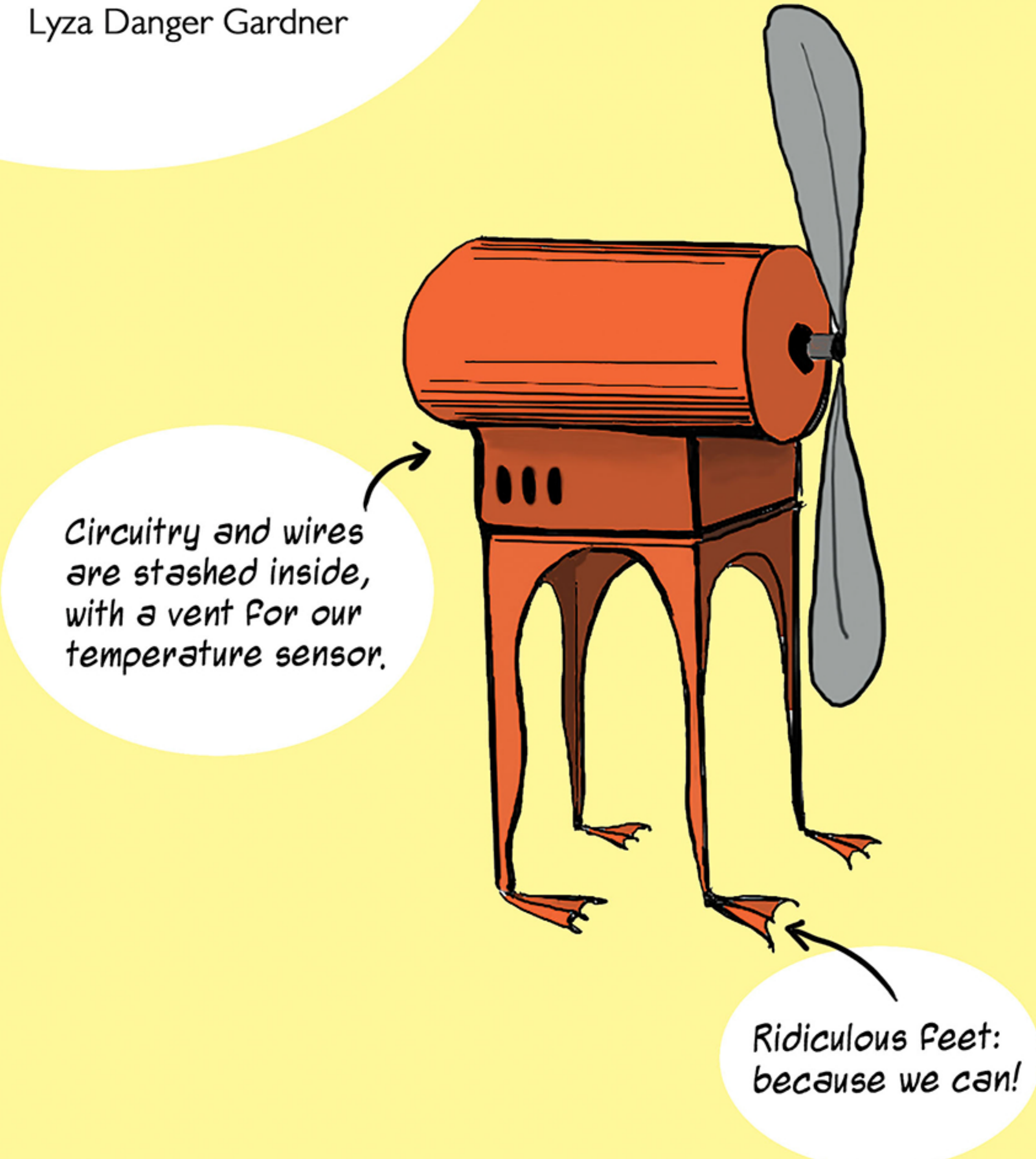
JAVASCRIPT ON THINGS

Hacking hardware for web developers

Lyza Danger Gardner



manning

SAMPLE CHAPTER

JavaScript on Things

by Lyza Danger Gardner

Chapter 3

brief contents

3 How to build circuits

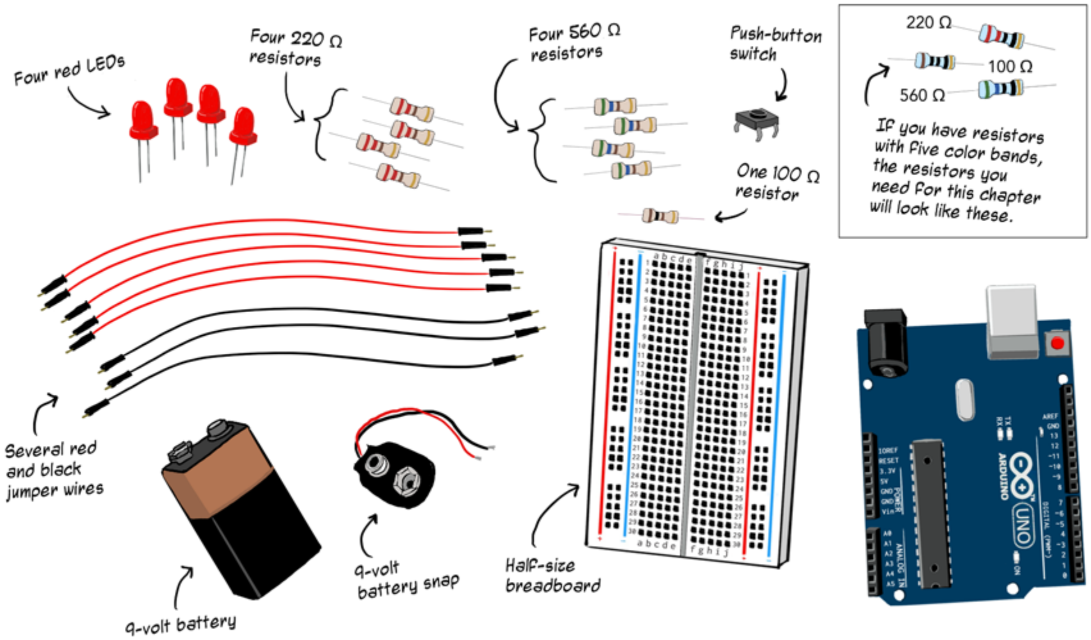
This chapter covers

- Using Ohm's law to manipulate voltage, current, and resistance in a circuit
- Prototyping basic circuits on breadboards
- The difference between parallel and series circuits
- Nitty-gritty details about LEDs and how to wire them up in several useful configurations
- Identifying and selecting the right resistor for different circuits and components
- Calculating the resistance in a series circuit and a parallel circuit

For this chapter, you'll need

- 1 Arduino Uno
- 4 standard LEDs, red
- 4 220 Ω resistors

- 4 560 Ω resistors
- 1 100 Ω resistor
- 1 push-button switch
- 1 9 V battery
- 1 9 V battery snap
- 5 red and two black jumper wires
- 1 half-size breadboard



Designing and building circuits may be completely new to you, and may seem intimidating. The good news is that there are just a handful of core concepts to wrap your head around. Once you understand the interplay of voltage, current, and resistance—as formalized in Ohm's law—you're well on your way to being able to understand basic circuits.

There are a couple of metaphors traditionally used to illustrate voltage, current, and resistance. The most common analogy is a hydraulic (water) system involving tanks and pipes. Effective, but not always memorable. Let's try a different adventure.

3.1 Voltage, current, and resistance

High in the mountains, deep in the forest of some place that doesn't exist, a tribe of gnomes found themselves inexplicably in possession of an infinite supply of jellyfish.

The gnomes, being ornery and mischievous, struck out to find a humorous use for the otherwise-inert creatures. They found great fun in dropping jellyfish over cliffs, watching them splash into the lake below or bounce off the roofs of local villages.

The nearby townspeople were initially inconvenienced but soon recognized that the plummeting invertebrates carried energy and could be a free source of power for their cookie factories—but only if the onslaught could be harnessed safely. So they observed, and, over time, came to understand and manipulate the core factors of electrical circuits: voltage, current, and resistance.

Townspeople noticed quickly, for example, that the higher and steeper the cliff, the more energy the tossed jellyfish have when they reach the lake on the valley floor. Lesser drop-offs don't provide as much potential energy for hijinks when the jellyfish splash down (figure 3.1).

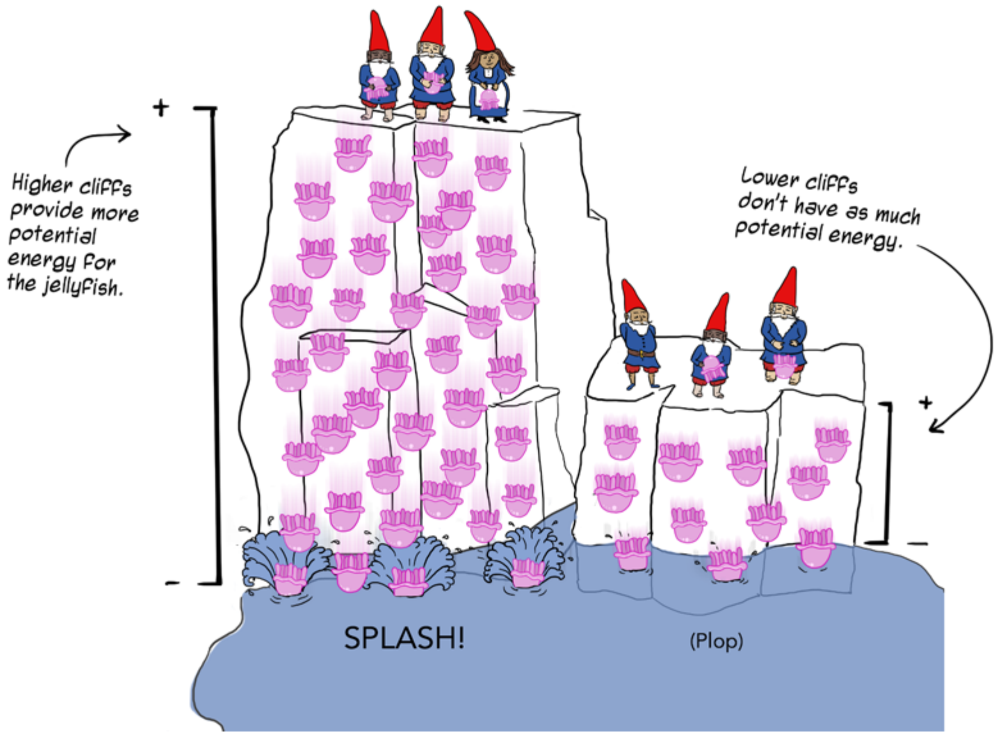


Figure 3.1 Higher cliffs provide more "voltage," that is, electrical potential. *Voltage* is like electrical "pressure," pushing the charges (jellyfish) from high potential energy toward a location of lower potential.

