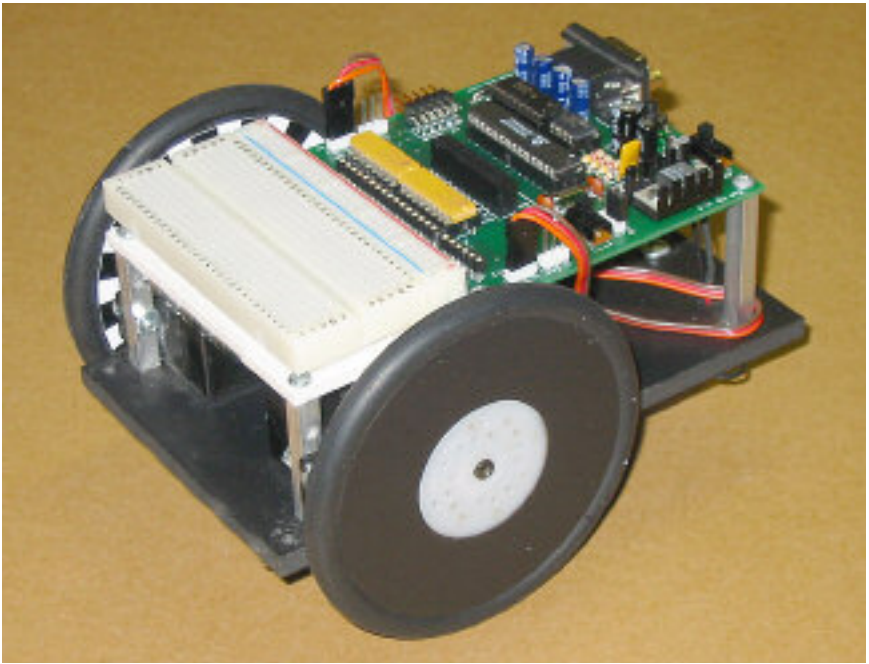


The CBA Robot



Assembly and Programming Manual

Table of contents

Introduction	1
History and Purpose of the CBA Robot	
What to expect from the kit	
Features of the CBA robot	
The Manual	
Tools and Supplies Needed	3
The Main Controller Board	3
Assembly	
Testing and Final Assembly	
Converting the RC Servos to Continuous Rotation	20
The Chassis	26
Assembling the Wheels	
Putting it all together	
The Connectors of CBA	34
How to Loading and Running Programs	37
Some Programming Basics for CBA	37
Building the Battery Replacer	49

History and Purpose of the CBA Robot

The CBA (ChiBot Alpha) robot kit arose from the desire to have a common club robot for The Chicago Area Robotics Group (ChiBots). The kit was designed with a number of criteria in mind:

- Low cost
- Suitable for the beginner
- Great line following ability (many of the ChiBot contests involve following lines)

With these concepts in mind, the CBA kit was born.

What to expect from the kit

A close eye has been kept on cost as well as allowing the builder to actually build something. Most of the available, unassembled robots are nothing more than “screw it together” kits. While there is nothing wrong with this approach, many people are interested in gaining the experience needed to build more complex robots. We have kept this in mind while developing the kit. The wheels are already machined, but need to be cleaned up and glued together. The main circuit board needs to be soldered and tested. The servos must be converted to continuous rotation. All of these tasks are well within the capabilities of any robot enthusiast, and help to develop the skills needed to progress in the field.

We have even left a few things that can be improved upon and have included plans and descriptions for how to do them. Along this line, we encourage others to let us know about the improvements they have made, so we can pass them on to all CBA builders.

Features of the CBA robot

The CBA robot is controlled with a BASIC Stamp 2 microcontroller. The BS2 has a long history in the field of hobby robotics and thus is well documented for this use. The choice of R/C servos for motors was made because the BS2 can easily drive them. The main board of CBA was designed to be expandable, easy to assemble and adaptable for experimentation. It includes a 16-pin header with all of the BS2 I/O pins (protected by 220 ohm resistors), as well as access to regulated 5-volt and full battery voltage. A special 10-pin header is used for connecting line following modules. There is also a 22-pin header that can be used to add additional modules, or even convert the main board to a totally new microcontroller. In addition there are two, 3-pin headers that supply power and I/O for wheel encoders.

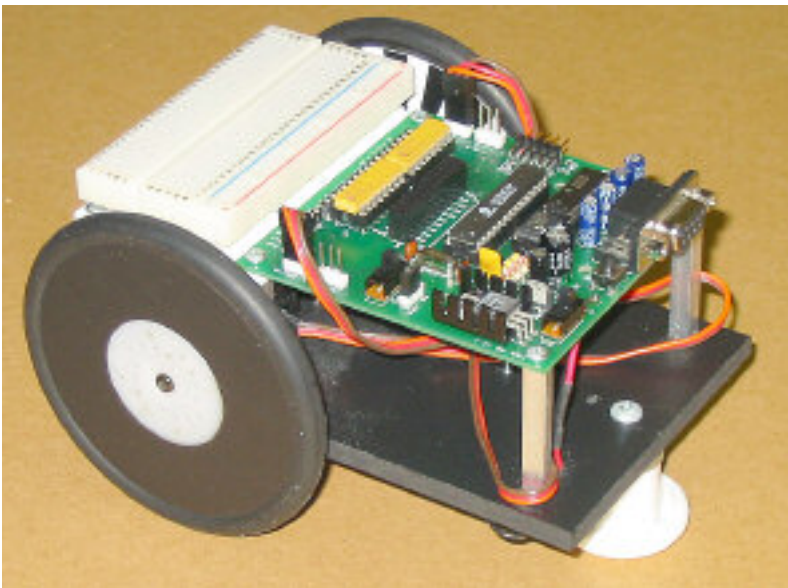
While the base of CBA is simple, it allows for modification, additions or total replacement. The breadboard area at the front of the chassis is a great place for experimentation, yet it can be removed and replaced by a printed circuit board containing a permanent circuit. The wheels and ground clearance of CBA were designed with carpet in mind. The rear “caster” is simple, but very effective in allowing any movement needed on almost any surface.

The Manual

This manual will step you through the construction and testing of CBA, as well as the fundamentals of programming the robot. The steps below are broken down into the basics of what needs to be done, with additional details for the novice builder. You can read as much or as little of the instructions as you need. We would, of course, recommend that you read every word, and we have tried to make them as interesting as possible.

Construction will proceed in a logical manner. We will first assemble and test the main controller board. Then we can convert the RC servos to continuous rotation. We will use the completed main board to adjust the servos as part of the conversion. The wheels will be put together and finally, all the parts will come together on the chassis.

After the robot is built and tested, we will look at the basics of making CBA move. There are also programs available on the website at: <http://www.budgetbot.com> that are tested and ready to run. These programs allow you to see your robot do something right away. We also encourage builders to post their programs to the website to allow others to see what you have done.



Tools and supplies needed

To assemble the kit you will need the tools and supplies listed below. You may already have many of these. If there is a tool, like a soldering iron, that you don't have and don't plan to buy right now, check with your local robotics group. Many groups have events like a Builders Day Out, where members get together to exchange ideas and work on projects. At an event like this, you're sure to find not only someone willing to let you use the equipment you need, but also someone willing to help you to learn the proper technique to use it.

Tools required (listed in order of appearance)

- Soldering iron or soldering pencil. It is best not to use a soldering gun as they aren't made for getting into the tight places of a printed circuit board.
- Solder
- Small needle-nosed pliers
- Small wire clippers. Ones that cut flush to a surface are the best.
- Wire strippers. You can do without these if you don't have them, but it's a good chance to buy a new tool.
- Magnifying glass. Once again not needed but very nice if you have old eyes like I do. On second thought, if you have old eyes like me, it's needed.
- Electrical tape or masking tape
- #0 and #1 phillips screwdrivers
- 5-minute style epoxy glue. You'll only need a small amount, but it's the best thing to use when the directions call for glue.
- Small slot screwdriver
- Small piece of medium to coarse sandpaper
- A hair dryer, heat gun, match, lighter or some other heat source to shrink a piece of heat shrink tube

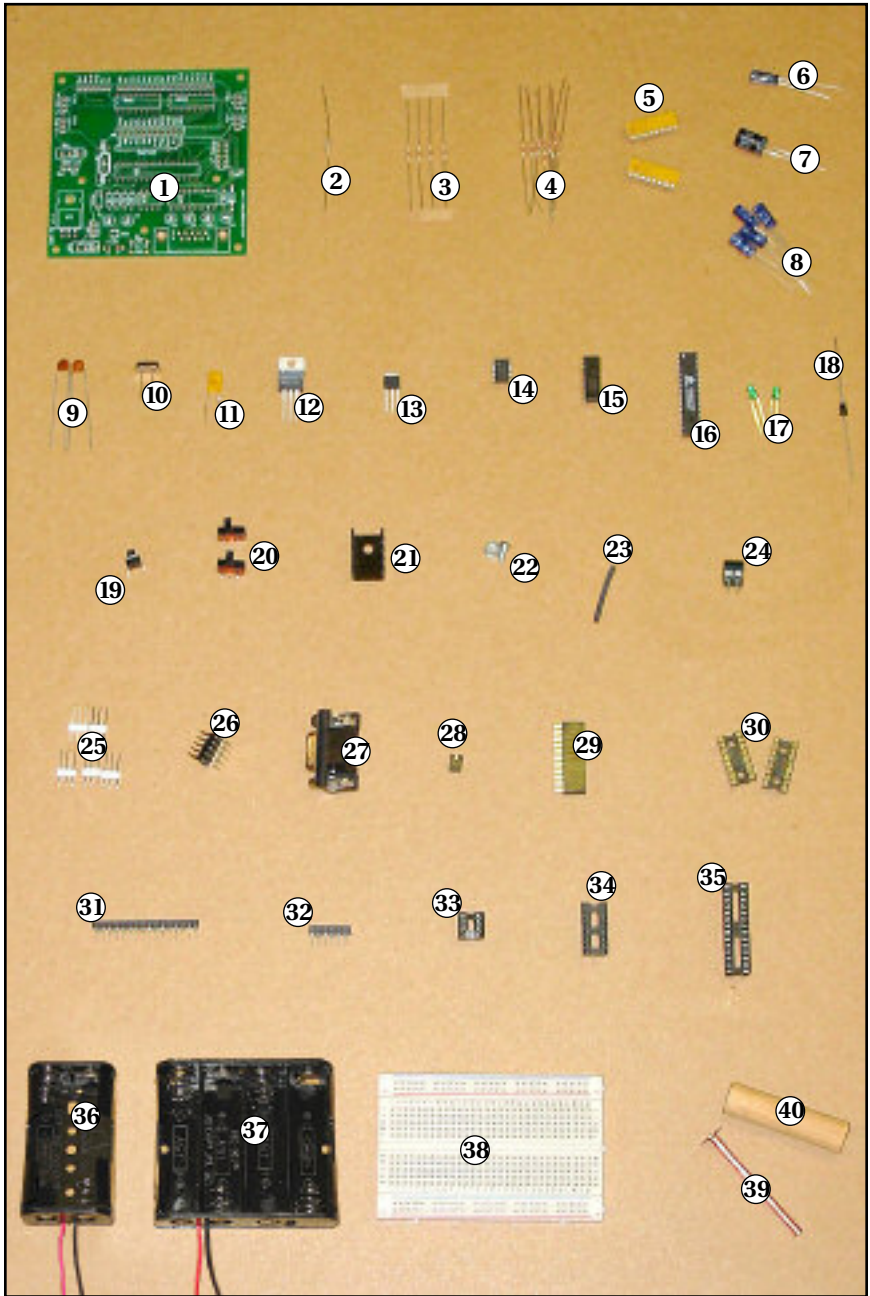
Remember; robot building, like life, can be dangerous. Use caution and work safely while you build.

