ROBOT SENSORS • ROBOBUSINESS • GOING BRUSHLESS

SERVO MAGAZINE

FOR THE ROBOT EXPERIMENTER
www.servomagazine.com

JULY 2007

Web-Based TeleRobotics

Turn An Ordinary Webcam Into An Extraordinary Controller

- Neural Networks For The PIC
- Build A Bluetooth Comm Unit
- Beginner's Guide To Programming
- Uncle Sam Wants You (To Develop UGVs)

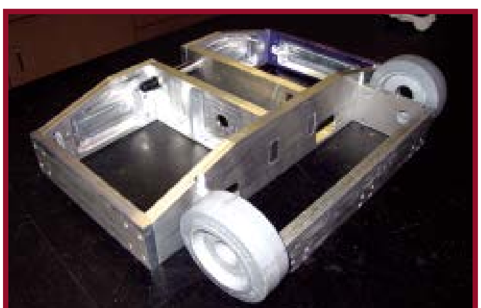
SERVO MAGAZINE

Features & Projects

Columns

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

07.2007
VOL. 5 NO. 7



Departments

ENTER WITH CAUTION!

Look for a "Heavy Power Month" in the August Combat Zone!

Mind / Iron

by Bryan Bergeron, Editor

As detailed later in this issue, RoboBusiness 2007 (held this May in Boston, MA) showcased a variety of commercial and military robots and robotic products. A major difference between the robots at the exposition and the robotic projects typically carried by *SERVO* is intellectual property protection. Most commercial robots are covered by multiple patents that can be used by the patent assignee to exclude others from making, using, offering for sale, or selling the components of the robots covered by the patents. However, as an enthusiast, you can freely use the intellectual property described in patents as the basis of your personal robot designs and as a general robotics reference source.

The easiest way to explore the intellectual property associated with commercial robots is to visit the US Patent and Trademark Office (USPTO) at **www.uspto.gov**. Once there, you can search by patent number, assignee, or search term, such as "robot." To illustrate the wealth of information available online through the patent office, let's suppose that you are interested in building an autonomous vacuum cleaner for your workroom. Of the dozen or so robotic vacuum cleaners on the market, the iRobot Roomba is the most popular, suggesting that they've done something right. A reasonable first step would be to search the USPTO for "iRobot" as assignee and see what turns up. The assignee or current legal holder of the patent is often different from the inventor, which is permanent.

One of the many patents assigned to iRobot (patent number 6,956,348, Debris Sensor for Cleaning Apparatus) describes a piezo electric debris sensor system that enables an autonomous robot to steer in the direction of debris. The patent includes a textual description of the apparatus, 10 detailed drawings that show sensor placement, a circuit block diagram, the mechanical construction of the Roomba, and low level circuit diagrams. In short, the patent provides everything you need to know to construct, install, and operate a debris sensor for your robot.

As another example, suppose you're designing an amphibious robot and want to see what's been patented in the way of propulsion systems. A search for "amphibious robot" in the patent title reveals several amphibious robot designs, including #7,007,626, Amphibious Robot Devices. The patent includes six figures detailing the shape and movement of fins, schemes for overcoming obstacles and trenches, and other practical considerations related to robot propulsion.

Another way to use the wealth of information in the USPTO is to search on the patent numbers affixed to a device or product of interest. Recently I was working with Floam — a microbead crafting compound sold in toy stores — to create temporary and permanent robot components. I had trouble with the product drying out, even though I returned it to a sealed container, and there were no remedies on the package. I searched for the patent (number 5157063) and discovered that the product could be reconstituted to its original consistency by adding a small amount of Lubriderm moisturizing lotion or K-Y jelly.

If you're like most enthusiasts, you'll find it almost impossible to read through the USPTOs holdings and not consider submitting a patent application for that

Mind/Iron Continued →

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

Subscriptions
Inside US **1-877-525-2539**
Outside US **1-818-487-4545**
P.O. Box 15277
North 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, Gordon McComb, David Geer, Pete Miles, R. Steven Rainwater, Michael Simpson, Kevin Berry, Fred Eady, Pete Smith, Ted Larson, Alan May, Brian Benson, Chad New, Bryce Woolley, Evan Woolley, Matt Maxham, Wendy Maxham, Heather Dewey-Hagborg, Dan Kara

CIRCULATION DIRECTOR
Tracy Kerley
subscribe@servomagazine.com

MARKETING COORDINATOR
Brian Kirkpatrick

WEB CONTENT/STORE
Michael Kaudze
sales@servomagazine.com

PRODUCTION/GRAPHICS
Shannon Lemieux

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher

Note to SERVO Readers:

I have a couple of updates to the June '07 article "Robot Simulation For Everyone."

Some of our new error checking has forced a minor change in some of the original programs. In particular, the simulated robot must always be initialized before any robot commands are issued. Because of that, there is a minor change that must be made to one of the program listings in the article.

The very beginning of Figure 1 currently looks like this:

```
// this is a comment   First we draw
// the line
gosub DrawLine
// Place the robot at the beginning of
// the line and face it left 90 degrees
rLocate  191, 71, -90
end
```

The second two lines need to be moved to the beginning of the list as below:

```
// Place the robot at the beginning of
// the line and face it left 90 degrees
rLocate  191, 71, -90
// Then we draw the line
gosub DrawLine
end
```

The latest version of RobotBASIC, as well as a number of demo programs, are available at **www.servomagazine.com**.

— John Blankenship

project you've been perfecting for the past few years. My advice is to stay clear of the inventor clearing houses and support agencies and instead find a reputable law firm specializing in intellectual property. Another suggestion — based on personal experience — is to consider your goal in obtaining a patent.

Some people like to collect patents. While a patent is a valid status symbol, the only thing a US patent guarantees you is recurring fees that must be submitted to the USPTO to keep your patent alive. If you intend to license your patent, then make certain you can cover the fees in your projected license income. If you intend to assign the patent to a third party, then you'll have to factor in the costs of obtaining the patent in the sale price. Recent figures from the IEEE suggest that you should expect to spend — on average — about $11,000 on the initial patent application.

The USPTO website has a good introduction to the patent process, fees, and schedules. Take a look at the Trademark and Copyright sections while you're there. It may be that your invention is better protected by a relatively inexpensive Trademark. Also, if you do find that patent protection is the way to go but your design isn't finalized, then consider filing an inexpensive provisional patent application. It provides a year extension before you have to submit a regular patent application. Whether or not you're interested in obtaining a patent, make a point of adding the USPTO to your browser's bookmarks. **SV**

Robytes

by Jeff Eckert

Bot Headed for Mars

A vital instrument on Phoenix, this robotic arm will dig into the Martian soil for analysis. Photo courtesy of NASA/JPL/UA/Lockheed Martin.

Digging in the dirt may not be a particularly piquant robotic function, but it helps when you're doing it on Mars. At present, NASA's Phoenix Mars Lander is scheduled to head that direction — weather permitting — on August 3rd. Phoenix is the first mission of NASA's Mars Scout Program of competitively proposed, relatively cheap missions to the red planet.

Selected in 2003, Phoenix saves money by using a lander structure and other components originally built for a cancelled 2001 mission. The robotic arm will scrape into the icy soil on a Martian arctic plain next spring, collecting samples and bringing them back onto the Phoenix's science deck where it will be analyzed in terms of aquatic history and possible complex organic materials. Details about Phoenix are available at **http://phoenix.lpl.arizona.edu**

The Phoenix Lander begins to shut down operations as winter sets in. The far northern latitudes on Mars experience no sunlight during winter. This marks the end of the mission because the solar cells can no longer charge the batteries on the lander, and the frost covering the region as the atmosphere cools will bury the lander in ice. Photo courtesy of Corby Waste of the Jet Propulsion Laboratory.

Artificial Snot Enhances Sensors

Leave it to the people who invented black pudding, the Bowler hat, and Imperial measurements to keep coming up with strange concepts. One of the latest is "artificial snot," which researchers at the University of Warwick (**www2.warwick.ac.uk**) and the University of Leicester (**www.le.ac.uk**) have devised to enhance the performance of electronic noses, which are commonly used in robotics and other applications ranging from food quality control to toxic substance sensing.

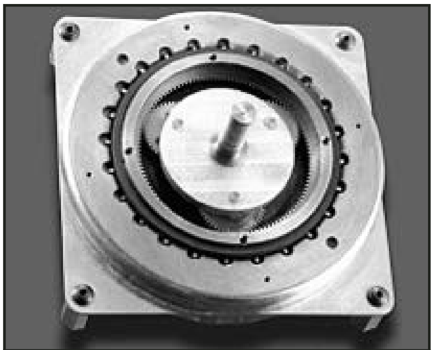
It seems that the human nose incorporates more than 100 million receptors that work together in very complex ways to identify the molecules they encounter. However, electronic noses often have fewer than 50 sensors, so they discern a much narrower range of smells.

One of the ways a natural nose accomplishes its mission is to dissolve the scents in mucus, allowing them to arrive at receptors at different speeds, and our brain somehow uses this information to sharpen the smelling operation. Mimicking this process, the Warwick and Leicester team placed a 10 micron thick layer of polymer, normally used to separate gases, over the sensors in their electronic nose. Apparently, the device can now make heretofore impossible distinctions, such as between milk and a banana. The improved device, including the sensors and mucus, can be produced for less than $10, so keep it in mind for your next project. Details are available in the Proceedings of the Royal Society (**www.pubs.royalsoc.ac.uk**).

Mucus meets artificial mucus: Prof. Julian Gardner and improved sensor. Photo courtesy of the University of Warwick.

High-Torque, Thin-Package Motor

Also on the component level is an improved planetary gear train pancake motor from Haydon Switch & Instrument (**www.hsi-inc.com**). By using a gear train located inside the motor, Haydon has devised a product with a package that is only 18.5 mm thick and 80 mm in diameter. Nevertheless, it provides up to 120 oz-in (85 N-cm) of torque and is available with a 3.75° step angle and a



Haydon's planetary gear train pancake motor is designed for applications with limited space and a need for accurate positioning and high torque. Photo courtesy of Haydon Switch & Instrument, Inc.

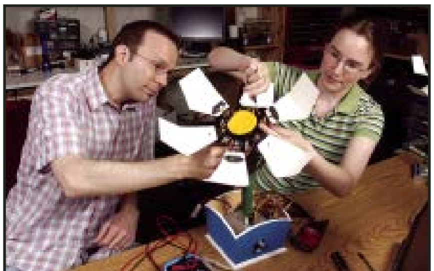
4:1 gear ratio. The company produces a variety of stepper-based linear actuators, rotary motors, lead screw assemblies, and switches.

Sewer Bot Catches Mugger

Naked suspect hauled out of Seoul sewer after being tracked down by an inspection robot. Source: Chosun Ilbo.

Even if you've heard about this one, it bears repeating. It seems that a 57-year-old resident of Seoul, Korea, recently snatched a woman's purse at a hospital. When witnesses tried to grab him, he inexplicably shed all of his clothes, scurried into the city's sewer system through a 1 meter dia. pipe, and holed out there for about four hours. The police adroitly enlisted a six-wheeled, camera-equipped inspection robot to track him down. Ironically, he was then taken back to the hospital rather than the clink, having developed hypothermia from the 1°C (34°F) temperature. When asked why he took off his clothes, the suspect reportedly just said, "Leave me alone. I'm not feeling well."

Simple iBots



Christopher Bartley and Emily Hamner make adjustments to a TeRK Flower, one of many possible assemblies based on the Telepresence Robot Kit. Photo by Ken Andreyo of CMU.

Some ongoing research at Carnegie Mellon University's Robotics Institute, specifically in the Community Robotics, Education, and Technology Empowerment (CREATE) Lab, has been aimed at the creation of a series of robots that are (a) bonehead simple enough for nearly anyone to build from off-the-shelf parts and (b) sophisticated enough to perform useful operations under wireless Internet control. The idea has manifested itself in the form of the Telepresence Robot Kit (TeRK), which is actually a set of "recipes" that one can follow to create a wide range of customized bots. They can take many forms, from a mobile model equipped with a digital camera to a flower loaded with infrared sensors (see photo). All TeRKs are based on the same controller, called Qwerk, which combines a computer with the various software and hardware components of the assembly.

Although the TeRK goal is to make available highly capable robots that are affordable for students and anyone else interested in robotics, the website says that a robotic flower will cost you about $750 to build, which is more than I paid for my last car, so be advised that "affordable" is a somewhat subjective concept. Recipes, software, technical support, and other information are available free at the TeRK website (**www.terk.ri.cmu.edu**). The Qwerk controller is available for sale from Charmed Labs (**www.charmedlabs.com/**). SV

GEER HEAD

by David Geer

Contact the author at geercom@alltel.net

Kiva's Robot Workhorse Systems Hustle in the Warehouse

Similar in color to a hive of worker bees, the little orange Kiva robots (drive units) move about, attaching themselves beneath the blue inventory racks (pods) and carrying them to their intended destinations ...

The Kiva Mobile Fulfillment System (MFS) is the umbrella name for the overall Kiva family of distribution center systems. It consists of three product groups including the CaseFetch™, ItemFetch™, and OrderFetch™ products.

CaseFetch is the system we see most of in the images presented here, with the little orange drive robots moving entire racks of inventory. ItemFetch picks individual product out from cases for a specific order. OrderFetch sorts orders and takes them to the correct shipping area. Today's distribution centers (DCs) can use one, two, or all three products in unison.

ItemFetch is used to pick individual items from cases (for example, for a consumer order), OrderFetch is used to sort the orders (in a box or tote) to the correct shipping door, and CaseFetch is used to move full pallets around so that workers can pick off complete cases.

The systems share common components and functions. All inventory is racked on a tall, blue pod, which is picked up from its foundation by one of the orange drive units. Hundreds of mobile drive units in a single warehouse system can communicate with the central server wirelessly.

The robots take their direction and course from a combination of where the server tells them to go and patterns of 2D bar codes attached to the floor as roadmaps.

The Kiva central server system, which integrates with the customer's warehouse management systems, enables order processing through the Kiva system so the right robots grab the right inventory and bring it to the right operator location. Workers can now wait for the products to come to them, which is faster and more precise than retrieving the inventory by hand.

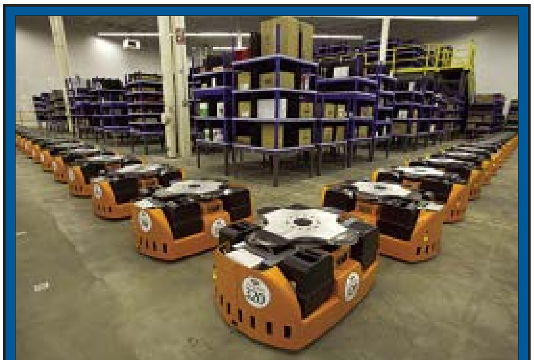
Little "Modules," Big Difference

The Kiva system is modular, highly standard, and conforms to itself in a way that enables it to scale to meet the needs of large warehouses, distribution centers, and storage facilities. All an organization needs to grow the Kiva system to fill its expanding needs is to add more pods, more robots, and more workstations for inventory workers.

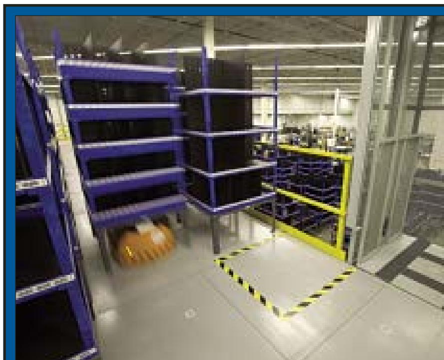
Customers can add to the central server system so it can handle the extra data and commands, as well. The larger system forms a computer cluster that shares the processing load like a grid computing system. Systems can scale upward starting with four stations, 12 robots, and a few hundred odd inventory pods initially.

From there, customers can expand the systems to include several dozen stations, several hundred robots, and even thousands of inventory pods. The system is controlled and routed without robots and racks

This is a grouping of the orange Kiva drive units (robots) posing outside of a set of inventory pods, which are mobilized by the drive units.



An operational Kiva-enabled warehouse with a blue inventory pod and orange drive unit on the move!



bombarding each other. Kiva simulates each system to ensure that it can handle the expected workload in the customer's actual environment.

Kiva-enabled warehouse with pod and drive unit going up elevator.

Operator loading inventory onto inventory pod and drive unit from a manual fork.

From Robot to Robot

Kiva's three systems use two kinds of robots between them. The Order and ItemFetch products both use a small robot drive unit that can lift up to 1,000 lbs. This smaller robot is about 36" x 20" x 16" (tall). The CaseFetch product must use a larger robot that can handle up to a 3,000 lb load. This larger robot is about 48" x 40" x 12" (tall).

All robot systems take advantage of both the wireless networking communications between them and the central server. They all use the 2D floor-based bar code tracking system to navigate through the given warehouse. The robots also possess several sensor systems to ensure they do not crash into employees, pods, products, or each other.

Navigation Station

The robot drive units read bar code marker trails patterned in a grid on the floor. From these, the robots can tell where they are on their way to their next pick-up or drop-off. The central server controls everything using highly sophisticated software.

Orders pass from customer computer systems and are translated by the Kiva system central server into the robot's specific jobs and paths. The Kiva computers send commands to the robots wirelessly.

Each robot's wireless radio network not only receives its commands, but also communicates back its position and confirms completion of its tasks. Each robot has a camera eye that monitors the floor, reading the bar code stickers from the grid. The robot updates its location information with every sticker it passes.

The robots are battery powered and will automatically home in on and return to specially equipped recharging stations as needed, rather than taking on another load. A power control software system in the robots monitors their battery charge levels.

When a bot reaches a certain level of low charge, the drive will request permission from the computer to go to a charger and recharge. When the robot is fully charged, it leaves the charger automatically so other robots can use it (how courteous!).

Play Fetch

With CaseFetch, pallets loaded with goods arrive at the DC and are forked off the trucks and onto the inventory pods, then delivered to storage. The pallets are stored in a very dense, uniform grid pattern that allows maximal use of available storage space.

Human operators stand at-the-ready at workstations around the storage area. When it is time to fulfill customer orders from the storage area inventory, the robots retrieve pallet and case inventory pods and bring them out to the human operator. With the ItemFetch system, the robots can open cases of products and take one or more out of the storage area to fulfill a customer order.

Curiosities

Kiva systems are first in largest systems of their type every installed using fully coordinated autonomous robots. Rather than being some experiment or trial, these systems are complete, fully developed and in production use in industry.

The huge office supplier Staples is making a good deal of use out of these systems in their warehouses. "These systems represent the simple, cheap, and reliable model of robotics that work in real-life conditions," says

THE MANY MOVES OF THE KIVA MOBILE FULFILLMENT SYSTEM

The Kiva Mobile Fulfillment System is the largest installed warehouse automated inventory order fulfillment mechanism to-date. It has many features and capabilities.

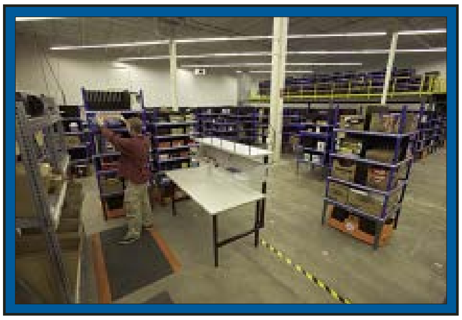
Kiva can install the system in most any warehouse in a day. A human operator can use the orange robots and blue inventory racks to pick inventory from a storage location virtually every six seconds, not that they can necessarily even operate their controls that fast.

The Kiva system fulfills orders instantaneously with high accuracy. When an operator requests an inventory item, it is verified by a bar code scanner and/or photo comparison to keep errors down and rid the demand for quality control.

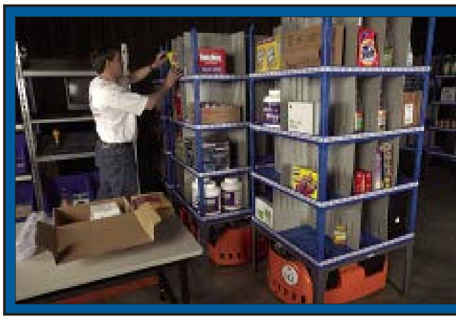
The mobile robots, inventory pods, trays, and bins are modular; there is no need for forklift aisles. Because the system uses the full vertical space of the warehouse, higher product density is possible.

The modular design enables easy expansion into the existing facility to deal with higher traffic seasons or overall growth. The system sorts and separates orders automatically. Because the rest of the system keeps working when a robot fails, there is no down time.

Inventory can be moved in and out of the warehouse at the same time with no traffic jams, making it possible to move inventory fast. Each operator's work is independent of the others, making it possible to gauge their individual work productivity all the time.



Inventory being picked and packed onto inventory pods for movement or retrieval by drive units.



An operator in a Kiva workstation interacting with an inventory pod. See orange robots at bottoms of inventory stacks.

RESOURCES

Kiva Systems Home
www.kivasystems.com
Includes a link to a Flash video and links to articles published about Kiva that include their own videos.

Online Kiva System Robots Demo
www.kivasystems.com/demonstration-login.php

Distribution Center Tour
www.kivasystems.com/solution-dctour.html

Scott Love, a Kiva representative.

Anecdotally Speaking

Mick Mountz, CEO and founder of Kiva Systems, was the design mind behind not only the Kiva Mobile Fulfillment System but also the next-generation distribution centers of the web-based grocer, WebVan, which shipped groceries direct to homes. His experience with WebVan played heavily into the creation of Kiva Systems.

To pick and place grocery items into orders, WebVan constructed a mix of conveyor belts and carousels in a system that — in theory — got the proper items assembled together for an order. However, the WebVan system was complex and often broke down. Orders were late and food ended up spoiling; it went bankrupt in 2001 signalling the beginng of the dotcom bust.

Then Mick came on an idea. If the inventory could move itself, and if it could communicate with you so you could tell it to come here, that might work. Mick began working on how to make his idea a reality. Eventually, with the help of Peter Wurman and Raffaello D'Andrea, Mick developed the complex, multi-agent system now known as the Kiva Mobile Fulfillment System. **SV**

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?

ASK MR. ROBOTO

by
Pete Miles

Q. Many new microcontrollers have low voltage supplies (around 3V or even down to 1.8V). This poses a problem to the logic levels needed to control a typical hobby R/C servo. Do you know of any low voltage servos or are logic level shifters the only viable solution?

— Alex
Lisbon, Portugal

A. I am not aware of any servos that are designed to operate with control signals down to 1.8V, though some Futaba servos will accept a 3.3V logic signal. Most R/C servos are designed to work with power supplies that are normally used to power the radio receivers — 4.8 or 6.0 volts. Some of the advanced robotic servos may work with logic levels down to about three volts, such as the AI modules from Magarobotics (**www.tribotix.com**), but this is operating at the very low end of the voltage specification, and may not be reliable.

The new digital R/C servos may operate below the three volt logic levels. For example, the Hitec HS-5645MG uses an Atmel AT90LS4433 microcontroller. This microcontroller has a minimum logic voltage threshold of 2.75V which should allow 3V control signals to control this servo.

If you are planning on using one of these new microcontrollers that operate below three volts, you will definitely need to use logic level shifters to control your servos. If you are operating between three and five volts, a logic level shifter is recommended, and you should be using the newer digital servos instead of the older analog generation servos.

Q. I am a regular reader of *SERVO* and have all the issues to date.

Currently, I am building a wall climbing robot, but I still have this problem navigating to a beacon. About six years ago, I had heard that the soccer playing robots had a detection range of only three feet. So I decided to make a robot with a greater detection range. In my first attempt, I got more than 15 feet and then later I was able to detect a ball at 30 feet. My website at **www.geocities.com/macx75/robotics.htm** shows a robot that plays on its own with a ball and kicks it around a room. I thought that this system could be used as a beacon in an open field so that a robot could navigate to it even if there were many obstacles blocking its mission.

Today's infrared sensors are more powerful, offer more range, and are also easier to program. There are many applications for their use: lawn mowing robots (especially in teams), or like with the golf robot question asked in the March '07 issue of *SERVO*, where you attached the sensor to yourself and the golf caddy kept following you at a fixed distance. Or, a model rocket finding robot to help with recovery of the rocket body. My question is about building a beacon.

1) What is the maximum range you can sense with infrared (so model rocket finding robots can search for long distances)?
2) How do you make a long range beacon (more than 300 feet)? This is a line-of-sight distance metric.
3) Are there any radio devices that emit signals from a particular location, making for a good robot beacon?
4) If I am in a four room house where the beacon is in one room and the robot is in another, is there any current device or sensor that can make my robot just point to the location of the beacon?

— Dr. Gopal Patel, INDIA

A. First off, I would like to encourage you to submit an article on the long range object detectors that you have built. Detecting a ball at 15 and 30 feet is quite impressive. Many of the readers of *SERVO Magazine* would be very interested in learning all about your sensors.

The maximum range you can detect infrared signals depends primarily on the intensity of the infrared source and the sensitivity of the detector. The more sensitive the detector becomes, the more likely it will respond to other light frequencies/wavelengths. To minimize the number of occurrences of false signals, you should use an optical bandpass filter. Such a filter will block all wavelengths of light except the one that you are trying to detect. Choose the wavelength of the bandpass filter to match the same wavelength of the infrared source you are using. One company that has a large selection of bandpass filters is Edmund Optics (**www.edmundoptics.com**).

To increase the intensity of the

infrared source, you need to either use multiple LEDs in series or parallel, or use some of the high powered (i.e., greater than 1W) infrared LEDs. Here are a couple of companies that sell high powered infrared LEDs: **www.rentron.com/remote_control/IRLED.htm**, **www.roithnerlaser.com/LED_HP_single_chip.html**. Higher powered LEDs will greatly increase the range of your signals.

As for model rockets, it is probably better to use an audio or radio frequency detection method since it is unlikely there will be a direct line-of-sight between the rocket and the robot during the recovery process. Model rocket tracking systems are common in the high powered model rocket community. The following companies sell complete radio frequency based tracking systems: **http://rockethunter.com**, **www.adeptrocketry.com**, and **www.ukrocketman.com**. These systems may be modified to interface with a robot.

Now, if you need to incorporate a tracking system in your robot, take a look at the various circuits at Jerry's Electronic Plans, Kits, and Curious Things website (**www.jbgizmo.com**). There are complete plans for building tracking systems for model rockets, in addition to tracking systems for animals, people, golf carts, etc.

Optical beacons are not practical in a building with many rooms since they require a direct line of sight between the beacon and the robot, unless you are planning on having your robot search the building for the beacon. Or, you have several robots working together as a team searching for the beacon. If you are using multiple robots, then the robots need a way to talk together so that the beacon location can be transmitted from robot to robot. A good system for communicating between robots is the BlueSMiRF radio modem sold by Spark Fun Electronics (**www.sparkfun.com**).

A better system for indoor tracking would be a radio frequency system, since radio frequencies pass through walls. So if your robot can detect the direction of the radio source, then it can navigate towards it by using short range obstacle sensors that will enable the robot to navigate around obstacles and walls as it moves towards the beacon. The same systems sold for model rockets or the circuits shown at Jerry's Electronic Plans website should work well for you.

Another place to look for information in how to do this is to conduct an Internet search for search and rescue robots. Search and Rescue robots can do the same things you asked about. Most of this information is presented in research papers from universities.

Q. I am looking into buying a mini-mill. For the most part, the mills from Grizzly and Harbor Freight look identical. The main difference is the spindle taper mounts. Grizzly uses a MT#3 mount and Harbor Freight uses a R8 mount. What is the difference between these two mounts? Also, does it matter that the Harbor Freight mini-mill has a slightly more powerful motor than the Grizzly mill (4/5 HP vs. 3/4 HP)?

— Phillip Bayne
San Diego, CA

A. For all practical purposes, these two mills are identical, except for color (Grizzly = green, Harbor Freight = red) and spindle taper. I don't know how the horsepower ratings are determined for these motors. These motors are actually 350W, which is equivalent to 0.47 HP (1/2 HP). The advertised 3/4 and 4/5 HP motor power ratings are a bit puzzling. The true specifications for these mills come from the mill's manufacturer, Shanghai Sieg Industrial Co. (**www.sigind.com**). So, there is no real power advantage between the two.

The Morse Taper #3 (MT3 or MT#3) is more commonly used on drill bits and lathe tooling, but work just fine for a spindle taper mount for the mini-mill. The R8 spindle taper is more commonly used with mills. There is no real performance difference between the MT#3 spindle mount and the R8 spindle mount. They both will do a fine job at holding your cutting tools and machining parts. The only drawback is that the MT#3 spindle taper is not a common size for the milling community, thus tooling is harder to find and is more expensive.

The R8 spindle taper is used on many different milling machines, especially the larger sized mills. Thus, tooling investments (which always cost much more than the cost of the mill) for R8 tooling will be transferable to other machines, whereas the MT#3 tooling will always stay with the mini-mill.

It is always best to make sure that your tools are as interchangeable as possible with other systems, so that they are easier to find, lower in cost, more options to choose from, easier to repair, and easier to sell to other people. If there are no other reasons for choosing one mill over the other, I would choose the mill with the R8 spindle since tool-holding parts will be easier to find. **SV**

This month, we have the opportunity to present a device that would instantly ratchet up the cool factor on any robotics project — a voice recognition module. The SR-06/SR-07 Speech Recognition Kit from Images SI (**www.imagesco.com**) is an exciting project in and of itself, but the possibility of hooking it up to a robot to literally have it at your beck and call makes it all the more enticing. We always liked giving our robots names, and now we had the means to have them respond to them. We thought it was high time for robots to learn our language instead of the other way around, anyway.

Do You Understand the Words That Are Coming Out of My Mouth?

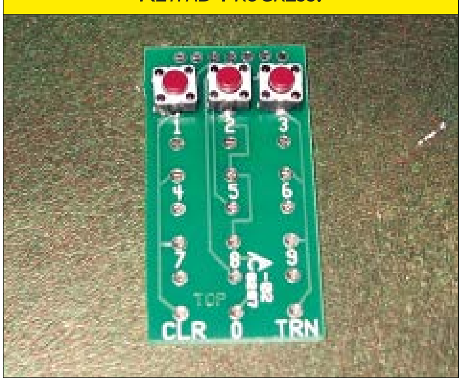
The speech recognition kit comes in pieces, so it needs to be assembled before you can start barking orders at your robotic minions. The kit comes with a short instruction manual that has clear and concise directions for soldering all of the electronic bits onto the PCB (printed circuit board). There are three PCBs, actually — the main board, the display board, and the keypad. The split-up boards create a nice situation for the busy tinkerer — you can work for just a little while and finish one of the boards, then come back later to finish the rest. The kit would certainly be possible to assemble all in one sitting, but with the fairly high number of parts, it would be a long sitting.

We worked on the board in waves, first finishing the keypad, then the display board, and finally the main board. The directions were very straightforward, and the kit went together easily.

The speech recognition circuit requires a nine volt battery for main power and a CR2032 coin cell as a backup that allows the circuit to remember words even after being turned off. The kit is a classic case of "batteries not included," but a quick trip to the electronics store had the circuit up and running.

VOICE RECOGNITION BITS.

KEYPAD PROGRESS.



You Talkin' to Me? *You Talkin' to Me?*

The final module turned out to be a bit bulkier than one might expect. The idea of a module conjures up images of a nicely contained unit that would be unobtrusive when attached to some other device. The speech recognition circuit, however, is not exactly the perfect picture of a compact module with its multiple boards sticking out to make it look like some sort of ill-proportioned electronic angelfish.

The extra boards were removable at least, and the tinkerer pressed for space could teach the circuit some words and then remove the keypad and display board. That could even well be what we would do, but first we had to teach the circuit some words.

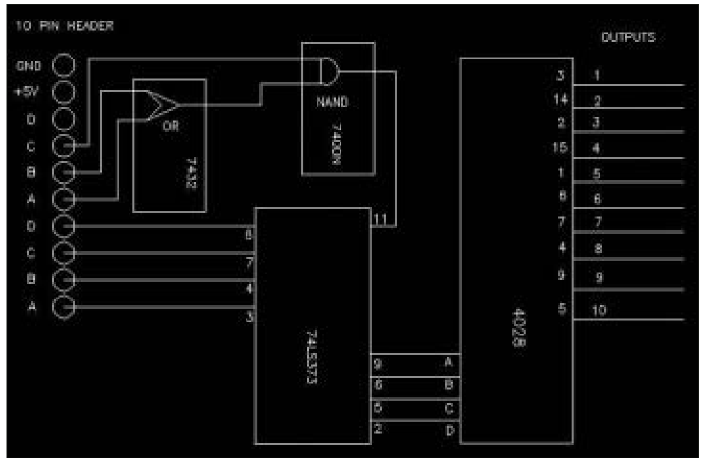
The instruction manual that comes with the kit also includes clear directions for teaching the circuit words, and even a nice examination of some of the potential difficulties, modifications, and applications that one might want to consider exploring with the device.

According to the manual, turning on the circuit (it comes with an on/off switch) should turn on the bright red LED. After a moment of suspense and a flip of a switch, we saw that we were on the right track. We also soon discovered that teaching the circuit words was a fairly painless process.

The default vocabulary for the circuit is a lexicon of 40 short words, each with only a length of 0.96 seconds. A robot probably wouldn't need commands much more complex than "right," "left," "back," "spin," "dance," "amalgamate," and that sort of thing, so the default vocabulary would be effective in most cases. But for the folks more along the line of insisting that their robots react to commands like "supercalifragalisticexpialadocious," the circuit comes with the option of changing the vocabulary to one of 20 words of a 1.92 seconds length. Not exactly supercalifragalisticexpialadocious (unless you're an auctioneer), but certainly long enough for reasonably detailed commands.

After turning on the circuit, all you had to do to teach the robot a word was select the number of the word you wanted to teach (a number between one and 20). Once the number is selected, all you have to do is hit the TRN (Train) button on the keypad and say the word into the headset microphone. Be sure to speak clearly and enunciate.

If there are any problems with teaching the circuit words, error codes will show up on the display board, and a quick look to the instruction manual will give you the right troubleshooting tips. The supremely helpful instruction manual comes with a plethora of tips and tricks to make sure that your words are recognized properly. It contains a nice discussion on how to make the circuit more robust by lowering the vocabulary to five words and giving each word four spots, each one with a different inflection. With some specific assignments to certain numbers, the circuit should be able to cope with different inflections of the same word.



VRC INTEGRATOR.

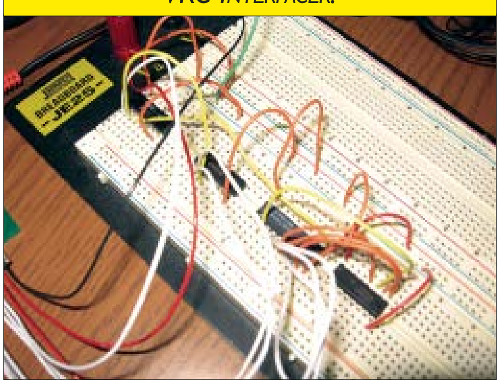
I'm Listening

The idea that a circuit can learn words is pretty exciting, but the circuit on its own doesn't provide much in the sense of feedback beyond identifying the taught words by showing their corresponding numbers on the display board. This can certainly be entertaining for a time, but we are sure that most tinkerers would agree with us that the real excitement comes from integrating the circuit into another project to make it voice controlled.

