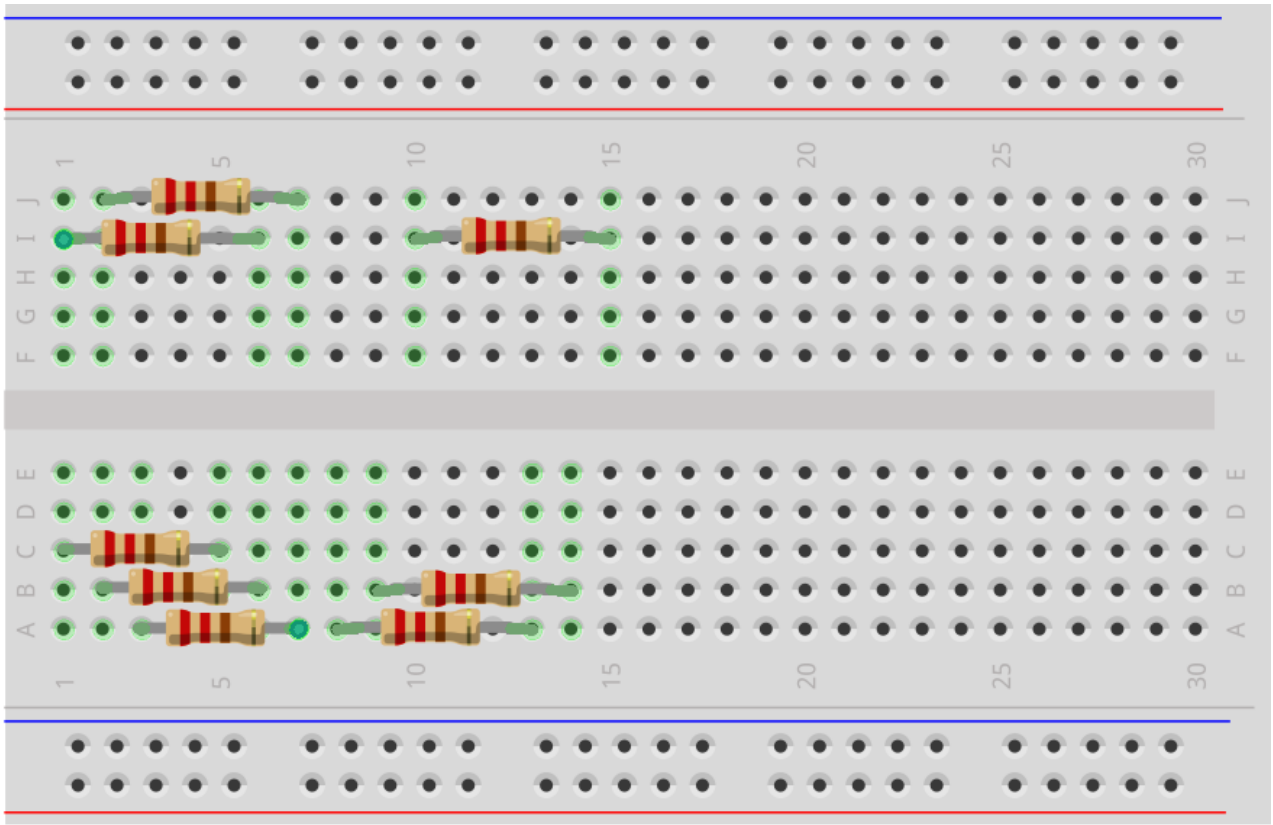
Seven Segment Displays

Everybody loves a 7-segment display! Seven little LED bars make up an outline digit “8” and by selectively turning the segments on and off we can make recognisable digits 0 through 9. With some imagination and poetic license it is possible to show most (OK, not all) letters of the alphabet too, and even some cool little animations.

As well as being fun to play with, a 7 segment display is often a good choice for a display in a simple Arduino project. So lets see how we can control a 7 segment display, with four digits, directly from an Arduino.

