

Design • Build • Program • Learn • Compete

SERVO

FOR THE ROBOT INNOVATOR

www.servomagazine.com

MAGAZINE

October 2012

♦ **Give Yourself
A Wedgie!**

*Step by Step
Construction
Details To
Build Your
First Combat
Robot!*

**Troubleshooting
Tips & Tricks**

ERROR

**How To Keep
Things From
Going Wrong
With Your
Arduino-based
Robot.**



U.S. \$5.50 CANADA \$7.00



BUY THE BEST



HS-430BH



HS-5585MH



HS-5646WP



HS-5685MH



HS-7954SH



HS-7950TH



HS-7955TG



HS-M7990TH

REPLACE THE REST



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



M1

RUBBER TREADS

STANDOFFS

MOTOR/ROBOT CONTROLLER

METAL GEARMOTOR

★ USE COUPON CODE!
SERVO383

ORDER AT POLOLU.COM!

NEEDED:
LASER-CUT CHASSIS!!
SHARP DISTANCE SENSORS
ORANGUTAN CONTROLLER
SIMPLE MOTOR CONTROLLER }?
TREX JR CONTROLLER
ACCELEROMETER?

Take your design from
idea to reality



Columns

08 Robytes

by Jeff Eckert

Stimulating Robot Tidbits

10 GeerHead

by David Geer

Collaborating With Hubo

14 Ask Mr. Roboto

by Dennis Clark

Your Problems Solved Here

74 Then and Now

by Tom Carroll

The Many Ways to Control a Robot

Departments

06 Mind/Iron

20 Events

Calendar

21 Showcase

22 New Products

26 Bots in Brief

66 SERVO

Webstore

80 Robo-Links

80 Advertiser's

Index



The Combat Zone...

32 BUILD REPORT:

Testing the Prototype: Klazo —
My 1 lb Drumbot From
Kitbots.com

34 Upcoming Events

35 Clash of the Bots 3

38 The History of Robot Combat: Robot Battles at Dragon*Con



42 Give Yourself a Wedgie!

by Zachary Lytle

A wedge robot is a simple and durable design, and is the perfect platform for your first bot. Follow these construction details to build one for yourself.

50 Finding NEMO10

by Fred Eady

Every microcontroller that exists can be instructed to produce an RS-232 data stream. However, a microcontroller may not be robust enough to handle a TCP/IP stack. If your robotic device can speak RS-232, the NEMO10 can be used to translate the resultant RS-232 data stream into a TCP/IP packet.

58 Troubleshooting Tips and Tricks

How to Keep Things From Going Wrong With Your Arduino-based Bot

by Gordon McComb

Not sure where to start when you need to diagnose a problem? This guide will help!

64 The ASABE Student Robotics Competition

by R. Steven Rainwater

Each year, the American Society of Agricultural & Biological Engineers hosts a student contest which has a different agricultural theme. This year, participants had to navigate a scale model of a feedlot.



69 Parallax Elev-8 Quadcopter — Part 2: The Electronics Setup

by Bryan Bergeron

Take a look at the testing and setup of this flying robotics platform, including integration with an R/C transmitter and receiver, the selection and care of Li-Po batteries, plus the never-ending task of maintenance.

Platforms

The pioneers in robotics faced and overcame numerous challenges. For example, if they wanted to develop, say, a new navigation algorithm, they had to first build a hardware platform. They couldn't simply go online and decide between dozens of off-the-shelf flying, swimming, crawling, and walking robotics platforms.

Today, the issues are affordability, functionality, popularity, and support — largely in that order. Sometimes this isn't the best order of consideration, however.

I've gone through my share of commercial platforms including various forms of the Parallax BoeBot, a six legged crawler and arm from CrustCrawler, at least two roamers from a now-defunct company, arms from Trossen Robotics and Lynxmotion, and flying platforms from Parallax, DIYDrones, and a few offshore companies. Some have been great time savers, and a few have been time sinks.

The two carpet roamers (from a company I can't recall) were time sinks because, well, the company is defunct. That means no technical support, no spare parts, and no user community. It's hard to avoid this sort of endgame unless you know something about the company.

The issues of affordability and functionality are easily quantified. Twenty minutes on the Web will reveal the best price, and user reviews are great for assessing functionality. The Web is also a great tool for assessing the popularity of a platform. Look for articles and postings on YouTube to give you an indication of whether you're considering a dinosaur or a hot platform.

The one area in which I've stumbled lately is support. In these cost-cutting times, it's common to offload support to a user forum. Sometimes these are fantastic responsive resources. Other times, it's a pool of questions with no answers. Plus, it's difficult to determine the level of support until you have a specific problem. I've found this most commonly associated with open source software.

For example, I'm working on a robotics book and needed a platform to illustrate some robotics principles. I identified two vendors of popular platforms and purchased the platform with the best specifications and apparently the greatest popularity among experimenters. Unfortunately, the software was open source and not supported by the hardware vendor. I wasted days trolling through the forums looking for an answer to an interface question without luck. The forum leader was away on vacation or grad school, or both apparently.

In the end, I ordered the somewhat dated hardware platform with a commercial software library, and was up and running in minutes. There was a professionally edited and indexed user manual online and other support that you'd expect for a commercial product. I had to give up the bleeding edge hardware, but it wouldn't have been fair to readers to feature a cool looking but inoperative robot.

I'm not suggesting that all open source software and hardware is suspect — I'm a fan of both. I am suggesting that you take a look at who or what is behind the product. Parallax, for example, sells an open source quadcopter, and both the hardware and Propeller software are amply documented and professionally supported. It's the same with the

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

PUBLISHER
Larry Lemieux
publisher@servomagazine.com

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

EDITOR
Bryan Bergeron
techedit-servo@yahoo.com

CONTRIBUTING EDITORS
Jeff Eckert
Tom Carroll
Dennis Clark
Kevin Berry
Morgan Berry
Gordon McComb
Andrea Suarez
Jenn Eckert
David Geer
R. Steven Rainwater
Michael Jeffries
Fred Eady
Zachary Lytle

CIRCULATION DEPARTMENT
subscribe@servomagazine.com

**MARKETING COORDINATOR
WEBSTORE**
Brian Kirkpatrick
sales@servomagazine.com

WEB CONTENT
Michael Kaudze
website@servomagazine.com

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

PRODUCTION/GRAPHICS
Sean Lemieux

Copyright 2012 by
T & L Publications, Inc.
All Rights Reserved

All advertising is subject to publisher's approval. We are not responsible for mistakes, misprints, or typographical errors. SERVO Magazine assumes no responsibility for the availability or condition of advertised items or for the honesty of the advertiser. The publisher makes no claims for the legality of any item advertised in SERVO. This is the sole responsibility of the advertiser. Advertisers and their agencies agree to indemnify and protect the publisher from any and all claims, action, or expense arising from advertising placed in SERVO. Please send all editorial correspondence, UPS, overnight mail, and artwork to: 430 Princeland Court, Corona, CA 92879.

Printed in the USA on SFI & FSC stock.




open source hardware and Arduino software from DIYDrones.

Bottom line is that you have to do your homework on the quality of support available when you're considering a commercial robotics platform. If you have lots of free time and enjoy debugging hardware and software, then support may not be an issue. You'll probably become a super-user, providing much needed advice on the user forums. However, if you're looking for a platform as a means to your real interest in robotics, then make certain support is right up there with the technical specifications when you're making a purchase decision. **SV**

"THE PHANTOM DRAW"

The **KILL A WATT** meter is the best way to help you determine your actual energy draw in **ON and OFF** home appliances. **Only \$29.95**

To order call
1 800 783-4624 or online www.servomagazine.com



2011 CD ROM

SERVO MAGAZINE 2011
Volume 9, No. 1-12

On Sale Now!

Contents include

- January through December 2011 issues of **SERVO** Magazine
- Supporting code and media files from each issue
- Windows compatible Adobe Reader
- MAC compatible Adobe Reader

Only \$24.95



Firgelli
www.firgelli.com

LINEAR SERVOS

L12-R Linear Servo

- Direct replacement for regular rotary servos
- Standard 3 wire connectors
- Compatible with most R/C receivers
- 1-2ms PWM control signal, 6v power
- 1", 2" and 4" strokes
- 3-10 lbs. force range
- 1/4" to 1" per second speed ranges
- Compatible with VEX

L16 Linear Actuators

- 2", 4" and 6" strokes
- 10 - 40 lbs. force range
- 1/2" to 1" per second speed ranges
- Options include Limit Switches and Position Feedback

New!

PQ12 Linear Actuator

- Miniature Linear Motion Devices
- 6 or 12 volts, 3/4" stroke
- Up to 5 lbs. force
- Integrated position feedback or limit switches at end of stroke
- External position control available

Linear Actuator Controller (LAC)

- Will drive any Linear Actuator with position feedback
- Up to 24v and 4 Amps
- USB connectivity to drive the actuator with your computer
- Adjustable speed, limits and sensitivity

L12-NXT Linear Servo

- Designed for LEGO Mindstorms NXT®
- Plugs directly into your NXT Brick
- NXT-G Block available for download
- Can be used with Technic and PF
- Max. speed: 1/2" per sec.
- Pushes up to 5 lbs.
- 2" and 4" strokes

Available Now @
www.firgelli.com







The easiest way to go wireless...

...especially if you use this world-leading microprocessor.

Got an MSP430-based product that you're ready to take into the wireless realm? Based on proven Texas Instruments low-power RF chips, Anaren Integrated Radio (AIR) modules offer:

- > Industry's easiest, most cost-effective RF implementation
- > Low-power RF solution
- > Deployment with virtually no RF engineering experience required
- > Tiny, common footprints
- > Pre-certification/compliance to FCC, IC & ETSI (as applicable)

To learn more, write AIR@anaren.com, visit www.anaren.com/air, or scan the QR code with your smart phone.

STARTING AT \$9.99 FOR 10K OR MORE!

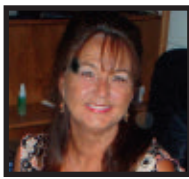
Anaren®
What'll we think of next?®

800-411-6596
www.anaren.com
In Europe, call +44-2392-232392

TEXAS INSTRUMENTS
Design Network





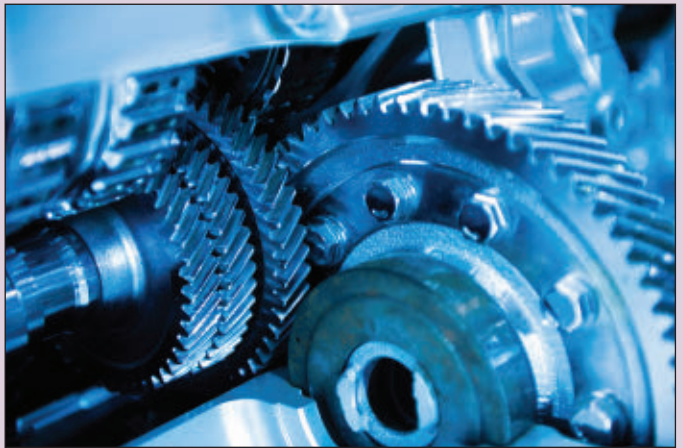


Turning Minutes Into Hours

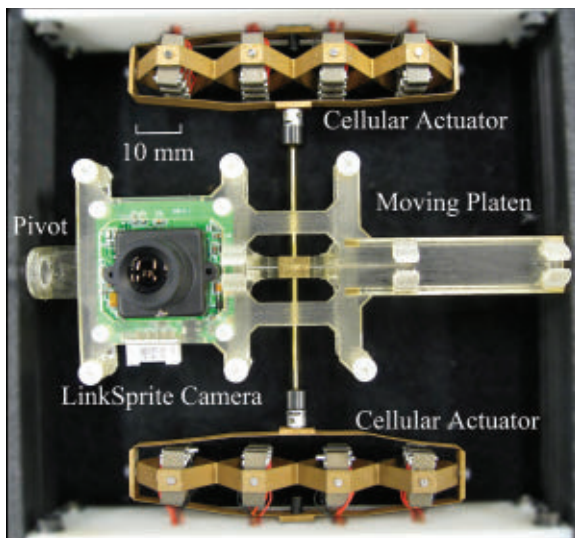
It's no secret that one of the primary constraints in mobile robotics is the limited power available from batteries. A great deal of research is aimed at improving battery performance, with progress reported in fields ranging from such novel approaches as lithium-air batteries to century-old technologies like nickel-iron (Edison) devices. Another way to tackle the problem is to improve the bot's output efficiency, which is the idea behind the M3 Actuation project — a subset of the Maximum Mobility and Manipulation (M3) program at the Defense Advanced Research Projects Agency (DARPA).

The goal is to achieve a 2,000 percent increase in overall efficiency. More specifically, it seems that the current Government Furnished Equipment (GFE) platform offers only 10 to 20 minutes of untethered operation, and DARPA wants to extend that to 200 minutes. If you think you have the know-how to come up with a solution, the reward could be as much as a cool \$5 million.

Offering some hints as to how to proceed, it was noted that "DARPA expects that solutions will require input from a broad array of scientific and engineering specialties to understand, develop, and apply actuation mechanisms inspired in part by humans and animals. Technical areas of interest include, but are not limited to: low-loss power modulation, variable recruitment of parallel transducer elements, high bandwidth variable impedance matching, adaptive inertial and gravitational load cancellation, and high efficiency power transmission between joints." The formal deadline for submitting a proposal was August 21st, but there is a six month grace period, so it's not too late to put your iron in the fire. "Contingent on the availability of funds," yours may still be selected. Details are available at www.darpa.mil, but you can just search "DARPA-BAA-12-52-1.pdf" to locate the pertinent document.



DARPA seeks a 2,000 percent increase in robot efficiency.



Camera positioning system used by researchers at Georgia Tech's School of Mechanical Engineering. (Photo courtesy of Joshua Schultz).

Here's Looking at You

Another area in which bot technology is always subject to improvement is vision, and some folks at Georgia Tech's Woodruff School of Mechanical Engineering (www.me.gatech.edu) have harnessed the piezoelectric effect to replicate the muscle motion of the human eye. "For a robot to be truly bio-inspired, it should possess actuation, or motion generators, with properties in common with the musculature of biological organisms," observed Ph.D. candidate Joshua Schultz. "The actuators developed in our lab embody many properties in common with biological muscle, especially a cellular structure. Essentially, in the human eye muscles are controlled by neural impulses. Eventually, the actuators we are developing will be used to capture the kinematics and performance of the human eye."

According to the developers, the new muscle-like action could help make robotic tools safer and more effective for MRI-guided surgery and robotic rehabilitation.

Humanoid Swimbot Introduced

Aquatic robots tend to be patterned after fish and other entities that are inherently good at swimming, but researchers at the Tokyo Institute of Technology (www.titech.ac.jp) have bucked the trend and built what they say is the first humanoid robot that can swim underwater using all four limbs.

"Swumanoid" is a 12 lb, three ft tall half-scale version of a former Japanese Olympic swimmer who apparently wants to remain anonymous. The aquabot's mission is to figure out how to create the least amount of drag while swimming, thus potentially teaching us how to do a better job of it. For example, the researchers will study how pulling its arms straight through the water compares with using a zig-zag pattern. Although powered by 20 motors, Swumanoid plods along at a measly 0.64 m/s (2.1 fps), compared to Nathan Adrian's 2.1 m/s (6.9 fps) in the summer Olympics 100 m freestyle. Its creator says that Swumanoid 2.0 — due sometime next year — will improve upon that.



Tokyo Tech's "Swumanoid" (not ready for the Olympics).

Robotic Architectural Printer

We had to view this item with a bit of skepticism, especially given that the documentation includes some obvious Photoshop concoctions. According to the website of the Institute for Advanced Architecture of Catalonia (www.iaac.net), it's for real. Building on the concept of 3D printing, some students came up with the idea of building architectural structures by spraying a mixture of dirt and a liquid binder to create a variety of solid forms and shapes. Moreover, the "Stone Spray" process is carried out by robots, can be powered entirely with solar energy, and employs nontoxic



"Stool" created from beach sand and wire using the Stone Spray process.

PolyPavement (www.polypavement.com) as the binder. It's a little early to think about spraying yourself a new beach bungalow, though, as the process seems to still have a few bugs in it (perhaps literally). For example, the "stool" shown in the photo required an internal wire skeleton, and it took four hours to create and solidify it, even though it measures only 200 mm in all dimensions. Such a structure is said to be "structurally strong and can support not only itself but even bear a load," but how much of a load is still in question. Still, it's an interesting concept.

Art Imitates Imitation Life

Finally, as we have observed upon occasion, whenever you mix art, robotics, and public funding, strange things can happen. In this case, it is a sculpture by Czech artist David Cerny, consisting of a six ton, 1957 double-decker bus that does push-ups. Also featured are video projections in the windows and recorded groaning sounds. The work of art was created as a tribute to the London Olympic games and is located outside of the Czech Olympic House (which usually havens the Business Design Center) in London. Explaining the point of the artwork, the artist noted that the push-up "is training for sport activities but at the same time it is also punishment in armies and prisons. So, the push-ups are a very universal physical activity ... It is in a way very ironic." Now don't you feel enlightened?

Previous Cerny creations include an object depicting Bulgaria as a squat toilet, a statue of Saddam Hussein floating in formaldehyde, and a display encouraging people to kill each other to control population growth. **SV**



"London Boosted," a piece of robotic art by David Cerny.



GEER: HEAD

by David Geer

Contact the author at geercom@windstream.net

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

Collaborating With HUBO



Dr. Youngmoo Kim and Dr. Paul Oh of Drexel University are working with HUBO robot platforms from the Korea Advanced Institute of Science and Technology (KAIST) to help advance the development of humanoid robotics. By sharing seven standardized HUBO robots among Drexel University, MIT, Carnegie-Mellon, Virginia Tech, the University of Southern California, Ohio State, Purdue, and Penn State, each school is able to work on and improve differing types of algorithms, programming, behaviors, and capabilities on its robot while ensuring that progress at one school translates to the robots at the other schools. A grant from the NSF made it all possible.

At Drexel, Dr. Kim (the Director of the

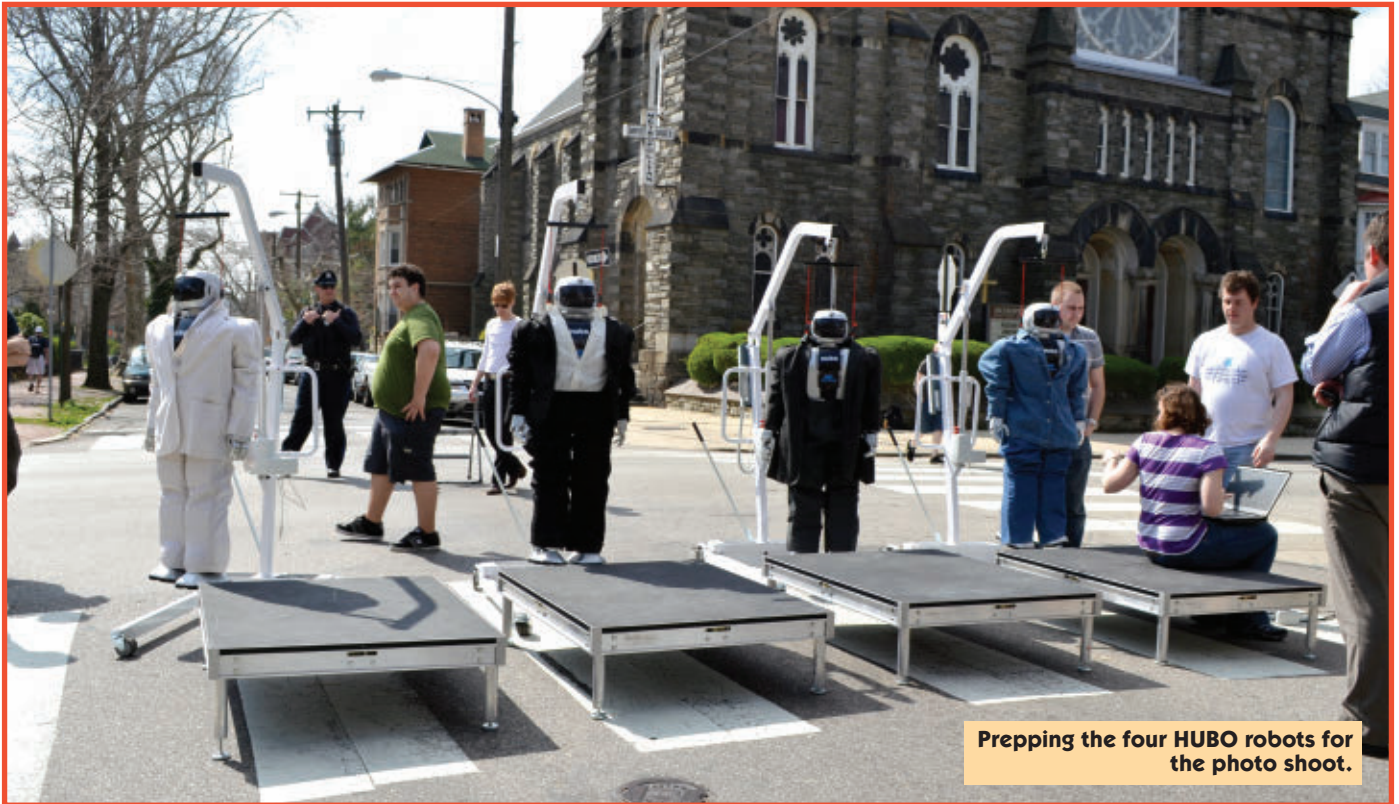
Music and Entertainment Technology Lab) is working on musical development on the HUBO platform which we'll take a look at later in this article.

The HUBO Platform

The first HUBO was part of a previous project that Dr. Oh kicked off. It was a multi-million dollar project to bring HUBO from KAIST to Drexel, and to trade students between KAIST and Drexel to share knowledge and training on HUBO.

The HUBO robot platform is a four foot, three inch tall,

fully actuated humanoid robot with similar joint movement to that of humans. HUBO's legs, arms, hands, fingers, and thumbs work about the same as human's do, as well. HUBO and HUBO 2 are among the latest robots in the HUBO series, showcasing a slighter design consisting of a polycarbonate frame and an aluminum endoskeleton. The new HUBO is taller and lighter. For this version, KAIST made improvements in the mechatronics for greater reliability.



The new HUBO is enabled with much more human-like motor skills including faster, more realistic arm movement and stretching legs that mimic human bi-pedal movement more closely. Natural human walking requires less energy than typical robot walking, so there is an energy savings. This is calculated using the Zero Moment Point trajectory calculation. The HUBO robot now has an increased walking speed of 1.4 km per hour and can now run at up to 3.6 km per hour.

The new HUBO platform leverages the following technologies: a smart power distributor and lithium polymer battery that is +48V, 8A; a shape adaptive hand that is tendon-driven with five fingers (one degree of freedom per finger) and an F/T sensor at the three degree of freedom wrist; an IMU (two axis) with Kalman filtering and a rate gyro and accelerometer; a two channel BLDC controller that is 90 mm x 65 mm with a 16-bit microprocessor; a CAN interface; an A/D converter; over-current protection; automatic return to initial position; a BLDC motor amplifier that is 90 mm x 90 mm 200W and 48V; and a full bridge MOSFET.

Drexel received one HUBO initially and six HUBO 2s later on. The last six are identical but they each have individual quirks. One has an ankle motor that is a little off; another has a different computer inside, but the platform is otherwise the same, says Dr. Kim. Drexel has given each one a number to tell them apart.

The first of the two HUBOs (introduced in 2004) did not have the power efficiency, custom hardware, and circuit

boards available on the newer HUBO 2. The HUBO 2 operates on battery power for nearly an hour. KAIST has lowered the center of gravity on this robot so that it will not fall over so easily. HUBO 2 has increased motor efficiency and faster processors in its internal computer. One of those processors is a standard PC 104 platform base. There is room for two processors in the robot: one for command, control, and movement; and the other for sensing, cognition, and higher-level tasks. "The challenge is in having the two processors communicate with each other," comments Dr. Kim.

The Research

Dr. Kim has developed algorithms for tracking the beat in music based on the audio. This is a hard problem to solve. It is difficult to teach a computerized robot to do that in a robust way. "We are working on having the HUBO robots play simple percussion instruments that we have built out of PVC pipe. We are working on having them play the notes and listen to the notes to see whether a note was a good note. This involves audio and force feedback capabilities," explains Dr. Kim.

Dr. Kim is using machine learning and as HUBO makes lots of differing strikes on the instruments, Dr. Kim presents some to the robot and tells it that these were good or bad notes. The force feedback sensing from the robot's hand and wrist together with the sound of the note tell the characteristics of the different notes.



Photo from the day that Drexel University set up four of the HUBO robots to pose as if crossing the street like the Beatles from their Abbey Road album.

Some videos of HUBO on YouTube include the robots playing “Come Together” from the Abbey Road album by the Beatles. Dr. Kim and colleagues took a photo of the HUBO robots crossing the street so that it looked like the Abbey Road album cover. The photo appeared on the cover of the Drexel online magazine last spring.

While Dr. Kim has developed the beat tracking capabilities for HUBO, he has not perfected the robot’s dancing capabilities. “The HUBOs cannot dance in a good, realistic way yet. They can perform programmed gestures at the right time. The goal is to have them perform the right dancing gestures linked together in the right way with some creativity,” says Dr. Kim.

The gesture and motion planning for dancing is a very complicated problem that is similar in complexity to the motion planning for having robots work with tools or operate doors and latches.

The long-term goal of Dr. Kim’s research is to have the robots play musical instruments alongside human musicians in an ensemble. “We eventually want robots to be autonomous enough to be personal assistants, help people get around, do dishes, clean up, and fold laundry. We are taking steps in that direction, as well,” comments Dr. Kim.

The HUBO robot is physically capable of these things. It can deal with curbs and stairs (walking), and can use its fully actuated hands to pick up light objects, including tools. “These are all individual capabilities. Linking them together takes baby steps,” expressed Dr. Kim.

Advances

Most recently, Dr. Kim and colleagues have made a lot of strides in the listening capabilities of the HUBO 2 robot. As the robots move, they make a certain amount of noise, so while they are trying to do something such as move or interact with other musicians, if someone is talking in the background as well, the motor noise makes it difficult for the robot to discern one sound from another. “We have worked out a couple of different algorithms to rid the noise,” explains Dr. Kim. He is also using better microphones that are directional, and filter out unwanted noise while filtering in the desired sounds. “We are working on a

microphone array for HUBO for this," affirmed Dr. Kim.

He is also employing a two-lense binocular camera with a high frame rate of 45-60 fpc to enable the high tracking rate that is necessary for some of these tasks.

Dr. Oh is working on mobility tasks for HUBO, such as climbing stairs and dealing with uneven terrain (though the robot is not very good at the latter task just yet). Dr. Oh is also developing the HUBO platform for skills like human-robot "teaming," such as helping a human carry a table. The robot will need to walk together with the human to move objects from one spot to another.

Dr. Oh is brainstorming on a lot of different strategies for that, and is also working on the challenge of having two robots move a table together without the aid of a human.

In addition, the researchers are working on speech recognition for HUBO using a standard speech recognition tool, though there are issues around acoustics and — again — noise. USC is working on adding touch-based sensing together with a company called SynTouch.

Conclusion

According to Dr. Kim, it will take about five to 10 years to get the HUBOs to play musical instruments together with

Resources

Dr. Youngmoo Kim
<http://music.ece.drexel.edu/people/ykim>

Dr. Paul Oh
<http://dasl.mem.drexel.edu/people.php>

YouTube video including HUBO 2 and related work
www.youtube.com/watch?v=dJRzJt-Dwic

Drexel Autonomous Systems Lab
<http://dasl.mem.drexel.edu/>

HUBO 1 accompanying music
www.youtube.com/watch?feature=endscreen&NR=1&v=L21_YZvd6Ck

HUBO 2 robots dancing
www.youtube.com/watch?v=PLn_BGfD84g

human musicians. "The more general problems like autonomous robot assistance, such as HUBO moving around in someone's home — that is a 10-20 year range problem. And everyone having a complex, commoditized humanoid robot in their home to help with multiple complex tasks, well, that is a 20-40 year range problem before we solve it," Dr. Kim concluded. **SV**



PCB-POOL
 THE ORIGINAL SINCE 1994
 Beta LAYOUT

FREE Stencil
 with every prototype order

EAGLE order button
pcb-pool.com/download-button
 20% off! on your first order

Call Tyler: 1 707 447 7744
sales@pcb-pool.us

PCB-POOL® is a registered trademark of **Beta LAYOUT**
www.pcb-pool.com



ROBOT POWER
www.robotpower.com

Extreme Robot Motor Control

High-Current motor control for Arduino™

Software and plug compatible with our MegaMoto shield

MEGAMOTO Plus Shield

- ◆ 30A Continuous with fan - 40A peak
- ◆ Current sensor outputs can be sent to any analog pin
- ◆ Stack up to three units on one Uno or compatible
- ◆ Independent half-bridges can control brushless/steppers too!

MADE IN THE USA

Scorpion XXL


- ◆ 7V - 28V
- ◆ Dual 40A+ Peak H-Bridges
- ◆ Current/Temp limiting
- ◆ R/C inputs w/mixing
- ◆ Optional enclosure
- ◆ 3.25" x 2.25" x 5"

Coming Soon!

The Vyper

- ◆ Single 120A+ H-Bridge
- ◆ 8V - 42V
- ◆ R/C inputs w/mixing (2 units req.)
- ◆ Analog pot input
- ◆ Supports 8 AWG wires!
- ◆ Optional enclosure
- ◆ 2.85" x 2.25" x 1.75"

Phone: 253-843-2504 ♦ sales@robotpower.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

As I write this, I have just returned from the Microchip Master's Conference 2012, and boy was that a blast. I should qualify that statement. It was a blast for a seriously geek crowd of hardware and firmware engineers that really love their work of making things that do stuff. The chipKIT folks at Digilent had some very fun classes and I discovered a seriously cool product series by Roving Networks that puts an entire Wi-Fi stack on a very affordable module. I'll be writing about that in a little bit, but first, I have some observations to make. The crowd going to these events is looking "grayer." We need more younger engineers coming into the "club" which leads me to my next set of musings ...

For many years now, I have been lamenting to anyone who would listen (a shrinking crowd) that today's kids aren't interested in how things work and how to make them. They are only interested in using the cool new gadgets that our technology wizards are creating for the consumer market. My concern was further fueled by an article in EETimes by Bill Schweber in 2008 which asked the question: "Are we becoming a 'Cargo Cult'?" Schweber goes on to describe a cargo cult which originated with a story about Pacific Island natives in WWII who built dummy replicas of radios, antennas, and microphones to call for

planes to land with their desired cargo, just like they saw the military forces doing. The natives didn't understand what was behind the technology they saw being used; they just saw things happen when it was used. Does that sound familiar to you with respect to our smartphone carrying society?

As more and more high technology is created by a shrinking pool of those who understand how to create and maintain these nearly magical marvels, are we allowing ourselves to simply go through the motions of understanding how our society's underpinnings work? That is a kind of scary thought, isn't it? There are plenty of Orwellian scenarios to play out in my mind dealing with that concept. How about yours?

Now that I have ruined your night's sleep, let me try to allay your fears! In the last few years, I've seen a reversal of that trend forming. I am referring to — of course — the "maker" renaissance that has been born in many places. The first (pun intended) origin I believe is Dean Kamen's FIRST organization, which spawned the First LEGO League. FLL hopes to create the next generation of our society's engineers and

entrepreneurs. Later, came the rise of the maker folks from Make Magazine and the Maker events that happen all around the world. The maker movement is all about creating new things of beauty and usefulness without needing huge corporate backing.

I place the Arduino "cult" firmly in the midst of the maker movement for bringing together ideas from many places to create the Arduino platform that help those who didn't start out as embedded engineers to make tools and art that use embedded engineering tools of the trade. Bravo. Now, we have lots of spinoffs of the Arduino scene for hardware and creative aids like the on-line site Instructables to boost that maker spirit out there.

This movement has given me back hope that our society can still improve and move forward from many creative origins and has not given up our future to the faceless mega-corporations of the world. Okay, so maybe that was a bit strong, but still, the small operations out there catering to the maker movement have been embraced and are successful because our culture has NOT become a cargo cult. We still seek to create and to understand. I tell my children that not all "magic" comes from fantasy books. The technological creations of our age are just as wondrous as any spell from a magic wand in a book. 'Nuff said!

New and Cool From the Conference

Now, on with the nifty new things that I saw at my Geekfest, er, conference, that is ...

The Diligent people who created the UNO32 and MAX32 Arduino compatible boards, along with the Rutgers gentlemen who helped create the *MPIDE* extension to the Arduino IDE, have been busy making the chipKIT boards even more compatible with Arduino scripts and hardware. Diligent has come up with both Internet client-side and server-side APIs for their chipKIT boards. This library works with a variety of their Pmod Internet boards and their UNO32 Wi-Fi shield, as well. It is my understanding that they have kept that library true to the original network library created for Arduino, and corrected and extended it for the faster UNO32 and MAX32 boards. I was drinking from the fire hydrant there, so I'll do some articles in the future that go into more detail with examples and code in coming months. The Diligent folks have also extended their Arduino-like chipKIT boards — the Cerebot — to use more of the Arduino libraries. In general, if you create Arduino scripts using the standard hardware abstraction layer and don't optimize for the AVR hardware, your scripts will work on both Arduino and chipKIT hardware. I like that. I was seeing a lot of things related to networking and especially wireless networking. Go to www.digilentinc.com to get more information about this and other fun things these guys are doing.

The other bug I'd like to drop in your ear is the Roving Networks Wi-Fi module. This is an entire Wi-Fi Internet stack on a tiny board that you talk to over a digital UART or SPI connection at up to 2 MB/s. Imagine giving your robot a Wi-Fi connection to your laptop or other devices by adding a board by simply plugging in a motor controller. I've GOT to get me a couple of these! To get one for



Figure 1.

yourself to play with, order the developer's kit (part number RN-174K) from www.microchipdirect.com or from the Roving Networks site at www.rovingnetworks.com.

That was all fun, but I'm going to go back to last month's topic for a moment and the wireless PS2 controller quest.

I'll write about these and other devices that are really handy and easy to use for our robot projects in later columns, so stay tuned!

