PROJECTS ■ THEORY ■ APPLICATIONS ■ CIRCUITS ■ TECHNOLOGY

NUTS AND VOLTS

www.nutsvolts.com
August 2012

EVERYTHING FOR ELECTRONICS

Machine To Machine Communications

The Internet of Things

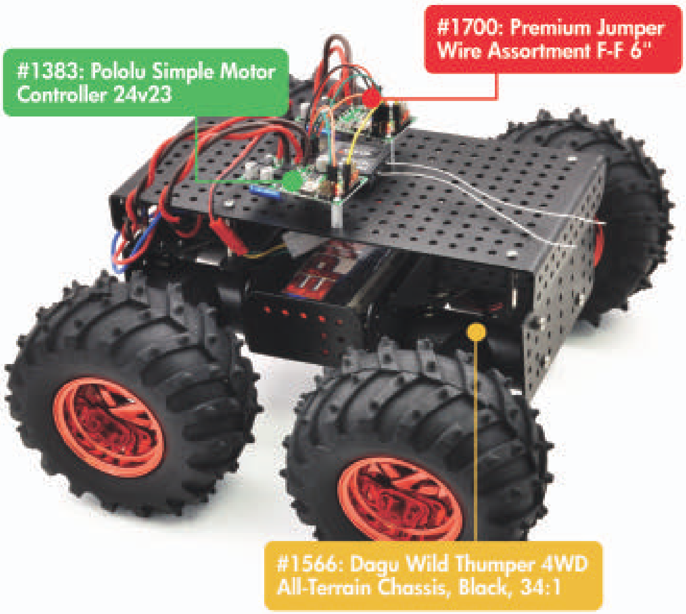
- PICAXE
- Arduino
- Android

Build An

- **Alarm Autodialer**
  Now your house can call you to report a problem.
- **Voltage Mirror**
  Mirror a device's input voltage, but with the polarity swapped.
- **Mystery Solar Pendulum**
  Amaze your friends with this fun, BEAM based gadget.

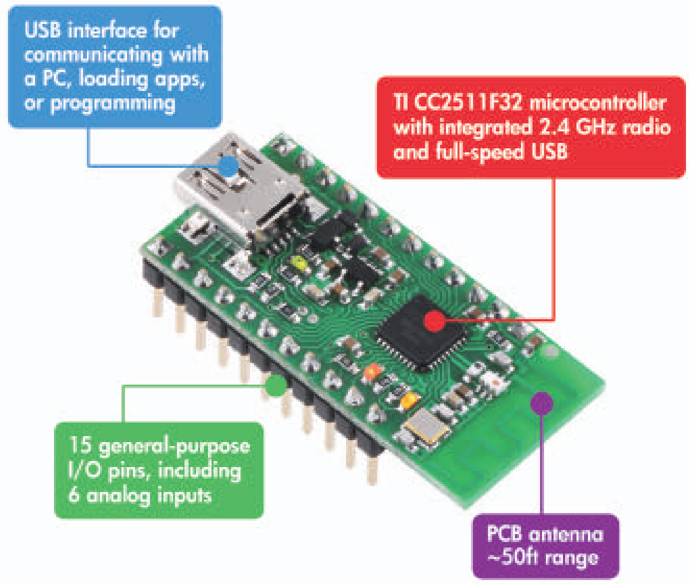
DIY Projects:

Wild Thumper-Based Robot



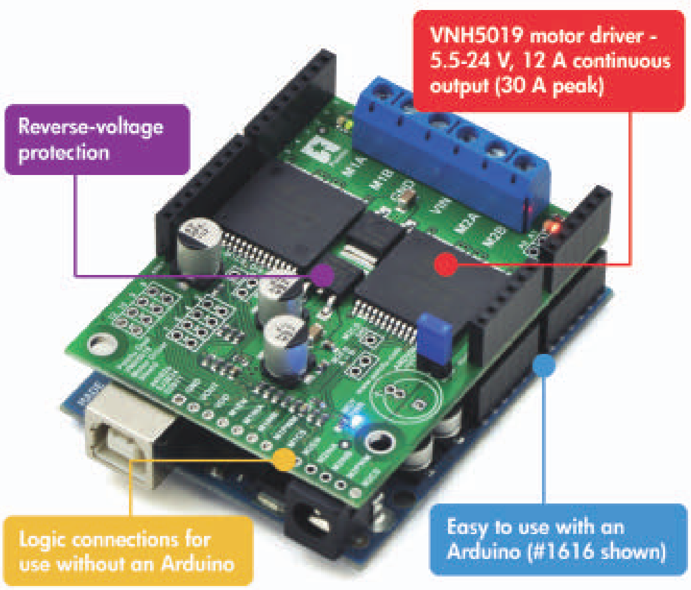
Programmable Controllers:

Item #1336: Wixel Programmable USB Wireless Module



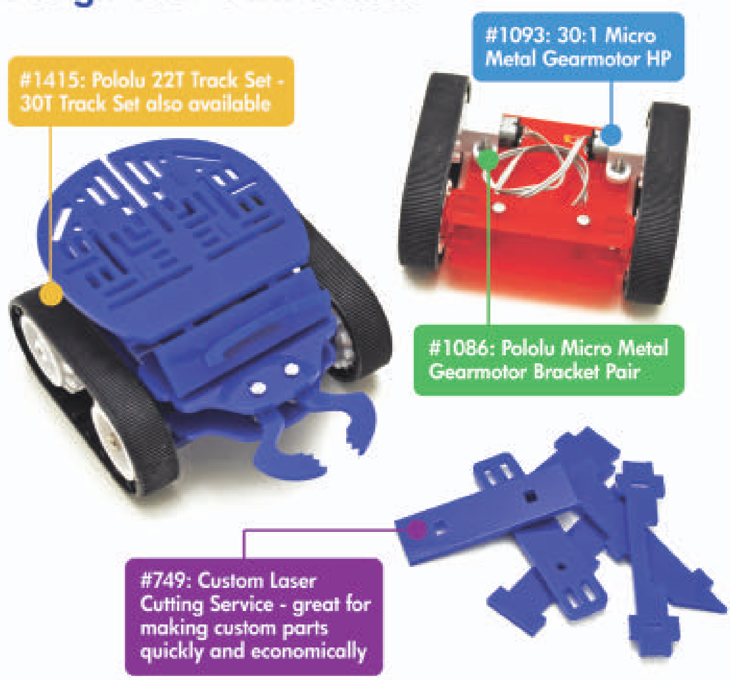
Motor Drivers:

Item #2502: Dual VNH5019 Motor Driver Shield for Arduino



Custom Laser Cutting:

Design Your Own Chassis



Finding the right parts for your robot can be difficult, but you also don't want to spend all your time reinventing the wheel (or motor controller). That's where we come in: Pololu has the unique products - from actuators to wireless modules - that can help you take your robot from idea to reality.

Pololu
Engage Your Brain

Find these products and more at **www.pololu.com**

August 2012

www.nutsvolts.com

Projects & Features

Page 32

Page 10

Columns

Departments

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

How Much to Invest?

I have a general rule when it comes to tools, electronics, and accessories: I buy the best that I can afford at the time. There's nothing more frustrating than having to buy another widget because I didn't spend the extra dollar to get a feature that I didn't know I needed. This defensive purchasing sounds simple, but in practice, it's a difficult rule to follow.

Let me give an example. I'm preparing a series of articles for our sister publication, *SERVO*, on autonomous aerial vehicles. Thanks to open source hardware autopilots and ground control software, it's possible to build a fully autonomous quadcopter (a helicopter with four motors) capable of carrying a 2 lb payload for about $1,000.

The ground control software — often referred to as waypoint software — is linked to GoogleEarth, enabling just about anyone to plan a graphical route for the UAV and have the payload delivered there autonomously. That is, you literally direct a drone to travel two blocks down Main Street, take a left at the corner, and then land at the doorstep of the fifth house on the right. Some think this is a great scheme for pizza delivery. Others think cruise missile.

Regardless of what the technology and capabilities bring to mind, it all depends on reliable battery power. So, as part of building and testing various quadcopter systems, I had to identify and purchase the 'best' batteries and charger that I could find and stay within my budget.

Simply going for the most expensive chargers and batteries and ignoring my budget wasn't an option — even if I had unlimited funds — because a huge, high voltage, high capacity lithium battery wasn't compatible with the airframes I was testing. No, I had to do my homework.

It turns out that lithium batteries generally offer the best energy density, weight, and cost combination, compared with the other common battery chemistries on the market. However, when you get into it, you'll discover that there are various formulations of lithium batteries — including LiPo, LiLo, and LiFe. I won't go into details here (as you can tell, I'm trying to entice you to take a look at *SERVO*), but each formulation has different costs and capabilities.

Then, there's the physical packaging. Some lithium batteries are — because of safety regulations — encased in hard, heavy plastic. I discovered this type of battery on the Parrot AR Drone 2.0 (the 'low end' vehicle in my study) and immediately ordered a replacement battery with the same weight and dimensions, but that provided double the capacity. The new battery is encased in shrinkwrap.

The tradeoff is that lithium cells are known to catch fire and explode when damaged in a crash — a tradeoff that I'm willing to make. There are lightweight, fireproof battery wraps available, if you want to lower the risk of fire somewhat.

The problem with the battery upgrade for the Parrot AR Drone was that the original charger couldn't handle the load, so I was forced to buy a new charger. The 'best' charger (in this case) had to handle not only the relatively minor demands of the AR Drone battery, but those of the batteries capable of powering the meatier motors and electronics of drones from DIYDrones (**www.diydrones.com**) and Parallax (**www.parallax.com**).

What's more, because it's illegal to fly a potentially lethal drone over a crowded city like Boston, I have to travel an hour outside of the city to test the drones. That means a supply of batteries. It also means that the 'best' charger for me had to include the ability to charge/discharge several battery packs simultaneously. So, I was forced to consider $100+, 1,000W battery chargers and multiple, $40+ LiPo batteries.

Then, there's the 'best' power supply to power the charger — which has to be powerful enough to meet the demands of charging a half dozen batteries simultaneously.

Repeat the above exercise for the R/C transmitter and receiver (check out the Spektrum DX7) and onboard camera system (see the GoPro HD Hero2), and we're talking a lot of research and a significant investment. Not the most expensive or capable components I could find, but I expect the investment in batteries, charger, camera, transmitter, and receiver will outlive the review.

For example, had I gone with a simple upgrade charger and battery pack for the Parrot drone, I would have had to buy yet another charger for the higher capacity batteries on the other drones.

I hope this helps you with your purchase planning. Do your best to look ahead to your next few system upgrades — whether it's test equipment or a microcontroller development kit — to determine the best investment for you. **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, Louis Frenzel, John Blankenship, Samuel Mishal, Chris Watson, Jim Stewart, Dan Gravatt, David Williams

CIRCULATION DEPARTMENT
subscribe@nutsvolts.com

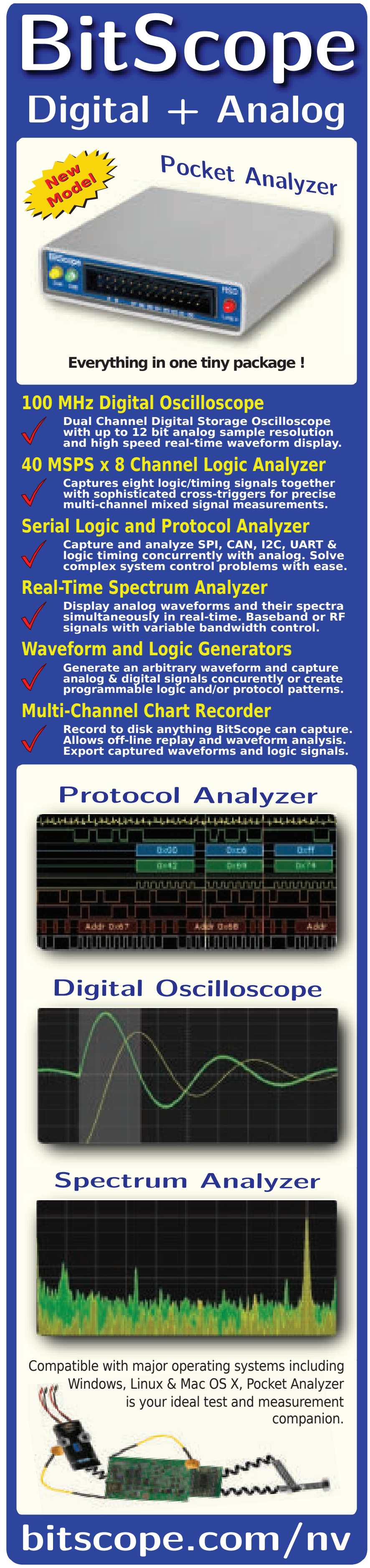
SHOW COORDINATOR
Audrey Lemieux

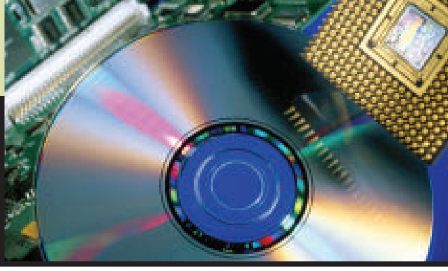
MARKETING COORDINATOR WEBSTORE
Brian Kirkpatrick
sales@nutsvolts.com

WEB CONTENT
Michael Kaudze
website@nutsvolts.com

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

PRODUCTION/GRAPHICS
Shannon Christensen
Sean Lemieux





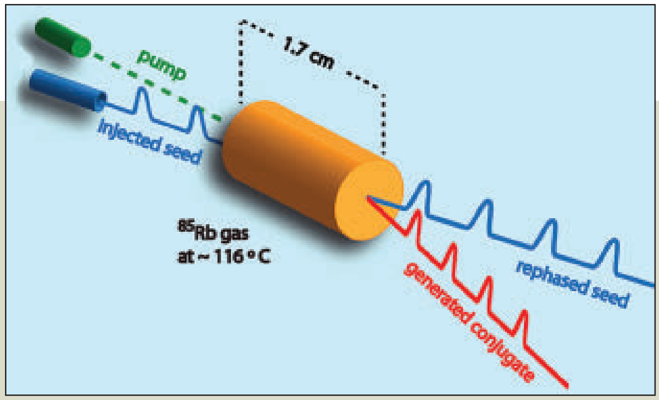
TECH KNOWLEDGEY 2012

EVENTS, ADVANCES, AND NEWS

■ BY JEFF ECKERT

ADVANCED TECHNOLOGY

LIGHT TRAVELS FASTER THAN LIGHT



■ Illustration of four-wave mixing to generate a superluminal pulse.

According to Einstein's special theory, nothing can move faster than light through a vacuum. However, it appears that some researchers at the National Institute of Standards and Technology (NIST, **www.nist.gov**) have figured out a way to beat the system (sort of) and generate a stream of "superluminal" pulses. The technique is called "four-wave mixing," and it involves sending 200 ns "seed" pulses of laser light into a heated cell that's filled with rubidium vapor. A separate "pump" beam — at a different frequency — is also injected. The seed pulse is amplified by the vapor which has the effect of shifting its peak forward. Thus, even though the pulse itself is still subject to Einstein's law, the peak arrives at its destination 50 ns sooner than it otherwise would. Picture an ocean wave traveling along at 10 mph. If you can figure out a way to tilt the crest forward, it will crash on the beach a little sooner, even though the water isn't really moving any faster.

Both injected beams react with the vapor to generate a second output pulse called the "conjugate" to indicate its mathematical relationship to the seed. Depending on how the laser is tuned and other conditions, this pulse can travel either faster or slower. The concept is said to be potentially useful in such abstract applications as the measurement of quantum discord, which isn't going to generate much excitement in the general public. It also could be used to improve the timing of communication signals, and that has widespread implications. ▲

■ An early RCA 808 vacuum tube.

NEXT HIGH TECH BREAKTHROUGH: THE VACUUM TUBE?

Vacuum tubes are generally thought of as technological dinosaurs, having been replaced by cheaper, lighter, more efficient semiconductors. Tube amps, however, are still prized by musicians and some audiophiles, and the technology still rules in things like high power transmitters and microwave ovens. In addition, tubes can withstand high radiation environments and — because electrons move about 600 times faster through a vacuum than silicon — they offer some electrical advantages. Proving that "everything old is new again," a team of researchers from NASA's Ames Research Center (**www.nasa.gov/centers/ames**) and the Korean National Nanofab Center have created a prototype that combines both technologies. As documented in a recent issue of the American Institute of Physics' journal, *Applied Physics Letters*, the "vacuum channel transistor" shrinks the concept to a device that is only 150 nm in length. The compact size allows it to operate at only 10V which researchers believe can be reduced to 1V with some further development, thus making it competitive with standard semiconductors. Plus, it can be fabricated using conventional semiconductor methods. According to the authors, potential applications include medical diagnostics, high speed telecom, and electromagnetically "dirty" environments. And maybe even a pocket-size Marshall stack. ▲

Discuss this article in the *Nuts & Volts* forums at http://forum.nutsvolts.com.

COMPUTERS AND NETWORKING

NEW PC FOR BOTTOM-FEEDERS

■ The AllWinner A10 Android PC.

It was in 1982 that I became the proud owner of my first computer, a Timex Sinclair 1000: a little Z80-based machine with 2 kB of memory and a processing speed of 3.25 MHz. Two days after shelling out $29.95 for the unit, it croaked, as did many of the others produced by the joint venture between Timex Corp. and Sinclair Research. Still, it was fascinating that — at least in theory — it was possible to build something for well under $100 that could perform many of the same functions as the IBM 4341 mainframe down the hall, which cost about $275,000 (but was loaded with a whopping 4 MB of main memory). Thirty years later, the almost-free concept has reappeared in the form of the AllWinner A10 Android Mini PC. Details about the Chinese-made machine are sketchy, but by picking data from various websites it seems to be powered by a 1.5 GHz CPU plus a Mali-400 GPU (as used in the Samsung Galaxy SII smartphone). It comes with 512 MB of DDR3 RAM and 4 GB of nand flash, and it offers Wi-Fi 802.11b/g connectivity. Video output is 1080 p HDMI, and it all runs on the Android 4.0 operating system. Included in the package is an AC adapter (EU plug), HDMI cable, mini USB to USB OTG adapter, and a user manual (both in Chinese and English). It's hard to say what you might want to use it for, but for only about $75 at various online stores, maybe you'll figure something out. ▲

■ Dell's Latitude E6430s offer extended battery life.

UP TO 32.7 HOURS BETWEEN CHARGES

If the laptop marketplace seems to be a confusing mass of machines that are essentially the same, that's because it is and they are. If your prime directive is to compute over long periods of time without a battery recharge, though, some new business models from Dell might be a good bet. The new Latitude E-series machines — which range in screen size from 12.5 to 15.6 inches — sport Intel's new Ivy Bridge processors which are more efficient in general, and can save power by shutting down idle cores. This contributes to the machine's ability to provide up to 32.7 hours of battery life (in average use), but the main contributor is a nine-cell internal battery plus external packs. In addition, users can remove the optical drive and replace it with a battery. Additional features include the ExpressCharge feature that allows for an 80% recharge in as little as one hour, USB 3.0, optional LTE mobile broadband and Bluetooth 4.0, and a ruggedized version for extreme environments. Price information had not been released as of this writing. ▲

GET YOUR HEAD IN THE CLOUD

You have no doubt heard a lot about "cloud computing" lately, but may not have a clear idea of what it means. The short definition provided by NIST is "a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources — for example, networks, servers, storage, applications, and services — that can be rapidly provisioned and released with minimal management effort or service provider interaction."

"Okay," you might say. "But what does that have to do with me?" Well, NIST has put together a publication — *Cloud Computing Synopsis and Recommendations* — that "explains cloud systems in plain language and provides recommendations for information technology decision makers, including chief information officers, information systems developers, system and network administrators, information system security officers, and systems owners." It's not immediately evident that the language is all that plain, and at 81 pages it isn't exactly concise. But it is free. If you're up to the challenge, you can download a PDF version at **www.jkeckert.com/freedownloads/cloud.pdf**. ▲

91% EFFICIENT POWER SUPPLY

A new introduction from Murata Power Solutions (**www.murata-ps.com**) is the MVAB120 series of open-frame, single output, 120W power supplies. Useful for a range of telecom and industrial applications, the supplies are up to 91% efficient and offer fan-free operation using convection cooling. The devices come in three output models — 12, 24, and 48 VDC — and they accommodate standard input ranges from 90 to 264 VAC and 120 to 300 VDC. Measuring just 2 x 4 x 1.35 inches (101.6 x 50.8 x 34.3 mm), they satisfy the 1U form factor specs. The MVAB supplies can deliver a full load in the range of -10°C to 50°C and include overcurrent, overvoltage, and overtemperature protection. They'll run you up to about $50 to $60 each, depending on quantity. ▲

■ Fanless power supply is up to 91% efficient.

AUTOMOTIVE MYSTERY SOLVED

One of the ancient mysteries of the automotive world is when to change your oil. Your dad probably told you to do it every 3,000-5,000 miles. Newer cars flash a message on the dash when you reach the prescribed milestone. But, do you truly believe that the useful life of motor oil depends entirely on how many miles you have driven? Doesn't start-and-stop driving put more of a strain on it than highway cruising? Doesn't temperature make a difference? Aren't some vehicles harder on oil than others? Don't some oils last longer than others? Well, you won't need to ponder these questions much longer, because by the time you read this, Lubricheck should be available. All you have to do is feed four or five drops of oil from the dipstick into the credit-card-size device, press a button, and it will instantly rate your oil on a scale of 1 to 10. Anything up to 7 places it in the "OK" zone; 8 or 9 indicates the "change soon" zone; and 10 means "change now."

Lubricheck uses a combination of capacitive and resistive sensing to make the test, and it will even display a flashing red LED if the sample is contaminated with antifreeze, water, or metal. (This makes it useful when shopping for a used car, as it might allow you to detect things like a leaky head gasket.) And, yes, there's an app for that, so you can record and track oil data on multiple vehicles and email customized reminders to users. Lubricheck is priced at $39.95 and can be found at **www.lubricheck.com**. ▲

■ Lubricheck, a "blood tester" for your vehicle.

AVOID THAT DUI

There was a time when you could go bar hopping, toss down a dozen or so shots of Jack, chase them with beer, and drive on home when you hit closing time. Unless you were weaving all over the road or speeding, the cops weren't likely to be concerned about you. However, those days are long gone. Today, the authorities keep their eyes peeled for any signs of impairment and will show you no mercy.

So, let's say you've been hanging out at the local pub for a couple hours, throwing darts with your buddies, and you've had a beer every 20 minutes. Let's also specify that you're a 180 lb male. You feel just fine; not a bit impaired. According to boozing charts, though, you now have a blood alcohol content (BAC) of 0.102 percent which is well over the 0.08 percent that qualifies you for a DUI in all 50 states. If you get caught driving, the result can be fines, legal fees, insurance rate increases, and treatment fees, which can cost $10,000 or more. You will also face license suspension, possible job loss, and some really dirty looks from the wife. So, if you do tipple a brew now and then, it might be a good idea to invest $29 in a gadget like the BACtrack Keychain Breathalyzer, available all over the Web. There's not much to say about it except that it provides results ranging from 0.00 to 0.40 percent BAC, provides a readout in five seconds, and comes with three mouthpieces so you can test a couple friends, too. If you put your car keys on it, you won't forget to use it. ▲

■ The BACtrack keychain breathalyzer, a $29 DUI solution.

INDUSTRY AND THE PROFESSION

MAC'S BOTTOM LINE AT ALL-TIME LOW

In January1997, Apple Computer, Inc., changed its name to simply Apple, Inc., to reflect its growing move into a wider field of electronics. At the time, Macs still drove 47 percent of the company's revenue, for a total of $2.5 billion. The change has proven to be highly appropriate, as figures from the first fiscal quarter of this year (which actually ended Dec. 31, 2011) show that Mac sales accounted for an all-time low of 13 percent of overall sales, according to Computerworld. That doesn't mean that sales are down; in fact, Apple posted first quarter sales of 5.2 million Macs — a 26 percent increase over 2011. That doesn't match the 111 percent increase in iPads (to 15.43 million), though. The only sour side was iPod sales, down 21 percent to 15.4 million. All told, the company came in with $46.33 billion in quarterly revenues.

Back in 1997, when Apple had lost a billion dollars and was looking shaky, Steve Jobs famously observed, "Apple doesn't have to lose for Microsoft to win. Microsoft doesn't have to lose for Apple to win." Turns out to have been pretty accurate. **NV**

Discuss this article in the *Nuts & Volts* forums at http://forum.nutsvolts.com.

■ BY RON HACKETT

FURTHER EXPLORATIONS WITH PICAXE TEMPERATURE MEASUREMENT

In this month's Primer, we're going to continue our explorations of temperature measurement with PICAXE processors, but before we get to that, I want to give you an update on my "Freezer Alarm" project. By the time I had finished writing the previous column, my freezer alarm had been functioning perfectly for a couple of weeks, and it continued to do so for another month or so. However, one morning I heard the alarm's repetitive beeping emanating from the basement. Expecting the worst, I went downstairs to see what had happened. When I opened the freezer door, the spring-based thermometer that I keep inside the freezer to check on the accuracy of the 20M2-based thermometer indicated that the freezer temperature was well below my alarm set point; in fact, it was slightly below 0º F.

It took me a while to figure out what had happened. Last time, I mentioned that while I was developing the freezer alarm project, I observed that its temperature varied between +5º F and +20º F. What I didn't mention was that this observation led me to assume that I didn't need to deal with the possibility that the freezer temperature might occasionally drop below 0º F, so the program didn't respond correctly to a sub-zero (Fahrenheit) temperature. The moral of the story is that taking the easy way out isn't always the best approach. You might think that I would have learned that by now, but, hey, no one's perfect!

If you look back at the *LED2x7-FreezerAlarm.bas* program from last time, you will see the following code snippet near the end of the C2F subroutine:

```
if negFlag = 0 then
'if above freezing
  temp = 32 + temp
'then add temp to 32
else
'else (below freezing)
  temp = 32 - temp
'subtract temp from 32
endif
```

The next to the last line (temp = 32 - temp) is the one that got me into trouble. As you may remember, we defined *temp* as a word variable because (earlier in the C2F subroutine) we needed to multiply it by nine as we were converting it from Celsius to Fahrenheit. In order to see what went wrong, suppose *temp* = 31 and *negFlag* = 1 (i.e., the current temperature is below 0º C) just before we execute the previous code snippet. Therefore, the *else* clause is executed, and *temp* turns out to equal 32 - 31, or 1º F. So far so good, but what happens if the magnitude of *temp* is a little larger, e.g., 33 (which means the temperature is slightly more negative when *negFlag* = 1)? Well, obviously, *temp* now equals 32 - 33, but what is that value in PICAXE-land? (Hint: Don't forget, *temp* is a word variable, and PICAXE BASIC can't handle negative numbers.)

If your answer is 65,535, you win the prize! When 33 is subtracted from 32, a PICAXE word variable wraps around to 65,535 which is certainly greater than my alarm set point, and explains why the alarm sounded when there really wasn't a problem (except for my faulty software).

So, how do we fix the program so that the alarm doesn't sound unnecessarily? For the time being, I'm going to take the easy way out, but I promise I'll correct that before we're done with our temperature-measurement projects. (Really, I do!) Since it doesn't matter to me if the freezer temperature is a couple of degrees below 0º F, I'll just assign a value of 0 to *temp* whenever a "wrap-around" occurs. Adding the bolded *if/then* statement shown below accomplishes that, and will keep the alarm from sounding whenever the freezer temperature drops below 0º F:

```
if negFlag = 0 then
'if above freezing
  temp = 32 + temp
'add temp to 32
else
'else (below freezing)
  temp = 32 - temp
'subtract temp from 32
  if temp > 65000 then
'if it wraps-around
     temp = 0
'let's call it zero
  endif
endif
```

BACK TO SQUARE ONE

Once I had resolved my little freezer alarm problem, I was eager to begin development on another one of my projects: a temperature sensor for my cold smoker. If you're not familiar with cold smoking, a Google search will provide lots of relevant information. Essentially, it involves smoking various food items at temperatures well below 100º F, so the DS18B20 is more than adequate for the task, and there's no risk of exceeding its maximum temperature of +257º F. (Another assumption?)

However, my only LED-2x7 board was busy monitoring the freezer, and I wanted to keep it there. At first I thought I would just build another LED-2x7 board, but then I realized that I have a couple of other projects that would also benefit from a two-digit LED display. If you've already built an LED-2x7 board, you know that it takes a fair amount of time to complete and I was in a hurry to get started on the next project, so I decided to see if I could simplify the circuitry a bit by eliminating the 20M2 processor, and redesigning the board so that it could be directly inserted into any 20M2 breadboard circuit. That way — in the immediate future — I only need one additional stripboard circuit which I can move from project to project as necessary. Then, when I complete a project I can construct a custom stripboard for it with whatever additional circuitry may be required (e.g., a power supply).

I won't take up a lot of space discussing the new version of the board (LED-2x7v2) because I realize that not everyone will want to build another display board. However, I do want to include the basics, so that you can decide whether it's a project you want to tackle. The schematic for the re-designed LED-2x7v2 board is shown in **Figure 1**. As you can see, it's much simpler than the original design. It also only requires one stripboard, rather than the two that were needed for our original project.

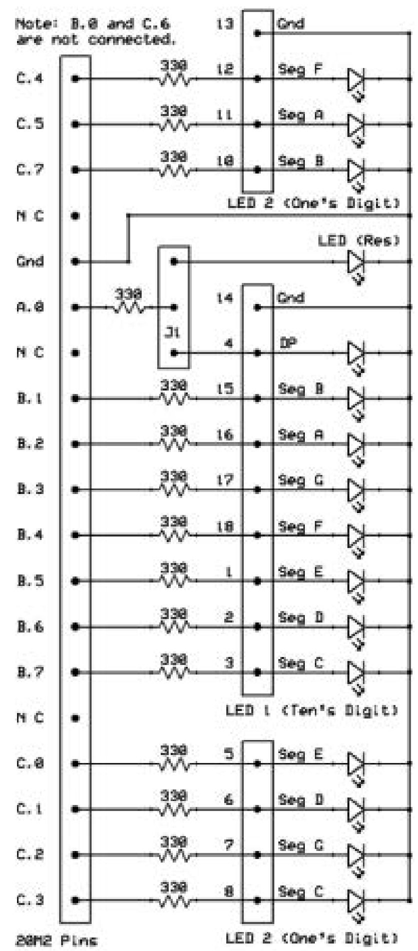


FIGURE 1. LED-2x7v2 schematic.

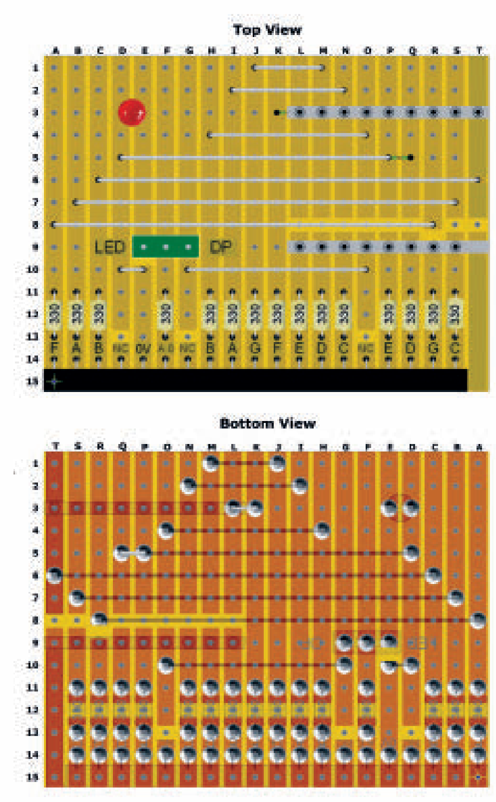
In retrospect, I wish I had thought of this approach first because — as it turns out — the necessary software is also simpler, so it would have been a better place to start our explorations of a two-digit LED display. I guess I just needed to climb a couple of trees before I could see the forest.

In the **Figure 1** schematic, the 19-pin connector along the left edge is a right-angle male header that enables the stripboard to be inserted into a breadboard. The 10 pins between "B.7" and "Gnd" (inclusive), exactly match the layout of pins 11 through 20 on the 20M2 processor. Therefore, the stripboard can be inserted into a breadboard directly adjacent to pins 11 through 20 on the 20M2. Also, note that there is no connection to pins B.0 and C.6, so those two pins are available for use in the 20M2's breadboard circuitry.

Besides the fact that there's no processor in this version of our project, the most important change is that the port B pins are all connected to LED 1 (the ten's digit), and the port C pins are all connected to LED 2 (the one's digit). As a result, each LED digit pattern can be stored in one eight-digit value, rather than the 16 bits that the original LED-2x7 project required. As we'll soon see, that's what simplifies the software.

Figure 2 presents the stripboard layout for the LED-2x7v2 board; a larger version is available for downloading at the article link. (The pins of the 19-pin male header aren't visible in the layout because I couldn't find a way to get LochMaster — the software I use to design stripboards — to let me draw anything beyond the outline of the board.) It's an easy board to construct, so I won't go into the details. However, I will mention a couple of points.

First, the board contains 20

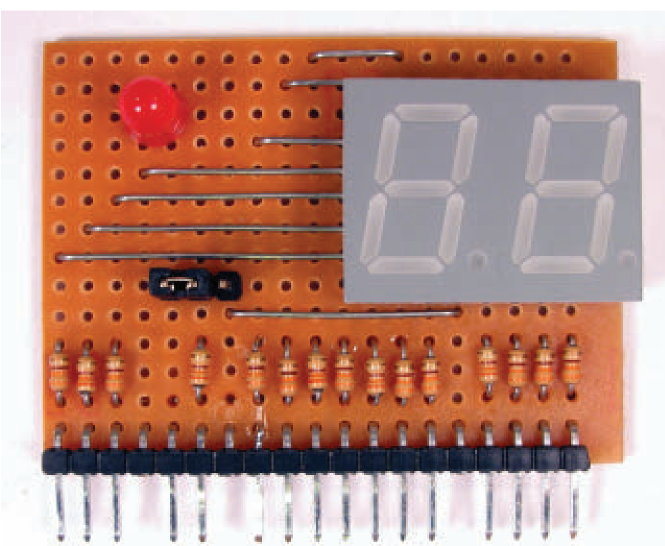


■ FIGURE 2. LED-2x7v2 stripboard layout.

traces; if you want to use a small stripboard (as I did), you will need to drill two holes in the extra solid trace on the right side of the board, at locations T3 and T6. Also (as I mentioned earlier), we're not using the decimal point for LED 2 (the one's digit), so in order to avoid drilling a third hole at location T9, I just snipped pin 9 from the display and soldered the display directly to the stripboard.

Of course, you can also use two female headers to mount the LED display. If you choose that option, just remove the corresponding pin from one of the header's before you solder it in place. A two-pin shunt can be used in conjunction with the three-pin male header at locations E9-G9 to select whether you want the 20M2's A.0 pin to be connected to the decimal point of LED 1 (ten's digit) or to the discrete resistorized LED on the board. You can also omit the shunt entirely if you prefer to use pin A.0 for some purpose on your breadboard circuit. Finally, note that I forgot to remove the holes from the bottom view of trace T; just pretend they aren't there!

The board is easy enough to assemble without detailed instructions. However, if you decide to build this version of our project, you may want to refer to the photo of my completed board that's shown in **Figure 3** for further clarification. Also (as you can see in the photo), I've snipped off the three pins that we are not using. Since those pins aren't connected to anything this isn't really necessary, but it does make it easier to correctly align the stripboard with the pins of the 20M2 on the breadboard. **Figure 4** is a photo of the breadboard circuit that I used to test the completed stripboard display. I'm including it so that you can see how I connected my Prog-03 programming adapter to the 20M2. You can also see that I have connected a resistorized LED to pin B.0 on the 20M2. (We'll soon see why.)



■ FIGURE 3. Completed LED-2x7v2 stripboard.

Figure 5 is a photo of the same breadboard after I inserted the LED-2x7v2 board. In order to test the board, I re-wrote the *LED-2x7count.bas* program that we used back in April. The updated program (*LED2x7v2-CountBad.bas*) is available at the article link.

Now would be a good time to download it, so that we can move on to discussing what's "bad" about the program. Even if you decide not to construct the LED-2x7v2 stripboard circuit, you may want to continue reading because the problem we're about to discuss illustrates an important aspect of the use of the *outpinsB* and/or *outpinsC* special function variables that are available for use with the 14M2, 18M2, and 20M2 processors.

WRITING SOFTWARE FOR THE LED 2x7v2 STRIPBOARD

For reference during the following discussion, you may want to print out a copy of the *LED-2x7v2-CountBad.bas* program. If you don't already have one handy, you may also want to print out a copy of April's *LED-2x7count.bas* program for comparison. The first thing you will probably notice is that the v2 version requires fewer variables and it has a much simpler data structure. As I've already mentioned, these simplifications are the result of the simplified I/O connections on the LED-2x7v2 board; this time, the port B pins only connect to LED 1, and the port C pins only connect to LED 2.

In the v2 program, I derived the 20 data values in the same way that we discussed for the 40 data values that were needed for version 1 of the software, so I won't repeat that process this time. However, as we saw earlier in the schematic of **Figure I**, we're not using pins C.6 or B.0 this time. As a result, note that the data values for bit 6 of port C and bit 0 of port B have all been set to 0. Since those two bits will not be used, it doesn't matter how we define them; it's just easier to be consistent. However (as we'll soon see), while the definitions for bit 6 for the port C values are not a problem, the definitions for bit 0 for the port B values will, in fact, create a problem.

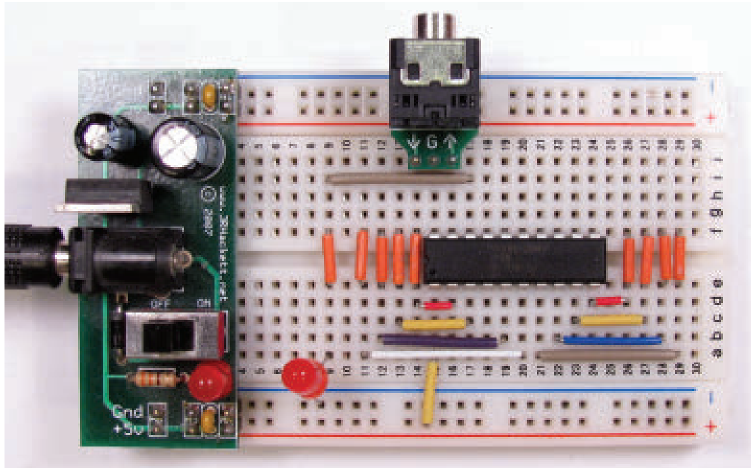
The main *do/loop* is functionally identical to the original version, except for the fact that I have added a *toggle B.0* statement which I included to clearly demonstrate the problem we're going to have. Finally, the display subroutine has also been simplified by our simpler data structure.

