Q&A: Electronics Engineering for the People
LOCATION: Australia
PAGE: 22

PROJECT WRAP UP: DIY Color Identifier
LOCATION: United States
PAGE: 28

INSIGHT: Linux Software Development Tools
LOCATION: United States
PAGE: 64

CIRCUIT CELLAR

THE WORLD'S SOURCE FOR EMBEDDED ELECTRONICS ENGINEERING INFORMATION

APRIL 2012
ISSUE 261

EMBEDDED PROGRAMMING

Build a Capacitive-Touch Personal Amp

An MCU for Low-Power Applications

Mesh Networking with the SNAP OS

An Examination of Parallel Capacitors

Sigma-Delta Modifiers Explained



PLUS

Software Simplified

Automation Reduces Repetitive Tasks

// Automatic Code Generators
// Configuration Control Programs
// Testing, Verification, & Validation
// And More

04>
7 25274 75349 9
$7.50 U.S. ($8.50 Canada)
www.circuitcellar.com

The NetBurner SB70LC



$49.00
Qty. 100
SB70LC Board

The complete hardware and software solution

- Low-cost
- 10/100 Ethernet
- Two Serial Ports
- 32-bit Performance
- Digital I/O
- SSH/SSL Option
- Flash Card Ready
- I²C • SPI Interface
- Example applications



$99.00
SB70LC Development Kit

The SB70LC Development Kit

The NetBurner SB70 LC Development Kit is available to customize any aspect of operation including web pages, data filtering, or custom network applications. The kit includes platform hardware, ANSI C/C++ compiler, TCP/IP stack, web server, e-mail protocols, RTOS, flash file system, Eclipse IDE, debugger, and cables. The kit enables you to communicate with peripherals that use SD/MMC Flash Card (including SDHC), SPI, I²C, or the general purpose digital I/O interface. The NetBurner security suite option includes SSH v1, v2 and SSL support.

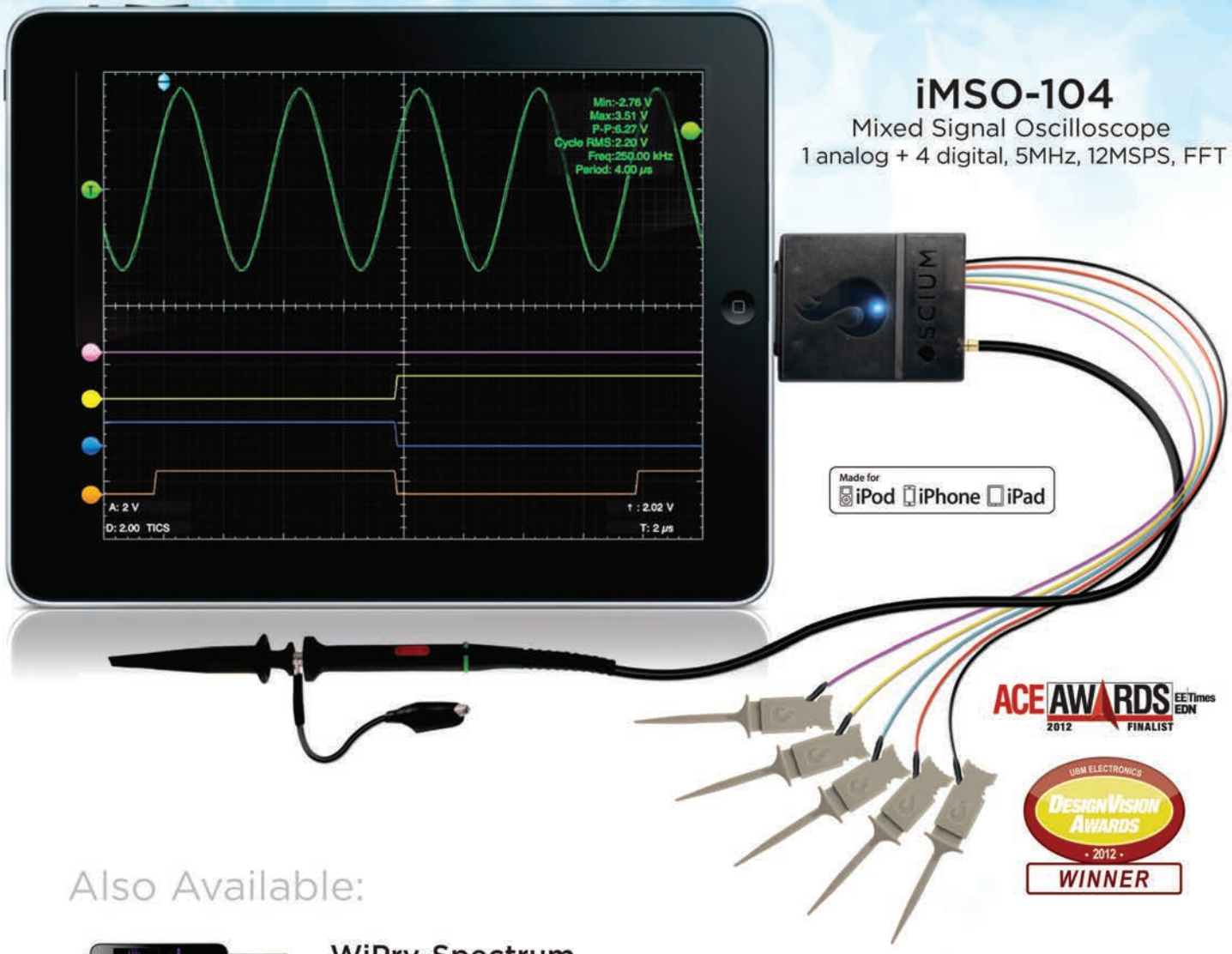
NetBurner
Networking in One Day!

Board Part Number | SB70LC-100IR
Development Kit Part Number | NNDK-SB70LC-KIT
Information and Sales | sales@netburner.com
Web | www.netburner.com
Telephone | 1-800-695-6828

freescale™
Alliance Member

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

Your iPad, iPhone or iPod becomes the instrument



iMSO-104

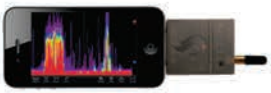
Mixed Signal Oscilloscope

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

ACE AWARDS 2012 FINALIST EETimes EDN

UBM ELECTRONICS DESIGNVISION AWARDS · 2012 · WINNER

Also Available:



WiPry-Spectrum

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

Resolution 2.0dBm, Sweep Time 200ms



WiPry-Power

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

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



WiPry-Combo

WiPry-Spectrum & WiPry-Power Combined!

OSCIUM

www.oscium.com

TASK MANAGER

Print and Online Progress

You'll notice some changes this month to *Circuit Cellar* magazine and our website. They're all for the better.

The magazine has an updated layout. We haven't changed fonts or style, but we did add the imprint you can see on pages 6–7. Its purpose is to show you that we are an ever-growing international company dedicated to bringing you essential information on a variety of important advanced electronics topics.

On this page you see that I added our editorial calendar, as well as a brief summary of the content we have in queue. The idea is to give you a clear idea of what we will cover and when you can expect it. Members frequently ask for this information, so it makes sense to make it easily accessible for everyone.

As for CircuitCellar.com, well, you've likely watched it change slowly over the past few months. We did this purposely. We developed the site in stages so readers wouldn't be burdened with dead links and redirects. So, what's new about the site?

The layout is a bit different. A few things are quickly apparent. One, the site is markedly brighter and easy on the eyes. Two, we created three distinct columns that provide you with easy access to handy articles, digital downloads, and more (see below). Three, we're tagging and categorizing all the content on our site. Thus, you'll find targeting specific information to be uncomplicated and immediately gratifying.

What sorts of content can you expect? The old site was fairly static. We'd make several changes each month and we'd run a few viewable articles. Now we're constantly posting relevant content of all sorts. This means you can rely on CircuitCellar.com for all of your electronics engineering needs: DIY articles, engineering tips, industry news, product reviews, vendor information, issue previews, links to source code, and even job openings in electronics engineering and embedded design.

If you are constantly plugged in, you'll find our website makes accessing your digital membership a cinch: just point and click to log in and download each issue! Plus, you can add our site to your RSS reader and read our content at your convenience (www.circuitcellar.com/feed/rss).

We are not finished building CircuitCellar.com. In the coming weeks and months, we'll enable more social interaction, post more videos, and broaden our areas of coverage. I suggest you visit our site each day to get your fix of embedded technology news and info. And please recommend the site to colleagues, friends, and others who have a passion for microcontrollers, programming, and everything else that's "inside the box."

cj@circuitcellar.com

C. Abate

CIRCUIT CELLAR®

THE WORLD'S SOURCE FOR EMBEDDED ELECTRONICS ENGINEERING INFORMATION

EDITORIAL CALENDAR

***

ISSUE	THEME
258 January	Embedded Applications
259 February	Wireless Communications
260 March	Robotics
261 April	Embedded Programming
262 May	Measurement & Sensors
263 June	Communications
264 July	Internet & Connectivity
265 August	Embedded Development
266 September	Data Acquisition
267 October	Signal Processing
268 November	Analog Techniques
269 December	Programmable Logic

Analog Techniques: Projects and components dealing with analog signal acquisition and generation (e.g., EMI/RF reduction, high-speed signal integrity, signal conditioning, A/D and D/A converters, analog programmable logic, and low-power, single-supply, mixed-voltage designs)

Communications: Projects that deal with computer networking, human-to-human interaction, human-to-computer interaction, and electronic information sharing (e.g., speech recognition, information transmission, network protocols, Ethernet, USB, I²C, and SPI)

Data Acquisition: Projects, technologies, and algorithms for real-world data gathering and monitoring (e.g., peripheral interfaces, sensors, sensor networks, signal conditioning, A/D and D/A converters, data analysis, and post-processing)

Embedded Applications: Projects that feature embedded controllers and MCU-based system design (e.g., automotive applications, test equipment, simulators, consumer electronics, real-time control, and low-power techniques)

Embedded Development: Tools and techniques used to develop new hardware or software (e.g., prototyping and simulation, emulators, development tools, programming languages, HDL, real-time operating systems, debugging tools, and useful tips and tricks)

Embedded Programming: The software used in embedded applications, as well as algorithms, tools, and techniques (e.g., programming languages, real-time operating systems, embedded Linux, file systems, drivers, network protocols, algorithms, and optimization)

Internet & Connectivity: Applications that deal with connectivity and Internet-enabled systems (e.g., networking chips, protocol stacks, device servers, and physical layer interfaces)

Measurement & Sensors: Projects and technologies that deal with sensors, interfaces, and actuators (e.g., environmental sensors, smart sensors, one-wire sensors, MEMS sensors, and sensor interface techniques)

Programmable Logic: Projects that utilize FPGAs, PLDs, and other programmable logic chips (e.g., dynamic reconfiguration, memory, and HDLs)

Robotics: Projects about robot systems, devices capable of repeating motion sequences, and MCU-based motor control designs (e.g., mobile robots devices, motor drives, proximity sensing, power control, battery technology/management, electronic compasses, and accelerometers)

Signal Processing: Projects, technology, and algorithms related to the real-time processing of signals (e.g., DSP chips and algorithms, signal conditioning, A/D and D/A converters, filters, compression, and comparisons of RISC, DSP, VLIW, etc.)

Wireless Communications: Technology and methods for going wireless (e.g., radio modems, Wi-Fi/IEEE 802.11x, Bluetooth, ZigBee/IEEE 802.15.4, cellular, infrared/IrDA, and security applications)

UPCOMING IN CIRCUIT CELLAR

***

FEATURES

The menU Universal Menu System, by Kyle Gilpin
Audio-Enhanced Touch Sensing, by Matt Oppenheim
Multilevel Audible Logic Probe, by Petre Tzetanov Petrov

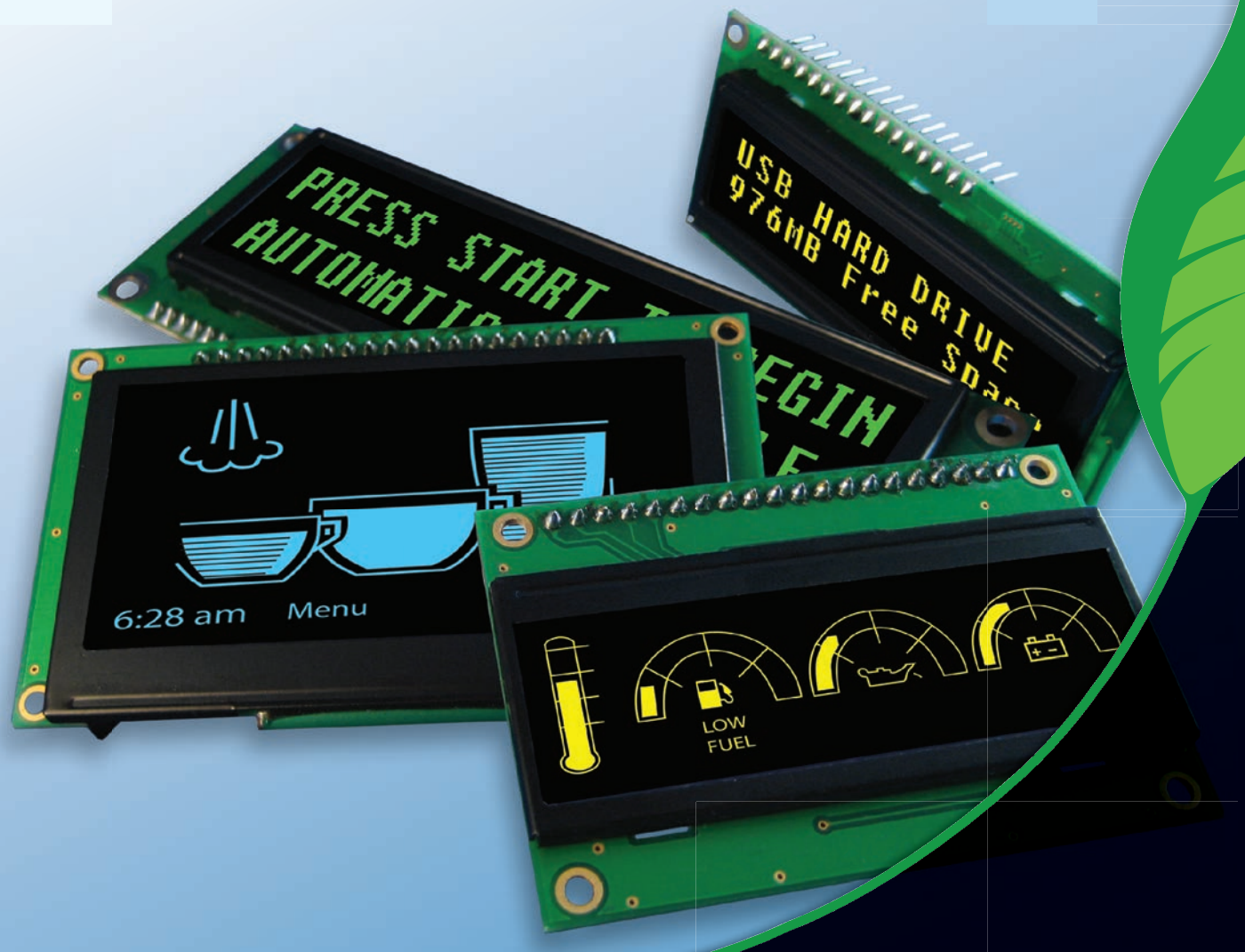
COLUMNS

SNAP to It (Part 2): SNAP Engine Versatility, by Jeff Bachiochi
Low-Power and Energy-Harvesting Applications, by George Martin
Accelerated Testing: A Review of Various Test Types, by George Novacek
One-Time Passwords from Your Watch, by Patrick Schaumont
Camera Image Stabilization, by Richard Wotiz

NEWHAVEN DISPLAY
INTERNATIONAL

New
Character and Graphic OLED Displays

- •Eco-Friendly Design
- •Self-Illuminated
- •Faster response
- •Better contrast
- •Wider viewing angles



NEWHAVEN DISPLAY INTERNATIONAL

Newhaven Display's OLED (Organic Light Emitting Diode) is a brighter, higher contrast display that has faster response times, wider viewing angles and consumes less power than conventional LCD, LED or VFD Displays. OLED displays are self-illuminating so no backlight is required for maximum visibility in all environments. This also allows OLEDs to be significantly thinner than standard VFD, LED or LCD displays. Newhaven Display offers Character and Graphic type OLED displays as complete easy-to-use modules.

For more information visit:
circuitcellar.newhavendisplay.com
Contact us at 847-844-8795

RoHS Compliant

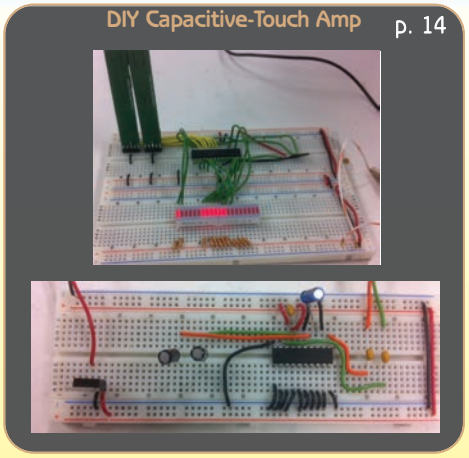
INSIDE ISSUE 261

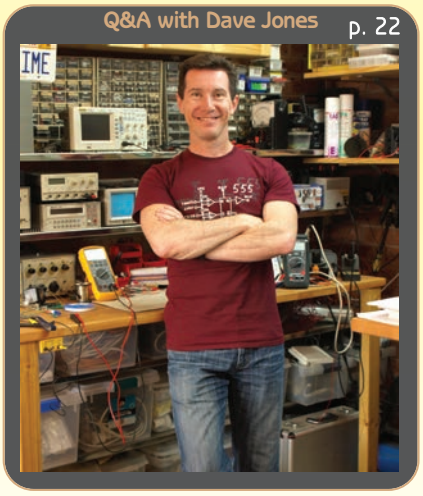
April 2012 • Embedded Programming

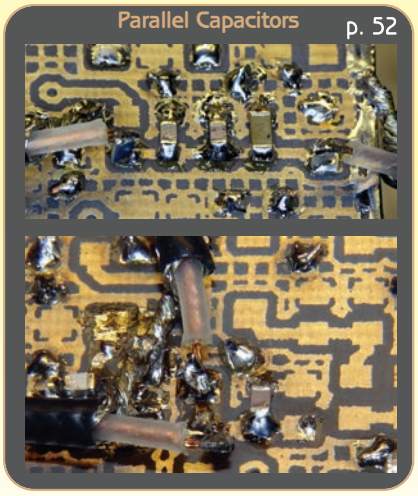
14 **Capacitive-Touch Gain-Controlled Personal Amplifier**
Coleton Denninger & Jeremy Lichtenfeld

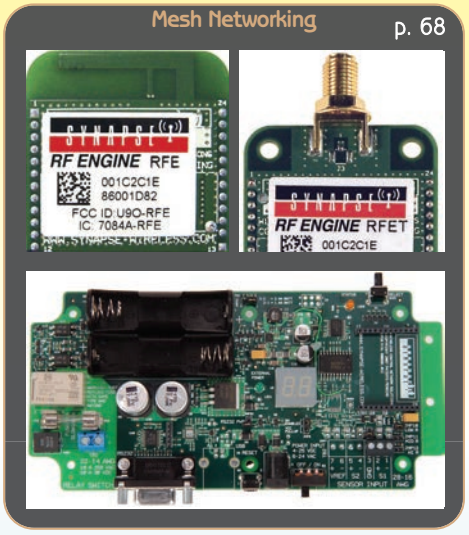
28 **The Rainbow Color Reader (Part 2)**
Color Distinction and Classification
Charles Edmondson

34 **The RL78 Microcontroller**
An MCU Family for Low-Power Applications
Clemens Valens









42 **THE DARKER SIDE**
Sigma-Delta Modulators 101
Robert Lacoste

52 **ABOVE THE GROUND PLANE**
RFI Bypassing (Part 2)
Parallel Capacitors
Ed Nisley

60 **THE CONSUMMATE ENGINEER**
Is Software Development an Art or a Science?
George Novacek

64 **EMBEDDED IN THIN SLICES**
Getting Started with Embedded Linux (Part 4)
Linux Software Development Tools
Bob Japenga

68 **FROM THE BENCH**
SNAP to It (Part 1)
Mesh Networking Simplified
Jeff Bachiochi

TASK MANAGER **2**
Print and Online Progress
C. J. Abate

NEW PRODUCT NEWS **10**

TEST YOUR EQ SOLUTIONS **13**

QUESTIONS & ANSWERS **22**
Electronics Engineering for the People
An Interview with David L. Jones

CROSSWORD **76**

PRIORITY INTERRUPT **80**
18-Bit Control Is Dead—No Way!
Steve Ciarcia

Scan Here

mouser.com
Semiconductors and electronic components for design engineers.

AS9120A Certified Distributor

The **most** advanced, multilingual, multicurrency mobile site for design engineers.



mousermobile™

Compatible with more than 25 mobile platforms, no one supports more phones and tablets than Mouser. Get What's Next now at **m.mouser.com**

mouser.com | The Newest Products for Your Newest Designs®

MOUSER ELECTRONICS®
a tti company

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 team

FOUNDER/EDITORIAL DIRECTOR:	Steve Ciarcia	CONTROLLER:	Jeff Yanco
EDITOR-IN-CHIEF:	C. J. Abate	PUBLISHER:	Hugo Van haecke
ASSOCIATE EDITOR:	Nan Price	ASSOCIATE PUBLISHER:	Shannon Barraclough
CONTRIBUTING EDITORS:	Jeff Bachiochi, Bob Japenga, Robert Lacoste, George Martin, Ed Nisley, George Novacek, Patrick Schaumont, Richard Wotiz	ART DIRECTOR:	KC Prescott
		CUSTOMER SERVICE:	Debbie Lavoie
		ADMINISTRATIVE COORDINATOR:	Valerie Luster
		MARKETING REPRESENTATIVE:	Amanda Anderson
		GRAPHIC DESIGNER:	Nordian Davis
PROJECT EDITORS:	Ken Davidson, David Tweed	ADVERTISING COORDINATOR:	Kim Hopkins

The network

audioXPRESS

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

elektor

techthefuture
Tech the Future explores the solutions for a sustainable future provided by technology, creativity and science.

VOICE COIL

Our international teams

United Kingdom
Wisse Hettinga
+31 (0)46 4389428
w.hettinga@elektor.com

USA
Hugo Van haecke
+1 860 875 2199
h.vanhaecke@elektor.com

Germany
Ferdinand te Walvaart
+49 (0)241 88 909-0
f.tewalvaart@elektor.de

France
Denis Meyer
+31 (0)46 4389435
d.meyer@elektor.fr

Netherlands
Harry Baggen
+31 (0)46 4389429
h.baggen@elektor.nl

Spain
Eduardo Corral
+34 91 101 93 85
e.corral@elektor.es

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

Sweden
Wisse Hettinga
+31 (0)46 4389428
w.hettinga@elektor.com

Brazil
João Martins
+351214131600
joao.martins@editorialbolina.com

Portugal
João Martins
+351214131600
joao.martins@editorialbolina.com

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

Russia
Nataliya Melnikova
8 10 7 (965) 395 33 36
nataliya-m-larionova@yandex.ru

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

South Africa
Johan Dijk
+27 78 2330 694 / +31 6 109 31 926
J.Dijk @elektor.com

China
Cees Baay
+86 (0)21 6445 2811
CeesBaay@gmail.com

Issue 261 April 2012 ISSN 1528-0608

CIRCUIT CELLAR® (ISSN 1528-0608) is published monthly by Circuit Cellar Incorporated, 4 Park Street, Vernon, CT 06066. Periodical rates paid at Vernon, CT and additional offices. One-year (12 issues) subscription rate USA and possessions $50, Canada $65, Foreign/ROW $75. All subscription orders payable in U.S. funds only via Visa, MasterCard, international postal money order, or check drawn on U.S. bank.

Cover photography by Chris Rakoczy—www.rakoczyphoto.com

Subscriptions
Circuit Cellar, P.O. Box 462256, Escondido, CA 92046
E-mail: circuitcellar@pcspublink.com
Phone: 800.269.6301, Internet: www.circuitcellar.com
Address Changes/Problems: circuitcellar@pcspublink.com

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

US Advertising
Strategic Media Marketing, Inc.
2 Main Street, Gloucester, MA 01930 USA
Phone: 978.281.7708, Fax: 978.281.7706, E-mail: peter@smmarketing.us
Internet: www.circuitcellar.com
Advertising rates and terms available on request.

New Products: New Products, Circuit Cellar, 4 Park Street, Vernon, CT 06066, E-mail: newproducts@circuitcellar.com

Membership Counter

We now have 275132 members in 84 countries...

Not a member yet?

Sign up at www.circuitcellar.com

Supporting Companies

Advanced Design Systems 65
All Electronics Corp. 78
AP Circuits 48
ARM Ltd. 46
Atria Technologies, Inc. 77
Bitscope Designs 17
Busboard Prototype Systems 79
CC Gold 40, 41
Comfile Technology, Inc. 51
Custom Computer Services, Inc. 78
Decade Engineering 77
DesignSpark chipKIT™ Challenge 47
Earth Computer Technologies 45
Elektor. 58, 59
Elektor. 67
Elprotronic, Inc. 13
Elsevier, Inc. 19
EMAC, Inc. 27
Express PCB 49
FlexiPanel Ltd. 77
FTDI Chip C3
Grid Connect, Inc. 27
HannoWare. 37
Humandata Ltd. 33
Imagineering, Inc. C4
Ironwood Electronics 78
Jameco Electronics 11
Jeffrey Kerr, LLC. 37
JK microsystems, Inc. 21, 79
Labcenter Electronics 31
Lakeview Research 37
Lemos International Co. 45, 78
LinkSprite 78
MaxBotix, Inc. 79
MCC, Micro Computer Control 77
Microchip Technolnogy, Inc. 25
microEngineering Labs, Inc 79
Mosaic Industries 79
Mouser Electronics, Inc. 5
NetBurner C2
Newhaven Display International. 3
Oscium 1
PCB Pool 13
Pico Technology Ltd. 70
Pololu Corp. 55
Reach Technology, Inc. 78
Renesas RL78 Green Energy Challenge . . 39
Saelig Co., Inc. 73
Scidyne. 78
Sealevel Systems 32
Sensors Expo & Conference 63
SiliconRay. 21
Synapse Wireless 71
Technologic Systems 8, 9
Tern, Inc. 77
Total Phase, Inc. 50
Triangle Reasearch International, Inc. . . . 78

Not a supporting company yet?

Contact Peter Wostrel (peter@smmarketing.us, Phone 978.281.7708, Fax 978.281.7706) to reserve your own space for the next issue of our member's magazine.

Head Office
Circuit Cellar, Inc.
4 Park Street, Vernon, CT 06066, Phone: 860.875.2199

Copyright Notice
Entire contents copyright © 2012 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.

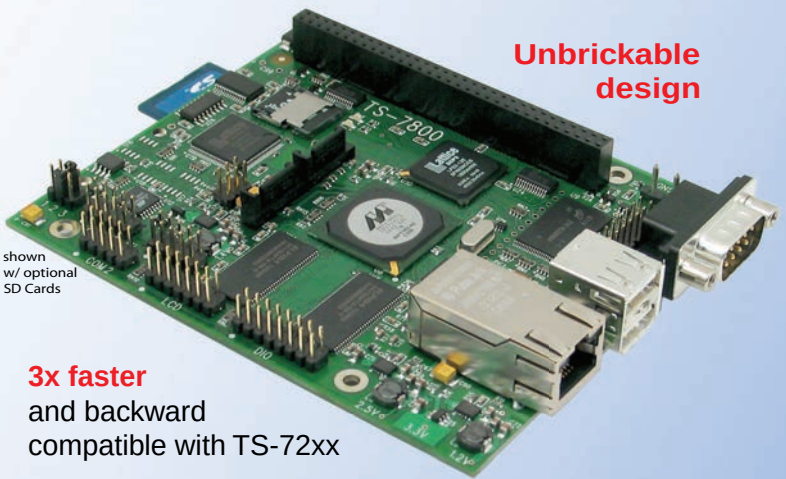
Disclaimer
Circuit Cellar® makes no warranties and assumes no responsibility or liability of any kind for errors in these programs or schematics or for the consequences of any such errors. Furthermore, because of possible variation in the quality and condition of materials and workmanship of reader-assembled 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 2012 Printed in the United States

Embedded Systems

High-End Performance with Embedded Ruggedness



Unbrickable design

3x faster
and backward
compatible with TS-72xx

TS-7800
500 MHz ARM9

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

TS-TPC-7390
7" Color Touch Panel Computer

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



$449 qty 1

More Touch Panel Computers on our website

- Over 25 years in business
- Open Source Vision
- Never discontinued a product
- Engineers on Tech Support

- Custom configurations and designs w/ excellent pricing and turn-around time
- Most products stocked and available for next day shipping

Design your solution with one of our engineers (480) 837-5200

New Products

TS-WIFIBOX-2
A Complete Solution for 802.11g WiFi Applications

NEW!

$185 qty 1

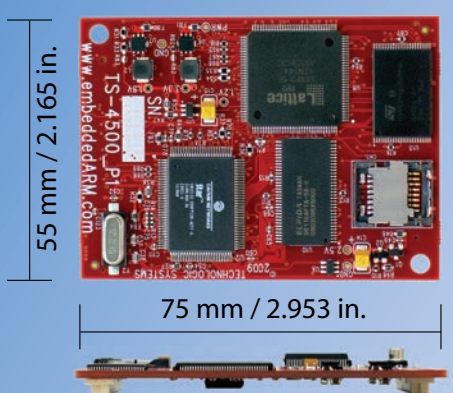


- Low power (3.2 watts), fanless
- Power via 5-12 VDC, USB, PoE (opt.)
- 64MB DDR-RAM
- 256MB ultra-reliable XNAND drive
- Micro-SD Card slot
- RS-232, RS-485, CAN, RTC
- Header with SPI and 11 DIO
- 480Mbit/s USB, Ethernet, PoE option
- Boots Linux 2.6.24 in < 3 seconds
- Un-brickable, boots from SD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

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

NEW!

series starts at $92 qty 100



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

TS-SOCKET Macrocontrollers
Jump Start Your Embedded System Design

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

Current projects include:

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

Technologic Systems

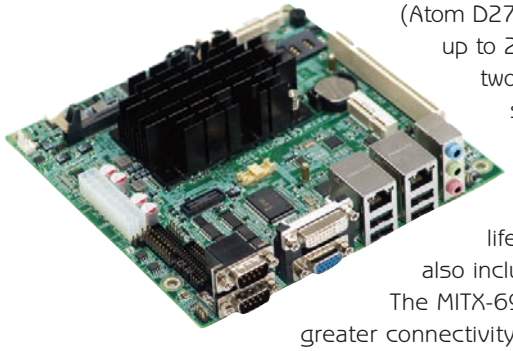
We use our stuff.

Visit our TS-7800 powered website at

www.embeddedARM.com

NEXT-GENERATION INTEL ATOM N2800/D2700 WITH 3G MINI-ITX BOARD

The **MITX-6930** is a mini-ITX embedded board based on the Intel NM10 chipset that is equipped with the next-generation dual core Intel Atom N2800 or D2700 processor. The environmentally friendly embedded board solution offers scalability in a small footprint fanless solution. The MITX-6930 is designed for digital signage, point-of-sale (POS), kiosks, enterprise security, digital surveillance, medicine, and factory automation.



Offering greater leverage for longer-life solutions, the MITX-6930 runs up to 2.13 GHz (Atom D2700) with I/O interfaces including: four USBs, two COM ports, two LANs, one VGA up to 2048 × 1536, one DVI up to 1920 × 1200, and 7.1 HD audio I/Os. Onboard are two Mini-PCIe slots, one Mini-PCIe with SSD support, one Mini-PCIe with SIM card socket for 3G/4G connectivity, one PCI slot, two USB 2.0, four COM ports, one LPT, one LPC, one DDR3 support up to 4-GB memory, two SATA, HDMI pin, and a LVDS display output up to 1366 × 768. The MITX-6930 operates in extended temperatures of –20°C to 70°C. Its processors include seven-year extended lifecycle support to protect development investment. The system architecture also includes many custom design options for OEM/ODM solution development.

The MITX-6930 features a small carbon footprint of 170 mm × 170 mm, and offers greater connectivity with dual Intel 82574L Gigabit Ethernet with options of expansion via PCI slot onboard, two serial ports for factory automation, and two mini-PCIe slots onboard for Wi-Fi and 3G/4G cellular connection. Onboard features for scalability also include a three-pin CPU fan, a watchdog timer (WDT), an 8-bit GPIO, and ATX 20-pin standard power connector.

Contact HABEY for pricing.

HABEY USA, Inc.
www.habeyusa.com

ACCURATE ULTRASONIC DISTANCE SENSING

The **HRLV-MaxSonar-EZ** sensor line features 1-mm resolution, target-size compensation for improved accuracy, simultaneous automatic multisensor operation, rejection of outside noise sources, temperature compensation, and a 5-m range. The sensors add TTL serial output to the RS-232, pulse width, and analog voltage outputs that are standard on other MaxSonar products.



The HRLV-MaxSonar-EZ sensor line is capable of compensating for target size differences. For example, if an object is large enough to be detected, the sensor will report the same distance regardless of target size. The reported range of the sensors is unaffected by other sensors, so MaxSonar-EZ sensors can be used with other sensors in close proximity.

In addition, when compared with other MaxSonar products, the noise filtering of the HRLV-MaxSonar-EZ sensors will work in the presence of more noise sources and outside noise sources with higher amplitudes. While most range readings are accurately reported, if the range readings are affected, the effect is typically less than 2 cm.

The HRLV-MaxSonar-EZ sensors are capable of accurate temperature compensation. The speed of sound changes about 0.6% per °C. The HR-MaxTemp is an external temperature sensor that can be easily connected to an HRLV-MaxSonar sensor. Once connected, the HRLV-MaxSonar sensor automatically detects the presence of the external temperature sensor and compensates for temperature effects. The HR-MaxTemp is available as an option for $4.95.

The HRLV-MaxSonar-EZ comes with the easy-to-use outputs and standard pin configuration of the prior MaxSonar products. In addition to the three standard sensor outputs of RS-232 serial, analog voltage, and pulse width, there is now a user-selectable TTL serial output. The sensors offer a controlled beam designed to reject small objects (clutter) outside of the central beam area and provide long-range sensitivity without detecting unwanted targets.

HRLV-MaxSonar-EZ sensors are less than 1 cubic inch, weigh 4.3 g, and operate from 3 V to 5 V. The sensors cost **$34.95**.

MaxBotix, Inc.
www.maxbotix.com

Can You Solve This?

Forrest M. Mims III Puzzler



Can you describe any of the three methods Joe used to detect the aerosol?



Satellites provide global coverage of clouds, water vapor, dust, smoke and the ozone layer. The colorful images provided by the data from these satellites looks very impressive. But satellite instruments don't always stay calibrated and problems can occur when satellite orbits drift. Amateur scientist, Joe Novice learned about this when he heard a satellite scientist say that the global aerosol cloud formed by the eruption of a giant volcano had dissipated much sooner than expected. Joe suspected the satellite was simply wrong, but he was not a satellite scientist. How did he use some everyday items and several electronic components to prove he was right?

What's your solution? See if you are correct at ***www.Jameco.com/teaser13*** where you will find all three of Joe's solutions.

Order Your Free Jameco Catalog!
Jameco.com/catalog

JAMECO® ELECTRONICS
1-800-831-4242

HIGH-PERFORMANCE BOARDS WITH FASTER PROCESSORS

The ESMini series now includes the **MM2**, an ultra-small computer-on-module (COM) that features an application-specific carrier board and upgraded processor. The semi-custom MM2 is only 95 mm × 55 mm. Combined with the Intel Atom E600 series processor and an EMC-proof enclosure, the board is well suited for industrial, harsh, mobile, and mission-critical environments with high graphics requirements in small spaces. These applications include avionics, railway, agricultural or construction machines, medical engineering, and industrial automation.



The Intel Atom processor offers a frequency up to 1.6 GHz, total power consumption of 5 W to 7 W maximum, and high I/O flexibility using the PCI Express standard for the processor-to-chip interface. The MM2 is capable of operating in environments from –40°C to 85°C and it provides multiple I/O options for a range of specific end-user requirements.

Supporting both serial and legacy I/O, the board offers two PCI Express x1 links, low-voltage differential signaling (LVDS) and serial digital video out (SDVO) for graphic interfaces, high-definition audio, Ethernet, SATA, USB, two I²C, CAN bus, and COM interfaces.

The MM2 module can accommodate up to 2-GB double data rate synchronous dynamic random-access memory (DDR2 SDRAM) of directly soldered main memory with mass storage media supported on the carrier board. Each processor includes 512 KB of L2 cache. The rugged, compact COM comes with a real-time clock (RTC) and a board-management controller for temperature and power supervision.

ESMini boards incorporate multiple design features for extreme environmental concerns including a rugged, shock-resistant, vibration-resistant housing that provides EMC protection and enables conductive cooling. High-pressure screw connections between the housing and carrier printed circuit board (PCB) facilitate thermal connection of components. The housing also can be connected to an external heat dissipation system (conduction) or combined with a heatsink on the top cover for heat dissipation (convection) if additional cooling is needed. The low-power version may also be operated without the frame and cover in moderate application temperatures. MM2 modules come equipped with rugged connectors supporting high-frequency and differential-signal connections. Conformal coating is also available.

The MM2 ESMini module costs **$805**.

