Surface Mount SOLDERING
Using The Tools You Probably Have On Hand Already

New Year's Resolutions
1. Learn about Power MOSFETs
2. Upgrade the flight recorder for my model rocket
3. Build another LED matrix clock
4. Get acquainted with my Proto Buddy
5. Add some "Flash" to my Design Cycle
6. Write a menu navigation system with C
7. SPICE up my simulations
Serial-to-Ethernet Converter

Available Now!
New 2008 Lower Pricing
PoE SitePlayer Telnet System
- $99.95
SitePlayer Telnet System
- $79.95
SitePlayer Telnet Modules
- $24.95

SitePlayer Telnet Module

Sample Browser Screenshot

Power Over Ethernet

- Configure and upgrade through web browser
- Access micro-controller or other RS232 serial device over The Internet
- Full RS232 handshaking

siteplayer.com
NetMedia Inc. Tucson AZ 85737 t.520.544.4567 f.520.544.0800 netmedia.com

- 50 to 11500 baud data rates
- DHCP, Link Local, and static IP Addresses
- TCP/IP and UDP data transfers
Sweet!

Introducing the MiniCore™ Series of Networking Modules

Smaller than a sugar packet, the Rabbit® MiniCore series of easy-to-use, ultra-compact, and low-cost networking modules come in several pin-compatible flavors. Optimized for real-time control, communications and networking applications such as energy management and intelligent building automation, MiniCore will surely add sweetness to your design.

- Wireless and wired interfaces
- Ultra-compact form factor
- Low-profile for design flexibility
- Priced for volume applications

MiniCore Module Development Kits
From $49
Limited time offer.

Buy now at: 1.888.411.7228 • wirelessrabbitkit.com

1.888.411.7228
wirelessrabbitkit.com
2900 Sphford Street, Davis, CA 95618

RABBIT®
The CUWIN combines a graphic display and touch interface with a high efficiency industrial controller.

- 7" / 10.2" wide color TFT display
- 800 x 480 resolution, 260K colors
- Touch panel
- SD card & Ethernet support
- RS232 x 2 / RS485 x 1 or RS232 x 3
- Mono Speaker and Stereo Jack
- Real time clock (Battery backup)
- Visual Basic and Visual C++ support
- USB 1/F (ActiveSync)
- Keyboard or Mouse support
- ARM3 32bit 200MHz processor
- 64MB FLASH, 64MB SDRAM
- Windows CE 5.0

CUWIN3200 $399 / Qty.1
CUWIN4300 $499 / Qty.1

COMFILE TECHNOLOGY
550 Pilgrim Drive Suite K Foster City, CA 94404
Call: 1-888-9CUBLOC (toll free) 1-888-928-2562
+ 30-Day Money-Back Guarantee
Visit www.cubloc.com to see more products!

New CUBLOC Core Modules Available!

CUBLOC

Use BASIC and Ladder Logic
80 ~ 200KB Flash Data Program memory
2 ~ 110KB Data memory
BASIC Speed : 36,000 instructions/sec
RS232, SPI, I2C, MODBUS

PLC on chip with BASIC
The CUBLOC's unique multi-tasking RTOS runs BASIC and Ladder Logic programs side-by-side, allowing you to combine the flexibility of BASIC with the industry-proven power of Ladder Logic.
Web-Enabled Catalog Breaks New Speed Record

Experience a Faster Way to Search for Components!

Jameco Electronics’ new catalog and enhanced Jameco.com website are two tools that are designed to work together to give electronic professionals faster access to the hottest components in the industry.

Color coded references throughout the catalog assist you in analyzing a wide range of brand choices (from franchise sourced to house brands to factory overruns) offering you more pricing options than you'll see anywhere else. Web codes allow you to quickly jump from catalog to website to view additional specifications, application notes, photos and more products. You'll find exactly what you're looking for without having to wade through hundreds of thousands of products on a complicated website or wielding a twenty pound catalog.

With a flip of the page or a click of the mouse, you have all the tools you need at your fingertips. Reach for Jameco first and order your catalog today.

Call 1-800-831-4242 or Visit www.Jameco.com/Speed for the ride of your life!
Features

50 Power MOSFETs  ■ By Gerard Fonte
Part 1 will cover the theory behind these useful devices that have a lot of overlooked features.

54 A Touch of Spice  ■ By Peter Stonard
This month, make discoveries about IC circuits that are well-known and learn more about the concepts of sub-circuits and macromodels.

60 How To: Surface Mount Soldering  ■ By Robert Doerr
Get over your fear of working with these tiny components.

Projects

34 Instrumentation for Model Rocketry
Incorporate additional instrumentation to a flight recorder to measure a rocket's rate of roll, relative altitude, and booster-stage separation detection. ■ By Michael Bessant

- Advanced Level

40 16x16 Tri-Colored LED Matrix — RealTime Clock for the Parallax SX52
This project allows the user to create their own scrolling display to show the time (or other messages) in multiple colors. ■ By Timothy Gilmore

- Advanced Level

44 The Proto Buddy
Take proto circuit building to the next level with this device that allows you to expand your existing board so you can create all kinds of circuits. ■ By Jim Stewart

- Intermediate Level

Columns

12 TechKnowledge 2009
Events, Advances, and News
Topics covered include reconnecting brain cells, a new unified storage appliance, don’t mix headphones and pacemakers, plus more.

15 Personal Robotics
Understanding, Designing & Constructing Robots
Robo resolutions for 2009.

18 Stamp Applications
Putting the Spotlight on BASIC Stamp Projects, Hints & Tips
Control from the couch — redux.

26 Q & A
Reader Questions Answered Here
High impedance preamp, audio to LED display driver, FM shutdown problem, more.

68 Getting Started With PICs
The Latest in Programming Microcontrollers
I'C communication.

72 The Design Cycle
Advanced Techniques for Design Engineers
Chatting up a thumbdrive.

80 Smiley’s Workshop
An AVR C Programming Series
A menu navigator.
**SERIAL TO ETHERNET SOLUTIONS**

$69  **SB70**  
2-port serial-to-Ethernet server with TTL support  
Qty. 100

$139  **SB72EX**  
2-port serial-to-Ethernet server with RS-232 & RS-485/422 support  
Qty. 100

$269  **PK70EX232**  
4-port serial-to-Ethernet server with RS-232 support  
Qty. 100

Simple Ethernet connectivity for serial devices  
Works out of the box - no programming is required  
Customize to suit any application with low-cost development kit

**Features:**  
10/100 Ethernet  
TCP/UDP mode  
DHCP/Static IP Support  
Data rates up to 921.6kbps  
Web-based configuration

Need a custom solution?  
NetBurner **Serial to Ethernet Development Kits** are available to customize any aspect of operation including web pages, data filtering, or custom network applications. All kits include platform hardware, ANSI C/C++ compiler, TCP/IP stack, web server, e-mail protocols, RTOS, flash file system, Eclipse IDE, debugger, cables and power supply.

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

NetBurner  
Networking in One Day!

freescale™  
Alliance Member

http://nutsvolts.texterity.com/nutsvolts/200901/templates/pageviewer_print?pg=1&p...  
12/02/2009
Building a Home Recording Studio

Want to test the extent of your knowledge of applied electronics? Then try designing a home recording studio. Thanks to the affordable music recording and playback devices you can have fun creating your own music. Moreover — from an electronics enthusiast’s perspective — a home recording studio is a place where you can apply your creativity. However, designing and then getting the most out of a home recording studio requires a considerable understanding of electronics, from power supplies and digital signal processing to acoustics.

Options

Various options available for creating a home recording studio range from a pocket digital recorder and built-in microphone in your bedroom to a dedicated, acoustically treated room with the latest computer hardware and software tools, professional microphones, mixers, and signal processors. Budget is obviously a consideration, as is available space.

A popular, low-cost option ($350 and up) is a studio in a box. This is an all-in-one, stand-alone recording appliance, such as the 64-track Boss BR-600 Digital Recorder, with built-in microphone, drum machine, and signal processing effects. A laptop or desktop computer with an analog-to-digital interface for capturing sound and associated software, as well as microphones and preamps, is a modest cost solution. A computer offers more flexibility and room for growth than a studio in a box.

Power

I’ve learned — the hard way — that the first thing to be aware of is the seemingly odd standards of the music industry. For example, the power supply polarity used by effects pedals and other audio peripherals is often ‘reversed.’ That is, it’s common practice with audio gear outfitted with coaxial power connectors to have the outside of the jacket positive and the inside negative — exactly the opposite of most consumer electronics. I’ve
mistakenly plugged in a standard power brick to a preamp, which resulted in a fried one. You can avoid this mistake by clearly marking your music power supply bricks to distinguish them from your standard polarity bricks.

**Computer Tools**

Another industry standard that you’ll discover is Digidesign’s Pro Tools. This software and hardware combination enables you to record and then manipulate sound on your PC. There are numerous software options for both PCs and Macs. For example, I use Logic Studio on a Mac. However, Pro Tools is the entrenched standard. Regardless of your computer hardware and software, you’ll need lots of disk storage and, more importantly, high throughput. Think fast terabyte drives. Expect to use 0.5 GB or more for an eight-track recording of a 10 minute song.

**Cables**

Cables and connectors for recording are specialized. When selecting cable, think thick copper to minimize resistance losses. Conversely, when you select high-impedance microphone and instrument cables, think shielding for noise reduction and minimum capacitance per foot to minimize high frequency loss. Bulky XLR and old-fashioned 1/4 inch still handle most of the audio signals in a home or professional recording studio. While on the topic of impedance matching, consider that microphones, guitars, and other instruments typically present a high-impedance output (15k–1M ohm) which is much higher than the typical low-impedance line input to an analog-to-digital converter. You’ll need a preamp or a direct box to compensate for the impedance mismatch and assure the signal level is appropriate for your input device. A direct box can be as simple as a transformer with input and output connectors.

**Signal Processors**

Stand-alone, floor, and rack-mounted signal processing options — filters, amplifiers, and various effects — can help you achieve the tone you demand. At a minimum, you should consider an eight channel virtual or hardware mixing board. A mixer allows you to adjust levels of various inputs and combine them into a single mono or stereo track. I prefer the feel of a real hardware mixer over a virtual mixer defined in software.

**Ground**

One of the headaches of connecting a half-dozen amplifiers, effect boxes, and computers together is that you have to pay attention to ground loops. Often, the only practical way to remove 60 Hz hum from the audio signal is to lift the ground on one or more devices in the signal chain. This can be accomplished by using a 3-to-2 pronged adapter. Some gear has a built-in ground lift switch. There are obvious safety issues associated with lifting the ground in an AC powered device. In short, don’t allow your body to complete a ground fault circuit.

Continued on page 10
Information

You can be the best circuit designer in the world and find yourself lost in the vocabulary and sheer volume of choices in the home recording world. Prepare to spend some time in your local music store or, better yet, with someone who has a system up and running. And avoid the common misstep of designing your home recording studio one component at a time. You could run out of money before you're through and end up with a mismatched, expensive system. Better to configure the entire system and then look for the best price.

Of course, you can save money—and have fun—by making your own cables, effects boxes, and signal monitoring devices.
Low Power, 32-bit Flash MCU with High-precision 16-bit ADC
Features 88dB SN-ratio, low power and ultra-small package

Renesas Technology
No. 1* supplier of microcontrollers in the world
Introduces the H8SX/1622 microcontroller with the H8SX CPU core at 50MIPS, built on advanced low power technology, and available in ultra-small packages (9mm x 9mm).
The device contains a 16-bit ΔΣ A/D converter with 88dB SN-ratio and a conversion time of 91.5μs. The A/D converter unit provides fully programmable gain and offset stages supporting both single-ended and differential analog inputs.
Full system integration is provided through high-speed UART, SCI, and FC serial channels assisted by direct memory access control (DMA and DTC units), and an external parallel data bus.
Low power consumption, with typically less than 1mA/MHz full-run operation and 4μA standby, makes this device ideal for medical monitoring, high-precision smart sensors, and digital consumer products.

H8SX Product Lineup

Top Reasons to Select H8SX/1622
- 32-bit CISC CPU core at 50MHz
- 256KB on-chip zero wait-state Flash memory
- 32KB SRAM
- Deep Software Standby consumes only 4μA (typical)
- Operation at 35MHz consumes 30mA (typical)
- 6ch 16-bit ΔΣ ADC with single and differential inputs
- 8ch 10-bit ADC (SAR) and 2ch 8-bit DAC
- Advanced data management (DMA and DTC)
- High speed serial interfaces (UART, SCI, FC)
- 16-bit and 8-bit timers and 16-bit Pattern Generator
- External bus to give connectivity to SRAM and Flash memory
- Small 9mm x 9mm LGA-145 and QFP-144 packages

*Source: Gartner "Semiconductor Applications Worldwide Annual Market Share Database" Hiroshi Shimizu, 27 March 2008, G000218 # This is 2007 ranking

Get Started Today -
Go online and register to access special promotions on starter kits.*
www.america.renesas.com/ReachH8/e
* Only available to customers in North and South America

Renesas Technology Corp.
BY JEFF ECKERT

ADVANCED TECHNOLOGY

RECONNECTING BRAIN CELLS

Let's say you have a toilet that works fine until the chain that connects the flush handle to the flapper breaks. One way to fix it would be to hook up a motion sensor and a transmitter to the handle such that whenever the system detects movement, it transmits a signal. This signal is picked up by a receiver that triggers a power supply and a solenoid in the tank which lifts the flapper and invokes a flush. Or, you could just replace the chain. Duh!

Now let's say you have a human being whose leg no longer functions because of spinal cord damage. One way to address that is to pick up some brain waves, decode and convert the neural signals so they can be processed by a computer, and have the computer send signals to a robotic prosthesis device that creates the desired movement.

Or, you could just reconnect the brain and the leg. Duh!

Enter Drs. Eberhard Fetz and Chet Moritz, both of the University of Washington (www.washington.edu), who reasoned that, given that spinal injuries leave nerve cells in the

brain and the muscles unimpaired, it might be possible to reconnect the muscles to the motor cortex and let them start communicating again. Their study, recently reported in the journal Nature, indicates that it could very well work. In research conducted at the Washington National Primate Center, they numbed some monkey's wrist nerves with a local anesthetic. The doctors then provided direct, artificial stimulation from arbitrarily chosen motor cortex cells to multiple muscles. Oddly enough, the monkeys learned to flex and extend their wrists to play video games.

Despite the nerve block, the monkeys were able to control the contraction strength of their wrist muscles to match a set of targets on a computer screen, and they got better at it as they learned to control the neurons. The odd thing is that even neurons usually unrelated to wrist movement could be used to control the muscles, which hints that stroke victims someday may learn to use undamaged brain areas to restore lost function.

The study was conducted using lab instruments, but Fetz and Moritz also built a portable device from off-the-shelf components to convert signals from the motor cortex neuron cells into stimuli. The box-size device runs off AA batteries, and further miniaturization should be relatively simple. Considerable refinement is needed, but with support from the National Institutes of Health (NIH) Neurology Institute, the concept looks promising.

ANOTHER LEAP TOWARD THE SUN?

One of the obstacles to practical solar energy is, of course, the relative inefficiency of the panels. Part of the problem is that existing panels absorb only about two thirds of the sunlight that strikes it, meaning that about a third of the power potential slips away unharvested. However, some researchers at Rensselaer Polytech (www.rpi.edu) have demonstrated a new antireflective coating that allows a panel to (a) make use of the entire solar spectrum and (b) gather light evenly and equally from all angles, thus eliminating the need to constantly reposition the panels to track the sun's position in the sky. The result is an absorption rate of 96.21 percent.

According to Prof. Shawn-Yu Lin, project leader, typical antireflective coatings are engineered to transmit light of one particular wavelength. The new coating stacks seven of these layers such that each layer enhances the antireflective properties of the layer below it. These additional layers also help to "bend" the flow of sunlight to an angle that augments the coating's antireflective properties, so each layer helps to capture any light that may have otherwise been reflected off of the layers below it.

The layers are made up of silicon dioxide and titanium dioxide nanorods positioned at an oblique angle so each layer looks and functions similar to a dense forest where sunlight is captured between the trees.

There was no speculation about how soon the panels might be
commercially available, and we have to keep in mind that other promising dye and coating formulations ultimately could not adequately withstand continuous exposure to sunlight. But this could be a significant step forward.

COMPUTERS AND NETWORKING

NEW UNIFIED STORAGE APPLIANCE

Bill as the "world's first unified storage appliances," Sun Microsystems recently introduced the Sun Storage 7000 family of "open storage" devices, designed to "radically simplify storage management and problem solving at breakthrough speed and scale." According to Sun, "These new systems radically simplify the way information is managed and can deliver up to 75% cost/performance savings to customers. This is the biggest thing to happen to storage in decades." Three models are available: the 7110 (ultracompact model with 2 TB of storage); the 7210 (mid-range storage featuring up to 48 TB of capacity); and the 7410 (support for up to 0.5 PB of capacity that includes support for read and write optimized SSDs and Sun's Flash Hybrid Storage Pool technology).

All of the 7000 systems include comprehensive data services at no extra cost, such as snap/cloning, restore, mirroring, RAID-5, RAID-6, replication, active-active clustering, compression, thin provisioning, CIFS, NFS, iSCSI, HTTP/FTP, and WebDAV. For details, visit www.sun.com. But check your bank account first. The devices start at $10,000 and top off at $89,490 for the 7410 in a clustered 12 TB configuration.

RUGGED MILITARY COMPUTER INTRODUCED

There may not be much going on in the consumer PC market right now, but the new MQS4U-20A from Chassis Plans (www.chassis-plans.com) is a pretty interesting beast. It's a rugged military-grade rackmount computer system based on an ultra high-performance Tytan Thunder n6550EX S4989 Quad AMD Opteron® processor motherboard and an EVGA e-GeForce 9800 GX2 graphics processing unit, which offers two DH6 outputs with up to 2560 x 1600@60 Hz graphics.

Intended for applications where a short depth, light weight, and sturdy rackmount system are required, the system is optimized for installation in transit cases or mobile installations such as aircraft and other vehicles. It meets all appropriate Military standards for shock, vibration, and environmental specifications, so you can beat it around all you want.

The machine can be configured with up to 128 GB of 667 MHz DDR2 RAM and two 1 TB enterprise-level drives. The single or dual redundant 1,200W power supplies are kept cool by four 120 CFM fans.

All this is wrapped in a 5052 high-strength aluminum case. As you might expect, no list price was offered. But if you're interested, you can get a quote via the company's website.

CIRCUITS AND DEVICES

ANY RATE, ANY OUTPUT CLOCK GENERATOR

Silicon Laboratories, Inc. (www.silabs.com), the Austin-based producer of high-performance, analog-intensive, mixed-signal ICs, has introduced a family of clock generators and buffers that provide a high level of frequency flexibility. Based on the company's MultiSynth technology, the Si5338 is capable of synthesizing any frequency from 0.16 to 350 MHz and select frequencies to 700 MHz on each of the device's four differential outputs, thus simplifying timing architectures by replacing four discrete phase-locked loop (PLL) devices with a single IC.

Silabs claims best-in-class performance and integration while shortening design cycles for applications such as next-generation communications equipment, wireless base stations, broadcast video, test and measurement, and data acquisition. At 1 ps RMS random jitter typical, the Si5338 can simultaneously generate low jitter clocks for a variety of ICs, including processors, FPGAs, ASICs, memory, and physical layer transceivers. The device generates four differential or eight single-ended outputs per device, eliminating the need for external clock distribution buffers. In addition to frequency, each output clock is independently configurable in terms of supply voltage (1.5, 1.8, 2.5, 3.3 volts) and signal format (LVPECL, LVDS, CMOS, HCSL, SSTL, HSTL), eliminating the need for external level translators, and reducing BOM cost and complexity.

If you don't need the programmability provided by the Si5338, there is the option of a broad line of Si5334 pin controlled clock generators for Ethernet, Fibre Channel, PCI Express, T1/E1, broadcast video (HDTV),

January 2009 NUTSVOLTS 13
and SONET/SDH OC-3/12/48 applications. Packaged in a 24 lead 4 x 4 mm QFN package, the Si5338 and Si5334 devices are available at prices starting at $9.31 in 10,000 piece quantities, depending on output frequency.

**MEMS MIKES FOR CONSUMER APPS**

Early in 2007, Scotland’s Wolfson Microelectronics (www.wolfsonmicro.com) acquired a local MEMS design company, Oligon Ltd., which was integrated into Wolfson’s plans to add more hardware and software technologies to its audio processing, A/D, and D/A technology. In October, this emerged as a new family of silicon microphones, beginning with the WM7110 and WM7120. These are compact, high signal-to-noise ratio (SNR) analog microphones for use in consumer applications that require low power consumption and high signal quality. Using Wolfson’s CMOS/MEMS membrane technology, the new devices promise high reliability and performance in a miniature, low profile package. Wolfson is also offering enhanced WM7110E and WM7120E versions, which are said to be the first MEMS silicon microphones to deliver a sensitivity tolerance of ±1 dB. The WM7110 (4.72 x 3.76 x 1.25 mm) and ultra-compact WM7120 (3.76 x 2.95 x 1.10 mm) devices typically consume only 160 mA, making them ideal for portable applications such as mobile phones, portable media players, digital still cameras, video cameras, navigation devices, and noise-cancelling headphones.

Both offer 62 dB SNR (A-weighted) performance and total harmonic distortion of 0.5% max at 100 dB SPL. Sample quantities should be available by the time you read this. Prices start at $1.64 in quantities of 1,000.

**INDUSTRY AND THE PROFESSION INTEL RECOGNIZED BY EPA**

An award ceremony hosted by the US Environmental Protection Agency (EPA) late in October, Intel Corporation (www.intel.com) received a 2008 Green Power Leadership Award and was also named a Green Power Partner of the Year. The EPA recognized the company’s voluntary efforts to address climate change through green power purchases and its impact in advancing the renewable energy market.

Early in the year, Intel signed a multi-year commitment to purchase more than 1.3 billion kWh of renewable energy certificates (REC) per year, which will meet approximately 47 percent of its purchased electricity use. The commitment put the company at the No. 1 spot on the EPA Fortune 500 Green Power Partners list.

The EPA estimates that Intel’s purchase has the equivalent environmental impact of taking more than 185,000 cars off the road or avoiding the amount of electricity needed to power more than 130,000 average American homes.

**DON’T MIX HEADPHONES AND PACEMAKERS**

According to a recent study conducted by Harvard Medical School's Beth Israel Deaconess Medical Center (www.bidmc.org), if placed within an inch of pacemakers and implantable cardioverter defibrillators (ICDs), may interfere with their operation.

When exposed to magnets, these devices automatically pace, sending low energy signals to the heart to make it beat. Defibrillators — which treat slow and dangerously fast heart rhythms — send either low or high energy signals to the heart. However, ICDs placed in the vicinity of magnets may temporarily stop looking for abnormal heart rhythms.

The bottom line? If your ears happen to be attached to your chest, be careful. You may have problems that extend beyond the annoying issues of how to hear through your raincoat and keep your eyeglasses from falling off.
ROBO RESOLUTIONS 2009

MY NEW YEAR'S RESOLUTION IS 1650x1280 ;)
It's a brand new year and a great opportunity to make some Robo Resolutions! Regular readers of this column are familiar with articles documenting my first hand experiences with the robotic, artistic, and sometimes silly contraptions I am apt to find myself in the thick of making or displaying. As this is the first column of the new year, I decided to take the opportunity to offer some New Year's Resolutions based on first-hand experience. (Remember, “experience” is what you get just after you need it!)

RESOLUTION #0: BUILD SOMETHING

Though I've emphasized this to many of the regulars that attend meetings of The Robot Group here in Austin, TX, I can't say it enough: Build something! You'll discover that Building = Experience + Learning! It's a completely different thing to build something yourself than it is to just read about a project or observe someone else making it. Until you actually build a device, you have nothing to fix, troubleshoot, or improve.

Once you have a prototype, you will inevitably come up with ideas and optimizations that you may carry to a new version of the same project or, in some cases, to an entirely different project.

As you assemble your parts to test your theories, things can get very complicated very quickly. These challenges encourage you to be imaginative and creative in order to deal with problems as they arise. This is exceptionally good exercise for your brain and the lessons learned from overcoming these real-world problems will stay with you a lot longer than ones you've simply read about.

RESOLUTION #1: TEACH SOMETHING!

The old saw goes "the best way to learn something is to teach it." My first hand experience in this area bears this out. Teaching IS learning! If there's something about which you are curious, you may find the best way to become well-versed is to teach that subject to others. Though your skills may not be PhD level, you probably know more about something than someone (especially kids, though they are sure to think otherwise!).

FIRST robotics and other student-centric competitions are usually dying for mentors, judges, or tutors to help with events. Don't be afraid to call and offer to help at these events. The lives you touch and the people you meet can change your life.

RESOLUTION #2: CHALLENGE YOURSELF!

Get out of your comfort zone! If you're an analog electronics person, it might be time to look into dipping a toe into some digital circuits. Or, if you're really good with software but are a little intimidated by mechanical linkages and servo motors, take a stab at building a software-controlled mechanical project. There is something particularly engaging about making electronics that move!

Giving yourself a task that is outside your comfort zone is a good way to keep your skills sharp and to increase your knowledge — in some cases, your professional value, as well.

Find an area where you have an interest but little or no practical experience and take a stab at it. You can do this by finding others you can "swap" skills with or by taking a course at a local community college. The idea is to expand your knowledge and explore your talents. For example, with a newly

Vern Graner at Dorkbot Austin in 2006 giving a presentation on the (then) forthcoming RoboSpinArt machine — how it should work, the work accomplished, the problems encountered, and how they were solved.

January 2009  NUTSVOLTS  15
If you want time, you must make it. For me, time is one of the hardest things to make. In this regard, I'm my own worst enemy as I give my time away as fast as I make it!  

In the last year, I had to make some hard choices about the amount of time I had and the number of things I wanted to do. I discovered that there is really only so much "me" to go around. In order to give your all to a project, you first have to have an inventory of "you" from which to draw.

If you have time challenges to deal with, your most powerful weapon can be one word: no. This is so much harder than it appears on the surface. I know I could fix that small stereo amplifier. I know I could find and replace the bad light in that string of rope lights. I know I could help judge the Science Fair at the local elementary school. Though we may be capable (as in have the skill), to perform a task, what we probably don't have is available hours. In addition, you want to make sure you don't give away so much of yourself that you don't have any time left for you!

For example, I discovered it's better to give 100% on one project than to only be able to give 50% on two projects. Measure the values and try to balance your time. Examine your motivations. Do you feel like you have to fix that rope light string just because you can, or do you want to do it as a learning exercise? There is a critical difference. If the goal of the work is to learn or keep your diagnostic chops up, then by all means, get to work. However, if the goal of the exercise is to have lights around the bake sale table before Friday (and you know darn well the Mrs. will be in a tizzy if the lights don't work!), it might be better to put the broken string into the junk parts bin and just buy a set of lights to get the job done.

RESOLUTION #5: KNOW WHEN TO SAY WHEN

It's a masterpiece — the embodiment of your vision. It's been months in the making and now exhibits the sheen of your high-gloss polishing. Of course, it's not really done yet, but it's close. Just a few changes left to do. A bit of optimizing ... add in that new bit of circuitry and another re-work of the software to support it, then it'll be ready. Except for that one more thing you thought of doing while you were re-working the software.

The preceding results are what I refer to as "The Perpetually Almost-Done Project." I have a few of these going (some I can see from where I sit right now, as a matter of fact).

I'm not saying don't improve on projects or designs as you go, just know it's important to remember the scope of the work and make sure you put the real good ideas down for the next version of the device and just do the needed stuff now. At some point, you just have to call your current iteration complete. Give it a revision string, i.e., MY PROJECT V1.0 and save those great ideas for MY PROJECT V2.0.

RESOLUTION #6: STAY HEALTHY

When our focus is centered on the project, when we're concentrating on solving a problem, or stressing to meet a deadline, we tend to forget that our bodies are the ultimate machine. Like any machine, we need to pay attention to operational safety and maintenance. In many cases, electronics work is sedentary by nature, sitting at a workbench, using the computer, or even reading an exciting electronics magazine. :-) All tends to lead to chair-butt syndrome (a.k.a., the middle-age spread).

Stretch, get up, get outside, see some sunshine, kick off work well before midnight or heck, take the entire night off and go to bed early. Tomorrow will wait. Lay off the junk food and have a salad now and then. Do all the things the old wife's tales (and sometimes the old wives!) tell us. It really is important to watch out for yourself since you
only get one body per lifetime. On a side note, I checked with my original manufacturer and it seems there’s no warranty on hair follicles, dang it!

Though we may preach it to others, sometimes we don’t always pay attention to safety precautions. For example, do you wear safety glasses while soldering, cutting, or doing any work that can create flying debris? Do you have an exhaust fan to take fumes from solvents and soldering out of your breathing area? Do you eat at your workbench and, if so, do you handle solder or tools that have come in contact with industrial cleaners and the like? Do you wear gloves when using cleaning agents, lead based solder and other known hazardous substances? Many solvents are trans-dermal and act as a vehicle to take some toxins right through your skin and into your blood stream. I learned this the hard way when I noticed a strange taste in the back of my throat after using a

trichloroethane-soaked rag to clean up a couple of copper clad circuit boards!

RESOLUTION #7: SHARE YOUR WORK

Document your projects. Take pictures of your work and then take a moment or two to write up descriptions and schematics. If you write software, spend a few moments going through and adding comments. This makes it easier for others to “get” what you’re doing and gives you aids to use for talking about it.

Having documentation allows you to engage with other folks that have similar interests. Also, having to explain a project to someone can help to solidify the concepts in your mind as articulating what it is you’re doing is an excellent way to reinforce and clarify project ideas. I’ve found that when I run into a problem on a build, sometimes just going over the problem verbally with someone else can help you see a possible solution that eluded you.

Consider writing about your project for a magazine. For those of you not otherwise aware, Nuts & Volts and their sister publication, SERVO Magazine, will pay you for well written articles! This is a great way to help finance your hobby.

The articles you read in these pages are (for the most part) written by amateur writers and fellow hobbyists that have taken a few moments to get their thoughts down on paper (well, on disk). If you’ve already created a blog or if you have a nice collection of photos on one of the many picture-sharing websites, you may already have everything you need to write up an article, share it with your fellow readers, have a nice bullet point for your resume (i.e., “Internationally published technology magazine article author”), and get PAID for doing it! Check the resources for a link to the Nuts & Volts writers guidelines page.

LETS GET STARTED!

I figure seven is a lucky number so I guess I’ll stop the resolutions here. I’m really looking forward to writing about some of the fun stuff I have planned in the coming months. I also want to invite you to let me know if there are topics or items you’d like to see reviewed, built, or covered in an article. Suggestions, comments, and observations are quite welcome. As always, you can reach me via email at vern@tixs.com. NIV

Vern Graner (seated) discusses a problem in the firing sequence of the Ponginator Software with André LaMothe at Maker Faire Austin 2007. This discussion led directly to a rewrite of the way the guns fired which eliminated motor positioning errors.

