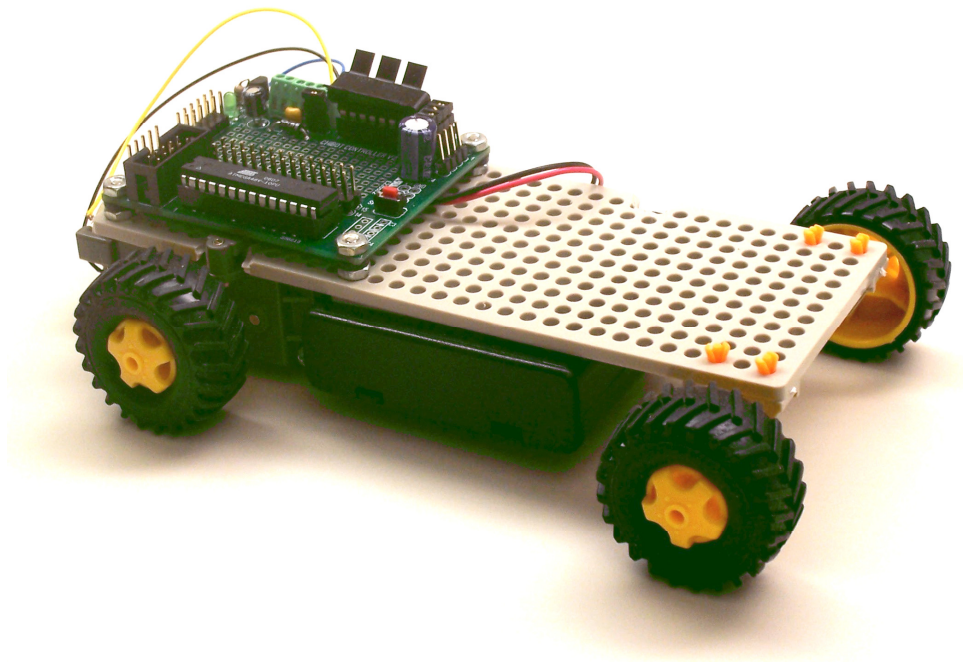


TTBot - Table Top Bot

Construction Manual
Version 1.5 March 23, 2008

Congratulations on your new TableTop Bot kit. This is one of the easiest and quickest ways to get into the field of robotics, whether you wish to just have a fun hobby, or you would like to go on to work professionally in robotics or industrial controls.



Written By: Art G. Granzeier III
Granzeier Consulting

For: Wright Hobbies Robotics

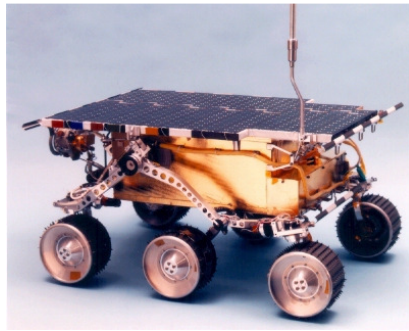
Table of Contents

Short Introduction to Robotics.....	5
Introduction to the TTBot	7
Getting Ready	9
Introduction to the Kit.....	11
The Brains of the Family	17
The Downloader Cable	19
The Controller Board	23
Programming Your Controller.....	33
Let's Hire Some Muscle	39
Sample Programs	57
Future Projects	59
Troubleshooting	61

Short Introduction to Robotics

Your TableTop Bot has a lot in common with other robots such as NASA's Sojourner which explored Mars for twelve times it's expected life span. The TTBot even shares some characteristics with fictional robots like R2D2 and C3PO from the Star Wars series.

All robots have sensors, manipulators, a power system and a control system (the robot's brain.) The sensors allow the robot to "see" different things in its environment. Not all robots will have actual vision sensors, actually most robots do not have true vision systems. Common sensors used in hobby robotics are more like the simple light sensors, or sonar distance sensors or even something as simple as bumper switches, which let the controller know that the 'bot has just hit something. Without some way of sensing things around it, the robot would not be able to interact with it's surrounding.



Sojourner



NASA's Robonaut Unit A

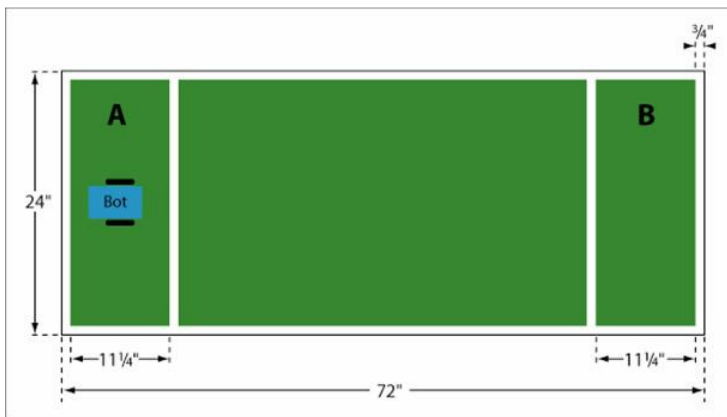
Likewise, your robot will also need a way of doing something. A robot which can sit there and sense everything around it, but can't do anything would be an extremely boring thing. Thus, a robot also has motors, servos, actuators and other ways of moving itself or the things around it.

Of course, without a controller, the sensors and manipulators would not have any way of getting anything done. The controller is the brain which takes the input from the sensors and makes the actuators do their work to have the robot interact with it's environment.

The power system is what provides the energy required by all of the other systems. If you think about it, your TableTop Bot even shares a lot of characteristics with animals; these things we just covered are also characteristics of most living animals.

Introduction to the TTBot

This robot kit was created to provide an easy way for people to get into the Chicago Area Robotics Group's There-and-Back contest. ChiBots has been working to encourage interest in robotics since the late 1990's. This contest is one of many that the Group runs on a routine basis and is a simple race where a robot starts at one end of a course and runs as fast as possible to the other end of the course and then reverses itself and runs back to the start. There is no steering or other more advanced features needed to get started in this competition. Because of how easy the There-and-Back contest is, we will use that competition for most of our programming to start out, and then we will touch on some other contests as more advanced projects.



The basic There-And-Back course

As you can see from this diagram, you will begin the contest by placing your robot in the section labeled A and then your 'bot must travel the length of the course to section B and then back to where it began.

The table below shows how one of these contests is graded. The speed is important, but don't overlook that you get 20 points just by having your robot

complete the course. Another thing to remember is the last category. Many roboticists are so concerned with getting their creation to work that they forget about aesthetics. Just by having their robot look cool or pretty, they can get over 10% of the maximum possible points.

Description	Point Assignment
Shortest time to complete the course	1st place: 30 2nd place: 20 3rd place: 10
Starts on remote command (sound, infrared, radio, etc)	10
Reaches Zone B	10
Returns to Zone A	10
Navigational skill — Zone A Bonus	0 - 20
Entertainment value	10

Basic Table Top Score Determination

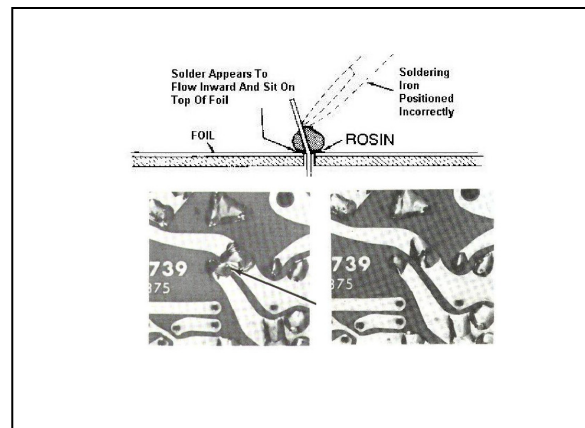
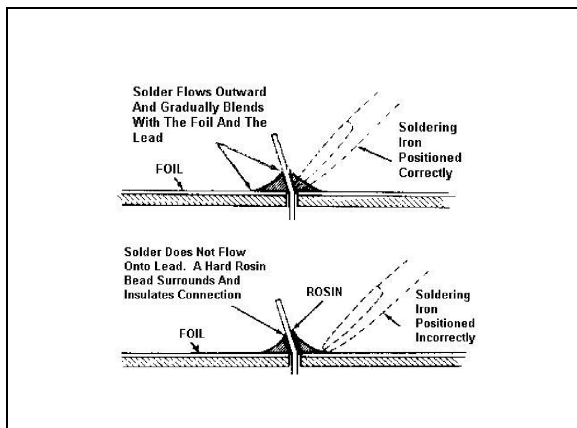
For a complete list of the rules, check out the TableTop Bot contest page at:
<http://www.chibots.org/drupal/?q=node/27>

Getting Ready

Soldering Refresher

Soldering is the process of connecting two or more electrical connectors or wires together and then melting a low-temperature metal, called solder, to hold the connection together. There are two very important things to remember when soldering: first you must have a good mechanical connection, and second you must have a good electrical connection. The first point means that you must make sure that the wires are physically connected before you begin soldering the joint. If you are working on a printed circuit board, the electrical connection of melting the solder will actually provide the mechanical connection.

The easiest way to ensure that you get a good solder joint is to start out by ensuring that the point of the soldering iron is touching both of the conductors that you wish to solder. Heat up both pieces of metal and then place the solder against the conductors and let it melt onto both conductors.



For more information on soldering, see www.wrighthobbies.net/guides/circ1.htm

Basic and Digital Electronics

It is beyond the scope of this instruction manual to give a course on Basic Electronics or Digital Electronics. We are trying to give you the information to make use of the TableTop Bot kit as an introduction to the field of robotics. If you would like more information on these fields, there is a huge amount of information on the Internet. For a good site with great practical information on these subjects (and more), please go to www.play-hookey.com. This site will provide you with many, many hours of practical edutainment.

For the best training on robotics, however, always remember the *Three Laws of Robot Builders*:

1. Build that robot.
2. Build that robot.
3. Build that robot.

Thanks to Marvin Green of PARTS (Portland Area RoboTics Society) for those Three Laws in Issue 2 of the PARTS Newsletter (www.portlandrobotics.org/uploads/media/issue02.pdf). With that in mind, let's move on to our robot itself.

The Major Components

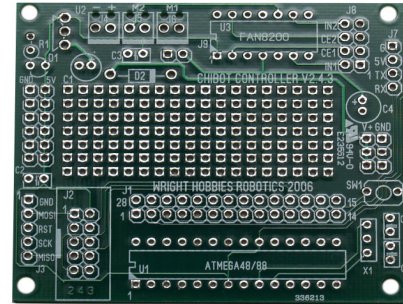
The image displays a variety of electronic components and kits. At the top, there are four small plastic bags containing: a white rectangular component, a bundle of colored wires, a coiled grey cable, and a small electronic module. Below these, on the left, is a black rectangular component with red and black wires. In the center is a breadboard with a green integrated circuit and several jumper wires. To the right of the breadboard is a small electronic module with a red LED and a USB cable. Below the breadboard is a box for a 'TRUCK TIRE SET' by Tamiya, featuring 36mm diameter tires. To the right of the tire set is a box for a 'DOUBLE GEARBOX' by Tamiya, which is a left-right independent speed gearbox. At the bottom right, there is a small electronic module with a red LED and a USB cable.

In order to assemble your controller board, you must be able to identify the correct parts to insert them in the proper place on the printed circuit board. There are seven different types of components that you need to know:

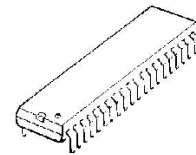
- Page: 11

- diodes
- light emitting diodes (LEDs)
- connectors and headers

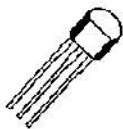
The printed circuit board, or PCB, - contains the circuit traces printed right onto a fiberglass board. The ChiBot Controller PCB is 2" by 3" (5 cm x 8 cm) and has a silkscreen printing on it showing where each component is supposed to go. If you look closely, you can see the copper traces running on the board from component pin to other component pin.



Sockets and ICs - are rectangular devices with several small, silver-colored pins along each longer side. There are two different rectangular ICs on the controller and each has a socket for easier trouble-shooting and repair. In integrated circuits, the actual circuit is much smaller than the “chip” that you see. The IC itself is actually about the size of your fingernail, and is packaged in a plastic or ceramic “carrier”. The carriers come in many different shapes and sizes, but your Mega-48 chip is packaged in what is called a PDIP-28 (or a Plastic Dual In-line Package with 28 pins). This means that there are two (dual) rows of pins, which are in-line (in parallel, on opposite sides of the rectangle) with each other. The two rows each contain 14 pins for a total of 28 pins. The socket for this chip is the same size and shape as the Mega-48, but has two rows of holes in the top for the legs, or pins, of the chip. The FAN4800 is also a PDIP, but this time there are 7 pins on a side for a total of 14 pins making the package a PDIP-14. IC’s have a mark on them to identify pin number 1. This is often a small circle or triangle in the corner of the carrier nearest pin 1, there may also be a notch in the end which has pin 1. The pins are numbered so that if you look at the chip with the notch up or the circle in the



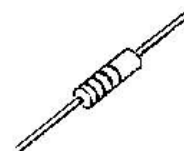
A 40-pin DIP package.



The TO-92 package IC.

upper left corner, pin number 2 is directly below pin 1. Going down, you have pins 3 then 4 and so on until you get to the bottom-left corner. Continue with the numbering by crossing over to the bottom-right pin and continue up the right side until you get to the highest number pin directly across from pin 1. There is also the L4931 Voltage Regulator IC. This is very small and comes in what is called a TO-92 package rather than a DIP package. The TO-92 package is very often used for transistors, it is a small black cylinder with one side flattened and three leads coming out the bottom.

Resistors - are small, cylinder-shaped devices with a wire lead coming out of each end of the cylinder. The resistors used in controller electronics are normally about 1/8" in (2-3 mm) diameter and about 1/2 inches (1 cm) long. They are usually either brown or tan colored and have three to five colored bands around the resistor. These bands tell the value of the resistor (how much resistance the device has), and they start near one end of the resistor and are read



band by band going towards the other end. Each color represents a number, and its position on the resistor tells the meaning.

The colors, and their meanings are listed here:

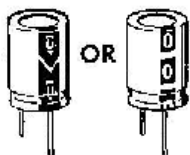
<i>Color</i>	<i>1st Band</i>	<i>2nd Band</i>	<i>3rd Band</i>
Black	N/A	0	X .1
Brown	1	1	X 10
Red	2	2	X 100
Orange	3	3	X 1,000
Yellow	4	4	X 10,000
Green	5	5	X 100,000
Blue	6	6	X 1,000,000
Violet	7	7	X 10,000,000
Gray	8	8	X 100,000,000
White	9	9	X 1,000,000,000

The colors for the 4th and 5th bands, if present, represent precision tolerances and are not important for our purposes here.

So, using the above table, if we have a resistor with the first three bands being: orange, blue and brown, we would take the first color meaning and list that as the first digit of the value (so orange means 3). The meaning of the next color would be the second digit (blue means 6, so thus far it's 36). Finally we would use the 3rd color to find out by how much to multiply the 36 to find the resistance (brown means that we multiply the 36 by 10 to come up with a total resistance of 360 Ω – the Greek letter Omega [Ω] stands for ohms, the unit of measure of resistance.)

Capacitors – are sometimes shaped, like resistors, as small cylinders (only with both leads usually coming out of the same end of the cylinder), and sometimes shaped like small disks with the leads coming out of the edge of the disk.

Occasionally, you will find capacitors that are like a small bead in the middle of two leads. The cylinder shaped capacitors are usually electrolytic capacitors and are, most often, used for power supplies and power circuits. The disk-shaped capacitors are usually used to condition signal lines and are called either disk capacitors or ceramic disk capacitors. The normal markings on a cylindrical



electrolytic capacitor are just labels with the value of the capacitor. They are also labeled with a black stripe or arrow on one side of the cylinder pointing to one of the leads. This black stripe has negative signs (-) on it and denotes the negative lead of the capacitor. Disk capacitor labels are a little bit more difficult to read. They usually have a three-digit code on one side. The first two digits are simply the first two digits of the capacitance and the third digit is just the number of zeros to add after the first two digits, this number is the number of pico-farads. So a disk capacitor with a code of 104 would be 10 with four zeros after it, or 100000 pico-farads or 0.1 micro-farads.

Diodes – are usually small cylinders that look like they are made of glass with wire leads coming out of each end. They are labeled with a black band circling around near one end. A diode

allows electricity to travel in only one direction, going from the lead near the black band through to the lead on the other end.

Light Emitting Diodes (LEDs) – are most often small colored plastic cylinder-shaped devices. They are usually rounded on one end of the cylinder and flat on the other end. There are two wire leads coming out of the flat end and one of those leads is shorter than the other. The side of the cylinder near where the short lead enters into the body of the LED is usually flattened. These two indicators show the cathode of the device and this is where the negative point of the circuit is connected.



Connectors and Headers – are devices that allow other circuits to be connected to your controller board. There are several types of connectors on your PCB. First, and most popular, are the snappable male headers. These black plastic components have copper colored pins sticking through them. The plastic is indented between the pins allowing for easy snapping of the full header into smaller headers. These headers are scattered across the entire board. Next are the three small green two-pin screw connectors. There are two used to connect the motors to your controller board and one used to provide power to the board. These are grouped up near the green LED power indicator. Finally, the 10-pin ISP programming cable connector is located in the lower-left corner near the Mega-48 processor chip.

The Parts List

Check each part against the following list. Make a check in the space provided (☐) as you identify each part. The parts may vary slightly from the picture. Any part that is individually packaged with a part number on it should be kept in its package after it is identified until you use it. Save all packaging material until all parts have been located. To order a replacement part, contact Wright Hobbies and indicate the model, value, and the description of the part to be replaced.

Each circuit part in this kit has its own circuit component number (R2, C1, etc.). Use these numbers when you want to positively identify the same part in the various sections of the Manual.