The Continuing Saga of the Quest for the Perfect PS2 Wireless Controller

When last we met, I had found my first working wireless PS2 controller by buying a new GameStop Predator S-Type unit that was all digital except for the joysticks. At that time, I had bought (but not yet received) a Lynxmotion wireless PS2 controller for US\$19.99 — the same price as the GameStop unit. The Lynxmotion PS2 controller was larger, meatier, and was (sigh) the proper Darth Vader black color (see **Figure 1**). The one down-side to the Lynxmotion unit is that its wireless module was the largest of all of the

units I've tested. However, *every single button was fully analog pressure sensing* except for the "Select" and "Start" buttons. The harder you pressed, the bigger the number that was returned. If you are looking for a controller that responds to the emotional "pressure" of its holder, this would be the unit! All of the buttons also gave their digital values to the button registers, so you could use this controller any way you wanted. It even has the "rumble" motors in it, so you could have your robot give haptic feedback when it hits something or wants your attention. (Hmm, that sounds like a neat idea!)

This month, I'm going to modify the Arduino PS2 library demo test code to read the joysticks and drive a small tracked vehicle around. The final use for this device will be to drive one of my humanoid bipeds — I'll publish details about that when I get it done. For the sake of *this* column, I'll be driving a hacked toy chassis around. I don't remember what toy this was or even when in my evil hacking days I stripped away all but the drive train, but it was on top of my junk box and had two motors so I used it. It even has tank treads which I think are cool, so that sealed the deal. Since I'm using an Arduino, my first thought was to call this project the YAAR for

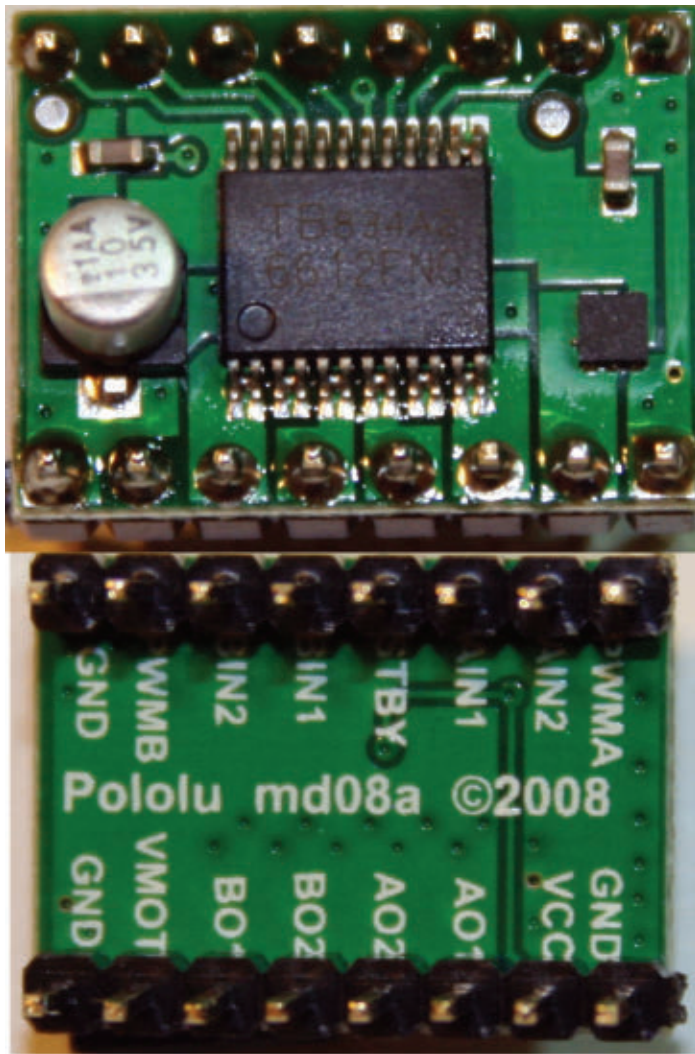


Figure 2.

Yet Another Arduino Robot. (But, wait! I'm driving it and I firmly reject the idea that any remotely controlled vehicle is a robot! Not to worry. The acronym can still be Yet Another Arduino Remote vehicle. So, YAAR it is.)

As with last month's PS2 project, I'm using a SparkFun Arduino Pro (a 5V Arduino running at 16 MHz) and the Arduino 1.01 IDE. This script will work with any Arduino Uno board, as well.

To make this project, all I need to add is a vehicle chassis and a low voltage dual motor controller. I like to fly electric RC aircraft, so I have no shortage of two and three cell lithium-ion battery packs lying around, but I didn't have any simple-to-use reversing low voltage motor controllers. Time to hit the Web. I found the Pololu TB6612FNG motor controller for about US\$9. It is a one amp continuous 3A peak controller for motors in the 4.5V to 13V range and it is TINY! **Figure 2** shows the board with some connectors attached. This comes semi-assembled; you have to install whatever connectors or wires you want to use.

I simply used the included "Berg" headers for this project since I knew that I was going to be using a solderless breadboard to simplify my wiring needs. The pins are very clearly labeled on the bottom of the board (again see **Figure 2**). To make my little demonstration vehicle, I needed to use four I/O lines for the PS2 controller and six I/O lines for the motor driver board. That leaves just two digital I/O lines: the serial port and the analog lines. Not much left to make a robot with! But, as I said, this is a "proof of concept" design, so I'll leave the I/O line optimization

Listing 1.

```
#include <PS2X_lib.h> //for v1.6

// Left motor driver I/O lines
const int lDir1 = 2;
const int lDir2 = 3;
const int PWML = 5;

// right motor driver I/O lines
const int rDir1 = 4;
const int rDir2 = 11;
const int PWMR = 6;

// create PS2 Controller Class
PS2X ps2x;

/*
 * You must have the controller turned on
 * when this program starts.
 */
int error = 0;
byte type = 0;
byte vibrate = 0;

// Full forward == 0, full reverse == 255
byte leftSpeed, rightSpeed;
```

```
byte leftDir = 0;
byte rightDir = 0;
```

```
void setup()
{
  pinMode(lDir1, OUTPUT);
  pinMode(lDir2, OUTPUT);
  pinMode(rDir1, OUTPUT);
  pinMode(rDir2, OUTPUT);
  pinMode(PWML, OUTPUT);
  pinMode(PWMR, OUTPUT);

  analogWrite(PWML, 0);
  analogWrite(PWMR, 0);

  Serial.begin(57600);

  // GamePad(clock, command, attention, data,
  // Pressures?, Rumble?)
  error = ps2x.config_gamepad(7,9,8,10, true,
  true);

  if (error == 0){
    Serial.println("Found Controller, configured
    successful");
  }
  else if (error == 1) {
```

as an exercise for the student (smile).

Figure 3 shows the (literal) rats nest of wiring I needed to make my remote vehicle. People ask why it is so hard to make a robot — all the wires, man! All the wires!

The last little bit of hardware I used to complete my design is a Hobby People two-cell, 25C 850 mAh Li-Po battery pack. At full charge, this pack is about 8.4V which is plenty of voltage for my vehicle motors and still well below the 13V maximum for the motor driver chip I'm using. Since most motor controllers don't sense the voltage and shut down before damaging the pack, remember *you* need to watch your lithium-ion pack's voltage. If the battery is drained below about 2.5V (depends on the

Arduino Pin Number	Function
7	PS2 SPI clock
8	PS2 ATN line
9	PS2 CMD line
10	PS2 Data line
2	Dir1A for the left motor
3	Dir2A for the left motor
5	PWMA for the left motor
4	Dir1B for the right motor
11	Dir2B for the right motor
6	PWMB for the right motor

Table 1. Arduino I/O pin assignments.

```

    Serial.println("No controller found, check
    wiring, see readme.txt to enable debug.
    visit www.billporter.info for troubleshooting
    tips");
}
else if(error == 2) {
    Serial.println("Controller found but not
    accepting commands. see readme.txt to enable
    debug. Visit www.billporter.info for
    troubleshooting tips");
}
else if(error == 3) {
    Serial.println("Controller refusing to enter
    Pressures mode, may not support it. ");
}

type = ps2x.readType();
switch(type) {
    case 0:
        Serial.println("Unknown Controller
        type");
        break;
    case 1:
        Serial.println("DualShock Controller
        Found");
        break;
    case 2:
        Serial.println("GuitarHero Controller
        Found");
        break;
}

void loop()
{
    /*
    * You must Read Gamepad to get new values
    * you should call this at least once a
    * second
    */

    if(error == 1) //skip loop if no controller
        //found
        return;

    /*
    * Read controller and set large motor to
    * 'vibrate'
    * this will set the large motor vibrate
    * speed based on
    * how hard you press the blue (X) button.
    */
    ps2x.read_gamepad(false, vibrate);
    vibrate = ps2x.Analog(PSAB_BLUE);

    /*
    * Get the stick Y axis. 128 is center,
    * leave

```

```

    * some dead band around the center and
    * adjust the
    * values to send to the PWM around 0-126.
    * Not too
    * fast.
    */
    leftSpeed = ps2x.Analog(PSS_LY);
    // left
    if (leftSpeed > 130) {
        leftSpeed -= 128;
        leftDir = 1;
        // reverse
    }
    else if (leftSpeed < 126) {
        leftSpeed = 126 - leftSpeed;
        leftDir = 0;
        // forward
    }
    else {
        // call it stopped
        leftSpeed = 0;
        leftDir = 1;
    }
    // Now set the driver direction lines and
    // PWM
    digitalWrite(lDir1, leftDir & 0x01);
    digitalWrite(lDir2, ~leftDir & 0x01);
    analogWrite(PWML, leftSpeed);

    rightSpeed = ps2x.Analog(PSS_RY);
    // right
    if (rightSpeed > 130) {
        rightSpeed -= 130;
        rightDir = 1;
        // reverse
    }
    else if (rightSpeed < 126) {
        rightSpeed = 126 - rightSpeed;
        rightDir = 0;
        // forward
    }
    else {
        rightSpeed = 0;
        // stop
        rightDir = 1;
    }
    // Set motor driver directions and PWM
    digitalWrite(rDir1, rightDir & 0x01);
    digitalWrite(rDir2, ~rightDir & 0x01);
    analogWrite(PWMR, rightSpeed);

    // Some debug output
    //Serial.print(leftSpeed, DEC);
    //Serial.println(rightSpeed, DEC);

    delay(50);
}

```

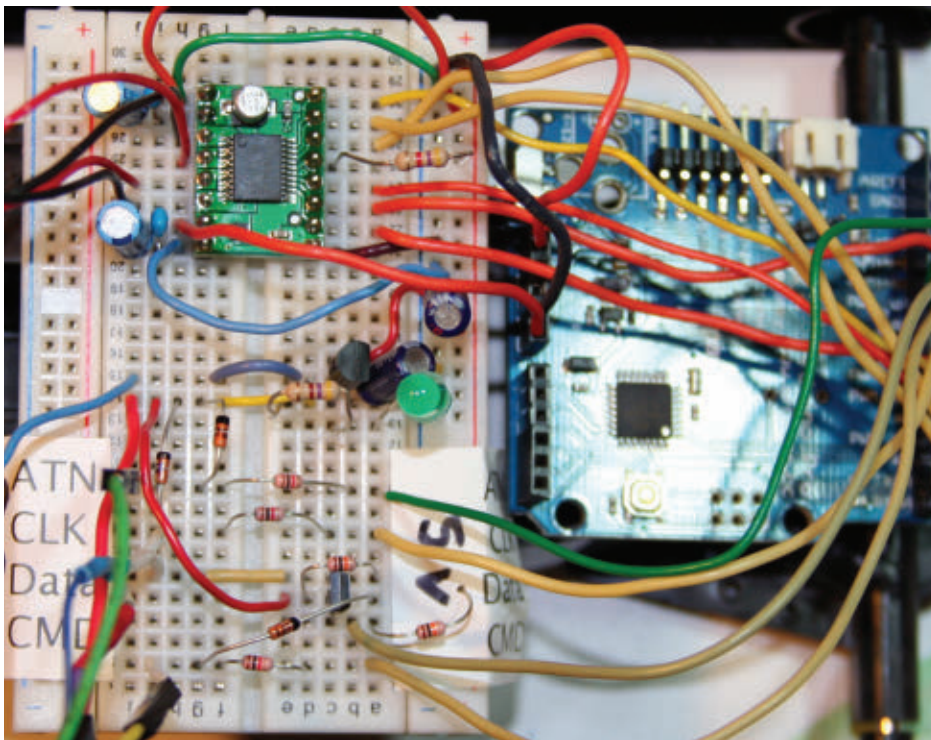



Figure 3.

handle this most basic of remote control operations. I am only looking at the Y axis joystick values so that I can drive my vehicle like a tank. Because I thought it was cool, I've left in the rumble ability. I could put a collision detector sensor on the thing and have it "rumble" my controller when something gets close. Cool, a haptic feedback device! **Listing 1** shows my code. It is cribbed from the demo code that comes with the PS2X Arduino library and modified to drive the motors on YAAR.

I've included the entire script here since — this being an Arduino — I don't need to explain how to set up PWM registers, set timers, mask I/O bits, and all the other "stuff" that an embedded program has to do before anything useful happens. The Arduino API hides those details behind its

battery), then the pack is dead and not recoverable.

Now, it's time to look at the program that you need to

hardware abstraction layer, so you can focus on just doing the job.

Model 324

WORLD'S MOST VERSATILE CIRCUIT BOARD HOLDERS

Our line of Circuit Board Holders add versatility & precision to your DIY electronics project. Solder, assemble & organize with ease.

MONTHLY CONTEST
Visit us on Facebook® to post a photo of your creative PanaVise project for a chance to win a PanaVise prize package.

PANA VISE®
Innovative Holding Solutions

7540 Colbert Drive • Reno • Nevada 89511
1 (800) 759-7535 | www.PanaVise.com

Model 201

VISIT US ON

AP CIRCUITS

PCB Fabrication Since 1984

As low as...

\$9.95

each!

Two Boards
Two Layers
Two Masks
One Legend

Unmasked boards ship next day!

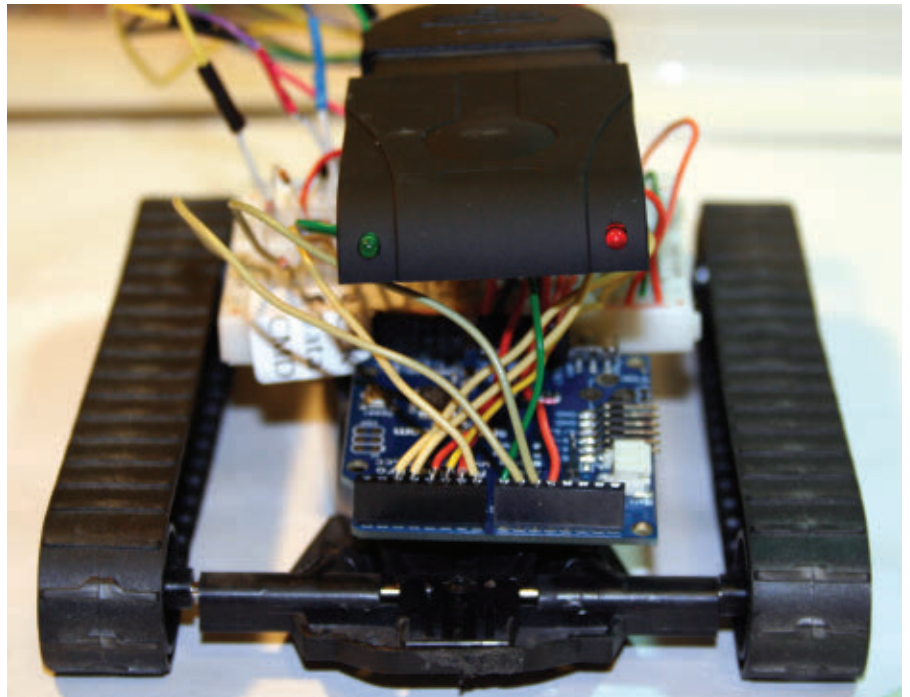
www.apcircuits.com

Figure 4.

The first code block defines all of our I/O lines and the PS2 controller object. The *Setup()* block configures the I/O lines and discovers which PS2 controller you are using. Finally, the *Loop()* block is the code that controls everything in real time. Notice at the top of the loop I left in a way to make the controller rumble if you push the blue "X" button on the right of the PS2 controller. Push it a little and it rumbles a little; push it a lot and, well, you get the idea. I left that in so you could experiment with having a sensor give some user feedback.

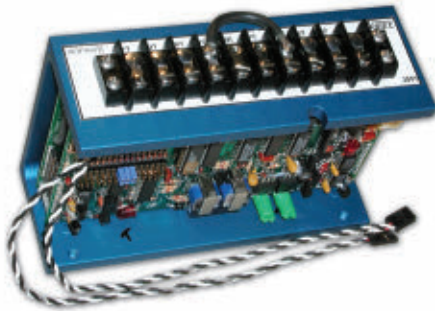
To get everything to work properly, turn on the PS2 controller, then power up the YAAR after you connect the battery. The blinking red LED on the wireless module will stop blinking and turn solid when the remote and the module have connected. Now go drive! **Figure 4** is a picture of my monster. It isn't pretty, but it works great!

Well, that's it for another month. Keep building robots and if you get stumped, drop me an email. As usual, if you



have any questions for Mr. Roboto, feel free to email me at roboto@servomagazine.com and I'll be happy to try to answer it! **SV**

STEER WINNING ROBOTS WITHOUT SERVOS!



Perform proportional speed, direction, and steering with only two Radio/Control channels for vehicles using two separate brush-type electric motors mounted right and left with our **mixing RDRF dual speed control**. Used in many successful competitive robots. Single joystick operation: up goes straight ahead, down is reverse. Pure right or left twirls vehicle as motors turn opposite directions. In between stick positions completely proportional. Plugs in like a servo to your Futaba, JR, Hitec, or similar radio. Compatible with gyro steering stabilization. Various volt and amp sizes available. The RDRF47E 55V 75A per motor unit pictured above.
www.vantec.com

VANTEC Order at
(888) 929-5055

AndyMark
Inspiring Mobility

Specializing in Unique Wheels,
Gearboxes, Aluminum Sprockets
and Drive Bases



8" Plastic Omni Wheel



c-Rio-Ready Full
Chassis Kit



Aluminum Sprockets



6" HD Mecanum Wheel Set

AndyMark, Inc.
sales@andymark.com

Toll Free: 877-868-4770

www.andymark.com

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

OCTOBER

- 1-4** **UAV Outback Challenge**
Kingaroy, Australia
Search and Rescue Challenge, Airborne Delivery Challenge, and Autonomous.
www.uavoutbackchallenge.com.au
- 5-7** **MindSpark**
Pune, India
Events include Micromouse, robot dog fights, and robot search and destroy.
www.mind-spark.org
- 6** **The Franklin Cup**
Philadelphia, PA
Remote control vehicle combat.
www.nerc.us
- 18-21** **Latin American Robotics Competition**
Fortaleza, Brazil
Events include the Brazilian Robotics Competition, Robocup Latin American Open, and Brazilian Robotics Fair.
www.crobotica.org
- 19-21** **Critter Crunch**
Hyatt Regency Tech Center, Denver, CO
The original remote control vehicle combat event.
www.milehicon.org

NOVEMBER

- 3** **Atlanta Hobby Robot Club (AHRC) Robot Rally**
Pinckneyville Community Center, Norcross, GA
Events include line maze solving, mini Sumo, and the Robot Polyathlon. The polyathlon is made up of six individual contests: Advanced Line Follower, Beacon Killer, Beacon Killer with Obstacles, Navigation by Dead Reckoning, and Bulldozer.
www.botlanta.org/robot-rally
- 23-25** **All Japan Micromouse Contest**
Toyosu, Koto-ku, Japan
Events for autonomous robots including classic Micromouse, half-size Micromouse, and robot race.
www.ntf.or.jp/mouse
- 23-25** **Robotex**
Tallinn, Estonia
This is the largest autonomous robot competition in Estonia. This year's events include robot football, line following, mini Sumo, and LEGO Sumo.
www.robotex.ee
- 25** **Robocon**
Tokyo, Japan
Student teams from all over Japan come together at Robocon, where the robots they've designed compete in the Robo Bowl.
www.official-robocon.com

DECEMBER

- 1-2** **South's BEST Competition**
Auburn University, Auburn, AL
Regional for the BEST student competition.
www.southsbest.org/site
- 15** **Robotic Arena**
Wroclaw, Poland
Lots of events including mini Sumo, micro Sumo, nano Sumo, Micromouse, line following, and freestyle.
www.roboticarena.org

Robotics Showcase

SERVO Connecting the world of personal robotics
MAGAZINE

Online Store

SERVO'S Online bookstore has more than 30 titles on robotics!

Robot Builder's Bonanza
Fourth Edition

123 ROBOTICS EXPERIMENTS FOR THE EVIL GENIUS

MECHANISMS MECHANICAL DEVICES SOURCEBOOK

Making Things Move
200 Projects for Kids

Just to name a few!

(800) 873-4624

Recycling & Remarketing High Technology
WEIRDSTUFF®
WAREHOUSE

Software, Computers, Electronics, Equipment, Doo-hickies

**384 W. Caribbean Dr.
Sunnyvale, CA 94089**
Mon-Sat: 9:30-6:00 Sun: 11:00-5:00
(408) 743-5650 Store x324

WE BUY AND SELL EXCESS & OBSOLETE INVENTORIES!

FREE COMPUTER RECYCLING
We recycle computers, monitors, and electronic equipment. **M-Sat 9:30-4:00**

GREAT DEALS!
Hi-tech items, electronics test equipment, and more!

GIANT AS-IS SECTION
10,000 sq. ft. of computers, electronics, software, doo-hickies, cables, and more!

also check out our...

eBay Store
stores.ebay.com/WeirdStuff-Inc

WWW.WEIRDSTUFF.COM

ALL ELECTRONICS
CORPORATION

THOUSANDS OF ELECTRONIC PARTS AND SUPPLIES

VISIT OUR ONLINE STORE AT www.allelectronics.com

WALL TRANSFORMERS, ALARMS, FUSES, CABLE TIES, RELAYS, OPTO ELECTRONICS, KNOBS, VIDEO ACCESSORIES, SIRENS, SOLDER ACCESSORIES, MOTORS, DIODES, HEAT SINKS, CAPACITORS, CHOKES, TOOLS, FASTENERS, TERMINAL STRIPS, CRIMP CONNECTORS, L.E.D.S., DISPLAYS, FANS, BREAD-BOARDS, RESISTORS, SOLAR CELLS, BUZZERS, BATTERIES, MAGNETS, CAMERAS, DC-DC CONVERTERS, HEADPHONES, LAMPS, PANEL METERS, SWITCHES, SPEAKERS, PELTIER DEVICES, and much more....

**ORDER TOLL FREE
1-800-826-5432**
Ask for our **FREE 96 page catalog**



**Brandon built his own
Mars Rover
out of LEGO bricks.**

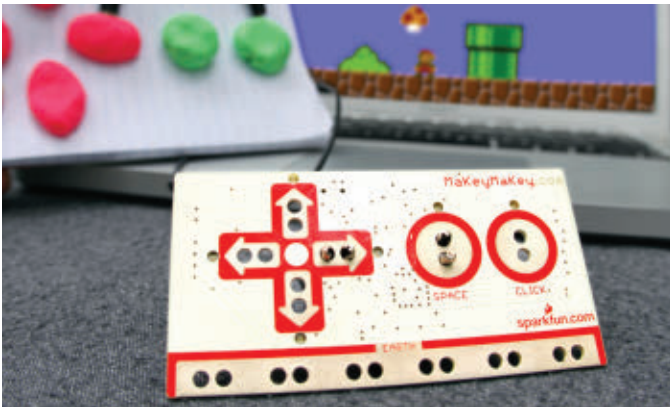
**His mom gave him
a subscription to *SERVO*
so he could take
the next step.**

Get access to the digital
edition for free with your
print subscription!

12 issues for \$24.95
Call 1-877-525-2539
or go online at
www.servomagazine.com

NEW PRODUCTS

MaKey MaKey Invention Kits



SparkFun Electronics is now shipping the MaKey MaKey — a new product that invites everyone to be an inventor.

MaKey MaKey was designed by MIT graduate students Jay Silver and Jay Rosenbaum, and is an invention kit that encourages builders to discover creative new ways to interact with computers. It allows users to create inventions using almost any everyday object such as a keyboard key or mouse. For example, you could replace your space key with a banana, use Play-Doh™ to move and click your mouse, or high-five your best friend to advance PowerPoint slides.

"The MaKey MaKey is an amazing kit because it really lowers the point of entry for beginning electronics enthusiasts," said AnnDrea Boe, SparkFun Director of Marketing Communications. "This kit allows users to start inventing immediately, right out of the box. It's a fantastic way to start exploring DIY electronics."

The MaKey MaKey uses very high resistance switching to detect when any of its 18 inputs are activated, even through materials that aren't very conductive (like leaves, pasta, or people). The MaKey MaKey communicates with your computer using the USB Human Interface Device (HID) specification which means that it can act just like a USB keyboard or mouse. In addition, the MaKey MaKey is also an Arduino-compatible platform, so users can easily re-map any of the keys, or further customize it to their needs.

The regular kit is \$39.95 and comes with the MaKey MaKey HID board, an alligator clip pack, and a mini-USB cable. The deluxe kit (\$49.95) includes everything that the basic kit does, in addition to a second alligator clip pack, a jumper wires pack, and a roll of copper tape for even more possibilities.

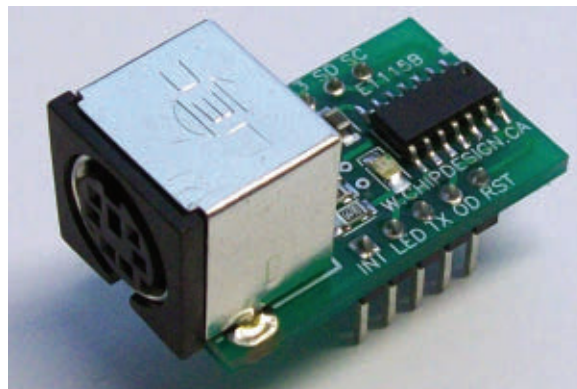
For further information, please contact:

SparkFun Electronics

Website: www.sparkfun.com

PS/2 Keyboard to ASCII Converter Module

CHiPdesign announces the availability of the E1115 — a compact PS/2 keyboard to ASCII converter module for embedded microcontroller applications. It offloads the process-intensive PS/2 scancode decoding, while providing a fast single byte output response for both ASCII and non-ASCII keys including multi-media. Requiring only +5V, the easy to use module provides a pin selectable 57.6K/115.2K baud rate, as well as a 100 kHz clocked serial data output. The module also flashes an LED and generates an interrupt pulse.



The converter allows for the quick development of any keyboard-based project. The E1115B is available as a 10-pin DIP style 1.3" x 0.7" module with integrated PS/2 connector, and also as a SOIC-14 chip. An application note on the website describes how to parse the output into commands and data. Pricing starts at \$16 for the module and \$8 for the chip.

For further information, please contact:

CHiPdesign

Website: www.CHiPdesign.ca

Text To Speech IC

The SP0-512 RoboVoice text to speech IC from Speech Chips is a pre-programmed microcontroller that

accepts English text from a serial connection, converts that text to phoneme codes, then generates audio. It is ideal for adding a robot voice to embedded designs.

Features include:



- Single chip text to speech IC.
- Low power.
- Communicates using a simple serial port (9600 N81).
- 800 rules that convert English text into phoneme codes.
- Built-in 16-bit at 16 kHz DAC. No external filter required.

The RoboVoice is based on the Microchip dsPIC33FJ64GP802 microcontroller. A datasheet is available at www1.microchip.com/downloads/en/DeviceDoc/70292F.pdf. User's should refer to this document for greater detail about the chip itself.

Source code licenses are available. The complete text to speech system fits in about 32K and is written in ANSI C, so it can be ported to almost any embedded processor. Price is \$24.99 each.

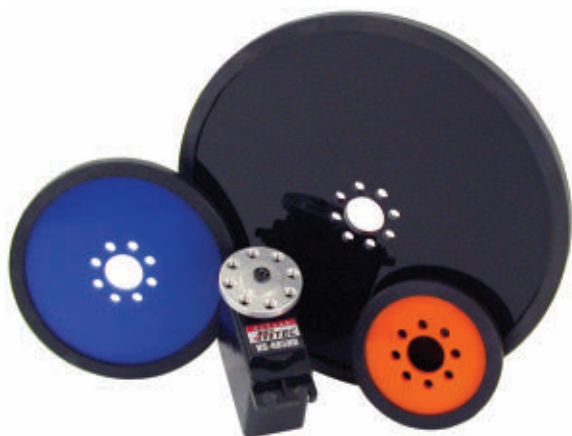
For further information, please contact:

Speech Chips

Website: www.speechchips.com

Servo Hub and Precision Disk Wheels

ServoCity introduces their new .770" aluminum servo hubs. These new hubs are machined from 7075 T6 aluminum and have eight 6-32 tapped orbital holes that allow users to attach any of ServoCity's components that have the .770" pattern. They mate up perfectly with ServoCity's line of gears, sprockets, pulleys, and wheels which utilize the same .770" hub pattern. The aluminum servo hubs have broached splines to allow for a solid connection to any standard size Hitec or Futaba spline. The 1/2" diameter protrusion in the center of the hubs ensures that the attachment is perfectly centered.



In order to complement the new aluminum servo hubs, ServoCity has updated its entire line of precision disk wheels to incorporate the new .770" hub pattern. The precision disk wheels are offered in a wide variety of colors and diameters. The wheels provide excellent traction due to the rubber ring which surrounds the wheel. Each wheel is capable of holding up to 15 lbs without flexing.



0.770" Gears and Sprockets

ServoCity has also updated their line of plastic gears and sprockets. The gears and sprockets now utilize the .770" hub pattern with 6-32 holes and a 1/2" center bore which is used throughout their entire line of products. The gears are constructed of acetyl and are offered in 1/8", as well as 3/16" face width, 32 and 48 pitch, and a vast range of sizes in order to fit almost any application. Like the hub gears, the new hub sprockets are constructed of acetyl for superior strength and durability. The sprockets are .100" thick and accept standard .250" (1/4") chain.

For further information, please contact:

ServoCity

Website: www.servocity.com

Wi-Fi Gateway Module Connects Devices to the Internet

WizFi630 is a high performance embedded Wi-Fi gateway module available from Saelig Co. that transforms RS-232 and TCP/IP protocols into the IEEE802.11 b/g/n wireless LAN protocol. WizFi630 enables a device with an RS-232 serial interface to connect to LAN or WLAN for remote control, measurement, and administration. WizFi630 can also operate as an IP router due to its internally embedded switch.

WizFi630 uses serial (UART), LAN, and Wi-Fi (WLAN) interfaces to perform functions such as serial-to-Wi-Fi, serial-to-Ethernet, and Ethernet-to-Wi-Fi. Users can connect to WizFi630's internal web server and use simple serial commands to change Wi-Fi settings. In addition to serial devices, 8/16/32-bit microcontrollers can use WizFi630's





UART interface for Wi-Fi functions. WizFi630 can operate in Gateway, AP, AP-Client, Client, and AD-HOC modes, and offers 2x UART and 3x Ethernet interfaces.

WizFi630 can simplify wireless module tasks like design, testing, and certification, and offers a fast solution for users who lack wireless network experience. WizFi630 follows the 802.11b/g/n standard, and supports wireless interface transfer speeds of up to 150 Mbps. Configuration as a built-in web server is easy via serial commands from the Windows utility software provided. WizFi630 is CE and FCC certified.

A WizFi630 evaluation board with ready-made interface connectors is available providing a complete test set-up with PC software and documentation, enabling anyone to quickly develop a wireless solution.

Applications for WizFi630 are wide ranging. For a 4G LTE repeater, WizFi630 can function as a serial-to-Wi-Fi access point; for routine management and upgrade of the repeater, WizFi630 provides serial-to-Wi-Fi gateway function, but can also enable a laptop or smartphone to access the repeater without using an additional access point. As a hotel room controller, WizFi630 can be used as an Ethernet-to-Wi-Fi AP-Client; in hotel room control, WizFi630 provides a multi-networking environment by combining Wi-Fi and Ethernet connections. Users can control lighting, TVs, or any electric devices in the room via Wi-Fi, and have Internet connection, as well.

WizFi630 Wi-Fi modules are available now starting at \$35, with evaluation boards priced at \$98.

For further information, please contact:

Saelig

Website: www.saelig.com

Grippers, Foot Weights, and Crawler Kits

RoboBrothers, Inc., has three new products available for robot builders.

The Philo Gripper is an easy drop-in replacement for a forearm, with two servos for grip and wrist turning. It also includes a torque limiter to protect the gripping servo and a silicon tip for better gripping force. Pre-assembled versions and kits are available.



Also new are their RoboPhilo foot weights. The kit includes two sets of weights and double-sided tape. This product adds weight to the foot areas, so provides more stable motion.



The RoboCrawler is a simplified robotic crawler which offers an affordable option to hobbyists interested in multi-legged walkers.



Features include:

- Eight servos for eight degrees of freedom for the legs.
- Each motion routine can have up to 30 sequences, and each sequence can have up to 15 poses.
- Sequence and pose can be reused for other motions to save Flash memories.
- One motion routine can have up to 450 pose transitions.
- RS-232 serial connection to PC for motion programming and execution.
- IR handheld remote to execute user-created program motions.
- Powered by five AA batteries.
- Spare ports available for installation of sensors for autonomous operation.

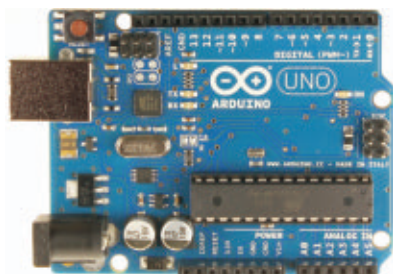
For further information, please contact:

RoboBrothers, Inc.