Assembling the Main Controller Board

Parts List

The parts for the main board are all in one pink plastic bag. There is another clear bag that contains some of the larger, non-static-sensitive parts. Carefully open the bags and empty the contents onto a clean work surface. Some of the parts are rather small, so use care in keeping them together. Using the parts list below check that you have the correct quantity of all the parts listed. If something is missing please contact us via email at: support@budgetbot.com, and we'll get replacement parts to you right away.

	Part	Specification	Quantity
1	PC Board	Main Circuit Board (3.25" x 3.35")	1
2	R1	680 Ohm Resistor (blue, gray, brown)	1
3	R2 - R5	4.7K Ohm Resistor (yellow, violet, red)	4
4	Test & Extras	220 Ohm Resistor (red, red, brown)	9
5	RN1 - RN2	220 Ohm Resistor networks	2
6	C1	10uF 25V Elec Cap	1
7	C2	220uF 25V Elec Cap	1
8	C3 - C6	1uF 25V Elec Cap	4
9	C7 - C8	22pF Disk Cap	2
10	Q1	20 mHz Crystal, 200ECSH	1
11	PPTC	PPTC Over Current Protector, RUE110	1
12	IC1	LM2940CT Voltage Regulator	1
13	IC2	IRFU5505	1
14	IC3	24LC128 EEPROM	1
15	IC4	MAX232	1
16	IC5	BS2e OEM 28pin DIP	1
17	LED1 & Tester	Green LED T1	2
18	D1	1N4004 Diode	1
19	SW1	Momentary Contact Switch NO	1
20	SW2 - SW3	SPDT Slide Switch PC Mount	2
21	Heat Sink	Heat Sink	1
22	Screw and Nut	4-40 x 1/4" Machine Screw and Nut	1
23	Shrink Tubing	1" Long	1
24	Terminal Post	2 Position Screw Terminal	1
Headers and Sockets			
25	3 Pin Male	3 Pin Male 0.1" Header	5
26	2x5 Pin Male	2x5 Pin Male 0.1" Header Right Angle	1
27	DB9	DB9 Female Socket for Programming	1
28	Jumper	0.1" Jumper for Servo Power	1
29	2x11 Pin Socket	2 x 11 pin Female Socket	1
30	Resister Sockets	16 Pin Machined DIP Socket	2
31	IO Header	16 Pin Machined Female Header	1
32	Power Header	6 Pin Machined Female Header	1
33	DIP 8 Socket	8 Pin DIP Socket	1
34	DIP 16 Socket	16 Pin DIP Socket	1
35	DIP 28 Socket	28 Pin DIP Socket	1
36	Battery Holder	2 x AA Battery Holder Flat	1
37	Battery Holder	4 x AA Battery Holder Flat	1
38	Bread Board	30 Pin Wide Bread Board	1
39	Replacer Core	2" Copper Nail	1
40	Replacer Body	2" long 1/2" Diameter Wooden Dowel	1



Assembly

We will assemble the main board starting with some of the least sensitive parts to solder. If you have soldered PC boards before, go ahead and dive right in. If not, we recommend doing a little reading first. There are some great soldering tutorials online. Here are some you can check out:

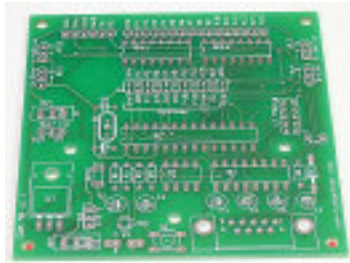
www.aaroncake.net/electronics/solder.htm

www.epemag.com/solderfaq/default.htm

Or search for “how to solder” with your favorite web search engine.

Board orientation

When we speak of locations on the board, we will always reference them looking at the board as shown below with the 9-pin serial connector closest to you (corresponding to the back end of CBA). The side with the white printing is called the component side. You will place all components on the component side. All the soldering will be done on the bottom, or solder side of the board.

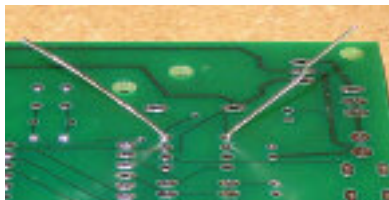


The formatting of the instructions

Finally a note on the formatting of these instructions. Most steps will be broken into two parts. The first part in **bold** type is what is to be done in that step. The second area in *italic* type contains some details about the step along with some helpful hints to make the step easier. If this is the twentieth robot and/or PC board you've built, feel free to skip this part and fly with your experience.

Step 1: Solder in resistors R1 through R5. R1 is 680 ohm (blue, gray, brown), R2, R3, R4, R5 are all 4.7K (yellow, violet, red).

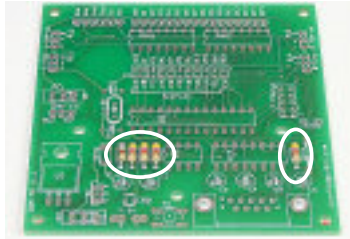
To solder in the resistors, bend the leads of the resistors 90 degrees at a point close to the body of the resistor. Bend one resistor and check to see how it fits into board at the place marked for that resistor. If it fits fine, go ahead and bend all the resistors the same way. Slip the resistors through the marked places on the board and hold them in place by bending the leads. The pictures below show how this is done.



It doesn't matter to the resistor which way it is placed into the board. On the other hand, it's nice to have a circuit board that's neat and tidy, so try to place all the resistors so that the gold bands are facing the same edge of the board.

Turn the board over and solder all the leads in place. Check that all your solder joints look good (shiny), and that there are no solder bridges between adjacent pads. If everything is fine, trim the excess leads off at the top of the solder joints.

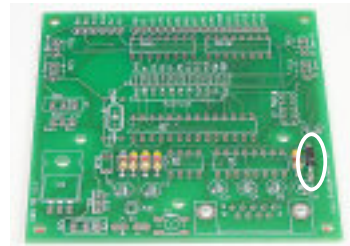
Sit back and smile. You've taken the first step to building your own robot.



Step 2: Solder in diode D1. Be sure to follow the polarity of the diode and the markings on the circuit board.

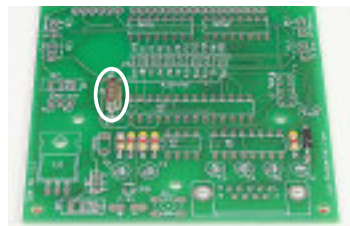
Bend the leads of the diode like you did with the resistors but not as close to the body of the diode as you did with the resistors. The holes for the diode are farther apart on the board than for the resistors to keep the body of the diode away from the heat of soldering and to keep from breaking the diode body when they are bent.

Put the leads through the boards while being sure to watch the direction they are inserted. The diode will have a white band on one end. Match this band with the white band marked on the board. Bend the leads to hold in place, solder, check and trim the leads just like you did with the resistors.



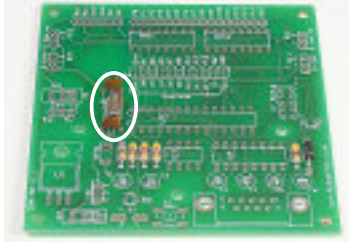
Step 3: Solder in crystal Q1.

There are three holes in the PC board for the crystal; use only the outer two. The crystal can go in either way. Be sure to use just enough heat to melt the solder, and allow 10 seconds to elapse before soldering the second pin to avoid overheating the crystal.



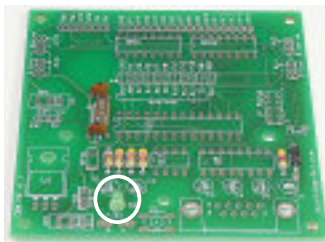
Step 4: Solder in capacitors C7 and C8.

These are the small disk-shaped capacitors. They can go in either way. Bend the leads to hold them in place, then solder and trim just like the other components.



Step 5: Solder LED1 in place. Watch the polarity of the LED. The cathode is marked with a flat side and has a shorter lead. The Cathode goes towards U2 (the left side of the board).

LEDs really do care which way they go in. If they're in wrong, they don't work and can be damaged. There is a small flat side (use your magnifying glass, if needed) on the base of the LED. It needs to go towards the left side of the board. The LED leads can go all the way in so that the LED sits flush with the board. Bend to hold in place, solder, check and trim the leads just like before.



Step 6: Solder in U2, the IRFU5505. The flat silver side faces the left.

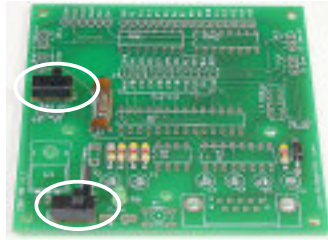
There are bumps on the leads that will keep this component from sitting tight to the board. This is okay and will keep the component about 1/8" above the board surface.



Step 7: Solder in the on-off power (slide) switch, SW2. Solder in the servo power (slide) switch, SW3.

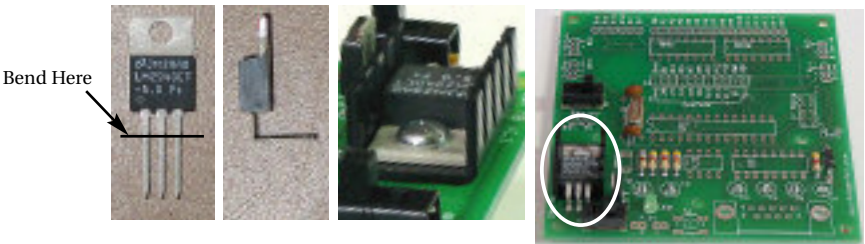
These switches can go in either way. It is handy to use a small piece of electrical or masking tape to hold the switch in place; see the picture below. Once it is taped in place, solder just one of the leads. Then remove the tape and check to

see if it is in straight. If it isn't, you can reheat the solder joint while holding the switch to straighten it. Once you are satisfied with its position you can solder the other two leads. You do not have to trim the leads of the switch.



Step 8: Put in U1, the LM2940CT voltage regulator. The regulator sits on a heat sink and both are held to the board with a screw and a nut. Bend the leads of the regulator 90 degrees away from the front labeled side of the regulator, right at the point where the leads narrow. See the pictures below. Then set the regulator into the heatsink and use a 4-40 x 1/4" long screw and matching nut to hold the regulator in place. The screw goes in from the top, through the regulator, the heat sink and then the board. The nut is on the solder side of the board. Tighten the nut and screw in place, then solder the regulator in place and trim the leads.

There's nothing left for the beginner italic parts of the instructions on this one. That should make the italic readers feel better; even the pros needed the information this time.



Step 9: Solder in the PPTC (Polymer Positive Temperature Coefficient, over-current protection device).

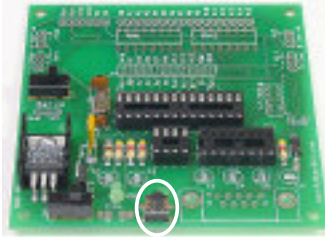
The PPTC can go in either direction. The component has bends in its leads that help keep it from sitting tight to the PC board. Keeping it off the board allows for better air flow around the component.



Step 10: Solder in the Reset Push Button, SW1.

This switch can be correctly placed in only two of the four possible orientations. The leads are farther apart one way than the other, so be sure to install it with the leads lining up nicely with the holes. The other way it can go in is 180 degrees from this and it will work just fine, too. When you push the switch into the holes in the board it will snap into place and hold itself firmly while you solder its leads.