So, what's the problem? Let's assume that a shunt has been inserted onto jumper J1 on the stripboard so that pin A.0 is connected to the discrete LED on the stripboard. In the main *do/loop*, the two discrete LEDs (the one on the stripboard that's connected to the Serial Out pin, and the other on the breadboard that's connected to pin B.0) are both toggled each time through the counting loop, so it's reasonable to assume that both LEDs will behave identically (ha!). Also, a reasonable (but incorrect) guess would be that they both will be lit during one count and then off during the next count, repetitively.

If you intend to construct the LED-2x7v2 board, you may want to do so before reading further. That way, you can download *LED-2x7v2-CountBad.bas* to your breadboard circuit, observe the two discrete LEDs, and have the fun of trying to figure out what's happening!

WHAT'S GOING ON HERE?

In case you don't intend to build the LED-2x7v2 board, I'll describe what happens when the program is run, so you too can have the fun of figuring out what's going on. The discrete LED on the stripboard that's connected to the Serial Out pin (a.k.a., A.0) behaves as expected; it's lit during one count and then off during the next count, repetitively.



■ FIGURE 4. 20M2 breadboard circuit.

However, the discrete LED on the breadboard (connected to pin B.0) blinks very briefly at the beginning of **every** count, and then turns off throughout the remaining time of each count. In order to understand why the breadboard LED is behaving unexpectedly, let's examine the program more closely, from the instant it starts running. (You may want to refer to your printout for the following discussion.)

The very first task the program carries out is to define all the I/O pins as outputs, except the C.6 pin which is fixed as an input. For all PICAXE processors, whenever an I/O pin is defined as an output, the pin is automatically set to a low level so that any connected mechanical devices aren't powered until we issue an instruction to do so. Therefore, before we enter the main *do/loop,* both discrete LEDs are turned off. The first time through the *do/loop,* the *number* variable is set to 0, then the two *toggle* instructions light the two discrete LEDs, and we jump to the *displayNum* subroutine (which is where we get into trouble).

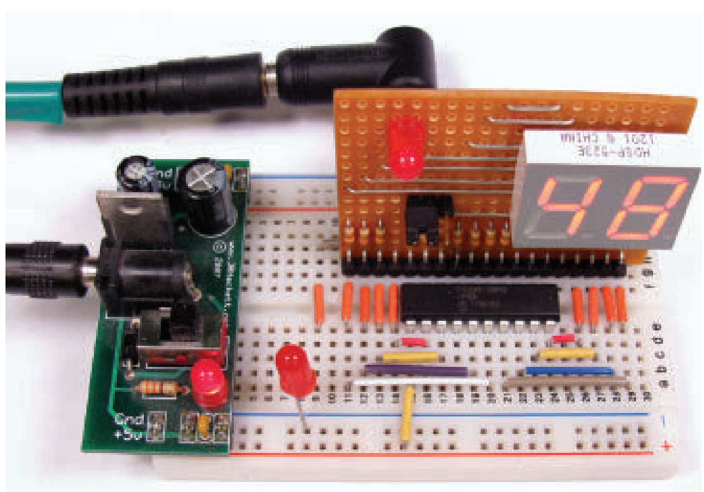
Essentially, this subroutine is a simplified version of the one we used back in April's Primer because, this time, we have also simplified our I/0 connections. (On the LED2x7v2 board, we only need one byte of data — rather than two — to define the segment pattern for each digit.) Consequently, I'm not going to discuss each program line in detail; I'll just focus on the problematic portion of the code.

We start to get into trouble at this point in the *displayNum* subroutine:

```
valTen = number / 10
valTen = valTen + 20
read valTen, segsTen
```

In the first line above, we divide the current count (which is 0 because we're still in the first iteration of the main *do/loop*) by

■ FIGURE 5. LED-2x7v2 inserted in breadboard.



10 to get the ten's digit (which, of course, is 0). Therefore, in the second line, *valTen* is assigned a value of 20 which is used in the third line to read the appropriate segment data for a ten's digit of "0," which we defined earlier at data location 20. If you look back at the data definitions, you will see that we have now assigned the binary value %11110110 to the *segsTen* variable.

The third program line above is where the problem actually starts, but it doesn't become visible until we execute the last instruction in the subroutine:

```
outpinsB = segsTen.
```

At this point, the digital value %11110110 is assigned to *outpinsB*; it's important to remember that each "1" results in the corresponding pin being set to a high level, and each "0" results in the corresponding pin being set to a low level. Look again at the program's data 20 instruction, and you will see why we're now in trouble.

The first seven digits of the binary data byte correspond to the LED segments listed directly above the data byte. Within those seven digits, only the "segment G" position contains a zero. As a result, when we assign the *segsTen* data to *outpinsB* later in the subroutine, every segment of the ten's digit LED — except segment G — will be lit. That's exactly what we want, because the standard seven-segment arrangement places segment G in the middle of the display, and that's the only segment that we don't want to light when we're displaying a zero.

All the above is precisely what we intended to do, but now let's turn our attention to the final "0" that's in the *bit0* position of data byte 20. Earlier, I mentioned that it doesn't matter whether we put a 0 or a 1 in this position because pin B.0 is not connected to the LED display anyway. However, I also warned that *bit0* would be problematic, and that's what's happening here. When the data value %11110110 is assigned to *outpinsB*, the 0 in the *bit0* position gets assigned to pin B.0.

In other words, the discrete LED that's attached to pin B.0 (which we toggled to a high state a few milliseconds ago) is immediately switched to a low state, so it turns off very shortly after it had turned on. The next time through the loop, the same thing happens; the *toggle* instruction turns the LED on again (because it was off), but the final statement in the *displayNum* subroutine almost immediately turns the LED off. Therefore, the LED is not lit most of the time; it only blinks briefly at the beginning of each new count.

At this point, I'm sure some readers are wondering why we don't just put a 1 in the *bit0* position of all the data bytes. To answer that question, assume we did that and trace through the program from the beginning to see what happens. (Hint: The B.0 LED won't misbehave the first time through the main *do/loop*, but what happens on the second iteration?)

Before we move on to solving our little problem, there are two points that need to be clarified. The first is that the same problem doesn't occur with pin C.6 because it's fixed as an input pin. When any data byte is assigned to *outpinsC*, the digit that's assigned to pin C.6 is simply ignored because this pin is an input. The second point is that this problem doesn't usually exist in microprocessor-based seven-segment LED circuits because the eighth bit in an output port is typically assigned to the digit's decimal point. Standard practice is to connect the decimal point to pin 7 of the output port, and to assign a value of 0 to bit 7 of the data byte. That way, when you want to light the digit's decimal point, you can just add 128 to the value of the data byte. You may remember that the MAX7219 LED display driver uses that approach. However, in our LED-2x7 project, we can't "waste" the only bi-directional pin we have left!

SOLVING THE PROBLEM

Of course, the simplest way to solve our problem would be to define pin B.0 as an input. However, pin B.0 is the only bi-directional I/O pin that isn't already busy driving our two-digit LED display, so we certainly don't want to take that approach. For example, if we did that we could no longer use the DS18B20 temperature sensor because it requires a bi-directional I/O pin. Fortunately, there's a software solution to the problem, and it only requires the addition of a simple *if/then* statement.

As you know, the 20M2's *outpinsB* special function variable stores the real time values for all the *portB* output pins. *OutpinsB* also has a feature that I don't think we have discussed until now; *outpinsB* is composed of the following eight individually accessible bit variables:

```
outpinB.7:outpinB.6:outpinB.5:
outpinB.4:outpinB.3:outpinB.2:
outpinB.1:outpinB.0.
```

Of course, it's the *outpinB.0* variable that's relevant right now because it stores the real time state of the B.0 output pin. Whenever pin B.0 is high, *outpinB.0* = 1; whenever pin B.0 is low, *outpinB.0* = 0. With this in mind, let's add the following statement to the *displayNum* subroutine just after the *read valTen, segsTen* statement:

```
if outpinB.0 = 1 then
  segsTen = segsTen + 1
endif
```

Now, let's see how that changes things. As we've already discussed, the first time the *displayNum* subroutine is called the binary value of %11110110 (segment data for digit 0) is assigned to the *segsTen* variable. In addition, the LED attached to pin B.0 is lit, so the real time value of *outpinB.0* is 1. Therefore, the "if" condition is true, so 1 is added to *segsTen* which changes its value to %11110111. When that value is assigned to *outpinsB* later in the subroutine, the B.0 LED remains lit due to the 1 in the bit 0 position.

If the B.0 LED happens to be turned off when the *if/then* statement is executed (i.e., *outpinB.0 = 0*), the "if" condition is false, so the value of *segsTen* remains the same (%11110110). When that value is assigned to *outpinsB* later in the subroutine, the B.0 LED remains turned off. Either way, the *if/then* instruction preserves the state of the B.0 LED.

You may be tempted to simplify things by replacing the above *if/then* statement with the following statement:

```
segsTen = segsTen + outpinsB.0.
```

You certainly can do that, because this statement also solves the problem. (If pin B.0 = 1, then 1 is added to *SegsTen*; if pin B.0 = 0, the value of *segsTen* remains the same.) However, it's important to understand that this approach only works because the LED is attached to pin B.0. For example, if the LED were attached to pin B.7, we would need to manipulate bit 7 of the *segsTen* variable. To place a 1 in that bit position requires that we add 128 to *segsTen*.

For example, if *segsTen* equals 5, then 5 (%00000101) plus 128 (&10000000) equals 133 (%10000101). That's why we add 128 to the data value to light the decimal point when we use the MAX7219 LED display driver.

So, modify the *LED-2x7v2-CountBad.bas* program by inserting the above *if/then* statement into the *displayNum* subroutine just after the *read valTen, segsTen* statement, and re-run the program. (You may want to first save the modified program as *LED-2x7v2-CountGood.bas*.) When you run the modified program, both LEDs should behave identically.

ARE WE DONE YET?

I thought that we would finish our discussion of PICAXE temperature measurement capabilities this month, but, as usual, I talk too much, so we're still not done! Next time, we'll take a look at the software that's needed to enable the DS18B20 to report accurate readings throughout its entire temperature range (-67º F to +257º F). I certainly don't expect my freezer to get that cold, or my cold-smoker to get that hot, but you never know.

Also, we'll experiment with the new M2-class *readinternaltemp* command, and compare its accuracy to the readings of the DS18B20.

One thing that we won't be covering is the PICAXE *readtemp12* command. It is more accurate than the *readtemp* command, but it's also much more complicated.

The *readtemp* command is accurate to within one or two degrees Fahrenheit, which is good enough for me!

See you next time. **NV**

Q&A

WHAT'S UP:

Join us as we delve into the basics of electronics as applied to every day problems, like:

- ✔ Simple Magnetizer
- ✔ Random Traffic Lights
- ✔ Headset Amplifier

WITH RUSSELL KINCAID

In this column, I answer 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**

UNIJUNCTION TRANSISTOR

Q You were asked to provide a circuit to blink LEDs in the February issue; something equivalent to the classic blinking neon NE-2 type light "relaxation oscillator," I gathered. There is another way to do this with LEDs that I have been using for years, using a unijunction transistor. It uses one cap, two resistors, one LED, and one UJT, and runs on 5-10 volts. Slightly more complicated than the old NE-2 version but runs on low voltage. You ought to do a story on this.

— Bob Waber

A I guess you are right, Bob. I thought the unijunction was obsolete. Someone gave me 5,000 of them once, but I finally threw them away. This device is 40 or 50 years old, but I see that they are still on the market, so it is worthwhile explaining how they work.

The 2N2646 and 2N4891 are the old style (non-programmable) UJT and are not available anywhere — except from NTE at an exorbitant price. The 2N6028 and 2N6027 are programmable unijunctions which have a different schematic but work the same. The 2N6028 and 2N6027 are the same device, but the 2N6028 has tighter specs. The programmable unijunction is a four-layer device, similar to an SCR (see **Figure 1**). The N layer is biased by RB1 and RB2, and sets the trigger point.

When the top P-N junction is forward-biased, current flows causing an avalanche (negative resistance effect). The anode is pulled down but can't go as low as the old style UJT because of all the junctions. The 2N2646 could go down to a diode drop of ground; the 2N6028 goes only to 1.5 volts. This avalanche effect is a problem for ICs which have many junctions. If the wrong PN junction is forward-biased, the IC will be destroyed.

The relaxation oscillator (**Figure 2**) is easy to design but there are two considerations:

1. R1 must provide enough current to start the avalanche. In the datasheet, this is called peak current (Ip).
2. R1 must not provide more than the holding current, or the UJT will not shut off and start a new cycle. On the datasheet, this is called valley current (Iv).

To find the maximum value for R1, the peak voltage (Vp) is found by: Vp = Vd + Vcc*Rb1/(Rb1+Rb2); then: R1max = (Vcc-Vp)/Ip.

Note: Vd is the P-N diode drop at Ip; about 0.4V.

To find the minimum value for R1, look up the maximum forward-voltage on the datasheet (about 1.5 volts) and the valley current, Iv; then: R1min = (Vcc-Vf)/Iv. Note that the valley current varies with R1, so this is only an approximation.

In **Figure 2**, D1 is an LED that flashes each time the capacitor discharges. Computing the frequency is difficult because the capacitor charging is exponential and depends on where the trigger voltage is relative to Vcc. If you set the gate voltage at 0.63*(Vcc-Vd-Vmin), the frequency will be: F = 1/(R*C).

FIGURE 1.

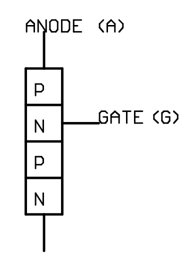
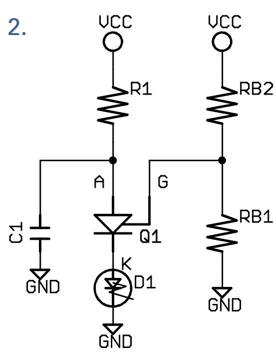
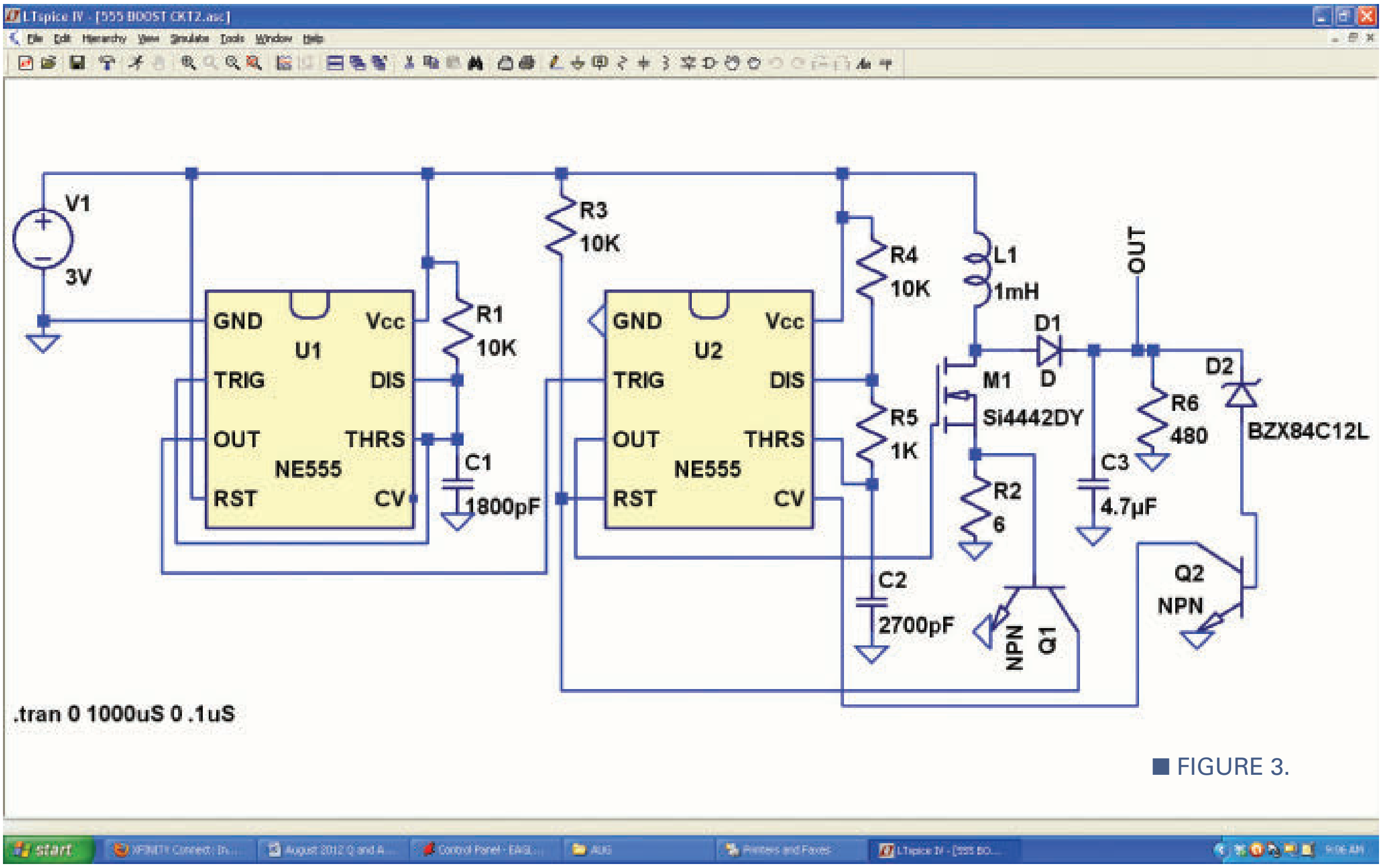


FIGURE 2.





■ FIGURE 3.

SMPS WITH 555

Q I remember seeing a SMPS (switched-mode power supply) boost converter based on a 555 (two actually) awhile back. I was wondering if you had the schematic still. I am challenged to get the classic timer chip to do a modern day function. I want to see how much efficiency I can get out of a simple design. There is one thing that makes the 555 desirable in some sense; the CMOS version can go down to 1.8V (don't quote me on this) or so. However, the TTL version can go down to 3V which is good for battery powered boost converters.

— Chris

A Actually, the TTL 555 is only specified to work from 5V to 15V. If you use it at three volts, you do it at your own risk because it is not guaranteed to work. Most CMOS 555s are specified to work from 2V to 15V, but the LMC555 is specified to work from 1.5V to 15V.

The TTL 555 will source or sink 200 mA at 15V, but the CMOS 555 will source 10 mA and sink 50 mA at 15V. The LMC555 is specified to source 0.25 mA and sink 1 mA at 1.5V. By the way, the 555 in the LTspice library is a behavioral model — not modeling any real world device. It will work down to 1V VCC.

It appears to me that you would like to make a boost converter operating from 3V using a 555. This is how I would go about it: The boost converter operates by charging up an inductor, then dumping the charge into a capacitor. It is necessary to repeat the operation at high frequency in order to have a reasonably sized inductor.

For this exercise, I will design a boost converter for the following specs:

Input = 3 VDC, Output = 12 VDC at 25 mA. The output ripple is to be less than 0.1 Vp/p. The schematic is shown in **Figure 3**. IC1 sets the frequency, triggering IC2 which is PWM'd by feedback to the control pin (pin 5). Q1 limits the peak current by resetting IC2.

Since this is a similar circuit, I will use the design equations for the LM78S40:

Ton/Toff = (Vo+Vd-Vi)/(Vi-Vsat)
where: Vo = output V = 12
Vd = diode drop = .5
Vi = input V = 3
Vsat = Q2 saturation V = .1
Ton/Toff = .305

If the frequency (F) is 50 kHz, then Ton + Toff = 1/50K = .02 mS. For simplicity, let Ton = 9*Toff = 18 µs. This ratio is set by R1/R2.

The capacitor, C1, of IC1 is = 1.44/F/R1 = 2,880 pF when R1 = 10K.

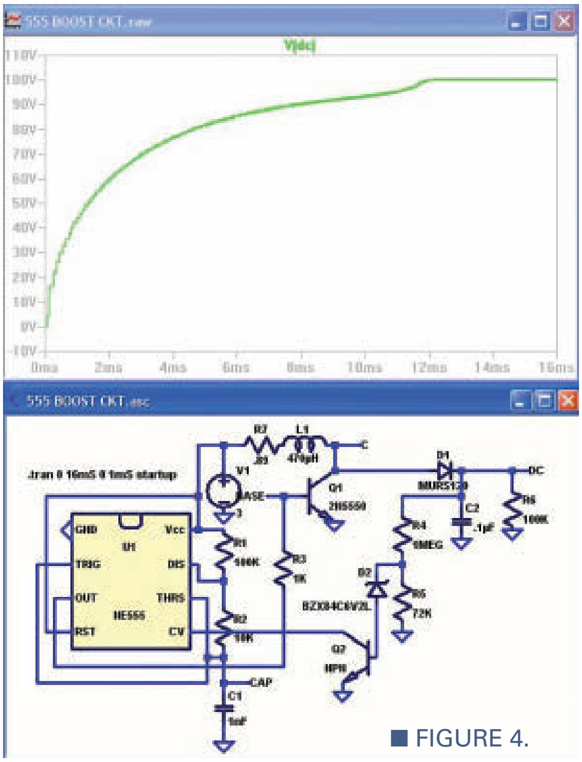
Let C = 2,700 pF (an available value), so then F = 53 kHz.

Ip = 2*Io*(Ton+Toff)/Toff = 2*0.025(.305+1) = 0.065A when

Io = 25 mA.
Lmin = (Vi-Vsat)*Ton/Ip = (3-.1)*18u/.065 = 800 uH.
Choose L = 1000 uH
Rsc = Vb/Ip = .4/.065 = 6 ohms where Vb = Q1 base to emitter voltage.
Output filter cap (Co) = Io*Ton/Vr = .025*18u/.1 = 4.5 µF where Vr = ripple voltage.

I tried to simulate the circuit, but after six hours I stopped it and the simulation had only got to five volts output. If you build it and have problems, let me know. I may be able to help tweak it.

Another circuit — which does simulate — is **Figure 4**. This circuit is intended to produce 100V at 1 mA output. The regulation is not very good and the frequency changes with voltage, but otherwise it is similar to **Figure 3**.



■ FIGURE 4.

SIMPLE MAGNETIZER

Q I have found an old iron magnet that originally was part of a crank telephone from the 1920s to 1950s. It is a horseshoe shape; each leg is 1/2 x 3/4 inches and the length is five inches. It has lost almost all its magnetism and I want to re-magnetize it back to full strength.

Also, I am looking for three books on the subject of magnetism (see **Table 1**). All the books were published before 1960 so may be hard to find, but I need them to own and study.

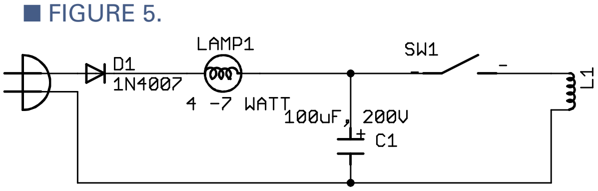
Electromagnetic Theory	Principles of Electricity and Magnetism	Experimental Researches in Electricity and Magnetism
by J. A. Stratton	by G.P. Harnwell	by Michael Faraday
Pub. McGraw-Hill	Pub. McGraw-Hill	Pub. E. P. Dutton

Table 1.

A All those books are available from **Amazon.com**; Faraday's book can be purchased from the National Book Network at (800) 462-6420; price is $175 for the three volume set. The same book from Amazon is more, but used is available for less.

My solution to your magnetizer problem is based on info that I got from the World Wide Web that at least 20,000 ampere-turns is needed to do a good job. The circuit (**Figure 5**) will do double that. If you have an isolation transformer, I recommend you use it. Otherwise, direct connection to the wall socket will work. Just be careful because 120 VAC can be lethal. The night light limits the current so that a one amp diode (like the 1N4007) can be used. The 100 µF capacitor is charged to 160 VDC and when the switch is closed, the current is limited only by the circuit resistance — about 1.6 ohms.

■ FIGURE 5.



The turns in the coil are 417, so the ampere-turns = 41,700 which is more than enough. The capacitor value is not critical — the larger the better — and has low ESR (like Mouser #80-ALC10C152EG250).

The coil is made from a spool of Belden #20 magnet wire (Mouser part 566-8050). The spool becomes the bobbin for the coil. The hole in the spool is one inch in diameter which will accept one leg of the horseshoe magnet. You wouldn't have to wind it except the start end is buried, so you have to unwind the whole spool.

I recommend that you temporarily wind the wire on another spool (or rolled oats box) in order to avoid a mess. Drill a 1/16 hole near the bottom of the winding for the start, and two above the winding to anchor the end wire.

To use the magnetizer, put one leg of the magnet through the hole in the bobbin, put the keeper in place, plug in the power, and when the light is dark, close the switch. One application is all it takes; multiple applications will not increase the magnetization. BTW, I think the keeper should be the same cross-section as the magnet (1/2 x 3/4). A local hardware store will have metal stock.

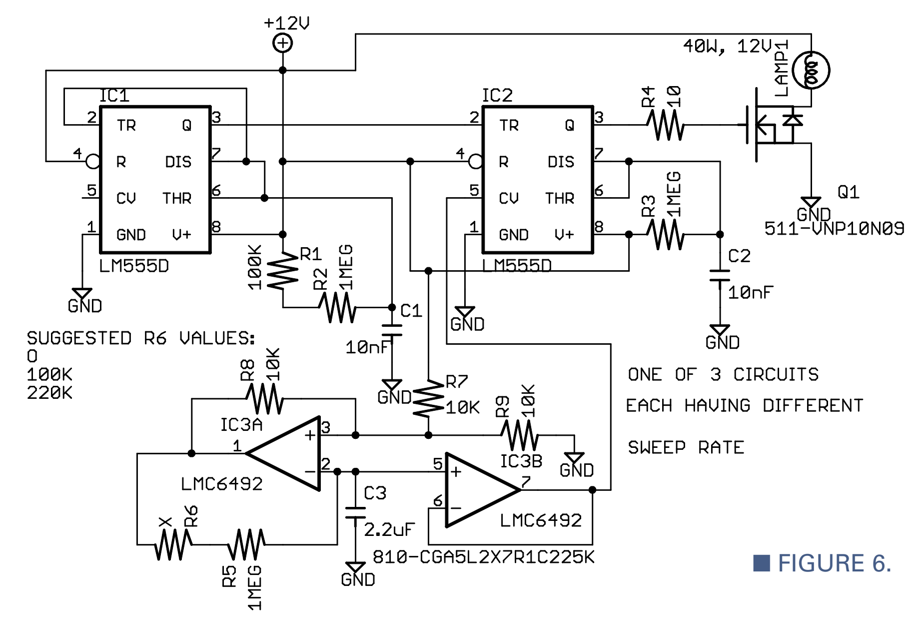
RANDOM TRAFFIC LIGHTS

Q My wife picked up a lighted sign on a stand that is fashioned after a stop light signal: three 110V AC bulbs in a red, green, and yellow housing. While I could design a simple relay driven circuit to step through the lights, I'd like something more eloquent. Could you suggest a circuit that does a "soft" bright/dim cycle (PWM style?) and — if possible — is random between the three lights?

— Mark Bender

A For randomness, you would need a microcontroller and that could also do the PWM. I suggest you change the lights from 110 VAC to 12 VDC; 12V bulbs are available from a camping store. Now, I see that a PIC PWM command does not accept a variable, so that makes it difficult to produce variable PWM. Consequently, I will use three separate sweeps at different frequencies to have a random effect.

The circuit in **Figure 6** is one of three needed for the three lamps. IC1 is the clock — about 100 Hz — and could be common to all three circuits if you want. R1 is included to ensure that the period of IC1 oscillation is longer than that of IC2. The 555 can't be re-triggered while it is timing, so if the period of IC1 is less or equal to the period of IC2, IC2 will not be triggered every time. If the lights suddenly dim during the sweep, R1 needs to be larger.



■ FIGURE 6.

IC2 is PWM'd by a triangle wave generated by IC3. IC3A is a hysteresis oscillator swinging between 1/3 VCC and 2/3 VCC. The frequency of the triangle wave is determined by R5, R6, and C3. The part number of C3 (810-CGA5L2X7R1C225K given on the schematic) is a surface-mount ceramic. If you need a thru-hole part, use 2.2 µF, 35 volt tantalum.

I chose the LM6492 op-amp because it is rail-to-rail input and output, but LM358 or similar will work.

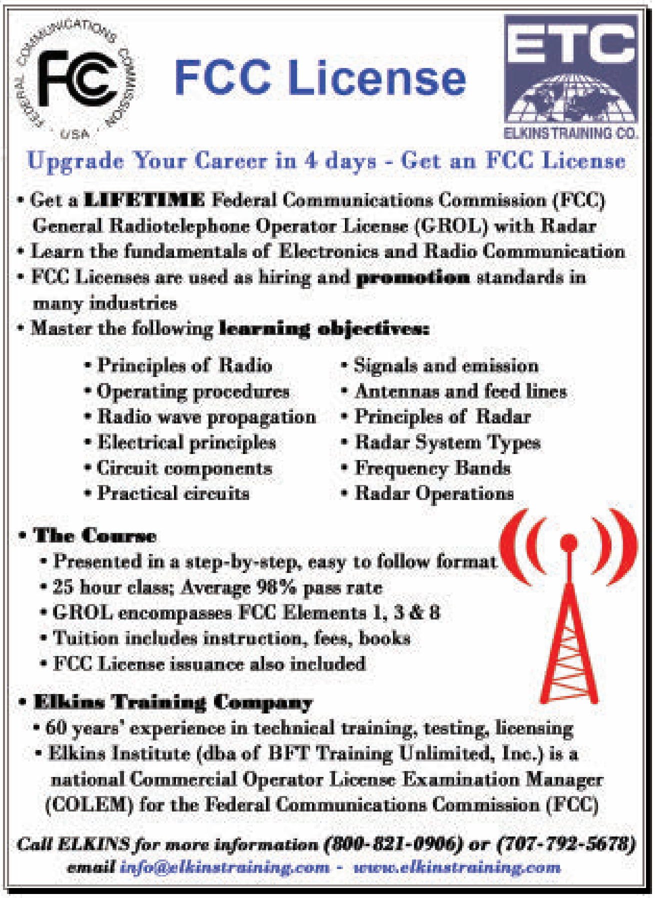
The output switch, Q1, is rated 70V 10A, and has RDSon of 0.1 ohms max. Any TO-220 MOSFET with low RDSon (less than 0.1 ohm) should work without a heatsink.

HEADSET AMPLIFIER

Q I participate in a lot of conference calls and have difficulty understanding some of the other participant's audio. Although I have a substantial high frequency loss, many of the other participants come through loud and clear. However, many other caller's audio sound is "muffled." It's loud enough — just lacking sufficient highs to make it clear. Can you provide a circuit and parts list for an amplifier with an adjustable high frequency boost that I could plug in between my headset and the cordless phone (2.5 mm)? There may be a product out there that will work but I haven't found one.

— Bill P-MA

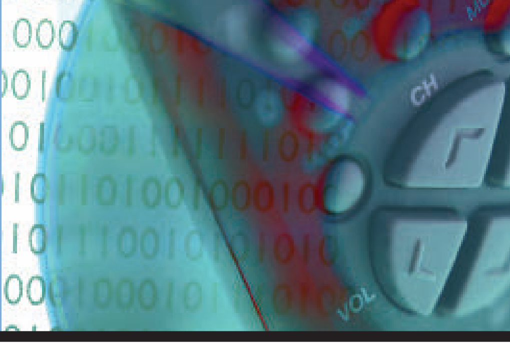
A I'm not sure this circuit (**Figure 7**) will solve your issue, but it provides a max of 20 dB boost above 1 kHz. Both the gain and boost frequency vary with the pot setting. **NV**


FEDERAL COMMUNICATIONS COMMISSION USA
FCC License
ETC
ELKINS TRAINING CO.
Upgrade Your Career in 4 days - Get an FCC License
• Get a LIFETIME Federal Communications Commission (FCC) General Radiotelephone Operator License (GROL) with Radar
• Learn the fundamentals of Electronics and Radio Communication
• FCC Licenses are used as hiring and promotion standards in many industries
• Master the following learning objectives:
• Principles of Radio
• Operating procedures
• Radio wave propagation
• Electrical principles
• Circuit components
• Practical circuits
• Signals and emission
• Antennas and feed lines
• Principles of Radar
• Radar System Types
• Frequency Bands
• Radar Operations
• The Course
• Presented in a step-by-step, easy to follow format
• 25 hour class; Average 98% pass rate
• GROL encompasses FCC Elements 1, 3 & 8
• Tuition includes instruction, fees, books
• FCC License issuance also included
• Elkins Training Company
• 60 years' experience in technical training, testing, licensing
• Elkins Institute (dba of BFT Training Unlimited, Inc.) is a national Commercial Operator License Examination Manager (COLEM) for the Federal Communications Commission (FCC)
Call ELKINS for more information (800-821-0906) or (707-792-5678)
email info@elkinstraining.com - www.elkinstraining.com


RF Specialists
RF Modules
From Part 15 to Part 90 Compliant
Narrow Band FM, UHF Multi-Channel
GSM/GPRS
M2M Solutions
GSM/GPRS modules and modem series
Industrial Bluetooth
OEM, Modules, Wireless Device Servers, RS-232
Long range options, low cost
Data Loggers
Stand Alone and
Wireless Mesh Networking Logger
GPS
OEM Modules and USB ZigBee Sticks,
Mesh Networks
ZigBee Pro
OEM Modules and USB ZigBee Sticks,
Mesh Networks
Wi-Fi
RS232 / 422 / 485 to Wi-Fi Adapter
Connect Data Acquisition Equipment through Serial Port to Wi-Fi network
LEMOS INTERNATIONAL
www.lemosint.com
866.345.3667
sales@lemosint.com


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


Model 324
WORLD'S MOST VERSATILE
CIRCUIT
BOARD HOLDERS
Our line of Circuit Board Holders add versatility & precision to your DIY electronics project. Solder, assemble & organize with ease.
MONTHLY CONTEST
Visit us on Facebook® to post a photo of your creative PanaVise project for a chance to win a PanaVise prize package.
Model 201
PANAVISE®
Innovative Holding Solutions
VISIT US ON
7540 Colbert Drive • Reno • Nevada 89511
1 (800) 759-7535 | www.PanaVise.com



NEW PRODUCTS

- HARDWARE
- SOFTWARE
- GADGETS
- TOOLS

LOGIC DESIGN **TRAINER**

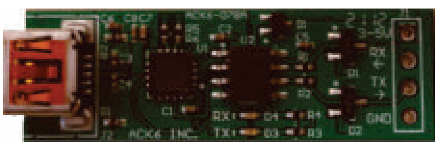
The new and improved Model PB-502 Advanced Logic Design Trainer from Global Specialties is compact and lightweight and lists

for $300. This new trainer offers a 840 tie point breadboard area and eight power rails with 25 tie points each. The PB-502 also features a 1 Hz, 1 kHz, and 100 kHz clock, pulsers, logic switch/input ports, eight SPST logic switches and buffered LED logic indicators, TTL/CMOS operation, seven-segment LCD display, two fully debounced pushbuttons with true logic output ports, and is powered by an external power pack adapter that is included. The new trainer contains all of the necessary functions and features for the design or studying of basic circuits, such as clocks, switches, and lights. The PB-502 is housed in a tough molded plastic black carrying case with integral hinged cover, which stacks conveniently for storage.

For more information, contact:
Global Specialties
Web: **www.globalspecialties.com**

ISOLATED USB-TO-TTL UART CONVERTER

Many electronic boards take advantage of an onboard UART for debugging, testing, and production purposes. The new ACK6-ISO-USB-UART-078A from ACK6 provides a convenient and reliable method to connect any such UART to a USB host such as a PC or laptop.



Board dimensions are 1.5" x 0.5" (38.1 mm x 12.7 mm). Typical current draw from the target board is ~8 mA (all baud rates). It works from 3V to 5V with no jumpers or solder bridges required.

Reverse polarity and ESD protection are built in and the board isolates the USB host from the target board for safety. No special USB drivers are needed since it automatically configures as a COM port. The four-pin 0.1" (2.54 mm) connector allows flexible attachment to a target board. It operates up to 230400 baud.

For more information, contact:
ACK6
Web: **www.ack6.com**

ULTRA HIGH PRECISION **OP-AMPS**

Microchip Technology Inc., has broadened its portfolio of zero-drift operational amplifiers (op-amps)

with the debut of the MCP6V11 and MCP6V31 single amplifiers. Operating with a single supply voltage as low as 1.6V and a quiescent current as low as 7.5 µA, these ultra-high performance devices offer some of the industry's lowest quiescent current for the given bandwidth without sacrificing the optimal performance essential for portable applications in the consumer, industrial, and medical markets.

With an aging world population in need of new therapies and early diagnostic tools, devices like the MCP6V11/31 enable the development of portable medical products integrated with higher efficiency and signal conditioning hardware and software which is critical to accommodate the continued push for lower costs and faster times to market. Designers of industrial applications — such as portable sensor conditioning and instrumentation requiring low power, smaller form factors, temperature considerations, and cost-management — can also benefit from the optimized performance, low quiescent current, and low operating voltage made possible by the MCP6V11/31 op-amps.

Employing Microchip's advanced CMOS technology, the devices require less current to operate the amplifier while simultaneously delivering longer battery life and minimal thermal-related challenges. The self-correcting architecture of the MCP6V11/31 family provides a maximum input offset voltage of 8 µV for ultra-low offset and low offset drift, enabling maximum accuracy across time and temperature. The MCP6V11 offers 80 kHz of gain bandwidth product, with a low typical quiescent current of only 7.5 µA, while the MCP6V31 provides 300 kHz of gain bandwidth product, coupled with a low typical quiescent current of 23 µA. Additionally, the MCP6V11 and MCP6V31 single amplifiers are both available in small five-pin SOT-23 and five-pin SC-70 packages, enabling minimal use of board space, ease of system design, and reduced cost.

For more information, contact:
Microchip
Web: **www.microchip.com**

PROJECT CLOCK VERSION 10 RELEASED

CyberMatrix Corporation has released version 10 of Project Clock — an easy to use project time recording and time billing program for Windows. Project Clock 10 contains several important improvements. The most significant change in version 10 is the new billing module that provides a more convenient way to bill clients for time and expenses. Full Unicode support was added to allow translations of all multi-byte languages such as Chinese, Japanese, and Russian. In addition, a new enterprise edition of Project Clock that can connect to virtually any database server is now available.

Project Clock is used by consultants, lawyers, engineers, contractors, project managers, or anyone else who must accurately record the time spent on different projects or tasks.

Continued on page 76

SHOWCASE

BUILD THE VOLTAGE MIRROR

By Jim Stewart

Recently, I designed and prototyped a battery-powered piece of analog equipment to run off two nine volt batteries. After the prototype was finished, it was decided that the device needed to have the option of using a wall-mount power supply, as well as batteries. Wall-mount supplies with +9V and -9V outputs are not easy to find, but ones supplying 12 to 15 volts DC are readily available.

Instead of redesigning the main printed circuit board (PCB), I decided to use a small add-on board that would fit inside the enclosure. On the board would be an LM317 regulator to bring the supply voltage down to nine volts. Also on the board would be a TC962 charge-pump DC-to-DC converter. You supply $+V_X$ to the TC962 and it gives back $-V_X$. It's a voltage mirror.