These numbers, which are especially useful when a part has to be replaced, appear:

- In the Parts List,
- In the description of each step where a component is installed,
- In the schematic,

The Chibot Controller Kit:

√	Description	Qty	Designator
	PCB	1	PCB1
	ATMega48(V) - Atmel Mega48 Microcontroller	1	U1
	Socket, 28-pin DIP	1	U1
	L4931 LDO 5V Pos Reg 250 mA	1	U2
	HBS-1 (FAN8200)	1	U3
	Socket, 14-pin DIP	1	U3

DIP Heatsink	1	HS1
10 uF Capacitor	1	C1
.1 uF axial ceramic capacitor	2	C2 & C3
220 uF Capacitor	1	C4
LED	2	D1 & Test
Power Diode	1	D2
Resistor, 1K ohm	1	R1
Resistor, 360 ohm	1	Test
Momentary Switch	2	SW1 & SW2
2x5 Shrouded IDC Connector	1	J2
Screw Terminals	3	J4, J5 & J6
40 Pin Snappable Male Header	2	
Terminal Shunts	5	
Mounting Hardware (4-40 mounting screws with nuts and screws)	4 sets	
26AWG Stranded Connecting Wire set	1 set	

The Programming Cable Kit:

√	Description	Qty
	SUB MINI D CONN./25 PIN MALE	1
	DB-25 HOOD	1
	6 COND. 26GA Multi Color Wire	10 ft
	Crimp Connectors	1
	black plastic connector housing	1
	Resistor, 360 ohm	3

The remainder of the kit is:

√	Description	Qty
	Tamiya Truck Tires with Wheels	1 set of 4
	Tamiya Double Gearbox	1
	Tamiya Universal Plate Set	1
	4 AA Battery Holder with leads (Battery not included)	1
	1" Square 1/16" Thick Foam Tape Squares (15)	1
	Photo-resistor	1
	Solderless Breadboard	1

Tools Required:

A needle-nosed pliers
A side-cutter wire cutter
A soldering iron, stand & solder
A solder sucker or solder wick

A “helping hands” clamp or masking tape
A set of hemostats
A tiny flathead screwdriver
Optional: A magnifying glass

Please note that, due to differences in suppliers and substitutions, your components may not exactly match the components shown in these photos.

The Brains of the Family

What is a Microcontroller

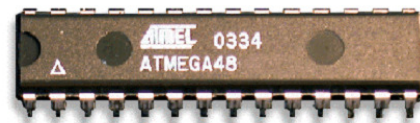
All computers are made up of four major parts. These include the Input unit, the Output unit, the Processing unit and the Memory units. The Input unit is the way that the computer gets the data on which it will do its work. The Processing unit performs the work of running the computer's programs. This is made up of a Control unit and an Arithmetic/Logic Unit (also called an ALU). Together these two units make up the Central Processor Unit, or CPU. While working, the CPU keeps it's instructions, or the program, along with the data on which it is working in the Memory unit. After the computer does it's computing, it will present the results of that work thru the output unit. A microcontroller combines all four parts of a computer into one single integrated circuit (IC) or chip. These microcontrollers are the brains in many products around the home and office. These microcontrollers can be found in watches, microwave ovens, telephones, cars and trucks, DVD players and robots, in fact almost every product which has a display will use at least one of these microcontrollers. A microcontroller also acts as the brain for your TableTop Bot. This microcontroller system was designed by Wright Hobbies for the Chicago Area Robotics Group and is called the ChiBot Controller Board.

The Atmel Family

Atmel is a company, headquartered in California, which specializes in building IC chips for the control industry. One of their product lines is the AVR family of microcontrollers. The AVR chips are some of the most powerful microcontrollers available for exceptionally low prices. Atmel produces the specific microcontroller, which is used as the control system for your new robot.

The Mega 48

The microcontroller that we use in the ChiBot controller is called the AVR-Mega48. This is one of Atmel's mid-range microcontrollers. It has 23 input/output (I/O) pins, including six analog-to-digital (A/D) converters and six digital-to-analog (D/A) converters. There are 4K bytes of program memory, 512 bytes of RAM and 256 bytes of data storage EEPROM. MCS Electronics produces a language compiler called BASCOM-AVR for Atmel's AVR family. The free trial version is limited only in the size of the program. This limit is 4K bytes, which makes the Mega-48 a perfect match for learning about microcontrollers with BASCOM-AVR.



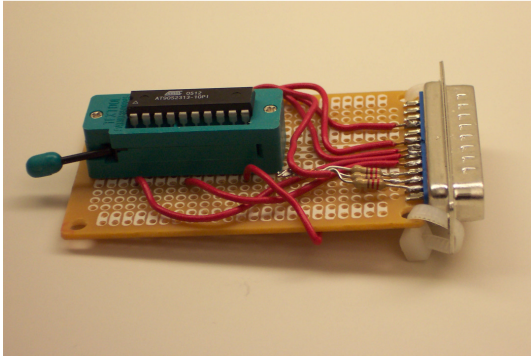
Atmel's AT Mega-48 Microcontroller

The FAN8200

While plenty strong enough in the brains department, most microcontrollers are slightly low in the muscle department. By itself the microcontroller is not strong enough to provide the electricity needed to light several lights, turn motors or even power a radio for it to transmit information. This is where the FAN8200 motor driver chip comes in. The FAN8200 can be thought of as an amplifier, which will take a relatively weak signal from the Mega48 and makes it hefty enough to power the motors that drive your robot across the table.

The Downloader Cable

In order to put a program into your microcontroller, you must have a way to get the program from your PC into the chip itself. The way



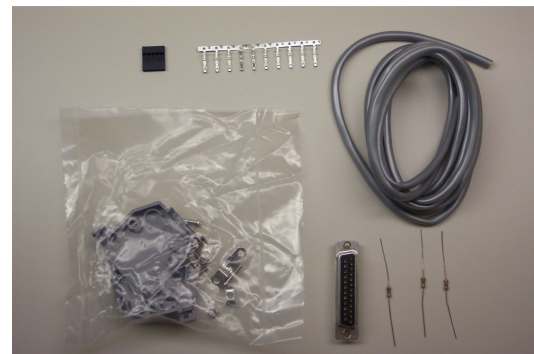
A simple device programmer for another Atmel chip. This is called the Fun Card programmer.

engineers used to do this was by taking the chip and placing it into a special device called a programmer. After running the programming software, you would take the chip with the program in it and place it into your circuit (or into the robot's board.) If, or rather *when*, you discovered an error in your robot's program, you would take the chip out of the robot, put it into a special device to erase the program (often taking hours), put it into the programmer, and start the whole process again. This would happen over and over until either you got the program correct, or (often) you just got tired and quit.

We will be using what is called an ISP Downloader Cable. The ISP abbreviation stands for In System Programmer, this is a way that engineers have developed to get around that old way of using a dedicated programmer. We have a special ISP connector on the microcontroller board, and will just connect the programming cable to the board in order to download the program. All of the programming activity is accomplished on the circuit board and the microcontroller chip itself. No need to pull the chip out and move it around.

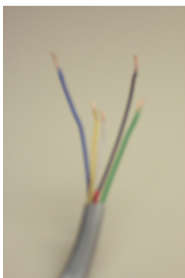
Before you start making your programming, or downloading cable, first check the parts to make sure that you are not missing anything. We work hard to ensure that you get every piece, but mistakes can happen. If you notice that anything is missing, please contact us immediately at: sales@wrighthobbies.net, by telephone at: 630-214-9534 or fax us at: 630-563-0860.

Your Downloader Cable kit should contain the following items. Referring to the "contents" photo, starting in the upper left corner and going clockwise, we have the black plastic connector housing, the female crimp-on pins (10 each), the



The contents of the Downloader Cable kit..

cable itself, three 360Ω (orange, blue, brown) carbon resistors, the DB-25 printer connector and finally the package containing the DB-25 hood parts.



Strip the outer jacket off.

To build your cable, start with the piece of gray cable. Strip about 2 inches (or 5 cm) of the outer jacket off one end of the cable to expose the 6 colored wires inside. Strip the insulation off the red wire and then lightly tin that bare wire. Next cut the short piece of tinned wire off, we do not need it for the cable, but we do for the connector.

Next get the DB-25 connector. This is the silver-colored connector that plugs

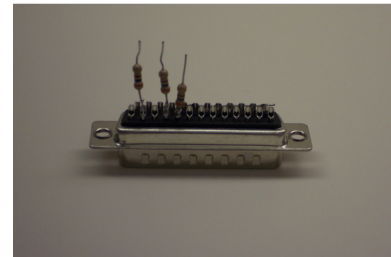


The DB-25 Female Connector

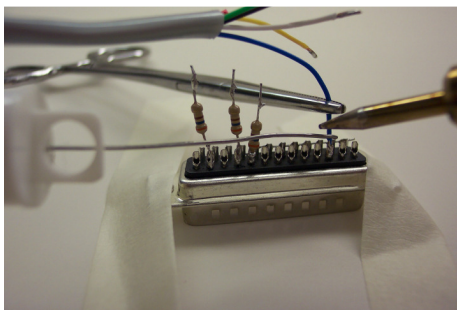
into the printer port of your computer. Place the connector into a small vice, a set of “Helping Hands” or just tape it to your workbench, with the pins down, this will leave the solder cups facing up. Set the connector so that the row of 13 pins is facing you and the row of 12 pins is away from you. We will be soldering the tinned, red wire from the cable across the ground pins of the connector. With the connector set the way I described, the eight pins to the left will be ground. Take the tinned wire from the cable and bend it 90° about 1/16” (or 1 mm)

from the end. Next lay it across the eight pins with the bent end touching in the solder cup of pin 18 and the straight end hanging off the connector past pin 25. Now solder the wire to each of the pins from 19 through 25. Do not solder pin 18 yet. All of these pins from 18 through 25 are the ground pins for the connector. Go back over each pin and visually check to make sure that there is a good solder joint between the pins and the wire. When you are satisfied that all seven pins are soldered together (remember we don’t want pin 18 soldered yet,) then trim the wire past pin 25.

Next take the three resistors and cut one of the leads on each resistor to about 1/4” (or 1/2 cm) from the resistors’ bodies. Now, lightly tin the short lead of each resistor, this end will go into the solder cups and be soldered into place. Turn the connector around so that the row of 12 pins is nearer to you. This will place pin 1 on the connector to your left. Pin two will be the pin next to the leftmost pin. Place the tinned lead of one resistor into the solder cup for pin 2. Now holding the resistor in place, solder it into the cup. After checking for any problems with your solder joint, repeat this step with a resistor for pin 4 and again for pin 5. These are, respectively, the 4th and 5th pins from the left.



Solder three resistors to connector.

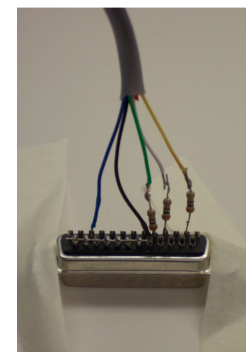


Solder the wires to the connector.

Now, get the cable again. Take and trim about 1/2” (or 1 cm) off the end of the yellow, white and green wires. These are the wires that will connect to the resistors. Now strip about 1/8” (or 1/4 cm) of insulation from each of the five wires. Lightly tin the ends of these five wires. Be careful not to get too much solder on the two longer wires because this will make it difficult to put the wire in the solder cups. Also, trim the leads of the three resistors to about 1/4” (or 1/2 cm) from

the resistors’ bodies. Then tin the short leads of the resistors.

After you get the wires and resistor leads tinned, place the end of the blue wire into the solder cup for pin 11 and solder it in place. Then turn the connector around again so that the row of 13 pins is near to you.



Solder the remaining wires to the resistors.

Now place the end of the black wire into the cup for pin 18, along with the wire soldered to the other ground pins. Now solder these wires into place.

Next, solder the remaining wires to the three resistors that stick up from the connector. The yellow wire will connect to the resistor on pin 2, the white wire will connect to the resistor on pin 4, and the green wire will connect to the resistor on pin 5. After you solder each of these wires to their resistors it is a good idea to wrap a small piece of electrical tape around the cable side of each resistor and their solder joints. Be very careful while doing this. You do not want to move the wires or resistors any more than absolutely necessary. This piece of tape will prevent electrical shorts inside the connector's hood.

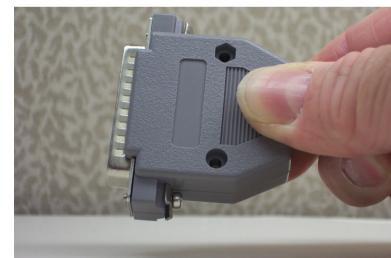


The two halves of the strain-relief clamp.

Get the strain-relief clamp and put the two halves together facing the same way (in the “spoon” position) because the cable is too small to be properly held when you put them opposite. Next, wrap the metal strain relief clamp around the cable just before where you stripped the outer jacket off. Tighten the strain-relief clamp down on the cable to hold it firmly in place on the cable. Note that the screws go into the clamps from opposite directions.

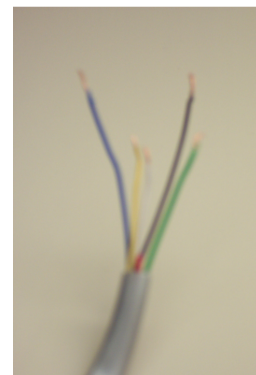
Now, carefully, take one half of the plastic hood and place it under the DB-25 connector. Then gently, place the other half of the hood on top of the DB-25 connector, aligning it with the bottom half. Gently push the strain-relief clamp into the carrier inside the hood (it looks like a set of parenthesis.) Finally, gently push the two halves of the hood together and then secure them to each other with the nut and bolt sets. Make sure that you put the nut in the hex-shaped hold, and you will not need a wrench.

Congratulations, you have completed the difficult end of the programming cable.



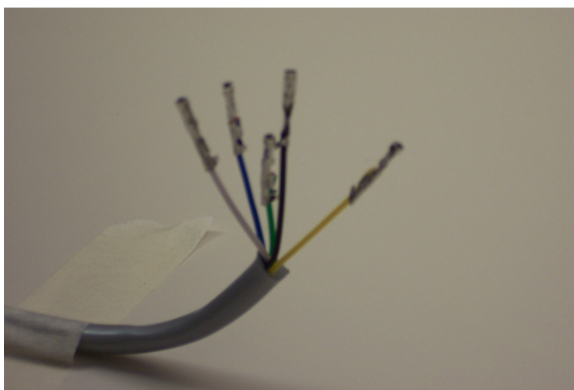
Hold the hood closed while you insert the nuts and bolts.

On the other end of the cable, strip about 1½” (or 3-4 cm) of the jacketing off the cable. Again, we do not use the red wire, so cut it off. Strip about 1/8” (or ¼ cm) of insulation off of the other five wires. Fold the bare copper of the black wire back on itself so that it lies against the insulation. Break one of the female crimp-on pins off the bunch and place the folded black wire into the crimp pin. Crimp the wire into place with a crimping tool or a needle-nosed plier. If you are using a needle-nosed plier, remember to crimp the pointed stress relief guides on the connector down to the wire insulation. Give the crimp pin a gentle tug to make sure that it won't come off the wire. A crimping tool made for this task will make your job much easier, but you can do it completely with a needle-nosed plier.



Strip the jacketing off the cable's other end.

When you are sure that your crimp job is secure, repeat that crimping process on each of the other four wires. If you make a mistake, you will probably need to use another pin and try again. The process of crimping will destroy the pin if it is not correct. Don't

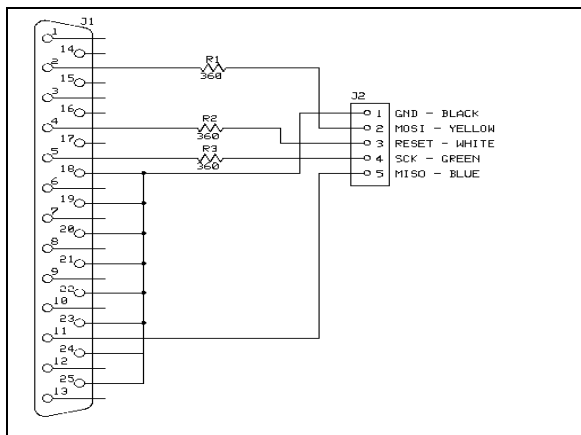


Crimp the female pins onto the cable's wires.

on the black wire into the hole on the top of the plastic housing, which is to your left. Make sure that the open end of the crimp connector is facing the side of the housing with the holes. Push the connector in until there is no metal from the top of the connector pin showing. This is pin 1 of the connector; it would be a good idea to mark this by painting that side of the connector with a small dab of "White Out" type correction fluid.

Repeat the above step with the other four female crimp-on connectors. These must go into the plastic housing in the correct order. Next to the black wire in pin 1, insert the yellow wire for pin 2. Then insert the white wire for pin 3, the green wire for pin 4 and the blue wire for pin 5.

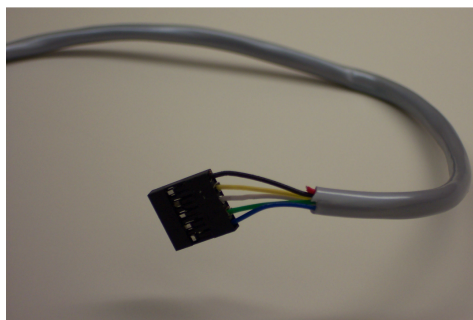
As a final test, plug the black end connector on your new programming cable into one of the male header sticks included with your controller kit. This should go on snug and not be loose or sloppy. If everything fits nicely, carefully remove the plug from the male header. Remember to always plug and unplug the connector by holding the black plastic body and never pull it by the wires. There is no strain-relief on this connector.



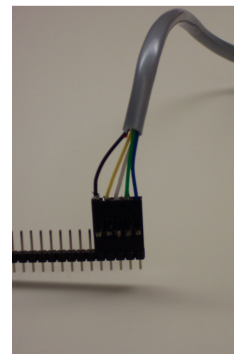
The Downloader Cable Schematic

worry too much about this since you have a few extra pins for this event. Do try to not use all of the pins, since we will be using a couple of them for testing your controller.

Finally, hold the black plastic connector housing with the five small holes on the side facing you and the larger set of end holes facing up. The holes on the side will be slightly closer to the bottom of the connector in this position. Push the crimp on connector



Insert the crimp-on pins into the plastic housing.

