Design • Build • Program • Learn • Compete

SERVO MAGAZINE

FOR THE ROBOT INNOVATOR

www.servomagazine.com

May 2012

Meet QUADPOD

Boca Bearings 2012 Contest Calls For Innovation

✦ **Sounding Off**
Add recorded sound effects to your robot

✦ **Roll Your Own Turtlebot**
Get your own homebrew Turtlebot up and running

U.S. $5.50 CANADA $7.00

LEADERS OF THE PACK!

We've upped the charger game with the introduction of our **X4 Eighty** and **X2 400**! Leading the pack in ease of operation, powerful output wattage and the ability to charge all popular battery chemistries; these heavyweights are exactly what you have been waiting for!

X4 80 Eighty

- Four 80-watt Power Outputs, Total Output Power 320 watts
- Charges all Types of Lithium, NiCd/NiMH and Lead-Acid Based Batteries
- Lithium to 6 Cells
- NiCd/NiMH to 15 cells
- Pb to 10 cells
- Individual Balancing Ports
- Twin-Fan Cooling System
- Stock# 44169

X2 400 Four Hundred

- Two 400-watt Power Outputs, Total Output Power 800 watts
- Charges all Types of Lithium, NiCd/NiMH and Lead-Acid Based Batteries
- Lithium to 6 Cells
- NiCd/NiMH to 15 cells
- Pb to 10 cells
- Individual Balancing Ports
- Internal Cell Balancers
- Aluminum Case for Rapid Heat Dissipation
- Stock# 44170

HITEC CHARGERS–WE PACK A PUNCH

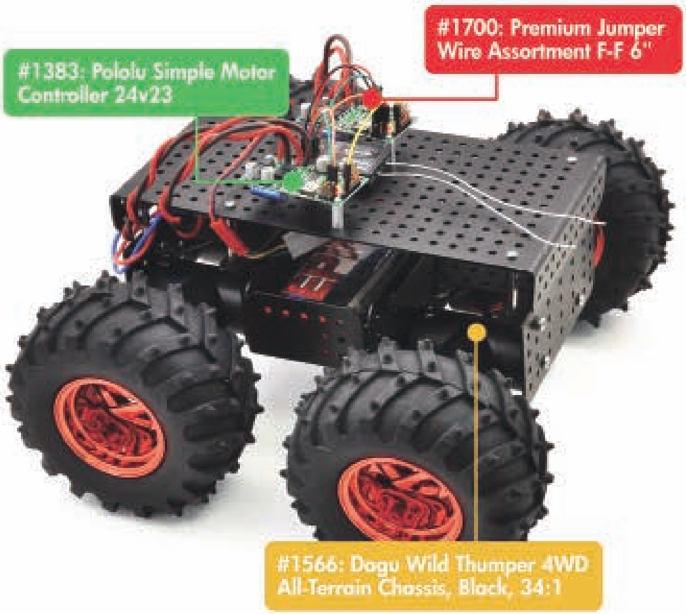
HITEC

12115 Paine Street • Poway, CA 92064 • 858-748-6948 • www.hitecrcd.com

All our chargers carry a one-year warranty against manufacturing defects from date of purchase. Customer service is available through phone support and email inquiries.

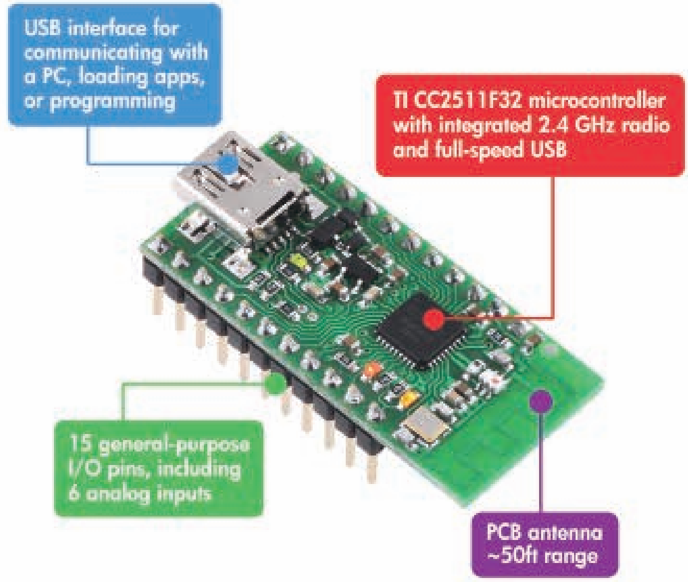
DIY Projects:

Wild Thumper-Based Robot



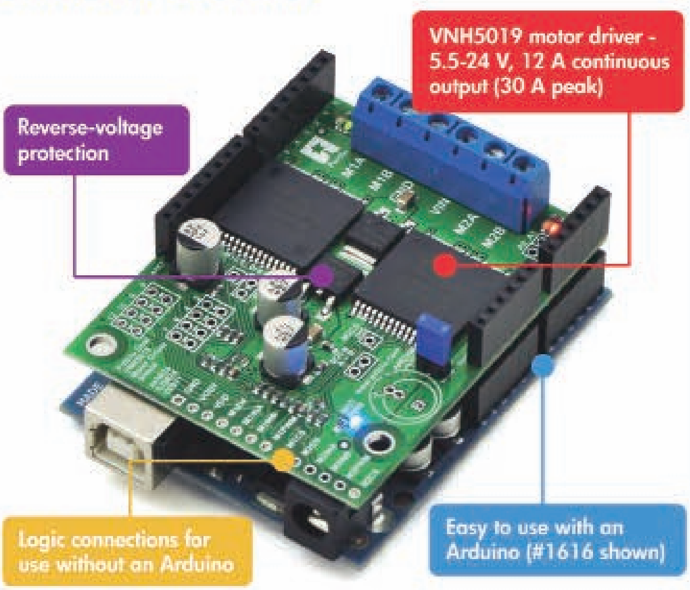
Programmable Controllers:

Item #1336: Wixel Programmable USB Wireless Module



Motor Drivers:

Item #2502: Dual VNH5019 Motor Driver Shield for Arduino



Custom Laser Cutting:

Design Your Own Chassis

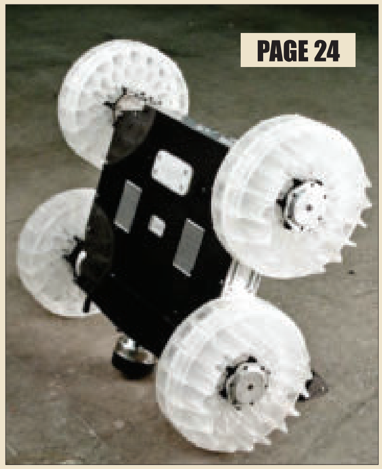
Finding the right parts for your robot can be difficult, but you also don't want to spend all your time reinventing the wheel (or motor controller). That's where we come in: Pololu has the unique products - from actuators to wireless modules - that can help you take your robot from idea to reality.

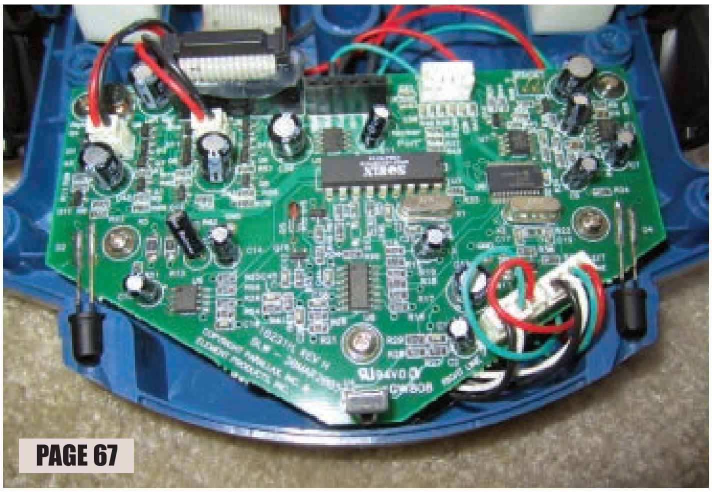
SERVO MAGAZINE

05.2012
VOL. 10 NO. 5

Columns

Departments

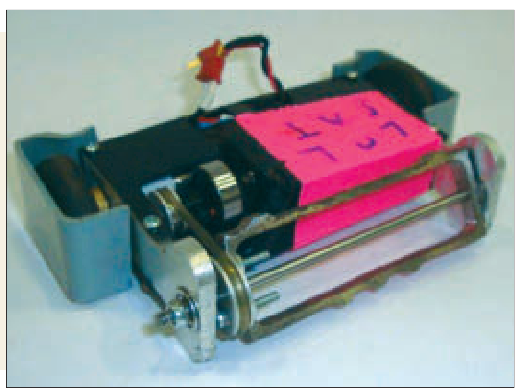

PAGE 24


PAGE 67

The Combat Zone...

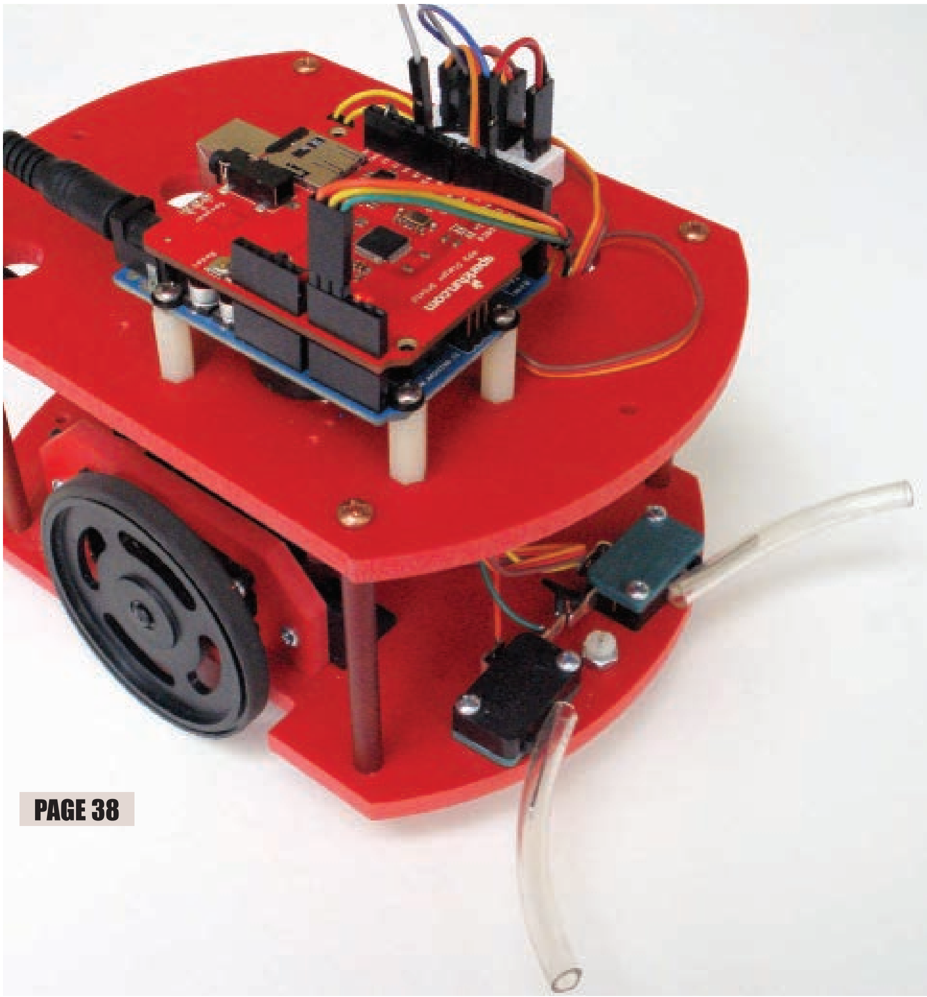
Features

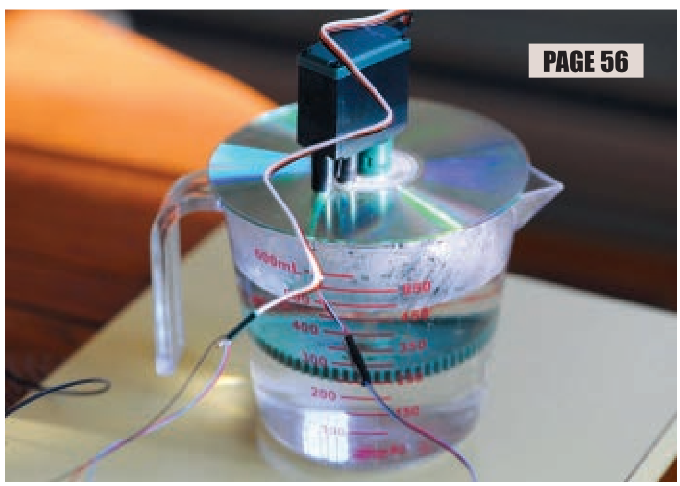
Events



SERVO Magazine **(ISSN 1546-0592/CDN Pub Agree#40702530)** is published monthly for $24.95 per year by T & L Publications, Inc., 430 Princeland Court, Corona, CA 92879. PERIODICALS POSTAGE PAID AT CORONA, CA AND AT ADDITIONAL ENTRY MAILING OFFICES. POSTMASTER: Send address changes to **SERVO Magazine, P.O. Box 15277, North Hollywood, CA 91615** or Station A, P.O. Box 54, Windsor ON N9A 6J5; cpcreturns@servomagazine.com

In This Issue ...





Mind / Iron

by Bryan Bergeron, Editor

Medicine Meets Virtual Reality

I'm just back from the annual Medicine Meets Virtual Reality conference (**www.nextmed.com**) in Newport Beach, CA. The topics discussed and demonstrated by engineers, students, military, and physicians ranged from mechatronics and sensors, to wearable electronics and biomedical simulation. There's nothing like cross-fertilization with specialists from different fields to get the ideas flowing. In addition to conferences such as MMVR, there are pockets of focused activity in academia, the government, military, and industry. For example, take a look at the US Army's Telemedicine and Advanced Technology Research Center (TATRC) site at **www.tatrc.org**. Not only is TATRC a showcase for advanced technologies — such as the use of embedded systems in healthcare — but the organization is also a source of funding and information.

There's no question that carpet rovers and crawlers are fun to design and program. However, if you want to make a career out of applied robotics, I suggest you take a hard look at the experimental and practical biomedical applications of haptics, embedded systems, and other technologies related to robotics. You'll quickly discover that all of the engineering courses in the world won't prepare you to adequately address the biological aspects of challenges in the field — and that's the real point of this editorial. If you're still in school, get some biology and anatomy under your belt. If you're in college majoring in engineering, don't think of biology as one of those electives you have to take to graduate, but attack it as part of your core curriculum. It'll pay off later. For example, let's say you want to build surgical simulators. You'll have to understand how skin and other tissues respond to cutting, tearing, and pressure. Then, there's basic anatomy. You wouldn't want to design a surgical robot that cuts right through vital nerves and arteries to get to a wound, would you? Of course, you can't know everything about biology, medicine, and robotics, but you need to be able to communicate intelligently with experts in each field.

If you've been out of school for a while and want to extend your knowledge of robotics to healthcare, options range from online courses and night classes at your local college, to internships or employment with a company that's already in that space. I like the latter two options because you can make money while you learn.

If you can't leave your current job or you don't yet have the skillset required to land a job or internship, consider online learning. I don't mean random YouTube videos, but courses in Biology, Physiology, and even Medicine from established sources. I'm a fan of Khan Academy (**www.khanacademy.org**) which is supported in part by the Gates Foundation. You can learn topics in Biology, for example, at the first year college level. Another place to start is iTunes University which hosts podcasts in Biology, Chemistry, Medicine, and related topics from Yale, Harvard, Stanford, UC Berkely, and other universities. I hope to see you presenting one day at MMVR or one of the other conferences that blend traditional robotics and engineering with medicine and biology. **SV**

BIO-FEEDBACK

Dear *SERVO*:
Very nice article on the NEATO XV-11 hack Bryan did. I found out that **protechrobotics.com** does carry all the parts for the NEATO XV-11 robotic cleaner; the LIDAR does list at $299. Maybe the prices might drop.

Fred Miller

Thanks for the kudos and the important info for other readers. It's a nice (neat!) robot. I hope the LIDAR price drops significantly, as well.

Bryan Bergeron

FOR THE ROBOT INNOVATOR
SERVO MAGAZINE

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

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

Life with Robot!
Not a dream anymore

THE MASTERPIECE
of ROBOT SERVO

24kgf.cm @ 7.4V
[333.6 ozf.in.]

Developed with the ease of use in mind, Herkulex Smart Servo Series are the best 'Green' energy efficient smart servos in the world with low voltage, high output, asynchronous bi-directional communication, and more than 50 set-up parameters.

HerkuleX

Specification (DRS-0101, DRS-0201)

- **Dimensions** : 44.5mm(W) X 24.0mm(D) X 32.0mm(H) [1.75in. X 0.94in. X 1.26in.]
- **Weight** : 45g (DRS–0101) / 60g (DRS–0201) [1.59oz (DRS–0101) / 2.12oz (DRS–0201)]
- **Input Voltage** : 7.4V DC (DRS-0101) / 7~12V DC (Optimized 7.4V) (DRS-0201)
- **Stall Torque** : 12kgf.cm@7.4V (DRS–0101) / 24kgf.cm@7.4V (DRS–0201) [166.8 ozf.in. (DRS–0101) / 333.6 ozf.in. (DRS–0201)]
- **Maximum Speed** : 0.166s/60˚@7.4V (DRS-0101) / 0.147s/60˚@7.4V (DRS-0201)
- **Operating Angle** : 320˚, Continuous Rotation
- **Communication** : Full Duplex Asynchronous Serial(TTL), Multi Drop, 0–254 ID, Maximum Baud Rate : 0.67Mbps
- **Motor** : Metal Brush DC Cored (DRS–0101) / Coreless DC (DRS-0201)
- **Gear** : Super Engineering Plastic (DRS–0101) / Reinforced Metal (DRS-0201)
- **Feedback** : Position, Speed, Temperature, Load, Voltage, etc.
- **Features** : PID, Feedforward, Trapezoidal Velocity Profile, Velocity Override, Torque Saturator & Offset, Overload Protection, Neutral Calibration, Deadband, 54 Selectable Setting Parameters (Sold Separately : HerkuleX Manager Kit)

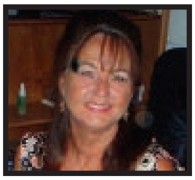
HerkuleX Manager is a bundled software that uses the GUI to maximize the ease of operation in setting up more than 50 servo operating parameters and servo maintenance using such a tool as the real time trend graph.

HOVIS Lite HOVIS Genie HOVIS Eco

Dongbu Robot

Dongbu Robot Co., Ltd. www.dongburobot.com

Headquarter : 11th Floor, Bucheon Techno Park 401, Yakdae-dong, Wonmi-gu, Bucheon-city, Gyunggi-do 420-734, Korea
TEL : +82-32-329-5551(ext.311), FAX : +82-32-329-5569, E-MAIL : robotsales@dongburobot.com
Factory : 27, 6 Gil, 4 Sandan, Jiksan-eup, Seobuk-gu, Cheonan-city, Chungcheongnam-do 331-814, Korea, TEL : +82-41-590-1700, FAX : +82-41-590-1701

Robytes

by Jeff and Jenn Eckert

Discuss this article in the *SERVO Magazine* forums at http://forum.servomagazine.com.

No Humans Required

A Learjet maneuvers into position beneath an Omega K-707 tanker aircraft.

It's no secret that the US military makes extensive use of unmanned aerial vehicles (UAVs) in a variety of shapes, sizes, and configurations. It is less widely known, however, that the Navy and Northrop Grumman have long been developing autonomous aerial refueling (AAR) technologies aimed at in-flight servicing of UAVs by other UAVs. The objective is to increase the endurance and range of carrier-based unmanned aircraft, and the focus appears to be on Northrop Grumman's X-47B — a tailless, strike-fighter-sized warbird currently undergoing flight testing. In a recent test, a research team installed X-47B hardware and software on a Learjet and paired it with an Omega K-707 tanker. Reportedly, the Learjet "successfully completed multiple air-refueling test points autonomously while commanded by a ground operator." The test was limited to verifying the AAR systems and navigation performance which is a sneaky way of saying that the K-707 didn't actually seal the deal by poking its hose into the Lear. However, "The next big step for the program is to demonstrate this capability with the unmanned X-47B and actually plug the aircraft autonomously," according to the Navy. That is scheduled to happen sometime in 2014, so stay tuned.

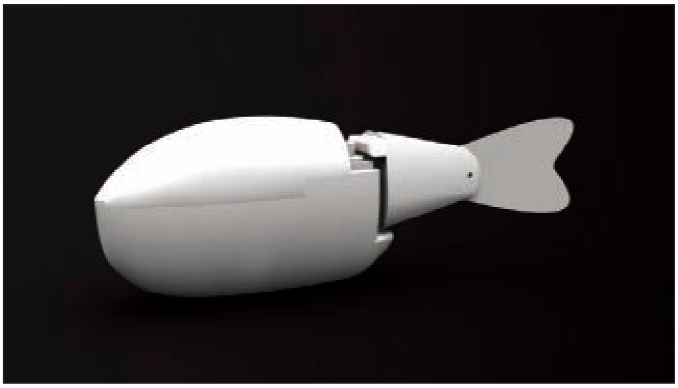
Good News on the Job Front

Things may be tough out there in the job market, but you're in luck if you have robotic skills. According to research firm Wanted Analytics (**www.wantedanalytics.com**), corporate recruiters and staffing organizations posted more than 2,100 online ads for robotic jobs in January alone, accounting for a 44 percent increase from January 2011 and more than double the number in January 2010. The highest number of offerings were in Atlanta, Boston, Detroit, Chicago, and Los Angeles, so it may be time to pack your bags.

Bioservo's Johan Ingvast demonstrates the SEM glove.

Get a Grip With Robotic Glove

The first product developed by Sweden's Bioservo Technologies (**bioservo.com**) is the Soft Extra Muscle (SEM) glove, designed to add robotic strength to hands that have been compromised by nerve injuries, weakened muscles, or other problems. The device is worn pretty much like any other glove except that the user also needs to wear a power unit on the upper arm or back. The fingertips are equipped with force sensors that register when the user grasps something, and a microcontroller decides how much extra force to add. This regulates artificial tendons within the glove. Gripping power levels can be adjusted for individual needs, and the device's lithium-ion batteries provide up to three days of service, depending on how it's used. The SEM was given the 2012 Robotdalen Innovation Award which comes in at EUR6000 (about $8,000 at current rates). If an assistive glove doesn't sound like earth-shattering news, consider that the Robotdalen jury found the device "to be an excellent example of non-invasive adaptive robot assistance under permanent and uncomplicated user control finding it roots in the simplicity of Sir Isaac Newton's discoveries." Wow. This thing is up there with universal gravitation, the laws of motion, and the reflecting telescope. It doesn't appear that you can actually purchase one yet, but it is said to be "ready for high-volume production."



NYU-Poly's biomimetic fish can lead a school of live ones.

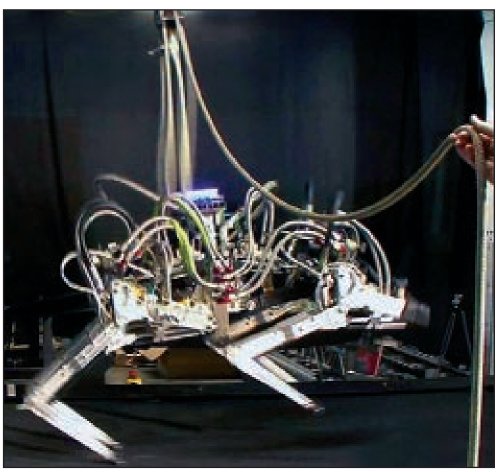
The Attraction is in the Wiggle

It seems a bit hard to believe. I can spend $19.99 for a scientifically designed and meticulously manufactured piece of bait that looks, moves, and smells exactly like a real shrimp, but somehow the fish sense that it's bogus and avoid it like the plague. However, they can't tell the difference between each other and a crude-looking robotic shiner built by researchers at the Polytechnic Institute of New York University (NYU-Poly, **www.poly.edu**)? Really? Apparently so. These folks built a bio-inspired robotic fish that mimics the tail flapping of the real thing, dropped it into the river, and tested it at varying flap frequencies and flow speeds. They discovered that the lead fish in a school wag their tails at a higher frequency than the followers and that by adjusting the robot's moves they can create a Pied Piper effect to change the school's behavior patterns. The NYU-Poly team sees this as a possible means of preserving schooling wildlife (it might work on birds, too) by leading them away from toxic chemical spills and other environmental hazards. Maybe so. It might also be a good way to coax them into my net.

Bot Drivers Licensed in NV

As of March 1st, Nevada has been offering licenses for robotic vehicles like Google's autonomous Prius which heretofore has been operating only on California highways. Last June, Gov. Sandoval signed AB511 into law, making their operation expressly legal, but it took the DMV until recently to establish pertinent rules and regulations. There probably won't be a rash of them showing up, though, as the license requires applicants to post a bond of up to $3 million per vehicle. This gets you a red license plate that marks the car as robotic and perhaps something real drivers should give a wide berth. If the car proves to be safe (which could take a few years), you'll be able to trade in the plate for a neon-green one. Similar measures are pending in Hawaii and Oklahoma, and even down here in Florida. I guess they figure it can't get much worse.

Google's robotic Prius, previously tested on CA roads.



DARPA's Cheetah, the world's fastest legged robot.

Speed Record Set

One of many projects run by the Defense Advanced Research Projects Agency (DARPA, **www.darpa.mil**) is the Maximum Mobility and Manipulation (M3) program which seeks to create engineering advances in robot mobility and manipulation capabilities so as to more effectively assist warfighters and other DoD personnel in a range of missions. A recent milestone in the M3 program is the setting of a new world's speed record for a legged robot. A top speed of 18 mph was recently achieved by the Cheetah bot — developed by Boston Dynamics (**www.bostondynamics.com**), which is also purveyor of the better-known BigDog and Squishbot machines. This breaks the previous record of 13.1 mph, set back in 1989. The robot increases its stride and running speed by flexing and unflexing its back on each step, much like a real cheetah. Because the current version is powered by an external hydraulic pump, it must employ a boom-like device to keep it in the center of a treadmill. However, tests of a free-running prototype are scheduled for later this year. To see it in action, visit **http://youtube.com/watch?v=d2D71CveQwo**. **SV**

GEERHEAD

by David Geer

Contact the author at geercom@windstream.net

Discuss this article in the SERVO Magazine forums at http://forum.servomagazine.com

Boca Bearing's Robot Contestants — Part 1

Boca Bearing — a maker of miniature bearings for industry, hobby, and recreation — is holding an on-going contest through 2012 with monthly winners to celebrate their 25 years in business. Contestants building and entering innovative mechanical projects including robots will vie for a piece of over $20,000 in cash and prizes. Participants enter videos of their robotic (or mechanical) creations that use "ball bearings, roller bearings, linear bearings, or any form of full ceramic or ceramic hybrid bearings."

Monthly winners will receive an iPad 2. Boca Bearing will ultimately select a Grand Prize winner and two runners-up from the monthly winners. Runners-up will each take home a 3D printer from Makerbot Industries. The Grand Prize winner will take home $10,000 in cash. "The Boca Bearing Company believes in supporting individuals and companies who focus on art, science, technology, engineering, and math. These are the creative people who push the limits of new technology and will drive our economy." You can get more details about the contest at www.bocabearings.com/innovation-contest/default.aspx.

Let's meet a couple of the entrant's creations ...

QuadPod in hand.

The Snellfight QuadPod Miniature Robotic Copter

Phil Jermyn from England submitted the Snellflight QuadPod miniature robotic copter. Snellflight intends for the four-rotor copter to be a hobby flight device for now. The copter uses small, brushless motors and is a low cost mechanism.

Jermyn kept the copter low cost by employing three common methods for reducing production cost: the design, parts sourcing, and manufacturing. "Our extensive experience in the toy business gave us an edge when choosing parts to design into the product," explains Jermyn. Jermyn and Snellflight understand the features and limits of various cheaper parts. This enables the company to build in the capabilities the design requires while keeping the cost as low as possible. Snellflight also keeps the manufacturing as labor-free as possible, allowing minimal overhead in tooling.

Another challenge (in addition to keeping the copter low cost) was keeping it light enough to fly and carry a payload. By using lithium rechargeable batteries and brushless motors, Jermyn and Snellflight are able to keep the copter light in weight. "This combination has increased available power-to-weight ratios by a factor of 10," says Jermyn. These technologies — which are critical to load carrying electric flight — make it possible for this copter to carry a small video camera payload.

With the revolution in electronic imaging, Snellflight was able to easily find a small, light video camera that could take pictures in-flight. Although the company has not integrated a camera into the copter as yet, Jermyn says they plan to do so in the future.

The QuadPod works with almost any commercial R/C system on any available frequency. "This includes the 35/72 MHz systems such as Spektrum. We do not, however, have our own proprietary wireless system," Jermyn explains. Snellflight wrote the QuadPod's control software in C. It runs on a 40 MHz PIC microprocessor.

The QuadPod's flight is limited to the orientation and altitude that keep it within sight of its operator. "This effectively restricts it to around 300 feet. However, we plan to introduce an automatic leveling system that will enable it to hover by itself and fly greater distances," says Jermyn.

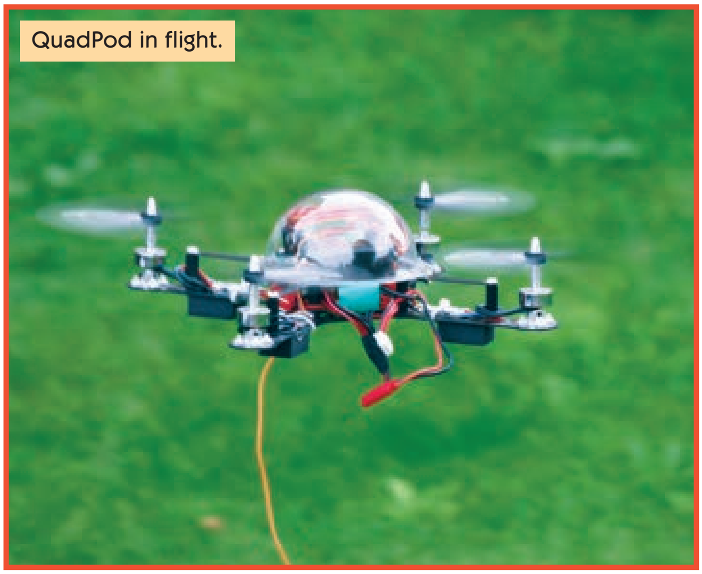
Users can upgrade the QuadPod in a number of different ways. They can reprogram it via a special connection that enables users to update the software and add new features as they come online. The main control circuit has two digital communications ports for adding equipment. These add-ons include a USB computer dongle for parameter adjustment using a computer software suite (available soon), and add-on board sensors and circuitry to enable features such as self-leveling and an autopilot capability. The design of the QuadPod airframe is simple and elegant, allowing extra features such as the necessary camera mounts for the video camera and a taller undercarriage or additional bodywork.

The company will add the autopilot capability in an ongoing fashion by adding auto leveling first. Snellflight will then add altitude control and an electronic compass which will enable the craft to fly itself. "We may add GPS navigation to allow the aircraft to fly pre-programmed flights," commented Jermyn.

Mountainboard Controlled by Wireless Glove

Andres Guzman-Ballen submitted a powered Mountainboard skateboard controlled by a wireless glove. This was his first real hands-on project. "The project is aesthetically pleasing and gives me the ability to travel around with ease," explained Guzman-Ballen.

With some help from his university's Electrical and Computer Engineering department, Andres built out the



QuadPod in flight.

Close-up of the wireless glove on the Mountainboard.

Full view of the wireless glove and Mountainboard.

casing for the board's battery and the housing for the motor. "I dealt with everything else including building the electronics for the board and the wireless glove, and soldering the wires together that connect the battery with the motor controller and the controller with the motor. I also handled the debugging," says Andres.

To debug the electronics, Andres built a simple wired controller using parts from the IEEE lab including resistors, LEDs, and potentiometers in order to interface with the motor. In one instance during debugging, the wheel would spin unpredictably. Andres could not figure out why. "I finally resorted to using an oscilloscope to try and figure out what the issue was. It turned out that my poor soldering work was the reason for the faulty connection," commented Andres.

After he re-soldered it, the board worked perfectly. Other troubleshooting issues included screws that came loose and wires that slid out of their ports due to the bumpy terrain the Mountainboard traveled over. Andres fixed those with hot glue.

The wireless glove control device presented its own challenges. "I had an issue trying to utilize SparkFun's flex sensor because I couldn't find a way to bend my fingers with the sensors without them moving out of place with respect to my fingers," Andres continued. He solved that one by attaching the sensor between the middle and ring fingers on the glove to maintain access to it. This way, he can hold and bend the tip of the flex sensor with the tips of both fingers and not worry about anything moving or blocking the sensor.

Getting the wireless transceivers to communicate was another issue. When Andres soldered the project together and saw that it was not functioning, he wasn't sure whether it was the code in the microcontroller or that he had hit the delicate receiver at some point with more than the 3.3V allowable and damaged it. To troubleshoot, Andres went to a basic control setup in which he programmed one microcontroller to tell the other to blink an LED. It lit up, so the problem was in the code. "It turned out to be a missing semicolon that I thought the Arduino compiler would have caught," says Andres.

Wireless glove and Mountainboard creator, Andres Guzman-Ballen.

Once Andres fixed the code, he integrated the glove with the board. While he had issues configuring the thresholds in the code at first and had to analyze the serial output and mess with the code, he was finally able to fine-tune the glove and get the board's behavior where he wanted it.

The Arduino microcontroller on the glove first checks to see whether the deadman switch is held on, which is an indicator that the rider does want to go the given speed. When the deadman switch is not held, the board does not speed up. The microcontroller then looks at a variable voltage coming from

SparkFun's flex sensor — a flexible potentiometer that changes resistance when the operator bends it — converts the analog signal to digital, and sends the data to the board.

"I also included a semaphore to make sure communication between the board and glove is bi-directional and that the board doesn't act unless it knows that the glove received the information," Andres explained. The board waits until the glove acknowledges receipt of the signal from the board. If the board doesn't get that message, it shuts off because it knows that something is wrong with the glove and that the data it contains is not up to date.

Once completed, the wireless glove and board pairing enable a smooth ride around campus and cruising over rough terrains without stopping. Andres had made some calculations about the correct gear ratios and the right Kv rating for the brushless motors to make sure he wouldn't fly off the board when he accelerated.

"To figure out how fast I wanted to accelerate on the board, I had to be creative. One night, I went to an empty parking lot and stuck 10 pieces of tape on the asphalt 10 feet away from each other and pushed myself as fast as I could go and recorded the times when I passed each marker with my watch," Andres revealed. Using Newtonian physics, he figured out how fast he was accelerating and that gave him a rough estimate of how fast he wanted the electric Mountainboard to accelerate.

Andres is happy with the smooth rides he can achieve on the Mountainboard. "It is built with a flexible and strong fiberglass deck with a wooden core. The slanted trucks give you a lower center of gravity, allowing you to be more balanced," concluded Andres.

Resources

Boca Bearing Contest Page
www.bocabearings.com/innovation-contest/default.aspx

Links to Contestants
www.bocabearings.com/innovation-contest/Contestants.aspx

Boca Bearing Site
www.bocabearings.com

That's a Wrap

Next month, we'll feature a robot band entry from the Boca Bearing contest. SV

Discuss this article in the SERVO Magazine forums at http://forum.servomagazine.com

Our resident expert on all things robotic is merely an email away.
roboto@servomagazine.com

Tap into the sum of *all human knowledge* and get your questions answered here! From software algorithms to material selection, Mr. Roboto strives to meet you where you are — and what more would you expect from a complex service droid?

by
Dennis Clark

ASK MR. ROBOTO

Not a lot of questions have been coming my way this last month, so either everyone is on holiday or everyone is busy. The season for robot competitions is upon us, so I assume the latter. A month ago, a reader asked a seemingly simple question and I assumed it would be simple to answer. Not so. Although no one needs to prove this, it proved to me that there is always a LOT more out there to learn! Look at the question about doing straight, non-Arduino programming on the Digilent chipKIT MAX32 board if you are curious about this discussion.

With the slew of YouTube videos about walking (and running!) robots, I'm starting to look more at my bipeds and other legged robots I haven't touched in a while and am getting inspired! How about you?

Last month, a reader asked how to program a chipKIT MAX32 board directly in MPLAB since he wanted direct access via C instead of Arduino. I did a little hardware interface recon using Arduino last month to dust off the hardware. This month, we're going to dive in with some introductory material.

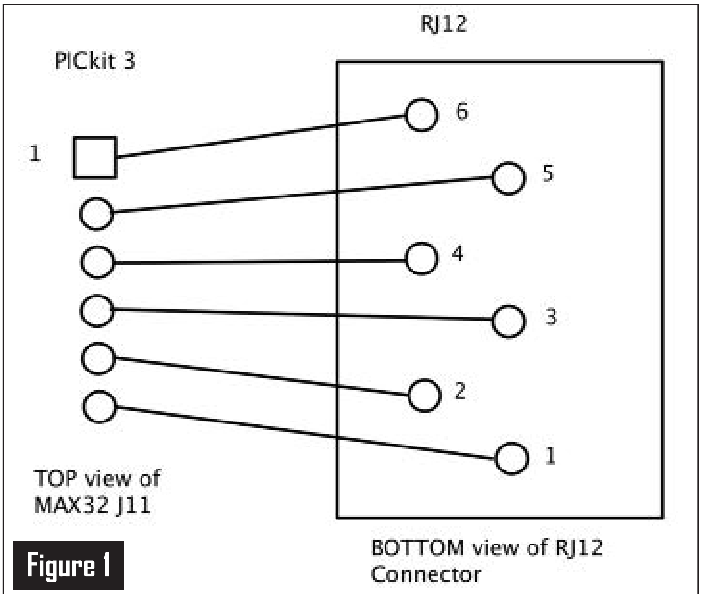


Figure 1

Q. (Reprise) I just wanted to reply to a previous issue when you talked about the chipKIT. I have been reading up a good deal about these, and I really want to explore the chipKIT MAX32. I was wondering if you could do a more in-depth write-up on this. I would love to see more of the capabilities of this guy. I know Fred Eady has done some articles on it, but his are more advanced topics; and I still consider myself a beginner/novice in the electronics world. Learning about CAN is still well above my head. Now I'm not saying I want to see a "Hello World" program, but I would like to see some of the more basic stuff such as utilizing the ADC with sensors, basic timer/interrupts, and the like. Also, I would like to see it interfaced with MPLAB utilizing C, instead of the Arduino processing structure. If I want help with that, I can just go to the Arduino site. I'm more interested in using this as a development board rather than a small projects board. Thanks!