Website: www.robobrothers.com

Motor Controllers • Arduino • Arduino Kits • Arduino Shields • Robot Platforms • Parts

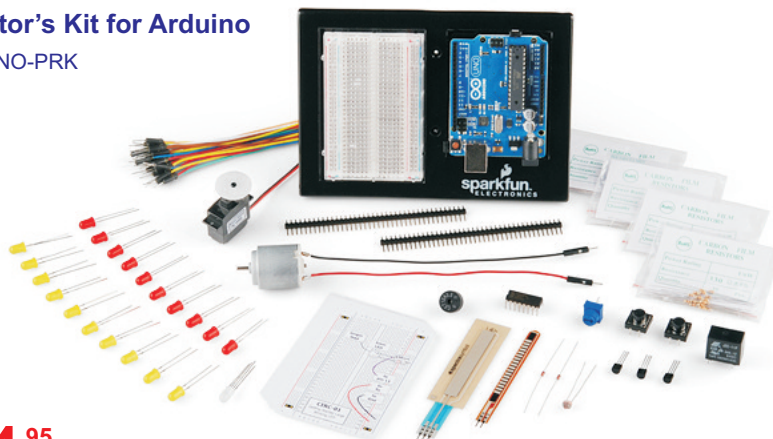
Arduino Uno



\$29^{.95}

ARDUINO-UNO
Latest "R3" Version!

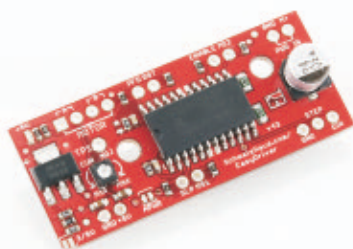
Inventor's Kit for Arduino
ARDUINO-PRK



\$94^{.95}

Includes 36-Page Printed Full-Color Manual

EasyDriver Stepper Motor Driver



SX09402

\$14^{.95}

Stepper Motor



SX09238

\$19^{.95}

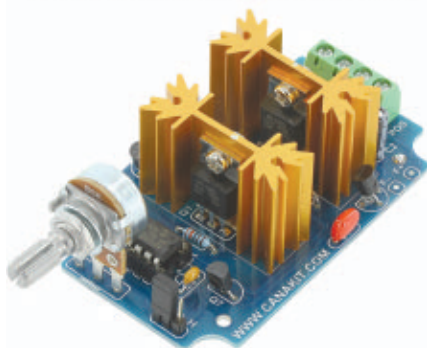
50A Motor Speed Controller (PWM)
UK1133



Optional LCD

\$59^{.95}

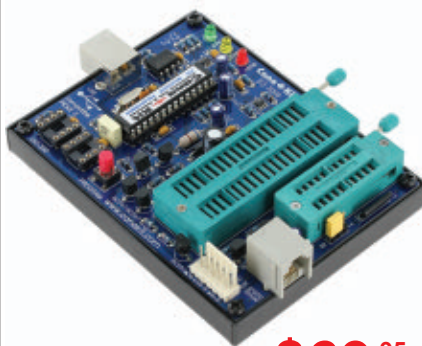
**7A Bi-Directional
Motor Speed Controller (PWM)**



UK1125

\$39^{.95}

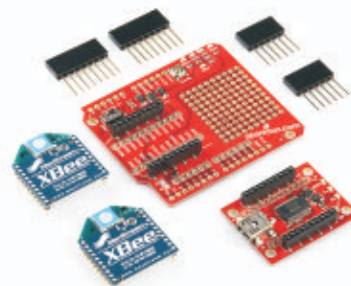
**USB PIC Programmer
(MPLAB Compatible)**



UK1300

\$69^{.95}

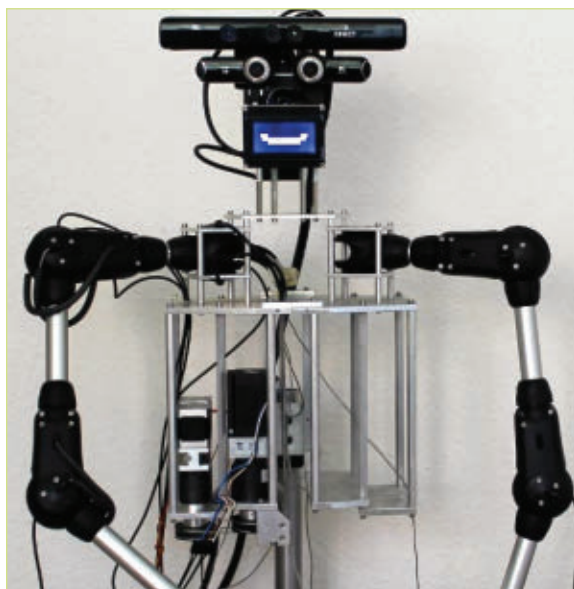
Xbee Wireless Kit



XBEE-KIT

\$94^{.95}

bots IN BRIEF

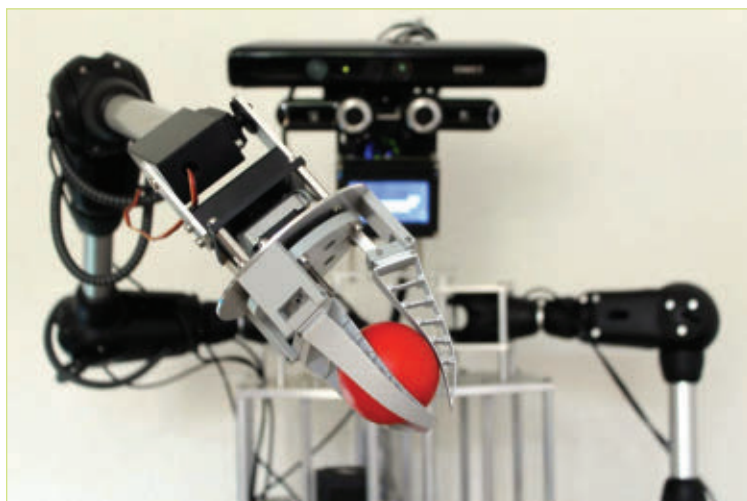


KINECT WITH ADAM

Malte Ahlers — a student of neurobiology in Germany — has built a humanoid robot torso called Adam (Advanced Dual Arm Manipulator), or AI for short. Although the project is still somewhat early in development, the hardware side of things has been in the works for around two years.

Adam's arms have five degrees of freedom actuated by robolink joints from Icus — a German robotics company. The robolink joints have both pivot and rotation — ideal for building robot arms — but they use external cables for their rotation (much like a pulley; one for clockwise and another for counterclockwise rotation). The resulting bundle of cables had to be routed to planetary gear motors inside the torso, and Ahlers had to build a motor controller to read out the position encoders in the joints in order to drive the motors with position control.

For the hands, he went with FESTO's flexible FinGrippers which use a simple, yet surprisingly effective design. FESTO is also a German company, famous for its biologically-inspired robots and the FinGrippers are no exception. Using just one motor, three fingers pinch together in unison and will naturally conform to roundish objects of varying size. Finally, Adam's head (sitting on a two DOF neck) has a Microsoft Kinect sensor and two Logitech QuickCam Pro 9000 cameras. A speaker provides for speech synthesis and an LCD panel features animated "lips." For now, Adam remains an impressive work-in-progress.



NOW THEY'RE (NOODLE) COOKIN'

Little by little, robotic chefs are taking over China. One of the latest is Chef Cui who specializes in noodle cutting. Designed by Cui Runquan, this chefbot is being mass-produced with a reasonable price tag of \$1,500. We call that reasonable since hiring a chef could cost \$4,700 yearly. Runquan says that the upcoming human generation isn't really into a career as a cook, so with about 3,000 already sold he expects these robotic counterparts will pick up the slack.

The Chefbots can "hand" slice noodles into a pot of boiling water. According to one human, "The robot chef can slice noodles better than human chefs."



bots IN BRIEF

WEEBOT FOR WEE ONES

This is a WeeBot, and it's one of the very rare times it's okay to combine robots with babies. That's because a WeeBot is basically a way of turning a real baby into an unstoppable fusion of biology and engineering.

Babies (as you may have noticed if you own one) like to get into all sorts of mischief, and studies show that exploring and interacting with the world is important for cognitive development. Babies who can't move around as well may not develop at the same rate as babies who can, which is why researchers from Ithaca College in New York are working on a way to fuse babies with robots to give mobility to all babies — even those with conditions that may delay independent mobility, like Down syndrome, spina bifida, or cerebral palsy.

WeeBots are built with Adept MobileRobots Pioneer P3-DX bases. On top of the bases are Nintendo Wii balance boards which are rectangular platforms with load sensors at the corners. A commercial infant seat is placed on top of the balance board, and the robot can then be calibrated to move in whichever direction the baby leans.

To test the WeeBots, researchers borrowed five infants, ranging in age from six months to nine months. Each infant was given five training sessions on the robot using a toy as bait, and by the end of those sessions, the babies were reliably able to control the WeeBot in goal-directed movement during periods of free play. You might think that a six month old baby wouldn't necessarily have the facility to control a robot like this, but they catch on surprisingly quickly. All of the babies in the study were developing typically for their age; none of them had the ability to crawl, so the robots were their only means of transportation.

This was just a pilot study to make sure that the WeeBot worked, but recently published continuing research has also tried using WeeBots with infants with mobility disorders. It's turning out to be difficult for some babies to sit up enough to control the WeeBot by leaning, but in at least one case, a 15 month old boy with cerebral palsy was able to learn to control a WeeBot — after which he started to develop crawling skills on his own.

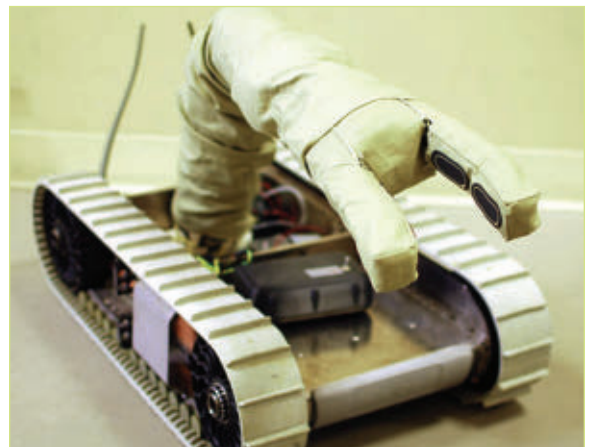


AIRarm YOURSELF

Here's a fully controllable robot arm that can be inflated and deflated like a balloon.

The AIRarm is lightweight, inexpensive, and stows compactly. It's inflated and deflated with an onboard pump, and uses actuators and strings to move its joints without embedded motors. While regular PackBot three-link arms are between 15 and 20 pounds, the AIRarm system only weighs about a tenth of that — a fact that would be appreciated by the soldiers that have to carry these robots around.

Despite its light weight, AIRarm is no slouch and can lift up to five pounds or possibly more, depending on how much it's inflated. By varying the level of inflation, it's also possible to vary the level of compliance of the arm; this makes the arm a little bit flexible when you need it to be which, in turn, makes it safer and more durable. Since it's mostly made of fabric and string, it's wicked cheap, at least compared to a conventional arm.

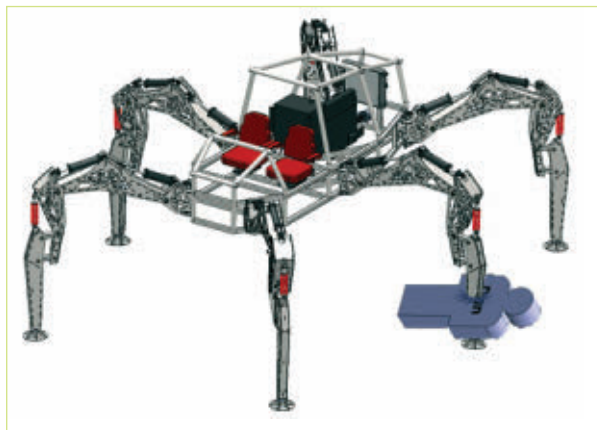


GET STOMPY

This is Stompy — a giant hexapod that you can ride in. (How awesome is that!?)

Stompy is the brainrobot of Artisan's Asylum, a hackerspace out in Boston. This is actually a serious undertaking. The guys behind it are experienced roboticists from places like Boston Dynamics, Barrett, and DEKA. The robot will be powered by a 135 horsepower engine driving a whole bunch of hydraulics, and while it's largely designed to stomp around in an exhibitory manner, the team has big plans for it.

The robot isn't just being built for fun. It has practical purposes, as well. With six force-sensitive legs and a ground clearance of six feet, the robot will be able to walk over broken terrain that varies from mountainous areas, to rubble piles, to water up to seven or eight



feet deep —

everywhere existing ground vehicles can't go. Not only that, while navigating such terrain Stompy could carry 1,000 pounds at 2-3 mph, and up to 4,000 pounds at 1 mph. This is important in disaster areas because Stompy (and the technology it represents) could easily reach people who can't be reached by other means.

So how big is big? Check out the photo.

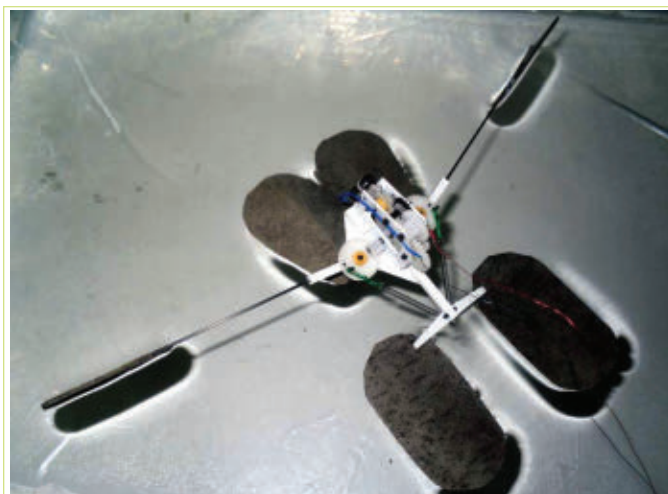
Yep. That's a person! A really tall person named Matt.



LEAPS OF FAITH

Roboticists at the Harbin Institute of Technology in China have managed to make a robotic insect that — in addition to walking on water — can also jump. Modeled after a water strider, the legs of this robot are made of a porous, water-repellent nickel foam. The concept is that if you spread the weight of the robot out enough, the surface tension of the water can support it. This is a tall order for a robot this large. Weighing in at 11 grams, this porker is over a thousand times the mass of its biological inspiration.

To get the robot to jump, a separate set of legs was added, bringing the total to five. By using these actuating legs to push against the surface of the water, the robot was able to make leaps 14 centimeters high and 35 centimeters long, taking off at nearly 65 kph which is impressive for such a little guy. Robots like these could skim across lakes and other bodies of water to monitor water quality or act as tiny spies.



Cool tidbits herein provided by www.botjunkie.com, www.robotsnob.com, www.plasticpals.com, <http://www.robots-dreams.com/>, and other places.



Twins Geoffrey Howe (left) and Michael Howe flank their new invention — a fire-fighting robot known as the Thermite.

THERMITE TAKES THE HEAT

Brothers Geoffrey and Michael Howe (who build robots for the military) decided there was a need for a bot that fights fires. The Thermite weighs 1,400 pounds, stands about four feet tall, and moves on a track (for now). Running on diesel, it has video and infrared cameras so it can be remotely operated.

Geoffrey calls it the "Swiss Army knife of robotic responses" since it can operate with various attachments like a hydraulic arm for saving humans. After spending a couple of years testing the firebot, the twins have already sold a few of them.

The Thermite has been described as a rugged powerhouse of a vehicle. At 34 inches wide, Michael says it's small enough to get into a burning room, yet powerful and well-equipped enough to extinguish fires when it gets there.

For example, a hydraulic arm can be bolted on with the strength to pull a human out of a burning building, among other things. "It can move a 55 gallon drum full of chemicals away from a fire," Geoffrey comments. "It can do all sorts of things that you may need done in an emergency situation including turning valves, cutting wires, or cutting rope."

The Thermite has been tested in a number of situations, including a scenario known as a bleve, "which is a boiling liquid expanding vapor explosion," Michael says. "This happens when a tanker truck rolls over, and you can see on the news a big fire shooting out of a tanker, and eventually it explodes. A lot of firefighters lose their lives trying to get in there and extinguish that. This can go in there. Also fuel farms — where they keep all the fuel — they catch on fire, and it's just too hot to get in there. Send a robot in. I believe this technology (in five to 10 years) will be standard operating procedure for every fire department, to have some sort of robotic solution."



NICE PRICED MITT

As part of DARPA's ARM program, Sandia has partnered with Stanford University to create a dexterous robot hand on the cheap.

As Sandia puts it, "The Sandia Hand addresses challenges that have prevented widespread adoption of other robotic hands such as cost, durability, dexterity, and modularity." You can attach tools like screwdrivers or flashlights (or laser cannons), and the modular design also makes the hand durable, since the fingers will just fall off if something smacks into them. As principal investigator Curt Salisbury explains, "If a finger pops off, the robot can actually pick it up with the remaining fingers, move it into position, and resocket the finger by itself." Also, the "skin" of the hand is designed to mimic the flexibility of human tissue, providing some shock absorption and allowing the hand to more firmly grasp objects.

This is all really cool stuff, but the cost is where the hand really comes through. In low volume production, the Sandia Hand should only cost about \$10,000 total. (Fingers included.) For the record, Sandia's press information says that's about 90% less than other commercially available robot hands with similar independently actuated degrees of freedom.

The operator controls the robot with a glove, and the lifelike design allows even first-time users to manipulate the robot easily.

Using Sandia's robotic hand to disable IEDs (improvised explosive devices) might help lead investigators to the bomb makers themselves. Often, bombs are disarmed simply by blowing them up. While effective, that destroys evidence and presents a challenge to investigators trying to catch the bomb maker. A robotic hand that can handle the delicate disarming operation while preserving the evidence could lead to more arrests and fewer bombs.

DOUBLE TROUBLE

Doing robotic telepresence can be tricky. It's one of those things that sounds good on paper, but when it comes to getting people to plunk down a pile of cash for the hardware, going commercial with the concept has proven to be more or less impossible outside of some very specific circumstances. Double Robotics — a startup out of Y Combinator — is taking a shot at the telepresence space with a slick new iPad-based platform called Double.

When we say it's "iPad-based," it's literal. Double is pretty much a mobile base for an iPad. You can log into the iPad from any computer or iOS device, and drive the robot around while streaming two-way audio and video. Double lets you talk to people, go sightseeing, or do whatever else you want while remotely inhabiting the body of a robot. The iPad can be extended vertically from three and a half to five feet to maintain eye level with people sitting or standing, and the Segway-style base uses high efficiency motors to zip around for up to eight hours on a charge. It has a futuristic, minimalist design, and it has a low preorder price of \$2,000 (iPad sold separately).

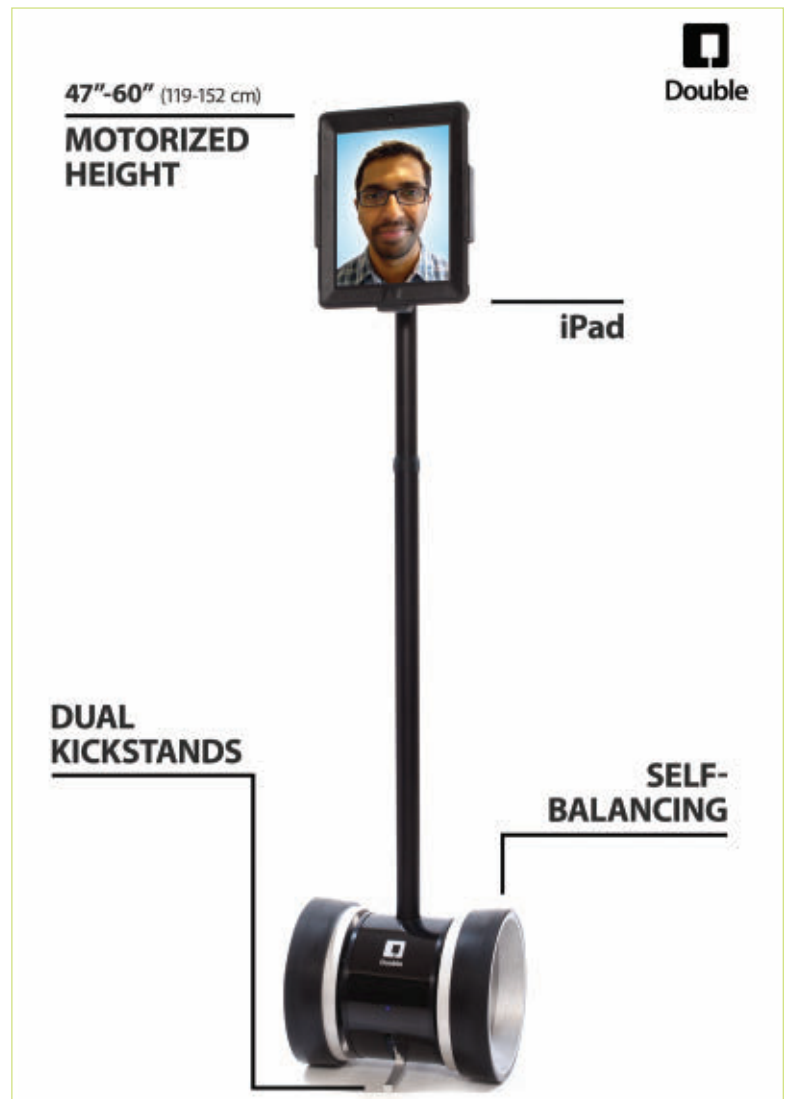


Double arrives fully functional as soon as you open the box — just insert your iPad. Touch the power switch to activate Double's self-balancing sensors (keeping itself upright). At only 15 pounds (7 kg), it's easy to move by hand and won't damage furniture.

Setting up Double on your iPad is as easy as downloading an app. The same app is used for the driver's iPad and the robot's iPad. Once you create an account, your driver's iPad will display all of your Doubles anywhere in the world from one screen.

Connect to Double by tapping its icon, which will initiate an interactive video call. During the call, touch the screen anywhere to engage the driving controls. Slide your thumb to drive and turn. Additional controls adjust your height remotely, park yourself with the kickstands, and more.

You can drive Double from an iPad, iPhone, iPod touch, or desktop web browser.



Build robots that
stand the test of time.



Solutions³

Robot motor controllers starting at \$20

Solutions Cubed... hardware made easy



www.solutions-cubed.com
designservices@solutions-cubed.com
530.891.8045

Use coupon code "SERVOMAG"
for additional 5% discount
(valid through Jan. 1st 2013)

COMBAT ZONE

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

Featured This Month:

32 BUILD REPORT:
*Testing the Prototype:
Klazo — My 1 lb Drumbot
From Kitbots.com*
by Michael Jeffries

34 *Upcoming Events*

35 *Clash of the Bots 3*
by Andrea Suarez

38 *The History of Robot
Combat: Robot Battles
at Dragon*Con*
by Morgan Berry

40 CARTOON

[www.servomagazine.com/
index.php?/magazine/article/
october2012_CombatZone](http://www.servomagazine.com/index.php?/magazine/article/october2012_CombatZone)

BUILD REPORT:

Testing the Prototype: Klazo — My 1 lb Drumbot From Kitbots.com

● by Michael Jeffries

I recently built and tested one of the prototype 1 lb drumbot kits from **Kitbots.com** as part of the development process, prior to the kit moving to production. At the beginning of the build, I received the fully assembled chassis along with four kit gearmotors that

formed the basis of this build. The chassis is a combination of UHMW plastic, aluminum, and the Kitbots product Nutstrip — much like the Weta and Trilobite kits. The design of the chassis closely resembles Weta, however, there is an important difference in

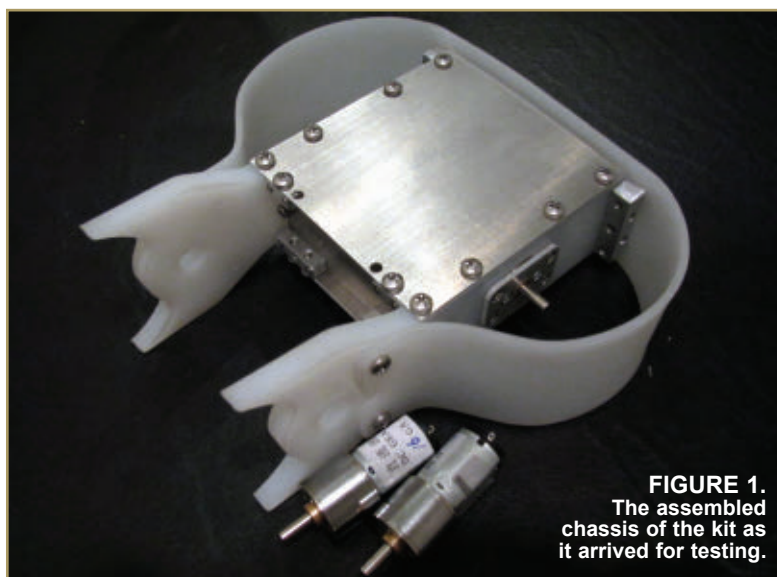


FIGURE 1.
The assembled
chassis of the kit as
it arrived for testing.

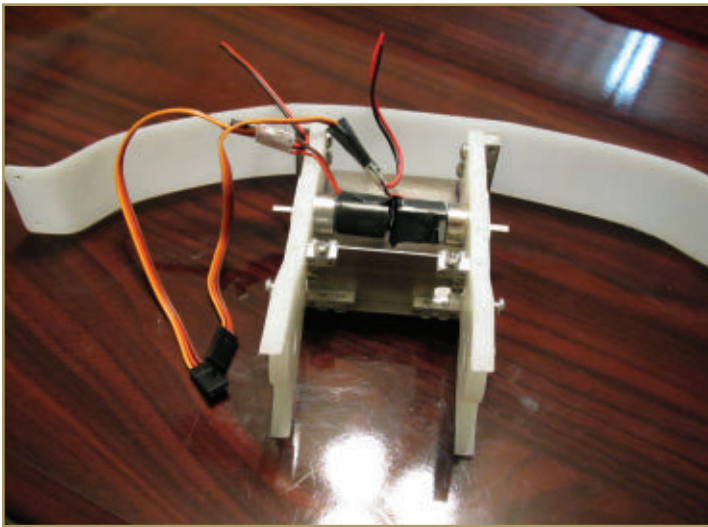


FIGURE 2. View of the internal layout of the chassis showing drive motors and TinyESCs.

the weapon system. Unlike with Weta, the brushless motor that powers the weapon is contained within the drum.

The first priority of the build was to get the electrical system working to allow for testing of the prototype gearboxes. I started with the drive portion of the electrical system

which meant the installation of FingerTech Robotics TinyESCs, a Rhino 460 7.4V Li-Po battery, Hobbyking R410 OrangeRX, and a small power switch. For this portion of the build, the power switch was left floating.

Once the initial portion of the wiring was completed, I spent some

time testing the drive system. After stalling one of the gearboxes, I noticed that the side of the drive system was not functioning at full capability. I dismantled the gearbox and found that some of the gtearteeth were damaged. As part of this process, I noticed that the gearboxes were visually identical to



FIGURE 3. Partially manufactured drum showing motor positioning.

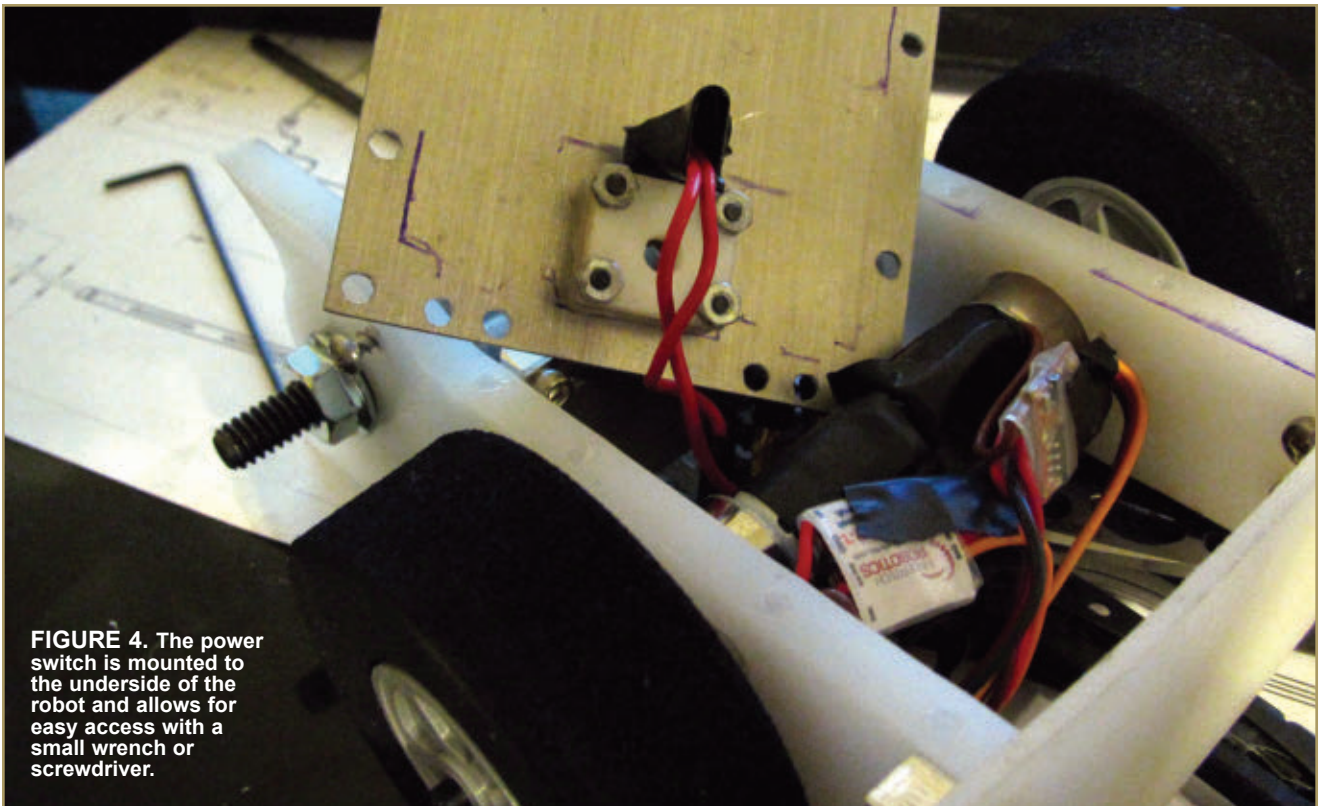


FIGURE 4. The power switch is mounted to the underside of the robot and allows for easy access with a small wrench or screwdriver.



FIGURE 5. Titanium wedgelets on the completed robot.



FIGURE 6. Photo of Klazo after Clash of the Bots 3 with the new heavier drum.

the FingerTech 22:1 Silver Spark gearboxes, though they were of lower quality. I swapped out the prototype gearboxes for the Silver Spark versions and have yet to have an issue with them.

I then moved on to the creation of the drum for Klazo. The drum is made from a 1" ID, 1.5" OD 6061 aluminum tube. I used my small lathe to bore out one side

of the tube to fit the weapon motor, then turned down the surface of the tube to reduce some weight and minimize the imbalance of the drum that would come from the lack of precision my lathe allows. After turning the drum to size, I drilled holes through both walls of the drum that were sized for a 10-24 tap. The holes were quickly tapped and the screw-based

teeth were attached to the drum. Once this was completed, the motor was glued into the drum with Goop (a silicone based adhesive). In preparation for the initial test spin, I mounted the power switch to the bottom armor plate to allow operation of the robot without exposed electronics.

In testing, the drum wasn't perfectly balanced but was close enough that the robot could drive reasonably well with the weapon at full speed. The final step before its debut at Clash of the Bots 3 was the addition of two 1/16" titanium "wedgelets" that would

increase the chances of it getting under an opponent prior to impact.

Klazo went 2-2 at Clash of the Bots 3, and survived a great deal of abuse across all of the matches. During the event, Klazo faced Saifu — the prototype kit built by Pete Smith. The match lasted a full two minutes and went in favor of Klazo. The deciding factor in the match appeared to be a gearbox failure on Saifu. The relevant differences between the two robots were the change in gearboxes to the FingerTech Silver Sparks and the use of lite flite wheels on Klazo instead of the hard plastic wheels that Saifu used.

After the event, I decided to make a more balanced and heavier drum since Klazo was well below the weight limit. The new drum has eight 10-24 flat head screws for teeth instead of the four used on the initial drum and has a slightly larger diameter. Outside of that modification, Klazo is ready for the next event (which was the Robot Micro Battles in Atlanta, GA during Dragon*Con).

Overall, I am happy with the durability of the kit and relative ease of construction. The most difficult portion of the build was making the drum. However, I suspect there will be at least one stock option available upon the release of the kit. **SV**

EVENTS

Upcoming Events

NERC Franklin Institute

2012 will be presented by the North East Robotics Club in Philadelphia, PA on October 6th.

www.nerc.us



Mecha-Mayhem 2012 will be presented by the Chicago Robotic Combat Association in

Cleveland, OH October 13th and 14th. www.theCRCA.org **SV**



Clash of the Bots 3

● by Andrea Suarez

Motorama Carolina Combat Robotics hosted its third Clash of the Bots on July 14 at the Schiele Museum of Natural History. Builders traveled to Gastonia, NC from across the country with almost 40 robots in three combat weight classes (Flea, Ant, and Beetle). As we approached the venue, the first thing that caught my eye was the "Clash of the Bots TODAY" on the museum's LED sign near the main road.

We were the first competitors to arrive at the event, and the venue had a sort of calm-before-the-storm feeling. All the pit tables were perfectly organized with our competitor packets nicely laid out. The arena sat in the dark, ready for action. Chuck Butler recalls: "Four years ago, I received a message from Christie at the Schiele Museum in Gastonia looking for someone that could do a robot presentation for a group of 100 Girl Scouts. We took on the task, and the girls were surprisingly very interested in the

destruction, especially when it involved a Barbie doll vs. a combat robot. We had been looking for a suitable venue and when the museum decided they would like to try a robot day it was a perfect fit, and Clash of the Bots was born with an average of over 500 visitors at each event."

As visitors entered the Schiele Museum, you could instantly hear the sounds of battle coming from the competition area, attracting crowds of spectators to the arena.

