

Electronics for Contraptions

One of my favorite things to do with computers is interact with the world in interesting and unexpected ways. A few years ago connecting things to computers required an engineering degree, but recently released products have made it fairly easy. Devices like Arduino boards do all of the tricky interfacing for us, and we merely have to build the physical controllers. This essay gives tips for doing that.

Essentials of Current

You do need to know a few basics of electronics or something could be damaged. Here's the bare minimum:

The Simplest Circuit:

We can make current flow in a circle (circuit) by connecting the terminals of a battery together. This will melt the wire, make sparks fly and maybe start a fire, so don't do it. Instead, connect something to control the current. The ability to control current is called resistance, and all materials have it to some degree--in fact we classify materials according to their resistance: those with very low resistance are conductors, those with lots of resistance are insulators. There are devices called resistors that are used in electronic gadgets--they have a resistance that is predictable. So figure 1 is a safe circuit.

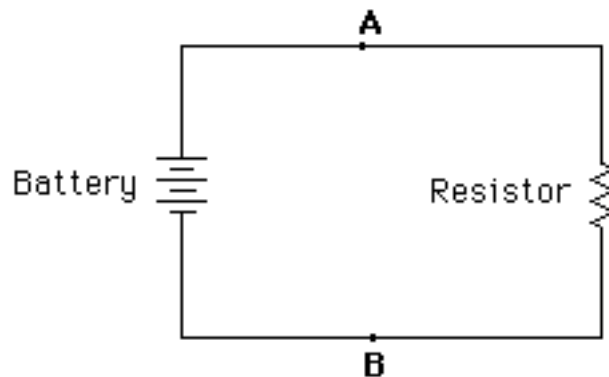


Figure 1.

The battery has a certain amount of push, called **electromotive force** or **EMF**. This is measured in units called **volts**. We indicate EMF (most commonly called **voltage**) in formulas with the letter **E**. Voltage has to be measured between two points in the circuit in the same way that height has to be measured between two points on the side of a mountain. There is no such thing as "0 volts" except that the voltage between two points is 0 if they are connected together. We put a sign with voltage to indicate the direction of current. The sign is sort of arbitrary-- + 9 volts just means we are looking at the + end of the battery. The electrons that make up current flow from - to +. Electrical engineers usually think of current as flowing from + to -.

Resistance is the opposition to the steady flow current. Everything in the circuit has resistance, but for convenience, we pretend all of the resistance of a circuit is in one device. The resistor has a specific amount of resistance measured in units called **ohms**. We indicate resistance in formulas with the letter **R**, and on drawings with an omega symbol (Ω).

When current is flowing, we measure it in units called **amperes**, and indicate it with a letter **I**.

The three are related by a simple formula called Ohm's Law

$$\mathbf{I=E/R}$$

Also written $\mathbf{E = IR}$ or $\mathbf{R=E/I}$.

It may be easier to remember if I say:

$$\text{Current} = \frac{\text{Voltage}}{\text{Resistance}}$$

That tells us the current if we know the voltage and resistance, the voltage if we know the current and resistance, or the resistance if we know the current and voltage. If this seems a bit circular to you, you're right. We can measure current by the strength of magnetic field it will generate, but there is no yardstick for voltage other than seeing how much current flows through a known resistance. And how is a resistance known? We apply a known voltage and see how much current flows¹.

The definition of the units is circular too: 1 ampere is the amount of current that flows through a 1 ohm resistor if 1 volt is applied. Finding how much current will flow with a specified voltage and resistance is the most common calculation in electronics work.

Power

There's another calculation we need to do nearly as often. The point of pushing electricity through a wire is usually to get some work done. This requires power, which is related to the current and voltage. In fact, the power is the current times the voltage.

$$\mathbf{W=IE}$$

¹ We try to keep current flow low, unless we are building toasters. Thus, most circuits are designed for currents of a few thousandths of an amp. The unit prefix for thousandth is milli. Currents of a milliamp (ma) usually imply resistances of 1000 ohms, or a kiloOhm (k Ω). 2k = 2000 Ohms.

Power is measured in watts (w), milliwatts (mw), or kilowatts (kw). We encounter power calculations in two contexts: power required to do a job (such as light a room) and the ability of electronic components to handle power. This last is known as power rating. If we try to run more power through a component than it is rated for, it will probably fail. We discover a device's power rating by looking at the specifications. Often this is given as a current limit at a rated voltage, but it is just common to see a rating in watts² and we have to do the math. Either way, we must know the limits of our components.

Two resistors in series:

The basic Ohm's law calculations get a bit more complicated if there are two resistors:

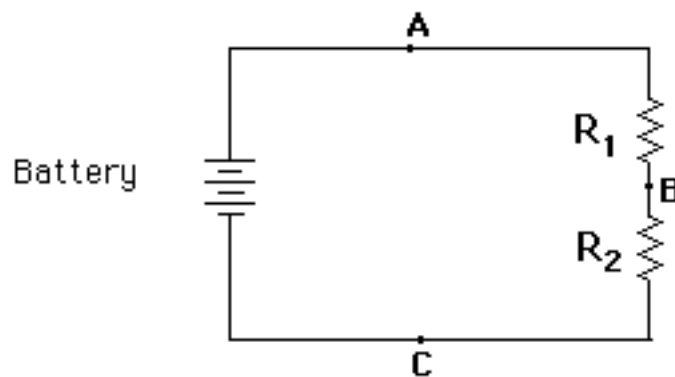


Figure 2.
Whatever the current is, it's the same at A, B, and C. (There isn't anywhere else for the current to flow.)

The voltage between A and C in figure 2 is equal to that between A and B added to that between B and C.

$$E_{AC} = E_{AB} + E_{BC}$$

The voltages add up, just as the height³ of a house is the sum of the heights of its stories.

The voltage across each resistor is proportional to the resistance of each resistor.

$$E_{AB} / R_1 = E_{BC} / R_2$$

Ohm's law is true for each part of a circuit as well as the circuit as a whole. The current in figure 2 is the same in each resistor, so the voltages will adjust themselves. This circuit is called a voltage divider. What we really want to know most of the time is the voltage at B:

² Or VA for volts times amperes.

³ The height metaphor is common enough that engineers often speak of "voltage drop".

$$E_{BC} = E_{AB} * R_2 / R_1 + R_2$$

The total resistance of figure 2 is $R_1 + R_2$.

Two Resistors in Parallel

When resistors are connected side by side, the math gets trickier;

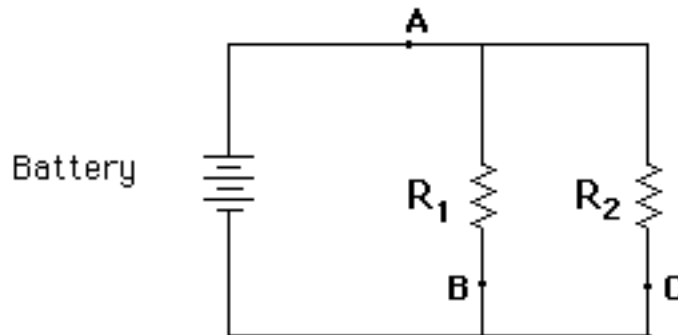


Figure 3.

The current through A in figure 3 equals the current through B plus the current through C. The current splits up and comes together, like water flowing around an island.

The voltage across R_1 is the same as the voltage across R_2 .

$$E_{AB} = E_{AC}, \text{ so } I_B R_1 = I_C R_2 \text{ and } I_B / R_2 = I_C / R_1$$

In other words, the current through each resistor is inversely proportional to the values of the resistors. The point to remember is a high value resistor passes a small current.

We can solve the above for total current ($I_B + I_C$) and get the equivalent resistance for the two resistors:

$$\frac{1}{\frac{1}{R_1} + \frac{1}{R_2}}$$

In the special case where resistors are the same, the equivalent resistance is $R_1/2$. This turns up more often than you might expect.

In another special case, where R_2 is more than 100 times the value of R_1 , R_2 accounts for such a small portion of the current that we don't bother to include it in the calculations. Then we say R_2 does not load the circuit.

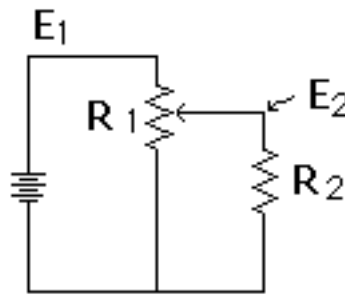
A more complex example:

Figure 4.

In figure 4, R_1 is a special type of resistor that has an adjustable tap in the middle⁴. This really makes R_1 behave like two resistors in series. If we say R_2 is 100 times R_1 , we can leave it out of the calculations, and find that the voltage E_2 will vary directly with the position of the tap.

If R_2 were comparable to R_1 in value, we would have to figure it in, first solving R_2 and the bottom part of R_1 as two resistors in parallel, and using the result of that in a series calculation to find the voltage E_2 and the total current. The resulting voltage curve makes a control behave oddly, so we really prefer to have an R_2 that does not load the circuit. A good rule of thumb is to keep R_2 10 times the value of R_1 . If R_2 is some complex device, such as the input pin of an arduino, the effective value is specified as the *impedance* of the input. A low impedance input is 10k or less and a high impedance input would be 100k or greater.⁵

Changing Currents

Current is not always flowing, and when it flows it may often reverse and flow in the opposite direction. We generally use changes in current to convey information such as audio signals. The study of changing currents (often called AC currents) is not really necessary for contraption building, but there are a few things that are useful to know. Some devices react in odd ways when the current through them is changed. Inductors (coils of wire around a magnetic core), for instance, develop a magnetic field when current is run through them, but when the current is stopped, the collapsing magnetic field generates extra current that flows for a moment after the source is removed. When the current is first applied, it takes a while for the field to build up, so the flow of current is late getting started. This is fascinating and immensely useful for engineers designing high frequency circuits, but the main effect on our contraption building is devices that have coils in them (motors and relays) will occasionally produce unexpected spikes of current.

⁴ It's called a potentiometer, or pot.

⁵ Arduinos have a high input impedance unless the pull-up resistors are engaged.

Capacitors

Another device that reacts oddly to changes in current is the capacitor⁶. This is made of two conductive elements separated by an insulator. Think of a stack of three pie pans, with the outer ones made of metal and the center one glass. When a voltage is applied across the two metal pans, some electrons will be pulled out of the glass toward the positive pan, and a similar number of electrons will be stuffed into the glass on the negative side. There is a limit to the number of electrons that can be redistributed this way, which is called the capacitance of the device. When this limit is reached, the capacitor is fully charged. If the two metal pans are then wired directly together, current will flow from one to another, in the opposite of the original direction. When the current is first applied to a discharged capacitor, a large current will flow, as if the resistance were small. As the capacitor charges up the current decreases as if the resistance were increasing. Ohm's law shows the voltage increases in a curve that levels off.

