ROGER ROBOT • ROBOT SAFETY • ROBOGEDDON

SERVO MAGAZINE

FOR THE ROBOT EXPERIMENTER

www.servomagazine.com

JANUARY 2007

Meet Vinnie

One of the Mighty Morphing Bioloids!

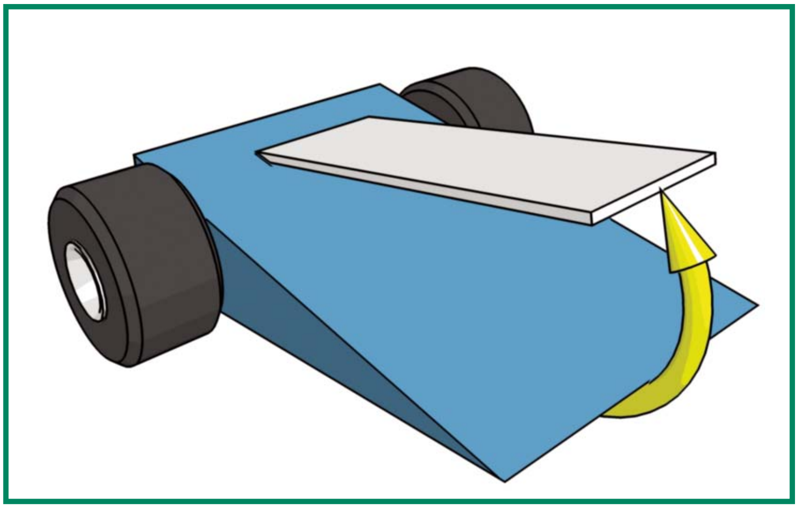
- Intel's Open CV
- Dividers and Timers
- Exploring Soundgin

SERVO MAGAZINE

Columns

Departments

ENTER WITH CAUTION!



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

01.2007
VOL. 5 NO. 1

Features & Projects

Page 20

Mind / Iron

by Michael Simpson

Last year, I built a walker robot for the Robot Fest. I christened this robot "Face Walker." It was featured in the August, September, and October issues of *SERVO Magazine*. My goal was to create a robot that would catch the attention of the spectators and hold their interest while I gave my presentation. I had a nice little speech all planned. What I had not planned on was the "freak factor."

What actually happened was that the spectators were so enthralled by the look of Face Walker, that they didn't hear a word I said. Many individuals left only to return with friends or family members. What is it that made the Face Walker so awe-inspiring? The Face Walker had what I now call the freak factor.

I had a chance to review videos that were taken of the spectators while they were watching the Face Walker in action. Almost all of them were watching the face. The face would animate and make noises as the robot would move. Individuals saw this walker robot thingy that looked something like a spider, but as it turned to face them, it would wink or say something. Immediately they would smile and point. It is when we start to add human characteristics to machines that we start to evoke emotions, which can range from amazement to outright fear.

So, I created a really cool robot, but what can it do? This is the kind of question I often get when showing walker robots. The power requirements for a fully articulated walker are massive. In many cases, you have 12-24 servos that are all energized at once.

The Face Walker always had to have three legs in contact with the ground at any one time to support the total weight of the robot. Even when standing still, a walker will use large amounts of power. This is not true with a wheel-based robot.

The 7.2V 3,000 mAh battery pack would power the Face Walker base for about five minutes before it needed an hour charge. On a wheeled robot of the same weight, you get over an hour of run time on the same battery. This makes walkers very inefficient for most tasks. However, when it comes to education or studying the human condition, you can't beat a walker.

In order to top last year's Robot Fest, I have started early on my next robot exhibit. This robot will be a biped walker with 19 servos controlling various limbs, as well as the neck. Let's call him Kronos. Kronos is still in the experimental stage but while he is in the sitting position, I added a random movement generator and created some routines to simulate breathing. The bot's chest would simply move in rhythm and the head would turn slightly at random intervals.

This was freaky enough, but I wanted to take it a step further and added some random fidget movements. He started moving his arms or would change the angle of his legs as though he was trying to get comfortable. Let me tell you, this even freaked me out. Is it memories of Chucky or is it that we just are not used to human attributes on a mechanical device?

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

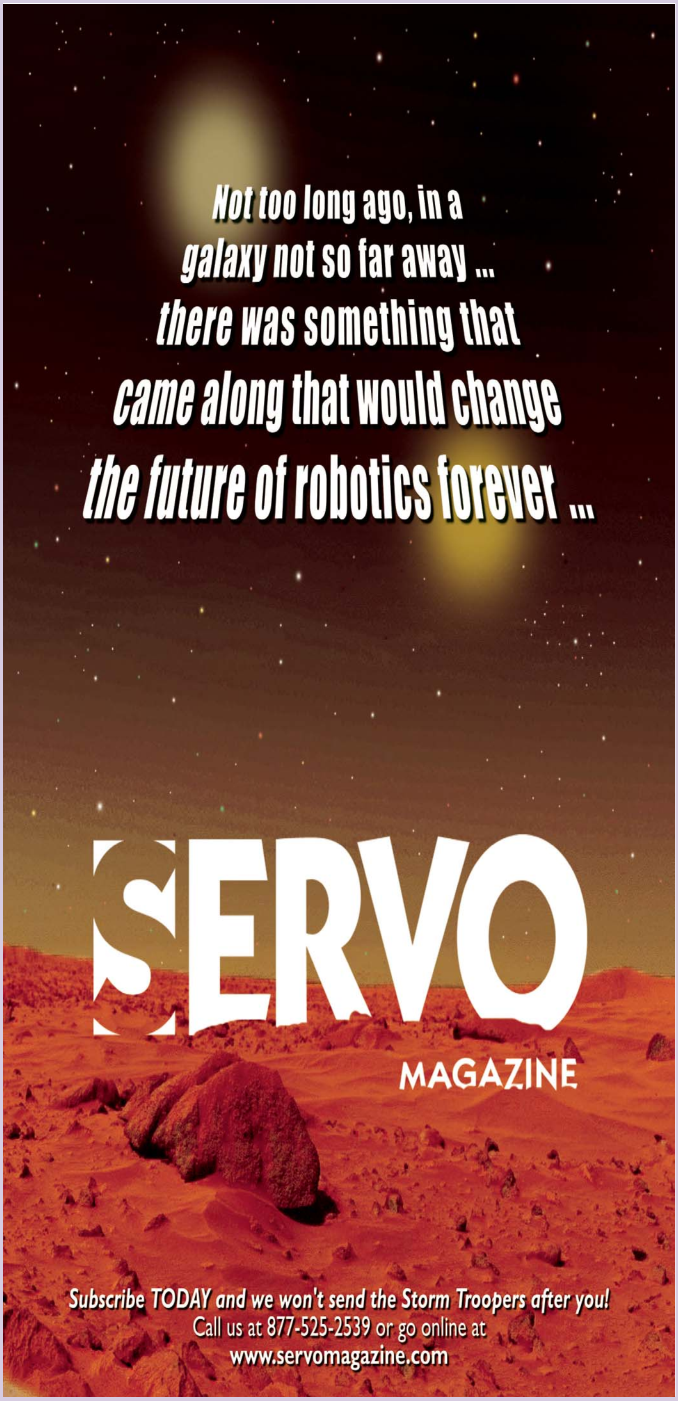
CONTRIBUTING EDITORS
Jeff Eckert, Tom Carroll
Gordon McComb, David Geer
Pete Miles, Kevin Berry
Dave Calkins, Bryan Bergeron
Karl Muecke, Patrick Cox
Dennis Hong, R. Steven Rainwater
Paul Pawelski, Robin Hewitt
Monty Reed, Lawrence Feir
Michael Simpson, Gerard Fonte
Bryce Woolley, Evan Woolley
Aaron Taggart, Brian Benson
Adam Wrigley, Bradley Hanstad

CIRCULATION DIRECTOR
Tracy Kerley
subscribe@servomagazine.com

WEB CONTENT/STORE
Michael Kaudze
sales@servomagazine.com

PRODUCTION/GRAPHICS
Shannon Lemieux

ADMINISTRATIVE ASSISTANT
Debbie Stauffacher



It's probably a little of both.

What is the Robot Fest?

Robot Fest is an annual event held each year in Linthicum, MD at the Historical Electronics Museum. The next Fest will be held on April 28, 2007. This is a free event that exposes many individuals — children and adults — to varying types of robots. Everything from the large BattleBots to the smallest walker will be on exhibit. Be sure to check out the Robot Fest website at **www.robotfest.com**.

Hope to see you there! **SV**

BIO-FEEDBACK

Dear SERVO:

Thank you for running Paul Pawelski's "Beginner's Robotics on $50 a month." Sure, it's fun to lust after multi-jointed 'bots costing four figures, but robotics doesn't have to be expensive. This series promises to be an excellent way for newcomers to become acquainted with the field or hobby without risking a lot of cash. Paul's article affirms that *SERVO Magazine* continues to appeal to a broad range of ages and personal budgets.

K. Bower
Kamloops BC

Robytes

by Jeff Eckert

Are you an avid Internet surfer who came across something cool that we all need to see? Are you on an interesting R&D group and want to share what you're developing? Then send me an email! To submit related press releases and news items, please visit www.jkeckert.com

— Jeff Eckert

Unmanned Helicopter Takes Off

Boeing's ULB Demonstrator on its first unmanned spin around the block. Photo courtesy of Boeing.

Late last year, Boeing Co. (**www.boeing.com**) flew what is dubbed the A/MH-6X light-turbine helicopter for the first time. Under development since 2004, it is actually a hybrid manned/unmanned aircraft that combines the abilities of the existing A/MH-6M Mission Enhanced Little Bird (MELB) with the unmanned aerial vehicle technologies of the Unmanned Little Bird (ULB) Demonstrator shown above. The latter is a modified MD 530F civil helicopter that is readily available from MD Helicopters, Inc. (**www.mdhelicopters.com**).

So far, the Demonstrator has logged about 500 flight hours. In the latest test, the A/MH-6X was flown for 14 minutes as a piloted aircraft, but future testing will involve both manned and unmanned operations. Aircraft performance will be similar to that of the Demonstrator, but with an additional 1,000 lbs (increased to 3,400+ lbs) of payload that can be used for increased range, endurance, or mission hardware. Interestingly, Boeing says that the unmanned hardware and paraphernalia developed for this program can be adapted to any helicopter.

Give 'em the Chair

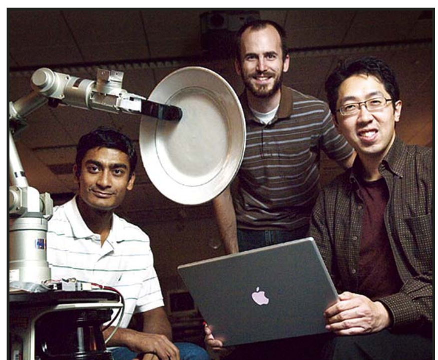
The robotic chair in different stages of collapse and reassembly. Photos by Raffaello D'Andrea, courtesy of Cornell University.

Like many things we encounter in life, the Robotic Chair, a creation of Cornell's (**www.cornell.edu**) Prof. Raffaello D'Andrea and artist Max Dean, admittedly has no utilitarian value. And, like many people we encounter in life, its "brain" is located in its seat. But not everything needs to have a mundane purpose, and the chair — designed as art for art's sake — has only one function: to fall apart and reassemble itself autonomously.

The chair's operation involves 14 motors, two gearboxes, and various other mechanical parts, and a computer uses special algorithms to tell the chair how to find missing components and rebuild itself. Waxing philosophical, Dean noted that its operation is "somewhat like what we do in our own lives. We fall apart and put ourselves back together."

Maybe it has no utilitarian value, but it could have significant entertainment value if you place a couple of them at the dinner table and invite the in-laws over. In any event, the chair will be exhibited in art shows and museums around the world and eventually sold to a gallery or collector.

Bot Cleans Up After You



A plate-grabbing robot, graduate students Ashutosh Saxena and Morgan Quigley, and Assistant Professor Andrew Ng (L to R) — all part of the STAIR project. Photo courtesy of Stanford University.

On a much more down-to-earth level is the latest creation of Prof. Andrew Ng's Stanford Artificial Intelligence Robot (STAIR) project. The dream here is to, within a decade, put a robot in every home and office to take care of routine jobs that you don't really want to do, such as cleaning up after a party, taking out the trash, loading the dishwasher, sobering up Uncle Ralph, and so on. According to Ng, a practical maidbot will need to unite areas of artificial intelligence, including speech processing — navigation, manipulation, planning, reasoning, machine learning, and vision — into one package, which poses a substantial challenge.

In the present stage of development, the team designed an algorithm that allowed STAIR to recognize familiar features in various objects and select the right grasp to pick them up. The robot was trained in a computer-generated environment to pick up a cup, a pencil, a brick, a book, and a martini glass. The algorithm locates the best place for the robot to grasp an object, such as a cup's handle or a pencil's midpoint. "The robot takes a few pictures, reasons about the 3-D shape of the object based upon computing the location, and reaches out and grasps the object," Ng said.

In tests, the robotic arm picked up items similar to those for which it was trained, as well as unfamiliar objects including keys, screwdrivers, and rolls of duct tape. Which brings up the question of what STAIR will do with the duct tape after picking it up. Is a Red Green robot in the works?

Add Your Nomination

Just a reminder that the fourth annual induction ceremony for the Carnegie Mellon Robot Hall of Fame is scheduled for this summer (date to be announced) in Pittsburgh. You are invited to nominate your favorite robot at **www.robothalloffame.org/nominate.php**. Only one nomination is allowed per computer, and (duh!) previous inductees are not eligible.

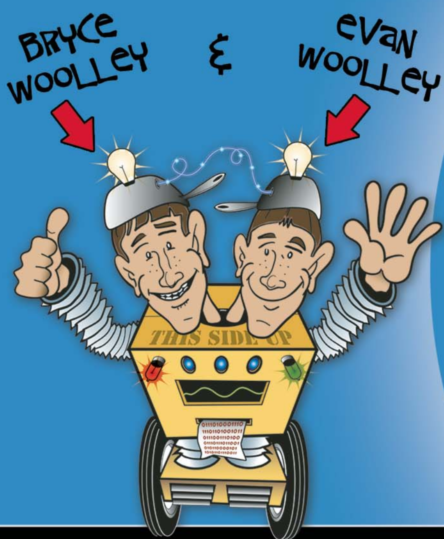
One for the Gripper

If your bot is having trouble grasping things using rubber-clad fingers or other standard end effectors, maybe you should consider grippers that are fitted with razor-sharp needles. Assuming you aren't doing a pick-and-place operation on hamsters, the operation could be improved with the GRN needle grippers from SAS Automation (**www.sasgripper.com**).

The company recently upgraded the product to incorporate titanium needles rather than the previous steel ones, thus offering longer production life and lower overall replacement costs. The needles extend from 3 to 5 mm from the gripper body and are pneumatically activated. The grippers are compatible with the entire SAS end-of-arm tooling (EOAT) line and come in both 14 and 20 mm mounting shaft sizes. Among the usual applications are gripping fabric or mesh, insert molding for floor mats and other automotive materials, and handling of lightweight porous matting. **SV**

The GRN grippers now feature titanium needles. Photo courtesy of SAS Automation.

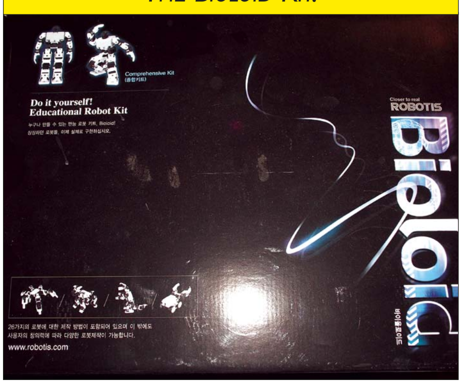
Twin Tweaks



THIS MONTH: Mighty Morphing Bioloid

Robots come in all shapes and sizes, and the Bioloid from the Korean company Robotis certainly takes that to heart. When we first received the Bioloid kit, we thought we would be dealing with a robot that looked like the one in the advertisements and on the box itself — a bipedal servo walker. We were pleasantly surprised to discover that the Bioloid is much more than just a humanoid walker; it is quite literally whatever you want it to be. While many bipedal servo walkers use modular design as a means for the end of achieving an anthropomorphic form, the Bioloid takes full advantage of the modular design and truly invites the tinkerer to let their imagination run wild.

THE BIOLOID KIT.



OOOOOOH, THE INSIDE!

Gestalt Assault

The specific kit that we received was the "comprehensive" Bioloid kit, which included the most structural pieces and 18 Dynamixel servo modules. The Bioloid is also available in two other kits: the beginner and intermediate. We think Robotis deserves a standing ovation for this innovative marketing strategy, because we really feel it provides a viable educational platform — the Bioloid is, after all, an educational robot kit. Apparently there is also an Expert kit that includes a canned curriculum, but we think the distinctions of the Beginner, Intermediate, and Comprehensive kits create an effective naturally progressive curriculum that will teach any roboticist about the intricacies of modular robotics.

The Beginner Bioloid kit comes with just the basics — the CM-5 (the main brain for the robot), some structural bits, four Dynamixel servo modules, and one Dynamixel sensor module. Many of the robots detailed in the instruction manual that can be built with the beginner kit are indeed rudimentary, the simplest being a "crossing gate" that activates one servo module with the touch of a button. But don't think that the beginner kit is by any means boring — it is, in fact, the kit with the greatest variety of robots detailed by step-by-step instructions in the manual. Other possible robots include a crocodile mouth and an interactive duck, and every different design teaches the user about a new aspect of construction or programming, like how to utilize the expansion PCB and the use of for loops.

The intermediate kit provides some more complicated designs that use up to eight Dynamixel modules, including a simple spider and a "battle droid." The intermediate designs are generally more interactive, and they usually make use of a sensor module that is part of the Bioloid kit. We think that the inclusion of a sensor module is a must for an educational robot, because it allows the Bioloid to achieve true autonomy and interact with its environment.

The comprehensive kit includes designs for four impressive robots with up to 18 joints. The four advanced designs are a puppy, a tyrannosaurus rex, the ultra-cool sounding King Spider and, of course, the iconic humanoid.

Apart from all of the obviously cool stuff in the kit, there are some final touches that we really appreciated. Probably the biggest pleasant surprise was that the Bioloid came with batteries included, a phrase practically unheard of nowadays. And if that wasn't cool enough, the batteries are rechargeable, and the kit comes with a charger. We had been working with the kit for a while before we realized that the charger was made for AC outlets, but adapters are pretty easy to find.

Commanding the Bioloid

The programming environment for the Bioloid is an interesting beast, and, like the kit itself, comes in multiple forms. The main program editor for the Bioloid looks like a Frankensteined version of Easy C and Visual Basic, with literal blocks of code that encompass familiar commands like if, else, and for loops. Sometimes the written commands inside the blocks are accompanied by curious pictures perhaps meant to appeal to the more visual programmer, but often times we found our selves scratching our heads and wishing for a Rosetta stone of programming.

The other programming environment is the Motion Editor, something akin to what most other servo based robots have. Individual servo motors are assigned values to move to certain positions. The cool thing is that a series of commands in the motion editor can be saved in a block and implemented in the behavior control editor. Synergy is a beautiful thing.

Robots in Disguise ... As Other Robots

We decided that an effective way to gauge the Bioloid's effectiveness as an educational tool would be to follow the natural curriculum of the kit. With that in mind, we set out to build the simplest model — a crossing gate. While this might seem like an underwhelming project to build first out of such a cool kit, we appreciate the fact that Robotis has provided a simple way for novice tinkerers to get their feet wet. A cursory glance at the kit reveals some pretty intimidating stuff — tons of tiny fasteners, intricate frame pieces, lots of cables of various lengths, and a whole mess of Dynamixel servo modules. With such a daunting kit on hand, a painfully simple design doesn't look so painful after all.

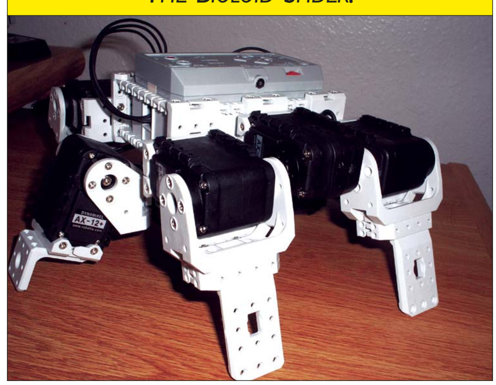
The crossing gate did indeed use very few parts, but it still provided an adequate introduction to the kit and its unique attributes. One such attribute is a component of the Bioloid's design, presumably intended to make construction easier: on every servo module and the CM-5, there are pockets that capture the nuts. This sounds like a nice way to free up some hands when building the robot, and it often is. Sometimes, though, we think the Bioloid kit falls victim to a stack up of tolerances. When you're dealing with mass produced plastic parts that are already a tight fit, small imperfections can stymie even the most tenacious of efforts. Unfortunately, it seems like to us that this might sometimes be the case with the Bioloid. It could just be us, but it seems like on some of the modules that we could only ever get three of the requisite four nuts captured in the pockets. Even so, the crossing gate came together without much difficulty. Most of the beginner designs can be built in a matter of minutes, even for novice roboticists.

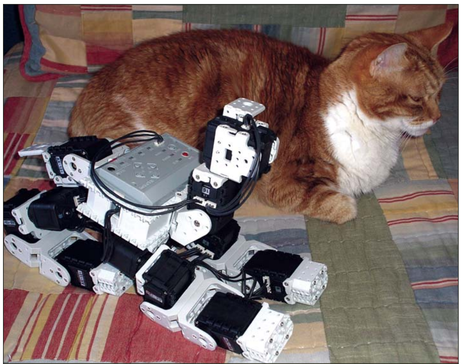
Next we tried our hand at an intermediate level robot — the spider. We found out that dealing with the pockets becomes somewhat easier with practice. A thin screwdriver is a handy way to provide some extra leverage on the nuts, as long as you're careful not to screw up the threads. Intermediate level bots take a bit longer to build — more like a few hours instead of tens of minutes. But once that spider was finished, it was really exciting to see that a ho-hum crossing gate could morph into something so cool.

After graduating from the intermediate level designs, we felt confident enough to tackle an advanced design. The puppy seemed like a good choice, because it would be interesting to see how this modular robot dog compared to other robots that were designed solely with imitating man's best friend in mind. The advanced designs take many hours to complete, so it might be a good idea for roboticists that cannot devote their undivided attention to the Bioloid to find a good stopping point in the middle. Fortunately, that's pretty easy to do with the Bioloid puppy — the limbs are built first, and then everything is connected to the body. The synthesis of the limbs into a complete bundle of puppy joy is by far the most difficult step, but the end result is wonderfully entertaining. All it takes after construction is a quick download of a sample program from the CD, and the puppy is ready to bring smiles to the faces of young and old roboticists alike. The robotic puppy can scamper along at a brisk pace, perhaps not with the agility of other

YOU GUESSED IT ... THE CROSSING GATE!

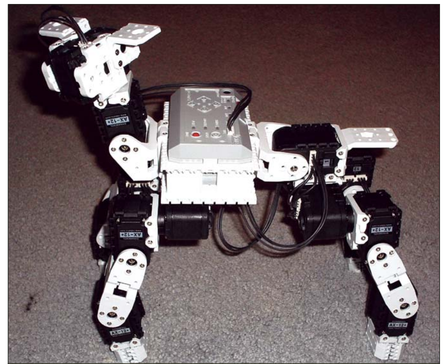
THE BIOLOID SPIDER.





SEE ANY RESEMBLANCE?

THE CM-5.



VINNIE!!

robot dogs, but it's certainly quick and quirky enough to hold its own. The robot puppy can sit, eat from your hand, and even do head stands. And as a testament to the Bioloid's interchangeable nature, a few modifications to the snout later we had a passable robot cat, quite effective at mimicking the real thing. All we had to do was reassign some of the behaviors in the program to buttons on the CM-5 (we had removed the sensor module to achieve a more cat-like face), and the new robot cat was capable of the behaviors characteristic of real cats; namely eating and sleeping. When one of the cat's ears fell off, we affectionately nicknamed him Vinnie.

Vinnie's ear problem actually alerted us to one of the pleasant aspects of the Bioloid's design. Apparently, all the captured nuts hullabaloo was worth something — when the ear fell off, not a single piece was lost. The Bioloid, like many other robotics kits, is afflicted with the robotic equivalent of the common cold — loose screws. It's an inevitable malady unless you use Loctite or Nylocks or some other preventative measure, all of which are out of the question for modular robots that are disassembled and reassembled repeatedly. While the folks at Robotis have not implemented a solution for loose screws in the Bioloid, their somewhat frustrating design elements of little pockets and such will hold onto the loose pieces for dear life, at least cutting down on the frustration of trekking out to your local hobby store to find replacements for lost mini nuts and screws.

SLITHERING ALONG ...

Modular Mayhem

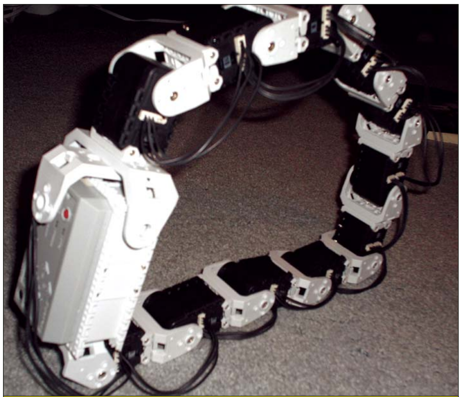
Once we were thoroughly acquainted with the designs included in the Bioloid kit, we were ready to branch out with our own creations. A classic guise taken by modular robots is that of a snake. The highly articulated nature of the slithery reptile is hard to replicate with more traditional robotic designs, so modular robots are the perfect candidates for technological mimics. The snake was literally quite straightforward to build, but since there was no example program on the CD, we had to come up with our own.

An inchworm-like motion was perfectly suited to a synthesis of the Bioloid's motion editor and behavior programming. The motion editor is reminiscent of the classic programming environments for many multiple servo based robots. By hooking the CM-5 into your computer, you can dictate the motion of each individual servo, and you even have a cool 3D animation for feedback.

Programming a snake motion was fairly easy — we just had to have our line of servos give a passable imitation of a sine wave. Our single pulse of snake motion could then be saved and implemented in the behavior control programmer. In many programming environments, a repetitive motion like the one we wanted for our slithering snake could best be achieved by the brute force method of copy and paste. The Bioloid's software allowed for the much more elegant solution of a for loop or some similar command. We think this inclusion of "real" programming in the Bioloid kit is very important for the Bioloid's educational goals, because programming in languages like C is much more common than the gait tables proffered by most modular kits. Soon, we had a slithering snake that was no where near as smooth as the real thing, but it did indeed move.

Another advantage of modular design is the prospect of shape shifting on the fly. Sure, we were able to turn a crossing gate into a spider into a puppy into a snake, but each time we had to

THE ALL-TOO-FAMILIAR ROBOT SNAKE.



SNAKE WHEEL!

DYNAMIXEL SENSOR MODULE.

DYNAMIXEL SERVO MODULE.

completely disassemble and reassemble the bot. What would truly be amazing is a robot that could take the shape of a crossing gate, spider, puppy, and snake all without human intervention. That's right. We're talking about transformers.

Shape Shifting Snake Eyes

If cool factor isn't compelling enough of an argument, there are a multitude of practical reasons that shape shifting robots are a popular quest among roboticists. Simply put, wheels are great for smooth terrain, but when the going gets rough, legs are more capable. So why not always use legs? Because wheels are so much faster on smooth terrain. A conundrum — there seems to be a tradeoff between speed, simplicity, and the ability to grapple with uneven terrain. With shape shifting modular robotics, there doesn't have to be a trade; you can have it all. There is the possibility of having a snake to tackle uneven terrain turn into a wheel to race across flat ground.

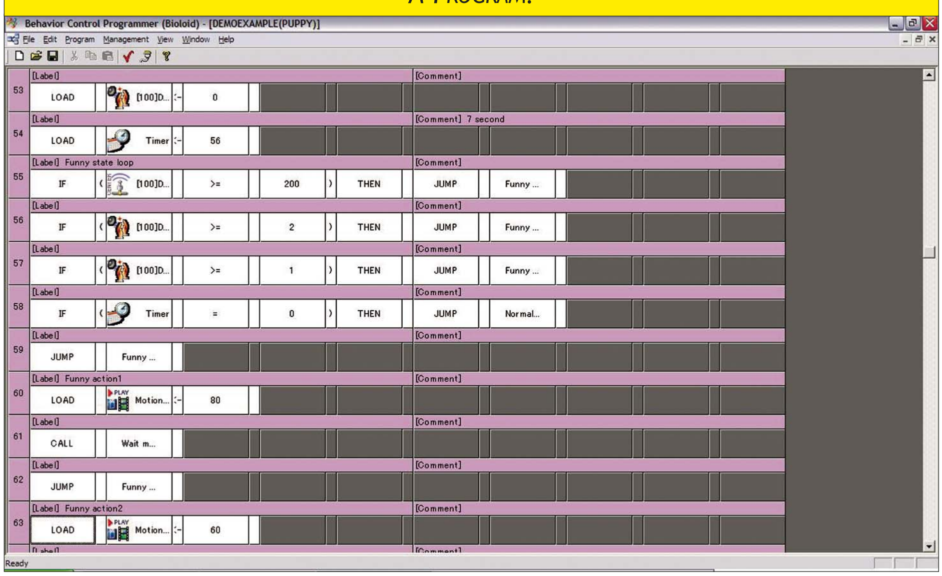
That's what we were aiming to achieve with our snake bot — a snake that could curl up into wheel. It seems a bit underwhelming, but it's harder than it sounds. Real shape shifting modular robots have ways of reconfiguring on the fly, something that the Bioloid lacks. Real shape shifting robots have ways of reconnecting and disconnecting modules without human intervention, either through autonomous latches or some other fancy bit of technology. The only way the Bioloid modules are connected are through nuts and screws, so there really wasn't an easy way to reconfigure on the fly. The best we could do was to add a claw to the end of the snake's tail and hope that would be enough to keep the wheel together. Unfortunately, we already had a flat tire because we needed to include the not so sleek CM-5 in our snake, and the shaky connection made for a wheel that any unicyclist would avoid. Perhaps the time had come to move onto something bigger and better.

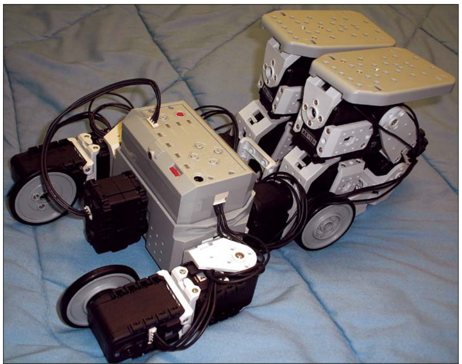
The Derivative of Optimus

Advertisements for the Bioloid and the box of the bot itself are emblazoned with the iconic form of the humanoid servo walker, so it seems appropriate that the humanoid bot is the final, and presumably most difficult, design detailed in the instruction manual. The humanoid design is indeed a challenge, as it consumes every Dynamixel module available in the kit. After assembling beginner and intermediate designs, constructing the limbs of the humanoid should be a snap for any roboticist, but attaching everything to the body is where the real challenge begins. At times we were wishing for more than four collective hands, but eventually we were able to complete the humanoid Bioloid.

The humanoid Bioloid is filled with personality. With the demo program, the Bioloid will fend off obstacles with karate moves, dance, lay down, and of course walk all over the place. While perhaps not as agile as other bipedal servo walkers that we have worked with, none of those other servo walkers could brag that they had been a puppy mere hours earlier. But is the Bioloid the proverbial jack of all trades, and therefore master of none? Perhaps the

A PROGRAM!





Bioloid Racecar ... Er ... Buggy!?

Bioloid cannot claim utter mastery in any of its more complex forms, but its versatility is nothing short of impressive.

Never ones to be anthropomorph-centric, we realized that the Bioloid humanoid might benefit from some shape shifting. Inspired by the robotic heroes on the television shows of our youth, we set out to give the Bioloid wheels. Not wheels to replace his feet, of course, but wheels that would allow the Bioloid to transform from a humanoid into a car. Without the luxury of reconfiguration on the fly, we had to come up with a way for the Bioloid to fold up in order to become a car. That's the way the Transformers did it too, so we were confident when we set about giving the Bioloid a new set of arms.

We think that our final transformed product looked kind of like a sprint racecar, or if that was perhaps too flattering maybe an old style buggy, the kind that supplanted horse drawn carriages. Since one of the preprogrammed behaviors of the humanoid Bioloid was to lay down, all we had to do was modify that to include a very painful looking yoga position and we had our buggy. Only the front two wheels were powered, so our buggy might have benefited from a robotic horse. If we had another Bioloid kit, we would have made one.

Saving the World, One Robot at a Time

Overall, we are overwhelmingly impressed with the Bioloid kit. It provides a truly viable educational platform while being undeniably entertaining. And even if it can't live up to the standards of reconfigureability of real world modular robots, that is hardly a bad thing. The folks at Robotis have created a kit that stimulates the imagination and educates roboticists of all skill levels.

We really think the offering of the Bioloid in three different kits in particular is very conducive to the robot's stated goal of being educational. As any type of robot, modular robots like the Bioloid can be very intimidating and confusing for novices. The beginner kit is, however, perfectly suited for beginners looking for something "closer to real" robots (as per the Bioloid's slogan) than something like LEGO Mindstorms. The Bioloid doesn't come with an explicit curriculum, but we really think that going through and building the robots detailed in the manual provides a comprehensive walkthrough of the construction and programming for the kit.

Also, we think that to be an effective educational tool, the kit has to be widely accessible. Any bipedal servo walker can teach a roboticist of any level something about human motion and mechanical limitations and adaptation, but kits upwards of one thousand dollars simply aren't going to reach that many people. The Bioloid beginner kit will presumably only run a couple of hundred dollars, making it far more accessible to roboticists, particularly novices looking for a kit with which to get involved in robotics.

Once new roboticists are hooked by the beginner kit, the Bioloid kit tells you exactly what additional parts you need to be able to build the intermediate and advanced robots. This step-by-step progression that leads up to a bipedal servo walker is effective for not tossing novices down a mercilessly steep learning curve and also for providing motivation to continue working and learning with the kit.

The great motivation that the Bioloid provides is that the new designs keep getting cooler and cooler. Once a budding roboticist has built the spider, of course they'll want to build the king spider next. And if just the idea of a cool new robot isn't motivation enough, the CD comes with videos demonstrating the abilities of every design detailed in the kit.

The Bioloid is a truly inspirational kit. Even though our first experiences building with it were a bit tedious, we quickly got past that to realize that the Bioloid is flat out cool. And when there's an educational robot kit with great potential for effectiveness that is undeniably cool, everybody wins. And that's how the real Transformers would have wanted it. SV

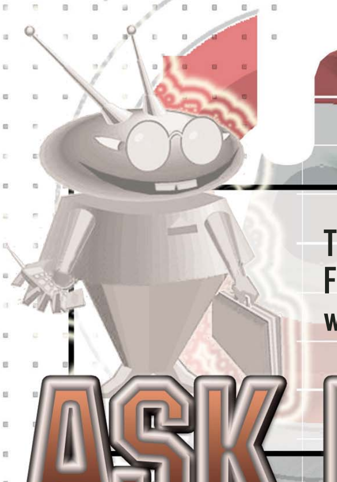
Recommended Website

For more information on the Bioloid, go to **www.robotis.com**

Our resident expert on all things robotic is merely an Email away.

roboto@servomagazine.com

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



ASK MR. ROBOTO

by **Pete Miles**

Q. I have an odd question for you. I am thinking about buying a mini-lathe, but find it hard to spend that much money on just one tool. Can you give some reasons why a hobbyist would want to buy one for their own use?

— Troy Alexson
Toronto, Canada

A. The first thing that needs to be considered is, if it is really needed. There are a lot of parts that can be bought that would meet your needs. Sometimes they are not the best fit, but they can be either modified, or the rest of the project's design altered to fit the parts. Even if you have the tools to make the parts yourself, many hobbyists still look for existing parts to use first before resorting to making them themselves. In most cases, it is less expensive to purchase parts than to build them yourself.

But it is always nice to have your own tools to make the exact parts that you want. Sometimes this is for function, sometimes it is for appearance, and sometimes it is to save money. When people need a special or critical part they can't buy off-the-shelf, they are left with three choices: abandon the need for it, have someone make it for you, or make it yourselves. If you have a friend make the parts for you, you usually end up compensating them for their time. If you go to a machine shop, this will cost you more money than having a friend make it for you. Depending on what the part is, a machine shop may be the only practical option.

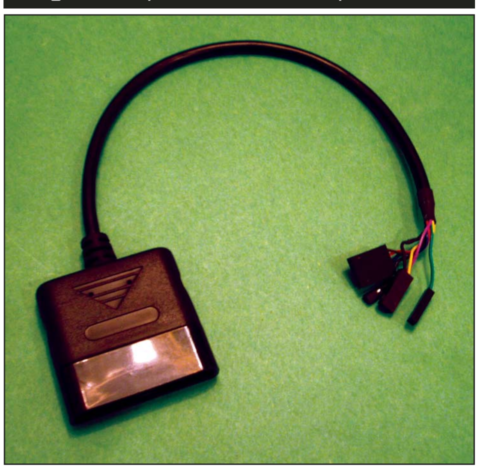
In my case, I bought a mini-lathe to save money. Well, I always wanted one, anyway. I had a project where I needed some special hubs to mount some R/C tires to an axle and hold a sprocket to the hub. I made up some drawings and sent them to local machine shops. The quotes I got were around $600 for the set of hubs. Well, at that time, the mini-lathe I wanted cost a little less than $400. So I bought a lathe and about $100 worth of tools to go with it, and made the hubs myself. In the end, I got the hubs made, saved a $100, and got a lathe for a bunch of other projects. So buying the lathe can actually save you money.

For most people, making parts with their own hands is what brings them joy, and it is the main reason why they buy the tools for their hobbies. It's a really addicting passion, making precision parts to make machines work. This is why the whole robotics hobby exists, making things for yourself.

If you decide to get a lathe, I would recommend that you get a bigger lathe than you think you will need (assuming you can afford it). Eventually, you will want to make things bigger than the lathe you are currently looking at can handle. The machines themselves are usually the cheapest part of the expense. All of the tooling that you get to use the lathe eventually costs much more than the lathe does. And if you end up getting a bigger lathe down the road, you may find out that all the tooling that you currently have no longer works with the new lathe. So, to save money in the long run, get the bigger lathe up front.

Figure 1. Lynxmotion Wireless PS2 Controller.

Figure 2. Lynxmotion PS2 Adapter Cable.



Q. I recently got a wireless PS2 controller from eBay to remotely control one of my robots. The problem I have is that I can't get it to work with my BASIC Stamp. I have tried using the example programs that you showed in the July '06 issue, but they don't work. I know the controller works fine since it works on my Playstation. Do you have any idea why your example program works with a regular PS2 controller, but doesn't work with a wireless PS2 controller?

— Mark Martin
Via Ethernet

A. Thanks for pointing this out. When I wrote that article, I assumed that a wireless controller would work the same way as a regular PS2 controller. Since I didn't have a wireless controller to test at that time, I made this incorrect assumption. The code I presented was based on the code by Aaron Dahlen on his PS2 controller article controlling a five-axis Lynxmotion (**www.lynxmotion.com**) arm published in June '03 in *Nuts & Volts Magazine* (**www.nutsvolts.com**), and Jon Williams' PS2 article that was published in September '03 in *Nuts & Volts*. These example programs all work well with a regular wired PS2 controller. Lynxmotion has quite a few example programs using the PS2 controller with their robots. Most of the example programs from Lynxmotion use the Basic Atom microcontroller (**www.basicmicro.com**), which is fast and quite powerful.

In order to try to figure out why a wireless controller doesn't work the same way as a regular wired PS2 controller, I obtained one of the new wireless controllers that Lynxmotion sells (model number RC-01); see Figure 1. At $19.95, it is a pretty good deal for a wireless controller, cheaper than the regular PS2 controller I have. As a side note, I like the feel of Lynxmotion's wireless controller in my hand because it is a little larger in size and is easier to hold. Lynxmotion also has a very handy adapter cable for the Playstation compatible controllers (Model Number PS2C-01) that has the odd shaped connector that plugs into the controller and has a set of regular 0.1 inch spacing connectors that easily interface with other electronics; see Figure 2. Their $4.95 price tag make this connector cable a better choice than buying and hacking the six foot extension cable I mentioned in my July article.

