant for our products.

### FOLLOW THE MONEY

You don't need a Deep Throat to tell you that you need to follow the money when you consider a carrier. The carriers are getting much more competitive with their data plans for M2Ms than when we first started. Initially, we could not get companies like Sprint to even talk with us. On early M2M designs, we could not even get them to return our calls. Now, things are very different. The carriers know about our concerns. Data plans for M2M devices can be very, very cheap if your data is very, very small. All of the carriers we have worked with support data usage pooling across your fleet. Here is what that means for you. Let's say that you only utilize 600 KB of data each month and you have a 1 MB-



#### 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](mailto:rjapenga@microtoolsinc.com).



Option	Advantages	Disadvantages
Multi-carrier module	<ul style="list-style-type: none"> <li>• One part to stock</li> <li>• Single Test procedure</li> <li>• Single certification</li> </ul>	<ul style="list-style-type: none"> <li>• Higher recurring costs</li> <li>• Slightly higher certification costs</li> <li>• Multiple carriers to negotiate with</li> </ul>
Worldwide OTA technology (UMTS) module	<ul style="list-style-type: none"> <li>• One part to stock</li> <li>• Single Test procedure</li> <li>• Single certification</li> </ul>	<ul style="list-style-type: none"> <li>• Higher data plan OR negotiate with many worldwide carriers</li> <li>• Not necessarily the best coverage in the US</li> </ul>
Separate pin and software compatible module	<ul style="list-style-type: none"> <li>• Negotiate the lowest recurring data plan cost</li> <li>• Can cost less based on the market</li> <li>• Prepares you for future module changes</li> </ul>	<ul style="list-style-type: none"> <li>• Multiple SKU's</li> <li>• Multiple parts to stock</li> <li>• Multiple test procedures</li> </ul>

per-month data plan. But every six months you update your software OTA. The software update takes 800 KB of data. You could update half your fleet one month and the other half the other month and stay within your 1 MB-per-month data plan.


Another area that used to be a pain has to do with device deployment and data plans. Previously, if we wanted to test the cell modem during manufacturing, we had to pay the monthly charge even if it sat in a warehouse for three months. Now the carriers are accommodating. Each has its own way of doing it, but most provide something that is workable with a manufacturing process that needs to test the cell operation before it ships. But you do need to find out what they offer so you can tailor your manufacturing process to their data plan.

## CHIPS AND SALSA?

In choosing your carrier and OTA wireless technology, you need to factor into your cost budget the antenna and the module itself. We've found that the modules for 2G and 3G typically are significantly cheaper than those for 4G and LTE. For LTE, you are required to have two antennas. That doubles your recurring cost right there. (By the way, I think that will get relaxed in the near future. For most M2M, we don't need that second antenna. It is used to increase throughput.) Your certification costs can also be significantly higher for the 4G technology parts. One quote we got was 3.5 times the cost of certifying a 1xRTT 3G solution. When you buy a module, you need to pick which carrier you want it to work on. For example, we buy Sierra Wireless modules for the AT&T network. We buy u-blox modules for the Verizon network. We also buy u-blox modules for the Aeris network.

## JOIN THE REVOLUTION

Designing a device that can join the IoT

revolution is not for the faint of heart. If you are new to it, there is a lot to learn in a very short period of time. And of course the sales people assume that you know all of this alphabet soup that is involved in taking the first steps in selecting an OTA technology and a carrier. As always, we have taken it in thin slices. And so can you. Next time, we will look at yet another factor in being part of the IoT revolution. Of course, only in thin slices. 

**Table 2**

Cell modem types



**TRACE32® - supports all**

64-BIT (ARMv8)

Cortex-A53 Cortex-A57 Cortex-M0 Cortex-M0+ Cortex-M1 Cortex-M3 Cortex-M4 Cortex-M7

APPLICATION

ARM926 ARM946 ARM968 ARM7

ARM1136 ARM1156 ARM1176

CLASSIC

Cortex-R4 Cortex-R5 Cortex-R7

REAL-TIME

SC000 SC100 SC300

SECURCORE

**ARM** CONNECTED

**LAUTERBACH** DEVELOPMENT TOOLS

[www.lauterbach.com/1553](http://www.lauterbach.com/1553)

## THE DARKER SIDE

# Vintage Electronic Calculators

COLUMNS

This month Robert takes a break from his usual focus on RF technologies to write about one of passions—vintage electronic calculators. He provides a few examples from his collection and even pops the hood on a few of them.

*By Robert Lacoste (France)*



**PHOTO 1**

Take a look at the 1968 Sharp Compet22 and its wonderful Nixie display.

**W**elcome back to the Darker Side. A couple of months ago, Jeff Bachiochi wrote an article about ladder logic programming and illustrated his article with pictures of vintage mechanical calculators (Circuit Cellar 297, "Ladder Logic, 2015"). While reading the article, I also happened to be looking for a subject for this column. Fortunately, some of my neurons fired and I got a great idea—vintage calculators!

This month I will not present another radio frequency technique or signal processing algorithm. I will just talk about vintage calculators. Why? A few reasons.

First, this article will appear in August and summer is a nice time for such a refreshing topic. Second, this is my 50th Darker Side column, so I want to celebrate the achievement with something a little different. (Yes, the first Darker Side was published back in August 2007. Prior to that, I published a couple of project-oriented articles and winning design contest entries in Circuit Cellar, but that's another story. Fifty Darker Side articles, this is something, right? I must admit that I wouldn't have bet to last so long

when Steve Ciarcia asked me to start this column.) And lastly, I must confess that I'm addicted to vintage calculators, which I've collected since I was a teenager and more seriously during the last 10 years. I now own around 500 of them (which is probably a ridiculous achievement compared to serious collectors).

My collection comprises mainly electronic pocket calculators from the 1970s and early 1980s. I could have collected cars, stamps, or wine bottles, but I collect calculators. Let's be honest, there are a couple of positive sides to it. Firstly, vintage calculators don't cost much. Except for some very rare models, vintage calculators are not perceived as high-value items (like watches) and are often sold at flea markets for a couple of dollars. OK, the prices are starting to be higher on eBay, but that's life. Secondly, they don't take too much real estate. Even if I have some large and heavy mechanical models, my full collection fits in a closet and this keeps my wife Isabelle happy. Lastly, and especially for an electronic designer like me, vintage calculators are really fun.



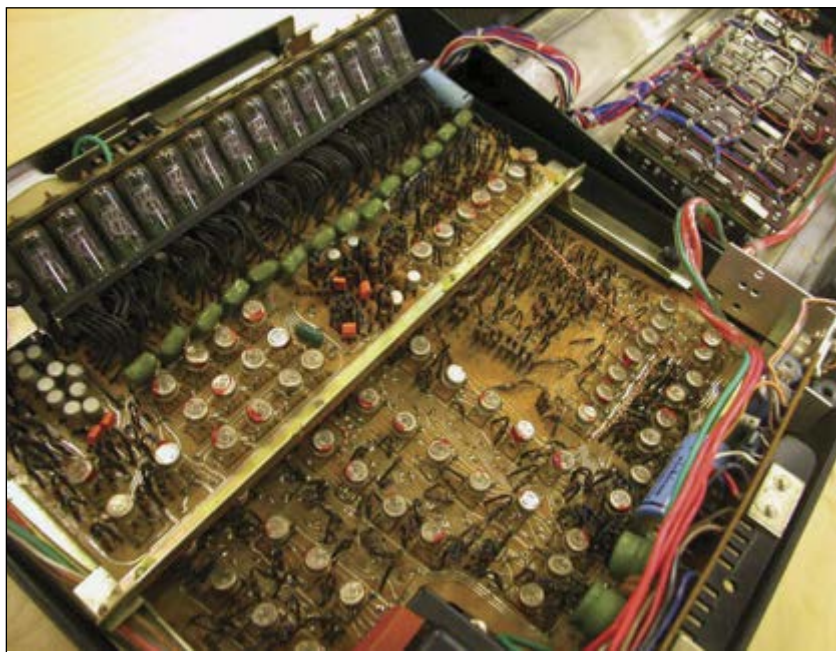
## EARLY ELECTRONIC CALCULATORS

In this article I will not present a formal history of calculators. (There are a few good books on the subject. I list one at the end of this article.) My aim is just to show you some samples from my collection, targeting early years pocket electronic calculators. As *Circuit Cellar* is about electronics, I'll also share some images of my calculators' electronics.

Let's start with the ancient world. Prior to electronic calculators, we had mechanical devices. As you might know, the first working mechanical calculator, La Pascaline, was designed by the French mathematician Blaise Pascal in the 17th century. However, mechanical calculators started to be manufactured only two centuries later. They became commonly used (at least for accountants) in the early 1900s. Skipping some early electromechanical devices and room-sized vacuum-tube systems, the first actual bench top electronic calculators appeared in the early 1960s. I'm not lucky enough to have one of those early models (e.g., ANITA, Friden EC130, and Sharp CS-10A), but my oldest electronic calculators still date back to late 1960s.

Check out my 1968 Sharp Compet 22 in **Photo 1**. In particular, look at the display, which is using my favorite technology—Nixie tubes. Such tubes are made of 10 neon digits stacked on each other, giving a nice 3-D experience. Very nice. This Sharp calculator is in fact part of electronics history. It was the second calculator worldwide to use metal-oxide silicon (MOS) integrated circuits. The first to do so was another Sharp model launched some months earlier. Refer to **Photo 2** for a look inside. No less than 83 TO100 integrated circuits and an amazing number of discrete components! This calculator has also an extension connector on the back, which allowed users to connect the so-called "CSA-12 memorizer unit," making it one of the first devices to support a kind of macro commands. Last but not least, it calculates on 24-digit numbers. Of course, only half of them are displayed at a time. Just for the fun I measured its calculation speed and found about 500 ms for a division. The nice thing is that all the digits become crazy during the calculation. Someone forgot to add a blanking signal.

Before jumping forward to pocket calculators, just have a look at the Addo-X 9354J in **Photo 3**. It is a rebranded version of the Sharp QT-8, nick-named "Micro-Compet" (1969). This model's development marked another milestone in electronics history. It was the first calculator built with large-scale integration circuits (LSI). Thanks to the advances of semiconductor technology,



**PHOTO 2**

Internally, the Sharp Compet22 is built around 83 MOS early integrated circuits and hundreds of passive components.



**PHOTO 3**

The Sharp QT-8B is one of the first LSI-based calculators. Here it is rebranded as the ADDO-X (1969). Look at its eight-segment display.



**PHOTO 4**

Here is another piece of history, the Canon Pocketronic and its tape-printer display.

**PHOTO 5**

The HP35 was the first scientific calculator. It was HP's first pocket calculator.

**PHOTO 6**

The first Texas Instruments pocket calculator, the TI2500, looks a little dated.

it required only four 42-pin ceramic ICs. Each IC, manufactured by Rockwell, contains no less than 900 MOS transistors. This may seem ridiculous compared to the 5.5 billion transistors in Intel's latest Xeon processors, but it is still impressive! Some months later, the Japanese team that developed the QT-8 worked for a new company named Busicom. They contracted with two small US companies to develop an IC for a new calculator. These US firms were Mostek and Intel, and the chip became the 4004, the first single-chip microprocessor.