MEN Micro, Inc.
www.menmicro.com

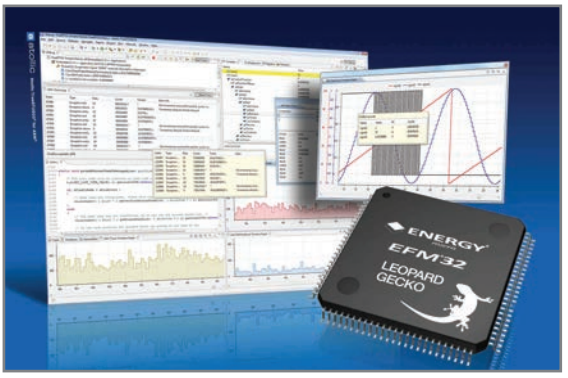
DEVELOPMENT TOOL SUPPORTS ENERGY MICRO EFM32 FAMILY

The **TrueSTUDIO** for ARM C/C++ development tool supports all currently available Energy Micro EFM 32 Cortex-M3 families, including 160 devices in the Tiny Gecko, Gecko, Giant Gecko, and Leopard Gecko families.

The TrueSTUDIO project wizard is capable of auto-generating sample projects for Energy Micro EFM32 evaluation boards, which enables developers to get up and running with new C/C++ projects on Energy Micro EFM32 boards.

The TrueSTUDIO development and debugging tool offers a state-of-the-art editor, an optimizing C/C++ compiler, and a multiprocessor-aware debugger with real-time tracing. The tool suite features ARM build and debug tools, system analysis and real-time tracing using Serial Wire Viewer (SWV) technology, and graphical unified modelling language (UML) diagram editors for model-based design and architecture.

The tool also offers an Eclipse-based integrated development environment (IDE) with a state-of-the-art editor, x86 C/C++ build and debug tools for development of PC command-line applications, parallel compilation and multiprocessor debugging, and an integrated version-control system client with revision graph visualization, which enables you to trace the history of code additions and revisions.



In addition, TrueSTUDIO includes an integrated client for accessing bug databases such as Trac and Bugzilla. It also includes integrated features for performing source code reviews and code review meetings.

Contact Atollic for pricing.

Atollic
www.atollic.com

Edited by David Tweed

Test Your EQ

ANSWERS for Issue 260

Answer 1—RS-232 was developed in the days when people were connecting electromechanical teletypes to telephone lines with modems, and in fact, at that time the design of any equipment connected to a phone line was tightly controlled by the phone company.

The reason RS-232 uses a negative voltage for its idle state is the same reason the phone lines themselves use a negative voltage relative to ground for power—copper wires in long cables potentially exposed to moisture are significantly less likely to corrode if they have a negative DC bias on them.

Answer 2—Again, back in the days when RS-232 was developed, the primary logic families in use were DTL and TTL. Both of these technologies draw significantly less power when a signal is in the high state than in the low state, so the high state is preferred for the inactive state of any signal.

Answer 3—Remember the algorithm that computes integer square roots by subtracting successive odd numbers from the input value? This function extends that concept to computing integer cube roots.

The reason this works is that taking the differences between successive values is the discrete equivalent of taking a derivative in the continuous world. The derivative of a cubic curve is a quadratic, and the derivative of a quadratic is a straight line. To generate a "straight line" in the discrete world, you just add a constant to a variable.

When you look at successive squares—0, 1, 4, 9, 16, 25, etc.—the differences are 1, 3, 5, 7, 9, etc. This is why the algorithm that subtracts odd numbers works for computing square roots. When you look at the successive cubes—0, 1, 8, 27, 64, 125, etc.—the first set of differences is the sequence 1, 7, 19, 37, 61, etc. This doesn't look very useful until you take the differences between those numbers, which are: 6, 12, 18, 24, etc., which is obviously another straight line.

Answer 4—The direct way to convert numbers in an arbitrary fixed-point representation to the equivalent floating-point value is to figure out what the representation for "1.000" would be in the fixed-point notation and then divide the given numbers by that constant.

In this case, with 7 integer bits and 9 fraction bits, "1.000" would be represented as binary 0000001.000000000, or 0x0200. So, if you are given a constant of, say, 0x5453, just divide it by 0x0200 to find out that it represents the value "42.162."

Contributed by David Tweed

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

You may contact the quizmasters at eq@circuitcellar.com





by Coleton Denninger and Jeremy Lichtenfeld (Canada)

Capacitive-Touch Gain-Controlled Personal Amplifier

Two Camosun College graduates designed an inexpensive do-it-yourself personal audio amplifier. The design features capacitive sensing, digital-signal mixing, and high-quality sound.

When did you stop thinking about how consumer electronics looked on the outside and ask yourself what happens on the inside? For many consumers, unfortunately, this epiphany will never take place. Most people will live their lives buying, using, and disposing of electronics without ever knowing how they actually work. But for some, this will happen naturally and their curiosity will give them the energy to explore to their hearts' content. We intend to engage and inspire these inquisitive-minded tech enthusiasts.

While studying at Camosun College in Canada, we were tasked with creating a "no-experience-needed" electronics kit that would appeal to middle/high school students. The students would be given a crash course in soldering, so we had to make it as solder-friendly as possible. For this reason, we weren't allowed to use surface-mount components. We could only use through-hole components. This was because the last thing we needed was a bunch of eighth graders, buzzing on sugar, armed with heat guns! The kit also needed to be sleek and modern to get kids excited to build it, but without being too expensive or time consuming.

Since every kid and his dog owns an iPod, MP3 player, or some other type of personal audio device, it made sense to build a personal audio amplifier (see Photo 1). The tough choices were how we were going to make it stand out enough to attract kids who already own high-end electronics and how we were going to do it with a budget of around $40.

SHOW ME THE PARTS

Our first concern was how we were going to mix and amplify the low-power audio input signals from iPods, microphones, and electric guitars. We decided to have a couple of different inputs, and we wanted stereo and mono outputs. After doing some extensive research, we chose the Cypress Semiconductor CY8C29466-24PXI programmable system-on-chip (PSoC). This enabled us to digitally mix and vary the low-power amplification using the programmable gain amplifiers and switched capacitor blocks. It also came in a convenient 28-pin DIP package that followed our design guidelines. Not only was it perfect for our design, but the product and

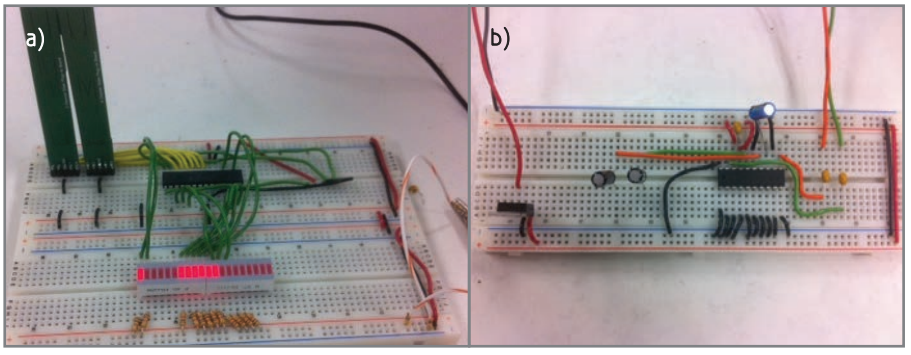


Photo 1a—This is the capacitive-touch stage of the personal audio amplifier. b—Here you see the power-amplification stage of the personal audio amplifier.

developer online support forums for all of Cypress's products were very helpful.

Let's face it: mechanical switches and potentiometers are fast becoming obsolete in the world of consumer electronics (not to mention costly when compared to other alternatives). This is why we decided to use capacitive-touch sensing to control the low-power gain. Why turn a potentiometer or push a switch when your finger comes pre-equipped with conductive electrolytes? We accomplished this capacitive touch using Microchip Technology's mTouch Sensing Solutions series of 8-bit microcontrollers. The project was originally done on the PIC16F1939 (40-pin DIP) microcontroller, but after we were confident that we could implement it on a smaller chip, we ended up going with the PIC16F1938 (28-pin DIP) microcontroller. This enabled us to have one capacitive slider for gain control and two capacitive buttons for choosing which input channel to control.

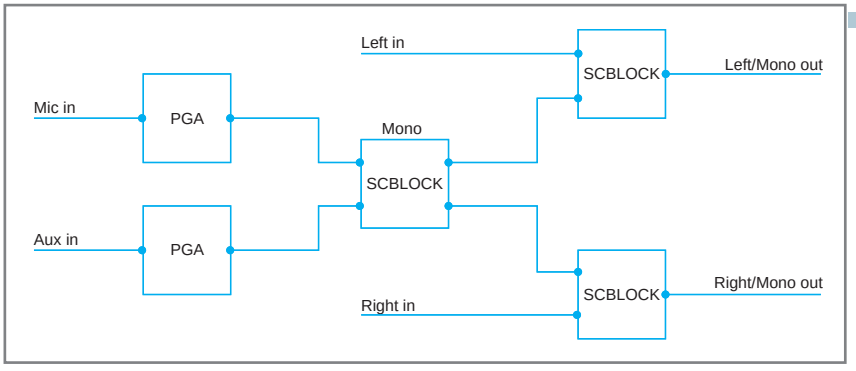


Figure 1—Audio mixer flowchart

Who doesn't like a little bit of a light show? We used the same aforementioned PIC, but implemented it as a voltage unit meter. This meter averaged out our output signal level and indicated via LEDs the peaks in the music played. Essentially, while you listen to your favorite beats, the amplifier will beat with you!

For this project, we used a total of three different microcontrollers. "Why didn't they just implement it all into one?" you might ask. We needed through-hole chips, which limited the amount of microcontrollers we had at our disposal. The capacitive sensing and VU meter all could have been implemented in one PIC, but due to time constraints (and the VU meter being a late addition), it was easier to just have it on two. We also wanted to show two totally different types of implementation styles. So, when students are taught about the amplifier they can get a good handle on the different types of products that are out there (e.g., PICs versus PSoCs).

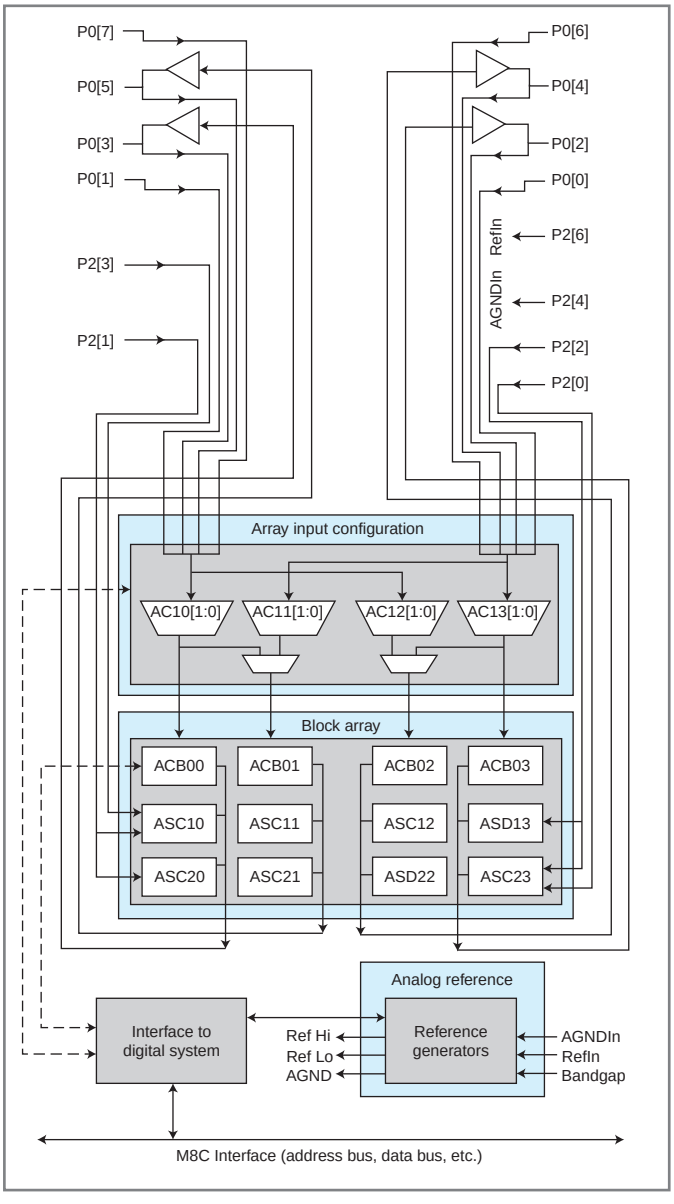


Figure 2—The analog system

BRING ON THE POWER

This amp needed to have a bit of kick when it came to the output. We're not talking about eardrum-bursting power, but we wanted to have decent quality with enough power to fill an average-sized room with sound. We decided to go with a Class AB audio amplifier—the TPA1517 from Texas Instruments (TI) to be exact. The TPA1517 is a stereo audio-power amplifier that contains two identical amplifiers capable of delivering 6 W per channel of continuous average power into a 4-Ω load. This quality chip is easy to implement. And at only a couple of bucks, it's an affordable choice!

KICK IT INTO REVERSE

"Wait a minute. So you're telling me by dragging a few funny-colored blocks around and writing a couple lines of code you can create a mixer with controllable gain via UART?" The simple answer is, yes! After wading through datasheets the size of a three-series novel just so we could set up a few simple I/O and peripheral functions, our prayers were answered! Thanks to Cypress's PSoC Designer 5.1, you can have what used to take a couple

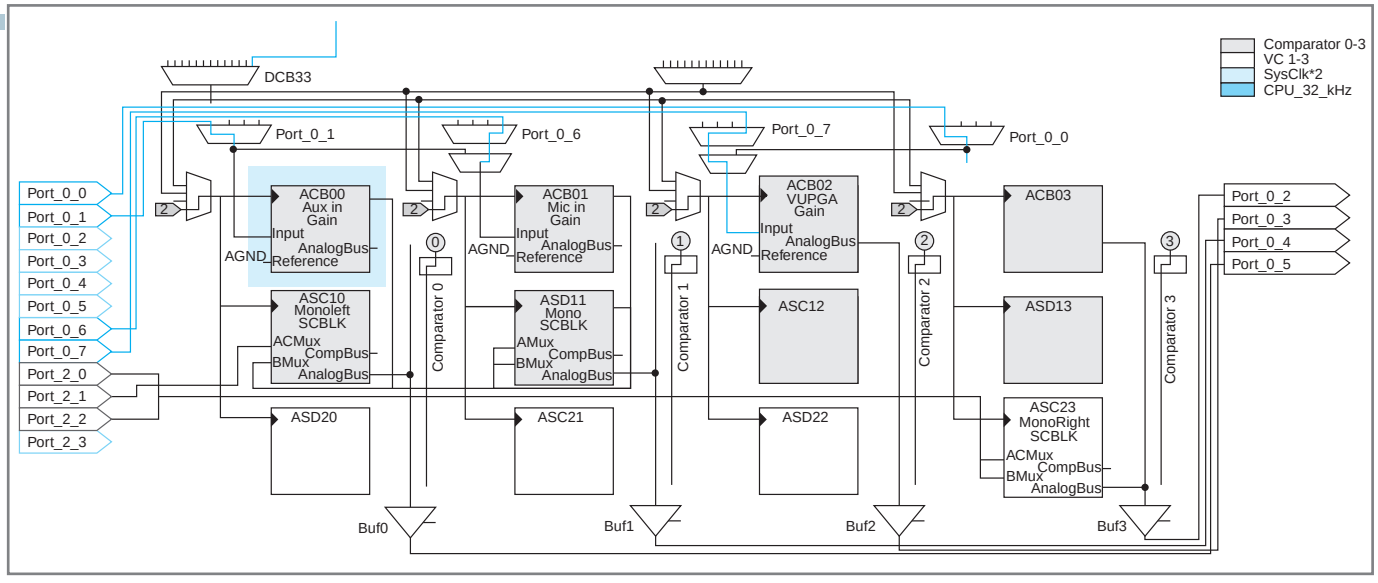


Figure 3—The analog user module block routing system

hours done within minutes! Its free Designer IDE enables you to quickly implement what you want how you want it. It has a huge library of precharacterized user modules, which is the equivalent to an embedded programmer's dream toolbox. Not only did it have everything we needed, but we only used a tiny fraction of the thousands of design possibilities available to the user.

The Designer IDE has a small learning curve. There are online tutorials that take you through making a simple project, as well as implementing some of the more advanced techniques available to the user. For this project, we used three different user modules: the programmable gain amplifier (PGA), the switched capacitor (SCBLOCK), and the universal asynchronous receiver/transmitter (UART) blocks. The PGAs were used to buffer the inputs, as well as to provide gain control over our inputs. We chose to use these for a microphone input and an auxiliary input. The SCBLOCKs were used to take in our player inputs, as well as to mix our microphone and auxiliary inputs together with our player inputs. Refer to Figure 1 for the physical representation of the mixer design flow. The UART user module comes complete with a transmitter block and a receiver block. We only needed the receiver because the mTouch controller would only be sending data to the PSoC.

The good thing about these user modules is that each of

Listing 1—This button decode routine enables us to store the data depending on which channel button is last selected. It also controls how the volume control meter works. (Listing continued on p. 17)

```
#elif defined SLIDER_FOUR_PLUS_8KEY
    signed long SliderTemp, PercentTemp;
    unsigned int SliderNumerator, SliderDenominator;
    unsigned int Aux;
    unsigned int Mic;

        SliderMax = SLIDER_MAX;

        //-------------------------------
        // Convert 4 button slider to 0-to-255 for a percentage
        // pressed.  LEDs light up like bar graph based on %
        //-------------------------------
        if(LED11 == 0){
        if (ChannelNumber == NumberChannels-1)
        {
            if (ButtonState[Button0] == Pressed || ButtonState[Button1]
            == Pressed || ButtonState[Button2] == Pressed ||
            ButtonState[Button3] == Pressed) {
                // Any of the 4ch slider sensors as pressed will trigger
                this slider to computation to occur
                Decode4ChSlider();
                TXREG = xabs;
                AuxStore = xabs;

            } else {
                xabs = 0xFFFF;
            }

            SliderPosition = xabs;
            Aux = xabs;
        }
             if (Aux != 0xFFFF) {
             if ( Aux > 25 )        LED1 = 0;     else      LED1 = 1;
             if ( Aux > 50 )        LED2 = 0;     else      LED2 = 1;
```

Listing 1—Continued from p. 16

```
        if ( Aux > 75 )         LED3 = 0;     else      LED3 = 1;
        if ( Aux > 100 )        LED4 = 0;     else      LED4 = 1;
        if ( Aux > 125 )        LED5 = 0;     else      LED5 = 1;
        if ( Aux > 150 )        LED6 = 0;     else      LED6 = 1;
        if ( Aux > 175 )        LED7 = 0;     else      LED7 = 1;
        if ( Aux > 200 )        LED8 = 0;     else      LED8 = 1;
        if ( Aux > 225 )        LED9 = 0;     else      LED9 = 1;
        if ( Aux > 250 )        LED10= 0;     else      LED10= 1;

    }
    else{
        if ( AuxStore > 25 )    LED1 = 0;     else      LED1 = 1;
        if ( AuxStore > 50 )    LED2 = 0;     else      LED2 = 1;
        if ( AuxStore > 75 )    LED3 = 0;     else      LED3 = 1;
        if ( AuxStore > 100 )   LED4 = 0;     else      LED4 = 1;
        if ( AuxStore > 125 )   LED5 = 0;     else      LED5 = 1;
        if ( AuxStore > 150 )   LED6 = 0;     else      LED6 = 1;
        if ( AuxStore > 175 )   LED7 = 0;     else      LED7 = 1;
        if ( AuxStore > 200 )   LED8 = 0;     else      LED8 = 1;
        if ( AuxStore > 225 )   LED9 = 0;     else      LED9 = 1;
        if ( AuxStore > 250 )   LED10= 0;     else      LED10= 1;
    }

    }

    if(LED11 == 1){
    if (ChannelNumber == NumberChannels-1)
    {

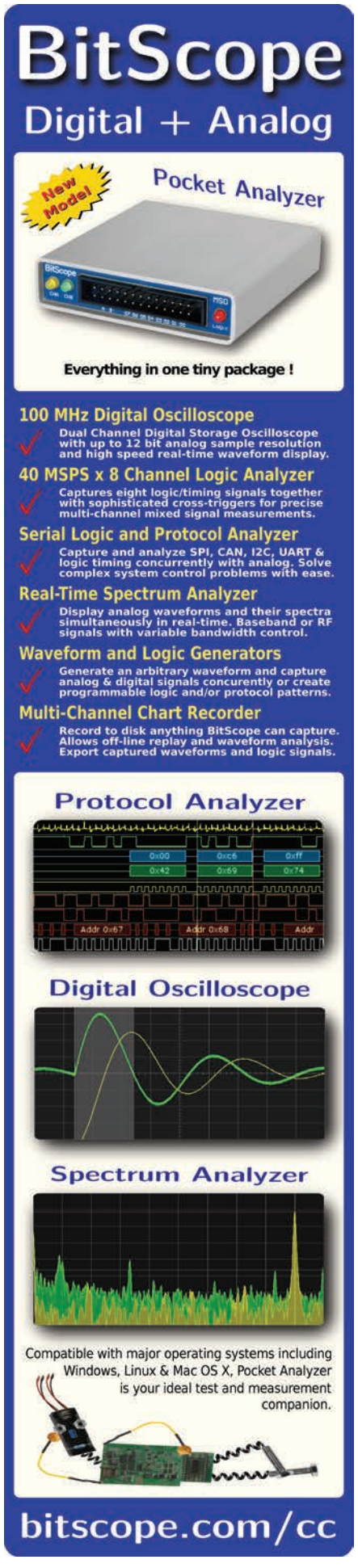
        if (ButtonState[Button0] == Pressed ||
        ButtonState[Button1] == Pressed || ButtonState[Button2]
        == Pressed || ButtonState[Button3] == Pressed) {
                // Any of the 4ch slider sensors as pressed
                will trigger this slider to computation to occur
                Decode4ChSlider();
                TXREG = xabs;
                MicStore = xabs;

        } else {
                xabs = 0xFFFF;
        }


        SliderPosition = xabs;
        Mic = xabs;
    }

    if (Mic != 0xFFFF) {
        if ( Mic > 25 )         LED1 = 0;     else      LED1 = 1;
        if ( Mic > 50 )         LED2 = 0;     else      LED2 = 1;
        if ( Mic > 75 )         LED3 = 0;     else      LED3 = 1;
        if ( Mic > 100 )        LED4 = 0;     else      LED4 = 1;
        if ( Mic > 125 )        LED5 = 0;     else      LED5 = 1;
        if ( Mic > 150 )        LED6 = 0;     else      LED6 = 1;
        if ( Mic > 175 )        LED7 = 0;     else      LED7 = 1;
        if ( Mic > 200 )        LED8 = 0;     else      LED8 = 1;
        if ( Mic > 225 )        LED9 = 0;     else      LED9 = 1;
        if ( Mic > 250 )        LED10= 0;     else      LED10= 1;

    }
    else{
        if ( MicStore > 25 )    LED1 = 0;     else      LED1 = 1;
        if ( MicStore > 50 )    LED2 = 0;     else      LED2 = 1;
        if ( MicStore > 75 )    LED3 = 0;     else      LED3 = 1;
        if ( MicStore > 100 )   LED4 = 0;     else      LED4 = 1;
        if ( MicStore > 125 )   LED5 = 0;     else      LED5 = 1;
        if ( MicStore > 150 )   LED6 = 0;     else      LED6 = 1;
        if ( MicStore > 175 )   LED7 = 0;     else      LED7 = 1;
        if ( MicStore > 200 )   LED8 = 0;     else      LED8 = 1;
        if ( MicStore > 225 )   LED9 = 0;     else      LED9 = 1;
        if ( MicStore > 250 )   LED10= 0;     else      LED10= 1;
    }
    }
```

them comes with a complete datasheet usually no longer than 10 pages. It tells you which resources the particular block uses, how it works, how to configure it, and most importantly how to use the application programming interface (API) routines. These come complete with sample code to get you up in running quickly in either C or Assembly languages. In a nutshell, you place the user modules you need, configure them to fit your design, and write the initialization code to start and power up the module. Seems pretty cut-and-dried, doesn't it?

THE PUZZLE BEGINS

As nice as the PSoC Designer 5.1 IDE sounds, it does have its limitations. After placing blocks in the design environment, it's necessary to connect them to input pins, connect certain blocks to others, and ultimately connect them to the output pins. This is easier said than done. Yes, we know there are probably a hundred ways we could have done our design, but we wanted to follow the design flow in Figure 1 as closely as possible. The design limitations only permit certain blocks to be configured to connect to other blocks. There are also certain inputs that are only available to certain blocks. To get a better understanding of this concept, the analog system block diagram is shown in Figure 2.

Since microphones and auxiliary devices usually don't have their own gain control, we knew we had to handle that with the PGA blocks. This gave us 100% control over the signal level from these inputs that entered the TI amplifier. Most personal music players have a volume control, so as much as it would be nice to have a capacitive volume control for this input, it wasn't entirely necessary. For this reason and because of block connection limitations, we decided to use SC blocks for these inputs.

After fighting with the design IDE to get the design exactly the way we wanted it, it always came down to that last connection you needed from block A to block B. The IDE routing limitations made some connections impossible. To say the least, we weren't going to let the machines win, even if it meant cheating. If we couldn't connect one block to another, we would just connect the output of a block to a pin, and do the same for the input of the block we wanted to connect to. Then we just externally jumped a wire from pin A to pin B. Take that, design limitations! Refer to Figure 3 for our analog block design layout.

Configuring the UART user module for the PSoC was a piece of cake. The user module datasheet says the clock to the UART should be eight times the baud clock. So, for a 9600 data rate (which we used), the clock to the UART should be 76.8 kHz (i.e., 8 × 9600). This can be easily derived by using divisions of the system clock. PSoC 1 enables the user to divide the 24-MHz clock almost every way needed. For a 9600 bps, we use a divider of 312 on the system clock, which results in a clock of 76.923 kHz against the requirement of 76.8 kHz. Pretty close and within an acceptable range of error, too.

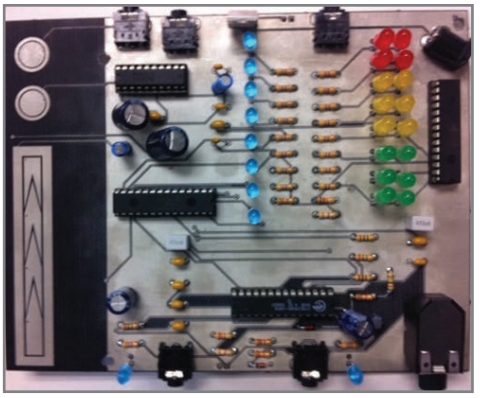


Photo 2—The mounted PCB

PUTTING CONDUCTIVE ELECTROLYTES TO WORK

Working with capacitive sensing was something we had never done before. Thanks to the mTouch Capacitive Touch Evaluation Kit from Microchip Technology, we were able to get up and running in a matter of minutes! The kit is a simple platform for developing a variety of capacitive-touch sensing applications using PIC16F, PIC18F, PIC24F, and PIC32 microcontrollers. Since we are using the PIC16F series of microcontrollers, this kit was a good way to help us understand how capacitive sensing works. The kit comes complete with preprogrammed boards and sample code that enables you to use capacitive-touch features right out of the box. Using the sample code as a base, we manipulated the code to start implementing our own features.

So, how does capacitive sensing work in conjunction with microcontrollers? Touching a copper sensor pad on a printed circuit board (PCB)—which has a nominal capacitance—creates an additional parallel capacitance essentially coupled to ground. This adds to the total overall capacitance generated by the touch sensor which is used to detect a finger press. The capacitance generated by the touch sensor is used in conjunction with a dual comparator with SR latch peripherals. This creates a relaxation oscillator that has a particular frequency of oscillation. The frequency of this oscillator is measured in a fixed interval using the Timer1 and Timer0 peripherals. Any shift in frequency caused by the user's touch is detected and validated in the software. Sounds easy right?

As previously mentioned, we had a capacitive slider and two capacitive buttons. The slider was used to control the gain in the PSoC and the buttons were used to determine which channel (either auxiliary or microphone) was being controlled. Using a four-section slider, the input data is converted to 0 to 255 for a percentage pressed. This then updates correlating display LEDs in a bar graph fashion. We used storage variables to store the value of the slider data with each different input. This meant when a button was pushed to go to a certain channel, its last changed value was kept and shown on the LED display. Refer to Listing 1 for the manipulated source code that enabled us to do this.

Configuring the UART for the PIC was a simple task. We just had to enable some bits as outlined in the datasheet and set the data rate to match what we set up

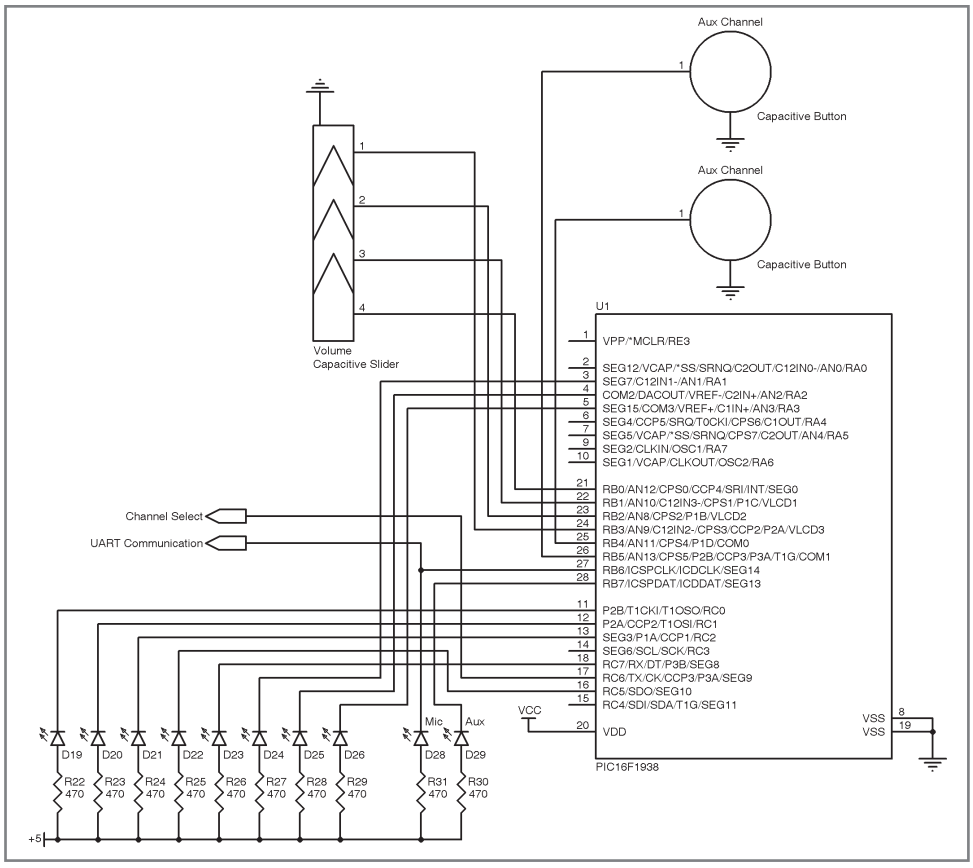


Figure 4—Schematic for the PodAmp (schematic continued on p. 20)

for our PSoC. We sent out the 8-bit data via the UART directly to the PSoC. This enabled us to talk directly from the PIC to the PSoC so we could reliably and accurately control the gain.

SLIDER & BUTTON DESIGN

We searched online for pre-built PCB footprints and found they were essentially nonexistent. This meant designing everything from scratch! After scouring datasheets and application notes, as well as reading about it online, we were confident that we could create a robust touch slider and button.

When creating the button, it seemed pretty straightforward as long as we followed one rule: the clearance between the pad and the ground plane should be at least 1/20 the diameter of the pad. After all the research we did, this was the one rule that will ensure your touch sensor is accurate and robust.



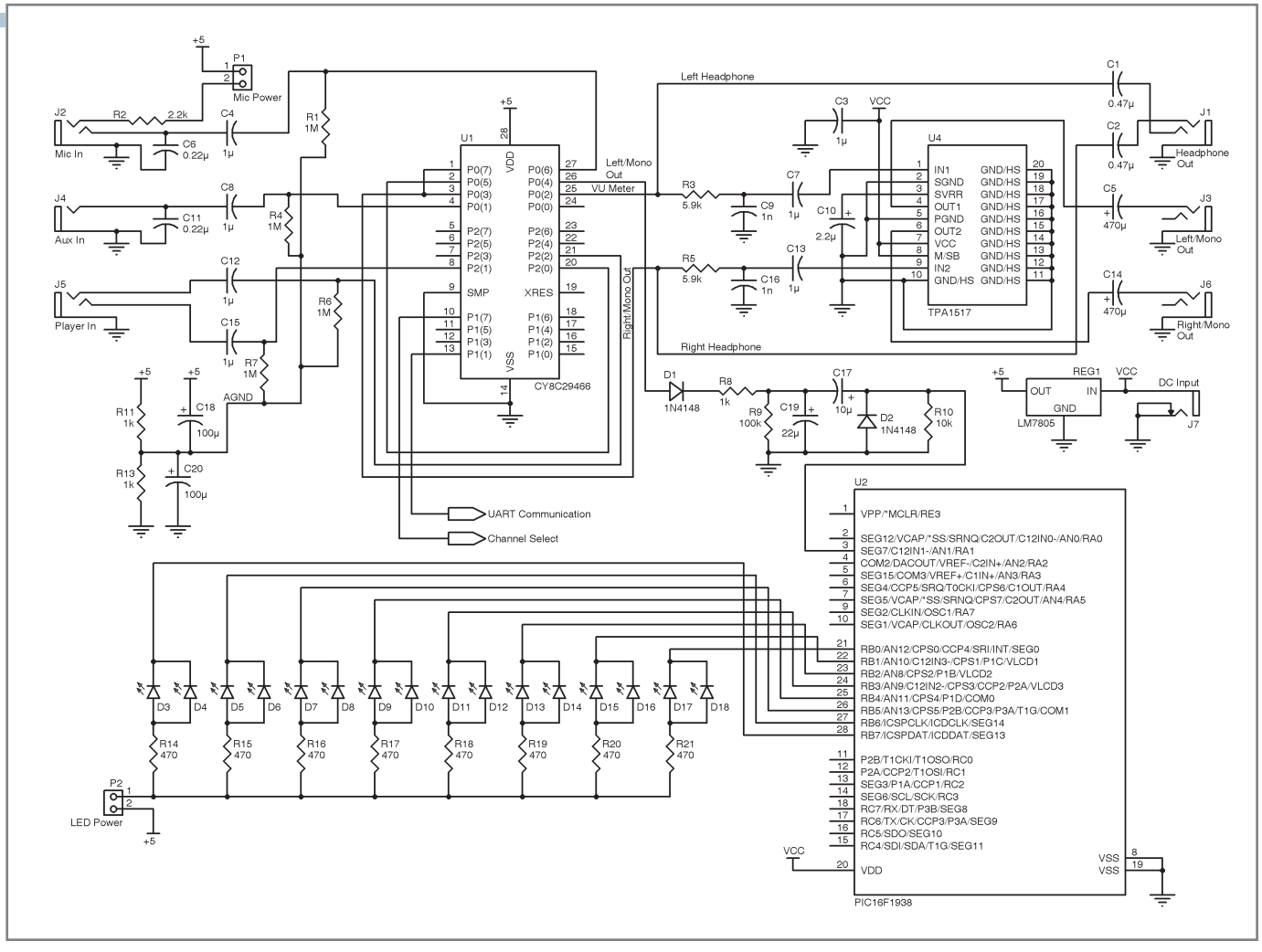


Figure 4—PodAmp schematic continued from p. 19

This meant if we used a pad with a diameter of 10 mm, a space of 0.5 mm between the pad and the ground plane should be sufficient. Because the larger the pad, the more capacitance (and therefore better touch accuracy), we made our buttons with a touchpad size of around 10 mm (give or take a couple millimeters). We figured this would be sufficient because it gave enough room for a finger to touch the pad without interfering without anything else.

Creating the slider was something we weren't too sure about. We followed the 1/20 ratio for spacing between each pad which gave us confidence that the sensor would be robust. After looking at a few different types of sliders, we decided to use a four-channel slider and started from scratch. As for length, we used a 3″ slider. This was a sufficient size for a finger to comfortably slide up and down and gave a nice feeling as far as gain-control resolution goes.

The good thing about making capacitive sliders is that any problems due to the actual slider can usually be fixed within the code. If your slider/button isn't sensitive enough (depending on how you made it) you can increase the sensitivity within the code. You can also change what each button does when it is pressed. This gives the user many different options that are eliminated when using a fixed analog button.

PUMP UP THE VOLUME

Implementing the Class AB amplifier chip from TI was pretty straightforward. As long as you follow the datasheet, you should be up and running within a couple of minutes. The nifty thing about this chip is that it comes with a muting pin. We didn't use it for our design, but by simply adding another capacitive button we could have easily implemented it as an extra feature for our amp.

When it comes to quality of sound, this chip was amazing. They always say "you get what you pay for," but in this case, for the couple of bucks, we got a lot more than we bargained for. This chip can output some pretty loud and clear audio when using 4-Ω speakers. Figure 4 is the final schematic.