When I got the controller, I hooked it up to the same test setup shown in my July article, and to my surprise, it did not work. And just like you mentioned, it worked perfectly when I plugged it into my Playstation game console. This was very puzzling.

To try to figure out what is going on, I tapped into the test circuit with my Parallax USB Oscilloscope (**www.parallax.com** part number 28119) to analyze the signals going between the controller and the BASIC Stamp. I discovered something very different than what was expected (see **http://sophiateam.undrgnd.free.fr/psx/index.html**). The spec says the data signal begins when the clock signal changes from high to low (leading edge); see Figure 3. But with the wireless controller, the data signal actually begins before the clock signal, and changes state when the clock signal changes from low to high (trailing edge); see Figure 4. Figure 5 shows a screen capture of the first two bytes of data, device ID $73 analog mode, and ready $5A.

With this bit of knowledge, it becomes obvious that the previous program shouldn't work with a wireless controller since the SHIFTIN function's Mode was configured to use the LSBPOST (read the data Least Significant Bit first after sending the clock signal) mode. Since the data bits actually occur before

Figure 3. Standard (wired) PS2 Data Signal.

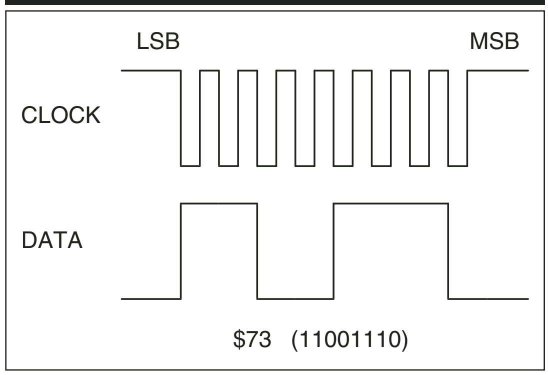


Figure 4. Wireless PS2 Data Signal.

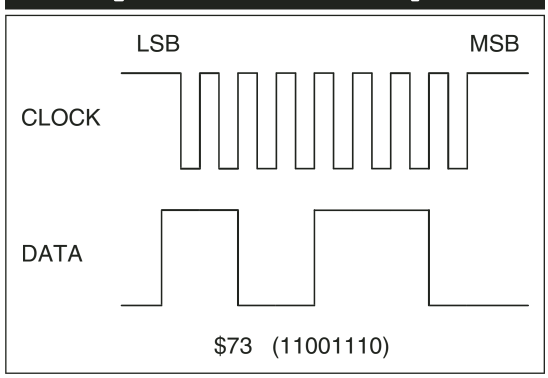
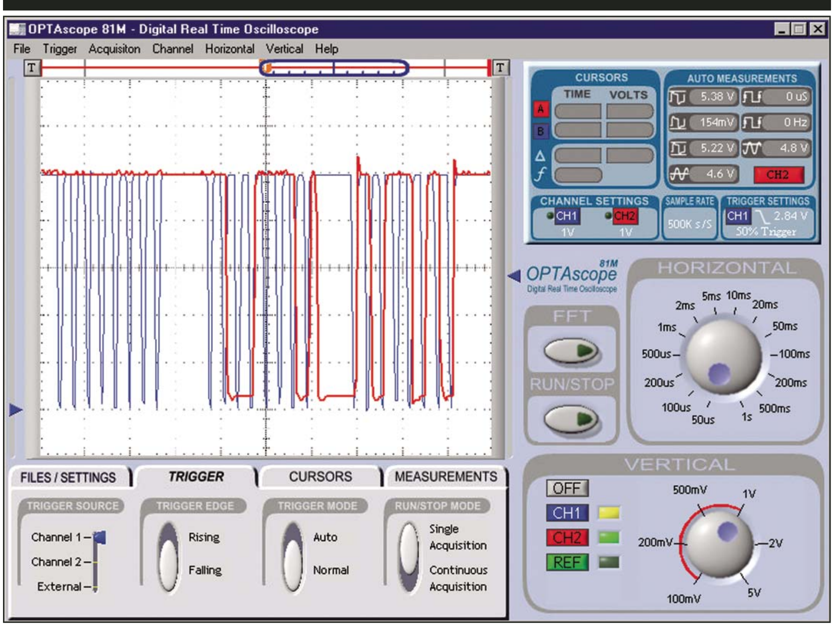


Figure 5. Screen capture of the Parallax USB Oscilloscope. Blue line is the clock line, and the red line is the data line.



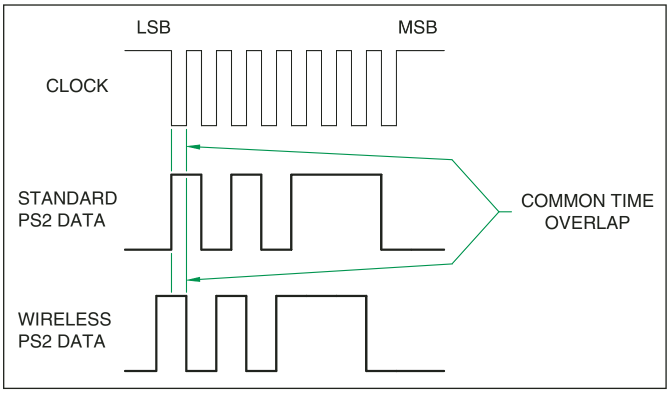


Figure 6. Data signal comparisons between wired and wireless controllers with respect to the same clock signal.

the clock signal, the SHIFTIN's mode needs to be changed to LSBPRE (read the LSB before sending the clock signal).

The program shown in Listing 1 includes these changes, and will work with both wired and wireless controllers. Only one line of code needs to be changed, depending on whether you are using a wired or wireless controller. If you are using a regular wired controller, set the Mode to 3, and set the mode to 1 for a wireless controller. Figure 6 shows the electrical wiring for this circuit.

This doesn't answer the whole question, however. The wireless controller has a much tighter timing requirement than the regular PS2 controller. If the Stamp takes too long to process the data to and from the controller, the wireless controller will time out for a moment and lose its wireless connection with the receiver, and only sporadic information is transmitted. I have tried three different BASIC Stamps with this program. The BS2p24 and the BS2px24 are fast enough to keep up with the communications with the wireless controller, but the regular BS2 Stamp doesn't work because it is too slow.

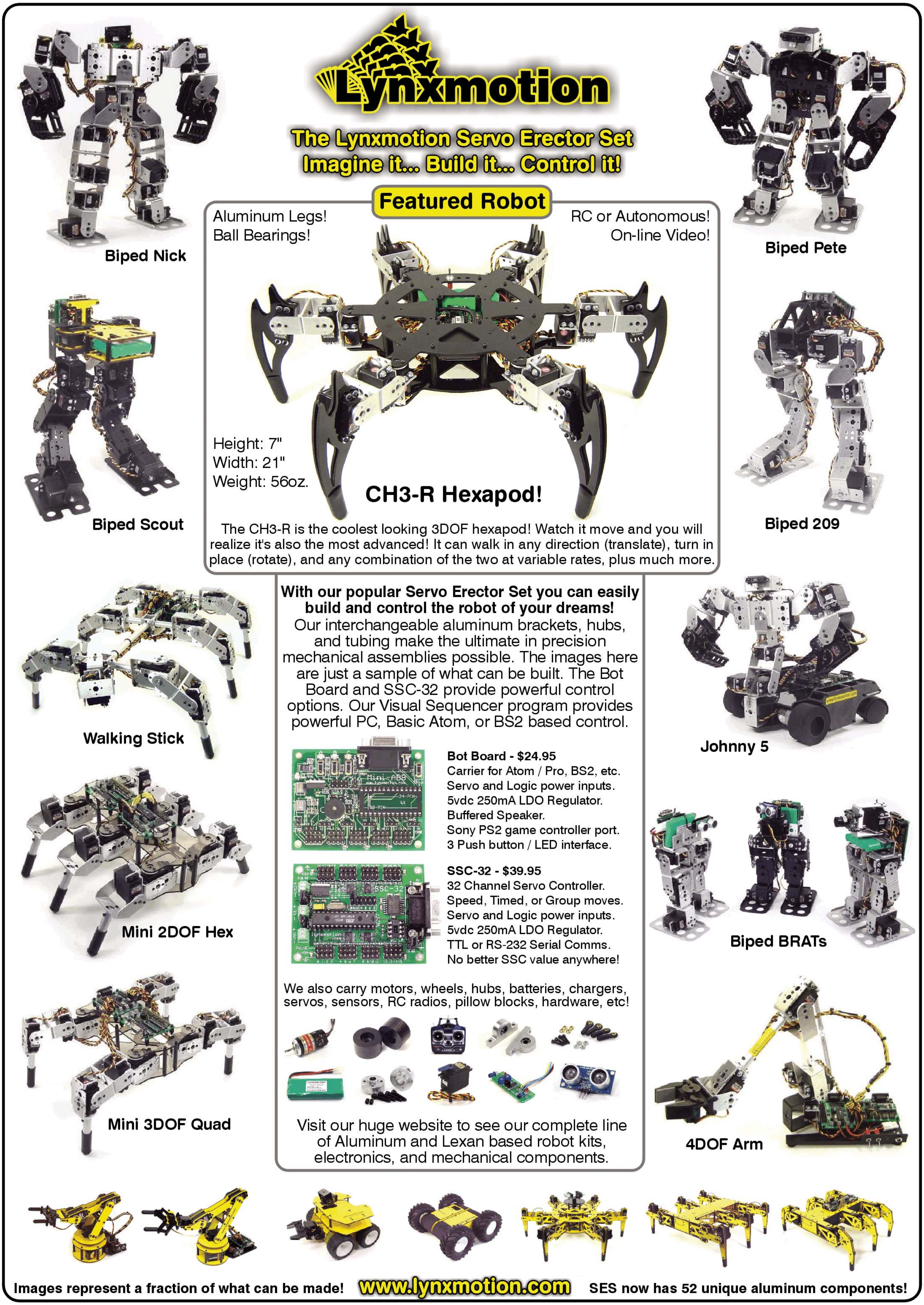
I am not the only one that has observed all this. Read the discussions on the Parallax forums (**http://forums.parallax.com/forums/default.aspx?f=5&p=5&m=138508**) about what other people have learned. It looks like the MadCatz (**www.madcatz.com**) and the Lynxmotion controllers require faster speeds to keep up with the controllers, whereas the Pelican (**www.pelicanperformance.com**) and Thrustmaster (**www.thrustmaster.com**) have been shown to work with regular BS2 Stamps. Logitech (**www.logitech.com**) has a wireless controller, but I haven't heard of anyone successfully interfacing their controller with a microcontroller.

It is interesting how processor speed has a significant effect on how well you can interface a wireless controller and not critical when interfacing with a regular wired PS2 controller. For example, a Basic Atom microcontroller doesn't need to read the data in before the clock pulse (LSBPRE mode), whereas the BASIC Stamp does. In fact, the Basic Atom will not even work if you try to read the data in before the clock pulse. It needs to use the LSBPOST mode in the ShiftIn function to properly read in the data. Measurements on the actual clock signal from a Playstation console indicate that its natural clock speed is about 125 kHz. Since it is transmitting 21 bytes of data, the time required to read in the data from both controller types is about 0.3 ms. A Basic Atom will read in the data in about 4.5 ms. This speed is probably why the Atom syncs up well with the controllers. A BS2px24 takes about 13.8 ms to read in all the data from the controller. Though the Stamp still works, it requires reading in the data in a different format. A regular BS2 Stamp is about six times slower, and does not sync up with the Lynxmotion wireless controller. This indicates that the common overlap time where both systems work is not much slower than 14 ms. Figure 6 illustrates this. Slowing the clock speed increases the common overlap time, but longer clock times can result in the wireless controllers timing out.

Interfacing a microcontroller to a wireless controller is pretty straightforward, but it should be done with the faster microcontrollers such as a BS2px24, Basic Atom, or even the new Propeller chip from Parallax. This is kind-of a long discussion about wireless controllers in general, I am hoping that it provides you enough information to get your controller to work with your robot. **SV**

Listing 1

```
'{$STAMP BS2px}
' {$PBASIC 2.5}
' This demo program has been shown to work with
' BS2p24 and BSpx24, and does not work with regular
' BS2 Stamps with the Lynxmotion wireless controller.

clk   PIN 7    ' Clock Line
att   PIN 6    ' Attention Line
dat   PIN 4    ' Data Line
cmd   PIN 5    ' Command Line

Mode  CON 3    ' 1 = Wireless Controller = LSBPRE
               ' 3 = Wired Controller = LSBPOST

Temp  VAR Byte(8)   ' Controller data
i      VAR Byte     ' Loop Counter

' Main Wireless Controller loop

Main:
  GOSUB Wireless_PS2
  GOTO main

Wireless_PS2:
  ' Read Controller mode $41=digital, $73=analog
  LOW att
  SHIFTOUT cmd, clk, LSBFIRST, [$01]
  SHIFTIN dat, clk, mode, [temp(1)]
  HIGH att
  ' Read controllerr
  LOW att
  SHIFTOUT cmd, clk, LSBFIRST, [$01,$42,$00]
  FOR i = 2 TO 7
    SHIFTIN dat, clk, mode, [temp(i)]
  NEXT
  HIGH att
  ' Display Results
  DEBUG CRSRXY, 0, 2,"Mode: ",IHEX2 temp(1), " ", CR
  DEBUG CRSRXY, 0, 3,BIN8 temp(2), " ", BIN8 temp(3),
         " ", DEC3 temp(4), " ", DEC3 temp(5), " ",
         DEC3 temp(6), " ", DEC3 temp(7), CR
RETURN
```



Lynxmotion
The Lynxmotion Servo Erector Set
Imagine it... Build it... Control it!
Featured Robot
Aluminum Legs!
Ball Bearings!
RC or Autonomous!
On-line Video!
Height: 7"
Width: 21"
Weight: 56oz.
CH3-R Hexapod!
The CH3-R is the coolest looking 3DOF hexapod! Watch it move and you will realize it's also the most advanced! It can walk in any direction (translate), turn in place (rotate), and any combination of the two at variable rates, plus much more.
Biped Nick
Biped Pete
Biped Scout
Biped 209
With our popular Servo Erector Set you can easily build and control the robot of your dreams!
Our interchangeable aluminum brackets, hubs, and tubing make the ultimate in precision mechanical assemblies possible. The images here are just a sample of what can be built. The Bot Board and SSC-32 provide powerful control options. Our Visual Sequencer program provides powerful PC, Basic Atom, or BS2 based control.
Walking Stick
Johnny 5
Bot Board - $24.95
Carrier for Atom / Pro, BS2, etc.
Servo and Logic power inputs.
5vdc 250mA LDO Regulator.
Buffered Speaker.
Sony PS2 game controller port.
3 Push button / LED interface.
SSC-32 - $39.95
32 Channel Servo Controller.
Speed, Timed, or Group moves.
Servo and Logic power inputs.
5vdc 250mA LDO Regulator.
TTL or RS-232 Serial Comms.
No better SSC value anywhere!
Mini 2DOF Hex
Biped BRATs
We also carry motors, wheels, hubs, batteries, chargers, servos, sensors, RC radios, pillow blocks, hardware, etc!
Mini 3DOF Quad
4DOF Arm
Visit our huge website to see our complete line of Aluminum and Lexan based robot kits, electronics, and mechanical components.
Images represent a fraction of what can be made!
www.lynxmotion.com
SES now has 52 unique aluminum components!

GEERHEAD

by David Geer

Contact the author at geercom@alltel.net

RoboGeddon It! Are You Getting It?

You could be getting into RoboGeddon — a good-sized scale version of events like BattleBots or Robot Wars — without traveling very far. The portable robot battle game comes to you via trailer at your local amusement park or event, as well as being available as a permanent install at some attractions (mostly in the UK for now).

The "mobile or static, High Octane Robotic Combat System" brings the arena and ready-made battle bots wherever an event or gathering hires the system and its operators/makers to come. Highly successful since 2004 when RoboGeddon debuted at LEGOLAND of Windsor (UK), the game has gone on to be a highly sought attraction by amusement vendors and consumers.

Part-speak

RoboGeddon's parts include the Arena (either portable or "static" as the creators like to say, meaning set permanently in one place) and four 80 kg robots. The robots can flip each other over and get themselves upright when fallen in battle (with player assistance by remote) by use of the same, single flipper.

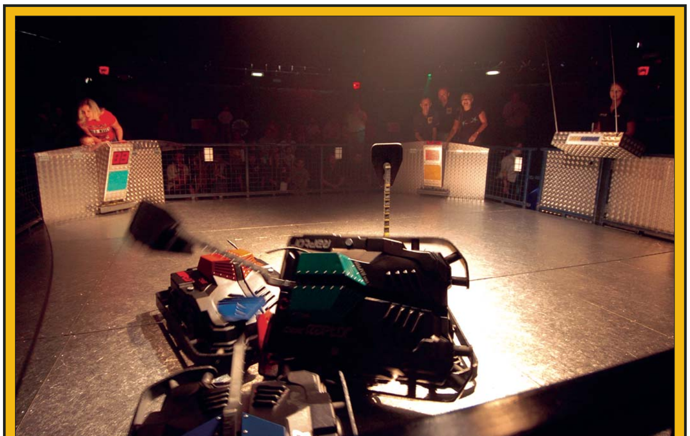
There is also a computerized game control system with music, a public address system, and sound effects that interact with game play. Robots are controlled by radio transmissions.

RoboGeddon can be coin-operated in "stand alone" mode (controlled by the master computer control system), as in a large arcade attraction or be operated by a live operator.

During game play, you try to score points by ramming your bot into the other player's "bump goals," behind which the respective players also control their robots. "The early prototype bumpers on the bots," says RoboGeddon representative Andrew Cotterell, "were made of steel but we discovered that a 80 kg robot traveling at 6 mph into another traveling at the same speed tended to bend the bumpers — we now use plastic bumpers with shock mounts."

A winner is decided by highest score after a round of two minutes.

A RoboGeddon battle in progress, flippers a-flying!



Fully inflated RoboDome surrounds RoboGeddon while crowds watch.

People Watching Assessments Surrounding RoboGeddon

Player console close-up with lots of colors and buttons.

RoboGeddon Raptor robot close-up.

In actual play, younger children generally defeat older kids because they have a much greater mastery of the joystick control and a familiarity with it, according to the RoboGeddon maker (RobotsRUs) company representative Andrew Cotterell.

Likewise, girls generally beat the boys as they concentrate more on how to score points and figure that part out more quickly. "All the boys want to do is flip and bash other bots," says Cotterell (not surprising). Sometimes, players will find out they have been watching the wrong robot and wonder why it isn't responding to the actions they take with the controls.

Back to the Tangibles

The mobile version of RoboGeddon comes with a trailer system with hydraulically-operated sides that drop down to form the actual gaming platform. The platform installs in about an hour and is quite sturdy and permanent looking. This is an "all-weather" gaming system with a site space requirement of 11 meters square.

The RoboGeddon makers had thought that it would take longer to put up the mobile system in bad weather, but it actually takes less time as the crew works harder to get it up and get out of the rain.

The trailer system is complete unto itself with a foldout "crowd barrier system." Ask the company about additional add-on features. In addition to coin-operated and human-operated game play, RoboGeddon can be set up for free play if you are providing it gratis for your family or guests. Game play is computer assisted.

The system is appropriate for indoor and outdoor attractions and anywhere guests or customers may congregate including theme parks, family entertainment centers, hotels, resorts, conference centers, fair grounds, trade shows, malls, museums, and go-kart tracks.

RoboGeddon's gaming arena is 36 square meters, topped by an inflatable RoboDome. Four Cyber Raptor robots come with each setup and can be manipulated to flip other robots over or out of the way or to put themselves upright again once they have been flipped. The flipping and repositioning helps your robot to successfully get to an opposing robot's corner to score.

Each robot is powered by a rapid recharge cell that can be recharged in the charge control system in about 45 minutes. There are four spare cells so that the robots don't ever have to be without power.

The system also comes with installation and maintenance manuals, as well as training manuals for operation. Outside maintenance contracts are available from the company, as well as a phone number for expert consultation.

Static Install Statistics

A permanently installed RoboGeddon can be had cheaper because the cost of the trailer system — which is not required — is subtracted from the total cost. Installs of the static system are highly customizable — including the size of the arena itself — to suit your permanent location.

This system is suitable for venues similar to those for the mobile system and includes piers, zoos, campgrounds, and other locations. Multiple game formats are available for "programmed computer-assisted" play.

Systems also come with four player controls/consoles with coin operation mechanisms, score panels, and bump goals. Remote score panels are also available.

RoboGeddon cost around 900,000 GBPs (about US $1.7 million) to build from initial research to today's product

ROBOSOCCER — UNDER THE MICROSCOPE

RoboSoccer involves four players and four robots in soccer play in a space of about 7.5 by 5 meters.

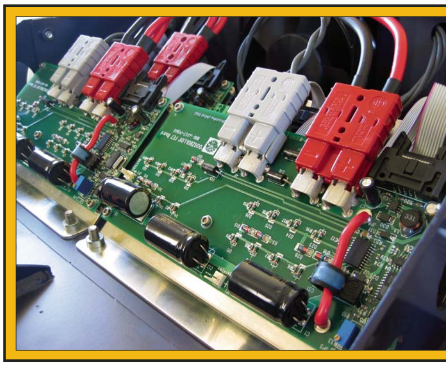
This rugged system is designed to be used constantly, so it is protected by impact-resistant barriers and surrounded by a crowd-retaining barrier.

Its integrated goal system and player control system are each fitted at both ends of the playing field, where players stand and manipulate the controls for a game of soccer that can last up to four minutes. The game system and equipment can be set up in three hours.

The system comes with four robot combat vehicles for soccer play that can maneuver the soccer ball and other players around the playing field. The actual area that is available for play is 24 feet by 16 feet. There are two goal stations, two player stations, and each station controls two robots. The game is operated by a computer system that is programmed to control cash payments to play or free play, as well as the sound system for effects and background music and the robots. The robots operate on four batteries/power cells — one each — and there are four spare cells, as well.

You can see RoboSoccer in the UK and via video clip (provided in the Resources).


Trailer arena close-up with robots.


Circuitry.


RoboGeddon trailer, arena open with dome, close-up.

over a three-year span.

Upgrades

Initially, the game format was extremely complex, too much so for the general public to easily use and appreciate. Much of the engineering and upgrades went into resolving this problem. The initial system was only static, for example, which was too expensive a system to succeed commercially. This led to the mobile system.

RESOURCES

www.robogeddon.com/index.asp
RoboGeddon site

www.Robosoccer.co.uk
RoboSoccer site

www.robogeddon.com/movies/Robosoccer_streamer.rm
RoboSoccer Streaming Video

www.robogeddon.com/links.asp
RoboGeddon permanent install locations and entertainment providers

Team Razer robot.

The mobile system worked well for events but theme parks and ongoing venues required the static systems.

The robots themselves have been improved upon many times for reliability. Improvements to the four robots have included improved drive belts, motor control, lifter mechanisms, bumper systems, crash absorption devices, power handling, heat control, sound use, playability, and the ability to use add-ons.

Further planned upgrades include robot feedback to the control system so it knows how much power the robots have left. The system will soon be able to tell for certain which robots do the actual flipping over. Right now, it only knows when points have been scored. The game currently factors who has won by the number of points given up by other opponents.

Rock 'em, Sock 'em, RoboSoccer and RoboDerby

RoboSoccer has just been launched. This robot game uses the same basic robot builds but no flippers. The game is played on a soccer field inside the game arena by two blue and two red robots. Players win by using the robots to score soccer goals.

The included soccer ball is 14 lbs. This is a coin-operated game for theme parks and similar venues. The robots run all day with no recharge.

RoboDerby is a soon-to-be-released robotic horse that can carry an adult. RobotsRUs — the company behind RoboGeddon, RoboSoccer, and RoboDerby — plans to sell sets of four robot horses so the new owners can transport them by trailer to events and run the horses as robot race horses. By pulling the tail of the horse in front of you, you can slow it down by 25 percent and gain the lead on it.

US Availability

RobotsRUs is in talks with many domestic event promoters who have expressed an interest in involving themselves in bringing the RoboGeddon system to the United States. Stay tuned! SV

ROBOGEDDON FUN FACTS

RoboGeddon is the first commercially sold, mobile robot combat system for game play, available in the UK for almost three years now.

The most unique design problem was giving the robots the ability to reposition themselves upright from whatever laying position they had been flipped in. Other challenges included keeping the system fun while also affordable.

The actual robots, according to RobotsRUs representative Andrew Cotterell, are a cross between a Bat Mobile and a Humvee. They're fast yet maneuverable, since they were designed by world champion robot battlers and engineers, Team Razer. Those who have followed Robot Wars and BattleBots may know of Team Razer, the Razer fighting robot, and Ian Lewis, designer and owner of Razer.

ROBO-LINKS

NEW PRODUCTS

SENSORS

Brushless Motor RPM Sensor

Eagle Tree Systems announces their innovative new Brushless Electric Motor RPM Sensor/Tachometer. The brushless RPM sensor makes it easy to measure RPM.

To measure RPM on a brushless electric plane, heli, car, or boat, just tap one wire from the tiny new sensor into one motor lead, enter motor parameters to the software, and you are ready to go. The brushless RPM sensor is compatible with Eagle Tree's MicroPower, PowerPanel, Seagull, and Data Recorder products. Using the Brushless RPM Sensor with Eagle Tree's Seagull and Data Recorder products requires firmware level 4.XX or higher, which has been shipping since approximately August 2005.

For further information, please contact:

Eagle Tree Systems

4957 Lakemont Blvd. SE, Ste. C-4, PMB 235
Bellevue, WA 98006
Email: **sales@eagletreesystems.com**
Website: **www.eagletreesystems.com**

SOUND PROCESSING

All New SoundPlexer8000

The SoundPlexer8000 unit from GB Innovations will allow you to record/playback sounds and route them to any of the eight speaker outputs, in any combination (for example, sound track 1 can be heard out of speakers 2, 5, and 7 or sound track 7 can be routed to all eight speakers simultaneously). The SoundPlexer8000 can also be configured to trigger these sounds via simple switches or PIR sensors (for example, sound track 2 can be sent to speakers 2, 6, and 8 when input 2 is triggered). Any combination of speakers/inputs/outputs can be programmed through the intuitive menu system via the built-in LCD display. The SoundPlexer8000 makes adding sound to any environment/project easy.

What Can the SoundPlexer8000 Do?

Imagine the sound of a ghost following someone down a hallway, or a haunted room with different sounds coming from different objects around the room. Or, how about a model train layout with different sound effects throughout the entire landscape, yet specific sounds at specific locations when the train arrives or passes by. Perhaps you want to add different sounds to every room of a dollhouse. Or, add sounds to your robot or animatronics character. All this is possible with the SoundPlexer8000.

SoundPlexer8000 features include:

- Records up to eight one-minute tracks.
- Built-in LCD display screen with an intuitive menu system for quick and simple programming, recording, and playback. No computer or host system needed.
- Record from either the built-in microphone or line in from any sound source line out.
- Playback any track through one or all eight speakers simultaneously.
- Each output is capable of driving an eight-ohm speaker with one watt of continuous and up to two watts of peak power.
- Master volume control.
- Eight selectable inputs which are edge triggered. Either low or high going sensor pulses will trigger the input.
- Programmable delays which will delay the track from playing after an input has been triggered.
- Program any switch input to trigger any track (sound) out of any of the eight speakers via the user programmable "Play list." Play lists are stored in EEPROM for data retention.
- Built-in +-5V supply, via phoenix-style connectors for all sensors/switches.
- PIR sensor compatibility.

SoundPlexer8000s are being distributed through Blue Point Engineering and can be ordered directly from Blue Point Engineering's website.

For further information, please contact:

GB Innovations

Tel: **631•891•8034**
Email: **Microcontroller@optonline.net**
Website: **www.bpesolutions.com/asoundeuipspecial.html**

COMBAT ZONE

Featured This Month

PARTICIPATION

First Aid for Bot Builders

● by Kevin Berry

Bot builders are famous for getting minor "dings" when building bots, both at home and in the pits. Every builder should have a minimal first aid kit around, with bandages, peroxide, antibiotic ointment, and aloe vera (for minor burns). The other thing builders should have in their kits is a sense of their own vulnerability. Some injuries can be safely treated at home, some require medical assistance. A few need immediate 9-1-1 calls!

I've seen a builder, in the absence of proper supplies, poke himself in the knuckle with a power drill, staunch the bleeding with a dirty paper towel, and slap a strip of duct tape over the whole enchilada. Two days later, after finishing the bot and competing in the event (I won, by the way), this mystery man finally treated the wound with proper supplies, and amazingly didn't get blood poisoning. I, er I mean, *he*, seemed to think germs had the same priorities he did — that is, finishing the bot counted more than treating wounds.

Some things, such as bleeding puncture wounds, chest pains, dizziness, and double vision, need immediate medical attention. A dead bot builder is good to no one, and by the way, can't compete in events (at least in most of them, anyway.) So build safe, and judge well when it comes to injuries and illnesses. To mangle an expression, "the builder who has himself for a doctor has an idiot for a patient." SV

Entering Combat Robotics at a Young Age

● by Aaron Taggart

So, you may have seen the show *Battlebots* on TV, or heard about the Robot Fighting League (RFL) or the North East Robotics Club (NERC) in the local news or on the Internet and liked what you saw. If you were like me, you probably thought to yourself "Hey, that looks like fun, I could do that!"

The first time I saw *Battlebots* on TV I was 11 years old and flipping through the channels one day when I was bored. I became hooked on the show, and followed it from then on until its final season on Comedy Central in 2002. Around 2000-2001, I started to get more interested, looked around online for more information, and found the Battlebots Forum on Delphi. I lurked around for a while before becoming a member in late 2001. I casually chatted with other builders in the chat room, who were very helpful.

Around 2002 I discovered the North East Robotics Club, which was composed of smaller events primarily based around the Pennsylvania–New Jersey area with weight classes ranging from 1 lb to 30 lbs. In June of 2003 when I was 13, I went to my first event, 12 lbs of Pain, hosted by Pound of Pain and the NERC in Concord, NH. I went to watch, and to see what going to an event in person was like, since I had only seen videos online and *Battlebots* on TV. The fights were great and the builders — some of which I had talked to previously online — were friendly. I then competed at my first event in August of the same year, in Syracuse, NY that was hosted by Team Infinity, the builders of the Super Heavy Weight robot "Final Destiny," who had appeared on TV on *Battlebots*. I did pretty well for my first time, placing 3rd with three wins and two losses, losing to the 1st and 2nd place robots.

Since then, I have competed at 20 events and attended 23. There have been times when I was frustrated enough to think about taking a break for a while after doing poorly at an event, but then I bounced back and did well for the next few. As of this writing, I have the #1 ranked active 12-lber, according to Botrank (an online robot ranking service), and the 3rd ranked 12-lber historically. There is no limit to what you can do if you apply yourself and are willing to learn.

The best advice I can give to young people is to look around. Look at people's websites, read build reports, and watch fight videos. This will help you get an idea of what is going on, and also give you some ideas of your own. Be prepared to learn a lot. There are a lot of tricks you can pick up by watching what others do. There are no real "tricks to winning," but there are plenty of ways not to lose. If you find an event going on near you, go check it out. You will have a good time, you will get to see the robots in person, and you will meet the builders. Do your best to make some friends in the community, which will help you immensely in the future; a good friend goes a long way. I would not be nearly as successful now if it weren't for the advice given to me by friends Jr. of Team Mad Cow, Jon of Team Anarchy, and James Arluck.

With their advice, tricks, and general knowledge I was able to learn a lot of things I would never have thought of on my own. The best part of going to an event is that you can see first hand what works and what doesn't. There isn't really anything to be taught in a classroom about this, as it is more of an acquired skill/understanding. You will pick up many skills that can be useful later in life if you decide to pursue a career in engineering, manufacturing, or machine work, and it also looks good for college applications as an extra-curricular activity.

One important thing to remember is if you come off as a jerk, people will generally react in an according manner. (Don't be the person who gets ribbed on all the time, you want to be with the in-crowd who *does* the ribbing.) The easiest way to get on the nerves of builders is by asking questions that have answers that are easy to find. When online, poor spelling turns off a lot of people — so use some kind of a spell checker. When you go to your first event as a competitor, remember to

Resources

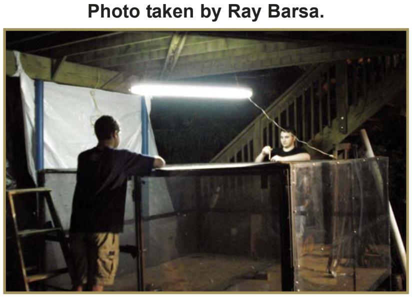
RFL — **www.botleague.net**

NERC — **www.nerc.us**

Botrank — **www.botrank.com**

BuildersDB — **www.buildersdb.com**

Author and Vadim Chernyak setting up the arena at the House of Slackers event. Photo taken by Ray Barsa.



From left to right: Brad Handstad, Chad New, Darci Trousdale, Aaron Taggart, Jon Durand. Photo is from 2005 RFL Nationals in San Francisco, CA.

Author at House of NERC 2006 working on Rants Pants. Photo taken by Jon Durand.

bring all of your parts and tools you will need. It is always important to have spares, as well as tools and other necessities such as a power-strip so that you don't constantly have to go hunting for tools from others. However, do note that many builders will be more than willing to lend you a tool or plug into their power strip, but remember to bring back what you borrow. There is a vast amount of information available on the Internet, and the best way to get ahead and off on the right foot is to use it to your advantage. SV

GONE POSTAL
THE BUILDING OF A 12 POUND ROBOT

● by Brian Benson

There are many different techniques and approaches to building 12 pound combat robots. They range from the extreme of designing the entire robot on the computer, to the gathering of a pile of parts and putting it together whatever way looks best. For builders that are just beginning, the second method is a great way to get a robot built, have some fun, and gain some experience. I had only two weeks to build Gone Postal, so this technique worked out well.

I decided I was going to build a 12 lb robot because many of the parts I had would fit easily into a robot that size, and I could cost-effectively use cordless drill motors and batteries. My first step was determining what type of weapon the robot would have. After considering a variety of designs, I concluded that a hammer robot would be the best choice. It would allow for the drive and weapon motors to be the same, it would be easy to build, and not many hammer robots were around so it would be a little different.

Components

My first choice before building the robot was picking the mechanical components and the materials to build the frame out of. For the mechanical and electrical components, I choose: two IFI Victor 833 speed controllers because I already had them on hand; three 12 volt Harbor Freight cordless drill motors; batteries from the drill for power; a Team Delta Bigger Dual Ended Switch (Part #RCE225) to control the hammer motor; a PCM receiver; and 4" Colson wheels. For building materials, I wanted something durable, lightweight, and easy to work with. I chose 1/2 inch thick UHMW (ultra high molecular weight polyethylene) for the sides, 1/8-inch thick 6061 aluminum for the base plate, and .09-inch thick surplus carbon fiber sheet for the top plate. With my pile of parts gathered, I could begin on the robot!

Organized Pile of Parts

I began by assembling the batteries; building two 12 volt packs using the cells that had come with the drills. Now that I had the basic components, I played with different layouts until I found one that I was happy with based on how well it would drive and perform and how compactly it fit together. I then weighed everything in order to get a basic idea of how well I would meet the 12 lb maximum requirement, shown in Figure 1.

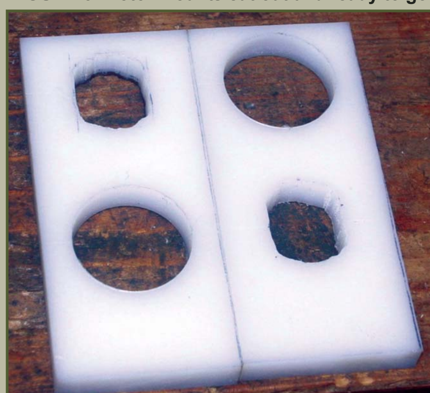
The Frame

With the components chosen, I was able to determine the basic frame size and begin. I cut the 1/2" UHMW outer frame with a wood saw and the aluminum base plate using a plasma cutter, although a jigsaw would have also been a fine substitute for the plasma cutter for those without one. Figure 2 shows the frame members and base plate laid out.

Drive Train

Now that I had a frame, I needed to start on the drive train. I used off-the-shelf wheel hubs made specifically for attaching Colsons to drill motors. I modified the drills to lock the clutch and used more of the 1/2" UHMW for the motor mounts. I cut out the center holes with a hole saw on the drill press and a scroll saw for the second square shaped hole. Figure 3 shows

FIGURE 3. Motor mounts cut out and ready to go.



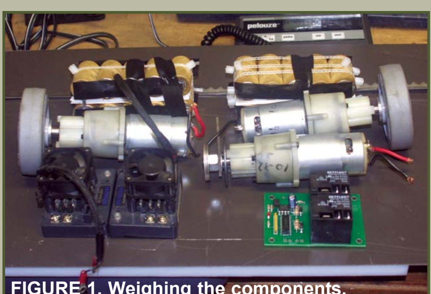


FIGURE 1. Weighing the components.

FIGURE 2. Test-fitting the frame pieces.

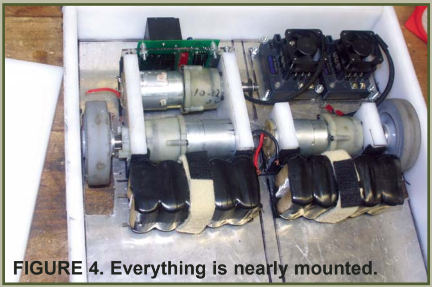


FIGURE 4. Everything is nearly mounted.


FIGURE 5. The wedge is completed and attached.

FIGURE 6. Gone Postal prepped and ready for battle!

the finished mounts ready for use.

Finishing Touches

With the motor mounts complete and sides cut out, all I had left to do on the base plate was to cut out the holes for the wheels, which I did using a plasma cutter. With each subcomponent complete, it was time to combine them. I decided to use 10-24 size screws to fasten it together. The motors were mounted, the sides were attached to the base plate, and the electronics shock mounted. Shock mounting is always critical for electrical components. As you can see in Figure 6, the speed controllers and relay board were shock mounted to an 1/8" sheet of polycarbonate, which was then shock mounted to the frame. The batteries must always be easily accessible; to meet this requirement, I chose to use Velcro straps to secure them.

At this point, as you can see in Figure 4 the robot was missing the front half. For this, I chose a wedge to allow me to gain control of other robots to help me use my hammer to its full effect. After choosing an angle, I made the wedge and attached it using the same methods as the rest of the frame. Figure 5 shows the robot nearly done!

Everything was complete except for the top armor, the hammer, and the wiring. I made the top armor out of .09 inch carbon fiber sheet for its strength and minimal weight. For this, I used a diamond wet saw, the key word being wet. Carbon fiber dust can be extremely harmful and should not be breathed in under any circumstances.

Through a combination of having water constantly pumping onto the cutting edge to eliminate dust and a respirator, I was able to easily and safely cut the top plate. The hammer was more fun, for this I found a piece of 1/8" steel and used the plasma cutter to create the hammer shape I was looking for. To attach the hammer to the drill, I took advantage of the threaded shaft, clamping the hammer arm between the shoulder of the shaft and a tightened nut. The nut and threaded shaft then had an 1/8" hole that a pin went into to keep the nut from backing off. With everything mounted and ready to go, I wired the robot up using primarily 14 gauge wire. Figure 6 shows Gone Postal at the event ready for battle.

Conclusion

Overall, Gone Postal has undergone four iterations since it was first built, gaining spring steel armor over the side UHMW and a titanium wedge and hammer arm. In each iteration, it has become a little smaller and a little tougher, but the same core building techniques have been used each time.

Figure 7 shows Gone Postal in its prime. It is ranked historically in 30th place out of over 250 robots, racking up a 35 fight record and known to many as the bot that just won't die. It has proved to be one of the most fun bots I have built and competed with, being cheap, effective, and different. Further details on Gone Postal including photos and videos can be found at **www.robotic-hobbies.com**. SV


FIGURE 7. Gone Postal version four with its 1/2" diameter titanium arm and spring steel skin.

