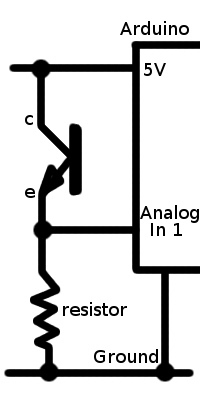
**How to Connect a Phototransistor for Arduino**

After serial link was up. I wanted to add my "IR Detection circuit" to the arduino.

It wasn't really difficult. The schematic below will do:



I used 10k resistor.

After connecting photoresistor, I wanted to see the change in values at Analog 1. This is where Serial Communication comes handy, I can use it to print the values of Analog 1 in a continuous loop so I can see what difference is there when I bring an IR LED in front of this sensor.

**I used following code:**

int val;

void setup()

{

Serial.begin(9600); // sets the serial port to 9600

}

void loop()

{

val = analogRead(0); // read analog input pin 0

Serial.print(val); // prints the value read

Serial.print(" "); // prints a space between the numbers

delay(10); // wait 10ms for next reading

}

**Motor control**

Lesson Summary

In the lesson you be taught how to interface with inductive loads, motors to be more precise, although you can use the same hardware to control solenoids and other inductive loads.

I'm assuming you have already read, understood and are comfortably using PWM as described in Lesson 2. We're also assuming you know how to connect an analog sensor and read it from the analog pins (Lesson 3).

If not, please take a look into it and try to understand the basics. If you're not comfortable with those lessons, then ask any of the guys in the workshop for a quick help.

**Powering electric motors with Arduino**

When Arduino is powered from the USB port, it can only drain a 0.5A maximum current from the computer. Although the motors you will be using now are small and might even be able to run without more than that, when you attach something, it'll crash your Arduino. Thus the move to external power. Don't worry about configuring your arduino because in most boards the power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. These can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can also be inserted in the Gnd and Vin pin headers of the POWER connector. The board can operate on external supply from 6 to 20 volts. However, if supplied with less than 7V, the 5V pin may supply less than five volts and render the board unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts. If you are supplying power through the 2.1mm jack, that same power is available for your own use at pin Vin. For more information please take a look at the Arduino's Hardware page, power section.

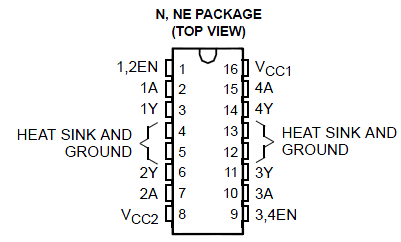
In order to make the motor's shaft rotate you only need to connect one wire to ground and the other one to the power source. There are, of course, two very important issues with this setup, you cannot control the motor's speed and the spinning direction is fixed, according to your wire connections. To change the direction of movement, polarity has to be inverted. This can be done by hardware. Doing it by hardware is the same as saying you go out and buy an Integrated Circuit (IC) which has the necessary logic implemented to receive a direction signal and invert the power polarity. There are several chips available on the market. The easiest solution is to use an Arduino Motor shield. It already has the IC soldered on as well as easy connectors so all the difficult part done for you. Just plug your motor and off you go. On the other hand, using an Arduino, you can control polarity by programming it to drive two transistors which connect the motor's leads to either ground or power. You cannot use the digital pins of the Arduino directly to drive the motor because they are control pins. The maximum currently you can safely withdraw is 40mA.

In your lesson, we'll be using an L293 IC to control the motor. For those of you who have the Arduino motor shield, it is based on the L298 IC. Ask the monitors for help in case you cannot figure how to wire and control it. Remember, it has direction control and speed control. Look for more information in the Arduino hardware page.

