Introduction to Arduino

with Adafruit in use-Part A

What is Arduino?

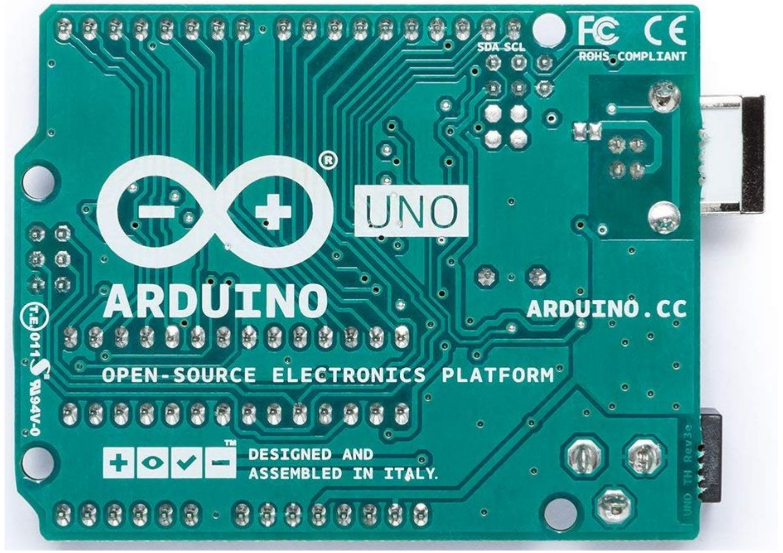
ARDUINO™

An open-source hardware & software company!

...but more commonly refers to....

A development platform!

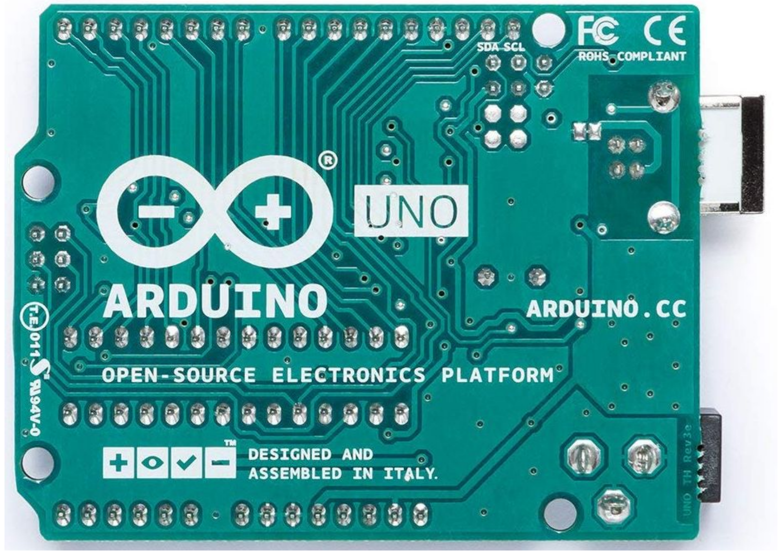
Why Arduino?


UNO
ARDUINO
ARDUINO.CC
OPEN-SOURCE ELECTRONICS PLATFORM
DESIGNED AND ASSEMBLED IN ITALY.
ROHS COMPLIANT
SDA SCL

- Easy to use
- Structure similar to C
- Ability to utilize both digital and analog signals
- Supports various components

  (e.g. vibrators, pressure sensors…)

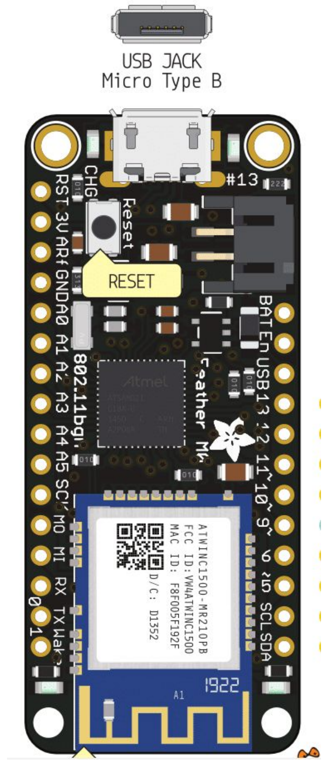
Why “not” Arduino?


UNO
ARDUINO
ARDUINO.CC
OPEN-SOURCE ELECTRONICS PLATFORM
DESIGNED AND ASSEMBLED IN ITALY.
ROHS COMPLIANT
SDA SCL

...can’t really think of, literally.

The cost buying one, maybe?

What are we going to do today?

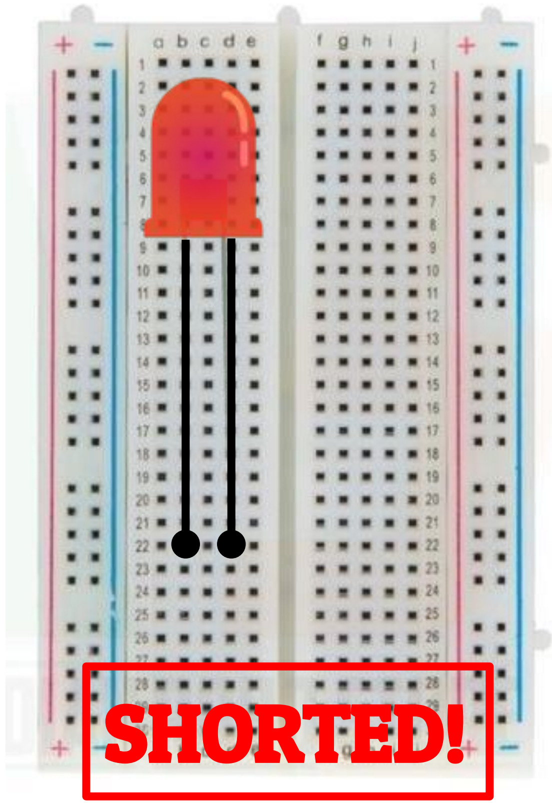


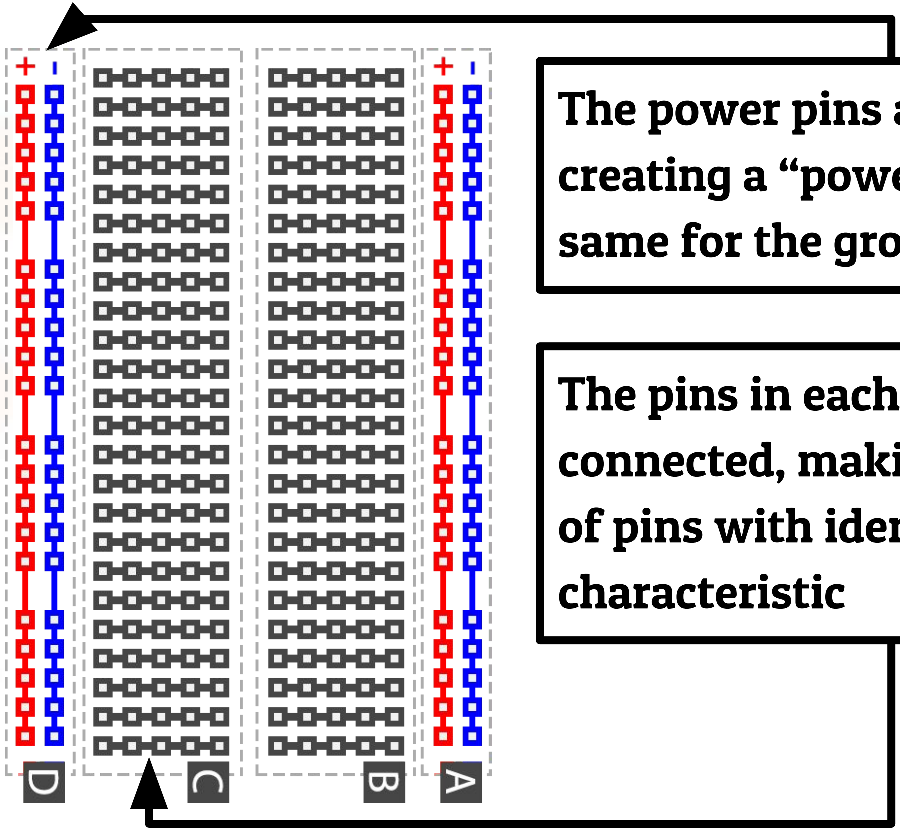
+

+

Before we start...some basic knowledge first!

