

Getting Started with LogoChips

Electronics for Everyone	3
LogoChip: A new electronic construction kit.....	5
Build Your Own LogoChip	5
<i>LogoChip Parts List</i>	6
<i>Serial Interface Parts List</i>	7
<i>Overview of the Construction Process</i>	7
<i>LogoChip Schematic.....</i>	10
<i>Preliminaries.....</i>	11
Tools required.....	11
The LogoChip Sticker.....	12
The art of debugging	12
<i>Step by Step Instructions</i>	14
Step 1 - Connect the power and ground busses.....	14
Step 2 - Add the LogoChip.....	15
Step 3 - Connect the LogoChip to the power and ground busses.....	16
Step 4 - Get a bootflash.....	17
Step 5 - Add the bypass capacitors.....	21
Step 6 - Add a power switch.....	23
Step 7 - Add the start/stop button	24
Talk to your LogoChip	26
<i>Preliminaries.....</i>	26
Desktop Software Installation	26
<i>Connecting to your LogoChip.....</i>	27
If you have a serial interface board.....	28
And if you don't have a serial board.....	30
Step 1 - Make a serial cable.....	30
Step 2 - Connect the cable to the breadboard.....	31
Step 3 - Add two resistors to complete the serial connection	33
<i>The LogoChip Logo User Interface</i>	34
Troubleshooting	35
Getting Flashy.....	37
LogoChip Ins and Outs.....	38
The Autonomous LogoChip	40

Getting Started with LogoChips

Making Colors.....	41
Let's Get Moving.....	42
Making Music.....	43
Getting a Sense of the World	44
<i>Digital Sensors</i>	<i>44</i>
<i>Analog Sensors</i>	<i>45</i>
Speeding Things Up.....	46
Appendix: Clone Your Own LogoChip.....	48

Getting Started with LogoChips

Electronics for Everyone

Eventually the art went out of radio tinkering. Children forgot the pleasures of opening and eviscerating their parents' old Kadettes and Clubs. Solid electronic blocks replaced the radio set's messy innards—so where once you could learn by tugging at soldered wires and staring into the orange glow of the vacuum tubes, eventually nothing remained but featureless ready-made chips, the old circuits compressed a thousandfold or more. The transistor, a microscopic quirk of silicon, supplanted the reliably breakable tube, and so the world lost a well-used path into science.

— James Gleick,
Genius: The Life and Science of Richard Feynman
[1992]

Electronics lies at the center of much of technology of our times. We find ourselves surrounded by ever more powerful and, in many ways, ever more complicated and mysterious electronic gizmos and gadgets. In spite of the central role that these devices often play in our lives, the inner workings of these “black boxes” are almost completely hidden from view and are often poorly understood by their users. Electronics is typically viewed as a formidable subject that is best tackled by expert practitioners.¹

Why is it that most people view the subject of electronics as inaccessible and esoteric? Well, for starters, the electronic world is inherently a more abstract and less familiar one than, say, the mechanical world. We interact with the mechanical world directly with our senses, which has led to a multitude of ways in everyday life for us to play with mechanical ideas, building with blocks or taking apart a car engine for example. There have never been many good opportunities for novices to play with electronics, but in recent times the situation has gotten noticeably worse. The ever increasing complexity of our electronic devices has served over time to diminish the depth of our experience with the inner workings of the electronic world. In the days before the coming of solid state electronics and integrated circuits, one could more easily look at a circuit and see how the electron stream flowed. Over the decades the capabilities of our electronic technologies may have progressed at incredibly rapid rates, but

¹ Even among the “technically minded”, one finds a surprising amount of discomfort with even relatively “simple” electronics. For example it is striking how often in casual conversation working scientists will volunteer that they feel under prepared in their knowledge of electronics.

Getting Started with LogoChips

our access to good ways to for non-experts to creatively play with electronic circuits and ideas has lagged far behind. For a young Richard Feynman, the process of tinkering with a broken radio could serve as an ideal playground for exploring electronics (and also act as a launching pad for making connections to other scientific ideas). But for the present age of digital electronics, few comparable experiences beckon.

The standard treatments of electronics in books or the curriculum are often too formal and theoretical. Or they adopt the style of a cookbook, providing recipes for constructing interesting electronic gadgets, but not aiming to provide a deep understanding of the underlying principles. By only constructing circuits that have been designed by others, readers miss an important opportunity to learn the art of debugging a flawed design. In either case these approaches fail students along a critical dimension; they typically do not bring most to the point where they are able or even want to design their own electronic circuits or build their own electronic inventions.

In most of the standard introductory electronics experiences the devices being constructed often do not have a particularly compelling use. Typical are “bulbs and batteries” exercises, in which students use a kit consisting of a few batteries, flashlight bulbs and wires to build and explore simple dc circuits. While the underlying ideas that are in play here (*e.g.* series vs. parallel circuits, Ohm’s Law, short circuits, *etc.*) may be accessible at a fairly early age, it is hard to see how someone would be inspired to extend this kind of activity into a long term project. The remarkable electronic gadgets that surround us in the world today raise expectations to a very high level. Kids are bound to be somewhat disappointed with an exercise that results simply in turning on a flashlight bulb.

Introductory electronics books and kits usually adopt a bland style that appeals to a narrow segment of geeky hobbyists. To see what the client base for these books and kits looks like, just walk into your neighborhood *Radio Shack* and check out the customers, particularly the group hovering in the electronic components section: The group tends to be overwhelmingly male and the projects that are highlighted represent only a small range of the possibilities. (One telltale symptom: Few if any craft materials appear in the books.)

But this need not be so. With a more eclectic set of sample projects and a more imaginative exposition there are many more people who would become fluent designers and creators of electronic inventions. For example, there are many ways to make connections to the arts community by using electronics to create interactive art.

Getting Started with LogoChips

If the important and powerful ideas of electronics are to be more widely accessible, new tools and new methods are needed. We envision an approach that is playful in spirit while still seeking to engage learners in the deep underlying principles of electronics and the broad connections of these ideas to other areas of engineering and science.

LogoChip: A new electronic construction kit

Paradoxically, the same microelectronic technologies that have contributed to the black-boxing of modern electronic circuits can also be used as the basis for a new kind of construction kit that will help reintroduce a vigorously creative and playful dimension into the design of electronic inventions. The basic building block that we have developed for this kit is an easily programmable and inexpensive embedded microcontroller called the LogoChip. Users can develop programs using a special version of the Logo programming language on a desktop or laptop computer and then download these programs to the LogoChip. LogoChip Logo combines all the power and elegance of the Logo programming language with the ability to directly configure and control the individual pins on the LogoChip.

Build Your Own LogoChip

Here are instructions for building a new kind of microcontroller called the LogoChip that is both powerful and easy to use.










You can build the LogoChip on a small breadboard following the instructions below.

Getting Started with LogoChips

LogoChip Parts List

The parts that you need to construct a LogoChip are listed below. They are all available from Digi-Key Corporation of Thief River Falls, Minnesota (www.digikey.com).

The LogoChip consists of a PIC18F2320 microcontroller (made by Microchip) that has had a Logo “virtual machine” installed on it. You can buy the “raw” microcontroller directly from Digi-Key , but in order to have the LogoChip firmware installed, please contact Robbie Berg (rberg@wellesley.edu).



Quantity	Part	Manufacturer	Digi-Key Part Number	Cost ² (\$)
1	Logochip = PIC microcontroller with LogoChip firmware installed 	Microchip	PIC18F2320-I/SP-ND	7.93
1	solderless breadboard	3M	923253-ND	15.44
1	pushbutton 	Omron	SW400-ND	0.20
1	47 microfarad tantalum capacitor 	Panasonic	P2017-ND	1.07
2	0.1 microfarad capacitors 	Kemet	399-2127-ND	0.24
1	bi-color LED 	Lite-on	160-1058-ND	0.30
2	10k resistor 	Yageo	10KQBK-ND	0.04
1	330 ohm resistor 	Yageo	330QBK-ND	0.02
1	power switch 	E-Switch	EG1903	0.64
1	"AAA" battery holder (4 cells)	Keystone	2482K-ND	1.01
4	"AAA" batteries	Panasonic	P146-ND	1.52
1	10 MHz ceramic resonator (optional) 	Panasonic	PX1000MC-ND	1.14

² As of June, 2004.

Getting Started with LogoChips

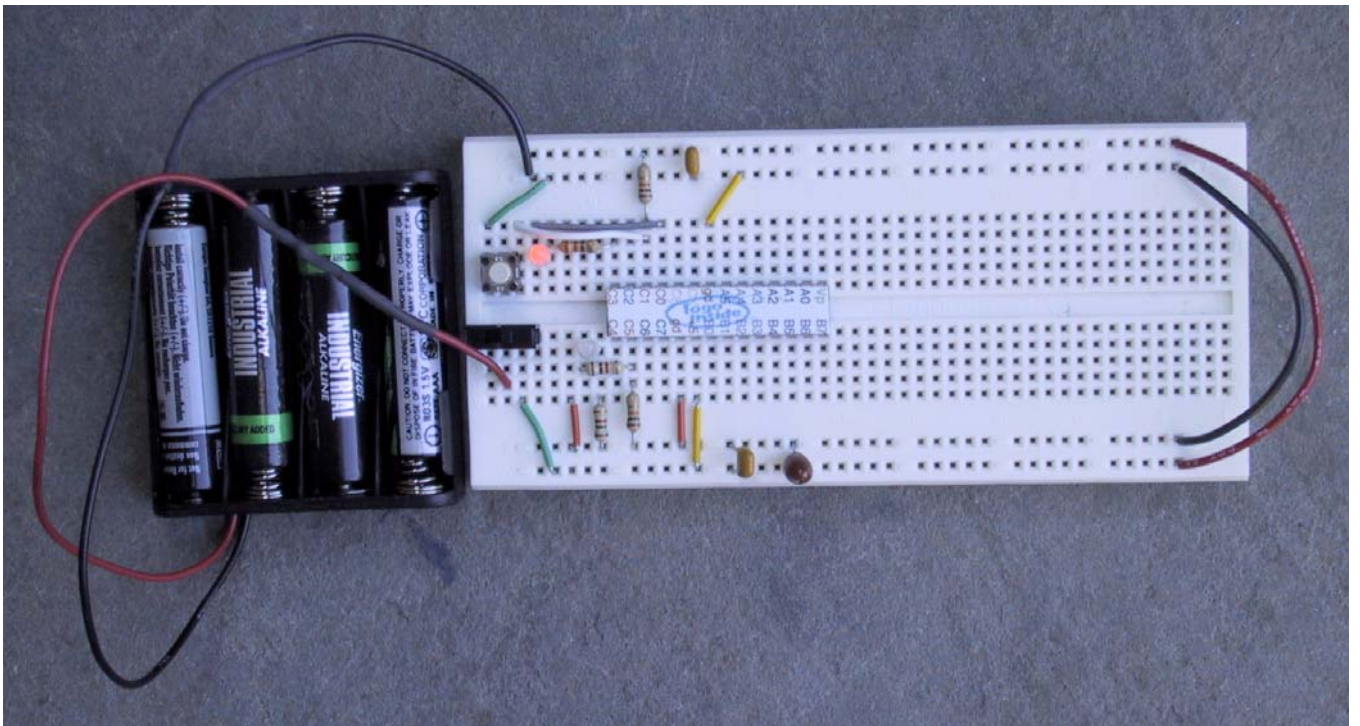
Serial Interface Parts List

To connect your LogoChip to a desktop computer you will also need the following parts.

