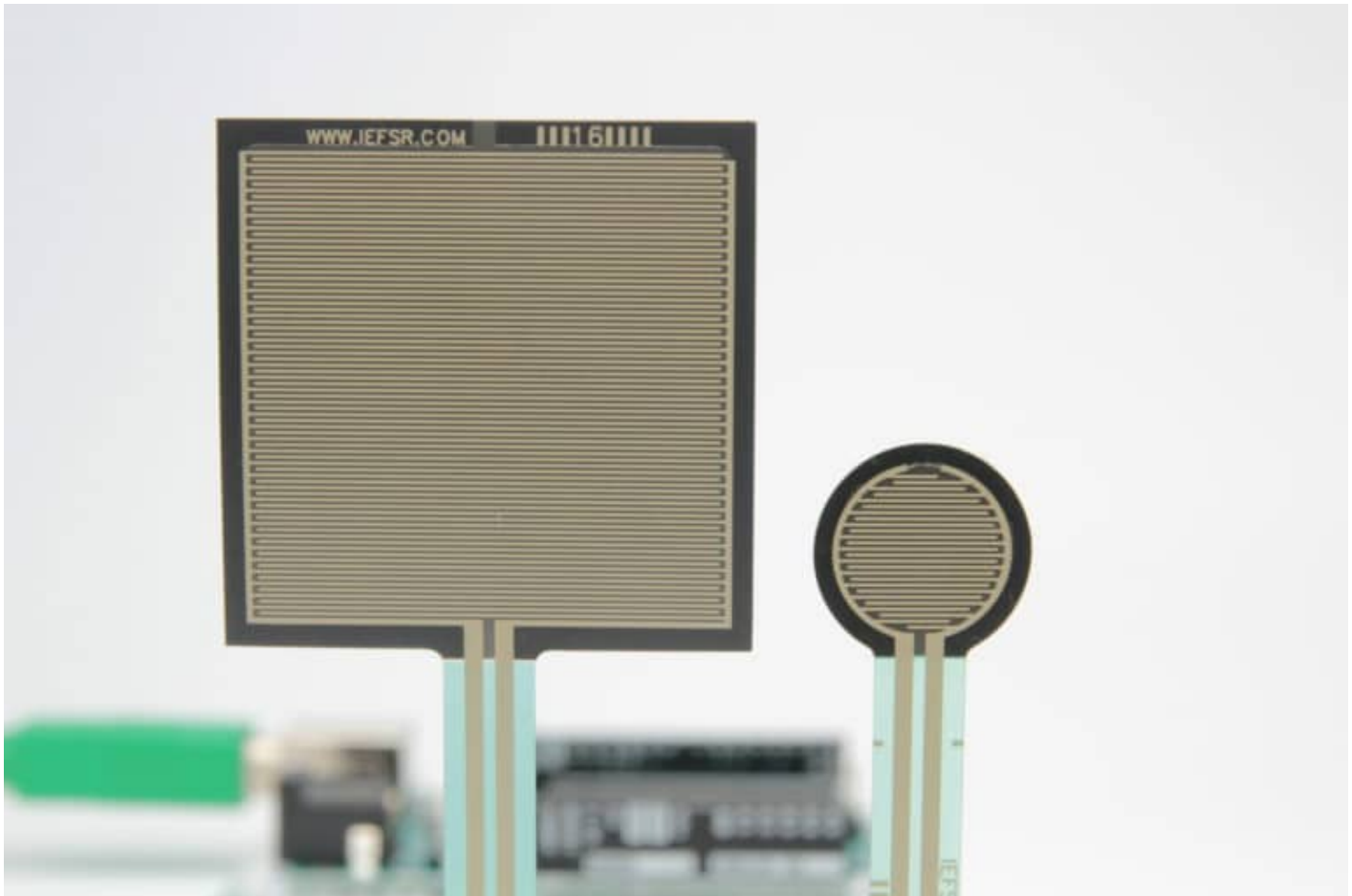


Makerguides.com

Force Sensing Resistor (FSR) with Arduino Tutorial

Written by Benne de Bakker (<https://www.makerguides.com/author/benne-de-bakker/>)



FSRs are super robust pressure sensors that are used in all kinds of industries. You will find them in electronic drums, mobile phones, handheld gaming devices and many more portable electronics. These sensors are easy to use and great for sensing pressure.

In this tutorial you will learn how an FSR works and how to use it with Arduino. I have included 3 examples with a wiring diagram and code so you can start experimenting with your sensor.




First I will show you the basic operation of the sensor. Next, we will look at using this sensor as a toggle switch. Lastly, I will show you how you can use LEDs to show the amount of pressure applied to the sensor.



If you would like to learn about other sensor, check out the articles below:

- How to use a SHARP GP2Y0A21YK0F IR Distance Sensor with Arduino (<https://www.makerguides.com/sharp-gp2y0a21yk0f-ir-distance-sensor-arduino-tutorial/>)
- How to use a SHARP GP2Y0A710K0F IR Distance Sensor with Arduino (<https://www.makerguides.com/sharp-gp2y0a710k0f-ir-distance-sensor-arduino-tutorial/>)
- How to use an HC-SR04 Ultrasonic Distance Sensor (<https://www.makerguides.com/hc-sr04-arduino-tutorial/>)

Supplies

Hardware components

| | | | |
|---|---|---|---|
|  | FSR 402 sensor (https://amzn.to/2IWmzU8) (round) | × | Amazon (https://amzn.to/2IWmzU8) |
|  | FSR 406 sensor (https://amzn.to/2IWmzU8) (square) | × | Amazon (https://amzn.to/2IWmzU8) |
|  | Arduino Uno Rev3 (https://amzn.to/374aJjX) | × | Amazon (https://amzn.to/374aJjX) |
| | Breadboard | × | Amazon |

| | | | |
|---|---|---------|---|
| | (https://amzn.to/2MQHicc) | 1 | (https://amzn.to/2MQHicc) |
| | Jumper wires (https://amzn.to/2EG9wDc) | ~ 10 | Amazon (https://amzn.to/2EG9wDc) |
|  | Resistor assortment box (https://amzn.to/2X4liy0) (see wiring for values) | × 1 | Amazon (https://amzn.to/2X4liy0) |
|  | LEDs (https://amzn.to/2RIVQgu) | ~ 10 | Amazon (https://amzn.to/2RIVQgu) |
| | USB cable type A/B (https://amzn.to/34SBuXf) | × 1 | Amazon (https://amzn.to/34SBuXf) |

Tools



Multimeter (<https://amzn.to/2IWncwY>)

Amazon

(<https://amzn.to/2IWncwY>)

Software



Arduino IDE (<https://www.arduino.cc/en/Main/Software>)

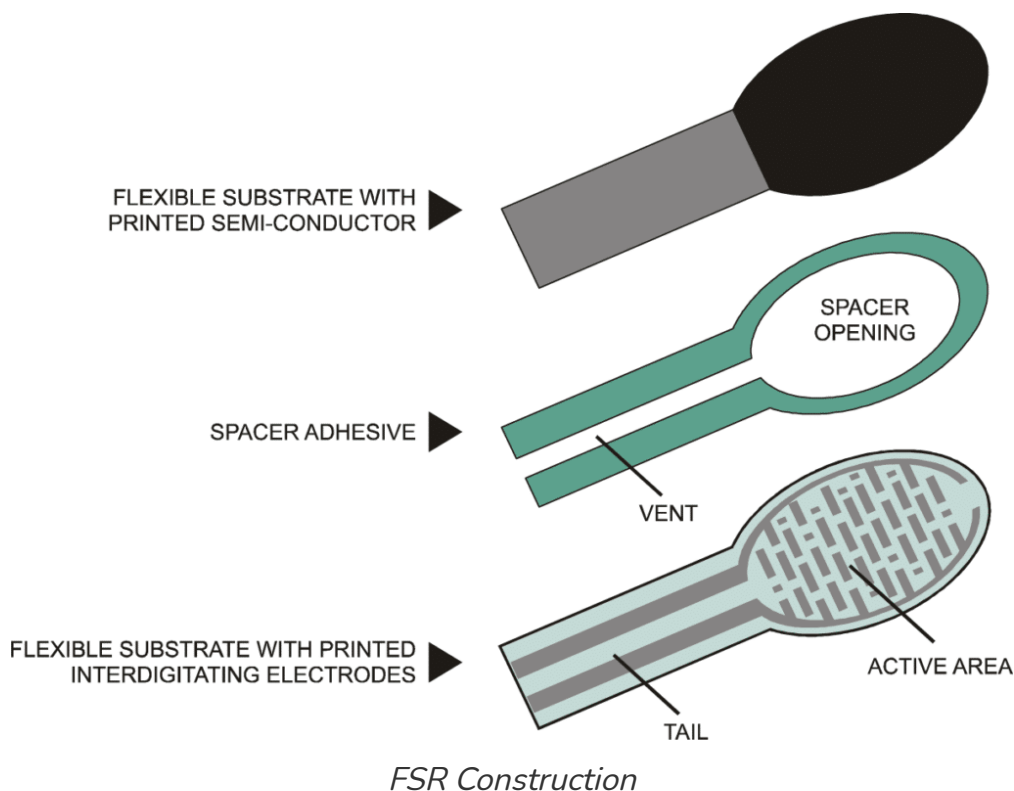
Makerguides.com is a participant in the Amazon Services LLC Associates Program, an affiliate advertising program designed to provide a means for sites to earn advertising fees by advertising and linking to products on Amazon.com.