Basic knowledge 01. Breadboard

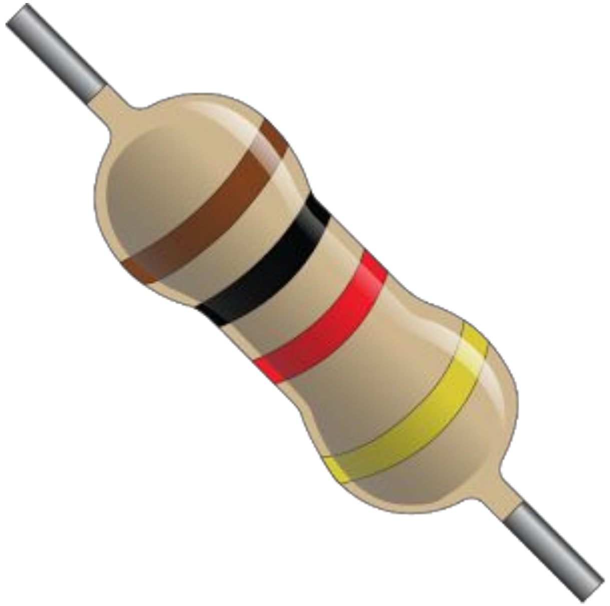




The power pins are all connected, creating a “power row”. It’s the same for the ground pins

The pins in each row are all connected, making each row a set of pins with identical characteristic

Basic knowledge 02. Resistors



What are these for?

- **Limiting the current in the circuit**

  ...don't burn your electrical components!

- **Reduce noise signals**

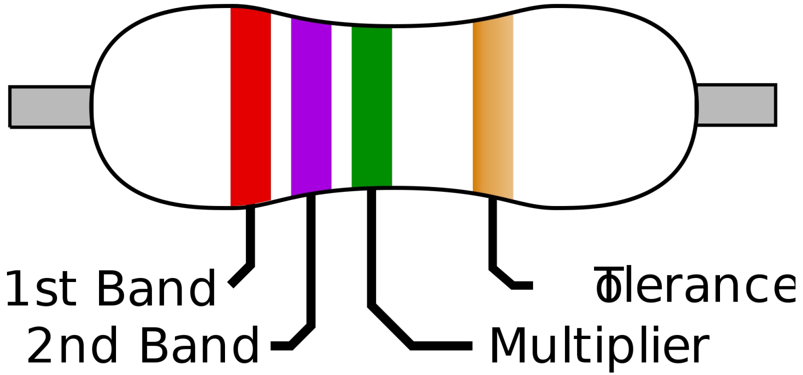
  extremely important if you are using ground as 0!

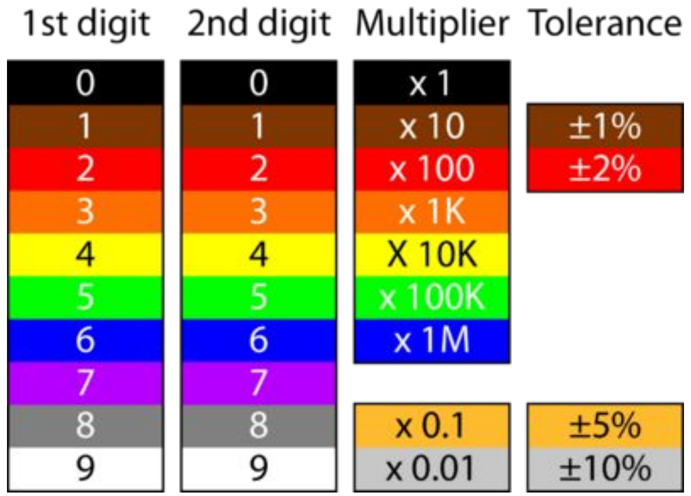
- **Prevents shorts**

Basic knowledge 02. Resistors

How do I identify the resistance value?

Read the colorful rings on it!





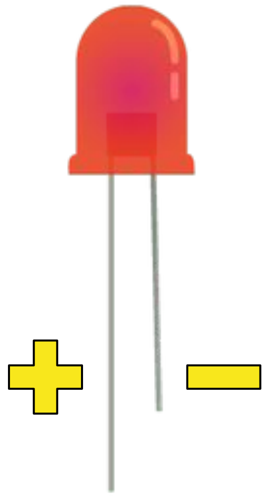
1st digit	2nd digit	Multiplier	Tolerance
0	0	x 1	
1	1	x 10	±1%
2	2	x 100	±2%
3	3	x 1K	
4	4	X 10K	
5	5	x 100K	
6	6	x 1M	
7	7		
8	8	x 0.1	±5%
9	9	x 0.01	±10%

Resistance value = $27*10^5$ = 2.7M, tolerance = 5%

Basic knowledge 03. LED

How do I know which end is positive?

By the length of the pin!

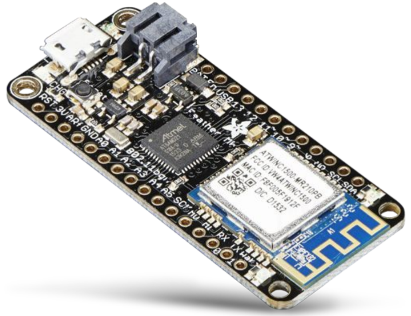


Longer pin	**->**	**Positive end**
Shorter pin	**->**	**Negative end**

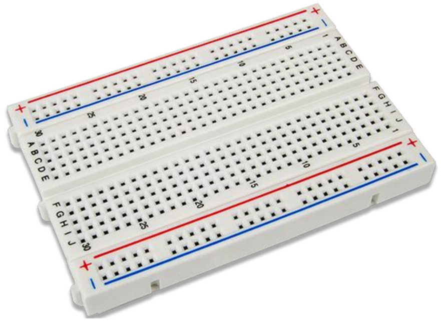
Some tools you might use

Now, send a person to the front and get your gadgets

Let’s check if you have everything



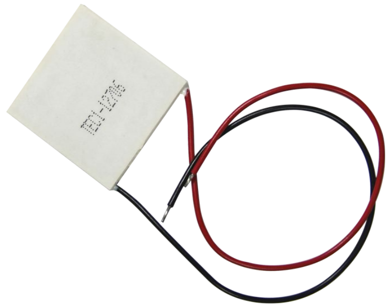
Adafruit *1



Breadboard *1

Servo motor *1

LED *1

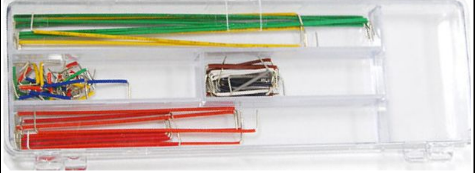


Peltier module *1

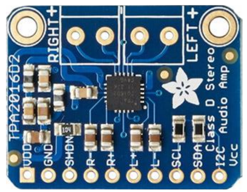
Vibrator *1

Button *1

Resistor *5



Box wires *1



Amplifier *1

Micro USB cable *1

Ba

Component list:

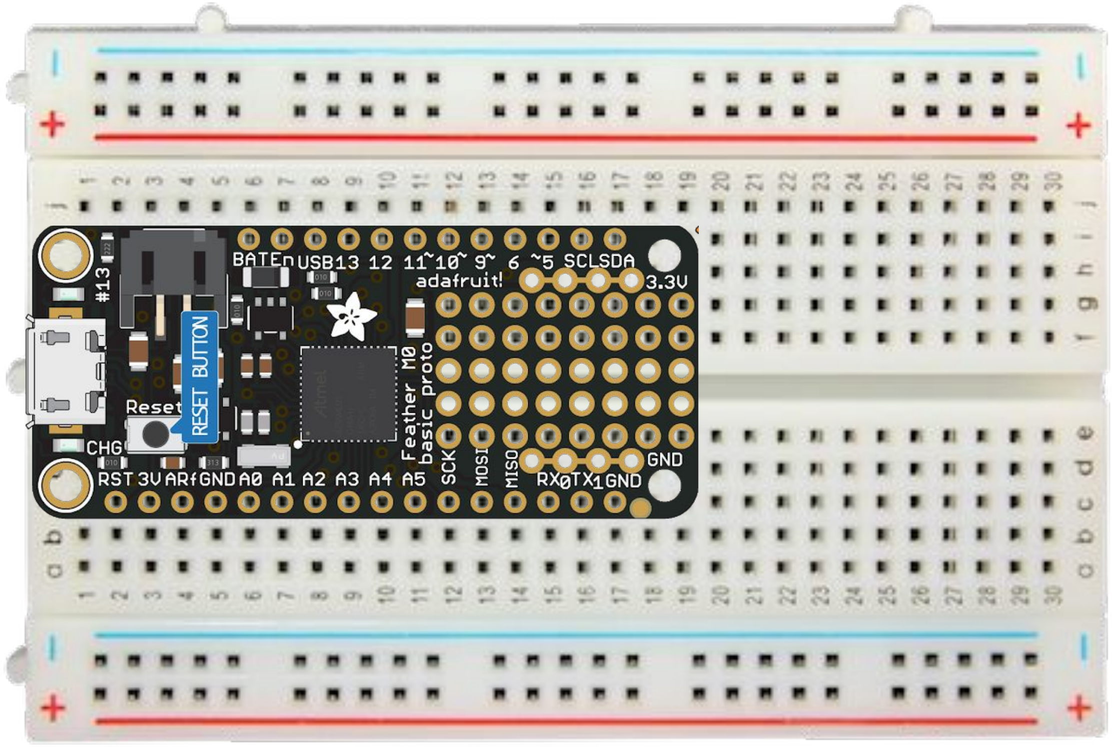
● Adafruit Feather M0 Wifi	*1
● Breadboard	*1
● Servo motor	*1
● LED	*1
~~● Peltier module(thermal module)~~	~~*1~~
● Vibrator	*1
● Button	*1
● Resistor(1.1k)	*5
● Box of wires	*1
● Adafruit TPA2016D2 Amplifier	*1
● Micro USB cable	*1
● Battery	*1

Make sure everything comes back the same in the end of the semester

...except resistors. They’re quite cheap actually

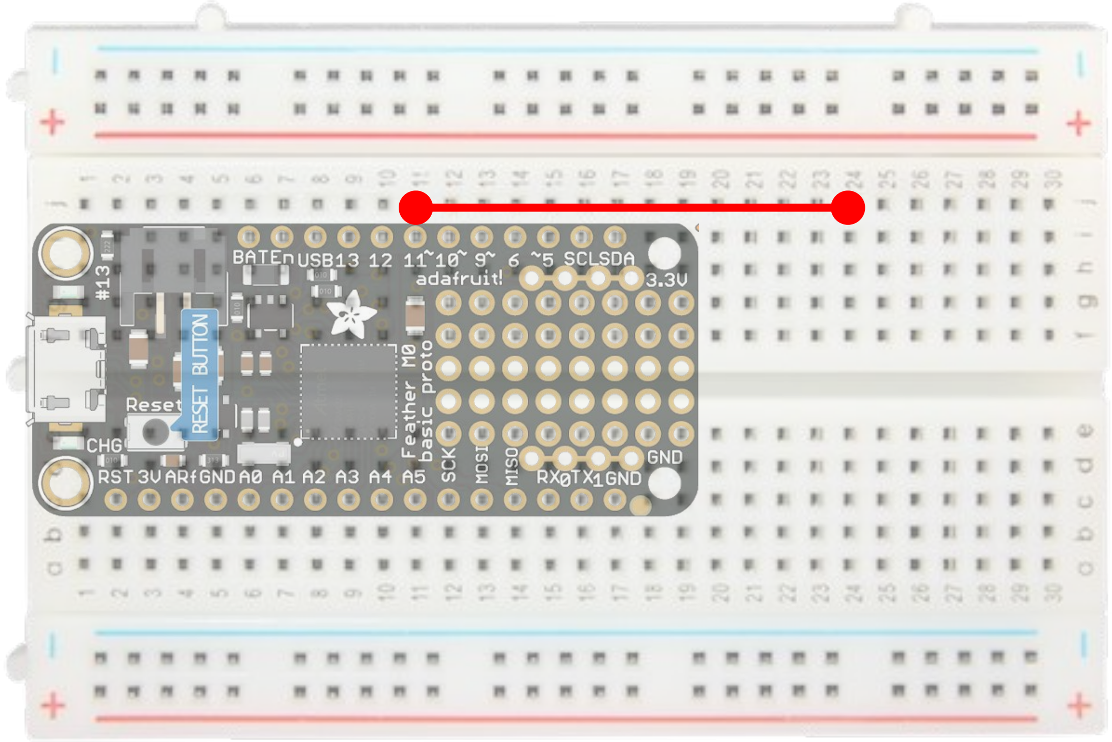
Let’s begin with LED!