RESOURCES

- The Robot Group
  www.robotgroup.org
- PROBOTIX FireBall v90 CNC router
  www.probotix.com
- André LaMothe NURVE software
  www.xgamestation.com
- FIRST Robotics
  www.usfirst.com
- RoBoMagellan — www.robo
games.net/rules/magellan.shtml
- Maker Faire
  www.makerfaire.com
- THE PONGINATOR MK3
  http://makerfaire.com/pub/e/2185
- Nuts & Volts Writers Guidelines
  www.nutsvolts.com/writers_
guidelines.php

January 2009

http://nutsvolts.texterity.com/nutsvolts/200901/templates/pageviewer_print?pg=1&p...
CONTROL FROM THE COUCH — REDUX

Some time back, I wrote about being a single guy, drinking milk right out of the carton, and having a bunch of IR remotes next to my favorite easy chair. Well ... I'm still single, still drinking milk right from the carton and — like every real man — still loving my remotes! That earlier article had to do with decoding the Sony IR Control Systems (SIRCS) protocol with a BASIC Stamp. With the SX and SX/B I think it's time to revisit SIRCS decoding and even couple it with serial I/O so that we can enable dual-mode control (IR plus serial) or have the ability to use our project as an IR-to-serial translator.

SIRCS PROTOCOL REVIEW

The SIRCS protocol uses pulse-width encoding transmitted over a modulated IR carrier. To get the signal from an IR beam into the SX, we can use a standard demodulator like the Panasonic PNA4602M. The output from the PNA4602M is a low-going pulse stream as shown in Figure 1.

Many consumer electronics devices use the 12-bit SIRCS protocol and since this version is so commonplace, it is what we will work with here. The stream starts with a 2.4 millisecond start bit which is followed by 12 data bits — sent LSB first — that contains seven bits for the command (channel number, volume control, etc.) and five bits for the device (e.g., TV, VCR, camera). The stream shown in Figure 1 (%%0001_0010001) corresponds to pressing the zero key on my television remote.

Decoding the SIRCS stream is actually quite easy. We start by watching for a low-going edge on the detector input pin and measuring the period that this pulse stays low. If the pulse is about 2.4 milliseconds, we have a valid start bit and can drop into a loop to measure and decode the following 12 data bits. A '1' bit has a width of 1.2 milliseconds and a '0' bit has a width of 0.6 milliseconds. Each pulse is padded with a 0.6 millisecond dead space which gives us a lot of time to take care of any inter-bit processing.

SIRCS DECODING: SX/B STYLE

For programs that don't use interrupts and can tolerate being "blocked" until an SIRCS code is received, the following function takes care of the grunt work for us and even splits the device (five bits) and command (seven bits) codes into separate bytes within the same word:

```
FUNC GET_SIRC
    width VAR tempBl
    IDX VAR tempBG
    irWork VAR tempBl

    DO
        width = GET_IR_PULSE
        LOOP UNTIL width > 216
        irWork = 0
        FOR tIdx = 0 TO 11
            irWork = irWork >> 1
            width = GET_IR_PULSE
            IF width > 108 THEN
                irWork.11 = 1
            ENDIF
        NEXT
    ENDIF
```

You'll see that within the function I've aliased my temporary variables to make the code easier to read. With SX/B 2.0, we could use locals, but I tend not to do this for simple programs that have plenty of variable space.
reason is that when using local variables we add a lot of compiler-generated overhead to handle them as they are, in fact, elements of a special array (the stack).

At the top of the function, we look for a start pulse by calling GET_IR_PULSE which is simply a shell for the SX/B PULSIN function. PULSIN returns a value in 10 microsecond intervals. The upper limit of our measurement is 700 usec (280 usec + 300 usec + 20 usec). It is important to account for the division of the system clock by 5, which means that we need to be a little bit more accurate. For example, if we are in the middle of the usec plus 500, the clock is not really at 500 usec but at 475 usec. We don't want to measure the start pulse at 475 usec because we want to count the first 500 usec, but we don't want to measure it at 500 usec because we want to count the last 500 usec. We want to measure it at 475 usec and then add the difference to the count.

Once a valid start pulse has been detected, we'll clear a word variable (0x00) that will hold the SIRCS code bits.

A loop is set up to measure and decode 12 bits. If the measured pulse is 90% of 1.2 milliseconds, then we set the bit to 1. We don't want to worry about zero bits because the right shift operator used to align the variable for the next bit pads with zeroes. Remember that the bits arrive LSB first, so we shift the output value right and stuff the new bit into the MSB position (bit 11). Using this technique, the last bit that arrives (bit 11) lands in the correct location.

For convenience, I like to split the device code and command code into separate bytes within the word that holds the SIRCS result. To do this, we need to shift the MSB (which holds the device code) one bit left and then move bit 7 of the LSB (command code) to bit 0 of the MSB. This is accomplished with a piece of embedded assembly to keep things trim. The first line copies bit 7 of 0x00_LSB to the Carry bit. The reason for this is that the low nibble of the high nibbles will shift the carry bit into the target byte. The second line takes care of shifting the MSB and getting its bit 0 properly in place. That last line clears the old copy of the device code bit 0 from bit 7 of the command code byte. The final value returned to the caller holds the five-bit device code in the MSB and the seven-bit command code in the LSB. By separating the device and command codes, our programs can more easily handle multiple remote types.

A GENERIC IR CONTROL PLATFORM

To put all my remotes to use, I decided to build a flexible platform that gives me the ability to receive IR, control eight outputs, and send and receive serial data over an RS-232 connection. Having learned my lesson from past projects and wanting to give myself lots of options with the board, I've designed it to be selectively-stuffed based on what I want to do with a particular application. As you can see in Figures 2, 3, and 4, there is nothing complicated about this circuit. In Figure 5, you can see that the outputs are pretty flexible — again, not all of these parts need to be stuffed into a given board. I've designed in the

ability to have servo headers for TTL I/O, and a ULN2803 for moderate-current outputs; both have output voltage selection. I thought that since I'm spending $60 with ExpressPCB I really should be able to use the board to do a lot of things — with this one, I can. You'll see that even though the RB pins are not defined on the schematic that I added pads to the PCB, anyway... just in case.

Figure 6 shows my completed prototype. I decided to use a Parallax servo extender cable to attach the IR decoder to the board. Note, though, that the pinout of the IR demodulator does not match the cable; no worries, this is easy to fix. You can use a pin or nail to unlock the female crimp sockets from the plastic shell and reinstall them in the proper position. If you look closely, you'll see that I've swapped the red and black wires to match the IR demodulator.

JR SNIFFER DEMO

Okay, let's start easy. Let's finish up a program that decodes the SIRCS stream and displays it on a terminal. We've already gone through the process of decoding, so it's just a matter of formatting the IR code for serial output:
After initialization, the pull-ups are activated for unused pins; we don't have to enable the pullups for port RC as the ULN2803 input circuits act as pull-downs. The transmit pin is brought high to put it into the idle state (for true mode) and a short delay is inserted to let the receiver know to get ready. A call to TX_STR with Banner as the target clears the terminal screen, and prints a banner and header as you can see in Figure 7.

At Main, we drop into a loop that waits for an IR code and then displays its constituent parts in binary format under the on-screen headers. Remotes repeat commands every 45 milliseconds or so if a key is held down, so a 250 millisecond delay is inserted to prevent the screen from getting filled up too quickly.

This is a simple program but important to analyze the characteristics of the remote we want to use. For example, I have a project coming up that will transmit SIRCS codes to a digital camera. By using this program, I was able to decode the buttons on the camera's remote; in fact, the first 10 lines on the screen in Figure 7 are from the numeric buttons on my TV remote; the last three correspond to the shutter, telephoto, and wide angle buttons on my camera remote. As you can see, the TV remote and the camera remote have different device codes.

**TAKING CONTROL**

Now that we can see the codes produced by the various buttons on the remote, we can write a simple control application. In my case, I'm going to use this to control high brightness LEDs with the ULN2803 Darlington array. Here's how my little control program works:

- [9] All outputs on
- [0] All outputs off
- [Ch+] Rotate one output right
- [Ch+] Rotate one output left
- [Vol+] Rotate all outputs right
- [Vol+] Rotate all outputs left

There's no real magic to this program, so we'll just look at one or two elements. For example, here's the bit that looks for a numeric button between 1 and 8 and toggles the selected output:

```plaintext
IF cmdCode > Ch1 THEN
IF cmdCode <= Ch8 THEN
mask = 1 << cmdCode
Control = Control XOR mask
DELAY_MS KeyDelay
GOTO Main
ENDIF
ENDIF
```

If the command code falls within the valid range of numeric keys, then a mask is created from the command code. We're helped here by the key assignments: key #1 has a command code of 0; key #8 has a command code of 7, so this works perfectly with the left shift operator to create a channel mask. When we XOR the mask with the control port (RC), the selected output will be toggled. To prevent crazy toggling, a short delay is inserted to allow the button to be released. You may want to experiment with this delay.

Here's the code segment that works with the [Vol+]
key to barrel shift all bits right:

IF cmdCode = VolUp THEN
AGM
MOVB C, Control.0
RR Control
ENDAGM
DELAY_MS KeyDelay
GOTO Main
ENDIF

With a barrel shift, we don't lose the end bit to the "bit bucket" — it gets wrapped around to the opposite end. To do this, we forego the use of the SX/B shift operators for a block of Assembly. The SX/B shift operators — like its PBASIC counterparts — do allow end bits to be lost and pad with zeros. What we do here is copy the end bit to the Carry and then to an Assembly shift with RR to move everything; the Carry bit is copied into the opposite end and we have a barrel shift. This is a useful trick.

MIX IT UP

We've seen how easy it is to decode SIRCS signals when there is no interrupt, but what happens when we want to fold SIRCS decoding into a project that requires buffered serial communications? In my opinion, the best approach is to put the SIRCS decoding into the interrupt with the UARTs. Here's how I do it:

Test_SIRCS_Tlx:
BANK sircs
INC irTlx
CJB irTlx, #3, SIRCS_Exit
CLR irTlx
JB irReady, SIRCS_Exit
JB irCmd, Check_Bit
JB IR, SIRCS_Exit
Start_Packet:
SETB irCmd
SETB irBit
CLR irBitWidth
CLR irBitCount
CLR irCode_LSB
CLR irCode_MSB
JMP SIRCS_Exit
Check_Bit:
JNB IR, Update_BitWidth
JNB irBit, SIRCS_Exit
CLR irBit
TEST irBitCount
JNE Tear_Bit
Test_Bit:
CJA irBitWidth, #216, Next_Bit
CJB irCmd
JMP SIRCS_Exit
Test_Bit:
CJL
RR irCode_MSB
RR irCode_LSB
CJB irBitWidth, #108, Next_Bit
SETB irCode_MSB,3

Okay, I know this is Assembly and can look a little scary at first, but it's really not; in fact, as you work through it you'll see that it very closely matches the high-level SX/B version we did earlier. The code starts by dividing the ISR by three to get the desired sample timing. The interrupt runs every 3.255 microseconds so dividing by three gets us to 9.766 microseconds — this is close enough to the 10 microsecond sample rate used by PULSIN that we can use the same sample values for bit measurements.

After dividing the ISR, we'll drop through to the actual code. This routine can only buffer one SIRCS command and if one has already been detected and not cleared by the foreground, we just exit. If we're not already receiving a stream, then we'll have a look at the input; if it has gone low (indicating a bit), then we can initiate the decoding process.

Everythings get set up at Start_Packet and on subsequent passes, will jump to Check_Bit. The variable called irBitCount is used to determine whether we're looking for the start bit (count is zero) or analyzing data bits. Once a valid start bit is detected, irBitCount is non-zero and subsequent passes will be routed to Test_Data which handles the bit-by-bit analysis of the incoming stream. After 13 bits (start plus 12 data bits), the irReady flag is set to alert the foreground and the word that holds the packed device and command codes is separated into clean bytes. Yeah, okay ... it's a little hairy at first but please trust me, after you've worked your way through it a couple times you'll see that it's not so bad. The good news for you is that I've already written and tested it — you just have to pop it into your own applications!

DUAL CONTROL

With the ability to decode the SIRCS stream in the interrupt, we enable two modes of control for the output port on our board. The receive UART and the SIRCS decoder both set flags when they have valid data, so all we have to do is monitor these flags and deal with them:

Main:
IF irReady THEN Process_TR
IF rxReady THEN Process_Serial
GOTO Main
The section at Process_IR was lifted right out of the simple control program. As far as serial control, I kept things easy by just using the numeric keys from a terminal program:

```cpp
Process_Serial:
    cmd = RCLRYTE
    IF cmd = &b1 THEN
    IF cmd = &b0 THEN
    IF cmd = 0 THEN
    Control = @0000_0000
    ELSEIF cmd = 9 THEN
    Control = @1111_1111
    ELSE
    DEC cmd
    mask = 1 << cmd
    Control = Control XOR mask
    ENDIF
    DELAY_MS KeyDelay
    GOTO Show_Port
    ENDIF
    ENDIF
    ' process other keys here
    GOTO Main
```

When a key comes in, we check to see if it falls within the range of numbers and if it does, convert it from ASCII to decimal by subtracting 0. If the value is zero, all control port outputs are turned off. If the value is nine, then all control port bits are turned on. If the value is one to eight, we create a mask for the appropriate pin. Note that we have to decrement the value before creating the mask so that a keystroke of 1 corresponds to bit 0 of the control port.

And there you have it — even more control from the couch! What will you do with it? Keep in mind that the hardware part of the project is very open ended; that is by design. That dual-control program has a transmit UART that is reporting status, but this code could easily be modified to be an SIRCS-to-serial translator, giving you IR control of a device that expects serial input. With the ULN2803, you can control moderate current outputs like relays, values, even stepper motors. One of my future personal projects is a stepper-controlled pan/tilt head for my camera; with this little board and a bit of code, I could control that pan/tilt head with a TV remote.

Give it some thought. I'm sure you'll find lots of cool projects for this little board, and you won't even have to get up from the couch to control them.

Until next time, Happy Stamping — SX/B style!  

---

**JON WILLIAMS**

jwilliams@efx-tek.com

**PARALLAX, INC.**

www.parallax.com

**SIRCS PROTOCOL**

www.sbprojects.com/knowledge/ir/sirc.htm

---

**MOUSER ELECTRONICS**

www.mouser.com

- Over A Million Products Online
- More Than 366 Manufacturers
- Easy Online Ordering
- No Minimum Order
- Fast Delivery, Same-day Shipping

(800) 346-6873

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

January 2009 NUTSVOLTS 23
It's A Great New Year... And Time To Have Fun With Electronics!

Build It, Learn It, Achieve It, Enjoy It!

Electronic Learning Labs
- Learn and build!
- 150, 300, 500 in one!
- Super comprehensive training

Whether you want to learn the basics of electricity, the theory of electronics, or advanced digital technology, our learning kits are for you! Starting with our PL130, we give you 130 different electronic projects, each with a comprehensive 162 page learning manual. A great start for the kids, young and old.

For the latest lab, check out our PL200! Its unique design has an integral front panel that includes controls, meter, display, and more for full control of your projects!

Not enough? Next is our PL300, which gives you 300 separate electronic projects along with a 165 page learning and theory manual. The PL300 walks you through the learning phase of digital electronics.

If you're looking for the ultimate lab kit check out our PL500. Includes a whopping 500 separate projects, a 152 page starter course manual, a 78 page advanced course manual, a hinged easy to carry case, and a 140 page programming course manual! The PL500 covers everything from the basics to digital programming! Learn about electronics and digital technology the fun and build some neat projects! Whenever one you choose, you will have hours of enjoyment!

HD Cables That Make Sense
- 100% certified to HDMI standards!
- Supports HDMI 1.3b Category 2!
- HDCP compliant!
- Transfer speeds up to 10.2Gbps
- Gold plated connectors

Home Theatre interconnect cables are like printer cartridges! You can buy a printer for $39 but the inkjet cartridges costs $40 each and you need 5-6 of them! DVD players are $49 but an HDMI cable for it at the same store is $80-$125! Stop the insanity! Stop the markup!

HDMI (High-Definition Multimedia Interface) has become the connection standard for HD. It permits the digital transmission of any video format along with uncompressed audio channels, DTS and other audio formats. These cables are the latest version 1.3b which supports lossless compressed audio streams like Dolby TrueHD and DTS-HD Master Audio. Why pay $50 to $150 for an HDMI cable at superstar down the street when you can get these high quality cables for a fraction of the price? Order yours today and save! (Check our site for other Home Theatre cables at a price that simply makes sense!)

Surface Mount Soldering Course & Kit
- Learn SMT soldering the easy way!
- Learn all about those tiny parts!
- Includes SMT training sessions
- End up with a neat 'Decision Maker'!

You’re going to have to go through some day, so it might as well be now. Virtually all electronic design has gone to Surface Mount Technology. SMT components are small, at times the size of a pin head, and require very special skills to assemble and solder. With a pair of good eyes, a magnifying glass, and a fine tip soldering iron, you will learn exactly how SMT components are attached to PC boards, how they work, and how to do it!

This kit takes you through all that, including soldering, desoldering, removing, and component placement. When you're done, you have a neat electronic Decision Maker you can use! Even includes flashing SMT LEDs! Includes a 9V battery!

Sub miniature 40W Stereo Amplifier
- 2 x 20W amplifiers in a single SMT package!
- Super efficient Class D spread spectrum design! Runs cool, no heat sink req’ed, all on a 2 1/2” board!

The UAM4 is the big brother to the extremely popular UM12. It uses the latest spread spectrum amplified technology to bring you clear, crisp, high-power audio without any of the heat which is normally associated with such an amplifier. Its extremely clean Class D design produces two independent 20 watt outputs! This can also be bridged to an extremely efficient (87%) single channel (mono) 40 watt amplifier!

The high impedance input is designed to use your choice of either a balanced line input or an unbalanced signal source using easy to connect Euro terminal blocks. Logic connections to ground are also provided to mute and/or shut down the amplifier. You can also enable the built-in over temperature signal to activate one of these controls automatically! Easy to use board jumpers offer selectable gain of +22dB, +25dB, +29.5dB or +35dB to match your input levels. Board jumpers also provide variable protection and shutdown options, as well as a stereo/mono/bridge mode. The amplifier also features built-in click and pop suppression to protect not only your ears and sanity, but your speakers and equipment!

Power input for maximum rated output is 18 VDC at 2.44A. Input voltage can be reduced to a minimum of 10VDC while maintaining the same high efficiency operation with reduced output power. If you’re looking for an incredible stand-alone stereo amplifier to build into your application, the UAM4 is it!

UAM4 40W Sub miniature Stereo Audio Amplifier Kit $69.95

800-446-2295
www.ramseykits.com

MORE than just friendly on-line ordering!

Clearance Sales, User Tips, Order Form, Dealer Information, FAQ, Monthly Broiler Guides, Downloads, Live Weather, Live Webcams, and much more!

Visit www.ramseykits.com for the latest pricing, seasonal, trend and condition. Copyright 2009 Ramsey Electronics, LLC...so there!

HXE2242C 5.8GHz Wireless Camera Pkg. Assembled & Tested $397.95

Betty Crockers - 8 sets makes 1702 cookies, even though it 2 cookies per pack, you've got to get all of it done, don't we?

http://nutsvolts.texterity.com/nutsvolts/200901/templates/pageviewer_print.pg=1&p... 12/02/2009 Page 25 of 100
Q&A

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

Q & A

AUD TO LED DISPLAY DRIVER CIRCUIT

Q

I need a circuit which will drive an 8x8 or 16x16 LED matrix from an audio input and generate random patterns. The audio input portion has to be very sensitive.

— Robert Christopher

A

A display based on the audio input will not be random. If the left and right channels of a stereo signal are applied to the X and Y inputs of a matrix, it will show the difference in the two signals. If the signal is mono, the display will be a 45 degree line that varies in length with the strength of the signal.

The LM3914 dot/bar graph LED driver is made for this application. The output is a current source so current limiting resistors are not needed. The IC has 10 outputs but can be expanded to 20 or more. I will design an 8x8 matrix display to show how it works (Figure 1).

The input range is set by the reference voltage. In this case, I set it to one volt, assuming that the input would be line level. The 1/2 input pin is biased up 1/2 volt so that one volt p/p input will not clip. R4 and R6 set the current sink of IC2 at 10 mA. The current sink of IC1 is set at 1 mA and the current gain of the PNP transistors will keep them saturated.

An array of PNP transistors would be nice but the ULN series are all NPN. I found a four transistor PNP array for $6 but with 2N3906s at $0.50, I went with singles. Pin 9 of the LM3914 is open for a dot display; you could connect pin 9 to Vcc for a bar display.

My schematic capture program (Eagle) does not do group copy of line drawings so I did not show all the diodes in the matrix.

DETECTION OF MOTOR SPEED AND DIRECTION

Q

I have attached the image of a plastic circular disk with 36 holes around the outer edge, driven by a DC motor. I need a circuit that will tell me speed and direction of the motor by the holes passing a detector.

— G. W. Heikkila

---

FIGURE 1

To tell direction, two detectors will be needed. The slotted type of detector will be the easiest to work with; I selected Digi-Key part number 425-1970-5-ND. The Mouser part doesn’t include a rise or fall time spec, so I can’t use it. The aperture for the sensor is 0.5 mm and the rise and fall times are 15 and 20 μs. That determines the hole size and maximum speed of the rim of the disc.

Rim Speed = RPM * Circumference = RPM * 2πR

where R is the radius to the hole center see (Figure 2).

The sensor aperture is a slot.
That, plus the hole diameter and rim speed, will determine the pulse width: Pw = (D + S)/SPEED. Since the minimum pulse width is determined by the rise and fall time of the sensor, then Tr + Tf = (Dmin + S)/SPEED and Dmin = RPM * 2πR * (Tr + Tf) / S.

If I make the distance between the holes equal to the hole diameter,

then the circumference = N * 2 * D where N is the number of holes. I want to work with seconds instead of minutes, so RPS = RPM/60. Now: Dmin = RPM * (Tr + Tf) * N * Dmin/30 - S. Solving for Dmin:Dmin = 30 * S/(RPM * N * (Tr + Tf)). In this problem, S = 0.5 mm, RPM = 10,000, N = 36, Tr = 15 μs, Tf = 20 μs Dmin is, therefore = 0.6 mm. I wouldn’t go that small; make D = 2 mm then the circumference becomes: 36 * 2 * D = 144 mm. The disc diameter is C/π= 45.9 mm.

I recently designed an analog tachometer for my truck. That circuit will be ideal to measure the RPM of the motor see Figure 3. The LM2907 (IC1), is a frequency to DC converter IC; Digi-Key part number LM2907-N-ND. A square wave signal at the output produces a DC voltage at the output that is a linear function of frequency. The LM2907 datasheet will explain the theory.

R1 and R2 set the input voltage 11 volt; R4 and R5 set the other input voltage at 1 volt. When the transistor is turned on, the collector goes close to ground so the input is close to square. Depending on your need for accuracy, you might want to divide the input by two to get a better square wave.

The circuit to tell direction (Figure 4), uses a D type flip-flop. If the data signal (D) comes before the clock, the Q output will be high. If the data signal comes after the clock, the Q output will be low. The capacitor, C1, stretches the pulse to ensure that it is concurrent with the clock when required.

**STABLE CONSTANT CURRENT**

I’d like to be able to use a Fluke 87 meter to read down to 300 micro ohms or so by injecting a stable
current, say 10 amps into the device being measured and then measuring the voltage drop across the device in question. Is this possible and if so, can you come up with any circuit suggestions to accomplish this?

— Jeff

In the circuit (Figure 5), I chose a supply voltage of five volts to keep the power dissipation in Q1 low and still allow measurement of up to 0.1 ohms. To measure up to one ohm, reduce the current to one amp by increasing R1, to one ohm. There is a small error due to the base current of Q1 flowing through R1, but this is less than 1% in the worst case and less than 0.1% typically. I would glue the diodes to the transistor case so they are at the same temperature; wait for thermal stability before taking a measurement.

The diodes will not exactly match the Vbe of the transistor, so you will need to adjust R4 for exactly one volt across R1. The transistor is in a TO-3 case and will need a good heatsink. Wakefield part number 423-A would be a good choice but I don't know who stocks it. Mouser has 567-423-K which doesn't have holes, so you will have to drill your own.

FM SHUTDOWN PROBLEM

Q: FM translators consisting of a receiver and a transmitter on another frequency still in the FM band require a transmitter shutdown if the signal from the receiver fails.

A: The problem is that when the signal fades or drops out below the receiver capability, the discriminator output turns to random noise. It seems that a variation of the night light control (Oct 08; page 23) would do the job if I can get a signal from the AGC or some other signal from the FM receiver. The transmitter load is less than 2.0 amps at 120 VAC.

— Bob Ziller

I don't see how the night light control circuit applies, and since an FM receiver does not have AGC, we have to look elsewhere. If you have a decent signal-to-noise ratio (6 dB or better), you can tap off some signal from an intermediate IF stage, probably just before the limiter. The circuit of (Figure 6) shows how this could work: The capacitor, C1, "remembers" the signal level, so when the signal drops out the switch is flipped. Since FM can still work when the signal is down at the noise level, the only alternative is to insert a pilot tone that can be detected and used to turn the transmitter off if the tone is not present.

The LM567 phase lock loop was made for this application. (Figure 7) is similar to a circuit I built some years ago to replace a tuning fork filter that was used in a business band radio for squelch control. The signal was 7 Hz and low level so it was not audible, but it was enough to be detected. The oscillator would be placed in the originating transmitter and added to the voice at a low level.

HYDRO PROJECT

Q: I contemplate installing a low voltage, mini hydro generator in a small river in my backyard. The unit...
cannot run without a load. It will feed a low voltage element in a water heater. When that thermostat is satisfied, it will feed a low voltage element in a small swimming pool. When that thermostat is satisfied, it will feed low voltage lights equal to the required load. I need a controller to insure that the water heater is always served first, then the pool, and then the lights.

— Bill Edgerton

A

If your thermostat switch is single pole, single throw (the usual case), then some relays will be needed. We can't rely on the voltage from the generator to operate the relays, so I am proposing 24 volt types. In the circuit (Figure 8), when the water heater is calling for power, voltage is removed from the other loads and applied to the water heater. When the water heater is not calling for power, voltage is passed on to the swimming pool. If it is not calling for power, the voltage is applied to the lights. The 24 volt transformer and relays should be available at your local plumbing shop.

HIGH IMPEDANCE PREAMP

I need an ultra low noise high impedance preamp for an instrument application. The preamp should be reasonably flat over 5 kHz to 500 kHz and be located on the sensor, fully shielded, battery powered, and driving a 50 ft. shielded cable to the remaining instrumentation. Short battery life is not a problem as the test runs are 30 minutes or less.

— James Ward

A

I am going to assume that your signal source is inductive with low resistance because that will give the best noise performance. I chose the MAX4448 because it has low noise and wide bandwidth; see (Figure 9).

Resistors have noise = 4kTR where k = Boltzman's Constant = 1.38x10^-23 T = Kelvin temperature = 273 at room

This equation takes into account all the input noise sources except 1/F noise which is important below 1 kHz. Since your band starts at 5 kHz, we can ignore the 1/F noise.

Et = √(en^2 + (Rp + Rn)^2 * ln^2 + 4kT*(Rp + Rn)) where:

Et is the equivalent input noise voltage in volts per root Hz en is the specified input noise voltage density = 4.5 nV/√(Hz)

Rp is the non-inverting input resistance (assume five ohms)

Rn is the inverting input equivalent resistance = R1*R2/(R1 + R2) = 986 ohms

In is the specified input noise current density = 0.5 fA/√(Hz)

Going through the math, I come up with Et = 5.9 nV/√(Hz)

The gain of this amplifier is 36 dB which leaves about 1 MHz bandwidth. To find the wideband noise, multiply by the square root of the bandwidth: 5.9 nV*√1 = 5.9 µV. To find the output noise, multiply by the gain. Since the gain doubles every 6 dB, 36 dB is a gain of 64, but this gain is 69.1 (rounding error): 5.9 * 69.1 = 407 µV. If the 1/F noise is a problem, put 470 pF across R2.

The MAX4448 won't have any trouble driving a 75 ohm cable at that level, but I added 75 ohms in series to isolate the capacitance of the line which could cause the preamp to oscillate. You can shut down the preamp between measurements to prolong battery life.

PIC PROGRAMMING

What program do you use to program the PIC 12F675? I'm new to PIC programming and have been following your examples. I'm

January 2009 NUTSVOLTS 29

still a bit confused about the ANSEL and CMCON command. Can you shed a little light on this? Thanks.

Bob

I use PICBASIC PRO from microEngineering Labs, Inc. You can download a demo version plus MicroCode Studio, plus the PICBASIC PRO manual from: www.mecanique.co.uk/products/compiler/php.html. You can download just the manual which will answer all your questions. I'm not the one to ask because I am a newbie myself.

TELEPHONE HEADSET ADAPTER

I am hard of hearing and have a difficult time hearing on the telephone. Headsets help but most cover only one ear. I have a set of headphones for my computer that cover both ears and have a boom microphone. I would love to be able to plug these into my phone.

Can you recommend a circuit that would allow some amplification for incoming audio and provide phantom power, as well as interfacing for the condenser mic which is what I think these computer mics are? Many thanks.

Denny Sioux City, IA

Not knowing anything about your phone or the microphone, I cannot come up with a circuit, but I do have a solution to your problem. I have a Panasonic model KXTG5672 portable telephone. The handset has a speaker phone button which increases the volume so it is audible across the room. My wife loves it; she is hard of hearing, also. The handset has a phone jack on the side that accepts a 2.5 mm plug, in case you want some privacy. You no doubt will have to change the 3.5 mm plug that is on your headphones, look for Mouser part number 17PP033-EX.

MAILBAG

Dear Russell,

Good job continuing the Q&A column! I enjoy reading it.

One comment: One reader asked you about a power supply for a battery charger, and you guessed the oddball transformer in question was a ferroresonant regulating transformer. Seems like a good guess to me. But at the end of your answer you said "batteries don't care about being pulsed; in fact it may be better than smooth DC." This seems not to be true of laptop batteries. If you look at the Maxim 8731A charger chip, the Intersil 8731, the TI bq24702, or any of the other modern chips, they all use buck converters and the battery charge current is regulated via the duty cycle of a power FET that controls the current in the secondary of the buck converter. The output of that buck converter is, of course, a non-smooth DC waveform. The charger chip makers all warn you to use the proper ripple-absorbing capacitors in parallel with the battery and go to great lengths to show you what types of caps work, which don't, what values the caps should have, and where they should be placed. Some of them specifically warn you not to let the batteries absorb the ripple themselves because it heats them up substantially.