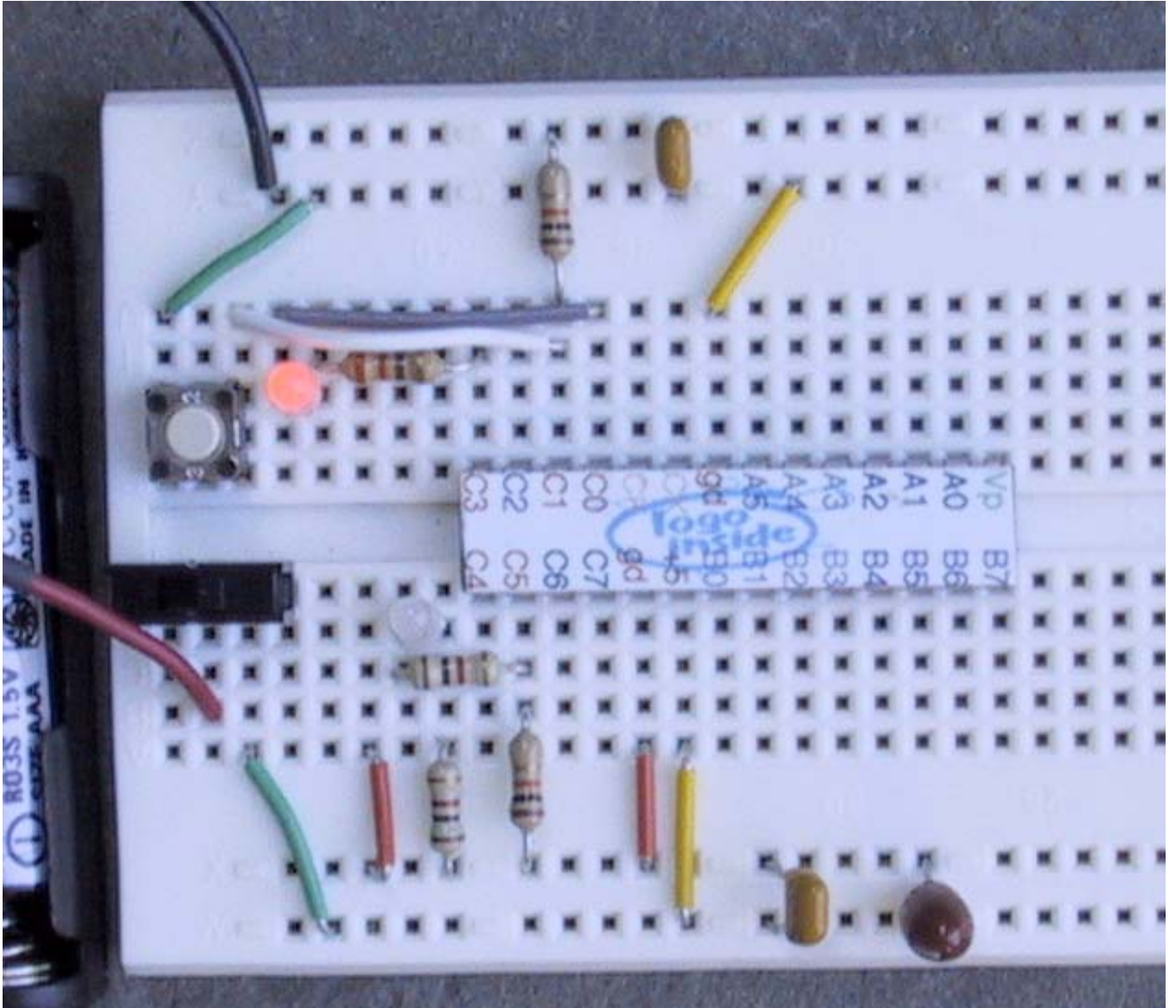
1	9-pin serial connector to ribbon cable	CW Industries	C7FXT-0906G-ND	4.33
1	0.1" male header (36 pin strip)	Sullins	S1032-36-ND	1.91
1	bi-color LED 	Lite-on	160-1058-ND	0.30
2	1k resistor 	Yageo	1.0KQBK-ND	0.04

Overview of the Construction Process

The completed LogoChip circuit, constructed using the parts listed above, is shown in the photos below.



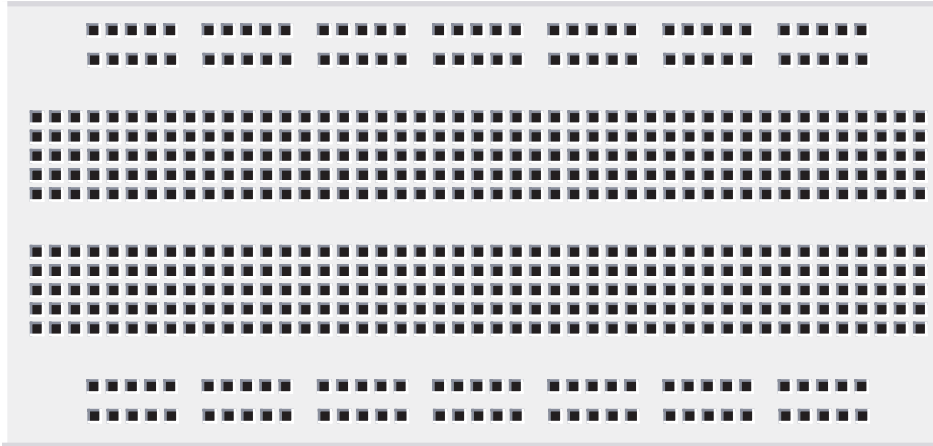
Getting Started with LogoChips



A completed LogoChip circuit, close-up

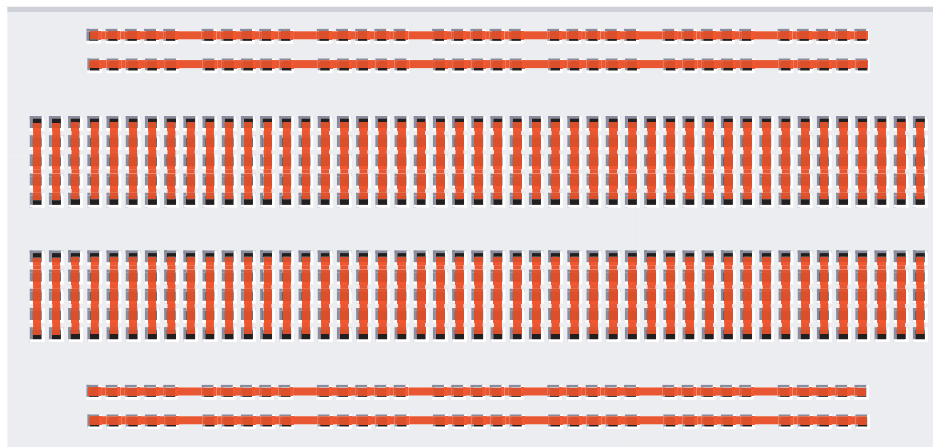
The circuit is constructed on an “electronic breadboard” of the kind shown in the photo below.

Getting Started with LogoChips



A blank breadboard

The red shaded regions in the figure below indicate which holes are connected on this particular breadboard (Digi-Key part number 923253-ND, made by 3M). Please note that other models of breadboards may have slightly different connections. In particular, there may be a gap in the middle of the long rows. If you are uncertain, the best way to check is to use an ohmmeter to see what is connected to what.



Breadboard connections

Getting Started with LogoChips

Preliminaries

First, some general advice and preliminary steps before you start building:

Tools required

- #22 *solid* hookup wire, (such as Digi-Key part number C2117W-100-ND)
- a good wire stripper (such as Digi-Key part number WS6-ND). Be careful when stripping the wire. Make sure the aperture in the stripper is the right size for #22 wire. (If it's too small you will nick the wire, which can lead to nasty problems down the road.)
- you can splurge and buy pre-stripped wire (Digi-Key part number 923351-ND), cut in a variety of lengths 0.1" increments and designed to fit perfectly into the digital breadboards.
- A "multi-meter" for measuring voltage, current, and resistance. Either an analog (such as Digi-Key part number BK117B-ND) or a digital (such as Digi-Key part number BK2408-ND) meter will do.
- a small soldering iron (such as Digi-Key part number WP25-ND) for making the serial cable

Getting Started with LogoChips

The LogoChip Sticker

Use the handy sticker below to help identify the pins on the LogoChip. Print out a copy of this figure, carefully cut out a sticker, and use a glue stick to attach the sticker to a PIC18F2320 microcontroller. **It is critical that the sticker be attached with the proper orientation. The top of the sticker (the end with the pins labeled Vp and B7) must be attached to the end of the microcontroller that has a notch in it.**



LogoChip Stickers



An enlarged view of a LogoChip sticker

The art of debugging

The last bit of preliminary advice is perhaps the most important point, though it is by far the hardest to express in words. As you build this circuit it is fairly likely that you will make some mistakes. Very likely, things won't work the first time through. Figuring out how to “debug the circuit”, that is, learning how to track down the cause of the problem, is an invaluable skill to develop. But the process of “debugging”, whether it involves an electronic circuit or some other complicated system, is something of an art, and therefore it is hard to give precise rules for how to go about doing it. There are however some basic underlying principles that underlie the art as it pertains to building the LogoChip. Two of the most important are:

- *Test frequently as you build.* A common mistake is that novices make when building a circuit is to build the entire circuit before doing any testing. With this approach, if the fully assembled circuit fails to work properly, the root cause could be in any one of a large number of places. By breaking the assembly process into well defined steps and then trying to devise a test after each step to check if things are working as expected, you will be in a much better position to catch a wiring mistake or a defective component. The step by

Getting Started with LogoChips

step instructions are explicitly intended to introduce you to this habit of frequent testing.

- *Be neat in your wiring.* It is hard to find a wiring error in a rat's nest of tangled wires. Keep your wires as short as possible and make sure when stripping the wires you only expose as much bare wire as needed to insert the wire in the breadboard.

Getting Started with LogoChips

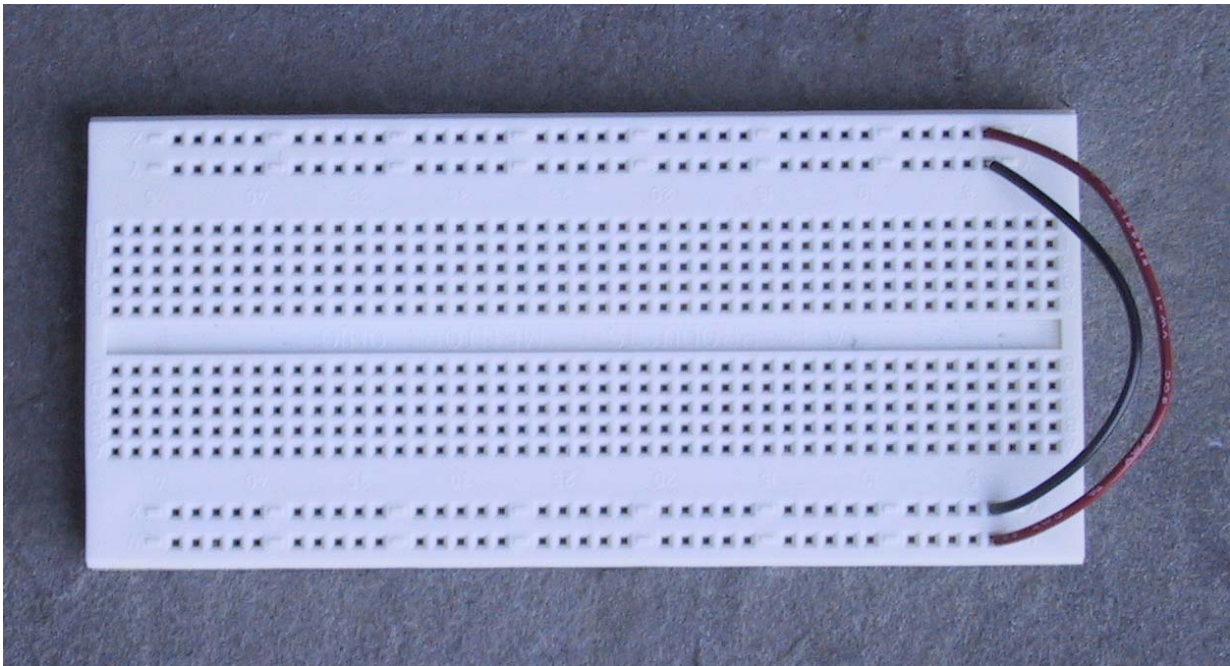
Step by Step Instructions

In the following step-by-step red asterisks (*) denote the new connections that are to be made in a given step.