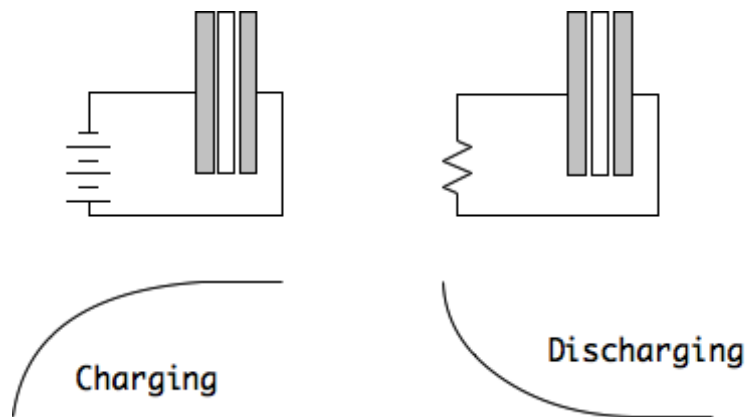


Figure 5.

We can take advantage of this effect to create a circuit where the voltage changes in a predictable amount of time. If a resistor is put in series with the capacitor, the time it takes the voltage at the point between the two to increase to 63% of the applied voltage is equal to RC , where C is capacitance in Farads and the time is in seconds. This time is known as the time constant of the circuit. We generally use capacitors in two ways: DC blocking and voltage smoothing. If you apply an AC signal to one side of a capacitive circuit with a long time constant, the AC component of the signal will appear to pass through the capacitor, but any DC component will be removed, leaving a signal that has equal positive and negative swing. We use capacitors this way to separate parts of a complex AC circuit.

⁶ Also known as a condenser.

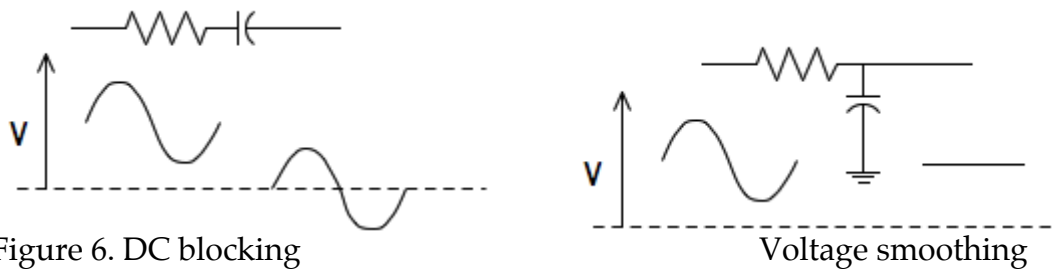


Figure 6. DC blocking

Voltage smoothing

If you connect the capacitor to ground, making a voltage divider, the voltage at the connection between the resistor and capacitor will be the average voltage with the signal removed. We do this in power supplies, where we must convert AC to a steady DC.

Impedance

Inductors and capacitors both obey Ohm's law, but you can see that while the current is changing Ohm's law is bent a bit. In fact, if the current is constantly changing direction, as in an audio signal, the voltage / current relationship is quite different from what you get with steady (DC) current. We account for this with the concept of reactance, defined as opposition to changing current. Reactance depends on the frequency of the current change. The reactance of an inductor is low for low frequency signals and increases with frequency. This makes sense when you remember an inductor is a continuous coil of wire. The reactance of a capacitor is high for low frequencies and decreases with frequency. This also makes sense-- the two plates of the capacitor are not actually connected and current only flows during the charge or discharge.

The combination of reactance and resistance is impedance. There are formulas for calculating this, but we won't be using them. All we need to know is that AC signals obey Ohm's law when impedance is used instead of resistance, and the frequency must be specified to know the impedance. The symbol for impedance is Z .

$$I = E / Z$$

Semiconductors

Semiconductors are devices that can carry current but don't follow Ohm's law. They come with a wide variety of exceptions and loopholes: current proportional to temperature, current proportional to light, and especially current proportional to other sources of current.

Diodes

The simplest semiconductor is the diode. Diodes only conduct current in one direction. This feature allows us to separate parts of circuits.

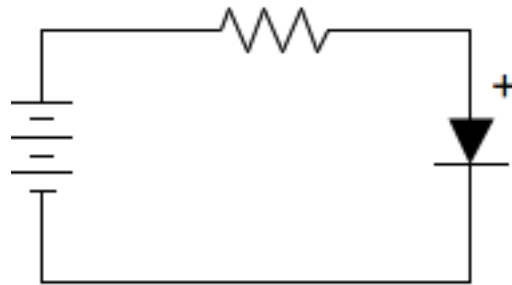


Figure 7.

The symbol for a diode is an arrow with a bar across it. When it is hooked up so the end with the bar (the cathode) is more positive than the end without the bar, (anode) no current will flow. When the hookup is reversed, current will flow, if the voltage difference is larger than a small amount called the forward voltage drop. The drop is typically about 0.7 V. Diodes have two important rating numbers- current capacity and reverse breakdown voltage. Exceed either value and the diode is cooked. Semiconductors are identified by a device number that often is a combination of letters and numbers. A good general purpose diode is a 1N148.

Transistors

The transistor is the gadget that revolutionized electronics and our entire way of life. A transistor has three leads called the collector, emitter and base (c e b). The basic principle of operation is that the current through the base and emitter controls the current between the collector and emitter.

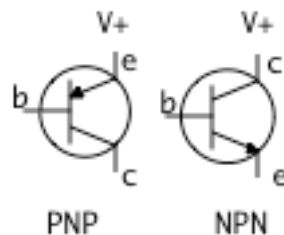


Figure 8.

Transistors come in two types called NPN and PNP. The difference is in the current flow⁷. For a PNP transistor, the emitter must be more positive than the collector, and the base voltage must be between the emitter and collector. For the NPN transistor, the collector must be more positive than the emitter. When the base voltage is close to the emitter, little current flows. (None if the difference is less than the voltage drop.) As the base voltage approaches the collector, collector current increases.

Transistors are used in two modes. In digital mode they serve as switches, with current either flowing or not. We don't worry about how much, as long as there

⁷ The arrow on the emitter shows this, if you think of current as flowing from positive to negative.

is enough to control the next transistor along the line. In analog mode, resistor networks are added to keep the currents in the range where the collector current is nearly proportional to the base current. This range is rather narrow, so you need several transistor stages to get a usable degree of amplification. It's usually simpler to use an integrated circuit when analog operations are required.

Transistors have several key ratings, but these two are vital:

- $H_{fe}(\text{min})$ is the minimum current gain
- $I_c(\text{max})$ is the maximum current it can take⁸.

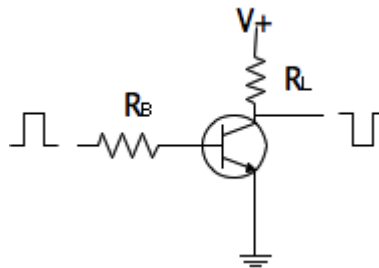


Figure 9.

Figure 9 shows an NPN transistor⁹ used as a switch. The resistor marked R_L represents the load, whatever the switch is controlling. This hookup is inverting, in that the collector voltage goes low when the base voltage goes high. When the collector goes low, current will flow through the load. The resistor marked R_B will limit the current through the base. It should be about $0.2 \times h_{fe} \times R_L$, but it may be omitted if the transistor is connected directly to an integrated circuit with output current limiting (like the Arduino).

Integrated Circuits

I seldom use more than a couple of transistors in a circuit. Instead, I reach for an integrated circuit. ICs are complete circuits designed to solve common tasks. Probably the most useful for the contraption builder is the microprocessor, such as the Atmega series of chips used in Arduino boards. Various ICs will be covered in the following applications, but there are some issues to consider with any of them. These include packaging, power supply range and available current.

Packages

Integrated circuits can have three to a hundred pins. Three pin packages look a lot like transistors. Often they have a metal tab as shown in figure 10.

⁸ Big transistors have two ratings, one if a heat sink is used, a lower one if not. A heat sink is a chunk of metal the transistor can be bolted to.

⁹ My favorite transistor for this trick is the 2N3904.



Figure 10.

ICs with 8 to 40 pins are usually available in DIP (dual in-line package) style, and bigger ones have ever more complex designs requiring specialized sockets. One package to avoid is surface mount (SMD) style. These are designed for machine made boards, and are practically impossible to solder in by hand.

Power

ICs may require as little as 3 volts or as much as 24. If you are working with Arduino, look for chips that work with 5 volts. One design to avoid is the bipolar type. These are op amps and other analog chips that require both positive and negative power. (Bipolar chips are preferred for analog design, but that's not what this essay is about.)

Amperage revisited

When we connect something to an output pin on an integrated circuit, the situation is similar to figure 2. (Two resistors in series.) R1 is in the device and we need to figure out a safe value for R2. In the real world, we are told the voltage available and instructed not to draw more than a certain amount of current. Look in the specs for the interface device for "terminal output current" or a similar item. For the Arduino, this is 40 ma. Using the 5 volts supplied by the device and Ohm's law from page 2 we get $5/0.04$ or 125 ohms for the minimum resistance we can use on an output terminal. In fact there is an overall limit of 150 ma for the entire device, so we need to use larger values if many outputs are used¹⁰.

When we connect to an input pin we are building a circuit like figure 4, where we connect an R1 of our choice to a voltage E1 and the terminal measures E2. The terminal includes R2, so there are two considerations. R1 should be large enough to avoid drawing too much current from the 500ma available, and should be small enough to avoid loading effects. Luckily the terminals have very high R2 so values up to 1 Megaohm will work. 10 k is common in the examples, but we can use practically any value above 10 ohms. **What we cannot do is connect a pin to an outside voltage source less than 0 or greater than the power supply (5 volts on Arduino)!**

Amplifiers

We have seen how to reduce the voltage in a circuit, but just as often we need to increase it. Any increase in voltage or current requires gain. Gain means the output of the circuit is greater than the signal input. (The extra power is provided by a direct connection to the power supply.) A circuit that provides gain is an

¹⁰ The outputs are electrically isolated, so we don't need the complicated resistance in parallel formula. Just calculate the current per output and add them up.

amplifier. The symbol for an amplifier is a triangle with the input(s) on the wide side and the output on the opposite point. (Power supply connections are often left out of the drawing but are always necessary.)

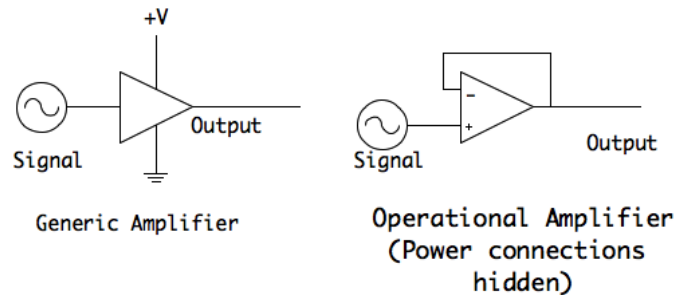


Figure 11.