Voltage is a measurement of the difference of potential energy between two points. It's something like pressure or tension or gravitational force, as electricity is always itching to move from higher voltage to lower voltage. Voltage, measured in volts, is *potential* energy, but voltage alone, without moving charged electrons (jellyfish), can't wreak any havoc (figure 3.2).

Figure 3.2 Voltage is potential energy.

For something interesting to happen, jellyfish need to get actively chucked over the edge of the cliff, a task that the gnomes are more than happy to perform.

The townspeople learned to measure jellyfish *current* by staking out a spot on the cliff and precisely counting the number of jellyfish that passed by, over a precise period of time (figure 3.3). Current, the flow of electric charge, is measured in *amperes*, often abbreviated as *amps*.

The townspeople needed to find a way to manage the current of jellyfish so that it wouldn't overwhelm the delicate cookie presses and ovens. This is the lynchpin of

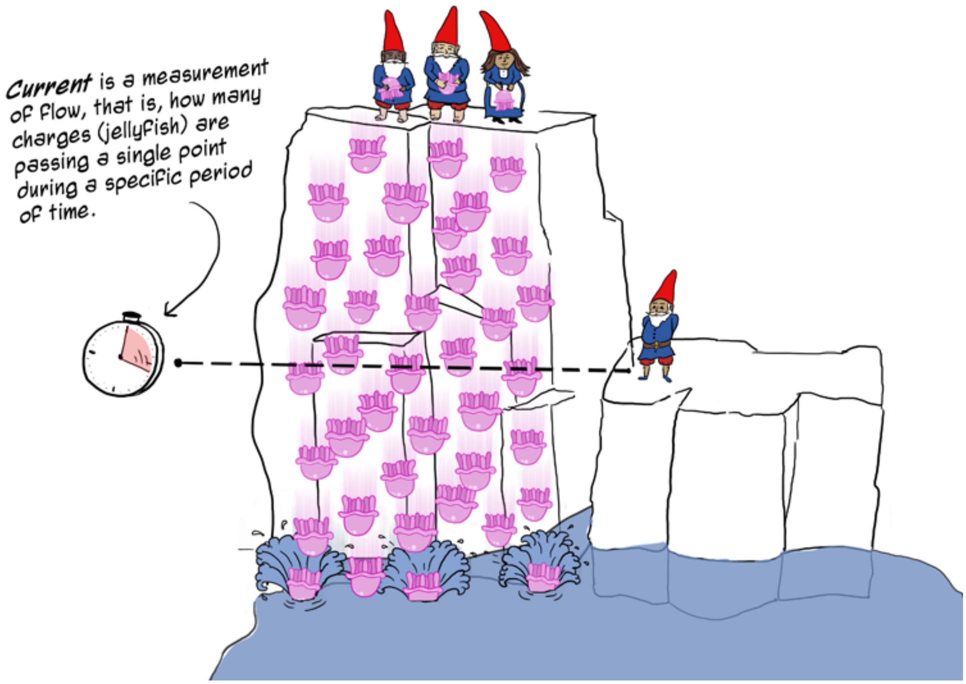


Figure 3.3 Current, the flow of electricity, can be measured by counting how many charges (jellyfish) pass a specific spot on a cliff during a defined period of time.

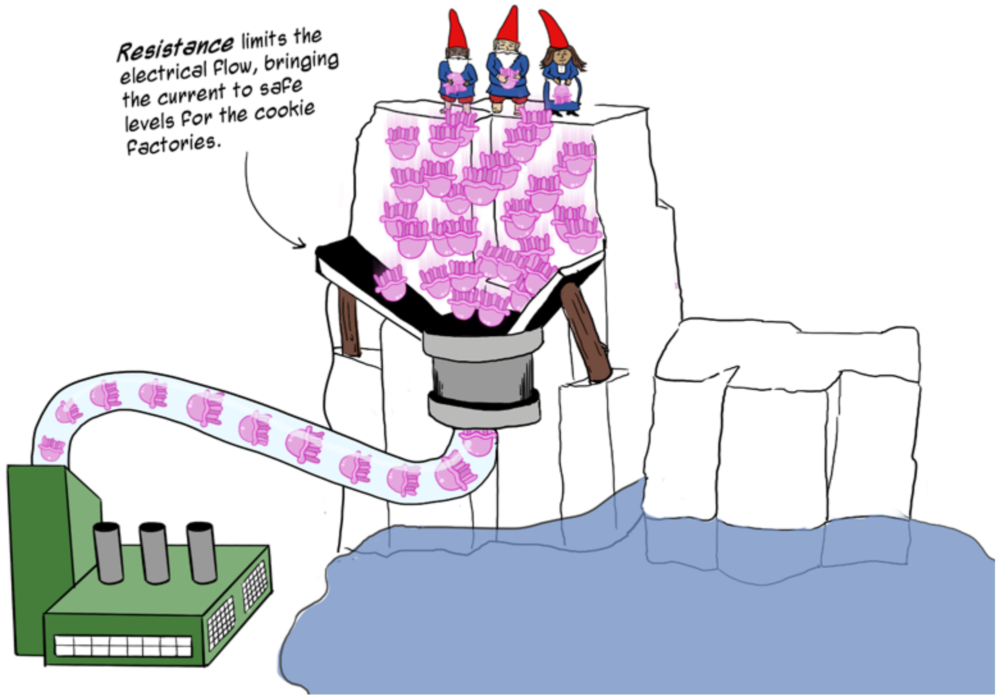


Figure 3.4 Townspeople add resistance to the circuit by channeling falling jellyfish through a series of tubes. Increasing resistance lowers the current.

jellyfish circuit control: *resistance*. Resistance is how much a material is able to resist electrical flow. It's measured in *ohms*.

They engineered jellyfish-channeling systems into the cliff faces (figure 3.4), restricting the jellyfish flow to a more reasonable level. For circuits near the higher cliffs (more voltage), these systems had to be more robust because of the immense jellyfish-falling pressure from above.

A summary of the townspeople's discoveries is shown in table 3.1.

Table 3.1 Voltage, current, and resistance

Factor	What it means	Abbreviated as	Measured in units
Voltage	The difference of electrical potential between two points, akin to electrical "pressure." It's what pushes electrical charges through a circuit.	V	Volts
Current	Electrical flow: how many electrical charges are passing a single point during a defined period of time.	I	Amperes (amps)
Resistance	A measurement of a material's ability to resist electrical flow.	R	Ohms (denoted by the Ω symbol)

In the end, the townspeople perfected the circuit and the jellyfish helped to make some of the best cookies around.

There's a power source—troops of gnomes—tossing jellyfish over a cliff. The higher the cliff, the more *voltage* (potential energy) is supplied to the circuit. The *current* (flow) of jellyfish heads toward the factory machinery.

To reduce the jellyfish current to manageable levels, channeling systems and pipes add *resistance*.

Once the jellyfish have given power to the cookie-making machinery and reached the floor of the factory, they reach the point of lowest potential in the circuit. Jet-pack-wearing gnomes act like a pump of sorts, hoisting the weary jellyfish back up the cliff where they can be thrown over again. And again and again… (figure 3.5).

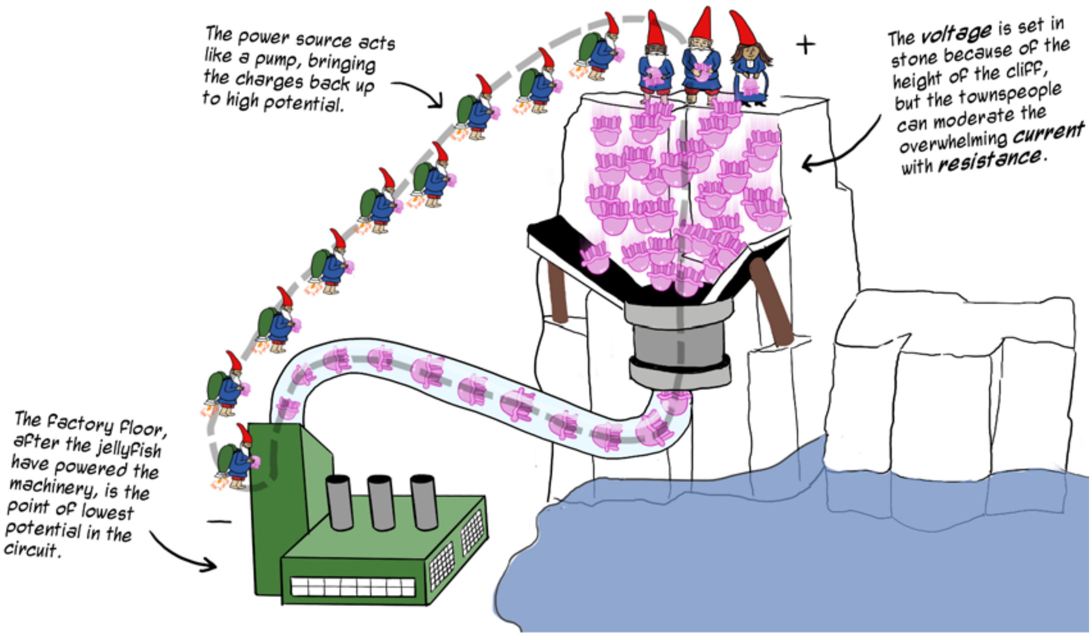


Figure 3.5 A complete gnome-and-jellyfish "circuit"

Voltage, current, and resistance are vital concepts of basic circuitry. The next step is to understand how these factors relate to each other, and how they apply to real-world circuits.

3.1.1 Ohm's law

Voltage, current, and resistance are related to each other in consistent ways. Each of the three factors is like a lever: tweak one and you'll affect the others. These interplays became so central to the town's populace that the factories started producing cookies that illustrated the relationships (figure 3.6).

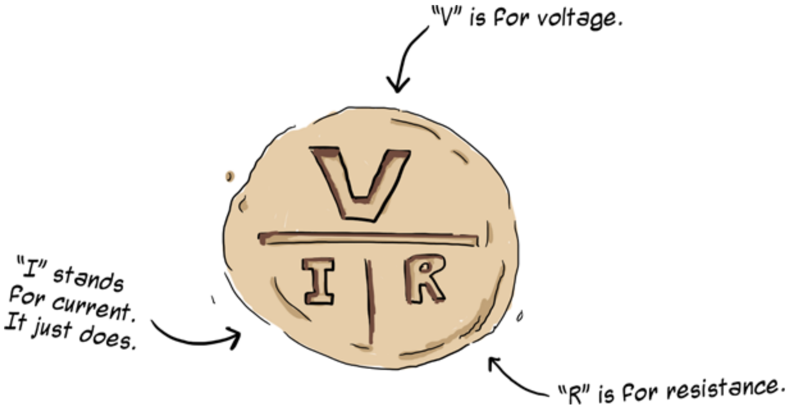


Figure 3.6 The townspeople's new signature cookie shows the relationship between voltage (V), current (I), and resistance (R).