— Corey Hastings

A. Corey, here is the chipKIT MAX32 introductory programming, *part deux* ... There is a time-honored method for learning a new hardware platform or technology base: Copy someone else's code and hack on it! I tried looking for just this thing for the MAX32, and pretty much fell flat. The 'net is just not filled with examples, and certainly not filled with examples for the chipKIT MAX32 when dealing with Microchip's C32 compiler and MPLAB. I also found that the documentation for the peripheral libraries and such for the PIC32 are a little bit in their early stages. I had to look directly into ".h" files to find out how to use some functions or find out what they did. (Oh, the thrills of discovery!)

Regardless, the C32 compiler libraries do a lot to help us with the quite complex internals of the PIC32 line. Most of us don't want to deal with figuring out cache configurations or similar esoteric subjects about configuring the peripherals to do common tasks. C32 compiler to the rescue! There are a number of functions and macros that appear to do a good job dealing with this, so we can move on and do what we want to do!

In this column, I'll show you how to hook up MPLAB directly to the MAX32 for writing code. I'm going to blow away the bootloader during the course of these exercises, but worry not! Digilent provides a .hex file we can use to put it back when we are done playing. After this article, I'm going to look into creating programs in MPLAB that can be directly downloaded to the MAX32 through the Arduino bootloader which will allow the best of both worlds!

Figure 2

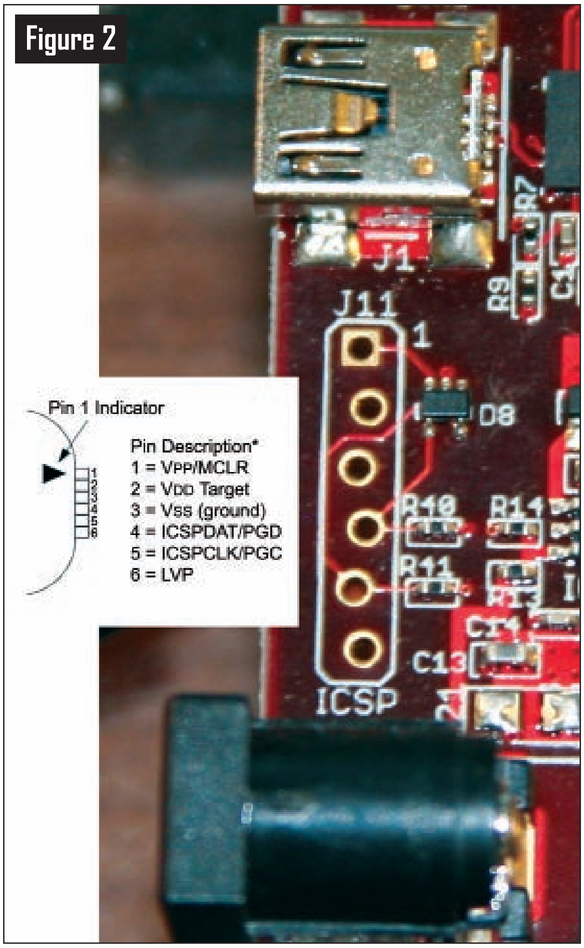
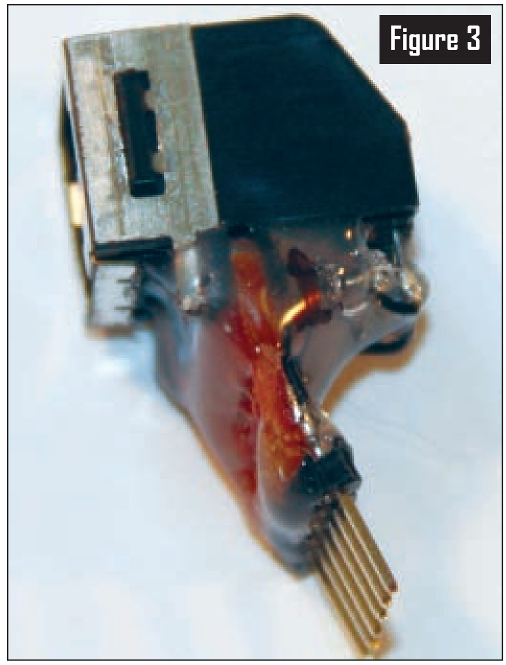


Figure 3



For now, we'll just concentrate on programming the MAX32 board from within MPLAB. The good thing about direct programming is that it will allow us to use a hardware In-Circuit-Debugger (ICD) to look at what is happening when the program doesn't do what we think it should be. I have a Microchip ICD 3 "hocky puck" for my ICD; there are many others available out there that will also work.

Connecting to MPLAB

The MAX32 has J11 (next to the power jack) which is in the PICkit 3 programmer/debugger pinout. The pins are staggered so that a "Berg strip" connector can be plugged into it without soldering, and the PICkit 3 can be attached. As I said, I use an ICD 3 for MPLAB work, so I needed a way to convert the RJ12 connector to the in-line pattern. It is easy to do; I'll show you how. This converter will probably have LOTS of uses for you if you stay with the chipKIT line. I really like their boards and their prices, so I too will get some mileage out of it!

This is a very simple job; just solder six wires from an RJ12 to a 0.1 inch six-pin header. **Figure 1** shows how to handle this. You need to be careful about your order. If you do what I did and confuse the ICD side pinout with the target side pinout, it won't work as you expected. You'll have to flip the converter around. So, I guess that the message is "don't panic." It'll work anyway, just flip it! The reason it is so simple to get this wrong is that the cable is straight through; pin 1 at one end will be pin 6 at the other if you look at the RJ12 in the same way (like I did). My drawing in **Figure 1** will hopefully keep you from making this same mistake.

NOTE: Pin 1 on the RJ12 connector is pin 6 on the MAX32 board! I drew the PICkit 3 in-line as a TOP view of the MAX32 board, but because my ability to draw in 3D isn't all that great, I drew the pins on the RJ12 from the BOTTOM view. (Really, it'll be easier to understand that way.) **Figure 2** shows the hole pattern with pin 1 labeled on the board, and the pinout of the PICkit 3 and what the pins are. **Figure 3** shows what mine looks like. I hot-glued the two connectors to a piece of proto-board after soldering the wires. This makes a VERY tough little connector jack.

"Hello World" for Embedded Systems

You know what this is, I'm sure: blinking an LED. On the MAX32, that is LED4 which (on the MAX32 schematic) is located here: **www.digilentinc.com/iProducts/Detail.cfm?NavPath=2,892,894&Prod=CHIPKIT-MAX32**.

The LED4 is on RA3 (port A, bit 3). The MAX32 uses the PIC32MX795F512L part; this is a 100-pin part with six ports. Each port can have up to 16 pins on it. Most of these pins can have multiple identities, as either analog or digital, SPI, I^{2}C, PWM, USART, etc. Fortunately,

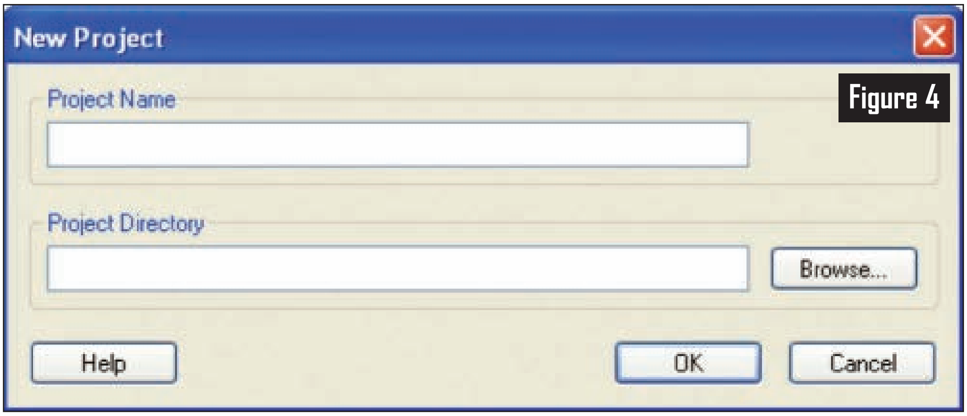


Figure 4

C32 has a peripheral library that makes configuring and using these ports easier to handle. This is great for us hobbyists because the PIC32 uses the MIPS core which includes interesting things like multiple device busses, "look ahead" caches, out-of-order instruction handling, and more stuff that can make the chip difficult to understand and configure. I'll show you how to use those libraries and macros built into the C32 compiler to make our lives easier.

Install and Use the IDE and Compiler

The first step down the code path is creating a project. You will need to install MPLAB and the C32 compiler first. Here is where you will find them:

MPLAB 8.84 IDE — **www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en019469&redirects=mplab**.

Get *MPLAB IDE v8.84.* This is a zip file. Install this first.

Microchip C32 compiler — **www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en534868&page=wwwCompilers**.

Get the C32 "Standard Edition." After 90 days, the advanced optimizations will time-bomb, but the compiler will still work fine. Get the 2.02a installed and the PDF compiler Users Guide and C Libraries manual. The Peripheral Library manual is more hidden; get it here: **www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=2680&dDocName=en554265**.

This link is on the C32 compiler page, but is in a subtle location.

Now, install the C32 compiler. Let it go to its default location.

You are all set to write code! There is more than one way to get anywhere. The same goes with writing a project for a PIC using MPLAB. You can either use the Project Wizard (Project->Project Wizard) to create your project or you can create it manually. The Wizard guides you through the selection and setup process, so I won't go through that here. I find it is just as easy to do it myself with a few steps.

Step 1: Create a folder where you want your project to live. Do this in Windows, not the IDE.

Step 2: Run MPLAB and use Project->New Project. Give the project a name, and use the *Browse* button to navigate to the project directory that you created in Step 1 (see **Figure 4**).

Step 3: Select your processor: Configure->Select Device ... You will be using the PIC32MX795F512L for the MAX32 board.

Step 4: Open the *Project* and *Output* views from the View menu.

Step 5: Create or add your files. If you already had files you wanted to use, you can put them in the folder you created for your project. Right-click on the *Source Files* folder in the Project view under your <name>.mcp heading and select *add files.* Alternatively, you can just start creating a new file, save it to your folder, and add it as above. In **Figure 5**, I have an example layout with the *Hello World* example set up.

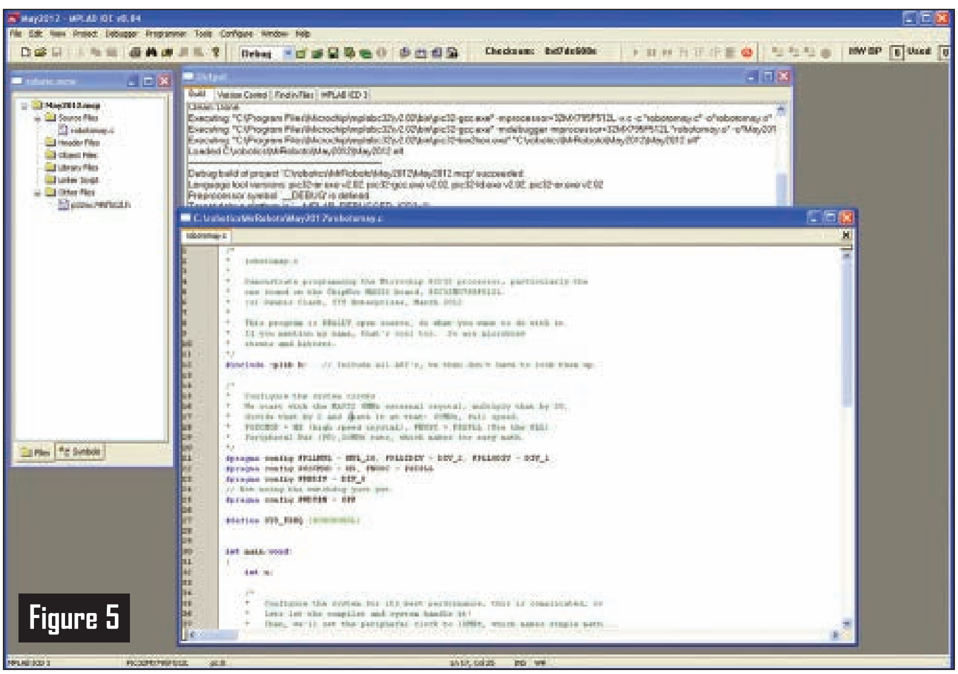


Figure 5

Write, Compile, and Download Your Program

Now, we're finally to the fun

stuff: code writing. This first program is "Hello World" for embedded programmers. We flash an LED. As you can see, while the PIC32 seems daunting to learn from its datasheets, the build environment does not have to be intimidating. **Listing 1** shows the whole thing with comments.

This looks pretty simple, right? The magic lies in the *pragma* macros and the peripheral library functions and macros. The *pragmas* at the beginning of the program tell the chip how to configure the incredibly versatile oscillator clocks on the PIC32. With the combinations of pre-scalers, post-scalers, and PLL multipliers, we can create many different frequencies. This allows us to turn the peripheral bus clock for SPI, I²C, USART, and the like to get as close as possible to their maximum values (or best divider values in the case of the USART). The *SystemConfig Performance()* function has the system tune the caches, technomumble, and peripheral bus to optimum levels without us having to understand how to do that. However, sometimes we DO want to set the peripheral bus to a specific speed, so the *mOSCSetPBDIV()* function allows us to do that. The *system.h* header file buried deep within the C32 compiler installation tells us exactly what is happening. If you are interested, it is here: C:\Program Files\Microchip\mplabc32\v2.02\pic32mx\include\peripheral.

Listing 1: "Hello World."

```
/*
*      robotomay1.c
*
*      Demonstrate programming the Microchip PIC32 processor, particularly the
*      one found on the ChipKit MAX32 board, PIC32MX795F512L.
*      (c) Dennis Clark, TTT Enterprises, March 2012
*
*      This program is REALLY open source, do what you want to do with it.
*      If you mention my name, that's cool too.  So are microbrew
*      stouts and bitters.
*/
#include <plib.h>    //Includes all API's
/*
*      Configure the system clocks
*      Start with the MAX32 8MHz external crystal, multiply that by 20,
*      divide that by 2 and leave it at that: 80MHz, full speed.
*      POSCMOD = HS (high speed crystal), FNOSC = PRIPLL (Use the PLL)
*      Peripheral Bus (PB),10MHz rate, which makes for easy math.
*/
#pragma config FPLLMUL = MUL_20, FPLLIDIV = DIV_2, FPLLODIV = DIV_1
#pragma config POSCMOD = HS, FNOSC = PRIPLL
#pragma config FPBDIV = DIV_8
// Not using the watchdog just yet.
#pragma config FWDTEN = OFF

#define SYS_FREQ (80000000L)


int main(void)
{
       int n;
       /*
       *      Configure the system for best performance, complex, so
       *      lets let the compiler and system handle it!
       *      Set the peripheral clock to 10MHz to make math simple...
       */
       SYSTEMConfigPerformance(SYS_FREQ);
       mOSCSetPBDIV(OSC_PB_DIV_8);

       /*
       *      Start with basics, set up a digital pin to blink an LED.
       *      This is the PORTA, RA3 with the MAX32 LED 4 on it.
       */
       PORTSetPinsDigitalOut(IOPORT_A,BIT_3);
       while (1) {
      // PIC32 has an instruction that toggles an I/O
              mPORTAToggleBits(BIT_3);
              for (n=0;n<65535*50;n++) {
                     ;
              }
       }
}
```

Because some I/O lines can be analog or digital, we can either hunt down the exact registers to set to turn off analog lines and set the digital pin to be an input or an output, or we can use functions from Section 10 of the *32bitPeripheralLibraryGuide* to handle the details. This function checks and sets things up properly while turning off functionality that we don't want. This does add some code overhead from doing it manually ourselves, but this PIC32 has 512 KB Flash and 128 KB of RAM. Compared to the programming of a typical small embedded part with 16K Flash and 2K of RAM, I feel like the MAX32 is a "big iron" computer! In other words, we have the Flash to burn right now. I like how clean code looks when we have abstracted the hardware somewhat, and simple function calls can deal with the details intelligently.

When you compile the program Project->Build All (or the funky icon in the middle of the icon bar that looks like an old rolodex card) and download it to the MAX32 using the first icon in the far right icon grouping (do mouse-overs until you find the right one), you will see LED4 blink at about a 1 Hz rate. I just guessed on the timing of that really big for{ } loop and got lucky.

Listing 2: Background Timer.

```
/*
*       robotomay2.c
*
*       Demonstrate programming the Microchip PIC32 processor, particularly the
*       one found on the ChipKit MAX32 board, PIC32MX795F512L.
*       (c) Dennis Clark, TTT Enterprises, March 2012
*
*       This program is REALLY open source, do what you want to do with it.
*       If you mention my name, that's cool too.  So are microbrew
*       stouts and bitters.
*/
#include <plib.h>    // Include all API's, so we don't have to look them up.

/*
*       Configure the system clocks
*       We start with the MAX32 8MHz external crystal, multiply that by 20,
*       divide that by 2 and leave it at that: 80MHz, full speed.
*       POSCMOD = HS (high speed crystal), FNOSC = PRIPLL (Use the PLL)
*       Peripheral Bus (PB),10MHz rate, which makes for easy math.
*/
#pragma config FPLLMUL = MUL_20, FPLLIDIV = DIV_2, FPLLODIV = DIV_1
#pragma config POSCMOD = HS, FNOSC = PRIPLL
#pragma config FPBDIV = DIV_8
// Not using the watchdog just yet.
#pragma config FWDTEN = OFF

#define SYS_FREQ (80000000L)

volatile long t_1ms = 0;

void __ISR(_TIMER_1_VECTOR, ipl2) Timer1Handler(void)
{
       // clear the interrupt flag
       mT1ClearIntFlag();
t_1ms++;
}

int main(void)
{
       int n;
       long delay;

       /*
       *      Configure the system for its best performance, complex, so
       *      lets let the compiler and system handle it!
       *      Then, set the peripheral clock to 10MHz, for simple math...
       */
       SYSTEMConfigPerformance(SYS_FREQ);
       mOSCSetPBDIV(OSC_PB_DIV_8);

       /*
       *      Configure Timer 1 to interrupt every 1ms to increment
       *      a background ticker that we'll call t_1ms.  We'll give
       *      Timer1 a priority 2 setting, which is a step above
       *      the lowest interrupt priority.
       *      We'll use vectored interrupts.
       */
       OpenTimer1(T1_ON | T1_SOURCE_INT | T1_PS_1_1, 10000);
      ConfigIntTimer1(T1_INT_ON | T1_INT_PRIOR_2);
      INTEnableSystemMultiVectoredInt();

       /*
       *      Start with basics, set up a digital pin to blink an LED.
       *      This is the PORTA, RA3 with the MAX32 LED on it.
       */
       PORTSetPinsDigitalOut(IOPORT_A,BIT_3);
       while (1) {
              // PIC32 has an instruction that toggles an I/O
              mPORTAToggleBits(BIT_3);
              // delay for half as second
              delay = t_1ms + 500;
              // wait for it...
              while (t_1ms < delay);
       }
}
```

Clearly, the next thing we need to do is use a timer to create a background ticker that we can use to time things. That is typically one of the first things that I do in an embedded project so that I can have a timer that is always running. I typically make it 1 ms or 10 ms so I have some resolution in my timing. Our next program will add a timer and an interrupt so that we can keep a background ticker going at 1 ms.

In **Listing 2**, I have a program that sets up Timer 1 as a background ticker to create a 1 ms resolution background clock that we can time other events with. Often in embedded programming (which includes our beloved robots), we want to do something for a bit, then do something else, or gate the action of a detection for a while before we look at the sensor again, or some other thing. We do these things in threads or in state machines. In either case, you need a timer running automatically in the background to keep track of the passing of time. This next program does that.

There are some new concepts here, and one we've discussed already. I'll touch on the latter first. I am using function calls and macros from the 32-bit peripheral library to handle the heavy lifting of setting all of the myriad configuration registers required by the timer and the interrupt vector and priority setup. For you "purists" out there, I (for one) believe that if someone has already done the work, I'll leverage it. I don't need to re-invent the wheel.

The PIC32 can handle interrupts in two ways. One way (the default after a reset) is to have all interrupts vector to a single address and sort it out there. This is like the old PIC16F line. While this works, it means your interrupt routine will get complex and cumbersome if you are using a lot of interrupt sources in your program. The second one gives every interrupt source its own vector (specific address location). This allows you to have a separate function for every interrupt, which means that you already know what is going on when the code gets there. Also, the vectored interrupt allows the use of a set of *shadow registers* for use by the ISRs (interrupt service routines) so that you don't have to do a lot of context saving if your ISR does a bunch of stuff. This can really save time.

My code above does little and it does that to a *volatile* variable that is never written to outside of the ISR. The ISR needs only minimal overhead to handle simple addition. My read of the assembly instructions generated in the pre-amble and post-amble showed what I expected. The *volatile* keyword on the t_1ms variable means that the compiler must generate code to read the value whenever it is referenced because it is changed outside of the context of the program (in the ISR), so it can't assume that it is the same value as the last time that it read it. I do not yet fully grok the nuances to the PIC32 interrupt system, but I will in time. This code works as intended using the peripheral library functions.

That's a Wrap

Whew! Next month, I'll add to our knowledge by getting code that will read a Sharp IR rangefinder which requires using ADC (Analog-to-Digital Conversion), and getting range information from an HC-SR04 SONAR device which will require using Input Capture hardware to measure pulse widths. What fun!

Thanks for this question, Corey, I've been learning a lot; I hope you do too. When I get a PICkit 3, I'll discuss any differences it might have over the ICD 3, and finally, I plan to get together the custom linker file that will allow us to create MPLAB projects whose hex files can be downloaded via the Arduino bootloader through avrdude. That last thing sounds SO weird — to load a file into a PIC using an AVR programmer program and protocol! Life is a journey, for certain.

Until next month, please let me know what is on your mind and I'll do my best to help you with it. You can contact me at roboto@servomagazine.com to ask me any question about robotics that you have. **SV**

EVENTS Calendar

ROBOTS.NET

Send updates, new listings, corrections, complaints, and suggestions to: steve@ncc.com or FAX 972-404-0269

Know of any robot competitions I've missed? Is your local school or robot group planning a contest? Send an email to steve@ncc.com and tell me about it. Be sure to include the date and location of your contest. If you have a website with contest info, send along the URL as well, so we can tell everyone else about it.

For last-minute updates and changes, you can always find the most recent version of the Robot Competition FAQ at Robots.net: **http://robots.net/rcfaq.html**.

— R. Steven Rainwater

MAY

3 **Robots Intellect**
Kaunas, Lithuania
Autonomous robots must fetch a 1 kg bag of "gold."
www.robotsintellect.lt

4 **SPURT**
University of Rostock, Rostock, Germany
Autonomous robots must race around the SPURT track.
http://spurt.uni-rostock.de/

5 **RoboRave**
Albuquerque, NM
Events include fire fighting and line following for autonomous robots.
www.roborave.org

12 **Baltic Robot Sumo**
Riga, Latvia
Autonomous bots compete in Mini Sumo for a traveling cup award.
www.balticrobotsumo.org

12 **DPRG Roborama**
Museum of Nature & Science, Dallas, TX
Events include line following, Square Dance, Table Top Challenge, and RoboColumbus — all for autonomous robots.
www.dprg.org/competitions

12 **Western Canadian Robot Games**
Calgary, Alberta, Canada
Lots of events including Sumo, Mini Sumo, LEGO Sumo, Art Bots, line following, high speed line following, Minesweeper, LEGO Mindstorms Landslide!, as well as humanoid and walker challenges.
www.robotgames.com

14-18 **ICRA Robot Challenge**
St. Paul, MN
A challenging event for robots that includes the Micro-robot Challenge, the Virtual Manufacturing Challenge, the Solutions in Perception Challenge, and the Modular and Reconfigurable Robot Challenge.
www.icra2012.org

17-20 **Eurobot**
France
This year's event is called Treasure Island. The details weren't available at press time but should be released on the website soon.
www.eurobot.org

19 **RoboFest World Championship**
Lawrence Technological University
Southfield, MI
Events include Game Competition, Robot Exhibition, RoboFashion Show, Mini Urban Challenge, fire fighting, and VEX challenges.
http://robofest.net

25 **NATCAR**
UC Davis Activities and Recreation Center
Davis, CA
High speed line following autonomous race cars race each other.
www.ece.ucdavis.edu/natcar

30 **NASA RASCAL Robo-Ops**
Johnson Space Center, Houston, TX
Competition of teleoperated planetary rovers built by university teams.
www.nianet.org/RoboOps-2012.aspx

31 **ION Autonomous Lawnmower Competition**
Beavercreek, OH
Runs through June 2. Autonomous lawn mowing robots compete to find out which can most accurately and quickly mow a field of grass.
www.automow.com

JUNE

2 **Los Angeles/Orange County Robot Challenge**
Santa Ana High School, Santa Ana, CA
A student robot competition for small (20 x 15 cm) autonomous robots.
www.sahsrobotics.org

2 **SRS Robothon**
Center House, Seattle Center, Seattle, WA
This year's competition includes line following, line maze, Mini Sumo, and SRS Robo-Magellan.
www.robothon.org

8-11 **AUVS International Ground Robotics Competition**
Oakland University, Rochester, MI
Autonomous ground robots must navigate an outdoor obstacle course within a prescribed time while staying within a 5 mph speed limit.
www.igvc.org

16 **SparkFun Autonomous Vehicle Competition**
SparkFun parking lot, Boulder, CO
Autonomous robots must circumnavigate around the SparkFun building by either ground or air.
www.sparkfun.com

2012 DIGILENT DESIGN CONTEST

The 8th annual US regional Digilent Design Contest will be held on May 6-7, 2012 in conjunction with the IEEE EIT conference in Indianapolis, IN. For further information, go to **www.digilentinc.com/events/ddc2012/** or **www.facebook.com/Digilent?sk=app_244940582228984**.

NEW PRODUCTS

New Joystick Modules

ServoCity now offers a new line of joystick modules. Six different module styles are available, ranging from a simple two function joystick to a compact four function joystick. The X and Y (and optional Z) axis offer 50 degrees of total movement; the optional fourth axis is a momentary button. Each axis is coupled with a 5K potentiometer. Internal springs center the joystick back to neutral (center). All incorporate a rubber dust seal and a mounting ring with hardware; the joysticks start at $19.99.

New Mega Servos

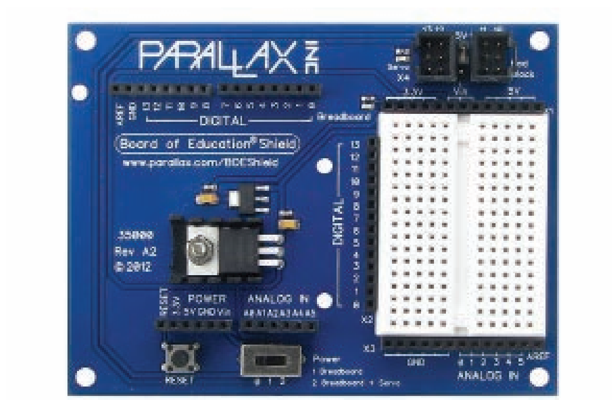
ServoCity is also introducing their new line of RobotZone Digital Mega Servos. The MS530-2 Mega Servo is able

to deliver up to 6,000 oz-in (432 kg-cm) of torque with a transit time of 0.9 seconds/60 degrees (on 7.4 VDC). The Mega Servos plug into any servo controller or receiver just like a regular-sized hobby servo. With an operating voltage of 4.8-7.4 VDC, the Mega Servos are able to be run in unison with hobby servos without the hassle of using a separate power source. The robust 6061-T6 anodized aluminum structure and 1" OD output shaft can handle even the toughest applications.

For further information, please contact:

ServoCity	Website: **www.servocity.com**

BOE Shield (for Arduino) and Robot Shield Kit



With Parallax's new Board of Education (BOE) Shield, you can plug into an Arduino Uno, Mega, Duemilanove, or any module with the same form-factor interface. It provides convenient breadboard prototyping, servo ports, and 3.3V and 5V voltage regulators with no soldering required. Also available is the Robot Shield Kit which includes everything needed to turn an Arduino module into a BOE Shield-Bot: a Board of Education Shield, the popular Boe-Bot chassis, servos, and electronic components, plus additional mounting hardware. Retail prices are: $29.99 (Board of Education Shield) and $119.99 (Robot Shield Kit).

For further information, please contact:

Parallax	Website: **www.parallax.com**

Audio Feature for ADAM Mobile Robot

Robotics® RMT announces ADAM RAP (Reactive Audio Playback) — a programmable sound system designed exclusively for the ADAM (Autonomous Delivery and Manipulation) mobile robot (**www.adamrobot.com**). ADAM RAP enhances the safety and flexibility of autonomous mobile robots through interactive voice messages and a mobile "vehicle in motion" jukebox.

The ADAM RAP application plays various sound bites or text-to-speech audio based on the particular function the robot is undertaking at the time. All AGVs have a standard beeper-based vehicle in motion alert system which is mandated by international safety standards. In many operations, however, noise proliferation combined with the monotonous beep of vehicles tends to diminish the alertness of workers with constant exposure. ADAM RAP's innovative design promotes a safer work environment and enhances the interaction between workers and robots.

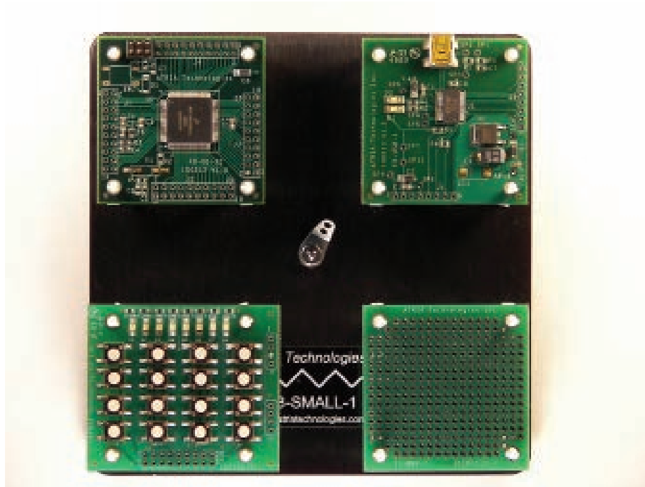
For further information, please contact:

RMT Robotics	Website: **www.rmtrobotics.com**

EXemplar Basic Kits

Atria Technologies is now offering EXemplar kits for under $75 that help users begin experimenting with microcontrollers and programming. The EX-BASIC-JM-32 and the EX-BASIC-QE-32 provide the essentials to get started. Both kits include:

- A 32-bit microcontroller (MCF51JM128 or MCF51QE128)
- BASIC ON BOARD
- 4x4 keypad
- Eight LEDs
- A prototype area
- USB communications interface
- USB cable
- An onboard power supply
- A stable platform to build your projects on. With a PC and a terminal emulation program, users are ready to begin. There is nothing extra to purchase.



Programming dongles or compilers are not required. The microcontroller is preprogrammed with BASIC ON BOARD.

Bascially, you solder a few wires to configure the kit, connect a PC to it, open a terminal emulator, hit ENTER, and begin programming. Power for the EXemplar kit is provided through the USB connection on a PC.

For further information, please contact:

Atria Technologies	Website: **www.AtriaTechnologies.com**

A-Pod Walking Robot

The new A-Pod robot from Lynxmotion is insect inspired. The angled legs provide additional range of movement; in fact, the three DOF (degree of freedom) leg design means the robot can walk in any direction.

The robot has been designed to use 18 HS-645 servos for the legs. The available combo kit includes everything necessary to make a functional robot except the batteries. This robot design was heavily inspired by the original A-Pod by Kåre Halvorsen.

The robot is made from precision-cut PVC and ultra-tough high quality aluminum Servo Erector Set brackets. The combo kit comes with the SSC-32, Bot Board II, and

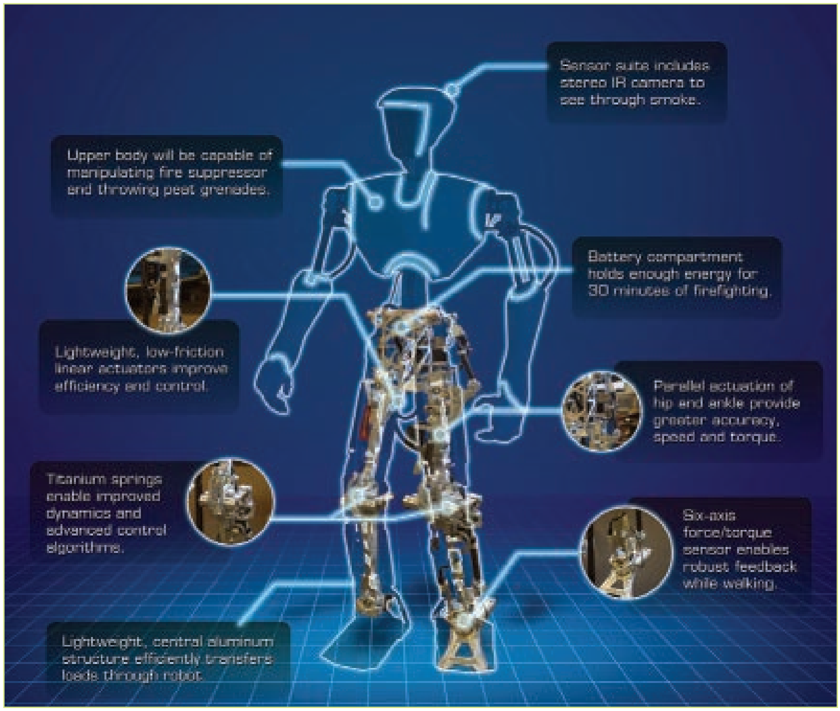
Continued on page 62

bots IN BRIEF

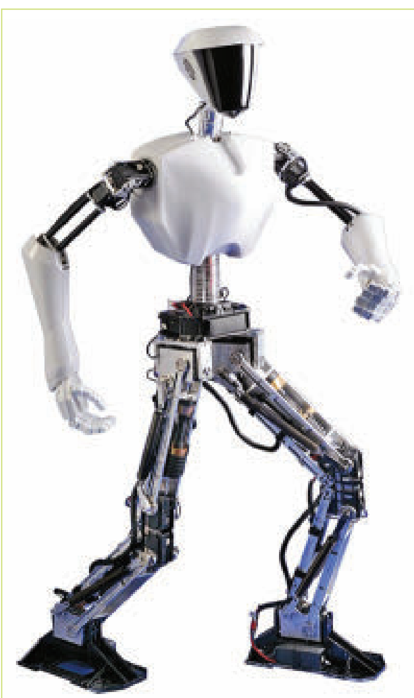
SEAFARING FIREFIGHTER

The Office of Naval Research has announced that they're developing SAFFiR — a humanoid firefighting robot designed to operate aboard ships that is going to be developed in partnership with Virginia Tech's RoMeLa (already famous for their CHARLI humanoid).

SAFFiR (which stands for Shipboard Autonomous Firefighting Robot) will be able to do the following:



- SAFFiR is designed with enhanced multi-modal sensor technology for advanced navigation and a sensor suite that includes a camera, gas sensor, and stereo IR camera to enable it to see through smoke.
- SAFFiR's upper body will be capable of manipulating fire suppressors and throwing propelled extinguishing agent technology (PEAT) grenades.
- The robot will be capable of walking in all directions, balancing in sea conditions, and traversing obstacles like ladders.
- SAFFiR will have multimodal interfaces that will enable the robot to track the focus of attention of the human team leader, as well as to allow the robot to understand and respond to gestures, such as pointing and hand signals. Where appropriate, natural language may also be incorporated, as well as other modes of communication and supervision.



The reason that SAFFIR is a humanoid (and not something far easier to manage like a quadruped) is that it's designed to be able to fight fires aboard ships which means that it's going to need to be able to climb up and down very steep staircases and ladders. However, getting a bipedal humanoid to pull off a feat like this is not going to be easy. It wouldn't be easy to do in a lab setting, much less in a ship that's on fire, dark, hot, smokey, and probably rolling and pitching on ocean waves. SAFFIR will have to be able to handle running into things and falling and getting up again without significantly damaging itself — all in overheated environments with poor sensor data.

A humanoid-type robot was chosen because it was deemed best suited to operate within the confines of an environment that was designed for human mobility, and it offers opportunities for other potential warfighting applications within the Navy and Marine Corps.

At this stage, SAFFIR seems like a very ambitious program, but since the Navy is involved, hopefully the ambition will be backed by a giant pile of cash.

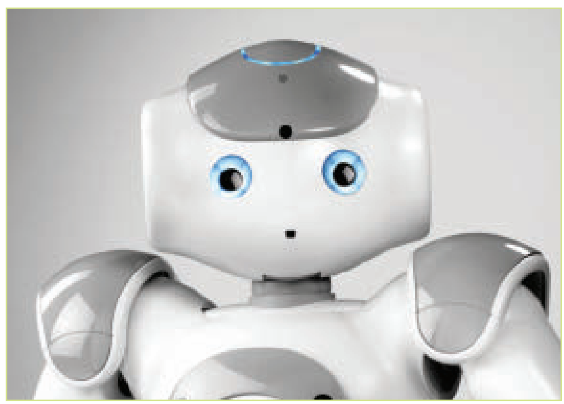
(NOT) CAMERA SHY

How did William Burrard-Lucas — a UK wildlife photographer — get this close-up shot and survive to tell the story? He used a remote-controlled robotic camera, of course!