EVENTS

RESULTS — October 14 - November 13

NERC www.robotconflict.com

House of NERC 2006 — This event was held on 10/14/2006, in Wichendon, MA. Results are as follows:

- *Antweights* — 1st: "Absolutely Naut VDD," spinner, Anarchy Robotics; 2nd: "Disctruction 2.0"; 3rd "Almost

There," Wedge, Small Bots.

• *Hobbyweights* — 1st "Rants Pants (of Doom)," Wedge, Not So-Boring Robots (Botrank #1); 2nd: "Igoo," spinner, Mad Scientist; 3rd: "Shake Appeal," spinner, EMF.

• *Featherweight* — 1st: "Gnome Portal," lifter, Robotic Hobbies; 2nd: "Mangi," spinner, Half Fast Astronaut; 3rd: "Power of Metal," beater, EMF.

HORD Fall 2006 — This event was held on 10/21/2006 in Omsted Falls, OH. Results are as follows:

• *Fleaweights* — 1st: "Mr. Bigglesworth," saw, Udanis, 2nd: "Low Lift."

• *Antweights* — 1st: "Hit or Miss," saw, AC/DC; 2nd: "Fred Fred Burger," wedge, Udanis.

• *Beetleweights* — 1st: "D2," drum, D2; 2nd: "Chop Chop."

Halloween Robot Terror — This event was held on 10/28/2006 in Gilroy, CA. Results are as follows:

• *Fleaweights* — 1st: "Change of Heart," wedge, Misfit; 2nd: "Atom Bomb."

• *Antweights* — 1st: "Fire Eagle," wedge, Missfit; 2nd: "Stumpy," wedge, DMV; 3rd: "Pooky," wedge, ICE.

• *Beetleweights* — 1st: "Toe Poke," lifter, Kick-Me; 2nd: "Unknown Avenger," flipper, ICE; 3rd: "Itsa," spinner, Bad Bot. SV

TECHNICAL KNOWLEDGE

Four Bar Lifters in Combat Robotics

● by Adam Wrigley

When it comes to flipping and lifting weapons, four bar mechanisms are the crème of the crop. You've seen them on TV or at your local combat robot competition, and now you want to know more about them. If you've never seen one, take a look at Figure 3 on the next page. The largest benefit of the four bar design lies in its ability to have any tip trajectory you desire. You can lift an opponent nearly straight up, or out and up, as seen in Figure 1. The "out and up" motion is what most people tend to use, and allows you to actually tip over the opposing robot. Simple lifters as seen in Figure 2 tend to lift up and away, causing the other robot to fall off the tip before full extension is reached.

How Are They Powered?

There are a huge number of different methods for powering a four bar lifter. Electric motors can be used along with several stages of gear reduction to power either the front or rear bar through torsion. Linear actuators can be geared using a rack and pinion method to accomplish the same end result. Other mechanisms include linear actuators to nearly any part of any bar or joint in the mechanism, or to any bar through a pin/slot technique. There really is no simple way to explain the system. In fact, there are entire college courses and textbooks devoted solely to analyzing four bar mechanisms. To keep this article shorter than a 400 page textbook, I'll simply concentrate on four bar systems powered by torsion. Torsion power is the simplest method of powering a four bar, both in construction and analysis.

Finding Input Torque

Now, we are looking at a four bar mechanism with torque being applied to either the front or rear bar. What do we do now? We could get an equation for the trajectory based on the powered bar angle input, do a force balance, and a dynamic analysis. We could do that. However, that isn't really necessary. The simplest way to solve this problem is to look at it as a work balance. The work you put in the system will equal the work you get out of the system. The work you put into the system is equal to:

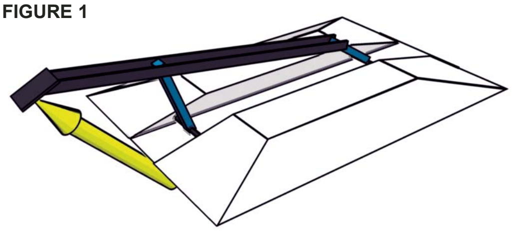
Work = Torque * Angle

With angle being in radians (radians = degrees * pi/180) and equal to total angle traveled by the input bar. The work output is:

Work = Weight * Height

where weight is the weight of the opposing robot and height is in the same units as your torque (if you used ft-lbs for torque, use ft for

FIGURE 1



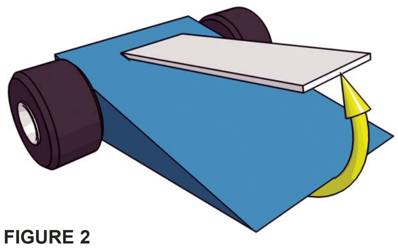


FIGURE 2

height). Drawing out your four bar system to scale in fully collapsed and fully extended forms will allow you to measure the height difference and angle difference from start to finish. This will allow you to figure out the average torque you will need to lift the opposing robot by setting the previous two equations equal to each other and using the result:

Average Torque = (Weight * Height)/Angle

Average torque can be useful, however, you will probably want to find the maximum torque. To find the maximum, you would need to draw many scale drawings of the system at many different locations through the trajectory, and do this procedure for each set of two drawings. After doing this, you would be able to plot a torque vs. angle graph. It would also be extremely time consuming. Luckily for you, I've made a program that does all of this work without you needing to draw anything.

Go to **www.totalinsanity.net/tut/mechanical/4barfrontbar.php**. This page has the download link, as well as a description of how to run the program. The program currently only solves systems powered by the front bar, but you will most likely want to use a front bar powered system, anyway. It will output a graph showing the trajectory of the tip, as well as a Torque vs. Angle graph. The graph can then be used to find a relative maximum torque to use in calculating your gear ratios.

Binding

Binding is a problem that occurs when any of the joints in the system gets close to 180° or 0°. The system simply gets stuck, and the forces go through the roof when this happens. If no force is applied to the system and the front bar is rotated, then whenever any angle in the system gets close to these values, there will be binding. Powering the front bar leverages the other robot in such a way that binding rarely occurs during extension of the arm.

It is important, however, to make sure that your mechanism does not fully extend, and you should have a mechanical stop to prevent this. If your mechanism fully extends, then it could have problems collapsing because of binding. A front bar powered system only avoids binding in extension, not retraction.

Using the Graph for Design

When using the program I mentioned earlier, it will show some forms of binding on the graph. If you see a spike, then your system is binding. Make sure to have your mechanical stop kick in before the spike in the graph. The spike should be at the end of the motion of the arm. If it is not, then you should change your design. When trying to read off a torque for your design calculations, there are also some other concerns that need to be taken into account.

Normally, the highest torque is at the start of the motion of the arm. There is a spike seen at the end of the Torque vs. Angle graph in most systems, but this value should not be used, since your mechanical stop will kick in before this. Also important to remember is that the torques shown with this program — or calculated by hand using the same method — are the torques needed to hold the opposing robot. You will want a larger torque value than this if you want to lift the opponent with some speed. One last mistake that is commonly made is using the stall torque of your motor when calculating your gear ratio. A conservative calculation for figuring out your gear ratio is this:

Gear Ratio = 4 * (Torque read from graph)/(Stall torque of motor)

The gear ratio you use should be read as Gear Ratio:1 and is the ratio going from the motor to the front arm. Using this equation, you can also look at the RPM of the motor and try to figure out how fast you will extend your arm. A simple equation for time is listed below:

Time = 60*{(Angle/360)/[(RPM/2)/(Gear Ratio)]}

Time will be in seconds. RPM is the no load RPM of the motor, and Angle is the total angle difference from start to finish of the front arm. This equation won't be exact, but it will give you a good approximation of the time needed.

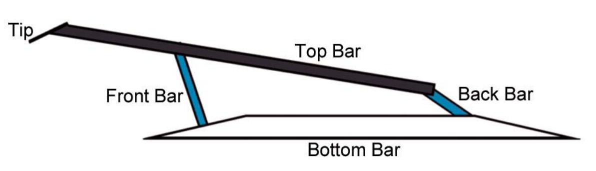
Final Design Ideas

You can start typing values into the program I mentioned earlier, but it is always best to start with good values at your first iteration. The rear bar of the four bar mechanism should always be the shortest. The bottom bar, which is normally part of the chassis, should be the largest. The top bar, which is part of the arm, should be the second longest, and the front bar should be the second shortest.

This is pretty much all you need to know to design your first four bar mechanism using an electric motor to power the front bar. If there is enough interest, I can provide a similar explanation for different four bar systems in future issues. SV

FIGURE 3

Generic Four Bar Lifter



PRODUCT REVIEW — Astroflight 109 Charger

● by Bradley Hanstad

The heart and soul of a combat robot comes down to the batteries you use, but one thing that most people never consider is the charger needed for said batteries. I have used several chargers over the years, and as times change, so do batteries. The shift to lithium-polymer, lithium-ion, and even lithium-manganese has evolved combat robots in the smaller classes. As the technologies ever expand and grow, bigger and higher capacity cells are being made. This meant I had to find a charger to support the batteries I wanted to use for a 12 pound combat robot.

Most would assume the Triton as a natural choice for charging as it has excellent flexibility (charges multiple types of batteries), as well as its two new forms (Triton 2 and Triton Jr.), but I have found my new favorite charger to be the Astroflight 109 (**www.astroflight.com**) lithium charger. The Triton is only so flexible, as it is limited to four cells in series (14.4V) and only five amps max cell capacity. The scheduled 12 pounder was going to have up to a 22.2V pack for the weapon setup. There was an obvious need for a new charger to handle the higher voltage.

I searched for a charger to handle over four cells, but many simply don't, until I found the Astroflight 109. I bought it and tested it right away on a smaller antweight robot t. It's simple it is to use, and gives tons of information.

Just plug your battery pack in before giving the charger power, and it will begin discharging the pack, but not into the all-so-critical dead voltage range. It automatically senses the number of cells, and won't damage them thanks to its low voltage cut-off. After discharging is done, unplug the pack and it automatically goes into charge mode. Adjust the knob to the desired AH rating of the pack, and plug it back in. It couldn't be simpler.

The Astroflight charger uses a three-phase charging sequence, which will act like a trickle charge at the end of phase 3 to make sure the pack is fully charged before damage can be done by over-charging. The Astroflight 109 can charge up to nine cells in series, which is a 33.3 volt pack! It can also charge at a rate of 50 mA to nine amps!

This is an awesome all-around lithium charger that will be used heavily in all of my robots that use lithium technology. I am so thankful to the kind people at Astroflight for making this available to people like me, so I can get my dreams of bigger and more destructive bots to be a realization! **SV**

EVENTS

UPCOMING — January and February

Kilobots X — This event will take place on 1/20/2007 at Cameco Spectrum 2007 at the University of Saskatchewan.

The Saskatoon Combat Robotics Club is having its 10th Kilobots event January 20-21, 2007. The venue is the largest student-run engineering exhibition in North America! Go to **www.kilobots.com** for further information on this event.

Bay Area Robot Fights — This event will take place on 1/27/2007 in St. Petersburg, FL. It is the fourth event in this annual series; a conventional insect battle run by some very unconventional people. Fun for the whole family. This event data is tentative at time of publication.

ComBots Cup 2007 — This event takes place on 2/9/2007-2/10/2007 in Oakland, CA. Go to **www.robogames.net** for further information. $10,000 Heavyweight prize, $3,000 Middleweight prize. Venue and schedule dates are tentative at time of publication.

Motorama 2007 — This event takes place on 2/16/2007-2/18/2007 in Harrisburg, PA. Go to **www.nerc.us** for further information. 150g-30lb Combat Event. Ants fight in 8' box, Fairies fight in the 5 x 5 insert. Beetles-Featherweights fight in 16 x 16 box. All completed forms and entry fees must be received by 1/15/07. This is going to be another awesome event at the Farm Show Complex! **SV**

Exploring SOUNDGIN

Six Voice Sound/Music/Speech Synthesizer

My first fascination with electronics was sound-making gizmos — sirens, warblers, choo-choo train effects — you name it, I built it! When the venerable 555 timer IC was introduced, it seemed that every other issue of the electronics magazines had a project using this chip to make noise. I made each one, of course.

by Gordon McComb

Robotics and sound generation are natural playmates. They go together like R2D2 and C3PO. Yet curiously, few of the robots you see demonstrated at clubs or on the Internet have a sound feature, even a simple one like chirping when an obstacle is detected. Nearly every toy robot includes sound effects, and our custom designs should, too. The reasoning is the same: Sound effects are an effective user interface. R2D2-like "bio sounds" are a useful diagnostic tool. Music and voice help to humanize a robot. And the sounds can aid the entertainment value of the bot.

However, simply wanting to add a sound effects module to a robot is different than actually doing it. Most sound circuits are analog; the typical robot controller is digital, so there are extra interfacing steps to take. While many of the popular microcontrollers have sound-related functions, they are limited to simple notes or DTMF telephone tones. "Music" is simplistic, and voice synthesis is impossible.

A new product just introduced aims to change all of this. The Soundgin, from Savage Innovations, is a six voice synthesizer that is capable of reproducing complex sound effects, music, and even speech. The Soundgin — which also goes by its more formal name of SSG01 Sound Coprocessor — is available as a standalone DIP or SOIC surface mount chip that you can integrate in your own designs. It is also available in a prototyping board that includes its own built-in amplifier, power regulator, and optional RS-232 serial converter — the latter if you wish to connect it to a PC. Probably the more common connection scheme is to wire it directly to a serial port on a microcontroller. The Soundgin is compatible with the OOPic (also made by Savage Innovations), Parallax BASIC Stamp, and many others.

Let's take a closer look at the Soundgin: what's inside, how it works, and ways you can use it to add cool sound, music, and speech effects to your robot.

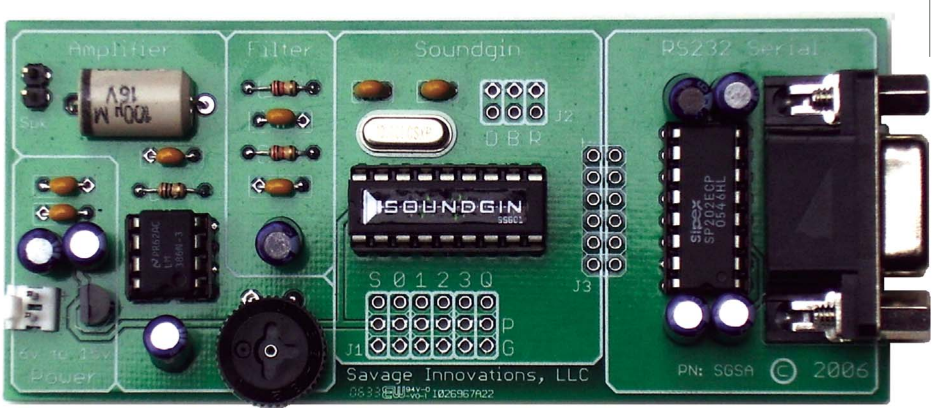


Figure 1. The Soundgin prototyping board includes the Soundgin chip, an integrated audio amplifier (with volume control), and an optional RS-232 level shifter with DB-9 serial connector. The latter is used if you want to operate the Soundgin from a PC.

From Oscillators to Mixers to Sound

In order to better understand the Soundgin — or any sound-generating device — it's handy to know a little about sound in general. Sound is waves that travel through some type of medium, like air or water. The size of the waves determines its amplitude, or volume, and how close the waves are from one another determines its frequency. The farther apart the waves, the lower the frequency, and vice versa.

When humans speak, a curious little doodad in the throat called a larynx vibrates, and as air from the lungs passes by the larynx, sound waves are produced. These waves travel through the air, and anyone nearby hears us when the waves enter their ears. A similar process is involved in making sound from a guitar string (the string vibrates, disturbing the air around it) or a saxophone (a wooden reed vibrates as air is blown over it).

The Soundgin doesn't use a larynx, string, or reed. But it does use the same general principles of making waves. Sound production begins in one of its six oscillators. These oscillators are akin to a larynx, except the Soundgin has six of them. As such, it is said to be a six voice sound synthesizer. The oscillators are grouped in two sets of three; each set is routed to its own mixer, so that the signals from the oscillators can further be controlled. More about this in a bit.

Finally, the outputs of the two mixers are combined into a final audio output. This output is connected to an amplifier to increase the level of the signal produced by the Soundgin chip. Most any audio amplifier will do. The Soundgin prototyping board uses the LM386 sound amplifier, which is inexpensive, easy to use, and requires few external parts.

The proto board, shown in Figure 1, can be powered by a nine volt battery, and has connections for a DB9 serial cable, power, and speaker terminals.

From the amplifier, the signal is sent to a speaker. Until now, the "sound" from the Soundgin has just been electrical impulses. The speaker turns those impulses into physical vibrations that move air. Our ears pick up that air movement and perceive it as sound.

A Closer Look at Soundgin's Oscillators

The heart of any sound-making module is its oscillators, and the Soundgin is no exception. As noted above, Soundgin consists of six independent oscillators. Each oscillator is a separate sound synthesizer, or voice. Any voice can be used by itself, though it is common to use them in combination to produce more elaborate sounds. For example, the output of one oscillator can feed into another to produce a slow rising and falling siren effect.

Each voice consists of a frequency generator, an amplitude modulator, an envelope generator, and a ramping/target console.

The frequency generator controls the pitch of the sound. The lower the frequency, the lower the pitch. If you picture a piano keyboard, the keys to the left have a decreasing frequency, so they have a lower pitch. The keys to the right have an increasing frequency, so they have a higher pitch.

Not all sounds are at the same volume level, and any sound-making device needs a way to control the overall level of any oscillation. Each of Soundgin's oscillators contains an amplitude (volume) control that can be varied from 0% — which is no signal output at all — to 100%.

Sound is often characterized by the shape of the waves that produce it, not just the frequency of those waves. The shape of the wave affects the harmonics of the sound — harmonics can be said to be the interaction of the wave with itself and other sound waves being produced at the same time. Soundgin produces eight types of wave shapes, including sine wave, square wave, and saw tooth.

The envelope generator is a kind of dynamic volume control that changes over time. The envelope is characterized by four distinct phases: Attack, decay, sustain, and release — often referred to simply as ADSR. These phases are graphically shown as a series of ramps. Attack indicates how fast the oscillation comes to full volume. Decay is how fast the oscillation ceases. Sustain is a constant volume of the sound after the decay until the note is released; and release is how quickly the sound fades out. These parameters — along with the wave shape — define the timbre of the sound. It's what helps makes the oscillation sound like something familiar to us — a piano or a violin, for instance. Each of these instruments has a peculiar ADSR envelope.

Soundgin provides further sophistication in the sounds it produces by using separate ramp and target controls for each oscillator. These controls have a similar function to the ADSR envelope, but are more robust — and trickier to use. When the oscillator's amplitude target is set, the amplitude of the oscillator will move towards the target at the rate determined by a special transition value, until the oscillator's amplitude has reached the target. One use of this feature is to make various rising and falling sound effects, like phaser guns or alien bio sounds.

Mixing and Matching

You can create a multitude of sounds simply by setting Soundgin's six oscillators to some frequency, adjusting the amplitude and wave shape, and having them run completely independently. Whether such a mix would sound pleasant is another matter! And it misses a lot of potential of Soundgin as a complex sound and effects generator.

As noted above, Soundgin allows you to control one oscillator with another. In music synthesis, this patching technique is used to produce an infinitely rich flavor of sounds. On the average analog synthesizer, any oscillator can be patched to any other oscillator. The Soundgin chip does not provide for unlimited inputs and output patching, but rather it predefines the patching between specific oscillators. This technique greatly simplifies the programming you'll need, yet you are still able to produce a wide variety of sound and music effects.

Recall the Soundgin has six oscillators, and that these oscillators are separated into two sets of three each. The oscillators for the first set are labeled A1 through A3; the oscillators for the second set are labeled B1 through B3. The functionality for both sets are identical. Oscillator 2 (A2 or B2) can control the amplitude modulation of Oscillator 1 (A1 or B1). Similarly, Oscillator 3 (A3 or B3) can control the frequency of Oscillator 1.

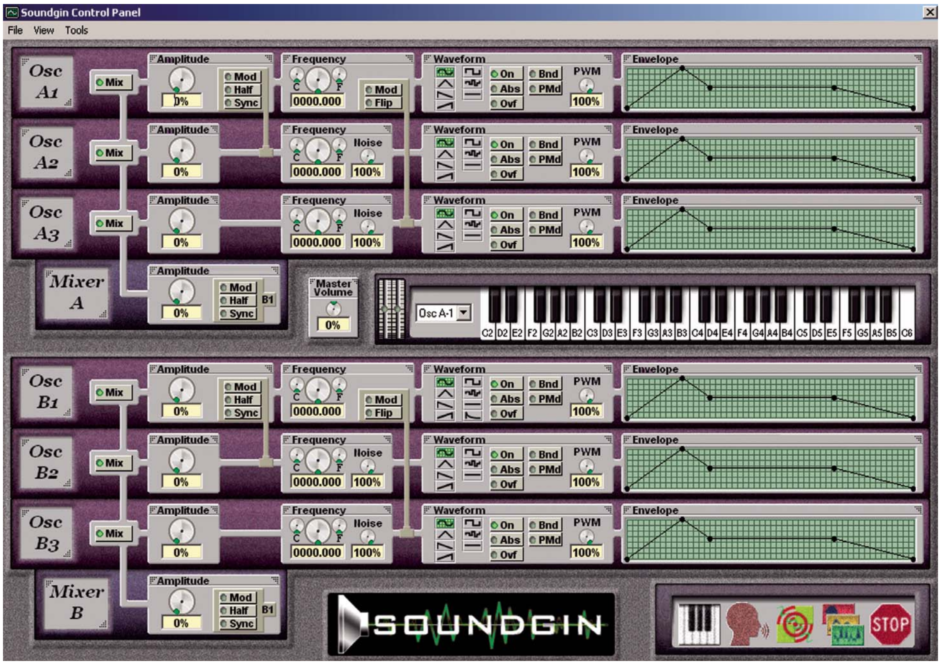


Figure 2. The Soundgin Windows control panel provides a graphical interface for programming and experimenting with the Soundgin chip.

Example of amplitude patching: Oscillator 1 produces a steady 1,000 Hz tone. Oscillator 2 produces a slow 1 Hz tone, which would ordinarily be below the range of human hearing. By patching the output of Oscillator 2 into Oscillator 1, the output changes in volume once a second. It creates a phasing effect that sounds like something out of an old science fiction movie.

Example of frequency patching: Oscillator 1 again produces a standard 1,000 Hz tone. Oscillator 2 produces a slow 1 Hz tone, and is patched into the frequency modulator of Oscillator 1. Rather than alter the volume (amplitude) in this patching, Oscillator 3 changes the frequency of Oscillator 1, producing a rise-and-fall wailing siren effect.

Soundgin provides additional mixing and patching options that greatly increase the variations in the sounds it can produce. The options are too numerous to mention here, but are reviewed in the product documentation. And remember that Soundgin has two fully independent mixers that feed into one final output. The A and B sets of oscillators can each produce their own sounds, and can be mixed together to make overlay effects: a siren on top of a warbler, for instance.

Using the Soundgin Windows Control Panel

Perhaps the best way to play with the sound making features of the Soundgin is to use its program control panel, which runs under the Windows PC environment (see Figure 2). The software can be downloaded from the Soundgin website — see the Sources box for additional information. The console graphically depicts Soundgin's six oscillators, and shows how the oscillators can be patched together. You also see the ADSR envelopes and other controls by which you can modify the sound effects.

For a great introduction to Soundgin and its capabilities, click on the Presets button in the lower-right corner. You will see a collection of 31 preset sound effects, as shown in Figure 3, such as Space Drive, Wow, Chopper, and Blip Chatter. Click each one to hear what they're like. As the effect plays on the Soundgin, note the action of the oscillators. (If any sound is too loud, you can adjust the volume by scaling down the Master Volume Control.)

Now try combining two sound effects together. Click Mix A beside the Presets list, and choose a sound effect. Click Mix B, and choose a different sound effect. Both sounds are now combined into the final output of the Soundgin. Play with different combinations by selecting Mix A or Mix B, and clicking on a new sound. You can quiet any mix by selecting preset number 32, which is silence.

Music Generation

Music is little more than sound that has certain qualities that we humans find aurally pleasant. By linking one tone to another, a musical score is produced. Soundgin generates music by setting its oscillators to frequency, wave shape, and ADSR envelopes to make piano, organ, and other instrument sounds.

As with general sound production, the Soundgin Windows control panel can be used to experiment with music synthesis. In the console window click the Keyboard icon, and a 49-key keyboard appears. Use the mouse to click any of the keys, and its corresponding note is played through the Soundgin chip.

Note that the keyboard uses just one oscillator to produce the sound, and you can select which oscillator to use. This allows you to readily experiment with different effects. Play around with the ADSR ramps, for example, and you change the timber characteristics of the notes. The sound wave buttons similarly change the color of the tones. Notice, for instance, that a saw tooth wave makes the sound a lot more "reedy," like that from a saxophone.

By combining two or more oscillators to produce separate voices you can create a kind of musical ensemble. The Soundgin Windows control panel can only play one note at a time through one oscillator at a time, but the Soundgin itself is capable of playing multiple notes through multiple oscillators, giving you a polyphonic sound and music synthesizer.

"I Am Tobor, Your Robot!"

Perhaps the most remarkable feature of Soundgin is its ability to produced synthesized voice. Again, it's all done by cleverly combining the chip's oscillators to produce the elements of speech. When these elements — called phonemes — are strung together, the sound is recognized as speech. This all appears simple enough, but in reality it's quite difficult to achieve completely synthesized speech, and still have the speech be recognizable. Soundgin does a remarkable job, and the mechanical voice that it makes is perfect for the average robot.

The Soundgin Windows control panel is the easiest way to experiment with, and develop, speech synthesis for your robot. Click the Speaker icon in the lower right, then click the Phrase Editor button right above. A Phrase Editor window appears, shown in Figure 4, where you enter the text and/or phonemes you want the Soundgin chip to speak.

For example, to have the chip say "hello," you merely type hello into the Say Data box, and click the Say It button. Soundgin responds by saying the word. The default voice — with its male-like pitch — is automatically selected for you. But you can change the pitch by clicking on one of the Musical Scale buttons on the left-hand side.

As noted previously, Soundgin works with parts of speech known as phonemes. These phonemes are displayed in the Phrase Editor dialog box. They include the th of "this," or the wuh (w) in "water." You can play with the various phonemes sounds just by clicking on their corresponding buttons. The sound continues until you click another phoneme, or hit the Shut Up button.

When you enter an English phrase such as "hello," the Phrase Editor actually does a lookup to see what phonemes are contained in that word. The dictionary lookup table is stored in a simple text file in Windows INI format. This dictionary file, named SGWords.txt, is found in the Soundgin directory on your PC. Looking up the definition for "hello," you see the phonemes that make it up are:

.he .e .le .le .oe

The leading periods indicate to Soundgin that these are phonemes, and not English words. You can enter these same phonemes into the Phrase Editor and Soundgin will speak them. This is a good way to experiment with phoneme-based speech synthesis. You can try substituting different phonemes to make different sounding words. In all, the SGWords.txt file contains several hundred word definitions, and you can add your own, as well as modify existing ones. You can also load new words into the dictionary file by entering them into the Phrase Editor, but I found manually editing the file to be easier.

Programming the Soundgin

So far, we've looked at operating the Soundgin from its Windows control panel. This is a good way to learn about the capabilities of the chip, but it is not a practical means of programming sounds for your mobile robot.

Communicating with the Soundgin is relatively straightforward. It uses a standard serial interface that is supported by the PC and most microcontrollers. Soundgin supports data rates of 2400 or 9600 bits per second.

When used with the PC, you need a Maxim MAX323 interface chip, or something similar, in order to convert the five volt TTL signals required by the Soundgin to RS-232 levels needed by the PC. Note that the Soundgin prototyping board is available with an optional RS-232 level shifter chip and DB9 connector for hooking up a standard serial cable.

Soundgin is programmed by setting its various internal registers. It's not a particularly easy process, but programs like the Soundgin Windows control panel, and others to come, promise to make programming the chip easier. To send data to the Soundgin, you send a series of bytes that always begin with the Escape character (decimal 27 or 1B hexadecimal). Program statements then consist of one or more bytes, until all the data has been sent.

Soundgin supports hardware flow control so that the chip requests data as it needs it. The Soundgin toggles one of its pins — the CTS line — when its buffer is half empty. In this way there is no interruption in sound playback, as long as your PC or controller is keeping a watch on the CTS line. Soundgin regulates the playback speed, so your robot's microcontroller can send the data at the required baud rate, then continue with its other chores. In the event of a series of programming commands, such as a continuous song, the microcontroller must listen on the CTS line to see when it toggles, and feed data to Soundgin before its buffer runs out.

Even the voice synthesis feature of Soundgin is programmed by sending serial bytes to the chip. After setting up the chip's registers for voice synthesis, the phonemes are sent one per byte. For example, to say "hello," the chip receives the sequence 212, 206, 221, 221, 228.

Maximizing the Sound Output

All sound-making devices are limited by the mechanism that turns

Figure 3. There are 31 sound effect presets built into the Windows control panel. Try them alone or in combination (one for Mix A, the other for Mix B) to see what the Soundgin can do.

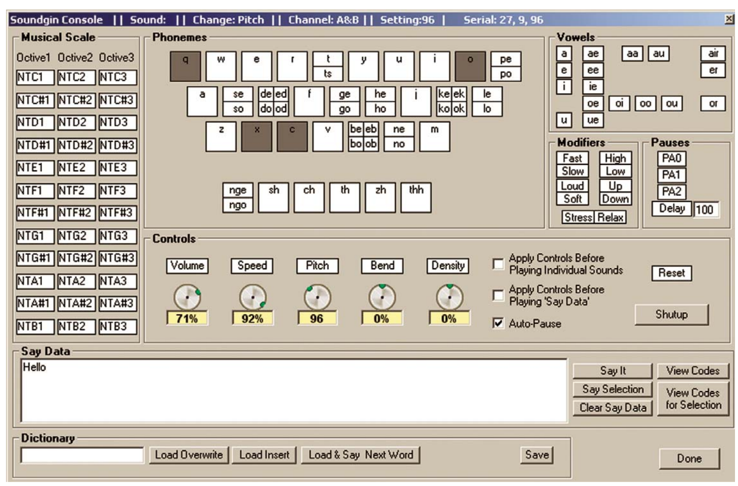


Figure 4. Use the Speech Phrase Editor window to produce intelligible speech output from the Soundgin.

RESOURCES

Soundgin integrated circuit (in SOIC and DIP packages), information and dealer contacts

Savage Innovations
www.soundgin.com

vibrations into air movement. With the Soundgin, that's a speaker of some kind. The rule is pretty simple: the better the speaker, the better the sound. You will find your Soundgin-based projects are greatly enhanced by using good quality speakers, preferably one already in an enclosure. The enclosure maximizes the air movement and increases volume, and it makes the sound richer. There are a number of small and inexpensive unpowered speakers you can use. Unless your design absolutely requires it because of size, weight, or both, avoid the use of small piezo speakers, or dynamic speakers that are not in an enclosure.

Though the Soundgin has its own amplifier, you may opt for a powered speaker. These have their own amplifier built in. I use an old pair of speakers designed to connect to a sound card in a PC. Soundgin is monaural only, so I use just one of the speakers, and leave the other one disconnected. The speaker uses a nine volt external power adapter, but I found that internally it requires only 5 VDC. This makes it easier to adapt the speaker for use on a mobile robot.

Learning More About Soundgin

I've only been able to scratch the surface of this remarkable workhorse. Savage Innovations provides a lengthy documentation file on understanding and using the Soundgin chip. It's in Adobe Acrobat format, and is available for free download on the Soundgin website. Also provided are schematics for the available prototyping board, as well as a rundown of the registers used in the chip.

Learning by example is perhaps the best way to master any new technology. The Soundgin Windows console download includes two sample files, named Sound1.sgs and Sound2.sgs. These are in Windows INI text format, and contain nothing more than register settings used to reproduce the sound. Try modifying these sound effects, or any of the presets, and saving them to a new sgs (Soundgin Sound) file. Open the resulting file in a text editor for review.

With all the options available with the Soundgin, make it a New Year's resolution to add some life to all your 2007 projects! **SV**

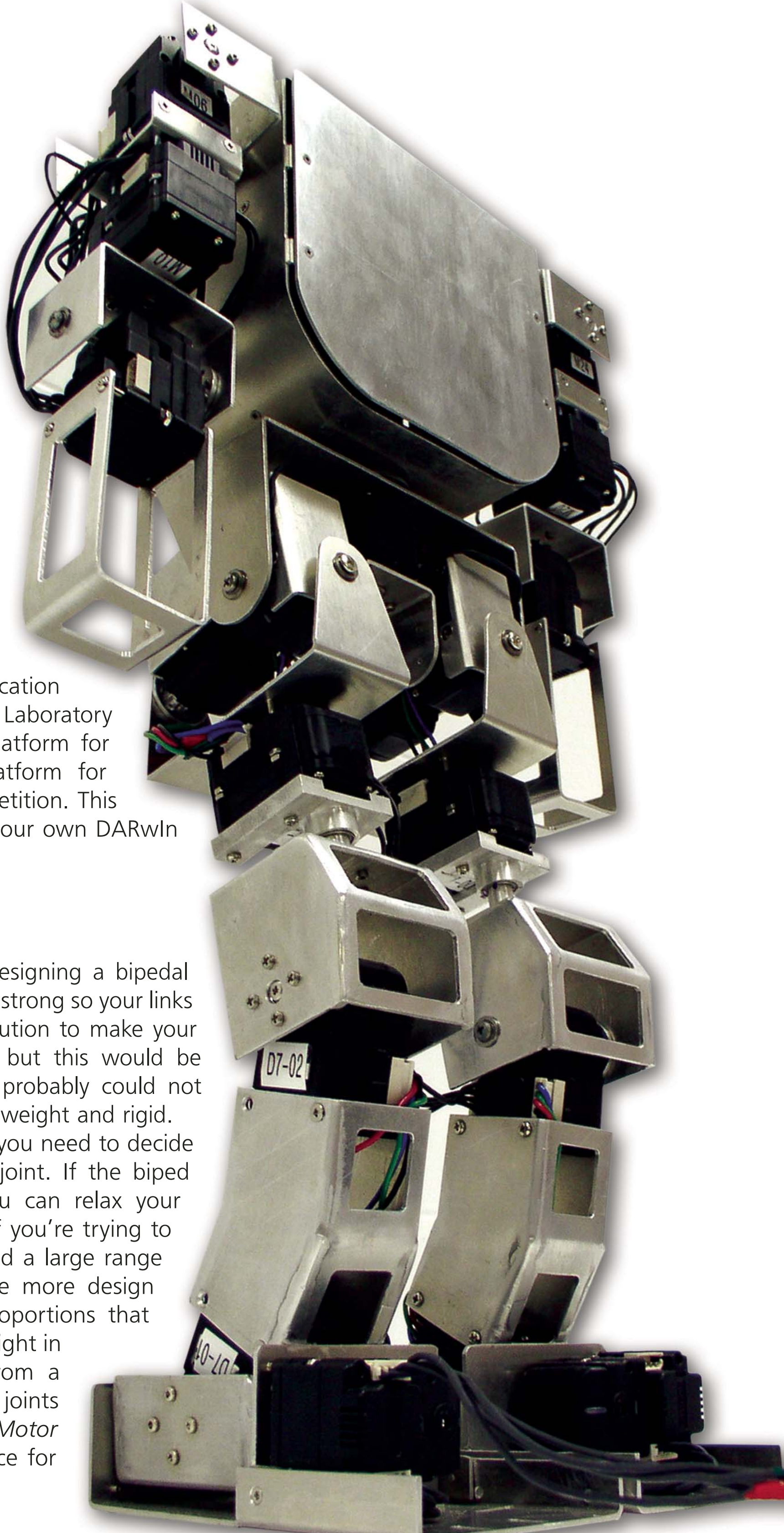
ABOUT THE AUTHOR

*Gordon McComb is the author of the best-selling Robot Builder's Bonanza, Robot Builder's Sourcebook, and Constructing Robot Bases — all from Tab/McGraw-Hill. In addition to writing books, he operates a small manufacturing company dedicated to low-cost amateur robotics (**www.budgetrobotics.com**). He can be reached at robots@robotoid.com.*

DARwIn

PART 2: Parts, Wires, and Motors

by: Karl Muecke, Patrick Cox, and Dennis Hong
RoMeLa (Robotics & Mechanisms Lab) at Virginia Tech;
www.me.vt.edu/romela

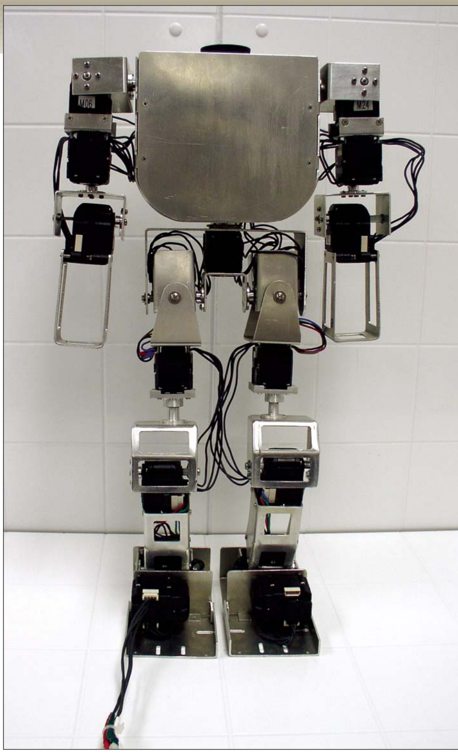


Last month, we introduced DARwIn (Dynamic Anthropomorphic Robot with Intelligence) — a humanoid robot capable of bipedal walking and performing human-like motions. This month, we will show you how we built DARwIn, including details on motors, wiring, parts, force sensors, and the fabrication process. Developed at the Robotics & Mechanisms Laboratory (RoMeLa) at Virginia Tech, DARwIn is a research platform for studying robot locomotion and also the base platform for Virginia Tech's first entry to the 2007 Robocup competition. This article should give you enough information to build your own DARwIn biped!

Design Considerations

There are a few items to keep in mind when designing a bipedal robot. You want to design the structure to be rigid and strong so your links and joints don't break or flex too much. An easy solution to make your robot rigid would be to design it all out of steel, but this would be a tremendous amount of weight that your motors probably could not handle. The goal is to design a robot that is both lightweight and rigid.

Depending on what you want your robot to do, you need to decide what kind of range of motion you need for each joint. If the biped is always walking and never has to stand up, you can relax your requirements for range of motion in certain areas. If you're trying to make your robot as human as possible, you will need a large range of motion in many of the joints. This leads to one more design consideration: human factors. There are certain proportions that dictate the distance between joints and the overall height in a human being. If you scale down your robot from a person, you need to make sure that all the links and joints are proportional. David Winter's *Biomechanics and Motor Control of Human Movement* is an excellent resource for human proportions.



SPECS	
CPU	PC/104 (Planned)
Height	600 mm
Mass	4 kg
Degree of Freedom	21
Power	2 Li-poly @ 7.4V
Sensors	Rate gyro, force sensors

< **FIGURE 1/TABLE 1.** Here is a photo and statistics table for DARwIn.

Select Your Motor

One of the biggest decisions you have to make when building your bipedal robot is what kind of actuators to use. We used Robtis' Dynamixel servo motors, model DX-117; and will upgrade to the RX-64 for the next version of DARwIn (Figures 2 and 3). The DX-117 has a maximum torque of 39 kg-cm and the RX-64 has a maximum torque of 64 kg-cm (Table 2). Both have built-in position and speed controllers. All user control is done with RS-485 serial communication.

v **TABLE 2.** Table of specifications for Robotis' Dynamixel DX-117 servo motor.

DYNAMIXEL DX-117	
Mass	66 g
Resolution	0.35°
Angle Range	300°
Voltage	12V-16V
Torque	29 kg-cm — 39 kg-cm

v **FIGURE 2.** RX-64 motor.

v **FIGURE 3.** DX-117 motor.

If you are on a tight budget, regular R/C servo motors are fairly inexpensive, but since you get what you pay for, the performance may be unacceptable. You can also use miniature DC motors, but you need to design your own position feedback controller or purchase one such as the "allmotion" boards, which can be expensive. Different actuators have different torque properties and mass properties; so if keeping your robot lightweight is a must, you may want to put the stronger, heavier motors in the joints that see the greatest load.