OK, to tell the truth you could bend the leads all kind of funny and crooked and get it in wrong, but why would you be doing that anyway?



Step 11: Put in the DIP sockets for U3, U4, and U5. **Do not put the chips in the sockets yet.**

*While the sockets themselves don't care which way they go in to the board, and they will fit either way, it is important to put them in correctly. Each socket will have a little notch cut in one of the short ends. This notch needs to line up with the notch marked on the board. This way, when you put the chips in the sockets, you can tell which way they should go in, and the chips **really** do care. All the notches face the same way, to the left side of the board.*

Use the tape trick here again to hold the sockets in place while you turn the board over. If you don't do this, you'll be stuck trying to solder standing on your head so they don't fall out, which is not a cool idea for making good solder joints. When you solder, start with just the two leads at opposite corners of the socket. This will hold it in place. Then take off the tape and check to see that the socket is sitting flat on the board. If it isn't, hold the socket tight to the board and reheat the joint at the raised corner of the socket. The socket should drop snug to the board. Once you are happy with the way the sockets fit, solder all of the pins in place. Take your time here; there are a lot of pins. Be very careful to avoid solder bridges between the pins. Use only a fine-tipped iron and remember to clean your iron tip every once in a while. This instruction applies to all of your soldering.

Do not put the chips in their sockets yet. I know you want to, it would look so cool, but don't. To avoid the possibility of destroying the chips, we will do that only after some serious testing of the finished board.



Step 12: Solder in the 2 machined pin 16 pin DIP sockets labeled as RN1 and RN2.

Line up the notches just like you did in step 9. Again use the tape, solder two leads and straighten steps like you did before. You will also leave these sockets empty for now.



Step 13: Solder in the 6-pin and 16-pin single row machined pin sockets. They are at the very top of the boards labeled “+5, gnd, bat and P0-P15”.

These sockets will also need to be taped in place and one or two leads soldered. These sockets have a hard time sitting straight when you solder them, so you will likely need to reheat and straighten. Do not try and bend them straight when cold. You stand a very good chance of pulling the copper right off of the PC board. When you reheat them, be very sure your finger is not pushing from the top on the pin you are reheating. The heat will flow right though the pin very quickly, and you will get burned. Anyone care to guess how I know this?



Step 14: Solder in the 2 row by 11-pin female header. It goes in the center of the board.

You'll need the tape here, too. Solder in the two far corner pins, remove the tape and check the fit. Reheat and adjust if needed, then solder all the pins.



Step 15: Solder in electrolytic capacitor C1. Watch polarity!

Electrolytic capacitors are polarized and thus care which way they are put in. The capacitors have a stripe up and down one side next to the negative lead, and are so-marked (-). The other unmarked lead is the positive one. The board, on the other hand, has a mark on it to note where the positive lead goes (+). I know it's odd that the part is marked for the negative and the board is marked for the positive, but that's just the way it's done. So, when you put the capacitor in, be sure it goes in the correct way. That is, with the lead that is not marked (the positive lead) going into the hole marked "+" on the board.

Step 16: Solder in electrolytic capacitor C2. Again watch polarity.

The same rules apply for putting in C2 as you did for C1. Please notice though that the two capacitors face different directions on the board. The negative stripes face one another.

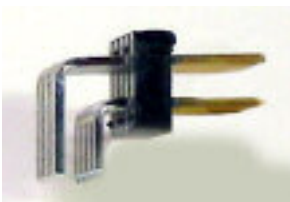
Step 17: Solder in electrolytic capacitors C3, C4, C5, C6. Watch polarity!

All four of these are the same type of part. Watch polarity like the other two you have done. Note that they are not all facing the same way. Match the markings on the board, and all will be fine.



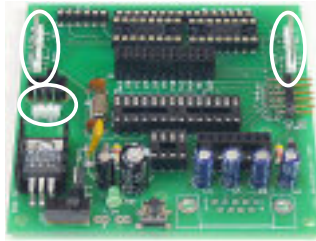
Step 18: Solder in the 2 row by 5-pin right-angle male header. It goes on the right side of the board in the spot labeled "Line Mod". The header needs to be oriented as shown below left, with the pins facing to the right.

You'll need tape for this one, too. You should be getting pretty good at this by now, and that's a good thing because these are harder to straighten without getting hot fingers. I would only solder one of the pins on one end to hold it in place. That way, if you need to reheat it, you will have the other end to put your finger on and hold.



Step 19: Solder in the five 3-pin male headers. Two for the servos, two for the wheel encoders, and one for the servo power jumper.

OK, these are about as small as the headers get. If you're really good at the tape trick, you should get these straight right away. If not, reheat like you did with the others but really watch what you hold on to as you do it.



Step 20: Solder in the 2 position screw terminal. Face the silver openings on the one side of the terminal towards the edge of the board.



Step 21: Solder the DB9 female connector in place. Also, solder the two lugs that snap into the big holes.

The connector will snap into the holes and be seated tight to the board when in all the way. It will hold itself there while you solder the pins. Also, solder the two lugs to the pads around them. If you have a small, lower-wattage soldering pencil, you may have a hard time getting these hot enough to melt the solder. Take your time and be sure the largest surface of the iron is in good contact with the lugs. You may need to put just a bit of solder on the iron where it touches the lugs. The solder will help conduct the heat from the iron to the lugs. You don't need to fill the holes with solder. Just use enough to stick the lugs to the pads.



Step 22: Insert the two resistor networks into the 16-pin machine pin sockets. These are the two black 16-pin “chips” with matching code number on top.

To get the chips to fit the sockets, you may need to bend the pins closer together. To do this, hold the chip by the end and gently press the flat side of the pins to the work table to press the pins closer to the body of the chip. Although it doesn't matter to these chips, it is best to place them in the correct orientation. Each chip will have a small notch in one end. This notch should align with the notches in the DIP sockets.



Step 23: Put the jumper on the servo power jumper.

The servo power jumper has three pins. The center pin is common. The voltage the servos get depend on which two pins the jumper is placed on. To start place the jumper so that it is on the left two pins of the servo power header. The board is marked with “bat” below where the jumper should go.

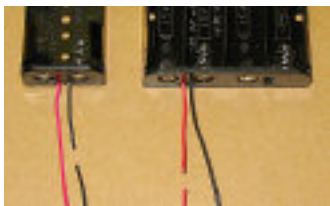


That's it. The main board is done and just needs testing. “I don't have the three chips in place,” you say. That's correct. They're not supposed to be there yet. Remember we need to test a few things before we put them in. So let's get on to the testing.

Testing and Final Assembly

Step 1: Take a few minutes to take a close look at the bottom of the board. Carefully inspect each solder joint. Check for dry or cold (dull, instead of shiny) joints. All the solder joints should look clean, rounded and bright. Check to see that you didn't miss any.

Step 2: Prepare the two battery holders by connecting them in series. To do this, cut the black lead of the 2-cell holder to 1" long. Cut the red lead of the 4-cell holder to 1-1/2" long. (Be sure to save the cutoff wire, you will use it to make a tester in step 11.) Strip 1/4" of insulation off the two newly cut leads. Slip the 1" piece of shrink tube over the red lead of the 4-cell holder. Be sure the two holders are positioned as shown below, the 2-cell holder on the left and the 4-cell holder on the right. Twist the two stripped leads together and solder. Once the solder cools, slip the shrink tube over the soldered joint and shrink into place with a heat gun or match.



Step 3: Attach the battery holder leads to the main board. These are connected to the 2-position terminal between the switches. The black lead goes to the terminal marked (-), it is the one closest to the push button (SW1). The red lead goes to the terminal marked (+), it is the one closest to the slide switch (SW2). The stripped end of the battery leads go into the side of the terminal and are connected by tightening the top screws.

Step 4: Be sure the main board is turned off. The Power Switch (SW2) should be slid toward the right (center of the board). The servo power switch (SW3) can be either off or on.

Step 5: Put batteries in the holder. If you are using nickel metal hydride (NiMH) or NiCad batteries, you will need six of them. If you are using alkaline batteries, then you should only use five. The sixth spot will be filled with the battery replacer. You will need to assemble the battery replacer if it's needed. See page 49 of this manual for assembly instructions.

Step 6: Now is a good time to clear off your work bench. Be very sure nothing is under the main board as it sits on the non-conductive work surface. We will power up the board for some basic testing, and we don't want it to short out on anything.

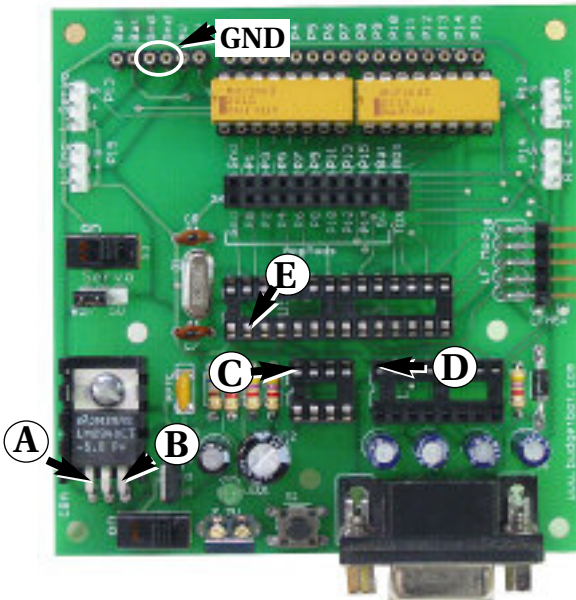
Step 7: Move the power switch to the on position. The green LED should light. Turn the power switch to off. If the light came on, congratulations! You've passed test number one. You can go on to step 8. If not...

If the light didn't come on, it's time to check a few things:

1. Check that the battery leads are connected correctly. Red wire to (+) left, black wire to (-)right.
2. Check that the LED is in the correct orientation. The flat surface should be to the left.
3. Check that the voltage regulator (IC1) is in correctly. The larger, flat surface should be down, toward the PC Board.
4. Check that diode IC2 is in correctly. The flat siver side should be facing left toward the voltage regulator.
5. Check that the two capacitors, C1 and C2 are in correctly. The negative stripe on both C1 and C2 should be to the right.
6. Check the solder joints on the whole board. Look for solder bridges (places where solder connects two solder pads together that shouldn't be). Look for pads that were never soldered.

It is likely that one of the above needs correcting. After you fix the problem, try the switch again. It is possible that more than one thing is wrong. So if it doesn't work the second time, don't give up – take a break, come back to it later and check everything again.

Step 8: If you have a voltmeter, you can check the voltage at a few key points. Turn the power switch back on. Set your voltmeter to DC Volts and to a range that will read up to 10 volts. Place the tip of your negative probe on one of the two center pins (GND) of the 6-pin header, labeled "GND" in illustration below.



The next part is very important. To avoid damage you must be very careful when making the following measurements to touch the tip of your positive probe ONLY to the points described. Work slowly so as to not accidentally create a short circuit between the point being measured and an adjacent pin. With a little care, everything will go just fine. Test each of the following points with the red probe:

- a. The left pin of the voltage regulator should read the battery voltage of about 7 to 8 volts. Labeled "A" below.
- b. The right pin of the voltage regulator should read between 4.9 and 5.1 volts. Labeled "B" below.
- c. Pin 8 of the 8 pin DIP socket should read the same as you read in test number 2. This is the top left pin, labeled "C" below.
- d. Pin 16 of the 16 pin DIP socket should read the same as you read in test number 2. This is the top left pin, labeled "D" below.
- e. Pin 2 of the 24 pin DIP socket should read the same as you read in test number 2. This is one pin to the left of the lower left most pin, labeled "E" below.