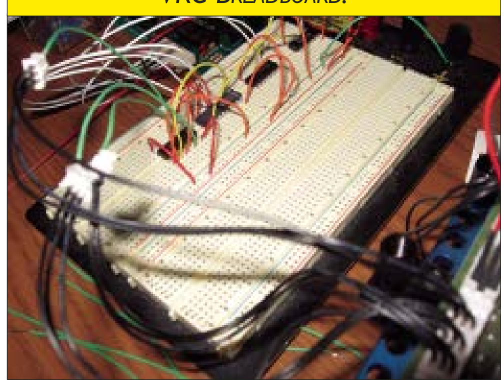
Fortunately, the folks at Images SI have anticipated the predilections of its audience, and the kit comes with several options to interface it with the outside world. Within the instruction manual is a schematic for an interface circuit, but this schematic is general and vague, perhaps intentionally so. Interface circuits can be purchased from Images SI via their website, but these interface circuits will run you about as much as the kit itself.

We took this as a throwing down of the gauntlet, and we were resolved to create an interface circuit on our own. That might not be a tall order for many of the fine electronics whizzes that read *SERVO*, but we are sure that we are not the only roboticists of a more mechanical predisposition. That, of course, simply means we have many opportunities to learn, and this project presented us with the perfect opportunity to become acquainted with one of the electronics tinkerers'

VRC INTERFACER.



VRC BREADBOARD.



VRC Main Progress.

best friends — a breadboard.

The breadboard made a cameo appearance in our article about the Microbric Viper, but that was a simpler circuit than the one required for the voice recognition module. A proper introduction was in order.

Breadboards are great tools to prototype circuits. They are a solderless board similar in spirit to a printed circuit board, but with the only requirement for an electrical connection being placement in one of the various pinholes on the board. After taking a look at a clear breadboard and then a completed prototype circuit, they can seem like complicated and intimidating devices, but the learning curve is pleasantly gradual.

At the top of the breadboard there are three pins — one power and two grounds (in case you need a common ground and another isolated ground). Below the pins is the meat of the breadboard — a board with a myriad of pinholes for transistors, resistors, and ICs galore. All you need to do to integrate a component into your circuit is to insert it into the board and make the requisite connections demanded by the schematic in front of you or in your brain.

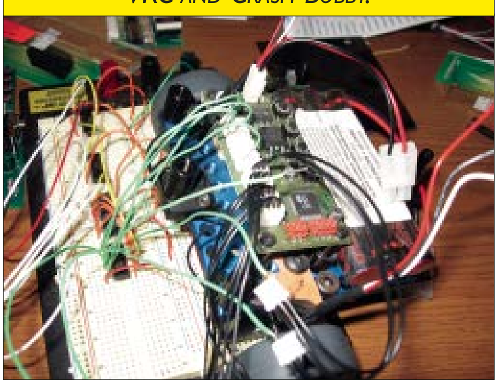
One thing that all circuits need is power. To get power to your bits, all you need to do it run a wire between the top power pole and one of the bus strips running vertically down the board. Also run a wire between the ground pole and the bus strip. The power should go to the red line, and the ground should go to the blue line. Now you are ready for the real fun.

Sandwiched between the bus strips on the breadboard are the pinholes where you can connect your electronic bits. The circuit we needed to construct used a variety of integrated circuits, and placement of ICs is very straightforward. There is a groove down the middle of the pinhole region of the breadboard that isolates the two sides of the region. ICs are simply placed lengthwise down the groove.

On either side of the groove there are rows of five pinholes each. One pinhole of each row has just been taken up by the component you placed, but that still leaves four pinholes. Now, anything you need to connect to your component can go in any of the four adjacent holes. It's that easy!

We were pleasantly surprised with how easy the breadboard was to use. It can get to be a bit messy once webs of wires run between all of your components, but that doesn't detract too much from its amazing usefulness as a prototyping tool. Now that we were formally introduced, we felt comfortable enough to use the breadboard to prototype the interface circuit for the voice recognition module.

VRC and Crash Bobby.



Voice Messages

The schematic included in the instruction manual may have been vague, but it provided a manageable starting point that mercifully included the rationale behind the circuit. The interface circuit would replace the display board on the 10 pin header, and the basic idea was that the action of flashing a number on the displays could be translated into logic highs and lows readable by another device, like a robot. The minimum requirements for the interface circuit would chop the vocabulary of the circuit down to 10 words, but the inclusion of a few more transistors and flip flops would boost the vocabulary back up to the normal level.

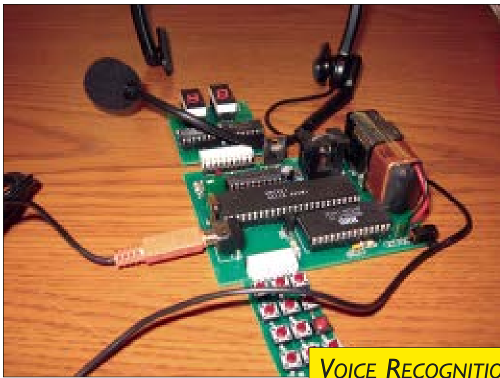
The schematic calls for a number of NAND and OR gates, and we chose to use 7400 and 7432 ICs instead. Each IC contained four of the desired gates, so we only had to use three pins from each IC. After we sorted out our components, there was one other item that needed to be addressed before we could dive into constructing the circuit. The three PCBs in the kit are attached to each other via headers and sockets, with nothing provided for an interface circuit. We had to come up with our own socket, but it was no great chore (except for the fact that we didn't have the ideal wire crimpers on hand, but that ordeal shouldn't befall any better prepared tinkerer). Now we were ready to prototype our circuit.

By following the schematic in the instruction manual, we came up with a simple circuit populated by four ICs — our NAND and OR gate replacements (the 7400 and 7432, respectively) and the two other ICs specifically called for by the schematic (a 74LS373 and a 4028). We thought our prototype wasn't totally hideous by breadboard standards, and it was certainly enough to test the ability of the interface circuit to connect the voice recognition module to the type of external device that we think *SERVO* readers would like to see — a robot!

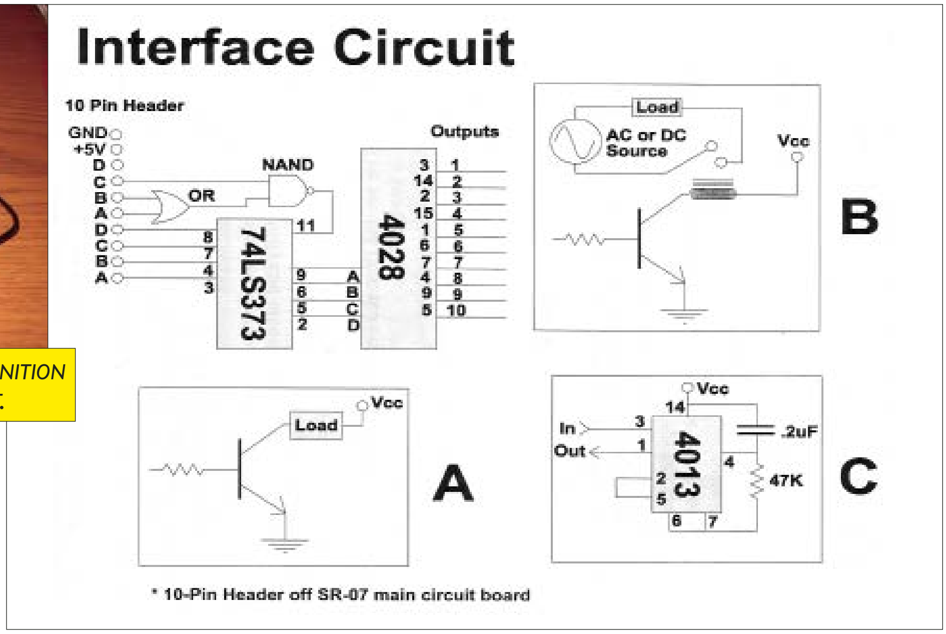
Most robot kits contain some kind of open port or some other similar feature to encourage hacking and modifications. We possessed many such kits from previous projects, but one that we thought particularly suited to the task at hand was Crash Bobby from German company qfix.

Heard Animals

Our previous adventure with Crash Bobby saw us outfit the bot with a custom sensor suite, complete with touch sensors and a line-following light sensor. Bobby took to the additions



VOICE RECOGNITION CIRCUIT.



naturally, so we thought it would be the perfect candidate to hook up to the voice recognition module.

One of the last issues we had to deal with to get the interface circuit (and through that the voice recognition module) attached to Crash Bobby was sorting out the right connectors. This is an important consideration for tinkerers, because in our experience, it's kind of like choosing the right pair of shoes. Various different robots use various different connectors — sometimes the same style will suit a few robots, but others require a different size or something completely different. If a tinkerer wanted to create a general connector that would suit a variety of bots, they would have to do a bit of homework, but since we had Bobby in front of us, our job was a bit easier.

Crash Bobby sported a series of three pin connectors. In our previous efforts with Bobby, we used super cool mil spec connectors that fit the pins perfectly, but we did not have any such connectors at our disposal this time. Instead we decided to go the simplest way possible and use Bobby's existing connectors.

We disconnected the wires from the infrared sensors on Bobby so we could use them for the interface circuit. After a quick diagnosis with the multimeter, we were able to determine which pin was the signal pin for each connector, and all that it took to connect each output from the interface circuit to Bobby was to insert an output wire into the socket hole corresponding to the signal pin. The wires that we used for the breadboard were not stranded, and the solid wire made the connection easily because they were most fortuitously the perfect size to fit into the socket.

While connecting the interface circuit to Crash Bobby, we came upon a problem that was obvious in hindsight, but not one that we had thought of before happening upon it. The interface circuit provides 10 outputs, so a receiving device would need to provide 10 inputs to be wholly compatible. We're confident that wouldn't be an issue with most devices, but for many robots, 10 free inputs is quite a tall order. We were fortunate with Bobby in that we had a few sensors that could stand to be disconnected (Bobby doesn't need eyes when we're giving him orders), but that might not be the case with a lot of bots.

Many kits are put together with the intention of being hackable and expandable platforms, but the range of that expandability can go from one or two available hacker ports to a battalion of free pins. Even with all of his other sensors disconnected, Bobby only had eight free inputs, and that, in our experience, is not on the lean side.

That doesn't mean our endeavors with Bobby were completely foiled. The two inputs we were shy only meant a reduced vocabulary for the circuit, though we have to use some clever word assignments to sidestep the vacancies.

Our last concern with the voice recognition module was that it was somewhat bulky and a bit cumbersome to handle. Without an interface circuit, the keypad and display board were only attached to the main board by the headers and sockets, and one needs to be careful in particular with the sockets. They are very vulnerable to any bending at the thin solder joints, so caution needs to be taken when picking up and moving the module.

A complete interface circuit could be wired up very cleanly and compactly on some perf board, and that would at least be a bit less cumbersome than the entire display board. Also, the keypad can be removed after teaching the circuit its vocabulary, making the circuit a bit more manageable. The interface circuit and main board alone are more akin to the compactness associated with a module, but it still isn't a very easy component to physically incorporate into a device like a robot.

The main board itself doesn't really contain any extra holes meant for mounting to any external device, and the size of the module could be a major factor in deciding whether or not a certain robot kit could even be outfitted with the module.

All this is simply stuff to think about when scheming about what to

Recommended Websites

For more information, go to:

www.imagesco.com

www.darpa.mil/ipto/programs/gale/index.htm

attach your module to. It might be a bit of a hassle, but nobody said getting your robot to recognize speech was going to be a cake walk.

Babble

The sensor inputs we used to connect Bobby to the interface circuit were a wise choice because it would simplify programming. The ports were already seeing logic highs and lows from the existing sensors, so we could base the bot's programmed responses very closely on Bobby's existing commands. All we needed to do was engage in some careful accounting of what word assignments corresponded to what inputs in the bot, and we would have a robot reacting to commands like "right," "left," and "attack."

Overall, the voice recognition circuit from Images SI is an ambitious project that encourages expansion and experimentation even though it won't coddle you through the process. Eager tinkerers that don't want to go through the hassle of constructing their own circuits can order plenty of parts from the Images SI website, but intrepid do-it-yourselfers are also given the means to strike out on their own. The detailed discussions about how to increase the robustness of the circuit were a pleasant surprise in the instruction manual, and they are a good way to galvanize the imagination of any tinkerer suffering from builder's block.

The voice recognition module is an adequately accessible effort to spread the word about a technology that is a hot topic that many of the upper echelons of engineering are talking about. There are annual conferences dedicated to furthering voice recognition technology, and the Defense Advanced Research Projects Agency even funds an annual voice recognition technology competition in the same spirit as the DARPA Grand Challenge.

The GALE (Global Autonomous Language Exploitation) Program seeks to create technology capable of recognizing and translating large volumes of speech in multiple languages and dialects. Getting a computer to recognize clearly articulated words is difficult enough, but creating something so versatile as to adapt to different languages, dialects, and even just nuances in inflection is certainly a challenge on par with driverless cars navigating deserts or urban environments.

This type of technology has the potential to literally save lives, so the humble circuit from Images SI is in good company. Join the conversation! **SV**

EVENTS CALENDAR

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

July

1-10 RoboCup Robot Soccer World Cup
Atlanta, GA
All the usual soccer events: small, mid, humanoid, and AIBO. Also a NIST rescue robot contest. In addition to these events, the RobotCup@Home competition will be held in conjunction with the World Cup again this year.
www.robocup.org

10-13 Botball National Tournament
Norman, OK
Teams compete with autonomous robots built from standardized kits. The contest involves moving black and white balls on a game board.
www.botball.org

11-15 AUVS International Undersea Robotics Competition
US Navy TRANSDEC, San Diego, CA
Autonomous underwater robots must complete a course with various requirements that change each year.
www.auvsi.org/competitions/water.cfm

16-20 K'NEX K*bot World Championships
Las Vegas, NV
Includes three events: Two-wheel drive K*bots (autonomous), Four-wheel drive K*bots (autonomous), and Cyber K*bot Division (R/C).
www.livingjungle.com

21-22 War-Bots Xtreme
Saskatoon Saskatchewan, Canada
Radio-controlled vehicles destroy each other Canadian-style.
www.warbotsxtreme.com

22-26 AAAI Mobile Robot Competition
Vancouver, British Columbia, Canada
This long-standing competition for autonomous robots includes some interesting events this year such as the Semantic Robot Vision Challenge, which is sort of a scavenger hunt for robots. Your robot will be given a list of objects which they must locate and recognize. Then there's the Human-Robot Interaction Challenge, the Integration Challenge, and a robot exhibition.
www.aaai.org/Conferences/National

23-27 AUVS International Aerial Robotics Competition
US Army Soldier Battle Lab, Fort Benning, GA
In this event, flying robots are required to complete a fully autonomous ingress of 3 km to an urban area, locate a particular structure from among many, identify all of the true openings in the correct structure, fly in or send in a sensor that can find one of three targets and relay video or still photographs back 3 km to the origin in under 15 minutes. And that's just one of three scenarios!
http://avdil.gtri.gatech.edu/AUVS/IARCLaunchPoint.html

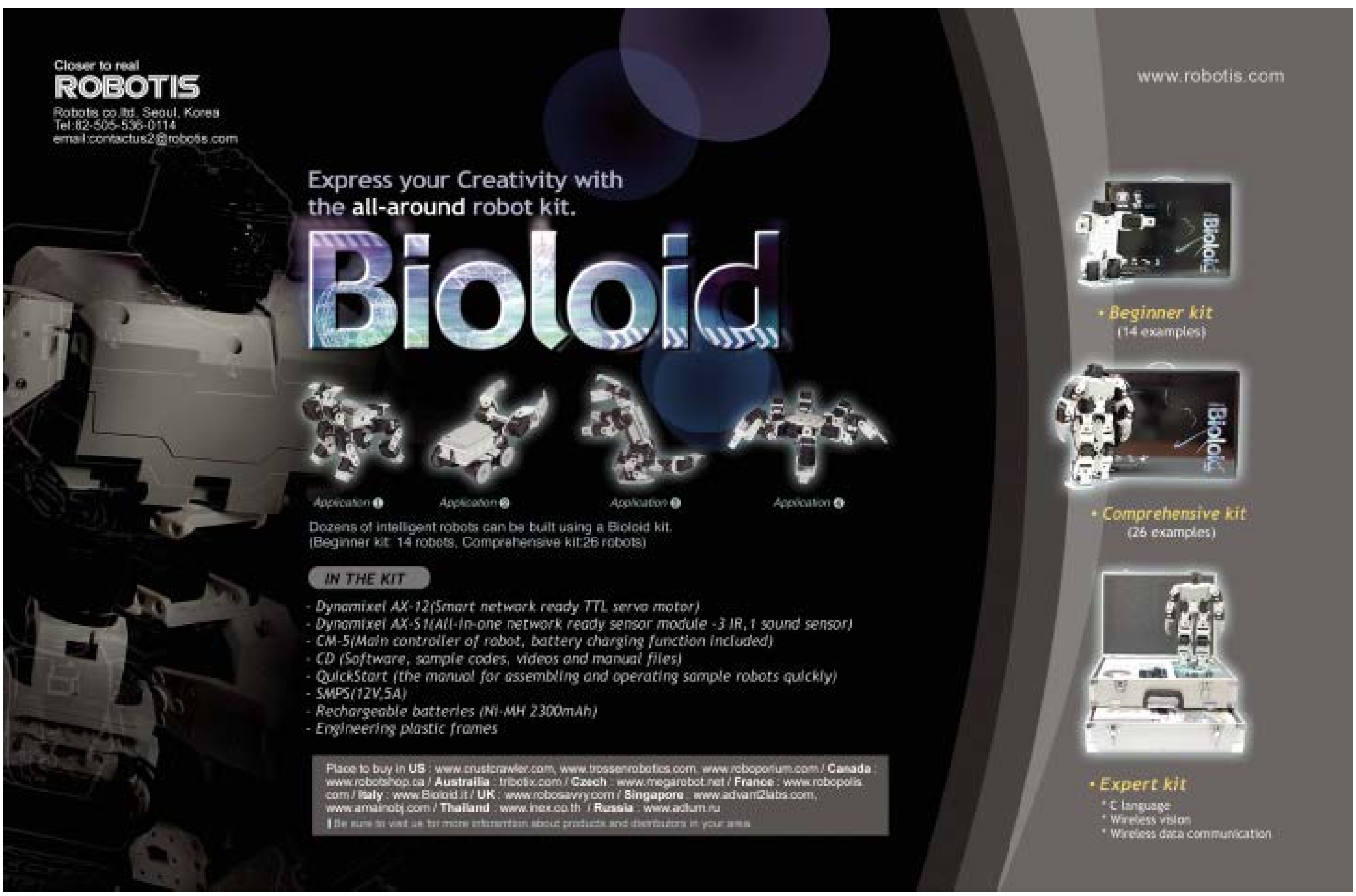
August

19 RoboCountry
Takamtsu City, Kagawa, Japan
Remote-control humanoid robots combat.
www.robocountry4.com

September

3 DragonCon Robot Battles
Atlanta, GA
Remote-control vehicles destroy each other at a well-known Atlanta Science Fiction convention.
www.dragoncon.org


SERVO Magazine on CD!
2006 CD-Rom Now Available!
Great news! Now you can own the 3rd published year of SERVO on CD. Stored as PDFs, you can print, search, and easily store your copies on digital media.
You can purchase said CD from us, for your own personal use, and donate your old paper copies to your local school or library. (Your spouse will love you for it!)
This makes a great gift, too! Give it to a friend or loved one to show how much you care!
$24.95
plus $5.06 s/h USD
To order, go to:
www.servomagazine.com
or call: 1.800.783.4624


Closer to real
ROBOTIS
Robotis co.ltd. Seoul, Korea
Tel:82-505-536-0114
email:contactus2@robotis.com
www.robotis.com
Express your Creativity with the all-around robot kit.
Bioloid
Application 1
Application 2
Application 3
Application 4
Dozens of intelligent robots can be built using a Bioloid kit.
(Beginner kit: 14 robots, Comprehensive kit:26 robots)
IN THE KIT
- Dynamixel AX-12(Smart network ready TTL servo motor)
- Dynamixel AX-S1(All-in-one network ready sensor module -3 IR,1 sound sensor)
- CM-5(Main controller of robot, battery charging function included)
- CD (Software, sample codes, videos and manual files)
- QuickStart (the manual for assembling and operating sample robots quickly)
- SMPS(12V,5A)
- Rechargeable batteries (Ni-MH 2300mAh)
- Engineering plastic frames
Place to buy in US : www.crustcrawler.com, www.trossenrobotics.com, www.roboporium.com / Canada : www.robotshop.ca / Australia : tribotix.com / Czech : www.megarobot.net / France : www.robopolis.com / Italy : www.Bioloid.it / UK : www.robosavvy.com / Singapore : www.advant2labs.com, www.amainobj.com / Thailand : www.inex.co.th / Russia : www.adlum.ru
• Beginner kit
(14 examples)
• Comprehensive kit
(26 examples)
• Expert kit
* C language
* Wireless vision
* Wireless data communication

NEW PRODUCTS

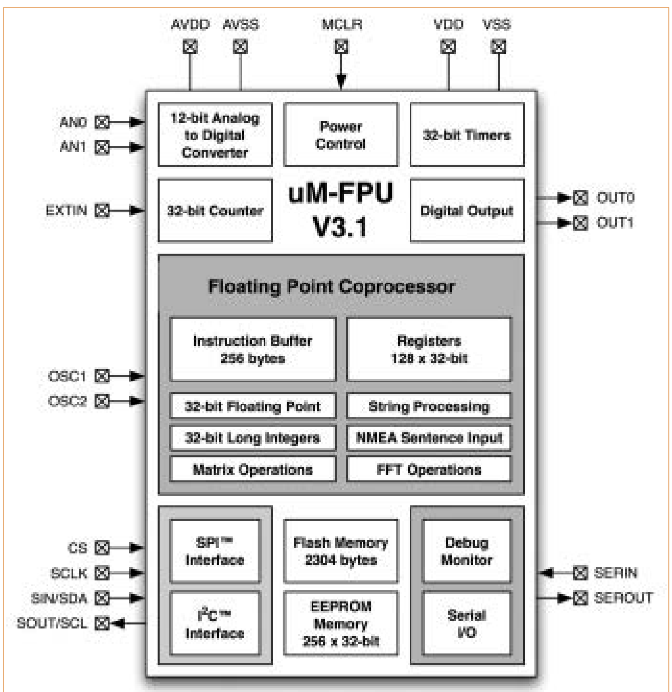
CONTROLLERS & PROCESSORS

uM-FPU V3.1 Floating Point Coprocessor

Micromega Corporation announces the release of the uM-FPU V3.1 Floating Point Coprocessor chip. The new chip extends the powerful feature set of the original uM-FPU V3 chip to include serial I/O support, NMEA sentence parsing, block transfers, additional matrix operations and string support, and many other enhancements.

The new serial I/O capabilities with NMEA sentence parsing make it easy to add GPS data to embedded system designs. GPS data can be read and processed directly by the uM-FPU V3.1 chip, saving I/O pins, memory space, and execution time on the microcontroller, which can then be used for the main application. As an added benefit, GPS data is immediately available on the uM-FPU V3.1 chip for further navigational calculations using the powerful floating point instruction set.

The uM-FPU V3.1 chip interfaces to virtually any microcontroller using an SPI interface or I^{2}C interface, making it ideal for microcontroller applications requiring floating point math, including GPS, sensor readings, robotic control, data transformations, and other embedded control applications.



The uM-FPU V3.1 chip supports 32-bit IEEE 754 compatible floating point and 32-bit integer operations. Advanced instructions are provided for fast data transfer, matrix operations, multiply and accumulate, FFT calculations, serial I/O, NMEA sentence parsing, and string handling. The chip also provides two 12-bit A/D channels, two digital outputs, an external event counter, Flash and EEPROM storage, and serial I/O up to 115,200 baud.

The uM-FPU V3 IDE (Integrated Development Environment) makes it easy to create, debug, and test floating point code. The IDE code generator takes traditional math expressions and automatically produces uM-FPU V3.1 code targeted for any one of the many microcontrollers and compilers supported. The IDE also supports code debugging and programming user-defined functions. User-defined functions can be stored in Flash using the IDE, or stored in EEPROM at run-time. Nested calls and conditional execution are supported. User-defined functions can provide significant speed improvements and reduce code space on the microcontroller.

The uM-FPU V3.1 chip is RoHS compliant and operates from a 2.7V, 3.3V, or 5V supply with power saving modes available. SPI interface speeds up to 15 MHz and I^{2}C interface speeds up to 400 kHz are supported. The chip is available in an 18-pin DIP, SOIC-18, or QFN-44 package. The single unit price is $19.95 with volume discounts available.

For further information, please contact:

Micromega Corporation
1664 St. Lawrence Ave.
Kingston, ON K7L 4V1 CANADA
Tel: **613 • 547 • 5193**
Website: **www.micromegacorp.com**

ELECTRONICS

Adjustable DC-DC Converters

AnyVolt Micro is the latest in Dimension Engineering's line of adjustable DC-DC converters. AnyVolt Micro is the successor to the popular AnyVolt Mini, adding thermal and overcurrent protection while simultaneously reducing size and weight.

AnyVolt Micro can take an input between 2.6V and 14V and convert it into another voltage between 2.6V and 14V. You choose the output voltage you want by adjusting the onboard potentiometer with a screwdriver.

The AnyVolt series of DC-DC converters is unique in that it allows you to step voltage up or down — effectively eliminating the problem of a drop-out voltage. For example, if you have a project you are powering with four Alkaline AA batteries and you need a regulated 5V source, AnyVolt Micro can operate across the battery pack's 4V-6V operating lifespan and give a constant 5V output. It is also a great choice for stepping up voltage from two AA batteries.

Currents of up to 0.5A can be drawn from the device — the exact limit will depend on your input/output voltage needs. The product's datasheet has a handy reference table showing the current limits at various input and output voltages.

AnyVolt Micro retails for $19.99 and is available from the Dimension Engineering website.

For further information, please contact:

Dimension Engineering — Website: **www.dimensionengineering.com**

PARTS & MATERIALS

Profiles, Fasteners, and Accessories for 80/20® and Non-80/20® T-Slot Profiles

Get hundreds of new parts for T-slotted aluminum framing at your fingertips. 80/20® announces the release of its second T-Slot Parts catalog, T-Slot Parts Supplement 2. Supplement 2 offers a new line of fasteners, connectors, joining plates, hinges, floor-to-frame, and other accessories that fit 80/20's 10, 15, 25, and 40 Series T-slotted aluminum profiles.

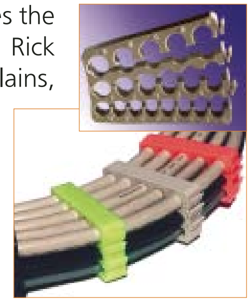
T-Slot Parts by 80/20 offers new modular solutions for applications involving automation, machine-framing or guarding, material handling, workstations, or safety.

For more information on T-Slot Parts Supplement 2, or to request your free catalog, visit **www.tslotparts.com** or contact:

80/20, Inc.
1701 South 400 East
Columbia City, IN 46725
Tel: **260•248•8030** *or* **877•248•8020**
Fax: **260•248•8029**
Website: **www.8020.net**

WIRING ACCESSORIES

The Ultimate Cord Organizer Clip



Delta 9 Products (DNP) introduces the Ultimate Cord Organizer Clip. Rick Nelson, DNP Product Manager explains, "Our product provides an innovative way for the professional to organize and track cables, cords, and wires between electronic devices." "Each slot in the Ultimate Cord Organizer Clip has a letter assigned to it and retains the cables and cords when open. The Ultimate Cord Organizer sorts by size and type, but also allows you to channel, isolate, and track cords, cables, and wires. The organizer is available in four standard colors: black, gray, neon orange, and neon green. Custom colors are available. The organizer is sold in five packs ($9.95), 10 packs ($18.90), 20 packs ($35.90), and 60-pack ($102.35) quantities.

For further information, please contact:

Delta 9 Products — Tel: **530•333•2014** Website: **www.ultimatecordorganizer.com**

COMBAT ZONE

Featured This Month

PARTICIPATION

Safety — Situation Awareness

● by Kevin Berry

One of the common hazards of the Third Millennium is driving while talking on a cell phone. We've all done it — arriving at our destination during a phone call, and having no recollection of how we got there! Well, many injuries in the pits and at events happen for the same reason: lack of situational awareness.

In the excitement of arriving, hauling, unpacking, setting up, charging batteries, registering, and chatting with long-unseen buddies, it's easy to forget to look at what you're doing. Tripping, head bumping, and finger mashing opportunities abound in the hectic environment of the pits. You may be working with familiar tools, on familiar bots, doing familiar tasks, but you aren't in your shop at home.

Then, there's turning around damaged bots. You are frantically whacking away at bent metal, grinding, cutting, rewiring, and drilling, trying to get back in the game. Tools get dropped where they are used, scrap materials pile up, and meanwhile, 20 other maniacs are doing the same thing all around you. Focusing on safety, what's on the ground, what's happening at tables around you, and other builders antics is a low, low priority.

How does a builder mitigate this dangerous tendency? First, get your dang bots built at home, and bring them ready to fight! Second, arrive early, and scope out the venue. Where are fire extinguishers? Emergency exits? Power drops? Set up your pit neatly, and keep it that way. One sure way to overload yourself — and your senses — is bringing too many bots to an event. I've found it's more enjoyable to fight fewer bots, and have more time to fix them, socialize, and watch other fights.

Third, and most important, is to just relax. Remind your teammates and opponents that this is supposed to be a fun activity, not a stress test! SV

Family Korner: Tim and Karl Wolter

● by Kevin Berry

As a new feature in our "Participating" category, Combat Zone will be showcasing families that have benefited from our sport. This month, Tim Wolter shares his experiences as a robot combat participant, organizer, and mentor to students along with his son, Karl.

If your family builds and fights bots, and you've got a story to tell, email a summary to the editor at legendaryrobotics@gmail.com, and we'll select some to publish in future issues. — Kevin Berry

Tim Wolter writes for Combat Zone on a regular basis, organizes building classes and tournaments at a middle school, fathers Karl, and spends a bit of time at his day job as a medical doctor. Here's Tim's story:

Professor Richard, the Mechwars Safety Czar, is a good friend of mine. I once mentioned to him that it was not easy being the parent of a kid like my son, Karl. "You mean, Scary Smart?" He understood perfectly.

There are some kids who just don't fit into the conventional world very well. A child whose first word was "Broken," and who insisted on being taught how to weld when he turned 13 is not going to be terribly interested in how the football team is doing. Oh, sure, we tried the normal father-son activities, but what do you do with a five year old who decides to run the basepaths the wrong way, then makes a very convincing argument that doing so makes just as much sense?

You put away the bat and glove and get out the wrenches.

Tim and Karl Wolter.

Karl and I have been building together for six years now. Our early projects were crude, what I call "monkeytech," but still crowd favorites due to our willingness to put entertainment value ahead of our won-lost record. The legacy of Team West Hill Robotics will likely be the first combat uses of Spam, flaming TeleTubby dolls, and the messy yet frighteningly effective deployment of a specially hardened Christmas fruit cake.

Other family members have had cameo roles now and then, especially when we did a build for Dragonfly TV, a PBS science show. This was a fun project, and preserves for posterity the image of a scrawny middle schooler enthusiastically boasting that "I won't stop until I am the most feared seventh grader in robotic combat."

Karl accomplished that, and much more. He won the state Precision Machining competition. He helped me teach a very successful and ongoing robotics class. He is the only summer camp maintenance lackey ever to report for duty with his own MIG welder. He is clearly the superior craftsman in our joint projects. I am mainly along for financing and comic relief.

Some kids just have to follow their own path, no matter how odd a journey it may seem to the unimaginative world. Karl will end up doing something fascinating, but neither he nor I have the faintest clue what it will be.

So if you are, or if you have, a kid like Karl, my advice is to defy conventions. Pursue whatever odd dream draws you. It really never is too early to learn things.

Nor is it ever too late. Karl is now teaching me how to weld.

Editor's note: Karl is also now writing for Combat Zone! **SV**

THE BUILDING OF A 30 LB. SPINNER

● by Brian Benson

Every robot design begins with a set of requirements; in industry these are often called design specifications. Throughout my years of combat robot building, I have learned that most successful robot designs incorporate a number of traits. These traits, roughly in order of importance, are as follows: reliability, ease of repair, invertability (able to drive upside down), adequate power, and flexibility for different scenarios.

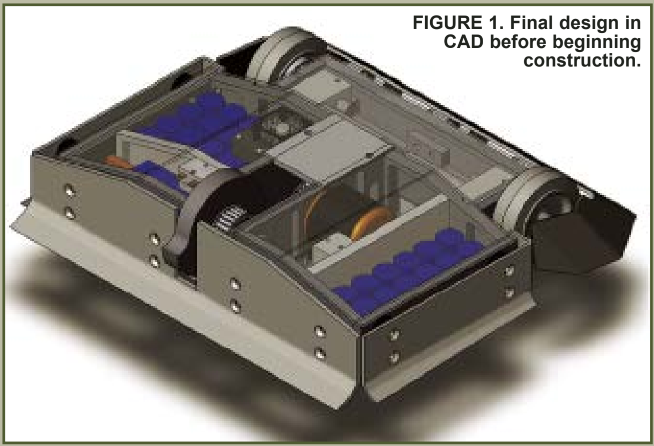


FIGURE 1. Final design in CAD before beginning construction.

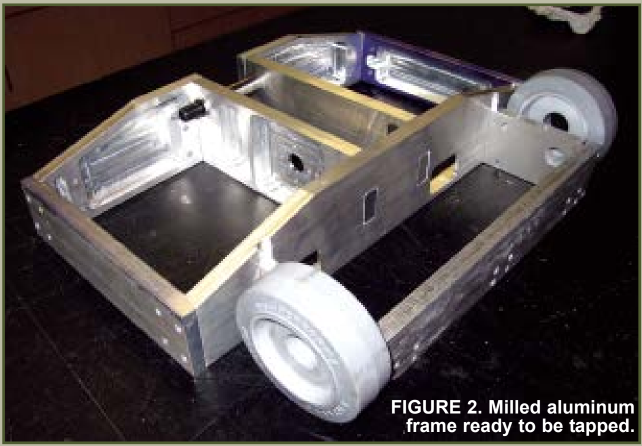


FIGURE 2. Milled aluminum frame ready to be tapped.

My design process began with these requirements in mind, along with a few others that were specific to my goals. The robot would be 30 lbs., because that was the highest weight class that local competitions allow. The robot would have a vertical spinning weapon to toss and break other robots because I had never built that type of robot before and it looked fun. Lastly the design would be balanced, so that the weapon, drive, and armor were equal in their ability to perform.

The Design

With all of these factors in mind, I played with various design concepts in a CAD (computer aided design) software package named SolidWorks until I decided on one that met all of my specifications. As can be seen in Figure 1, the design incorporates a two wheeled drive train, a milled aluminum base frame, and a hinged titanium rear wedge to deflect spinners and protect the wheels. The weapon is a one tooth blade that maximizes the RPM but minimizes the number of hits per revolution which, in turn, maximizes the amount of tooth bite.

The armor is rubber shock mounted steel to minimize shock to the base frame while making it easy to repair with a hammer and welder at events. The frame is slanted on the top to minimize the size of the robot; the smaller the robot, the less surface area of armor necessary. To meet the requirement of being invertible, the geometry of the robot is such that when upside down the wheels still touch the ground. For the components of the robot, I chose ones that would give me the best power-to-weight ratio.