Figure 11 shows two typical amplifier diagrams. These things actually come in plastic packages with many connections-- there may be several independent amplifiers or just one with some options. Exactly how the chip is connected will depend on the particular IC, so always get a data sheet when you work with a new one.

The Operational Amplifier or Opamp is the most common amplifier chip. It has two inputs, and amplifies the voltage difference between them. The current output will be determined by Ohm's law-- the opamp establishes E and the R is whatever is connected to the output. If too much current is demanded, the opamp will shut down or fail. One input is marked with a minus sign and called the inverting input. (A positive voltage applied to the inverting input results in a negative voltage at the output.) The other input is non-inverting and marked with a plus. There are thousands of applications for opamps, but all I will use here is the simplest case where the output is connected back to the inverting input. This gives us unity voltage gain (the output matches the input) but plenty of current. This configuration is also called a voltage follower.

The one vital rule when using amplifier ICs is simple: **When you connect the positive and negative power pins, you establish a voltage range for the chip. Never connect any pin directly to a voltage above or below this range¹¹.**

Connecting Controls and Sensors:

The devices that a user interfaces with are called controls or sensors. Almost all of these turn out to be either a potentiometer or a switch.

Switches

Switches are used for digital input. They are simple mechanical or electronic devices that make or break a connection between two terminals.

¹¹ Some chips have two positive power pins. In that case some pins will be limited to the lower voltage, others to the higher voltage. Consult the data sheet.



Figure 12. Various switches

Switches may be the finger operated kind, or may be actuated by a magnet, light, or various lever arrangements. Switches may have several switch elements or poles, and may have more than 2 positions. Switch action is sometimes called "throw", so a simple on-off switch would be a single pole, single throw or SPST switch. A double throw switch is connected to something in both positions, and may have a center detent that is not connected. Even more positions are found on rotary switches. Switches may also be momentary, springing back when the pressure is released. Such switches (usually buttons) are indicated as "normally open" or "normally closed", and may also have multiple elements.

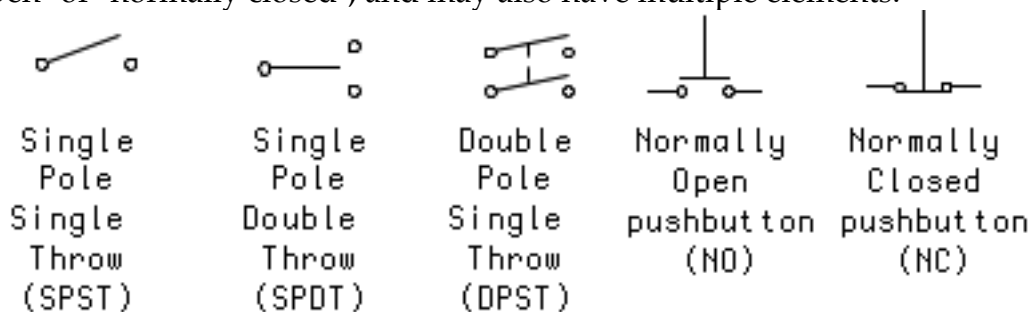


Figure 13. Some typical switch configurations.

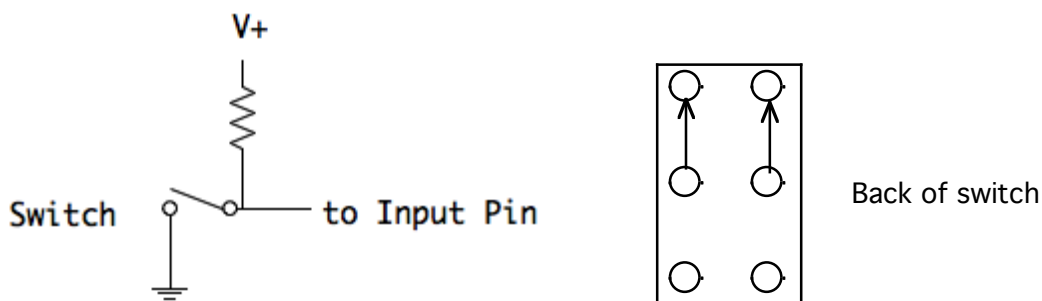


Figure 14. Connecting a switch to an Arduino pin

Figure 14 shows the basic digital input circuit. When the switch is open, the resistor pulls the voltage to the positive supply, which is definitely on. Closed, the ground connection makes it definitely off. Most switches have more connections than we need: only two are used for digital sensing. You will have to experiment a bit with a resistance meter to discover just which pins are connected when the switch moves.

Solid State Switches

Solid state switches are based on transistors. They are useful when you want to detect effects other than button pushes.

A *phototransistor* conducts when it is exposed to light. Here is a typical hookup for an NPN style phototransistor. The collector and emitter are just like any transistor, current flows from collector to emitter. The base function is served by the light input in the phototransistor. The leads may be marked by letters (E or C the base is light input) on the case, or you may need to refer to a data sheet to figure out which is which. If you get the emitter and collector mixed up, you will damage the device.

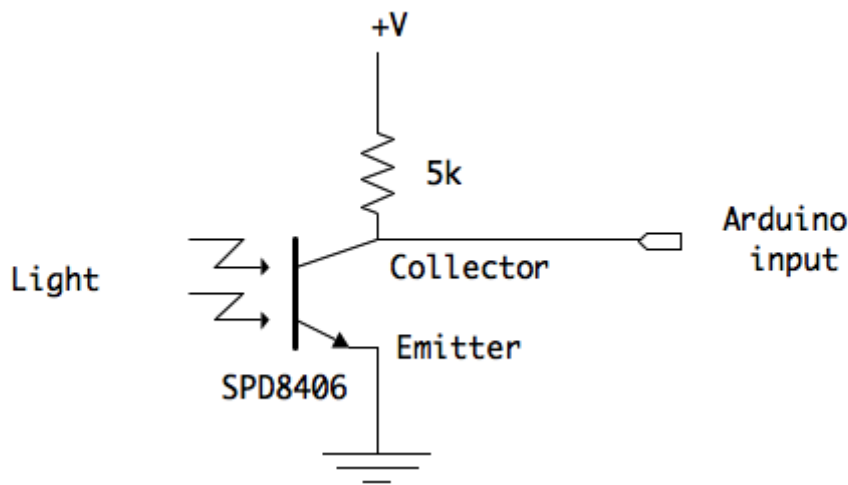


Figure 15. Phototransistor Circuit (NPN)

We often use switches triggered by infrared light for proximity detection. There is a phototransistor at the heart of such devices, but a naked phototransistor has such a broad area of sensitivity that we can't get the precise location we want. I generally use commercial motion detectors for this. The devices are available in a wide range of sensitivities and detection angles, and interface to the Arduino as a simple switch.

A *Hall Effect transducer* is a device that is triggered by a magnetic field. It has three pins, power, output and ground.

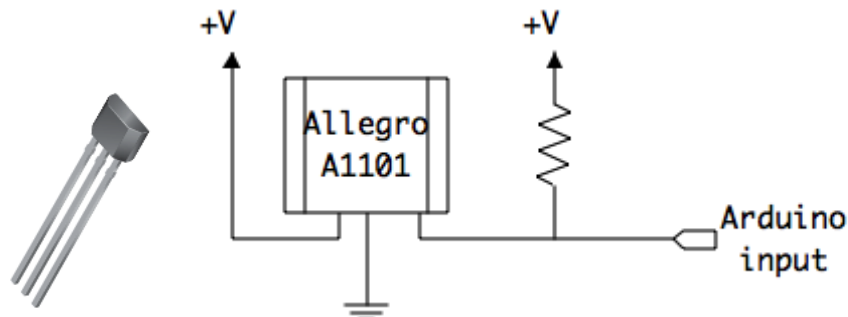


Figure 16. Hall effect circuit

When the south pole of a magnet is brought close to the back of the sensor, current will flow, making the output wire fall to ground. Hall effect switches are very fast, so you can use them to measure the speed of a motor. Figure 17 shows a contraption that follows the motion of a wheelchair.

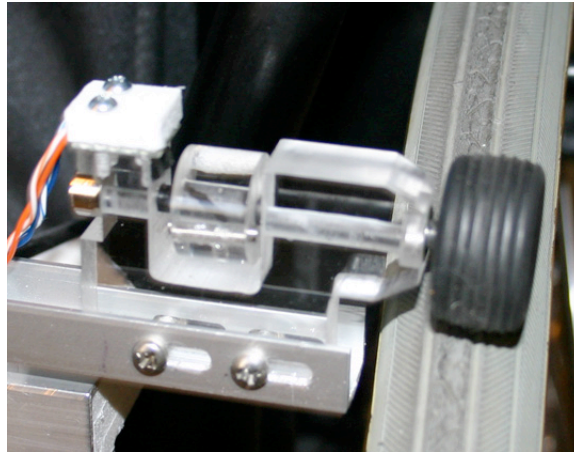


Figure 17.

There are magnets in the plastic rotor, and two hall effect sensors sticking down from the white piece on the left end.

Potentiometers

The analog inputs of the Arduino measure voltage in the range of 0 to 5 volts. The easiest way to generate this is with the potentiometer circuit of figure 4.



Figure 18.

The most common potentiometers are either rotary or slide types, but there are others that are turned with a screwdriver, or respond to flexing, pressure or nearly any kind of motion. In most cases, there are three pins, one at each end of the resistive element and one for the movable tap or wiper. A little bit of experimentation with a meter will identify them.

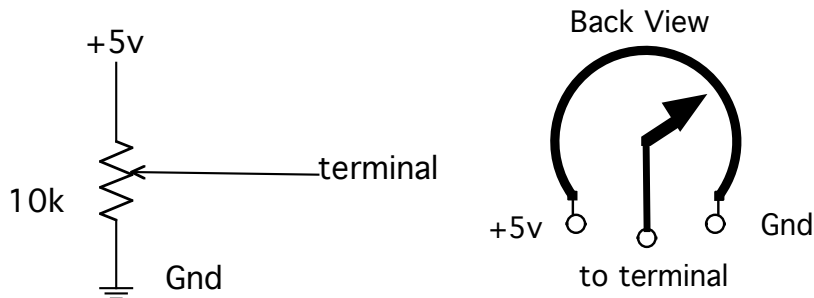


Figure 19.

Figure 19 shows how potentiometers are hooked up. The general rule is that the impedance on the wiper must be 10x the resistance of the pot. Since the input impedance of the Arduino pin is quite high, the actual value of the potentiometer is not important, as long as we don't overload the power supply. 10k is a common value that will draw half a milliamp at 5 Volts.