Ram Robotics drove from Florida to North Carolina to compete their fleet of robots across all the weight classes, including two new robots: iKid (Flea), and Calamity Kid (Ant). David Liaw flew from California for the event, but his robots were lost at the airport! The event started with a builder's meeting where Chuck welcomed competitors and explained the rules for the day.

Even without arena hazards, every two minute match provided a

show of sparks and flying robots. It was all captured in high speed at 300 frames per second by Robert Woodhead and provided to the competitors free of charge. Robert also provided free DVDs of past events to the competitors and spectators.

FLEAWEIGHT CLASS: Seven robots made up the Fleaweight bracket. Caterpillar, last year's winner, returned with its massive servo-powered lifting plow that was sliced up by Paul Grata's Pissed Off Unicorn as both robots went spinning across the arena in a first round match-up. Busted Nuts Robotics brought five of the Fleaweight bots, which provided for plenty of competition among the teammates.

Dirty Sanchez's two vertical discs met seemingly weightless one-wheeled melty-brain Berzerker for some big air time almost every hit. Dirty Sanchez even did four and a half flips through the air on a single



FIGURE 1. Schiele Museum.



FIGURE 2. Sam McAmis does some last minute weight adjustments to his new Fleaweight bot, iKid.



FIGURE 3. The Clash of the Bots staff. (Photo by Yong Ye.)



FIGURE 4. Algos vs. Saifu.



FIGURE 5. Jen Villa, Duke Robotics President, watches as teammate Caleb competes his first bot, Shaka Zulu. (Photo by Yong Ye.)



FIGURE 6. Caterpillar vs. Pissed Off Unicorn.

blow. Duke Robotics' Trash used its vertical oversized beater to take the horizontal disc off Invertigo in the first round. In the semi-finals, Pissed Off Unicorn easily took out both of Invertigo's wheels, advancing to the finals to fight an undefeated Thrash. Pissed Off Unicorn used its eight inch spinning weapon to defeat Thrash twice for the Fleaweight title.

ANTWEIGHT CLASS: Thirteen robots battled it out in the Antweight division. Early in the bracket, Poco Tambor sent Klazo high in the air hit after hit, almost getting Klazo stuck between the arena barrier and the wall as it bounced off the arena. Caleb Boothe from Duke Robotics competed with his first robot, a nicely machined full-body spinner Shaka Zulu, and received a tough first match-up against Jamison Go's well known DDT.

Both robots took big hits, until

Shaka Zulu flipped upside down and danced circles around DDT while it tried to self-right. DDT's luck was running short, however, as he lost his next match against Poco Tambor and then lost to Algos. Saifu, built by Pete Smith of Kitbots, did well in the first round with its reversible direct drive brushless drum weapon. After winning its match against Algos, Saifu was sent to the loser's bracket by Capricant and was then eliminated by Klazo, which is based off the same kit as Saifu.

Last year's winner, Gilbert from Team MH Robotics, returned to reclaim his title at Clash of the Bots 3. After beating Ting Tang and Blue Devil, Gilbert lost its match against Poco Tambor and fought its way through the loser's bracket. Defeating WhipperSnapper and Calamity Kid secured Gilbert a spot in the finals.

Capricant stayed in the winner's bracket throughout the event, sending Poco Tambor to the loser's bracket late in the competition to battle Gilbert in the semi-finals. In the final match, Poco Tambor sent Capricant spinning in the air until Capricant lost power. Poco Tambor ended the Antweight matches with a gyro victory dance for the crowd.

BEETLEWEIGHT CLASS: Twelve 3 lb competitors put on a great show for the crowd. The first match paired Grande Tambor vs. OpenRobotix's TomZax — both with powerful vertical weapons. Eight year old Thomas Parrish behind the controls of TomZax tried to use his saw blade wheels to outdrive Grande Tambor, but Grande Tambor won the match after several big hits.

Two Florida teams took the stage next, as Voodoo Magic's titanium vertical disc cut



FIGURE 7. Berzerker vs. Dirty Sanchez.

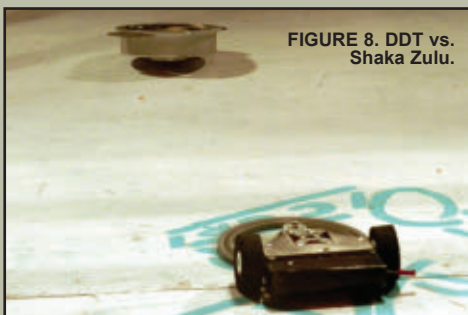


FIGURE 8. DDT vs. Shaka Zulu.

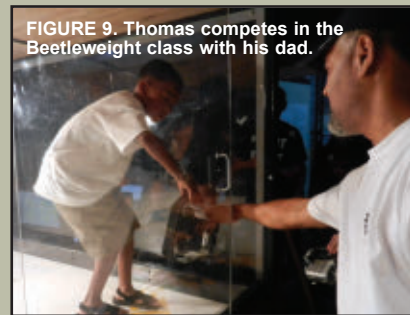


FIGURE 9. Thomas competes in the Beetleweight class with his dad.



FIGURE 10. OpenRobotix's Chobham 2.0, winner of Coolest Robot.



FIGURE 11. Bot hockey draws a crowd.

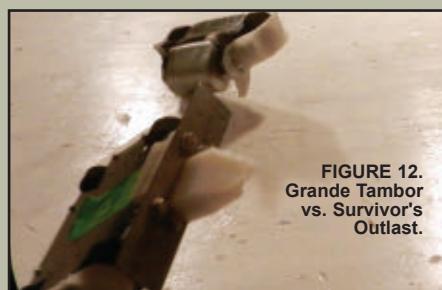


FIGURE 12. Grande Tambor vs. Survivor's Outlast.

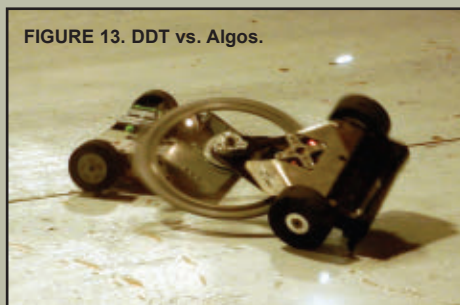


FIGURE 13. DDT vs. Algos.



FIGURE 14. Capricant vs. Poco Tambor.



FIGURE 15. Pissed Off Unicorn vs. Thrash.

horizontal-bar-spinner Steve's power switch early in the match.

Jamison Go brought a powerful new drum bot to this event, Dominant Mode, which delivered some devastating hits against Ramvac and Weta God of Ugly Things (last year's champion). In the latter match, Dominant Mode was flipped upside down, allowing Weta to get some good hits against Dominant Mode's titanium armor. All the kids took a step back from the arena as the shower of bright sparks hit the arena wall. Dominant Mode was eventually able to self right, and after two minutes of constant contact between the bots, the judges deemed Dominant Mode the winner.

DeJa Voodoo and Grande Tambor took the stage next for a forceful match between these two

drum bots that ended with DeJa Voodoo sending Grande Tambor into the ceiling. In the rematch between these two bots later in the bracket, Grande Tambor landed on the floor with so much force that both drive shafts sheared off, rendering Grande Tambor immobile.

Shame Spiral worked its way through the winner's bracket undefeated to meet DeJa Voodoo in the finals. After a serious driving skill show from both competitors, DeJa Voodoo's drum couldn't penetrate Shame Spiral's wedge. The match ended with DeJa Voodoo wedged between the arena barrier and the outside wall, and the judges named Shame Spiral the Beetleweight champion.

BOT HOCKEY: The bot hockey division drew a lot of attention as

each team of three robots tried to get the puck into the opponent's goal. First place went to Team Meatheads which consisted of Thomas Kenny from Team Meatheads, David Liaw of Team Ice, and Mike Jeffries of Team Near Chaos. Team Scotch Pies came in second place, and Team Pneusance took third place.

The "Coolest Robot" winner was OpenRobotix's Beetleweight Chobham 2.0, who received a Kitbots kit for its unique look. The other winners received prizes from the event sponsors: Kitbots, FingerTech Robotics, and *SERVO Magazine*. Chuck Butler would like to extend a "thank you" to the Schiele Museum for hosting the event and providing lunch to all the competitors, to all the event sponsors and volunteers that made the event a success, and to Robert Woodhead for recording the event in high speed. It was a great event with high quality robots, and I am already looking forward to Clash of the Bots 4! **SV**

TABLE 1. CLASH OF THE BOTS 3 WINNERS.

1st:	Fleaweight Pissed Off Unicorn - <i>Busted Nuts Robotics</i>	Antweight Poco Tambor - <i>Team Pneusance</i>	Beetleweight Shame Spiral - <i>MH Robotics</i>
2nd:	Thrash - <i>Busted Nuts Robotics</i>	Capricant - <i>Ram Robotics</i>	DeJa Voodoo - <i>Busted Nuts Robotics</i>
3rd:	Invertigo - <i>Busted Nuts Robotics</i>	Gilbert - <i>MH Robotics</i>	Grande Tambor - <i>Team Pneusance</i>
Bot Hockey:	Team Meatheads	Team Scotch Pies	Team Pneusance
Coolest Robot:	Chobham 2.0 - OpenRobotix		

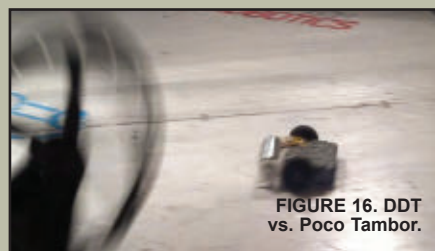


FIGURE 16. DDT vs. Poco Tambor.

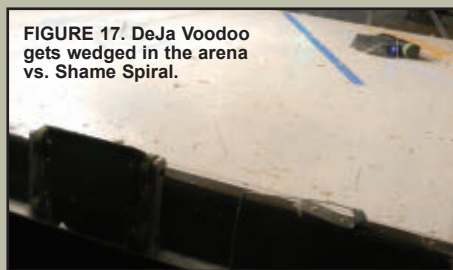


FIGURE 17. DeJa Voodoo gets wedged in the arena vs. Shame Spiral.



FIGURE 18. Dominant Mode vs. Weta.

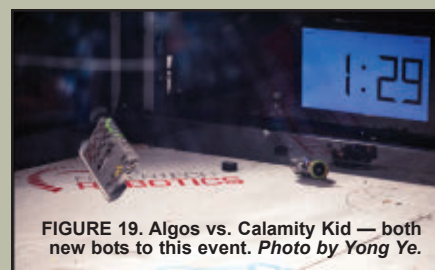


FIGURE 19. Algos vs. Calamity Kid — both new bots to this event. Photo by Yong Ye.



FIGURE 20. Grande Tambor vs. The Liberator. Photo by Yong Ye.



FIGURE 21. Shame Spiral vs. DeJa Voodoo. Photo by Yong Ye.



FIGURE 22. Poco Tambor vs. Gilbert. Photo by Yong Ye.

The History of Robot Combat

Robot Battles at Dragon*Con

● by Morgan Berry

Robot Battles at Dragon*Con in Atlanta, GA is one of the oldest robot combat events. In 1991 (for perspective, that's the same year that the World Wide Web was launched), a few years after the Denver Mad Scientists Society created Critter Crunch at MileHiCon in Denver (as discussed in the first History of Robot Combat article in the January 2012 issue), Kelly Lockhart was inspired to create a combat competition of his own. He used Critter Crunch as his model and hosted a simple event which soon became a mainstay of the robot combat world.

Lockhart was kind enough to tell the story of the early Robot Battle events:

"The genesis of Robot Battles started in Denver, CO back in 1987 when a group of engineering types created the 'Denver Mad Scientists Society' and staged a small robotics-oriented competition at a local

science fiction convention. The 'Criticrunch' event caught on (and is still going strong to this day), and word spread of the cool new idea.

"Come 1991, the organizers of the Dragon*Con convention in Atlanta approached me about staging something similar. So, I got a copy of the Critter Crunch rules from the Mad Scientists (of which I later became a member), and figured out how to stage it simply and with as little fuss and muss as possible.

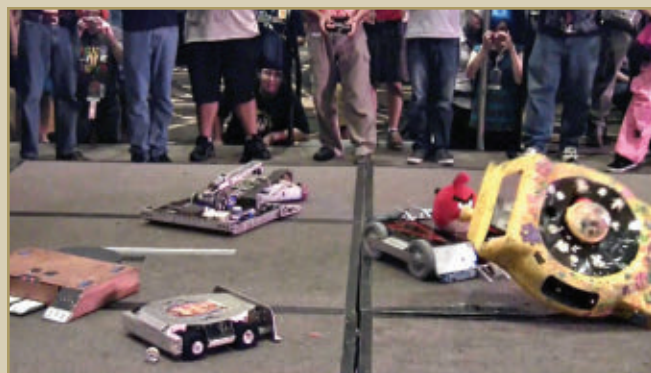
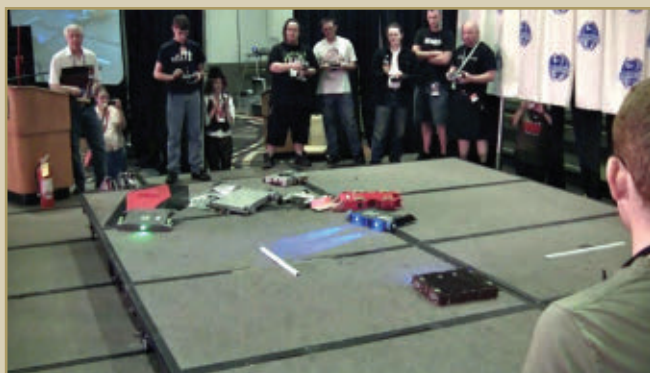
"My original vision is the one I still follow to this day: Have fun. I strive for everyone to have as much fun as possible — the builders, the drivers, and most of all, the audience.

"This vision is, I believe, what has kept Robot Battles going strong into our third decade. Because we never got bogged down by internal or external politics, we never got into money conflicts, and we have always

worked together to keep everything as entertaining as possible, we've survived when so many other events and competitions have fallen by the wayside.

"The first event had two robots entered. One was from a team from Georgia Tech, the other was built and driven by a student from the University of Georgia. Needless to say, there was a built-in rivalry from the very start. We were set up in the basement level of the Atlanta Hilton which at that time was bare concrete. We didn't have a stage, a PA system, lighting, video screens, or any of the things we take for granted today. What we did have was nearly 300 people surrounding the combat area (which was marked out with duct tape), cheering on the eventual winner ... the driver from UGA.

"The second year, we had seven competitors. Within five years, we had over 20 bots



FIGURES 1 and 2. In these shots of the rumble at the end of Robot Battles at Dragon*Con last year, you can see the raised platform arena that is unique to Robot Battles.

entered, had the use of a large ballroom to handle the SRO crowds, lights, sound system, the whole nine yards. It was obvious that we had hit on something.

"As we went along, we've changed weight classes, adapted to changing technology, moved to ever large venues, and spread out to other conventions around the country (including back in Denver for several years). Currently, we hold events in Chattanooga, Nashville, as well as Atlanta, and may be returning to Orlando after a two year absence.

"But even with all the changes and growth, we've stayed as true to the spirit of the original rules and vision as possible."

Aside from its long history, Robot Battles at Dragon*Con is unique in other ways. Unlike most robot combat events that take place in an enclosed arena, the bots at Robot Battles compete in an open air arena.

As a result, the competition is less about "the high energy whirrrrr-bang aspects of conventional tournaments" and more about "strategy, driving, and at times, straight up weirdness," competitor

Charles Guan said. This all makes for a very good spectator sport — a fact that the competitors happily embrace. "People regularly come up on stage in costume, or the robots look ludicrous themselves. My favorite in recent years was a very well driven and solid wedge robot that was coated in brown fur and made to look like a beaver."

The unique setup also leads to technological creativity. "There have been robots in the past which have automatic opponent-seeking sensors, as well as ones which could translate as they were spinning the entire robot frame by varying motor speed periodically.

You get robots made from materials that don't get used normally in arena combat like hardware store plastic window glazing and aluminum tubes because they're just not durable enough or hard enough," Guan said.

STEM Education is one of the most important issues in the technology world today. The folks at Robot Battles are doing their part to get the younger generation involved.

"Our audiences range in age

from very young (under 10) to folks that were around in the day when even the concept of robots was pure science fiction. And it's been that way from the very beginning," Lockhart says. "From the very first years, we have always encouraged young people to get involved. Our youngest tournament winner was eight years old, and he wasn't the only single digit age winner over the years. Some of our most popular and successful builders started when they were in their early teens, some even younger ... the other builders have always gone out of their way to teach and encourage the new builders."

Long time competitor Guan echoed Lockhart's sentiment. "The most well-known story of new builder triumph is an eight year old boy who built a plywood wedge shell around an otherwise unmodified R/C monster truck, and went on to defeat an entry from Georgia Tech."

Guan is himself an example of the youth presence at Robot Battles — this student has participated in the event for many years.

"Robot Battles has become one of the more popular events at

ADVICE AND TIPS FROM A COMBAT VETERAN

Charles Guan offers this advice to people considering competing in Robot Battles (or any robot combat event) for the first time:

"I encourage everyone who is thinking of entering a robot or interested in getting their feet wet in engineering hobbies to come and watch a competition. Once you experience the competition firsthand and talk to the builders and see the robots, you are far more likely to pick up and do it.

Viewing robot competitions over the Internet or hearing about it from friends, people often get discouraged because they think it's above their level or their ability. There is no place where this is more untrue than at a Robot Battles event — many winners in the past have been children or teens, some with their first and second robots. Once you finish your first contraption, I sincerely recommend driving it as often as you can. Get to know how your machine moves, because having a cool weapon is only half of the equation — keeping your robot moving and avoiding (or attacking) the opponent is the other.

Finally, when you get to the event, don't stress out. Relax, talk to fellow builders, introduce yourself, and spend some time practicing on the stage before the match starts. Remember that you are being watched by the audience just as much as the robot is. Some people get stage fright and stress out during the match, and that leads to poor performance and often hurt prides.

And no matter what, even if you don't win — especially if you don't win — always think of how you can learn from the experience, whether it is a new robot design, knowing what to fix for next time, or just from picking other builder's brains about how they accomplished something."

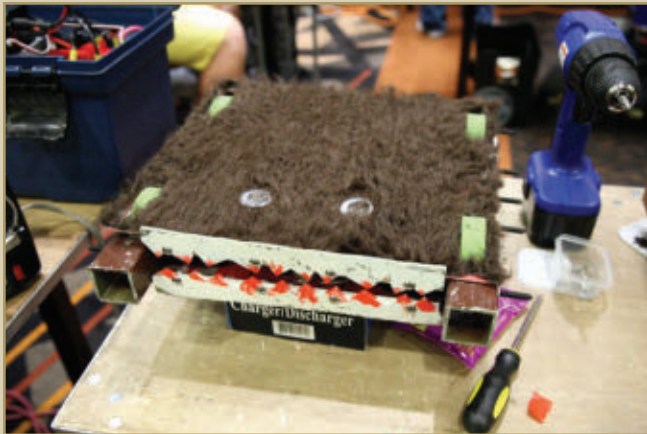
Dragon*Con. We have to hold the event over two days to accommodate the number of entries in the four weight classes (one, three, 12, and 30 pounds), in fact.

There are — on average — 10-20 robots entered in each weight class. And crowds range from 500+ for the Microbattles — which could easily double in size given a larger ballroom — to over 2,500 for the main event on Monday in the

second largest ballroom available at the Hyatt Regency," Lockhart commented.

When asked what the future held for Robot Battles, Lockhart had this to say: "My vision of the future of Robot Battles is the same as it has always been: Keep having fun. There are always small changes to venues, adaptation of new technologies, changes in marketing and publicity, and over time some builders move on while

new faces appear. But as long as we stick to the core philosophy to have fun for everyone, I think we'll keep being successful. I started Robot Battles when I was 22 years old. Twenty-two years later, it's still one of the most fun things I get to do every year, and I see no reason why I can't keep doing this for another two decades, or more. 'Two bots enter, one bot leaves ... often in pieces.' That is what I call fun." **SV**



FIGURES 3 and 4. These bots are an example of the kind of creative and lighthearted approach the builders at Dragon*Con take to Robot Battles.

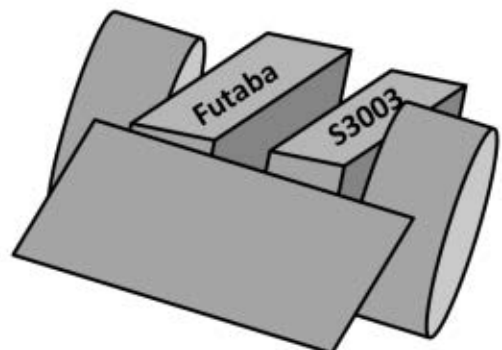
Melty Brains

by Kevin Berry

This month on Antique Botshow: Name that artifact!



"Frequency Clip"



"Servo Driven Antweight Wedge"

The Solution for Full Scaled Robots

DYNAMIXEL PRO

coming
soon



High Power, High Precision

Up to 200W BLDC motor and up to 500,000 pulse per turn (4096 absolute resolution)

Full Modular Solution

Ease of implementation (no additional design for parts needed)
Ease of maintenance (simple part swapping)

Sophisticated Control Algorithms

Position and speed input commands with dual-loop current control

Novel Gear Reduction System

High torque output, light weight, with high impact resistance

ROBOTIS
www.robotis.com

USA E-mail : america@robotis.com Tel : +1-949-333-3635
JAPAN E-mail : japan@robotis.com Tel : +81-3-4330-3660
Etc. E-mail : contactus2@robotis.com Tel : +82-70-8671-2609
OLLO, BIOLOID, and DYNAMIXEL are registered trademarks of ROBOTIS Co., Ltd.



Give Yourself a Wedgie!

by Zachary Lytle



As a four-time RoboGames champion, I am frequently asked how to get started in combat robotics. Personally, I always recommend building a wedge for your first robot because you don't want to start off with anything too complicated. Wedges are durable and easy to repair, which are the two most important qualities in your first robot. Plus, let's not forget that the current RoboGames heavyweight champion Original Sin is basically a gigantic wedge robot. In this article, I will show you how to build a wedge platform from scratch using common household tools. The boxes are locked, the lights are up, and the arena is waiting for you, so let's get started!

The tools and parts you'll need to construct your wedge are listed on the next page. The total cost of these parts should be around \$320. Starting with the appropriate materials can save you a lot of trouble down the road. Let's begin.

BUILDER TIP: Always weigh your parts to see how much poundage you have to work with.





The tools you'll need are:

- Hand drill
- Soldering iron with solder
- Six inch caliper
- Lighter
- English Allen wrench set
- 1/8" drill
- Tin snips
- Wire stripper
- Sharpie™ pen
- Fine-grained file
- 2 x 3 inch clamps
- Bottle of CA glue

Optional:

- Dremel with cutoff wheel
- Drill press
- Five pound scale



The parts you'll need are:

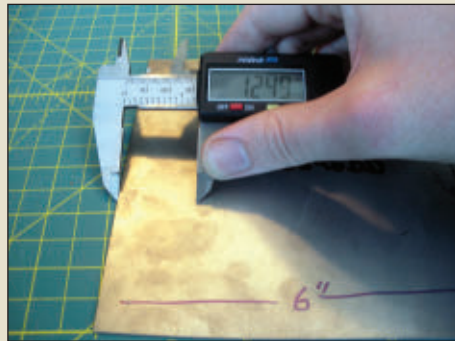
- 6" x 6" titanium sheet
- 2x FingerTech Robotics 22-1 Silver Spark motors
- 2x FingerTech tinyESC v2 motor controllers
- 2x Lite Flite wheels, two inches in diameter
- 2x FingerTech Lite Hubs
- JST connector (female)
- DX 5E radio system
- Rhino 460 mAh 2S 7.4V Li-Poly pack and charger
- 6" x 6" polycarbonate sheet .040 inches thick
- 25x 4-40 button head screws 1/4 inch in length
- 2x BaneBots motor mounts

Most of the parts can be found at **FingerTechRobotics.com**; the metal and screws can be found at **www.mcmaster.com**; and the motor mounts are from **Banebots.com**.

MAKING THE TOP PLATE

This is the primary armor and structure for the entire robot. All of your components will be mounted to this single piece of metal. Some might call this "putting all your eggs in one basket," but I can tell you from experience this is the best approach.

By putting all your weight into a singular piece of metal, it allows you to build the frame and armor on top of each other, and this makes them inseparable. Designs that make the armor a removable piece typically get it ripped off and destroyed.



middle, and scribe two additional lines 1-1/4 inches from the left and right hand side. This will create three cross hairs along the horizontal line. When you are done, you will have seven crosshairs.

STEP 1: LAYING THE HOLE PATTERNS

Lay out the bolt hole patterns with the caliper. By locking the caliper at the desired dimension, you can use the caliper as a ruler to scribe lines.

Set the caliper at 1/8 inch and scribe a line down each side of the titanium sheet.

Set the caliper at 1/4 inch and scribe two additional lines, measuring from the top and intersecting your first two 1/8 inch lines. This will make a cross hair you will use to line up the drill.

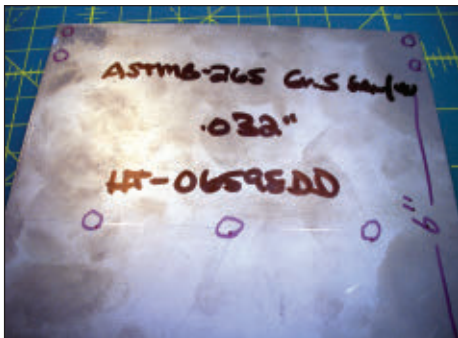
Scribe a second set of small lines, 3/4 inches down from the top.

Next, scribe a line two inches off the bottom edge all the way across.

Then, scribe a line directly in the

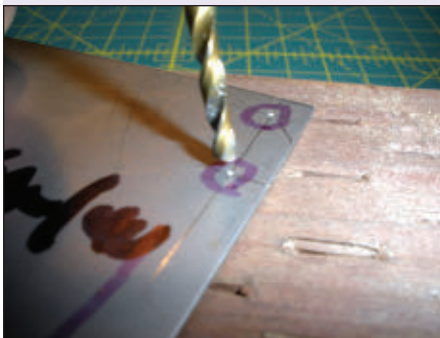
STEP 2: MARKING THE HOLES

Be sure to take your time on this step. After making the seven scribed crosshairs, take the punch and the hammer and create a small divot over each cross hair. These small divots will guide the drill through the material. Hard metals — like grade 5 titanium — typically cause drills to “walk” off the desired location. By scribing and marking holes with the punch, you will gain a much higher level of accuracy with the drill.



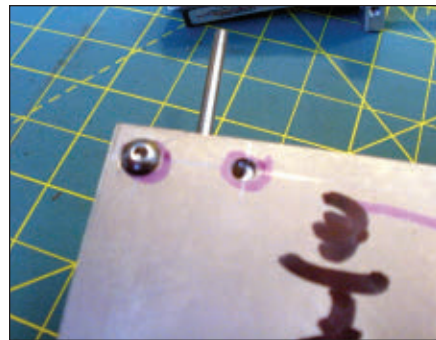
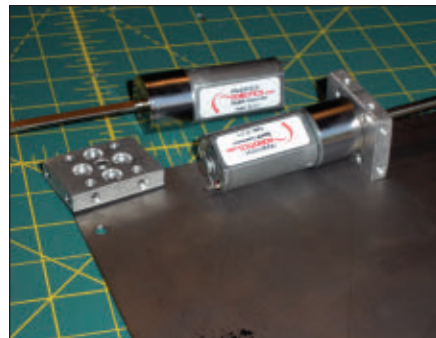
STEP 3: DRILLING THE HOLES

The last step for the titanium is to drill it. Put your hand drill on a low speed at approximately 500 rpm, and make sure the 1/8 inch bit is securely locked in the chuck. Place a block of wood underneath the titanium plate. Take your spray can of WD-40 and spray the drill and sheet before you start drilling. Place the drill in the divot and get lined up, straight up and down. Apply moderate pressure while drilling.



STEP 4: CHECKING YOUR HOLES

Hold the motor mount up to the titanium plate and visually check to be sure you can see both tapped holes of the motor mount through the 1/8 inch holes in the titanium plate. If the holes are not visible, drill the holes out with a larger sized bit. Or, use a Dremel tool with an end mill ball attachment to turn the holes into slots. Make sure the motor mount will fit before moving to the next step.



BUILDER TIP:

If the titanium proves too hard to penetrate with a hand drill, use a drill press.



ASSEMBLING THE DRIVETRAIN

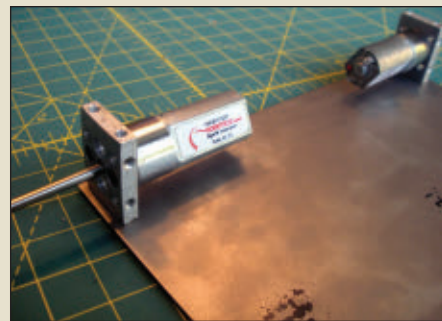
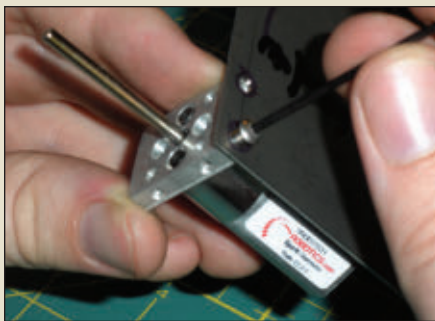
STEP 5: MOUNTING THE MOTOR MOUNTS

Attach the two motor mounts to the motors. Slide the motor mount over the motor shaft, and place it on the front of the gear box. Rotate the motor mount until the two tapped holes are visible through the guide holes on the motor mount. Using the 256 screws and the .05 inch Allen wrench, screw in two of the screws through the guide holes into the tapped holes on the motor. After mounting both motors to



the motor mounts, you will notice the motor mounts have tapped holes on all sides. Pick one of the two long sides and line up the tapped holes with the 1/16 inch holes on the titanium plate.

Using the 1/16 Allen wrench and two 440 screws, screw them through the guide holes on the titanium plate into the tapped holes on the motor mount. Repeat this process on the other side of the robot, so both motors are mounted to the titanium plate.



STEP 6: MOUNTING THE WHEELS

Take the FingerTech hubs and glue them to the lite flite tires. Start by running a line of glue down the length of the Fingertech hub, and slide a lite flite wheel over the shaft of the hub. Repeat this process on both wheels.

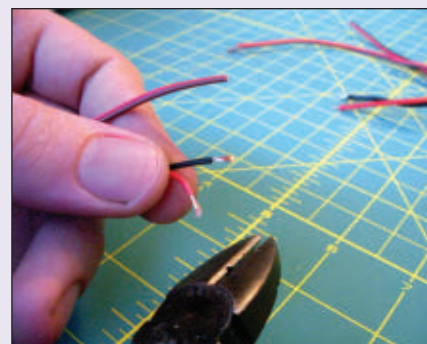
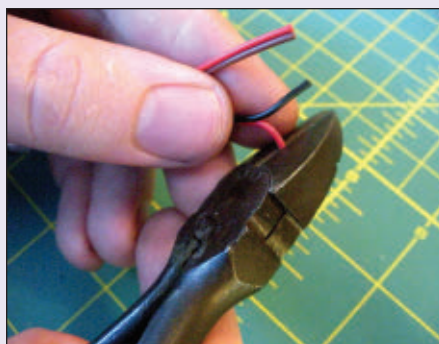
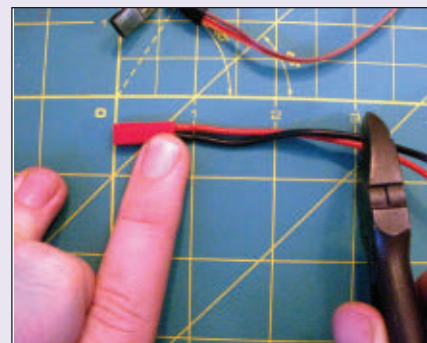
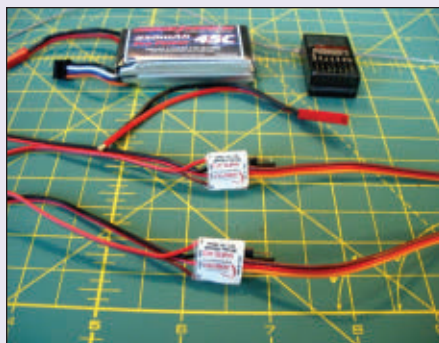
Slide both assembled wheels over the drive shafts of your robot. Line the set screw of the hub to the flat on the drive shaft. Warning: If this step is skipped, your wheels will fall off! Use the .05 inch Allen wrench to tighten the set screw in the hub.



SOLDERING THE ELECTRONICS TOGETHER

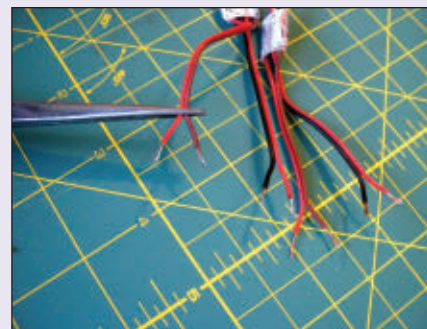
STEP 7: STRIPPING THE WIRES

We are going to start by stripping the wires on both FingerTech controllers. Also strip the red and black wires on the female JST plug.

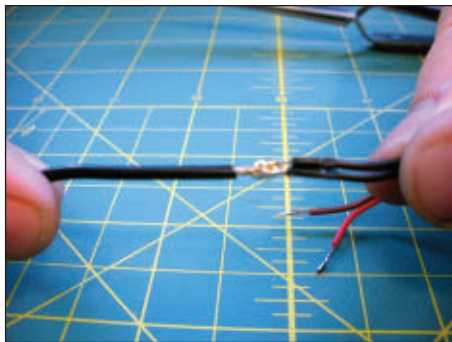
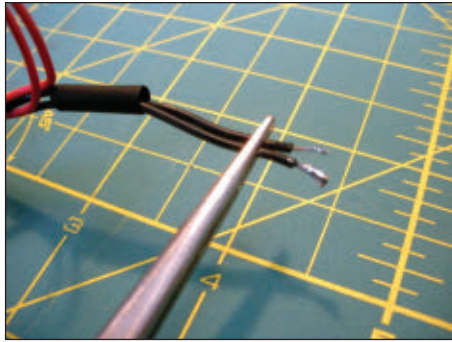
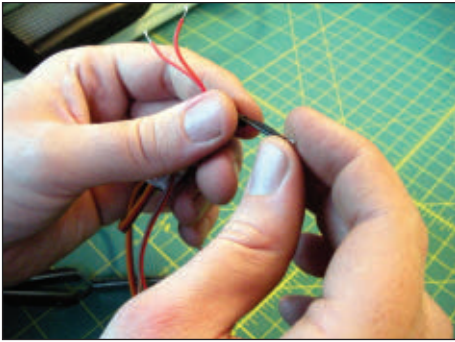


Tin all six leads. To tin each lead, get the soldering iron to full temperature. It usually takes about 10 minutes on most irons.

