ITP Physical Computing

LAB: USING A TRANSISTOR TO CONTROL HIGH CURRENT LOADS WITH AN ARDUINO

last edited 31 August 2014 by Tom Igoe

Introduction

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. Microcontrollers can only output a very small amount of current from their output pins. These pins are meant to send control signals, not to act as power supplies. The most common way to control another direct current device from a microcontroller is to use a transistor. Transistors allow you to control the flow of a high-current circuit from a low-current source.

Contents [hide]

- Video: Transistor Schematics
- Video: Meet the motors
- Video: Pulse-width modulating a transistor to to control a fan motor
- Video: NPN Transistors
- Video: Darlingtons and MOSFETs

What You'll Need to Know

To get the most out of this Lab you should be familiar with the following concepts beforehand. If you're not, review the links below:

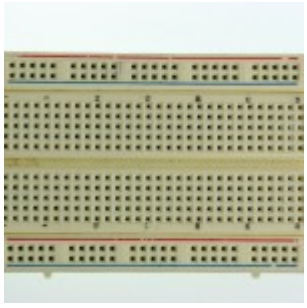
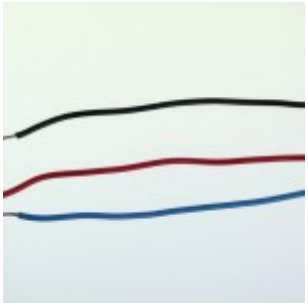
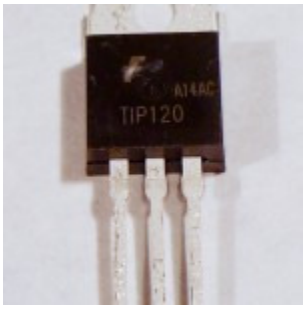
- What is a microcontroller

- Beginning programming terms
- What is a solderless breadboard and how to use one
- Digital Input and output
- Analog Output
- Basic Electronics

- **Safety Warning:** This tutorial shows you how to control high-current loads. This comes with a higher danger of injury from electricity than the earlier tutorials. Please be careful and double-check your wiring before plugging anything in, and never change your wiring while your circuit is powered.

Things You'll Need

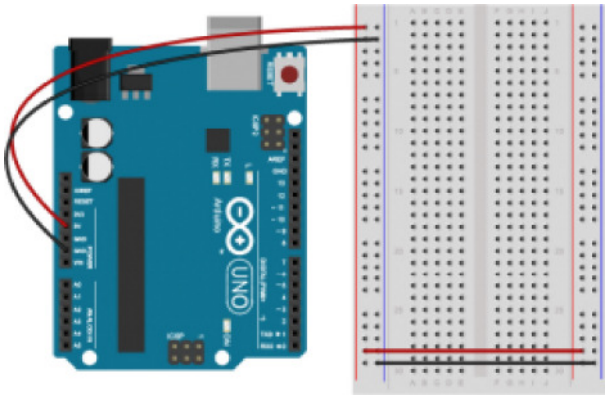
For this lab you'll need:

			
Solderless breadboard	*22-AWG hookup wire*	*Arduino Microcontroller module*	*10Kohm potentiome*
			
Power diodes (for DC Motor version only)	*DC power supply*	*TIP120 transistor*	*DC Motor*
OR			

	Incandescent lamp and socket

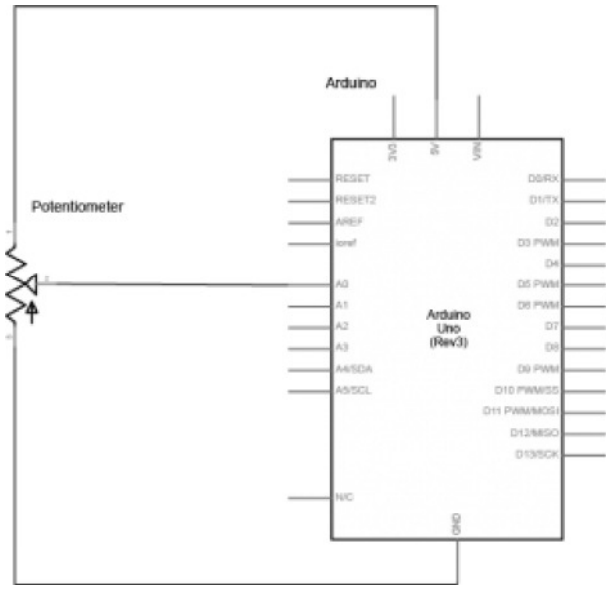
Connect the Breadboard

Connect the breadboard to the Arduino, running 5V and ground to the side rails:

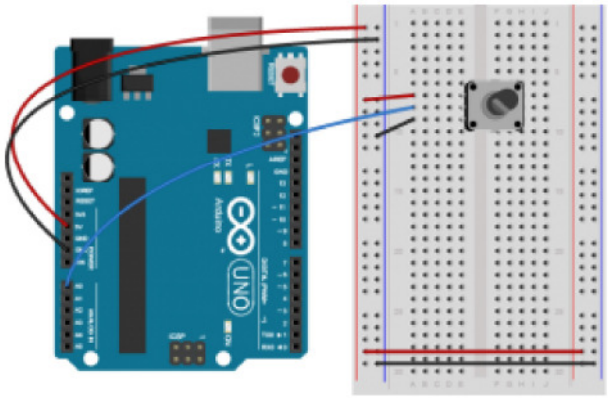


Add a potentiometer

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



Schematic view

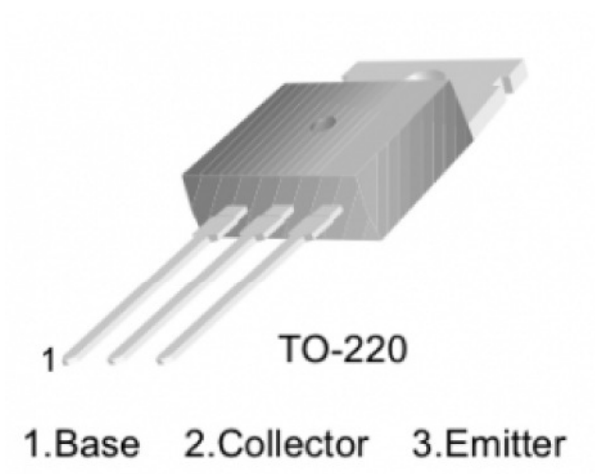


Arduino with Potentiometer

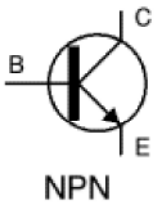
Connect a transistor to the microcontroller

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



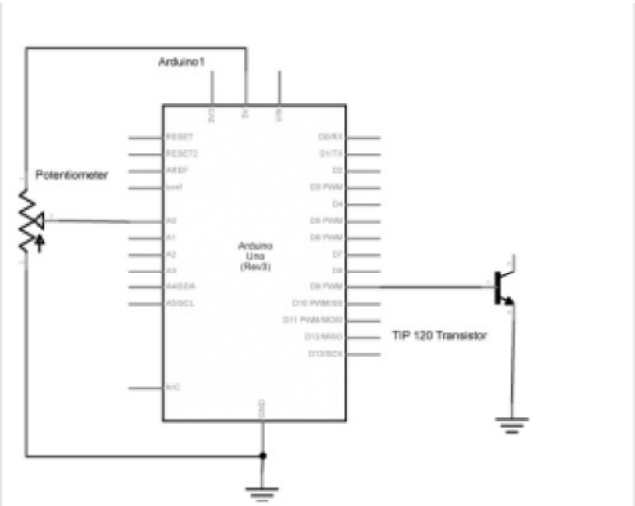
Schematic symbol of a TIP-120 transistor