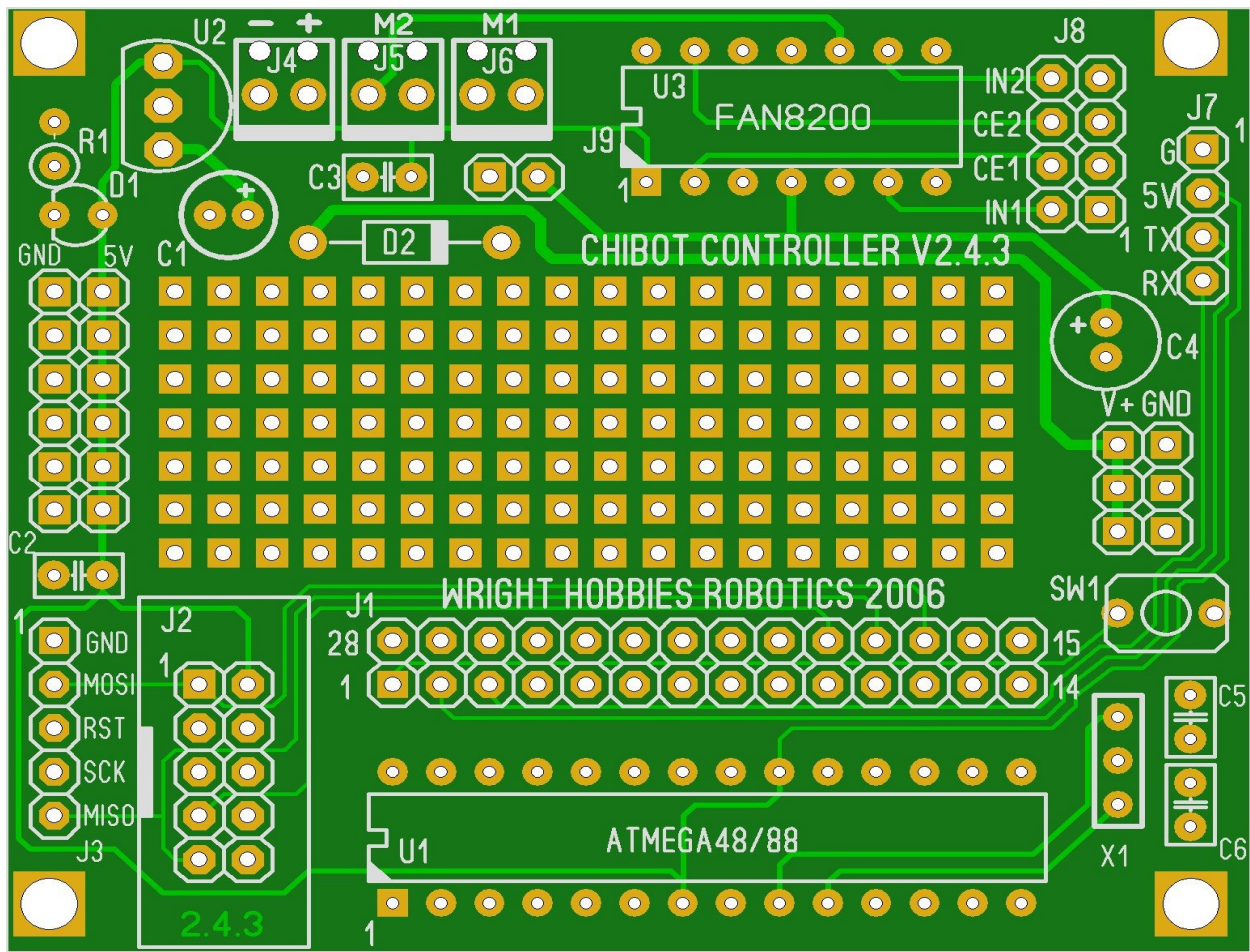


Test your connector.

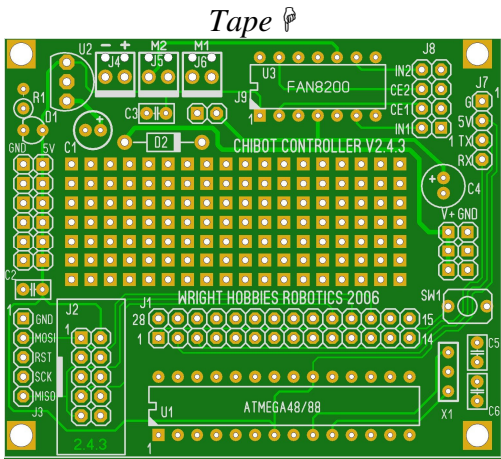
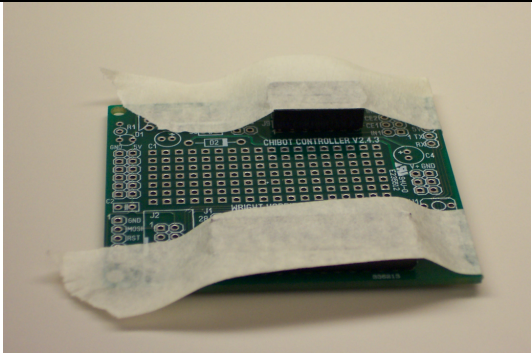
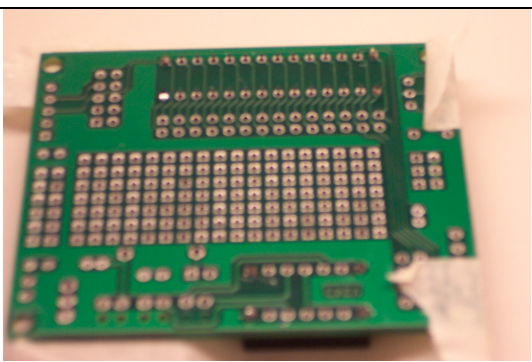
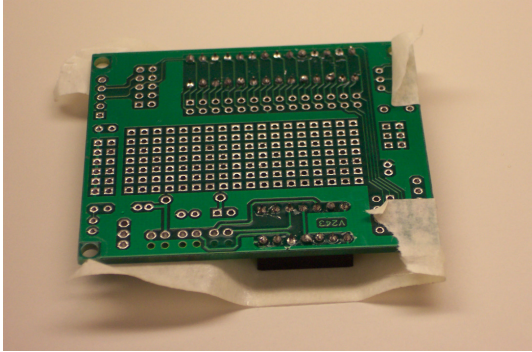
The Controller Board

To build the ChiBot Controller Board, you will need to make sure that you have the tools that you need. This will include a soldering iron with a fine point, some rosin-core solder, a wire cutter (side- or flush-cut), a small needle-nosed pliers and some masking tape.

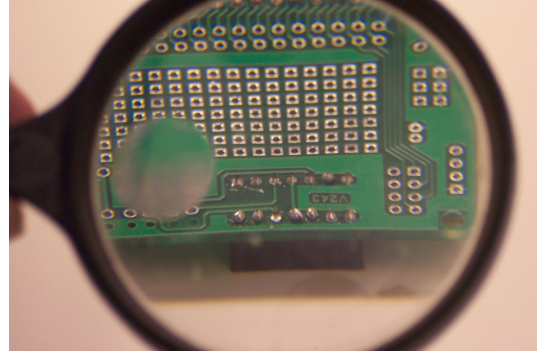
To get ready, set aside the two 40-pin headers for safekeeping. Then take the small components and stick their leads into the black anti-static foam. This will keep the components from rolling around and going missing.



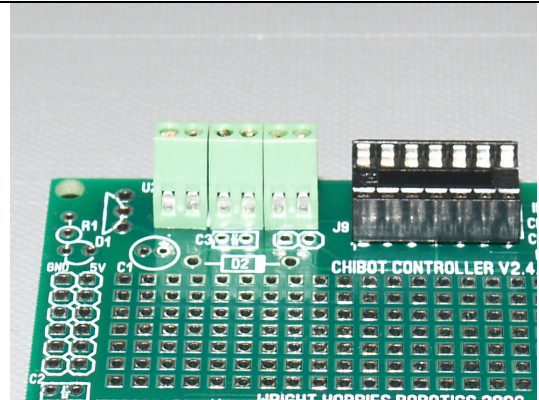
Study this diagram of your ChiBot Controller board. You may want to go through the instructions step-by-step with this diagram before starting to do any actual soldering.

<p>Orient the Printed Circuit Board (PCB) so that the printed side is up and the bottoms of the letters are towards you. Take two strips of masking tape about 1½ - 2 inches (5-6 cm) long and set them aside so that they are ready for use. Take the smaller, 14-pin IC socket and place it into the board in the position marked U3 (FAN8200). Make certain that the notched end of the socket goes to the left and lines up with the notch in the printout on the board. Make sure that each pin lines up with and goes through its corresponding hole in the PCB and no pins are bent under the socket. When you are certain that you have inserted the socket properly, place a piece of tape over the socket and tape it tightly to the board.</p>	
<p>In the same way, insert the longer 28-pin IC socket into the PCB at the position marked U1. Tape this socket tightly to the board too. Like the socket for U3, the notch in the end of the socket will go to the left. When you have both sockets taped down tightly, turn the PCB over.</p>	
<p>Now, solder each of the four corner pins on both sockets. Inspect the top (component) side of the PCB to ensure that both sockets are mounted flush with the surface of the board. If the socket is not tight to the board, you will need to reheat the adjacent pin(s) while gently applying pressure to the top of the socket using a small pad to protect your finger. This pad should be made from at least eight layers of a paper towel. How we discovered the need for a pad is another story.</p>	
<p>Once the sockets are correctly mounted, solder the remaining pins. We prefer to solder every other pin all the way around the socket and then start at the beginning and fill in the missed pins. This minimizes heat build-up in the plastic part of the socket.</p>	

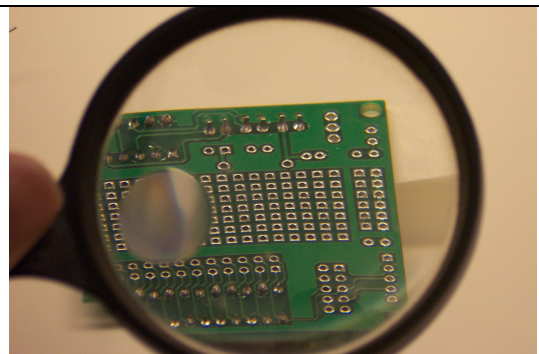
Next, remove the tape and check your work again. Check each of the 42 pin's solder joints very carefully to make sure that you have a good connection and do not have any solder bridges. Note that pin 1 on both sockets (and every jack) is indicated by a square pad. On U1 (the mega48 socket) there may be a tiny blob of solder on the ground plane next to pin 8 or pin 22. This happens sometimes, and is not a problem. These pins are connected to ground by a tiny bridge of copper tracing and sometimes the solder flows across it looking like a solder bridge. This "non-problem" can also sometimes occur with pins 7, 8 or 14 of U2. Any other solder bridges are an error and the solder bridge must be removed with solder wick or just reheating and allowing the solder to flow onto the iron.



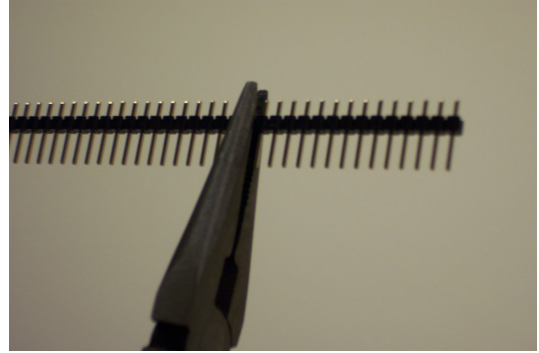
Locate and insert the three double-screw terminal blocks into J4, J5 and J6 positions. Note that there are two metal pins on the bottom of each block and two plastic tabs. On the board there are four holes in each position. The two holes near the edge of the board do not have any copper plating, and this is where the plastic tabs will fit. The metal pins will go in the holes further away from the edge of the board. Look at the terminals, the wires should be inserted from the edge of the board and not run across the middle of the board. Make sure that the terminal blocks are flush to the board and then tape them down. One end of the tape will be going over the socket for U3 while the other end folds over the end of the PCB.



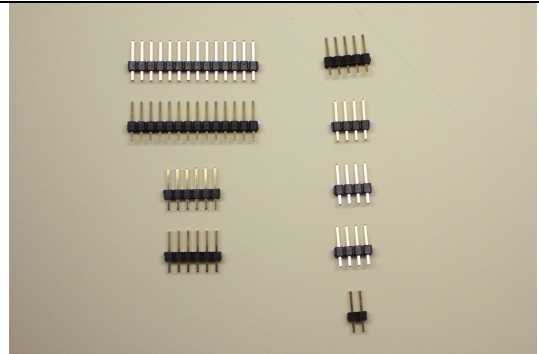
Now turn the PCB over and solder all six pins of the terminal blocks. Inspect your work again, and then remove the tape. Note that the pin furthest away from the U3 socket is grounded, but there must be no other pins bridged to ground.



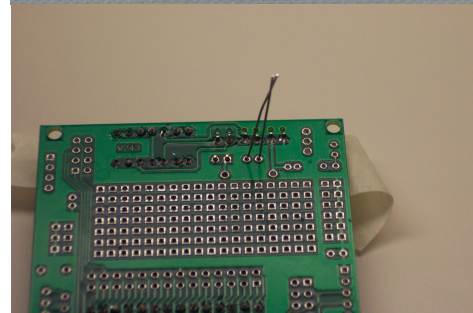
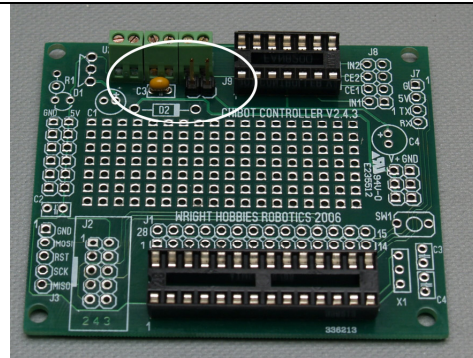
Now, it's time to prepare the headers. There are at least three ways to break the snappable headers: 1) grasp the header with a pair of needle-nosed pliers on each side of, and parallel to, the separation groove and gently bend back and forth until the header snaps; 2) use diagonal wire-cutters to fracture the header at the separation groove; and 3) cut through the separation groove with a micro-thin hobby saw. With # 3 you should place the long end of the header strip in a vise to safely hold it while you saw. After cutting the header off the strip, you may want to use some fine (#100 or #120) sand paper or an emery board to remove the rough edge of the plastic. Please clear this with your wife, girlfriend or daughter before you use their emery board – they won't like what the plastic does to their manicure equipment.



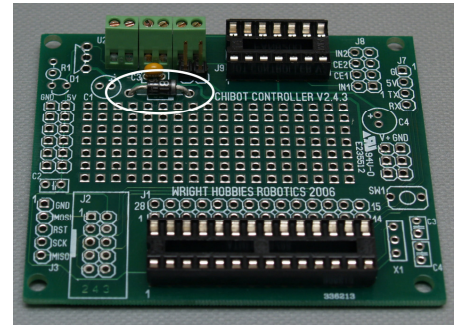
Take one of the 40-pin headers and snap off two 14-pin headers and two 5-pin headers. These will be for J1 and the two power headers for ground and +5V. One pad for each power trace will be wired to the breadboard. From the other 40-pin header, snap off one 5-pin header for J3, the programming header. Snap off another three 4-pin headers for J7 and J8. Finally, snap off one 2-pin header for J9. This will enable or disable the motor driver chip.



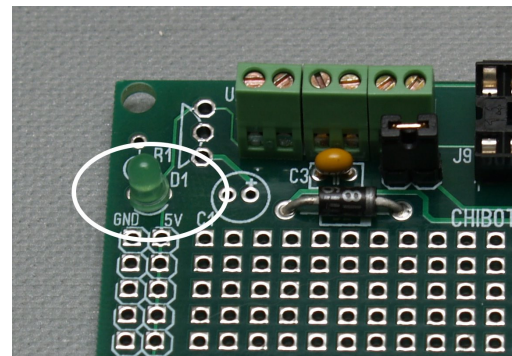
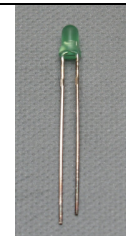
Now, it's time to start mounting the rest of the connectors and components. Start up near the three terminal blocks. Take one of the two tiny brown capacitors, they both have writing on them, but it may be so small that it's difficult to see. The marking on the capacitors will include the number 104. This represents the rating of the capacitors, 0.1 μ F. Place this capacitor into the holes for C3. Take the 2-pin male header and place this into the holes for J9. With both of these components, there is no polarity so you can put them in either direction. When you have them in place, tape them down. This will take some playing around to get the tape in place and hold the components down. When you get those taped down, turn the PCB over again and solder them in place.



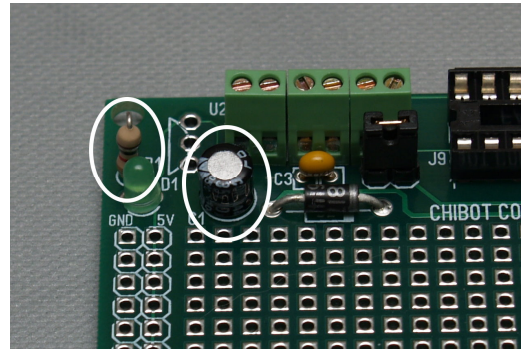
Take the diode and gently bend the leads 90° so that they are parallel to each other. Now place it into the PCB in the position labeled D2, this is located just below the capacitor, which you just inserted. While you are inserting the diode, be certain to get the polarity correct. There is a gray band around the end of the black body of the diode. This band must line up with the white band to the right of the D2 designator printed on the circuit board. On your ChiBot Controller board, this band is nearer to the U3 socket.



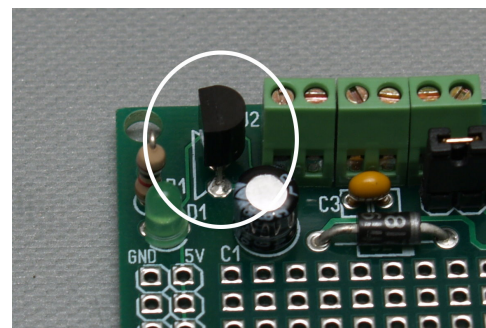
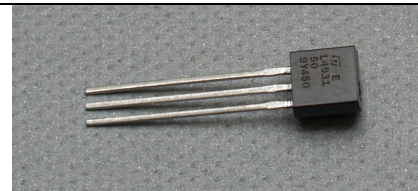
The power indicator is labeled D1. This is the small green LED. If you take a look at your LED, you will notice that one of the leads is shorter than the other one. If you carefully look at the body of the LED, you may notice that the round body is flattened on the side where the short lead goes into the plastic. This is often difficult to notice on small LEDs like the one we are using. This shorter lead is the cathode. The cathode, or the negative lead, of the LED must be placed nearer to the edge of the board. Once you get the LED in place, tape it down and again solder it in place. The diode is sensitive to heat, so be careful to not heat any one lead for more than about six seconds. If you are not experienced in soldering semiconductors, it would be a good idea to pause between leads to let the device cool down.



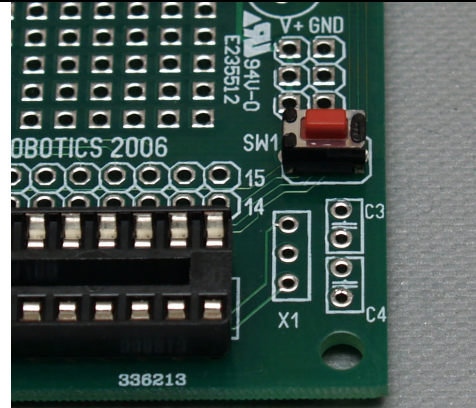
Now get the resistor, this is labeled R1 on the board. This is a 1 Kilo-ohm, or $1K\Omega$, resistor, and has at least three bands around it. Those bands will be brown (for 1) and then black (for 0) followed by a red band (for 2 additional 0s). There is no polarity so take either one of the leads and bend it back over the body so that it is parallel to the body of the resistor. Take the resistor and insert the unbent lead into the hole with the circle around it, right next to the LED. The bent lead will then go into the hole right below that, next to the 3-pins for U2. Next insert capacitor C1. This is the smaller of the two can-type caps, the rating is typed right on the side of the capacitor and should read $10\ \mu\text{F}$. This capacitor is an electrolytic capacitor and you must pay attention to positive and negative. The negative lead is marked by a gray stripe on the side of the capacitor near the shorter lead. There are negative signs (-) in the gray stripe. Insert the capacitor with the positive (longer) lead closer to U3.