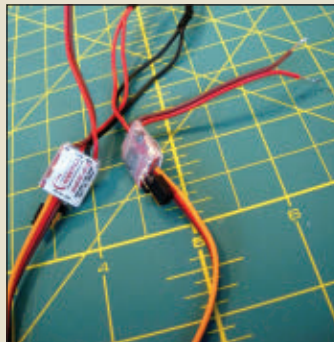
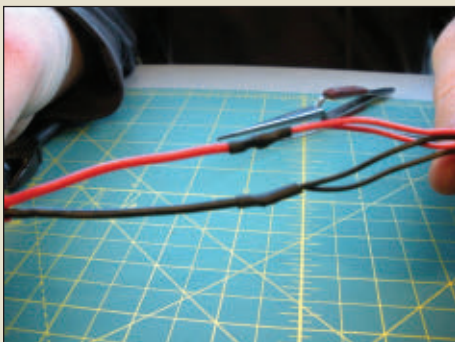
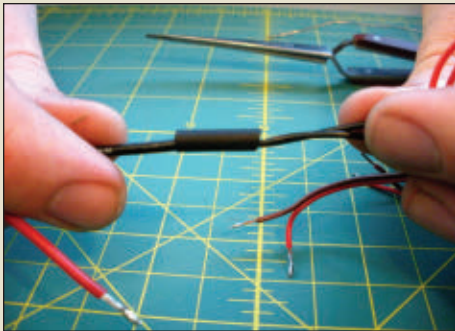
Heat the wire with the iron while placing the solder on the other side of the wire. If the wires are not taking the solder, apply a little bit of solder to the iron. After tinning all six leads, we will now join the three black leads together.



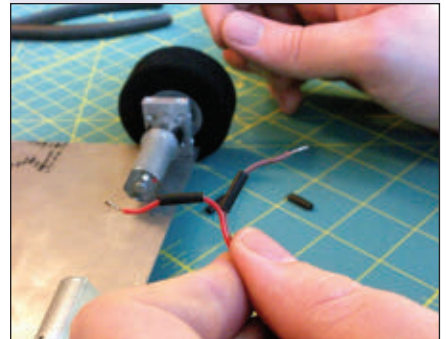
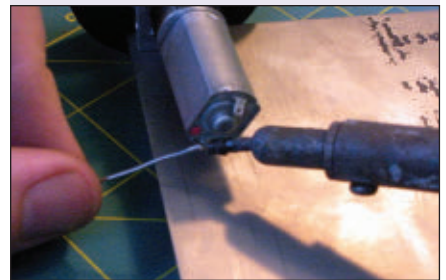
Slip a piece of heat shrink over the two black speed controller leads. Solder the JST connector to one of the speed controller leads. I recommend putting the speed controller lead in a vice or pair of pliers, then holding the soldering iron in your dominant hand and the JST connector in the other hand. Place them together so they are horizontally touching. Heat both wires with the iron until the tinned leads melt together.



Once all three wires are soldered together, slip the heat shrink over the solder joint; make sure to cover all exposed wire. Use a lighter or other small flame to compress the heat shrink around the solder joint; make sure the flame does not touch the heat shrink. After joining the black leads together, repeat the process and join the red wires together using the same steps you did on the three black wires.



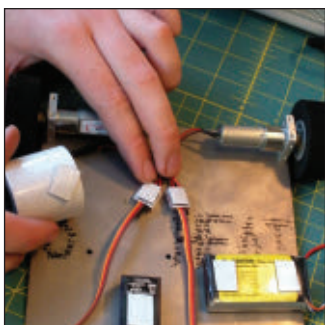
Now, let's solder the speed controllers to the motors. Tin the red and brown leads on both speed controllers, then tin both motor leads. When looking at the back of the motor, you will notice one lead has a red dot next to it. This indicates it is the positive lead. You will then solder the red wire to the positive lead on the motor. On the second motor lead — which is not marked — you will solder the brown wire. Repeat this process on both motors and speed controllers. Once you've soldered the last brown lead, you will have officially soldered together your first combat robot!



MOUNTING THE COMPONENTS

Cut out pieces of double-stick tape, and stick one piece to the back of the receiver and a second piece to the back of the Li-Poly battery. Stick the receiver just below the right motor, and stick the battery just below the left.

Now, cut two more small pieces of double-stick tape and attach them to the speed controllers down between the motors. Plug the long receiver leads coming off the speed controllers into the receiver in the aileron and elevator slots.



BUILDER TIP:

Put the double-stick tape on the label side of the speed controllers. Although this will cover the label, it will make the LED visible.



CONNECTING THE RADIO TO THE RECEIVER

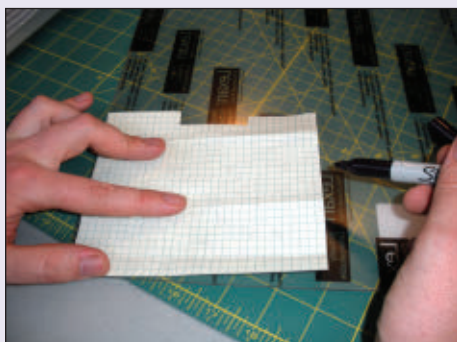
Plug the "bind" plug into the receiver on the battery port. Then, power on the robot by plugging the 7.4 volt battery into the JST lead. The receiver should start, showing a blinking orange light. Now, on your radio, flip the mix channel to the upright position. Flip the elevator and mix switches to the upright position, as well.

Hold the "train" switch and turn the radio on. You will wait to see the blinking light on the receiver and the transmitter go solid. This indicates the transmitter has "bound" to the receiver.



MAKING THE BOTTOM PLATE

When I originally designed the bottom plate for this wedge, I used a piece of cardboard (or "cardboard aided design"). I have provided a cut-out for you at the article link to trace onto polycarbonate. With polycarbonate this thin, you can cut it with hand shears and bend it in a vice. Carefully lay the pattern out and with either a pencil or fine-tip Sharpie, trace the pattern onto the polycarbonate. Then, using the hand shears, cut out the bottom plate. The dotted lines on the pattern represent where it will be bent in the vice.



One special note: I don't drill the holes until after the material is bent. This goes against many shop practices, however, for garage and hand tools I believe this is a better approach. Start by bending the front of the plate to a 30 degree angle. With polycarbonate, you will need to bend it past the point and see where it returns. Move to the dotted line — 1.5 inches from the top — and bend it to 30 degrees. The tip of the plate and the middle of the plate should be lined up and pointed in the same direction.



Flip the plate over and bend the back panel to 90 degrees. Bend the two small side tabs up to 90 degrees. Place the covering over the two motor mounts, and check the fit. You may need some additional tweaking to get the piece to sit flat against the two motor mounts and the front of the titanium sheet.



FINISHING YOUR ROBOT

Once the bottom plate sits flat on the robot, use the fine-tip Sharpie and mark the locations of the tapped holes on the motor mount. Because the polycarbonate is clear, you can lay it on the robot and see where the tapped holes are. Once you have marked them with a pen, use the same punch, hammer, and drill technique described previously to drill out the marked holes.



Screw down the polycarbonate plate with four 440 screws. Then, place your robot face up on a block of wood; make sure you can see the wood through the three titanium holes on the front of the robot. Using your 1/8 inch drill, penetrate through the polycarbonate sheet and titanium plate. After drilling each hole, add 440 screws to help hold the polycarbonate sheet in place.



BUILDER TIP:

You can make your wedge flush to the ground by filing the edge of it.



"WHEELS ON THE GROUND" TEST

If the robot's steering is incorrect, try flipping the aileron and elevator switches to different positions. There should be four different combinations. If none of the four combinations work, you will have to open the robot and flip the two speed controller plugs to opposite ports. However, your first power-on should be correct.

You have now finished a one pound version of my 150 gram robot Wadgie. Wadgie has served me faithfully for five years now. He has endured more hits than any of my other robots. The design has worked well for me. I hope it will work well for you.



READY FOR THE ROAR OF THE CROWD

Learning how to construct an actual platform for a robot is an important first step. Once you have mastered the basics with your durable wedge bot, you can then put your imagination and your newly learned building techniques to work to construct a robot entirely of your own creation. A further benefit is you will have a basic wedge robot to practice against. I never started winning matches until I had two robots and practiced driving them at home.

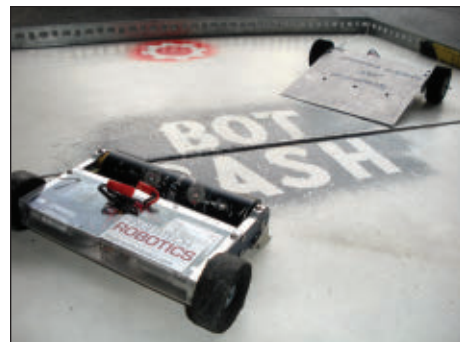
Please remember to be safe and follow good safety procedures. Wear safety glasses through the entire build process. Do not use power tools you're unfamiliar with.

Stay tuned to *SERVO Magazine* for a future article where I will show you how to add a lifter or a clamp onto your robot.

I wish you good building and I

hope to see you in the ring.

SV



www.servomagazine.com/index.php?/magazine/article/ctober2012_Lytle

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

FINDING NEMO10

Take the pain out of putting your robot's remote sensor to work by translating its RS-232 data stream into a TCP/IP packet.

by Fred Eady

Gathering sensor data with a robotic device usually entails some sort of 802.15.4 network. If you want to forward the collected data to a remote site, then TCP/IP is required. The hardware and firmware to move those special little bits of sensor data can be very complicated to build and program. However, I've devised a way to take the pain out of putting a remote sensor to work. My method does not require one byte of user written TCP/IP code and you don't have to scratch-build any Ethernet devices.

www.servomagazine.com/index.php?/magazine/article/october2012_Eady

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

The Big Picture

Let's start at the sensor end. It really doesn't matter what you are sensing or what you are using for a sensor as long as it can provide analog or digital data that can be represented in binary. The sensor passes the binary data along to a resident microcontroller via the microcontroller's analog or digital inputs. Once the microcontroller massages the data and assembles it into a data package, the formatted data is passed to a short-range low power 802.15.4-based radio.

Radios need each other. So, there's probably another 802.15.4 radio within ear shot of the sensor radio. The data stops here unless you have equipment installed that will take those sensor bits on a ride to somewhere else.

I'll leave the nature of the sensor to you. However, I'm going to put a PIC18F4620 between your sensor and a Microchip MRF24J40MA 802.15.4 data radio. On the remote end, I'll place another PIC18F4620 behind yet

another MRF24J40MA data radio. To keep the sensor-generated bits from piling up in the bit bucket, I'll attach the remote PIC18F4620 to a device server.

Device Server??

That's what I said. A device server is an electronic module that accepts a non-IP-based protocol and formats the data represented by the protocol into an addressable IP-based protocol that can travel on a LAN, a WAN, or the Internet. In our design, the non-IP protocol will begin with a START bit followed by eight data bits and a STOP bit. If you added NO PARITY to the aforementioned data stream, you correctly identified our mystery protocol as RS-232. Thanks to the PIC front end, our device server can accept any serial protocol that can be represented in byte format. The device server of choice is offered by the SENA Corporation and is called the NEMO10.

NEMO10 is billed as a single-chip network enabler module. A glance at **Photo 1** seems to prove that out. Actually, NEMO10 is a multi-chip serial-to-Ethernet converter housed in a compact DIL configuration. NEMO10 is composed of an 8032 microcontroller, 256 KB of static RAM, program Flash, 64 KB of EEPROM, and a NIC (Network Interface Controller). The NEMO10 is network-ready because it is shipped with a unique burned-in MAC address. The only networking hardware component missing is the Ethernet magnetics module. The NEMO10 is designed to interface with the XFMRs, Inc., XF10BASE-COMBO1-2S Ethernet magnetics/filter combo module which is under the lens in **Photo 2**.

Like many of today's modern microcontrollers, the NEMO10's UART is indigenous to the microcontroller's silicon. In addition to the Ethernet and

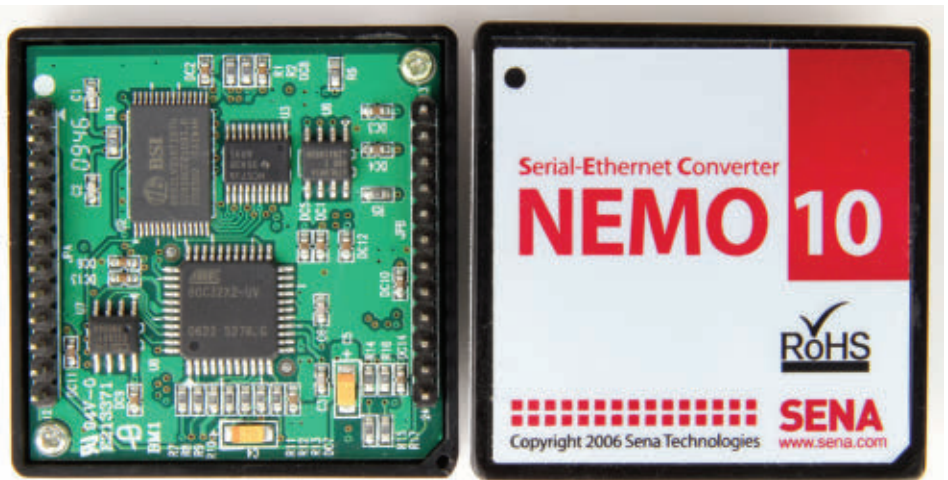


PHOTO 1. The NEMO10 is a compact serial-to-Ethernet converter module designed to interface any RS-232-based device to a LAN or the Internet.

serial interfaces, the NEMO10 provides all of the necessary I/O drive for indicator LEDs.

Using ARP, NEMO10 can resolve hardware (MAC) addresses using known IP addresses. The NEMO10's ability to act as an ICMP client allows it to respond to a network ping. NEMO10 can request an IP address from a DHCP server and communicate

via Telnet. NEMO10 can request an IP address, which infers that it may be able to perform other IP-based client and server functions using IP's best buddy, TCP. In fact, the NEMO10 can be configured as a TCP client, a TCP server, or a combination of both.

If the NEMO10's serial port is not available as a console port, a Telnet session can be used to manage it

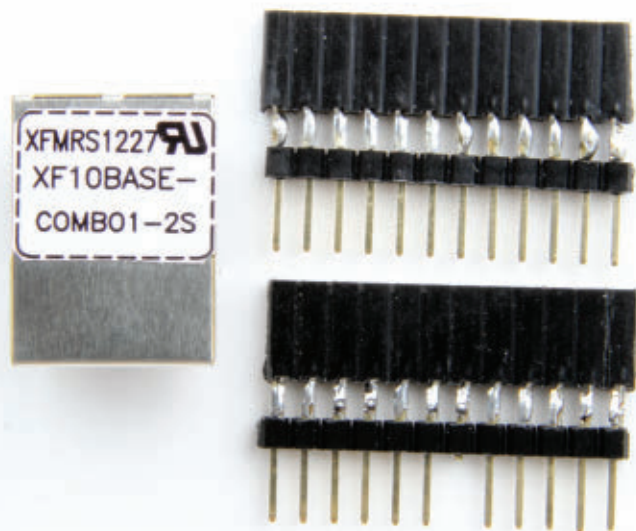


PHOTO 2. You'll need this baby on the other side of the NEMO10 which is designed to interface to Ethernet devices using this XFMRs, Inc., Ethernet magnetics module. There's also a pair of homemade headers in this shot. You'll find out why one is missing a pin later on.

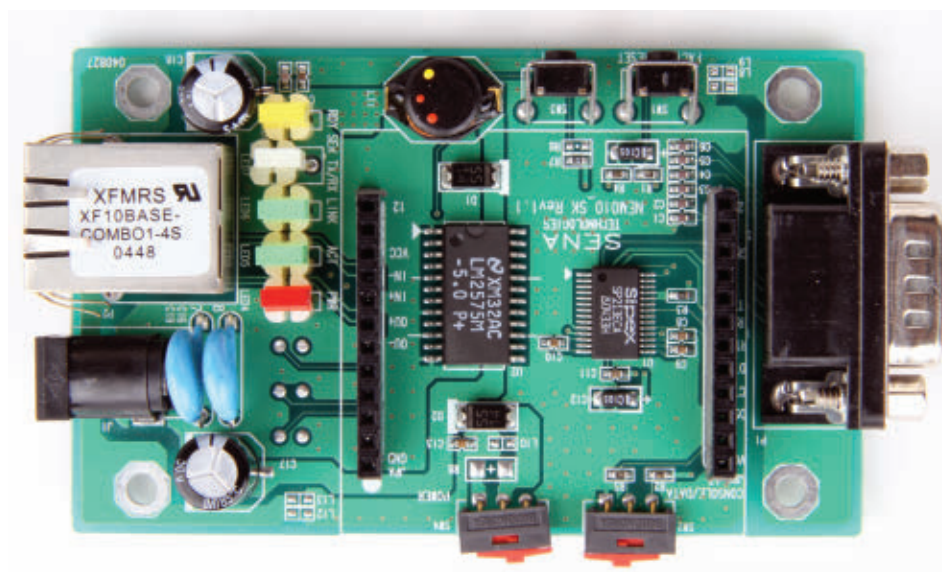


PHOTO 3. The NEMO10-SK is a full-blown NEMO10 support system that includes RS-232 and Ethernet interfaces.

locally or remotely. There's also a PC application called HelloDevice Manager that can be used to manage the NEMO10.

NEMO10 Setup

We can choose to configure the NEMO10 via its serial port or by using a Telnet session. At this point, our NEMO10 network consists only of what you see in **Photo 1**. To

configure the NEMO10 by Telnet, we'll need to attach the XF10BASE Ethernet magnetics/filter combo module to it. If we decide to configure the NEMO10 using RS-232, we'll need an RS-232-to-TTL converter and a console/data toggle switch to allow us to force the NEMO10 to enter console mode. In either case, we'll also need to power the NEMO10 with a power supply that can provide a minimum of +5 volts at 60 mA.

I promised we wouldn't scratch-

build any Ethernet hardware. The same goes for RS-232 hardware. So, instead of swinging a soldering iron, we'll support the NEMO10 with a test bed called the NEMO10-SK. The NEMO10-SK shown in **Photo 3** incorporates a male nine-pin RS-232 connector that feeds an RS-232-to-TTL converter IC. An XF10BASE Ethernet magnetics/filter combo module is providing a 10 Mbps Ethernet interface at the opposite end of the NEMO10-SK. Power for the NEMO10 is produced by a switching power supply that is based on an LM2575. To take advantage of the NEMO10's entire range of functionality, the NEMO10-SK also includes a reset switch, a factory reset switch, a data/console switch, a power switch, and a set of status LEDs. The NEMO10-SK test bed interfaces to the NEMO10 module with a set of 0.1 inch pitch female headers. The NEMO10 is shown attached to the test bed in

Photo 4.

If you don't have a network set up to support the NEMO10, using a serial connection is the easiest way to configure it. The NEMO10's serial interface defaults to 9600 bps, eight data bits, no parity, and one stop bit. **Screenshot 1** was produced by flipping the data/console switch to the console position and using HyperTerminal to serially log into the



PHOTO 4. The NEMO10-SK's female headers supply power to the NEMO10 and connect its module's internal electronics to the RS-232 and Ethernet interfaces.

NEMO10. I requested the NEMO10's current IP and serial port settings using its *get ip* and *get serial* commands. As you can see, the NEMO10 comes up with DHCP mode enabled. Since there is no network, DHCP is useless and there is no IP address at this point.

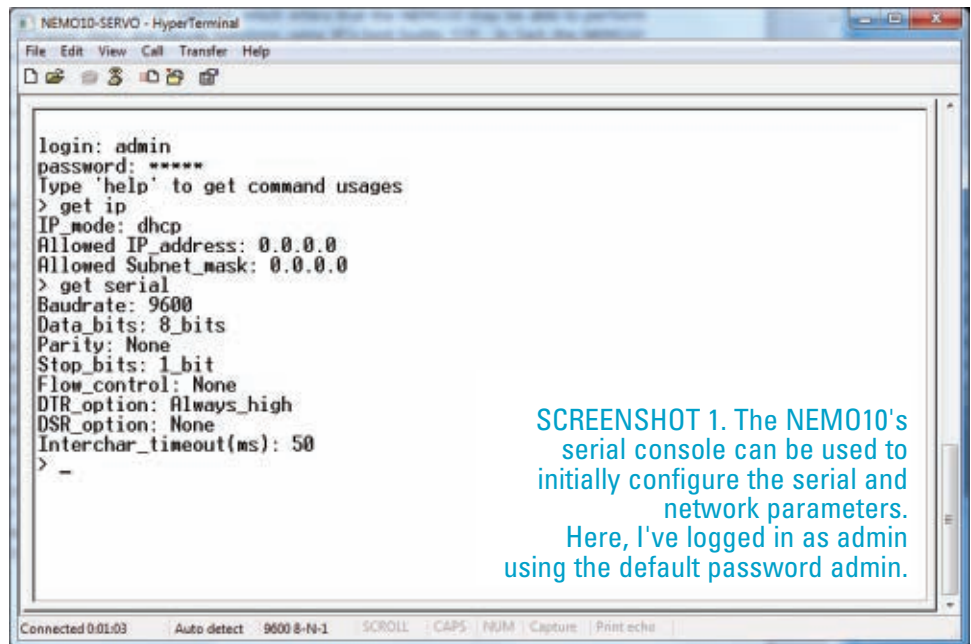
If we left the NEMO10 in DHCP mode, we may never be able to contact it remotely. Normally, DHCP IP address leases are set to expire and force the node to request a new and different IP address. We must always depend on the NEMO10 to be identified by one particular IP address. So, our first command will be to assure the NEMO10 can always be located via its IP address:

```
set ip static 192.168.1.97
255.255.255.0 192.168.1.1
```

With the NEMO10 *set ip* command, we assigned a static IP address of 192.168.1.97 regulated by a subnet mask of 255.255.255.0. The network router's (gateway's) IP address is 192.168.1.1. Since we didn't specify and filter parameters, the IP address and subnet mask filtering remain disabled by default. We must use the NEMO10 *save* command to retain our configuration changes and the *reboot* command to bring them into effect.

Now that we can operate on a DHCP-enabled network with a static IP address, just for grins let's throw the data/console switch back to data mode and do the rest of the NEMO10 configuration with a Telnet session.

Using HyperTerminal in Winsock mode, I logged into the NEMO10's Telnet console port (23) using its newly assigned static IP address (192.168.1.97). As you can see in **Screenshot 2**, the NEMO10 is now in static IP mode and all of the IP configuration data we



SCREENSHOT 1. The NEMO10's serial console can be used to initially configure the serial and network parameters. Here, I've logged in as admin using the default password admin.

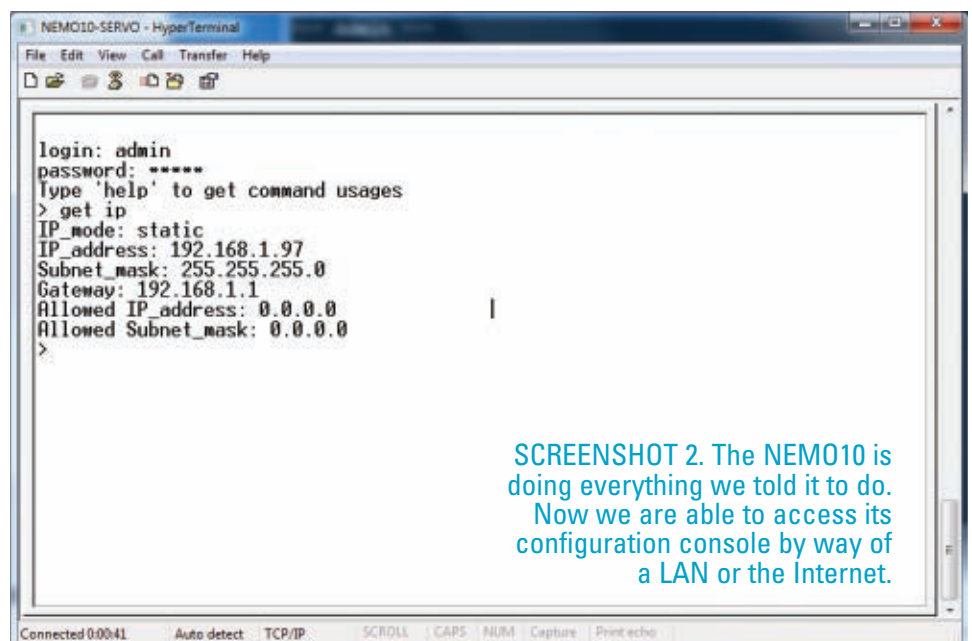
entered is in place.

I issued a *get* command to retrieve the NEMO10's Admin, IP, Host, and Serial status you see in **Screenshot 3**. The IP configuration is just like we want it. However, we need to do a little bit of twiddling in the Host area. The *Host_mode* defaults to TCP Server (tcps); that's fine since we want to serve our sensor

data. I don't like the *Local_port* assignment and the *Inactivity_timeout* is a bit much. So, let's use the *set host* command to alter the Host configuration:

```
set host tcps 9000 10
```

Our TCP server now times out in 10 seconds instead of 300



SCREENSHOT 2. The NEMO10 is doing everything we told it to do. Now we are able to access its configuration console by way of a LAN or the Internet.


```
NEMO10-SERVO - HyperTerminal
File Edit View Call Transfer Help
[Icons]

Username: admin
Password: admin
Devicename: NEMO10 Device
--- IP ---
IP_mode: static
IP_address: 192.168.1.97
Subnet_mask: 255.255.255.0
Gateway: 192.168.1.1
Allowed IP_address: 0.0.0.0
Allowed Subnet_mask: 0.0.0.0
--- Host ---
Host_mode: tcps
Local_port: 6001
Inactivity_timeout(sec): 300
--- Serial ---
Baudrate: 9600
Data_bits: 8_bits
Parity: None
Stop_bits: 1_bit
Flow_control: None
DTR_option: Always_high
DSR_option: None
Interchar_timeout(ms): 50
> -

Connected 0:01:45  ANSW  TCP/IP  SCROLL  CAPS  NUM  Capture  Print echo
```

SCREENSHOT 3.
Here's how the NEMO10 is configured at this point. There are still a few parameters we'll want to wiggle.

seconds, and the TCP server's local listening port is 9000.

The NEMO10 serial console is hard-coded to run at 9600-8-N-1. However, we can specify differing baud rates, bit lengths, parity, and stop bits that comply with the serial configuration of the device. With that, let's serve the remote PIC18F4620 at 19200 bps, eight data bits, no

parity, and one stop:

```
set serial 19200 8 n 1 n h n
50
```

The *set serial* command we just issued sets our new device baud rate at 19200 bps with eight data bits, no parity, one stop bit, no flow control, DTR always high, DSR "don't care"

with an inter-character timeout of 50 ms. While we're at it, let's identify the sensor node as SERVO-SENSOR:

```
set admin * * SERVO-SENSOR
```

The asterisks retain the previous values of the *set admin* command's *Username* and *Password* parameters. Only the *Devicename* parameter is changed as verified by **Screenshot 4**. Note also that the device serial port now has a baud rate of 19200 and the *Devicename* has changed. At this point, another *save* and *reboot* is in order.

We have one more configuration task to complete. It's not a physical NEMO10 task, but it affects the NEMO10 directly. We've got to punch a hole in our gateway (router) to allow the outside world to access the NEMO10's server functionality. All we have to do is tell the gateway to accept and allow passage of incoming TCP traffic addressed to 192.168.1.97 port 9000. I've done this in **Screenshot 5**.

NEMO10 at Your Service

We don't have an 802.15.4 network running yet. However, that won't stop us from serving up some ASCII with SERVO-SERVER. I've always shown you data transfers with either HyperTerminal or Tera Term Pro. There's a really good terminal emulator/serial analyzer that comes with the CCS PIC C compiler. It's called the Serial Input/Output Monitor. I've set up a Winsock connection using the Serial Input/Output Monitor in **Screenshot 6**.

The PIC18F4620 connected serially to the NEMO10 is part of a PICDEM Z development board. The PICDEM Z development board you

```
NEMO10-SERVO - HyperTerminal
File Edit View Call Transfer Help
[Icons]

Username: admin
Password: admin
Devicename: SERVO-SENSOR
--- IP ---
IP_mode: static
IP_address: 192.168.1.97
Subnet_mask: 255.255.255.0
Gateway: 192.168.1.1
Allowed IP_address: 0.0.0.0
Allowed Subnet_mask: 0.0.0.0
--- Host ---
Host_mode: tcps
Local_port: 9000
Inactivity_timeout(sec): 5
--- Serial ---
Baudrate: 19200
Data_bits: 8_bits
Parity: None
Stop_bits: 1_bit
Flow_control: None
DTR_option: Always_high
DSR_option: None
Interchar_timeout(ms): 50
> -

Connected 0:46:25  ANSW  TCP/IP  SCROLL  CAPS  NUM  Capture  Print echo
```

SCREENSHOT 4.
With all of the IP, Host, and Serial parameters configured to our taste, it's time to test our SERVO-SENSOR device server node.

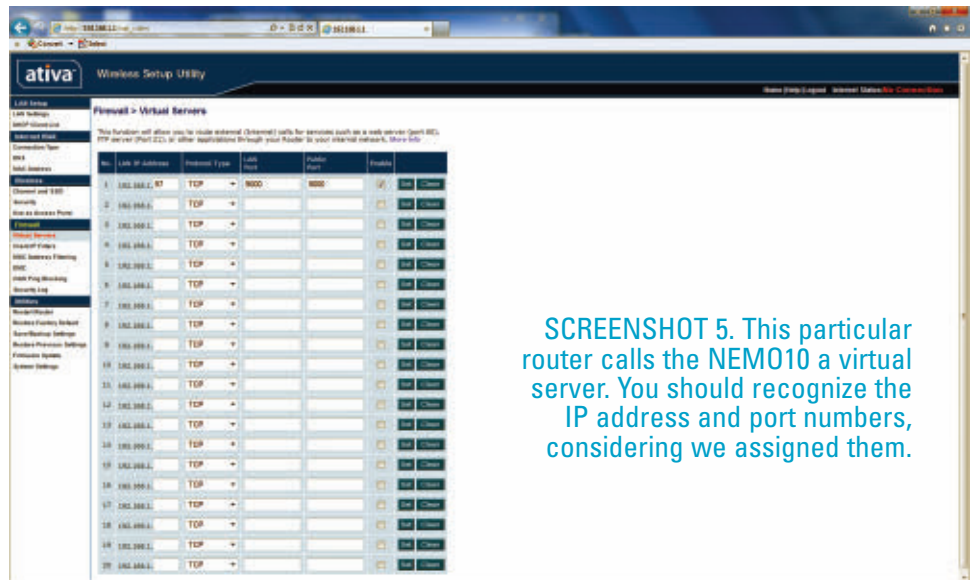
see attached to the NEMO10 in **Photo 5** is one of a pair of 802.15.4-capable development boards. The NEMO10-to-PICDEM Z direct serial port connection is possible because the NEMO10 is a DTE (Data Terminal Equipment) device. Your PC is configured as a DTE device. The PICDEM Z is wired for DCE (Data Communications Equipment) operation. Modems are DCE devices. A DTE device's transmit pin is positioned in the connector to directly connect to the DCE device's receive pin. The same holds true for the DTE's receive pin and DCE's transmit pin. Add a ground pin to this equation and you have what is termed the three-wire serial interface.

Now that we have a device server to device serial connection, let's write a simple piece of code to spin something out of the PICDEM Z's serial port:

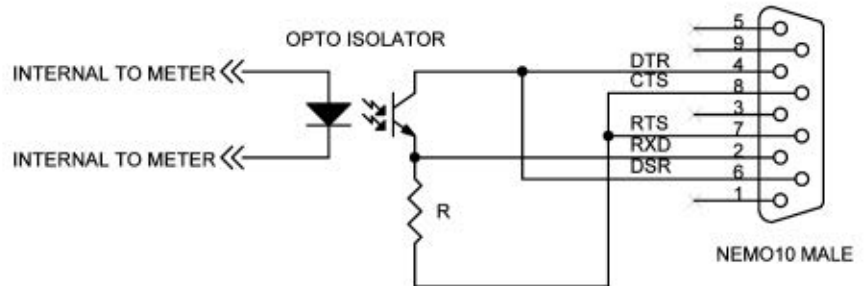
```
unsigned int servo = 0;
do{
    Printf("YOU HAVE CONTACTED
    SERVO-SERVER ");
    PrintDec(servo++);
    Printf("\r");
    DelayMs(1000);
}while(1);
```

The *Printf* statements are not your normal *printf* statements. The *Printf* and *PrintDec* are user written routines that do not accept any of the normal *printf* formatting. Basically, the code snippet runs forever, sending the incrementing servo variable message to the NEMO10's serial port.

Everything is now in place and we can perform a preliminary test. Clicking on the Serial Input/Output Monitor CONNECT button should establish a TCP session with the NEMO10. If our little test program is running, we should see the message from the PIC18F4620. I've started the PIC and I can see that the NEMO10's serial RX LED is blinking.

