Getting Started with LogoChips

Electronics for Everyone ................................................................. 1
LogoChip: A new electronic construction kit ........................................ 3
Build Your Own LogoChip ................................................................ 3
  LogoChip Parts List ......................................................................... 4
  Overview .......................................................................................... 4
  Step by Step Instructions .................................................................. 9
Talk to your LogoChip ...................................................................... 15
  Preliminaries .................................................................................... 15
  The LogoChip Logo User Interface .................................................. 18
Getting Flashy ................................................................................... 19
LogoChip Ins and Outs ..................................................................... 20
The Autonomous LogoChip .............................................................. 23
Making Colors ................................................................................... 24
Let’s Get Moving ............................................................................... 25
Making Music ..................................................................................... 26
Getting a Sense of the World ............................................................ 27
  Digital Sensors ................................................................................ 27
  Analog Sensors ............................................................................... 28
Appendix: Clone Your Own LogoChip .............................................. 29

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,

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 lead 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 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

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

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.

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
Getting Started with LogoChips

that is both powerful and easy to use.

You can build the LogoChip on a small breadboard following the instructions below. Use a 6V battery pack (4 AAA batteries) to power your device.

**LogoChip Parts List**

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Part</th>
<th>Manufacturer</th>
<th>Digikey Part Number</th>
<th>Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PIC microcontroller</td>
<td>Microchip</td>
<td>PIC18F252-I/SP-ND</td>
<td>5.03</td>
</tr>
<tr>
<td>1</td>
<td>10 MHz ceramic resonator</td>
<td>Panasonic</td>
<td>PX1000MC-ND</td>
<td>1.14</td>
</tr>
<tr>
<td>1</td>
<td>solderless breadboard</td>
<td>3M</td>
<td>923253-ND</td>
<td>15.44</td>
</tr>
<tr>
<td>1</td>
<td>pushbutton</td>
<td>Omron</td>
<td>SW400-ND</td>
<td>0.20</td>
</tr>
<tr>
<td>1</td>
<td>47 microfarad tantalum capacitor</td>
<td>Panasonic</td>
<td>P2017-ND</td>
<td>1.07</td>
</tr>
<tr>
<td>2</td>
<td>0.1 microfarad capacitors</td>
<td>Kemet</td>
<td>399-2127-ND</td>
<td>0.24</td>
</tr>
<tr>
<td>1</td>
<td>bi-color LED</td>
<td>Lite-on</td>
<td>160-1058-ND</td>
<td>0.30</td>
</tr>
<tr>
<td>1</td>
<td>1k resistor</td>
<td>Yageo</td>
<td>1.0KQBK-ND</td>
<td>0.02</td>
</tr>
<tr>
<td>1</td>
<td>10k resistor</td>
<td>Yageo</td>
<td>10KQBK-ND</td>
<td>0.02</td>
</tr>
<tr>
<td>1</td>
<td>330 ohm resistor</td>
<td>Yageo</td>
<td>330QBK-ND</td>
<td>0.02</td>
</tr>
<tr>
<td>1</td>
<td>power switch</td>
<td>E-Switch</td>
<td>EG1903</td>
<td>0.64</td>
</tr>
<tr>
<td>1</td>
<td>&quot;AAA&quot; battery holder (4 cells)</td>
<td>Keystone</td>
<td>2482K-ND</td>
<td>1.01</td>
</tr>
<tr>
<td>4</td>
<td>&quot;AAA&quot; batteries</td>
<td>Panasonic</td>
<td>P146-ND</td>
<td>1.52</td>
</tr>
</tbody>
</table>

The LogoChip consists of a PIC18F252 microcontroller (made by Microchip) that has had a Logo "virtual machine" installed on it. If you need help in obtaining a LogoChip with the firmware installed, please contact Robbie Berg (rberg@wellesley.edu)

**Overview**

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

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

A blank breadboard

The red shaded regions indicate which holes are connected on this particular breadboard (Digikey model 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 (which are intended to serve power and ground “buses”. If you are uncertain, the best way to check is to use an ohmmeter to see what is connected to what.

Breadboard connections
The figure above shows a **schematic drawing** of the LogoChip circuit.
Getting Started with LogoChips

**Step by Step Instructions**

First, some general advice:

- It is important to be neat and especially important to keep the wires connecting the resonator to the CLK inputs on the microcontroller as short as possible.

- Make use of the columns on both sides of the breadboard to make power and ground “buses”.

- Do not connect the power until the last step, when you’re pretty sure the wiring is correct.

**Step 1)** 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 PIC18F252 microcontroller. **It is critical that the sticker be attached with the proper orientation. The top of the sticker (the end with the pins labeled CLR and B7) must be attached to the end of the microcontroller that has a notch in it.**
Step 2) Insert the LogoChip into the breadboard and make the power and ground connections, as shown in the figure below. In the last step below the outer long rows of the breadboard will be connected to the +6V terminal of the battery 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”.