Wiring: step 1



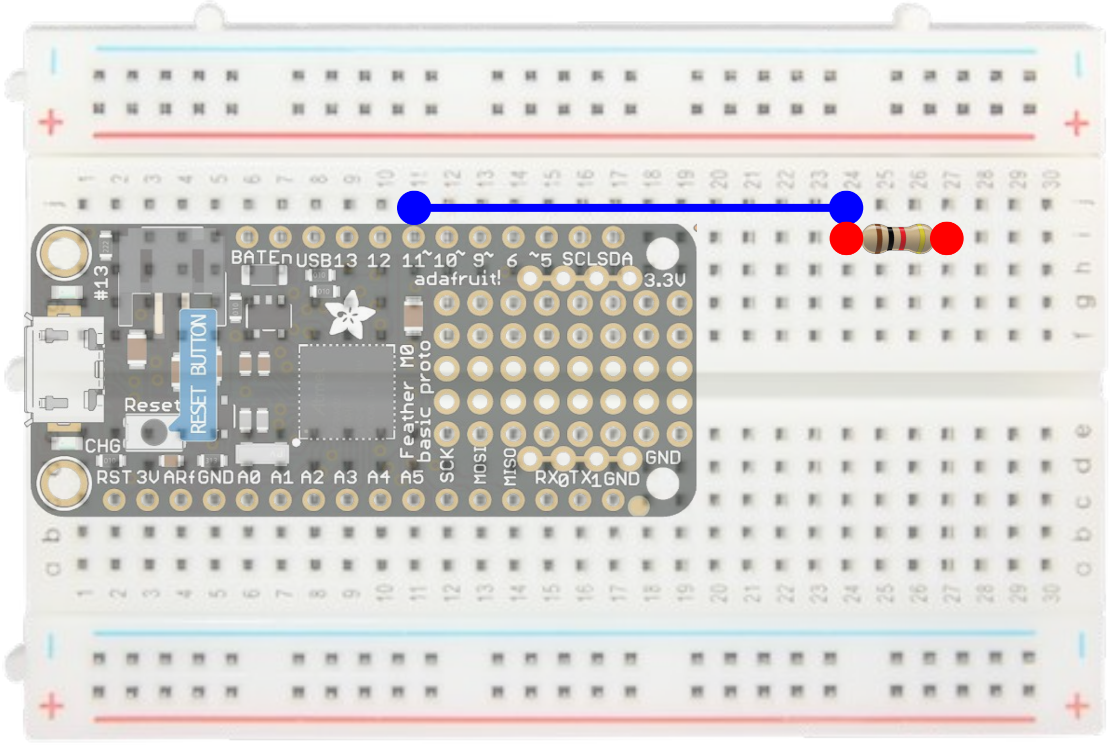
Plug Adafruit onto the breadboard, with pin 11 of Adafruit matching row 11 of the breadboard

Wiring: step 2



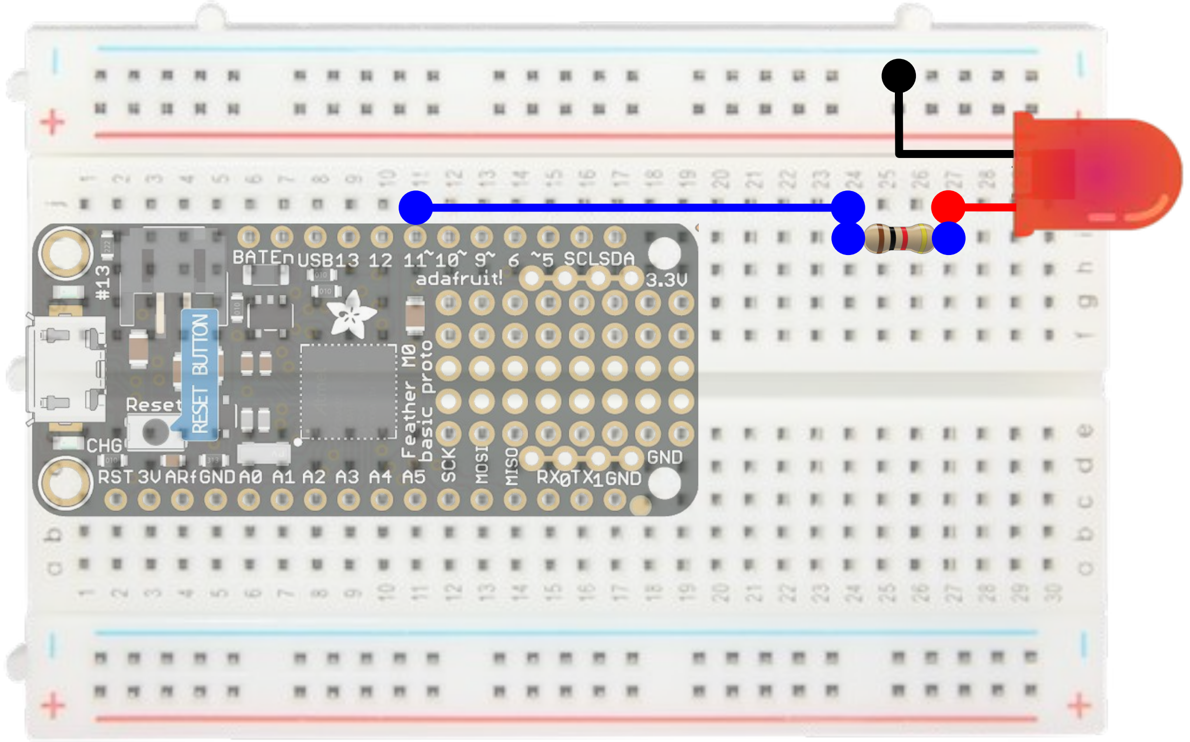
Connect row 11 and row 24 with a wire

Wiring: step 3



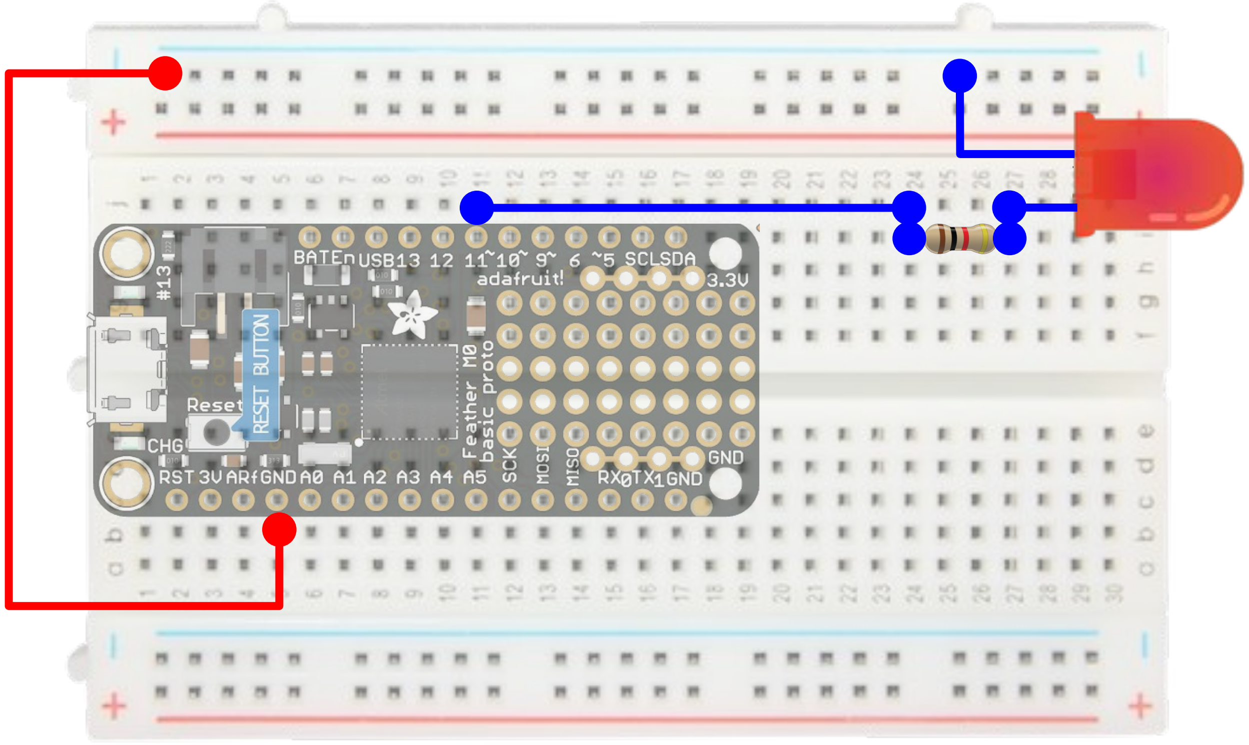
Add a resistor which connects row 24 and row 27

Wiring: step 4



Add LED, with positive end plugged into row 27, and negative end plugged into the ground(negative) column

Wiring: step 5



Connect the GND pin of Adafruit with the breadboard's ground column

All done? Let’s move on to coding!