If all of the tests are fine, you're ready to go to the next step. If not, check all the items listed in step 7 again. If there is still a problem, DO NOT go on to Step 9. The board will NOT work and the ICs might be damaged. Instead, have someone else check over your work. From personal experience I can assure you that it IS possible to look at an error repeatedly and just not recognize it. And, you are not imposing when you ask for help, as the other person will derive a great deal of satisfaction from helping you find and resolve the problem.

Step 9: Be sure the power is off, then put the three ICs into their DIP sockets. While modern chips are much more resistant to static discharge than in the past, it is still wise to take some precautions. If you have a grounding wrist strap, follow the directions that came with it and use it while handling the chips. If not, at least reach over and touch a solid ground e.g., the back of your computer case, an exposed bit of conduit, etc., before handling the chips. As always, we assume that you are working in a safe area, with no possible presence of exposed power line voltages.

To get the chips to fit the sockets, you may need to bend the pins closer together. To do this, hold the chip by the end and gently press the flat side of the pins to the work table to press the pins closer to the body of the chip.

When you place the chips in the sockets be sure to place them in the correct orientation. Each chip will have a small notch in one end. This notch will align with the notches in the DIP sockets. On this board, they all happen to be on the left side.

When you have finished, take a moment to carefully inspect the installed chips. In particular, check that the notch is to your left when facing the DB9 connector, that no pins are bent under the IC, and that every pin is inserted into its respective socket.

Step 10: Next, we will load a BASIC Stamp program into the board. To do this you will need to have the BASIC Stamp Editor loaded onto your computer. The software can be found on the Parallax CD included with your kit.

To install the BASIC Stamp Editor software, insert the Parallax CD in your computer's CD-ROM drive. When the window for the CD opens select "Software". Then double click on BASIC Stamps, and again on BASIC Stamp Editor (Windows). Then select the disk icon next to Stamp Editor v2.1 and click on install. Version 2.1 of the BASIC Stamp Editor will run in Windows 95, 98, ME, NT4, 2000 and XP.

You will also need the CBA robot test files. They can be found in the CBA folder on the BudgetBot.com CD included with your kit. They can also be downloaded from the Web site at: <http://www.budgetbot.com/downloads>

- a. Place the "Testing" directory of programs in an accessible location. We recommend that you put them into the Parallax folder in the Program Files folder on your hard drive.
- b. Connect your computer to the main board with a nine pin serial cable. This needs to be a normal "straight through" cable, not a null modem cable.
- c. Open the BASIC Stamp Editor, it will be in your Start menu.
- d. Load the program from the samples files called "Test All Pins.bs2".
- e. Turn the power switch on the main board to on.
- f. If you smell or see smoke, turn the power switch off. This is very unlikely, but you never know. If something like this happens, check all the points in the trouble shooting section below.
- g. In the BASIC Stamp Editor, select Run from the Run menu. A window will open showing the progress of the download to the controller board. It should only take a few seconds to download the program. If the program loads correctly, you are very close to being done with the tests. If you get a "Stamp not found" message, check the trouble shooting section below.
- h. Turn the main board power switch off.

Step 11: Build the LED Test Probe. We will use an LED, 220 ohm resistor and the piece of black wire you saved. Follow the instructions below to make your probe.

- a. Trim the short lead of the LED so that it is about 1/2" long. This is the cathode connection and corresponds to the small flat on the LED's base.
- b. Trim The leads of a 220 ohm (red, red, brown) resistor so that each lead is about 3/8" long.
- c. Strip about 1/4" of insulation off both ends of the black wire.
- d. Solder one end of the resistor to the short lead of the LED. You may want to tape the components to your work surface to hold them in place as you solder them.
- e. Solder one end of the wire to the free end of the resistor.

- f. Tin the free end of the wire by placing some solder on the loose end of the wire. This will keep the strands of wire together.
- g. Plug the long lead (not the one we added the wire to) of the LED into one of the 5V connection points on the main board.
- h. Bend the wire up so that you have a test probe like the one in the picture below.



Step 12: Turn the main board power switch back on. This will run the “Test All Pins.bs2” program that you loaded earlier. Use the LED test probe to check each of the BASIC Stamps 16 pins. Touch the wire end of the probe to each of the connection points on the P0 through P15 header. Leave the wire at each point being tested for a few seconds. The LED should flash on and off about once a second. If all the pins check out, you’ve done it. You have a fully functional BASIC Stamp 2 CBA controller board.

The “Test All Pins.bs2” program makes each pin an output , and writes a logic “zero” (about 0 volts) to it. Current flows from +5V, through the LED and a 220 ohm resistor to 0V at the pin, lighting the LED. CAUTION: under no circumstances touch anything other than pins on the P0 – P15 header, damage to the LED and other components may occur.

Step 13: Take a rest, show your family the flashing lights and smile.

Trouble Shooting

- Check the solder joints again. This is the most common problem with a PC board. Look for bad or missing joints. Look for solder bridges. Look for leads that are too long. Also, look all over the PC board for solder bridging pads at places other than along the rows of sockets and header pins.
- Check that you have the correct components in the correct locations and with the correct orientations.
- Check that your programming cable is the correct type.

Converting the RC Servos to Continuous Rotation

Overview

A standard RC servo is made so that it moves its control shaft to a particular location in response to a specific input signal. For most servos this means the shaft rotates about 180 degrees. Inside the servo there is a feedback mechanism that tells the servo electronics the position of the output shaft, as well as stops that keep the shaft from rotating more than a set number of degrees, usually about 180.

For a standard RC servo to work as a robot drive motor, the stop and feedback electronics must be converted so that the output shaft can rotate continuously. Before you panic, take a deep breath, relax and read though the instructions. We'll take you step by step through the process of converting a GWS S03N 2BB servo, with lots of pictures. It's not too difficult and you'll be that much closer to understanding one of the basic rules of robot building; find something that is close to what you need and change it to make it do what you want.

There are many ways to convert or "hack" a servo. We have picked one of the simpler ways, but it will give excellent results. If you have hacked servos before and want to use your own method, please do.

We'll follow the same rules for these instructions as we did for the main board assembly. There will be **bold** copy that tells what the step is, if you're an expert it may be all the information you need. The bold sections will be followed with a section of *italic* text. This section will give you the details and some tips to make the steps easier.

Tools needed

- Small #0 or #1 phillips screwdriver
- Small pair of sharp flush-cutting wire cutters or a very fine Hobby saw blade
- Hobby knife
- A small amount of 5-minute epoxy
- A small holder for the parts you remove. A glass or cup works, just don't drink them.
- Cotton swab or paper towel
- Alcohol or cleaner to remove grease before gluing

Step 1: Clean off your work surface.

There are small parts inside the servo that can fall out when you take the cover off. A clean work area makes it easier to find the escapees. Also the gears in the servo are coated with grease that can easily pick up loose dirt.

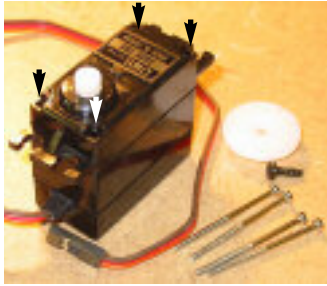
Step 2: Remove the servo horn from the output shaft.

The horn will have a small phillips head screw holding it in place. Remove the screw and save it, you'll need it to mount the wheels later. Gently pull the horn off the shaft. A little wiggling will help.



Step 3: Remove the four screws from the top of the servo case.

As you take the screws out be sure to keep the servo case together. You can hold the servo from the top and bottom as you take the last screw out to keep it together. Keep the screws in the small container.



Step 4: Remove the top of the case. Use care to keep the parts together in the case.

The best way to remove the top is to set the servo on your bench, then with the finger of one hand push down on the output shaft. With the other hand lift the top of the servo case top up. A little wiggling of the case top will help it slide up. When the case top is even with the top of the output shaft and it can't go any more because your finger is in the way, remove your finger and lift the top off. It should come off easily at this point. If you've done this correctly, all the gears and parts should still be in the main servo case.

If for some reason the gears have fallen out, it's okay. Your bench is clean, right? They should be right there. We'll put them back later.



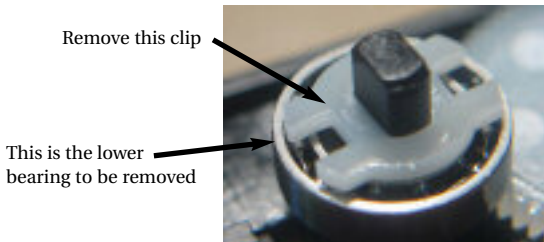
Step 5: Remove the output shaft and the gear that is next to it.

Take the two out as a unit. They should slip right off of their shafts. Put them both away in that safe place.



Step 6: Remove the white output shaft-to-potentiometer link clip.

Under the output shaft you will find a small white clip sitting on a black rectangular shaft. Remove it by pulling it up; you may need to use a small screwdriver to get it out. The small black shaft it fits to is the potentiometer (pot) the servo uses for feedback. Removing the clip disconnects the pot from the output shaft.

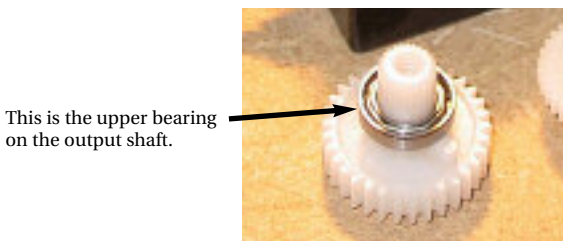


Step 7: Remove the lower bearing from around the pot.

The lower bearing is press fit onto the area around the pot shaft. Remove it from this area by lifting it with your fingernails. If needed you can use a small screwdriver to help pry it up; just be careful to pry it evenly, and not all at once from one side. Put it safely aside.

Step 8: Remove or at least move, the upper bearing from the output shaft.

Find the output shaft in its safe place and remove the bearing from the shaft. It is also press fit onto the shaft, but it is tighter than the lower bearing. If you get it part way and then it stops, that's okay, it doesn't need to come all the way off. You just need to move it about 1/8" for the next step.



Step 9: Cut the rotation stop from the output shaft.

This is really the only step that can get you into serious trouble. Look at the output shaft closely. Observe the little protrusion that sticks up from one part of the gear and runs over to the shaft. We need to remove this protrusion smoothly and close to the gear. There are a number of ways to do this. One of the fastest is to use a small sharp pair of flush cutting wire cutters. Don't try and take it all in one cut, make two cuts. Holding the flush cutting part of the clippers toward the gear, clip off part of the protrusion. Take about half off with the first cut, and the rest with the second. It is also possible to use a very fine hobby saw and cut off the protrusion.

Look at your work closely. The cut needs to be flush with the gear. If it's still not flush clean it up with a sharp hobby knife. Use care not to harm the gear teeth or your fingers.

Double check at this point that there is nothing stuck in the gear teeth. The teeth are coated with grease and this causes them to pick up little odds and ends off your work area. You did clean it first, didn't you?



Step 10: Connect the servo being converted to the CBA main controller board. Then load and run the “servo center setup.bs2” program.

The servo plugs into the main CBA board at the connector noted. You can plug the servo into either the right or the left connector. The plug will go in either way but only one is correct. The board is marked with (-, +, and S). The servo has wires colored brown, red and orange. Plug the connector into the board so that the brown wire goes to the (-), the orange wire to the (S), the red wire will go the middle or (+).