Kinematic Design

Once you pick your actuators, you need to decide on your kinematic structure — this is one of the most important things to do in the design process! You can make your own or use ours (Figure 4). Your kinematic model will determine how you design your joints and how you attach the actuators.

Where to Put Stuff

Next, decide on a location for your extra hardware: CPU, batteries, sensors, etc. We decided to house everything except for the force sensors in the chest for simplicity (Figure 5). One issue with placing everything this high is it creates a larger torque on the motors in the legs. It is possible to house hardware in other cavities in the robot, some such cavities may be in the leg or foot (depending on the design). Though the batteries are in the chest for the current design, next year, we plan on placing the batteries in the feet of the robot. This placement will make room for our new PC/104 board computer, IEEE 1394 PC board camera, rate gyro, DC-DC converter, and protection circuits.

Let's Be Honest: Materials and Machining

Finally, before you start designing the links and joints, determine what materials you can afford to buy and what machining tools you have access to (milling machines, welding equipment, etc.). This will dictate how ornate (or how simple) your robot's design can be. If you have access to a four-axis CNC mill and large amounts of bar stock aluminum, you have much more freedom in your design than if you only use sheet aluminum.

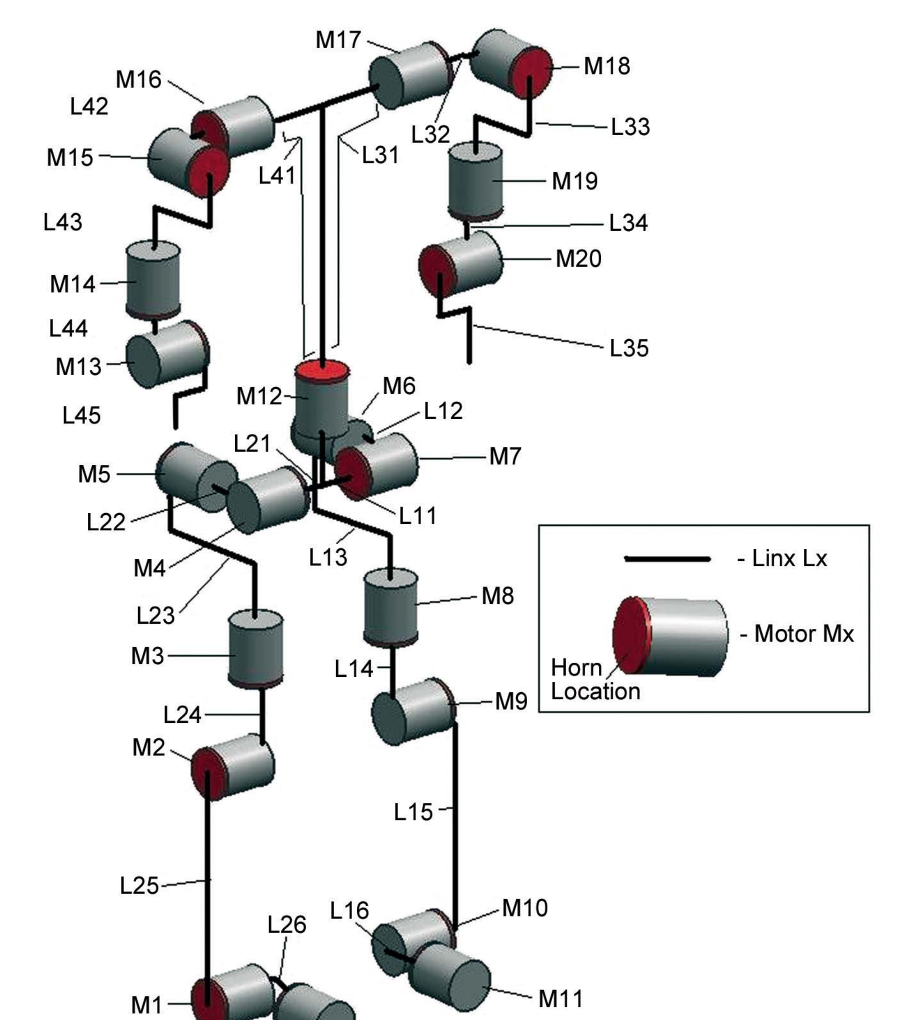
For DARwIn, time constraints and availability led us to use sheet aluminum for almost the entire structure. With some experience under our belt, we're designing and building DARwIn 2.0 in half the time and using a four-axis CNC mill to create most of our parts. Figure 6 shows a CAD drawing of our new hip design. We recommend using aluminum for the structure of the robot because of its price, weight, strength, and the ease of machining it. If you have the funds, using rapid prototyping would be an excellent option.

Design

Now that you've nailed down the essentials, you're ready to start designing the links and joints of your robot. You can get as creative as your imagination lets you, but if you need a starting point, here are some tips and examples from DARwIn.

Start Simple — The Elbow

A good place to start design is coming up with a simple elbow/knee



> FIGURE 4. Kinematic diagram of DARwIn.

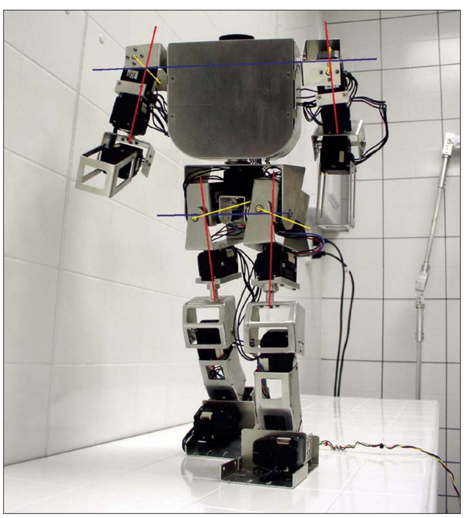
joint. They are the simplest joints in the entire robot since they are only one degree of freedom (DOF). Figure 7 shows a simple way of designing a bracket for the elbow joint.

Kinematically Spherical Joint Design — The Easy One

In our opinion, the next "easy" joint to design is the kinematically spherical joint at the shoulder. You have a lot of freedom when placing the motors and mounting them for the shoulders since the shoulders are naturally broad. In order to create a kinematically spherical joint, all three axes of rotation must intersect at the same point (Figure 8). This is fairly simple in concept, but getting configuration and dimensions correct for the design can be tricky. Figure 7 shows how we used a series of brackets to create kinematically spherical joints for the shoulders.

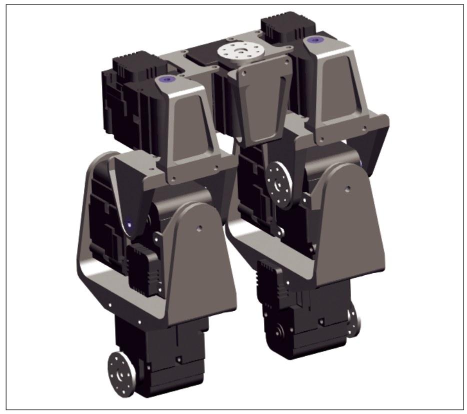
A Step Up — The Ankle

If you are comfortable with designing the arms and knees, try designing



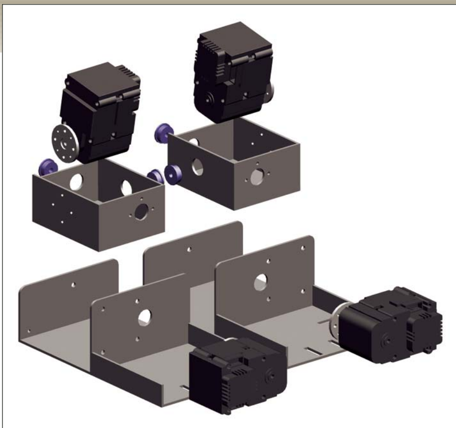
> FIGURE 8. Photo with lines showing spherical joints on DARwIn.

^ FIGURE 5. Exploded view of chest.



^ FIGURE 6. New hip design that will use a CNC mill.

^ FIGURE 7. Exploded view of arm.



< **FIGURE 9.** Exploded view of the feet.

the ankle joint next. A deceptively tricky joint, the ankle's design challenge comes in allowing for a large range of motion while keeping a lower pivot. The ankle from our design has two DOF. Both axes of rotation intersect to more closely imitate a human's ankle and make the kinematic and dynamic equations simpler for control of the robot. Figure 9 shows how we made our ankle and feet. Notice how it can be challenging to figure out how to support your links. We used ball bearings in most cases to reduce the bending moment where the servo motor attaches to the link.

The Hard Part — Hips

Even if you can handle the design of the rest of the body, the hips can stop you in your tracks. Creating kinematically spherical joints with human proportions (optional), good range of motion, and good structural stability is very difficult — especially if you have a waist motor in the area. Figure 10 shows how we made our hip joints. Two of the motors are joined together in the hip to reduce the space requirements of each motor. If you have trouble fitting all of the motors together in the hips, moving the waist motor into the chest can free up some space.

Links

Once you design the joints, you're about halfway done. Next, decide how you want to join each link together. There is a careful design balance between structural rigidity and range of motion. You'll notice that as you start connecting joints together, your links start to interfere with each other and limit your range of motion. Figure 11 shows how we designed the link going from the knee to the ankle. We put a bend in the link to add structural rigidity and to increase the range of motion. You may also notice that the rendered drawings are a little different from our final design. In our final design, we removed material from the knees, shins, and forearms to decrease weight.

This is also a good point in time to consider how you want to mount the motors to your joints and links. What kind of screws and nuts do you need? What kind of holes (threaded or not) and hole placement do you need? Deciding on a standard screw pitch and size greatly helps in assembly.

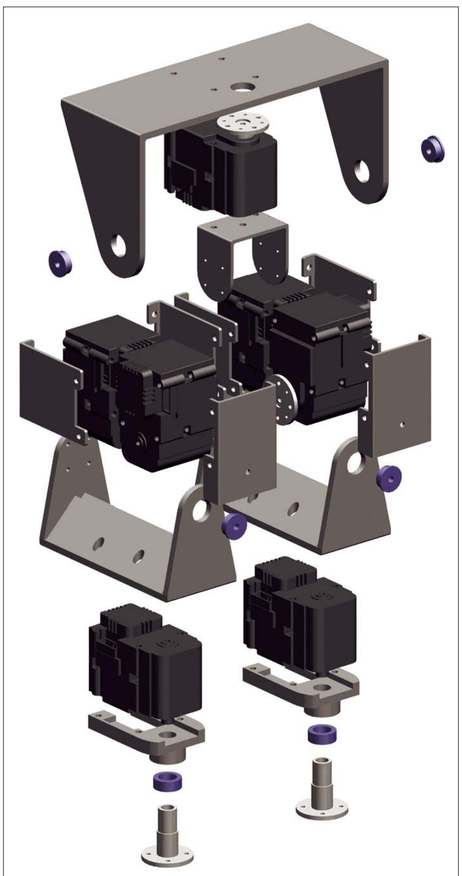
Link design is also an appropriate time to start considering wiring issues. How are you going to get power/ communications to your actuators? How are you going to connect your sensors, CPU, and power source?

We created many of our own wiring connections since we wanted to reduce any excess weight or volume. Robotis provides male and female connectors so you can create wires to your desired length for wiring up your servo motors (Figure 12). Figure 13 shows a picture of one of our custom-made wiring connections.

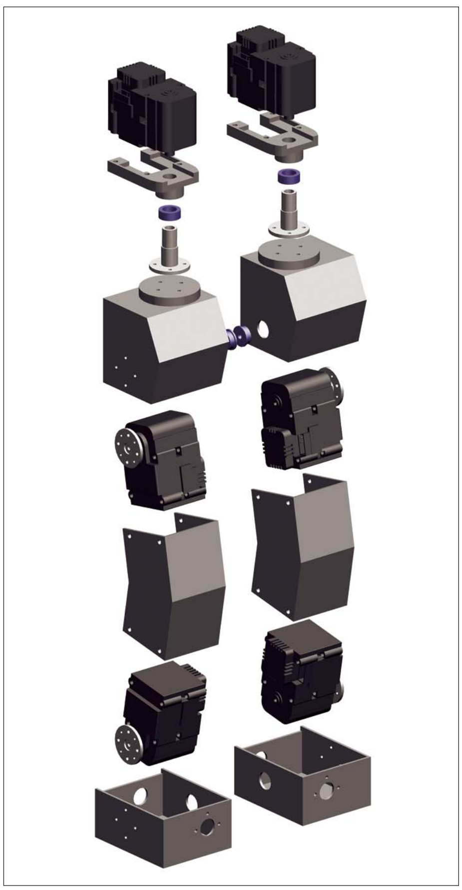
Force Sensors

Obviously an optional feature for your robot — but important enough for consideration during physical design — force sensors can be used to give information to your CPU to help stabilize the walking motion of the robot. The number and placement of the force sensors can be tricky. We recommend using four for each foot: one placed at each corner. Once you place the force

v **FIGURE 10.** Exploded view of the hip.

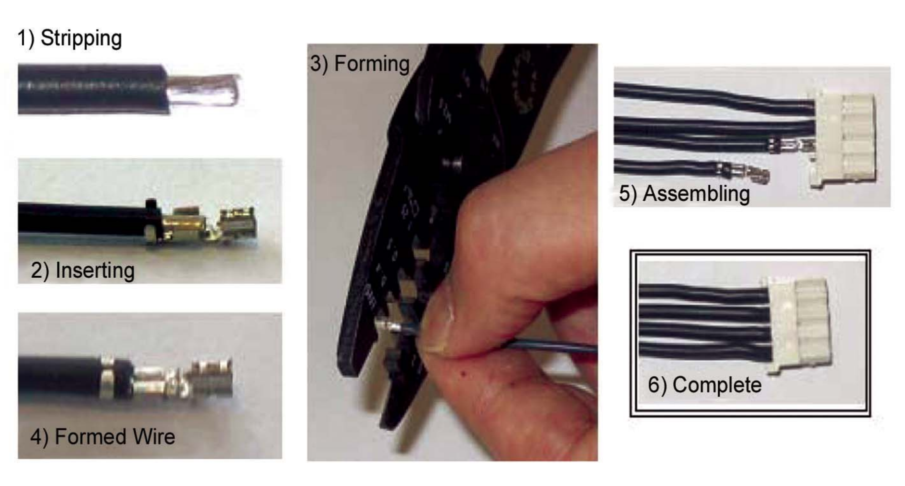


v **FIGURE 11.** Exploded view of the legs.

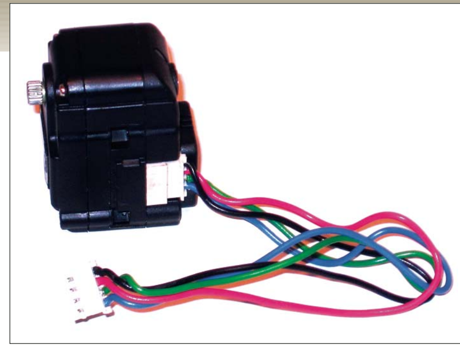


Thank Yous

Thanks to the 2005-2006 senior design team members that built DARwIn 1.0 (Patrick Cox, Joo Gil, Chris Greenway, Jeff Kanetzky, Karl Muecke, Patrick Mulliken, Raghav Sampath, Daniel Zokaites) and to our advisor Prof. Dennis Hong.



^ **FIGURE 12.** Photo of how to make custom wiring (**www.robotis.com**).



^ **FIGURE 13.** Photo of custom wiring.

^ **FIGURE 14.** Photo of FlexiForce force sensor (**www.tekscan.com**).

sensors, you need a way to connect to them since they are on the bottom of the foot. We cut out slots for each force sensor to slide through.

There are many different types of force sensors you can use. Tekscan makes FlexiForce sensors which are very thin piezoresistive force sensors (Figure 14; **www.tekscan.com**). When using a resistor to measure force, you usually need to construct a calibration circuit to get the greatest sensitivity and range of load detection. After calibration, you also need an Analog-to-Digital converter to convert the signal to a digital signal that the computer can read.

Next Month ...

We showed you the physical aspects of DARwIn and how you could make your own. Next month, the last part of this series will detail some of the software used to control DARwIn, as well as a peek of next year's design. **SV**

Hacking ROGER

FIGURE 1. Roger Robot just out of the box before dissection begins.

by Monty Reed

In the basement of North Seattle Community College (in Seattle, Washington), there is a Robotics Club. Members meet in a room called the "TECHTRONS LAB" that resembles a weird science laboratory from a sci-fi movie. "It is a little cramped, but it is ours!" says one of the students as they squeeze into the room to work on robotics projects.

The room has been in the control of students for three decades, as far as any of the old timers who work in facilities can remember. It's great for the students because there is freedom to work on projects without being graded. For many students, the Lab is a refuge allowing a few hours to relax and work on robots or electronics projects, or just have a cup of coffee. For others, it is a place to ask for help on an electronics project, math, or computer programming. All students are welcome. Occasionally, students or visitors from the community drop in just to see what the others are up to.

We were looking for an affordable, simple project anyone could do. North Seattle Robotics Group (The official Robotics Club at North Seattle Community College) is always looking for inexpensive toys to hack into robot projects. Roger Robot (Figure 1) from Big Lots (**www.biglots.com**) was $19.95. It is currently available from **www.eBay.com** or other resellers. The price is right for students and just about anyone who wants to learn about robotics and have fun doing it. "When considering buying a toy to hack, look for one with a tethered or wired controller. This way you are not paying for the wireless capability," advises Doug Bell, a Volunteer Community Advisor to North Seattle Community College Robotics Club since the Fall of 2001. He comes to the lab twice a week to volunteer, helping students with their projects. Many of these projects just could not be done without his advice and help.

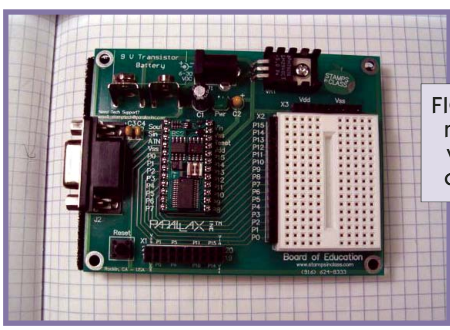


FIGURE 2. Parallax microcontroller with the Board of Education™.

We have several microcontrollers in the lab to choose from and plenty of support from advisors, faculty, and staff. The Parallax BASIC Stamp™ microcontroller (see Figure 2) is used in classes at NSCC and is a popular choice among students. There is a lot of tech support, documentation, and information available. Most of my experience has been with the BASIC Stamp, so that was my choice to use.

For this project, I wanted to do as little hacking as possible to make the robot controlled by a microcontroller and I wanted to also be able to continue to use it as a tethered remote control, so I planned for a 25-pin connector to be placed in line with the tether from the controller. This way, the microcontroller or the handheld remote could be plugged in. To make this more sophisticated, you could add a switch instead of plugging and unplugging the remote or the microcontroller. Also for this project, we will not install a switch.

NOTE: All codes/programs are available on the *SERVO* website at **www.servomagazine.com**.

Another goal for this project was to learn about subroutines and to get some experience using them with the BASIC Stamp. This became part of the pseudocode (programming outline) for the programs that would run Roger Robot and give him new life.

You can do this project even if you do not have much experience.

First things first — check your parts and be sure you have everything you need for the project before you begin (see Parts List 1). Nothing is more frustrating than being at a crucial point in your project at 2:00 am and realizing you are missing a part and cannot go on. You have consumed all the coffee and pizza needed to finish the project, but a single component keeps you from success. Do not let this happen to you or your crew. Be prepared by checking your parts before you begin.

Be sure you also have all the tools you need before you begin (see Figure 3 and Parts List 2).

It's a good idea to have a clean workbench or workspace when you start a project like this. A friend of mine has a roll of butcher paper mounted on one end of his workbench in a dispenser. It is the width of his workbench and whenever the paper on the work bench gets dirty or he begins a new project he clears the bench, pulls a new sheet of paper the length of the bench, and tapes it down. The clean white paper makes it easy to find small parts.

To help you in reassembly, you may want to grab your digital camera and take some pictures as you build, just to be sure you know were everything goes.

Next, take Roger Robot out of the package, follow the instructions to put in his batteries, and power him up. Become familiar with his controls. Roger Robot has tank-style steering, which is very desirable for robotics. A robot can have tank steering without having tank treads — Roger Robot happens to come with treads. Tank steering is when a robot has a left and right propulsion device that can be controlled independently or in concert. This allows for a full range of steering controls. All Ahead Full uses both drives in a forward motion. Reverse is driving them both backwards. Turn Left can be accomplished by driving the right forward or the left back. For a Spin Left the left side drives reverse and the right side drives forward. This type of steering is much easier for a robot to use and easier for us, as humans, to program the robots movements. The treads are desirable because it is easier for the robot to navigate rugged terrain.


FIGURE 3. Tools and the robot layed out on a clean workbench with the remote opened up.

Become familiar with Roger Robot's controls. Drive him forward and back, make him spin in place, and operate his arms. Roger Robot's arms open and close for grasping things and the arms go up and down for picking things up and setting them down. For simplicity, my goal was to pick up a cup, back up, turn around, and drive away. The next step in future programming could include picking up and placing the cup in a specific location, moving other items, etc. Roger Robot also has "light-up" eyes. We decided to not use this function in this project because we wanted to save the batteries. We will explore this in a future project. If you like what you see in this project and want more info, go to **www.theyshallwalk.org** and watch the pages of *SERVO Magazine* for future projects.


FIGURE 4. Remote with cover.

Open Controller to Trace Wires

Remove the screws that hold the controller together. Place them into a secure tray or zip lock bag to ensure you have all the pieces so you can put Roger Robot back together again. I strongly recommend the zip lock bag approach, especially if you have multiple projects like I do. Mark the bag with the name of your project and the date you began it, as well as the contents of the bag. Usually in a "Disassembly Project," I will have a series of snack, sandwich, and gallon size bags placed inside of each other. The zip lock bag approach is also very important if you share a workbench space with others. Boxes or trays can tip over and the contents become mixed or lost.

After the controller is open, examine the wires and where the wires are connected. The idea is to determine what each wire does by looking at the color that goes to each individual switch and what happens when that switch is in different positions. Look at the front cover (see Figure 4) and compare it to the inside wire locations (see Figure 5a).

To start with, look for the black and red wires. These should be ground and power, respectively, but there is no guarantee — especially when it comes to low-cost toys.

We discovered in our version of Roger Robot that there was a red wire connected to the majority of the switches. After we verified that the red wire was power, we traced the colored wire connected to the other part of the switch

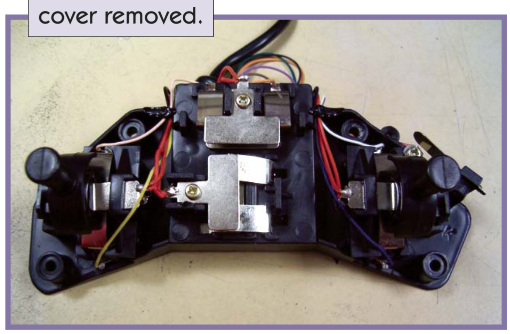


FIGURE 5a. Remote with cover removed.

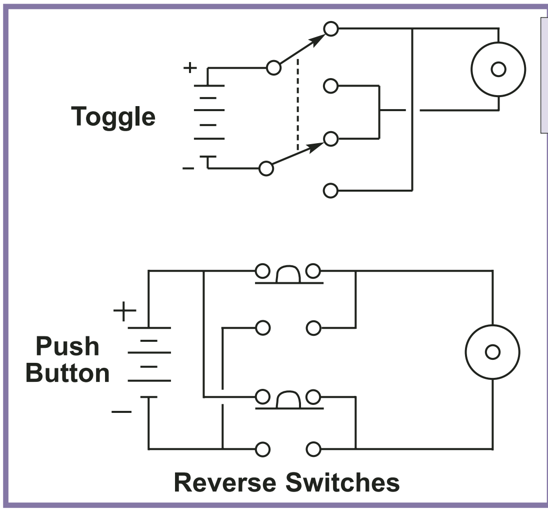


FIGURE 5b. Drawing of Reverse switch; top toggle type reverse switch, (bottom) push button type reverse switch.

to determine what component each switch controlled. Our results are listed in Figure 6.

This version of the robot has several reverse switches (see Figure 5b) in it. These are the push button type. When they are not being pushed, the wires are connected in one way and when they are pushed, the wires are connected in another way.

Tether Disassembly

The next few steps will take some forethought and planning.

FIGURE 6. Tether Wire Results Chart.

Red = 9 VDC power **Black = Ground**

Gray	+	Red	= Lights up Robot's Eyes
Brown	+	Black +	
Purple	+	Red	= Arms Down
Purple	+	Black +	
Brown	+	Red	= Arms Up
Green	+	Black +	
Orange	+	Red	= Arms In
Orange	+	Black +	
Green	+	Red	= Arms Out
White	+	Black +	
Blue	+	Red	= Right Tracks Forward
Blue	+	Black +	
White	+	Red	= Right Tracks Backward
Pink	+	Black +	
Yellow	+	Red	= Left Tracks Forward
Yellow	+	Black +	
Pink	+	Red	= Left Tracks Backward

Read through these steps and think about how each component interrelates before proceeding to cut, bend, attach, mutilate, or glue!

Before you cut any wire, take some time to think about wire placement. At the ME Lab at the University of Washington, you will be constantly reminded by Russ Noe to "measure twice and cut once." This applies to anything you are cutting. It's much easier to shorten something than to lengthen it.

Measure the placement of the project box and decide where you will put it. I put mine on the back and to the left a little. The project box makes it look like the bot has a little backpack on (see Figure 7).

We will be mounting a 25-pin connector on the robot. You have a parts option of purchasing 25-pin connectors or hacking a 25-pin cable. Either way, the solderable side of the connector will be attached to the original wires inside the robot that lead to the various components. The pin side will be connected either to the original controller or the cable coming out of the project box that is connected to the microcontroller. Mine is mounted on the back and to the right a little. However, it can be mounted anywhere you prefer. Make sure that you choose the location for your connector at this step because you need to connect the robot wires (noted in the following steps) before you attach it to the robot frame.

Determine where to cut the access holes in the project box for the cable coming from the microcontroller; keep in mind this cable needs to reach your 25-pin connector.

If you are not sure how to decide for yourself, you can just cut the tether in the same way we did. Measure from Roger Robot 12 inches and make the cut with wire cutters.

For a cheap way to get the connectors, check your local thrift store for a standard parallel extension cable. The common use is as an extension cable for printers and other peripherals. I have seen them priced from \$1-\$3. Look for the cable that has a female and a male connector. Disassemble the cable for the parts.

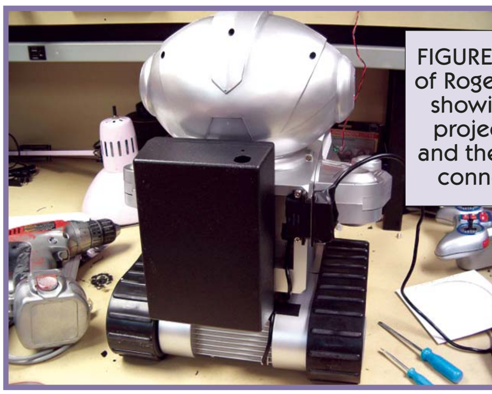


FIGURE 7. Back of Roger Robot showing the project box and the 25-pin connector.

You may choose to mount the 25-pin connector in any other location. Decide where you are going to mount the connector before cutting the tether.

Slice open the insulation carefully for the robot side of the tether with an Exacto™ knife (see Figure 8) or razor blade. Be careful not to cut or damage the insulation on the smaller wires inside. Remove enough of the outer insulation so that all of the wires can be seen. Usually two inches is enough, if you prefer, you can cut less.

Using wire strippers,

FIGURE 8. Tether being sliced open with an Exacto™ knife.

strip a quarter inch of the insulation off of each of the wires from the tether (see Figure 9). Repeat opening and stripping the tether for its controller side.

Solder the wires from the robot tether to the male 25-pin connector and the wires from the controller tether to the female 25-pin connector. When soldering a lot of wires to a connector, it is handy to have another person hold the wires in place while you solder so you can move quickly from wire to wire. If you do not have a helper available, I recommend the Helping Hands soldering tool from Harbor Freight. It has a weighted base and a couple of alligator clips to hold the wire being soldered and the connector you are soldering to. Solder the wires to the 25-pin connector, as shown in Figure 10.

As long as the wire color codes you found in your robot's controller match the table in Figure 6, the order in Figure 11 should be followed in soldering the wires to the connectors. If your wire color codes are different, you will need to create your own chart.

After you have completed soldering your male connector, mount it to the spot you previously selected (see Figure 12).

Plug in the controller tether and test to see that all of the switches do all of the same functions they did before you cut the tether. If there are problems, troubleshoot now. Refer to the chart of functions and wires (Figures 6 and 11) to see where you may have gone wrong. When this test is successful, continue.

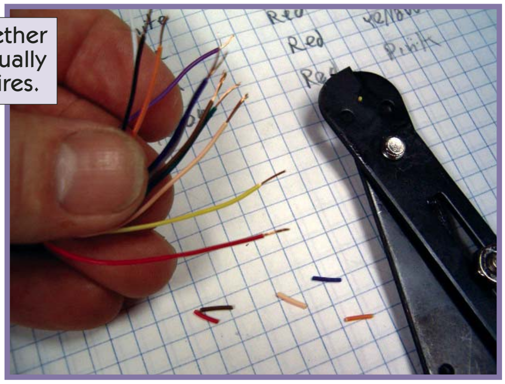


FIGURE 9. Tether with individually stripped wires.

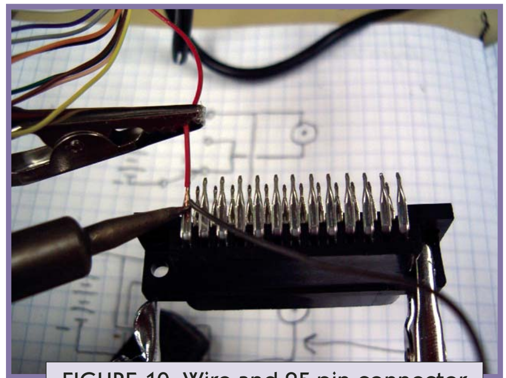


FIGURE 10. Wire and 25-pin connector being held by the Helping Hands tool while the solder and soldering iron is being held by the author.

Mounting the Project Box

In this phase, you will attach the project box on the back of the robot. It will look like a backpack. Four screws will be enough to hold it securely in place. You may choose to use Super Glue instead. I like to use screws for this so that I can remove the box later if I choose to for aesthetics, repairs, or modifications. If your Roger Robot has a bracket for the tethered remote on the back, remove it by loosening the screws. Later, we will install it on the back of the project box so you can still use the remote mounting bracket.

When drilling pilot holes for self-tapping screws in plastic, I like to test the size of the drill bit and the screws on scrap plastic. Find a local plastics supplier and purchase some scraps by the pound. These will come in handy for all kinds of other projects, as well.

Hold the box on the back of Roger Robot and see where the box touches his back. Pick a spot near each of the corners inside the box to place your screws. If you have a box like ours, you may choose to place your screws where we did. Measure one-quarter inch from the corner and mark the hole. You can do this with a wite out™ type marking pen if you are using a black box. You can use a Sharpie marker if your box is any other color. Alternatively, you could place masking tape over the area near the corner, then measure for the cut.

I prefer to drill through the box into the robot at the same time. Some people prefer to measure and drill the hole in the box first, then hold the box on the robot and mark the robot through the holes in the box, then drill the holes in the robot. It is best to tape the box in place on the robot and then drill your holes.

After the holes have been drilled, drive the self-tapping screws into the robot through the project box. Use the appropriate screwdriver (philips or standard). Mount the remote control bracket on the back of the project box (see Figure 20). You can use the screws that came with it or some of the self-tapping screws.

Placing and Cutting Access Holes

Your project box may come with pre-cut holes for the cables. If this is the case, use them.

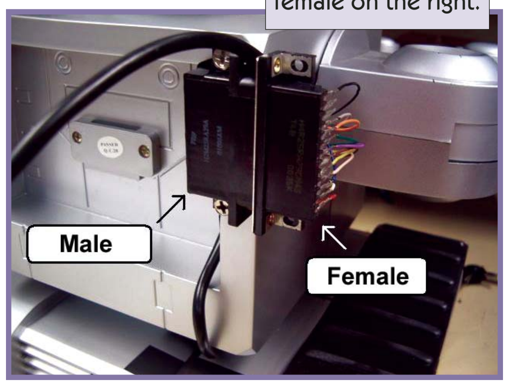


FIGURE 12. 25-Pin connector in place showing the male on the left and the female on the right.

Pin #	Wire Color	Action/Purpose
1	Red	(9 VDC)
2	White	(Right Tracks Backward)
3	Blue	(Right Tracks Forward)
4	Pink	(Left Tracks Backward)
5	Yellow	(Left Tracks Forward)
6	Brown	(Arms Up)
7	Purple	(Arms Down)
8	Green	(Arms Out)
9	Orange	(Arms In)
10	Gray	(The first step of the project)
11	empty	
12	empty	
13	Black	(Ground)
14-25	Available for future development.	

FIGURE 11. 25-Pin Connector Wiring Chart.

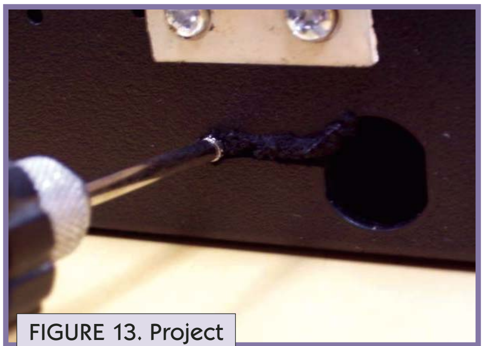


FIGURE 13. Project box with Dremel™ cutting the plastic like a hot knife through butter.

If you do not have precut holes or you want to run cables in a different way, follow the hole cutting procedure outlined here. Measure and mark the hole on the bottom of the box. Be sure to make the cut big enough for the connector to pass through.

You will want a hole cut for the ribbon cable, DB9 connector, and two switches. I decided to cut the hole for the ribbon cable on the bottom of the box, the hole for the DB9 on the side, and one on the top and the other on the side for the switches.

I used a standard cabinet hinge for the project box. I like to use a hinge because I know I will return to the robot to hack and hack again. If you are planning on very little modifications, you can use the standard screws to close the box and leave it that way.

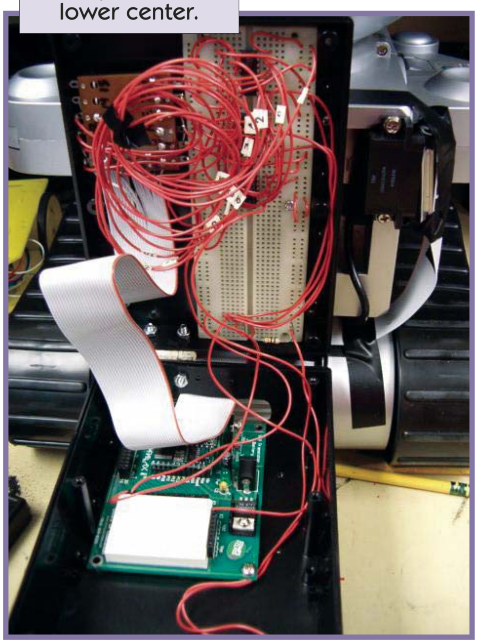


FIGURE 15. Back of Roger showing the terminal block on the upper left, breadboard upper right, and BOE lower center.

The best way to cut holes in plastic that I have found is a Dremel™ type hand tool with a metal striated bit. With the bit spinning at high speed, the friction heats the plastic up and it cuts like a hot knife through butter (see Figures 13 and 14).

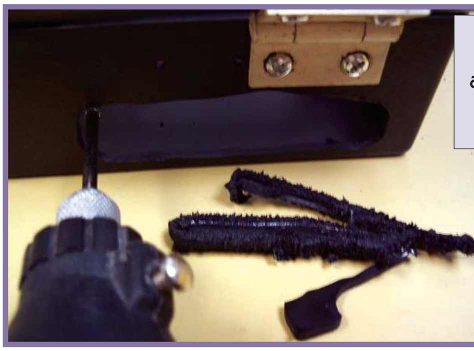


FIGURE 14. Project box after cutting large hole for cable.

Mount the BASIC Stamp and Parallax Board of Education

Place the Board of Education (BOE; see Figure 2) or your microcontroller board of choice into the project box. Make sure it will fit before you start drilling holes. It is best to mark the holes inside the project box so you know were to drill before you start. Doublecheck to be sure all of the holes match up, then start drilling. Use a bit that is smaller than the self-taping screws so the screws will have something to grab onto when you tighten them. Place the stand-off spacers on the holes, place the BOE on the spacers, and drive the screws through the BOE and spacer to secure it to the inside of the project box.

Inside the box, I mounted a standard breadboard on the robot's right side and a terminal block on the robot's left. I used self-tapping screws and some hot glue to hold them both in place. The BOE was mounted inside the hinged lid opposite of the robot's back (see Figure 15).

Breadboard Layout

Place both of the L293D motor driver chips (quadruple half H drivers). Place the 74HCT164 shift register chip.

Starting at the top of the breadboard, place the first L293D centered on the breadboard. The top of the L293D (pin 1) should be in the fourth row, and be sure the other chips have three holes (rows) between them. Place the 74HCT164 shift register last (see Figure 16). Follow the schematic and photos in Figures 15-17 for details of component placement.

Terminal Block and Breadboard Layout

Pick a terminal block that you like and be sure it will fit inside of your project box. In Figure 18, I used the one in the upper right hand corner. Number the terminal block 1-15. Solder the ribbon cable wires to the terminal block according to the Cable Pin Connection Chart in Figure 11. On the other side of the terminal block attach the lead wires for the breadboard in the order in Figure 19. Use wire markers or masking tape and a pen to number the wires near the end that plugs into the breadboard. My breadboard came with two columns, then a space, five columns, a gap, five more columns, a space, and two more columns. I used the first two for the Ground buss (G1 and G2). For this chart, "G" stands for ground and on the breadboard it is the first and second column. I numbered the next 10 columns 1-10. The last two columns are numbered P1 and

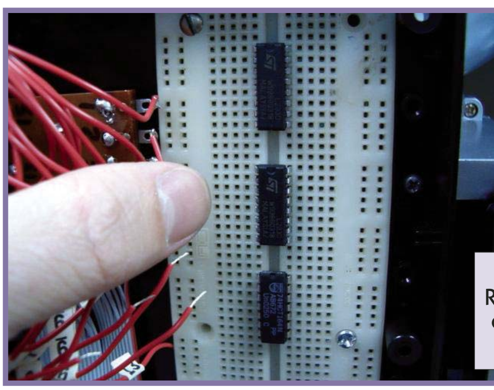


FIGURE 16. Breadboard inside of Roger Robot's backpack. Two L293D chips at the top and the 74HCT164 shift register at the bottom.