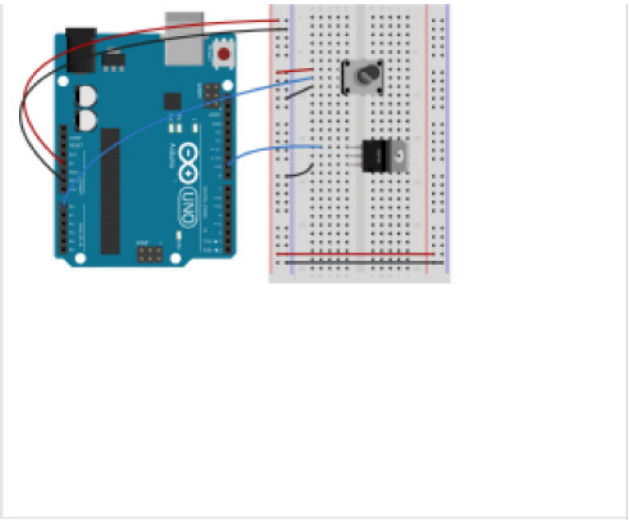
Note: Using MOSFETS instead

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. MOSFETs are grouped into N-Channel and P-Channel, which are equivalent to NPN and PNP bipolar transistors. Here's a quick translation table for the pin names on both:

BIPOLAR TRANSISTOR	MOSFET
Base	Gate
Collector	Drain
Emitter	Source

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

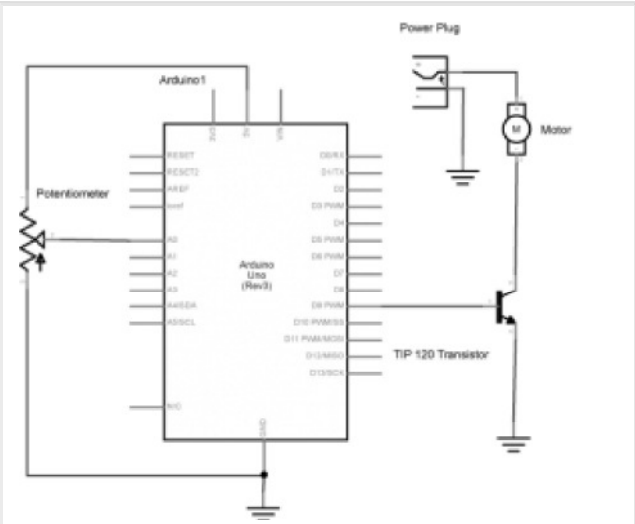


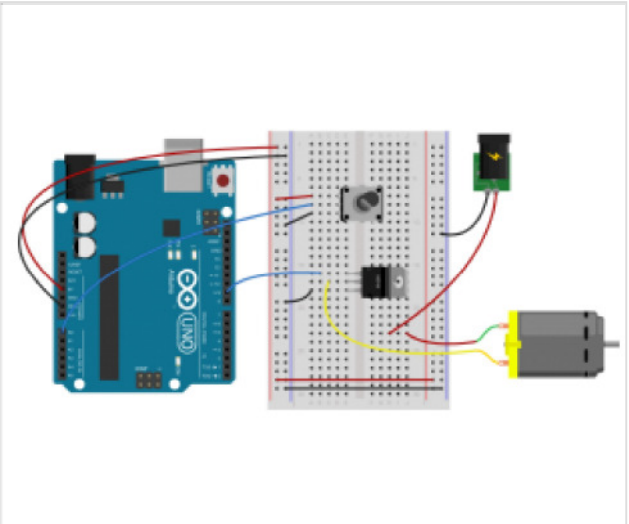


Safety Warning: You can generally connect the base to a microcontroller's pin directly without a current limiting resistor because the current from the pin is low enough. But it's necessary if you're controlling a transistor circuit without a microcontroller.

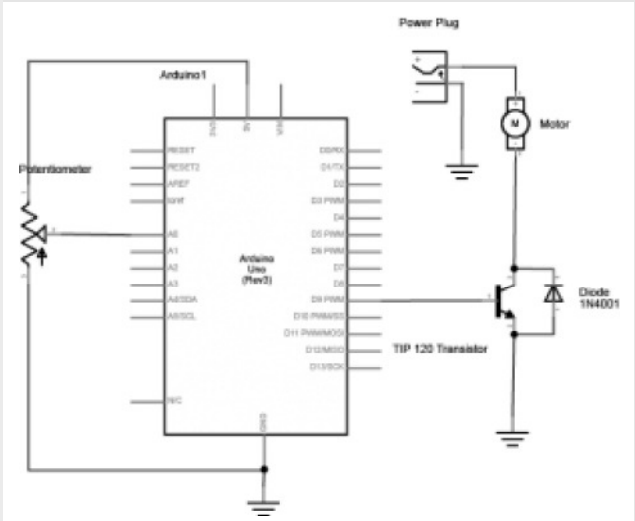
Connect a motor and power supply

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.