Set the servo flat on your work bench. Be sure nothing is close to it and the wire for the servo is not in the gears. Also see that the two gears left on the servo are in their original locations.

On your computer open the “servo center setup.bs2” program in the BASIC Stamp Editor. This program will be located in the “Testing” folder you copied from the BudgetBot.com CD when you tested the main controller board. You could also use the BASIC Stamp program that is on the next page.

```
{ $STAMP BS2e}
{ $PBASIC 2.5}

DO
  PULSOUT 12, 750      'Right Servo pin P12
  PULSOUT 13, 750      'Left Servo pin P13
  PAUSE 20
LOOP
```

These programs send 1.5 ms pulses to both Stamp pins P12 and P13. This pulse width is considered “center” for most servos.

Plug the serial cable from your computer into the main controller. Turn the main controller and servo power switches on. Then load (run) the program. The servo should start to rotate. While the servo runs, carefully rotate the black pot shaft back and forth. As you turn it you should find a place where the motor stops and then turns the other way. There will be a very small spot where the servo is not turning at all. Set the pot to this position and turn off the main controller power.

Step 11: Fix the pot shaft in position with epoxy.

Check that there is no grease on the area where the pot shaft contacts the area around it. Clean it off with a cotton swab damp with alcohol. Mix up a very small batch of 5-minute epoxy. With a toothpick, put a small drop of epoxy at the joint between the pot shaft and the body of the servo. Take care to not get it in the area where the bearing sits. Turn the power back on. You may need to move the pot shaft slightly to get the motor to stop again. Once you have it stopped, turn the power off again and let the glue harden. As the glue hardens, you can turn the power on from time to time to be sure the shaft is still in position.

If at the end the motor still turns slowly don't worry, it can be compensated for in software later.



Step 12: Reassemble all the parts, but leave the small white connecting clip out.

- a. *Put the lower bearing back on the body around the pot. Press it back on and see that it fits all the way down on the body.*
- b. *Put the upper bearing back onto the output shaft. If you only slid it up the shaft part way, push it back down to its normal place.*

- c. Replace the output shaft and the gear next to it that you removed in step 5. They will need to go back in as a unit. To do this hold the two gears together and slide them down onto their shafts. You may need to wiggle them some to get the gears to all mesh correctly.
- d. Replace the top cover.
- e. Replace the four screws in the top cover. They should be tight but not too tight, just snug them up to pull the top cover into place.
- f. When everything is back together you should only have two things left over – the little white connector clip that went between the output shaft and pot, and the little scraps you clipped off the output shaft.



Step 13: Test the servo again but this time use the “servo test.bs2” program. The servo should run in one direction, slow down, stop and then turn in the other direction. This will repeat until you turn off the power. The servo should turn smoothly. If it jumps or has a click or knock that happens time after time as it rotates, you’ll need to check it out. The most likely cause is the protrusion on the output shaft was not cut closely enough or smoothly enough. It can also be caused by bits of dirt in the gears. Take the top off the servo again and watch it run with the output shaft in place. Look for things that catch on or are stuck in the gear teeth.

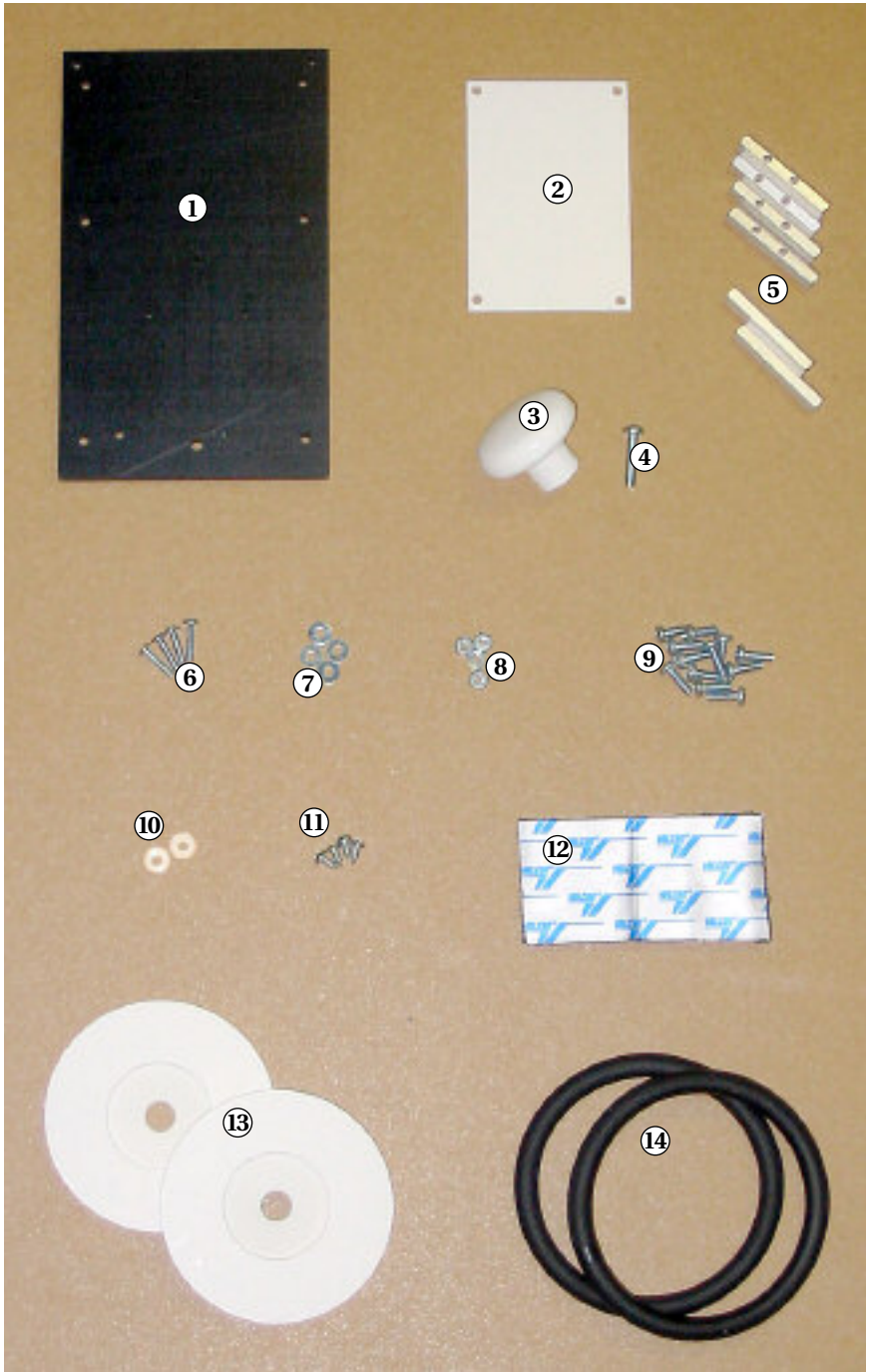
Congratulations! You have just converted your first servo. Follow the steps again to do the second servo. This one will go much faster, you’re almost an expert now.

Assembling the Chassis

Chassis Parts List

All the parts for the chassis are in one clear plastic bag. Open the bag and check the parts to the parts list that follows. If something is missing please contact us via email at: support@budgetbot.com, and we'll get replacement parts to you right away.

	Part	Specification	Quantity
1	Main Base	6mm Black Sintra	1
2	Breadboard Base	3mm White Sintra	1
3	Caster	White Plastic Drawer Pull	1
4	Caster Screw	8-32 x 3/4" Pan Head Screw	1
5	Standoffs	1/4 Hex Aluminum Standoffs	6
6	Servo Screws	4-40 x 1/2" Panhead screws	4
7	Servo Washers	#4 Washers	4
8	Servo Nuts	4-40 Nuts	4
9	Standoff Screws	6-32 x 7/16" Pan Head Screws	12
10	Spacers	1/8" thick Nylon Spacers	2
11	Battery Screws	4-40 x 1/4" Sheet Metal Screws	4
12	Velcro	6" of 3/4" wide or 3" of 1-1/2" wide	1
13	Wheel Blanks	6mm White Sintra	2
14	Tires	5/16 Neopreme o-rings	2
	Servos	GWS S03N 2BB Drive Servos	2

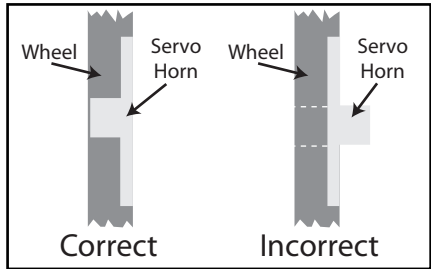
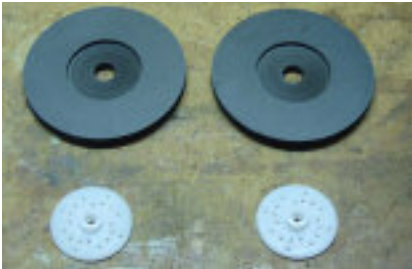


Assembling the Wheels

Step 1: Find the two larger round servo horns that came in the servo packages. We said to save them so I take it you did. Your wheels will be white, we have used black in the pictures so you can tell the wheels from the servo horns.

Step 2: Test fit the horns into the recess in the side of the wheels. They go with the shaft of the horn through the hole in the wheel. When in all the way, the face of the servo horn should fit flush with the face of the wheel.

If needed use your hobby knife or some sandpaper to take off any bumps or ridges on the horn to make them fit. Pay attention to the diagram below for the correct orientation of the horns in the wheels.



Step 3: Sand the back surface of the horn to roughen it up.

The back of the horn will be glued to the wheel with epoxy. The epoxy will not stick well to the slick plastic of the horn. You're just looking to scuff the gloss off of the back surface (that's the side that will face the wheel).

Step 4: Recheck the fit and clean the surfaces to be glued.

If the servo horn and wheel recess are not clean, use some paper toweling damp with alcohol to clean them.

Step 5: Mix up enough 5-minute epoxy to glue the two horns in place on the wheels.

You'll need an amount of epoxy about the size of a quarter and about 1/8" deep.

Step 6: Spread epoxy onto the recess area of the wheel. Do one first to see how much epoxy you need. The epoxy should cover the large flat area of the recess and maybe up the sides a little. When it is covered, place the horn into its place in the wheel.



Step 7: Place a piece of plastic wrap or the bag the parts came in on your work bench. Place the glued wheel/horn assembly onto the plastic with the horn side up. Place another piece of plastic over the top of the wheel and place a smooth flat weight on top of that. If you had enough glue to do both wheels at the same time you can stack one wheel on top of the other, just be sure to have plastic between them. Allow the glue to set for the recommended time.

Step 8: Place the o-ring tire onto the outside of the wheels.

The simplest way to do this is to start the tire onto about half the wheel. Then with a rolling action roll the tire up and into the groove on the edge of the tire. Once it is in place you can lift it a little to take the twist out of the tire that you put in when you rolled it on.

Your wheels are ready to roll.

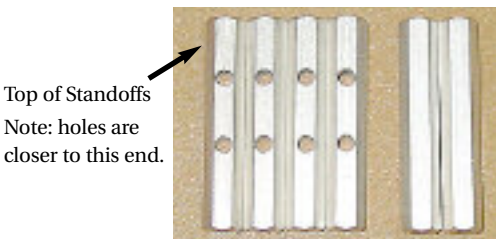


Putting it all together

We will now put all the parts of the chassis together with the main controller board, servos and wheels.