Bob Cohoon

Thanks for the feedback, Bob; I was thinking of automotive type batteries which absorb pulses easily. A capacitor in parallel with the battery will only work if the battery has high impedance which may be the case in these types. Certainly a temperature rise due to internal losses would not be good in a fast charger that determines full charge by a temperature rise.

Dear Russell,

Regarding the "Battery Charger" inquiry by Michael Craft in the October 08 N&V issue, page 24...my take on the question is this: The charger is used to charge and to maintain a fully charged state of a string of lead-acid cells while simultaneously supporting an external load. The reader describes a float voltage of 36 volts. The "float" or recharge voltage under which a lead acid cell will experience long life is 2.25 volts; this suggests a battery composed of 16 cells, having a float voltage of 36 volts, a fully charged open circuit voltage of 3.52 volts, and an end-of-discharge voltage of 2.20 volts.

The load characteristic of a lead acid battery — that is, the impedance presented to the charger — is non-linear with voltage. As the battery charges (from a discharged state), the voltage impressed across the battery terminals rises so that the charging current — if delivered from a constant voltage source — decreases. As the float voltage point is approached, the charging current drops to a very small value, and it is important not to force charging at this point, else electrolysis will occur and water will be lost from the electrolyte solution.

The voltage and current charging characteristics look like (Figure A); the charge current drops rapidly as end-of-charge conditions are approached.

Thus, a lead acid battery charger needs to be a two step device, delivering current (without regard to battery terminal voltage) at the onset of charge, and a constant float voltage at the end-of-charge point.

Transformer T2 operates from the unregulated AC line voltage. The voltage delivered to the rectifiers is the sum of the T2 secondary voltage plus the secondary winding voltages on T1. We can surmise that T2 is designed such that float voltage will be provided out of the transformer, and that the rectifier section (T1) would not be needed.

On the other hand, at the start of charge, the battery terminal voltage is significantly smaller than the voltage out of the rectifiers, and without the moderating influence of T1, this could result in forcing excessive charge current into the battery. T1 is effectively fed by a current source — the series-connected inductor L1 — and this limits the battery charging current.

T1 also provides a degree of line regulation. L1 and C1 form a 60 Hz resonant tank circuit and for constant load, the output voltage from T1 remains relatively constant. If the AC line voltage decreases below design center, the output voltage from T2 will decrease proportionally. The battery current, however, will remain nearly constant during charge. Once the charging event has completed and float conditions are obtained, T1 will compensate for variations in T2's secondary voltage caused by corresponding variations in AC line voltage.

Inductor L2 makes the battery appear as a current fed load. L2 also acts to remove the 120 Hz ripples from the battery bus, which suggests that this charger might be part of a telephone system in which it is important to isolate a hum voltage component from the 36 volt DC.

Peter A. Goodwin

Rockport, MA

Thanks for the feedback; I forwarded your comments to Michael Craft.
A NEW STARTER KIT

SchmartzBoard has announced the availability of the new “Solder By Numbers (SBN) Starter Kit” – a starter kit for people who are new to electronics. The SBN Starter Kit includes:

- An Elenco 40 watt electronic soldering station
- Assorted sized tips for the soldering iron
- Water soluble solder flux in pen form
- Safety goggles
- A lighted magnifying glass
- Two screwdrivers
- Wire cutters
- Wire strippers
- Needle-nose pliers
- SchmartzSolder
- A Solder By Numbers project with SchmartzBoard and Components (LED paperweight)

The Starter Kit costs $99 and its release coincides with the launch of www.solderybnumbers.com.

“It takes two things to build an electronic circuit: the ability to solder the components and the ability to take an electronic schematic and transfer the schematic to a real circuit board,” said Neal Greenberg, SchmartzBoard VP of Marketing. “SchmartzBoard has made these issues easy which opens up electronics to a

GET YOUR AMATEUR HAM RADIO LICENSE!

Enjoy the fun of Amateur Radio broadcasting! You’ll be able to transmit over hundreds of miles with two-way handheld radios anywhere!

Protect your family & friends when traveling, during power outages, cell phone blackouts, emergencies and natural disasters.

All you need to do is pass a short 35 question multiple-choice test to get your License.

GUARANTEED PASS! You will get your license or your money refunded - no questions asked.

Just listen to the 2 Audio CDs in your car or truck. Look over the simplified Guidebook. Plus – a data disc is included if you want to study the test on your computer.

Only $39.95 (S&H $5.00)

Call now: 1-800-877-8433

For more information, contact:
Integrated Ideas & Technologies, Inc.
Web: www.integratedideas.com

EARN MORE $$$

Be a FCC Licensed Wireless Technician!

Make $100,000 a year with NO college degree!

Learn Wireless Communications and get your “FCC Commercial License” with our proven Home-Study course.

Move to the front of the employment line in Radio-TV, Communications, Avionics, Radar, Maritime and more.

No previous experience needed!

Learn at Home in your spare time!

Signed:
Warren Weagant's FCC License Training

480 Gate Five Road - Suite 107
Sausalito, CA 94965-1412

January 2009

http://nutsvolts.texterity.com/nutsvolts/200901/templates/pageviewer_print?pg=1&p...
wider audience than previously possible.

With SchmartBoard prototyping technology, anyone can solder any type of SMT electronic component.

- SchmartSolder makes it possible for anyone to hand solder through-hole components.
- Solder By Numbers tells you where to solder them onto the SchmartBoards to create your circuits.

“For more information, contact: SchmartBoard
Web: www.schmartboard.com/index.asp?page=starter_kit

The Standard for checking Capacitors in-circuit
Good enough to be the choice of Panasonic, Pioneer, NBC, ABC, Ford, JVC, NASA and thousands of independent service technicians.

Inexpensive enough to pay for itself in just one day’s repairs. At $229, it’s affordable.

And with a 60 day trial period, satisfaction guaranteed or money-back policy, the only thing you can lose is all the time you’re currently spending on trying to repair all those dogs you’ve given up on.

CapAnalyzer 88A
Available at your distributor, or call 561-487-6103

Electronic Design Specialists www.eds-inc.com

Locate shorted or leaky components or conditions to the exact spot in-circuit
Still cutting up the pcb, and unsoldering every part trying to guess at where the short is?

$229

Your DVM shows the same shorted reading all along the pcb trace. LeakSeeker 82B has the resolution to find the defective component. Touch pads along the trace, and LeakSeeker beeps highest in pitch at the defect’s pad. Now you can locate a shorted part only a quarter of an inch away from a good part. Short can be from 0 to 150 ohms.

LeakSeeker 82B

FPGA Experimenter’s Board
Data Download

- Linux or Windows
- No JTAG drivers needed
- Xilinx SpartanII FPGA
- Customise peripherals with:
  * Wireswap card ($20)
  * Code samples

Demand Peripherals www.demandperipherals.com

Great Robots. Great Prices.

• 4WD - 4 servo motors!
• Rugged construction
• High traction over carpet, concrete, grass, and more!
• Rubber tires (2.5")
• Multiple 1/4" thick decks
• Over 60 square inches of mounting space
• Measures 6.75 x 5", and 4.5" tall
• Available in black or dark blue
• Add your own microcontroller, sensors, video camera, GPS, or other electronics
• All for under $100!

RIGEL

BUDGET ROBOTICS

Robots for the Rest of Us

WWW.BUDGETROBOTS.COM
INSTRUMENTATION FOR MODEL ROCKETRY

BY MICHAEL BESSANT

In two of my previous *Nuts & Volts* articles (August and September 2007 issues), I presented the design of a compact and lightweight flight recorder for use in model rocket development. An example application that was given described taking recorded signals from a three-axis accelerometer and exporting the post-flight data to a PC spreadsheet for display and analysis. The design was based on a PICAXE-18X microcontroller and non-volatile FRAM storage, powered from a single three volt supply. Connections to the flight recorder's analog inputs and digital I/O lines were via a sensor port that can be programmed in Basic to interface with signal conditioning circuitry and a wide range of sensors. Figure 1 shows a copy of the original flight recorder schematic. The previous articles also introduced a number of compatible sensors, but did not describe the associated instrumentation circuitry.

This article covers the addition of instrumentation to measure a rocket's rate of roll, relative altitude, and detect booster-stage separation. Such information is very useful when developing multi-stage model rockets that are capable of reaching speeds in excess of 300 km/h. Important design objectives can often include using fin-induced roll to improve directional stability (but without tangling the recovery chute) and minimizing the risk of airframe damage by deploying the chute when the rocket is near apogee and travelling at low speed. These and other flight characteristics can be investigated using the sensors and circuitry shown in Figure 2 and described in detail below.

**Vertical Acceleration**

The three-axis MMA7260 accelerometer breakout board described in my previous articles is retained (available from SparkFun Electronics; [www.sparkfun.com](http://www.sparkfun.com)), but only the vertical axis is recorded. Compared with the eight-bit `readadc` command used previously to measure acceleration, improving resolution is achieved by the use of the `readadc10` command, but this has made the revised record subroutine slightly more complicated. This is because the latter command generates word variables and the `write12c` command can only write bytes to memory. In addition, using two memory locations to store each 10-bit ADC sample is less efficient (and using an entire byte for only two bits, is even less), but the storage capacity remains adequate for recording 400 samples/second throughout the duration of a typical model rocket flight.
The revised PICAXE code is listed in Figure 3.

**Relative Altitude**

Recordings of vertical acceleration can be integrated to yield a velocity profile and, by a second integration, an estimate of peak altitude. Unfortunately, such derived altitude information becomes less accurate as a slowing rocket's flight path departs from the vertical, and may not therefore provide a reliable means of detecting apogee. Although attempts at determining absolute altitude using a calibrated pressure sensor can often be frustrated by errors caused by instrumentation bay venting and air turbulence around the model's airframe, these need not present a problem when detecting the point of minimum pressure at apogee. Therefore, a low cost (~$20) Honeywell 24PPC is used to measure the rocket's relative altitude. This basic pressure sensor employs a 5K bridge that has a typical output sensitivity of 15 mV/psi when powered from 10 volts. (See the sidebar “An Introduction to Pressure Sensing.”)

As model rockets rarely reach beyond a kilometer in altitude, the typical output range of the chosen pressure sensor will only be around 7 mV. Unfortunately, this low level signal is inconveniently positioned at the maximum (near sea level) end of its total pressure range and must therefore be offset before amplification. The simplest way to introduce this offset is by adding resistance (VR1 + R6) between one side of the bridge and ground. Although this marginally degrades the sensor's specified sensitivity and linearity, it is a perfectly acceptable (low component count) solution for this application.

The differential output signal from the pressure sensor is amplified, by a MAX 4194 single supply instrumentation amplifier before being presented to a 10 bit ADC channel. The amplifier output may swing to within about 30 mV of the ground and supply rails when operating with an REF input of 1.5 volts. The MAX 4194 implements a classic three amplifier topology that enables gain to be set by a single resistor (R3) between pin 1 and pin 8. The gain (G) of the instrumentation can be determined by $G = 1 + 50K/R3$, where R3 is the gain setting resistor. In order to accommodate the normal variations in atmospheric pressure, plus the specified spread in sensor characteristics, the amplifier gain was set at a rather modest figure of 228 and VR1 adjusted to give an output offset of 600 mV. This provides adequate measurement resolution over the expected altitude range without the risk of exceeding the three-volt ADC input limit.

**Roll Rate**

A photoelectric device can be used to measure the rate of roll (rotation about the rocket's vertical axis) by detecting the direction of the sun through a narrow vertical slot in the side of the instrumentation bay. The detector circuit outlined in Figure 2 employs a general-purpose yellow LED that generates a maximum change in
output of approximately 500 mV between viewing background sky and direct sunlight. This high impedance signal is conditioned by a non-inverting BiCMOS amplifier before being presented to an eight bit ADC channel. The TS922IN dual op-amp is capable of rail-to-rail operation down to a supply level of 2.7 volts (the second op-amp in this package is configured as a voltage-follower to drive the REF input of the MAX 4194 instrumentation amplifier).

**FIGURE 2. Acceleration, altitude and roll interface circuitry.**

An Introduction to Pressure Sensing

Many different units are used to measure pressure, depending on the country and target application (to convert between three of the most commonly used units, you need to know that 1 psi equals 6.8947 kilopascals [kPa] or 68.947 mbar). Mean atmospheric pressure is defined as 1013.25 mbar (14.69 psi) at sea level. However, the actual value can fluctuate by ±20 mbar due to normal weather variations. If we take into account sensor errors and normal variations in atmospheric pressure and air temperature, the apparent altitude can easily deviate by hundreds of meters — not an insignificant error compared with the maximum altitudes achieved by model rockets. Near sea level air pressure decreases almost linearly with increasing altitude but the relationship becomes increasingly non-linear above a few kilometers. For a detailed demonstration of the relationship between pressure, altitude, and temperature, visit the NASA interactive atmosphere simulator (www.grc.nasa.gov/www/K-12/airplane/stmo.html).

Integrated circuit pressure sensors employ piezo-resistive technology to convert pressure to an electrical signal. The main element used is a silicon chip, which has been micro-machined to produce a flexible diaphragm, around which four identical resistors are diffused to form a bridge. The force of pressure on the diaphragm causes the balance of the bridge to change, creating a differential voltage output proportional to the applied pressure. This technology can be used to assemble the following three types of pressure sensor.

- **Gauge:** Measured pressure relative to ambient pressure
- **Differential:** Measured pressure relative to another pressure
- **Absolute:** Measured pressure relative to a vacuum

Sensors are available for a variety of pressure ranges and outputs. The most basic pressure sensors only provide mV level differential outputs direct from a bridge, while more expensive versions have onboard amplification and volt level outputs. A recent trend is towards including an integral ADC that enables direct connection to a microcontroller I²C or SPI interface. The appropriate pressure sensor for measuring altitude is an absolute type with a maximum rating of 1,013 mbar. For model rocketry, the sensor would only be used over a narrow part of this pressure range and there would be little advantage in using an expensive type that offered general-purpose signal conditioning.
Stage Separation

Previously, digital input IN 6 of the sensor port was connected to a switch that triggered a recording run when the rocket left the launch rail. An alternative method of triggering is now used that frees this input for other tasks (e.g., detecting booster-stage separation). The trigger subroutine now monitors vertical acceleration and initiates recording when a pre-defined threshold is exceeded twice within a specified period of time. Although it is no longer possible

```
setfreq m8
symbol memin = 0   'Start address
symbol memmax = 32766   'End address
symbol memadrss = W6   'Memory counter (word)
symbol liftoff = 160   'Set trigger level at 2G
symbol redled = 2   'Define constants.....
symbol download = 3
symbol booster = pin6
symbol cable = pin7
symbol power = 0
high power
pause 4000   'Hold power on
'Wait for power to settle
playback:
pause 2000
pause 2000
serout download, T4800,
{"LABEL","",","","","Acc,Apo,Sun,Env",CR)
pause 2000
serout download, T4800,
{"2 Stage Rocket Instrumentation V.6",CR)
12slave %{10100000, l2cfast, l2cword
for memadrss = memin to memmax step 6
  readl2c memadrss, (b0,b1,b2,b3,b4,b5)
  serout download, T4800,"U18","",","","W0","","W1","","W2","","W3",CR
next memadrss
if cable = 0 then trigger
  low power
  'Start trigger routine if cable disconnected,
  'else, power down
  trigger:
  readadc 0.b0
  low redled
  pause 100
  high redled
  readadc 0.bl
  pause 100
  if b0> liftoff AND b1> liftoff then record
  goto trigger
record:
12slave %{10100000, l2cfast, l2cword
for memadrss = memin to memmax step 6
  b5 = pin AND %11000000
  readadc10 0, w0
  readadc10 2, w1
  next memadrss
  write12c memadrss, (b0,b1,b2,b3,b4,b5)
  pause rate
  next memadrss
power low
```

FIGURE 3. Revised (10-bit ADC) PICAXE-18X program for flight recorder.
to record the very start of engine ignition, the new technique has proved to be very reliable in operation. The general configuration of the instrumented rocket is shown in Figure 4.

**Test Results**

The Excel plot shown in Figure 5 was chosen as an interesting example of a 'worst case' test flight. The spent booster did not separate cleanly, causing the rocket to tilt over before the second-stage engine ignited. As a result, the remainder of the flight was nearly horizontal at an altitude of not more than 20 meters! However, even this dangerously shallow trajectory yielded a useful plot of the relative altitude.

It is possible to see that the recovery chute deployed successfully after the rocket reached apogee but that the time to impact was alarmingly brief. If we now turn our attention to the 'sun' trace, it is possible to identify cyclic variations in the light level caused by airframe roll, even though the sky was completely overcast on the launch day.

The acceleration trace provides a useful guide to the timing of these events. For example, it is possible to identify booster stage separation, second stage ignition, chute deployment, and final impact. Note that only the acceleration trace is calibrated (the other two traces have been expanded in the interests of clarity) and that the booster separation detector switch failed.

**Potential Enhancements**

The low component count/power instrumentation described here is capable of providing useful flight data and has the potential for further development. Although additional analog and digital channels can currently be accommodated by connecting an I2C interface device to the sensor port, a more direct solution would be to base the flight recorder on a larger member of the PICAXE microcontroller family. For example, the PICAXE-28X1 (28 pin) would provide four analog and six digital inputs. It can store 1,000 lines of code in onboard Flash memory and has a maximum clock speed of 16 MHz.

To order back issues of the August/September 07
Nuts & Volts, go to our webstore at
http://store.nutsvolts.com or call 800-783-4624.
16x16 TRI-COLORED LED MATRIX - REAL TIME CLOCK FOR THE PARALLAX SX52

BY TIMOTHY D. GILMORE

In my second year of college, some students programmed a huge light display standing about 12 feet high in 8085 assembly language. It was round with colored light bulbs and reminded me of the lights on a Ferris wheel. As an electrical engineering student, I was captured by the coolness of being able to control something as beautiful as this display. A couple of years ago, I walked into a Brookstone store in the Atlanta airport and saw a floating display clock. I had just started getting involved with Parallax microcontrollers, beginning with a year worth of experience with their BASIC Stamp 2 microcontroller and then moving into SX chips.

I wanted to create a similar light emitting diode (LED) display project but found very little on the Internet at the time that was related to moving LED displays. There were some Microchip PIC and Atmel AVR LED display projects but nothing really for the SX chips. The SX chips at the time were relatively new but also powerful, inexpensive, and very fast. I found one LED display project on the Internet that used a PIC chip. I converted the PIC Basic code over to work on an SX28. It used a 20x7 LED matrix display that I hand-soldered point-to-point. It was my first microcontroller LED persistence of vision (PoV) project.

I liked the SX chips as they could also be programmed in an easy SX/B Basic like language, as well as assembly language if required. I studied as much as I could find on PoV and LED matrix theory as I knew this would not be easy to immediately learn. I was new to SX chips but it was a huge step up in capability as compared to the BASIC Stamp 2 microcontrollers. I was and still am a very active member of the Parallax user forums (http://forums.parallax.com/forums/) and have received a great deal of help from all of the moderators and forum users such as JonnyMac, Bean, JDOhio, Sparks-R-Fun, and others. It is from their help that this project is possible.

Three Colors From Two

This project allows the user to display the time and date not only using one color, but three color LEDs. Actually, the LEDs are bi-colored 8x8 LED matrix modules I originally purchased inexpensively on eBay from a company in Hong Kong named Sure Electronics (www.sureelectronics.net/goods.php?id=230). Later this year, their website was available and I now purchase directly through that.

I used the trick of turning on both LEDs at the same
time to create a third color. This allowed me to utilize three colors (red, green, and orange) with only two actual LEDs (red and green). Figure 1 illustrates the 8x8 bi-color LED module connection layout.

**Lots of I/O Needed**

Being able to control only a single 8x8 LED matrix was possible with an SX28 chip because one would only require 20 I/O ports for the anode and cathode LED lines, as well as related control lines. However, combining four 8x8 LED matrix arrays into a larger 16x16 array would require 36-40 I/O ports so an SX48 or SX52 chip would be needed to handle the additional I/O, as well as provide more program data space and Random Access Memory (RAM) variable space.

This design uses an SX52 which is now obsolete by Parallax but can still be found from other online sources (e.g., www.hobbyengineering.com/H2225.html). It may also be possible to modify the design for use with an SX48 proto board. The main physical difference between the SX chips is the available I/O. The SX28 provides 20 I/O ports, the SX48 provides 36 and the SX52 provides 40. All of the Parallax SX chip proto boards are inexpensively priced. Besides the SX52 proto board, the only other required SX chip items from Parallax are a programming tool, being either the USB SX-Key (debugging capability) or the USB SX Blitz module (no debugging capability), a 4 MHz resonator, and a 7.5 VDC one amp power supply.

The key to this project is to arrange the four LED modules in an expanded matrix to have 16 LED anodes and 16 LED cathodes available to the SX52. I accomplished this with a point-to-point solder approach so no expensive PCB is required. This was, however, a bit time-consuming. Once the 16 LED anodes and 16 LED cathodes are wired correctly, they can be taken out to the SX52 proto board to their respective I/O ports of RDE and RBC. A DS1302 Real Time Clock (RTC) chip is used with a 32.768 kHz crystal also available from Parallax. The RTC and crystal use three data lines connecting to the SX52 RA I/O ports. Two other RA I/O ports are used for the LED color control hardware four (74HC573) ICs to decide whether to select red, green, or yellow LEDs via software selection.

Finally, a 74HC165 chip is used with the final RA I/O ports for up to eight buttons for expansion control. However, only three buttons are used in this design (e.g., up, down, and next) to set the time and date, so it is possible to redesign this section and wire in three buttons directly into the RA I/O ports with some software modifications and not even use the 74HC165 at all.

**Additional Hardware**

Besides the SX52 proto board, programmer, four LED matrix modules, and RTC, the design also requires color and button control hardware as stated earlier. The color control is comprised of four 74HC573 octal D-type transparent latch ICs. Software control for either red, green, or both enables the appropriate set of 16 I/O lines going to either the red or green ULN2803 Darlington array ICs for current amplification to the 16 LED matrix module cathodes.

Besides the four ULN2803 Darlington array ICs, the 16 LED matrix module anodes require additional current amplification accomplished by 16 very small surface mount FDN304PZ P-channel FETs. It would also be possible to redesign the circuit and use N-channel FETs in place of the ULN2803 ICs for better current control across the LEDs. However, for this design it is adequate. I had to
make a custom printed circuit board (PCB) with etchant solution to solder the SMT FETs and run wires to the LEDs as they were a bit hard to work with, being so small.

A last minute design addition was to add a single 74HC165 8 bit serial in/parallel out shift register to control up to eight push buttons for time and date setting capability. A previous software version allowed the user to bypass this and set variables within the software using either an SX-Key or SX Blitz module. This, however, brought an additional expense for users without an SX-key or SX Blitz, but may still be acceptable for a single application.

The case I used came from RadioShack. However, I do admit that I should have used a larger case as this one was a tight fit. I have four push buttons I also bought from RadioShack of which I was able to leave one unused at the moment for future expansion. The design presented here only requires three push buttons. I also crudely cut an open face hole in the case using a utility knife and glued a piece of thin smoke colored Plexiglas to somewhat protect the display. I mounted an extra power socket and ran the connections to the SX52 proto board. I was also careful to place the SX52 proto board towards the bottom of the case so I could cut out a hole for access for any additional SX-Key/SX Blitz module programming of the SX52 for future expansion. Finally, thicker rubber feet (from RadioShack) are mounted on the bottom of the case for appeal.

(Figure 2) illustrates the complete hardware schematic.

**The Software**

SX/B or SX Basic is the compiler Basic used with the SX52 chip. The latest SX-Key software can be downloaded from the Parallax website (www.parallax.com) which also includes their free SXB compiler Basic 1.51.03. The program file DS1302_to_LED16x16DisplaySX52_v3.8b.scb is open into the SX Key compiler software. The user simply plugs their SX-Key or SX Blitz module into the SX-52 proto board, applies power to the proto board and associated LED hardware, and clicks on the Compile button.

In a moment, the user will see a scrolling 06 on change up or down with the up/down push buttons to set the hours. Once set, the user presses the next button and the LED will change color and display a scrolling 30 for the minutes. This process continues for the seconds, month, date, year, and day code. The day code represents the day or the week (e.g., 01 = Monday, 02 = Tuesday, ... 07 = Sunday).

This is used because the algorithm in the DS1302 RTC IC only reads a two digit year code, so by telling the DS1302 what day it is, it will know if it is 1808, 1908, or 2008. (This is pretty smart technology!) Once the time and date are set by the push buttons, the user can just sit back and enjoy the scrolling display.

The software basically repeats the scrolling of the time on the top eight rows of LEDs and the date on the bottom.
eight rows of LEDs across the 16 columns; three times in red, green, and yellow. There is an added bonus, however, which is another routine that allows 16x16 LED display pictures to be scrolled across the display between each LED color cycle. It defaults to a smiley face, but the user can change that with the WDATA statements that can be found in the SXB program file.

There is also an Excel file (Tile Calculator_16x16.xls) that allows the user to enter a 1 or 0 in the white and black squares that will automatically generate the WDATA code for the user to copy and paste into the SXB program file. This makes it very convenient to create custom 16x16 LED displays.

Virtually anything the user wants can be set up to flash on the screen or scroll from the left to the right. It is possible to scroll something upwards, but the LEDs are a bit dimmer during the process because the scanning routine does not efficiently use the P-channel FETs to provide more current to the display through the ULN2803 ICs.

The heart of the software is its interrupt service routine. Running at 1 kHz, it displays the LED’s current time, date, and picture status. The software is well commented to see what is happening throughout most of the program. The StringWriter and other subroutines are provided compliments of Parallax forum moderators who helped with example code. This routine converts an array byte into a display character that is then scrolled by the interrupt routine. It was used in a Parallax product called the Robolympic Badge that scrolled messages across a smaller 10x14 display.

**Closing Comments**

I started this project a year ago and have learned so much from the help of the Parallax forum moderators and forum users. It’s been almost three years now since I have been on the forum as T&E Engineer with over a thousand posts.

Most of my completed projects lend themselves to LED or LCD visual type projects. I hope you enjoy building this fun and interesting project as much as I did. I hope I have opened the door for others to get into LED matrix projects with persistence-of-illusion. I have a couple of other similar display projects which even are bigger and better. So, look for those in an upcoming issue of Nuts & Volts!

---

**PARTS LIST**

<table>
<thead>
<tr>
<th>ITEM</th>
<th>QTY</th>
<th>MODEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Enclosure (7&quot; x 5&quot; x 3&quot;)</td>
<td>1</td>
<td>270-1807</td>
</tr>
<tr>
<td>Jumbo Self-Stick Cushion Feet eight-pack</td>
<td>1</td>
<td>64-2342</td>
</tr>
<tr>
<td>Size M Panel-Mount Coaxial DC Power Jack</td>
<td>1</td>
<td>274-1563</td>
</tr>
<tr>
<td>50 ft. Blue Insulated Wrapping Wire (30 AWG)</td>
<td>1</td>
<td>278-503</td>
</tr>
<tr>
<td>50 ft. White Insulated Wrapping Wire (30 AWG)</td>
<td>1</td>
<td>278-502</td>
</tr>
<tr>
<td>50 ft. Red Insulated Wrapping Wire (30 AWG)</td>
<td>1</td>
<td>278-501</td>
</tr>
<tr>
<td>Pre-Punched IC Spacing Perf Board</td>
<td>1</td>
<td>276-1385</td>
</tr>
<tr>
<td>Eight Pin Retention Contact - IC socket</td>
<td>1</td>
<td>276-1995</td>
</tr>
<tr>
<td>16-Pin Retention Contact - IC socket</td>
<td>1</td>
<td>276-1998</td>
</tr>
<tr>
<td>18-Pin Retention Contact - IC socket</td>
<td>2</td>
<td>276-1992</td>
</tr>
<tr>
<td>20-Pin Retention Contact - IC socket</td>
<td>4</td>
<td>276-1991</td>
</tr>
<tr>
<td>Mini SPST 0.5 amp Momentary Switch (four pack) - Pushbutton N/O</td>
<td>1</td>
<td>275-1547</td>
</tr>
<tr>
<td>10K ohm 1/4 watt Carbon Film Resistor (five-pack)</td>
<td>1</td>
<td>271-1335</td>
</tr>
</tbody>
</table>

Mouser Electronics — www.mouser.com

| (16-25) FDN304PZ P-channel FET (surface-mount)                      |     |         |
| (4) 74HC573 three state Octal D-Latch IC (DIP)                       |     |         |
| (1) SX52 Proto Board from Parallax¹                                |     |         |

eStreetPlastics — www.estreetplastics.com

| (1) 1/8" thick - 6" x 12" Gray "Smoked" Plexiglass Acrylic Sheet - #2064|     |         |

Hobby Engineering/Mouser Electronics — www.hobbyengineering.com

| (1) SX52 Proto Board from Parallax²                                |     |         |

Parallax — www.parallax.com

| (1) SX-Key (debugging capability) (USB)                            |     |         |
| (1) SX Blitz module (no debugging capability) (USB)               |     |         |
| (1) 4 MHz Resonator (DIP)                                         |     |         |
| (1) 15 VDC one amp Power Supply                                   |     |         |
| (1) DS1302 Timekeeping Chip                                       |     |         |
| (1) XTAL-32.768 kHz                                              |     |         |
| (1) 74HC168 Parallel to Serial IC (DIP)                           |     |         |
| (4) ULN2803A Darlington Array IC (DIP)                            |     |         |

Sure Electronics — www.sureelectronics.net/gooods.php?id=230 (or eBay - for better shipping cost)

| (1) (10 pins 8x8 Dot-Matrix 3 mm dia. Bicolor LED Display:        |     |         |

MISC: 22 AWG solid wire (three colors or more), soldering iron (15-30W), small diameter resin core (~60/40) solder, 20 AWG-30 AWG wire strippers, drill set and/or Dremel tool, PCB etchant kit, and mini Phillips screwdriver set (for the case).

¹Actual case used was Project Enclosure (6" x 4" x 2") - Model 270-1806. However, it is a very tight fit. Recommend using Project Enclosure (7" x 5" x 3") - Model 270-1807 or larger.

²The SX52 proto board is obsolete from Parallax. However, these sources may provide a limited quantity of the product. A search on the Internet is also possible. Contact the author for using an SX48 proto board instead with limited capability.

Here is a link to the completed projects Parallax forum that also has a YouTube video link and additional information: http://forums.parallax.com/forums/default.aspx?ft=21&m=189908.
THE PROTO BUDDY

For those just getting started in electronics as a hobby, a solderless breadboard (SBB) is the perfect platform for building those first circuits.

Also known as a solderless proto board, an SBB has rows of springy metal contacts embedded in a plastic base. The contacts hold components in place and the rows can be interconnected with wires. Changing a circuit is quick and easy. And when done, the circuit can be disassembled and the components can be saved for reuse in the next circuit.