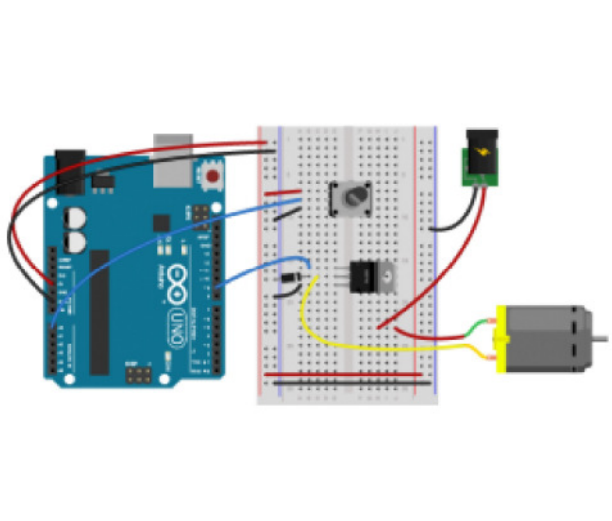




Next, add a diode in parallel with the collector and emitter of the transistor, pointing away from ground. The diode to protects the transistor from back voltage generated when the motor shuts off, or if the motor is turned in the reverse direction.

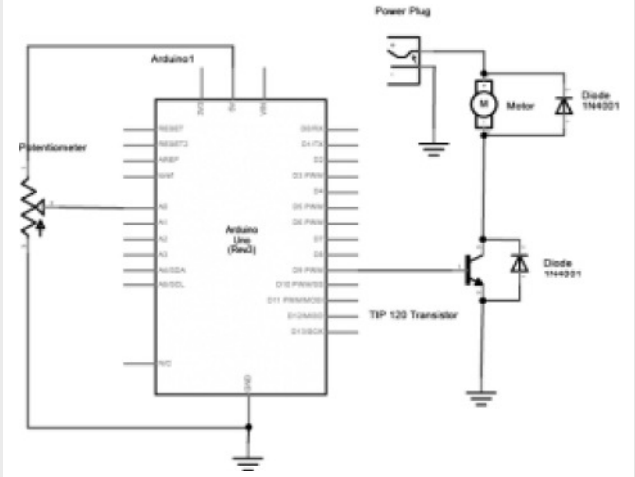


The circuit with protection diode across the transistor

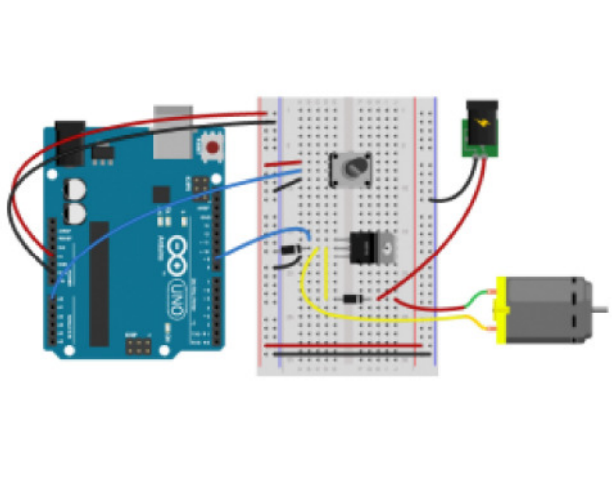


The TIP120 can be replaced with a MOSFET if you prefer.