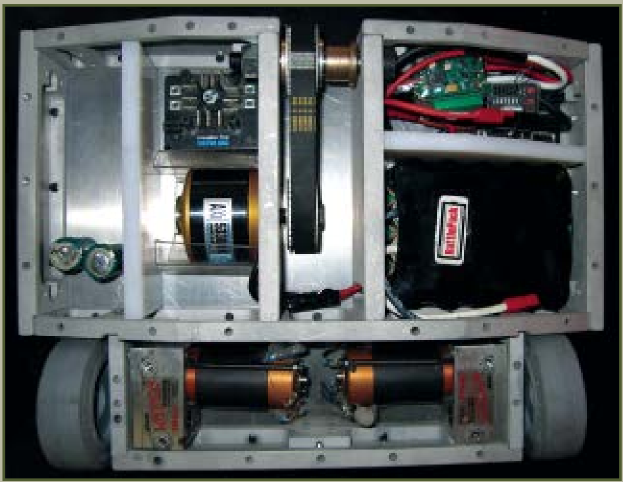
For the drive train, I chose two Astro 940 motors at 14.4 volts coupled with Team Whyachi gearboxes and 3.5" Colson wheels. The weapon motor is an Axi 5330/18 brushless motor run at 24 volts with a 1:1 timing belt reduction to a 3.5 lb blade. This setup gave me a measured blade speed of 5,500 RPM.

For speed controllers, I chose Victor 883s for the drive and a Phoenix 110-HV to control the weapon. This collection of components gave me the greatest amount of performance at minimum weight.

Frame

The frame was fabricated from 1/2" thick aluminum bar stock. The bar stock was cut with a horizontal band saw and then pocketed and shaped using a vertical milling machine. As shown in Figure 2, the frame fits together much like a 3D jigsaw puzzle for maximum strength. The frame is held together with

FIGURE 3. Frame assembled and parts being test-fitted.



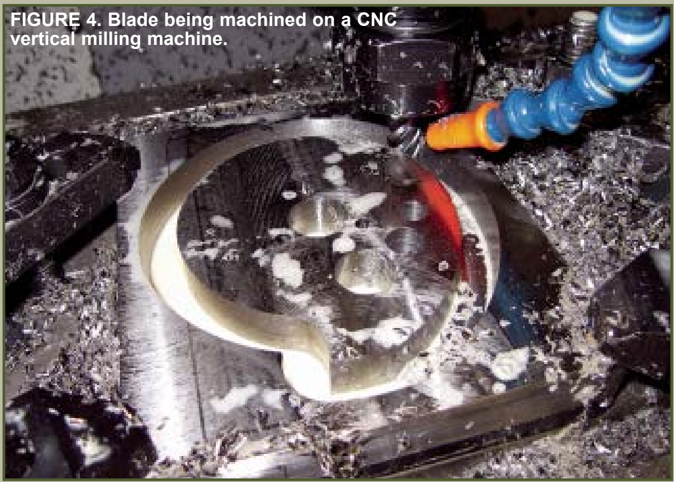


FIGURE 4. Blade being machined on a CNC vertical milling machine.

FIGURE 5. Completed blade ready to do some damage!

FIGURE 6. Test-fitting the steel armor before final fabrication.

countersunk 1/4-20 screws. Figure 2 shows all of the holes drilled and ready to be countersunk or tapped.

After completing all machining on the frame, I sandblasted it to prepare it for painting. Next, I cut out and mounted the base plate, which was 1/16" thick 6061 aluminum. With this done, I could test-fit all of my components to verify that everything would work how it was designed to. Quickly making the plastic spacers, I assembled the frame and put all the parts I had finished into place as shown in Figure 3.

Offense and Defense

With the frame completed, I began to focus on the weapon and the armor. For the weapon, I had acquired a piece of 3/4" S7 tool steel to machine the blade out of. Using a vertical CNC milling machine, the complex blade shape was machined. The milling process can be seen in Figure 4. In Figure 5, the final product is ready to go! The three varying diameter holes act as lightening holes to balance the blade. The three holes that are evenly placed around the center bore transfer the rotational force from the pulley to the timing belt pulley through a set of hardened steel pins. This allowed the weapon shaft to be non-rotating, or a dead shaft, so that if the frame bent it would be less likely to cause the weapon to stop working.

With the weapon system ready to go, I now concentrated on the armor system. This consisted of a 3/16" steel front plate and 1/8" steel side plates. I cut out the steel on a vertical band saw and then milled slots along the lines that I wanted to bend the plates at. Also holes were drilled and countersunk to mount the plates to the rubber shock mounts as shown in Figure 6.

The next part to make was the back wedge. This was cut out of a sheet of 1/8" titanium using a plasma cutter. I made a template by printing out the design onto two sheets of paper and then taped them to the titanium. Using an oxyacetylene torch, the titanium and steel armor was bent at the proper angles. The steel was welded and ground down to give it a smooth curve on the lower section as shown in Figure 7. Figure 7 also shows all of the components mounted in their final position along with the completed wiring. The rear wedge is held on with three custom hinges, each one consisting of six countersunk screws as shown in Figure 8.

Final Details

With all of the components mounted and working, all that was left to finish were some small details.

FIGURE 7. Parts mounted, armor complete, and wiring done.

FIGURE 8. A rear view of the hinged titanium wedge.

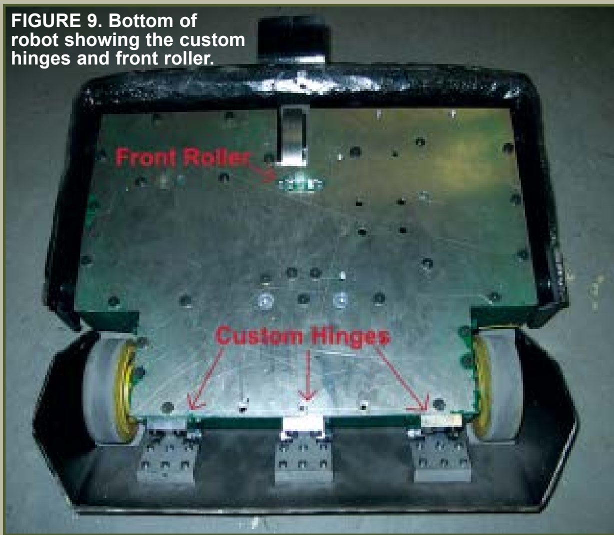


FIGURE 9. Bottom of robot showing the custom hinges and front roller.


FIGURE 10. Billy Bob, ready for action!

First, I needed to create a roller or skid at the front of the robot to allow it to move better. Instead of having the front half drag on the front steel armor, this gave it a smoother, smaller contact patch and a smaller coefficient of friction. The custom hinges and front roller are visible in Figure 9.

With all of the mechanical aspects of the robot complete, I was ready for the most important step — painting! Considering the robot's name was Billy Bob, I decided to give it a yellow and green John Deere theme. In Figure 10, the final robot is complete and ready for battle.

Conclusion

The last step in any engineering process is the evaluation of the final product. This is best done with real world testing. After completing Billy Bob, I competed at one of NERC's (North East Robot Club) annual events — Motorama. During the event, I found three weaknesses in the design. It didn't drive very well because not enough of the weight was on the wheels, so it didn't have enough traction. Also, the blade didn't extend beyond the wedge on the steel armor.

This meant that if I couldn't get the other bot onto the wedge, I couldn't hit them. This, coupled with weak driving, kept the weapon from being as effective as it could have been. Lastly, the weapon wasn't as powerful as I wanted. However, even with these drawbacks I was able to take first place going undefeated through the competition.

Billy Bob will be competing at Robogames with a few new improvements. It will have a bigger, faster blade to increase its maximum kinetic energy and reach. I will also be installing magnets under the wheels to increase the down force so that it has enough traction to move around effectively.

Overall, Billy Bob was a success, and with a few modifications, I expect it will meet all of the requirements that I had set in the initial design. For more information on Billy Bob and Robotic Hobbies, visit **www.robotic-hobbies.com**. SV

EVENTS

RESULTS — April 15th - May 13th

Carolina Combat was held May 5th in Greensboro, NC, sponsored by Carolina Combat Robots.

Results are as follows:

- *Antweight (1 lb)* — 1st: "Unithorn," thwack, Team Thorn.
- *Beetleweight (3 lb)* — 1st: "Pure Dead Brilliant," Spinner, Team Rolling Thunder.
- *College Events* — Obstacle course, Tug of War, Sumo, Combat, all won by Central Carolina Community College.
- *Hobbyweight (12 lb)* — 1st: "CheepShot 3.0," Wedge, Team Rolling Thunder.
- *Featherweight (30 lb)* — 1st: "Totally Offensive," Spinner, Team Mad Overlord.
- *Lightweight (60 lb)* — 1st: "CrockBot," Wedge, Team Gator. SV

A Powerful Drive Train Solution

● by Matt and Wendy Maxham

We believe the most important system on a combat robot is the drive system. You have to be able to reliably get the power from the motors on to the arena floor. In our robots, we aim for about 12 miles per hour — fast enough to get across the arena before a spinner has reached their full potential, but not so fast that you can't control the robot.

Motors normally output 3,000-6,000 RPMs. To reach the 12 MPH goal, the motor RPM needs to be reduced from the high speed/low torque RPMs to a low speed/high torque RPMs.

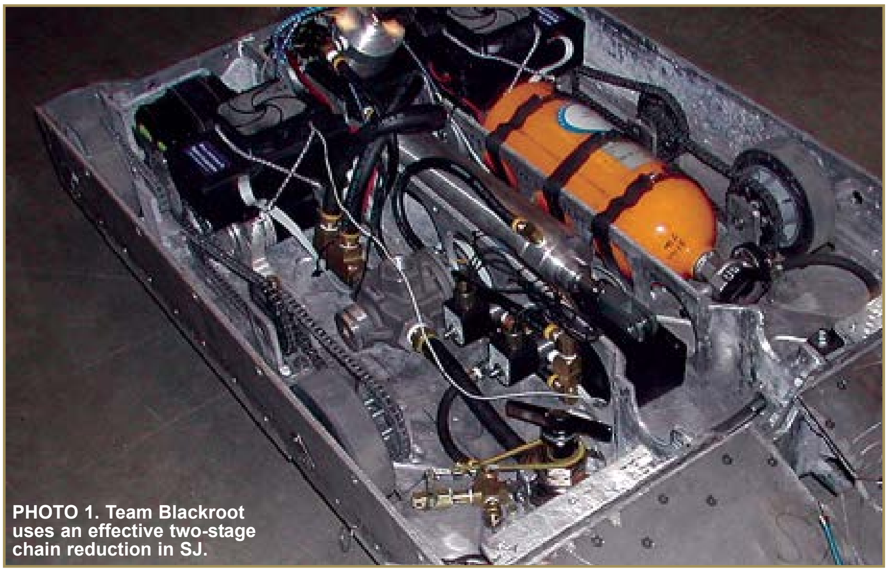


PHOTO 1. Team Blackroot uses an effective two-stage chain reduction in SJ.

MPH = Motor RPM/ Gear Reduction * Tire Dimension (inches)/336

There are a variety of ways to get the gear reduction job done. Chain reduction is relatively inexpensive, simple, and efficient (Photo 1). To achieve the necessary reduction with chain, you either need to use multiple stages or a large sprocket size differential. Both methods take up valuable space and can be unreliable in the extreme conditions of combat.

Another popular method is a direct drive motor/gearbox combo like a wheelchair motor. These plug-and-play components give you the tire, motor, and useable gear reduction in one ready-to-mount package (Photo 2). One drawback of these systems is the proximity of the tire to the motor/gearbox combo can limit the flexibility of your robot design. You have one drive tire per gearbox/motor, which means for a four wheel drive system, you need four motor/gearbox combos.

Another issue is that wheelchair motors aren't designed for the abuse that is dished out in combat robotics. Direct hits to the tires transmit shock directly to the gearboxes, which can disable your drive, say, if they explode.

After researching the options available for drive systems, we chose a gearbox reduction to a chain driven live axle. This method allows some design flexibility since you don't have to have the motors exactly where the tires are. Plus, you can drive multiple axles from one motor. It also gives you component flexibility with a variety of manufacturers to choose from.

Ultimately, we chose to combine a S28-400 Magmotor with an Apex Dynamics AB60 gearbox for our drive train in heavyweight Sewer Snake and middleweight Devil's Plunger

PHOTO 3. Middleweight Devil's Plunger carries superheavyweight Ziggy around the arena using the same drive system as Sewer Snake.

PHOTO 2. Tombstone protects his NPC drive system with 97 pounds of spinning tool steel! Photo courtesy of Hardcore Robotics.

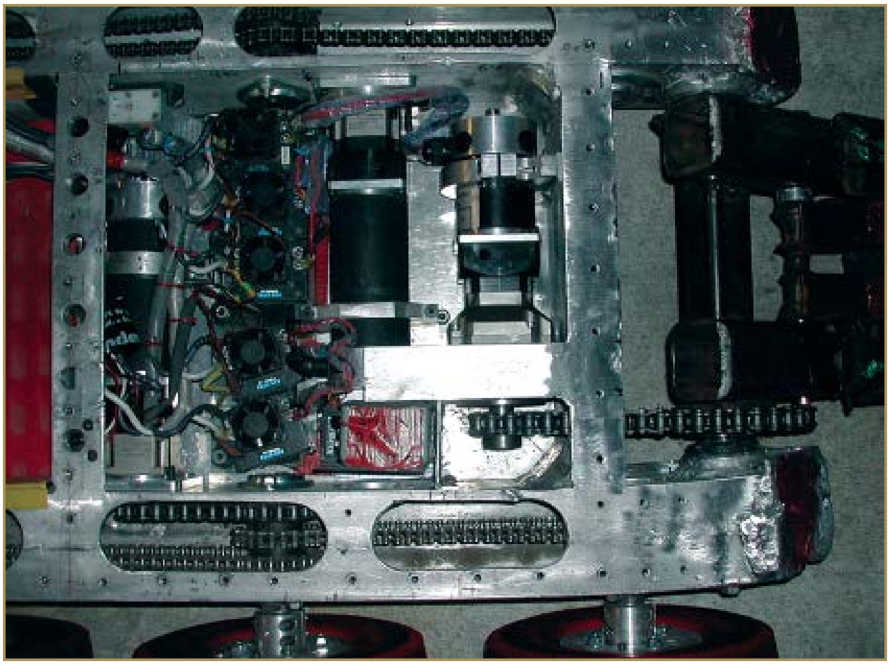


PHOTO 4. Component layout flexibility ... drive motors can be placed where space is available.

(Photo 3). Sewer Snake is a six-wheel drive robot driven by one Magmotor/Apex gearbox combo per side (Photo 4). The 5:1 ratio gearbox chain drives the center axle which, in turn, chain drives the front and back axles. With 10 inch tires and approximately 4,500 output RPM from the motor under load, Sewer Snake would run at 27 MPH with just the gearbox reduction.

One of the benefits of this drive system is being able to fine tune the MPH through the chain system. We run an additional 2:1 reduction from the gearbox to the center axle, bringing the speed down to just over 13 MPH.

Another benefit of this combination of gearbox and chain drive train is the more expensive components (gearbox and motor) are buffered from the destructive forces of robot combat (Photo 5). Sewer Snake have taken some nasty tire hits — bending axles and even breaking a couple of sprockets, but we haven't broken a gearbox since our second fight back in 2002 while using a 10:1 ratio gearbox in the first version of Sewer Snake. Never use a 10:1 planetary gearbox in a high shock application – it is the weakest ratio with a very small sun gear that can fail under extreme torque loads.

One of the reasons we chose Apex Dynamics gearboxes was that they feature helical gears (Photo 6). Helical gears mesh better, increasing the tooth-to-tooth contact ratio by over 33% vs. spur gearing, according to Apex's specs. What does that really mean?

More tooth-to-tooth contact means the motor power gets transferred more efficiently and effectively. The 60 mm Apex gearbox had a better torque rating than the larger NEMA34 gearboxes we used previously, so we were able to shed over 10 pounds on the drive system. Since switching to Apex gearboxes, we have yet to destroy one — and we use them HARD!

The main drawback of the Magmotor/Apex gearhead combo is cost. With one S28-400 Magmotor and one Apex AB60 gearbox per side, you're looking at about $2,000 in drive components.

Although the initial component cost is high, the durability and longevity of these components can make the overall cost less over time. When your components can survive devastating hits from robots like Megabyte and Shrederator, they can save you money in the long run.

While your final choice of drive system will depend on your overall design and budget, this system has proven successful in several top-ranked robots. **SV**

Links

www.TeamPlumbCrazy.com
www.ApexDynamicsUSA.com
www.RobotMarketplace.com

PHOTO 5. Damage to drive tires and axles was extensive and frame was retired ... but motors and gearboxes were fine!

PHOTO 6. Cutaway view of an Apex gearbox showing the helical gears.

PRODUCT REVIEW — GWS Pico Receiver

● by Chad New

One of the most important parts inside any robot is the radio receiver. Without the receiver, even if everything else is working 100%, your robot will do nothing but sit in the same spot. Perhaps it will glow an LED if you have wired one into the robot. A receiver performs the same basic task no matter what type you use or what it is in. Its job is to take the commands which you send remotely from your transmitter and send them to the equipment inside the robot doing whatever you so desire.

When you are building a robot, you must take everything into account when you come up with your design. If you fail to remember all the different components, you may end up with it being over weight or, there may not be enough room inside the robot for the parts; this is especially important with insect robots (150g, 1lb., and 3lb.) where space and weight are at a premium.

When you choose a receiver for an insect, you want to choose one which is small, light, and tough enough to withstand the extreme forces which the current generations of robots are able to produce. Team Wazio uses the four channel GWS Pico mini receiver almost exclusively in all our insect robots and have had very good experiences with them. They are small, light, reliable, cheap (usually less than $20), and glitch a lot less compared to some other mini receivers which I have used.

The GWS receivers are available with many different options including a positive and negative shift which makes them compatible with any 72 MHz radio that you might have. They also have the option of horizontal or vertical pins which can be very helpful when designing a robot. Most robots use only three channels, however, if you need more you can also purchase ones which have six or eight channels for whatever creation you can dream.

Another good aspect of the GWS Pico is that they use a mini crystal which also helps keep the size and weight down inside the robot. The GWS Pico is all about small size and weight which is something that all builders can agree is very convenient.

Despite all the positive points that the GWS Pico has, it also has some drawbacks which one should take into account. The lack of a hard outer shell makes them somewhat vulnerable to debris entering inside and also to damage coming directly from other robots. I have noticed that with strong impacts, the crystals can break from the shock waves sent through the robot.

When you mount your GWS Pico, you need to keep these two things in the front of your mind. Mount them securely with a little padding and make sure they are covered so you won't short circuit anything.

Overall, I believe that the GWS Pico is the cheapest, most reliable, and best option for your insect robot; buy one and put them to the test yourself. SV

EVENTS

UPCOMING: July and August

WBX-IV Bushwacked will take place in Saskatoon, Saskatchwan, Canada 7/21/2007 through 7/22/2007. It will be presented by War-Bots Xtreme. WBX will adopt a new location for this event. A rural setting, 25 minutes south west of Saskatoon, will see combat robots competing for prizes and cash. For more information, visit **www.warbotsxtreme.com**. SV

WBX-IV BUSHWACKED

Build a Bluetooth COMM UNIT

by Fred Eady

As far as we know, whales use sea water as their communications medium. Humans, animals, and most insects use air as their primary means of passing recognizable sounds between each other. What do whales have to do with this month's subject? Absolutely nothing. However — unlike the whale — humans have the ability to communicate intelligently using water, air, light, and magnetic waves. Thus, many of the technological creations offered up by humans that are designed to interface with the human being use the media of light, air, or magnetic wave bending to establish a communications link. You can add water to the list if the Navy is involved.

Will Robinson's robot pal could report its status vocally to any human within earshot. If things got really bad (you know, "Danger, Will Robinson!"), and the robot got whacked, Will's mobile intelligence unit and all of its blinking lights could be attached to the Jupiter 2's robot docking station for diagnostic therapy. In many cases, an alien assisted Will's robot friend but not always with good intentions.

As a real-world builder of things robotic, you know that assembling and programming Will's robot would be akin to designing and implementing the data bank and computer system on the starship Enterprise. Fortunately, there are easier methods of implementing a data-carrying communications link with your electromechanical sidekick.

What's That in Your Ear?

There's a good chance that these days you don't put your cell phone against your head to take and initiate calls. The wired microphone/earphone lashup you used to use is history too. Instead, you probably use a wireless earpiece that is electronically slaved to your cell phone. Ninety-nine percent of the time, that wireless earpiece is based on Bluetooth technology. Bluetooth is ideal for cell phone applications as Bluetooth was designed to eliminate cables over short distances like those between your head and your cell phone. Bluetooth is also capable of maintaining a high-speed data link that rides along with the voice channel it supports.

The typical Bluetooth device operates within the ISM (Industrial Scientific and Medical) band of frequencies at an effective range of up to 10 meters line-of-sight. No license is needed to use the ISM band. Thus, there are a multitude of devices using the ISM frequencies including Wi-Fi devices and microwave ovens. Adaptive frequency hopping technology allows us to employ Bluetooth with minimal interference from competing devices that also use the services of the ISM frequency segment. The key to the Bluetooth frequency hopping scheme is "adaptive," which means the Bluetooth radio will attempt to avoid — or hop around — frequencies that are already in use by other devices.

The main idea behind this article involves using the data transfer portion of Bluetooth technology to open a high-speed asynchronous communications link that can effortlessly deliver data bidirectionally between a robotic device and a PC. The Bluetooth communications link we will bring to life can also be used to connect your electromechanical device to other robotic devices in a network or to other Bluetooth-capable devices within range that support the Bluetooth Serial Port Profile and the Bluetooth Generic Access Profile.

Our Bluetooth Hardware

The Bluetooth radios we will use as the basis of our Bluetooth data link are off-the-shelf units and can be purchased from Lemos International (**www.lemosint.com**). As you can see in Photo 1, the Lemos Technologies Bluetooth radio modules are designed

PHOTO 1. The nine-pin D-shell connector is intended to connect the Lemos Intl. Bluetooth radio module to the serial port of a personal computer. The external antenna implies that this unit is designed for long range communications. Note the +5 VDC USB power portal and the DTE/DCE switch in this shot.

as high-speed wireless RS-232 links. Judging by the robust external antenna, the Lemos Bluetooth unit will most likely operate at extended ranges. Lower powered short range Bluetooth radio devices are usually fitted with a chip antenna, which resides on the Bluetooth device's printed circuit board (PCB). A closer examination of the Lemos Bluetooth radio module documentation states that the Bluetooth unit in Photo 1 is a Class 1 device. Class 1 devices contain industrial strength Bluetooth radios that can reach out to a maximum range of 100 meters. That's 328.08 feet to the metrically challenged.

Bluetooth devices operate within what is called a piconet. A piconet is a network of two or more Bluetooth devices. The Lemos module documentation implies that up to eight devices can coexist in a Bluetooth piconet comprised of the Lemos units as only addresses 1 through 8 are mentioned in the Lemos module's command structure. If you search through the Bluetooth specification, you'll find that each Bluetooth device is pegged with a three-bit address, which theoretically limits the typical piconet to seven slave devices and one master device.

The master device runs the show on a Bluetooth piconet. All the piconet slaves talk to the master only and cannot perform peer-to-peer communications. In addition, all of the frequency hopping, clocking, and communications time slot parameters within a piconet are governed by the piconet master device.

The Lemos Bluetooth modules do not require the continuous support of an external computing device or a software driver. Initial configuration of the Lemos modules is performed using a subset of the ubiquitous Hayes AT command set. If you've ever used a modem that was equipped with the Hayes AT command set, you can quickly put a Bluetooth network on the air with the Lemos Bluetooth products.

So, what we have here is a self-contained Bluetooth device that is capable of interfacing a Bluetooth piconet with a maximum of eight RS-232 points. RS-232 speeds on the Bluetooth data link can range from 4800 bps to 230.4 Kbps and the Lemos module's Class 1 radio circuitry can extend the RS-232 port's reach out to 100 meters.

It's rather obvious that the Lemos Bluetooth module was designed to be used with PC serial ports as the module's power can be obtained from the host PC's USB subsystem. Let's continue with our Bluetooth network design and I'll show you that the Lemos Technologies Bluetooth modules are also easily adapted for use with a PIC microcontroller.

Get Schmart

The idea here is to transfer to you the knowledge necessary to deploy the Lemos module using a PIC microcontroller. With that, we won't design a dedicated PCB as your I/O and data acquisition requirements will differ from person to person and application to application. Instead, I'll show you how to use a very innovative SchmartBoard prototyping system (**www.schmartboard.com**) to put your Bluetooth platform on the air.

The Lemos module ships with several features that play right into the hands of a PIC microcontroller that is acting as a host to the Bluetooth module. Each Lemos unit is fitted with a physical slide switch (see Photo 1) that configures the Bluetooth module's RS-232 interface for either DTE or DCE. Normally, your PC is a DTE device and external RS-232 devices such as modems are DCE devices. DTE devices are usually terminated with male RS-232 D-shell connectors while DCE devices are most often equipped with female RS-232 D-shell connectors.

The modules must be configured as DCE devices to allow them to directly attach to a standard PC RS-232 port. Thus, the Bluetooth module's RS-232 D-shell connector is female. To keep in sync with the DTE/DCE traditions and standards, the Lemos module package also includes a male-to-male gender changer. The gender changer allows us to enable the RS-232 port with male pins when the Bluetooth module's RS-232 port is physically switched into DTE mode.

If DTE and DCE are mumbo jumbo to you, just remember that the "T" in DTE stands for Terminal and the "C" in DCE stands for Communications. Most of the time a PC is the terminal and anything attached to the terminal is a communications device. DTE and DCE devices are designed to be directly connected to each other without requiring crossover cabling. A crossover cable is only used to connect like devices (DTE to DTE or DCE to DCE).

For instance, if you want to connect the DTE serial ports of a pair of PCs, you must use a crossover cable. The same holds true for connecting a pair of DCE devices. The DTE/DCE slide switch on the Lemos Bluetooth module acts as an instant crossover cable allowing the module to directly attach to a PC's DTE serial port or our SchmartBoard DCE RS-232 interface.

In a nutshell, the Bluetooth module's DTE/DCE switch and a crossover cable simply swap the TXD/RXD and CTS/RTS pin assignments at the D-shell connector. No matter if a device is DTE or DCE, the device's TXD signal must feed the RXD signal and the RTS signal must trigger a response from the CTS pin.

Earlier, I pointed out that the Lemos Bluetooth modules get their

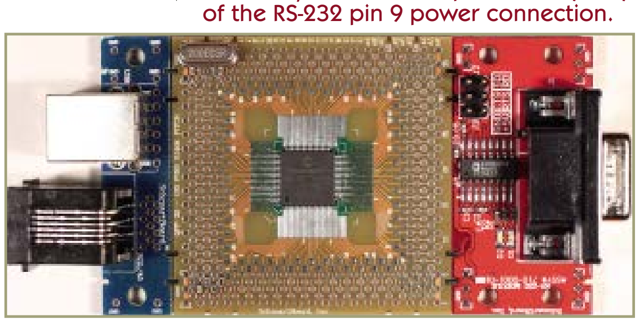
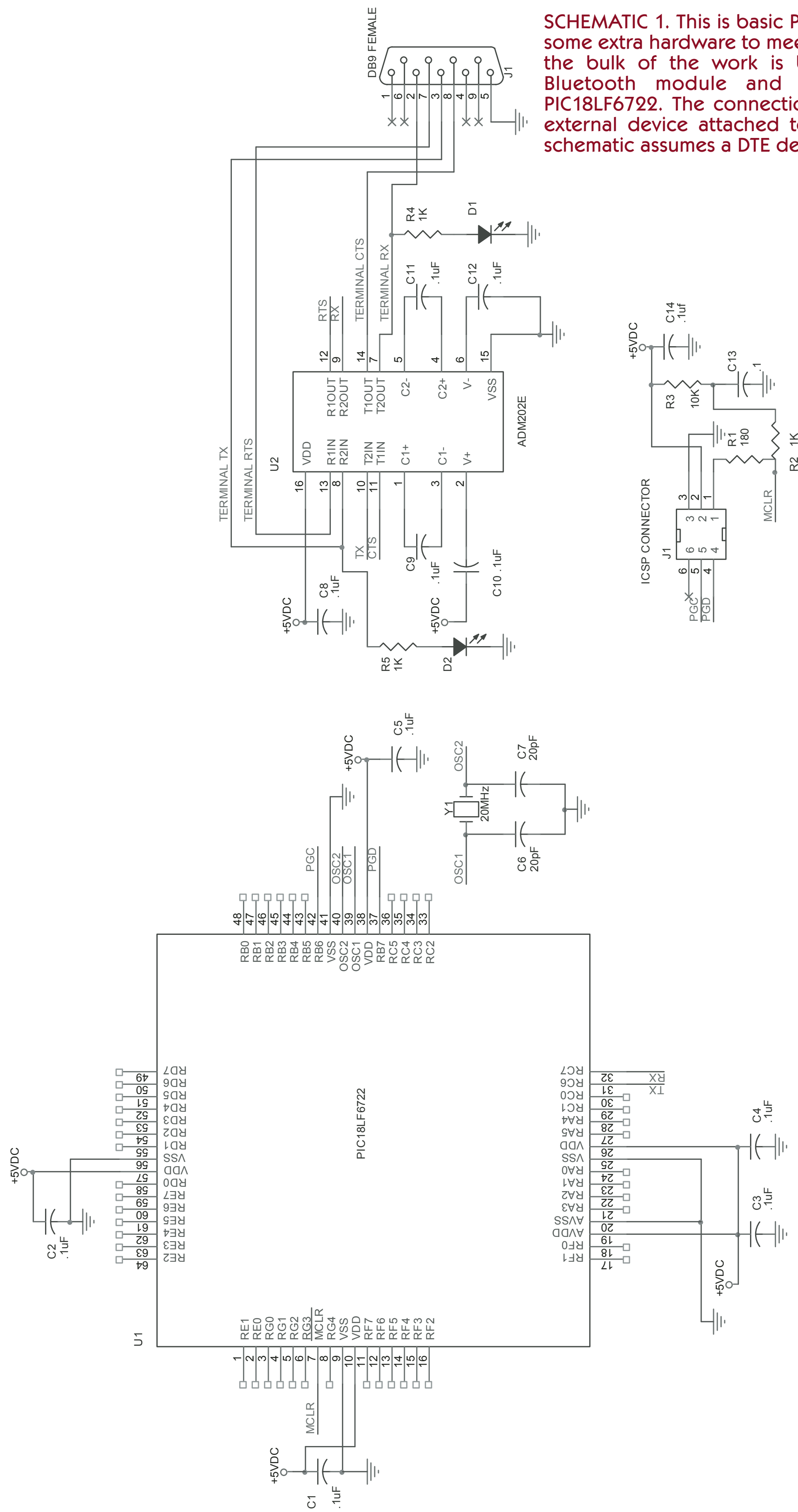


PHOTO 2. The PIC microcontroller-based host is constructed on a trio of Schmart prototyping boards and is ready to be wired up. The USB power connector on the Schmart RJ11/45-USB board makes it convenient to grab +5 VDC from the host personal computer's USB subsystem while the PIC18LF6722 is being programmed and debugged. If the Lemos Technologies Bluetooth module is attached, it too will be powered by way of the RS-232 pin 9 power connection.



SCHEMATIC 1. This is basic PIC hardware design. You may need to add some extra hardware to meet the needs of your application. However, the bulk of the work is being done by the Lemos Technologies Bluetooth module and the firmware embedded within the PIC18LF6722. The connections marked with "TERMINAL" refer to the external device attached to the SchmartBoard RS-232 module. The schematic assumes a DTE device in the "TERMINAL" position.

power from the host PC's USB subsystem. A special USB power cable is supplied with each Lemos unit for this purpose. The Lemos modules can also be powered by applying +5 VDC to pin 9 of the RS-232 connector. The alternate connector power source feature of the Lemos module eliminates the need to include special USB power connectors in our microcontroller hosted Bluetooth design.

I've chosen the PIC18LF6722 to demonstrate the application of the Bluetooth module. My choice of microcontrollers is not cast in stone as your choice of microcontrollers will depend upon your application. The PIC18LF6722 is the largest PIC microcontroller in its class providing 128K of program Flash and 3,936 bytes of SRAM in a 64-pin TQFP form factor. Since the computing platform supporting the Bluetooth module in a robotic application will most likely be collecting data, the PIC18LF6722 is a good choice here as it is equipped with 12 10-bit analog-to-digital converter inputs. In addition, the PIC18LF6722's pair of EUSARTs make easy work of interfacing the PIC18LF6722 asynchronous serial interface to the Lemos Bluetooth module's RS-232 port.

Since we are not drawing up and fabricating a PCB, you're probably wondering how I'm going to support that 64-pin PIC. Without the services of a PCB, the "Schmart" thing to do in this situation is shown in Photo 2.

What you see in Photo 2 is actually a collection of prototyping boards called SchmartBoards. The PIC18LF6722 is mounted on a SchmartBoard EZ that can accept 0.5 mm pitch QFP parts with 32 to 100 pins. A quick look at Schematic 1 shows us that there are supporting passive components (resistors and capacitors) attached to the MCLR, clock, and power pins of the

PIC18LF6722. As you can see in Photo 3, in addition to the PIC18LF6722, the QFP-laden SchmartBoard is able to accommodate the required passive PIC18LF6722 support components in the 0603 SMT form factor. All of the PIC18LF6722's power supply bypass capacitors, the PIC18LF6722 clock crystal, the PIC18LF6722 clock crystal's supporting oscillator capacitors, and the MCLR reset/programming passive components are mounted on the SchmartBoard that is cradling the PIC18LF6722 microcontroller.

The central QFP SchmartBoard is flanked by an RS-232 and RJ11/45-USB SchmartBoard. The RS-232 SchmartBoard (shown in Photo 4) comes as an assembled unit and includes an active RS-232/TTL voltage converter IC. All we have to do is wire the RJ11/45-USB SchmartBoard into the PIC18LF6722's EUSART transmit and receive pins to bring up a regulation SchmartBoard-based RS-232 port. Since the RS-232 SchmartBoard is equipped with a female nine-pin D-shell connector and is wired as a DCE device, we will have to attach the male-to-male gender changer to the Lemos Bluetooth module's RS-232 connector and physically switch RS-232 interface to DTE.

I've also wired the RS-232 SchmartBoard's D-shell connector pin 9 to +5 VDC, which will allow us to power the Bluetooth module through the pair of RS-232 connectors. A schematic of the SchmartBoard RS-232 interface board is available on the SchmartBoard website. Detailed PDF diagrams of the SchmartBoards I've used in this Bluetooth design are also readily available there.

While we're on the subject of power, the USB interface of the RJ11/45-USB SchmartBoard is being used to siphon +5 VDC from the host PC's USB subsystem while the PIC18LF6722 is being programmed and debugged by way of the RJ11 jack. In our case, the other end of the cable attached to the SchmartBoard's RJ11 jack is attached to an MPLAB ICD2. The MPLAB ICD2 can be used as both a debugging platform and a PIC programmer when coupled with Microchip's MPLAB IDE.

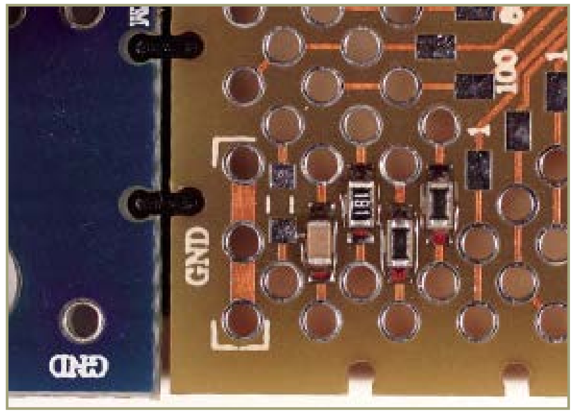


PHOTO 3. There are more than enough 0603 pads to support the passive components required by the PIC18LF6722. This shot was taken before I wired the components in. I made the point-to-point connections between the passive components and the PIC18LF6722 using wirewrap wire. If you look closely, you can see the bridge strips that hold the SchmartBoards together.