It's called BeetleCam, and it's basically a DSLR camera mounted on a six-wheel mobile platform that he can control from a safe distance. Burrard-Lucas created the first version of BeetleCam a few years ago to take close-up, wide-angle photographs of dangerous African animals in the wild. (Emphasis on dangerous.)

Now, he's upgraded his original BeetleCam, building new models with more advanced capabilities including HD video recording, wireless "live view," and remotely operated camera tilt. He also constructed an armored, lion-proof carapace to (hopefully) protect the equipment. Last year, he took the new camera bots to Kenya to photograph the lions of the Masai Mara.

SHOW ME THE MONEY



In news that will send a ripple of excitement through the robotics community, French robot maker Aldebaran has reportedly sold a majority stake in its business for about US $100 million.

The *Financial Times* reports that the buyer — Japan's Softbank — will provide the financial backing Aldebaran needs to "bring robots into the mainstream consumer market."

According to the *Financial Times* story, Softbank is understood to have paid about $100m for more than an 80 percent stake in the business, and will invest another $40m to $50m to accelerate development. Softbank bought out existing shareholders in the company, including Intel Capital, CDC Innovation, iSource, and Crédit Agricole Private Equity. Intel Capital had only recently become an investor in a $13m fundraising round last summer.

Although Softbank has not previously been involved in robotics, Masayoshi Son — the company's chief executive — is interested in addressing a number of societal problems, for example, backing the development of green energy in Japan. Securing care for an aging population is in line with this.

This is huge news for Aldebaran — famous for its advanced humanoid robots — but an even bigger deal for the robotics field which has long been trying to prove to investors that there's money to be made in robots. Though industrial robotics is a multibillion dollar sector, consumer and service robotics remain relatively small markets. Hopefully, the Aldebaran deal will show investors that robots can not only be cool, but profitable too.

ARMED AND READY

HDT Global has just introduced some new robotic limbs to give explosive ordnance disposal (EOD) robots like PackBots and Talons a helping hand (or two) when it comes to complex and delicate tasks like defusing bombs. This is a very good idea, since just poking high explosives with a simple gripper doesn't always work out the way everyone would like.

The MK2 limbs can be mounted either singly or as a dual-arm torso on top of an EOD robot, replacing the much simpler open/close gripper systems. Instead of grippers, MK2 comes with actual jointed arms and hands with four degrees of freedom and an opposable thumb. The idea is to make the system similar enough to human arms and hands so that an operator can do just about everything they'd want to do with their own arms and hands while still staying as far away as possible from things like bombs.

The full torso offers a total of 27 degrees of freedom in a package that only weighs 23 kilograms (51 pounds). These arms are actually more muscular than they look. Together, they can lift 50 kilograms (110 pounds) with approximately the same speed as a human, and they're dexterous enough to unzip backpacks, disassemble complex devices, and even use tools.

While there's no doubt that the HDT arm (or arms) are more versatile than the standard arms that come with Talon robots or PackBots, the HDT's hardware is also more expensive. It's ruggedized and watertight, of course, but since it's a more complex system overall, more things can go wrong. These sorts of things are always trade-offs, though, and considering how much more the MK2 arms are capable of, it seems likely that they'll find a place in a robot arsenal doing something useful; namely, risking getting blown up so that we humanoids don't have to.

ROBOSQUIRREL ACTS NUTS

Animals generally tend to treat robots with either indifference or — more commonly — curiosity. After all, robots are clearly not food, and they're not usually threatening. So, more often than not, animals are satisfied to just try and figure out what they are. Turns out, however, if you build a robot that's deliberately designed to provoke an animal, that actually works out pretty well. Meet RoboSquirrel.

RoboSquirrel is modeled (very) closely after a ground squirrel, and features a real squirrel skin, heated innards, and a heated and movable tail. RoboSquirrel even sleeps in a squirrel bedding to make it smell just like the real thing.

Why go to all this trouble? Well, researchers from San Diego State University's Behavioral Ecology Lab are trying to figure out how the squirrels interact with their nemesis — rattlesnakes.

Ground squirrels and rattlesnakes have been going at it for a long time and each has done its level best to try to out-evolve the other, resulting in a continuously changing stalemate. Since the snakes are predators, the squirrels arguably have the harder job, but they've developed a resistance to rattlesnake venom and this cool tail-flagging behavior that might help to distract or confuse the snakes. To determine what the deal is from the perspective of the snake, RoboSquirrel gets sent into striking range of a wild rattler, and then either does or does not flag its tail, according to Bree Putman, a member of the RoboSquirrel project.

While the research is still in preliminary stages, RoboSquirrel does seem to suggest that tail-flagging has some sort of effect on the snakes, possibly discouraging them from striking. The research group is already planning for the deployment of RoboSquirrel 2.0 later this summer, and they've got a RoboKangarooRat in the works, as well.

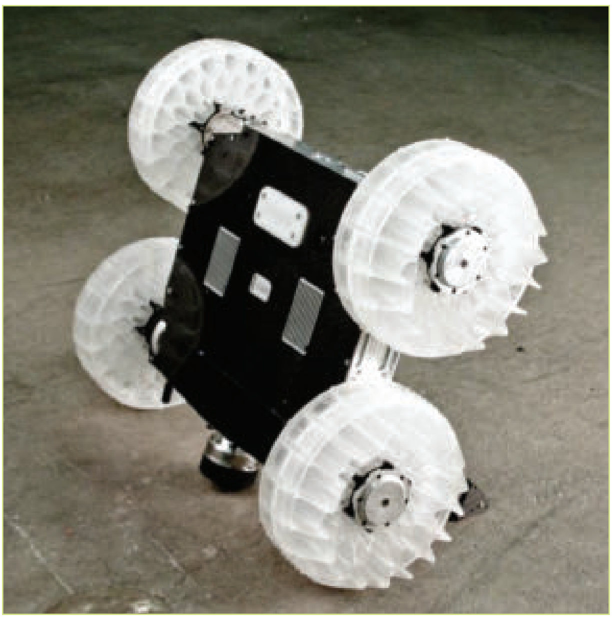
GETTING A SENSE OF THINGS

R2-D2 from Star Wars arguably has more personality than many robots twice its size, but without a face or limbs to speak of where does it all come from? The answer, of course, is sound. UK researchers are now trying to do the same with real robots, teaching them to communicate information and emotions to humans using beeps, boops, and squeaks.

Robin Read and Tony Belpaeme from Plymouth University's Centre for Robotics and Neural in the UK, are investigating the relationship between things like the pitch and rhythm of sounds and their perceived emotional connotations. Funded by the ALIZ-E Project — a European effort to create robots that can form meaningful bonds with humans in a hospital setting — the researchers asked several dozen six to eight year old kids to try to match sounds with expressions.

It turned out to be quite striking how the children showed strong categorical perception when interpreting the robot's utterances. There was no subtlety in their interpretation: The robot was — in their words — either sad, happy, angry, scared, surprised, or tired, but they seldomly interpreted utterances in more subtle emotions. It is believed that upon closer inspection, categorical perception will be observed in other modalities also, having a significant impact on the design of HRI [human-robot interaction] for younger children. Basically, any effort to convey subtlety might be a lost effort.

Non-verbal communication (whether or not subtlety is involved) is going to be a critical skill for human-robot interaction in the short term, since it doesn't require any complex hardware or software to implement and it's generally language and age independent. Roomba owners, for example, will immediately recognize their robot's "I'm charged!" tune. Communication isn't limited to audio, either.



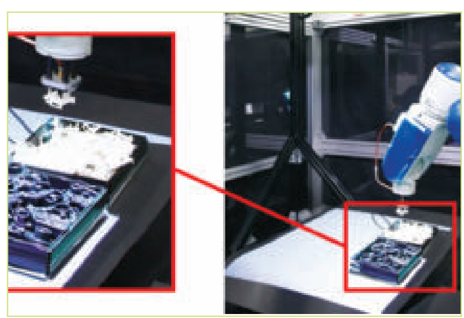
BETTER THAN A JUMPING BEAN

The Sand Flea had its origins in the Precision Urban Hopper which was born from a collaboration between Sandia National Labs and Boston Dynamics back in 2009. However, there are some significant differences in the latest version of Sand Flea. For example, instead of jumping while moving (like the Precision Urban Hopper did) Sand Flea stops, rears back, and launches itself into the air:

It has no trouble clearing a 10 meter obstacle (about 30 feet), and it's accurate enough that you can ask it to jump through a window two stories up and it'll do it. The piston (which looks like it fires out the back of the robot, as opposed to downwards) is powered by CO_2, and Sand Flea can make 25 jumps in a row before it needs to juice itself up again. Sand Flea is intended to be used in places like Afghanistan to hop over walls, take a look around, and hop right back home again.

The tricky part is keeping Sand Flea oriented as steadily as possible during the jump. The idea is that the robot will be able to send back useful video while in midair, which a haphazard aerial tumble would preclude. It looks like it does a halfway decent job, but this robot seriously needs a tail.

FOCUS ON COST REDUCTION



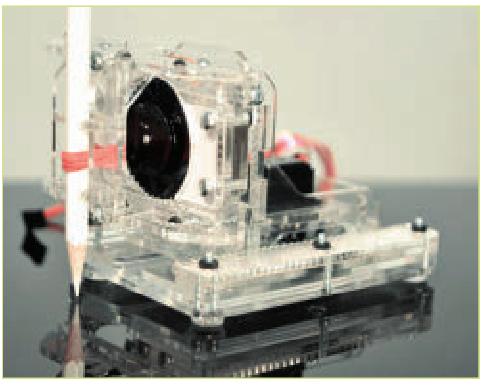
It seems that Canon plans to cut production costs by about ¥400 billion (~$4.8 billion) by using robotics. The company will be utilizing robots in both their toner department and the assembly line for SLR camera lenses. Canon hopes the move will increase their profit margin over the next few years. Considering that the company is already in the robot building business, it certainly sounds logical.

PET PROJECT

New (sort of) from Neato Robotics is the XV-21: an autonomous robot vacuum that's been enhanced to provide better cleaning for people with an excess of pets. Neato seems to be sort of taking the same approach as iRobot did when they introduced their Roomba 572 "Pet Series." The 572 was essentially just a 500 series Roomba with a new name and some slightly different accessories that cost a bit more. The XV-21 is pretty much the same as the XV-11, except for the following:

- A new bristled brush to improve the pick-up of pet hair, dander, and fibers, and to allow the vacuum cleaner to run more quietly than previous models,especially on hardwood floors.
- A new filter which is cleanable and designed for long-term use that provides increased airflow and suction power which allows the Neato XV-21 to pick up and retain more than three times the fine dust particles of the company's standard filter.
- A new paint job that's "misty white with purple highlights."

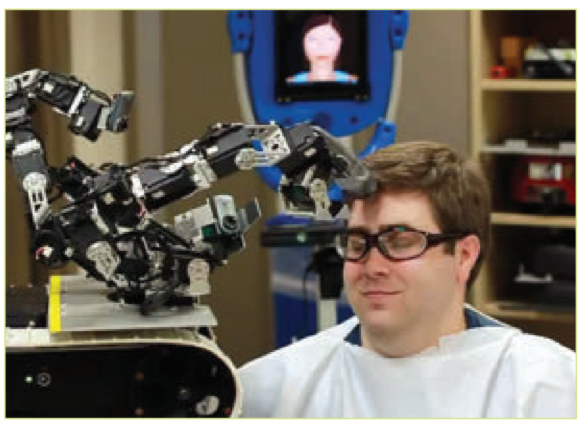
If you like the sound of the XV-21 but already have an XV-11, it's easy to upgrade. You can buy the brush and filter separately (for $60) and install them into your robot, which suggests that there aren't any fundamental hardware differences to the XV-21 (which you can buy on its own for $430).



PICCOLO PETITE

Piccolo is a pocket-sized stand-alone CNC platform. For under $70, you will be able to assemble your personal Arduino-compatible kit for tinkering and playing with basic CNC output. Whether it's plotting a quick graffiti, printing a one-off business card on the fly, or multiple Piccolos working together to create a large mural, this kit provides a platform for experimenting with 2D or 3D digital fabrication at a small scale. The folks at Diatom Studio are currently refining the Piccolo prototype into an open source design that is simple, quick to assemble, and easy to use, and is entirely composed of digitally manufactured components and inexpensive off-the-shelf hardware. The Piccolo project includes Arduino and Processing libraries so Piccolo can be used in a variety of ways such as moving autonomously or responding to sensors and data, whilst providing an accessible educational tool and a new output for Processing sketches. If you would like to be notified when Piccolo is released, you can leave your email address at **http://piccolo.cc/**.

A CUT ABOVE



Here's a robot that Tim Judkins and his colleagues at Intelligent Automation in Rockville, MD designed to shave his head. Why the shaved head? Tim decided to get bald in support of the St. Baldrick's Foundation — a volunteer-driven charity committed to funding the most promising research to find cures for childhood cancers. Volunteers get bald at the Foundation's worldwide signature head-shaving events to stand in solidarity with kids with cancer who typically lose their hair during treatment and to raise money to support life-saving childhood cancer research. Tim and his team of fellow engineers equipped the multi-arm UGV (Unmanned Ground Vehicle) with a pair of clippers and two cameras, put on a pair of safety glasses, and prepared Tim for his shearly fantastic shave. As the last piece of hair hit the floor, Tim had raised more than $1,500 for life-saving research. The barber bot sports three arms that provide lots of degrees of freedom which are typically used for tasks like backpack inspection, tool handling, shovel manipulation, door breaching, knot tying, and tackling IEDs.

FROM SOCCER BALLS TO PALLETS

Photo by Joel Eden Photography/Kiva Systems.

Looks like Amazon is getting some robots. LOTS of robots.

The giant online retailer is acquiring Kiva Systems, a North Reading, MA-based company that invented a revolutionary way of managing vast warehouses by using fleets of mobile robots to sort, organize, and transport inventory. Amazon agreed to acquire all of the outstanding shares of Kiva for approximately US $775 million in cash. The companies expect to close the acquisition in the second quarter of 2012.

So, what does Kiva do that got Amazon so interested? Basically, Kiva reinvented the centuries-old warehouse business, transforming distribution centers — which previously relied on slow-moving humans to walk around picking and packing goods — into a buzzing hive of super-efficient, tireless robotic workers.

"This is a great validation of the innovation model that I have been encouraging for years as a university professor: Engage in research that pushes the boundary of autonomous systems capabilities, without worrying about whether it has a direct or immediate application," commented Raffaello D'Andrea, an ETH Zurich professor and one of Kiva's co-founders.

"The robotic aspects of Kiva Systems had their genesis in robot soccer. Many of Kiva's key, initial technical hires were former Cornell RoboCup team members with expertise in dynamics and control, mechanical engineering, electrical engineering, and computer science," he said. "In addition, one of Kiva's earliest hires was a collaborator on the interactive art installation 'The Table,' currently part of the National Gallery of Canada's permanent collection. Who would have thought that autonomous, soccer-playing robots — or that a robotic artwork — would enable a business like Kiva Systems?"

Yeah. Who would have thought that robot soccer would one day lead to a $700 million idea?

STILL CLEANING UP

iRobot is introducing their new Scooba 390 (it's the one in the back of the photo). It's a lot like its predecessor, the Scooba 380. It's actually nothing completely new or revolutionary, but there are a few unique features that are worth mentioning.

iRobot is highlighting several differences between the 390 and previous Scooba models. First, the battery life is 30 percent better thanks to "extended power life management." The 390 can clean up to 450 square feet (42 square meters) per run.

Next, iRobot points out that the 390 has a simplified interface that's easier to use, and while the buttons have been labeled a bit better, what they're really talking about is the fact that the 390 is easier to clean and maintain. Every part that you might need to pay attention to is now clearly marked with orange icons. iRobot's Scooba 390 is on sale now for US $499 — the same price as the Scooba 380 that it replaces.

Continued on page 62

Cool tidbits herein provided by Evan Ackerman at www.botjunkie.com, www.robotsnob.com, www.plasticpals.com, and other places.

COMBAT ZONE

Discuss this section in the *SERVO Magazine* forums at http://forum.servomagazine.com

Featured This Month:

www.servomagazine.com/index.php?/magazine/article/may2012_CombatZone

PARTS IS PARTS

Product Review: Turnigy Simplex 1 ~ 4 Cell LiPo/LiFe AC/DC Charger

● by Pete Smith

Combat robotics is essentially an indoor sport where access to an AC outlet is not usually a problem. However, most chargers on the market are designed for outdoor hobbies like R/C cars, aircraft, etc., where charging from an automobile battery is required.

To use these at a robot event

FIGURE 1. Turnigy Simplex charger.

requires the addition of a power supply to convert the AC line voltage down to the 12V required by the charger.

This is just one more cost and complication for someone entering the sport, so a unit that combines the two at a lower cost makes things simpler.

The unit I'm referring to is the Turnigy Simplex from **www.hobbycity.com.** I got it for about $42 including shipping.

The charger is housed in a small ABS plastic housing with an attached AC lead (**Figure 1**). Also included are a DC power lead (so you can use it with your vehicle), a set of battery-charging leads fitted with a mini JST charging plug, and an instruction leaflet (also available online at **www.hobbyking.com/hobbyking/store/uploads/980548998X286631X4.pdf**).

The first thing to do is replace the round two-pin AC power plug (I think it's suitable in France) with a standard US plug. This is easy enough; I got one at my local Home Depot for around $5.

The plug fitted to the charger leads is common on Antweight batteries but I use the larger Deans and XT60s on my Beetleweight, so I also replaced the connector on the leads with a male XT60.

FIGURE 2. Operator panel.

The operator panel is clearly marked and easy to use (**Figure 2**). You can charge both LiPos and LiFe cells, up to 4S packs. The balance plug is the JST-XH type, so make sure this is the same as your batteries. With a max charge rate of 2A, it can charge most Ant or Beetle packs in 0.5-1 hour.

The only problem I found with the charger was that the case made plugging the balance lead in quite difficult. I fixed this by trimming a small amount of plastic off the top of the slot with a craft knife.

The Turnigy Simplex provides a nice, simple, and easy to use charger suitable for anyone starting up in Ant or Beetleweight combat robotics. **SV**

Product Review: Spektrum DX5e 2.4 GHz Transmitter

● by Pete Smith

The 2.4 GHz spread spectrum transmitters and receivers have transformed the R/C market. Worries about other people using the same frequency all but disappear when you use one. I described one of the cheap PC programmable radios in an earlier issue of *SERVO*, but the complexities involved in setting it up may have put some potential users off.

Spektrum (**www.spektrumrc.com**) kick-started the 2.4 GHz revolution a few years back, but there have been a few hiccups along the way. The first radios (like the DX6) used a protocol called DSM but only one receiver — the BR6000 — was available that would correctly fail-safe for combat robots. When they moved to the DSM2 and now the DSMX, the BR6000 became unavailable. Unfortunately, there are still fail-safe issues with several of the new receivers.

The answer to the receiver question are the new (and cheap) receivers from Hobby King (**www.hobbyking.com**). Their R610 and R410 receivers both bind

FIGURE 1. The DX5e.

FIGURE 2. Servo reverse and mix switches.

and fail-safe without issues with the Spektrum DSM2 and DSMX radios (they will not work with the older DSM sets).

The DX5e featured here is a particularly good starting point for a beginner in the sport.

It's relatively cheap at $70 including shipping from sources like the Robot MarketPlace (**www.robotmarketplace.com**). The Hobby King R610 and R410 receivers are both cheap and small, and are easy to set up for most combat robotics uses.

The four main channels can be reversed using switches on the front panel (**Figure 2**) which makes getting your drive wheels going in the direction you want much easier. If you are not using a mixing ESC in the bot for your drive, then the built-in delta wing "mix" allows you to do this in the transmitter instead.

Binding is done with the usual binding plug on the receiver and use of the "buddy" switch on the transmitter. I have found this to be trouble free on the two DX5es that I have owned.

The only downsides to this transmitter are a cheapish construction and its use of dry cell, non-rechargeable batteries. I use one with my competition-winning Beetleweight "Trilobite" and it has proven trouble free in about one years' worth of use. It may not have all the functions of more expensive sets, but it has all you really need! **SV**

TGIF at Motorama

● by Dave Graham

President's Day weekend in Harrisburg, PA means only one thing: Motorama! Billed as the nation's largest all-indoor motorsports event, Motorama is held at the sprawling Pennsylvania State Farm Show Complex. The two-day event boasts one million square feet of exhibition space; draws over 1,500 riders, drivers, and competitors; and entertains in excess of 50,000 spectators. If it has wheels and a motor, there's an event for it at Motorama!

The NorthEast Robotics Club (NERC) partners with Motorama to bring fighting robots to the venue. Officially known as Robot Conflict, the NERC event has taken the moniker of the overall event and has come to be known and advertised as simply Motorama. This year, an international group of 100 fighting robots in five open weight classes ranging from 150 gram Fleaweight (a.k.a., Fairy) to 30 pound Featherweights met in the box to fight it out. There were also two BotsIQ 15 pound weight classes: one for high school and one for college students.

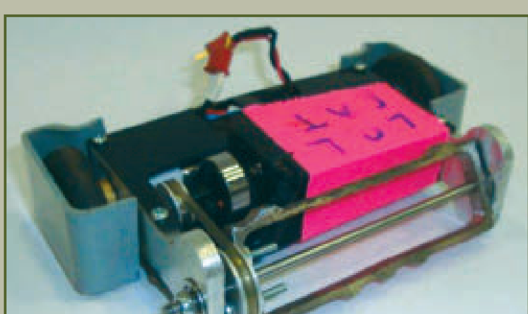


FIGURE 1. Fleaweight bot Lolcat.

NERC uses Friday at Motorama to conduct the 150 gram Fleaweight and one pound Antweight robot competitions. Two minute matches take place in an eight foot arena, and follow a double elimination format. What I like about Friday at Motorama is it's really the setup day for the overall event, so there are no crowds. There are no spectator admission charges (the $20 admission to Motorama is included in the NERC registration fee for competitors), you can park right next to the Farm Show Complex, and parking is free (normally $8 daily). That makes Friday an enjoyable day for competitors, a free day for spectators, and a perfect opportunity for school field

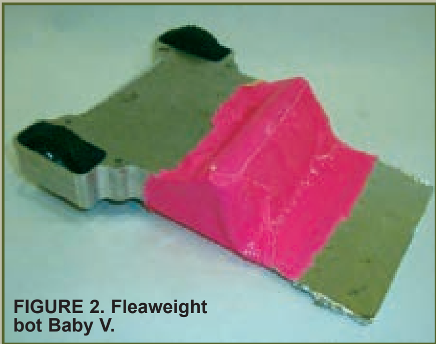


FIGURE 2. Fleaweight bot Baby V.

FIGURE 3. Fleaweight bot Rosie the Littler.

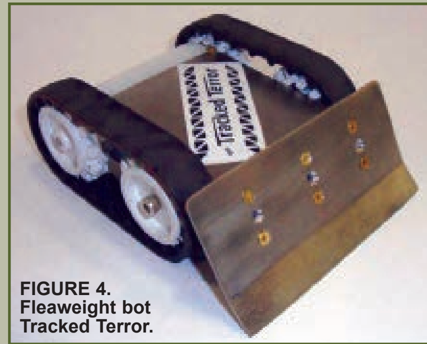


FIGURE 4. Fleaweight bot Tracked Terror.

trips. (Hence the name and focus of this article.)

Nine Fleaweight robots brought their wedges, spinners, an expertly crafted beater, and a servo-articulated chop saw to the fight. It was an incredible collection of bots and a great competition. My two spinners, Hedgehog and Tomahawk, both lost two consecutive matches and were eliminated early. In winner's bracket action, John Parsons and his beater bot Lolcat (**Figure 1**) sent my Mateo teammate Matt Benjamin and his bot Baby V (**Figure 2**) to the loser's bracket, along with Rosie the Littler (**Figure 3**) and Transcendental Terror. Also in top bracket action, Chris Atwood of team Sandman and his bot Tracked Terror (**Figure 4**) sprinkled some sleepy dust on Kyle Singer's chop saw bot Kongol (**Figure 5**) and Jeremy Campbell's spinner bot Rebound, sending them both to the bottom bracket while propelling himself to the winner's bracket semi-final showdown with Lolcat.

Tracked Terror won the semi-final match and moved on to the finals. Baby V fought its way back in the loser's bracket and earned a semi-final rematch against Lolcat. This time, Baby V won the judge's decision in a close match, and moved on to the finals to face unbeaten Tracked Terror. In the championship bout, Tracked Terror was just too much for Baby V and won an easy victory with an unstoppable combination of speed, power, and driving skill.

Tracked Terror uses a set of lightweight Pololu hubs and tracks. You can check out the track system and other great robot products on the Pololu website at **www.pololu.com**.

Between the Fleaweight and Antweight bouts, competitor Kyle Singer of team Twisted Sick Robots entertained us by demonstrating his proficiency with drift skates. When I first saw the skates (**Figure 6**), I thought they were casters for his pit equipment. That was right up until Kyle threw them on the ground, somehow balanced himself on them, and then took off by moving his lower body to generate momentum (**Figure 7**). It was impressive. You can find drift skates on the Web — use them at your own risk! Kyle said it took him about a week to get used to them.

This year, Motorama welcomed several couples in either a supportive or competitive role. Event Safety Coordinator Ed McCarron and his wife Beth (**Figure 8**) are staples at Motorama and oversee virtually every aspect of the event. Long-time competitor John Durand and his girlfriend Gina Marchiano

FIGURE 7. Kyle Singer on drift skates.

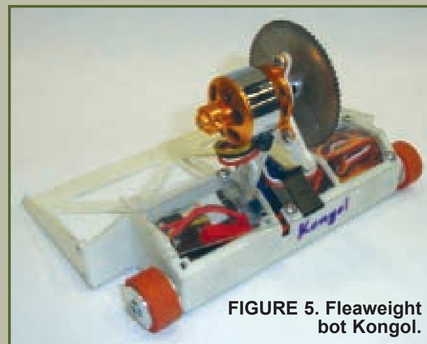


FIGURE 5. Fleaweight bot Kongol.

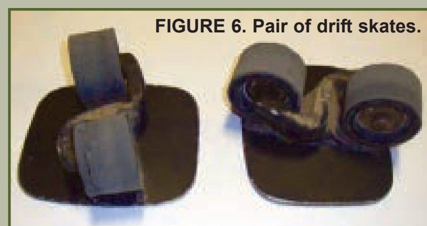


FIGURE 6. Pair of drift skates.

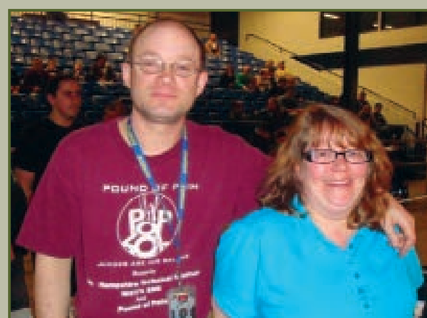


FIGURE 8. Event Safety Coordinator Ed McCarron and his wife Beth.

FIGURE 9. Emcee John Durand and his girlfriend Gina Marchiano.

FIGURE 10. Registration Coordinator Jim Iocca.

FIGURE 11. Antweight bot See You Next Wednesday.

FIGURE 12. Team Near Chaos Robotics couple Mike Jeffries and Julie Simancek.

FIGURE 13. Antweight bot Kobalos.

FIGURE 14. Antweight bot Motor City Massacre.

FIGURE 15. Team Radicus couple Tony Fowlie and Lauren Greenwood.

(**Figure 9**) provide the audio equipment and John emcees. Jim Iocca (**Figure 10**) is the event Registration Coordinator, and he and his wife Julie (unavailable for a photo) tag-team drive their Antweight spinner bot See You Next Wednesday (**Figure 11**).

Team Near Chaos Robotics couple Mike Jefferies and Julie Simancek (**Figure 12**) compete with Antweight bots Kobalos (**Figure 13**) and Motor City Massacre (**Figure 14**), and team Radicus couple Tony Fowlie and Lauren Greenwood (**Figure 15**) bring a fleet of bots and compete in the Antweight class with bots Poor Punctuation 2.0 (**Figure 16**) and Malicious Mule (**Figure 17**).

Twenty Antweight bots answered the Motorama challenge and engaged in an entertaining and destructive competition. In second round action, my spinning drum bot Snaggletooth (**Figure 18**) drew undercutter Odahviing (**Figure 19**), and I thought I'd have a short day. Fortunately for me, driver Branden Youssef of team Y2K Robotics couldn't get Odahviing's weapon to spin up, so I had a pretty easy time pushing Odahviing around the ring. Snaggletooth managed to stand Odahviing on its nose (**Figure 20**) with about 10 seconds left in the match. Odahviing ended up having the short day, losing its consolation match to Malicious Mule.

Snaggletooth went on to send Joey Maffei and team Green Machines to the loser's bracket after flipping Swamp Woman (**Figure 21**) and earning the judge's decision. Snaggletooth's day ended in the next match when Warren Purvin of team Pretzel Robotics and his bot Vile Ant (**Figure 22**) put a vicious hit on him that racked his chassis (**Figure 23**) and sent him to the pits for the day. Note the hardened steel teeth on Vile Ant that were added by team Pretzel Robotics specifically for Motorama.

In other upper bracket action, Brandon Young of team Bone Dead Robotics made it to the quarterfinal match by sending Malicious Mule and Amatol to the lower bracket. His opponent in that quarterfinal

FIGURE 16. Antweight bot Poor Punctuation.

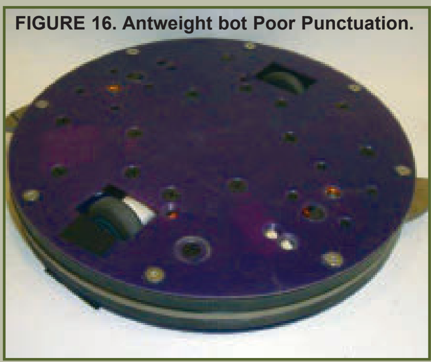
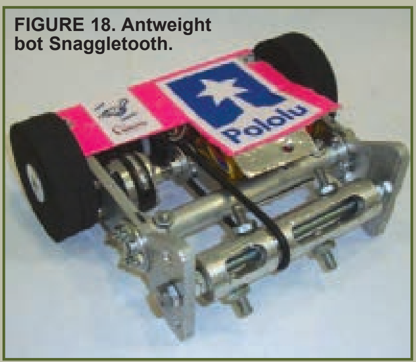


FIGURE 17. Antweight bot Malicious Mule.

FIGURE 18. Antweight bot Snaggletooth.



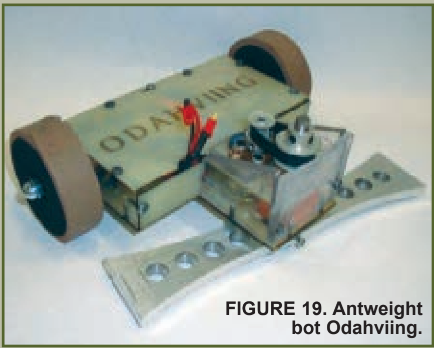


FIGURE 19. Antweight bot Odahviing.

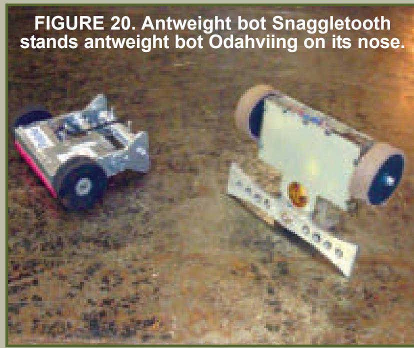


FIGURE 20. Antweight bot Snaggletooth stands antweight bot Odahviing on its nose.

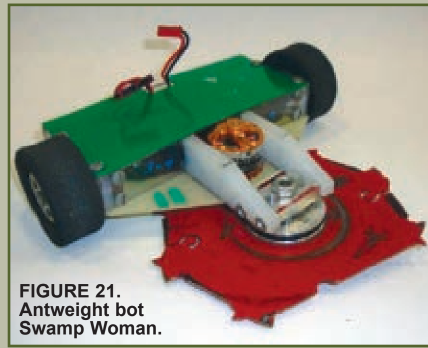


FIGURE 21. Antweight bot Swamp Woman.

match was Jim Iocca driving See You Next Wednesday, who made the journey by knocking off Motor City Massacre, Gyroscopic, and Kyle's Cutter. See You Next Wednesday took the quarterfinal match and moved on to meet young Purvin and his dangerous undercutter Vile Ant in the semi-final match. Vile Ant won that match and said, "See You Next Wednesday" to Jim and Julie, who moved to the loser's bracket semi-final match.

In the Ant loser's bracket, Chris Atwood and his wedge bot Antelope (**Figure 24**) eliminated Motor City Massacre, Amatol, Low Blow, and Snaggletooth (via forfeit) to get to the quarterfinal match. In that match, he met Mike Jeffries and his wedge bot Kobalos. Kobalos earned a berth in the quarterfinal match by eliminating Malicious Mule, Gyroscopic, and Fericious.

Kobalos prevailed in the quarterfinal match and moved on to meet See You Next Wednesday in the semi-final match. In the semi-final match, See You Next Wednesday seemed to have trouble spinning up and as a result Kobalos batted it around the arena like a hockey puck, earning the easy win on hits and driver aggression.

The victory set up a classic Antweight final match between the unstoppable wedge Kobalos and the big spinning undercutter Vile Ant. Kobalos had the upper hand early in the final match, pushing Vile Ant around. Eventually, Vile Ant ran over the top of Kobalos, taking a wheel off the wedge in the process. Kobalos limped around the arena until Vile Ant snatched his other wheel to take the victory. It was Warren Purvin's first Ant championship. Warren posed with his father (and pitman) Glenn after the match (**Figure 25**).

A complete list of all the winners is in **Table 1**. The top three winners in each weight class received beautifully etched Plexiglas trophies and a bounty of merchandise, gift certificates, and discount coupons from event sponsors Castle (**www.castlecreations.com**), Holmes Hobbies (**www.holmeshobbies.com**), Kitbots (**www.kitbots.com**),


FIGURE 22. Antweight bot Vile Ant.

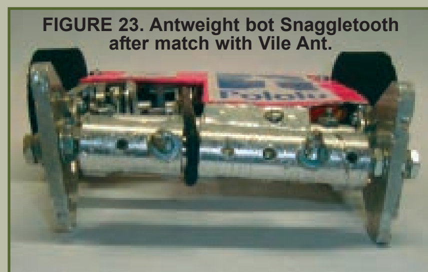


FIGURE 23. Antweight bot Snaggletooth after match with Vile Ant.

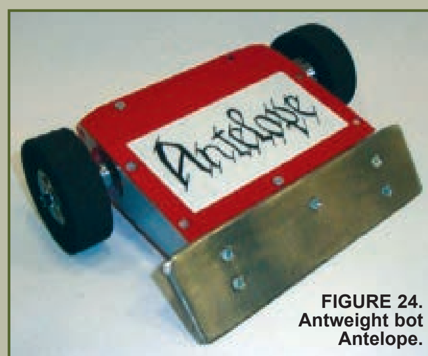


FIGURE 24. Antweight bot Antelope.

TABLE 1 — Winners

	FLEA	ANT
1st:	Tracked Terror *Chris Atwood*	Vile Ant *Warren Purvin*
2nd:	Baby V *Matt Benjamin*	Kobalos *Mike Jeffries*
3rd:	Lolcat *John Parsons*	See You Next Wednesday *Jim & Julie Iocca*

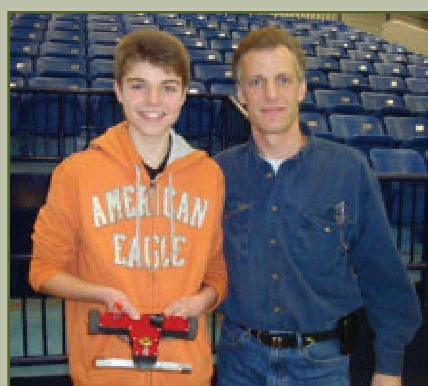


FIGURE 25. Warren Purvin (left) and father Glenn (right) with Antweight champion Vile Ant.

FIGURE 26. Eight year old Josh and five year old sister Kiera McCarron in the eight foot arena.

FIGURE 27. Steel bar peeled off 12 pound wedge bot Tough Nut by spinner bot Fiasco.

FingerTech Robotics (**www.fingertechrobotics.com**), and The Robot MarketPlace (**www.robotmarketplace.com**).

My final shout-out goes to two Motorama lifetime attendees — five year old Kiera and eight year old Josh McCarron shown posing in the eight foot arena door (**Figure 26**). Both are the kidbots of event organizers Ed and Beth McCarren. Beth credits each of her children with one more year of attendance since she was at Motorama while pregnant — twice.

I'd be remiss if I didn't give you a peek at the competition on Saturday and Sunday in the big (16 foot) arena. In 12 pound action, spinning bar bot Fiasco peeled a steel bar off of wedge bot Tough Nut (**Figure 27**). Thirty pounders Upheaval and Lock Jaw went at it in the big arena (**Figure 28**).

FIGURE 28. Thirty pound bots Upheaval and Lock Jaw go at it in the big arena.