Coding

`void setup() {`	The setup function is executed once every time the system is powered
`  pinMode(11, OUTPUT);`	This line sets the mode of pin 11 to output
`}`	
`void loop() {`	Loop function will be executed recursively
`  digitalWrite(11, HIGH);`	This line sets the output of pin 11 to HIGH, which is 3.3V on Adafruit
`  delay(50);`	This line delays(pauses) the system by 50ms
`  digitalWrite(11, LOW);`	This line sets the output of pin 11 to LOW, which is 0V
`  delay(50);`	
`}`	

Coding

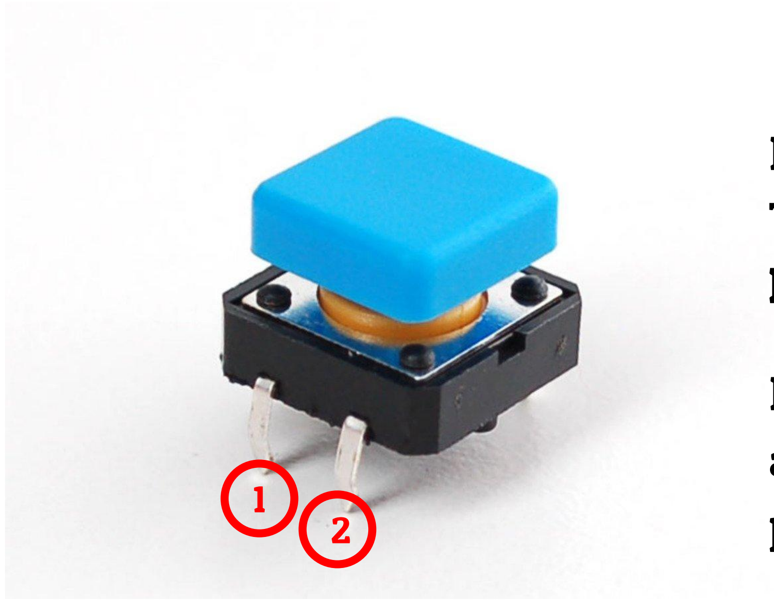
After pasting the code in previous page, hit "Upload" and see what happens!

If you want to change the flashing frequency, just adjust the length of pause!

Now, let’s add a button to control it!

Wiring: step 0

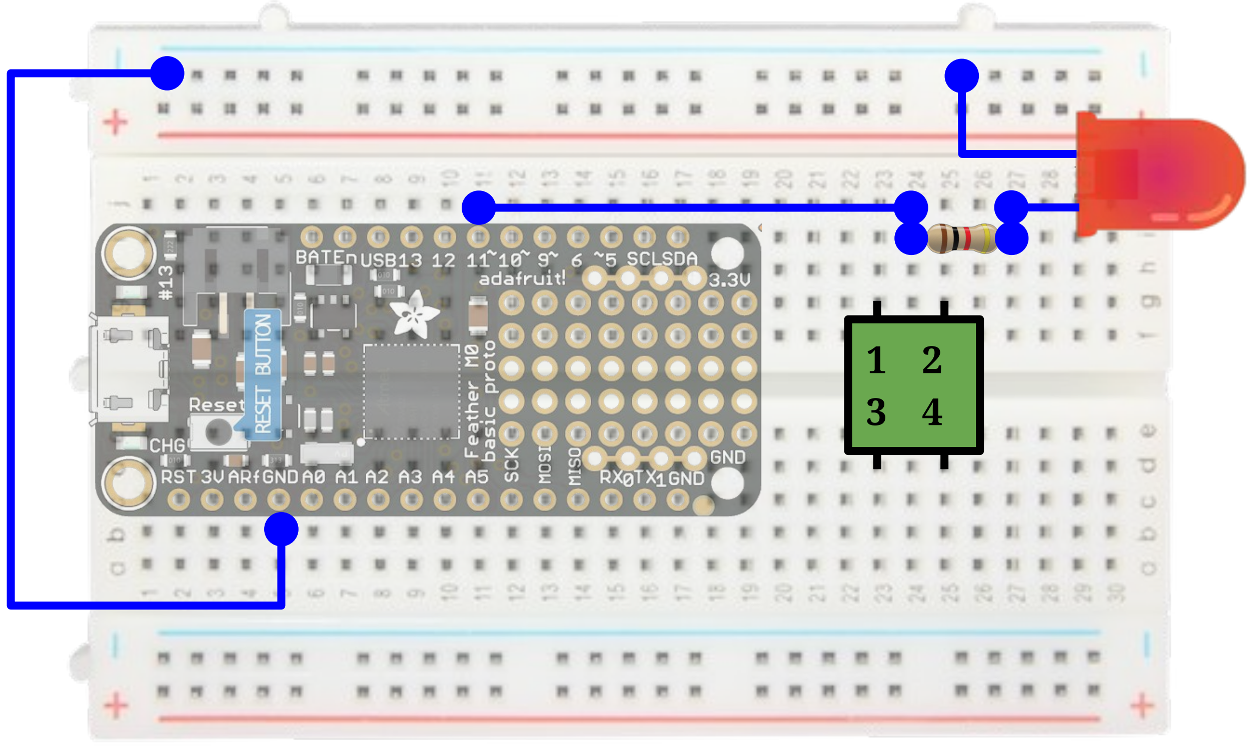
Let’s start with introduction to the button we provided



Pin 1 and pin 2 shown are **NOT CONNECTED.** They will be connected **ONLY WHEN THE BUTTON IS PRESSED.**