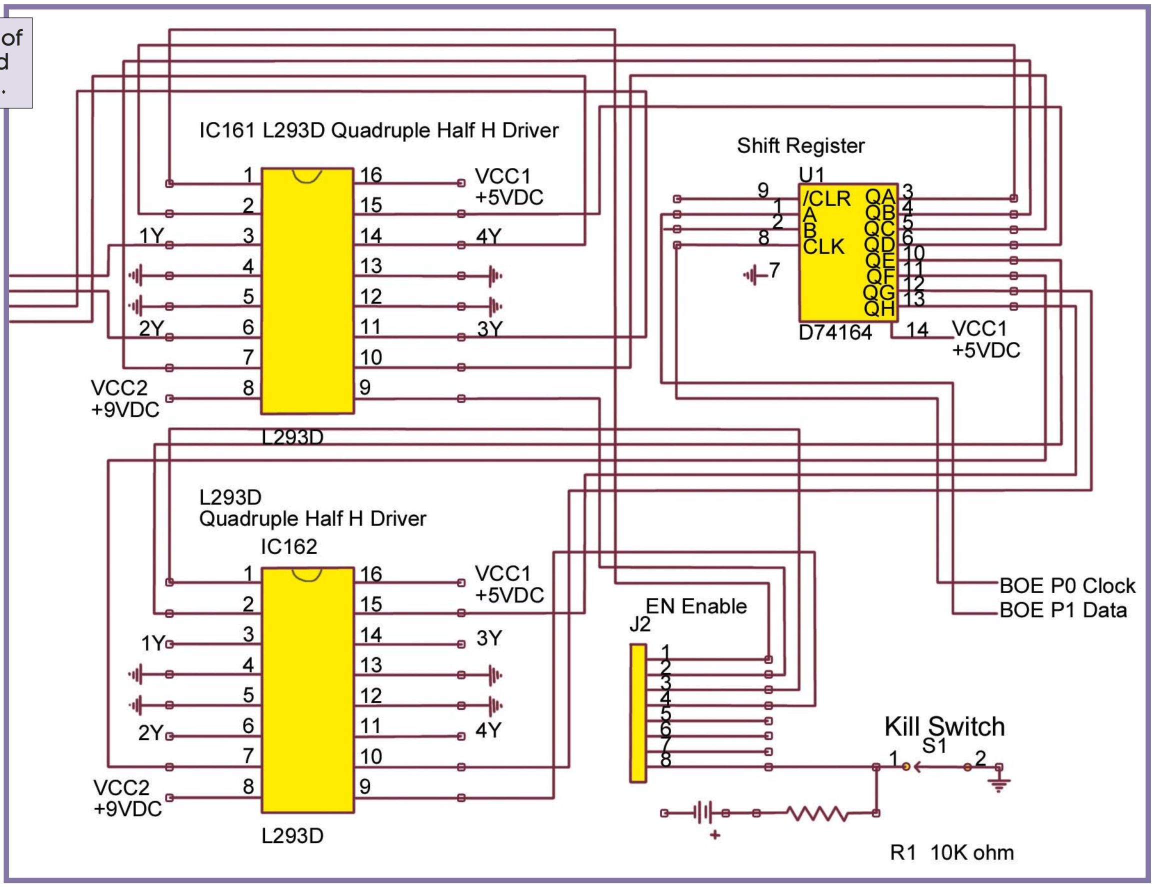


FIGURE 17. Schematic of the shift registers and Quad Half H Bridges.

P2. "P" stands for power. These last two columns are the power buss. Note: Some breadboards need a lead wire to connect the columns at the mid point and some do not. It is always best to check it with an ohmmeter to be sure whether or not it is connected throughout. The breadboard I used for this project does not connect all the way through so I had to add jumper lead wires from row G25 to row G26 in columns 1 and 2 — the first two columns. I added jumpers on P25 and P26, as well.

Cabinet Hinge

Mount the cabinet hinge on the bottom of the project box (see Figure 14) if you have a project box that does not come with hinges. Close the box and place the hinge centered between the top and the bottom. Mark the holes and then drill making sure the bit is small enough for the self-tapping screws to hold the hinge in place.

Kill Switch and Power Switch

I mounted a kill switch on top of the project box (see Figure 20). I like to use a toggle switch with a one-inch or longer toggle and I feel it is important to put it on top of the robot so if something goes wrong, it is easy to "kill" the program or power. Place the switch so that flipping it backwards puts it in the kill position. This way, when the robot is trying to run away, you can hit the switch and shut down the system. Hold the switch up to the box where you want to mount it and look to be sure it will fit and that you can get the wires connected. Use the Dremel to cut a hole for the switch in the top of the box as described earlier. After cutting the hole, place the switch in it and tighten the mounting nut.

The kill switch will be wired between ground and the "enable resistor" (EN on the schematic). The EN is connected to the five-volt DC power supply buss. On the schematic, there are connections from the L293Ds marked "EN." These will all be wired to row 58, column 1-4. In the "Go" position, the H-bridges can drive the motors. In the kill position, they cannot.

Repeat this procedure for the power switch on the side. You can mount it in a different location but be sure it

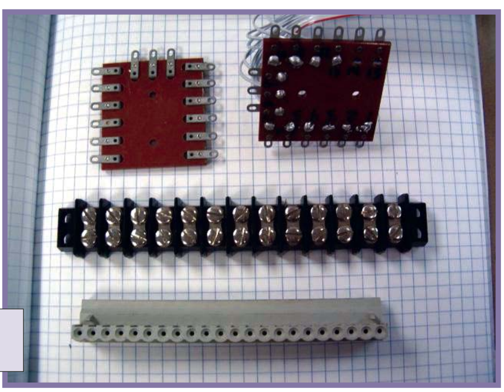


FIGURE 18. Terminal blocks.

FIGURE 19. Terminal block lead wires chart.

Terminal Block -	Breadboard	
From 1	to Row 1	Column 6
From 2	to Row 6	Column 3
From 3	to Row 9	Column 4
From 4	to Row 9	Column 7
From 5	to Row 6	Column 7
From 6	to Row 17	Column 3
From 7	to Row 20	Column 3
From 8	to Row 20	Column 10
From 9	to Row 17	Column 10
From 10	to Row 40	Column 1 (Use later)
From 11	to Row G1	Column G3
From 12	to Row G1	Column G2
From 13	to Row G1	Column G1
From 14	to Row	Column
From 15	to Row	Column

FIGURE 20. Kill switch on project box.


FIGURE 21. Robot grasping the cup and lifting it.

is not near the kill switch. The power switch is in line with the main power from the battery box in the base of the robot.

The power switch will be wired between the incoming power wire from the terminal block (wire number 1) and the power buss. From the power switch, one wire will plug into row 1, column 8 and the other will plug into row P1, column P2.

Connect the Microcontroller

Be sure the switches are in the kill and off positions. Disconnect the wired remote at the 25-pin connector and plug in the microcontroller ribbon cable. Using nine-inch lead wires, make a connection from "P0" on the BOE to row 32, column 8, from P1 on the BOE to row 26, column 2. Connect Vdd from the BOE to the power buss row P1 column P41 (see Figure 15).

Parts List 1

Qty.	Item	Supplier
1	Roger Robot	Big Lots — **www.biglotswholesale.com**
1	BASIC Stamp II	Parallax — **www.parallax.com**
1	Board of Education	Parallax — **www.parallax.com**
2	L293D motor driver chip	Texas Instruments — **www.ti.com**
1	74HCT164 shift register chip	**www.semiconductors.philips.com**
1	Breadboard	Electronics Goldmine — **www.goldmine-elec.com**
1	Jumper wire assortment	All Electronics — **www.allelectronics.com**
1	Ribbon cable 25-conductor 18 inches	All Electronics — **www.allelectronics.com**
2	Toggle switch CAT# MTS-5	All Electronics — **www.allelectronics.com**
1	22 position terminal strip CAT# TER-220	All Electronics — **www.allelectronics.com**
2	Resistor 10K ohm	RadioShack — **www.radioshack.com**
1	Spool of 28-gauge pre-tinned insulated wire	RadioShack — **www.radioshack.com**
1	Project box 7x5x3" 270-1807	RadioShack — **www.radioshack.com**
1	Machine screws	RadioShack — **www.radioshack.com** #4-40. Pack of 42.#64-3011
1	Self-tapping screws	RadioShack — **www.radioshack.com**
8	Stand-off Spacers	RadioShack — **www.radioshack.com**
10	Ceramic capacitor	RadioShack — **www.radioshack.com**
	Catalog #: 272-134. If you are going to do more projects, we recommend you buy the 272-801 set of 100 disc capacitors for $4.95 instead of the ones above.	
1	Cabinet hinge	Home Depot — **www.homedepot.com**
	Additional items	**www.eBay.com**; **http://stores.eBay.com/They-Shall-Walk-Foundation**

(Be sure you can solder to these)

2	25-pin parallel extension cable with male and female connectors *or*
2	25-pin female connector DB25
1	25-pin male connector DB25

Note: *You can purchase the machine screws and self-tapping screws at Home Depot or Harbor Freight (**www.harborfreight.com**) if it is more convenient or a better deal.*

Project Box — We used a spare one that was sitting around the lab. The one from RadioShack has an aluminum lid and a plastic lid included, and has partial cutouts for both DB-9 or HD-15 computer connectors and DB-25 connector. This will work better and require less fabrication.

Write the Program or Copy and Paste From *SERVO* Website

Start by loading a standard blinking LED or "Hello World" program to be sure the BASIC Stamp is connected properly. These standard programs can be downloaded from the Parallax™ website.

Once you know all is well with the BASIC Stamp and your connections, I recommend you try loading the GoForward program. Both left and right tracks will move forward. Run it and then if all is well, load the "Pick Up Cup" program.

Since we have not added any sensors to Roger Robot yet, he is blind and he cannot find a cup and pick it up so you will need to experiment with the

Thank Yous

A special thank you to the NSCC faculty members who have been very helpful: Frank Jump, Lynda Wilkinson, Chris Saunders, Tracy Heinlein, and Deborah Bedford. Thank you to Doug and Cecelia for all you do.

location of the cup. Sensors will be added in a future article.

For the program "Pick Up Cup" to work, you will need a paper or plastic cup (plastic is better in case the robot crushes it by squeezing too hard). Place the cup 12 inches in front of the robot. Turn on the power and push the reset button so the program will start. Since I did not add any sensors, this robot is running blind and requires you to set up the environment in a way that the program will function correctly. Watch how the robot and the program work. If the cup is too far to the left or right, or too far forward or back, move it so the robot will be able to pick it up.

You can move the cup until you find the right distance for placement every time. With my robot on my floor, the cup placed 12 inches in front and centered works perfectly.

I then modified the program so the cup could be placed two feet directly in front of the robot.

When the program is launched, the robot moves forward while the arms go up and out. Then the robot stops, the arms go down around the cup, the arms close grasping the cup, the arms go up lifting the cup (see Figure 21), the robot turns 180 degrees and, moves forward taking the cup.

Have fun changing the program to change the directions Roger Robot will go and the amount of time he will run.

Troubleshooting

Capacitors and resistors have been included in the Parts List in case you have a problem with electronic noise from the low-cost parts of this robot. I wrote the program in a way that the noise should not bother the function of the robot. The symptom of the electronic noise shows up when you run a program that operates any of the onboard motors. When the program tells a motor to move one direction is when the problem may show its ugly head. Symptom: motor moves a little, stops, then moves in a random direction and moves for random amounts of time. Also, other motors may randomly move different directions and for different times. I will go into the placement of filter components in a future article. **SV**

Parts List 2

- Dremel
- Dremel bits
- Drill
- Drill bits
- Exacto or razor knife
- Electrical tape
- Helping Hands (to help with soldering)
- Multimeter
- Needle nose pliers
- Pliers
- Screwdriver philips and standard
- Soldering Iron
- Solder
- Vise
- Wire cutters
- Wire markers (or tape you can write on)
- Wire stripers
- Wite Out™ Marking Pen (or white paint or correction fluid)

ROBOT SAFETY

by Bryan Bergeron

My first robotic arm kit — a six DOF, all-metal model with high-speed, heavy-duty servos, and ample power supply — taught me not to be lulled into the apparent plug-and-play world of hobby robots.

When I applied power to the servos, the joints abruptly hyper-extended, flinging a hot soldering iron against a wall. Then, when I applied power to the servo controller, the uncalibrated servo chain contracted, digging the gripper into the work-bench. The unsecured arm flipped onto its side, knocking over a hot glue gun in the process. Fortunately, the gun and iron landed harmlessly on the shop floor and the arm was undamaged.

Robotics presents a real-word test of AI, computer vision, and sensor technologies. Moreover, because robotics involves physical, as well as computational challenges, the dangers associated with building and operating robots extend beyond keyboard-induced carpal tunnel syndrome. This is especially true with the current trend toward heavier, faster, and more complex robots. Following is a discussion of robot design, construction, and operating practices that can improve the survivability of your robots and minimize the safety risks to you and bystanders.

Design

Safety should be explicitly designed into a robot, as opposed to an add-on made in response to a injury or robot failure. Appropriate platform layout, the use of safety subroutines, as well as physical sensors, intuitive state indicators, energy management hardware and software, and kill switches all contribute to a comprehensive safety design.

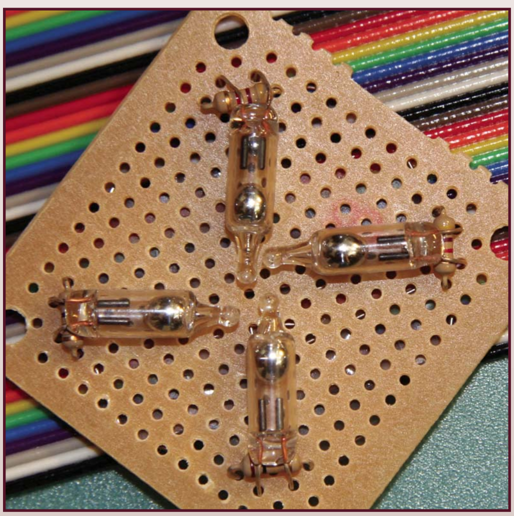


FIGURE 1. Physical tilt sensor based on mercury switches.

Platform Layout

In designing a robot platform layout, the primary goal is to provide a stable support for sensors and associated electronics within a structure capable of withstanding the mechanical stress of motion. In addition, the platform should be configured in a way that protects onboard electronics and minimizes damage to anything or anyone unfortunate enough to be hit by the robot body or effectors.

For example, in designing a crawler, consider using rubber cushions to provide hard limits on the range of leg motion so that the legs don't clash into the body while you experiment with various gaits. This safety precaution can limit damage to the robot body and servos, and minimize pinch points that can entangle your fingers.

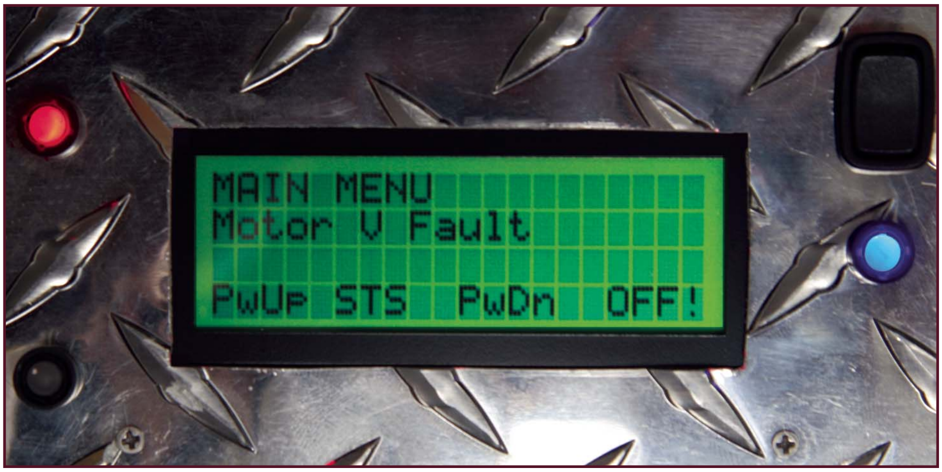
Similarly, avoid sharp edges and protruding corners on fast-moving wheeled robots over a few pounds, which have a way of colliding with furniture legs and shins. I have several robots based on the Traxxas e-Maxx truck chassis — a popular platform for robot development. Because the e-Maxx drive system is capable of speeds in excess of 30 mph, the potential for damage to both person and robot is significant.

If you're working with a high-speed platform, consider a roll bar and sturdy frame to protect electronics and a bumper system that will minimize damage to human and furniture legs. I've found a pair of calf-high boots handy when debugging high-speed robots capable of sudden and unexpected acceleration.

Safety Subroutines

Safety subroutines can be the easiest safety measures to implement – if you do your homework in selecting

FIGURE 2. An LCD screen and multi-state LEDs provide internal state information — in this case, low motor voltage (red LED, left, and LCD), but adequate main voltage (blue LED, right). Control buttons are not shown.



components and microcontrollers. One of my favorite processor boards — the ATMega128-based Mavric-IIB — features a watchdog timer. This independent timer can be used to reset the microcontroller if the program executing on the controller hangs. Using the watchdog timer, you can assure that the microcontroller resets in a controlled manner following a power glitch, robot impact with something in the environment, or a variety of other causes.

The less powerful but easier to use Parallax BS2 and related chips don't provide this independent watchdog timer function. However, safety routines needn't reside exclusively in the microcontroller. Consider the Parallax HB-25 motor controller, which has a communications timeout function that stops motor activity if it doesn't receive a refresh pulse every four seconds or less. With the timeout function enabled, the HB-25 will render the robot immobile following an accident or component failure that results in the loss of refresh pulses.

Although several motor controllers support a timeout feature, most components require custom safety subroutines to automatically disable autonomous functions. The basis of operation can be a specified time of inactivity or sensor data, such as a tilt reading from an onboard accelerometer.

Physical Sensors

Software subroutines assume functional sensor-microcontroller communications, but this assumption frequently doesn't hold following a mishap. For this reason, it's a good idea to supplement soft safety subroutines with hard wired physical sensors that are independent of microprocessor operation.

I like to install a hard tilt switch across the control input of solid-state relays that control motor current. The physical tilt switch cluster shown in Figure 1 is composed of four mercury switches, each accompanied by a 1K series resistor. When the tilt of the robot platform is greater than the angle defined by the position of the mercury switch on a given axis, the microcontroller control signal is shunted to ground, thereby opening the relay output circuit.

A more compact circuit can be constructed with an accelerometer, but it can be rendered useless if the microcontroller is damaged or runs into

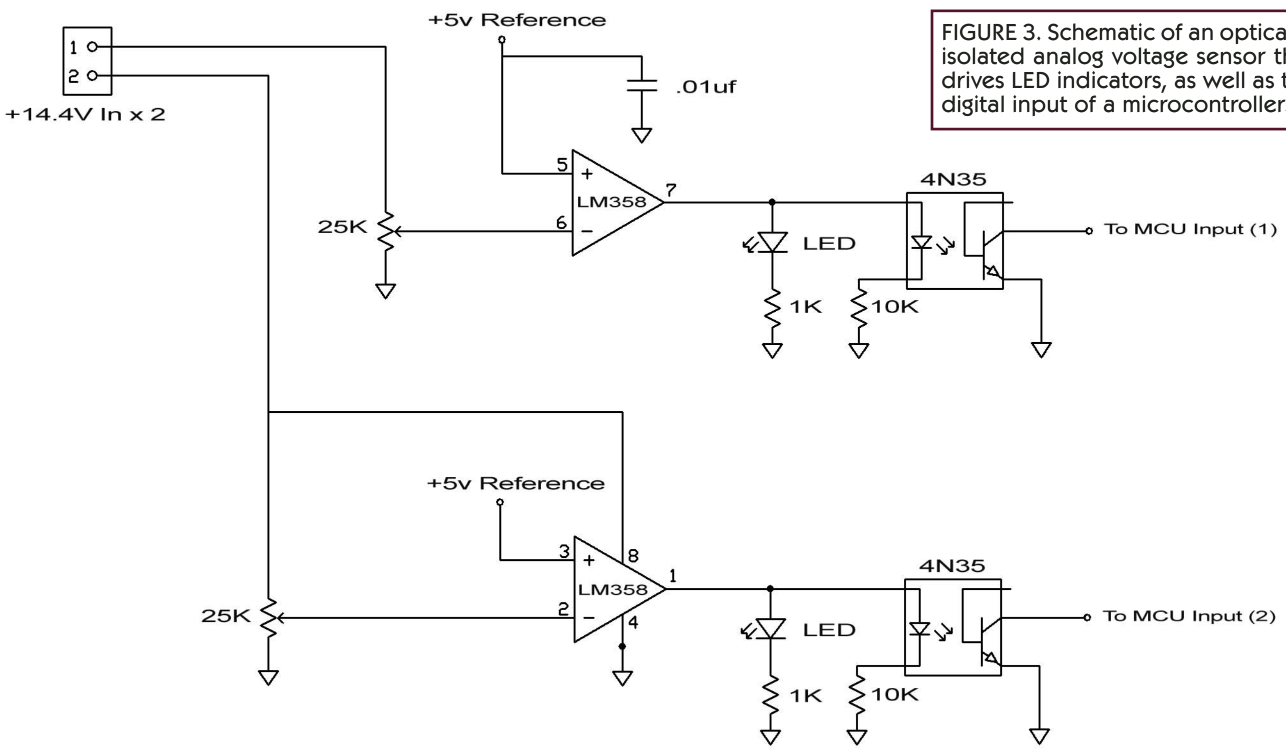


FIGURE 3. Schematic of an optically-isolated analog voltage sensor that drives LED indicators, as well as the digital input of a microcontroller.

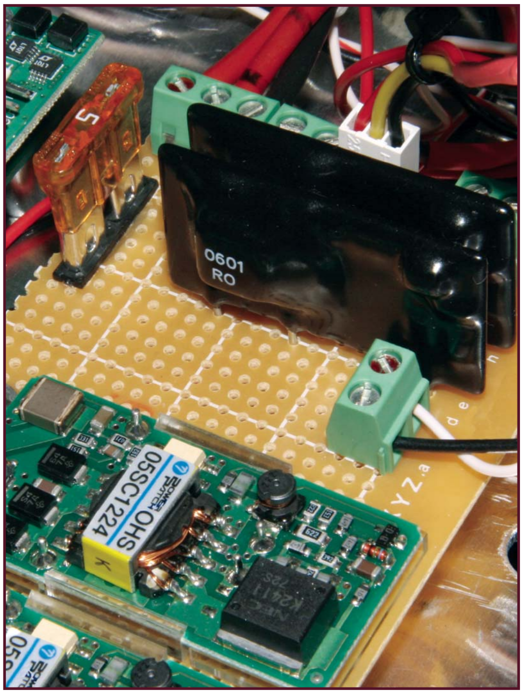


FIGURE 4. Energy management with solid-state relays (upper right) and blade fuse (upper left).

a software glitch. Space permitting, the best approach is to use both a microcontroller-accelerometer circuit and a physical tilt switch. An advantage of this dual configuration is certainty that that robot motion will be halted and the ability to use the event to modify robot behavior.

State Indicators

Once a semi-autonomous robot is programmed and set free, it's difficult to foresee and respond to problems without a real-time indication of internal state. It's a good safety practice to use LEDs and other indicators to communicate autonomous behaviors, such as backing up after sensing an object ahead, as well as system function, such as the status of the controlling communications link.

In addition to general status information provided by single and multi-state LEDs, backlit LCDs can provide a rich source of robot state information. Figure 2 illustrates a robot interface composed of both LEDs and a backlit LCD panel indicating adequate main power but low motor battery voltage. Not visible in the figure is the blinking of the LCD panel and the audible alarm from a piezo buzzer that accompany the error state.

The state indicators in Figure 2 illustrate a parallel or layered safety system. The low motor voltage indicator LED functions independently of the LCD panel and of the driving microcontroller. Battery voltage — in this case, from two separate 14.4V NiMH packs — is fed to two LM386 op-amps that are used to drive red LEDs when the voltage of the corresponding battery pack drops below 11V, as set by the 25K potentiometers (see Figure 3).

You can respond to the LEDs by powering down the robot, replacing the battery packs, and rebooting the robot operating system. In addition, because the output of each op-amp is fed through a 4N35 optocoupler to the input of a microcontroller, a software routine running on the microcontroller can respond to the battery voltage. This response can take the form of an LCD message, the sounding of a piezo buzzer, and activation of system shutdown routines that minimize risk of robot damage.

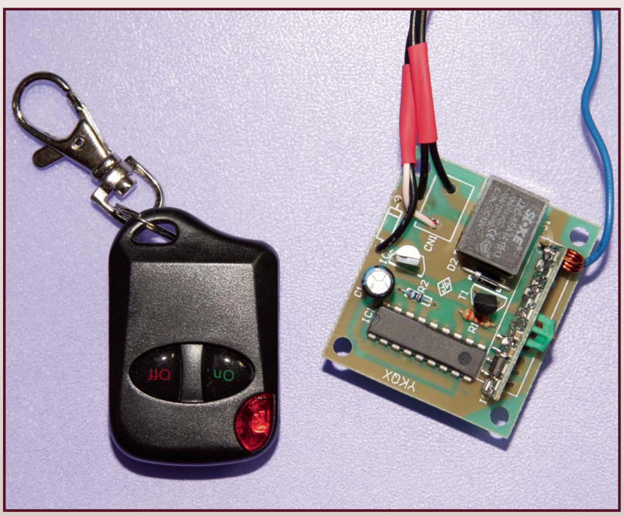


FIGURE 5. An RF on-off switch and transmitter fob reconfigured as a remote kill switch.

Energy Management Hardware and Software

Good energy management safety begins with a physical main power switch that overrides programmatic control systems based on soft switching through relays or transistors. Appropriate capacity standard or self-healing fuses on the battery circuit and power supply output can save microcontrollers, sensors, and battery packs. Mini blade fuses, such as the five amp model shown in Figure 4, are inexpensive and readily available in a range of current ratings down to 1A. Voltage rating is normally not a concern because of the low voltage battery and component circuitry typically used in robots.

The most appropriate current rating and fuse type depends on the time-varying amplitude of the expected current. For example, a slow-blow fuse is appropriate to protect motors and components with a momentary high initial current, such as laser rangefinders. Fast-blow fuses are suitable for microcontroller boards and sensors with relatively constant initial and operational current. The most appropriate fuse rating is as near to the full load current of the circuit as possible. An excellent on-line resource for fuse selection is the *Surge Protection Device Handbook*, listed in the Resources section.

Solid-state relays are useful in secondary power switching because they enable a microcontroller to directly control high currents in response to sensor readings and remote control signals. For example, one of the Crydom 10A solid-state relays in Figure 4 is used to switch on and off a bank of power supply modules, including those shown in the lower half of the figure.

The second relay controls current to a single-board PC with a built-in voltage regulator. Solid-state relays with input requirements of only a few milliamps and output ratings to 60A and above are available from Crydom and other manufacturers. On large, complex robots, I devote a processor to sequencing solid-state relays, powering servos and controllers, monitoring current, and other aspects of safe energy management.

Kill Switch

Flipping the physical main power switch to avoid an impending disaster isn't always convenient, especially if the robot is moving, there are multiple, active robots, or if the robot is out of reach. The simplest solution to stopping a robot in an emergency is to install a wireless kill switch that disables motor drive circuitry when activated.

An inexpensive kill switch system is available from All Electronics Corp., in the form of a keychain remote control shown in Figure 5. The $19 system — which includes a pair of 27 MHz wireless key fobs — is designed to supply 12V to a remote device. However, the main board can be reconfigured so that the contacts of the mechanical relay are available for direct or indirect control of the robot motor system. I purchased several of these units and modified the solder pads on the boards and remotes so that any fob can control any and all robots with a kill switch – an especially handy feature when working with multiple, fast-moving robots. Line-of-sight range is about 100 feet, and about half of that indoors.

A major limitation of using a kill switch as a remote safety device is that it requires you to recognize something is wrong and then press the “off” button to deactivate a robot. An alternative approach is to use a dead man switch, which is roughly equivalent to a watchdog timer in that an active signal is required to keep the robot energized. If the onboard receiver doesn’t detect a signal because you released a momentary-contact switch on the remote, the receiver is out of range, the battery is dead, or any number of other reasons, the robot can be configured to shut down.

Although it’s hard to beat the form factor of the fob shown in Figure 5, the Parallax 433 MHz transmitter and matched receiver can be used to create a relatively compact dead man switch. As with most Parallax devices, setup is quick and painless. Other than supplying each device with 5V at 5 mA, installation is a simple matter of interpreting the analog received signal strength value with an onboard processor or using it to directly control a solid-state relay. Line-of-sight range is about 500 feet.

Construction

The process of creating a robot, whether from a kit or sheet of aluminum and plastic, poses a potential threat to eyes, ears, and extremities. The simple act of trimming a solder joint with a pair of diagonal cutters can launch a wire with enough velocity to embed it in an eye.

Unfortunately, not all construction safety techniques can be gleaned from common sense. If you’re unsure how to do something, then get help. Someone skilled in cutting and bending aluminum sheet can show you safe techniques that could take years to master. Given this caveat, some basics that apply to all robot construction projects include:

• Wear eye protection when using power tools and when cutting wire, plastic, or metal with hand tools.

• Wear ear protection when working with power tools to prevent hearing loss. The high-pitched whine of a Dremel can be muffled with in-ear plugs available at most hardware stores. In-ear plugs, while not as comfortable as over-ear protection, offer superior sound reduction and protection from potential hearing loss.

• Wear short sleeves when working with power tools. Similarly, avoid loose jewelry and anything else that might become caught in a power tool.

• Use clamps with a drill press to avoid a trip to the emergency room to reattach your fingers. Clamps are especially important when step bits are used, because they are more likely to bind than straight bits (See Figure 6).

• Use a lubricant when drilling to minimize binding and reduce noise. I like to use a few drops of rubbing alcohol when drilling aluminum sheeting.

• Smooth freshly cut edges with a file.

• Equip your work area with a fire extinguisher rated for class A (common combustibles), B (electrical), and C (flammable liquids) fires. An ABC extinguisher based on Halotron is ideal for robotics work because the hydrochlorofluorocarbon is an electrical insulator that doesn’t leave a residue.

• Ventilate your work area when soldering or using adhesives. Avoid breathing flux vapors, which can cause permanent respiratory problems.

• Use leadless solder and components, and wash your hands after handling solder. Children are especially susceptible to lead poisoning.

• Use multi-strand, insulated wire for connections that require flexibility, and insulate exposed terminals and wires with shrink-wrap tubing or electrical tape. Solid wire is fine for use with plastic prototyping boards, but more flexible stranded wire is generally a better choice for interconnecting batteries, sensors, power supplies, and processors – especially when there may be relative movement between components.

• Use sealed, leak-proof batteries whenever possible. In addition, don’t leave batteries in your robots for extended periods – even if they are leak-proof.

• Use secure construction techniques, such as locking pins to secure parts, instead of relying on gravity or tape.

Operation

Safe robot operation depends on

FIGURE 6. Use a clamp — not hands — to stabilize items on a drill press — especially with a step bit.

the type of robot, the operating environment, and the purpose of the robot. What constitutes safe operation of a BattleBot engaged in a fight to the death in a BattleBot arena doesn't necessarily apply to a robot designed to explore robot-robot communications. A few globally applicable safe operation heuristics are:

- Charge batteries in a well ventilated area and use a charger that guards against overheating and overcharging.

- Affix robot arms and other stationary robots securely on a stable platform prior to power-up.

- Establish a safety zone around a robot within which no one may enter during operation.

From Here

The above safety measures apply in varying degrees to experimental robots and those designed for exploration and assorted tasks. Robots intended for competition — such as BattleBot-style competitions — must abide by specific safety guidelines. Even if you are not intending to enter your robot in a competition, it's a good idea to review safety guidelines associated with events for ideas on how you can improve the safety of your robot.

A common question regarding robot safety is how much safety overhead is enough. In general, the greater the safety risk, the more safety should be emphasized in robot design. A kill switch and comprehensive energy management circuitry generally don't make sense for a slow moving, 12 ounce carpet rover. However, a highly mobile, 20 pound robot equipped with an expensive sensor array, significant on-board processing hardware, and a high-current drive system may dictate a full suite of safety technologies.

The obvious tradeoffs are safety for increased complexity, weight, cost, and development time. Increased complexity usually translates to a greater likelihood of failure because of an increased number of failure points. For example, a dead battery in a remote enable/kill switch can disable an otherwise operational robot. However, replacing a battery is a trivial matter compared with replacing an expensive video camera because you couldn't stop your robot from slamming into a table leg. **SV**

RESOURCES

All Electronics Corp.
www.allelectronics.com

BattleBot Rules & Guidelines
www.battlebots.com

Crydom, Inc.
www.crydom.com

Parallax
www.parallax.com

Surge Protection Device Handbook.
Cooper Bussman, Inc., 2002.
www.bussman.com/apen/pubs/spd/index.asp

Beginner's Robotics on $50 a Month

– by Paul Pawelski

PART 2: Building the CIRC Bot

Amateur robotics is FUN.

Amateur robotics is EDUCATIONAL.

Amateur robotics is EXPENSIVE!

Last month, you purchased the basic hand tools and supplies you need to build a robot and you learned to solder. This month, you will be building a complete robot. This robot will be more than just a platform. It will be mobile, sense its environment, change how it moves based on what it senses, and be easily modifiable.

Figure 1 shows the complete CIRC Bot. The main body of the robot is a solderless breadboard on a plastic frame. If you have been reading *SERVO* for awhile, you have seen other breadboard bots (and if you have read *Nuts & Volts* since the mid-1990s, you have seen several more). The big difference between this robot and the others that have been presented is that it is built to a tight budget. Most of the time, a beginner's robot is shown with a BASIC Stamp for its microcontroller and modified servos for its motors. Both of those choices are made with

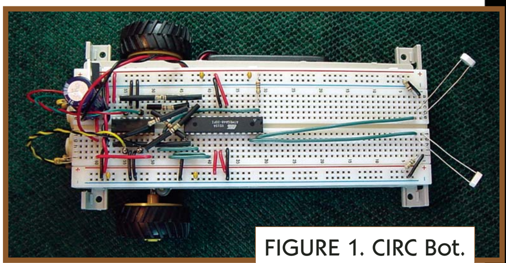


FIGURE 1. CIRC Bot.

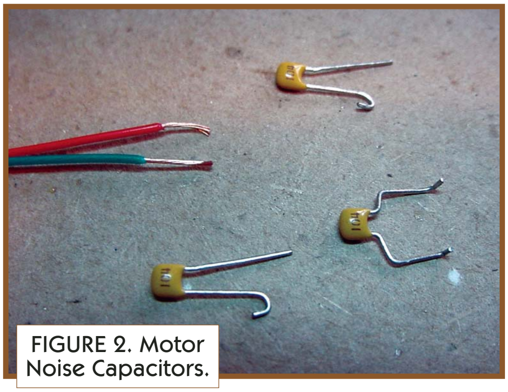


FIGURE 2. Motor Noise Capacitors.

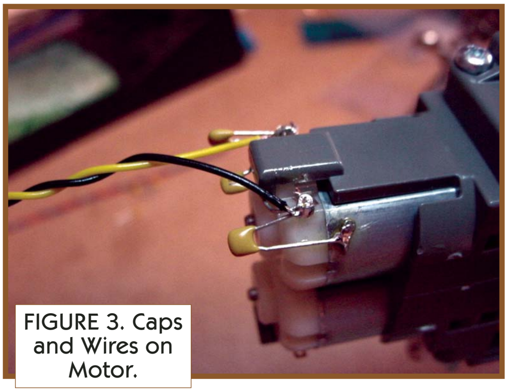


FIGURE 3. Caps and Wires on Motor.

the assumption that it will be easier for the new builder to use these components. However, they add over $50 to the cost of the robot. That is more than our entire budget!

For the CIRC Bot, you will be using a $2.99 AVR Mega48 microcontroller. The Mega48 is not only less than a tenth of the cost of a BASIC Stamp, it is also more powerful.

The robot will be programmed using BASCOM-AVR from MCS Electronics. BASCOM is a version of the Basic programming language. You will need to download a free version of the BASCOM-AVR compiler complete with manual from **www.mcselec.com/index.php?option=com_content&task=view&id=14&Itemid=41**. BASCOM runs on Windows 95 or newer PCs. We will be programming using the parallel (printer) port, so you will need to have a PC available with a parallel port. If you do not already have one, you can often pick up an older PC at a computer repair shop or a second-hand store for less than $50. Just make sure that the computer's operating system is Windows 95, 98, 2000, NT, or XP.

Description	Qty
Solderless Breadboard 830 contacts	1
Wire 3' each of red, green, and black	1
.1 µF Axial Ceramic Capacitor	11
7805T 5V Positive Regulator, 1 Amp	1
100K ohm 1/4 Watt	10
360 ohm 1/4 Watt	11
4 AA Battery Holder w/leads	1
Heat Shrink Tubing	1
40 Pin Header	1
D-Sub Mini Connector/25 Pin Male	1
DB-25 Hood	1
6 Conductor 26GA Multicolor Wire	10
1" Squares 1/16" Thick Foam Squares (15)	1
SN754410 Quad Half-Bridge	1
ATMega48 — Atmel Mega48 Microcontroller	1
Tamiya Truck Tires (1 Pr)	1
Tamiya Double Gearbox	1
Tamiya Universal Plate Set	1
Tamiya Ball Caster (1)	1
CDS Photoresistor	2
2,200 µF 16V Radial Electrolytic Capacitor	1

TABLE 1. Bill of Materials.

Ordering the Parts

The list of components necessary to build the CIRC Bot is shown in Table 1. The entire component list can be purchased as a kit from Wright Hobbies.

Subassemblies

Building the Gearbox

The gearbox kit comes with instructions; however, they are a little limited. Wright Hobbies has made a video of the gearbox assembly procedure. You can find it online at **www.wrighthobbies.net/guides/**. You will be assembling the gearbox in its highest gear ratio configuration (344.2:1).

Adding Caps to the Motors

This is probably the hardest part of the build, but if you practiced your soldering using the LED project described in last month's article, you will be able to handle it. Three of the 0.1 µF capacitors need to be soldered to each motor. One cap will be connected between the two motor leads. The other two will connect each lead to the motor case. In an ideal situation, the caps between the leads and the case would both connect to the case in the same location. However, with the configuration of the Gearbox we are using, this will not be possible, so we will have to solder to two different locations on each motor.

Bend the leads on three capacitors as shown in Figure 2. Hook the leads from the capacitor with both of its leads bent into the two motor lugs. The leads bent into a "J" shape on the other two capacitors also hook into the motor leads. These capacitors should be positioned so that their straight leads touch the metal part of the motor case. Before soldering the capacitors into place, lightly sand the parts of the motor case (which are under the capacitor leads) with a very rough sand paper or emery board. The goal is to make the surface rougher so that solder will bond to it easier. Clean the roughed areas with alcohol. Apply rosin paste to the roughed areas and the leads so that the solder flows into all of the crevices and makes a good bond between the motor and both of the leads. The motor is a fairly large heatsink so you will need to take some time with your iron. *Do not solder the connection yet*.

You need to add hookup wires. Cut two five-inch pieces of wire: one red and one green. Strip about a quarter inch of insulation off of each end. With the wires side by side, twist them together leaving only about one inch at either end untwisted. Take a stripped end of the red wire, twist it around the capacitor lead going through one of the terminal lugs, and take a stripped end of the green wire and twist it around the capacitor lead going through the other terminal lug.

Now, you can position everything

Pin on DB25 Male Connector	Signal	Header Pin	Wire Color	Pin on AVR Mega 48
18-25 (ground)	GND	1	Black	10
11 (busy)	MISO	2	Green	18
5 (D3)	CLOCK	3	Red	19
4 (D2)	RESET	4	Yellow	1
2 (D0)	MOSI	5	Blue	17

TABLE 2. Programming Cable Connections.

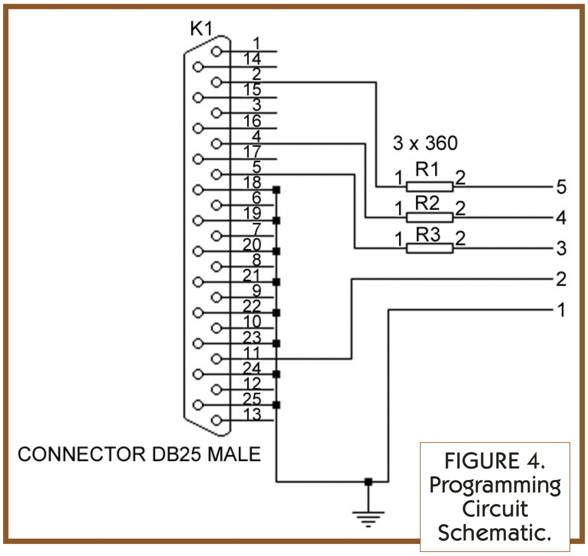


FIGURE 4. Programming Circuit Schematic.

and solder it so that it looks like Figure 3. You will probably have to make several tries to get the joints between the capacitor leads and the motor case to hold. Just take your time and don't give up.

Repeat this process for the other motor.

Making the Programming Cable

The simple cable programmer you will be making is based on a design originally in *Sample Electronics* which is also described on the Dontronics website at **www.dontronics.com/dt006_programming.html**

The only differences between this cable and the one on the Dontronics site are the pinout on the end that will connect to the bot and the size of the resistors. Figure 4 and Table 2 show the connections. Note that pin 18, 19, 20, 21, 22, 23, 24, and 25 of the DB25 male connector must be connected to each other. To protect your computer's printer port, 360 ohm resistors have been added in series with the MOSI, RESET, and CLOCK signals.