Mark your calendars now and plan to spend 2013 President's Day weekend in Harrisburg, PA at Motorama. You can follow Motorama on their website at **www.motoramaevents.com**. Keep up with NERC and see all of the results from Motorama on their website at **www.nerc.us**. SV

EVENTS

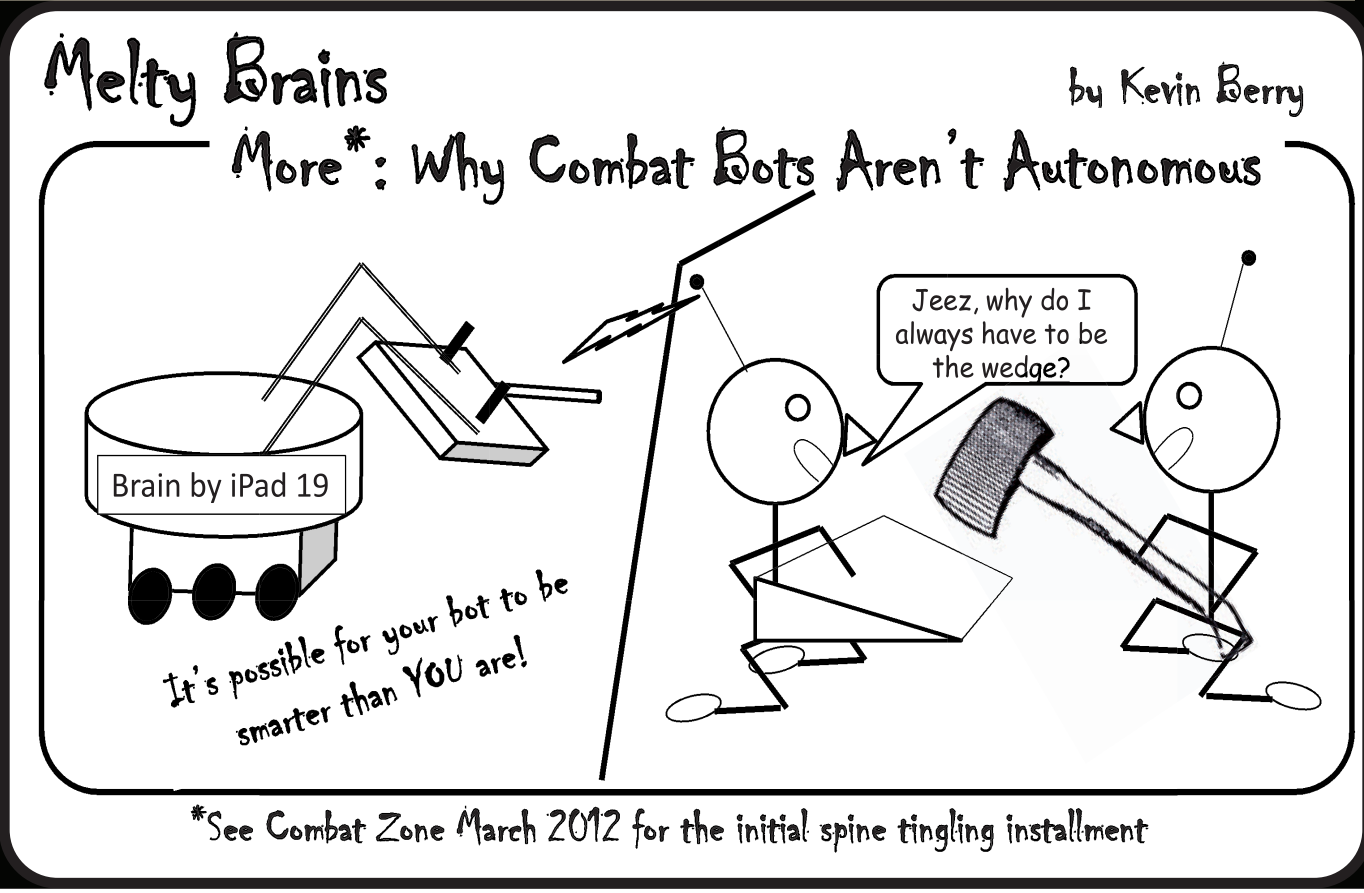
Upcoming Events for the Ohio Robotics Club

2012 is promising to be a banner year for the Ohio Robotics Club. On Saturday, May 19th, we will be returning to The Gate — an Indoor R/C racing club in Brunswick, OH — to present House of Robotic Destruction. Last fall's event went very well and NORCAR has invited us back for what should be another great time. The Gate is located in a large strip mall with plenty of foot traffic, and multiple local dining opportunities. It is a clean and well light facility run by some really great guys. Registration is already open, so get over to the Builder's Database (**www.buildersdb.com**) and get your robots signed up.

This year, we are privileged to help host Mecha-Mayhem during the iHobby Show on October 10–14, 2012. The iHobby Show is the premier hobby trade event hosted by the Hobby Manufacturers Association (HMA). It features booths from dozens of top hobby manufacturers and draws tens of thousands of spectators. In the past, this show was held in Chicago and featured an outstanding Insectweight event hosted by the Chicago Robotic Combat Association (CRCA). This year, the show is moving to the IX Center in Cleveland, OH. This gives ORC the chance to be a vital part of this great event. An undertaking of this size will require a lot more volunteer labor then normal, so we will need "all hands on deck." Due to show rules, the arena will have to be unloaded and set up on Wednesday, October 10th. Thursday and Friday, the show will be open to HMA members and retail representatives only. Builders are welcome to come early, safety their robots, and hunt for sponsors. Saturday and Sunday, the show is open to the general public. We will be holding fights on both days. There will also be R/C car races and aircraft aerobatics. The show host will be giving specific time slots to run in, which will have to be abided by. Typically, fights are held in three or four sessions of an hour or so each day, with breaks of an hour or more in between.

Brian Schwarz from CRCA will be running the event and handling registration via the Builder's Database. ORC will be providing local manpower support. The event should be on the database some time this week. We also plan on continuing the "bot dinner" transition on Saturday. We hope you will plan on attending this great get-together. **SV**

HORD House Of Robotic Destruction



Sounding Off Part 3

Building an MP3 Sound-Making Robot

by Gordon McComb

Discuss this article in the *SERVO Magazine* forums at **http://forum.servomagazine.com**.

Ever notice how the robots in movies like to make noise? Sound helps them to connect with their human counterparts — cheery advise, warnings, whatever. Audible feedback makes them more interesting. Who can forget the *tweeps* and *braaps* of R2-D2, the incessant warnings to Will Robinson by the robot in *Lost in Space*, or the nasal-sounding jabberings of those countless Daleks?

For the last several months, we've been talking about ways to add sound to your robot creations. Simple sounds can be made using just a microcontroller and small speaker; songs and sound effects are easy using a low cost MIDI synthesizer board. This month, you'll learn how to add recorded effects to your robot using MP3 digital audio files.

In this installment, I'll show you how to build the Musicbot shown in **Figure 1**. It uses an Arduino and SparkFun MP3 player shield to play MP3 digital audio files. The shield contains an MP3 decoder, plus a micro-SD card slot for storing all your tunes and effects.

There are numerous digital audio formats; I chose MP3 because it's well-known and easy to work with. MP3 audio files are compressed, so not only do they take up less space on the storage medium, they reduce the amount of data the Arduino must shuttle every second. That means the Arduino has more time for robotic tasks, like motivating motors or sensing sensors.

A Quick Overview of MP3

First, some basics: MP3 is an audio compression standard originally created for the MPEG-1 format of low resolution video. The technical name for the standard is MPEG-1 Audio Layer III. MP3 was later expanded for use in MPEG-2 video which supports higher resolution images.

Though meant to be paired with video, MP3 audio is perfectly happy living on its own; it's this variation most people are familiar with. The MP3 format gained popularity in the mid 1990s as a way to save and play sound and music; compared to compact disc audio tracks, it drastically reduces the size of the stored audio file. Whereas an hour long CD might consume 600+ megabytes of disk space, the equivalent file encoded using MP3 could be just a tenth of that.

Processing a sound file as MP3 is a two-step procedure: encoding and decoding.

- *Encoding* transforms the original digital sound to a highly compressed form. A number of methods are used to achieve this compression, including something referred to as perceptual coding. The technique leverages a science known as psychoacoustics which deals with the way the human ear receives and processes sound. For example, certain frequencies of sound cannot (normally) be discerned when accompanied by

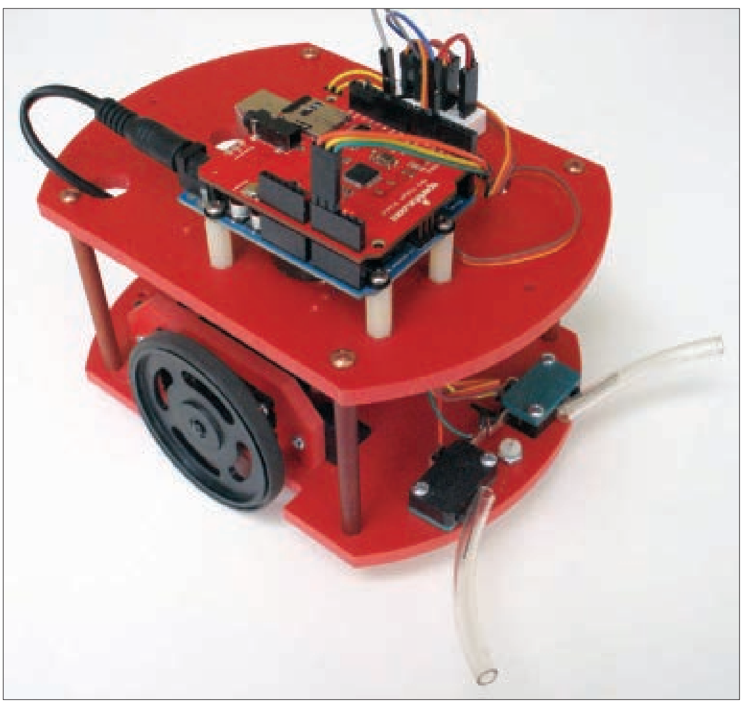


FIGURE 1. The Musicbot uses an ArdBotII robot chassis, Arduino Uno (or compatible) development board, and SparkFun MP3 player shield.

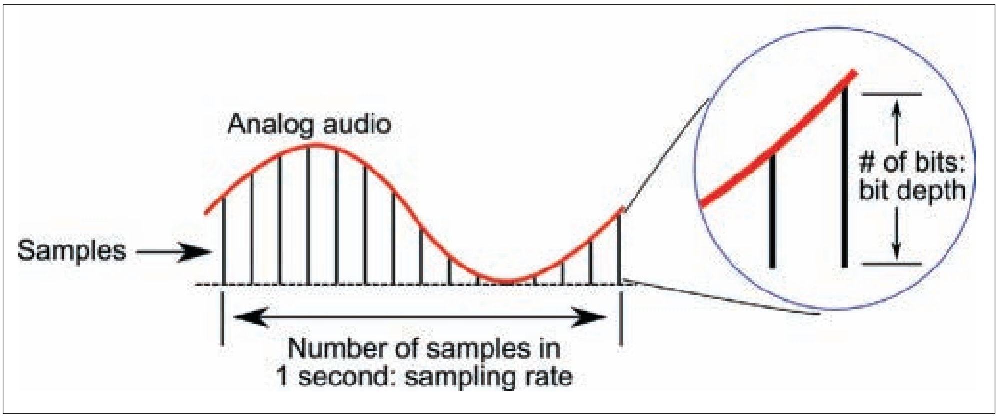


FIGURE 2. Digital sound is created by sampling the audio at regular intervals. The more often the audio is sampled, the higher the sampling rate. Bit depth is the number of bits used to store each sample.

other frequencies. With psychoacoustic modeling, the frequencies that the ear can't hear anyway are eliminated. That saves data space.

- Decoding reverses the process. Data is decompressed, and then may be processed with various audio enhancements such as bass boost and reverb. The MP3 file is composed of multiple frames, each containing a small section of the original audio. The decoding process is built into the MP3 player which can be either software- or hardware-based. For the Musicbot, the player uses separate MP3 decoding hardware since the Arduino is not capable of processing the data all on its own.

MP3 is what's known as lossy compression; that is, during the encoding process, some data is irretrievably lost. Once removed, it cannot be put back. This is why when you re-compress an MP3 file, it doesn't sound as good as the original.

Sampling Rates, Bit Depth, and Bit Rates

Regardless of the compression technique used, analog audio is converted to digital data by taking quick snapshots of the sound at regular — but frequent — intervals. This is called sampling, and is shown in **Figure 2**. The number of times the sound is sampled in one second is the sampling rate. Music on a CD is digitized — converted from analog to digital — at 44,100 times a second. The CD audio is said to have a sampling rate of 44,100 Hz (44.1 kHz).

The instantaneous frequency and volume at each sample of the analog signal is converted to a binary number. The number of bits — called bit depth — of this binary number defines the possible range of digital values that can be stored. The broader the range, the higher the resolution of the sound reproduction. With higher resolution comes better sound.

Assuming a bit depth of 16 bits, the range of values can vary from 0 to 65,536. With an eight-bit depth, the range is only 0 to 255. That's a significant decrease in resolution, and so audio quality may be greatly diminished.

The bit rate is the calculated product of the sample rate times the bit depth times the number of channels used for the recording — stereo audio clips use two channels. Because the sample rate is expressed as the number of "snapshots" per second, bit rate is likewise expressed as the number of bits processed per second. The higher the bit rate, the larger the encoded file will be. Larger files consume more storage space, and they place increased demands on the decoder which has to process more data each second.

Given a 44.1 kHz sampling rate, bit depth of 16 bits, and two channels, the uncompressed bit rate is 1,411,200 bits per second; more commonly specified as 1.411 megabits per second (mbps). For MP3, the bit rate is noted for compressed data. MP3 supports a wide variety of compressed bit rates — all in kilobits per second (kbps) — spanning a standardized range from about 32 kbps to 320 kbps (but higher and lower values are also permitted). The most common are 96, 128, 160, and 192 kbps.

Bit rates can be variable or constant. Variable bit rates can make the resulting file smaller. The process works by intelligently altering the bit rate to whatever minimum is needed to record the audio. Conversely, constant bit rates use one bit rate, regardless of the actual compression that is required. While variable bit rates allow for small files without unduly sacrificing quality, from a programmatic

Sources

Budget Robotics
ArdBotII chassis: Precut body parts, assembly hardware
Vishay TSOP38238 infrared receiver
Universal remote control
www.budgetrobotics.com

Parallax
Infrared receiver (#350-00014)
www.parallax.com

SparkFun Electronics
MP3 player shield
(RTL-10779; retail version comes with pin headers)
Vishay TSOP38238 infrared receiver (SEN-10266)
www.sparkfun.com

Prerecorded sound clips
www.robotoid.com

MP3 info at Wikipedia
en.wikipedia.org/wiki/MP3

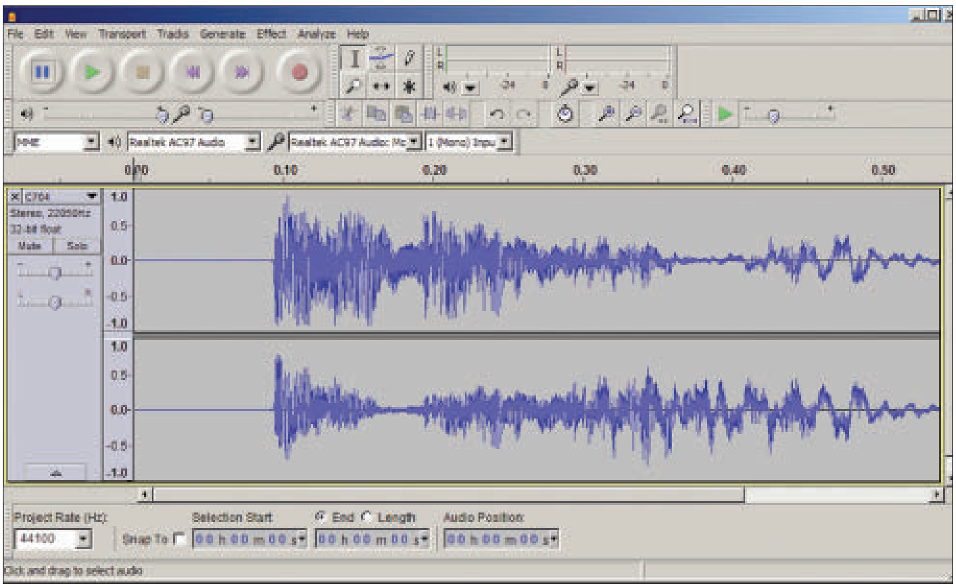
Encoding software
Audacity - **audacity.sourceforge.net**
Goldwave - **www.goldwave.com**

LISTING 1

```
#include <SPI.h>                                // SPI library (comes with Arduino)
#include <SdFat.h>                              // code.google.com/p/sdfatlib
#include <SdFatUtil.h>                          // Part of SdFat
#include <SFEMP3Shield.h>                       // See text

SFEMP3Shield MP3player;                         // MP3 player object

char trackName[] = "track000.mp3";              // Char array for filename

void setup() {
  Serial.begin(9600);                           // Serial Monitor for debugging

  MP3player.begin();                            // Start shield
  MP3player.SetVolume(10, 10);                  // Set volume to about 75%

  for(int i = 0; i<10; i++) {                   // Iterate files 000 to 009
    sprintf(trackName, "track%03d.mp3", i);     // Format: track###.mp3
    MP3player.playMP3(trackName);               // Play track
    Serial.println(trackName);                  // Show track name in Serial Monitor
    while(MP3player.isPlaying());               // Wait until track is done
  }
}

void loop() {
  // Empty loop
}
```

standpoint constant bit rates are easier to work with. Use a constant bit rate whenever you can. The audio clips for this project — see **Sources** for where to find samples you can use — were encoded at 160 kbps constant bit rate. Feel free to experiment with other variations.

Table 1 compares several commonly used MP3 bit rates. Compression ratio, file sizes, and quality comparisons are approximate.



Making MP3 Audio Clips

You can make your own MP3 clips for your robot projects, either by recording them using a microphone or musical instrument, or by converting an already existing audio file. There are numerous free and paid software for creating MP3 files. Two favorites are Audacity and

FIGURE 3. Audacity is a popular open source software for recording, editing, and exporting sound files, including MP3 (requires a separate download of the LAME MP3 plug-in).

Using Arduino Libraries

The sketches in this article require the use of several third-party libraries that are not included with the Arduino IDE software. The libraries are available for download from the locations listed below, but for your convenience they are also provided as part of the downloads available on the *SERVO* website; refer to the link at the start of this article.

The libraries must be moved to your Arduino sketchbook libraries folder. On Windows, this is found at Documents\Arduino\libraries (or My Documents\Arduino\libraries when using Windows XP). If your sketchbook doesn't already have a libraries folder, you'll need to create one first.

The extra libraries are:

SdFat – **code.google.com/p/sdfatlib**. Reads the MP3 files on the micro-SD card attached to the SparkFun MP3 player shield.

SFEMP3Shield – **github.com/madsci1016/Sparkfun-MP3-Player-Shield-Arduino-Library**. Plays and manages MP3 clips. This library, by electrical engineer Bill Porter, makes using MP3 on the Arduino much easier.

IRremote – **arcfn.com/2009/08/multi-protocol-infrared-remote-library.html**. Handy library written by Ken Shirriff decodes infrared signals from an ordinary universal remote control. Note: As of this writing, the version of the IRremote library on Ken's site is not Arduino 1.0-ready. Use the version included with the sketches for this article.

Important! The Getting Started guide provided by SparkFun for the MP3 player shield talks about modifying the SdFat library to alter the Slave Select (SS) pin used by the Arduino to activate the micro-SD card. This step is not required as long as you use a current version of the SdFat library and the sketches provided with this article. The Slave Select pin is now an optional parameter when setting up the SdFat object.

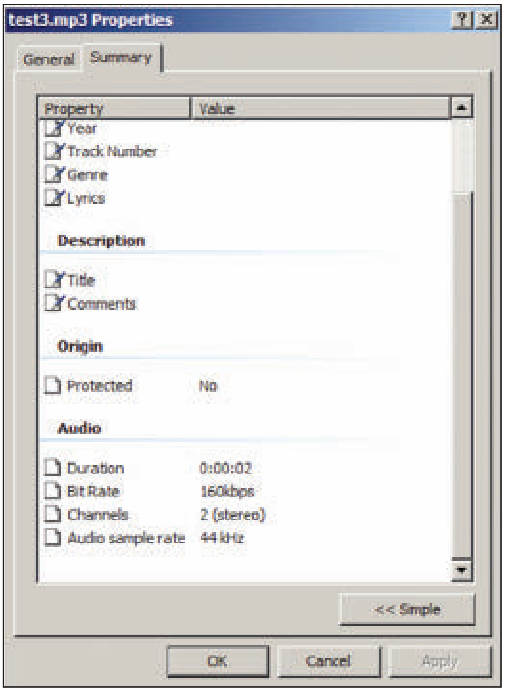


FIGURE 4. Verify proper settings in the MP3 files you use by checking their properties.

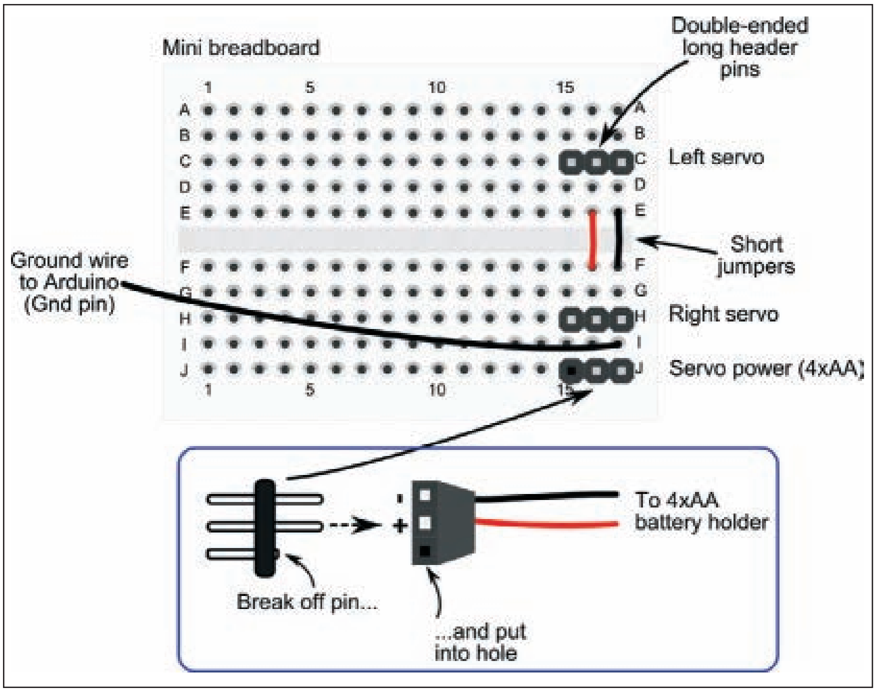


FIGURE 5. Basic breadboard wiring diagram for the Musicbot. The two continuous rotation servos that move the bot are powered from their own 4xAA battery pack. The servo power supply and the Arduino share a common ground. Use double-ended long header pins to attach the servos and other cabling to the breadboard.

Goldwave. Audacity is free. Goldwave is paid software; either an outright lifetime purchase, or a less expensive annual license.

Both allow you to record, edit, and export clips to MP3 format. A slate of effects let you modify the sound — add echo and reverb, create a robotic voice, or modify the pitch and speed.

Here are the steps for processing an existing sound file to MP3 in Audacity. It's not complicated, but it does involve a number of commands that you must select in the proper order.

Note: Before you can create an MP3 file in Audacity, you must first download a copy of the LAME MP3 encoder program. Instructions are provided in the Audacity help. (In Audacity, choose Help->Quick Help, then in the main page click on Index; next, find and click on LAME MP3.)

1. Open an existing WAV or other sound clip (see **Figure 3**). If prompted, choose the option "Make a copy of the files before editing."
2. Examine the Project Rate setting in the lower left corner. Set it to 44100, if it's not already.
3. Choose File->Export. In the Export File dialog box, choose MP3 Files in the Save as type list.
4. Click the Options button, then select the following:
   Bit Rate Mode: Constant
   Quality: 160 kbps
   Channel Mode: Joint Stereo
   then click OK.
5. Provide a name for the clip. The Arduino MP3 player code in this month's installment assumes a specific filename format — track*nnn*.mp3 — where *nnn* is a three-digit number starting at 000. For

FIGURE 6. A self-contained capsule speaker incorporates its own amplifier, speaker element, and rechargeable battery.

Bit Rate	Compression Ratio	File Size Per Minute*	Quality Similar To
64 kbps	22:1	480K	AM Radio
96 kbps	15:1	720K	FM Radio
128 kbps	11:1	1 MB	TV Sound
160 kbps	9:1	1.2 MB	Tabletop CD
192 kbps	7:1	1.4 MB	Hi-Fi CD
*Calculated using this standard formula: (Bit Rate * 60) / 8.			

Table 1.

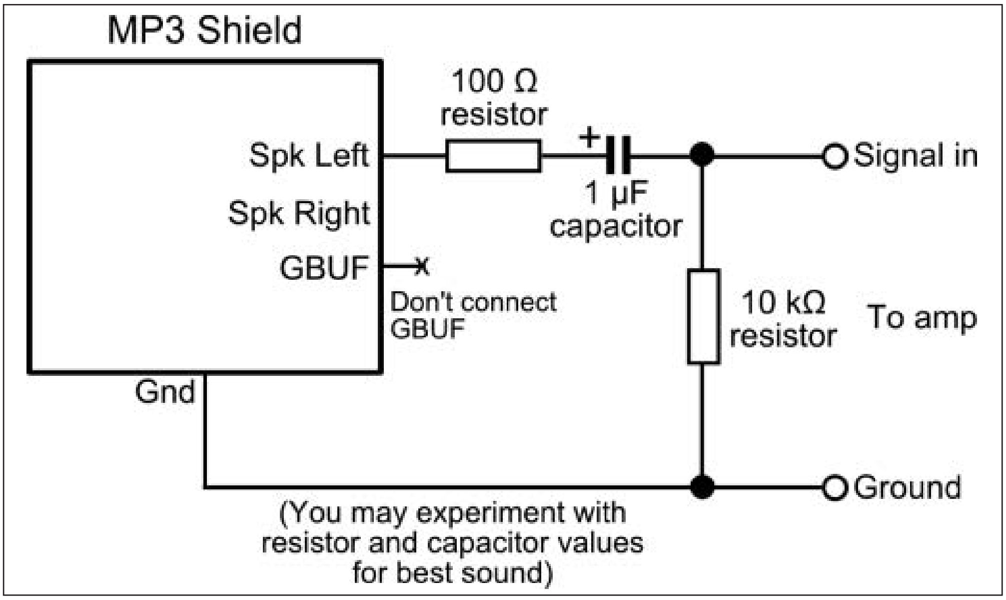


FIGURE 7. A recommended interface circuit for the output of the MP3 player shield. It's important that you don't connect the GBUF line as a ground. You can experiment with the capacitor and resistor values.

example, track000.mp3, track001.mp3, and so forth. (You can use other filenames in the sketches you write, but in order to be used by the MP3 Player shield they must be in standard 8+3 filename format.)

6. Click the Save button. Fill in the metadata fields if you wish to use them. You can leave them blank; the metadata has no effect over the actual playing of the clip.

If the sound you want to export doesn't already exist, you can record your own directly in Audacity. Connect a microphone to your PC and choose 44100 for the Project Rate. Click on the Record button, then talk, sing, play an instrument, make rude noises, whatever. When you're done, click the Stop button to terminate recording. Follow Steps 3 through 6 above.

Building and Using the MP3 Player Hardware

As noted, the Musicbot uses an Arduino Uno and a SparkFun MP3 player shield. The shield plugs directly into the Arduino, making all the necessary electrical connections between the two. The shield uses the majority of pins on the Arduino, but there's still enough free for experimentation. For initial testing of the shield, use a pair of headphones to audition the MP3 sound.

Before placing the shield on the Arduino, however, first load the sketch in **Listing 1** to your Arduino (this sketch and the others in this article are for the Arduino 1.0 software). Doing this as a first step ensures there are no I/O pin settings from a previous sketch that could damage the shield. The MP3

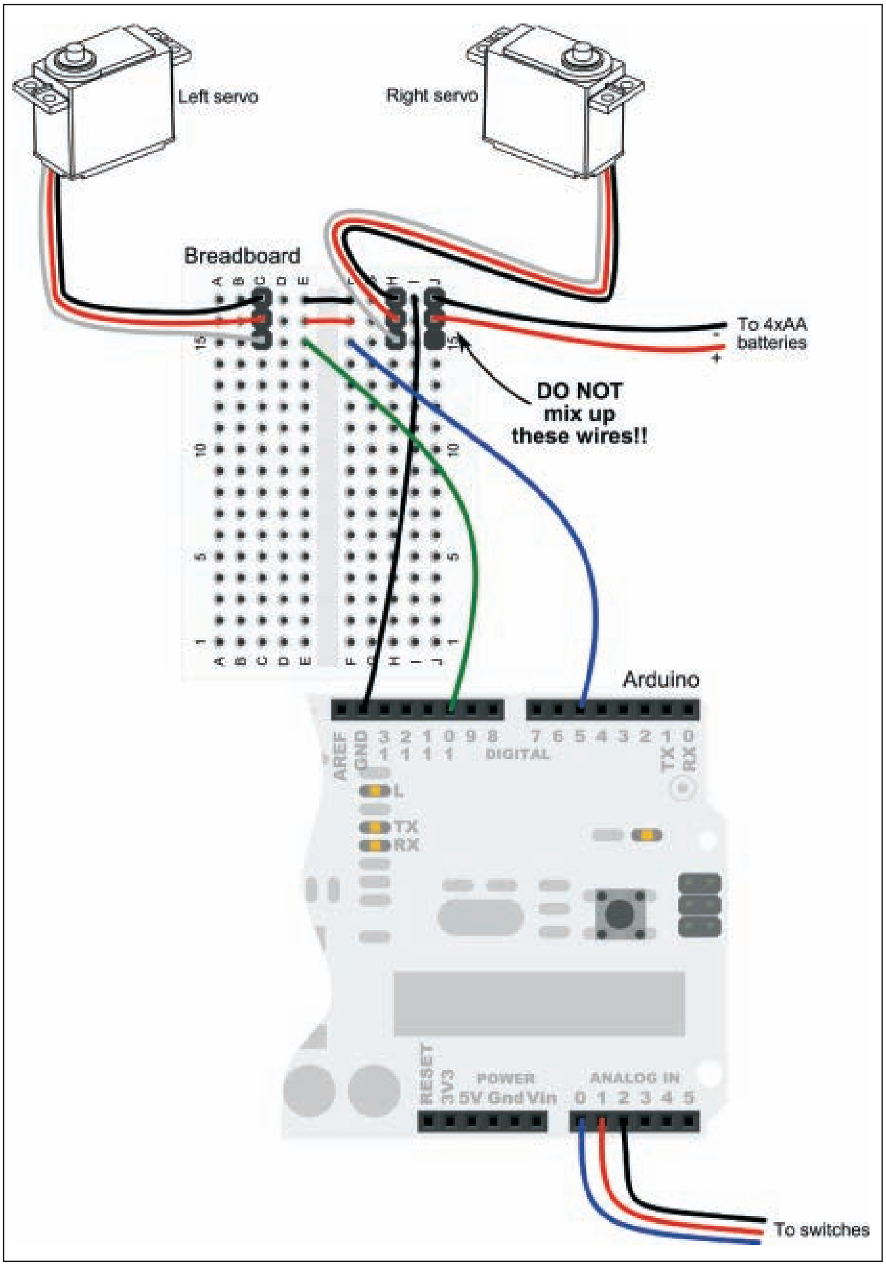


FIGURE 8. Full wiring for the servos, bumper switches, and Arduino. Be sure not to cross up the power wiring for the servos, or they may be instantly damaged.

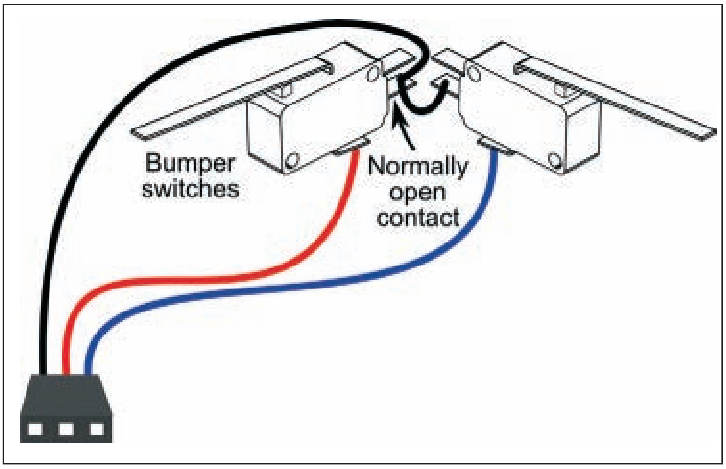


FIGURE 9. Connect the twin bumper switches as shown. Many leaf switches have both normally open and normally closed contacts; you want to use the normally open contacts.

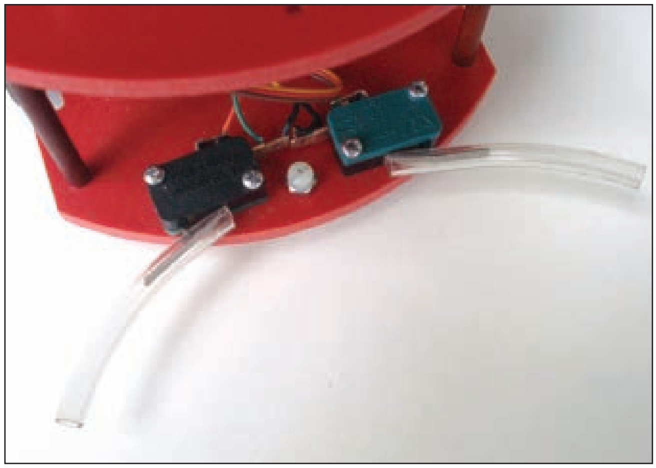


FIGURE 10. Use flexible aquarium tubing to extend the reach of the leaf switches. You'll need to gently bend out the leafs to make room for the thickness of the tube.

functionality is provided by third-party libraries that must be added to your Arduino installation. See the *Using Arduino Libraries* **sidebar** for details.

Once the sketch is uploaded, carefully attach the shield to the Arduino. Copy up to 10 MP3 clips to a micro-SD card — the card must be formatted for FAT-16 or FAT-32 (FAT-16 preferred), and the filenames should be in the format track*nnn*.mp3, where nnn is a number starting with 000. Insert the card into the card reader on the shield. Press the Reset button on the shield; the MP3 files you loaded onto the micro-SD card should play one right after the other, in order of their numbering.

Oops — don't hear anything? Check the headphone connection, and make sure that the micro-SD card is plugged in completely and correctly. Double-check the naming of the files; they must be in the track*nnn*.mp3 format noted above. Verify that the clips have been exported properly. This can be done, for example, in Windows by right-clicking on the file and selecting Properties. Click the Summary tab, and if not already shown, click the Advanced button to show the full list of file properties. (See **Figure 4** for details.) Check that

LISTING 2

```
#include <SPI.h>
#include <SdFat.h>
#include <SdFatUtil.h>
#include <SFEMP3Shield.h>
#include <Servo.h>                 // Servo libary(comes with Arduino)

Servo servoLeft;                   // Define left servo
Servo servoRight;                  // Define right servo
SFEMP3Shield MP3player;

byte lastPlayed = 255;

void setup() {

  Serial.begin(9600);

  servoLeft.attach(10);            // Left servo to pin D10
  servoRight.attach(5);            // Right sevo to pin D5

  pinMode(A0, INPUT);              // Set pin A0 to input
  pinMode(A1, INPUT);              // Set pin A1 to input
  pinMode(A2, OUTPUT);             // Set pin A2 to output
  digitalWrite(A0, HIGH);          // Activate A0 pullup resistor
  digitalWrite(A1, HIGH);          // Activate A1 pullup resistor
  digitalWrite(A2, LOW);           // Make low for switch ground

  MP3player.begin();
  MP3player.SetVolume(10, 10);
  Serial.println ("MP3 started");

}

void loop() {

  if (digitalRead(A0) == 0) {              // If right bumper hit
    triggerTrack(A1, "track001.mp3");      // Play this track
    reverse();                             // Reverse robot for 400 ms
    delay(400);
    spinLeft();                            // Spin left
    while(MP3player.isPlaying()) {         // Wait while clip is played
      delay(1);
    }
    forward();                             // Go forward again
    Serial.println("pbRight");             // Display status
  }

  if (digitalRead(A1) == 0) {              // If left bumper hit
    triggerTrack(A0, "track002.mp3");
```

Continued

LISTING 2 – continued

```
    reverse();
    delay(400);
    spinRight();                    // Spin right
    while(MP3player.isPlaying()) {
      delay(1);
    }
    forward();
    Serial.println("pbLeft");
  }
}

// Trigger Track and motion routines
void triggerTrack(byte pin, char* track) {
  // Test if an MP3 is playing, and trigger if not the same as before
  if (lastPlayed != pin && MP3player.isPlaying()) {
    MP3player.stopTrack();         // Stop track if another is playing
  }
  MP3player.playMP3(track);        // Play the new track
  lastPlayed = pin;                // Set this pin number as last played
}

// Motion routines for forward, reverse, turns, and stop
void forward() {
  servoLeft.write(180);
  servoRight.write(0);
}
void reverse() {
  servoLeft.write(0);
  servoRight.write(180);
}
void spinLeft() {
  servoLeft.write(0);
  servoRight.write(0);
}
void spinRight() {
  servoLeft.write(180);
  servoRight.write(180);
}
void stopRobot() {
  servoLeft.write(90);
  servoRight.write(90);
}
```

the Bit Rate says 160 kbps, and the Audio sample rate is 44 kHz.

Building the Musicbot

You can build your Musicbot using most any robot base. I've made mine with an ArdBotII base, available from my online store Budget Robotics. ArdBotII uses a pair of CNC-machined 7" x 5" PVC plastic decks for easy expansion, and is powered by two continuous rotation servo motors. **Figure 5** shows the wiring for the mini solderless breadboard; the board serves as a convenient means to connect the servo motors to the Arduino.

Use an Arduino Uno or compatible board, and mount the SparkFun MP3 player shield over it. Use a small capsule amplifier (see **Figure 6**) for sound output. Depending on the amplifier you use, it may not be directly compatible with the shield. **Figure 7** shows an interface circuit for providing AC coupling between the shield and the amplifier. This circuit is for one channel only; you need to duplicate the circuit for stereo. The values for the resistors and capacitors are typical. Experiment.

Important! As shown in the diagram, do not connect the GBUF pin on the side of the shield to anything. Use the L and R pins, plus the shield's ground. Leave the GBUF pin — it provides a 1.23V reference voltage for use with headphones — unconnected.