The SchmartBoard layout you see in Photo 2 is optimized for the application debugging phase. Once the PIC18LF6722's Bluetooth application is working just the way you want it, you can power the PIC18LF6722, the RS-232 SchmartBoard, and the Lemos Bluetooth module with the best method your electromechanical device offers. The RJ11/45-USB SchmartBoard is optional equipment at this point.

If the SchmartBoard concept is new to you, the idea behind the family of SchmartBoards is to allow you to easily integrate fine-pitch devices such as the PIC18LF6722 into your everyday designs. Hassle-free soldering of fine-pitch components such as the 64-pin PIC18LF6722 is made possible by the unique design of the SchmartBoard's pin channels. Each of the hosted part's fine-pitch IC pins actually sits in a solder-filled channel on the SchmartBoard, which allows the human designer to easily solder each pin perfectly into place.

Just in case your design can't get away with using only a single SchmartBoard, every SchmartBoard is identically notched at its edges to accept a mechanical bridge that allows various SchmartBoards to be physically connected as I have done in Photo 2. You can easily see the mechanical bridge in Photos 3 and 4.

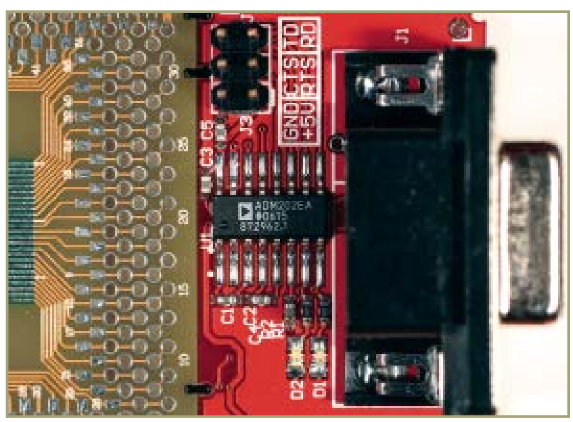


PHOTO 4. Here's a bird's-eye-view of the SchmartBoard RS-232 interface. A pair of LEDs illuminate in unison with the transmit/receive data flow. I included this module's circuitry for logical clarity in Schematic 1. Refer to the RS-232 module schematic you get from SchmartBoard when you need to identify the physical components on this module.

Configuring the Bluetooth Modules

Okay ... Our PIC18LF6722-hosted Bluetooth device hardware design is in place. The next step of our design process involves configuring the Lemos modules. There are two ways we can go about the configuration of our pair of Bluetooth modules. We can simply connect the Bluetooth modules to a PC serial port and configure them there. Or, we can configure the Bluetooth modules using an algorithm that we must place within the PIC18LF6722 application firmware. Either way, the AT commands issued will be identical. Only the medium of command delivery (PC or PIC18LF6722) will differ.

The first order of configuration business is to establish a master Bluetooth node. The Lemos modules ship as slave units by default. Odds are your electromechanical being will be sending its collected data to a central point for processing. Thus, the receiver of the data in this case would serve as master and your Bluetooth-equipped robotic device would be configured as a Bluetooth slave.

If you wish to have your robotic device control the transfer of data in the piconet, then it should be configured as the master of the piconet. Naturally, if no PC is involved and all of the Bluetooth nodes are robotic in nature, one of your artificially sentient beings must assume the role of master as none of the Bluetooth peers will be able to communicate with each other directly.

Each and every one of the Lemos configuration commands are prefixed with "AT" and are followed by a carriage return (0x0D). So, to override the default slave assignment and configure the target Bluetooth module as a master, we must issue the command "ATR0" followed by a carriage return. You will also want to disable the prompt messages (OK, ERROR, CONNECT, DISCONNECT) that are returned by the Bluetooth unit by issuing the command "ATQ1." There's no need for the prompt messages unless you want to process them for some reason in your PIC18LF6722 application firmware.

Another look at Schematic 1 tells us that none of the modem control signals (RTS, CTS) are implemented via the PIC18LF6722 I/O pins. The Bluetooth modules ship with flow control enabled. So, to complement the lack of any PIC18LF6722 modem control hardware configuration, we must disable flow control within the Lemos units. Flow control is disabled by issuing "ATC0." Since I only have a pair of Lemos Bluetooth modules, assigning a master unit, disabling prompt messages, and disabling flow control are all that is necessary to establish a wireless RS-232 data link between my pair of Bluetooth modules. Disabling flow control is allowable here as we will not be streaming asynchronous data between the master and slave devices of our little piconet. If your application will require flow control, you will have to designate an RTS and CTS pin on the PIC and fill in the firmware blanks accordingly.

The Bluetooth units default to automatic connection for both the master and slave modules. If more than a pair of Bluetooth modules will be put into action, you'll need to disable the Bluetooth master module's auto connect feature (ATO1) and optionally assign names (ATN=xxxx with a maxi-

SOURCES

Lemos Technologies —
www.lemosint.com
LM058 Lemos Technologies
Bluetooth Modules

Microchip — **www.microchip.com**
PIC18LF6722; MPLAB ICD2; MPLAB IDE

SchmartBoards
www.schmartboard.com

mum of 16 characters), pin numbers (ATP=xxxx with a maximum of eight characters), and security strings (ATD=xxxxxxxxxxxx) to all of the nodes.

When using security strings, the master connects to a particular slave configured with the matching 12-character security string by issuing the command "ATA." When the "ATD=xxxxxxxxxxxx" command is issued in slave mode, the 12-character security string becomes the receive filter mask used by the receiver to determine whether or not to connect to the requesting master. Issuing the "ATD=xxxxxxxxxxxx" in master mode enables the master to use "ATA" to connect to the slave Bluetooth node that is loaded with the matching 12-character security code. If you decide against using security strings, you may also connect to a slave device after querying for devices with the "ATF?" command. If any Bluetooth devices are found, they will be listed by name and address. The addresses returned will range from one to eight. Connection to a slave with an address of one would be affected by issuing the command "ATA1." A device's address can be queried locally using the command "ATB?."

The default Bluetooth module baud rate of 19200 bps may not suit your application. You may alter the Lemos module's baud rate using the K (stop bits), L (baud rate), and M (parity bit) AT commands. Issuing "ATZ0" will reset the module to its factory default settings.

Where's the Code?

Code is on the *SERVO* FTP site awaiting a download request from you. You can access this through the website at **www.servomagazine.com**. I've supplied the basic firmware building blocks you will need to configure and use the Lemos Technologies Bluetooth modules. You can adapt my code package to wirelessly monitor and control voltages, pressures, and temperatures, among other things, inside of and external to your robotic creation.

Lemos Technologies usually publishes the user guides for all of the products they offer on their website. Thus, you can get your hands on all of the options offered by the Lemos AT command set with a download of the Bluetooth module's user guide. The pair of Lemos Technologies Bluetooth modules I've described in this text are part number LM058.

In my opinion, nothing is easier to implement on a microcontroller than an RS-232 communications link. The folks at SchmartBoard have taken the pain out of assembling the PIC18LF6722 hardware and the Lemos Technologies engineers have tamed the RS-232 Bluetooth interface. All that's left is for you to apply the Bluetooth knowledge you've gleaned from this article and get the accompanying source code package to your robotic application. SV

Beginner's Guide to Programming: Lesson 1

by Michael Simpson

I recently attended the RobotFest held each year in Linthicum, MD. The one question I was asked the most was "How do I get started in robotics?" There are many ways to answer this question, but the bottom line is that you really need to learn to program.

I know several programming languages and have had the opportunity to teach a few of them. By far, the easiest to learn is the Basic programming language. In this series, I am going to teach you the basics of Basic.

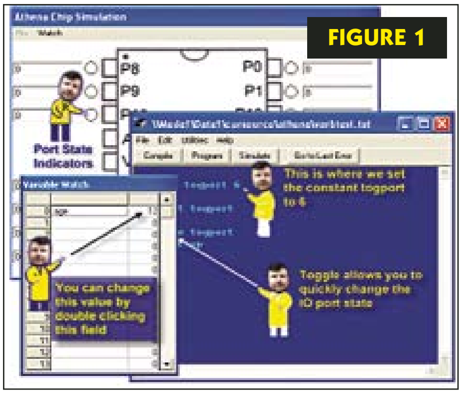


FIGURE 1

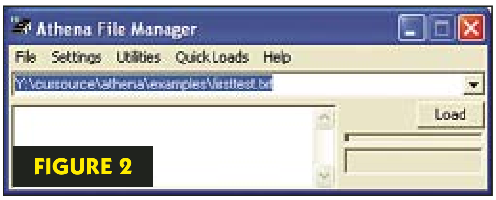


FIGURE 2

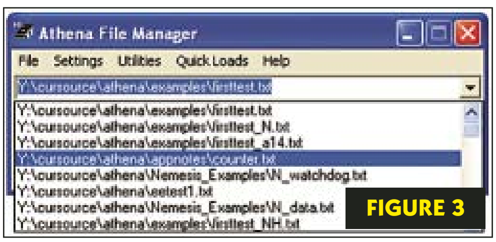


FIGURE 3

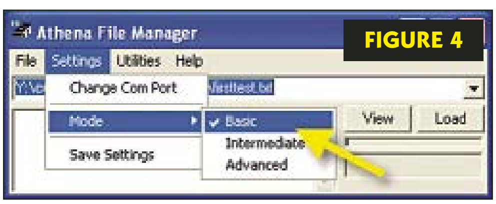


FIGURE 4

Kronos Robotics has a free Basic compiler that has a simulator that will allow you to learn to program a microcontroller without ever touching one. With this compiler, I will show you step-by-step how to program.

Athena Compiler

The first thing you will need to do is to download the free Basic compiler from the Kronos Robotics website at **www.kronosrobotics.com/downloads/AthenaSetup.exe**.

Once downloaded, you will need to install it. Follow the install instructions. The compiler will run on all Windows platforms and installs all components needed to run the simulator. Once installed, run the compiler to get started.

The heart of the Athena Compiler is the File Manager shown in Figure 2. This form will allow you to open or create new programs. It also keeps track of the last 50 files you have worked on (see the dropdown list as shown in Figure 3). There are other shortcuts and features, so feel free to experiment.

Under the Settings menu, you need to set the mode to Basic as shown in Figure 4. This will make things easier to follow as we continue. You can always change it later once you get the hang of things. Save the mode by selecting the Save Settings menu option.

You will also notice the Change Com Port setting. Since we are going to be using the simulator, it is not necessary to change this setting until you actually want to program a chip.

Creating Your First Program

Just about every book I have read on programming starts out with the "Hello World" program. Because I am a bit old fashioned, I will do the same. This will quickly demonstrate the basics for creating your own programs.

From the File Manager File menu, select the New Athena File option. This will create an empty edit form. This is the starting point of all new programs. The main content area (blue) is where you type in your program code. Type the line Print "Hello world" and hit the Simulate button as shown in Figure 5.

Several things happen when you hit the Simulate button. The Chip Simulation Form is loaded as shown in Figure 6. This form is used to show you the various pin conditions on the microcontroller's I/O ports. You can also set input conditions as well, which we will be getting into a bit later. Each I/O port has a small circle located next to it. The color of this port represents the state of the port. White indicates the port is in input mode. Gray indicates the port is in output mode and in the low state. Red indicates the port is in output mode and in the high state.

The Edit Form of the program you are simulating has also changed as shown in Figure 7. The Program Control Bar was added and the command Print was highlighted. When you are in simulation mode, the program is in single step mode and the highlighted command indicates the next command that will be executed when the Single Step button is hit. Figure 8 shows all the commands in the Program Control Bar. Go ahead and hit the Run button. This will cause the program to run until the end of the program is reached.

The Print command sends data to the Debug Terminal, so after the program is run, the words "Hello World" are displayed as shown in Figure 9. That's how you use the simulator. It's not much different in actually programming a chip.

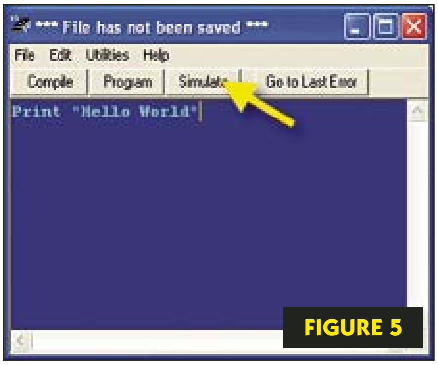


FIGURE 5

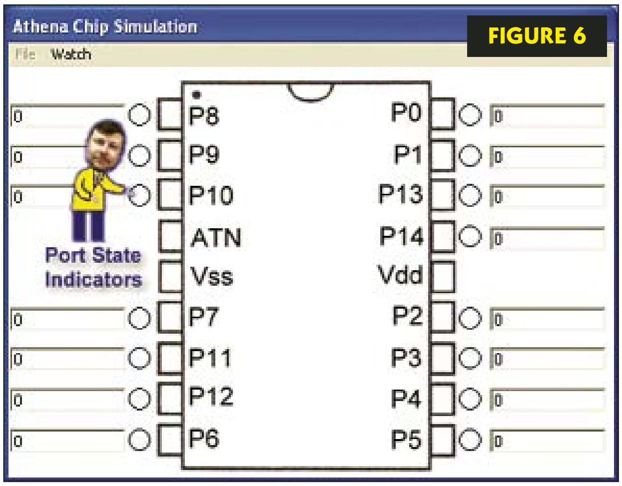


FIGURE 6

Programming 101

The Basic language is a procedural one. That is to say that it starts at the beginning and continues executing instructions as they are encountered until it reaches the end. Take a look at the program in Figure 10. Type it in and run the simulator. Step through each instruction and watch the Debug Terminal as the program executes. Once the program reaches the end and there are no more instructions, it simply stops.

Display Commands

Display commands are used to send text, characters, and control codes to various devices connected to the I/O ports, including the Debug Terminal. We used the Print command in the last example. Other Display commands include the Debug, Serout, and LCDWrite commands, the most popular being the Debug and Print commands. The Print command is identical to the Debug command except that it automatically adds a linefeed and carriage return to the end of whatever you are printing.

When using the Display commands, data is sent in ASCII form. For example, the value 65 with send the character "A." If you want to display the value instead, preface it with the dec modifier as shown in Figure 11. The Print command always displays the value, so the dec modifier is inferred.

Some Basic language dialects use different types of modifiers. For instance, Visual Basic's Print command always infers the dec modifier, and you must use other means to send the raw ASCII value.

The Print and Debug commands send raw serial data to the debug terminal via the program ports when the chip is connected to the PC. This is why the I/O port 12 is set to output when you run the simulator. In most instances, I/O ports 11 and 12 are reserved for programming and Debug/Print commands. Other micro-controllers will use other ports for this.

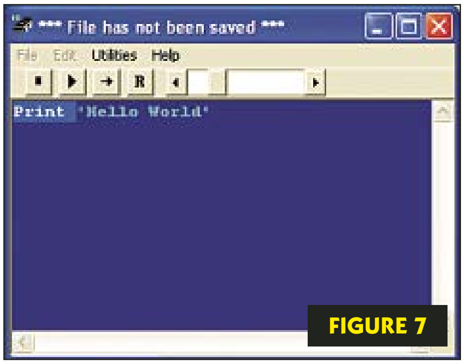


FIGURE 7

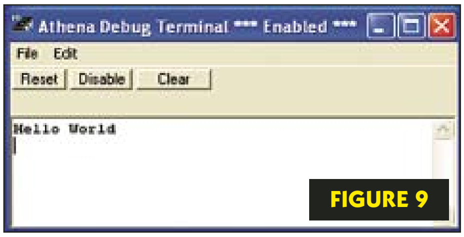


FIGURE 9

Program Flow

As I stated earlier, the program flows from top to bottom. You can, however, change the program flow with commands like Goto and Gosub. Before you can use the commands, you must define a label in your program. Load the program in Figure 12 and run the simulator. Notice how the program keeps jumping back to the Print instruction. Labels are just place holders. They represent a location in your program. The Gosub command works much the

FIGURE 8


TIP #1

The Basic language is by definition not case sensitive. If you noticed, we used the Print command. You could also have used print or PRINT and all would have worked. Just keep in mind that this may not be true of all dialects of Basic.

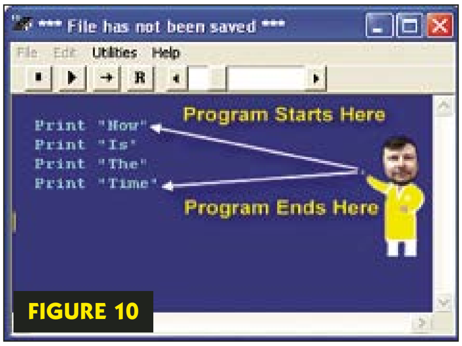


FIGURE 10

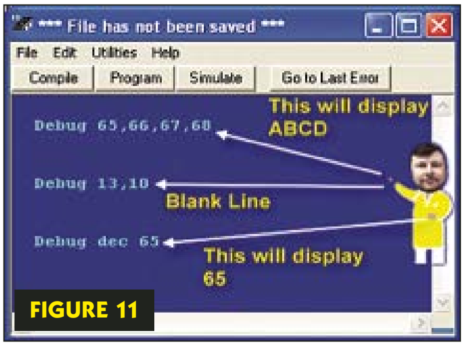


FIGURE 11

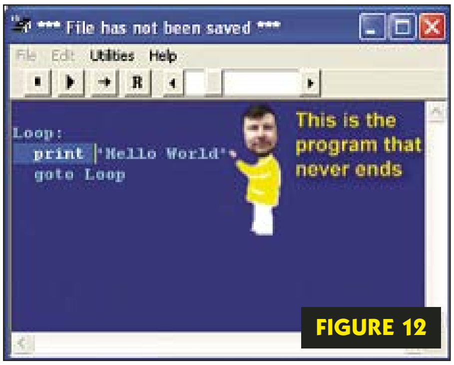


FIGURE 12

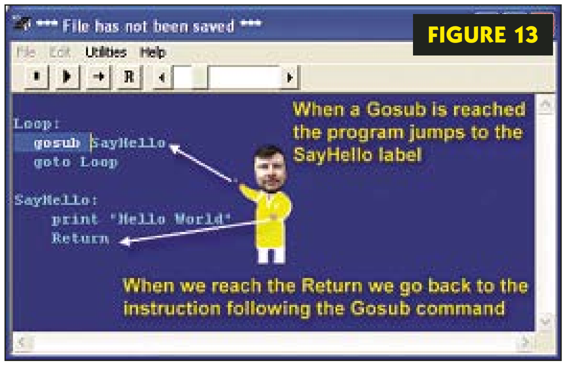


FIGURE 13

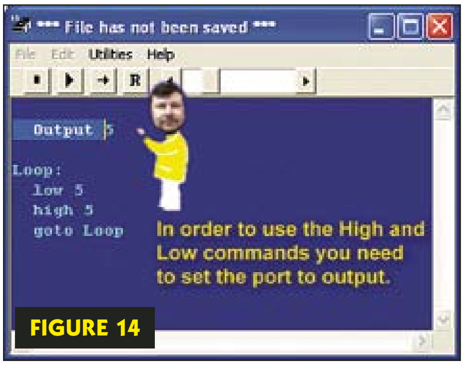


FIGURE 14

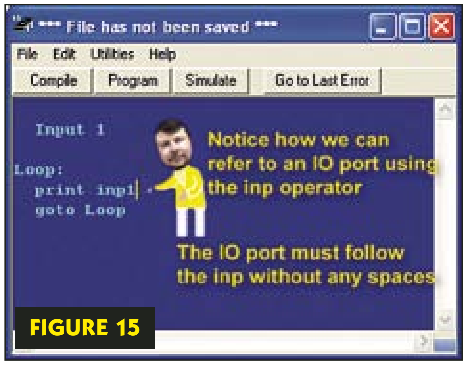


FIGURE 15

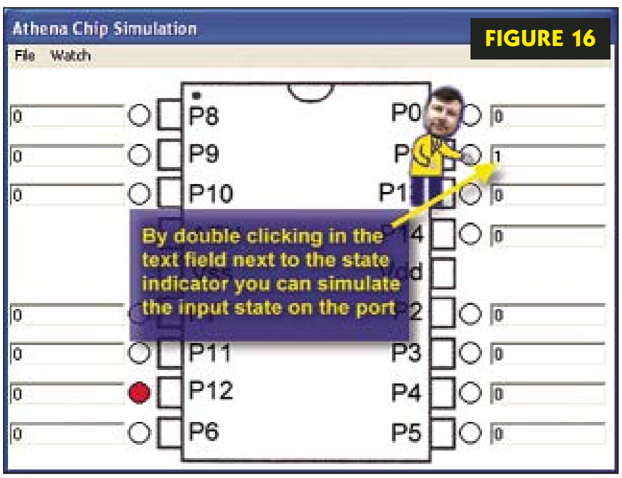


FIGURE 16

same way as the Goto command. When the Gosub is encountered, it saves its position in the program just before it jumps to the indicated label. The program then executes as normal until the Return command is reached. At that point, the saved location is retrieved and the program continues at the next command following the original Gosub command. Load and run the program in Figure 13. Single step through the code and you will see how the Gosub command changes program flow.

If you use too many Goto commands to jump around the program, it will start to get unwieldy and hard to follow. This is called spaghetti code. The use of the Gosub command can help you modularize your code into functional chunks. It can also save valuable program space by allowing you to reuse code.

I/O Ports

The I/O ports on a microcontroller are used to control or communicate with the outside world. We can set the I/O ports as input or output. This is sometimes referred to as the port's direction. To change the direction of the I/O ports, use the Input and Output commands. When an I/O port is in the output mode, you can set the level to high or low using the High or Low commands.

Load and run the program in Figure 14. Single step through the program to see how the various commands affect the I/O port. Once you do that, run the program in continuous mode and you can watch the port toggle on and off.

We can also test the condition of an I/O port when the port is set to input mode. To do this, you use the inp operator. Load and run the program in Figure 15. Run the program in continuous mode. Once the program is running on its own, double click the text field next to I/O port 1 as shown in Figure 16. Some Basic dialects used in other microcontrollers use the command IN instead of the inp command.

The text fields on the Chip Simulator form allow you to change the external stimuli to a particular port. This allows you to test your program based on the input port state. As you toggle the stimuli field, notice how the program changes what it prints in the Debug Window.

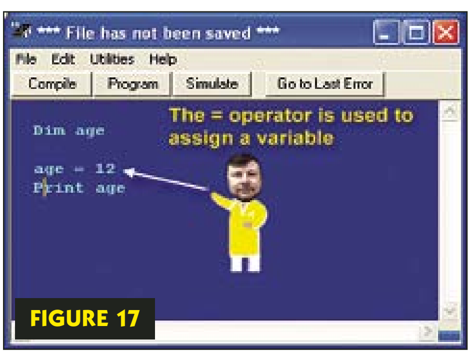


FIGURE 17

FIGURE 18

Variables

In order to understand how variables work, think about it like this. Pretend that

TIP #2

The inp operator can be used in various ways. You can use inpX or inp.X where X is the port number. You can also use inp.const where const is a defined constant indicating the port number.

a food recipe is our program. The various pots, pans, and mixing bowls are our variables. The actual food ingredients are our values. The program (recipe) tells us what values (ingredients) to place in what variables (pots, pans, and bowls). When we mix ingredients, we are performing math operations on them. When the program is all done, we end up with a final set of variables (plates) that contain the results of all our efforts.

Variables are what we use in our program to store values that our program tells us to retrieve or manipulate. In order to create a variable for our program to use, we use the Dim statement. Once a variable is created using the Dim statement, we can then use it in our program. The most basic operation we can do with a variable is to assign it a value.

Load and run the program shown in Figure 17. When the program prints the age variable, the value assigned will be displayed in the Debug Terminal. The Print command is very versatile. It can display variables, registers, numbers, constants, I/O ports, and quoted strings. This is true of all the display commands.

The simulator has a way of looking at your variables. From the Chip Simulator form, select the Show Variables option from the Watch menu. This will cause the small Variable Watch form to pop up as shown in Figure 18. This form will allow you to view all the defined variables. You can even change the values, overriding the program.

Since the Athena was designed for beginners, the variables are very simple. A variable can contain a value of 0-255. As you get into other microcontrollers and Basic language dialects, you will have other variable types and sizes.

Constants

Constants allow you to assign a useful name to a number. This allows us to define a value like an I/O port number and reference it by name throughout our program. Load and run the program in Figure 19. Notice that I/O port 6 changes state as the program runs. Stop the program and change the togport constant to 7; run the program again. See how we only had to change a single entry in our program in order to modify the port number. This becomes more important as our programs grow in size.

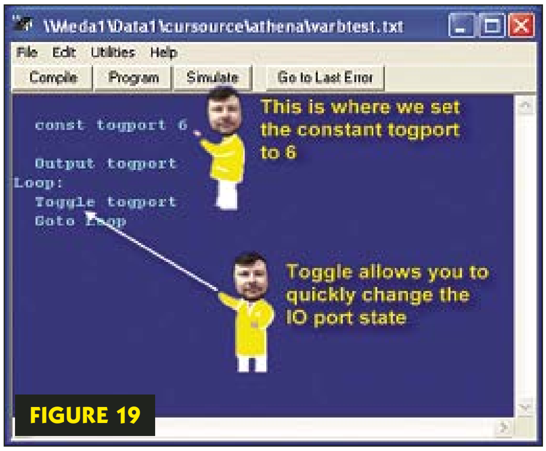


FIGURE 19

Math Expressions

When you assign a value to a variable, you can perform a math operation on the values that you are assigning as shown in Figure 20. You may also use constants in your mathematical expressions.

FIGURE 20

You may perform multiple math expressions such as those shown in Figure 21. Basic language compilers have an order in which certain operations take place in the expression. This is greatly simplified with the Athena Basic compiler. All operations are in the order in which they are encountered.

You may also assign values to variables and use those variables in your expressions as shown in Figure 22. The Athena compiler only supports integer math. If a mathematical expression yields a value that contains a fractional value, only the whole number portion of the result is returned.

Summary

This month, we covered the operation of the Athena simulator and the basics of the Basic language. At this point, you should understand how the program flows and how to change that flow with the use of the Gosub and Goto commands. You should understand how to manipulate the I/O ports when they are in output mode. You should understand how to define a variable and assign it a value using a mathematical expression. If you can answer the following questions, then you understand this month's lesson and can move on to the next lesson with confidence.

1) What is the main difference between the Goto and Gosub commands?

2) When setting the state of an I/O port, what direction must they be set to?

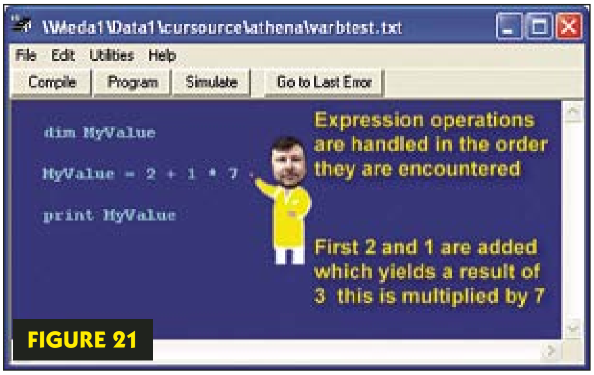
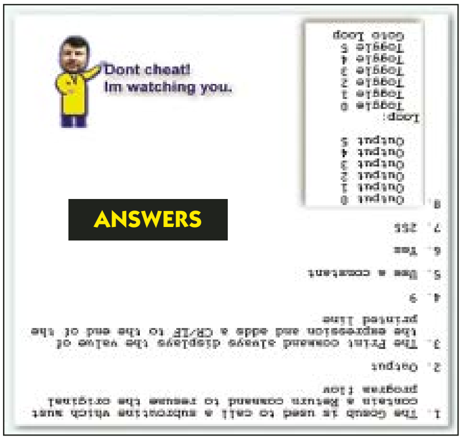


FIGURE 21

FIGURE 22



3) What are the differences between the Print and Debug commands?

4) What is the result of the following expression: 5 * 2 + 6 * 3 / 5?

5) How do you give a name to a numeric value?

6) Can constants be used in mathematical expressions?

7) What is the maximum value an Athena variable may contain?

8) Using only the commands that we covered, create a program that toggles each of the I/O ports 0-5 in order, one after another.

Next month, we will cover looping commands and conditional branching. Remember that you can pick up a free copy of the Athena compiler/simulator at **www.kronosrobotics.com**.

The program installs a complete manual and as several sample programs. **SV**

ROBOBUSINESS 2007

HIGHLIGHTS


Foster-Miller TALON.


Helen Greiner of iRobot, with Packbot.


Honeywell MAV.

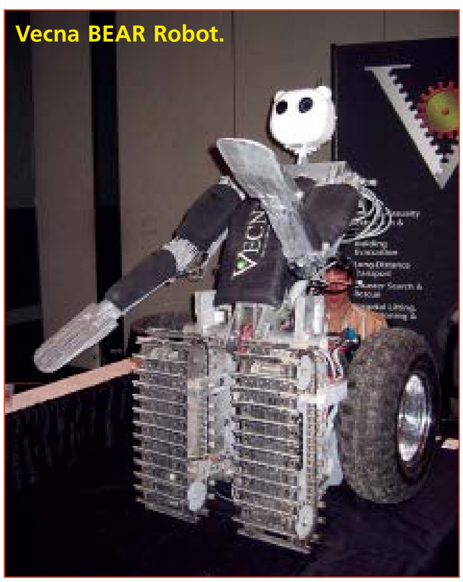
by Ted Larson

The RoboBusiness show is now in its fourth year and it continues to grow. We were not able to attend RoboBusiness last year in Pittsburgh, PA but consensus seems to be that it is doubling in size every year. Two years ago, it was held in a ballroom, in a hotel in Cambridge, MA and there were perhaps 15 exhibitors. I could have taken a photo of every single robot there, and easily fit it into this article. This year, it was held at the Hynes Convention Center in downtown Boston, and there were easily 60 exhibitors, and too many robots to list. Vision, sensors, military/government and service robotics seemed to dominate the show, and there was a ton of stuff to see in only two days. I'll cover just the highlights of what I saw, which was plenty!

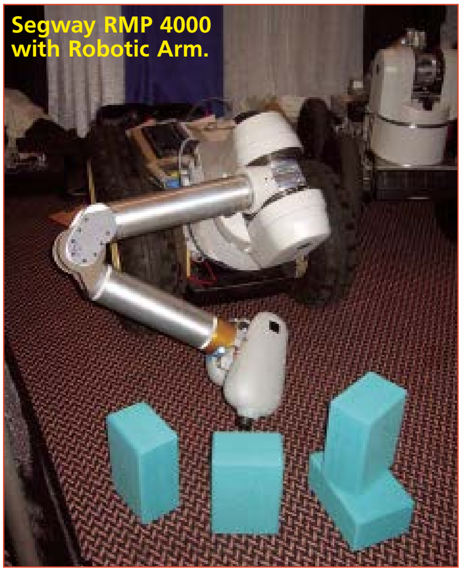

Black-I Robotics LandShark.

There was an impressive array of robots designed for military applications — many for explosive ordnance disposal (EOD) and improvised explosive device (IED) detection and elimination. Foster-Miller (**www.foster-miller.com/lemming.htm**) was there with several of their TALON robots. They even drove them around a bit on the show floor. TALON's are currently in use in Iraq, being used for explosive ordnance disposal, reconnaissance, security, defense, and rescue. Literature in their booth claimed "Numerous TALON robots have survived high-order detonations in Iraq. One robot has actually been blown up three times yet is still in the fight with new arms, wrists, wiring, and cameras." Now that's a robot I want on my side!

iRobot (**www.irobot.com**) was there with their new Packbot 510 with an EOD kit mounted on it. Of course, in its gripper was what looked like a dummy mortar round. I snapped a great photo of Helen Greiner, Co-Founder and Chairman of iRobot, posing with the


Vecna BEAR Robot.

Allen-Vanguard EOD Robot.


Segway RMP 4000 with Robotic Arm.

robot. She laughed when I asked her for the photo, because last time I took a picture of her with a robot, it ended up on the cover of this magazine.

Next to the new Packbot was iRobot's new Warrior X700, which is much larger than the Packbot, and is designed to carry greater than 150 lb payloads while traveling through rough terrain or climbing stairs. It is a multi-mission robot that can be used for EOD, reconnaissance, hazardous materials work, battlefield casualty extraction, firefighting, weaponized missions, and more. At the show, iRobot had a very rugged looking arm mounted to it, and they demonstrated it picking up and manipulating a very heavy, dummy cannon shell that could easily be the size of an existing IED.

I spent some time talking to Colin Angle, CEO of iRobot, about the future of service robots, and he is very optimistic that the robot industry is moving in the right direction. He feels confident we will see more and more service robots in our homes. His optimism is partially based upon the fact that iRobot has recently passed the two million mark in sales of their Roomba vacuuming robots and over 900 Packbot Tactical Mobile Robots. They are continuing to release new, innovative


Blue-Fin Robotics AUV.

service robotic products. Notable, recent product introductions have been the iRobot Dirt Dog Workshop Robot and the Verro Pool Cleaning Robot.

Honeywell (**www.honeywell.com**) was exhibiting their Micro Air Vehicle (MAV), which looks like something right out of a science fiction movie or a video game. I play some X-Box 360 here and there, and most of the first person shooters have some kind of drone you can send out to search for enemy troops. That is exactly what the MAV is, but for real!

It is a hovering platform that uses directional, downward thrusting engines to fly and can detect and recognize man-sized targets at 250 m during the day or 125 me at night. It can fly autonomously with up to 10 preplanned flight plans, or can just loiter above for aerial reconnaissance to let ground troops know what is ahead by beaming data back to a handheld ground station about the size of a large laptop computer.

The MAV weighs about 14 pounds and is nearly impossible to see at night since it's painted black. There is some fantastic video of this robot in action on Honeywell's website (**www.honeywell.com/sites/portal?smap=aerospace&page=mav_video&theme=T8**). It is currently under development and is scheduled for deployment with the military in 2008.

A newcomer to the military robot scene is Black-I Robotics (**www.blackirobotics.com**) with an inspirational story of how the company was started. Brian Hart, co-founder of the company, started Black-I Robotics after his son was killed in Iraq at the age of 20. The founders of the company were alarmed at the slow rate of fielding robust and cost-effective robotic platforms which save lives of soldiers and innocent civilians. One of their slogans is "Men shouldn't be asked to do a machine's job."


Excel Auto G5 Robot w/Founder Fernando Ramirez.

Their newest product called LandShark is an open source/open platform chassis, designed with off-the-shelf parts to keep the cost down. It is currently a tele-operated, unmanned ground vehicle (UGV) with mission specific modules that can be added on. Some upcoming modules Black-I will be releasing are for autonomous research, SWAT, HazMat, Homeland Security EOD, and explosive ordnance demolition including IED or VBIED neutralization. The LandShark looks like a robust platform and the videos on their website of it driving around look fantastic. I hope to see more interesting things out of Black-I in the not too distant future.

There were several other mentionable military robots at the show. Vecna (**www.vecna.com**) was there with their BEAR robot, designed for battlefield extraction and search and rescue. Allen-Vanguard (**www.allen-vanguard.com**) had their EOD/ROV robot on display, and Segway (**www.segway.com**) presented their new line of heavy duty,

"One of the great things about these Robotics Trends conferences is that they encourage everyone to get together, socialize, and share ideas."

four-wheel-drive rovers, along with their other Segway RMP products.

One of the great things about these Robotics Trends conferences is that they encourage everyone to get together, socialize, and share ideas. Moday night, before the show opened, Holland+Knight (**www.hklaw.com**), held a wonderful mixer with lots of free drinks and hors d'oeuvres, and a showcase of all the robotics companies they represent. Many of them were from the Mass Tech Robotics Cluster (**www.masstlc.org/clu/robotics**), which represents a huge list of robotics companies in Massachusetts, like iRobot, Boston Dynamics, Foster-Miller, and Raytheon, just to name a few. Boston and the surrounding area definitely seems like the center of the universe when it comes to robotics in the US.