These instructions are best viewed on your computer screen using pdf document viewing software such as Adobe Reader. The colors of the parts and wires will be readily visible and you can use Reader's "zoom feature" to get a close-up view of the area you are wiring.

Step 1 - Connect the power and ground busses

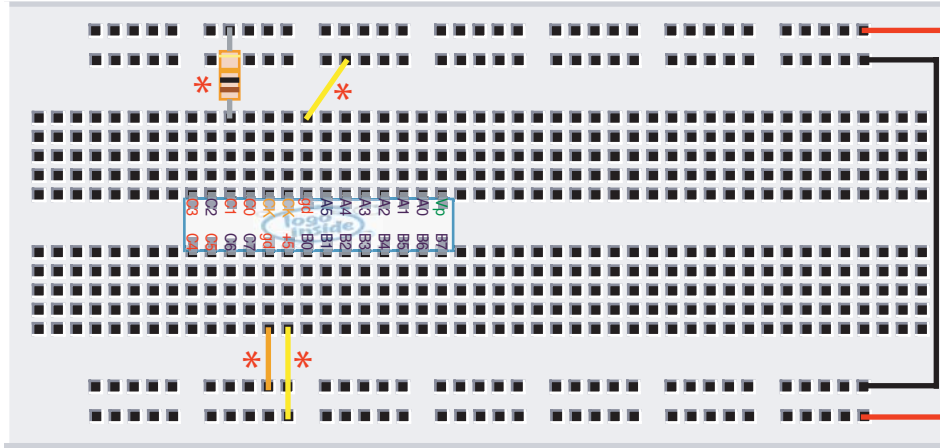
Make power and ground "busses" as shown in the figures below.³ The outer long rows of the breadboard will be connected to the +6V terminal of the battery (the LogoChip will operate over voltage range from +4 V to + 6V) and will form the "power bus" while the inner long rows will be connected to the negative terminal of the battery and will form the "ground bus".



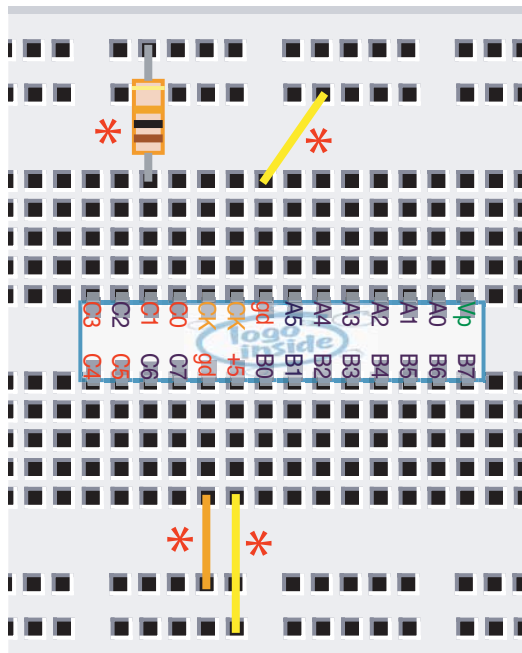
³ The name "bus" is meant to evoke the image of a public bus line that provides riders with frequent access points.

Getting Started with LogoChips

Step 3 - Connect the LogoChip to the power and ground busses⁴



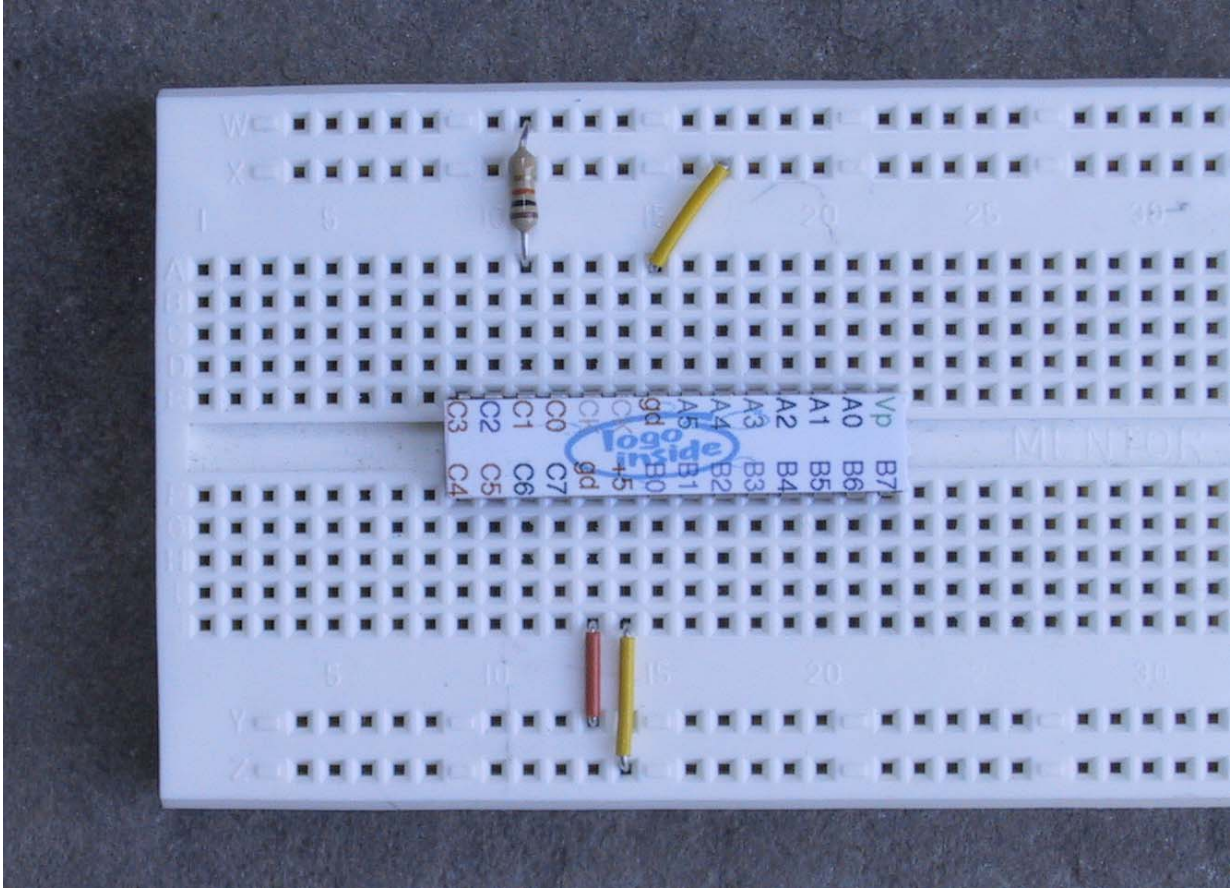
Step 3- Overview



Step 3 - close-up

⁴ The wires used in the remaining steps are color-coded to indicate their length, in the manner of the pre-stripped wire available as Digi-Key part number 923351-ND). These wires are cut in a variety of lengths 0.1" increments and designed to fit perfectly into the digital breadboards. For example, the orange wires are 0.3" long, while the yellow wires are 0.4" long.

Getting Started with LogoChips



Step 3- photo

The resistor used in this step is a 10 k Ω resistor. (The third color band is orange.)

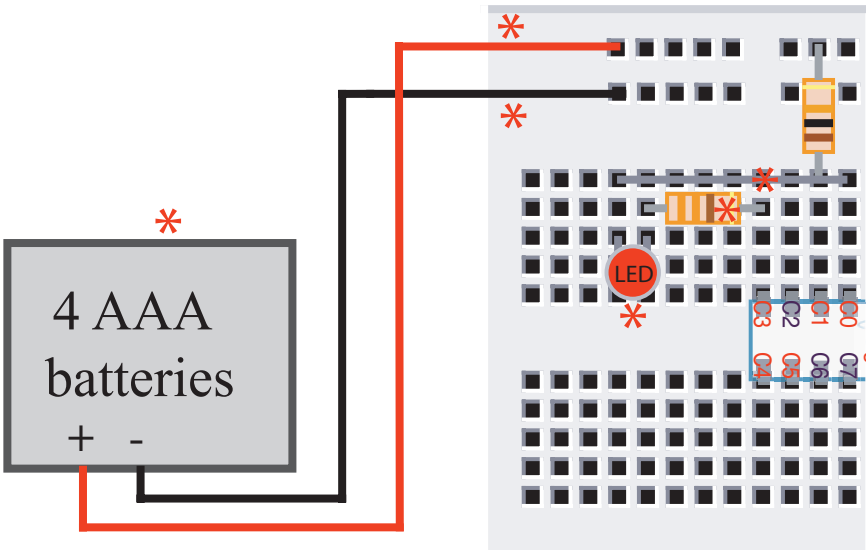
Step 4 - Get a bootflash

When the indicator LED is added to the circuit and power is supplied you should see a “bootflash”: the indicator LED rapidly flash red and green a few times and then remains red. This “bootflash” is the LogoChip’s way of saying “I’m alive!”

Add the red/green indicator LED and the 330 Ω resistor, as shown in the figure below.⁵ The orientation of the LED matters. The LED comes from the

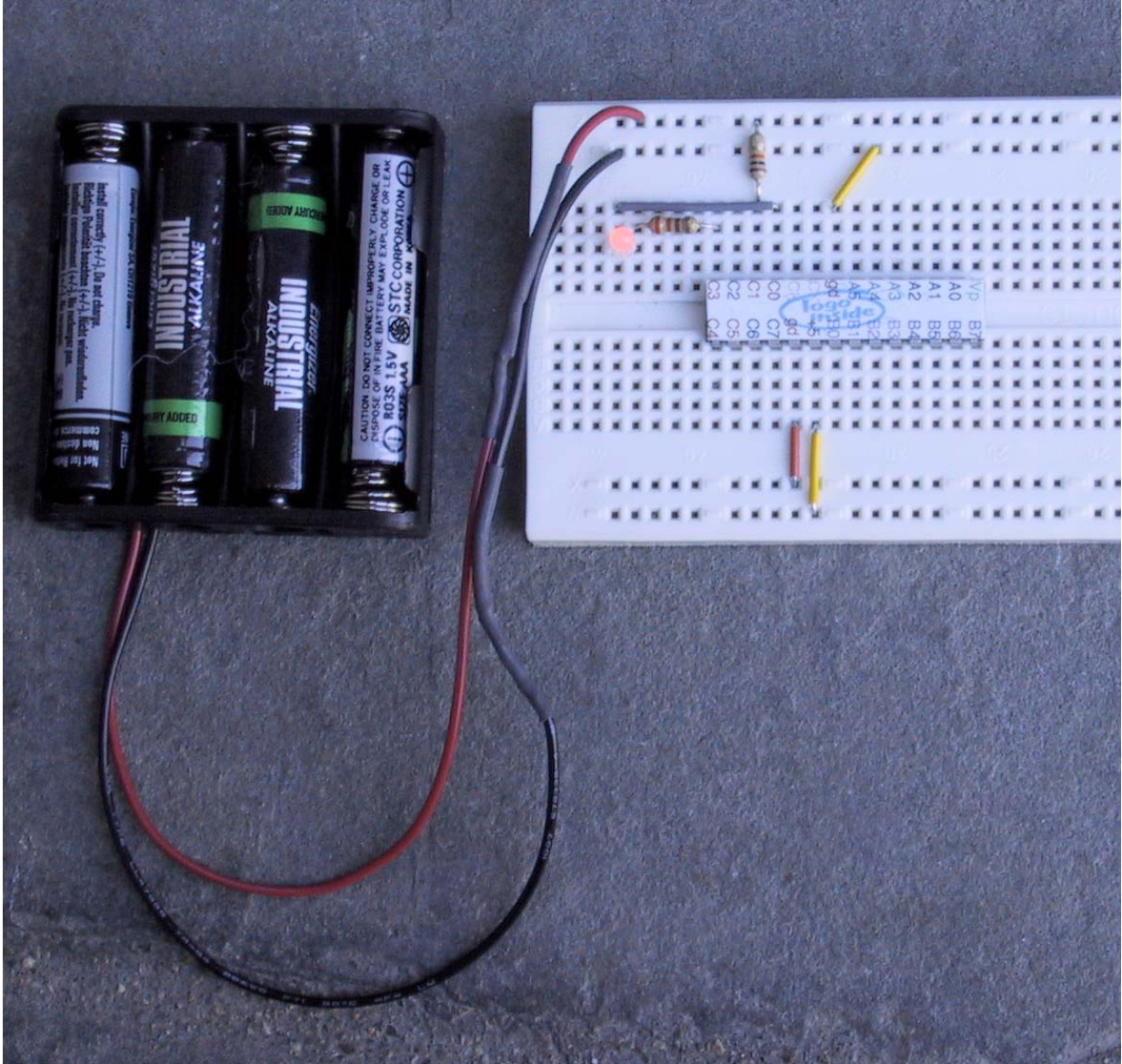
⁵ The value of the resistor used here determines how much current flows through the LED when it is on, and therefore how bright the LED will be. A 330 Ω resistor leads to a current of about 10 mA and a fairly bright LED. By comparison, the LogoChip itself draws only about 4mA. So the indicator LED can easily end up being the dominant power consumer in the circuit. If you’re