The ArdBotII base runs under two separate battery supplies: a 9V battery connects to the power-in plug on the Arduino, while a 4xAA battery holder provides the 4.8V or 6V juice for the servos. Servo power is routed to the solderless breadboard and shared between the two servos. Be *absolutely sure* not to swap the red and black (+ and –) battery

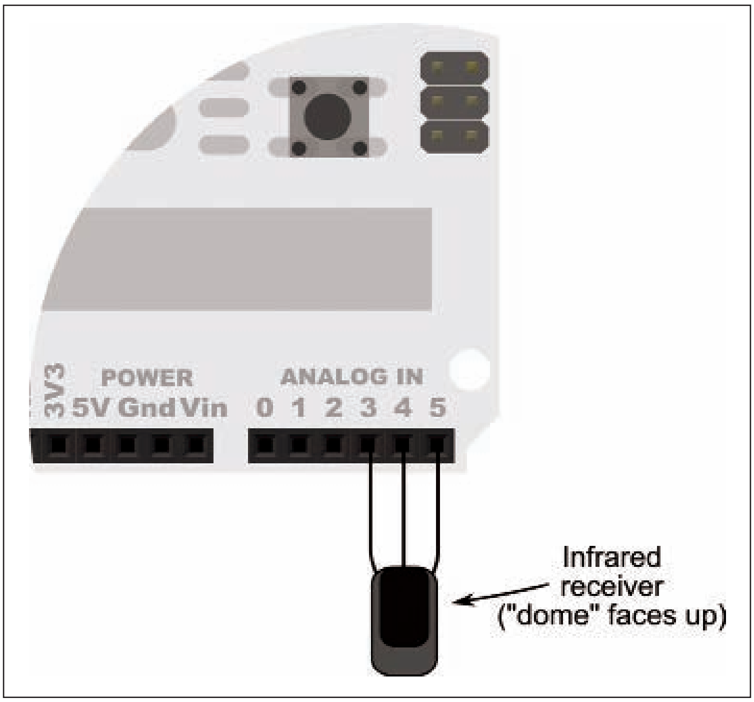


FIGURE 11. Orientation for the Vishay TSOP38238 infrared receiver module. You may use most any IR receiver tuned to operate at 38 kHz. Be sure to check its datasheet to verify the pinout connections.

leads, or the servos may be irreparably damaged. **Figure 8** shows the main wiring diagram for the Arduino, servos, servo power, and switches.

Note the ground connection between the servo battery supply and the Arduino. This is necessary for proper operation, as it provides a common ground between all subsystems.

Figure 9 shows how to attach a pair of standard leaf switches to the Musicbot. The switches go on the front of the robot (see **Figure 10**) and provide an easy way to detect collisions with objects on either side of the bot. The levers on the leaf switches can be extended by using 1/4" (inside diameter) rubber tubing — the kind used in home aquariums. Cut the tube to about 3", and stuff over the lever. *Gently* bend out the lever so that the tube actuates the switch when it's pressed in.

Listing 2 demonstrates how to control the Musicbot using the two leaf switches as bumpers. Whenever one of the bumpers is activated, the robot will briefly reverse then spin to the right or left, depending on which switch made contact.

Load up two short MP3 clips onto your micro-SD card. Name them track000.mp3 and track001.mp3. The ideal length of each clip is 1.5 to two seconds. The length of the clip is important because the Musicbot will spin for as long as the clip plays. Once the clip is finished, the robot will go forward again.

Adding Infrared Remote Control

Extend the sound-playing features of the Musicbot by adding a self-contained infrared receiver module. These work by detecting the high frequency pulses of an infrared remote control, and decoding the signal into a serial data train. Each button on the remote provides a

LISTING 3

```
#include <SPI.h>
#include <SdFat.h>
#include <SdFatUtil.h>
#include <SFEMP3Shield.h>
#include <Servo.h>            // Servo libary(comes with Arduino)
#include <IRremote.h>         // Infrared remote library; see text

Servo servoLeft;              // Define left servo
Servo servoRight;             // Define right servo
SFEMP3Shield MP3player;

const int RECV_PIN = A5;      // Receiver input pin on A5
IRrecv irrecv(RECV_PIN);      // Define IR receiver object
decode_results results;

byte lastPlayed = 255;

void setup() {

  Serial.begin(9600);

  servoLeft.attach(10);       // Left servo to pin D10
  servoRight.attach(5);       // Right sevo to pin D5

  pinMode(A0, INPUT);         // Set pin A0 to input
  pinMode(A1, INPUT);         // Set pin A1 to input
  pinMode(A2, OUTPUT);        // Set pin A2 to output
  digitalWrite(A0, HIGH);     // Activate A0 pullup resistor
  digitalWrite(A1, HIGH);     // Activate A1 pullup resistor
  digitalWrite(A2, LOW);      // Make low for switch ground

  pinMode(A4, OUTPUT);        // IR power, ground pins
  pinMode(A3, OUTPUT);
  digitalWrite(A4, LOW);      // IR ground
  digitalWrite(A3, HIGH);     // IR power
  irrecv.enableIRIn();        // Start the receiver

  MP3player.begin();
  MP3player.SetVolume(10, 10);
  Serial.println ("MP3 started");

}

void loop() {

  // Make tracks 000 and 001 slightly longer (about 1 second)
  // Make the other clips shorter, 1/2 second or less
  if (digitalRead(A0) == 0) {           // If right bumper hit
    triggerTrack(A1, "track000.mp3");   // Play this track; triggers
                                        // as #14
    reverse();                          // Reverse robot for 400 ms
    delay(400);
    spinLeft();                         // Spin left
    while(MP3player.isPlaying()) {      // Wait while clip is played
      delay(1);
    }
    forward();                          // Go forward again
    Serial.println("pbRight");          // Display status
  }

  if (digitalRead(A1) == 0) {           // If left bumper hit
    triggerTrack(A0, "track001.mp3");   // Triggers as #15
    reverse();
    delay(400);
    spinRight();                        // Spin right
    while(MP3player.isPlaying()) {
      delay(1);
    }
    forward();
    Serial.println("pbLeft");
```

Continued

LISTING 3 – continued

```
   }

    if (irrecv.decode(&results)) {  // If valid value was received
      switch (results.value) {      // Match button against Sony TV codes
        case 0x10:
          Serial.println("1");
          break;
        case 0x810:
          triggerTrack(2, "track002.mp3");
          forward();
          Serial.println("2");
          break;
        case 0x410:
          Serial.println("3");
          break;
        case 0xC10:
          triggerTrack(5, "track003.mp3");
          spinLeft();
          Serial.println("4");
          break;
        case 0x210:
          triggerTrack(5, "track004.mp3");
          stopRobot();
          Serial.println("5");
          break;
        case 0xA10:
          triggerTrack(5, "track005.mp3");
          spinRight();
          Serial.println("6");
          break;
        case 0x610:
          Serial.println("7");
          break;
        case 0xE10:
          triggerTrack(8, "track006.mp3");
          reverse();
          Serial.println("8");
          break;
        case 0x110:
          Serial.println("9");
          break;
      }
     irrecv.resume();         // Receive the next value
     delay(10);               // 10ms delay
   }

}

// Insert Trigger Track and Motion Routines from Listing 2
```

different sequence of data. The Arduino can examine this data and determine which button on the remote is being pressed. The IR decoding functionality is provided by a third-party library that must be added to your Arduino installation. See the *Using Arduino Libraries* **sidebar** for details.

I'm using a TSOP38238 infrared module from Vishay that is "tuned" to receive IR signals transmitted at 38 kHz; see **Sources** for details on resellers of this component. Many other IR receivers will also work, such as #350-00014 from Parallax — as long as it is intended for 38 kHz operation and shares the same pinout.

Connect the IR module as shown in **Figure 11**. Bend the leads at 90 degrees as illustrated. Be sure to properly orient the module, being careful not to reverse the leads in the Arduino pin sockets. Program a universe infrared remote to a Sony TV code — you may need to try several Sony TV codes until you find one that works.

Upload the sketch in **Listing 3** to the Arduino. When uploading is complete, open the Serial Monitor window (keep the USB cable to the Arduino plugged in). For this test, you can unplug the batteries for the servos. Press the number buttons on the remote, and note the responses in the window. As you press the number buttons, you should see the numbers echoed in the Serial Monitor window.

Next, load seven MP3 clips — named track000.mp3 to track006.mp3 — onto the micro-SD card. As in **Listing 2**, the first two files should be from 1.5 to two seconds in length; the remaining can be shorter — half a second is fine.

Unplug the USB cable, reattach the servo power, and reset the Arduino. Use the remote to operate the robot:

Button	**Function**
1	Forward
4	Spin Left
5	Stop
6	Spin Right
8	Reverse

You should hear your MP3 clips play at each button press. As in **Listing 2**, the clips named track000.mp3 and track001.mp3 play when the leaf switches are activated.

Making Your Own Sound Co-Processor

Generating various kinds of sounds and effects can consume not only I/O pins, but precious processing time. The more pins and processing that are devoted to generating sound and music, the less your Arduino has for doing its robotic tasks.

In the next and final installment of Sounding Off, you'll discover how to create your own programmable sound co-processor, freeing up your Arduino so it can handle more elaborate functions. With the sound co-processor, you'll be able to create and control all sorts of sound making using simple commands and just a couple of I/O pins. SV

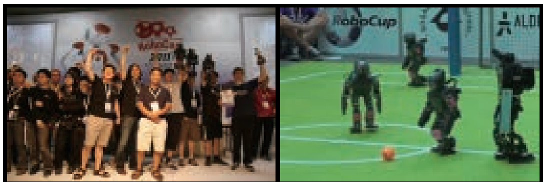
Open Platform Humanoid Project

DARwIn-OP

Open Platform
- All PC-based sources open (works on Linux Ubuntu)
- Sub board circuit & firmware open
- Physical specifications of H/W and 3D modeling data open
- Peripheral expansion (13×GP I/O&ADC)

High Performance
- Default walking speed : 24cm/sec (9.5 in/sec) – user modifiable gait
- Built-in PC : 1.6 GHz Intel Atom Z530 on-board 4GB SSD
- 2MP USB Camera, 3-axis gyro, 3-axis accelerometer

Easy Maintenance
- Single type actuators with durable metallic gears
- Modular Structure
- Part replacements can be easily made by user

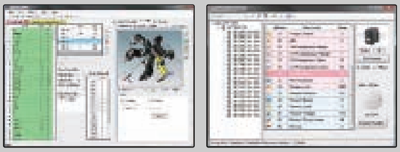


Team DARwIn: RoboCup 2011 World Champion

Specification (20 DOF)
- Weight : 2.9 kgs (6.39 lbs)
- Height : 454.5 mm (17.90 inches)
- For more information, please visit www.robotsource.org

DYNAMIXEL MX-Series

S/W Tools



MATLAB® python ROS
Microsoft Visual Studio LabVIEW eclipse
C/C++ C# Visual Basic Java

Contactless Absolute Encoder
12-bit resolution, 360˚ operating range with super-durability

User Programmable PID Control Gain
Precise and reliable PID control

Powerful Resource & Organized Design
Maxon motor, 32 bit controller, high communication speed
RX and EX series compatible dimension

Current Based Torque Control
Position and speed control with dual-loop current control
(**MX-64, MX-106 only**)

[USA]	E-mail : america@robotis.com	Tel : +1-949-333-3635
[KOREA]	E-mail : korea@robotis.com	Tel : +82-70-8671-2600
[JAPAN]	E-mail : japan@robotis.com	Tel : +81-3-4330-3660
[OTHERS]	E-mail : contactus2@robotis.com	Tel : +82-70-8671-2609

ROBOTIS
www.robotis.com

Building Maxwell: Mechanics, Electronics

Part 2

by Michael Ferguson

Last month, I briefly introduced the new MX-series of Robotis Dynamixel servos and how these servos can help in building a mobile manipulator. This article will take a step back through time to show the development and evolution of the mechanical and electrical side of my mobile manipulator — a robot called Maxwell.

FIGURE 1. Maxwell.

Motivation

There were a number of design goals with the development of Maxwell. First and foremost, I very much wanted to avoid the hassle of setting up a test environment for my robot (as I had long needed to do when building smaller competition robots). I needed a robust mobile manipulator that could operate in a typical home environment. I decided that I didn't particularly want to have to work mainly on things on the floor, and so the arm was targeted at a table top height. As I was frequently transporting my robot to various demos, it really needed to be easy to carry around and/or ship. The goals became:

- To use as much open source hardware and software as possible.
- To be low cost, by using off-the-shelf components wherever possible. Custom parts are mostly laser-cut ABS sheet stock.
- To be human scale, without sacrificing portability.

• To have robust onboard sensory/computation.

From this set of requirements, I developed a fairly straightforward base and torso, since I generally prefer to be writing software rather than debugging hardware or mechanical problems.

Evolution

People often ask me "Which version of Maxwell is this?" The truth is, Maxwell has slowly evolved over the past year and a half, and has undergone a number of revisions. The original build of Maxwell occurred in January 2011, with some minor updates over the following month. The version shown in **Figure 2** used a mix of AX-12 and RX-64 servos, and had no vertical movement for the arm.

In late 2011, Maxwell got a new head, replacing the large Microsoft Kinect with a much smaller Asus Xtion sensor. This greatly improved the performance of the neck servos. At this time, I also added a vertical linear actuator for the arm so that it could reach the floor, as well as tabletops. Unfortunately, the portability was sacrificed because Maxwell no longer fits in his handy pelican case for transport (at some point, I hope to remedy that). In early 2012, I upgraded the RX-64 servos to new MX-64 servos which have a much higher resolution encoder, as well as being designed for my 12V power system, which also allows me to use a single bus for all the servos. Maxwell, as he stands today, is shown in **Figure 1**.

The entire robot was designed in Autodesk Inventor, allowing parts to quickly be exported for laser cutting, and avoiding costly revisions that might have been caused by parts not fitting together. **Figure 3** shows the robot as rendered in Inventor.

The Base

The mobile base is fairly simple but very robust. It has a 16" x 16" footprint and is constructed of mostly 3/16" thick

Parts List	
Frame materials	Laser-cut ABS, mostly 1/8" and 3/16" 1"x1" 8020 aluminum framing
Drivetrain	(2) Zagros Robotics REXC motors (2) 4-7/8" BaneBots wheels and hubs (1) 3" caster
Arm	(4) MX-64 servos; (1) AX-12 servo (1) 20" linear actuator
Gripper	(2) AX-12 servos; Bioloid brackets for fingers
Battery	12V sealed lead-acid, 8 Ah
Sensors	(1) Head-mounted ASUS Xtion Pro Live (similar to Microsoft Kinect) mounted on a pan-tilt neck. (1) Hokoyu URG-04LX-UG01 laser
Main Processing	(1) Standard laptop; fits on back deck
Motion Control	(1) ArbotiX w/ dual 30A motor drivers

FIGURE 2. Maxwell, circa March 2011.

FIGURE 3. Maxwell as designed in Autodesk Inventor.

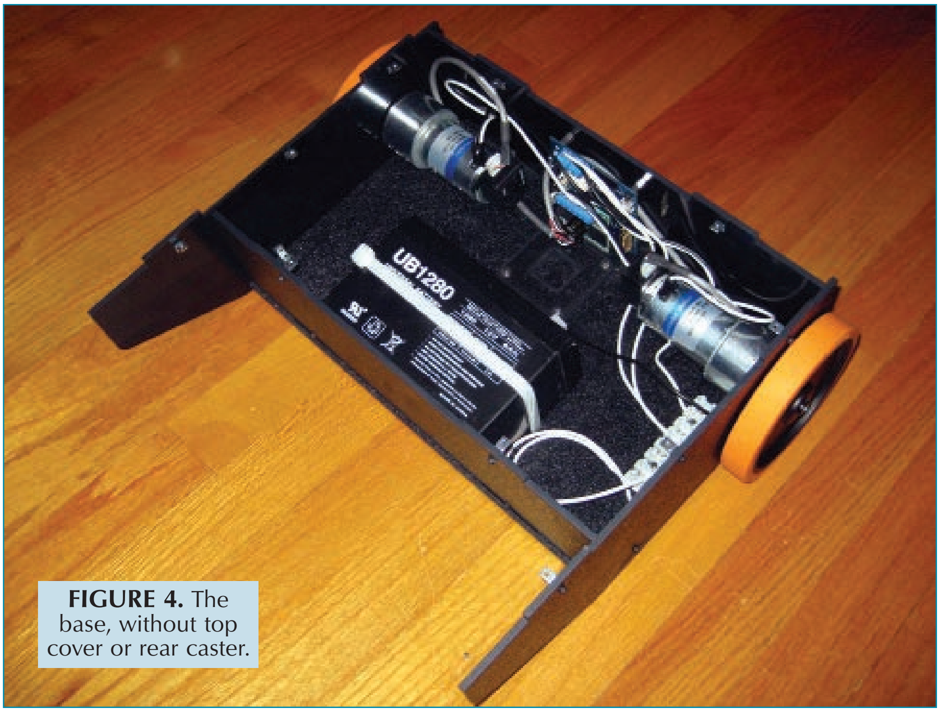


FIGURE 4. The base, without top cover or rear caster.

laser-cut ABS. The pieces go together using a collection of tabs and slots, and are held in place using small #4-40 corner brackets (Digi-Key P/N 621K-ND). This construction method has proven to be quite reliable on Maxwell, as well as in other projects.

The base uses a differential drive setup with two DC gearhead motors from Zagros Robotics (**www.zagrosrobotics.com**) which can power the robot to speeds of up to 2 ft/sec. The BaneBots (**www.banebots.com**) wheels provide a sturdy grounding, with lots of grip. I had to drill out the 6 mm hubs on the wheels to attach them properly to the 1/4" shafts on the motors, but other than this quick modification, the entire base can be assembled with just a set of hex keys. The battery is tie-strapped down, and held in place by the same piece of ABS that forms a socket for the 8020 column to insert into. Wire routing for the power switch and a charger socket is done using a European-style terminal strip and 14 ga wire.

The base design allows a full-size (14.1" screen) laptop to fit on the back deck. The computer interfaces to I/O through a Vanadium Labs ArbotiX RoboController which is tucked in the front of the base. The ArbotiX is used as a co-processor — connected over USB — which handles all real time operations and passes serial commands between the PC and the arm/neck servos, as well as implementing closed-loop PID control of the mobile base. The base is shown in **Figure 4** without the top cover or caster (which attaches to the top cover).

The Torso and Head

Maxwell's torso is very lightweight, constructed of three pieces of 1" x 1" 8020 extruded aluminum. The 8020 offers an easy way to connect everything together and to vary the height of the neck. Maxwell's overall height is about 5 ft tall.

Three pieces of 8020 were used with joining plates, so that Maxwell could easily be broken down for shipment or transport. By loosening only six screws, the entire robot can be broken down and fit into a Pelican case. **Figure 5** shows how the three pieces are put into the case. A layer of protective foam then covers the components and the base is put on top. This configuration has been successfully shipped across the country several times when traveling to RoboGames and other events.

The head is composed of two Dynamixel AX-12 servos, and allows the Asus Xtion RGBD camera to look all around the robot. This sensor is very much like the Microsoft Kinect, but weighs much less and is much smaller. Mounted on the back of the torso is a big red button. This is the emergency stop which cuts power to all the servos and the base motors. When building a robot this large, an emergency stop is always a good idea

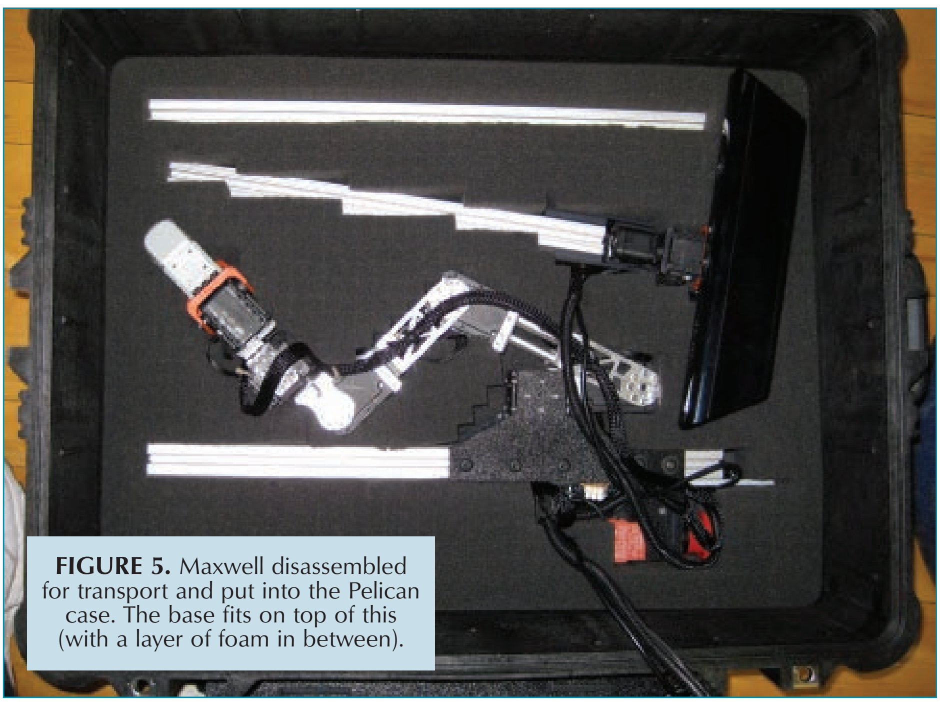


FIGURE 5. Maxwell disassembled for transport and put into the Pelican case. The base fits on top of this (with a layer of foam in between).

because your code will eventually have a bug, and the robot might damage itself or the things around it.

The Arm

Maxwell's arm consists of a five degree of freedom (DOF) manipulator with a two-servo gripper. The arm and gripper use four Dynamixel MX-64 servos and three AX-12 servos. The lowest four joints of the arm are MX-64s for extra strength. A lower cost AX-12 is used for the wrist rotation joint, and for both of the fingers on the gripper.

While a five DOF arm is not capable of as many varied grasps as a seven DOF arm (as found on the PR2), it is quite a bit less expensive to build since the arm itself weighs much less, and you have several fewer joints.

The most recent addition to the arm is the linear actuator which allows it to move up and down about 20". This allows Maxwell to reach things located on the floor, as well as greatly improving the overall workspace of what the gripper can reach. The linear actuator was purchased from Firgelli Automation, and has a speed of about 2"/sec. Unfortunately, it lacked any sort of positional feedback and I had to hack on an optical encoder. I hope to eventually replace the linear actuator with something more compact and quieter when in operation.

Conclusion

This article laid out the mechanical and electrical design, and evolution of my robot, Maxwell. Next month, we will delve into the software used to control Maxwell. While building Maxwell, I documented many of the smaller iterations on my blog which can be found at **http://showusyoursensors.com**. **SV**

www.servomagazine.com/index.php?/magazine/article/may2012_Ferguson

Discuss this article in the *SERVO Magazine* forums at http://forum.servomagazine.com

Roll Your Own Turtlebot

by Alan N. Federman

Discuss this article in the *SERVO Magazine* forums at http://forum.servomagazine.com.
www.servomagazine.com/index.php?/magazine/article/may2012_Federman

At the Homebrew Robotics Club meeting in February, at least six Turtlebots were rolling around the floor. This popular platform is an easy way to begin experimenting with cutting-edge robotics technology. If you have the funds, you can get a complete Willow Garage Turtlebot kit for $1,500. However, with a little ingenuity and some recycled parts, you can get a Turtlebot up and running for less.

Building a robot is a lot like making a cake. You need to assemble your ingredients, your pots and pans and utensils, and then follow a recipe to complete your masterpiece.

Here are your Turtlebot ingredients:

- An iRobot Roomba 5000 series robot vacuum cleaner with charger.
- At least one good battery for the Roomba (AllBattery has these for $40 and I got mine in one day!).
- A laptop or netbook (running Ubuntu or dual bootable into Unbuntu) for the robot (at least 2 GB of memory; 30 GB of storage; must have Wi-Fi).
- A Wi-Fi enabled (Ubuntu) laptop, workstation, or Android device for a controller.
- A USB to TTL dongle (a $15 Freeduino BUB will work).
- A Kinect 360 (used, these run $100 at Gamestop). An Asus can be substituted, and doesn't need a 12V supply.
- A 'DIN" cable from an old PS/2 style mouse or keyboard.
- A 12V voltage regulator, a couple of 1 µF capacitors, wire, solder, some crimp connectors, and a strip of headers, available at RadioShack.
- A 1/2" aluminum rod (3 ft), 4' by 4' 1/4" plywood, 8-32 all thread 1', a package of eight 3/4" 8-32 nuts and bolts, available at Home Depot.

Tools that you'll need:

Soldering iron, crimping tool, wire cutters, strippers, saws, drills, a small tapping set, etc. One tool I had access to that is not typical is a laser cutter, which made quick work of cutting out the support plates. It is possible to do this using a jig or band saw and a hand drill.

Preparing the Roomba

Preparing the Roomba 500 series is straightforward. First, remove and empty the dustbin, pry off the cover plate, and remove the carrying handle. The 500 series does not have a quick change battery. The battery is removed by flipping the unit on its back and removing four screws. You

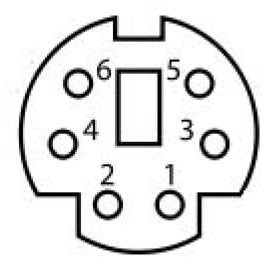


Figure 2. The male connector. Break out the center key by working it back and forth with a small screwdriver.

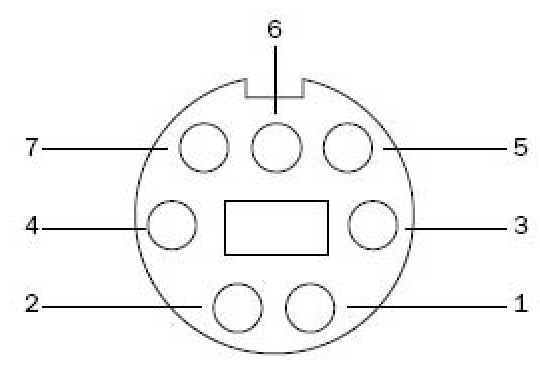


Figure 1. The female connector on the Roomba side (refer to table below for pin descriptions).

Pin	Name	Description
1	Vpwr	Roomba battery + (unregulated)
2	Vpwr	Roomba battery + (unregulated)
3	RXD	0 - 5V Serial input to Roomba
4	TXD	0 - 5V Serial output from Roomba
5	DD	Device Detect Input (active low) - used to wake up Roomba from sleep
6	GND	Roomba battery ground
7	GND	Roomba battery ground

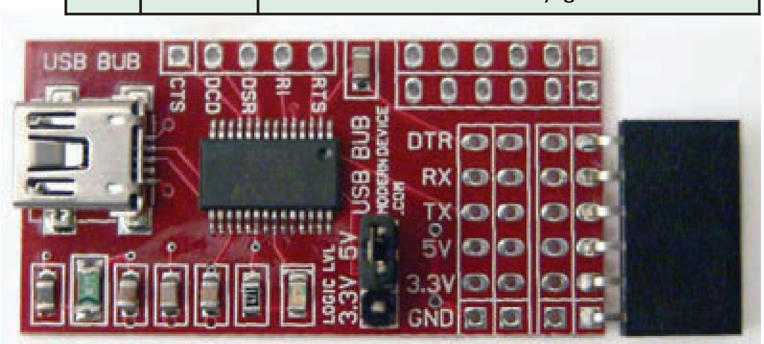


Figure 3. Old-style Freeduino USB BUB. The expansion space can be used for a 12V regulator.

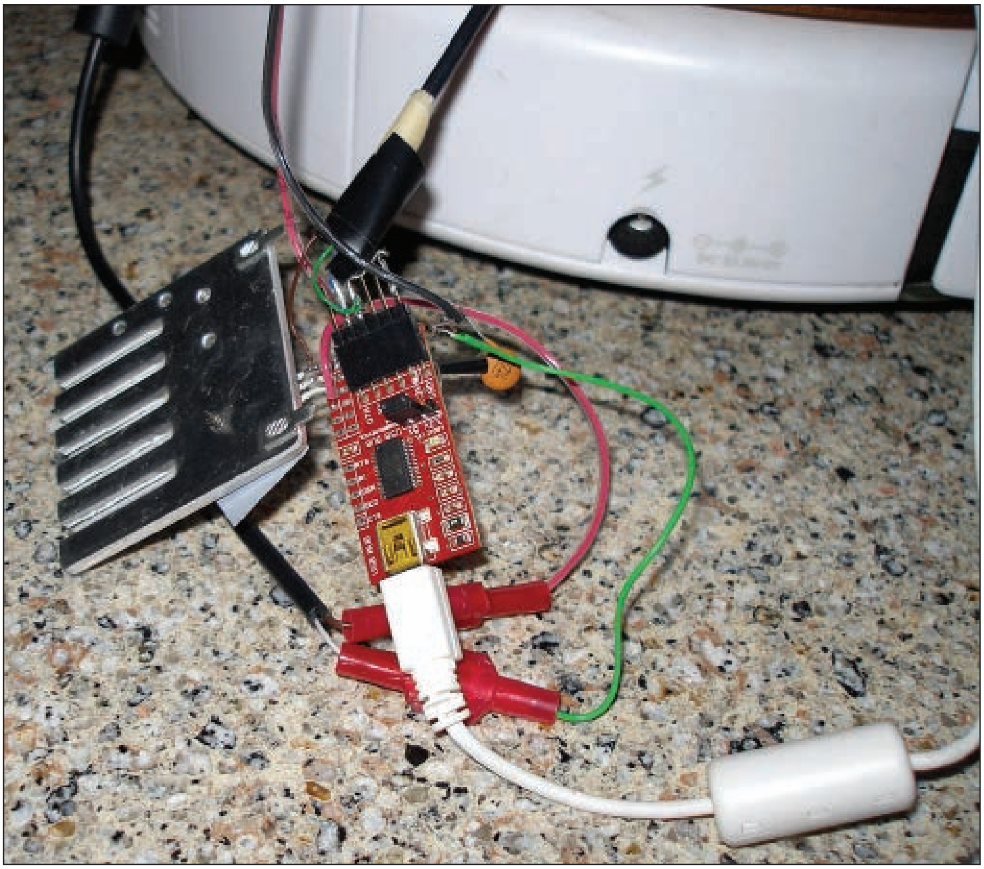


Figure 4. The interface wired up with the 12V regulator, attached to a heatsink.

can remove the brushes or leave them in — there is no reason to care other than for weight. You can always decide later to add some programming to your Turtlebot so it cleans up, as well as maps rooms. The cover plate can be pried up by hand or by using a thin screw driver.

I decided to keep the cover plate, but I cut away the portion covering the mini-DIN port to allow easy insertion of the interface cable. I attached the snap-in cover plate to the bottom of one of the mounting plates, and used two 1/2" wood screws to secure it to the Roomba.

The old battery in the Roomba would not hold a charge, so I ordered a new one. Having a good battery will make your life a lot easier.

Making the Interface

If you have a male eight-pin DIN cable from a Mac, you are in business. If you don't, a recycled cable from a PS/2 style mouse or keyboard will work. Just break the center square tab out by gently prying back and forth with a screw driver. You won't need the center pin.

Referring to **Figure 2**, if this is the male side of the PS/2 cable, note that 5 will be ground, and 1 and 2 will be unregulated power (up to 20 volts when on a charger). Shorting pins 1 or 2 to 5 will kill the battery; 3 is TXD; 4 is RXD.

To complete the interface, I wired DIN pin 5 to GND, pin 3 to TX, and pin 4 to RX. I also used the expansion area to hold a 12V (7805) regulator with two capacitors powered from the unregulated pin 1 (to provide 12V for the Kinect).

The 'DD' (pin 6) signal was not used and did not appear to be needed. I wasted a lot of time trying to get the MAX232 circuit I found on the Web to work. I finally gave up. The interface

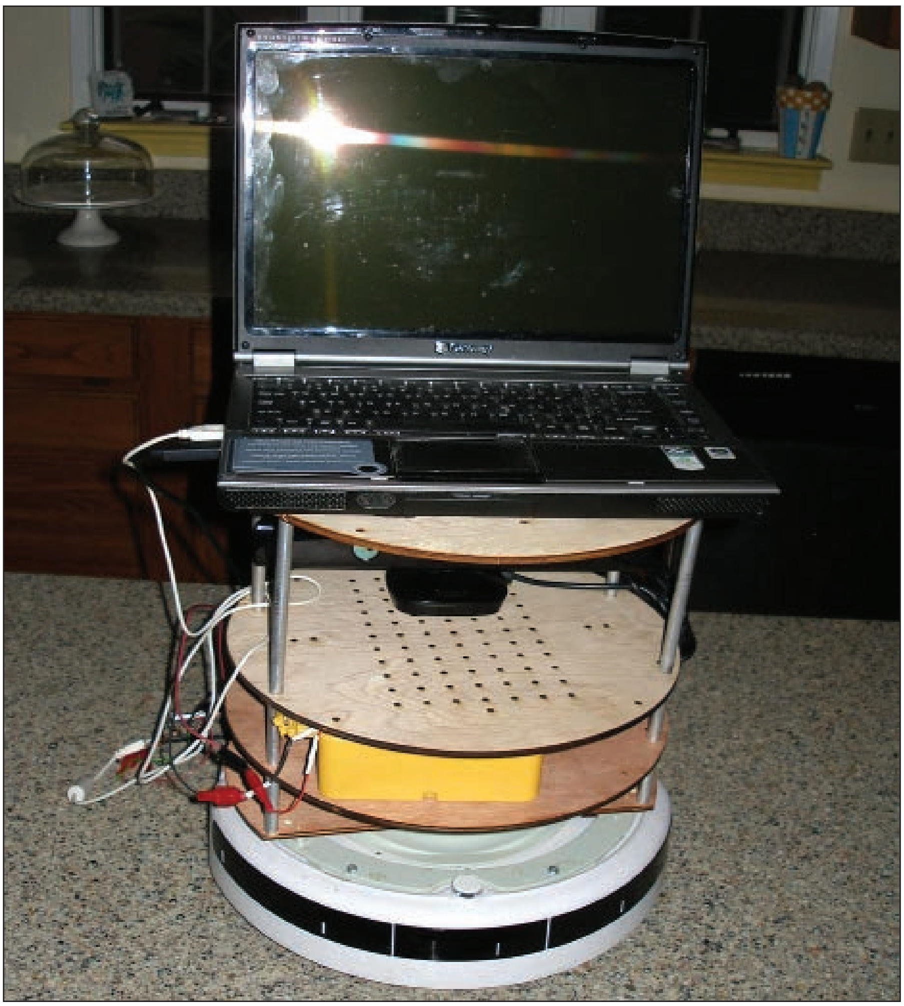


Figure 5. The assembled Turtlebot.

Figure 6. The assembled Turtlebot showing the Kinect and secondary battery.

worked the first time I tried it.

You can download the complete plans for a Turtlebot from **http://turtlebot.com/build**. I decided to laser-cut my own plates and build my own standoffs. The plates are for the Create base, and need slight modification to use with a Roomba 500 (**http://makeprojects.com/Project/Build-Your-Own-TurtleBot-2-Inch-Standoff/1328/1**).

Turtlebot Complete

This Turtlebot is running on the ROS Diamondback release. The default software baud rate needs to be changed in the setup-node.py file from 57600 to 11520. There are some significant differences between the Create base and the Roomba 500. Most importantly, there is no gyro chip in the Turtlebot-EU. Also, there is no automatic way to turn off the battery to the Kinect, so I am using a secondary battery to extend the runtime between charging. A good tip is to always keep the robot on the charger when not in use.

The latest version of ROS (Electric) will have some special drivers that reflect the differences between a Turtlebot using a Roomba versus a Create as a base. I am looking forward to upgrading. So far, I've been able to run the basic ROS Teleop, Dashboard, and Follow programs on my RYO_Turtlebot. **SV**

Acknowledgements

I'd Like to thank Melonee Wise and Mike Gregg for their generous donation of the Roomba 510, the Homebrew Robotics Club for that free one month loan of a Turtlebot, and TechShop. They should arrest me for loitering, but instead they actually let me teach classes there.

Using Advanced Sensors with VEX — Feel the Heat

By Daniel Ramirez

Discuss this article in the *SERVO Magazine* forums at **http://forum.servomagazine.com**.

Remember the movie comedy "The Nutty Professor" — a Dr. Jekyll and Mr. Hyde comedy spoof starring Jerry Lewis and Stella Stevens that came out in the late '50s (see **Figure 1**)? Scenes from the movie showed beakers full of colorful mixtures bubbling up in Bunsen burners, and spinning centrifuges all working together. Just think how great it would have been for the nutty professor if he had the advantage of automatic timer controls.

FIGURE 1. A still from the movie comedy "The Nutty Professor" — a Dr. Jekyll and Mr. Hyde spoof starring Jerry Lewis and Stella Stevens that came out in the late '50s.

Did you know that VEX Robotics kits could be used in chemistry or biology labs? With the timers, digital I/O, and analog I/O, we can control almost any kind of experiment that requires mixing at specific time intervals, temperature monitoring, or even measuring light levels. Using VEX components, we can make useful lab equipment such as motorized paddles to stir liquids or timers to control chemical processes. Instead of having to constantly watch an experiment, the VEX microcontroller can monitor it and record the data for you. Think of how much time you would save, so that you could move on to other experiments and learn so much more. In previous articles, we've talked about how to use the microcontroller to track the sun using astronomical equations. Now, it's time to introduce the VEX microcontroller to the chemistry, physics, or even the biology lab.

The VEX microcontroller provides the timers and electronic control needed for your experiments. It rivals older electronic controllers that are far more costly. Individual VEX microcontrollers can be purchased used on eBay at a lower cost.

A VEX microcontroller has the following features that make it ideal for use in the lab:

- Built-in timers that can measure durations in microseconds to days.
- Reacts very quickly to real time events triggering interrupts, using its internal timers.
- Responds by using its PWM (pulse width modulation) outputs to control motors, servos, relays, or solenoids.
- Can switch electronic devices connected to it on or off, depending on events or sensor readings using

the 16 digital I/Os.
- Senses various physical properties including temperature, humidity, voltages, currents, light levels, sound levels, etc., using the 16 analog inputs (ADC).

