ITP PHYSICAL COMPUTING

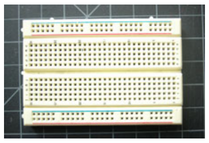
Using a transistor to control high current loads with an Arduino

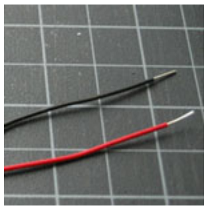
In this tutorial, you'll learn how to control a high-current DC load such as a DC motor or an incandescent light from a microcontroller.

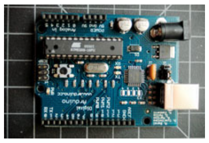
(:toc Table of Contents:)

Parts

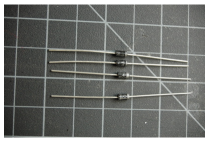
You will need the following parts for this tutorial.


Solderless breadboard

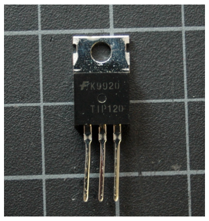

22-AWG hookup wire

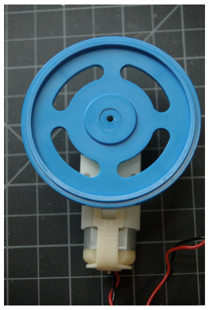

Arduino Microcontroller module

10Kohm potentiometer

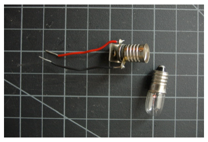

power diodes
(for DC Motor version only)