The bearer of the cookie can bite off the factor they wish to determine—then see how it can be derived from the other two factors (figure 3.7).

Georg Ohm figured out these key relationships between voltage, resistance, and current back in the 1820s, well before the clever cookie-townspeople, which is why *Ohm's law* bears his name. If you prefer your math in non-cookie form, these are the relevant equations:

```
V = I x R (voltage equals current times resistance)
I = V / R (current equals voltage divided by resistance)
R = V / I (resistance equals voltage divided by current)
```

"OK," you might be thinking, "but how do I apply this in the real world?"

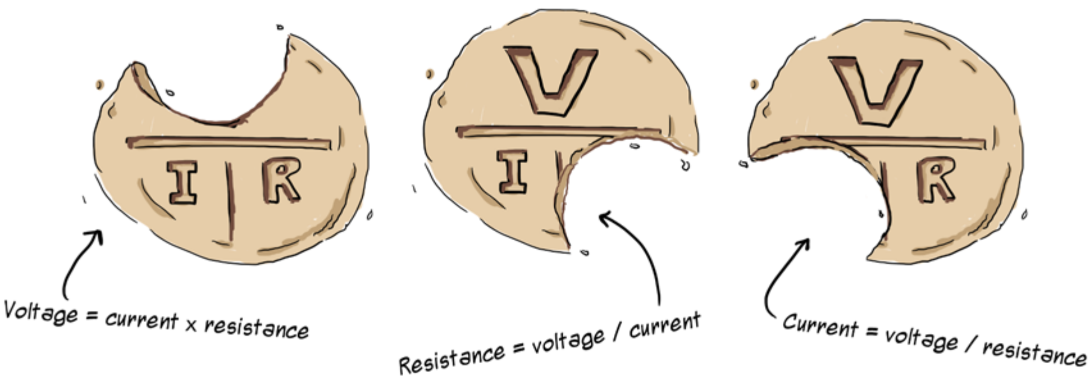


Figure 3.7 By biting off the edge of the cookie imprinted with the factor the cookie-eater wants to figure out, they can quickly see the equation they need to solve. For example, if they want to determine resistance (R), they could bite that off and see that R = voltage (V) divided by current (I).

Applying Ohm's law to real-world circuits

Designing and building basic circuits starts with the right balance of the key factors: voltage, current, and resistance.

Table 3.2 outlines a few common examples of how to adjust voltage, current, and resistance in basic circuits. The examples aren't exhaustive (for example, there are additional ways to adjust voltage in a circuit) but they highlight the kinds of things we'll be doing in the short term to make our circuits work correctly.

Table 3.2 Adjusting voltage, current, and resistance in hobby electronics

Factor	Relationship	Example of common way to increase	Example of common way to decrease
Voltage	V = IR	Use a power supply with a higher voltage.	Use a power supply with a lower voltage.
Current	I = V / R	Remove resistance by removing resistors or using resistors with lower resistance. Current can also be increased by raising the voltage of the power supply.	Add resistance by adding resistors or using resistors with higher resistance. Current can also be decreased by lowering the voltage of the power supply.
Resistance	R = V / I	Add resistors or use resistors with higher resistance.	Remove resistors or use resistors with lower resistance.

One of the most common calculation needs that comes up in hobby electronics hacking is "given a supply *voltage*, what *resistor* do I need to use to make sure my component is supplied with a desired *current*" (figure 3.8)?

Voltage is often defined by the power supply for the project—batteries, USB power, DC adapter—and you know you want to provide a particular current to a component in the circuit. Voltage and current are, then, defined, which means you need to solve for R, resistance.

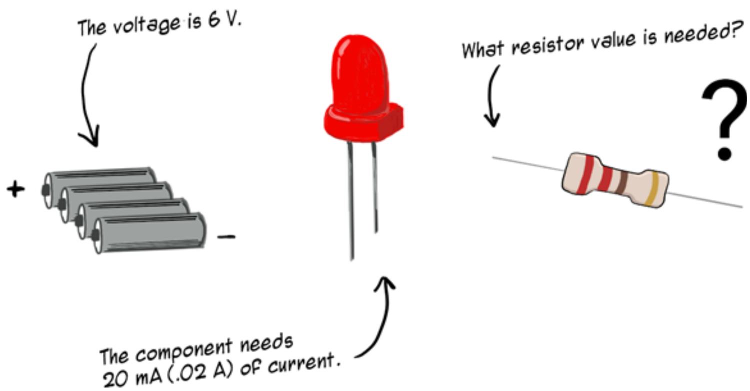


Figure 3.8 A common real-world Ohm's law problem: what resistor value is needed to provide an LED with 20 mA of current in a circuit with a 6 V supply voltage?

Say you know that your supply voltage is going to be 6 V, and you have a component that needs 20 mA (.02 A, or 20 thousandths of an amp) of current (figure 3.10). Solving for R (resistance in ohms) means dividing V (voltage in volts) by I (current in amps) because R = V / I:

```
R = 6 V / .02 A
so
R = 300 Ω
```

WATCH YOUR UNITS! Make sure to keep your units consistent when using Ohm's law equations. Current should always be measured in amps (A), voltage in volts (V), and resistance in ohms (Ω). Volts and ohms tend to be straightforward, but when you're dealing with current in hobby-electronics ranges—often tens of milliamps—don't forget to express those values in amps or you'll get the wrong answer:

```
300 (Ω) = 6 (V) / .02 (A)
```

but

```
300 (Ω) ≠ 6 (V) / 20 (mA)
```

OK! Almost ready. But before we start cobbling together circuits with this new understanding, let's have a grown-up moment and protect ourselves against some potential problems.

3.1.2 Problems and dangers

In the kind of electronics hacking we're doing, we're working with voltages that are quite low—5 V or 3.3 V are typical examples—using components that draw current measured in tens of milliamps (mA). These kinds of current and voltage combinations aren't going to throw you across the room (or worse) if you do something slightly daft. But there are a couple of things to be aware of (and avoid).

AVOID TOO MUCH CURRENT

The first of the two problems arises when you provide too much current to a component in a circuit (often by using too low of a resistor value or forgetting to add one to the circuit entirely). Some of that energy is converted into what is likely a desired outcome—light in the case of an LED, for example—but the rest of it has to get used up too. If you provide 100 mA of current to an LED that's rated for a maximum of 20 mA for steady (not blinking) use, that's not going to go well in the long run. The first sign of too much current is often warmth—the LED will start to feel hot to the touch. At a certain point, it'll get overwhelmed and burn out completely.

AVOID CREATING A SHORT CIRCUIT

The second "uh oh" has the potential to be much worse. If there's no *load* in a circuit—that is, no components or resistors to draw or resist current—things get nasty indeed.

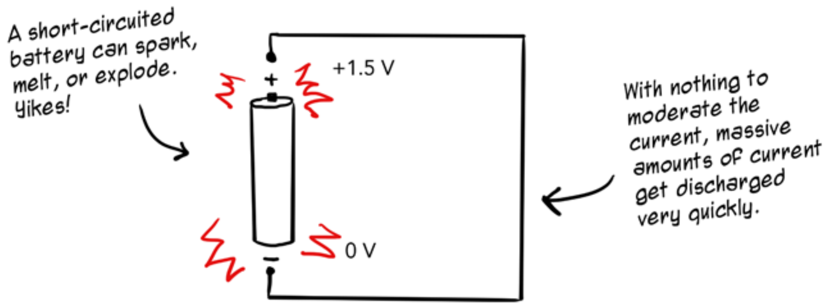


Figure 3.9 A short circuit has no load to moderate the current. A common illustration of a short circuit is running a wire directly from the positive to the negative terminal of a battery (don't do this in real life).

If there is, say, a path running directly from the positive to the negative terminal of a battery, there's no resistance to limit the current running through the circuit. This is a *short circuit* (figure 3.9), and it causes immense amounts of current to discharge through the circuit very, very quickly. This energy can cause heat, fire, even explosions.

Development boards can protect you from the worst of this. If you short-circuit the Uno's power to ground (P.S., don't), you won't blow up because the board has *current limiters* on its output pins (your board may be toast, though). The most current you'll ever get out of its 5 V pin is 450 mA; the most out of a single I/O pin is about 40 mA. You get no such protection against outrageous currents if you're working with batteries—you can cause a regrettable festival of sparks, or worse. So be careful.

3.2 Building circuits

Now that you've been debriefed, let's experiment! As you start adding components to circuits, you're going to need a way to lay them out without losing your mind. Twisting wire together or trying to hold several components in place in a circuit with your fingers is impractical (and maybe a bit risky).

Instead, *breadboards* make great foundations for laying out circuits. They play a role sort of like a LEGO base, providing a grid to plug components and wires into.

3.2.1 Using breadboards to prototype circuits

Breadboards for prototyping circuits are *solderless*, meaning you can stick things right into the board without any need for solder. They come in various shapes and sizes but are consistent in how the board's connections are wired. Figure 3.10 shows the layout of a *half-size* breadboard (*full-size* breadboards are like two half-sized breadboards connected end to end; they're twice as long).

A typical breadboard combines horizontal *terminal rows* (a technical-sounding term that really means "spots to plug components into") with vertical *power rails* (holes meant for connecting power between power sources and components) on both sides.

Terminal rows often have a notch between each set of five holes. A ten-hole terminal row—two sets of five connected holes divided by a notch—is a common layout.