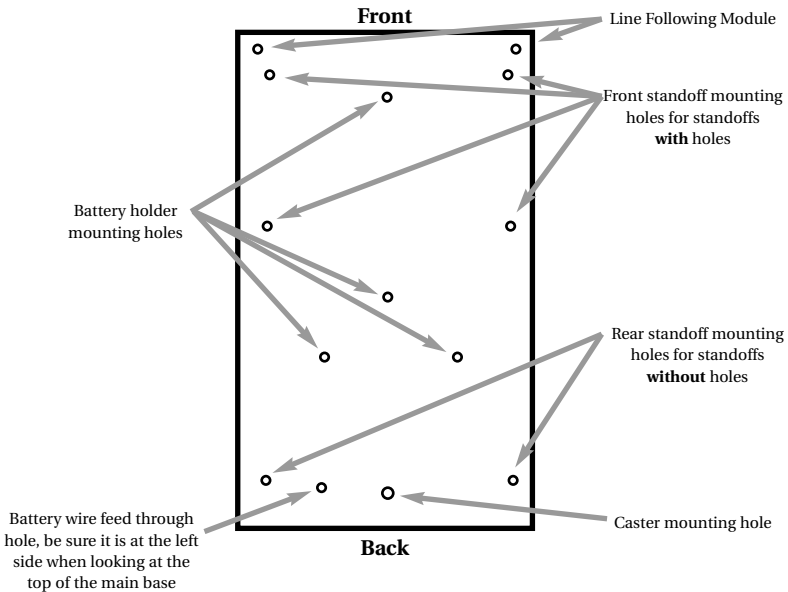
Step 1: Screw the six aluminum standoffs to the chassis base with six 6-32 x 7/16" screws. Do not tighten all the way but leave them just slightly loose.

Note that there are two types of standoffs; four of them have holes and two do not. The four with holes go toward the front of the robot, near the battery holder, but on the opposite side. The picture below is a side view of the top of the base, and shows how the standoffs are located. There is one hole towards the back of the base that is only on one side. Refer to the diagram on page 30 entitled "The Main Base as Viewed from the Top". Orient your base to match the diagram,



with the two widely-spaced holes near the edge at the top and the battery wire feed-through hole on the lower left. If this hole is on the right, turn the base over and temporarily stick a little piece of tape on the base near the caster mounting hole to mark the top. The standoffs with the holes must be put in the correct way. The holes in the standoffs are farther away from one end and this is the end that should be attached to the base.

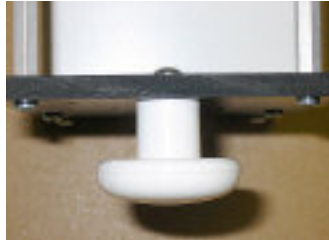
The Main Base as Viewed from the Top



Step 2: Attach the servos to the front standoffs with 4-40 x 1/2" panhead machine screws, washers and nuts. Place the servos between the front standoffs so that the wire of the servo faces the back of the base. The servos go flat to the base of the robot like in the picture. Put the washers onto the screws and then fit the screws through the servo holes and through the standoff holes. We are only using the top hole at each end of the servo. Therefore, each servo is held by only two screws. Put the nuts on the screws where they come through the standoffs, but only snug them down for now. The top hole in each standoff is not used at this time, as they are included for mounting the optional wheel encoders.



Step 3: Put the rear caster onto the bottom of the main base. It is attached with a 8-32 x 3/4" panhead machine screw. The screw comes from the top of the base and is tighten in place.



Step 4: Attach the breadboard support base to the two front standoffs using 6-32 x 7/16" screws. Attach with just the two front screws for now.

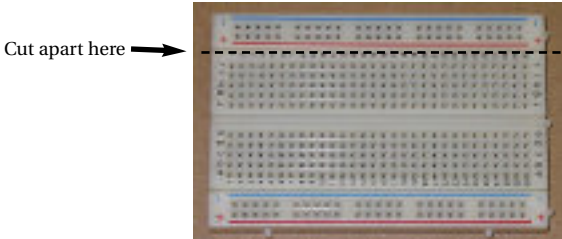


Step 5: Remove the battery wires from the main controller board by unscrewing them. Set the battery holders aside, they will be reconnected after the base is together. Attach the main controller board to the four rear standoffs with four 6-32 x 7/16" screws. The back two standoffs get the 1/8" thick nylon spacers between the main board and the top of the standoffs. The DB9 serial connector goes to the back of the robot (the end with the caster). You can now tighten all the screws holding the standoffs and servos.

Plug the servos into the servo headers on the main board. Plug the connectors into the board so that the brown wire goes to the (-), the orange wire to the (S), the red wire will go the the middle or (+). You can route the wires around the standoffs as shown in the picture below. Be sure the right servo wire gets plugged into the right servo header and like wise for the left.



Step 6: Prepare the breadboard for installation. Look closely at the breadboard in the kit. It will have two double rows of holes along both long sides. We need to remove one of these rows to make it fit. Place the board so that the numbers that run along the two sides are right-side-up, with “1” at the top and “30” at the bottom. We want to remove the two rows at the right labeled + and -. To do this bend that strip of the board back and use a hobby knife to cut the foam tape that is holding them together.

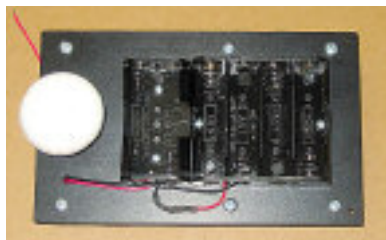


Step 7: Attach the breadboard to the breadboard support base with velcro.

Remove the protective paper backing from the back of the breadboard's foam tape. Also remove the protective backing from the back of the velcro hook part. Place the velcro onto the bottom of the breadboard sticking it to the foam tape on the back of the breadboard. The strip of hook velcro should just cover the bottom of the breadboard. Remove the protective paper from the back of the pile velcro and apply it to the breadboard support base. Keep it back from the edge some so it doesn't cover the screws holding the support base. Align the breadboard with the remaining (+ -) strip closest to the main controller board and stick it onto the support base.



Step 8: Screw the battery holders to the bottom of the main base with four 4-40 x 1/4" sheet metal screws. The 4-cell holder goes to the front of the robot. Position the holders so that the wires come out of the holder on the same side of the robot as the battery wire hole, as shown in the illustration.

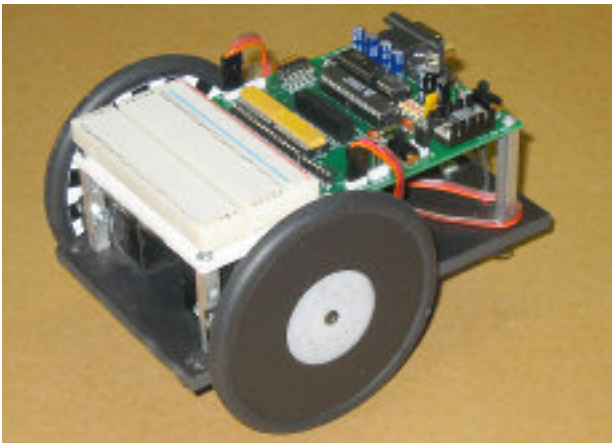


Step 9: Reconnect the battery wires to the main controller board.

Route the battery wires through the hole in the base close to the caster. You may want to trim the wires so they will reach the power holes in the main controller board with about 1/2" of slack. Be sure to connect the black lead to the terminal marked (-), it is the one closest to the push button (SW1). The red lead goes to the terminal marked (+), it is the one closest to the slide switch (SW2).

Step 10: Attach the wheels to the servo shafts. They are held in place with the screws you removed from the servo horn when you converted the servos to continuous rotation.

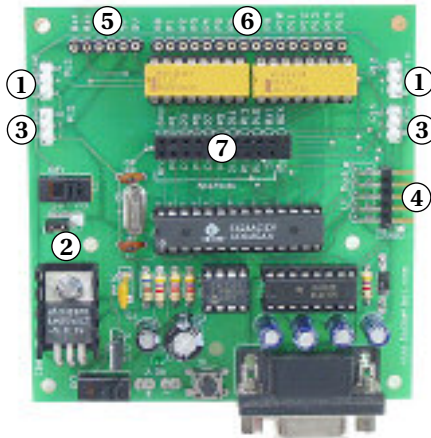
Put the batteries back into the holder and your CBA robot is done and ready to run.




The Connectors of CBA

The main controller board of CBA contains many connection points that can expand and allow experimentation with CBA. It takes less than 10 seconds to convert CBA from a highly-capable, autonomous, wheel-encoded bot with line following capabilities to a fully-developed BS2e prototyping system with a solderless breadboard, heavy duty power supply and serial connection to the BASIC Stamp Editor running on a PC.


The illustrations are all shown with the front of CBA to the top of the page.



(1) Servo Headers

 There are two connectors dedicated to driving the servos. The lower pin is ground. The center pin is power to the servo. The center pin will supply either regulated 5 volts or full battery voltage depending on the jumper setting of the servo power jumper (2). The top pin is the signal directly from the BASIC Stamp. The right servo is connected to Stamp pin P12 and the left servo is connected to Stamp pin P13.

(2) Servo Power Jumpers

 There is a header next to the servo power switch that controls the voltage supplied to the servos. The black jumper supplied with the kit goes on the pins of the header. If the jumper is on the left two pins, the servos will be supplied full battery voltage. If the jumper is on the right two pins, the servos will be supplied a regulated +5 volts.

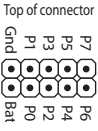
The servos in the kit can be used at up to 8 volts. If you use six rechargeable (NiMH are recommended) batteries, or five alkaline batteries and the battery replacer, you can use full battery voltage for the servos. If you are using six alkaline batteries, you should use only the regulated 5 volt setting.

(3) Wheel Encoder Connectors


 There are two headers used for connecting the wheel encoder add-on kit. The lower pin is ground. The center pin is +5 volts. The top pin is the signal directly to the BASIC Stamp. The right side is connected to Stamp pin P14 and the left side is connected to Stamp pin P15.

If you want, the wheel encoder connectors can be used as general purpose connectors for any type of sensor that needs power and one I/O pin. However, be aware that at these points P12 – P15 are connected directly to the BS2e I/O pins, and therefore are NOT protected by 220 ohm series resistors.


(4) Line Following Module Connector

 The line following module connector is designed specifically for the CBA line following module. The connections are shown in the illustration. While this connector is not keyed (nor are any of the others), there will be no problem if the line follower cable is routed to the bottom, directly over the edge of the board. If the line following module is not connected, this header can connect to other custom designed modules. Again, you must remember that at this point P0 – P7 are connected directly to BS2e I/O pins, and therefore are vulnerable to damage if improperly connected to anything other than the CBA line follower module. Also, the voltage to the connector is full battery voltage and would need to be regulated by whatever is connected.

(5) Power Connectors for Bread Boarding

 The power connector for bread boarding is a handy place to connect wires from when you are using the bread board area. This connection has two points to connect to for each of the three types of power. Full battery power, ground and regulated 5 volts are all available. Use 24 or 26-gauge solid wire to connect these header pins to points on the bread board.

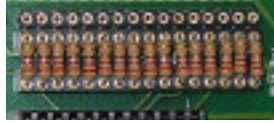
(6) BASIC Stamp I/O Pins for Bread Boarding

 This set of connectors brings all of the BASIC Stamp 2's 16 I/O pins to the front of CBA. The pins are labeled P0 through P15. These match the 16 pins on a "normal" BS2e with a notable exception. That exception is the 220 ohm resistor networks next to this header. These resistors protect the BASIC Stamp from driving too much current in the case of a shorted condition on a pin. In most cases these resistors can be left in place. Should they interfere with some circuit you build on the bread board, they can be removed and replaced by a short piece of wire. If you remove a resistor network you will break the connection between the BASIC Stamp and the connectors. To keep them connected you will need to jumper between each set of DIP pins with either a piece of wire or one of the extra supplied 220 ohm 1/4 watt resistors.