In this article, we will experiment with analog temperature sensors. Temperature sensors include thermocouples, thermistors, and solid-state devices. Temperature sensors are associated with the sense of touch and/or feeling.

We use temperature sensors every day to monitor the indoor/outdoor temperature; for weather applications; commercial appliances, industrial applications, and scientific applications. In HVAC, thermostats control the room temperature by measuring the ambient room levels and comparing those to the set temperature. This way, the furnace activates when the temperature is below the set point. In automation, factory process control and robotics are used everywhere. Even your laptop or PC uses a sensor to monitor the CPU temperature. You can see then how important this sensor is to us.

Types of Analog Temperature Sensors

There are various types of analog temperature sensors, including lab glass thermometers; these include biological sensors such as our own nerve cells near the surface of the skin and electronic temperature sensors including analog and digital kinds. One of the early analog temperature sensors was developed in 1821, by the German-Estonian physicist Thomas Johann Seebeck who discovered that when any conductor is subjected to a thermal gradient, it will generate a voltage. This is known as the thermoelectric effect or Seebeck effect used in thermocouples.

Thermocouples have many commercial and research uses including process control and factory automation. These kinds of sensors can also be used for consumer and commercial high temperature applications such as ovens, furnaces, and stoves. For more information on the thermoelectric effect, see the article relating to it on Wikipedia (**www.wikipedia.com**).

Type K or type J thermocouples (shown in **Figure 2A**) are made by welding two wires of dissimilar metals together; they measure high temperatures and are accurate to within ±1 degree Celsius. They do require signal conditioning using op-amps and a Wheatstone bridge, since they generate a very weak voltage. This requires an op-amp to amplify the signal to a range that our microcontroller can read using the 10-bit ADC. Thermocouples measure the temperature difference between two points, not the absolute temperature. In order to measure a single temperature, one of the junctions — normally the cold one — is maintained at a known reference temperature; the other junction is at the temperature to be sensed.

Thermistors shown in **Figure 2B** are another type of analog temperature sensor similar to a resistor but whose resistance varies significantly with temperature. They are low-cost sensors that can be used for high temperature measurements. Although I was able to use them with the VEX microcontroller, I found them to not be as accurate as the LM34 temperature sensor shown in **Figure 2C** .They are not recommended for these experiments since they are not as accurate, and are much harder to use.

The low-cost LM34 Fahrenheit temperature sensor is a solid-state device with three wire leads that return the temperature readings in millivolts proportional to the current ambient temperature surrounding the sensor. For example, a temperature reading of 650 indicates a temperature of 65.0 degrees Fahrenheit. These analog readings are digitized using the microcontroller's 10-bit analog-to-digital converter (ADC). LM34s are linear in nature and do not require calibration. Although they read temperature in degrees Fahrenheit to ±1 degree precision,

FIGURE 2A. A type K thermocouple used for scientific consumer and factory automation applications.

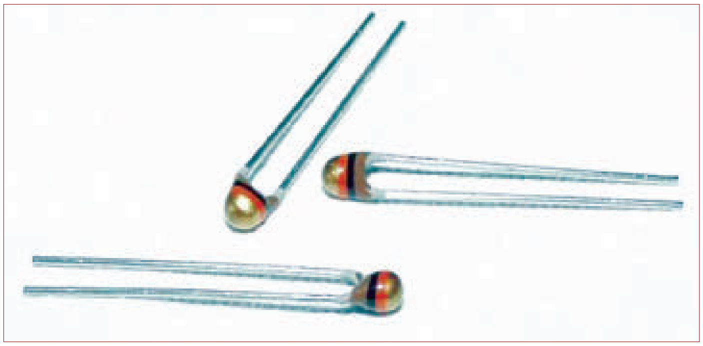


FIGURE 2B. This is what thermistors look like. These are low-cost sensors but a bit more difficult to use due to their non-linear nature.

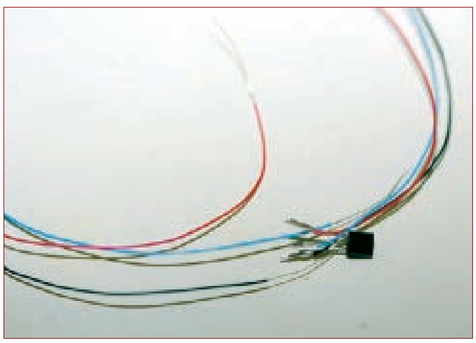


FIGURE 2C. The low-cost LM34 Fahrenheit temperature sensor is a solid-state device with three wire leads that return the temperature readings in millivolts proportional to the ambient temperature surrounding the sensor.

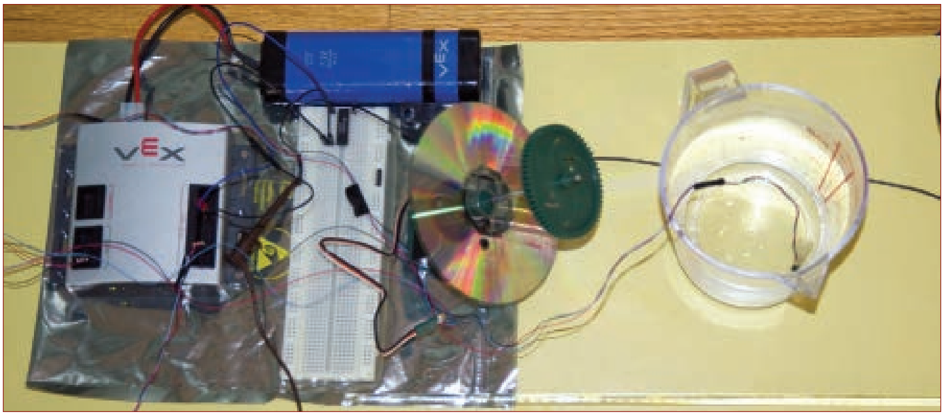


FIGURE 3. The setup for our experiment is shown here. Note the temperature sensor; in this case, a solid-state LM34 connected to analog I/O pin IO04 in the digital/analog I/O block using a three-pin .001 header or jumper wires.

they can easily be converted to degrees Celsius. This sensor is perfect for VEX scientific explorations. Although it is not currently sold by Innovations First, Inc. (IFI), it can be found from other sources at a very low cost (check All Electronics at **www.allelectronics.com**).

Through some simple experiments, you can see how using the VEX microcontroller in the chemistry or biology lab can help in automating more complicated experiments that would normally be too time-consuming to carry out by hand, given a very heavy class schedule. This application could also be a good starting point for a science fair project in Chemistry or Biology. We'll start by carrying out the temperature experiment described next which will show you how to use the LM34 with VEX.

Temperature Experiment: The Great Dead Sea Experiment

While it is rare to see temperature sensors used for VEX robotics applications, we see that this sensor is very important to our everyday lives (medical applications, heating, cooling, global warming, etc.). In chemistry, for example, a chemist can measure the temperature of various solutions and chemical reactions to monitor a particular experiment. Let's learn more about chemistry ourselves by carrying out this experiment.

Setup is shown in **Figure 3** and the necessary parts are listed in the Bill of Materials in **Table 1**. The LM34 (which will be immersed in a measuring cup) is protected from liquids and corrosive chemicals short-circuiting or corroding it by its coating of high temperature epoxy glue applied to the base of the sensor with the three wire leads. The mixing rod connected to the motor can also be protected in this manner.

To carry out this experiment, we need to wire the circuit shown in the **Figure 4** schematic. As you can see, the microcontroller and other VEX parts used here make it easy to carry out automated experiments. Interfacing analog temperature sensors to the microcontroller (such as the LM34) is a simple matter of connecting it to one of the 16 analog I/O pins or digital/analog I/O block using a three-pin .001 header or jumper wires, so that we can read it using one of the 10-bit analog-to-digital inputs; in this case, IO04. When building the circuit, make sure that you connect the correct power, ground, and signal wires to the LM34 as shown in the schematic. In order to reduce noise and get more accurate readings, we could connect up to 16 LM34 sensors and average the readings across all of the sensors.

To begin this experiment, fill a medium sized beaker or measuring cup with two cups (473 ml) of warm water at 122 degrees Fahrenheit or 50 degrees Celsius as measured from a calibrated lab thermometer from a sink faucet. Immerse the epoxy sealed LM34 into the measuring cup full of warm water and immediately run the Easy C application to start collecting readings to a text file using the text capture on the Easy C terminal. Run the Easy C application until the water temperature has dropped to room temperature. Verify these temperatures using a calibrated lab thermometer.

Next, we monitor the temperature of the water at 20 second intervals until the water reaches room temperature. Again, we collect this data using the Easy C debug window and save these results. Collect the data over a period of time until the water temperature has cooled to room temperature as measured by the lab thermometer.

TABLE 1. Bill of Materials needed for our experiment.

ITEM	QTY	DESCRIPTION	SOURCE
1	1	VEX microcontroller	Innovation First, Inc. (IFI) **www.vexforum.com**
2	1	VEX 7.2 volt battery	Innovation First, Inc. **www.vexforum.com**
3	1	Wire-wrap cable	RadioShack **www.radioshack.com**
4	1	*VEX motor	Innovation First, Inc. **www.vexforum.com**
5	1	*VEX pushbutton	Innovation First, Inc. **www.vexforum.com**
6	1	Medium size VEX gear	Innovation First, Inc. **www.vexforum.com**
7	1	LM34 temperature sensor	All Electronics **www.allelectronics.com**
8	1	Large glass beaker or measuring cup	Local grocery store or chemistry lab
9	1	Salt (NaCl)	Kitchen
10	1	Calibrated lab thermometer	Chemistry lab or local drug store

*Items are optional.

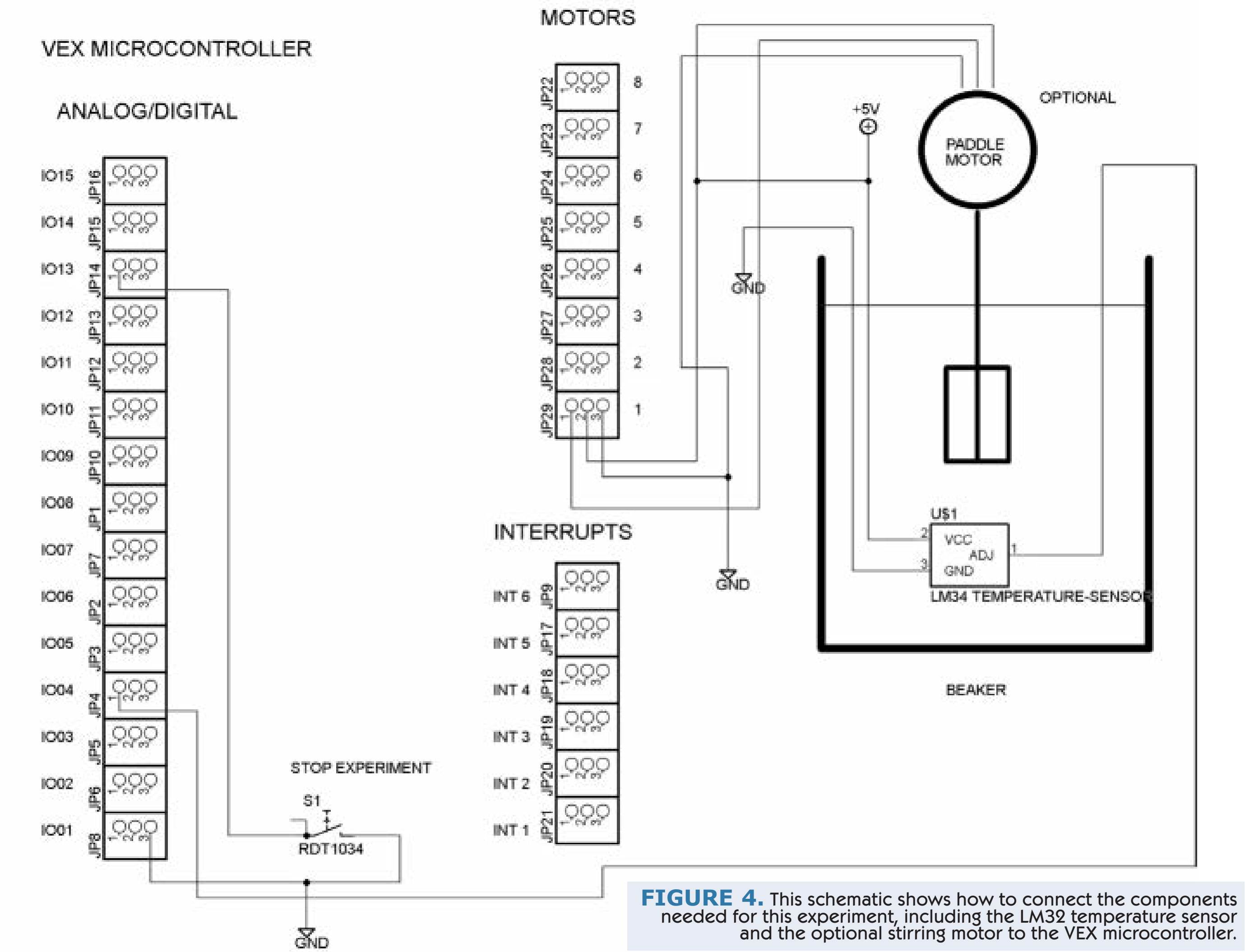


FIGURE 4. This schematic shows how to connect the components needed for this experiment, including the LM32 temperature sensor and the optional stirring motor to the VEX microcontroller.

We will now repeat the same experiment described above, but dissolve one tablespoon of salt (or sodium chloride) in the warm water [H20 + NaCl]. Try to keep the starting and ending temperatures the same. Record temperature readings captured by the Easy C terminal to another file, since we will use both data files for analysis.

Finally, use the optional motor with a plastic gear or paddle attached (which is described next) to stir the pure water and salt water at 20 second intervals. Again, we use the microcontroller to monitor the temperature of the water at these 20 second intervals until the water reaches room temperature. Collect this data using the Easy C debug window and save the results.

Safety Warning: Use only warm water from the sink so as to avoid getting scalded. Wear goggles for this experiment. Place a splash guard over the measuring cup. Don't boil the water on the stove to heat it; just use warm water from the faucet.

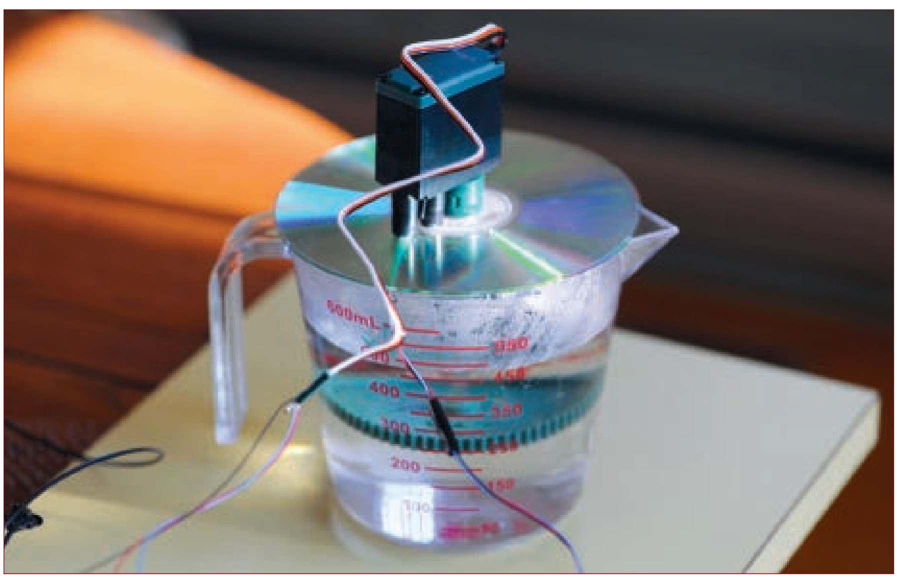


FIGURE 5. The measuring cup, temperature sensor, and optional VEX motor used to stir the solution are shown here. Most of these components are available from the VEX kit or around the house.

```
#include "Main.h"

void main ( void )
{

    long Resistance = 0;
    // Assuming a 10K Thermistor
    double Temperature = 0.0;
    // Scaled temperature reading in Degrees
    long Cycle = 0;
    // Number of cycles
    int Period = 0;
    // Measure temperature every n seconds
    int SetPoint = 75;
    // Set water temperature to 75 degrees
    double Sum_Temperature = 0.0;
    // Compute average temperature

    Cycle=0 ;
    // Initialize the temperature reading cycle
    // count (every n seconds)

    Sum_Temperature = 0.0 ;
    // Initialize the temperature sum

    while ( 1 )
    // Main control loop for temperature
    // experiment #1

    {


    // Read the temperature from the LM34 solid
    // state temperature sensor
    Temperature_Sensor_1 = GetAnalogInput
    ( LM34 ) ;

    // Display the raw temperature sensor 1
    // reading
    // PrintToScreen ( "Temperature 1 = %d\n" ,
    // (int)Temperature_Sensor_1 ) ;

    // Convert raw temperature to degrees
    // Fahrenheit
    Temperature = ConvertToFahrenheit
    (Temperature_Sensor_1) ;

    // Accumulate the temperature sum for
    // 20 readings
    Sum_Temperature += Temperature ;

    // Compute average temperature and display
    // every 20 Cycles
        if ( Cycle %20 == 0 )

        {

            PrintToScreen ( "Cycle = %d" ,
            (int)Cycle/20 ) ;

            // Compute the average temperature for
            // 20 cycles
            Temperature = Sum_Temperature / 20.0 ;

            // Display the converted temperature
            // sensor 1 reading
            PrintToScreen ( " Temperature F = %ld" ,
            (long)Temperature ) ;

            // Convert temperature in degrees
            // Fahrenheit to degrees Celsius
            Temperature = ConvertToCelsius
            (Temperature) ;

            // Display the converted temperature
            // sensor 1 reading
            PrintToScreen ( " Temperature C = %ld\n"
            , (long)Temperature ) ;

            // Wait a bit between temperature
            // readings
            Wait ( 1000 ) ;

            Sum_Temperature = 0.0 ; /
            // Initialize the temperature sum

        }

        // Increment the temperature reading cycle
        // count (every n seconds)
        Cycle++ ;
        Wait ( 50 ) ; // Wait a bit between
temperature readings

        }
}
```

LISTING 1. This code demonstrates how you can read the LM34 using the VEX microcontroller's ADC port. Remember it reads analog values and converts them to 10-bit digital values ranging from 0 to 1,023. It also shows you how to convert the digital values to specific temperatures in floating point in degrees Fahrenheit and Celsius, and shows you how to signal condition (filter) the LM34 using data collected from it at specific time intervals and then averaging them.

The Mixer Assembly

The optional mixer assembly shown in **Figure 5** provides a means of stirring solutions at periodic intervals under control of the microcontroller. It is mounted on top of the beaker or measuring cup to prevent the liquid from splashing around. Most of the components used for this assembly are available from the VEX kit or from around your house. You may want to repeat the first experiment using the optional mixer to periodically stir the solution and see how it affects the liquid cooling rates.

The mixing motor is mounted centrally on a discarded CD or DVD by drilling two holes for the motor mount (refer to **Figure 5** again). It is used as a splash guard to waterproof the motor. A chemical-resistant plastic paddle is attached to the motor axle and is used to stir chemical solutions at periodic time intervals as needed.

The mixing rod connected to the motor is coated with epoxy glue or plastic tubing to prevent it from exposure to corrosive chemicals; in this case, salt. A VEX three-wire motor is connected to the Motor 1 input on the motor block so that it can be switched on or off by the microcontroller at periodic intervals, as specified in the Easy C application. The motor activation periods are measured using the microcontroller's internal timers, making it the perfect control timer. The optional pushbutton switch shown in the **schematic** can be used to start/stop the experiment.

Data Collection

The temperature is read as a raw digital value between 0 and 1,023, representing the current analog temperature. So how do we scale these values to readings that we commonly use, such as degrees Fahrenheit or Celsius? The

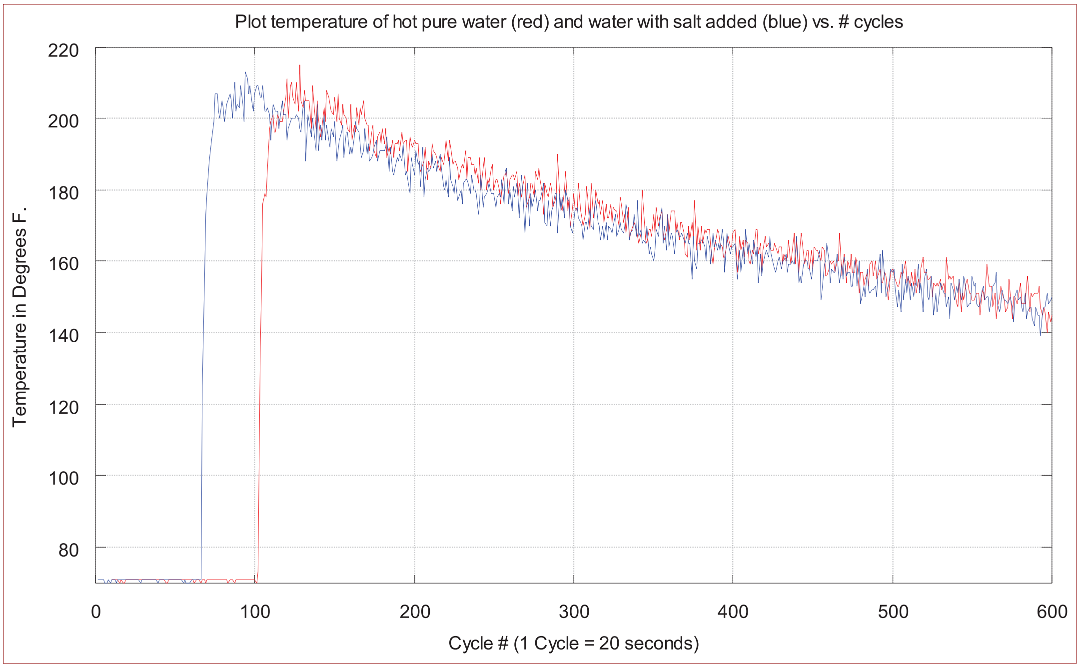


FIGURE 6. A plot of the cooling curve of warm water vs. the cooling curve of warm water with salt added to it. You can see for yourself what the effect of adding salt to water is and how it changes the heating/cooling characteristics.

answer to this question is the Easy C application used to collect the necessary temperature data shown in **Listing 1**. Data is captured to a text file using the Easy C terminal. **Listing 1** provides answers to these questions. It also shows you how to convert the digital values to specific temperatures in floating point for degrees Fahrenheit and Celsius. **Listing 1** shows you how to perform signal conditioning for the LM34 by using data collected from it at specific time intervals and averaging them. The Easy C source code is available with the article downloads.

The Easy C application Temperature_Experiment_1.c will collect the data for us and average every 20 readings per cycle in order to reduce noise in the temperature readings (this process is signal conditioning). Other techniques can be applied including using multiple temperature sensors, since up to 16 can be connected to the VEX microcontroller's analog/digital inputs. In addition, using shielded wire such as microphone or coaxial video cable, or twisted pair wire can help to cancel out the electrical noise. Sometimes electronic RC filters are used on the analog inputs. Each temperature reading cycle takes 20 seconds to complete.

The period and number of samples taken before averaging can be easily modified in the Easy C application.

Data Analysis

After completing these experiments, we can plot the data using data analysis tools such as Microsoft Excel, Open Office Calc, Matlab, or even Open Octave (which is very similar to Matlab). Plot the cooling curve of warm water vs. the cooling curve of warm water with salt added to it as shown in **Figure 6**. It shows the plot of the cooling curve of warm water vs. the cooling curve of warm water with salt added to it so you can see the effect adding salt has and how it changes the heating/cooling characteristics. It also helps to explain why salt is added to water for cooking and is spread on roads during icy weather conditions.

Listing 1 demonstrates how you can read the LM34 using the microcontroller's ADC port. Remember it reads analog values and converts them to 10-bit digital values ranging from 0 to 1,023. It also shows you how to convert the digital values to specific temperatures in floating point in degrees Fahrenheit and Celsius, and shows you how to signal condition (filter) the LM34 using data collected from it at specific time intervals and averaging them.

Applications

A VEX microcontroller on its own makes a great timer/intervalometer. Using the temperature sensor, we can control fans, heaters, and other types of equipment requiring thermal sensing. We can use it to heat or cool liquids for specific time periods or intervals depending on the experiment requirements. In fact, in photography, developing black and white or color film or slides requires a very accurate time and temperature process. These processes could be automated similar to the color labs at department stores or photography shops. The LM34 temperature sensor can also be used for weather or medical applications. For example, you can record the outdoor temperature over a period of days and plot it using the tools described in this article.

Conclusion

We discussed how analog temperature sensors work and what they can be used for. We showed how to use the LM34 analog temperature sensors with the VEX microcontroller by carrying out a simple experiment to measure liquid cooling rates with pure water, and then added sodium chloride to the water to see how it changed the heating/cooling properties. We also demonstrated how we can use VEX in the lab to help automate experiments. Next time, we will continue with analog temperature sensors and another temperature experiment. **SV**

Continued from page 23

BASIC Atom Pro 28. By offloading the servo pulse generation and sequence movement timing to the SSC-32, the BASIC Atom has plenty of power.

The included Phoenix program allows the robot to walk with variable speed in any direction (translation), or turn in place (rotation), or any combination of the two. The leg lift and ride height is adjustable, as well as the gait walking speed. The body can rotate in every axis.

There are preset walking modes and gaits to choose from. All of these are accessible from the controller. Lynxmotion recommends the wireless PS2 controller (RC-01) to get the robot up and running quickly. The Phoenix code also supports a serial control mode for controlling the robot via a serial connection. The Phoenix code was written by Jeroen Janssen.

Programming is already done for the robot. The control options are: PS2 remote control; Xbee/DIY R/C stick radio control; and TTL serial control. The robot is compatible with the NiCad and Ni-MH Universal Smart Charger batteries and a 6.0 volt Ni-MH 2,800 mAh battery pack, both available from Lynxmotion.

For further information, please contact:

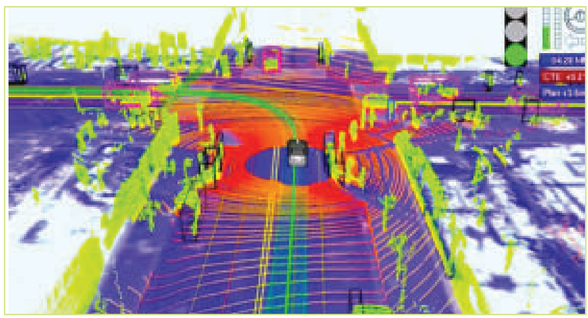
Lynxmotion Website: **www.lynxmotion.com**

Is your product innovative, less expensive, more functional, or just plain cool? If you have a new product that you would like us to run in our *New Products* section, please email a short description (300-500 words) and a photo of your product to:

newproducts@servomagazine.com

Continued from page 29

DRIVE ME CRAZY



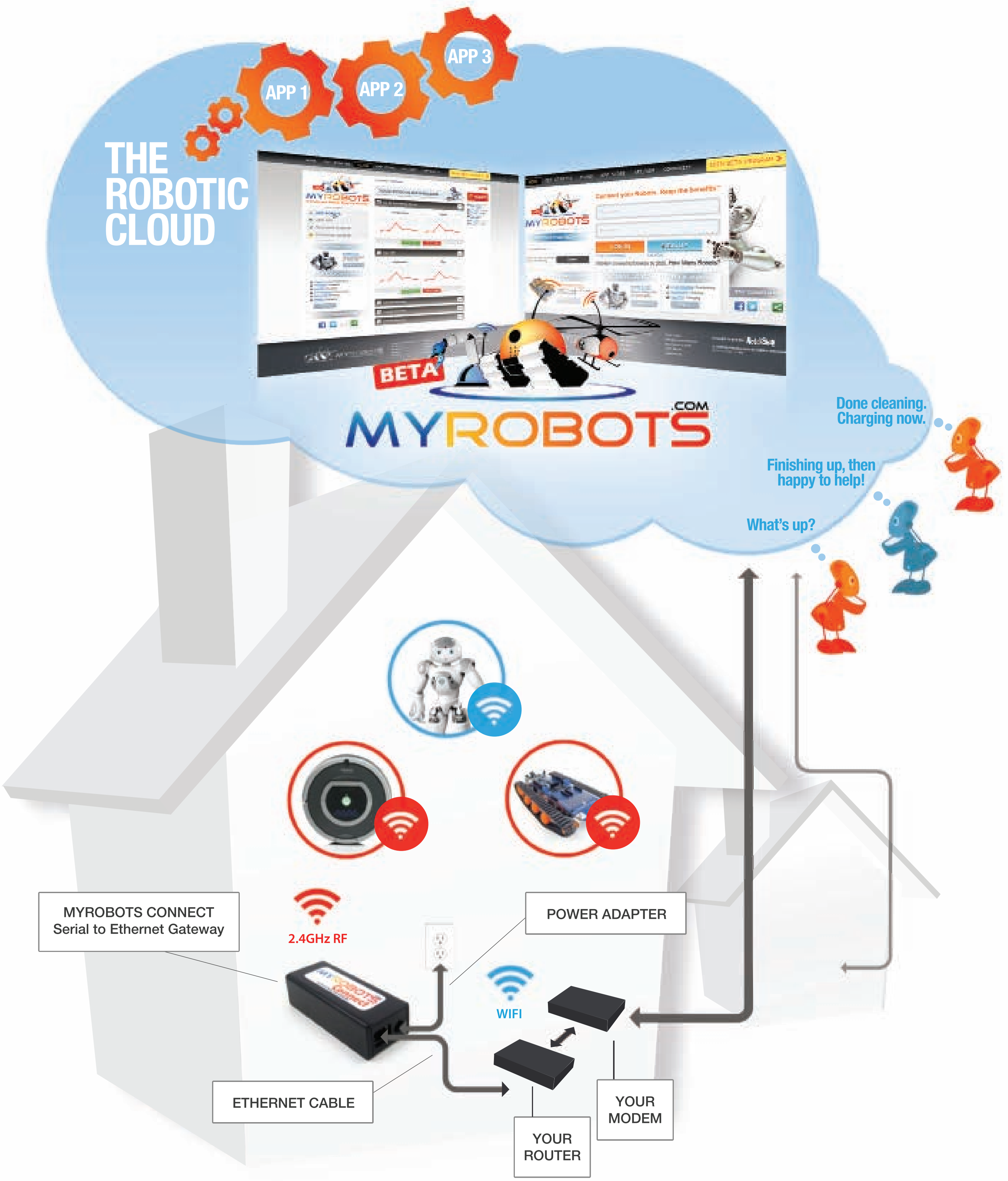
Robot cars are getting pretty good at parking themselves without crashing, or abducting their passengers. Robot cars also know how to drive like maniacs and even how to powerslide. These are all very neat tricks, but what's going to happen when all cars are this "talented?"

It's not just the sensor-driven skills that will soon be common to individual cars that will shape the future of automotive transportation, but also the ability for cars to communicate with each other, sharing constant updates about exactly where they are and where they're going. With enough detailed information being shared at a fast enough pace between all vehicles on the road, things like traffic lights will become completely redundant.

So, how close are we to something like this? It's hard to say. We have cars now that can drive themselves just about as reliably as a human can, and many automakers are working at inter-car communication. However, there are a lot of legal and social issues standing in the way of widespread adoption, and it's going to take a concerted effort to provide a framework in which we can safely allow progress to be achieved.

"The technology is pretty much already there," says Peter Stone, a computer scientist at the University of Texas at Austin. Stone is thinking of the advantages for the disabled and elderly who can't currently drive; for parents who don't have time to take their kids to soccer (they can take themselves!); and above all, for traffic safety and the more efficient movement of people everywhere. It's one thing, though, to realize that Google engineers have been zipping through our midst in autonomous concept cars. It's another to picture what will happen when we're all in these things – when the eye contact and social rules that currently govern urban driving are replaced by computer systems chatting with each other. "When they do interact," Stone says, "it will be at intersections as much as anywhere else on the road."

Stone and one of his doctoral students, Kurt Dresner, realized intersections will change not just because they'll need to accommodate driverless cars, but because driverless cars will make intersections much more efficient. Right now, you might wind up sitting at a red light for 45 seconds even though no one is passing through the green light in the opposite direction. You won't have to do that in a world where traffic flows according to computer communication instead of the systems that have been built with human behavior in mind, however.



Connect your robots. Reap the benefits ™

MyRobots.com strives to make cloud robotics a reality accessible to everyone and everything by enabling all robots and smart devices to connect to the Internet.

You can think of MyRobots.com as a social network for robots and smart devices. In the same way humans benefit from socializing, collaborating and sharing, robots can benefit from interacting and sharing their sensors' information, which provides insight on their current state, and allows them to be controlled and monitored remotely.

Through the **MyRobots App Store**, robots augment their capabilities so they can transcend their limitations, allowing them to do more than what they were originally designed for. This means robot owners can reap the full benefits of their robots.

Brought to you by RobotShop.com Robotics at Your Service!™

Powered by ThingSpeak

The *SERVO* Webstore

Attention Subscribers ask about your discount on prices marked with an *

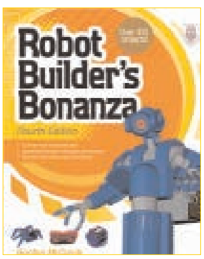
CD-ROM SPECIALS

NEW RELEASE

ROBOTICS

Robot Builder's Bonanza, Fourth Edition

by Gordon McComb



Robot Builder's Bonanza, Fourth Edition includes step-by-step plans for a number of motorized platforms. The book is described as a compendium of robotics topics, containing more than 100 projects, including 10 robot designs new to the fourth edition. These modular robots are low cost, and are made to be reproduced by readers with no training in mechanical construction.

$29.95*

Mechanisms and Mechanical Devices Sourcebook 5th Edition

NEW!

by Neil Sclater

Fully revised throughout, this abundantly illustrated reference describes proven mechanisms and mechanical devices. Each illustration represents a design concept that can easily be recycled for use in new or modified mechanical, electromechanical, or mechatronic products. Tutorials on the basics of mechanisms and motion control systems introduce you to those subjects or act as a refresher.

Reg $89.95 Sale Price $79.95

Making Things Move: DIY Mechanisms for Inventors, Hobbyists, and Artists

by Dustyn Roberts

In *Making Things Move: DIY Mechanisms for Inventors, Hobbyists, and Artists*, you'll learn how to successfully build moving mechanisms through non-technical explanations, examples, and do-it-yourself projects — from kinetic art installations to creative toys to energy-harvesting devices. Photographs, illustrations, screenshots, and images of 3D models are included for each project.

$29.95*

Build Your Own Humanoid Robots

by Karl Williams

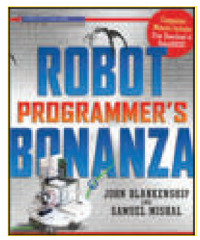
GREAT 'DROIDS, INDEED!

This unique guide to sophisticated robotics projects brings humanoid robot construction home to the hobbyist. Written by a well-known figure in the robotics community, *Build Your Own Humanoid Robots* provides step-by-step directions for six exciting projects, each costing less than $300. Together, they form the essential ingredients for making your own humanoid robot.

$24.95*

Robot Programmer's Bonanza

by John Blankenship, Samuel Mishal



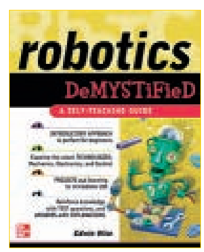
The first hands-on programming guide for today's robot hobbyist! Get ready to reach into your programming toolbox and control a robot like never before! *Robot Programmer's Bonanza* is the one-stop guide for everyone from robot novices to advanced hobbyists who are ready to go beyond just building robots and start programming them to perform useful tasks.

$29.95

Robotics Demystified

by Edwin Wise

YOU DON'T NEED ARTIFICIAL INTELLIGENCE TO LEARN ROBOTICS!

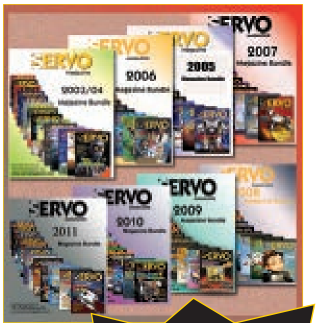


Now anyone with an interest in robotics can gain a deeper understanding — without formal training, unlimited time, or a genius IQ. In *Robotics Demystified*, expert robot builder and author Edwin Wise provides an effective and totally painless way to learn about the technologies used to build robots! **$19.95**

We accept VISA, MC, AMEX, and DISCOVER
Prices do not include shipping and may be subject to change.

To order call 1-800-783-4624

SERVO Magazine Bundles



Save $10.00 Only $57.95!

Now you can get one year's worth of all your favorite articles from *SERVO Magazine* in a convenient bundle of print copies. Available for years 04, 05, 06, 07, 08, 09, 10 and 2011.

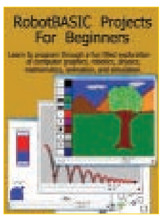
Any bot builders out there? Get cool robotics stuff from my store!

Call me at my order desk!

Visit my online store @ www.servomagazine.com

RobotBASIC Projects For Beginners

by John Blankenship, Samuel Mishal

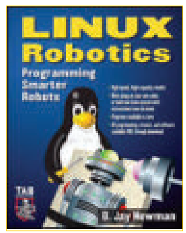


If you want to learn how to program, this is the book for you. Most texts on programming offer dry, boring examples that are difficult to follow. In this book, a wide variety of interesting and relevant subjects are explored using a problem-solving methodology that develops logical thinking skills while making learning fun. RobotBASIC is an easy-to-use computer language available for any Windows-based PC and is used throughout the text.