Let's stay with the Sharp QT-8 calculator. Another interesting feature is its display. Look closely at Photo 3. The display is neither Nixie tubes nor seven segments. It uses very exotic fluorescent tubes based on an eight-segment arrangement! This provides nicely stylized digits, even if the zeros are far from readable. I don't know any other device made with such a display, except some derivatives like the pocket-sized Sharp EL-8 launched the next year.

## POCKET CALCULATORS

Let's move on to the first "pocket" calculators. Back in 1965, a team of Texas Instruments guys, led by the well-known Jack Kilby (no less than the inventor of the IC in 1958), started to work on a hand-help calculator. The calculator, named "Cal-Tech," was more of a demo system than a manufactured product. However, the concept was sold to Canon (Japan), and used to design the Pocketronic calculator (fall of 1970). There are very few Cal-Tech units worldwide, but I do own a Pocketronic (see **Photo 4**). Once again the display is the most interesting part of this calculator. It isn't actually a display, but a small horizontal-side thermal printer. The printed result was simply visible through a window before exiting the calculator on the left side.

From 1970 to 1972, the pocket calculator market really exploded with a long list of manufacturers: Sharp, Sanyo, Bowmar, Craig, Commodore, Brother, Casio, Rapid-Data, and a few others. As the market was starting to be significant, two heavy new players came to the playground in 1972. The first was Hewlett-Packard with its HP35 (see **Photo 5**). Another important milestone, this was the first pocket scientific calculator worldwide. By the way, do you know why it was named the HP35? Just count the number of buttons on its keyboard. Hewlett-Packard then kept the innovative edge for years, with its first financial calculator (HP80, 1973), the first calculator with a "shift" key (HP45, 1973), the first magnetic-card programmable calculator (HP65, 1974), the first alphanumerical calculator (HP41C, 1979, my favorite), and a few others. HP was renowned for its reverse-polish notation (RPN), which proved to be significantly faster to use than standard algebraic entry systems. In a nutshell an RPN calculator is based on a stack, and rather than entering "1+2=", you have to key in "1 Enter 2 +". This may not seem obvious, but this is far easier on more complex calculations. Just as an example, if you want to calculate  $12 \times (\log(10 + 5) + 1)$ , you would do "12 enter 10 enter 5 + log 1 + \*", no parenthesis needed. RPN aficionados are still numerous. That's probably why HP re-manufactured some RPN units these last years.

Let's go back to 1972. The other company who started to make their own calculators that year was Texas Instruments (TI). TI was actually building chips for plenty of other manufacturers, but its first branded product was the TI2500, which launched in June 1972. Honestly, it was fat and not really competitive even in 1972 (see **Photo 6**). Fortunately, TI had the resources to quickly develop more



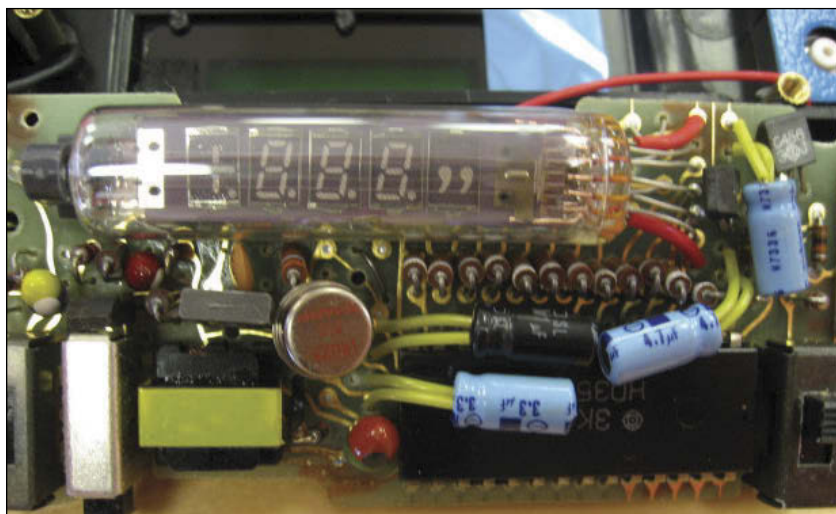
**PHOTO 7**

This is the National Semiconductors NS600. Can you find a decimal key?



**PHOTO 8**

The Sharp EL120 was an attempt for a low-cost unit. Interestingly, the white button behind the display adds one to the result.



**PHOTO 9**

The display of the Sharp EL120 is just three digits long. Can you imagine that?



**PHOTO 10**

The Sharp EL808 has one of the first LCDs.

advanced products like the SR10 (the first TI with the square root function), SR50 (the first TI scientific calculator), and SR56 (the first TI programmable calculator). This culminated in 1976 with the massive widespread of the TI30, which sold for no more than \$25, making it a huge success for scholars. According to Wikipedia, 15 million TI30 units were sold from 1976 to 1983.

## EXOTIC CALCULATORS

Now let's have a look at some calculators that are for me significantly exotic. The first

one could be the National Semiconductors NS600, which dates back to 1973 (see **Photo 7**). Do you see anything strange? Firstly, it doesn't have any equal key, as it uses a kind of RPN. Nor does it have an Enter key, which makes it quite difficult to use. And there isn't a decimal point key. The calculator works only on integers! I'm not sure how it ever sold.

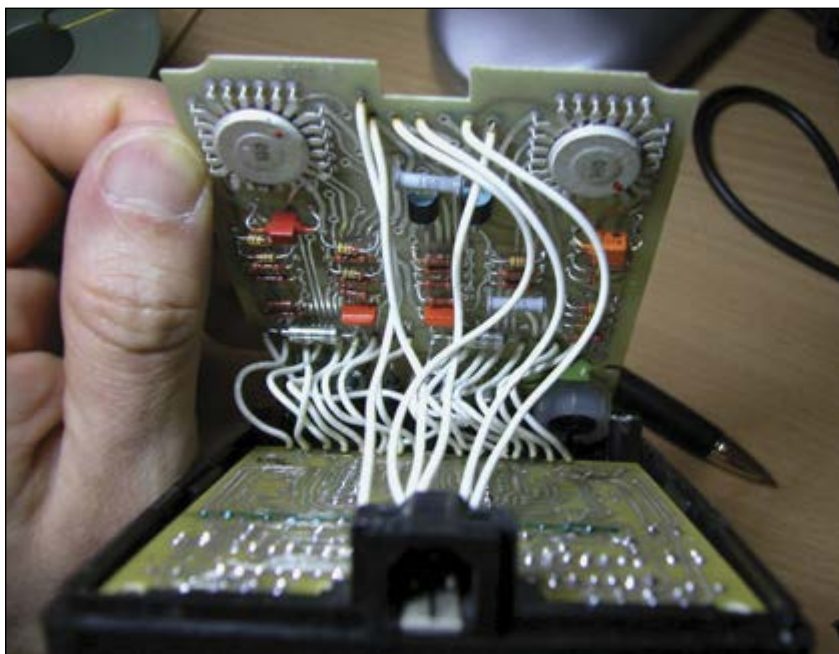
Another interesting example is the 1973 Sharp EL120 (see **Photo 8**). This was very probably an attempt to make a low-cost calculator. Its display is only three digits! It can calculate eight-digit numbers, but you have to write down the result three digits by three digits. Very easy! **Photo 9** shows you what's inside, with its custom glass-tube display and capacitors with strangely long legs.

The next year, in 1974, the first liquid crystal displays (LCDs) appeared. Of course, they grabbed the market very quickly thanks to their low power requirements. However, the units from that era weren't as sexy as today's current ultra-thin LCDs. The Sharp EL808 in **Photo 10** was produced a couple of months after the first actual LCD-based calculator (aka the Sharp EL805). They both used the same LCD, which is a very low contrast so-called COS display. Due to the poor contrast, the calculator required a backlight and a flap to hide the display from ambient light.

Of course, the United States and Japan were not the only countries to design and manufacture calculators. As an example of, well, more exotic technology, look at **Photo 11**, which shows the printed circuit boards in a Russian (sorry, USSR at that time) calculator—The Elektronika C3-15 (1975). As far as I know it is the first Russian pocket scientific calculator, and I love its integrated circuits and long wires. I'm sure electromagnetic compatibility was less of a concern then than it is today!

In the 1970s, software developers were not programming in Java and XML. They were significantly closer to the actual hardware, as assembly language was ubiquitous. That's why Texas Instruments produced a variant of the TI30 dedicated to software developers, the TI Programmer (see **Photo 12**). Launched in 1977, it enabled users to calculate in decimal, binary, octal, and hexadecimal. The mathematical operations were replaced by logical functions: OR, AND, XOR, shifts, and so on. In 1982, Hewlett-Packard addressed the same needs with a much more advanced programmer-oriented calculator, the HP16C. This one is quite rare. It originally sold for \$150 and will cost you roughly the same today.

Lastly, I can't resist showing you one of my recent findings, the Sharp EL8048 (see

**PHOTO 11**

Take a look inside the Russian Elektronika C3-15. Very exotic, isn't it?



**Photo 13).** I don't know what the designer was thinking of, but apparently in 1979 there were some users who couldn't decide between an electronic calculator and an abacus.

## TECH EVOLUTION

Vintage calculators are great demonstrations of technological progress. The first electronic calculators were developed only 10 years after the invention of the integrated circuits. These machines, around 1969, were lab projects. They were bulky, ultra-expensive, four-operation machines. Only seven years later, in 1976, millions of \$25 pocket scientific calculators were selling worldwide. Few technologies moved so fast. Moreover, calculators actually pulled the technology forward. It is the demand for electronic calculators which accelerated the development of low-cost, large-scale integration circuits, and ultimately the invention of the microprocessor. It is the calculator business that forced designers to find innovative display technologies, from Nixie tubes to vacuum fluorescent devices (VFD) to LEDs to LCDs.

Things are different today. Display technology is pulled by mobile phones, and calculator makers reuse it. As an example,



Google and look at the latest HP Prime graphing calculator. Where do you think its multi-touch color 3.5" 320 x 240 TFT display

**PHOTO 12**

The TI Programmer was intended for software developers with multi-base and logical operations.

## ROBOBusiness

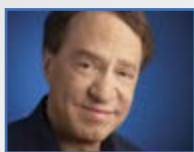
INVEST ► INNOVATE ► IMPLEMENT

September 23-24, 2015

SAN JOSE CONVENTION CENTER

San Jose, CA

### World-Renowned Keynote Speakers



**Ray Kurzweil**

A Director of Engineering - Google  
Co-Founder & Chancellor -  
Singularity University



**Robert High**

IBM Fellow  
Vice President & Chief Technology  
Officer - IBM Watson Group



## Driving Business Transformation through Robotics Innovation

Meet industry leaders, investors and new customers at the forefront of robotics.

### The RoboBusiness Conference Shows You How to Leverage Robotics:

- Identify & Grasp Key Markets & Opportunities
- Apply Proven Business & Innovation Strategies
- Understand Technology Development & Today's Solutions
- Gain Inside Intelligence on Companies & Startups
- Make Critical Connections Through World-Class Networking

**The Expo is the hub for hands-on learning with hundreds of the world's latest robots**

**Save Over 35% - Use Code RBPCC**

**Register Today! • [robobusiness.com](http://robobusiness.com) • 800-305-0634**

## PHOTO 13

Abacus or calculator? With the Sharp EL8048, it's your choice!




comes from? Well, the bad news is that calculators also benefit from the bad aspects of mobile phone evolution. This HP Prime no longer works for months on batteries like my trustee 35-year-old HP41C, but it had a rechargeable Lithium-Ion battery rated for 20 h of normal usage. Similarly the HP Prime no longer starts immediately when you need it, but it requires a long 8-s booting process, even with its on-board 400-MHz ARM9 processor. Does this remind you something? I guess this is progress.