How does an FSR work?

The resistance of an FSR depends on the pressure that is applied to the sensing area. The more pressure you apply, the lower the resistance. The resistance range is actually quite large: $> 10 \text{ M}\Omega$ (no pressure) to $\sim 200 \Omega$ (max pressure). Most FSRs can sense force in the range of 100 g to 10 kg.

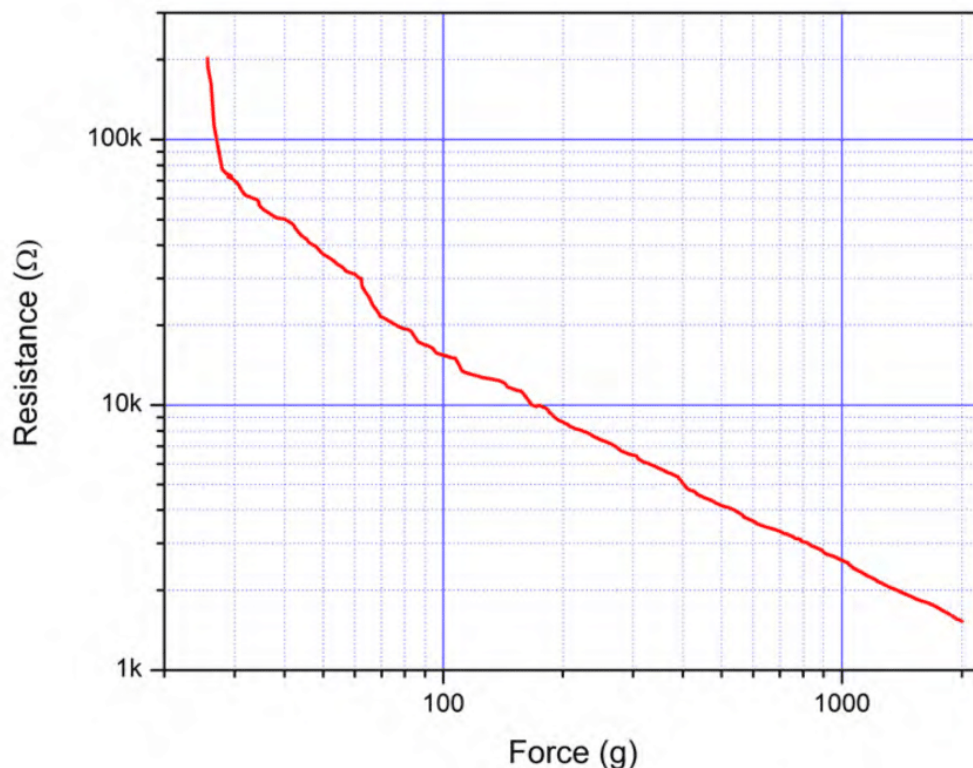
Basic construction

An FSR consists of two membranes and a spacer adhesive. The conducting membranes are separated by a thin air gap when no pressure is applied. One of the membranes contains two traces running from the tail to the sensing area (the round part). These traces are woven together, but not touching. The other membrane is coated with a conducting ink. When you push on the sensor, the ink shorts the two traces together with a resistance that depends on the pressure.



How to read an FSR?

The graph below displays the resistance vs force curve for the FSR 402 sensor. Note that the data is plotted on logarithmic scales. The response is not linear! As you can see there is a huge drop in resistance when a small amount of pressure is applied. After that, the resistance is inversely proportional to the applied force. At around 10 kg (not shown in the graph) the sensor is saturated and an increase in force yields little to no decrease in resistance.

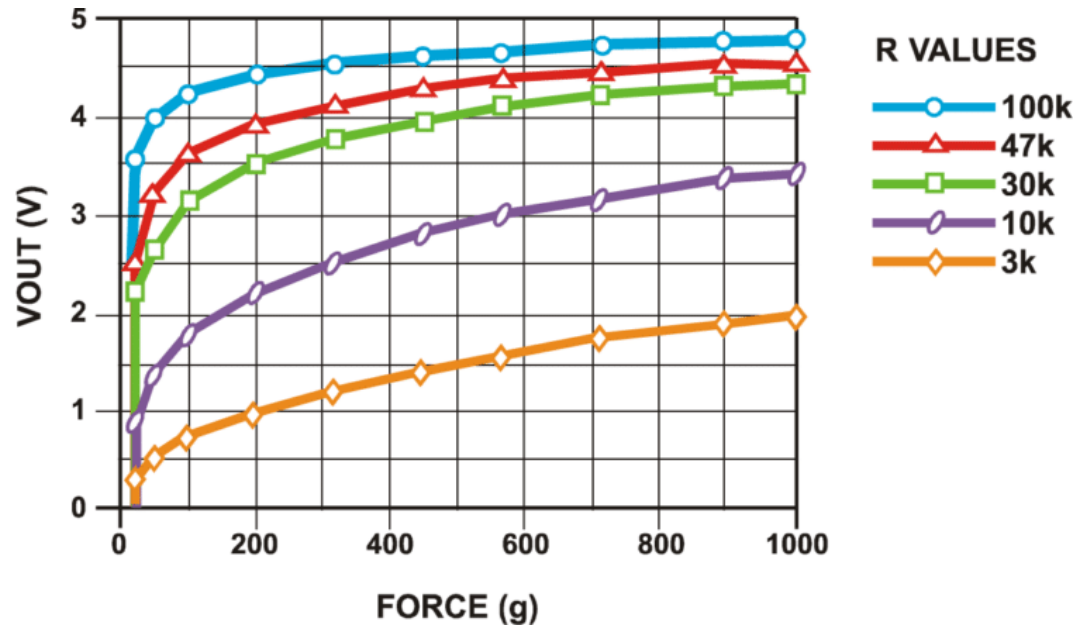
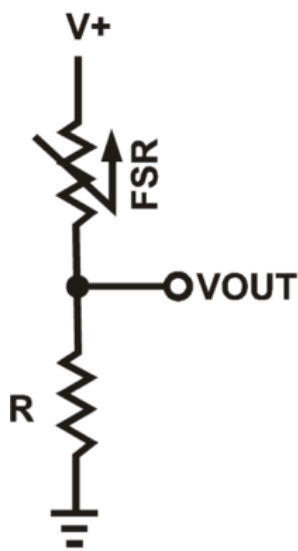


Resistance vs Force curve for FSR 402

Voltage divider circuit

In order to measure the applied force with an Arduino, you will need to build a voltage divider circuit with the FSR and a pull-down resistor. This circuit creates a variable voltage output that can be read by the ADC (analog to digital converter) input of the

microcontroller.



Voltage divider circuit and V_{out} vs Force curves for different R values. Data represents output for Interlink 402 FSR and $V+$ equal to 5 V. Reference: Interlink integration guide.

Selecting the right size resistor to match your sensor can be a bit tricky and depends on the force range you want to measure.

The graph above shows the V_{out} vs Force curves for different values of R (the pull-down resistor). A **10 k Ω resistor** works well if you want to use the sensor over its entire force range (100 g to 10 kg). I like to buy these assortment boxes from Amazon (<https://amzn.to/2X4liy0>) so I always have a range of resistors on hand.

Calculation example

The output voltage (V_{out}) that we measure with the Arduino is described by the following equation:

$$V_{out} = V_{cc} \times R / (R + R_{fsr})$$

So the voltage is inversely proportional to the FSR resistance. Note that the output voltage you measure is the voltage drop across the pull-down resistor, not across the FSR.

When no force is applied, the FSR resistance will be really high, take 10 M Ω as an example. I used a 10 k Ω pull-down resistor and a Vcc of 5 V for this tutorial, which results in the following output when no force is applied:

$$V_{out} = 5 \text{ V} \times 10 \text{ k}\Omega / (10 \text{ k}\Omega + 10 \text{ M}\Omega) = 0.005 \text{ V}$$

So almost 0 V. If you press really hard on the FSR, the resistance will go down to roughly 200 Ω . This results in the following output voltage:

$$V_{out} = 5 \text{ V} \times 10 \text{ k}\Omega / (10 \text{ k}\Omega + 200 \Omega) = 4.9 \text{ V}$$

As you can see, you should be able to measure an output voltage between 0 and 4.9V depending on the amount of force you apply to the sensor.