WHEN IT ALL COMES TOGETHER

Once the PCB has been made, using a simple 5-V regulator and a 12-V/1-A (or 9-V) DC power supply, everything on the board will have sufficient power. We mounted the PCB on an angled piece of Plexiglas (like a < shape) as a stand and then

put a laminated piece of paper in between the Plexiglas and the PCB to indicate the different I/O which can be customized to each individual (see Photo 2).

Programming headers were never implemented onto this project. This was because, for our purposes, the kit comes preprogrammed. If a student wants to change or explore the firmware, they can easily pop out the microcontroller and construct their own programming headers via a breadboard. The benefit of having through-hole microcontrollers is that by using 28- and 40-pin sockets on the PCB the chips are easily accessible to students in the event of a burnt chip or otherwise.

How many personal amplifiers are out there that have capacitive sensing, digital-signal mixing, and good quality sound for only around $40? The answer is simply none. Having the ability to take new technology and implement it into your own design for a fraction of the price is an amazing skill! This project will hopefully teach students as much as it has taught us. Knowing your way around the world of electronics definitely has some great benefits, and we want to inspire young minds to explore it!

Coleton Denninger (coletondenninger@gmail.com) and Jeremy Lichtenfeld (jeremy.lichtenfeld@gmail.com) both studied at Camosun College in Victoria, BC. They earned Electronics Engineering Technology diplomas and are planning to continue their studies at the University of Victoria.

RESOURCES

Cypress Semiconductor Corp., "AN16833: PSoC 1 – Signal Mixing with PSoC Switched Capacitor Blocks," www.cypress.com/?rID=2925.

Microchip Technology, Inc., "AN1250: Microchip CTMU for Capacitive Touch Applications," 2009, ww1.microchip.com/downloads/en/AppNotes/01250a.pdf.

Texas Instruments, Inc., "TPA1517: 6-W Stereo Audio Power Amplifier," 2007, http://focus.ti.com/lit/ds/symlink/tpa1517.pdf.

SOURCES

CY8C29466-24PXI PSoC and PSoC Designer integrated development environment
Cypress Semiconductor Corp. | www.cypress.com

PIC16F1939 and PIC16F1938 Microcontrollers and mTouch Capacitive-touch evaluation kit
Microchip Technology, Inc. | www.microchip.com

TPA1517NE Class-AB audio amplifier
Texas Instruments, Inc. | www.ti.com





Electronics Engineering for the People



An Interview with David L. Jones

David L. Jones is a risk taker. In addition to jumping off cliffs in the name of product testing, the long-time engineer recently switched to full-time blogging. In February 2012, Dave and I discussed his passion for electronics, his product review process, and what it means to be a full-time video blogger.—Nan Price, Associate Editor

NAN: Tell us a little about where you live and work.

DAVE: I live in the Hills district of Sydney, Australia, and I was born and raised in the Sydney suburbs. I have just purchased a 50-square-meter office space to use as the new EEVblog lab, moving out of my garage lab which was a tiny 4 square meters. The funny thing is, the surface area of my new benches alone is bigger than my entire previous lab, what a luxury.

I used to shoot most of my videos in the non-air-conditioned corner of my garage, so in summertime you could actually see the sweat running down my face in the videos!

NAN: When did you first start working with electronics?

DAVE: The video story can be found at "EEVblog #54 – Electronics – When I was a boy..." www.youtube.com/watch?v=XpayYlJdbJk.

I was very young, maybe six or so, when I was taking apart stuff to see how it worked, so my parents got me a 50-in-1 Tandy (RadioShack) electronics kit and that was it, I was hooked, electronics became my life. And indeed, this seems to be fairly typical of how many engineers of the era got started.

By the time I was eight, I already had my own lab and was working on my own projects. All my pocket money went into tools, parts, and magazines.

The electronics magazine industry was everything back then before the Internet and communications revolution. I would eagerly await every issue of the Australian electronics magazines like *Electronics Australia*, *Electronics Today International (ETI)*, *Applied and Australian Electronics Monthly (AEM)*, *Talking Electronics*, and later *Silicon Chip*.

NAN: Tell us about some of your early projects.

DAVE: Given that it was over 30 years ago, it's hard to recall I'm afraid. Unfortunately, I just didn't think to use a (film) camera back then to record stuff, it just wasn't something that you did as a kid. The family camera only came out on special occasions. So those projects have been lost in the annals of time.

My first big published magazine project was a digital storage oscilloscope (DSO) adapter for PCs, in a 1993 issue of *Electronics Australia*. I originally designed this in the late 1980s. (See "electronics.alternatezone.com, http://alternatezone.com/electronics/dsoa.htm.)

NAN: You have many interests and talents. What made you choose engineering as your full-time gig?

DAVE: There was no choice, electronics has been my main hobby since I can remember, so electronics engineering was all I ever wanted to do to. I've branched out into a few other hobbies over the years, but electronics has always remained what I've wanted to do.



One of Dave's old projects from the 1980s. It's a classic rat's nest Veroboard construction using items from his junk bin.

NAN: The Electronics Engineering Video Blog—EEVBlog—is touted as "an off-the-cuff video blog for electronics engineers, hobbyists, hackers, and makers." Tell us about EEVBlog and what inspired you to begin it.

DAVE: I've always been into sharing my electronics, either through magazines, via my website, or on newsgroups, so I guess it's natural that I'd end up doing something like this.

In early 2009 I saw that (Wordpress-type) blogs were really taking off for all



Dave does ad hoc reviews of many types of multimeters

sorts of topics and some people were even doing "video blogs" on YouTube.

I wondered if there were any blogs for electronics, and after a search I found a lot of text-based blogs, but it seemed like no one was doing a video blog about electronics, like a weekly show that people could watch. Whilst some text blogs were quite good, they all had a very similar look and feel, and I didn't see much value in having "just another text blog." And, just like the magazines of old (and current), you don't really get to know the person behind the blog. You just have their (usually carefully thought through and edited) words and maybe a small photo on the About page. But what are they like in real life? Are they dull and boring? Enthusiastic? Passionate? Quirky? Funny?

I had been watching some (unrelated to engineering) video blogs, and they really seemed much more exciting than text blogs, and you really did get to love (or hate!) the personality. So I thought it'd be fun to do an electronics video blog and blaze a new trail and see what happened.

Being fairly impulsive, I didn't think about it much; I just dusted off a horrible old 320 × 240 webcam, sat down in front of my computer, and recorded 10 minutes (the YouTube limit back then) of whatever came into my head. I figured a product review, a book review, a chip review, and some industry news was a good mix. But the result was horrible, I hated it!

But not being someone who cared about embarrassing myself though, I knew enough to know that you have to start somewhere, even if you don't like it, and it's not what you intended. I wrote a post to the aus.electronics usenet group with a link to the video, and hit send before I could think it through twice.

To my surprise, about 50 people watched it and said they were waiting for the next episode. And they only gave positive feedback about how to improve it. So for the next episode, I moved the camera into my lab to make it look a bit more authentic, and the blog has continued onward and upward ever since.

I've had constant linear growth since then, and now have a regular weekly audience of over 10,000 viewers and over 4 million views on YouTube. Not to mention that it's now my full-time job.

The "off the cuff" part came about due to my natural tendency to just impulsively rant. I couldn't act, let alone write or read scripts on camera, so I just hit record and let it happen. And that's the way the blog is still done to this day—hit record and talk.

I'm still amazed at how my life has changed over the last few years, all from a simple impulsive YouTube video that I thought wouldn't get any views at all. I really did expect to go cold on the idea pretty quickly, but from all the positive comments received, I found it fun to bring out a new episode each week knowing that people were waiting for it.

NAN: Your blog's tagline is: "No script. No fear. All opinion." That pretty much sums it up. Tell us about your review process. Do companies send you products? Do you purchase things you might need for your own projects and then test and review them?

DAVE: Both. I used to review gear that I either owned or could get my hands on through work or wherever, but now companies are realizing the value of my blog and sending me gear for review.

So there was no pressure early on to "tone it down" and not "tell it like it is" from the start, which is my natural behavior—often to my detriment, that means I'm not really management material. So I built up a bit of a reputation for telling it like it is.

Some companies are still too scared to participate though, because of my unorthodox "tell it like it is" reviews and teardowns. Marketing people in these companies still aren't used to the new social media approach, as they are used to magazines and websites that simply regurgitate their marketing copy, so they don't know how to handle "bad press" when someone doesn't like their product or something goes wrong.

But the smart companies have figured out that even if their product publicly fails (I've had review gear actually blow up!), then they have found they can get even more respect by publicly responding to the incident and fixing the problem. After all, engineers understand that things fail, and small things get overlooked, and it's how a company responds that matters.

So what would previously be considered a marketing disaster—having 10,000 people watch a video of their new product blowing up—they are learning to accept the risk and stand by their product.

My review process is pretty ad hoc. I use the product on and off for a bit if possible, and then sit in front of the camera and just rattle off comments on each feature and design aspect I can find.

It's actually pretty easy for me to get a "feel" for the product almost immediately, so most of my comments are of the "gut feel" type.

Naturally, marketing departments don't like that approach, and often won't agree with anything negative I say. But what's a good review without some personal opinion?

NAN: What were, say, the top three products you reviewed?

DAVE: The Fluke 28-II multimeter is pretty memorable. Fluke gave me one and said I could do what I liked with it. Given that it was advertised as waterproof and drop proof to 3 m, I set about to test that in a fun way. So I took it through a canyon trip here in the Blue Mountains, throwing it down waterfalls, and jumping off cliffs into pools with the multimeter hanging from my harness. It survived that, so then I proceeded to abuse it some more by throwing it out of a moving car and dropping it from various heights until I finally got it to fail, well above its rated spec. An amazingly rugged meter.

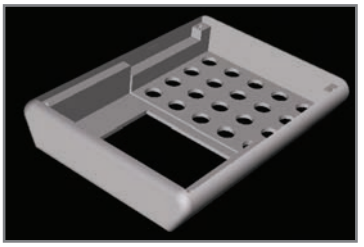
It's always fun when review products fail, and there seems to be somewhat of an EEVblog curse here, where many products I get seem to be the outliers on the manufacturing bell curve and have an issue in some way. It was hard to beat the Extech Instruments RC200 SMD Tweezer multimeter for this, with it actually blowing up in my hand as I measured the mains wall socket. It turned out there was a design oversight in the unit which, as luck would have it, I was the first to discover. But to Extech's credit, they investigated, admitted the problem, and redesigned the product in record time.

It's not every day you get to drop a $30,000 oscilloscope, but that's exactly what I did with Tektronix's new MDO4000. They originally were very hesitant to let me do a teardown of the product, as they were scared I would break their precious $30,000 scope, but at the 11th hour they finally relented. So I thought it'd be fun to give them a scare. They didn't say I couldn't drop it, so that's what I did—and I had them very worried for a while about what condition they'd get it back in! (Note: No oscilloscope was actually harmed in the filming of that video, thanks to some prestidigitation.)

NAN: You mentioned in EEVblog #231 that 2011 was big year for you. You took many "gambles," including becoming a father, switching to full-time blogging, and purchasing an office lab. As the first quarter of 2012 comes to a close, how are these changes affecting you?

DAVE: The big change is positive, in that now I have some of my life back. When I was working full time and doing the blog on the side, it was consuming most of my spare time, so to now get my weekends back has been a big improvement in family relations. Something I'm sure any midnight engineer can relate to. Little Sagan has changed things a lot, for the better, coming along just as I went full time, my best project by far! (See "EEVblog #175 - Project Sagan," www.youtube.com/watch?v=dzrtEYmuQJQ.)

NAN: Your videos are viewed as many as 6,000 times a day. What types of responses do you get from viewers?



µWatch concept design

DAVE: Most are positive, but unfortunately anonymity on the Internet does also bring out the worst in people, so you have to learn to ignore the negative comments, which can be quite harsh.

One thing I've found rather disheartening is that the high volume of comments and e-mail means I can no longer keep up with replying to everyone's questions. And I have always enjoyed helping people out with any questions they might have—right back to the early pre-Internet magazine days when I would handwrite a reply, stick it in an envelope, and lick a stamp.

And, thanks to continuous growth, that's now upwards of 8,000 to 10,000 views per day, thanks to YouTube being the world's second biggest search engine.

NAN: You also mentioned that your plan for 2012 is to "settle into your lab." Tell us a little about your lab. What types of projects are you currently working on?

DAVE: I'm working on a novel power-supply kit, in which I'm now up to Part 9 in a video design series for. I'm working on (hopefully) the world's first canyon navigating quadcopter with a friend of mine. And building a Makerbot 3-D printer as well.

Plus, I've always got many abandoned projects that never saw the light of day that I occasionally resurrect and work on again, like my µWatch, the world's first credit-card size scientific calculator watch using all off-the-shelf parts and techniques, my µCalc, which is a spin-off from the µWatch, and other small projects.

Like many design engineers I'm sure, I have a rather short attention span, so I've got a lot more half-finished projects than finished! (See "EEWeb Pulse," EEWeb.com, Issue 18, 2011, http://s.eeweb.com/pulse/eeweb-pulse-2011-18.pdf.)

NAN: An interesting recent project featured on your blog is the µCurrent. Tell us more about it and how you typically come to feature projects.

DAVE: This is actually an old design I did before I started the blog, and had published in the Australian *Silicon Chip* magazine. Like many projects, it grew out of a personal need to accurately measure current in a project, where it just so happened that the multimeter's burden voltage was causing a big problem.

Sure, I could have overcome that the traditional way by winding up the wick on the power supply and using several meters to compensate for the burden voltage, but that's not very elegant.

I looked around and couldn't find a product that would

NEW chipKIT™ Development Platforms from Microchip

The first 32-bit MCU platforms compatible with Arduino™ HW and SW



The chipKIT™ Uno32 and Max32 development boards are the first 32-bit microcontroller-based platforms that are compatible with existing Arduino™ hardware and software.

The chipKIT platform allows hobbyists and academics from many disciplines, such as mechanical engineering, computer science and artists to develop original embedded applications easily and quickly including: motor control, LCD display, wired and wireless communications, LED matrix control and sensor networks

Key Features:

- Application development using an environment based on the original Arduino IDE, modified to support PIC32 devices while still supporting the original Arduino line. Leverages existing code examples, tutorials and resources.
- Pin-out compatibility with many existing Arduino shields
- Higher performance at a lower price-point than existing solutions
- Advanced capabilities including:
  - Integrated USB (Device/Host, OTG)
  - Integrated Ethernet
  - CAN

Development Board Comparison

Feature	Core	Performance	Program Memory (KB)	RAM (KB)	Additional Features
chipKIT™ Uno32	32-bit	80 MHz	128	16	PMP/PSP/RTCC
chipKIT Max32	32-bit	80 MHz	512	128	USB, 2x CAN, Ethernet, DMA, RTCC

IT'S EASY TO GET STARTED!

- Visit **www.microchip.com/chipkit**
- Purchase a chipKIT development board
- Download the free software
- Order free PIC32 samples



chipKIT Uno32 (TDGL002)



chipKIT Max32 (TDGL003)

The Microchip name and logo, the Microchip logo and PIC are registered trademarks and chipKIT is a trademark of Microchip Technology Incorporated in the USA and other countries. All other trademarks mentioned herein are the property of their respective trademark holders. These trademark holders are not affiliated with Microchip Technology Incorporated, and do not support, sponsor or endorse chipKIT products or solutions. © 2011 Microchip Technology Incorporated. All rights reserved. 5/11

MICROCHIP

help reduce or eliminate burden voltage on a multimeter. And a straw poll of some colleagues was met with a rather surprising "what's burden voltage?" So that's when I knew it would make an interesting project and article.

A lot of my projects over the years have been test gear. I just seem to have an affinity for test gear, and maybe that stems from the old magazine days when building your own test gear was the thing to do.

Not all of them make it to print or the web though, they are just one-offs.

NAN: In February 2012, I spoke with Chris Gammell of *Chris Gammel's Analog Life*, who co-hosts the radio show, *The Amp Hour* (www.theamphour.com), with you. According to Chris, in the beginning, when the show was solely hosted by him, it was "dreadful." What prompted you to get on board?

DAVE: The history of *The Amp Hour* is rather interesting. Chris wasn't doing *The Amp Hour* back then; it was



The µWatch programmed to show the time, date, and temperature

a different podcast, more of the "learning electronics" tutorial variety, not the general rant-style show we have now. He started that because someone on Reddit asked if there were any such audio electronics tutorial, so Chris thought he'd give it a go.

I actually had the idea for a podcast electronics radio show long before the video blog idea started, as I already had a podcasting mic I used on another project at the time (a novel fitness program for iPods, before that market became saturated), so it was natural that I wanted to use it in some way. But I couldn't come up with any good reason why anyone would listen to me talk on my own about electronics for a hour, and my voice isn't exactly made for radio(!), so I dropped the idea, and the enthusiasm eventually went into the EEVblog.

But, by coincidence, just before that Reddit post, I had resurfaced the idea for the radio show as I figured that a large part of my audience might like an audio podcast version of the show they could listen to while at work, walking, or in the car, etc. Something that allowed them to get their electronics "fix" where it wasn't convenient to watch video.

I knew the only way such a show would work was with a co-host, and that's when I saw the Reddit post and I listened to Chris's first tutorial podcast. I thought it sounded OK, just like many people liked my first video when I actually hated it, so I guess we are our own worst critics!

I liked that Chris was a Yank, and he seemed a more reserved personality, so

Get PUBLISHED. Get NOTICED. Get PAID.

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

Do you have what it takes?

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

editor@circuitcellar.com

CIRCUIT CELLAR



I thought an Aussie/Yank combo with a clash of personalities might work well on show. But more importantly, Chris seemed enthusiastic, which is the main requirement for a project such as this. So I drafted an e-mail to him about my idea for a show, but I didn't send it because I wasn't sure if I had the time to commit to both projects.

A few weeks later, again by coincidence, Chris contacted me for an interview he wanted to do with me. So I told him about the co-hosted show idea, and why not answer the questions on the show as a start? We both liked the idea, and like my video blog, didn't really think about it too much. We just picked up some mics, made a Skype call, and hit record. And that recording became our first show. At the time it didn't have a name, so we just marketed it on our respective blogs as the *Dave Jones & Chris Gammell Show*. The show was popular and we got quite a bit of feedback, and within time the show would get a name, its own website, and now a loyal following as the world's most popular electronics radio show.



Dave's in-progress µCalc project

NAN: Who has been your favorite or most memorable guest?

DAVE: The best shows are always when we have guests on who have many interesting stories to tell. Dr. Howard Johnson of *High-Speed Digital Design, A Handbook of Black Magic* fame was very memorable. He spoke well and regaled us with many industry stories.

NAN: What was the last electronics-design related product you purchased and what type of project did you use it for?

DAVE: I decided to include Arduino compatibility in my new power supply project (an Atmel ATmega328) so it would possibly appeal to a broader range of beginners. But I initially thought that turned out to be a big mistake when Murphy ensured that I struck every bug in the Atmel AVR and Arduino tools!

NAN: Full-time blogging is a rather new endeavor for you. Where do you see yourself in the next five to 10 years?

DAVE: I honestly don't know what the future will hold, and that's the great thing about it. If you told me two years ago that I would be a full-time engineering video blogger, I would have just chuckled. Five to 10 years is too far in the future. I have a hard time trying to predict next month let alone that far ahead, so I don't try. I just do what I enjoy, do it my way, and see where it goes. Because if you aren't doing what you enjoy, then I don't see the point.

System on Module
New - SoM-3517
• TI ARM Cortex-A8 600 MHZ Fanless Processor
• Up to 512 MB of DDR2 SDRAM
• UP TO 1gb of NAND Flash
• 2 Full Speed USB 1.1/2.0 Host ports
• 1 Full Speed USB OTG port
• 4 Serial Ports, 2 I2C and 2 SPI ports
• Processor Bus Expansion
• 10/100 BaseT Fast Ethernet
• CAN 2.0 B Controller
• Neon Vector Floating Point Unit
• 16-bit DSTN/TFT LCD Interface
• 2D/3D Accelerated Video w/ Resistive Touch
• Small, 200 pin SODIMM form factor (2.66 x 2.375")
Windows CE
2.6 KERNEL
The SoM 3517 uses the same small SODIMM form-factor utilized by other EMAC SoM modules, and is the ideal processor engine for your next design. All of the ARM processor core is included on this tiny board including: Flash, Memory, Serial Ports, Ethernet, SPI, I2C, I2S Audio, CAN 2.0B, PWMs, Timer/Counters, A/D, digital I/O lines, Video, Clock/Calendar, and more. The SoM-3517M additionally provides a math coprocessor, and 2D/3D accelerated video with image scaling/rotation. Like other modules in EMAC's SoM product line, the SoM-3517 is designed to plug into a custom or off-the-shelf Carrier board containing all the connectors and any additional I/O components that may be required. The SoM approach provides the flexibility of a fully customized product at a greatly reduced cost. Contact EMAC for pricing & further information.
http://www.emacinc.com/som/som3517.htm
Since 1985 OVER 25 YEARS OF SINGLE BOARD SOLUTIONS
EMAC, inc.
EQUIPMENT MONITOR AND CONTROL
Phone: (618) 529-4525 • Fax: (618) 457-0110 • Web: www.emacinc.com

Gigabit Technology
• ARM-based
• System on a Chip
• Gigabit Ethernet
• Small, Cheap, Fast
• Both QFP and BGA Packages
• Standard Development Tools
• 10 Year Life Guarantee
• Royalty Free RTOS, TCP/IP V4/6
$10.00 each (Qty 100)
The gridARM™ System on a Chip (SOC) is a high performance, low cost, low power, highly integrated single chip with 10 / 100 / 1000 Mbps Ethernet, USB, CAN, Serial, SRAM Memory, SPI, I2C, RTC and internal peripherals designed to provide a complete solution for embedded applications.
THE NETWORKING EXPERTS
gridconnect
800.975.4743 USA • 1 630.245.1445
gridconnect.com/gridarm.html
Leaders in the embedded and networking marketplace providing network hardware, high quality software and services

by Charles Edmondson (USA)

The Rainbow Color Reader (Part 2)

Color Distinction & Classification

The first article in this two-part series introduced the Rainbow Color Reader project, which is a color identifier built on a single processor that stores sound samples for the visually impaired. This article focuses on hue, saturation, and luminance. You also learn how to reliably distinguish colors.

In the first part of this article series, I described my Rainbow Color Reader design. I also detailed how I added recorded audio to the system. Now I'll explain how to distinguish colors reliably.

COLOR THEORY PRIMER

In order to determine the color detected by a sensor, you must understand color theory. While there are many theories about color and several schemes for classifying colors, I went with hue, saturation, and luminance (HSL) for this project.

Hue has to do with the basic color of an object (see Figure 1). For the HSL color scheme, think of hue as a circle with red at 0°, green at 120°, and blue at 240°. If you take those values and make them have a total value of 1, you have red = 0 or 1.0, green = 0.33, and blue = 0.66. Other colors are visible: yellow at 0.33, cyan at 0.5, magenta at 0.77, and so forth. However, hue relates only to a sample's predominate color. You also need to determine the luminance and saturation of the sample.

Luminance, or lightness, is basically a color's brightness. A sample with zero luminance is black. A sample with a luminance of 1.0 is white. Samples with a luminance of 0.5 are in our primary color circle.

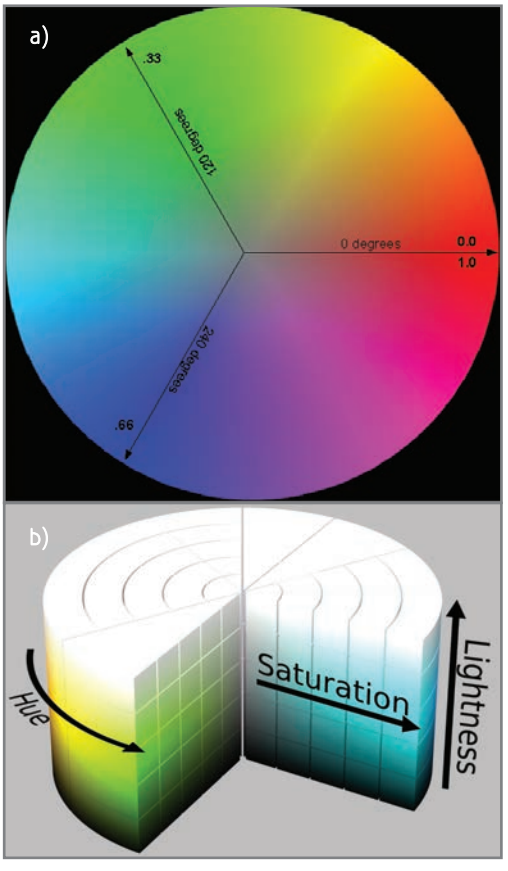


Figure 1—Hue circle (a) and HSL cylinder (b)

Saturation is basically a color's purity. Samples with low saturations appear gray or brown. Samples with high saturations appear to be pure, bright colors.

As you can see, all three of these parameters interact when trying to determine a color. Samples with high L levels as well as high saturation appear to be light colors, but as saturation diminishes, they begin to fade into grays and beiges. Low saturation samples in the red to orange hues appear brown or beige, depending on the L level, while blues and greens appear grayer. These interactions make the classification of a color a very interesting proposition.

CALCULATING AN HSL VALUE

The first step in working with any sensor is calibrating it. For the formulas that I was using, I needed three measurements to determine the HSL value: the values of red, blue, and green scaled

to a number between 0 and 1. I consider these two values as white and black. However, my sensor is an ambient light sensor that feeds through a controllable gain amplifier into a 10-bit analog-to-digital converter (ADC). How can I get the most accurate measurement of each color, while retaining the best accuracy and precision possible?

In order to obtain that range of 0 to 1, I need to determine which values I read for these two points. To determine the white value for each color, I use a known white sample of a piece of clean white felt. Resting the unit vertically on the folded felt, I take a measurement of the reflection with one of the LEDs lit. If the value recorded is above a threshold, (e.g., 1,018), I decrease the value of the gain (the setting for the digital potentiometer) and measure again. If the value is below a different threshold, such as 1,012, I increase the gain and try again. I continue to measure and adjust the gain until it either meets the standard, or I reach either extreme of the potentiometer's settings. I then record as simulated EEPROM the gain and calibration value for that LED.

After I find the white calibration values for each LED, I need the 0 level for each color. I move the unit to my "black hole box," which is a small cardboard box lined with black felt with a hole in one side just big enough for the front of the unit. When the sensor is in Calibration mode, it detects that there is little reflected light coming back into the sensor. So, it first sets the gain for each color and reads the first ADC channel. It stores this value as the low-gain black level. It then reads each color on the high-gain channel and stores the value as the high-gain black level.

What do I use these calibration values for? Theoretically, what I need for calculations is a `float` value between 0 and 1. What I have when I measure an unknown sample is an `int` between 0 and 2,048. From calibration tests, I know where black should be, so that will be the 0 point. And I know where white should be, so that will be the 1 point. So, the value for each color becomes:

$$\frac{\text{MEASURED} - \text{BLACK}}{\text{WHITE} - \text{BLACK}}$$

Once I have these values for red, green, and blue (R, G, and B), I can determine the HSL values. First, I determine the MAX and MIN of these values. I can then quickly find the L value as simply the average of MAX and MIN. Second, I could determine the saturation (SAT.) It is simply:

If L is less than 0.5:

$$\frac{\text{MAX} - \text{MIN}}{2 - (2 \times \text{L})}$$

If L is more than 0.5:

$$\frac{\text{MAX} - \text{MIN}}{(2 \times \text{L})}$$

Determining the hue is a little more complicated. The code is shown in Listing 1. Basically, you determine which of the three primary colors is strongest. You then determine the measured color's adjustment from the primary color.

Listing 1—The code used to determine the hue

```
// Determine hue
if (R == max)
          {
          HUE = ((G-B)/(max-min))/6;
          }
else if (G == max)
          {
          HUE = (2 + (B-R)/(max-min))/6;
          }
else HUE = (4 + (R-G)/(max-min))/6;

if (HUE < 0) HUE = HUE + 1;
```

REAL-WORLD CONSIDERATIONS

When you actually try to classify colors with this type of sensor, you find out that it is not as simple as it looks. First, there are inaccuracies in the sensor itself. The optical path is not perfect. Some of the LEDs' light reflects off the front window and onto the sensor. Other light leaks through the printed circuit board (PCB) itself, reflects off the edges of the PCB, and otherwise finds its way to the sensor. You need to limit ambient light from the environment from reaching the sensor. To help with these problems, a light shield made from black construction paper with only a small 0.5" square hole is attached to the enclosure and limits light from outside from reaching the sensor. The PCB was painted black with a Sharpie to reduce light transmitted through the board. These measures are far from perfect.

There are variations in the LEDs from unit to unit, and as they age, output may both increase and decrease. I had to replace more than 10% of my original red LEDs because their performance would not calibrate properly. There are only 256 steps of gain in my amplifier, which means reaching my targeted calibration levels could be problematic.

Finally, I experienced a problem with dark colors. As the reflected signal fell to within 10% of the black level, odd trends began to occur. Response from unit to unit drastically changed. In some units, red would be stronger. In others, green would be stronger. It became very difficult to tell the difference between navy blue, black, dark brown, or dark gray. This problem took nearly a year to solve, and even then the solution wasn't satisfactory. I calibrated each unit, and then performed a test run on a set of dark colors—navy, dark brown, dark-dark gray, dark-dark purple, and black—and recorded the HSL and RGB values of each sample. From this, I could tell from the errors whether red, green, or blue was strong or weak, and I would reprogram and recalibrate adding a set of "fudge factors" to correct for the variations between units. With this added step, I brought the dark color

accuracy up to 90% to 95%, making this a viable commercial product.

COLOR CLASSIFICATION

Determining a sample's color is a multiple-step process. First, I take a reading with no LEDs on at all to get a base illumination reading. If this is high enough, I just say "light" and I'm done. Next, I read each color's reflected signal and use the aforementioned process to determine the HSL values. If the luminosity level is less than 10%, I then reread the colors on the high-gain channel and process the sample through the dark colors process. Otherwise, I begin the color classification process.

To determine if the sample was white, I see if the luminosity is above a specified threshold. Hue and saturation are not yet involved.

Next, I determine if the sample is gray, beige, or brown. These colors are mainly determined by saturation, but hue and luminosity come into play for telling the difference between gray and brown, since browns tend to be in the red/orange/yellow hue range. Luminosity indicates whether the sample is dark, normal, or light, and whether it is brown or beige.

From here, I go around the color wheel from red to orange to yellow and so on, and then back to red. I rely mainly on the hue, but also test the luminosity for colors such as pink and gold as well as light and dark shades of each color. As soon as I determine the color, I speak it and shut down.

Determining threshold locations—from white to black to all the colors in between—was a difficult process. Initial attempts to do a purely theoretical determination failed miserably. My sensor is not perfect in either its color purity or intensity measurements. Instead, I relied on the tried-and-true method of trial and error. I purchased an assortment of felt color squares and tested and measured the results on multiple units, which I recorded in Excel spreadsheets. I also pulled a lot of clothes from the closet to test and recorded the results in Excel. This was made easier by implementing a special test mode in the software that not only spoke the determined color, but also spoke out the RED, GREEN, and BLUE raw data and the HUE, SAT, and LUM of the tested object. This enabled me to see how one unit differed from another in its performance, helped me characterize the various components in the sensor, and enabled me to determine appropriate thresholds between colors.

Item	Modifier	Color	Lum	Sat	Hue	Red	Green	Blue	Reads
Black felt		black	0	0	0	584	592	596	black
Black cadence shirt		black	0.004	0.989	0	629	582	595	black
Dark brown leather skirt	dark	brown	0.029	0.369	0.882	712	651	722	dark-dark purple
Dark purple velvet skirt	dark	purple	0.022	0.476	0.706	659	635	713	navy blue
Black polo shirt (CH)		black	0.017	0.574	0.94	678	624	660	black
Navy pants	navy	blue	0.01	0.994	0.77	643	592	681	black
Black cotton pants (CH)		black	0.006	0.992	0.982	642	591	606	black
Navy skirt	navy	blue	0.014	0.88	0.875	681	607	677	black
Dark purple turtle	dark	purple	0.02	0.879	0.898	715	610	692	dark-dark purple
Smooth costume	dark	brown	0.019	0.971	0.005	715	605	604	dark brown
Graduation gown		black	0.011	0.995	0.959	673	593	622	black
Fuzzy costume	dark	brown	0.031	0.998	0	796	594	580	dark brown
Dark gray suit	dark	gray	0.037	0.228	0.869	731	673	752	dark-dark gray

Table 1—Tests recorded as I tried to develop reliable dark color classifications

Table 1 shows an example of the tests I recorded while trying to develop reliable dark color classifications. In most tests, there would be 20 to 50 color samples and as many as 10 different units. It usually took days to fully analyze a large test. I would love to be able to say I created a complex data analysis program to determine the color classification scheme, but I can't! Instead, I spent hours sorting and testing the results until the color patterns became apparent. Then, it was just a task of creating the IF-THEN-ELSE statements to implement the classifications. A similar process using the high-gain channel gave me a set of tests to do for the dark color region.

GOING TO MANUFACTURING

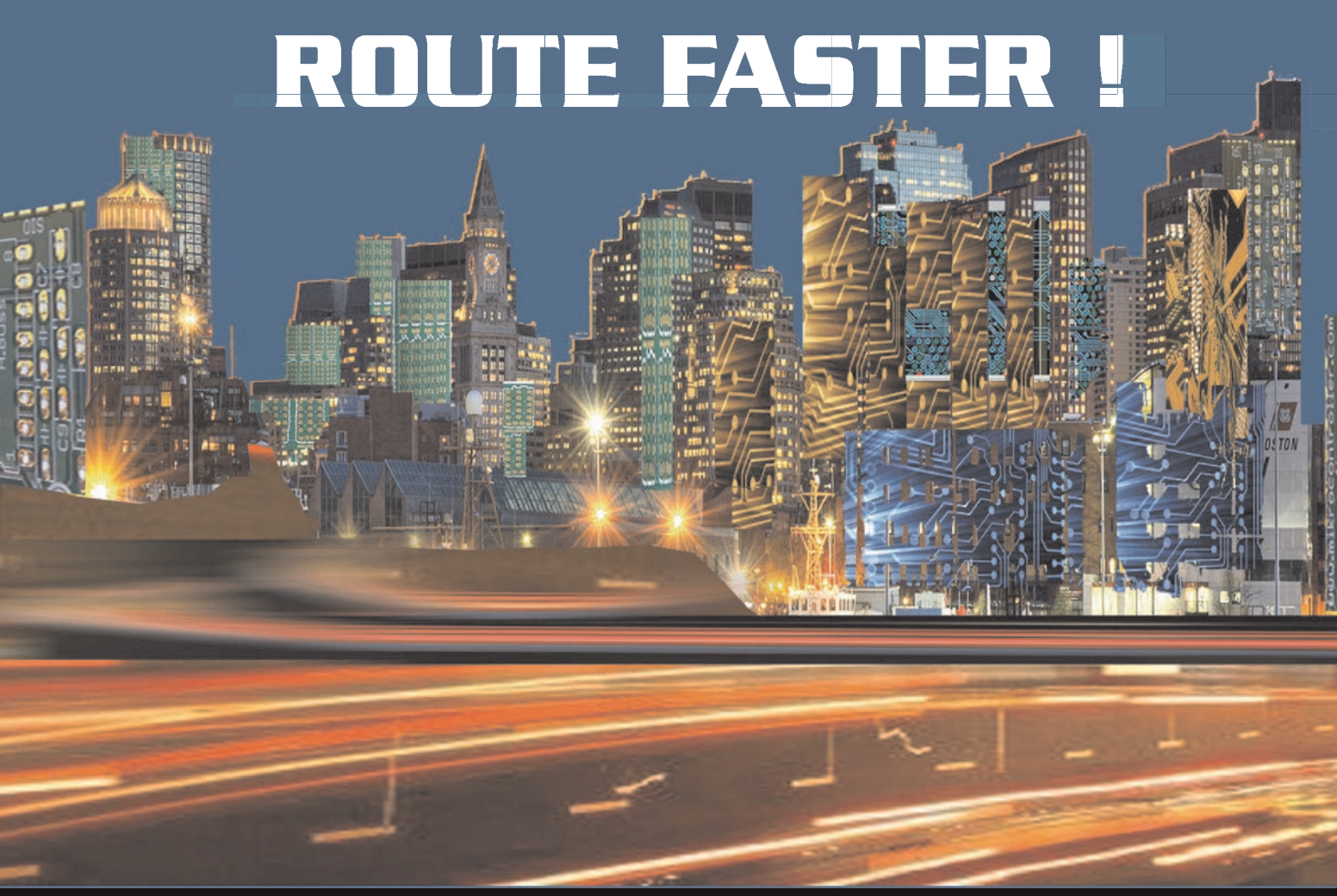
All of my prototypes had been commercially made PCB boards, which I had hand-soldered. When I went to high-density parts, I ordered solder paste stencils, which I then bulk-soldered on a hot plate. (Thanks, SparkFun!) This was adequate for my prototypes, but it definitely was not accurate or fast enough for mass production, even for the 100-unit runs I was shooting for. I only needed the boards made and the SMT parts assembled on the boards. I would do the three through-hole parts and program and assemble them into the cases. A commercial assembler was going to be needed.

I attended several electronic manufacturing shows, including the MCM conference in Anaheim, CA, and the Del Mar Electronics Show in San Diego, CA. I met with several assembly firms, showed them the units I needed produced, and found several companies that expressed interest in helping me out. After sending several vendors our Gerbers and BOMs, I received a great quote from Aimtron and ordered my first set of boards. I was on my way!