Figure 1 is a photograph of a typical SBB while Figure 2 is a simplified drawing showing how an SBB is wired internally. An SBB can be purchased from Jameco (part #194299), Electronix Express (part #03MB102), and other vendors.

A pair of contact strips marked by red and blue lines run along both of the long sides. The plus and minus signs show that those strips are often used as power busses. (Not all SBBs have those stripes or +/- signs.) In the center are many short contact strips on either side of the groove that run down the middle of the SBB. The connections do not span the groove.

Components on an SBB

Figure 3 is a detailed view of components mounted on a solderless breadboard. Note how the IC is inserted so that its rows of pins are on either side of the groove. The SBB is designed for easy use of DIP (dual inline package) ICs such as the one shown. The contact holes are spaced 0.1 inches apart from each other and 0.300 inches across the groove.

It's easy to mount leaded parts such as resistors, capacitors, diodes, and ICs on an SBB. However, switches can be a problem; many are not designed to fit an SBB. It's also difficult to connect stranded wire leads such as on a battery connector to an SBB. For stranded wire, something like a screw-terminal would be handy. This is where the Proto Buddy comes in.

The Proto Buddy

The Proto Buddy board (PRB) is shown in Figure 4. It has two SPST switches, two N.O. momentary pushbutton switches, two LEDs with 1K current limiting resistors, and a three-position terminal-block for stranded wire. It uses three sets of pins along one edge to plug into the connector strip on one edge of the SBB as shown. All components on the PRB are wired to pin jacks along its edge. Pieces of solid hook-up wire are used to connect the PRB to the SBB. A Proto Buddy attached to an SBB costs a lot less than the powered breadboards sold by many vendors but with a battery or two attached to it the combo has a lot of the same functionality as those expensive units.

Figure 5 is the schematic of the PRB. The two pushbutton switches are connected by a removable jumper, as are the two LED circuits. The jumpers are there for convenience. For pushbutton switches, it's often the case that both connect to ground. For LEDs, it's often the case that both anodes are connected to +V through resistors. The jumpers can eliminate a few wires in many circuits.
Construction

The components of the Proto Buddy are mounted on a double-sided printed circuit board (PCB) as shown in Figure 6. The side with PRB-1 printed on it is the component side (C side). The other side is the solder side (S side). As the names imply, components are mounted on the C side and soldered on the S side. However, three sets of mounting pins will be mounted on the S side and soldered on the C side.

Start by inserting the 15 pin jacks into the PCB.

CAUTION: Make sure you have inserted them on the correct side of the board before soldering. The pin jacks sit loosely in their holes, so they must be held in place before turning over the board to solder them. I used an emery board held by a metal binder clip as shown in Figure 7. You could use anything of that size as long as the material doesn’t melt; maybe a popsicle stick or a piece of scrap PCB.

Once the pin jacks are soldered in place, mount the two position DIP switch (S1 and S2) and bend the four pins to secure it to the board. Insert S3 and S4; they have springy leads that should hold them on the board without bending the leads over. Turn the board over and solder all the switches.

The LEDs must be oriented correctly before mounting. One LED lead is longer than the other. The long lead is the anode and the short lead is the cathode. The cathode lead is closest to the pin jack. If you look carefully, you might see a flat section on the plastic base of the LED next to the cathode lead. (Refer to Figure 6).

Mount the two LEDs and bend their leads slightly to hold them in place. Mount the two resistors and bend their leads slightly. Turn the board over and solder the leads. Clip off any excess lead length.

The three position terminal block is mounted next. Its leads are too heavy to bend, so secure it to the board with rubber bands. Make sure the bands are positioned away from the pins so they don’t melt, then solder them to the board. Remove the rubber bands.

The two jumpers (designated jmp) are implemented with pairs of header pins on 0.1 inch centers. Usually header pins come in multiple pin strips (e.g., an eight pin header strip). Such strips are designed to be snapped into pieces, so snap off two pairs. You can use a bit of putty to hold the pairs of pins to the board. Solder them to the board and remove the putty. Push a jumper block onto each pair of header pins.

Three pairs of header pins are used to hold the Proto Buddy to the SBB; they are mounted on the S side of the board and soldered on the C side. Follow the same procedure used for the jumper pins.

The soldering is done. Use some rubbing alcohol and an old toothbrush to remove any flux.
residue and examine the board for missing or bad solder connections; fix any bad ones.

The final step in construction is to add the two “legs.” For the Proto Buddy to be level with the SBB, two aluminum, male/female, 4-40 threaded hex standoffs are required: one in each of the two corner holes on the PCB board as shown in Figure 8. Note the nylon screw used to prevent scratching when the Proto Buddy sits on a table. (The hardware in Figure 8 is shown loose.

Testing

Mount the Proto Buddy to a solderless breadboard by inserting the three sets of mounting pins into the contact strip on the edge of the SBB (refer to Figure 4). Connect a 9V battery to the terminal block (also shown in Figure 4). Cut six four inch pieces from a roll of 22 gauge solid insulated hook-up wire. Strip off about 1/4 inch of insulation from both ends of each piece.

Insert one end of a wire into the pin jack connected to the positive (+) side of the battery and connect the other end to one of the LED pin jacks marked with ‘J’. Use another piece of wire to connect one of the S1 pin jacks to the L1 pin jack marked with ‘C’. Use another piece of wire to connect one of the S2 pin jacks to the L2 pin jack marked with ‘C’. Use two pieces of wire to connect the other pin jacks of S1 and S2 to a single connector strip on the SBB. Use a piece of wire to connect the same connector strip to the pin jack that goes to the negative (-) side of the battery.

Figure 9 shows the circuit you just built. As you operate S1, L1 should go on and off. Likewise, as you operate S2, L2 should go on and off. Once you verify the circuit works, rewire it to use S3 and S4 instead of S1 and S2.

Build a Digital Circuit

We will build a circuit called a latch using two transistors. Inputs are the two pushbutton switches and outputs are the two LEDs. Figure 10 is a schematic while Figure 11 shows the wiring. Q1 and Q2 are NPN transistors such as PN2222 or 2N3904. Almost any NPN will do. You can find datasheets on the Internet.

When you first apply power, one LED will be lit and the other will be off. Momentarily press the pushbutton
going to the of LED; it will become lit and the other one will go off. It will stay that way until you press the other pushbutton. The circuit is often called an SR Latch where S and R stand for SET and RESET. It’s also a flip-flop: one bit of memory.

Build an LED Blinker

Just for fun, let’s build a circuit that will blink the two LEDs in an alternate pattern; when one is on, the other will be off. We will use a CMOS digital IC: the CD4093. The IC contains four two input NAND gates as shown in Figure 12.

The signals in digital ICs are either high (+V) or low (0V). The output of a gate will switch between high and low, depending on whether its inputs are high or low. The input-output behavior of a gate is defined by its truth table. Table 1 is the truth table for a two-input NAND gate, where A and B are the inputs and X is the output. A 0 represents low (0V) while a 1 represents high (+V).

The CD4093 has what’s called Schmitt trigger inputs. That means the value of the input voltage required for a 1 depends on whether the input is switching from low to high, or from high to low. It’s meant to guarantee a “clean” transition on the output, but, as we will see, it also allows us to build a simple square wave oscillator.

The circuit you will build is shown in Figure 13. Unlike Figure 11, this circuit is shown as a schematic. Part of your job will be to translate Figure 11 into something like Figure 11 before you start to build the circuit on your solderless breadboard. Q1 and Q2 are MOSFET transistors. Use a device like the 2N7000 or the BS170. Use a 1 μF capacitor for C1; either monolithic or tantalum. For R1, start out with a 1 megohm resistor which will give a slow blink. Try using smaller resistors to make it blink faster. When the LEDs blink fast enough, they will appear to be on all the time.

Wrap-up

Now that you’ve got your Proto Buddy and an SBB, you can breadboard all sorts of circuits. You can find many circuits in issues of Nuts & Volts, and on the Internet. It’s even more fun to dream up your own circuits and try them out. Instead of a battery, you can get a wall wart power module from many vendors; the surplus units are usually inexpensive. So, go have some fun with your new Buddy.

![A complete kit for this project can be purchased from the Nuts & Volts Webstore](https://www.nutsvolts.com) or call our order desk, 800 783-4624.

### PARTS LIST

<table>
<thead>
<tr>
<th>ITEM</th>
<th>QTY</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1, S2</td>
<td>1</td>
<td>DIP Switch, Two SPST Switches</td>
</tr>
<tr>
<td>S3, S4</td>
<td>2</td>
<td>N.O. Momentary Push Button Switch</td>
</tr>
<tr>
<td>TB</td>
<td>1</td>
<td>Three Position Terminal Block</td>
</tr>
<tr>
<td>L1, L2</td>
<td>2</td>
<td>LED, T1 Bright</td>
</tr>
<tr>
<td>R1, R2</td>
<td>2</td>
<td>Resistor, 1K , 1/4W, 5%</td>
</tr>
<tr>
<td>R1 (EXP)</td>
<td>1</td>
<td>Resistor, 1M , 1/4W, 5%</td>
</tr>
<tr>
<td>C1 (EXP)</td>
<td>1</td>
<td>Capacitor, 1 μF</td>
</tr>
<tr>
<td>IC1 (EXP)</td>
<td>1</td>
<td>CD4093B</td>
</tr>
<tr>
<td>Q1, Q2 (EXP)</td>
<td>2</td>
<td>Transistor 2N7000</td>
</tr>
<tr>
<td>N/A</td>
<td>15</td>
<td>Pin Jacks</td>
</tr>
<tr>
<td>N/A</td>
<td>10</td>
<td>Header Pins (Five Pair)</td>
</tr>
<tr>
<td>N/A</td>
<td>2</td>
<td>Jumper Block</td>
</tr>
<tr>
<td>N/A</td>
<td>2</td>
<td>HEX Standoff, M/F, 4-40, 1/4 in, 3/8 in long</td>
</tr>
<tr>
<td>N/A</td>
<td>2</td>
<td>4-40 Nut</td>
</tr>
<tr>
<td>N/A</td>
<td>2</td>
<td>#4 Lock Washer</td>
</tr>
<tr>
<td>N/A</td>
<td>1</td>
<td>Printed Circuit Board</td>
</tr>
</tbody>
</table>

It might be useful to include some 22 gauge, solid, insulated hook-up wire; maybe five feet of red, five feet of black, and five feet of yellow.

January 2009  NUTSVOLTS  47
Gizmos that bring your dreams to life!

Surplus Gizmos Your Science and Hobby Electronics Superstore

DO YOU NEED A GIZMO?

Retro Civil Defense Radiation Detector $25.00
Power Supplies 9V 1 Amp $4
12V 2 Amp $10
Weather Stations from $35

Robot Wheel from $5.00
Robot Arm Kit $54
Multimeter Big display $7.50
100Ks of LED in stock
10Mhz oscilloscope $140
2Mhz oscilloscope $270
USB oscilloscope $220
1/2 watt $5
1 watt $20
RGB 5mm $1.50
UV 5mm $5.50

Solar Kits $10
Hobby Motors $2

WWW.SurplusGizmos.com
503-439-1249
5797 NW Cornelius Pass Road
Hillsboro Oregon 97124

Order online at: www.melabs.com
Development Tools for PIC® MCUs

USB Programmer for PIC® MCUs $89.95
RoHS Compliant
Programs PIC MCUs including low-voltage (2.5V) devices
Includes Software for Windows 98, Me, NT, XP and Vista
With Accessories for $19.95
Includes Programmer, Software, USB Cable, and Programming Adapter for 9 to 40-pin DIP

LAB-X Experimenter Boards Pre-Assembled Board
Available for 8, 14, 18, 28, and 40-pin PIC® MCUs
2-line, 20-character LCD Module
9-pin Serial Port
Sample Programs
Full Schematic Diagram

PICPROTO™ Prototyping Boards
Double-sided with Plate-Thru Holes
Circuitry for Power Supply and Clock
Large Prototype Area
Boards Available for Most PIC® MCUs
Documentation and Schematic

Phone: (719) 520-5323
Fax: (719) 520-1657
In Colorado 80906

See our full range of products, including books, accessories, and components at:
www.melabs.com

For Print Pages

ELECTRONIX EXPRESS

RSR... 3Mhz Sweep Function Generators & Testing Functions Test
Counter, timing sweep
MODEL P03-30
MODEL P03-30
(No Digital Display)
(With Digital Display)

INSTEK VOLTMETER
MODEL VM2003
Model VM2004
2-Line, 20-character LCD Module
9-pin Serial Port
Sample Programs
Full Schematic Diagram

BENCH DMM with RS232 Interface
MODEL 98438B
True RMS, digital and bar graph display, AC/DC Cap, Res., Frequency functions. Includes software. AC or DC operation.

DC Power Supplies
MODEL HY2003 - Digital Supply
MODEL HY2003-2 - Triple Output

Wallace® Soldering Station
#1 Best Selling Station
Model WALL 100

3-WIRE IRON

ALLIGATOR LEADS
SET OF 10

950°

DIODE MULTIMETER
32 Ranges; 3½ Digit
MODEL 41V-64
AC/DC Volt, Current, Res., Frequency, Rubber Holder Included

POTENTIOMETERS
13 1Ω 10Ω 100Ω

Ceramic (SST Series)
85Ω 756Ω 15Ω

Multiturn (MRS Series)
2Ω 15Ω 15Ω

Panel Mount (PA Series)
1.15 85Ω 756Ω

Standard Values Available

SOUND SENSOR CAR
Requires Soldering Reverse
Reverses direction when it detects range or touches an obstacle

ROSS®... Digital Multimeter
10½

SUPER ECONOMY MODEL 4200

Less-More

More-Less

Price Range

300 Page Catalog

Junior High Performance

High Performance

In NJ: 732-381-8020
365 Blair Road* Avenel, NJ 07001
800-972-2225
Fax: 732-381-1106

January 2009
http://www.explex.com
email: electron@explex.com

$35.00

$30.00

$145.00

$184.00

$139.00

$305.00

$5.00

$2.00

$10.00

$1.25

$1.00

$1.25

$5.00

$5.00

$1.50

$1.00

 потерапию

ятолити

WWW.SurplusGizmos.com
503-439-1249
5797 NW Cornelius Pass Road
Hillsboro Oregon 97124

Order online at: www.melabs.com
Development Tools for PIC® MCUs

USB Programmer for PIC® MCUs $89.95
RoHS Compliant
Programs PIC MCUs including low-voltage (2.5V) devices
Includes Software for Windows 98, Me, NT, XP and Vista
With Accessories for $19.95
Includes Programmer, Software, USB Cable, and Programming Adapter for 9 to 40-pin DIP

LAB-X Experimenter Boards Pre-Assembled Board
Available for 8, 14, 18, 28, and 40-pin PIC® MCUs
2-line, 20-character LCD Module
9-pin Serial Port
Sample Programs
Full Schematic Diagram

PICPROTO™ Prototyping Boards
Double-sided with Plate-Thru Holes
Circuitry for Power Supply and Clock
Large Prototype Area
Boards Available for Most PIC® MCUs
Documentation and Schematic

Phone: (719) 520-5323
Fax: (719) 520-1657
In Colorado 80906

See our full range of products, including books, accessories, and components at:
www.melabs.com

For Print Pages

ELECTRONIX EXPRESS

RSR... 3Mhz Sweep Function Generators & Testing Functions Test
Counter, timing sweep
MODEL P03-30
MODEL P03-30
(No Digital Display)
(With Digital Display)

INSTEK VOLTMETER
MODEL VM2003
Model VM2004
2-Line, 20-character LCD Module
9-pin Serial Port
Sample Programs
Full Schematic Diagram

BENCH DMM with RS232 Interface
MODEL 98438B
True RMS, digital and bar graph display, AC/DC Cap, Res., Frequency functions. Includes software. AC or DC operation.

DC Power Supplies
MODEL HY2003 - Digital Supply
MODEL HY2003-2 - Triple Output

Wallace® Soldering Station
#1 Best Selling Station
Model WALL 100

3-WIRE IRON

ALLIGATOR LEADS
SET OF 10

950°

DIODE MULTIMETER
32 Ranges; 3½ Digit
MODEL 41V-64
AC/DC Volt, Current, Res., Frequency, Rubber Holder Included

POTENTIOMETERS
13 1Ω 10Ω 100Ω

Ceramic (SST Series)
85Ω 756Ω 15Ω

Multiturn (MRS Series)
2Ω 15Ω 15Ω

Panel Mount (PA Series)
1.15 85Ω 756Ω

Standard Values Available

SOUND SENSOR CAR
Requires Soldering Reverse
Reverses direction when it detects range or touches an obstacle

ROSS®... Digital Multimeter
10½

SUPER ECONOMY MODEL 4200

Less-More

More-Less

Price Range

300 Page Catalog

Junior High Performance

High Performance

In NJ: 732-381-8020
365 Blair Road* Avenel, NJ 07001
800-972-2225
Fax: 732-381-1106

January 2009
http://www.explex.com
email: electron@explex.com

$35.00

$30.00

$145.00

$184.00

$139.00

$305.00

$5.00

$2.00

$10.00

$1.25

$1.00

$1.25

$5.00

$5.00

$1.50

$1.00

$1.25

$5.00

$5.00

$1.50

$1.00

$1.25

$5.00
Power MOSFET Basics

Power MOSFETs (HEXFETs are International Rectifier’s trademarked name for their products) are typically used in power switching applications and are classified as transistors. They have three leads like a transistor but are voltage in, current out devices. This is because the gate is completely isolated from the rest of the device. No significant DC current can flow from the gate to either the source or drain. Instead, the charge on the gate affects the conductivity between the drain and source. This is like the charge on one capacitor plate affecting the other plate. Typically, about eight volts will turn on the device completely. When on, the part exhibits a low resistance without any non-linear forward voltage drop as seen in bipolar transistors. Modern MOSFETs can have on resistances of less than 10 milliohms.

A little math shows that this device can handle 10 amps with one watt converted into waste heat (power = current^2 x resistance). Since many MOSFETs come in TO-220 packages, no heatsink is needed in this instance. So, if the voltage is 100 volts at 10 amps, then 1,000 watts of power are switched with only one watt lost. That’s 99.9% power efficiency. The IRFB4410 has these specifications and costs about $4.50. That’s fairly expensive for a power MOSFET.

The resistance in the off state is so high that it is usually not stated in the datasheet. Instead, they typically define the breakdown voltage as when 250 μA of current flows through the part. For most practical purposes, the power MOSFET can be considered a switch: either it’s on or it’s off. (Next time, we’ll look at linear/non-switching applications.) However, the key point is that it takes some time to go from a very high resistance to a very low resistance. This switching time determines the efficiency of the system and will be examined in more detail shortly.

MOSFETs come in two flavors: P-channel and N-channel. However, because of the physics involved, the P-channel types cannot match the low on resistance of the N-channel type. For that reason, there are many more N-channel parts available and at a lower cost. Most designs will use an N-channel device even if it requires additional effort. P-channel parts are not often seen except in special applications.

Unlike bipolar transistors, MOSFETs have a positive temperature coefficient. This means that their resistance increases with temperature. This can be extremely useful. As they heat up, they impede current flow more which tends to stabilize the system. Bipolar transistors allow more current to flow as they get hot. This increased current heats them up more so that they pass more current, which further increases their heat and so forth and so on. This is also called thermal runaway. When it happens, the transistor is usually lost – possibly along with additional downstream damage.

The positive temperature coefficient means that paralleling identical MOSFETs for additional power is relatively easy. If one device gets too hot, its increased resistance effectively pushes excess current to the other parts auto-
matically. In fact, this is how power MOSFETs are made. There are hundreds or even thousands of tiny MOSFETs connected in parallel to form a high power device.

**Using MOSFETs**

First of all, just because they are power products doesn't mean that they are immune to static. In particular, the gate is isolated from the source and drain by an incredibly thin layer of insulating oxide. If this layer is pierced by an Electrostatic Discharge (ESD), it can seriously affect how well the part operates. Voltage spikes above the rated gate voltage — due to poor board layout or circuit design — are also something to be avoided for the same reason.

The key concept in using MOSFETs is that the gate voltage controls the resistance between the source and drain. No real current is required for DC operation. This is seen in Figure 1. Here, reversed biased diodes are used to supply the gate voltage. The current through a reversed biased 1N4148 is estimated to be about 10 nanoamps, in this instance. So, you can see that it doesn't take much to turn a power MOSFET on or off. If you actually try out Figure 2, you will see that it takes several seconds to go from off to on and vice versa. It may take much longer if the part has been in one state for a long time. Note that the meter reads zero volts when the part is conducting or "on."

The reason why it takes time to switch is because all of those paralleled and isolated gates act like the plate of a capacitor. Probably the most important characteristic of the gate is its capacitance. For the IRF540 shown in Figure 1 (which costs about $0.75), the on resistance is 0.077 ohms and the gate capacitance is 1,500 pF. Generally, the lower the on resistance the greater the input capacitance. This makes perfect sense. In order to get a lower resistance, you have to parallel more tiny devices. This means more gates and more gate surface area. This increased gate surface area translates to a larger capacitance.

This brings us to the basic practical concern in the use of power MOSFETs. In order to switch the device on or off, you have to charge and discharge the gate/capacitor quickly. If you don't, it will spend considerable time in the linear region and dissipate a great deal of heat. Therefore, to turn it on and off quickly, you have to supply a very low resistance path. This is because gate/capacitor and drive resistance combine to create an RC network.

This RC network determines how quickly the part can change state. This brings us to the reverse drive characteristics of the MOSFET. While it takes virtually no continuous current to turn the device on and off, it does take considerable pulse current to charge and discharge the gate quickly. Hundreds of millamps or more are often needed for very fast switching. MOSFETs themselves can switch very quickly — often in 20 ns. To do so requires a powerful kick to the gate.

**Other Considerations**

Before we get into the drive details, it is important to mention some other points. Power MOSFETs are often used in motor-control applications. They can be ideal for this. It is imperative, however, to remember and design for inductive kickback. With high voltages and high currents, these spikes can be devastating to your circuit. They're too involved to describe here so refer to the manufacturer's application notes for guidance. This is something that they have studied in detail.

Then, there is the point of heat dissipation. Using these parts at high power can certainly result in many watts of power being generated. Many parts are rated for up to 150 watts of power dissipation. Again, this discussion on heat control is too involved. Refer to the datasheet and manufacturer's application notes.

Over-driving the gate can significantly reduce the expected lifetime of the part. For example, the IR datasheet shows that 99% of typical IRF540 parts will last about 100,000,000,000 hours (11.4 million years) with an eight volt gate drive (at 150 degrees F). Driving the gate with 20 volts reduces the time to only 1,000,000 hours (114 years).

The required drive speed depends upon the application. If you are just switching a load on and off every second or so, it doesn't really matter much if the MOSFET switches in 50 ns or 50 μs. The amount of heat generated (and energy lost) during the transition is a tiny fraction in either case. However, if you are switching at 100 kHz, a 50 μs transition time simply won't work. Even a 1 μs transition time at 100 kHz means that the MOSFET is in the linear region 10% of the total time.
Low-Side Driving

The simplest and most common circuit is called 'low-side' driving (see Figure 2A). (Note, we will mostly limit the discussion to N-channel devices for brevity and simplicity.) In this case, the MOSFET is connected directly to ground. High-side driving (Figure 2B) places the load at ground and the MOSFET connects to the power source. To turn on a low-side MOSFET, all you have to do is raise the gate above eight volts above ground. Grounding the gate turns it off. It is driveable to use five volt TTL logic level signals (a.k.a., microprocessor) to directly drive a MOSFET. However, this won't turn the device completely on. Nevertheless, oftentimes this is sufficient. For the IRF540, a five volt gate drive will allow about 10 amps to be switched (typically) instead of the 28 amps specified. So, if your application doesn't require the full power of the part, TTL signals may work. You can always use an 'open-collector' part that allows you to pull up the logic output above five volts. Also, there are special MOSFETs that are specified to operate with a five volt gate drive. Naturally, they're more expensive, but they may be worthwhile considering the additional expense and difficulty of designing a higher voltage gate drive circuit. Typically, these are identified as 'logic-level' devices.

CMOS logic has the advantage of being able to operate on eight volts or more without a problem. However, they are terrible when it comes to drive current— even with paralleled outputs. Typically, they only provide a couple of mA or so per output. So, it's difficult to drive a MOSFET gate at high speed. Many applications don't need high speed switching, however.

A 555 timer works quite well as a driver. Be sure to use a bipolar part (NE555) rather than a CMOS part (sxC555) (see Photos 1 and 2). There are also many discrete transistor drivers as in Figure 3. The drawback with this design is that the pullup resistor limits the current so the turn on speed is slower than the turn off speed. A totem pole design (Figure 4) can be very effective. You can also use a NPN/PNP design to eliminate the need for the inverter.

High Side Driving

High side driving of an N-channel part can be tricky (see Figure 2). The gate has to be about eight volts above the source voltage to turn it on. However, because of the low resistance when it is on, there is very little voltage drop between the drain and source. Thus, the source pin voltage is often very close to VCC. So, to turn on the device you may need a gate voltage greater than VCC. There are some ways around this problem. The first is to build a voltage multiplier. Obviously, this is not an elegant solution. A P-channel part might be the easy solution here despite the higher cost and poorer performance. Figure 5 shows the typical hook-up. Note that the source is connected to the positive voltage. In this configuration, the P-channel device will turn on with a gate voltage eight volts below the source pin. So, if VCC/source is 10 volts, the part will start to conduct when the gate drops to about seven volts and will be fully

![Figure 6. This bootstrap design increases the gate voltage but is very slow ... about 30 μs turn on time. Voltage increase is determined by the ratio of R1 and R2. R1 tends to pull any gate voltage down to five volts.

![Figure 7. Using an open-collector design eliminates pullup resistor and gate resistor. Switching speed is much faster than Figure 6. Turn-on time is about 4 μs.

![Photo 1. A bipolar 555 timer (Texas Instruments NE555) turns the MOSFET on and off in about 50 ns.](image-url)
on at two volts, or eight volts below the source voltage.

If you are continuously switching the load on and off in less than a second or so, there is another approach that can be used with N-channel parts. This is called bootstrapping and is shown in Figure 6. This is a modified circuit found in the Siliconix MOSPOWER Applications Handbook. The conceptual design of the circuit is subtle but fairly simple. When the transistor is on, the MOSFET's gate is pulled low and the capacitor is charged to VCC (10 volts) through the isolation diode. When the transistor is turned off, the gate drive rises to VCC because of the charged capacitor (mostly via R2 and R3). Since the MOSFET is off, the source pin voltage is pulled low through the load. This means that the gate voltage is well above the source pin voltage and the MOSFET will start to conduct. As this happens, the capacitor acts as a voltage source in series with the source pin. So, any voltage on the source pin is added through the capacitor to the gate. (The negative side of the capacitor gets pushed up by the increasing source pin voltage which pushes up the positive side by an equal amount.) In effect, the part pulls the gate voltage up two voltages (VCC) to VCC, typically.

Naturally, theory and practice are different. The capacitor should be at least 10 times the gate capacitance. In most cases, 0.1 µF will work. The diode is any power type with an appropriate voltage rating. The resistors R1 and R2 are the tricky components.

The pullup resistor (R1) determines how big the voltage increase is. This is because it connects the gate drive to the five volt supply. Any voltage greater than five volts will be pulled down to five volts through this resistor. Note that this resistor may not always be visible. For example, a 555 timer (connected to R2-R3) can supply 100 mA for an equivalent pullup resistance of about 50 ohms (at five volts). Obviously, a 10K resistor above five volts by any significant amount.

As shown in Figure 6, the 10K value for R2 only supplies 10 volts (referenced to ground) to the gate which is not adequate if the source is also at 10 volts. If R2 is increased to 100K ohms, over 17 volts is applied to the gate which is probably okay for most applications with the IRF540. Note that the turn on time is also controlled by R2 (the turn off time is controlled by R3). Charging the 1,500 pF gate through 100K takes about 30 µs to turn on the device (measured). So, you trade off speed versus voltage. The general rule of thumb is that R2 should be about 1/10 of the equivalent pull up resistor, R1.

These problems can be eliminated by using an open-collector transistor circuit which is shown in Figure 7. In this case, there is no connection to the five volt supply so there is no pulldown problem. This eliminates R1 and allows the use of a much smaller resistor for R2. This resistor is now chosen to limit the current into the transistor to a safe level (100 mA as shown). Resistor R3 can also be eliminated. This circuit provides about 18 volts to the gate and switching time is about 4 µs for turn on and 500 ns for turn off.

However, if you want to use a high-side N-channel part you should really consider using a driver chip. The LM5109B only costs about $1.60 and drives a high-side and low-side MOSFET in a half-bridge configuration. It's rated for 90 volts (to the MOSFET) and can turn them on and off in 15 ns with 1.0 pF gate capacitance. Considering the time and effort in designing your own high-side driver, this is a good deal. There are lots of other parts available, as well.

**Conclusion**

This month we've looked at power MOSFETs and found that they have some very useful attributes. They're cheap and powerful, and can be fairly easy to implement. Naturally, there are practical considerations in addition to the theory.

Next time, we will build two projects. The first is a transformerless, high current voltage doubler using a full-bridge design that has many other useful applications (like motor control). The second is a linear, constant current power supply capable of providing 20 amps or more.

**TABLE 1. Low-side drive summary (times are measured at load).**

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Turn off time</th>
<th>Turn on time</th>
<th>Comments (10V D-S with 100 mA load)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TTL LS04</td>
<td>500 ns</td>
<td>100,000 ns</td>
<td>Can't drive fully. Logic-level parts available.</td>
</tr>
<tr>
<td>TTL/pullup</td>
<td>200 ns</td>
<td>3,000 nsS</td>
<td>Speed limited by pullup 1K resistor. (7414S)</td>
</tr>
<tr>
<td>CMOS</td>
<td>8,000 ns</td>
<td>1,000 ns</td>
<td>Very slow but turns on all the way. Easy. (CD4069)</td>
</tr>
<tr>
<td>CMOS x 6</td>
<td>2,000 ns</td>
<td>400 ns</td>
<td>Better than above, but still slow. (CD4069)</td>
</tr>
<tr>
<td>N555</td>
<td>175 ns</td>
<td>60 ns</td>
<td>Bipolar good, CMOS poor. (Photos 1 and 2)</td>
</tr>
<tr>
<td>DISCREET</td>
<td>400 ns</td>
<td>2,500 ns</td>
<td>Speed limited by 1K pullup resistor. (Figure 4)</td>
</tr>
<tr>
<td>Totem pole</td>
<td>175 ns</td>
<td>150 ns</td>
<td>Very good. (Figure 5)</td>
</tr>
<tr>
<td>DRIVER</td>
<td>175 ns</td>
<td>50 ns</td>
<td>Best speed (see text). (LM5109B)</td>
</tr>
</tbody>
</table>

January 2009 NUTSVOLTS 53
To demonstrate how SPICE simulation helps even modest hobby electronics projects, let’s take a fresh look at two very well-known analog ICs.