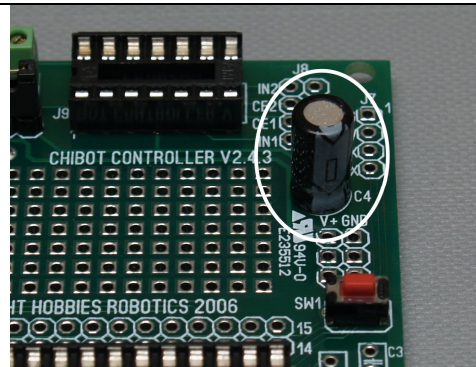
Next, insert the voltage regulator in the position for U2. This IC is packaged in a TO-92 case and looks exactly like a simple transistor. It has only three pins and they stick out of the bottom of the case. One side of the case is flattened and that must match up with the flat side of the semi-circle outline printed on the circuit board. The pins on the bottom of the chip are in a line and will be inserted into the three holes in the PCB. Once these components are in place, tape them down and, again, solder carefully. The voltage regulator is also sensitive to heat, so be careful to not heat any one lead for more than about six seconds. Like the diode, it would be a good idea to pause between leads to let the device cool down.



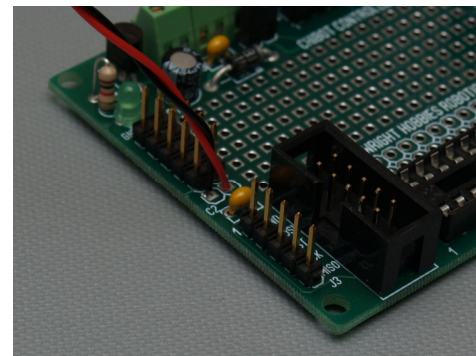
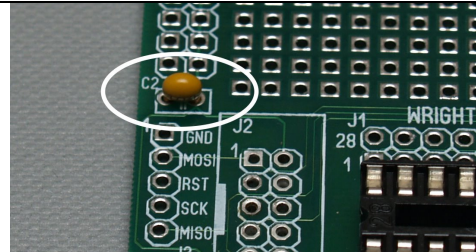
Now, on to the reset switch, SW1. This is the small, red pushbutton switch included in your kit. These are normally open, momentary action, SPST switches. You have two of them, but only need one for this reset function. The other one will be used in some experiments later on. Take one of the switches and place it in position on the right side of the circuit board. Once this switch is in place, tape it down and, again, solder carefully. Below this switch, are positions for X1 and C5 & C6. These are not used in the TableTop Bot. These allow for an optional ceramic resonator or crystal to be installed.



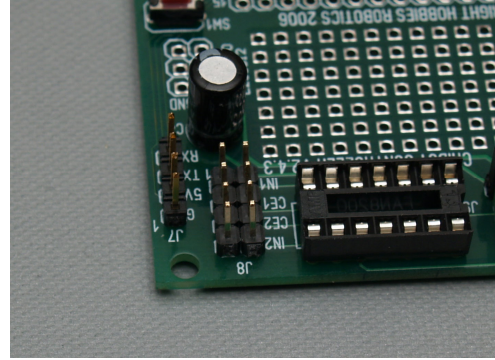
The next component that we will need to place is C4. This is the larger of the two can type capacitors. Like C1, this capacitor has a short lead for the negative pole and also a gray strip on the side of the body of the capacitor. Place this capacitor on the far right side of the board just over 1" (2½ cm) down from the top. The pads below this capacitor are for the unregulated power coming directly from the battery pack and not through the voltage regulator.



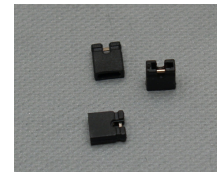
Next take the other tiny, capacitor, the 0.1 μ F one, and insert it in the space for C2. This is located along the left side of the board. Once again, the polarity does not matter. Tape this capacitor down to the board, and once again solder it into place. After checking your solder joint, take the 5-pin male header and the other two 5-pin headers. Insert the 5-pin header in to the position marked J3 on the board. This is for the programming cable that you built earlier. Insert the two 5-pin headers right above J3 in the positions marked GND and 5V. Leave the holes nearest C2 empty for the breadboard power lines (shown in picture). These headers will provide power to any additional circuits that you may add to your ChiBot Controller. Insert the 2 X 5 shrouded IDC connector in the position for J2, this is for a standard STK200/300 compatible programming cable. This is optional and we will not be using it here. After you have these three headers in position tape them down and then solder them into place.



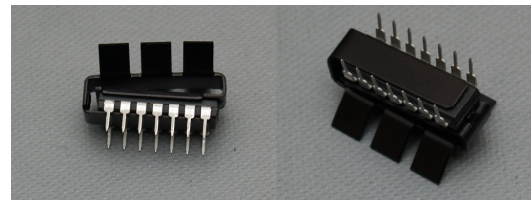
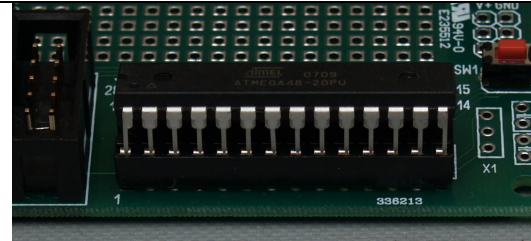
Finally, we use the last three male headers. These will make up the J7 & J8 connectors. J7 will be for an optional serial interface to let your controller communicate with another controller or a desktop computer. J8 is a shorting connector, so that you can change which controller ports drive the motors. For our purposes, we will use the shorting blocks. Take the three connectors and insert one in J7's position and two into J8's position. Tape these down tightly to the PCB. This may take some jiggling, since you cannot put the tape very far over the edge of the board without it covering the solder pads. After you tape the headers down, turn the PCB over and solder them in place. *Hint – you can place the 2-pin jumper shunts on J8 before you solder to help hold them in place.*



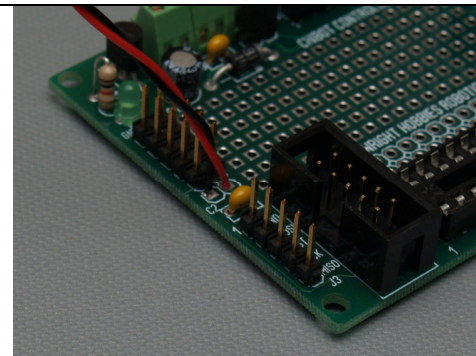
Board is turned around to see it better.



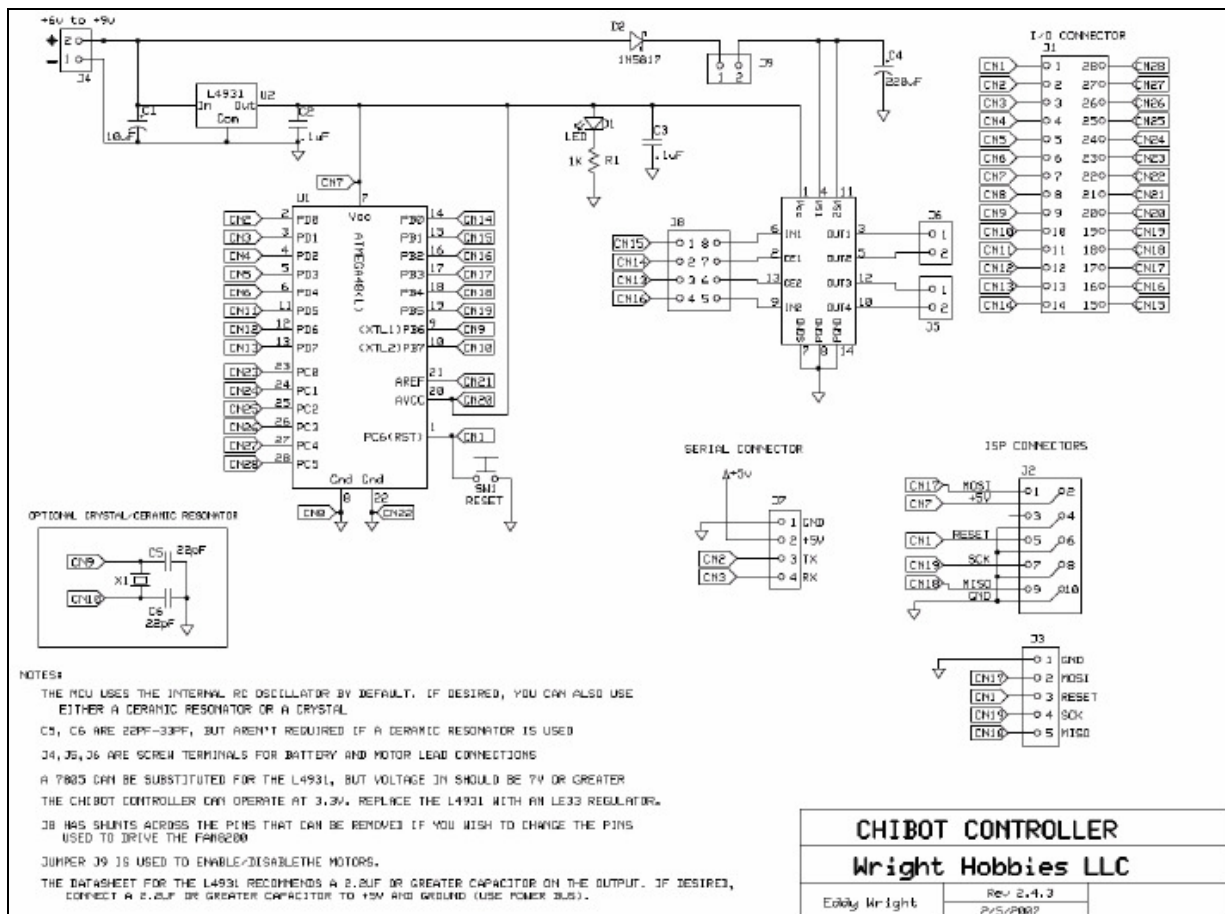
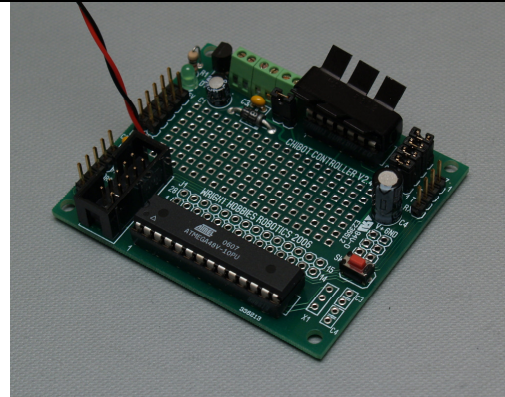
Almost done! Place the 2-pin jumper shunts on the pins of J8, one over each the two pins labeled IN2, CE2, CE1 and IN1. Place another jumper shunt on J9, located by D2. The pins on DIP chips flare out a little and will require that you gently bend the two rows of pins back towards each other. To safely bend the pins, place the chip on its side on a flat surface. Hold the chip by the ends and gently roll the chip until the pins bend slightly. Turn the chip over and repeat the process for the other side. Place the chip into the socket slowly; aligning the notch in the chip with the notch in the socket and ensuring that all the pins are going into the socket. Carefully push the chip into its socket until it is seated fully. For U3 (FAN8200), place the heatsink over the chip before inserting it into the socket. The chip slides between the metal plates of heatsink. The small tab on the heatsink fits into the notch in the chip.



Cut two short wires about four inches (10 cm) each, one red and one black. Strip about 1/4" (1/2 cm) of insulation off both ends of each wire. Solder the black wire into the bottom hole (nearest C2) of the GND bus. Solder the red wire into the bottom hole of the 5V bus. These wires will provide power to the breadboard.



Congratulations, you now have a usable brain for your robot. As a final step, you should plug the programming cable into the 5-pin programming port on the ChiBot Controller board and use a red permanent marker or a bottle of white correction fluid that has a brush in the lid to paint the male and female connectors on their edge near pin 1 (give this several minutes to dry). This will make it easier to ensure that you plug the programming cable in properly.



This is the schematic diagram for the ChiBot controller.

Programming Your Controller

BASCOM-AVR

A computer is nothing more than a bunch of circuits, which will just sit there and wait if you don't tell them what to do. The process of telling a computer what to do, and exactly how to do it, is called programming. Just like in any instructions, you need to speak the person's language. Our Mega48 controller speaks a language called AVR Machine Code. To us this looks like a bunch of random ones and zeros. We use a translation program to take a program written in a human language we call BASIC and "compile" or translate it into the Machine Code for the Mega48.

A good analogy of a computer program is a recipe from a cookbook. It is a list of directions, which you follow to do a job. Although, a recipe is actually nowhere near as explicit as a program. In a computer program, you must normally give every single step of the job; a computer cannot think to fill in small steps that you may have forgotten. If you forget a step, the computer will skip that step and you will end up with bad results.

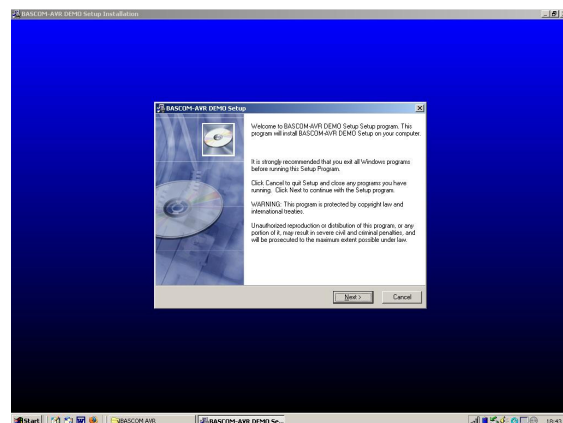
You don't really need to worry about that in the beginning. The BASCOM compiler that we use will take care of most of our work for us. All of the programs we write will be simple enough that the programs will only take a short time to write (though it may take a while to fine-tune it for best performance.)

Getting the Latest Version

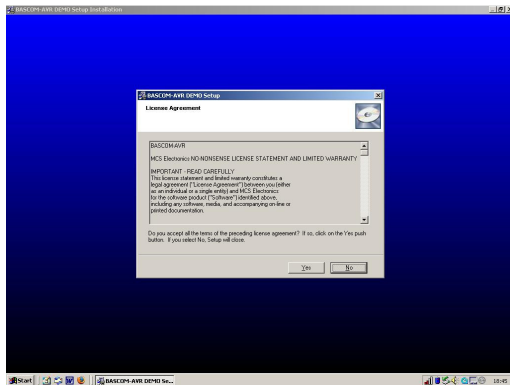
In this manual, we will be installing and using BASCOM-AVR version 1.11.8.3. This is the version that comes with your kit. All of our pictures and examples will be from this version. If you know enough about computers and are able to compensate for any differences you will see, you can get the newest version. You will get this from MCS Electronics, the publisher. Go on the web to www.mcselec.com and click on Downloads on the left side of the screen. On the Downloads page, click on the BASCOM link at the bottom of the page. On the BASCOM page, click on the BASCOM-AVR link at the top of the page. On this page, you may also find more information about BASCOM-AVR, including articles written about this compiler. On the BASCOM-AVR page, click on the link to download the BASCOM-AVR Demo version. While you are on this page, also download the manual. This has lots of information and is very well organized.

Installing BASCOM-AVR

First insert the CD that came with your kit. This has the BASCOM compiler installation program on it. When the TableTop Bot menu comes up, select "Install BASCOM-AVR". This will begin the installation. The first screen that the installation gives you will simply be a welcome screen. Click on the **[Next>]** button to begin the installation. Next, the Readme file will be displayed. Read over this note file for any information more recent than this manual. When you are done with reading the notes, click



The BASCOM-AVR Welcome Screen



The License Agreement

saving now. After you read and agree to the License Agreement, click **Yes**. If you click **No**, the program will not install and you will need to find your own way of programming your TableTop Bot (there are other ways, but they are a little more difficult and we will not cover them in this book.) When you get to the Destination Location screen, unless you have a reason not to, just click **Next>** to accept the default locations. Next the installation will ask you if you would like to backup the files that it replaces. This is a good idea, and does not take up much disk space. Go ahead and leave the Yes radio button selected, and click **Next>**. The last thing that the installer needs before it begins to install the files for BASCOM-AVR is which Program Manager group to put the installation into. Again, just click **Next>** to accept the default BASCOM-AVR group. At this point, the installation is ready to actually start copying files onto your hard drive. Go ahead and click on **Next>** again to begin the copying. The installation program will begin copying files and will keep you advised as to it's progress by two "progress bars."

Setting up BASCOM for our ChiBot Controller

There are a few settings which we need to make to prepare BASCOM for our specific setup. First we need to tell BASCOM which programmer cable we are using. The cable which you put together earlier is based on a programmer from a company called Sample Electronics. Their programmer is built into BASCOM and so our ChiBot controller can be programmed directly from within BASCOM.

Launch BASCOM AVR. Now set the specific programmer that you will use by clicking on the Options menu and then selecting Programmer. In the Programmer field, use the drop-down box to select the Sample Electronics programmer. Click **OK** to save your selection.

Your First Program

If you have been following these instructions, you are already in BASCOM, otherwise launch BASCOM-AVR by double-clicking on it's icon on your computer's desktop. Click on the New icon (looks like a blank sheet of paper directly under the **File** menu.) This will open a new, empty, workspace window. Click inside that workspace and start typing the following program:



```

'Chibots Tabletop Bot
'Author: Art Granzeier, Wright Hobbies Robotics
'This program is designed to verify operation of the Chibots Controller Kit
'For More information, please visit http://www.wrighthobbies.net/bots/ttbot

'The following parameters can also be set under Options/Compiler/Chip
$regfile = "m48def.dat" ' specify the used micro
$crystal = 1000000 ' used crystal frequency
$hwstack = 32 ' default use 32 for the hardware stack
$swstack = 10 ' default use 10 for the SW stack
$framesize = 40 ' default use 40 for the frame space

Config PortB = Output

Do
  toggle PortB.1      ' Create a loop
                      ' Toggle the pin
  Waitms 500          ' Wait for half a second
Loop                  ' Repeat the loop

End

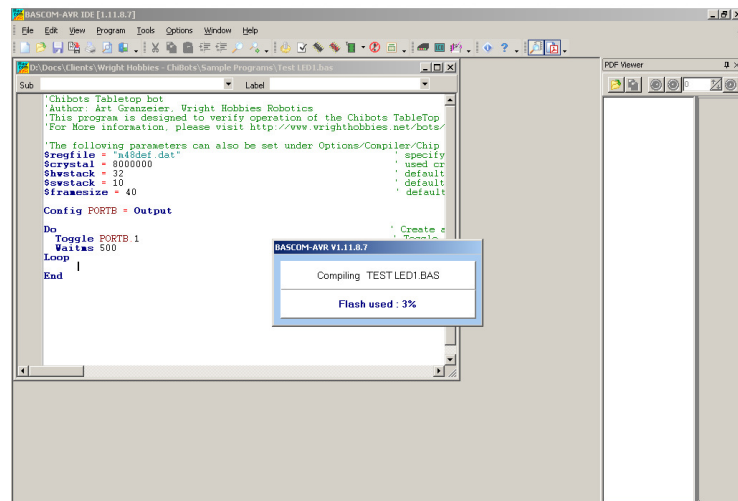
```

This is the microcontroller equivalent of the “Hello World” program that almost every programming student writes. What the program does is this: the first five lines are comments for the programmer. They give information about what the programmer was doing in this program, to help him remember what he was thinking while writing this program. The next five lines, which begin with the dollar sign (\$), are instructions to the compiler to set itself up for our Mega48 chip. Next, we need to tell the compiler to set up Port B as an output port. Next the four lines, which form the loop, are the main part of the program. They start a loop to repeat the instructions inside the loop. Next they toggle Port B bit 1 (pin 15 on the controller chip) on or off, and then wait for 500 milliseconds. The loop command tells the controller to go back to the beginning of the loop. Finally, we end the program with an End command. This is just good programming practice.