Sometimes we connect the wiper of a potentiometer to one of the end terminals to produce a variable resistor. One situation where this is useful is with a *flex sensor*. A flex sensor is a device with resistance that increases as it is bent. The most common type has a resistance of 10k when it is straight and somewhere between 60k and 110k at maximum flex. We can get a voltage for the Arduino by using a flex sensor as the bottom half of a voltage divider as in figure 20a. With a 100k resistor as shown, a sensor with a maximum resistance of 100k would show the Arduino a range of 0.45 to 2.5 volts. However, a different sensor might only have 60k max resistance and the circuit could only produce 1.875V. In the circuit of 20b, the top half of the voltage divider can be adjusted¹² to match the maximum resistance of the sensor, so the Arduino would again get 2.5V at full flex. In the worst case, the low end of the range would be 0.7 volts, but this is still the widest range¹³ available with that sensor.

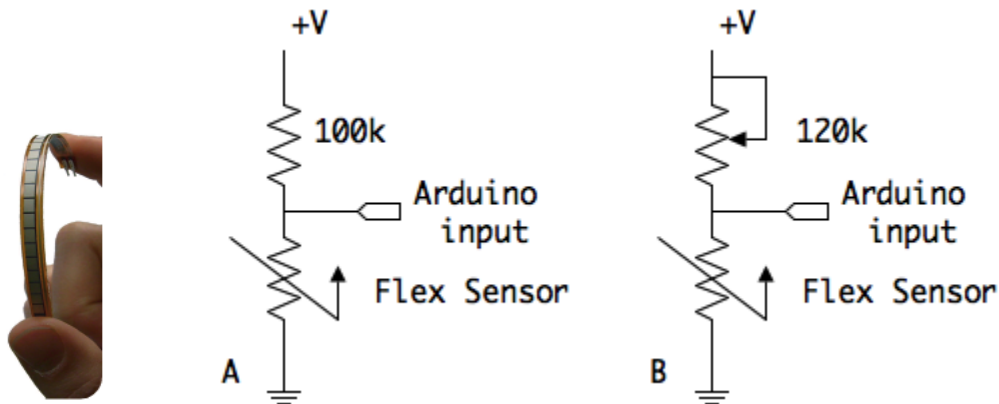


Figure 20.

¹² This is a good place to use a screw adjustable potentiometer.

¹³ Getting a 0 to 5V range requires a circuit that uses negative voltage. It can be done, but is more trouble than it's worth.

Other Analog Sources

There are other devices that produce variable voltages suitable for the analog inputs. For example, the Maxbotix ultrasonic range finder. This produces a voltage proportional to the distance to an object. Many such devices can be connected directly to an Arduino pin if the voltage range is guaranteed not to exceed 0-5v. A voltage range larger than 0-5v can be reduced with a voltage divider (figure 4.). Remember the formula for the middle voltage:

$$E_{\text{out}} = E_{\text{in}} * R_2 / R_1 + R_2$$

The ratio of the resistors determines the voltage reduction, and the total of the resistors determines the current draw on the source.

Protecting the analog inputs.

If there is any concern that the input to an analog pin may go below 0 or exceed 5 volts, the input can be protected by a Zener diode. This is a diode with a specified breakdown voltage, and is beefy enough that it does not mind being broken down.

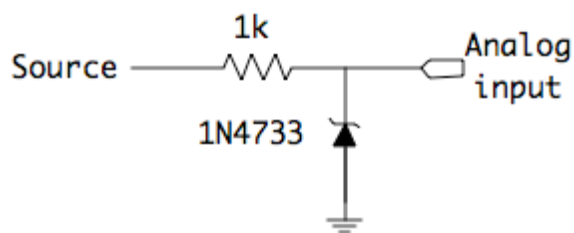


Figure 20.1

The Zener is connected as shown in figure 20.1. Normally, the diode does not conduct, because it is backwards to a positive voltage from the source. If the source goes negative (below 0 v) the diode will conduct and the input will be clamped at ground (or close enough to do no damage). If the source exceeds the Zener voltage (5.1v for the 1N4733) the diode will conduct backwards and the input will be clamped to the Zener voltage. A circuit like this should be included if the voltage source is at all unpredictable, such as a piezoelectric pickup or anything with a coil.

Input Amplifiers

The high impedance of the Arduino input makes it easy to use, but it is also susceptible to noise pickup, especially if the wire from the sensor to the Arduino is longer than a couple of feet. In that situation the value measured will jump around enough to affect the process. If the Arduino is controlling a light, for instance, the input noise would make the light flicker. Some of this can be fixed in the software, but the most effective solution is to add a low impedance buffer. This is built from an opamp as shown in figure 21.

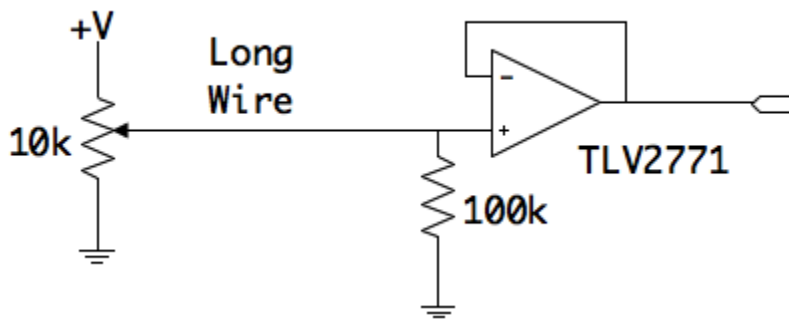


Figure 21.

The opamp and 100k resistor are placed close to the Arduino card. They don't take up much space-- you can get four opamps in a 14 pin package. The potentiometer can be located many feet away. Of course the +V power and ground also have to be long wires, but these are inherently low impedance and won't add noise to the system. I often use 8 wire CAT5 cable for these runs, which allows 6 controls in a remote box.

Some sensors won't work well connected to a low impedance input. If so, an opamp located near the sensor can add enough drive to prevent noise problems. The opamp is wired as a voltage follower, so it does not change the operation of the sensor in any way. (Remember, the power connections to the opamps are hidden in these drawings. Consult the data sheet to see how to hook them up.)

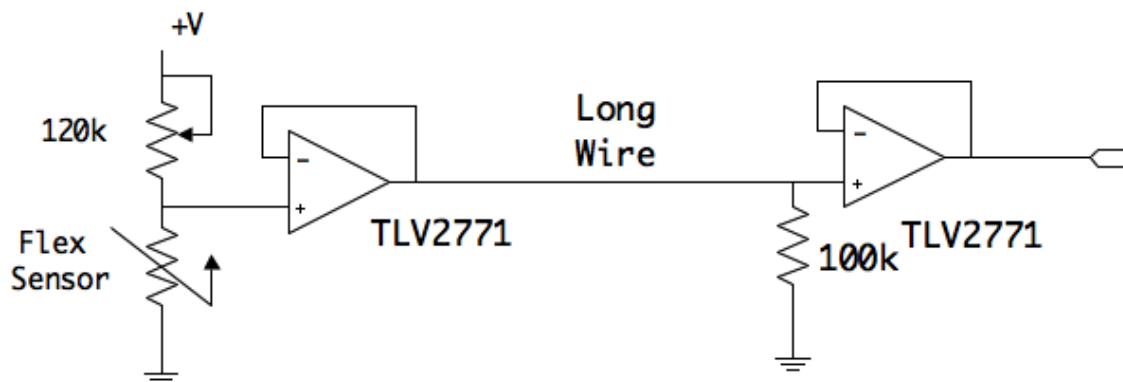


Figure 22.

There are hundreds of opamps available, all with different specifications. The key features for our applications are the ability to run on single sided 5V power supply, rail-to-rail operation¹⁴, and low cost. The Texas Instruments TLV2771 or TLV2772 (dual package) are an excellent choice.

¹⁴ Many opamp outputs will not go down to ground or up to the full power supply.

Output from the Arduino

LEDs

If an output pin blows out with 40 ma current, prudent design practice is to plan for half of that. 20ma is not a lot of output current. It's enough to light an LED, so the circuit of figure 24 on an output makes sense:

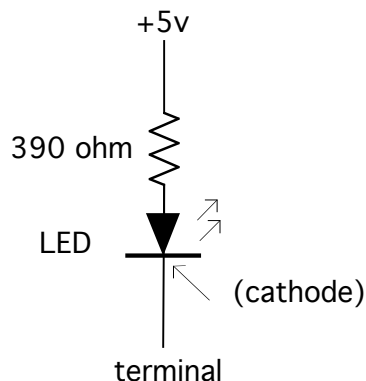


Figure 23.

When the terminal goes to 0, the LED will light up. With a 390 ohm resistor, the circuit will draw less than 10 ma, because the LED steals 1.6 volts from the total¹⁵. This allows as much current as the LED will stand. Datasheets for the LED should be available from the supplier. The maximum current rating will be prominently displayed on the datasheet, and possibly a recommended current. (80% of maximum is a good choice.) The formula for calculating the resistor value needed to get a specified current is $R = (\text{voltage} - 1.6) / \text{current}$. You can light 2 LEDs by putting them in series, but the voltage drop is doubled.

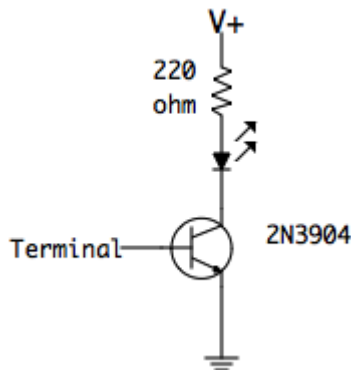


Figure 24.

Even though a single Arduino pin will provide plenty of current to light an LED, the total current for the Arduino is limited to 200ma. If you need to drive a lot of

¹⁵ This is the forward voltage drop. It can be as high as 3.4 V with big LEDs.

LEDs, add a transistor switch as in figure 24. When a transistor is used this way, the LED will light when the terminal is positive.

The LED will only work if it is installed right way around. The bar (cathode) corresponds to the shorter wire. If the wires have been cut, there may be a flat spot on the cathode side as illustrated in figure 25¹⁶.

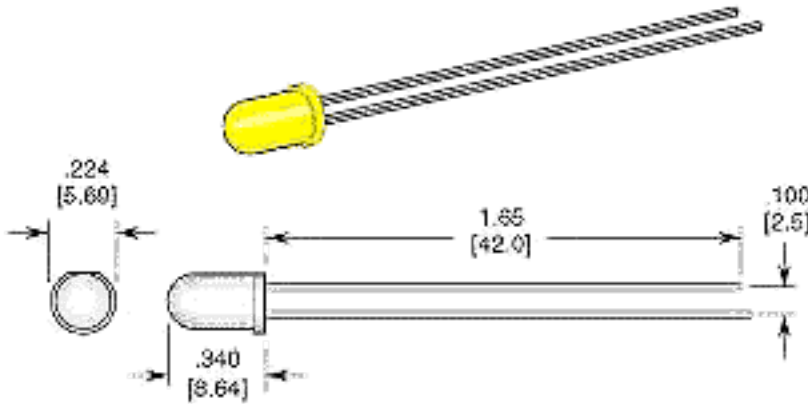


Figure 25. Typical LED

Driver Transistors

For bigger loads, you need an external current source, which could be a wall wart style DC power supply or a battery. A simple IC called a darlington transistor pair will allow the Arduino to switch this current.