After receiving the new boards, I immediately finished up the first 12, programmed them, and tested them. Nothing! The LEDs flashed, but there was no sound at all. After several days of troubleshooting, I finally determined that two of the capacitors, both of the same value, were bad. Contacting Aimtron, they tested and found out that they had received an entire reel of mismarked capacitors. They offered to replace them for me, but I foolishly decided to do it myself. They sent me the correct caps, and I then had another step in my assembly process of changing out those two capacitors.

With the caps replaced, and these first 12 boards installed in their cases, it was time for testing and calibration. It was quickly apparent that my prior "release" software had

ROUTE FASTER !



WITH PROTEUS PCB DESIGN

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

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

PROTEUS DESIGN SUITE

- Hardware Accelerated Performance.
- Unique Thru-View™ Board Transparency.
- Over 35k Schematic & PCB library parts.
- Integrated Shape Based Auto-router.
- Flexible Design Rule Management.
- Polygonal and Split Power Plane Support.

Features:

- Board Autoplacement & Gateswap Optimiser.
- Direct CADCAM, ODB++, IDF & PDF Output.
- Integrated 3D Viewer with 3DS and DXF export.
- Mixed Mode SPICE Simulation Engine.
- Co-Simulation of PIC, AVR, 8051 and ARM7.
- Direct Technical Support at no additional cost.

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

Labcenter Electronics www.labcenter.com

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

SEALEVEL®
Computing/HMI | Serial | I/O

RISC computing power. A wealth of I/O features.

The R9 family is an application-ready platform for your next product design. R9 products deliver RISC computing power using the latest Windows® Embedded CE 6.0 software environment and unmatched I/O features that extend capabilities beyond traditional RISC applications. Choose from single board computers, rugged embedded systems, and panel mount flat panel touchscreen computers. All R9 solutions offer small size, wide operating temperature range, and flexible I/O connectivity.

R9 Features:

- Atmel AT91SAM9263 ARM® Processor
- On-board Serial, Digital, and Analog I/O
- Compatible with Windows Embedded CE 6.0 and Linux
- Rugged Mechanical Construction
- Compact Designs as Small as 4.9" x 3.9"
- -40°C — +85°C Operating Temperature Range



Custom RISC solutions available – contact us for more information.



sealevel.com > sales@sealevel.com > 864.843.4343

Learn more about RISC computing solutions at **sealevel.com/cir/risc** or scan this QR code with your smart phone.

© 1986-2011, Sealevel Systems, Inc. All rights reserved.

some major deficiencies. Accuracy, especially for the dark colors, was very poor. It took another three months of intensive testing, analysis, and software revisions before my final software was complete. It was during this period that I discovered 10% of the red LEDs were considerably stronger than the rest and would not calibrate under any set of conditions I could tolerate. I had to develop and follow a complete test and calibration procedure. But since then I've produced 99 units for sale! Now all I need are customers.

Charles Edmondson (charles@edmondsonengineering.com) is the VP of Engineering at Edmondson Engineering, Inc. (EEI), where he specializes in embedded and mixed-signal design. Prior to starting EEI, he worked for 10 years at Cadence/Orcad/Microsim, where he was a lead applications engineer for PSpice. He also performed design and installation duties on the 91 express lanes and other toll roads, and he conducted AV system design at University of California, Santa Barbara.

PROJECT FILES

To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2012/261.

SOURCES

Explorer 16 Development board, PIC24 microcontroller, MCP6024 quad op-amp, MCP1640 DC-DC converter, and MCP4151 SPI digital potentiometer
Microchip Technology, Inc. | www.microchip.com

M-6 and M-4 Series Enclosures
Serpac Electronic Enclosures | www.serpac.com

TPA301 Audio amplifier
Texas Instruments, Inc. | www.ti.com

TEPT4400 Ambient light sensor
Vishay Intertechnology, Inc. | www.vishay.com

HUMANDATA® FPGA / CPLD Boards from JAPAN

SAVING COST=TIME with readily available FPGA boards

- Basic and simple features, single power supply operation
- Quality and reliability is provided by years of sales
- Same board size and connector layout – ACM/XCM series
- All stocked items are ready to be shipped immediately
- Over 100 varieties of FPGA/CPLD boards are available
- Customizing speed grade and/or any features are possible
- Free download technical documents before purchasing
- High quality and highly reliable FPGA/CPLD boards from Japan
- Almost all products are RoHS compliance

ALTERA FPGA Board

Arria II GX F572 FPGA board

ACM-025 series NEW

Arria II GX | DDR2

EP2AGX45DF25C6N
EP2AGX65DF25C6N
EP2AGX95DF25C6N
EP2AGX125DF25C6N
Credit card size (86 x 54 mm)

RoHS compliant





Cyclone IV E F484 FPGA board

ACM-023 series

Cyclone IV E | MRAM

EP4CE55F23C8N
EP4CE75F23C8N
EP4CE115F23C8N
Credit card size (86 x 54 mm)

RoHS compliant



CycloneIV GX F484 FPGA board

ACM-024 series

Cyclone IV GX | DDR2 | SIF40

EP4CGX50CF23C8N
EP4CGX75CF23C8N
EP4CGX110CF23C8N
EP4CGX150CF23C7N
Credit card size (86 x 54 mm)

RoHS compliant





Cyclone IV E F484 FPGA board

ACM-107 series

Cyclone IV E | MRAM

EP4CE55F23C8N
EP4CE75F23C8N
EP4CE115F23C8N
Compact size (43 x 54 mm)

RoHS compliant



XILINX FPGA Board

Spartan-6 FGG484 FPGA board

XCM-018/018Z series

Spartan-6 | MRAM | DDR2

XC6SLX45-2FGG484C
XC6SLX75-2FGG484C
XC6SLX100-2FGG484C
XC6SLX150-2FGG484C
Credit card size (86 x 54 mm)

RoHS compliant



Spartan-6 FGG484 FPGA board

XCM-110/110Z series

Spartan-6 | MRAM | DDR2

XC6SLX45-2FGG484C
XC6SLX75-2FGG484C
XC6SLX100-2FGG484C
XC6SLX150-2FGG484C
Compact size (43 x 54 mm)

RoHS compliant



Virtex-5 FFG676 FPGA board

XCM-011 series

Virtex-5 | FRAM | SDRAM

XC5VLX30-1FFG676C
XC5VLX50-1FFG676C
XC5VLX85-1FFG676C
XC5VLX110-1FFG676C
Credit card size (86 x 54 mm)

RoHS compliant



Virtex-5 LXT FFG665 FPGA board

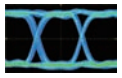
XCM-017 series

Virtex-5 | SDRAM | RocketIO | SIF40

XC5VLX30T-1FFG665C
XC5VLX50T-1FFG665C
Credit card size (86 x 54 mm)

RoHS compliant





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

www.hdl.co.jp/CC/

FPGA/CPLD Stamp Module PLCC68 Series

Easy and Quickly Mountable Module

FPGA Module IC socket mountable



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



XILINX PLCC68 Series

XP68-03 Spartan-6 PLCC68 FPGA Module



NEW | Spartan-6 | PLCC 68

XC6SLX45-2CSG324C
3.3V single power supply operation
On-board oscillator, 50MHz

RoHS compliant

XP68-02 Spartan-3AN PLCC68 FPGA Module



Spartan-3AN | PLCC 68

XC3S200AN-4FTG256C
FPGA internal configuration ROM
Two User LEDs

RoHS compliant

ALTERA PLCC68 Series

AP68-04 Cyclone III PLCC68 FPGA Module



NEW | Cyclone III | PLCC 68

EP3C25U256C8N
3.3V single power supply operation
On-board oscillator, 50MHz

RoHS compliant

AP68-03 Cyclone III PLCC68 FPGA Module



Cyclone III | PLCC 68

EP3C10U256C8N
4Mbit Configuration Device
Two User LEDs
One User Switch(Slide)

RoHS compliant

AP68-02 MAX V PLCC68 CPLD Module



MAX V | PLCC 68

5M570ZF256C5N
External Clock inputs
On-board Voltage regulator

RoHS compliant

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

by Clemens Valens (France)

The RL78 Microcontroller



An MCU Family for Low-Power Applications

When Renesas bought NEC, the former got its hands on a large range of electronic devices from the latter, including the popular 78K series of 8- and 16-bit microcontrollers. Quickly after the merger, the RL78 MCU family was announced, targeted at low-power applications.

The low-power 8/16-bit microcontroller (MCU) market is a bit of a warzone with several MCU manufacturers proposing "the industry's lowest power solution." In a YouTube video, Texas Instruments boasts a best active figure of 160 µA/MIPS for their MSP430 family. In application note AN1267, Microchip Technology claims 110 µA at "1 MHz Run" for their PIC16LF72X. And Renesas Electronics announced 70 µA at "1-MHz normal operation" on their RL78 product website.[1, 2, 3] The absence of justification on how exactly these figures were obtained makes comparing them rather useless. But then again, you don't really have to because, as most low-power developers know from experience, if you don't get the hardware and software design right, you will never attain the promised 20-year battery lifetime no matter how low the MCU's active, sleep, or standby current may be. In this article, I will take a closer look at Renesas's quickly expanding RL78 family to see what they offer that may help you create a low-power design.

THE RL78 FAMILY

The RL78 family of 16-bit MCUs currently has two branches, "generic" and "application specific," but a third "display" branch is forthcoming. The generic branch contains the subfamilies G12, G13, and G1A, all based on the 78K core, and the G14, which is based on the R8C core. In the application-specific branch there is the 1A and F12. I am not sure about their core origins as these products are still very new and, at the time of writing, documentation is missing. It doesn't really matter; from now on it is the new RL78 core for all. Since they share the same core, I will concentrate on the G13 for which I have a nice evaluation board (see Photo 1 and "The Renesas Demonstration Kit for RL78" sidebar).

RL78/G13

This family comes in a large number of variants (I counted 182), with devices having from 20 up to 128 pins (see Figure 1). Note that the parts themselves are labelled R5F10*xx*. The differences between all these variants are, besides the package type, the amounts of flash memory (program and data) and RAM. Program flash memory starts at 16 KB and currently ends at 512 KB, data flash sizes can be 0, 4, or 8 KB and RAM is 2 KB for the small devices and up to 32 KB for the big ones.

The CPU is 16-bit, but the internal memory architecture is 8 bit. Its 32 general-purpose registers are organized in four banks of eight and can be used as 8- or 16-bit registers.

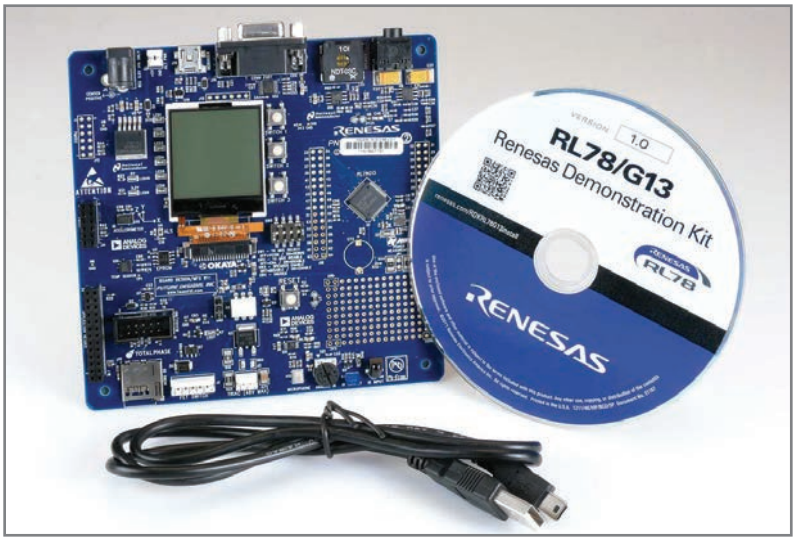


Photo 1—The Renesas Demonstration Kit for RL78

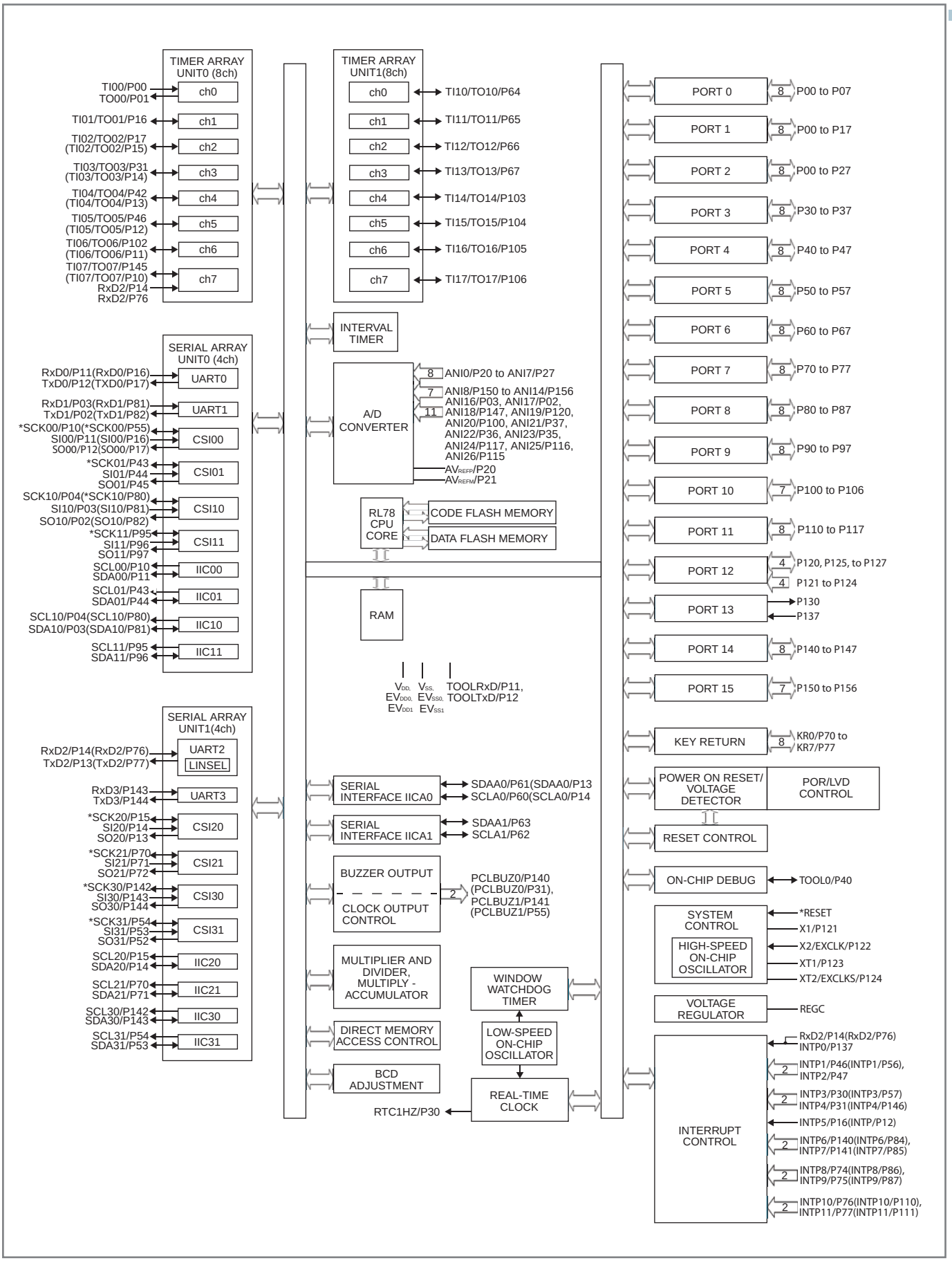


Figure 1—Block diagram of the 128-pin RL78/G13 devices. Devices with fewer pins have fewer and/or smaller blocks.

The memory-mapped special function registers (SFRs) that control the on-chip peripherals can be addressed per bit, per byte, or as 16-bit registers, depending on the register. A second set of SFRs, the extended or second SFRs, are available too, but they need longer instructions to be accessed.

For those who need to squeeze the maximum out of MCU performance, it may be interesting to know that the CPU offers a short addressing mode enabling you to access a page of 256 bytes with a minimum amount of code.

The maximum clock frequency of the processor is 32 MHz, but the hardware user's manual, which is almost 1,100 pages, interestingly also boasts about the ultra-low-speed capabilities of the processor as it can run from a 32.768-kHz clock.

The RL78 core features 15 I/O ports, most of which are 8-bit wide. Port 13 is 2-bit wide and ports 10 and 15 are 7-bit wide. The port pins that are actually available depend on the device. Inputs and outputs are highly configurable. Inputs can be analog, CMOS, or TTL. Outputs can be CMOS or N-channel open drain. Pull-up resistors are available too. The exact configuration possibilities depend on the port pin, so consult the datasheet. Because of the many configuration options, it is possible to use the MCU in multi-voltage systems without level-shifting circuitry except for the occasional external pull-up resistor. The chip can be powered from 1.6 V to 5.5 V, the core itself runs from 1.8 V provided by an internal voltage regulator.

The Renesas Demonstration Kit (RDK) for RL78

A square development board (5.1″ × 5.1″) for the RL78/G13 MCU is available loaded with many useful peripherals: a graphical LCD with SPI; a stereo audio/PWM output; a MEMS microphone; a miniature loudspeaker with its own amplifier; three push buttons; some LEDs; communication interfaces such as IR, RS-232, I²C, and SPI; a microSD card slot; a 512-KB serial EEPROM; a three-axis I²C accelerometer; I²C temperature and ambient light sensors; extension connectors; and more. Less common are the two power-switching outputs, one controlled by a MOSFET (2 A, 60 VDC, RDSON = 78 mW) and the other by a TRIAC (48 VACRMS maximum). The RDK has an on-board Renesas TK debugger/programmer supported by the IAR compiler (kick-start edition included in the kit).

The extension headers come in several flavors: there are two connectors compatible with Digilent's SPI-based PMOD standard, an application header that accepts add-on modules like the Wi-Fi modules from RedPine and Gainspan, and two expansion headers that enable direct access to MCU pins. For the owners of a Beagle I²C or SPI protocol analyzer (manufactured by Total Phase) it will be interesting to know that a compatible port is available on the board providing easy access to the I²C and SPI buses.

The board can be powered over USB or from an external 5-VDC regulated power supply, but be careful: there are no over- or under-voltage protections. A supercap can be mounted for battery-operated applications.

TIME MANAGEMENT

Several options are available for the MCU clock. When clock precision is not too important, the MCU can be run from its internal clock, up to 32 MHz, otherwise it is possible to connect an external crystal, resonator, or oscillator. An internal low-speed clock (15 kHz) is also available, but not for the CPU, only for the watchdog timer (WDT), the real-time clock (RTC), and the interval timer.

The timers of the RL78 are flexible and offer many functions. Depending on the pin size of the device, you can have up to 16 16-bit timers, grouped in two arrays of eight. Each timer (called a "channel") can function as an interval timer, square-wave generator, event counter, frequency divider, pulse-interval timer, pulse-duration timer, and delay counter. For even more possibilities, timers can be combined to create monostable multivibrators or to do pulse-width modulation (PWM). This way, up to seven PWM signals can be generated from one master timer. If you need more timers but resolution is less important, you can split some 16-bit timers in two 8-bit timers (this is not possible with all timers). Timer 7 of array 0 is extra special as it features local interconnect network (LIN) network support (see below).

Aside from date and time keeping with alarms, the RTC also provides constant period interrupts at 2 Hz and 1 Hz and also every minute, hour, day, or month. A 1-Hz output is available on devices with 40 or more pins. For extra precision, the RTC offers a correction register for fine tuning the 32,768-kHz clock. Unsurprisingly, the RTC continues operation when the MCU is stopped.

Now that I mentioned Stop mode, a special interval timer peripheral enables wakeup from this mode at periodic intervals. This timer is also used for the analog-to-digital converter's (ADC's) Snooze mode. More on that later. With a clock frequency of 32,768 Hz, the lowest interval rate is 8 Hz (0.125 ms).

Yet another time-related peripheral on the RL78 is the buzzer controller (not available on 20-pin devices). This is a clock output destined at IR comms carrier generation, to clock other chips in a system or to produce sound from a buzzer. A gate bit enables modulation of this output in such a way that pulses always have the same width.

Finally, a WDT completes the timing peripherals. It

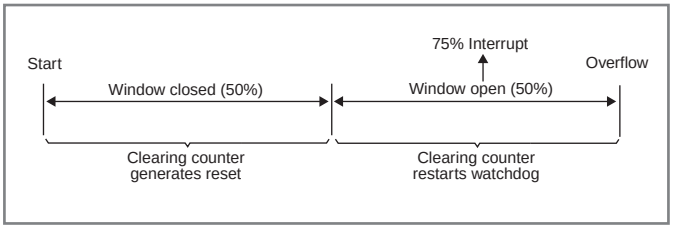


Figure 2—Trying to reset the watchdog counter when the window is closed results in an internal reset signal being generated. An interrupt can be used to signal that the window is open.

has a special Window mode that limits the time frame during which you can reset the watchdog to a fraction of the watchdog interval (50%, 75%, or 100%). Resetting the watchdog counter outside the window results in a reset. The window is open in the second part of the interval. An interrupt can be generated when the WDT reaches 75% of its time-out value, (i.e., when the watchdog reset window is known to be open in all cases). Figure 2 illustrates the mechanism.

ADC

The ADC is of the 10-bit successive approximation type and can have up to 26 inputs. Several triggering options are provided, hardware and software, where hardware triggering means triggering by a timer module (timer channel 1 end of count or capture, interval timer, or RTC). The time it takes to do a conversion depends partly on the triggering mode. When input stabilization is not too much of an issue (i.e., when you don't switch inputs) you can achieve conversion times of just over 2 µs.

Two registers enable comparing the ADC's output to maximum and minimum values, producing an interrupt when the new value is either in or out of bounds. This function is also available in Snooze mode. In this mode, the processor itself is stopped and consumes very little power, but ADC conversions continue under control of the hardware trigger. When a conversion triggers an ADC interrupt, the processor can then wake up from Snooze mode and resume normal operation.

COMMUNICATIONS

The RL78 features multifunction serial units. The devices with 25 pins or less have one such unit, the others have two. Only serial unit 2 provides LIN bus support.

A serial unit can function in asynchronous UART mode, in synchronous CSI mode (three-wire bus with clock, data in and data out signals, master and slave mode supported), and in simplified (master-only) I²C mode. Again, depending on the device, you can have up to four UARTs or eight CSI and/or simplified I²C ports. Of course a mix is also possible. Full I²C is possible with the specialized I²C unit.

UART0 and UART2, CSI00 and CSI20 provide Snooze mode functionality similar to the ADC. In Snooze mode, these ports can be made to wake up on the arrival of incoming data without waking up the CPU. If the received data is interesting enough, it is also possible to wake up the CPU.

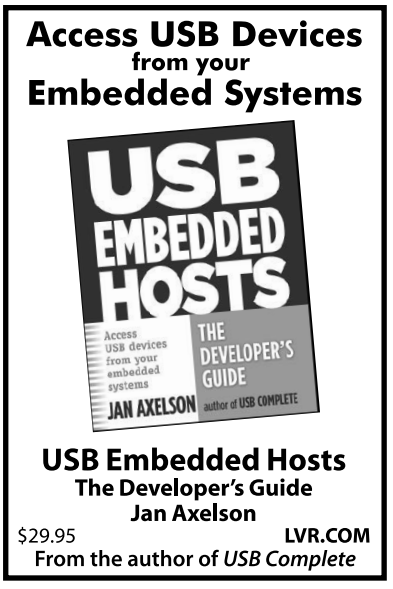
LIN communications are possible with UART2 together with Timer 7 of Array 0. The LIN bus is an inexpensive alternative to the CAN bus in automotive systems to control simple devices like switches, sensors, and actuators. LIN only uses one wire and is rather low speed (20 Kbps maximum). The timer takes care of the LIN synchronization issues and the UART performs the (de)serialisation of the data.

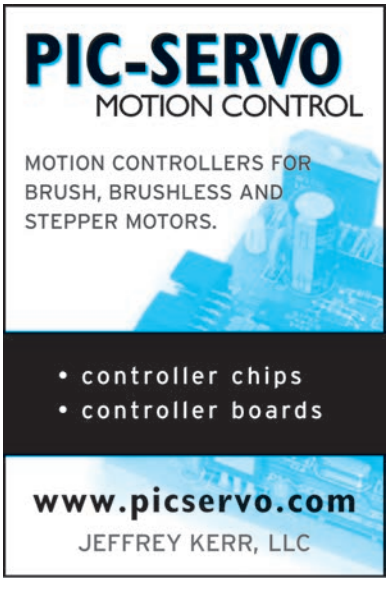
Full blown I²C communication is possible with the specialized I²C peripheral IICA. The 80-pin and more devices have two channels, the others only one. Communication speeds up to 20 MHz are permitted to enable I²C "fast mode" (3.5 MHz) and "fast mode plus" (10 MHz). This module is capable of waking up the CPU from Stop mode.

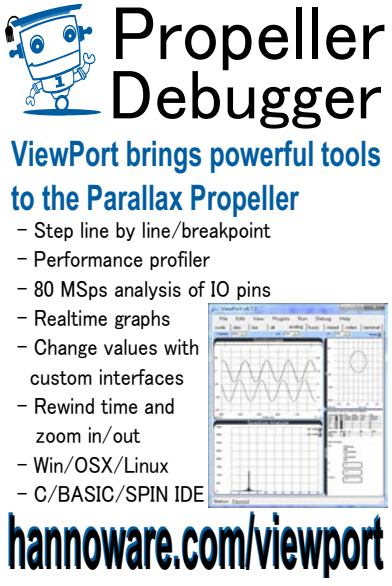
MATH ACCELERATORS

Of interest is the hardware multiplier and divider module intended for filtering and FFT functions. This module is capable of 16 × 16 bits signed and unsigned multiplications and divisions producing 32-bit results. It can also do 16 × 16 bit multiply-accumulate. We are talking about a module here, not an instruction, meaning that you have to load the operands yourself in special registers and get the result from yet another. The multiplication itself is done in one clock cycle, a division takes 16. The accumulate operation adds another cycle.

Another special math function is the binary-coded decimals (BCD) correction register that enables you to easily transform binary calculation results into BCD results.







DIRECT MEMORY ACCESS

To speed up data transport without loading the CPU, the RL78 core features direct memory access (DMA), up to four channels. DMA transfers up to 1,024 words of data (8 or 16 bit) to and from SFRs and RAM and they can be started by a range of interrupts (e.g., ADC, serial, timer). Although DMA transfers are done in parallel with normal CPU operation, it does slow down the CPU. For time-critical situations, it is possible to put a DMA transfer on hold for a number of clock cycles and let the CPU finish its job first.

INTERRUPTS

Interrupts are pretty standard on the RL78 and many sources are available. The "key interrupt" function on the other hand is less common. It provides up to eight (depending on the device, you guessed it) key or push button inputs that are ORed together to generate an interrupt on a key press (active low).

OPERATING MODES & SECURITY

Besides the aforementioned Stop and Snooze modes, the RL78 also provides a Halt mode. In this mode, the CPU is stopped but the clocks keep running, making a fast resume possible. In Stop mode, the clocks are stopped too reducing power consumption more than in Halt mode. Snooze mode is like Stop mode, but with one or more peripherals in a snoozing state, ready to wake up when something interesting happens. Interrupts can be used to wake up from Snooze, Stop, or Halt mode. A reset usually works too.

Reset, by the way, can have seven origins, three of which are related to safety functions: illegal instruction, RAM parity, and illegal memory access. Two others involve the power supply: power-on reset (POR) and low-voltage detection (LVD). All these reset options are needed to conform to the International Electrotechnical Commission (IEC) 60730-1 ("Automatic Electrical Controls for Household and Similar Use; Part 1: General Requirements") and IEC 61508-SER ("Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems") safety standards. Since the RL78 is compliant, it also implements flash memory CRC checking, protections to prevent RAM and SFRs to be modified when the CPU stops functioning, an oscillator frequency-detection circuit, and an ADC self-test function.

The hardware used for the flash memory CRC check is also available as a general-purpose CRC module for user programs. It implements the standard CCITT CRC-16 polynomial ($X^{16} + X^{12} + X^5 + 1$).

The RAM guard function protects only up to 512 bytes, so be careful where you put your sensitive data.

FLASH & FUSES

Those familiar with the fuse bytes of PIC and AVR processors will be happy to know that the RL78 contains four of them, the option bytes that configure such things as the WDT, low-voltage detection, flash memory modes, clock frequencies, and debugging modes.

Flash memory is divided into two parts, program memory and data memory, and it can be programmed in-circuit over a serial interface. A boot partition is available too. This partition uses a kind of ping-pong mechanism called "boot swapping" to ensure that a valid bootloader is always programmed into the boot partition so that even power failures during bootloader programming will not harm the boot partition. A flash window function protects the memory against unintentionally reprogramming parts of it.

SOUNDING OFF

This concludes our voyage through the Renesas RL78 core. As you have seen, the RL78 offers many interesting peripherals all combined in a flexible low-power optimized design. Thanks to the integrated oscillator and other functions, an RL78 MCU can be used with very little external hardware, enabling inexpensive and compact designs. Once you master its Snooze mode and your low-power design skills, you can use this MCU family in battery-operated metering applications, for instance, but I am sure you can think of something more surprising.

Clemens Valens (c.valens@elektor.fr) is Editor-in-Chief of Elektor Online. *He has more than 15 years of experience in embedded systems design. Clemens is currently interested in sound synthesis techniques, rapid prototyping, and the popularization of technology.*

REFERENCES

[1] Texas Instruments, Inc., "Ultra-Low Power MSP430 – The World's Lowest Power MCU," 2011, www.youtube.com/watch?v=eG4xDCq17jc.

[2] Microchip Technology, Inc., "AN1267: nanoWatt and nanoWatt XLP Technologies: An Introduction to Microchip's Low-Power Devices," 2009, http://ww1.microchip.com/downloads/en/AppNotes/01267a.pdf.

[3] Renesas Electronics Corp., "RL78 Family," www.renesas.com/pr/mcu/rl78/index.html.

RESOURCES

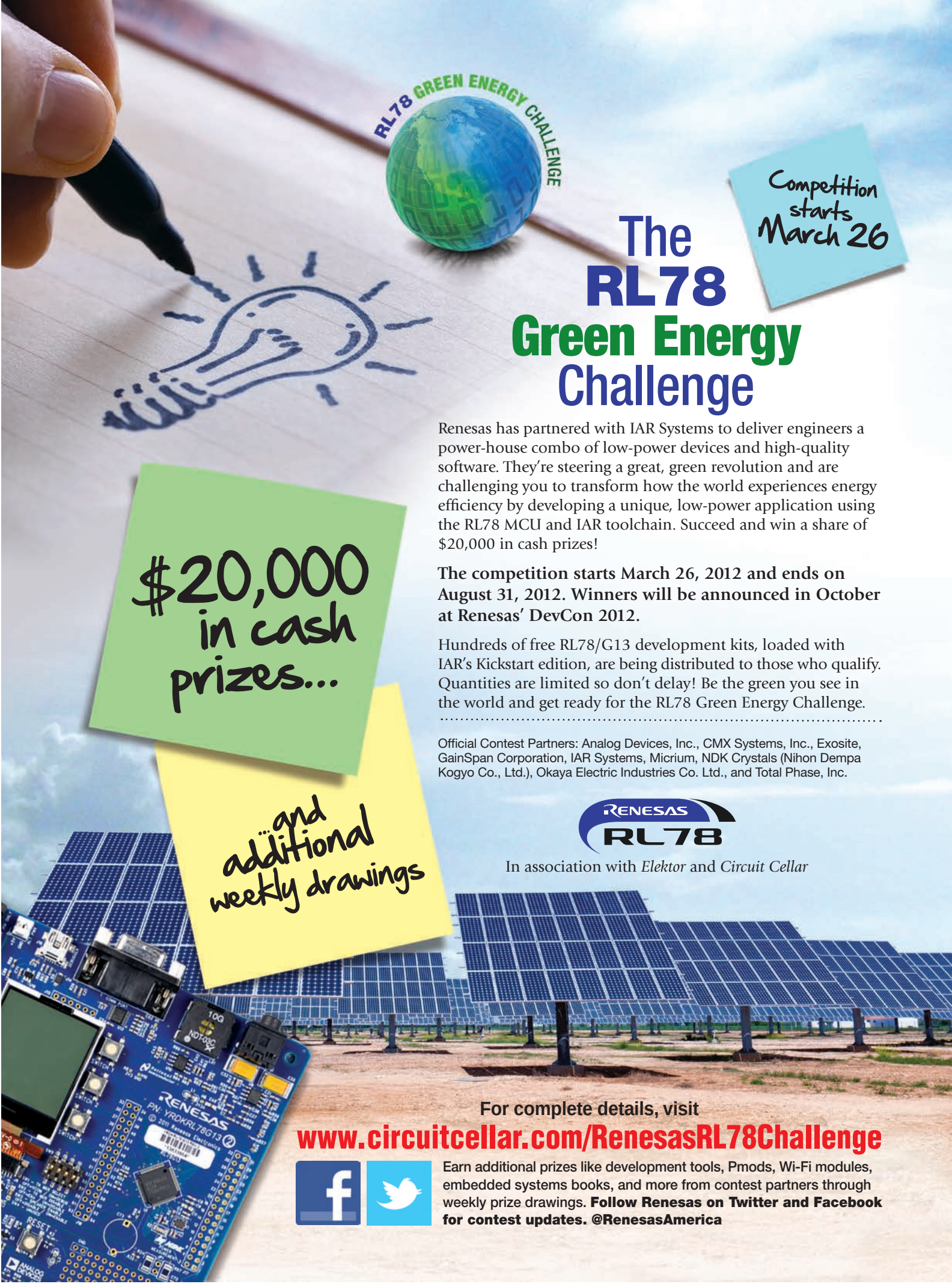
International Electrotechnical Commission (IEC), "60730-1, Automatic Electrical Controls for Household and Similar Use; Part 1: General Requirements," 2002.

———, "61508-SER, Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems," 2010.

Renesas Electronics Corp., Renesas Rulz, "RL78/G13 Demonstration Kit," www.renesasrulz.com/community/demoboards/rdkrl78g13.

SOURCE

RL78 Family of microcontrollers
Renesas Electronics Corp. | www.renesas.com



The RL78 Green Energy Challenge

Renesas has partnered with IAR Systems to deliver engineers a power-house combo of low-power devices and high-quality software. They're steering a great, green revolution and are challenging you to transform how the world experiences energy efficiency by developing a unique, low-power application using the RL78 MCU and IAR toolchain. Succeed and win a share of $20,000 in cash prizes!

The competition starts March 26, 2012 and ends on August 31, 2012. Winners will be announced in October at Renesas' DevCon 2012.

Hundreds of free RL78/G13 development kits, loaded with IAR's Kickstart edition, are being distributed to those who qualify. Quantities are limited so don't delay! Be the green you see in the world and get ready for the RL78 Green Energy Challenge.

Official Contest Partners: Analog Devices, Inc., CMX Systems, Inc., Exosite, GainSpan Corporation, IAR Systems, Micrium, NDK Crystals (Nihon Dempa Kogyo Co., Ltd.), Okaya Electric Industries Co. Ltd., and Total Phase, Inc.

RENESAS RL78

In association with *Elektor* and *Circuit Cellar*

For complete details, visit

www.circuitcellar.com/RenesasRL78Challenge

Earn additional prizes like development tools, Pmods, Wi-Fi modules, embedded systems books, and more from contest partners through weekly prize drawings. **Follow Renesas on Twitter and Facebook for contest updates. @RenesasAmerica**

If we haven't convinced you that the Circuit Cellar Archive...



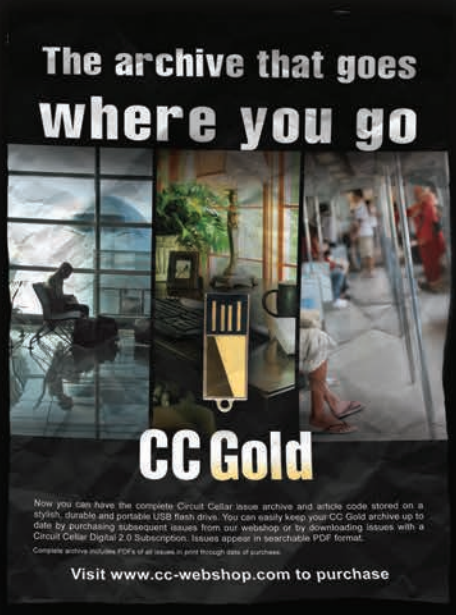
Has over two decades of engineering excellence

Which key opens the most doors?



Now you can have the complete Circuit Cellar issue archive and article code stored on a stylish, durable and portable USB flash drive. You can easily keep your CC Key archive up to date by purchasing subsequent issues from our webshop or by downloading issues with a Circuit Cellar Digital 2.0 Subscription. Issues appear in searchable PDF format.

Opens doors



Goes where you go



Compliments your style



Is the ultimate archive



Then maybe this will help.
Visit www.circuitcellar.com/bigdeal

Now you can have the complete Circuit Cellar issue archive and article code stored on a stylish, durable and portable USB flash drive. You can easily keep your archive up to date by purchasing subsequent issues from our webshop or by downloading issues with a Circuit Cellar Digital 2.0 Subscription. Issues appear in searchable PDF format.

Complete archive includes PDFs of all issues in print through date of purchase.
(For display purposes) USB drive shown is larger than the actual size.

by Robert Lacoste (France)



Sigma-Delta Modulators 101

You're familiar with high-speed analog-to-digital converters (ADCs). Now let's delve into the world of slower ADCs, which are also known as sigma-delta converters. These low-cost converters are used in audio devices and can provide up to 24 bits of resolution. Now you'll learn how they actually work.