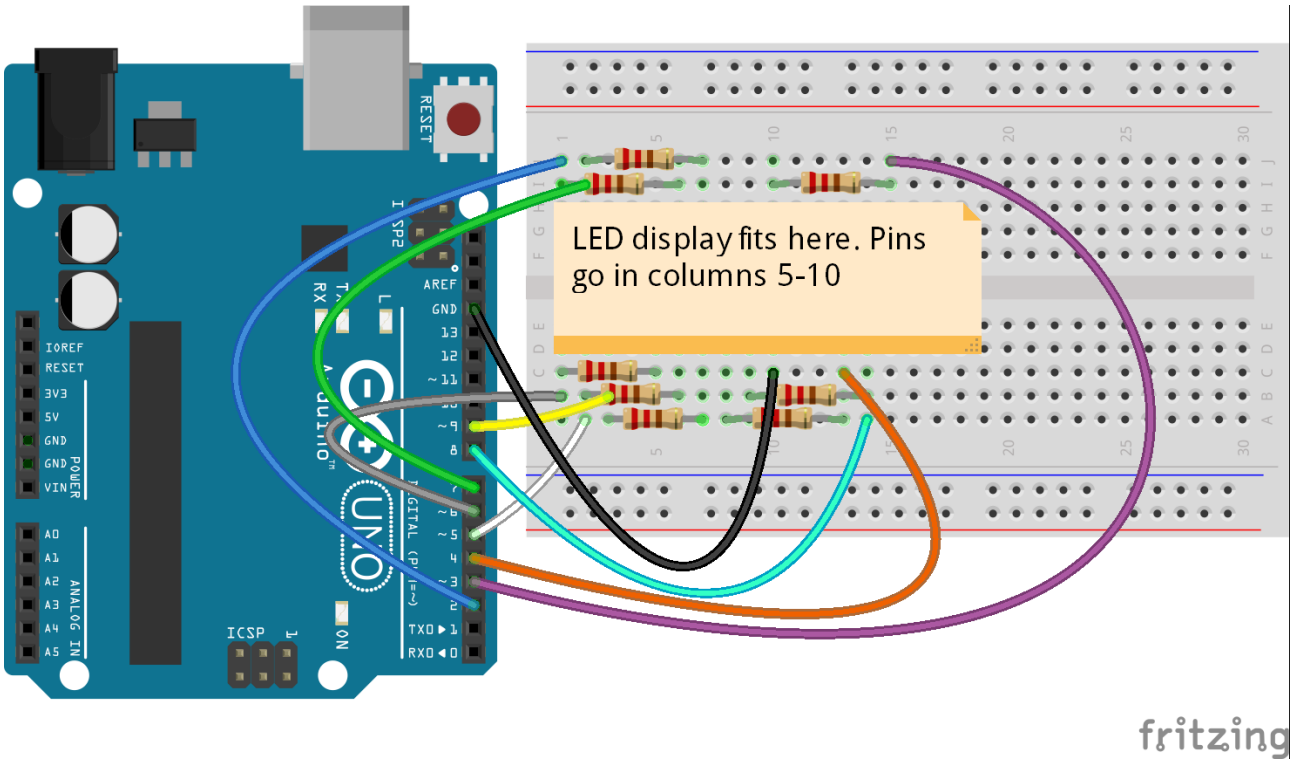
I won't lie to you.. there is some pretty extreme breadboarding here.. and you do need to make sure you do it right. Check all your connections **before** you power up - it is possible to damage the LED display and burn out segment with incorrect connections (by missing resistors out of circuit) so please be careful!

Lets start by insert eight 220 ohm resistors into the breadboard. Because we will be tight for space, it is pretty important you use exactly the same locations shown below. Use the row letters and column numbers to get the right location.



Now add the wires shown (don't get hung up on the colours - they are just to help tell them apart). Once the wires are in place, place the LED display where shown.

Fritzing does not have a suitable part for the LED display, but basically it is about the same size as the text box shown in the diagram and goes with the decimal points to the bottom of the display. The first column of pins placed in column number 5 of the breadboard There are 6 pairs of pins, so they should extend to breadboard column 10.