FSR Specifications

The technology used in FSRs is patented by Interlink Electronics (<https://www.interlinkelectronics.com/>) which has been in operation since 1985. The most common types of FSR that you will find are the Interlink FSR 402 (<https://amzn.to/2TDmdVn>) and 406 (<https://amzn.to/2WOei9E>).

This are the specifications of the round 402 sensor that I used in this tutorial.

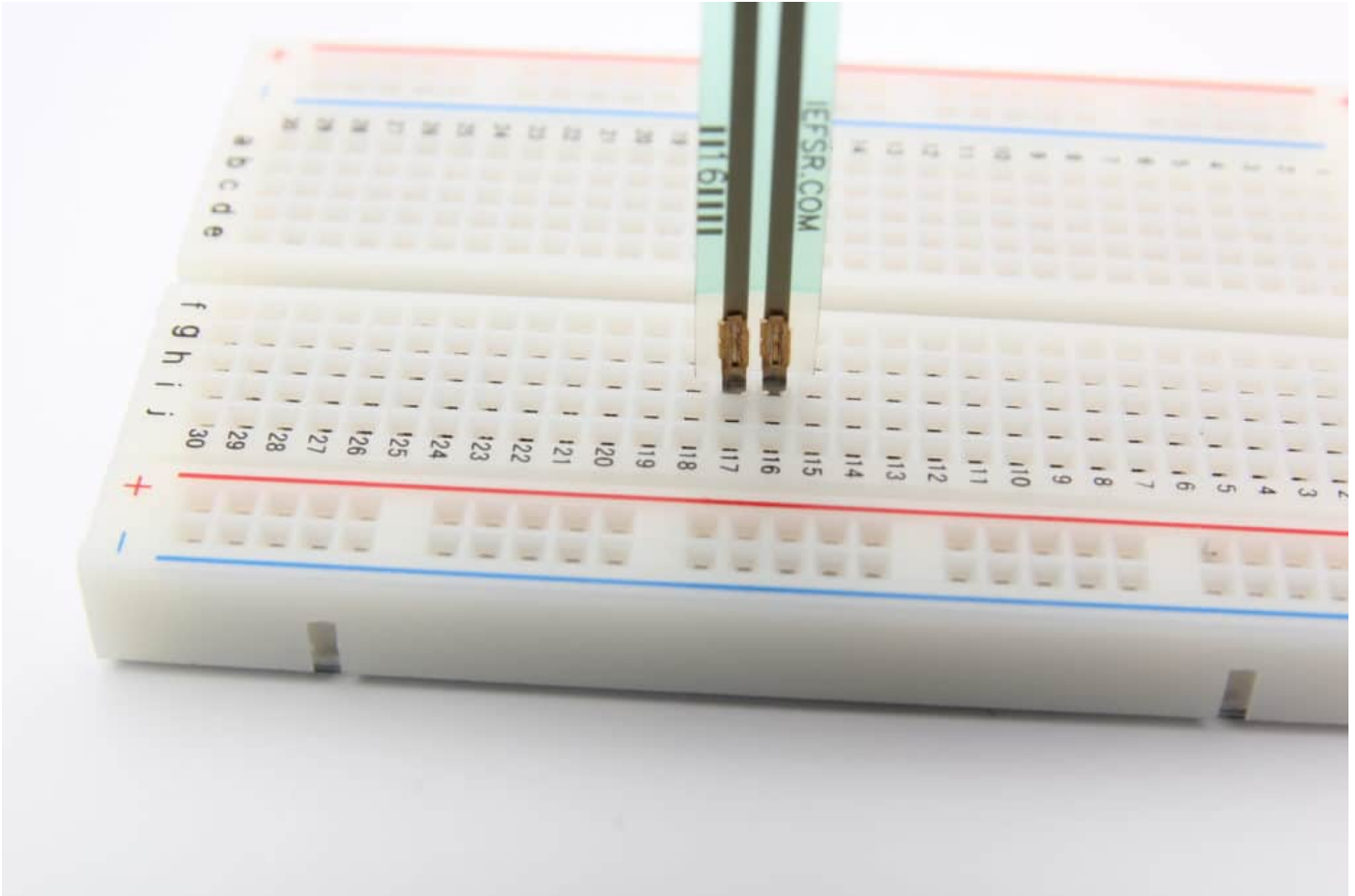
FSR 402 Specifications

| | |
|-------------------------|--|
| Actuation Force | ~0.1N minimum |
| Force Sensitivity Range | ~0.1N – 100N |
| Resistance Range | >10 M Ω (open circuit) – ~200 Ω |
| Force Resolution | Continuous (analog) |
| Force Repeatability | \pm 6% |
| Active Area | \varnothing 12.7 mm |
| Nominal Thickness | 0.55 mm |
| Switch Travel | 0.15 mm |
| Lifetime | > 10 million actuations |
| Power Supply | Any! Uses less than 1mA of current, depending on the resistor used in the voltage divider. |
| Cost | Check price (https://amzn.to/2ZB2wn8) |

For more information, you can check out the datasheet here. It also includes data for the other sensors of the 400 Series.

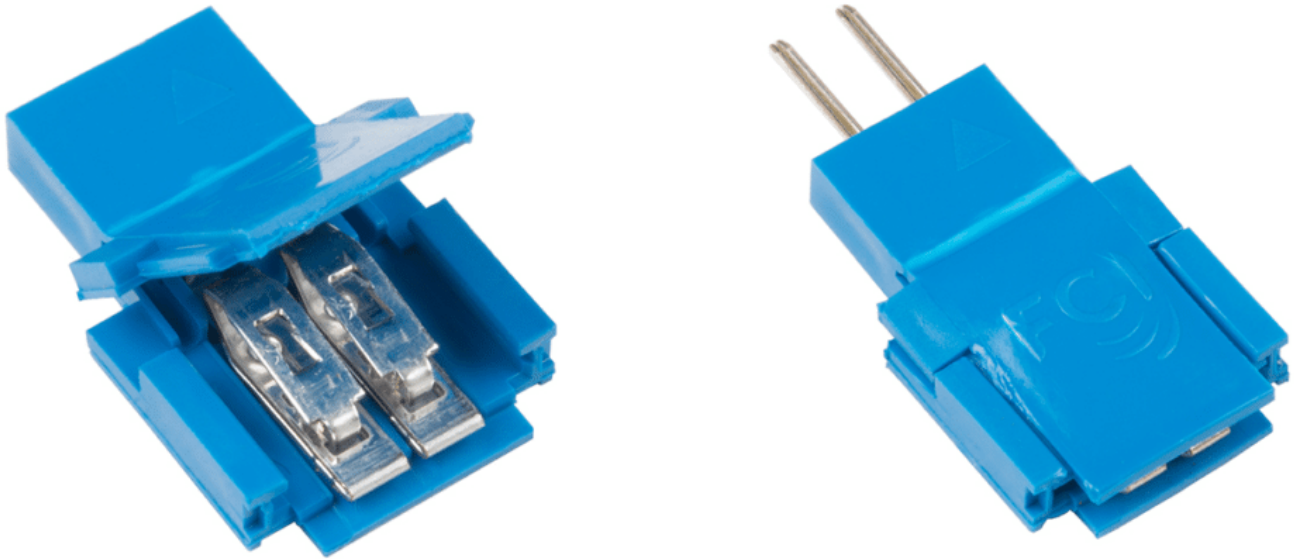
FSR 400 Series Datasheet 

Connecting to an FSR



You can easily connect to an FSR by using a breadboard.

The easiest way to connect to an FSR is to use a breadboard. This works great for prototyping and testing. If you need a more permanent solution, I highly recommend the Amphenol FCI Clincher Connector (<https://amzn.to/2ZYlbWw>). You can just clamp these connectors around the silver traces of the connector and easily attach jumper or dupont cables.



