■ PROJECTS ■ THEORY ■ APPLICATIONS ■ CIRCUITS ■ TECHNOLOGY

NUTS AND VOLTS

EVERYTHING FOR ELECTRONICS

www.nutsvolts.com
December 2012

Create A CHOOSE YOUR OWN ADVENTURE GAME For A Boredom-Free Commute

New! Nuts & Volts Mobile App
Available For iOS - Android Kindle Fire

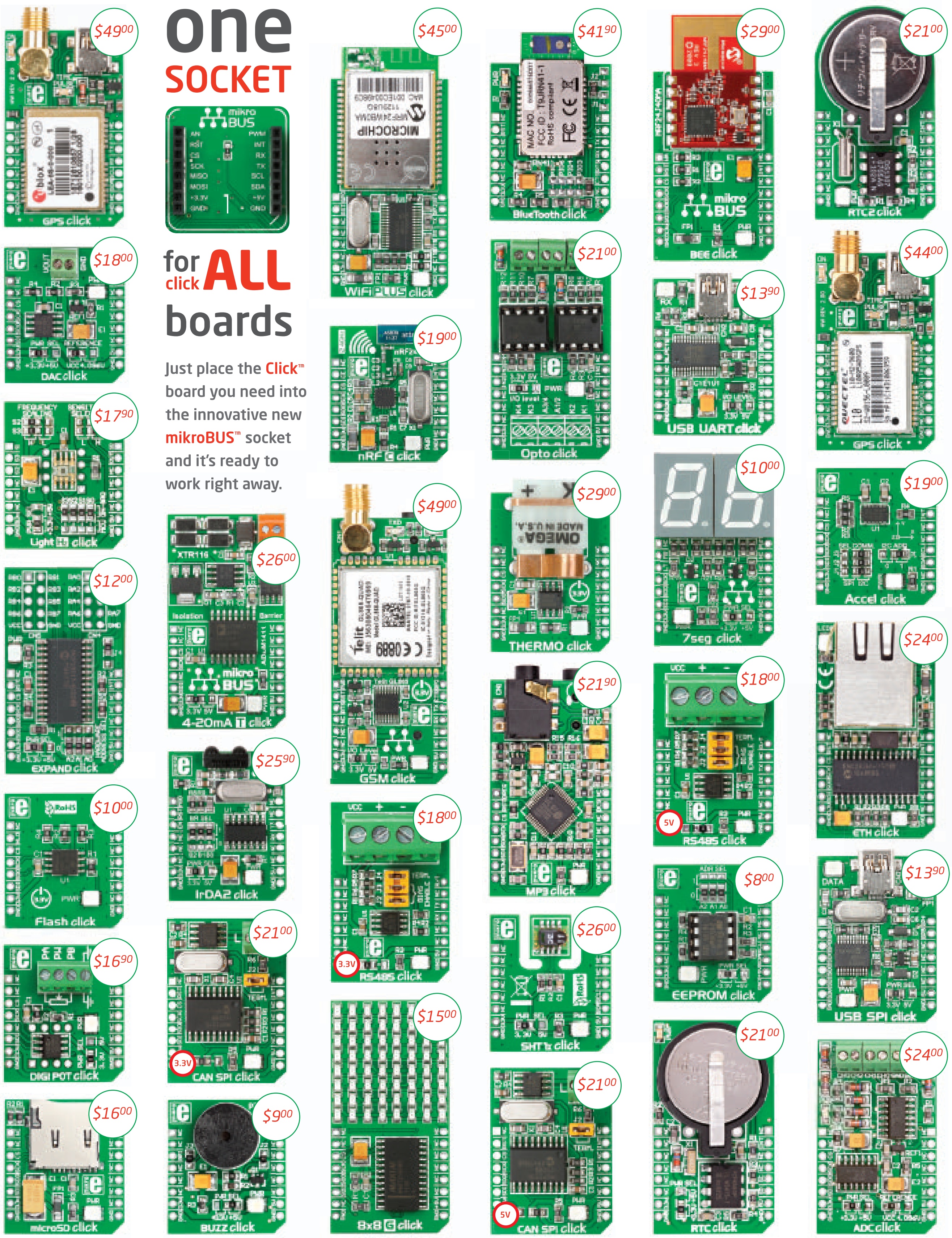
✦ Build The Desktop Contemplator

Gaze into its palette of colorosity to both amaze and calm yourself.

✦ Wireless Sensor Network Using XBee And The 32-bit Experimenter

It's back with two new wireless sensor network applications you can include in your next project.

U.S. $6.50 CANADA $7.50
12
0 71486 02421 7



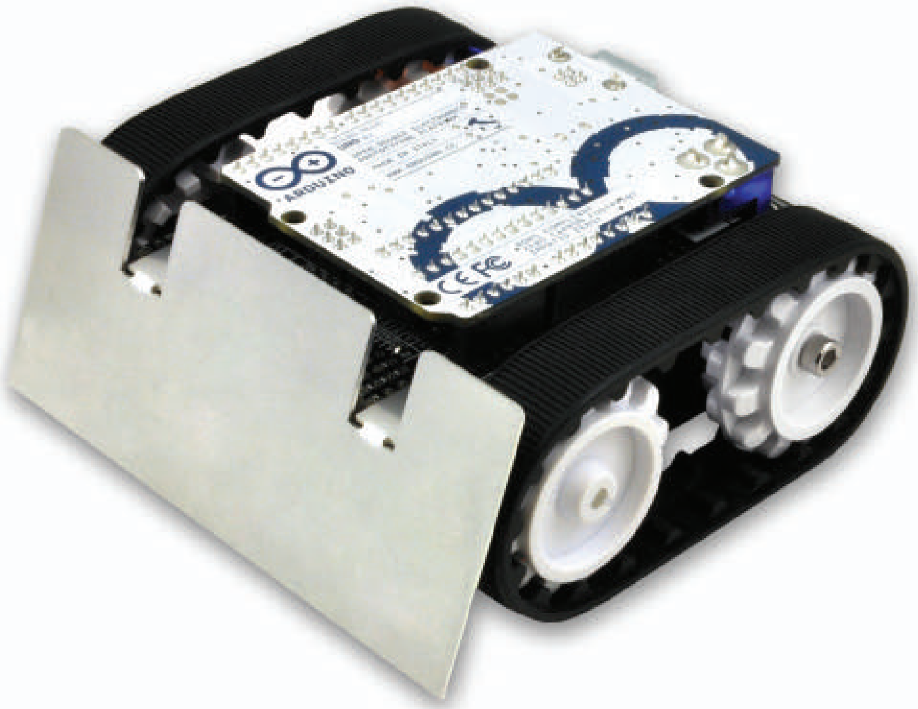
MikroElektronika
DEVELOPMENT TOOLS | COMPILERS | BOOKS

www.libstock.com

GET IT NOW
www.mikroe.com

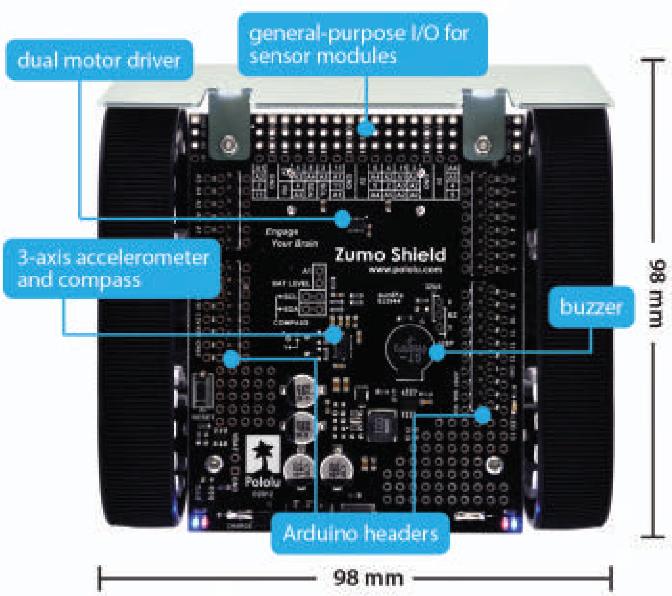
NOTE: Two 2x8 male headers are included in the package with each click board.

Get a little pushy.

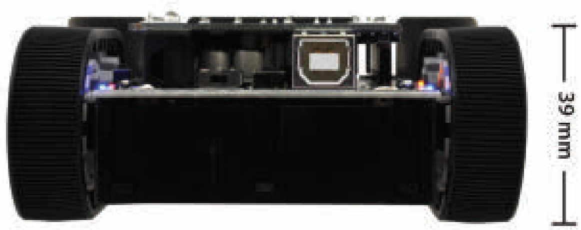


Zumo

Small enough for Mini-Sumo; flexible enough to make it your own.



Put your Arduino on the right tracks with the new Zumo chassis and Arduino shield! The Zumo is a small, tracked robot platform that works with a variety of micro metal gearmotors to allow for a customizable combination of torque and speed. Add a Zumo shield, which includes a dual motor driver, buzzer, and three-axis accelerometer and compass, to make an Arduino-controlled robot that can really throw its weight around!



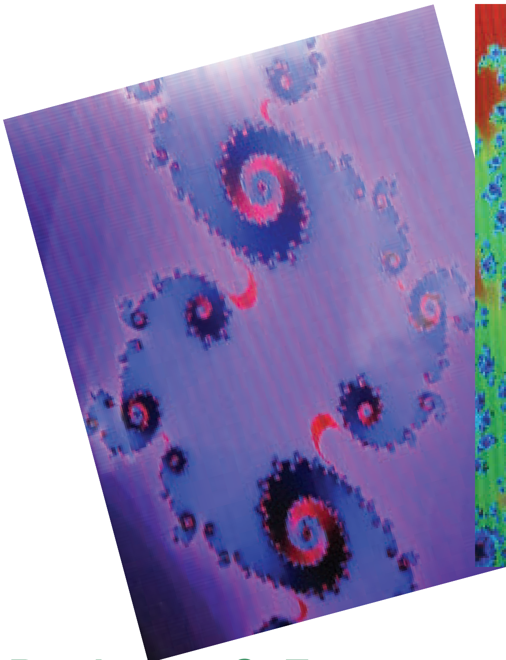
Pololu
Robotics & Electronics

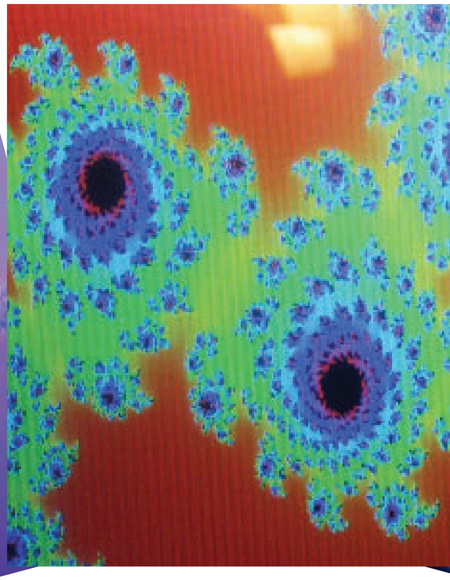
Learn more at www.pololu.com/zumo

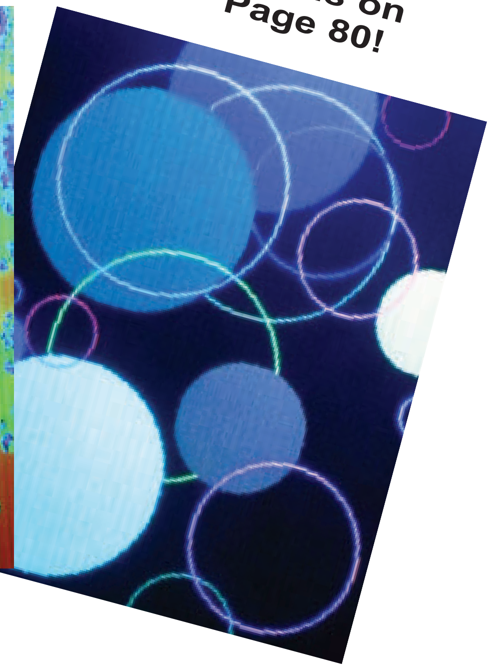
December 2012

www.nutsvolts.com

NEW! NV Mobile App now available! Details on Page 80!







Page 36

Projects & Features

Departments

Columns

Nuts & Volts (ISSN 1528-9885/CDN Pub Agree #40702530) is published monthly for $26.95 per year by T & L Publications, Inc., 430 Princeland Court, Corona, CA 92879. PERIODICALS POSTAGE PAID AT CORONA, CA AND AT ADDITIONAL MAILING OFFICES. POSTMASTER: Send address changes to **Nuts & Volts, P.O. Box 15277, North Hollywood, CA 91615** or Station A, P.O. Box 54, Windsor ON N9A 6J5; cpcreturns@nutsvolts.com.

DEVELOPING PERSPECTIVES

The Lost World of Tubes

Despite the popularity of the iPod and other svelte electronic music gadgets, the bulky, hot, temperamental tube amplifier is alive and well in the world of guitarists and bass players. Solid-state amplifiers can be had for about a dollar a watt, but you can pay $800 per watt or more for a quality tube amp. The difference is the 'tone' produced by a tube amp — the unique distortion that's created when a tube is driven near saturation.

The tube sound can be simulated on a chip and, of course, there's an app for that. However, the pros insist that there's still a gap between what a processer and an old fashioned tube can produce. I own both tube and solid-state guitar amps and admit that — while the tubes do sound subjectively better — they're a pain to maintain and use properly.

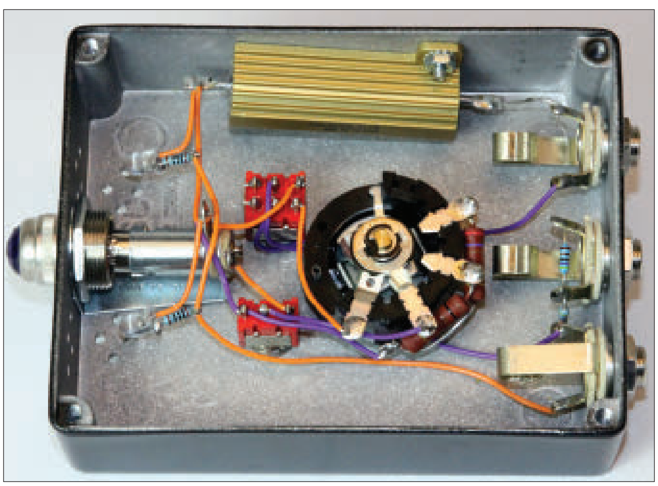
One of the universal truths of tube amps is that the volume level associated with saturation is more than neighbors or roommates will bear. At low volume settings, tube amps make annoying hisses and pops. Solid-state amps, in contrast, sound about the same at any volume level.

A common work-around is to put a tube amp in a closet, close the door, and run a microphone to the amp. I haven't tried this because of the obvious fire hazard posed by clothes hanging over and on a hot piece of electronics (I don't have a spare closet). The second approach is to purchase an attenuator that's placed between the amp output and the speaker. The idea is to dissipate the unwanted audio power as heat.

The problem with the attenuator approach — it will be obvious to readers who have experience building their own ham radio transmitters — is that placing high powered wire wound resistors, capacitors, and other components in the output circuit of a power amplifier has the potential for encouraging RF oscillations.

That is, you can transform a tube audio amp to an RF transmitter by placing inductance and capacitance in the output circuit. The potential for this problem seems to have been lost to the current generation of attenuator developers.

Another approach is to simply place series resistors in the speaker circuit. For example, in the attenuator shown in the accompanying **photo**, there's an eight ohm resistor in parallel with the output, and either zero, 100, or 200 ohms in series with the speaker.



This does the job of decreasing the speaker output, while matching the output impedance of the amp. Tube amplifiers don't tolerate significant mismatches in output impedance — tube life is shortened.

Another issue with tube-type components is price. Believe it or not, the attenuator in the photo sells for $200, and that's on the low end of what's commonly charged for a few power resistors in a box.

A better option would have been to purchase a 100W eight ohm 'L Pad' from Parts Express or Amazon for about $14. L pads are designed to vary the power delivered to a speaker while maintaining a constant load on the amplifier — typically four, eight, or 16 ohms.

Because knowledge of working with tubes is so limited, there are amazing products on the market that do very well. A popular attenuator that's based on two power resistors and a wire wound rheostat sells for $380 — quite a profit margin. There are similar markups for other peripherals for tube-type amps.

The takeaway is that there's money to be made if you know your way around a vacuum tube circuit or take the time to learn.

A painless way to rediscover the lost world of tube circuits is to pick up an inexpensive guitar amp (meaning cosmetically lacking) or a DIY tube amp kit from any number of suppliers such as **diyaudio.com**.

Of course, the definitive tube amp and audio reference — the Audio Cyclopedia — is a must-read for the tube audio enthusiast and experimenter. **NV**

NUTS AND VOLTS
EVERYTHING FOR ELECTRONICS

Published Monthly By
T & L Publications, Inc.
430 Princeland Ct.
Corona, CA 92879-1300
(951) 371-8497
FAX **(951) 371-3052**
Webstore orders only **1-800-783-4624**
www.nutsvolts.com

Subscriptions
Toll Free **1-877-525-2539**
Outside US **1-818-487-4545**
P.O. Box 15277
North Hollywood, CA 91615

FOUNDER/ASSOCIATE PUBLISHER
Jack Lemieux

PUBLISHER
Larry Lemieux
publisher@nutsvolts.com

ASSOCIATE PUBLISHER/ VP OF SALES/MARKETING
Robin Lemieux
display@nutsvolts.com

EDITOR
Bryan Bergeron
techedit-nutsvolts@yahoo.com

CONTRIBUTING EDITORS
Jeff Eckert, Russ Kincaid, Joe Pardue, Fred Eady, Ron Hackett, Lou Frenzel, Nuno Alves, Craig Lindley, Tom Kibalo

CIRCULATION DEPARTMENT
subscribe@nutsvolts.com

SHOW COORDINATOR
Audrey Lemieux

MARKETING COORDINATOR WEBSTORE
Brian Kirkpatrick
sales@nutsvolts.com

WEB CONTENT
Michael Kaudze
website@nutsvolts.com

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

PRODUCTION/GRAPHICS
Sean Lemieux

READER FEEDBACK

Arduino Newbie

I just finished reading Bryan Bergeron's October 2012 column. It was very appropriate since I have been thinking about getting on board with the Arduino wave. I first started as a hobbyist with the PICAXE microcontroller and followed along with Ron Hackett's initial articles from *N&V*. I read through the column and went to the various websites to find out some basic information and have a few questions:

1. Which is a better choice regarding which platform to go with for a beginner/hobbyist: the ATmega or the chipKIT?
2. Is the code compatible between the different IDEs (didn't see any info regarding this on the websites)?
3. Can I assume all the shields out there can be used with either?
4. Can I also assume Pmods are compatible with the ATmega Arduino?

Thank you for a great magazine!

Dave Pollatta
Ontario, NY

If this is your first venture into Arduino territory, I'd go with a vanilla Arduino UNO. Tons of support and virtually every shield designed for the Arduino will work. The chipKIT is nice — I use both the standard Arduino and chipKIT — but the IDE is different and the code is often just slightly different. Enough to cause confusion if you're a beginner. Sort of like learning to drive in a little sports car. Unless you really need the 80 MHz clock, I'd go with the 16 MHz Arduino.

The chipKIT works with the shields I've tried, but I'm sure there are exceptions. Have fun!

Bryan

License to Use

I am writing to express my concern over the article titled "A Simpler-To-Make Near Space Flight Computer," by L. Paul Verhage in the November 2012 issue. The article discusses the use of a Radiometrix BiM1HT-144.390-10 transmitter module which is a transmitter operating in the licensed amateur radio

Continued on page 71

TECH 2012 KNOWLEDGEY

■ BY JEFF ECKERT

EVENTS, ADVANCES, AND NEWS

ADVANCED TECHNOLOGY

DARK ENERGY SURVEY UNDERWAY

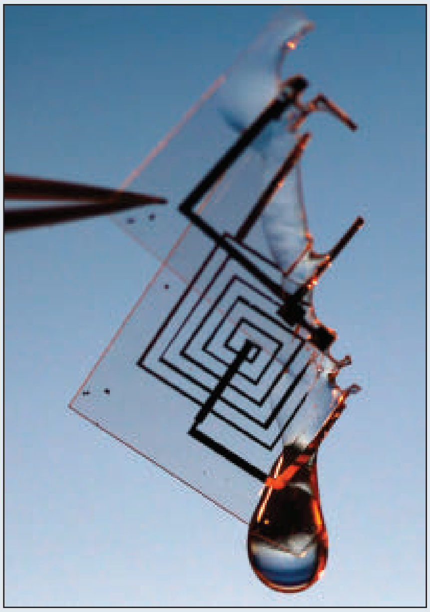
■ Image of the barred spiral galaxy NGC 1365 — about 60 million light years away. *Courtesy of Dark Energy Survey Collaboration.*

This fall, scientists collaborating on the Dark Energy Survey down at Chile's Cerro Tololo Inter-American Observatory switched on the 570 megapixel dark energy camera for the first time and captured rays of light that have been traveling from distant galaxies for some eight billion years. The camera — said to be the most powerful instrument of its kind — is designed to probe the mystery of dark energy which is thought to be the force causing the universe to expand at an ever-increasing rate. The camera — which is about the size of a telephone booth — employs a 13 ft diameter light-gathering mirror and 62 CCDs with extreme sensitivity to red light.

The survey — beginning this month and continuing over the next five years — will create detailed color images of one-eighth of the sky (or 5,000 square degrees) to discover and measure 300 million galaxies, 100,000 galaxy clusters, and 4,000 supernovae. According to Argonne Labs physicist Steve Kuhlmann, "With all of our amazing scientific progress in the last century, we still only understand the four percent of the universe that is made of normal matter. The Dark Energy Survey will help us understand the other 96 percent, which we believe is made of dark matter and dark energy." (Many of the camera's critical systems are controlled with Argonne-designed technology.) You can follow the survey's progress at **www.darkenergysurvey.org**. ▲

TRANSIENTS CAN BE GOOD

■ Silk-silicon electrical circuit is dissolved by a drop of water.



Electronic transients are usually undesirable things, but transient electronics are a different story, according to a team led by biomedical engineers at Tufts University (**www.tufts.edu**) and researchers at the University of Illinois at Urbana-Champaign (**illinois.edu**). These folks have developed a new class of silk-silicon devices that should prove useful in medical implants, environmental sensors, and a range of consumer electronics. For example, a medical implant within a human body could function for its intended life span and then dissolve away harmlessly, or consumer electronics such as cell phones could be built to decompose quickly when dropped into a landfill instead of slowly deteriorating for years. The time frame can range from minutes to years, depending on the mix of fabrication materials.

The circuits use traditional IC materials such as silicon and magnesium, but they are constructed in an ultra-thin form and are encapsulated in silk protein. "These devices are the polar opposite of conventional electronics whose integrated circuits are designed for long-term physical and electronic stability," noted Prof. Fiorenzo Omenetto of the Tufts School of Engineering. "While silicon may appear to be impermeable, eventually it dissolves in water." The trick is to fine-tune the silk properties so the circuits dissolve at the right time.

In the future, the researchers are looking toward devices whose properties can be adjusted in real time or that can respond to environmental factors such as chemistry, light, or pressure. A paper detailing the concept published in the September 28, 2012 issue of *Science* is available at **scipak@aaas.org**. ▲

COMPUTERS AND NETWORKING

DESKTOPS: NOT DEAD YET

Although rumors of the death of the desktop are not entirely exaggerated, traditional style PCs are still a staple of medium to large businesses. That is exactly the targeted market for Lenovo's latest addition to its M Series lineup — the ThinkCentre M78 — which is built around the latest AMD A-Series accelerated processing units. (An APU is a combined CPU and graphics processor, in case you haven't been keeping up with the terminology.)

Available in a choice of tower or small form factor boxes, the M78 can be configured with up to 32 GB of DDR3 memory and a 1 TB SATA drive, and it comes with the company's RapidBoot HDD accelerator for speedier boot-up. It also includes support for up to three independent monitors, four USB 3.0 ports, the AMD Intelligent Cooling Engine (i.e., automatic fan speed adjustment), an optional 80 Plus Platinum power supply for added efficiency, and enhanced data protection. Okay, it's not as sexy as an all-in-one unit or a top-of-the-line laptop, but what do you want for $479? Full specs are available at **www.lenovo.com**. ▲

■ Lenovo's ThinkCentre M78 desktop models.

HELIUM-FILLED HDDS COMING SOON

Although solid-state drive technology is advancing and prices are dropping, mechanical hard drive technology is still moving ahead as evidenced by a "radically new" helium-filled platform from Western Digital's HGST division (formerly Hitachi Global Storage Technologies). With products expected to hit the market early next year, the new platform extends the company's five-platter designs to seven 3.5 inch platters to "significantly improve datacenter total cost of ownership (TCO) on virtually every level: capacity, power, cooling, and storage density."

Because helium is only one-seventh as dense as air, the disk stack encounters less drag, resulting in a 23 percent reduction in required mechanical power. It also allows the disks and data tracks to be placed closer together, so the drives will run cooler and with less noise. The combined reduction in wattage and higher density is said to deliver a 45 percent improvement in watts per terabyte. As of this writing, HGST has not released the specifics on capacities and prices, but by the time you read this it may be posted on **www.hgst.com**. The big question is whether audio files stored on the drives will sound like chipmunks on playback. ▲

■ NASA's Free Drive simulator lets you command the Curiosity rover.

PUT YOUR CURIOSITY IN GEAR

If you are a fan of the Mars Rover project (and isn't everyone?), you'll want to check out NASA's "Explore Mars: Free Drive" online experience which puts you in command of a virtual Curiosity in the Gale Crater on Mars. You can explore the landing site, a series of fractures, a canyon, some sand dunes, and even a phyllosilicate trough. The rover's six wheels even interact with the surroundings, so you can do doughnuts or write things in the sand. You will have to download and install the Unity Web Player to run the simulation, but that shouldn't be a problem if you are running a relatively new operating system. So, drop into **mars.jpl.nasa.gov/explore/freedrive** and take her for a spin. ▲

CIRCUITS AND DEVICES

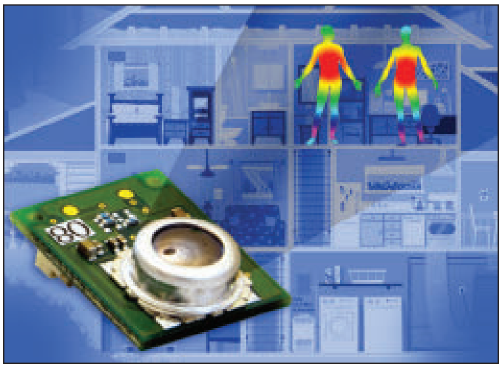
LED LAMP FEATURES LIQUID COOLING

The SWITCH60 LED lamp produces an incandescent-like spectrum for up to 25,000 hours.

Evolution in the LED lighting industry continues, and one of the latest is the SWITCH60 from SWITCH Lighting (**www.switchlightingco.com**). The 60 watt equivalent unit provides the same warm light spectrum of an incandescent while using 80 percent less energy. Perhaps the most interesting characteristic is that it uses the company's LQD Cooling System™ for optimal thermal management.

The system "is a blend of the company's overall electronic design and proprietary technologies: a coolant made of liquid silicone and a patented, highly efficient, and reliable electronic driver." The SWITCH60 isn't commercially available as of this writing, but it should be on the market soon with a price tag somewhere between $40 and $60 per unit. That's a bit on the high side for home use, so the product is presently aimed primarily at the hospitality industry (hotels and so forth). In fact, it was selected as a "Best of Boutique" winner at a recent trade show sponsored by *Hospitality Design* magazine. The reduced power consumption plus a 25,000 hour lifetime might make it cost-competitive, so check your local power rates and run the numbers. ▲

NEW IR MEMS SENSOR



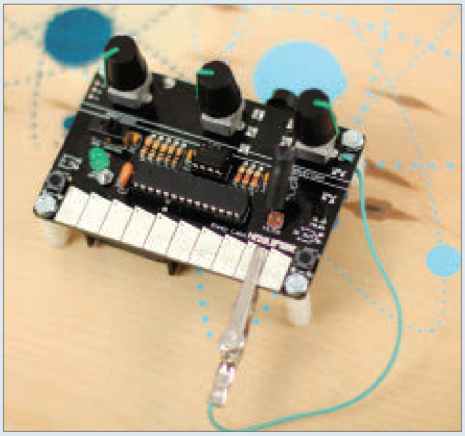
Omron's D6T senses human presence without the need for movement.

Omron (**www.omron.com**) has come up with a new, highly sensitive sensor aimed primarily at home and building automation, health care, security, and industrial applications. The D6T sensor operates on the principle of infrared radiation detection and is said to be particularly useful for sensing the presence of humans. Thus, it can be used to automatically switch off lighting, air conditioning, and other services when a room is no longer occupied.

Unlike many detectors, it does not require movement to sense people in an area. As D6T sensors are also able to monitor the temperature of a room, they can be used to control both heating and air conditioning systems to maintain energy-efficient room temperature levels. Additionally, the sensors can be used to watch for unusual temperature changes so as to identify hot spots where a fire could break out, and even to monitor hospital rooms to ensure that the patient has not left the bed. As explained by the manufacturer, "The technology behind Omron's D6T thermal sensors combines a MEMS micro-mirror structure for efficient IR radiation detection with a high performance silicon lens to focus the infrared rays onto its thermopiles. Proprietary application-specific integrated circuits then make the necessary computations and convert sensor signals into digital I^2C outputs. The result is excellent ±1.5°C accuracy with high noise immunity." The sensors — which measure 18 x 14 x 8.8 mm (0.7 x 0.6 x 0.34 in) — offer a detection range of 5°C to 50°C (41°F to 122°F). The units were introduced in November at Electronica 2012 (Munich, Germany), but no price information has been released. ▲

BUILD YOUR OWN NEBULOPHONE

According to their "About" web page, Bleep Labs "designs and manufactures noisy objects in Austin, TX." So, if you like noisy things, know how to operate a soldering iron, and have $55 to spare, might we suggest the Nebulophone — perhapds the brainchild of Dr. Bleep himself? What you get is an Arduino-based synthesizer that (despite its toy-like appearance) actually features "perfect tuning across six octaves," eight waveforms, six adjustable arpeggio modes, a programmable sequencer, a portamento sound with three speed settings, and a range of other capabilities. The stylus keyboard will probably take some getting used to, but the sounds are pretty cool — just check out **bleeplabs.com/nebulophone** for a demo. If you're not too confident about your soldering skills, Bleep will give you a tutorial on their Soldering 101 web page so to get you up to speed. Or, you can just shell out an extra $25 for a preassembled unit and start annoying people right away. **NV**



Bleep Labs' Nebulophone — a $55 synthesizer kit.

Jaycar Electronics
Better, More Technical

Order online: www.jaycar.us
Free call orders: 1800 784 0263
December 2012

Kits for Electronics Enthusiasts

18m IR Light Barrier Kit
Cat. KG-9096

This kit consists of an infrared receiver and transmitter and will shoot an IR beam 18 meters. Use with driveway or pathway monitoring, automatic garage door triggering or shop front/office entry monitoring. Includes a daylight filter to eliminate false triggering in sunlight. The receiver's relay can be used to trigger an alarm, siren or relevant triggering device.

- TX requires 9VDC 90mA; RX 12VDC 100mA.

$29.00*

Also available:
50m IR Light Barrier (KG-9196 $36.00)

Annunciator Kit
Cat. KC-5420

Need people to take a number when waiting to be served? This electronic signalling device has digits 75mm high, each using 28 high intensity red LEDs. The numbers display from 00 to 99 is incremented by pressing a button on the separate small control box. Kit includes case, PCB and electronic components. Requires 12VDC power.

- Main Box Size: 197 x 113 x 63mm
- Control Box size: 76 x 46mm

$43.25*

Driving Light Anti-Theft Kit
Cat. KC-5337

Protect your expensive Driving or Fog Lights that are otherwise vulnerable to theft. Simply connect it to the power cable on the lights you want to protect, and to your car alarm or siren and you are set. If the lights are removed or the cable is cut, the alarm will sound.

- Kit includes: PCB, wire and electronic components

$14.50*

RFID Security Module Receiver Kit
Cat. KC-5393

Radio Frequency Identity (RFID) is a contact-less method of controlling an event such as a door strike or alarm etc. An "RFID Tag" transmits a unique code when energized by the receiver's magnetic field. As long as a pre-programmed tag is recognized by the receiver, access is granted. This module provides normally open and normally closed relay contacts for flexibility. It works with all EM-4001 compliant RFID tags.

- Kit supplied with PCB, tag, and all electronic components

$65.00*

Infrared Floodlight Kit
Cat. KG-9068

Let your CCD camera see in the dark! This infrared light is powered from any 12-14VDC source and uses 32 x infrared LEDs to illuminate an area of up to 5m (will vary with light conditions).

- Kit is supplied with silkscreened/ gold plated/ solder-masked PCB, 32 x Infrared LEDs and all electronic components
- PCB: 74 x 55mm

$21.75*

Note: Not suitable for color CMOS cameras

Infrared Detector Kit
Cat. KG-9086

This kit will switch on a LED when it detects infrared light from sources such as IR remote controls. Connect it to the Relay Card kit KG-9142 to make an infrared remote controlled relay. Project requires 9VDC. Can be battery powered.

- Kit supplied with Kwik Kit PCB, Infrared receiver and all electronic components
- PCB size: 55 x 15mm

$7.25*

4Ch CCTV Camera Switcher Kit
Cat. KC-5316

This project allows you to keep an eye on what's happening in each camera's field of view. It can accommodate up to 4 cameras and features variable scan rate as well as a pause button to freeze the scan for specific camera viewing. LEDs on the front panel indicate the camera channel being monitored. 12VDC 110mA required.

- Kit includes case, punched & silk-screened panelling plus electronic components

$37.50*

45 Second Voice Recorder Kit
Cat. KC-5454

This kit easily record two, four or eight different messages for random-access playback or a single message for 'tape mode' playback. It also provides cleaner and glitch-free line-level audio output suitable for feeding an amplifier or PA system. It can be powered from any source of 9-14VDC.

- Supplied with silk screened and solder masked PCB and all electronic components
- PCB: 120 x 58mm

$25.25*

POPULAR KITS

Jacob's Ladder High Voltage Display Kit MK2
Cat. KC-5445

With this kit and the purchase of a 12V ignition coil (available from auto stores and parts recyclers), create an awesome rising ladder of noisy sparks that emits the distinct smell of ozone. This improved circuit is suited to modern high power ignition coils and will deliver a spectacular visual display. Kit includes PCB, pre-cut wire/ ladder and electronic components.

- 12V car battery, 7Ah SLA or > 5A DC power supply required
- PCB: 170 x 76mm

$31.00*

Voltage Monitor Kit
Cat. KC-5424

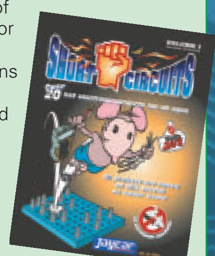
This versatile kit will allow you to monitor the battery voltage, the airflow meter or oxygen sensor in your car. The kit features 10 LEDs that illuminate in response to the measured voltage, preset 9-16V, 0-5V or 0-1V ranges. Complete with a fast response time, high input impedance and auto dimming for night time driving. Kit includes PCB with overlay, LED bar graph and electronic components.

- 12VDC
- PCB: 74 x 47 mm
- Recommended box: UB5 (use **HB-6015 $2.00***)

$16.75*

Short Circuits - Volume 1

This volume will teach you everything you need to get started in electronics and is suitable for ages 8+. We give you the option of buying the book on its own, or together with the accompanying kit that contains the components for each of the 20-odd projects described in the book. Some of the exciting projects include a Police Siren, Electronic Organ, Sound Effects Unit, Light Chaser and many, many more! The full color 96 page book, is lavishly illustrated with over 100 drawings and diagrams. No prior knowledge of electronics is needed, projects are fun and safe to build.