Now load up sketch **SevenSegment1.** With luck you should see that the rightmost digit of the display is counting up from 0 to 9. Woop!

Now lets try something. Pull out the wire connected to breadboard pin C10 (the one that goes to GND) and instead plug it into pin location J5 (with the other end still connected to GND).

Hopefully, you still see the counting continue... but now it is in the left most digit position. Again change the position of this wire to pin location J8, then to J9 and back to C10. You can see that with just one wire we can select which digit is active. So how does that work?

Well, this display (and it is pretty standard) is wired so that the “anode” (positive terminal) of each LED is wired to the anode of the same position LED in each of the other 3 digits, and they all share a single pin. On the “cathode” (ground terminal) side, all of the LEDs on a given digit are wired together to a single pin. Effectively we have a grid or “matrix” where to turn each LED on we need to select the segment anode and digit cathode. This configuration is termed “common cathode”. There is also a “common anode” type of wiring which is the opposite way around.

So now, this wiring scheme seems to restrict us to having all the four digits display the same thing, right? You may be wondering how we can possibly have different digits shown in different locations at the same time, like showing the four-digit number “1234”. Well, more about this shortly....

First thing... we want to be able to switch between digits using the Arduino, to save us all this unplugging of wires!

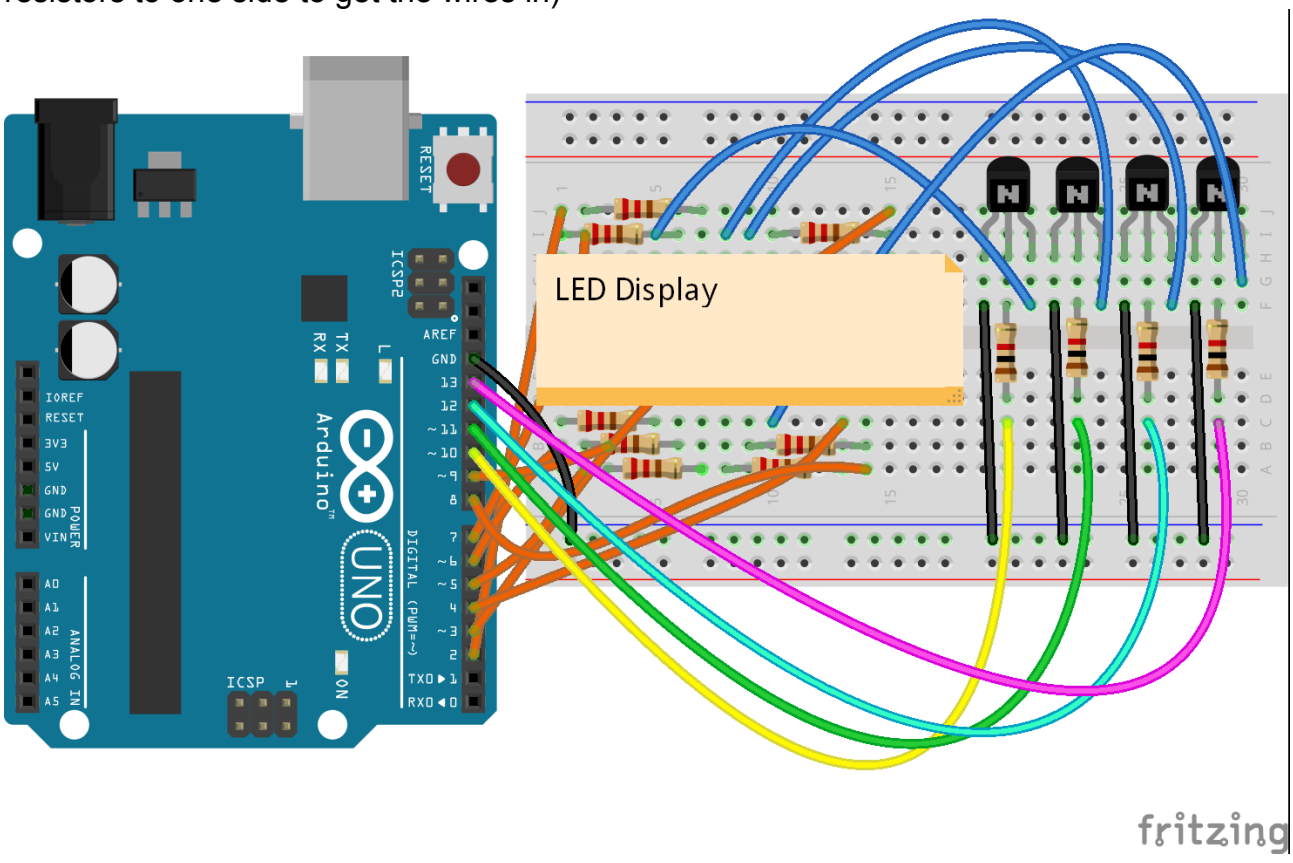
It might look like we could simply connect each digit's cathode wire to an Arduino pin instead of GND. Then we could just set that pin to LOW to activate the digit and HIGH to turn it off.