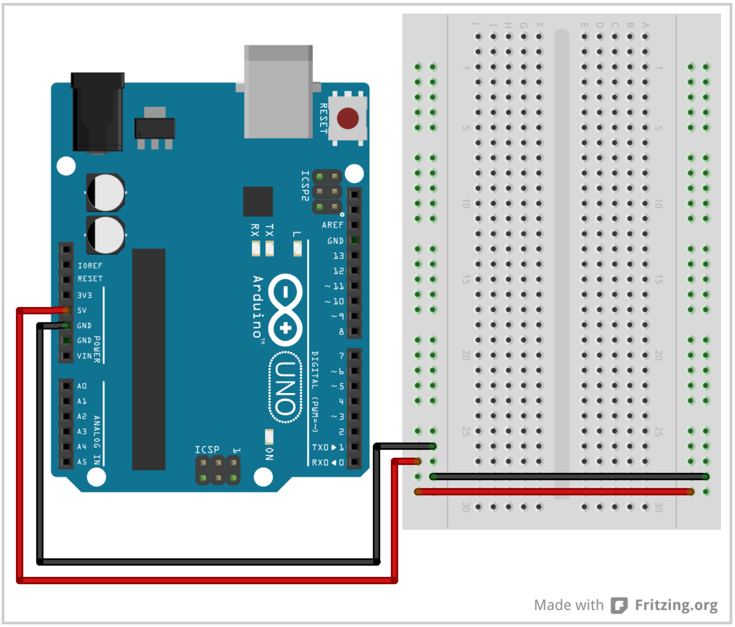
DC power supply


TIP120 transistor


DC Motor

- or -

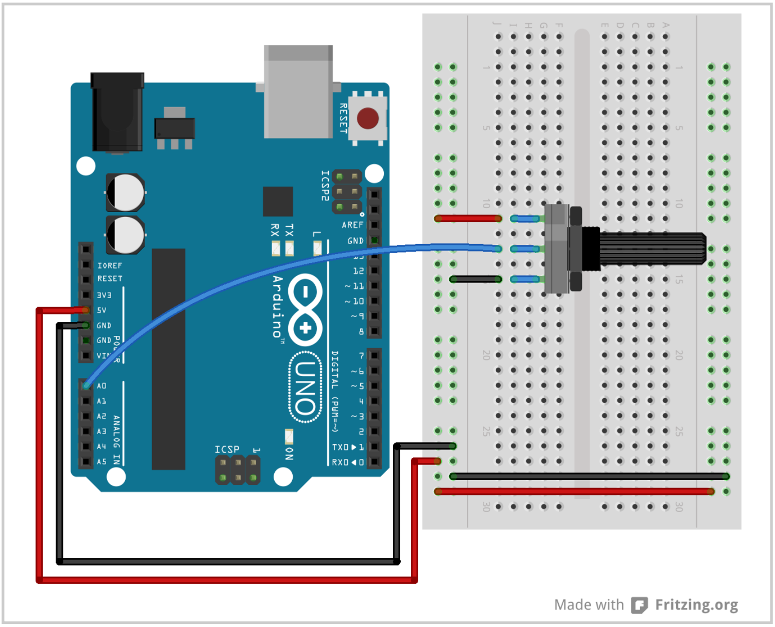

Incandescent lamp and socket

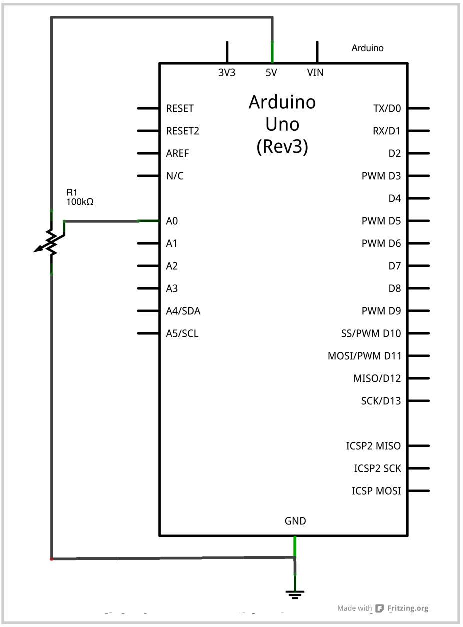


(Diagram made with Fritzing)

Add a potentiometer

Connect a potentiometer to analog in pin 0 of the module:

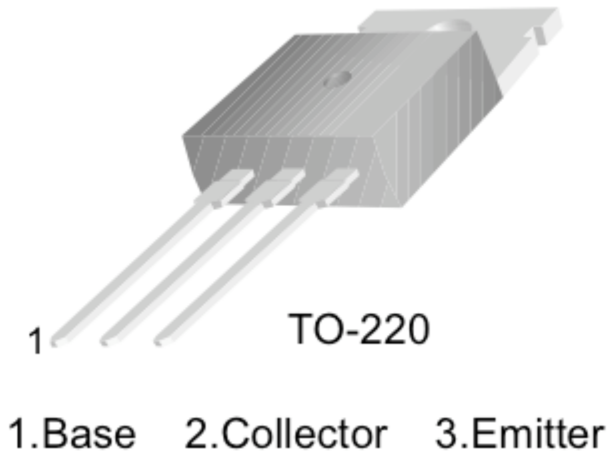




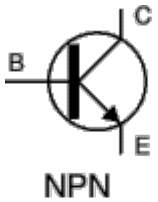
(Diagram made with Fritzing)

Connect a transistor to the microcontroller

The transistor allows you to control a circuit that's carrying higher current and voltage from the microcontroller. It acts as an electronic switch. The one you're using for this lab is an NPN-type transistor called a TIP120. The datasheet for it can be found here. It's designed for switching high-current loads. It has three connections, the base, the collector, and the emitter. The base is connected to the microcontroller's output. The high-current load (i.e. the motor or light) is attached to its power source, and then to the collector of the transistor. The emitter of the transistor is connected to ground.