Welcome to the Darker Side again. In my last column ("Playing with High-Speed ADCs," *Circuit Cellar* 259, 2012), I wrote about ultra-high-speed analog-to-digital converters (ADCs). These chips are really impressive, with pipelined architectures and speeds up to several hundred million samples per second. By the way, I just read that the latest high-end Infiniium 90000X digital oscilloscopes from Agilent have front-ends built around a custom astonishing 40-Gsps ADC. Do you know how they achieve it? By integrating no less than 80 precisely synchronized 500-Msps ADCs in parallel! As it may not be enough for your day-to-day measurements, two input channels can be paired to get an 80-Gsps acquisition speed, which is barely enough for the 33-GHz real-time bandwidth of these monsters. I had the pleasure of playing with such an instrument during a seminar. Well, I will buy one as soon as I have the budget. The problem is that it is, well, more than 50 times more expensive than what even my company could afford for an oscilloscope!

Let's come back to earth. Slower ADCs are interesting too. You have surely met one of the so-called sigma-delta converters. These chips are used everywhere: in weight scales, in thermometers, and in virtually any audio device from PCs up to high-end Hi-Fi systems. They provide up to 24 bits of resolution at an inexpensive cost. But how do they actually work? Why are they used so broadly? In fact, I must confess that I discovered sigma-delta converters thanks to *Circuit Cellar*. It was back in December 1998, through a very good article written by David Tweed ("Digital Processing in an Analog World, Part 3: Dithering Your Conversion," *Circuit Cellar* 101). Well, 14 years

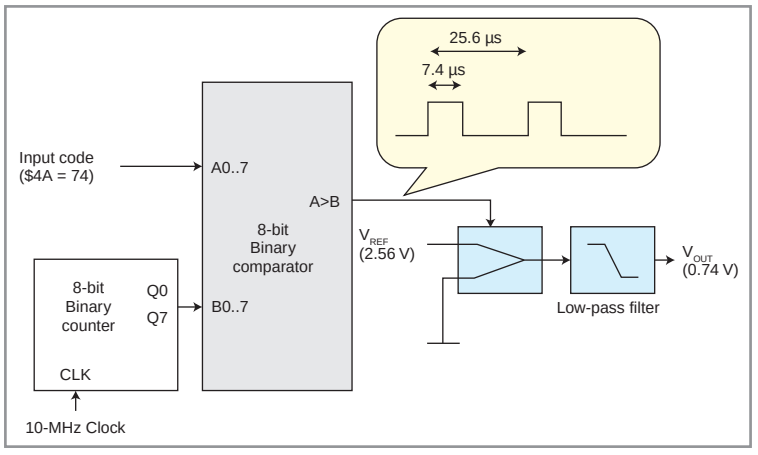


Figure 1—A basic way to implement a DAC with a single digital output is to use a pulse-width modulator (PWM). Here an 8-bit PWM generates a 7.4-µs long 2.56-V pulse each 25.6 µs, giving an average output voltage of exactly 0.74 V after a low-pass filter.

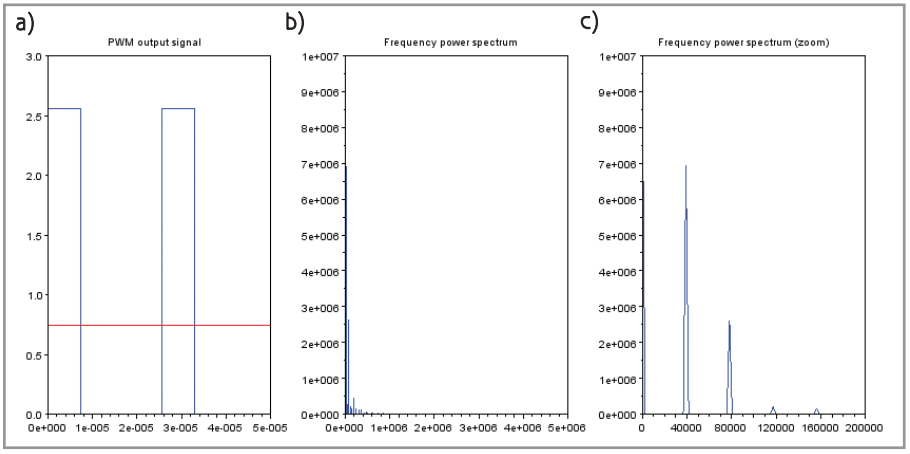


Figure 2—The disadvantage of a PWM-based DAC is its output spectrum. The center plot (b) is the frequency spectrum of the PWM signal shown on the left (a), and it is zoomed on the right (c). A high-power frequency component is visible at 39 kHz, which is the PWM frequency.

have passed; now it is my turn! Take your seat, this month I provide a journey to the sigma-delta world.

THE BASIC PWM-BASED DAC

Let's start with a digital-to-analog converter (DAC), as I guess it is easier to understand than its ADC counterpart. Imagine you need to generate a DC voltage adjustable from 0 V to 2.55 V (well, selected just to facilitate my explanations) with a resolution of 10 mV. You have a microcontroller and don't want to add too many external components so you are looking for a nearly full digital solution. What would you do?

If you are a regular reader, you may recall David Ludington's 2011 two-part article series, "High-Accuracy Voltage Reference Using PWM" (*Circuit Cellar* 255–256). You know you could solve this problem using a single digital output of the microcontroller and a software- or hardware-based pulse-width modulator (PWM) as shown in Figure 1. Assuming that the DAC is set for a 0.74-V output, the principle is the following: First, you generate a rectangular signal with a 74/256 duty cycle. This is easily done with an 8-bit binary counter and an 8-bit magnitude comparator, which are available in timer modules of all microcontrollers. This rectangular signal then drives a switch, oscillating between, say, a 2.56-V reference voltage and 0 V. Therefore, the output voltage has an average DC voltage of 0.74 V (i.e., 74/256 × 2.56 V). You just have to low-pass the output signal and you will get your desired clean 0.74 voltage.

Easy, isn't it? As Ludington's article showed us, this PWM approach, when properly implemented, enables you to build precise DACs with impressive resolution at nearly no cost. But what is their main issue? Assume that the maximum clock speed you could use is 10 MHz. In my example, we are using a 8-bit binary divider, so the output signal will have a frequency of 39.06 kHz (i.e., 10 MHz/256). I made an INRIA Scilab simulation, which is available on *Circuit Cellar*'s FTP site. The resulting spectrum is shown in Figure 2. In order to get a clean DC output you will then have to implement a low-pass filter with a high rejection at 39 kHz or above. I already talked about antialiasing filters in my last column, so you know you will have to use a cutoff frequency of only a couple of kilohertz if you don't want to require a complex and costly filter. This is not a problem if you don't need to change the DAC setting too often, but in other applications, this filter will dramatically limit the speed of your DAC down to some thousands of samples per second only, due to this output filter.

THE SIGMA-DELTA DAC

How can you do better with a nearly fully digital design? With a sigma-delta DAC architecture, of course! According to Walt Kester in *The Data Conversion Handbook*, sigma-delta modulators have their roots in differential pulse-coded-modulation (PCM) systems invented in 1946 at the ITT Labs in France, and rediscovered a couple of years later both by Philips engineers in Holland and C.C. Cutler of Bell Labs in the U.S.[1] The idea is very simple: like a PWM, a sigma-delta modulator generates a digital signal with a duty cycle proportional to the desired output value (74/256 in our example). However, rather than generating a long sequence of ones first and then a long sequence of zeros as the PWM does, the sigma-delta solution spreads the ones and zeros through time as much as possible. You must understand that this doesn't change the average output value, so the filtered DC output is the same. However, as I will show you, this method—which is a form of what signal processing devotees call noise shaping—has a

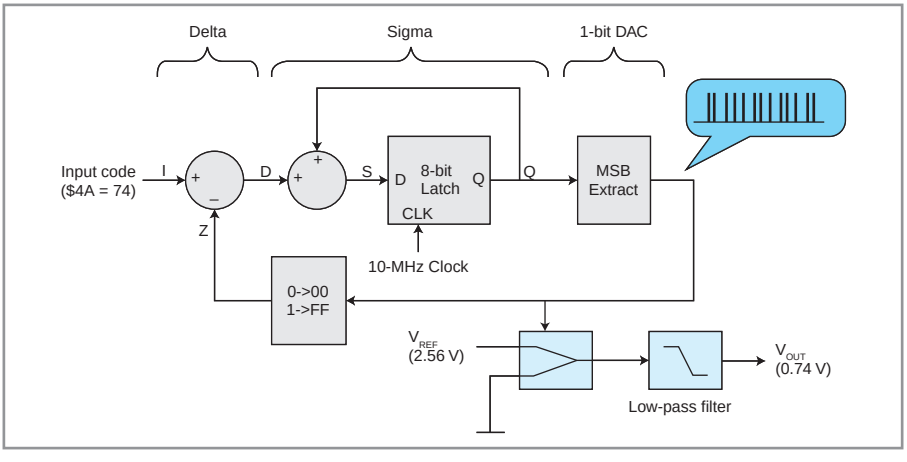


Figure 3—A sigma-delta DAC is also built around a 1-bit DAC, but its architecture enables it to spread the ones and zeros in a far better way than a PWM generator. The average value of the output signal is still the same.

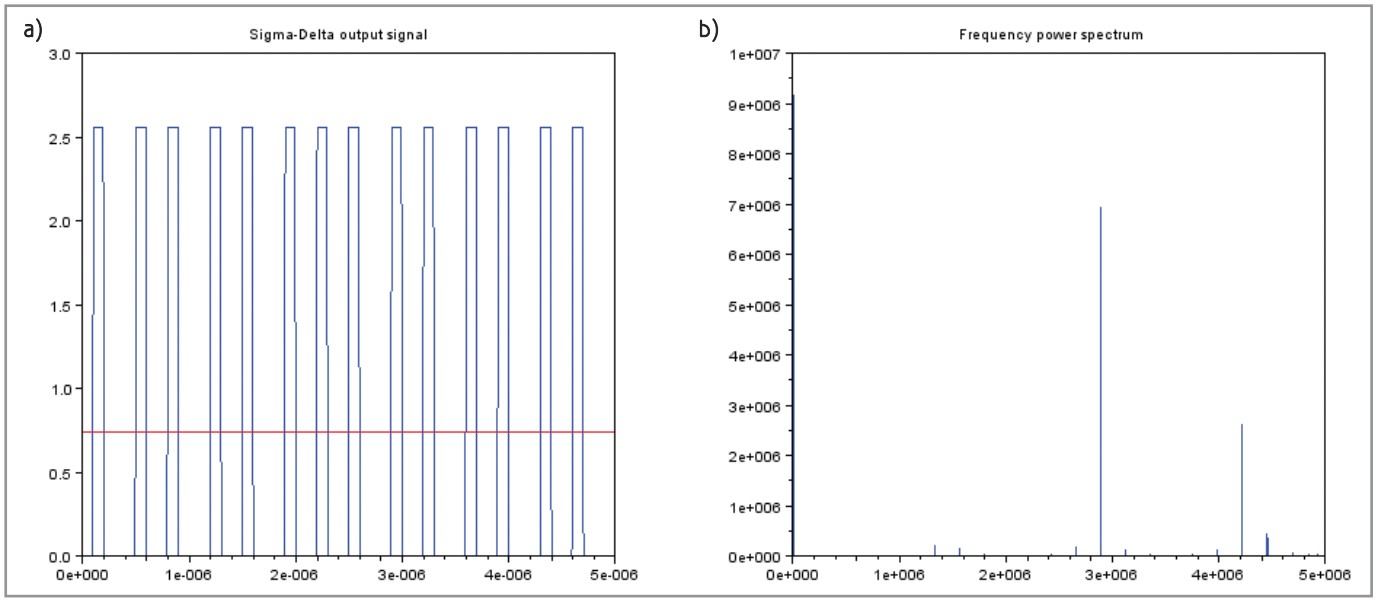


Figure 4a—This INRIA Scilab simulation shows the frequency spectrum of the sigma-delta DAC output illustrated in Figure 3. b—Here, the spurious energy is mainly close to 3 MHz, which is far easier to filter than with the PWM solution.

huge advantage on the filter side.

Let's start with the architecture. An 8-bit basic sigma-delta DAC is shown in Figure 3. It is a little more complex than the PWM, but it is still based on a single digital output, which is converted to an analog output value thanks to a voltage reference, a switch, and a low-pass filter. The difference is in the digital section. It starts with a subtraction (the "delta") between the desired DAC value and the output of the block, which is always full off (0) or full on ($FF). The result of this subtraction goes through a digital integrator (the "sigma"), and the most significant bit of its output is simply the output of the delta-sigma block. How does it work? In the long term, an integrator would diverge if its input is not null on average. Therefore, as its output is connected through a negative feedback to its input, we could be sure that the input of the integrator is null on average. But this input is the output of the subtractor. So, on average, the output signal is equal to the input code, which is exactly what we were looking for!

Input (I)	Delta (D = I – Z)	Integral (S = D + Q)	Latch (Q = S(n – 1))	1-bit DAC (Z = MSB of Q)
74	74	74	***0***	0
74	74	148	74	0
74	–181	–33	148	255
74	74	41	–33	0
74	74	115	41	0
74	74	189	115	0
74	–181	8	189	255
74	74	82	8	0
74	74	156	82	0
74	–181	–25	156	255
74	74	49	–25	0
74	74	123	49	0
74	74	197	123	0
74	–181	16	197	255
74	74	90	16	0
74	74	164	90	0
74	–181	–17	164	255
74	74	57	–17	0
74	74	131	57	0
74	–181	–50	131	255
74	74	24	–50	0
74	74	98	24	0
74	74	172	98	0
74	–181	–9	172	255
Average	**–0.375**	**75**	**75.375**	**74.375**

Table 1—The detailed, step-by-step behavior of the sigma-delta DAC shown in Figure 3. You will see that the output has one 1 for every two to three 0s, giving an average duty cycle of 0.29 (i.e., 74/256).

Let's start with the architecture. An 8-bit basic sigma-delta DAC is shown in Figure 3. It is a little more complex than the PWM, but it is still based on a single digital output, which is converted to an analog output value thanks to a voltage reference, a switch, and a low-pass filter. The difference is in the digital section. It starts with a subtraction (the "delta") between the desired DAC value and the output of the block, which is always full off (0) or full on ($FF). The result of this subtraction goes through a digital integrator (the "sigma"), and the most significant bit of its output is simply the output of the delta-sigma block. How does it work? In the long term, an integrator would diverge if its input is not null on average. Therefore, as its output is connected through a negative feedback to its input, we could be sure that the input of the integrator is null on average. But this input is the output of the subtractor. So, on average, the output signal is equal to the input code, which is exactly what we were looking for!

It is probably easier to understand it in an example. I simulated this sigma-delta DAC with a simple spreadsheet (see Table 1). Take your time and check the calculation of the first couple of rows, this is the best way to understand it. Then look at the output of the modulator, shown in the rightmost column. You will see that each one is separated by alternating sequences of two or three zeros. Remember that we wanted a duty cycle of 74/256, so for each one we must have, on average, 2.45 [i.e., (256 – 74)/74] zeros. And this is exactly what the sigma-delta modulator generates, thanks to the integrator.

As promised, the key advantage of this solution is visible when we switch to the frequency domain. Just compare Figure 4, which is the output spectrum of the sigma-delta modulator as simulated with Scilab, with the PWM case shown in Figure 2. Thanks to the inherent spreading of the one and zeros, the sigma-delta output has much higher frequency components. In this example, the first significant contributor is at 2.95 MHz, 75 times higher than the 39-kHz PWM frequency. The consequence? The output filter will be far easier to design. For example, if you required a DAC with

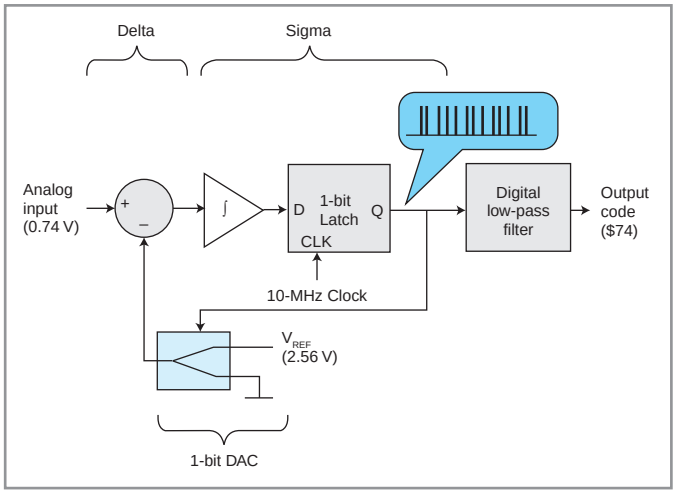


Figure 5—A sigma-delta ADC has roughly the same architecture as a sigma-delta DAC. Its digital and analog sections are just swapped. Compare this with Figure 3.

a band-pass of 20 kHz, you will just have to design a filter with a low attenuation at 20 kHz and a high attenuation at 2.95 MHz, which is very easy to do. If you used a PWM with the same clock rate, then you would need a high attenuation at only 39 kHz, which would be far more difficult.

WHAT ABOUT THE SIGMA-DELTA ADC ?

Let's move on to the ADC. A sigma-delta ADC has roughly the same architecture as a sigma-delta DAC; you just have to swap the analog and digital sections as shown in Figure 5. Here the subtractor and integrator are analog components, and the output filter is a digital low-pass filter. The principle is exactly the same. Thanks to the integrator, the output digital signal has a duty cycle proportional to the input voltage and includes only high-frequency spurious components. It is important to keep in mind that, like for a PWM, the output of a standard sigma-delta ADC is a single bit toggling between 0 and 1. Therefore the digital filter is intrinsically required for all sigma-delta ADCs. It removes as much noise as possible and provides the signal average value which is the representation of the input voltage. In a nutshell, a sigma-delta ADC is a 1-bit ADC with a very high oversampling ratio, built in such a way that most of the quantization noise goes in high frequencies, and followed by a good low-pass digital filter. To be precise, this digital filter also reduces the output data rate down to what is required to satisfy Shannon's limit, a step called decimation.

What are the drawbacks? Due to this digital filter, sigma-delta ADCs are more limited in speed than their cousins, such as successive approximation register (SAR), flash, or pipelined ADCs. They require a reference clock that is orders of magnitude faster than the actual ADC data rate, which typically limits their performance to a few hundred thousand samples per second. However, their advantages are numerous. First, they don't require any costly laser-trimmed precision resistor networks as they use no more than a simple 1-bit DAC. All their analog sections are quite





simple. Their architecture is also well matched to low-cost CMOS implementations, which provides huge cost benefits. And lastly, their resolution is only linked to the output digital filter, as all the remaining parts are only 1-bit. What this means is that with a sigma-delta converter you can easily increase the bit resolution by reducing the output data rate accordingly and implementing a better digital filter. That's why you can find a dual 24-bit sigma-delta ADC in your $30 PC sound card, whereas 24-bit converters built using other techniques would cost a fortune, if at all possible.

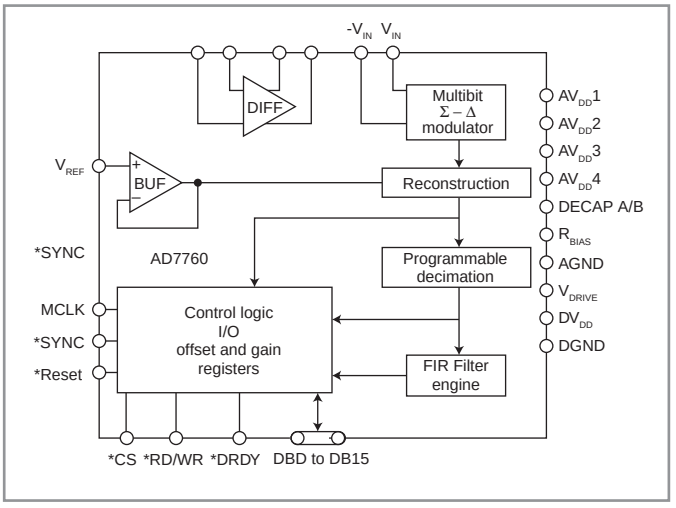


Figure 6—An example of the top-range sigma-delta ADC. The AD7760 from Analog Devices provides 24 bits at no less than 2.5 Msps.

HOW TO IMPROVE IT?

This presentation of sigma-delta ADCs is, of course, quite simplified. Engineers have been working in this area since the 1950s. The first improvement, used in a lot of actual ADCs and DACs, is about the feedback loop between the output of the converter and the delta subtracting stage. The architectures I've showed you are called first-order sigma-delta modulators. There is a way to build second-order or higher-order modulators which includes more than one feedback path and more than one integrator. I don't want to go into the details here, but these techniques enable you to achieve an even better noise shaping, thus improving the overall performances at the expense of a more complex architecture and possible risk of instability.

The other usual improvement is to replace the 1-bit DAC used in the feedback loop with a multi-bit DAC, even if it is only 2 or 4 bits wide. This enables you to improve the dynamic range of the converter and to reduce the risks of instability so it is commonly used with second-order converters. The drawback is that the multi-bit DAC may degrade the overall performances if it is not accurate enough, so this DAC usually needs laser-trimmed resistors to get 16-bit or more overall linearity, and this increases the bill.

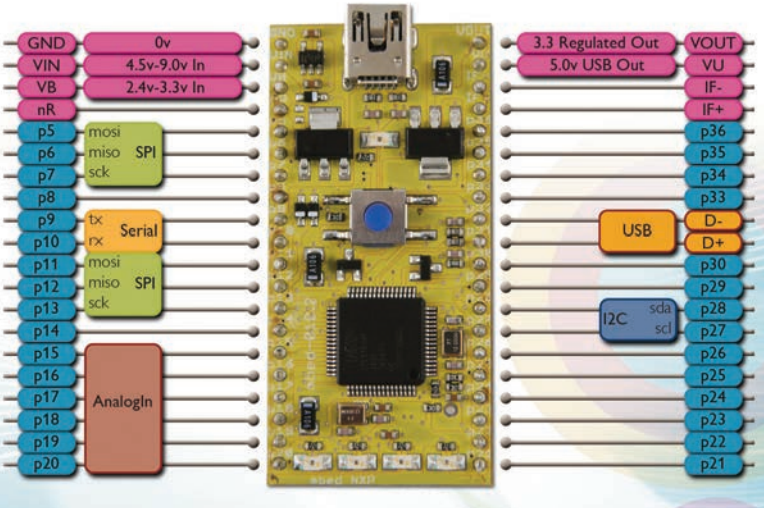
Other improvements are often included in the digital filter section. For example, measurement sigma-delta ADCs typically include a 50/60-Hz noise rejection filter in their digital-filter block to avoid any powerline-generated noise. This is easy, as all is digital here.

SOME ACTUAL CHIPS

I will not list every known sigma-delta chip. Just surf to the website of any ADC supplier and you will find plenty of them. Maybe we could just have a look at the extremes? On the high side, you may be interested in the Analog Devices AD7760 ADC (see Figure 6). This impressive piece of silicon provides a 24-bit resolution up to a 2.5 Msps maximum conversion rate. Who said sigma-delta converters are slow? Even more impressive, it achieves a 100-dB dynamic range at that speed, and up to 120 dB for audio applications. As always, its performances are in particular due to its output digital filter. Here the AD7760 integrates no less than three digital FIR filters in series, and you can program the response of these filters yourself, if you wish. (For more information on FIR filters, see my article, "No Fear with FIR: Put a FIR Filter to Work," *Circuit Cellar* 207, 2007.) This chip is also a good

mbed

mbed NXP LPC11U24 Microcontroller



Rapid prototyping for USB Devices, Battery Powered designs and 32-bit ARM® Cortex™-M0 applications

http://mbed.org

DESIGNSPARK

WINNERS ANNOUNCED ON MAY 7!

Turn a
hot idea
into a
cool solution.

It is time.

March 27 marks the end of the
DesignSpark chipKIT™ Challenge!

Soon, your eco-friendly and energy efficient design solutions will be in the hands of Elektor and Circuit Cellar's expert panel of judges.

Did you have what it takes to unleash the low-power combination of DesignSpark's free PCB software and Microchip Technology's chipKIT™ Max32™ development board? **Find out on May 7, 2012 when the Grand Prize Winners will be announced.**



$10,000 in cash prizes!

DesignSpark chipKIT™ Challenge

Visit **www.chipkitchallenge.com**

IN ASSOCIATION WITH:

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

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

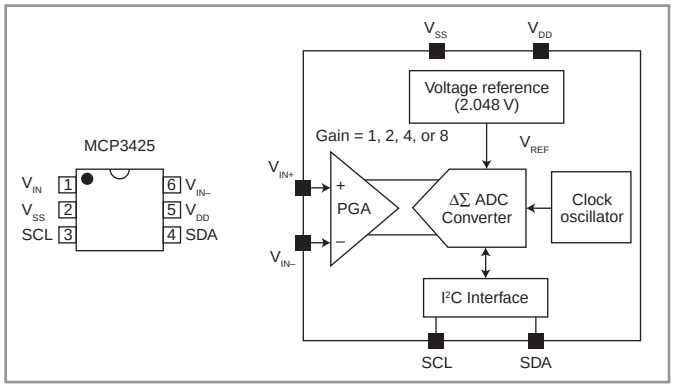


Figure 7—Microchip Technology has a number of low-speed,low-cost, low-power sigma-delta ADCs, like this MCP3425.

illustration of the advantage of the sigma-delta architecture in terms of linearity. Do you remember when I introduced the concept of integral nonlinearity (INL) two articles ago? The AD7760 shows a typical INL of 0.00076%! Just try to find a chip with equivalent performances with an architecture other than sigma-delta. Last but not least, its budgetary price in medium quantities is about $25, which doesn't seem too much for these performances, once again thanks to the sigma-delta design.

At the other end, if you're not interested in speed, but looking for an easy-to-use low-cost precision ADC with a simple I²C interface, Microchip Technology may have solutions for you, with ADCs such as the MCP3425 (see Figure 7). This low-power chip (145-µA typical at 15 sps) is packaged in a small six-pin SOT23-6 package and includes a full 16-bit sigma-delta ADC, a differential input buffer with programmable gain, and a voltage reference! It costs less than $2 in low volumes. Such a chip seems ideal for Wheatstone bridge applications (e.g., weight scales, magneto-resistive sensors, etc.), but also for precision temperature measurement or current sensing. Look at the datasheet for some applications.

As a last example, are you familiar with class-D audio power amplifiers? They are now used everywhere and in particular in low-power devices such as mobiles phones and tablets. A class-D amplifier is actually a 1-bit oversampling DAC directly driving the loudspeaker through a pair of push-pull power transistors. The filtering is done directly by the loudspeaker itself or with the help of a small external inductor. And do you know how to build an efficient 1-bit oversampling DAC? With a sigma-delta modulator, of course! Chips are numerous, but a typical example is the National Semiconductor LM4670 switching audio amplifier. Its architecture is shown in Figure 8. Such audio amplifiers have two key advantages: they don't require any large components, and they have a far higher efficiency than traditional class AB amplifiers. Here the LM4670 can drive a 4-Ω loudspeaker up to 2.3 W continuously, with efficiency above 75% at typical audio levels.

BUILT IT YOURSELF!

The sigma-delta technology can also be used without any specific component. Long-time readers may remember my NeuralStamp project, which won the Circuit Cellar/Freescale Design99 contest ("Neural Stamp: A Low-Cost Neural Network Processor," *Circuit Cellar* 114, 2000). In this design, I used a Motorola MC68HC908GP20 microcontroller to drive eight analog outputs. And each output was in fact done through a software-based 7-bit sigma-delta DAC and a small external RC low-pass filter. You can use the same technique anywhere, as soon as you need something a little cleaner than a classical PWM-based DAC, with the disadvantage of some extra firmware and CPU time. The code is quite simple

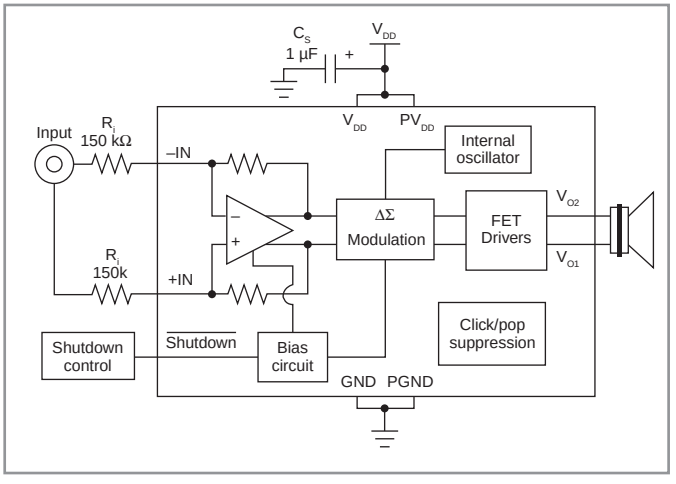


Figure 8—A class-D audio amplifier is also often built around a sigma-delta modulator, like this National Semiconductor LM4670.

AP CIRCUITS
PCB Fabrication Since 1984
As low as...
$9.95
each!
Two Boards
Two Layers
Two Masks
One Legend
Unmasked boards ship next day!
www.apcircuits.com
VISA MasterCard PayPal IPC MEMBER ASSOCIATION CONNECTING ELECTRONICS INDUSTRIES BBB

Listing 1—Code for a sigma-delta DAC (see Figure 3)

```
Variables:
            S = sigma-delta accumulator (-128 to +127)
            Vo = desired output mean value (0 to 127)
            X(N)=previous output binary value (0/1)
            X(N+1)=next output binary value (0/1)

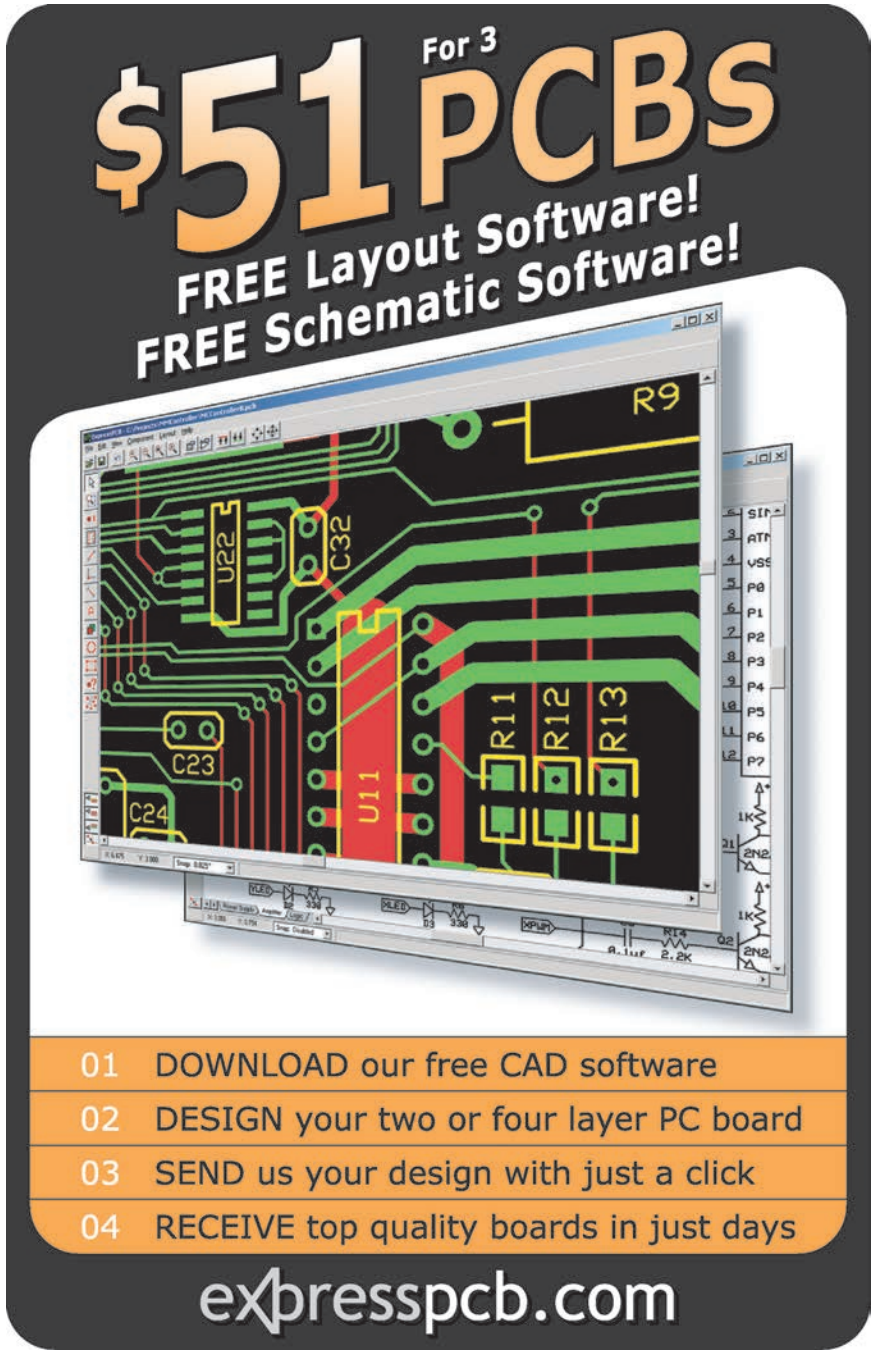
Algorithm:
            For each time step
                        S = S + Vo
                        if X(N)=1 then S=S-128
                        if S>0 then X(N+1)=1
                        else X(N+1)=0
            Done
```

(see Listing 1), and you will understand it by looking back to Figure 3.

You can also implement a sigma-delta ADC yourself. I have never done it personally, but you will find very clever application notes from microcontroller suppliers. For example, Marc Buccini from Texas Instruments has published an application report showing how to build a 12-bit sigma-delta millivoltmeter using an MSP430 microcontroller and its internal analog comparator.[2] The only external components are a resistor and a capacitor, which form the analog integrator! The same idea is also used by Dieter Peter from Microchip Technology. His application note shows how to build a 10-bit sigma-delta ADC at no cost on a Microchip Technology PIC16C6*xx* which doesn't have any ADC on board.[3] Nice idea, isn't it? Both use a very simple digital filter design in their firmware. Basically, they count the number of ones and zeros, which makes a moving-average digital filter. Such a filter is easy to implement but has some weaknesses compared to more complex digital filters, such as FIR. They have a sin(x)/x frequency response. Roughly speaking, this means they have a poor attenuation at some frequencies, and in particular at frequencies close to 1.5 times the averaging frequency. Argh, I guess I will need another column to go into more detail! Anyway, don't hesitate to experiment and to improve these design ideas.

WRAPPING UP

Here we are. I hope you have a better concept of sigma-delta technology. Basically, it's nothing more than a clever way to implement a 1-bit oversampling DAC or ADC, with the key advantage of good noise shaping (which is a complicated way to say that noise is at high frequencies in order to easily filter it). You also know that the output filter is key to the behavior of a sigma-delta converter, whether it is a simple RC filter for a DAC output or a digital filter for an ADC output. Now just play with them because they are no longer on your darker side!



A CAN Engineer's Most Valuable Tool



Komodo CAN Duo Interface

FINALIST
2012 BEST IN TEST

Total Phase's powerful Komodo CAN Duo Interface is a USB-to-CAN adapter and analyzer.

Host Adapters & Protocol Analyzers for USB, I2C, and SPI also available



For a special discount, visit
www.totalphase.com/offer/CC521968

TOTAL PHASE
www.totalphase.com
Industry-leading embedded systems tools

Robert Lacoste lives near Paris, France. He has 22 years of experience working on embedded systems, analog designs, and wireless telecommunications. He has won prizes in more than 15 international design contests. In 2003, Robert started a consulting company, ALCIOM, to share his passion for innovative mixed-signal designs. You can reach him at rlacoste@alciom.com. Don't forget to write "Darker Side" in the subject line to bypass his spam filters.

PROJECT FILES

To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2012/261.

REFERENCES

[1] W. Kester, *The Data Conversion Handbook*, Analog Devices, Inc., 2004, www.analog.com/library/analogDialogue/archives/39-06/data_conversion_handbook.html.

[2] M. Buccini, "An MSP430F11x1 Sigma-Delta Type Millivolt Meter," Texas Instruments, Application Report SLAA104," 2000, www.ti.com/lit/an/slaa104/slaa104.pdf.

[3] D. Peter et al, "Make a Delta-Sigma Converter Using a Microcontroller's Analog Comparator Module," AN700, Microchip Technology, 2002 ww1.microchip.com/downloads/en/AppNotes/00700a.pdf.

RESOURCES

U. Beis, "An Introduction to Delta Sigma Converters," 2008, www.beis.de/Elektronik/DeltaSigma/DeltaSigma.html.

R. Lacoste, "Playing with High-Speed ADCs," *Circuit Cellar* 259, 2011.

———, "High-Accuracy Voltage Reference Using PWM (Part 2): Hardware Design," *Circuit Cellar* 256, 2011.

———, "No Fear with FIR: Put a FIR Filter to Work," *Circuit Cellar* 207, 2007.

———, "Neural Stamp: A Low-Cost Neural Network Processor," *Circuit Cellar* 114, 2000.

D. Ludington, "High-Accuracy Voltage Reference Using PWM (Part 1): Pulse-Width Modulation Theory," *Circuit Cellar* 255, 2011.