This does make sense, but we need to think about **current**. Including the decimal point, a single digit is made up of 8 LEDs. If they are all on at the same time, it is like us putting 8 LEDs on a single pin - it can't cope with all that and at the very least we'd see the LEDs get dimmer the more that are switched on (at worst we might damage our Arduino).

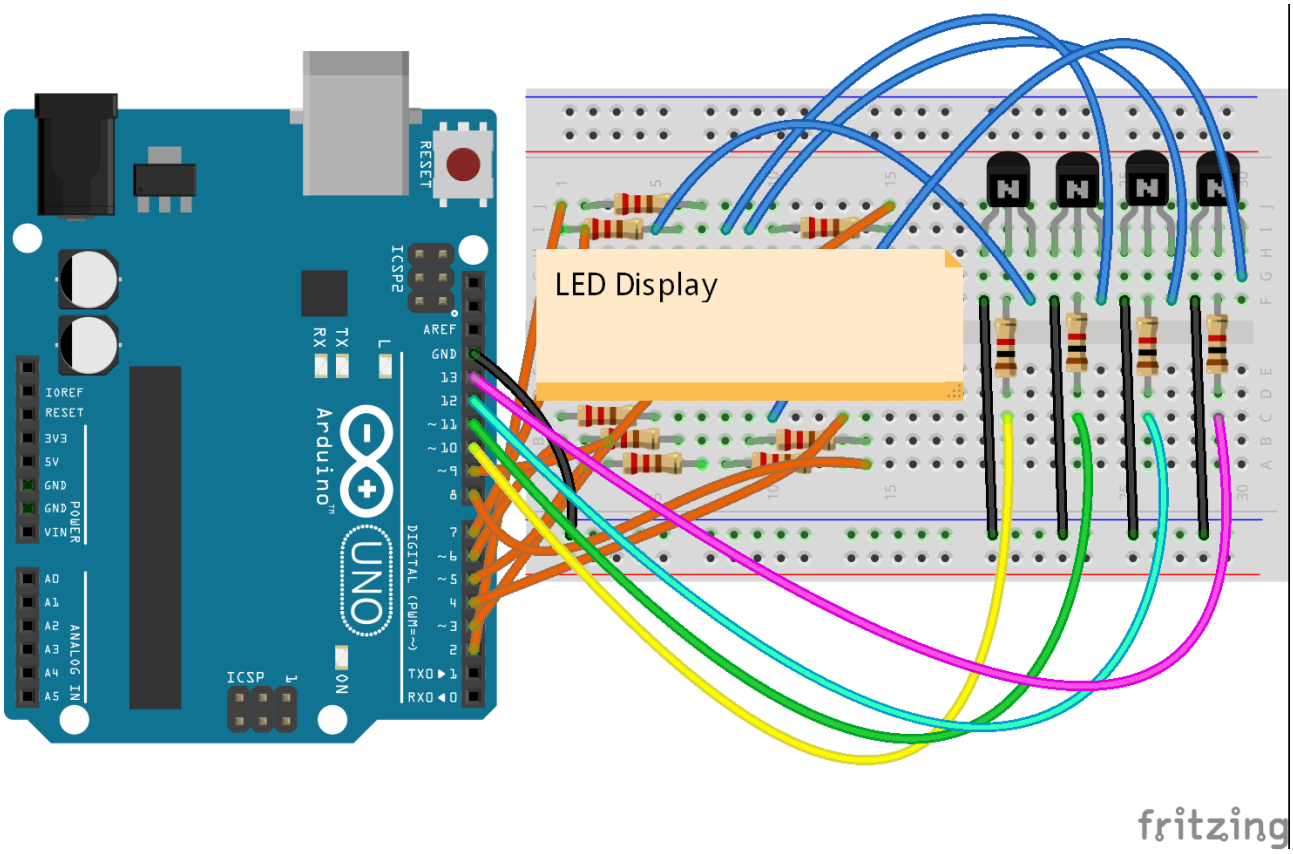
The reason is that the pin has a maximum amount of current (Amps) that can go through it (in either direction - HIGH or LOW). Each LED needs about 15/1000 of an amp (15mA) but our Arduino pin can only deal with about 40mA (about two LED's worth). If we want to switch up to 8 LEDs we're going to need to bring out the big guns... Enter the transistor!