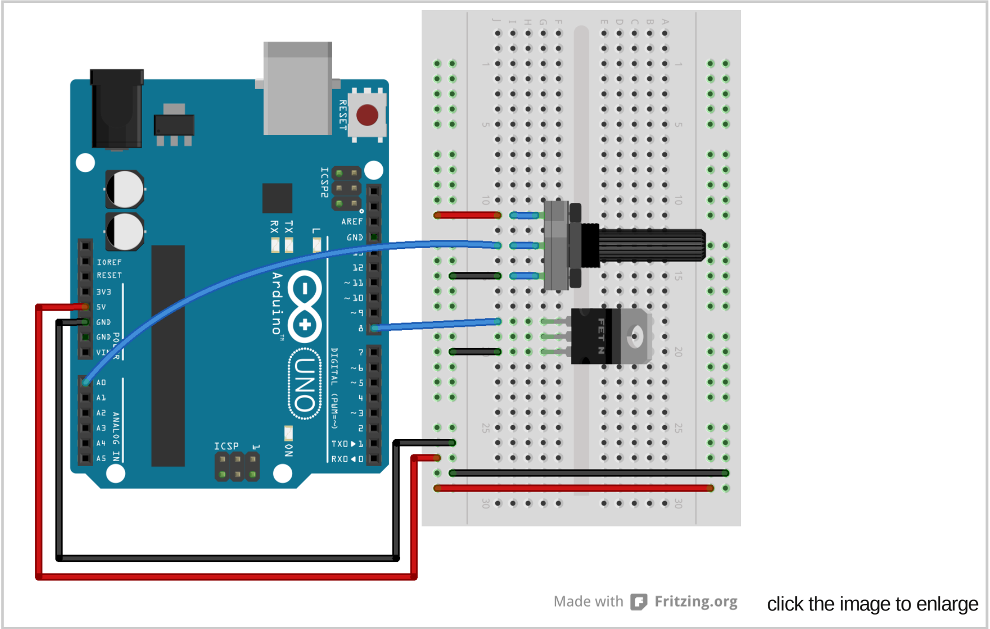
Pinout of a TIP-120 transistor, from left to right: base, collector, emitter.



Note: you can also use an IRF510 or IRF520 MOSFET transistor for this. They have the same pin configuration as the TIP120, and perform similarly. They can handle more amperage and voltage, but are more sensitive to static electricity damage.

The schematic symbol of an NPN transistor where B is the base, C is the collector, and E is the emitter.

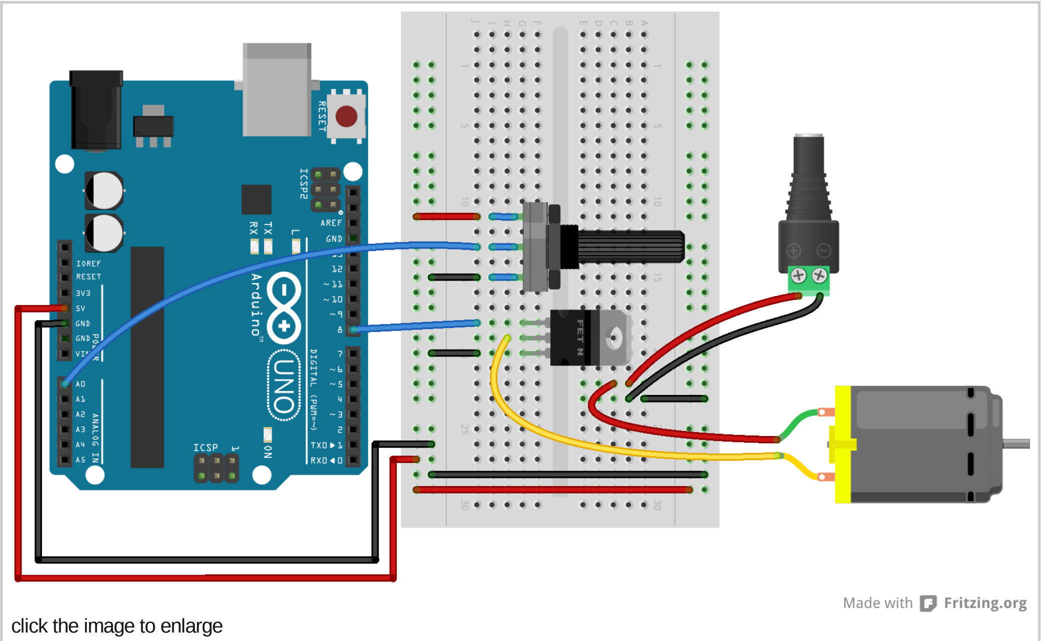
Connect the base to an output pin of the microcontroller, and the emitter to ground like so:



click the image to enlarge

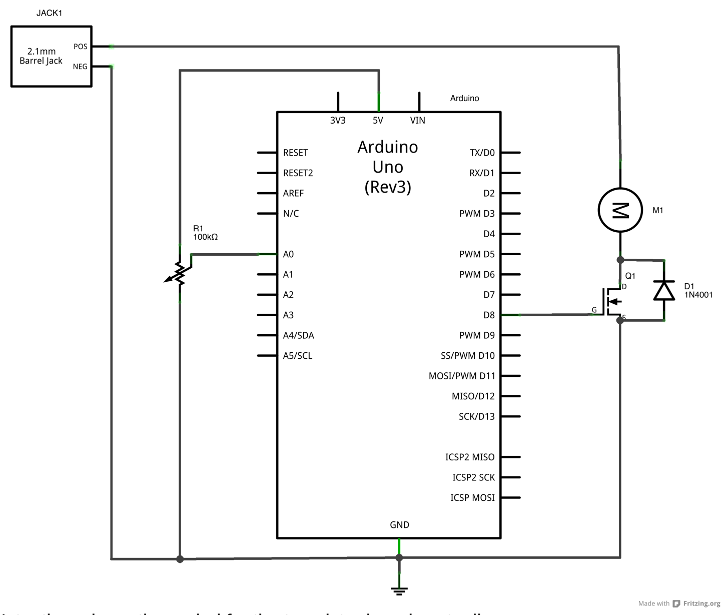
Connect a motor and power supply

Attach a DC motor to the collector of the transistor. Most motors will require more amperage than the microcontroller can supply, so you will need to add a separate power supply as well. If your motor runs on around 9V, you could use a 9V battery. A 5V motor might run on 4 AA batteries. a 12V battery may need a 12V wall wart, or a 12V battery. The ground of the motor power supply should connect to the ground of the microcontroller, on the breadboard.

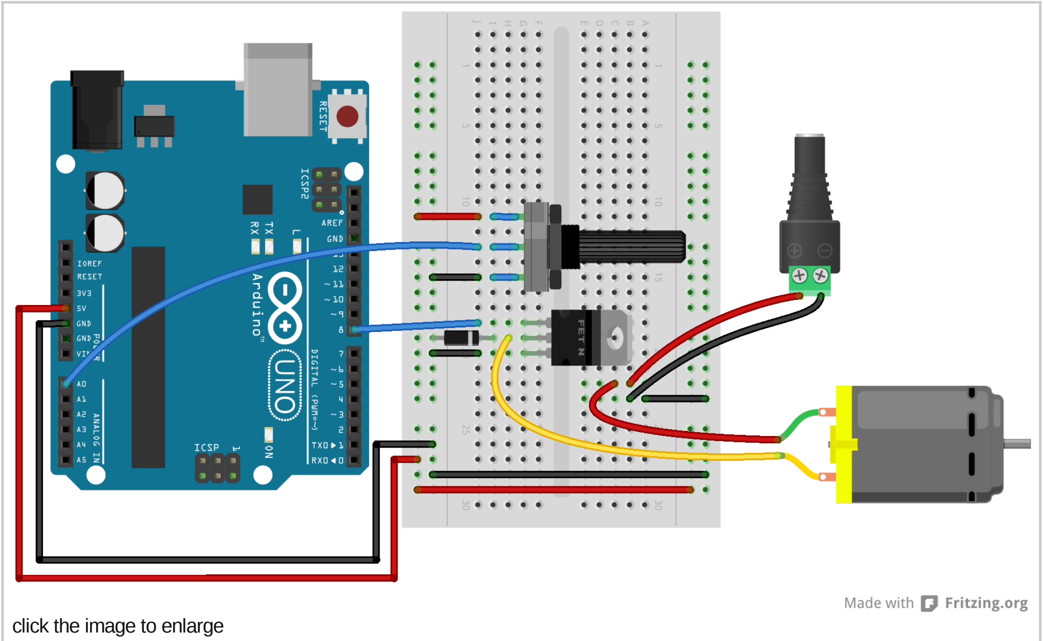


click the image to enlarge

FInally, add diode in parallel with the collector and emitter of the transistor, pointing away from ground. The diode to protects the transistor from back voltage generated when the motor shuts off, or if the motor is turned in the reverse direction.