Step 3) Connect the 10 MHz resonator, as shown in the figure below. 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. (Note that the “new” connections in the figure below are shown in color, while the “old” connections are gray.)
Step 3

Step 4) Add the START/STOP button and the 10 kΩ resistor, as shown in the figure below.

Step 4

A common mistake is to get the orientation of the start/stop button wrong by 90 degrees. The figure below shows that the pushbutton consists of two vertically oriented wires. The wires are brought into electrical contact whenever the button is pressed. This construction implies that the proper orientation for this part is the one in which the two ends of a given wire are plugged into breadboard holes that are internally connected. This orientation insures that the two distinct wires that make up the pushbutton will remain unconnected until the pushbutton is pressed.
Getting Started with LogoChips

(If you have an ohmmeter you can easily perform a test to see if you connected the button circuit properly. Connect the ohmmeter to measure the resistance between the power and ground “busses”. Pressing the button should change to resistance from about 10 kΩ to 0 kΩ.)

Step 5) Add the red/green LED and the 330 Ω resistor, as shown in the figure below. The orientation of the LED matters. The LED comes from the manufacturer with one lead shorter than the other. The shorter lead should be the one connected directly to pin C4. (If you happen get the LED orientation backwards, it’s not a big problem. Nothing bad happens and it will be easy to correct shortly.)

Step 6) 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”.

page 12
Step 6

Step 7) Add the power switch and the 4 AAA battery pack, as shown in the figure below. **It is once again critical to get the polarity correct here.** (As a precaution, don’t install the batteries into the battery pack until you are sure you have the wiring correct.) The negative terminal from the battery pack should be 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. Before installing the batteries, check with an ohmmeter to make sure that there is not a power to ground short circuit: there should be at least a few kΩ of resistance between the power and ground busses. When the slide switch is in the left-position there should be infinite resistance 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 resistance become zero.

With the slide switch in the left-position, install the 4 AAA batteries. Now turn the power on by flipping the slide switch to the right-position. If everything is correct, you should see the indicator LED rapidly flash red and green a few times and then stay red. If it flashes and stays green, you’ve probably connected the LED backwards.) This “bootflash” is the LogoChip’s way of saying “I’m alive!”
Step 7
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

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

for upgrades.). 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

Serial Connection
The LogoChip communicates the desktop computer via a serial connection. The connection consists of a serial interface board that plugs into the LogoChip’s “serial communication pins” on the LogoChip. (shown in red on the sticker) and a serial cable that connects the interface board back to the desktop computer, as shown in the figures below. Make sure to get the orientation of the interface board right!! You may damage the serial interface it if you connect it improperly!!
If you are using a PC that has serial port #1 (COM 1) available then the interface board can be connected to COM 1 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
Getting Started with LogoChips

converter cable. We recommend the cable made by Prolific, which is available from

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

If you don’t have a serial interface board it is easy to build your own using a Maxim 233 chip, as shown in the following schematic:
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 called LogoChip.jar.

The user LogoChip Logo interface that should open is shown below. 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.

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`.

- `√ flash wait 20 flash` Flashes, waits 20 tenths of a second, & flashes again.
- `√ repeat 4 [flash wait 20]` Repeats flash/wait 4 times
- `√ loop [flash wait 20]` Repeats flash/wait infinitely

If you see any message other than just “Welcome to LogoChip Logo” it means there is a problem. 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. Java-related messages may mean that the correct version of Java is not installed. (In OS X, occasionally Java-related problems appear to be corrected by shutting down other applications and/or restarting the machine.)
Getting Started with LogoChips

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 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
  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
```
Getting Started with LogoChips

- 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.

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

\begin{verbatim}
clearbit 2 portb-ddr
\end{verbatim}

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.
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
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. 3

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

write portb-ddr 0

as described above.
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
end
```

```
to rd
  togglebit 4 portb
  togglebit 5 portb
end
```
Getting Started with LogoChips

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

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

One 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` 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
  setbit 6 portb
end

to click-off
  clearbit 6 portb
end
```
Getting Started with LogoChips

```lisp
(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:

```lisp
(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 diagram of a "touch sensor" connected to pin C0](image)

Add the following procedure to your procedures file.

```lisp
(to touch?
  output testbit 0 portc
end)
Getting Started with LogoChips

Try typing the following command in the command-center:

```plaintext
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.

![Voltage Divider Diagram](image)

Try typing the following commands in the command-center:

```plaintext
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.

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

You’ve built your first analog sensor!

page 28
Appendix: Clone Your Own LogoChip

Once you have one LogoChip you can easily create more through a “cloning” procedure in which the LogoChip firmware couples itself onto a blank PIC18F252 microcontroller. 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 1kΩ and a 10 kΩ 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 PIC18F252 microcontroller. Simply pressing the START button initiates the process, which takes about a minute. (If all is well, the LED included in the new circuitry should turn on during the cloning procedure.) 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.