Price $14.95

Linux Robotics

by D. Jay Newman

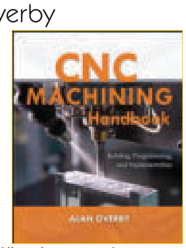


If you want your robot to have more brains than microcontrollers can deliver — if you want a truly intelligent, high-capability robot — everything you need is right here. *Linux Robotics* gives you step-by-step directions for "Zeppo," a super-smart, single-board-powered robot that can be built by any hobbyist. You also get complete instructions for incorporating Linux single boards into your own unique robotic designs. No programming experience is required. This book includes access to all the downloadable programs you need.

$34.95

CNC Machining Handbook: Building, Programming, and Implementation

by Alan Overby

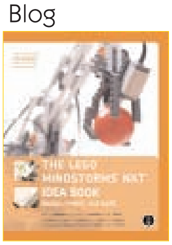


The *CNC Machining Handbook* describes the steps involved in building a CNC machine and successfully implementing it in a real world application. Helpful photos and illustrations are featured throughout. Whether you're a student, hobbyist, or business owner looking to move from a manual manufacturing process to the accuracy and repeatability of what CNC has to offer, you'll benefit from the in-depth information in this comprehensive resource. **$34.95**

SPECIAL OFFERS

LEGO MINDSTORMS NXT Idea Book

by the Contributors to The NXT Step Blog



If you're serious about having fun with LEGO® robotics, you've come to the right place. The team behind The NXT STEP blog — the authoritative online source for MINDSTORMS® NXT information and advice — has packaged its considerable skills and experience in this book. Inside, you'll find some of the team's best ideas for creating cool and sophisticated models, including instructions for eight robots you can build yourself.

Reg $29.95 Sale Price $24.95

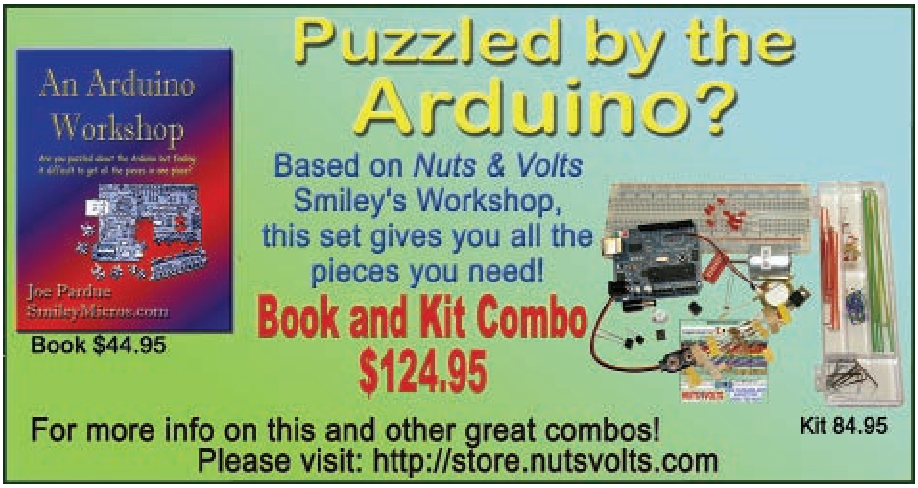
Technology Education Package for Everyone Starting in Electronics

This lab – from the good people at GSS Tech Ed – will show you 40 of the most simple and interesting experiments and lessons you have ever seen on a solderless circuit board. As you do each experiment, you learn how basic components work in a circuit. Along with the purchase of the lab, you will receive a special password to access the fantastic online interactive software to help you fully understand all the electronic principles. For a complete product description and sample software, please visit our webstore.

Regular Price $99.95 **Subscriber's Price $95.95**

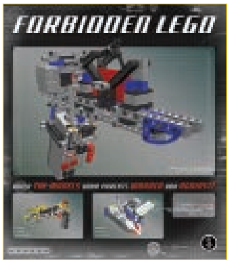
Or order online www.servomagazine.com

SPECIAL OFFERS



Forbidden LEGO

by Ulrik Pilegaard / Mike Dooley

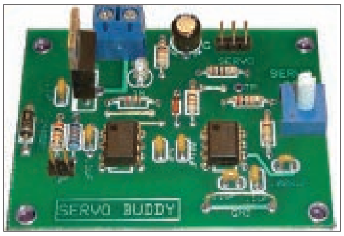


Forbidden LEGO introduces you to the type of free-style building that LEGO's master builders do for fun in the back room. Using LEGO bricks in combination with common household materials (from rubber bands and glue to plastic spoons and ping-pong balls) along with some very unorthodox building techniques, you'll learn to create working models that LEGO would never endorse.

Reg $24.95 Sale Price $19.95

PROJECTS

The SERVO Buddy Kit



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

For more information, please check out the May 2008 issue or go to the SERVO webstore.

Includes an article reprint.

Subscriber's Price **$39.55**

Non-Subscriber's Price **$43.95**

3D LED Cube Kit

From the article "Build the 3D LED Matrix Cube" as seen in the August 2011 issue of Nuts & Volts Magazine.

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

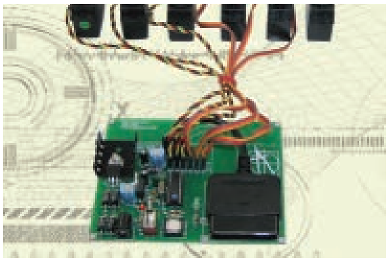
Colors available: Green, Red, Yellow & Blue.

Jig and plastic cases also available.

Subscriber's Price **$57.95**

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

PS2 Servomotor Controller Kit



This kit accompanied with your own PlayStation controller will allow you to control up to six servomotors. Includes all components and instruction manual.

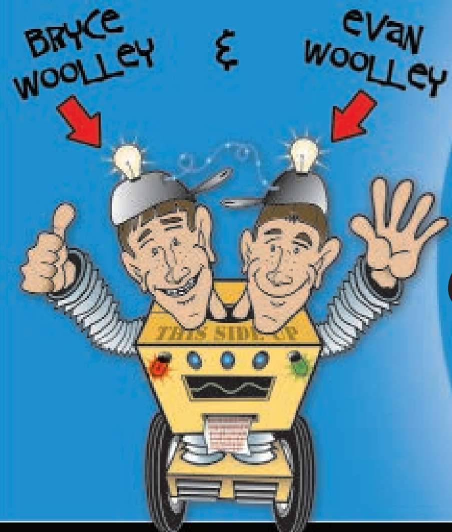
For more information, please see the February 2011 edition of *SERVO Magazine*.

Assembled units available!

Subscriber's Price **$79.95**

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

Twin Tweaks



THIS MONTH: *The Sensor Olympics 2 – Going the Distance*

Discuss this article in the SERVO Magazine forums at http://forum.servomagazine.com

THE SCRIBBLER IS LOOKING TO TURN ITS OPENING VICTORY INTO A WINNING STREAK.

www.servomagazine.com/index.php?/magazine/article/may2012_TwinTweaks

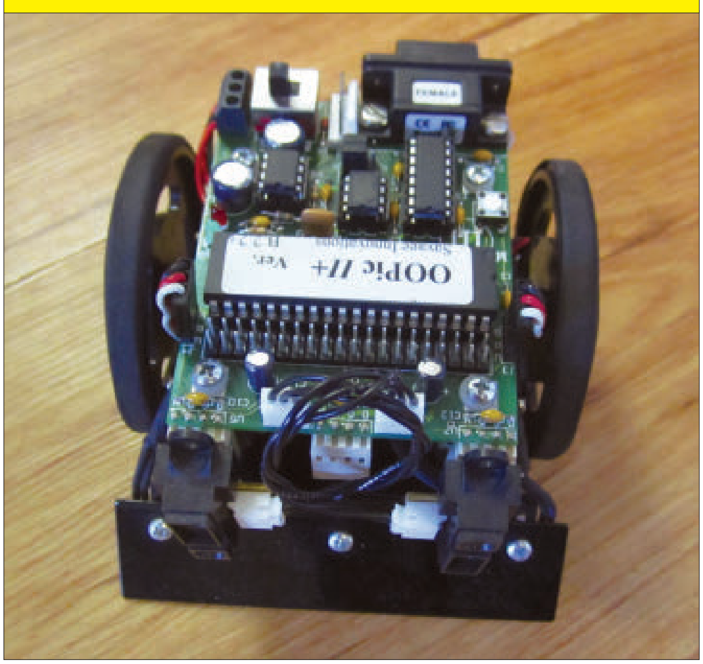
Last time, we pitted some members of our robot menagerie against each other in a thrilling contest for infrared sensor supremacy. The slow and steady Scribbler came out on top with its enigmatic IR sensors, and the perennial competitor — the Mark III — is hungry for redemption this month. For the second event of the Sensor Olympics, we'll be testing sensors used for obstacle avoidance – namely infrared rangefinders. Will the Scribbler extend its winning streak, or will the Mark III pull a surprise upset?

Open Rangefinder

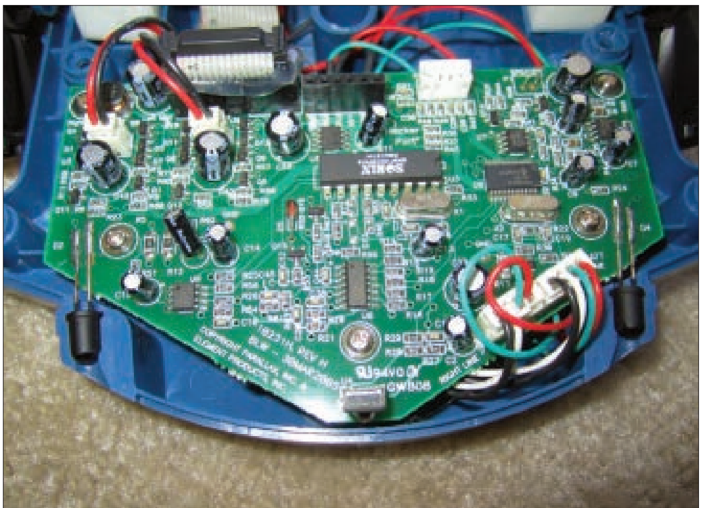
Obstacle avoidance sensors are a popular feature for robot kits. A classic ingredient for a maze navigating bot, obstacle avoidance sensors generally work by sending out some sort of pulse, waiting for it to reflect off of an obstacle, and sensing the reflected pulse. The strength and delay of the reflected pulse give valuable clues as to the distance of an obstacle. The Scribbler and the Mark III both come with a classic solution for obstacle detection: infrared sensors. Both robots use a combination of infrared transmitters to send out a beam of infrared light and an infrared detector to sense the reflected beam. The Scribbler uses transmitters and detectors that are mounted in the very front of the bot, while the Mark III uses the extremely popular 2YOA21 Sharp rangefinders. Before we devised some tests for the sensors, we wanted to get a bit more acquainted with them.

The Scribbler obstacle avoidance sensors are a bit more accessible than the line following sensors, and can be reached simply by removing the Scribbler's lid. Much like the line following sensors, the components themselves lacked any identifying information that might lead to a datasheet or other specifications. The official Scribbler information from Parallax is also less than illuminating – the Scribbler page describes the sensors as IR phototransistors, and the schematic does not indicate whether the sensors are implemented as digital or analog inputs. We wanted to know whether the sensors were used as digital or analog inputs because that would affect the way we

THE MARK III IS LOOKING FOR REDEMPTION.



The Scribbler's IR sensors are ready to stare down some obstacles.



A look inside reveals little about the enigmatic sensors.

Investigating the nature of the Scribbler sensors.

compared the Scribbler to the Mark III. Surely other tinkerers have had similar burning questions about the artistic bot, so the answers could likely be found on a robotics forum somewhere. However, we preferred to live by the sage mantra of Reading Rainbow's LeVar Burton, so we wanted to see for ourselves instead of taking their word for it.

We pulled out our trusty multimeter to do a simple test that would determine if the sensors were analog or digital. We pressed the contacts against the ground and signal pins of the infrared receiver and moved an obstacle in front of the bot. If the multimeter gave a reading that jumped from zero volts to five volts once an obstacle got close enough, then it was treated like a digital input. If the multimeter gave a reading that gradually increased from zero to five volts as an obstacle approached, then the sensor was being used as an analog input. The results of our test were somewhat difficult to interpret given the difficulty in placing the contacts only on the individual legs of the infrared receiver, but we did only see dramatic jumps from zero to about five volts and no gradual change.

To seal the deal, we inspected the Scribbler program itself. The specialized Scribbler GUI — even with all of its colorful building blocks and images of brick walls that flagged the blocks dealing with obstacle avoidance — was about as forthcoming with technical details as Raj Koothrappali is with conversation around attractive women. Thankfully, the Scribbler program offers the option to see the code in Basic form, and that is where we found our definitive answer. The IR sensors were treated as either true or false — the hallmark of a digital input. Now that we had solved the mystery, the next question was what were we going to do with that prized knowledge.

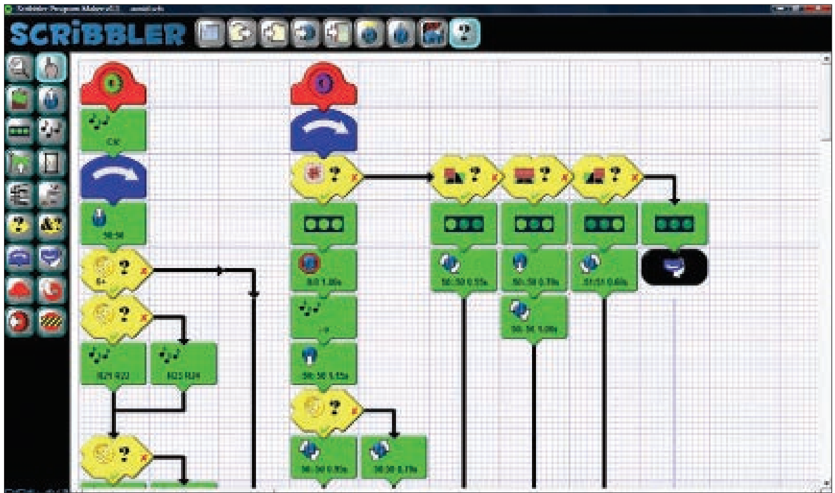
A Sharper Image

One of the challenges in assessing the rangefinder sensors was to devise a test that would allow us to accurately track the results. Very often, rangefinding sensors are used for obstacle avoidance, usually in the context of a maze challenge. So when the sensor works, the robot doesn't hit the obstacle. Sometimes when the sensor doesn't work, the robot might miss the obstacle anyway out of pure luck or happenstance. So, what we needed was an obstacle directly in the path of the bot and a behavior that would unambiguously indicate sensing of the obstacle. For the Scribbler, this was an easy task. The LEDs could light up when an obstacle was sensed, and we could have the robot back up. A similar behavior could be programmed into the Mark III, minus the helpful LEDs.

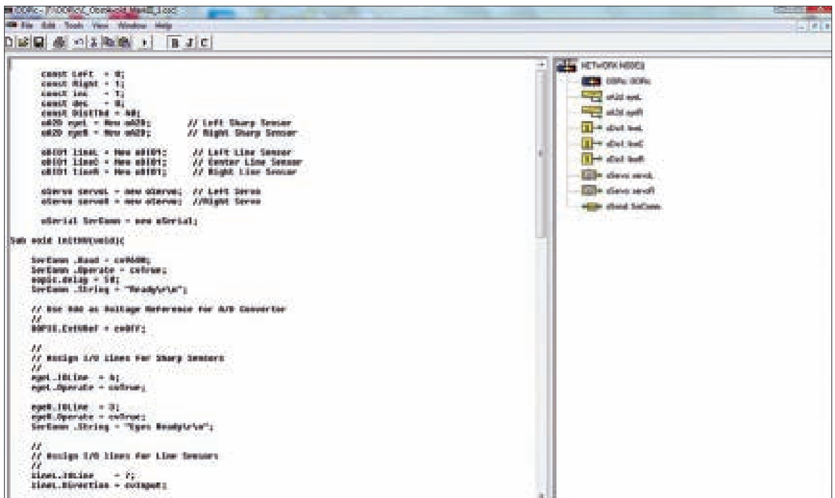
The other challenge with devising a test was to determine a way to fairly compare the digital inputs on the Scribbler with the analog rangefinders on the Mark III. The Mark III uses two 2YOA21 Sharp rangefinders. The 2YOA21 sensors are the de facto successors to the ubiquitous GP2D12 sensors, with comparatively improved range and reduced output fluctuations. The 2YOA21 sensors can sense obstacles in the range between 5 cm and 150 cm. The output voltage from the sensor can be converted into a range in centimeters using a simple calculation. The comprehensive datasheet for the sensor comes with all of this information and more, just as much a font of useful physics info and unsolicited trivia as Dr. Sheldon Cooper.

To compare this sophisticated sensor to the Scribbler's more rudimentary array seemed like comparing apples to oranges. It seemed like a foregone conclusion that the 2YOA21 would have a better range, which was the intuitive metric for comparison. However, that very conflict inspired us as to what a useful comparison would be. The 2YOA21 was the type of top shelf sensor one might order specifically for a project, knowing that it will be up to snuff and super effective but the mysterious phototransistors in the Scribbler were more akin to the type of sensors that you might find buried in your toolbox – incidental discoveries when you are looking for something just good enough to get the job done.

It seemed like a useful question to have answered – can random toolbox sensors compete with cutting-edge improvements on popular designs? We wouldn't necessarily look at range and other metrics often covered in datasheets, but rather we wanted to see how these sensors dealt with non-ideal conditions. Sure the Scribbler and Mark III can avoid bright walls well enough, but what about more ephemeral obstacles?



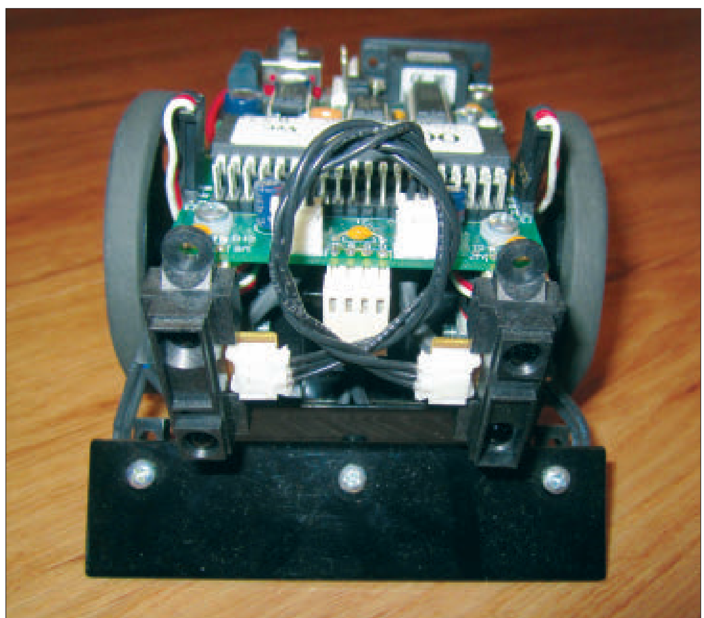
PROGRAMMING THE SCRIBBLER WITH AN OBSTACLE AVOIDANCE PROGRAM.



PROGRAMMING THE MARK III WITH AN OBSTACLE AVOIDANCE PROGRAM.

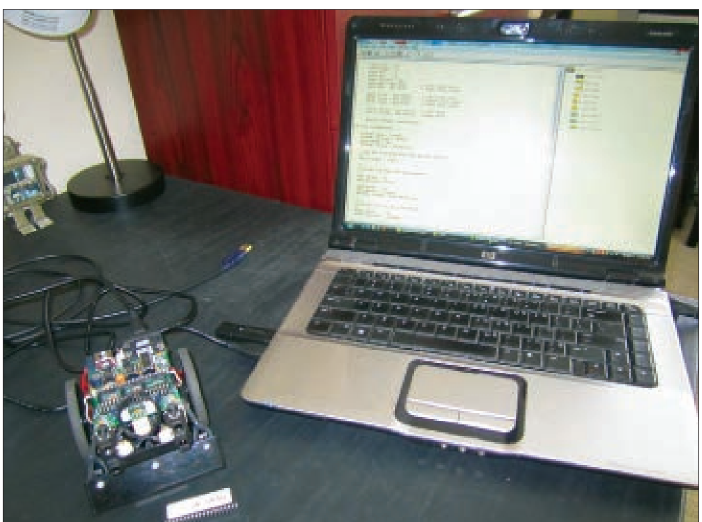
No Block on the Horizon

Before we moved on to more whimsical obstacles, we wanted to get a baseline for performance using a standard obstacle – something bright, something solid, something to bring relief on those sniffly days. A tissue box suited our purposes perfectly. To accurately track when our bots saw the obstacle, we marked distances in centimeters on the back of our line following track, and then we were ready to let the arch rivals duke it out once again. We started with the Scribbler. The LEDs and sudden backtracking made it obvious when the bot saw the box, and we marked off the distance over a number of trials. The Scribbler sensed the obstacle at distances ranging from 35 cm to 45 cm away, but showed a lot of variation within that range. The Scribbler did, however, always sense the obstacle and did not make the embarrassing gaffe of running headlong into the box.



THE MARK III'S RANGEFINDERS ARE LOOKING SHARP.

TESTING WITH OUR STANDARD OBSTACLE.


FIXING THE COSMIC WEDGIE.

Up next was the Mark III. Even without LEDs, the robot's sudden backtracking made it easy to note when it saw the obstacle. The Mark III also consistently spotted the obstacle, and it did so at a far more consistent distance than the Scribbler. Every trial resulted in the bot starting its moonwalk at about 34 to 35 cm. A little variation is to be expected. The motors aren't perfectly calibrated, so the slightly different angle of approach could mean slightly different distances when the object is detected. This was a very pleasing result, however, given that the obstacle avoidance program we downloaded to the Mark III dictated that the bot take evasive maneuvers once an obstacle was 36 cm away. The discrepancy between the program and our measurements could be explained by the fact that our measurements used the front wedge of the Mark III as a guide while the rangefinders were set slightly back (about a centimeter) into the bot.

After our first round, it looked like the Mark III was ahead based on its excellent consistency. Now, we wanted to devise a test that would really separate the winners from the also-rans.

Once Upon a Time in the West

Much like with our light box last time, we wanted to test the sensor's vulnerability to interference and non-ideal conditions. With the infrared sensors, we wanted to do something similar with our disappearing track but in a way that seemed more practically relevant to obstacle avoiders. Pondering what a problematic obstacle would look like conjured memories regarding a certain race through the desert in 2004.

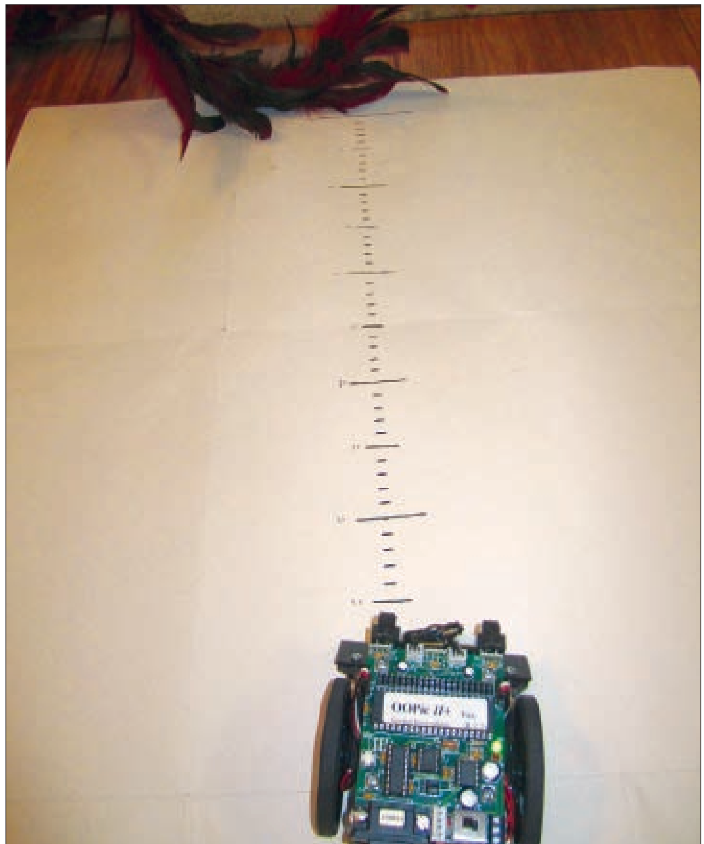
It was a brisk March morning in Barstow, CA when a lumbering mechanical beast struck out on its ill-fated journey across the desert, doomed never to reach its destination at the gateway to the hedonist's paradise in Primm, NV. The lumbering beast was TerraMax — the impressive entry from Big Truck Robotics. TerraMax was easily the biggest entry in the 2004 DARPA Grand Challenge, as it was based on a tactical military vehicle. The robot was outfitted with LIDAR and cameras for navigation, and when we first watched TerraMax at the qualifying races it exemplified the adage that slow and steady wins the race.

On race day, however, TerraMax's dreams of greatness were stymied shortly into the race. After lumbering along for a little while, the massive bot stopped in its tracks. Had it gone off course and come face to face with an impassable ravine or massive boulder? Had some catastrophic mechanical failure crippled the behemoth to stop its inevitable march to the finish line? No, the reason for the robot's stoppage was much smaller than that – a bush was in the way. A small, unsuspecting, prickly desert bush turned out to be the David that slayed the Goliath.

This was not because TerraMax was a die-hard environmentalist, but rather because the bot's sensors overestimated the solidity of the obstacle. Of course, the vehicle could trample a prickly bush without a second thought, but a boulder in the middle of the road was another matter entirely. If it was larger than the bot's ground clearance, it could cause a major problem.

Fortunately, TerraMax returned in 2005 and completed the race but the experience in 2004 is illustrative of a real problem that obstacle avoidance sensors can have: They can overestimate the solidity and danger of ephemeral obstacles — things that could easily be gone through instead of around. An ephemeral obstacle would be a good way to test whether the IR sensors on the Scribbler and Mark III were sensitive, and perhaps overly so.

We didn't have any tumbleweeds handy, so instead we repurposed some of our apartment décor. A peacock decoration with a feathery tail provided a perfect ephemeral obstacle – one that the Scribbler or Mark III could plow right through, but might prefer not to. We set up our testing ground and readied the Scribbler. After just a few tests, we were surprised at just how apparent the effects of changing the obstacle were. Before, the Scribbler sensed the tissue box at about 40 cm away. Now, however, the Scribbler was only taking evasive maneuvers at 10 cm if at all. Ambivalent as to whether this was a positive or a negative for the Scribbler's sensors, we set about to test the Mark III.

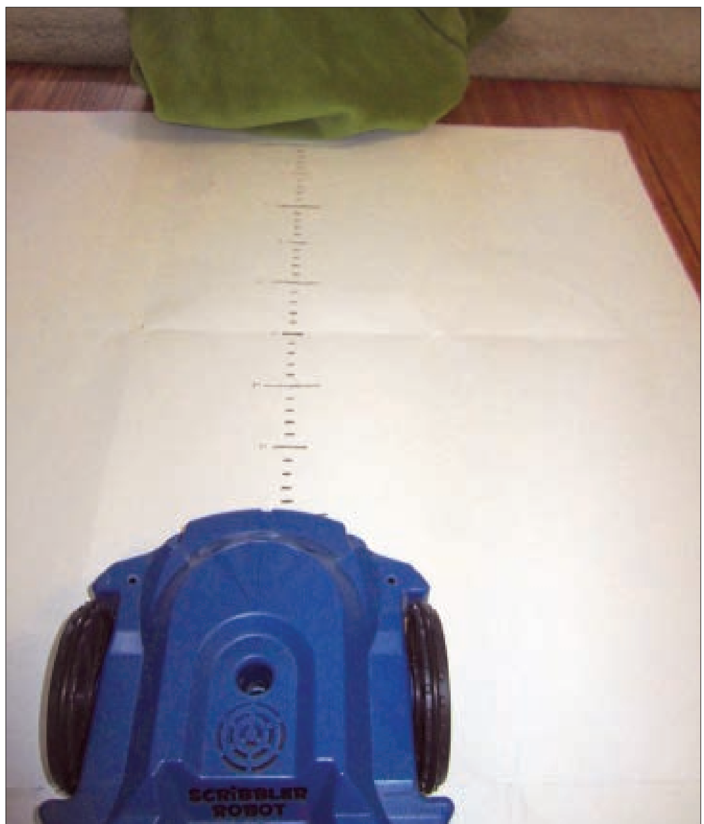


TESTING WITH AN EPHEMERAL OBSTACLE.

A Force Majeure Interrupts the Sensor Olympics

We expected the Mark III to start its tests much like it did with the ephemeral obstacle – a steady approach followed by sudden backtracking when the obstacle was detected. This, however, is not what we saw. When we turned on the Mark III, it started moving backwards at a brisk pace. Afraid that perhaps a hand had been in front of a rangefinder when we activated the bot, we tried again. Again, the Mark III made a hurried retreat, and no amount of hand waving in front of the Sharp sensor would change its mind. We tried again, but this time the Mark III was spinning around in a counterclockwise motion. We weren't playing dizzy bat, so we tried again, only to have the Mark III drive around in larger circles. Every new activation of the Mark III led to a random selection of these behaviors which struck us as extremely odd given that none of them really resembled the behavior we had programmed.

Our first instinct was to check the website for Savage Innovations, the creators of the OOPic. Unfortunately, the Savage Innovations website no longer exists, so support for the Mark III has become somewhat of an oral history of which this article may become a part. A quick bit of searching on the Internet led us to the proper diagnosis. Our Mark III exhibited the signs and symptoms of what has been termed by OOPic extraordinaire Richard Stofer as a "cosmic wedgie." A



TESTING WITH AN ANTI-ECHOIC OBSTACLE.


Anti-echoic?


And we have our winner!

cosmic wedgie occurs when the EEPROM doesn't have enough time to reset at the beginning of a program. When this happens, the robot wanders off to never never land and engages in odd behaviors like we were seeing. Fortunately, there is an easy fix: All you need to do is reset the EEPROM.

This was an easier fix for us than it is for a lot of folks. All of the chips on our Mark III were socketed because it was always built as a platform for expansion. Some folks, however, insist on soldering the OOPic brain directly into the board. We personally think that sockets as a general policy are a good thing, but if the brain is directly soldered to the board it can be removed if you are a fan of desoldering. The whimsical novelty of solder suckers and solder braid wear off quickly though, and thankfully there is another solution for those that want to reset the EEPROM without having to remove the chip itself. The Mark III board includes two jumpers, and when the jumpers are shorted together before resetting the board and trying to download a new program, it will be treated the same as if you had removed the EEPROM. We tried both methods and they both worked like a charm.

Resetting the EEPROM, however, was only enough to bring the bot back from never never land temporarily. To keep the cosmic wedgie from becoming a chronic condition, we had to treat the cause. The cause, fortunately, is simple. The treatment for a cosmic wedgie is to give the OOPic enough time to reset at the beginning of the program. The consensus is that five seconds is sufficient, so one of the most important lines of code in an OOPic program is oopic.delay = 500 at the beginning of the program. The offending line in our program was easy to find, and it read oopic.delay = 50. Just as with the IRS or SEC, being one zero off is indeed a big deal. We added a zero, reprogrammed the bot, and finally resumed the Sensor Olympics.

We like to include these experiences because we think dealing with unexpected problems is a part of the process experienced by all tinkerers. Working with a kit is never as smooth as following the step-by-step instructions. There will always be things left unmentioned by the manual; there will always be unexpected speed bumps on the road to your completed project.

Once we were able to test the Mark III with our ephemeral obstacle, we were surprised that once again it gave a result markedly different from the Scribbler. The Mark III consistently spotted the obstacle from a distance of 36 cm and evaded it accordingly. On one hand, this was an impressive result for the Mark III. The sensors were sensitive enough so that they even spotted our feathery simulacrum of the bush that foiled TerraMax. On the other hand, too much sensitivity to ephemeral obstacles could leave the bot feeling trapped when, in fact, it could plow obliviously through like the Scribbler.

We decided that the Mark III's result was the better one. There are numerous ways to compensate for oversensitivity. The program could average a few readings to sort of gauge the solidity of an obstacle. Presumably, an ephemeral obstacle will inconsistently reflect the IR beams leading to a low average, while a solid obstacle will consistently reflect the beams. Also, an array of sensors would be able to provide more information, and that may be one reason why the Mark III did indeed see the obstacle – it is outfitted with two of the Sharp rangefinders.

Closing Ceremonies

After our ephemeral obstacle, we had grand plans for a third challenge based on our anticipation of receiving a third type of sensor to test – the SRF05 from Devantech, an ultrasonic rangefinder that we would equip to the Mark III. Unfortunately, Devantech seems to be the Matt Damon to our Jimmy Kimmel Live, and we just ran out of time because the sensor didn't arrive. We have heard nothing but good things about Devantech from other tinkerers, but we always seem to have bad luck. In any event, stay on the lookout for a future project involving the SRF05. We still tested the Scribbler with our rendition of an anti-echoic obstacle meant to stymie an ultrasonic sensor, but both bots had no problem avoiding that obstacle as easily as the tissue box.

Our goal with the Sensor Olympics is fairly straightforward: to gain an intuitive understanding of the advantages and disadvantages of different types of sensors. Sure, many sensors have nicely detailed datasheets, but many of these stats about peak voltage and power consumption don't necessarily reveal whether *this* sensor will work to get you through *that* maze. Datasheets often don't reveal how the sensor will react to non-ideal conditions. This, however, is an understandable limitation because it would be foolish to expect concise datasheets to comprehensively list all of the ways in which conditions can stray from the ideal.

Of course, many times intrepid tinkerers are left without the guidance of datasheets. Sometimes, you'll be rummaging through your toolbox or cannibalizing old kits for parts, and the sensors you find may not have the identifying marks necessary to lead you to a datasheet. In that case, it is very useful to know how sensitive sensors are to different situations. Are most sensors dramatically affected by the reflectivity of the obstacles? Or, will most sensors be good enough to get my robot through the labyrinth safely?

In the end, we had no reservations about declaring the Mark III the winner of this event. The 2YOA21 Sharp rangefinders are impressive sensors that proved to us why they are, in fact, the standard in many kits. The Scribbler's simple IR phototransistors were a bit more flummoxed by changes in obstacle type, but it still gave a respectable effort that would be enough to get it through most mazes. SV

Then and NOW

Sensors For Mobile Robots

by Tom Carroll

That is exactly the title of Bart Everett's great sensor book from 1995, shown in Figure 1. Well, it's no coincidence because Bart's book has literally been a 'bible' for me in robot writing and consulting, as well as for thousands of other experimental robot builders. Bart is a friend of mine and works in one of the most enviable positions for a robotics enthusiast at the Navy's SPAWAR Systems Center in San Diego, CA. I have visited the site on several occasions and have seen some of the most amazing robot creations you could imagine.

In his book, Bart starts out discussing robots that he built as a kid, starting with Walter back in 1965. Bart's teleoperated anthropomorphic robot met its untimely demise when the household cleaning lady went into young Bart's bedroom (he was a sophomore in high school at the time) and turned on an old vacuum cleaner that emitted so much electromagnetic static that the five foot tall radio-controlled robot went berserk. When he came home and saw his decapitated robot on the floor, the dented vacuum cleaner on its side still running, and the front screen door off its hinges, it was obvious that the maid had 'killed it' with one mighty swing of the vacuum and left in such a hurry as to never be seen again.

FIGURE 1. ***Sensors for Mobile Robots*** **by Bart Everett.**

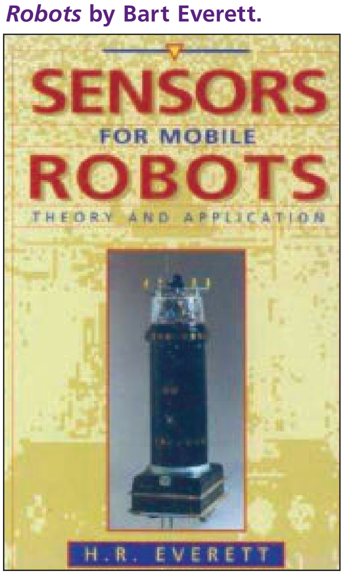


FIGURE 2. Bart Everett's Robart II.

Walter had no onboard sensors for perceiving its physical surroundings. Its successor was Crawler I — a small tracked robot equipped with tactile feelers made from looped guitar strings, and barely enough onboard intelligence to support a very primitive bump-and-recover mobility behavior. Non-contact sensing had to wait for Everett's first computer-controlled robot — ROBART I — which was his thesis project at the Naval Postgraduate School in the early 1980s. This robot utilized two types of sensors. One was an active ranging system based on the National Semiconductor LM-1812 monolithic sonar transceiver chip used in underwater fish finders and the other was a modified near-infrared proximity sensor from surplus circuit boards used on toys. ROBART I also had considerable collision detection capability in the form of an extensive array of tactile sensors, along with over-current sensing for its tandem drive motors.

The big sensing breakthrough on ROBART II came in 1982 with the introduction of Polaroid's ultrasonic ranging system which employed a novel electrostatic transducer specifically designed for operation in air; the original application, of course, being automatic camera focusing. The LM-1812 compatible transducers were designed for underwater use and had trouble achieving an effective impedance match with air. Thanks to the high-volume camera application, the Polaroid system was both small and inexpensive which allowed a total of 36 of these sensors to ultimately be incorporated on ROBART II (**Figure 2**). These were augmented with a number of improved near-infrared proximity sensors for enhanced collision avoidance.

According to Everett, "The basic idea was synergistic fusion of two different inexpensive sensor modalities to increase the chances of target detection and improve the accuracy. The slow speed of sound facilitates time-of-flight range measurements, but the associated beam divergence makes for poor angular resolution. Optical energy is easy to focus into a tight beam, but the speed of light is much harder to measure over short ranges."

Everett's long-time friend Anita Flynn — then an MIT co-op student working for the Navy — wrote her Master's thesis on this concept using ROBART II's sonar and optical sensors to collect empirical data for a variety of target sources.

Sensors Have Unlimited Applications

Sensors are part of almost every electro-mechanical device, whether industrial or commercial devices for the home. Robots — as mechanical devices that move — need all sorts of sensors, especially *mobile* robots. For those with interests in the more technical aspects of sensors, the NASA Tech Briefs SENSORS Tech Forum this October in Anaheim, CA is just one of many conferences dealing with sensors.