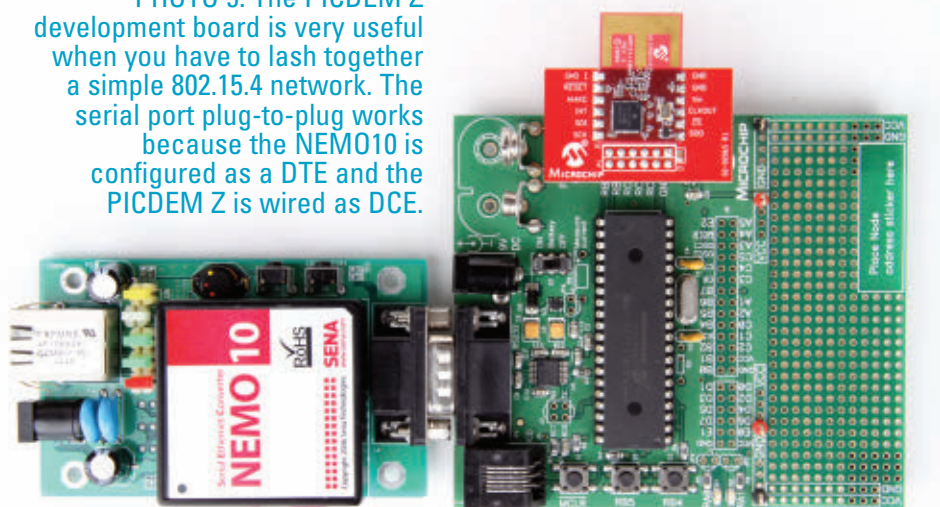


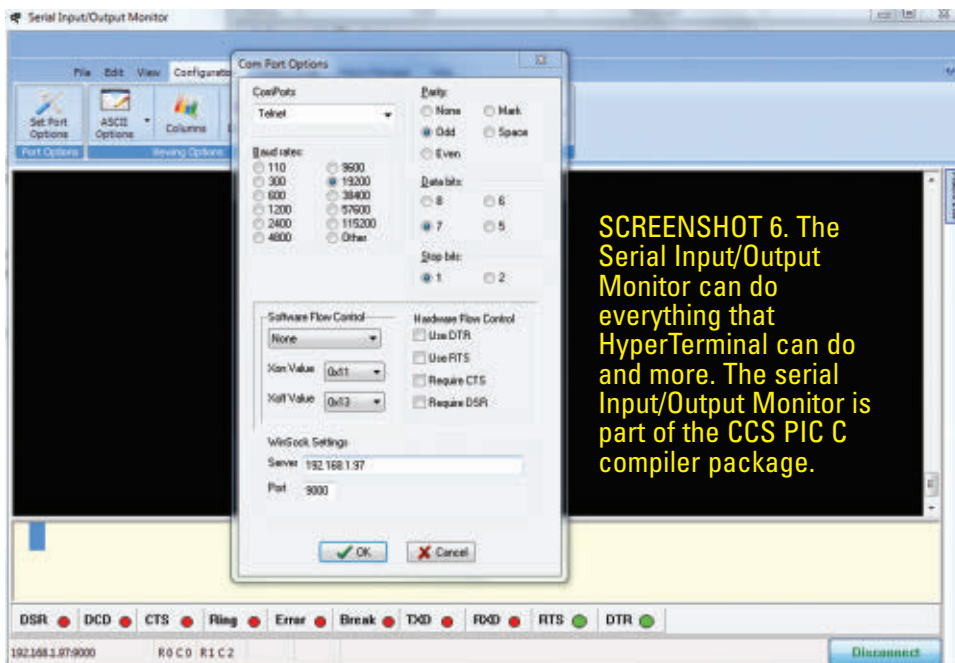
SCREENSHOT 5. This particular router calls the NEMO10 a virtual server. You should recognize the IP address and port numbers, considering we assigned them.



SCHEMATIC 1. This is very clever and cheap. This circuit eliminates the need for a more costly RS-232-to-TTL converter.

PHOTO 5. The PICDEM Z development board is very useful when you have to lash together a simple 802.15.4 network. The serial port plug-to-plug works because the NEMO10 is configured as a DTE and the PICDEM Z is wired as DCE.





I also have an illuminated Ethernet link LED and a glowing System Ready LED. After establishing the connection — in addition to the LEDs I just listed — I see a blinking Ethernet Activity LED and **Screenshot 7**.

The TCP connection will stay up as long as the PIC18F4620 is sending

data. Recall that we set the inactivity timeout for five seconds. If the PIC stops sending on its serial port for more than five seconds, the TCP session will be terminated by the NEMO10. To test this, I stopped the PIC and five seconds later the Serial In/Output Monitor DISCONNECT

button returned to the CONNECT state, and all of the status LEDs indicated a dormant interface. I then clicked on the Serial Input/Output Monitor CONNECT button without restarting the PIC18F4620. Sure enough, five seconds later the NEMO10 dropped the session.

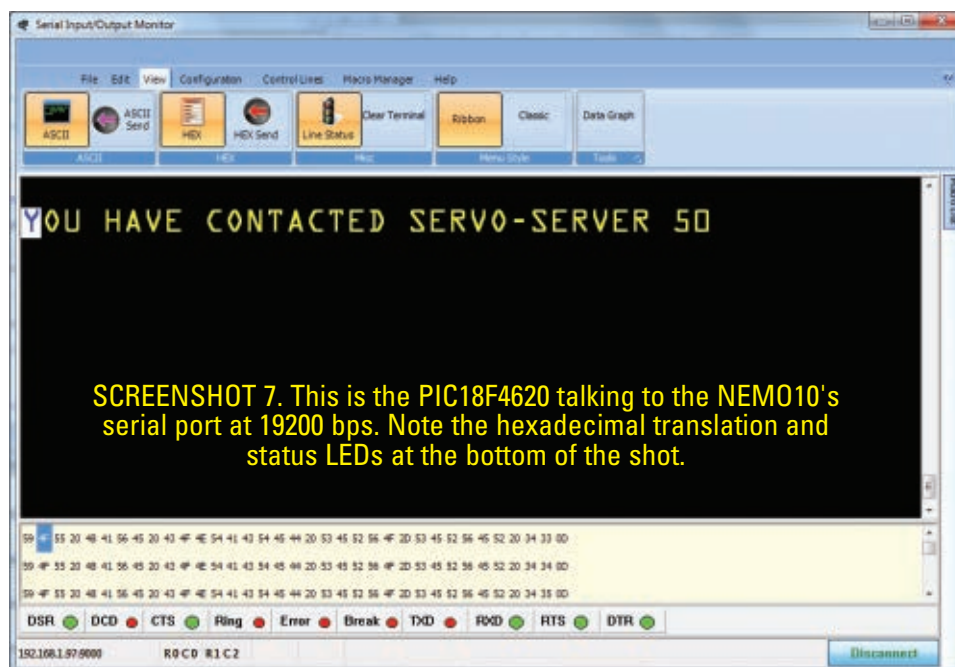
Serve a Meter

I just happen to have a Tenma 72-7750 modern digital multimeter on the bench. This little orange drop can do capacitance, frequency, and temperature measurements, as well as the standard voltage and amperage measurements. However, before we can tap into its features using the NEMO10, there's a little bit of work we have to do.

The first task is to understand the Tenma's serial interface which isn't really a regulation RS-232 serial interface at all. I've outlined the Tenma's "RS-232" interface in **Schematic 1**. The opto-isolator consists of a light transmitting

element (IR LED) internal to the meter and a light receiving module (IR transistor) terminated with a female DB9 connector that plugs into the meter. The circuit works by allowing the NEMO10's DTR line to drive the collector of the opto-isolator's transistor logically high (1). The NEMO10's RTS line provides a virtual ground return which is really a logical low (0) perpetuated by driving the RTS line to a logically low level. When the modulated IR light source is illuminated, the opto-isolator's transistor conducts and a logical high is presented to the NEMO10's RXD pin.

Conversely, when the IR LED is extinguished, a logical low is presented to the NEMO10's RXD pin. The resistor between the opto-isolator's emitter and virtual



ground makes the logic shifts possible.

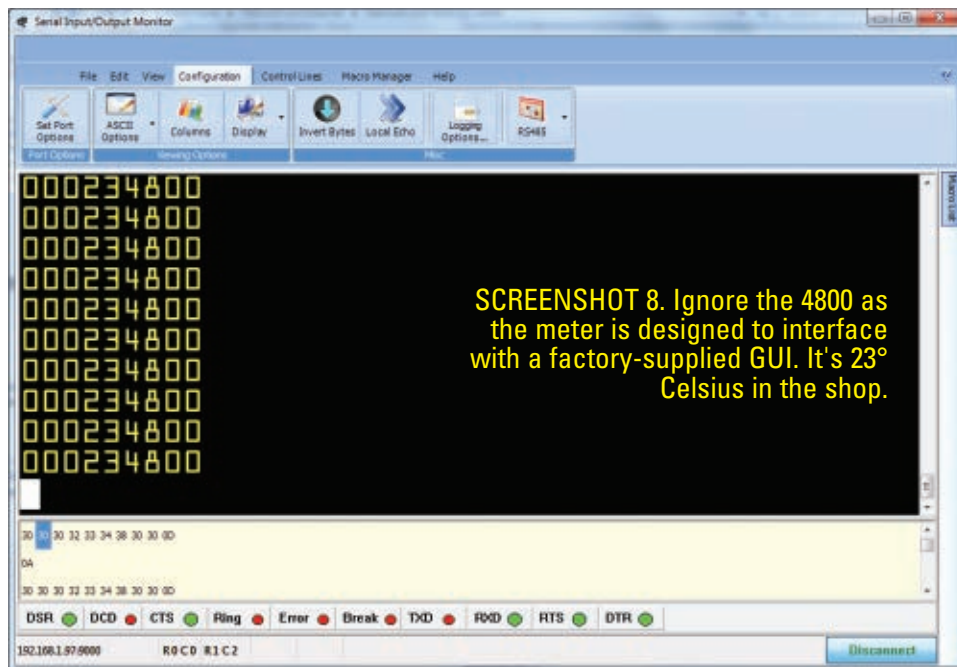
Recall that we can configure the NEMO10's DTR line to remain logically high at all times. The problem is that we can't programmatically control the NEMO10's RTS line. Being a DTE device, the NEMO10 will drive the RTS line logically high by default. So, to get the right logic level at its RTS pin, we simply isolate the electrical connection between the NEMO10 and the RS-232 converter input.

Removing the electrical connection forces the RTS input pin to float. The pin really isn't floating since there is an internal pull-up resistor on the RTS input. Since the translator gates are all inverters, a logical high on the RS-232-to-TTL converter's RTS input pin produces a logical low at its output pin. I performed the isolation by building up a couple of 12-pin male-to-female headers, plugging the NEMO10 into the new set of headers, and clipping off the new header RTS pin.

The use of RS-232 logic levels allows the circuit depicted in **Schematic 1** to replace a more expensive RS-232 converter IC. By the way, the Tenma runs at 19200-7-O-None-1 (19200 bps, seven data bits, odd parity, no flow control, one stop bit). According to **Screenshot 8**, it's 23° Celsius in the shop.

RS-232 is Far From Dead

The only soldering we performed was to assemble the 12-pin headers. The only code we wrote was for test purposes and didn't include any TCP/IP stack elements. Using the NEMO10, we managed to serve up some ASCII data from a PIC18F4620 and some temperature data from an off-the-shelf multimeter. Now that you know what a serial device server is and how to use it, apply what you have learned to that robotic device you're building. **SV**



<h3>Sources</h3> <p>Lemos International Co., Inc. NEMO10 www.lemosint.com</p>	<p>Custom Computer Services Serial Input/Output Monitor CCS PIC C Compiler www.ccsinfo.com</p> <p>Microchip PICDEM Z www.microchip.com</p>
--	--

RF Specialists

RF Modules

From Part 15 to Part 90 Compliant
Narrow Band FM, UHF Multi-Channel



Data Loggers

Stand Alone and
Wireless Mesh Networking Logger



Industrial Bluetooth

OEM, Modules, Wireless Device Servers, RS-232
Long range options, low cost



ZigBee Pro

OEM Modules and USB ZigBee Sticks,
Mesh Networks



Precision Quartz Oscillators

Excellent Phase Noise Performance

OCXO, TCXO, VCXO and PLL designs - 1MHz to 2.4GHz and GPS timing modules



RF Design Services

Prepared to work with your in-house engineers,
or support your RF project from initial design to implementation.

Industrial • Military • Space • Medical • Smart Grid Metering • SCADA • Lighting Control



LEMOS INTERNATIONAL

www.lemosint.com
866.345.3667
sales@lemosint.com

"Making your RF ideas into profitable products."

TROUBLESHOOTING TIPS and TRICKS



HOW TO KEEP THINGS FROM GOING ~~WRONG~~ WRONG WITH YOUR ARDUINO-BASED BOT

by Gordon McComb

www.servomagazine.com/index.php?/magazine/article/october2012_McComb

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

"Where nothing can possibly go wrong!" That was the catch phrase to the 1973 movie, *Westworld*. You remember what happens ... some programming glitch causes all the human-like robots in the Westworld theme park to go haywire. A particularly nasty gunslinger bot goes after the two heroes of the movie, shooting one dead (Josh Brolin's dad no less!) and then chasing after the other until its face gets burned off with acid.

I never give my robots six-shooters for this very reason. I know sooner or later something won't work the way it should. It's bad enough to have to troubleshoot problems with batteries, let alone worry if my robot will shoot me in the back one dark night.

Troubleshooting is the step-by-step process of determining the root cause of a particular problem, whether it's electronic or mechanical. It's one of the core arts and sciences of robot building. In this article, I'll provide some basic advice on how to troubleshoot your Arduino-based bots — gun fighter or otherwise.

Systematic Approach to Development and Troubleshooting

Simple bots only need simple troubleshooting. They only have one or two features, and when

something's wrong, the reason is usually obvious. Add complexity to the design — even something as innocuous as a separate set of batteries for the servo motors — and you're left wondering where to start in troubleshooting any problems. Layer upon layer of features and functionality create ever-increasing combinations of trouble hot spots.

As a system gets larger, working out the kinks becomes much more difficult. One of the best ways to test these kinds of systems is to break them down into smaller and more manageable units.

- Avoid the shock of a non-functional robot by building it in smaller pieces. Begin with the basic drivetrain — connect just the motors and battery to the Arduino (through a suitable H-bridge or other suitable circuit, as needed).

- If this part works, you're ready to move on by building and testing each new capability separately. Add

Start With a Preflight Check

Ever heard the phrase “kicking the tires?” In the early days of the automobile, tires were made of a much thinner rubber than they are now. “Kicking” them before setting off on a trip would help a motorist know if the tires were ready for the road. Better to catch problems early. You can “kick the tires” of your Arduino robot by going through a kind of preflight check list that demonstrates all systems are ready.

Start with a visual inspection of the Arduino itself, its battery or other power connection (like the type in **Figure 1**), and all wiring. Note if anything’s become disconnected or loose. If you’re programming the Arduino or powering it from your PC’s USB port, be sure the cable is snugly inserted on both ends. USB cables and their jacks can become damaged during normal use — wires kink and plug contacts get bent or broken off.

If you suspect a problem has developed but aren’t sure of the cause or how extensive it is, then upload a “signs-of-life” sketch to the Arduino to test basic operation. A good signs-of-life demonstration is the LED blink example sketch, or load up something like the one in **Listing 1**.

Upload it to the Arduino, then watch for the tell-tale flashing of the LED on pin 13 of the Arduino. (Naturally, you can’t use this method if your Arduino has something plugged into pin 13 that otherwise might alter the behavior of the LED. An example is a switch input where you’ve added your own pull-up or pull-down resistor. These external components will override the operation of the Arduino’s built-in LED, causing it to either stay on all the time or not flash at all.)

If you have a shield plugged into your Arduino, you may wish to temporarily disconnect it before running any signs-of-life sketches. Removing the shield disconnects any electrical components that might be interfering with the operation of the Arduino. Parts on a shield can go bad or short out, and those faults can prevent the Arduino from powering up or working correctly.

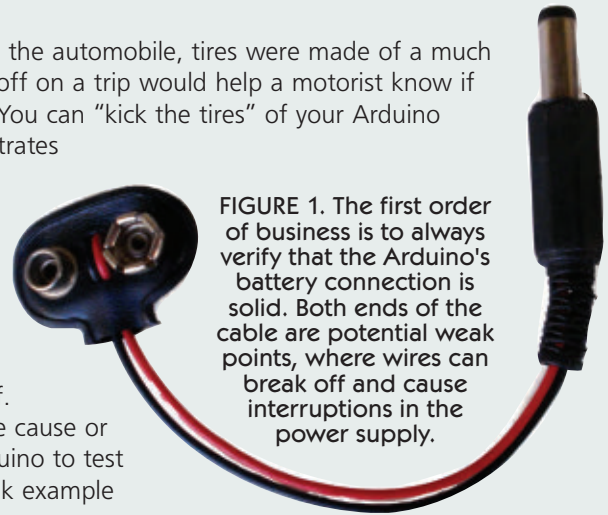


FIGURE 1. The first order of business is to always verify that the Arduino’s battery connection is solid. Both ends of the cable are potential weak points, where wires can break off and cause interruptions in the power supply.

Listing 1 — Blinker.ino

```
boolean ledState = false;

void setup() {
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, ledState);
  ledState = !ledState;
  delay(500);
}
```

just one feature or capability at a time. Avoid the temptation to implement ultrasonic range finding, infrared detection, and music playing all at the same time. Each of these is a discrete unit; test each one to determine if that function is working properly.

- After building and testing the core blocks of your robot, methodically integrate them with the rest of your system. Start with those that are critical to the design of your robot, plus have the most impact on the operation of the Arduino. This allows you to more readily isolate problem areas up front. For example, music and sound playing can consume considerable processing time. Integrate it early with your robot’s basic motor function. Add the other features (one at a time), like the ultrasonic and infrared sensors. By introducing the most demanding functions first,

you can readily determine which additional function “breaks the camel’s back,” so to speak. You can then decide if another programming approach is preferred, or which features to keep.

The same systematic techniques may be used to troubleshoot a fractious bot. Test each subsystem separately; ideally, electrically isolated from the rest of the components. That may mean partially disassembling your robot and disconnecting parts you don’t need. Circuitry on the music playing shield may be interfering with some other piece on your robot; remove the shield completely, don’t just rely on commenting out the music

playing code. Similarly, sensors and other components individually plugged into your robot’s Arduino should all be removed, then replaced one at a time for testing.

Or, circuits like cellular phone boards may consume copious amounts of current which can cause constant or intermittent errors. In the case of a cell phone SMS shield, for example, the circuit may draw an amp or two when transmitting. If your robot’s power supply can’t handle the peak load, the sudden draw can cause the Arduino to momentarily shut down.

These so-called brownout events can be hard to track down, because the problem may only rear its ugly head when other current-consuming circuits are added to the mix. Remove any single circuit, and your bot functions again. Add it back in, and the problem returns. On the surface, this kind of mayhem

Tip! Apart from easily resolvable problems like loose wiring or worn out batteries, the interaction of the specific combination of circuitry on your robot is the main cause behind technical glitches. Some hardware may not be widely compatible with others, for example.

can look like a fault in the individual circuits when it's really just a matter of overtaxing the battery. In cases like these, using a bigger battery or completely eliminating circuits that draw excessive currents usually solves this problem.

Using the Serial Monitor to Debug Problems

Get into the habit of validating the operation of your robot and its Arduino processor by using the Serial Monitor window. This tool allows you to send messages from the Arduino back to your PC. These messages serve as a way to quickly and efficiently debug many aspects of your robot/Arduino — all at the same time. While you must take a few moments to set the debugging environment, it can save considerable time in the long run.

No doubt you've already used the

```
boolean ledState = false;

void setup() {
  Serial.begin(9600); //Match in Serial Monitor
  delay(250);
  Serial.println("Setup complete!");
}

void loop() {
  digitalWrite(13, ledState); // LED on or off
  Serial.print("LED on: ");   // LED state in
                               // Serial Monitor
  Serial.println(ledState);    // Shows 0 or 1
  ledState = !ledState;        // Toggle LED state
  delay(500);                  // Wait 1/2 second
}
```

Listing 2 – Debugger.ino

Serial Monitor to display messages from the Arduino, so you already know the basic concepts. Just in case the idea is new to you, here's what's involved:

1. The hardware serial port is configured in the `setup()` function. Merely add:

```
Serial.begin(9600);
```

some place within `setup()`, preferably

near the beginning. The 9600 value is the baud rate — the data rate that the Arduino and PC will communicate with one another. This value is actually rather low, and both your computer and Arduino are capable of talking to one another at much faster speeds. However, 9600 baud is the most common rate used in the various Arduino examples.

2. Debug messages are sent from the Arduino to the PC by using one or

Good places to insert a *Serial.print* statement include:

Inside the `setup()` function to verify that the Arduino is progressing to at least that point in its program. The *Serial.print* statement must appear after *Serial.begin*. If printing text immediately after *Serial.begin*, consider adding a quarter to half second delay:

```
Serial.begin(9600);
delay(500); // Wait half a second
Serial.println("Setup!");
```

Inserting a debug statement inside `setup()` also helps you to verify that the Arduino runs the setup code just once. If you see the "Setup!" message repeat, it could mean your Arduino is being reset over and over again. This can occur, for instance, if the Arduino is not getting enough voltage to operate reliably.

At the start and/or end of the `loop()` function to check that the Arduino is properly completing each loop. You can use static text, as in:

```
Serial.println("Looping!");
```

or set up a more elaborate system where you indicate the number of loops that have elapsed so far:

```
int counter = 0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  Serial.print("Loop: ");
  Serial.println(counter++);
  // rest of code
}
```

Be careful when using debug statements in the `loop()` to prevent over-running the communications between the Arduino and your PC. If the loop doesn't contain programming statements that require at least half a second to execute, (temporarily) slow things by inserting a delay at the bottom of the loop:

```
delay(500); // Wait half a second
           // before proceeding
```

Inside a user-defined function to verify that the function is being called. User-defined (your own) functions are a handy way to execute the same code whenever a certain event occurs. A good example is making the robot stop, back up, turn, and head off in a new direction whenever one of its touch sensor switches is activated. You can verify that the Arduino is reaching the user-defined function simply by adding a debugging statement as the first program line within it:

more *Serial.print* statements, like this:

```
Serial.print("ok");
```

which sends a simple "ok" to the PC. This statement sends just the text and no new line character. So, when you view the debugging information in the Serial Monitor window, all the text will appear on the same line. If you'd like it to show up on different lines, use the variation:

```
Serial.println("ok");
```

3. To actually see the debugging messages on your PC, you need to open the Serial Monitor window which is a feature that's built into the Arduino IDE software. After a sketch download is complete, just click on the Serial Monitor icon and the Serial Monitor window will appear. Verify that the baud rate setting in the lower corner of the window is the same as the *Serial.begin* statement used in your sketch. Otherwise, the Arduino

Arduino Serial Monitor Control Characters

The Arduino's Serial Monitor supports a couple of control characters — also called escape sequences — that can come in handy when you want to display several debugging values at once. All control characters begin with a \ (backslash). Probably the most useful control character is \t which inserts a tab in the print statement line:

```
Serial.print("Some text here");  
Serial.print("\t");  
Serial.println("Some more text");
```

You don't need to emplace the \t control character in its own *Serial.print* statement when using just literal strings like that shown above. You will need to if you want to mix literal strings plus numeric values from variables:

```
Serial.print("Sensors reading\t"); // Explanatory text plus tab  
Serial.println(mySensor, DEC); // Value from mySensor variable
```

or

```
Serial.print("Sensor 1 is: "); // Explanatory text  
Serial.print(Sensor1, DEC); // Value in sensor1 variable  
Serial.print("\t"); // Tab  
Serial.print("Sensor 2 is: "); // Repeat for sensor 2  
Serial.println(Sensor2, DEC);
```

and PC will not sync up with one another, and the Serial Monitor window will display either nothing or gibberish characters. Use *Serial*

Monitor debugging whenever you want to verify a part of your code is working as it should. **Listing 2** provides a short working example.

```
void rightBumperButton() {  
  Serial.println("Right bumper  
  activated!");  
  // rest of code  
}
```

At parts of the sketch that use loops where you want to be sure that program execution eventually continues beyond the loop. Suppose your robot stops and waits for a bumper switch to activate. The code is part of a loop, like so:

```
while (digitalRead(10)==0) {}
```

which means check the value of digital pin 10, and keep looping as long as the pin is low (0). All fine and good, but sometimes programming errors, wiring mistakes, and other gremlins can cause your program to simply stop at this loop and never continue. You can check that the loop eventually exits just by placing a debugging statement immediately after:

```
while (digitalRead(10)==0) {  
  Serial.println("While loop ended!");  
}
```

Whenever you need to preview a value such as the result of a sensor. Imagine you have an ultrasonic range finding sensor, but aren't sure if you're experiencing problems because the value from the sensor isn't what you're expecting it to be. You can visually check any numeric or boolean value from within the

sketch simply by sending that value to the Serial Monitor window. For instance:

```
int ping = getPing();  
Serial.print("Distance is: ");  
Serial.println (ping, DEC);  
// rest of code.
```

```
int getPing() {  
  // Code to read ultrasonic sensor  
}
```

In this abbreviated example, the code that does the actual reading from the ultrasonic sensor is contained in a user-defined function named *getPing*. You call that function with the line

```
int ping = getPing();
```

The numeric value that represents the reading from the sensor is contained in the *ping* variable which you can use in a debugging statement. Note the DEC qualifier — it ensures that the number held in the *ping* variable is expressed as a decimal value.

The Arduino supports other qualifiers, as well — such as HEX and BIN — which display the value as a hexadecimal (base 16) or binary number, respectively. Check the Arduino reference at the **arduino.cc** website for additional options. By far, you'll use the DEC qualifier the most.

Here are some of the most common problems that beset Arduino-powered bots, and the most common ways to fix things.

No Power, No Lights — It's As Primitive As Can Be

A completely non-functional robot strongly suggests a problem with the power supply. If you're having these kinds of problems, start first with your robot's brain — the Arduino. If the Arduino's power light is not on, the most likely reason is that it's not getting power. Start with these checks:

- *Weak or dead batteries.* Recharge or replace.
- *Possible bad USB connection if powered from your PC.* Check the connection to see if the cable is merely loose. Try a different cable if one is available.
- *Faulty power switch or wiring.* Small switches can go bad, especially if struck hard during a collision. Inspect and replace as needed. While looking, see if the wires from the battery have come loose. Check the power connection (barrel plug) as it goes to the Arduino.

Erratic Behavior

Your robot is working, it just doesn't work all the time, or it occasionally behaves in bizarre ways. The most common reasons are:

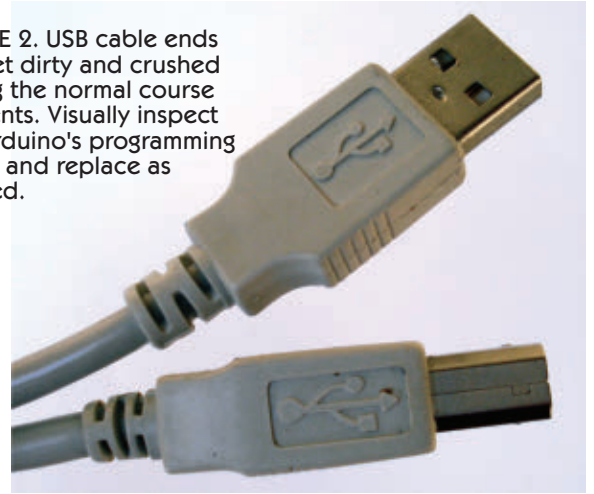
- *The battery charge is running low.* Replace, recharge, or increase the battery capacity (larger battery and/or higher voltage, as needed).
- *Bad wiring connection.* A broken or loose wire anywhere — especially ground related — can cause all manner of problems. If using solid conductor wire, check with your meter while gently moving the wire side to side. You'll get intermittent continuity if the wire is broken inside the insulation.
- *Wet or dirty contacts.* Water and dirt can cause intermittent connections. Wipe clean, dry, and try again.

Lights On, No Serial Debug Messages

Suppose the Arduino's power LED is glowing, but nothing seems to happen. What's more, you've used debugging messages in your sketch, and these messages are not appearing when you open the Serial Monitor window.

- *The USB cable is bad, or the wrong serial port is selected.* This error can occur if you've previously uploaded your sketch and have reconnected the Arduino to your computer. Verify the integrity of the cable (see **Figure 2**; check the ends themselves) and make sure the proper serial port is selected. (If you have more than one Arduino, each one may be associated with a different serial port. Be sure you've picked the right one.)

FIGURE 2. USB cable ends can get dirty and crushed during the normal course of events. Visually inspect the Arduino's programming cable, and replace as needed.



- *Wrong baud rate selected in the Serial Monitor window.* Be sure it matches the baud rate specified in your sketch.
- *Serial communications out of sync.* Sometimes your Arduino and PC fall out of sync, and they will not re-sync unless you: A) reset the Arduino, and/or B) reset the Arduino IDE (close, then reopen it).

Serial Window Displays Garbage

When you open the Serial Monitor window to look at the debugging messages sent from the Arduino, you get garbage text instead.

- *Wrong baud rate selected in the Serial Monitor window.* Be sure it matches the baud rate specified in your sketch.
- *The USB cable is bad.* Verify the integrity of the cable, or try a different one.

Debugging Messages Repeat When They're Not Supposed To

You've added a debugging message to the setup() function of your sketch. It should display once in the Serial Monitor window, but instead it keeps repeating.

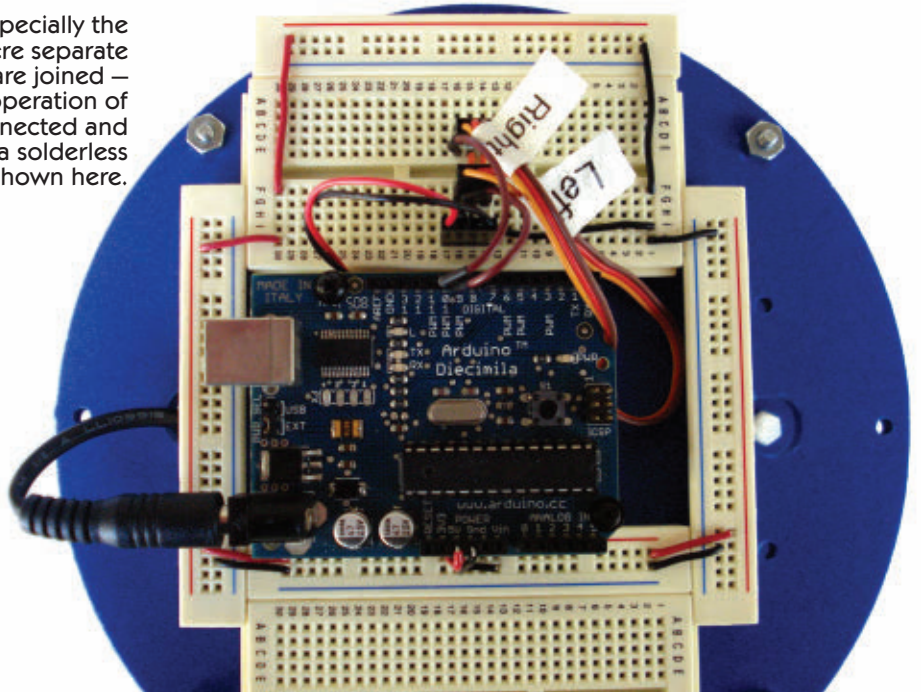
- *Arduino is resetting by itself.* Look for causes of power brownout — the voltage to the Arduino is falling below the minimum required. When this occurs, the sketch spontaneously restarts.
- *Arduino is resetting via the serial port.* On most versions of the Arduino, plugging or unplugging the USB cable causes the serial port to reset which, in turn, causes the sketch to reload from the beginning. Look for causes of spontaneous reset of the serial port which can occur, for example, if the cable or plug connection is bad, or if the Arduino is drawing too much current from your PC's USB port.

Data Results Are Wrong/Unexpected

You're getting debugging messages, plus the data shown is wrong or unexpected.

- *The sensor or other device is misconfigured.* If you're checking the results of a sensor, make sure it's

FIGURE 3. Loose or missing wiring — especially the common ground connections where separate electronics and motor power supplies are joined — can create intermittent errors in the operation of your bot. Be sure all wires are firmly connected and properly seated, especially when using a solderless breadboard as shown here.



properly configured and programmed. For example, an ultrasonic range finder sensor may be programmed to return distance in inches or centimeters (or raw microsecond delay of the echoes). The data will appear wrong if you're expecting it in a different format.

- *Sensor or other data in the wrong format.* The `Serial.print` statement can use an optional data formatting argument, specifying if the data is to be shown in DECimal, BINary, HEXadecimal, or some other format. Be sure to use the proper formatting argument based on the type of data value.
- *Programming error.* Though we try to avoid bugs in code, sometimes they creep in and muddle the results. This is what debugging is for — catching those mistakes so they can be fixed. Carefully analyze your programming code, and look for math mistakes, incorrect assumptions, and other errors that can cause the invalid data. For example, did you really mean to add those two numbers together or multiply them?

Sketch (Program) Won't Compile

The most annoying glitch when using the Arduino is some type of error that prevents the sketch from compiling and uploading. Though there are many reasons this can occur — including an improperly installed IDE software package — following are some of the most common reasons.

- *Syntax error in code.* Check for proper spelling and capitalization of all Arduino keywords ... remember, it's `digitalWrite`, not `DigitalWrite`, `digitalwrite`, `digitalRight`, or some other variation. Also check for missing semi-colons, parentheses, braces, and other typical errors.
- *Incompatible version of IDE software.* The Arduino IDE software has gone through many changes over the years, where some changes have "broken" existing sketch code. Be sure you're using the correct IDE version for your sketch. Unless there is a specific reason otherwise, it's best to keep your Arduino IDE up to date with the latest stable release. However,

you may keep previous iterations installed on your computer, should you need to compile your sketch with an earlier version.

- *Missing or incorrect library.* Sketches that must be compiled with one or more libraries require those libraries in specific locations — either the main *libraries* folder in the Arduino IDE program directory, or the *libraries* folder within your sketchbook directory. If installing a new library, be sure to exit and restart the IDE before attempting to use it.

Everything Works Except The Motors

Your Arduino appears to be functioning normally, but your robot's motors aren't turning.