Getting Started with LogoChips



Step 4 - close-up

Getting Started with LogoChips



Step 4 - photo

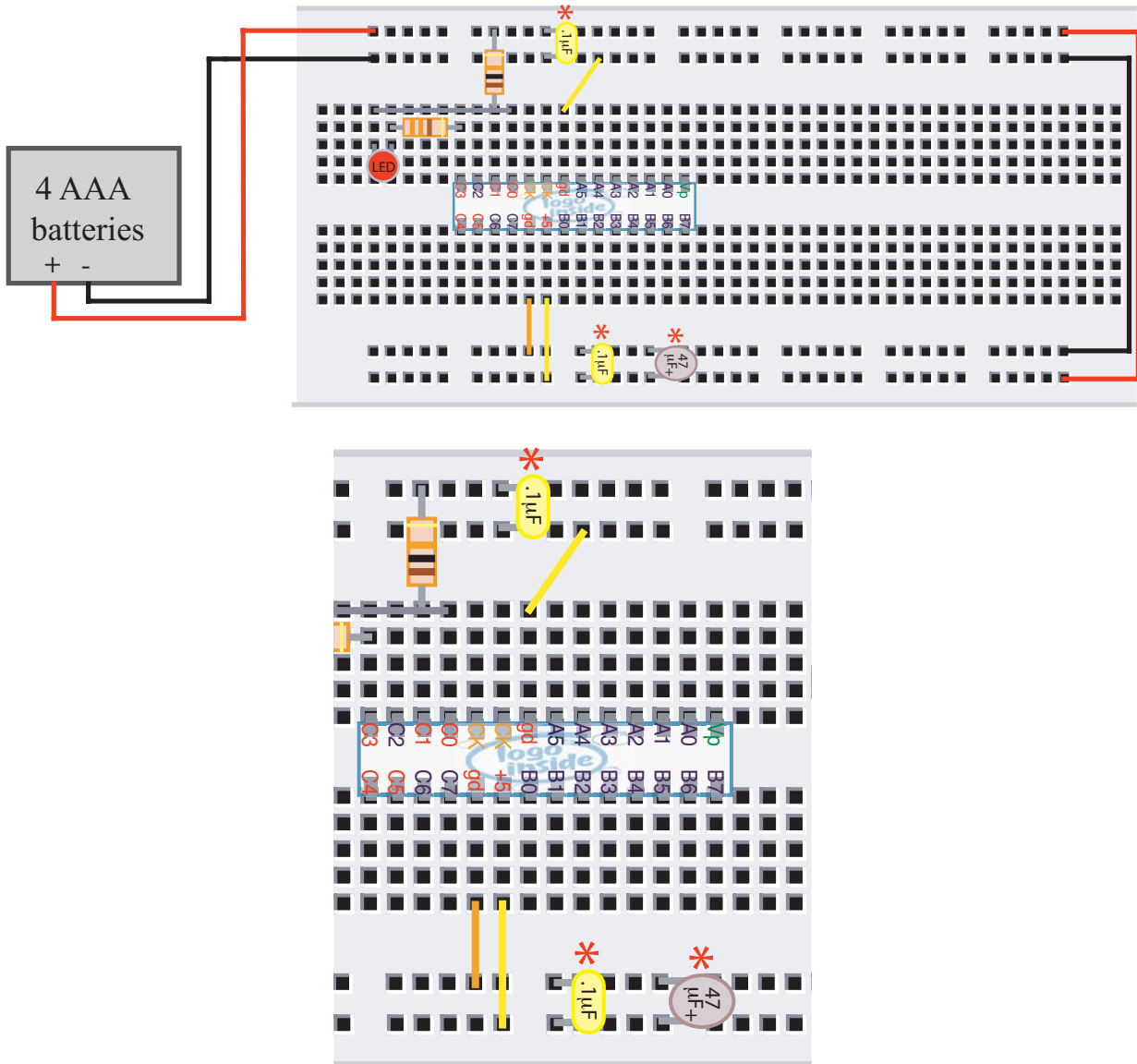
If the LED flashes and stays green, you've almost certainly connected the LED backwards. This is not a problem, simply pull out the LED and plug it in the other way. If, after the bootflash, you see the LED blink completely off and on twice, this is an indication that the batteries are low. Install fresh batteries before proceeding.

If you don't get a bootflash: check with voltmeter to see if there is voltage difference between power and ground busses.

Getting Started with LogoChips

Step 5 - Add the bypass capacitors

Add the 47 μF capacitor and the two 0.1 μF “bypass” capacitors between the outer “power bus” and the inner “ground bus”, as shown in the figure below. **The polarity of the 47 μF capacitor matters and it is important to get this one right.** The little “+” symbol must go into the “power bus”. (The polarity of 0.1 μF capacitors doesn’t matter; you can plug them in either way.)



The purpose of adding these “bypass capacitors” is to “keep the busses quiet”; that is to try to help the battery maintain a constant voltage difference between

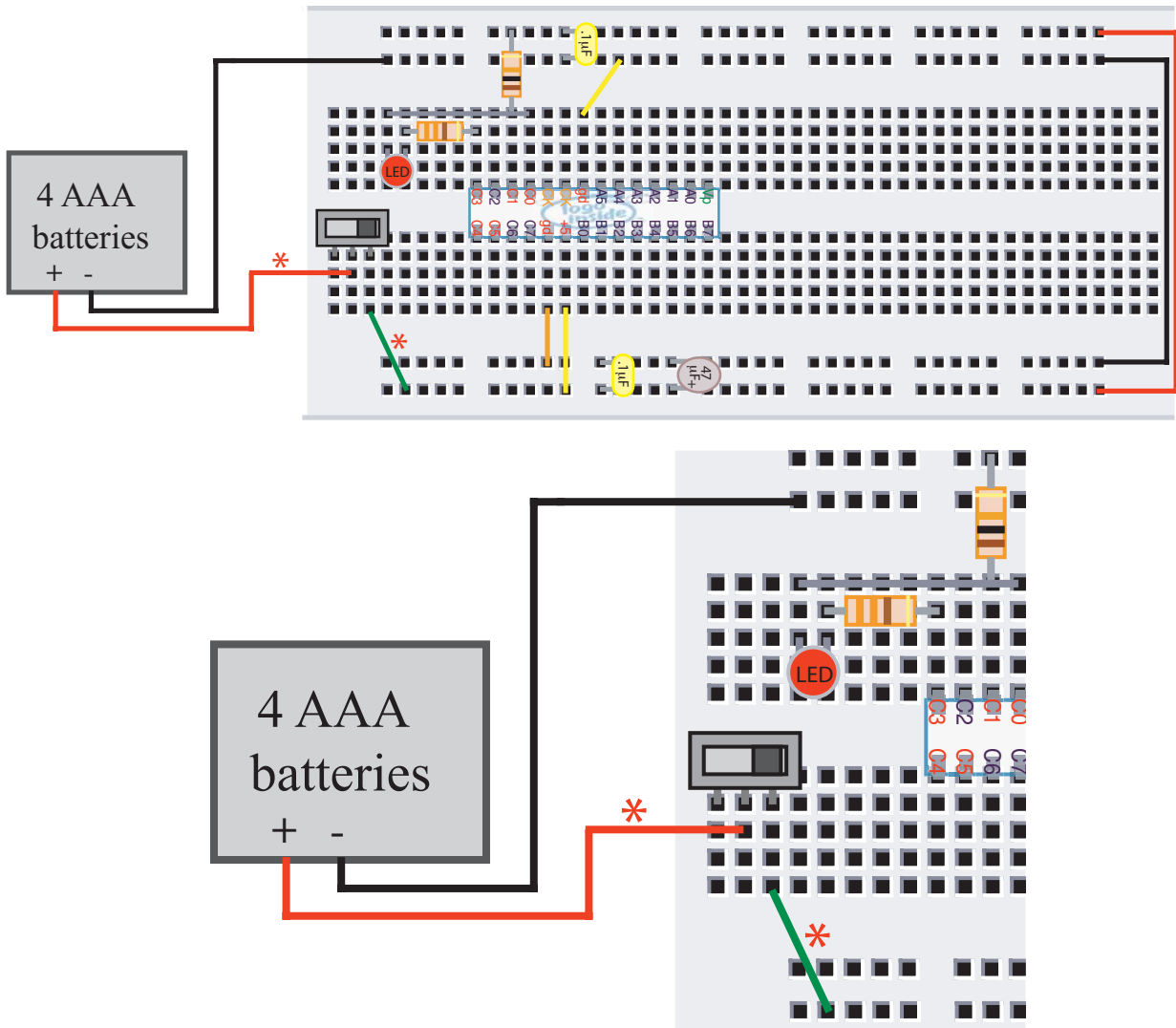
Getting Started with LogoChips

the power and ground busses.