We can use a transistor as an electronic switch (they can also be used as amplifiers) and as they are outside our Arduino board, we can select a type with an appropriate current handling ability for our task. Your kit includes four transistors (2N3904 type) each of which can switch up to 200mA... Sorted! Lets get them in that breadboard.

Start off with the four transistors (make sure they are the right way round, with the curved part of the case towards the top of the breadboard) and add in four 1k Ohm resistors as shown. Add the four wires (shown in different colours) to each of the LED display cathodes at locations I5, I8, I9, C10 (you might need to carefully nudge some of the 220 ohm resistors to one side to get the wires in)



Finish up by adding four more wires from the 1k resistors back to pins 10, 11, 12, 13 of the Arduino.



Lets load up the sketch **SevenSegment2.** Now we can see how the Arduino is able to electronically switch between the digits. We can show a 1 on the first digit, a 2 on the second and so on! That's easier than unplugging wires.

Q: How are the transistors used?
These are 2N3904 transistors, which are a type of transistor called an “NPN bipolar silicon transistor”. Without dwelling on this too much, this type of transistor has 3 terminals called the Emitter, Base and Collector (left to right as shown). When a small current flows into the base connection down to the emitter, a much larger current can be switched from the collector down to the emitter.

We can use them as a switch.. when the base is LOW no current can flows from collector to the emitter but when we set the base HIGH that current can flow. In our setup the emitter is connected to Ground and we are switching the cathodes of the display digits.

Q: Why do we need the 10k resistors?
Only a small current is needed to do the switching. We want to use a resistor to keep that current deliberately low, because the base to emitter resistance is otherwise rather low and is a near-short circuit to ground for our output pin which could possibly damage our Arduino output.

Well this is all very good, but still we're not able to show four different digits at the same time.... Or are we? Lets reduce the delay in the **SevenSegment2** sketch from 500 milliseconds down to 1 millisecond.

Now the digits 1,2,3,4 are all showing at the same time.. but its just a cheap optical illusion right? All we are doing is flicking between them very fast. Isn't that cheating?

Well.. actually no. This is how it is done. It's a technique called **multiplexing**, which means that multiple sets of data (i.e. numerals) all share the same single channel (i.e. the same eight anode lines)

By an optical illusion called **persistence of vision**, the switching takes place so fast our eyes can't keep up and all the digits look like they are on at the same time. This is just now these things work! digital clocks, LED matrix signs, old school TV tubes, they all do this.

You can tell from the limited number of pins on our 4 digit display that it could not be used in any other way. **Its all a cheap trick I tell you!** but its certainly fun to play around with :)