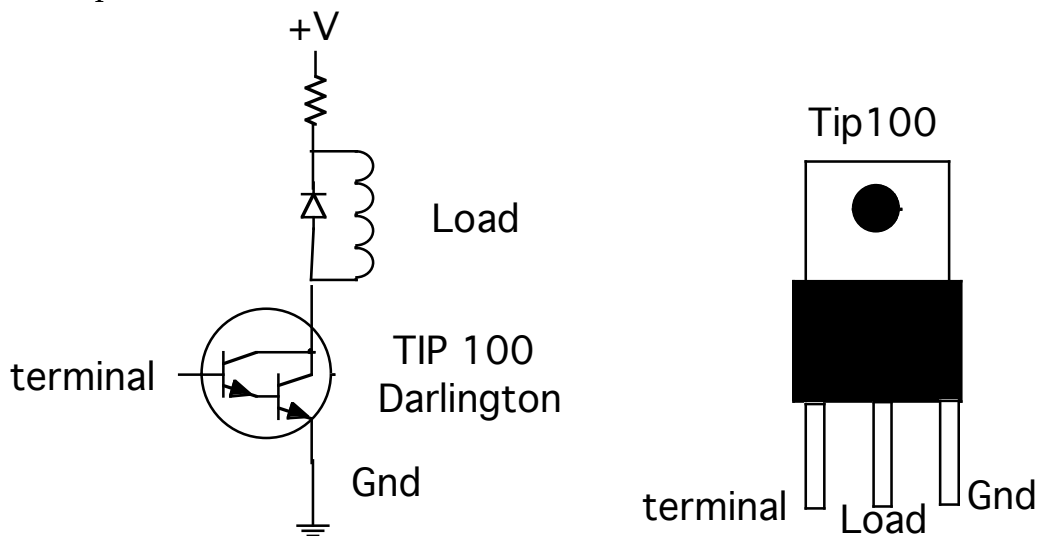


Figure 26.

The TIP 100 will switch up to 8 amps. The resistor should be sized to provide the proper current with the voltage used. The ground must be connected to both the ground of the Arduino and the extra voltage source. The load shown is an

¹⁶ Otherwise, test it with a meter. An LED will conduct (showing low resistance) when the black lead of the meter is connected to the cathode.

inductor, which could be a relay or a motor. Any inductive load needs a diode¹⁷ as shown to protect the transistor from backwards voltage spikes. If you are just powering lights, the diode is not needed.

Note: The TIP family actually has a lot of members, some of which are NPN, some PNP. Make sure you use the proper type-- the TIP100, TIP102 and TIP120 are NPN, TIP125 and TIP127 are PNP.

Relays

For really big loads such as AC powered equipment, relays are the safest approach. Relays can be mechanical or solid state. Mechanical relays usually need 12V or 24V for control, so they would be driven by a TIP. (Figure 26.) Solid state relays that are controlled by 5V can be connected directly to an Arduino pin. Wall grade power can be dangerous if you don't know what you are doing, so I'm not going into any detail here. If you need to deal with industrial loads, it is worth your time and money to investigate DMX controlled systems. DMX is covered in another tutorial.

Connecting Motors

Small motors come in three types, DC, servo and stepper. Making any motor work is going to require study of the motor specs and a fair amount of experimentation.

DC Motors

DC motors are familiar to anyone who has worked with electric trains. When a voltage is applied, the motor turns at a rate somewhat proportional to the voltage. Only somewhat; at low voltage the motor stalls, and at the maximum voltage the motor turns at the rated RPM. In between, as voltage increases, the motor speeds up. The motor will change direction if the polarity of the voltage is reversed. The current a motor draws depends on the voltage applied and the speed of rotation¹⁸, which is dependent on the work (load) the motor is asked to do. As the load is increased, the current is increased and the rotation speed will drop. (A motor draws maximum current when it is stalled.)

There are two ways to control motor speed. One is to vary the driving voltage. This is not very satisfactory, because it requires a more complicated circuit and is kind of wasteful. The preferred way to control DC motors is by pulse width modulation (PWM). This drives the motor with full voltage (as figure 27), but interrupts the current periodically, maybe 400 times a second. As the interruption gets longer, the motor slows down.

¹⁷ Note that it is connected cathode to power. It will not conduct unless the coil kicks enough current back to raise the collector of the transistor above 5 volts. 1N4001 diodes will work fine for any motor you are likely to use.

¹⁸ When a DC motor turns, it acts as a generator, but the voltage generated is the wrong way. This so-called back EMF is subtracted from the applied voltage to get the current.

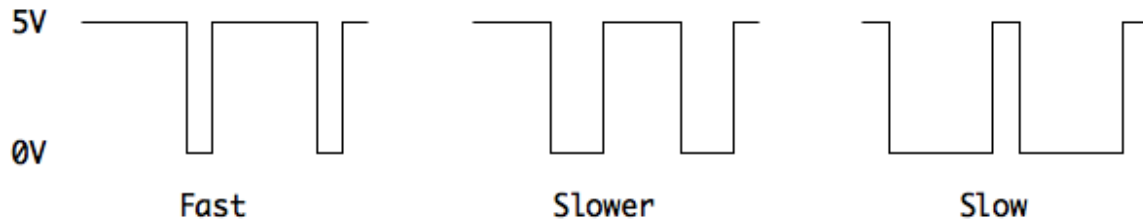


Figure 27. PWM

This sort of waveform is easily generated by the Arduino. The `AnalogWrite(pin, value)` function makes it especially easy on certain pins.

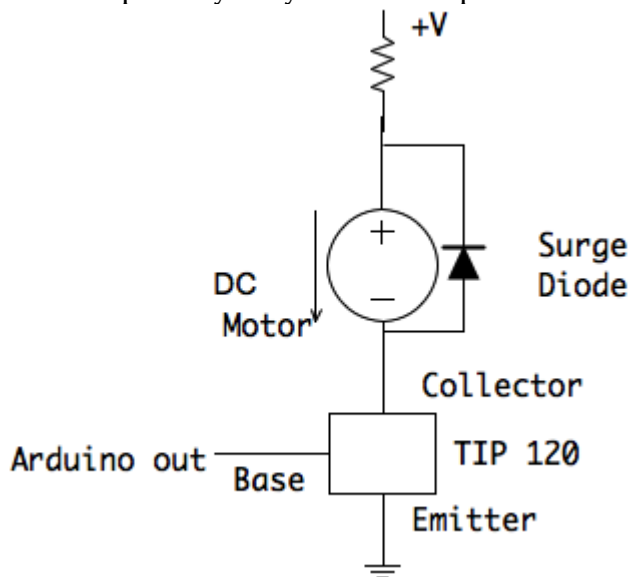


Figure 28. Driving a DC Motor.

The circuit of figure 28 will control the speed of a motor, but the direction depends on how the wires are connected. If you want to reverse the motor you need more drivers. You can get 4 on a chip called the L293. Figure 29 shows how it is hooked up. The lines labeled P1 and P2 are connected to Arduino outputs. Holding P2 at 0 and pulsing P1 will run the motor clockwise-- holding P1 at 0 and pulsing P2 will run it counterclockwise. The diodes protect the L293 from any surges produced by the motor. They are not required on the L293D version as they are included on the chip.

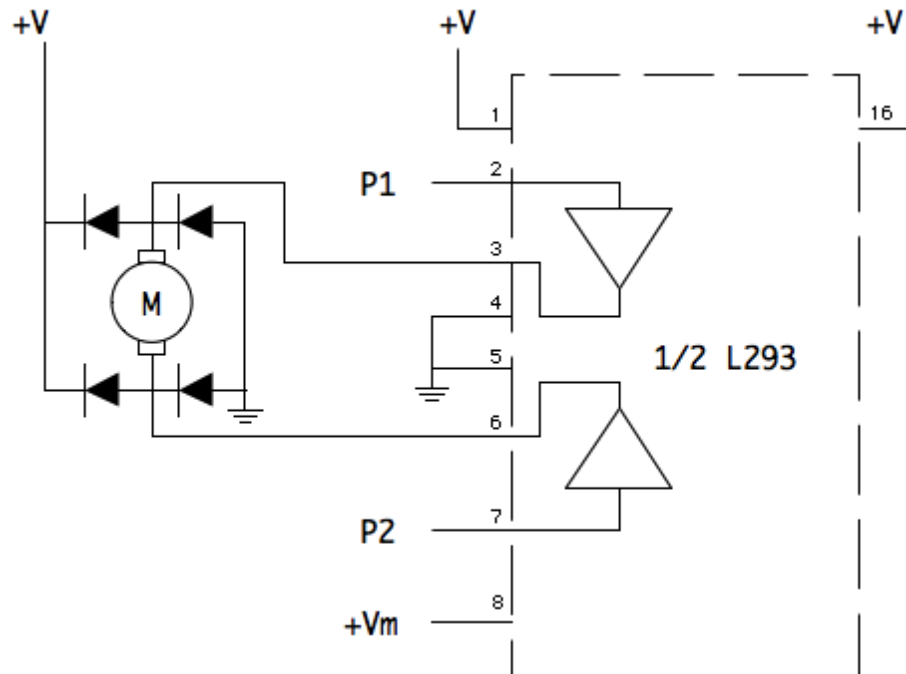


Figure 29.

One useful feature of the L293 is it has two power supply pins. The voltage applied to pin 16 is the supply for the inputs, and must match the Arduino power, usually 5V. The power for the outputs, which drives the motor is on pin 8. This can be as high as 36V. This setup allows a safe interconnection between the Arduino and high voltage devices. The L293 can provide 1 Ampere of current, and the L293D can provide 600mA.

Servo Motors

A servo motor includes a motor and a system to sense the position of the shaft. Some electronics (the servo amplifier) compare the shaft position with an incoming command signal and power the motor until the shaft is in the right spot. Servo motors come in a wide range of sizes and styles including linear action and ultra low frequency loudspeakers. The needs of artists can usually be met by a style of servos developed for model airplanes called RC servos. These are powered by Arduino range voltages and include a built-in servo amplifier.

Servo motors are used when you want to point something in a particular direction. Most RC servos have a limited range of action, often 180° or less. They are controlled by the width of a pulse wave at 50hz. The pulse is applied to a control lead and power comes from a second (usually red) lead. Pulses of 1.5 milliseconds set the shaft at the neutral position. This is half the maximum angle of rotation. If the motor has a range of 180°, the neutral position would be 90°. A pulse of 1 ms width will turn the shaft to 0°, and a pulse of 2ms will turn the shaft to 180°¹⁹.