Getting Started with LogoChips

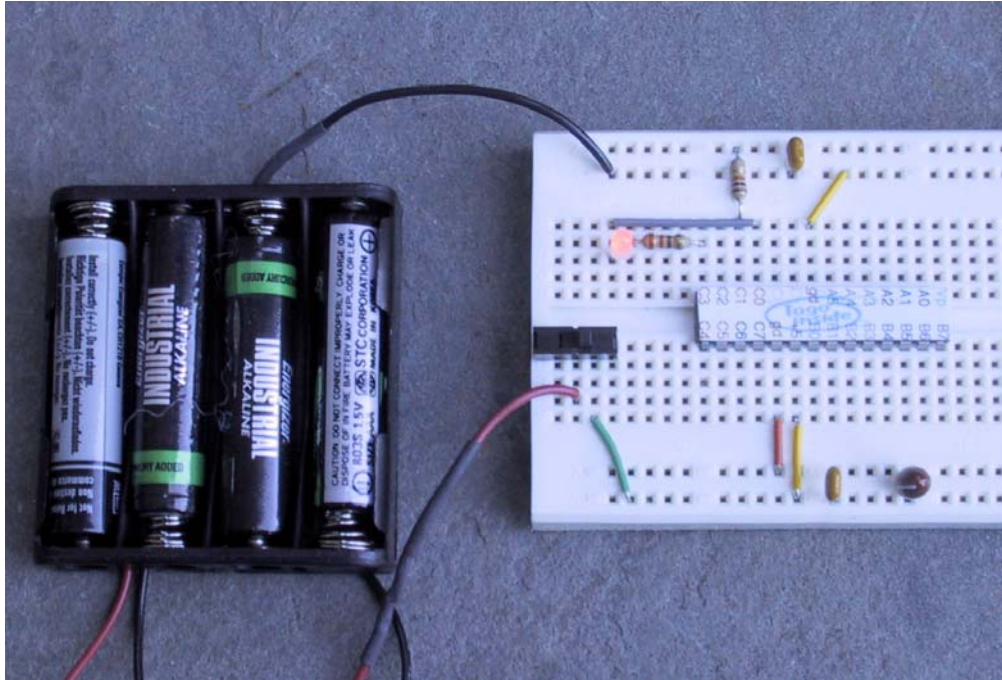
Step 6 - Add a power switch

Now add a power switch so that you don't drain the batteries when your circuit is not in use.



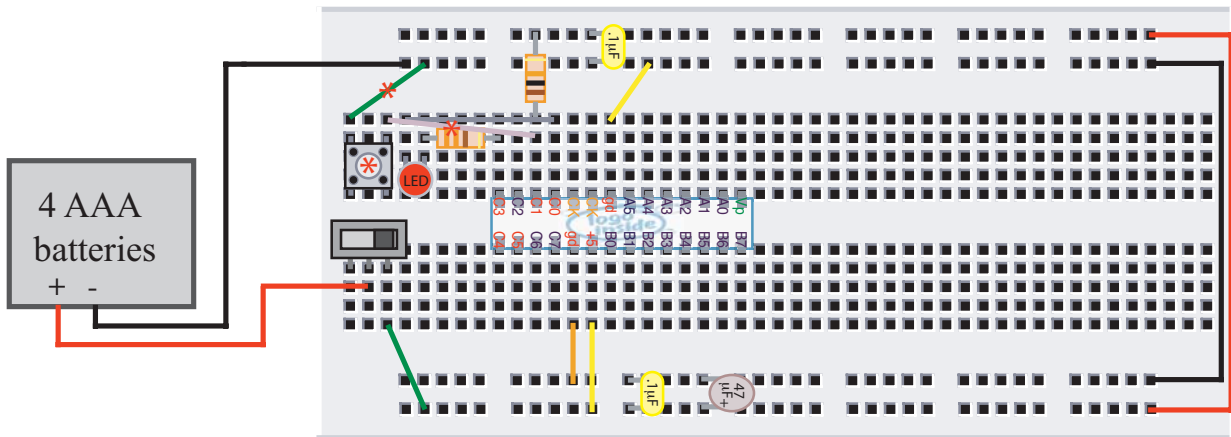
The negative terminal from the battery pack should remain connected directly to the ground bus while the positive terminal from the battery pack should be connected to the middle pin of the slide switch. When the slide switch is in the left-position there should be no connection between the positive terminal of the battery pack and the power bus on the breadboard. Flipping the slide switch to the right-position should make this connection.

Getting Started with LogoChips

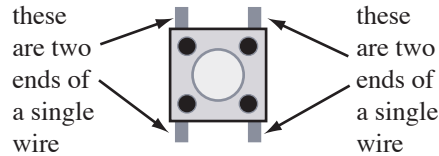


Step 7 - Add the start/stop button

The start/stop button allows you to start and stop LogoChip programs running.



Getting Started with LogoChips



If you have an ohmmeter you can easily perform a test to see if you connected the button circuit properly. Temporarily remove the LogoChip from the breadboard and connect the ohmmeter to measure the resistance between the power and ground “busses”. Pressing the button should change the resistance from an open circuit to 10 k Ω .

Here’s another way to check whether the button is connected correctly. When the LogoChip is first turned on, the LogoChip firmware checks to see if the button is pressed and, if it is, the LogoChip goes into a special “bootstrap mode” and there will not be a “bootflash”. Thus, a properly installed button should inhibit the bootflash if it is held down as the power is turned on.

Talk to your LogoChip

Preliminaries

Desktop Software Installation

1) Download and Unzip the latest version of the desktop LogoChip software from the following URL:

<http://www.wellesley.edu/Physics/Rberg/logochip/distribution>

2)The desktop LogoChip Logo software runs on machines running either the Macintosh OS X or Microsoft Windows operating systems. It is a Java application and you must have a recent version of Java installed on your machine.

Windows users should install Java version 1.4.2 from

<http://java.sun.com/j2se/1.4.2/download.html>

(Select the JRE version.)

Java comes installed with Mac OS X. Make sure you have Java version 1.4.1 or later. (See

Getting Started with LogoChips

`http://www.apple.com/macosx/features/java/`

for upgrades.).

To check the java version on your computer, try the following:

- Open the Command Prompt (Windows) or Terminal (Apple).
- Type `java -version`

If you need to install or upgrade Java on a Windows machine, you can download it from [Sun Microsystems website](http://www.sun.com).

`http://java.sun.com/downloads/ea/index.html`

Mac OS X will also have to install a Java Serial Communications driver, a copy of which is available at:

`http://www.wellesley.edu/Physics/Rberg/logochip/distribution/macjavacomm/JavaCommInstaller.bin`

If you are using a PC that has serial port #1 (COM1) available then the interface board can be connected to COM1 with a standard 9-pin serial cable. If you are using a PC that does not have a serial board or a Macintosh, then you will need a USB to Serial converter cable. We recommend the cable made by Prolific, which is available from

`http://shopnow123.com/store/index.htm`

(Note: if you are using a Windows machine and COM1 is not available, you will need to edit the file called `config.txt` that is included with the LogoChip software. For example, if you want the LogoChip software to communicate via COM2, you should change the `config.txt` file to read

`com-port: COM2`

.)

Connecting to your LogoChip

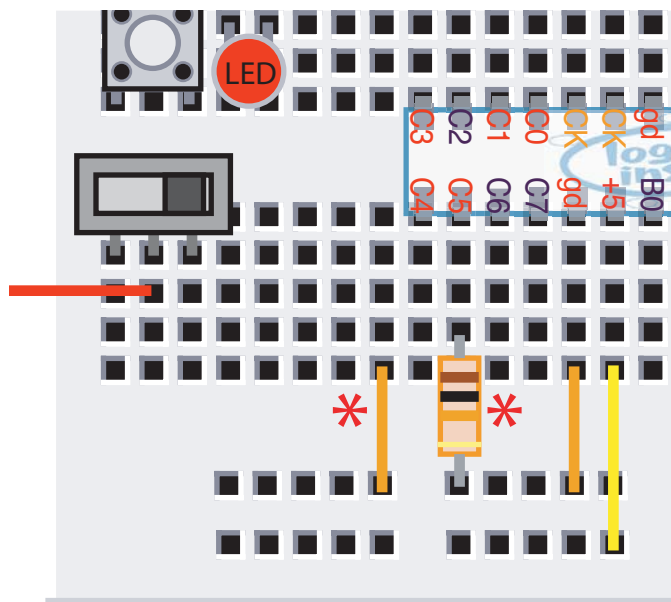
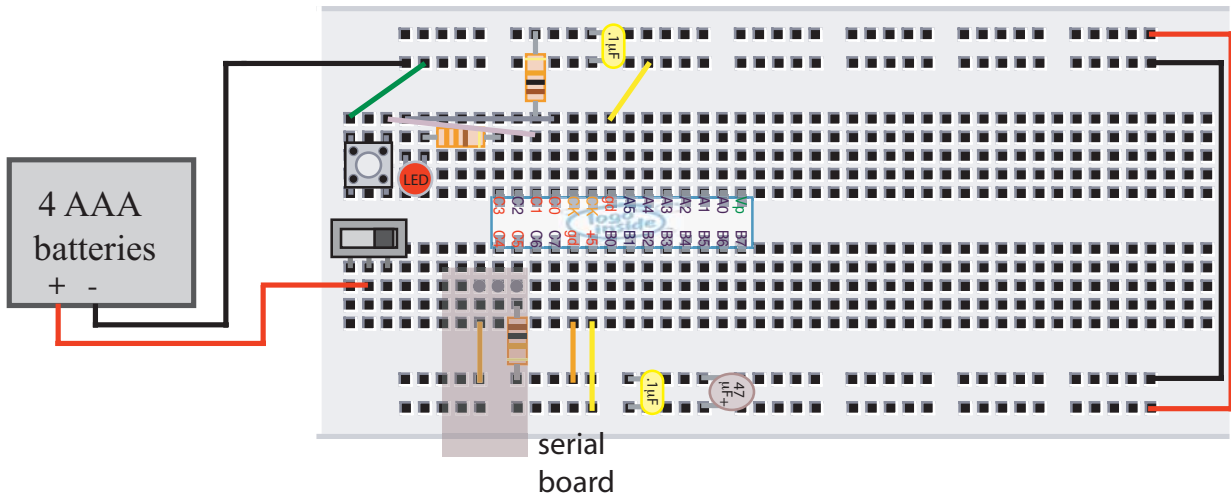
The LogoChip communicates with a host (desktop or laptop) computer via a serial connection. We have designed a simple serial interface board that can be used to connect your LogoChip to the serial port of a host computer. If you don't have access to one of these serial boards, it is still easy to make the

Getting Started with LogoChips

connection if you are willing to do a small amount of soldering.

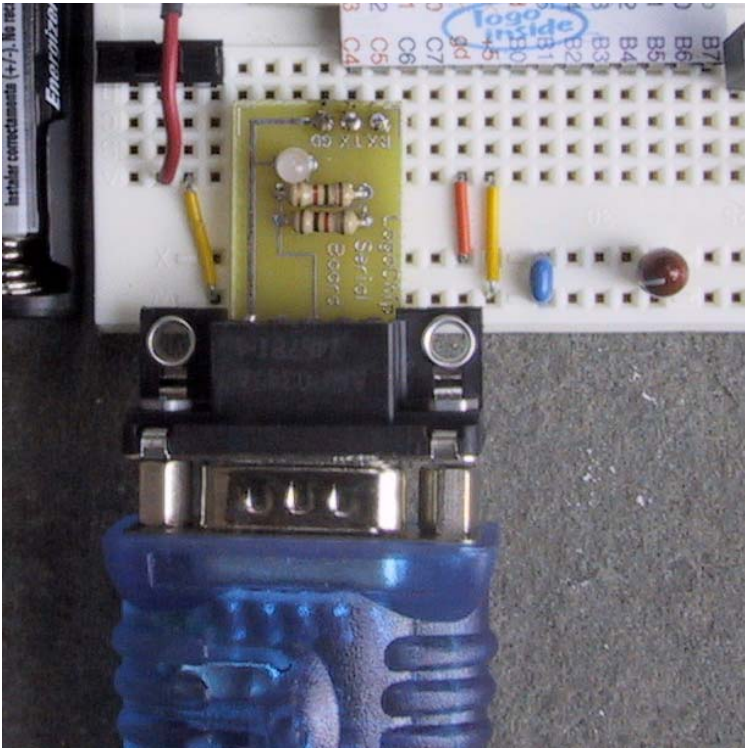
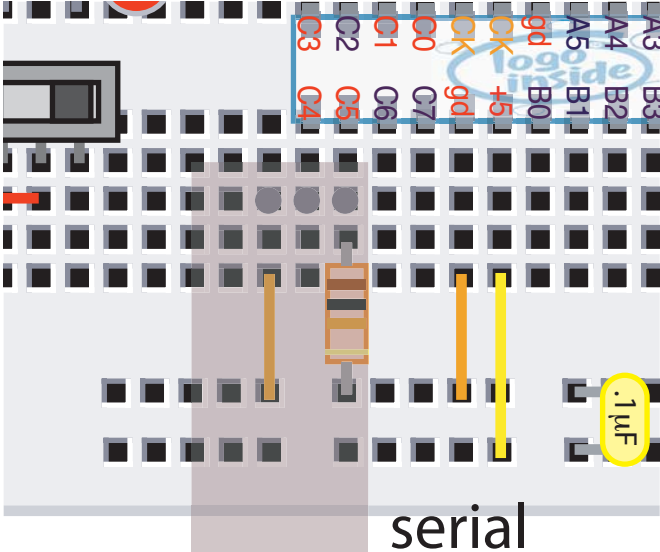
If you have a serial interface board....