Tuesday night, there was another opportunity to consume free drinks — courtesy of iRobot and Foster-Miller . The mixer was held at the Prudential Center Skywalk Observatory which gives you a 360 degree view of the Boston area — complete with maps on the wall to show the important historical sites in Boston. It was great to be surrounded by all that history and talking about the future of robotics!

One of the more interesting keynote comments was made by Dan Kara from Robotics Trends (**www.roboticstrends.com**), now part of EH Publishing. "The original name for the RoboBusiness event was the 'Emerging Robotics Technologies and Applications Conference and Exposition.' That is quite a mouthful, and really does not connote anything. It became very clear at that first show what everyone at the event was focused on was the business side of the robotics business — hence 'RoboBusiness'."

This comment really made it clear to me how this show has evolved to its specific focus on the business of robotics. What still amazes me the most is how the robotics industry is still small enough that when there is an event like this, you can meet many of the people in-person who are shaping its future. Founders and CEOs of companies like iRobot, Evolution Robotics (**www.evolution.com**), BlueFin Robotics (**www.bluefinrobotics.com**), Black-I Robotics, and many, many more. I met too many people that night to remember everybody's name, but it was definitely one for the memory books. I talked about robots until I was almost hoarse, and it was only the first night of the conference!

During setup day at the show, my business partner Bob comes back to our booth and says, "I just saw one of the coolest ideas for a robot I have seen in awhile. You gotta go check this thing out." He had been down the aisle at the Excel Auto booth.

Excel Auto (**www.excelsg5.com**) has an integrated police vehicle and robot system called Guardian 5 (or G5 for short). According to company founder, Fernando Ramirez, his initial inspiration for the product was two highway patrol incidents that occurred last year. One officer was killed when a person hit him while performing a routine traffic stop, and the other officer was shot when he pulled over a car and approached the driver who turned out to be an escaped convict.

The goal of the robot is to minimize danger to the officers by allowing the robot system to be operated from the driver seat of the patrol car. It has a special carrier system that allows the robot to be deployed and stored without interfering with existing systems in the vehicle. When the robot is stored, the batteries recharge via electrical contacts on the carrier that tap into the patrol car's charging system.

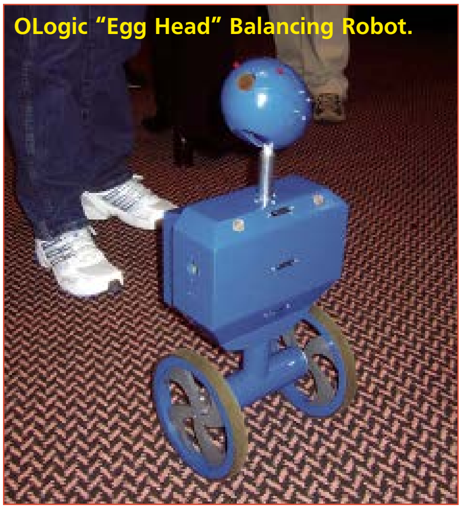
The robot can be used in many police situations to keep the officers safe. The whole idea could easily be extended to military vehicles as well, to keep our troops safer in Iraq or other conflict zones in the world. There are tons of photos and videos of the robot and storage system in-action on their website and it is definitely worth a look.

As I was walking through the expo, I encountered a gentleman pulling an acrylic tube with wheels on it. Inside there appeared to be an entire robotic forearm, with a hand. I got down on the ground and took a closer look at it, and it was an amazing looking piece of technology. Inside was the Shadow Dexterous Hand made by Shadow Robot Company (**www.shadowrobot.com**). (You can check out the full length article on it in the November '05 issue of *SERVO*.)

BrainTech - Nerf Shooting Robot.


Shadow Robot Arm in a Box.


InTouch Health RP-7 Robot.



OLogic "Egg Head" Balancing Robot.

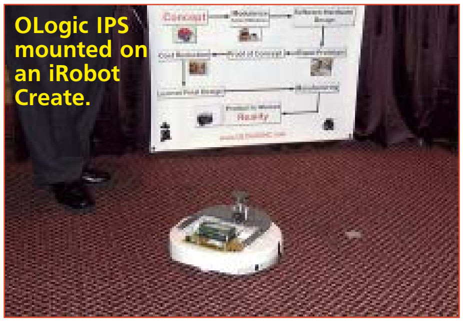
The Dexterous Hand is designed for humanoid robotic applications and reproduces as closely as possible the 24 degrees-of-freedom in the human hand. The forearm contains approximately 40 air muscles that manipulate the hand. It follows biologically-inspired principles in its design, and artificial tendons couple the air muscles to the joints. Integrated electronics at the base of the hand system drive the pneumatic valves for each muscle and also manage corresponding muscle pressure sensors. There are tactile sensors distributed across all the finger tips, so it can sense exactly how much force the hand is exerting, or for sensory feedback and positioning.

Best of all, the whole hand system is controlled via a Controller Area Network (CAN) bus interface, so it is easy to use, too. They recently sold one to NASA to test on their Robonaut project. You could have one too for approximately USD$15,000.

I came across one booth with a tall, sleek, friendly looking robot with a smiling face on a video monitor, ready to have a chat about just about anything. It was the RP-7 Remote Presence Robot, by Intouch Health (**www.intouchhealth.com**).

This robot is designed to handle the bottleneck of physician's visits in a hospital setting. It may be a patient waiting for a specialist to determine if intervention is required, or a post-treatment patient waiting to learn the outcome of a test.

The physician-patient encounter is critical in hospital care. The number of hours patients simply sit around waiting for physician interaction is staggering, and has many implications for hospital capacity and management. A couple of years ago, I was in the hospital for a night, after eating some bad swordfish and I can attest to this, personally! I spent half the next day just waiting to get the doctor signoff, so I could be discharged. With the RP-7 robot, all the doctor has to do is log-in, drive the robot around to their patients by remote, and quickly buzz through their rounds. Intouch Health has done several trials with this robot at places like Johns Hopkins and UC Davis Medical Center, and it received some rave feedback from both patients and staff alike. It is a great idea, and it looks cool, too!



OLogic IPS mounted on an iRobot Create.

There were several companies showing vision systems for use in robotics. One of the more impressive demonstrations was at the Braintech booth. Braintech (**www.braintech.com**) develops and deploys Vision Guided Robotic (VGR) software and guidance technologies that enable adaptive robotic processes. They were demonstrating their 3D vision guidance system with a little PC camera capable of locking on and shooting a Nerf™ dart at a target. They were giving away T-shirts in their booth with big targets painted on them, so you could get shot by the Nerf gun.

It is an industrial strength vision system you can mount on the end of a robot for precisely positioning the arm by vision alone without any specialized vision expertise needed. Their newest software development kit (SDK) is running with Microsoft Robotics Studio, so it should be really easy to plug into any Windows-based robot development project.

Of course, OLogic (**www.ologicinc.com**) was there, with more new designs for robotic applications. The star attraction was an ultra-low-cost Indoor Positioning System (IPS) design, based upon the same initial technology showcased in the IGOR robot, shown at CES 2007.

The IPS uses ultrasonic beaconing to precisely compute its position in a room down to less than a 10th of a foot. The low-cost version could be embedded in an existing, high-volume consumer robotic application for as little as $6.

An iRobot Create robot, was outfitted with IPS sensors so that it could drive around in front of the booth and navigate to waypoints on the floor marked by pieces of tape (as well as display where it was while driving). OLogic also unveiled some new balancing robot designs, along with their newest balancing robot, Egg Head. According to Bob Allen, co-Founder of OLogic and primary designer of Egg Head, "Egg" was designed to "demonstrate our capabilities to produce a more finished end-product, complete with wide composites of OLogic technology."

I could probably go on and on about all the cool robots I saw at RoboBusiness this year, but there isn't enough space. For those of you who don't know me well, I am one of those guys who talks and talks and never shuts up, especially when it is a topic I am enthusiastic about — like robotics. My business partner Bob often jokes that the only time he ever sees me get quiet is after four days wagging my jaws at the Consumer Electronics Show.

Regardless, if you haven't ever gone to RoboBusiness, it is worth the trip next year to Pittsburg for RoboBusiness 2008 (**www.roboevent.com**). If the doubling size trend continues there should be enough robots there to satiate even the most avid robot enthusiast. I know I will try and be there for sure! **SV**

About the Author

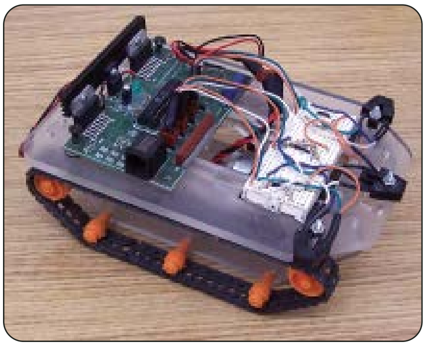
Ted Larson is the CEO of OLogic, Inc., and an active member in the Home Brew Robotics Club of Silicon Valley. OLogic is an embedded systems research and development company with a focus on robotics. OLogic is currently working with clients across a wide spectrum of application domains such as consumer electronics, toys, medical products, and education.

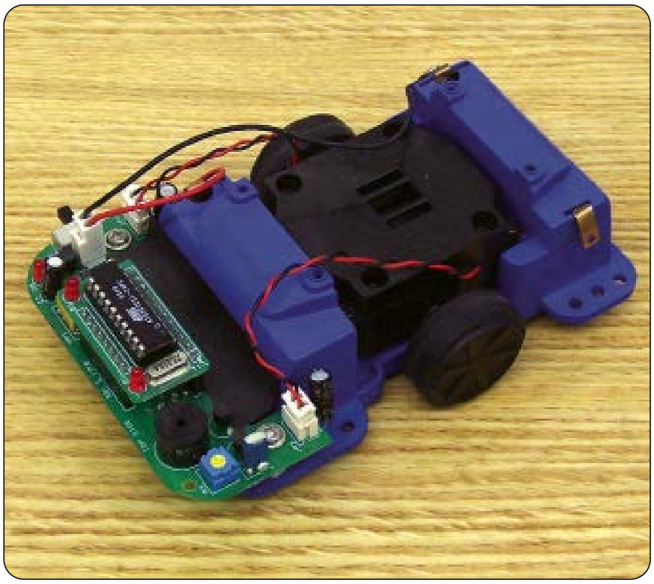
Robotics Showcase

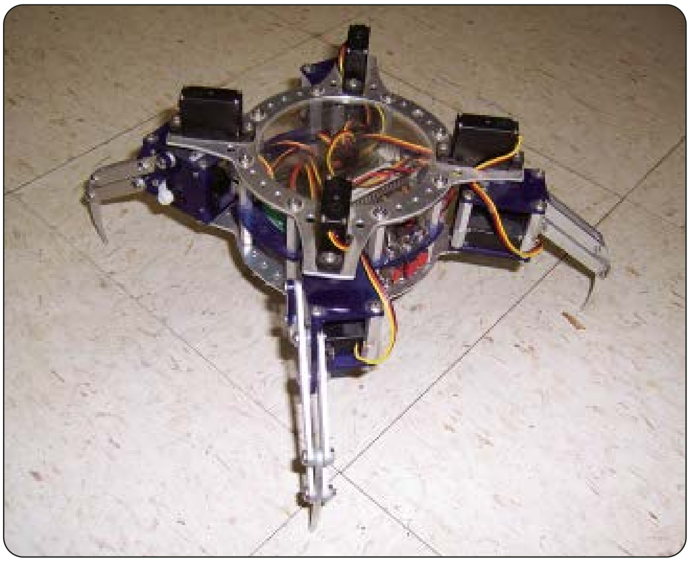
MENAGERIE

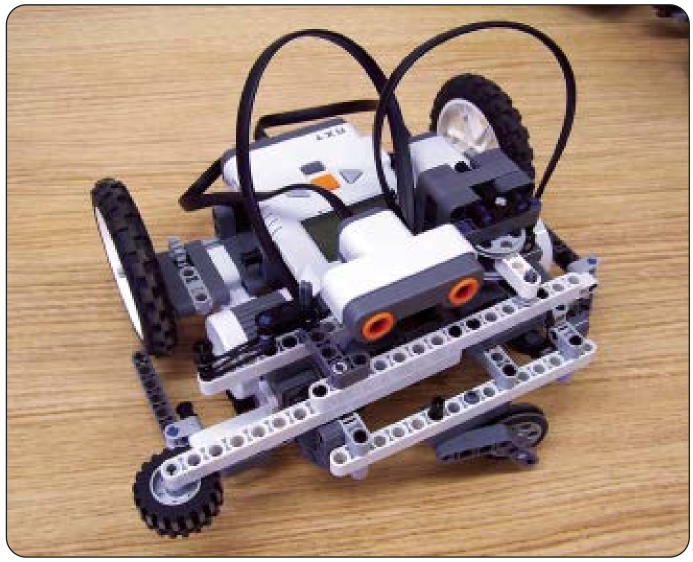
ROBOMO MAZE 2007 COMPETITION

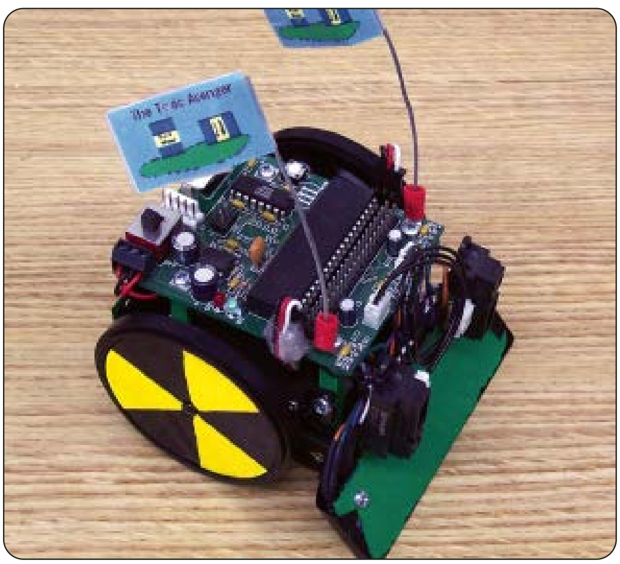
Sponsored by the St. Louis Area Robotics Group
www.robomo.com











GOING BRUSHLESS!

by Pete Smith

Our 12 lb. combat robot "Surgical Strike" was designed to use a cheap (less than $4) surplus "HTI" Speed 700 sized motor to drive the 13" steel blade (seen on the left in Figure 1). We had some initial success at Motorama 2006 where we took third place in a field of about 30 bots.

After that, we got greedy and fitted a 15" titanium blade with heavy steel teeth. This proved too much for the motors and we were burning them out with an average of just more than one a fight!

We wanted the bigger hit that the long Ti blade would give us, but not at the expense of losing fights when the motors fail. This meant we either had to gear down the blade speed a bit to reduce loading on the motor or we needed a more powerful motor.

It looked like we would have to change the blade gearing a little to get the motor failure rate to less than one every three minutes (the length of a combat match). The downside of doing this was that kinetic energy varies by the square of the rotational speed, i.e., a blade turning half the speed has only one quarter the kinetic energy. We could have ended up being able to use the bigger blade but with even less hitting power than before.

The usual solution for this problem is to change to a brushless motor with their higher power and greater efficiency. The problem with this has always been the high cost of both the motors and the special speed controllers that they require.

Motors like the excellent Axi 2826/12 and suitable 70 amp Jeti "controller cost around $200 a set. This was out of our price range.

We were more or less resigned to having to change the gearing or go back to the shorter steel blade when someone on the Robot Fighting League posted a link to a Hong Kong company (**www.unitedhobbies.com**) that was selling similar motors and controllers for about a third of the cost. Brushless motors were now definitely in our price range.

We had several factors to take into account when choosing a motor from the broad range available. First, we wanted the blade to turn as fast or faster than before; we wanted extra power to reduce spin up time and reduce motor stress; and finally, as a bonus, we wanted the motor to weigh less so it would free up weight for either a heavier blade or some protective armor.

The original motor specs were:

Kv	**1,260 RPM/volt**
Power	**562 watts (at 18V, the operating voltage of the robot)**
Weight	**11oz**
Efficiency	**73%**
Cost	**$4.00**

(Source www.robotcombat.com)

The most likely brushless candidate was the HXT 42-50A (seen on the right of Figure 1) which has the following specs:

Kv	**700 RPM/volt**
Power	**630 watts (at 18V) to over 1,000W**
Weight	**7oz**
Efficiency	**83%**
Cost	**$40**

Reviews indicated that the motor would handle a lot more power than we were asking of it, so that would improve the reliability enough to allow it to keep running through a reasonable number of fights.

The one major difference between the motor's specifications is the Kv figures. Kv is a constant that indicates how fast a motor will rotate for every volt applied, i.e., the RPM at 18V = 18 * Kv.

This means that the HTI brushed motor rotates at 22,680 RPM @ 18V while the brushless HXT only turns at 12,600 RPM at the same voltage.

The original drive for the motor

FIGURE 1

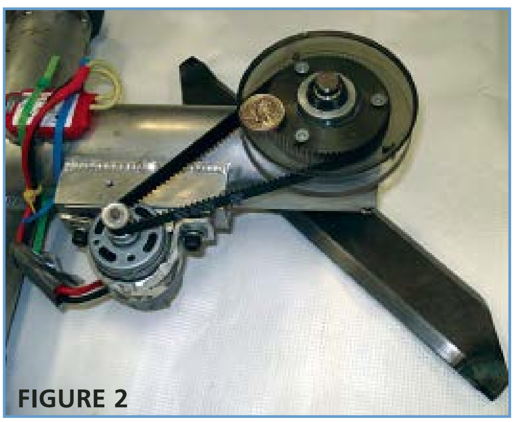


FIGURE 2


FIGURE 3


FIGURE 4

(Figure 2) had a 12 tooth pulley on the motor and a 72 tooth pulley on the blade axle. This gave a ratio of 6:1 and hence theoretical blade speed of about 22,680/6 = 3,780 RPM. (In reality, friction, drag, etc., will reduce this figure somewhat.)

In order to keep the kinetic energy of the weapon the same as before, we would need to change the gearing to allow the slower brushless motor to turn the blade at a similar speed as it did previously. This new gear ratio is calculated by dividing the brushless motor's speed by the required blade speed i.e., 12,600/3,780 = 3.33 to 1.

The change in gearing ratio can — in theory — be done by increasing the teeth on the motor's pulley to 72/3.33 = 22 teeth, or by reducing the number of teeth on the weapon pulley to 12 * 3.33 = 40 teeth. It would have been easier to change the motor pulley but we could not get one with 22 teeth and a 5 mm bore to match the motor shaft. This meant that we would have to replace the custom pulley that was on the axle.

I ordered two of the HXT motors with matching 60 amp ESCs from United Hobbies for a total of $160, and two 40 tooth 3 mm HTD pulleys and a couple of new shorter drive belts from **www.sdp-si.com**

When the new motors arrived, they were a different color than I expected (the ones on the website are green). They had the correct Kv marked on them, so they appeared to be the correct model.

I cut a new motor mounting bracket from a piece of 3/16" aluminum angle (Figure 3) that mounted the motor at the correct height relative to the axle and the right center-to-center distance to match the belts I had ordered.

It is possible to get flanged 40 tooth pulleys but the flanges available are small and flimsy, so I ordered flangeless ones and then turned some flanges on my lathe from a suitable piece of round aluminum bar. The top one was a press fit and the bottom one was attached by a couple of 6-32 screws.

The completed pulley (Figure 4) was a lot smaller and lighter than the old ones and, incidentally, a lot less vulnerable to damage. This brought the total weight savings to about 1/2 lb.

The new speed controllers (Figure 5) had three output wires rather than the two you have on a conventional (brushed) ESC. The motor came with the correct gold plated connectors and these were soldered to the output wires. The other connections are to the battery and to the receiver as normal. This ESC did not have a battery eliminator circuit. There was a slightly complex setup procedure for the speed control that involved a lot of loud "beeps" (this can be downloaded from the website) and we configured it as follows:

Battery type	**NiCad/NiMH**
Brake	**Off (it is only designed for stopping a light propeller)**
Timing mode	**Three degrees (the lowest setting)**
PWM	**8 kHz (the slowest)**

I need to do some research into what the ideal settings for the timing mode and PWM would be, but the above proved to work well.

After rebuilding the robot with the new pulley and belt (Figure 6), we had our first test of the new motor. This was disappointing as the motor emitted a loud squealing noise and had difficulty getting the blade turning. We stopped the test and noticed that the motor body had seemed to have moved relative to the axle. I then remembered one of the reviewers had mentioned that they had had to tighten a small setscrew because of a similar problem. I found the setscrew (Figure 7) and tightened it after adding a little drop of green "Loctite 290" (it keeps set screws tight without making them too difficult to loosen if this becomes necessary).

We had also noticed that the blade was turning in the wrong direction so we reversed the motor's rotation simply by swapping two of the leads. A second test showed a dramatic improvement. The blade spun up very quickly and seemed to have a higher top speed, as well.

We have only had one fight to-date using the new motor and it performed very well, exceeding our expectations, before an unrelated mechanical issue lost us the fight. You can see that and other fights in the "Competitions" section of our website at **www.teamrollingthunder.com**

In conclusion, it appears that the new, cheap brushless motors are going to have quite a future in combat robotics. They are lighter, more powerful, and now affordable, as well. **SV**


FIGURE 5


FIGURE 6


FIGURE 7

WEB-Based Telerobotics

"A key feature of web-based telerobotics is accessibility ..."

FIGURE 1. Wireless telerobotic controller atop a Parallax Boe-Bot.

by Bryan Bergeron

Accessibility, in this case, means that students — as well as seasoned roboticists — can interactively explore new environments from their PDA, laptop, or home PC. Furthermore, as more cities become free WiFi zones, there are new opportunities for wireless web-based telerobotics for practical applications ranging from security to monitoring elderly parents.

Thanks to the availability of inexpensive WiFi webcams, creating a robot that can be controlled and monitored via a web browser is no longer an engineering feat that requires extensive knowledge of WiFi hardware and signaling protocols, video signal interfaces, and low-level programming languages. This article describes how you can repurpose a common wireless webcam to create a web-based telerobotics controller for the Parallax Boe-Bot or any other robotics platform of your choice.

Wireless Webcams

Wireless webcams are commodity items, with affordable systems available from Panasonic, Sony, Motorola, Linksys, D-link, Logitech, SMC, Creative, Hawking Technology, and Grandtec, among others. Prices for "home units" range from about $100 for units with basic video capabilities to about $900 for cameras with optical zoom, motorized tilt-pan, and two-way audio. Commercial units with high resolution video, weatherproof housings, advanced motion control, and bundled security software cost considerably more.

While wireless video-only webcams can add remote vision capabilities to robots operating within the vicinity of a WiFi access point, cameras with remote tilt-pan can be used to create a telerobotic control system. Instead of directing the optics of the camera, the control signals can be used to command the drive mechanism of a carpet roamer, battle bot, or hardened security robot.

Configuring a wireless webcam as part of a closed security system is generally straightforward, even for the units with advanced features. However, making a webcam globally accessible — such as the webcams showing the skylines of major cities and ski slopes (e.g., **www.globecams.com**) — can be expensive and difficult.

Most routers assign webcams a dynamic IP address by default. A benefit of this approach is that computers, printers, webcams, and other networked devices are plug-and-play. Every time a device on a plug-and-play network reboots, it requests an IP address from the router, which assigns the device an open IP address. In this scenario, the IP address associated with a webcam, as viewed from the outside world, changes over time.

If you have experience configuring and managing complex networks, it's possible to configure a PC and other networked devices for static IP addresses. However, a more reliable — and more expensive — option is to pay your ISP for a static global IP address. An increasingly popular alternative is to use a commercial domain name service that provides access to webcams through a static, globally accessible IP address, regardless of the dynamic IP address assigned to the webcam.

Panasonic BL-C30

Panasonic offers several webcams that support remote pan-tilt operation, as well as a free dynamic DNS service, **Viewnetcam.com**. The service enables you and your guests to interact with your webcam through a globally accessible URL that is password protected. For example, the URL associated with my camera is **WebRobot.ViewNetCam.com**, regardless of changes in my local IP address due to rebooting or adding devices to my network. However, you won't be able to view the video or activate the mechanical tilt-pan or image zoom controls without the appropriate username and password.

The camera used for this project — the Panasonic BL-C30 — is available for about $250 from Amazon. This "home use" camera, shown in Figure 2, has a thick 75 mm x 75 mm (3" x 3") footprint and weighs about 200 g. The 12V @ 750 mA brick power supply is used for testing and setup, but not during operation.

The BL-C30 is rated up to a 640 x 480 resolution, 1 Lux sensitivity, and seven frames/sec at 640 x 480. Even though the webcam supports 802.11b/g at up to 54 Mbps, I achieved an actual throughput of about two frames/second using a Linksys WRT54G wireless router. Frame rate was about four frames/second with the webcam connected to my router with a Cat 5 patch cable. Based on my review of similar webcams, this performance is typical of non-professional (i.e., affordable) webcams.

The camera and free DNS service is rated for up to 20 simultaneous viewers — an important consideration for classroom demonstrations. Additional features that may be useful in robotics applications include a privacy button, image buffering, and a built-in PIR sensor. You'll need Internet Explorer 6.0 or above and a Windows environment to operate the browser controls and view the camera output.

FIGURE 2. Panasonic BL-C30 and a BASIC Stamp for size comparison.

Construction

Repurposing the BL-C30 involves disassembling the unit, disconnecting the two tilt-pan stepper motors, and feeding the stepper control signals to a BASIC Stamp. For the operation, you'll need a small Phillips screwdriver, solder braid, and a temperature-controlled soldering iron. If you're careful during disassembly, you can reconnect the stepper motors later and use the unit as a standard webcam.

Before removing the two screws holding the back cover in place, first install the webcam and verify that it works as advertised. Follow the instruction manual and turn off your antivirus software, load the control program, verify that your router settings are correct, and register your camera with the **ViewNet.com** service. I found the wired setup painless, but had to upgrade my router firmware for wireless operation. Be sure to make use of the online documentation, especially when it comes to configuring your router/wireless access point.

When you're satisfied that the camera works as advertised, carefully dismantle the unit with a small Phillips screwdriver. Figure 3 shows the disassembled BL-C30. You don't have to tear down the unit to this level for the modification. The objective is to gain access to the stepper motor solder connections, as shown in Figure 4.

Using the solder braid and soldering iron, carefully remove the flexible circuit cables from the stepper motors. Note that the smaller, vertical stepper motor assembly (shown on the right in Figure 4) has a curved cable trace that connects to the cable carrying video signals.

If you plan to use the webcam only for telerobotics, then consider removing the stepper motors from the unit to reduce the weight of the controller. Removing the motors (34 g) and back cover (60 g) reduces the

FIGURE 3. Disassembled BL-C30. The vertical stepper motor is attached to the spherical camera assembly.

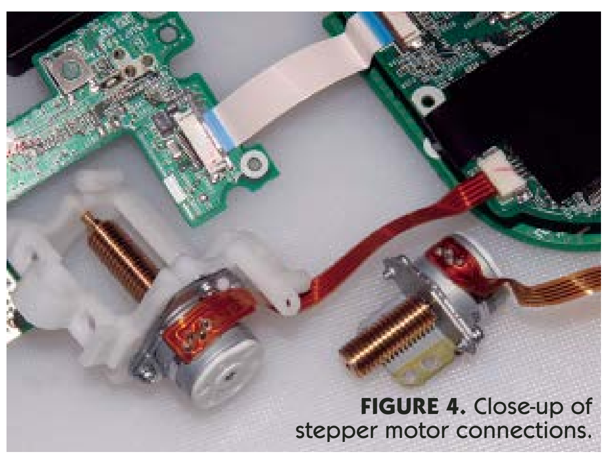


FIGURE 4. Close-up of stepper motor connections.

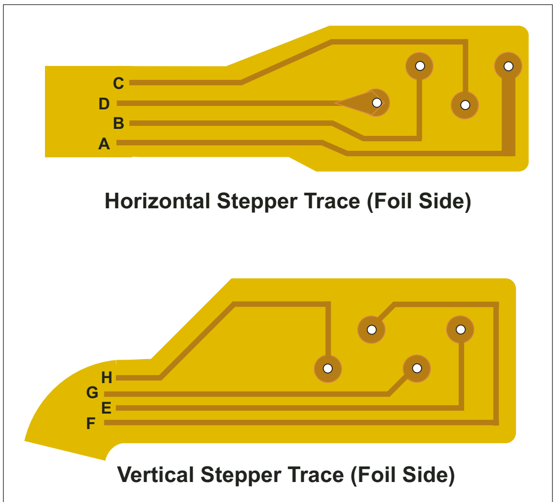


FIGURE 5. Flexible circuit cable traces for horizontal (top) and vertical (bottom) stepper motor connections. Letters correspond to connections in Figure 6.

weight of the unit by 50% (105 g).

Solder one end of a 6", four-conductor ribbon cable to solder pads on each of the stepper motor cables. Consider using multi-colored ribbon cable to avoid confusing the connections, as shown in Figure 5.

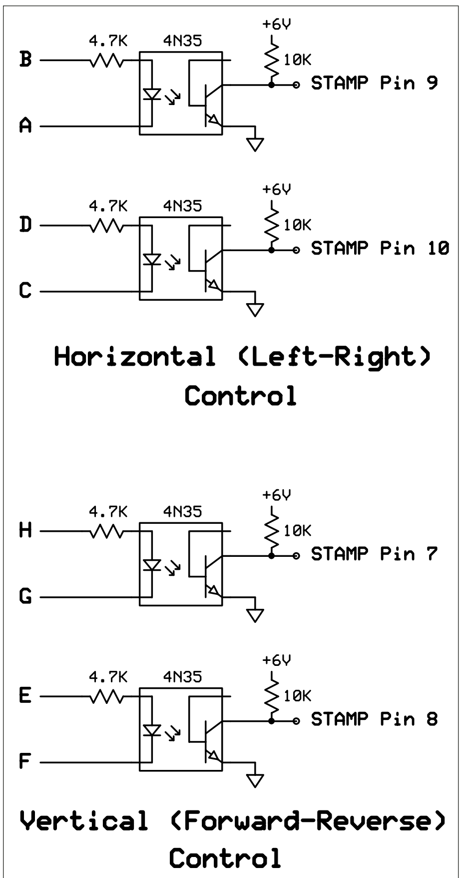


FIGURE 6. Optocoupler circuit for stepper-to-Stamp interface. See Figure 5 for trace connections.

Interface Circuit

Interfacing the 50V peak-to-peak square wave signals from the stepper motor drive circuitry to the Boe-Bot's BASIC Stamp is accomplished with the optocoupler circuit shown in Figure 6. The 4N35 optocoupler is a six-pin DIP chip that incorporates a gallium arsenide LED and silicon NPN phototransistor to provide over 5 KV of isolation. Four 4.7K, 1/8 watt resistors limit the current through the LEDs and four 10K, 1/8 watt resistors serve as pull-ups for the Stamp pins. When an LED in a 4N36 is excited, the phototransistor conducts, bringing the Stamp pin low.

As shown previously in Figure 1, the BL-C30, optocouplers, and associated components fit nicely on a circuit board fitted as a second tier on the basic Boe-Bot chassis. You can hard-wire the power and signal connections to the circuit board, but consider Molex connectors if you plan to use the controller on multiple robot platforms.

You'll have to provide 12V at 300 mA for the BL-C30, as well as 6V at 250 mA for the Boe-Bot. I use two four-cell banks of AA NiMH batteries wired in series and tucked under the Boe-Bot for power. Four cells power the Boe-Bot and eight power the BL-C30. A heavier and bulkier alternative is to use separate 12V and 6V supplies, but this is pushing the capabilities of the servos on the Boe-Bot. Of course, dual 7V NiMH packs and DC-DC converters provide ample power on platforms capable of supporting the bulk and weight.

Stepper Motor Basics

The crux of this application is interpreting the stepper motor control signals. Figure 7 shows the phase sequencing of the unipolar stepper motors used in the BL-C30. If you attach a dual-channel oscilloscope to the vertical or horizontal stepper motor connections and click on the browser controls, you'll see that the motors are energized by pulse trains with the leading pulse either positive or negative, depending on which button is pressed.

In addition, the pulse trains are of constant duration — about 500 ms for the vertical stepper and 800 ms for the horizontal stepper. So, if we detect the direction of the first pulse in a train and ignore the remaining pulses for the duration of the train, we can determine the intended direction of the stepper motor and use the signal to control the Boe-Bot's drive servos.

Returning to the schematic shown in Figure 6, note that the connections to the 4N35 optocouplers are polarity specific. For example, pin 9 of the Stamp will be forced low when connection B is positive relative to connection A, and pin 10 will be forced low when D is positive relative to A.

Code

The bulk of the BASIC Stamp code required to interface the BL-C30 to the Boe-Bot's Stamp is shown in Listing 1 (the full listing is available on the *SERVO* website at **www.servomagazine.com**). The main loop continuously checks the current state of the robot (State) and the input from

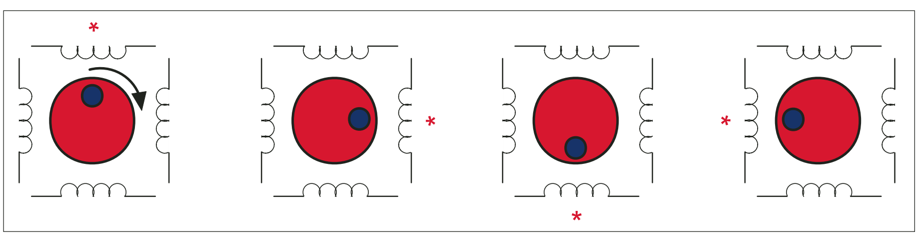


FIGURE 7. Phase sequencing of unipolar stepper motor.

FIGURE 8. View from controller at 1X magnification and 640 x 480 resolution.

the wireless controller (Right, Left, Front, or Back). If the robot is stopped (State = S) and the optocoupler circuit indicates an initial left event (Left = 0), then the state of the robot is changed to left (L), and program execution pauses for 800 ms — the duration of the remainder of the pulse train.

The Motor subroutine uses the current state to direct servo motor activity, using the standard Pulsout directives as described in the Boe-Bot manual. In the code snippet shown, a left turn is accomplished by pulsing the two Boe-Bot servos with identical signals. Because the servos are mounted facing away from each other, one servo turns clockwise while the other turns counterclockwise. Modify the Motor subroutine to drive other microprocessors and robotic platforms.

Operation

Operating the telerobotic Boe-Bot is straightforward. Using the browser-based control, as shown in Figure 8, click on the up arrow to move the Boe-Bot forward. Click on the down arrow or center button to stop. Click on the right button for a right turn, and on the left or center button to stop. Click and drag on the image to trigger the 1X-to-10X digital zoom feature.

I configured my Boe-Bot to move one track forward and one in reverse for tight turns, but you can determine the rate and type of turning to suit your particular platform and operating environment. Because of the lag time associated with waiting for the pulse trains to end and the relatively low frame rate, you have to compensate the degree of turning and stopping distances by clicking on the center stop button a second before reaching your target.

From Here

Consider this simple webcam controller as a basic building block in your telerobotics arsenal. An obvious addition is local intelligence, such as object detection, to supplement teleoperation. You can experiment with the PIR detector by feeding the signal to the Stamp through another optocoupler. Extend the simple logic to respond to multiple forward, reverse, or turn clicks for incremental turning, movement at different speeds, or to control grippers and other effectors.