With LTspice running on your computer’s desktop and last month’s article for navigation hints, we make discoveries about IC circuits that we probably use and know well. We expand LTspice with the concepts of sub-circuits and macromodels.

Spicing Up Some Old Friends

Imagine building hobby circuits without voltage regulator ICs and op-amps. Their flexibility and low cost make them popular and useful, and when we require amplification of analog signals, typically we use op-amps. Plus, all circuits operate better from a regulated power supply. Let’s start with the three terminal linear regulator IC.

Even our 9V battery had limitations, as we saw in Part 1. Most of these can be overcome by adding a linear regulator such as the ubiquitous 7805 device. First, we’ll do a bit of hands-on work on the solderless breadboard (see Figure 1).

Using the three terminal 5V 100 mA regulator IC (LM7805 or one of its variants), connect the circuit in Figure 2. There should be no surprises here; the 9V battery powers the regulator which, in turn, outputs 5V. Adding a load resistor at the output does not change the output voltage (significantly), but it would if we’d used a plain dropping resistor in place of the regulator IC. To bring the voltage down from 9V to 5V with a single resistor would require that we know the load current, and assume that it is fairly constant.

I’ve added two capacitors per the datasheet to prevent the part from self-oscillating; also a few extra resistors (R1, R2, and R3) so you can use your DMM to record voltage drops and then calculate currents. This makes it easier than breaking into the circuit with an ammeter (or a DMM on its current ranges) and my measured voltages and calculated currents along with the LTspice results are shown in Figure 3. The downside is that the

regulation performance is degraded (which actually helps us understand the test data and device circuit theory).

Switching over to LTspice, we model the same circuit either by creating a new schematic in the editor or downloading the file NV_SPICE_21.asc from the Nuts & Volts website (www.nutsvolts.com). You will need to add the regulator model to your copy of LTspice by carefully following the steps in the sidebar Adding New Macromodels To LTspice.

The SPICE circuit should look like Figure 4. Notice the voltage regulator shows up as a single block; soon we’ll take a look inside at the IC’s transistor level circuitry. As this is a DC simulation, we don’t need those capacitors required by the breadboard circuit.

We run the simulation to find the DC operating point first, using the now familiar “.op” command and — as expected — the regulator’s output is 5V. Loading the regulator first with 100 ohms (100R) and then with the combination 25R
resistor (R4,5,6,7 in parallel) gives us very similar simulation results to the hands-on experiment (Figure 3 again).

### Deliberate Overload

On both the breadboard and the simulation, we forced the regulator into current limiting by changing the load resistor from 100 ohms down to 25 or so — which I made from four parallel 100 ohm resistors. I chose an LM78L05 with its 100 mA load current spec instead of the more popular one amp LM7805 for demonstration. To force an LM78L05 into current limit, we must draw a minimum of 140 mA, so once again we are taxing our 9V battery (make a quick reading and remove the battery ASAP). If all you have is a plain Jane LM7805, you likely will kill the 9V battery long before reaching the regulator’s current limit (of two amps!)

Although we ran the LTspice simulation twice by changing the load resistor, there are other ways to get there. LTspice has a very handy technique to sweep the voltage source (i.e., our 9V battery equivalent) and plot a time variable graph of the circuit’s output. In effect, we can see how the circuit behaves at power turn on. To do so, change the spice directive from “.op” (DC operating point) to “.tran 100m startup” which tells the simulation to start at zero and run for 100 ms; the battery has been changed to a voltage source that starts at zero and ramps up in 50 ms. First, using the 100R load, look at Figure 5.

Doing this with 25 ohms on the output causes a current overload. We observe the regulator changing to a current-limited output, shown in Figure 6.

### Introducing The NV78L05 and NV7805

Our LTspice schematic uses a simple voltage regulator symbol with three terminals (Vin, Vout, and Comm). Inside, the real IC part is quite complex with more than a dozen transistors and several diodes, as shown in Figure 7. This diagram was lifted from the datasheet for the LM78L05x family, as used in our solderless breadboard circuit.

The popular LM7805 (a one amp minimum output part) diagram is shown in Figure 8. Both were designed and fabricated in a bipolar transistor process, although more modern ICs will likely use FET transistors instead. The real datasheets for these commercial parts are available from the Nuts & Volts website (www.nutsvolts.com) as part of the download for this article.

Notice that I’ve “tuned” these LTspice models a bit because some component values were missing and
that lead me to do a little reverse engineering. For the exercises here, use my NV78L05 and NV7805 components which you can add to your LTspice model library (see the sidebar again.)

More About Sub-Circuits And Models

We can drop that simple three terminal symbol on to our SPICE schematic, just as we can drop the three terminal parts onto the breadboard. In either case, we don't have to wrangle with the internal circuit diagram or even give it much thought. The symbol placed in our LTspice diagram is a SPICE sub-circuit, representing the more complex circuit. It removes a lot of clutter from our schematic, and when we run the simulation LTspice "unpacks" all sub-circuits and runs the whole thing.

The internal diagrams found in datasheets are often called a macromodel. The important point is that for our needs, the macromodel is equivalent to the real device. This leap of faith is a little confusing but we would see why it's done if we could look over the shoulder of the IC design engineer that created these and other IC parts.

Firstly, the IC designer works with a different set of models (to our macromodels) that exist at the silicon transistor level. These device-level models are based upon the IC fabrication process used to build physical parts. Much of this is proprietary information and not of much value to us or anyone else designing at the component or board level. To make reliable ICs that meet or exceed the published datasheet specs requires thorough testing under all extremes of the IC fabrication process — often known as "process corners." Again, this is proprietary and IC process-dependent. I bring it into the discussion to illustrate that as hobbyists we use or create component level diagrams that, in turn, use ICs which are represented by their macromodels. The original IC design was done (often with different simulation tools) at the device level. A lot of work goes into creating good macromodels (which also avoid giving away factory process secrets). We all benefit directly from this effort. Wikipedia has a nice write-up for further reading [http://tinyurl.com/5z2jtv and http://tinyurl.com/56bwqq].

Can I Make My Own IC?

Now that we have a macromodel diagram, why can't we build that circuit on a breadboard? Perhaps we can, and it may even work! I hedge my bets because the breadboard is quite different from a silicon wafer.

Device model simulation is the only practical way to predict how silicon will work, and building the silicon is the only practical way to prove it. Both are costly and complicated.

Before we move on, take a look at Figure 9 — a generic voltage regulator block diagram. Using this as a template, can you identify circuitry blocks in the macromodels? (Look at Figures 7 and 8 again.)

Netlist Vs. GUI — Looking At Source Codes

LTspice is very easy to use (compared to earlier SPICE software) because it has a GUI Graphical User Interface that works with the Microsoft Windows Operating System. If you are interested in reading LTspice project files, all you need is a text editor such as WordPad (bundled with Windows). Within LTspice, there is also a netlist viewer that displays the traditional SPICE netlist, and a tool to export it as a net file.

Load the file NV_SPICE_21.asc, then select SPICE Netlist from the View pull-down menu which will look like Figure 10. The coded lines should be familiar — it's text instructions that mimic what we did with the schematic editor to create the circuit. Using WordPad, other project files can be opened for
viewing and editing (with a bit of caution). These files are vaguely familiar as we can recognize LTspice code. Hint: Any line starting with an asterisk is a comment ignored by LTspice, and lines starting with a dot are instructions to the simulator. The other lines represent nets that connect the nodes together, for example, “R5 Vload 0 100” indicates that resistor R5 is connected from node “Vload” to node “0” (common) and has a value of 100 ohms.

Other elements in an LTspice project are also created with text listings — for example, the macromodel graphic symbol (NV780L05.assy) and the subcircuit (NV780L05.lib) so take a look at these with WordPad. For advanced users of LTspice, it’s possible to create, edit, or just hack existing files as needed.

Many LTspice device models supplied by LTC are proprietary binary files that can’t be viewed or edited, but do have enhanced features to run LTspice much faster and with more accurate simulation than open-source SPICE macromodels.

Quick! Name A Famous Op-Amp IC

Given your age and exposure to electronics you may guess differently, but for a couple of decades the only answer was an LM741 (or its relatives). It has stood the test of time and we’re going to use the LM741 here as it fits our needs well; it’s robust and cheap, and we have a good macromodel of it for LTspice. Install the LM741 op-amp model to your LTspice library by follow the steps in the sidebar. In Figure 11, we see the macromodel circuit that was lifted from the datasheet (included in this article’s download.)

Once again, we’ll rig up the actual IC part on the solderless breadboard and compare results to LTspice. With only a 9V battery and some resistors, an op-amp doesn’t do very much unless you also have access to an audio sinewave generator and a ‘scope. We are limited to just making DC measurements on the breadboard. The breadboard schematic is shown in Figure 13; the LTspice circuit in Figure 14; and our results in Figure 15.

Op-Amp Refresher

Before we dive into the circuit shown in Figures 12 and 13, here’s a very quick op-amp theory refresher. Recall that an op-amp behaves in a circuit as defined by external components (notably the feedback path resistors) up to the point where the op-amp runs out of voltage gain (amplification) which happens to all op-amps sooner or later. If the op-amp is not biased correctly, the supplies are too small or too big; if the load is too heavy at the op-amp’s output, the signal will distort or not appear at all. Popular configurations with an op-amp are either inverting or non-inverting, depending upon where we inject the input signal. The output may have a DC component (by design or by limitation of the op-amp’s internal circuit). Oddly, op-amps don’t have a ground or common terminal, and that point is important when understanding the circuit in Figures 12 and 13. A comforting fact is that an op-amp’s circuit gain is determined by the ratio of just two resistors:

\[ \text{Inverting Gain } \left(\text{Av} \right) = \frac{R1}{R2} \]
\[ \text{Non-Inverting Gain } \left(\text{Av} \right) = 1 + \left(\frac{R2}{R1}\right) \]

Can you predict the op-amp’s output voltage from the circuit? Hints: \( R1 = R5 + R6 \), and \( Rn = R7 \), the bias string (R1, R2, R3, R4) generates a two volt (and an unused seven volt) DC input “signal,” but are dependant upon the health of the battery and actual resistor tolerances. Compare your work with the data in Figure 15.

Sweeping SPICE Changes

The solderless breadboard circuit (Figure 12 again) is recreated in LTspice’s schematic editor (or by downloading the file NV_SPICE_22.asc); see Figure

January 2009

NUTSVOLTS 57
14. We are interested in both the DC behavior (using the _op Spice directive) and the op-amp’s AC performance which would be measured on the bench with test equipment, as noted earlier. LTspice applies a sinewave, calculates the phase relationship of the input and output signals, and everything is plotted graphically. These graphs are commonly called Bode plots, and are explained in detail at http://tinyurl.com /2v4qzt. LTspice does the heavy lifting for our simulation; we just need to tell LTspice what signal to sweep (initial conditions and range). To create the Bode plot in Figure 16, I added an AC swept signal to a second voltage generator that covers 1 Hz to 3 MHz (a bit beyond the LM741’s useful upper frequency.)

**Riding A Bicycle**

If you’ve mastered riding a bicycle, you already know two things: no matter how many books you read about bike riding, you have to actually get on the bike! Also, you’ll fall off a few times as you master the skills. Same with SPICE tools! If you’ve followed this session, you’re ready to try a few simulations of your own, which are waiting to be downloaded from the magazine’s website. These exercises cover audio filters using op-amps, and are based on the work of Sallen and Key as noted in Wikipedia, at http://tinyurl.com/6a67v2

**Wrap-Up For This Session**

LTspice is a versatile simulation tool for all analog circuits, from DC to moderately high frequencies. In each case, we need a representative Macromodel of the components to generate meaningful data. Most (if not all) of the major IC houses provide these models for their products freely on their websites. Higher performance binary models are available for LTC-specific components, and these are already in the component library installed with LTspice. I encourage everyone to play with LTspice to gain confidence; it’s a powerful circuit design tool.

Have a question? Please contact the author via email (psotnord@ix.netcom.com), or join us on-line at the Nuts & Volts forum. 

**Adding New Macromodels To LTspice**

All electrical components used in a SPICE simulation require models: a set of parameters that define how the component behaves in the simulation. Because LTspice uses a GUI, newly added components require two distinct parts: the model data file and the graphical symbol that appears in the schematic.

When LTspice is first installed,

![FIGURE A]

the library (lib directory) is created and contains three separate subdirectories: called cmp, sub, and sym. These contain the standard component definitions, the subcircuits, and the graphical symbol, respectively. For example, if we place an op-amp in our LTspice schematic, it is represented by a graphic symbol; it is composed of a subcircuit of active devices such as diodes and transistors and passive devices connected to nodes by nets; and these devices are, in turn, defined by the standard models from the cmp directory.

**Adding A New Component Model**

To add a new component requires a little bit of file manipulation, and failure to follow these instructions closely will likely
end in a non-functioning component and simulation.
Most (if not all) semiconductor houses provide SPICE macromodels of their products which can be run in various flavors of SPICE (Pspice and Hspice are popular).
Most of the time there won’t be a graphical symbol for the desired device, so we must make one by either drawing it from scratch or modifying an existing symbol. The Copy and Paste method is preferred for obvious reasons! If given a choice, use the Pspice version of the macromodel, as LTspice is based on Pspice. There are third-party companies that construct macromodels, so you may find legal disclaimers or owner’s data as text comments in the files that you download.

Before You Begin

Close LTspice and open Windows Explorer and navigate to the Programs directory on the hard drive. Drill down to LTC, then to SwCADII, and finally to lib; compare your file structure to Figure A.

(1) Create Private Folders
Add a new folder (sometimes called a sub-dir) in the existing sub folder and call it Private; create a new folder in the existing sym folder and also call it Private to contain your private collection of components. By doing it this way, your installation of LTspice will follow the practice of other users, which will help when you share your work with them.

(2) Create A Symbol
The easiest method is to open an existing symbol and save it under a new name in your sym/private folder; remember to change some of its data first.
For the LM741 op-amp, use the LT1013 symbol as follows:
Launch LTspice and resize the frame to about 1/4 the screen size by using the mouse and grabbing the frame’s borders. Next, do the same for Windows Explorer, so that both are visible on your screen.
Search the directory tree to find the LT1013.sch file in the sym/opamps folder, and finally drag it to the LTspice window, where it will open in the LTspice symbol editor (see Figure B).
Maximize the LTspice window and use Ctrl-A (or navigate from the Edit drop-down menu) to invoke the Symbol Attribute Editor (Figure C).
Select the SpiceModel data and retype “Private/LM741.lib”; select Value and retype “LM741/ns”, again for Value2; and finally edit the Description to “Operational Amplifier” (Figure D). Hit OK and from File, select “Save As” and change to your sym/private folder. Save the new symbol as “LM741.sch”. You now have a new component symbol!

(3) Obtain A Macromodel From The Manufacturer Or Third Party
Using a search engine, locate the SPICE model of the part you need. In this example, I found the file on the National Semiconductor site (http://tinyurl.com/5g2bj8).

The required model is called LM741.mod, and it should be saved to your sub/private folder and renamed Nat.lib. If you collect other model subcircuits from National Semiconductor, append them to the same file using a text editor such as Wordpad, creating a single file for one vendor. In the sub folder, you will find several LTC*.lib (where * is a digit) files, containing all the sub-circuits for Linear Technology. Open one with Wordpad to see how the individual devices are presented. Hint: Look for the tags:
SUBCKT (devicename) 1 2 3 4 5 6 7 8 ENDS (devicename)

(4) Close And Re-Open LTspice
I spent quite a bit of time trying to make my new components play in LTspice before I discovered that the LTspice program must be closed and reopened to activate these edits!

(5) Check Your Work
Start a new schematic in LTspice and place your new component. For a very quick sanity check, also place a ground symbol and connect all the new component pins together and to ground (Figure E). Add the "op" spice directive and run the simulation. If all is well, LTspice will run the circuit without any error messages (the output data window will be meaningless). On first use, this process will seem frustratingly complex. However, after doing a few you will quickly learn the steps and be able to add new components effortlessly. If you are simulating the examples used in this series, you will need to install the components from the Nuts & Volts website download for these articles.
"How To" BASICS

Surface mount soldering

BY ROBERT L. DOERR

Some people tend to shy away from using surface mount components in their projects. It seems to be too difficult or needs an array of specialized equipment. In the past, I found myself in this same mindset — wary of using these types of parts. That all changed when I got involved with an open source motor controller project (OSMC). It used a mix of surface mount and through-hole components. I took the plunge and built up a pair of OSMC H-bridge boards and the MOB (Modular OSMC Brain) controller board which I used in the Battlebot Crash Test Junior. At the time, there was little information available on how to mount these parts using an ordinary soldering station and tools that most hobbyists would have on hand. Not wanting to invest in a whole new set of tools (hot air stations, etc.), I experimented a bit and used common sense techniques to get the job done. A point I'd like to stress is the myth about requiring anything exotic to work with most surface-mount parts. I don't own or use any special soldering equipment for this. All of the soldering that I've done on surface mount boards is built with an old Weller WTCPT station and TC201 soldering iron. It has the fine tip that came standard on it. If you happen to have access to specialized tools, go ahead and use them but you still may find these tips helpful. It has been my experience that depending upon your techniques, you can get by just fine in most instances without specialized tools. The techniques and methodology you use when initially assembling a project using surface mount parts can make all the difference! It takes a little getting used to but quickly becomes second nature. It does require good eyesight (or a big magnifying glass) and a steady hand though. Now, if you start talking about parts that use a ball grid array (BGA) connection or others like that then you would need special tools. Careful selection of parts to avoid difficult package styles can make using tools most people already have on their bench a viable option.

Surface mount parts can be soldered with

- A USB interface for my HERO 1 robot
  (surface mount and through-hole parts).
traditional hobbyist soldering equipment. Some of the construction methods are very similar to traditional methods but are a bit more delicate. Tools and supplies you should have on hand are:

- Soldering iron (pencil type) with fine tip
- Needle nose pliers
- Diagonal cutter
- Tweezers
- Vise (or something similar to hold a PCB)
- X-acto (or utility blade)
- Good quality solder (60/40 rosin core preferred);
- RoHS can be used
- Solder wick
- Liquid/paste solder flux (no acid flux or solder paste!)
- Magnifying glass
- Des-fluxing spray (or rubbing alcohol)
- Cotton swaps

NOTE: You'll find that the results tend to be better with standard tin/lead based solder instead of the newer RoHS formulas. The RoHS based solder seems to do well with small caps and resistors, but the lead based solder is still easier to work with. It just seems to flow better and provides a much nicer looking solder connection. The RoHS solder ends up looking like a cold solder joint even when it is actually okay.

**Parts Suitable For Soldering With Standard Equipment**

Most surface mount parts can be used with standard soldering equipment, but there are a few exceptions. Parts that come in a BGA or QFN (quad flat no leads) package have pads underneath the part which make them almost impossible to solder without using an oven. Other package styles like the SSOP, QFP (quad flat pack), TQFP, TSOP, SOIC (small outline IC), or SOT (small outline transistor) have exposed leads that can be easily soldered. (There is a link at the end of the article to Wikipedia that has a nice description of each package style.) Standard chip capacitors and resistors which just have a single connection on each exposed end are probably the easiest of all to work with. As long as you can get to the leads with the tip of the iron, you should be able to solder them.

How small is too small? When building with surface mount parts, about the smallest I work with are the 1206 capacitors and resistors. I have used some of the tiny 0603 parts, but they are hard to hold onto. It felt like I was trying to build a ship in a bottle. It was necessary to have a few spares on hand, because if one was dropped it was gone, never to be seen again. They are quite difficult to pick up, so I only use ones that little if I really have to.

**It's All In The Technique**

You don't need any previous surface mount experience and as long as you are adept at soldering you can learn to work with these parts. There are a couple different methods that can work well for hobbyists. One uses a hacked toaster oven to solder the boards. The oven method uses solder paste which goes on all the pads and contains both solder and paste to hold the parts in place. The parts are then set into the solder paste and a hacked toaster oven heats the whole thing to solder them. I've never used that method, so I won't focus any more on that here. For those interested in using an oven, there are links at the end of the article to more detailed information online.

Do a search for surface mount components on the Nuts & Volts website (www.nutsvolts.com) and you'll find all sorts of info. There was also an article in the June 2003 Nuts & Volts which specifically covers modifying a toaster oven for this exact purpose. Back issues are available for sale in the Nuts & Volts webstore at
Do not rush! After perfecting the techniques, you'll naturally get faster along the way. A good surface mount solder connection forms the electrical connection between two parts, such as a component lead and a circuit board foil. With surface mount parts, it also provides the mechanical connection, as well. Make sure there is enough solder to leave a nice fillet between the component lead and the pad.

**NOTE:** Before installing any components, plan ahead and think about the final placement of each part and what the final board will look like. With that in mind, it should help with the assembly order so that each component will not block the next one installed.

The alignment of the parts is critical. Small components like the 1206 capacitors and resistors are more forgiving and are good to start out with. Others like a TQFP microcontroller are absolutely critical in regards to placement with their fine pitch leads. Before installing a surface mount component, apply a small amount of solder to one (and only one) of the pads on the PCB where the part is going to be installed. Then, while warming the solder on that pad use the tweezers to set the part in place. If the alignment isn’t right, you can do adjustments while the solder is still molten. If it takes too long, let the solder cool to ensure the part is not damaged by excessive heat. Once cool, the joint can be warmed up again and the alignment can be adjusted until it is perfect. Before soldering any other joints on the part, use the magnifying glass to verify that the alignment is okay. At the moment, the exact orientation of the part in relation to the pads is what we’re concerned with. Next, go ahead and solder an opposing lead on the part. As long as everything looks good, continue soldering each one.
I usually go back and re-solder the first pin since it may be messed up from re-positioning the part.

NOTE: Some people prefer to use a small dot of glue to hold each component in place before soldering. If this works out well for you, use that method but keep in mind it adds another step and can make rework more troublesome down the road if you have to replace any parts.

**Working with Fine Pitch Parts**

Care needs to be taken to ensure that there are no solder bridges causing shorts. When dealing with fine pitch parts, do not be concerned about solder bridges at first. Since the pitch is so close on these pins, just solder each side without worrying about solder bridges and let it cool before going on to the next side. When done, let the whole component cool. Then, go back over each side with solder wick to remove the excess solder. This will remove most of the solder and leave just enough to make a proper connection. Repeat the process if solder bridges still exist.

Solder wick is your friend when working with surface mount parts. If you ever get a bridge or too much solder on a connection, just use the wick to remove the excess. In the event you remove too much, you can always add more solder and repeat the process. Be sure to let the component cool between each side so the part will not be damaged by excessive heat. The results can be fantastic if you take your time.

**Replacing Surface Mount Parts**

Sometimes you may need to replace a surface mount component. There is one method I have found works very well for fine pitch parts. I use a sharp utility knife to cut the leads right at the body of the part. It requires a steady hand since

- Small hybrid modules assembled with ordinary soldering gear.
for the installation of the new chip.

**Other Uses For Surface Mount Parts**

Some surface mount parts are particularly useful for making corrections and upgrades to existing circuits and during prototyping. Since they are small, you can easily use 1206 resistors as pull-up resistors or even use the small caps between existing pins for improved filtering. They can really help. Since they mount on the surface of the board, you can use them on either side which opens up more options.

**Conclusion**

Hopefully, this has provided a good overview of how you can assemble boards that have surface mount components using many of the tools you already have on hand. You can do it! Just start with some of the surface mount caps and resistors in the 1206 package style. That way, you can start with some of the easier parts and get confidence up. Once you get used to them, you can try other components with fine pitch leads. If you lay out your own boards, you can easily place 1206 pads right along with the pads for through-hole components so that either style can be used.

I’ve found that some simple parts like the surface mount caps and resistors can be installed faster than their through-hole companions and can reduce the time to build a project. They can also help reduce the size and weight of your project. Overall, learning surface mount skills can be quite rewarding and I hope that those of you making the attempt will enjoy the process and perhaps learn a few new techniques. Once you start getting used to surface mount parts, they are not that hard to work with.

**CONTACT THE AUTHOR**

Robert L. Doerr can be reached at rdoerr@bizserve.com

**WEBSITE REFERENCES**

- Robot Workshop (author’s website). [www.robotworkshop.com](http://www.robotworkshop.com)
- Another surface mount tutorial. [www.kevinro.com/newdocs/learningcenter/surface.htm](http://www.kevinro.com/newdocs/learningcenter/surface.htm)
- Intel information about surface mount technology. [www.intel.com/design/packtech/packbook.htm](http://www.intel.com/design/packtech/packbook.htm)
- Surface mount tutorial. [www.tutorialaweb.com/smt/smt.htm](http://www.tutorialaweb.com/smt/smt.htm)

---

IDEAL KITS FOR
ELECTRONIC ENTHUSIASTS

RFID SECURITY
MODULE RECEIVER KIT
KC-5393 $54.50 plus postage & packing
Radio Frequency Identity (RFID) is a contact-less method of controlling an event such as a door strike or alarm etc. An "RFID Tag" transmits a unique code when energized by the receiver's magnetic field. As long as a pre-programmed tag is recognised by the receiver, access is granted. This module provides normally open and normally closed relay contacts for flexibility, it will operate with all EM-4001 compliant RFID tags.
- Kit supplied with PCB, tag, and all electronic components.

ROLLING CODE INFRARED KEYLESS ENTRY SYSTEM KIT
KC-5458 $29.00 plus postage & packing
This excellent keyless entry system features two independent door strike outputs and will recognize up to 16 separate key fobs. The system keeps the coded key fobs synchronized to the receiver and compensates for random button presses while the fobs are out of range.
- Supplied with solder masked and silk screen printed PCB, two programmed micros, battery and all electronic components. The receiver requires a 12VDC 1.5A power supply. Five-line LCD ordering is required for fobs.

INFRARED FLOODLIGHT KIT
KG-9098 $15.95 plus postage & packing
Let your CCD camera see in the dark! This infrared light is powered from any 12-14VDC source and uses 32 x infrared LEDs to illuminate an area of up to 5 metres (will vary with light conditions). The PCB measures 56 x 75mm and draws a current of about 300mA.
- Kit is supplied with silkscreened/ gold plated solder masked PCB, 32 x infrared LEDs and all electronic components.
- Use our plugpack supply, MP3008.
- Not suitable for colour CCD cameras.

INFRARED DETECTOR KIT
KG-9086 $4.60 plus postage & packing
This kit will switch on a LED when it detects infrared light from sources such as IR remote controls. Connect it to the Relay Card kit KG-9142 to make an infrared remote controlled relay. Project requires KG102.
- PCB dimensions: 55 x 15mm
- Kit supplied with Kwik Kit PCB, infrared receiver and all electronic components.
- Can be battery powered.

SCEECHEER CAR ALARM MKII
KA-1813 $18.95 plus postage & packing
The Screcher MK II is very effective and produces an ear-piercing scream that will scare the pants off any would be thief. It is easy to construct and features entry delay with a soft warning tone, exit delay and high intensity deterrent LED.
- Kit includes PCB, alarm, all electronic components and two adhesive warning stickers for no extra cost.

ENGINE IMMOBILISER KIT MKII
KC-5256 $18.95 plus postage & packing
This immobiliser repeatedly starts the engine when a thief tries to start your car. The circuit allows the engine to start, but will stall the engine after about two seconds, giving the thief the impression that there is an intermittent problem.
- Kit supplied with PCB, plastic enclosure and all specified electronic components.

POST & PACKING CHARGES
Order Value Cost
$25 = $49.99 $7.50
$50 = $99.99 $20
$100 = $199.99 $40
$250 = $499.99 $80
$500+ $75
Max weight 12lbs (kg).
Heavy parcels P&H.
Minimum order $25.
Note: Products are despatched from Australia, so local customs duty & taxes may apply.
Prices valid 31/01/09

HOW TO ORDER
- ORDER ON-LINE: www.jaycar.com
- PHONE: 1-800-784-0263
- FAX: +61 2 8322 3118
- EMAIL: losch@jaycar.com
- POST: P.O. Box 117, Rydalmere NSW 2116 Australia
- ALL PRICING IN US DOLLARS
- MINIMUM ORDER $50
*Australian Eastern Standard Time (Monday - Friday)
9:00 to 17:30 GMT + 10 hours only
Expect 10-14 days for air parcel delivery

FREE CATALOG
Checkout Jaycar's extensive range
We have kits & electronic projects for use in:
- Audio & Video
- Car & Automotive
- Computer
- Lighting
- Power
- Test & Meters
- Learning & Educational
- General Electronics Projects
- Gifts, Gadgets & Just for fun!
For your FREE catalog logon to www.jaycar.com/catalog or check out the range at www.jaycar.com

QUALITY Parts
FAST Shipping
DISCOUNT Pricing

CALL, WRITE, FAX or E-MAIL
for a FREE 96 page catalog.
Outside the U.S.A. send $3.00 postage.

2-CONDUCTOR WEATHER-RESISTANT CONNECTORS
6" cord. Mating 2-conductor weather-resistant connectors on each end. CAT# CON-322 $1.25 each

6-18 VDC MOTOR
Mabuchi # RK370CA-1B290. 24 4mm diameter x 30.8mm long. 2 threaded mounting holes (M3.0 x 0.5) on 17.0mm centers. 2mm diameter shaft x 10.5mm long. No load rating: 8500RPM @ 13V, 0.065A. Solder terminals prepped with 3" leads. CAT# DCM-339 $2.25 each

PAINFULLY LOUD MINI-SIREN
One of the loudest, most annoying piezoelectric sirens we have ever heard. High-pitched 100 db warble tone strongly encourages intruders to leave. Designed for 12Vdc, 200mA operation. Operates 6-12Vdc. Compact - only 2.27" long x 1.69" x 1.53". Metal mounting bracket. 6" wire leads. CAT# ES-11 $4.20 each

40MM 12VDC FAN WITH COPPER HEATSINK
CoolerMaster # EEP-M41CS. 12/24v fan and copper heatsink for MiniATX, EmbATX board. CPU type: Intel Pentium M/ Celeron M. 478 pin socket (Micro-FCPGA)

Fan Specs:
12Vdc, 0.10A.
40 x 40 x 10mm
Speed: 5.500 RPM
Airflow: 5.3 CFM
Dual Ball Bearing
Noise Level: <25 dBA
Connector: 3 Pins

Heat Sink:
Copper Base + Copper Stacked Fin
Overall Dimensions: 50 x 50 x 40 mm
CAT# CF-350 $9.50 each

SAW III DIGITAL VOICE RECORDER
30 second voice recorder. 5.50" high. Operates on 2 AAA batteries (included).
CAT# SAW-3 $2.00 each

LITHIUM BATTERY CR123A
High-energy ReLiant Lithium Battery. 3V. 1300 mAh. Long-life power source used in high-performance flashlights, cameras and digital electronics. 1.34" long x 0.65" dia. Replaces: Energizer EL123AP32. Duracell DL202, Panasonic CR123A, Kodak K123A, GE Sanyo CR123A. Best if used by 2015.
CAT# LBAT-123 $1.50 each

TEST LEADS, BANANA PLUG TO MINI-GRABBERS
Red and black lead set with banana plug on one end and mini-grabbers on other end. 36" long. Mini-hooks provide a firm, insulated connection to small terminals and wires in high-density areas. CAT# TL-5 $3.75 each

20 GAUGE STRANDED WIRE, 100' SPOOL
White 20 AWG stranded (10/30) hook-up wire. UL Style 1015. CSA type TEW. 105C degree. 600V. 175°C. PVC insulation.
CAT# WT-100 $6.00 each

Shop ON-LINE www.allelectronics.com
ORDER TOLL FREE 1-800-826-5432

MAIL ORDERS TO:
ALL ELECTRONICS CORP
14928 OXNARD ST., VAN NUYS, CA 91411-2610

FAX (818) 781-2653 • INFO (818) 904-0524
E-MAIL allcorp@allcorp.com

NO MINIMUM ORDER • All Orders Can Be Chari to Visa, MasterCard, American Express or Discover • Checks and Money Orders Accepted by Mail • Orders Delivered in the State of California must include California State Sales Tax • NO C.O.D. • Shipping and Handling $7.00 for the 48 Continental United States - ALL OTHERS including Alaska, Hawaii, P.R. and Canada Must Pay Full Shipping • Quantities Limited • Prices Subject to change without notice.