Talking about processors, the vast majority of vintage calculators were based on dedicated chips developed by Texas Instruments, Hitachi, Rockwell, and a few other companies. However, general-purpose

microprocessors were also used, at least in the 1990s. For example, the TI81—the first graphing calculator made by Texas Instruments in 1990—was based on a 2-MHz Zilog Z80. A couple of years later, the TI92 (1995) used a 16/32-bit early processor—the well-known Motorola MC68000—running at a blazing speed of 10 MHz.

Another interesting gizmo is the HP20B. This is a financial calculator launched by Hewlett-Packard in 2008, so you might not find it too sexy. But, dear Circuit Cellar reader, there is something about this calculator that should interest you. It's the first hackable and nearly open-hardware calculator! It has an ARM processor (i.e., an Atmel AT91 variant) and a user-accessible JTAG connector. HP published its full schematics as well as a software development kit. If you are interested, browse to the WP34S wiki site, which explains how to repurpose the HP20B into a full-featured scientific calculator. Maybe could you find an HP20B, or its sister HP30B, and build something more exciting with it!

## WRAPPING UP

Now you all know my secret: I love vintage calculators. And you also know how to make me happy. Do you have an old calculator in your basement? Don't you think it will be happier with all its friends in my collection? Have a look on my personal website ([www.alciom.com/perso/cal](http://www.alciom.com/perso/cal)), which list all of my calculators, and if yours isn't there, then just send me an email—or, well, ship it directly to me! You will have at least all my thanks, and may be some French specialties in return. Of course, if you prefer to start your own collection, I will never blame you. If you look around, and ask your friends and family, you'll likely end up several free calculators, which would be a good starting point. Then you can buy a copy of Guy Ball and Bruce Flamm's excellent book, *The Complete Collector's Guide to Pocket Calculators* (Wilson/Barnett Publishing, 1997), and you'll be ready to launch! Just don't forget to always remove the batteries from your treasures, as leaking acid has led to the death of many calculators. Have fun! 

## ABOUT THE AUTHOR



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



[circuitcellar.com/ccmaterials](http://circuitcellar.com/ccmaterials)

R. Lacoste, Calculator Collection, [www.alciom.com/perso/cal](http://www.alciom.com/perso/cal).

N. Tout, Vintage Calculators Web Museum, [www.vintagecalculators.com](http://www.vintagecalculators.com).

WP34S Repurposing Project, <http://sourceforge.net/projects/wp34s/>.

## RESOURCES

G. Ball and B. Flamm, *The Complete Collector's Guide to Pocket Calculators*, Wilson/Barnett Publishing, 1997.

# ASSEMBLY LANGUAGE ESSENTIALS

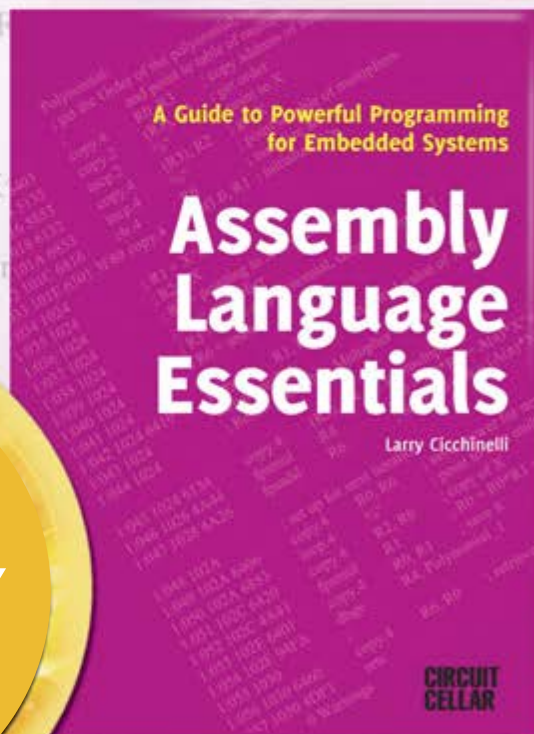
*Assembly Language Essentials* is a matter-of-fact guide that will introduce you to the most fundamental programming language of a processor.

Larry Cicchinelli provides readers with:

- Essential terminology pertaining to higher-level programming languages & computer architecture
- Important algorithms that may be built into high-level languages — multiplication, division, and polynomial evaluation
- Overview of Interrupt Service Routines
- Free downloadable Assembler program...and more!



**ORDER  
YOUR COPY  
TODAY!**





## FROM THE BENCH

# Light Detection and Ranging (Part 2)

## Laser Sensor Exploration

COLUMNS

Last month, Jeff explained how he used an Arduino Mega to trigger a LIDAR-lite and measure its pulse output. This month he takes a closer look at laser sensor exploration and looks into the communication via the I2C interface.

*By Jeff Bachiochi (US)*

**T**he first guide dog training school on the West Coast began in the early 1940s to assist servicemen who had lost their sight in World War II. Prior to the guide dog program, a blind or visually impaired individual's only option was to use a white cane. The cane became an extension of the hand and enabled the user to locate ground-level objects (or the lack of objects) such as a curb or staircase. Today, we have electronic canes with ultrasonic capabilities enable users to sense distance without actually having to touch an object. The technology works in a way similar to how a bat flies at night and catches insects while avoiding the ground and other objects.

Most automobiles have the same needs. Maneuvering through their environment will continue to be a challenge without the power of sight. In the short term, our automobiles might be the only mobile platform large enough to contain the computational and power source requirements to approximate the power of our occipital system. Like the white cane, bumpers are by far the simplest devices we can use

to determine open pathways. Today's auto commercials tout available IR and ultrasonic sensors, which keep vehicles at some minimum distance from the objects they can detect.

Last month, I covered some of the shortcomings of IR and ultrasonic sensors and introduced PulsedLight's LIDAR-Lite laser sensor. Unlike most laser devices, the LIDAR-Lite costs less than \$90 and provides spot distances of an adequately sized target of up to about 60 m. The key is a small field of view (FOV) of 3°. Those familiar with ultrasonics know its FOV of ~60° makes it difficult to use in many situations.

One of the most advantageous features of LIDAR-Lite is its ease of use. The simple trigger input/pulse output pin lets you use the device right out of the box. Once triggered, the user need only measure the pulse width of an output pulse to be able to calculate distance. The application presented last month used this approach using an Arduino Mega to trigger the LIDAR-lite and measure its pulse output. A DOG LCD interface displayed the corresponding

distance. For those of you that are willing to communicate (I<sup>2</sup>C) with the LIDAR-Lite module, we can delve into its expert mode.

## SERIOUSLY

The key to this new technology is PulsedLight's SoC, which handles the transmission signal, storing return data, an I<sup>2</sup>C interface, and a digital processing core capable of correlating the transmit data with received data to calculate time of flight (TOF). I'd bet they would be happy just to sell SoCs, but the reality is that unless you can get companies to buy in, that just won't happen. I am pleased with the open approach they are offering. Users have full access to the SoC's register set, which is divided into the internal registers (those dealing with the system microprocessor) and external registers (those working with the correlation processor.) Before getting into these, let's take a quick tour of what's going on.

The transmitting laser is modulated by formatted data at a 50-MHz rate. The modulating data is a 500-ns Barker code burst (ideal data chosen for its autocorrelation properties). This burst is first sent to the receiver, where it is sampled as a reference. Then it is sent out the transmitting optics to be hopefully received when the signal is reflected

off an object and bounces back through the receiving optics. The receiver samples data (via a comparator) at a very fast single bit rate of 500 MHz. It's not the actual samples that are stored, but the accumulation of edges seen (changes in data) during a particular 2-ns window. For those of you keeping track, 2 ns is the time for light to travel 2' or a measurement distance of 1', so each sample has the resolution of about 1' in distance. Each sampling period continues from zero time (the start of the transmitted burst) until the allotted sample memory is full. Multiple transmission bursts are required until the total accumulated edges for any sample window reaches some arbitrary value (or the maximum number of bursts have been transmitted. Somewhere in this memory there is a wave shape that resembles the reference.

Now we've got to do two things. Find the point in the sampled data that best matches the reference data. The correlation algorithm relies on the ability of the special Barker sequence data such that an exact match can be found. The difference between the amount of time (2-ns samples) from zero to the correlated sample point and the amount of time between zero and the correlation reference point is the distance to the nearest foot. The best match may actually occur between sample points,

Internal Registers		
Number	Name	Function
0x00	Command Control	Acquisition control
0x01	Mode/Status	Process status
0x02	Maximum Acquisition Count	Maximum allowable bursts
0x03 (WO)	Correlation Record Length	Number of blocks used (block = 64 2ns samples)
0x04	Acquisition Mode	Function control
0x05	Measured Threshold Offset (Acquisition)	Signed Sampler Ratio
0x06/7 (RO)	Measured Delay of Reference (Correlation)	Correlation Time Zero Offset
0x08	Reference Correlation Measured Peak	Correlation Maximum Value
0x09	Velocity Measurement (0.1m/s or 1m/s)	Calculated Speed
0x0A/B (RO)	Measured Delay of Signal (Correlation)	Correlation Time Zero Offset
0x0C (RO)	Signal Correlation Measured Peak	Correlation Maximum Value
0x0D (RO)	Correlation Noise Floor	Correlation Maximum Value
0x0E (RO)	Receive Signal Strength	Correlation Maximum Value/ Bursts
0x0F/10 (RO)	Calculated Distance (cm)	Calculated Distance
0x11	DC Threshold Command Value	DC Offset Setting
0x12 (WO)	Added Delay	reduces demand on transmitter
0x13	Distance Calibration	Signed Distance Offset
0x14/15 (RO)	Previous Measured Distance	Last Calculated Distance

**TABLE 1**

External registers refer to those within the microprocessor and are mapped to begin at 0x00

where adjacent correlation values reverse polarity. By interpolating between these two adjacent sample points, a fractional portion of a foot can be obtained.

## USING THE I<sup>2</sup>C THE DATA

The I<sup>2</sup>C interface is fixed to address 0x62. Like many devices, there are a number of registers that are available to the user. I<sup>2</sup>C transfers are based on a register pointer. The first data byte following a write will always be loaded into this pointer. Therefore, most reads will begin with a write to set the pointer to the register of interest. Writes containing more

than one data byte will fill registers with data.