Add the 10 k Ω resistor and the ground position in the precise locations shown in the figures below:



and then plug the serial board in as shown below.

Getting Started with LogoChips



Getting Started with LogoChips

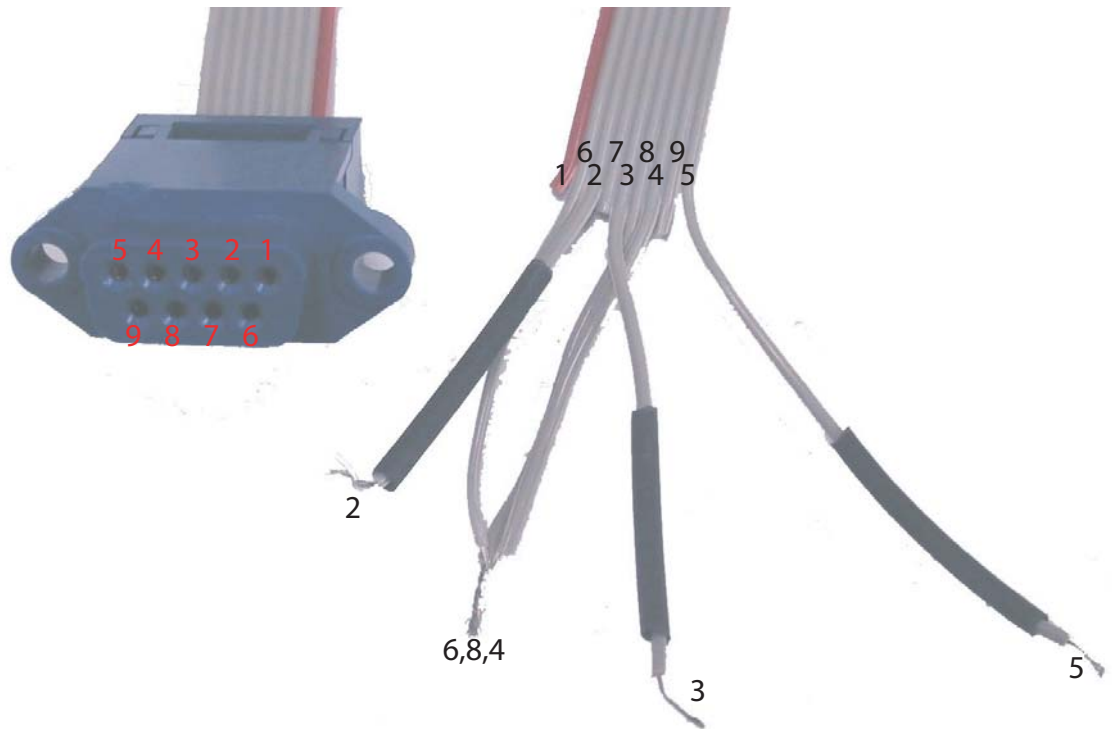
And if you don't have a serial board....

If you don't have a serial interface board it is easy to build your own serial connection using the following steps:

Step 1 - Make a serial cable

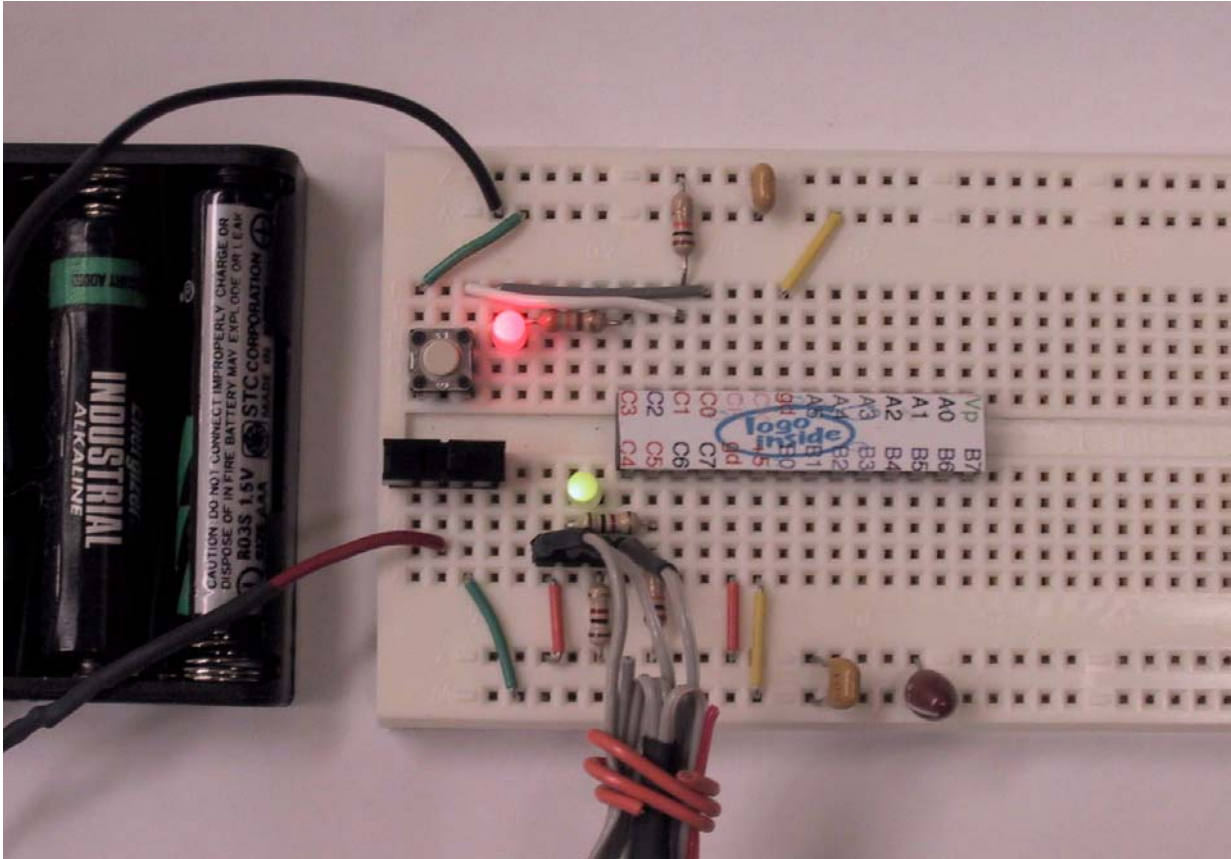
You can make a serial cable using the 9-pin serial connector to ribbon cable part that is available from Digi-Key (part number C7FXT-0906G-ND).

The instructions for making the cable are summarized in the figure below. You'll need to identify and then strip the ends of the wires on the ribbon cable that are connected to pins 2, 3, 4, 5, 6, and 8 on the serial connector. The wires connected to pins 4, 6, and 8 should be soldered together (and then insulated with heat-shrinkable tubing, which is not shown.) Slip a short length of heat-shrinkable tubing onto wires 2, 3, and 5.



Now solder wires number 2, 3, and 5 to a male header (obtained from Digi-Key part number S1032-36-ND), as shown in the figure below. After the soldering is complete, slip the heat-shrinkable tubing over the solder joint and heat with a heat gun to shrink the tubing and insulate the joint.

Getting Started with LogoChips



The LogoChip Logo User Interface

With your serial connect in place and the proper version of Java installed you are now ready to proceed. Open LogoChip Logo by clicking on the file called LogoChip.jar.

The user LogoChip Logo interface that should open is shown below.⁶ With the LogoChip powered up, type the command `flash` in the command-center and hit the <enter> button on the keyboard . The red/green indicator LED should flash just as it did upon powering up.

6

Getting Started with LogoChips



Troubleshooting

If you see any message other than just “Welcome to LogoChip Logo” it means there is a problem. Java-related error messages may mean that the correct version of Java is not installed. (See the previous discussion in the section on Desktop Software Installation.) In OS X, occasionally Java-related problems can arise if the Classic environment is currently running. These problems can be corrected by shutting down the Classic environment and/or restarting the machine.

A “can’t open serial port” message may mean that there is another application using the serial port. For example, another copy of the LogoChip Logo may already be open and have control of the serial port. On a Windows machine you can change the default COM port used by LogoChip Logo with the following steps:

- Open the file config.txt with a simple text editor (i.e. NotePad).
- In the line **com-port: default** Change the word "default " for the name of the COM port where your serial cable is plugged in (i.e. **com-port: COM1**).
- Save the file and perform the above test once again.

If you get a “LogoChip not connected” message when you type `flash`, a number of things could be wrong. Make sure that the LogoChip is powered up. (The start/stop indicator LED should be red.) Check to make sure your serial cable is correctly wired and that there is only one version of LogoChip Logo running. Also check to make sure that no other programs are competing for use of the serial port. You can test the serial connection with the following steps:

- Type `! test-serial` the command-center.

Getting Started with LogoChips

- Before pressing Enter/Return be sure that you are looking at the red/green serial indicator LED. If the serial connection is working properly, the LED should briefly turn red.

Are the batteries low?

If your LogoChip is acting strangely (or not working at all), it could be your batteries are running low. The LogoChip does a battery test when it is first turned on. When the batteries are running low, the “red/green” indicator LED will briefly blink on and off twice immediately following the normal “bootflash”.

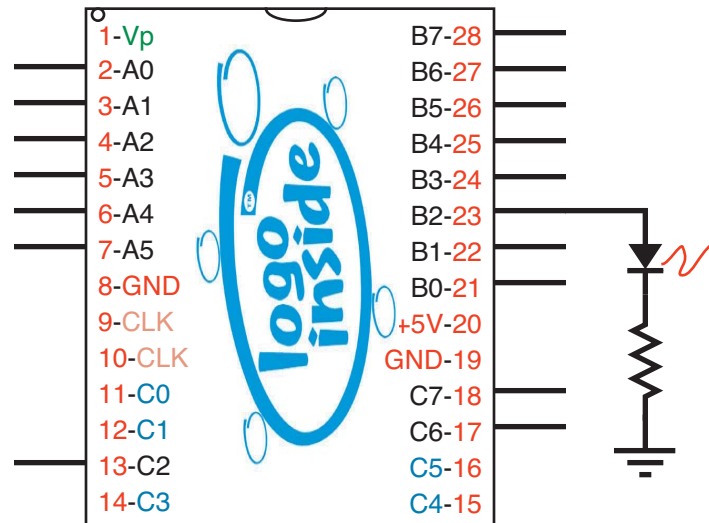
You can write programs for your LogoChip on a desktop or laptop computer and then download these programs to the LogoChip through the computer’s serial port. The programming language we will use is a special version of the Logo programming language called LogoChip Logo that was written by Brian Silverman with help from Bakhtiar Mikhak and Robbie Berg. Details of the LogoChip Logo language are found in the *LogoChip Logo Language Reference*. In addition to `flash` a brand new LogoChip knows how to execute certain “primitive” commands already. For, example, try typing the following commands in the **command-center**.