A glass mirror gives an exact copy of an image but with left and right swapped. A voltage mirror gives a copy of the input voltage but with the polarity swapped.

Circuit Description

Figure 1 shows the schematic of the circuit. Jumpers J1 and J2 allow the circuit to be used with either a nine volt battery or a DC wall-mount supply. The bars above the jumpers (J1 and J2) show which ones to use for a wall supply. For a battery, slide the J1 jumper left and the J2 jumper right. When jumpered for a battery, the LM317 is bypassed. We'll discuss J3 later. For the LM317, a 0.1 µF capacitor at the input is recommended when the IC is more than six inches from the DC source. Such would be the case when using a wall supply, so we have C1 at the input.

Diode D1 is there because of my firm belief in "Murphy's Law." In this case, Murphy says that if the input polarity *can* be reversed, then eventually it *will* be reversed. If you're not a believer, you can replace D1 with a jumper.

The output voltage of the LM317 regulator is given by the formula:

V_{OUT} = 1.25V ´ (1 + R2/R1) + R2 ´ I_{ADJ}

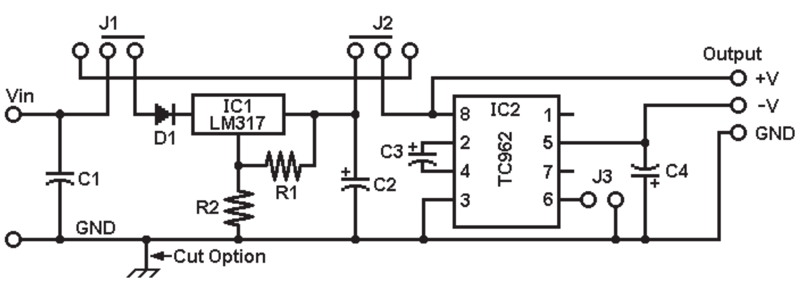
I wanted nine volts out, so I chose R1 = 200 Ω and R2 = 1.24 kΩ; both 1% resistors. A different value of V_{OUT} requires a different pair of values. Keep R2 low so that the R2 x I_{ADJ} term is relatively small. I_{ADJ} is in the range of 40 µA to 100 µA, and changes with temperature.

Capacitors C2, C3, and C4 are 10 µF electrolytics. I used tantalum, but a good aluminum cap should work fine. C2 is the output filter for the LM317. C4 is the filter for the negative output voltage of the TC962. C3 is the charge pump capacitor for the TC962. (See the **sidebar** for a description of how a charge pump works.) The TC962 charge pump runs at 12 kHz, but if you connect pin 6 to ground, the speed doubles to 24 kHz. The higher frequency allows a smaller value for the output filter capacitor (C4). J3 allows you to connect pin 6 to ground.

The Circuit Board

Figure 2A shows the component side of a small PCB for the circuit. The board dimensions are 2.8 by 1.3 inches. The D-PAK surface-mount version of the LM317 is used; it's soldered to an area of copper to dissipate heat. (Note: A D-PAK is easy to solder, so don't worry if you've never used surface-mount devices.) **Figure 2B** shows the solder side. There is a mounting hole at each corner of the board big enough for a #4 screw. The mounting hole with the square outline is connected to circuit ground, so mounting the board to a metal chassis will connect ground to the chassis. If that's not desirable, you can cut the etch as indicated.

Figure 3 is a photo of the assembled board. I used terminal blocks for input and output



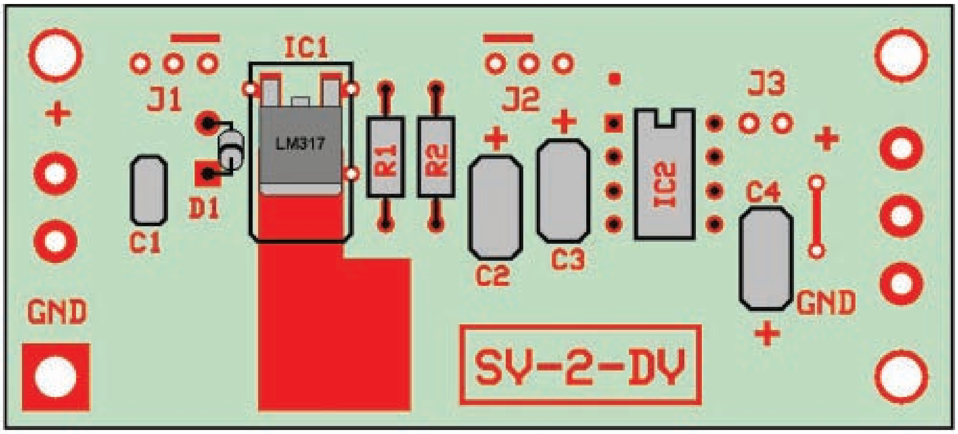
■ **FIGURE 1. Voltage mirror schematic.**

voltages, but that's optional. Also, I used pins and jumper blocks for J1, J2, and J3. You may want to just solder in jumper wires. Note the stand-offs mounted at each corner.

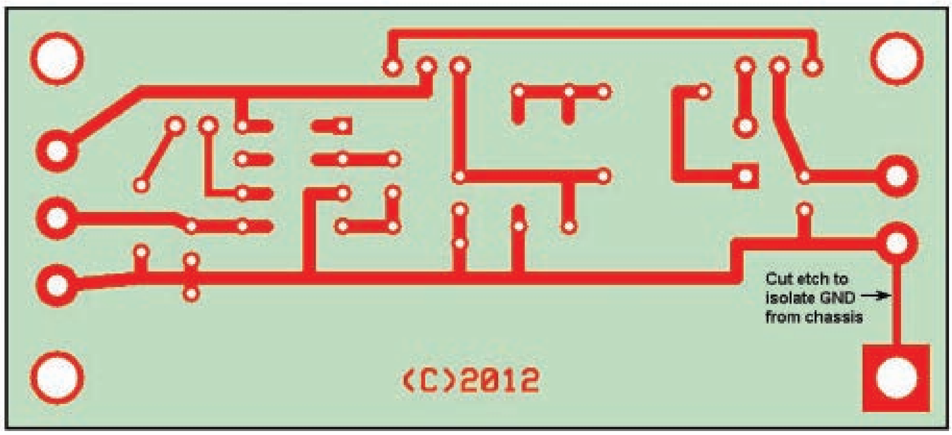
Performance

The first board I assembled started to smoke when I applied the input voltage. After checking the board several times, I decided it really *should* have worked. As it turned out, several of the tantalum capacitors I had were marked incorrectly; the polarity was reversed. That's what I get for buying them at a flea-market. Once the caps were replaced, it worked fine. With no load, V_{OUT} was -9.01V when V_{in} was +9.02V. Using a 500Ω load and measuring the drop in output voltage, I calculated that the circuit had a source resistance of 35 ohms. That was consistent with the TC962 datasheet value. The maximum output current of a TC962 is 80 mA.

To measure ripple from the switching circuit, I put a 1 kΩ load on the output of the TC962. With the J3 jumper installed, I measured 50 mVpp at 24 kHz. With the jumper removed, I measured 90 mVpp at 12 kHz. Increasing the value of C4 should reduce the ripple.



■ **FIGURE 2A. Voltage mirror circuit board.**



■ **FIGURE 2B. Solder side.**

Assembly

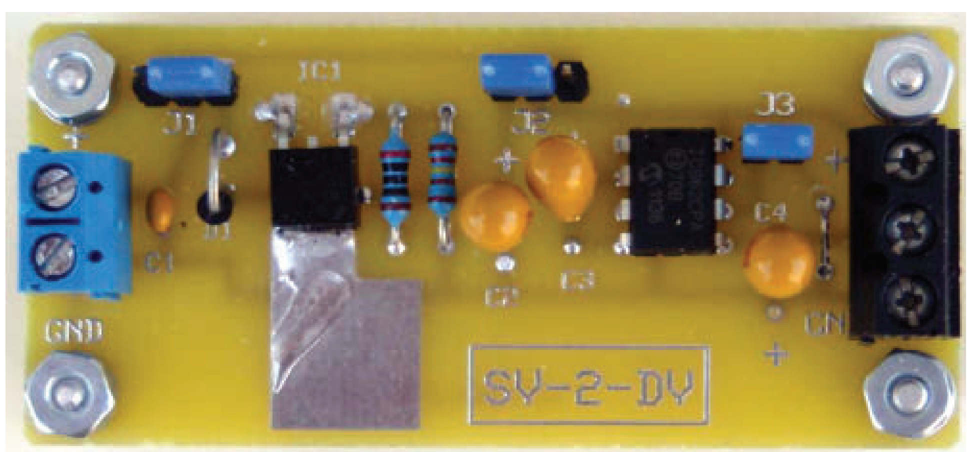
Assembly is easy. Start with the LM317 (refer to **Figure 2A**). First, apply a bit of solder to the two pin pads and the copper that will be under the chip. (That's a *bit* of solder, not a big glob.) Position the D-PAK chip onto the pads and hold it in place with a finger; use just enough pressure to hold it steady.

Then, apply your soldering iron to one of the two small pins. The solder you applied earlier should melt and flow over the pin. Remove the iron, and let the pin cool for a few seconds. Now, do the same to the other small pin. At this point, you no longer need to keep your finger on the chip.

Apply solder to the metal edge of the D-PAK that is lying on the patch of copper. Make sure the solder flows evenly and bonds the metal edge to the copper. Let it cool for a few seconds.

Next, mount the eight-pin DIP TC962. Make sure pin 1 is in the hole with the rectangular outline as shown in **Figure 2A**. On the solder side, bend the leads over to secure the chip in place. Then, solder the pins.

Next, mount the diode. Note that it's mounted



■ **FIGURE 3. Assembled board.**

Charge Pumps

To step a DC voltage up or down, or to convert +V to -V, it is common to use circuits that contain inductors or transformers. However, inductors and transformers can be bulky and expensive. So, when power requirements are low, it often makes more sense to use capacitor-based charge pumps. Charge pumps are available as inexpensive integrated circuits such as the TC962.

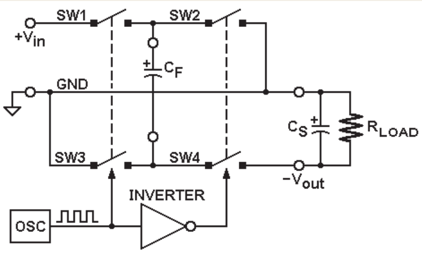


Figure SB1. Charge Pump Block Diagram.

Figure SB1 shows the block diagram of a charge pump used to convert a positive DC voltage to a negative DC voltage. An internal oscillator drives transistor-switches that control the charging and discharging of the two external capacitors C_F and C_S. C_F is the so-called flying capacitor, while C_S is the charge-storing capacitor for the output.

Operation is as follows:

- Step 1: Switches SW1 and SW3 are closed which connects the top of C_F to $+V_{in}$ and the bottom of C_F to ground. So, C_F charges up to +V. Switches SW2 and SW4 are open.
- Step 2: Switches SW1 and SW3 open, and switches SW2 and SW4 close. SW2 connects the top of C_F (the positive end) to ground, while SW4 connects the bottom of C_F to $-V_{out}$.
- Step 3: While SW2 and SW4 are closed, charge is transferred from C_F to C_S.
- Step 4: SW2 and SW4 open, while SW1 and SW3 close. The cycle repeats.

Voltage Doubler

A charge pump can also be used as a voltage doubler as shown in **Figure SB2**. The value of the output voltage is actually Vout = 2V - 2Vd, where Vd is the voltage drop across the diode.

Charge pumps can also generate 3V and V/2, as well as other voltage ratios such as 3V/2, 4V/3, 2V/3, and so on.

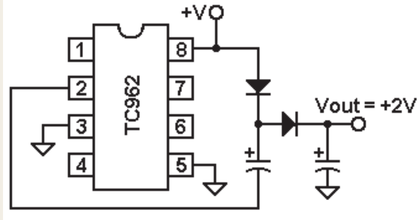


Figure SB2. Voltage Doubler.

Source Impedance

A charge pump can be modeled as a voltage (V) in series with an impedance (Z_S) as shown in **Figure SB3**. Z_S is composed of the reactance of C_F (X_C) and the resistance of the switch (R_S). A high switching frequency (f) keeps X_C low since $X_C = 1/(2 \pi f C)$. R_S depends on the size of the MOSFET switching transistors; in an IC, the transistors are relatively small. So, the value of Z_S is mostly determined by R_S. For the TC962, R_S is about 35 ohms.

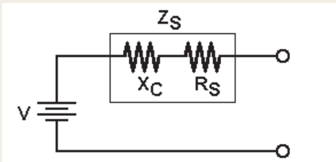


Figure SB3. Source Impedance.

A value of 35 ohms may not sound like much, but at a load current of 20 mA it causes a 0.7 volt drop in output voltage. The bottom line is that devices like the TC962 work best in low power applications.

Efficiency

Charge pumps have two specifications for efficiency: voltage conversion efficiency (ηV) and power conversion efficiency (η_P). Voltage conversion efficiency is the measure of how well the magnitude of V_{out} matches the magnitude of V_{in}: $\eta V = |V_{out} / V_{in}| \times 100\%$. Power conversion efficiency is $\eta P = (P_{out} / P_{in}) \times 100\%$.

For the TC962, voltage conversion efficiency is 99.9%, so the two magnitudes would differ by no more than 0.1% with no load. The power conversion efficiency is 97%, so 3% of the input power is lost in the process of converting +V to -V.

vertically. Make sure the cathode (the banded end) is in the hole with a square outline (refer to **Figure 2A**). On the solder side, bend the leads out to hold the diode in place. Now, solder them and cut off the excess lead lengths. Next, insert the two resistors, being careful not to mix them up. The smaller value (200Ω) should be closer to the LM317. Solder them in as you did with the diode.

Next, insert the capacitors. Verify that C2, C3, and C4 are mounted with the correct polarity. The positive end for each cap is indicated by a plus sign (**+**) on the board. Solder them in as you did with the resistors.

Next, insert jumpers at J1 and J2. On the board, a line of copper is above each set of holes showing the connections for a wall-mount supply. To use a battery, insert a jumper between the middle hole and the left hole at J1, and a jumper between the middle hole and the right hole at J2. If you're using terminal blocks for V_{IN} and V_{OUT}, mount them now. Okay, you're finished building it.

Wrapping Up

Clean off the solder side with some rubbing alcohol and a scrub brush. Then, *before* applying power, give the assembled board a good visual inspection. Fix any problems you find. Apply V_{IN} and measure $+V_{OUT}$ and $-V_{OUT}$ with your multimeter. You should have a working unit. Now, find a project where you can put it to good use. **NV**

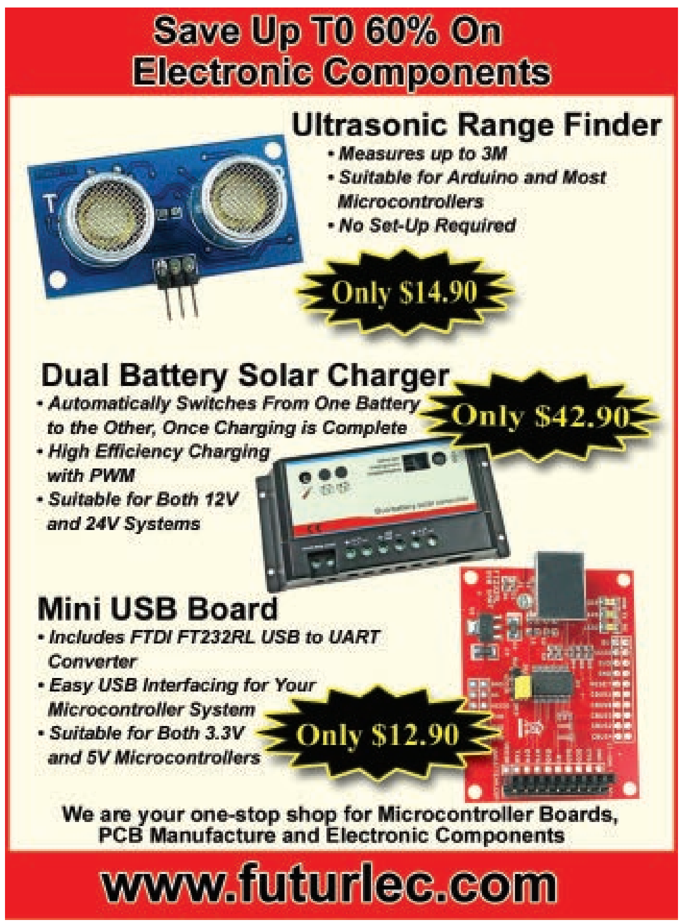
PARTS LIST

ITEM	DESCRIPTION	SOURCE
IC1	LM317 D-PAK package	Jameco part #838006
IC2	TC962 eight-pin DIP	
D1	1N4001 or equivalent	
R1	200 Ω, 1/4W, 1%	
R2	1.24 kΩ , 1/4W, 1%	
C1	0.1 µF, ceramic, 50V	
C2	10 µF, electrolytic, 35V	
C3	10 µF, electrolytic, 35V	
C4	10 µF, electrolytic, 35V	
Printed circuit board	SV-2-DV	
Optional components		
Terminal block: two-position	5 mm spacing	Jameco part #2094485
Terminal block: three-position	5 mm spacing	Jameco part #2094493
Shorting block: three pieces	0.1 in spacing	Jameco part #112432
Male header strip, eight-pin,	0.1 in spacing	Jameco part #153702


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


RS485/422/232/TTL
ASC24T
RS232<=>RS485 ATE Converter
$45.00 board only
IBS485HV
5 Port Isolated RS485 Repeater
$349.00
Enclosures, Cables, Power Supplies and other Accessories
Converters / Repeaters
Multi-Repeaters / Hubs
Fiber Optics / Isolators
Extended Distance Units
Serial to Digital I/O
Large Multi-Drop Networks
Custom Units & Smart Units
Industrial, 3.0 KV Isolation
Call the RS485 Wizards
513-874-4796
RES R.E. Smith Inc.
www.rs485.com


Hobby Boards
Monitoring and automation for professionals and hobbyists alike.
Many Uses:
Weather Monitoring
Agriculture
Home Automation
Yard and Garden
New Products:
Moisture Meters
1-Wire Sniffer
Visit our newly re-launched website, www.hobbyboards.com, for more information and a complete listing of products.
HOBBY BOARDS
Not just for hobbyists!
www.hobbyboards.com


Save Up T0 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

BUILD THE MYSTERY SOLAR POWERED PENDULUM

By David Williams

Powered using the energy of the sun, this solar pendulum does not require batteries or an electrical outlet to operate. So, how does the solar energy make the pendulum swing with no visible connections?

It will have your family and friends guessing until you reveal the secrets of the electromagnetic coil and circuit hidden in the base.

This device is a wonderful demonstration of clean, green solar power and is based on BEAM technology. It was inspired by the Magbot pendulum project in Dave Hrynkiw's book titled, *Junkbots, Bugbots & Bots On Wheels*. When the solar cell is exposed to the sun or any bright light source, the pendulum will start to swing — almost magically. Each time the pendulum passes the bottom of its swing, a hidden coil in the base gives the pendulum a little nudge, using stored electricity generated by the solar cell.

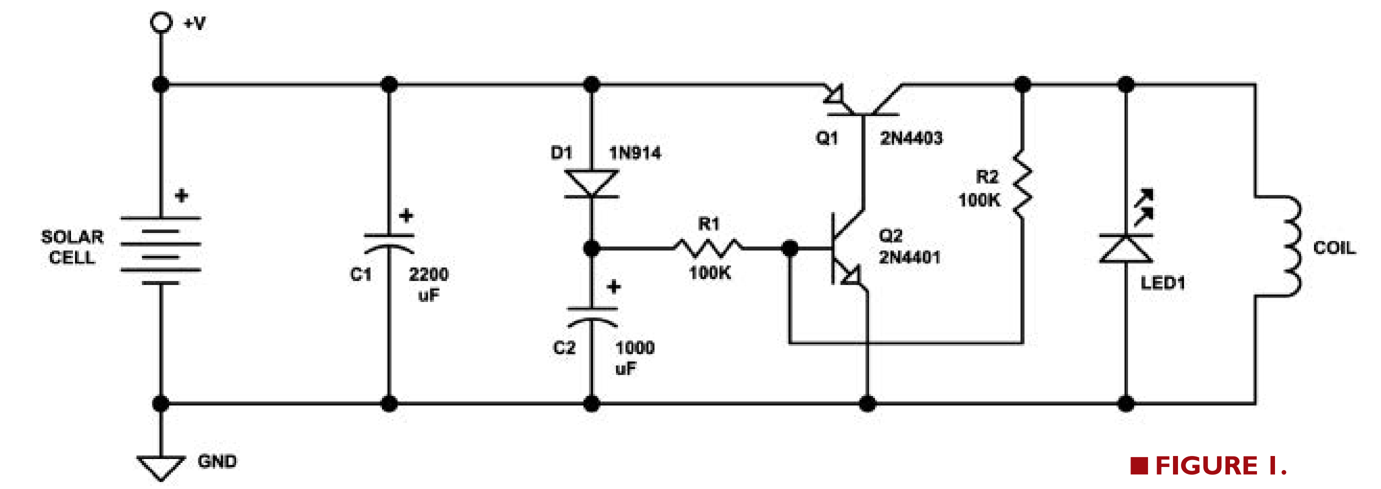
The invisible nudges keep the pendulum in perpetual motion as long as there is enough light. The solar pendulum is both fun to watch and is a learning tool, all in one. It can also be a great school science project, demonstrating both solar energy and electromagnetic principles.

How the Circuit Works

Figure 1 shows the simple schematic for this project. As light strikes the solar cell, it generates a DC voltage that begins to charge up the two electrolytic capacitors, C1 and C2. If the pendulum is not already swinging, the circuit acts in a mode that tries to "kick-start" the pendulum into motion. During the *kick start* mode, both electrolytic capacitors continue to charge until the voltage on C2 reaches approximately 0.8 volts. At that point, the NPN transistor Q2 turns on which, in turn, causes the PNP transistor Q1 to turn on. When Q1 turns on, it dumps all the power stored in C1 into the coil. The current flowing through the coil briefly creates a magnetic field around it, as C1 quickly discharges. Then with no more current flow, both transistors turn off and the magnetic field collapses, generating a negative voltage spike which causes the LED (light emitting diode) to flash.

The brief magnetic field created around the coil will cause the magnet attached to the pendulum to jump as it interacts with this invisible force. The *kick start* cycle will repeat if the pendulum does not start swinging right away. With a bit of luck, the jump of the magnet should eventually start the pendulum swinging, even if it is just a small swing at first. Of course if you are impatient, you can always skip the *kick start* by giving the pendulum your own nudge to start it swinging.

Once the pendulum is already swinging, the circuit will act in a way to keep it swinging. It should be noted that C2 does not discharge into the coil like C1 because of diode D1. As the magnet on the moving pendulum passes over the coil, it induces a current in the coil which triggers Q2 to turn on via resistor R2. When Q2 turns on, the same sequence of events occurs as in the *kick start*. In this mode, however, the magnetic field around the coil gives the swinging pendulum a boost to increase its momentum. This boost is like a parent pushing a child on a swing. Without the added pushes,



■ FIGURE 1.

the swing would eventually slow to a stop.

As long as the solar cell can keep recharging capacitor C1 (even if it is only a partial charge), the pendulum will trigger this invisible nudge on each swing past the coil.

Construction

As you look through the **Parts List**, you will see that this project uses fairly common parts and materials, but it does involve some fabrication. You can view a short video of the finished pendulum in action at **www.techkits.com/kits/#solpend**.

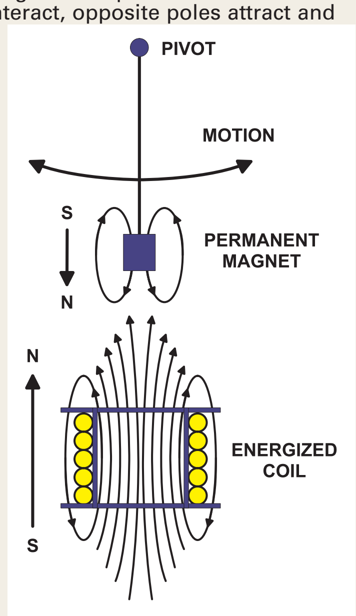
Start construction with the printed circuit board (PCB). Locate the two 100K ohm resistors (R1 and R2), then install and solder them both in place as shown in **Figure 2**. Next, install the 1N914 diode (D1), making sure the polarity band is oriented properly. Locate the

Motion From Magnets

Ampere's Law tells us that when a current is passed through a wire, an invisible magnetic field is produced around it. When the wire is wrapped into a circular coil, those lines of magnetic force work together to increase the strength of the field. The strength of the field in a coil is proportional to the current flowing in the wire, and the number of turns of wire.

An energized coil of wire behaves as a round magnet, with one end being a north pole and the other end becoming a south pole. When two magnetic fields interact, opposite poles attract and like poles repel each other. If one field is held stationary and the other is allowed to move, these attracting and repelling forces can be used to create motion.

This fundamental principle is the basis for many industrial products including stepper motors, servo motors, permanent magnet DC motors, and permanent magnet generators.



PARTS LIST

ITEM	DESCRIPTION	SOURCE
R1, R2	100K, 1/4W Resistor	
D1	1N914 or 1N4148 Diode	Jameco (179207)
Q1	2N4403 PNP Transistor	Digi-Key (2N4403-ND)
Q2	2N4401 NPN Transistor	Digi-Key (2N4401TFCT-ND)
LED1	T1 size, LED	Digi-Key (754-1218-ND)
C1	2,200 µF, 10V Electrolytic Capacitor	Jameco (606563)
C2	1,000 µF, 16V Electrolytic Capacitor	Jameco (30015)
Solar Cell	4V @ 20 mA	Futurlec (SZGD4026)
Base	Keystone 705 Phenolic Enclosure	Digi-Key (705K-ND)
Lid for Base	Keystone 705 Cover	Digi-Key (2042K-ND)

Miscellaneous
Magnet, coil, printed circuit board, #4-40 screws and nuts, Aluminum hex spacer, #8-32 threaded rods and nuts

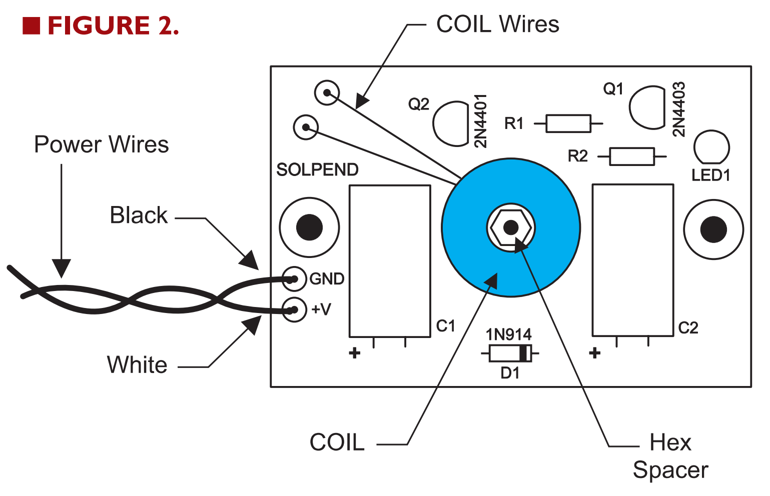
Parts Suppliers

Digi-Key Corp.	**www.digikey.com**
Jameco Electronics	**www.jameco.com**
Tap Plastics	**www.tapplastics.com**
Futurlec	**www.futurlec.com**

Files to create your own PCB are available at the article link. The coil for this project is rated at 6 VDC/32 ohms and can be constructed by winding 150 feet of #33 magnet wire on a plastic round core bobbin (available from Cosmo Corporation; **www.cosmocorp.com**; part #7226-0).

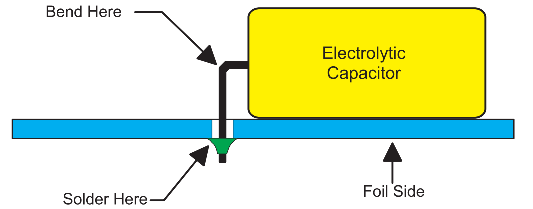
The following items can be purchased online from:
LNS Technologies
PO Box 501
Vacaville, CA 95696
www.techkits.com/kits
Email: LNSTECH@TECHKITS.COM

SOLPEND-KIT	The complete kit of all parts for $39 plus $8 shipping.
SOLPEND-PCB	Just the printed circuit board for this project for $10 plus $4 shipping.
SOLPEND-COIL	Just the coil for this project for $8 plus $4 shipping.



■ FIGURE 2.

■ FIGURE 3.

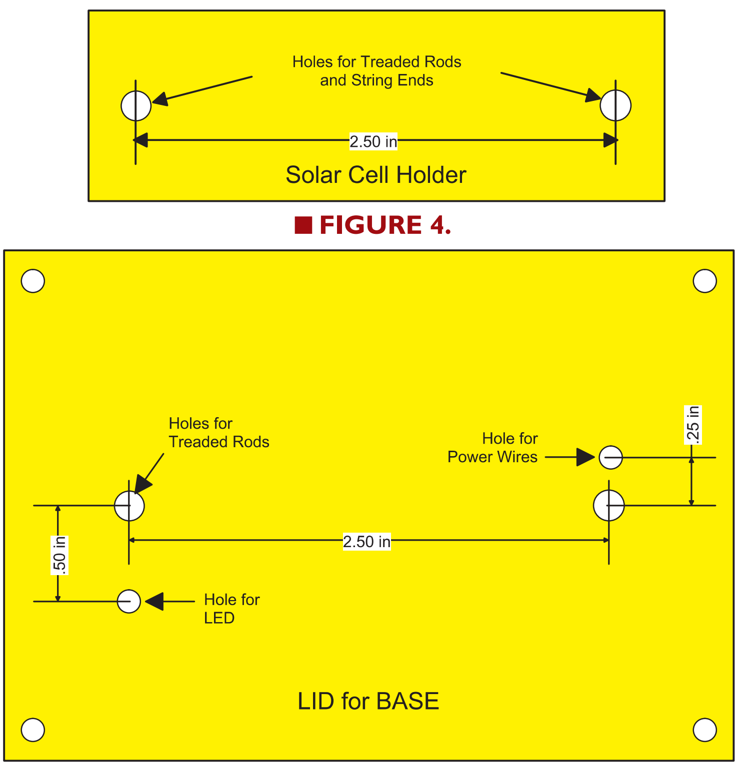


two transistors (Q1 and Q2). They look similar, but Q1 is a PNP and Q2 is an NPN, so they are not interchangeable. Be sure each transistor gets mounted in the proper location and orientation (refer again to **Figure 2**).

Before installing capacitors C1 and C2, take a look at **Figure 3**. Both capacitors C1 and C2 have to be mounted on their sides — against the circuit board — to provide clearance for the enclosure's plastic cover. Bend both wires as shown before soldering each capacitor. An aluminum spacer is mounted on the PCB in order to position the coil directly below the pendulum. Make sure that the spacer is on the component side and the screw is inserted on the bottom side of the board. Now, place the coil over the spacer and solder the two wires as shown in **Figure 2**.

Next, install the T1 LED on the PCB. If you look carefully at the LED, you will see that one leg is shorter than the other; the LED may have a flat edge near one of the legs. The flat spot or short leg indicates the cathode leg. The LED should be mounted at a height of 3/4" from the PCB. This height will allow the LED to protrude through the hole in the enclosure's cover during final assembly. Lastly, attach two 10" wires to the PCB as shown in **Figure 2**. The other ends of these wires will attach to the solar cell later.

Now, move on to the enclosure that will become the pendulum's base, and will house the circuit board and coil. Carefully drill the four holes in the lid, at the locations shown in **Figure 4**. The solar cell should be mounted at the top of the threaded rods to collect as much light as possible. Cut a small piece of plastic 1" x 3" to hold the solar cell, and drill two holes for the rods.



■ FIGURE 4.

Then, assemble the threaded rods, hex nuts, circuit board, and solar cell holder to the enclosure's lid as shown in **Figure 5**. Be sure to use stainless steel rods and nuts, so the metal parts will not interfere with the magnetic field. Choose some thread and a small object for the pendulum. Use the thread to suspend the pendulum object from the plastic piece at the top of the rods as shown in **Figure 5**.

The solar cell for this project should output at least 4.0V @ 20 mA. On the bottom side of the solar cell, you will see two pads for soldering near the corners. Be sure to note the polarity marks for "+" and "-" next to each pad. The power wire that goes to +V on the PCB gets soldered to the + pad on the solar cell. Likewise, the power wire from GND on the PCB gets soldered to the **-** pad. Try not to apply excessive heat when soldering the wires to the solar cell to avoid damaging it.

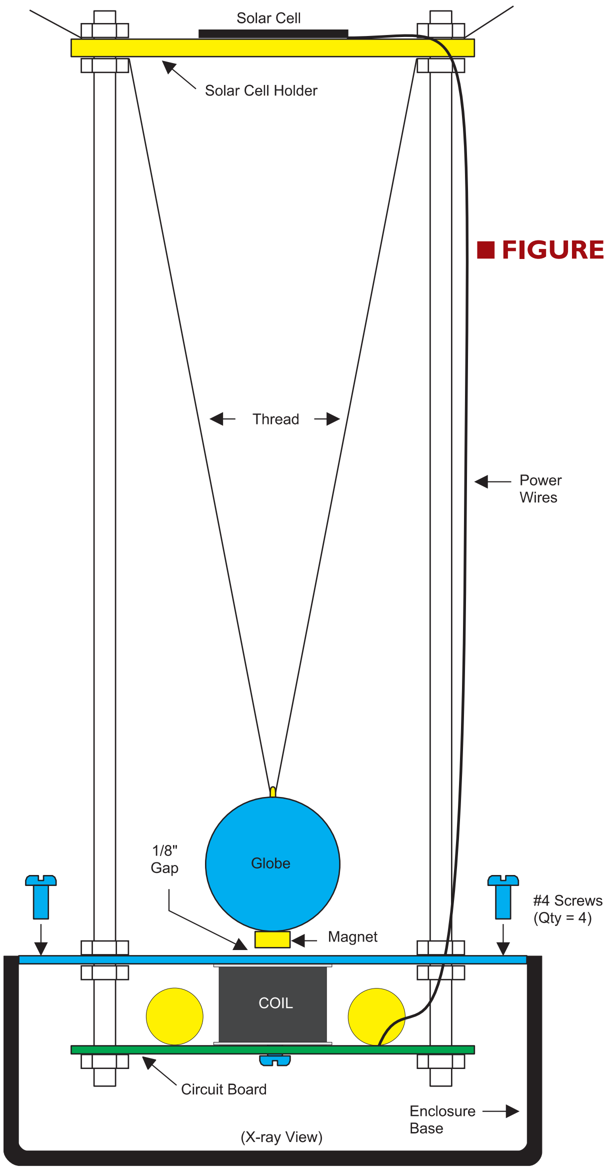
Final Assembly

Refer again to the final assembly diagram (**Figure 5**). The black plastic enclosure (shown in the **Parts List**) becomes the base of the solar pendulum. The whole previous assembly is installed on the base by using four #4 screws to attach the cover to the enclosure.

The pendulum object needs to have a small neodymium magnet attached to the very bottom. If you use a steel globe like in the **photos**, the magnet will stick by itself. If you use a non-metallic object for the pendulum, the magnet will need to be glued in place.

The last assembly step is to adjust the string so that the gap between the magnet and the cover is at least 1/8 inch and no more than 1/4 inch. The upper nuts on the top plastic holder must be loosened in order to adjust the string.

After each adjustment of the string, make sure the pendulum object is still hanging centered between the two threaded rods before checking the gap. For the best performance, try to get the gap at 1/8 inch if possible. Tighten the top nuts to lock the string in place once the pendulum gap is set.



■ FIGURE 5.

Operation

Operation of your solar pendulum is straightforward — simply expose it to sufficient light and the pendulum object should start swinging on its own. Or, you can always give it a starting nudge. Once it is swinging, you should see the LED flash each time the pendulum passes over the hidden coil. Note that direct or indirect sunlight works the best. An incandescent bulb works okay if held close enough, but a fluorescent light will not work at all.

This project is a hit with audiences of all ages and it will keep you and anybody else who watches it mesmerized for hours. **NV**

BUILD THE ALARM AUTODIALER

By Dan Gravatt

Anyone who owns a house and is away at work or on a vacation will at some point wonder "What's going on at home?" Not necessarily with the kids, but with the house itself. Even newer homes can have problems with heating, cooling, plumbing, or electrical systems which are easier and less expensive to fix when discovered sooner rather than later. This device will detect and report several common failures with household systems, so you can deal with them while they are still easy to fix. It's inexpensive ($60 or less, depending on your parts collection), easy to build, and doesn't require any contracts with monitoring services.