The LIDAR-Lite register set contain two groups of registers, internal registers 0x00–0x15 from the microcontroller and external registers 0x40–0x68 come from the correlation processor. You may notice some similarity between some internal registers in **Table 1** and the external registers in **Table 2**. Many registers are Read Only (RO) or Write Only (WO). Note all registers are less than 128 (0x00–0x7F). One highly important fact about this particular implementation of the I<sup>2</sup>C interface is that the register pointer only autoincrements when you set bit 7 to “1”

External Registers		
Number	Name	Function
0x40	Command Control	Begin Process
0x41	Hardware Version	
0x42	Preamplifier DC Control	DC Offset Setting
0x43	Transmit Power Control	Reference and Signal Power
0x44	Processing Range Gate (low)	Correlation starting point LSB
0x45	Processing Range Gate (high)	Correlation starting point MSB
0x46/7	Range Measurement PWM Output	Trig/PWM Pin status
0x48	Acquisition Status	Process status
0x49 (RO)	Measured Threshold Offset (Acquisition)	Signed Sampler Ratio
0x4A (WO)	Output Port	Control Port
0x4B	Range Processing Criteria (2 echoes)	Selection bits
0x4C (RO)	2nd Largest Detected Peak (Correlation)	Maximum Value Found
0x4F	Software Version	
0x51 (WO)	Correlation Record Length	Number of blocks used (block = 64 2ns samples)
0x52 (RO)	Correlation Data Access Port	Memory Data (byte)
0x53 (RO)	Correlation Data Access Port	Memory Data (sign bit)
0x57/58 (RO)	Measured Delay (Correlation)	Correlation Time Zero Offset
0x59 (RO)	Correlation Peak Value (Correlation)	Correlation Maximum Value
0x5A (RO)	Correlation Noise Floor	Correlation Maximum Value
0x5B (RO)	Received Signal Strength	Correlation Maximum Value/ Bursts
0x5C (WO)	Reset Correlator / Increment Burst Pattern	Reset/Control
0x5D (WO)	Acquisition Settings	Memory Selection
0x5F (RO)	Measured Transmit Power	Hardware Measurement
0x60 (RO)	Measured Fine Delay	Interpolation Value
0x61/62 (RO)	Course Delay	Correlation Point
0x63	Positive Correlation Sample (BZC)	Data Value of Correlation Point
0x64	Negative Correlation Sample (AZC)	Data Value of Correlation Point+1
0x65 (WO)	Power Control Settings	Power bits
0x68	Velocity Measurement Window	Interpulse Spacing Period (delta distance/time)

**TABLE 2**

External registers refer to those within the correlation processor and are mapped to begin at 0x40.



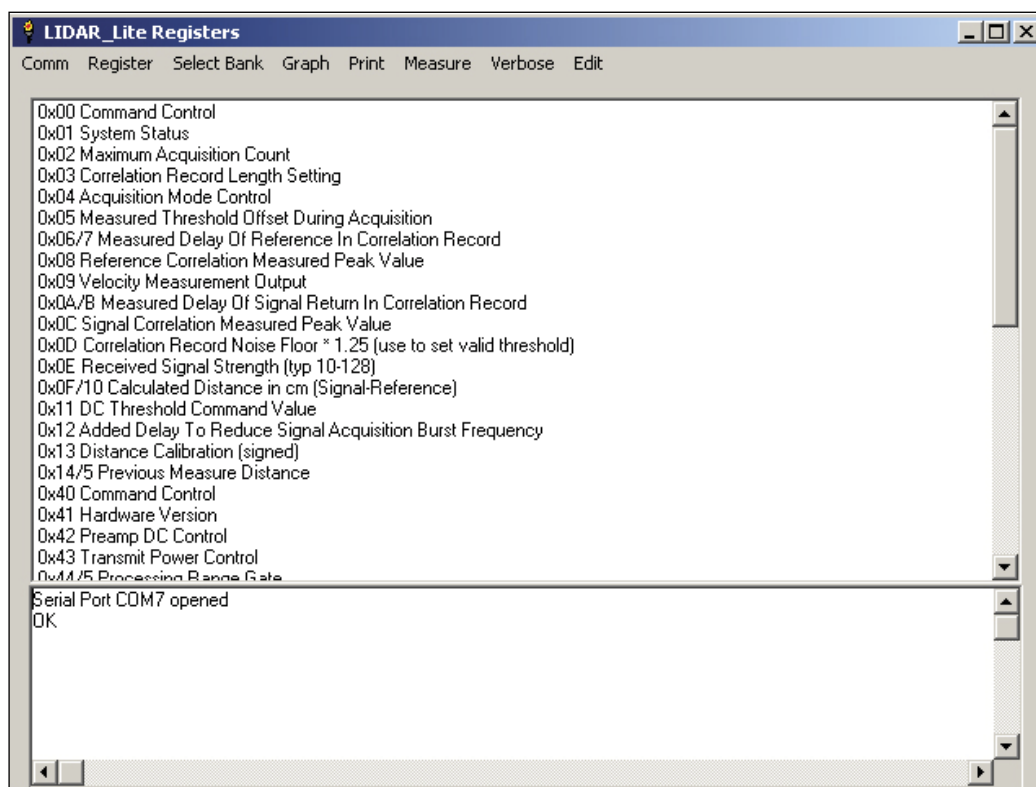


PHOTO 1

My first application makes it easy to read and write to LIDAR-Lite's registers. A drop-down box displays all the registers for easy picking. Besides interacting with registers, other selections automate activities like gathering data from any of the memory areas and graphing and printing the data.

## Sign up for the **FREE Circuit Cellar Newsletter!**

You'll receive electrical engineering tips, interesting electronics projects, embedded systems industry news, and exclusive product deals via e-mail to your inbox on a regular basis. If you're looking for essential electrical engineering-related information, we've got you covered: microcontroller-based projects, embedded development, programmable logic, wireless communications, robotics, analog techniques, embedded programming, and more!

*Subscribe now to stay up-to-date with our latest content, news, and offers!*

**circuitcellar.com**



when addressing any register. This allows you the option to repeatedly read a register without having to set the register pointer prior to each additional read. This is especially important when reading data records through a register pair, if 7-bit = "0", you'll read the same data repeatedly.

To begin the acquisition process with the default configuration, we need only write 0x04 to register 0x00, the Command Register. The resulting calculated distance will be placed into register pair 0x0F:0x10. You can read a byte from 0x0F and a byte from 0x10 or read two bytes from 0x8F. (Remember the autoincrement function uses bit7 = "1".) Note that I<sup>2</sup>C reads and writes use an ACK/NACK bit to allow the selected device to indicate that it is

too busy for a response. Make sure you check for this and retry the read/write if the device was busy. Since the resulting 16-bit value is presented in centimeters, you may wish to apply a conversion factor to the value if you wish the distance result to be in some other units like inches for instance. Otherwise, that's all that's necessary to take a measurement and retrieve the data.

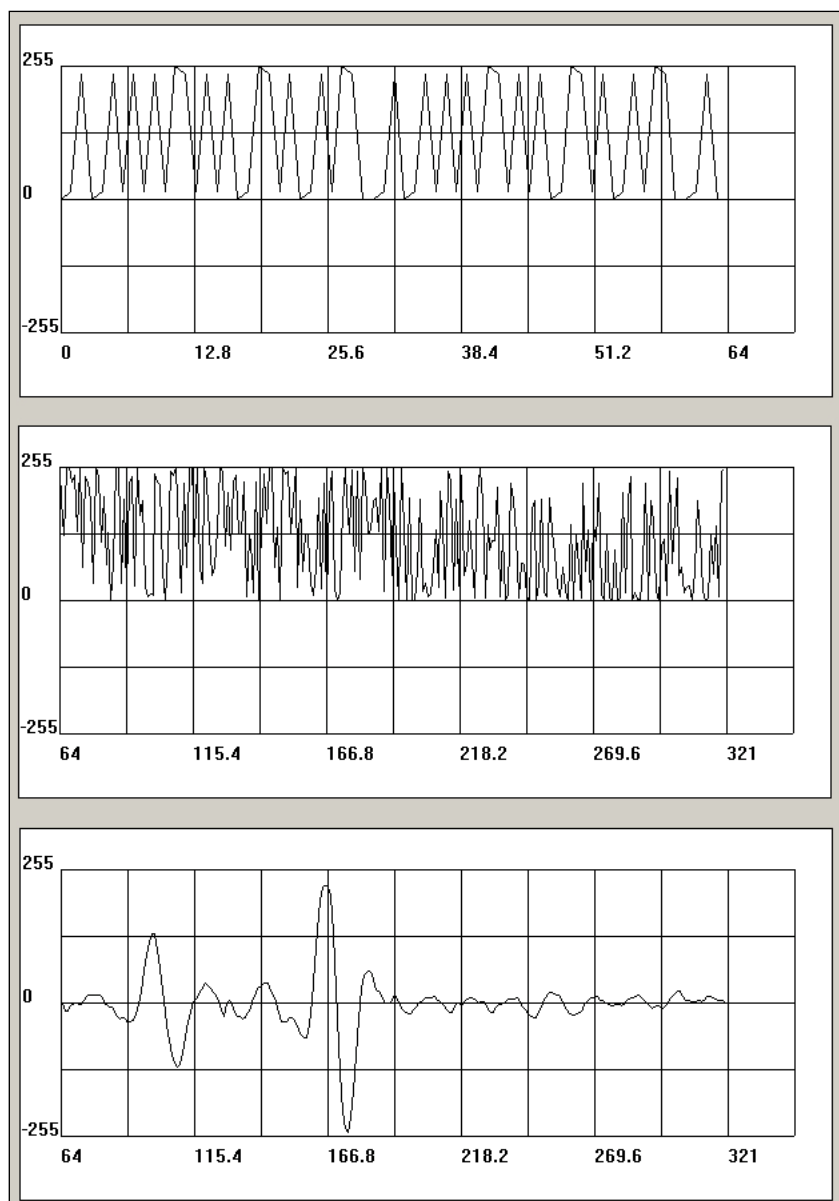
Since PulsedLight has opened up many of the registers to us, we can get a good idea of what is going on by dumping some of the memory areas. We can use register 0x5D to request one of three areas, template, signal, and correlation. The template area holds the samples for the reference burst (64 samples). The signal area holds the samples received for the burst echo. The correlation area holds the correlation result on the template and sample data.

To help me see this, I modified the Arduino program I used last month to act as a go between the LIDAR-Lite and Arduino's USB port. The Arduino recognizes a few ASCII commands (Wxx yy=zz and Rxx yy) to write to and read from the LIDAR-Lite I<sup>2</sup>C registers, where xx is the base I<sup>2</sup>C device address, yy is the register of interest, and zz is the data, all in hexadecimal. This would allow a serial terminal to talk to the LIDAR with simple commands typed from the keyboard using a serial terminal program. Or, one could write an application to make it even easier to mess around with. The Liberty BASIC application I wrote (see **Photo 1**) shows Registers selected from the menu bar. Here you can choose any register to read from or write to (WO registers read as zero). Other menu bar items Select Bank, Graph, and Print are used to obtain, graph, and print data from the three memory choices template, signal, and correlation. I pasted graphs of the three memory areas into **Photo 2** for comparison. While the template's 64 bytes are stretched to fit screen, the sample and correction lengths are based on register 0x51, the start and stop addresses (in 64-byte chunks.)

The measurement and verbose menu options let you toggle On/Off continuous measurements (and display the result in your choice of units as in **Photo 3**) and see all of the conversations between the application and the LIDAR-Lite/Arduino hardware.

## MAPPING

The high-end LIDAR units use a rotating laser to take measurements in a 360° pattern. While this might be on the drawing board at PulsedLight, I can take it to an intermediate level by adding a servo to pan the LIDAR-Lite module. Once again, I modified the Arduino interface to include an RC servo board. Yes, I realize that the Arduino has a servo command, but I wanted to avoid potential critical timing issues, so I used



**PHOTO 2**

Here we see sampled data from the reference (top) and sample (center). The bottom graph is the correlation reference and sample data

the Adafruit's 16-channel 12-bit PWM/Servo Driver I2C interface. An additional command 'Sx y' allows any servo (0-15) to be set to any absolute position ( $-90^\circ$  to  $90^\circ$ ).