<code>√ flash wait 20 flash</code>	Flashes, waits 20 tenths of a second, & flashes again. wait grabs control
<code>√ repeat 4 [flash wait 20]</code>	Repeats flash/wait 4 times
<code>√ loop [flash wait 20]</code>	Repeats flash/wait infinitely

Press the LogoChip’s START/STOP button to stop the loop (or any LogoChip Logo program) from running. You can tell if a program is running by looking at the indicator LED. “Green” means a program is running, “red” means the LogoChip is powered up, but no program is running. If the indicator LED is off, it means the LogoChip is not receiving power.

Getting Started with LogoChips

Getting Flashy



Suppose a red LED is connected to pin B2 of the LogoChip as shown above. Here is what the Logo code that would make the LED flash once might look like:

```
to light-on
clearbit 2 portb-ddr           ; turns B2 into an output
                               ; see LogoChip Ins and Outs
                               ; discussion below
setbit 2 portb                 ; makes the level on B2 HIGH
                               (+6V)
end

to light-off
clearbit 2 portb               ;makes pin 2 of portb 0V
end

to blink
light-on
wait 1
light-off
end
```

- Try typing this code in a text file and save the text file in the same folder as the LogoChip Logo software.
- Type the name of the text file in the text box in the lower left hand corner of the LogoChip Logo screen. Click on the <download> button on the screen to download the code to the LogoChip.

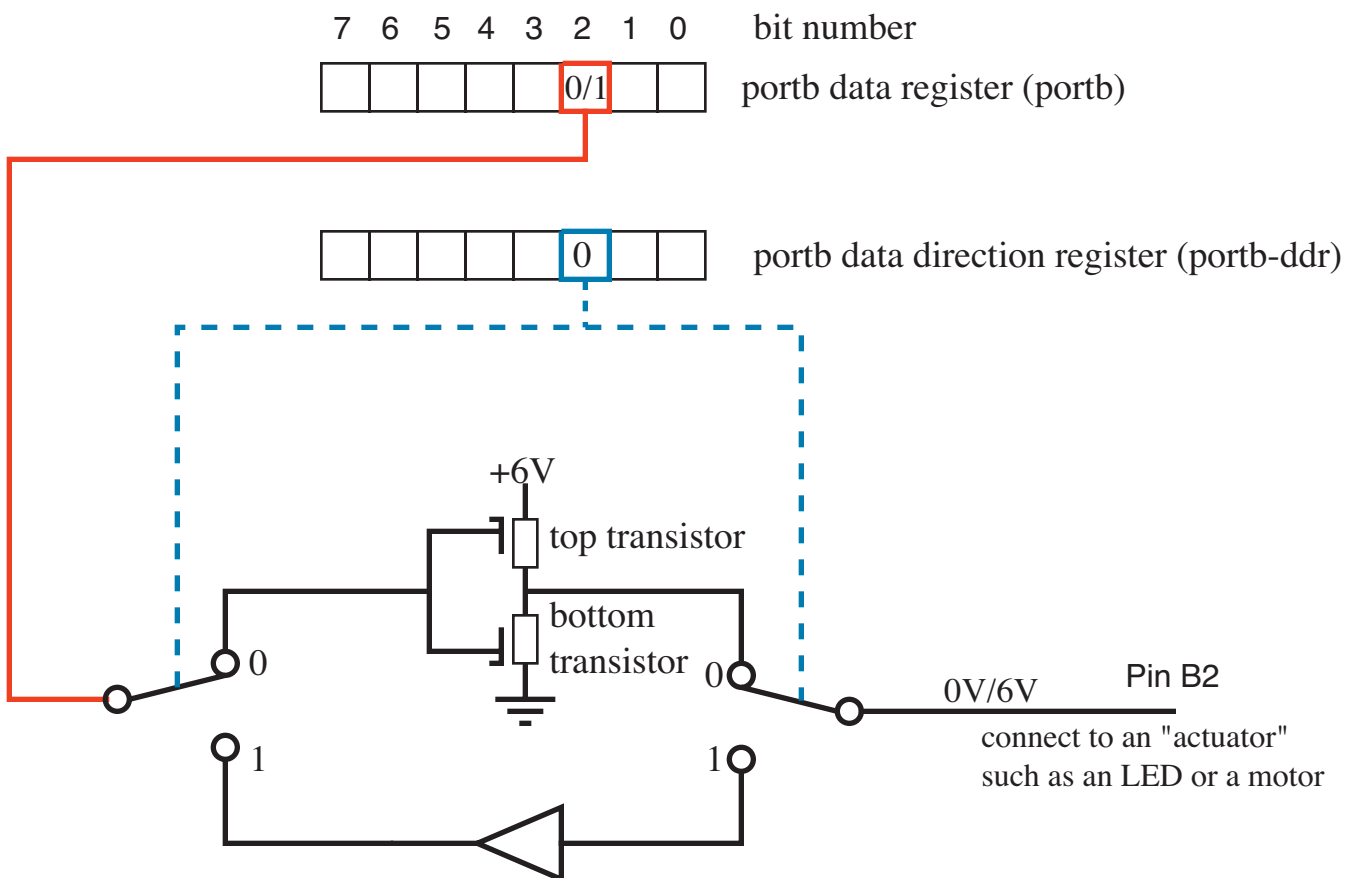
Getting Started with LogoChips

- Try typing the command `light-on` in the command-center. Next try `light-off`. Try `blink`.

LogoChip Ins and Outs

The inside of the LogoChip is organized into entities that we will call **registers**. Each register is a collection of 8 bits (1 byte). Each bit can be either a “0” or a “1”.

There are lots of registers inside the LogoChip, but for now let’s just focus on two of them that are associated with the 8 pins on the LogoChip that we refer to collectively as **portb**. These registers are illustrated in the somewhat cartoonish drawing below: and are called the **portb data register** and the **portb data direction register**. (Similar registers exist for `porta` and `portc`.)



Pin B2 configured as an output

Each of the `portb` pins on the LogoChip can be configured to be either an “input”

Getting Started with LogoChips

or an “output”. The “direction” of each pin is set by a corresponding “data direction bit” the data direction register. If a data direction bit is “set” (made equal to “1”) then the corresponding pin becomes an input. If the data direction bit is “cleared” (made equal to a “0”) the corresponding pin comes an output. We can think of the data direction bit as controlling the positions of the two switches in the above drawing.

A pin that is configured as an output will have a low output resistance and will have a voltage that is either 0V or 6V depending on the contents of the corresponding bit in the data register. Consider the example illustrated above. Suppose bit 2 of the portb data direction register is made to contain a “0” by running the command

```
clearbit 2 portb-ddr
```

This causes the switches to be as shown in the drawing, making pin B2 an **output**. Now, if bit 2 of the portb data register were to contain a “0” then the “bottom transistor” would turn “on” while the “top transistor” would turn “off”, thereby connecting pin B2 to 0 V. If, on the other hand, bit 2 of the portb data register were made to contain a “1” by typing

```
setbit 2 portb
```

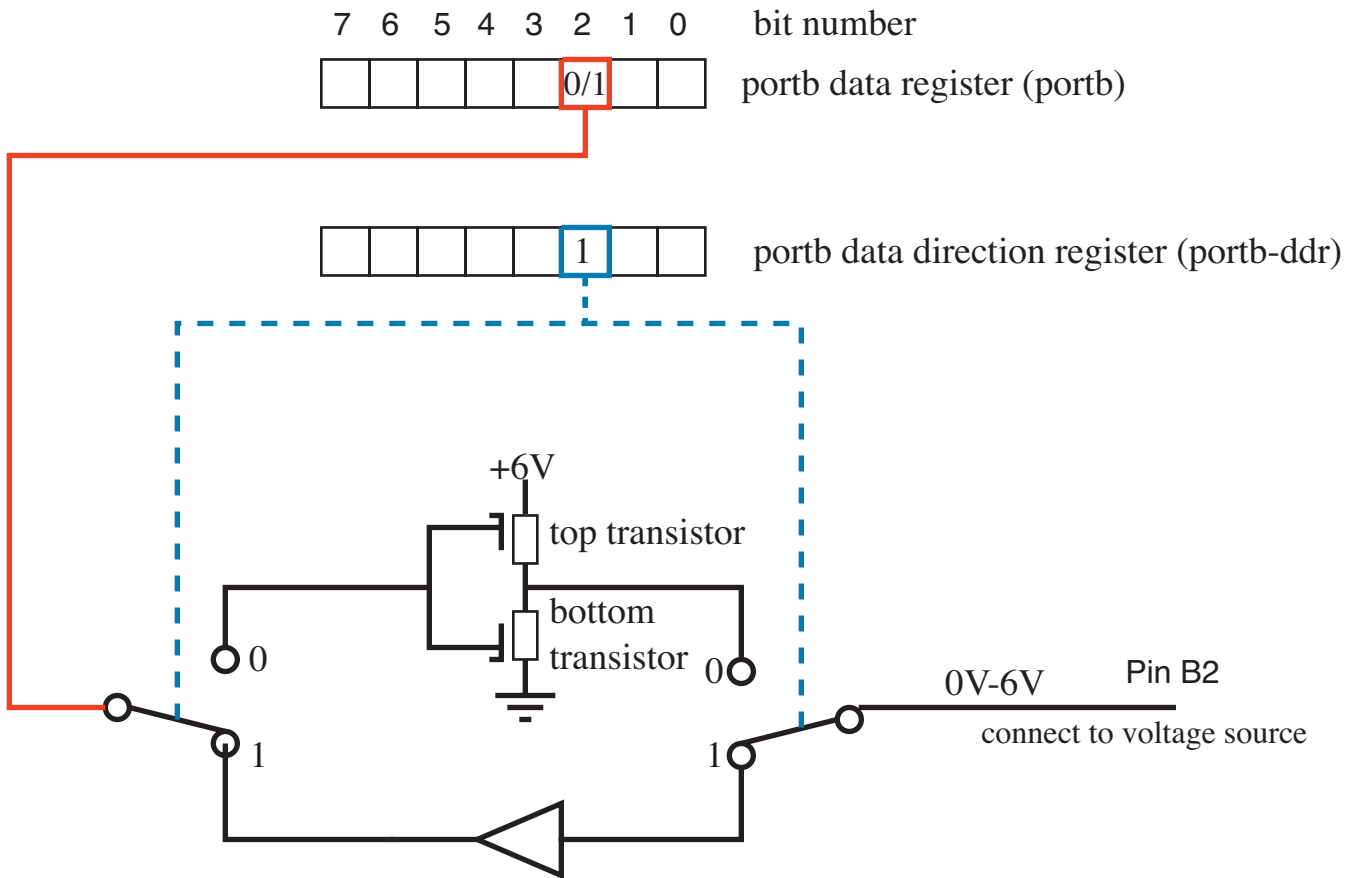
then this would cause the “bottom transistor” would turn “off” while the “top transistor” would turn “on”, thereby connecting pin B2 to 6 V. Thus in the case where a pin is configured as an output the corresponding bit in the data register determines whether the output is HIGH or LOW. When configured as an output a pin can be connected to and used to control various “**actuators**”, such as lights, beepers, or motors.

Now suppose bit 2 of the portb data direction register is “set” (made into a “1”):

```
setbit 2 portb-ddr
```

Pin B2 is now configured as an **input** and the switches are both moved into the positions shown in the figure below.

Getting Started with LogoChips



Pin B2 configured as an input

In this position pin B2 has a high input resistance and can easily be driven “HIGH”(6 V) or “LOW” (0 V) by an external voltage source, such as some sort of “sensor”.