BJ-8502

KJ-8504

Short Circuits Book
BJ-8502 $7.25*

Short Circuits Project Kit
KJ-8504 $24.50*

Short Circuits Book and Project Kit
KJ-8502 $29.00*

KJ-8502

Ideal Kids Christmas Presents

SHORT CIRCUITS

NOW SHIPPING VIA DHL
• FAST DELIVERY • TRACK SHIPMENT

5-10 working day delivery

Post & Packing Charges

Order Value	Cost
$25 - $49.99	$7.50
$50 - $99.99	$20
$100 - $199.99	$40
$200 - $499.99	$60
$500+	$75

- Max weight 12lb (5kg).
- Heavier parcels POA.
- Minimum order $25.

Note: Products are dispatched from Australia, so local customs duty & taxes may apply.

HOW TO ORDER

WEB:	www.jaycar.us
PHONE:	1-800-784-0263*
FAX:	+61 2 8832 3118*
EMAIL:	techstore@jaycar.com
POST:	P.O. Box 107, Rydalmere NSW 2116 Australia

*Australian Eastern Standard Time (Monday - Friday 09.00 to 17.30 GMT + 10 hours)

• ALL PRICING IN US DOLLARS

Prices valid until 31/12/2012

***All prices EXCLUDE postage & packing**

PICAXE PRIMER

■ BY RON HACKETT

SHARPENING YOUR TOOLS OF CREATIVITY

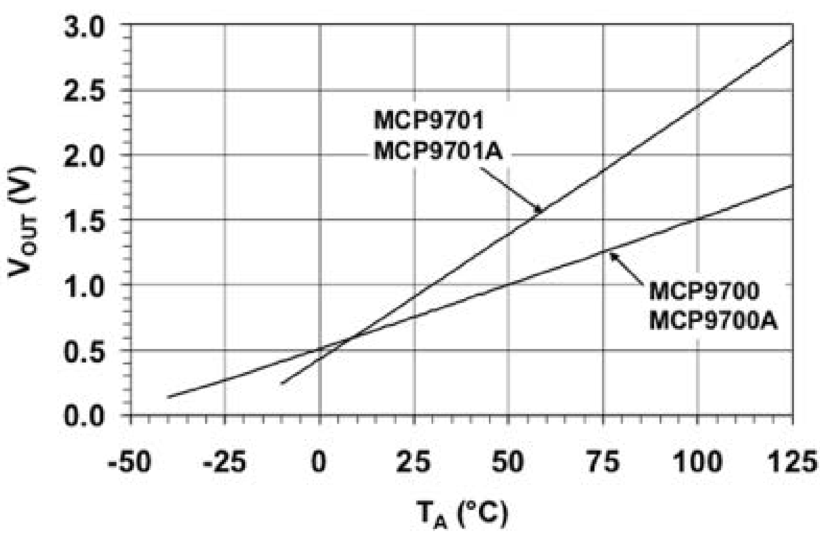
WRAPPING UP AND MOVING ON: THE MCP9700A AND BATTERY-POWERED SYSTEMS

At this point, we've spent a fair amount of time on our LED-2x7 project and on measuring temperature with the DS18B20 digital thermometer, so I thought we were ready to move on to an entirely new topic. However, I did have one lingering doubt; the LED-2x7 hardware and the DS18B20 turned out to be such a useful combination, that I found myself thinking about other possible applications for them. The problem is that the DS18B20 is a relatively expensive component. In fact, an 18B20 costs more than the 20M2 processor itself. I don't know about you, but that doesn't seem right to me!

THE MICROCHIP MCP9700A ACTIVE THERMISTOR TO THE RESCUE

Coincidentally, as I was considering possible alternatives to the DS18B20, I happened to read a post by "Svejk" on the PICAXE forum in which he suggested the Microchip MCP9700A as an inexpensive alternative to the 18B20. He also provided a link to a brief paper he had written on the subject, so I immediately downloaded it. (Svejk also gave me permission to make his paper available on my website; it's in the "sensor" section if you're interested.) I was surprised to see how easy it is to interface the 9700A with a PICAXE processor, so I also downloaded and read the 9700A datasheet (which is also available in the sensors section of my site).

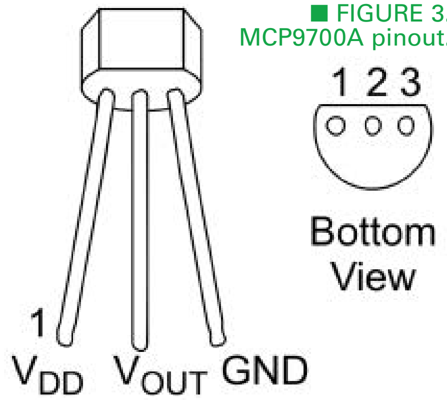
Microchip calls the 9700A a "low-power linear active thermistor," which was greatly reassuring to me. I have used thermistors in the past, and I found it tedious to write the software that's necessary to correct for the lack of linearity in their temperature response. However, those thermistors were not "active" — they were passive components whose resistance varied with



■ FIGURE 1. MCP9700A output voltage vs. ambient temperature.

	DS18B20	MCP9700A
Celsius Temp. Range	-55°C to +125°C	-40°C to +125°C
Fahrenheit Temp. Range	-67°F to +257°F	-40°F to +257°F
Accuracy	±0.5°C from -10°C to +85°C	±2°C from -0°C to +70°C
Supply Range	+3.0V to +5.5V	+2.3V to +5.5V
Turn-on Time	800µS	800µS
Interface Requirements	Bi-directional Pin	Input Pin
Active Current Draw	1.5mA	6µA
Access Speed (See Text)	750mS	1.5mS

■ FIGURE 2. Comparison of major features of the 18B20 and 9700A.



■ FIGURE 3. MCP9700A pinout.

temperature. In comparison, the 9700A is "active" because it produces an analog voltage that varies directly with the temperature.

Fortunately, the voltage output of the 9700A is — in fact — very linear, as you can see in **Figure 1** (source: MCP9700A datasheet). In addition, the output is optimized for use with most analog-to-digital converters (ADC), including the PICAXE *readadc10* command; at 0ºC, the output is set at 500 mV, and it varies directly at a consistent 10.0 mV/ºC. As we'll soon see, this simplifies the software that's necessary to convert the voltage reading to a corresponding temperature.

Figure 2 compares the major features of the DS18B20 and the MCP9700A that are relevant for our projects. It seems to me that the only disadvantage of the 9700A is that it's not as accurate as the 18B20. However, for my purposes, an accuracy of ± 2ºC is good enough — especially considering the 9700A's other major advantages, e.g., a lower minimum operating voltage of 2.3V, as well as significantly lower power consumption and shorter access (command execution) speed. (If you want to conserve as much power as possible, the active power consumption of both devices is low enough to be able to power either one of them directly from a PICAXE output pin.)

The difference in access speed is due to the difference in speed of execution of the PICAXE *readtemp12* command — which is required by the 18B20 — and the *readadc10* command — which is required by the 9700A.

In case you're wondering why low power operation is so important to me, let me explain. Ages ago, in the August 2008 installment of the Primer, I mentioned that one of the projects we were going to develop would be a battery-powered data collection system. That project is still on my to-do list because there always seems to be something new and exciting to explore in the Primer.

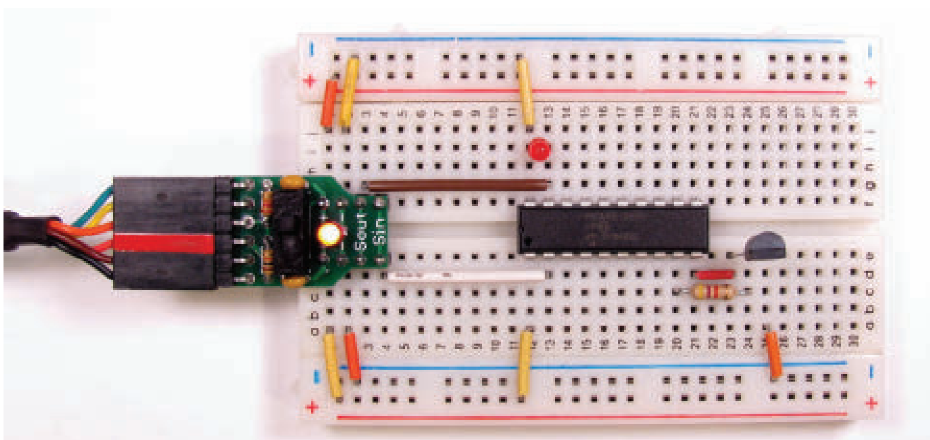
However, I'm determined to head in that direction, and as a first step I plan to use a battery-powered breadboard setup for some of our upcoming experiments. My plan is to leave my breadboard setup powered on 24 hours a day so that I can collect some data on its real world battery performance. We'll talk more about a data collection project after we finish our exploration of the MCP9700A.

TESTING THE MCP9700A

The pinout of the MCP9700A is shown in **Figure 3** (source: MCP9700A datasheet). If you compare it to the pinout of the DS18B20, you'll see that the order of the pins is reversed which means that you can replace an 18B20 with a 9700A in any existing circuit by simply rotating the 9700A 180º (compared to the 18B20 it is replacing).

In order to test the 9700A, I set up the simple breadboard circuit shown in **Figure 4**. I used a 20M2 for my testing, but any current PICAXE processor will work. I also powered the 9700A via the 20M2's C.0 pin, just to show how simple it is to save a little power by powering down the device when it's not in use. (The same technique should work with any powered I/O device that draws less than 25 mA.)

Note that I included a 4.7K resistor in the connection between the 9700A output pin and the



■ FIGURE 4. Breadboard setup for MCP9700A experiments.

20M2's C.1 pin. This isn't required by the 9700A; its function is to protect the 9700A and the 20M2 from possible damage. If pin C.1 were accidentally configured as an output, there would be two different voltages on the same line. That would constitute a direct short, and possibly result in damage to either or both devices.

As usual, the software for our first test of the 9700A (*Test9700A.bas*), along with the other programs for this month, are available for downloading at the article link provided. When you have downloaded the *Test9700A.bas* program to your breadboard setup, you should see its raw output data repetitively received in the terminal window. If not, you will need to re-check your wiring.

USING THE FVRSETUP AND ADCCONFIG COMMANDS

One of the advantages of the MCP9700A that I mentioned earlier is that its output is optimized for use with the PICAXE *readadc10* command. Before I can explain what I mean by that statement, we first need to discuss a PICAXE hardware feature that we haven't seen before.

All M2-class (and the 28X2 and 40X2) processors have what is known as an internal fixed voltage reference (FVR). In order to understand how it functions, we need to briefly review our previous usage of the *readadc* and/or *readadc10* commands.

In the past whenever we've used either one of these commands, the ADC reading was based on the processor's supply voltage. There are three limitations or issues related to that approach to ADC inputs, especially when we want to determine the actual value of the voltage that's being measured:

1. The *readadc* command returns an eight-bit byte value, so it uses a 256 level A-to-D (analog-to-digital) conversion. Similarly, the *readadc10* command returns a 10-bit word value, so it uses a 1024 level A-to-D conversion. In either case, if we want to convert the A-to-D level back to a voltage value, we need to do some arithmetic that involves dividing 5.0 by 256 or by 1024. Given the fact that PICAXE BASIC is integer-based, coding either one of those calculations is not fun to say the least!

2. When we do our A-to-D calculations, we assume that the output of our regulated power supply is exactly 5.0V which is almost never true, and the discrepancy can easily be large enough to produce a significant error in our results.

3. If we are working with a battery-powered project and want to monitor the supply voltage itself, we have a major problem. Suppose we want our project to sound an alarm when its battery voltage drops below a certain value. Using a voltage divider definitely won't work. For example, suppose we set up a voltage divider on the supply line with two resistors of the same size. When the battery is fully charged, a *readadc* value will be about 128, because two equal resistors divide the ADC range in half. When the battery supply is on the verge of total exhaustion, the *readadc* value will still be about 128; in fact, it will always be at the midpoint no matter what the condition of the battery.

Fortunately, the internal FVR solves all three of these problems; let's see how it does so. By default, the FVR hardware is disabled, but we can use the *fvrsetup* command to activate and configure the FVR (or to deactivate it again) by issuing the command in one of the following four forms:

```
fvrsetup FVR1024
'enable FVR at 1024mV
fvrsetup FVR2048
'enable FVR at 2048mV
fvrsetup FVR4096
'enable FVR at 4096mV
fvrsetup off
'disable FVR
```

There are a couple of FVR limitations that we need to keep in mind. First, the FVR output can't be greater than the supply voltage. For example, with a 3.3V supply, an *fvrsetup FVR4096* statement cannot be used. Second, an FVR level of 1,024 mV (1.024V) can't be used in conjunction with an ADC command. For the purpose of monitoring a battery-powered supply line, our only choice is the *fvrsetup FVR2048* statement which is the one I would want to use anyway. (We'll talk more about that shortly.)

Enabling the FVR hardware doesn't automatically set its output voltage as the reference for an ADC command. In order to do that, we also need to issue an *adcconfig* command which has the following format: *adcconfig config*.

The *config* parameter is a three-bit binary number that specifies the ADC reference voltages. With M2-class processors, *config* can have one of six different values. However, only one of the possible values (%011) is suitable for monitoring a battery-powered supply line, so we won't go into the details of the other five options. (If you're curious, refer to the *adcconfig* documentation in Part 2 of the manual.)

By default, the Vref+ signal for an ADC command is the supply voltage (V+), and the Vref- signal for an ADC command is ground (0V). In order to change the ADC defaults to what we want (Vref+ = FVR, and Vref- = ground), all we need to do is include the following statement in our program: *adcconfig %011*.

There's one final point about the *fvrsetup* command that I need to mention. Enabling the internal FVR hardware slightly increases the processor's power consumption. Therefore, in order to conserve power, the FVR hardware is automatically disabled after every *ADC* command. As a result, if we use an ADC command in a loop, we need to precede it with an *fvrsetup* command in order to re-enable the setup that we want.

Assuming that the *sensor* and *ADCval* variables have been previously defined, the following code snippet illustrates everything we have been discussing:

```
' === Begin Main Program ======
  dirsB = %11111111
    'portB pins all outputs
  dirsC = %10111101
    'ADC input on pin C.1
  adcconfig %011
    'Vref+=FVR, Vref-=gnd

do
  high pwr9700
    'power up MCP9700A
  pause 10
    'startup delay
  fvrsetup FVR2048
    'set FVR at 2048mV
  readadc10 sensor,ADCval
    'fetch raw data
  low pwr9700
    'turn off MCP9700A
  sertxd (#ADCval,CR,LF)
    'display data
  wait 5
    'slow down a bit
loop
```

The only aspect of the above code that I haven't mentioned is the *pause 10* statement. There's a slight time delay involved as the 9700A powers up each time through the *do/loop*, so the pause statement simply allows time for the 9700A to stabilize before the ADC reading occurs.

CONVERTING THE MCP9700A RAW DATA TO DEGREES CELSIUS

I still haven't explained how the PICAXE M2-class FVR hardware is going to simplify life for us, so now's a good time to do so. The key is the fixed voltage reference of 2,048 mV. The *readadc10* command divides that value into 1,024 steps, which means that each ADC step is exactly 2 mV — how convenient is that!

Also, as we saw earlier in **Figure 1**, the 9700A output equals 500 mV at 0ºC, and it varies directly at a consistent rate of 10.0 mV/ºC. All this combines to simplify the math, as we're about to see.

At this point, we need to go back to the graph of **Figure 1**, and derive an equation for the relationship between ambient temperature and the ADC value. (If algebra isn't your forte, you can skip this part, and just use the final equation.) We'll start with the standard algebraic equation for a straight line:

$$y = mx + b$$

Then, we'll use the "two point" version we learned in high school:

$$y = \frac{(y_2 - y_1)}{(x_2 - x_1)} x + b$$

From **Figure 1**, we'll choose two convenient points [p_1 = (50, 1.0) and p_2 = (100, 1.5)], and convert the voltage measurements to millivolts [p_1 = (50, 1000) and p_2 = (100, 1500)]:

$$V_{out} = \frac{1500 - 1000}{100 - 50} * tempC + 500$$

Next, we'll simplify the equation:

$$V_{out} = 10 * tempC + 500$$

and then solve the resulting equation for the *tempC* variable:

$$10 * tempC = V_{out} - 500$$

At this point, we would normally divide everything by 10, but as you know we can't do that because PICAXE BASIC is limited to integers. We'll just have to remember to "trick" the compiler by inserting a decimal point in the correct position when we write the actual code.

However, we aren't quite done yet because we don't actually have a value for V_{out}; instead, we have the ADC value that is returned by the *readadc10* command. Thanks to the FVR hardware, $V_{out} = 2 * ADC_{val}$, so we can just substitute that for V_{out} in the equation:

$$10 * tempC = 2 * ADC_{val} - 500$$

Now, we can use the final equation that we just derived to write the code that's necessary to convert the raw ADC reading into the Celsius temperature. Of course, we also have to be aware of the fact that PICAXE BASIC can't do computations that involve negative integers, and whenever ADC_{val} < 250, 10 * *tempC* will be negative. Therefore, we'll need an *if/then* statement to "reverse" the subtraction in order to avoid a negative result whenever ADC_{val} is < 250. Assuming that all the variables have been previously defined, the following code snippet illustrates the conversion from ADC_{val} to *tempC*:

```
if ADCval < 250 then
  tempC = 2 * ADCval
  tempC = 500 - tempC
  sign = "-"
else
  tempC = 2 * ADCval - 500
  sign = "+"
endif
```

Our next program (*Temp 9700A.bas*) incorporates the two code snippets we just discussed. In addition, it includes a Celsius to Fahrenheit conversion subroutine so it will output the temperature in both scales. I won't go into the details of the conversion, but the process is similar to the conversion routine we discussed in the previous Primer installment, so it shouldn't be too hard to follow.

Download *Temp9700A.bas* to your breadboard setup. Assuming you are in a typical indoor environment, you should see a Celsius temperature reading in the low 20s and a Fahrenheit reading around 70 degrees. If not, you're in for a little troubleshooting.

For the time being, that's it for our discussion of the MCP9700A but I do have a little more to say about battery-powered data collection systems. So, read on.

CUTTING THE CORD

When I first started thinking about data collection systems, I was considering a line-powered project with battery backup. However, I soon realized that my choice was based on a fear of the unknown. Back then,

■ FIGURE 5. Operating voltages for selected components.

	Operating Voltage Min.	Max.
DS3231 Real Time Clock	2.3V	5.5V
24LC256 Serial EEPROM	2.5V	5.5V
24FC256 Serial EEPROM	1.7V	5.5V
FM24W256 Serial F-RAM	2.7V	5.5V
MCP9700A Active Thermistor	2.3V	5.5V
DS18B20 Digital Thermometer	3.0V	5.5V

Cell Size	Capacity
AAA	1200mAh
AA	2700mAh
C	8000mAh
D	12000mAh

■ FIGURE 6. The mAh capacities for common sizes of alkaline batteries.

PICAXE projects could easily be powered by a three-cell 4.5V battery pack, but a regulated supply just seem simpler and I never got around to doing much experimentation with battery power.

In addition, many components required approximately +5V to operate reliably, so a three-cell battery pack wouldn't last very long before its voltage was too low to be useful.

Now, however, the situation has changed dramatically. All M2-class processors can operate with a supply voltage as low as 1.8V, and many other components can also function with a 3.0V supply — or even lower. **Figure 5** lists the operating voltage range for some components that I'm currently considering for a data collection project.

You probably noticed that I didn't include the DS1307 real time clock chip in the list. I realize it's a popular component, but its minimum operating voltage is 4.5V and it's a fairly inaccurate clock chip anyway, so I ruled it out.

As you can see, all the components that I'm considering can operate with a supply voltage anywhere from 5.5V down to 3.0V. New 1.5V alkaline batteries are actually charged to a level of about 1.6V, and are considered to be "exhausted" at 1.0V which is about 2/3 of their rated voltage.

Therefore, the functional voltage range of a typical three-cell alkaline battery pack is approximately 4.8V down to a level of 3.0V. This makes a three-cell alkaline battery pack a very suitable power source for a stand-alone data collection system. The only question is: Which size cells should we choose?

Figure 6 presents the usable capacity in milliamp hours (mAh) of the common sizes of 1.5V alkaline batteries. For example, the capacity of 12,000 mAh listed for D-cell batteries theoretically means that an alkaline D-cell can produce 12,000 mA for one hour, or 1 mA for 12,000 hours (or approximately 1.37 mA for one year) before the battery is depleted. However, the important word here is "theoretical." In the real world — many factors, such as rate of discharge and ambient temperature — can significantly affect the capacity of any battery.

All the components that I'm considering can operate with a supply voltage anywhere from 5.5V down to 3.0V.

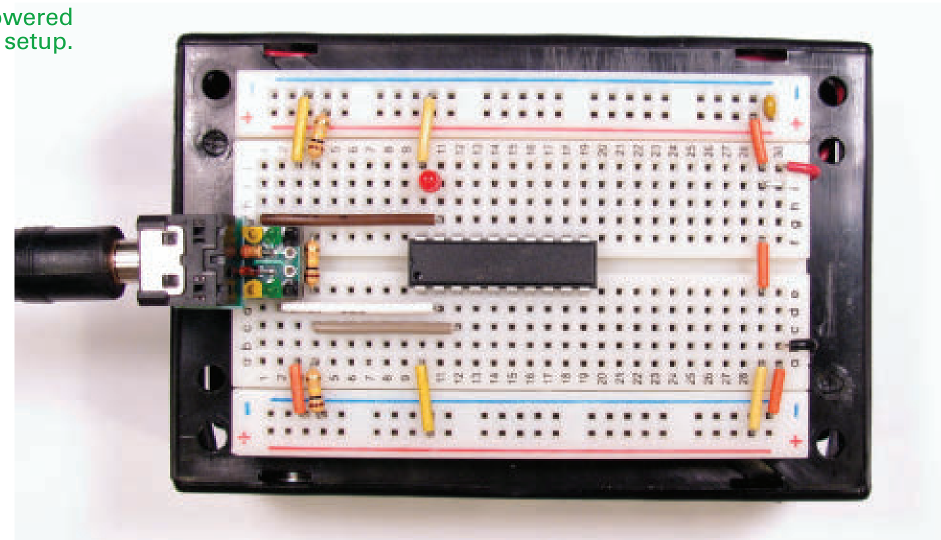
To complicate matters even further, it would be extremely difficult (if not impossible) to determine the average current draw of any data collection system without actually testing the system. To begin with, you would need to know the current requirement of every component in the system.

Of course, that information should be in the relevant datasheets, but the really difficult part is the microprocessor. Its power consumption will change considerably, depending on what it's doing at the moment.

Rather than subject myself (and you!) to all the calculations involved, I've decided to take a very pragmatic approach. I've set up a very simple breadboard system that will operate 24/7 for the next few months, and will monitor the level of its own battery supply. During that time, I'll begin adding components to the system, and track the effect each one has on power consumption. My goal is to have a system that can function for a full year or more, and I'm hoping that after six months or so, I'll have a fairly good idea of whether my battery pack is up to the task.

In order to maximize the chances of success, I'm using a three D-cell battery pack which a 20M2 processor is monitoring via a voltage divider that consists of three 100K resistors; the complete setup is shown in **Figure 7**. The breadboard is adhered directly to the bottom of the inverted battery holder, and pin C.7 is connected to the voltage divider at a point that measures 1/3 of the real time battery pack voltage which means that the measured voltage will never be much greater than 1.5V (1,500 mV). So, this arrangement works well with the FVR setup that we discussed earlier.

You probably noticed that there is no power switch on the breadboard. The only time I intend to turn off the power is when I am adding components to the system, so I didn't include a power switch. When I do need to power down the



■ FIGURE 7. Battery-powered breadboard setup.

system, I'll just remove the silver jumper that you can barely see in the upper-right corner of the breadboard.

In case you may want to conduct some experiments of your own, I've included a copy of the program that I'm currently using in the downloads (*BattMon20M2.bas*). It doesn't do much at all; the main *do/loop* simply measures the battery voltage, sends the results to the terminal window, and then puts the 20M2 to sleep for about seven seconds. The LED that's attached to the *serout* line flickers briefly during each serial transmission to the terminal window. It definitely draws much more current than the voltage divider does, even though the resistors are powered all the time.

If you do the math, you'll see that the voltage divider only draws about 16 µA; if there were no other power drain on the battery pack, it would theoretically last for more than 85 years! That's one good reason why you can't always trust theory — especially where batteries are involved. I guess we'll just have to wait to gather some reality-based data.

Finally, in case you're wondering, most of the time the system will not be connected to my Mac, but there's no harm in it continuing to send the data. Whenever I do connect the system to my Mac, all I need to do is run the Programming Editor and open the terminal window; it will start to receive the data.

When I started the "system" running, the initial battery voltage was 4.81V. Now (exactly one week later), it's 4.80V. As soon as I have the time, I plan to begin adding components in order to make the battery measurements a little more realistic. In the next installment of the Primer, I'll let you know how it's going.

ONE LAST THING

As usual, the MCP9700A will be available on my website (**www.jrhackett.net**). However, during the past few years we have implemented so many projects that I have become somewhat overrun with parts in my basement work area.

As a result, I've decided to make a change.

I will continue to carry the parts for each new Primer project, but when my stock of a specific part is depleted, I will not longer re-order them.

Of course, I'll continue to carry the parts we use in many different projects (e.g., headers, LEDs, resistors, stripboards, etc., etc.), but the more project-specific parts (e.g., the MCP9700A) will now be available on a first-come first-serve basis.

When my supply of one of these parts is depleted, it will be discontinued.

I apologize for any inconvenience my new system may cause, but I really have run out of space! **NV**

QUESTIONS & ANSWERS

WITH RUSSELL KINCAID

In this column, Russ answers questions about all aspects of electronics, including computer hardware, software, circuits, electronic theory, troubleshooting, and anything else of interest to the hobbyist. Feel free to participate with your questions, comments, or suggestions. **Send all questions and comments to: Q&A@nutsvolts.com**

BURNED PARTS ON WEED WHIP CHARGER

Q I am not sure why the charging circuit on my weed charger fried (guessing battery went bad as it will no longer hold a charge), but the parts are not recognizable. The resulting fire appears to be localized to the transistor, and melted adjacent parts. I would like to fix the circuit board. My questions are:

- What would be a reasonable guess for the replacement of the TO92 transistor (or a better upgrade) and pass diode?
- Why do other charging circuits typically heatsink the transistor?
- What is the value of what looks to be a zener diode?

The schematic is shown in **Figure 1**.

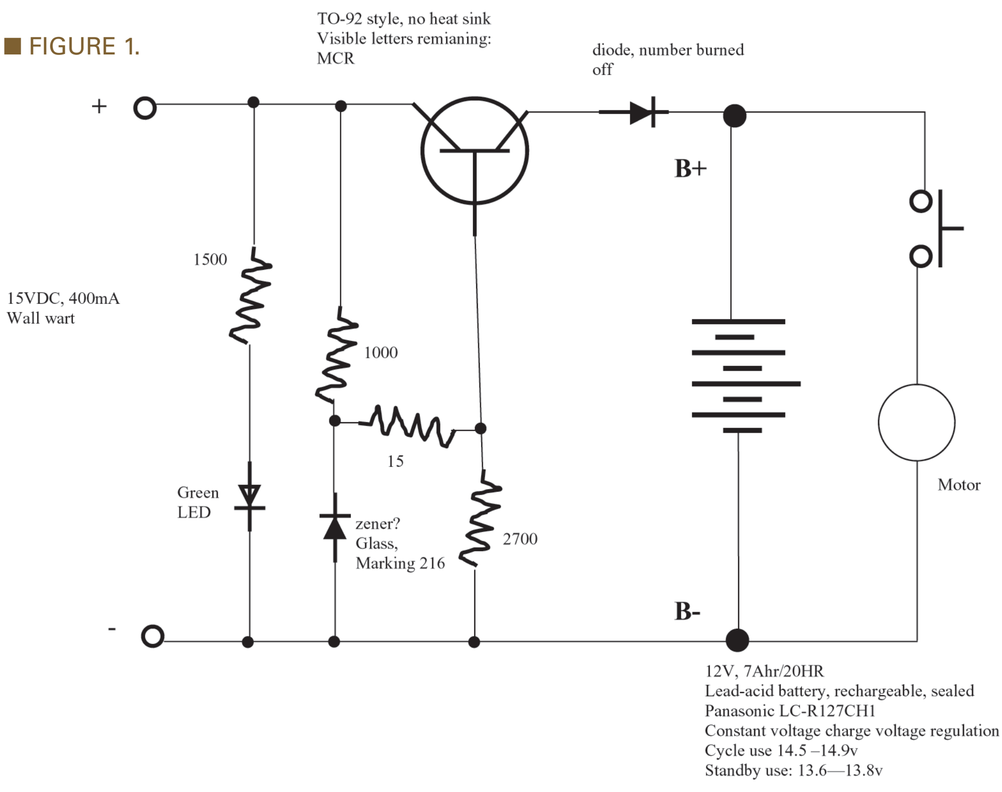
— Dwayne Stenlund

A Starting from the battery which is 13.8 VDC at full charge and adding .5V for the diode plus another .5V for the transistor Vbe (assuming it is an NPN), the base voltage is 14.8 VDC. I compute the current through the 2,700 ohm resistor to be 5.5 mA. If the battery is being charged at a 20 hour rate, the current is 7 Ah/20 = 350 mA. Assuming that the transistor current gain is 100, the base current is 3.5 mA. The current through the 15 ohms is at least 8 mA which puts the zener voltage at 14.98 volts; call it 15V. The drop across 1,000 ohms is at least eight volts, so the input should be 15 + 8 = 23 volts. The power dissipation is .35A*(23-14.3) = 3 watts — too much for a TO92.

A metal can transistor (such as a 2N2219) will have better heat dissipation and the diode should be a DO41 (such as a 1N4004).

There is no current limit in the circuit, except the gain of the transistor reduces at high current. When charging a dead battery, the current could approach one amp. Fortunately, the internal resistance of the source will reduce the input voltage — let's say it is 15 volts — then the power dissipation is (15-11)*1 = 4 watts. That is too much for even a heatsunk TO92, so this is a marginal design at best.

FIGURE 1.

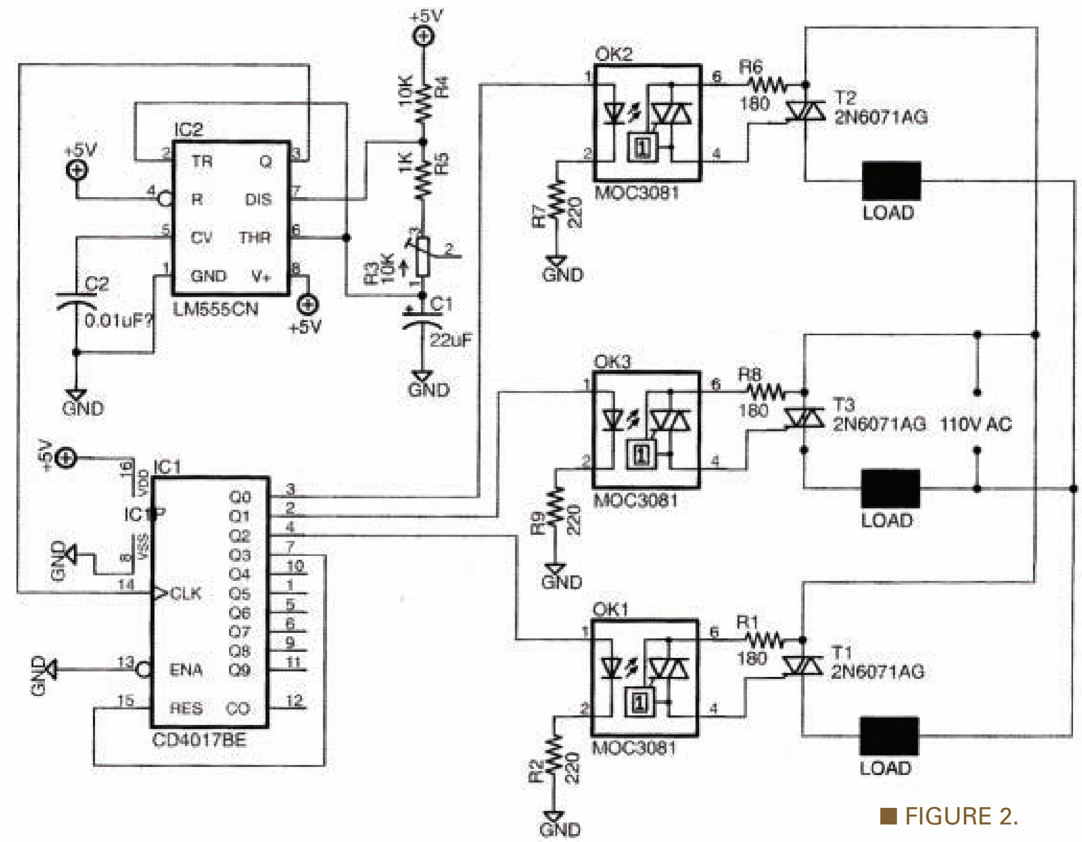


LIGHT CHASER CIRCUIT

Q After searching the Internet, I built and tested the chaser light circuit in **Figure 2**.

I want to replace an electro-mechanical unit with this circuit on my outdoor pole sign. The sign has 76 LED bulbs (120V < 3W each), so max 26 bulbs on at a time. Do I need to add a resistor between the triac gate and MT2 terminal? Do I need a snubber network of 39 ohm res and 0.01 µF across MT1 and MT2? Can you suggest any improvement in the circuit? Do you think there will be a problem with EMR (FCC)?

— Ankur Bhakta



■ FIGURE 2.

A There is already 180 ohms in series; I assume that is adequate. At any rate, you don't need a second resistor. The load is non-inductive, so a snubber is not needed but it might reduce EMR. The current is low and the photo-triac module switches at zero crossings, so if the wires to the bulbs are twisted, EMR should not be a problem. Adding a power line filter at the input will reduce conducted EMI on the power line.

MODEL TRAIN SPEED CONTROLLER

Q Could you publish a circuit for an electric train speed controller? The input from a transformer is 20 volts AC. The controlled motors run on AC at zero to 20 volts at up to five amps.

— Al Rothman

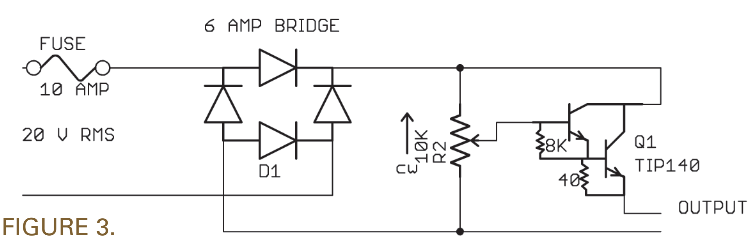
A Model trains use universal motors and will run on AC or DC. To stay with AC, you could use a variac variable transformer but a five amp unit is expensive. A light dimmer could work, but they are not usually rated for so much current. I favor DC power because it is quieter (less EMI) and cheaper. A simple circuit is shown in **Figure 3**.

The speed and current draw is determined by the back EMF of the motor; if the motor stalls, a fuse could be blown. However, I believe that model train transformers are impedance protected; that is, if you short the output, it will not produce enough current to burn up — at least not right away. Even so, you should have a fuse.