Cut a four-foot piece of the six conductor cable. Cut off one inch of the outside cover from one end of the cable and 1-1/2 inches of the outside cover from the other end. Trim the white wire so that it ends where the outer cover ends on both ends. On the end with 1-1/2 inches of the cover removed, trim the blue, yellow, and red wires so that only an inch of each of them is exposed. Strip one half inch of insulation from both ends of all the wires so that the cable looks like Figure 5.

The resistors and the pin jumpers will be added inside the connector hood. Because of the short distances involved, it will be easier to make jumpers out of leads trimmed off of components than out of the hookup wire. You will have to take your time while doing this to make sure that everything fits. Read the rest of this subsection several times and lay the pieces out to make sure that you understand how everything fits before you start cutting.

Take three 360 ohm resistors and trim their leads so that only 8 mm (a little more than a quarter of an inch) is left on each end. Save the pieces you trimmed off. The solder cups on the DB25 connector have numbers next to them. Solder one end of one of the trimmed resistors into the solder cup for pin 2 (don't forget the rosin paste). Do the same thing with another resistor in the cup for pin 4 and take the third resistor and solder it into the cup for pin 5.

Lay one of the pieces you trimmed off of the resistors across the solder cups for pins 18 to 25. Solder it to each of these cups.

The DB25 hood came with several pieces of hardware which are used to secure the cable and hold the two pieces of the hood together. The two curved pieces of metal are called cable clamps. Normally, they go around the cable where it enters the hood with their concave sides facing each other to form a ring around the cable. However, our cable is smaller so we will be placing the two pieces around the cable with the curved parts facing the same way. Attach the cable clamps around the cable at the end with the 1-1/2 inches of exposed wires as shown in Figure 6. You now need to solder the wires as shown in the photo: the blue wire to the resistor for pin 2, the yellow wire to the resistor for pin 4, the red wire to the

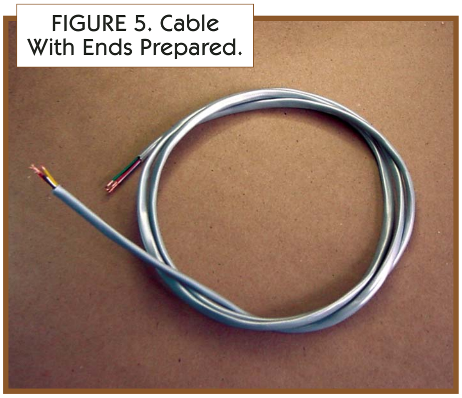


FIGURE 5. Cable With Ends Prepared.

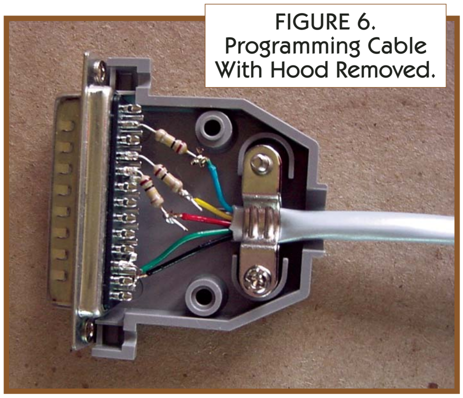


FIGURE 6. Programming Cable With Hood Removed.

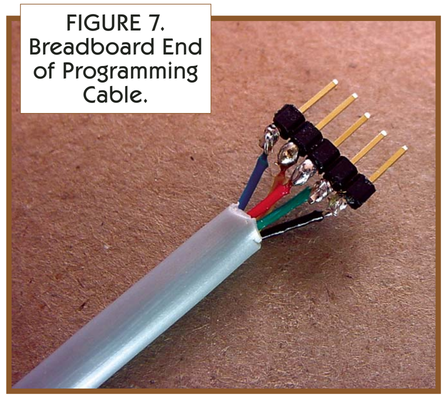


FIGURE 7. Breadboard End of Programming Cable.

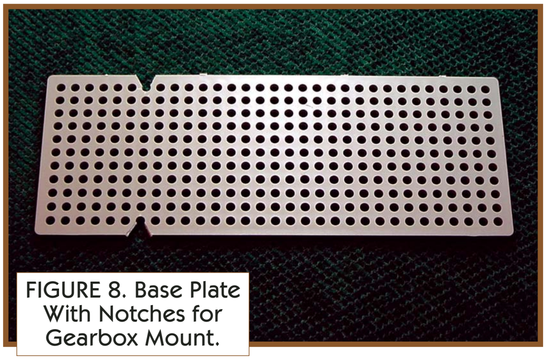


FIGURE 8. Base Plate With Notches for Gearbox Mount.

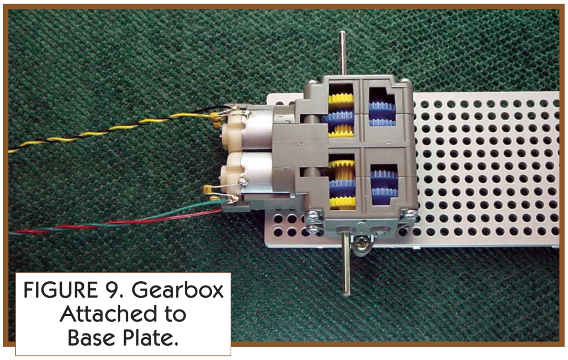


FIGURE 9. Gearbox Attached to Base Plate.

resistor for pin 5, the green wire to the solder cup for pin 11, and the black wire to the jumper you soldered across pins 10 to 25. Assemble the rest of the hood around the assembly.

On the other end of the programming cable, we will solder the wires to five header pins. Break off a set of five pins from the 40-pin header. Solder the wires as shown in Figure 7. The order of the connections on the header pin side is GND (Black), MISO (Green), Clock (Red), Reset (Yellow), and MOSI (Blue). The plastic used to hold the header pins together will soften fast due to the heat from your soldering iron. You will need to work in fairly quick bursts. If the plastic softens too much around one of the pins, the pin will be able to move once the plastic is cool. If this happens, use a drop of super glue to hold the pin in the plastic.

Since the programming cable is the part that will connect your low-cost robot to your expensive computer, you want to make sure that you built it correctly. Take your multimeter and confirm that each header pin is wired to only the intended connector pin. To do this, set your meter to resistance (or "continuity" if your meter has a continuity tone setting). Touch one lead of the meter to header pin 1 (GND) and touch the other lead to each pin on the connector (one at a time). The meter should show infinite resistance on any pin except pins 18 through 25, which should show nearly zero resistance.

Move the lead to header pin 2 (MISO) and repeat the process. This time, you should get infinite resistance on all connector pins except on pin 11 where resistance will be close to zero. Repeat with the other three header pins. Since these three pins have resistors in their circuits, you should get resistance values close to 360 ohms when you touch the correct connector pins instead of zero ohms as you did on the previous header pins.

Base Assembly

The base for the CIRC Bot is made using the Tamiya Universal Plate Set. Take one of the plate pieces from the base kit. Count in seven rows from one edge. Using angled snips, cut a "v" from the edges of the plate to the first and last holes in the seventh row as shown in Figure 8. These notches will allow you to attach the gearbox to the base.

Place the gearbox under the plate with the mounting tabs facing up and located under the notches you just cut in the plate. Place the gearbox mounting rod across the top of the plate with the end stubs going into the notches in the plate. Holding everything together in one hand, turn it over and install the mounting screws through the tabs and into the stubs. The screws are self-tapping in the plastic so you must be careful not to over-tighten them or they will strip out the stubs. Just tighten them until the heads of the screws make contact with the tabs. The gearbox assembled to the base is shown in Figure 9.

Next, the battery box will be attached. Take three pieces of the double-stick foam tape and stick them to the side of the box where the switch is located as shown in Figure 10. Stick the battery box to the bottom of the plate just in front of the gearbox with the switch just clearing the side of the plate as shown in Figure 11.

The final piece that will be added to the bottom of the plate is the roller-ball. Assembly instructions come with the roller-ball kit. Assemble the unit for the 25 mm height. The roller-ball is attached using the two-piece plastic rivets from the base kit. Locate the roller-ball so that it is centered on the plate with its front mounting holes over the third row of holes in the plate. From the top of the plate, push the hollow part of the rivet through the plate hole and the mounting hole in the roller-ball assembly. Push the solid part of the rivet through the hollow part to cause it to expand and grip the roller-ball assembly. Repeat this for the three remaining mounting holes to

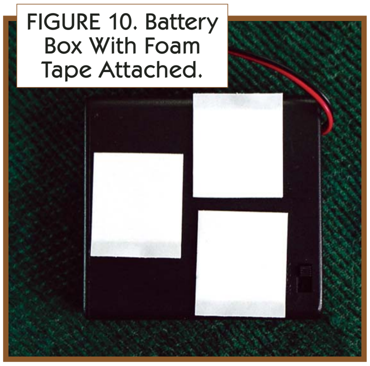


FIGURE 10. Battery Box With Foam Tape Attached.

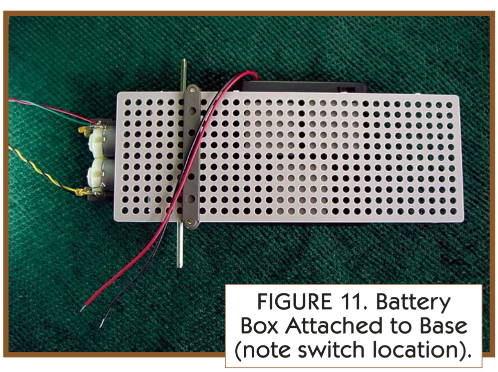


FIGURE 11. Battery Box Attached to Base (note switch location).

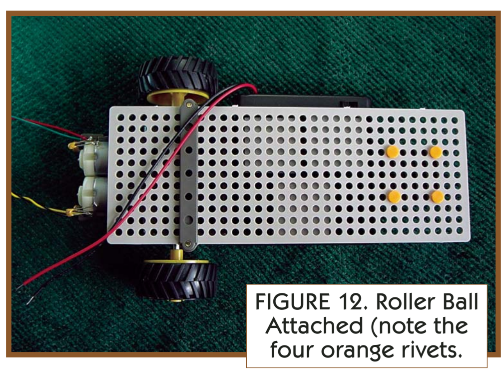


FIGURE 12. Roller Ball Attached (note the four orange rivets.

make the finished assembly.

Now, press the wheels and tires onto the axles extending from the gearbox. The base will now look like Figure 12.

On the top of the plate, two long angle pieces need to be added. These angle pieces will hold standoffs, which will be added to the robot next month. Place one over the very front row of holes as shown in Figure 13. Use two plastic rivets to attach it to the plate. The second angle piece will be attached in the same manner at the other end of the plate. However, it will need to have part of its structure trimmed as shown in Figure 14.

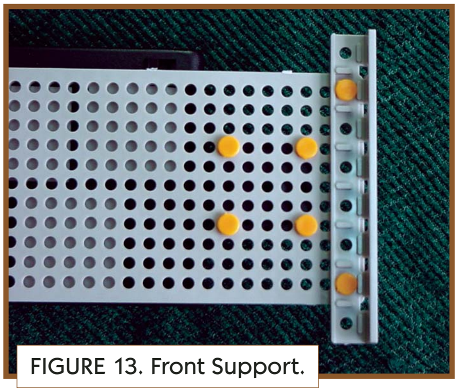


FIGURE 13. Front Support.

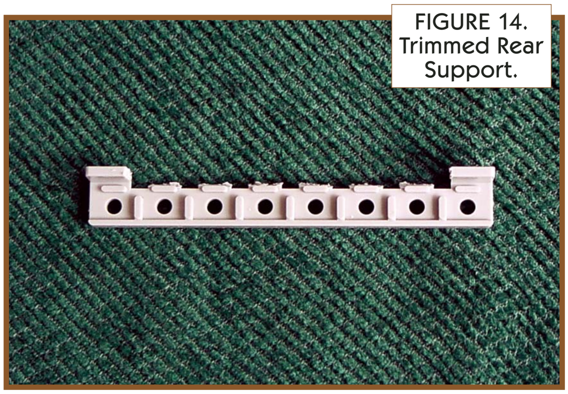


FIGURE 14. Trimmed Rear Support.

Now, the second plate will be attached to the first plate. Make four stacks of double-stick foam tape as shown in Figure 15. Each stack should be three pieces deep. Two of the stacks are behind the motor mount and will need to be trimmed slightly to fit. Attach the second plate to the top of the stacks. Twist the two battery box wires together and route them through the holes in the base plates immediately in front of the motor mounts so that the completed base looks like Figure 16.

The final piece for the base is the solderless breadboard. This comes with double-stick tape on its backside. Position the breadboard *before removing the protective paper* from the tape. Note the numbered rows and lettered columns on the breadboard. The board should be placed with the edge near row one, flush with the edge of the angle support on the front of the base and the edge nearest column J, flush with the side of the base as shown in Figure 17.

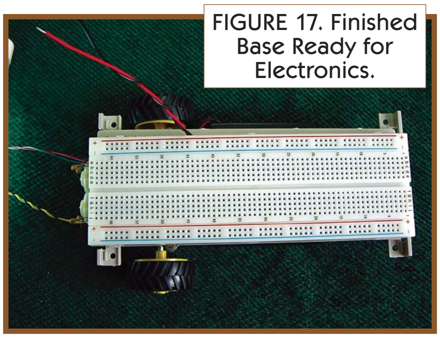


FIGURE 17. Finished Base Ready for Electronics.

Top Assembly, Programming, and Troubleshooting

The instructions for completing the rest of the CIRC Bot can be found at **www.wrighthobbies.net/guides/**. They will also tell you how to program the robot and troubleshoot it if everything does not go perfect the first time.

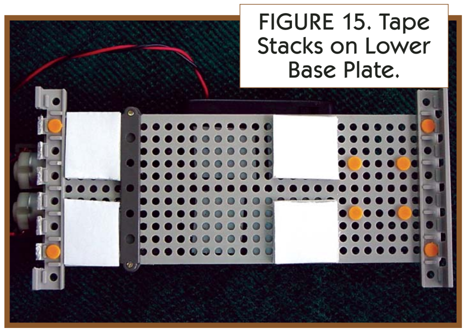


FIGURE 15. Tape Stacks on Lower Base Plate.

Conclusion

When you finish the work described in this month's article and the online supplement, you can honestly say that you built a robot. Next month, we will greatly enhance your robot's sensor suite and we will add an LCD display so your bot can tell you what it is thinking. Until then, have fun with your project! SV

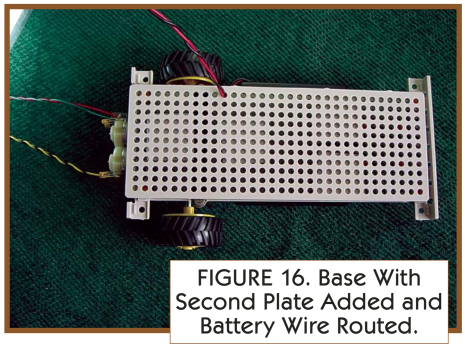


FIGURE 16. Base With Second Plate Added and Battery Wire Routed.

Seeing With OpenCV

A Computer-Vision Library

by Robin Hewitt

PART 1

OpenCV — Intel's free, open-source computer-vision library — can greatly simplify computer-vision programming. It includes advanced capabilities — face detection, face tracking, face recognition, Kalman filtering, and a variety of artificial-intelligence (AI) methods — in ready-to-use form. In addition, it provides many basic computer-vision algorithms via its lower-level APIs.

A good understanding of how these methods work is the key to getting good results when using OpenCV. In this five-part series, I'll introduce you to OpenCV and show you how to use it to implement face detection, face tracking, and face recognition. Then, I'll take you behind the scenes to explain how each of these methods works and give you tips and tricks for getting the most out of them.

This first article introduces OpenCV. I'll tell you how to get it and give you a few pointers for setting it up on your computer. You'll learn how to read and write image files, capture video, convert between color formats, and access pixel data — all through OpenCV interfaces.

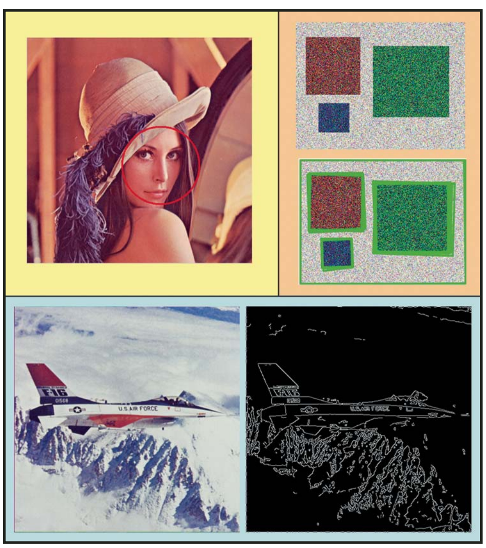


FIGURE 1. Among OpenCV's many capabilities are face detection (top left), contour detection (top right), and edge detection (bottom).

OpenCV Overview

OpenCV is a free, open-source computer vision library for C/C++ programmers. You can download it from **http://sourceforge.net/projects/opencvlibrary**.

Intel released the first version of OpenCV in 1999. Initially, it required Intel's Image Processing Library. That dependency was eventually removed, and you can now use OpenCV as a standalone library.

OpenCV is multi-platform. It supports both Windows and Linux, and more recently, MacOSX. With one exception (CVCAM, which I'll describe later in this article), its interfaces are platform independent.

Features

OpenCV has so many capabilities, it can seem overwhelming at first. Fortunately, you'll need only a few to get started. I'll walk you through a useful subset in this series.

Here's a summary of the major functionality categories in OpenCV, version 1.0, which was just released at the time of this writing:

Image and video I/O

These interfaces let you read in image data from files, or from live video feed. You can also create image and video files.

General computer-vision and mage-processing algorithms (mid- and low-level APIs)

Using these interfaces, you can experiment with many standard computer vision algorithms without having to code them yourself. These include edge, line, and corner detection, ellipse fitting, image pyramids for multiscale processing, template matching, various transforms (Fourier, discrete cosine, and distance transforms), and more.

High-level computer-vision modules

OpenCV includes several high-level capabilities. In addition to face-detection, recognition, and tracking, it includes optical flow (using camera motion to determine 3D structure), camera calibration, and stereo.

AI and machine-learning methods

Computer-vision applications often require machine learning or other AI methods. Some of these are available in OpenCV's Machine Learning package.

Image sampling and view transformations

It's often useful to process a group of pixels as a unit. OpenCV includes interfaces for extracting image subregions, random sampling, resizing, warping, rotating, and applying perspective effects.

Methods for creating and analyzing binary (two-valued) images

Binary images are frequently used in inspection systems that scan for shape defects or count parts. A binary representation is also convenient when locating an object to grasp.

Methods for computing 3D information

These functions are useful for mapping and localization – either with a stereo rig or with multiple views from a single camera.

Math routines for image processing, computer vision, and image interpretation

OpenCV includes math commonly used, algorithms from linear algebra, statistics, and computational geometry.

Graphics

These interfaces let you write text and draw on images. In addition to various fun and creative possibilities, these functions are useful for labeling and marking. For example, if you write a program that detects objects, it's helpful to label images with their sizes and locations.

GUI methods

OpenCV includes its own windowing interfaces. While these are limited compared to what can be done on each platform, they provide a simple, multi-platform API to display images, accept user input via mouse or keyboard, and implement slider controls.

Datastructures and algorithms

With these interfaces, you can efficiently store, search, save, and manipulate large lists, collections (also called sets), graphs, and trees.

Data persistence

These methods provide convenient interfaces for storing various types of data to disk and retrieving them later.

Figure 1 shows a few examples of OpenCV's capabilities in action: face detection, contour detection, and edge detection.

Organization

OpenCV's functionality is contained in several modules.

CXCORE contains basic datatype definitions. For example, the data structures for image, point, and rectangle are defined in cxtypes.h. CXCORE also contains linear algebra and statistics methods, the persistence functions, and error handlers. Somewhat oddly, the graphics functions for drawing on images are located here, as well.

CV contains image processing and camera calibration methods. The computational geometry functions are also located here.

CVAUX is described in OpenCV's documentation as containing obsolete and experimental code. However, the simplest interfaces for face recognition are in this module. The code behind them is specialized for face recognition, and they're widely used for that purpose.

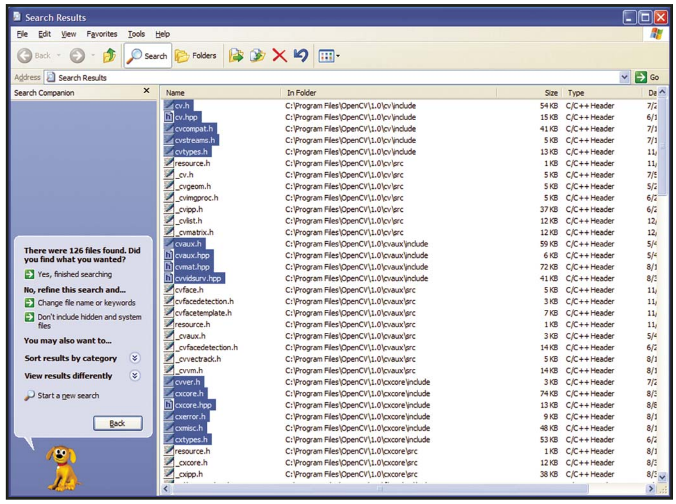


FIGURE 2. Selecting OpenCV header files in Windows to place into a single include directory.

ML contains machine-learning interfaces

The remaining functionality is contained in HighGUI and CVCAM. Both of these are located in a directory named "otherlibs," making them easy to miss. Since HighGUI contains the basic I/O interfaces, you'll want to be sure you don't overlook it! It also contains the multi-platform windowing capabilities.

CVCAM contains interfaces for video access through DirectX on 32-bit Windows platforms. However, HighGUI also contains video interfaces. In this article, I'll cover only the interfaces in HighGUI. They're simpler to use, and they work on all platforms. If you're using Windows XP or 2000, you may get a performance boost by switching to the CVCAM interfaces, but for learning OpenCV, the simpler ones in HighGUI are just fine.

Installing OpenCV

Basic Install

OpenCV for Linux or MacOSX is packaged as a source-code archive. You'll need to build both static and shared-object libraries. You can either build an RPM first and install from that, or compile and install it directly. Instructions for doing both are in INSTALL.

The Windows download is packaged as an executable that installs OpenCV when you run it. It places OpenCV files into a directory of your choice, optionally modifies your system path to include the OpenCV binaries, and registers several DirectX filters. By default, it installs to C:/Program Files/OpenCV/<version>.

Customizing a Windows Install

For Windows users, OpenCV is easy to install, and the default installation will work. But a bit of advance planning may leave you happier with the results. Here are a few suggestions.

Since OpenCV is a developers' toolkit — not a program — you may want to locate it somewhere other than your Program Files directory. If you do prefer to locate it elsewhere, decide that before you run the installer, and enter that location when asked.

I suggest you also decide — before installing — how you want Windows to find the OpenCV dlls. You can either modify your system's PATH variable to include their location, or you can move them, after installing, from OpenCV's "bin" directory to your SYSTEM_ROOT directory.

If you prefer to move the dlls, but aren't sure where your SYSTEM_ROOT directory is, you can locate it by running the sysinfo utility available at **www.cognotics.com/utilities**.

If you prefer to modify the PATH rather than moving the dlls, you can have the installer do that for you by selecting the check box "Add bin directory to PATH."

After Installing

The OpenCV directory contains several subdirectories. The docs directory contains html documentation for all the OpenCV functions and datatypes. Since the best documentation is a working example, you might also want to browse the "samples" directory.

The header files you'll need to include when you compile programs that use OpenCV are distributed among the

```
1 // ImageIO.c
2 //
3 // Example showing how to read and write images
4
5 #include "cv.h"
6 #include "highgui.h"
7 #include <stdio.h>
8
9 int main(int argc, char** argv)
10 {
11    IplImage * pInpImg = 0;
12
13    // Load an image from file
14    cvLoadImage("my_image.jpg", CV_LOAD_IMAGE_UNCHANGED);
15    if(!pInpImg)
16    {
17       fprintf(stderr, "failed to load input image\n");
18       return -1;
19    }
20
21    // Write the image to a file with a different name,
22    // using a different image format -- .png instead of .jpg
23    if( !cvSaveImage("my_image_copy.png", pInpImg) )
24    {
25       fprintf(stderr, "failed to write image file\n");
26    }
27
28    // Remember to free image memory after using it!
29    cvReleaseImage(&pInpImg);
30
31    return 0;
32 }
```

FIGURE 3. Example program that reads an image from a file and writes it to a second file in a different compression format.

OpenCV modules. Although you don't need to do this, I like to gather them together into a single include directory.

On both Linux and Windows, you can locate the headers by searching the install directory and subdirectories for filenames that match the pattern *.h, *.hpp. There will be lots of matches. You don't need all of them. Headers for all modules except HighGUI are in separate "include" directories inside each module. You can skip headers in the "src" directories for these modules. For HighGUI, you'll need highgui.h, located in otherlibs/highgui.

Programming with OpenCV: Some Basics

More about Headers and Libraries

Most OpenCV programs need to include cv.h and highgui.h. Later, for face recognition, we'll also include cvaux.h. The remaining header files are included by these top-level headers.

If you've left the header files in multiple directories (default installation), make sure your compiler's include path contains these directories. If you've gathered the headers into one include directory, make sure that directory is on your compiler's include path.

Your linker will need both the library path and the names of the static libraries to use. The static libraries you need to link to are cxcore.lib, cv.lib, and highgui.lib. Later, for face recognition, you'll also link to cvaux.lib. These are in OpenCV's "lib" directory.

```
1 // Capture.c
2 //
3 // Example showing how to connect to a webcam and capture
4 // video frames
5
6 #include "stdio.h"
7 #include "string.h"
8 #include "cv.h"
9 #include "highgui.h"
10
11 int main(int argc, char ** argv)
12 {
13    CvCapture * pCapture    = 0;
14    IplImage *  pVideoFrame = 0;
15    int         i;
16    char        filename[50];
17
18    // Initialize video capture
19    pCapture = cvCaptureFromCAM( CV_CAP_ANY );
20    if( !pCapture )
21    {
22       fprintf(stderr, "failed to initialize video capture\n");
23       return -1;
24    }
25
26    // Capture three video frames and write them as files
27    for(i=0; i<3; i++)
28    {
29       pVideoFrame = cvQueryFrame( pCapture );
30       if( !pVideoFrame )
31       {
32          fprintf(stderr, "failed to get a video frame\n");
33       }
34
35       // Write the captured video frame as an image file
36       sprintf(filename, "VideoFrame%d.jpg", i+1);
37       if( !cvSaveImage(filename, pVideoFrame) )
38       {
39          fprintf(stderr, "failed to write image file %s\n", filename);
40       }
41
42       // IMPORTANT: Don't release or modify the image returned
43       // from cvQueryFrame() !
44    }
45
46    // Terminate video capture and free capture resources
47    cvReleaseCapture( &pCapture );
48
49    return 0;
50 }
```

FIGURE 4. Example program that captures live video frames and stores them as files.

Reading and Writing Images

Image I/O is easy with OpenCV. Figure 3 shows a complete program listing for reading an image from file and writing it as a second file, in a different compression format.

To read an image file, simply call cvLoadImage(), passing it the filename (line 14). OpenCV supports most common image formats, including JPEG, PNG, and BMP. You don't need to provide format information. cvLoadImage() determines file format by reading the file header.

To write an image to file, call cvSaveImage(). This function decides which file format to use from the file extension. In this example, the extension is "png," so it will write the image data in PNG format.

Both cvLoadImage() and cvSaveImage() are in the HighGUI module.

When you're finished using the input image received from cvLoadImage(), free it by calling cvReleaseImage(), as on line 29. This function takes an address of a pointer as its input because it does a "safe release." It frees the image structure only if it's non-null. After freeing it, it sets the image pointer to 0.

Live Video Input

Capturing image frames from a webcam, or other digital video device, is nearly as easy as loading from file. Figure 4 shows a complete program listing to initialize frame capture, capture and store several video frames, and close the capture interface.

The capture interface is initialized, on line 19, by calling cvCaptureFromCAM(). This function returns a pointer to a CvCapture structure. You won't access this structure directly. Instead, you'll store the pointer to pass to cvQueryFrame().

When you're finished using video input, call cvReleaseCapture() to release video resources. As with cvReleaseImage(), you pass the address of the CvCapture pointer to cvReleaseCapture().

Don't release or otherwise modify

the IplImage you receive from cvQueryFrame()! If you need to modify image data, create a copy to work with:

```
// Copy the video frame
IplImage *pImgToChange =
     cvCloneImage(pVideoFrame);

// Insert your image-processing code here ...

// Free the copy after using it
cvReleaseImage(&pImgToChange);
```

Color Conversions

Figure 5 shows code for converting a color image to grayscale. OpenCV has built-in support for converting to and from many useful color models, including RGB, HSV, YCrCb, and CIELAB. (For a discussion of color models, see "The World of Color," *SERVO Magazine*, November 2005.)

Note that the conversion function, cvCvtColor(), requires two images in its input list. The first one, pRGBImg, is the source image. The second, pGrayImg, is the destination image. It will contain the conversion result when cvCvtColor() returns.

Because this paradigm of passing source and destination images to a processing function is common in OpenCV, you'll frequently need to create a destination image. On line 25, a call to cvCreateImage() creates an image the same size as the original, with uninitialized pixel data.

How OpenCV Stores Images

OpenCV stores images as a C structure, IplImage. IPL stands for Image Processing Library, a legacy from the original OpenCV versions that required this product.

The IplImage datatype is defined in CXCORE. In addition to raw pixel data, it contains a number of descriptive fields, collectively called the Image Header. These include

- Width — Image width in pixels

- Height — Image height in pixels

- Depth — One of several predefined constants that indicate the number of bits per pixel per channel. For example, if depth=IPL_DEPTH_8U, data for each pixel channel are stored as eight-bit, unsigned values.

- nChannels — The number of data channels (from one to four). Each channel contains one type of pixel data. For example, RGB images have three channels – red, green, and blue intensities. (These are sometimes called BGR images, because pixel data are stored as blue, green, then red values.) Grayscale images contain only one channel – pixel brightness.

Accessing Pixel Values

It's possible to create many types of functionality using OpenCV without directly accessing raw pixel data. For example, the face detection, tracking, and recognition programs described later in this series never manipulate raw pixel data directly. Instead, they work with image pointers and other high-level constructs. All pixel-level calculations are performed inside OpenCV functions. However, if you write your own image-processing algorithms, you may need to access raw pixel values. Here are two ways to do that:

1. Simple Pixel Access

The easiest way to read individual pixels is with the cvGet2D() function:

```
CvScalar cvGet2D(const CvArr*,
              int row, int col);
```

This function takes three parameters: a pointer to a data container (CVArr*), and array indices for row and column location. The data container can be an IplImage structure. The topmost row of pixels is row=0, and the bottommost is row=height-1.

The cvGet2D()function returns a C structure, CvScalar, defined as

```
typedef struct CvScalar
{
              double val[4];
}
CvScalar;
```

The pixel values for each channel are in val[i]. For grayscale images, val[0] contains pixel brightness. The other three values are set to 0. For a three-channel, BGR image, blue=val[0], green=val[1], and red=val[2].

The complementary function, cvSet2D(), allows you to modify pixel values. It's defined as

```
1 // ConvertToGray.c
2 //
3 // Example showing how to convert an image from color
4 // to grayscale
5
6 #include "stdio.h"
7 #include "string.h"
8 #include "cv.h"
9 #include "highgui.h"
10
11 int main(int argc, char** argv)
12 {
13    IplImage * pRGBImg  = 0;
14    IplImage * pGrayImg = 0;
15
16    // Load the RGB image from file
17    pRGBImg = cvLoadImage("my_image.jpg", CV_LOAD_IMAGE_UNCHANGED);
18    if(!pRGBImg)
19    {
20       fprintf(stderr, "failed to load input image\n");
21       return -1;
22    }
23
24    // Allocate the grayscale image
25    pGrayImg = cvCreateImage
26       ( cvSize(pRGBImg->width, pRGBImg->height), pRGBImg->depth, 1 );
27
28    // Convert it to grayscale
29    cvCvtColor(pRGBImg, pGrayImg, CV_RGB2GRAY);
30
31    // Write the grayscale image to a file
32    if( !cvSaveImage("my_image_gray.jpg", pGrayImg) )
33    {
34       fprintf(stderr, "failed to write image file\n");
35    }
36
37    // Free image memory
38    cvReleaseImage(&pRGBImg);
39    cvReleaseImage(&pGrayImg);
40
41    return 0;
42 }
```

FIGURE 5. Example program for converting a color image to grayscale.

Resources

Sourceforge site
http://sourceforge.net/projects/opencvlibrary

Official OpenCV usergroup
http://tech.groups.yahoo.com/group/OpenCV

OpenCV Wiki
http://opencvlibrary.sourceforge.net

Source code for the program listings in this article are available for download at **www.cognotics.com/opencv/servo**.

```
void cvSet2D(CvArr*, int row, int col,
             CvScalar);
```

2. Fast Pixel Access

Although cvGet2D() and cvSet2D() are easy to use, if you want to access more than a few pixel values, and performance matters, you'll want to read values directly from the raw data buffer, IplImage.imageData.

Image data in the buffer are stored as a 1D array, in row-major order. That is, all pixel values in the first row are listed first, followed by pixel values in the second row, and so on.

For performance reasons, pixel data are aligned, and padded if necessary, so that each row starts on an even four-byte multiple. A second field, IplImage.widthStep, indicates the number of bytes between the start of each row's pixel data. That is, row i starts at IplImage.imageData + i*IplImage.widthStep.

IplImage.imageData is defined as type char*, so you may need to cast the data type. For example, if your image data are unsigned bytes (the most common input type), you'd cast each value to unsigned char* before assigning, or otherwise using, it.

If you're accessing data from a grayscale (single-channel) image, and the data depth is eight bits (one byte per pixel), you'd access pixel[row][col] with

```
pixel[row][col] = ((uchar*)
  (pImg->imageData +
  row*pImg->widthStep + col));
```

In multi-channel images, channel values are interlaced. Here's a code snippet to access blue, green, and red pixel values:

```
step  = pImg->widthStep;
nChan = pImg->nChannels;
// = 3 for a BGR image
buf   = pImg->imageData;

blue[row][col] =
              ((uchar*)(buf + row*widthStep +
              nChan*col);
green[row][col] =
              ((uchar*)(buf + row*widthStep +
              nChan*col + 1);
red[row][col] =
              ((uchar*)(buf + row*widthStep +
              nChan*col + 2);
```

Finally, if image depth is greater than eight bits (for example, IPL_DEPTH_32S), you'd need to transfer multiple bytes for each value and multiply the buffer offset by the number of data bytes for your image depth. It's very unlikely, however, that you'll encounter a situation in which you must access multi-byte pixel values directly.

Finding Help

If you have problems installing or using OpenCV, the first place to turn for help is the FAQ (faq.htm) in your OpenCV docs directory. The INSTALL file, at the root of your OpenCV directory, also contains helpful setup and troubleshooting tips. If these don't answer your question, you may want to post a query to the official Yahoo! user group. The group's URL is in the Resources sidebar.

API documentation for each module is in the docs/ref subdirectory. All reference manuals except the one for CVAUX are linked from index.htm, in the docs directory.

Coming Up ...

Next month, I'll show you how to detect faces with OpenCV and explain the algorithm behind the interface. Be seeing you! **SV**

About the Author

Robin Hewitt is an independent software consultant working in the areas of computer vision and robotics. She has worked as a Computer Vision Algorithm Developer at Evolution Robotics and is a member of SO(3), a computer-vision research group at UC San Diego. She is one of the original developers of SodaVision, an experimental face-recognition system at UC San Diego. SodaVision was built with OpenCV.

EVENTS CALENDAR

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

Happy New Year, robot builders! The 2007 contest dates are beginning to roll in as more organizations finalize their plans for the new year.

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

January 2007

16-19 **Singapore Robotic Games**
Republic of Singapore
Lots of events including wall climbing, pole balancing, Micromouse, sumo, legged robot races, robot soccer, and more.
http://guppy.mpe.nus.edu.sg/srg

20 **Robot Sumo in DC**
Washington, DC
The name says it all. This is a robot sumo contest held in Washington, DC.
www.societyofrobots.com/sumo_robots_in_DC.shtml

26-28 **Techfest**
Indian Institute of Technology, Bombay, India
Micromouse and two other events with the intriguing names of SNAP and Full Throttle: Afterburn.
www.techfest.org

February

1-4 **Robotix**
IIT Khargpur, West Bengal, India
A national-level competition. Events include Fastrack Manual, Fastrack Auto, and Softandroid.
http://gymkhana.iitkgp.ac.in/robotix

26 **APEC Micromouse Contest**
Anaheim, CA
One of the best-known micromouse competitions in the United States. Expect to see some very advanced and fast micromouse robots.
www.apec-conf.org

March

3 **RoboWars**
Montreal, Canada
Sumo and BEAM Solaroller events.
www.robowars.ca

9-10 **AMD Jerry Sanders Creative Design Contest**
University of Illinois at Urbana-Champaign, IL
Check the website for the details of this year's contest.
http://dc.cen.uiuc.edu

9-10 **National Robotics Challenge**
Veterans Memorial Coliseum, Marion, OH
In addition to Sumo and maze solving events, this student competition includes two unusual ones: a robotic workcell event and a pick-and-place event.
www.nationalroboticschallenge.org

10 **CIRC Central Illinois Bot Brawl**
Peoria, IL
Includes several classes of autonomous sumo and remote-control vehicle destruction.
www.circ.mtco.com

17-18 **Manitoba Robot Games**
Winnipeg, Manitoba, Canada
Events may include both Japanese and Western style sumo, mini-tractor pull, and Atomic Hockey.
www.scmb.mb.ca

31 **Penn State Abington Fire-Fighting Robot Contest**
Penn State Abington, Abington, PA
Regional for the Trinity Fire Fighting contest.
www.ecsel.psu.edu/~avanzato/robots/contests/outdoor/contest05.htm

31 **Penn State Abington Mini Grand Challenge**
Penn State Abington, Abington, PA
Autonomous outdoor ground robots must navigate on and off-road around the campus, avoiding obstacles.
www.ecsel.psu.edu/~avanzato/robots/contests/outdoor/contest05.htm

by Dave Calkins

ROBOGAMES PREP:

Tetsujin! SERVO's Original Robot Challenge

Tetsujin — the ultimate robot competition. When we build other robots, we're always putting out time, money, and efforts into the projects. More often than not, we put our souls into the project. But how often do we put *ourselves* into the project? I mean that literally. No one would enter a combat robot arena, or risk riding in a Grand Challenge vehicle. But Tetsujin takes it to the ultimate — man and machine become one.

And to what end? Is this just something that we do as yet another robot competition? Something that's fun and can win us medals or cash? Or is it something more? Something that can truly benefit man.

Not only could the ultimate Tetsujin suit make you look like Sigourney Weaver in *Alien* and help you move those shipping containers around the room, but the ultimate Tetsujin suit can truly advance man. How many crippled people could get out of a wheelchair and back to standing in line at the supermarket? Running to catch the bus? That's what Monty Reed of "They Shall Walk" had in mind when he started building walking suits.

Monty was injured in a parachute accident, and the threat of never walking again was very real. Thanks to research and hard work on his part, he did learn to walk again, but the dream of mechanically assisted walking lives on. At last year's RoboGames, Monty took up not only the weightlifting challenge of Tetsujin, but also the walking challenge.

To the untrained eye, it appeared as though Monty was just walking around the arena with a bunch of metal and tubes surrounding him. Yet his stomping and jumping were all done with the suit. The dream is not far off! The mechanics of making a ROBO-one walk and helping the lame walk are essentially the same problems — the major difference being that the bones of the lame become part of the support structure along with the metal, and the motors are outside the skeleton rather than in it.

Ah, but what's this got to do with RoboGames? Well, it's one of the premier events! Broken up into three stages — weightlifting, walking, and dexterity — Tetsujin is *SERVO*'s own robot event.

Much too expensive, you think? Well, Alex Sulkowski — winner of the first $25,000 Tetsujin challenge — bought all the parts for his suit for $1,100. "Many parts were bought cheaply on eBay," says Alex. That's cheaper than the cost of the average middleweight combat robot. Bryan Hood, who was a high school student during the first competition, said "our suit cost roughly $2,000. Because of the simplicity of the suit, there were only three major purchases. These included the pneumatics, the chromoly steel tubing, and an air tank. Clippard Pneumatics sponsored 50% of the pneumatics cost for our team." If a high school student can do it, so can you.