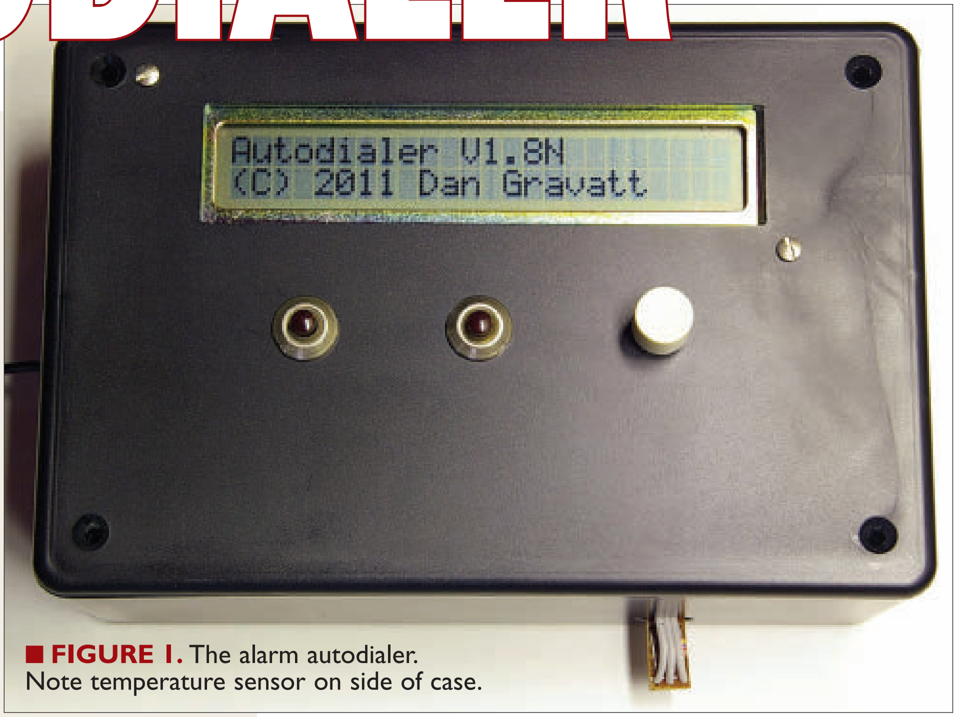
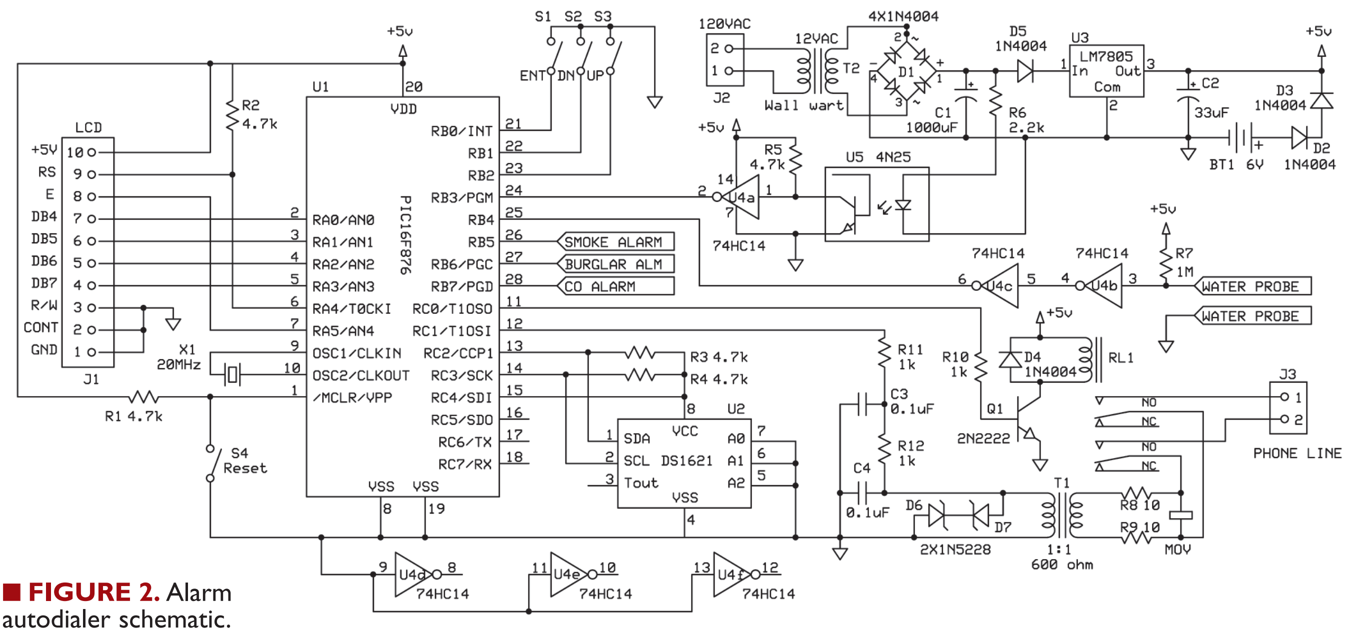


FIGURE 1. The alarm autodialer. Note temperature sensor on side of case.

PROJECT GOALS

Of course, there are off-the-shelf systems and services out there for automated reporting from smoke alarms and security systems, but I wasn't interested in simply duplicating those capabilities (though this project can be used to monitor those types of alarms). I wanted to monitor what I think of as the "Big 4" failure types: extended power failure; air conditioner failure; furnace failure; and flooding. I wanted to be notified by phone of each different type of problem as it occurred. Finally, I wanted to be notified if the problems corrected themselves (electrical power being restored, for example) before I cut my vacation short or called my neighbor to check on my house.

This project meets those goals without needing a lot of direct connections to the systems being monitored. Power failure is detected with an optocoupler interface to the AC mains (through a wall wart type power supply) and a battery backup power supply which keeps the system running and reporting additional problems as they occur. Air conditioner and furnace operation is monitored with a DS1621 I^2C temperature sensor and user-defined high and low temperature limits. Flooding is detected with simple pairs of electrodes which can be located to check for burst pipes (in the winter), failed sump pumps (in the rainy season), or both. Three additional inputs are available for monitoring other systems as needed.



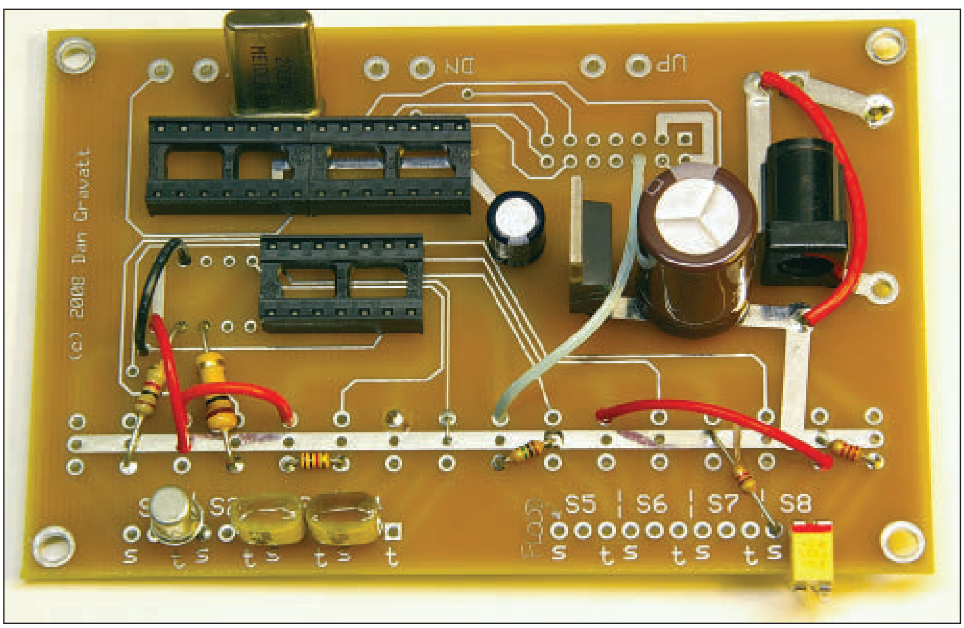
■ **FIGURE 2.** Alarm autodialer schematic.

CIRCUIT DESIGN

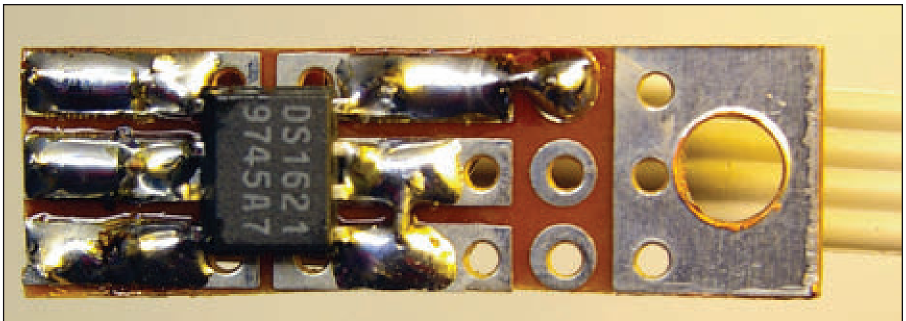
As shown in **Figure 2**, the autodialer is based on a PIC16F876A microcontroller which controls the user interface (a 2x20 character LCD and three pushbuttons), monitors the sensors, and sends data to the telephone line interface. The LCD is based on the standard HD44780 controller and uses a four-bit data interface. Note the presence of R2, a 4.7K pull-up resistor on open-drain pin RA4 that is needed for reliable output function. Although I have used RA4 as an output without a pull-up resistor in other projects, I wasted several hours with this device proving to myself that it doesn't always work without the pull-up resistor. **Figure 3** shows the main board which is actually a leftover custom-made part from a previous project. I recommend that any time you have a custom board made, design as much flexibility into it as you can (extra pads, traces, and pads on all IC pins, etc.) so that you can use any extras for other projects.

The Maxim (formerly Dallas Semiconductor) DS1621 is a very capable and easy-to-use part for measuring and controlling temperatures in degrees Centigrade. A standard two-wire I^2C interface to the PIC with two pull-up resistors is all that is needed. As it is the only device on our I^2C bus, its address is set to zero by grounding all three of its address pins. The Tout pin — which can be used as a thermostat output — is not used in this project. The part I'm using was scavenged from an old laptop where the Tout function was used to control the CPU fan. I have mounted it on a separate circuit board (**Figure 4**) that protrudes from the case, so it can better measure the ambient temperature.

I have had this project in operation in my house for more than a year now, and have found that the DS1621 will lock up every few weeks or so and cease to report



■ **FIGURE 3.** Main board. Note creative reuse of custom board from an old project.



■ **FIGURE 4.** DS1621 sensor board.

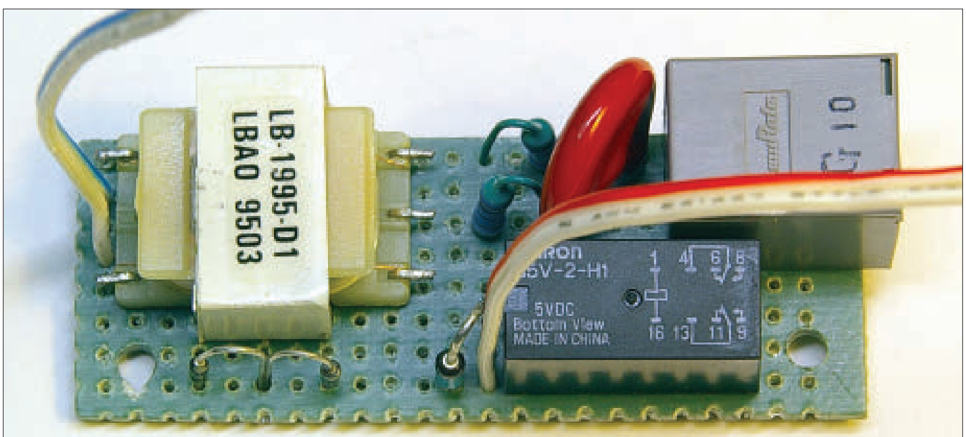
current temperatures to the PIC. Resetting the PIC does not reset the DS1621. A search of the Internet found that this lock-up is a known issue with the DS1621, but I did not find a good explanation of the cause or a fix for the problem. Thus, in this project, the DS1621 is powered by one of the PIC's I/O pins, and the PIC simply power-cycles the DS1621 about once an hour to reset it. The DS1621's supply current is only about one milliamp — well within the limits of the PIC's I/O pins.

The power supply is based on a very basic linear circuit fed by a wall wart with a 12 volt AC secondary. Optoisolator U5 provides a "power good" signal by tapping into the unregulated DC input to the 7805. During a blackout, the LED within the optoisolator turns off which causes the phototransistor output to stop conducting. The input of inverter U4A then goes high, sending an active-low signal to the PIC. The battery backup function to keep the autodialer working during a blackout is provided by four AA alkaline batteries and diodes D2, D3, and D5. D2 and D3 drop the nominal 6.2 volts provided by four fresh alkaline batteries down to about 4.75 volts. As long as the AC power is on, negligible current is drawn from the batteries since the 7805's five volt output is above the 4.75 volt output from the batteries. D5 prevents current from the batteries leaking backwards through the 7805 during a power failure and keeps the optoisolator's LED lit. The PIC and DS1621 work fine down to four volts or less, and capacitor C2 ensures that they won't notice the switch from AC to battery power.

The flooding sensor consists of two wires placed about a centimeter apart, where floodwaters will touch both of them and complete a high impedance circuit to the input of inverter U4B. The resistance of a centimeter of water is less than the one megohm pull-up resistor, so the input is pulled low. U4C then inverts the signal again to provide an active-low signal to the PIC. Unless the cable from the flooding sensor to the inverter input is very short (a few feet at most), it would be a good idea to build a separate circuit board for the sensor and a dedicated 74HC14 inverter chip.

Use one of the inverter's outputs to send the signal through the cable back to the PIC, since the low impedance output is a lot less sensitive to interference and glitches over a long cable run than a high impedance input would be. If necessary, multiple sensors can be installed wherever flooding may be encountered, with their inverter outputs paralleled in diode-OR fashion.

The telephone line interface I am using (**Figure 5**) is based on the circuit provided in the Parallax BASIC Stamp programming manual (**www.parallaxinc.com**). When relay RL1 is energized, it takes the phone "off hook," so a dial tone is generated. R11, R12, C3, and C4 filter the synthesized DTMF tones from the PIC so that the phone company's equipment will recognize them. The MOV across T1's primary provides some protection against high voltage spikes damaging the PIC, while D6 and D7 clamp voltages on the secondary at about 4.5 volts. The phone company is touchy about non-certified equipment on their lines, but while the relay is not energized there is absolutely nothing connected to the phone line.

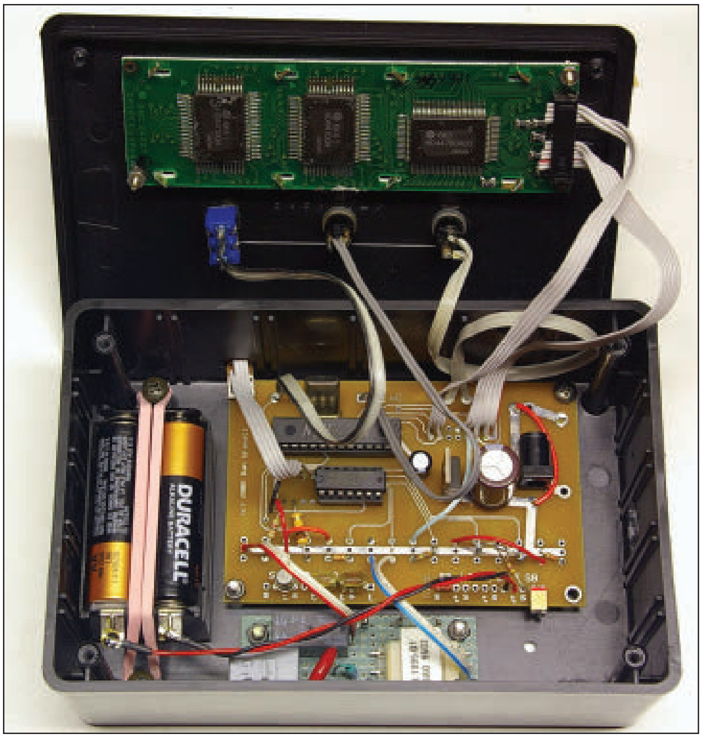


■ **FIGURE 5.** Telephone interface board.

CODING, CODING, CODING ...

The code for this project is extensive but fairly straightforward (see files autodialer.pbp and autodialer.hex at the article link). I have added numerous comments to explain the functions of the various sections of code, so please take a look. The first two subroutines allow the user to enter two phone numbers to be called to report alarms, and they

■ **FIGURE 6.** Autodialer interior layout.



automatically detect whether the number is local (seven-digit) or long-distance (1+area code+seven-digit).

Next, two subroutines allow the user to enter the low and high temperature alarm limits. After this is complete, the PIC begins monitoring the alarm inputs, and the user interface pushbuttons no longer function. If you need to change the user-entered data, simply press the reset button and re-enter it.

The alarm monitoring loop checks each alarm input sequentially, and if any of the external inputs are found to be low, the code waits 15 seconds and then checks again. This should minimize false alarms from glitches or brief "blips" in your electrical service during thunderstorms. If the alarm input is still low after 15 seconds, the PIC dials the first phone number and reports the appropriate alarm using a two-letter message in Morse code (such as "PF" for power failure). Even if you don't know Morse code, the pattern of each message is distinctive and fairly easy to recognize. I could have added a digital voice recorder chip instead, but decided against it due to the increased cost and complexity of the circuit.

Once the first call is complete, the PIC dials the second phone number and plays the same message. Finally, the code sets a flag to remember that this particular alarm has already been reported, so you don't get called again and again.

The code can report multiple alarms in series. For example, if your power fails in the middle of winter, first you will receive a PF alarm, and then a while later you will receive a LO alarm indicating that the house temperature has fallen below minimum. When power is eventually restored and your furnace warms up the house again, you will receive an AC message indicating that the alarms are cleared, but only after each of the individual alarms have actually cleared. I considered reporting each individual alarm as it cleared, but decided that if you have multiple alarms going off, just fixing one of them really doesn't count for much. When the AC message is sent, all alarm flags are cleared so that new alarms can be reported in the future.

BUILD IT!

The circuit can be built on any general-purpose board, but for the most accurate temperature sensing the DS1621 should be exposed to ambient air outside whatever project case you use. I recommend installing some sort of guard over the sensor to protect it, while letting air flow through.

PARTS LIST

DESCRIPTION	DIGI-KEY PART #
U1: PIC16F876A microcontroller	PIC16F876A-I/SP-ND
U2: DS1621 I²C Centigrade temperature sensor	DS1621+-ND
U3: LM7805 five volt linear regulator, TO-220 case	296-13996-5-ND
U4: 74HC14 hex Schmitt trigger inverter	497-1777-5-ND
U5: 4N25 optocoupler (*)	4N25MFS-ND
X1: 20 MHz crystal	631-1091-ND
Q1: 2N2222 transistor	PN2222BU-ND
R1-R5: 4.7K ohm resistor, 1/8 watt	4.7KEBK-ND
R6: 2.2K ohm resistor, 1/8 watt	2.2KEBK-ND
R7: 1M ohm resistor, 1/8 watt	1.0MEBK-ND
R8-R9: 10 ohm resistor, 1/2 watt	10H-ND
R10-R12: 1K ohm resistor, 1/8 watt	1.0KEBK-ND
D1-D5: 1N4004 diode	1N4004FSCT-ND
D6-D7: 1N5228 zener diode	1N5228BFSCT-ND
C1: 1,000 µF 35V electrolytic capacitor	493-1085-ND
C2: 33 µF 16V electrolytic capacitor	493-1038-ND
C3-C4: 0.1 µF capacitor	EF1104-ND
S1-S4: SPST momentary switch (*)	CKN9098-ND
T1: 1:1 isolation transformer, 600 ohm (*)	237-1121-ND
T2: 12 VAC 500 mA wall wart (*)	T1007-ND
RL1: DPDT relay, five volt coil (*)	255-1062-ND
MOV: 160V Sidactor surge suppressor (*)	P1300EAL-ND
LCD: Two line, 20 character, HD44780 controller (*)	73-1344-ND
BT1: Holder for four AA alkaline batteries (*)	BH24AAL-ND

(*) I used junk-box parts in my circuit; these part numbers should be equivalent, but are suggestions only.

Place the reset button somewhere it can't be easily pressed by curious hands. I mounted mine on the back side of the circuit board; I use a paper clip or pen point to press it through a pinhole in the case. Mount everything securely in the case with enough room for sensor cables, phone, and power connections (**Figure 6**).

Before installing the backup battery and diodes D2 and D3, build a small test circuit with a test load of about 200 ohms to verify that the voltage across the test load is indeed about 4.75V, since diode voltage drops do vary somewhat. You may need to adjust the number of diodes to tweak the voltage, and you'll definitely need three or more 1N4004s if you choose to use lithium AA batteries due to their higher cell voltage. Using a low power NAP command in the code loop reduces the current draw, while monitoring the alarm inputs to only 10 mA. This rises to about 15 mA while checking for glitches (using PAUSE instead of NAP), and will go up to 50 mA or more when the relay is energized.

The three extra inputs are coded to send alarms for a smoke alarm (SM), burglar alarm (BU), and carbon monoxide (CO). These alarm codes can be changed to match whatever inputs you desire, but you'll need the full version of PICBASIC PRO to recompile after any changes. It's up to you to figure out an interface to these devices which will generate an active-low input to the PIC; there are three unused inverters in U4 available for this purpose.

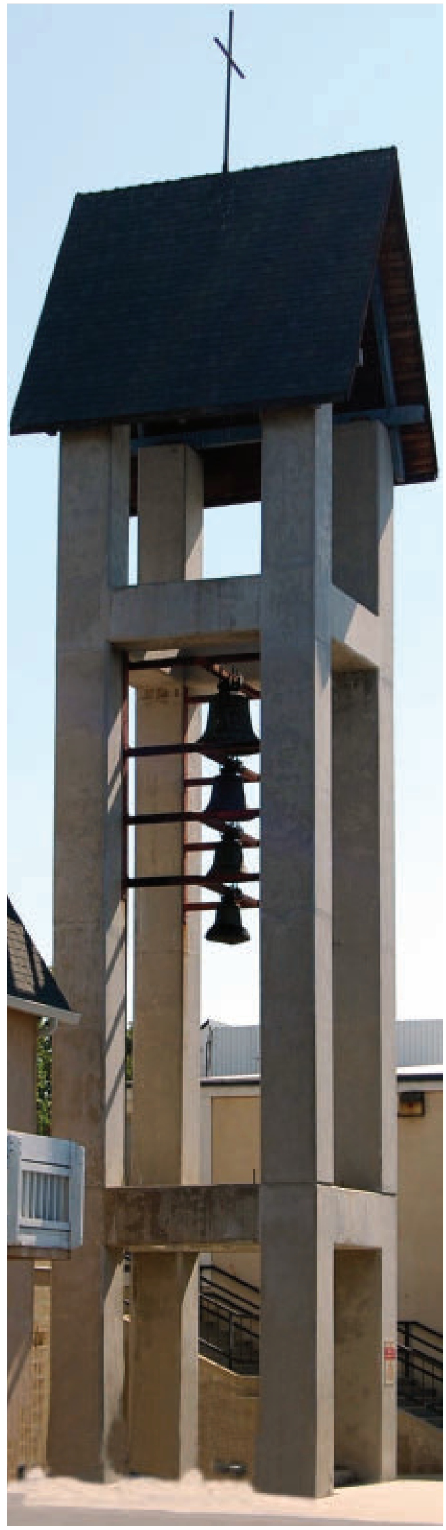
I **strongly** suggest that you purchase a smoke alarm or carbon monoxide alarm specifically to connect to the autodialer, rather than modifying ones you may already have installed in your home. This will ensure that any modifications you make to them won't compromise your existing alarms and put your family's safety at risk. That way, you can travel with peace of mind, knowing that if something happens back home you'll get the call. **NV**

Dan Gravatt can be reached at dgravatt@juno.com.

A Proper Ringing of the Bells

By Chris Watson

The art of engineering is to recognize and understand a problem, to conceive and design a solution, and to implement it in an appropriate way. I had the opportunity to practice this when church bells in the tower at St Mark's Episcopal Church (located in Upland, CA) chimed at 3:00 am in the morning, and continued chiming every hour on the hour thereafter. The following day, the neighbors were complaining. This was a problem, however, the short term solution was clear. The computer was unplugged from its power and the bells were silenced.


St. Mark's Bell Tower.

There are four bells at St. Mark's Church which are cast in bronze and are named after the four gospels: Matthew, Mark, Luke, and John. They are mounted one above the other in a free standing tower and when required, are struck by an electrically driven clapper. The clapper is excited by the closure of a relay that applies 24 volts to the clapper coil at the appropriate time and in the correct sequence. A controller with its computer and clock are programmed to chime as desired.

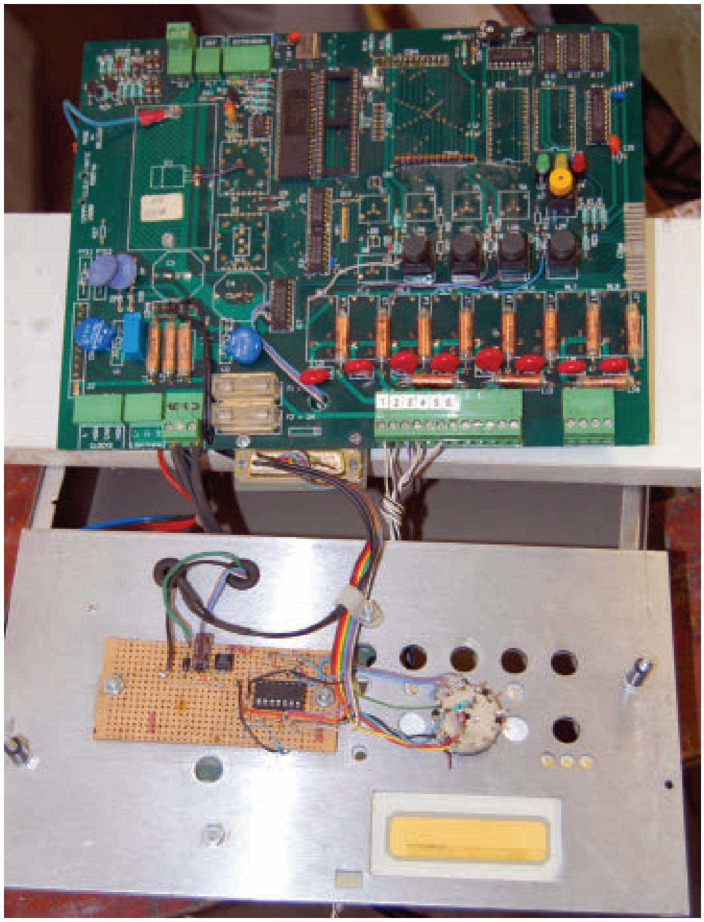
Simple, you say? Well, in the early 1980s when the bell system was installed, personal computers were in their infancy, and the friendly user interface had yet to arrive. We followed the programming instructions and inserted ones and zeros in some of the 76 locations which contained the program as part of our valiant attempt to make the chimes occur correctly on cue. Unfortunately, we failed. The controller needed a make-over.

We used a remote controller from the tower that was connected by only five wires. Connections were labeled and disconnected, and the unit was disassembled. The metal case, the circuit board, front panel, the power supply, the switches, and relays were all useful. Components no longer needed were discarded. We were ready to design.

We bought a sprinkler timer which contained an accurate battery supported clock. It could be

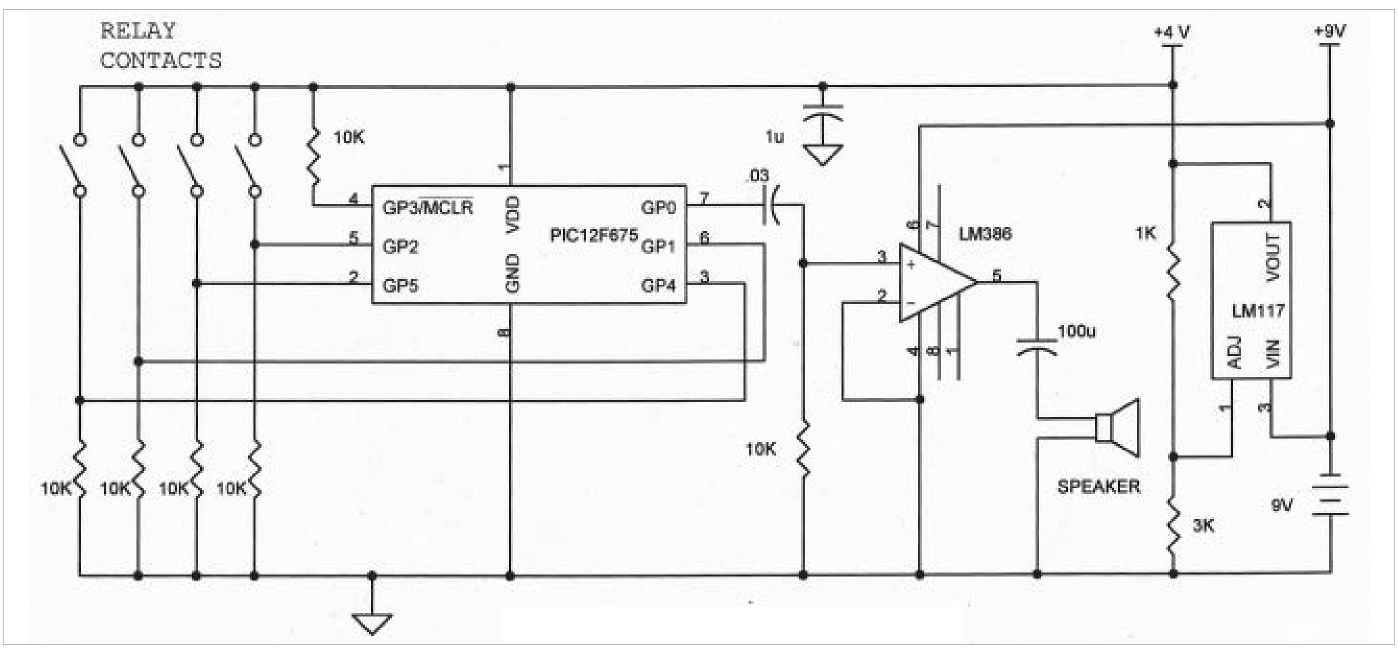
www.nutsvolts.com/index.php?/magazine/article/august2012_Watson

Discuss this article in the *Nuts & Volts* forums at http://forum.nutsvolts.com.



Original PCB and front panel, showing new parts and modifications.

The bells (Tintinnabulum !!)

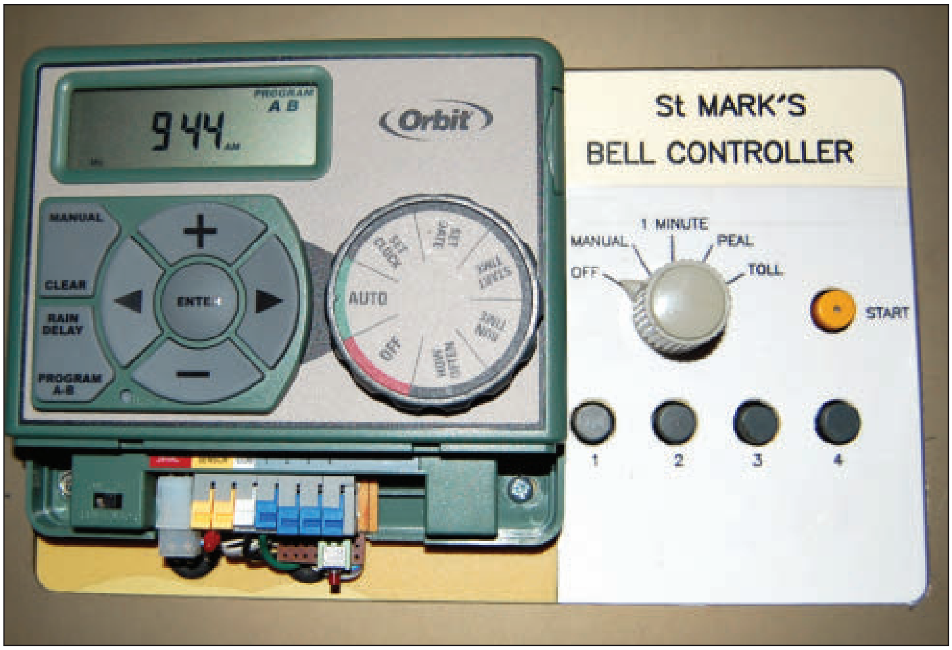


Model bell tower schematic. The circuit has four inputs switches. When any switch is closed, the circuit generates a burst of tone which simulates the sounding of the bell.

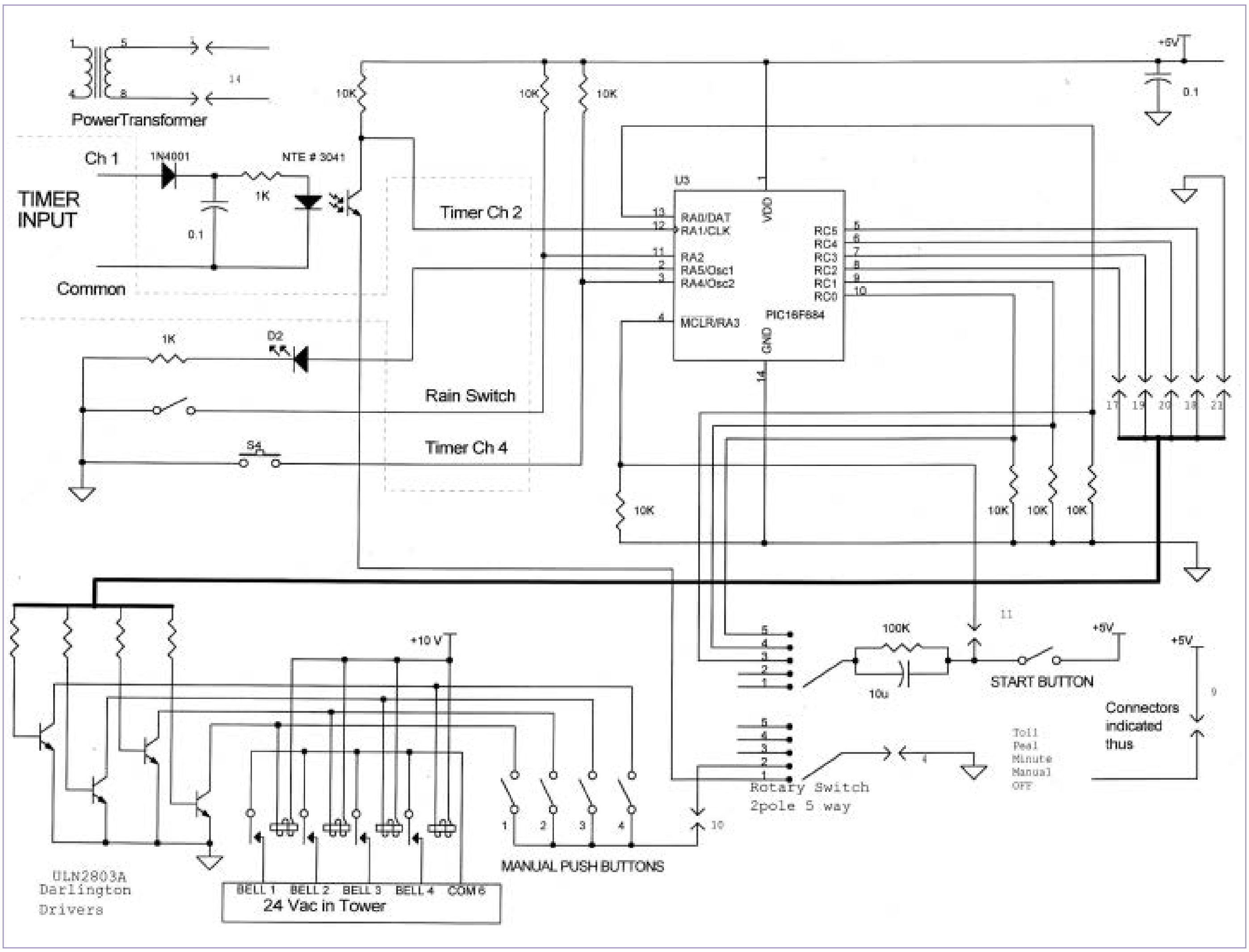
programmed to switch sprinklers on up to eight times a day, on the days of choice. Just what we needed! The plan was to use a PIC and custom electronics that would be programmed to accept input from the timer, and produce appropriate drive for the clapper relays.

The features we required were:

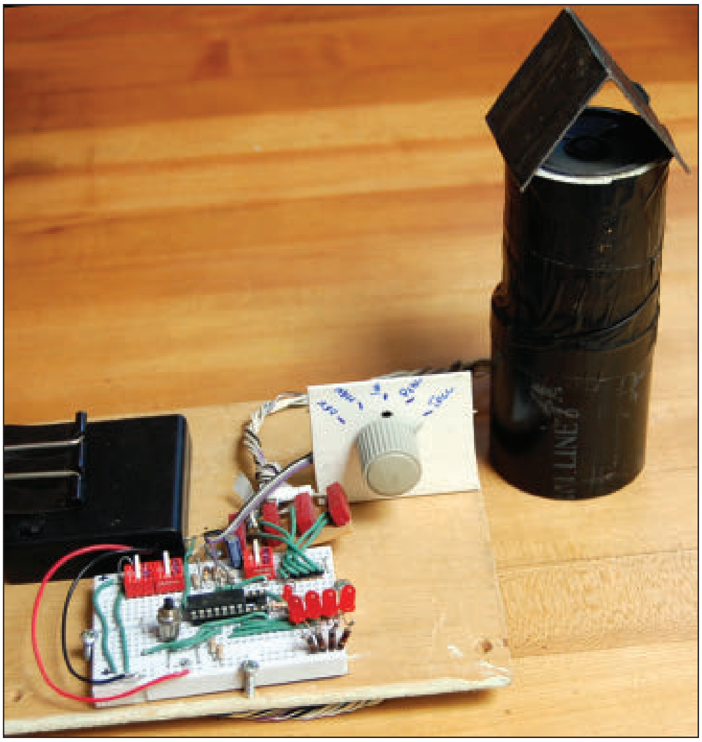
1) Westminster chime to play four times each weekday, only at the desired times.
2) Manual use as an option.
3) An urgent calling bell played prior to the start of a worship service.
4) A joyous peal that could be rung on occasions such as weddings.
5) A solemn toll that would be appropriate for funerals or times of mourning.



Final front panel of the new controller.



The complete system schematic. Timer input is from the sprinkler controller. The PIC is mounted on the board behind the panel, and other parts of the circuitry are on the original PCB.



Test-bed controller that simulates all system functions but lets you make changes and developments without annoying the neighbors !

The Westminster chime sequence is the tune played by Big Ben in London — a sequence of 16 chimes, followed by the tolling of the hour count. It requires four bells.

Our requirement was that the bells play the Westminster chimes at 9:00 am, noon, 3:00 pm, and 6:00 pm on Monday through Friday only. Our bells are A4, G4, F4, and C4 or similar, and the Westminster sequence is: *A, F, G, C, C, G, A, F, A, F, G, C, C, G, A, F.*

The mechanics of the construction were to mount an extra circuit board with additional parts behind the front panel, modify the wiring on the existing board to reconfigure needed components, and interconnect the boards with plugs and sockets to provide added reliability. The front panel folds over and mounts to the main circuit board.