WIRELESS LISTENING DEVICE

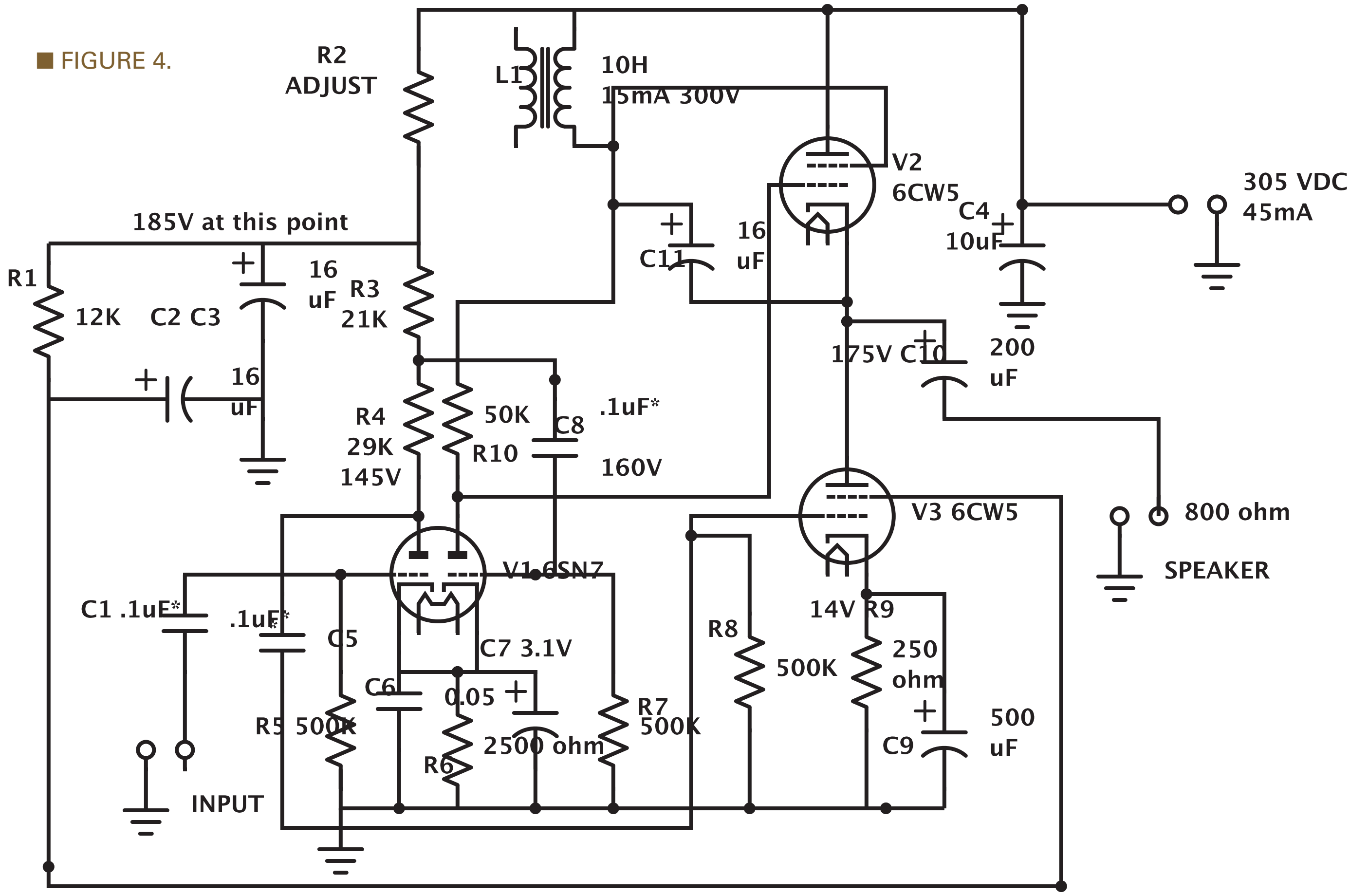
Q I have significant hearing loss. To aid in listening to the TV, I use a SurfLink Media device that plugs into the audio output of the TV and transmits a signal throughout the room. For power, it uses a wall wart that outputs 5V DC at 1.0 amps.

I would like to use this device in my work shop to listen to the radio.



■ FIGURE 3.

■ FIGURE 4.



However, due to fluorescent lighting, battery chargers, electric motors, etc., there is too much interference to use the wall wart. I would like to use a voltage regulator circuit powered by two 9V batteries in parallel.

Have you any suggestions for such a circuit? Any assistance is greatly appreciated.

— Dale P. Appleton

A I assure you the wall wart is not the problem. It appears to me that the SurfLink Media is using amplitude modulation which is susceptible to interference. I don't understand why they are not using FM. An alternative would be RadioShack wireless headphones which will allow you to listen to FM radio.

Also, you can buy a Ramsey FM transmitter kit to connect to your audio system and transmit to the RadioShack headphones.

AN OTL AMPLIFIER USING TUBES

Q I have been experimenting with the Output Transformer-Less (OTL) tube circuit in **Figure 4**.

The output — without measurable distortion — is six watts (70 VRMS into 800 ohms). The frequency response is flat from 17 Hz to 50 kHz; dropping by 3 dB at 250 kHz. The output impedance is about 20 ohms, so it is able to drive an 800 ohm load.

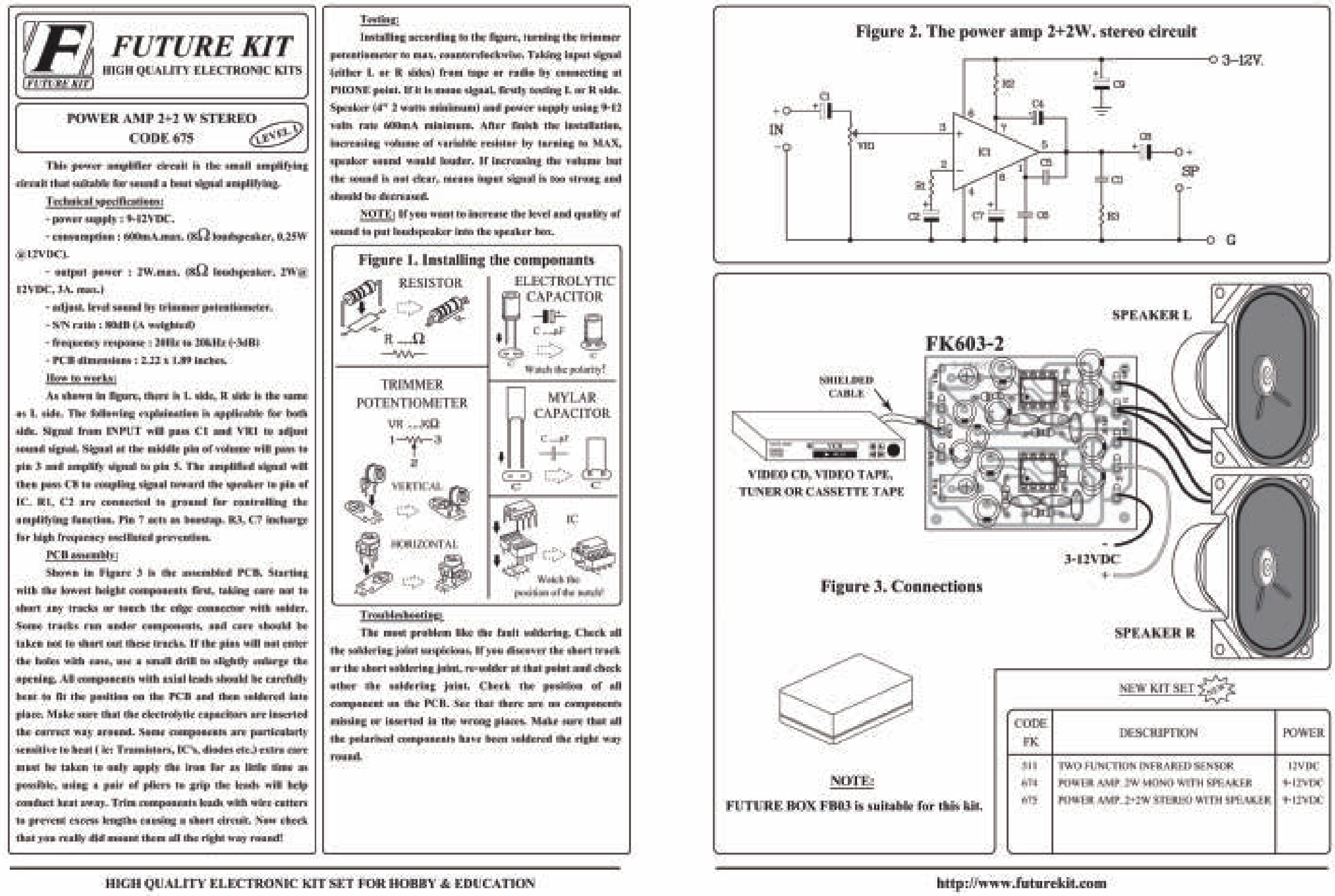
The two output tubes are essentially in push-pull, so the grids have to be driven at 180 phase. An ordinary phase splitter can't be used because the cathode of V2 has the output signal on it, and therefore the grid signal must be greater than for V3. In order to make the screen voltage more constant with respect to the cathode, the screen voltage is supplied by a 10 Henry choke which is bypassed with a 16 µF capacitor to the cathode.

A resistor could be used in place of the 10 Henry choke, but the reduced voltage would reduce the available output power. V1 is a voltage amplifier; the plate load is tapped to provide a signal of correct amplitude to the second half of V1 (6SN7) which is a phase inverter.

The capacitors, C1, C5, and C8, must be of high quality so that leakage does not throw off the biases which are marked with asterisks. I will be interested in your comments.

— Kendrick Sellen

A I am glad you sent this in because I had never heard of an OTL tube circuit. It seems that it is the new fad among audiophiles. I see that most circuits use multiple parallel output



FUTURE KIT
HIGH QUALITY ELECTRONIC KITS

POWER AMP 2+2 W STEREO
CODE 675

This power amplifier circuit is the small amplifying circuit that suitable for sound a bout signal amplifying.

Technical specifications:

- power supply : 9-12VDC.
- consumption : 600mA.max. (8Ω loudspeaker, 0.25W @12VDC).
- output power : 2W.max. (8Ω loudspeaker, 2W@ 12VDC, 3A. max.)
- adjust. level sound by trimmer potentiometer.
- S/N ratio : 80dB (A weighted)
- frequency response : 20Hz to 20kHz (-3dB)
- PCB dimensions : 2.22 x 1.89 inches.

How to works:

As shown in figure, there is L side, R side is the same as L side. The following explaination is applicable for both side. Signal from INPUT will pass C1 and VR1 to adjust sound signal. Signal at the middle pin of volume will pass to pin 3 and amplify signal to pin 5. The amplified signal will then pass C8 to coupling signal toward the speaker to pin of IC. R1, C2 are connected to ground for controlling the amplifying function. Pin 7 acts as bootstap. R3, C7 incharge for high frequency oscillated prevention.

PCB assembly:

Shown in Figure 3 is the assembled PCB. Starting with the lowest height components first, taking care not to short any tracks or touch the edge connector with solder. Some tracks run under components, and care should be taken not to short out these tracks. If the pins will not enter the holes with ease, use a small drill to slightly enlarge the opening. All components with axial leads should be carefully bent to fit the position on the PCB and then soldered into place. Make sure that the electrolytic capacitors are inserted the correct way around. Some components are particularly sensitive to heat (ie: Transistors, IC's, diodes etc.) extra care must be taken to only apply the iron for as little time as possible, using a pair of pliers to grip the leads will help conduct heat away. Trim components leads with wire cutters to prevent excess lengths causing a short circuit. Now check that you really did mount them all the right way round!

Testing:

Installing according to the figure, turning the trimmer potentiometer to max. counterclockwise. Taking input signal (either L or R sides) from tape or radio by connecting at PHONE point. If it is mono signal, firstly testing L or R side. Speaker (4" 2 watts minimum) and power supply using 9-12 volts rate 600mA minimum. After finish the installation, increasing volume of variable resistor by turning to MAX, speaker sound would louder. If increasing the volume but the sound is not clear, means input signal is too strong and should be decreased.

NOTE: If you want to increase the level and quality of sound to put loudspeaker into the speaker box.

Figure 1. Installing the componants

Troubleshooting:

The most problem like the fault soldering. Check all the soldering joint suspicious. If you discover the short track or the short soldering joint, re-solder at that point and check other the soldering joint. Check the position of all component on the PCB. See that there are no components missing or inserted in the wrong places. Make sure that all the polarised components have been soldered the right way round.

Figure 2. The power amp 2+2W. stereo circuit

Figure 3. Connections

NOTE:
FUTURE BOX FB03 is suitable for this kit.

NEW KIT SET

CODE FK	DESCRIPTION	POWER
511	TWO FUNCTION INFRARED SENSOR	12VDC
674	POWER AMP. 2W MONO WITH SPEAKER	9-12VDC
675	POWER AMP. 2+2W STEREO WITH SPEAKER	9-12VDC

HIGH QUALITY ELECTRONIC KIT SET FOR HOBBY & EDUCATION

http://www.futurekit.com

■ FIGURE 5.

tubes to drive eight ohm speakers, but your circuit drives an 800 ohm speaker. Where do you find a high impedance speaker? The 10 Henry choke allows the screen voltage to be higher than the supply at higher frequencies, but does little at 30 Hz. It seems to work, though. I would like to see some feedback, but the input and output are in phase so you would need to add an inverter; perhaps a one stage preamp.

PARABOLIC EAR

Q I got my *Nuts & Volts* magazine and I read your reply (germanium to silicon transistor) to my question. THANK YOU! It's so easy to modify a circuit ... when you know how!

I have another question. I found the two watt stereo amplifier kit in **Figure 5** to which I'd like to add an ultra-sensitive electret microphone (preferably from Mouser). However, I don't know how and which one would work better. The microphone will be mounted right in the focal point of a large metal bowl.

Can you please modify this amplifier (#FK675/$15.95 from **qkits.com**) to which I'll be adding (instead of speakers at one end) a set of headphones and — to the other end — the ultra-sensitive electret microphone?

I got this idea from a 'Big Ear' Soundscope introduced by a science teacher a long time ago.

— Nate

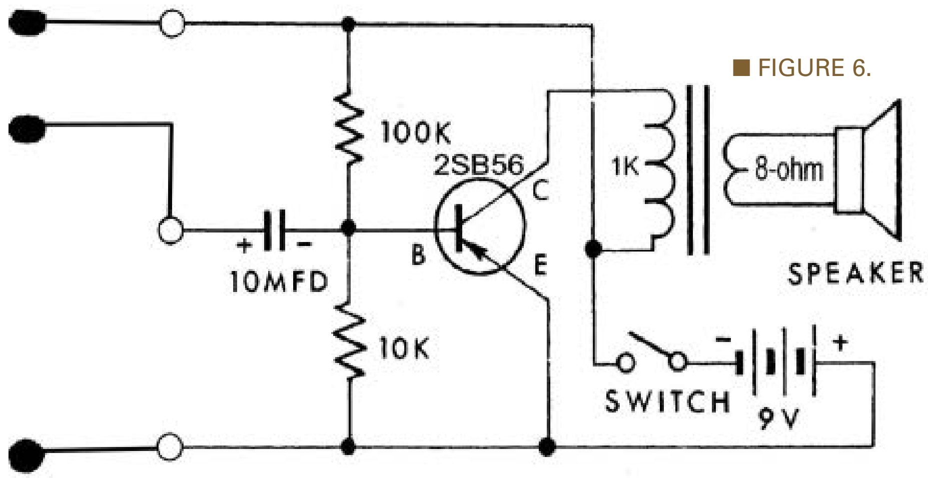
A Any of the Mouser electret microphones could work, but #665-POM5238PR is the most sensitive. Electret microphones are high impedance and don't draw any current, so the fact that these microphones do draw current and have a defined impedance tells me that there is internal circuitry.

To operate the microphone, you need a voltage source of no more than 10 volts and a resistor that — with 0.5 mA — will leave at least two volts at the microphone.

In this case, with a nine volt battery (which is seven volts at end of life), the resistor is 14K. The next lower standard value is 12K, or use 10K if the amplifier will work at less than seven volts.

The output of the microphone is less than 15 mV. The output of the amplifier at two watts into eight ohms is 4 VRMS, so the gain has to be at least 4,000 mV/15 mV = 267, which is 48.5 dB.

The IC is a TBA820; I don't know what the gain is, but if you change R1 to 22 ohms, the gain will be about 48 dB. The second half of the stereo amp is not needed.



■ FIGURE 6.

SHORT-WAVE RADIO SECOND STAGE

Q I believe you missed modifying the two amplifiers I originally attached to my email you answered in the October 12 issue.

So, I'll opt for just one modified amplifier. What changes would you also make to the amplifier in **Figure 6** so that another silicon transistor could be used instead of the 2SB56, in order to work with the transistor changes you suggested for the regenerative SW radio?

— Nate

A This is a terrible design; there is no control over the collector current. Every time you build it, the value of the 10K resistor will have to be adjusted.

A 2N2907 PNP silicon transistor can be dropped right in. Replace the 10K resistor with a 25K rheostat, and adjust with a 1 kHz signal such that the output signal is rail to rail without clipping. If you want to use an NPN, 2N2222 will work. **NV**

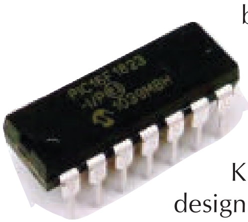
Remember! Send any questions and/or comments to: Q&A@nutsvolts.com

NEW PRODUCTS

- HARDWARE
- SOFTWARE
- GADGETS
- TOOLS

KEYBOARD-TO-ASCII CONVERTER CHIP

Lucid Technologies announces the availability of the KB1 — a PS/2 keyboard-to-ASCII converter chip for embedded microcontroller applications. The easy to use KB1 is based on the PIC16F1823 extended mid-range processor. The KB1 provides design flexibility with its pin selectable choice of UART, SPI, or I^2C interface modes. It has a seven-character buffer and generates interrupts in SPI and I^2C modes.



The KB1 is available as a 14-pin DIP. The data manual is available online and includes example PIC assembly language routines to communicate with the KB1 in SPI and I^2C modes. Pricing for the KB1 is $6.25 for one.

For more information, contact:
Lucid Technologies
www.lucidtechnologies.info

ONE-TRIP TEMPERATURE DATA LOGGER

The compact and easy-to-use disposable PicoLite USB data logger from Lemos International Co. provides a simple solution for temperature monitoring throughout

mass transportation cold chain processes.

Featuring a built-in temperature sensor, activation button, bi-color LED, and direct USB connectivity, the PicoLite offers a plug and record solution for various applications. The PicoLite is configured by Fourtec's intuitive DataSuite software, offering comprehensive data analysis features and alarm notifications.

Other features include:

- Low cost, one-trip, disposable logger for an unbroken cold chain.
- Internal, high accuracy temperature sensor.
- Unique Boomerang feature that automatically creates and emails a PDF data report when logger is connected to a PC.
- Direct USB interface for PC communication.
- Six month battery life at one minute sampling.
- 16,000 sample memory accommodating long distance transports.
- Software-enabled activation button to start/stop logging and mark time stamps.
- Water and dust resistant built-in bi-color LED for logging and immediate alarm indication.

For more information, contact:
Lemos International Co.
www.lemosint.com

TINY MULTIFUNCTIONAL LASER-BASED PARKING SENSOR

A-Elektronik Company unveiled the successor to their Guardian line of laser based driving aids with their AL Priority sensors.

These sensors have evolved and are the smallest multifunctional laser-based car sensors currently available. Besides being the smallest, they perform with unsurpassed laser sensitivity, guaranteed by the manufacturer's patented technology (US20100264301).

While the sensor is smaller, the control box has gotten bigger, offering new functions and add-ons. The system can consist of up to five outer sensors. It differentiates front and rear sensors, making it more user friendly. Cars with factory-built laser cruise control systems are no longer a problem since the AL Priority can use a special sensor that will enable keeping both the laser cruise control and AL parking aid.

AL Priority can be firmware upgraded and have a custom setup via a USB Flash drive. Using the AL setup application installed on a home computer allows users to create preferred setups on the Flash drive. To upload them to the AL Priority unit, all that is needed is to plug the USB drive into the AL Priority USB socket. The same method is used for Internet available firmware upgrades.

AL Priority works on the 905 nm light frequency. It constantly emits laser signals and recognizes them when reflected from an obstacle, consequently warning the user. Thanks to the advanced program code, the system will differentiate the laser signals coming from other laser sources. A special fog lights version of sensors is also available that incorporate LED illumination intended to be used in reduced visibility.

For more information, contact:
A-Elektronik
www.a-elektronik.hr

ANDROID™ APP FOR REMOTE CLAMP METER READINGS

Extech Instruments announces the release of their new industrial app, "EX845 METERLiNK for Android." The new app allows electrical contractors and plant maintenance professionals who use Extech's EX845 clamp meter to

remotely view and display readings from the meter on an Android phone or tablet up 30 feet away.

The new industrial app displays large, easy-to-read values for the following functions: AC/DC voltage, AC/DC current, capacitance, resistance, frequency, diode test, and temperature (both from a Type-K thermistor probe and from the meter's built-in non-contact, laser

SHOWCASE

infrared thermometer). Function indicators highlight which measurement mode is selected on the meter. Additionally, data hold and min/max functions are also remotely displayed. The METERLiNK app features a log of recent readings for quick reference, basic trending, and more context.

For more information, contact:
Extech Instruments
www.extech.com/instruments

NEW VERSION OF DIPTRACE PCB DESIGN SOFTWARE

Novarm Limited announces the availability of DipTrace 2.3 — a fully-functional package with significant improvements. Version 2.3 features Online Design Rules Check; VRML 2.0 3D export; custom non-signal layers; as well as advances in hierarchy, file compatibility, working speed, and more.

DipTrace is bundled with four modules: schematic, PCB layout with high speed shape-based autorouter, pattern editor, and a component editor.

Other features include:

- Easy-to-learn user interface.
- Multi-sheet and multi-level hierarchy.
- Comfortable manual routing.
- Multiple verifications with online DRC.
- Smart project structure.
- Wide import/export opportunities.
- Manufacturing output.
- Real-time 3D preview with export.

DipTrace 2.3 runs on XP/Vista, Windows 7/8 (32- and 64-bit platforms), and MacOS and Linux. Current DipTrace users get the 2.3 version for free.

For more information, contact:
Novarm Limited
www.novarm.com

PIC TRAINING AND DEVELOPMENT BOARD

Gooligum Electronics has released a PIC training and development board, specially designed to complement its existing PIC programming tutorials. The tutorials use C and assembler to introduce the baseline and mid-range PIC families, starting with easy-to-master baseline devices such as the 10F200, and eventually moving on to topics such as displaying analog signals on multiplexed seven-segment displays or using mid-range PICs such as the 16F684. Each topic is illustrated via detailed and fully explained examples.

The training board comes with all 40+ lessons (including source code and project files) on CD and every component — including five PIC devices — needed for each example.

It provides an ICSP connector for easy connection to Microchip's popular PICkit 2 or PICkit 3 programmers (not supplied).

Besides including LEDs, pushbutton switches, seven-segment displays, a piezo speaker, and a number of oscillator options on board, every PIC pin is brought out to a 16-pin expansion header alongside a solderless breadboard making the training board useful as a general development board for small (six-to14-pin) PICs, even after the course is finished.

Cost is $89 fully assembled, or it is available in kit form for $69. Shipping from Australia is $10.

For more information, contact:
Gooligum Electronics
www.gooligum.com.au

CHOOSE YOUR OWN ADVENTURE GAME

By Nuno Alves

An Entertainment System for the Daily Commuter

One of the things that bores me the most is driving. Ten minutes into my daily commute and the urge to check my email and tweeter account grows stronger and stronger. Being a law abiding citizen, I needed to find a way to stave off my boredom. The other day as I was driving home, I envisioned a car-based audio gaming system which does not require the driver to take his eyes off the road. Immediately, many soundbased games came into mind: casino games such as blackjack, "choose your own adventure" gamebooks, and even role playing games (RPG). The underlying principle for each game was similar. A computer synthesized voice would provide some choice or description, and the commuter choices would be executed through a pushbutton that could be placed on the steering wheel. So, why not voice activated? I have a thick accent!

In this article, I will describe my implementation of a "choose your own adventure" game. Since my proposed system is so modular, the diligent reader can easily implement any other type of game. In any case, those born in the late '70s or early '80s will instantly recognize the "choose your own adventure" genre.

These are a series of gamebooks where each story is written from a second-person point of view, with the reader assuming the role of the protagonist and making choices that determine the main character's actions and the plot's outcome.

The choices can be as simple as deciding which direction to take at an intersection, or as complex as picking the weapon to slaughter a three-headed Komodo dragon that just boarded a spaceship.

The books are formatted such that, after a couple of pages of reading, the reader faces many options — each leading to one of the many distinct endings.

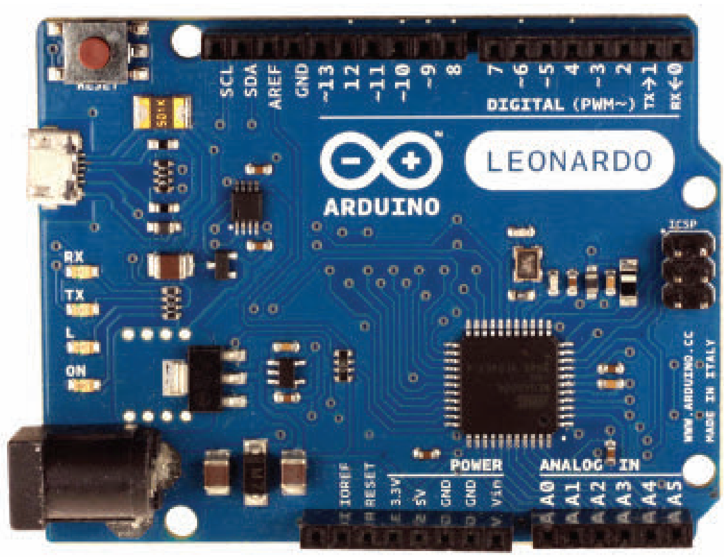
Needless to say, these "choose your own adventure" books were a staple of my childhood, and now that I am married I am proud to admit that at one point I owned over 100 of them.

Arduino Leonardo

Since I could not take another day of driving boredom, I quickly made a prototype of my system using an Arduino Leonardo, some pushbuttons, and a laptop computer with an audio cable connected straight into the AUX port of my car stereo. I call my system "Choose Your Own Destination."

The laptop computer loads the game engine, and an open source library will voice-synthesize any text or choices. The commuter can then send the game choices by pressing a particular pushbutton attached to the Arduino Leonardo microcontroller. Thanks to the latest Arduino features, the Leonardo can now act as a USB keyboard.

In my project, it sends a particular character to the laptop every time a button is pushed. In less than three hours, I was able to modify an off-the-shelf enclosure, wire everything, and develop the microcontroller code. Before going into the project details, it is necessary to learn a bit more about these new Arduino features.



■ **FIGURE 1.** Arduino Leonardo is the latest entry in the Arduino family, powered up by a ATmega32U4 microcontroller clocked at 16 MHz.

The Arduino (shown in **Figure 1**) is a popular open source single-board microcontroller. It contains 11 usable digital pins in which six can "fake" an analog signal through PWM. The Arduino also contains six analog input pins with 10-bit resolution. Since this microcontroller operates at a 5V level, a 10-bit resolution means that it can read analog data — for example, from a sensor — in $5/2^{10}$ = 4.9 mV steps. More details about the analog input capabilities of the Arduino can be found at **http://arduino.cc/playground/CourseWare/AnalogInput**.

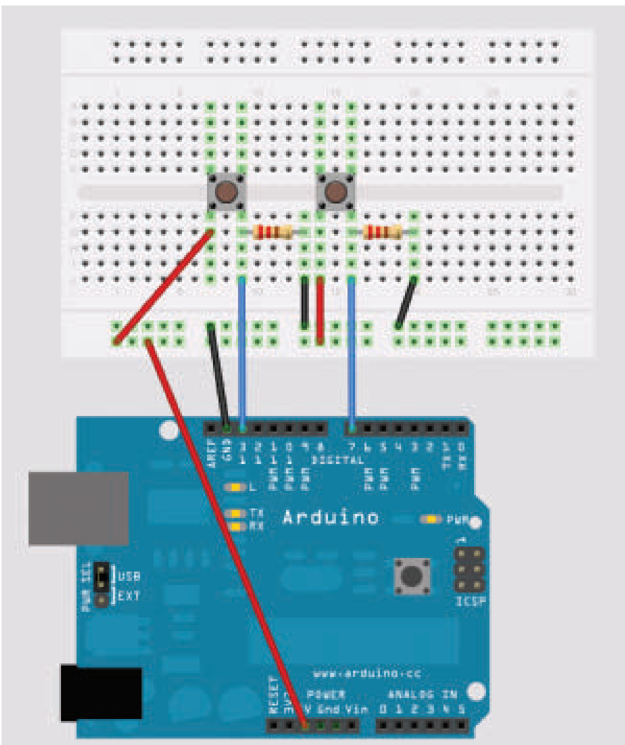
Developing applications for the Arduino prototyping system is fairly painless, thanks to the simplified C++ routines and a bare-bones IDE (integrated development environment).

Another defining element of the Arduino platform is the standardized "unusual" pin layout. This allows third parties and hackers to develop add-ons (shields) which expand the functionality of the system with things such as RF communications through Zigbee, Ethernet, or even GPS.

The latest revision of the Arduino system replaced the ATMega328P chip with a ATmega32U4. This new chip allows for serial (CDC) communication over USB, and appears as a virtual COM port to software on the computer. This makes the Leonardo appear as a generic (driverless) keyboard and mouse which can be programmed using some functions provided by the Arduino IDE package.

Leonardo as a Keyboard

Consider the circuit in **Figure 2** with two buttons: one connected to pin 13 and the other to pin 7. In this circuit, the Arduino digital input pin will read 5V (HIGH) if the button is pressed, and 0V (LOW) otherwise. Let's say you want to write something every time a button is pressed into Microsoft Word (which is installed on your computer).



■ **FIGURE 2.** Reading the status of two buttons: one is connected to pin 13 (button_A), and the other (button_B) to pin 7.

```
// Make sure to open a blank Notepad window on your computer

int button_A_Pin = 13; //connect button_A to pin #13
int button_B_Pin = 7; //connect button_B on pin #7

int counter = 0; //button push counter

void setup() {
  //make both pushButtons as inputs:
  pinMode(button_A_Pin, INPUT);
  pinMode(button_B_Pin, INPUT);
  //allow the Arduino to behave as a keyboard
  Keyboard.begin();
}

void loop() {

  //if the button A was pressed, type "Hello there!"
  if(digitalRead(button_A_Pin)==HIGH)
  {
    Keyboard.println("Hello there!");
    //a "quick and dirty" way to debounce the button is to
    //way 0.5 seconds after the Arduino noticed it was pushed.
    delay(500);
  }

  if(digitalRead(button_B_Pin)==HIGH)
  {
    Keyboard.print("Button B was pressed ");
    Keyboard.print(counter);
    Keyboard.println(" times...");
    //increment the button counter
    counter++;
    delay(500);
  }
```

■ **FIGURE 3.** If button_A (pin 13) is pressed, the Arduino will type "Hello there!" on the foreground application of the host computer. If button_B (pin 7) is pressed, then the number of times that the button was pressed will be typed.

```
int button_A_Pin = 13; //connect button_A to pin #13
int button_B_Pin = 7; //connect button_B on pin #7

void setup() {
  //make both pushButtons as inputs:
  pinMode(button_A_Pin, INPUT);
  pinMode(button_B_Pin, INPUT);
  //allow the Arduino to behave as a mouse
  Keyboard.begin();
}

void loop() {

  //if the button A was pressed, type "Hello there!"
  if(digitalRead(button_A_Pin)==HIGH)
  {
    Mouse.move(-7, 10, 0);
    //a "quick and dirty" way to debounce the button is to
    //way 0.5 seconds after the Arduino noticed it was pushed.
    delay(500);
  }

  if(digitalRead(button_B_Pin)==HIGH)
  {
    Mouse.press(MIDDLE);
    delay(500);
  }

}
```

■ **FIGURE 4.** Faking the movement and button presses of a USB mouse.

When the **Figure 3** code is sent to the Arduino, it will type "Hello there!" whenever button_A is pressed, while the other button types the number of button presses. Keep in mind that this program will send text to the application that is currently running in the foreground of your computer and not just Microsoft Word.

The class that makes this work is *keyboard*, and the two methods used in this example are *Keyboard.println()* and *Keyboard.print()*. These allow a variable value or string to be sent as if they were typed into the keyboard — with or without a line feed. Other useful methods — such as *Keyboard.write()* — are listed at **http://arduino.cc/en/Reference/MouseKeyboard**.

Leonardo as a Mouse

Mouse is another new class that allows the Arduino to control the position and the button presses of a mouse. Let's re-use the same circuit (**Figure 2**), but when button_A is pressed we want the mouse to move up (y direction) 10 units and left (x direction) seven units.

To make things more interesting, we want to emulate a press of the middle mouse button when button_B is pressed. The updated code (listed on **Figure 4**) is not that much different.

Anything you can do with a physical mouse can be programmed using the Arduino Leonardo. (Refer to the aforementioned link for some additional methods and examples.) Keep in mind that when using these functions, the Arduino will take over your computer.

Unless you are a legendary programmer (like John Carmack), save yourself some heartache and log out from your web-based bank account and close all your work applications before you let the Arduino loose in your system.

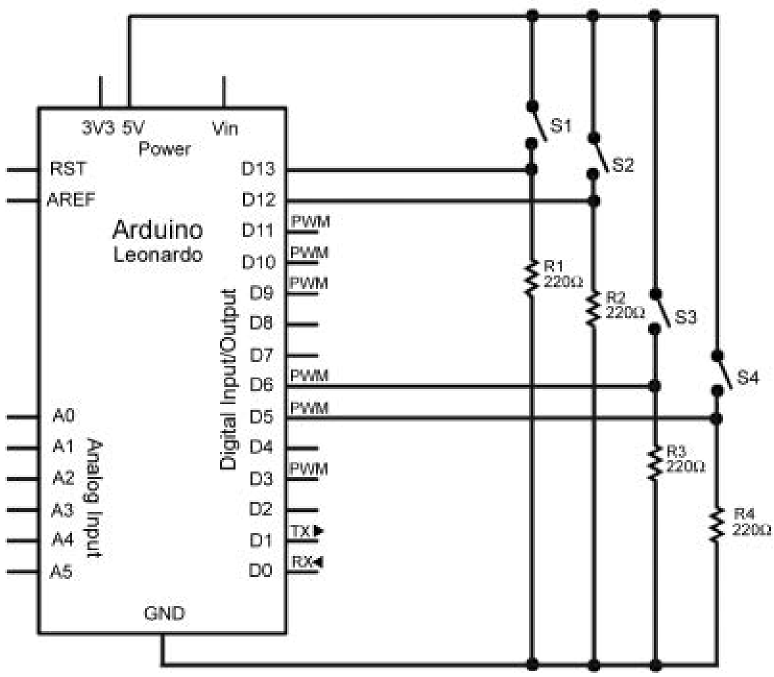
Choose Your Own Adventure/ Destination

My implementation of the "Choose Your Own Adventure" game is even simpler than the previous examples. Since there are never more than four choices in adventure books, I limited my design to four buttons. As shown in **Figure 5**, these buttons are really the extent of my circuit design. My prototype enclosure (shown in the **photos** at the end of the article) was a slightly modified wooden box purchased at Michaels, and the four buttons were ordered from Adafruit.

Every time a commuter pushes a button, the appropriate button identifier is sent to a computer with the **Figure 6** code.

The final step requires you to connect the prototype box into a computer which has the game engine running. For this article, I developed a small Python program (cyod.py; **www.nunoalves.com/source/cyod.zip**) that is the game master for all your games. This program reads a text file with the story together with all possible decisions.