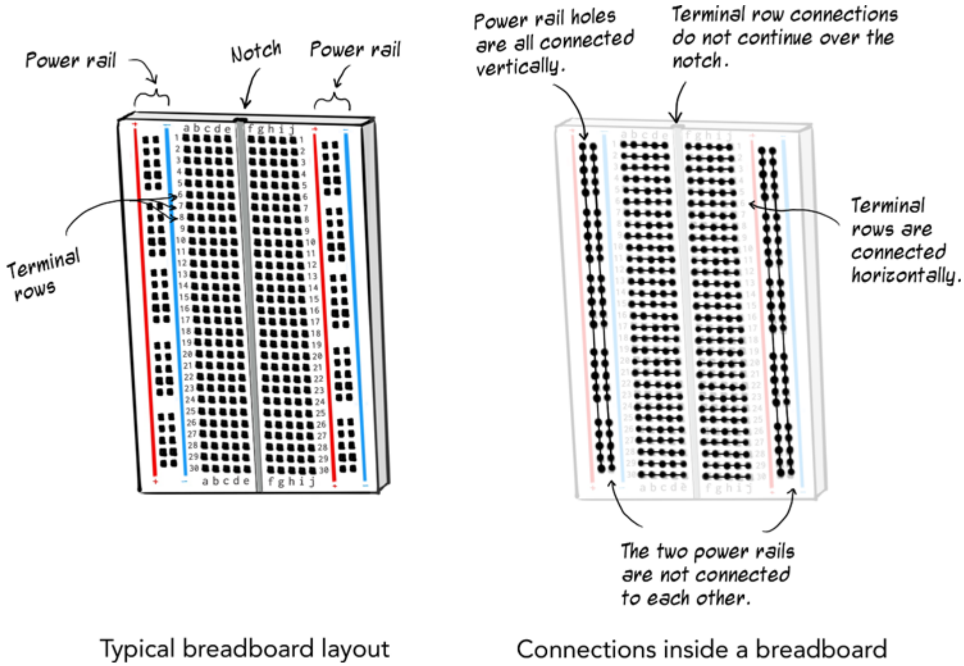


Figure 3.10 A typical breadboard and the connections inside it

Each hole in a five-unit row is electrically connected, but connections do not continue across the notch.

Power rails have two holes per row: one for positive and one for negative. These are usually helpfully marked for you in red (positive) and blue or black (negative). Connections in the power rails run vertically down the length of the board. The power rail on one side of the board isn't connected to the power rail on the other side of the board.

Let's get a feel for how these connections work by using a breadboard to rewire the simple LED circuit from the last chapter.

3.2.2 Wiring a simple LED circuit on a breadboard

What you'll need

- 1 Arduino Uno and USB cable
- Jumper wires: two red, one black
- 1 standard red LED
- 1 220 Ω resistor

First, let's check in with Ohm's law to figure out what we'll need to adjust to make the circuit work correctly. Here's what we know:

- The Arduino will provide a 5 V supply voltage.
- The maximum current we should run through the LED is somewhere around 20 mA (0.02 amps)—for most standard LEDs, 20 mA is a general rule of thumb.

Selecting a resistor for the LED

Because we have a fixed voltage and a target current, the variable value is resistance. What resistor value is needed to create the circuit? Remember,

```
R = V / I
```

so

```
R = 5 V / 0.02 A
R = 250 Ω
```

Resistors come in certain common resistance values, and 250 Ω is not a commonly produced resistor. Calculating a needed resistance value only to find that there is no such resistor happens all the time—not to worry. Typically, the rule of thumb is to round *up* to the next common resistor value (having too much resistance is safer than not enough, ordinarily).

For the moment—trust me, I'll explain shortly—we're going to do the opposite and round *down* a bit to the nearest common resistor: 220 Ω.

Finding the right resistor

Resistors are color-coded in a standard way to aid identification. There are two striping systems out there: four-band resistors and five-band resistors. Four-band resistors are somewhat more common.

Every resistor has bands of color representing *leading digits* in the resistor's value. Four-band resistors have two of these bands, whereas five-band resistors have three.

The last two bands of a resistor are its *multiplier* band and its *tolerance* band, respectively.

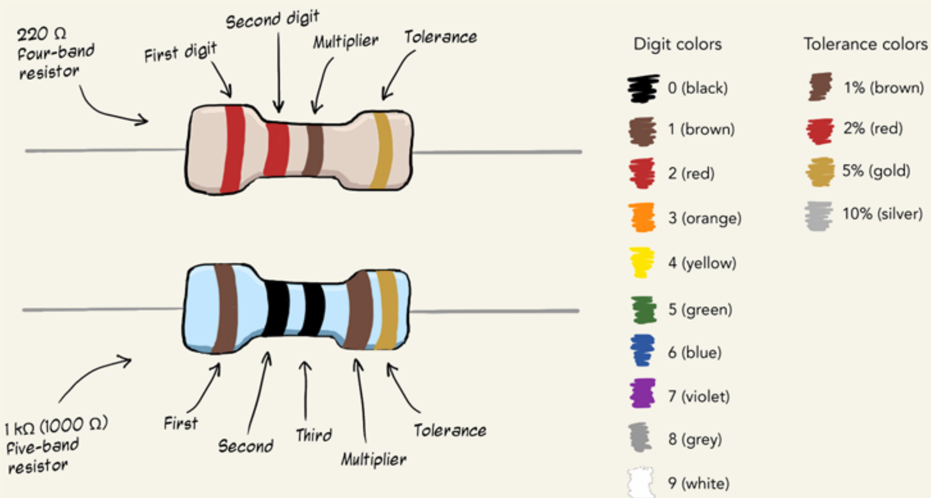
The color of the *multiplier band* indicates how many zeros to add after the value indicated by the preceding digit bands. In other words, multiplying the leading digits by this power of ten gives you the resistor's *value*.

The *tolerance band* color indicates how accurate the resistance is guaranteed to be. A tolerance of +/-5% (gold) is common for the types of resistors we're using.

A four-band resistor has two digit bands. The four-band resistor in the following figure is coded as follows:

1. First digit: 2 (red)
2. Second digit: 2 (red)
3. Multiplier: 1 (brown) = 101
4. Tolerance: +/- 5% (gold)

(continued)



Resistors are coded with colored bands to indicate their resistance and tolerance.

Its value is

```
22 * 10^1 = 220 Ω @ +/-5%
```

The five-band resistor in the figure is coded as follows:

1. First digit: 1 (brown)
2. Second digit: 0 (black)
3. Third digit: 0 (black)
4. Multiplier: 1 (brown)
5. Tolerance: +/-5% (gold)

The value is

```
1 0 0 * 10 ^ 1 = 1000 Ω (or 1 kΩ) @ +/- 5%
```

CIRCUIT DIAGRAMS AND SCHEMATICS

There are a couple of ways to represent a circuit visually, to provide other builders with a "map" to recreate the circuit. The most formal way to represent a circuit is through a schematic, as seen in figure 3.11. A *schematic* is a graphical representation of components using standardized notation and symbols. It can be thought of as a sort of visual, abstracted graphic: the position of components in a schematic don't necessarily correspond to their layout in physical space.

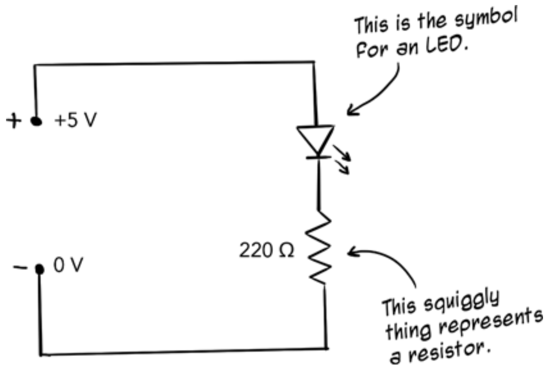


Figure 3.11 Schematic of the simple LED circuit. Schematics are concise and standardized representations of circuits, but they don't show how to position components in physical space.

There are many possible ways to physically implement the schematic in figure 3.11, so a wiring *diagram* can be a helpful tool. Diagrams, like the one in figure 3.12 (created with the open source Fritzing desktop application), show one specific implementation of the circuit and how it could be laid out (in this case, on a breadboard).

Schematics can feel mathy and abstract at first, but they're universally used in the electronics community. You'll get to know more symbols and conventions as we continue our journey, and they'll start feeling more comfortable. To improve your understanding of schematics, make a habit of comparing wiring diagrams back to their source schematics.

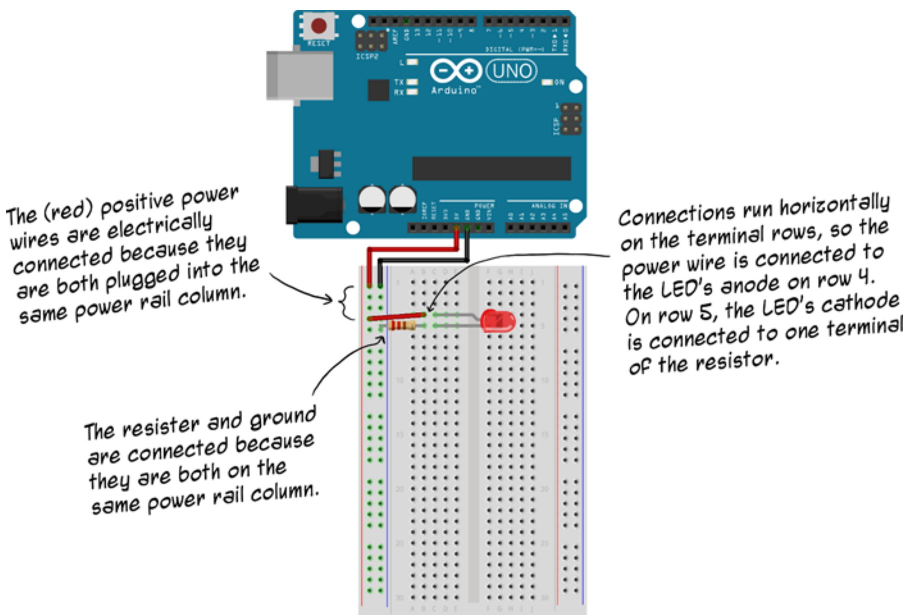


Figure 3.12 Wiring diagram of the simple LED circuit, showing a specific physical layout to implement the circuit from its schematic. This diagram was created in the open source Fritzing desktop application.

Plugging things in the right way: polarity

LEDs are *polarized*, meaning they need to be plugged in in a certain way to function correctly. As you learned earlier, the anode is the positive terminal and the cathode is the negative terminal.

Resistors are not polarized, meaning it doesn't matter which direction you plug them in. They work just fine oriented either way.

Another thing to note is that although I'll give you precise row and column breadboard coordinates in this exercise, you could just as successfully plug the components into any row or column as long as the connections work out in the end (terminal rows are connected horizontally; power rails, vertically).

Time to build the circuit. Referring to figure 3.12, begin by connecting components into the terminal rows. Plug the anode of the red LED into hole 5C (row 5, hole C) and its cathode into hole 5D (the next row down). Plug one end of the resistor into hole 5B, next to the LED's cathode.

POWERING THE CIRCUIT

The circuit needs to be connected to power. First, you'll need to connect the components in the terminal rows to the power rail—recall that terminal rows are isolated (for good reason!) from the power rails.

Plugging things in the right way: jumper wire colors

If you have a packet of colored jumper wires, you might be wondering which colors are for what purpose. By convention, power connections are usually made with red (positive) and black (ground) wires. Other colors you may see representing negative power connections include white or other dark colors (brown or purple).