A. Rosti, "1-bit A/D and D/A Converters," 2004, www.cs.tut.fi/sgn/arg/rosti/1-bit.

D. Tweed, "Digital Processing in an Analog World (Part 3): Dithering Your Conversion," *Circuit Cellar* 101, 1998.

SOURCES

Infiniium 90000X Digital oscilloscope
Agilent Technologies, Inc. | www.agilent.com

AD7760 ADC
Analog Devices, Inc. | www.analog.com

MCP3425 ADC and PIC16C6xx microcontroller
Microchip Technology, Inc. | www.microchip.com

MC68HC908GP20 HCMOS Microcontroller
Motorola, Inc. | www.motorola.com

LM4670 Audio amplifier
National Semiconductor Corp. | www.national.com

MSP430 Microcontroller
Texas Instruments, Inc. | www.ti.com



▶ CUWIN6300

▶ CUWIN5500

▶ CUWIN5200

Windows CE 6.0 Touch Controller

CUWIN

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

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



C-Programmable Modular Industrial Controller

MOACON

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

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

BASIC with LADDER LOGIC CONTROLLER



only $29

New

Integrated CUBLOC Controller and I/O Board

CB210

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

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

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

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

CUPC

Industrial Touch Panel PC with ATOM Processor



▶ CUPC-P200

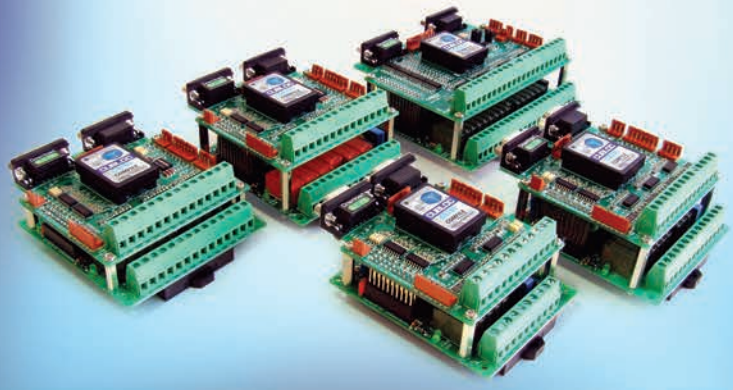
▶ CUPC-P220

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

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

CUSB

Integrated Industrial Controllers



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

by Ed Nisley (USA)



RFI Bypassing (Part 2)

Parallel Capacitors

The first part of this two-part series covered capacitor self-resonance, RF energy, and isolated capacitors. This article describes the process of figuring the total capacitance of several capacitors in parallel and explores why that process doesn't work with real capacitors and signals in the VHF and UHF bands.

First-year EEs know that capacitive impedance starts near infinity at DC and decreases toward zero with increasing frequency because that's what the equation for a capacitor's simplest circuit model tells them. My February column about self-resonance in real capacitors could jumpstart their knowledge and salvage some homework problems.

The rest of us know how to figure the total capacitance of several ideal capacitors in parallel: just add them all together. In this column, I'll explore why *that* simple process falls apart with real capacitors and signals in the VHF and UHF bands.

I'd like to use ideal capacitors, too, but we're stuck with real ones. After you understand how they behave at RF, though, you can use them to solve real problems.

PARALLEL CAPACITORS

The homebrew printed circuit board (PCB) in Photo 1 mounts on the back of a Wouxun KG-UV3D amateur radio from Powerwerx, replacing its standard lithium-ion battery pack. The circuit combines the datastream from a Byonics TinyTrak3+ GPS receiver with voice audio from a boom mic and earbuds, much as the circuit described in my October 2008 column ("HT Audio, GPS for APRS, and What Works," *Circuit Cellar* 219) did for my now-deceased Icom IC-Z1A. Check that column for details on the audio circuitry, because in this column we're more interested in RF.

The overall configuration resembles the test fixture in my February column, but with PCB traces and components replacing the air wiring. The measurements come from my HP8591 spectrum analyzer, with the right-hand RG-174 coaxial cable delivering the tracking generator signal through a 22-Ω surface-mount device (SMD) resistor tombstone-soldered to a pad and the left-hand cable connecting the analyzer input through a 470-Ω SMD resistor. Unlike the axial resistors I used in my last article, these resistors have about 1 nH of parasitic inductance



Photo 1—The signal from the tracking generator passing through these three SMD caps produces the waveform in Photo 2. The isolated solder blobs surrounding the caps connect the top ground plane islands to the continuous bottom surface ground plane. Copper foil soldered around the board edges bonds the perimeter. The rest of the circuitry isn't relevant for this test.

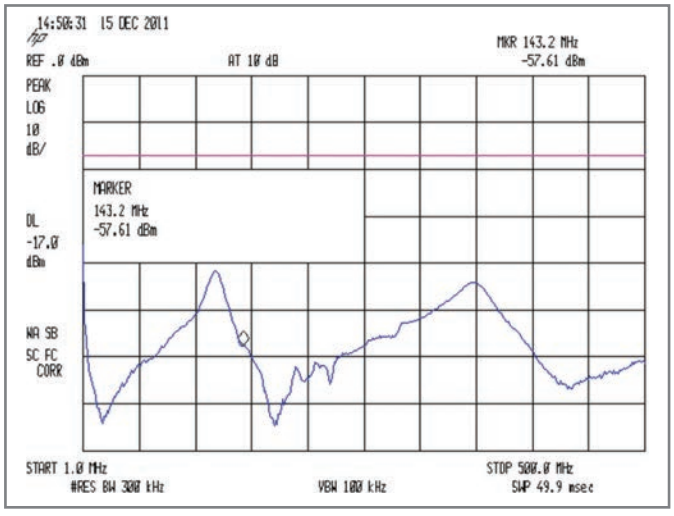


Photo 2—Each of the three distinct minima in this waveform corresponds to the self-resonant frequency of a capacitor in Photo 1. The purple line marks the maximum possible signal level set by the isolating resistors in series with the tracking generator output and spectrum analyzer input.

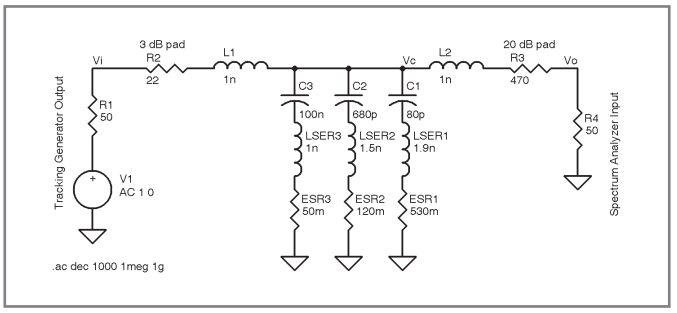


Figure 1—Plugging the measured values into this simple SPICE model produces results that closely match the actual circuit: SMD components really do behave that nicely!

and won't introduce any measurement complications.

The three paralleled SMD capacitors measure 80 pF, 680 pF, and 100 nF, respectively. They're hand-soldered in place, because I'm building this board component-by-component while taking measurements.

Photo 2 shows the circuit's response from 1 MHz through 500 MHz, with the three notches at 17, 159, and 405 MHz corresponding to each capacitor's self-resonant frequency. The purple line at –17 dBm shows the open-circuit response from only the padding resistors: the maximum possible output.

I measured each capacitor before soldering it in place, so we can compute each one's parasitic inductance from its self-resonant frequency using the familiar equation:

$$F_R = \frac{1}{2\pi\sqrt{LC}}$$

Rearranging the equation to produce the inductance L and plugging in the measured values shows that all three capacitors have about 2 nH of parasitic inductance, roughly what you'd expect from SMD components.

Although the PCB contributes some additional inductance and capacitance, it's now reasonably constant for a single capacitor. Later on, I'll show how even a slight change affects the results.

The 3 dB bandwidth of each resonance gives its Q:

$$Q = \frac{F_R}{BW}$$

At resonance, the capacitor's inductive and capacitive reactances will be numerically equal, with opposite signs. Because you can measure the actual capacitance and resonant frequency, the equation for capacitive reactance also gives the inductive reactance:

$$-X_L = X_C = \frac{-1}{2\pi f C}$$

Then, knowing X_C and Q, you can find the capacitor's equivalent series resistance:

$$ESR = \frac{|X_C|}{Q}$$

Although you can't read the numbers directly from Photo 2, a more detailed look with the spectrum analyzer puts the ESRs between 50 mΩ and 530 mΩ. Remember that the ESR is not a physical resistance, but a measure of the capacitor's behavior.

Plugging all those numbers into the SPICE model in Figure 1 produces a result that's quite close to Photo 2, as you can see in Photo 3. The spectrum analyzer has a linear frequency scale that dramatically compresses the low-frequency response; the simulation's logarithmic scale provides a better view of the details.

Regardless of the details, it's obvious that the impedance isn't the simple value predicted by the low-frequency model of adding together three ideal capacitors in parallel. In fact, that model applies only for frequencies well below the 17 MHz self-resonant frequency of the 100-nF capacitor.

You'll recall that I started this investigation to determine whether I could choose specific capacitor values to suppress RFI from the HT's transmissions. The amateur radio VHF band ranges from 140 MHz to 148 MHz, with the

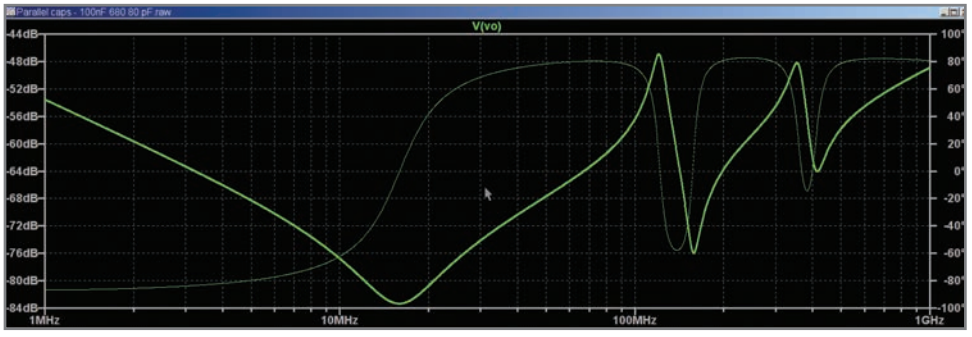


Photo 3—This simulation matches the response in Photo 2, after you account for the logarithmic frequency scale and the different axis ranges.

marker in Photo 2 near the 144.39 MHz APRS frequency. The UHF band from 420 MHz to 450 MHz lies within the broad rightmost notch.

Although the self-resonant frequency of the 680-pF cap lies slightly above the VHF band, the overall result seems to be good, with the response 40 to 50 dB below the open-circuit level. That's much better than the results of the single 100 nF capacitor I'd used in the previous circuit!

After you ponder Photos 2 and 3 for a bit longer, you might begin to wonder about those sharp peaks at 120 MHz and 350 MHz. Therein lies a tale.

INADVERTENT TANKS

In order to understand what's going on, consider the response shown in Photo 4, with 100-nF and 80-pF capacitors in parallel on a different part of the board. Due to the different layout, the two self-resonant notches occur at 17 and 430 MHz, with a single peak at 275 MHz about 17 dB below the reference line.

The impedance of a capacitor at its self-resonant frequency is purely resistive, because its capacitive and inductive reactances have equal magnitudes and opposite signs. Below that frequency, the impedance is capacitive:

$$|X_C| > |X_L|$$

Above self-resonance, however, the capacitor's overall impedance becomes *inductive*:

$$|X_C| < |X_L|$$

Therefore, at any frequency between 17 MHz and 430 MHz, the 100-nF cap acts as an inductor and the 80-pF cap acts as a capacitor, forming a classic resonant tank circuit. That's true even though the "inductor" looks like a surface-mount capacitor!

Analyzing a tank circuit built from components operating well below their self-resonant frequencies with small parasitic effects poses no particular challenge for anyone with a calculator. The situation shown in Photo 4, however, doesn't lend itself to easy analysis, because the parasitic effects become so large near self resonance.

Fortunately, we can deploy circuit simulation programs to handle the details!

I set the value of ESR2 in Figure 1 to 120 MΩ (which Linear Technology LTSpice abbreviates as "meg") to remove the effects of the 680-pF capacitor from the circuit.

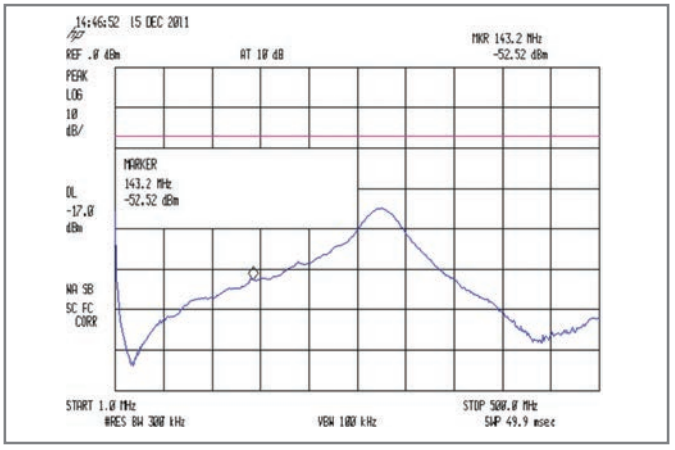


Photo 4—The response of paralleled 100-nF and 80-pF capacitors has a peak where their impedances form a resonant tank circuit.

The other simulated components aren't identical to the physical components and layout that produced Photo 4, so the results don't exactly match, but Photo 5 shows that it's close enough for our purposes.

The amplitude peak at 325 MHz corresponds with the phase crossing zero degrees, which shows that it's indeed a resonance where the inductive and capacitive reactances cancel.

The two peaks in Photo 2 and their simulated equivalents in Photo 3 come from the same phenomenon. In fact, any paralleled capacitors form a tank circuit with a resonant frequency between the two self-resonant frequencies.

Conventional wisdom tells you to parallel many bypass capacitors having widely spaced values ranging from microfarads to picofarads in order to reduce their collective impedance over a wide frequency range. While that's true below the lowest self-resonant frequency, the total impedance becomes very uneven above that point. If one of those unexpected tank circuit resonances happens to line up with the frequency of an undesired signal, the capacitor bank won't provide nearly the noise reduction you expect.

In this circuit, I want to suppress VHF and UHF signals radiated from the HT's antenna and coupled to the PCB through the audio cables. Those cables won't pick up other RF, so the lack of suppression adjacent to the amateur VHF band won't matter. For an application requiring suppression of low-VHF land-mobile or aviation signals, these capacitors wouldn't be appropriate.

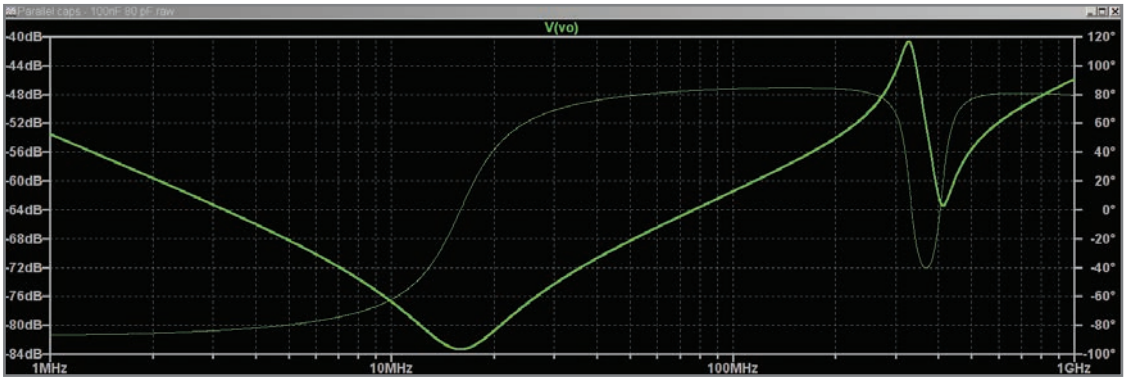


Photo 5—Simulating the parallel combination of a 100-nF and an 80-pF cap shows the same general behavior as the real devices in Photo 4. The frequency of the peak isn't exactly the same because the component values don't quite match.

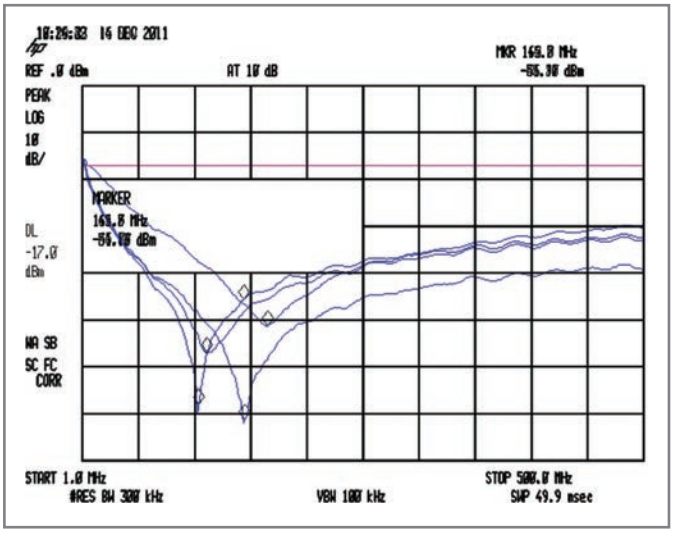


Photo 6—Three different 470-pF SMD caps had the three lowest self-resonant frequencies in this superimposed image, while a 150-pF cap was only slightly higher. The resonant frequency depends on both the capacitance and parasitic inductance, so you cannot predict it from the nominal capacitance value.

VARIATIONS ON A THEME

A capacitor's self-resonant frequency depends on both its capacitance and parasitic inductance. While the capacitance will (generally!) fall within a specified tolerance of the nominal value, ordinary ceramic SMD capacitors don't have an inductance specification. As a result, you cannot predict a capacitor's self-resonant frequency from its nominal capacitance.

Photo 6 superimposes four separate spectrum analyzer traces: three 470-pF capacitors and a 150-pF cap, all from different suppliers. If they all had the same inductance, you'd expect the 150-pF cap to have a much higher self-resonant frequency, but they're nearly indistinguishable.

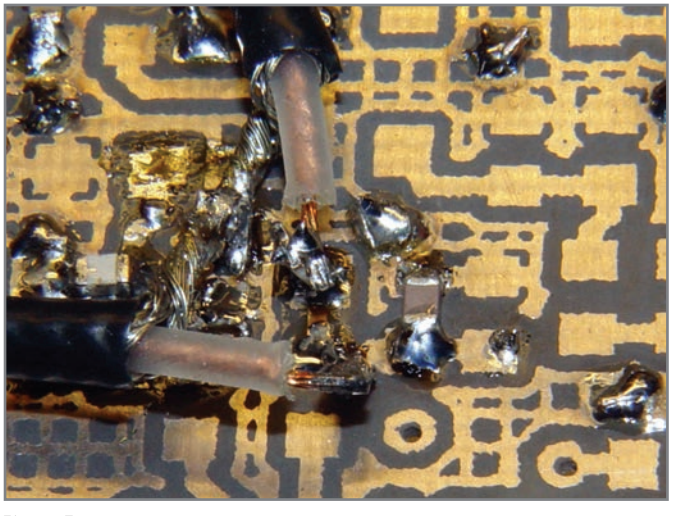


Photo 7—The tracking generator output and spectrum analyzer input connect to adjacent PCB pads on the left of the SMD capacitor. Connecting the spectrum analyzer to the pad on the right side changes the measured self-resonant frequency.

DIY Projects:
Simple Tracked Robot
#1327: Orangutan SVP-1284 Robot Controller
#1551: Rover 5 Tracked Chassis
#1490: 170-Point Breadboard
#1531: Pololu RP5 Expansion Plate
#1336: Wixel USB wireless programmable module
Sensors:
Orientation, Reflectance and More
#1255: CHR-UM6 Orientation Sensor
#958: QTR-1A Reflectance Sensor
#1265: MinIMU-9 Gyro, Accelerometer, and Compass - 9-channel inertial measurement unit (IMU)
Robots and Robot Kits:
Pololu 3pi and m3pi
#2150: 32-bit ARM mbed Dev. Board
#975: 3pi Robot - high-performance, C-programmable with ATmega328P MCU
#2152: m3pi Expansion Kit - enables use of the mbed Dev. Board with the 3pi Robot
Finding the right parts for your robot can be difficult, but you also don't want to spend all your time reinventing the wheel (or motor controller). That's where we come in: Pololu has the unique products - from actuators to wireless modules - that can help you take your robot from idea to reality.
Pololu
Robotics & Electronics
www.pololu.com

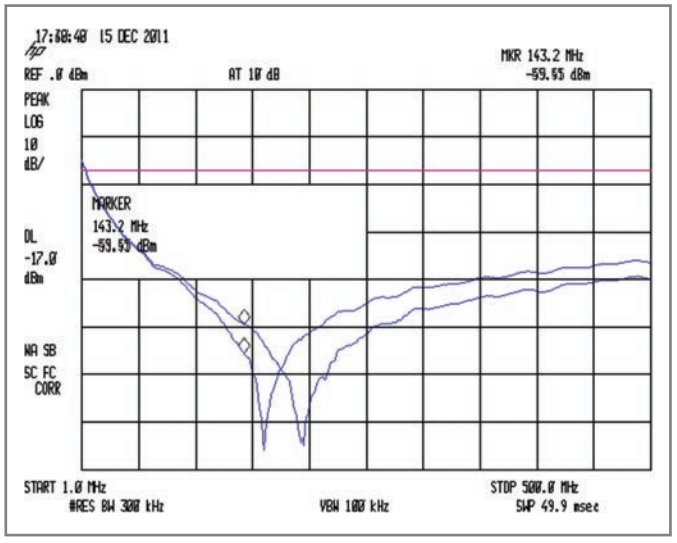


Photo 8—Measurements made at each end of the PCB trace in Photo 7 produce different self-resonant frequencies. It's not clear which value one should believe!

Homework: Assuming 2-nF parasitic inductance for all four caps, determine whether their capacitances fall within 10% of their nominal values.

Double Bonus: Given that the 150-pF cap resonated at 166 MHz, find its actual inductance and ESR.

Although I could hand-pick capacitors for specific self-resonant frequencies, the standard values found in the usual E12 or E24 series may not allow fine-tuning. I measured each capacitor on its intended PCB footprint, which became exceedingly tedious.

Worse, even when you find a suitable capacitor, the next batch (from a different supplier, of course) will have completely different properties. Explaining to your manager why the new capacitors won't work, even though they have the same nominal value and tolerance, may cause hard feelings.

Although we've been concentrating on each capacitor's internal parasitic inductance, the circuit surrounding it adds both capacitance and inductance. The 150-pF capacitor near the center of Photo 7 had a 190-MHz self-resonant frequency, at least with the tracking generator output and spectrum analyzer input connected as shown: both shield braids soldered to a common ground point and SMD resistors tombstoned on adjacent SMD pads.

Photo 8 shows the effect of moving the spectrum analyzer input to the pad on the other side of the SMD capacitor. Due to the different layout, the measured self-resonant frequency dropped by about 30 MHz, far more than the 3 dB width of the resonant notch. Note that the notch bandwidth is essentially unchanged, which means the ESR is about the same.

That degree of layout sensitivity is common to RF circuit designs, forcing engineers to specify not only component values, but also positions and even orientations. The length of a power trace, the number and size of the PCB vias along a signal trace, and the regularity of the ground plane can convert an RF amplifier into a vigorous oscillator or turn a bandpass filter into a signal stopper.

I had originally intended to measure the effect of audio-grade cables on the capacitor response, but it's fairly obvious that the notches are too narrow for any meaningful results. If moving the probe point by 8 mm changes the frequency by more than the width of the entire amateur VHF band, there's not much point in worrying about the details!

CONTACT RELEASE

Reader Lance Doyle uses KEMET Spice software to select KEMET chip capacitors. He observes: "I have used [it] to determine the package size for capacitors [...] in switching power supplies as the switching frequency of some of the newer chips is in the SRF range of chip caps." While KEMET Spice only deals with KEMET parts, you can use it to explore capacitor self-resonance in real parts.

Based on what I've measured, the parallel combination of a 100-nF and 470-pF capacitor will provide somewhat better VHF rejection than either one alone. The VHF notch may not be in exactly the right spot, but it'll be close enough for my simple needs.

After all this tinkering, it'll be a relief to finish that board and get it on the bike!

Ed Nisley is an EE and author in Poughkeepsie, NY. Contact him at ed.nisley@ieee.org with "Circuit Cellar" in the subject to avoid spam filters.

PROJECT FILES

To download the code, go to ftp://ftp.circuitcellar.com/pub/Circuit_Cellar/2012/261.

RESOURCES

B. Bruninga, "Automatic Packet Reporting System (APRS)," WB4APR, www.aprs.org.

E. Nisley, "Capacitors at RF," *Circuit Cellar* 175, 2005.

———, "HT Audio, GPS for APRS, and What Works," *Circuit Cellar* 219, 2008.

———, "RFI Bypassing (Part 1): Capacitor Self-Resonance," *Circuit Cellar* 259, 2012.

SOURCES

TinyTrak 3+ GPS Encoder
Byonics | www.byonics.com/tinytrak

KEMET Spice software
KEMET | www.kemet.com/kemet/web/homepage/kechome.nsf/weben/kemsoft

LTSpice IV
Linear Technology Corp.| www.linear.com/designtools/software

Wouxun KG-UV3D VHF/UHF amateur radio
Powerwerx | www.powerwerx.com/wouxun-radios

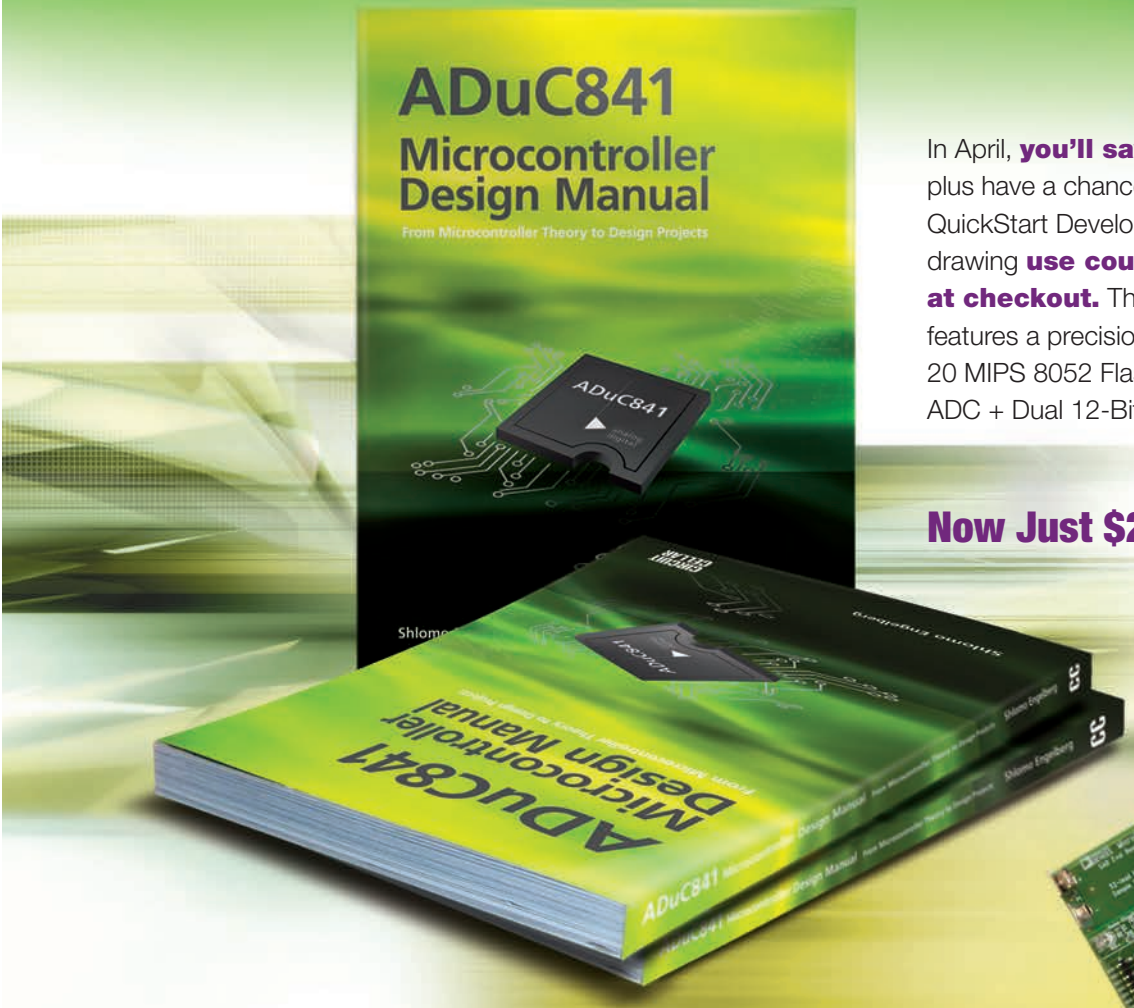
CIRCUIT CELLAR

Spring Book Deal!

Save 15% and enter to win a free development kit!

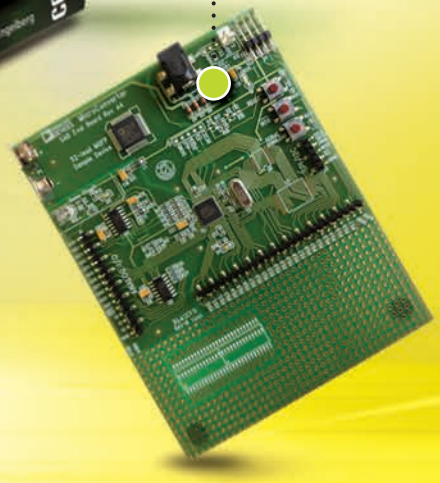
ADuC841 Microcontroller Design Manual:
From Microcontroller Theory to Design Projects

If you've ever wanted to design and program with the ADuC841 microcontroller, or other microcontrollers in the 8051 family, this is the book for you. With introductory and advanced labs, you'll soon master the many ways to use a microcontroller. Perfect for academics!



In April, **you'll save 15%** off the cover price plus have a chance to win a free ADuC841 QuickStart Development Kit! To enter the drawing **use coupon code ADKIT0412 at checkout.** The Analog Devices ADuC841 features a precision analog microcontroller: 20 MIPS 8052 Flash MCU + 8-channel 12-Bit ADC + Dual 12-Bit DAC.

Now Just $29.75



Buy it today!

www.cc-webshop.com

* By entering coupon code ADKIT0412 at checkout you are indicating that you have read and agree to the terms and conditions posted at www.circuitcellar.com/books. The ADuC841 kit drawing is open to U.S. residents only and ends April 30, 2012.

Elektor Shop

The world of electronics at your fingertips!



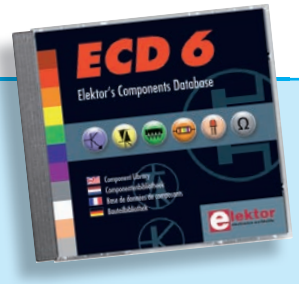
CD/DVD-ROMs

A whole year of Elektor magazine onto a single disk

DVD Elektor 2011

The year volume DVD/CD-ROMs are among the most popular items in Elektor's poduct range. This DVD-ROM contains all editorial articles published in Volume 2011 of the English, American, Spanish, Dutch, French and German editions of Elektor. Using the supplied Adobe Reader program, articles are presented in the same layout as originally found in the magazine. An extensive search machine is available to locate keywords in any article. With this DVD you can also produce hard copy of PCB layouts at printer resolution, adapt PCB layouts using your favourite graphics program, zoom in / out on selected PCB areas and export circuit diagrams and illustrations to other programs.

ISBN 978-90-5381-276-1 • $37.90



Elektor's Components Database

CD ECD 6

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

ISBN 978-90-5381-258-7 • $40.20



Circuits, ideas, tips and tricks from Elektor

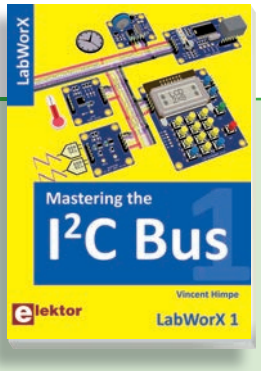
CD 1001 Circuits

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

ISBN 978-1-907920-06-6 • $55.70

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

Books

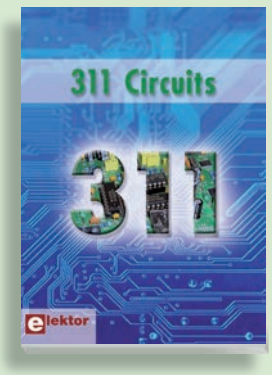


LabWorX: Straight from the Lab to your Brain

Mastering the I²C Bus

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

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

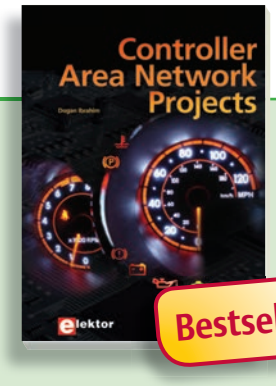


Creative solutions for all areas of electronics

311 Circuits

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

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



Free mikroC compiler CD-ROM included

Controller Area Network Projects

The aim of the book is to teach you the basic principles of CAN networks and in addition the development of microcontroller based projects using the CAN bus. You will learn how to design microcontroller based CAN bus nodes, build a CAN bus, develop high-level programs, and then exchange data in real-time over the bus. You will also learn how to build microcontroller hardware and interface it to LEDs, LCDs, and A/D converters. This book comes complete with a free CD-ROM.

260 pages • ISBN 978-1-907920-04-2 • $47.60

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

Elektor US
4 Park Street
Vernon, CT 06066
USA
Phone: 860-875-2199
Fax: 860-871-0411
E-mail: order@elektor.com

Kits & Modules



AndroPod

(February 2012)

With their high-resolution touchscreens, ample computing power, WLAN support and telephone functions, Android smartphones and tablets are ideal for use as control centres in your own projects. However, up to now it has been rather difficult to connect them to external circuitry. Our AndroPod interface board, which adds a serial TTL port and an RS485 port to the picture, changes this situation.

Andropod module with RS485 Extension

Art.# 110405-91 • $74.00



Improved Radiation Meter

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

Kit of parts incl. display and programmed controller

Art.# 110538-71 • $57.30

by George Novacek (Canada)



Is Software Development an Art or a Science?

This article takes a serious look at software engineering and programming. Topics such as automatic code generators, simulation programs, and testing are covered.

Most engineering disciplines started as an art. Gifted people developed a skill, often with barely enough understanding of its underlying principles, and practiced it successfully. Manufacturing steel is a good example. Early on, only a few highly paid artisans knew how to make it. But later, in the mid 19th century, the invention of the Bessemer process put steel manufacturing on a scientific basis. The skilled artisans were no longer needed and the price of steel plummeted.

Software development has followed a similar path. Originally the domain of a handful of gurus, it is becoming a scientifically well-defined activity that, ultimately, management would love to see performed by those proverbial trained monkeys. Some products make me wonder if we haven't already reached that goal.

People producing software generally fall into two categories. Software engineers are the creators, the people who find solutions, invent new algorithms and, in short, are the drivers of the industry. Programmers are the workers who implement engineers' ideas given to them in pseudocode, flowcharts, or other methods. They do the hard work of turning ideas into code. But this doesn't mean the engineer and the programmer can't be the same person. In small companies, they often are.

While it doesn't seem that science will successfully automate creative engineering processes any time soon (there have been attempts), in software development, the science is already making great strides by taking over mundane, repetitive tasks, such as source-code generation, version control, traceability, and testing.

AUTOMATIC CODE GENERATORS

Automatic code generators, albeit not very sophisticated, have been with us for a long time. In the mid-1990s I purchased a fuzzy logic design suite. I never used it, because customers shuddered at the word "fuzzy" and insisted on our time-proven technique of gain scheduling. Recently, I discovered the suite in a drawer and decided to try it. I was surprised to see that the code it generated was essentially what we did for gain scheduling, with one significant difference. Using the suite's graphic interface made the development much faster.

Since those early days, automatic code generators have come a long way. A good example of a modern automatic code generator I'm familiar with is the SCADE (an acronym for Safety and mission Critical Application Development Environment) Suite, a suite developed by Esterel Technologies. Unlike most other similar programs, it has been officially qualified for the development of safety-critical applications up to and including Level A, as per the DO-178B software standard. SCADE has been used by companies developing applications for industries including automotive, rail, communications, aerospace, medical, and military. A few examples are Dassault, Audi, Boeing, Lockheed, Siemens, and Airbus.