The following diagram shows how to cut and bend the leads of the resistors so they will fit the DIP sockets. The sample pictures shows resistors at

all 15 locations although only eight extras are supplied in the kit. You can mix and match resistors and jumper wires as needed.

Bend and cut leads to match this



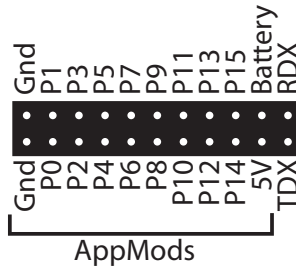
Note that in the normal configuration of CBA, some of these pins are in use by other items. Pins P12 and P13 are used for the servos. Other pins may be in use depending upon installation of the wheel encoders or the line following module. We have included all the pins to provide flexibility in your future experiments and expansions. If you pull the plugs on all of the connectors, you are left with a BS2e package that is larger, but otherwise similar (and electrically identical) to the Parallax BS2 Homework Board. CBA's prototyping area is almost twice as large, and it can deliver four times as much regulated power to your experimental circuits from a battery pack that dwarfs the Homework Board's 9V supply.

(7) Expansion Header



This 22-pin header allows expansion of the CBA controller board. It contains all the BS2e I/O pins, the three power types, and connections to the Max 232 serial level shifter. Be aware that none of the BS2 I/O pins are protected on this header. CBA expansions could range from an LCD display to a complete replacement for the BASIC Stamp 2 controller. Pinouts for this header are shown below.

Of special note is that the left most 20 pins of this header match the pins of the Parallax® AppMods. So if you have any of the AppMods you can plug them right into the CBA main board. We have even included a mounting hole that matches the mounting hole on the AppMods. Again note though that in the normal configuration of CBA, some of these pins are in use by other items. Pins P12 and P13 are used for the servos. Other pins may be in use depending upon installation of the wheel encoders or the line following module and may interfere with the AppMods.



Loading and Running Programs on Your CBA Robot

Because CBA uses a standard BASIC Stamp 2 microcontroller it is programmed the same as any other BASIC Stamp 2. A programming manual, "BASIC Stamp Manual Version 2.1" for the BASIC Stamp is included on the Parallax CD that came with your kit. You can access it in the Documentation area of the CD. It is in the BASIC Stamps folder.

If you'd like a printed copy of "BASIC Stamp Manual Version 2.1" it can be purchased through Parallax on their web site.

The Basics of Programming the CBA

The programming environment for CBA is Parallax's BASIC Stamp Editor. The BASIC Stamp Editor along with documentation is available on the included Parallax CD. The CD contains a wealth of information about the BASIC Stamp microcontroller.

The version of the BASIC Stamp Editor on the CD is v2.1. If you have an older version installed on your computer you should really install the newest version. It is a great improvement over older versions and all the sample programs for CBA are written using it. Install the BASIC Stamp Editor software using the instructions with the installer.

The CBA connects to your computer with a 9 pin serial cable. This needs to be a normal "straight through" cable, not a null modem cable. The female end of the cable connects to your computer, while the male end connects to the DB9 connector on CBA.

Programs are written using the BASIC Stamp Editor. Once written they are loaded into the BASIC Stamp on CBA using the RUN command in the Editor. We'll do a short example here to give you the feel for the process. For a complete explanation of how to use the BASIC Stamp Editor see the BASIC Stamp manual listed above.

1. Start the BASIC Stamp Editor program. It will open a blank programming window.
2. Enter the following into the programming window. Watch to be sure you use { } and not () in the first line:

```
'{$STAMP BS2e}  
DEBUG "Hello, World!"
```
3. Be sure the serial programming cable is connected and the power switch on CBA is on.
4. To download the program to the BASIC Stamp on the CBA do the following. In the RUN menu select the RUN command, or click on the black, right-facing triangle

If the program was typed correctly, a window will appear briefly showing the progress of the download. This may be less than a second. After this window closes another window will appear. This is the debug window. The debug window will show "Hello, World!"

There are two things that could have gone wrong if the program didn't work.

The first is the program was typed wrong. If this is the case, it will not load and you will get an error message. When you close the error message, some part of the program will be selected. The editor thinks this selected area is part of the problem. The error will be in this general area, but may not be the item selected. Correct the error and try the RUN command again.

The second thing that could have happened is that the Editor couldn't find the BASIC Stamp. Check that the serial cable is connected and the power on CBA is on. Also, check that the first line of the program is: '{\$STAMP BS2e}.

What Happens When You Run a Program

When you selected the RUN command the following happened:

1. The Editor processes all your programming code and sends it in a compressed format through the serial cable to the BASIC Stamp 2 on CBA.
2. The BASIC Stamp on CBA stores these program codes in the memory on CBA. At the same time it erases anything that was stored there before.
3. The Editor then resets the BASIC Stamp controller on CBA.
4. CBA then runs the program that was placed in its memory.
5. In this case the programs only real instruction was to send "Hello, World!" back through the serial cable to your computer. The Editor knows what to do with information coming back to it and displays it on your screen, using the Debug Terminal.

To show that it was really CBA that was sending the information, and not just your computer being tricky, try an experiment. Press the reset button (the push button next to the DB9 serial connector) on CBA and watch the debug window. Every time you press the reset button the debug screen will get another Hello World! from CBA. You can even turn CBA's power off and then back on a few seconds later. CBA will restart and run the program again. If for some reason a new "Hello, World!" didn't show up press the reset button to start the program again.

Those are really the basics of programming the BASIC Stamp in CBA. You can load and run programs as many times as you want. You can only have one program loaded at a time because the BASIC Stamp erases any other program before it loads the new one. However, your CBA robot is very versatile. You can plug a 4PST DIP switch into the breadboard and connect it to P8 – P11. 10K resistors can be connected from the pins to +5V, and the switches themselves to ground. When a switch is open, the pin will read "high", but when a switch is closed, the pin will read "low". It is possible to have up to 16 different sub-programs loaded at the same time that can be selected by setting the DIP switch and then pressing "Reset". You will find a sample program on the BudgetBot website that provides the switch-selection code for this advanced function.

Some Programming Basics for CBA

The BASIC Stamp and thus, CBA, are relatively simple to program. We'll touch on some of the basics for making the servos drive CBA around. For more information on programming a BASIC Stamp to drive servos you should check out the manual from Parallax called "Robotics with the Boe-Bot (v2.2)". It is on the included Parallax CD. You can find it in the Documentation are listed under "Educational Curriculum".

It's a great, in-depth exploration of using a BASIC Stamp to drive a robot. Because the Boe-Bot™ uses the same I/O pins to drive the servos as CBA does, you can use most of the programs in the book with no alterations.

The following examples are presented with the understanding that you have some knowledge of BASIC Stamp programming. If you are starting with programming in general, or are unfamiliar with the BASIC Stamp you would be wise to start with the basics presented in either "BASIC Stamp Manual Version 2.1" or "Robotics with the Boe-Bot". If you haven't loaded programs to your CBA yet, take a few moments to read the section on Loading and Running Programs on Your CBA Robot. It gives the basics of getting programs into CBA. If you don't want to type in the sample programs, They are on the BudgetBOT.com CD that is included with the kit. They are also available for downloading at: <http://www.budgetbot.com/downloads/>

Making a Servo Move

To make a servo move, you have to send it a particular pulse for a set period of time. This pulse is a change on the I/O pin connected to the servo. A pulse is a change from 0 volts to 5 volts and back to 0 volts. In an electronic circuit the 0 volts is known as "low" and the 5 volts is known as "high". These LOW - HIGH - LOW transition make the pulse. The length of time the I/O pin is HIGH is the length of the pulse. For the control pulse going to a servo, these pulses must be between 1 ms (millisecond) and 2 ms in length. A millisecond is 1/1000 of a second.

With the basic stamp these pulses are sent with the `PULSOUT` command. The format for the command is:

```
PULSOUT pin, duration
```

The "pin" is the pin number of the BASIC Stamp to which the servo is connected. On CBA, that is pin 12 for the right servo and pin 13 for the left servo.

The `duration`, is the length of time in 2 microsecond (1/1,000,000 of a second) intervals that the pulse is HIGH. There are 1000 microseconds in a millisecond. For a servo, we need durations set somewhere between 500 (1 ms = 1000 microseconds / 2) and 1000 (2 ms = 2000 microseconds / 2).

These pulses need to be sent in a repeating fashion. If we send only one pulse the servo may move some but then stop. For the servo to continue to turn it must receive a pulse every 20 ms to 30 ms. To do this we must repeat the `PULSOUT` command on a regular basis. To show how this works, enter and run the short program below.

“Sample Program 1.bs2”

```
{ $STAMP BS2e}
{ $PBASIC 2.5}

DO
    PULSOUT 12, 600
    PAUSE 20
LOOP
```

The right servo will turn, when you run this program . This program works by sending a pulse, then waiting with the `PAUSE` command for 20 ms, then looping back up to the `PULSOUT` again. It runs over and over until you turn the power off, or load another program.

Experiment by changing the “600” value, the duration of the `PULSOUT`, to other numbers between 500 and 1000. Watch the direction and speed of the wheel. Try values like 500, 730, 750, 800 and 900. Each time you change the number just use the `RUN` command in the BASIC Stamp Editor. There is no need to turn CBA off between times you `RUN` a program.

You will see that values less than 750 make the wheel turn in one direction and values above 750 make the wheel turn in the other direction. Also, note that the closer a number is to 750, the slower the wheel turns.

Making CBA Go Forward and Turn

If you had set CBA down while it was running the last program, you would have seen it turn in a circle. With only one wheel moving, CBA will not get too far. To make CBA go forward, we need to add the left servo to our loop. Revise the program you entered before to include the changes highlighted below.

“Sample Program 2.bs2”

```
{ $STAMP BS2e}
{ $PBASIC 2.5}

DO
    PULSOUT 12, 650    'Right Servo (<750 runs forward)
    PULSOUT 13, 850    'Left Servo (>750 runs forward)
    PAUSE 20
LOOP
```

Run this program, turn CBA off, then place CBA on the floor and turn him back on. He should drive ahead at close to full speed. If he doesn't start moving when you turn the power on, momentarily press the reset button and he will start.

Experiment with changing the duration for the two `PULSOUT` commands. Try switching the two numbers by making 650 into 850 and 850 into 650. This will make CBA go backwards. Or try 700 for the right servo and 850 for the left. Then CBA will do a wide right turn.

Making CBA Move, Then Stop

Lets take things just a bit further. Up until now we have been using a `DO . . . LOOP` to loop over and over again sending one pulse for every loop. Lets change this to a `FOR . . . NEXT` loop so that it can run for a set number of loops before the program stops. Enter and run the program below.

“Sample Program 3.bs2”

```
{ $STAMP BS2e}
{ $PBASIC 2.5}

X   var Byte

FOR X = 1 to 90           'loop thru 90 times
    PULSOUT 12, 650      'Right Servo (<750 runs forward)
    PULSOUT 13, 850      'Left Servo (>750 runs forward)
    PAUSE 20
NEXT
```

Run this program, turn CBA off, then place CBA on the floor and turn him back on like you did before. CBA will move forward for 90 loops, or about 2 seconds. At that point, the program will stop and no more pulses will be sent. This will cause CBA to stop.