Green and yellow are often used for input and output connections, which will come up in later chapters. Although there is widespread consistency in using red and black/white for power connections, different hackers use different color combinations for various other things. Long story short: there are no hard and fast rules, but try to be consistent in whichever combination you implement.

Jumper wires aren't polarized. They can be connected in either direction.

Use a red jumper wire to connect hole 5B—electrically connected to the anode of the LED—to a hole in the red (positive) column of the power rail. The fourth row down should do nicely if you can't decide on a favorite. Connect the resistor's free end directly into the negative power rail as shown in figure 3.12. Now there's an unbroken path leading from the positive power rail, through the red wire to the LED, out of the LED and through the resistor back to the negative power rail.

The current is flowing… *which* way now?

The kinds of projects we're building with development boards and embedded systems use DC (direct current) circuits. Current flow in a DC circuit is in a single direction, usually rendered as going from positive source (highest potential energy) toward negative, or ground (lowest potential energy).

Technically, this isn't correct—current can be more accurately described as flowing from negative toward positive, and even that's an oversimplification. But the convention of drawing circuits with flow in the positive to negative (+ toward –) direction is deeply entrenched, and there's nothing inherently harmful in representing DC current flow in this traditional way, as long as you're consistent. Fun fact: the practice of envisioning current flow from positive to negative was established by none other than electricity pioneer Benjamin Franklin.

Besides DC, the other type of current flow is AC (alternating current), in which the flow of current reverses directions periodically. Wall-outlet power is AC, oscillating direction 50 or 60 times per second, depending on what part of the world you live in.

UNPLUG THE UNO! As ever, make sure your Uno is unplugged from USB or wall power before connecting it to components or circuits.

The Uno can provide a nice, steady 5 V from its 5 V power pin. Using a red jumper wire, plug one end into the Uno's 5 V pin and the other into the top row of the positive power rail. Run a black jumper wire from the Uno's GND pin to the top row of the negative power rail. This completes the circuit wiring.

Now plug your Uno into USB or wall power. Your LED should light up!

Troubleshooting the circuit

If your LED doesn't light up, there are a few things to check. The most common reason for a circuit like this to fail is that it's an *open circuit*, meaning there's a gap in the path from the positive to negative ends of the circuit. Check the positions of your components and wires, and make sure wires are snugly plugged into breadboard holes. Make sure that none of the exposed metal terminals on the components are touching each other. Also double-check that your Uno's power LED is lit.

If those steps don't do the trick, try swapping out your LED with a fresh one on the off chance that the LED is dead. On occasion, a breadboard's connections can be wonky (though this is more typical for breadboards you've been using for a long time). As a last resort, try wiring the components onto a different part of the breadboard or try with another breadboard.

SERIES AND PARALLEL CIRCUITS

This simple LED circuit (figure 3.13) is a *series circuit*. A series circuit has only one path for electrons—current, those individual flowing jellyfish of the gnome world—to take. Because there's only one possible route, all jellyfish/electrons go through the whole

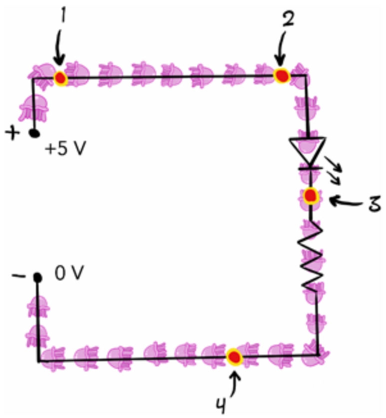


Figure 3.13 A series circuit has only one possible path for the charges (jellyfish) to flow through—there are no branches. The current is the same at any point in a series circuit, meaning that points 1, 2, 3, and 4 all have equal current.

route; they're not able to wander off on some shortcut or side road. That means the current is the same at all points in the circuit (figure 3.13).

When modifying the current of a series circuit with resistors, it doesn't matter whether you use, say, one resistor rated at 200 Ω or two resistors at 100 Ω. The values of the resistors add together and modify the current across the whole circuit (figure 3.14).

VOLTAGE AND SERIES CIRCUITS Although *current* is the same at any point in a series circuit, *voltage* may differ from point to point. We'll talk about this when we build *voltage dividers* in chapter 4.

One detail of these series circuits that might have given you pause is the position of the *current-limiting resistor(s)* (a current-limiting resistor is one that's placed to moderate the current in the circuit).

In figure 3.14, the resistors are connected between the LED and ground; that is, they're positioned "after" the LED. As it turns out, it doesn't matter where the resistors are positioned relative to the LED in a series circuit.

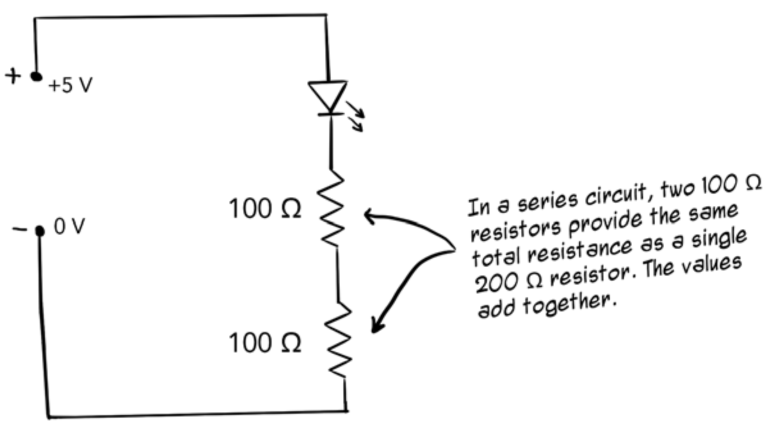


Figure 3.14 Resistor values add together in a series circuit.

A current-limiting resistor placed in a series circuit affects current throughout the whole circuit, no matter its position—remember, current is the same at all points in a series circuit.

3.2.3 Expanding a series circuit with a button

What you'll need

- 1 Arduino Uno and USB cable
- 1 push button
- 1 standard red LED
- 1 220 Ω resistor
- 3 jumper wires

In the simple LED series circuit you wired, all of the current runs through the LED, then through the resistor, and then to ground (figure 3.14). A single gap in a series circuit will cause the entire circuit to stop working, because it's a gap in the only path through the circuit. That makes it possible to activate and deactivate the whole circuit with one switch.

You can see how this works by adding a button to the circuit. A *button* is a kind of switch that only completes a connection when it's pressed down (sometimes buttons are called *momentary switches*) (figure 3.15).

In figure 3.16, the button is connected to a breadboard, oriented such that the "always-connected" pins span the notch in the middle. That means the top row's highlighted connections are always connected electrically, as are those highlighted in the bottom row. While the button is inactive (not pressed), the two rows are isolated from each other. When the button is pressed, however, a connection is made between the top and bottom pins on the button's left side, and the top and bottom pins on its right

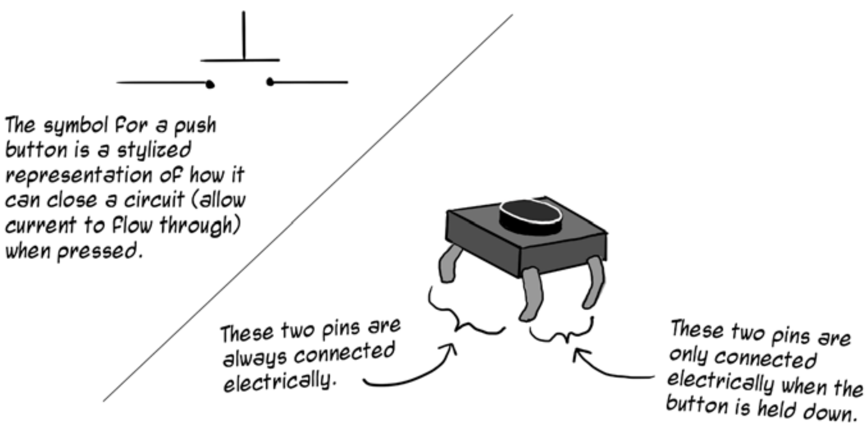


Figure 3.15 Pins on opposite sides of the button are always electrically connected, whereas pins sharing a single side are only connected when the button is held down.

side. The effect is that both highlighted rows are electrically connected to each other while the button is pressed.

When a properly connected button is pressed and held, the circuit is *closed*, completing the path and allowing electrons to flow through the circuit. When released, the circuit is *open*—it has a gap and no current flows. The schematic of the circuit you need to construct is shown in figure 3.17.

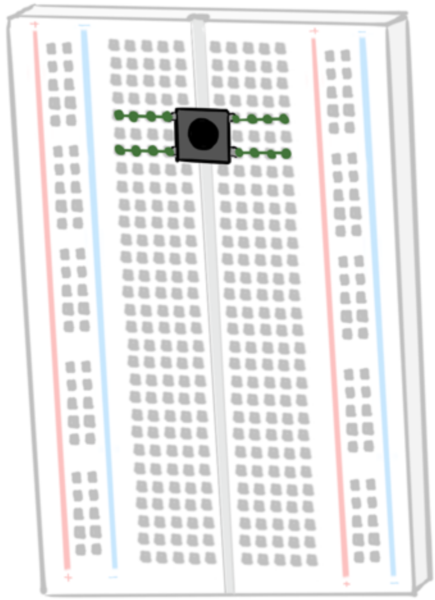


Figure 3.16 A push button connected to a breadboard, spanning the center notch. When not pressed, the pairs of pins at the top and bottom of the button are connected electrically (horizontally). When pressed and held, the pairs of pins on the left and right sides of the button are electrically connected (vertically).

BUILD THE CIRCUIT: BUTTON AND LED

To build the button circuit in figure 3.18, follow these steps:

1. Disconnect the Uno from power.
2. Remove the LED and the resistor from the breadboard.
3. Using a black jumper wire, connect GND on the Uno to the top row of the negative power rail on the right side of the breadboard.
4. Connect the button to the breadboard. Your button might be a different size and fit more comfortably between different rows, which is fine.
5. Plug the LED's anode (longer leg) into a slot in the same row as the bottom two legs of the button.
6. Plug the LED's cathode in one row down.
7. Connect the resistor from a slot in the same row as the LED's cathode to the negative power rail. You can leave the red wire connecting the left-hand power rail to row 4 in the same place as it was in the previous exercise.
8. Reconnect the Uno's power.

The LED should not be lit initially. Press the button. The LED should light up for as long as you hold the button down.