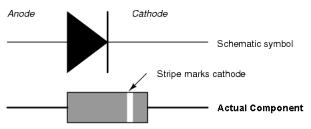


Note: the schematic symbol for the transistor here is actually for an IRF510 MOSFET. But it can be replaced with a TIP120



click the image to enlarge

Be sure to add the diode to your circuit correctly. The silver band on the diode denotes the cathode which is the tip of the arrow in the schematic, like so:

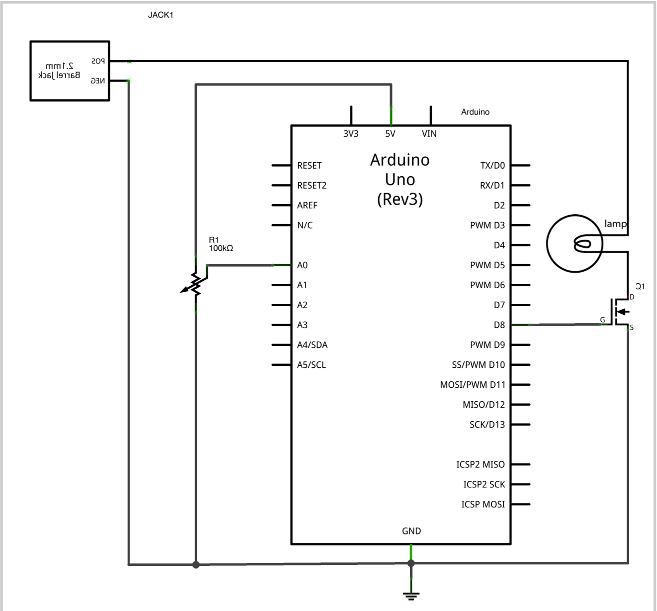


click the image to enlarge

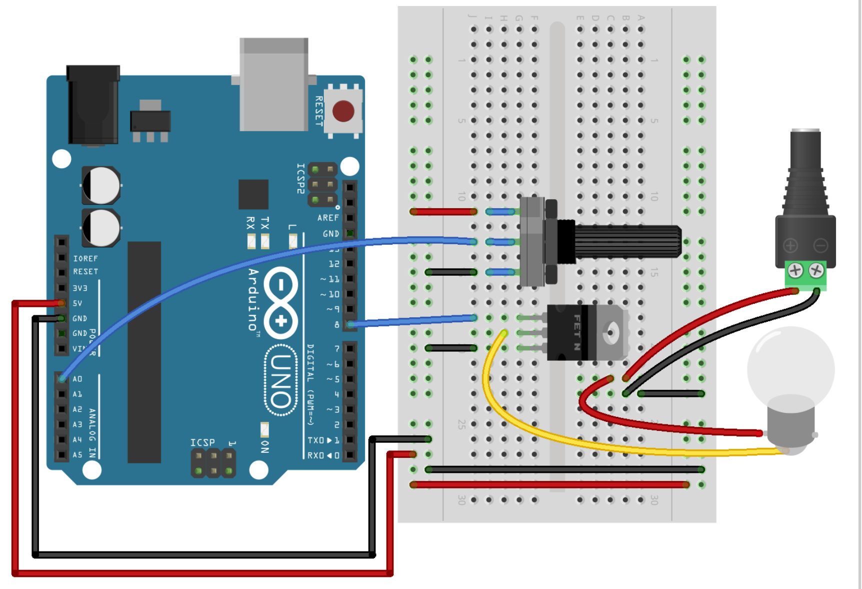
This circuit assumes you're using a 12V motor. If your motor requires a different voltage, make sure to use a power supply that's appropriate. Connect the ground of the motor's supply to the ground of your microcontroller circuit, though, or the circuit won't work properly.

Connect a lamp instead

You could also attach a lamp using a transistor. Like the motor, the lamp circuit below assumes a 12V lamp. Change your power supply accordingly if you're using a different lamp. In the lamp circuit, the protection diode is not needed, since there's no way for the polarity to get reversed in this circuit:



click the image to enlarge



click the image to enlarge

Program the microcontroller

Write a quick program to test the circuit. Your program should make the transistor pin an output in the setup method. Then in the loop, it should turn the motor on and off every second, just like the blink sketch does.

Now that you see it working, try changing the speed of the motor or the intensity of the lamp using the potentiometer.

To do that, read the voltage of the potentiometer using `analogRead()`. Then map the result to a range from 0 to 255 and save it in a new variable. Use that variable to set the speed of the motor or the brightness of the lamp using `analogWrite()`.

Notes

For the motor users:
A motor controlled like this can only be turned in one direction. To be able to reverse the direction of the motor, an H-bridge circuit is required. For more on controlling DC motors with H-bridges, see the DC Motor Control lab

Page last modified on September 21, 2013, at 08:45 AM