I wanted to see what my office would look like using LIDAR-Lite as if it were attached to a robot that scanned the area from left to right. I wrote another application to instruct the servo to start at  $-90^\circ$  (left) and to take a measurement at every degree to  $90^\circ$  (right). To assure accuracy, I assumed the servo takes 1 s to traverse  $180^\circ$  (it's actually  $\sim 0.6$  s). So, I pause 60 s/ $180^\circ$ , or 333 ms after each  $1^\circ$  step, before taking a measurement. In reality, one horizontal scan requires 9 s. If I move vertically  $1^\circ$  each horizontal scan, the total time will now take  $9 \text{ s} \times 50 \text{ scans} = 450 \text{ s}$ , or  $\sim 8$  minutes, and require larger arrays for azimuth, elevation, and measurement. I added a delay between each move and measurement to allow the servo to reach position and cease movement before a measurement was taken. A couple of arrays are used to store position and distance measured.

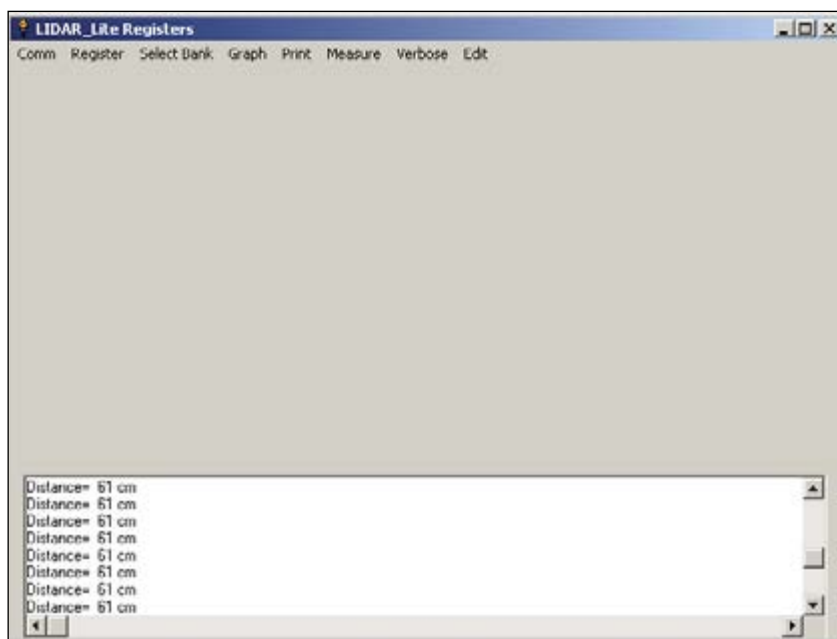
Once the arrays were filled with the 181 sample points taken in a horizontal scan line, the collected data could be displayed on a graph as if viewed from above. The resulting graph is shown in **Photo 4**. The LIDAR module is located atop a printer just to left of my desk and pointing away from the desk. A worktable and file cabinets are located in front of the module and you can see free space out to the module's right. This is actually a closet I use to store active projects. A robot would have no problem finding its way into the closet.

Once I'd seen the graph, it was all over. I wanted more. Thus, a second servo was added to allow the LIDAR-Lite to see up and down with a change in elevation. Most of the support was already there. I just needed to add a routine to change the elevation servo between each horizontal azimuth scan. I chose to use elevations between  $-10^\circ$  and  $40^\circ$ . This would extend a 9-s scan to  $9 \times 50 = 450 \text{ s}$ , or  $\sim 8$  minutes, and require larger arrays for azimuth, elevation, and measurement.

I added file save, load, and print graph routines so I would have a permanent record of the data. When graphed, it was difficult to make sense out of the additional data when viewed from above as in **Photo 5**. Oh, how nice it would be to see this in 3-D! I didn't have the time to devote to writing a 3-D viewer and was in a deep funk. I could, however, use my application to calculate Cartesian coordinates for each Spherical measurement I had taken. Most CAD programs allow import and export of DFX files and this used Cartesian coordinates.

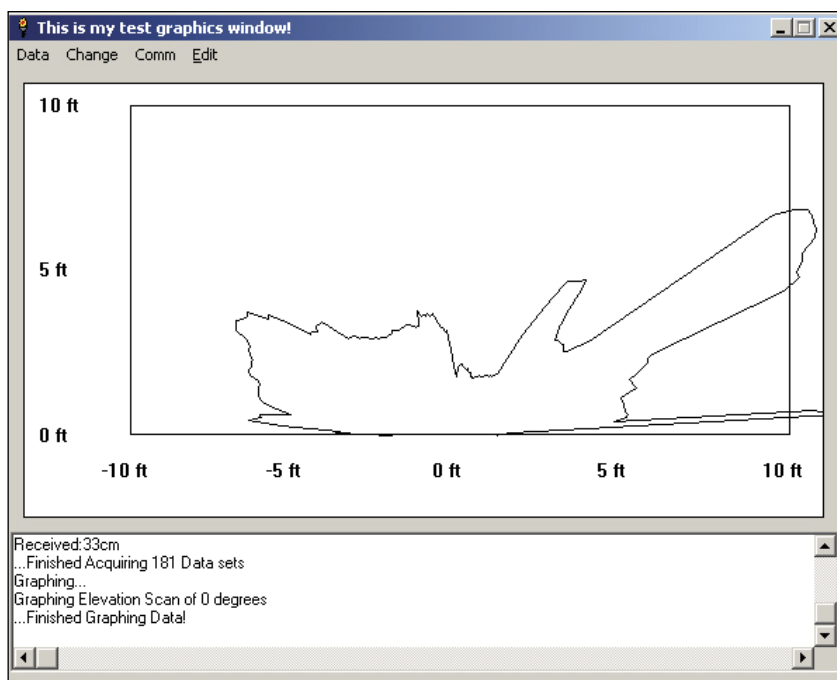
## SPHERICAL VS. CARTESIAN

Spherical coordinate system use two angles (azimuth and elevation) and distance to select a



**PHOTO 3**

Of course we can read the measurement results via I2C (as opposed to measuring the pulse width output from the Trig/Pulse pin). While data is presented in centimeter units, a simple conversion will put this into any format you may want.



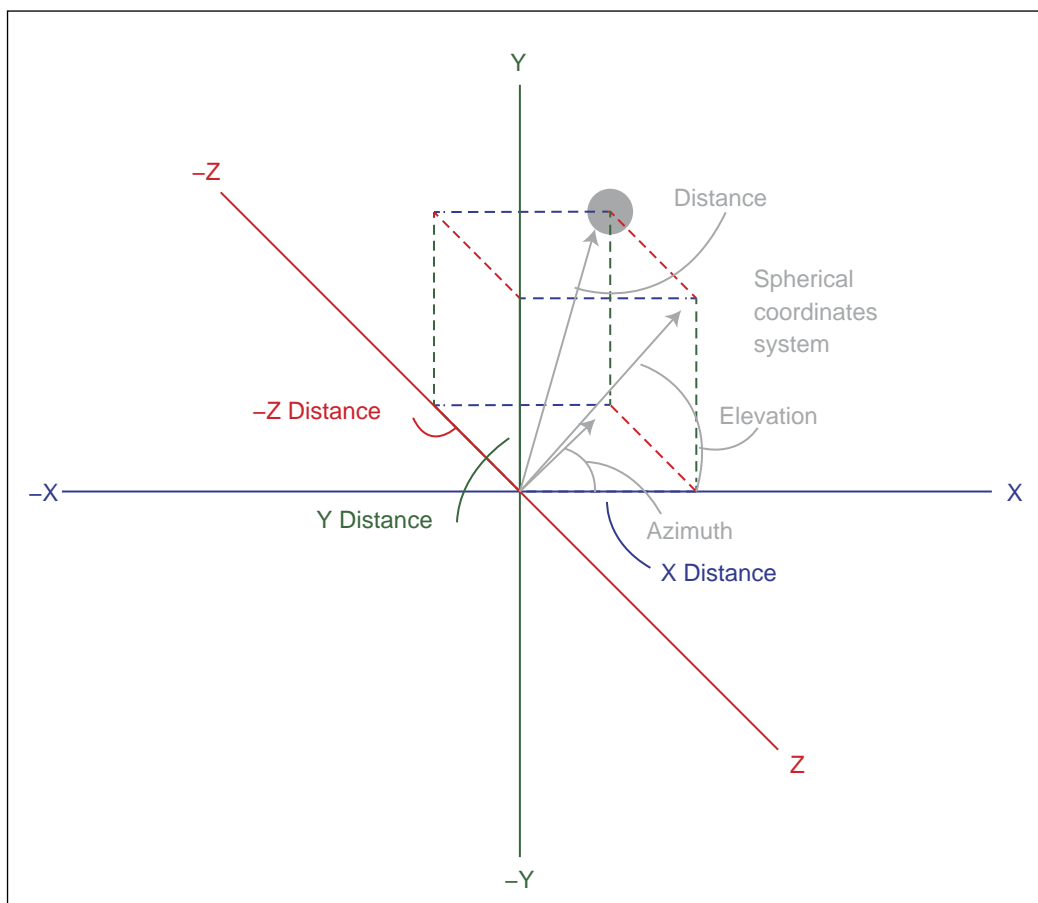
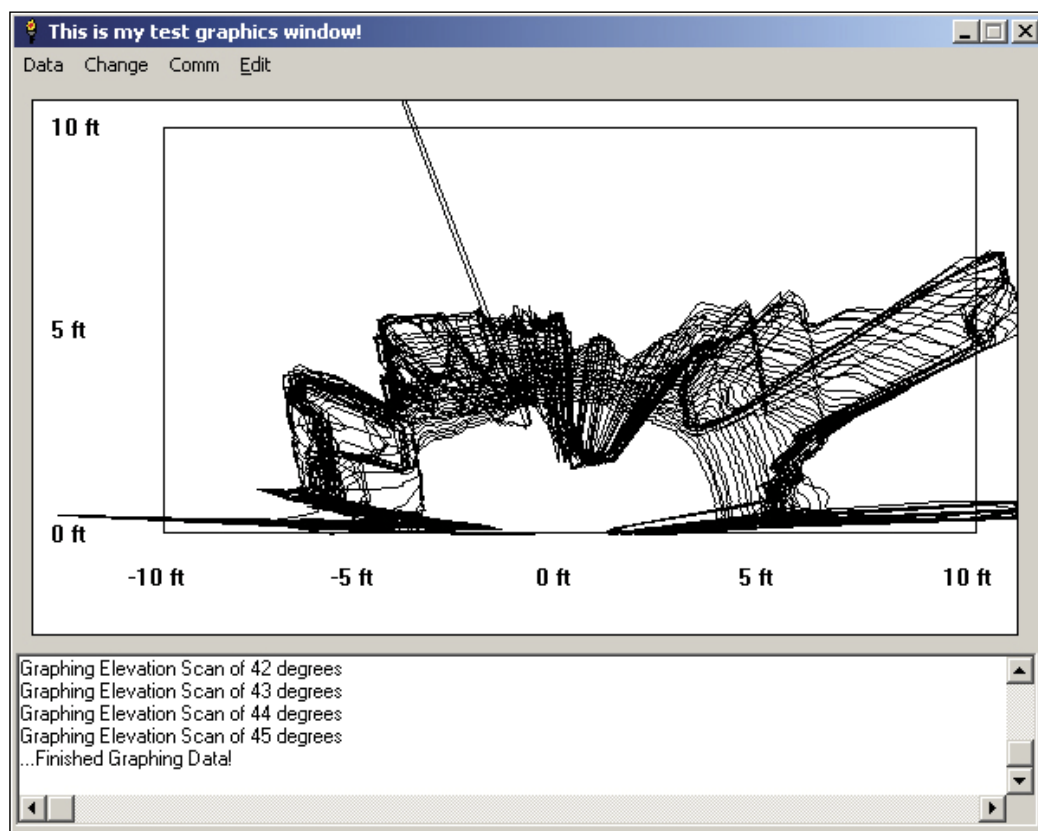
**PHOTO 4**

Plotting a bird's eye view of the data lets you see exactly what's out there in a horizontal scan of my cluttered office. There isn't much room right out in front of the sensor, but off to the right, there is a clear shot into a closet.