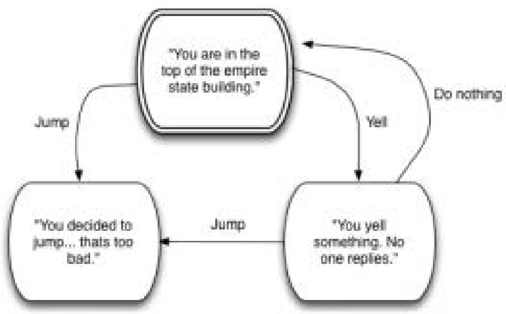
You can create as many as you like, since they will be read on-the-fly through a speech synthesizer. The Mac OS X already has a built-in speech synthesizer, whereas if you use Windows you will have to download an external program such as the open source eSpeak



■ **FIGURE 5.** A circuit with four pushbuttons that will be used to control the flow of the games.

```
int pushButton[4]={13,12,5,6};

void setup() {
  //make each pushbutton pin an input
  for (int i=0;i<4;i++)
    pinMode(pushButton[i], INPUT);
}

void loop() {
  //array with the status of button presses
  int buttonState[4];
  for (int i=0 ; i<4 ; i++)
    buttonState[i] = digitalRead(pushButton[i]);

  if (buttonState[0]==HIGH) { Keyboard.print("1\n"); }
  if (buttonState[1]==HIGH) { Keyboard.print("2\n"); }
  if (buttonState[2]==HIGH) { Keyboard.print("3\n"); }
  if (buttonState[3]==HIGH) { Keyboard.print("4\n"); }

  delay(100); //de-bouncing the push-button
}
```

■ **FIGURE 6.** Code that sends choices back to the computer.

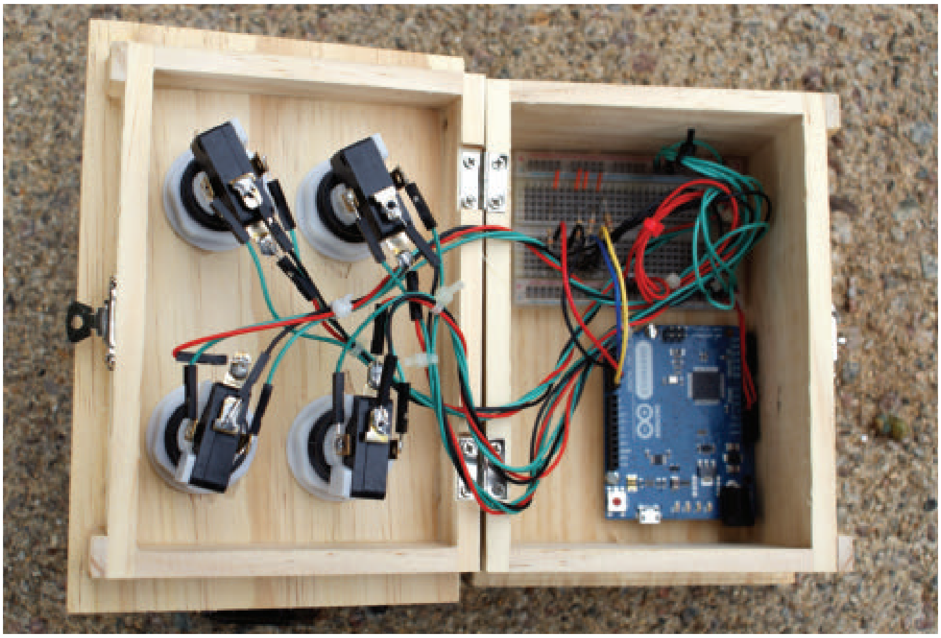


■ **FIGURE 7.** The box that will be used to pick our decisions.

PARTS LIST

QTY	COMPONENT NAME	PRICE*	LINK
1	Arduino Leonardo ATmega32u4 with headers	$24.95	www.adafruit.com/products/849
1	USB cable - A/MiniB - three feet	$4.00	www.adafruit.com/products/260
1	Half-size breadboard	$5.00	www.adafruit.com/products/64
1	Premium male/male jumper wires - 40 x 6"	$7.95	www.adafruit.com/products/758
1	Tactile switch (6 mm) x 10 pack (buttons)	$2.50	www.adafruit.com/products/367
4	220 ohm resistors, 1/2 watt	Varies	Any electronics store
4	LED illuminated pushbuttons - 30 mm round	$3.95	www.adafruit.com/products/492
1	Large plastic project enclosure (weatherproof with clear top)	$19.95	www.adafruit.com/products/905

*Prices may be subject to change.



■ **PHOTO 1.**

available at **http://espeak.sourceforge.net**.

Developing new stories is painless. As an example, consider the poignant tale from **Figure 7**. Initially, you are placed on top of the empire state building and you are only given two choices: jump or yell. Eventually you will have to jump. Sadness.

The following text file (test_story.txt) is the equivalent input to the Choose Your Own Destination game engine:

```
<1>
You are in the top of the
  empire state building.
[choices]
- Jump <2>
- Yell <3>
- Do nothing <1>
[end choices]
<2>
You decided to jump...
  thats too bad.
[end]
<3>
You yell something. No one
```

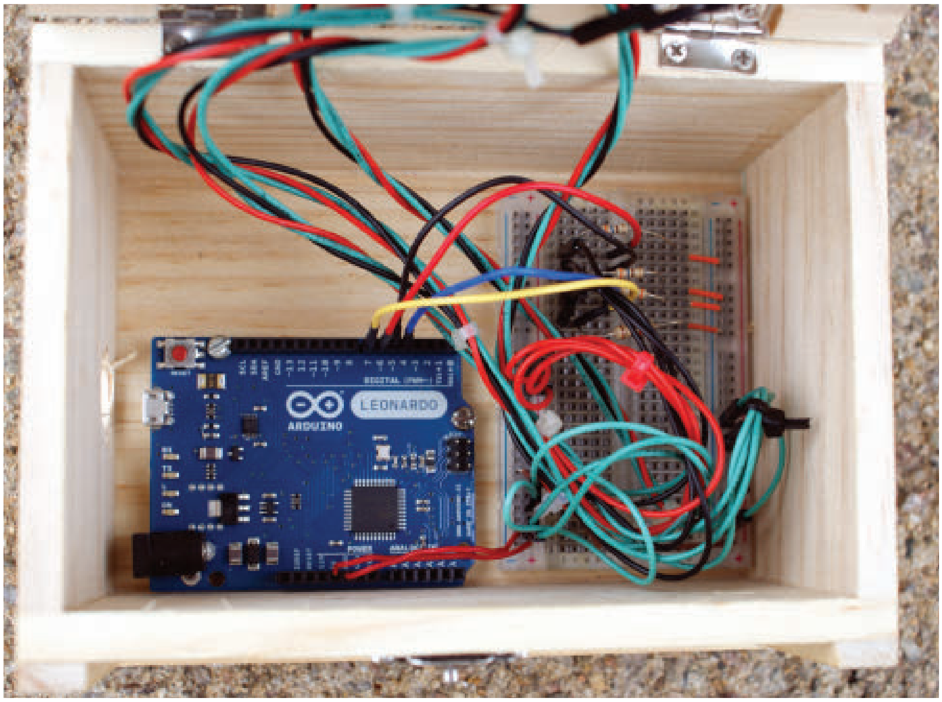
```
replies.
[choices]
- Do nothing <1>
- Jump <2>
[end choices]
```

To play this particular story, the commuter only has to run the Python script in the terminal with something like:

```
python cyod.py test_story.txt
```

Last Turn

Of course, you can replace the test_story.txt with your own adventure. Once the program is loaded and running, simply attach your Arduino Leonardo box to your computer and play. Happy gaming, and keep your eyes on the road! **NV**



■ PHOTO 2.

THE DESKTOP CONTEMPLATOR

By Craig A. Lindley

I didn't start out to build a personal light show for my desk, but in the end that is what happened. It all started when I saw a SainSmart 3.2" 320 x 240 TFT LCD display with built-in display controller, touch screen controller, and SD card interface for sale on Amazon for $16. I already had an Arduino Uno on hand, so connecting these two devices seemed like a natural thing to attempt. Having never connected an LCD display to a microcontroller before, I was anxious to do so. Finding a well written driver library (see Resources) for this display put the icing on the cake, so I got to work wiring things together. In an afternoon, I wired the LCD display to the Arduino, downloaded and installed the UTFT driver on my Mac, and compiled and ran the demos that came with the driver. I was amazed at how easy this came together, and I had the basis for my personal light show running in less than a day.

After seeing the demos, I started thinking about what else I could make this LCD display do. So, I started pulling out code I had written over the years and began porting it to the Arduino Uno. First off was the Mandelbrot set. While the code worked and the results were beautiful, it took a full five minutes to generate which was okay because I wasn't in a hurry. The long generation time was not too surprising because the Mandelbrot set requires a lot of floating point calculations which are time consuming on any eight-bit processor.

Equally as pretty and in general less processor-intensive are the calculations of Julia sets. So, this is what I tried next, and some of the images produced took my breath away. I was starting to think this hardware combination had some merit, so one thing lead to the next and pretty soon I had about 16 different display patterns running including: the Mandelbrot set, Julia sets, plasma patterns, numerous spirograph patterns, star burst patterns, concentric squares and circles, and much more.

Not being satisfied with static repeating imagery, I added a large dose of randomness into the pattern generation process, and with the addition of numerous

ELECTRONICS PARTS LIST

ITEM	NOTES
Arduino Uno board	Other Arduinos can be used but this will necessitate changes in the software.
.1" break-away male header pins	You need enough of these pins to populate all of the female I/O connectors on the Arduino Uno.
SainSmart 320x240 TFT LCD display	This is a 3.2" display, although other SainSmart displays based on the SSD1289 controller will probably work as well.
USB cable	The USB cable will become a permanent part of the Contemplator if you package it like I did.
USB charger	This is for running the Contemplator after programming is performed.
Hook-up wire	Light gauge flexible wire is best
27 ohm 1/2 watt resistor	This is required to power the LED back lights of the LCD display. See **Table 1**.

palettes (some of which also contain randomness) I was soon generating some very unique images.

It was then that it hit me. I should package this thing up as a desktop device and let it generate images ad infinitum. After all, the hardware cost less than $40, so why not.

Since having this device on my desk/bench, I have found it helps me think. When I get stuck on something I am working on, I glance over at the display for a few minutes taking my mind off of things which sometimes helps in finding a solution. For this reason, I've decided to call this device a "Desktop Contemplator."

The Hardware

Building a Contemplator requires the electronic components shown in the **Parts List**.

Once you have gathered the required components, wiring the Arduino Uno to the LCD display is easy though rather tedious due to the number of wires involved. A lot of wires are required because I used a 16-bit interface between the Arduino Uno and the LCD display. I chose this instead of an eight-bit or serial interface in the interest of performance.

A drawback to using the 16-bit interface with the Arduino Uno is that it uses up every available I/O pin. This means neither the touch screen component nor the SD card interface available on the LCD display's PCB can be used. Luckily, neither were necessary for this application.

All required connections between the Arduino Uno and the LCD display are shown in **Table 1** and the LCD display connector pinout is shown in **Figure 1**. Take your time when doing this wiring and double-check your work when you're finished before applying power.

The Contemplator is programmed and powered via the USB cable plugged into the Arduino Uno. Once programming is completed, a USB charger can be used to power the Contemplator.

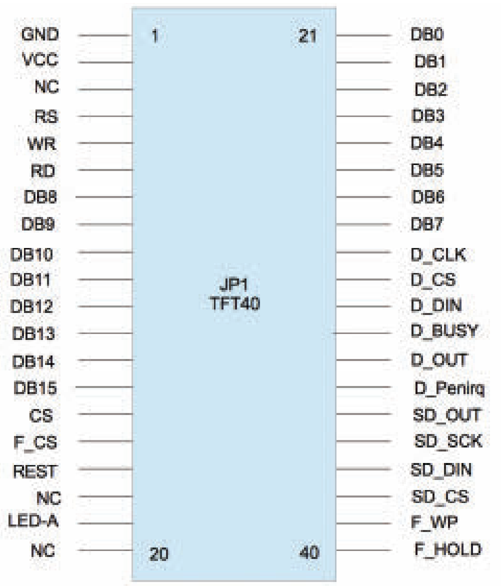
Optional Hardware

As I was writing this article, I discovered SainSmart also sells an assembly consisting of an Arduino Mega2560, a shield for connecting the LCD display to the

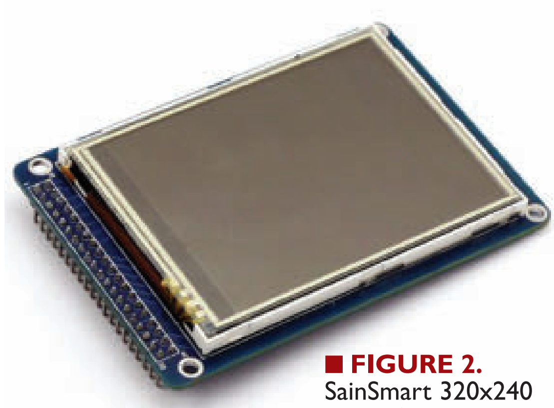
TABLE 1. LCD display to Arduino Uno connections for 16-bit data interface.

LCD Display Pin Number	LCD DisplaySignal Name	Arduino Uno Signal Name	Notes
21	DB0	D8	16-bit data
22	DB1	D9	16-bit data
23	DB2	D10	16-bit data
24	DB3	D11	16-bit data
25	DB4	D12	16-bit data
26	DB5	D13	16-bit data
27	DB6	A0	16-bit data
28	DB7	A1	16-bit data
7	DB8	D0	16-bit data
8	DB9	D1	16-bit data
9	DB10	D2	16-bit data
10	DB11	D3	16-bit data
11	DB12	D4	16-bit data
12	DB13	D5	16-bit data
13	DB14	D6	16-bit data
14	DB15	D7	16-bit data
4	RS	A5	Data/Command Selector
5	WR	A4	Write Enable (active low)
6	RD	5V	Read Enable RD (active low) tied to 5V
15	CS	A3	Chip Select (active low)
17	REST	A2	Chip Reset
19	LED-A	5V	LCD signal tied to 5V through 27 ohm resistor
1	GND	GND	Ground
2	VCC	5V	Power

Note: There are no I/O lines left over to connect the Arduino Uno to the touch screen controller or SD card interfaces, so these aspects of the circuitry are not available.

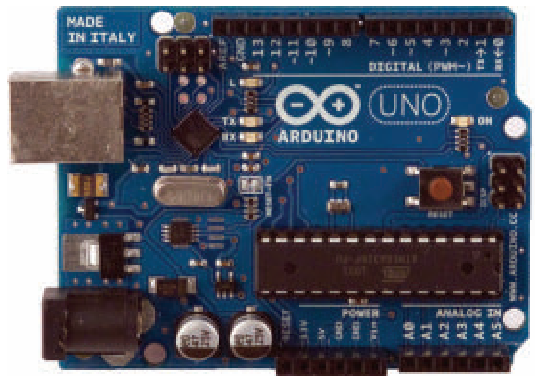


■ **FIGURE 1.** SainSmart 320x240 TFT LCD display connector pinout.



■ **FIGURE 2.** SainSmart 320x240 TFT LCD display.

Arduino, and the same LCD display we are using here. They officially call it the SainSmart Mega2560 Board+3.2 TFT LCD Module Display+Shield Kit for Atmel Atmega



■ **FIGURE 3.** An Arduino Uno.

AVR 16AU Atmega8U2 and it's available from their website or on eBay. If you were to buy this assembly at ~$55, you could use it without having to do any wiring at all. A minor software change (to be described shortly) is required, however, to run the Contemplator sketch on this hardware.

A major advantage to using this hardware is that there are many more I/O lines available which allow access to the touch screen controller and the SD card interface if these are important to your application. Additionally, the Mega2560 has four times the RAM (8K) and eight times the Flash (256K) which would allow many more display patterns to be developed.

The Software

The Contemplator requires two pieces of software for its operation. The first is the UTFT driver library for the LCD display and the second is the Arduino sketch I wrote called Contemplator.ino (see **Resources** for both).

Installing the UTFT Library

You can skip this section if you already know how to install an Arduino library. If you don't, read on.

1. Exit the Arduino IDE if you have it running.
2. Download the UTFT library (UTFT.rar) using the link specified in **Resources**.
3. Extract the library files directory using whatever file extraction tool is appropriate for your platform (Windows, Linux, OSX, etc.).
4. Copy the UTFT directory to your Arduino/libraries directory. The

RESOURCES

The official source of Arduino information **http://arduino.cc**.

The free and open source Arduino development tools for Windows, OSX, and Linux are available for free at **http://arduino.cc/en/Main/Software**. I used version 1.0.1 to develop this software.

The LCD display is available at **www.sainsmart.com/module /lcd-module/tft-lcd-module/sainsmart-3-2-tft-lcd-display-touch-panel-pcb-adapter-sd-slot-for-arduino-2560.html**. It is also sold through Amazon.

The combination of an Arduino Mega2560, a display shield, and LCD display is available at **www.sainsmart.com/home-page-view/sainsmart-mega2560-board-3-5-tft-lcd-module-display-shield-kit-for-atmel-atmega-avr-16au-atmega8u2.html** and through Amazon.

The Arduino sketch described in this article is available from the article link and is called Contemplator.ino.

Information about the UTFT LCD driver written by Henning Karlsen is available at **www.henningkarlsen.com/electronics/ library.php?id=51**.

directory structure should be Arduino/libraries/UTFT.

5. The library should now be installed.

To verify correct installation of the library, start up the Arduino IDE and select File/Examples from the menu. You should see the UTFT entry listed.

Once you have the hardware wired up, the UTFT driver installed, and the Contemplator sketch available, you can download the sketch via the Arduino IDE and watch the magic happen.

You don't need to understand how the Contemplator sketch works in order to use and/or appreciate it.

If, however, you would like to know how the various graphic display patterns work or if you would like to change the sketch to add new display patterns or remove existing ones, more in-depth knowledge is required. The best way to gather this knowledge is by knowing how the Contemplator is supposed to work and by studying the Contemplator sketch.

Basic Operation

As mentioned in the introduction, the Contemplator provides 16 display patterns for our viewing pleasure. Code in the sketch randomly selects which display pattern is shown and makes sure that all 16 patterns are displayed before any are allowed to repeat. The *flags* array in the code controls this. A display pattern will be shown for either 30 seconds or the time it takes for the pattern to generate and display itself — whichever is shorter. If display pattern generation takes less than 30 seconds, some patterns will repeat so you will see them numerous times in succession.

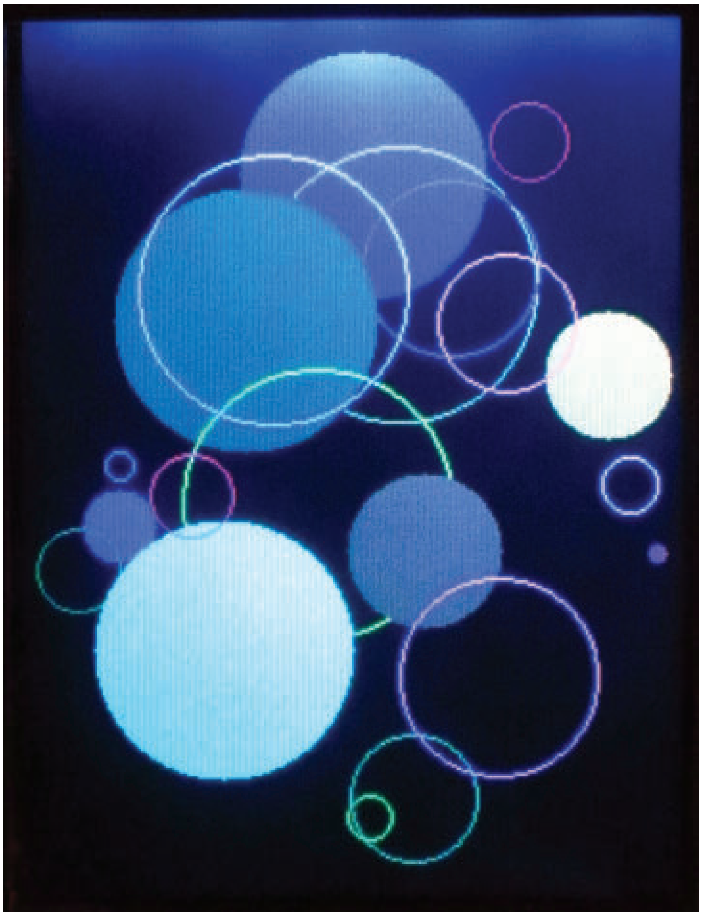
The software was designed so that the individual display patterns do not need to concern themselves with display duration timing. All they need to do is to call the function *checkForTimeout()* periodically and if their display time period has expired, their execution will come to an end and the next display pattern will be invoked. The *setjmp* and *longjmp* mechanism built into the Arduino programming language allows this to work. Google these terms if you are interested in how.

Another interesting aspect of the code is the use of an array (called *patternFunctions* in the code) of function pointers to the display pattern routines. A display pattern is selected and executed via an index into the array with this single line of code:

```
(*patternFunctions[index])();
```

A typical display pattern routine is shown next. This code draws a series of connected line segments until its time is up. Any display pattern routines you write would resemble this code.

```
void linesPattern() {
```



```
int x, y, xPrev, yPrev;
// Pick a random palette
int paletteNumber = random(NUM_OF_PALETTES);
generatePalette(paletteNumber);
// Pick a random place to start within the
// palette
int colorIndex = random(PALETTE_SIZE);
xPrev = WIDTH_LANDSCAPE / 2;
yPrev = HEIGHT_LANDSCAPE / 2;
while (true) {
for (int i = 0; i < 80; i++) {
lcd.setColor(palette[colorIndex].red,
palette[colorIndex].green,
palette[colorIndex].blue);
colorIndex++;
colorIndex %= PALETTE_SIZE;
x = random(MAX_X_LANDSCAPE);
y = random(MAX_Y_LANDSCAPE);
lcd.drawLine(x, y, xPrev, yPrev);
xPrev = x;
yPrev = y;
delay(60);
}
checkForTimeout();
delay(1000 * BETWEEN_PATTERN_DELAY_SECONDS);
lcd.clrScr();
}
}
```

Important things to note about this code are:

- The palette from which the connected lines will be colored is selected randomly.
- The color picked for the first line is selected randomly from the colors contained in the palette.
- A while loop is used to continually cycle this display pattern.
- The *checkForTimeout* function is called within the while loop.
- There is a delay after all 80 lines have been displayed. Then, the display is cleared and the process starts over.

As mentioned, a software change is required to the Contemplator.ino sketch if a SainSmart Mega2560 is used instead of an Arduino Uno. The normal instantiation of the LCD driver for an Arduino Uno is as follows:

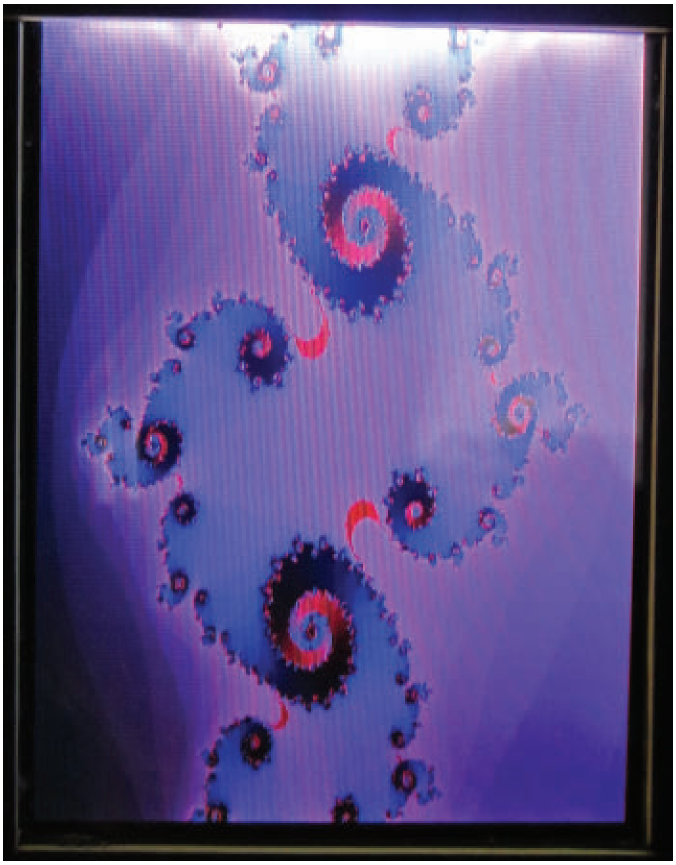
```
// Uncomment out next line when Sainsmart LCD
// display connected directly to Arduino Uno
UTFT lcd(SSD1289, 19, 18, 17, 16, LANDSCAPE);
```

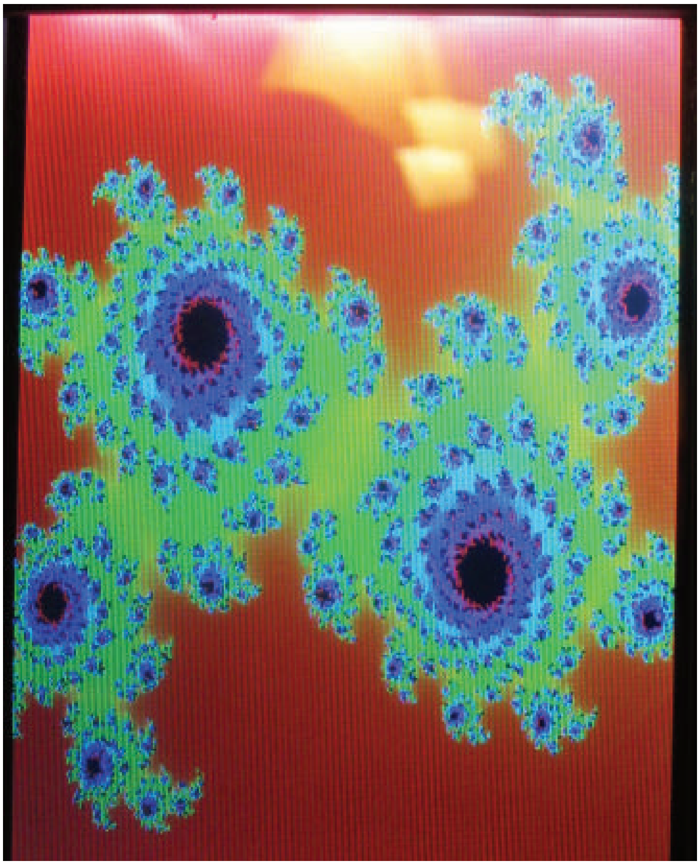
If you are going to use the Mega2560, you would comment out the line above and uncomment the UTFT line below:

```
// Uncomment out next line when running on
// Sainsmart Arduino Mega 2560 + LCD shield +
// LCD display
// UTFT lcd(SSD1289, 38, 39, 40, 41, LANDSCAPE);
```

BOX PARTS LIST

QTY	MATERIAL	LENGTH	WIDTH	THICKNESS	FUNCTION
2	Ash	4.75″	1.5″	3/8″	Sides
2	Ash	3″	1.5″	3/8″	Top/bottom
1	Walnut	4.75″	3.75″	3/8″	Front panel with cutout for the LCD to protrude
1	Walnut	4.75″	3.75″	3/8″	Rear panel with hole drilled for USB cable
4	Brass				#6 screws for holding the back on
	Glue				For holding the box together





With that change, you should be good to go. Make sure you change the board type in the IDE to reflect which Arduino you are using.

Construction and Packaging

The Contemplator was designed to sit on my desk, so I wanted to make it as small and unobtrusive as possible. I also wanted to make it structurally sound so it would last a long time. To this end, I epoxied four 3/4" wooden dowels (1/8" in diameter) to the top corners (component side) of the Arduino PCB (printed circuit board). I then inserted .1" male header pins into all of the Arduino's female headers, to which I would solder wires from the LCD display's connector.

I then epoxied the other end of the four dowels glued to the Arduino onto the back of the LCD display's PCB, making sure to miss all of the components and to orient the Arduino so the USB connector was opposite to the connector on the LCD display. I made sure I left enough space so I could solder to the display's connector pins.

I wrapped this assembly with a rubber band while the epoxy dried. Once the glue cured and the assembly was stable, I soldered short wires from the display connector to the appropriate header pins on the Arduino as specified in **Table 1**.

I packaged my Contemplator in a custom wooden box, but you can package yours anyway you like. I made

my box out of ash and walnut for a nice contrast and a furniture-like appearance. Overall, the dimensions are 3.75" (wide) x 4.75" (tall) x 2.25" (deep); see the specific **Parts List** for the box.

After completing the box, I gave it a couple coats of natural finishing oil and — when dry — steel-wooled it and then rubbed it with paste wax. The finish turned out very smooth and nice (see the **photos**).

The cutout for the LCD display in the front panel was sized so that the actual display fit through the cutout but the display's PCB did not. During final assembly, I put a small amount of silicon caulking onto each corner of the display's PCB and pressed the display into the cutout. Once dried, the silicon holds the display/Arduino assembly in place, but it can easily be removed for servicing.

I had to modify a USB cable to make it work for this application. I cut off the end of the cable that plugs into the Arduino, removed the plastic encapsulation around the connector, and then unsoldered the wires from the connector (taking note of which color wire went where). I then passed the cable through the hole drilled in the rear panel of the box, stripped and tinned the wires, and reconnected them to the USB connector. Finally, the USB connector was plugged into the Arduino.

At this point, I downloaded code into the Arduino to make sure everything was still working. Once I was satisfied all was well, I placed some 3/8" foam on the back of the Arduino's PCB, put the rear panel in place, and screwed it on. The foam provided just enough thickness to gently hold the display/Arduino assembly in place.

Finally — because the frame around the LCD is white — I cut pieces of black cardboard and glued them onto the white frame to cover it. With that, the Contemplator was complete and has been sitting on my workbench/desk ever since.

Flashing Thoughts

Flashing LCD displays and other blinky things are not for everyone. For me, having a personal light show on my desk is a treat. There is something neat about have a little device with its one purpose in life to continually generate images to calm and amaze me.

When I get stuck on something I am working on, I look over at the Contemplator and many times the change in focus is enough to get me going again. I hope this will work for you, as well. **NV**

Craig can be contacted at calhjh@gmail.com.

A BETTER PLACE FOR ELECTRONICS KITS
It's only a great project if I can build it myself!
• A wild collection of maker-designed electronics projects
• All projects packaged as step-by-step kits available for sale
• Kit designers earn a commission on every sale
• Request a project and help influence the design process
• DIY inspiration
Dust off that workbench, fire up the soldering iron and get back in the garage. Visit www.ClubJameco.com/Fun to join the fun.
club JAMECO®

Build Your Own Wireless Sensor Network Using XBee and the 32-bit Experimenter

By Tom Kibalo

In this article, we will apply the full power of the 32-bit Micro Experimenter platform (Experimenter for short) in two different wireless sensor network applications. Each application will demonstrate the power and use of the Microchip PIC32 as a base station, and the XBee series 1 805.4 (ISM 2 band) RF unit from Digi International for both a base station (see Figure 1) and remote sensor RF modules.

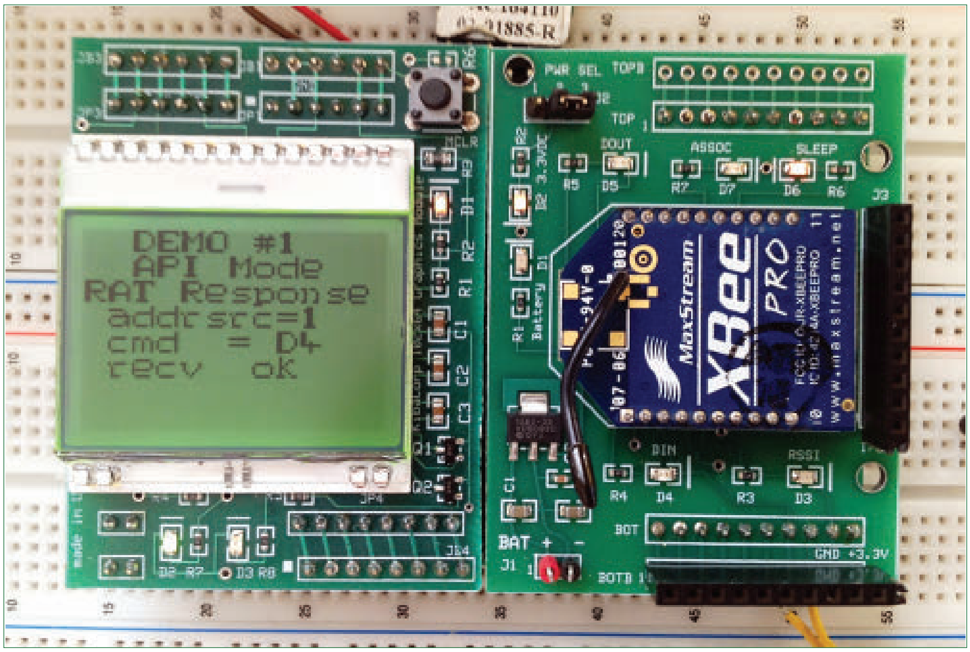


Figure 1. The 32-bit Experimenter XBee base station with graphics display and XBee carrier module.

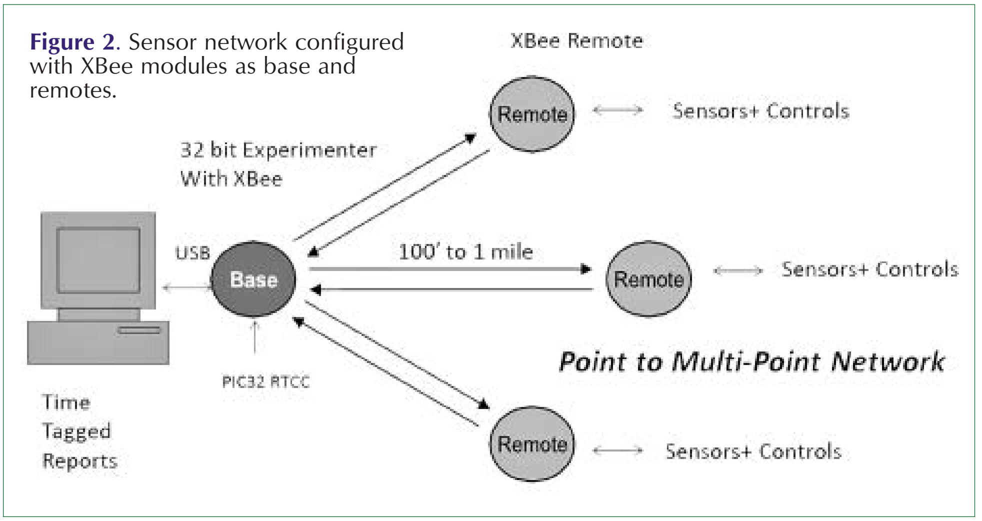
The XBee is a de facto standard wireless sensor module that is easy to use, inexpensive, and readily available for the hobbyist and engineering communities. It is well supported with PC dongles as USB to XBee interfaces can be purchased from a number of vendors. Digi International X-CTU software is available to assist and test XBee setups and configuration. A very cool thing about the XBee is that it can be used as a stand-alone unit, as well as with a microcontroller. We will examine both uses in our demos as we construct our wireless sensor network.

We will focus on minimal use of the dongle and X-CTU, and rely on having our networks remotely configured from the 32-bit Experimenter base station. In the course of our demos, we will also introduce a simplified Application Programming Interface (API) that can be easily adapted to your own wireless network applications.