A more capable webcam opens up additional control possibilities. For example, with a webcam with two-way audio — such as the D-Link SCS-3220G and Panasonic BB-HCM371A — you can experiment with sending audio control signals to your robot. An LMC567-based tone decoder on the robot can provide dozens of additional channels that can be used to control robot effectors, respond to onboard events, and provide emergency shutdown. Ramsey Electronics offers an inexpensive ($10), lightweight tone decoder/encoder kit based on the LMC567 that is suitable for this application. **SV**

Resources

Lindsay, Andy. *Robotics with the Boe-Bot.* Parallax.
www.parallax.com

Panasonic BL-C30 Web Camera Documentation.
www.panasonic.co.uk/ip-cameras/blc30.html

TD1 Dual Tone Decoder-Encoder Kit. Ramsey Electronics, Inc.
www.RamseyKits.com

Listing 1

```
Main:

    IF (STATE = S) AND (Left = 0) THEN
         State = L
         PAUSE 800
    ENDIF

    IF (STATE = S) AND (Right = 0) THEN
         State = R
         PAUSE 800
    ENDIF

    IF (STATE = S) AND (Front = 0) THEN
         State = F
         PAUSE 400
    ENDIF

    IF (STATE = S) AND (Back = 0) THEN
         State = B
         PAUSE 400
    ENDIF

    IF (STATE = R) AND (Left = 0) THEN
         State = S
         PAUSE 800
    ENDIF

    IF (STATE = L) AND (Right = 0) THEN
         State = S
         PAUSE 800
    ENDIF

    IF (STATE = F) AND (Front = 0) THEN
         State = S
         PAUSE 800
    ENDIF

    IF (STATE = B) AND (Back = 0) THEN
         State = S
         PAUSE 800
    ENDIF

GOSUB Motor

GOTO Main


Motor:
    .
    .
    IF STATE = L THEN
         PULSOUT RightServo, 650
         PULSOUT LeftServo, 650
         PAUSE 20
    ENDIF
    .
    .
Return
```

HOW TO MAKE A SMALL CIRCUIT BOARD USING IRON-ON RESIST

by Alan May

This article shows how anyone can make circuit boards with iron-on resist in just a few minutes. For a number of years, I had poor results with iron-on resist, but not long ago I stumbled on an approach which produces near perfect circuit boards every time.

Why I Make Circuit Boards

When I get going on a project, I want to keep going until I am done. I have never been interested in sending out for a circuit board because of the cost and delay. I don't like wire wrap as it is too hard to trace and repair (for me, anyway).

Wire wrap boards always need a lot of repair as I keep making mistakes when I try that system. I have the same problem with point-to-point wiring using a perforated board.

But I can check and recheck computer artwork. You can print it out and check to see that all the parts will fit. If you do make a mistake, you can fix it without starting over. And you can easily duplicate any project. Photo fixed resist will work with computer and magazine patterns, but trying to work in a dark room, etc., intimidates me.

What You Need

I have had excellent results using Press-N-Peel Blue from Techniks, Inc., iron-on film for etching resist (**www.techniks.com**).

Tools needed include a household iron, a piece of smooth wood for an ironing board, regular and double sided cellophane tape, iron-on resist film, access to a laser printer or copier, etchant (ferric chloride), and a plastic dish for etching. And, of course, one sided blank copper circuit board. You also need some sort of drill press to drill the board.

Techniks sells another type of film (Press-n-Peel Wet), but I have never tried it as it costs the same, and Techniks suggests it produces "hobby quality" work. I don't want that, even if the rest of my work is hobby quality. Press-N-Peel Blue costs about $30 for 20 sheets. You can get smaller packages other places for about $2 per sheet. All Electronics offers five sheet packages. Using a "carrier" as described next conserves film so that the cost of film used should not be an issue. Etchant is widely available from catalog sources.

I have only tried small boards with this process, the largest being about 3" by 4", but I would not be afraid to make a larger board.

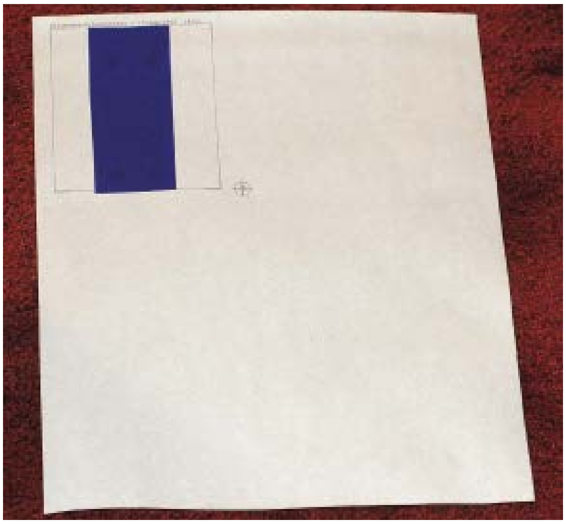


FIGURE 1. The film is pasted to the paper carrier using double sided tape and covers the pattern printed on the first pass through the printer.

Quality Control

When I think I am done with a circuit board design, I print the top and bottom views together, as large as possible. I trace over each section of the circuit with different colored highlight pens to make sure I have it right. Designing the copper pattern is the hard part as far as I am concerned.

I make a paper film carrier to save film. When the artwork is right, print (on paper) an x-ray positive view of the copper, actual size. This is the same as a mirror image of the bottom view. You must use a laser printer. If you are making a top side, surface mount style board, print a mirror image of the top view pattern.

Leave at least an inch between the pattern and the top of the page. This printed paper is the film carrier. Put a piece of double-sided cellophane tape on the paper a little above the pattern printed on the first

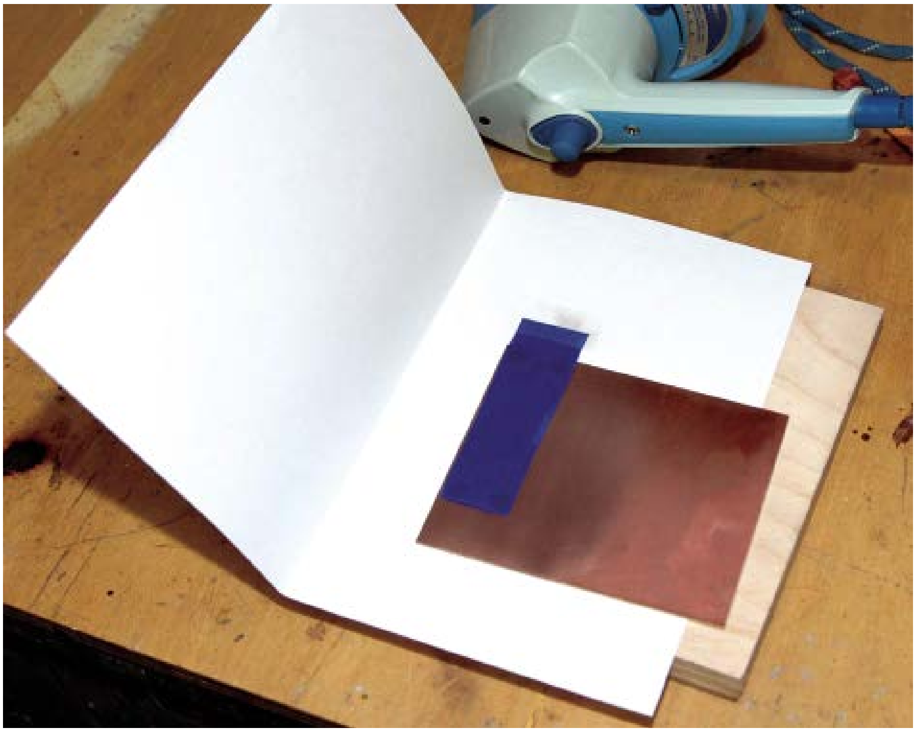


FIGURE 2. The paper, film, and blank board "sandwich" ready for ironing. The folded paper keeps the iron from touching the film.

pass. The tape should not be as long as the pattern is wide.

Cut a piece of iron-on film a little bit larger than the pattern on three sides, and with 3/4" extra at the top. Stick the film — rough side up — on the tape you put on the carrier paper. Be sure the pattern printed on the first pass and the tape are covered by the film (see Figure 1). Run it through the printer or copier again. If you haven't changed anything in your computer or moved the original on the copier, the printout should be pretty close to where it was on the first pass, and the pattern should now be on the film. You could run a whole piece of film through, and forget the carrier, but that will use up your film much faster.

Carefully pull the film from the carrier and even more carefully remove the two sided tape from the film. Trim the film close to the pattern along the sides and bottom, but leave the extra space at the top.

Transfer the Pattern

Fold a piece of printer paper in half, and stick the prepared film pattern side down on one side inside the fold using regular cellophane tape at the top where there is no pattern. Rub off any "stickum" that may be on the top of the tape. Make a reference mark on the paper, even with the top of the pattern, to either side of the film to aid in positioning the blank board.

Turn the iron on to heat up while you prepare the blank circuit board. I use an old travel iron I liberated from my wife's travel supplies. I set the temperature at just below the wool setting. I learned the hard way that a higher setting can blister the copper. Find a smooth scrap of wood for use as an ironing board. (Actually, any smooth flat surface should do.) I have never scorched anything (paper or tape) during this process, but I avoid the good furniture, anyway. I use a scrap of 3/4" birch plywood.

Clean and polish the copper side of blank circuit board with sand paper, steel wool, or a Scotch Bright pad. Get it really shiny. I don't cut the board to size first. If you do, be sure that the rough cut edges are filed smooth so the board is completely flat.

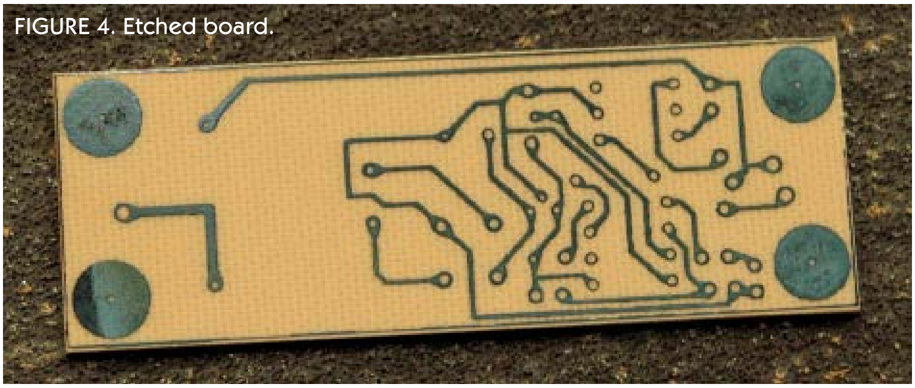
Next — this is important — wash the copper side of the board with a powder type household cleanser and warm water. Use one with some abrasive quality. I have had success with Bon-Ami and Boraxo hand cleaner. I have tried cleaning the board with alcohol, paint thinner, and lacquer thinner, but with poor results. I have never tried a liquid household cleaner.

You will note that the cleanser turns dark when you scrub. I just use my fingers. One application is enough, but two won't hurt. I believe that this step is the secret to success. I never got really good results until I washed the blank board with cleanser. Dry both sides thoroughly.

Now slide the board under the film, copper side up, lined up with the pencil reference mark, and fold the paper over carefully. The paper sandwich keeps the film from moving during the ironing process. Also, my

FIGURE 3. Etching setup.

FIGURE 4. Etched board.



iron tends to stick to the film. Be sure there is no grit anywhere on top of the film or either side of the top section of the paper (Figure 2).

Now press the iron on top of the paper, film, and circuit board sandwich you have created. Press pretty hard, say 10 pounds worth. You want to get the board hot. I move the iron around slowly because I don't trust it to heat evenly. Heat and pressure are all that are necessary.

Lift the top paper and peek at the film. When transfer is complete, you can see the pattern on the slick side of the film. If you don't see it or parts are missing, keep ironing. The board edges seem to be the last to finish. Perhaps they are harder to heat, or cool quickly. If the paper scorches, the iron is probably too hot.

Too much heat and pressure can “squish” the traces, so some moderation is appropriate. This has not been a problem for me once I learned the best heat setting for my iron. One minute of ironing should be enough for a small board. If the first try doesn’t work, clean off the board and try again with a new piece of film, using more heat and pressure.

Dip the board in a container of water. This is not for any part of the process, but because the board is hot. If you are on track, the board will be too hot to handle. Peel the paper off, and admire your work. If one or two traces are weak or missing, repair with liquid or press-on resist. I have had very poor results from felt tip resist pens.

If any traces have been pressed too wide, scratch needed clearance with a sharp pointed tool. I have used a ruler the few times I have had to do this.

Trim and Etch

Trim the board if you have not already done so. Several years ago, I bought a small (8”) sheet metal bending break which included a shear, thinking I could make my own enclosures. So far, I have not mastered that. It is great for trimming circuit boards though, and bending stock for small brackets and cutting small metal sheets. Before I acquired this device, I used a small hand saw, and cleaned up the edges with a file and coarse sandpaper.

If there are any large areas that don't need to be etched away and are not coated with resist, cover them with a piece of electrical tape to save etching solution.

I use a little tray I saved from a TV dinner to hold the etching solution. To speed up the process a little, I heat it with a light bulb (Figure 3). The board will be finished in about an hour.

I use ice cream sticks to lift the board to monitor progress. Be aware that any etchant spattered on your clothes (or anything) will likely leave a permanent spot. When etching is complete, rinse the board with water. Put the solution back in the bottle with the help of a plastic funnel. I let it cool before I tighten the top, or the bottle deforms as the solution cools. Figure 4 shows the etched board with the resist still in place.

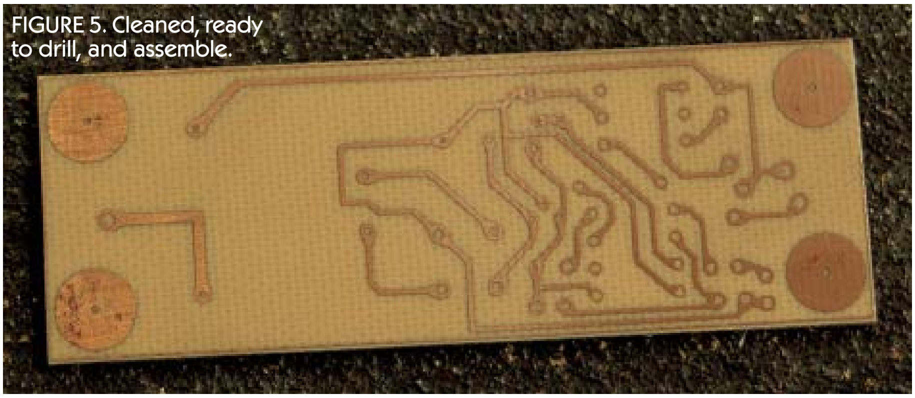
Remove the resist with an abrasive or with lacquer thinner. If you drill before removing the resist, the burrs left by the drill will make the resist harder to remove. Figure 5 shows a finished board.

Drilling

I drill with a full size drill press, but any type of drill press should do, including those that hold a hand-held engraving tool like a Moto or Dremel. I don't think I could manage very well with a hand-held drill because the holes have to be accurately placed. I bought a bunch of reconditioned circuit board drill bits I saw advertised somewhere. They work fine and are cheap.

This process is so easy I almost never use any other approach. I am sure you will feel the same way. Just don’t forget to wash the blank board with cleanser and always place a piece of paper between the iron and the film. SV

FIGURE 5. Cleaned, ready to drill, and assemble.



DIFFERENT BITS

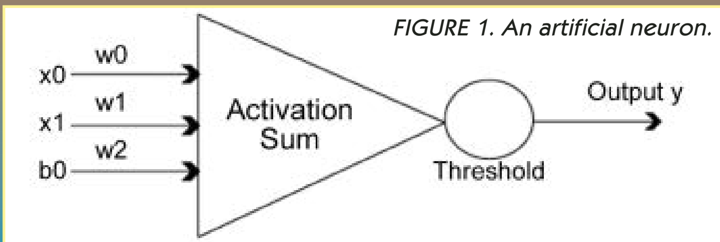


FIGURE 1. An artificial neuron.

NEURAL NETWORKS FOR THE PIC MICROCONTROLLER

PART 1 — PERCEPTRONS

by Heather Dewey-Hagborg

Neural networks are a machine learning technique that simulates capabilities of the biological brain. Also referred to as connectionist models — or parallel distributed processing — neural networks are systems of simple processing units called neurons connected by links called weights which amplify or diminish a charge passing through them.

Weights behave similarly to variable resistors, and adjustment of the values of this complex system of connecting weights allows higher level intelligence to arise from very simple low-level interactions.

Neural networks are traditionally good at tasks like pattern recognition, classification, time series prediction, and optimization, but also can form the basis of content addressable memories and psychological models. In this article, we will look at some very simple pattern recognition and classification algorithms as an introduction to the topic. Later, we will progress to memory algorithms and a discussion of the possibilities of creating robots with built-in psychological models.

What is a Perceptron?

The fundamental component of any neural network is the artificial neuron (Figure 1). Artifical neurons consist of a set of fairly standardized components derived from the structure of biological neurons (Figure 2):

- *Inputs (x0, x1)*. Sensors or signals from other neurons in the system. This simulates the synapses of a biological neuron.

- *Bias (b0)*. An additional input to the neuron unconnected to any environmental stimulus or other neuron. The bias allows more versatile and fast programming.

- *Weights (w0 – w3)*. The strength or resistance of connections between inputs and the neuron. These resemble a biological neuron's dendrites.

- *Activation.* Simply the sum of all charges passing through each input weight to the neuron.

- *Threshold Function.* A second step in the neuron which takes the activation sum and thresholds it in some way to provide the final output of the neuron.

- *Output*. Each neuron has exactly one output which is the activation sum passed through the threshold function. Though there is only one output, it can branch off to form an input to several other neurons. This is the electronic analogy of a biological axon.

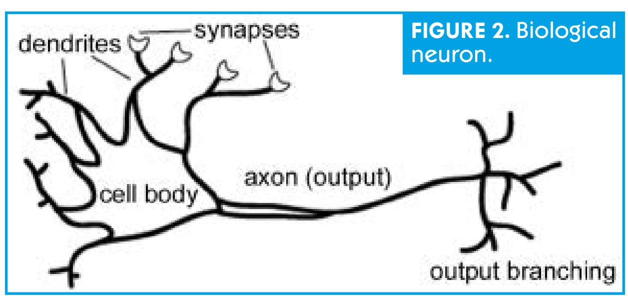


FIGURE 2. Biological neuron.

Let's start with a concrete example called a perceptron based on Figure 1. Take the following input values:

input x0 = 1
input x1= -1
bias b0 = .5

and randomly initialized weight values:

weight w0= .613
weight w1= .033
weight w1= .236

The first thing we do is find the activation sum of the neuron. This is the total sum of each input multiplied

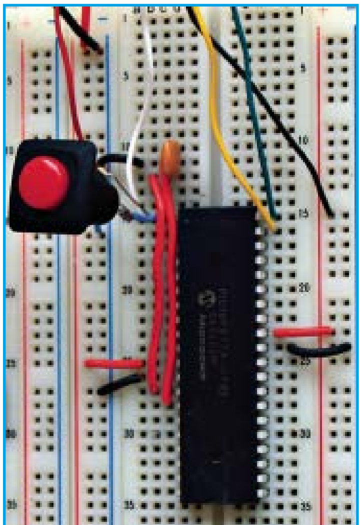


FIGURE 3. PIC chip setup.

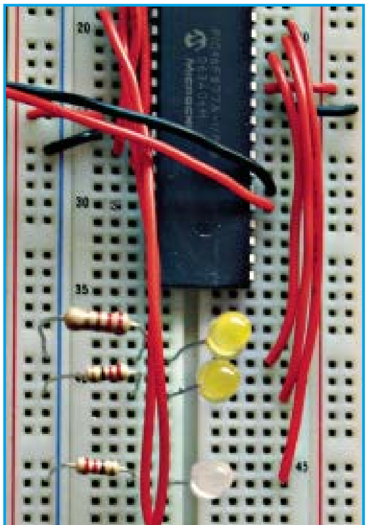


FIGURE 4. Add the LEDs.

by its corresponding weight value. In our example:

activation = (x0*w0)+ (x1*w1)+ (b0*w2)
activation = (1*.613)+(-1*.033)+(.5*.236)
activation = .7

Next we pass this activation value through a hard limiting, threshold function which simply says that if the activation level is above value t, the output is 1; otherwise, the output is -1. In our example, we will set the threshold variable t to 0, yielding a final output value of 1.

So how does this artificial neuron actually compute anything useful? This is where learning comes in. If we know in advance what the output should be, we can gradually adjust the weight values of the neuron until we get the correct output answer. By iterating through sets of input and output pairs, we can then teach the neuron how to respond to various input patterns, and it can then extrapolate from this to categorize new input combinations correctly.

Imagine that we want our neuron to learn to perform a specific logic function, for example, the logical AND function. The perceptron truth table would look like this:

x0	x1	output
1	1	1
-1	1	-1
1	-1	-1
-1	-1	-1

In this example, we gave the neuron an input of 1,-1 and it output a 1 which is the incorrect answer. We need to adjust our weight values. We change each weight individually, according to the following rule:

Change in weight = learning rate * (desired output – actual output) * input

Let's start with a learning rate of .1 and apply the rule to our example.

Weight w0:
Change in weight = .1 * (-1 – 1) * 1
Change in weight = -.2

The new value of weight w0 is (.613 – .2) or .413

Weight w1:
Change in weight = .1 * (-1 – 1) * -1
Change in weight = .2

The new value of weight w1 is .233.

Weight w2:
Change in weight = .1 * (-1 – 1) * .5
Change in weight = -.1

The new value of weight w2 is .136.

If we continue to iterate through the truth table and adjust the neuron's weights as we did above, we will eventually settle on a set of weights which allows the calculation to work perfectly. For example:

Weight w0 = .413
Weight w1 = .233
Weight w2 = -.463

With these weights, the neuron is able not only to classify the exact input correctly but also to classify similar inputs correctly. For example, inputs of .7, .8 will still result in an output of 1 and inputs of -.7, .8 will still result in an output of -1.

Microcontroller Implementation

Now that we understand what a perceptron is and how it works, the challenge becomes porting it to a microcontroller. This tutorial will walk you through the simplest possible implementation. We will use a PIC 16F877A microcontroller with two CdS photocells as input, and three LEDs plus the built-in serial port as output. The example is coded in CCS C Compiler (**www.ccsinfo.com**), but should be fairly easy to port to other languages.

First, let's wire up the breadboard, and then we will walk step by step through the code.

Begin by wiring up the PIC chip as usual, including a 10K resistor to positive and a pushbutton to ground on the reset pin (Figure 3). Connect the positive sides of three LEDs to PIC pins RB1-3 and the ground sides through three 220 ohm resistors to ground (Figure 4). Connect the two photocells to the PIC analog inputs RA1 and RA3 (pins 3 and 5) and then on the other side through 220 ohm resistors to positive (Figure 5). Next, get your serial communication ready. If using a

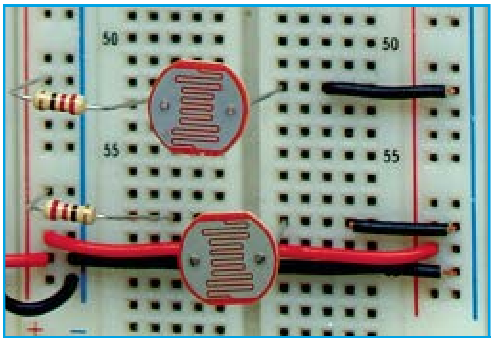


FIGURE 5. Add the photocells.

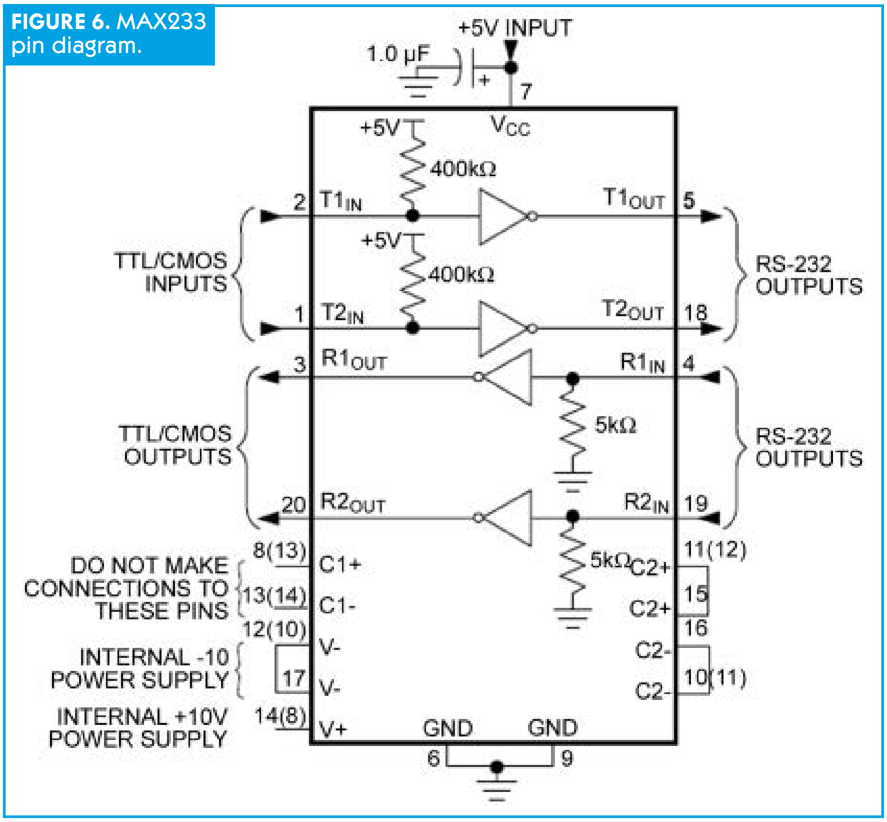


FIGURE 6. MAX233 pin diagram.

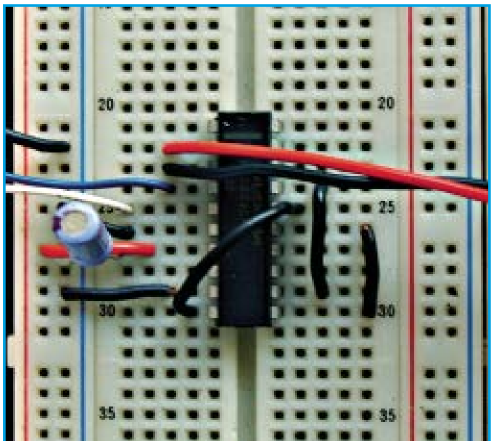


FIGURE 7. MAX233 on breadboard.

MAX233 chip for level shifting, wire it as in Figures 6 and 7 with pin 10 connected to pin 16, pin 11 connected to pin 15, pin 12 connected to pin 17, pins 6 and 9 connected to ground, and pin 7 connected to positive and a 1 µF capacitor to ground. Connect the serial transmit pin of the PIC (RC6 or pin 25) to MAX233 T1in (pin 2). Finally, connect your serial port cable's receive pin to MAX233 T1out (pin 5). When you are finished, your breadboard should look like Figure 8.

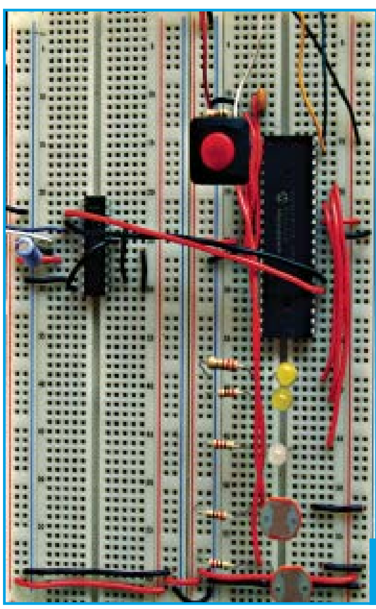


FIGURE 8. The completed breadboard.

Now you are ready to tackle the software. I am going to walk through the C code in small chunks, but the code in its entirety will be available on the *SERVO Magazine* website (**www.servomagazine.com**) in both CCS C and hex formats.

There are two files that make up the perceptron project: the perceptron.c code file and the perceptron.h header file. Let's look at the short header file first.

```
#include <16F877A.h>
#device ICD=TRUE
#device adc=10
#use delay(clock=20000000)
#fuses
HS,NOWDT,PUT,NOPROTECT,NOBROWNOUT,NOLVP,NOWRT,NOCPD
#use
rs232(baud=9600,parity=N,xmit=PIN_C6,rcv=PIN_C7,bits=8)
```

These lines specify the processor type we are using, enable in-circuit debugging, specify that the analog-to-digital converter should use 10 bits, that the clock is 20 MHz, as well as the fuse and serial communication settings. Now let's look at the actual code.

```
#include "C:\Program Files\PICC\projects\perceptron\per-
ceptron.h"
#include <math.h>
//set the ceiling for random number generation
#define RAND_MAX 1000
#include <stdlib.h>
//define our three leds, 2 showing input
// and 1 showing output
#define input0 PIN_B1
#define input1 PIN_B2
#define output PIN_B3
```

The "#include" lines tell the compiler to include standard libraries for functionality in the program. The "#define" lines

```
hi!
random seed: 17
randoms: 107, 320, 125
randoms: -.393000, -.180000, -.375000
original weights: -.393000, -.180000, -.375000
iteration 1
activation: -.760500
thresholded: -1
output: -1
desired output: 1
new weights: -.193000,  .019999, -.275000
activation: -.350500
thresholded: -1
output: -1
desired output: -1
activation:  .075500
thresholded: 1
output: 1
desired output: -1
new weights:  .006999, -.180000, -.375000
activation: -.014499
thresholded: -1
output: -1
desired output: -1
total incorrect: 2
iteration 2
activation: -.360500
thresholded: -1
output: -1
desired output: 1
new weights:  .206999,  .019999, -.275000
activation:  .049499
thresholded: 1
output: 1
desired output: -1
new weights:  .006999,  .219999, -.375000
activation:  .025500
thresholded: 1
output: 1
```

FIGURE 9. The serial print stream from the programmed chip.

define global constants such as I/O pins and the maximum size the random number generator can generate numbers to. Comments are indicated by "//".

```
//public variables
signed int inputs[4][2] = {{1,1},{1,-1},{-1,1},{-1,-1}};
//input sets
int input_length = 4;
signed int outputs[4] = {1,-1,-1,-1}; //correct answers
float weights[3] = {0.0,0.0,0.0}; //weight values
float test_set[2] = {0.0,0.0}; //for testing later
float bias = 0.5; //bias value
float c = 0.1; //learning rate
float t = 0.0; //threshold var
```

DEBUGGING YOUR BEHAVIORAL EQUATIONS

Doing math on a PIC is difficult and confusing, and don't let anyone tell you otherwise! My method usually involves testing each mathematical step one at a time and having the PIC spit the results of its calculations back at me through the serial port. Even with this method, things can get messy due to various PICC printf format options. Take my instructions and code as a guide and play around with variations. The only way to get the hang of PIC math is to make mistakes.

Most of the variables in this program are global and are specified here. This simply allows us to keep the code for this example as simple as possible without going into too many C language specific details.

```
void main() {
   int incorrect=4; //number of incorrect answers
   int count = 1; //number of iterations
   int j;
   signed int o;
   float a;
   long seed;
   long r0, r1, r2;
   float r3, r4, r5;
   setup_adc(ADC_CLOCK_INTERNAL);
   setup_adc_ports(RA0_RA1_RA3_ANALOG);
   set_adc_channel(0);
   setup_psp(PSP_DISABLED);
   setup_spi(FALSE);
   setup_counters(RTCC_INTERNAL,RTCC_DIV_2);
   setup_timer_1(T1_DISABLED);
   setup_timer_2(T2_DISABLED,0,1);
   setup_comparator(NC_NC_NC_NC);
   setup_vref(FALSE);
```

This is the beginning of the main loop of the program. We begin by declaring function specific variables and then call setup functions built in to CCS C to initialize the PIC hardware.

```
   //startup sequence
   output_high(input0);
   output_high(input1);
   output_high(output);
   delay_ms(500);
   output_low(input0);
   output_low(input1);
   output_low(output);
   delay_ms(500);
   //make sure serial transmit working properly
   printf("hi!\n");
```

We go through a short startup sequence, turning the LEDs on and off and sending a test message out the serial port to make sure our hardware is working the way we expect it to.

```
   //init random
   seed = read_adc();
   printf("random seed: %lu\n", seed);
   srand(seed);
```

Here we seed the random number generator by reading an analog pin which is left floating on the chip and print the value to the serial port.

```
   //now initialize weights randomly
   r0 = rand();
   r1 = rand();
   r2 = rand();
   printf("randoms: %lu, %lu, %lu\n",r0,r1,r2);
```

Then we generate three random numbers which we will use for the initial weights in our perceptron and print those, as well.

```
   //turn long unsigned ints into floats
   r3 = ((float)r0-500)/1000;
   r4 = ((float)r1-500)/1000;
   r5 = ((float)r2-500)/1000;
   printf("randoms: %f, %f, %f\n",r3,r4,r5);
   //assign random floats to weight array
   weights[0]=r3;
   weights[1]=r4;
   weights[2]=r5;
   printf( "original weights: %f, %f, %f\n",
weights[0],weights[1],weights[2]);
```

We need floating point numbers between -1 and 1 for our weights, so in this chunk of code we normalize the random integers to that range and print the values to the serial port. Then we assign the values to the global weight array we created earlier.

```
while (incorrect>0){
      printf("iteration %u\n",count);
      delay_ms(500);
      incorrect=0;
      for (j=0;j<input_length;j++){
         a = getActivation(inputs[j]);
         o = getThresholded(a);
         printf("output: %D\n",o);
         printf("desired output: %D\n",outputs[j]);
         delay_ms(500);
         if (o != outputs[j]){
            updateWeights(inputs[j], o, outputs[j]);
            printf( "new weights: %f, %f, %f\n",
weights[0],weights[1],weights[2]);
            delay_ms(500);
            incorrect++;
         }
      }
     printf("total incorrect: %u\n",incorrect);
     delay_ms(500);
     count++;
   }
```

This loop is where the learning takes place. When the perceptron answers incorrectly, we iterate through our training set of inputs and outputs and adjust the weight values according to the learning rule we discussed in the beginning of the article. Along the way, we print our progress to the serial port. In my experience, this loop generally takes 3-10 iterations to complete. We will go over the "getActivation," "getThresholded," and "updateWeights" functions after we finish with the main loop.

When this loop completes, the perceptron has finished learning. Congratulations! From here, there are two different paths we can explore to complete our main loop. The first implements the experiment discussed above, testing inputs which are close to the learning set, but slightly different:

```
   //now test similar input
   printf("input .7, .8\n");
   output_high(input0);
   output_high(input1);

   delay_ms(500);
   test_set[0]=0.7;
   test_set[1]=0.8;
   a = getFloatActivation();
   printf("activation: %f\n", a);
   delay_ms(500);
   o = getThresholded(a);
   printf("output: %D\n", o);
   if (o > 0)
      output_high(output);
   else
      output_low(output);
   delay_ms(500);

   printf("input -.7, .8\n");
   delay_ms(500);
   test_set[0]=-0.7;
   a = getFloatActivation();
   printf("activation: %f\n", a);
   delay_ms(500);
   o = getThresholded(a);
   printf("output: %D\n", o);
   if (o > 0)
      output_high(output);
   else
      output_low(output);
   delay_ms(500);
}//end main function
```

The second option is to make the code dynamic by utilizing our photocells as inputs to the perceptron. To choose this option, we can just comment out the code above by enclosing it in the beginning symbols " /* " and the ending symbol " */ ".

```
//now use dynamic input
//if both photocells are dark turn on
while (true){
   set_adc_channel(1);
   // reuse r0 and r1 variables to save space
   r0 = read_adc();
   set_adc_channel(3);
   r1 = read_adc();
   printf("photocell inputs: %lu, %lu\n",r0, r1);
   //convert 10 bit readings to floats
   //normalized to -1 to +1 range
   test_set[0] = ((float)r0-511)/512;
   test_set[1] = ((float)r1-511)/512;
   printf("photocell float inputs: %f, %f\n",
test_set[0], test_set[1]);
   a = getFloatActivation();
   printf("activation: %f\n", a);
   delay_ms(500);
   o = getThresholded(a);
   printf("output: %D\n", o);
   if (o > 0)
      output_high(output);
   else
      output_low(output);
   delay_ms(500);
}
```

BILL OF MATERIALS

- Breadboard
- PIC16F877A Microcontroller
- 20 MHz Ceramic Resonator with Built-in Capacitors (or equivalent)
- Five 220 ohm Resistors (or similar)
- One 10K Resistor
- Three light emitting dDiodes (LEDs)
- Two CdS Photocells
- One Pushbutton (for reset)
- RS-232 Level Shifter (MAX233 or similar)
- 1 µF capacitor (if using MAX233)
- Serial cable with receive pin available for breadboard use

This is an infinite loop which reads the values from the two photocells and feeds them to the two inputs of the perceptron. It prints the values out the serial port and changes the output LED accordingly.