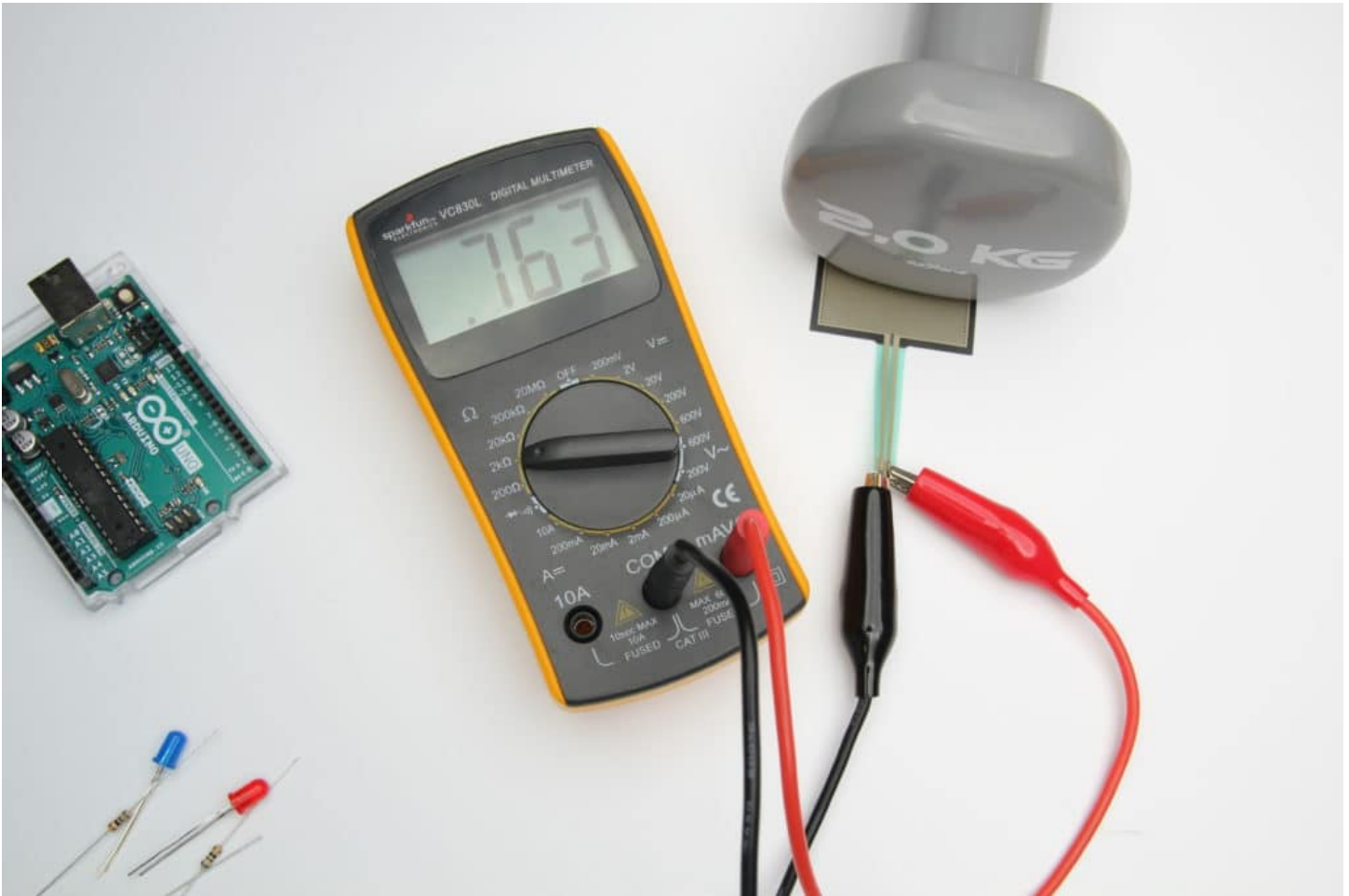
Amphenol FCI Clincher Connector

Warning

It is NOT recommended to solder directly to the exposed silver traces of the sensor. The substrate will melt during soldering and the solder joint won't hold. Do not kink or crease the tail of the FSR if you are bending it; this can cause breaks in the printed silver traces. Interlink suggests a minimum bending radius of 2.5 mm.

Testing an FSR

The easiest way to see if your FSR is working correctly is to connect it to a multimeter. I used alligator test leads (<https://amzn.to/2GhyWdg>) to connect the multimeter to the exposed leads of the sensor. Put your sensor in resistance (Ω) measuring mode and you should see the resistance value change when you press on the sensor.



Measuring the resistance change when applying a load.

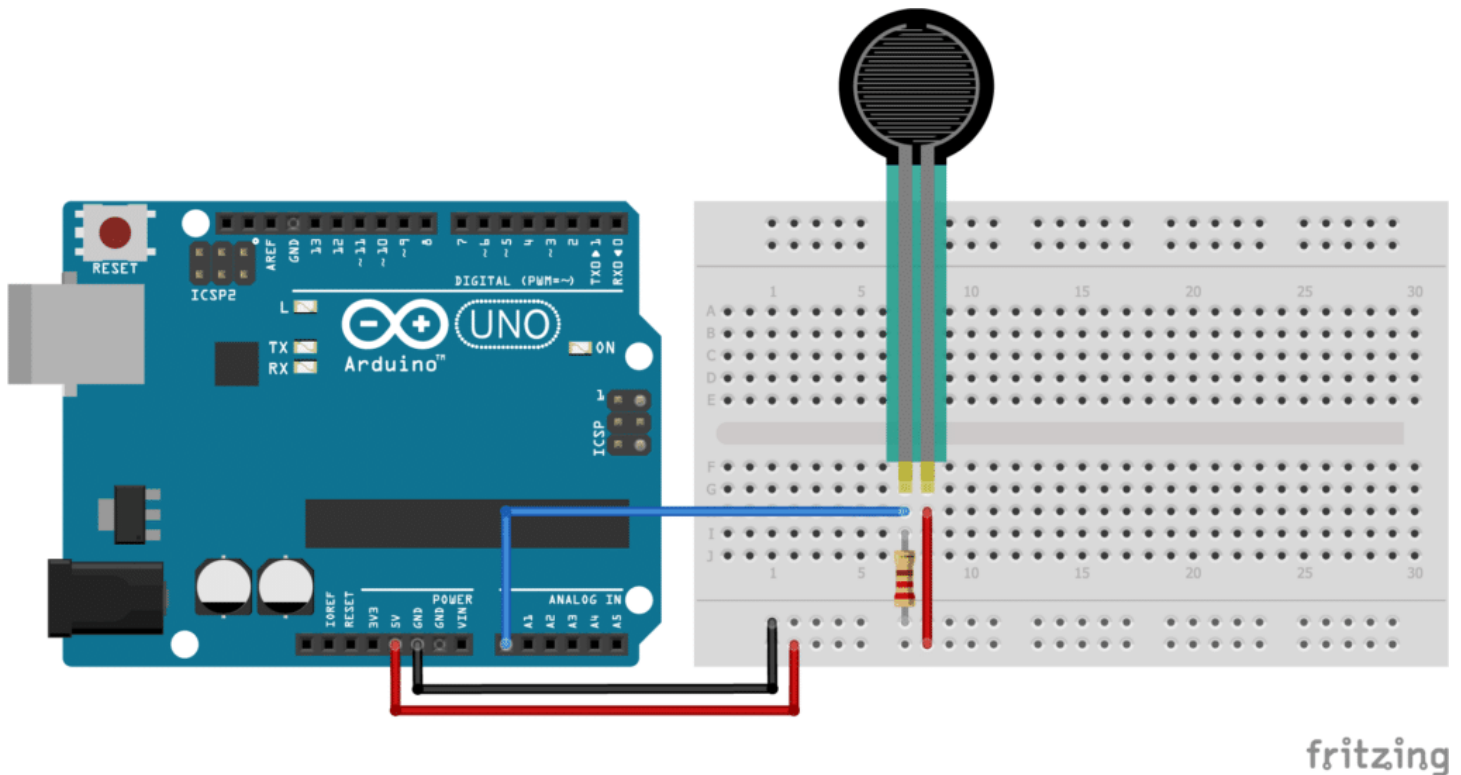
Because the resistance range is so big (200 k Ω to 200 Ω) it's best to use a multimeter with autorange function. If you don't have one of those, just play around with the range settings. 200 k Ω should enable you to see most of the range.

Wiring – Connecting a Force Sensing Resistor (FSR) to Arduino UNO

Now that you know the sensor is working correctly, it is time to connect it to the Arduino. We will be using a breadboard and jumper wires, as this is the easiest way to prototype a circuit.