"The hardest thing about building a Tetsujin suit is it requires understanding and integrating multiple disciplines (mechanical engineering, electrical engineering, bio-medical engineering) while being extremely careful so that you do not get hurt," said Alex. "Most of us understand one or two of these, but few are masters at all three. This is where team members who complement your skills come in handy. The most difficult aspect of building a suit is the control system that commands the mechanical suit based on your movement. The three main choices for powering a suit are electrical, pneumatic, and hydraulic. Each one has major pros and cons that impact the control system. On top of that, a failure in the control system can result in major damage to your body."

Bryan adds, "Time, money, and knowledge just about sum it up. As I recall, I spent only about two weeks actually building the suit and several

> “Not only could the ultimate Tetsujin suit make you look like Sigourney Weaver in *Alien* and help you move those shipping containers around the room, but the ultimate Tetsujin suit can truly advance man.”

months planning, researching, and finding parts." Two weeks to build a suit – and here you have six months to get ready for the June 2007 competition.

"As a 17-year-old kid at the time, I really didn't know what I was doing, nor did I have a lot of capital available. All I had was my hobbyist knowledge and whatever calculus and physics I had learned in high school. I had no CAD software so I had to be very careful designing things. The entire project was done in my garage with only basic tools and a Sherline milling machine. I had just about everything working against me and I still believe it was a miracle that the suit worked at all." But it did work! Think what Bryan will be able to do next time.

If you're worried about control, you should be. Both Bryan and Alex make the point that balance and controls are crucial. "Without good controls, it is very difficult to attain balance which is the biggest problem that I faced in the competition," says Brian. "My suit lost balance on several occasions throughout the competition which was a pretty scary experience."

Alex adds, "Most people focus their attention on the mechanics because that is the area they are most comfortable with. They ignore the control system which can be very complex."

"The coolest part of the suit was that the control system read the electrical signals directly from my muscles and controlled the pneumatic suit based on these signals. This resulted in my muscles receiving assistance from the suit 'on demand' (as my muscles required more assistance, the pneumatic system would deliver more assistance to the muscles that required it)."

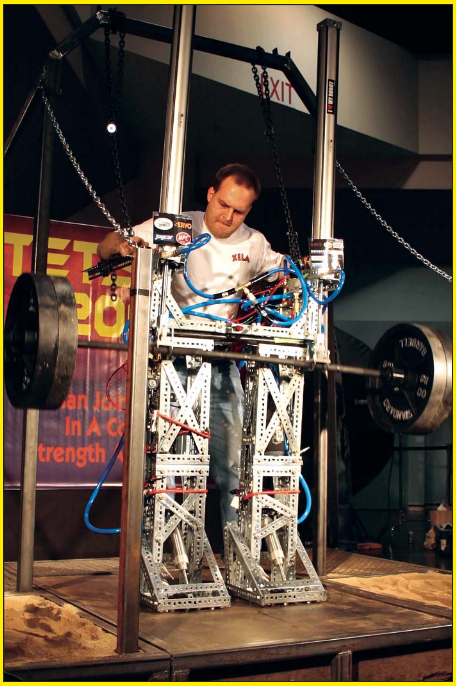
Like all robots, failures are bound to happen while building your suit. Although Alex had tested all of the components of the suit, he did not leave enough time to test all components of the suit together. His pneumatic hose connections developed leaks because they were not properly fastened, and he ended up losing air pressure. This resulted in him not being able to complete the walking competition at RoboGames 2006.

Start simple and enhance as you succeed. As Alex says, "In the original Tetsujin competition and in RoboGames 2006, I talked to many people who were interested in competing. Most of them had grand ideas that sounded really cool but I have never seen one of them work. Start with small successes and build from there."

But just building a suit is success in itself. Bryan says, "to this day, I contribute more and more of my success [in life] to the Tetsujin competition. It was amazing to finally get the suit working the morning of the competition and to get up on the platform in front of hundreds of people to perform the lifts. That was the instant success but it was only a taste of what was to come. As I was only a high school junior at the time of the competition, I

TETSUJIN RULES

- Suit can be up to three meters (9.84 feet) in height and cannot exceed 1.5 meters (4.92 feet) in diameter.
- Challenge 1 — Ascend stairs in your suit to the lifting platform and lift a load of from 100 to 1,000 lbs from a squatting position to a height of at least 24 inches, return the load to the ground in a controlled manner, and descend the stairs. Stair-climbing may be unpowered. The winner is the competitor who lifts the largest weight.
- Challenge 2 — *Dexterity*. Stack nine concrete cylinders weighing ~70 pounds each in a 4-3-2 vertical arrangement. The winner is the competitor who arranges the cylinders in the shortest time.
- Challenge 3 — *Walking Race*. Walk the challenge course carrying a load in the shortest time. A time bonus is granted based on the load carried.
- The operator must be inside the suit in order to operate it.
- The suit must provide sufficient articulation in the powered components to track the movements of the human operator to meet the challenge.
- Movement of the suit's powered components must be initiated by — and track — the movements of the human operator (e.g., to make the suit squat, the operator squats; to lift a load with the suit arms, the operator's arms make the necessary lifting motions). This can be as simple as micro-switches and limit switches or as complex as closed-loop servo control and bio-electric sensors.
- All power for the exoskeleton and any required support systems must be provided by a self-contained system. The system may incorporate remote components connected via a tether, wireless link, etc.
- The tasks(s) must be accomplished by powered elements that are an integral part of the exoskeleton in response to the operator's movements. External devices such as winches, support frames, jacks, etc., are not allowed.
- The exoskeleton must be able to balance and support itself, the operator, and the load without the use of outriggers, auxiliary braces, etc., other than its own legs and feet. A fixed base is allowed for the Cylinder Stacking challenge.
- Stored mechanical energy devices such as springs, torsion bars, gas struts flywheels, etc., are all allowed.
- The exoskeleton shall have a removable power link that disables all operation.



used my Tetsujin experience to help me get an internship at KIVA Systems, a robotics start-up company in MA which is revolutionizing warehouse robotics. I continued working on robotics projects through my senior year. All of this resulted in me getting the Lombardi Scholarship at the University of Florida. This scholarship pays all of my expenses and more, as well as sending me abroad to Mexico, Greece, Japan, and South Africa each summer. The scholarship also gives me a mentor, which will be very helpful for my future robotic endeavors. Now here at the University of Florida, I am working on a double major in Electrical and Mechanical Engineering. My experience in Tetsujin got my foot through the door and the success keeps building on itself. I look back at what I knew then and compare it to what I know now and I realize that I have come such a long way. I don't know how it could have worked out better."

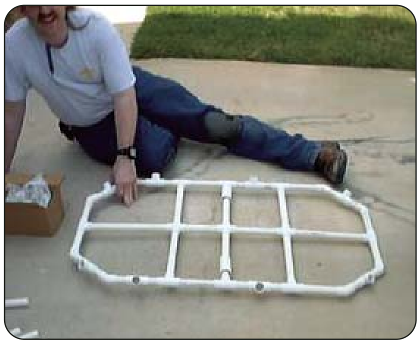
And that's the way it is with all robots you build. Many people start out building them to win a competition, but along the way realize that it's not just about winning — it's about making the best robot you can make. It's about expanding your knowledge of how both mechanics and electronics work.

And of course, winning a medal doesn't suck either. SV

Robotics Showcase

MENAGERIE

Wacky Super Spider



Vern Graner, CNE/CNA/SSE is a Senior Systems Engineer at Texas Information Services. This super spider is just one of many of his builds.

Programmable Logic:

by Gerard Fonte

Dividers and Timers

The goal of this bimonthly column is to provide a basic understanding of the various programmable logic techniques.

There are a lot of powerful low-cost components available today that are rarely considered by hobbyists — and even some engineers — because of unfamiliarity.

You have to be comfortable with the idea and concepts of programmable logic before you will be likely to employ them.

Digital logic depends upon the proper timing relationships between signals for correct operation. In this part, we will examine various techniques for changing the frequency and/or period of a given signal because many designs require one or more clocks with frequencies that are not directly available. These approaches can be used for general frequency synthesis but have wider applications. Additionally, while the approach will be oriented towards programmable logic solutions, these ideas can be implemented with discrete logic, as well.

Frequency Division

The simplest and most basic method for reducing a frequency is with a simple flip-flop. Figure 1 shows this. This circuit divides the incoming frequency by two. Additionally, the output is a perfect square wave. The signal is high for precisely 50% of the time and low for precisely 50% of the time. This is often a useful feature.

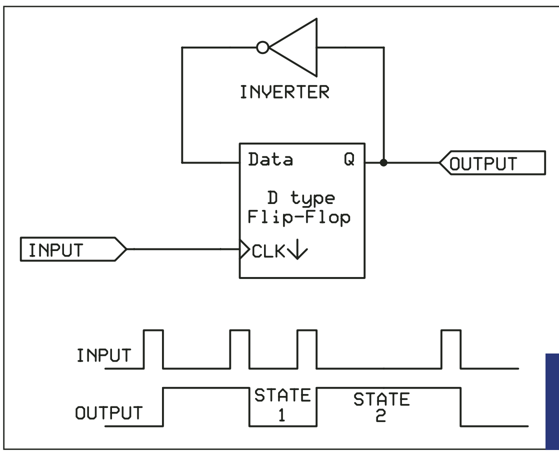


FIGURE 1. The simplest divider is a flip-flop that reduces the input frequency by two.

Note that the frequency is the inverse of the period and the period of a signal is measured in time. Typically, one period of some signal is used as a digital timer. So, by changing the frequency, the timing is also changed. This discussion will concentrate on frequency modification, but it is important to see that this also applies directly to timing.

The circuit shown in Figure 1 is severely limited. It only divides by two. Additional stages can be added to further divide the signal (shown in Figure 2). This is a simple binary ripple counter. Each stage reduces the frequency by a factor of two or precisely one octave. This provides a little more utility but is not really much of an improvement. However, if your main clock is a power of two, this can provide useful and convenient timing intervals. For example, a 32,768 Hz watch crystal will provide precise one-second signals if 15 divide-by-two flip-flops are used. (You might also choose to use a 4.096 MHz crystal for your design instead of the more standard 4.000 MHz crystal for a similar reason. There is no engineering rule that says that you have to make things difficult for yourself.)

A more useful design is the divide-by-N circuit. This will divide by any integer that is less than the maximum value of the counter. Figure 3 shows a four-bit binary counter wired as a divide-by-10. Under normal circumstances, the counter would divide by 16, but the

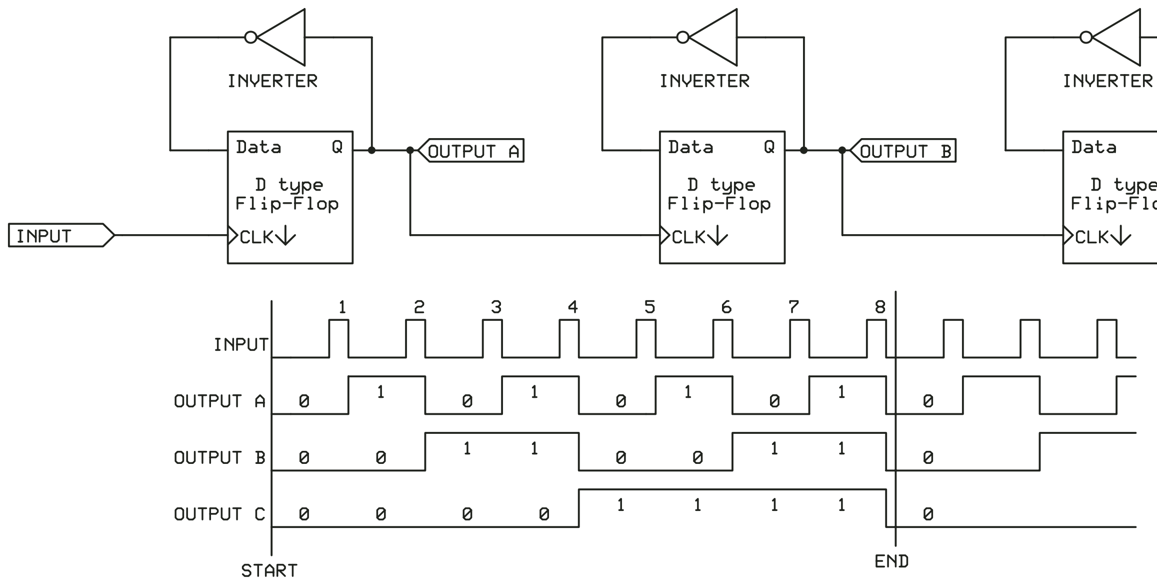


FIGURE 2. Simple divide-by-two circuits can be cascaded for binary division. This is a divide-by-eight circuit.

counter is automatically reset when 10 counts are registered (note that zero is one of the counts). It is also important to realize that many different types of counters can be used here, but the concept is the same. (Refer to the previous article on counter types in this column.)

The operation is straightforward. The circuit counts normally for 10 counts. At the start of the 11th count, the output of the AND gate immediately goes high and resets the counter to 0000. The AND gate output also goes to zero when these bits change. In fact, the output of the AND gate is an extremely narrow pulse that is related to the propagation speed of the circuit. Often, this pulse is less than 10 nS and can be difficult to observe. The pulse is exaggerated in the figure for clarity.

There are several points concerning the decoding of the outputs. An AND gate will function properly without any glitches as long as it is used to reset the ripple-counter to zero. However, an AND gate used to decode state nine of a 16 state ripple-counter *will* show glitches on some state changes beyond nine. (The ripple delay is not as significant a factor as it was for the counters discussed the last time.)

With programmable logic devices, the routing of the decode and reset lines can cause problems. Not all of the bits may be reset properly. In this case, either use a synchronous counter design or else use a separate reset flip-flop that produces a predictable-length pulse.

These divide-by-N circuits can also be cascaded to create any integer division. Figure 4 shows a divide-by-1,005 circuit (with a flip-flop reset). The operation isn't as complicated as it first appears. There are three cascaded divide-by-10 counters that are basically the same as Figure 3.

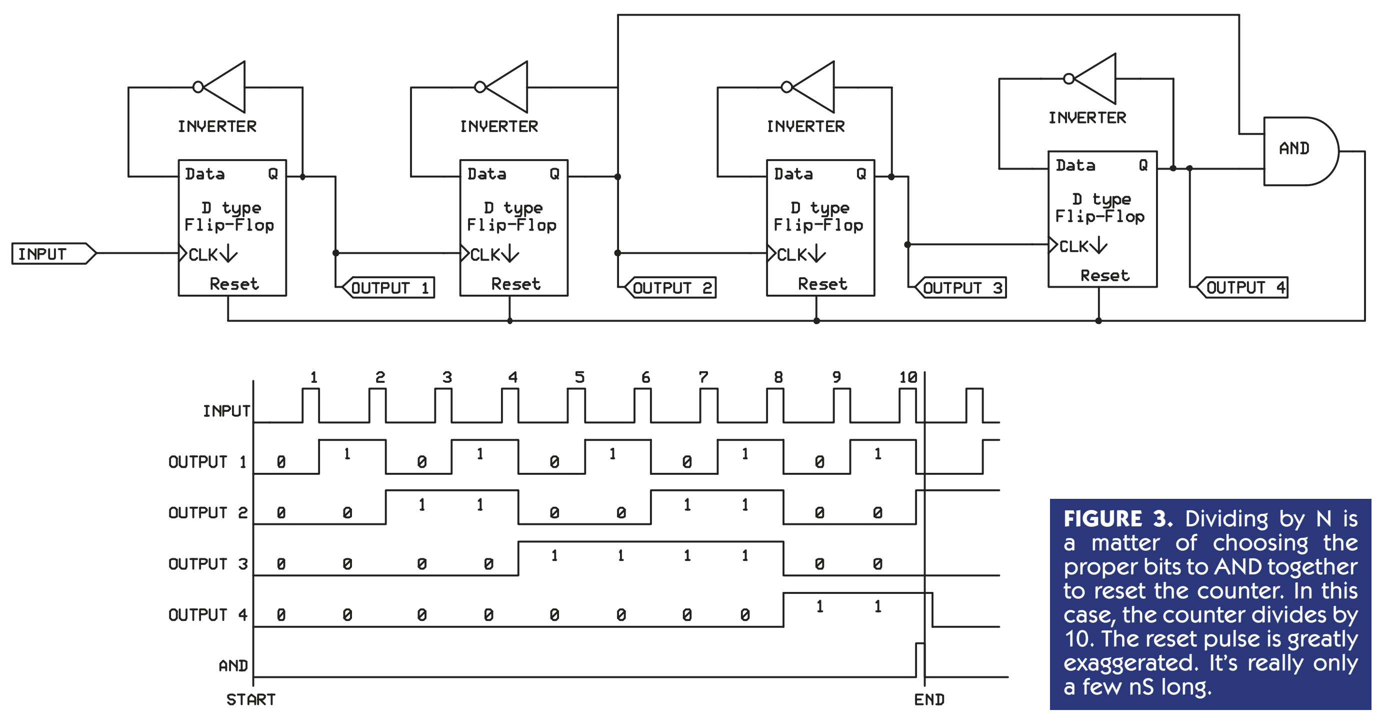


FIGURE 3. Dividing by N is a matter of choosing the proper bits to AND together to reset the counter. In this case, the counter divides by 10. The reset pulse is greatly exaggerated. It's really only a few nS long.

This means that the second counter is actually counting 10s of counts and the third is counting 100s of counts. After 1,000 counts, the reset pulse turns on a flip-flop that enables the main input clock to pass through to a divide-by-five counter. This counts five additional pulses past the already completed 1,000 pulses.

After these five pulses are completed, another flip-flip is turned on. This generates a controlled "master reset" pulse. Since the counter counts on the negative-going edge of the input clock, the reset to this flip-flop is low when it is clocked. It will remain low for half of a full input clock signal. When the main input clock goes high, the master reset pulse goes low because the controlling flip-flop is reset. The master reset also resets all the other counters via the OR gates. (Each divide-by-10 counter could have incorporated its own flip-flop reset, if desired.) Also note that the delay of the circuit (as measured by the master reset signal) is only the divide-by-five circuit. The ripple delays through the divide-by-10 counters are not a factor. Note that the master reset flip-flop and the enable flip-flop are clocked on a rising edge rather than a falling edge.

Unfortunately, the division is a fixed value. The only way to change this is to change the reset decoding. This is certainly possible but it requires additional resources. Figure 5 illustrates a basic programmable divide-by-N circuit. The XNOR (exclusive NOR) gates go high when the "program" input matches the bit output. When all of the XNOR gates are high, the count is correct and the circuit is reset.

There are many variations on this. In particular, a flip-flip reset would be very useful (as shown in Figure 4). Another point is that all of these divide-by-N designs create an asymmetrical output

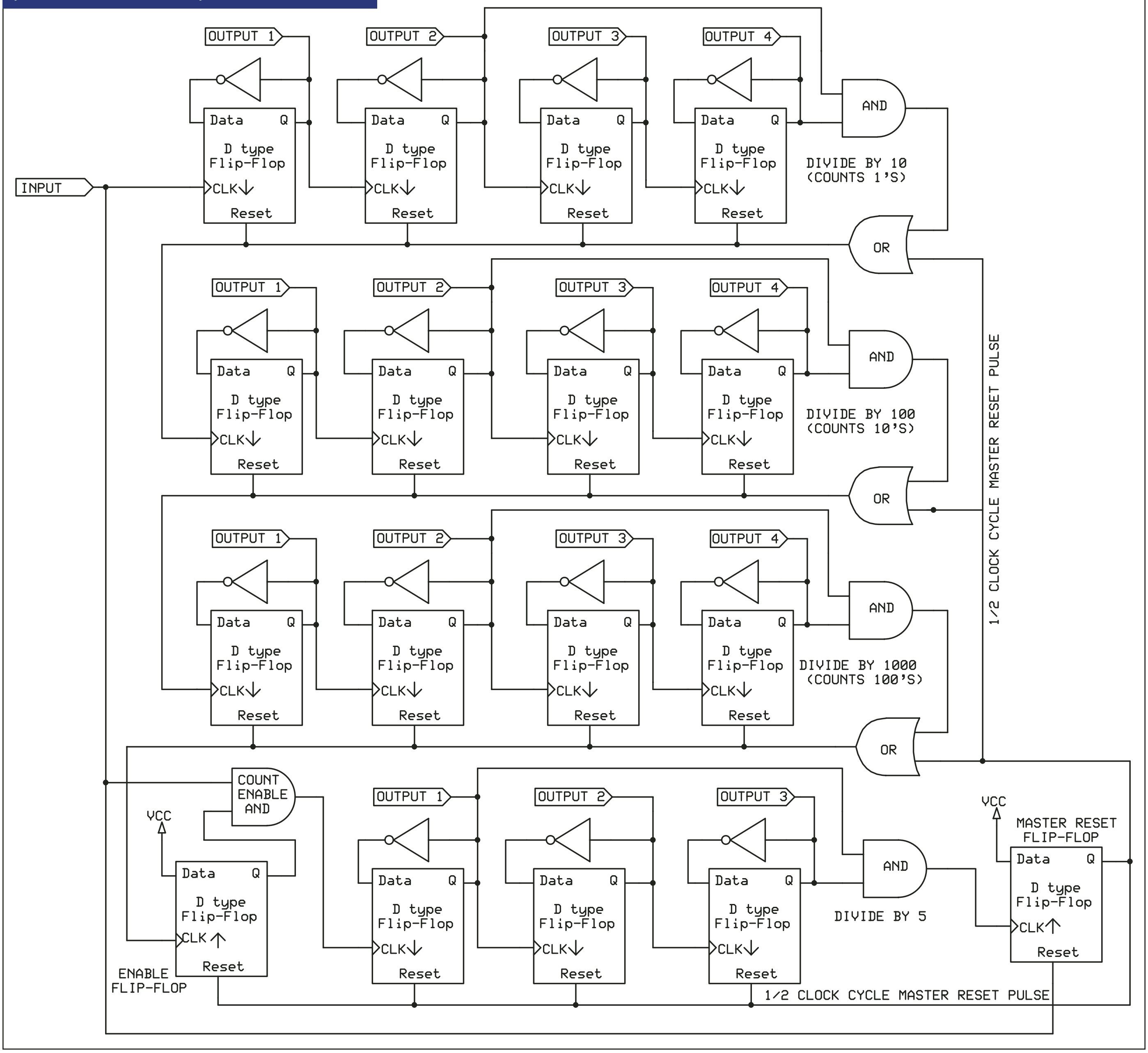


FIGURE 4. This circuit uses three cascaded divide-by-10 circuits to divide by 1,000. When this occurs, a second counter is enabled that counts five more main clock cycles and then resets every flip-flop. The result is a divide-by-1,005 counter. Note that the reset flip-flip provides a half-clock pulse.

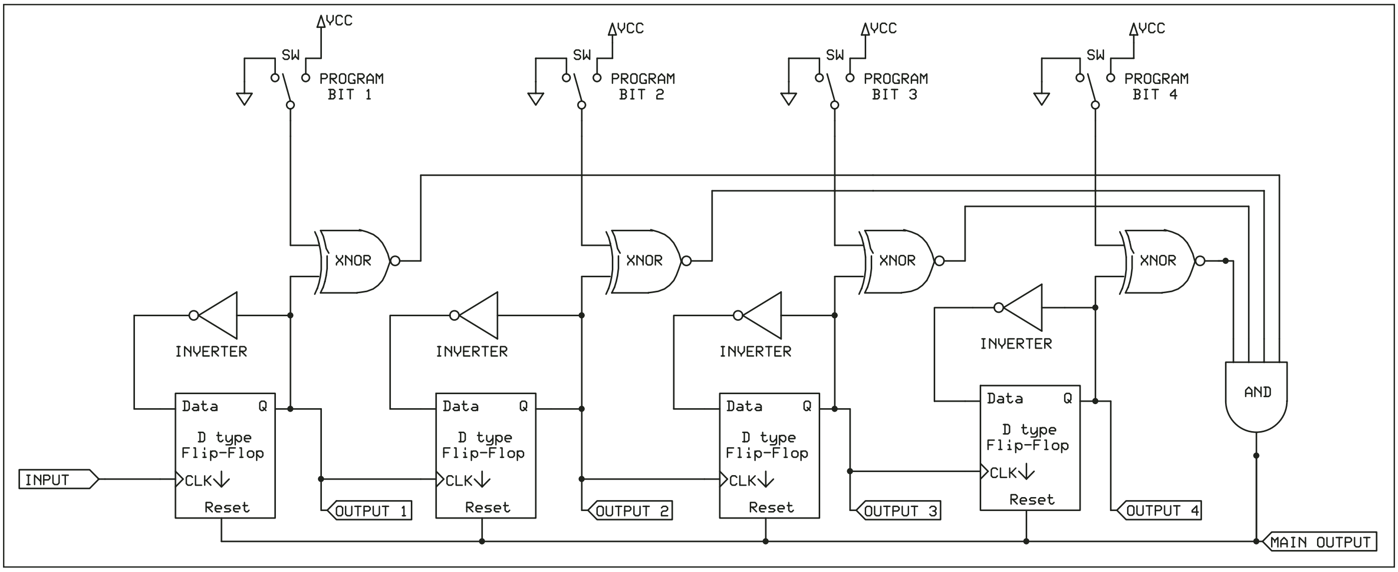


FIGURE 5. A programmable divider uses XNOR gates to match the bit outputs to the desired values. When all the bits match, the counter is reset. Note that the "program" bits do not have to be switches. Other flip-flops or logic circuits could be used.

(except for special cases). Most typically the output is a relatively small pulse. However, often something approaching 50% can be obtained for non-programmable dividers by choosing the proper counter bit as the "output." For example, in the divide-by-10 circuit (Figure 3), bit 3 is on for 40% of the time.

One way to eliminate this is to follow the divide-by-N circuit with a simple flip-flop to square the output (as noted above). This works but has two drawbacks. The first is fairly obvious — the frequency is now reduced by 50%. Thus, the primary division ratio must be changed to compensate for this. The second problem is a bit more subtle. The addition of the squaring flip-flop limits the division values to even numbers. It becomes impossible to provide a final division ratio that is odd.

This makes more sense when examined from a mathematical perspective. These divider circuits have the effect of multiplying the period by an integer. The final "squarer" flip-flop multiplies the final result by two. Therefore, the final result must be even because everything multiplied by two is an even number.

It is possible to create symmetrical waveforms with odd-division ratios but it requires additional resources. Basically, an extra flip-flop alternatively adds and subtracts half of a clock to the output. This evens out the asymmetrical signal and creates a nice square wave.

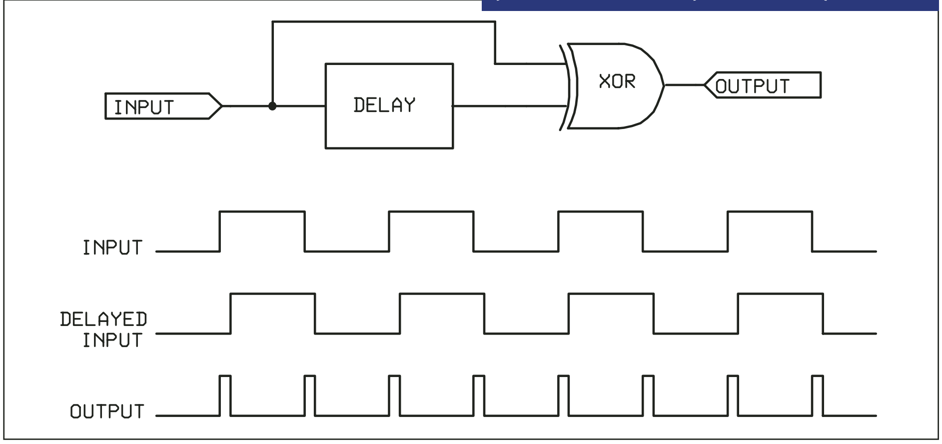
Frequency Multiplication

The typical method for frequency multiplication is with the use of a phase-locked loop (PLL). This is an analog technique that is not usually available for programmable logic. Digital designers tend to shy away from analog PLLs, as well. This is regrettable because the PLL has a lot of nice applications and is not really that difficult to master. Unfortunately, a proper discussion of PLLs is too long to include here. There are other analog frequency multiplication techniques besides PLLs, as well.

The typical method of frequency multiplication for digital designers is to use an XOR gate (exclusive OR). This gate has a high output when the inputs are different and has a low output when the inputs are the same. Figure 6 provides a simple circuit to double an input signal. The operation is fairly clear. The input signal is delayed in some manner. This can be done by using the internal propagation delay through additional gates, internal routing delays, or an external resistor-capacitor (RC) network. The result is a pulse at the start and end of the input signal. The pulse length is exactly equal to the time interval of the delay circuit.

There are a number of issues with this approach. The first is that the output is usually a very narrow pulse of a few ns. This means that high-speed concerns will have to be examined. If a high-speed input signal is used, say 40 MHz or so (the most common situation), the output will often be at a frequency where transmission line factors will become important. Short delay times (about 10 nS) create the equivalent of a 100 MHz

FIGURE 6. An input frequency can be doubled with an XOR gate and a delay. The output will be a pulse at the leading and trailing edges of the input signal. The pulse width will equal the delay.



5 MHz Cycles	3.3333 MHz Cycles	MHz Sum	Average of Cycles	% Error
1	1	8.3333	4.1666	-7.408
2	1	13.3333	4.4444	-1.236
3	1	18.3333	4.5833	+6.172
3	2	21.6666	4.3333	-3.704
4	2	26.6666	4.4444	-1.236
5	2	31.6666	4.5238	+0.5289
5	3	35.0000	4.3570	-3.178
6	3	40.0000	4.4444	-1.236
7	3	45.0000	4.5000	0.0000

TABLE 1. Making 4.5 MHz from 10 MHz using two different dividers. If the average frequency is too low, add a cycle of the higher frequency and vice versa. Eventually, the result will be as close to the proper value as needed. In this case, the exact frequency (4.5000 MHz) is achieved.

signal and again, transmission line considerations will be required. This is not too much of a concern with programmable logic chips where the internal path lengths are very short. However, it is very important with discrete designs.

Another problem with this approach is the variation of the delay circuit. A temperature shift can cause a significant change in the output if an internal delay is used. This is because the internal speed of the gates and routing resources changes considerably with temperature. This is especially true with CMOS circuits. It is possible that an XOR frequency doubling circuit cannot be guaranteed to work over a given temperature range. Operation speed also depends upon the power supply voltage. However, this is usually a second order consideration because most systems today regulate the power quite well. But, if batteries are used, this becomes an issue.

The external RC delay can generally be made to be reasonably stable over temperature. But because of stray capacitance, the delay must be relatively large. External stray capacitance usually varies more from unit to unit rather than with temperature and is often in the 10 pF range. Good layout can reduce this to a few pF. Also, because the delay is now off-chip, it is more susceptible to noise. Of course, the tolerances of these components must be considered too.

Average Frequency Division

Sometimes the instantaneous frequency is not as important as the average frequency. If so, then there is another approach that allows fractional frequency division to arbitrary precision. Suppose you need a 4.5 MHz clock and only have a 10 MHz clock. If you divide 10 MHz by two, you get 5 MHz. If you divide it by three, you get 3.3333 MHz. What you really need is to divide by 2.2222. But it's not obvious how to do that. The solution is to alternate between dividing by two and dividing by three.

The value needed is closer to a division by two than a division by three. If the 10 MHz signal was divided by two twice as often than being divided by three, what would the result be? There would be two 5 MHz cycles and one 3.333 MHz cycle. The average frequency of these four cycles is 4.444 MHz. This is an error of about 1.2%. This is certainly much closer to the desired value than is 5.0 MHz (error of 10%) or 3.3333 MHz (error of 35%). This can be further refined as necessary by adjusting the division ratios as needed. In theory, this approach will allow you to close in on your desired frequency to whatever accuracy you need.

This idea is based on the mathematical principle that any rational number can be expressed by the ratio of two integers. In this case, the fraction is 2.2222. The easiest available integers are two and three. It becomes clear that it is possible to take a certain number of twos and threes such that their average becomes 2.2222.

The quick and dirty way to determine the proper ratio is by trial-and-error with the two division values that bracket the desired value. Start with one of each and determine the average. In this case, 5 MHz and 3.3333 MHz average to 4.166 MHz. This is lower than the target value of 4.5 MHz so add another 5 MHz cycle. Two 5 MHz cycles and one 3.333 cycle average to 4.444 MHz. If this is not accurate enough, continue adding a 5 MHz cycle if the result is below 4.5 MHz and add a 3.333 MHz cycle if the result is above 4.5 MHz. The result will dither around the value you want. Eventually, you will reach a value that is as close as you choose. As you can see from Table 1, an exact solution occurs with seven cycles of 5 MHz and 3 cycles of 3.3333 MHz.

Figure 7 provides a block diagram circuit for the "perfect" solution. I chose to alternate between the different division circuits to distribute the variations over the full

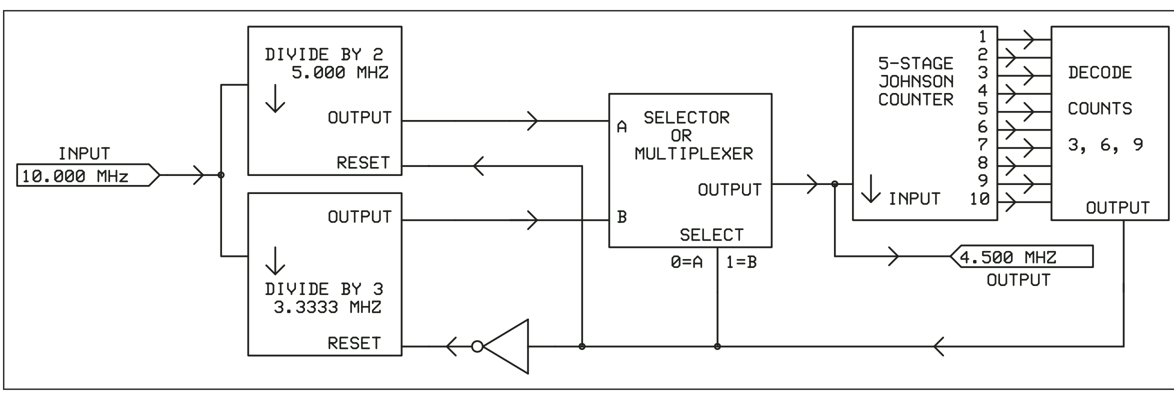


FIGURE 7. With two simple counters, a frequency averaging circuit will allow most any output division ratio. In this case, the input frequency is 10.0000 MHz and the output frequency averages 4.5000 MHz for a division ratio of 2.2222. The output consists of seven 5 MHz cycles and three 3.3333 MHz cycles.

10 cycles. It is certainly possible to use seven consecutive 5 MHZ cycles followed by three consecutive 3.3333 MHz cycles. The design might be marginally easier.

Whichever approach is selected, an additional counter and number of additional gates is needed. With this example, a total of 10 flip-flops is required. This uses five of the Xilinx 3000-series CLBs (Configurable Logic Blocks). But it saves an extra crystal and two capacitors which reduces cost and size and improves the reliability. Additionally, the output is perfectly synchronized with the input clock. There is no concern about frequency drift between the clocks.

You don't have to use divisions by two and three. Any two values can be incorporated as long as one is greater than and one is less than the target. You can use three values or more. However, using the closest integer values creates the least amount of variability in the output signal. And, realistically, this approach works best if the ratios are small. (A ratio of 1,077 to 1,993 may make the output signal perfect, but it's going to be impractical to implement.)

Conclusion

There are a number of ways to change the frequency of a signal. Some are simple and direct, while others are more complicated and subtle. The approaches presented here are not exhaustive but represent the majority of typical applications. Being able to control the frequency and timing of your design is useful and important. **SV**

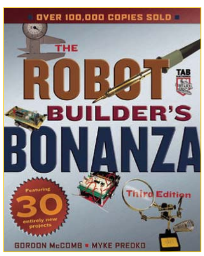
Mind Candy For Today's Roboticist

The *SERVO* Store

Practical Electronics for Inventors
by Paul Scherz

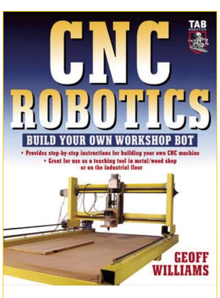
This intuitive, applications-driven guide to electronics for hobbyists, engineers, and students doesn't overload readers with technical detail. Instead, it tells you — and shows you — what basic and advanced electronics parts and components do, and how they work. Chock-full of illustrations, *Practical Electronics for Inventors* offers over 750 hand-drawn images that provide clear, detailed instructions that can help turn theoretical ideas into real-life inventions and gadgets. **$39.95**

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. Projects vary in complexity so everyone from novices to advanced hobbyists will find something of interest. Among the many new editions, this book features 30 completely new projects! **$27.95**

CNC Robotics
by Geoff Williams

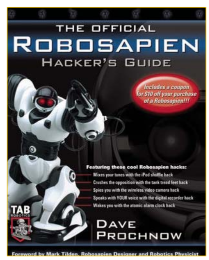


CNC Robotics gives you step-by-step, illustrated directions for designing, constructing, and testing a fully functional CNC robot that saves you 80 percent of the price of an off-the-shelf bot — and that can be customized to suit your purposes exactly, because you designed it. Written by an accomplished workshop bot designer/builder, this book gives you all the information you'll need on CNC robotics! **$34.95**

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**

PIC Basic Projects
by Dogan Ibrahim

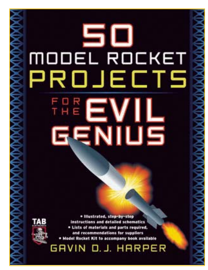
Covering the PIC BASIC and PIC BASIC PRO compilers, *PIC Basic Projects* provides an easy-to-use toolkit for developing applications with PIC BASIC. Numerous simple projects give clear and concrete examples of how PIC BASIC can be used to develop electronics applications, while larger and more advanced projects describe program operation in detail and give useful insights into developing more involved microcontroller applications. **$29.95**

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

LEGO MINDSTORMS NXT Hacker's Guide
by Dave Prochnow

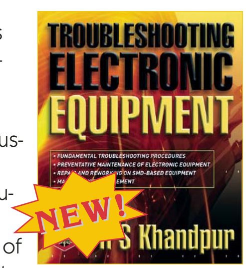
Here is an awesome next-generation collection of LEGO MINDSTORMS projects that enables you to build and program a real working robot in just 30 minutes! New technologies and expanded sensor capabilities make it easier than ever to add a level of sophistication to robotic and architectural creations. This cutting-edge guide describes new advances that make LEGO MINDSTORMS NXT such a great robotics resource. The book explains the all-new NXT intelligent brick ... the interactive servo motors with rotation sensors that align speed for precise control ... the ultrasonic sensor that allows robots to "see" by responding to movement ... the improved light and touch sensors that let robots detect color and feel ... and much more. **$24.95**

50 Model Rocket Projects for the Evil Genius
by Gavin D. J. Harper



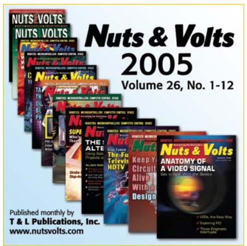
Yes, as a matter of fact, it IS rocket science! And because this book is written for the popular Evil Genius format, it means you can learn about this fascinating and growing hobby while having run creating 50 great projects. You will find a detailed list of materials, sources for parts, schematics, and lots of clear, well-illustrated instructions. **$24.95**

Troubleshooting Electronic Equipment
by R. S. Khandpur



From cell phones to medical instruments to digital and microprocessor based equipment, this hands-on, heavily illustrated guide clearly explains how to troubleshoot, maintain, and repair all types of electrical equipment. The author covers all the essentials such as necessary tools, soldering techniques, testing, fundamental procedures, and mechanical and electrical components. **$49.95**

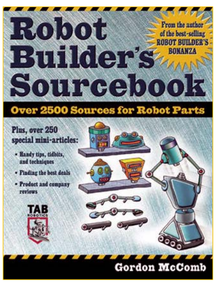
Nuts & Volts CD-Rom



Here's some good news for *Nuts & Volts* readers! Along with all 12 issues of *Nuts & Volts* from the 2004 calendar year, the 2005 issues are now available, as well. These CDs include all of Volumes 25 and 26, issues 1-12, for a total of 24 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 Builder's Sourcebook

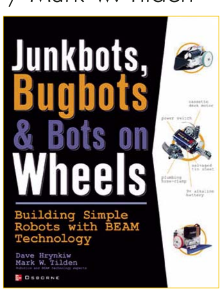
by Gordon McComb



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 clearinghouse of information for that will open 2,500+ new doors and spark almost as many new ideas. **$24.95**

JunkBots, Bugbots, and Bots on Wheels

by Dave Hrynkiw / Mark W. Tilden

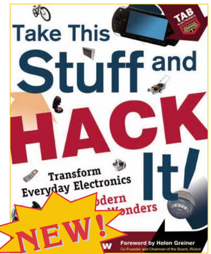


From the publishers of *BattleBots: The Official Guide* comes this do-it-yourself guide to BEAM (Biology, Electronics, Aesthetics, Mechanics) robots. They're cheap, simple, and can be built by beginners in just a few hours, with help from this expert guide complete with full-color photos. Get ready for some dumpster-diving! **$24.99**

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

Take This Stuff and Hack It!