To appreciate how SCADE works, let's consider the development of an aircraft landing gear control system. The system engineers specify what is needed for its control in a System Requirements document. After the system requirements decomposition, where hardware and software requirements are defined for the

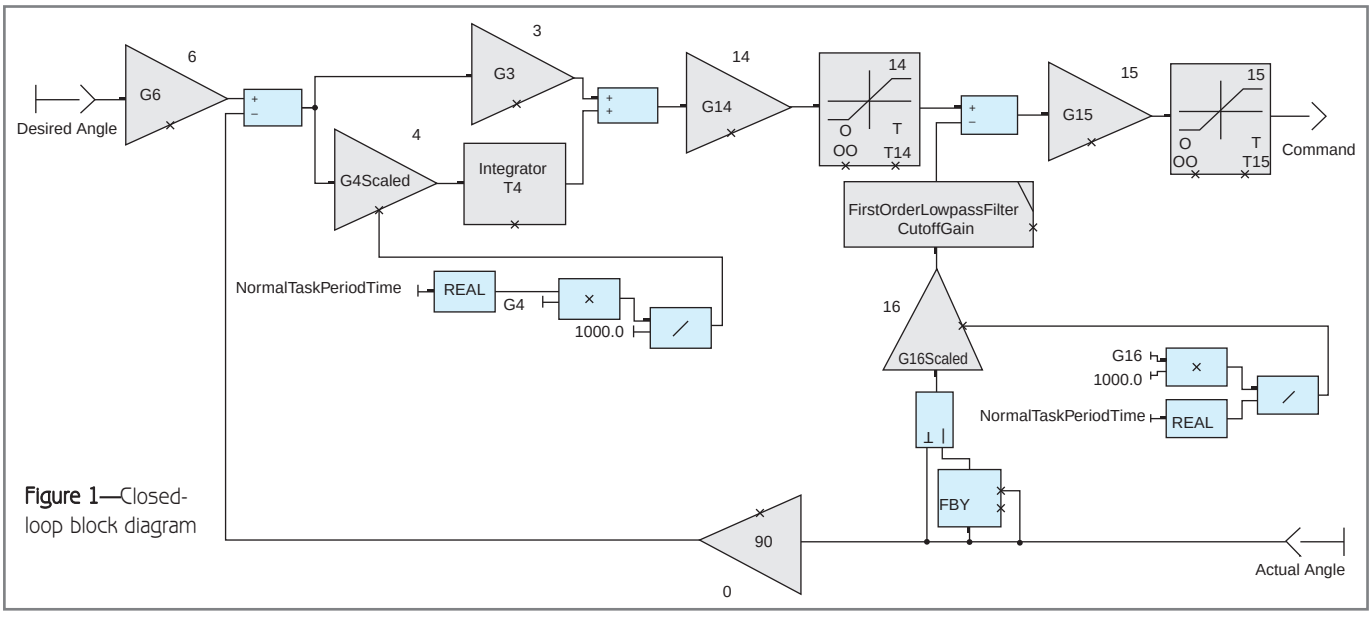


Figure 1—Closed-loop block diagram

embedded controller, the software engineer is given a Software Requirements document. Let's look at two specific functions contained in this document: the closed-loop control of the nosewheel angle and sequencing of the extension and retraction of the landing gear.

From the System Requirements document, the system engineer obtains the basic gains and limits for the nosewheel control and includes them in the Software Requirements document. The software engineer develops a block diagram of the software architecture, which is then simulated and tweaked, and the gains are modified, typically using Simulink from the MathWorks (or some other simulation program) until the simulated performance of the closed loop is compliant with the spec. The resulting block diagram is shown in Figure 1. The graphical rendering of the closed-loop control in Figure 1 becomes a part of the Software Design Description document.

Sequencing of the main landing gears is determined by a state machine. Just as with the closed-loop control, the engineer develops the state machine block diagram, which is also simulated until a satisfactory result is obtained. This is also rendered graphically, as shown in Figure 2, and it becomes a part of the Software Design Description document.

The Software Design Description document goes to a programmer to develop the source code. When it comes to modules such as the closed-loop control or the landing gear state machine shown in Figures 1 and 2, the design is fed through a graphical interface into the SCADE suite. The SCADE suite in turn generates the source code, which is then integrated into the entire embedded system firmware, usually as function calls. During hardware/software integration, some parameters may need to be adjusted, as they are affected by the accuracy of mathematical models for simulation.

CONFIGURATION CONTROL PROGRAMS

Version control is a crucial function during any development. There is nothing worse than a software team working on different versions of documents or source code. There were times when one of my suppliers kept updating a buggy Assembler without bothering to bump up the version number or even include the file modification date. Since every Assembler/compiler version change requires retesting of critical software, keeping track of the Assembler version was a nightmare. I also witnessed an aerospace manufacturer losing configuration control due to carelessness. The repercussions were disastrous. The certificating agency forced the manufacturer to scrap several months of development work and to go back to the last verified configuration.

Such occurrences can be prevented if the developer uses a configuration control program. Among the best known is IBM's Rational DOORS, which stands for dynamic object-oriented requirement system. DOORS is a database that keeps track of multiple users opening and modifying multiple documents or source codes and updates the version number accordingly.

TESTING, VERIFICATION, & VALIDATION

Another mundane, critical, and time-consuming task left to programmers is the process of testing, verification, and validation. Let's review each one.

Testing is a generic term meaning module testing, module integration, hardware/software integration, and so forth, simply ensuring that the code works as designed.

Verification is, to a large degree, a bureaucratic task. It ensures all specification requirements have been met and can be traced through documentation down to the source code, and every line of the source code has a corresponding requirement in the specification. It ascertains proper stack usage, data typing, worst-case timing, interrupt-conflict avoidance, range violation, and so forth. Verification can be performed on a host computer (e.g., a PC) or the target system. Using a host computer is often more convenient. The verification can be concurrently performed by a number of programmers without incurring the cost of several target systems.

Verification has also been automated to a large degree. Polyspace from MathWorks is one of several programs alleviating

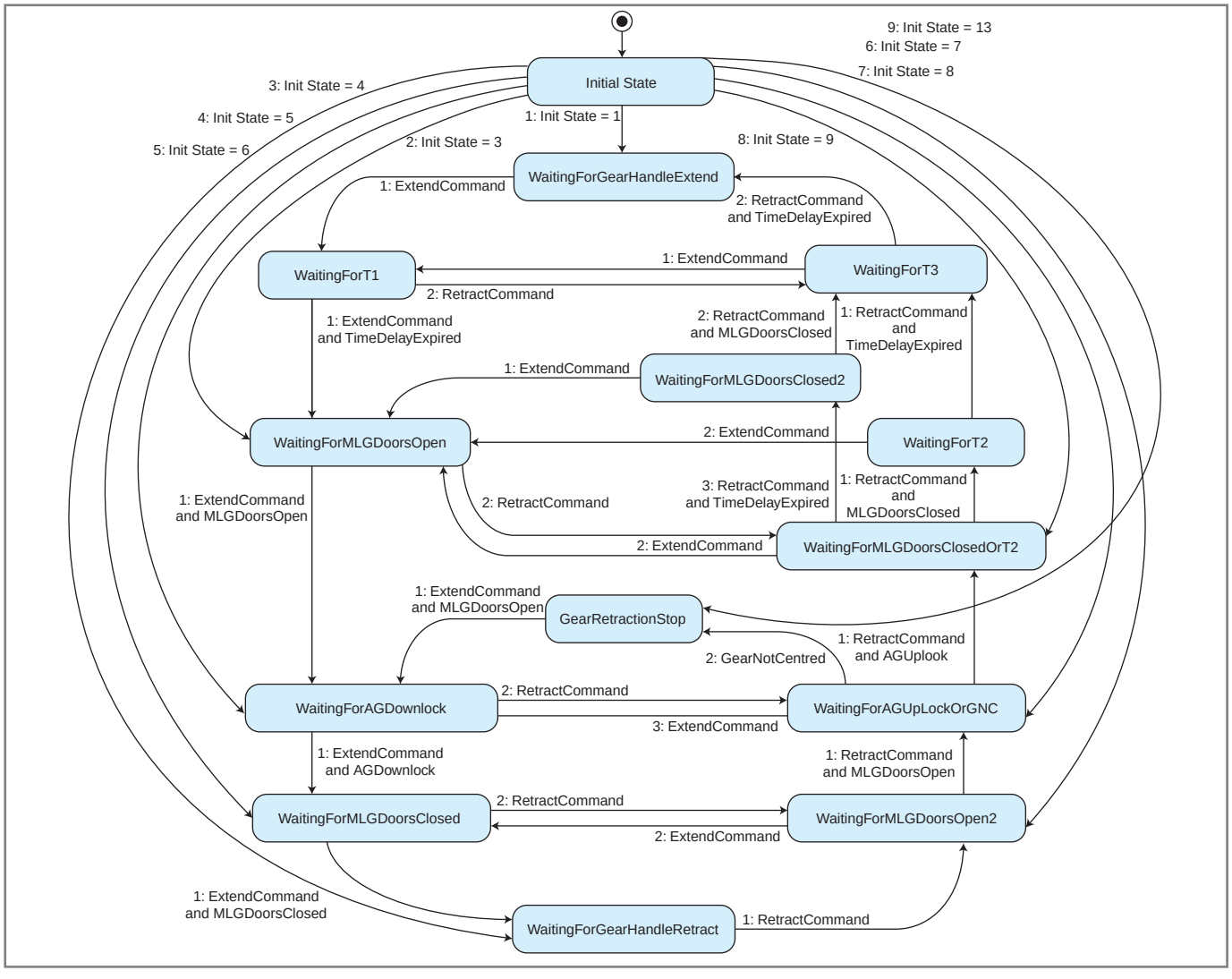


Figure 2—State machine graphical interface

the mundane tasks from the programmers' shoulders. Software verification is too large a subject to address here. I will take a closer look at this process in a future column.

Finally, validation is performed on the target system, to ensure the entire system works as expected. Here is when "the rubber meets the road" literally, and top-level specification errors may be discovered. Systems comprising mechanical and electrical parts are notorious for not behaving exactly as simulated. This is mainly because the mathematical models can't take into account all the variables, especially those of mechanical parts. In a well-designed system, the necessary changes can usually be confined to modifications of gains and limits. It is a good idea in an embedded software design to keep all values that could potentially need adjustment in a password-protected table, accessible and modifiable through a PC interface, such as USB.

Today, there are many tools available to software developers. Here, I only mentioned a few. I also want to allay software developers' fears of losing their jobs to trained monkeys. That is not yet on the horizon. In the meantime, the science will continue to make our lives easier and our products much less prone to human error.

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

RESOURCE

RTCA, Inc., "DO-178B Software Considerations in Airborne Systems and Equipment," 1992.

SOURCES

SCADE Suite model-based development environment
Esterel Technologies, Inc. | www.esterel-technologies.com

Rational DOORS requirements management tool
IBM Corp. | www.ibm.com

Simulink model-based design and Polyspace code verifiers
MathWorks, Inc. | www.mathworks.com

sensors expo & conference

The Leading Sensors Event in North America

Pre-Conference: June 5, 2012
Conference & Expo: June 6-7, 2012
Donald E. Stephens Convention Center • Rosemont, IL
www.sensorsexpo.com

25+ YEARS OF EXCELLENCE

The Only Industry Event in North America focused exclusively on sensors and sensor integrated systems.

Gain the knowledge you need from the leading experts and peers in the industry. This year's Conference Program includes more Technical Sessions in Tracks covering the topics that are most important to you and your business, including:

- Applied Sensing (NEW!)
- Energy Harvesting
- Novel Approaches to Measurement & Detection
- MEMS
- Mobile Sensing
- RFID (NEW!)
- Sensor and System Design (NEW!)
- Wireless Sensing Solutions



Identify specific solutions to your most difficult sensing, detection and control-related challenges on the Expo floor.

Sensors Expo brings together the largest and best-in-class showcase of sensing technologies and systems for attendees to evaluate and make informed decision. **Plus – see new Pavilions on the Expo floor to help you easily find your specific sensor solution.**

- BIG DATA & ANALYTICS PAVILION (NEW!)
- MEMS PAVILION
- MEMS INNOVATION AREA
- ENERGY HARVESTING PAVILION
- WIRELESS PAVILION (NEW!)





JUST ANNOUNCED!



Day 2 Keynote:
Assaf Biderman, Associate Director, SENSEable City Laboratory

Thursday, June 7, 2012 • 9:00 AM – 10:00 AM

Subscribers: Visit **www.sensorsexpo.com** to register today! Use discount code **A305F** for an ***EXTRA $50*** off a Gold or Main Conference Pass!

Scan **QR code** to register on the go!

Get the free mobile app at http://gettag.mobi

For more information, visit: www.sensorsexpo.com

INDUSTRY PARTNER: CIRCUIT CELLAR

PRODUCED BY: QUESTEX MEDIA

OFFICIAL PUBLICATION: sensors

by Bob Japenga (USA)



Getting Started with Embedded Linux (Part 4)

Linux Software Development Tools

This is the final installment in a four-part series about working with embedded Linux. Here you learn about the tools required for the software development process.

This article will discuss some of the software development tools available for developing embedded Linux. Let me start by identifying the basic steps and required tools in the software development process. The first tool that you need is a means to enter the software code into the computer. We'll call this "the editor." Then you need the means to take this representation of the code and turn it into instructions that the machine can read. Called in various environments the assembler, compiler, linker, and loader, we will call this "the code generator." Finally you need a means to debug, analyze and test your code. We will call this "the debugger." I am excluding some other management style tools (e.g., configuration control, librarians, etc.) because these don't usually need to be unique to the hardware and software you are developing. In addition, a nice tool to have is called an integrated development environment (IDE) that ties all of these together into a single user interface.

THE OLD DAYS

The first software program I wrote was in 1969. It was programmed in FORTRAN and was supposed to solve a differential equation. The development process involved creating a flowchart (the design), writing the code on punch cards (the editor), and submitting the punch card deck to the computer center to be run in batch mode (the code generator). Several hours later, you would get a printout of the results. Debugging consisted of reading the results of your printout and figuring out what you did wrong. Often the first results did not indicate the answer to the equation but that the instructions were not in the proper syntax. Debugging was similar to the old adage from your shampoo bottle: "lather, rinse, repeat." We found ourselves in a seemingly endless cycle of punching the program in, submitting the batch to run, and repeating the process until the program ran cleanly.

A few years later, I was doing embedded programming on PDP-8s and PDP-11s. Software development was initially done by entering the programs from a printer with a keyboard (!) into a line editor that enabled us to change and view one line of our code at a time. We had a code generator that could be run from the command line from the terminal. The program would be installed and run. We looked at the results and tried to figure out what we did wrong. Most debugging was done by inserting some form of instrumentation into the software that would turn on a light or print something to a printer or to the user interface to indicate progress, status, or some result. Sometimes we would insert a halt or a software trap into the code and read the results from the switch panel on the front of the computer. Such was the state of debugging embedded software in the early 1970s. Using these "tools," we developed (albeit slowly) "powerful" controls that revolutionized an inherently unsafe bottle-making process. Development continued this way for several years.

POWER TO THE PROGRAMMER

One day in the late 1970s, I was at a trade show and saw an IDE for developing software on a Motorola 6800 microprocessor. All the tools were

together in one seamless user interface. You could write your code with their editor and could see 24 lines of the text at a time! You could do the code generation and the screen would point you to the exact place where the error existed. You could run the code from the same screen. You could actually stop the code in specific locations (breakpoints) and view the results on the screen. You could step through the code and watch your variables change by name. You could stop the code when a location in memory was changed. I was in awe of having such a powerful tool. This was revolutionary.

Over the next 30 years, the software development tools advanced significantly enabling us to create even more powerful software. Real-time profiling (showing where your code is spending its time), memory-leak detectors (monitoring buffer over-runs and dynamic memory allocation), code coverage tools (telling us that we have exercised every line of code), and many other enhancements came along.

DÉJÀ-VU ALL OVER AGAIN

Then we started developing embedded Linux applications. With the exception of our graphical editors, we went back to 1970s development. We did code generation from the command line. Errors were reported by their line numbers. We debugged by looking at the results. It was awful! I was incredulous. Over the first few years, we purchased a number of integrated development tools and tried some open-source tools but found they were totally inadequate. They were slow. They were buggy. They didn't work. They were nonintuitive. Where were the memory-leak detectors? Where were the code-coverage tools? Where were the profiling tools? We had products to create, so we went back to the 1970s development paradigm with the exception of our editors. We compiled from the command line. We debugged by adding instrumentation (`printf` statements to the console). Lather, rinse, repeat.

HOW DO YOU SPELL RELIEF?

Thankfully, over the past five years, this has changed for the better. I want to talk about two tools that are currently available. If you tried to develop embedded Linux systems more than four years ago and gave up in disgust because of the inadequacy of the tools, it is time to take another look. I will describe two open-source IDEs we use in our company. There are many, many more. That's why we call this column "Embedded in Thin Slices."

ECLIPSE

Initially developed by IBM in 2001, Eclipse is a full-featured and extensible (through plug-ins) cross-platform IDE written in Java available for Windows (32 bit and 64 bit), Linux, and Mac OS X (Cocoa). With it, you can design, develop, and debug C/C++, Java, JavaScript, Perl, PHP, Python, Ruby (including Ruby on Rails), and many other languages for Linux (and other systems). In 2004, IBM released Eclipse to the open-source community and formed The Eclipse Foundation, a separate nonprofit foundation to support it. It is funded by membership dues in the foundation. Many of the major players in the software industry are members such as IBM, Oracle, Cisco, NEC, Intel, and Mentor Graphics.

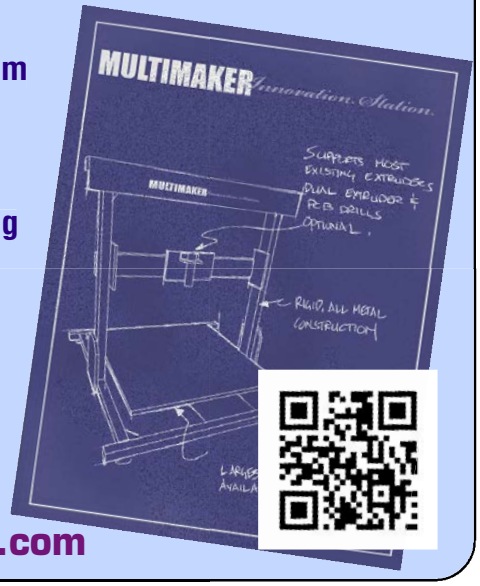
We have used Eclipse since 2007 to develop embedded Linux applications. Of all of the IDEs available for Linux development, it has by far the largest number of plug-ins to extend the functionality of the tool.

EDITOR

We are using the standard editor provided with Eclipse for almost all uses. As old-time developers, we compare everything to the old DOS-based Brief. It was the most powerful and intuitive programming editor we have ever used. In our opinion, it has all been downhill from there. After Brief's demise, we switched to CodeWrite and it was almost as good a Brief. The good news is that we have found the Eclipse editor now almost as good as CodeWrite. We still cannot do macros as easily as we did with Brief. It doesn't work well for shell script programming—but we can easily configure Eclipse to seamlessly use a different editor for different file types. On the plus side, it does an

Want to see
your idea
become a
working prototype
in hours not weeks?

Visit MultiMakerSystems.com to find out about our new Innovation Station™ and how it can help you save time and money by speeding up the innovation process right there in your office or home.



Your own low cost 3D printer, Printed Circuit Board Maker and more ....

MultiMakerSystems.com

adequate job displaying your errors following a compile, including knowing when the problem is a make file error. It handles conditional code (# defines in C) seamlessly—knowing from your project what code is functional and "graying" out the code not being used. It nicely points to where variables are used. As long as you are using a reasonably modern PC and are the only user, it runs acceptably. You can tell the Eclipse editor what your coding standard is and impose it upon all of your projects. We like the ability to handle refractoring (changing the source code throughout the whole project without affecting the executable code).

In summary, "Welcome to the 1990s Mr. Banks." The editor has almost gotten us back to what we had 20 years ago.

CODE GENERATION

All the IDEs we have used for embedded Linux use the GNU C/C++ compiler and linker for code generation. Depending on the processor for which you are developing, you can obtain the GNU cross compiler that meets your needs. Having used C compilers for more than 30 years, we have found fewer bugs in this compiler than all our previous compilers. As with most general-purpose IDEs, Eclipse enables you to define your toolchain on a per-project basis. One of the big advantages of Eclipse in the embedded world is the fact that the Yocto Project is building plug-ins for Eclipse to better integrate with OpenEmbedded. When completed, the ease of development and managing embedded Linux systems should be greatly increased.

DEBUGGING

Support for JTAG- and Ethernet-based target debugging has been around for a long time. You can launch the code on a target using the GNU debugger over an Ethernet port and debug your application or, using a JTAG device, debug your kernel driver changes. Don't have an Ethernet port? Use a USB-to-Ethernet adapter or serial-to-Ethernet adapter for connectivity. Many of the JTAG manufacturers provide Eclipse plug-ins for their devices. In addition, Eclipse is doing a better job handling the debugging of threads. Even TurboC and Visual Studio never did that very well.

OTHER TOOLS

Eclipse provides good integration with source-control tools like subversion and Git. Even more important, independent of source control, it transparently manages your versions of code. For memory-leak detection, we have successfully integrated with Dmalloc but have not been able to make it seamless with Eclipse. This is often the problem with any software package as large and as flexible as Eclipse. The learning curve can be steep.

WISH LIST

We would like to see a plug-in that would give us better visibility into the vast array of information Linux provides. For example, we would like to be able to monitor queues and semaphores at a higher level of abstraction. We would like it not to crash as much. We would like to see it run faster without having to liquid cool my PC. We wish that the learning curve wasn't so steep. Finally, isn't there some way they could keep the interface the same? It seems to be always changing. If you need to upgrade to use a new plug-in or to fix a bug, you feel like someone has completely rearranged all of the rooms, closets, and drawers in your home.

NetBeans

NetBeans originated as a Java IDE but has been expanded to become an IDE for PHP, C/C++, and JavaScript. MicroTools originally started using it in 2007 as an IDE for Java development on an embedded Linux system. Developed as a student project in 1996 by Sun Microsystems and released as open source in 2000, it initially was funded by Sun but is now primarily supported by Oracle.

Overall, NetBeans offers all that Eclipse offers. All of the pros and all of the cons are summarized in one word: ditto.

NetBeans manages multiple projects a little bit better than Eclipse. One single-board computer manufacturer is moving away from Eclipse because of that very reason (but they are moving to Code::Block with which we have little experience). NetBeans is a little faster on the same machine. On the downside, it is not tied into OpenEmbedded or the Yocto Project.

TAKE YOUR PICK

We have developers at MicroTools who are satisfied with both. Eclipse has more big-name software companies behind it as members than NetBeans. Eclipse has many more plug-in than NetBeans. That means there will be more bugs. (Who can test every plug-in's interaction with every other plug-in?) There are more JTAG interfaces available for Eclipse than for NetBeans.

The bottom line is that the development tools for Linux are now almost as good as what we had in the 1990s but much better than what we had in the 1960s and 1970s.

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.

RESOURCES

Code::Blocks, www.codeblocks.org.

Dmalloc - Debug Malloc Library, http://dmalloc.com.

The Eclipse Foundation, www.eclipse.org.

Git, http://git-scm.com.

NetBeans, "A Brief History of NetBeans," http://netbeans.org/about/history.html.

Subversion, http://subversion.apache.org.

Yocto Project, www.yoctoproject.org.

Elektor Print

Classical reading: on paper



+

Elektor Digital

Contemporary & innovative reading: on PC, notebook or tablet

=

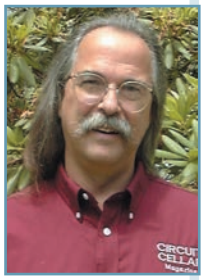
Elektor PLUS

Comprehensive reading: at home and on the road

Read Elektor with the premium PLUS subscription!

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

by Jeff Bachiochi (USA)



SNAP to It (Part 1)

Mesh Networking Simplified

Mesh networking relies on nodes to capture, distribute, and reproduce data. It is the glue that keeps the Internet and cellular towers connected. While there are many standard protocols available for localized networks (e.g., Wi-Fi, Bluetooth, or wireless USB), this article focuses on the SNAP network operating system created by Synapse Wireless.

The telephone has all but eliminated letter writing, the long-time de facto standard for keeping in touch with friends and families. Today, not only has this convenience gone wireless, but it is being used for far more than staying in contact with one another. Parents arm their kids with cell phones for peace of mind. Kids have a texting agility that is inherited from their video gaming skills. It's a sad commentary that there are far fewer "minutes" devoted to cell conversations than the senseless plethora of audio and video candy available.

When our Boy scout troop goes camping, we encourage the boys to leave all of their electronics at home. When traveling any distance, we use walkie-talkies for communication between vehicles, even though the adults have cell phones. The boys share a responsibility to keep tabs on every vehicle in the group and make sure no one takes an unexpected turn. (It amazes me how each GPS will calculate certain routes differently.) This convoy-style string can sometimes exceed the distance of a walkie-talkie, making the middle vehicles the only link between the front runner and the rear sweeper. Communication must be relayed (repeated) by one radio that has contact with those ahead and behind. The number of repeats or hops is related to how much the vehicles are spread out and the range of the radios.

So it is with low-power communications. Since transmission distance is a function of power output, low-power radios have a rather limited range. This can be a good thing if you wish to limit the interference you cause. It can be a bad thing if you need to span a considerable distance within a group of radios. When a radio must not only transmit and receive its own data, but also relay the data of others, it is being used as part of a mesh network. As one may expect, the strategies for this may be as simple as every radio repeating what it has heard, or as sophisticated as dynamic mapping of the most efficient route. The simplest strategies can create a throughput clogging uproar of activity as each receives and retransmits the same message. So, at least a little bit of smarts are required to cut this to an acceptable level.

MESH NETWORKS

You can think of the Internet as the world's largest wired mesh network. Information travels from one router (repeater/director) to the next until it reaches its destination. Your IP

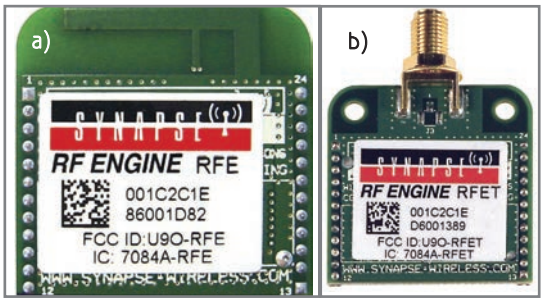


Photo 1—SNAP engine modules are available with an integrated antenna (**a**) and an RP-SMA connector for using an external antenna (**b**)

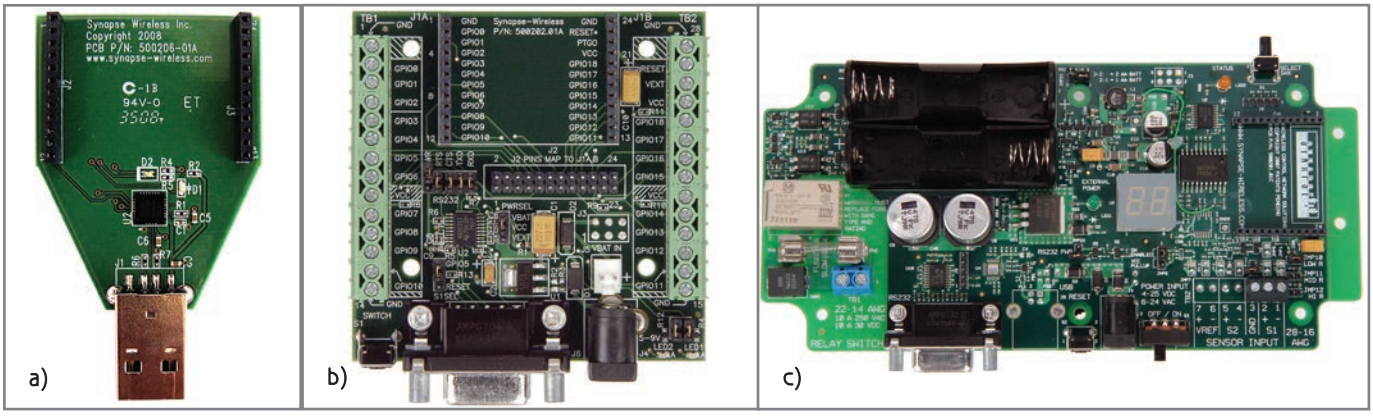


Photo 2—SNAP evaluation boards that accept all SNAP engine modules include the USB bridge module (**a**), the Prototype1 PCB (**b**), and the Prototype 2 PCB (**c**).

address changes when you disconnect your laptop and reconnect at some other location. (The IP is fixed to a physical connection.) To remain integrated in the mesh, when you move, you must reestablish your new location with others, either directly or through the grapevine.

On a smaller scale, if you work with microcontroller devices, you may be familiar with ZigBee mesh networks. ZigBee was conceived as the first low-power ad-hoc protocol in 2003, with the latest incarnation being ZigBee PRO. It should be noted that the ZigBee Alliance has recognized the need for simpler networks that remain within earshot (no repeaters) and has introduced ZigBee Radio Frequency for Consumer Electronics (RF4CE). A typical ZigBee network may consist of three types of devices, a ZigBee Controller (ZC), a ZigBee Router (ZR), and a ZigBee End Device (ZED). A single ZC originates and coordinates the network keeping track of all active nodes. ZRs can pass communication as a repeater and act as an endpoint for service, while ZEDs are strictly endpoints.

There are other protocols available you should consider when looking for something to use with your new widget. You may want to consider Wi-Fi, Bluetooth, or wireless USB as capable candidates depending on the application. Or, read on further as I take you through the SNAP network operating system (OS) created by Synapse Wireless.

SYNAPSE WIRELESS

Not to be confused with USNAP (see my article, "Smart Network Access Point," *Circuit Cellar* 246, 2011), Synapse Wireless has created a family of products around the SNAP OS, which is a communication and control protocol used to create and maintain a mesh network of SNAP nodes. SNAP nodes contain a SNAP device (a microcontroller executing the SNAP OS) that interfaces with communication hardware (i.e., radio, I/O). A SNAP device can be integrated directly into your product, but is also available as part of a SNAP engine. The SNAP engine contains a SNAP device and radio prepackaged as a standard footprint module, providing a common development interface (see Photo 1).

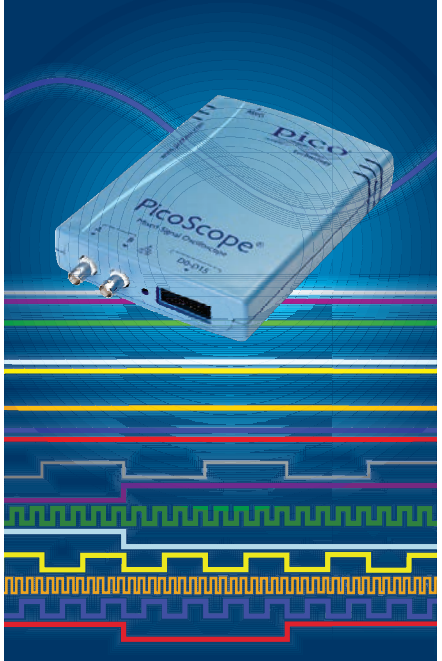
A SNAP bridge is a SNAP node used to connect two networks such as a SNAP network and the Internet (SNAP Connect E10), a PC, or a second SNAP network. For instance, the SNAP Connect E10 is an end product incorporating a SNAP node as a bridge between a

ID 0	Reserved for Synapse use	ID 18	Collision avoidance	ID 40	SNAPPy CRC
ID 1	Reserved for Synapse use	ID 19	Radio unicast retries	ID 41	Platform
ID 2	MAC Address	ID 20	Mesh routing maximum timeout	ID 42–49	Reserved for future use
ID 3	Network ID	ID 21	Mesh routing minimum timeout	ID 50	Enable encryption
ID 4	Channel	ID 22	Mesh routing new timeout	ID 51	Encryption key
ID 5	Multicast processed groups	ID 23	Mesh routing used timeout	ID 52	Lockdown
ID 6	Multicast forwarded groups	ID 24	Mesh routing delete timeout	ID 53	Maximum loyalty
ID 7	Manufacturing date	ID 25	Mesh routing RREQ retries	ID 54–59	Reserved for future use
ID 8	Device name	ID 26	Mesh routing RREQ wait time	ID 60	Last version booted *(deprecated)*
ID 9	Last system error	ID 27	Mesh routing initial hop limit	ID 61	Reboots remaining
ID 10	Device type	ID 28	Mesh routing maximum hop limit	ID 62	Reserved for future use
ID 11	Feature bits	ID 29	Mesh sequence number	ID 63	Alternate radio trim value
ID 12	Default UART	ID 30	Mesh override	ID 64	Vendor-specific settings
ID 13	Buffering timeout	ID 31	Mesh routing LQ threshold	ID 65	Clock regulator
ID 14	Buffering threshold	ID 32	Mesh rejection LQ threshold	ID 66–127	Reserved for future use
ID 15	Intercharacter timeout	ID 33	Noise floor	ID 70	Transmit power limit
ID 16	Carrier sense	ID 34–38	Reserved for future use	ID 128–254	Available for user definition
ID 17	Collision detect	ID 39	Radio LQ threshold	ID 255	Reserved for Synapse use

Table 1—Nonvolatile memory is set aside for internal parameters that support function configuration. Additional slots are reserved for the user.

THE NEW PICOSCOPE
2205 MSO
MIXED SIGNAL OSCILLOSCOPE

GREAT VALUE, PORTABLE, HIGH END FEATURES AS STANDARD AND EASY TO USE



Think Logically...

pico Technology

Channels	2 Analog, 16 Digital
Resolution	8 bit
Bandwidth Digital frequency	Analog 25 MHz, Digital 100MHz combined
Sampling rate	200MS/s
Trigger modes	Edge, Window, Pulse width, Window pulse width, Dropout, Window dropout, Interval, Runt pulse, Digital, Logic
Price	$575

www.picotech.com/pco467
1-800-591-2796

```
Datamode.py

from synapse.switchboard import *

otherNodeAddr = "\x00\x00\xFF" # <= put the address of the OTHER node here

@setHook(HOOK_STARTUP)
def startupEvent():
      initUart(1, 9600) # <= put your desired baudrate here!
      flowControl(1, False) # <= set flow control to True or False as needed
      crossConnect(DS_UART1, DS_TRANSPARENT)
      ucastSerial(otherNodeAddr)
```

Figure 1—This simple script can be loaded into two nodes enabling them to act as a wireless serial link between two RS-232 devices.

ADC	
readAdc(channel)	Sample ADC on specified input channel, returns raw reading
CBUS Master Emulation	
cbusRd(numToRead)	Reads numToRead bytes from CBUS, returns string
cbusWr(byteStr)	Writes every byte in byteStr to the CBUS
GPIO	
setPinDir(pin, isOutput)	Set direction for parallel I/O pin
setPinPullup(pin, isEnabled)	Enable pull-up resistor for Input pin
setPinSlew(pin, isRateCon-	Enable slew rate control for Output pin
monitorPin(pin, isMonitored)	Enable GPIN events on Input pin
pulsePin(pin, msWidth, isPositive)	Apply pulse to Output pin
readPin(pin)	Read current level of pin
writePin(pin, isHigh)	Set Output pin level
setRate(rateCode)	Set pin sampling rate to off (0), 100 ms (1), 10 ms (2), or 1 ms (3)
I²C Master Emulation	
getI2cResult()	Returns the result of the most recent I²C operation
i2cInit(enablePullups)	Prepare for I²C operations
i2cRead(str, numBytes, retries, ignoreFirstAck)	Write str out, then read numBytes back in from I²C bus. Parameters retries and ignoreFirstAck are used with slow or special case devices
i2cWrite(str, retries, ignore-FirstAck)	Write str out over the I²C bus. Parameters retries and ignoreFirstAck are used with slow or special case devices
Miscellaneous	
setSegments(segments)	Set eval board LED segments (clockwise bitmask)
bist()	Built-in self test
eraseImage()	Erase user-application flash memory
resetVm()	Reset the embedded virtual machine (prep for upload)
initVm()	Initialize embedded virtual machine
vmStat(statusCode, args...)	Solicit a tellVmStat for system parameters
writeChunk(ofs, str)	Write string to user-application flash memory
chr(number)	Returns the character string representation of "number"
str(obj)	Returns the string representation of obj
int(obj)	Returns the integer representation of obj. Notice that you cannot specify the base. Decimal is assumed.
len(str)	Returns the length of string str (0–255)
random()	Returns a pseudo-random number 0–4,095
stdinMode(mode, echo)	Mode is 0 for line, 1 for character at a time

Table 2 continued on p. 71

Table 2—The built-in SNAPPy API functions available grouped together by the category.

Network	
getNetId()	Current network ID
setNetId(netId)	Set network ID (1–0xFFFE)
localAddr()	Local network address (3-byte binary string)
rpcSourceAddr()	Originating address of current RPC context (none if called outside RPC)
mcastSerial(dstGroups, ttl)	Set serial transparent mode to multicast
ucastSerial(dstAddr)	Set serial transparent mode to unicast
callback(callbackFnObj, remoteFnObj, args...)	RPC (back to the original invoker) of RPC results
callout(addr, callbackFnObj, remoteFnObj, args...)	RPC (to an arbitrary node address) of RPC results
rpc(dstAddr, remoteFnObj, args...)	RPC (unicast)
mcastRpc(dstGroups, ttl, remoteFnObj, args...)	RPC (multicast)
	Nonvolatile (NV) Parameters
loadNvParam(id)	Load indexed parameter from NV storage
saveNvParam(id, obj)	Save object to indexed NV storage location
	Radio
rx(isEnabled)	Enable/disable radio receiver
txPwr(power)	Adjust radio transmit level (0 is lowest, 17 is highest)
setChannel(channel)	Set radio channel
getChannel()	Radio channel
getLq()	Link quality in (–) dBm
getEnergy()	Detected RF energy in (–) dBm (current channel)
scanEnergy()	Detected RF energy in (–) dBm (all 16 channels)
peekRadio(addr)	Read a memory location from inside the radio
pokeRadio(addr, byteVal)	Write a memory location inside the radio

Table 2 continued from p. 70, continues on p. 72

SNAP network and a Linux environment running SNAP Connect. SNAP Connect is a licensed API that enables client applications to access the SNAP network using XML-RPC. XML-RPC is a standard communication protocol utilizing hypertext transfer protocol (HTTP) to transport remote procedure calls (RPCs) encoded using eXtensible Markup Language (XML).