We realized a test device would be needed to aid in developing the program for the bells, so a mini test tower was built which made bursts of sound with correct tones; there was no attempt to imitate the sonorous bell

```
' Bell Controller - Use  PIC16F684   -  Bells6.PBP
' May 30 2011
'
' Timer is ch1 triggers RA1 - active at 8.59am,
' 11.59am, 2.59pm, & 5.59pm
' Timer ch3 is connected to the Rain Switch and RA2
' Timer ch2 is modified to mount an output LED
' driven on RA5
' Timer ch4 is modified to mount a push button to
' ground on RA4
' Rain Switch and push button are used to program
' circuit variables.
'       Switch select RA0 - 1 MINUTE ring, RC1 -
'       PEAL, RC1 - TOLL
'       Switch select options are active when
START
'       button is pressed
'       output drive RC2 -  #1(A), RC5 -  #2(G),
'       RC4 -  #3(F), RC3 -  #4(C)
' Timing functions are performed by the
' programmable Orbit model 57874
' Trigger pulse occurs 1 minute before the hour
' There is delay of 37 sec so that 1st Bell chime
' occurs on the hour
' Westminster chime sequence A,F,G,C,C,G,A,F,A,
' F,G,C,C,G,A,F, N 'notes
   DATA @0,1,3,2,4,4,2,1,3,1,3,2,4,4,2,1,3, 9 'data
  CMCON0  = %00000111  ' turn comparators off
  ANSEL   = %00000000  ' all digital IO
  TRISA   = %00011111  ' Output 0,  Input 1
  TRISC   = %00000011  ' Output 0,  Input 1
  N       VAR    BYTE ' number of Dongs
  M       VAR    BYTE ' counter for Dongs
  Q       VAR    BYTE ' count notes in chimes
  Bell    VAR    BYTE ' bell number read from data
  T       VAR    BYTE ' TOLL counting variable
  P       VAR    WORD ' P mS interval between
                      ' hourly Dongs
  D       VAR    WORD ' D mS interval between
                         ' Westminster Chimes
  S       VAR    WORD ' S is Period between Peals

  PORTC.2=0 : PORTC.3=0 : PORTC.4=0 : PORTC.5=0 :
  PORTA.5 = 0
  PAUSE 500    ' wait half second to let power up
  READ 16,N    ' last value of N  ( 9, 12, 3, 6 )

Idle:   WHILE PORTA.3 = 0
        IF PORTA.1 = 0 THEN Ritime
        ' active only with Switch OFF
        IF PORTA.0 = 1 THEN Minute
        ' execute toll          pin 10
        IF PORTC.0 = 1 THEN Toll
        ' execute minute chime  pin 13
        IF PORTC.1 = 1 THEN Peal
        ' execute peal          pin  9
        IF PORTA.2 = 0 THEN Ready
        ' Rain switch ON to sync N
        WEND
        GOTO Idle

Toll:   FOR T = 1 TO 30         ' Toll #4 30 times
        PULSOUT PORTC.3,20000  ' 200 mS pulse
        PAUSE 1800             ' wait 1.8 Sec
        IF PORTA.3 = 1  THEN Idle
                               ' Button pressed to
                               ' stop
        NEXT T : GOTO Idle     ' normal end to toll

Minute: FOR T = 1 TO 60        ' Chime for 1 minute
        PULSOUT PORTC.2,20000 : PAUSE 800
                               ' 200 mS pulse
        IF PORTA.3 = 1  THEN Idle
                               ' Button pressed to
                               ' stop
        NEXT T : GOTO Idle

Peal:   S = 1500 : FOR T = 1 TO 60   'S = 1.5 Sec
```

chimes. The model tower used a PIC12F675 with inputs connected to the controller relay contacts. Each input triggered a burst of a square wave of a specific frequency but of fixed duration. All the signals were fed to an LM386 audio amplifier chip and fed to a speaker. The program was written in PICBasic.

The system was designed as follows: The signal given by the timer is 24 VAC and is intended to drive a sprinkler solenoid. We use this signal to initiate a single bell chiming sequence; it can be edge triggered if you like. The AC signal is rectified and fed to an opto-isolator, producing the trigger for the microcontroller.

It became apparent that since all the hour chimes are triggered by the same single sprinkler channel that the electronics have to remember how many hour chimes (or dongs) are played at each ringing. It is possible for an incorrect number of dongs to be played at the hour, but the ringing will occur on the correct hour. We must be able to adjust the number of dongs if for any reason the system gets out of sync.

To this end, a push switch, an indicating LED, and a *RAIN* switch were added, with additional software to make this correction if needed.

The **schematics** show the parts that were used from the original equipment, the new board and its components, the connections to the timer, and the switches and pushbutton controls.

SOUNDING OFF

In all development, as projects unfold, challenges arise and are met with diligence and cunning. This project was no exception. One problem was that once installed, the controller program could only be modified in a very public place.

The solution was to build a test bed controller to work with our tower model so that we could test any program modifications easily.

Typical adjustments that our community wanted were to change the duration of intervals between the chime of bells, and the rate at which successive chimes were played. In the future, someone may want to compose an alternate peal made from sequences of the 24 changes that four bells provide. This could be a testbed for such an exercise.

For now, our project is complete, the bells ring their joyous melodies, and the neighbors are not disturbed during the middle of the night. **NV**

```
        PULSOUT PORTC.2,20000 : PAUSE S      ' A
        PULSOUT PORTC.5,20000 : PAUSE S      ' G
        PULSOUT PORTC.4,20000 : PAUSE S      ' F
        PULSOUT PORTC.3,20000 : PAUSE S      ' C
        IF PORTA.3 = 1  THEN Idle     ' Button
                ' pressed to stop
        NEXT T  : GOTO Idle

Ritime: N = N+3 ' N cycles thru 9,12,3,6
        IF N = 15 THEN N = 3
                ' and only controls Dong number
        WRITE 16,N     ' store current value of N
        PORTA.5 = 1    ' Turn LED ON
        FOR M = 1 TO 74
                ' pause 37Sec to chime at hour
        TOGGLE PORTA.5 : PAUSE 250 : TOGGLE PORTA.5
        : PAUSE 250
        NEXT M : PORTA.5 = 0  ' Turn LED OFF
        D = 1500 : FOR Q = 0 TO 15
                                ' pause 1.5 sec
        READ Q,Bell    ' This section controls
        IF Bell = 1 THEN  GOSUB BellOne
                        ' the daytime hourly chimes.
        IF Bell = 2 THEN  GOSUB BellTwo
        IF Bell = 3 THEN  GOSUB BellThree
        IF Bell = 4 THEN  GOSUB BellFour
        NEXT Q

        PAUSE 1000
        FOR M = 1 TO N ' This bell is the Hour Chime
        PULSOUT  PORTC.3, 20000
                        ' output 200mS pulse
        PAUSE 1800     ' delay 1.8 S
        NEXT M
        PORTA.5 = 1    ' Flash LED at 1 sec rate
        FOR M = 1 TO 300      ' wait 5min - avoid
                              ' extra triggers
        TOGGLE PORTA.5 : PAUSE 700 : TOGGLE PORTA.5
        : PAUSE 300
        NEXT M : PORTA.5 = 0  ' Turn LED OFF
        GOTO Idle

BellOne:   PULSOUT PORTC.2,20000 : PAUSE D : RETURN
        '200mS pulse wait D mS
BellTwo:   PULSOUT PORTC.5,20000 : PAUSE D : RETURN
        '200mS pulse wait D mS
BellThree: PULSOUT PORTC.4,20000 : PAUSE D : RETURN
        '200mS pulse wait D mS
BellFour : PULSOUT PORTC.3,20000 : PAUSE D : RETURN
        '200mS pulse wait D mS

      'Rain switch ON  -  Start button increments N
Ready:  PAUSE 1000     ' wait a half second
        READ 16,N      ' last value of N  ( 9, 12,
                       ' 3, 6 )
        WHILE PORTA.2 = 0      ' Rain Switch is ON
        IF PORTA.3 = 1 THEN Getout
                       ' START BUTTON - increments N
        IF PORTA.4 = 0 THEN Disp
                       ' show N only - no increment
        WEND
        GOTO Idle
Getout: PAUSE 500 : N = N+3   ' increments N by 3
        IF N = 15 THEN N = 3  ' only 3,6,9, or 12
                              ' allowed
        WRITE 16,N            ' save new value of N
        WHILE PORTA.2 = 0     ' Wait till Rain
                              ' Switch is OFF
        WEND
Disp:   PORTA.5 = 0    :  FOR M = 1 TO N
        PULSOUT PORTA.5,50000 ' 500mS pulse to LED
                              ' indicates N count
        PAUSE 500      ' wait 500 mS
        NEXT M  : GOTO Idle

        END
```

Design a Customized Security System Using RobotBASIC

By John Blankenship and Samuel Mishal

Nowadays, home security systems are commonplace, but often have limited capabilities. If you could design your own system, it could meet your specific needs and perhaps even save you money. If your home could call your cell phone when a violation occurs, for example, you might be able to dispense with monthly monitoring fees. If you travel a lot, imagine the piece of mind you would have if you received an email every day telling you everything is secure.

Assume that in addition to the standard window and door sensors, your home has motion sensors mounted outside the house to give an early warning so the system can react *before* a break-in actually happens. When outside motion is detected, lamps and perhaps a radio could be turned on to give the impression that someone is home. Furthermore, the times of the outside movements could be logged and included in an email report.

Consider what you could do if your system could interface with a camera. Instead of just saying that the house is secure, the daily emails could actually include one or more pictures as proof that everything is okay. And, should an intrusion actually occur, the system could take a picture of the intruder and immediately email it to you.

Once the system has the ability to interface with a camera, imagine the added functionality you would have if it could be mounted on a motorized turret. You might even want the ability to point the camera from a remote laptop and request pictures in real time. If you need more flexibility, imagine mounting the camera on a mobile robot that can be controlled over the Internet from anywhere in the world.

Remember, the whole idea is to create a custom system that meets *your* needs. You may or may not want all these features, and you probably have additional requirements we haven't even considered. With all these complexities, you might be wondering how difficult it would be to implement such a robust design. The software for such a project — especially if it grows over time — could eventually become quite complex. For that reason, the language we choose for implementation should have strings, floating-point math, arrays, and a significant amount of memory which would be beyond the capabilities of most microcontrollers.

If we center the hardware portion of the project around a PC, we can take advantage of low cost web cams, and memory will certainly not be a problem. With Microsoft's constant pressure to make Windows more stable, though, many PC languages cannot even handle basic I/O operations any more, let alone our more advanced needs. Just capturing images from a web cam, for example, is difficult (if not impossible) to do in many languages, and support for Internet communication and the sending of emails is even less common.

FIGURE 1. Standard magnetic window and door switches are easy to interface.

Fortunately, there is a language that meets all our needs. RobotBASIC is very powerful with nearly 700 commands and functions. It can handle all standard I/O operations including serial, parallel, Bluetooth, and many USB devices (including direct support for USBmicro I/O boards). As an interpreter, RobotBASIC simplifies program development and provides for interactive debugging, yet finished programs can be compiled into stand-alone executables. RobotBASIC is easy to learn, and totally free for individuals, clubs, and schools.

RobotBASIC also has commands for capturing images from TWAIN compliant web cams, as well as a host of image processing functions for dealing with the images once they are captured. Internet communications (both TCP and UDP) are supported, as is the sending of email through an SMTP host. These features make RobotBASIC an ideal language for many control applications, including our security project.

Let's start by seeing how easy it is to implement the basic functionality of a security system. Once we have a basic framework established, you can add custom features to meet your particular needs and desires.

The first thing our system needs to do is obtain pertinent sensory information. One way to accomplish this is to utilize conventional sensors such as the magnetic window switches used with standard alarm systems as shown in **Figure 1**. Typically, such switches can be obtained in either N.O. (normally open) or N.C. (normally closed) varieties.

Standard switches are not your only option. Many alarm sensors — including the infrared motion detector and glass breakage detectors shown in **Figure 2** — are just as easy to interface. The setscrew connections (shown in the open unit) provide both N.O. and N.C. terminals that can be interfaced exactly like physical switches.

The output from switches and switch-like devices can be arranged in series and parallel to implement zones and easily interface to a computer input port as shown in **Figure 3**. For those new to interfacing, let's examine this **figure** in more detail.

Figure 3 shows details of two zones: one using N.O. switches and one using N.C. switches. As you can see, the N.O. devices (you can use as many as you like in each zone) are wired in parallel, while N.C. contacts are wired in series. The two pull-up resistors shown ensure that the associated port pin is normally high (a logical one) unless the state of the switches pulls the input to ground (a logical low). Notice that means that the N.O. pin will be HIGH unless any of the associated switches are activated, while the N.C. pin will be LOW until any of its switches are opened. A typical input port has eight inputs. The unused pins can be used for additional zones as indicated with Zone 3.

In order to turn on external devices, signals from the computer's output port must be sent to a control circuit. If

FIGURE 2. A WIDE VARIETY OF SENSORS ARE READILY AVAILABLE.

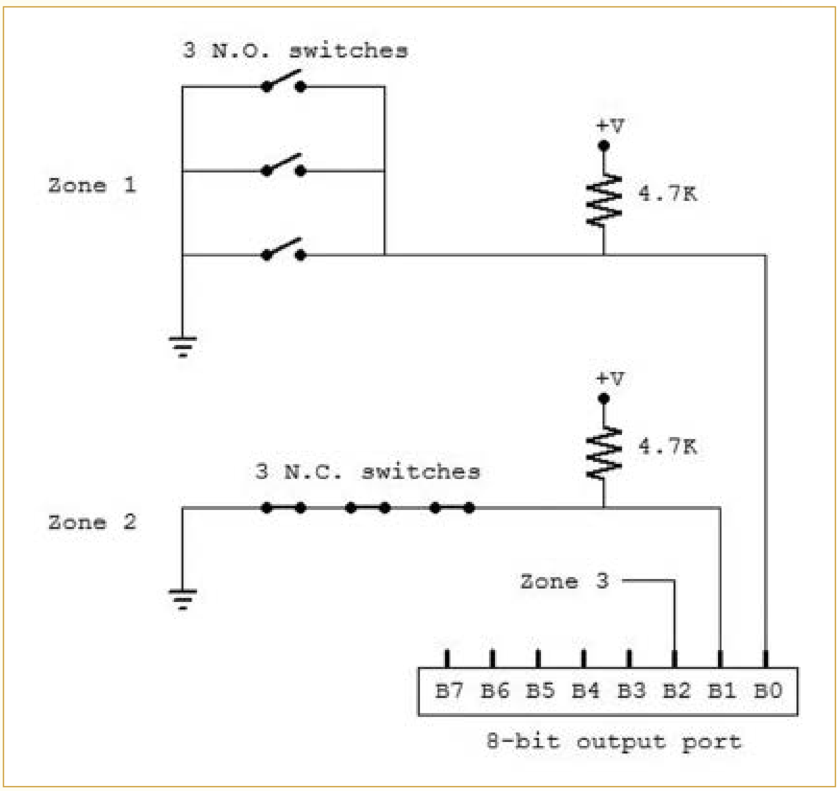


FIGURE 3. WINDOW AND DOOR SWITCHES ARE EASILY INTERFACED TO A COMPUTER.

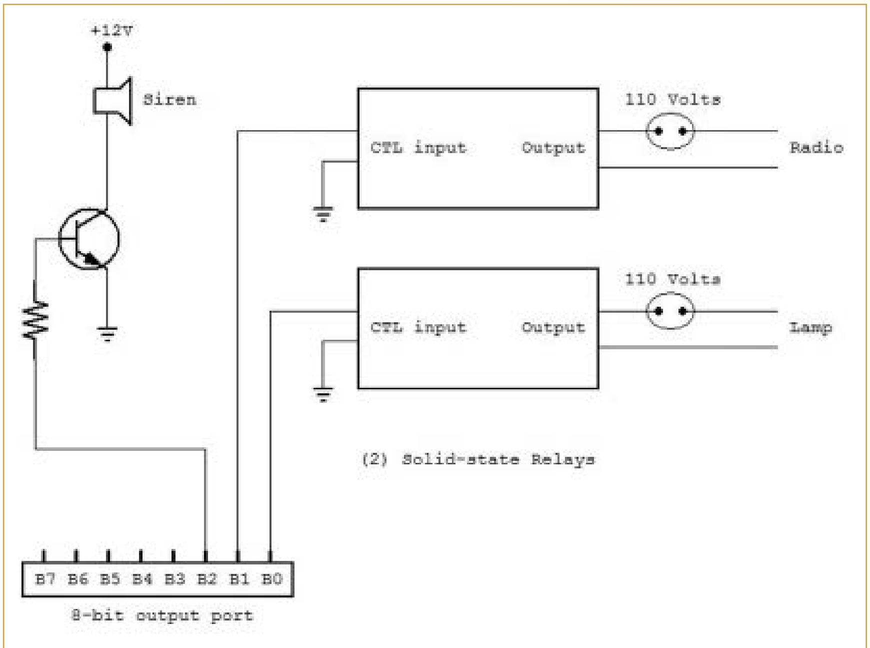


FIGURE 4. SOLID-STATE RELAYS MAKE IT EASY TO ACTIVATE EXTERNAL 110 VOLT DEVICES.

you are experienced with electronics, you could build such a circuit using transistors, an old-fashion relay, or — if you are controlling 110 volt devices — a TRIAC (or other thyristor). Fortunately nowadays, many companies offer ready-to-use solid-state relays so an in-depth knowledge of electronics is not required for many applications involving higher voltages.

Figure 4 shows how two solid-state relays can be used to activate 110V devices such as a lamp or radio. It also shows how low voltage devices (such as a 12 volt siren) can be interfaced with a general-purpose transistor. The relays are controlled with signals from the two least significant bits on a computer output port while the siren is activated with the third bit position. The remaining unused lines could be used to control additional devices as needed. In our examples, we will assume a logical one or HIGH from the computer will activate each device.

As you recall from **Figure 3**, our system has three zones of input sensors. To make things easy to follow, let's assume that Zone 1 is connected to all the doors in the house and Zone 2 is connected to all the windows. Furthermore, let's assume Zone 3 is an N.C. zone (like Zone 2) and is connected to some motion sensors mounted outside the home. We will also assume there is a lamp, radio, and siren connected as depicted by **Figure 4**.

Before we write any code, we must decide how we want our system to respond to external events. If an intruder is detected *outside* the home (Zone 3), for example, we might want the system to turn on the lamp and the radio as an attempt to scare them away. Since Zone 3 is connected to the outside sensors, our software must monitor the B2 input pin which is the third bit position from the right, and can be indicated by the one in the binary number 100 (decimal 4).

Before we can analyze the sensory data, we must actually read it from the input port. For purposes of this article, we will assume we have a parallel input port installed on our machine at address 2000 (arbitrary selection) which means that the following subroutine could read the input data into the variable **SensorData**:

```
CheckPort:
  InPort 2000,SensorData
return
```

It is important to realize that this is only one of many options. RobotBASIC has other commands that allow you to read and write data through standard and enhanced printer ports or from many commercial parallel ports that are interfaced to your computer through a bus card, a serial port, or a USB cable.

If we assume our output port is located at address 2001, we can use the **OutPort** command to turn on various devices as shown in these examples:

```
// either of these lines will turn on the radio
OutPort 2001, 2      // use decimal 2
OutPort 2001,0%10    // use binary 2

OutPort 2001, 1      // turns on the lamp

OutPort 2001, 0%110 // turns on siren and radio
```

If we can determine which zone has been violated, we can make our system react appropriately. For example, let's assume that we want any movement outside the house (Zone 3) to turn on the radio (perhaps tuned to some talk channel that will sound like people talking inside the home) and the lamp (as if the residents got up to see what was happening). Remember, since N.C. sensors are used in this zone, the input pin will normally be LOW and go HIGH when an intruder is detected. The following subroutine monitors Zone 3 and reacts as discussed:

```
CheckOutside:
  if (SensorData bAND 0%100)    // check Zone 3
     OutPort 2001,0%11 // turn on radio and lamp
  endif
return
```

Notice that the IF statement does a binary AND with a *mask* of 0%100 (we could have used 4). This masking effectively erases all bits in the data (for purposes of the comparison) except where there is a 1 in the mask. This is necessary so that the state of the other zones is ignored in this decision.

Let's see how we might use the siren shown in **Figure 4**. The previous code tries to scare away intruders with the light and radio, but, if that fails, the system can turn on the siren (and perhaps the light) if a window is opened. This can be accomplished with the following code:

```
CheckWindows:
  if (SensorData bAND 0%10)    // check Zone 2
    OutPort 2001,0&101 // turn on siren and lamp
  endif
return
```

Notice we did not check the doors (Zone 1) in the above subroutine because we will want to give residents a short time to enter the home and disable the alarm. In order to keep things simple, we will let the user disable the alarm by pressing any mouse button for at least one second. The following code shows a simple way of implementing this feature in RobotBASIC. Notice that since the sensors in this zone are N.O., this zone provides a LOW in its bit position when it is violated.

```
CheckDoors:
  if (SensorData bAND 1) = 0  // check Zone 1
                              // for a low input
      // wait 15 seconds or until the mouse
      // is pressed
      for i = 1 to 15
          Delay 1000          // wait one second
          ReadMouse x,y,b     // b will hold the
                              // button data
        if b<>0 then break   // exit the FOR-LOOP
                             // early
        // notice the use of a single line
        // IF-THEN here
      next
      // if b is still zero, then no one pressed
      // the mouse
      if b=0 then OutPort 2001,5  // turn on
                                  // siren and
                                  // light
  endif
return
```

Using the mouse to turn off the alarm could be an acceptable option, but most people would probably prefer to enter a code to turn the alarm on and off, as is done in commercial security systems. This is actually very easy to implement using RobotBASIC's event-driven capabilities which allow the creation of a subroutine that is executed whenever a keystroke (or other event) is detected.

If you would prefer to have a remote keypad like those on commercial alarm systems, that too is easy since RobotBASIC supports serial ports — both real and virtual. Just program a small microcontroller (such as a Parallax BASIC Stamp) to read information from a keypad, and transfer that information to the PC using a serial connection.

Of course, after you implement all of the necessary subroutines, they must be placed in a loop that calls them properly, as shown in the next example. The code in the loop is simple and easy to follow because keystroke entries can be handled in the background (simply change the value of variables such as **AlarmOn**). The **ResetOutputs** routine in this code will automatically turn off all devices (lamp, radio, and siren) after a preset amount of time so that your neighbors will not be disturbed if you are unable to return to your home in a timely manner.

```
MainProgram:
  gosub Init
  while True // forever
    if AlarmOn
      gosub CheckDoors
      gosub CheckOutside
      gosub CheckWindows
      gosub ResetOutputs
    endif
  wend
end
```

These are all simplified examples, but hopefully you get the idea of how easy it is to create a usable system with a relatively small program. It is important to realize though, that as you add features and functionality, that your projects can become as complex as you desire.

Since RobotBASIC has commands for sending emails, you can add this feature by simply inserting the appropriate email code inside the proper IF block. Implementing a web cam is easy, too. After RobotBASIC captures two consecutive images, image-processing commands can be used to compare the two pictures to detect motion. This makes it easy for your system to detect when an intruder is within view of the web cam and then send the picture as an email attachment.

If you really want to get creative, you could make your system interactive. With RobotBASIC's ability to communicate over the Internet, you could use your laptop to take control of your home from anywhere in the world. This gives you many options. If you mount your web cam on a servomotor-controlled turret, for example, you could use a remote mouse to point the local web cam to obtain a panoramic view of your home (or alternatively, use RobotBASIC's internal robot protocol and a Bluetooth connection to control a mobile robot equipped with a wireless web cam to see anywhere you like).

RobotBASIC offers many features for implementing complex PC and multi-processor based projects. These include support for all standard I/O operations including USB and Bluetooth connectivity, as well as Internet communication, web cam support, and much more. Download your free copy (including extensive documentation) at **www. RobotBASIC.com**.

FIGURE 5. COMMERCIAL PHONE DIALERS SUCH AS THIS ONE MAKE IT EASY FOR YOUR HOME TO CALL YOU AND YOUR NEIGHBORS WHEN AN INTRUDER HAS BEEN DETECTED.

Even if you are just a novice, don't think that these features are out of your reach. Controlling servomotors, for example, can be handled easily in RobotBASIC by using a USB-based servomotor controller. Once you set up the controller as a virtual serial port, use RobotBASIC's **SerOut** command to send commands to the controller with the desired positions for each servomotor, and that's it. The controller will continue sending the necessary pulses to hold each motor's position until commanded differently.

Since it is easy for RobotBASIC to communicate with microcontrollers, you can incorporate them in your system in a variety of ways. Suppose you want your intelligent security system to monitor the windows in a detached garage or outbuilding, and you don't want the hassle of running wires between the house and the remote structure.

One easy solution would be to program a microcontroller to monitor the garage windows and transmit the information to the system PC using a Bluetooth link. Once you interface a serial Bluetooth transceiver with the microcontroller and a USB Bluetooth transceiver with the PC, RobotBASIC has commands to handle all the communication.

Even with all these improvements, our intelligent house still has many potential options. We could interface photoresistors and perhaps thermistors in appropriate rooms so the system can utilize that data, as well as clock and calendar information to determine when drapes should be opened or closed (in order to minimize air conditioning costs in the summer or to make an empty home appear to be occupied).

Remember, the cost of building a custom security system does not have to be an issue since you have the option of dispensing with standard monthly monitoring fees by interfacing with a commercial phone dialer, such as the one shown in **Figure 5**.

To help you get started, the complete code for a basic alarm system can be downloaded from **www.RobotBASIC.com**. To make it easy to test your ideas, the program includes a working simulation (see **Figure 6** for a sample output screen) that allows you to see how the system responds to sensory data (you can trigger different zones by clicking them with the mouse). Such a simulation can greatly simplify the development of your own customized system.

One point should now be clear. If you are truly interested in designing a custom security system or even a fully intelligent house, there is nothing standing in your way. Remember, you don't have to do everything at once. After you get a basic system operational, you will often find that software enhancements alone can boost its functionality. If that's not enough, just add additional hardware over time, and expand and enhance your software to implement the features you desire.

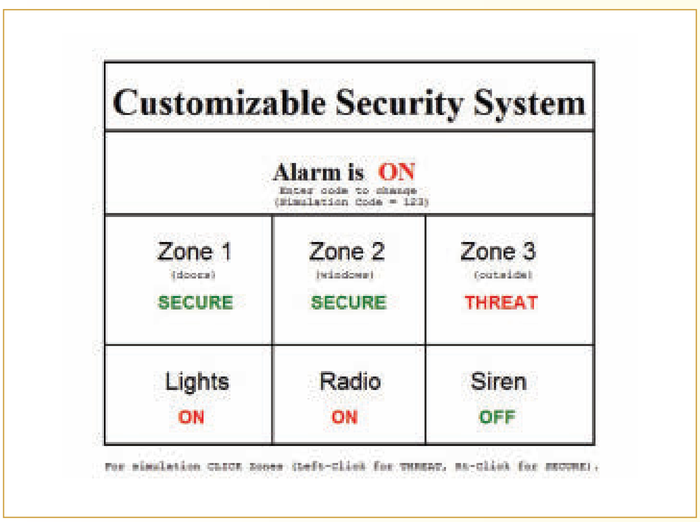


FIGURE 6. THIS BASIC SECURITY SYSTEM INCLUDES A SIMULATION THAT MAKES IT EASY TO TEST YOUR IDEAS.

SUMMARY

Implementing a custom security system for your home can be far less intimidating than you might think, especially with the right software. This article guides you through the basic principles you need to get started before discussing enhancements that can turn a simple alarm system into an intelligent house. **NV**

#49 SMILEY'S WORKSHOP

C PROGRAMMING - HARDWARE - PROJECTS

by Joe Pardue

Discuss this article in the *Nuts & Volts* forums at **http://forum.nutsvolts.com.**

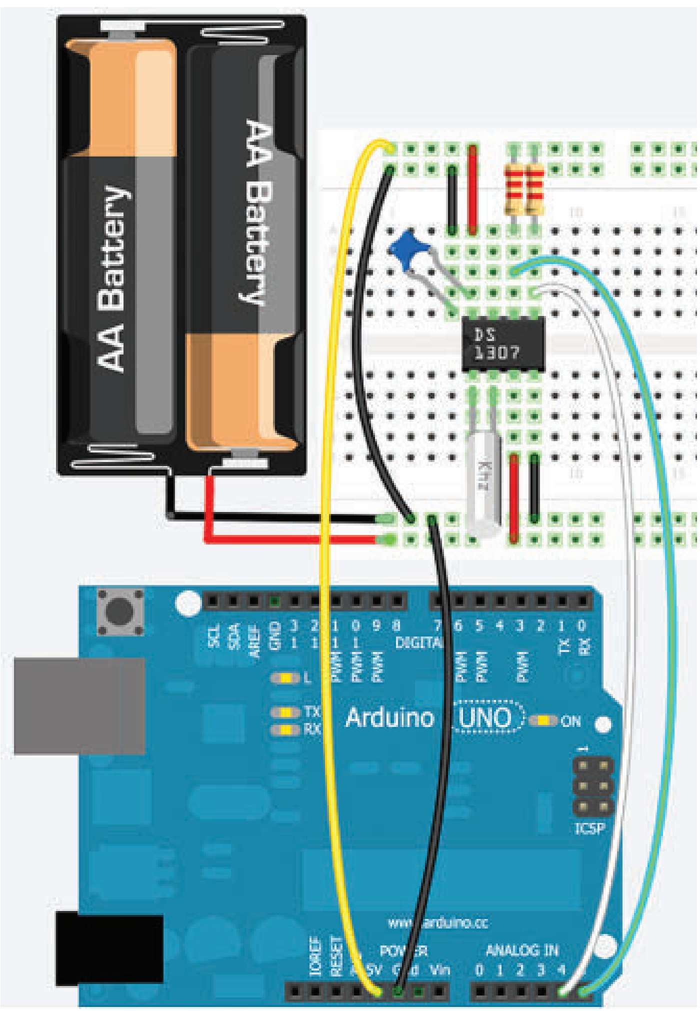
Fritzing With the Arduino — Part 1

Recap

Last month in the C theory section, we looked at C programming concepts for enums and data structures. In the lab section, we got started with the DS1307 real time clock (RTC) IC. Then, right in the midst of exploring these concepts I stumbled across Fritzing — an open source application that attempts to do for hardware design (electronics computer-aided design) what the Arduino did for software. Fritzing is a very novice-friendly open software package that you can use to design shields for Arduinos. Even though it is still in beta, it has many very interesting and useful features. So, I decided to put our study of time on the shelf and do two workshops on Fritzing, then come back to how C programmers deal with time later.

Fritzing is being developed by the Interaction Design Lab of the University of Applied Sciences Potsdam, Germany. It is open source, so there are contributors from all over the world. You can get more information from **www.fritzing.org**.

Since this is breadboard centric, you may want to take a look at my blog post on breadboards at **http://smileymicros.com/blog/2012/06/09/introduction-to-breadboards-and-schematics**.



■ FIGURE 1. Real time clock all wired up.

Fritzing

I am constantly reminded about the great variety of skill levels present in readers of my Workshops. Some folks can program circles around me, but wouldn't know which end of a soldering iron to use if their life depended on it. Others can hand-solder surface-mount parts on PCBs of their own design but can barely use an Internet browser, much less write a program. Most readers are somewhere in between having enough hardware and software knowledge to be dangerous, but still learning both. I've been slowly introducing the C programming language software while also applying these concepts to microcontroller hardware projects so that folks from both ends of the spectrum can get some incentive to learn the parts they don't already know.

This month, I'm going to cater a bit to the hardware novice by introducing Fritzing, which can help a beginner get started with hardware design.

First, we will use Fritzing to design a DS1307 RTC circuit we were introduced to in the May '12 Workshop. In this first installment, we will follow **Figure 1** to build a circuit on a breadboard that we can use with some Arduino code, then we will convert the breadboard design to a schematic. In Part 2, we will see how to use the breadboard/schematic designs to generate a PCB layout that we can use to have a circuit board fabricated. Essentially, you'll see the full hardware design cycle from the breadboard prototype to the finished product.

What is Fritzing?

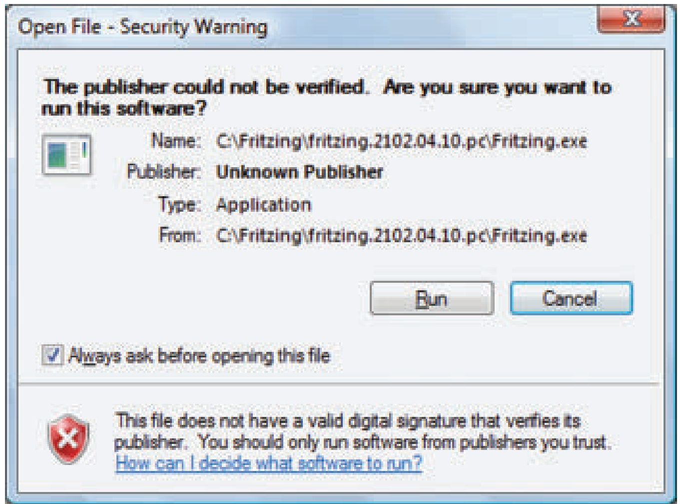
You can get the Fritzing application from **www.fritzing.org**. They introduce themselves as follows:

Fritzing is an open source initiative to support designers, artists, researchers, and hobbyists to work creatively with interactive electronics. We are creating a software and website in the spirit of processing and Arduino, developing a tool that allows users to document their prototypes, share them with others, teach electronics in a classroom, and to create a PCB layout for professional manufacturing.

I downloaded the 0.74 b April 10, 2012 version and used it in the illustrations here. It is 18.6 MB which isn't terribly large nowadays, but you'll probably want a broadband connection to download it. Do note that Fritzing is in beta release and this isn't even version 1.0 yet; since it is an open source project, it might just stay in beta for a while. The creators are very aware that Fritzing is a work in progress and you should expect some problems, but remember the price is free. If you like this tool and want to see it improved, you could send them a donation at **http://fritzing.org/shop/donations**.

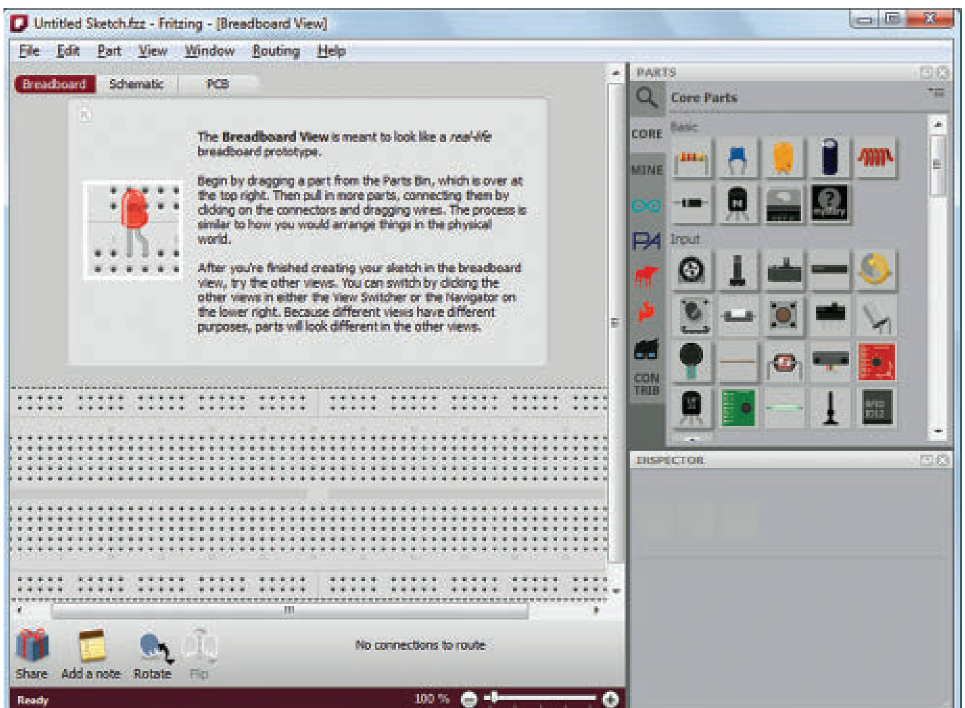
When you run the program, the first thing you see (shown in **Figure 2**) — as is so often the case with Microsoft — is a completely useless security warning that — trust me — you can ignore.

After dismissing the stupid security warning, notice the Fritzing splash screen shown in **Figure 3**. On the left, you see an Arduino connected to a

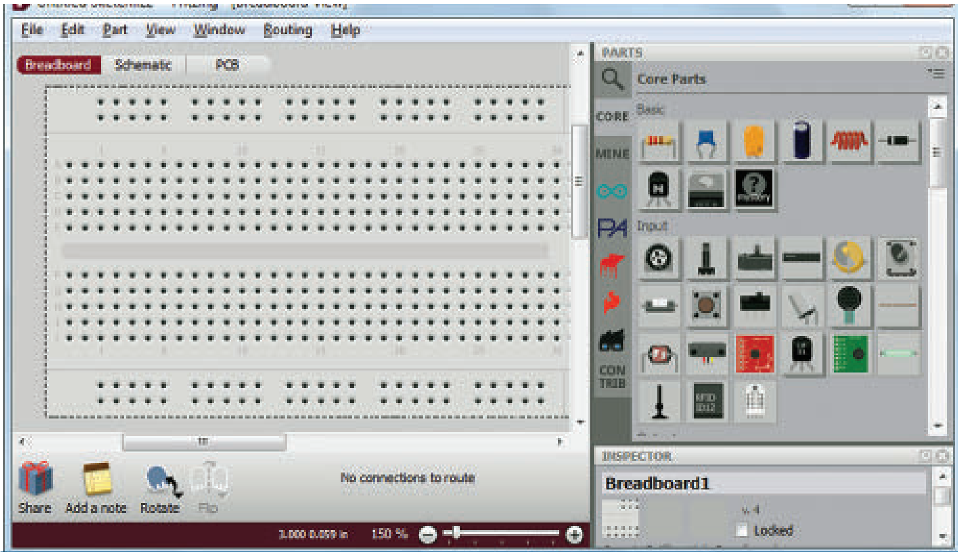


■ FIGURE 2. Security warning.

■ FIGURE 3. Fritzing splash screen.



■ FIGURE 4. Fritzing breadboard view.

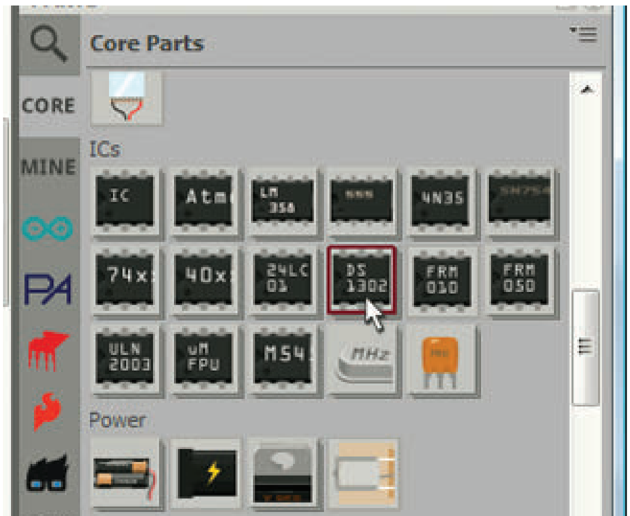


■ FIGURE 5. Resize.

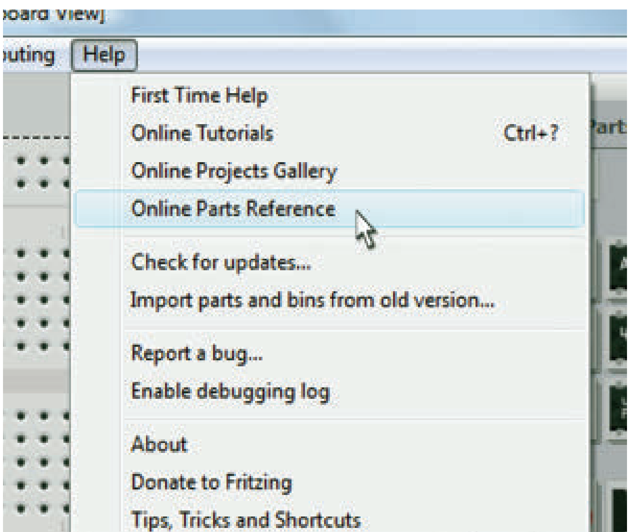
breadboard; in the middle, you see a computer with the design in Fritzing connected; and then on the right, there's a replicator that is outputting tiny flying submarines. This cute drawing pretty much covers what the folks at Fritzing think they are about and although they have a long way to go, they at least are making a light-hearted first step in the right direction.