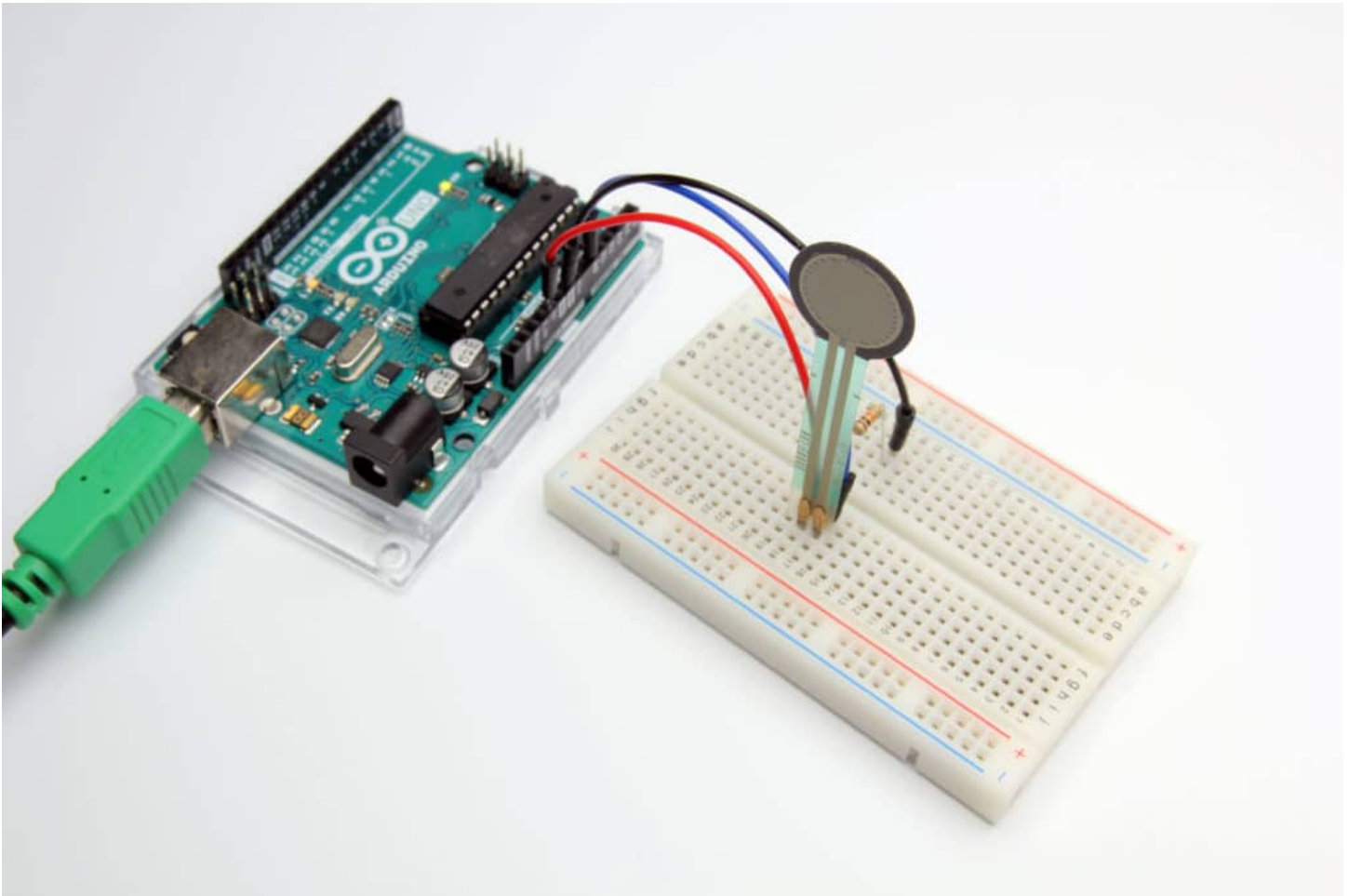
As mentioned in the introduction, you need to create a circuit with a 10 k Ω pulldown resistor.

The wiring diagram below shows you how to connect the FSR sensor to the Arduino. Note that an FSR is non-polarized, just like normal resistors. There is no positive or negative side, just connect them in the orientation you want.



FSR with Arduino UNO wiring diagram

Connect one of the leads of the FSR to power (5 V, but 3.3 V works just fine too) and the other lead to the analog in of the Arduino (A0). The 10 k Ω pulldown resistor gets connected between GND and A0.



1. FSR with Arduino example code – Analog voltage reading

Now that you have wired up the sensor, you can upload the following example code using the Arduino IDE (<https://www.arduino.cc/en/main/software>).

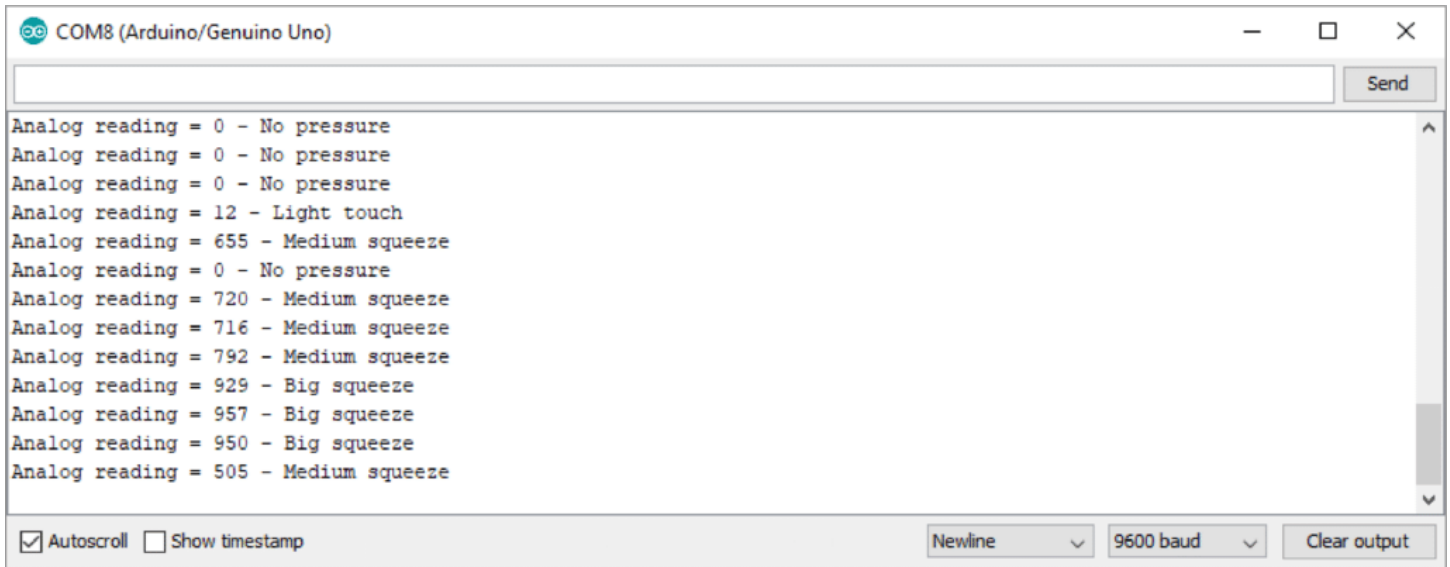
This sketch will read out the sensor data from the analog input of the Arduino and display the output in the serial monitor.

As mentioned earlier, the output voltage of the sensor will be between 0 V (no pressure applied) and roughly 5 V (maximum pressure applied). The Arduino boards contain a multichannel, 10-bit analog to digital converter. This means that it will map the input

voltage between 0 and 5 V into integer values between 0 and 1023. So you should see a value between 0 and 1023 in the serial monitor, depending on how hard you squeeze the sensor.

```
1.  /* Simple example code for Force Sensitive Resistor (FSR) with Arduino. More info:
    https://www.makerguides.com */
2.
3.  // Define FSR pin:
4.  #define fsrpin A0
5.
6.  //Define variable to store sensor readings:
7.  int fsrreading; //Variable to store FSR value
8.
9.  void setup() {
10.     // Begin serial communication at a baud rate of 9600:
11.     Serial.begin(9600);
12. }
13.
14. void loop() {
15.     // Read the FSR pin and store the output as fsrreading:
16.     fsrreading = analogRead(fsrpin);
17.
18.     // Print the fsrreading in the serial monitor:
19.     // Print the string "Analog reading = ".
20.     Serial.print("Analog reading = ");
21.     // Print the fsrreading:
22.     Serial.print(fsrreading);
23.
24.     // We can set some thresholds to display how much pressure is roughly applied:
25.     if (fsrreading < 10) {
26.         Serial.println(" - No pressure");
27.     } else if (fsrreading < 200) {
28.         Serial.println(" - Light touch");
29.     } else if (fsrreading < 500) {
30.         Serial.println(" - Light squeeze");
31.     } else if (fsrreading < 800) {
32.         Serial.println(" - Medium squeeze");
33.     } else {
34.         Serial.println(" - Big squeeze");
35.     }
36.
37.     delay(500); //Delay 500 ms.
38. }
```

You should see the following output in the serial monitor:

The image shows a screenshot of the Serial Monitor window in the Arduino IDE. The window title is "COM8 (Arduino/Genuino Uno)". It features a text input field at the top with a "Send" button. The main area displays the following output:

```
Analog reading = 0 - No pressure
Analog reading = 0 - No pressure
Analog reading = 0 - No pressure
Analog reading = 12 - Light touch
Analog reading = 655 - Medium squeeze
Analog reading = 0 - No pressure
Analog reading = 720 - Medium squeeze
Analog reading = 716 - Medium squeeze
Analog reading = 792 - Medium squeeze
Analog reading = 929 - Big squeeze
Analog reading = 957 - Big squeeze
Analog reading = 950 - Big squeeze
Analog reading = 505 - Medium squeeze
```

At the bottom, there are checkboxes for "Autoscroll" (checked) and "Show timestamp" (unchecked). To the right, there are dropdown menus for "Newline" and "9600 baud", and a "Clear output" button.