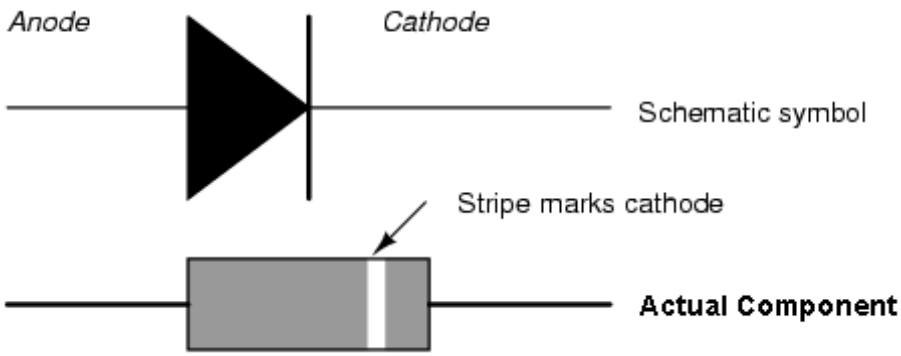
You may also find that adding a diode across the motor helps with back voltage protection as well, particularly when you're running multiple transistor-motor circuits. If you plan to add a diode across the motor, here's the circuit:



A diode across the motor helps with back voltage protection as well



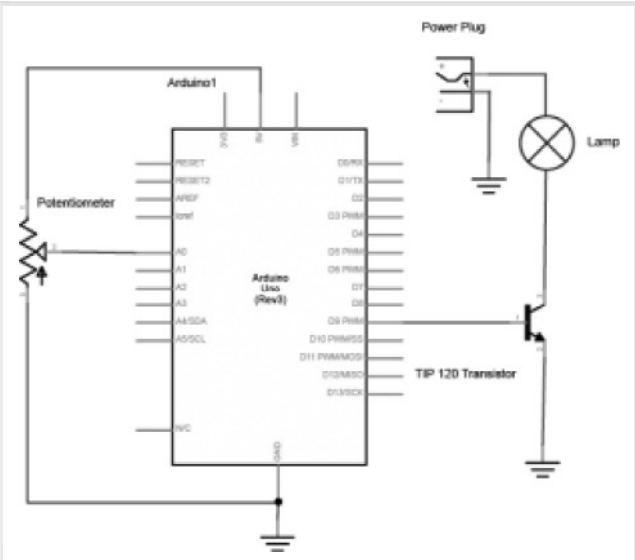
Be sure to add the diode to your circuit correctly. The silver band on the diode denotes the cathode which is the tip of the arrow in the schematic, like so:



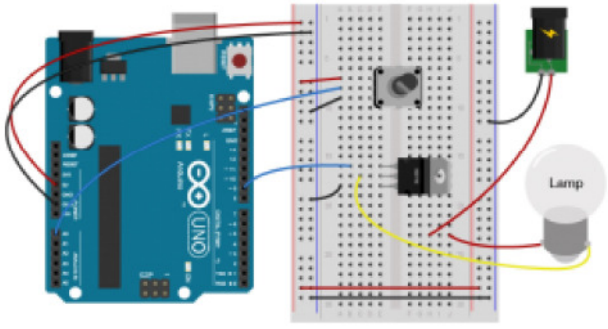
This circuit assumes you're using a 12V motor. If your motor requires a different voltage, make sure to use a power supply that's appropriate. Connect the ground of the motor's supply to the ground of your microcontroller circuit, though, or the circuit won't work properly.

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:



Schematic view



When the motor is replaced with a lamp there is no need for the protection diode

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.

```
const int transistorPin = 9;    // connected to the base of the transist

 void setup() {
   // set  the transistor pin as output:
   pinMode(transistorPin, OUTPUT);
 }

 void loop() {
   digitalWrite(transistorPin, HIGH);
   delay(1000);
   digitalWrite(transistorPin, LOW);
   delay(1000);
 }
```

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()`.

```
const int transistorPin = 9;    // connected to the base of the transist

 void setup() {
   // set  the transistor pin as output:
   pinMode(transistorPin, OUTPUT);
 }

 void loop() {
   // read the potentiometer:
   int sensorValue = analogRead(A0);
   // map the sensor value to a range from 0 - 255:
   int outputValue = map(sensorValue, 0, 1023, 0, 255);
   // use that to control the transistor:
   analogWrite(transistorPin, outputValue);
 }
```

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