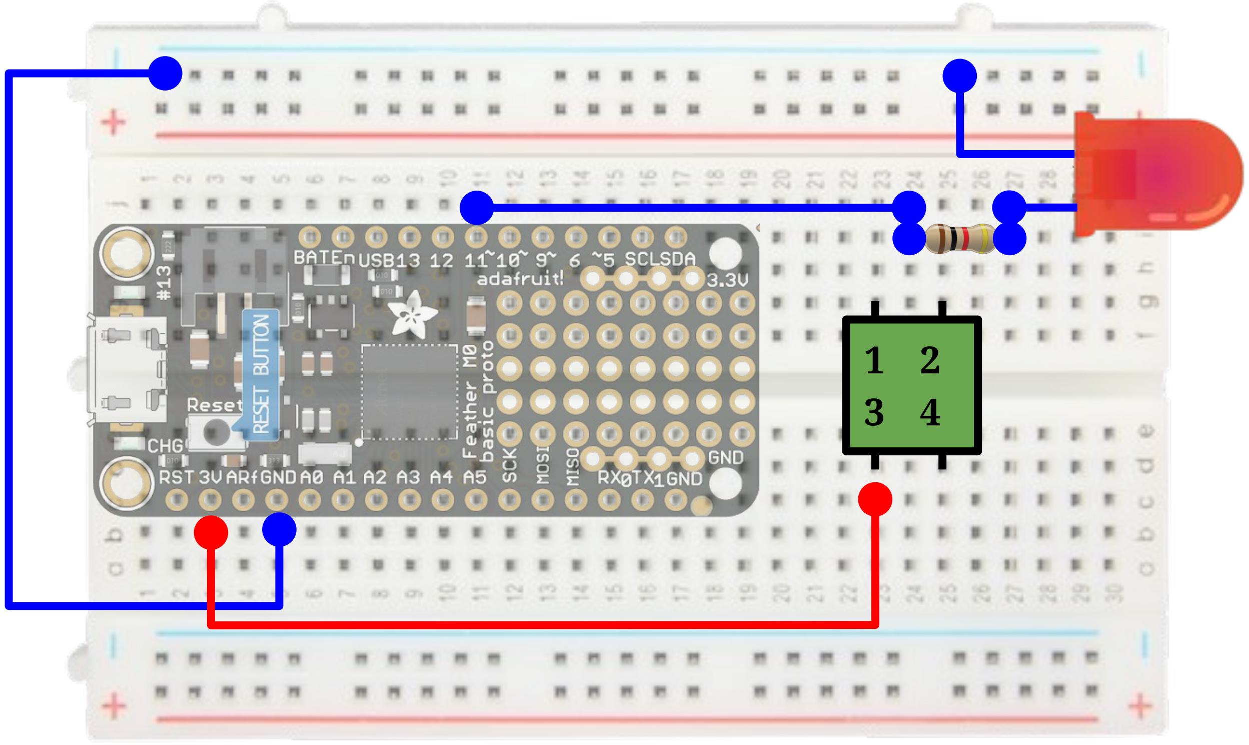
However, pin 1 is connected to the pin right across the other side; i.e. they act as a single pin. It is the same for pin 2

Wiring: step 1



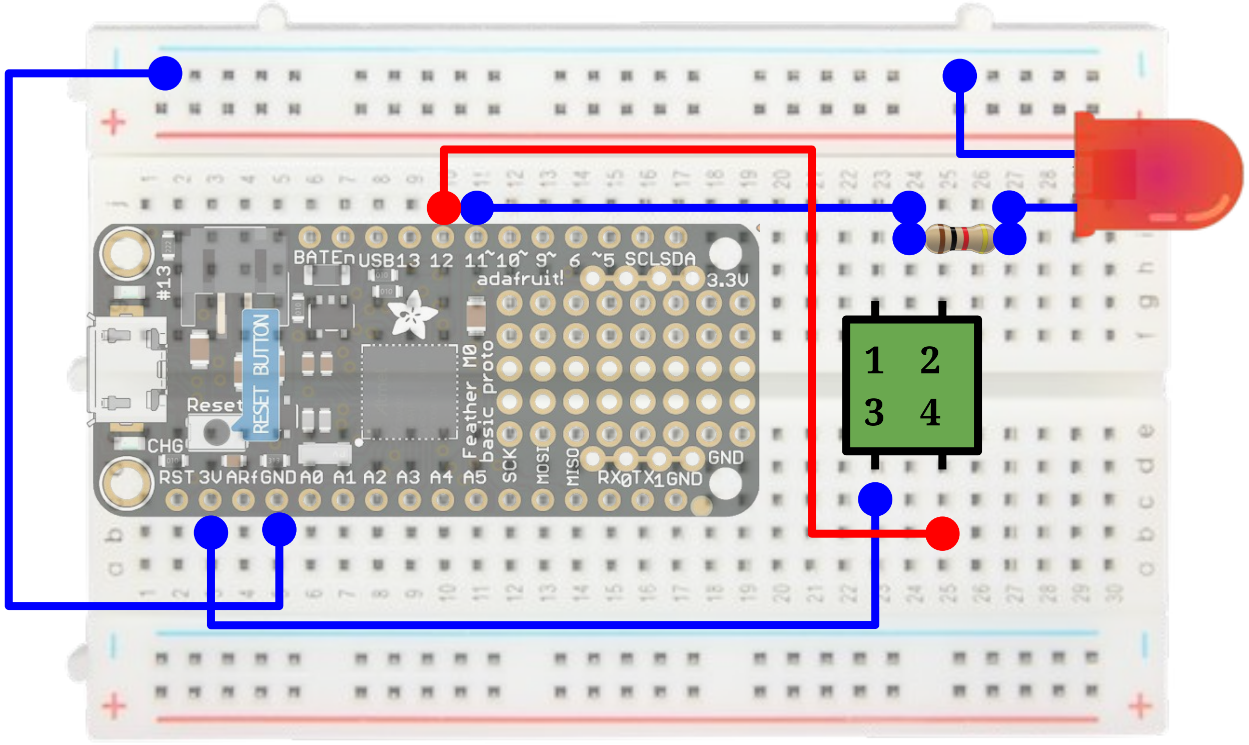
Plug the button onto the breadboard, with 2 of the pins plugged into row 23, and the other 2 into row 25

Wiring: step 2



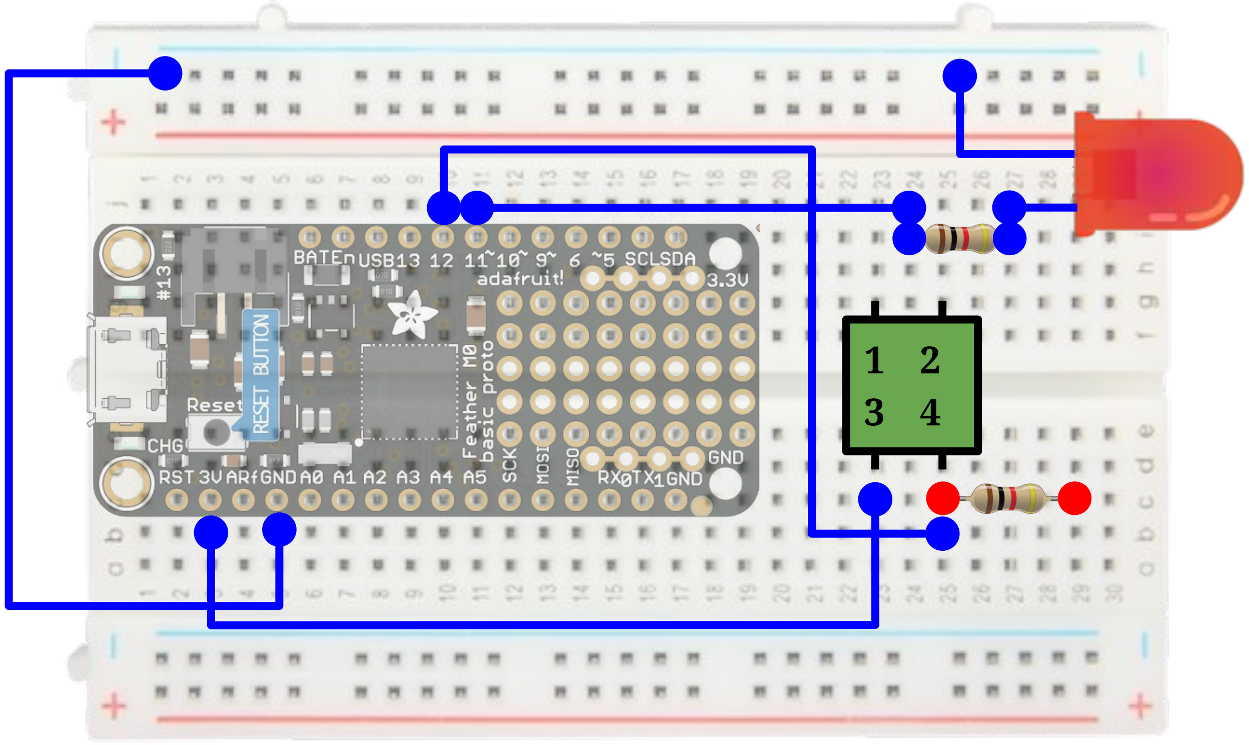
Connect the 3V pin of Adafruit with pin 3 of the button