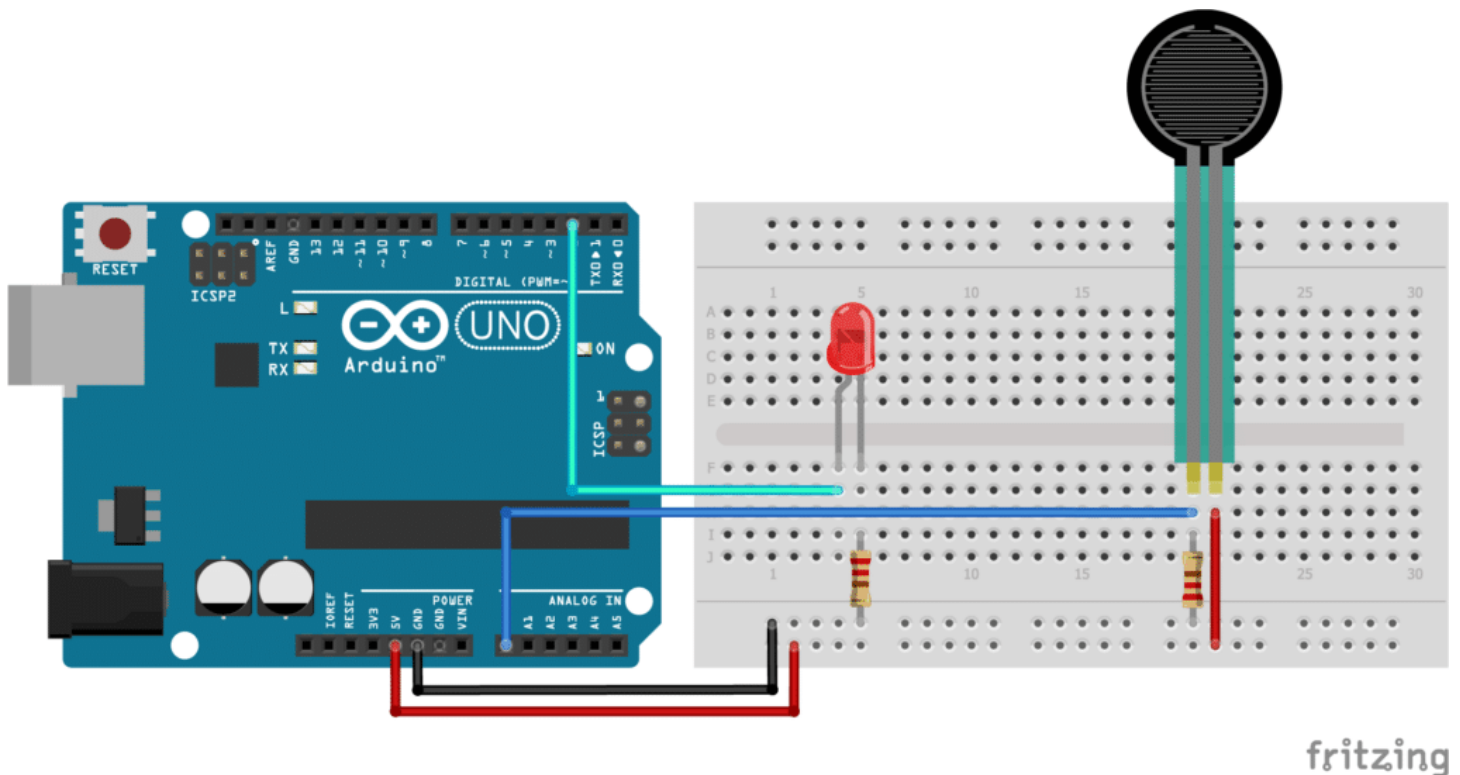
Serial monitor output

Make sure the serial monitor is also set to a baudrate of 9600.

2. Using a Force Sensing Resistor (FSR) as a toggle switch

In this example you will be using the FSR sensor as a toggle switch. You can use this program to control all kinds of other functions, in this case to switch on and off an LED.

You will have to add a LED with a resistor to the circuit, which is shown in the diagram below.



FSR with Arduino and LED wiring diagram

The negative lead of the LED (the short lead) gets connected to GND via a resistor and the positive lead to digital pin 2. The value of the resistor depends on the color LED you are using. You can use the following values as a guide:

- Blue, Green, White or UV: 68 Ω
- Red, Yellow or Yellow-Green: 150 Ω

If you don't have any of these resistor values, try to find one that is close. You can also put multiple resistors in series, to get the correct value.

The sketch below will toggle the LED on and off when you press on the FSR. It looks at the value of the analog input pin and changes the state of the LED when the value exceeds 500. This means that a really light press won't be detected.

This example also debounces the input and is based on the Arduino Switch (<https://www.arduino.cc/en/tutorial/switch>) tutorial.

```
1.  /* Example code to use Force Sensitive Resistor (FSR) as toggle switch to control LED.
    More info: https://www.makerguides.com */
2.
3.  // Define pins:
4.  #define fsrpin A0
5.  #define ledpin 2
6.
7.  // Define variables:
8.  int fsrreading; // The current reading from the FSR
9.  int state = HIGH; // The current state of the output pin
10. int previous = 0; // The previous reading from the FSR
11. // The follow variables are long's because the time, measured in milliseconds, will quickly
    become a bigger number than can be stored in an int:
12. long time = 0; // The last time the output pin was toggled
13. long debounce = 40; // The debounce time, increase if the output flickers
14.
15. void setup() {
16.     // Begin serial communication at a baud rate of 9600:
17.     Serial.begin(9600);
18.     // Set ledpin as output:
19.     pinMode(ledpin, OUTPUT);
20. }
21.
22. void loop() {
23.     // Read the FSR pin and store the output as fsrreading:
24.     fsrreading = analogRead(fsrpin);
25.
26.     // Print the fsrreading in the serial monitor:
27.     Serial.println(fsrreading);
28.
29.     // If the input just went from below 500 to above 500 and we've waited long enough to
    ignore any noise on the circuit, toggle the output pin and remember the time:
30.     if (fsrreading > 500 && previous < 500 && millis() - time > debounce) {
31.         if (state == HIGH)
32.             state = LOW;
33.         else
34.             state = HIGH;
35.         time = millis();
36.     }
37.
```

```

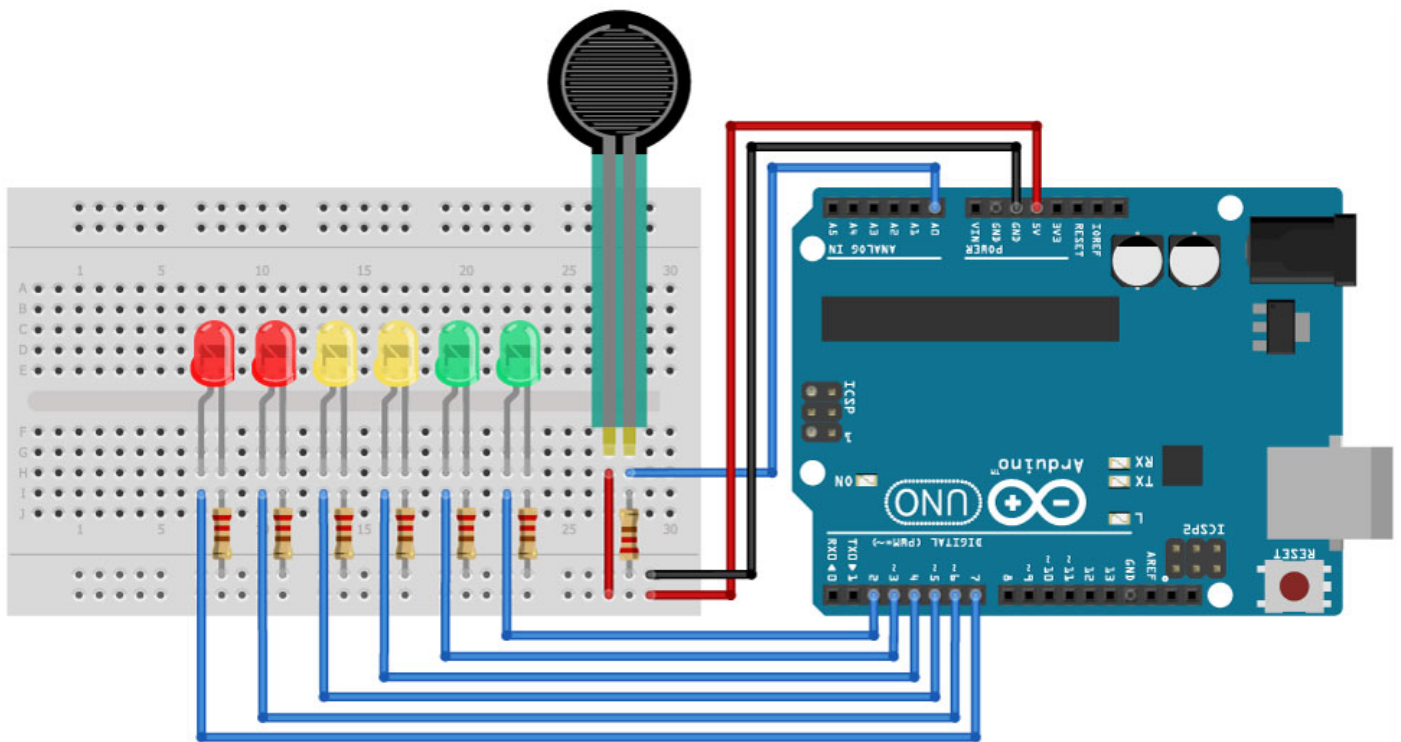
38.     digitalWrite(ledpin, state);
39.
40.     previous = fsrreading;
41. }

```

3. Control multiple LEDs with an FSR as pressure sensor

The example below makes it easy to see how much pressure you apply to the FSR. The more pressure you apply, the more LEDs will turn on.