- *Not enough current to power the motors.* Motors can draw a lot of current from your robot's batteries. If the batteries don't deliver enough current, the motors may not turn or they may only turn slowly. Beef up the batteries to ensure proper motor current levels.
- *Separate motor power disconnected, discharged.* If your robot uses a separate power source for the Arduino and its motors, the motor power may have become disconnected or the motor batteries may be discharged.
- *Shared ground disconnected.* Robots that use separate battery supplies for the motors need to have the ground of the two power sources connected. (There are exceptions to this, such as when the control board for the motor is interfaced to the Arduino using an opto-isolator.) Otherwise, the robot may not operate reliably, or at all. So double-check the common ground connection (refer to **Figure 3**) between the two power sources. **SV**

The ASABE Student Robotics Competition

by R. Steven Rainwater

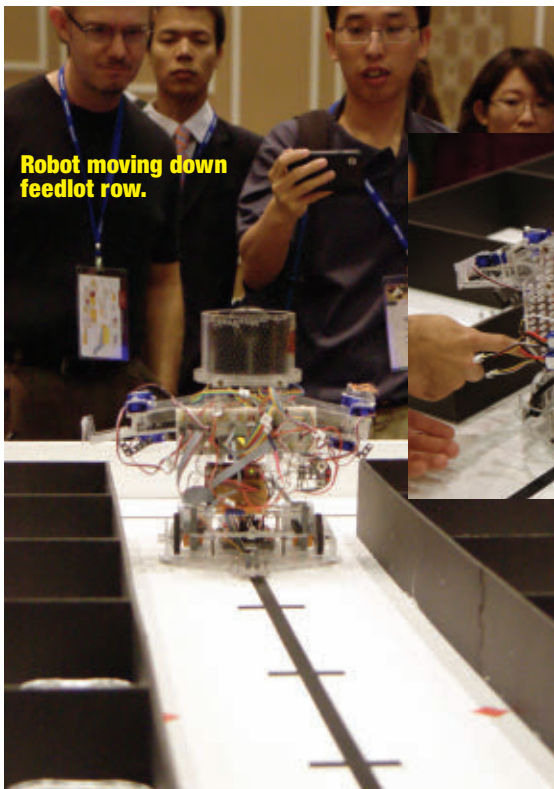


The American Society of Agricultural & Biological Engineers (ASABE) is an educational and scientific organization that promotes the advancement of engineering and science. Among many other activities, ASABE sponsors an annual robotics competition for graduate and undergraduate university students.

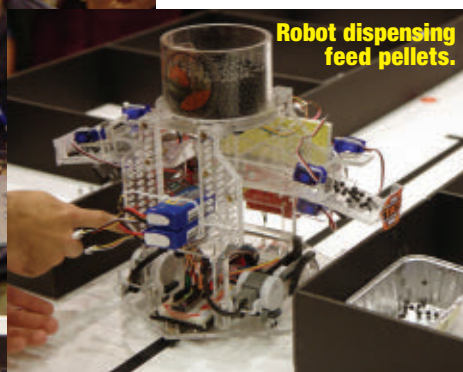
The ASABE Student Robotics Competition presents students with a different agriculturally-themed contest each year. For the 2012 contest, teams of students designed and built autonomous robots that were required to navigate a scale model of a cattle feedlot. The robots had to move along rows of cattle pens divided by fences. Each pen had a feed container. Feed was represented by a cargo of 6 mm Airsoft pellets which was carried by the robots. At the start of each run, a judge provided the contestant with an SD card containing the required weight of feed to be dropped into each pen's feed container by the robot. Each run was timed.

When a run was completed, the feed containers from each pen were weighed. Teams were awarded points based on objective measures such as the speed and accuracy of their robot, and for subjective qualities such as the elegance of design. Teams lost points when their robot struck fences surrounding the pens, required human assistance to complete a run, or dropped feed outside of a feed container. At first glance, this contest looks similar to a lot of other student robot contests but the addition of the feed pellets added some interesting and fun dynamics. When a robot missed the feed container and dumped a load of pellets onto the ground, it didn't just lose points; it created an obstacle for itself as it had to navigate through the rolling, bouncing flood of pellets to get to the next pen.

The contest is held every year at the ASABE International Meeting. If you get a chance, it's definitely worth checking out. **SV**



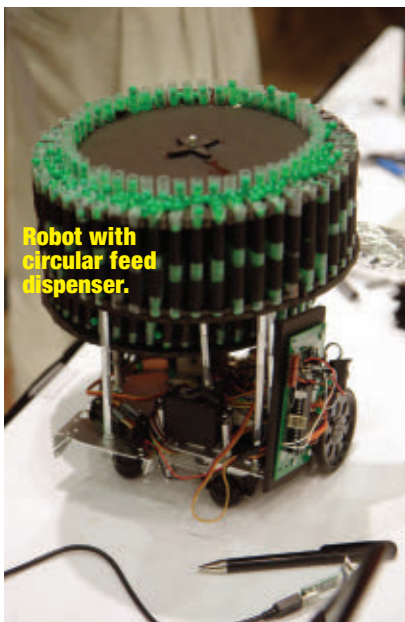
Robot moving down feedlot row.



Robot dispensing feed pellets.

You can read more about the ASABE Student Robotics Competition at <http://abe-research.illinois.edu/ASABERobotics>

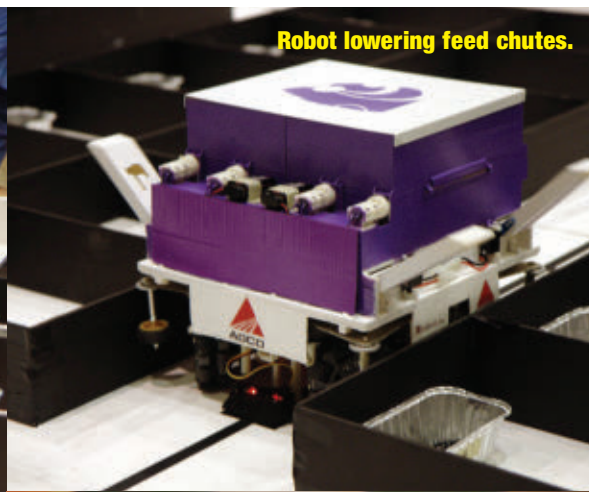
You can find out more about the ASABE organization at www.asabe.org



Robot with circular feed dispenser.



Robot dispensing feed pellets.



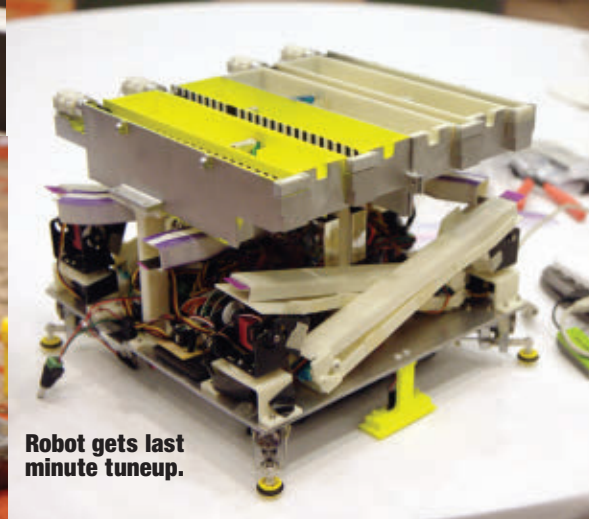
Robot lowering feed chutes.



Two robots working as team.



Robot being tested before run.



Robot gets last minute tuneup.

See a full gallery of photos from the 2012 contest at <http://flickr.com/steevithak/sets/72157631289859604>



Judge weighing contestant's feed bags.



Judges evaluating team's feed containers.



Robot begins run down feed lot row.

The *SERVO* Webstore

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

CD-ROM SPECIALS



Free Shipping!

8 CD-ROMs & Hat Special
Only \$ 169.95 or \$24.95 each.
www.servomagazine.com

NEW RELEASE



ONLY \$24.95!

BRILLIANT LED PROJECTS
20 ELECTRONIC DESIGNS FOR ARTISTS, HOBBYISTS, AND EXPERIMENTERS
NICK DOSSIS

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

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.



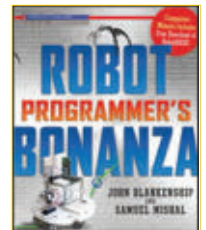
\$24.95*

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!**

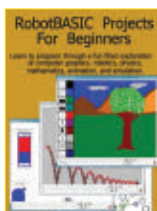
Now you can get one year's worth of all your favorite articles from *SERVO Magazine* in a convenient bundle of print copies. Available for years 04, 05, 06, 07, 08, 09, 10 and 2011.

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

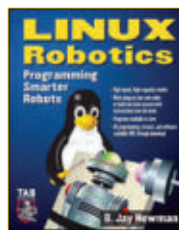


Linux Robotics

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

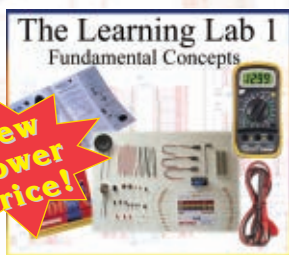
by Alan Overby

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.

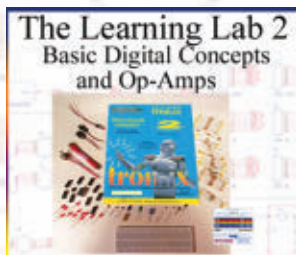
\$34.95



FOR BEGINNER BOT BUILDERS



\$59.95



\$49.95



\$39.95

The labs in this series — from GSS Tech Ed — show simple and interesting experiments and lessons, all done on a solderless circuit board. As you do each experiment, you learn how basic components work in a circuit, and continue to build your arsenal of knowledge with each successive experiment.

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

Or order online
www.servomagazine.com

SPECIAL OFFERS

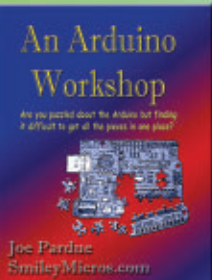
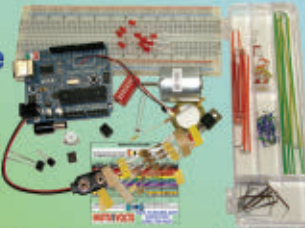
Puzzled by the Arduino?

Based on *Nuts & Volts* Smiley's Workshop, this set gives you all the pieces you need!

Book and Kit Combo \$124.95

Kit 84.95

For more info on this and other great combos! Please visit: <http://store.nutsvolts.com>

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



Beginner's Guide Vol 3 Combo

Combo Price \$139.95

For complete details, visit our webstore @ www.servomagazine.com.



"THE PHANTOM DRAW"

The KILL A WATT meter is the best way to help you determine your actual energy draw in ON and OFF home appliances.

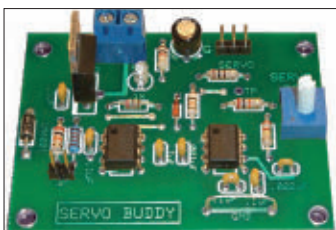
Only \$29.95

To order call 1 800 783-4624 or online www.servomagazine.com



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.

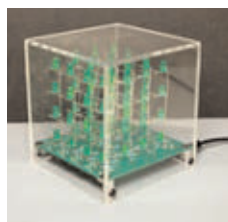
Includes an article reprint.

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*.

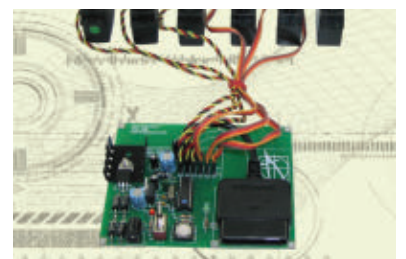


This kit shows you how to build a really cool 3D cube with a 4 x 4 x 4 monochromatic LED matrix which has a total of 64 LEDs. The preprogrammed microcontroller that includes 29 patterns that will automatically play with a runtime of approximately 6-1/2 minutes. Colors available: Green, Red, Yellow &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

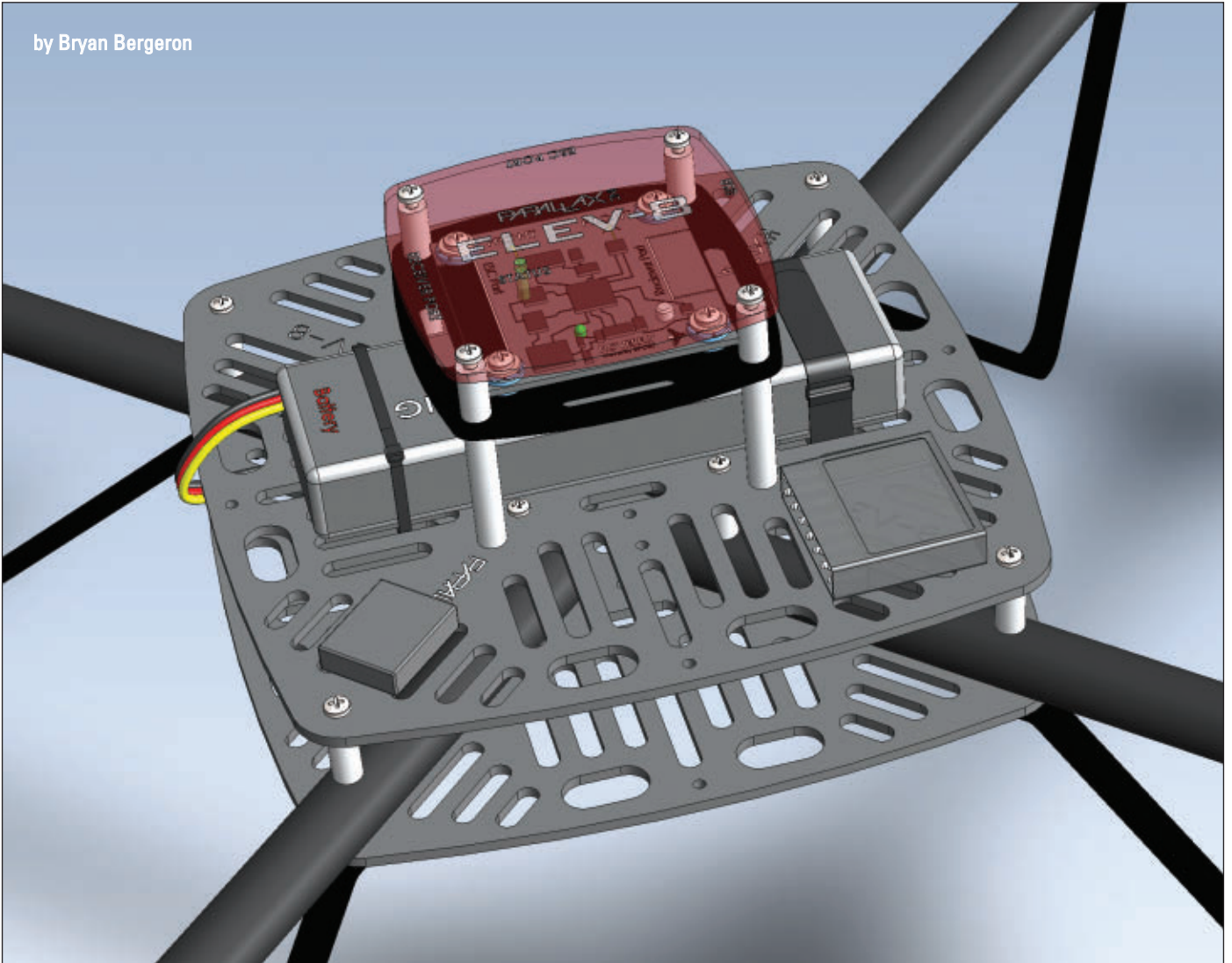
\$79.95

Non-Subscriber's Price

\$84.95

Parallax Elev-8 Quadcopter — Part 2: The Electronics Setup

by Bryan Bergeron



www.servomagazine.com/index.php?/magazine/article/october2012_Bergeron
Discuss this article in the *SERVO Magazine* forums at <http://forum.servomagazine.com>

This is Part 2 of a review on the Elev-8 Quadcopter robotics platform from Parallax. Whereas Part 1 focused on the physical build, this part walks you through the testing and setup, including integration with an R/C transmitter and receiver, selection and care of Li-Po batteries, and the never-ending task of maintenance.



Introduction

If you've successfully completed the tasks outlined in Part 1 you've built and prepared the basic platform, meaning you've balanced the props, tested and programmed the ESCs, checked the direction of motor rotation, installed the power harness, and verified that every nut and bolt is in its proper place. Now it's time to hook up and verify the operation of the onboard computer and the R/C receiver.

HoverFly Open Board Setup

The HoverFly Open Board onboard computer is what allows you to manipulate the motors with your R/C transmitter. For example, if you move the left lever of your R/C transmitter forward to increase throttle, the transmitter sends a signal to the Open Board which sends control signals to each ESC to increase the speed of rotation of each motor. You could simply assign one channel to each motor and – given enough practice – increase or decrease the speed of each motor simultaneously as needed.

However, there's more to the onboard computer than allowing you to control four motors with a single control stick. Separate manual control of each motor is virtually impossible when it comes to pitch (tilting and moving to the left or right), roll (tilting and moving front or back), and yaw (rotating clockwise or counterclockwise) –

especially when executed simultaneously in a modest wind. While it might be possible for someone to manually control all four motors while the craft performs a figure "8" in variable wind, it would probably take years of practice. They'd have no time to experiment with the robotics-specific aspects of the platform.

The Open Board takes over the details of differential motor speeds, enabling you to control the craft with a set of joysticks built into the transmitter – even in a modest breeze. It accomplishes the feat with the help of a three-axis gyroscope. Moreover, if you purchase the optional ultrasonic range finder (within the operational range of the craft), you can set and forget the hover height.

Photo 1 shows the HoverFly Board atop the Elev-8, wired for testing. The front of the board (which faces up in the **photo**) holds the ESC motor connector. On the left of the board are the connections to the receiver; in my case, the receiver that accompanies the Spektrum Dx6i transmitter. The USB connector, prop chip, buzzer, and LED status light are also visible in the **photo**. In this example, the purple light signifies there's a problem – I haven't finished connecting the ESCs to the board.

Setup involves first downloading the latest firmware from the HoverFly site. You'll need a PC running Windows to operate the program. Then, run the HoverFly utility to verify that your transmitter and receiver are properly configured. In all, software setup and verification is a 20 minute painless operation. The most difficult part is determining where to plug in which cable. The color-coded diagram on the HoverFly site indicating which channel from the Spektrum receiver plugged into the ports on the board wasn't immediately intuitive, but it eventually made sense. As I noted in Part 1, inserting and removing the

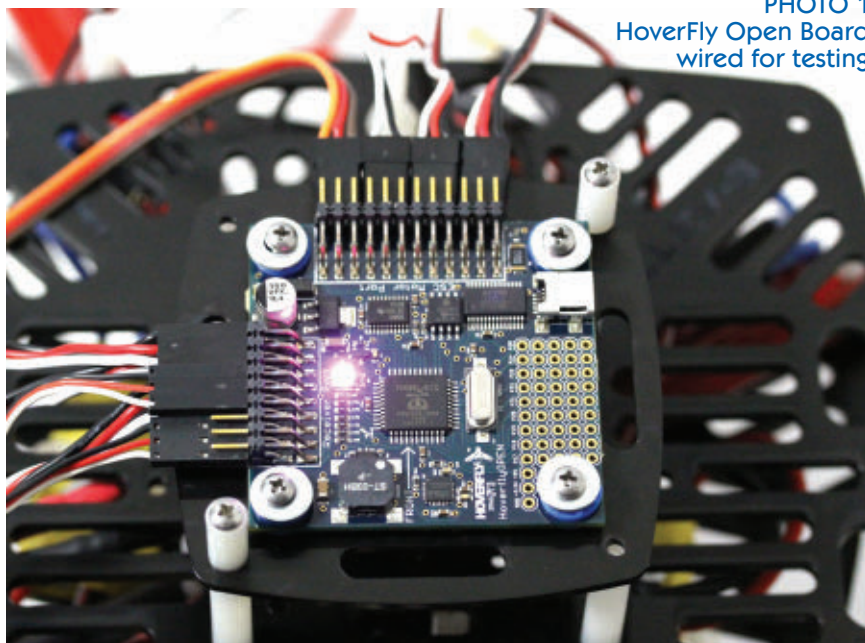
various cables from the ESCs and receiver stresses the four metal-on-plastic anchors of the board. I managed to strip two of the connections during this part of the testing. Use tie wraps to secure the board until you're ready to fly. If you happen to strip the board, use 4-40 plastic bolts and nuts to lightly secure it in place. They're lightweight and strong.

The translucent top shield (shown in the lead **photo**) is an important part of the overall design. Don't neglect it to save a few grams. If your copter lands head first in a field or if it starts to rain, you'll wish you had the shield in place.

Failsafe

One of the worst things that can happen during flight is to lose the connection between the handheld R/C transmitter and the onboard receiver.

PHOTO 1.
HoverFly Open Board,
wired for testing.





Whether due to interference, weak transmitter batteries, or accidentally dropping the transmitter onto hard asphalt, without a transmitted signal your quadcopter is out of control. To prevent this, you need to define the failsafe control signals that the receiver should emit when the transmitter signal is lost. Every R/C manufacturer has a way of addressing the failsafe issue.

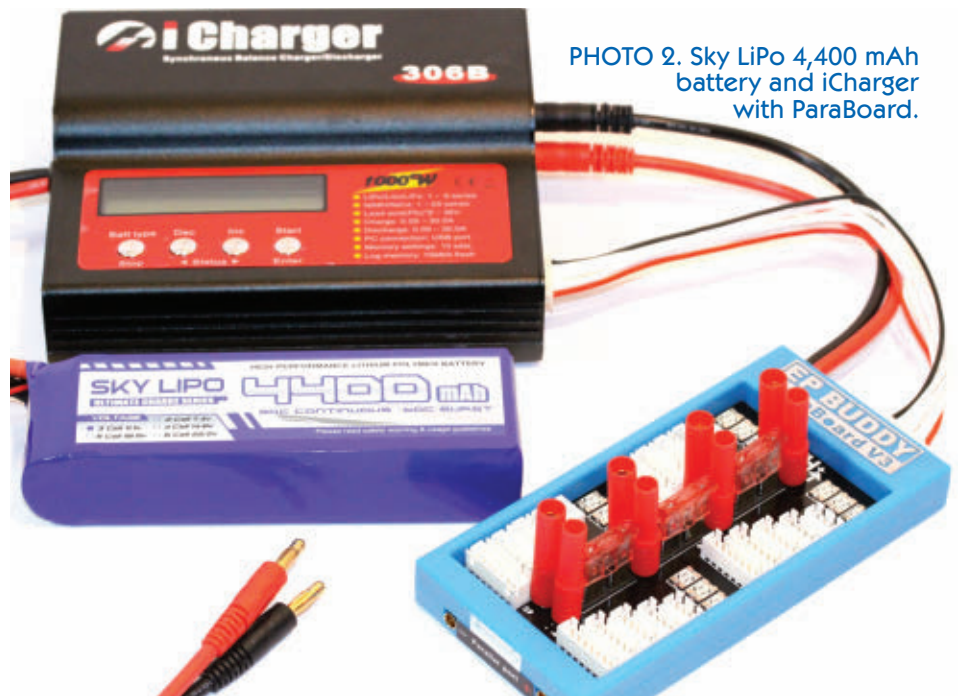
In the Spektrum Sx6i system, failsafe options include throttle to off, throttle to idle, and hold on last command (that is, if in a turn, keep turning). The best choice for you depends on your environment. I prefer throttle to 'idle,' which hopefully results in a controlled crash. Throttle to 'off' guarantees a hard crash, but minimizes chances that the copter will drift out of the flight area.

Power

For setup and testing, I used a laboratory grade switching power supply rated at 40V at 30A instead of a battery. The power supply enables me to monitor and precisely limit current during testing. I also don't like the idea of providing an untested quad with a self-contained power supply. I have the power supply cable tethered so that if the quad manages to move more than an inch above the testing table, the cable automatically disconnects and the quad comes to rest.

For battery power — based on recommendations from Parallax support — I selected a three-cell, 11.1V, 4,400 mAh Li-Po (see **Photo 2**). The battery is rated at 30C continuous and 60C burst. The C rate is the rate at which a battery is discharged relative to its maximum capacity. A 1C rate is equivalent to discharging the entire battery capacity — 4,400 mAh — in one hour; 30C means that the battery can discharge at a rate of $30 \times 4,400$ mAh, or 132A. Of course, the battery will be depleted in only two minutes. At the 60C burst level, the battery will deliver an amazing 264A. These are theoretical ratings — your mileage will vary. Still, you can see the need for short, low resistance connections between the battery and ESCs. Even a few tenths of an ohm can result in significant power loss. As a consequence

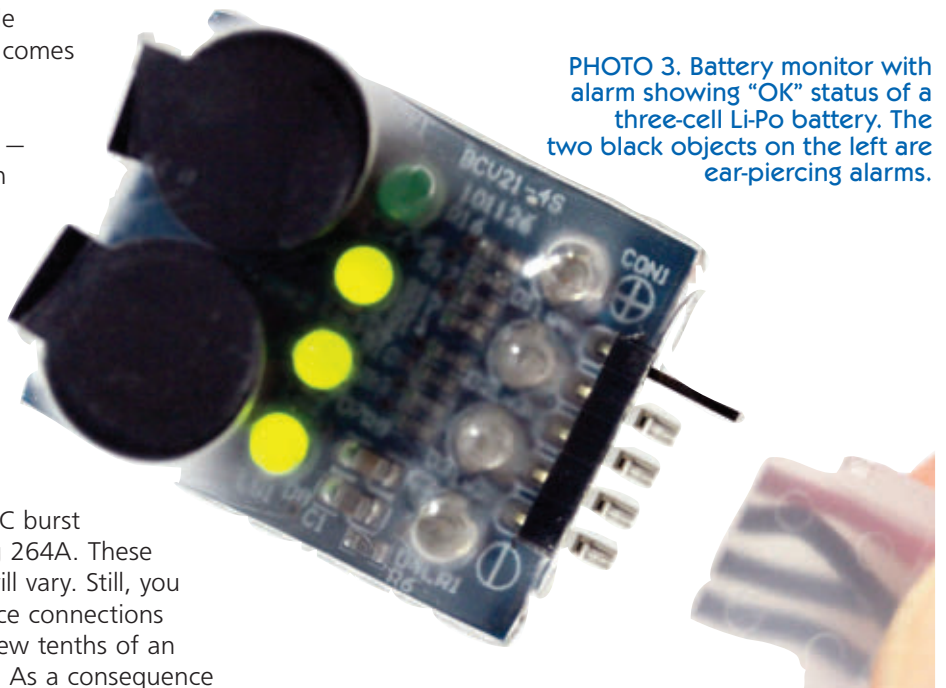
PHOTO 2. Sky LiPo 4,400 mAh battery and iCharger with ParaBoard.



of selecting Sky Li-Po over other comparable brands, I have to work with 4 mm bullet connectors.

The next issue related to power is care and feeding of your Li-Po batteries. You need a good charger and power supply. I've had good luck with the iCharger 306B and matching ParaBoard (refer again to **Photo 2**). The charger is on the expensive side (\$160), but with the ParaBoard (parallel charging board) I can charge four Li-Po batteries simultaneously. In other words, I can have a new set of

PHOTO 3. Battery monitor with alarm showing "OK" status of a three-cell Li-Po battery. The two black objects on the left are ear-piercing alarms.





batteries in about an hour instead of half a day of connecting and disconnecting batteries from a single-cell charger.

Another issue related to power is knowing when you're empty. Check out HobbyKing or Tower Hobbies for low voltage fingernail-sized alarm modules that sound off when battery voltage falls below a preset value. The best systems monitor the individual cells in the battery and sound an alarm when any of the three cells fall below critical voltage. **Photo 3** shows one such system, available from HobbyKing (\$3). Finally, although it might seem like a trivial matter, install plastic extenders on the Li-Po battery JST connectors, as in **Photo 4**. The extenders — which sell for about five cents each — will save your batteries from premature failure. Whenever you charge a Li-Po in a smart charger, you have to plug in the JST connector. Tugging on the wires to remove the plug is inevitable, and this will inevitably ruin the plug.

Take Pictures

At this point, take some pictures. If you've ever built and flown model rockets, planes, or copters, you know that the Elev-8 is going to show wear and tear, even if you manage to avoid an outright crash. Of course, your pictures should include more than simple camera shots. Include a “snapshot” of exactly where you're starting from, in terms of motor efficiency, battery output, and overall mechanical integrity.

Create a logbook and catalog the actual KV, maximum rpm, and power draw from each of the four ESC motor circuits. You can expect these values to decay with normal wear and tear and the inevitable crashes. Fortunately, there is a variety of affordable meters on the market, starting with dedicated KV and rpm meters (\$20, HobbyKing). For about twice that price, you can get a meter that also measures watts, battery internal resistance, and temperature, and that doubles as a servo tester. As you might expect, the 1,000 KV motors don't

provide anywhere near the theoretical 1,000 x 11.1 or 11,100 rpm at full throttle. Plus, the KV value will decrease as wear accumulates from improperly balanced propellers, impact, and the accumulation of dust and grit.

Li-Po batteries also deteriorate with time and use. Track the internal resistance and pull a battery from service as soon as you notice a significant deviation from normal. You don't want a cell to fail in flight.

Maintenance

One of the best features of the Elev-8 is ease of maintenance. Given time and a workspace, everything is field-replaceable. With the Parallax Crash Kit, you're covered for motor mounts, motor parts, and props. You could even carry a spare aluminum tube, pre-drilled to accept the 4-40 mounting hardware.

Post flight, you'll want to check the integrity of the top and bottom plates, and verify that the bolts are secure. If there's been a crash or one of the props grazed a bush or shrub, then remove the prop and check it with your balancing jig. **Photo 5** shows my balancing station with an experimental 10" x 4.5" pitch prop. The prop is heavy on the right, meaning I'll have to sand off a bit of the training edge on that side of it.

A related part of maintenance is checking the integrity of the four motors. As in **Photo 6**, make a habit of revving up your motors and looking at the collet at high speed. As in this **figure**, the outline is crisp with no evidence of wobbling. When the image of the collet starts to blur, it's time to rebalance the prop and/or rebuild the motor.

Preflight

As with a real copter, you should have and use a pre-flight checklist that includes battery check, prop collet check, transmitter battery check, and failsafe setting check. You'll also want to note the wind and weather forecast before you head out to the flight field.

PHOTO 4. JST connector without (left) and with (right) protective sleeve. Recommended inexpensive insurance against JST connector failure from unplugging the connector by the wires.



The Fun Part

Up to this point, it might seem like owning and maintaining a quadcopter is like owning a sailboat or an old house. You're always fixing something. However, that can be enjoyable. Of course, the real fun is experimenting with the craft. For example, on my to-do list is to work with the optional ultrasonic range finder and modify the flight computer so that instead of simply hovering at a given distance from the ground, the quadcopter takes



off and lands automatically. I'm also thinking of implementing a failsafe mode where the craft powers down to idle and then — as soon as the range finder registers an echo — hit the throttle to bring the craft down as softly as possible.

With multiple channels, lots of space, and an onboard carrying capacity of two pounds, you're limited by your budget and imagination. The GoPro HD Hero2 video camera (\$300) is especially popular among quadcopter enthusiasts, as are the inexpensive (\$40) dice mini cameras with night vision. I have my eye on a remote telemetry system that lets me monitor battery and motor conditions in real time. For now, I'm using an iPod with iChat for Wi-Fi based real time video.

Part of the fun of working with the Elev-8 is discovering what's possible. Spend some time on the R/C supply sites including HobbyKing, Tower Hobbies, and Xheli, and you'll no doubt discover technologies that you didn't know existed. If you have the patience to wait for overseas delivery, HobbyKing offers amazing discounts and a huge selection. If you need supplies tomorrow, then Tower Hobbies is worth considering. Of course, Parallax stocks spare motors and parts for the Elev-8.

Closing Thoughts

This fun and (in my opinion) easy to build kit probably shouldn't be your introduction to robotic R/C flight. Think about it. As a robotics enthusiast, you already need to know electronics, computers, and at least rudimentary mechanical engineering. Now, you've added another dimension — it's like having the scientists that developed the space shuttle suit up and take over the role of astronaut. It's possible, but not overnight.

You'll need to find a local R/C enthusiast to show you the ropes — from where it's safe to fly to how to judge wind speed and direction (there's an app for that). You might consider a short training tether of 1" diameter rope, with one end affixed to the bottom plate of the Elev-8 and the other free. Because the load increases as the craft rises, the craft can't shoot up and crash on to a rooftop before you manage to hit the power switch on your transmitter. Make sure the rope is dense enough to stay put with the full force of the motors pushing the Elev-8 upward. If you haven't touched an R/C transmitter in years, then go to Amazon and buy a Syma S107 R/C helicopter in the color of your choice for \$20. Fly figure 8s

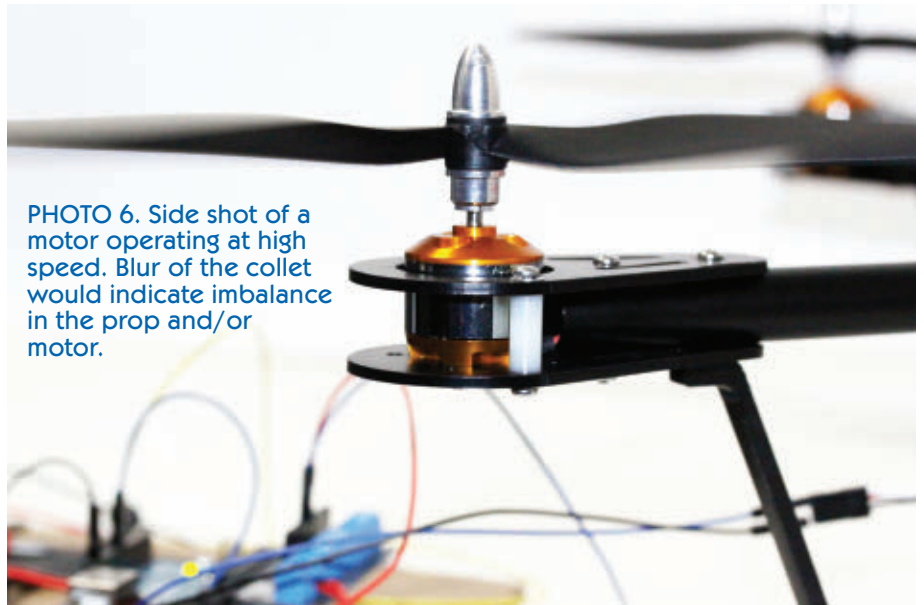


PHOTO 6. Side shot of a motor operating at high speed. Blur of the collet would indicate imbalance in the prop and/or motor.

in front of you, overhead, in front of active air conditioners, and at night with the lights out. Don't even think about powering up the Elev-8 until you can take off, fly, and land the Syma flawlessly.