Now we are ready to look at the functions mentioned above, one at a time.

```
float getActivation(signed int in){
   float ret_val;
   ret_val =  (((float)in[0]*weights[0])+
              ((float)in[1]*weights[1])+
              (bias*weights[2]));
   printf("activation: %f\n",ret_val);
   return ret_val;
}
```

This function simply multiplies each input (including the bias) to the perceptron by its coordinating weight value and sums them all together. It prints and returns the value.

```
float getFloatActivation(){
   float ret_val;
   float temp;
   temp =  ((test_set[0]*weights[0])+
           (test_set[1]*weights[1])+
           (bias*weights[2]));
   ret_val = temp;
   printf("activation: %f\n",ret_val);
   return ret_val;
}
```

This function does the same thing as "getActivation" but with floating point input from the global variable array "test_set."

```
signed int getThresholded(float act){
   if (act > t){
      printf("thresholded: 1\n");
      return 1;
   }
   else {
      printf("thresholded: -1\n");
      return -1;
   }
}
```

This function thresholds the activation value based on the threshold specified in the variable "t." In our example, that threshold is 0, so all negative activations become -1 and all positive activations become +1.

```
void updateWeights(signed int in, signed int out, signed
int des_out){
   float delta0;
   float delta1;
   float delta2;

   delta0 = des_out - out;
   delta0 = c*delta0;
   delta0 = delta0*(float)in[0];

   delta1 = des_out - out;
   delta1 = c*delta1;
   delta1 = delta1*(float)in[1];

   delta2 = des_out - out;
   delta2 = c*delta2;
   delta2 = delta2*bias;

   weights[0] += delta0;
   weights[1] += delta1;
   weights[2] += delta2;
}
```

This function is where the learning happens. For each weight, we compute a delta (change value) by subtracting the actual output from the desired output, multiplying that by a learning constant, and multiplying that by the value input to that weight.

That is it! All the code you need to implement a simple perceptron on a PIC. After programming, you should see the following behavior from your circuit:

1) The LEDS perform their startup blink and the "hi!" message prints out the serial port.

2) The learning data is printed to the serial port as it occurs.

3) If you choose to program in the dynamic photocell mode, dynamic data streams to the serial port and the output LED shows the logical AND result of the two photocells.

Modify my code! The fun is in the experimenting. Next time, we will tackle a more complex example using a multi-layered perceptron network called a feed-forward neural network and back-propagation learning. **SV**

ANOTHER TIP!

Just a helpful suggestion — if something doesn't make intuitive sense in the code, add a printf statement and watch how the values change. This should clarify it.

Robotics Resources

Tune in each month for a heads-up on where to get all of your "robotics resources" for the best prices!

By Gordon McComb

How to Pick the Right Motor for Your Robot

Motors are like muscles — connect a motor with wheels to a battery and your robot scoots across the floor. Or attach a motor to a lever, and you've made a shoulder joint. Or attach a motor to a turntable, and your robot's head can scan back and forth. Motors are the primary means of articulating a robot. In this month's Robotics Resources, we'll examine some of the more common types of motors and how they are used.

Motors for AC or DC

If you're a scrounger, you probably have a couple of motors laying around you'd like to use in your robot. But some of these motors may be wholly unsuitable. Most 'bots are powered by batteries, which means direct current (unless you use a power inverter, but we won't get into that as it's not as efficient). Direct current (DC) from batteries and power supplies dominates robotics, so these are the kinds of motors you need to look for. Shelve the alternating current (AC) motor you pulled from that old fan for another project.

DC motors may be the motors of choice. But that doesn't mean just any DC motor should or can be used in your robot designs. When looking for suitable motors, be sure the ones you use are reversible. Few robotic applications call for just unidirectional (one direction) motors. You must be able to operate the motor in one direction, stop it, and change its direction. Most DC motors are bi-directional, but some design limitations may prevent reversibility.

How can you tell if a motor is reversible? If it doesn't say, the best and easiest test is to try the motor with a suitable battery or DC power supply. Apply the power leads from the motor to the terminals of the battery or supply. Note the direction of rotation of the motor shaft. Now, reverse the power leads from the motor. The motor shaft should now go the other way.

Continuous Rotation, Stepper, and Brushless Motors

Among the variety of DC motors is a dizzying array of sub-types. Motors that run when you simply apply a DC voltage are continuous rotation and have the most value to robot builders. Internally, most motors of this ilk have a permanent magnet in the casing, and use a metal core that is connected to the motor's shaft. Around this core are several sets of wires wound in a tight loop. These lead to two brushes — they can also be simple bares wires — that connect to the power source. The motor rotates when the wires (also called coils or windings) are energized. Continuous rotation is made possible due to the geometry of the windings around the core.

Variations on this theme may place the magnet in the core and the windings around the casing. There are many other designs, and the Internet — as well as any good book on motor design — is full of pictorial discussions of how DC motors operate. There's no need for me to cover the other approaches here.

Stepper motors are similar to standard DC motors except they are constructed in such a way that they turn only a degree or so when DC power is applied to a winding. To make the motor turn more, you need to manually energize the windings in a specific sequence. To enable this, stepper motors have more than two wires; most steppers have between four and eight wires. The number of wires indicates the type of stepper (bipolar or unipolar) and the way the windings are connected internally within the motor.

At one time this was done using mechanical relays, but these days it all happens using electronics. You can build a stepper motor controller circuit out of discrete logic parts, buy a specialized stepper motor chip such as the L297, or program a microcontroller to apply the right pulses to the windings.

Stepper motors are handy when you want to control the angle of movement and amount of rotation. You can use stepper motors (like those pulled from old disk drives) in lieu of a standard DC motor, but a more common use is for things like rotating sensor turrets, grippers, or arm mechanisms.

Yet another type of DC motor is the brushless motor. These most often use optics, hall effect sensors, or other technology to switch power to the windings inside the motor. Brushless motors are common when speed control is required, such as the spindle motor of a CD player. They require a controller circuit that senses the precise position of the core at any given time so that the correct windings may be energized. For the most part, brushless DC motors aren't as useful in amateur robotics, due in part to their complexity, and also they don't tend to be as powerful as their continuous rotation and stepper cousins.

R/C Servo Motors

When a motor is connected to a feedback circuit, it's called a servo motor. The feedback tells the motor how far it's gone and in what direction. This feedback can be accomplished using a simple potentiometer, a tachometer, an optical encoder, or some other means.

The most common — and inexpensive — servo motor is the R/C servo; R/C stands for radio control, which is the original application of these motors. They are used in model and hobby radio-controlled cars and planes.

R/C servos are in plentiful supply, and cost is reasonable (about $10-$12 for basic units). Though R/C servos are continuous DC motors at heart, they aren't controlled in the same way. In addition to power leads, you need to provide a control signal to an R/C servo. This signal — varying from one to two milliseconds — controls the direction and angle of the motor. Most robots based on R/C servos use a microcontroller to provide this signal. Most any microcontroller is capable of creating the signal and the programming code is simple. Use of R/C servos in robots is so common now that the programming language for some microcontrollers incorporate commands specific to operating the motor. This is the case of the OOPic controller, for example.

There are other forms of servo motors and these are different from the R/C type. If you're browsing surplus catalogs, you may encounter some large and unreasonably expensive servo motors. They're a distinct breed from the less expensive R/C type, and are intended for process control, such as factory automation. The typical industrial servo motor is not a complete unit — you cannot simply wire it to a battery and make it work. At a minimum, it needs a control circuit, if not additional components.

While the R/C servo is intended to precisely control angular movement, it's also possible to retrofit them for continuous rotation. In fact, this is one of the most popular methods of motorizing a robot. The modification involves partial disassembly of the motor so that the mechanical stops — the things that prevent the motor from turning a full 360 degrees — are removed. In addition, the feedback potentiometer is disengaged and set to its center. In this way, the control signal operates the direction and to some degree the speed of the motor.

Motor Specifications

Motors carry with them numerous specifications. The meaning and purpose of some of the specifications are obvious; others aren't. Here is a broad and brief overview of some of the more common specifications:

• **Operating Voltage**. All motors are rated by their operating voltage. With small DC hobby motors, the rating is actually a range, usually 1.5 to 12 volts. DC motors for automation usually require 24 to 48 volts. The kinds of motors of most interest to robot builders are the low-voltage variety — those that operate at 1.5 to 12 volts. Most motors can be operated satisfactorily at voltages somewhat higher or lower than that specified. For continuous DC motors, changing the voltage alters the rotation speed.

• **Current Draw**. This specifies the current, in milliamps or amps, that the motor pulls from the power supply at a given voltage and a given load (motors are designed to perform some work; this is their "load"). Be sure to compare apples to apples. The current draw of a free-running (no load) motor can be quite low, but under load the motor may draw five, 10, even 50 times the current.

• **Speed**. The rotational speed of a motor is usually given in revolutions per minute (RPM), with most continuous DC motors rotating at between 4,000 to 12,000 RPM. Stepper motors are rated by their maximum steps per second which, when combined with their step angle, can be used to determine revolutions per second or minute. R/C servo motors are rated by their transit time, usually in fractions of a second to cover a 60 degree arc. Except for stepper motors, speed is influenced by voltage.

• **Torque**. The torque of a motor is the force it exerts upon its load. The higher the torque, the larger the load can be, and the faster the motor will spin under that load. Reduce the torque and the motor slows down, straining under the workload. Torque specifications vary wildly; some are given in old Imperial units of ounce-inches or foot-pounds, while others are force newton-meters or some other standards. When comparing torque specifications that don't use the same units of measure, you can use an online conversion page, such as at **www.convert-me.com**.

Motors With and Without Gears

R/C servo motors already incorporate their own internal gearing. The gears reduce the speed of the motor used inside the servo from its usual 7,000 to 10,000 revolutions per minute to roughly 60 RPM. In reducing the speed, the torque of the motor is proportionally increased.

For most robots, the raw speed of a DC motor is far too high, so you usually want some kind of gear reduction. You can create your own gearbox using gears from various sources (bought new or surplus, or pulled out of toys and discarded appliances). But tolerances are tight with gears. Too loose and the gears don't mesh properly; too tight and the gears bind up, adding a lot of friction.

While you can sometimes add a gearbox to an existing motor, it's usually easier to simply get a motor with a gearbox already attached. You can find plenty of gear motors on the surplus market of various sizes and styles. Another good source is the Tamiya gear motor kits, available through several online resources such as Pololu (**www.pololu.com**) or Tower Hobbies (**www.towerhobbies.com**). You build the motor from parts, and most let you select the gear ratio you want.

Adapting a Motor for Your Robot

Simply having a motor and a robot doesn't mean the two will be happy together. Some types of motors defy being attached to a robot — there are no mounting flanges or screw holes anywhere. The more adaptable motors have at least two holes for securing the thing to a robot.

Sometimes these holes are in the face of the motor (the part where the shaft sticks out). These require screws of a particular length. Some motors use metric threads (example: M3) and others use Imperial threads (example: 4-40). Be sure to use the right type of screws or you'll mess up the threads in the motor.

Yet other motors have flanges that make mounting easier. This is the case with many stepper motors, which often have a flanged mounting plate with two or four holes. It's also the case with R/C servos, which are flanged and have two or four eyelets for screws. With some ingenuity and commercially sold brackets (such as those from my small online company, Budget Robotics at **www.budgetrobotics.com**), you can fit an R/C servo motor just about anywhere. As a last resort, you can always use double-sided foam tape, Velcro, or 3M Dual Lock to secure the motor to your robot.

Next comes attaching a wheel, lever, or other mechanism to the motor shaft, and this is where things often get tricky. I like using R/C motors specifically because they make it very easy to attach just about anything, including wheels. R/C motors come with an assortment of "horns" — or extras are available for it — in a variety of shapes and sizes. You can drill into the horn so you can adapt it to your particular needs. The horn then securely screws into the output shaft of the motor.

For continuous DC and stepper motors, you need a way to mount something to the shaft. Motor shafts comes in all sizes, lengths, and dimensions. Most are made of a hardened metal, so drilling into them is a tough proposition. The most effective designs use setscrews. An example is a setscrew that is part of the hub of a gear. You tighten the setscrew to secure the gear to the shaft.

The diameter of the shaft matters. The hole that fits around it — be it the bore of a wheel or the hub of a gear or sprocket — should be about the same size. If the hole is a little too small, you can usually make things work by drilling it out a little.

Sources

The Internet makes it very convenient to find motors for your robot. You can choose from a variety of motors from different types of vendors. Some vendors specialize in used and surplus motors. Rely on them for larger robots that requires hefty motors with extra torque.

No list of resources for motors can hope to mention everyone. Here is a selection to get you started. You'll want to use your favorite online search engine to help you find more.

All Electronics
www.allelectronics.com

Sells surplus motors, including stepper, DC, and gear motors. Also offers a printed catalog mailed to current customers. Retail stores in the Los Angeles, CA area.

American Science & Surplus
www.sciplus.com

Often carries surplus stepper motors. Printed and online catalog available.

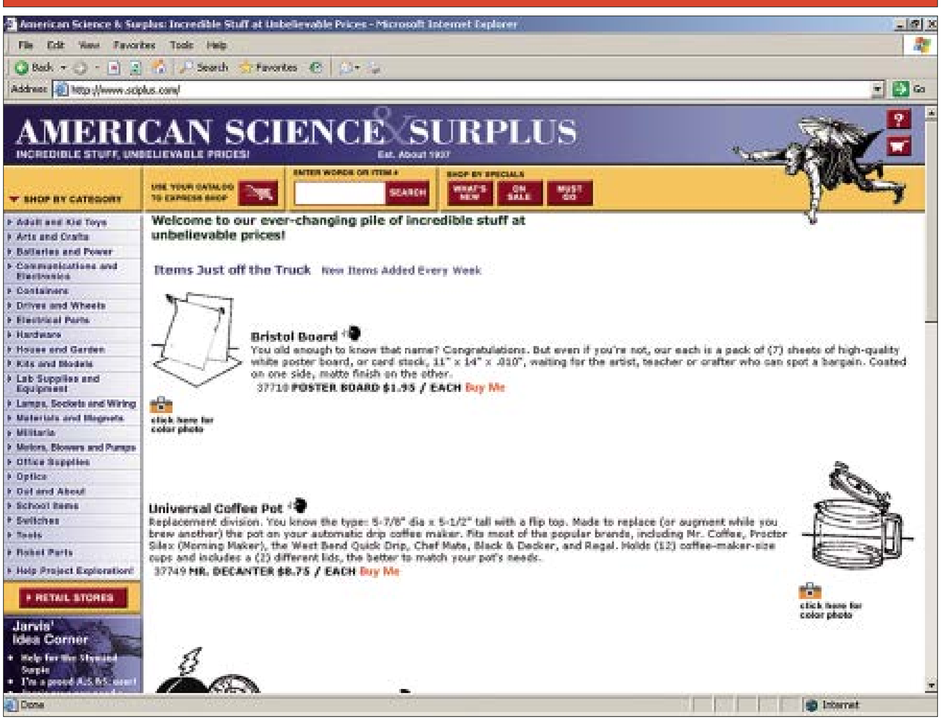
B.G. Micro
www.bgmicro.com

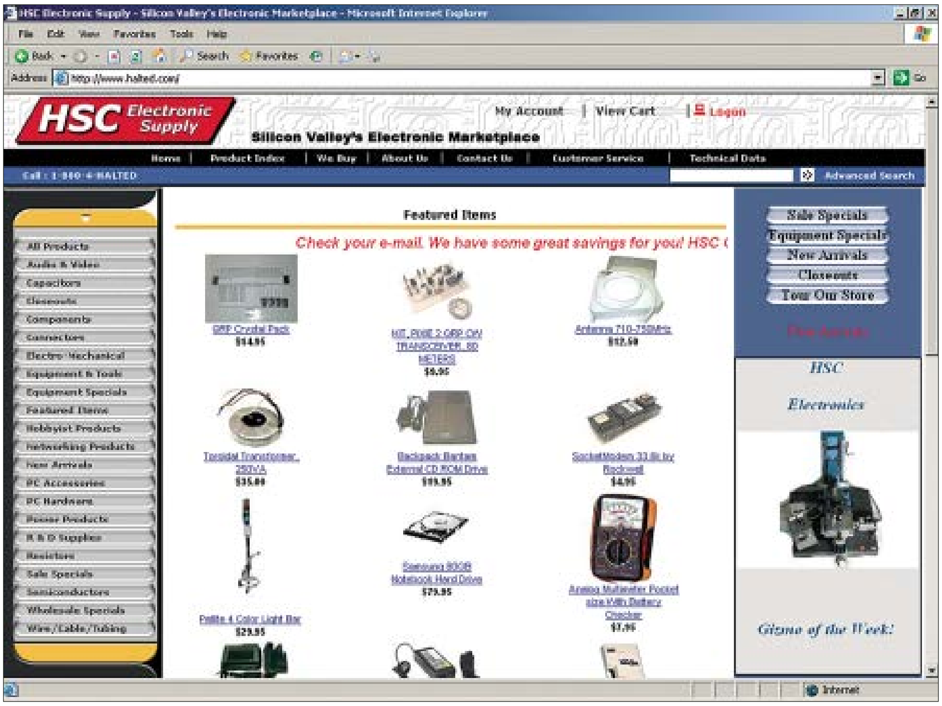
Source for some surplus motors. Check their site often so you are sure to catch the best deals.

C & H Sales
www.candhsales.com

C & H sells motors of all types, gears, pneumatics, pumps, solenoids, relays, and lots of odds and ends. Their catalog (both printed and online) regularly contain dozens of quality surplus DC (geared and non) and stepper motors. If you're in the Los

American Science and Surplus offers surplus stepper motors.





HSC offers surplus mechanics, such as motors, relays, gears, optics, and much more!

Angeles, CA area, be sure to take a trip to the C & H retail store.

Electronic Goldmine
www.goldmine-elec.com

Electronic Goldmine sells new and used electronic components, robot items, motors, and more.

Electronix Express
www.elexp.com

Electronics parts, supplies, components, hardware, switches, relays, test gear, tools. New and surplus; large inventory.

Fair Radio Sales
www.fairradio.com

While Fair Radio primarily caters to ham operators, they also often sell surplus motors and other mechanical parts.

Gateway Electronics, Inc.
www.gatewayelex.com

Gateway is a general electronics mail order and retailer. Among their product is passive and active components, motors, electronic kits, gadgets, books, and tools. Some of their goods are new; others are surplus. They operate local stores in St. Louis, MO, San Diego, CA, and Denver, CO.

H&R Company, Inc. (Herbach and Rademan)
www.herbach.com

Surplus mechanicals: motors, relays, gears, optics, and lots, lots more.

HSC Electronic Supply
www.halted.com

Mail order, with walk-in retail stores in Northern California.

Jameco Electronics/Robot Store
www.jameco.com *or* **www.robotstore.com**

Jameco carries just about everything you need, including surplus and new motors. They're a good source for small gear motors. Also check out their sister website, The Robot Store.

Marlin P. Jones & Assoc., Inc.
www.mpja.com

MPJA sells both new and surplus electronic and mechanical products. Their assortment of motors comes and goes, but I like to keep them on my "hunt list" as they sometimes have some great deals.

MECI
www.meci.com

New and surplus components, motors, and more.

Surplus Center
www.surpluscenter.com

The Surplus Center offers a wide variety of motors for the robotics tinkerer.

A haven for the robotics tinkerer, with a large variety of all kinds of motors. You may search by type. Their online system shows the remaining quantity of any item, helping you plan your projects.

R/C Servo Sources

There are lots and lots of places to buy R/C servo motors. These should get you started.

Balsa Products
www.balsapr.com
Reseller of several servo brands.

Budget Robotics
www.budgetrobotics.com
Several variations of specialty servo mounts and servo wheels for robotics, and a small assortment of servos. (Note: This is the company I operate.)

Lynxmotion
www.lynxmotion.com
Hitec servos, custom-made mounts, and wheels.

Parallax
www.parallax.com
In addition to manufacturing the popular BASIC Stamp microcontroller, Parallax offers a number of robotics components, including branded servos.

Pololu
www.pololu.com
Pololu carries servos, wheels, and treads and sprockets for building tracked robots.

Servo City
www.servocity.com
Reseller of Hitec and Futaba servos, plus manufacturer of custom servo horns, mounts, external gearboxes, and other accessories.

Tower Hobbies
www.towerhobbies.com
Slightly higher prices than the others, but good selection and customer service.

R/C Servo Motor Manufacturers

Check here for datasheets and specifications of popular servo brands. Some companies do not sell directly to the public.

Airtronics: **www.airtronics.net**

Futaba: **www.futaba-rc.com**

GWS: **www.gws.com.tw**

Hitec USA: **www.hitecrcd.com**

Robotis: **www.robotis.com**
Specialized form of small servo motor, specifically intended for small robots. The motor is controlled by serial communications from a computer or microcontroller. Check the dealer page. **SV**

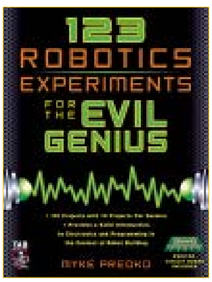
The *SERVO* Store

SERVO CD-ROM

Are you ready for some good news? Along with the first 26 issues of *SERVO Magazine*, all issues from the 2006 calendar year are now available, as well. These CDs include all of Volume 1, issues 11-12, Volume 2, issues 1-12, Volume 3, issues 1-12, and Volume 4, issues 1-12 for a total of 38 issues all together. These CD-ROMs are PC and Mac compatible. They require Adobe Acrobat Reader version 6 or above. Adobe Acrobat Reader version 7 is included on the discs. **$24.95 – Buy 2 or more at $19.95 each!**

123 Robotics Projects for the Evil Genius

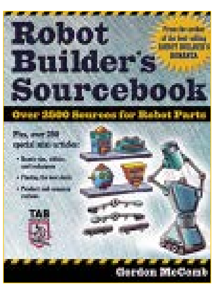
by Myke Predko



If you enjoy tinkering in your workshop and have a fascination for robotics, you'll have hours of fun working through the 123 experiments found in this innovative project book. More than just an enjoyable way to spend time, these exciting experiments also provide a solid grounding in robotics, electronics, and programming. Each experiment builds on the skills acquired in those before it so you develop a hands-on, nuts-and-bolts understanding of robotics — from the ground up. **$25.00**

Robot Builder's Sourcebook

by Gordon McComb



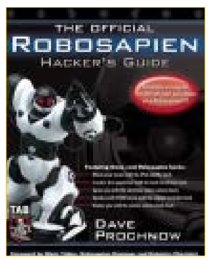
Fascinated by the world of robotics but don't know how to tap into the incredible amount of information available on the subject? Clueless as to locating specific information on robotics? Want the names, addresses, phone numbers, and websites of companies that can supply the exact part, plan, kit, building material, programming language, operating system, computer system, or publication you've been searching for? Turn to *Robot Builder's Sourcebook* — a unique clearing-house of information that will open 2,500+ new doors and spark almost as many new ideas. **$24.95**

SERVO CD-Rom

Are you ready for some good news? Along with the first 14 issues of *SERVO Magazine*, all issues from the 2005 calendar year are now available, as well. These CDs include all of Volume 1, issues 11-12, Volume 2, issues 1-12, and Volume 3, issues 1-12, for a total of 26 issues all together. These CD-ROMs are PC and Mac compatible. They require Adobe Acrobat Reader version 6 or above. Adobe Acrobat Reader version 7 is included on the discs. **$24.95 – Buy 2 or more at $19.95 each!**

The Official Robosapien Hacker's Guide

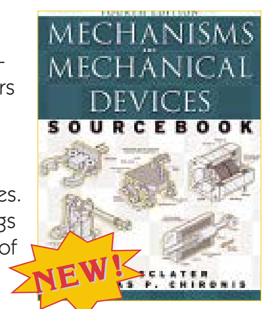
by Dave Prochnow



The Robosapien robot was one of the most popular hobbyist gifts of the 2004 holiday season, selling approximately 1.5 million units at major retail outlets. The brief manual accompanying the robot covered only basic movements and maneuvers — the robot's real power and potential remain undiscovered by most owners — until now! This timely book covers all the possible design additions, programming possibilities, and "hacks" not found anyplace else. **$24.95**

Mechanisms and Mechanical Devices Sourcebook

by Neil Sclater / Nicholas Chironis



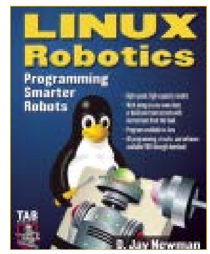
The fourth edition of this invention-inspiring engineering resource covers the past, present, and future of mechanisms and mechanical devices. You'll find drawings and descriptions of more than 2,000 components that have proven themselves over time and can be incorporated into the very latest mechanical, electromechanical, and mechatronic products and systems. Overviews of robotics, rapid prototyping, MEMS, and nanotechnology, along with tutorial chapters on the basics of mechanisms and motion control, will bring you up-to-speed quickly on these cutting-edge topics. **$89.95**

SERVO CD-Rom

Are you ready for some good news? Starting with the first *SERVO Magazine* issue — November 2003 — all of the issues through the 2004 calendar year are now available on a CD that can be searched, printed, and easily stored. This CD includes all of Volume 1, issues 11-12 and Volume 2, issues 1-12, for a total of 14 issues. The CD-Rom is PC and Mac compatible. It requires Adobe Acrobat Reader version 6 or above. Adobe Acrobat Reader version 7 is included on the disc. **$24.95 – Buy 2 or more at $19.95 each!**

Linux Robotics

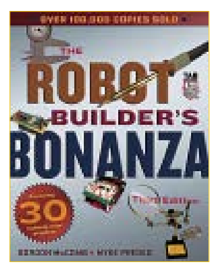
by D. Jay Newman



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, plus complete training in doing original programming. **$34.95**

Robot Builder's Bonanza Third Edition

by Gordon McComb/Myke Predko



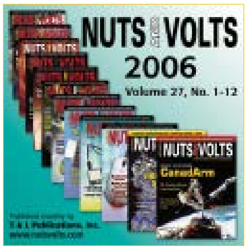
Everybody's favorite amateur robotics book is bolder and better than ever — and now features the field's "grand master" Myke Predko as the new author! Author duo McComb and Predko bring their expertise to this fully-illustrated robotics "bible" to enhance the already incomparable content on how to build — and have a universe of fun — with robots. **$27.95**

**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 or go to our website at www.servomagazine.com

Mind Candy For Today's Roboticist

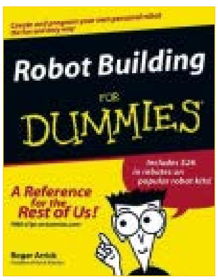
Nuts & Volts CD-ROM



Here's some good news for *Nuts & Volts* readers! Along with all 24 issues of *Nuts & Volts* from the 2004 and 2005 calendar years, the 2006 issues are now available, as well. These CDs include all of Volumes 25, 26, and 27, issues 1-12, for a total of 36 issues (12 on each CD). These CD-ROMs are PC and Mac compatible. They require Adobe Acrobat Reader version 6 or above. Adobe Acrobat Reader version 7 is included on the discs. **$24.95 – Buy 2 or more at $19.95 each!**

Robot Building for Dummies

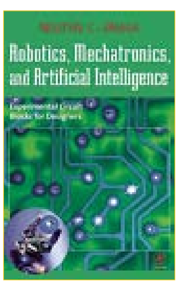
by Roger Arrick / Nancy Stevenson



Ready to enter the robot world? This book is your passport! It walks you through building your very own little metal assistant from a kit, dressing it up, giving it a brain, programming it to do things, even making it talk. Along the way, you'll gather some tidbits about robot history, enthusiasts' groups, and more. Do it the Dummies' way — explanations in plain English, "get in, get out" information, icons and other navigational aids, tear-out cheat sheet, top 10 lists, and more. **$21.00**

Robotics, Mechatronics, and Artificial Intelligence

by Newton C. Braga



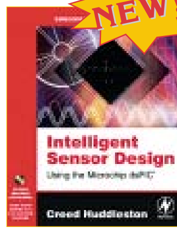
Accessible to all readers, *Robotics, Mechatronics, and Artificial Intelligence* simplifies the process of finding basic circuits to perform simple tasks. From how to control a DC or step motor to instruction on creating moving robotic parts, this book covers them all. With this new book by Newton Braga, hobbyists and experimenters around the world will be able to decide what skills they want to feature in a project and then choose the right "building blocks" to create the ideal results. **$33.95**

Check out our online bookstore at www.servomagazine.com for a complete listing of all the books that are available.

Intelligent Sensor Design Using the Microchip dsPIC

by Creed Huddleston

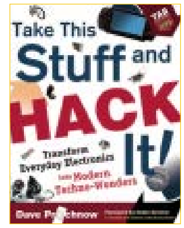
NEW!



Unlike many embedded systems books that confine themselves strictly to firmware and software, this book also delves into the supporting electronic hardware, providing the reader with a complete understanding of the issues involved when interfacing to specific types of sensors and offering insight into the real-world problems designers will face. Meaningful software examples are implemented in both C and assembly language, and the source code is included on the accompanying CD. **$59.95**

Take This Stuff and Hack It!

by Dave Prochnow

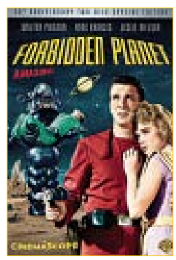


Transform common household items into really cool stuff. You don't need to be an electronics genius to get started turning everyday items into high-performing wonders. With how-to guru Dave Prochnow's step-by-step directions and fully illustrated plans, even beginners can hack their way to a high-tech home, cooler toys, and less yard work. Certain to fire your imagination and start you plotting new, original, and even more creative wonders you can make from ordinary household items, *Take This Stuff and Hack It!* is the perfect gift for your inner inventor. **$27.95**

NEW! The SERVO Store now offers DVDs!

Forbidden Planet: 50th Anniversary Edition

Director: Fred M. Wilcox
Run Time: 98 minutes

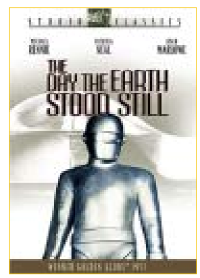


A starship crew goes to investigate the silence of a planet's colony only to find two survivors and a deadly secret that one of them has. When Adams and his crew are sent to investigate the silence from a planet inhabited by scientists, he finds all but two have died. Dr. Morbius and his daughter Altaira have somehow survived a hideous monster which roams the planet. Unknown to Adams, Morbius has made a discovery.

Actors: Walter Pidgeon, Anne Francis, Leslie Nielsen, Warren Stevens, Jack Kelly. **$26.95**

The Day the Earth Stood Still

Director: Robert Wise
Run Time: 92 minutes



An alien (Klaatu) with his mighty robot (Gort) land their spacecraft on Cold War-era Earth just after the end of World War II. They bring an important message to the planet that Klaatu wishes to tell to representatives of all nations. However, communication turns out to be difficult, so, after learning something about the natives, Klaatu decides on an alternative approach.

Actors: Michael Rennie, Patricia Neal, Hugh Marlowe, Sam Jaffe, Billy Gray. **$14.95**

SUMMER SPECIAL $14.59!

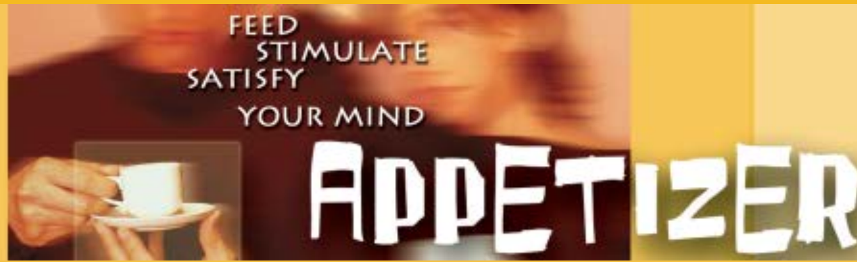
From HomoSapien to RoboSapien

SERVO Magazine T-Shirts

For men and women

Before R2D2 there was R1D1

Get your very own limited-edition *SERVO Magazine* T-shirt. Shirts come in sizes S, M, L, and XL, and are available in either black or white. All shirts are 100% preshrunk cotton.



Artificial Intelligence and the State of Robotics Today

by Tom Carroll

Every so often, one of the many robotics group's server lists that I subscribe to has a very interesting series of comments from members and interested outsiders. Such was the case with the Dallas Personal Robotics Group's discussion the last part of May on artificial intelligence. Boy, did it get interesting as different responders talked about state machines, ultimate robot capabilities, Isomax software, various processors and microcontrollers, what is artificial intelligence (AI), and so many sub topics that are too numerous to mention. AI seems to be the one subject (after deciding on just what is a robot) that sets people off on their own separate directions.

I am always amazed at just how informed so-called robot amateurs and hobbyists are when it comes to complex subjects such as AI. Many of today's robot experimenters who have ventured past the early stage of simple line-following robots have delved into AI to an extent that would amaze the university AI grad students of the 80's.

I remember back in the early '80s when I first got involved with robotics professionally and people were discussing the moniker artificial intelligence. Some felt that the term synthetic intelligence was more appropriate for a machine's computational capabilities as AI was based on a human's way of solving a problem. But who really cares? We know what AI means in our own minds.

Most people who delve into the intricacies of AI always seem to go back to Alan Turing's paper: "Computing Machinery and Intelligence," published back in 1950 while he was employed at the Computing Laboratory at Manchester University, UK. It has become one of the most cited in philosophical literature. He had worked quite a bit with the university's computer that used stored programs. With the war effort now over, Turing told the *London Times* in 1949 that the main interest of the new computing machine lay in discovering to what extent it could show intelligence. Turing held that computers — in time — could be programmed to acquire abilities that rivaled human intelligence.

Turing is most famous for his test of machine intelligence. Turing's 'imitation game,' as he originally called it, put forth the idea of a human being and a computer being interrogated by a separate person or 'judge' under conditions where the judge would not know which was which. The communication was entered into the computer entirely by text messages. Turing argued that "if the interrogator could not distinguish them by questioning, then it would be reasonable to call the computer intelligent, because we judge other people's thoughts externally in this way." Remember now, that this was way back in 1950 when a 'computer' was a room full of tubes and whirring tape drives used for military purposes just a few years prior. The Turing Test remains one of the premier tests used in AI today and no computer has yet to pass as a human as evaluated by a talented judge.

Joseph Weizenbaum developed one of the more entertaining attempts at AI in 1966. Originally meant to be a hoax, the computer program ELIZA became an instant hit with anybody with access to any sort of computer with an input/output device such as a teletype machine. It reached its peak of popularity in the late '70s to early '80s as microprocessor-based personal computers became popular. I know many an hour was wasted at Rockwell as other engineers and I inputted questions to Weizenbaum's program that was formatted as a psychologist. I later got a version that would run on my Apple II and kept my son and me happy at home.