Wiring: step 3



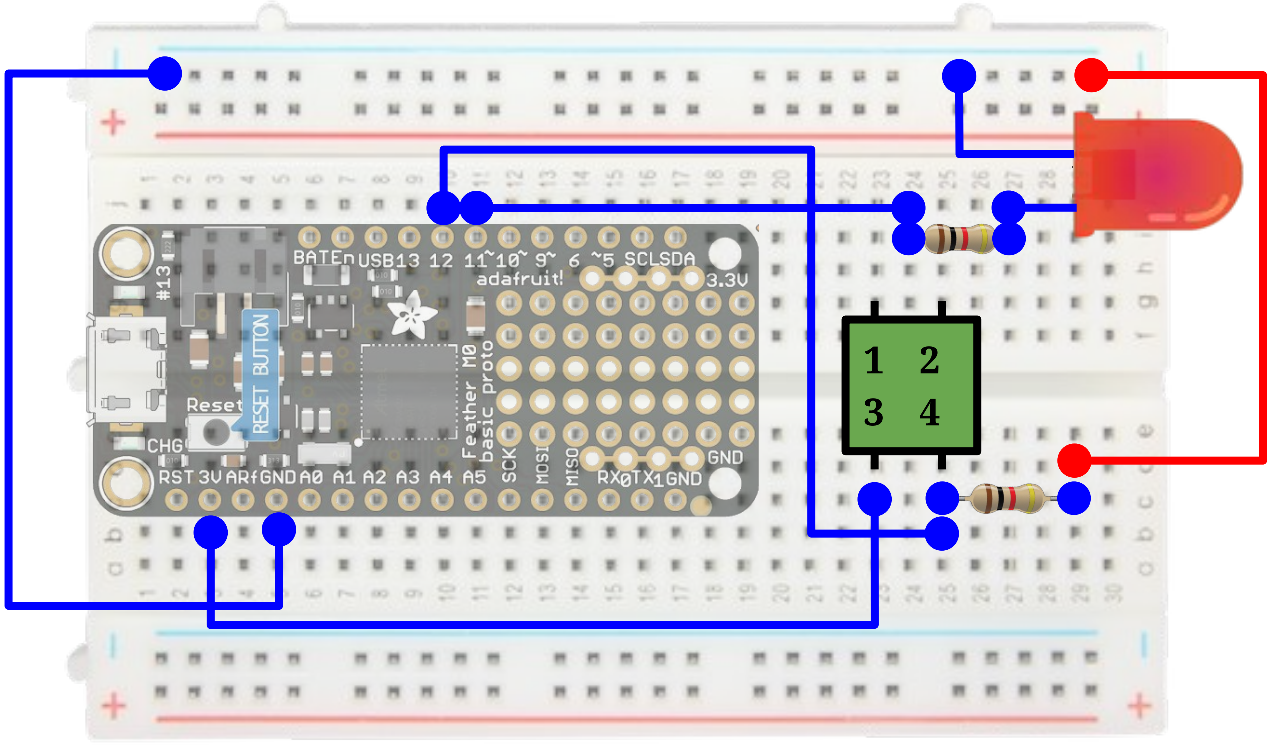
Connect pin 12 of Adafruit with pin 4 of the button

Wiring: step 4



Add a resistor crossing pin 4 of the button(row 25) and row 29

Wiring: step 5



Connect the right end of the resistor with the ground column

It’s time for coding!

Coding: part 1

```
bool button = false;

void setup() {
  pinMode(11, OUTPUT);
  pinMode(12, INPUT);
}

void loop() {
  if(digitalRead(12)){
   button = true;
  }
```

This boolean variable is used to track whether the button is pressed

Sets the pin mode of pin 12 to INPUT so we can read the value

This line reads the digital value on pin 12. It will be either 0 or 1

If 1 is read from pin 12, then it means the button is pressed. Thus, we set the value of this tracking boolean to "true"

Coding: part 2

```
else{
  button = false;
 }

 if(button){
  digitalWrite(11, HIGH);
  delay(50);
  digitalWrite(11, LOW);
  delay(50);
 }
}
```

If the button is released, then reset the button value to “false

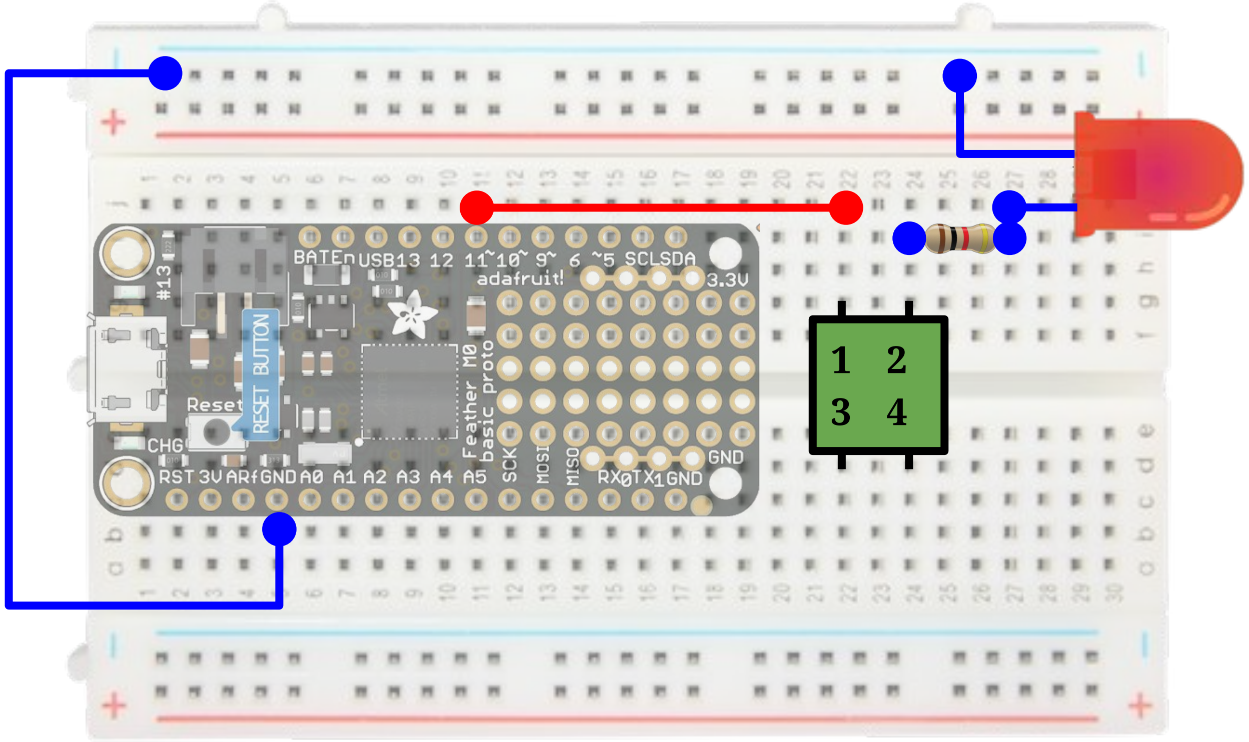
Enable the output to LED only when the button is pressed

Coding

Now, you should be able to control the flashing of LED by the button!