Experiment here again. Change the number of loops from 80 to some number between 1 and 255. See how long the different setting takes to run.

The Servo Pulse Loop

Until now, we have been using a rather simple program to show how the servos of CBA work. For the next example we will make a more functional program. To control the speed of CBA, we will make some constants that can be used to control the duration part of the `PULSOUT` command. We will also name the servo output pins as constants so they are easier to remember.

“Sample Program 4.bs2”

```
{ $STAMP BS2e}
{ $PBASIC 2.5}

X   var Byte           'Variable for use in FOR..NEXT loop
MStop   CON 750        'Pulse width for the servos while stopped
Speed100 CON 250       'This amount added or subtracted from stop
                               'makes the servos turn full speed

RMotor   CON 12        'Servo Pin for right servo
LMotor   CON 13        'Servo Pin for left servo

PulseWidthRt VAR Word   'Variables for pulse widths to servos
PulseWidthLt VAR Word   'These hold the duration value

PulseWidthRt = MStop - Speed100      'Same as above but minus for
                                       'right servo to go forward
```

```

PulseWidthLt = MStop + Speed100      'Set Speed to stop speed
                                      'plus full speed

'*** Start of Main Program ***
FOR X = 1 to 90                        'Normal servo pulse loop
    PULSOUT Lmotor, PulseWidthLt      'Send pulse to left
    PULSOUT Rmotor, PulseWidthRt      'Send pulse to right
    PAUSE 20                           'Pause until next pulse
NEXT

```

Once you have this program entered and saved, RUN it like you have before. You will see that it works just like the last program we had. The constants MStop, Speed100, LMotor and RMotor are just ways to make the program more flexible and understandable.

The use of PulseWidthRt and PulseWidthLt allows us to use the same PULSOUT command later in a more complicated program where we change the pulse duration during the program.

Going Straight

You may have noticed that CBA tends to drift off to one side instead of going in a straight line. This happens because two servos are never the same and because servos receiving equal, but opposite offsets from “stop” (1.5 ms), tend to spin at a different speed going forwards than they do going backwards.

With the next program, we will take care of these problems and expand on how to use variables to change the speed of CBA during a program.

“Sample Program 5.bs2”

```

'{$STAMP BS2e}
'{$PBASIC 2.5}

X          var Byte      'Variable for use in FOR..NEXT loop
ListItem  var nib       'This is used for the Branch statement
PulseWidthRt VAR Word   'Variables for pulse widths to servos
PulseWidthLt VAR Word   'these hold the duration value

LMotor    CON 13        'Servo Pin for left servo
RMotor    CON 12        'Servo Pin for right servo

Speed100L CON 870      'Note that each Speed setting
Speed075L CON 820      'is different for each servo
Speed050L CON 805      'These were found by trial and error
Speed025L CON 790
Speed010L CON 780
Speed100R CON 500
Speed075R CON 650
Speed050R CON 690
Speed025R CON 705
Speed010R CON 720

```

```

*** Start of Main Program ***
FOR ListItem = 0 to 4
  Branch ListItem, [Slowest, Slow, Med, Fast, Fastest]

  PulseLoop:
  FOR X = 1 to 45
    PULSOUT Lmotor, PulseWidthLt 'Normal servo pulse loop
    PULSOUT Rmotor, PulseWidthRt 'Send pulse to left
    PAUSE 20 'Send pulse to right
    PAUSE 20 'Pause until next pulse
  NEXT
  PAUSE 1000 'Pause for 1 sec. after one speed is done
NEXT
END
Slowest:
  PulseWidthLt = Speed010L
  PulseWidthRt = Speed010R
GOTO PulseLoop

Slow:
  PulseWidthLt = Speed025L
  PulseWidthRt = Speed025R
GOTO PulseLoop

Med:
  PulseWidthLt = Speed050L
  PulseWidthRt = Speed050R
GOTO PulseLoop

Fast:
  PulseWidthLt = Speed075L
  PulseWidthRt = Speed075R
GOTO PulseLoop

Fastest:
  PulseWidthLt = Speed100L
  PulseWidthRt = Speed100R
GOTO PulseLoop

```

Once you have this program entered and saved, RUN it like the other programs. CBA will run forward slowly for two seconds, stop for a second then move forward again faster. It will do this 5 times until it is at full speed. It will then stop. It should also be moving in close to a straight line. If CBA is not moving in a straight line don't worry. We'll touch on how to change the program to make him do that in a minute.

We have two new concepts added here. The first is the use of a number of constants to set different speeds. You will see them at the top of the program as:

```

Speed100L   CON   870
Speed075L   CON   831
.
.
Speed010R   CON   720

```


These are the different pulse widths needed to make the each servo go at about 100%, 75%, 50%, 25% and 10% of the full servo speed. These pulse widths were determined with some trial and error using this program. You will note that there is a constant for each servo for each speed. This will allow us to change the speed for each servo and make CBA go in a straight line.

The other new concept is the `BRANCH` command. The branch command is in a `FOR . . . NEXT` loop that surrounds the servo pulse loop. Each time the outside `FOR . . . NEXT` loops, it increments the variable `ListItem`. The `BRANCH` command sends the program to the point called in the list of labels at the end of the `BRANCH` command. The first time though it branches to `Slowest :` because `ListItem = 0`. On the next loop `ListItem = 1` and it will branch to `Slow :`. At the end of each of the labels that the program branches to is a `GOTO` that sends the program back to the servo pulse loop. The loop runs 45 times (about 1 second) and then pauses for 1 second. The programs goes through the loop 5 times; once at each speed.

The real advantage to the constants is that they are all in one nice neat place in the program. It is simple to change the value in one place and know it has been changed throughout the program.

If your CBA didn't travel in a very straight line, do the following:

1. Make note of which way it turns at each of the speed levels. In order it will run at 10%, 25%, 50%, 75% and 100%.
2. Change the values of the constants for that speed. Normally you want to slow down the wheel that is going too fast. That is if CBA is turning to the right, you need to slow down the left wheel. To slow down a servo, change the pulse width value to a number that is closer to 750.
3. Make note of what you changed, and `RUN` the program again. If CBA still wanders off a straight line, go back to step 1 and repeat as needed.

Enter the values that you find work best for your CBA in the chart below. You will be able to use these values in the constants for any future programs you write.

Servo Pulse Width Chart

Speed	Left Pulse Width	Right Pulse Width
100% Forward		
75% Forward		
50% Forward		
25% Forward		
10% Forward		
Stopped		

Scripted Movement

We'll leave you with one last program here to get you going. This one will build on the last, giving CBA a planned route to drive.

If you came up with new numbers for the different speed/servo combinations in the last program you should use them in this program. Just change the values shown below to the values you developed for your CBA by experimenting.

“Sample Program 6.bs2”

```
{ $STAMP BS2e}
{ $PBASIC 2.5}

X          VAR Byte  'Variable for use in FOR..NEXT loop
ListItem  VAR Nib   'This is used for the Branch statement
LoopTimes  VAR Word  'This is used for timing loop
DataAddress VAR Byte 'Tracks where we are in the data
PulseWidthRt VAR Word 'Variables for pulse widths to servos
PulseWidthLt VAR Word 'these hold the duration value

LMotor    CON 13      'Servo Pin for left servo
RMotor    CON 12      'Servo Pin for right servo

MStop     CON 750     'Basic Speed setting for servos
Speed100L CON 870     'Note that each Speed setting
Speed075L CON 820     'is different for each servo
Speed050L CON 805     'These were found by trial and error
Speed025L CON 790
Speed010L CON 780
SpeedRevL CON 690     'This is reverse at 50% left servo
Speed100R CON 500
Speed075R CON 650
Speed050R CON 690
Speed025R CON 705
Speed010R CON 720
SpeedRevR CON 815     'This is reverse at 50% right servo

'The Data format below is a follows, the first number is the
'control command. 0=Slow, 1=Med, 2=Fast, 3=Turn Right,
'4=Turn left, 5=Reverse, 6=Stop, if command = 7 end program
'the next number is how long in servo loops to do the above
'command. 45 loops is about 1 second.

DATA 1, 20
DATA 2, 40
DATA 3, 20
DATA 2, 40
DATA 3, 20
DATA 2, 40
DATA 6, 80
DATA 5, 80
```

```

DATA 4, 100
DATA 7, 7
DataAddress = 0
PAUSE 4000
'*** Start of Main Program ***
DO
  READ DataAddress, ListItem
  READ DataAddress + 1, LoopTimes
  DataAddress = DataAddress + 2
  Branch ListItem [ Slowest, Med, Fastest, RightTurn,
    LeftTurn, Backwards, StopMoving, StopProgram]

  PulseLoop:
  FOR X = 1 to LoopTimes           'Normal servo pulse loop
    PULSOUT Lmotor, PulseWidthLt 'Send pulse to left
    PULSOUT Rmotor, PulseWidthRt 'Send pulse to right
    PAUSE 20                       'Pause until next pulse
  Next
LOOP

Slowest:
  PulseWidthLt = Speed010L
  PulseWidthRt = Speed010R
GOTO PulseLoop

Med:
  PulseWidthLt = Speed050L
  PulseWidthRt = Speed050R
GOTO PulseLoop

Fastest:
  PulseWidthLt = Speed100L
  PulseWidthRt = Speed100R
GOTO PulseLoop

RightTurn:
  PulseWidthLt = Speed100L
  PulseWidthRt = MStop
GOTO PulseLoop

LeftTurn:
  PulseWidthLt = MStop
  PulseWidthRt = Speed100R
GOTO PulseLoop

Backwards:
  PulseWidthLt = SpeedRevL
  PulseWidthRt = SpeedRevR
GOTO PulseLoop

StopMoving:
  PulseWidthLt = MStop
  PulseWidthRt = MStop
GOTO PulseLoop

StopProgram:
END

```

← (Continued on next line
but type all on one line)

Once you have this program entered and saved, RUN it like the other programs. CBA will run through a set list of commands, going forward, turning, stopping, backing up and the like. The behavior is controlled with the DATA statements that are before the main program. The DATA command stores the numbers listed behind it into the memory of CBA. When it gets to the main program it READS the data one number at a time and loads them into the ListItem and LoopTimes variables. These two variables control what command the servo pulse loop gets and how many times the loop runs.

I know it may look complicated, but it's not that different from the last program we did. Run it a few times, then change some of the numbers in the DATA statements. You can add or subtract statements if you need. What the numbers do is listed in the comments of the program.

These basics should get you on your way to making CBA move about as you want. For other programs check the BudgetBot.com web site at: www.budgetbot.com. We will have a place for uploading and sharing programs with fellow CBA builders. Stop by and join in the fun.

Building the Battery Replacer

If you are using alkaline batteries in your CBA you will need to construct the battery replacer. This simple item takes the place of one battery in the battery holder. This is needed to keep the voltage supplied to the servos at less than 8 volts. If you were to use 6 AA alkaline batteries the total voltage would be 6×1.5 volts or 9 volts total. By replacing one of the batteries with the battery replacer you will have 5×1.5 volts or 7.5 volts total.

Step 1: Take the copper nail and place it into the hole that runs through the center of the wooden dowel. Press it all the way in until the head of the nail is tight to the end of the dowel. The nail should be held tight in the hole. If for some reason the nail seems loose, use a small spot of epoxy under the head of the nail to hold it in place.

To use the battery replacer, insert it into the battery holder just like a battery. The end of the replacer that has the head of the nail goes to the spring end of the battery holder. You can put it in any of the battery positions but the position closest to the back of the robot works best for keeping CBA balanced.