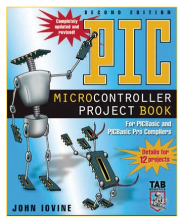
by Dave Prochnow



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**

PIC Microcontroller Project Book

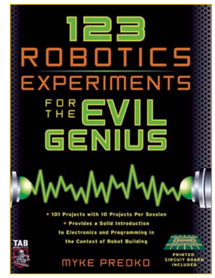
by John Iovine



The PIC microcontroller is enormously popular both in the US and abroad. The first edition of this book was a tremendous success because of that. However, in the four years that have passed since the book was first published, the electronics hobbyist market has become more sophisticated. Many users of the PIC are now comfortable shelling out the $250 for the price of the Professional version of the PIC Basic (the regular version sells for $100). This new edition is fully updated and revised to include detailed directions on using both versions of the microcontroller, with no-nonsense recommendations on which is better served in different situations. **$29.95**

123 Robotics Experiments for the Evil Genius

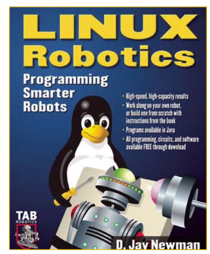
by Myke Predko



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**

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**

$20 each 15% off if you buy two or more.

From HomoSapien to RoboSapien

Before R2D2 there was R1D1

SERVO Magazine T-Shirts

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% pre shrunk cotton.

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

Shopping the Electronics General Store

The general store once formed the cornerstone of every town in the country. A single store provided just about everything: food, clothing, hardware, animal feed — you name it, they had it, or could get it for you. Today, the small town general store has been all but supplanted by large retailers such as WalMart and Target, but the idea is the same: the more the store offers within its four walls, the more likely its customers will stay there and do all their shopping.

In the world of electronics, Internet mail order has taken over the role of the general store for resistors, capacitors, integrated circuits, and other parts for our robots. While no Internet retailer stocks everything, odds are you'll find 80%+ of what you need for your next project in the catalog pages of a single electronics retailer.

In this month's Robotics Resources, we'll take a look at the Internet electronics general store: what's out there, what you can expect to find, and how to get the most out of them.

Start Local, Then Go Global

Before clicking on that mouse, it pays to first check out any stores local to you. Depending on where you live, you may be just down the road from the world's best selection of inexpensive electronic components. In that case, you can save yourself the costs of shipping and get what you want immediately. Check the Yellow Pages under the main Electronics heading.

Most of us, however, aren't so lucky. These days, RadioShack is about all that's left of the national chain store for electronics, and of late, the company has been closing stores and revamping their product lineup as they change their focus.

Fortunately, the Internet has taken up the slack from the dearth of the neighborhood electronics outlet. It's been and will continue to be a tremendous boost to the art and science of robot building. Through the Internet, you can now search for and find the most elusive part for your robot. Just about all of the major surplus and electronics mail order companies provide an online electronic catalog. You can visit the retailer at their website, and either browse their offerings by category, or use a search feature to quickly locate exactly what you want.

And, with the help of Google, Yahoo!, MSN, or other web search engines, you can find items of interest from among the millions of websites throughout the world. Search engines provide a list of possible matches to your search query. You can then visit the web pages to see if they offer what you're looking for.

And remember, there are many auction websites like eBay for buying and selling goodies, including pretty elusive robot parts and kits. If your design requires you to pull the guts out of a certain toy that's no longer made, try finding a used one at a web auction site. The price should be reasonable as long as the toy is not a collector's item.

Consider that the Internet is world wide, and that some of the sites you find may not be located in your country. Though many Internet businesses ship internationally, not all will do so, or the shipping cost may be prohibitive. Read the fine print of the website to determine if the company will ship to your country, and note any specific payment requirements. If a check or money order is accepted, the denomination of the check or money order usually must be in the company's native currency.

Checking Out the Wares

Most Internet retailers use shopping cart software running on their site to allow customers to browse and select items they wish to purchase. No doubt you've used one such shopping cart in the recent past so I won't bore you with the details — except to note that shopping cart software varies from site to site, and not every Internet retailer has an easy-to-navigate store front.

However, just because a retailer has an under-average shopping cart doesn't mean you should avoid them. Some general electronics vendors have great shopping carts, and some are not so great. You'd be surprised what gems you can find in the simplest of shopping carts. Once you've found a site that offers the kind of wares you want, it pays to be patient and wade through the store to find the real values. A few extra minutes of your time may pay off in finding just the right part, or saving dollars off your bill.

The better websites will provide a search feature where you can find specific products. This is doubly important for a general electronics store, as the site may offer tens and even hundreds of thousands of items. Ideally, you will want to find items by part number and also by category.

The search feature at **Digikey.com** — which is one of the largest mail order electronics depots — is a good example. If you know precisely what you want — an Atmel AVR ATTINY13 rated at 10 MHz in an eight-pin DIP package, just enter its standard part number — ATTINY13V-10PU — and the Digi-Key system takes you right to the parts page where you can place your order. Conversely, if you don't know exactly what to order, you can enter a generic keyword or even partial part number. Typing in "microcontroller," for example, will display a list of numerous related categories, including books and CDs, development boards, kits, and, of course, the microcontroller chips themselves.

Not all websites have shopping cart software. Smaller outfits, especially those that deal with the constantly-changing inventory of surplus goods, may simply list what they have on a web page. You contact them via phone or email to arrange a purchase. I've seen a few sites publish a weekly or even daily inventory in PDF format. You can use the search feature in the Adobe Acrobat program to locate items of interest to you.

Know the Details Before You Order

Before placing any order, be sure to read through ordering, shipping, payment, and other terms, especially the company's policy as it relates to minimum order amounts and merchandise return. If all you're looking for is a \$2 part, but the store imposes a \$15 minimum, you'll either want to look elsewhere, or wait until you need to order something else.

Be sure you understand the shipping and handling fees. Some mail order retailers don't charge shipping, but they have to make up for the cost somehow, and usually that's in a higher selling price for the products themselves. When comparing prices from online shops, be sure to include the costs of shipping.

More common is for online retailers to pass on the shipping costs to you as a separate charge. The costs are either calculated by item quantity or by weight, and go up with the more things you buy, or you are charged a set amount, usually by order total. Electronics components can often be shipped very inexpensively, as they are small and lightweight. For very small orders, the parts can be shipped First Class in a padded envelope, usually for about a dollar or two. Parcel orders — via FedEx, US Post Office, UPS, or other shippers — are typically several times this amount, even if you're buying a single 14-pin DIP IC.

If your order is more than \$20, I advise making sure insurance is included in the shipping price. Most shippers include up to \$100 in insurance for parcels at no additional cost. If your order is over that, request additional insurance if the online store doesn't automatically add it in.

Finally, you'll want to ensure the website accepts the kind of payment you want to use. Almost everyone accepts credit cards, though the selection may be limited to Visa and MasterCard. Not all US-based merchants are set up to accept American Express and Discover. Increasingly, web merchants accept PayPal, ePassport, and other online virtual payment options. And, of course, most still accept checks or money orders sent through the mail. When sending a personal check, expect the retailer to hold your order until the check clears.

Is It New or Surplus?

There has always been a thriving market for electronics surplus, parts that someone else — often the government or a large manufacturer — thought they needed, and decided not to use. Good thing, too, as some electronic components can be frightfully expensive when purchased new. Sometimes the components are used (surplus or not), but this is rare for smaller, general-purpose parts. You are more likely to encounter used parts in the amateur radio, antique radio, and similar fields.

It is not uncommon for the average general electronics parts retailer to sell a mix of new and surplus goods. Technically, there is little or no difference, except the surplus parts might be a little older. This could matter if buying something like large electrolytic capacitors, which can leak with age, but in general has no bearing on components like run-of-the-mill signal transistors, integrated circuits, LEDs, resistors, diodes, and so forth.

Lastly, most online retailers mention if a product is surplus. You will want to make note of this, as it could indicate a limited availability. In most cases, when a surplus item sells out, it's gone for good, at least from that vendor. If you are designing a new product that uses surplus components, you'll want to make a "lifetime buy" to ensure you have enough to last you, should there be a shortage down the road.

Sources

Found here are online retailers that supply general electronics parts, such as capacitors, resistors, transistors, ICs, fuses, switches, connectors, soldering irons and other electronics tools, wire, cable, and printed circuit board makings. These stores are your front-line of defense for collecting the electronic components necessary for your robotics projects.

As you might guess, no list of general electronics resources can hope to mention everyone. I have selected from among the companies that have been around the longest, that I am personally familiar with, and/or are regular advertisers in *SERVO* and its sister publication *Nuts & Volts*. You can find even more electronics parts outlets on Google, Yahoo!, MSN, or other search engines. For search terms, you are better off looking for a particular component you're interested in, rather than simply entering "electronics."

All Electronics
www.allelectronics.com

All Electronics is one of the primary sources in the United States for new and used robotics components. Prices and selection are good. Walk-in stores in the Los Angeles area are located in LA and Van Nuys. Product line includes motors, switches, discrete components, semiconductors, LEDs, infrared and CdS sensors, batteries, LCDs, kits, and much more. Specifications sheet for many products are available at the website.

Allied Electronics
www.alliedelec.com

Allied Electronics is a prime source for all electronics (components, chemicals, tools, you name it), in single or multiple quantities. Their prices are often lower than the competition's. Do note the minimum order amount.

Alltronics
www.alltronics.com

Not to be confused with All Electronics in Southern California, this Northern California electronics retailer is known for a good assortment and reasonable prices. New and surplus merchandise.

Arrow Electronics, Inc.
www.arrow.com

Arrow Electronics, Inc. distributes a full line of electronic components to industry. Products can be located by manufacturer, part number, or category.

Avnet, Inc. (Avnet Electronics)
www.avnet.com

Full line distributor with local offices worldwide. Refer to the website for locations.

B.G. Micro
www.bgmicro.com

B.G. Micro is a haven for the electronics tinkerer and robotics enthusiast. Much of the stock is surplus, so it comes and goes, but while it's being offered, it has a good price attached to it. Get it while you can, because someone else surely will.

Brigar Electronics
www.brigarelectronics.com

Brigar Electronics sells electronic components and parts; new and surplus. Excellent selection of electronics and mechanical components for robot builders everywhere.

Circuit Specialists, Inc.
www.web-tronics.com

Circuit Specialists, Inc. are sellers of ICs, active and passive components, test equipment, tools, microcontrollers and programmers, switches, relays, kits, lab trainers, chemicals, and more.

Dick Smith Electronics
www.dse.com.au

Dick Smith Electronics is Australia's biggest electronics retailer, well known for Mr. Smith's head pasted over everything — though he is no longer connected with the company. For a time, DSE also had stores in the United States, but they exited the market during the downturn of the late 1980s and early 1990s. Stores across Oz and New Zealand, and they ship worldwide.

Digi-Key
www.digikey.com

Digi-Key is one of the largest mail order retailers/distributors of electronic components in North America. They offer a very fast and efficient online ordering system, complete with links to datasheets (when available).

Electronic Goldmine
www.goldmine-elec.com

Electronic Goldmine sells new and used electronic components, robot items, electronic project kits, 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

Fair Radio Sales primarily caters to ham operators, with their radio sets and old gear. But they have plenty of test equipment and general surplus electronics to tide anyone over.

Farnell
www.farnell.com

U.K. based electronics mail order. Will ship most anywhere. Offers a complete line of electronics.

Findchips
www.findchips.com

Free search engine for locating parts, by part number. Highly recommended.

Fry's Electronics
www.frys.com

Fry's Electronics is an electronics superstore chain operating primarily in the West Coast (US West Coast). They offer a subset of products via their web page, including electronics parts and kits.

Future Electronics
www.futureelectronics.com

Large electronics distributor. Mostly for industry, but they will also sell to individuals.

Gateway Electronics, Inc.
www.gatewayelex.com

Gateway Electronics is a general electronics mail order and retailer. Among their products are 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.

Hosfelt Electronics
www.hosfelt.com

General electronics. New and surplus.

HSC Electronic Supply
www.halted.com

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

Jameco Electronics
www.jameco.com

Full service general electronics mail order. Jameco carries just about everything you need, and often at a price less than the other guys. Resistors, capacitors, ICs, transistors, diodes, and other active components; connectors and more.

Jaycar Electronics
www.jaycarelectronics.com

Jaycar Electronics carries a wide selection of components, test gear, prototyping, soldering equipment, and more.

JDR Microdevices
www.jdr.com

JDR is a direct marketer of electronic components to hobbyists and the technical engineering communities. Their catalog is skewed towards PC components, but they carry plenty of general electronics (active and passive components, tools, wire and cable, etc.), as well.

JK Electronics
www.jkelectronics.com

JK Electronics carries a complete line of resistors, capacitors, relays, switches, semiconductors, electromechanical, batteries, adhesives, tools, and lots more.

Maplin Electronics
www.maplin.co.uk

Maplin Electronics is a got-it, have-that electronic components super store, based in the UK. Of course they offer the traditional passive and electronics components, relays, connectors, and other prime parts.

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

MPJA sells both new and surplus electronic and mechanical products. Their assortment of such items as motors is fairly small, but they make up for it with a wide selection of other common (and some not-so-common) products.

MCM Electronics
www.mcmelectronics.com

MCM is a mail order retailer of general electronics and repair parts for consumer electronics.

MECI
www.meci.com

New and surplus components, motors, and more.

Mouser Electronics
www.mouser.com

Mouser Electronics is a mail order components distributor, providing a full line of products for industry and hobbyists. All general electronics are carried, including active and passive components, wire and cable, hardware, relays, switches, fans, heatsinks, batteries, component kits, chemicals, and tools.

Newark Electronics
www.newark.com

Prime-component distributor to business and industry; also caters to hobbyist market, but minimum orders may apply. Sells large quantities, when available.

Nu Horizons Electronics Corp.
www.nuhorizons.com

Nu-Horizons is a full-line electronics distributor, offering the major brands — and some not-so-major.

Parts Express
www.partsexpress.com

Parts Express is an all-around electronics retailer, selling everything from sound systems to test equipment, and from stage lighting to electronic components.

Quickar Electronics
www.quickar.com

Surplus electronics, tools. Good selection.

RadioShack
www.radioshack.com

These days, The Shack has fewer components and other electronics items for sale at each store, but they do carry the basics — common value resistors, capacitors, switches, solder, electronics construction tools, that sort of thing. Additional items can be ordered through the RadioShack online store.

Skycraft Parts & Surplus, Inc.
www.skycraftsurplus.com

Skycraft is a veritable surplus mall of accessories, power supplies, transistors, relays, ICs, wire, cable, heat shrink, transformers, motors, fiber optics, test equipment, resistors, diodes, and other goodies.

Surplus Sales of Nebraska
www.surplussales.com

Surplus electronic parts, including connectors, passive and active components, electronics hardware, relays and solenoids, chemicals, and more.

Tri-State Electronics
www.tselectronic.com

Distributor/online e-tailer of passive and active components (resistors, capacitors, transistors, ICs, etc.), batteries, relays, switches, solder and soldering stations, tools, and wire and cable. **SV**

ABOUT THE AUTHOR

*Gordon McComb is the author of the best-selling Robot Builder's Bonanza and Electronics for Dummies. In addition to writing books, he operates a small manufacturing company dedicated to low-cost amateur robotics (**www.budgetrobotics.com**). He can be reached at robots@robotoid.com.*

Take Me to Your Leader

by Lawrence Feir

A robot walks into a bar, orders a drink, and lays down a bill. The bartender says "Hey, we don't serve robots!" and the robot says "No, but someday you will."

Robbie the Robot hit the ground running — let's make that rolling — at the North Carolina Science Teachers Association national meeting in Greensboro. Robbie prepared to take over the convention center armed only with a silver tray of chocolate confections and a silver tongue to match. Robbie made his way down each isle greeting each vender and attendee alike. "Take me to your leader," he would demand in a low mechanical voice. Then after a brief pregnant pause, he would laugh like a fiend and offer the human a sweet treat of Hershey's chocolate kisses much to the delight of one and all.

Robbie is one of the latest creations to come out of my company, Innovation Robotics, in Greensboro. The robot was built as a teaching and entertainment tool. Robbie weighs 125 lbs. He measures in at just over five feet in height and has a top speed of five mph. The robot's mid-section is lighted by 160 LEDs and his entire midriff revolves around his "waist." His arms, wrists, and hands are fully articulated and are powered by pneumatic cylinders running regulated carbon dioxide gas. The limbs are controlled through a gang of solenoid valves interfaced with a standard JR radio control unit.

Robbie the Robot.

Robbie's voice comes from a speaker mounted in the robot's head. A hacked cordless phone allows two-way communication between the robot and the operator. When the operator speaks into the handset, the sound is emitted from the robot. When the robot is spoken to, a microphone in the robot's head picks up the sound and transmits it back to the operator. If the operator is clever, the audience hasn't a clue why the robot is so smart. Robbie asks questions and gives real answers. He comments on your T-shirt and the color of your shoes. He's even been known to flirt with the ladies by complimenting them on their exquisite taste in fashion.

Future improvements will incorporate a PLC (programmable logic controller) to allow complex motion routines at the touch of a button. For example, one flip of switch on the robot's transmitter would command the robot to move his head up and down and side to side as though laughing and gesturing while his arm and hand reach down and pick up an object from the ground and offer it to the audience. Robbie is also being equipped with a device that will allow him to offer a business card of advertisement to passers by.

Reaction to Robbie has been positive and interesting. He has a 1950's retro look and is non-threatening to even the youngest children. He serves hors d'oeuvres with a certain definitive panache that is sure to be remembered. He puts a smile on every face.

Robbie is being developed as a powerful marketing tool for high tech companies who see the need for interactive advertising. Essentially, he is a rolling advertisement, as well as a mechanical ambassador for the company he is working for.

Robbie is available to host your corporate event, tradeshow, or advertising venue on a contract basis. SV

Then and NO

SHUTTLE REMOTE MANIPULATOR SYSTEM/CANADARM

by Tom Carroll

Without a doubt, the most expensive robot system in the world is NASA's Remote Manipulator System used on the space shuttle and space station and made by a Canadian company — Spar Aerospace. Yet, by many people's account, it is not a robot at all, but a very expensive teleoperator.

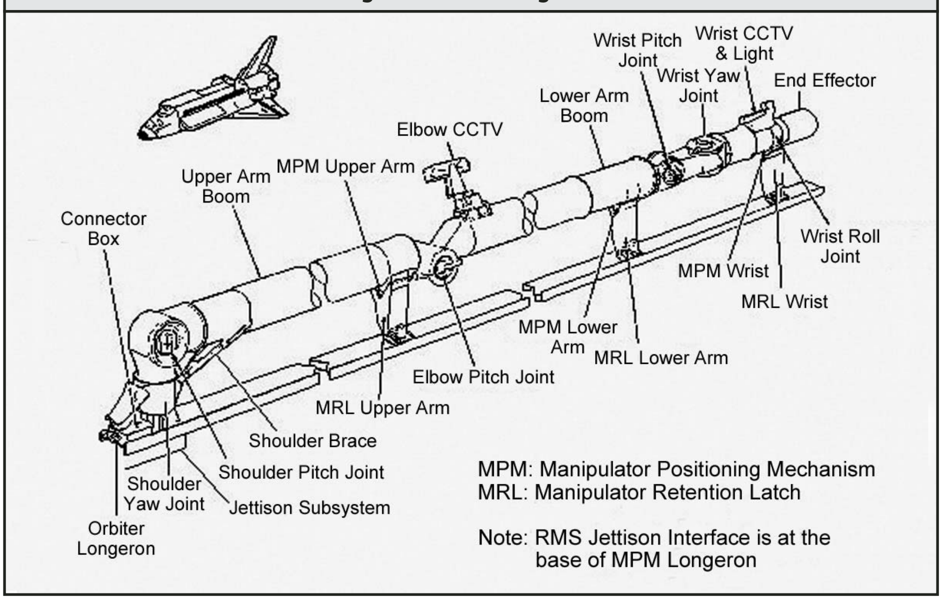
Who really cares about the fine details and definition of this amazing piece of space hardware as it has accomplished many 'impossible' tasks in a most inhospitable place — space. It holds a special place in my heart as I spent a very interesting five years exploring its use for Rockwell's Space Station Program.

As Robotics Systems Lead Engineer for Rockwell's Space Station Systems Division, I was involved with implementing the already-developed Canadarm Remote Manipulator System into our various designs of the Space Station. The RMS arm is 50'3" long and 15" in diameter and has six degrees of freedom. The arm itself weighs 905 pounds and the total system (including the control panel) weighs 994 pounds (see Figure 1). The original Remote Manipulator System or RMS has six axes of freedom, or joints that closely match the joints of the human arm, with shoulder yaw and pitch joints; an elbow pitch joint; and wrist pitch, yaw, and roll joints. Pilots know these terms: 'roll' is a twisting motion; 'pitch' is the end rising up or moving down; and 'yaw' is turning from side to side. The end effector is the cylinder at the end of the wrist that actually grabs the payload. The two boom sections are called the upper and lower arms. The upper boom is between the shoulder and elbow joints, and the lower boom is between the elbow and wrist joints. The RMS arm attaches to the space shuttle's payload bay 'longeron' (upper sill or edge of the payload bay) at the shoulder positioning mechanism. The diagram in Figure 2 shows the RMS components and their attachment to the space shuttle.

You might wonder why have such a heavy, complex, and expensive piece of machinery to manipulate objects in space. Yes, heavy payloads can be manipulated by astronauts in space suits in the zero gravity realm, but these payloads still have the same mass as they do on Earth. Considering that all space payloads cost millions if not billions of dollars, they must be handled

Figure 1. Shuttle RMS.

Figure 2. RMS Diagram.



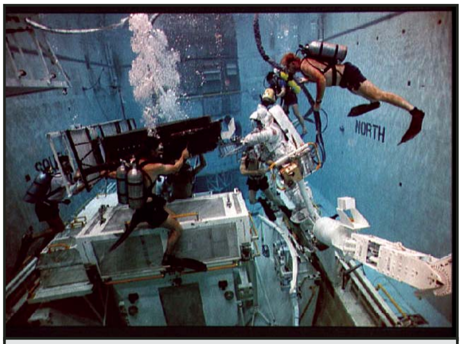


Figure 4. Astronaut in WETF.

very carefully. NASA drew up a detailed list of requirements for a system to handle payloads on the space shuttle. Having heard a bit about some of Canada's aerospace expertise, the then NASA Administrator, Thomas Paine went to our northern neighbors in 1974 to ask for their assistance in building part of the entirely-new space transportation system based on a reusable space shuttle. Little did NASA, we at Rockwell, or our Canadian counterparts realize just how expensive something reusable would end up costing to develop and build.

The Canadian government worked with the National Research Council to determine the best approach. Spar Aerospace Ltd., a Canadian company based in Weston, Ontario, was chosen after determining that the manipulator arm system was the most appropriate part of the shuttle project to undertake. Spar designed, developed, tested, and built the RMS after winning the NASA contract. The terms of the contract specified that Canada would pay for the development and construction of the initial Canadarm flight hardware. NASA also contracted to buy at least three more Canadarms as part of the deal. Also in the contract was stated that Canada would have preferential access to

Figure 5. End Effector.

Figure 3. RMS on Air Bearing Floor.

shuttle launches as part of its developing space program. Spar was later acquired by Richmond, B.C. Canada-based MacDonald Dettwiler & Associates Ltd. in 1999 and formed into MD Robotics.

Spar Aerospace's Difficult Task

One might assume that a large robot arm would be fairly simple to design and build. Take a couple of light weight carbon fiber composite tubes 23 feet long and attach a two axis shoulder joint, a single axis elbow joint, and a three axis wrist/end effector, attach it to a good location on the shuttle, and you have a space robot arm. Toss in a bit of industrial robot technology and couple that with a 'typical' crane and you're set to go. Right!

Gravity, or the lack thereof, a vacuum, and extreme temperatures are just a few of the obstacles that make the design and construction of a 50-foot robot for space tasks difficult. One might think that the zero gravity environment would allow the program to use a weaker type of system. That's true, to a point. On Earth, the arm cannot even lift itself off the floor, but the shuttle's RMS cannot use the usual several hundred horsepower diesel engine that you might find on a 'cherry picker' or solid boom crane, or even the equivalent with an electric motor. A winch cable arrangement can't be used, as there is no gravity pulling 'down' on a boom to keep the cables taut.

NASA had to use a jointed stiff 'arm' with powered joints. Hydraulics or pneumatics certainly could not be used. The joint mechanisms had to be quite precise with virtually no gear slop. A one-degree slop in the shoulder joint could cause a 10-1/2 inch loss of accuracy at the end effector. This is unacceptable when the positioning of payloads in space sometimes require sub-millimeter movement in three axes. Unlike on the ground where gravity can hold a boom in one position, the micro-gravity space environment would allow a RMS boom with mechanical slop to bounce back and forth like a clock's pendulum.

Precision and Reliability

Gear slop is sometimes minimized or eliminated by using two gears that act as a single gear. The two gears are spring-loaded so they push in opposite directions and effectively enlarge each tooth just enough so that the gear that is driving them sees no slop at all. This cannot be used on the RMS arm, as the payload mass is far too large, so precision machining was required. NASA specified type 302 stainless steel alloy as the material for the gears, but, when the machined gears were heat-treated for strength, they shrank a few micrometers — too much to pass inspection. Since the gears were too hard to be machined after heat-treating, Spar had to calculate the shrinkage and machine them a bit larger to fit specs after the treating. It worked. Lubrication of mechanisms in a vacuum under extreme heat and cold conditions was also solved.

After solving literally thousands of similar problems, Spar had to then test the RMS system under predicted space conditions. Temperature extremes, vibration, duty cycles, and even excess gravity loading are easy to simulate on the ground, but not the lack of gravity. Despite what some SciFi movies may show, there is no way to produce zero or micro-gravity on Earth. The RMS arm is too weak to even lift itself off the floor in Earth's gravity so zero G must be attained in some manner for successful testing. You can take NASA's converted KC135 'Vomit Comet' tanker and simulate 25 second bursts of zero gravity in parabolic dives (followed by 35 seconds of 2 Gs — over and over again), or go to space.

Since the latter was too expensive and the other offered test periods that were too short, Spar used what several aerospace companies and NASA have used for years — an air bearing floor. We did this at Rockwell when we tested two spacecraft closing in to dock in space. Think of an air hockey table in reverse. Instead of air entering from holes in a flat floor to allow pucks to float about on a cushion of air, drive air into the pucks themselves. Figure 3 shows the Canadarm being tested on an extremely flat floor, gliding on a series of pads with a flow of nitrogen gas into them, much like a bunch of hover cars. Only a single axis can be tested at a time on a single plane, not three planes as in space, but repositioning the RMS test item to the new plane can effectively test all axes of motion. Much of the $100 million initial cost of the RMS arm went into this type of testing.

After the RMS was handed over to NASA and Rockwell to install on the Space Shuttle, the only true test ahead was a live test on the shuttle itself. Astronauts had several systems set up to practice future space operations on at Johnson Space Center in Houston, TX. The Weightless Environment Training Facility (WETF) in Building 29 — also known as the Neutral Buoyancy Lab — allowed astronauts to float about in water to simulate weightlessness, and an RMS simulator was sometimes placed there also for them to interact with simulated payloads, such as the Hubble Space Telescope.

Figure 4 shows an astronaut attached to the end of the simulated RMS in the WETF tank being assisted by six divers as he practices with a payload. This tank works best when the astronauts are practicing EVAs, but the Building 9 High Bay Shuttle RMS Trainer is what all prospective payload/mission specialists must spend many hours upon honing their mission's skills.

Trying to Snare a Payload Simulator

As the engineer in charge of developing manipulator systems for Rockwell's bid on the Space Station, I frequently flew to Johnson Space Center to interface with my NASA counterparts. I was fortunate to have Judith Resnik — one of the astronauts who died in the Challenger disaster several months later — as my instructor on the RMS trainer in Building 9 one morning. I certainly wasn't overly confident about my abilities with robotic manipulators but I did have a bit of self-assuredness about trying my skills on the trainer. I had programmed several different industrial robots in my day so this big arm should be no problem. Wrong! Though the trainer arm was nothing like the shuttle arm except for a similar appearance, it did operate in a similar manner. I had spent the previous night reviewing an operator's manual of sorts. To operate on Earth, this arm was far more robust and powerful than the shuttle's RMS.

The RMS End Effector

My task was to snare a simulated payload that was nothing more than a satellite-shaped cloth-covered helium balloon with a post grapple fixture on the side that looked a lot like a large nail head sticking out of a round disk. The end effector was a unique design that used three heavy steel wires that surrounded the head of the nail and were pulled around it much like the iris petals in a camera. Figures 6a through Figure 6e show the sequence of snaring and capturing a payload's grapple fixture. All I had to do was shove the end effector's mouth over the grapple fixture's nail head (a and b), rotate the inner ring (c and d) until the three wires were snug on the head, and draw the rings and wires down into the end effector (e) until the grapple fixture was snug against the rim of the end effector. Easy. Not.

The payload was the same size as a typical satellite but had only a few pounds of mass so I always managed to knock it with the end effector so it bounced away. I had two sets of windows and TV monitors to watch the action through, yet it was very difficult. Fortunately the payload was tethered so I finally managed to snare the sucker. Judy smiled at my frustration but I'm sure she snickered at me under her breath and was ready to take control if I went astray.

Figure 7 shows a fake astronaut on the end of the training RMS arm in Building 9. Notice the difference in the real RMS, the WETF variety, and the Building 9 trainer arm.

Placing the RMS on the Space Station

My task was to implement Spar RMS designs for the Space Station and they were already at work with their newer configurations. The SS is huge — larger than a football field — so a single RMS, even with a several hundred foot long arm, was impractical. The only solution

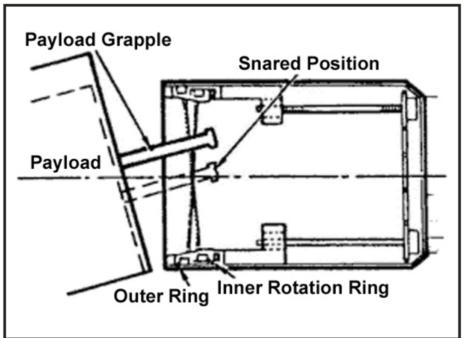


Figure 6a

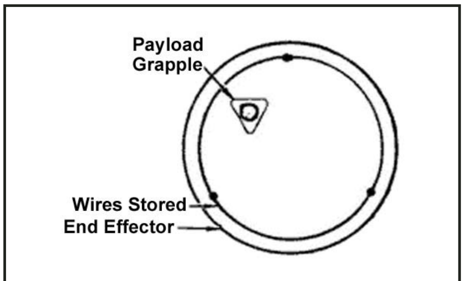


Figure 6b

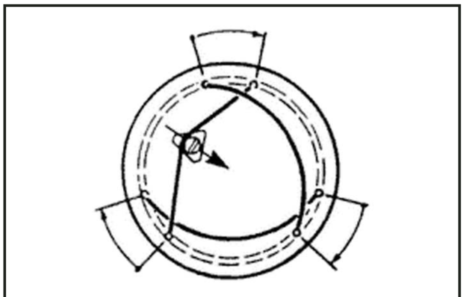


Figure 6c

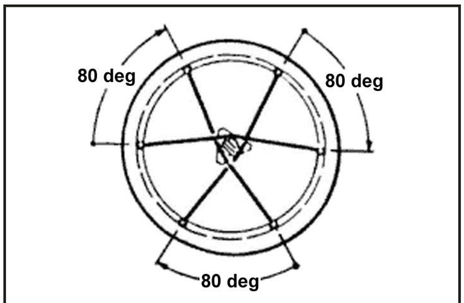


Figure 6d

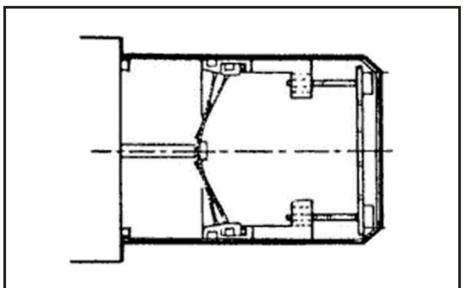


Figure 6e

Figure 7. Fake Astronaut on RMS.

was to be able to move a specially modified RMS arm about the SS. Brian Fuller and Tom Young of Spar supplied me with RMS technical data and our group developed a Mobile Transporter to carry the space station version of the RMS over all the faces of the space station structure. We had to keep the weight to a minimum so we decided not to use an attached rail system all over the SS.

The design that we came up with used a series of doorknob-sized pins all about the SS structural surface. The mobile transporter used two sets of forks to grasp the knobs and pull itself along the surface. Figure 8 is a model of the MT that I'm holding and shows the fork assemblies and the two slotted rails through which the knobs traveled as it pulled itself along like a person climbing a ladder. The MT could turn at a node, rotate 90° to a new surface by means of swivel joints in the middle of the two rails, and carry the RMS while it was attached to a large payload. The white box contains a series of nickel-hydrogen batteries, the control electronics, and the fork actuators.

Figure 8. The Author and His Mobile Transporter Module.

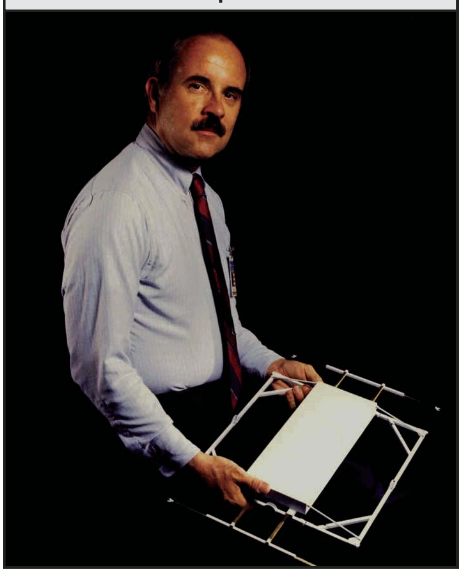


Figure 9. SSRMS with Manipulator.

The Canadarm 2 Space Station RMS

The Space Station is quite a bit further along since my days on the program. The new 58 foot, 7 degree-of-freedom Canadarm 2 Space Station RMS was installed on the SS in April of 2001 (see Figure 9). The new system is self-relocatable (it can relocate itself anywhere on the SS) and is the key component of the $896 million Mobile Servicing System (MSS).

The MSS is composed of two other components besides the SSRMS — The Special Purpose Dexterous Manipulator (SPDM) and the Mobile Remote Servicer Base System (MBS). The SPDM is a small, advanced detachable two-armed robot that can be placed on the end of the RMS arm to install and remove small payloads and similar delicate operations. The (MBS) is a movable platform for the Canadarm2 and the SPDM that slides along rails on the Space Station's main structure to transport Canadarm2 to various points on the Station.

A lot of progress has been made on these space systems since I worked on the program in the late '80s and early '90s. The International Space Station is a reality and can be seen by the naked eye as it crosses the sky. I truly enjoyed my small part in developing robotic systems for space. As I always say, use your favorite search engine to keep abreast of the Space Station's robotic developments.

The author can be reached at TWCarroll@aol.com and welcomes questions and comments. **SV**

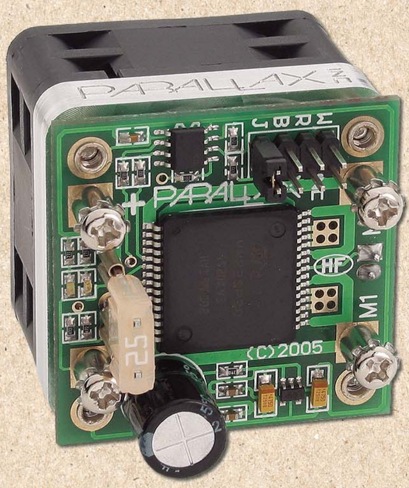
Advertiser Index


THESE ARE THE REAL DEAL
SerialLCD™
2X16 and 4X20
Serial Displays
RS232 and TTL Inputs
9600/2400 Baud
254 Backlight levels
RoHS Compliant
Input Serial Data
Normal/Inverted
Outputs for
Buzzer or Relay
Easy to use!
Priced from $34.95
Complete info
at BasicX.com
COLOR MAKES THE ORDINARY... EXTRAORDINARY!
NetMedia
NetMedia Inc., 10940 N Stallard Pl, Tucson Arizona 85737 tel.520.544.4567 fax.520.544.0800 www.BasicX.com

HB-25

Motor Controller from Parallax

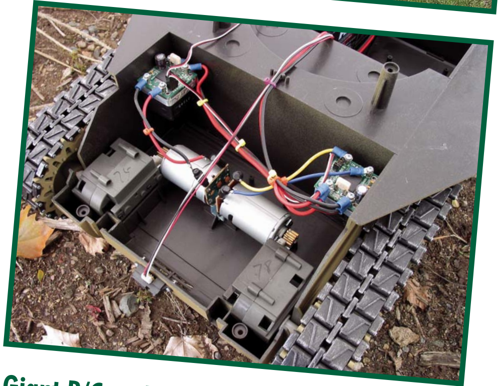
Manages Big Loads!



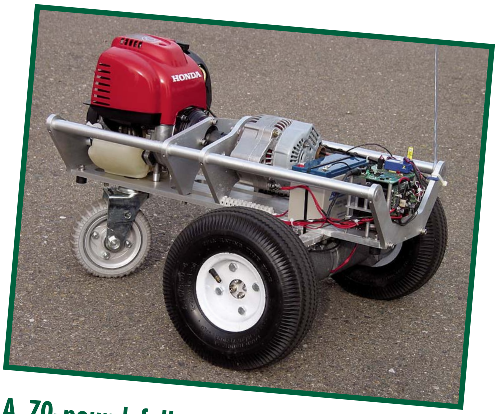
Get an HB-25 Two-pack (#29150) for only $79.95 *and save $20!* The Parallax HB-25 Motor Controller (#29144; $49.95) provides a simple and dependable motor control system for your motorized projects. With an efficiently integrated heat dissipation design, the HB-25 easily manages 12V motors drawing 25A of current! These are sizable motors, as shown in our two example robots which are both controlled by HB-25s.

Motor Size:	0.5 HP Max - No Minimum
Motor Supply:	6.0 vdc min - 16.0 vdc max
Load Current:	25 A Continuous 35 A Surge (13.8 V)
Standby Current:	50 mA @6 V 80 mA @13.8 V (fan on)
PWM Frequency:	9.2 kHz
Pulse Input:	1.0ms Full Reverse, 1.5ms Neutral (off), 2.0ms Full Forward
# of Motors:	1 (or 2 in same direction)
Protection Circuits:	Over Voltage, Over Current, Over Temp.
Indicators:	Power (green), Fault (red)
Cooling:	Forced Air - Ball Bearing Fan
Terminals:	Screw Post with 35 A Rating
Weight:	2.5 oz (71 grams)
Size:	1.6″ x 1.6″ x 1.9″
Mounting:	(2) 6-32 screws on .800″ centers
Included:	(1) 25 A Fuse

HB-25's in Action!



Giant R/C tank from a toy store provides an excellent autonomous robot platform.



A 70-pound full custom hybrid robot uses the power of a gas engine combined with an alternator and battery for easy control.

Order the HB-25 online at www.parallax.com or call the Parallax Sales Department toll-free 888-512-1024 (M-F, 7am-5pm, PT).

PARALLAX

www.parallax.com