Running Your Program

Before this program can be run, it must be translated into the language used by the Mega-48 controller. This process is called compiling. You compile a BASCOM-AVR program by pressing the F7 key on your keyboard. As in most Windows programs there are several ways to do this, I will normally only cover one way, but if you prefer a different way that works, by all means use your way. When you first compile a new program, the compiler will ask for a name for the program. You can just use “Test LED” for this program.

Enter the name and click on **Save**. When the window pops up asking if you want to leave the old CFG file, click **Yes**. BASCOM should compile your program and show “No errors found” in the status bar at the bottom of the BASCOM screen.



Compiling the Test LED program in BASCOM-AVR.

If you get errors, recheck your typing. This program was directly copied and pasted from the BASCOM environment with no compiling errors.

Setting up the Circuit

At this point, we need to get the controller ready. First, take the black wire that is soldered in to the ground pin of the power pads and plug it into the ground bus strip of the breadboard. Next, plug the red wire into the 5V bus strip of the breadboard. Now, take one of the short pieces of wire, about 2-3" or 5-7 cm, from your hook-up wire kit and strip about 1/4 " (or 1/2 cm) of insulation off each end. Use one of the remaining crimp-on connectors from your programming cable and crimp it onto one of the stripped ends of the wire. Take the wire and plug the crimp-on connector into pin 15 of J1, this is the pin closest to the reset switch, SW1. This equates to pin 15 on the Mega48 chip itself, which is bit 1 of port B. Plug the bare end of the wire into one of the connector strips on the breadboard. Next, take the test LED and plug the positive lead (the anode or longer lead) into the same connector strip and the negative lead (the cathode or shorter lead) into the connector strip opposite the center trough from the connector strip you first used. Take the extra 360Ω resistor (orange, blue, brown) and insert one lead into the same connector strip as the negative lead of the LED. The other lead of the resistor will plug into the ground bus strip (the same strip as the black wire from your PC Board's power connector.) Double-check that the negative lead of the LED is connected through the resistor to the GND.

There should now be one path from one crimp-on connector on J1 pin 15, through the LED (positive lead to negative lead) then through the resistor and to the ground strip of the breadboard and back to the ground of your controller's power connector.

Put four fresh AA cells into your battery pack, paying attention to the orientation of the cells. Place the stripped wires from the battery pack into the power screw terminal. Put the black lead into the negative (-) terminal and the red wire into the positive (+) terminal. Using a tiny flat-head screwdriver to tighten the screws to hold the wires in place. Turn the switch on the battery pack to the on position. The power LED on your controller will light.

Loading Your Program on the Controller

Plug your programming downloader cable into the printer port on your computer and then into J3, the 5-pin programming port on your controller.



If you are not using LPT1 on your computer, you will need to change the parallel port that BASCOM-AVR uses. To do this, click on the Options menu and select Programmer. Near the bottom of the Programmer window, find the LPT-Address and select the address of your parallel port. Every computer is

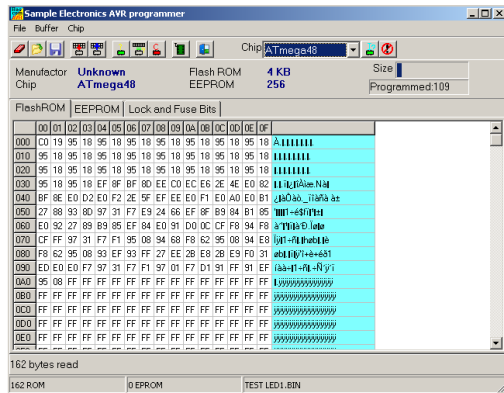
different, but most often LPT1 is located at address 378H. The address of LPT2 is often 278H and the other address (3BCH) is often LPT3. If you have a non-standard port, you may still be able to add it by using the Add port button. Check the BASCOM help file for more information.



In BASCOM-AVR, press the F4 key on your keyboard. The programmer window will pop up. If you get a window pop up saying that it could not identify the chip you have a problem.



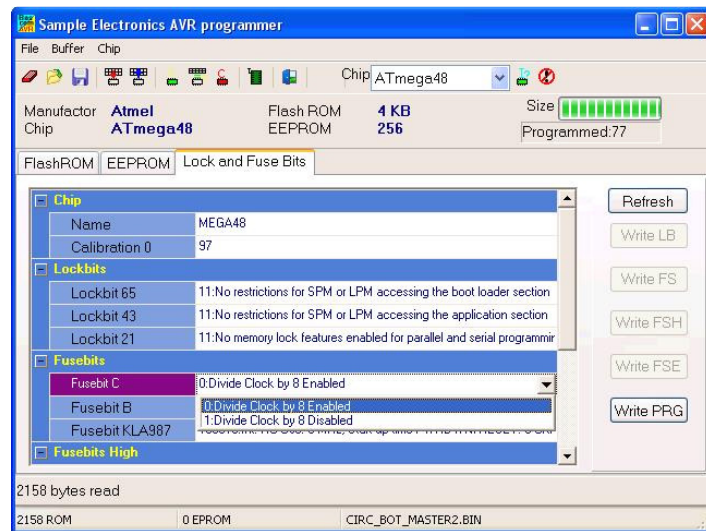
Check your cable and refer to the troubleshooting procedures at the end of this manual. When you get into the programmer, click on the Autoprogram button in the toolbar. This button looks like a green IC with a T next to where pin 1 would be. This will transfer your program to the Mega48 and then read the program back out of the chip to verify that it got programmed correctly.



Once the controller has been programmed, your test LED will start flashing on and off at about 1/2 second each. Congratulations, you have verified that you built the Chibot controller and programmed it correctly.

Before we continue on to further experimentation, we want to take this time to set up the clock speed on our Mega-48 chip. The chip comes preprogrammed to divide the clock speed by eight. This effectively slows down the processor. We are going to override that preprogrammed setting to give us greater computing speed and finer control of the robot.

In BASCOM, bring up the programmer again by pressing F4. However, this time, click on the Lock and Fuse Bits tab above the hex representation of your LED Test program. This brings up the dialog box to change the fuse bit settings. Find the setting under fuse bits labeled Fusebit C. It should show 0:Divide Clock by 8 Enabled. Click in that field to change the setting to 1:Divide Clock by 8 Disabled. Changing this setting will enable the Write FS button on the right. **CAUTION: Do not change any other settings at this time!** Changing some of the other settings on this screen can make the controller non-responsive, effectively destroying it for our purposes. Now, click on the Write FS button to save the new clock setting. The microcontroller is now set to 8 MHz.



Change the \$crystal command in your LED Test program to 8000000 and use that value for any programs which you write for that particular chip from now on. This will give you a faster processor and therefore much finer control over your robot.

Some More Experiments

- Try changing the Waitms command to make the LED flash faster and slower.
- Play with the \$crystal command by changing the 8000000 up or down to get the timing more accurate. (Hint: The clock that runs the Chibot controller is called an RC tank. These timing circuits are not very accurate – thus the option to use a crystal, which is much more accurate. Modify the \$crystal number to try to make it more accurate for your controller.)
- The Set command sets the pin to +5V or high, and the Reset command sets the pin to 0V or low. Use these commands to turn the LED on for a set amount of time and then back off. (hint: look in the Help file for Newbie Problems for information about Port and Pin commands)
- Using the Set and Reset commands, make the LED spell your name out in Morse code (look the code up on Google, Yahoo or another Internet Search Engine) (hint: a dot is 100 milliseconds and a dash is 300 milliseconds, space between the dots and dashes of a letter is the same as a dot and space between letters is the same as a dash.

Note: Any time that you begin to change a program to a new program (as opposed to clicking on New), begin with clicking on File and then clicking on Save as... and give the program a new name.

Let's Hire Some Muscle

Frame/Plate Set

There is a company, originally from Japan, which has been supplying hobbyists with quality parts and kits for many years. Tamiya USA was founded in 1989 in Los Angeles, California, and continues to be one of the biggest hobby companies in the world. The body or chassis of our TableTop Bot is made up of one of Tamiya's Universal Plate Sets.

Gearbox

Tamiya also makes a line of inexpensive but useful gearboxes. The hardest part of using the Tamiya gearboxes is the assembly. The instruction sheet is very compact, making it hard to understand and follow. So to make everyone's life a little easier, this guide details the steps required to assemble the Double Gearbox using the 38.2:1 gear ratio.

Step 1 - Unpacking the Parts - The Double Gearbox comes unassembled with a 1-page instruction sheet. The sheet shows an exploded view of the gearbox for each gear ratio. This type of assembly instructions can be frustrating since it doesn't really tell in what order to assemble the parts.

In the box, you'll find:

- The Instruction Sheet
- 2 bags containing plastic gears
- 1 bag containing the motors, axles and screws
- The plastic gearbox housing



The kit contains many small parts that can be easily dropped and lost. I recommend using a small bowl or other container to hold the parts while you're assembling the gearbox. One lost pinion gear and you're stuck!



As I had mentioned, there are a lot of small parts in this kit. The picture to the right can help you identify the different parts.

From left to right, each row:

Purple Pinion Gear

Yellow G2 Crown Gear

Blue G3 Gear

Yellow G4 Final Gear

Aluminum Spacer

Short Round Steel Axle

Yellow Long Plastic Bushing

Yellow Short Plastic Bushing

Brass Eyelet

Brass Gear Hub

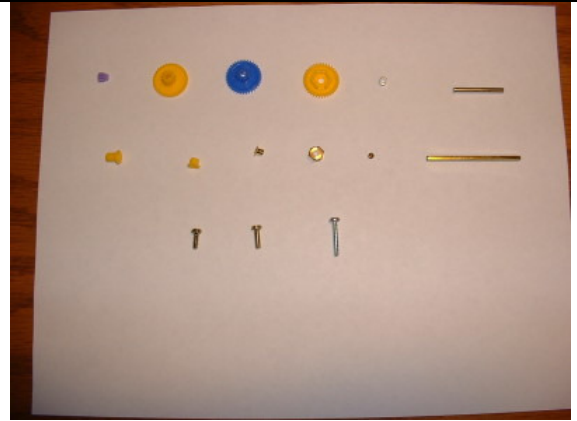
Brass Set Screw

Long Hexagon Steel Axle

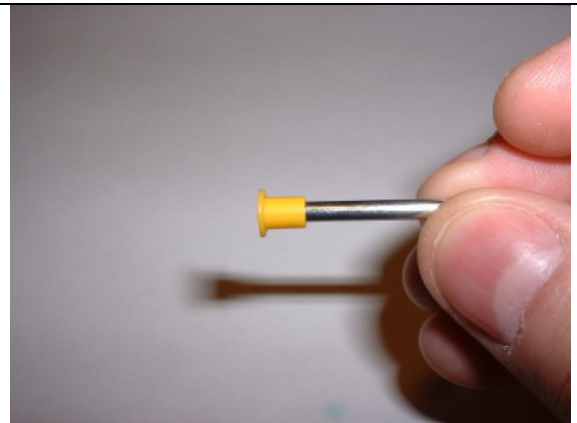
Short Machine Screw (not used)

8mm Mounting Screw

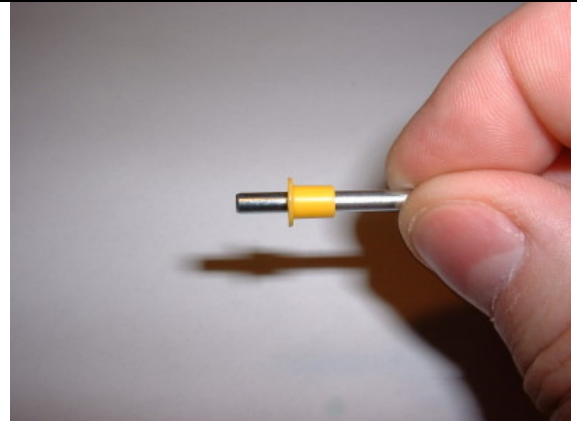
18mm Tapping Screw



There are 4 axles used in the gearbox, 2 for each half of the Double Gearbox. Each axle needs a bushing placed on it. We will do all 4 axles together. Start with the small steel axle and place the long yellow bushing on the end as shown in the picture.

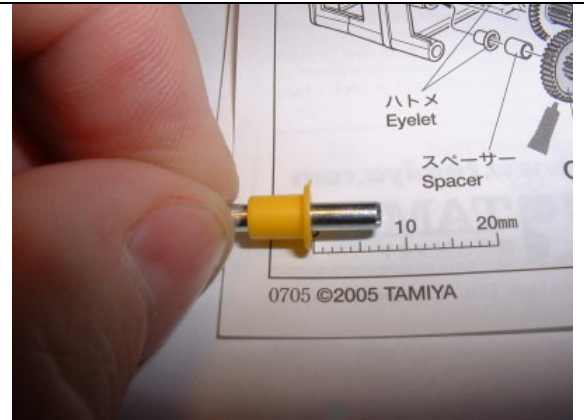


Next, slide the bushing on until about 7mm of axle is protruding beyond the bushing (don't worry about getting it exact, we'll measure it in the next step).



Use the metric ruler at the bottom of the instruction page to measure the axle to ensure that 7mm is protruding.

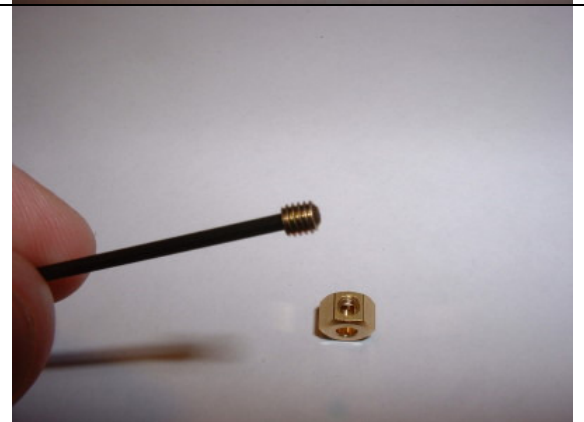
Repeat this process with the second short axle.



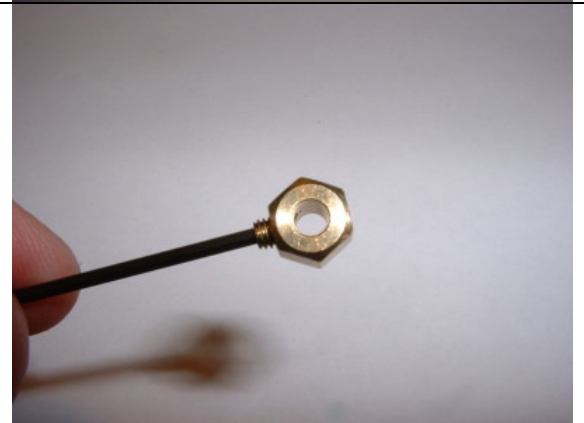
Locate the gearhub, set screw and hex wrench. The gear hub is held in place by the small set screw (called a grub screw in the instructions).



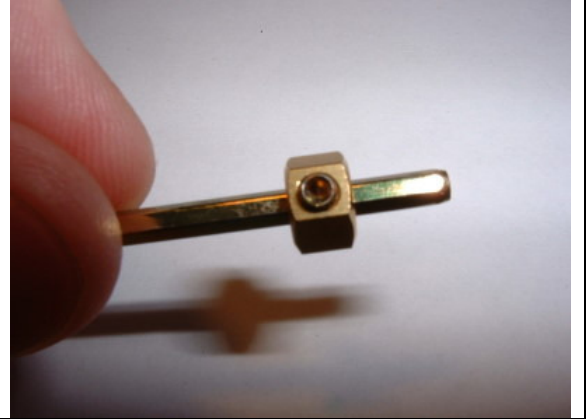
The set screw is very small. to make it easier to install the set screw into the gear hub, place it on the end of the hex wrench as shown in the picture.



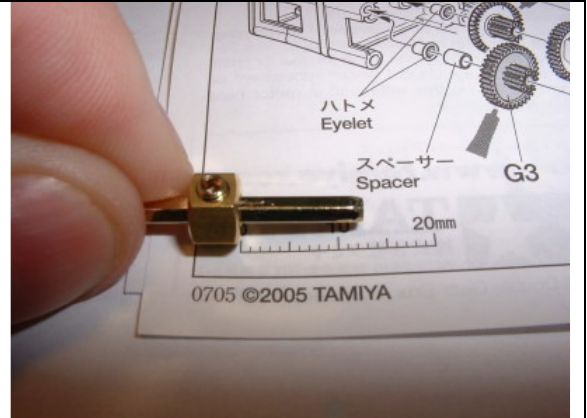
now thread the set screw into the hub 1-2 turns. If you thread it in too far, the hub won't slide onto the hex shaft.



Now slide the gear hub onto the hex shaft. If the hub won't slide on, back out the set screw until it does. Snug the screw slightly but don't tighten it yet.



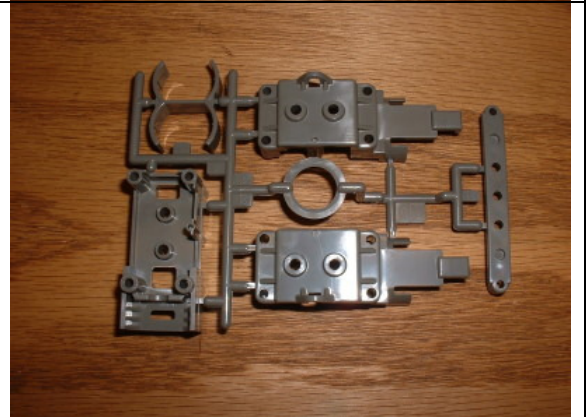
Use the metric ruler on the instructions sheet to measure the amount of axle protruding from the gear hub. Slide the hex axle until 13mm is showing. Tighten the set screw firmly.