The base XBee is connected to the Experimenter PIC32 through a serial interface using a UART (Universal Asynchronous Receiver Transmitter) peripheral in the PIC32. All the other remote XBee modules are stand-alone without any microcontroller. All the remote XBees are configured over the network by the base running a setup program executed by the Experimenter. The remotes support nine configurable I/O that can be programmed as either analog or digital. With its built-in USB, the Experimenter can optionally communicate all sensor data to a PC (see **Figure 2**). The Experimenter's Real Time Clock Calendar (RTCC) peripheral can also (optionally) be used to time stamp incoming network data packets. You can find a discussion of the USB to PC and RTCC capabilities in several earlier articles on the Experimenter.

Sound interesting? Let's get started! As mentioned earlier, demonstration code is provided to help assist you in reconstructing and developing your own sensor network experiments. As in all other articles in this series, a familiarity with C code is required.

Figure 2. Sensor network configured with XBee modules as base and remotes.



The XBee RF Module Series and Its Interface

The XBee module has pinouts that provide up to nine I/O. Six of these can be configured as analog with 10-bit ADC (analog-to-digital converter) capability, or can be configured as programmable digital input or output. The remaining pins are fixed digital input and output. The XBee and dongles are available from a number of sources. The XBee is a +3.3 VDC 20-pin DIP module. The +3.3V operation make it directly compatible with the Experimenter.

The XBee has two major versions: XBee or XBee-Pro 802.15.4 (Series 1). These versions come in low transmitting power (1 mW for 100 feet indoor to 300 feet outdoors) and high transmitting power (63 mW for 300 feet indoor to one mile outdoors) units. Another variant on the XBee is an antenna type. The demos covered in this article can use any of these variants, depending on what range coverage you are trying to achieve with your network.

The XBee has a pretty extensive command/control interface. It can operate in two distinct modes of operation. In mode 1 — the AT Command Mode — the XBee interfaces to a host microcontroller through a UART asynchronous serial interface using an ASCII (American Standard for Communication Interchange) command set. In mode 2 — API Mode — the XBee uses lower level network packets with binary data and checksum error checking (we will discuss checksum in more detail a little later). To achieve what we need to do in direct control of the remote XBee, the API mode is required. We will use the AT mode only during power-up of the Experimenter.

The API mode allows the Experimenter base to communicate directly with a remote stand-alone XBee (no additional microcontrollers are required). API mode is the preferred means to configure remote modules and route data around the wireless network. A base module can send API data packets to a remote module with a destination address and command. The remote XBee, in turn, will send data packets to the base module containing status, address of remote, signal strength, and sensor data. In summary, the API mode facilitates all the necessary operations for a single point (base XBee and Experimenter) to multi-point (remote XBees) network configuration. This is shown in **Figure 2**.

A plug-in carrier module for the XBee is available through the *Nuts & Volts* Webstore. This plug-in module supports a direct XBee plug-in with the Experimenter, as well as functioning as a remote with battery operation and

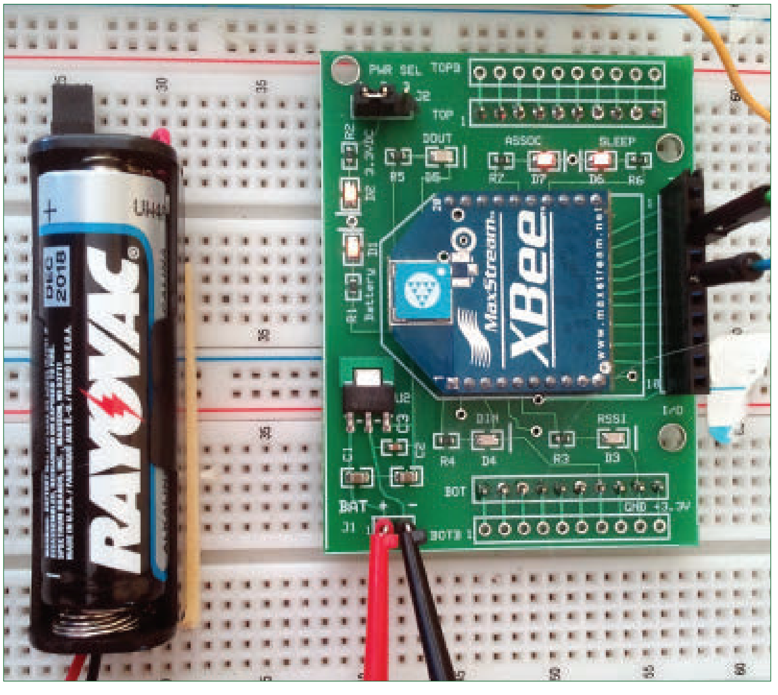


Figure 3. XBee carrier module functioning as a remote sensor.

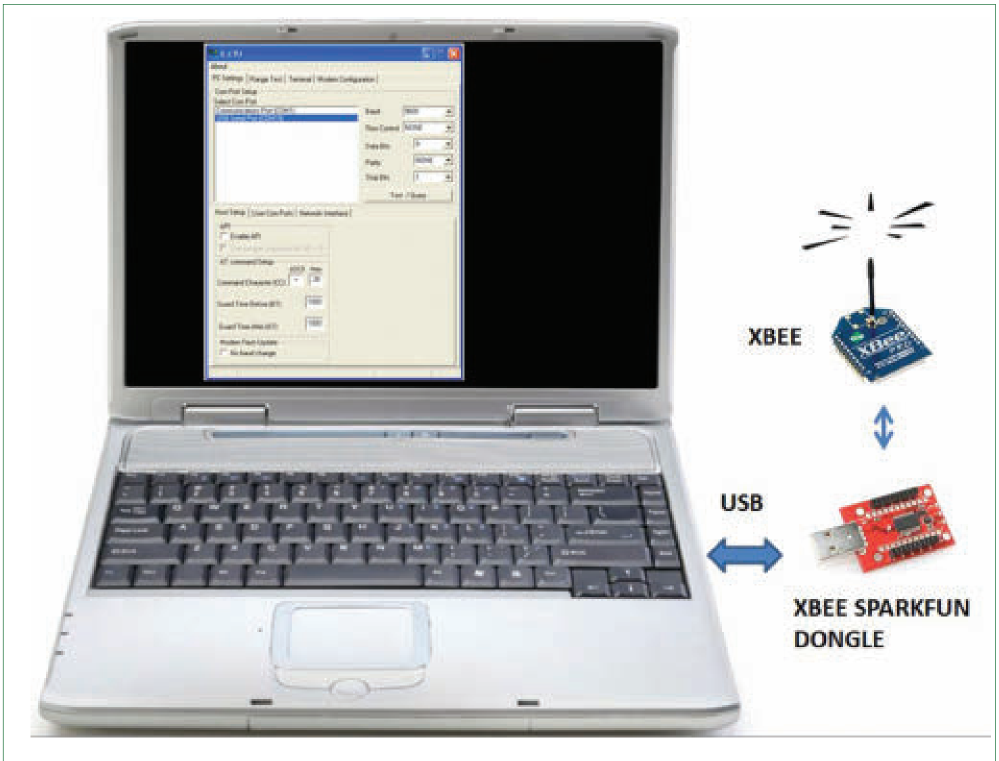


Figure 4. XBee hardware and software configuration tools.

are available from any vendor who supplies the dongle.

Once a COM port is selected, you can test communications with the XBee, reconfigure the module firmware with the appropriate settings, and program the module.

To run the demos, we want to configure up to two remote XBee modules with different module addresses 1 and 2, and set these remote modules to API mode. The Experimenter base XBee module will automatically be configured using the PIC32 on power-up.

an external connector for digital/analog sensor and output control hook-up (see **Figure 3**). The carrier module has onboard +3.3V regulation for battery operation, and supports power and activity LEDs. The activity LEDs are associated with data in, data out, received signal strength (brightness varies with strength of received signal), active sleep condition, and finally, network association (normally blinks on power-up). The available carrier module sensor I/O is as follows:

- DIO0–DIO4 and DIO6: These are used as standard 3.3V digital inputs and outputs. ADC is a programmable option for DIO0-DIO6. These are 10-bit ADC inputs to the XBee.
- Fixed digital input DI7 and fixed digital output DO8.
- +3.3V and GND outputs.
- Battery input terminals.
- Power source selection of either an external battery (<= 6 VDC) or the Experimenter +3.3V and GND.

First Time Configuration of the XBee Modules

Free X-CTU software can be downloaded from Digi International (**ftp://ftp1.digi.com/support/utilities/Setup_XCTU_5260.exe**) and used as a serial terminal for configuration changes, testing RSSI (receive signal) levels, and for downloading different firmware to the XBee module. It works though a pluggable USB dongle for the XBee module (see **Figure 4**). You must first install X-CTU, plug in the dongle with an XBee module inserted, configure the dongle USB (available from **SparkFun.com**) as a COM port, and then finally select the required COM port to talk to the XBee with dongle. Detailed instructions

API Communications

A user guide describing the command types and XBee module operation is freely available at **http://ftp1.digi.com/support/documentation/90000991_E.pdf**.

There is no getting around the need to understand the extensive command set. In fact, in our examples we will use sleep mode to have the remote XBee turn off periodically between transmissions, set sample rates for how often sensor data is collected or sampled, and configure I/O for analog or digital. Being somewhat fluent in the command set is important for building your own network control, so we recommend a review of the XBee user guide. We will use a highly simplified API to issue XBee commands, retrieve responses, and gather data.

The simplified API utilizes all the power of the XBee API mode through the use of five functional calls. There are two main functions for initialization, and three main functions for communication.

Initialization Functions

initU3 () — Initializes the PIC32 serial port (UART 1B) for use with XBee communication. The serial communication is 9600 baud, eight bits, no parity, and one stop bit.

ResetXbee () — Activates a hard reset to the XBee module from the Experimenter to initialize the XBee.

Communication Functions

Send_RAT (int cmdparameters, char* cmd, int addr, byte option, byte datalength) — Sends a remote

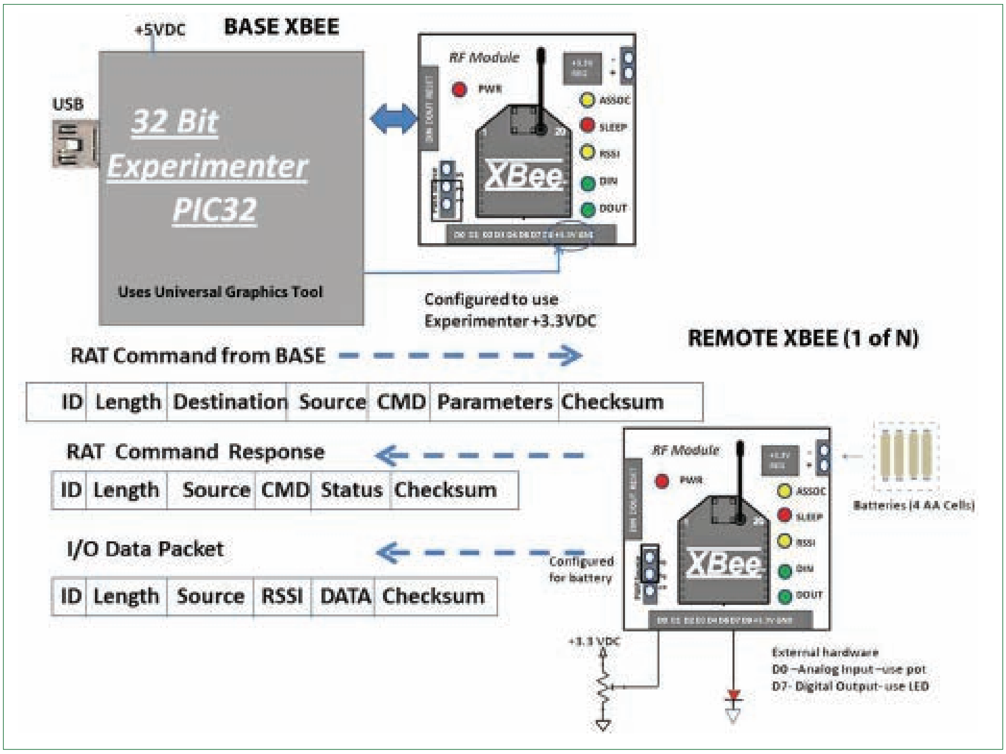


Figure 5. Wireless sensor network concept of operation.

command and associated command parameters to a 16-bit remote sensor address. Data length is the size of the command parameters which is fixed at four bytes. Both the base and remote unit must be in API mode. The remote unit will send a command response packet back to the base.

Send_LAT (int cmdparameters, char* cmd, byte datalength) — Sends a local command and associated command parameters to the local unit. Data length is the size of the command parameters which is fixed at four bytes. The base must be in API mode. The local unit will send a command response based on receiving this command.

Int recv_packets (void) — This function is called to determine if a network packet has been received at the base unit. The type of packet and associated data is shown on the graphics display. It is also stored in a global memory location designated "mypacket" for any additional processing you may want to do. Responses to command response packets are processed, as well as data packets from remote data sensors. If a data packet is received, the data will exist in a global memory array sample [9]. The function returns an *int* indicating that a packet is available.

Network Operation

A block diagram example of network operations using the API mode is shown in **Figure 5**. In an outgoing command packet (base to remote), ID represents the packet type as command(RAT, LAT) or data; length is the byte size of the entire packet; destination is the 16-bit remote address for where the packet needs to go. The source is the 16-bit address for the packet originator; in this case, the base. There are packet fields for CMD (command type) and CMD parameters associated with the command, and then finally checksum. Checksum is used for error detection of transmitted packet data. It is a unique byte signature that is the sum of all the data content within the transmitted packet.

The command response packet is similar, and includes the remote source address, the CMD the remote has received, and the status of the CMD execution (OK, Unknown CMD, or error). The I/O data packet is used to transmit sensor data from the remote back to the base. Besides the data itself, the only new field is RSSI which represents the received signal strength of the base signal as measured by the remote.

Initial XBee Setup and Configuration

Here's where the dongle and X-CTU come in handy. You need to pre-configure your remote sensor address and set up API mode to allow the Experimenter to communicate with the remote. For the demos, we use two remote addresses (MY = 1 and MY = 2). It is very straightforward.

Plug the remote XBee module into the dongle, and then hook up the dongle to the PC USB. Open up X-CTU. The X-CTU will come into a setup mode showing the COM port that is now available to use. Select the COM port, then test it to insure the dongle communication is working. If all is okay, select the modem configuration, and read the XBee module using the GUI tool "Read." Scroll to MY setting under configuration and set the address required in the text edit box. Scroll to the API setting and select API as active. Finally, using the GUI tool "Write," program this new configuration to the XBee. Work both remote modules in a similar way. You are done!

Demo Descriptions

The demos are built with MPLAB 8.80 and the PIC32 free C compiler. Make sure you have properly configured the remote XBee using X-CTU and the dongle with their required addresses and API mode. In our first demo, the Experimenter will configure its local D7 as a digital output, and then configure remote (MY = 1) D0 as a digital

```
// Power up AT Command Mode at Base
sprintf( buf, "+++" );  // enter command mode
 XBee_Write(  (uchar*)buf, strlen(buf) );  //write to XBee
 Delayms(1200);     //delay to insure command mode is entered
 // set my =0 and D7 to  high and enter API mode
printf( buf, "ATMY0,D75,API1,CN\r" );
 XBee_Write( (uchar*)buf, strlen(buf) );
 /* Local API Command for base LED blink */
Send_LAT (4,"D7",4 ); //base my=0 D7- high
  recv_packets (); //receive response
   Delayms(2000);
    Send_LAT (5,"D7",4 ); //base my=0 D7-low
    recv_packets ();   //receive response
  Delayms(2000);
/* remote API Commands for remote Led Blinks */
while(1) {
  Send_RAT (5,"D0", 0x0001,2 , 4 ); //my =1 D0 -high
  recv_packets ();  //receive response
      Delayms(2000);
    Send_RAT (4,"D0", 0x0001,2 , 4 ); //my =1 D0 -low
    recv_packets (); //receive response
     Delayms(2000);
   Send_RAT (5,"D1", 0x0002,2 , 4 );  //my =2 D0-low
  recv_packets (); //receive response
      Delayms(2000);
    Send_RAT (4,"D1", 0x0002,2 , 4 ); //my =2 D0- high
    recv_packets ();  //receive response
     Delayms(2000);
}
```

Figure 6. Demo 1 code snippets.

output and remote (MY = 2) D1 as a digital output.

1. DEMO1 "Hello World" – My Network

Open project Demo1 in MPLAB IDE. Configure the Experimenter with the plug-in Universal Graphics Display and the plug-in RF Module with XBee, and connect the Experimenter ICSP to your PICKIT3 or REAL ICE. The Universal Graphics Display module is a plug-in for the Experimenter; the display is monochrome 128x64 with fixed 8x8 fonts. Run "build" and program. A sample of the API code snippets in the Main function are shown in **Figure 6**.

The first snippet is the power-up communications with the XBee base that uses the AT command mode in MAIN.C. The mode is entered using the character string "+++." The commands are AT MY 0 (set MY = 0), D7 5 (make D7 digital low), API 1 (set API mode), and CN (exit command mode). All responses to these commands from the XBee should be the ASCII "OK" string. You can see the individual responses on the graphics display.

In the second snippet, we are no longer in AT mode, so use the API LAT command (local AT command packet) to command the local base XBee D7 output to blink low and high. The commands are D7 4 (D7 output digital high), and D7 5 (D7 output digital low). Connect an LED in series with a 1K resistor to ground. Make sure the anode of the LED is connected to D7. As LAT is executed, you should see the LED blink and all the API responses displayed on the graphics module.

In the final snippet, each remote (two in this example with MY = 1 and MY = 2) should have an LED attached in series with a 1K resistor to ground. Again, make sure the LED anode is connected to D0 for MY = 1 and D1 for MY = 2. Using RAT (remote AT command packet), the base will command each remote sensor — using its address — to blink its LED in turn, going round robin through the network. The commands are D0 4 (D0 output digital high) and D0 5 (D0 output digital low) for MY = 1, and D1 5 (D1 output digital high) and D1 5 (D1 output digital low) for MY = 2. All RAT command responses are captured on the graphics display.

For our second demo, the Experimenter will configure MY = 1 with D4 as an analog input.

2. DEMO2 "Wake Up Sensor" and "Give Me Data Network."

Open project Demo2 in the MPLAB IDE. Configure the Experimenter again with a plug-in graphics display and the plug-in RF module with XBee. Connect the Experimenter ICSP to your PICKIT3 or REAL ICE. Run "build" and program. A sample of the API code snippet in Main function is shown in **Figure 7**. Here, a remote (MY = 1) has an external port attached to D4 that is configured as an analog input source. The base will configure the remote with sample time, a transmit time based on the number of samples, and a sleep/wake-up mode. On wake-up, the remote will report its data to the base in a 16-bit I/O data packet. Sleep mode is a neat trick to help conserve battery power for the remote sensors.

For this particular demo, our RAT command sequence is D4 2 (ADC analog input for D4 pin), Sample rate IR 0x1f40 (< 8 sec), transmit rate one sample before transmit (IT1), SP (sleep period 4 sec) 0x190, and Cyclic Sleep mode SM 4. All responses will appear on the graphics display.

So, let's summarize. We have enabled sensor MY = 1 to perform a cyclic sleep of four seconds (four seconds on/off) and commanded it to report (transmit)

```
// remote sensor my=1 sleep mode and reporting
Send_RAT (2,"D4", 0x0001,2 , 4 ); //my =1 D4 is ADC
  recv_packets (); // receive response and display
     Delayms(2000); //  delay for Display response
   Send_RAT (0x1f40,"IR", 0x0001,2 , 4 ); //my =1 IR 0xfa0 (8sec)
    recv_packets ();  // receive response and display
     Delayms(2000);
   Send_RAT (1,"IT", 0x0001,2 , 4 );  //my =1 samples before TX
  recv_packets (); // receive response and display
     Delayms(2000);
   Send_RAT (0x190,"SP", 0x0001,2 , 4 );  //my =1 SLEEP period = 4 seconds
  recv_packets (); // receive response and display
     Delayms(2000);
   Send_RAT (4,"SM", 0x0001,2 , 4 );  //my =1 SLEEP mode = cyclic
        while (1) {
   if( recv_packets ()) //  wait until receive response and display
     Delayms(2000);
    }
```

Figure 7. Demo 2 code snippets.

one sample set (IT1) at a time, after an I/O sampling occurs (once per eight seconds). The end result is that a data packet is sent from the remote to the base. With sleep enabled, the remote sensor should be asleep about half the time. You can detect sleep mode by noting whether all the LEDs (including sleep) on the carrier module are turned off.

The received data will be displayed on the Experimenter display module to include the received signal strength and sensor address — all part of the data packet. The received data is also stored for internal use in an array sample [9]. The display module green LED will blink for each display update. Data will remain on the display for two seconds and then refresh when another sensor update occurs.

What's Next?

We introduced a simplified API to help you construct your own wireless network. In addition, a new Experimenter RF plug-in carrier module for XBee was introduced to help facilitate implementation. The carrier module works with both the Experimenter and stand-alone as a remote wireless sensor. These tools should make wireless networks easy to experiment with, and help you facilitate your own ideas.

Try integrating the sensor network with your PC using the Experimenter's built-in USB. An additional feature is to try to use the Experimenter's RTCC for time tagging.

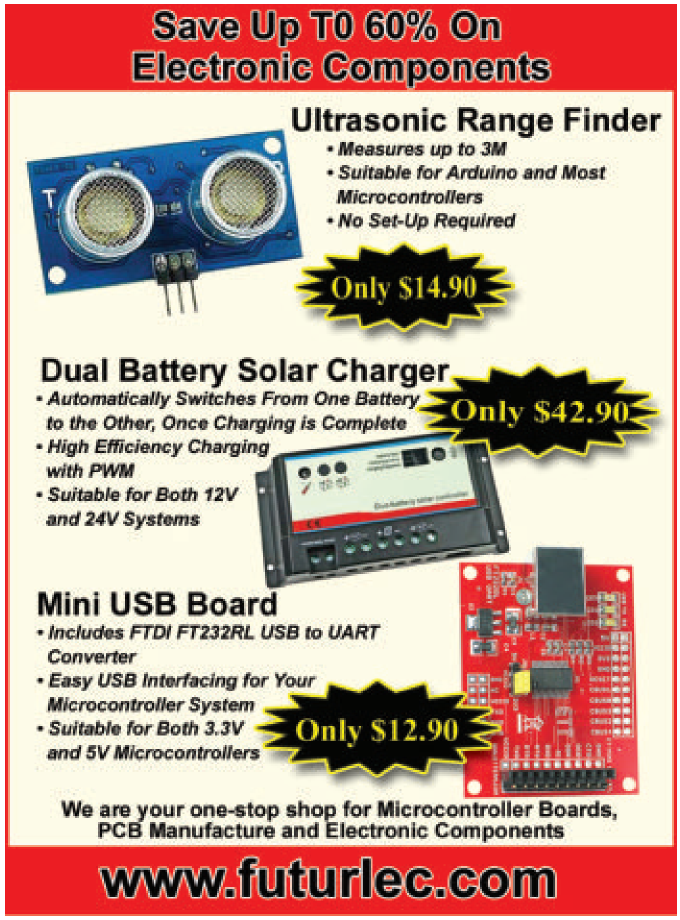
In the next article, we will see how to use the carrier module with Microchip's new Roving Network 802.11 Wi-Fi module to explore the Experimenter in wireless Internet applications. Can anyone say "wireless sensor network to Internet gateway?" Until next time, happy 32-bit networking! **NV**

Some API Tips and Tricks

Follow these guidelines when using the simplified API:

1. Only 16-bit addressing is applicable.
2. Command packets execute commands with command parameters. A command response will always occur from a remote. Command parameters are four bytes in length.
3. Only one TX per sample set is supported to minimize PIC32 memory usage. Samples are stored in an integer array size nine sample [8] with one integer per each I/O pin.
4. You need to configure the remote with its "MY" and set API mode before use.
5. The recv_packets() function must be called to process incoming packets. The function can only process one packet at a time.


Custom Front Panels & Enclosures
FREE Software
Front Panel Designer
Sample price $57.32 + S&H
Designed by you using our FREE software, Front Panel Designer
Cost effective prototypes and production runs
Powder-coated finish and panel thickness up to 10mm now available
Choose from aluminum, acrylic or customer provided material
1, 3 and 5-day lead times available
FRONT PANEL EXPRESS
FrontPanelExpress.com
1(800)FPE-9060


Save Up TO 60% On Electronic Components
Ultrasonic Range Finder
• Measures up to 3M
• Suitable for Arduino and Most Microcontrollers
• No Set-Up Required
Only $14.90
Dual Battery Solar Charger
• Automatically Switches From One Battery to the Other, Once Charging is Complete
• High Efficiency Charging with PWM
• Suitable for Both 12V and 24V Systems
Only $42.90
Mini USB Board
• Includes FTDI FT232RL USB to UART Converter
• Easy USB Interfacing for Your Microcontroller System
• Suitable for Both 3.3V and 5V Microcontrollers
Only $12.90
We are your one-stop shop for Microcontroller Boards, PCB Manufacture and Electronic Components
www.futurlec.com


Web-based Control and Monitoring
Get your project online... in minutes!
No Programming Required
No Port Forwarding or Dynamic DNS
No Network Configuration Necessary
No Need to Host a Web Server
IO-204 Ethernet Module
WHAT WILL YOU DO?
Real-time, secure control and monitoring...
Servo and Relay Control
X10 Home Automation
RS-232 Serial Communication
Temperature and Analog Sensors
Digital Inputs and Outputs
Garage Door
Open | Close
Outside
81.6°F
Outside Light
On | Off
Inside
71.8°F
Driveway Cam
Left | Right
Feed Sparky
Feed Fish
Solar Panel Output 141W
all from a web browser.
Event-based Alerts, Email and SMS
Embed Controls in your Web Site
Encrypted, Secure Access
Data-logging Service
Open API for Web Applications
(example dashboard)
Discount Code:
NVDiscount
10% Off
ioBridge
iobridge.com/store
sales@iobridge.com
1-508-630-2165


THE ORIGINAL SINCE 1994
PCB-POOL®
Beta LAYOUT
FREE Stencil
with every prototype order
EAGLE order button
pcb-pool.com/download-button
20% off!
on your first order
Call Tyler: 1 707 447 7744
sales@pcb-pool.us
PCB-POOL® is a registered trademark of
Beta LAYOUT
www.pcb-pool.com

CanaKit™

Electronic Kits • Modules • Parts

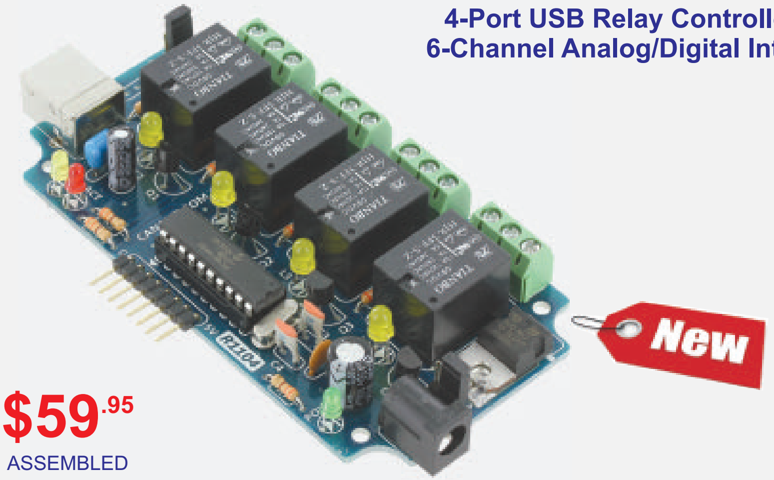
Save 10% using Coupon Code NV2012*
when ordering online at www.canakit.com

Motor Controllers • Arduino • Signal Generators • FM Transmitters • Timers • Audio Amplifiers

200 KHz Function Generator

$49.95 ASSEMBLED — UK307

4-Port USB Relay Controller with 6-Channel Analog/Digital Interface



$59.95 ASSEMBLED — UK1104

5A Motor Speed Controller (PWM)

$22.95 ASSEMBLED

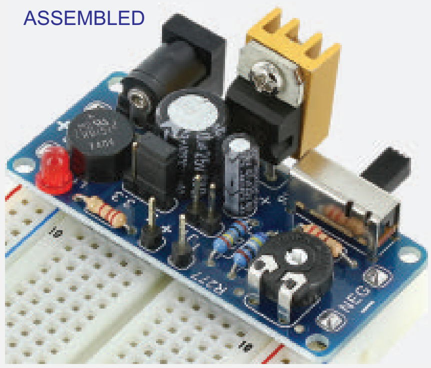
UK1102

Stereo Audio Amplifier (10W)

$35.95 ASSEMBLED — UK560

Plug-In Breadboard Power Supply

$19.95 ASSEMBLED — UK277



7A Bi-Directional Motor Speed Controller (PWM)

$39.95 ASSEMBLED

UK1125

USB PIC Programmer (MPLAB Compatible)

$69.95 ASSEMBLED — UK1300

Digital LED Timer (Up/Down)

UK258

$46.95 ASSEMBLED

1•888•540-KITS (5487) info@canakit.com **www.canakit.com**

* Limit one per customer. Can not be combined with other coupons or volume discounts.
Does not apply to products already on sale. Prices subject to change without notice.

BY JOE PARDUE

SMILEY'S WORKSHOP #53

C PROGRAMMING - HARDWARE - PROJECTS

Part 5 ARDUINO PROTO SHIELD

Last time, we continued learning about Fritzing — a novice-friendly electronics hardware design package — useful for things like designing shields for an Arduino.

This month, we are going to finish our Fritzing workshops by tying up a few loose ends and then introducing some practical applications with a couple of very useful new kits that will let us easily implement Fritzing designs.

We will look at an Arduino proto shield and a breadboard battery that we will use to design an Arduino-based alarm clock on a breadboard (Figure 1) and then we will transfer that design to the prototyping area on an Arduino proto shield (Figure 2).

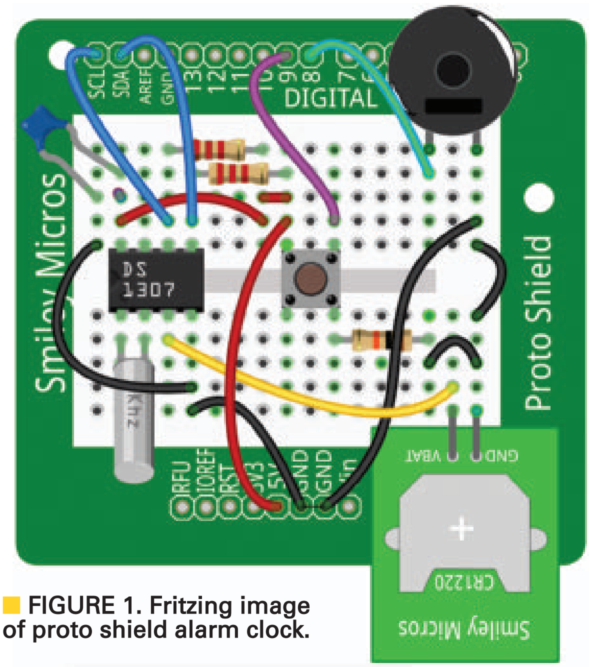


FIGURE 1. Fritzing image of proto shield alarm clock.

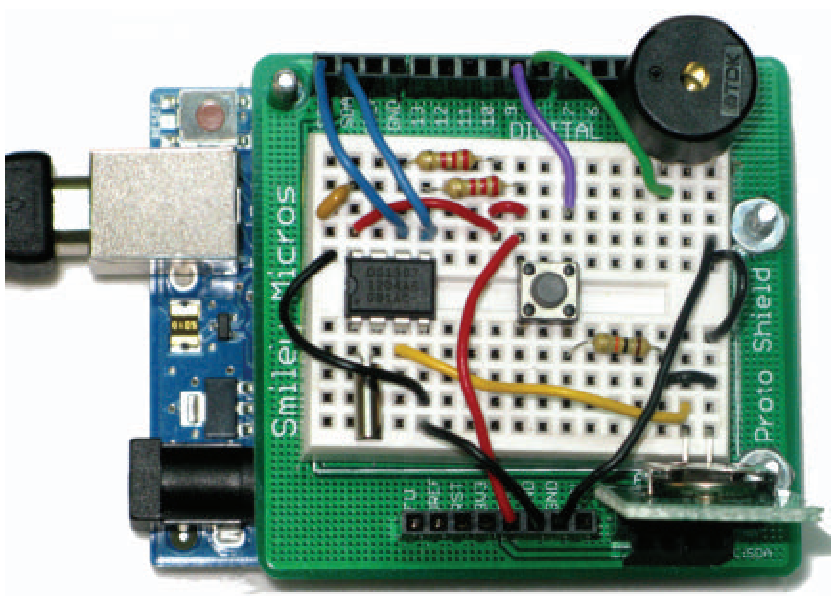


FIGURE 2. Arduino proto shield with alarm clock.

FINISHING OUR FRITZING WORKSHOPS

Fritzing helps us see the connections

One of the main problems with photographs of wired-up breadboards is that the wiring can be incomprehensible and quite daunting to a novice. How on earth can you figure out where all those wires go and what is connected to what? One of the things that Fritzing is good at is allowing you to highlight a set of common connections on the board to help you see where things are located.

For example, if you hover the cursor over the GND header pin and press the left mouse button, you get all the connections appearing in yellow as shown in **Figure 3**. Now, you can clearly see that both the upper left and upper right columns are connected to GND as is the capacitor plugged into the left upper column, the piezo speaker plugged into the right upper column, and the breadboard battery plugged into the lower right column.

We can likewise see all the +5 volt connections by highlighting the 5V header as in **Figure 4**. This shows how the two 10K ohm resistors, the left side of the pushbutton, and the leftmost upper pin of the DS1307 are all connected to +5V. I realize it is still a bit like trying to read a message spelled out in a bowl of spaghetti, but nonetheless it is better than a raw photograph (as you can see by comparing these images to the photo in **Figure 2**, which is actually quite clear compared to many you'll see).

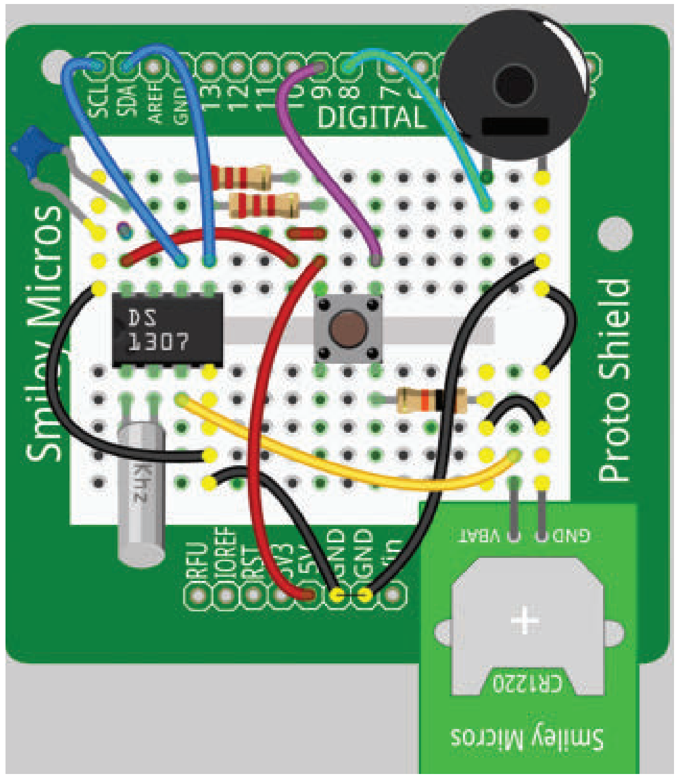


FIGURE 3. Highlighting GND.