Then, consider a simulator add-on for your R/C transmitter, such as the RealFlight R/C flight simulator. These simulator packages contain a standard cable for your R/C transmitter and software for your PC. Select the simplest single-engine helicopter model and start flying. The advantage of this system is that you get used to the same transmitter interface you'll be using on the Elev-8. The disadvantage is that there are no Elev-8 models to fly — you'll have to make do with a helicopter.

Still, you'll get a feel for the transmitter controls, and this newfound dexterity will apply directly to your control of the Elev-8. Happy flying! **SV**



PHOTO 5. Balancing props after a crash is mandatory. This balancing tool uses miniature bearings. More sensitive tools use magnetic suspension.



Then and NOW

The Many Ways to Control a Robot

b y T o m C a r r o l l

I had an interesting conversation with a friend about some of my recent columns, in particular, we talked about ways to control robots other than visual sensors such as the Kinect and verbal control via speech recognition. We discussed early robots that used crude wireless or wired control, as well as more modern concepts such as NASA's one ton Curiosity that is being controlled from Earth all the way to Mars — despite the 10 to 15 minute signal delays. These rovers require a lot of autonomy because of these signal delays, but remote control capability of some functions is also a must.

As designers and builders of our robotic creations, we still want some sort of control over our machines, even if the basic control is through autonomous sensor interaction with the outside world. Quite frankly, there are many ways that allow a human to control a robot outside of programming an autonomous bot. I won't be discussing internal robot control methods (such as a sensor controlling movements). I am going to concentrate on strictly 'hands-on' human control methods and steer away from total autonomy.

Robots have fascinated me since I was a kid and first read *I Robot* by Isaac Asimov. In June 1956, the Boy Scout magazine, *Boys' Life* had an article entitled *Gismo and I*, by Sherwood Fuehrer. Gismo the Great — shown in **Figure 1** — was human sized and was 'so neat' in my young mind. Fuehrer had used parts that he got from his dad, an engineer, and other scrounged parts. He used an old tool box as the control panel. It had no autonomy other than what switch its young inventor pushed or flipped. (You can see his 'control box' sitting on a stool.)

I collected a pile of parts to build my own 'Cosmo' that was mostly heavy plywood, furnace ducting, juke box parts and motors, a heavy tape recorder, and assorted light bulbs. I tried to make it move on some toy rubber tires and two juke box motors,



FIGURE 1. Original Gismo the Great from 1956 *Boys' Life*.

but the heavy beast fell over once and shook our whole house. (Tall robots don't stand for long when the builder uses single speed motors for the drive wheels.) After that, I just kept it stationary with a recorded voice and flashing lights. The controls for all the lights, motors, relays, and tape recorder were what interested me the most. It was nowhere as cool as Gismo in *Boys' Life*, but it kept me busy.

Years later in 1986, as a scout master and robotics engineer at Rockwell, I was asked by *Boys' Life* to design and write about a simple, but more modern robot that scouts could build on their own, and to keep the total cost below \$50. The editor at *Boys' Life* decided to name it 'Gismo2BL' as the "son of Gismo" from the earlier article. I used a typical plastic waste basket as the body,

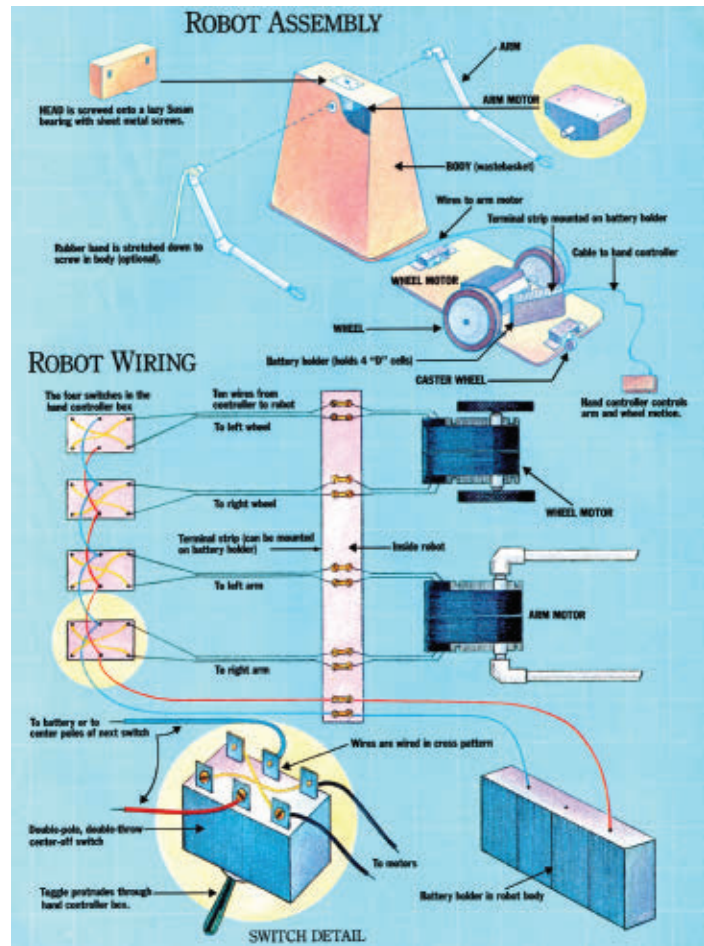
plastic pipe for the arms, and two dual 6 VDC motors to power the arms and wheels. **Figure 2** shows the basic layout of the robot and a simple schematic from the February 1987 article.

I had to avoid the use of a 68HC11 microcontroller (or similar microcontroller of that era) to keep the cost down and for simplicity. Control was by two sets of momentary DPDT switches from RadioShack for the two arms and the two wheels. I felt I should use flattened 'tin can' metal for the wheel mounts since few kids had the machine shop shears and metal stock that I had available in my garage. My twin sons, Jimmy and Tommy, who were scouts at the time were my test subjects to verify that a kid could make the robot from scratch. Herbach and Rademan — the surplus house in Philadelphia — was the source for the \$4.95 dual motors. They had to temporarily add more staff and actually manufacture more of the motors as 27,000 kids built or tried to build the robot.

Switches — the Most Basic Control System

Simple switch control is used in many toys and toy robots, as well as in most power tools, industrial machines, and appliances. I remember one toy Jeep that I had that used a two-button wired remote control: one button each for forward and reverse, with the front wheels mechanically steering to the right in reverse. To change direction, you had to back up as far as it took to face the new direction that you wanted to go; the wheels were straight for 'forward.' Crude, but it worked. It was just a simple on-off type of control. Add in polarity reversal for battery-powered tools and variable speed control, and you still have on-off switch control. To manipulate something in this manner from a remote location, one usually has to use wires from two to as many as hundreds, depending on the number of controlled functions.

FIGURE 2. Simple control schematic for *Boys' Life* 1987 Gismo2BL robot.



Robots with many functions can operate remarkably well with switch control. Add in variable speed and polarity reversal, and you can control almost anything. Pilots of small underwater ROVs (remotely-operated vehicles) using a cable to the surface can not only control various thrusters, lights, and robotic arms, but they literally control all parts and subsystems of the vessel. A variable speed multi-axis joystick works the best for a robotic arm, though. With TV feedback, pilots can literally feel as if they are right there in the water.

This is just one example of how the simplest of control systems can apply to the most complex machine. Certainly, the more complex of these research robots have many computers and microcontroller systems, but switched controls are prevalent in everything from police and military

robots, boats, cars, and even aircraft and spacecraft.

Cutting the Cord With Remote Control

Many of us eventually want to have our robots autonomously find their way around a shop or house, but autonomy is not necessarily a function of all robots — despite what purists may say. When it comes to a human controlling a robot, the first thing any experimenter wants to do with a machine that has a cord for control and/or power is to sever that cord. I have seen other people's robot projects, and have tried many methods myself of 'cutting the cord' in robot control.

One of the most popular methods for any type of technical experimenter is to use methods and parts from toys

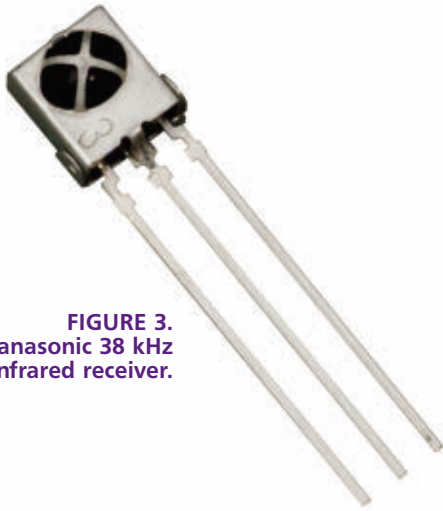


FIGURE 3.
Panasonic 38 kHz
infrared receiver.

— especially low cost walkie-talkies. They are cheap — sometimes too cheap — easily obtainable, and easily hackable.

Refining RF and IR Remote Control Methods

Cordless remote control can be accomplished via an RF (radio frequency) link, an IR (infrared) beam, or by an ultrasonic signal. The latter was used for remote control of TVs several decades ago, but lost out to today's IR remote controls. IR control works well for television and various audio systems that require only a single control function at a time.

I have seen so many ways that people have used to get controlling signals from a handheld device to their robot. Experimenters have used DTMF telephone key pads to send 10 or 12 different dual-tone 'codes' via a walkie-talkie to a walkie-talkie placed within the robot with a DTMF decoder attached. Unfortunately, most of these ended up as on-off control only.

One could have a button depressed that closed a circuit in the robot, as long as the button remained depressed. Or, it could be a toggle on/toggle off proposition with the button depressed once to start, and then again to stop. Surplus houses had IR decoders for TVs — not just the little receiver cubes — that were

hacked to receive an RF signal sent via a TV remote control that was also hacked to send an RF signal. There were so many versions that I saw in the '70s and '80s.

Many robot experimenters have used surplus handheld IR TV remote controls for their robots. Simple IR receivers such as the Panasonic module from Parallax (shown in **Figure 3**) are available for just a few dollars. The coded signals can be received, decoded by a microcontroller, and used to control many functions on a robot. Most use a 38 kHz carrier frequency and are easily converted to small robot controllers.

For example, the volume or DVD player arrows on a controller can be used for forward and reverse on a robot, and the remaining buttons can be used for controlling other functions. An experimenter can view the output of the remote on an inexpensive IR receiver module and view it on an oscilloscope or computer. They copy and reproduce the code pulses, and then program their microcontroller to recognize each signal as a particular robot command.

There really is no limit to the methods that can be used to control a computer, or even a robot.

I know of several experimenters who gutted a TV remote control and hardwired wires from the copper traces beneath the rubber buttons, then ran them to a more conventional keypad. They didn't like the TV-specific wording on the remote and fashioned a keypad more to their liking, while still using the remote's electronics.

IR control is cheap and reliable, but it does have drawbacks. Outdoors, IR is subject to interference by sunlight. Basically, the IR signal is lost

due to being overpowered by the sun, though a lot of experimental robots are used indoors in subdued lighting. Because of these types of factors and limitations, most remote control of a robot is accomplished via an RF radio link which has a much longer range, no problems with the sun or walls, and large bandwidth.

Early Model Radio Control

Let's face it. The first real radio control for hobbyists was for model airplanes and boats. When airplane modelers got tired of spinning in a circle while holding onto a pair of control lines that led to the speeding model, they began to think about radio control. The same went for model power boat builders who watched their boats heading for the opposite shore of a lake. There had to be a better way. There was. Radio control.

Before World War II, there was little available to construct small and affordable radio systems that included the transmitter and receiver. Radio tubes were large, and drew a lot of power for the filament and the high voltages required. After the war, surplus electronic equipment became available to civilians, and some good (but still bulky) radio control equipment was being built for model control.

One of the more successful commercially-available systems was the tuned reed receiver. It was essentially an electromechanical device consisting of a series of resonate reeds that would respond to a transmitted signal of the particular audible frequency. The resonating reed would vibrate a set of contacts enough to allow a current to pass through and close an appropriate relay, and thus control a specific function. These systems were used from the 1950s to the early 1970s when transistorized equipment became readily available to all.

Another system used a single button transmitter and what was

known as an 'escapement relay' connected to a receiver in a model plane. When the button on the transmitter was depressed, the signal caused the escapement to move, say, 90 degrees to the right, and move the plane's rudder fully to the right. Another push of the button would cause the escapement to move the rudder back to center. Another push moved the left rudder, and another push brought it back to center. The escapement motion was powered by a wound-up rubber band the length of the fuselage's interior.

Figure 4 is a display of four early 1960's rubber-band powered escapement control systems from www.mccrash-racing.co.uk.

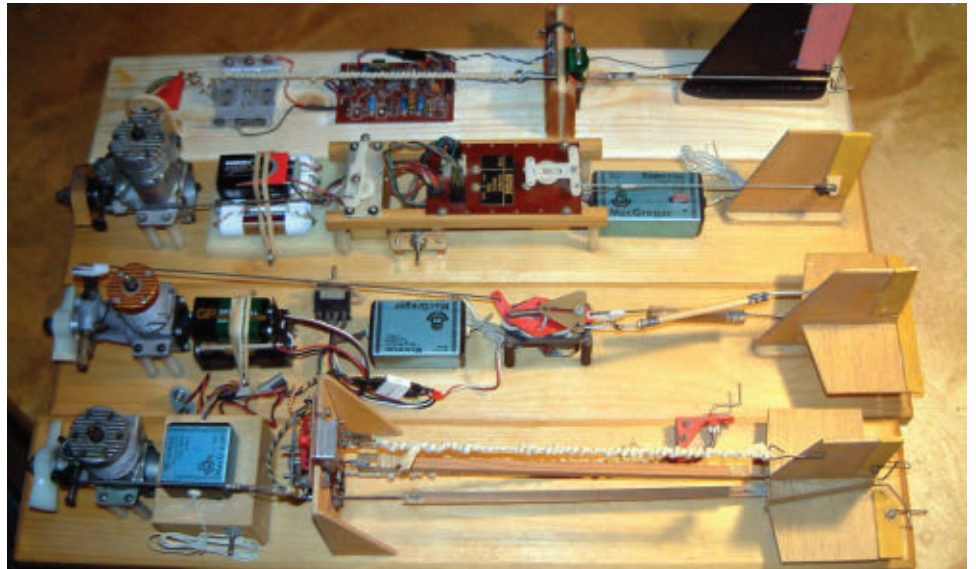


FIGURE 4. Early super-regen receiver rubber-band escapement demos.

Radio Control Systems Incorporate Servos

Jumping ahead several decades, it is quite clear that radio control systems have dramatically improved. The main improvement has been *proportional* control. Instead of the 'bang-bang' one way or via another type of control, movements can be slight and at any speed or direction that the operator desires. This is the same type of physical movement that you would find in a real airplane, car, or robot. This proportional movement is accomplished by servos: small gear motors with an attached potentiometer that feed positions back to an internal circuit board that responds to a received series of pulses. It was servos that not only allowed remote control of many movements, but these small and inexpensive devices became the main method to provide numerous motions to experimenter's robots.

When I first started to use radio control for robot movie props (such as for the 1984 film, *"Revenge of the Nerds"*), I used Futaba radios operating at 75 MHz since the 72 MHz band was limited to airborne use only. Movie prop and promotional robots really had size requirements

just the opposite of flying models. Airplanes needed the lightest and, therefore, the smallest receivers possible.

Large robots — especially promotional ones — could get by with any size receiver, however, the very smallest transmitters were used so that the operation of the robot could be hidden. Vantec supplied virtually all of the modified systems that I used

for the props. Larger robots such as action props required large motors and proportionally higher current speed controllers.

'Feedback' was achieved by human vision. The operator controlled speed, direction, and arm motions by looking at the robot as it moved around a movie set or amongst a crowd of people at a convention. To satisfy one of my customer's needs for a concealed transmitter for his promotional bot, I built the transmitter into a gutted 35 mm camera with a joystick and control buttons on the back of the camera, out of sight. The actual transmitter was in a small camera bag with multiple wires from the joystick and slide pots running into the camera bag transmitter.

A wireless microphone was also in the back of the camera. I could stand off to the side with the camera at my waist and easily control the robot. When the bot needed to speak, I would raise the camera to my face, pretending to take a photo, and speak through the mic. The attached flash worked to assist in the illusion.

The new transmitters and receivers for today's radio control have features too numerous to list. The Hitec 7 Eclipse Pro shown in **Figure 5**



FIGURE 5. Hitec 7 Eclipse Pro.

FIGURE 6.
Hitec Optic 6 Sport.



is one of this latest breed of R/C transmitters. The 72-75 MHz bands have lost favor to the new 2.4 GHz spread-spectrum systems. The more affordable Hitec Optic 6 Sport shown in **Figure 6** is a six-channel spread-spectrum system that has been used on many combat robots. The extra channels provide control through two joysticks for the wheel drives and a weapon system, with spares for firing the weapon. With microprocessor

control and informative LCD displays, there are so many functions and capabilities available in high-end transmitters. Most are applicable to model airplane and helicopter flying, but are also programmable to help make combat robots easier to control.

Radio Control Transmitter Configurations

Commercial radio control manufacturers designed two very different transmitter configurations for aircraft and for ground-based cars. The rectangular two-joystick configuration (shown back in **Figure 4**) has become the favorite for remote robot control due to the availability for more control channels. Combat robots have more functions needed other than just two differentially-driven wheels. Combatants need to control various weapons with possible requirements for unique movements. Some robots need to reverse wheel driving direction if they get flipped over in a bout.

The gun-style transmitter shown in **Figure 7** is a favorite with model race car drivers since it allows easy and quick direction control of the car's two front wheels in a race, with a

trigger controlling the car's speed. The thumb can be used to fire/trigger a weapon for a combat robot. However, most robots do not use the Ackermann-style steering that is used in race cars, but instead use differential steering.

Differential steering guides the robot by controlling the speed and direction of the right and left drive wheels. The gun-style transmitters don't have as many channels as the dual joystick transmitters have. Many non-combat robots require even more channels than their battling cousins — especially those that might have a complex arm and/or claw attached.

Model R/C control systems will continue to dominate combat robots, as well as many other experimental robots that are not under total autonomous control.

ZigBee and XBee for Robots

XBee has become a popular robot communications and control system used by many robot builders. Many confuse ZigBee and XBee. ZigBee is a specification for a suite of high-level communication protocols using small, low power digital radios based on an IEEE 802.14.4 standard for personal area networks. XBee is the name of a product made by the Digi Corporation. Digi manufactures over 70 different varieties of XBee modules with different antennas, power levels, and capabilities. ZigBee remote control has found favor with some robot experimenters, mainly because of its low power consumption.

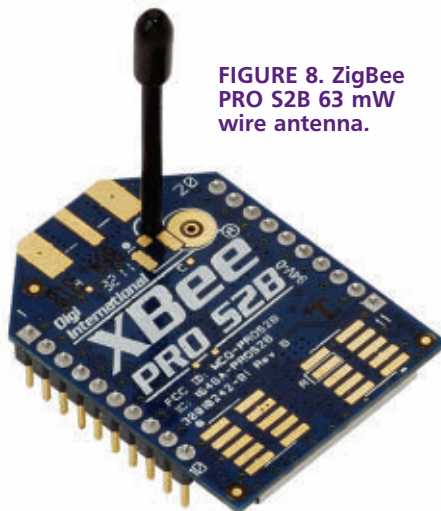
ZigBee devices are often used in mesh network form to transmit data over long distances, passing data through intermediate devices to reach more distant ones. This allows ZigBee networks to be formed ad-hoc, with no centralized control or high power transmitter/receiver able to reach all of the devices. ZigBee is targeted at applications that require a low data rate (250 kbit/s), long battery life, and secure networking.

ZigBee is suited for periodic or

FIGURE 7. Futaba 4PKS 2.4 GHz spread-spectrum transmitter.



FIGURE 8. ZigBee PRO S2B 63 mW wire antenna.



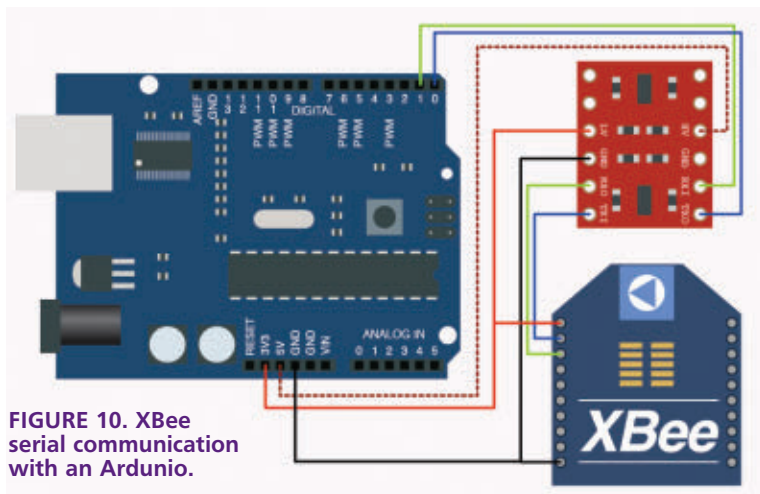


FIGURE 10. XBee serial communication with an Arduino.

intermittent data, or a single signal transmission from a sensor or input device. Applications include simple two-way robot control, traffic management systems, and other consumer and industrial equipment that requires short-range wireless transfer of data at relatively low rates. The technology defined by the ZigBee specification is intended to be simpler and less expensive than other WPANs, such as Bluetooth.

ZigBee operates in the industrial, scientific, and medical (ISM) radio bands of 915 MHz in the USA and Australia, and 2.4 GHz in most other areas worldwide. Data transmission rates vary from 20 to 900 kilobits/second. I recently received some XBee S2B Pro modules (**Figure 8**) and I quickly hooked them up for a test. I was communicating with the XStick ZB AT USB dongle (shown in **Figure 9**) on my laptop. I was hoping for a bit more range, but the dongle is low power — just a bit over 2 mW — and I had a bit of trouble with longer range due to the small, internal antenna.

I used an old spectrum analyzer to look at the 2.4 GHz signal, but it was gone by the time I brought the wireless test board with the XBee module attached down to my garage. The dongle receiver does have

a sensitivity of -90 dBm — not too bad for such a small device.

The XBee module with a wire antenna has a 63 mW output, and should be able to transmit a mile or more outdoors and several hundred feet indoors. However, to transmit in both directions with the greater range, both 'ends' must have the higher 63 mW output. Not just one.

When mounting the module, be sure to have the antenna pointing straight up and not blocked by any conductive covering. The XBee module has a baud rate of up to 1 Mbps and the dongle has 250 Kbps.

Figure 10 shows an XBee/Arduino setup with an interconnecting logic



FIGURE 9. XStick ZB AT (AT command mode ready).

level converter from the **Bildr.org** blog site. I have seen these used on RoboMagellan entries and many other types where experimenters needed data exchange for their operations.

Final Thoughts

There are many more varieties of RF and control links that have been used by robot experimenters. Wi-Fi (that uses the IEEE 802.11 standard) has been used for robot control, though we mostly use this RF technology for a close proximity PC network. It has its security problems, but is fine for robot use. Bluetooth — operating in the 2.4 to 2.48 GHz band — is another control link

technology that has been used for robot control. Bluetooth is a packet-based protocol with a master-slave structure using frequency-hopping spread-spectrum technology.

There really is no limit to the methods that can be used to control a computer, or even a robot. **Figure 11** shows a computer application that can easily be converted to robot control. These glasses developed at the Imperial College in England allow a disabled person to control a computer with just their eyes. Just think of the possibilities! **SV**



FIGURE 11. GT3D device can control a computer or robot through eye commands.

ROBO-LINKS

RobotShop.com



Pololu
Robotics & Electronics
WWW.POLOLU.COM



THE NEXT GENERATION OF MAXSONAR

The HRLV- MaxSonar Sensors

- Amazing 1mm Resolution
- Simultaneous Multi-Sensor Operation
- Superior Noise Rejection
- Target Size Compensation

\$34.95 (MSRP) www.MaxBotix.com



BRAND NEW! www.ServoCity.com
MEGA SERVOS
Incredibly powerful
4,000 oz-in of torque!



superbrightleds.com

Component LEDs - LED Bulbs - LED Products
St. Louis, Missouri - USA Fast Online Ordering superbrightleds.com



www.canakit.com

- Over 200 Kits & Modules to choose from!
- Easy Worldwide Online Ordering
- Fast Delivery & Same Day Shipping

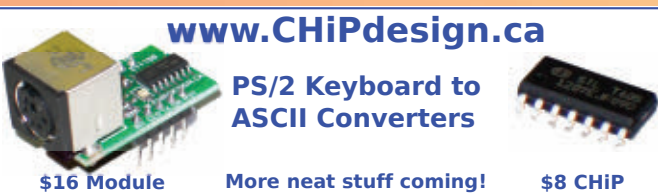
FM Transmitters • Timers • Audio Amplifiers • Motor Controllers



www.CHiPdesign.ca

PS/2 Keyboard to ASCII Converters

\$16 Module More neat stuff coming! \$8 CHiP



THE ORIGINAL SINCE 1994
PCB-POOL
Beta LAYOUT

- Low Cost PCB prototypes
- Free laser SMT stencil with all Proto orders

WWW.PCB-POOL.COM



AndyMark
Inspiring Mobility
www.andymark.com



ALL ELECTRONICS
CORPORATION
Electronic Parts & Supplies
Since 1967



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



INVEST in your BOT!

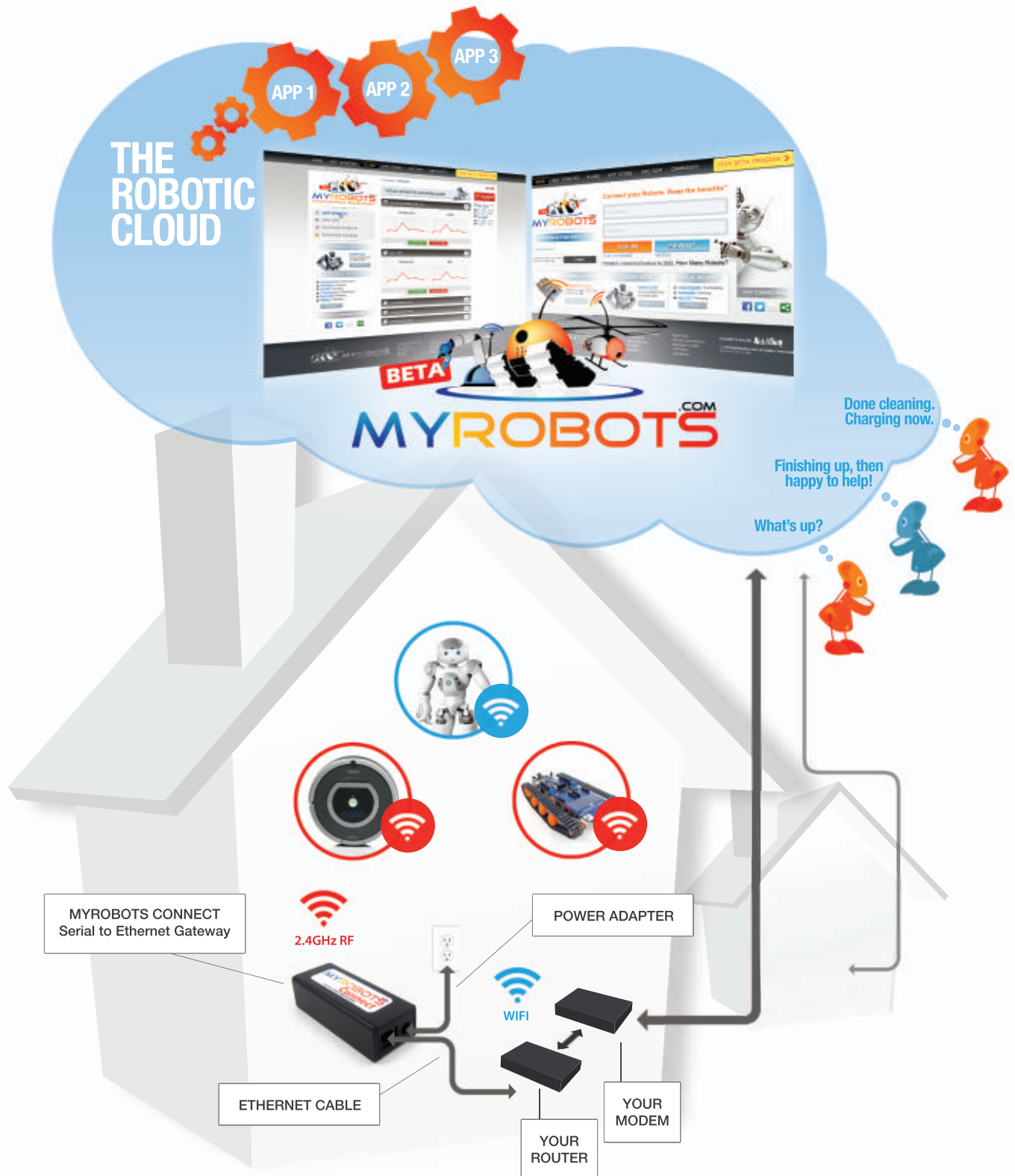
HiTEC

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



ADVERTISER INDEX

All Electronics Corp.	21, 80	HiTec	2, 80	RobotShop, Inc	80, 81
AndyMark	19, 80	Lemos International Co., Inc.	57	Robot Power	13
Anaren	7	Lynxmotion, Inc.	82	Servo City/Robot Zone	80, 83
AP Circuits	18	Maxbotix	80	Solutions Cubed	31
Cana Kit Corp.	25, 80	PanaVise	18	superbrightleds.com	80
ChiPdesign	80	PCB Pool	13, 80	Vantec	19
Dongbu Robot Co.	Back Cover	Pololu Robotics & Electronics .	3, 80	Weirdstuff Warehouse	21
Firgelli	7	Robotis	41		



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.



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



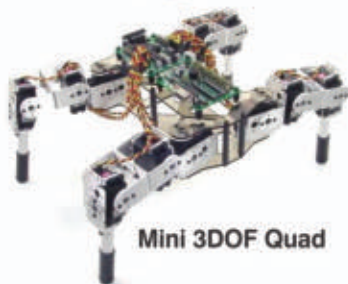
Biped 209



Walking Stick



T-Hex



Mini 3DOF Quad

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.



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.



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!

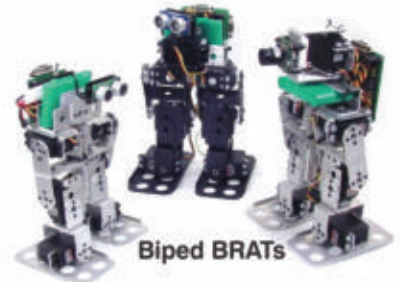
We also carry motors, wheels, hubs, batteries, chargers, servos, sensors, RC radios, pillow blocks, hardware, etc!



Visit our huge website to see our complete line of robots, electronics, and mechanical components.



CH3-R



Biped BRATs



Phoenix



Images represent a fraction of what can be made! www.lynxmotion.com The SES now has over 200 unique components!

SERVOCITY

A DIVISION OF ROBOTZONE, LLC

WE HAVE THE PARTS FOR YOUR IDEAS!

@ServoCity
Follow us on Twitter

**HUGE
SELECTION**

From **micro** to **MIGHTY**

We have the perfect pan & tilt for your application!

SPT400 Tilt

Starting at
\$94⁹⁹

**SPT50 Micro
Pan & Tilt Kit**

\$19⁹⁹

Just 0.29oz!

**SPT100
Pan & Tilt Kit**

\$19⁹⁹

**SPT200
Pan & Tilt Kit**

\$45⁹⁹

Perfect for
your GoPro!

GoPro
HERO

**PT785-S
Pan & Tilt**

\$349⁹⁹

Great for point
and shoot
photography

Closed Loop

- Aluminum Hollow Shafts
- 6061-T6 Aluminum Construction
- 48 Pitch Gear Drive
- Precision HS-785HB Servos
- 4.8-6.0 Volt Operation
- Fully Assembled

Top Mount Pan

Starting at \$69⁹⁹

**DDP155
Pan Kit**

\$39⁹⁹

DDP155 shown
with DDT540

Worm Drive Gearbox

\$59⁹⁹

Shown with
round ABS plate

**Belt Drive
Pan**

\$159⁹⁹

**R-2200
Roll
Attachment**
(For PT-2100)

\$599⁹⁹

Upright or
hanging position!

MPT1100-SS Pan & Tilt \$649⁹⁹

Ideal for DSLR and
smaller bodied cameras!

Mount directly
to jib handle

Open Loop

Extremely smooth
and precise!

**PT-2100 Super Duty
Pan & Tilt**

\$1499⁹⁹

Open Loop

- Adjustable Framework
- Stainless Steel Hollow Shafts
- 6061-T6 Aluminum Construction
- Kevlar Belt Drive
- Ultra Precision Gearmotors
- 12 Volt Operation
- RTR (Ready to Run) out of the box

**Bottom Mount
Pan System**

Starting at
\$79⁹⁹

You Tube

Want to see these systems in action?

Watch our product demos online at www.YouTube.com/ServoCity



Wired Controllers



Power Supplies



Gears & Sprockets



Pulleys & Belts



Servos & Access



Jib Mounting Kits



Gearmotors



Wireless Controllers

Like us
on Facebook



To view our entire selection of products visit us online at

WWW.SERVOCITY.COM

Specializing in Servos, Radios, Motors, Pan & Tilt Systems,
Batteries, Wiring, Connectors, Gearboxes and more!

Watch our
videos on YouTube



Copyright 1999-2012 Robotzone, LLC. - All Rights Reserved
ServoCity is a registered trademark of Robotzone, LLC.

Phone: (620) 221-0123
E-mail: Sales@ServoCity.com

Prices and availability subject to change without notice.

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