You can wire up the LEDs in the same way as before, see the wiring diagram below. The LEDs are connected to digital pin 2 to 7. The FSR is also connected in the same way as before.



fritzing

FSR with multiple LEDs and Arduino wiring diagram

Because the output voltage of the FSR is non-linear I set up a custom range for each LED to turn on. You might need to tweak this slightly for your own sensor.

```
1.  /* Arduino example code to control multiple LEDs with a Force Sensitive Resistor (FSR) as
2.  pressure sensor. More info: https://www.makerguides.com */
3.
4.  // Define pins:
5.  #define fsrpin A0
6.  #define led1 2
7.  #define led2 3
8.  #define led3 4
9.  #define led4 5
10. #define led5 6
11. #define led6 7
12.
13. // Define variables:
14. int fsrreading;
15.
16. void setup() {
17.     // Begin serial communication at a baud rate of 9600:
18.     Serial.begin(9600);
19.     // Set LED pins as output:
20.     pinMode(led1, OUTPUT);
21.     pinMode(led2, OUTPUT);
22.     pinMode(led3, OUTPUT);
23.     pinMode(led4, OUTPUT);
24.     pinMode(led5, OUTPUT);
25.     pinMode(led6, OUTPUT);
26. }
27.
28. void loop() {
29.     // Read the FSR pin and store the output as fsrreading:
30.     fsrreading = analogRead(fsrpin);
31.
32.     // Print the fsrreading in the serial monitor:
33.     Serial.println(fsrreading);
34.
35.     // Control the LEDs:
36.     if (fsrreading > 200) {
37.         digitalWrite(led1, HIGH);
38.     }
39.     else digitalWrite(led1, LOW);
40.     if (fsrreading > 450) {
41.         digitalWrite(led2, HIGH);
42.     }
43.     else digitalWrite(led2, LOW);
44.     if (fsrreading > 550) {
45.         digitalWrite(led3, HIGH);
46.     }
47.     else digitalWrite(led3, LOW);
48.     if (fsrreading > 650) {
49.         digitalWrite(led4, HIGH);
50.     }
51.     else digitalWrite(led4, LOW);
```

```
51.     if (fsrreading > 800) {  
52.         digitalWrite(led5, HIGH);  
53.     }  
54.     else digitalWrite(led5, LOW);  
55.     if (fsrreading > 900) {  
56.         digitalWrite(led6, HIGH);  
57.     }  
58.     else digitalWrite(led6, LOW);  
59. }
```

CAD files

Below you can find all the CAD files for sensors of the interlink 400 Series.

[FSR400.zip](#) 

[FSR402.zip](#) 

[FSR404.zip](#) 

[FSR406.zip](#) 

[FSR408.zip](#) 

[FSR402Short.zip](#) 

[FSR400Short.zip](#) 

Conclusion

In this article, I have shown you how an FSR works and how you can use it with Arduino. I hope you found it useful and informative. If you did, please share it with a friend that also likes electronics!

I would love to know what projects you plan on building (or have already built) with an FSR. If you have any questions, suggestions, or if you think that things are missing in this tutorial, **please leave a comment down below.**

Note that comments are held for moderation to prevent spam.

Beginner



What to read next?

LM35 analog temperature sensor with Arduino tutorial
(<https://www.makerguides.com/lm35-arduino-tutorial/>)

TMP36 analog temperature sensor with Arduino tutorial
(<https://www.makerguides.com/tmp36-arduino-tutorial/>)

Arduino Nano Board Guide (Pinout, Specifications, Comparison)
(<https://www.makerguides.com/arduino-nano/>)

The complete guide for DS18B20 digital temperature sensors with Arduino
(<https://www.makerguides.com/ds18b20-arduino-tutorial/>)

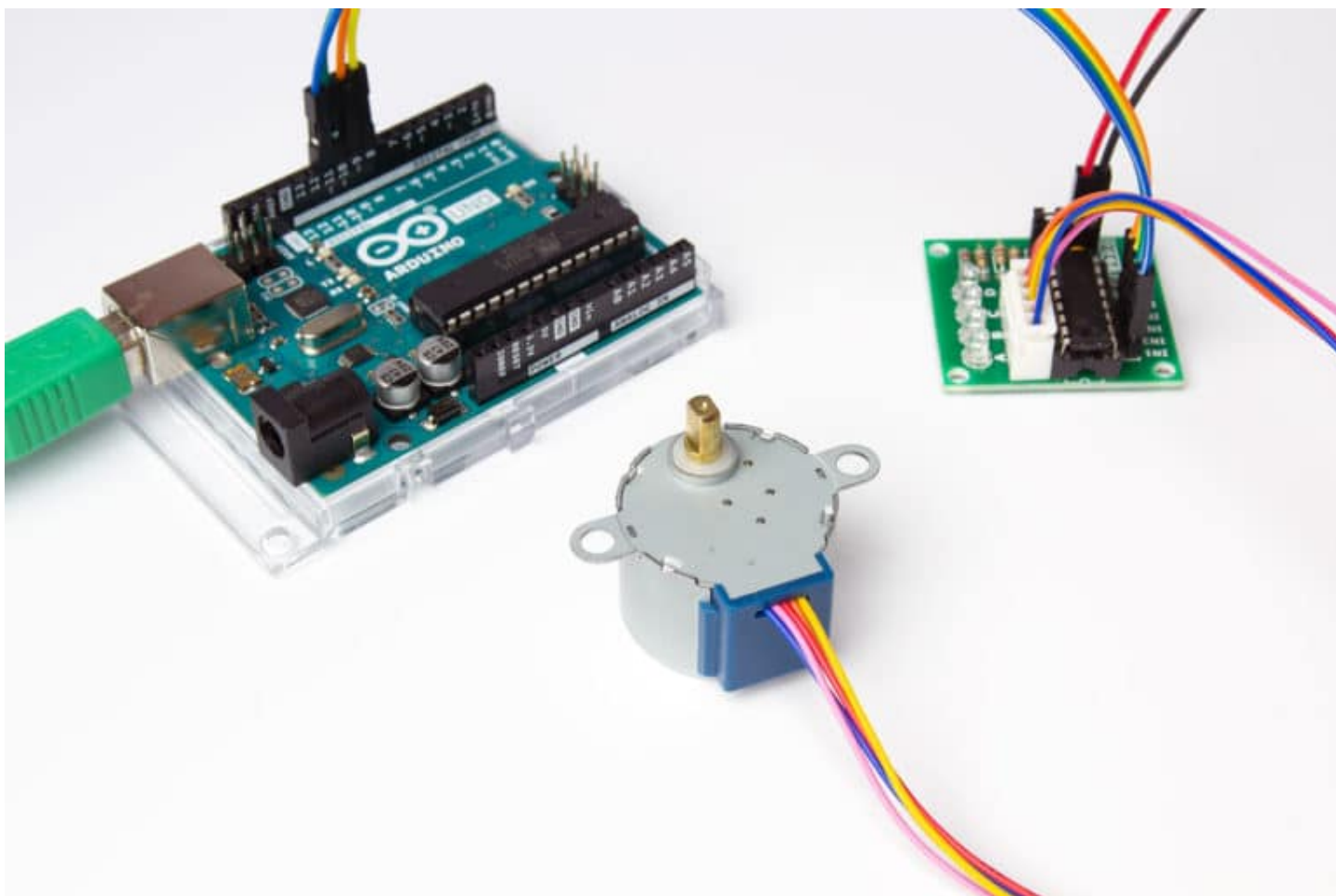
How to use an IR receiver and remote with Arduino
(<https://www.makerguides.com/ir-receiver-remote-arduino-tutorial/>)

Trackbacks

Project #2: Pressure Sensor Switch Cat Helmet – Art 150: Intro to New Media
(<https://intronma.wordpress.com/2019/12/14/project-2-pressure-sensor-switch-cat-helmet/>) says:

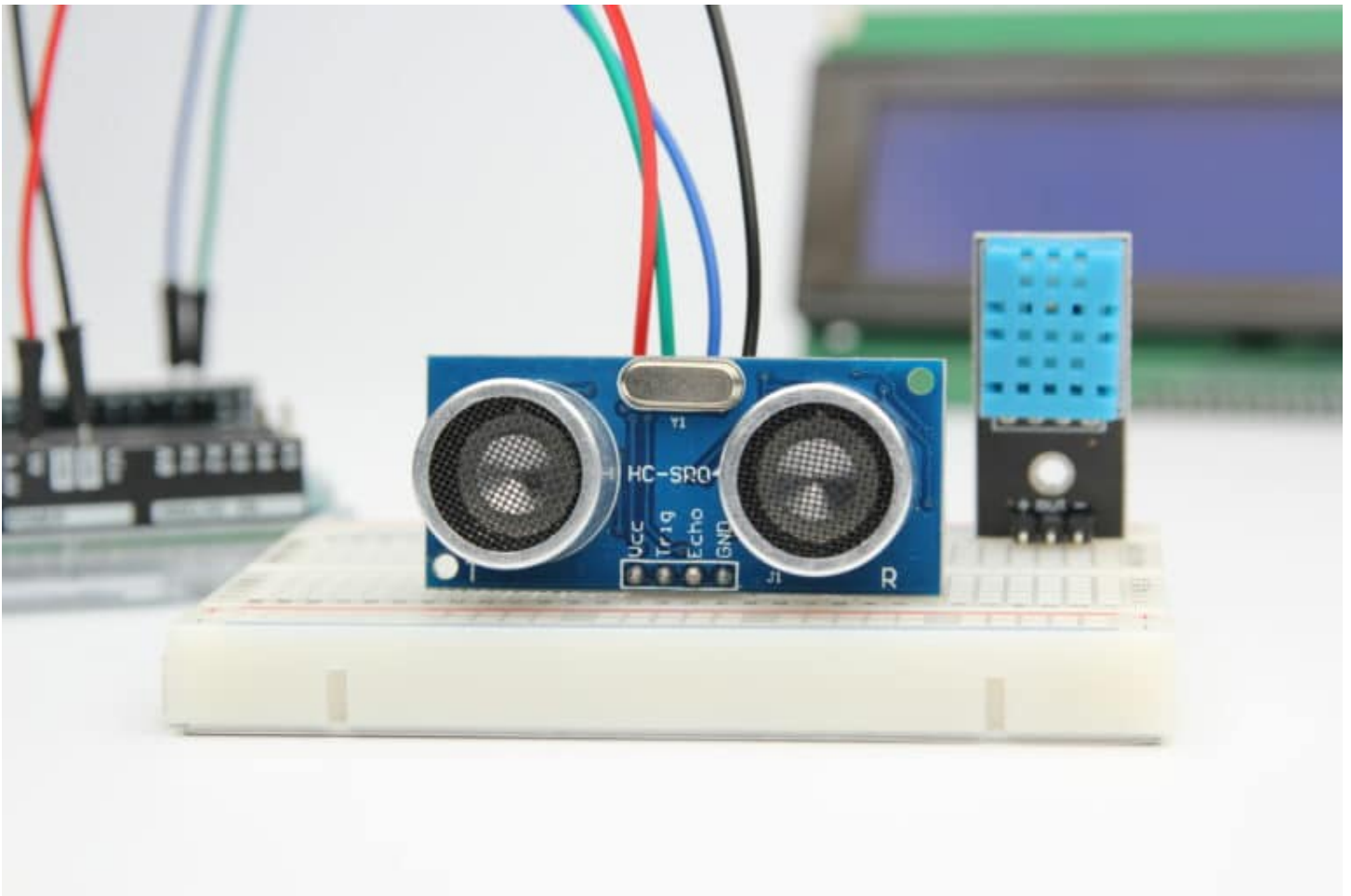
December 14, 2019 at 5:17 am (<https://www.makerguides.com/fsr-arduino-tutorial/#comment-1329>)

[...] Link to code source: <https://www.makerguides.com/fsr-arduino-tutorial/>
(<https://www.makerguides.com/fsr-arduino-tutorial/>) [...]



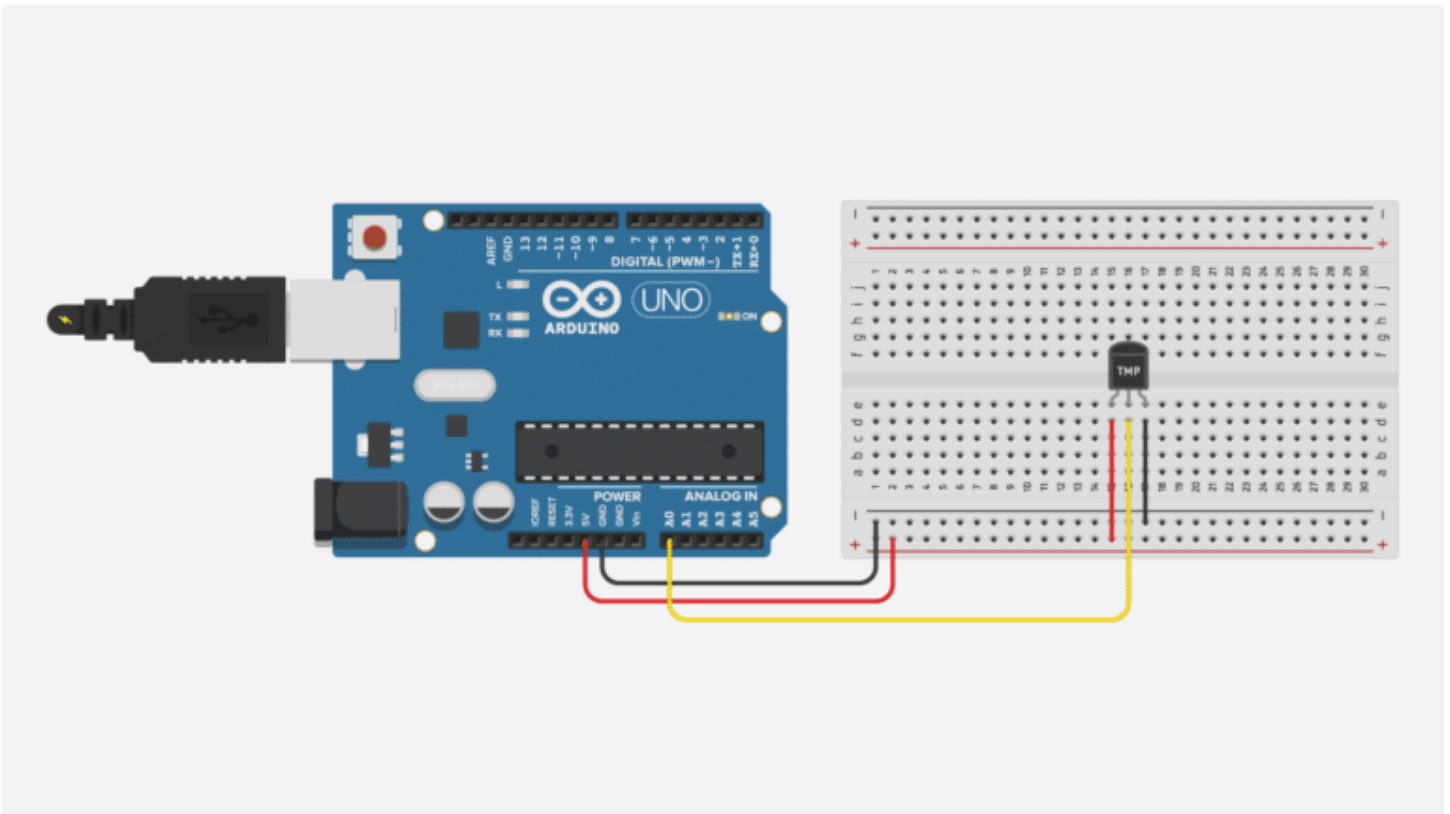
(<https://www.makerguides.com/28byj-48-stepper-motor-arduino-tutorial/>)

28BYJ-48 Stepper Motor with ULN2003 Driver and Arduino Tutorial
(<https://www.makerguides.com/28byj-48-stepper-motor-arduino-tutorial/>)




(<https://www.makerguides.com/hc-sr04-arduino-tutorial/>)

How to use an HC-SR04 Ultrasonic Distance Sensor with Arduino
(<https://www.makerguides.com/hc-sr04-arduino-tutorial/>)



(<https://www.makerguides.com/tmp36-arduino-tutorial/>)

TMP36 analog temperature sensor with Arduino tutorial (<https://www.makerguides.com/tmp36-arduino-tutorial/>)

 Ezoic (<https://www.ezoic.com/what-is-ezoic/>)

report this ad

© 2021 Makerguides.com - All Rights Reserved