**PHOTO 5**

I was excited to take multiple scans at increasing elevation. However, plotting these scans one atop another gave me a less interesting plot. Looking at 3-D data in 2-D doesn't cut it!

**FIGURE 1**

Here is an attempt to show a particular point of references by both the Spherical and Cartesian coordinate systems. The Spherical system uses two angles and a distance. The Cartesian system uses three distances along three perpendicular axes.



# 27

## ADVANCE PROGRAM

August 23-25, 2014

A Symposium on High-Performance Chips  
Flint Center for the Performing Arts-Cupertino, CA

<http://www.hotchips.org>

HOTCHIPS brings together designers and architects of high-performance chips, software, and systems. The tutorial and presentation sessions focus on up-to-the-minute developments in leading-edge industrial designs and research projects.

Register Now: <https://www.123signup.com/register?id=ytdtb>

Sunday August 23	<b>Tutorial 1: Deep Learning</b> <ul style="list-style-type: none"> <li>Architectures, Algorithms and Applications</li> <li>Common Software Tools, Research Questions, Outlook</li> </ul>	University of Montreal	
	<b>Tutorial 2: Makers from Hobbyists to Professionals</b> <ul style="list-style-type: none"> <li>Maker Trends: The Path of Least Resistance</li> <li>IoT Device Development Challenges and Solutions</li> <li>Implementing Software Defined Radio on the Parallella</li> <li>Current Trends for Hardware &amp; Software Developers</li> </ul>	UC Davis SparkFun Redpine Signals Adapteva ARM	
Monday August 24	<b>Internet of Things</b> <ul style="list-style-type: none"> <li>PULP: A Parallel Ultra-Low-Power Platform for Next Generation IoT Applications</li> <li>Design of an Ultra-low Power SoC Testchip for Wearables &amp; IOT</li> <li>Ultra-Low Power Wireless SoCs Enabling a Batteryless IoT</li> </ul>	DEI, ETH, STMicro Intel PsiKick	
	<b>Keynote 1 Convolutional Neural Networks</b> <b>Yann LeCun</b>	Facebook	
	<b>Multimedia and Signal Processing</b> <ul style="list-style-type: none"> <li>Architecture of the V6x Hexagon DSP for Mobile Imaging and Always-On Applications</li> <li>A Scalable Heterogeneous Multicore Architecture for the Advanced Driver Assistance System</li> <li>Energy-Efficient Graphics and Multi-media in 28nm Carrizo APU</li> <li>Revisiting DSP Acceleration with the Kalray MPPA Manycore Processor</li> </ul>	Qualcomm TI AMD Kalray	
	<b>High Performance and Cloud Computing</b> <ul style="list-style-type: none"> <li>Scale-Out Computing and the X-Gene Processor Family</li> <li>A 64-Core ARM v8 Processor for HPC</li> <li>Oracle's Sonoma Processor: Advanced Low-Cost SPARC Processor for Enterprise Workloads</li> <li>I/O Virtualization and System Acceleration in Power8</li> </ul>	Applied Micro Phytium Oracle IBM	
	<b>FPGAs</b> <ul style="list-style-type: none"> <li>Xilinx 16nm UltraScale+ MPSoC and FPGA Families</li> <li>Stratix 10: Altera's 14nm FPGA Targeting 1GHz Performance</li> <li>Toward Accelerating Deep Learning at Scale Using Specialized Logic</li> </ul>	Xilinx Altera Microsoft Research	
Tuesday August 25	<b>GPUs</b> <ul style="list-style-type: none"> <li>MIAOW – An Open Source GPGPU</li> <li>AMD's next Generation GPU and Memory Architecture</li> <li>The ARM Mali-T880 Mobile GPU</li> </ul>	University of Wisconsin AMD ARM	
	<b>Keynote 2 The Road to 5G</b> <b>Matt Grob, CTO</b>	Qualcomm	
	<b>Applications</b> <ul style="list-style-type: none"> <li>Professional H.265/HEVC Encoder LSI Toward High-Quality 4K/8K Broadcast Infrastructure</li> <li>Ultra-Low-Light CMOS Biosensor Helps Tackle Infectious Diseases</li> <li>Five-Speed PHY Enables 2.5Gbps and 5Gbps Ethernet Rates Over Legacy Copper Cables</li> </ul>	NTT Anitoa Aquantia	
	<b>Processors</b> <ul style="list-style-type: none"> <li>Knights Landing: 2nd Generation Intel "Xeon Phi" Processor,</li> <li>Intel "Xeon" Processor D: The first Xeon Processor Optimized for Dense Solutions</li> <li>Raven: A 28nm RISC-V Vector Processor with Integrated Switched-Capacitor DC-DC Converters and Adaptive Clocking</li> <li>Intel Atom-X5/X7-8000 Series Processors, Codenamed Cherry Trail</li> </ul>	Intel Intel UC Berkeley Intel	
	<b>Keynote 3 The Future of the Cloud</b> <b>Mark Zuckerberg</b>	Facebook	



A Symposium of the Technical Committee on Microprocessors and Microcomputers  
of the IEEE Computer Society and the Solid-State Circuits Society

```
[AngleDistanceElevationToXYZ]
  ' x axis front to back
  x=cos(Elevation/57.29577951) *
sin(Azimuth/57.29577951) * Distance
  ' z axis ear to ear
  z=sin(Elevation/57.29577951) * Distance
  ' y axis head to toe
  y=cos(Elevation/57.29577951) *
cos(Azimuth/57.29577951) * Distance
  return
```

**LISTING 1**

This routine translates data from the Spherical to Cartesian Coordinate System.

```
...
-90, 0, 5.01, -5.0820000, 0.0000000, 0.0000000
-89, 0, 4.93, -4.9492461, 0.0863894, 0.0000000
...
```

**LISTING 2**

Here we see a few records of the data file saved, which hold reference points as measured and translated from the Spherical to Cartesian Coordinate System.

point in space. Azimuth is most often associated with a compass reading, which is referenced to magnetic north. A negative azimuth or angle would be to the left of reference and to the right the azimuth would be positive. If you went either left  $-180^\circ$  or right  $180^\circ$ , you would end up at the same place directly to the rear of the reference. If you continued in both directions back to the reference you would have traveled either  $-360^\circ$  or  $+360^\circ$  back to where you started. You could measure the distance to any object in your vicinity by turning toward that object and noting the compass azimuth, and then noting how many paces (steps) from home base (where you started) to the object. Anyone knowing where home base was, could in turn

use a compass to turn toward your azimuth and step off your pace count to locate your object.

This idea of angle and distance can be extended into 3-D space where a second angle is added, elevation. With elevation, the same rules apply as with azimuth. We begin with some reference, with azimuth the reference was magnetic north, with elevation we can use eye level, the height of your eye (or the sensor) above and level with the ground. Any elevation above level is considered a positive angle. Below level would be a negative elevation angle. From a point of reference, all other points can be specified using an azimuth angle, an elevation angle, and a distance. The angles define a direction from the reference point at the center of a sphere to some point on the sphere. Distance defines the radius of that sphere. Of course, both angles and distances do not need to be whole numbers.

The Cartesian coordinate system is based on three perpendicular axis normally labeled x, y, and z. The point where the three axes meet is the reference point. The reference point is zero for each axis with distances along each axis extending both in a positive direction and a negative direction. Like the Spherical coordinate system, the Cartesian coordinate system requires three values to define a point (see **Figure 1**). However, the Cartesian coordinate system uses no angles, just distance measurements along each of the three axes. Imagine that each axis is a ladder and you must climb each ladder and look for the point using top and bottom blinders perpendicular to ladder. The value for that axis is how far up the ladder you had to go (positive or negative) until you were able to get the point in your narrow field of view. These three distances define the point in space. Any point in space has coordinates in each coordinate system. And thanks to trigonometry, we can convert from one system to the other.

My application in Liberty Basic has built-in sin/cos functions, so a simple function can be called to convert each measurement (see **Listing 1**). Three more arrays hold these x, y, and z values and the file function now saves all six values: azimuth, elevation, distance(ft), x, y, and z (see **Listing 2**).

While I could have saved just the azimuth, elevation, and distance, and calculated the x, y, and z externally, having all six values in the file makes it more meaningful when you look through the data. Now I needed just one additional step: prepare the data to be accepted via the import function of some CAD application. I chose Sketchup Make, a free application by Trimble Navigation. After searching for a little background on the Internet for some info on the format of the DXF<sup>1</sup> file, I was able to prepare a small frame



circuitcellar.com/ccmaterials

**SOURCES**

16-Channel 12-bit PWM/Servo Driver I2C Interface  
Adafruit Industries | [www.adafruit.com](http://www.adafruit.com)

Arduino Mega  
Arduino | [www.arduino.cc](http://www.arduino.cc)

LIDAR-Lite Laser Module  
PulsedLight | [www.pulsedlight3d.com](http://www.pulsedlight3d.com)

Liberty Basic  
Shoptalk Systems | [www.libertybasic.com](http://www.libertybasic.com)

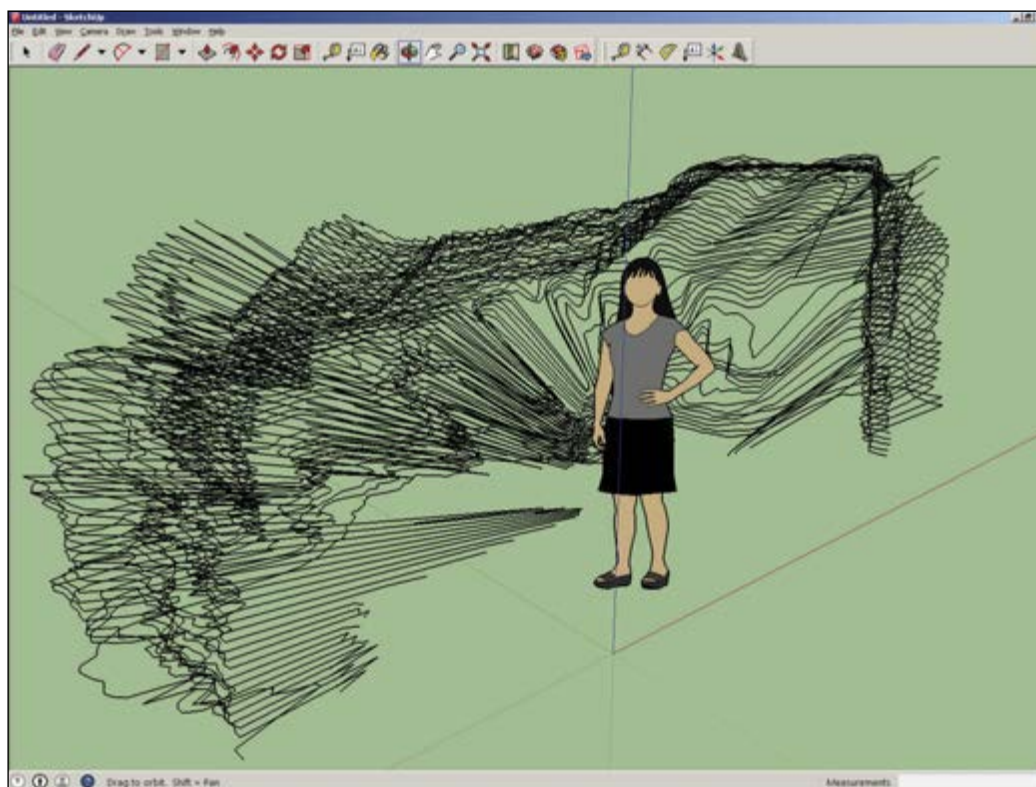
**RESOURCES**

Autodesk, Inc., "AutoCAD DXF Reference," 2011, [http://images.autodesk.com/adsk/files/autocad\\_2012\\_pdf\\_dxf-reference\\_enu.pdf](http://images.autodesk.com/adsk/files/autocad_2012_pdf_dxf-reference_enu.pdf).