MANUFACTURERS - We Purchase EXCESS INVENTORIES... Call, Write, E-MAIL or Fax YOUR LIST.

January 2009 NUTSVOLTS 67
I²C Communication

It's a new year and, unfortunately, economic times are looking quite dismal.

Our industry is also going through a recession, with people losing their jobs as companies downsize. Acquiring skills outside your normal job requirements provides a great advantage to you. If you have experience designing electronics, you should have different options even in the worst economic conditions, because electronics control most of the things we depend on everyday. Luckily, you don't have to spend a fortune on a college education to gain electronic design experience. Just reading through and experimenting with the projects in Nuts & Volts may help set you on a new path toward success.

This time around, I want to do my part in educating those getting started with Microchip PIC microcontrollers (MCUs) by demonstrating how to use an I²C (pronounced 'squared-C') Electrically Erasable Programmable Read Only Memory (EEPROM) chip. This may be helpful for other purposes, because Microchip offers many I²C-based products.

WHAT IS I²C?

I²C is a two-wire communication standard that only uses two input/output (I/O) pins on an MCU to control multiple devices. Each device on the bus connects to those two lines via open-drain pins, meaning they can pull the bus line low but require an external voltage source to pull it high. The key thing to understand about I²C is that each device on the bus has a unique control byte because all devices receive messages from the master device. However, they cannot respond or react to the message unless the control byte sent by the master matches their own. Figure 1 shows an I²C bus setup. The I²C bus has a clock line and a data line, and sends out a series of byte values serially by putting a data bit on the data line and then loading it into the slaves via the clock line. Each slave device will follow a standard type of communication, but also requires a unique set of byte-sized information sent from the master. To understand what the slave requires, review the component specifications which detail any specific communications. The main thing to understand is that I²C is a master/slave relationship, and typically, an MCU will handle the master role.

MASTER AND SLAVE

The I²C protocol defines what a master is and what a slave is. The master has control of the bus. It generates the clock signal and sends the start and stop signals that frame the complete I²C message. The slave must listen to every message sent, and respond if its control byte was received. The slave then responds to the master with data. There are systems with multiple masters but, for this article, I will only focus on a single master setup.

As mentioned, the I²C bus has two lines: a data line and a clock line. If you are familiar with how a shift register works, then this should be easy to understand. The master puts a bit on the data line by actively pulling the line low or passively letting the external resistor pull the line high. Then, the master is pulsed the clock line by pulling it low and then releasing it high again. The circuit at the slave receives that bit of data the same way a shift register clocks in a bit. This pulsing of data and clock lines continues until the first part of the I²C message is transmitted.

Figure 2 shows the format of the data line I²C message.

The signal in Figure 2 shows a write and read sequence for communicating with an external EEPROM device. Microchip offers many different types of memory chips. I will use the 24LC01B to demonstrate how to store and read back data in an external EEPROM, using I²C as the communication protocol. The control byte is a seven bit wide value, with the eighth bit being either zero for a write or one for a read operation. As you can see, the read sequence requires two control bytes, while a write only requires one. This is because a read operation requires the master to first write the memory-address byte to the EEPROM so it can locate the data and then read the data, from the EEPROM.
During the write operation, the master sends both the address and data bytes to the EEPROM, therefore only one write-control byte is sent. Notice also that Figure 2 shows an optional plural data byte(s). This is because you can send or receive a string of bytes from one read or write control byte command.

The ACK bits are the acknowledge bits. The slave pulls the data line low while the master is still pulsing the clock line. The master looks for that ACK bit before sending the next command byte. This is the only way the slave can tell the master it has received the byte of information. I²C is not as fast as other communication techniques such as the Serial Peripheral Interface (SPI) —which I hope to cover in a future article — but it's still fast. Typical clock speeds of 10 to 100 kHz are common. A high speed mode runs at 400 kHz. You have to make sure your slave parts can handle the speed of the master clock.

I²C EEPROM PROJECT

Let's create something with this knowledge. I will once again use my favorite programming tools — the Microchip PICKit 2 Starter Kit (part number DV164120). The kit includes a programmer, development board, and PIC16F690. You can get this starter kit from many sources including microchip DIRECT (www.microchipdirect.com) and the Nuts & Volts Webstore (http://store.nutsvolts.com). Nuts & Volts also offers the PICKit 2 Starter Kit with my Beginner's Guide to Embedded C Programming book, for those who want to learn the basics of C programming. For this project, I will once again use microEngineering Labs' PICBASICPRO compiler sample version to write the software. I will write the software in Microchip's MPLAB IDE. I will also use a second PICKit 2 programmer as the communication port, so I can serially send data read from the EEPROM to the PC screen. The PICKit 2's built-in UART tool handles this nicely. Figure 3 shows the schematic of this simple EEPROM setup.

HARDWARE

The schematic is quite simple. The PICBASIC PRO manual shows a very similar schematic. I added a few parts, including the 100 ohm resistors between the PIC16F690 and the 24LC01B. This allows the in Circuit Serial Programming (ICSP) technology used by the PICKit 2 to operate without any interference from the rest of the circuit. Pin C0 is brought out to a connection point that will hook up to the second PICKit 2, which is used for the serial communication. The pinout for the PICKit 2 UART connections is shown in the UART screen in Figure 4. The 24LC01B has the A0, A1, and A2 pins connected to ground, along with the WP pin. This sets the adjustable part of the control byte to 000, which I will explain a little more about in the software section of this column. The WP bit is always in write-enable mode. If it was pulled to VDD, then it would be protected from being overwritten; the part can always be read though, even if the WP bit is high.

LISTING 1

* Write to the first 16 locations of an external serial EEPROM
* Read first 16 locations back and send to serial out repeatedly

```plaintext
TWIE = 0
PORTC = 0
TWEN = 0
CHGCOND = 0
MD1 = 0
MD0 = 0
MD2 = 0
MD3 = 0
MD4 = 0
MD5 = 0
MD6 = 0
MD7 = 0
SDA = 0
SCL = 0
```

Define serial output pin
Set serial mode
12C data pin
12C clock pin
Temporary byte
Temporary byte

```
loop:
    For B0 = 0 To 15
    I2CWRITE DPIN, CPIN, SA0, B0, [B1]
    Pause 10
    Next B0
```

Loop 16 times
Write each location's address to itself
Delay 10ms after each write

```
For B0 = 0 To 15
    I2CREAD DPIN, CPIN, SA0, B0, [B1]
    Serout SO, T2400, [#results:*]
    Serout SO, T2400, [#3S, $0D, $0A]
Next B0
```

Loop 16 times
Print each location
Print "results:" label
Send ASCII value and then carriage return and line feed

```
Serout SO, T2400, [11, 10]
Goto loop
```

Print linefeed for spacing

January 2009 NUTSVOLTS 69

http://nutsvolts.textuality.com/nutsvolts/200901/templates/pageviewer_print?pg=1&p...
THE SOFTWARE AND HOW IT WORKS

The software is fairly simple and is shown in Listing 1. The program initially sets up the control registers in the PIC16F690. PortC is made into an output and then initially set to all zeros. The PIC16F690 defaults to analog pin mode, so the pins have to be made digital by clearing the ANSEL register. The internal comparators are turned off by clearing their respective registers.

TRISC = 0
PORTC = 0
ANSEL = 0
CM1CON0 = 0
CM2CON0 = 0

A few constants and variables are created. The serial communication pin is PORTC.0, which is also pin 8 in the PICBASIC PRO PIC16F690 setup. The PICKit 2 acts as a true UART hardware interface, even though it sends the data back to the USB port. Therefore, we need the True mode — or T2400 — for 2400 baud communication. The N2400 is there just to work with a direct connection to another PIC MCU instead of a PC.

S0 con 8 ' Define serial output pin
N2400 con 4 ' Set serial mode
T2400 con 0
DPIN var PORTA.2 ' I2C data pin
CPIN var PORTA.1 ' I2C clock pin
B0 var byte ' Temporary Byte
B1 var byte ' Temporary Byte

Here is another example of why I like the PICBASIC PRO compiler for the beginning programmer. The compiler has I2CWRITE and I2CREAD commands to make this very easy. The commands can use any digital port since they "bit-bang" the I2C signal. The command handles all the timing and reading of the ACK bits. It also handles the read or write bit in the control byte. The only things you need to supply in the command line are the connection designation for the data and clock pins, plus the control byte. The control byte for the 24LC01B is in the datasheet and is (binary) %1010xxxx, where xxx is the setting for the A0, A1, and A2 pins of the EEPROM. You can have more than one EEPROM connected to the I2C bus, so setting those pins to something other than ground allows you to create a new control byte for that particular part. If we had a second EEPROM hooked up to the bus, its control byte could be %1010001. This would require the A0 pin be pulled to VDD instead of ground.

For B0 = 0 To 15
I2CWRITE DPIN,CPIN,5#0B0,[B0] ' Loop 16 times
' Write each
location's
address to
'self
Pause 10
' Delay 10ms
' after each
' write
Next B0

The program just "For-Loops" through setting each memory address to the same value as the address (i.e., address 0 has data value 0; address 1 has data value 1, etc.). This creates the initial data in the EEPROM. Next, another For-Loop reads each address and sends the data read to the PICKit 2 UART screen using the SEROUT command. Figure 4 shows the UART screen and the 16 address locations read back from the EEPROM. The "#" symbol in front of the B1 variable converts the decimal value into an ASCII value for the UART. In other words, a value of 15 is sent as two bytes — an ASCII "1" and ASCII "5" so it is displayed as 15, rather than the ASCII character associated with decimal 15.

Figure 5 shows the MPLAB IDE window complete with the PICKit 2 running. You need MPLAB IDE v. 8.14 or later to use two PICKit 2 programmers at the same time; otherwise, the software won't know which one goes to the MPLAB IDE and which one goes to the UART. The MPLAB IDE software or the PICKit 2 GUI will first ask which PICKit 2 you want to connect to. I've covered how you can store a unique name inside the PICKit 2's memory to make it easier to select the correct programmer in past articles. Check the PICKit 2 manual for instructions on how to "set the unit ID." The UART software is under the tool menu of the PICKit 2 standalone GUI software that comes with the programmer. Both the MPLAB IDE and PICKit 2 software can be downloaded from Microchip's website.
if you cannot find the disk or just want to check out what I refer to before buying a couple of these programmers. The final setup is shown in Figure 6. Notice how I added a small breadboard area to the low pin count demo board. This addition makes it easier to add circuitry to a PIC16F690 project when using this development board. Another option would be to use one of the many plug-in modules available for this type of development. Figure 7 shows the breadboard EEPROM module I designed many years ago, which is now sold exclusively through Beginner Electronics (www.beginnerelectronics.com). The socketed chip allows you to change the chip to different sized EEPROMs. The module also has pull-up resistors installed on the data and clock lines.

Another option is the multi-EEPROM board from ETI, shown in Figure 8. This module has four EEPROM devices on the board, along with optional pull-up resistors via jumpers. Finally, the developer's dream board is the PICKit Serial I/C Demo Board from Microchip, shown in Figure 9 (part number PKSERIAL-I2C1). This board has an I/C EEPROM, I/C temperature sensor, I/C Analog-to-Digital Converter (ADC), I/C Digital-to-Analog Converter (DAC), and an eight-bit I/C I/O expander — all connected to a common I/C bus. Any of these modules make I/C development a lot easier. I'm sure I'll use them in future projects.

Another Microchip tool you might want to check out for I/C development is the PICKit Serial Analyzer (part number DV164122), shown in Figure 10. This tool uses the same packaging as the PICKit 2 but is designed strictly for serial communication including I/C, SPI, Microwire, and USART. The kit comes with its own GUI for the PC and includes a 28-pin demo board with a PIC16F886 MCU that can also be plugged into a PICKit 2 for programming. I have recommended this development board in previous columns.

You can plug the PICKit serial analyzer into the Microchip I/C demo board (see Figure 9) to use it as a master. You can also send data to the PICKit serial analyzer from your I/C master circuit so that you can verify your code is communicating properly to a slave. More information is available at www.microchip.com/pickitserial. The kit shown in Figure 10 sells for $49.95.

**CONCLUSION**

Once again, PICBASIC PRO makes a software task very easy to complete. The PICKit 2 comes through once again to handle the programming and monitoring tasks. Many PIC MCUs have a hardware I/C peripheral on-chip, so you can run the I/C in the background. This becomes very handy when running multiple functions from a single master MCU. For most home projects, though, the bit-banging method — also often officially termed firmware — is sufficient. The 24LC01B that I used here is only a one Kbit EEPROM, with 128 bytes of storage space (1024/8 bits = 128 bytes). You can get larger memory parts up to one Megabit (Mb), but some of the larger memory parts require a two-byte address sent from the master to the EEPROM. PICBASIC PRO handles this for you when you specify the address in the command line. A byte address will be sent if the address is a byte variable. A two-byte value will be sent if the address is a word variable. Read over the PICBASIC PRO manual for additional details.

Hopefully, I taught you something new to expand your electronics knowledge. Try out some of the other I/C components on your own setup. By doing that, you can gain a lot of experience that might just prepare you for an opportunity that keeps you employed or gives you some extra income in the future. There is so much to know in the world of electronics, and having hands-on experience is priceless.

Email your thoughts to me at chuck@elproducts.com. If the spam filter doesn't catch it, I'll read it. Please put “N&V” in the subject line to help me filter the incoming email.

Check out my books too, at www.elproducts.com. I hope to have more released soon. NV
BY FRED EADY

CHATTING UP A THUMBDRIVE

I NEVER SEEM TO HAVE A USB FLASH DRIVE when I need one. So, when I saw this great deal on some 8GB DRIVEs, I picked up a couple for around the shop. I figured ITS SIZE was enough for just about any sneaker-net data transfer job I’d have to do. Plus, the price was really good. Just last week I came across another Internet gotta-have-it USB flash drive deal. This time, I bit on some bargain 16GB flash drives.

As it turns out, I have plenty of USB Flash drives to plug into this month’s project. I’ve mounted a VIDP2 and a PIC18LF2620 on a perfboard and plumbed it in some power. We’re going to pick up where we left off last time and put some C code together to access the innards of that herd of USB Flash drives grazing on the bench in my office.

THE HOOKUP

Thanks to the PIC18LF2620’s internal 8 MHz oscillator block and its ability to operate at 3.3 volts, the interface between the PIC18LF2620 and the VIDP2 is very simple and clean. The VIDP2 is externally powered with five volts. The VIDP2’s on-board 3.3 volt regulator supplies the internal 3.3 volts that the VIDP2’s VNC11 IC requires. The Microchip MCP1700 has enough headroom to supply up to 200 mA of 3.3 volt power to devices external to the VIDP2. Take a look at Schematic 1 and you’ll see that the PIC and its supporting circuitry are powered by this.

Note that the PIC18LF2620’s EUSART and the VIDP2 serial interface are directly coupled. We eliminate the need to translate the serial port logic levels by powering the PIC with the same 3.3 volt power rail required by the VIDP2.

With the exception of the addition of some debugging LEDs attached to I/O pins RC3, RC4, and RC5, you have an abundance of PIC18LF2620 I/O to use at your leisure. The VIDP2 multiplexes its communications lines with its I/O lines, depending on the communications configuration you select. We are configured for UART mode. The VIDP2 I/O lines AD0 and AD1 flex from bidirectional I/O to serial transmit and receive, respectively. Just in case we’ll need to implement handshaking, I’ve pulled out the VIDP2’s RTS (Request to Send) and CTS (Clear to Send) modem signals and terminated them at the PIC18LF2620’s RC0 and RC1 I/O pins.

For those of you that are not communications experts, the DTE (Data Terminal Equipment) device raises the RTS signal when it wants to transmit. The DCE (Data Communications Equipment) attached to the DTE sees the RTS signal and raises its CTS signal in response if all is clear for transmission. The DTE and DCE terms refer to a host computer which is the DTE device, and a modem which is
the DCE device. In our project, the DTE device is VDIP2 while the PIC18LF2620 is acting as the DCE device.

The VDIP2 interface hardware you see in Photo 1 was assembled in less than an hour. The VDIP2 is mounted to the perf board using a high quality 40 pin DIP socket. The same holds true for the PIC18LF2620 except it is riding on a 28 pin socket. I used wirewrap wire to make all of the soldered point-to-point electrical connections.

Our VDIP2 hardware is simple due to the fact that firmware rules the roost in the project. There's plenty of good VDAP firmware residing within the VDIP2's VNC1L. So, let's return the favor and put some worthy VDIP2 interface firmware into the PIC18LF2620.

CODING AN API FOR THE VDIP2

There is absolutely no doubt in my mind that you'll use your VDIP2/PIC18LF2620 combination in a project that is unique to you. With that thought in mind, I've decided to put together a set of firmware routines you can pick and choose from when applying the VDIP2 in your own projects. The good news is that you'll find the VDIP2 API code to be just as minimal and basic as the interface hardware. The entire VDIP2 API is available as a download package from the Nuts & Volts FTP website (www.nutsvolts.com).

Let's start with the API base procedures and work our way up to the API main() function. The very first function we will call from the main() function is the init() function:

```c
// * INITIALIZE ROUTINE - Init clock and I/O
void init() {
    OSCON = 0x70; //set for 8 MHz
    FPLLEN = 0;   //no PLL
    TRISA = 0x0b111111; //pin 2
    TRISB = 0x0b111111; //pin 4
    TRISC = 0x0b10000001; //RTS is input
}
```

The very first code that we will execute overrides the default power-up oscillator frequency of 1 MHz and sets the PIC18LF2620 up with an 8 MHz processor clock. We could enable the processor clock 4X PLL to run at

![Schematic 1](http://nutsvolts.texterity.com/nutsvolts/200901/templates/pageviewer_print?pg=1&p... 12/02/2009)

The majority of the physical electronic parts used in this design are mounted on the VDIP2 printed circuit board. Thanks to the VNC1L firmware, all of the hard work is shifted away from the hardware and is done with firmware.

January 2009 NUTSVOLTS 73
32 MHz if our application demanded it. The 8 MHz clock frequency will provide plenty of computational throughput for our VDIP2 application. Recall that the VDIP2 will issue an RTS signal to the PIC. So, RC0, which is the PIC's RTS input, is set up as an input pin. The other PORTC input is the RX (receive) line for the EUSART. Regardless of our manual manipulation of the direction of the TX (RC6) and RX (RC7) pins, the PIC18LF2620's EUSART automatically sets the RC6 and RC7 pins for input or output as required. The remaining PORTC pins are defined as follows:

```c
/* VDIP2 I/O DEFINITIONS */
#define pRTE LATC0
#define pCTS LATC1
#define pRTS LATC2

/* LED DEFINITIONS */
#define LED3 LATC3
#define LED2 LATC4
#define LED1 LATC5
```

The PORTC VDIP2 pin definitions include a p prefix for the modem control signals and the VDIP2 reset pin. LEDs occupy the remainder of the PORTC pins. I've assigned LED1 as the heartbeat LED. The heartbeat LED will flash at a rate of 1 Hz, indicating that the VDIP2 API code is executing. The VDIP2 reset pin is a convenience connection that you'll find very useful. Without the VDIP2 reset pin connection, you would have to power-down and power-up the VDIP2 every time you made a change to the PIC firmware. Right now, we have no use for the PIC18LF2620's analog-to-digital (A-to-D) converter module or the comparators. So, let's turn all of that stuff off:

```c
/* CONFIGURE A2D AND COMPARATORS */
ADCON1 = 0x00001111; // All digital I/O
ADON = 0; // ADC off
CMCON = 0x07; // comparators off
```

Since the EUSART is our desired means of communication with the VDIP2, we need to get the PIC18LF2620 EUSART online as quickly as possible. Once the PIC18LF2620's I/O ports are configured and the (A-to-D) converter and comparators are disabled, the code within the `init()` function fires up the PIC's EUSART with the following function call:

```c
/* INITIALIZE EUSART */
init_EUSART();
```

The code I use to service the PIC18LF2620's EUSART is battle tested. All we have to do to put the interrupt-driven EUSART code to work for us is to specify a baud rate in the EUSART initialization code:

```c
/* Init EUSART Function */
```

```c
void init_EUSART(void) {
    SPBEG = 51; // BM12 = 51 FOR 9600BAUD
    TRISC7 = 1; // receive pin
    TRISC6 = 0; // transmit pin
    TXSTA = 0x04; // highspeed baud, BRGH = 1
    RCSTA = 0x80; // enable serial port

    EUSART_RxTail = 0x00; // flush rx buffer
    EUSART_RxHead = 0x00;
    EUSART_TxTail = 0x00; // flush tx buffer
    EUSART_TxHead = 0x00;
    RCIP = 1; // rx interrupt = high pri
    TXIP = 1; // tx interrupt = high pri
    RCIE = 1; // enable rx interrupt
    PEIE = 1; // enable all unmasked
    GIE = 1; // enable all unmasked ints
    CREN = 1; // enable EUSART1 receiver
    TXIE = 0; // disable EUSART1 tx int
    TXEN = 1; // transmitter enabled
}
```

Yep, I put my fingers on the TRIS (data direction) bits of the EUSART's TX and RX pin yet again in the `init_EUSART()` function. In reality, the TRISC7 and TRISC6 statements in the `init_EUSART()` function could probably be eliminated. However, when something works as well as this EUSART code has just as it is, leave it alone. In addition to enabling the EUSART interrupt mechanism, the code within the `init_EUSART()` function initializes the transmit and receive buffer pointers which invalidates any spurious data that resides in the buffers at power-up. The EUSART transmit and receive buffer area is defined in the code that follows:

```c
/* EUSART BUFFER DEFINITIONS */
// 1, 2, 4, 8, 16, 32, 64, 128 or 256 bytes
#define EUSART_RX_BUFFER_SIZE 256
#define EUSART_RX_BUFFER_MASK (EUSART_RX_BUFFER_SIZE - 1)
#define EUSART_TX_BUFFER_SIZE 256
#define EUSART_TX_BUFFER_MASK (EUSART_TX_BUFFER_SIZE - 1)

unsigned char EUSART_RxBuf[EUSART_RX_BUFFER_SIZE];
unsigned char EUSART_TxBuf[EUSART_TX_BUFFER_SIZE];
unsigned char EUSART_TxHead, EUSART_TxTail,
EUSART_RxHead, EUSART_RxTail;
```

I've set the transmit and receive buffer sizes at their maximums of 256 bytes each. The transmit and receive buffer masks are used to convert the EUSART buffer spaces into a pair of circular buffers. The EUSART transmit and receive buffers are serviced by an interrupt mechanism called `EUSART_TIMER1`: 

/* INTERRUPT HANDLER ROUTINE

void interrupt EUSART_TIMER1(void) {
    char uart_data, tmphead, tmpTail;

    The temporary transmit and receive buffer pointer bytes in the interrupt handler char declaration. The
    uart_data byte is used to hold the incoming bytes that are removed from the EUSART's hardware buffer. The
    TIMER1 portion of the PIC18LF2620 interrupt handler is represented by code that "ticks" every millisecond:

    if((TMR1IF & TMR1IE)) {
        TMR1IF = 0;
        TMR1H = 0xF8;
        TMR1L = 0x31;
        --tmscc1;
        if(++tmscc1 == 1000) {
            tmscc1 = 0;
            +iscc1;
            +tscc1;
            LED1 ^= 1;
        }
    }

    Every time that TIMER1 overflows, it is reloaded with an integer that causes the timer to overflow again in one
    millisecond. The milliseconds are collected into seconds and are used to provide timing for the millisecond
    and second timer functions. As you can see in the TIMER1 interrupt handler code, LED1 is toggled every 1,000 ms
    or every second.

    Every time the EUSART signals that a character is captured in its hardware receive buffer, the interrupt
    handler code fetches the received byte from the EUSART's hardware buffer and stores it in the next available byte slot
    of the receive buffer. The idea behind the circular buffer scheme is to never let the head pointer value of the buffer
    catch up to the tail pointer value of the buffer. I've never had to execute the error statements in this receive
    interrupt handler code:

    if(RCIF) {
        uart_data = RCREG; //read received data
        // calculate new buffer index
        tmphead = ( EUSART_RxHead + 1 ) &
            EUSART_RX_BUFFER_MASK;
        EUSART_RxHead = tmphead; // store new index
        if (tmphead == EUSART_RxTail) {
            //ERROR! Receive buffer overflow
            status.buf_ovrf = 1;
        }
        // store received uart_data in buffer
        EUSART_RxBuf[tmphead] = uart_data;
    }

The transmit interrupt handler code is only used by the sendchar() function. When the sendchar() function is
called, the outgoing byte is placed in the PIC18LF2620's transmit buffer and the transmit interrupt enable bit is
activated. The interrupt handler code keys on the transmit interrupt bit, retrieves the byte from the transmit buffer,
and stuffs it into the EUSART's hardware transmit buffer. The EUSART -- seeing a byte in its TXREG register
-- sends that byte along its way. Like the EUSART receive interrupt handler code, the transmit interrupt handler
code declares the associated buffer empty when the buffer head pointer is equal to the tail buffer pointer. If
more than one character is waiting in the PIC's transmit buffer, the transmit interrupt handler code will attempt
to transmit until the transmit buffer is found to be empty. Here's what the transmit interrupt handler code looks like:

    if(TXIF) {
        //check if all uart_data is transmitted
        if ( EUSART_TxHead == EUSART_TxTail ) {
            //calculate new buffer index
            tmptail = ( EUSART_TxTail + 1 ) &
                EUSART_TX_BUFFER_MASK;
            EUSART_TxTail = tmptail; //save new idx
            TXREG = EUSART_TxBuf[tmptail]; //send it
        }
        else {
            TXIE = 0; // disable TX interrupt
        }
    }

The CharInQueue(), recvchar(), and sendchar() functions interact with the interrupt handler routines we've just discussed. We examined the inner workings of the CharInQueue(), recvchar(), and sendchar() functions in the previous installment of Design Cycle.

After powering up the PIC18LF2620's EUSART, we turn our attention to TIMER1:

    //* CONFIGURE AND START TIMER1,
    //* SET TO OVERFLOW EVERY 1ms
    //F831 = 8MHz
    //EDC1 = 32MHz
    TIMER1OFF;
    TICON = 0b00000000;
    TMR1H = 0xF8;
    TMR1L = 0x31;
    TIMER1ON;

Basically, we let TIMER1 do its thing with the processor clock unhindered by prescalers. Just in case
you want to put the pedal to the silicon, I've included the TIMER1 32 MHz clock value in a comment of the TIMER1
initialization code.

At this point, we can open the gate and let the horses run:

January 2009 NUTSVOLTS 75
SCREENSHOT 1. The very first byte of the receive buffer is not used initially due to the way that its mask manipulates the receive buffer pointer. However, you can easily pick out the echo responses in this capture.

SCREENSHOT 2. At this point, we’ve successfully initialized the VDIP2 and synchronized to it. The message you see in the capture says it all.

```c
/**
 * CONFIGURE EXTERNAL INTERRUPTS
 * enable_TMR1Int;
 * enable_GLOBALInt;
 *
 * The PIC18LF2620 interrupt engine is started and the interrupt handler code begins to process the incoming interrupts. The final initialization acts we perform tell the VDIP2 that it can transmit to the PIC18LF2620 EUSART, to make sure that the VDIP2 is not in a forced reset, extinguish the remaining LEDs (LED1 is flashing), and set up the operational mode:
 */

/** INITIALIZE I/O
 * pCTS = 0;
 * pRST = 1;
 * LED2 = OFF;
 * LED3 = OFF;
 * current_mode = waiting_for_work;
 * sdelay1(1);
 */

The operational modes are enumerated as follows:

```enum {
waiting_for_work,    //0
disk_ready,         //1
file_create,        //2
file_write,         //3
file_end,           //4
waiting_for_remove  //5
};```

Using `enum` is just a fancy way of assigning each operational mode to a number. Using names associated with numbers makes the code a bit easier to read and understand.

Now that we've initialized the PIC18LF2620, we're ready to converse with the VDIP2. The first order of business is to bring our VDIP2 VDAP firmware up to date. There are a couple of ways to perform the VNC1L update. The bootloader method requires some hardware and a bit of coding to get the latest level of VDAP disk and peripheral firmware into the VNC1L IC.

The second firmware upgrade choice involves initializing the VDIP2 and plugging in a Flash drive that contains the updated firmware file in its root directory. Thus far, we've almost written just enough code to initialize the VDIP2 using the PIC18LF2620.

So, we'll update our VDIP2 using what the Vinculum engineers call the Reflash method, which utilizes a special file loaded in the root directory of a Flash drive. However, before we can reflash our VNC1L, we still have a couple of API routines to construct.

First, let's build a VDIP2 reset function:

```c
/** VDIP2 RESET
 * void vreset(void) {
 *     pRST = 0;
 *     mdelay1(100);
 *     pRST = 1;
 * }
 *
 * The logic associated with the VDIP2 reset pin is active low. So, to reset the VNC1L that rides on the VDIP2 printed circuit board, all we need to do is pull the VDIP2 reset pin low for a period of time and then return it to a logical high state. To pull off the VDIP2 reset, we need a supporting timing function:
 */
```

```c
#define mdelay1(msecdelay) TIMER1OFF; \  
     TMR1IF = 0; \  
     imsec1 = 0; \  
     tmsec1 = 0; \  
     TIMER1ON; \  
     while(tmsec1 < msecdelay);
```

The `mdelay1()` delay routine is really not a function. It is a C macro. As you can see the inner workings of the `mdelay1()` macro above, TIMER1 is disabled, the TIMER1 interrupt handler values are cleared, the timer is restarted, and time is marked in place until the number of milliseconds in the macro argument have passed.