¹⁹ This actually varies from motor to motor. 1.5ms is the neutral position though.

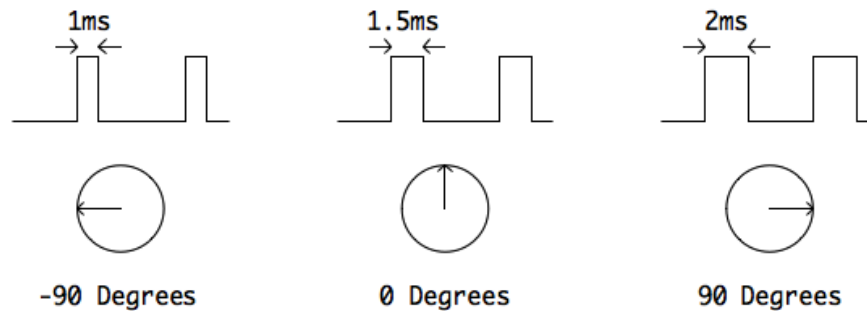


Figure 30. Servo rotation control.

RC servos can be controlled directly by an Arduino pin. The code to set the pulse timing is fairly straightforward, and can be simplified by the installation of a servo library.

Stepper Motors

Stepper motors combine features of DC motors and servos. They are usually more powerful than servos²⁰, and can rotate continuously. You can keep track of where they are pointing if you have a way to initialize your count, such as a hall effect sensor to detect home position.

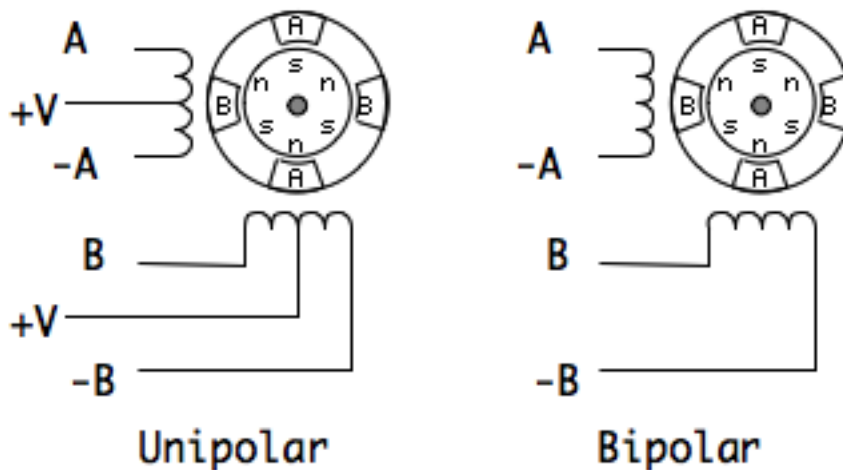


Figure 31. Stepper Motor

The heart of a stepper motor is a central rotor that is magnetized with at least six poles. There are two coils wrapped around the rotor (along the axis of the shaft), which show up in cross section in figure 31. Coil A is at the top and bottom, Coil B on the sides. If coil A has current flowing through it, it is magnetized with a north pole at the top and a south at the bottom. There are two versions of this, bipolar and unipolar. On the unipolar style, the coils have a center tap, which is connected to a positive voltage. Grounding the A wire energizes part of the coil

²⁰ But slower.

with a north pole at the top. Grounding the -A wire produces a south pole at the top. On the bipolar version we reverse the polarity across the winding to get the same effect. The B winding is the same, with the initial north pole at the left. The position shown is the result of energizing A, which attracted a south pole of the rotor to the top (and the north to the bottom). Notice that the rotor poles don't line up with B windings. If A is turned off, and B is energized, a south will be attracted to the left, turning the rotor 30°. If -A is now energized there will be a south pole at the top, and the rotor will turn another 30°²¹.

You can see that moving the motor is a matter of grounding leads in the right sequence. A, B, -A, -B will turn the rotor clockwise, and the reverse order will turn it counter-clockwise. This may be easier to follow with a graph that shows how each lead is pulsed:

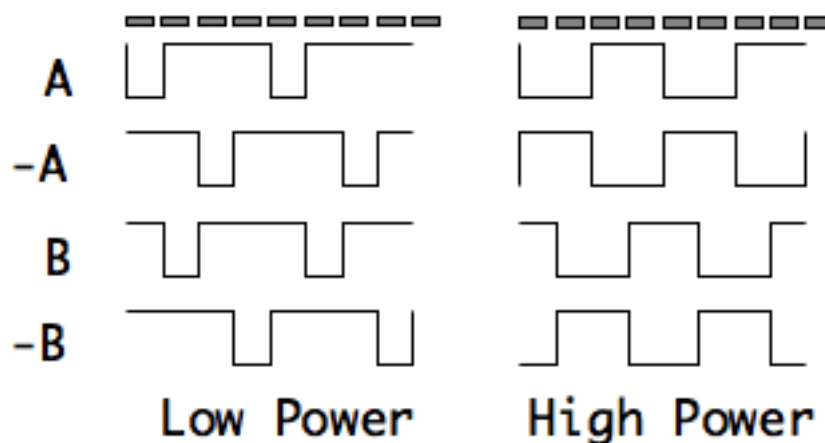


Figure 32. Stepper pulses-- low energizes winding.

There are two ways to do this. If each wire is pulsed in sequence, only one half coil is on at any one time. This is thrifty, but uses only half of the motor power. If the A and B coils are both energized, the rotor will step to positions between the windings, and there will be more torque²². Note that the two parts of a winding are never energized at the same time. A bipolar stepper has to run in the high power mode.

²¹ Of course, real motors have more poles and windings in them. Step angles go down to 1.8°

²² About 1.4 times the torque, and the motor will run a little warm.

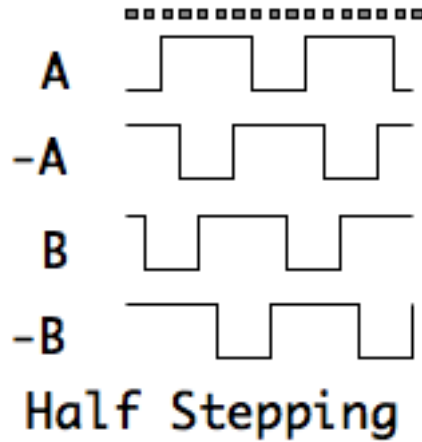


Figure 33.

A pattern that combines low and high power modes will allow more precise positioning. In figure 33, you can see that there are points where one coil is energized, and others where both are energized.

Since the pulse current is the only thing turning the motor, you can't run this directly from Arduino pins. We need a TIP100 switch for each of the four leads, or more conveniently, an IC with 4 motor drive amps in it, such as the L293.

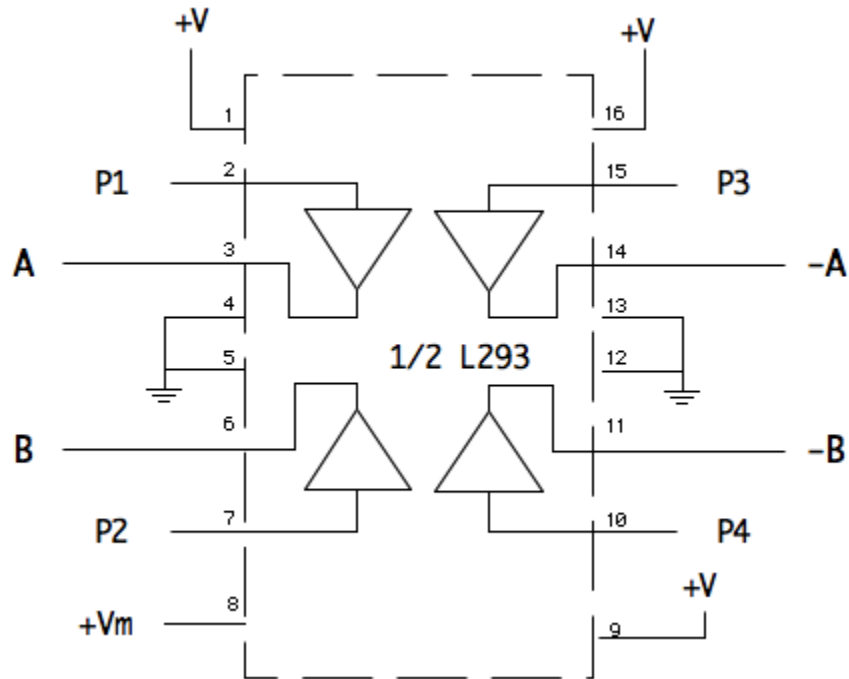


Figure 34. L293 wired for stepper control

Many motors need more current than the 293 will handle. For these use a pair of LMD18201 or other high power control chips.

Additional Circuits

Circuits that come up often in art work:

Multiplexing Switches

We often want to detect the action of a lot of switches, as in a keyboard. If we use the simple connection of figure 14, we would quickly run out of pins. The solution to this dilemma is to set up the switches as a matrix like figure 34.²³ Test lines (shown in red) go through a diode to one pin of every switch in a row. Sense lines connect the other pin of each switch in each column. The sense lines are also tied to the power supply through a pull-up resistor.

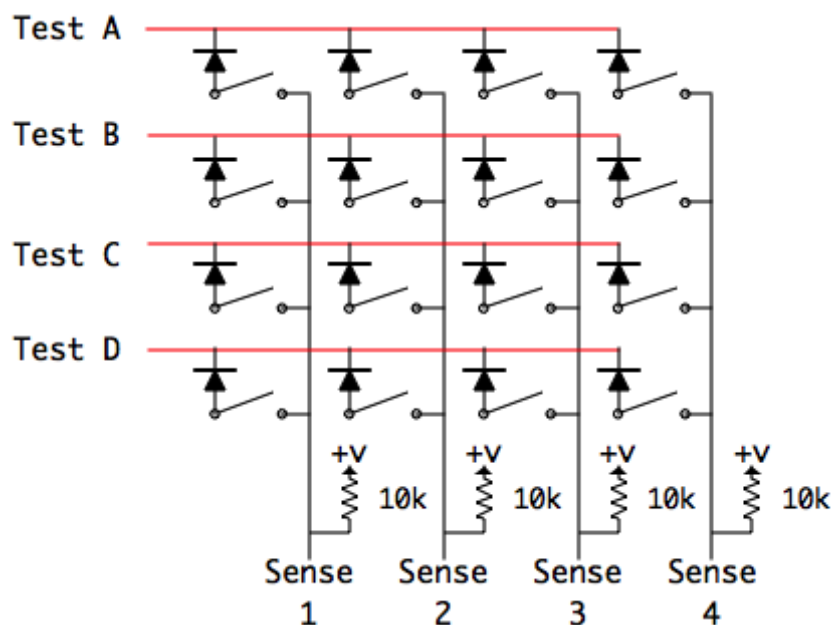


Figure 35.