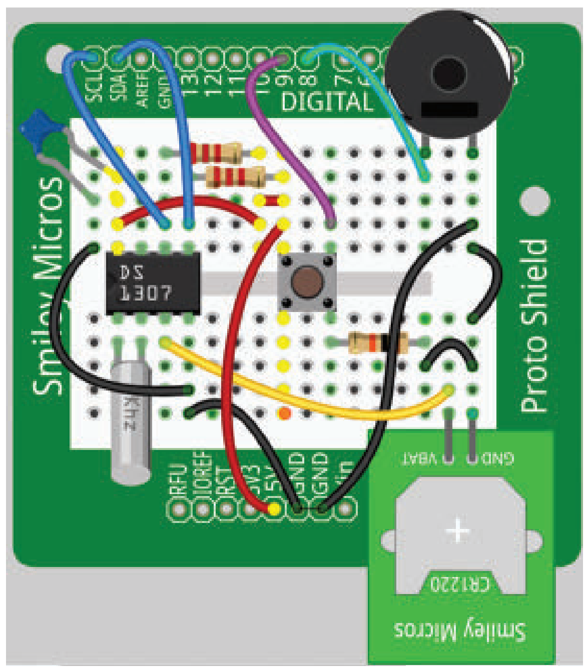


FIGURE 4. Highlighting +5V.

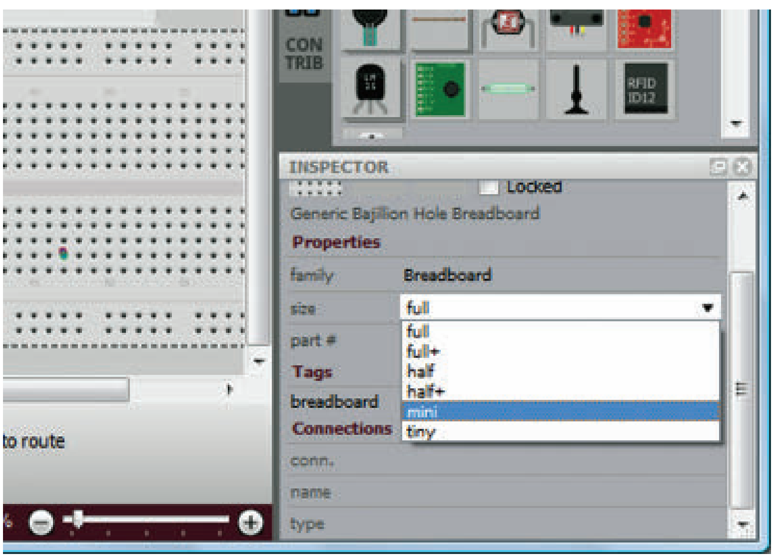


FIGURE 5. Select the mini breadboard.

FRITZING MINI BREADBOARD – LIGHTEN UP!

When you open Fritzing, you'll see the full size breadboard in the breadboard view window. To change the full-sized board to a mini breadboard, go to the Part Inspector and select the size "mini" as shown in **Figure 5**.

The mini breadboard has 170 tie points and no power bus. In my opinion, it has a problem — it is too darn dark and hardly stands out against the background. So, I decided to lighten it up a bit and in so doing, show yet another example of how to modify a part by directly editing the .svg (scaled vector graphic) file. I opened the file Fritzing\parts\svg\core\breadboard\miniBreadboard.svg in Programmer's NotePad and changed the fill value from #D9D9D9 to #F9F9F9:

```
<g id="background">
       <rect x="0" fill="#D9D9D9" width="129.839"
       height="100.914"/>
</g>

to:

<g id="background">
       <rect x="0" fill="#F9F9F9" width="129.839"
       height="100.914"/>
</g>
```

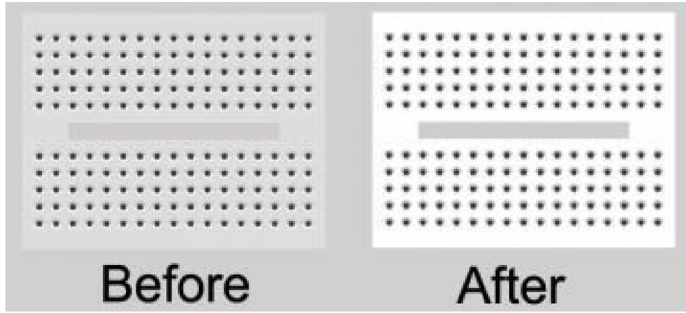


FIGURE 6. Piezo .svg drawing.

This gives us the lighter image shown in **Figure 6**. This is — of course — a matter of personal preference, but I prefer the lighter version, so that's what I'll use. This bit of file editing is deceptively easy though, and you'll need to be very careful when trying to figure out what other sections of the file to do.

I suggest making a backup copy first; then for each little change you make, save the change and open the .svg file in a browser to see what you really did (drag and drop, then refresh to see the changes). You can learn a lot this way. You might also want to take a look at **http://weblogs.asp.net/bleroy/archive/2011/08/09/building-a-simple-fritzing-component.aspx**.

"FRITZING" AN ARDUINO PROTO SHIELD

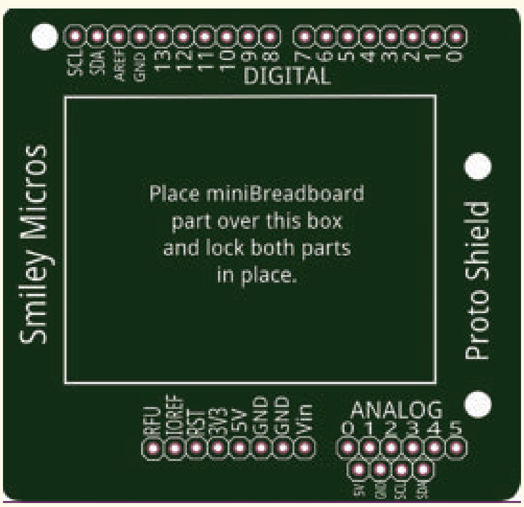


FIGURE 7. Arduino proto shield without breadboard.

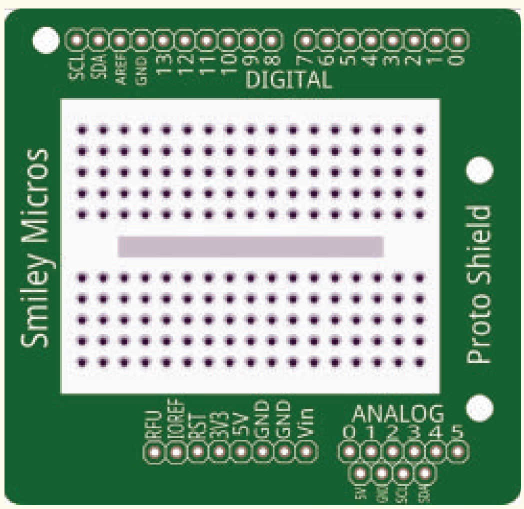


FIGURE 9. Proto shield with breadboard.

We've already looked at Inkscape and decided to let you learn how to use that excellent program on your own. It isn't easy (none of the vector graphics programs are easy), but it is free.

If you want to make illustrations like the ones I'm using, it is an excellent tool. I used it to generate the Arduino proto shield image shown in **Figure 7**.

You can learn more about generating your own custom PCB image at **http://fritzing.org/learning/tutorials/designing-pcb/pcb-custom-shape**.

You can get this Fritzing part at the article link for this month's Workshop.

To use this in Fritzing, you add the shield to the breadboard view, then right-click the board, and select 'Raise and Lower' \ 'Send Backward as shown in **Figure 8**. Then, slide the board under the mini breadboard as shown in **Figure 9**.

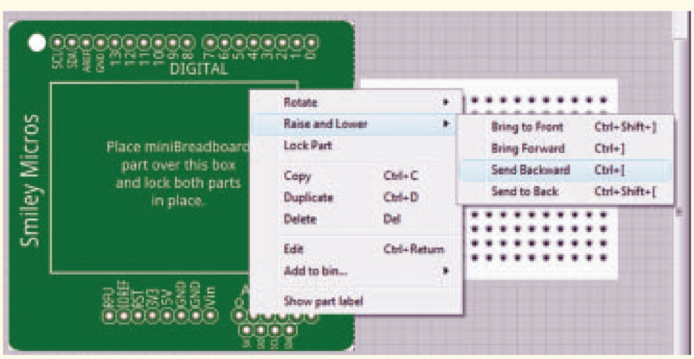


FIGURE 8. Send board to back.

INTRODUCING THE ARDUINO PROTO SHIELD

As I said at the beginning, the actual PCB is a bit different from what we show in Fritzing in that it has pads on the board that match the connections on the mini breadboard, and it has an extra set of connection points for the Arduino pins. This is shown in the photo in **Figure 10**. These pads will allow us to directly transfer a design that we got working on a mini breadboard to a proto shield, and solder the parts to the PCB. We'll do that very thing first thing in next month's Workshop.

I didn't produce a Fritzing version of the final PCB showing all the holes under the breadboard area because it was just too darn difficult. The part editor kept messing up (but they do warn you that it is buggy) when I was trying to add all the pads to match the mini breadboard connections. I finally gave up and just used another CAD package that I'm competent with (Eagle) to design the PCB.

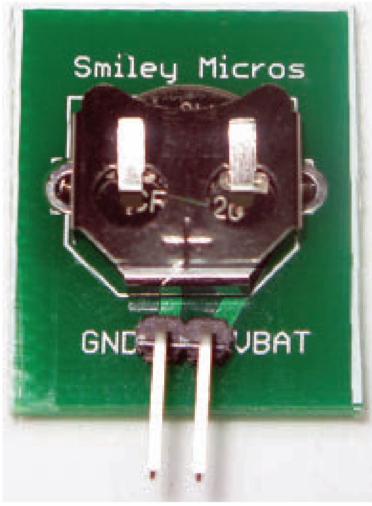


FIGURE 11. Breadboard battery with parts soldered on.

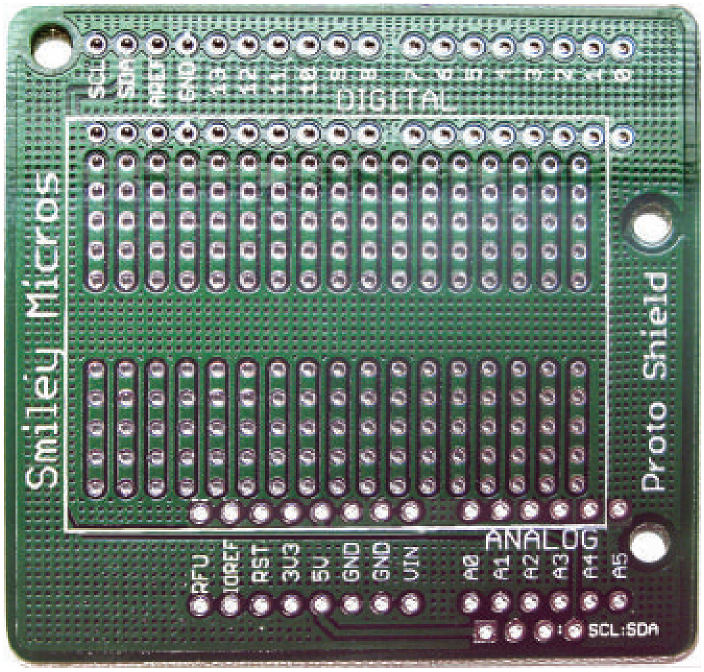


FIGURE 10. Proto shield PCB layout.

BREADBOARD BATTERY BACKUP

A real time clock IC like the DS1307 needs a three volt battery backup to keep track of time when the main power is removed. Since we can't be certain that the Arduino will always have power or even that the proto shield will always be plugged into the Arduino, we need a backup battery for our alarm clock. In the August 2012 Workshop, I issued a dire warning about not soldering wires to a coin cell battery for use with a breadboard — hey, they might blow up and you wouldn't want that. But, then again, the coin cell batteries are so darn convenient for battery backup, plus wouldn't it be great if there were some way you could use one with a breadboard? Yup, I'm leading to something here. You can get a breadboard backup battery like the one shown in **Figure 11** from the *Nuts & Volts* webstore. **Figure 12** shows the part we'll use in Fritzing illustrations like in **Figure 1**. The kit shown in **Figure 13** requires soldering — but not to the battery — and the battery is included. [IMPORTANT CAVEAT: These boards — to keep the costs low — are cut from panels using shears, so the edges may not be as smooth as you are used to. They will work just fine, but don't expect perfection in the finish.] We'll use this device to backup our DS1307 RTC on the following

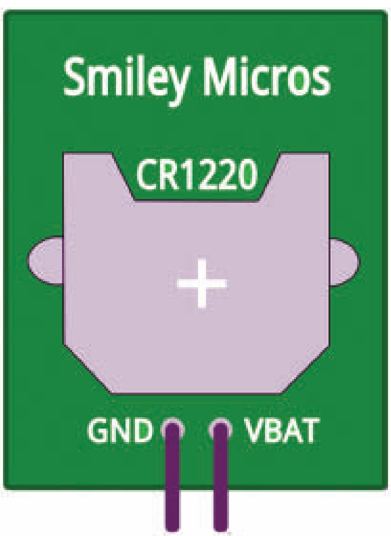


FIGURE 12. Fritzing part for the breadboard battery.

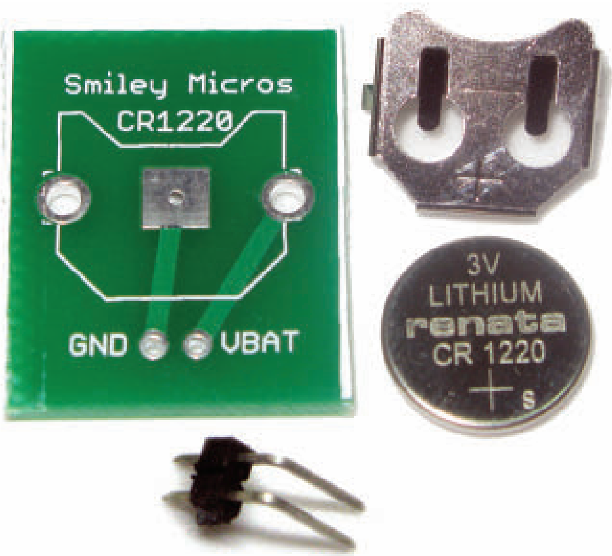


FIGURE 13. Breadboard battery kit.

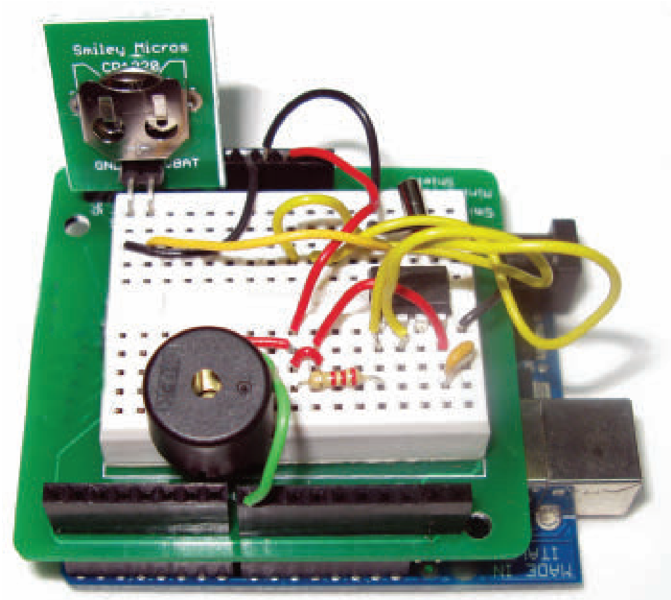


FIGURE 14. Breadboard battery in use.

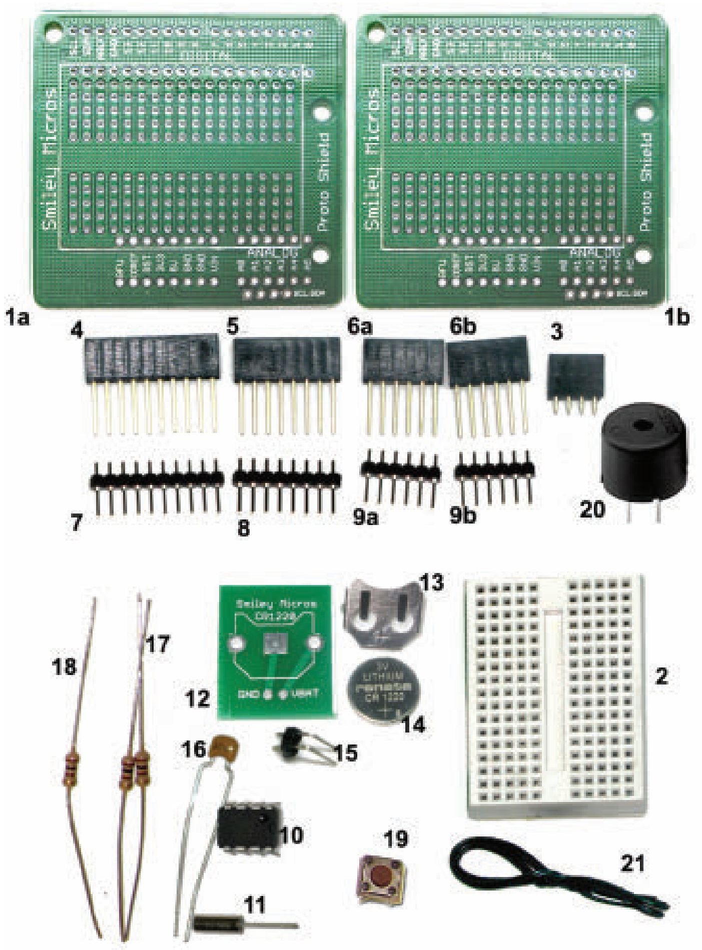


FIGURE 15. Arduino proto shield alarm clock project.

alarm clock project shown in **Figure 14**.

AN ARDUINO ALARM CLOCK

Let's say we take all we've learned recently about Fritzing and real time clocks, and build an Arduino alarm clock like the one back in **Figure 2**. This is far cooler than an ordinary alarm clock because it can talk to a PC over the Arduino USB port. This allows us to set the time with great accuracy from Internet sources, and it lets us set up alarms using a PC terminal interface (instead of a few buttons like you'd see on a regular alarm clock).

First, we will build the circuit on the mini breadboard, which is a great way to test that a prototype works. Once we get it working, we will transfer all the parts and wires to PCB pads that exactly duplicate the mini breadboard connections — this will let us take the tested design on the breadboard — which isn't all that robust — and move it to a PCB which is very robust. By robust, I mean how the circuit will handle being jostled around or dropped.

The breadboard will allow parts to come loose, but the PCB will have the parts firmly soldered down and be much more resistant to being thrown about. We will continue this into next month's Workshop. (If you want to follow along, you can get all the components shown in **Figure 15** and listed in **Table 1** in a kit from the *Nuts & Volts* webstore.)

Item	Description
1a, 1b	Two proto shield PCBs
2	One mini breadboard
3	Four-pin female header
4	10-pin shield header
5	Eight-pin shield header
6a, 6b	Six-pin shield header
7	10-pin male header
8	Eight-pin male header
9a, 9b	Six-pin male header
10	One DS1307 RTC IC
11	One 32.768 kHz watch crystal
12	One breadboard battery PCB
13	One battery holder 12 mm coin
14	One battery CR1220
15	One two-pin 90 degree male header
16	One capacitor 0.1 µF
17	Two resistors 2.2K ohm
18	One 10K ohm pull-up resistor
19	One pushbutton
20	One piezo buzzer
21	Two feet of uncut jumper wire

Table 1. Bill of Materials.

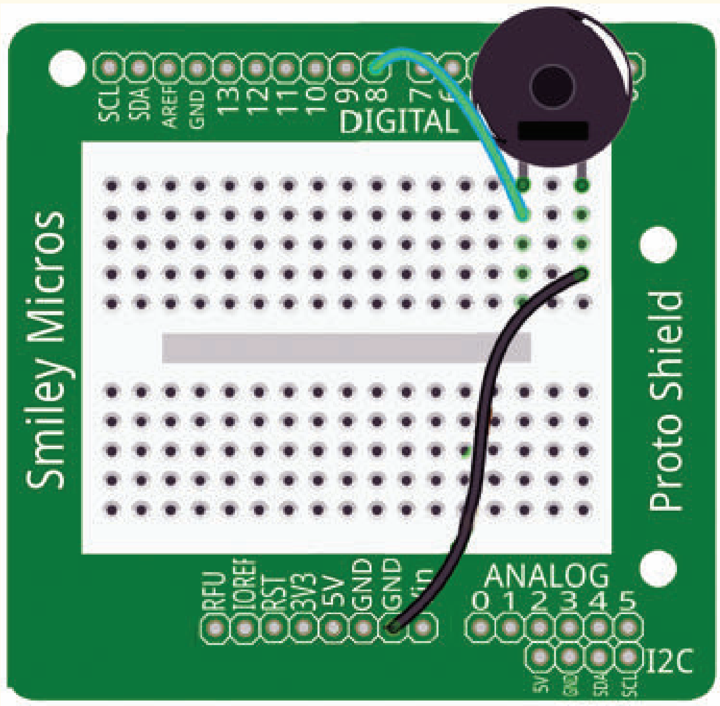


FIGURE 16. Proto shield with piezo buzzer.

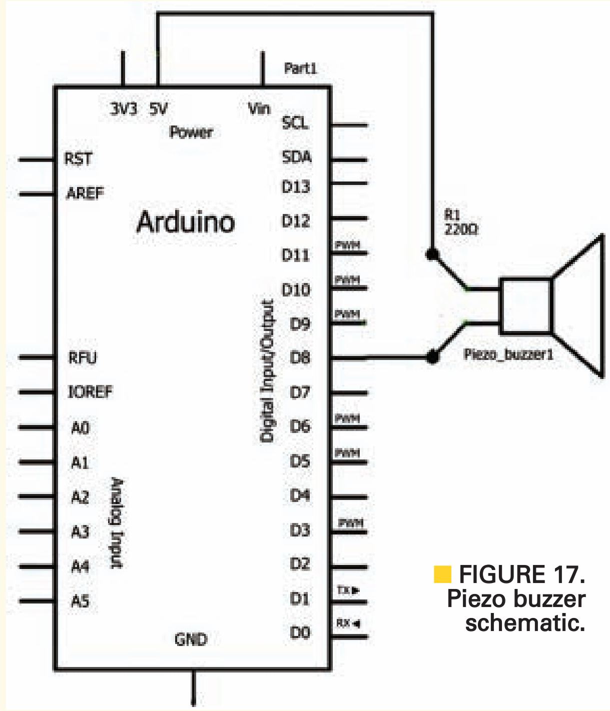


FIGURE 17. Piezo buzzer schematic.

BUILD IT ON THE MINI BREADBOARD

Over the past several Workshops, we've seen how to build a DS1307-based real time clock, so let's apply that here and add a feature we haven't seen yet: an audible alarm. We will use the Fritzing piezo buzzer part we created last month; let's test that first.

You'll need a proto shield with the headers soldered on and a mini breadboard taped to the top like the one shown in **Figure 2**. Plug this into an Arduino, and add a piezo element and wires as shown in **Figure 16**. This circuit is very simple as you can see in the schematic in **Figure 17**.

TESTING THE PIEZO BUZZER

We will use a program that comes with the Arduino IDE to do our preliminary tests. Open the toneMelody file in the Arduino IDE File\Examples\Digital\toneMelody as shown in **Figure 18**. This program should compile and upload to the Arduino without any modifications, and the piezo speaker should start playing the melody.

MAKING A LOUDER ALARM

So, that was kind of underwhelming. Yeah, it plays tunes but not very loud. We see from **Figure 19** that the piezo buzzer volume is highly dependent on the frequency of the tone, so that while we can hear the music, most of the tones being played are fairly low volume. Note that the peaks are around 2,000 Hz and 6,000 Hz.

This is an intentional design tradeoff for the piezo speaker since it wasn't really designed to play music, but to output annoying alarm tones that will grab our attention. Since that is exactly what we want, let's write a program to beep it around those datasheet frequencies

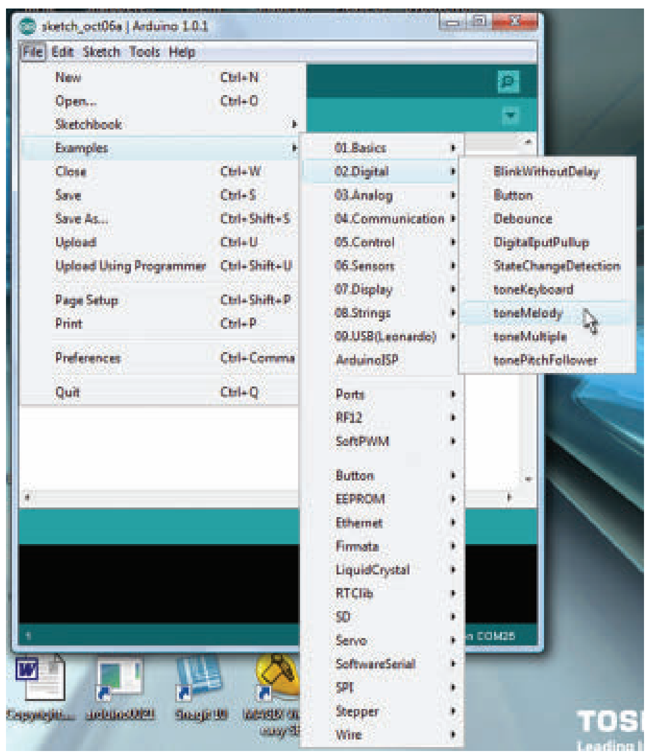


FIGURE 18. Select toneMelody.

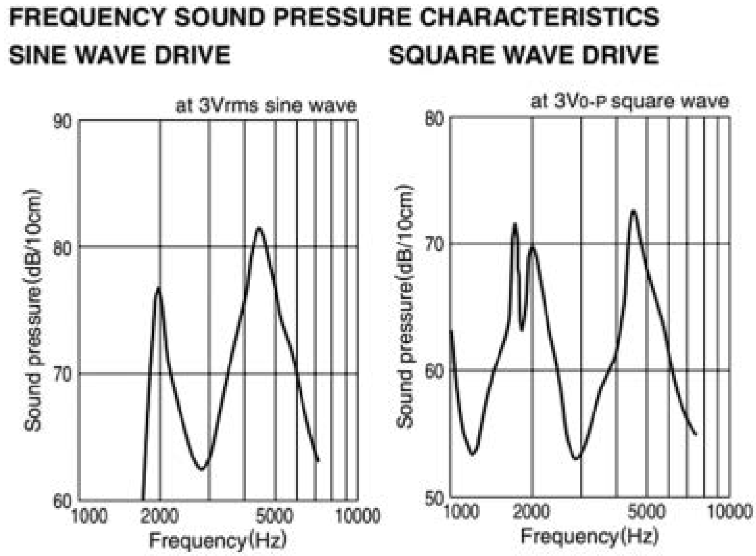


FIGURE 19. From the TDK PS1420P02CT datasheet.

and see if we can pinpoint a value that is loud enough to be worthy of the word alarm.

We first note that by using a real sine wave to drive the piezo versus a PWM square wave we get an additional 10 decibels in sound pressure. Is that worth adding the circuitry to convert the PWM to a sine wave?

Well, the square wave maxes out at a bit over 70 dB which is about as loud as normal table conversation (by normal I don't mean at *my* family dinner table, but say that in a restaurant with babies and toddlers excluded).

A 10 dB increase is perceived as twice as loud. So, 80 dB would be about as loud as a food processor or garbage disposal. That would be ideal for an alarm, but let's listen to the square wave induced tones first and see if they are loud enough for our purposes.

TEST THE TONE VOLUME AS A FUNCTION OF FREQUENCY

You can find this sketch at the article link. The program allows you to play three tones each for 1/4 second, then wait a second before repeating them. You then manually change the frequencies to bracket the loudest tone. [As an aside, this is one of the greatest advantages of the Arduino. You can make minor changes in the code and reload rapidly, allowing you to make quick changes like this for testing purposes.]

I started off at 4,000, 4,500, and 5,000, and ran the program multiple times to bracket in on the tone I thought was loudest and most annoying (which, in my case, is 4,300 Hz). My hearing is a bit weird, so you may want to run your own tests to determine what is loudest and most annoying to you.

```
/* toneTest */

void setup() {
 // do nothing
}

void loop() {
  for(int i = 0; i < 5; i++)
  {
    tone(8,4300);
 // AND THE WINNER IS
 // 4300!
    delay(250);
    tone(8,0);
    delay(250);
  }

  for(int i = 0; i < 5; i++)
  {
    tone(8,4312);
    delay(250);
    tone(8,0);
    delay(250);
  }

  for(int i = 0; i < 5; i++)
  {
    tone(8,4325);
    delay(250);
    tone(8,0);
    delay(250);
  }

  delay(1000);
}
```

THE REAL TIME CLOCK CIRCUIT

With the alarm program complete, we can add this to our earlier work with real time clocks from the July 2012 Workshop. You can construct the circuit shown in **Figures 1** and **2** following the schematic shown in **Figure 20**.

TEST IT WITH ARDUINO SOFTWARE

Jean-Claude Wippler at JeeLabs wrote a most excellent library for using the DS1307 with an Arduino, so rather than reinvent a wheel, let's use his work. You can find it at **http://jeelabs.net/projects/11/wiki/RTClib**. Unzip the library and load it into your Arduino directory at \libraries\RTClib. [This repeats the tests we saw in the July 2012 Workshop, but rather than have to run that down we'll redo it.]

This library provides the DS1307 example shown in **Figure 21**. Open it and upload it to your Arduino with the proto shield. When you first run this, it

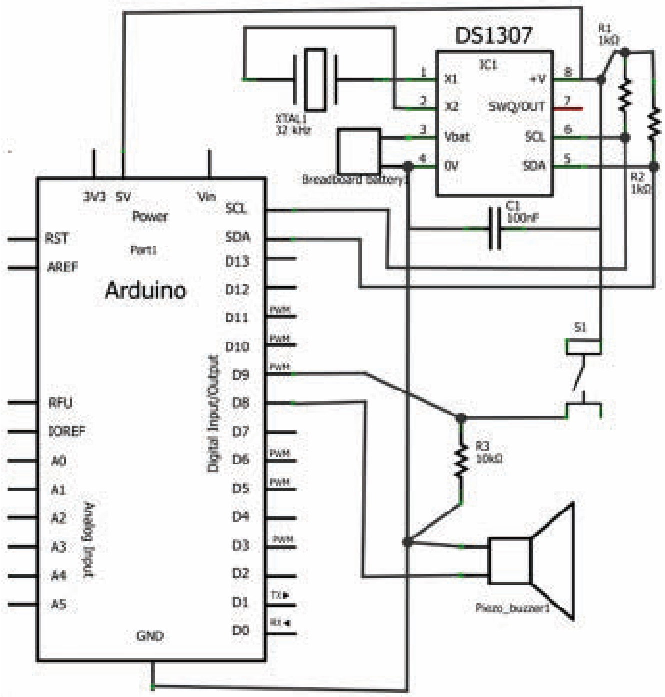


FIGURE 20. Proto shield alarm clock.

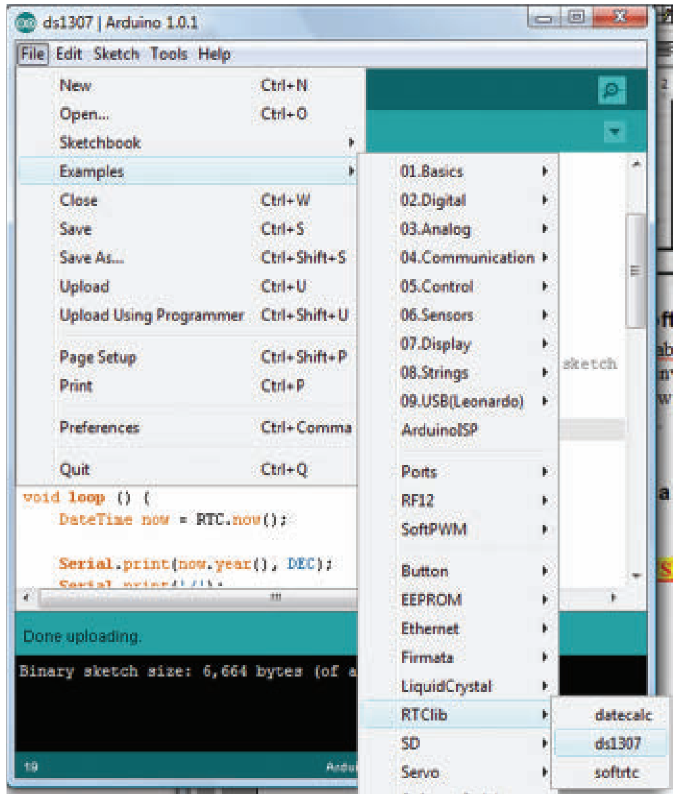


FIGURE 21. Select DS1307 in RTClib.

will set the correct date and time based on what it reads rom your PC. Thereafter, if you keep the backup battery running at three volts (we'll get to that in a minute), it will continue to keep time.

Figure 22 shows the output of the program on the Arduino serial monitor. In the first line, we see the year/month/day hour:minute:second. In the next line, we see the elapsed time since the Unix base time of midnight 1/1/1970 expressed as seconds and days. In the third line, we see the results of calculating a date which is seven days and 30 seconds into the future. Take a look at Jean-Claude's code and you'll see he has given you a really useful toolset for working with the DS1307 RTC.

So, now we have it all put together on a mini breadboard and tested it with some simple Arduino code.

Next month, we will continue with using the Arduino proto shield alarm clock kit by first transferring the parts from the mini-breadboard to the PCB to give us more robust hardware. Then, we'll write software for an alarm clock that is not only useful in itself, but provides a good basis for more advanced data logging projects.

This software will allow us to set alarms from the PC over the USB port, and use that button to shut the alarm off and generate various alarm tone sequences as needed. **NV**

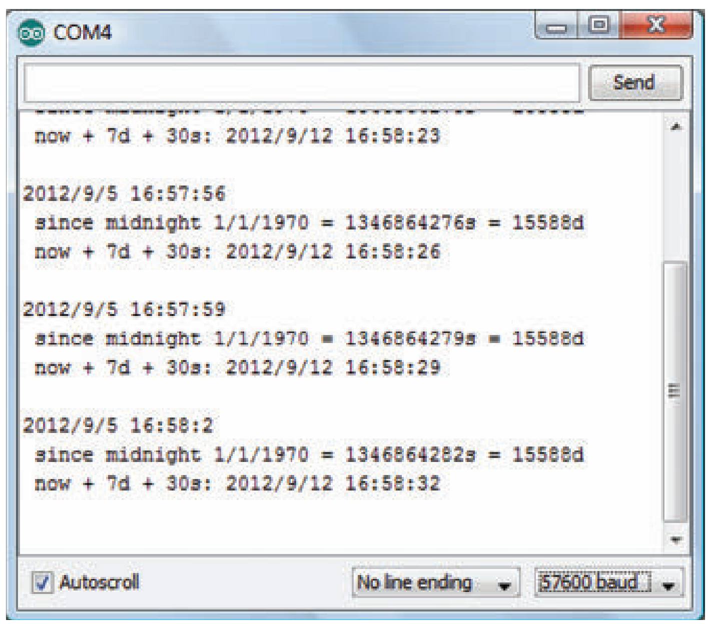


FIGURE 22. Serial monitor input.

OPEN COMMUNICATION

■ BY LOU FRENZEL W5LEF

THE LATEST IN NETWORKING AND WIRELESS TECHNOLOGIES

WHITE SPACE SPECTRUM Benefits All Wireless Applications

In case you haven't heard, there is a spectrum shortage worldwide.