When I first opened Fritzing, it came up in the breadboard view shown in **Figure 4**. Let's close the breadboard welcome and move the breadboard up and resize the whole window as shown in **Figure 5**.

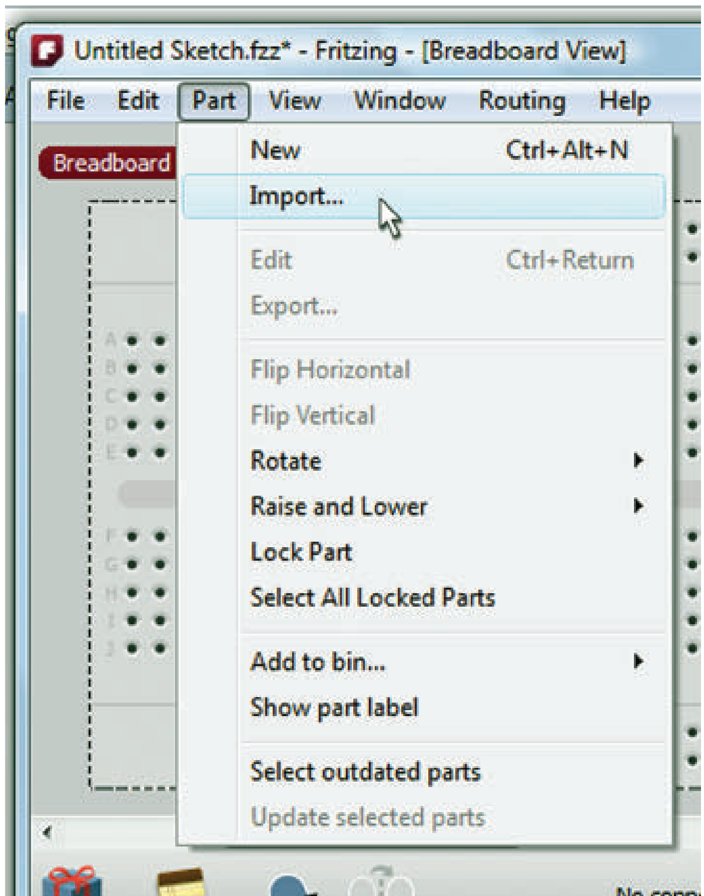
Now, in the parts window open the core parts and scroll down to the ICs. Hmmm ... they have a DS1302 (as shown in **Figure 6**), but we are using the DS1307 for our RTC. Do they have the same pinout? I did some Internet searching for the datasheets and discovered the DS1302 and DS1307 aren't the same. So, let's see if there is any help for this problem. [NOTE: I found a better way to get the DS1307



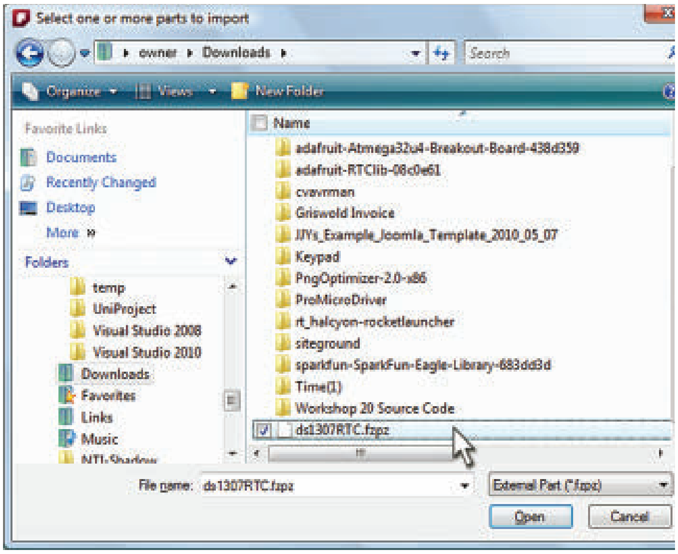
■ FIGURE 6. Core ICs DS1302.



■ FIGURE 7. Online Parts Reference.

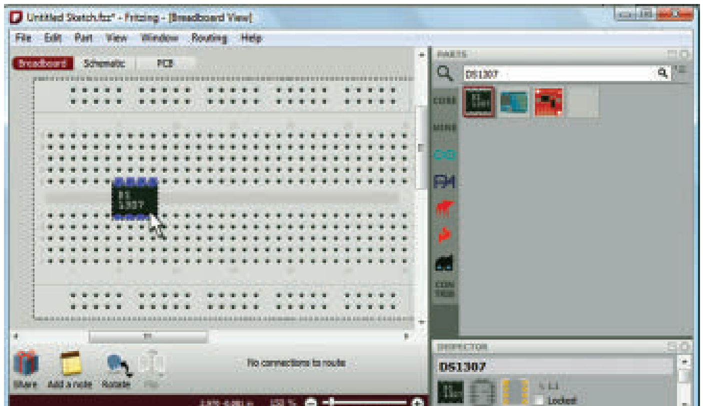


■ FIGURE 8. Import.



■ FIGURE 9. Select the part to import.

■ FIGURE 10. Parts search for DS1307.



■ FIGURE 11. Drag it to the breadboard.

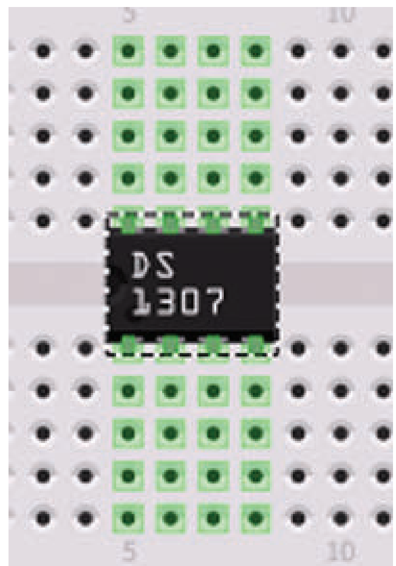
and will discuss it later in the article. For now, just think of this section as a generic discussion on how to import a part.] **Figure 7** show the link to Fritzing's Online Parts Reference.

That led to a Fritzing web page that has a link to the Fritzing parts repository that led to the DS1307RTC.fzpz that I downloaded. In **Figure 8**, I select the import item in the Fritzing Part menu. Then, in the directory browser window shown in **Figure 9**, I select the part to import.

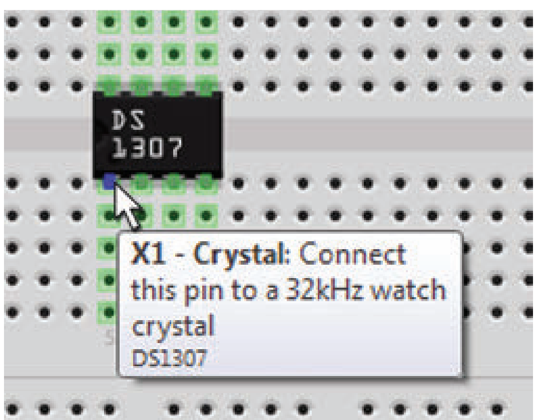
Next, I searched for the DS1307 using the (magnifying glass denoted) parts search facility shown in **Figure 10**. Then, I clicked on the part and dragged and dropped it on the breadboard as shown in **Figure 11**. Notice how the part locks onto the holes, putting the DS1307 pins into them. When you release the part, the connections are shown in green (see **Figure 12**).

What isn't clearly shown is which pin is pin 1. **Figure 13** shows what happens if we let our cursor hover over a pin. We see that this is the X1 pin, and the next pin is X2. By referring to **Figure 14** which shows an illustration from the DS1307 datasheet, we see that these are pins number 1 and 2. We can tell this because the drawing has a circle divot on the IC to indicate the end with pin 1, and that pin is always on the left in the depicted orientation.

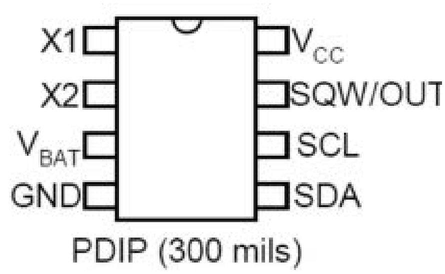
Next, we consult the datasheet illustration of a typical operating circuit (as shown in **Figure 15**) to decide what other parts are needed and how we will wire them up. Let's be honest here. We selected the DS1307 in the first place because there are a bunch of circuits on the Internet that show how other folks have done this design, so we will be highly influenced by those observations. [One thing to note is that I used a 0.1 µF bypass capacitor that isn't shown in a typical circuit. I do this because I've had enough experience to know that these help prevent digital glitches. Although we might get by without it, they are cheap and



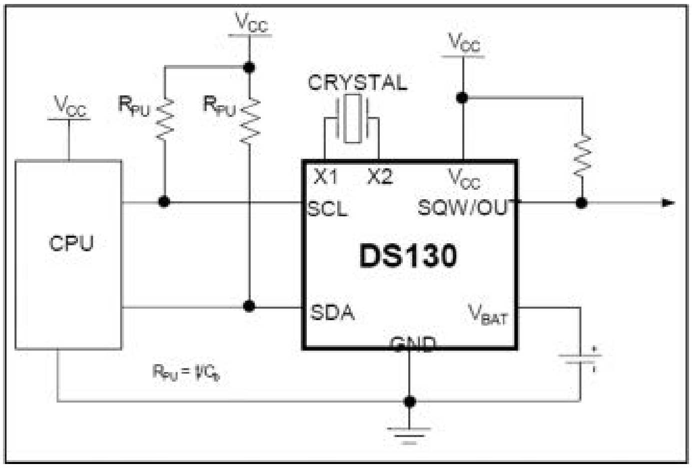
■ FIGURE 12. Connections shown in green.



■ FIGURE 13. Identifying pin numbers.



■ FIGURE 14. DS1307 pins in datasheet.

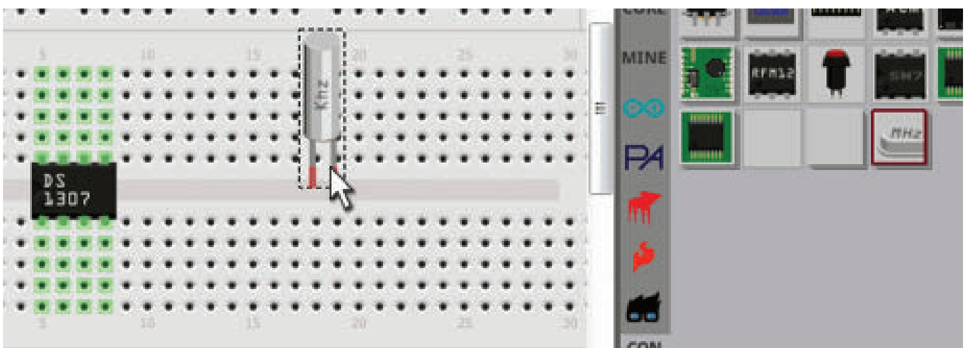


■ FIGURE 15. Typical operating circuit.

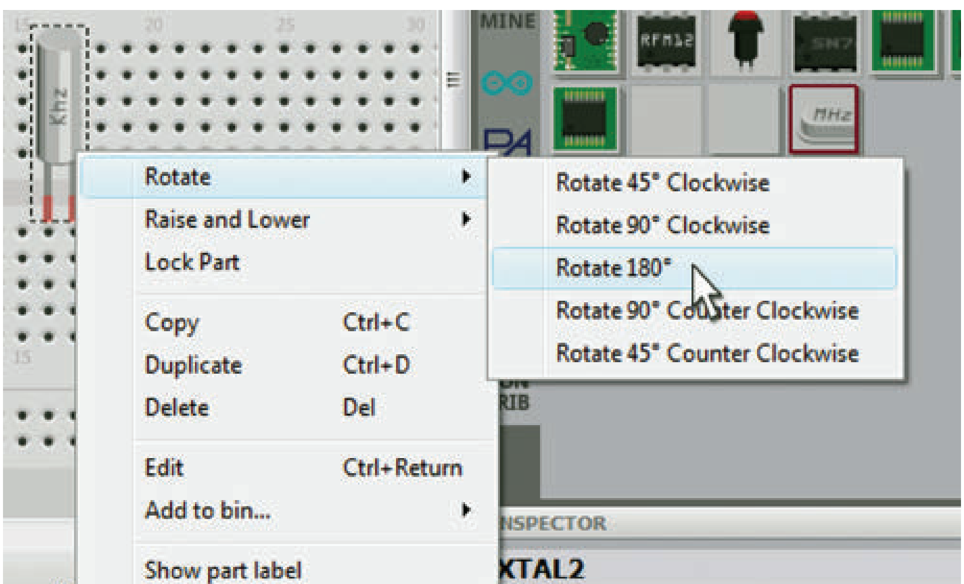
have proven necessary often enough in the past that I just throw one in for every digital IC and sleep better.]

So, based on **Figure 15** and the datasheet specifying the crystal, the next thing I wanted was a 32.768 kHz watch crystal to use on the breadboard. So, I imported "CRYSTAL – kHz.fzpz" as before and it almost killed my Fritzing session! I searched a bit further and found a more recent version of that file and it worked just fine. Lesson: You might have to mess around a bit to find these parts, and they might not work. (This is just one of the joys of open source design.)

Now, notice in **Figure 16** that the part selector box shows 'MHz' which probably means the author of this part reused an image (since the price is right, we won't fault him). The part itself looks like the watch crystal we will be using, so we drag it over to the DS1307 and drop it onto pins 1 and 2. Unfortunately it covers the IC and blocks most of the breadboard connections for pins 7 and 8, so let's flip the part around 180º as shown in **Figure 17**.



■ FIGURE 16. Select and move the crystal.



■ FIGURE 17. Rotate the crystal.

The Inspector Window

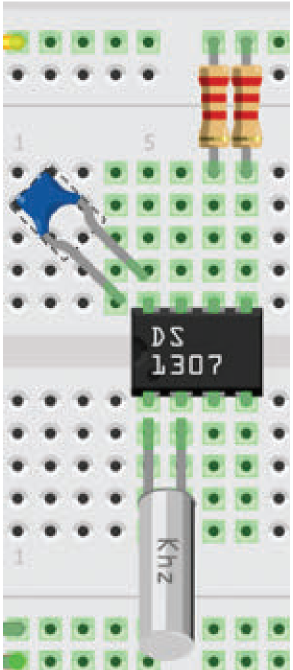
Boy howdy! Am I embarrassed! I was writing this tutorial on the fly while learning to use Fritzing, and after all that rigamarole about how to import a DS1307 since they only have a DS1302, I find I was wrong, wrong, wrong! The DS1302 part has a DS1307 variant that you can select in the Inspector window. So, rather than rewrite the whole article and since knowing how to import files is a good thing, I'll just leave all this import stuff in and tell you how to do it the right way.

You select a DS1302 from the parts bin and drop it on the breadboard, then (as shown in **Figure 18a**) you open the 'variant' dropbox and click on DS1307. Voilà! You have the correct part. We repeat this pattern when we add the resistors (shown in **Figure 18b** rotated as with the crystal). The default is 220 ohms, but you can click on the resistance dropbox in the resistor's Inspector window and select 2.2K, which changes both the value and the color bands.

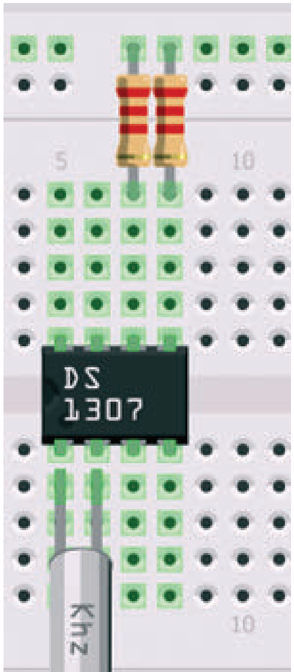
If an exact part you want to use is not in the parts bin, you may find a similar part can be modified in the Inspector window to match your needs. If the exact part you need just isn't available, the folks at Fritzing will create a part for you (for a fee, of course).

Back to the Breadboard

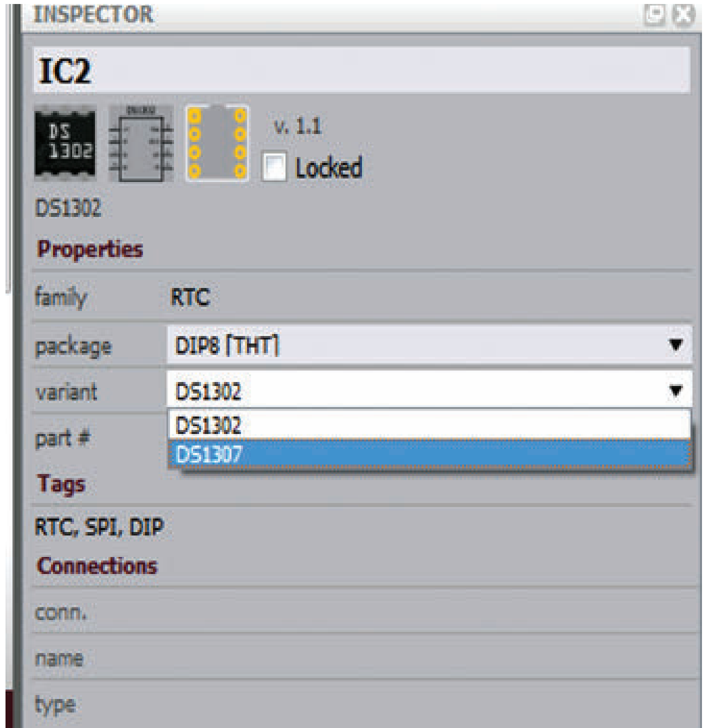
We connected the resistors between pins 7 and 8, the upper breadboard row that we will need to connect to VCC, but let's get all



■ FIGURE 19. Add a bypass capacitor.



■ FIGURE 18B. Add resistors.



■ FIGURE 18A. Inspector window.

the parts down before we flip the power switch.

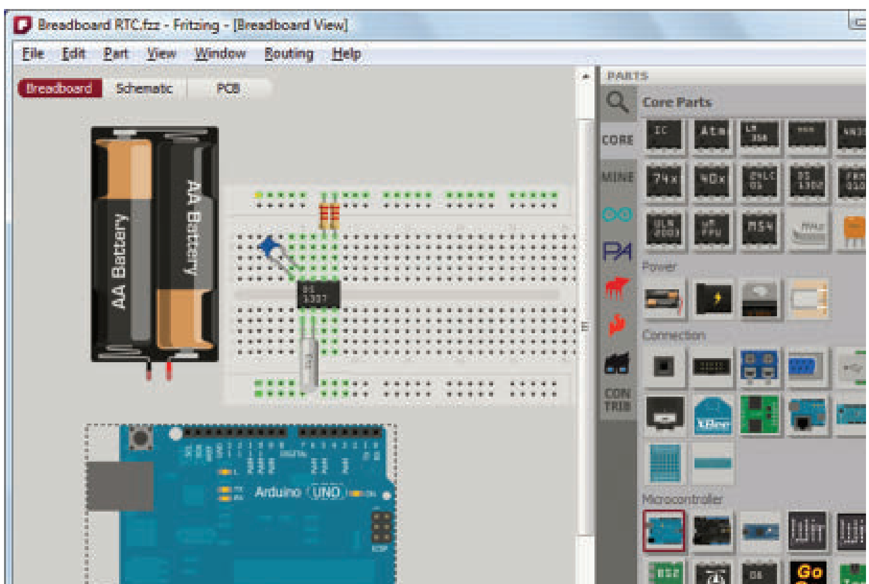
We add a 0.1 µF bypass capacitor as shown in **Figure 19**. Note that I chose to rotate the part by 45º so that it wouldn't block the pins that we will need to attach to VCC and GND.

Now that we have all the parts on the board, we will want to add the wires. Before we do that, we will want to add an Arduino that will be reading the date and time from the DS1307, and will be providing the main power. We will also want to add a three volt battery for backup. While we almost certainly would be using a coin cell for the backup battery in a 'real' design, for this drawing let's just use a battery box with two AA batteries since we have on at hand. [Okay, *I* have one at hand.] **NOTE: DON'T BE TEMPTED TO SOLDER WIRES TO A COIN CELL BATTERY TO USE WITH A BREADBOARD – IT MIGHT BLOW UP!**

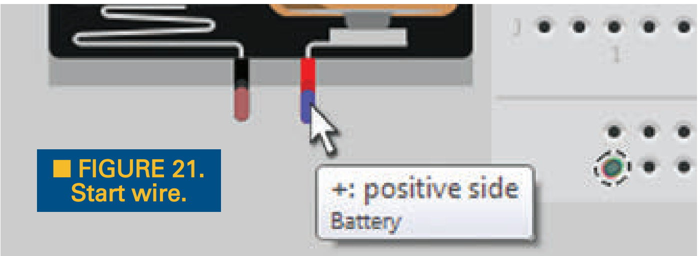
Next, you will want to zoom out to make room for an Arduino and a battery box that you can find among the parts. Drag them and drop them into the breadboard window as shown in **Figure 20**.

Now we have all the parts, so let's wire it up. You'll want to refer back to **Figure 15** for this. First, let's do the power and ground wiring. Note that there are two power sources; according to the datasheet, we want +5 volts on pin 8 and +3 volts on pin 3 (for our backup supply). In **Figures 21** and **22**, we see that we can left-click the cursor on a part's connector, drag the wire to the connector we want to attach to, and then release the wire. Since this forms a straight line that might get in the way of seeing all the wires on a board, we add a bend (as shown in **Figures 23** and **24**) by first right-clicking on the wire to get the 'add bendpoint' indication and then left-clicking the cursor to grab the bend so that we can move it to where we want it.

Now that you see how to add a wire, you should make all the connections shown in **Figure 1**. When I ran the +5 volts up to the top of the breadboard, I was given a



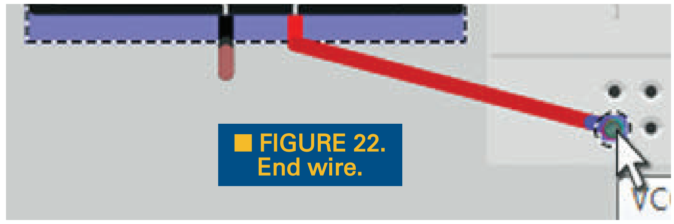
■ FIGURE 20. Add the Arduino and the battery box.



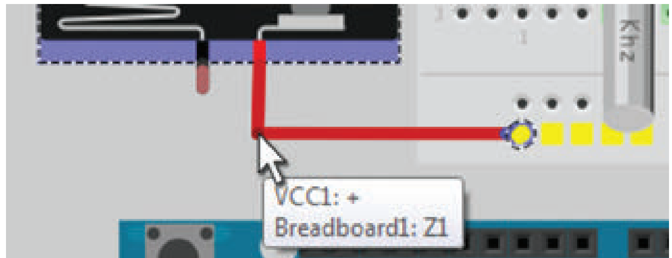
■ FIGURE 21. Start wire.

■ FIGURE 23. Start bend.

■ FIGURE 25. The DS1307RTC kit.



■ FIGURE 22. End wire.



■ FIGURE 24. End bend.

blue wire that wasn't very visible over the Arduino, so I right-clicked on it and got the option to change the wire color to yellow. I then changed the Arduino ground wire color from blue to black. I suggest you change all the wire colors to match the illustration in **Figure 1**, just to keep things simple.

Right angle bends are fine, but when you build this with real wires you are going to have a lot of curves. Fritzing helps illustrate this by allowing you to add curved bends. Just hold down the CTRL key when you left-click on a wire, and you'll get a rounded bend. This is something you need to fool around with a bit since describing it isn't as useful as actually watching what happens. You should get something like the curves shown in **Figure 1**.

Now we know how last month's Arduino/DS1307 RTC on a breadboard was generated. We are now ready to use this design in Fritzing to generate a schematic and layout for a printed circuit board — which we will do next month in Part 2.

Another Real Time Clock Option

This two-part Fritzing tutorial shows you how to design and build your own DS1307-based real time clock. If you just want one to play with, you can purchase the DS1307RTC kit (which contains all the parts you'll need) from the *Nuts & Volts* Webstore. See you next time. **NV**

BY LOUIS E. FRENZEL W5LEF

M2M
Machine-to-Machine Communications
The Next Big Thing

Electronic communications has always been about humans communicating with one another whether it is by radio, TV, ham radio, email, texting, or Skyping. All the while — in the background — there has been an ongoing expansion of communications between machines or non-human objects. This movement is known as machine-to-machine (M2M) communications. Think of it as communications in the background because we don't usually see it or feel it in any way. It just happens as it does the specific job assigned to it. Now, because of low cost wireless and other forms of electronic communications, M2M is a rapidly growing phenomenon. It may be affecting you now, and in the future it will certainly have a positive impact on your life.

JUST WHAT IS M2M?

M2M is essentially automatic communications between two "things." An example is a vending machine sending data to a remote computer indicating that it needs refilling and the money box needs emptying. Or, it could be a remote oil pipeline sending a temperature or flow rate reading back to a pumping station. Another instance is a truck sending its location back to the home office via GPS coordinates. Then, there is the air quality sensor sending data back to a monitoring station that can give an alert. The list goes on. All of these operations take place without human interaction. Although, at some point, a human does benefit in some way from the communicated data. The more recognizable name for such communications is telemetry. Telemetry is the science of measuring things at a distance. It is generally associated with measurements on a missile or satellite. Measurements in a chemical plant are another example. Again, the goal is data collection from a remote location. In some cases, feedback control can take place automatically. The whole idea of M2M is automated remote monitoring and control. Wireless and wired methods of communications make that possible, but that's not all. Because we have computers, such systems can make decisions and initiate actions without human interaction. The computers may be large fast PCs, but today embedded microcontrollers are small enough and cheap enough to build into any device no matter how

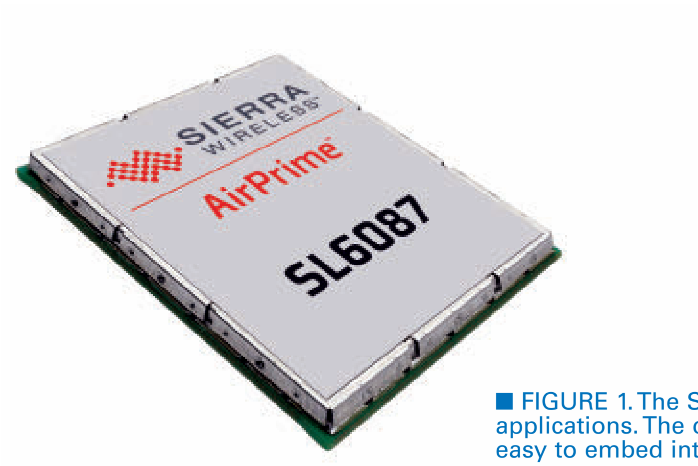


FIGURE 1. The Sierra wireless SL6087 GSM/GPRS module for M2M applications. The dimensions are only 25 x 30 x 2.65 mm, making it easy to embed into the smallest products.

■ FIGURE 2. The Telit Wireless Solutions HE910 module supports the HSPA+ 3G wireless standard but can fall back to slower GSM/GPRS/EDGE if needed. It covers five of the most popular cellular bands.

small or inexpensive it may be. The computers make the M2M automatic. While the pure definition of M2M is strictly machine-to-machine, it can also be thought of as machine-to-human or human-to-machine. The main common element is remote monitoring, control, or access. One subtle example is the embedded cell phone in your Kindle or iPad e-reader that can access a remote data base and download a book. Or, what about GM's OnStar system in Chevys, Buicks, and Cadillacs? It has a built-in cell phone that can be used to alert the system of an accident or break-in. Or, it can open your doors remotely if you lock your keys in or lose them. The remote monitoring and control of Predator drones can be considered a form of M2M.

M2M has many current uses and lots of potential. The health care industry is beginning to adopt it for in-hospital as well as remote patient monitoring. Transportation applications are big now especially for vehicle tracking. Each car or truck in a fleet contains a GPS satellite navigation receiver that sends the location via an M2M connection to a remote monitoring point. Look for more of that, as well as some yet to be identified uses.

Home monitoring and control is another growing application area. With M2M, you can monitor and control the home environment and implement energy-saving procedures. You can monitor and control appliances, security systems, and alternative energy sources like solar panels. The connection of a smart electric from the utility lets the electricity or gas usage be monitored by M2M. M2M is beginning to play an even larger role in the smart grid, with the growing use of communications and computers to better monitor and control energy usage for cost savings and pollution mitigation.

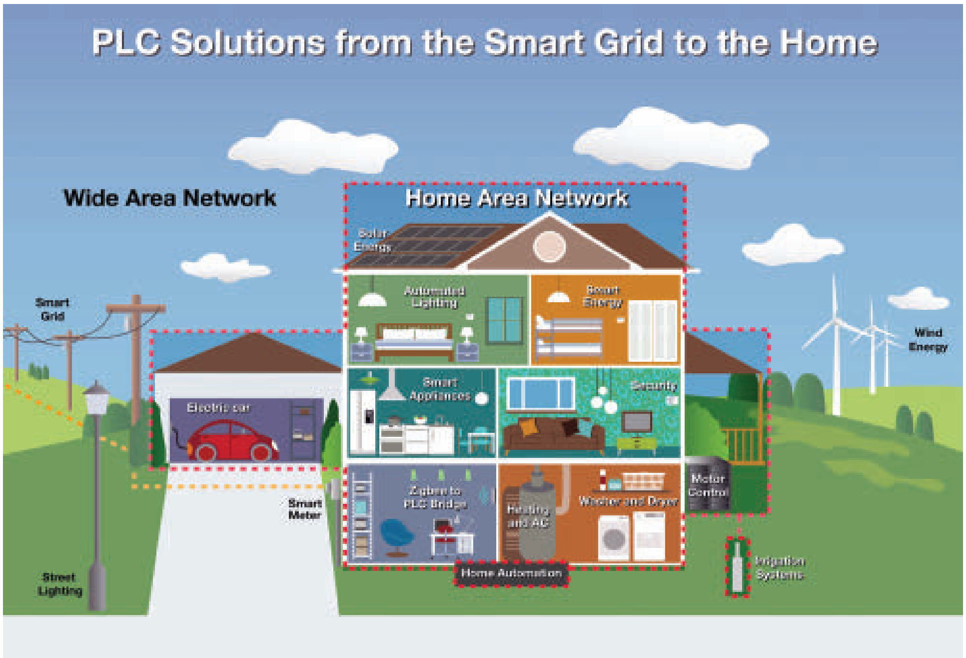
FACILITATING M2M

What makes M2M so simple to do is all the available wireless products that are easily adapted to the myriad of tasks. Today, cellular radios dominate the M2M landscape but Wi-Fi, ZigBee, and other popular technologies can be used. With cell phone usage nearing saturation, the cellular carriers are looking for new revenue from their networks. The smartphone revolution is providing some room for growth from data plans that feature more video and less texting and email. However, M2M is an untapped source of growth. Most of the big carriers like AT&T, Sprint, T-Mobile, and Verizon are building M2M services. Since most M2M applications have a low data rate, existing GSM/GPRS/EDGE and cdma2000 networks are adequate. The newest 3G and 4G cellular networks can provide even faster M2M data services.

Figure 1 shows a typical cellular transceiver that can be built into almost any other product. This is Sierra Wireless' SL6087 module that supports the 2G GSM/GPRS cellular standard. It can interface to almost any sensor or actuator (light, relay, solenoid, motor, etc.). Its ARM9 embedded controller with the appropriate software implements an M2M node that connects to some remote computer through the existing cellular system. **Figure 2** shows a 3G module. This one from Telit Wireless Solutions uses HSPA+ 3G technology, and works in the five most popular cellular bands. HSPA+ can achieve data rates up to 21 and 42 Mbps. It is also GSM/GPRS/EDGE capable.

In some cases, the sensor or actuator may be connected to a Wi-Fi public hot spot or other access point that, in turn, provides connection to the remote computer via an Ethernet network. Some systems use an

■ FIGURE 3. The B&B Electronics Wi-Fi system for M2M applications is designed for industrial monitoring and control applications. The Access Point accepts inputs from the embedded modules on the lower right. The application communicates back to the control point via an Ethernet connection.



■ FIGURE 4. This graphic shows the many possible uses of PLC for M2M applications in the home. It includes appliance monitoring and control, security, and connection back to the electric grid via the smart meter. (*Courtesy Texas Instruments.)*

intermediate gateway between the sensor node and the cellular network. The sensor talks to the gateway via Wi-Fi. Other sensor nodes may also use the same gateway. The gateway serves as an aggregation point for a whole network of sensors and/or actuators for a specific application. The gateway then is used as the backhaul connection through a cellular network or an Ethernet link to the remote collection and control computer. An example of such a system is one from B&B Electronics (see **Figure 3**). The embeddable Wi-Fi module is shown in the lower right. The Airborne Access Point is shown in two places and has an Ethernet port and two serial ports. This system uses the IEEE 802.11b/g standards. These products are designed primarily for industrial applications but will no doubt find other M2M applications.

The ZigBee wireless standard is another option for remote sensor networks. It is even lower in cost than Wi-Fi and cellular connections, and features even lower power consumption — something that benefits remote sensors whose battery life is essential over a long period of time. ZigBee also has a mesh network mode that allows one node to talk to any other node either directly or indirectly. ZigBee is a low power, short range technology limited to 10 meters or so.

If a sensor node is not near enough to the gateway, it can send its data through closer adjacent nodes that serve as repeaters. The data may travel through multiple nodes in the mesh network to get to the collection point. Other wireless technologies with potential for M2M are Z-Wave which is popular in home control and Bluetooth that has both low power and mesh networking capability making it an option for some applications. Another possibility is proprietary systems based on ISM (industrial-scientific-medical) band radios in the 433 and 915 MHz ranges.

Even short range technologies like RFID and NFC could find a use in some M2M applications. Radio frequency identification (RFID) is the use of passive (no battery power) tags to locate and track items to which they are attached. The transmit/receive range is short from a few inches to several feet which limits their use, but they're a good fit for applications such as inventory and shipping control.

Near-field communications (NFC) is that short range technology using the 13.56 MHz band for payments and access. An NFC powered smartphone can use NFC like a credit card for paying for purchases or for access to trains or buildings.

While wireless communication technologies dominate M2M, power line communications (PLC) is another option. This is the use of the AC power lines for data transmission. The binary data from a sensor of a control signal to an actuator is used to modulate a carrier that is superimposed on the 60 Hz AC power voltage for transmission.

While transmission distances are generally limited to a home or neighborhood, it is a low cost way to transmit data. In some smart energy systems, a PLC node sends its data to a nearby collection or aggregation point that is then connected via cellular backhaul to the utility. **Figure 4** shows the full potential of PLC in a home network environment.

THE INTERNET OF THINGS

You may have already heard of this movement which has the goal of connecting everything to the Internet. Since we have already networked virtually all the PCs, laptops, tablets, and cell phones, it's time to move on to everything else. Also called the "Internet of Everything," this is the use of M2M to provide a way to connect even the smallest, cheapest, and non-significant devices to the Internet for potential monitoring or control. It is being done right now but with low cost wireless and PLC transceivers with Internet interfaces. Literally anything can be attached. For example, you could control a specific light switch in your home from your smartphone. Or, you could use your iPad or other tablet to monitor a security camera in your home from anywhere in the world.

There are so many examples of this and so much potential, it is difficult to cover them all. You can think up your own applications, such as your refrigerator reporting its temperature or contents. Or, how about turning on your sprinkler system from anywhere or checking the presence or location of your golf cart? Many applications make use of some location technology like GPS to pinpoint the position of the object being monitored or controlled.

Many possibilities are totally worthless and ridiculous. Yet many will make sense, and these will be viable because of the low cost of the hardware to do it.

M2M and the Internet of Things are in the early stages now but development and adoption is on-going. Massive growth is predicted over the years as the industry figures out the best applications. With new standards in the works, M2M is set to become a real influence in your life. **NV**

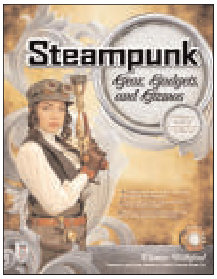
For complete product details, visit our webstore!!

The Nuts & Volts WEBSTORE

GREAT FOR DIYers!

Steampunk Gear, Gadgets, and Gizmos
by Thomas Willeford

A Maker's Guide to Creating Modern Artifacts

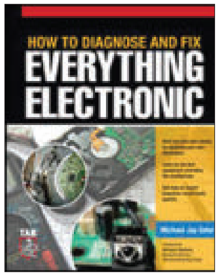


Welcome to the wondrous world of Thomas Willeford (a.k.a., Lord Archibald "Feathers" Featherstone) in which he shares his closely guarded secrets of Steampunkery. Filled with do-it-yourself projects, *Steampunk Gear, Gadgets, and Gizmos: A Maker's Guide to Creating Modern Artifacts* shows you how to build exquisite, ingenious contraptions on a budget.

$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

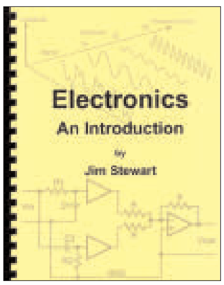
Programming Arduino Getting Started with Sketches
by Simon Monk

Program Arduino with ease!

Using clear, easy-to-follow examples, *Programming Arduino: Getting Started with Sketches* reveals the software side of Arduino and explains how to write well-crafted sketches using the modified C language of Arduino. No prior programming experience is required! The downloadable sample programs featured in the book can be used as-is or modified to suit your purposes.

$14.95

Electronics An Introduction
by Jim Stewart

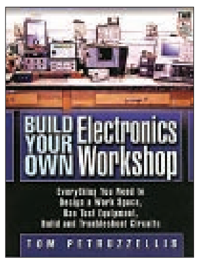


This book is designed as an in-depth introduction to important concepts in electronics. While electronics can be highly mathematical, this text is not about calculations. It is about how electronic equipment is able to extract, process, and present information held in electrical signals. If you are in — 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!

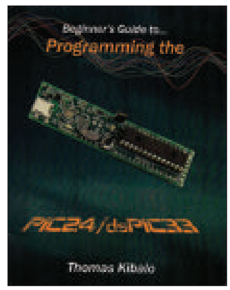


This value-packed resource provides everything needed to put together a fully functioning home electronics workshop! From finding space to stocking it with components to putting the shop into action — building, testing, and troubleshooting systems. This great book has it all! And the best part is, it 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

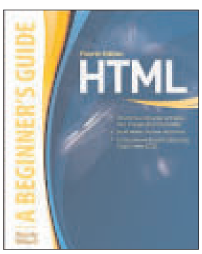


Kibalo takes you step by step through the fundamentals of programming the PIC24H which can equally be applied to the dsPIC33. His clear explanation of the inner workings make learning the PIC24H/dsPIC33 16-bit architecture easy. His code examples demonstrate how to perform the functions most applications 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*

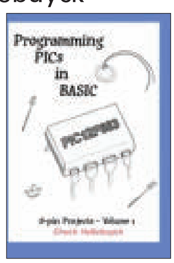
HTML: A Beginner's Guide
by Wendy Willard

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.