Simple toy robots may not have sensors, but they certainly could use sensors to allow them to detect a wall and make a turn, rather than continually trying to go through it. An edge-detecting sensor might help a simple robot from rolling off a table top and smashing itself on the floor. iRobot certainly was aware that they would have to place a lot of sensors in their Roomba vacuuming robot to allow it to perform even the simplest of room cleaning tasks. The Roomba has both of the sensor types mentioned above, and many more.

Active IR sensors detect wall proximity and slow the robot down or cause it to turn. An edge-detecting sensor allows it to see a dangerous top of the stairs and causes it to back and turn before it can perform a dive to its death. I wrote about robot sensors five years ago in *SERVO* but many new types have been developed since then. This time, I'd like to concentrate on sensors that detect objects and barriers, and as a result of this detection, are motion controlled. We'll concentrate on just active and passive object locator sensors.

Switches are the Earliest Examples of Robot Obstacle Detection

In the early days of robotics and for more simple robots these days, the use of bumper switches and whiskers to detect objects is a simple solution. My first robot (eons ago) used two four-pole double-throw switches — one at each end of the robot to change the polarity of the motors, thus changing the direction of the robot. All it could do was bang back and forth when it hit a wall in the front or back.

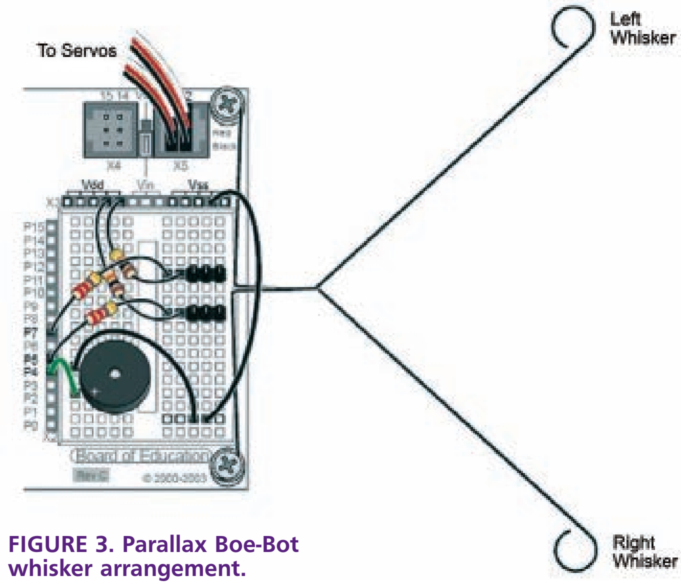


FIGURE 3. Parallax Boe-Bot whisker arrangement.

That seemed stupid to me, so I changed things a bit with four separate double-pole double-throw (DPDT) switches — one at each corner with a bumper. A curved bumper around the front part of the robot's base had a single DPDT switch behind it that would trigger when that part of the bumper was depressed. That really screwed up the works because the robot would sometimes spin out of control, so I tried a stepping relay, then a small bank of logic relays, and so on.

Later, I added a set of small coat-hanger wire whiskers attached to the front of the robot that could detect an object if one of the whiskers was moved. Rather than using sensitive micro-switches attached to each of the whiskers, I found it easier to have a conductive whisker surrounded by a circular ring contact surface so that any motion of the whisker up/down or right/left would cause the whisker to contact the circular ring and trigger a circuit. The Parallax Boe-Bot uses a similar set of whiskers mounted in front of two sets of conductive pins as shown in **Figure 3**. These early examples of sensors worked, but other builders and I found many better ways for our robots to detect and respond to the outside world.

Detecting Obstacles Away From the Robot

It is nice for your robot to be able to detect an object or obstacle before it slams into it or even gently touches it. There are many types of sensors that can accomplish this detection, with the most popular two being active ultrasonic and active infrared. The word 'active' implies that the sensor sends out a signal and then receives a returned echo. Detection before contact allows the robot to easily make a course change, or at least slow down (like the Roomba). Non-contact detection also prevents scaring or nicking of walls, molding, furniture, and objects. There are still a few marks on the lower moldings in my house where

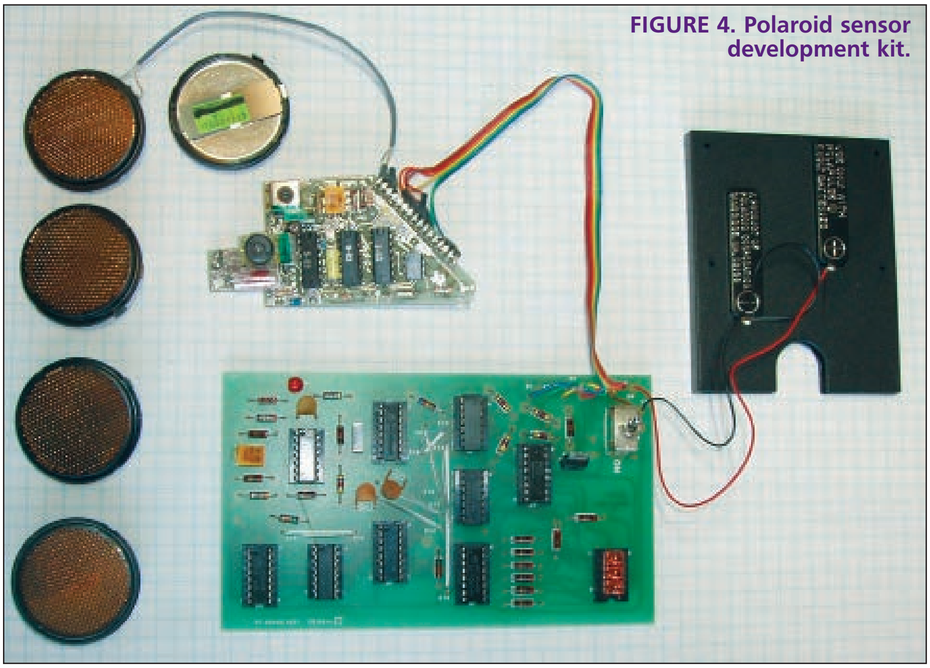


FIGURE 4. Polaroid sensor development kit.

my first Roomba slammed into them a bit too hard.

Ultrasonic Object Detection and Ranging

Let's examine some different distance measuring transducers. Consumer ultrasonic systems have been around for decades, such as intrusion alarms and early TV remote controls. These intrusion systems were adapted by some robot experimenters for object detection since many used simple time delay measurements for determining distance. Doppler frequency shift techniques can be used for speed sensing when an object is drawing closer; the returning signal's frequency shifts higher with shorter wavelengths, and vice versa on moving away. When Polaroid came out with their series of autofocus cameras (mentioned earlier) that used a single electrostatic transducer, robot builders and hackers were just waiting to tear into them and use them for robot detection methods. Polaroid has since gone through difficult financial times. The company has been shredded and one small company is trying to supply the instant film portion while another — SensComp — purchased the OEM products division of Polaroid and has expanded its line of electrostatic and piezoelectric technologies.

The electrostatic type of transducer requires a bit more circuitry to complete the overall system but has much longer distance measuring capability of six inches to 35 feet. I still have four of the original Polaroid development kits from the '80s, one of which is shown in **Figure 4**. The original angled camera circuit board has the transducer attached, and the large development board has a three-digit LED display for the distance readout. The black Polapulse battery holder was for the six volt battery developed especially by Polaroid; it's a flat, high current battery that was inside each pack of instant film. When I received these, I immediately inserted one of the supplied batteries and turned the system on. Holding the transducer in my hand, I was aiming it around my room when I accidentally touched the back of the transducer and was zapped by 300 volts. One only does that once.

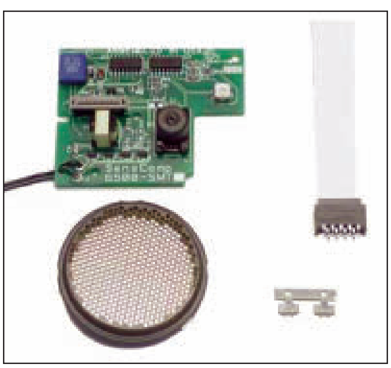


FIGURE 5. SensComp R14 sonar from Acroname.

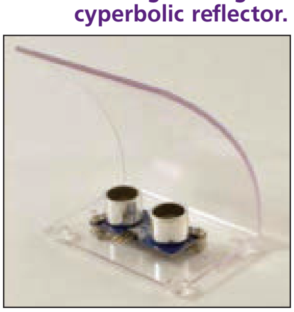


FIGURE 6. Parallax Ping))) with Gadget Gangster cyperbolic reflector.

SensComp now makes an excellent sonar unit (shown in **Figure 5**) based on the original Polaroid design. It's available at Acroname Robotics. It combines the 600 series transducer and a SensComp 6500 series ranging module. The system costs $56 and is Acroname part number R14-SONAR1.

If you need the extended range for your robot design, I highly recommend this sonar. It has a very narrow beam width of 15º with a drop of 6 dB which is great for distancing smaller objects. As you might know, these active sonar systems use the delay of an ultra-high frequency sound (ultrasonic) pulse that travels at the speed of sound at 1,126 feet per second, or 768 mph to and from the target object. The further away, the longer the delay.

The SensComp 6500 operates on 4.5V to 6.8V at 100 mA current draw. I found out that it works just fine on five and six volts with a crude bench setup I used. I could get reliable return reflected signals at 30 or more degrees from a painted wall. The system has 80 mS recycle cycles with 16- 49.4 kHz bursts. It's larger and more expensive than the smaller two-transducer units but has much greater range. The Acroname site (***acroname.com/robotics***) has a link to the code and hookup of the sensor system to a BrainStem microcontroller ("Data Logging with a BrainStem GP 1.0"); links to other controllers can be found on search engines.

Piezoelectric Distance-Ranging Transducers

The more popular piezoelectric transducers are sold by many companies and robot suppliers. These types of sensors emit an ultrasonic frequency sound burst in the

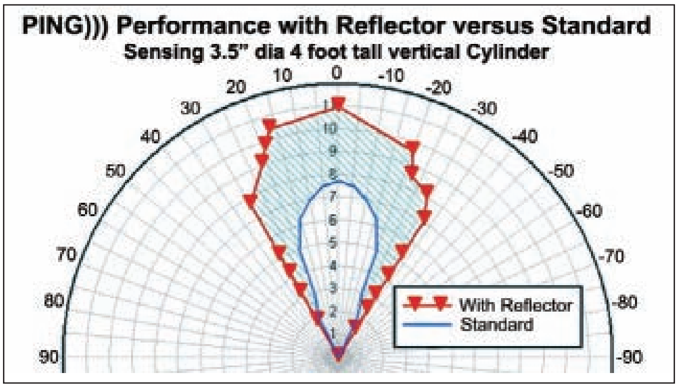


FIGURE 7. Ping))) reflector performance.

range of 40 to 42 kHz that travels to the target and is reflected back into either the same transducer that sent the pulse or into another transducer. In the same manner as the electrostatic system, a microcontroller measures the elapsed time from the time that the pulse is sent until it is received, and converts it to a digital representation of the distance measured.

The Parallax Ping)))

One of the more popular ultrasonic distance sensors used by many robot experimenters is the Ping))) by Parallax (shown attached to an optional Gadget Gangster $15 cyperbolic reflector in **Figure 6**). The ultrasonic distance sensor's sensitivity performance with and without the reflector is shown in **Figure 7**. Ultrasonic sensing cannot be compared with visual sighting because a 3.5" diameter cylinder does not reflect anywhere the same amount of sound waves back to the sensor as would a 3.5" square/flat object at the same distance, whereas the two objects would appear about the same to a video system. Soft objects as well as slanted surfaces also reflect a much attenuated signal. The sensor provides an output pulse to the robot's microcontroller at the time the sensor sends the 40 kHz, 200 µS pulse that will terminate when the return echo is received and the time differential is calculated to determine the distance.

Parallax Ping))) specifications include:
- Supply Voltage – 5 VDC
- Supply Current – 30 mA typical; 35 mA max
- Range – 2 cm to 3 m (0.8 in to 10 ft)
- Input Trigger – Positive TTL pulse, 2 µS min, 5 µS typical
- Echo Pulse – Positive TTL pulse, 115 µS to 18.5 mS
- Echo Hold-off – 750 µS from fall of trigger pulse
- Burst Frequency – 40 kHz for 200 µS pulse
- Burst Indicator LED shows sensor activity
- Delay before next measurement – 200 µS
- Size – 22 mm H x 46 mm W x 16 mm

Single-Transducer Range Finders

MaxBotix specializes in ultrasonic sensors and has an extensive line that is very appropriate for people detection, remote monitoring, industrial applications, as well as mobile robotics. With over 30 different sensors in their arsenal, I have selected and tested just a few that I feel are great for entry to mid-level mobile robot platforms.

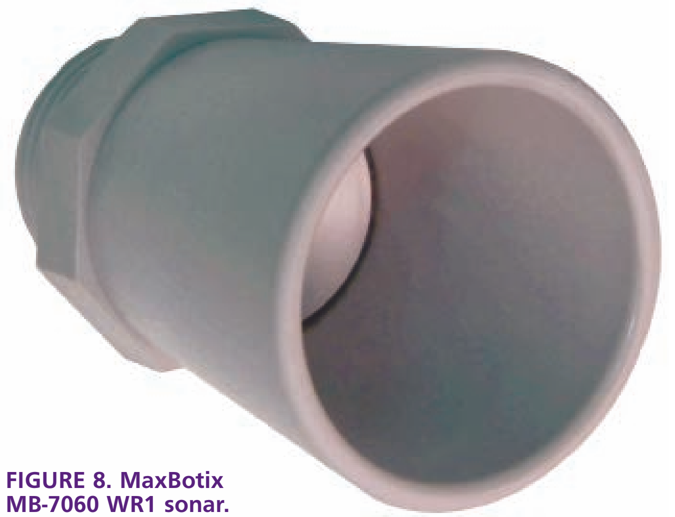


FIGURE 8. MaxBotix MB-7060 WR1 sonar.

The MB7060 shown in **Figure 8** is a weatherproof unit that is intended for outdoor use. I found it actually works in the rain. The only time that the unit began to not work was when I was out in the rain and tried facing it upwards to detect the bottom of my upper deck. This was because the cone started to fill with water and distorted the pattern.

This would be a great robot transducer for Robo-Magellan robots that can operate in foul weather. With a 42 kHz output and a fairly narrow beam width, it has a resolution of 1 cm and has a range of 25 feet. Operating on 3.0V to 5.5V @ 3.4 mA, the output into a BASIC Stamp or Arduino microcontroller can be a straight analog voltage, serial output, or a pulse width measurement. The mounting is a typical 3/4" NP thread.

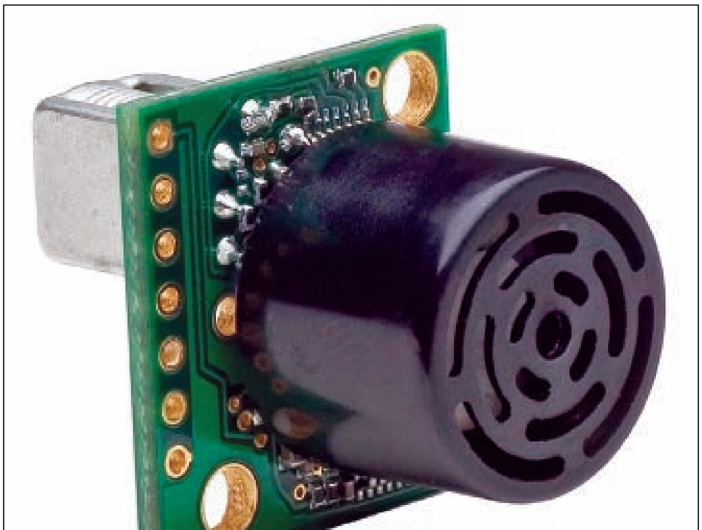


FIGURE 9. MaxBotix MB 1200 and 1240 sonar.

FIGURE 10. MaxBotix MB1010 EZ-1 sonar.

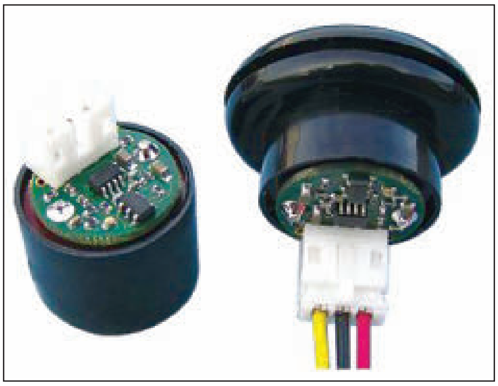


FIGURE 11. Devantech SRF01 ultrasonic range finder from RobotShop.

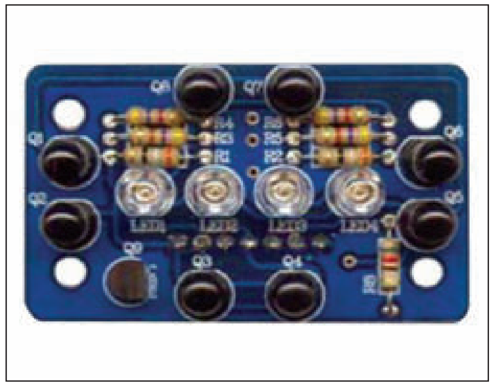


FIGURE 12. Dagu compound IR sensor from RobotShop.

The MB1200 shown in **Figure 9** is a wide beam, indoor sensor with similar characteristics to the 7060, and would make a great sensor for a mobile robot that needs a broad pattern for people and object detection. The wide beam is quite adept at detecting small targets, such as the plastic ruler stuck in my grass at 20 feet away. It is a very small sensor mounted on a 7/8" x 3/4" board; it's 5/8" long and would be great on a very small robot, as well as a larger FIRST-type build.

The MB1240 is virtually identical to the 1200, but has the narrowest beam of any MaxBotix sensor and seems to work the best with a larger, hard target. The MB1010 EZ-1 shown in **Figure 10** is their most popular indoor sensor, and the least expensive at $29.95. It is appropriate for any indoor robot. It has a lower operating voltage range of 2.5-5.5V @ 2.0 mA. The MaxBotix site has detailed specs and links for more information.

RobotShop — the large Canadian robot hobbyist supply company — has a large selection of robot sensors. Their popular British-made Devantech SRF01 ultrasonic range finder shown in **Figure 11** can be calibrated to measure from zero distance to almost 20 feet. Operating from 3.3 to 12 volts @ 11 to 25 mA, the 5/8" dia by 3/4" long transducer is a perfect match for smaller robots that need a bit of measurement range. A rubber mounting grommet is also supplied. At $35.99 US, the 2.7 gram unit communicates on a single pin serial output at 9600 baud.

RobotShop also has a unique IR sensor called the Dagu compound infrared sensor shown in **Figure 12**. Designed to be with their Dagu Mr. General robot kit, the $9 US board is a series of four IR LEDs, each paired with two focused IR transistors. The sensor board can be mounted on a $14.99 US pan & tilt mount; the kit (shown in **Figure 13**) is for use on any small robot.

Another popular IR range finder is the GP2YOA710YK module made by Sharp and handled by Acroname for $19.50. With a range of three to 18 feet operating on 4.5V to 5.5V @ 33 to 50 mA, the sensor shown in **Figure 14** includes a connector kit and has been a perfect match for

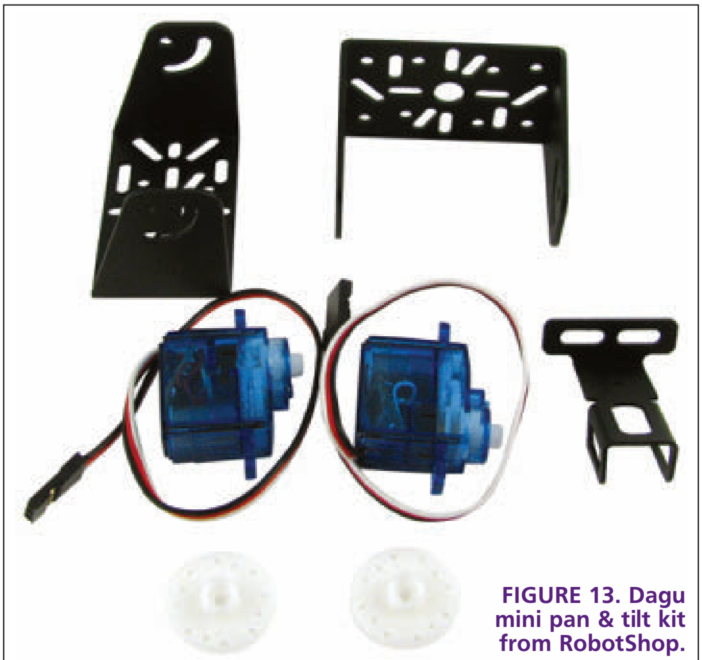


FIGURE 13. Dagu mini pan & tilt kit from RobotShop.

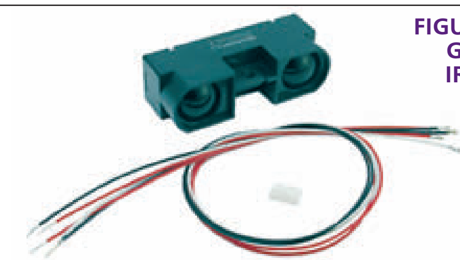


FIGURE 14. Sharp GP2Y0A710YK IR sensor from Acroname.

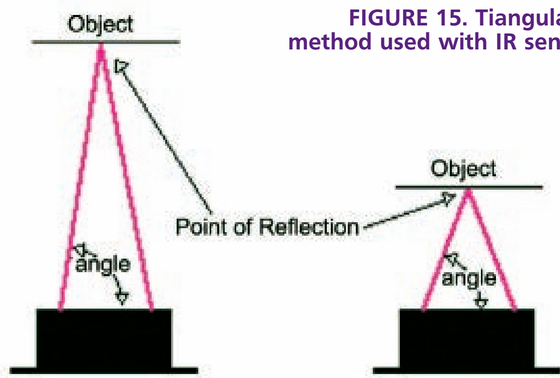


FIGURE 15. Tiangulation method used with IR sensors.

FIGURE 16. Parallax laser range finder.

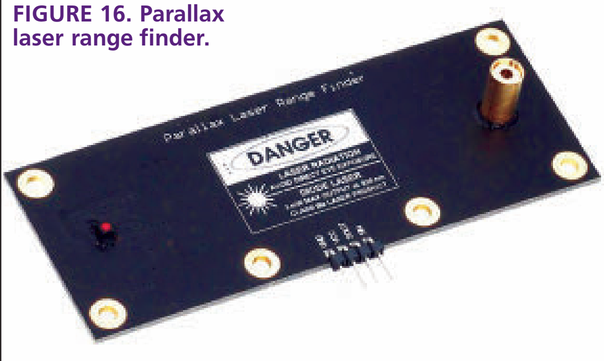
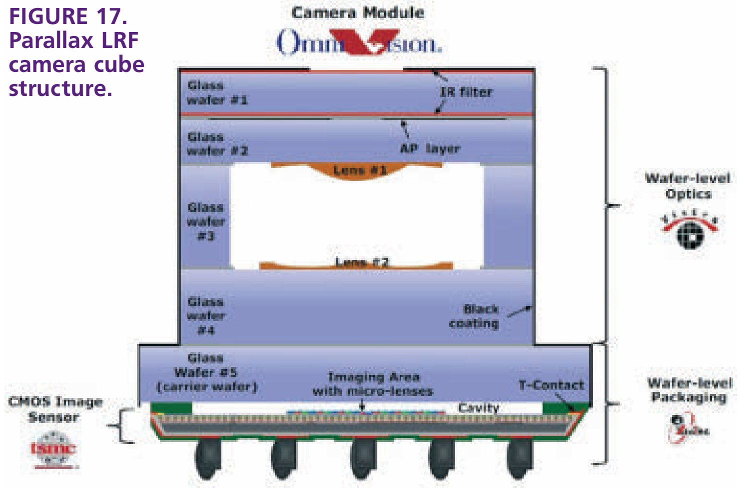


FIGURE 17. Parallax LRF camera cube structure.



many small robot platforms. **Figure 15** shows a sketch of the triangulation method this and most IR range finders use. Very complete information can be found at "Demystifying the Sharp IR Rangers" on the Acroname site.

Parallax Laser Range Finder

Many of us have longingly looked at the Hokuyo and SICK laser range finders and wish that we could shovel out $1,200 to over $5,000 for one of these quality units. They've adorned the fronts of DARPA Grand Challenge vehicles, as well as top-of-the-line industrial robots. Some builders have taken the laser range finders that hunters or golfers use and have converted them for robot use. However, Parallax — in conjunction with Grand Idea Studio — has produced an affordable $129.99 laser range finder that is made to use on robots. Shown in **Figure 16**, the 3.95" x 1.55" x 0.67" module operates off of 5.0V @ 150 mA. Communication is by an asynchronous serial 300-115,200 baud output.

The Arima laser on the right in **Figure 16** is tiny enough, but the tiny object with the red dot on the left is a full VGA camera that detects the projected red 635 nm (visible) red laser dot's position angle to determine distance by triangulation. Note in **Figure 15** how the dot's position appears at a greater angle at a closer range for the Sharp linear array. The difference here is that the camera and laser are 78 mm apart, and the detector is a VGA camera rather than a linear array (as in the Sharp sensor). **Figure 17** shows the inner structure of the tiny 1/13" camera with two filters and two lenses. Check out the Parallax, Grand Idea Studio, and OmniVision sites for some very interesting information, and also the October and November '11 issues of *SERVO* in which Joe Grand (President of Grand Idea Studio) wrote a complete description of the board.

Final Thoughts

I have covered only a small portion of the many sensors used just for object detection and range finding for robots. I will cover compass, location, and robot positional sensors in next month's column. I would like to thank my friend Bart Everett, Technical Director for Robotics at the SPAWAR Systems Center, for help with my intro about him and his friendship. Personnel at Parallax, MaxBotix, the RobotShop, and Acroname have been most helpful in assisting me with their products and specific information. It is companies like these (and others) that allow us to build the robots of our dreams. **SV**

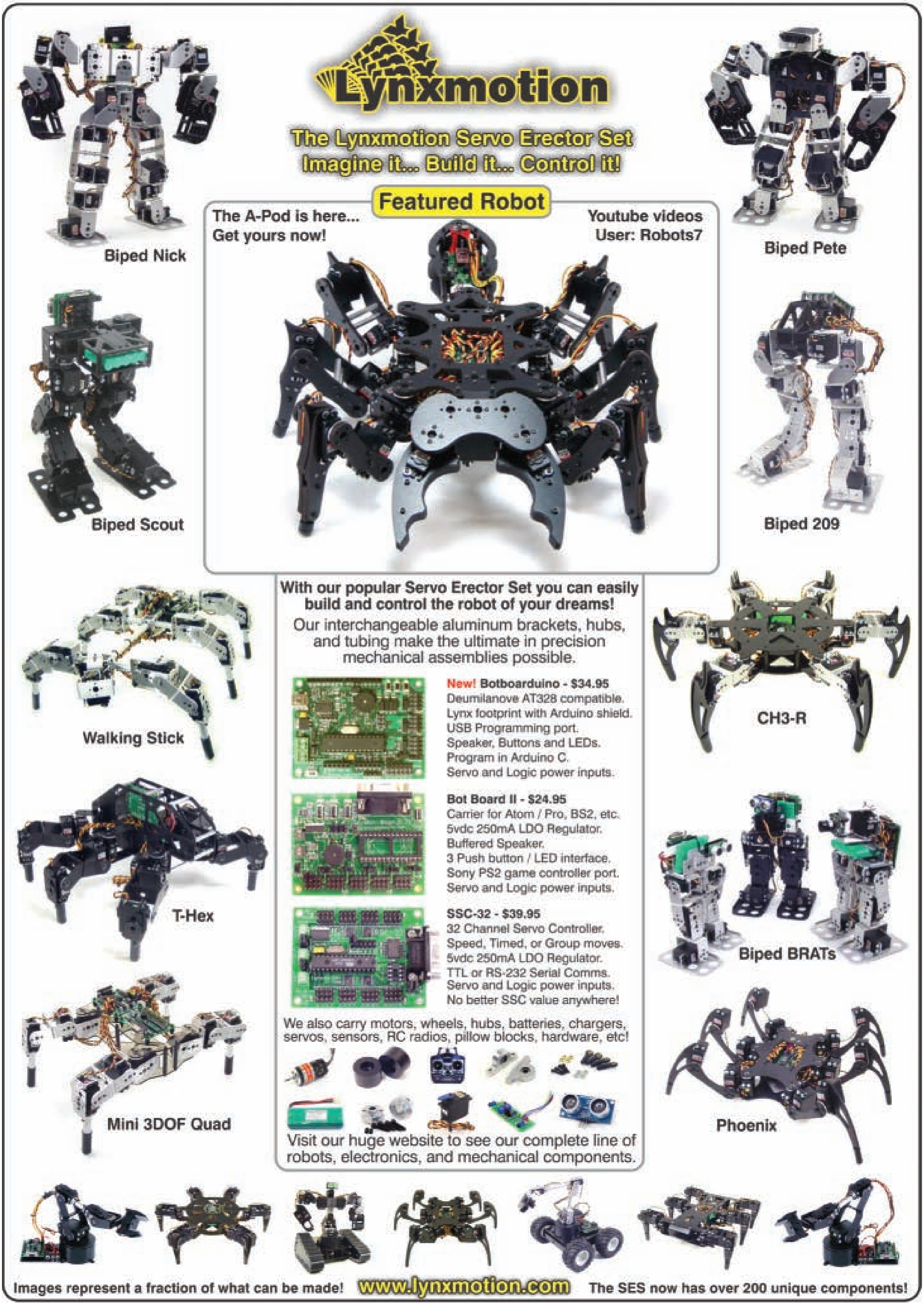
Tom Carroll can be reached at TWCarroll@aol.com.

ROBO-LINKS

For the finest in robots, parts, and services, go to www.servomagazine.com and click on Robo-Links.

Advertiser Index

To Advertise Call 951 371-8497


Lynxmotion
The Lynxmotion Servo Erector Set
Imagine it... Build it... Control it!
Featured Robot
The A-Pod is here...
Get yours now!
Youtube videos
User: Robots7
Biped Nick
Biped Pete
Biped Scout
Biped 209
With our popular Servo Erector Set you can easily build and control the robot of your dreams!
Our interchangeable aluminum brackets, hubs, and tubing make the ultimate in precision mechanical assemblies possible.
Walking Stick
CH3-R
New! Botboarduino - $34.95
Deumilanove AT328 compatible.
Lynx footprint with Arduino shield.
USB Programming port.
Speaker, Buttons and LEDs.
Program in Arduino C.
Servo and Logic power inputs.
Bot Board II - $24.95
Carrier for Atom / Pro, BS2, etc.
5vdc 250mA LDO Regulator.
Buffered Speaker.
3 Push button / LED interface.
Sony PS2 game controller port.
Servo and Logic power inputs.
T-Hex
SSC-32 - $39.95
32 Channel Servo Controller.
Speed, Timed, or Group moves.
5vdc 250mA LDO Regulator.
TTL or RS-232 Serial Comms.
Servo and Logic power inputs.
No better SSC value anywhere!
Biped BRATs
We also carry motors, wheels, hubs, batteries, chargers, servos, sensors, RC radios, pillow blocks, hardware, etc!
Mini 3DOF Quad
Phoenix
Visit our huge website to see our complete line of robots, electronics, and mechanical components.
Images represent a fraction of what can be made!
www.lynxmotion.com
The SES now has over 200 unique components!


ServoCity
making things move
powerfully reliably affordably
12V Heavy Duty Linear Actuators
Offered in six stroke lengths, from 2" to 12"
Built-in potentiometer can be easily accessed for position feedback
Aluminum stroke rod and outer tube
Metal drive gears ensure tremedous durability
25 pound thrust
(2.0" per second)
$129.99
Limit switches at both ends to prevent over-run
115 pound thrust
(0.50" per second)
$129.99
1/4" diameter mounting points that bolt up to our mounting brackets (sold seperately)
180 pound thrust
(0.30" per second)
$129.99
12V Super Duty Linear Actuators
Offered in six stroke lengths, from 4" to 24"
Power for those extra large projects!
Precision ball screw-drive
1.125" diameter extension shaft
450 pound thrust
(2.90" per second)
$399.99
Powder-coated aluminum base
1010 pound thrust
(0.98" per second)
$399.99
Stainless steel extension rod and steel protection tube
1570 pound thrust
(0.30" per second)
$399.99
7.2V - 12V Heavy Duty Linear Servos
Great for industrial applications!
To Controller
+7.2 - 12V
Ground
Digital Controller
✓ Plug directly into a receiver or any other type of servo controller
✓ Advanced digital control offers precise, accurate movements
✓ The digital controller is able to supply 5V and 1.5A to the receiver or any other type of servo controller so a seperate battery is not needed
✓ All aluminum construction ensures tremendous durability and reliability
✓ Offered in six stroke lengths, from 2" to 12"
Linear Servos just $299.99
Actuator Accessories
Using our actuator mounts allow you to easily and quickly attach the actuator to various contact points.
This switch runs a motor in both forward and reverse.
HDM250
Heavy Duty Mount
$12.99
HDM500
Super Duty Mount
$24.99
275-709
20A Actuator Switch
$9.99
Have a smartphone?
Scan this code with your bar reader app to check out ServoCity on the go!
Find us on:
www.youtube.com/ServoCity
www.facebook.com/ServoCity
Twitter: @ServoCity
To view our entire selection of products please visit us online at
www.SERVOCITY.com
Specializing in servos, radios, motors, pan & tilt systems, batteries, wiring, robot kits, hardware, electronics, actuators, gearboxes, and much more!
Copyright 1999-2012 Robotzone, LLC - All rights reserved. ServoCity is a registered trademark of Robotzone, LLC.

ANALOG™ DISCOVERY

The lowest cost, highest performance USB2-powered test and measurement instrument in its class.

Developed with:

ANALOG DEVICES

Two Oscilloscopes

- Fully differential inputs, up to 100 MSPS, 14-bit A/D, up to 16KSa/channel memory
- 1MΩ/24pF inputs, ±20V Max
- 250uV to 5V/division with variable gain settings
- Real-time FFTs, X-Y plots, complex math and measurements on all channels

Two Waveform Generators

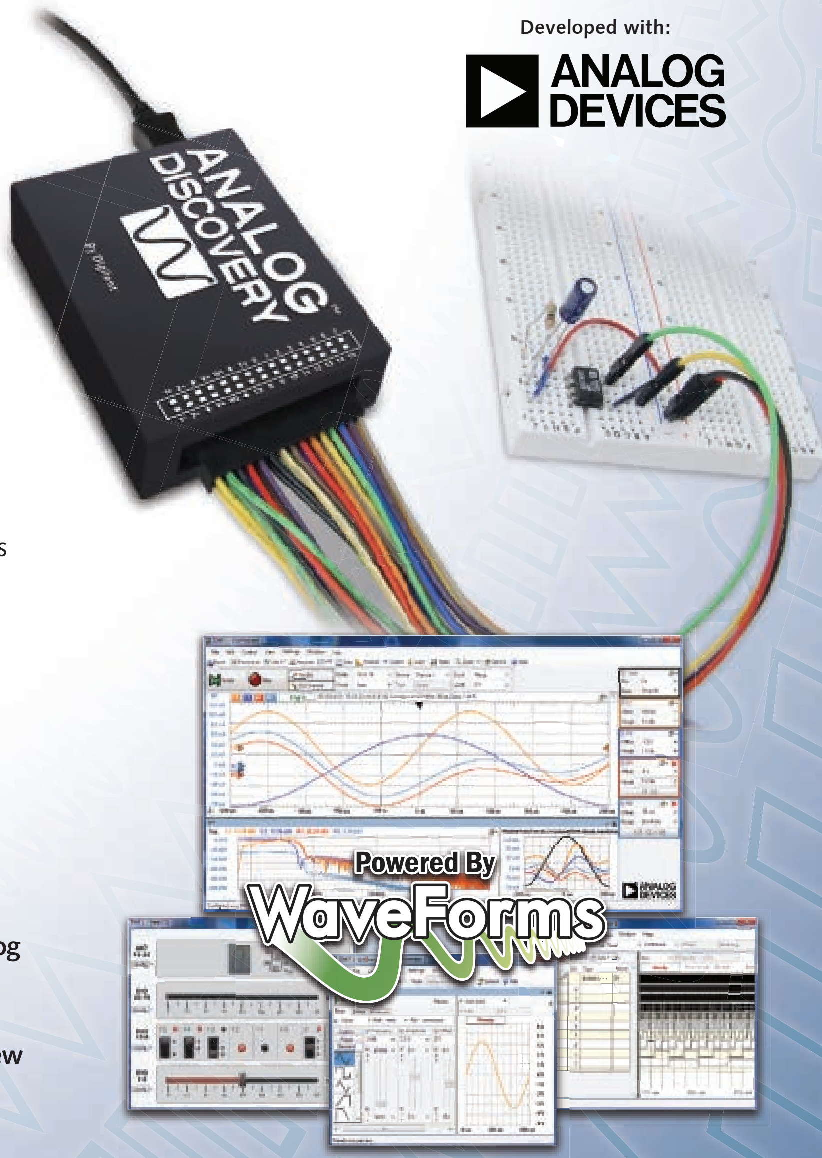
- Single-ended outputs, up to 100 MSPS, 14-bit D/A, up to 16KSa/channel memory
- Arbitrary and pre-defined waves up to ±4V
- Sweeps, envelopes, AM and FM modulation
- Bode plot feature using standard, Nyquist & Nichols coordinates

Digital I/O

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

Power Supplies

- Two fixed ±4.5V, 50mA



Great for engineers and hobbyists: Experiment with real analog circuits anytime, anywhere - you only need a PC.

FREE educational materials, including "first-touch" labs for new users, a complete college-level textbook, lab projects and more.

$99.00
(US student price)

$149.00
(Academic price)

$249.00
(List price)

DIGILENT®
BEYOND THEORY

www.digilentinc.com/analog