Spectrum, of course, is the electromagnetic radio spectrum that all of our wireless devices occupy. This free space spectrum is allocated by governments to the various radio/TV broadcast and wireless services to avoid interference and conflicts between users.

After years of wireless progress and growth, there is precious little spectrum left for further development. This has become a serious crisis and growth-inhibitor for the cellular operators who need more spectrum to build out their 4G networks, and for wireless Internet providers who want to establish broadband services to rural areas.

One potential solution to part of this problem is white space.

Shared unused TV channels help solve the spectrum crisis.

WHITE SPACE

White space is the name given to unused TV channels. Channels 2 through 51 from 54 MHz to 698 MHz are still used for over-the-air TV across the US. Less than 20% of the population gets their TV over-the-air, but it's still free if you have a TV set.

One important fact is that not all channels are used in all areas. Most local areas and large cities only use a few channels, so there are blank spaces between assigned stations. These unused 6 MHz wide channels are effectively wasted spectrum. The whole idea of white space is to put these channels to use. The problem is that the unused channels are not the same across the country. Channel 21 in one city may not be available in another city. So, the challenge of white space wireless is to figure out how to use the blank channels regardless of their location or frequency assignment.

COGNITIVE RADIO

Cognitive radio is a technology that makes a radio smart. A cognitive radio is a software-defined radio with intelligence. Cognitive radios implement spectrum sharing, that is making it possible for one channel to be used by many at different times and different places as required. What a cognitive radio does is listen on a desired channel to see if it is occupied before transmitting. If the channel is occupied, it switches frequencies and tries another channel. This means that the radio must be frequency-agile so that it can change channels quickly over a wide range of spectrum. This is certainly possible today thanks to fast frequency synthesizers that can set a receiver or transmitter to any desired frequency in microseconds or less.

Cognitive radios also rely on external data bases to determine what channels are available in what geographical areas. The radio desiring to communicate accesses the data base remotely over the Internet to determine available channels. An internal GPS receiver physically locates the transmitter for the data base access. Once available channels have been determined, the communications link can be established.

THE BENEFITS OF WHITE SPACE

One key benefit of white space is that it is unlicensed spectrum. As long as the user deploys approved radio equipment, no for-fee license is required. The primary benefit is the long range coverage that the TV bands permit. Most wireless data services like Wi-Fi have a very short range, typically less than 100 meters and in most cases less than that. Bluetooth, ZigBee, and other data radios have an even shorter range that peaks out at about 10 meters. The reason for this is strictly physics. It is a fact that the shorter the wavelength or the higher the frequency, the shorter the distance a signal can travel. That means that the lower frequency radios can cover a longer distance for the same power, antenna type, and other factors than a higher frequency. A 900 MHz radio can reach farther than a 2.4 GHz radio. This is also true for cell phones and basestations. Cell phones using the 850 MHz band transmit over a longer distance than one using the 1900 MHz band. As a result, the higher frequency cellular systems require more basestations spaced closer together to cover the same area as a system using lower frequencies.

With TV white space frequencies, it is estimated that a range up to 100 kilometers or a little over 60 miles is possible. Furthermore, these lower frequencies — although of the line-of-sight variety — do a better job at penetrating objects along the way such as trees, buildings, and other obstacles.

One downside to these lower frequencies is the need for longer antennas. Antennas at cell phone frequencies are only an inch or so long. For the white space frequencies, antenna lengths run from a foot to nearly 10 feet in length. The lower VHF bands can also produce unusual skip conditions that are wave propagation effects that let the signal travel for hundreds of miles. That is undesirable as it can cause interference.

NATIONAL BROADBAND PLAN

In early 2009, Congress directed the Federal Communications Commission (FCC) to develop a National Broadband Plan to ensure every American has "access to broadband capability."

Congress also required that this plan include a detailed strategy for achieving affordability and maximizing use of broadband to advance "consumer welfare, civic participation, public safety and homeland security, community development, health care delivery, energy independence and efficiency, education, employee training, private sector investment, entrepreneurial activity, job creation and economic growth, and other national purposes."

For more information on the National Broadband Plan, visit **www.broadband.gov/plan/**

WHITE SPACE APPLICATIONS

One of the target applications for white space is broadband wireless access. This refers to Internet access by wireless methods. Most Internet access in homes today is via the cable TV system or DSL on the telephone lines. While some wireless access is available in a few areas, it is not common. White space spectrum makes wireless Internet access practical and affordable. One basestation could serve multiple homes with speeds of 1.5 Mb/s or higher.

Today, there are many rural areas and small towns that still do not have high speed Internet access. The **National Broadband Plan** hopes to provide such access to those underserved areas via wireless. Wireless Internet Service Providers (WISPs) can be established for this purpose. White space basestations and consumer end equipment will be simple and relatively inexpensive.

Another possible application is wireless backhaul. Backhaul refers to the link between a basestation or hot spot and its connection point to the Internet. In cell phone systems, backhaul from the cell sites to the main office is by wired T1 or Ethernet cables, fiber optic cable, or

■ FIGURE 1.
The Carlson Wireless RuralConnect, Generation II white space radios. The unit on the right is the basestation and the device on the left is the customer premise unit designed to be mounted on the antenna mast. Both use 75 ohm coax cable and common F-connectors for the antennas.

microwave radio link. Wi-Fi hot spots use a mix of wired and wireless links. White space radios will make a good low cost and long range backhaul link for hot spots and other wireless services.

M2M is another application possibility. Machine to machine (or M2M) is the connection of sensors for remote monitoring of machines, devices, and facilities. It can also be used for remote control. It is widely used in industry for checking on remote items of interest. Today, M2M is largely implemented with cellular radios over the regular cellular networks. This is expensive. Other shorter range M2M applications may use Wi-Fi or ZigBee, but the range is severely limited. A white space M2M system would allow long range coverage at lower costs.

While most white space applications are initially fixed wireless, it is possible to make mobile or portable white space radios. Maybe they will become an option on laptops, ultrabooks, or tablets.

WIRELESS TECHNOLOGY

Wireless technology refers to the details of how the radio works. It covers modulation, access methods, frequency bands, data protocols, coding, security, and other features. White space can accommodate many different forms of technology, but so far, the industry hasn't settled on just one yet. Right now there are several proprietary systems in use, but it appears as though the industry may be headed to some common standard.

One possibility is Wi-Fi — the IEEE standard called 802.11. It is widely used and has been considered as an option. Its main problem is the wide bandwidth it occupies with its OFDM (Orthogonal frequency-division multiplexing) modulation schemes. For example, the common 802.11n version requires a 22 MHz channel in the 2.4 GHz unlicensed band. Since white space channels are only 6 MHz wide, this is not a good fit. Perhaps multiple channels could be concatenated or aggregated to make it possible.

In any case, the IEEE has formed a study group to see how the Wi-Fi standards can work in the white spaces. Called 802.11af, this standards group is trying to define a way to use some or most of the current Wi-Fi standard in the TV white spaces. You might hear this referred to as white-fi. The term Super Wi-Fi has been applied to white space radios, but the Wi-Fi Alliance is not too happy about the use of their trademarked name. In the meantime, we will have to wait and see what develops.

Another IEEE standard that was developed directly targeting white space is the 802.22 standard. This is called the Wireless Regional Area Network (WRAN). It uses OFDM in the 6 MHz channels to produce a data rate up to 22 Mb/s. The maximum range is up to 60 miles or so under the right conditions. The standard also includes all of the features of a cognitive radio. While 802.22 radios are not available yet, it appears there is a movement to make this "the" white space standard. The industry group White Space Alliance is

■ FIGURE 2. **Carlson's latest basestation antenna.** It covers the 470 to 786 MHz range and is omni-directional. It is vertically polarized and has a gain of 5.8 dBi.

promoting their version called Wi-Far.

While it is not necessary to have a single radio standard, the equipment does have to be compatible within the local area where it is used. A single standard is especially helpful for mobile devices like laptops and tablets that can move from one geographic location to another. We will have to wait and see what standard wins out, if any.

A LOOK AT A WHITE SPACE RADIO

One of the first manufacturers to offer commercial equipment to the white space market is Carlson Wireless. They make a basestation, as well as customer premise equipment. These units are shown in **Figure 1**. The frequency range is 470 to 786 MHz which covers the upper US. TV channels, as well as those in Europe and Japan. It uses the 6 MHz channels in the US and the 8 MHz channels in Europe. The modulation is either QPSK or 16QAM depending on distance, noise levels, and other factors. It can deliver a data rate from 1 to 16 Mb/s depending on the configuration and environmental conditions. Typical distance is in the three to eight mile range. When used in an Internet Service Provider mode, the maximum client capacity is 4,096 users. Yet in a more common arrangement, one unit will serve about 40 clients with 3 Mb/s downstream and 1 Mb/s upstream. Transmit power levels are in the 27 to 28 dBm range. Computer connections are via a standard Ethernet 10/100 cable.

Figure 2 shows Carlson's new basestation antenna. This patent-pending antenna is unique in that it is omni-directional and has gain — a rare combination. Furthermore, it has wide bandwidth and covers the 470 to 786 MHz range. It uses vertical polarization that helps to minimize interference to any nearby TV broadcast channels that use horizontal polarization.

SPECTRUM SHARING

This whole idea of sharing underused spectrum is catching on within the government agencies that regulate spectrum. In the US, this is the Federal Communications Commission (FCC) and the National Telecommunications Information Administration (NTIA). The FCC regulates commercial and personal spectrum while the NTIA regulates spectrum for the government and military. A study of the spectrum indicates there are numerous chunks of under-deployed spectrum. Some of this could be shared with different services. One example the government is looking at is the band from 1,755 to 1,850 MHz. It is currently assigned to military radar and some other lesser used applications. This could, in the future, become another band that will get shared with new wireless services. Other bands may also be identified in the future.

FINAL THOUGHTS

In any case, the idea of white space and shared spectrum offers one good way to provide growth in the wireless industry without overly disrupting existing services and avoiding massive spectrum reassignments. We have the cognitive radio technology developments to thank for this. **NV**

ELECTRONET

For the Gearheads who have it all:
Give A Gift
Get A Gift
Send a friend or loved one a special gift this holiday season.
When you buy or renew your own subscription, get a 2nd One-Year Subscription to *Nuts & Volts* *absolutely* ***FREE!*** (US only).
Both subscriptions include full access to the print, online and mobile editions!
NEW!
Nuts & Volts
Mobile App
Included For
iOS - Android
Kindle Fire
Special 2 for 1 holiday offer
2 one-year subscriptions to
Nuts & Volts for just $26.95!
Act now! *Offer good through December 31st*
That's 2 subscriptions for the price of 1!
Order by phone **(877) 525-2539** Or Online **www.Subnow.com/NV/2For1**
Use promotion code Y2FXMN

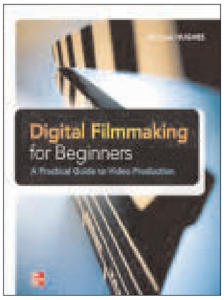
For complete product details, visit our webstore!!

The Nuts & Volts WEBSTORE

GREAT FOR DIYers!

Digital Filmmaking for Beginners A Practical Guide to Video Production
NEW!

by Michael Hughes

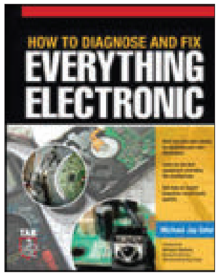


Discover how to create professional-quality digital videos — faster than you can say "lights, camera, action." Digital video cameras are everywhere — even on our phones! But cameras don't make great movies; filmmakers do. Written by a seasoned pro with 40 years of teaching experience, *Digital Filmmaking for Beginners* is your fully illustrated introduction to all technical aspects of digital filmmaking.

$24.95

How to Diagnose and Fix Everything Electronic
by Michael Jay Geier

Master the Art of Electronics Repair!



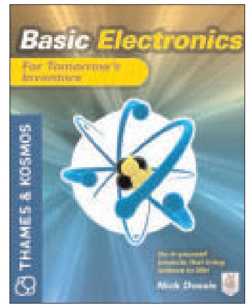
In this hands-on guide, a lifelong electronics repair guru shares his tested techniques and invaluable insights. *How to Diagnose and Fix Everything Electronic* shows you how to repair and extend the life of all kinds of solid-state devices, from modern digital gadgetry to cherished analog products of yesteryear. **About the Author:** Michael Jay Geier began operating a neighborhood electronics repair service at age eight that was profiled in *The Miami News*.

$24.95

Basic Electronics for Tomorrow's Inventors
NEW!

by Nick Dossis

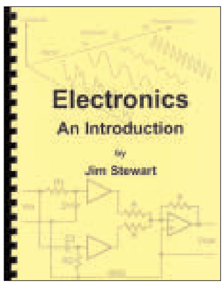
Learn all about electronics with hands-on experiments and projects.



This great book features a variety of simple, solder-free projects, including an LED reading light, electronic security keypad, IR target practice game, a real working telephone, temperature and moisture sensors, spy gadgets, and other neat stuff. Best of all, these experiments require only plug-and-play "breadboards" and other commonly available parts.

$19.95

Electronics An Introduction
by Jim Stewart

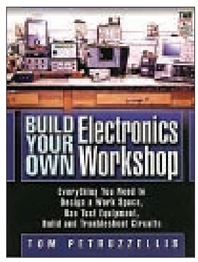


This book is designed as an in-depth introduction to important concepts in electronics. While electronics can be highly mathematical, this text is not about calculations. It is about how electronic equipment is able to extract, process, and present information held in electrical signals. If you are in — or studying to be in — a profession that requires the use of electronic equipment, then this book will provide the insight necessary to use such equipment effectively.

$33.95*

Build Your Own Electronics Workshop
by Thomas Petruzzellis

BUILD YOUR OWN DREAM ELECTRONICS LAB!

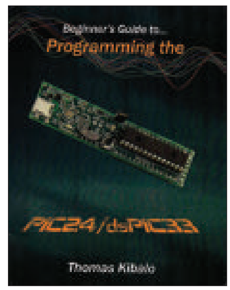


This value-packed resource provides everything needed to put together a fully functioning home electronics workshop! From finding space to stocking it with components to putting the shop into action — building, testing, and troubleshooting systems. This great book has it all! And the best part is, it shows you how to build many pieces of equipment yourself and save money, big time!

Reg Price $29.95

Sale Price $26.95

Beginner's Guide to ... Programming the PIC24/dsPIC33
by Thomas Kibalo

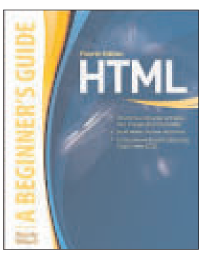


Kibalo takes you step by step through the fundamentals of programming the PIC24H which can equally be applied to the dsPIC33. His clear explanation of the inner workings make learning the PIC24H/dsPIC33 16-bit architecture easy. His code examples demonstrate how to perform the functions most applications require. The hardware is shown in a simple breadboard setup so even a beginner can build it, along with very few extra components needed.

$39.95*

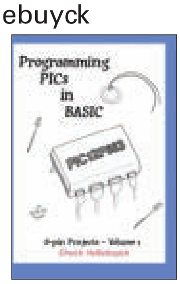
HTML: A Beginner's Guide
by Wendy Willard

Create highly functional, impressive websites in no time. Fully updated and revised, *HTML: A Beginner's Guide*, Fourth Edition explains how to structure a page, place images, format text, create links, add color, work with multimedia, and use forms. You'll also go beyond the basics and learn how to save your own web graphics, use Cascading Style Sheets (CSS), create dynamic web content with basic JavaScript, and upload your site to the web. By the end of the book, you'll be able to build custom websites using the latest HTML techniques.



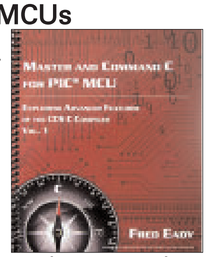
$29.95

Programming PICs in Basic
by Chuck Hellebuyck



If you wanted to learn how to program microcontrollers, then you've found the right book! Microchip PIC microcontrollers are being designed into electronics throughout the world and none is more popular than the eight-pin version. Now the home hobbyist can create projects with these little microcontrollers using a low cost development tool called the CHIPAXE system and the Basic software language. Chuck Hellebuyck introduces how to use this development setup to build useful projects with an eight-pin PIC12F683 microcontroller. **$14.95**

Master and Command C for PIC MCUs
by Fred Eady



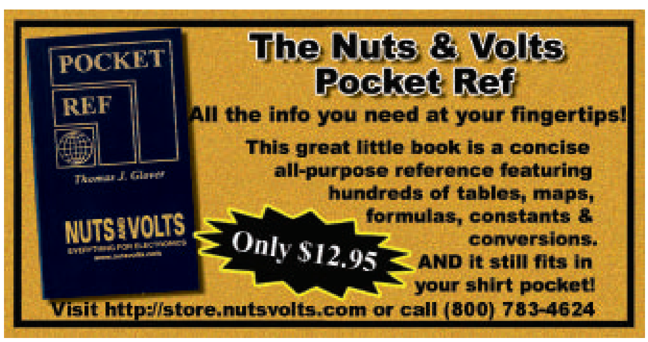
Master and Command C for PIC MCU, Volume 1 aims to help readers get the most out of the Custom Computer Services C compiler for PIC microcontrollers. The author describes some basic compiler operations that will help programmers particularly those new to the craft create solid code that lends itself to easy debugging and testing. As Eady notes in his preface, a single built-in CCS compiler call (output_bit) can serve as a basic aid to let programmers know about the "health" of their PIC code.

$14.95

Order online @ www.store.nutsvolts.com
Or CALL 1-800-783-4624 today!

BOOK & KIT COMBOS

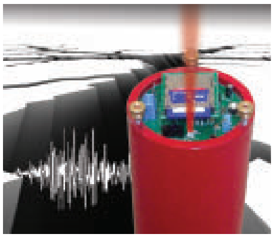
CD-ROM SPECIAL



CALL 1-800-783-4624 today!
Order online @ www.nutsvolts.com

PROJECTS

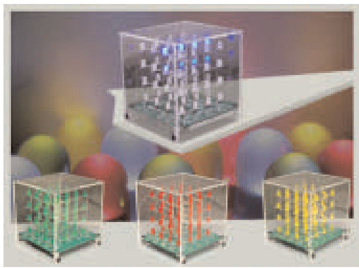
Seismograph Kit



As seen in the May 2012 issue.
Now you can record your own shaking, rattling, and rolling.
The Poor Man's Seismograph is a great project /device to record any movement in an area where you normally shouldn't have any. The kit includes everything needed to build the seismograph. All you need is your PC, SD card, and to download the free software to view the seismic event graph.

Subscriber's Price **$79.95**
Non-Subscriber's Price **$84.95**

3D LED Cube Kit



This kit shows you how to build a really cool 3D cube with a 4 x 4 x 4 monochromatic LED matrix which has a total of 64 LEDs. The preprogrammed microcontroller that includes 29 patterns that will automatically play with a runtime of approximately 6-1/2 minutes.
Colors available: Green, Red, Yellow & Blue

Subscriber's Price **$57.95**
Non-Subscriber's Price **$59.95**

Battery Marvel Kit

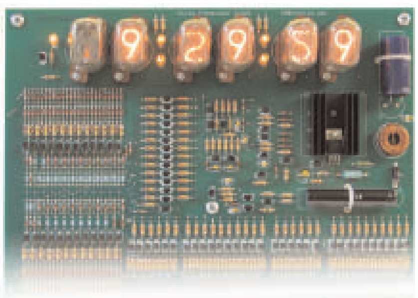
As seen in the November 2011 issue.

Battery Marvel helps protect cars, trucks, motorcycles, boats, and any other 12V vehicles from sudden battery failure. This easy-to-build kit features a single LED that glows green, yellow, or red, indicating battery health at a glance. An extra-loud piezo driver alerts you to any problems.

For more info, please visit our website.

Subscriber's Price **$18.95**
Non-Subscriber's Price **$19.95**

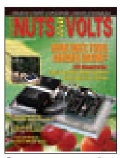
Neon Transistor Clock Kit



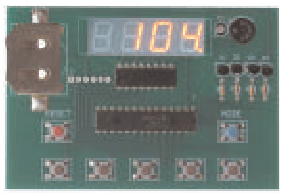
Add HIGH VOLTAGE to your clock!
This is a Nixie Tube display version of the Transistor Clock. It uses only discrete components — no integrated circuits.
For more info, see the April 2012 issue.

Subscriber's Price **$245.95**
Non-Subscriber's Price **$249.95**

Sorting Counter Kit



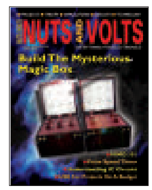
As seen in the July 2011 issue.



Sorting counters have many uses — keeping score, counting parts, counting people — it is just a handy gadget to have around. This is a very simple project for those who want to learn to solder or are interested in using microprocessors and how they function. No special tools are needed, just a small tip soldering iron. It has no box as it stands alone, therefore there is no drilling.

Subscriber's Price **$33.95**
Non-Subscriber's Price **$39.95**

Magic Box Kit



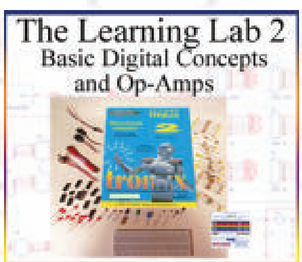
As seen on the April 2007 cover.

This unique DIY construction project blends electronics technology with carefully planned handcraftsmanship. This clever trick has the observer remove one of six pawns while you are out of the room and upon re-entering you indicate the missing pawn without ever opening the box.

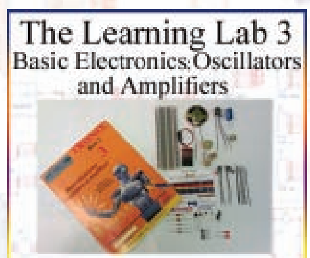
Subscriber's Price **$39.95**
Non-Subscriber's Price **$45.95**

FOR BEGINNER GEEKS!

$59.95



$49.95



$39.95

The labs in this series — from GSS Tech Ed — show simple and interesting experiments and lessons, all done on a solderless circuit board.

As you do each experiment, you learn how basic components work in a circuit, and continue to build your arsenal of knowledge with each successive experiment.

For more info and a promotional video, please visit our webstore.

CLASSIFIEDS

READER FEEDBACK

Continued from page 9

VHF two-meter allocation on a frequency of 144.390 MHz. This is the nationwide APRS frequency in heavy use by the amateur service.

Nowhere in the article is it mentioned that this is an amateur service licensed frequency and that an amateur radio license of Technician Class or higher is required to operate that transmitter on that frequency. A note "or other frequency" is made in the parts list, but no supporting information about why an alternate frequency should be chosen is given.

Please update your readers as to the importance of compliance with FCC rules, regulations, and licensing regarding the radio transmitter discussed in this article. There are similar issues in other countries with other regulatory authorities that will also affect your international readers.

Chris Elmquist
ARO N0JCF
St. Paul, MN

Yes, that's correct. I was assuming it was understood to be a licensed radio operator. Chris brings up a good point and I'll need to be more explicit about the transmitter requiring an amateur radio license. Thanks for bringing that up and I'll make sure to state that in every article with amateur radio equipment.

Paul

THE DESIGN CYCLE

BY FRED EADY

ADVANCED TECHNIQUES FOR DESIGN ENGINEERS

USING THE MPLAB X FACTOR WITH DATA RADIOS

I live to scratch-build embedded hardware and write firmware drivers. So, when I come across an interesting piece of hardware, it gives me an excuse to fold the newly found hardware into a microcontroller-based system that will adequately support it. Once I've absorbed the device's datasheet, I choose a microcontroller that is best suited to meld with the target device. My microcontroller choice is also based on other devices I wish to include in the hardware build. Once the hardware design morphs from virtual to physical, I choose a programming language that is most suitable for the hardware I have assembled. The hardware is verified using small snippets of code that exercise the I/O subsystems of each active device in the design. Once hardware verification is complete, the firmware build begins.

THE X FACTOR

When it comes to features, the CCS C compiler is unrivaled. I particularly like their built-in millisecond delay routine. The C compiler also takes the tedium out of deploying the Microchip PIC's UARTs. During the "think time" of the system design we're about to implement, I decided to use the CCS C compiler in conjunction with the PIC18F46J13 microcontroller to drive the SPI portal that links the target device to the PIC.

The version of the C compiler I will use is designed to run within an IDE (Integrated Development Environment). The C compiler's IDE feature allows it to function under the control of a number of differing development tools. If you choose to purchase the IDE version of the CCS compiler, it comes out of the box with the IDE. The C compiler can also run under the umbrella of Microchip's IDEs.

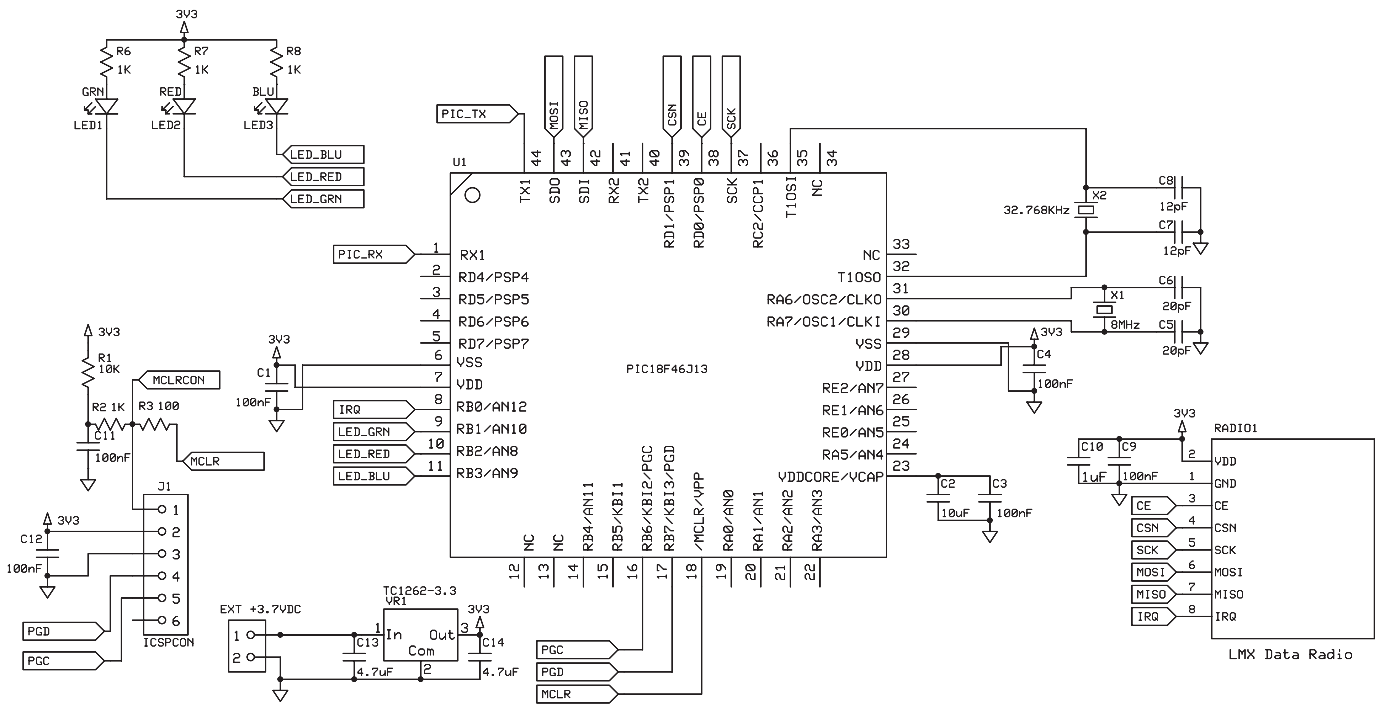
I'm very comfortable with the old school Microchip MPLAB 8 IDE. However, for this project, I've decided to break the paradigm and start developing with the new MPLAB X environment. There are many advantages to moving to the X development environment. I can choose to use any of the Microchip C compilers for firmware development. I can also select various third party compilers such as the one from CCS. If I decide that BASIC is the language of choice for a project, microEngineering Labs's PICBASIC PRO is X capable.

My debugger/programmer of choice these days is the Microchip PICkit 3. The PICkit 3 is capable of programming and debugging the majority of Microchip's line of PICs. We will use the PICkit 3 to move firmware back and forth between the PIC18F46J13 and MPLAB X.

THE PIC18F46J13 SYSTEM

The PIC18F46J13 was chosen for this project because of its unique capabilities and absence of built-in USB. The addition of a 32.768 kHz crystal enables us to call upon the services of the PIC18F46J13's internal RTCC (Real Time Clock Calendar) module. This PIC sleeps deeply and the RTCC is fully capable of awakening it at predefined intervals. Just in case we need them, the PIC18F46J13 is loaded with a pair of UARTs and a pair of SPI portals. The secondary UART and SPI portals can be moved about the PIC's pin set at will using its PPS (Peripheral Pin Select) mechanism.

The PIC18F46J13 can operate at battery-generated voltage levels. Combining battery power with the PIC's snooze capability makes for a stand-alone battery-operated system that can operate independently for very long periods of time. The battery-powered operation is very



■ SCHEMATIC 1. There are some hidden resources lurking in this simplistic design. The PIC18F46J13 contains a powerful internal RTCC module and is capable of doubling the UART and SPI interfaces. The PIC18F46J13 is a perfect choice to drive the LMX data radio.

important to this project as we will be writing a driver for the new series of LMX low-power data radios.

Our initial LMX hardware design will be relatively simple as our goal is to build a suitable firmware driver for the data radio. With that, all we will really need are a few LED status indicators and an RS-232 port.

If you've already taken a peek at **Schematic 1**, you've noted that there is an RS-232 interface defined but no RS-232 hardware has been drawn up. That's because the RS-232 circuitry exists on an external Digilent PModUSBUART module.

The PModUSBUART is a standard implementation of the FTDI FT232R USB UART IC. Including the PModUSBUART in our design eliminates the need for the microcontroller to be USB capable. In addition, the PModUSBUART can be removed and need not be considered in the battery life calculations.

Like the microcontroller design, the power supply design is simple and efficient. A Microchip LDO (Low Drop Out) voltage regulator and a couple of ceramic capacitors are all that is needed to insure that the circuitry is fed from a 3.3 volt power rail.

The PIC18F46J13 is pin compatible with its USB-enabled cousin — the PIC18F46J50. The 46J50's VUSB pin is recovered at RC3 on the 46J13. The 46J50 USB D- and D+ pins are also recovered by the 46J13 as I/O pins RC4 and RC5, respectively. This pin-to-pin relationship allowed me to assemble the bulk of the 18F46J13 hardware on a 18F46J50 circuit board we used in a previous Design Cycle project.

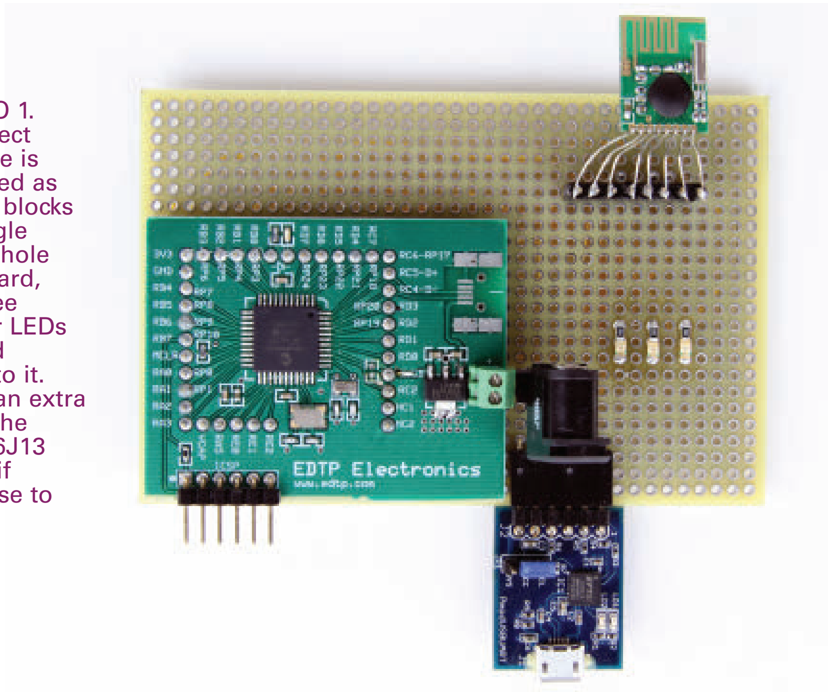
The LMX data radio is tiny, and so is its interface. Four of the eight 1.27 mm pitched pins present the LMX's SPI portal to the 18F46J13. An IRQ pin is included in the LMX pinout mix to alert the 18F46J13 of incoming packet activity. The other LMX control pin, CE, is an input pin that is driven by this same PIC. The state of the LMX's CE pin determines if the data radio is in transmit or receive mode. The two remaining LMX pins connect the data radio to the system power.

The PModUSBUART module, PIC18F46J13 assembly, and LMX data radio are shown in **Photo 1**. I used tinned hookup wire to adapt the LMX to a standard 0.1 inch pitch male header which is plugged into a similar pitched female header. The LMX data radio support hardware has turned out to be a collection of electronic building blocks assembled on a plated-through breadboard.

FROM PINS AND PADS TO BITS AND BYTES

The CCS compiler and MPLAB X IDE got along from the time they were paired. However, the PIC18F46J13 wasn't as happy with the C compiler as I figured it would be. The very first thing for us to accomplish when writing a driver is to code functions that establish a data communications path between the microcontroller and the target device. In this case, the target device is an SPI slave.

The natural thing to do when using the C compiler is to take advantage of its built-in functions.



■ PHOTO 1. The project prototype is assembled as building blocks on a single through hole breadboard, with three indicator LEDs mounted directly to it. There's an extra LED on the PIC18F46J13 module if we choose to use it.

Since the LMX data radio communicates with the PIC18F46J13 using SPI, it's a no-brainer to issue the compiler's *spi_read()* and *spi_write()* function calls.

Two days later — after invoking hundreds of *spi_read()* and *spi_write()* function calls — I was about to throw in the towel on the use of the CCS compiler in this project. I just could not get the LMX data radio to spit out the data that I thought I was inserting into its registers. I decided to put my Saleae logic analyzer to work on the problem. This logic analyzer has the ability to decode an SPI frame. **Logic Capture 1** was generated using the CCS compiler *spi_write()* function. Here's the code:

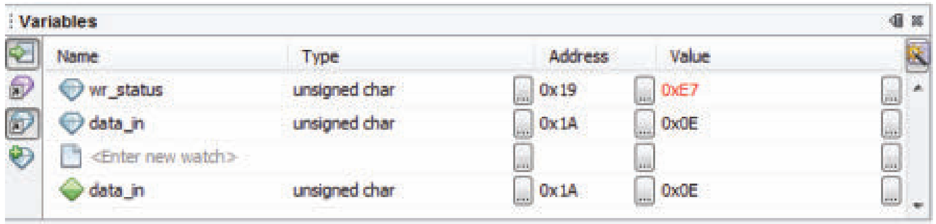
```
output_bit(rCSN,0);
  //select the LMX Data Radio
spi_write(0x55);
  //send 0x55 out on MOSI
output_bit(rCSN,1);
  //release the LMX Data Radio
```

As you can see, the *rCSN* enable signal denotes the beginning and end of the SPI frame. There are eight *SCK* clock pulses in the frame, and the 0x55 was correctly transmitted by the PIC18F46J13's MOSI (Master Out Slave In) pin.

The LMX *STATUS* byte is clocked in on the PIC's MISO (Master In Slave Out) pin. Everything looked okay on the logic capture, so I went ahead and coded up a LMX register write/read sequence:

■ LOGIC CAPTURE 1. This sanity check has been brought to you by Saleae logic. This capture verifies that the SPI signals are all doing what is natural.