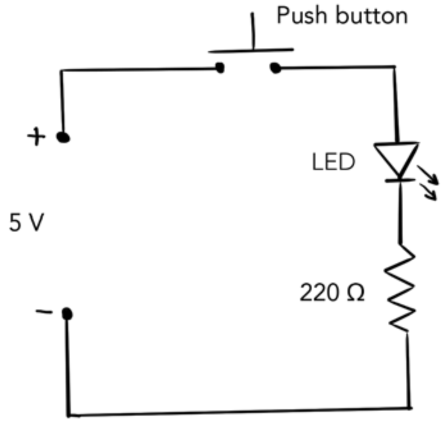


Figure 3.17 Schematic of the updated circuit, integrating a push button

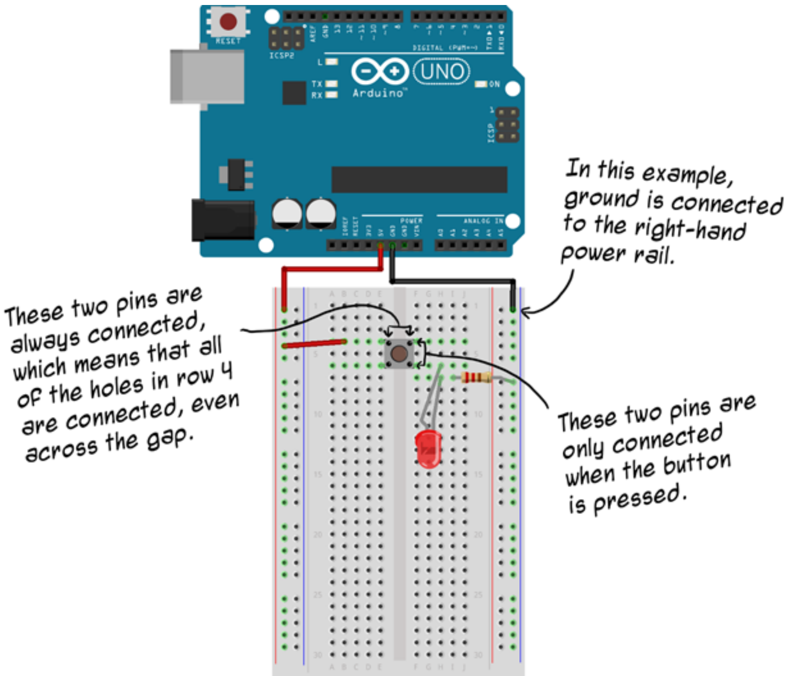


Figure 3.18 Wiring diagram of adapted LED circuit with a push button

3.2.4 *LEDs in series*

There's one more stop on the series-circuit discovery tour. Let's construct a circuit containing multiple LEDs wired in series—that is, multiple LEDs on a circuit that only has one path. Here's where I can come clean about the detail left out in the LED-resistor calculation we did earlier in the chapter.

To jog your memory, the calculation was aimed at finding the right resistor value for an LED—target current 20 mA—in a 5 V circuit:

```
R = 5 V / .02 A
R = 250 Ω
```

Instead of rounding up to the next common resistor value, we rounded down to 220 Ω.

The reason that a 220 Ω resistor is more than plenty for an LED in a 5 V circuit at 20 mA is that, because of a characteristic of LEDs, we don't need to account for the full 5 V when calculating for the right resistor value. In fact, 220 Ω is slightly high.

There's a relevant law, called Kirchoff's voltage law (*KVL* to those in the know). It states that all of the voltage in the circuit has to be in balance: the amount generated has to be the same as the amount used. Voltage in, voltage out.

In the LED-series circuit, 250 Ω would be the correct approximate resistor to use if the resistor were the only thing in the circuit "using" some voltage. But it's not.

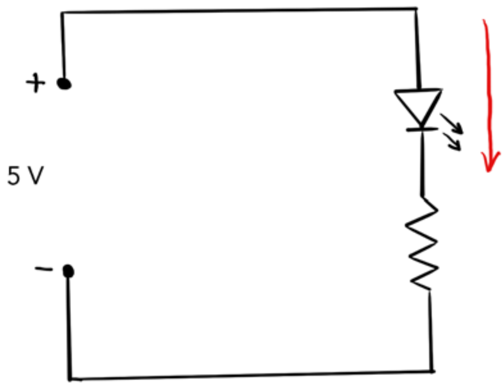


Figure 3.19 As electricity travels across an LED, some voltage is used, or *dropped*. The amount of voltage used is called the LED's *forward voltage*. It ranges between about 1.8 V and 3.5 V, depending on the color of the LED—the higher the frequency of the emitted light, the higher the voltage drop.

LEDs have a metric called *forward voltage drop*. There's a bit of detail to that, but for our purposes it's the approximate amount of voltage the LED will use up in the circuit (figures 3.19 and 3.20).

While most workaday standard LEDs have a consistent *forward current* (for purposes of brevity, you can think of that as "roughly the current it should receive") of 20 mA, forward voltage differs between LEDs, mostly related to the color of the LED.

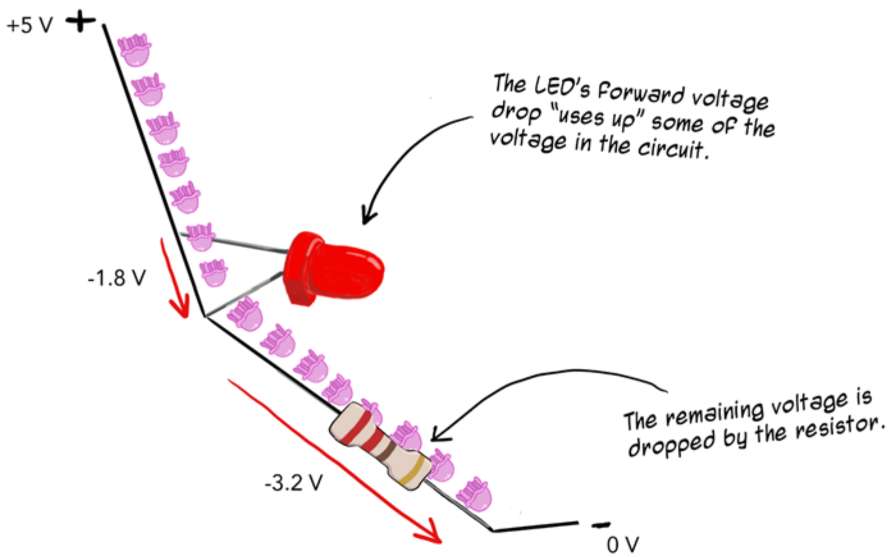


Figure 3.20 Hearkening back to the analogy of voltage as cliff steepness or height, the same series-LED circuit can be seen from a different perspective. As the jellyfish current moves through the LED, 1.8 V is "used up," reducing the remaining voltage to 3.2 V (reducing the steepness). That 3.2 V is then used up by the current-limiting resistor. When the jellyfish reach the point of lowest electrical potential, all of the voltage in the circuit has been accounted for.

Red LEDs have a forward voltage that varies for the most part between 1.8 and 2 V. If the LED in the circuit has a forward voltage drop of 1.8 V, we can subtract that from the system voltage of 5 V, and that is the voltage that the resistor needs to account for:

```
5 V (supply voltage)
- 1.8 V (red LED forward voltage)
--------------------------------
= 3.2 V (remaining voltage)
```

Given that we want to aim for that 20 mA current (0.02 A), the resulting Ohm's law equation is

```
R = 3.2 V / 0.02 A
```

or

```
R = 160 Ω
```

A 220 Ω resistor is close enough in value to be just dandy—rounding up to the next common resistor value as per convention. Now, 220 Ω is more resistance than 160 Ω, so you might wonder what effect using a higher resistance value has on the LED and the circuit. Higher resistance with steady voltage means the current goes down (because, as always, Ohm's law):

```
I = V / R
I = 3.2 / 220 Ω
I = 0.0145 A
```

In the end, the LED gets less current than 20 mA—about 15 mA (.0145 A). The amount of current provided to an LED is directly proportional to its brightness: an LED getting 15 mA will not be as blindingly bright as one receiving 20 mA.

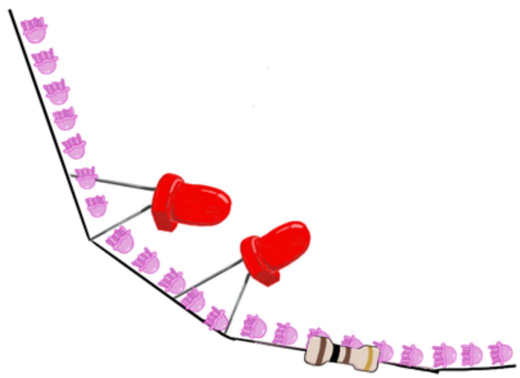


Figure 3.21 Each of the two LEDs drops some of the voltage in the circuit, and, again, the remaining voltage is dropped by the resistor.

What's the situation if another identical red LED is added into this series circuit? What's the needed resistor? We'll need to account for the voltage drop of both LEDs (figure 3.21).

Fortunately, this is a question of straightforward arithmetic. After subtracting the voltage drop of each LED from the total circuit voltage, we're left with 1.4 V:

```
5 V (supply)
- 1.8 V (LED 1)
- 1.8 V (LED 2)
--------------
= 1.4 V

R = 1.4 V / 0.02 A
R = 70 Ω
```

Rounding up to the next common resistor value, a 100 Ω resistor will do nicely, as shown in the schematic in figure 3.22.

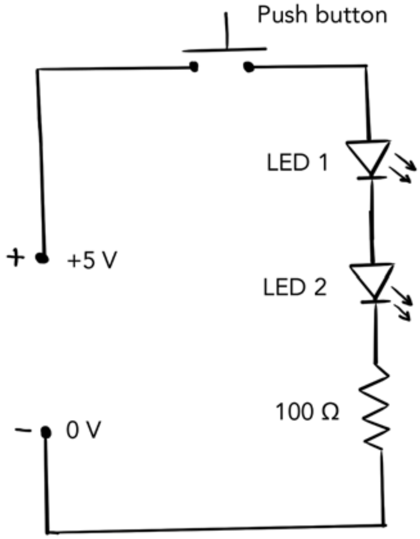


Figure 3.22 The two LEDs are wired in series with a 100 Ω resistor. There's only a single path through this circuit when the button is pressed: through the first LED, through the second LED, through the resistor, and back to ground.

BUILD THE CIRCUIT: TWO LEDS IN SERIES

What you'll need

- 1 Arduino Uno and USB cable
- 1 push button
- 2 standard red LEDs
- 1 100 Ω resistor
- 3 jumper wires

To build the circuit in figure 3.23, follow these steps:

1. Unplug your Uno from power.
2. Unplug the 220 Ω resistor from the breadboard and put it away.
3. Plug the anode of a second LED into the same row as the cathode from the first LED (row 7 in figure 3.23).
4. Connect a 100 Ω resistor from the negative power rail to the row containing the second LED's cathode (row 8).

