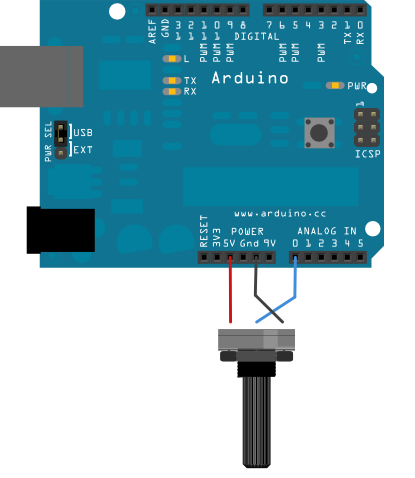
1. Analog Input

In this example we use a variable resistor (potentiometer), we read its value using one analog input of an Arduino or Genuino board and we change the blink rate of the built-in LED accordingly. The resistor's analog value is read as a voltage because this is how the analog inputs work.

Hardware Required

* Arduino or Genuino Board
* Potentiometer
* built-in LED on pin 13 *or* 220 ohm resistor and red LED

Circuit



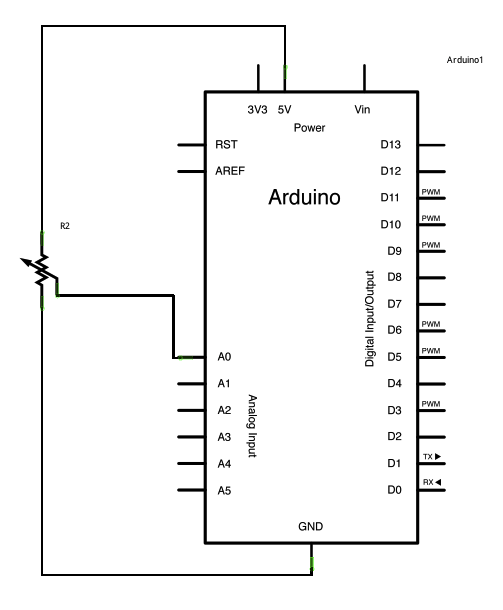
Connect three wires to the Arduino or Genuino board. The first goes to ground from one of the outer pins of the potentiometer. The second goes from 5 volts to the other outer pin of the potentiometer. The third goes from analog input 0 to the middle pin of the potentiometer.

For this example, it is possible to use the board's built in LED attached to pin 13. To use an additional LED, attach its longer leg (the positive leg, or anode), to digital pin 13 in series with the 220 ohm resistor, and it's shorter leg (the negative leg, or cathode) to the ground (GND) pin next to pin 13.

The voltage measured (Vout) follows this formula:

Vout = Vin \* (R2/(R1+R2))

Schematic



R1

Code

At the beginning of this sketch, the variable sensorPin is set to to analog pin 0, where your potentiometer is attached, and ledPin is set to digital pin 13. You'll also create another variable, sensorValue to store the values read from your sensor.

The [analogRead()](https://www.arduino.cc/en/Reference/AnalogRead) command converts the input voltage range, 0 to 5 volts, to a digital value between 0 and 1023. This is done by a circuit inside the microcontroller called an *analog-to-digital converter* or *ADC*.

By turning the shaft of the potentiometer, you change the amount of resistance on either side of the center pin (or wiper) of the potentiometer. This changes the relative resistances between the center pin and the two outside pins, giving you a different voltage at the analog input. When the shaft is turned all the way in one direction, there is no resistance between the center pin and the pin connected to ground. The voltage at the center pin then is 0 volts, and analogRead() returns 0. When the shaft is turned all the way in the other direction, there is no resistance between the center pin and the pin connected to +5 volts. The voltage at the center pin then is 5 volts, and analogRead()returns 1023. In between, analogRead() returns a number between 0 and 1023 that is proportional to the amount of voltage being applied to the pin.