■ SCREENSHOT 1. The value received as the STATUS byte is believable per the LMX data radio datasheet. However, 0x0E — the clocking in CONFIG register value — is not equal to 0x0F — the clocked out CONFIG register value.




```
output_bit(rCSN,0);
  //select the LMX Data Radio
spi_write(0x00);
  //write a zero to CONFIG
  //register (address 0x00)
wr_status = spi_read();
  //read returned STATUS byte
spi_write(0x0F);
  //write data byte to CONFIG
  //register
wr_status = spi_read();
  //read returned STATUS byte
output_bit(rCSN,1);
  //end SPI write to CONFIG
  //register
output_bit(rCSN,0);
  //select the LMX Data Radio
spi_write(CONFIG);
  //write CONFIG address to
  //LMX Data Radio
wr_status = spi_read();
  //read returned STATUS byte
data_in = spi_read(0);
  //send 8 clocks the LMX Data
  //Radio
output_bit(rCSN,1);
  //end SPI read of CONFIG
  //register
```

The sequence starts by selecting the LMX data radio as an SPI slave. This is done by driving the SPI chip select line *rCSN* (Chip Select Not) logically low. The LMX *CONFIG* register is located at address 0x00. The *CONFIG* address byte is followed by the data we wish to load into the *CONFIG* register (0x0F). If you check the PIC18F46J13 datasheet, we must read the returned *STATUS* byte to avoid overrunning the PIC's SPI buffer register *SSP1BUF*. Since the C compiler's user's manual does not specifically state that the built-in SPI functions do this buffer read, we'll perform the *SSP1BUF* reads following each *spi_write()* function call.

Immediately following the write to the LMX *CONFIG* register, we drop the *rCSN* line again to kick off an SPI read cycle. After performing the obligatory *SSP1BUF* read, we generate eight clock pulses on the PIC18F46J13's *SCK* line by invoking the *spi_write()* function with a zero as the argument. The eight clock pulses clock in eight bits of data that represent the contents of the LMX data radio's *CONFIG* register.

Houston, we have a problem. Take a look at **Screenshot 1**. The data we clocked out with an SPI write to the LMX data radio *CONFIG* register is not the data we clocked in with the SPI read.

I checked, double-checked, and quadruple-checked the C compiler SPI setup values. Everything looked good but nothing was working as designed. So, in desperation, I started to carefully reread the PIC18F46J13 datasheet SPI section. I decided to load the C compiler with all of the PIC18F46J13's SPI control registers and start twiddling bits that I could not twiddle with the stock compiler.

Adding PIC18F46J13 registers is a relatively easy task and has some benefits. Here is how I added all of the PIC18F46J13's primary SPI control registers:

```
#byte  SSP1CON2 = 0x0FC5
#byte  SSP1CON1 = 0x0FC6
#byte  SSP1STAT = 0x0FC7
#byte  SSP1ADD  = 0x0FC8
#byte  SSP1BUF  = 0x0FC9
```

These SPI register locations were inserted in the main program file. The *#byte* directive is used to associate a variable with a memory location. This is a dangerous memory control function as you can easily place more than one variable at the same memory location. Since we're using the official Microchip naming convention and the official Microchip address for the variables, we are safe.

For instance, if the C compiler already knows about *SSP1BUF*, it does not allow the *#byte* address allocation to take place. Now that we have the PIC18F46J13's primary SPI control registers in place, let's fill out the bits:

```
#define SMP    SSP1STAT,7
#define CKE    SSP1STAT,6
#define BF     SSP1STAT,0

#define WCOL   SSP1CON1,7
#define SSPOV  SSP1CON1,6
#define SSPEN  SSP1CON1,5
#define CKP    SSP1CON1,4
#define SSPM3  SSP1CON1,3
#define SSPM2  SSP1CON1,2
#define SSPM1  SSP1CON1,1
#define SSPM0  SSP1CON1,0
```

Again — to be safe and to make the code easy to follow — we use the Microchip datasheet names for the bits. It's a bunch of work, but the benefits we gain in our code are well worth it. Now we can replace the C compiler built-in output_bit(x,y) function with the simpler *bit_set()* and *bit_clear*() built-in functions.

Armed with my new code cannon, I went about replacing the C compiler-generated SPI setup string with my own homegrown code. This is how the CCS compiler sets up a primary SPI portal:

```
#USE SPI (MASTER, SPI1,
ENABLE=PIN_D1, MODE=0, BITS=8,
ENABLE_ACTIVE=0)
```

There's nothing wrong with the C compiler *#USE SPI* directive. The root of our problem lies in the fact that we don't know what the LMX data radio's SPI portal wants to see. The data radio datasheet is a bit weak in this area. However, we'll clear that discrepancy up in our discussion here.

I followed the PIC18F46J13 datasheet's recommendations on setting up a master SPI portal. The resultant code looked like this:

```
bit_clear(SSPEN);
  //disable the SPI1 module
SSP1CON1 = 0b00000000;
  //clear the register
SSP1STAT = 0b00000000;
```

```
   //clear the register
bit_set(SSPM1);
   //Master Mode with the
   //slowest clock (divide
   //by 64)
bit_clear(CKE);
   //transmit data on idle to
   //active clock transition
bit_set(SSPEN);
   //enable the SPI1 module
bit_set(rCSN);
   //deselect LMX Data Radio
```

At this point, everything looks good as the new code I inserted compiled successfully. The next logical step was to rewrite the code we used earlier to write and read the data radio's *CONFIG* register:

```
void spi_wrt_reg(BYTE addr,
BYTE dataout)
{
  addr |= W_REGISTER;
  bit_clear(rCSN);
  SSP1BUF = addr;
  while(!bit_test(BF));
  lmx_status = SSP1BUF;

  SSP1BUF = dataout;
  while(!bit_test(BF));
  lmx_status = SSP1BUF;
  bit_set(rCSN);
}
```

The address of the data radio is combined with its *W_REGISTER* command (0x20) before being transmitted on the PIC18F46J13's MOSI pin. The *W_REGISTER* command bit pattern takes this form where the command occupies the three most significant bits and the letter *A* denotes an address bit:

```
001A AAAA
```

Note the generous use of the C compiler's built-in *bit_set()* and *bit_clear* functions. When the combined *addr* byte is loaded into *SSP1BUF*, the Buffer Full status bit (BF) is set and the SPI engine begins to clock it out bit by bit. When all eight bits of *addr* are clocked out, the *BF* bit is cleared by the PIC18F46J13's SPI engine.

Sensing the clear *BF* bit, our code moves to read the data radio *STATUS* bit that was clocked in as the *addr* byte was clocked out. This is the status byte read that prevents an *SSP1BUF* overflow error. With *SSP1BUF* clear, we load the data byte that is destined for the data radio's *CONFIG* register. Once the *BF* flag indicates that all of the bits have left the building, we read the status byte and end the session by raising the slave chip select line (rCSN).

To check the accuracy of our new code cannon, the next step is to implement a homegrown SPI read register routine:

```
BYTE spi_rd_reg(BYTE addr)
{
  BYTE datain;

  bit_clear(rCSN);
  SSP1BUF = addr;
  while(!bit_test(BF));
  datain = SSP1BUF;
  SSP1BUF = 0x00;
  while(!bit_test(BF));
  datain = SSP1BUF;
  bit_set(rCSN);
  return(datain);
}
```

The LMX data radio *R_REGISTER* command is binary 000. Since its register set address range is less than six bits in length, we don't have to physically combine the register address and *R_REGISTER* command. The *R_REGISTER* command looks like this:

```
000A AAAA
```

With that, we can immediately drop the combined command/ address byte (addr) into the SPI buffer. The command/address information is followed by a dummy transmission which generates eight clock pulses that clock the contents of the *CONFIG* register into the *SSP1BUF*.

The code we just assembled is invoked in this manner:

```
spi_wrt_reg(CONFIG,0x0F);
data_in = spi_rd_reg(CONFIG);
```

You don't see a new debug window graphic because this code produced the same results as our C compiler code. Now what? I'm beginning to doubt the validity of the *STATUS* byte, as well.

Okay. Let's look at what we know to be true. The data radio datasheet tells us that the SPI clock emitted by the PIC18F46J13 should be logically low during idle. The data radio command sequence is also very clear. To write data to the data radio register, we send the command/address byte followed by the data to be loaded into the register. Reading an LMX data radio register involves providing the data radio with the address of the register you want to read, and clocking in the data from the register. The data radio SPI portal's top speed is 8 MHz. Our PIC18F46J13 system clock runs at 12 MHz. No matter how you look at it, dividing 12 MHz by 64 is less than 8 MHz.

I was so desperate that I swapped the MOSI and MISO lines between the PIC18F46J13 and the data radio. Just to make sure the PIC was clocking, I pushed some data out of the RS-232 port. In the end, the MOSI and MISO lines were wired correctly and I easily sent RS-232 data to a terminal emulator. Naturally, I checked all of the solder joints and wire wrap connections. I even blinked the LEDs.

Blinking LEDs usually makes me feel better, but it does nothing to solve the bigger problem. I've had TRIS settings cause problems, as well. Check these against **Schematic 1** and you'll see that everything is in order:

```
//1 = input pin - 0 = output
//pin
set_tris_a(0b11111111);
set_tris_b(0b11110001);
set_tris_c(0b10010011);
set_tris_d(0b11111000);
set_tris_e(0b11111111);
```

SOURCES

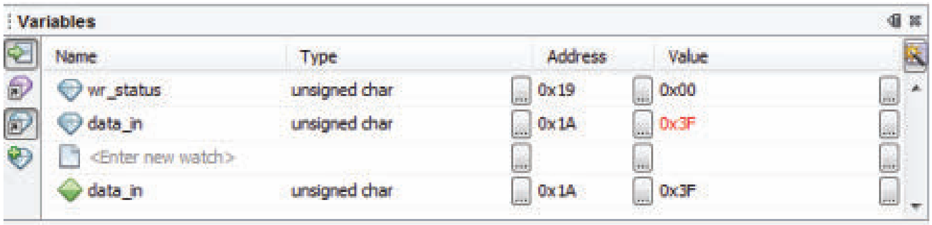
Custom Computer Services
CCS C Compiler
www.ccsinfo.com

Lemos International
LMX Data Radio
LMX Data Radio Hardware Development Kit
www.lemosint.com

Microchip
MPLAB X
PIC18F46J13
www.microchip.com

microEngineering Labs
PICBASIC PRO
www.melabs.com

Saleae LLC
Logic Analyzer
www.saleae.com



■ SCREENSHOT 2. Using AX-MicroLab is akin to going to the supermarket with unlimited funds.

Now for things we aren't sure of. The data radio datasheet does not specify when to sample the data. Nor does it tell us when to present the data relative to the SPI clock. In our code, we have exposed the bits that determine where to sample the data (SMP) and when to present it relative to the SPI master clock (CKE).

Twiddling the *SMP* bit did nothing to change the results. Looking back at our SPI setup code, the *CKE* bit is clear indicating that the data is transmitted on the rising edge of the clock. So, let's set *CKE* and see what happens. Setting *CKE* will transmit the data on the falling edge of the SPI master clock. We'll run our homegrown code first:

```
spi_wrt_reg(CONFIG,0x0F);
data_in = spi_rd_reg(CONFIG);
```

Remember Herman's Hermits? Something tells me I'm into something good. Take a look at **Screenshot 2**.

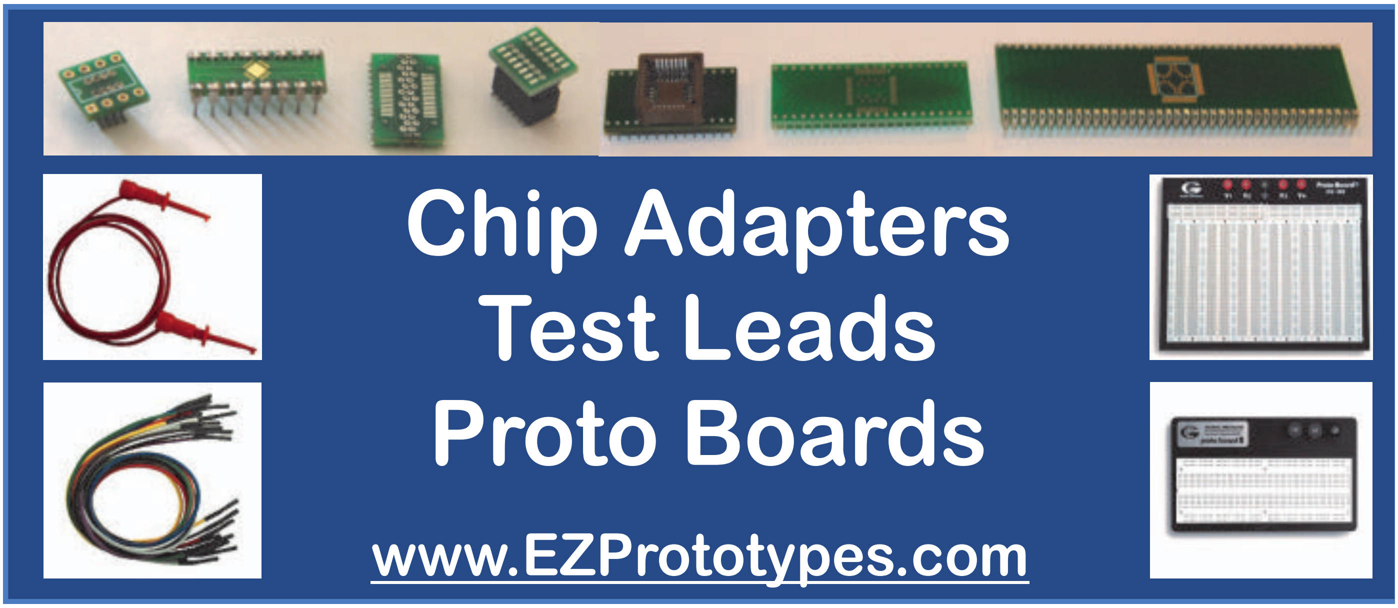
THE FLOOD GATES ARE OPEN

Now that we have the ability to reach out and touch the LMX data radio registers, we can begin to assemble more complex functions. For instance, some of the data that is passed between the PIC18F46J13 and the data radio is in multi-byte format. That means we will need to read and write multiple bytes which will most likely be stored and retrieved from a set of SRAM and ROM buffers we will allocate in the PIC18F46J13's program and data memory.

The data radio is available from Lemos International. I'm building the Design Cycle LMX data radio project from scratch. However, you can get a data radio hardware development kit from Lemos International.

Meanwhile, I'll continue to work on the data radio firmware driver so you can add it to your Design Cycle. **NV**


SurplusGizmos.com
We Sell The Gizmos That Bring Your Ideas To Life
TEST Cables and Probes
60 MHZ Scope from $20.00
Banana to Mini grabber $3.50
Banana to Probe $1.75
Alligator to Alligator 10pk $4.50
Tons of LEDs!
Blinking 5mm
RGB shifting
1.8mm to arrays
Surface Mount to
large 20 Watt units
Waterproof Strips
and much more!
Watch Our Site For
Holiday Sale Information
We Have Several
This month!
ATMEL DEV TOOLS
Atmega prototype board STK48 $35
Atmega-Lite Arduino compatible $6.75
STK500/AVRISP Atmel programmer $38
USB to TTL 5v/3.3 cables FTDI $16.50
USB to TTL board $11.50
M48+EX Development Board $22.50
Solder
Stations!
RoHS and
Non-RoHS
0 to 99v DC LED
Panel Meter $8.50
In Red, Green, or Blue
Enter Code
"SNOWY12"
Online To Get
A Free Gizmo
With Your
$10 Order!
"Retro Style"
Cool Mobile
Handsets make
for a neat gift!
Easy to build Easy to hack
GPS Kits
MERRY
CHRISTMAS!
www.SurplusGizmos.com
503-439-1249
Retail Store Location:
5797 NW Cornelius Pass Road
Hillsboro Oregon 97124
SurplusGizmos,
Your Science and Hobby Electronics Superstore!


Chip Adapters
Test Leads
Proto Boards
www.EZPrototypes.com


PBP3
PICBASIC PRO™ Compiler
Experimenter - $49.95
Silver Edition - $119.95
Gold Edition - $269.95
The industry-standard BASIC compiler for Microchip PIC® microcontrollers.
Multi-Seat Licensing
for Educational Institutions
Upgrade from
PICBASIC™ Compiler (PBC)
Download a FREE trial version now.
www.PBP3.com
microEngineering Labs, Inc.
www.melabs.com
888-316-1753
PICBASIC and PICBASIC PRO are trademarks of Microchip Technology Inc. in the USA and other countries. PIC is a registered trademark of Microchip Technology Inc. in the USA and other countries.

Imagine a complete analog circuit design lab...

Without the lab!

Dual Channel Oscilloscope

- Fully differential inputs, up to 100 MSPS, 14-bit ADC, up to 16KSa/channel memory
- 1MΩ/24pF inputs, ±20V Max
- Real-time FFTs, X-Y plots, complex math and measurements on all channels
- 5 MHz analog signal bandwidth

Dual channel Function Generator

- Single-ended outputs, up to 100 MSPS, 14-bit D/A, up to 16KSa/channel memory
- Arbitrary and pre-defined waves up to ±5V
- Sweeps, envelopes, AM and FM modulation
- Bode plot feature using standard, Nyquist & Nichols coordinates
- Can drive 50Ω

Digital I/O

- 16 signals, 100MSPS, 4KSa/channel memory
- Any signal can be input (logic analyzer) or output (pattern generator or virtual I/O)
- Supports cross-triggers with scope channels

Power Supplies

- Two fixed ±5V, 50mA

Designed in cooperation with:

ANALOG DEVICES XILINX®

ANALOG DISCOVERY™

Turn your PC into a powerful circuit learning platform that can measure, visualize, analyze, record and control mixed signal circuits! The Analog Discovery is small enough to fit in your pocket and costs less than a textbook, but it is powerful enough to replace a stack of lab equipment. Powered by a Hi-Speed USB port and the free WaveForms software, the Analog Discovery lets you build and test analog and digital circuits outside of the lab.

$199.00
(List price)

For courseware & more information, please visit:
www.digilentinc.com/AnalogDiscovery

DIGILENT®
BEYOND THEORY

READER-TO-READER TECHFORUM

>>> QUESTIONS

Visit our Tech Forum online at **www.nutsvolts.com/tech-forum** to view and answer new questions as they come in.

Does anyone have a schematic for a simple light meter? I'm trying to determine the difference in illumination provided by CFLs as opposed to incandescent light bulbs. I'm not too interested in the absolute value of the light hitting the surface, just the relative difference between the two.

#12121 **Pete Schestopol**
Marietta, GA

>>> ANSWERS

[#10124 - October 2012]
Vintage Parts Hunter

Does anyone have or happen to know where I can obtain a pair of 455 kHz IF transformer cans, and a Local oscillator coil for my homebrew AA five tube super heterodyne AM receiver?

I am using the 12BE6, 12BA6, 12AV6, 50C5, and my friendly 35W4 to build my set. Are there any "old new stock parts" establishments still around these days that may have the vintage parts I require?

I know I can still buy yesteryear radio sets that have what I need. However, I would be more likely to repair them as they are, than to scrap any of them for the parts.

I have my trusty 365 pF dual gang air core variable tuning capacitor with the reduced capacity second rotor set. I hope to have no difficulty obtaining all the rest of the parts, as well.

Can any of you high tech experts direct me to where I can get the above named parts I am lacking? Thanks for any assistance.

Look at **www.tubesandmore.com** They have both local oscillator coils and IF transformers for AM tube radios listed. P-C208B, P-C70-OSC, and P-C78 look like the ones.

Jim English
Mission, KS

[#10126 - October 2012]
Remagnetizing Coils

I have not used my turntable on my sound system for some years, but the other day, I had the urge to rediscover some of my old vinyl discs.

To my dismay, there was no sound when I put on a disc. Troubleshooting led to the phono cartridge, an AKG P8ES.

I went online to see if any of these are still available, and at the same time came across some audio blogs describing the exact same problem being that these particular cartridges are moving magnet types and appear to be prone to losing their magnetism over a period of time. This does not occur with moving coil cartridges. Would anyone know of a way to remagnetize the coils in these types of cartridges?

Chances are the magnets are okay on your stylus; the problem is most likely with the coil system in the cartridge body. The coils are not to be magnetized; their job is to generate a voltage from the mechanical movement of the magnets on the stylus shank when playing a record. My experience with the AKG P8 is that it does not age well, so rather than fight a losing battle I suggest replacing it with something new. I just purchased a Shure M97xe and I have no complaints.

Kevin Kaas, C.E. SPC-TV
Largo, FL

Send all questions and answers by email to **forum@nutsvolts.com** or via the online form at **www.nutsvolts.com/tech-forum**

All questions *AND* answers are submitted by *Nuts & Volts* readers and are intended to promote the exchange of ideas and provide assistance for solving technical problems. All submissions are subject to editing and will be published on a space available basis if deemed suitable by the publisher. Answers are submitted by readers and **NO GUARANTEES WHATSOEVER** are made by the publisher. The implementation of any answer printed in this column may require varying degrees of technical experience and should only be attempted by qualified individuals.

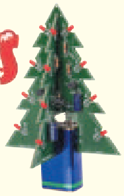
Always use common sense and good judgment!

LOOK FOR ■ SEARCH FOR ■ FIND

Find your favorite advertisers here!

ADvertiser INDEX

Get Ready For The Holidays With a Ramsey Kit!

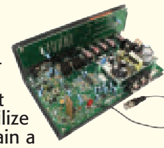


Super-Pro FM Stereo Radio Station

- ✔ PLL synthesized for drift-free operation
- ✔ Built-in mixer - 2 line inputs and one microphone input, line level monitor output!
- ✔ Frequency range 88.0 to 108.0, 100 kHz steps
- ✔ Precision active low-pass "brick wall" audio filter!
- ✔ Dual LED bar graph audio level meters!
- ✔ Automatic adjustable microphone ducking!
- ✔ Easy to build through-hole design!

This professional synthesized transmitter is adjustable directly from the front panel with a large LED digital readout of the operating frequency. Just enter the setup mode and set your frequency. Once selected and locked you are assured of a rock stable carrier with zero drift. The power output is continuously adjustable throughout the power range of the model selected. In addition, a new layer of anti-static protection for the final RF amplifier stage and audio inputs has been added to protect you from sudden static and power surges.

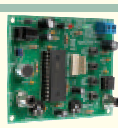
Audio quality is equally impressive. A precision active low-pass brick wall audio filter and peak level limiters give your signal maximum "punch" while preventing overmodulation. Two sets of rear panel stereo line level inputs are provided with front panel level control for both. Standard unbalanced "RCA" line inputs are used to make it simple to connect to the audio output of your computer, MP3 player, DVD player, cassette deck or any other consumer audio source. Get even more creative and use our K8094 below for digital storage and playback of short announcements and ID's! In addition to the line level inputs, there is a separate front panel microphone input. All three inputs have independent level controls eliminating the need for a separate audio mixer! Just pot-up the source control when ready, and cross fade to the 2nd line input or mic! It's that simple! In addition to the dual stereo line inputs, a stereo monitor output is provided. This is perfect to drive studio monitors or local in-house PA systems.



The FM100B series includes an attractive metal case, whip antenna and built in 110/220VAC power supply. A BNC connector is also provided for an external antenna. Check out our Tru-Match FM antenna kit, for the perfect mate to the FM100B transmitter. We also offer a high power kit as well as an export-only assembled version that provides a variable RF output power up to 1 watt. The 1 watt unit must utilize an external antenna properly matched to the operating frequency to maintain a proper VSWR to protect the transmitter. *(Note: The FM100B and FM100BEX are do-it-yourself learning kits that you assemble. The end user is responsible for complying with all FCC rules & regulations within the US or any regulations of their respective governing body. The FM100BWT is for export use and can only be shipped to locations outside the continental US, valid APO/FPO addresses or valid customs brokers for documented end delivery outside the continental US).*

FM100B	**Super-Pro FM Stereo Radio Station Kit, 5uW to 25mW Output**	**$269.95**
FM100BEX	**Super-Pro FM Stereo Radio Station Kit, 5uW to 1W Output**	**$349.95**

Audio Recorder & Player



Record and playback up to 8 minutes of messages from this little board! Built-in condenser mic plus line input, line & speaker outputs. Adjustable sample rate for recording quality. 4-switch operation that can be remote controlled! Runs on 9-12VDC at 500mA.

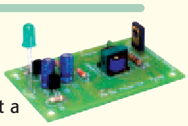
K8094 Audio Recorder/Player Kit $32.95

Water Sensor Alarm

This little $7 kit really shined during Sandy! Simply mount the alarm where you want to detect water level problems. When the water level rises it touches the contacts and the alarm goes off! Sensor can even be remotely located. Runs on a 9V battery (not included).

MK108 Water Sensor Alarm Kit $6.95

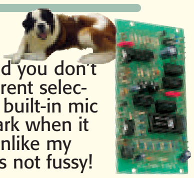
Tickle-Stick Shocker



The kit has a pulsing 80 volt tickle output and a mischievous blinking LED. And who can resist a blinking light and an unlabeled switch! Great fun for your desk, build a Panic Button... "Hey, I told you not to touch!" Runs on 3-6 VDC.

TS4 Tickle Stick Kit $9.95

Electronic Watch Dog



A barking dog on a PC board! And you don't have to feed it! Generates 2 different selectable barking dog sounds. Plus a built-in mic senses noise and can be set to bark when it hears it! Adjustable sensitivity! Unlike my Saint, eats 2-8VAC or 9-12VDC, it's not fussy!

K2655 Electronic Watch Dog Kit $39.95

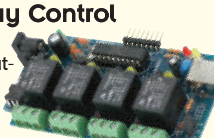
Passive Aircraft Monitor

PATENTED!

The hit of the decade! Our patented receiver hears the entire aircraft band without any tuning! Passive design has no LO, therefore can be used on board aircraft! Perfect for airshows, hears the active traffic as it happens! Available kit or factory assembled.

ABM1 Passive Aircraft Receiver Kit $89.95

4-Channel USB Relay Control

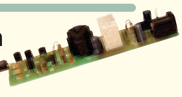


This professional quality USB relay controller allows computer controlled switching of external devices, plus full bi-directional communication with the external world using the USB port of your computer!

The controller features four onboard relay outputs with a current rating of 10A each. Also onboard is a 6-channel Input/Output interface, with each channel individually configurable as Digital Input, Digital Output, Analog Input (10-bit Resolution). In Digital Input/Output modes, each channel can support a TTL compatible or ST input or a 5V output signal. In Analog Input mode, each channel can convert a voltage of between 0 to 5V into a 10-bit digital representation.

UK1104 4-Ch USB Relay Interface Kit $59.95

Laser Trip Sensor Alarm

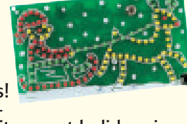


True laser protects over 500 yards! At last within the reach of the hobbyist, this neat kit uses a standard laser pointer (included) to provide both audible and visual alert of a broken path. 5A relay makes it simple to interface! Breakaway board to separate sections.

LTS1 Laser Trip Sensor Alarm Kit $29.95

The Holiday Spotlight!

LED Animated Santa

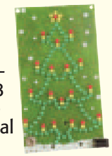


This animated Santa and reindeer display has been our most popular holiday display for years! It contains a whopping 126 dazzling colored LEDs which make it a great holiday sign that is guaranteed to draw attention!

LED animated motion makes it come alive. Runs on standard 9V battery or 9-12VDC external power supply. Dazzle your friends this great display!

MK116 LED Animated Santa Kit $21.95

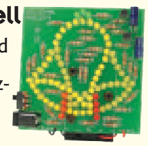
LED Christmas Tree



Electronic Christmas tree features 134 bright colored LEDs in the shape of a gorgeous holiday Christmas tree. Includes 18 random flashing blinking "candles" on the PC board! Runs on a 9V battery or external 9-12VDC power supply.

MK117 LED Christmas Tree Kit $21.95

LED Animated Holiday Bell



This PC board holiday bell is animated to simulate a bell swinging back and forth! 84 bright colored LEDs will dazzle you with holiday cheer! Includes an on/off switch. Runs on 9V.

MK122 LED Animated Bell Kit $16.95

3D LED Christmas Tree

Not your average LED display! 4 branch sections give this tree a true 3D look! 16 red LEDs light it up with yellow LEDs for you to customize your tree! The base of the tree is actually the 9V battery acting as a self supporting base! Now that's pretty neat!

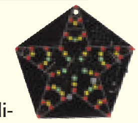
MK130 3D LED Christmas Tree Kit $7.95

SMT LED Christmas Tree

Build this subminiature Christmas tree and learn SMT at the same time. Small enough to wear as a badge or pendant! Extra SMT parts are included so you can't go wrong! Runs on Li-Ion cell.

MK142 SMT LED Christmas Tree Kit $10.95

Sound Activated LED Star



A built-in microphone picks up music and room audio and the LEDs respond just like a professional LED Vu meter! Adjustable sensitivity creates a great holiday display in sync with your music or audio! Runs on 9-12VDC.

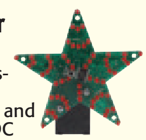
MK172 Sound Activated LED Star Kit $19.95

LED Flashing Holiday Star

A classic holiday star shaped PC board containes 35 brilliant red or yellow LEDs that can be selected to provide a steady or flashing display. The built-in 9VDC battery holder acts as the base for the star, making it easy to add to your decorations!

MK169 LED Flashing Holiday Star $12.95

LED Animated Effects Star



The ultimate LED star display features 60 brilliant LEDs that are micro-processor controlled to provide 24 different effects with sequencing! Includes red and yellow LEDs. Runs on a standard 9VDC battery or 9-12VDC external supply.

MK170 LED Animated Star Kit $29.95

Next Generation Classic Nixie Tube Clocks!

- ✔ Our brand-new next generation clock for 2013!
- ✔ Today's latest technology with yesterday's display!
- ✔ Choice of hand rubbed Teak/Maple, polished aluminum, or acrylic base!
- ✔ Choice of Nixie tubes types
- ✔ Low-cost GPS receiver timebase option!
- ✔ Programmable color LED tube mood lighting!
- ✔ The ultimate conversation piece!

NEW FOR 2013!

Our next generation of classic Nixie tube clocks perfectly mesh today's technology with the Nixie era technology of the 60's. Of course, features you'd expect with a typical clock are all supported with the Nixie clock... and a whole lot more! Time wise, the clocks are designed around a quartz crystal timebase, therefore are not AC power frequency dependent like a lot of clocks. This means they can be used in any country regardless of power frequency, with the included 12VDC regulated power supply. The clocks are also programmable for 12 or 24 hour mode, various AM/PM indications, programmable leading zero blanking, and include a programmable alarm with snooze.

Unlike most Nixie clocks, the clocks also display the date in DD.MM.YY, MM.DD.YY, or YY.MM.DD format, which can be programmed to display for a few seconds at the end of each minute either as a static display, or by a neat scrolling in/out from alternating sides of the display. Display wise, the clocks feature a programmable night mode with dim or blank display, a programmable master blank tube saver, hard or soft fade digit change, and even have a built-in "Slot Machine" cathode poisoning prevention routine. Programming and setting the clock is a breeze with simple 2-button entries on the rear panel. The clocks are available in our signature hand rubbed Teak & Maple, polished aluminum, or clear acrylic bases.

WE CROSSED THE TECHNOLOGY TIMELINE!

We then jumped the technological time line of the 60's Nixie displays by adding the latest multi-colored LEDs to the base of the Nixie tubes to provide hundreds of illumination colors to highlight the glass tubes! The LED lighting can be programmed to any color and brightness combination of the colors red, green, or blue to suite your mood or environment. Then we leaped over the technological time line by integrating an optional GPS time base reference for the ultimate in clock accuracy! The small optional GPS receiver module is factory assembled and tested, and plugs directly into the back of the clock to give your Nixie clock accuracy you could only dream of! The new series clocks are available in 6-tube and 4-tube versions, with your choice of bases, and your choice of kit or factory assembled & tested. If you're looking for the ultimate conversation piece, with a trip down nostalgia lane, check out our clocks at www.ramseykits.com/nixie.

NIXIE CLOCKS **Classic Nixie Tube Clocks, Teak/Maple, Polished Aluminum, or See-Through Acrylic Base, Kit or Factory Assembled** **From $229.95**

The Learning Center!

- ✔ Learn and build!
- ✔ 130, 200, 300, & 500 in one electronic labs!
- ✔ Practical through hole and SMT soldering labs!
- ✔ Integrated circuit AM/FM radio lab!
- ✔ Beginner's non-soldering & soldering kits!

130-In-One Learning Lab

Learn the basics and theory of electronics the fun way while building 130+ projects that you actually see working when you're done! The detailed instruction manual covers the hows and whys of every circuit, theory, parts identification, and description.

PL130A **130-In-One Learning Lab** **$39.95**

200-In-One Super Fun Lab

The front panel contains a built-in speaker, earphone, meter, 7-segment LED digital display, two controls and more to make the finished projects fun to use. Now you can build your own AM broadcast station, burglar alarm, telegraph, sound effects, radios, and more!

PL200 **200-In-One Learning Lab** **$84.95**

300-In-One Digital Lab

Take that next step and jump up to 300 separate projects that start walking you through the learning phase of digital electronics. This learning lab along with its comprehensive manual makes digital electronics learning fun!

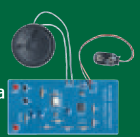
PL300 **300-In-One Learning Lab** **$109.95**

500-In-One Ultimate Lab

The ultimate electronics lab that includes 500 separate projects that cover it all, from the basics all the way to digital programming. Allows you to start at the beginning and progress at your own speed with 500 projects! Heavy duty case!

PL500 **500-In-One Ultimate Lab** **$249.95**

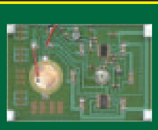
Practical Soldering Lab



Whether young or old, there's always a need to hone your soldering skills. Either learn from scratch or consider it a refresher, and end up with a neat little project when you're done! Covers through-hole design & soldering.

SP1A **Practical Soldering Lab** **$9.95**

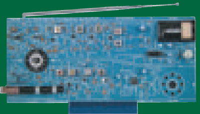
SMT Soldering Lab



Includes the PC board, all the SMT and other components required, and a comprehensive training course covering all aspects of SMT soldering, desoldering, removing, and component placement. The finished project is a neat little "Decision Maker".

SM200K **SMT Soldering Lab** **$22.95**

AM/FM Radio Lab



The unique design includes an IC, transistors, resistors, capacitors, coils, and misc. parts. A soldering guide is also provided. The radio kit includes a hi-tech blue PC board with the easy to follow schematic printed on the top surface.

AMFM108K **AM/FM Radio Lab** **$36.95**

RC Race Car Learning Kit

One of the neatest and most exciting learning kits for the kids! You will be building the Turbo King R/C speedster from the ground up, ending up with a super fast car! At the same time learning transmitters, receivers, switches, gears, and motors!

AK870 **R/C Car Learning Kit** **$29.95**

GET THE NUTS & VOLTS DISCOUNT!

Mention or enter the coupon code ***NVRMZ12*** *and receive 10% off your order!*

800-446-2295

www.ramseykits.com

Prices, availability, and specifications are subject to change. We are not responsible for typos, stupids, printer's bleed, or watching too much football on TV! Robin is STILL waiting for a new sailboat from Santa! Visit www.ramseykits.com for the latest pricing, specials, terms and conditions. Copyright 2012 Ramsey Electronics®... so there!

RAMSEY ELECTRONICS®
590 Fishers Station Drive
Victor, NY 14564
(800) 446-2295
(585) 924-4560


Detention Punishment:
Write "I will not play video games in class"
300 Times
10 X = 0
20 Print "I will not play video games in class"
30 X = X + 1
40 IF X < 300 GOTO 20
50 END
President Bush