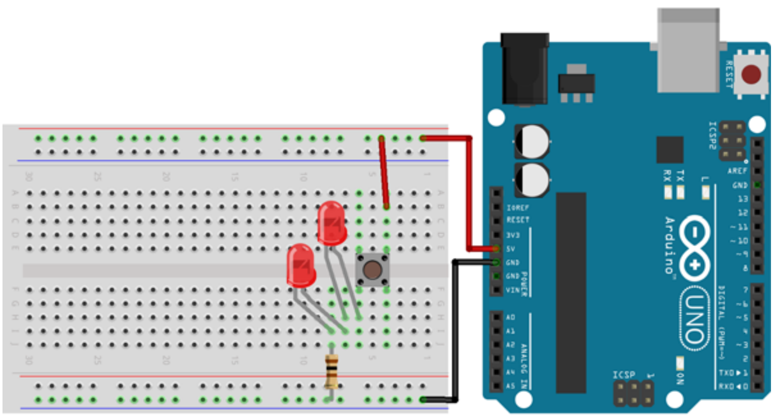


Figure 3.23 Wiring diagram of two LEDs in series

If you plug your Uno into power and then press the button, both LEDs should light up. They're wired in series, which may be easier to visualize by looking at figure 3.22 again.

And what if you add a third LED? Trick question: you can't really. Not if you want to produce reliable and bright light. There's only 1.4 V "left" after the first two LEDs' voltage drops. That's not quite enough to power a third LED—an LED requires at least its voltage drop value to light up. You might be able to get the three LEDs to light up weakly, but they wouldn't be robust.

It is possible to steadily power more than two LEDs with 5 V, but to do so you need a parallel circuit.

3.2.5 Parallel circuits and current dividers

In a series circuit, there's only one path for electrons to take, but in a parallel circuit, there are two or more possible paths, or *branches* (figure 3.24).

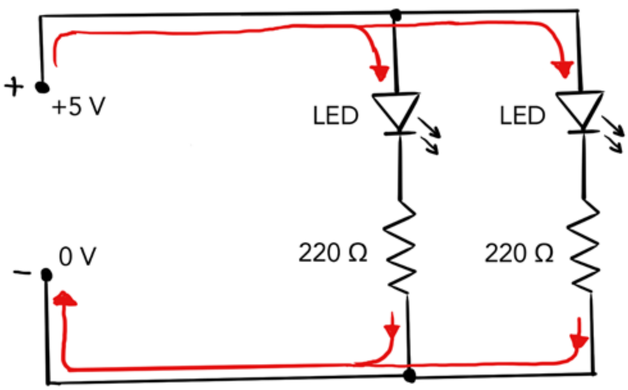


Figure 3.24 In a parallel circuit, there's more than one path, or branch, current can take. In this example, there are two branches: each has its own LED and resistor.

As electrons move through the circuit and encounter a fork in the road, they each make a decision about which branch to take. Current prefers a path with less resistance (figure 3.25).

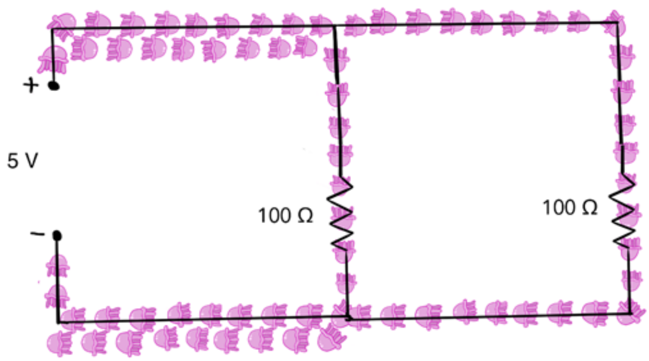


Figure 3.25 Because the two branches in this parallel circuit have the same resistance (100 Ω), half of the current will take one branch and half will take the other.

When calculating how much resistance is in the parallel circuit, things get a bit weird and counterintuitive. Look again at the parallel circuit in figure 3.25. What's the total resistance that the two 100 Ω resistors provide?

The total resistance of series circuits is easy to figure out—just add 'em up and voilà! You've got the total resistance. It's tempting to do the same at first with parallel circuits—to assume the total resistance is 200 Ω. Nope. Or maybe you spied that any given charge going through the circuit is only going through one resistor, not both. So that must mean there's a total resistance of 100 Ω? Also, sorry, nope.

The correct answer is 50 Ω.

I know, I know. That doesn't feel right, but it's true: in a parallel circuit, the total resistance will always be *less than* the smallest resistor value. Let's arm ourselves with Ohm's law and a few deep breaths and examine how this can possibly be. Calculating equivalent resistance in parallel circuits is one of the more challenging concepts to novice electronics hackers, so don't pull your hair out just yet.

Let's break it down and put it back together. The 5 V series circuit with a 100 Ω resistor shown in figure 3.26 will draw 50 mA of current (I) because

```
.005 A (I) = 5 V (V) / 100 Ω (R)
```

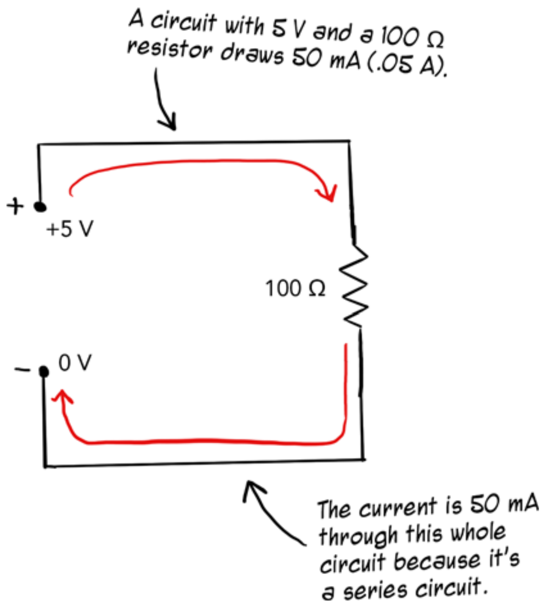


Figure 3.26 This series circuit draws 50 mA because 5 V / 100 Ω = .05 A.

Now say you duplicate that same path—5 V and a 100 Ω resistor—and glom it onto the circuit. That second path will also independently draw 50 mA because the supply voltage (5 V) and the resistance (100 Ω) stay the same (figure 3.27).

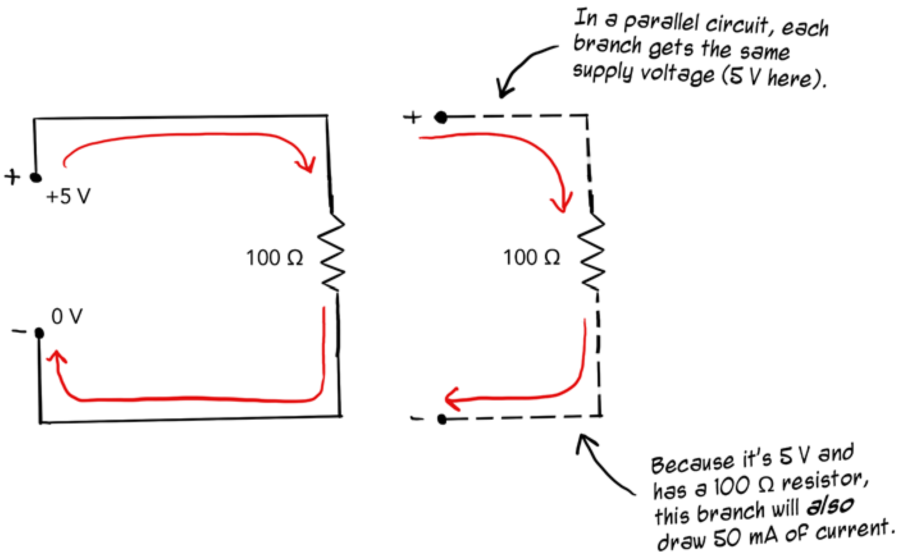


Figure 3.27 Each branch of a parallel circuit in isolation acts like its own series circuit. We can use Ohm's law to verify that this branch will also draw 50 mA.

Added together, the current drawn by the two branches is 100 mA—the total current draw of the circuit has increased (figure 3.28).

Looking at the circuit as a whole now, its total current is 100 mA (.1 A) and supply voltage is, as ever, 5 V. Plugging that into Ohm's law,

```
Total resistance (R) = 5 V (V) / .1 A (I)

R = 50 Ω
```

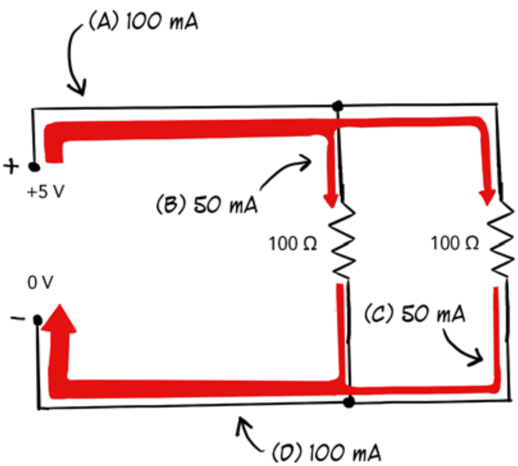
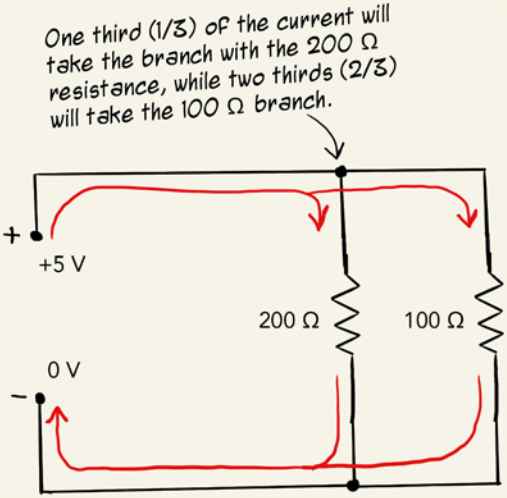


Figure 3.28 The total current going in to the circuit is 100 mA (point A). It splits equally into two branches—each branch gets 50 mA (points B, C). The current rejoins and at point D is again 100 mA. In a parallel circuit, the supplied voltage is constant for each branch but the current varies.

The resistance provided by each resistor in a parallel circuit is reduced because each branch in a parallel circuit increases the total current in the circuit. Current is going up while voltage is constant: resistance goes down.