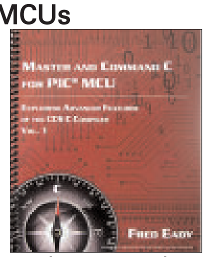
$29.95

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



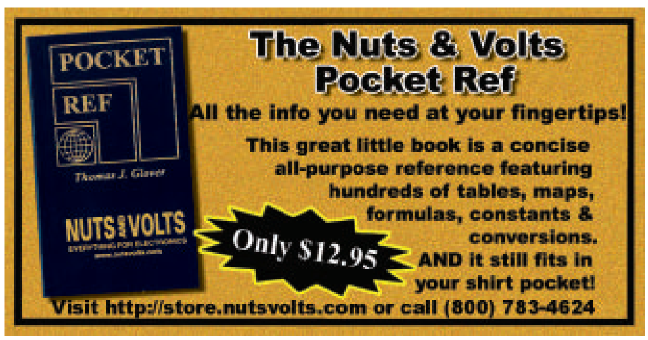
Master and Command C for PIC MCU, Volume 1 aims to help readers get the most out of the Custom Computer Services C compiler for PIC microcontrollers. The author describes some basic compiler operations that will help programmers particularly those new to the craft create solid code that lends itself 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!

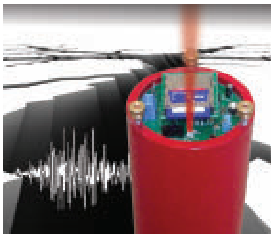
BOOK & KIT COMBOS

CD-ROM SPECIAL



PROJECTS

Seismograph Kit

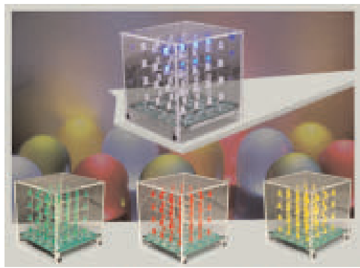


As seen in the May 2012 issue. Now you can record your own shaking, rattling, and rolling.

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**

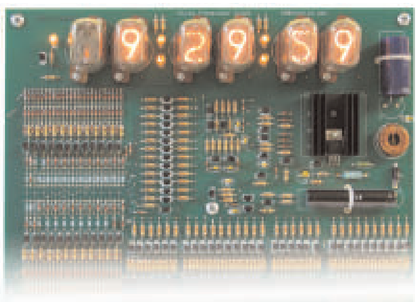
Battery Marvel Kit

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**

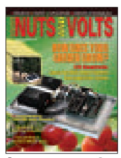
Neon Transistor Clock Kit



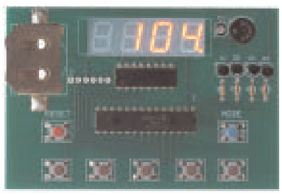
Add HIGH VOLTAGE to your clock!
This is a Nixie Tube display version of the Transistor Clock. It uses only discrete components — no integrated circuits.
For more info, see the April 2012 issue.

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

Sorting Counter Kit



As seen in the July 2011 issue.

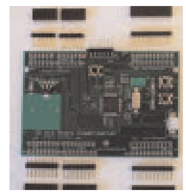


Sorting counters have many uses — keeping score, counting parts, counting people — it is just a handy gadget to have around. This is a very simple project for those who want to learn to solder or are interested in using microprocessors and how they function. No special tools are needed, just a small 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**

32-Bit Micro Experimenter Board

The 32-Bit Micro Experimenter is the fastest way to learn 32-bit microcontrollers.

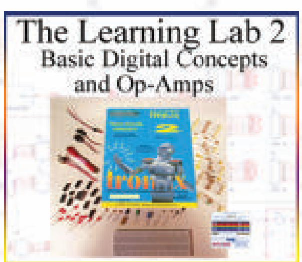


The kit includes onboard 46 programmable I/O and USB, free software, carefully documented step-by-step experiments for USB, embedded web server, graphics and audio, wireless, RTOS, and file I/O. User pushbuttons, LEDs, and 32 kHz clock crystal. Can be used in solderless breadboard environment or stand-alone.

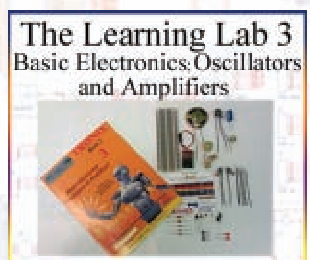
Subscriber's Price **$89.95**
Non-Subscriber's Price **$93.95**

FOR BEGINNER GEEKS!

$59.95



$49.95



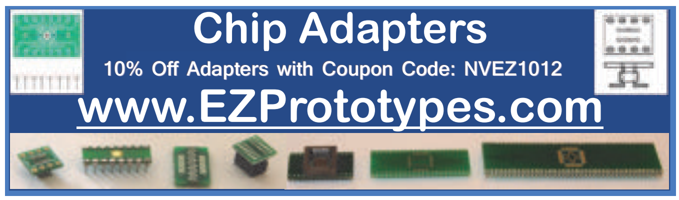
$39.95

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

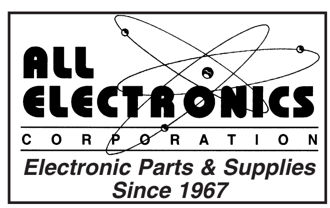
As you do each experiment, you learn how basic components work in a circuit, and continue to build your arsenal of knowledge with each successive experiment.

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

ELECTRONET



For the ElectroNet online, go to **www.nutsvolts.com** click ***Electro-Net***



■ BY FRED EADY

Discuss this article in the *Nuts & Volts* forums at http://forum.nutsvolts.com.

PUTTING BASIC4ANDROID IN THE DRIVER'S SEAT

I've gone Android crazy! I had no use for a smartphone and didn't have any notions of owning one. Now, I have a Droid Bionic AND a Samsung Galaxy Tab 10.1. Oh, yeah, and an HTC Droid Eris my wife discarded when she got her Droid RAZR. Frankly, I use my Bionic more as a computing device than as a phone. The Bionic has effectively replaced my trusty 30 year old HP programmer's calculator. Datasheets are my life. I used to read and collect them on my laptop. Not anymore. My "office"-equipped Galaxy Tab is now the primary PDF display device. My wife won't let me plug into her new RAZR, but you can bet your paycheck that I've plugged into my Eris, Bionic, and Galaxy Tab. Honestly, that was my main reason behind obtaining them. I'm a programmer and the Android devices are programmable. If you're interested in plugging into that intelligent Android brick you call a phone, get in the truck.

SO YOU WANT TO PROGRAM YOUR PHONE

If you want to follow the traditional Android program development path, you'll have to use Java and XML. Java is a great programming language if you know how to use it. In addition to Java and XML, you'll need to install and absorb an Android-bent IDE like Eclipse. The whole Java, XML, Eclipse ball of Android wax is underwritten by the Android SDK. Regardless of your Java knowledge level, you can't beat the price. All of the aforementioned Android programming tools are FREE. All you have to invest is your time to write meaningful Android applications using Java and Eclipse. If you don't already know how to code in Java, you'll invest a ton of time to produce that very first app.

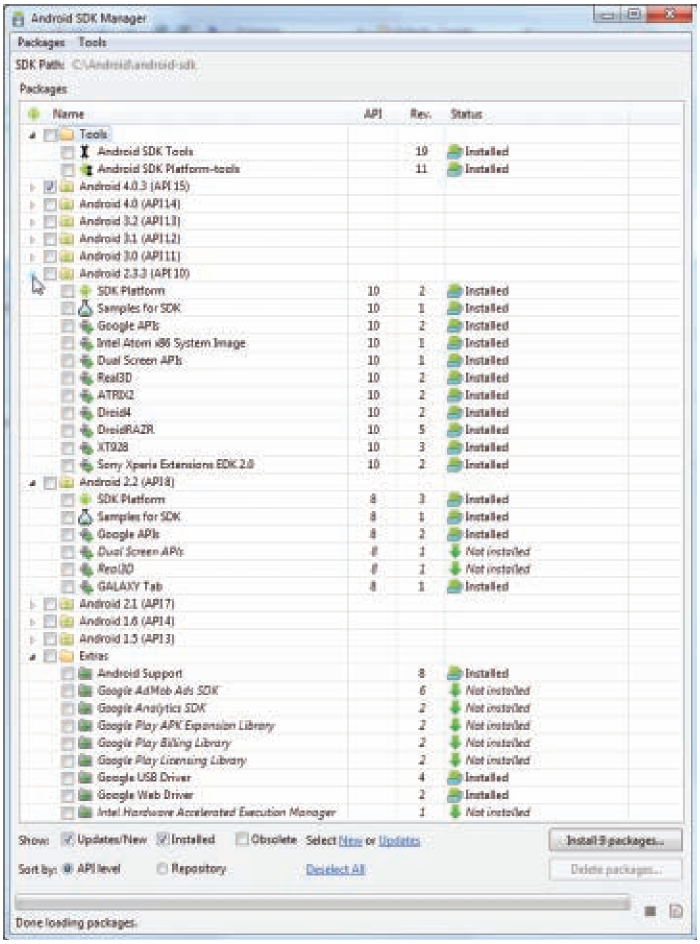
Now that you're in the truck, you'll notice that I'm not driving down that old familiar road with you. When it comes to writing and executing Android code, understanding and using the Android SDK is a must. However, you don't need to be a Java expert to write Android apps. Eclipse is a wonderful Android tool. You don't need it either. Pull that seatbelt tight and enjoy the ride.

INSTANT ANDROID GRATIFICATION

Well, almost. Before we can push some program bits through that smartphone USB portal, we need to establish an application development base. As I mentioned before, that base is the Android SDK. The Android SDK is a collection of API libraries and developer tools that allow us to build, test, and debug our Android firmware. The Android SDK is a free tool that can be downloaded from the Android Developer's site (**http://developer.android.com**). Like most everything offered up in binary these days, you can get the code in three flavors which are Windows, MAC, and Linux.

You don't have to have a Java degree to write Android apps in the town we're headed toward. However, Java is the gun that fires the Android SDK bullets. With that, we'll also need to carry the Java JDK in our Android toolbox. Like the Android SDK, the Java JDK is a free tool that can be obtained from the Oracle site (**www.oracle.com**).

■ SCREENSHOT 1. Everything revolves around the Android APIs. Plus, you can't beat the price of this application.



Before beginning the discussion, I drove down the traditional Android development road. I downloaded and installed the Android SDK, the Java JDK, and Eclipse. I actually got a simple app to compile and run. Other than writing Android apps, I went through the Eclipse-based Android development cycle in order to be able to describe it to you here. My other interest in producing Android apps involves "talking" to my PIC microcontroller-based gadgets using my Bionic and Galaxy Tab. Before the sun sets on my dreams, I plan to use the Android Bluetooth and USB communications portals as the "other end" of my PIC data pipes. Unfortunately, I couldn't see myself writing a 100 column series on Android programming. So, I searched for a higher road.

BACK TO BASICS

Basic4android, that is. Most of us have built PC applications using Visual Basic or Visual C++. Now, we can build Android apps in a similar fashion. Basic4android is very similar in touch and feel to Visual Basic. However, instead of sliding on Windows API calls, Basic4android rides on the Android SDK and Java JDK. Basic4android generates pure compilations of native Android code that don't require extra run-time dependencies.

Before deciding to purchase a copy of Basic4android, I was very skeptical. How could something so detailed and complex be pounded into a Visual Basic-like package that is easy to use? I decided to take a chance and join the 25,000 developers that took the plunge before me.

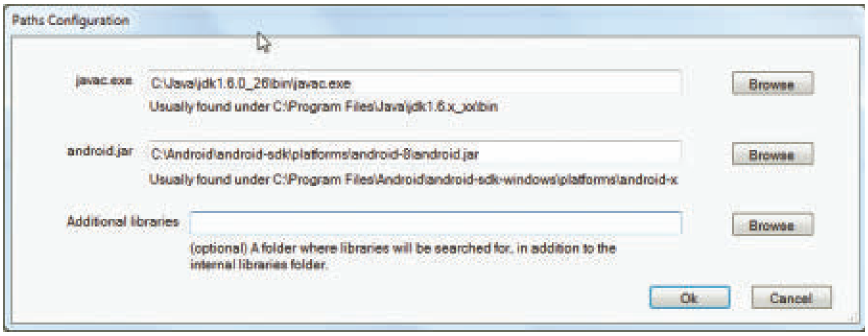
NOT WASTING ANY TIME

Let's stop talking and get cranking. The first task that has to be performed is the downloading and installation of the Java JDK. Basic4android depends on the Android SDK. The Android SDK does not play well with the 64-bit version of Java. So, even if you are running Basic4android on a 64-bit PC, you must load the 32-bit version of the Java 6 JDK.

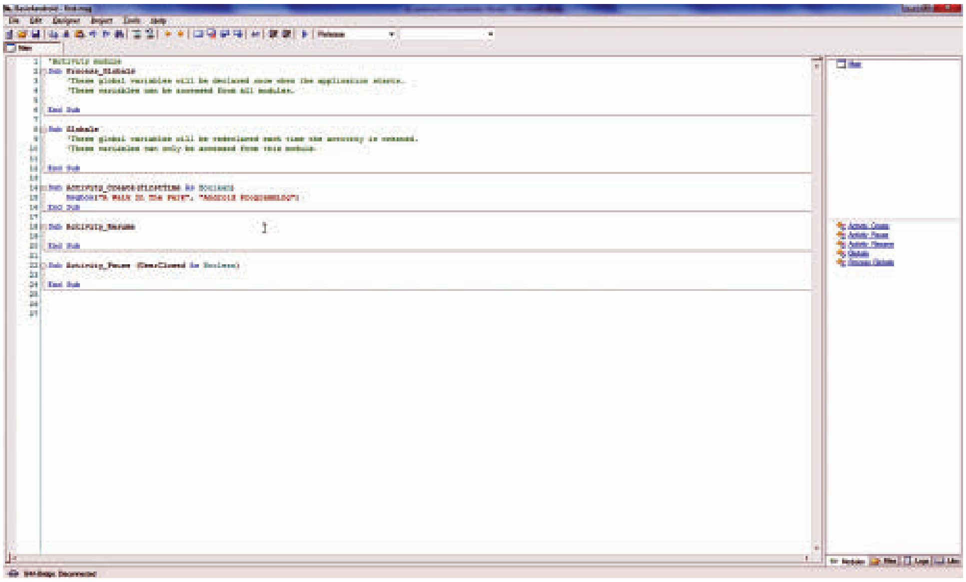
The next milestone on the road to real Android app building is the installation of the Android SDK. Once you've installed it, you'll need to install at least one version of the Android APIs. As you can see in **Screenshot 1**, I've got API 8 and API 10 loaded. It's also a very good idea to load the Google USB driver. Installing this driver will allow you to bypass the Android emulator and develop your apps on the real Android device. As you get more comfortable with the Android way of life, you'll find yourself loading additional APIs and drivers.

Now that the bedrock is laid, you can savor the installation of Basic4android. Following the completion of the install, you'll need to inform Basic4android as to the location of the Java 6 JDK and the Android API set you wish to use. **Screenshot 2** tells you where my Java 6 JDK resides and that I'm compiling using API 8.

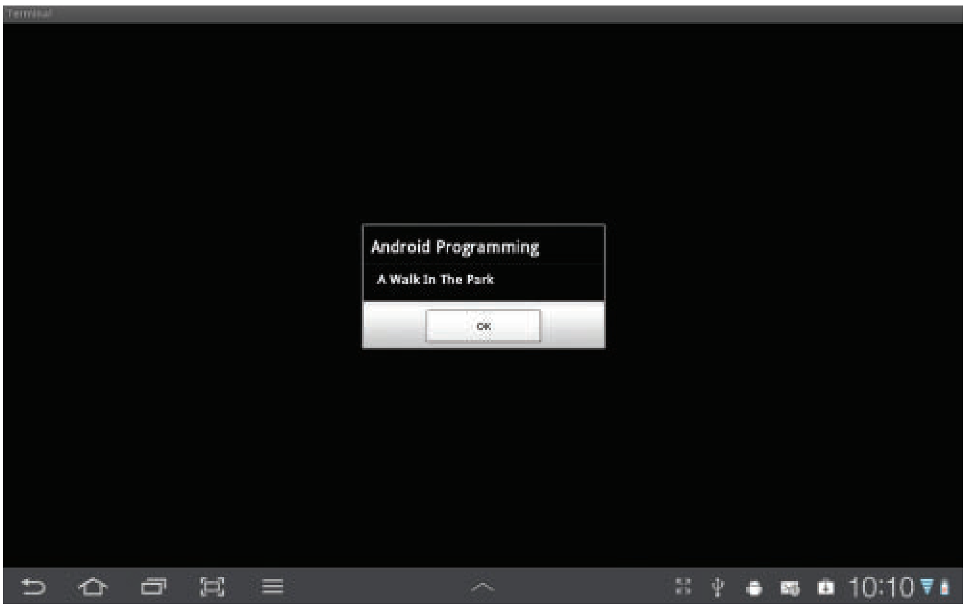
If you are not familiar with Java and XML, at this point you would be scratching your head and doing lots of reading. In this case, we simply start the Basic4android application. To prove how easy it is to get positive results when coding with Basic4android, the one line of code in



■ SCREENSHOT 2. This is where you specify the Android API set you want to use in your app.



■ SCREENSHOT 3. This little piece of code isn't exactly a program, but it is just enough to display the message you see in Screenshot 4 on my Galaxy Tab.



■ SCREENSHOT 4. The Android Msgbox code works just like its Visual Basic cousin.

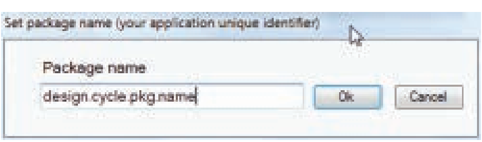
Screenshot 3 results in what you see in **Screenshot 4**.

THE IMPORTANT STUFF

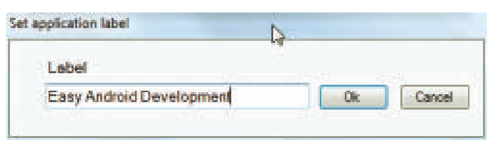
When you really get down to it, the coding is secondary to the look and feel. So, let's not do that "Hello World" thing just yet. I'm anxious to see something on the Android device's display that I can touch and see something happen. Once we do that, the application ideas will form all by themselves. Let's do some work on that Msgbox code and see what happens when we punch and twist some of the Basic4android's buttons and knobs.

The first order of business is to make sure that we have the correct driver for the Android device we wish to use in the app development process. Obviously, I have already done that. If your Android device is giving you fits and won't "connect" to the Basic4android IDE, you will have to go through an install or update driver process. One great thing about Android is the support level. The Android Developer's site has an intense section that walks you through getting the Google USB driver package and installing drivers. In the case of my HTC Eris, I came across an Android developer that was a keeper of the flame. He provided the USB VID and PID information for my Eris. The Android USB VID and PID information for supported Android phones and devices is found within an Android SDK file called android_winusb.inf. Here's the HTC Eris information that I added to the android_winusb.inf file:

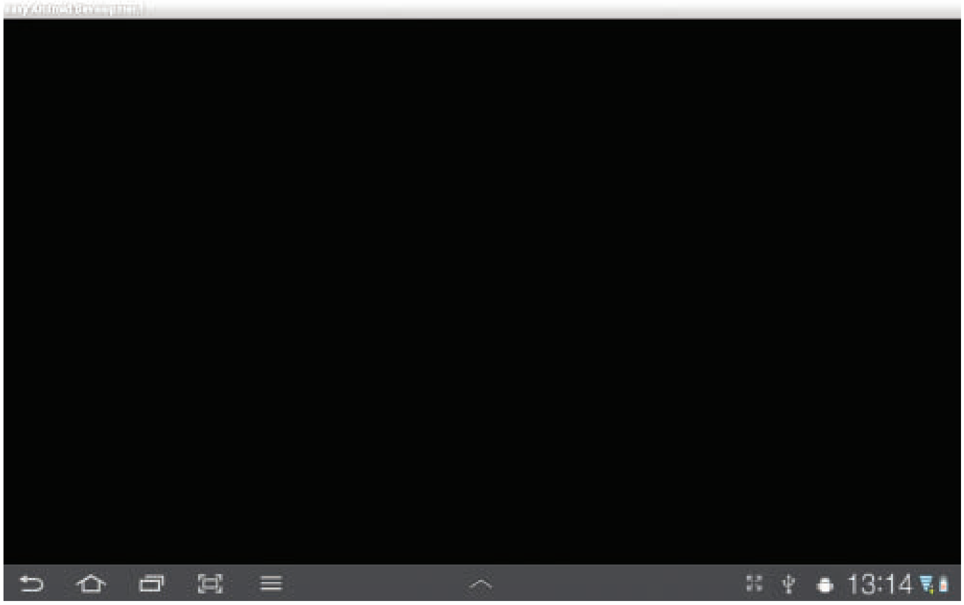
```
[Google.NTx86]
; HTC ERIS
%CompositeAdbInterface% = USB_Install,
USB\VID_0BB4&PID_0C98&MI_01
```



■ SCREENSHOT 5. The package name is a unique identifier for your Android app. The package name you enter has to be in a dotted division format. For instance, this.is.my.package.name.



■ SCREENSHOT 6. The Basic4android application label is used to identify the application in a visual manner to the user.



■ SCREENSHOT 7. This is the result of answering a couple of questions and compiling the Basic4android default Activity Module skeleton.

```
; HTC Dream
%SingleAdbInterface%
= USB_Install, USB\VID_0BB4
&PID_0C01
%CompositeAdbInterface%
= USB_Install, USB\VID_0BB4
&PID_0C02&MI_01
%SingleBootLoaderInterface%
= USB_Install, USB\VID_0BB4
&PID_0FFF
; HTC Magic
%CompositeAdbInterface%
= USB_Install, USB\VID_0BB4
&PID_0C03&MI_01
```

I included the USB VID/PID entries for an HTC Dream and HTC Magic to give you an idea of the contents of the android_winusb.inf file. The PC's recognition of your Android device's SD card is a good indication that your Android device driver is working.

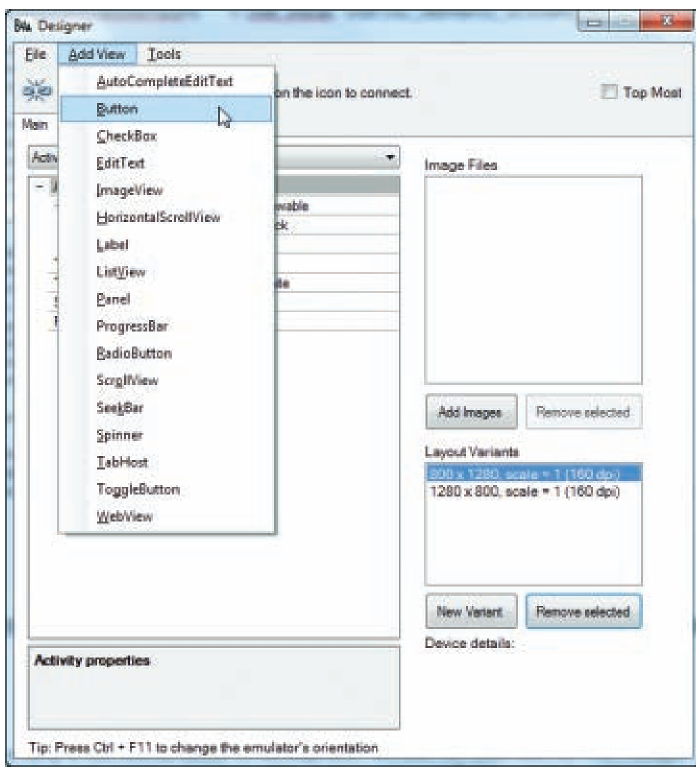
Once you get your Android device to register with the PC, start Basic4android and try to compile the default Activity Module skeleton. Basic4android will force you to first give the project a name and then save the project. For instance, I created a folder called *b4a-workspace*. Inside *of b4a-workspace*, I created a subfolder called *design-cycle*. The Basic4android project called *first-run* was saved in the *design-cycle* folder.

Basic4android's next move will be to collect a package name. The package name is a unique app identifier. It must be entered in a dot delimited format. So, as you can see in **Screenshot 5**, we will call our package *design.cycle.pkg.name*. Basic4android also requires an application label like the one entered in **Screenshot 6**. If you're wondering what the application label is good for, check out the upper leftmost portion of **Screenshot 7**. After entering the application label, Basic4android will compile the skeleton and load it into your attached Android device. In my case, the target Android device is my Galaxy Tab. At this point, we're ready to start building the visual portion of our Android app.

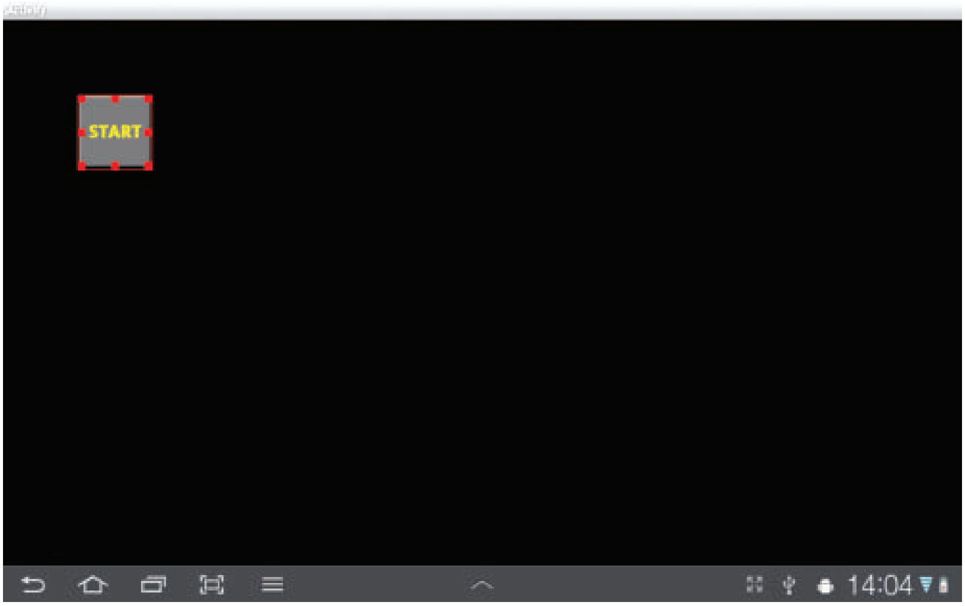
IN THE COMFORT ZONE

Like its counterpart Eclipse, the Basic4android IDE includes an Android emulator which is ruled by the Android Virtual Device Manager. The Device Manager is part of the Android SDK and is the tool that allows you to create a virtual Android device. Since we are using a real Android device for development, we need not fiddle with the AVD Manager. With that, let's move on and start adding some meat to our fledgling Android application.

If you've ever written a Visual Basic or Visual C++ program, here's where that knowledge can be applied to your Basic4android app. We're going to invoke the Basic4android Designer from within the Basic4android IDE and add a button to our infant Android application. Note that in **Screenshot 8**, I have opened Basic4android Designer and added a couple of layouts for the Galaxy Tab 10.1. I've also selected to add a button to our application. By taking advantage of the Basic4android IDE's Screenshot tool, I am able to show you the button I



■ SCREENSHOT 8. The further we delve into the Basic4android application development process, the more it begins to smell like developing with good old Visual Basic.



■ SCREENSHOT 9. The Galaxy Tab's screenshot capability is overridden when it is in debug mode. So, to show you the new button, I resorted to the Basic4android IDE's Screenshot tool.

just instantiated in **Screenshot 9**. As you can see, by using the Basic4android Designer attribute controls I've added some text and color to the START button. The Basic4android attribute control process is almost identical to the Visual Basic attribute control process.

Screenshot 10 is a view of all of the Basic4android Designer's tools. Generate Members is used to create a subroutine for the selected objects we place on the Android palette. Since we've only created a START button thus far, the Generate Members tool will only create a single subroutine called *btnStart_Click*. We could also have generated *btnStart_Up* or *btn_Start_Down* by selecting them in the Generate Members selection window. For now, just clicking the button will do fine. Naturally, the Basic4android IDE wants us to name our new single-button layout. I've done so in **Screenshot 11**. Now, we need to write just a wee bit of code:

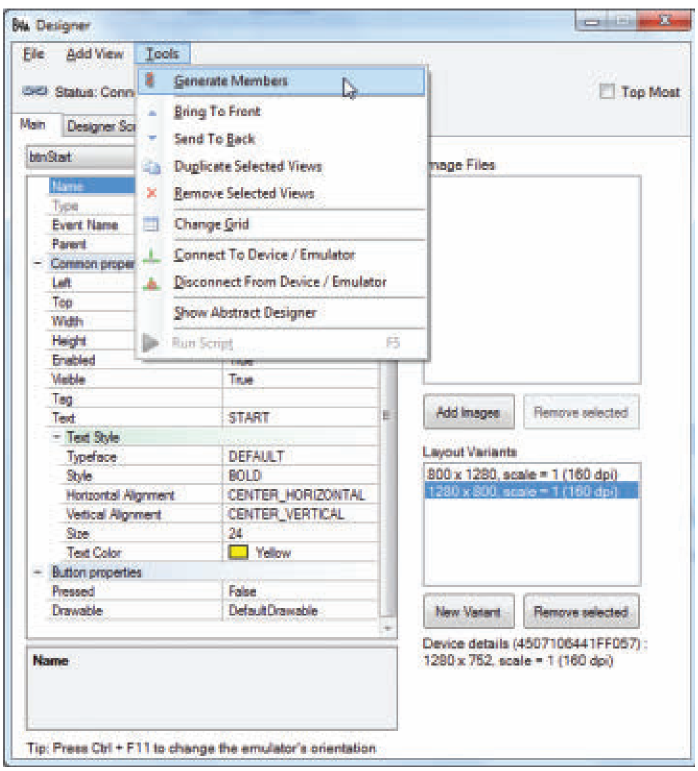
```
Sub Activity_Create(FirstTime As Boolean)
      Activity.LoadLayout("designcyclelayout")
End Sub
```

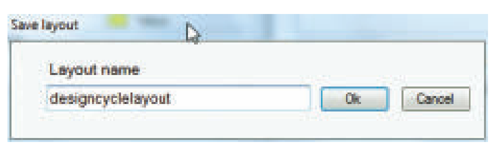
Let's compile what we have so far and see what turns up. Behold **Screenshot 12**. Note that the screen title is START BUTTON. That is actually the Activity name which I specified in the Basic4android Designer. By the way, in addition to compiling this little START Button app, Basic4android installed it on my Galaxy Tab. I can actually disconnect my Galaxy Tab from the PC and run the START Button app by tapping its icon. That's pretty sweet for just one line of code. With that, let's write another line of code to have the START button kick off what is called a Toast Message:

```
Sub btnStart_Click
ToastMessageShow("START BUTTON PRESSED",True)
End Sub
```

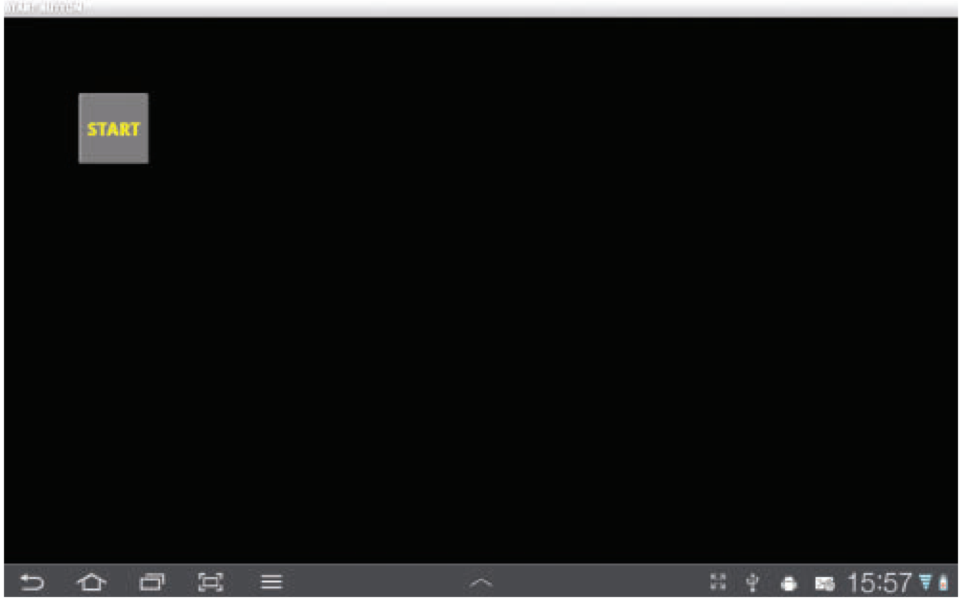
A Toast Message displays its text for a short period of time and then disappears. The "True" at the end of the text extends the duration of the Toast Message. I extended its appearance time so I could capture **Screenshot 13** for you.

■ SCREENSHOT 10. I really wanted to show all of the options available to you within the Basic4android Designer's Tools menu.





■ SCREENSHOT 11. We will use this layout name in our application. It will be the argument of what is equivalent to a VG Load Form statement.



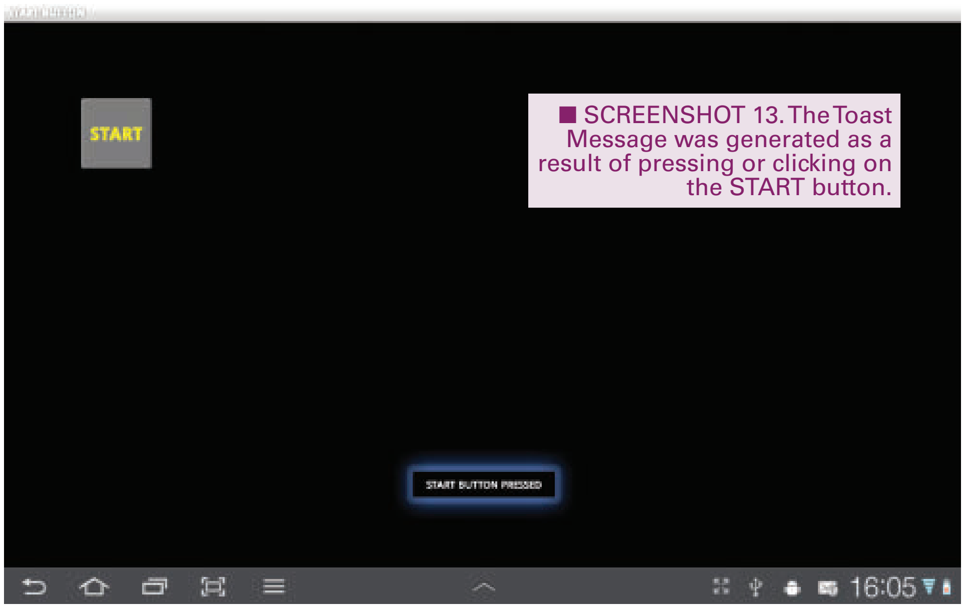
■ SCREENSHOT 12. Well, that's good for a "START." One line of code.

ANDROID IN A DAY

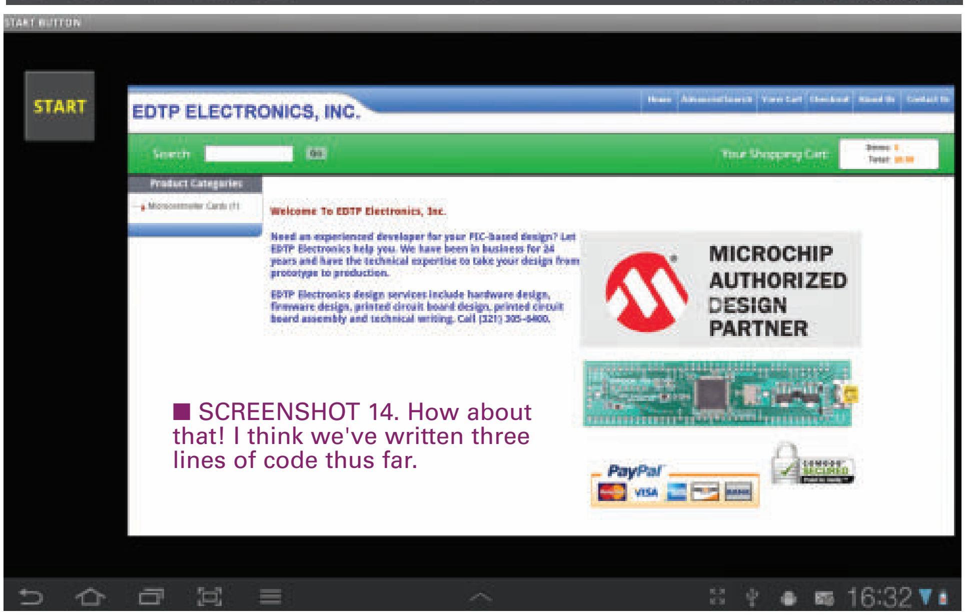
I am no longer a skeptic. Basic4android works and it works well. In just a few pages of text and a couple of lines of Basic4android code, we programmed the action kicked off by the click of a button. What do you think this code will do?

```
'Activity module
Sub Process_Globals
   'These global variables
   'will be declared once when
   'the application starts.
   'These variables can be
   'accessed from all modules.
End Sub

Sub Globals
   'These global variables
   'will be redeclared each
   'time the activity is
   'created.
   'These variables can only
   'be accessed from this
   'module.
   Dim WebView1 As WebView
End Sub

Sub Activity_Create(FirstTime
As Boolean)
         Activity.LoadLayout
         "designcyclelayout")
End Sub

Sub Activity_Resume
End Sub

Sub Activity_Pause
(UserClosed As Boolean)
End Sub

Sub btnStart_Click
         WebView1.LoadUrl
         "http://www.edtp.com")
End Sub
```

I'll leave you with the contents of **Screenshot 14** and the satisfaction of knowing that you can add Android application development to your Design Cycle. **NV**



■ SCREENSHOT 13. The Toast Message was generated as a result of pressing or clicking on the START button.



■ SCREENSHOT 14. How about that! I think we've written three lines of code thus far.

SOURCES

Basic4android
Anywhere Software
www.basic4android.com

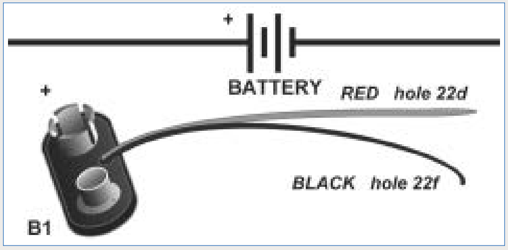
FUNDAMENTALS

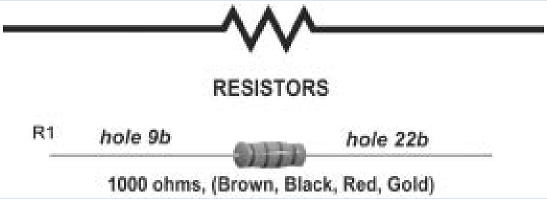
CIRCUITS AND BASICS FOR THE BEGINNER

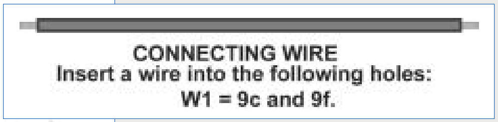
Ohm's Law

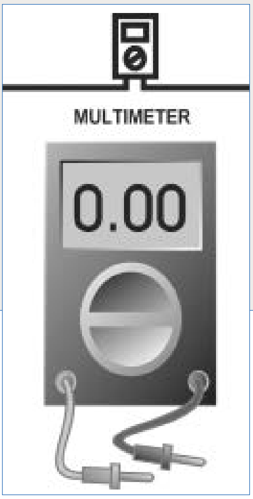
A kit for these experiments can be purchased from the *Nuts & Volts* Webstore at **www.nutsvolts.com** or call us at **800-783-4624**.