The Autonomous LogoChip

Upon powering up, all of the user accessible pins on the LogoChip are configured as “inputs”. In the first set of exercises below you will be connecting the portb pins to various actuators. Therefore, for the exercises in this document, we would like to always have all of the portb pins configured as outputs. It is convenient therefore to automatically turn all of the portb pins into outputs when you first turn on the LogoChip. This can be done by creating the following special powerup procedure.

```
to powerup
write portb-ddr 0           ; turns all of portb into
outputs
```

Getting Started with LogoChips

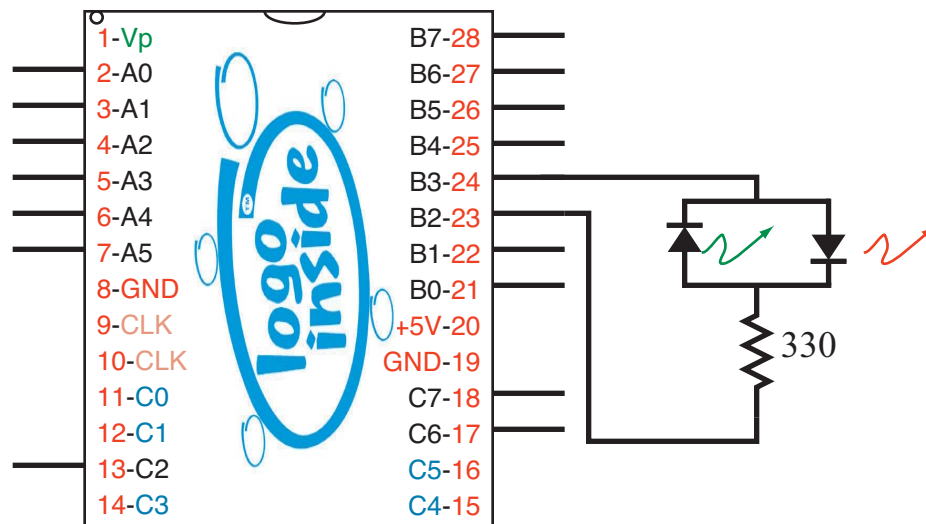
end

When you download a procedure with the special name `powerup` onto the LogoChip then this procedure is run automatically when the LogoChip is turned on. So the `powerup` procedure defined above insures that all of the portb pins are configured as outputs whenever you power up the LogoChip.

Similarly, if you have downloaded a procedure with the name `startup` onto the LogoChip, then, if the LogoChip is idle (LED is red), this procedure is run when the button on the LogoChip breadboard is pressed. This is an important feature. It allows the LogoChip to become autonomous from the desktop computer. You can take the LogoChip away from the computer (and even turn off the power for a long time). You can then run the `startup` procedure by powering up the LogoChip and pressing the `START/STOP` button .

Making Colors

Suppose we now connect a “red/green” LED, between pins B2 and B3 of the LogoChip. (The red/green LED is simply a red LED and a green LED located in close proximity in a common housing and oriented in opposite directions, as indicated in the schematic drawing below.)



We can use the procedures below to make the LED turn red.⁷

⁷ Remember, for these procedures to work, make sure you have configured the portb pins as outputs, by running the command

Getting Started with LogoChips

```
to red
  setbit 2 portb
  clearbit 3 portb
end
```

or green:

```
to green
  clearbit 2 portb
  setbit 3 portb
end
```

or, by rapidly switching between red and green states, we can make it appear yellow:

```
to yellow
  loop [red green]
end
```

or some other color:

```
to other-color
  loop [red red green]
end
```

Let's Get Moving

Now trying adding a red LEGO micro-motor between B4 and B5 and playing with the following new procedures.

```
to on-thisway
  setbit 4 portb
  clearbit 5 portb
end
```

```
to on-thatway
  clearbit 4 portb
  setbit 5 portb
end
```

```
to off
  clearbit 5 portb
  clearbit 4 portb
```

```
write portb-ddr 0
```

as described above.

Getting Started with LogoChips

```
end

to rd
togglebit 4 portb
togglebit 5 portb
end

to on-thisway-for :n
on-thisway
wait :n
off
end

to on-thatway-for :n
on-thatway
wait :n
off
end
```

Once you've compiled these procedures, try typing the following commands in the command-center:

on-thisway	Turns on motor in the “thisway” direction
on-thatway direction	Turns on motor in the “thatway” direction
rd	Reverses direction of a spinning motor
off	Turns off the motor
on-thisway-for 20	Turns on the motor in the “thisway” direction for 2 seconds
repeat 4 [on-thisway-for 10 wait 10]	Turns motor on and off 4 times
repeat 4 [on-thisway-for 10 on-thatway-for 10]	Motor moves back and forth 4 times

Making Music

Now connect a piezoelectric beeper between pin B6 and ground and add the following new procedures

```
to click-on
```

Getting Started with LogoChips

```
setbit 6 portb
end

to click-off
clearbit 6 portb
end

to beep
repeat 100 [click-on delay 50 click-off delay 50]
end

to delay :n                                ; a variable length short delay
repeat :n [no-op]                          ; no-op does nothing except take
                                           ; about 10 ; micro-seconds to execute

end
```

Play with parameters in `beep` until you get a beep of your liking.

Try writing your own `note` command of the form:

```
to note :pitch :duration
...
end
```

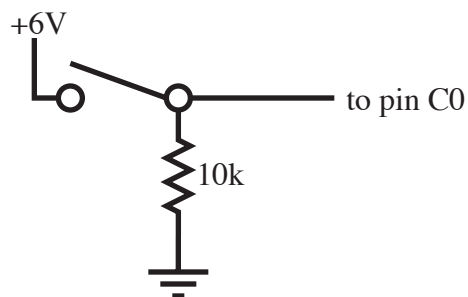
That is, `note` has two “input parameters”. The first parameter determines the pitch of the note and the second parameter determines the duration of the note.

Can you compose a melody?

Getting a Sense of the World

Digital Sensors

Connect a pushbutton switch to pin C0, as shown in the figure below:



a "touch sensor"

Getting Started with LogoChips

Add the following procedure to your procedures file.

```
to touch?  
  output testbit 0 portc  
end
```

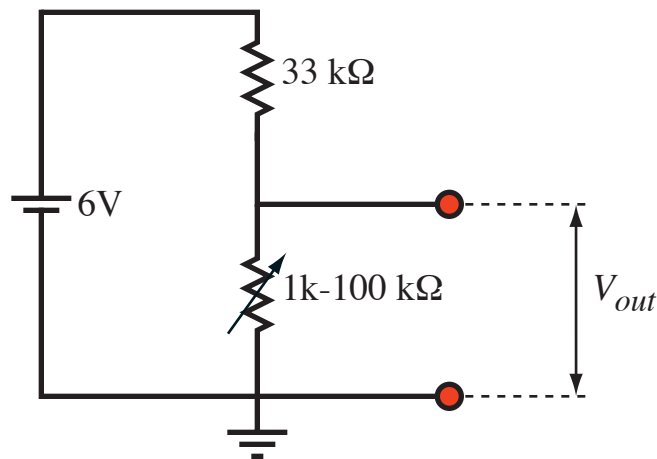
Try typing the following command in the command-center:

```
waituntil [touch?] beep
```

You've built your first digital sensor!

Analog Sensors

Build the following simple voltage divider on your breadboard. Try using a photocell or perhaps a thermistor as the variable resistor. Then connect the output of the voltage divider to pin A0.



Try typing the following commands in the command-center:

```
print read-ad 0
```

The `read-ad` primitive reports the value of the 10-bit analog to digital converter associated with pin A0. The `print` primitive causes the result to be displayed in the text box on the right side of the LogoChip Logo screen.

```
loop [print read-ad 0 wait 5]  
waituntil [(read-ad 0) < 500] beep
```

You've built your first analog sensor!

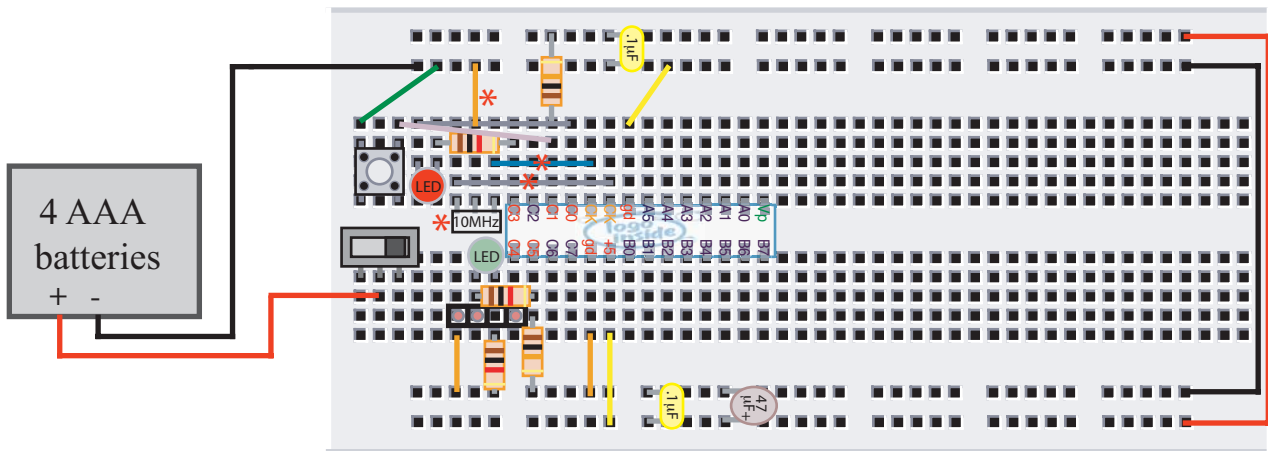
Getting Started with LogoChips

Speeding Things Up

The speed at which a piece of LogoChip code runs depends on the rate of the clock signal. The LogoChip you built above uses an internal oscillator to generate a 8 MHz clock signal. To get a sense of how fast the LogoChip code is running we can use the following simple benchmark

```
resett repeat 10000 [no-op] print timer
```

If you run this line from the command should find that the number 433 (or something close to it) is printed in the command center. This means that it takes the LogoChip 433 milliseconds to go through the repeat loop 10,000 times, executing a “no-op” each time. Thus the LogoChip as it stands can execute more than 20,000 simple Logo instructions per second. For most applications this is more than fast enough. But it is easy to get the LogoChip to run five times faster by adding an external 10 MHz resonator (Digi-Key part # PX1000MC-ND) to the circuit, as shown in the figures below.⁸ The polarity of the resonator does not matter, but it is important to keep the wires connecting the resonator to the LogoChip as short as possible.



⁸ The LogoChip uses the external 10 MHz circuit along with some internal circuitry to generate a 40 MHz clock signal, leading to a factor of five increase in speed compared to when the internal oscillator is used.

Getting Started with LogoChips

Appendix: Clone Your Own LogoChip

Once you have one LogoChip you can easily create more through a “cloning” procedure in which the LogoChip firmware copies itself onto a blank PIC18F2320 microcontroller.

To clone a LogoChip, simply add the circuitry shown in red on the schematic below to your existing LogoChip. The new circuitry consists of a “pnp” transistor such as a “3906” (*e.g.* Digi-Key part# 2N3906D26ZCT-ND), a second 6 volt battery pack, and an LED (a bi-color LED like the one used for the LogoChip indicator light will do) and a 1k Ω and a 10 k Ω resistor. When the new circuitry has been added, download the program called clone.txt (which is included along with the LogoChip software) onto the existing LogoChip.

Clone.txt contains a `startup` procedure which causes all of the LogoChip program memory (including the clone.txt Logo program!) to be copied to the blank PIC18F2320 microcontroller. Simply pressing the START button initiates the process, which takes about 4 minutes (assuming the LogoChip is running at 8 MHz). If all is well, the LED included in the new circuitry should turn on during the cloning procedure and the voltage on the pin 1 of the blank PIC18F2320 should rise to about 11 V. As soon as the LogoChip’s green run light turns back to red, you’re done. You can test to see if the cloning was successful by replacing the old LogoChip with the new one and making sure it works properly.

Getting Started with LogoChips