——, "Writing DXF Interface Programs," [www.autodesk.com/techpubs/autocad/acadr14/dxf/writing\\_dxf\\_interface\\_programs\\_al\\_u05\\_b.htm](http://www.autodesk.com/techpubs/autocad/acadr14/dxf/writing_dxf_interface_programs_al_u05_b.htm).

P. Bourke, "Minimum Requirements for Creating a DXF File of a 3D Model," 1993, <http://paulbourke.net/dataformats/dxf/min3d.html>.



**PHOTO 6**

This photo of SketchUp is displaying a DXF import of my converted coordinates from the LIDAR-Lite. The camera perspective allows the user to move in, out, and around the plotted data. It was very satisfying to fly into the plot and feel like I knew where I was.

work of some basic information where I could insert some line segments. The DXF format for a line drawn between two points is as follows:

```
0
LINE
8
0
10
-5.082
20
0
30
0
11
-4.9492461
21
0.0863894
31
0
```


Each section starts with a group code 0 followed by the name of the entity, in this example, LINE. Next is the group code 8 followed by the name of the layer, 0. Next are the x, y, z coordinates for each end point. Each group of coordinates for each vertex is preceded by an identifier. The first vertices are identified with 10, 20, 30 for (x1, y1, z1). The second identifiers are 11, 21, and 31 for (x2, y2, z2). Just repeat this pattern for each line segment and place that in the minimum template.

Again, it was back to Liberty Basic to build a translation application to read in the outputted

coordinate file and produce a DXF file. I produce a line segments for pairs of coordinates (first and second, second and third, third and fourth, etc.) for each scan (elevation). When the file was imported into SketchUp, it did a file check and reported 10,080 line segments. The plot displayed in SketchUp is very dynamic, in that you can use the tools provided to move, pan, and zoom the viewer's perspective from anywhere. **Photo 6** hardly does it justice.

## EXPLORING MORE

You never know what direction a little experimentation will take you. I didn't think I would be talking about converting coordinates and creating 3-D images when I first began. There are plenty other avenues that deserve more exploration. The open interface of LIDAR-Lite begs for more touching. We might choose to look at changing registers to lengthen the sampling window (for more distant target recognition). What about locating multiple targets from multiple echoes received within a sampling? Are you interested in measuring the speed of objects? In case you didn't notice earlier, LIDAR-Lite can return speed (register 0x68) based on the time between successive measurements.

It seems I have just scratched the surface of this device. I hope you will take what I've presented and run with it down your own path. I believe this is one of those products that will have a large impact on what comes next. What say you? 



## ABOUT THE AUTHOR

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

## CC SHOP



1

## 2 CC 2014 CD

2014 was an exciting year for electronics engineers! The continued success of open-source solutions, Internet of Things (IoT) revolutions, and green-energy consciousness has changed the face of embedded design indefinitely. In *Circuit Cellar's* 2014 archive CD, you can find all of these hot topics and gain insight into how experts, as well as your peers, are putting the newest technologies to the test. You'll have access to all articles, schematics, and source code published from January to December 2014.

Item #: CD-018-CC2014

Previous Years Also Available

## 3 ADuC841 MICROCONTROLLER DESIGN MANUAL

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

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

Author: Shlomo Engelberg

Item #: CC-BK-9780963013347



4

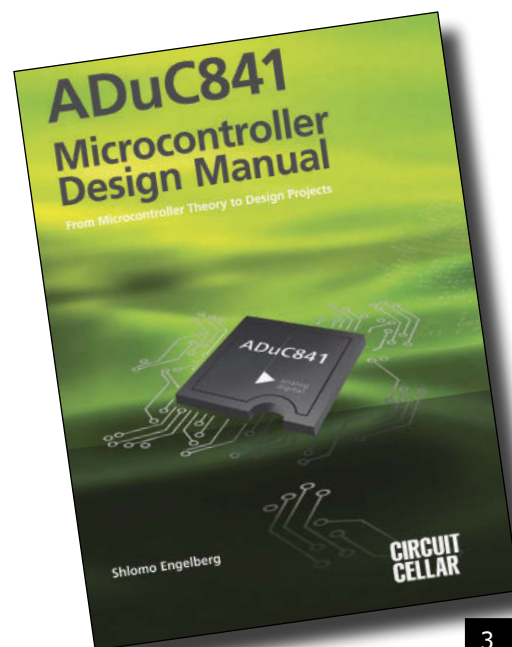
## 1 CC VAULT

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

Item #: CCVAULT



2



3

## 4 CC 2014 DIGITAL ARCHIVE SUBSCRIPTION

Just when you thought it couldn't get any easier than a thumb drive...you can now access a full year of *Circuit Cellar* from any device connected to the Internet! (2014: 12 issues)

You get all the benefits of a printed copy—bookmark pages, make annotations, and write in the margins—combined with the digital advantages of easy storage, zoom, links, and search features.

Item #: CC-DA-2014

Further information and ordering: [www.cc-webshop.com](http://www.cc-webshop.com)

CONTACT US: Circuit Cellar, Inc. | Phone: 860.289.0800 | E-mail: [custservice@circuitcellar.com](mailto:custservice@circuitcellar.com)



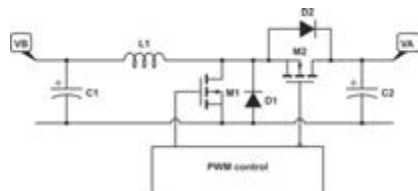
The answers to the EQ problems that appeared in *Circuit Cellar 300* (July 2015).

**ANSWER 1**—In normal operation, M2 is switched on first, and current flows through it and L1, charging the inductor with magnetic energy. When M2 switches off and M1 switches on, the current continues to flow through L1, discharging its stored energy.

Now, if M1 weren't there, the circuit would still work, because the discharge current would still flow through D1. However, once L1's current drops to zero, the diode would block any further flow — this is known as “discontinuous conduction mode”. Whereas, with M1 present, the current flow can actually reverse. In other words, with active (synchronous) rectification, the converter can both source and sink current at its output. This is known as “continuous conduction mode.” This means that the relationship between the input voltage  $V_A$  and the output voltage  $V_B$  is only a function of the duty cycle of the switching:  $V_B = V_A(T_{M2}/T_{period})$ .

**ANSWER 2**—Yes, as mentioned above, it can indeed sink current. When the current in L1 goes negative, the current flows through M1 to ground as long as M1 is on. But when M1 switches off and M2 switches on, this forces current back toward  $V_A$  and C2, until the voltage across L1 causes the current to ramp back up to zero and then positive again.

**ANSWER 3**—Here is the corresponding diagram for a “boost” converter:



# TEST YOUR EQ

Contributed by David Tweed

In normal operation, M1 switches on first, charging L1 with magnetic energy. Then, M1 switches off and M2 switches on, allowing the stored energy to discharge into C2. The remarkable thing about this diagram is that it is an exact mirror image of the buck converter!

**ANSWER 4**—Again, with the boost converter, we could eliminate M2 and allow D2 to do the output switching, but M2 allows current to flow either way during the discharge phase. And just like with the buck converter, this means that the input-output voltage relationship becomes a function of only the switching duty cycle:  $V_A = V_B(T_{period}/T_{M2})$ .

Note that this is a simple rearrangement of the terms in the equation for the buck converter—in other words, it's the same equation. This tells us that regardless of which way the power is flowing, the relationship between  $V_A$  and  $V_B$  is simply a function of the switching duty cycle.

So, to turn this into a concrete example, if the PWM control is set up so that M2 is on  $5/12 = 42\%$  of the time, you could apply 12 V at  $V_A$  and get 5 V out at  $V_B$ , or you could apply 5 V at  $V_B$  and get 12 V out at  $V_A$ !

One final note about regulation: This circuit provides a specific ratiometric relationship between the two voltages that is based on the duty cycle of the switching. If the input voltage is unregulated, but you want a regulated output voltage, then you need to provide a mechanism that varies the duty cycle of the switch in order to cancel out the input variations. Note that this control could be based on measuring the input voltage directly (feedforward control) or measuring the output voltage (feedback control).

If you're going to build a practical bidirectional power converter with regulation, you'll have to pay extra attention to how this control mechanism works in both modes of operation.