That value, stored in sensorValue, is used to set a delay() for your blink cycle. The higher the value, the longer the cycle, the smaller the value, the shorter the cycle. The value is read at the beginning of the cycle, therefore the on/off time is always equal.

int sensorPin = A0;    *// select the input pin for the potentiometer*  
int ledPin = 13;      *// select the pin for the LED*  
int sensorValue = 0;  *// variable to store the value coming from the sensor*  
  
void **setup**() {  
  *// declare the ledPin as an OUTPUT:*  
  pinMode(ledPin, OUTPUT);  
}  
  
void **loop**() {  
  *// read the value from the sensor:*  
  sensorValue = analogRead(sensorPin);  
  *// turn the ledPin on*  
  digitalWrite(ledPin, HIGH);  
  *// stop the program for <sensorValue> milliseconds:*  
  delay(sensorValue);  
  *// turn the ledPin off:*  
  digitalWrite(ledPin, LOW);  
  *// stop the program for for <sensorValue> milliseconds:*  
  delay(sensorValue);  
}

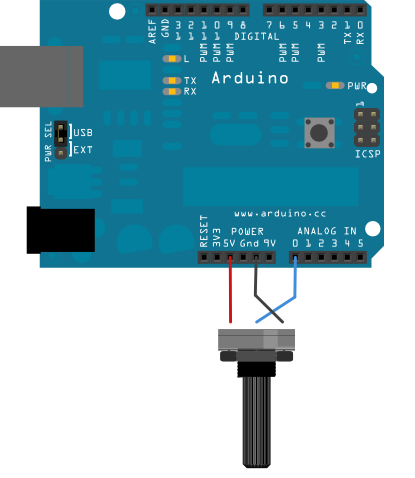
2. On / Off the Led use If Statement

We'll use if statements all the time. The example below turns on an LED on pin 13 (the built-in LED on many Arduino boards) if the value read on an analog input goes above a certain threshold.

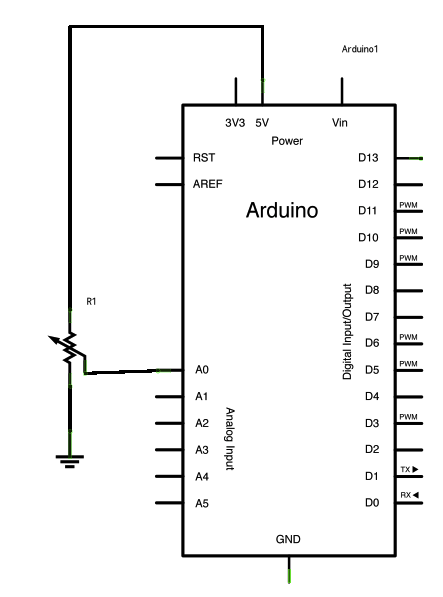
Hardware Required

* Arduino or Genuino Board
* Potentiometer or variable resistor

Circuit



Schematic



Code

In the code below, a variable called analogValue is used to store the data collected from a potentiometer connected to the board on analogPin 0. This data is then compared to a threshold value. If the analog value is found to be above the set threshold the built-in LED connected to digital pin 13 is turned on. If analogValue is found to be < (less than) threshold, the LED remains off.

*// These constants won't change:*  
const int analogPin = A0;    *// pin that the sensor is attached to*  
const int ledPin = 13;       *// pin that the LED is attached to*  
const int threshold = 400;   *// an arbitrary threshold level that's in the range of the analog input*  
  
void **setup**() {  
  *// initialize the LED pin as an output:*  
  pinMode(ledPin, OUTPUT);  
  *// initialize serial communications:*  
  // Serial.begin(9600);  
}  
  
void **loop**() {  
  *// read the value of the potentiometer:*  
  int analogValue = analogRead(analogPin);  
  
  *// if the analog value is high enough, turn on the LED:*  
  if (analogValue > threshold) {  
    digitalWrite(ledPin, HIGH);  
  } else {  
    digitalWrite(ledPin, LOW);  
  }  
  
  *// print the analog value:*  
  // Serial.println(analogValue);  
  delay(1);        *// delay in between reads for stability*  
}