**Material needed**

1. Arduino + USB cable;
2. 2 DC motors;
3. 1 L293 IC;
4. 2 potentiometer;
5. 2 10 kΩ resistors;
6. 2 QTR-1A infrared sensors;
7. Breadboard;
8. Plenty jumper wires;

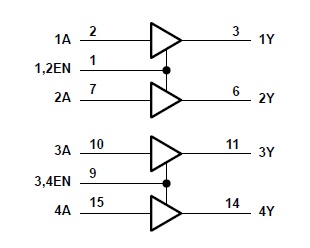
**Setup**



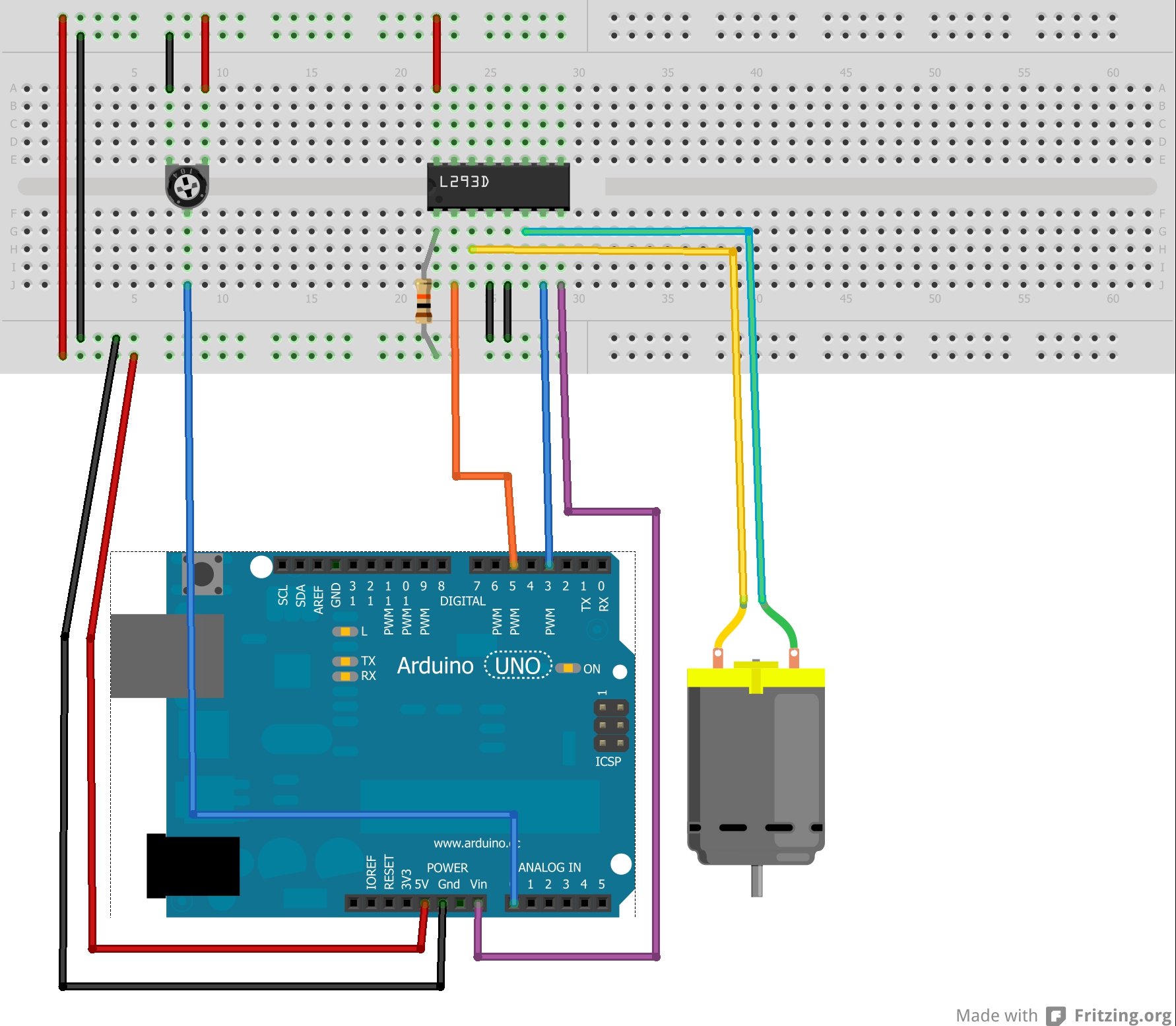
Start by placing the L293 on the breadboard. Take special attention to the orientation of the IC. Use the datasheet to identify the various pins. If you look closely at the picture below, you'll notice pins are labelled and numbered 1-16. Below is a brief description of what each pin is for:

* 1,2 EN - Enable input pin for drivers 1 and 2. When an enable input is high, the associated drivers are enabled and their outputs are active and in phase with their inputs. When the enable input is low, those drivers are disabled and their outputs are off and in the high-impedance state;
* 3,4 EN - Same as 1,2 EN but for drivers 3 and 4;
* 1A, 2A, 3A, 4A - driver input pins. These pins should be connected to your Arduino's motor control pins (where PWM signals are coming from);
* 1Y, 2Y, 3Y, 4Y - driver output pins. Please wire these pins' to your motor terminals;
* Vcc1 - Logic supply voltage. This pin is used to supply voltage to the IC's control logic;
* Vcc2 - Power supply voltage. You should connect this pin to your power source (9-12V). Pins \*Y will be powered from here;
* Pins 4,5,12,13 - the pins should be connect to ground. If we had a large load to power, our IC would tend to heat and these pins could be soldered to a copper plate to help cooling.

Please notice the driver's pins are paired (1/2 and 3/4) and, therefore, this IC is able to drive two separate motors. It is advisable to wire the motor on paired pins. Here's a logic schematic:



For ease of reference, please take a look at the wiring scheme below. Click on the image to enlarge. Please notice we've wired the EN pins using resistors so as to avoid excessive current drain.



[CAUTION]: Do not forget your circuit will have ate least two different voltage levels. Take your time to review your wiring before powering the system to avoid short circuits and material damage.

**Example 1**

Your first attempt and making a motor rotate will be done with the aid of a potentiometer and using only one motor. This example's goal is to control the motor's speed with just the potentiometer. As mentioned before, we will resort to PWM to control the motor driver.

If you want, you can try to write the code yourself but we've provided a commented example for you to use as a stating point. You can get the Arduino code here.

What this bit of code does is read Analog port 0, whose value is between 0-1023, convert it to a value from 0 to 255 (using the map function) and move the motor according to that value.

**Exercises 1:**

Replace the potentiometers with an IR sensor. Use it to control the speed of your motors with your hand. The closer your hand is to the sensor, the faster the motor should rotate. Please refer to Lesson 3 for details on how to use the IR sensor. You'll find the connections' labels under the sensor.

**Example 2**

What about controlling the motors direction? In the first example we just turned the motor in one direction. Now we'll adapt the code to make usage of the second driver (remember we said drivers were paired?).