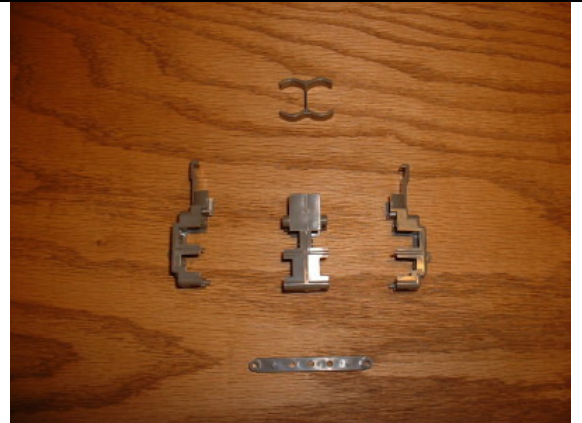
Repeat the process for the other 2 axles and set them aside. We will use in the next couple of steps.



Locate the gearbox housing parts as shown in the picture. The parts need to be separated from the plastic frame.



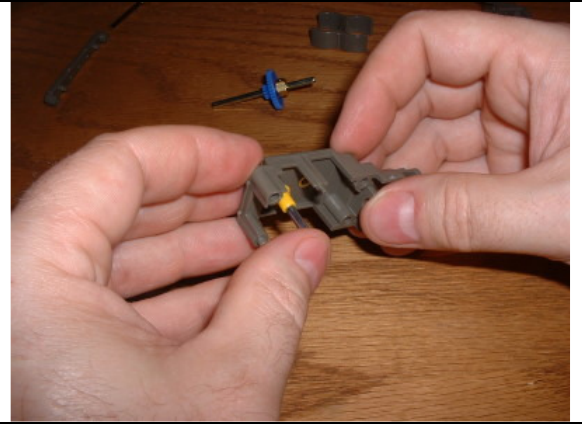
Use a utility/hobby knife or a pair of close cut snips to remove each piece. Clean off any excess plastic that may be left where the parts were attached to the plastic frame.



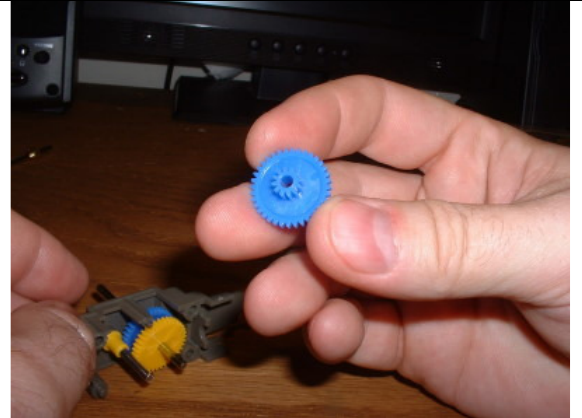
Locate the left side housing part (use the picture as a reference) and 2 of the brass eyelets. Place the eyelets into the holes as shown in the picture. Also place 2 of the brass eyelets into the holes of the middle piece of the housing.



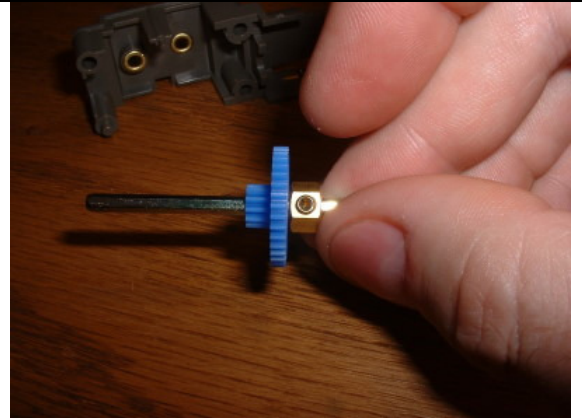
Place the short side of the round shaft into the left side hole as shown in the picture.



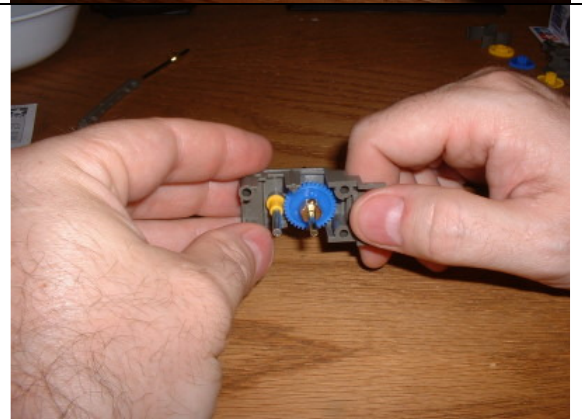
Locate one of the blue G4 gears, using the picture as a guide.



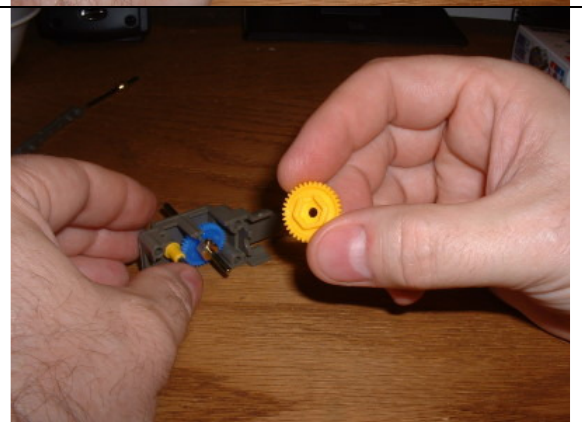
Hold the short end of the hex shaft and slide the blue gear onto the long end as shown in the picture. Be sure that the large gear is facing the gear hub. The gear goes onto the long side of the shaft.



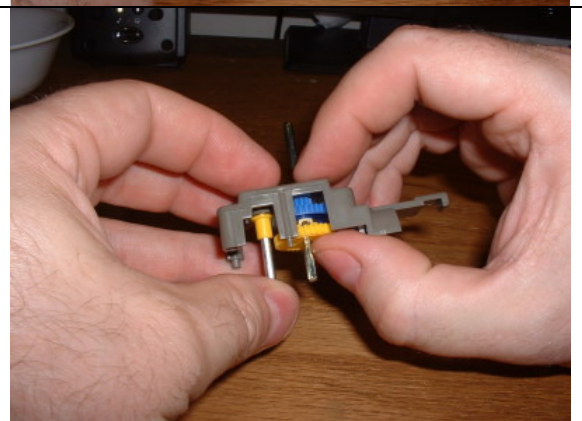
Slide the hex shaft into the right hole as shown. The short side of the 2 axes should now be about lined up with each other.



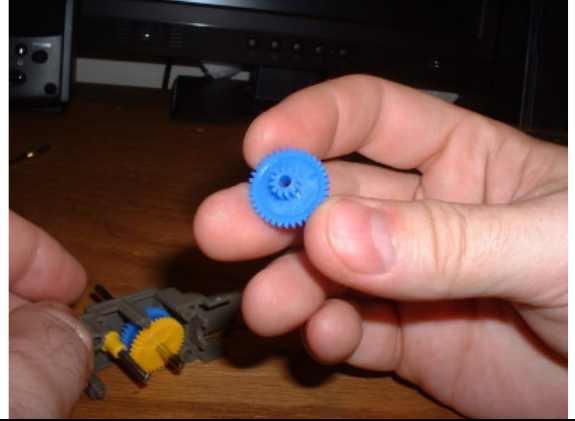
Find the yellow G4 gear, it has a hex shaped protrusion on one side that will fit onto the gear hub on the hex axle.



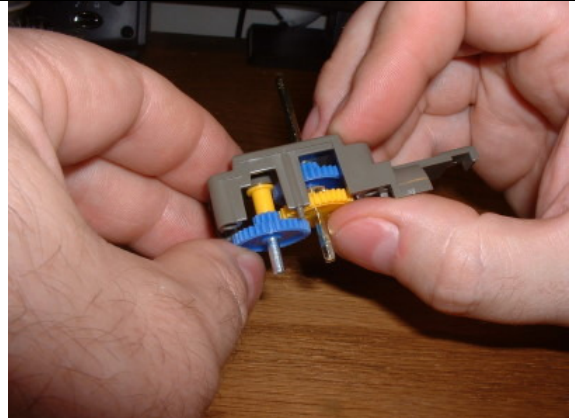
Slip the yellow gear onto the hex shaft as shown. Be sure to align the set screw with the slot in the G4 gear. You should be able to clearly see the set screw. If not, rotate the gear until set screw can be seen.



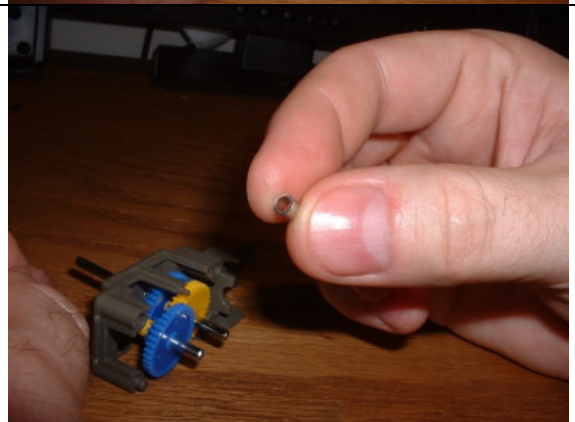
Select another blue G3 gear.



Place the blue G3 gear onto the round steel axle as shown in the picture. The teeth of the smaller gear on the G4 should mesh with the teeth on the larger gear on the yellow G4 gear.



Find the small aluminum spacer - refer to the picture.



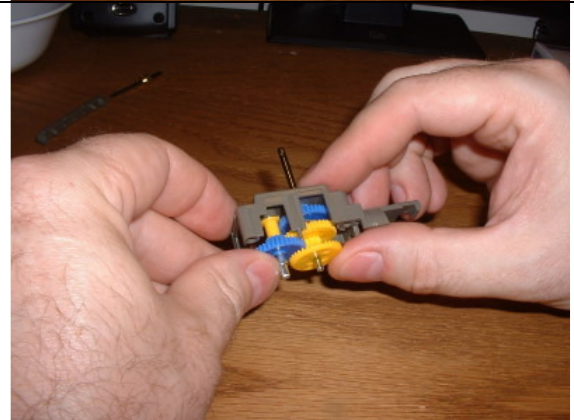
Slide the spacer onto the round steel axle as shown in the picture. The spacer slides up against the blue gear.



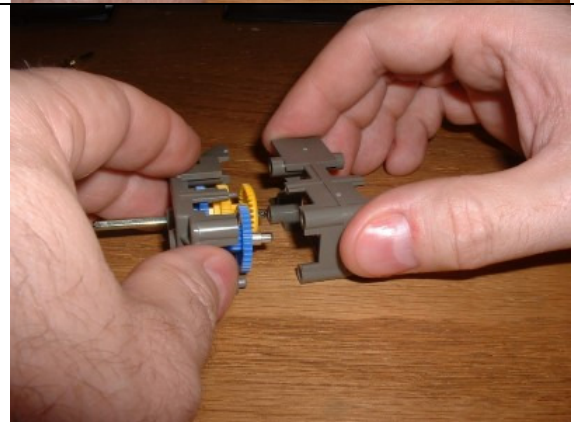
Find the yellow G2 gear. This gear has teeth on the face instead of the edge.



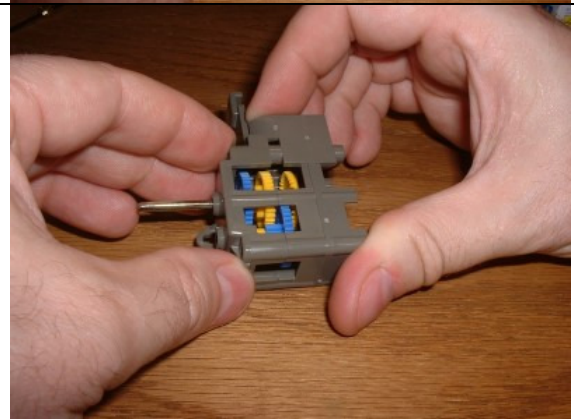
Place the yellow G2 gear onto the hex shaft as shown in the picture. The small gear on the G2 should mesh with the larger gear on the G3



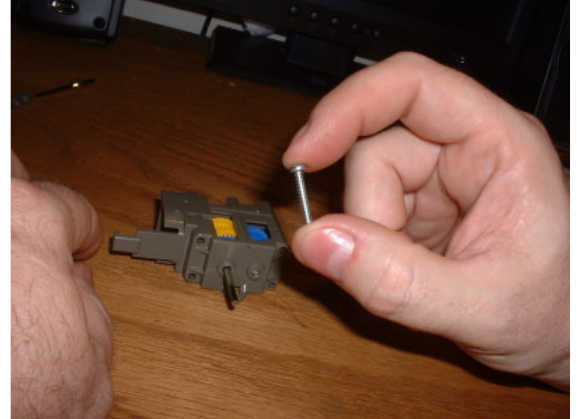
Now we can slide the 2 housing parts together. Make sure the 2 axle holes in the middle housing section have the brass eyelets in place so the axles will fit into the brass eyelets and not into the plastic.



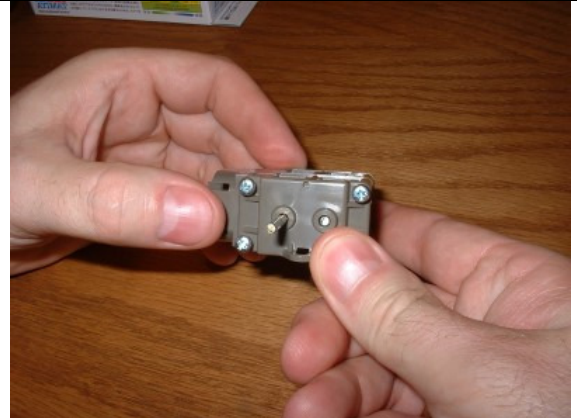
Once the 2 axles slide into place, the 2 housing parts should fit together easily. Do not try to force the parts if they don't easily slide into each other. check the axles and make sure the gears are aligned as shown in the pictures above. If you still can't get it fit together, pull the sections apart and reassemble the gears as shown the guide so far.



Find 3 of the 18mm coarse threaded screws like the one shown in the picture.



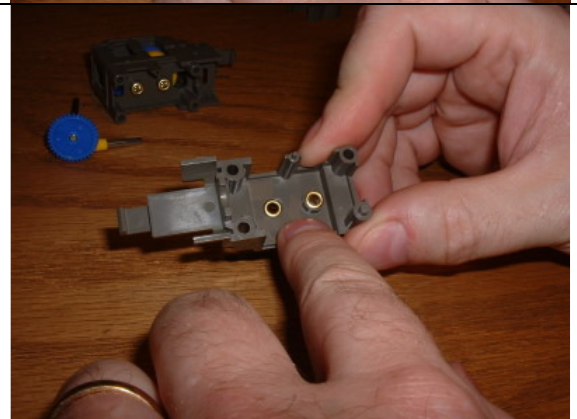
Place the 3 screws into the holes on the outside housing as shown in the picture. There are actually 4 holes but only 3 will accept the screws. The fourth hole is under my thumb in the picture.



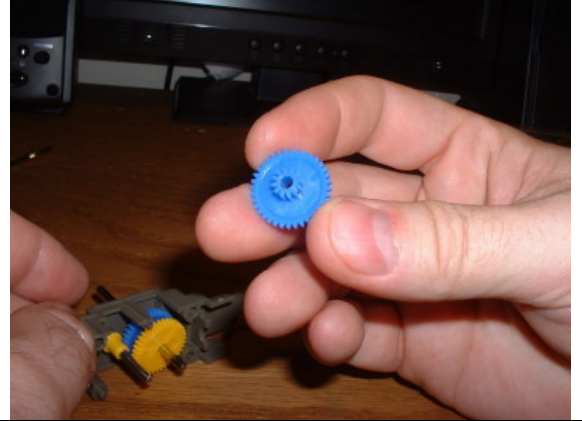
Place 2 of the brass eyelets into the holes of the middle piece of the housing as shown in the picture..



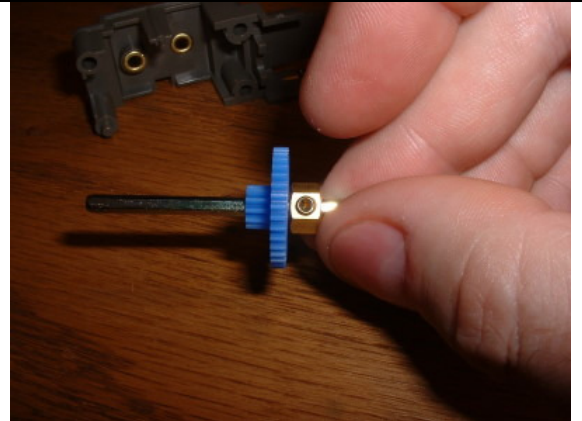
Locate the right side housing part (should be the last part left) and the last 2 brass eyelets. Place the eyelets into the holes as shown in the picture.



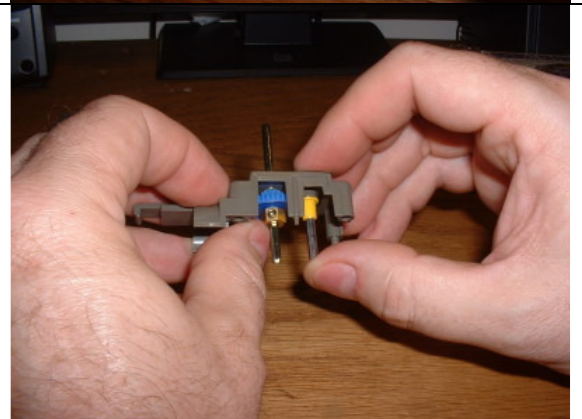
Locate one of the blue G4 gears, using the picture as a guide.



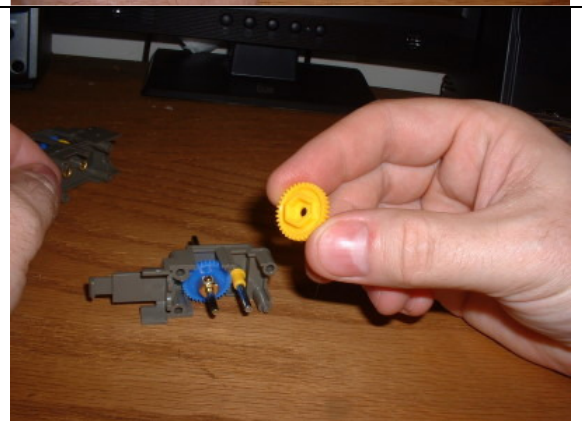
Hold the short end of the hex shaft and slide the blue gear onto the long end as shown in the picture. Be sure that the large gear is facing the gear hub. The gear goes onto the long side of the shaft.