Current dividers

Any circuit that splits the current coming from the power source into more than one branch is called a *current divider*. The parallel circuit in the following figure is an example of a current divider: some current follows one branch while the rest follows the other.



If the branches have unequal resistance, current will proportionally flow through each branch.

If a charge finds itself at a fork in the road and both available paths have the same resistance, the charge is equally likely to take either route (like the parallel circuit in figure 3.28). But if the resistance is unequal, more charges will opt for the road with less resistance—that is, more current will take the less-resistant branch.

You can calculate the total resistance in the circuit shown in this figure by looking at each branch in isolation and figuring out the circuit's total current draw:

```
Branch 1:
5 V / 200 Ω = .025 A (25 mA) because I = V / R

Branch 2:
5 V / 100 Ω = .05 A (50 mA) because I = V / R

  .025 A (branch 1)
+ .05 A  (branch 2)
=========================
= .075 A (75 mA) total current

R(Total) = 5 V / .075 A because R = V / I
R(Total) = 66.667 Ω
```

Doing calculations for total resistance in this manner can become cumbersome as the number of branches increases. We won't be architecting complex current dividers with lots of branches with different resistances, but if you're the curious type, there's a formula for calculating equivalent (total) resistance in any parallel circuit:

```
1 / R(Total) = 1 / R1 + 1 / R2 + 1 / R3 ... + 1/Rn
```

Some of this parallel-circuit calculation may seem pointlessly convoluted, but it does have useful applications.

Here's the rub: each branch of the parallel circuit is provided with the same voltage. So if we take the series-circuit LED and add more branches to it (turning it into a parallel circuit—figure 3.29), each branch gets "its own" 5 V to work with. This way, we can wire three, four, or even more LEDs on the same circuit and not run out of voltage like we would in a series circuit.

Build the circuit: LEDs in parallel

What you'll need

- 1 Arduino Uno and USB cable
- 4 standard red LEDs
- 4 220 Ω resistors
- 6 jumper wires

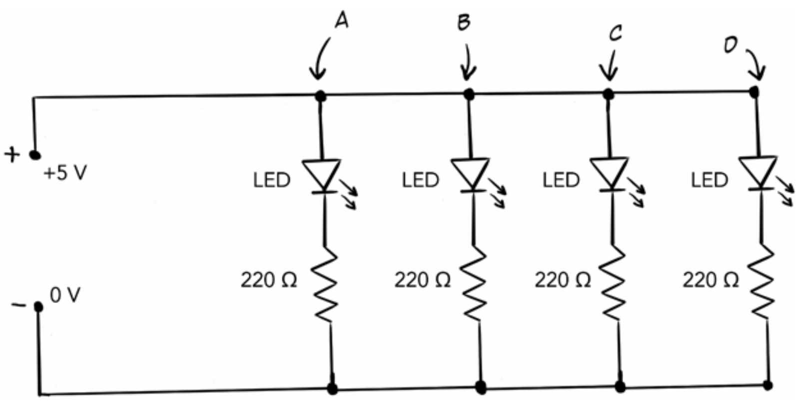


Figure 3.29 In the schematic for the parallel-LED circuit, each of the branches A, B, C, and D is supplied with 5 V. Each branch receives 1/4 of the total current in the circuit.

Figure 3.30 shows a physical layout of the same circuit. To wire four LEDs on the same circuit, start with a fresh, clean breadboard and follow these steps:

1. Connect the anode terminals of four LEDs to a hole in rows 2, 9, 16, and 23.
2. Plug the cathode ends into the same terminal row but on the other side of the gap (such that they aren't electrically connected).

3. Run four red jumper wires from the positive power rail on the left side of the board to the anode row of each LED.
4. Connect a 220 Ω resistor from each LED's cathode row to the negative power rail on the right side of the board.
5. Connect the power rails to the Uno's 5 V and GND pins (using a red and black jumper wire, respectively).
6. Plug the Uno into USB or DC power.

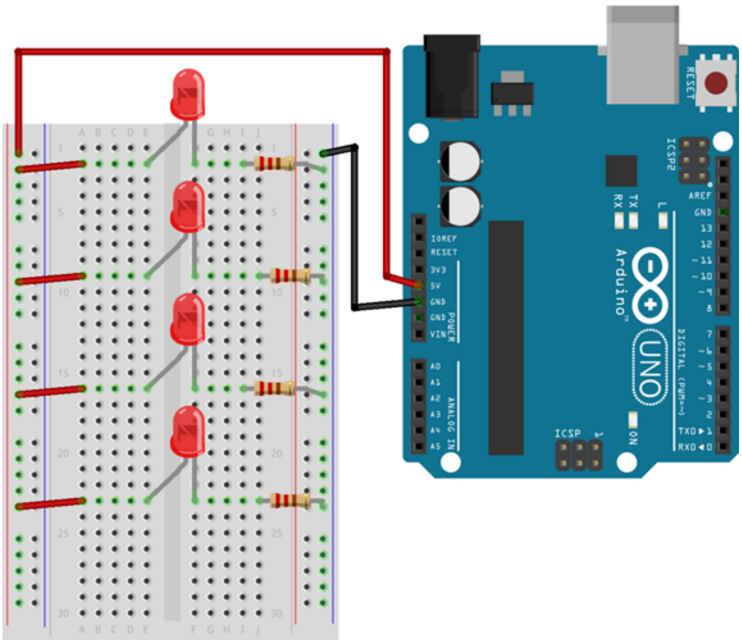


Figure 3.30 Wiring LEDs in parallel allows you to wire more in a single circuit because each branch gets the same voltage (5 V).

If everything goes right, all four LEDs should happily light up.

Parallel circuits have another useful feature. If you were to remove the red wire between the LED in row 2 and the power rail, the other 3 LEDs would still light up. The circuit still has three other complete paths it can use. Contrast this to a series circuit, where a single gap stops current from flowing to any component.

3.2.6 *Powering your project with batteries*

So far, you've been providing power to the breadboard using the Arduino Uno's onboard 5 V power, but there are other ways to provide power to projects. One (obvious) option is to use batteries.

A single, 9 V battery is a convenient power source, and in the case of the LED circuits you've been building, it removes the reliance on a development board.

BUILD THE CIRCUIT: 9 V POWERED LEDS

What you'll need

- 1 breadboard
- 9 V battery and snap, with wires
- 4 standard red LEDs
- 4 jumper wires
- 4 560 Ω resistors

With a 9 V battery, the supply voltage is (obviously) different than the Arduino's 5 V. That means we'll need to use different resistors for the LEDs in the parallel circuit.

Recall that each branch in a parallel circuit "gets" the full 9 V supply voltage to work with, so you can calculate the resistor needed on each branch using Ohm's law:

```
R = 9 V / 20 mA
R = 450 Ω
```

A 560 Ω resistor is the nearest common resistor value, and it will do fine (have you noticed there's a bit of fudge room, as long as you round up?).

Disconnect the breadboard's power rails from the Arduino and swap out each of the 220 Ω resistors in favor of the brawnier 560 Ω resistors. Now plug the battery case's positive and negative wires into the breadboard's power rails (figure 3.31). All done!

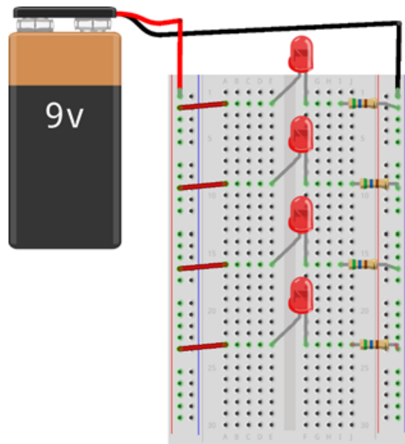


Figure 3.31 Powering the parallel LED circuit with a 9 V battery involves swapping out some resistors and connecting a 9 V battery snap to the power rails.

Summary

- The relationships between voltage, current, and resistance—as formalized in Ohm's law—are the keys to understanding basic circuitry.
- Breadboards provide a tangible and convenient prototyping platform, with standard connection patterns, for trying out circuits.
- A series circuit provides one single path for electrical flow. A parallel circuit has two or more paths.
- LEDs have a characteristic called *forward voltage drop*. When calculating the right resistor for an LED wired in series, first subtract this forward voltage drop from the supply voltage.
- In a series circuit, the current is equal at all points of the circuit, whereas in a parallel circuit, all branches have equal voltages.

- In a series circuit, it's straightforward to calculate total resistance in the circuit: add the resistor values together.
- A circuit with more than one path splits up the current in the circuit and is called a *current divider*. Determining the total resistance in such a parallel circuit can be accomplished using a current-division formula.

JAVASCRIPT/HARDWARE

JAVASCRIPT ON THINGS

Hacking hardware for web developers

Lyza Danger Gardner

Free eBook
See first page

Are you ready to make things move? If you can build a web app, you can create robots, weather stations, and other funky gadgets! In this incredibly fun, project-based guide, JavaScript hardware hacker Lyza Danger Gardner takes you on an incredible journey from your first flashing LED through atmospheric sensors, motorized rovers, Bluetooth doorbells, and more. With JavaScript, some easy-to-get hardware, and a bit of creativity, you'll be beeping, spinning, and glowing in no time.

JavaScript on Things introduces the exciting world of programming small electronics! You'll start building things immediately, beginning with basic blinking on Arduino. This fully illustrated, hands-on book surveys JavaScript toolkits like Johnny-Five along with platforms including Raspberry Pi, Tessel, and BeagleBone. As you build project after interesting project, you'll learn to wire in sensors, hook up motors, transmit data, and handle user input. So be warned: once you start, you won't want to stop.

What's inside

- Controlling hardware with JavaScript
- Designing and assembling robots and gadgets
- A crash course in electronics
- Over a dozen hands-on projects!

Written for readers with intermediate JavaScript and Node.js skills. No experience with electronics required.

Lyza Danger Gardner has been a web developer for over 20 years. She's part of the NodeBots community and a contributor to the Johnny-Five Node.js library.

To download their free eBook in PDF, ePub, and Kindle formats, owners of this book should visit www.manning.com/books/javascript-on-things

"JavaScript breaks free of the browser with this book!"
—Amit Lamba, Tech Overture

"The most accessible and enjoyable book on IoT tech I have ever read."
—Andrew Meredith
Quantum Metric

"Calling all developers! Unlock your inner hardware hacker with this exciting book."
—Kevin Liao, Sotheby's

"Makes developing low-level circuits fun. A pleasure to read!"
—Earl Bingham
You Technology

ISBN-13: 978-1-61729-386-3
ISBN-10: 1-61729-386-5
53999
9 781617 293863

manning US $39.99 | Can $52.99 [Including eBOOK]