The test lines are normally high. To detect if a switch is closed, bring its test line low and read the sense line. If the switch is closed, the low from the test line will be read. Otherwise, the high from the pull-up will be read. The diodes prevent false readings when more than two switches are pressed. Without the diodes, if two switches in the same row are pressed, the respective sense lines would be connected together, and a third button pressed in either column would register in both.

The number of switches you can read is the product of the test (output) pins and the sense (input) pins, so 8 pins can read 16 switches, 10 pins can read 25 and so on. Synchronizing the output and input pins is a little tricky in Firmata, so this is best managed by programming the arduino directly. If you really want to, the Notes on Firmata tutorial shows how.

²³ They don't have to be in a square of course, they are just wired this way.

A Duemilanove will handle 72 switches, but that would tie up all of the pins. You can access more switches with fewer pins by using a multiplexer chip to handle the row addressing. The CD4051 is an inexpensive, easy to use multiplexer. The technique is shown in figure 36. The CD4051 has eight inputs that can be connected to one output. Which input is connected at any time is controlled by three address pins. If the address is 010 (binary) input 2 is connected.²⁴ We can read 8 switches on one pin by stepping through the addresses and noting the input value for each. This requires 4 pins (instead of 6) to read 8 switches, but it gets better. The same address lines can be connected to as many CD4051s as you like. Each bank of 8 inputs only requires one more pin. A Duemilanove can read 96 switches with a couple of pins left over.

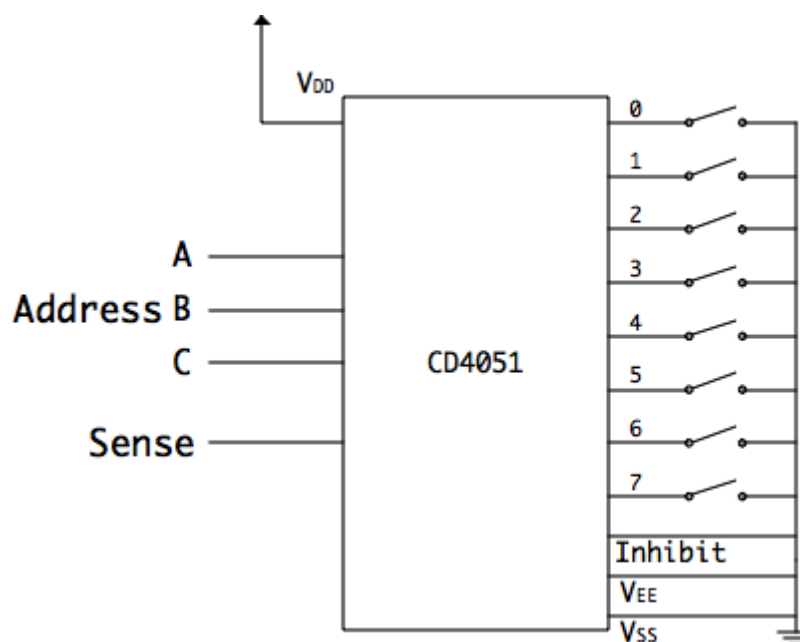


Figure 36. Reading switches with the CD4051.

For the really ambitious, you can connect several CD4051s to a single Arduino input pin. A CD4051 has a pin labeled *inhibit*, which can disconnect the chip from the sense line. Note that that is tied to ground in figure 36. When inhibit is high, the output pin is entirely disconnected. It's easy to configure another CD4051 to provide inhibit addressing logic for a set of 8 chips. That's 64 switches on 6 address lines and one input. There's no real limit to this except the number of switches you can read in reasonable time. Again, the best performance will come from directly programming the Arduino.

²⁴ The inputs are numbered 0 - 7.

Multiplex output

The CD 4051 is a bidirectional chip. You can use it to choose one of 8 inputs to connect to an output, or to send one input to one of 8 outputs. The address and inhibit lines work the same in each case. Figure 37 shows how to hook one up to control LEDs. Only one LED is lit at a time, but if you change addresses fast enough they all seem to be on. This is another device that works better with custom programming than with firmata.

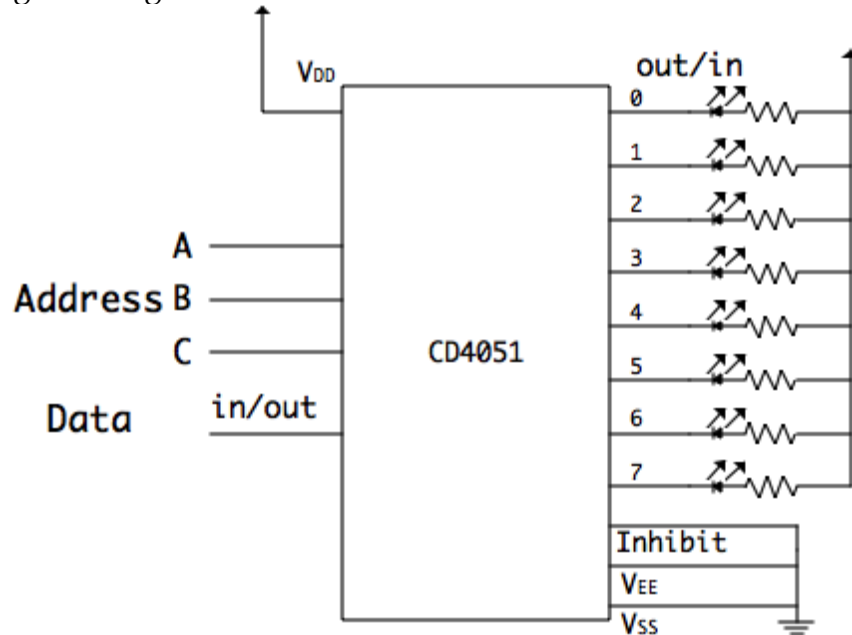


Figure 37. Using CD4051 to control output.

There is about 125 ohms of resistance through the on path, so take that into consideration when figuring the current limiting resistors for the LEDs. The switches can handle up to 18 Volts, so the current limit is about 144 ma.

The Opamp Revisited

Sometimes we want to modify the signal that we are applying to an Arduino analog input. This usually happens when the signal is too small. For instance, the flex sensor circuit in figure 21 can only get up to half voltage at the best of times. Many installations will not bend the sensor very far, so the actual voltage change is even less. We can amplify the voltage with a simple opamp circuit.

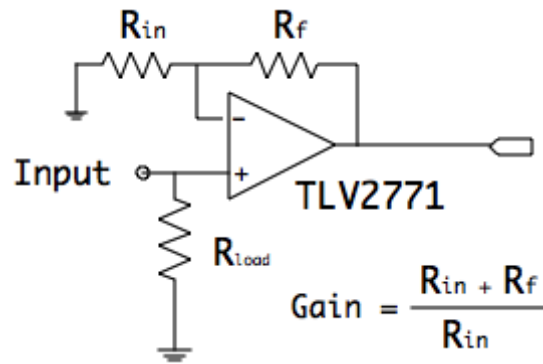


Figure 38. The non-inverting opamp circuit.

Figure 38 illustrates how to set the gain of an opamp. The circuit in figure 22 used a direct connection from the output to the inverting input to set the gain at unity. If you want more gain, apply the output to the inverting input through a voltage divider. This means there will be one resistor from the output to the inverting input (R_f in the circuit) and a second resistor from the inverting input to ground (R_{in}). The voltage gain for a signal applied to the non-inverting input will be determined by the ratio of these resistors, specifically

$$\text{Gain} = (R_f + R_{in}) / R_{in}$$

Note that if the resistors are of equal value, the gain will be 2. The actual value is not important, on the ratio of R_f to R_{in} matters. It's convenient to use 10k for R_{in} , as that makes the gain obvious at a glance.

The resistor shown on the non-inverting input is optional. It provides some noise immunity and may minimize offset error found on cheap opamps. It will usually equal R_{in} . The schematic specifies a TLV2271 (or TLV2272) opamp. These will work properly with the 0 - 5v power supply of the Arduino board. Do not connect it to the 9v or any outside supply, because once you start amplifying signals, it's easy to exceed the acceptable voltage of the Arduino pins. If the opamp power is only 5V, the Arduino is safe.

Opamps are an excellent pathway to the world of analog electronics and are worth further study. Any electronic text will cover them, but the bible is still "The IC Opamp Cookbook" by Walter Jung.

Detecting short events with a 555 timer.

Some switch events can be hard to detect. If the contact is held closed for only a few microseconds, the Arduino may not poll the input often enough to catch it. If the Arduino is set for very fast polling, the action of some switches is so messy that it may be counted as several events. (This is known as switch bounce.) The 555 timer is an excellent answer to either problem. The basic action of a 555 is for the output to go high for a defined amount of time after the input is pulled low.

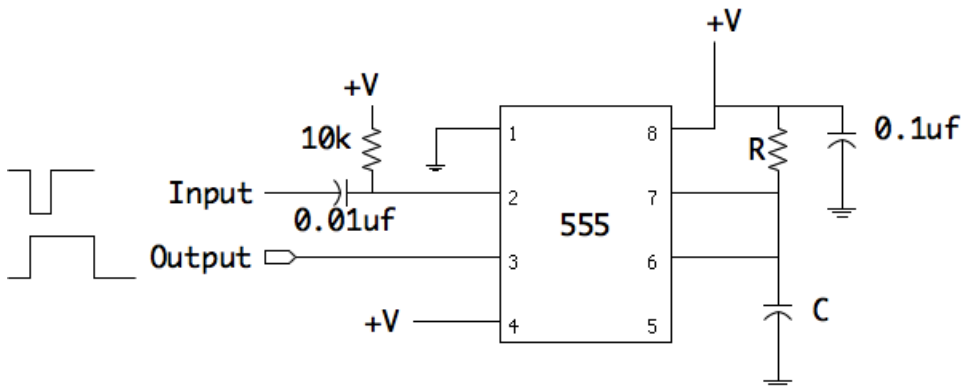



Figure 39.

The symbols that look like  represent capacitors. The duration of the 555's on pulse is set by the resistor labeled R and the capacitor labeled C. The formula is approximately:

$$\text{Time in seconds} = 1.1 \times R \text{ in ohms} \times C \text{ in farads.}$$

Since capacitors usually come in sizes measured in microfarads²⁵, an R of 10k and a C of 0.1 μf will give an on time of 1.1 ms. The capacitor by the power pin prevents the impulse of the 555 switching on or off from disturbing other devices on the same power supply. The capacitor on the input line will prevent the circuit from retriggering if the button is held down a long time.