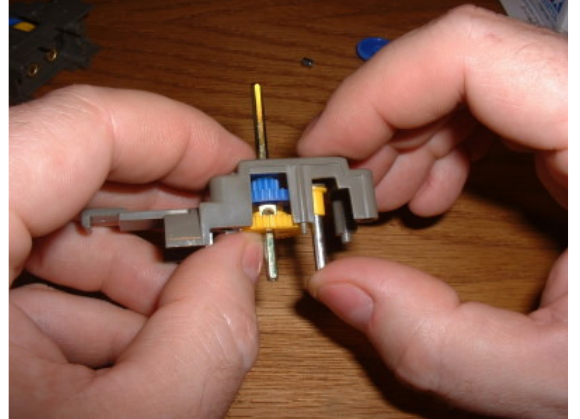
Slide the hex shaft into the left hole as shown. Place the round steel axle in the right hole as shown. The short side of the 2 axles should now be about lined up with each other



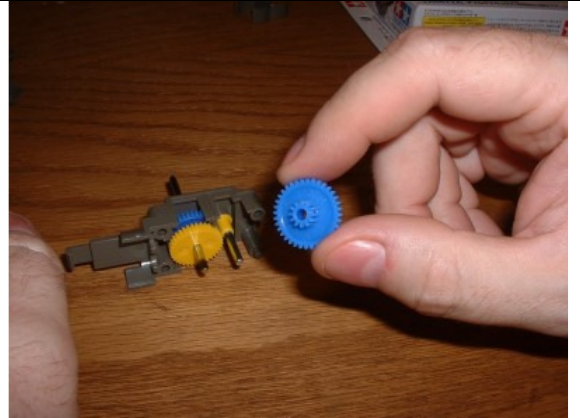
Find the yellow G4 gear, it has a hex shaped protrusion on one side that will fit onto the gear hub on the hex axle.



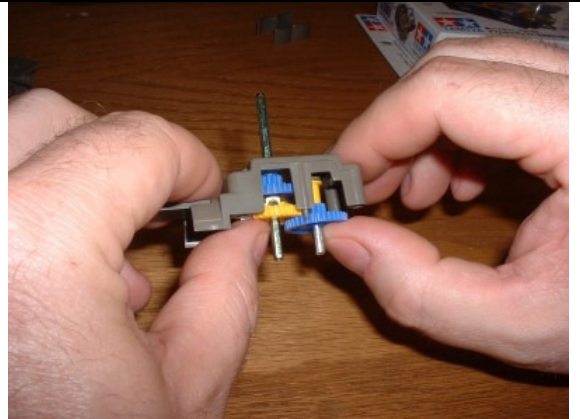
Slip the yellow gear onto the hex shaft as shown. Be sure to align the set screw with the slot in the G4 gear. You should be able to clearly see the set screw. If not, rotate the gear until set screw can be seen.



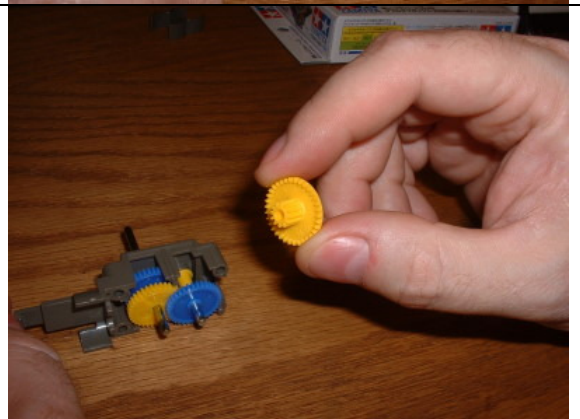
Select another blue G3 gear.



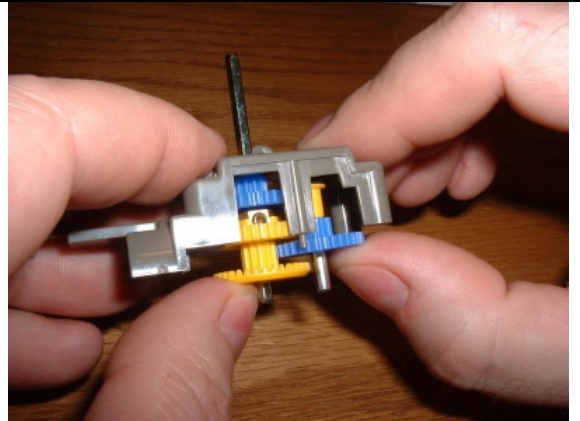
Place the blue G3 gear onto the round steel axle as shown in the picture. The teeth of the smaller gear on the G4 should mesh with the teeth on the larger gear on the yellow G4 gear.



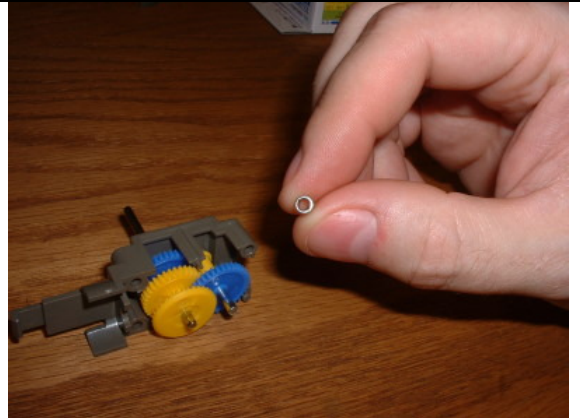
Find the yellow G2 gear. This gear has teeth on the face instead of the edge,



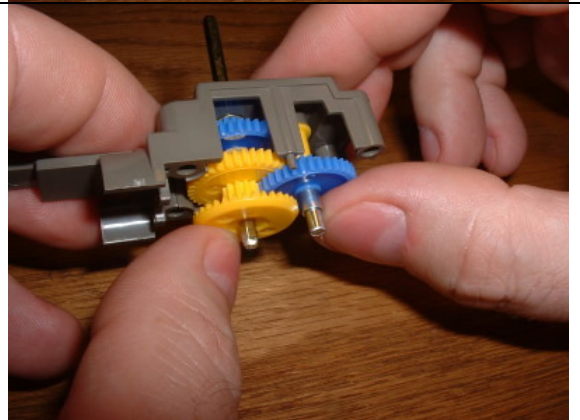
Place the yellow G2 gear onto the hex shaft as shown in the picture. The small gear on the G2 should mesh with the larger gear on the G3.



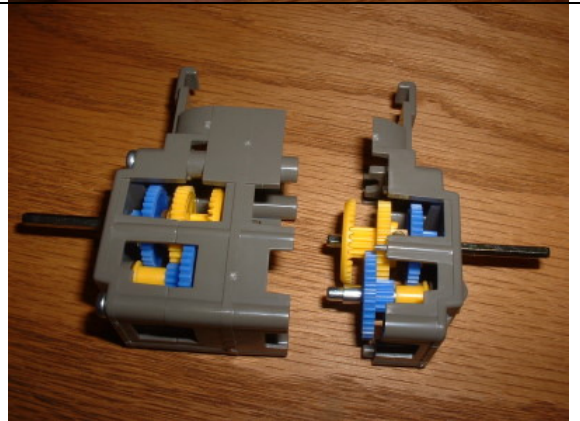
Find the small aluminum spacer - refer to the picture.



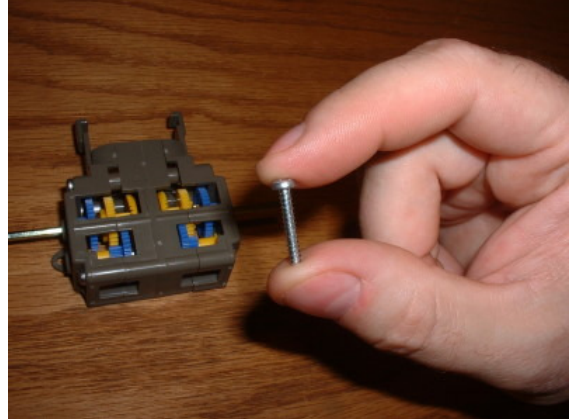
Slide the spacer onto the round steel axle as shown in the picture. The spacer slides up against the blue gear.



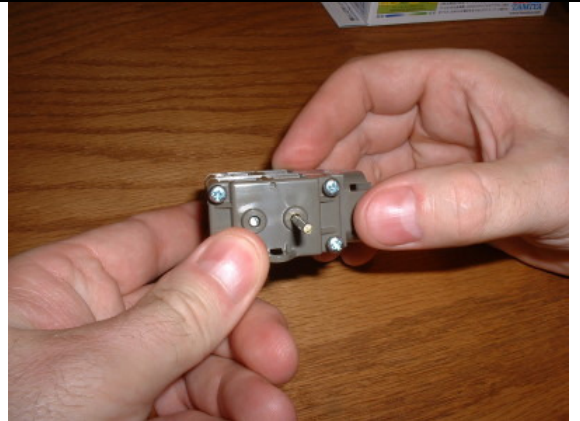
Now we can slide the 2 housing parts together. Make sure the 2 axle holes in the middle housing section have the brass eyelets in place so the axles will fit into the brass eyelets and not into the plastic.



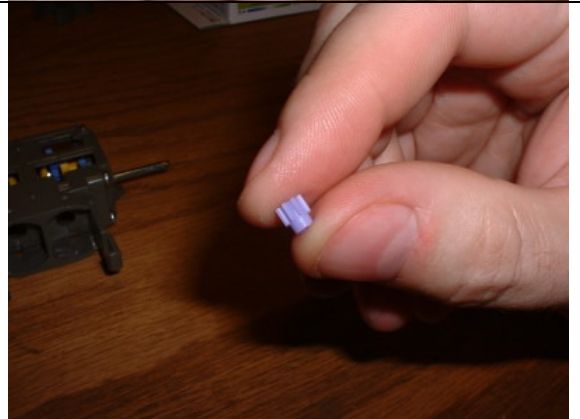
Find 3 of the 18mm coarse threaded screws like the one shown in the picture.



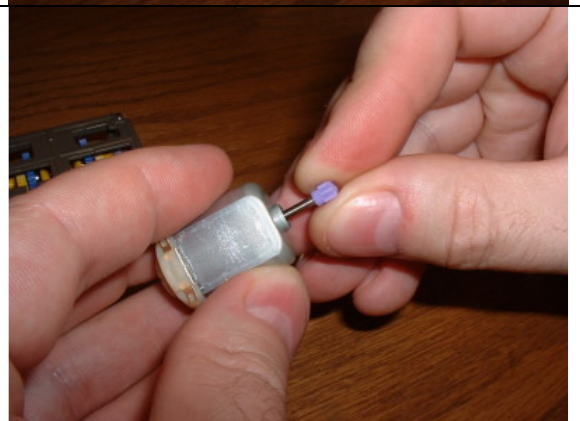
Place the 3 screws into the holes on the outside housing as shown in the picture. There are actually 4 holes but only 3 will accept the screws. The fourth hole is under my thumb in the picture.



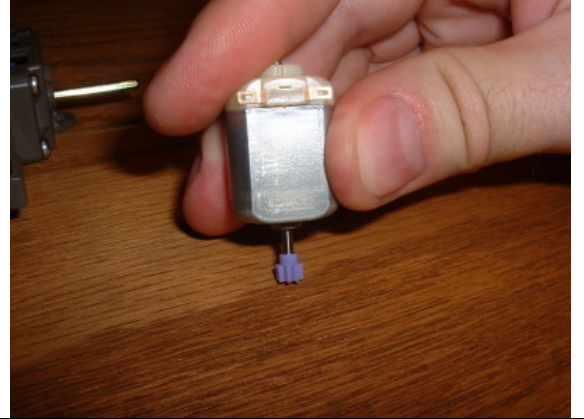
Locate the small light purple spur gear - G1. There are 2; one for each motor.



Place the G1 gear on end of each motor shaft as shown in the photo. Slide it far enough on that it stays in place.

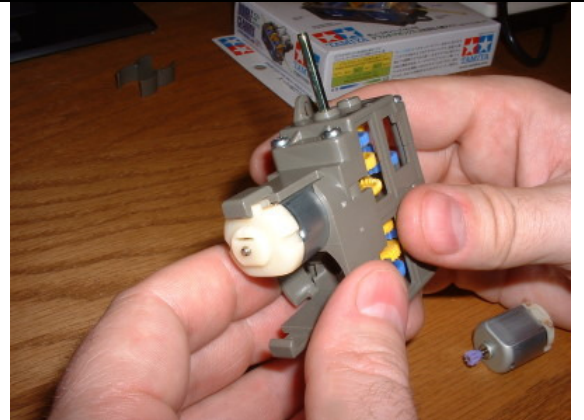


Hold the motor in your hand and place the G1 gear on a flat surface as shown in the picture. Press down on the motor until the motor shaft slide all the way through the G1 gear. The motor shaft should be flush with the gear. Repeat this process for the second motor.

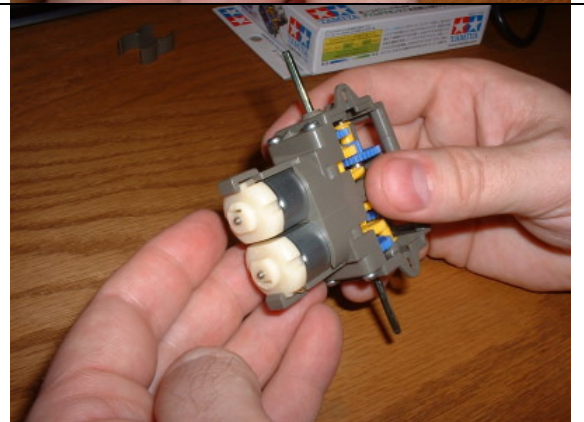


Place the motor into the gearbox housing as shown in the picture. The motor should slide into the slot easily and should be held in place by the plastic tab that hooks over the end of the motor.

The motor leads should be towards the outside to make it easier to solder wires to them later.



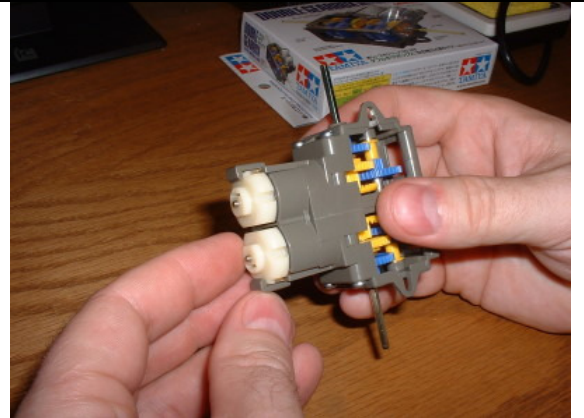
Place the second motor into the gearbox housing. Make sure the motor leads are towards the outside when placing the motor.



Locate the plastic motor spacer. This piece slides in between the two motors to help secure them in place.



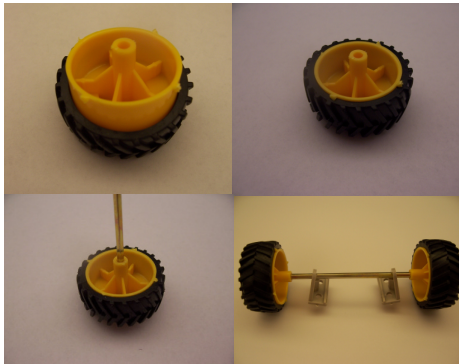
You now have a completely assembled the Tamiya double gearbox!



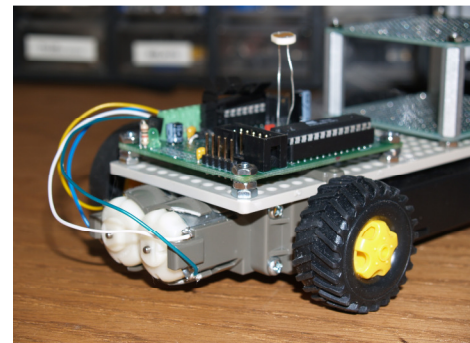
Now take four wires of 2"-3" (5-7 cm) each and strip about 1/4" (1/2 cm) of the insulation from each end of each wire. Solder one end of each wire to one of the terminals on the motors.

Tire Set

Follow the directions for the Tire Set kit from Tamiya. Basically you will simply cut the tires and wheels off the molding and then push the tires on over the wheels. Once that is done, place one wheel on each end of the



motor/gear box axle, place one wheel on a table with the other wheel up, and then gently push the top wheel down so that both wheels slide onto the axle.

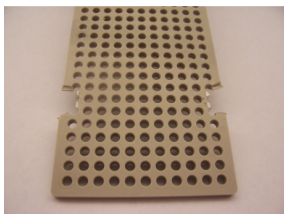


Repeat this with the other axle, but make sure that you slide the axle through the two brackets, which

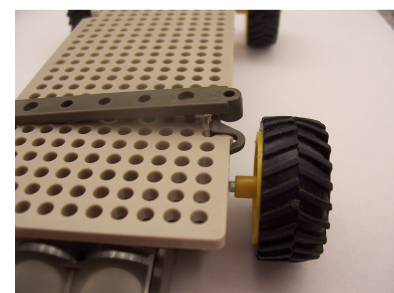
will hold the axle on to the chassis.

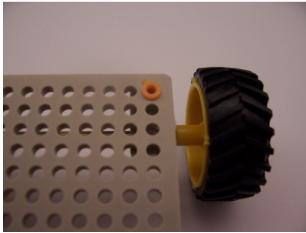
Assembling the Chassis

First, prepare the chassis plate for the gear/motor assembly by using a wire cutter to notch the plate as shown in the photo. Cut the plastic from the edge to the second hole in the sixth and eighth rows from the rear. Do this on both left and right sides near the back of the body. The

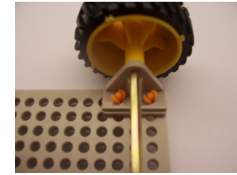


motor/gear box assembly attaches to the chassis with a flat, narrow plate which screws to the motor/gear box after being "strapped" across the chassis board. Mount the gearbox so that the two motors stick slightly out the back of the robot. This will be a pretty snug fit and may take some work to get it mounted correctly. Try screwing one screw in



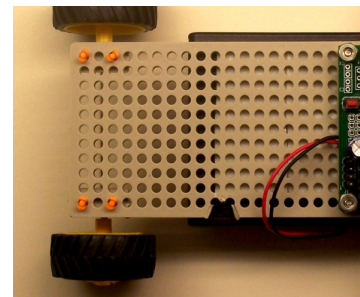


slightly and then sliding the gearbox assembly and the “strap” over the chassis board like a paper clip, and then put the other screw in. Finally tighten both screws down securely.