It sounded so much better when the output was fed through a speech synthesizer. It sometimes was a bit eerie. The program went somewhat like this as a 'person' — a young girl in this case — would enter a question and ELIZA would output an answer:

Person: Men are all alike.
ELIZA: In what way?
Person: They're always bugging us about something or other.
ELIZA: Can you think of a specific example?
Person: Well, my boyfriend made me come here.
ELIZA: Your boyfriend made you come here?
Person: He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
Person: It's true. I am unhappy.
ELIZA: Do you think coming here will help you not to be unhappy?
Person: I need some help, that much seems certain.
ELIZA: What would it mean to you if you got some help?

Person: Perhaps I could learn to get along with my mother.
ELIZA: Tell me more about your family
Person: My mother takes care of me.
ELIZA: Who else in your family takes care of you?
Person: My father.
ELIZA: Your father?
Person: You are like my father in some ways.
ELIZA: What resemblance do you see?
Person: You are not very aggressive but I think you don't want me to notice that.

And so on until you get tired or run out of questions ...

Does this in any way tell you just what artificial intelligence is? Absolutely not! Is it insightful as to the state of the art in present day AI research? Not really. AI is making some amazing strides in developing smart machines to help us in war, in day-to-day life, medicine, security, transportation technology, manufacturing, and many other facets of today's world.

The DARPA Grand Challenge made some amazing progress in just a few years. The first autonomous vehicles to traverse a desert course failed at the very beginning of the run. The next contest was completed in style with many vehicles finishing the over 100 miles of rough terrain and Stanford and Red Whittaker's teams enjoying a close 1, 2 finish. This was AI at its best — great computing power fed by excellent suites of sensors and applied to ready-to-roll off-road vehicles.

These were some great 'robots' designed by some of the best robot experts in the world. These teams will next set their machines off in an urban environment — the next level for an autonomous vehicle performance demonstration.

An article by Bill Softky in the May 25th issue of the Internet newsletter, *The Register*, really bothered me. Entitled 'Why do Robot Experts Build such Lousy Robots?,' Bill went on to elaborate how "the main appeal of robots, their autonomy, and intelligence, are decades away from reality even in the lab." Softky is a scientist at the Redwood Neuroscience Institute in Menlo Park, CA and should be aware of the state-of-the-art in robotics. He appears to be totally unaware of the many small but mighty steps that are being made in labs around the world to further the science of robotics.

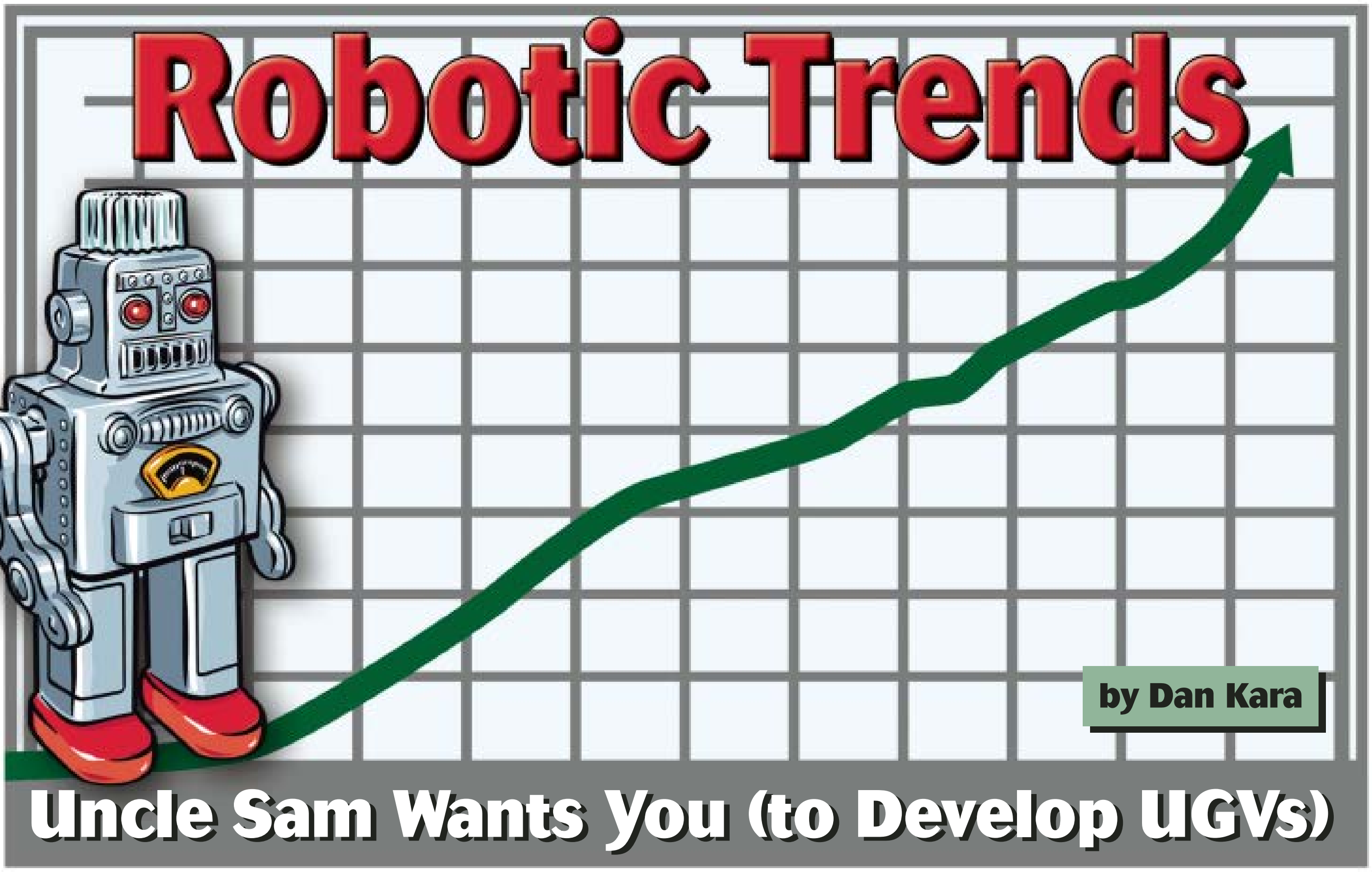
The Register, published in London with several US offices, appears to be more of a sensationalist news source than offering a clear view of reality. They use an image of a buzzard as their logo; does this imply the content is the carrion of the scientific world? Softky will have a second installment of my article after this article goes to press.

We may not have humanoid robots serving us in our homes that was predicted as 'the future' 50 years ago, but $200 Roombas are keeping the homes of over two million people fairly clean. We may not have autonomous cars zipping down our highways but Stanford's autonomous VW Touareg did a fine job of sucking up a $2 million DARPA prize after bouncing through 132 miles of inhospitable desert. Softky speaks of 'tinkerers' as today's robot experimenters but has probably never been into the workshop of an advanced robotics hobbyist.

The US may no longer be the world's main source of industrial robots, but some great non-industrial robots are being devised in labs across the country. Tandy Trower and his team at Microsoft are looking far past their successful Microsoft Robotics Studio software package into applications for future robot systems. Colin Angle and the team of iRobot are not relying on the success of the Roomba but see military and other applications as the future.

Lousy robots, he says? I'd love to invite Bill Softky into the labs of Bart Everett, Marvin Minsky, Tandy Trower, Rodney Brooks, Red Whittaker, and so many more robot designers and builders to see robots that would amaze anyone. People say that the robotics field needs the next killer app, much like VisiCalc of the '70s. Hey, we've passed that point just as the computer industry did when Microsoft developed Excel, Word, and keeps on bringing new products to the market.

Maybe we turned a corner from industrial applications to non-industrial robots when astounding robot toy dogs, autonomous vacuum cleaners, and Microsoft Robotics Studio made the scene, but no robotics person is resting on his or her laurels. There is much work to be done and amazing things will keep emerging from robotics labs across the land. As the late Walt Musser, inventor of the harmonic drive and many other robot-related devices, stated: "It is never a question as to whether it can be done — it is only whether one cares to spend the time and effort." SV



Robotic Trends

by Dan Kara

Uncle Sam Wants You (to Develop UGVs)

The Army has big plans for battlefield unmanned ground vehicles and they need your help to realize their 'Future Force' vision.

In previous *SERVO* columns, we have touched upon a number of issues related to careers in the robotics space, with particular attention applied to high growth markets. The educational robotics market serves as an example. Other 'hot' robotics sectors include medical and consumer robotics. One of the fastest growing robotics sectors is military robotics, the subject of this column. Given the focus of *SERVO Magazine* (largely ground based robots) and the length constraints of the column, let's limit the discussion of military robots to unmanned ground vehicles (UGVs). Let me apologize beforehand for the number of acronyms, but to discuss military vehicles, we must use military speak.

The US military — like the militaries of most developed countries — understands the value of applying robotics technology to military requirements and are willing to spend time and money to deliver operational systems. Unmanned aerial vehicles, which are now proving their worth over the skies of Afghanistan and Iraq, were under development for more than 40 years and cost plenty in R&D dollars. Unmanned ground vehicles, too, are receiving much interest from the military (read funding), and this interest translates into career opportunities.

At this time, there are approximately 4,000 ground robots employed by the US military (13 systems), up from less than 100 in 2001. These robotics systems are in action in multiple theaters including the European Command (EUCOM), Iraq, Afghanistan, and Central Command (CENTCOM). Improvised explosive device (IED) detection, reconnaissance, and explosive ordnance disposal (EOD) are typical applications.

The Army's key defense acquisition program to develop and field light, medium, and heavy unmanned ground vehicles is the Future Combat Systems (FCS) initiative, a $161B modernization program (the Pentagon's second most costly program, behind the $276B Joint Strike Fighter). UGV development efforts have also been supported by the Army's Joint Ground Robotics Enterprise (JGRE) and the Defense Advanced Research Projects Agency's (DARPA) Unmanned Ground Combat Vehicle (UGCV) Perception for Off-Road Robots Integration (UPI) program. The UGVs include fielded systems and prototypes, as well as commercial off-the-shelf (COTS) purchases. Classes of unmanned ground vehicles under these programs include:

- *Armed Reconnaissance Vehicles (ARV)* — These systems provide reconnaissance, surveillance, and target acquisition (RSTA), as well as force protection services (the armed version).

- *Unmanned Ground Combat Vehicle* — Large, weaponized UGVs designed

> **The US military — like the militaries of most developed countries — understands the value of applying robotics technology to military requirements and are willing to spend time and money to deliver operational systems."**

for ground combat.

• *Soldier Unmanned Ground Vehicles (SUGV)* — These are small, man-packable robot systems that can be configured for a large number of missions including reconnaissance and explosive detection and disposal.

• *Multifunction Utility/Logistics Equipment Vehicles (MULE)* — These systems support the soldier in the field, providing transportation services for equipment and supplies.

More Than the War

The war in Iraq drives much of the military's current robotic activity. However, ever since 1990, Department of Defense (DoD) ground robotics investments have steadily increased over the years and will continue to do so. For example, since 1990, the level of annual JGRE appropriations has increased from approximately $20M to almost $50M in FY2006. Moreover, it is projected that DoD research investments during FY2006-2012 will approach $1.7B. More importantly for those evaluating military robotics as a career, this research will transition to acquisition programs.

More Than Casualty Reduction

There are many reasons for the military's interest in unmanned ground vehicle systems. This is a good thing for those considering a career in military robotics as it reduces risks. And risks there are with any career choice. Although there have been efforts at scaling back the Army's Future Combat Systems initiative, the benefits of the military application of robots and robotic technology are so compelling that projects will continue even if substantial program funding cuts are made.

Casualty reduction is one of the primary reasons for the military's battlefield robotics initiatives. Robots can be used in place of humans for many of the dangerous and life threatening tasks that soldiers perform on a daily basis. Also, robots are able to accomplish some tasks better than their human counterparts, or undertake tasks that humans simply cannot perform. That is, robotic systems can increase battlefield operational effectiveness, in addition to lives.

In fact, Army officials have requested that research in UGV functionality should focus on what robots can do better that humans, as opposed to what humans already do well. For their part, the Army has developed an ordered list of areas where they need support. Robotics technology can be applied to meet many of these requirements (see Figure 1).

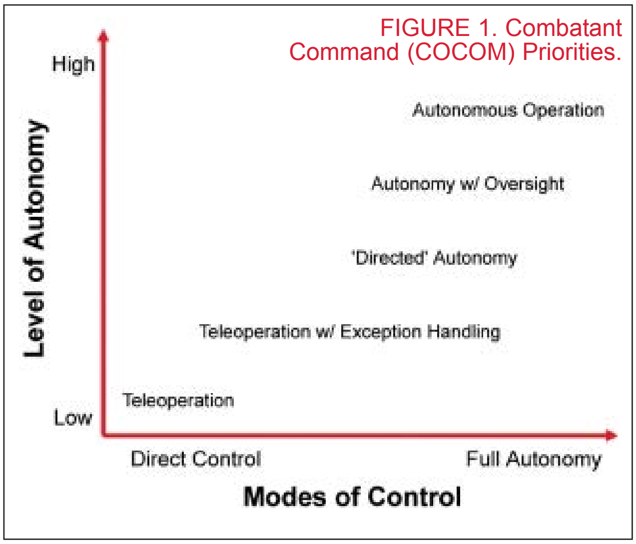
Robotics in the form of unmanned ground vehicles can also be used to completely change the way in which military force is applied. The Army's current transformation from a slow moving, heavily armored (and heavily armed) force to a highly flexible, responsive, and agile entity provides an additional driver for the increased use of robotics within the military. As it is now envisioned, the Army's 'Future Force' will rely heavily upon unmanned ground systems to extend perception (reconnaissance, surveillance, and target acquisition) and affect action (counter-mine operations, transport, as well as weapons platforms, extraction of the wounded, etc.) on the battlefield.

Reducing Costs

At this time, cost savings does not appear to be a driving force in military robotics acquisition programs. First, the individual robots themselves are expensive. Second, for the near term, the employment of robots will not result in a reduction of military personnel. In fact, they could increase the number of Army personnel as the robots themselves require operators and maintenance teams. There is also concern in some quarters that the systems themselves will not be able to fulfill mission goals due to limited functionality and a lack of robustness.

As robotic systems increase their perception and reasoning capabilities — and therefore become more autonomous — operator demand will decrease. Similarly, as the systems themselves become more functional and robust (and standardized), the maintenance load will drop and the systems can be applied in an increasing number of areas. It can be seen, therefore, that advances in autonomy,

FIGURE 1. Combatant Command (COCOM) Priorities.



Mission	Company	Brigade Combat Team	Division	Overall Ranking
Reconnaissance	1	1	1	1
Mine Detection	2	2	2	2
Precision Target Location and Designation	3	3	5	3
Chem/Bio Reconnaissance	6	4	3	4
Weaponization/Strike	4	6	6	5
Battle Management	8	5	4	6
Communications/Data Relay	5	7	7	7
Signals Intelligence	7	8	8	8
Covert Sensor Insertion	9	9	10	9
Litteral Warfare	13	10	9	10
Counter Cam/Con/Deception	10	11	11	11
Special Operations Force Team Resupply	11	12	12	12
Combat Search and Rescue	12	13	13	13
Information Warfare	14	14	14	14
Decoy/Pathfinder	15	15	15	15

* Slide courtesy Helen Purdy, Enterprise Director, Joint Ground Robotics, Department of Defense, from presentation delivered at RoboBusiness 2007 Conference and Exposition (May 2007)

FIGURE 2. Levels of Autonomous Operation in Unmanned Vehicles.

functionality, and robustness will act to increase mission capabilities and reduce costs.

Where the Jobs Are

When evaluating career opportunities, a sound approach is to look where there is the greatest need (What are the critical requirements?). Military robotics is similar to all other careers in this respect.

Requirements prioritization is a thorny and uncertain undertaking, and the criticality of unmanned systems make that effort that much more difficult. However, it can be seen that much of the UGV program's success is predicated on the ability of the systems to be able to work autonomously (after all we are talking about 'unmanned' ground vehicles). As we have seen, increases in the level of autonomy will increase the operational effectiveness of the systems, and will eventually reduce overall costs.

Currently, the Army's unmanned systems are teleoperated, but the goal is to have some systems become fully autonomous over time. This will occur in a stepwise fashion beginning with adding semiautonomous mobility capabilities to current systems. Degrees of autonomy can be described as follows (see Figure 2):

• *Teleoperation* — Dedicated, continuous remote operation without exception handling. Human operator makes all decisions.

• *Teleoperation w/Exception Handling* — Continuous remote operation with exception notification when problems occur ('wheels spinning, unable to proceed').

• *'Directed' Autonomy* — Systems directed to 'go there' and 'go there next' with minimal, non-continuous, direct control. Operator guidance provided when problems arise.

• *Autonomy w/Oversight* — Systems 'go there' with no direct, continuous control by human operators making path following decisions by themselves.

• *Autonomous Operation* — Complete autonomy, problem resolution, and correction capability ('go there, perform this task, and return').

High end robotics technologies such as military robotic systems are characterized as having significant technical and production risks. These technical challenges must be addressed before unmanned ground systems can become a fixture in the Army's Future Force. Power management, human-machine interfaces, and integration with other UGV and manned systems are just a few of the UGV functional areas in need of solutions. Technical hurdles in these areas and many more (particularly those related to achieving autonomy), must be addressed to realize the Army's vision for Future Force UGVs.

Conclusion

The United States is the world leader in the area of military robotics, but many other industrialized countries are embracing robots and robotic technology as a means to increase the efficacy of their militaries and to reduce casualties. Robust, practical military robots are now deployed in the field and new technologies and systems are under development (and test). Cost reduction is not a driver at this time, but could be realized as systems become more autonomous, functional, and durable.

If you are considering military robotics as a career or a direction for your company, the news is all good ... roadmaps are in place for the development and deployment of UGV systems and funding is available. Moreover, the technical challenges for developing unmanned ground systems are daunting, and therefore, there will be no shortage of work, especially in the area of autonomous mobility.

For those of you uneasy with the notion of working for the military, take heart. The same work can be applied to first responder systems and autonomous transportation for commercial markets. **SV**

*Dan Kara is President of Robotics Trends, the producer of the RoboBusiness (**www.roboevent.com**). RoboDevelopment (**www.robodevelopment.com**) and RoboNexus (**www.robonexus.com**) conferences, and publisher of Robotics Trends (www.roboticstrends.com), an online news, information, and analysis portal covering the personal, service, and mobile robotics market. He can be reached at dk@roboticstrends.com.*

Then and NOW

ROBOT SENSORS

by Tom Carroll

Sensors are a very critical part of any robot, whether autonomous or teleoperated. Sensors are the robot's contact with the outside world or its own inner workings. There was a time when 'sensors' for an experimenter's robot were just a few whiskers connected to microswitches to sense walls and such. When the robot banged into a wall or obstacle, the switches were tripped and its simple logic steered it in another direction. I mentioned such a crude robot that I built years ago in last month's column. Certainly some early machines used photo tube, selenium, or cadmium sulfide cells to detect light sources and react accordingly, but sensors for the most part were some sort of bumpers or whiskers to detect tangible objects and barriers.

Those were the days before microcontrollers were available that could make sense of a suite of sophisticated sensors that we have available today. A typical robot emerging from the workshops of *SERVO*'s readers might have upwards of 100 sensors to guide it about its daily activities. Sensors can be used to sense the robot's environment or its own internal parameters. These might include active and passive IR sensors; sound and voice sensors; ultrasonic range sensors, positional encoders on arm joints, head and wheels; compasses, navigational and GPS sensors; active and passive light and laser sensors; a number of bumper switches; and sensors to detect acceleration, turning, tilt, odor detection, magnetic fields, ionizing radiation, temperature, tactile, force, torque, RF, UV, video, and numerous other types.

Figures 1 through 4 illustrate just a few of the industrial quality sensor types manufactured by SICK, a company who supplied many of the sensors used in DARPA's Grand Challenge. There are many similar companies producing sensors for use on robots.

IR Sensors

I'll start with IR sensors as they are among the most popular for experimental robots. Infrared sensors are usually divided into two basic types: the passive or PIR sensors that emit no IR radiation and the active types that emit an IR beam that is again detected by reflection. We all have used the PIR types to detect the presence of a human outside our homes and have it turn on an outside light for a specified number of minutes. Since hundreds of millions are sold each year for security purposes, the prices are quite low. The PIR sensor uses a crystal of lithium tantalate ($LiTaO_3$), a compound that possesses unique pyroelectric properties in that it can sense the temperature range of a human being's body. The crystal does not detect the actual temperature but a change in temperature. It does this by seeing a series of images focused upon the crystal by a row of Fresnel lenses as a person (or animal) crosses the viewing path. This crossing movement causes a series of images to cross the crystal's surface, thus, a succession of temperature changes in the crystal's charge status. Open up a typical outdoor security light fixture and you'll find a PIR sensor like in Figure 5 — the heart of the 'warm body' detector.

To detect a non-moving warm image, an image 'chopper' with wide teeth like a gear can be used to interrupt the image and cause the crystal to detect a series of changes in charge status, just like the Fresnel lens

Figure 1. Color sensors.

Figure 2. Distance sensors.

Figure 3. Photoelectric sensors.

Figure 4. Inductive, magnetic, and capacitive sensors.

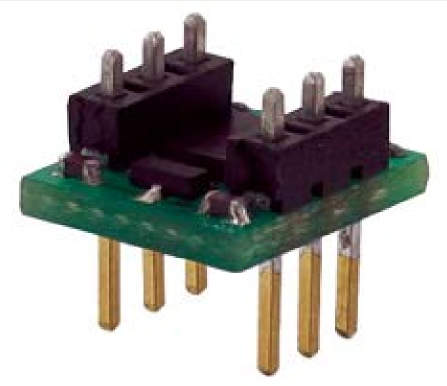


Figure 6. Parallax Hitachi compass.

array with a moving warm object. Check this technology out on Google; it's quite interesting. Passive IR sensors can also be used for two-way control and communications from a laptop computer or a hand-held remote control. They actually become active when they send a signal back to a computer or another robot.

The active IR sensors generally use an IR LED emitting an invisible beam that is, in turn, picked up as a reflected spot on a wall or object by a photo transistor. This same technology can be used as a range finder by having a focused beam emitted from the side or front of the robot at an angle and another series of IR detectors mounted behind a lens pointing straight out. The further away the sensed object, the greater change in detected angle by the detector array. You can try this out with a laser pointer held at an angle as you close in on a wall. If the laser is pointing to the right at, say, 45 degrees, the spot will move to the left as you get closer to the wall.

Lasers are particularly adaptable to robot ranging and object detection. The very inexpensive diode lasers available as pointers and power tool line generators make great robot add-ons. I have seen very cheap (under $5) builders levels with a built-in laser with a line generating lens at Harbor Freight Tools. This straight line projected onto an object and observed from above the laser can determine the shape of the object by some fairly simple edge detection software and a cheap camera. Lasers also generate a collimated beam that does not require a lens assembly to produce a small spot of light.

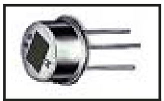


Figure 5. PIR sensor.

Positional Encoders

Positional encoders are probably the second most popular sensor on a robot. Most experimental robots do not have arms and do not use positional encoders to determine the positions of an arm's different joints. They do use shaft encoders on the wheels or motor shafts to determine the number of revolutions of the wheels and thus, the distance traveled by the wheels.

These encoders can use electrical contacts, magnetic Hall-effect detectors, or the more popular optical path broken by rotating teeth or opaque and clear graphics etched on a wheel. Absolute encoders output a binary word for each incremental position and are complex and expensive. Incremental encoders provide a pulse for each increment of shaft movement. The use of two optical channels enable the determination of the direction of rotation. Again, Google these unique devices for detailed information.

The use of potentiometric encoders is popular for the model aircraft servos (before hacking) and are used for 180° or less rotation. Multi-turn pots, partial turn pots, linear pots, and even trim pots can be used as feedback positional sensors. A 25 turn trim pot can be attached to a 25 turn leadscrew and hacked model airplane servo and be used as the feedback device for a fairly powerful linear actuator.

Compasses and Navigation Sensors

I tried to use a standard Boy Scout type of compass in one of my early robots to determine direction. It was one of those liquid filled types that dampened the moving disk that contained the tiny magnet and the north arrow. I pulled it apart so many times to place reflective stickers on the disk that it always leaked and the disk seemed to never point north as it was overloaded by stickers that eventually floated off the surface. It was so sluggish and my crude CdS and flashlight bulb detector array didn't work very well. Needless to say, I soon scrapped this robot compass (*but what a clever idea! - Ed.*)

Much better electronic compasses are available today for the experimenter. Magnetometer and fluxgate compasses began to be available to the experimenter in the '80s and first saw uses in consumer automobile compasses. These devices made autonomous navigation possible for the first computer-controlled machines. However, it was the inexpensive compass modules made by Devantech, Parallax, and others that made interfacing compasses to microcontrollers so easy. As Parallax states: "This compass module made exclusively by Parallax is a dual-axis magnetic field sensor built around the Hitachi HM55B IC. Parallax has made this compass IC accessible by providing Hitachi's surface mount sensor chip with a 3V onboard voltage regulator and resistor protection, all in a 0.3" wide six pin DIP module. The Hitachi HM55B compass module is compatible with the BASIC Stamp's 5V supply and signal levels. Acquiring measurements from the module is made easy with a synchronous serial interface, and even easier with the BASIC Stamp 2 commands SHIFTIN and SHIFTOUT." (See Figure 6.) I've used this one as it is quite small and easily placed in small robots.

Devantech uses a magnetic field sensor as the core of their device and states that "This compass module has been specifically designed for use in robots as an aid to navigation. The aim was to produce a unique number to represent the direction the robot is facing. The compass uses the Philips KMZ51 magnetic field sensor, which is sensitive enough to detect the Earth's magnetic field. The output from two of them mounted at right angles to each other is used to compute the direction of the horizontal component of the Earth's magnetic field." (See Figure 7.) Both of these devices are available from many of the advertisers in *SERVO* and *Nuts & Volts*.

Ultrasonic Range Sensors

Back in the '80s, Polaroid developed an automatic focusing camera that used an ultrasonic range sensor (transducer) to focus the camera's lens on an object, usually a person. I bought one, not to take the fairly expensive instant pictures, but to hack it. After getting into it a bit, I decided to not completely ruin it so I ended up using it as a camera. It piqued the interest of other experimenters also who soon saw many uses for the technology and Polaroid was quick to serve this unique market. They began marketing an experimenter's kit that contained two of the electrostatic transducers, an electronics board that was developed for the camera with a separate LED distance readout, and several flat 6V battery packs. I was fortunate to be given two of the kits by Polaroid to experiment with. (Actually I was accidentally given four, but that's another story!)

The transducers and drivers required a high current pulse, too high for AA cells, so Polaroid also developed a higher current flat battery pack that was behind each film pack. These range finders found their way into almost every robot that was marketed in the '80s. The October '00 issue of the *Seattle Robotics Society's Encoder* featured an article by Dennis Clark entitled "SONAR as I Have Done it" on the hacking of a Polaroid camera for the sonar module and transducer.

Figure 8 shows the uniquely shaped circuit board in the camera and the transducer on the table. A bit of a warning if you ever start playing with one of these — there's a pretty high voltage, like 300 VDC within the module that will certainly wake you up if you touch the wrong areas. I have been awakened a few times in the past. Newer units by other manufacturers use piezoelectric transducers that are a bit smaller.

Sensors for Toys

I have taken *Design News Magazine* for many years, an excellent source of mechatronics ideas for engineers, and the April 30th edition had a supplement highlighting sensors for toy design. We sometimes dismiss toys as low-cost, low-quality products just for kids to play with and eventually tear up, but toy manufacturers have come up with some of the most unique and complex things for just a few dollars. Such is the case with the Zig Zag Zog UFO Saucer featured on the front cover of the "Sensors ... in Toy Design" supplement of *Design News* (see Figure 9).

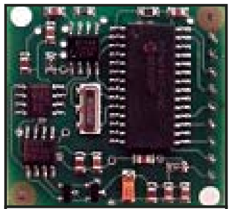


Figure 7. Devantech compass module.

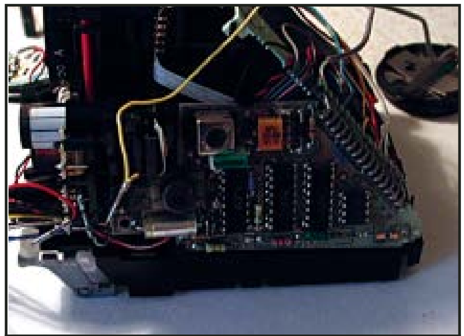
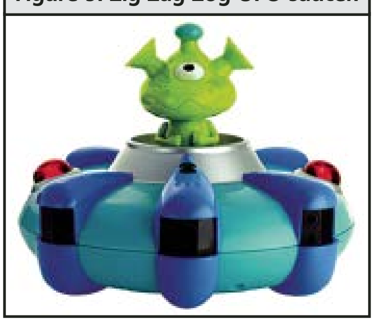


Figure 8. Hacked Polaroid camera sonar.

Designed for toddlers, I know that this $30 toy will be purchased, disassembled, and hacked by any number of adult robot experimenters before spring is over. The objective of the toy is for the little alien to try to avoid being captured by a toddler utilizing three IR sensors to detect obstacles in its path and two more IR sensors in its head to detect the kid trying to bop it there. If it is caught, the toy operates at successively higher speeds to avoid capture.

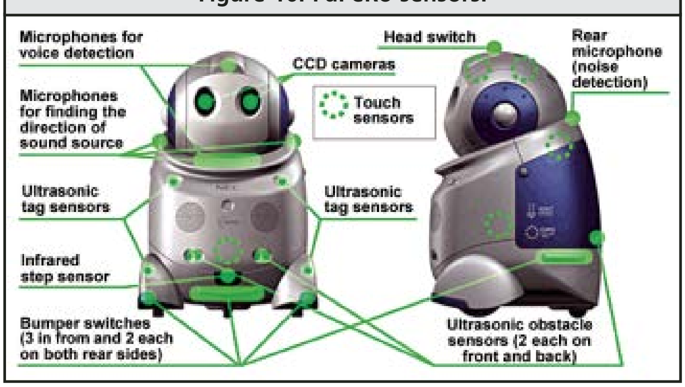
Figure 9. Zig Zag Zog UFO saucer.



NEC's PaPeRo Personal Robot

The large Japanese company, NEC, recently brought a personal robot to the market — the R-100 called PaPeRo through its Personal Robot Research Center. There is no mistaking that this machine is of Japanese origin as it's a bit too cute for most adults but the technology is amazing, as is the case with most Japanese robots. Just as with Sony's Aibo — rest its soul — this robot has all types of sensors strewn about its body to assist in its interaction with humans. Voice and video image (people) recognition are among its most noteworthy. Figure 10 is a chart of some of its sensors. It may look like a cute toy, but it is anything but.

Figure 10. PaPeRo sensors.



Various Other Sensors for Robots

There are hundreds — if not thousands — of types of sensors made today to sense virtually any type of phenomenon you can think of. Besides some of the ones that I delved into, one type that I've used on several occasions is the Figaro gas sensor. Back in the early '70s when I first heard of it, the Figaro gas sensor was used to detect — you guessed it — gas vapors. The company has evolved it line of sensors to detect virtually all types of anything that can be sensed as a vapor. The sensing surface is made of sintered metal oxides of various ratios and metals according to the type of gas to be sensed.

The surface's conductivity changes according to concentration of the gas when reducing gases are absorbed on the sensor's surface. These sensors can detect alcohol in one's breath, propane, HVAC air, gasoline vapor, natural gas, and even cooking smells.

Figure 11. Figaro gas sensors.

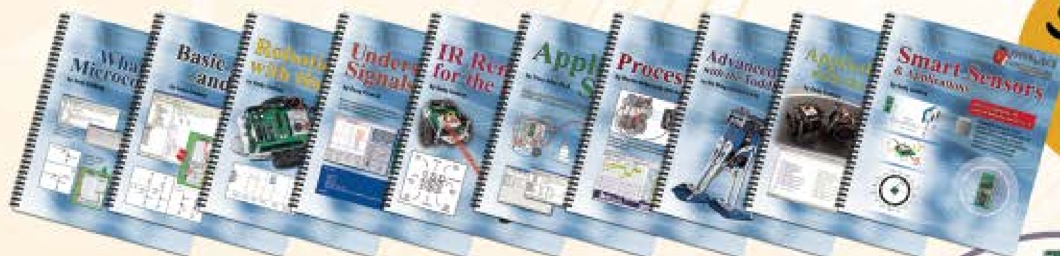
Figure 11 shows a few of the gas sensors available today.

Temperature is easily determined by a simple thermistor, a resistor that changes resistance according to temperature. Atmospheric pressure and other weather-related phenomena can be determined from home weather station components. Smoke detectors are very cheap and great add-ons for a home security robot. Accelerometers and gyro systems are available from *SERVO* advertisers and model helicopter suppliers. Cheap GPS receivers abound and are great for longer-range outdoor robots. Radiation detectors can be used for high security robots. There is no limit of what type of sensor you can place on a robot.

If you want to know more about the hundreds of types of sensors available on the market today, visit the Internet and Google through millions of sites. A book I highly recommend is H.R, 'Bart' Everett's *Sensors for Mobile Robots — Theory and Applications*. This 1995 book still has a great amount of information that is applicable for today's experimental machines and is written by one of the world's most respected robot designers and builders. Bart has built a series of robots that are second to none, especially when it comes to sensor technology. **SV**

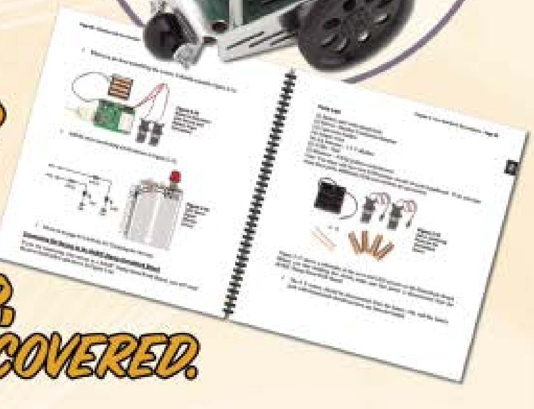
Advertiser Index

MICROCONTROLLER EDUCATION
WE WROTE THE BOOK!



SINCE 1997

10 years ago Parallax created the Stamps in Class program to address the need for affordable electronics and programming educational resources. We now offer 10 comprehensive tutorials designed to introduce students and educators to electronics, programming, robotics, and sensor management with the easy-to-use BASIC Stamp microcontroller. Our interchangeable hardware platform and our free software, tutorials and sample code downloads make outfitting your classroom affordable. Our BASIC Stamp Educators Courses and broad support base help you get started and keep your program growing. **To learn more about Stamps in Class, visit www.parallax.com/sic.**



FROM BASIC STAMP PROGRAMMING TO THE NEW PROPELLER CHIP, WE'VE GOT YOU COVERED.

In the spirit of Stamps in Class, we have created our Propeller Education Kit and Labs to provide an affordable platform for learning cutting-edge multi-processing microcontroller technology. With 8 processors that run simultaneously to perform independent or cooperative tasks, the Propeller chip is an ideal controller for integrated embedded system design projects. As always, our manual, labs, software and sample code are free downloads. Propeller Education Kit (#32305; $79.95) **Visit www.parallax.com/propeller for details.**

To learn more about Parallax BASIC Stamp education visit www.parallax.com/sic. Quantity discounts are available. Parallax Sales Department: 888-512-1024 (Mon-Fri, 7am-5pm, PDT).

PARALLAX INC

www.parallax.com