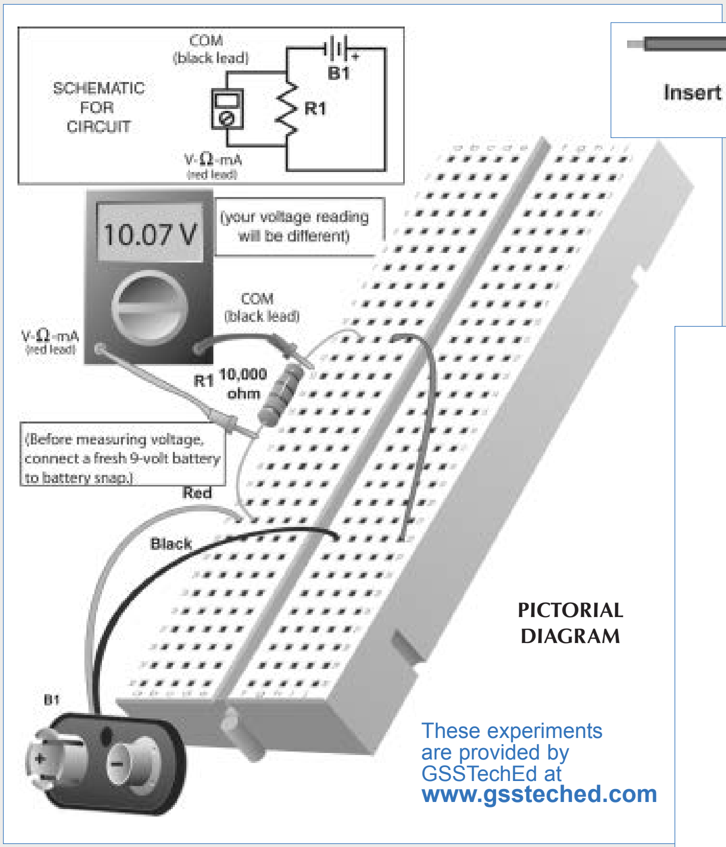
Ohm's Law states that the Voltage is equal to the Resistance times the Current. It is written E = I times R. (E is Voltage in volts; I is the Current in amperes; and R is Resistance in Ohms.) This time, you are going to connect a 10,000 ohm resistor to a nine volt battery and measure the battery voltage. Then, you are going to measure the amount of current flowing in this circuit. Next — using Ohm's Law — you are going to insert your measured battery voltage, your measured current, and your measured resistance value into the Ohm's Law formula to verify that it really works.



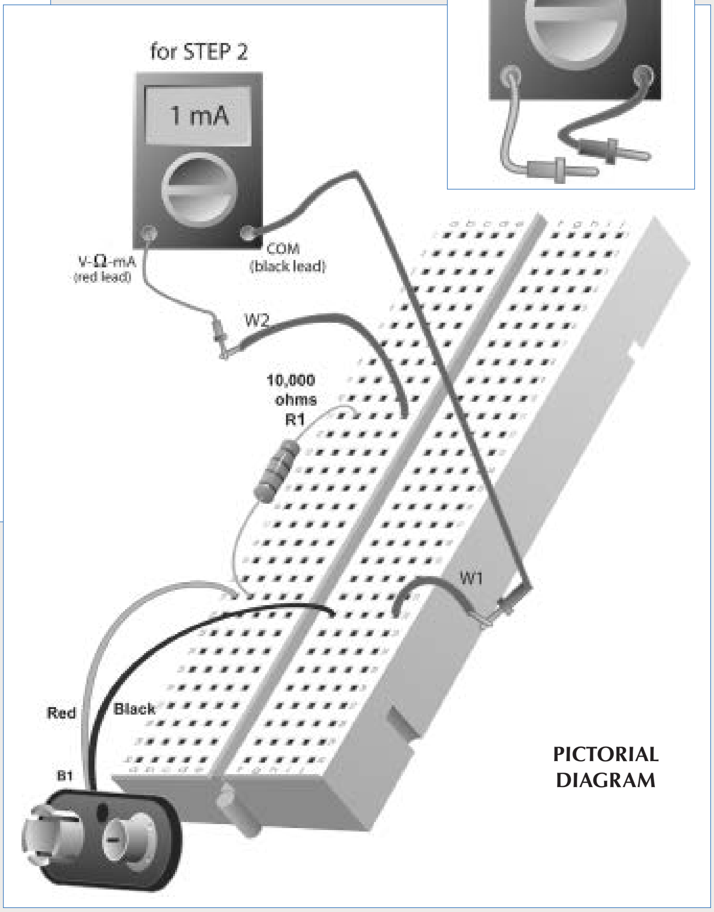








These experiments are provided by GSSTechEd at **www.gssteched.com**

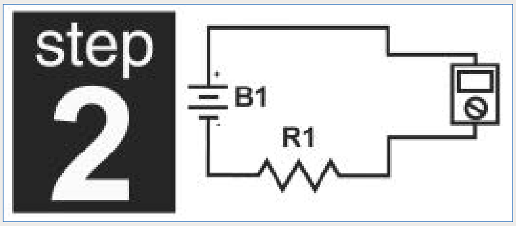


You can order parts for this experiment from their website as follows:

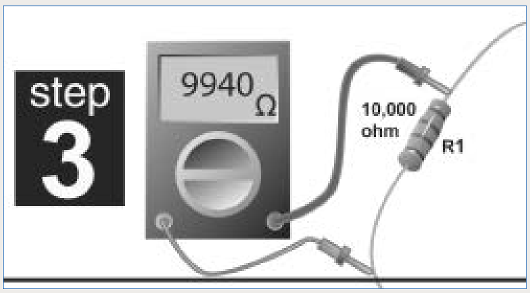
GK35002	Nine volt battery snap
9VB	Standard nine volt battery
GK45010	Two 4" solid wire stripped on ends
GK01089	Resistor 10,000 ohms; 1/2 watt 5%
CB111	Solderless circuit board; 600 holes
V-DVM850BL	Digital multimeter

step 1

Put the red lead of the battery snap in hole 21a and the black lead in 22f as shown in the pictorial. Connect a nine volt battery to the snap. Connect a 10,000 ohm resistor from hole 21b to 11b. Put a solid wire in hole 11d to 22j. Set the knob on your digital multimeter to DCV 20 to measure the voltage of the battery. Your meter should show about nine volts. (Ours measured 10.07 volts.)



Measure the current in the circuit. Refer to the schematic shown here. Build the circuit as shown in the pictorial diagram for Step 2; attach a nine volt battery to the battery snap. Set the meter to measure 20 mA and then connect the multimeter leads between the loose ends of W1 and W2. The meter should show about one milliampere of current flowing. It should measure about 0.001 amps or 1.00 mA. (Ours measured .001001 mA.)



Remove the resistor from the circuit to measure Ohms. Set your meter to measure 20K ohms and touch the leads to the ends of the resistor to measure its resistance. (Our resistor measured about 9,940 ohms.)

step 4

Verify Ohm's Law

Measured battery voltage = ______

Measured current = ________

Measured resistance = _______

E = I x R

______ = ______ x ______

E should equal approximately the value of I times R. Now try doubling the value of R in the circuit and do the measurements and calculations again.

Does Ohm's Law work?

Continued from page 27

Project Clock Standard and Web costs $35 (US) for a single-user license, Project Clock Pro costs $55 (US) for a single-user license, Project Clock C/S and Enterprise costs $75 (US) for a single license. Project Clock CE costs $20 for a single license. Multi-user and site licenses are available. Time-limited trial versions can be downloaded free of charge from the website.

For more information, contact:
CyberMatrix Corporation
Web: **www.cybermatrix.com**

5-6,000 MHz PHASE NOISE TESTER

Saelig Company, Inc., has introduced the APPH6000RM-IS

— an all-in-one comprehensive automated measurement system for evaluating RF signal sources. SSB phase noise, amplitude noise, AM noise measurement, additive or residual noise characterization, and baseband noise measurements up to 6 GHz can be made for sources such as crystal oscillators, PLL synthesizers, clocks, phase-locked VCOs, DROs, and many others.

Direct access to a built-in FFT analyzer mode allows baseband signal and (LF) noise analysis. The APPH6000RM-IS features a cross-correlator engine and internal low noise reference sources to enable fast and accurate "one-click" measurements.

Fully automated frequency acquisition and self-calibration simplify the use and applicability of the APPH6000RM-IS, resulting in fast measurement throughput and operator ease-of-use. Using proven cross-correlation measurement procedures and self-calibration routines, reproducible and accurate measurements are obtained even under changing environmental conditions.

The APPH6000RM-IS's operation can be facilitated using Ethernet 100BaseT LAN (VXI-11) or USB 2.0 host interfaces, or via a GPIB "listen and talk" option. A platform-independent intuitive graphical user interface (GUI) is used to control the device from a PC, while an API library and powerful SCPI command language set is also available for custom test applications.

Designed to operate from external 6V DC (110V adapter provided) at up to 15,000 feet and 0 to 45 degC, the APPH6000RM-IS can be configured to meet user requirements such as: selectable internal or external reference source; phase detector models; and frequency offset ranges.

Two-channel cross-correlation is also supported for lowest noise measurements down to -180 dBc/Hz. Applications include general-purpose phase noise tests, crystal oscillator and VCO testing, PLL synthesizer locking and characterization, supply noise verification, and automated production testing.

LOW COST 300 MHz OSCILLOSCOPE

Saelig Company also introduces the SDS9302 — a new, two-channel bench-top scope sporting a huge record length for each channel (10 MSa when sampling at up to 800 MSa/s; 10 KSa when sampling the maximum 3.2 G Sa/s).

This allows for zooming in on fast, intermittent pulses or glitches. The SDS9302 is a low cost versatile scope that has many useful features normally only seen on higher-end DSOs, including deep memory, external video-capable trigger, auto-measurements, auto-scaling, large 8" full color TFT LCD display, external monitor/projector output, XY mode, auto-set, averaging, math functions, USB output, waveform storage, pass/fail output, and a three year warranty.

FFT functionality is included for frequency spectrum display. The SDS9302 comes in a sleek, slim case and can be fitted with an optional battery — ideal for field or isolated operation.

The Autoscale feature automatically adjusts the vertical gain or the horizontal time base, or both together. This is useful for circuit probing. As the probe is moved from point to point on a circuit board, the display auto-adjusts for best trace presentation.

It works like AutoSet but instead of being a one-time function, it's active until turned off, keeping it "hands-free." An instant display of the frequency spectrum of the signal under test is provided with the FFT function.

The SDS9302 can automatically measure and display frequency and peak-peak/RMS/mean values, but cursors can also be moved to make individual readings. A built-in self-calibration facility improves measurement accuracy. Video monitoring is also possible since it can trigger on NTSC/PAL/SECAM line or field waveforms.

Other SDS9302 features include: manual cursor measurements, up to 19 automatic measurements (including frequency), high speed screen update, storage for up to 15 waveforms and set-up parameters, convenient USB interface with PC software, 400V (DC+AV peak) maximum input, optional rechargeable battery pack, and multi-language capabilities.

The lightweight SDS9302 oscilloscope is perfect for any engineer's or student's desk. The USB master and slave connections make

CLASSIFIEDS

KITS/PLANS

ENERGY SOLUTIONS

HARDWARE WANTED

LIGHTING

ROBOTICS

SURPLUS

COMPONENTS

ENGINEERING/ DESIGN SVCS

OSCILLOSCOPES

printing or storing results simple, as well. With a large 8" 800 x 600 pixel color TFT LCD, this scope is small enough and light enough to carry anywhere, weighing under four lbs.

Sensitivity range is 2 mV/div to 10V/div with maximum display of ±100V; horizontal display is from 1 ns/div to 100 s/div. Up to 15 waveforms may be stored internally for comparison with live inputs.

Another useful feature is that waveforms can be stored on plug-in USB memory sticks. A persistence control is also available which simulates older analog scopes to compare slow-moving waveforms with previous scans. 'A+B' and 'A-B' math capability is included, while the X-Y display feature reveals phase delays.

The SDS9302 offers onboard storage and USB output, making it useful in design, maintenance, and lab applications.

For more information, contact:
Saelig
Web: **www.saelig.com**

READER-TO-READER TECHFORUM

All New TechFORUM Online At www.nutsvolts.com

>>> QUESTIONS

Cell Phone Amplifier

I need a parts list and schematic to build a cell phone amplifier to connect to standard headphones.

#8121 **James Moore**
via email

Powering a Hydrogen Generator

I'm building a HHO generator and am trying to figure out the best way to power it, keeping the power usage low. A high voltage, low amp method like using a Tesla coil might do the trick. Any ideas to help me figure it out?

#8122 **Matthew**
via email

Audio Sound Spectrum

Is it possible to take the sound heard in the frequency range of 0 to 40 kHz and space it evenly into the 10 Hz to 20 kHz range? I want to hear what a dog hears but in the human range. I know this will make people's voices sound funny, but how would this be done and is there such a device? Is there even a mic for that upper range of up to 40 kHz?

#8123 **Robert Spencer**
Phoenix, AZ

Connecting an Old Printer to a Current PC

I have an IMP-24 printer with a Centronics interface. I would like to print to it from a Windows 7 or XP computer using some form of Basic. I have VB and Visual Studio 2010 Ultimate. However, there seems to be no drivers for this old printer. Is there a way to address LPT1 from Visual Basic or Visual C so I can send the data to the printer directly without going through the operating system?

#8124 **Mike**
Lake Forest, CA

>>> ANSWERS

[#6124 - June 2012]
Compressor Starter

I am looking for a positive temperature coefficient motor starter for a compressor. Or, I need help using different parts such as a single pole contactor/relay for a hard start.

The part I want to replace has the following number: RF-6850-18, and consists of a PTC starter and an overload protector. Can I replace this with a hard starter contacter?

I can photocopy the schematics and email anyone who can help me.

#1 Go to any well stocked refrigeration supply house and ask for a universal compressor starter.

This will be a device with leads. (I have no idea what's inside.) Connect it as shown on the package, removing the existing starter device.

These things work every time. Be sure to leave any original thermal protective device intact.

They can also be used on small air compressors that use PTC devices as starters.

Don Pomeroy
Manchester, NH

#2 I've bought PTC starters for refrigeration compressors from **www.repairclinic.com** on a few different occasions. A local appliance shop with a parts counter should be able to supply them too, either OEM or a universal replacement. The "hard start" kits you can get are nothing more than a standard PTC starter, overload protector, and capacitor stuffed into a housing. The one time I tried one of those, it failed within a few weeks.

If you want to try rolling your own, the schematic in **Figure 1** is said to work although I've not personally tried it. Digi-Key, Mouser, and other electronics distributors carry a selection of PTCs and bimetal circuit protectors that could be used.

James Sweet
via email

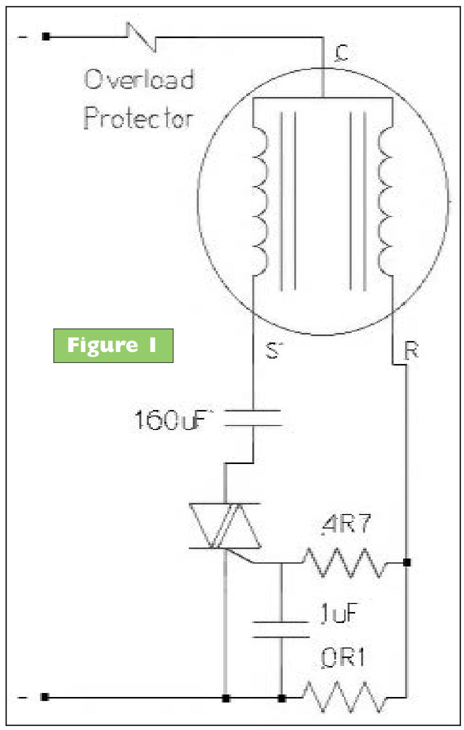


Figure 1

Check out the *ALL NEW* Tech Forum at **www.nutsvolts.com** where you'll find additional material to this column even before it's seen in print. You can participate, comment on answers, even get emailed when a new answer or comment is posted. This new feature is still under construction as of this writing, but should be ready by the time this is printed. *Check It Out!*

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!

Send all questions and answers by email to **forum@nutsvolts.com**
*Check at **www.nutsvolts.com** for tips and info on submitting to the forum.*

#3 I work for a major appliance repair company/retailer. Replacing the factory compressor start device with an after-market hard start kit is not recommended. Most of those kits — if not used or sized correctly — can cause permanent compressor failure. I would suggest trying to source the correct part by model and serial number of the refrigerator. There is a major appliance parts dealer in Cincinnati and Dayton that can help. If you can not find a model or serial, I suggest Whirlpool part number 8201786. This kit covers several different compressors, as well as wiring connections.

Sean Battito
Mohnton, PA

#4 The original part is shown at **www.repairclinic.com/PartDetail/Start-Device/RF685018/1569121**.

The three terminals on the device are C, S, and R. C is connected to the COMMON of the compressor, with S being START winding, and R being RUN winding.

I'd suggest replacing the original device, and adding the proper MARS hard start device to kick the compressor. You don't want to bypass the overload! PERIOD!

MARS Motors & Armatures, Inc., has hard start devices and torque multipliers for refrigeration/AC compressors. I've used a smaller MARS hard start device on several refrigerators, and they work fine. (**www.marsm-a.com/CGI-BIN/LANSAWEB?WEBEVENT+R0663 0136135017015394091+ML6+ENG**).

I can't find the paperwork on the last MARS device I purchased. Any local HVAC distributor should be able to locate what you need, depending on the compressor size. It's just a two wire hookup.

I'm including a PDF document from MARS and my refrigeration document for you to view. *Download them at **www.nutsvolts.com** in the Tech Forum.*

Larry Kraemer
via email

[#6125 - June 2012]
Generator Conversion

I'm considering converting an old car with manual steering to electric.

However, DC motors about 10 HP or more are expensive. I saw that the Northern Tool catalog has a 10 kw generator head with brushless technology (item 165928). Is this a 13 HP brushless DC motor I could modify?

You can ask the expert at Northern Tool, but I am sure he will tell you that this is a single phase, 220 VAC generator and cannot be used as a motor. A three-phase generator could possibly be used as a motor, but 60 Hz machines don't like to be run at other frequencies so variable speed would be a problem. You would need a clutch and transmission.

Russ Kincaid
Milford, NH

[#6126 - June 2012]
Thermostat Schematic

I need a simple schematic diagram which shows how the home thermostat controller is connected to the gas heater and to the air conditioning at the same time.

Here is documentation for a typical thermostat. Make sure you know what voltage your existing system uses before making a purchase.

I think the schematic is self-explanatory (**Figures 2A** and **2B**). *Additional documentation can be found at **www.nutsvolts.com** in the Tech Forum.*

Larry Kraemer
via email

[#6128 - June 2012]
DTMF Encoder

Where can I find a schematic or kit for a 16 button keypad encoder, without having to program a PIC? That's a regular telephone keypad plus A - D on the right side.

#1 The IC for DTMF encoding is a TP5089 from National, or a second source is TCM5089 from Texas Instruments.

It is available from Jameco (**Jameco.com**) as TP5089; their item #32803.

It's listed as available from stock on their website. You can also download datasheets from them to

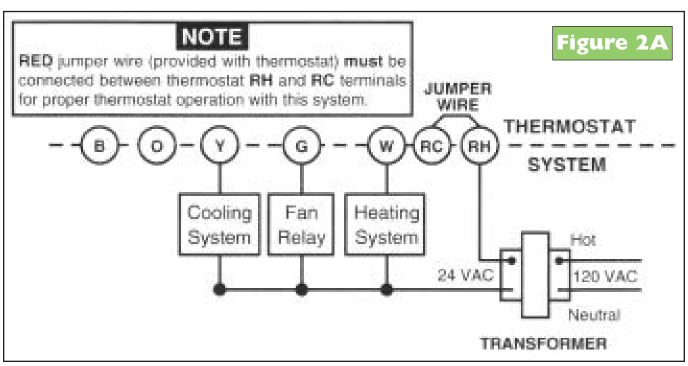


Figure 2A

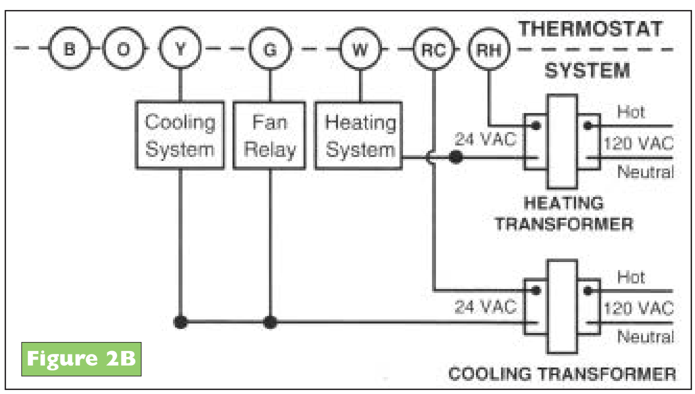


Figure 2B

check before ordering.

Don Pomeroy
Manchester, NH

#2 Try searching on eBay for "DTMF encoder." I found more than a few there, but they aren't cheap.

DTMF ENCODER GENERATOR
*** $34.00 *** 11 Days
www.ebay.com/itm/DTMF-ENCODER-GENERATOR-/400302522698?pt=LH_DefaultDomain_0&hash=item5d33e3c14a

HT9315C HOLTEK DTMF MULTI-FUNCTION ENCODER CHIP
*** $3.89 *** Only 1 day
www.ebay.com/itm/HT9315C-HOLTEK-DTMF-MULTIFUNTION-ENCODER-CHIP-/150826386913?pt=UK_BOI_Electrical_Components_Supplies_ET&hash=item231df405e1

DTMF ENCODER KIT WITH 16 KEY KEYPAD + TX KEYING OUTPUT
*** $30.32 *** 10 days
www.ebay.com/itm/DTMF-Encoder-kit-with-16-key-Keypad-TX-keying-output-/270993370582?pt=LH_Default Domain_3&hash=item3f1876add6

While these will all probably be sold before you read this, you'll most likely find others. I'd ask the "Electrical_Components_Supplies_ET" place since they may have other chips.

Phil Karras
via email

[#7122 - July 2012]
Timer Needed

I need a 2-3 second timer to operate a 24 volt relay, to open and close a garage door using a Desa Int'l wireless doorbell.

#1 This sounds like a classic application for the ubiquitous 555 timer IC.

The datasheet for the 555 includes many examples of using it, including the one-shot circuit that I think will meet your needs nicely. You can use a common transistor like the 2N2222 to drive the relay coil. Don't forget to put a reversed diode across the relay coil to protect the transistor.

James Sweet
via email

#2 I assume that the need is to have a delay before opening or closing the garage door. The circuit diagram below will do this.

The circuit (**Figure 3**) uses a 1Hz oscillator (ICL555) which clocks a 4015 shift register as a sequencer. When the output of pin 2 goes high, the 4N35 optocoupler conducts. The output shorts across a garage door pushbutton.

The "Desa Int'l wireless doorbell" will need to close the switch shown as "Enable PB." This can be done with either another optocoupler or suitable transistor logic to simulate the switch closure. In operation, the Enable signal (pin 15) must be active high long enough to be captured on the next clock edge. When the yellow LED goes on, the input has been captured. For the circuit shown, the delay is about four seconds. Increase the clock frequency by decreasing the 560K and/or 330K resistors in the clock section.

Note the series wiring of the RLED resistor, optocoupler (LED), and red LED. The 47 ohm resistor must be changed if the circuit is to operate on

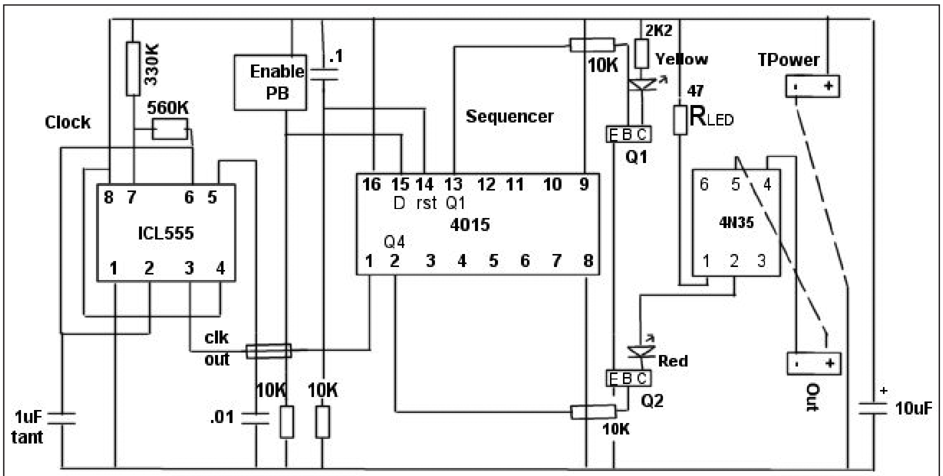


Figure 3A
All transistors 2N3904 or equivalent
MUST operate on 6 volts with RLED value shown

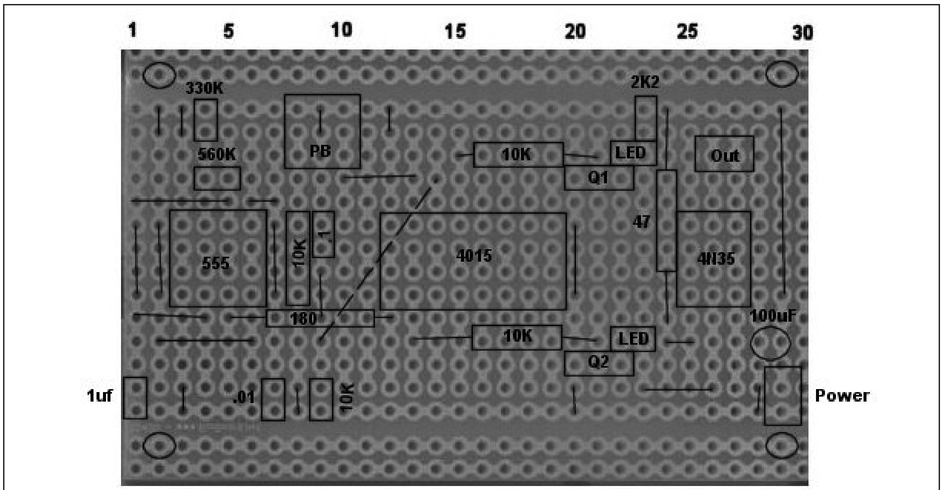


Figure 3B
top view

higher voltages, up to the 15 volt limit of CMOS. Use Ohm's Law to get 20 mA of current through the optocoupler / RLED combo. For the RLED value shown (47 ohm), a six volt supply is required. In practice, I took the circuit board from a cell phone car adapter and "floated" it with a diode to get the six volt regulated output. From a 24 volt source, I use a 12 volt series zener to drop the voltage to 12 volts to the cell phone car adapter. A one watt zener is sufficient.

Originally, I tried using a CD4060 for this, but found the circuit to be more predictable. The approximate component placement is shown below the circuit. Actual wiring may vary! The circuit has been in operation for over six months now and works great. If an actual pushbutton is used, the circuit may have potential electrostatic discharge susceptibility through the pushbutton to the D input. While I have not had issues with this, I have a small grounded loop near the pushbutton to reduce static discharge through it. Using a pushbutton with a grounded static shield is recommended.

Most parts (including the solderable breadboard) are from **AllElectonics.com**, though the 4015 is from Digi-Key.

Jim Lacenski
via email

ADvertiser INDEX

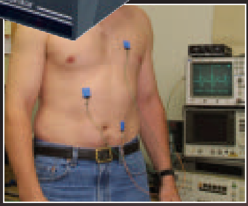
Beat The Heat...Build a Kit!

Ramsey Kits Are Always Neat, Even In The Dog Day Heat!

Electrocardiogram ECG Heart Monitor

- **Visible and audible display of your heart rhythm!**
- **Bright LED "Beat" indicator for easy viewing!**
- **Re-usable hospital grade sensors included!**
- **Monitor output for professional scope display**
- **Simple and safe 9V battery operation**

When we think summer, we normally think of vacations, traveling, and all the activities you have been waiting for all winter! And whether that includes hiking that new trail you've heard about or simply riding the new rides (and waiting in line in the scorching heat!) at Wally World, there WILL be physical exertion involved! While we are frequently reminded that February is national Heart Smart month, we think every month should be Heart Smart month. Heart Smart is a way of life, and certainly shouldn't be limited to one month a year. We kept that in mind when we designed the ECG1!

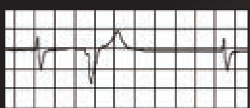


One of our engineers/guinea pigs, checking his heart!

Not only will building an actual ECG be a thrill, but you'll get hands-on knowledge of the relationship between electrical activity and the human body. Each time the human heart beats, the heart muscle causes small electrical changes across your skin. By monitoring and amplifying these changes, the ECG1C detects the heartbeat and allows you to accurately display it, and hear it, giving you a window into the inner workings of the human heart and body!

Use the ECG1C to astound your physician with your knowledge of ECG/EKG systems. Enjoy learning about the inner workings of the heart while, at the same time, covering the stage-by-stage electronic circuit theory used in the kit to monitor it. The three probe wire pick-ups allow for easy application and experimentation without the cumbersome harness normally associated with ECG monitors.

The documentation with the ECG1C covers everything from the circuit description of the kit to the circuit description of the heart! Multiple "beat" indicators include a bright front panel LED that flashes with the actions of the heart along with an adjustable level audio speaker output that supports both mono and stereo hook-ups. In addition, a monitor output is provided to connect to any standard oscilloscope to view the traditional style ECG/EKG waveforms just like you see in a real ER or on one of the medical TV shows!



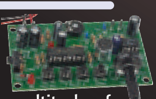
Look what I found!

On a personal note... See the display to the left? That's me! In between writing this montly ad copy, catalog copy, and plethora of other tasks here, I noticed some skipped beats in my pulse! An immediate cardiac check found I had something called Trigeminy, or PVCs that occur at intervals of 2 normal beats to one PVC! And I saw it with our ECG1 kit!

The fully adjustable gain control on the front panel allows the user to custom tune the differential signal picked up by the probes giving you a perfect reading and display every time! 10 hospital grade re-usable probe patches are included together with the matching custom case set shown. Additional patches are available in 10-packs. Operates on a standard 9VDC battery (not included) for safe and simple operation. Note, while the ECG1C professionally monitors and displays your heart rhythms and functions, it is intended for hobbyist usage only. If you experience any cardiac symptoms, seek proper medical help immediately!

ECG1C	**Electrocardiogram Heart Monitor Kit With Case & Patches**	**$44.95**
ECG1WT	**Electrocardiogram Heart Monitor, Factory Assembled & Tested**	**$89.95**
ECGP10	**Electrocardiogram Re-Usable Probe Patches, 10-Pack**	**$7.95**

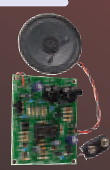
Digital Voice Changer



This voice changer kit is a riot! Just like the expensive units you hear the DJ's use, it changes your voice with a multitude of effects! You can sound just like a robot, you can even ad vibrato to your voice! 1.5W speaker output plus a line level output! Runs on a standard 9V battery.

MK171 Voice Changer Kit $14.95

Steam Engine & Whistle



Simulates the sound of a vintage steam engine locomotive and whistle! Also provides variable "engine speed" as well as volume, and at the touch of a button the steam whistle blows! Includes speaker. Runs on a standard 9V battery.

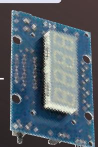
MK134 Steam Engine & Whistle Kit $11.95

5A PWM Motor Controller

This handy controller uses a pulse width modulated output to control the speed of a motor without sacrificing torque! Handles a continuous current of 5A and includes an LED to indicate speed, as well as an oversized gold heatsink! Also available factory assembled.

CK1102 5A PWM Motor Controller Kit $14.95

Digital LED Thermometer



This handy thermometer reads Celsius or Fahrenheit on an eye-catching .56" LED display! Based on the DS18B20 sensor and controlled by a PIC, it has a range of -67°F to 257°F (-55°C to 125°C) with a wired remote range of 325 feet!

CK127 Digital LED Thermometer Kit $29.95

Laser Trip Senser 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

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 the Saint, eats 2-8VAC or 9-12VDC, it's not fussy!

K2655 Electronic Watch Dog Kit $39.95

The Learning Center!

Beginners To Advanced... *It's Fun!*

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

For over 3 decades we've become famous for making electronics fun, while at the same time making it a great learning experience. As technology has changed over these years, we have continued that goal!

PL130A Gives you 130 different electronic projects together with a comprehensive learning manual describing the theory behind all the projects.

PL200 Includes 200 very creative fun projects and includes a neat interactive front panel with 2 controls, speaker, LED display and a meter.

PL300 Jump up to 300 separate projects that start walking you through the learning phase of digital electronics.

PL500 The ultimate electronics lab that includes 500 separate projects that cover it all, from the basics all the way to digital programming.

SP3B 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!

SM200K Move up to Surface Mount Technology (SMT) soldering, and learn exactly how to solder those tiny little components to a board!

AMFM108K We not only take you through AM and FM radio theory but we guide you through IC's. When you're done you've built yourself an IC based AM/FM radio that works great!

KNS10 With a reversible PEM fuel cell that combines electrolysis and power conversion into a single device you end up building your own fuel cell car! Learn tomorrows technology today!

KNS11 Learn alternative fuel technology while you build your own H-Racer car and refueling station!

KNS13 Convert ethanol alcohol to run a PEM fuel cell and watch it all work in front of your eyes!

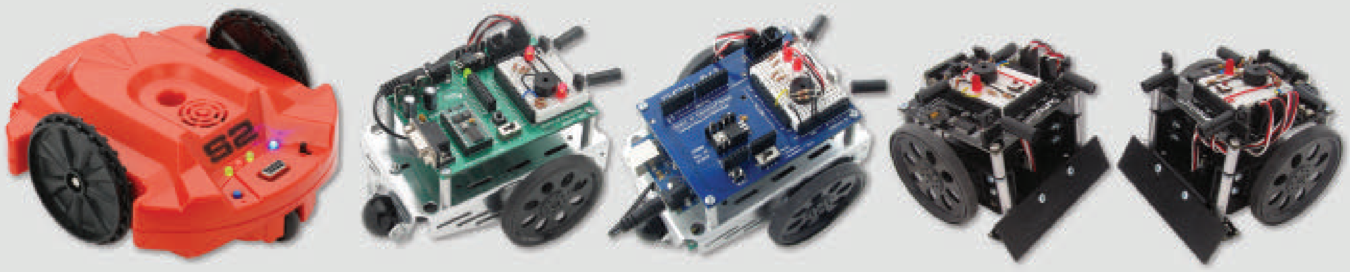
KNS1 A great beginner's kit for the dinosaur enthusiast in the family, young and old! A wooden hobby kit that teaches motor and gear driven operation that requires no soldering.

PL130A	**130-In-One Lab Kit**	**$39.95**
PL200	**200-In-One Lab Kit**	**$84.95**
PL300	**300-In-One Lab Kit**	**$109.95**
PL500	**500-In-One Lab Kit**	**$249.95**
SP1A	**Through Hole Soldering Lab**	**$9.95**
SM200K	**SMT Practical Soldering Lab**	**$22.95**
AMFM108K	**AM/FM IC Lab Kit & Course**	**$34.95**
KNS10	**Fuel Cell Car Science Kit**	**$82.95**
KNS11	**H-Racer & Refueling Station Kit**	**$144.95**
KNS13	**Bio-Energy Fuel Cell Kit**	**$129.95**
KNS1	**Tyrannomech Motorized Kit**	**$17.95**

ROBOTS

Roll or Fly - Autonomous or Remote Control - Controlled by BASIC Stamp, Propeller, even Arduino!

Parallax has been your source for robotics and education since 1998.

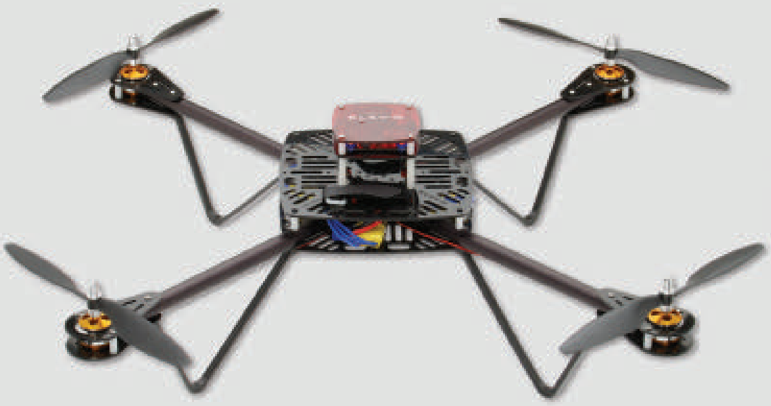


S2 (Scribbler 2) Robot *(#28136; $129.99)* Arrives pre-programmed and super simple for beginners. Reprogram with graphics or text, and hack it too if you like!

Boe-Bot Robot Kit *(Serial w/USB Adapter #28132; or USB #28832; $159.99)* Our best-selling education robot kit can teach you programming and basic robotics.

Robotics Shield Kit (for Arduino) *(#130-35000; $119.99)* Add your own Arduino module (not included) and build this versatile robot! *ROBOTC compatible.*

SumoBot Competition Kit *(#27402; $239.99)* Ready for an autonomous robot battle? This kit gives you two robots, complete programming instructions, and a ring for the ensuing competition.



ELEV-8 Quadcopter Kit *(#80000; $599.00)* - A flying robotic platform that is lifted and propelled by four fixed rotors. Large enough for outdoor flight, it has plenty of room for payload and attachments (up to 2 lbs). 8 hours assembly time. *Not recommended for beginners.*

Eddie Robot Platform *(#28992; $1249.00)* - Connect your own laptop and Kinect (not included) over USB and Eddie is up roaming the house in minutes. Compatible with Microsoft Robotics Developer Studio 4 (RDS 4), Eddie can navigate autonomously and see in 3D using the power of the Microsoft Kinect™.

See Parallax robots at **www.parallax.com/robots**.
Order online or call us toll-free at 888-512-1024
(Mon-Fri, 8am-5pm, PDT).

Friendly microcontrollers, legendary resources.™

PARALLAX INC

"ParallaxInc" on Twitter, Facebook, and YouTube