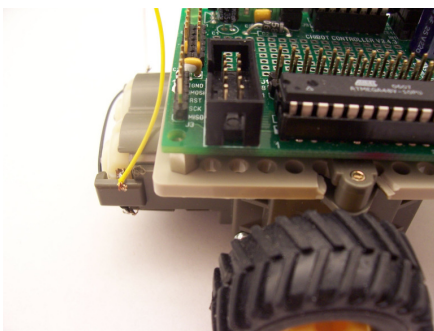
The front wheel mounts connect to the chassis with the small, orange, plastic rivets. Place the axle mount against the chassis plate as shown in these pictures and then push one of the rivets with the hole through the center through the chassis plate and then through the axle mount. Then push the smaller rivet inside the center hole of the first rivet and push in until it locks the pair in place, securing the wheel mount to the chassis plate. See the photographs in the previous section to see how the wheels mount to the chassis.

Finally, using a small wire cutter or sharp hobby knife, cut a notch in the chassis board for the power switch on the battery box. This notch will be on the right side as you look down, and will be from the eleventh to the twelfth holes as in the picture. Then using some of the double-sided foam tape, mount the battery box so that the power switch sticks up through that notch.



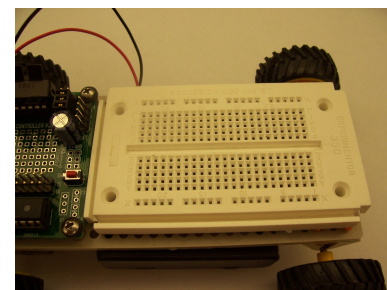
Connecting the Board to the Chassis

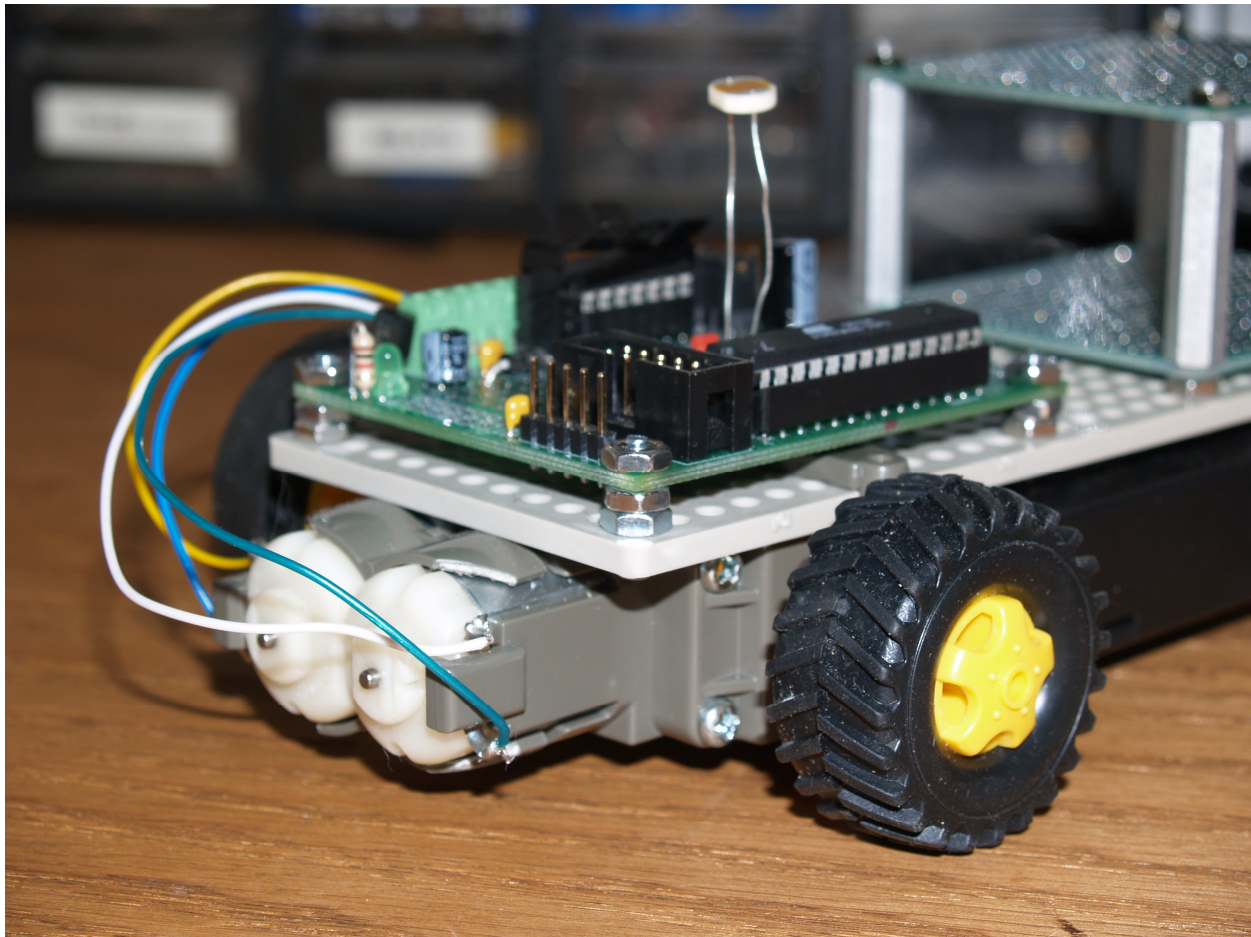
The controller board connects to the chassis with the four nut/spacer/washer/screw sets as you can see in the photograph below. Push the screw through the chassis, then put the spacers on them and then slide the PC Board over the end of the screw and tighten the nut down.



Finally, using the double-sided foam tape, mount the breadboard on the front of the chassis plate above the battery box. Be sure

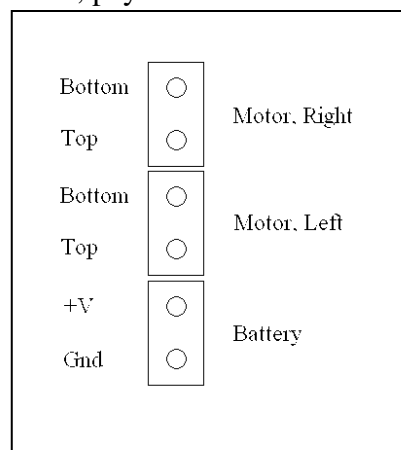
to leave enough room on the side of the breadboard to be able to access the power switch. Now take the red and black wires from the controller and plug them back in to the breadboard power bus strips as when you tested the controller.





Finally, connect the battery box and the motors to the terminal connectors using the same technique as in the controller testing section earlier in this book.

When you plug the motor wires in to the screw terminal connectors, pay attention to the orientation of each wire. The wire soldered to the bottom terminal of each motor should go to the screw terminal towards the front of the robot. The wire soldered to the top terminal of each motor should go to the rear screw terminal. This will ensure that the motors both run the same direction for the same commands. If after you hook up the motors, your robot goes backwards when you think that it should be going forward, check these wires and make sure that the motors are wired up correctly.

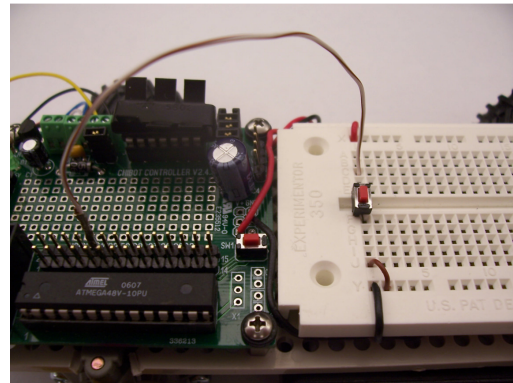


Sample Programs

There-And-Back

This is a sample program to run the There-and-Back contest. Since every system is slightly different, you will want to play with the delay times to try to get your positioning perfect. Do not run this on a tabletop until you get the timing delays down pat. Also, I noticed that my motor/gear box had a slight drift when I simply turned off the motors, and the drift after running forward was different than the drift after running in reverse. I got much better accuracy when I applied full opposite power for a very short time (about 250 – 300 milliseconds) immediately after a run. This reverse power acted as a brake, and improved my repeatability greatly.

This program has a Do loop near the beginning to delay until you press the Start button. To connect the button, take the test button and place it into the breadboard crossing the center trough. Next take a very short piece of wire, strip both ends and place it between ground and the connector strip for one pin of the button. Next take the wire with the female crimp-on pin that you made for testing the controller, and connect it to pin 24 of J1. Plug the other end in to the breadboard connector strip with the other side of the switch.



The Start button on the breadboard

After the initialization, the heart of this program is the setspeed subroutine. This routine centralizes the control for running the There-and-Back course. It ties the left and right motors together and allows the main program to simply call the subroutine with a speed and not worry about the two motors.

```
'Chibots Tabletop Bot
'Author: Art Granzeier, Wright Hobbies Robotics
'This program is designed to verify operation of the Chibots TableTop Robot Kit
'For More information, please visit http://www.wrighthobbies.net/bots/ttbot

'The following parameters can also be set under Options/Compiler/Chip
$regfile = "m48def.dat"
$crystal = 8000000
$hwstack = 32
$swstack = 10
$framesize = 40
' specify the used micro
' used crystal frequency
' default use 32 for the hardware stack
' default use 10 for the SW stack
' default use 40 for the frame space

' Do you have anything to declare?
Declare Sub Setspeed(byval Speed As Integer)
Dim Speed As Integer , X As Byte
' Speed between -127 and +127

Button Alias Pinc.1
Leftenable Alias Portb.0
Rightenable Alias Portd.7
'Enable/disable line for left motor
'Enable/disable line for right motor

'Config Statements - Determine how I/O lines and other devices will behave
Config Portd = Output
Config Portc = Input
Config Portb = Output
Config Timer1 = Pwm , Prescale = 1 , Pwm = 8 , Compare A Pwm = Clear Up , Compare B Pwm = Clear Up

'Main Program

' Initialize ports
Rightenable = 0
Leftenable = 0
Portc.1 = 1
'Disable Right Motor
'Disable Left Motor
'Set Internal Pull-Up Resistor

Do
  X = Button
' Don't start until the button is pressed
```

```

Loop Until X = 0

Wait 1

'Start - Run Forward
Speed = 127
Call Setspeed(speed)
'
Waitms 1650                                'Run for 1.65 second

Rightenable = 0                            'Disable Right Motor
Leftenable = 0                             'Disable Left Motor

Speed = -127
Call Setspeed(speed)

Waitms 250                                'Full Reverse to Brake
Rightenable = 0                            'Disable Right Motor
Leftenable = 0                             'Disable Left Motor

' Delay for turnaround
Waitms 1000                                ' Pause for 1 sec

'Start - Run Reverse
Speed = -127
Call Setspeed(speed)

Waitms 1650                                'Run for 1.65 second

Rightenable = 0                            'Disable Right Motor
Leftenable = 0                             'Disable Left Motor

' Brake
Speed = 127
Call Setspeed(speed)

Waitms 250                                'Full Reverse to Brake

Rightenable = 0                            'Disable Right Motor
Leftenable = 0                             'Disable Left Motor

End                                          'end program

Sub Setspeed(byval Speed As Integer)
    Speed = Speed + 128
    Pwmla = Speed
    Pwmlb = Speed
    Rightenable = 1
    Leftenable = 1
End Sub                                    'Set duty cycle
                                          'Turn on right motor
                                          'Turn on left motor

```

Improvements:

- Try to tighten up the front axle mount. Sloppiness here can cause the 'bot to swerve, making it come back to a different position from the origin.
- You can start with the same time for forward as reverse, but will need to modify the forward time to reach the stopping point and then play with the reverse time for the back leg. – *Caution: test the program on a smooth (no rug) floor until you get the distances correct.*
- I found that my 'bot tended to coast differently after backing than forward. This made it either pass up or not reach the origin. Try running the motors at full backwards speed for a quarter second (250 ms) after the forward run for braking. Use forward braking after the reverse run.
- Try modifying the left and right speed to modify the left/right positioning.

Future Projects

As we mentioned before, the TableTop Contest is only the beginning of what you can do with this kit. You are supplied with everything that you need to compete in the There-and-Back contest, but your TTB is a wonderful foundation for future projects in robotics.

PhotoVore

If a carnivore is an animal that looks for and eats meat (carne) and an herbivore is an animal that looks for and eats plants (herbs), then we could call a robot that hunts for, or moves toward, light (photo) a photovore.

With the addition of an extra photocell, you can reposition the two cells to look for light towards the front and angled 45 degrees out left and right. Connect the photocells up to the on-board Analog-To-Digital (A/D) ports of the Mega-48 chip and then take readings on each cell. Compare the two readings (left and right) and have your robot turn towards the direction of the sensor which has a higher reading (more light). In order to enable your 'bot to turn, you will need to remove the front wheel assembly (remember, the front wheels are to *prevent* turning). Wright Hobbies has a roller ball assembly with which you may replace the front wheel. That roller ball assembly will allow your 'bot to turn, using differential turning. This is done by running the motors at different speeds or even in different directions. Start off with very slow speeds until you get the hang of things.

To get more information about how to do this, check out the Wright Hobbies example of reading a potentiometer at: www.wrighthobbies.net/examples/readingpot.htm. Also take a look at the Mega-48 manual at: http://www.atmel.com/dyn/resources/prod_documents/doc2545.pdf and the BASCOM-AVR manual in your help system of BASCOM or at: http://www.mcselec.com/index.php?option=com_docman&task=doc_download&gid=140&Itemid=54

PhotoPhobe

Like a photovore, a photophobe is a robot which is affected by the light. In this case, the robot goes away from the light. A phobe is a fear, as in hydrophobe being someone who is afraid of the water.

The programming would be very much the same as the photovore, except that you would measure the two light intensity levels and turn the robot *away* from the direction of the higher light level.

Line Follower

The line-following contests are just like their names imply. Your robot is placed on a course and is expected to follow a line drawn on the floor. To “see” the line, you must aim your photocells down towards the ground. When light is reflecting up, that means that the line is not detected (for a dark line). When your 'bot detect the line in the left sensor, you must move more to the right to get the robot centered again. Conversely, when it detects the line in the right sensor, it must move more to the left.

This is actually harder than it seems at first. Speed has a lot to do with your ability to detect the line; the faster your robot goes, the easier it is to skip over the line and “jump the track.” Also, you may want to use more than two sensors, and when the first left sensor detects the line move a little to the right, and when the second (or more) left sensor detects the line, move faster to the right. The little bot at: <http://elm-chan.org/works/lrc/report.html>, actually has six sensors, and is pretty cool. Google “line following robots” for more information.

For some discussions about line following robots, look at:
www.wright hobbies.net/guides/linefollower.htm.

Ideas for Your Own Programs

Having been a programmer since 1975 and having taught numerous programming courses, I can say with certainty that one of the best ways to learn to program is to take someone else’s program and study it and then try to write your own version of that program. In the same way, you will learn a lot by studying other’s microcontroller/robotics/engineering projects and trying to duplicate their work in your own way.

Some great sites to look for project ideas are:

Wright Hobbies Forum: <http://www.wright hobbies.net/phpbb2/index.php>

The PIC List Contest: <http://www.piclist.com/techref/piclist/pcbcontest.htm>

The List of Stamp Apps: <http://www.hth.com/losa>

Cornell - ECE476 Final Projects: <http://instruct1.cit.cornell.edu/courses/ee476/FinalProjects/>

Hack-A-Day Robotics Page: <http://www.hackaday.com/category/robots/>

Arrick Robotics Robot Menu: <http://www.robotics.com/robomenu/index.html>

The Robotics News Group: <news://comp.robotics.misc>

uC Hobby Site: <http://www.uchobby.com>

There are gigabytes of information about robotics and robotic projects on the internet. Just try looking on any search engine for robotics projects.

These sites/groups will give many, many projects, not just on robotics, but also on microcontrollers and computing in general also. Remember that not only do you have a great introductory robotic kit, but you also have a powerful general-purpose microcontroller kit in your ChiBot controller. To temporarily disconnect your ChiBot controller from your Table Top Bot, just pull all four shorting jumpers off of J8 in the upper-right corner of your PCB. This will remove the FAN-8200 chip from the Mega-48 and allow you to use PB0, PB1, PB2 & PD7 in your design. Also, remove the jumper at J9 to remove power from the FAN-8200, this will reduce battery drain, and prevent the FAN-8200 from doing unexpected things. Be sure to keep your shorting jumpers so that you will be able to get back into robotics quickly. Better yet, you can purchase an extra ChiBot controller and another MEGA-48 from Wright Hobbies to continue your controller training without interfering with your TableTop Bot.

Troubleshooting

The main idea in troubleshooting anything is to try to narrow down your search. Understand what we covered at the beginning of this book about the major sections of your robot, and how they interact with the other systems. For example, If your robot won't do anything at all (including lighting up it's power LED), look for the trouble in your power system. That is the only system which could cause everything to die. Other things to look for would include:

Solder joints – One of the leading causes of troubles in an electronic kit is poor soldering. If you are having any troubles, please recheck all of your solder joints. It would be a good idea to use a magnifying glass to examine the connections looking for both cold solder joints as well as solder bridges. Cold solder joints occur when you do not get one or more of the metal contacts heated properly and the solder does not flow onto the contact. This will appear as if the solder had pulled away from the pin or the circuit pad. Correct these by simply reheating the joint with a hot soldering iron to reflow the solder. Solder bridges are excess solder which flows off of the circuit trace and onto a neighboring trace or pin. These can be fixed by removing the excess solder with some solder wick. Often, just re-melting the solder with a hot iron will allow the excess to flow onto the iron.

Power – Battery / polarity – One of the easiest troubles to miss is the power leads being reversed. Be very careful when hooking up your battery pack. Getting the polarity wrong can possibly destroy your controller. If, after connecting the power leads, you turn the switch on and the LED does not light up, ***IMMEDIATELY*** turn the battery pack off and recheck your leads. A related trouble with the power is, of course, the battery being of too low power. I mention this because I see it so often in troubleshooting systems. I prefer to use the rechargeable AA cells available from Wal-Mart for only about \$10.00. These are only 3 times more expensive than non-rechargeable, but can be used over 500 times.

Compile after every change – If you modify your program and then your robot doesn't seem to use the new changes, try to recompile and download the program. It's amazing how often even experienced people miss this one (even those of us who have been in the field for decades.)

Motor only runs one direction, check J8 shorting jumpers – While working with my 'bot, it had a rough time of one of my trips. It seems that my poor robot got banged around pretty bad. While trying to get it up and running again, I noticed that the left motor would only go forward and would not reverse. It also seemed to be running a bit sluggishly and not up to it's normal speed. After checking and rechecking the motor connections to the board and the program itself, I finally saw that one of the shorting blocks on J8 had come off. Replacing that fixed the problem.

Wright Hobbies Forums – If you can't find your trouble with these guides, try looking on the forum at: www.wrighthobbies.net/phpBB2/index.php. Click on the Table Top Bot link under the General section, and look for other mentions of your trouble. If you don't see anything, post a message asking the members.

If you still can't find your answer, contact us at: support@wrighthobbies.net or call 630-214-9534.