Of course this is also an excellent choice for stand-alone circuits. If you just want to light an LED for 1 second after a button push, this will work fine.

²⁵ That's one millionth of a farad. A farad is the unit of capacitance (named for physicist Michael Faraday). We prefer to work with tiny amounts of current, which means big resistors and tiny capacitors.

Using the 555 as an oscillator.

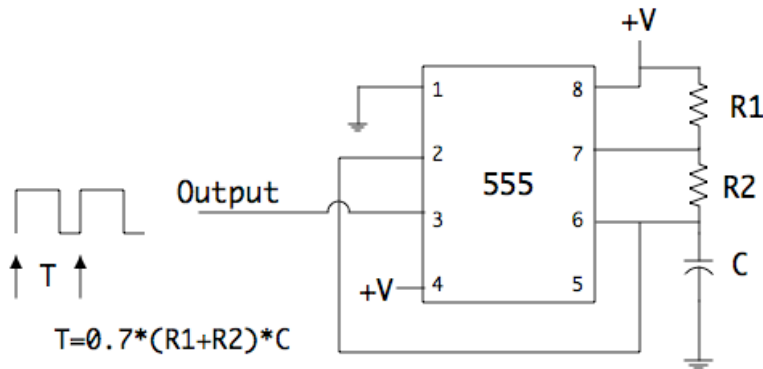


Figure 40.

The 555 chip is quite versatile. It can take care of practically any simple timing chore. One common application is as an oscillator, either blinking a light or generating a tone. Figure 40 shows the oscillator schematic, which produces a pulse wave.

The period (T) of the circuit is determined by both resistors and the capacitor. With the formula given. For frequency, the formula is:

$$f = 1.4 / (R1 + R2) * C1$$

Note that the sum of the resistors is used in both cases. If you want the output to be a square wave, make R2 much larger than R1. To get T up to a second you need a fairly large capacitor, maybe 50 μ f.

The web is littered with circuits that use the 555 in interesting ways, and there have been at least two books about it, by Walter Jung and Forest Mims. The chip is available in dual (556) and quad packages (558) for complex circuits like cascaded delays.

Simple Audio Amplifier

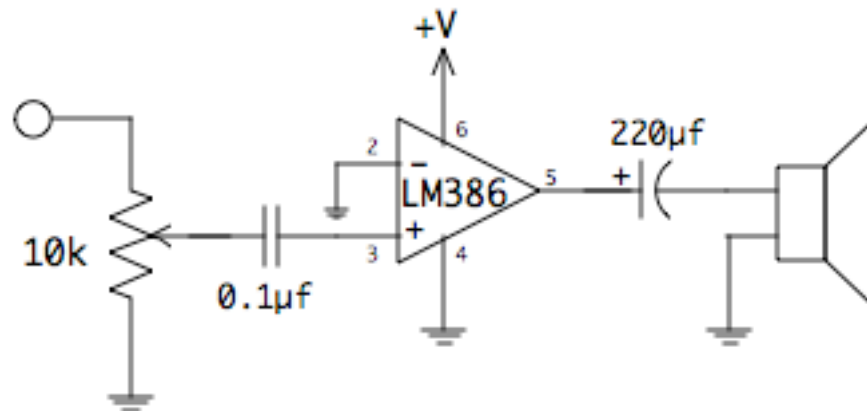


Figure 41.

Sometimes we need a modest audio amplifier to power headphones or a small speaker. The LM386 integrated circuit was designed specifically for such tasks. It can be powered by a 9v battery or a power supply of 6 to 12 v. (Ensure that the power supply is free from any hum.) Battery life will depend on how loud you play the music. All it needs is three additional parts. It won't win any prizes for fidelity, but it will work.

The LM386 is an opamp with internal feedback to set the gain at 20. You can change the gain by adding components, but this usually works. Consult the data sheet for more complex circuits.

Simple Power Supply

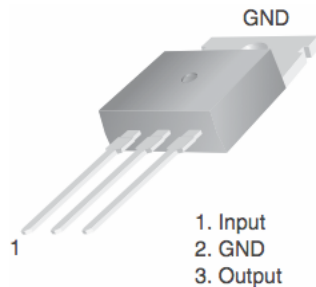


Figure 42.

Every circuit I've discussed needs power to operate. If a project is based on an Arduino board we can use the power there, but it does have limited current, especially on the new tiny boards. We can use a wall wart power supply for functions such as motors and lights, but these are unregulated, meaning the output voltage varies with the load (amount of current). In fact, you will be surprised if you measure the unloaded output of one—a nominal 9v supply may show 18 volts! This is not good for integrated circuits. The answer is to add a voltage regulator to the wall wart.

My favorite regulator ICs are the LM78xx series. The last two digits are the voltage, so an LM7805 is 5 volts, an LM7809 is 9 volts and so on. You can get values up to 24 volts. The chip is shown in figure 42 and the circuit in figure 43.

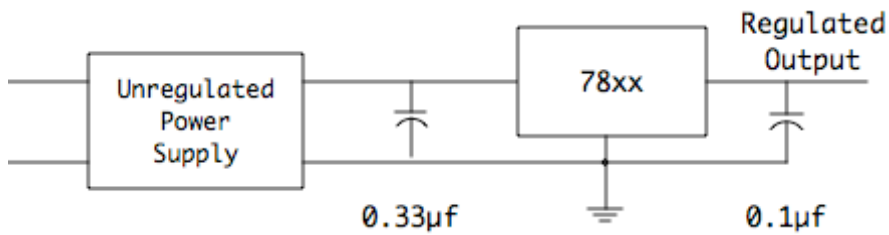


Figure 43.

The circuit is quite simple—all that is needed is a couple of capacitors to prevent ripple and noise in the output. The values are not critical, except the capacitor on the input side must be at least 3 times the output capacitor so that the device is not back-biased when the power is shut off. The unregulated power supply must be 2 volts more than the output but can be as much as 35 volts DC.

You can pull as much as an amp out of one of these, but if you do the tab will get hot. You can get bolt-on heat sinks to help keep it cool.

Converting PWM to Voltage

Several of the Arduino pins can generate a pulse width modulated signal as shown in figure 27, and that is dandy for making lights go dim or slowing down a motor. However, sometimes you want an actual voltage. The circuit in figure 44 will do the job, but there are some tradeoffs to make.

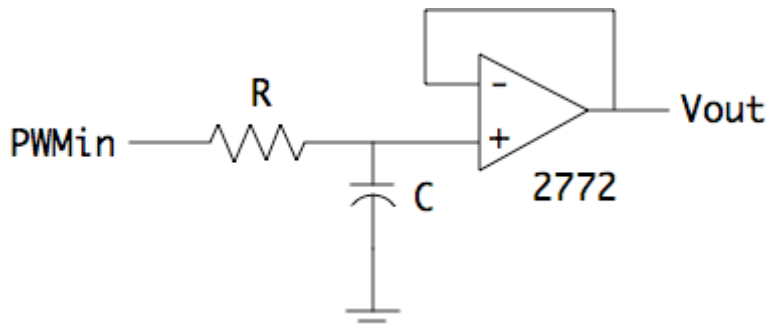


Figure 44. PWM to voltage conversion.

This is a simple high pass filter, with a very low cutoff frequency. The values R and C set the time constant of the circuit, with R in ohms \times C in farads = time in seconds. Since practical capacitors have values in microfarads (μf) the resistors are usually thousands of ohms. In fact, a one second time constant with a one μf capacitor requires a one mega-ohm resistor. The trade off here is if the time constant is too long, the voltage will change slowly. If it is too short, the pulses will not be completely filtered and there will be ripple in the output. The frequency of the PWM output is 500 Hz so a time constant of 2 ms will turn the signal into something like a sine wave. Here's a table of measured ripple:

C	R	TC	Ripple
1 μf	22Kohm	22 μs	120 mV
1 μf	47Kohm	47 μs	80 mV
1 μf	100Kohm	100 μs	40 mV

40 mV is as low as it will get because the USB based power has that much. The transition time when the voltage changes is approximately 8 time constants, so the values with the best ripple will take a second to swing from 0 to full on.

Debouncing Switches

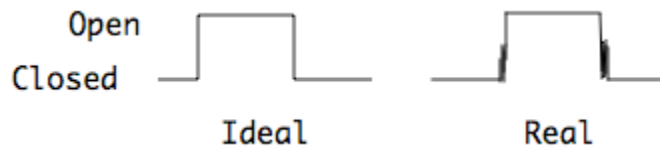


Figure 45.

Switch bounce is a common problem in high speed circuits. Switch bounce occurs when the switch is just opening or closing. Instead of a clean transition from open to closed (or vice versa) there is a moment of uncertainty, as illustrated in figure 45. This is caused by electrons prematurely jumping the gap of the nearly closed switch. Often there is no harm done, (if the switch is carrying audio there will be a pop), but if the switch is connected to an Arduino pin there will be multiple 0 to 1 transitions. This can be addressed in the software, but the resulting code is complex and will slow the system down.

Debouncing can be provided by a simple circuit built around a CD4584 Schmidt trigger. Schmidt triggers are logic circuits that invert the signal (0 becomes 1 and 1 becomes 0) with a special qualification. If the input is a rising pulse, the transition occurs at about 3/5 of the voltage. If the input is a falling pulse, the transition occurs at about 2/5 of the voltage. Thus with a 5 volt system, the switch noise can be nearly 1 volt and there will only be one transition either way. This trick is called hysteresis. Figure 46 shows the circuit.

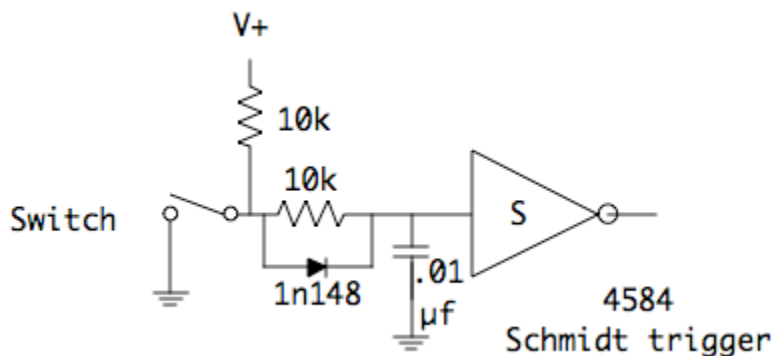


Figure 46.

The Schmidt trigger is represented by the triangle (they come 6 on a chip, so consult the data sheet to find the pin numbers for matching inputs and outputs.) The capacitor is the heart of the circuit. If the switch is opened, it charges through the diode and upper 10k resistor, smoothing out the noise enough that the Schmidt trigger can do its thing. When the switch is closed, the capacitor is discharged through the other resistor. With this circuit, the capacitor time constant is the same either way (1 ms). Note that since the Schmidt inverts the input, the Arduino sees a data 1 when the switch is closed.