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

## Now offering student SUBSCRIPTIONS!

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

Sign up today and **SAVE 50%**

Sign up today and **SAVE 50%**

Sign up today and **SAVE 50%**

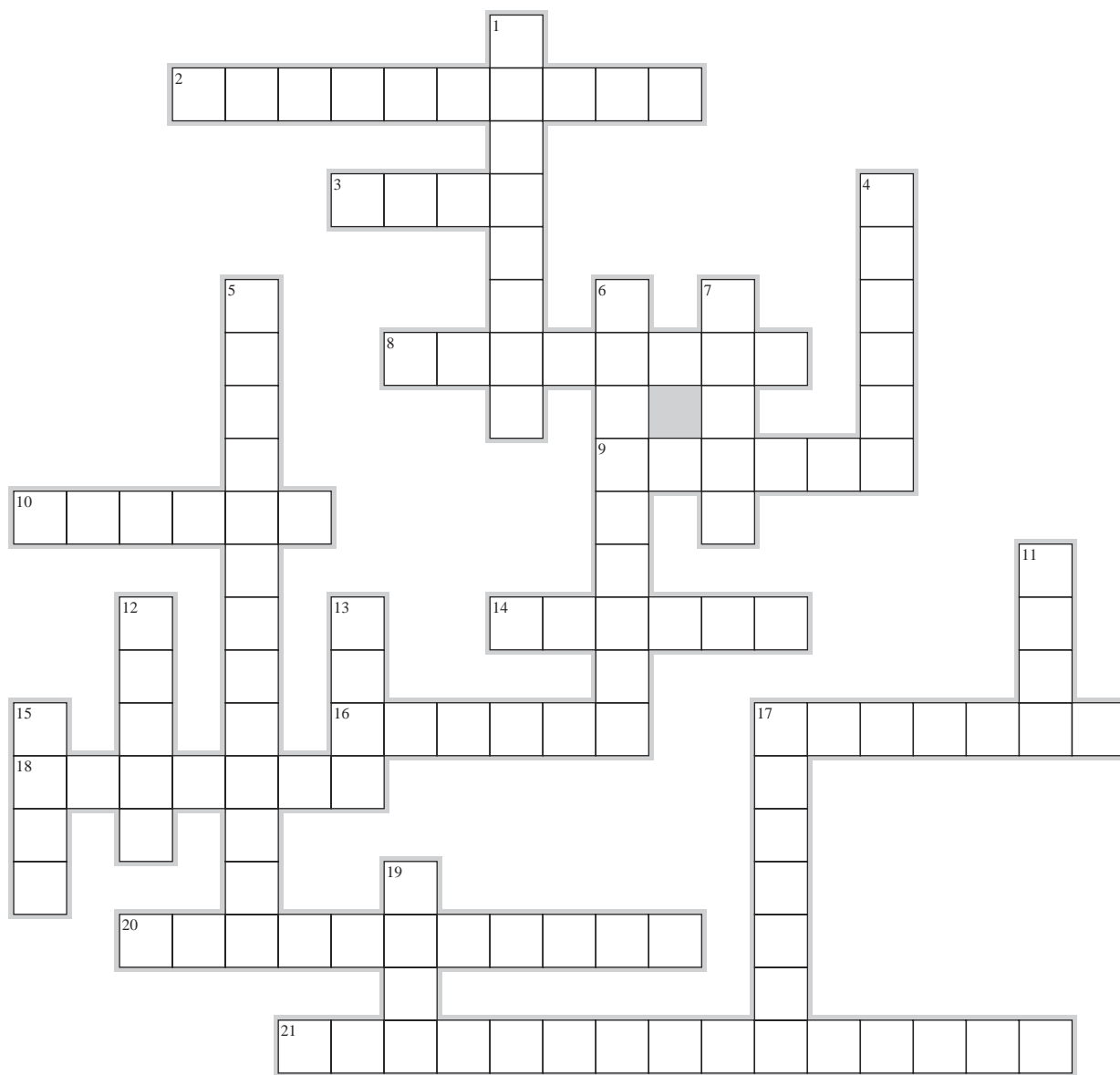
[www.circuitcellar.com/subscription](http://www.circuitcellar.com/subscription)



# CROSSWORD

**AUGUST 2015**

The answers will be available at [circuitcellar.com/category/crossword/](http://circuitcellar.com/category/crossword/)



## ACROSS

2. X
3. Point of minimum amplitude
8.  $6.0221415 \times 10^{23}$  atoms/mole
9. Covert a digital signal back to an analog signal
10. Founded MIT in 186
14. Non-rotating part of an electric motor
16. A half-byte
17. M/V
18. Vacuum tube with four elements
20. The steepness or severity of a filter's attenuation [two words]
21. Uniform propagation of energy in all directions

## DOWN

1. The loss of a single bit in a digital word [two words]
4. Poor solution
5. A figure-eight polar pattern
6. Signal exchange to start or finish a function
7. A line of conductive material on a PCB
11. FET in which the gate consists of a p-n junction
12. 1,000,000,000,000,000,000,000,000
13. Nonet
15. Analog television system
17. Set limits
19. Fe

# IDEA BOX

## the directory of PRODUCTS & SERVICES

For current rates, deadlines, and more information contact Peter Wostrel at 978.281.7708 or [circuitcellar@smmarketing.us](mailto:circuitcellar@smmarketing.us).

**\$20 for 5 PCBs**

**Standard PCB**

- 2 layer, 4"x4", FR4(RoHS), 0.063", 1 oz
- 2LPI, Green, Lead free HASL



PCB PCBA

Small to Mass QTY  
Instant Quote at:  
**[www.myropcb.com](http://www.myropcb.com)**  
Or Call 1-888-PCB-MYRO

**GHz BGA/QFN Sockets**  
**0.3mm to 1.27mm**

**Industry's Smallest Footprint**

- Up to 500,000 insertions
- Bandwidth up to 75 GHz
- 2.5mm per side larger than IC
- Ball Count over 3500
- Body Size 2 - 100mm
- Five different contactor options
- Optional heatsinking to 100W
- Six different LID Options
- <25 mOhm Contact Resistance throughout life



RoHS

**Ironwood ELECTRONICS** 1-800-404-0204  
**[www.ironwoodelectronics.com](http://www.ironwoodelectronics.com)**

**MaxBotix®**  
**High Performance Ultrasonic Rangefinders**



**Call Carlson today for application specific sensors**

Phone: 218-454-0766  
Email: [info@maxbotix.com](mailto:info@maxbotix.com)  
[www.maxbotix.com](http://www.maxbotix.com)

**Microprocessor Design**  
Using **Verilog HDL**



Shop for this book, and others, at [www.cc-webshop.com](http://www.cc-webshop.com)

**LEARN C FAST!**

**\$85!**  
book & board bundle

**EMBEDDED C PROGRAMMING**  
Techniques and Applications of C and PIC MCUS



Mark Siegesmund

**Book Bundle Includes:**  
Single-Chip Compiler & Dev Board!

**BUY NOW:**  
**[ccsinfo.com/CC815](http://ccsinfo.com/CC815)**

sales@ccsinfo.com  
262-522-6500 x 35

**CCS**

**ALL ELECTRONICS CORPORATION**

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

We have what you need for your next project.



**[www.allelectronics.com](http://www.allelectronics.com)**  
Free 96 page catalog 1-800-826-5432

**Assembly Language Essentials**

A Guide to Powerful Programming for Embedded Systems

**ASSEMBLY LANGUAGE ESSENTIALS**

A matter-of-fact guide that introduces the most fundamental programming language of a processor.


Complete with downloadable Assembler program, important algorithms, and more!

**Only \$47.50**

[www.cc-webshop.com](http://www.cc-webshop.com)

**PIC-SERVO**  
MOTION CONTROL

MOTION CONTROLLERS FOR BRUSH, BRUSHLESS AND STEPPER MOTORS.



- controller chips
- controller boards

**[www.picservo.com](http://www.picservo.com)**  
JEFFREY KERR, LLC

# Trends in Custom Peripheral Cores for Digital Sensor Interfaces

By Daniel Casner

**B**uilding ever-smarter technology, we perpetually require more sensors to collect increasing amounts of data for our decision-making machines. Power and bandwidth constraints require signals from individual sensors to be aggregated, fused and condensed locally by sensor hubs before being passed to a local application processor or transmitted to the cloud.

FPGAs are often used for sensor hubs because they handle multiple parallel data paths in real time extremely well and can be very low power. ADC parallel interfaces and simple serial shift register interfaces are straightforward to implement in FPGA logic. However, interfacing FPGAs with more complex serial devices—which are becoming more common as analog and digital circuitry are integrated—or serializing collected data is often less straightforward. Typically, serial interfaces are implemented in FPGA fabric as a state machine where a set of registers represents the state of the serial interface, and each clock cycle, logic executes depending on the inputs and state registers. For anything but the most trivial serial interface, the HDL code for these state machines quickly balloons into a forest of parallel if-elseif-else trees that are difficult to understand or maintain and take large amounts of FPGA fabric to implement. Iterating the behavior of these state machines requires recompiling the HDL and reprogramming the FPGA for each change which is frustratingly time consuming.

Custom soft cores offer an alternate solution. Soft cores, sometimes known as IP cores, are not new in FPGA development, and most FPGA design tools include a library of cores that can be imported for free or purchased. Often these soft cores take the form of microcontrollers such as the Cortex M1, Microblaze, lowRISC, etc., which execute a program from memory and enable applications to be implemented as a combination of HDL (Verilog, VHDL, etc.) and procedural microcode (assembly, C, C++, etc.).

While off-the-shelf soft core microprocessors are overkill and too resource intensive for implementing single serial interfaces, we can easily create our own custom soft cores when we need them that use fewer resources and are easier to program than a state machine. For the purpose of this article, a custom soft core is a microcontroller with an instruction set, registers, and peripheral interfaces created specifically to efficiently accomplish a given task. The soft core executes a program from memory on the FPGA, which makes program iteration rapid because the memory can be reprogrammed without resynthesizing or reconfiguring the whole FPGA fabric. We program the soft core procedurally in assembly, which mentally maps to serial interface protocols more easily than HDL. Sensor data is made available to the FPGA fabric through register interfaces, which we also define


according to the needs of our application.

Having implemented custom soft cores many times in FPGA applications, I am presently developing an open-source example/template soft core that is posted on GitHub ([https://github.com/DanielCasner/i2c\\_softcore](https://github.com/DanielCasner/i2c_softcore)). For this example, I am interfacing with a Linear Technology LTC2991 sensor that has internal configuration, status, and data registers, which must be set and read over I<sup>2</sup>C (which is notoriously difficult to implement in HDL). The soft core has 16-bit instructions defined specifically for this application and executes from block ram. The serial program is written in assembly and compiled by a Python script. I hope that this example will demonstrate how straightforward and beneficial creating custom soft cores can be.

*While off-the-shelf soft core microprocessors are overkill and too resource intensive for implementing single serial interfaces, we can easily create our own custom soft cores when we need them that use fewer resources and are easier to program than a state machine.*

While I have been discussing soft cores for FPGAs in this article, an interesting related trend in microprocessors is the inclusion of minion cores, sometimes also called programmable real-time

units (PRUs) or programmable peripherals. While not fully customizable as in FPGA fabric, these cores are very similar to the soft cores discussed, as they have limited instruction sets optimized for serial interfaces and are intended to have simple programs that execute independently of the application to interface with sensors and other peripherals. By freeing the main processor core of direct interface requirements, they can improve performance and often simplify development. In the future, I would expect more and more MCUs to include minion cores among their peripherals.

As the amount of data to be processed and efficiently requirements increase, we should expect to see heterogeneous processing in FPGAs and microcontrollers increasing and be ready to shift our mental programming models to take advantage of the many different paradigms available. 



Daniel Casner is a robotics engineer at Anki ([anki.com](http://anki.com)), co-founder of Robot Garden ([Robotgarden.org](http://Robotgarden.org)), hardware start-up adviser, and embedded systems consultant. He is passionate about building clever consumer devices, adding intelligence to objects, and smart buildings or any other cyber-physical system. His specialties include: design for manufacture and salable production; cyber-physical security; reverse engineering; electronics and firmware; signal processing; and prototype development.



# CC Vault

Unlock the power of embedded design.



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

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

A vault of need-to-know information in the fields of embedded hardware, embedded software, and computer applications

\*CC Vault is a 16-GB USB drive.

Order yours today! [cc-webshop.com](http://cc-webshop.com)



***We bring the full range of Electronic Contract Manufacturing services to your fingertip!***

***FABRICATION***



***ASSEMBLY***



***KEYPADS***



***ENCLOSURES***



***This is the only place where you would put all your eggs in one basket to get fastest time to market. From concept design to prototype to full turnkey production on all your electronic products.***



***imagineering inc.***

***www.PCBnet.com***

***847-806-0003   sales@PCBnet.com  
Certified Woman-Owned Small Business***