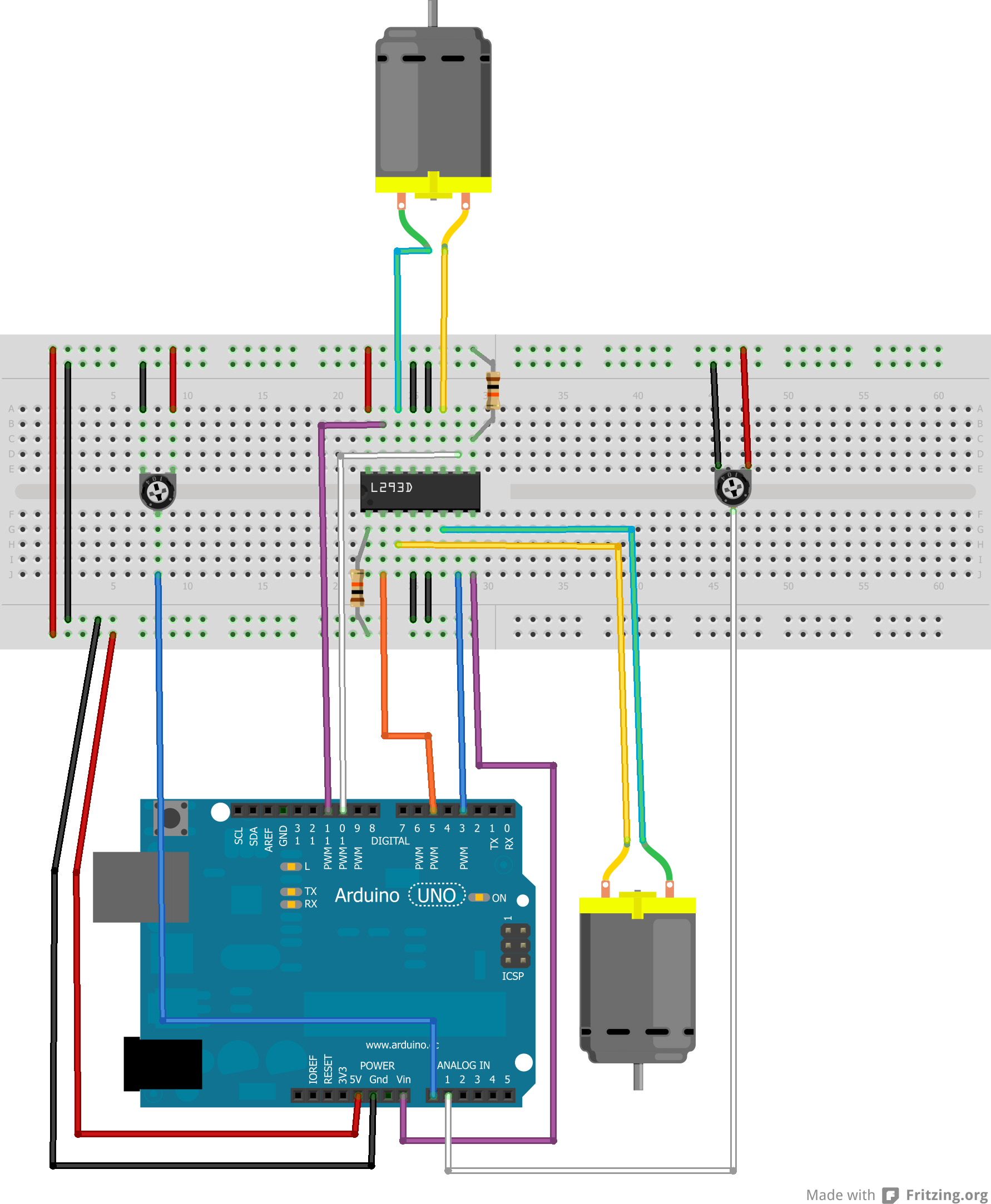
Please re-wire your circuit according to original scheme. The goal is to have the motor stopped when the potentiometer is in the middle of it's range. When turning the potentiometer to either side, the motor should move in said direction and its speed should be proportional to the potentiometer's position.

If you want, you can try to write the code yourself but we've provided a commented example for you to use as a starting point. You can get the Arduino code here.

What this bit of code does is read Analog port 0, whose value is between 0-1023, convert it to a value from -255 to 255 (using the map function) and move the motor according to that value (both speed and direction).

**Exercises 2:**

What if you wanted to control two motors independently? Use a second potentiometer to control motor B while controlling motor A as in the example. Use the schematic below to wire the second motor.



**Engineering Challenge Training:**

Just as in other lessons, lets use Arduino's serial port to interface with your computer. Your program should allow for the motor to rotate when it receives a character ("1" for instance) or disable it when it receives a "0". You can also define the turning direction with "L" and "R" characters. Speed control should be achieved with the IR. Can you implement an independent control for each motor? [Hint: make use of the switch/case function. Click here to read about it in the Arduino Reference.]

http://workshops.ist.utl.pt/arduino-lessons/lesson-ii-3-inductive-load-control/