The next bit of code we need to assemble to initialize the VDIP2 is a synchronization function. Let's examine the VDIP2 synchronization I've written snippet by snippet:

```c
/** SYNC WITH VINCULUM
 */
```
char sync(void) {
  char rc;

  rc = 0;
  sync_loops = 0; //Init sync loop counter
  status.E = 0; //clear E received status
  status.e = 0; //clear e received status
  cleaner(); //clear receive buffer

  The VDIP2 sync() function sends an $E$ and an $e$ in succession to the VDIP2 with each letter followed by a carriage return (0x0D). The VDIP2 echoes the received character indicating that is in "in sync" with the PIC18LF2620 and its firmware. The results of each echo operation are recorded in a status bit which is defined in a structure of bits:

  typedef struct{
    char E;1;
    char e;1;
    char buf_ovrf;1;
  }SSTATS;

  SSTATS status;

  Note that the EUSART receive buffer overflow bit is also defined in the status bit structure. As we build upon the API, bits will be added to the status structure and the enum list as required.

  You’re probably wondering what the cleaner() function does for us. The cleaner() function is not necessary to the operation of the VDIP2 system. This function is actually just a convenience function that is used to clear the PIC’s EUSART transmit and receive buffers. The “cleaner” also resets the buffer head and tail pointers to indicate that the buffers are empty. I used this function quite a bit while debugging the API code:

  /// CLEAR THE RECEIVE BUFFER
  void cleaner(void) {
    unsigned int x;

    for(x=0;x<EUSART_RX_BUFFER_SIZE-1;++x)
      EUSART_RxBuf[x] = 0;

    for(x=0;x<EUSART_TX_BUFFER_SIZE-1;++x)
      EUSART_TxBuf[x] = 0;

    EUSART_RxTail = 0x00; //flush rx buffer
    EUSART_RxHead = 0x00;
    EUSART_TxTail = 0x00; //flush tx buffer
    EUSART_TxHead = 0x00;
  }

  Syncing up with the VDIP2 is a straightforward process. The PIC18LF2620 is instructed to send an $E$ followed by a carriage return and wait for a response.

  The VDIP2 should respond with the sequence of characters it received, which are an $E$ followed by a carriage return. If the correct characters are received by the PIC, the status.E bit is set.

  The process is repeated until the correct character sequence is received or the attempt counter overflows. Once the initial $E$ sequence is successful, an identical transmission/reception process is executed using an $e$. If the $e$ echo sequence is successful, the status.e bit is set. The $E$ echo and sync code snippet looks like this:

  do{
    cmd_echo('E'); //send E
    sync_loops++; //inc sync loop counter
    bytein_ptr = 0; //init rx char pointer

    //wait for character
    while((!CharInQueue()));
    //get 2 bytes
    do{
      bytein[bytein_ptr++] = recvchar();
    }while(CharInQueue());

    //check for echoed E and carriage return
    if(bytein_ptr) {
      if ((bytein[0] == 'E') &&
          (bytein[1] == 0x0D)) {
        status.E = 1;
      }
    else {
      delay1(100);
    }
  }while((status.E == 0) &&
            (sync_loops < 0xFF));

  If the status.E bit gets set, all is good and permission to proceed with the execution of the $e$ echo and sync code is given:

  if (status.E) {
    do{
      cmd_echo('e'); //send e
      sync_loops++; //inc sync counter
      bytein_ptr = 0; //init rx char ptr

      //wait for character
      while((!CharInQueue()));

      //get 2 bytes
      do{
        bytein[bytein_ptr++] = recvchar();
      }while(CharInQueue());

      //check for echoed e and CR
      if(bytein_ptr) {
        if ((bytein[0] == 'e') &&
            (bytein[1] == 0x0D)) {
          status.e = 1;
        }
    }
    }
  }
SCREENSHOT 3. The VDIP2 is now responding to commands and issuing status information. We have arrived!

```c
(bytein[1] == Ox0D)) {
    status.e = 1;
} else {
    delay(100);
}
} while((status.e == 0) && (sync_loops < OxFF));

if(status.E && status.e)
    rc = 1;
return rc;
```

A return code of 1 is passed back to the caller if the echo and sync operations were successful. At this point, here's what our API main() function code looks like:

```c
/* MAIN MAIN MAIN MAIN MAIN

void main() {
    init();
    cleaner();
    vreset();
    sync();
    do{
        ++sync_loops; //loop here forever
    )while(1);
}

The HI-TECH C PRO for the PIC18 MCU family compiler placed the PIC18LF2620's EUSART transmit buffer at offset 0x0100 and the receive buffer at offset 0x0200 in the PIC18LF2620's SRAM. The results of the execution of the API main() function code we've written thus far is captured in Screenshot 1. I produced the data in Screenshot 1 and all subsequent screenshots from the MPLAB File Register views using the MPLAB IDE and a Microchip REAL ICE in debug mode.

Believe it or not, we're one character away from initializing the VDIP2. Let's see what happens when we add these two lines of code to our current API main() function just after the sync() function call:

```c
/* MAIN MAIN MAIN MAIN MAIN
void main() {
    init();
    cleaner();
    vreset();
    sync();
    sendchar(0x0D);
    sendchar('V');
    sendchar(0x0D);
    do{
        ++sync_loops;
    )while(1);
}
```

I threw in a call to the "cleaner" after the synchronization so we would get a screenshot with a fresh look at what the carriage return did. Screenshot 2 shows that the VDIP2 got a bit wordy. That's a good thing and the VDIP2 speaks the truth. A Flash drive is not mounted.

All we have to do to update the VDIP2's VNC1L with new VDAP firmware is power-down the VDIP2 and PIC18LF2620, mount a Flash drive containing the upgrade file, power-up the VDIP2 and PIC18LF2620, and run the API code we have written up to this point.

Remember those bargain Flash drives I bought? Well, they don't work with the VDIP2. After a few dozen unsuccessful attempts to upgrade the VNC1L firmware, I resorted to a more careful read of the VDIP2 User Manual. I came across a blurh that informed me that some Flash drives don't follow the rules of a hard drive and thus, won't be recognized as valid Flash drives by the VNC1L.

After reading that, I put out an APB for a different brand or model of Flash drive and my wife pulled one out of her computer bag. I plugged it in and when its activity LED stopped bouncing around, I performed a POR (Power On Reset) of the VDIP2 hardware. I decided to go for broke and executed the following code:

```c
/* MAIN MAIN MAIN MAIN MAIN
void main() {
    init();
    cleaner();
    vreset();
    sync();
    sendchar(0x0D);
    sendchar('M');
    sendchar(0x0D);
    do{
        ++sync_loops;
    )while(1);
```
THE DESIGN CYCLE

| 200 | 00 | 0D | 56 | 65 | 72 | 20 | 30 | 33 | 2E | 36 | 36 | 56 | 44 | 41 | 50 | 46 | .Ver 03 .66VDAPF |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|                |
| 210 | 20 | 4F | 6E | 2D | 4C | 69 | 6E | 65 | 3A | 0D | 45 | 4D | 0D | 65 | 0D | On-Line ;E.E.e. |
| 220 | 65 | 0D | 65 | 0D | 4E | 6F | 20 | 44 | 69 | 73 | 68 | 0D | 44 | 65 | 76 | 69 | e.e.No D isk.Devi |
| 230 | 63 | 65 | 20 | 44 | 65 | 74 | 65 | 63 | 74 | 65 | 64 | 20 | 50 | 32 | 0D | 4E | ce Detec ted P2.N |
| 240 | 6F | 20 | 55 | 70 | 67 | 62 | 61 | 64 | 65 | 0D | 44 | 3A | 5C | 0D | 00 | o upgrad e d:

Screenshot 3 tells the story. The alternate flash drive worked. The reset operation caused the version to be displayed, followed by an on-line indication. Next, you can see the sync characters being acknowledged. The carriage return that I issued after the sync() function call forced the NO Disk message.

Here's where I sucked it up and issued a real VDIP2 monitor command. The firmware version of the VNC11 is returned when the character set FWV followed by a carriage return is issued to the VDIP2's monitor. The version information begins with the characters MAIN in Screenshot 3. Note that a prompt is issued even though a Flash drive was not mounted.

ONE MORE HURDLE

Everything has worked as designed up to this point. However, we have one more test to perform before moving on to put the rest of the meat into the API. Here's the code we'll execute:

```c
/** MAIN MAIN MAIN MAIN MAIN
void main() {
  init();
  cleaner();
  vreset();
  sync();
  sendchar(0x0D);
  do{
    ++sync_loops;
  }while(1);
}
```

I'll run the code without mounting a Flash drive. Once the VDIP2 indicates that it is initialized and synced (the on-board VDIP2 LEDs stop flashing and extinguish), I'll mount a Flash drive. Follow along referencing Screenshot 4 and let's see if we're ready to start reading and writing to the Flash drive.

All of the monitor messages up through No Disk in Screenshot 4 were issued immediately after power-up with no Flash drive mounted. The monitor message Device Detected P2 says that a Flash drive was detected on USB Port 2. This is good as the Device

Detected P2 message was sent when I mounted the Flash drive. The No Upgrade monitor message is telling us that the upgrade file (FTRFB.FTD) was not found in the root directory of the newly mounted Flash drive. Most importantly, the D:/> prompt indicates that the Flash drive I mounted is read/write compatible with the VDIP2. Welcome to Flash Drive City!

DOWNLOAD YOUR COPY OF THE API CODE

You have witnessed what it takes to issue a command. Therefore, you won't have any problems deciphering the C source in the download package that makes up the VDIP2 API. I've written the VDIP2 API to include Disk Commands, Monitor Configuration Commands, and I/O Commands. I've also included an example file read/write application in the download package.

Using the VDIP2 API coupled with the VDIP2 hardware interface we've designed and built, you should have no problems including USB Flash drives in your Design Cycle.

It writes your USB Code!

NEW! HIDmaker FS for Full Speed FLASH PIC18F4550

Creates complete PC and Peripheral programs that talk to each other over USB. Ready to compile and run!

- Large data Reports
- 64,000 bytes/sec per interface
- Easily creates devices with multiple interfaces, even multiple identifiers!
- Automatically does MULTITASKING
- Makes standard or special USB HID devices

NEW! "Developers Guide for USB HID Peripherals" shows you how to make devices for special requirements.

Both PC and Peripheral programs understand your data items (even odd sized ones), and give you convenient variables to handle them.

**PIC18F Compilers:** PICBASIC Pro, MPASM, C18, Hi-Tech C.

**PIC16C Compilers:** PICBASIC Pro, MPASM, Hi-Tech C, CCS C.

**PC Compilers:** Delphi, C++ Builder, Visual Basic 6.

HIDmaker FS Combo: Only $599.95

DOWNLOAD the HIDmaker FS Test Drive today!

www.TraceSystemsInc.com

301-262-0309

January 2009 NUTSVOLTS 79

PART 6: A Menu Navigator

I know just enough French to be dangerous, so on a visit to Paris, I insisted on using the regular menu and got some very strange meals. After a few such mishaps, I discovered that they usually had an English menu hidden away for les barbères so I started using that one and still got some very strange meals. C'est la Vie. In this month's workshop, after learning more about C syntax for decision making, we are going to write a menu navigation system similar in concept to the one on your cell phone, but for the Butterfly using its LCD and joystick. Important safety tip: Make sure you understand the concepts in Workshops 1 through 5 before slamming your head into this article.

C Control Flow

Statements and Blocks

Expressions such as PORTD = -i or j = 128 become statements when they are followed by a semicolon.

PORTD = -i;
j = 128;

The semicolon terminates the statement.

Tale of a Bug

I wrote the following statement:

while(QuarterSecondCount < 17600);
QuarterSecondCount = 0;

then decided that the 17600 wait count was too long, so I changed it to 2200:

while(QuarterSecondCount < 2200) ;//17600);
QuarterSecondCount = 0;

But I wanted to remember the 17600 in case I ever needed it again, so I commented it out and added the new value. Do you see a problem here?

Well, what I meant to say was:

while(QuarterSecondCount < 2200);
QuarterSecondCount = 0;

which is two statements: the first waits while an interrupt running in the background increments QuarterSecondCount, and once that is finished the QuarterSecondCount is set to zero. What the compiler saw was:

while(QuarterSecondCount < 2200) ;
QuarterSecondCount = 0;

But the compiler doesn’t see anything following the // comment delimiter. See the problem yet?

Well, how about the equivalent statement:

while(QuarterSecondCount < 2200) ;
QuarterSecondCount = 0;

I had accidentally ‘commented out’ the terminating semicolon from the first statement. The compiler doesn’t know about the line breaks; all it sees is a single statement which says that while QuarterSecondCount is less than 2200, set QuarterSecondCount to 0. So, each time the interrupt incremented QuarterSecondCount, this statement set it back to zero. One lousy semicolon gone and everything changes!

This is the kind of bug that after spending X amount of time locating, you carefully hide it from your boss lest she think you are stupid or careless or both. Fortunately, I am my own boss, so I’ve learned to live with my stupid and careless employee. (I fired myself once, but that just didn’t work out.)

Compound statements are made by enclosing a group of statements or declarations in a block delimited by braces '{' and '}’. This causes the compiler to handle the block as a unit.

If-else and else-if

We can make decisions using the if-else statement:

80 NUTSVOLTS January 2009
if (expression1)
{
    statement1a;
    statement1b;
}
else if(expression2)
{
    statement2a;
    statement2b;
}
else
{
    statement3a;
    statement3b;
}

In this case, each expression will be evaluated sequentially looking for the first non-zero (true) expression and if they all equal 0 (false), we do the final block of statements. You can omit the final else statement if you want to do nothing if all the expressions are 0 (false). Note also that only the first true expression is used; if expression1 is true and expression2 is also true, it won’t matter because the code will exit the if-else after the first true case. We could use this construction to write a block of code for interpreting joystick input positions:

```c
if(input == KEY_UP) keyUP();
else if(input == KEY_DOWN) keyDOWN();
else if(input == KEY_LEFT) keyLEFT();
else if(input == KEY_RIGHT) keyRIGHT();
else if(input == KEY_PUSH) keyPUSH();
else keyERROR();
```

This says: if the input is equal to KEY_UP, then call the keyUP() function. If the first line is true, then the rest of the statements are skipped. If the first line isn’t true, then each line is evaluated sequentially until a true expression is found or it calls the final 'else' keyError() function.

**Switch**

The 'if else' construction limits us to expressions that are either true or false. If we want to make decisions using expressions that can have any numeric result, we use the switch statement that selects an expression with results equal to a specified constant.

We can redo the if-else block used in the joystick interrupt example using a switch statement as follows:

```c
switch(input){
    case KEY_UP:
        keyUP();
        break;
    case KEY_DOWN:
        keyDOWN();
        break;
    case KEY_LEFT:
        keyLEFT();
        break;
    case KEY_RIGHT:
        keyRIGHT();
        break;
    case KEY_PUSH:
        keyPUSH();
        break;
    default:
        keyERROR();
        break;
}
```

This works just like the if-else block. The 'break' statement causes an immediate exit from the switch block — there is no need to check the rest as we have found our case. If you want to continue evaluating cases against the input, leave out the break and the next statements will be looked at. You can let cases fall through, which can be handy in circumstances such as evaluating character input where you don’t care if the character is a capital or lower case letter, or perhaps you want the same response for a range of integers:

```c
switch(input){
    case 'a':
    case 'A':
        doA();
        break;
    case 'b':
    case 'B':
        doB();
        break;
    case '0':
    case '1':
    case '2':
    case '3':
        gofer0123();
        break;
    case '4':
    case '5':
    case '6':
    case '7':
        gofer4567();
        break;
    default:
        doDefault();
        break;
}
```

Switch statements are error prone and a frequent source of head bonking bugs (one where you bonk your head for being dumb enough to leave out a break statement). The break after default: isn’t even necessary, but is recommended (by K&R) as a good practice to help you remember to use it when you add a statement
Loops - While, For, and Do-while

We've been using `while` for a while (harr!).

```c
while(expression)
{
    // Do stuff while expression is true
}
```

While will repeat the associated statement or block as long as the expression is true. The code fragment:

```c
int i;
while( i < 128 )
{
    PORTD = i;
    _delay_loop_2(30000);
    i = i*2;
}
```

This does the same thing as the following `for` loop:

```c
int i;
for( i = 1; i < 128; i = i*2)
{
    PORTD = i;
    _delay_loop_2(30000);
}
```

The `for` loop is constructed as follows:

```c
for(expression1; expression2; expression3)
{
    // Do stuff
}
```

Usually, `expression1` and `expression3` are assignments or function calls and `expression2` is a test of some sort. The expressions can be any expression including the empty expression which is nothing followed by a semicolon:

```c
for(;;)
{
    // Do stuff forever
}
```

This is an alternative way to do the `while(1)` eternal loop. You can accomplish the same goal using either `while` or `for` statements. Generally, it is clearer to use for loops with a simple initialization and incrementing such as:

```c
for(i = 1; i < 128; i = i*2)
{
    // Do stuff while 'i' less than or equal 128
}
```

It's really a matter of personal preference though most C dudes will want to smack you around a little if you don't do it their way. While and for loops test for the termination condition before running the block; 'do while' runs the block first before checking the expression, insuring that the block will be run at least once:

```c
do
{
    // Do stuff at least once
}
while(expression):
```

A Butterfly Menu System

If you still have the box your Butterfly came in, you can see the full Butterfly menu laid out for you (or you can find it in Figure 2-2 Application Menu in the Butterfly_UserGuide.pdf in the workshop6.zip file on the Nuts & Volts website). For this project, we will recreate roughly the first half of the Butterfly menu system (Figure 2: Partial Butterfly menu). Look at this figure for a while and think about how you might do this task based on the C syntax flow control discussion above.

We will make heavy use of the switch statement for creating our menu system. For our hardware, we will use the Butterfly LCD to display the menu text and the joystick to navigate around the menu and make menu
item selections. The joystick gives us the option of ‘moving’ the menu up or down to view main menu items, left or right to view submenu items, or press to the center to select an item. We will hide all the details of using the LCD and joystick in the smws6 object module that I’ve precompiled for you and put in the C:\smws6\default directory so that you don’t have to blow your mind looking at the source code for a lot of LCD and joystick support functions that are a bit advanced and messy (you don’t need to understand them to understand menus). I strongly suggest that you copy the entire source directory (in workshop6.zip) and keep a virgin copy somewhere convenient so that as you mess with these ideas you can go back to a version that works.

The original Butterfly source code uses some intense C ideas to accomplish this task, but we will use something a bit clearer for early learning purposes. More advanced programmers will react to this code much like the waiter in Figure 1. We start off in an initial menu state which, in our case, is state00 (AVR BUTTERFLY). After running the action function for this state: menuState00Func(), we enter an eternally loop and check for input from the UART or the joystick buttons. (We won’t discuss the UART functions — they were useful for debugging and I left them in.) When the checkJoystick() function tells us that there is a button press pending, we call the parseJoystickInput() function.

```c
int main(void)

{  
    volatile uint8_t menuState;

    // what state are we in?
    volatile uint8_t menuState;

    This is a global eight-bit unsigned (uint8_t) variable declared as volatile so that the compiler knows it may change unpredictably and won’t try to second-guess you and eliminate it if it thinks it isn’t really being used (compilers are also a lot like the waiter in Figure 1). The parseJoystickInput function is a 17 case switch statement that calls

    state function for the current menuState variable:

    void parseJoystickInput()
    {
        switch(menuState)
        {
            case 0:
                state00();
                break;
            case 1:
                state01();
        }
    }
```

---

$51 PCBs
FREE Layout Software!
FREE Schematic Software!

01 DOWNLOAD our free CAD software
02 DESIGN your two or four layer PC board
03 SEND us your design with just a click
04 RECEIVE top quality boards in just days

expresspcb.com

January 2009 NUTSVOLTS 83
Joe Pardue can be reached at www.smileymicros.com

break;
case 2:
    state02();
    break;
// Cases 17 through 16 deleted to save space

    state07();
    break;
default:
    // TODO: DEAL WITH ERROR
    break;
}

Note that this function is not related to any particular joystick button; it simply calls the state function for the current state if there is joystick input. The state function that gets called will look at the global variable ‘joystickInput’ value and act accordingly.

We will navigate the menu states by using the joystick buttons: UP, DOWN, LEFT, and RIGHT. We will decide if an action other than moving about the menu states should be done by looking at the joystick button PUSH. So, if we are in a given menu state and we get any of the movement states, we move if there is another state in that direction (and change the menu state to the new state) or we ignore it if there is no valid state in that direction (no change to the menu state). If we see that the joystick button is PUSH, we keep the state the same, but process the PUSH action specified for that state. If, at this point, your personal state is thinking about tearing up the magazine and finding a French waiter to punch out, try to bear with this for a while. Read the code snippets below and reread this section if necessary. It may take a while to get your head around these concepts. We will also keep track of whether the joystick button has been pressed to the center, which we will use to take actions not related to menu state changes:

// do we need to something?
volatile uint8_t keyPush;

To repeat: When we get a new joystick button state, we can do one of two things: change the menu state or take an action for the current menu state.

Let’s look at a single state and see what can happen. Look at state03 (CLOCK) in Figure 2 and think about what happens next when the joystick button changes to:

1. Up - Do nothing.
2. Down - Change to state07.
3. Left - Change to state02.
4. Right - Change to state04.
5. Push - Set keyPush equal true.

When we change menu states, we will call a function with the actions for that new state. If the button is not a menu state change but is PUSH, we will call the action function for the current menu state. We can write this as:

void state03()
{
    switch(joystickInput)
    {
    case KEY_UP:
        // Do nothing
        break;
    case KEY_DOWN:
        // Change to menu state07
        menuState = 7;
        menuState07Func();
        break;
    case KEY_LEFT:
        // Change to menu state02
        menuState = 2;
        menuState02Func();
        break;
    case KEY_RIGHT:
        // Change to menu state04
        menuState = 4;
        menuState04Func();
        break;
    case KEY_PUSH:
        // Call this menu state function
        // with keyPress = keyPush = 1; //true
        // take action for this state
        menuState03Func();
        break;
    default:
        // Do nothing
        break;
    }
    joystickInput = KEY_INVALID;
}

As usual, this is just an overview of the actual code. You can play with it by getting Workshop6.zip from www.smileymicros.com or the Nuts & Volts website at www.nutsvolts.com. We are doing something different this time. Instead of using a library for the LCD and joystick functions, we are using an object module (smw68.0) that must be located in the C:\MenuTest\default directory so that the MenuTest project can find it. I avoided mentioning ‘state machine’ up to this point since these two words — when connected — tend to send folks running for the exits. But, guess what? You just studied a state machine and your head didn’t explode (I hope). Yes, state machines are often the topics of computer science doctoral dissertations, but they can also be as simple as a switch statement.

We have three more workshops that will complete our study of introductory C syntax, then we will assume the reader already knows enough C and move on to some hardware oriented projects. Next month, we will continue with more C syntax and build a Butterfly alarm clock.
Build Your Own: THEREMIN

Theremax
The Ultimate Theremin

PAIA
Electronic Music Kits and More

Wireless Made Simple
Bring your product quickly and legally to market

RF Modules
Add instant wireless analog/digital capability to your product

Low-Cost TX, RX
& TRX Modules

Multi-Channel Modules

Long-Range Modules

OEM Products
FCC Pre-Certified & ready to customize for your application

Handheld TXs

Function Modules

Feature Products
LINX-COST - LONG RANGE TRANSCEIVER

- Direct serial interface
- Low power consumption
- PLL-synthesized architecture
- RSSI and power-down function
- Compact surface mount package
- No external RF components (except antenna)

A closer look at LINX Innovation

REMOTE CONTROL TRANSDUCER IC

- Up to 8 inputs
- Bi-directional control
- Transmitter ID output
- Automatic confirmation
- Secure 2* possible addresses
- Latched and/or momentary outputs

800-736-8677
169 Or Lane, Merlin, OR 97532
www.linxtechnologies.com

The Lynxmotion Servo Erector Set
Imagine it... Build it... Control it!

With our popular Servo Erector Set you can easily build and control the robot of your dreams. Our interchangeable aluminum brackets make the ultimate precision mechanical assemblies. Our Visual Sequencer for SSC-32 provides powerful PC, Atom, or BS2 based control. Visit our huge website to see our complete line of aluminum and Lexan based robot kits, electronics, and mechanical components.

Bipeds

www.lynxmotion.com
Toll Free: 866-512-1024

SES now has 62 Unique Aluminum Components!

January 2009

http://nutsvolts.texterity.com/nutsvolts/200901/templates/pageviewer_print?pg=1&p...
**“EDITOR’S PICKS”**

**Electronics Projects for Dummies**

Whether you’re new to hands-on electronics or you want to gently introduce someone else to the art, this book provides the information and instruction you need, from how to read a circuit diagram to building some cool circuits. **$24.95**

**46 Science Fair Projects for the Evil Genius**

A perfect starter text for budding scientists, this approachable and fun book has something for everyone. The projects are both affordable and intellectually challenging. **$19.95**

Editor Bryan Bergeron’s recommended reads. Find these and many more great titles in the NUTS & VOLTS Webstore!

---

**2008 CD-ROM**

**NUTS AND VOLTS**

*Volume 58 No. 1-12*

The 2008 CD-ROM has arrived! 12 issues on 1 CD, with all your favorite articles, projects, media downloads, and advertisers.

---

**ELECTRONICS**

**Beginner's Guide to Embedded C Programming Combo**

Book and Kit can be purchased separately. The C language has been covered in many books but none as dedicated to the embedded microcontroller beginner. Through his down-to-earth style of writing, Chuck Hellebuyck delivers a step-by-step introduction to learning how to program microcontrollers with the C language. The PICkit 2 Starter Package is used throughout the book and includes the HI-TECH PICC-Lite Compiler and a PIC16F84A. **Reg $89.95 Sale $86.95**

**Fundamental Electrical and Electronic Principles**

A thorough introduction to electrical and electronic engineering at Level 3. Fundamental Electrical and Electronic Principles covers the essential principles that form the foundations for electrical and electronic engineering courses. The coverage of this new edition has been carefully brought in line with the core unit ‘Electrical and Electronic Principles’ of the 2007 BTEC National Engineering specification from Edexcel. **Reg $36.95 Sale $35.95**

**Build Your Own Electronics Workshop**

**YOUR DREAM ELECTRONICS LAB IS WAITING INSIDE!**

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 — popular electronics author Tom Petrussellis’ Build Your Own Electronics Workshop is all! And the best part is, this book will save you money, big time! **Reg $29.95 Sale Price $24.95**

**Getting Started in Electronics**

**by Forrest M. Mims III**

Author Forrest Mims teaches you the basics, takes you on a tour of analog and digital components, explains how they work, and shows you how they are combined for various applications. Includes circuit assembly tips and 100 electronic circuits and projects you can build and test. Forrest Mims, III, has written dozens of books, hundreds of articles, invented scientific devices, and loves to share his knowledge with eager students. **$19.95**

**Getting Started with PICs**

A Collection of 2006 Nuts & Volts Magazine Articles **by Chuck Hellebuyck**

In the January 2006 Nuts & Volts Magazine, Chuck Hellebuyck started a column titled “Getting Started with PICs” that focused on the beginner. It became a popular column and continues in Nuts & Volts Magazine today. Many readers missed the early columns so this book contains the original 2006 articles compiled into a book for easy reference. **$29.95**

**Blu-ray Disc Demystified**

by Jim Taylor, Michael Zink, Charles Crawford, Christen Armbrust

**Pub Date:**
November 10, 2008

**The Latest Tips and Techniques for Getting the Maximum from Blu-ray Technology.**

Written by a team of experts with a wide range of experience in DVD and high-definition production, this thorough, skill-building guide details Blu-ray's many features, formats, and applications. **$59.95**
THE EVIL GENIUS SERIES

If you like books on projects, check out The Evil Genius Series. We have all 23 books in stock and ready to ship. Buy three or more books from this series and get free USPS shipping! (US only)

Virtual Serial Port Cookbook by Joe Pardue
As talked about in the Nuts & Volts June issue, “Long Live The Serial Port”
Book $44.95
This is a cookbook for communicating between a PC and a microcontroller using the FTDI FT232R USB UART IC. The book has lots of software and hardware examples. The code is in C and Visual Basic Express allowing you to build graphical user interfaces and add serial port functions to create communications programs.
The Virtual Serial Port Parts Kit and CD (also available, above right)
Combo Price $115.95 Plus S/H $8.95

Programming and Customizing the PICAXE Microcontroller by David Lincoln
Pub Date 09/2005
Here’s everything you need to harness the power of PICAXE, the inexpensive yet versatile microcontroller. This beginners’ guide with a PICAXE expert David Lincoln shows you just what Revolution Education’s PICAXE can do and helps you make it do it!
Reg $39.95 Sale $34.95

Encyclopedia of Electronic Circuits, Volume 7 by Rudolf F. Graf / William Sheets
Publish Date: August 31, 1998
An extensive library of over 1,000 circuits from the bestselling, seven-volume Encyclopedia of Electronic Circuits. A virtual treasurehouse. An invaluable reference tool for every hobbyist, technician, student, and design professional.
$39.95

Newnes Guide to Television and Video Technology by K. F. Ibrahim
Starting with TV fundamentals, the bulk of the book covers the many new technologies that are bringing color to the TV and video market, such as plasma and LCD, DLP (digital light processing), DVD, Blu-ray technology, high-definition television (HDTV), and video projection systems.
$34.95

BOOK & KIT COMBOS

From the Smiley’s Workshop
C Programming for Microcontrollers by Joe Pardue
Book $44.95
Kit $66.95
Do you want a low cost way to learn C programming for microcontrollers? This 300 page book and software CD show you how to use ATMEL’s AVR Butterfly board (available in the kit, above right) and the FREE WinAVR C compiler to make a very inexpensive system for using C to develop microcontroller projects.
Combo Price $99.95 Plus S/H $8.95

GREAT ARTICLES!

Catch up on 2003-2008 back issues
These CD-Roms are PC and MAC compatible. They require Adobe Acrobat Reader Ver.6.

$24.95
More offers go to: www.nutsvolts.com

CD-ROMS

GREAT PROJECTS!

January 2009 NUTSVOLTS 87
### High Voltage Power Supply Kit

It's fun to collect and experiment with forgotten technology! But, you'll need a stable high voltage power supply to get started.

To conserve space, this kit uses a mix of TH and SMT components. Sold as a kit or the printed circuit boards can be purchased separately. **For more information, please check out your April 2008 issue or go to our website @ www.nutsvolts.com**

**KIT**
- **Subscriber’s Price**: $49.45
- **Non-Subscriber’s Price**: $54.95

Includes an article reprint.

---

### Magic Box Kit

As seen on the April 2008 cover

This project blends electronics technology with hand craftsmanship. The clever trick has the observer remove one or two of six pawns while you are out of the room and upon re-entering you indicate the missing pawns without ever opening the box.

**For more info go to our webstore @ www.nutsvolts.com**

**Reg. Price**: $47.55
**Subscriber’s Price**: $42.95

Includes an article reprint.

---

### Switching Regulator Kit

Need to regulate your power output? Take a look at this new and improved regulator. **For more information, please check out your June 2008 issue or go to our webstore @ www.nutsvolts.com**

**Subscriber’s Price**: $29.95
**Non-Subscriber’s Price**: $32.95

Kit includes an article reprint.

PCBs can be bought separately.

---

### The Amateur Scientist 3.0

**The Complete Collection**

By Bright Science, LLC

There are 1,000 projects on this CD, not to mention the additional technical info and bonus features. It doesn’t matter if you’re a complete novice looking to do their first science fair project or a super tech-head gadget freak; there are enough projects on the single CD-ROM to keep you and 50 of your friends busy for a lifetime! $26.99

---

### Freeze Fountain Kit

This kit has all the components you need to start amazing your friends by creating the illusion of water drops frozen in time. You can even show them dripping upwards!

**For more information, please check out your May 2008 issue**

**Regular Price**: $109.55

**Sale Price**: $88.88

Includes an article reprint.

---

### IN-14 Nixie Tube Clock Kit

Cherry Wood Clock Case

Nixie tube clocks fuse the spirit, drama, and eerie beauty of cold war technology with modern inner works to create uncommon handcrafted timepieces.

Clock kits sold in a variety of configurations. **Get more info @ www.nutsvolts.com**

**IN-14 Nixie Tube Clock Kit with Cherry Wood Case**
- **Subscriber’s Price**: $146.95
- **Non-Subscriber’s Price**: $169.95

Kit includes an article reprint, complete instructions, and parts list.

---

### The SERVO Buddy Kit

An inexpensive circuit you can build to control a servo without a microcontroller!

**For more information, please check out our website @ www.nutsvolts.com**

**Subscriber’s Price**: $39.55
**Non-Subscriber’s Price**: $43.95

Includes an article reprint.
Alternative Energy Demystified
by Stan Gibilisco
Publish Date: October 23, 2006
The fast and easy way to get up-to-speed on alternative energy. Because of current events, geopolitics, and natural disasters, the cost of fuel is front and center in our lives. This book provides a concise look at all forms of energy, including fossil fuels, electric, solar, biodiesel, nuclear, hydroelectric, wind, and renewable fuel cells. You will get explanations, definitions, and analysis of each alternative energy source from a technological point-of-view. $19.95

Kill A Watt Motor
Tired of wasting money from THE PHANTOM DRAW? The KILL A WATT meter is the best way to determine your actual energy draw from ON and OFF home appliances. $29.95

Build Your Own Electric Vehicle
by Seth Leitman, Bob Brant
Publish Date: October 10, 2008
Go Green-Go Electric! Faster, Cheaper, More Reliable While Saving Energy and the Environment!
This comprehensive how-to goes through the process of transforming an internal combustion engine vehicle to electric or even building an EV from scratch for as much or even cheaper than purchasing a traditional car. The book describes each component in detail—motor, battery, controller, charger, and chassis—and provides step-by-step instructions on how to put them all together. $29.95

Solar Power Your Home For Dummies
by Rik DeGunther
Publish Date: Dec 2007
This friendly, hands-on guide is packed with tips for making your home more energy-efficient through solar power—and helping the planet at the same time. You'll see how to survey your home to determine your current household energy efficiency and use, and evaluate where solar power would best benefit you. You'll also calculate what the return on your investment will be before you make any decisions. $19.95

Run Your Diesel Vehicle on Biofuels
A Do-It-Yourself Manual
by Jon Starbuck, Gavin D.J. Harper
CONVERT TO BIODIESEL FOR A MORE ENVIRONMENTALLY FRIENDLY RIDE!
Run Your Diesel Vehicle on Biofuels has everything you need to make the switch from expensive, environment-damaging carbon fuel to cheap (and, in many cases, free), clean fuel for your vehicle. Practical and decidedly apolitical, this unique guide focuses on technical details, parts, and instructions. $24.95

Energy Technology and Directions for the Future
by John R. Fanchi, PhD
Energy Technology and Directions for the Future presents the fundamentals of energy for scientists and engineers. It is a survey of energy sources that will be available for use in the 21st century energy mix. The reader will learn about the history and science of several energy sources as well as the technology and social significance of energy. Reg. $78.95 Sale $61.95

Solar Hydrogen Power Generation Set
The Solar Hydrogen Generation Set includes a solar cell, a PEM reversible fuel cell, as well as oxygen and hydrogen gas containers. The set only needs pure water to create hydrogen and produce electricity. Perfect for science labs, classroom use, or demonstration purposes. $67.00

HOLIDAY SPECIAL OFFER
FREE! with any purchase over $125.00 Phone & orders only!
FREE Nuts & Volts Embroidered Hat with orders over $125.00 of any product combination!
Offer ends January 31st, 2008. Call 1 800 765-4624

January 2009 NUTSVOLTS 89
**CLASSIFIEDS**

**SURPLUS**

**SURPLUS ELECTRONIC PARTS & ACCESSORIES**

- **Cables**
- **Connectors**
- **Displays**
- **Fans**
- **Hardware**
- **LEDs**
- **Motors**
- **Potentiometers**
- **Speakers**
- **WIRING & ACCESSORIES**
- **Tools**

**Surplus Material Components**

SME ELECTRONICS

www.smeelectronics.com

No Minimum Order. Credit Cards and PayPal Accepted. Flat $4.95 per order USA Shipping.

**COMPONENTS**

**RF PARTS™ CO.**

Tubes, Transistors, Power Components

800-737-2787 Fax 888-744-1943

**STA PARTS**

Obsolete & current production parts are out there if you know where to find them. STA specializes in finding those hard to find parts for you. Call today and end your search with STA.

**uM-FPU V3.1**

Floating Point Coprocessor

32-bit IEEE 754
SPL or IEC
DIG-18
SOG-18

Fast – Easy to Use

Extensive floating point support,
GFP input, FFT, 12-bit A/D, matrix operations,
user-defined functions.

www.micromegacorp.com

**Starter Components**

Create your own - RGB LED Ribbon
RGB LED display + LED Painters
Prop Blade

www.brillidea.com

**MILITARY SURPLUS**

Electronic Military Surplus

30 Foot Mast Kit, $139.50
PRC-47 HF Transceiver, $150.00
6 Position Ant Tap Switch, $13.95
SMRDI 1.2KV Capacitors, $17.50
UHRM-25 HF Signal Generator, $110
RT-334 Ground Radio 2-30MHz, $300
Yagi Antenna 900-950MHz: $19.95
ME-165 Power Meter 600W, $125.00
PRO-26 Backpack Radio, $300.00
HP-231A Distortion Analyzer, $225
5KV Doorknobs 10, 25, 50, 75, $6 ea

various military electronic kits

Visit our website for more info!

**KITS/PLANS**

**OKITS.COM**

50 AMP PWM

1-888-GO 4 KITS

Electronic Kits

- Fun and Educational
- Discount White and Blue LEDs
- Quantity Discounts

www.bakatronics.com

**PCB SOFTWARE**

EXCLUSIVE U.S. DISTRIBUTOR

Circuit Wizard

YOUR COMPLETE SOLUTION!

- PCB Design
- PCB Simulation
- CAD/CAM Menu

121747, Hobby Vers., $175
121743, Pro. Version, $275

www.KELVIN.com

**BOOKS**

**Books**

PIC® Microcontroller

sq-1.com

Stepper Motors

stepperstuff.com

CNC

cncintro.com

**CONNECTORS/ WIRE/CABLE**

SCSI

1-EZ39 20Pin-68Pin-80Pin
1 to 8 Bay Case Enclosures
Adaptors Cables Terminators
Low Prices - Qty Discounts!
(Also FireWire, USB, Video)

www.mepb.com

**SOLAR PRODUCTS/KITS**

**BATTERY DESULFATORS**

www.wizbangplus.com

**Micro Circuit Labs**

ELECTRONIC KITS

www.microcircuitslabs.com

Solar Data Logger

Digi Temp

$199.95

Solar Irradiance Meter Kit $22.95

Adjustable Power Supply Kit $11.95

**LCDs/DISPLAYS**

**Smart Board**

$39.95

2004.01

150-degree

256x128

20mA

www.411techsystems.com

- Print Pages 90 of 100 -
This is a READER-TO-READER Column.

All questions AND answers are submitted by Nuts & Volts readers and are intended to promote the exchange of ideas and provide assistance for solving problems of a technical nature. Questions 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 judgement!

All questions and answers should be sent by email to forum@nutsvolts.com All diagrams should be computer generated and sent with your submission as an attachment.

QUESTIONS
To be considered, all questions should relate to one or more of the following:
- Circuit Design
- Electronic Theory
- Problem Solving
- Other Similar Topics
- Be brief but include all pertinent information. If no one knows what you’re asking, you won’t get any response (and we probably won’t print it either).
- Include your Name, Address, Phone Number, and email. Only your Name, City, and State will be published with the question, but we may need to contact you.
- No questions will be accepted that offer equipment for sale or equipment wanted to buy.
- Selected questions will be printed one time on a space available basis.
- Questions are subject to editing.

ANSWERS
- Include in the subject line of your email, the question number that appears directly below the question you are responding to.
- Payment of $25.00 will be sent if your answer is printed. Be sure to include your mailing address or we cannot send payment.
- Only your Name, City, and State, will be printed, unless you say otherwise. If you want your email address included, indicate to that effect.
- Comments regarding answers printed in this column may be printed in the Reader Feedback section if space allows.

92 NUTSVOLTS January 2009

>>> QUESTIONS

Three-Phase Power —
Rotary Phase Converters
Is there a simple design guide available for calculating proper capacitor sizes used in the construction of rotary phase converters that use ordinary three-phase motors as the rotary transformer?
#1091 Jack Kleiss Cloverdale, IN

Three-Phase Power
Does anyone know of a simple, inexpensive, but very accurate way, to monitor three-phase current in brushless motor circuits? The method should require, at most, one channel of ADC.
#1092 Jack Kleiss Cloverdale, IN

Video Adapter
I would like to feed the modulated video output from my DVD player to a SVGA video monitor. Other than stripping the modulation from the signal, I'm not sure what to do next.
#1093 Gary Johnson Camarillo, CA

Svideo Luminance
I have a DVD player with Svideo out; I need to build a simple circuit to use the luminance pins to trigger a relay as soon as luminance is black.
#1094 Karpis Los Angeles, CA

One Minute (or More) Timer
I need a way to reliably trigger a 555 from a loss of input signal. I've tried several schemes involving capacitor discharge when the input disappears but they have not proven to be reliable. Keeping the input high apparently doesn't let the device time out, so it needs to be a positive going pulse.
#1095 David Schoepf Marianna, FL

Logic Analyzer
I have a LEADER Model 300 DMM/scope with logic analyzer. I need help/info to build an interface circuit that would allow me to use the eight channel logic analyzer.
#1096 Jeff Clark Sardinia, OH

SSR
I would like to know how to calculate input resistance and design with a solid-state relay used as a switch.
#1097 Cesar Romero Leonia, NJ

Humidity Sensor
I am looking for a small humidity sensor circuit powered by 120V that can trigger a relay based on an adjustable humidity level. My ultimate goal is to wire it into a bathroom fan and have the relay trigger an X10 command via an X-10/Insteon I/O module. The X10 command would turn the fan on or off based on the humidity in the room and the
humidity trigger level set. It needs to be small and self-contained (no wall wart).

Michael McNabb
Issaquah, WA

Electronic Simulation Software
I see simulations of circuits in many articles. I used to have a copy of Electronic Workbench 12 years ago, but found it to be unreliable in switch mode designs. I would like to know what package you recommend for mixed signal and switch mode design simulation that is easy to use and under $1,000. Also, I have been out of electronics for six years and need a single source for info on SMT packaging.

Jeff Crutchfield
Oak Ridge, NC

No On-Screen Keyboard Please
I have Windows Vista Home Basic Edition installed on my ACER (Aspire 3690-2900) laptop. How do I stop from ever seeing the on-screen keyboard, which I found to be useless. Please explain step-by-step how to stop it from appearing on the screen each time I turn on my laptop.

Michael via email

ANSWERS

Accessing Serial Ports With VB
I am using MComm32.ocx in an MS ACCESS application, using VBA. I need to send and transmit data with this control. I have the mcomm32.ocx file installed on my PC, but I have never transmitted data over a serial port before this project. Please help.

The MComm32.ocx control is a component provided with Microsoft Visual Basic (VB). Attempting to use this control on a computer that does not have VB installed may result in the error "You do not have a license to use this ActiveX control." This is a common scenario when a VBA application is developed on one machine and deployed to another. Two registration steps that are performed automatically by the VB setup program on the development machine must be performed manually on the deploy-

#2 It sounds like you are working on a later model car and are connecting to one of the existing O2 sensors. This is a normal response for the vehicle to establish catalytic converter operation. The lean cycles provide air for the catalytic converter thus eliminating the need for an air pump. You don’t want to introduce anything to alter the operation of this O2 sensor. However, you can have a muffler shop install an insert with a plug for an O2 sensor in your exhaust system, before the catalytic converter. (Post cat will not have mixture readings.)

Locate and purchase another O2 sensor, but check with the mixture gauge manufacturer for recommendations. What you are looking for is an O2 sensor with a slower response, possibly a single wire from an earlier car. These were commonly called Lambda sensors or Hego sensors. They usually use Zirconia as the active material. Finally, remove the plug and install your new sensor and follow the instructions to center the mixture gauge.

John Golding Jr
Conyers, GA

Guitar Tuner
I am designing a microcontroller-based guitar tuner. What are the frequencies of the open strings (in Hertz)?

The frequency of guitar strings is based on the notes assigned to the open strings. The thickest string is the sixth string and the thinnest is the first string. The quick answer to your question is in the following table.

<table>
<thead>
<tr>
<th>String #</th>
<th>Note</th>
<th>Frequency Hz</th>
</tr>
</thead>
<tbody>
<tr>
<td>6th</td>
<td>E</td>
<td>82.4</td>
</tr>
<tr>
<td>5th</td>
<td>A</td>
<td>110.0</td>
</tr>
<tr>
<td>4th</td>
<td>D</td>
<td>146.8</td>
</tr>
<tr>
<td>3rd</td>
<td>G</td>
<td>196.0</td>
</tr>
<tr>
<td>2nd</td>
<td>B</td>
<td>246.9</td>
</tr>
<tr>
<td>1st</td>
<td>E</td>
<td>329.6</td>
</tr>
</tbody>
</table>

Some background is probably in order as to where the notes come from and the physics, or dare I say, the mathematics involved. (Please keep reading, this will be very basic.)

Western music is based off the A

January 2009 NUTSVOLTS 93
note set at 440.0 Hz and all other notes are based off of that. This is a tempered scale and each note is a multiple of the note preceding it. The multiplication factor is the 12th root of 2 or about 1.0595. To go up a note from 440 A, you multiply 440 by 1.0595 to get 466.18 Hz or A#. To go down a note from A, you divide 440 by 1.0595 to get 415.29 Hz or G#. To go down still another note from G# to G, you divide 415.29 by 1.0595 to get 391.96 Hz.

The 440.0 Hz note A on a guitar is the first string, fifth fret. Holding the first string down at the fifth fret and tuning this string to 440 Hz puts it at A. When released, the string will be tuned to 329.6 Hz E. Holding down the second string at the fifth fret is the same as E on the open first string. Hold down the second string at the fifth fret and tune it to match the open string. Now it gets tricky. The fourth fret on the third string is the same as the 246.9 Hz B on the second open string. Hold down the third string at the fourth fret and tune it to match the open string. The fifth fret on the fifth string is the same as the fourth open string. And the fifth fret on the sixth string is the same as the open fifth string. Tuning a guitar in this manner puts it in what is called concert pitch.

Tim Naami
Poplar Grove, IL

[11088 - November 2008]
Using a PC Power Supply as a Stand-Alone Supply

I would like to use a spare PC power supply as a stand-alone source for the 5V and 12V it provides. What is the simplest way to provide the 5V feedback for a power good signal? Is it possible to use the PC signal to also act as a power on signal, as well?

The first thing to do is to check out the article "How to Convert a Computer ATX Power Supply to a Lab Power Supply" at www.wikihow.com/Convert-a-Computer-ATX-Power-Supply-to-a-Lab-Power-Supply.

The principal reference for the modern PC power supply is the Intel ATX Specification; a copy of which can be found at http://download.intel.com/design/motherbd/atx_201.pdf. Section 4 "ATX Power Supply" discusses the power-supply interface. The 5VSB output can be used to operate a small pilot light (e.g., an LED in series with a 1K ohm resistor); its maximum output rating is 0.72 amperes. You must hold PS-ON low (tie it to the COMmon conductor) in order for the power supply to energize the major outputs. PWR_OK is held high (to the +5V level) by the power supply whenever the +3.3V and +5V outputs are properly energized; you should connect it to COM through a 1K ohm resistor so that it can be pulled low to indicate power failure.

One of the +3.3V contacts in the main connector is used to sense the voltage at the mating connector (i.e., on the PC board in its normal application) in order to compensate for voltage loss in the power supply cable. Remember that you are dealing with low voltage at several tens of amperes of current here. In any event, it's very important to tie all of the like-named contacts together at the mating connector (or on the PC board) -- COM to COM, +3.3V to +3.3V, etc.

If your power supply has the optional connector with a 3.3V sense line, tie the sense line to the +3.3V rail at the LOAD TERMINALS.

Also, read www.ibm.com/developerworks/power/library/pa-spec9.html about some of the exceptions that you may find across the universe of PC power supplies. And beware of Dell supplies which, in my experience, do not conform to the ATX connector wiring standard.

Peter A. Goodwin
Rockport, MA

[11088 - November 2008]
Parallel Port

How does the parallel transfer of data work? Does data flow out from the computer alone, or can data be read into the computer from the port?

A number of bidirectional parallel port configurations have been made available for the PC over the years, but even the standard parallel port (SPP) provides for TTL level communication on eight output and eight input lines.

The accompanying table shows the DB25 pins which are used. Digital output is carried on the printer data lines. Digital input is read from status and control lines. Note that three lines are active low (0 VDC), while all others are active high (5 VDC). It is advisable to buffer the signals using the LS74LS244 or similar, to protect against shorts and other incorrect wiring while experimenting.

Typically, LPT1 is assigned to I/O base address 03BCh, LPT2 to 0378h, and LPT3 to 0278h. To output data, write the bit pattern directly to the port. For example, outporth(0x03BC, 0xFF) to turn on all bits. The status inputs are read at base address + 1 (bits 3-7), and the control inputs are read at base address + 2 (bits 0-2).

So, a complete input byte is read using
inporth(0x03BC + 1) & 0xFF | (inporth(0x03BC + 2) & 0x07) and all bits are converted to represent active high using inbyte ^= 0x83. While my examples are written in C language, other languages provide similar port reading and writing facilities.
SPP Cable Pin Assignments

Pin Signal
- OUTPUT — - INPUT —
2 Data Bit 0 1 Strobe (IN bit 0)
3 Data Bit 1 14 Auto Feed (IN bit 1)
4 Data Bit 2 16 Initialize (IN bit 2)
5 Data Bit 3 13 Error (IN bit 3)
6 Data Bit 4 13 Select (IN bit 4)
7 Data Bit 5 12 Paper Out (IN bit 5)
8 Data Bit 6 10 Acknowledge (IN bit 6)
9 Data Bit 7 11 Busy (IN bit 7)
18-25 Ground
*Note: Active low signals

John Reynolds
Rochester, NY

[12/085 - December 2008]

Timer Circuit

I am looking for a timer circuit to turn on and provide a source of standard 120 VAC household power for a cycle/duration of 24 hours, once every set number of days (e.g., every second day, or every third day, up to every 12th day). The start and stop times would be at the same hour of day.

Figure 1 and Listing 1 show a simple design and program code with a minimum parts count using a PICAXE 08M microprocessor. No fancy clock or display circuitry required. Unit operates from a 12 VDC wall wart and includes a 9V battery for clock back-up if AC power is lost during operation. Since the start and stop times will always be at the same hour of the day, the circuit will be initially programmed at that hour. This will start a 24 hour reference clock in the program code.

Program procedure is as follows:

1) Power switch on, program checks nine-volt battery and blinks three times rapidly if battery is low or not connected.
2) If battery "low" is indicated, the battery must be replaced or connected to continue programming.
3) LED blinks once every second 12 times indicating the "Days Off" to be selected, pauses two seconds and repeats.
4) Press start button after number of blinks (days off) required; i.e., two blinks indicates a three day cycle (one on, two off).
5) Program will now blink the LED the number of "Days Off" previously selected; pause one second then repeat.
6) If incorrect, turn power switch off, then on to reset unit back to step 1.
7) Press and hold start button until LED is steady on to begin operation.
8) The LED shuts off after two seconds.

Unit is now programmed and a 24 hour "AC ON" operation begins at this hour of the day. The LED will blink the "Days Off" selected once a minute during the "AC ON" and "Days Off" operations to indicate it is functioning. The 120 volt AC power receptacle must be wired to local code requirements.

Steve Ghioto
Jacksonville, FL

Listing 1

```plaintext
*Timer Code

days_off: pause 2000
readsec: 4, b7
if b7 < 95 then low_batt
for b0 = 1 to 12
pulsout 2,10000
next b0
next b4
go to days_off

required

for b1 = 1 to 10
if pin1 = 0 then
check_off
pause 100
next b1
next b0
go to days_off

check_off: pause 1000
for b1 = 1 to b0
pulsout 2,10000
pause 400
if pin1 = 0 then
start
next b1
go to check_off

start: high 2
high 0
pause 2000
low 2
for b2 = 0 to 23
for b3 = 0 to 59
for b4 = 0 to 49
pause 1000
next b4
gobub day_cycle
next b3
next b2
low 6
for b1 = 1 to b0
for b2 = 0 to 23
for b3 = 0 to 59
for b4 = 0 to 49
pause 1000
next b4
next b3
gobub day_cycle
next b2
next b1
go to start

day_cycle: b5 = 1
for b6 = 1 to b0
pulsout 2,10000
pause 400
lnc b6
next b6
w4 = 20 + b5 * 500 + 30
pause w4
return

low_batt: pulsout 2,10000
pause 100
pulsout 2,10000
pause 100
pulsout 2,10000
go to days_off
```

January 2009 NUTSVOLTS 95
Enter the DLP Design

**DLP-232PC DESIGN CONTEST**

for your chance at winning a 65-inch Hi-Definition LCD TV!

When small size and low cost are at the top of the design constraints list, look no further! The DLP-232PC offers 14 I/O channels (eight can be configured for analog input) and USB communication with a host PC at up to 460K baud. Pit your design skills against your colleagues and come up with the most innovative, smallest and lowest-cost design possible using this tiny USB-to-microcontroller module, and you just might walk away with a new Hi-Def LCD TV or one of five other cool prizes.

**DLP-232PC FEATURES:**
- FT232RQ USB IC from FTDI
- 14 I/O (eight 10-bit analog channels)
- PIC18F2410 16K ROM, 768 RAM
- USB port powered
- Tiny Footprint: 1.375x.6 inches
- No in-depth knowledge of USB required!

**PRIZES:**
- 1st Place: 65-inch Hi-Def 1080P LCD TV or $5,000 Cash
- 2nd Place: Dell XPS M1730 Laptop with 17-inch Hi-Def Screen or $2,500 Cash
- 3rd Place: Dell Latitude Laptop or $1,300 Cash
- 4th – 6th Place: 16GB iPod Touch or $400 Cash

Sponsored by:

[Logo of Digi-Key]

[Logo of FTDI Chip]

For more information and to download a contest entry form, visit [www.dlpdesign.com/designcontest](http://www.dlpdesign.com/designcontest)

Brought to you by DLP Design and *Nuts & Volts* Magazine.
CircuitSpecialists.com

100's of Items Online!
1-800-528-1417
Fax: 480-464-5824
Since 1971

Stepper Motors
Stepper Motor Controllers

Part #:       Frame          Holding Torque       Price
428BYG2006  NEMA 17       2.4kg.cm/47oz.in      $15.90
57BYG4001   NEMA 23       8kg.cm/112oz.in       $21.50
57BYG4003   NEMA 23       15kg.cm/209oz.in       $25.95
57BYG4005   NEMA 23       20kg.cm/273oz.in       $28.30
8SBY1500S3-34 NEMA 34     48kg.m/665oz.in       $66.00
8SBY1500S5-32 NEMA 34     63kg.m/874oz.in       $89.00

Part #:       Dim:                  MicroStep       Price:
ACW220 100 x 61 x 19mm 128/128, 1/16, 1/32, 1/64, 1/128, 1/256  $39.95
CW230 115 x 72 x 32mm 128/128, 1/16, 1/32, 1/64, 1/128, 1/256  $49.50
CW250 149 x 94 x 42mm 128/128, 1/16, 1/32, 1/64, 1/128, 1/256  $54.90
CW860 147 x 97 x 36mm 128/128, 1/16, 1/32, 1/64, 1/128, 1/256  $96.00

CSI-2901 LEAD Free Soldering Station

Comes with all lead-free alloy solder and standard solder. Excellent thermal recovery without a large increase in tip temperature. Utilizes an integrated ceramic heater, sensor, control circuit and tip for greater efficiency, along with a highly dependable 24V output transformer. The effortless replacement of soldering tips makes for quick changes and the optional shutter-down setting turns the unit off after 30 minutes of idle time. Various tips are available at our Web Site!

- Power Consumption: 70W
- Output Voltage: 24VAC
- Temp Range: 200-480°C (362-896°F)

CSI-2901 Special Bundle Deal!!!

Item #        Only $39.00

Soldering Station w/Ceramic Element & Solder Stand

- Ceramic heating element for more accurate temp control
- Temp control knob in F(32° to 868°) & C(0° to 489°)
- 3-prong grounded power cord/static safe tip
- Soldering tip for heavy duty iron stand
- Replacable iron/easy disconnect
- Extra tips etc. can be found at our web site

Special Bundle Deal!!!
Order Item #     Item #
PB-272A        PBB-272A

PBB-272A Powered Breadboard w/ LCD Voltage Displays

Provides the user with a quick and efficient system for breadboarding electronic circuits. Comes with three regulated power supplies along with a deluxe, easy-to-use breadboard. Two LCD's conveniently show the 0-15 positive and 0-15 negative outputs.

- 5 Distribution Strip (500 tie points)
- 3 Terminal Strips (1850 tie points)
- 4 Binding Posts
- One Ground
- One 5VDC (1 AMP) Constant Power
- One 0 to +5VDC (500mA) Variable power Supply
- One 0 to +15VDC (500mA) Variable Power Supply
- 110VAC Input Power, ±10%

Item #        Only $99.00!

CSI-2901

CS11003X-S 0-30VDC/0-3A $119.00 $112.00
CS11005X-S 0-50VDC/0-3A $127.00 $119.00
CS11005X 0-30VDC/0-5A $129.00 $122.00

CSI-2205D 4,000 Count True RMS Hi-Accuracy DMM

The CS2205D Micro Control Unit auto-ranging DMM is UL listed and designed for measuring resistance, capacitance, DC & True RMS AC voltage, DC & True RMS AC current, frequency, duty cycle and temperature, along with the ability to test diodes, transistors and continuity. A special feature includes the Auto Test Lead Input Indication Technology. This will ensure that whenever you switch functions on the DMM, the proper positive port LED will light indicating where to plug the red test lead into so that no mistakes can be made by the user. It even goes so far as to give a warning tone if you plug the lead into the incorrect jack! A very helpful feature for the novice and even for the experienced user who is using the meter in less than ideal lighting conditions. Overall, we find this to be a meter that compares very favorably with much higher priced competitors on the market today!

Item #        Only $59.00!

Triple Output DC Bench Power Supplies

- Output: 0-30VDC x 2 @ 3 or 5 Amps & 1kWd output @ 5VDC/3A
- Stepped Current: 30mA + 1mA

Item #        Price 1-4        Price 5+
CS1003X-S 0-30VDC/3A $198.00 $193.00
CS1005X-S 0-30VDC/5A $259.00 $244.00

Circuit Specialists, Inc.
220 S. Country Club Dr.
Mesa, AZ 85210
Phone: 850-528-1417 / 480-464-2610
Fax: 480-446-5824

http://nutsvolts.texterity.com/nutsvolts/200901/templates/pageviewer_print?page=1&pres...
CHIPS FOR EVERY INVENTION!

**BASIC Stamp microcontrollers**

Pre-programmed with our PBASIC Interpreter, chips are available for every BASIC Stamp model along with compatible resonators and EEPROM chips.

A BASIC Stamp module is a single-board computer that runs the Parallax PBASIC language interpreter in its microcontroller. Developer’s code and data are stored in an EEPROM.

Choose PBASIC Interpreter Chips for:

- Rapid prototyping with BASIC Stamp modules
- Easy migration from your BASIC Stamp design to your custom PCB
- The option to distribute self-contained firmware updates
- Support for end-user reprogramming in an easy-to-learn high-level language

parallax.com/basicstamp

**SX microcontroller chips**

Parallax SX chips are 8-bit, high-speed microcontrollers with flash program memory and in-system debugging capability.

SX microcontrollers are an affordable solution for large-volume production. Parallax is the world-wide supplier for all models in production.

Choose the SX for your design if you need:

- Speed - operating frequencies up to 75 MHz
- A pipelined instruction set
- Jitter-free 3-cycle internal interrupt response
- Flexible I/O
- Outputs capable of sourcing/sinking 30 mA.
- A built-in 8-bit real-time clock, watchdog timer, and brown-out detector

parallax.com/sx

**Propeller microcontroller chips**

The extremely versatile Propeller microcontroller is available as a DIP chip for prototyping, and in two different package types for volume manufacturing.

With its 8 parallel-processing cogs, the Propeller chip can perform multiple functions simultaneously.

Choose the Propeller for:

- Simplified integration of multiple devices
- Reduced component count in your design for high-volume production
- Up to 160 MIPS performance (8 cogs x 20 MIPS)
- Built-in video generation
- Object-based language options: high-level Spin, low-level assembly, third-party C compiler
- Public Object Exchange for device drivers, floating-point functions, and more

parallax.com/propeller

Order 24-hours a day at www.parallax.com or call our Sales Department toll-free at 888-512-1024 (Monday - Friday, 7:00 a.m. - 5:00 p.m., PST).

BASIC Stamp is a registered trademark of Parallax Inc. Propeller, Parallax, and the Parallax logo are trademarks of Parallax Inc.