SNAP Portal is a free PC application that turns your PC into a SNAP node enabling it to be part of the SNAP network through a bridge node connected via a serial or USB connection. I used a SNAP stick, a USB SNAP node dongle (see Photo 2), as this bridge. SNAP Portal is an incredible application tool that enables you to develop, download, and interact with every node on the SNAP network. Wait, if SNAP devices already handle all the networking tasks for me, what am I developing? While it's true you don't have to worry about the network stuff, you will most likely want to gather, disseminate, and make use of data. This requires application programming specific to your task at hand.

Mesh Networking, Simplified!



Experience the Fastest Growing Mesh Network Technology– SNAP® from Synapse Wireless.

Easiest platform for rapid prototyping and development.
Over-the-air programming with free IDE called *Portal.*
Batteries included for remote control applications.
Get ready for a world of Smart Machines.
Get SNAP!

SNAP Network Starter kit
Order yours today! MSRP $99.00
synapse-wireless.com/kit

Risk Free:
30-day money
back guarantee.

SYNAPSE

Wireless Technology to Control and Monitor Anything from Anywhere™

© 2007-2012 Synapse Wireless Inc. All Rights Reserved. All Synapse products are patent pending.
500 Discovery Drive, Huntsville, Alabama · 877-982-7888

One of the most useful applications for a node is to become a SNAP sniffer. When a node is programmed as a SNAP sniffer, it monitors all network traffic and, when connected to a PC running the free sniffer application, you can see and log the data for debug purposes.

The SNAP OS handles all the networking tasks and provides the user with a Python-like virtual machine within each SNAP device. Your applications are written in this script language called SNAPPy (a subset of Python.) SNAPPy scripts are text files that are compiled into SNAPPy images to save space. You can create a single SNAPPy script for all nodes or create specialized scripts for each node. Before I get into the SNAPPy scripting language, let's look at mesh networks in general and see how SNAP handles different situations.

A MISHMASH OF MESHING

One feature of SNAP networks is that by design the protocol doesn't require any coordinator to keep track of everything. Each SNAP node can automatically retransmit a request to help reach nodes beyond the range of the sender. If every node was to do this, there could be a lot of unnecessary chatter. Therefore, it is important that SNAP recognizes who is not only within range, but who has the best shot at communicating with a neighbor (signal strength).

SNAPPy script functions that send data include parameters that adjust how a message is sent and retransmitted. These include the ability to isolate certain nodes by forming subgroups and defining how many times a message can be retransmitted.

Many functions are based on a configured parameter. SNAP uses nonvolatile memory to store these configurations between power cycles and resets. About half of the allocated space for these parameters is available for user-defined parameters (see Table 1). Network parameters can be adjusted to optimize how the nodes react, depending on your network's physical configuration.

SPI Master Emulation	
spiInit(cpol, cpha, isMsbFirst, isFourWire)	Setup for SPI with specified clock polarity, clock phase, bit order, and physical interface
spiRead(byteCount, bitsInLastByte)	Receive data in from SPI—returns response string (three-wire SPI only)
spiWrite(byteStr, bitsInLastByte)	Send data out SPI—bitsInLastByte defaults to 8, can be less
spiXfer(byteStr, bitsInLastByte)	Bidirectional SPI transfer—returns response string (four-wire SPI only)
Switchboard	
crossConnect(dataSrc1, dataSrc2)	Cross-connect SNAP datasources
uniConnect(dst, src)	Connect src->dst SNAP datasources
System	
getMs()	System ms tick (16 bit)
getInfo(which)	Get specified system information
getStat(which)	Get radio traffic status information
call()	Invoke a user-defined binary function
peek(addr)	Read a memory location
poke(addr, byteVal)	Write a memory location
errno()	Read and reset last error code
imageName()	Name of current SNAPPy image
random()	Returns a random number (0–4095)
reboot()	Reboot the device
sleep(mode, ticks)	Enter Sleep mode for specified number of ticks resolution, accuracy, and maximum duration vary between hardware platforms. For example, on an RF100 SNAP engine: In mode 0, ticks are 1.024 s each, ±30%. In mode 1, ticks are 1 s each, and can be 0–1073. On some platforms, negative values for ticks produce times shorter than 1 s.
UARTs	
initUart(uartNum, bps)	Enable UART at specified rate (zero rate to disable)
initUart(uartNum, bps, dataBits, parity, stop)	Enable UART at specified rate (zero rate to disable), data bits, parity, and stop bits
flowControl(uartNum, isEnabled)	Enable RTS/CTS flow control. If enabled, the CTS pin functions as a "Clear to Send" indicator
flowControl(uartNum, isEnabled, isTxEnable)	Enable RTS/CTS flow control. If enabled and parameter isTxEnable is True, then the CTS pin functions as a transmit enable (TXENA) signal. If enabled and isTxEnable is False, then the CTS pin functions as a "Clear to Send" indicator
	Table 2, continued from p. 71

SNAPPy

Table 2 shows the built-in functions that each node has available for use within SNAPPy scripts that a node will execute. The best way to learn how these are used is to look at the demos that are available for use. Many of these were designed for use with the SNAP prototyping or demo boards that form the basis of evaluation kits. These boards will accept the complete line of SNAP engines that plug in with the standard footprint. While two nodes form the minimum network, I will be using three nodes, one of which is the node used exclusively as a bridge from the PC to the network. A bridge node makes development much easier because you do not have to plug each node into the PC to program your SNAPPy script into it. This can be accomplished wirelessly using the bridge node.

Let's start with a simple script that enables two nodes to act as a wireless serial connection. This is a typical application for connecting two serial devices wirelessly (without a physical connection between them). One important note here is that there must be some way for nodes to identify their counterpart. A node is uniquely identified by its node address, which is the last 3 bytes of its MAC address printed on the label attached to each device (and pre-programmed into configuration register ID2.) The script in Figure 1 is the same for each node except that the node address of its counterpart is fixed within the script that is downloaded to a node. Let's go over this example script.

The first line imports the switchboard.py script from

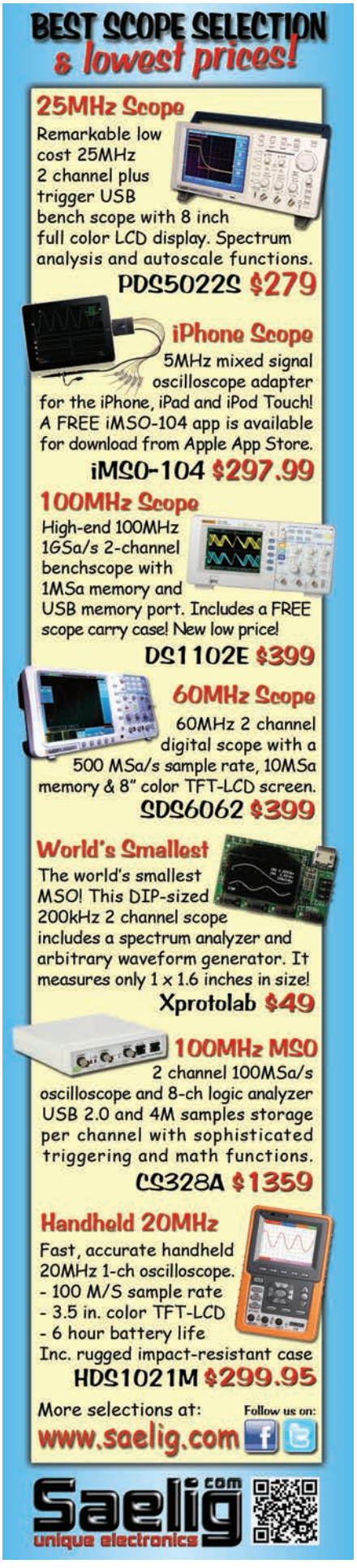
the Synapse subdirectory. This script defines each data source (DS) object that can be used by the `crossConnect()` and `uniConnect()` functions. Here `DS_UART1` and `DS_TRANSPARENT` are used. The second line defines the other node's address, which is used as the ucast destination for all outgoing communication. The third line indicates the beginning of an event, indicated by the `@setHook()` function. See Table 3 for a list of the available event hooks. This event is called upon reset and will run once. The user-defined function `startupEvent()` is all that is needed to complete this application. This function initializes UART1 for 9600 bps (note the databits, parity, and stopbits are not explicitly defined and so they default to 8, N, 1). Next the flow control for UART1 is disabled (CTS1 and RTS1 pins can be used as additional I/O). The `crossConnect()` function ties both inputs and outputs of one object to another object (here `DS_UART1` and `DS_TRANSPARENT` are connected). All data received by UART1 is routed to the radio's transmitter and all data coming into the radio's receiver is routed to UART1's transmitter using transparent mode. Each radio can handle data as "transparent" or serial-packet (which is how RPCs are used). Finally, the `ucastSerial()` function limits transmissions to a single node. `UcastSerial()` has the advantage of being acknowledged by the recipient. An unacknowledged transmission will be resent based on the NV configuration ID 19 (Radio Unicast Retries). Assuming the edited script is loaded into two nodes, which have RS-232 level shifters, you can connect the nodes to serial devices and you're good to go. (Note: You could connect to the SNAP engine directly and use the TTL-level serial signals, where RS-232 levels are not needed.)

Since you have a number of NV ID reserved for your use, you may want to define one of these as the `otherNodesAddr`. This would enable the script to read this NV ID and use the address stored there as its counterpart. Why bother doing this? The ability to use a nonvolatile value enables it to be changed without having to rewrite the script using a hard coded address each time you want to pair up a set of nodes, as in the first example. Wouldn't it make sense to be able to pair SNAP nodes dynamically?

There are two ways of transmitting data. The first was just discussed, `unicastSerial()`, the second is `multicastSerial()`. When you send data using `multicastSerial()` it is sent as an unacknowledged message to all nodes. You can think of it as a general shout out. We've already seen that the user can define a function, as in the `startupEvent()` of the first example.

Suppose I defined a `checkForNodes()` function consisting of `mcastRpc(1, 1, "whoIsOutThere")`. This RPC would be multicast to all nodes (in group1 (all), using 1 hop, looking to invoke the function `whoIsOutThere()` on all those nodes listening. If the `whoIsOutThere()` function is defined consisting of the RPC function, `rpc(rpcSourceAddr(), "iAm()")`, then this enables the node to respond to the one who originated the request with a unicast RPC requesting to invode function "`iAm()`. The originator's function "`iAm()` consisted of `otherNodesAddr = rpc(SourceAddr())`, the variable `otherNodesAddr` would then hold the node ID of the first responding node. Finally, if the originator unicast back the `rpc(rpcSourceAddr(), "iAm()")` function, the responder could complete transaction knowing it was chosen as the pairing node. While this description is a bit simplistic, I hope you can see how one node may discover another node and swap node addresses by a mix of multicast and unicast transmissions.

Here are a few things you must think about for this to work. Assuming all nodes are using the same script with all of the above functions, there must be a way of determining when these functions should be allowed. What if there are more than two nodes out there? If a paired set of nodes has already saved the NV ID defined as `otherNodesAddr`, you could skip all of this pairing. If a `findNewNodeAddress` push button is implemented using `@setHook(HOOK_GPIN)`, then a function might clear the NV ID `otherNodesAddr`. Any nodes with an erased address would be



Hook Name	When Invoked	Parameters	Sample Signature
HOOK_STARTUP	Called on device bootup	HOOK_STARTUP passes no parameters.	setHook(HOOK_STARTUP) def onBoot() : pass
HOOK_GPIN	Called on transition of a monitored hardware pin	**pinNum** – The pin number of the pin that has transitioned.1 isSet – A Boolean value indicating whether the pin is set.	setHook(HOOK_GPIN) def pinChg(pinNum, isSet) : pass
HOOK_1MS	Called every millisecond	**tick** – A rolling 16-bit integer incremented every millisecond indicating the current count on the internal clock. The same counter is used for all four timing hooks.	setHook(HOOK_1MS) def doEvery1ms(tick) : pass
HOOK_10MS	Called every 10 ms	**tick** – A rolling 16-bit integer incremented every millisecond indicating the current count on the internal clock. The same counter is used for all four timing hooks.	setHook(HOOK_10MS) def doEvery10ms(tick) : pass
HOOK_100MS	Called every 100 ms	**tick** – A rolling 16-bit integer incremented every millisecond indicating the current count on the internal clock. The same counter is used for all four timing hooks.	setHook(HOOK_100MS) def doEvery100ms(tick) : pass
HOOK_1S	Called every second	**tick** – A rolling 16-bit integer incremented every millisecond indicating the current count on the internal clock. The same counter is used for all four timing hooks.	setHook(HOOK_1S) def doEverySec(tick) : pass
HOOK_STDIN	Called when "user input" data is received	**data** – A data buffer containing one or more received characters.	setHook(HOOK_STDIN) def getInput(data) : pass
HOOK_STDOUT	Called when "user output" data is sent	**HOOK_STDOUT** passes no parameters.	setHook(HOOK_STDOUT) def printed() : pass
HOOK_RPC_SENT	Called when the buffer for an outgoing RPC call is cleared	**bufRef** – an integer reference to the packet that the RPC call attempted to send. This integer will correspond to the value returned from getInfo(9) when called immediately after an RPC call is made. The receipt of a value from HOOK_RPC_SENT does not necessarily indicate that the packet was sent and received successfully. It is an indication that SNAP has completed processing the packet.	setHook(HOOK_RPC_SENT) def rpcDone (bufRef) : pass

Table 3—SNAP is an event-driven language. These functions are built-in events that a SNAPPy script can use within its application.

looking for a pairing request by enabling the whoIsOutThere() function. A second push button, "iAmLooking" could invoke the whoIsOutThere() function using @setHook(HOOK_GPIN). The first responder would get paired.

Another approach might be to use the node's serial connection and a terminal program. A command-line interpreter script would enable you to request and collect responding node addresses, display, and choose from among the available nodes. How does a command-line interpreter relate to a node that is being used as a transparent serial link? The transparent application just shuffles data from endpoint to endpoint (the UART at each end of the wireless link). A command-line interpreter will recognize specific data to initiate some event. In the crossConnect(DS_UART1, DS_TRANSPARENT) of the first example, there is no way for any of the data passing through to be analyzed. If you use crossConnect(DS_UART1, DS_STDIO) then you are connecting user I/O to UART1. Data received by the UART is available through the event @setHook(HOOK_STDOUT). Any function can send data to the UART for transmission via the print statement. Data is available as characters (Mode=1) or carriage return (CR) terminated lines (Mode=0) using stdin Mode(Mode, Echo). Echo=1 will resend the characters back to the sender. With this setup you can monitor the data and look for a specific string (like the "~~~" used in systems to provide an "AT" command set) to trigger a redirection of data. This might be the checkForNodes() function described earlier, after which data would again become "transparent."

If you followed this, you may be thinking transparent with what? There is no connection now to DS_TRANSPARENT and the radio? That is correct and it is illegal to connect a source, like the UART1 with two destinations like DS_STDIO and DS_TRANSPARENT. However, we can use the Rpc(address, function, arguments) statement to request the execution of a user function which could pass any data as the argument of that function. On the far end, that function can perform any necessary dissection of the data or merely use the print statement to pass it onto the UART. In this case, the UART and the STDIO objects are cross connected and the radio "connection" is made through RPC functions of the SNAPPy API.

LOOK MOM, NO HANDS

Most of the discussion so far has been on wirelessly

connecting two UART objects. It's an activity that most can relate to. What happens when the object is not a UART? The talk of connecting the UART object to the TRANSPARENT object makes a good mental picture, however there are other objects that don't require a connection at all. These are all SNAPPy API functions and can be used at any time. Referring back to Table 1 you can see that the scripts can contain functions that communicate with external CBUS, SPI, and I²C slave devices physically attached to predefined I/O pins. Internally, there are GPIO, digital I/O pins for logic monitoring and control, as well as an analog-to-digital converter (ADC), for converting analog inputs. Let's consider an example that uses only GPIO.

The simplest example I can think of might be a wireless doorbell. One node has a push button connected to a digital input pin and the other node has a piezo element connected to a digital output pin. If the push-button event uses a multicast function like `mcastRcp(group, ttl, function, arguments)`, then it doesn't even need to know where it is sending the message. The function will execute on any node that hears it. If the second node has its piezo element connected to the GPIO associated with pulse-width modulation (PWM), then the script can simply beep or, if you're clever, it can play a little ditty. You could simply add more push-button nodes for the other doors. In fact, each door button could request a different tone pattern.

To eliminate having to program each button node with a different script (requesting a different "ring tone") you can do this by using multiple inputs. You might create multiple button inputs, with each input associated with a particular value. Or, you might assign one input for the button and other inputs to configuration jumpers that the script can read as a selected value. In either case, the value is sent as the function's argument, defining which bell has been requested.

It can be noted that additional bell nodes could be placed in every room of the house all happy to execute their annunciation when requested by any door. Suppose the piezo element and associated transistor driver were not installed on the node? The SNAP device doesn't care, in fact it doesn't even know! So we could do additional things on this node like flash a light on and off, simply by using an unused I/O pin to drive any support circuitry needed for the task and include the script statements necessary to support that task. This way any bell node could have a different arrangement of circuitry and still use the same node script.

SPREADING THE WEALTH

Perhaps you can begin to see the possibilities with this type of network where everyone is created equal. Every node has the potential to act on data received from any other node in the network. Imagine your spouse complaining that the room lighting is a safety hazard at night. You could jump onto the network and tell the reading lamp to pay attention to the hall light switch and *voila*, an extra light now goes on and off with the hall switch.

I looked briefly at using a command-line interpreter with a SNAP node, which can give your PC applications the ability to interact with your scripts via a serial connection. Optionally, the licensed SNAP connect server application will enable your applications (written in almost any language) to directly invoke functions on any SNAP node, even over the Internet, using XML-RPC.

The term machine-to-machine (M2M) is becoming a popular phrase today as it refers to the technologies that enable systems to communicate with one another. You know things are getting serious by the number of standards committees that have been formed. This just may be one market where we can flex our technological muscles. And, if security is an issue for you, each SNAP device has a Basic encryption algorithm built in. AES-128 encryption is also available on most devices by simply downloading a special version of the SNAP OS. It is good programming practice to be confident in your application before turning on any encryption, as debugging and use of the sniffer will be all but futile.

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

RESOURCES

BiTXml, www.bitxml.org.

Connected World magazine, www.connectedworldmag.com.

M2M magazine, www.machinetomachinemagazine.com.

M2MXML, m2mxml.sourceforge.net.

Synapse Wireless, Inc., "SNAP Network Operating System Reference Manual for Version 2.4," 2011, http://forums.synapse-wireless.com/upload/SNAP%20Reference%20Manual.pdf.

———, "Technical Manual SNAP Hardware Document Revision v1.3," 2010, http://forums.synapse-wireless.com/upload/SNAP%20Hardware%20Technical%20Manual.pdf.

Telecommunications Industry Association, "TR-50 Smart Device Communications," www.tiaonline.org/standards/committees/committee.cfm?comm=tr-50.

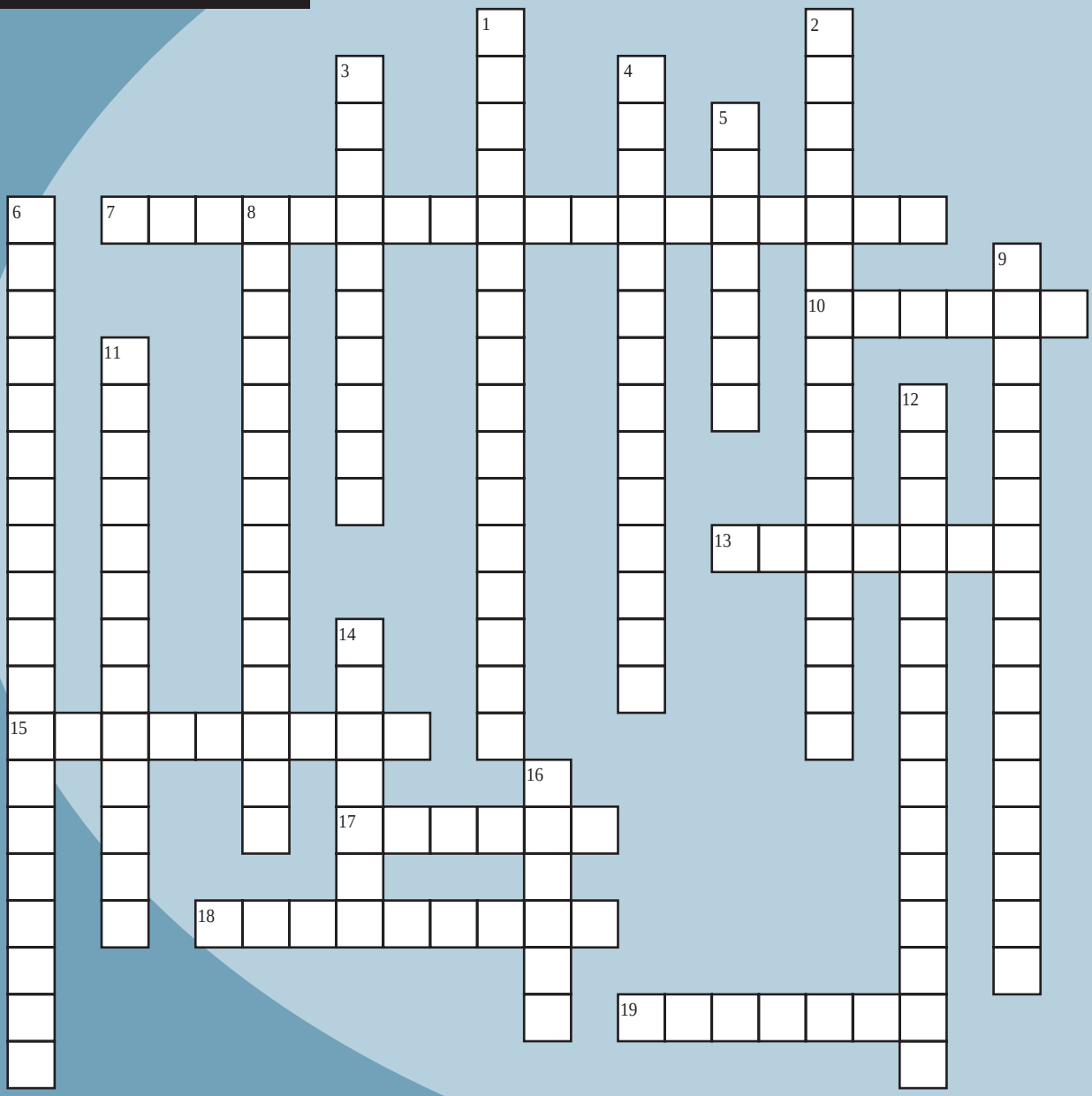
———, "M2M Standardization Task Force (MSTF)," www.tiaonline.org/standards/mstf/index.cfm.

ZigBee Alliance, "ZigBee RF4CE Overview," 2009, www.zigbee.org/Specifications/ZigBeeRF4CE/Overview.aspx.

SOURCE

SNAP Connect E10 embedded Linux computer
Synapse Wireless, Inc. | www.synapse-wireless.com

CROSSWORD



Across

7. A form of BPSK [two words]
10. Restart
13. The # label
15. File index
17. Takes place behind the scenes
18. Open-source programming language compiler
19. Bad code

Down

1. Transports data by altering a reference signal's phase [three words]
2. Changes a signal from the time domain to the frequency domain [two words]
3. Used to measure signal loss in decibels
4. The noisy-channel coding theorem [two words]
5. Logic system named after George Boole
6. Opposite of AM [two words]
8. Measures battery current [two words]
9. 4G [two words]
11. 25th percentile [two words]
12. Converts a current to a voltage [two words]
14. Four switching components with the load in the center
16. A way to organize

The answers will be available in the next issue.

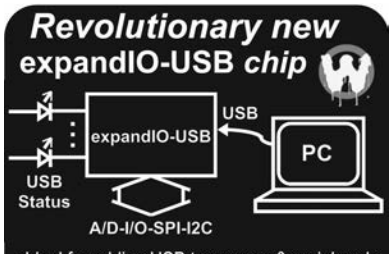
IDEA BOX

THE DIRECTORY OF PRODUCTS AND SERVICES

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

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

Revolutionary new expandIO-USB chip

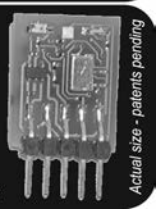


- Ideal for adding USB to sensors & peripherals
- No drivers needed for Windows, Mac, Linux
- No microcontroller programming required
- Also check out our USB-232 USB to UART

www.hexwax.com - Buy from Mouser & Farnell

Amazing PIC programmer

Most devices supported
ICSP, SQTP, & copy limits



$32 at Digikey & Mouser

www.flexipanel.com

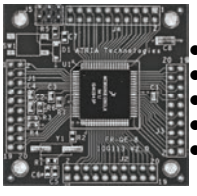
NEW BUD Board

BUD is a compact Atmel ATmega640 microcontroller board with a big extra: when a BOB-4 module is installed, BUD easily generates complex text and graphics on TV monitors. Example software allows first-time users to immediately exercise the combined functionality of BUD and BOB-4 with programs such as GPS Data Display, TV Typewriter, and Real Time Clock Display. Introductory pricing through June 2012!



www.decadenet.com

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



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

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

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

Kits Available

Learn about BASIC ON BOARD

ATRIA *Technologies Inc.*
www.AtriaTechnologies.com

i2cStick™



Discover I²C
Read, Write, Connect...

MCC Micro Computer Control

www.mcc-us.com

B-Engine (BE)™

$69 OEM
$99 Qty 1

- *OEM CPU board with high speed ±10V bipolar 16-bit ADC (AD7606), DACs*
- *100M Base-T Ethernet, CompactFlash*
- *3.6" x 2.3", C/C++ programmable.*



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

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

ANALOG and DIGITAL for PC/104



GPIO-104
8 ANALOG INPUTS
4 ANALOG OUTPUTS
24 DIGITAL I/O

Need More Analog I/O?



ADIO-104
16 ANALOG INPUTS
8 ANALOG OUTPUTS
24 DIGITAL I/O
PULSE ACCUMULATOR

SCIDYNE®
Also Available
•96 DIGITAL I/O
•24 HIGH POWER I/O
•RELAY/OPTO I/O
•EXTERNAL EXPANSION
•PC/104 PROTOTYPING

1-877-SCIDYNE
info@scidyne.com
www.scidyne.com



Add a Touch Screen to Your Embedded Product

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

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

REACH TECHNOLOGY INC.

See us in Booth 1441 at Design West (ESC) in San Jose.

Giga-snaP BGA Sockets and Adapters

Giga-snaP BGA SMT Adapters allow affordable socketing

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



RoHS

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

ALL ELECTRONICS CORPORATION

Electronic and Electro-mechanical Devices, Parts and Supplies. Wall Transformers, Alarms, Fuses, Relays, Opto Electronics, Knobs, Video Accessories, Sirens, Solder Accessories, Motors, Heat Sinks, Terminal Strips, L.E.D.S., Displays, Fans, Solar Cells, Buzzers, Batteries, Magnets, Cameras, Panel Meters, Switches, Speakers, Peltier Devices, and much more....

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

Stream Data Through the PIC®MCU ICSP™ Interface

Real-Time Data Access Without Halting MCU



FREE w/ IDE Compilers version 4.129 or better
www.ccsinfo.com/ccds

• Eliminate a diagnostic serial port and save money by streaming data through the ICSP interface
• Set serial #'s and calibration values in production
• Review log data in the field

CCS Inc.
sales@ccsinfo.com
262-522-6500 ext. 35

PIC® MCU, dsPIC and MPLAB are registered tradmarks of Microchip Technology, Inc.

LISTEN TO YOUR MACHINES

Ethernet PLCs for OEMs



FMD88-10 and FMD1616-10

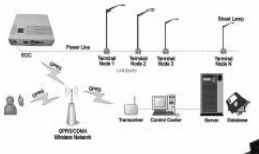
Integrated Features :
- ETHERNET / Modbus TCP/IP
- 16 or 32 digital I/Os
- 10 analog I/Os
- RS232 and RS485
- LCD Display Port
- I/O Expansion Port
- Ladder + BASIC Programming

$229 and $295
before OEM Qty Discount

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

TRI TRIANGLE RESEARCH INTERNATIONAL

Power-Line Communication, Street Lighting System, UHF EPC Gen2 RFID Reader, Arduino Shields (WiFi)



■Power-Line Communication Street Lighting System



■Power-Line Communication Modules
- Low-Speed Modem Pro
- Low-Speed Shield for Arduino – The first one in the world!!!
- High-Speed Modem
- High-Speed Modem for DMX512



■Arduino Shields: WiFi, GPRS, Charger, IR, Smart Outlet
■JPEG Serial Camera
■Cellular modules
■GPS, Bluetooth modules

■ LS6410 embedded system with Win CE 6.0, Linux, and Android.

www.linksprite.com

RF Specialists

RF Modules
GSM/GPRS
Bluetooth
Data Loggers
GPS
Wi-Fi
ZigBee Pro

LEMOS INTERNATIONAL
www.lemosint.com

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

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

THE NEXT GENERATION OF MAXSONAR®
The HRLV - MaxSonar Sensors
- Amazing One-Millimeter Resolution
- Simultaneous Multiple Sensor Operation
- Superior Noise Rejection
- Target Size Compensation for Accuracy
- Temperature Compensation ($4.95)
- Outputs now include TTL Serial
$34.95 (MSRP) www.MaxBotix.com

microEngineering Labs, Inc.
www.melabs.com 888-316-1753
Programmers for Microchip PIC® Microcontrollers
melabs ME U2 Programmer
microEngineering Labs, Inc.
www.melabs.com
Starting at $79.95
PC-Tethered USB Model (shown):
- Standalone software
- Command-line operation
- Hide GUI for automated use
- Override configuration with drop-downs
Stand-Alone Field Programmer:
- Power from target device or adapter
- Program file stored on SD-CARD
- Programming options stored in file
- Single-button operation
Program in-circuit or use adapters for unmounted chips.
Zero-Insertion-Force Adapters available for DIP, SOIC, SSOP, TQFP, and more.
PIC is a registered trademark of Microchip Technology Inc. in the USA and other countries.

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

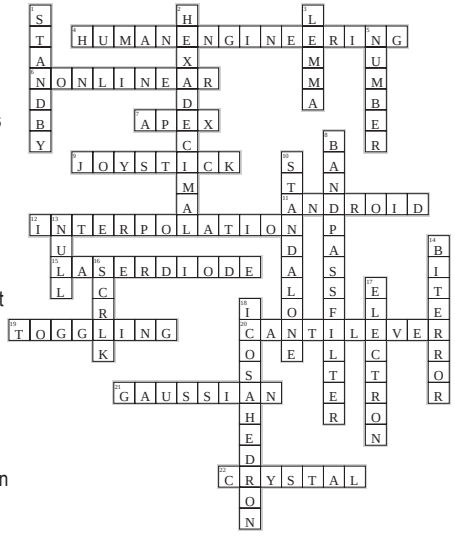
CROSSWORD ANSWERS from Issue 260

Across

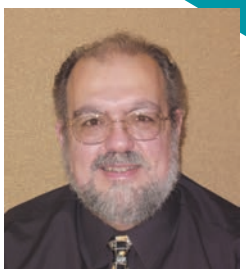
4. **HUMANENGINEERING**—Applied science of ergonomics [two words]
6. **NONLINEAR**—Output not proportional to input
7. **APEX**—It isn't aligned above the center of an oblique cone's base
9. **JOYSTICK**—A switch, control, or lever used for guidance
11. **ANDROID**—A robot with human-like qualities, or the opposite of iOS
12. **INTERPOLATION**—Value estimation
15. **LASERDIODE**—An LED emitting coherent light [two words]
19. **TOGGLING**—Switching from high to low or low to high
20. **CANTILEVER**—A beam with only one fixed end
21. **GAUSSIAN**—Filter type associated with GRSK
22. **CRYSTAL**—Natural substance capable of producing voltage

Down

1. **STANDBY**—Operating below average
2. **HEXADECIMAL**—Base of 16
3. **LEMMA**—True, but not a theorem; helps prove a theorem
5. **NUMBER**—Used in math to indicate a quantity or amount
8. **BANDPASSFILTER**—Enables only a narrow band of frequencies access to a circuit [two words]
10. **STANDALONE**—Able to function without connection
13. **NULL**—Empty set
14. **BITERROR**—What happens when 0 is 1 and 1 is 0? [two words]
16. **SCRLK**—Screen lock
17. **ELECTRON**—1.602×10^{-19}
18. **ICOSAHEDRON**—A 20-faced polyhedron



PRIORITY INTERRUPT



by Steve Ciarcia, Founder and Editorial Director

8-Bit Control Is Dead – No Way!

For years tech pundits have been predicting the end of 8-bit micros. Apparently, with the prices of 16- and 32-bit MCUs constantly dropping, and presuming you always want your application to do more stuff, there is no reason not to replace a less powerful MCU, right? In my opinion, it was a false assumption then, and it still is today.

We can't look at this as a zero-sum game. Yes, 32-bitters open up all kinds of new opportunities for embedded processing, especially in the area of network-connected personal entertainment and information devices. But this doesn't mean they're a better fit in the low-end control and text-based applications that the 8-bitters have occupied for so long. The boundaries are certainly "fuzzy," but consider how we tend to generally categorize MCUs.

At the low end, we have the 8-bit controllers which typically have 8-bit data and registers along with 16-bit address paths. This is a sweet spot for all kinds of control and text-based functions that simply don't need to handle more than 64 KB of data at a time. The price/performance of the 8-bit chip should win this fight every time.

In the midrange, we have the 16-bit MCUs and lower-end 16-bit DSP chips. These chips can do a bunch more because they handle 16-bit data and have at least 24-bit address paths. There is often a hardware multiplier as well, which makes this class of chip ideal for many types of signal processing and audio applications.

At the high end, there is the 32-bit MCU/MPU (and higher-end DSPs) that have 32-bit data and address paths. These are the chips that have the power to drive an interactive graphical user interface and process video signals in real time.

It's clear that chip manufacturers believe in the future of all three classes of MCU; just look at the innovations they continue to introduce at all levels. Fundamentally, as the silicon improves in terms of transistor density, more memory fits onto a smaller chip, and there's more room for on-chip peripherals. Also, clock and power management has become a lot more flexible than ever before. The lower-end and midrange MCUs are all available with some combination of hardware timers (e.g., PWM, pulse capture, and motor control), communications (e.g., UART, SPI, I^2C, CAN, USB, etc.), and analog interface (e.g., ADC, DAC, and touch). Some include hardware controllers for multiplexed LCDs or Ethernet interfaces.

At the higher end, in addition to all of that, we also see options like on-chip SDRAM controllers, SD memory and I/O controllers, Ethernet MAC (and sometimes PHY), mass storage (ATAPI, SATA) and video support, including in some cases a separate GPU core. Basically, everything you need to run a full-up operating system like Windows, MacOS, or Android.

Probably the greatest result of across-the-board lower MCU costs is that we will be seeing multiple chips where just one was used before. This has been the situation with automobiles for years where reliability has increased with lots of "smart"-control modules all networked together. Certainly, this make senses in a $30,000 car, but the concept is moving down the cost spectrum as well. Take your typical household washing machine or dryer that has a motor or two and a control panel. Instead of one chip handling all of the control functions and user interface I/O, there will be one (or two) motor controller chip with a communications interface (e.g, SPI, I^2C, CAN, etc.) and a second chip with a communications interface along with an LCD controller and touch sensor support.

If the system designers are forward-thinking when they define the protocol by which these subsystems communicate, they'll end up with intelligent building blocks (e.g., "smart motor," "smart valve," "smart sensor") that can be easily reused in other products, keeping manufacturing costs low. The modules themselves will be reliable and energy-efficient, contributing substantially to end-user satisfaction and low recurring costs. The key is to make each module just smart enough without going overboard on processing power or overloading it with a top-heavy protocol.

And, that's where the lowly 8-bit MCU shines. A smart valve that just needs to sit on a LIN or 1-wire bus, operate a solenoid, and verify that it opened or closed doesn't need a lot of CPU cycles or 32-bit addressing to do the job. One of the tiny 8-bitters in a six- or eight-pin package will do nicely, and might even cost less than the manufacturing cost and testing of the dedicated wiring harness needed to do the job in the traditional way. There's no way a 16-bit or 32-bit MCU makes sense in this context. But more importantly, these lowly control tasks aren't going to go away. In fact, I think you'll be seeing a lot more of them and they'll all need MCUs. So, although it will be less visible, the 8-bit MCU will still be deeply embedded in increasingly subtle, but important, parts of your life, working hard so you don't have to.

steve.ciarcia@circuitcellar.com

FTDI Chip

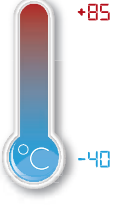
ARE YOU READY FOR THE X-CHIP FACTOR?

INTRODUCING THE X-CHIP USB BRIDGE FAMILY



EXtensive Interface Selection
UART, FIFO, SPI, I2C, FT1248

EXtended Temperature Range



Battery Charger DeteXion



EXpandable Clocking
INTERNAL CLOCK GENERATION
EXTERNAL CLOCK OUT



EXceptional Support



Visit FTDI at Embedded Systems Conference San Jose

Additional features:

- Royalty free, validated USB driver support
  Windows, Android, Linux & Mac OS
- Low active power: 8mA
- Internal MTP memory for flexibility
- Integrated 3.3V level converter

us.sales@ftdichip.com
www.ftdichip.com


PC Boards, Components, Assembly
because no one
should ever have
to wait again...
... imagineering has
Revolutionized the Online
Turnkey Quoting Process
Join the
Quoting Revolution
Happening Now @
PCBnet.com