However, there’s a much easier way to achieve the same goal using the same code given in section 1...

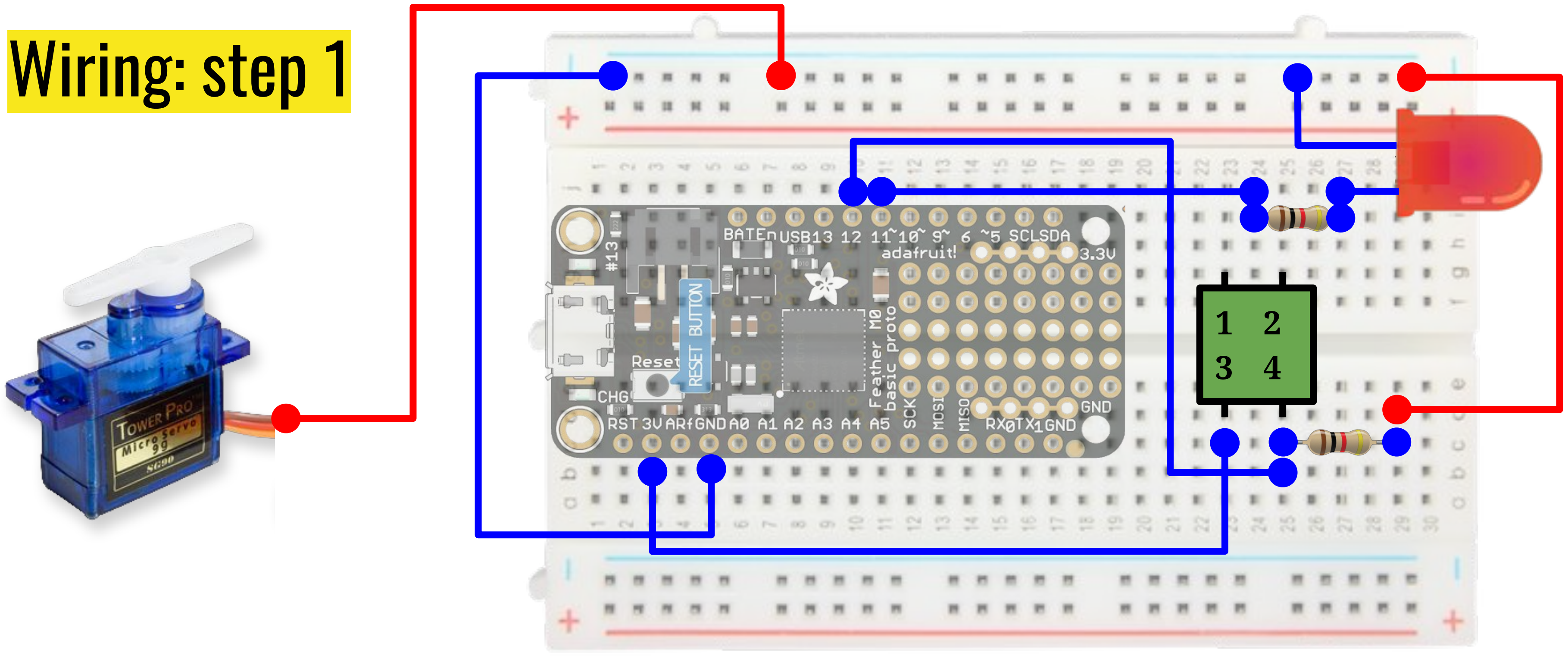
Alternative



If wired up like this, you don't have to upload a new code. The idea of this layout is directly control the output current of pin 11. Here the button acts as a switch

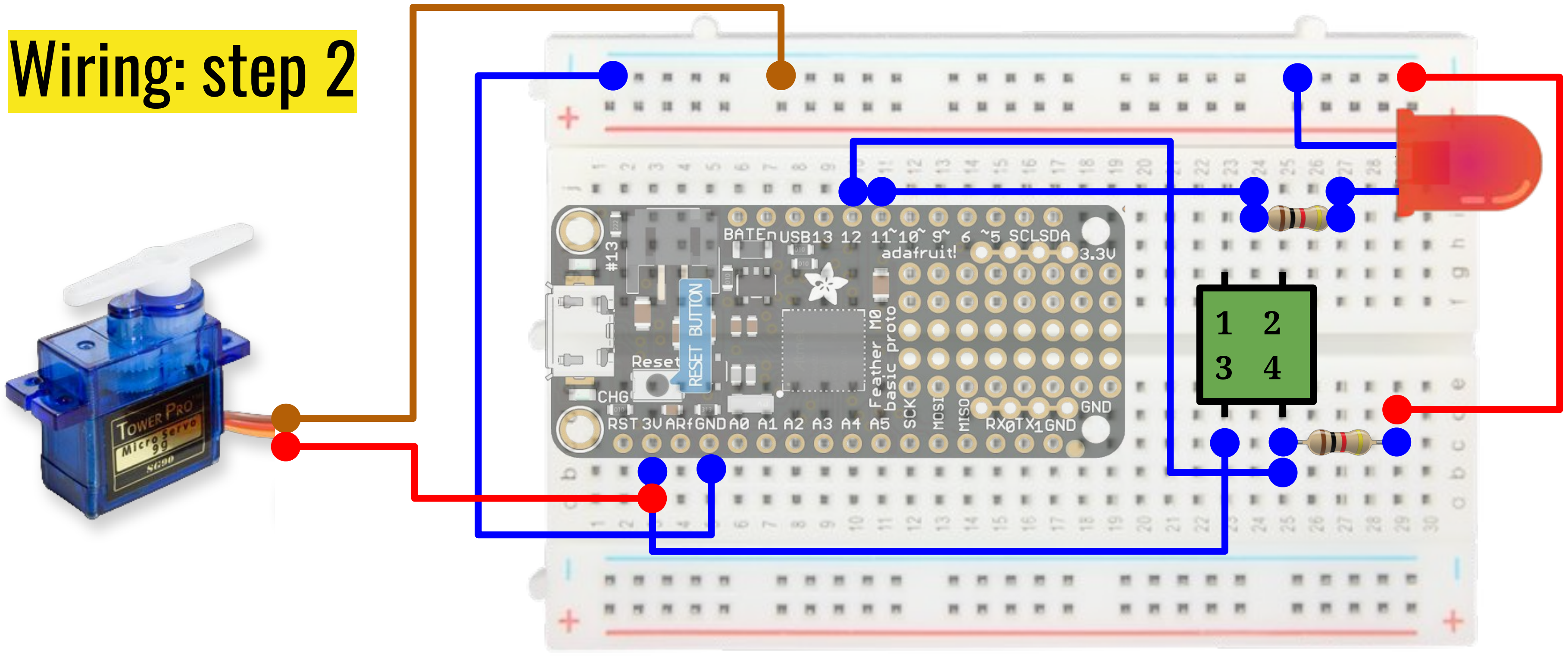
Section 3: Servo motor

Wiring: step 1



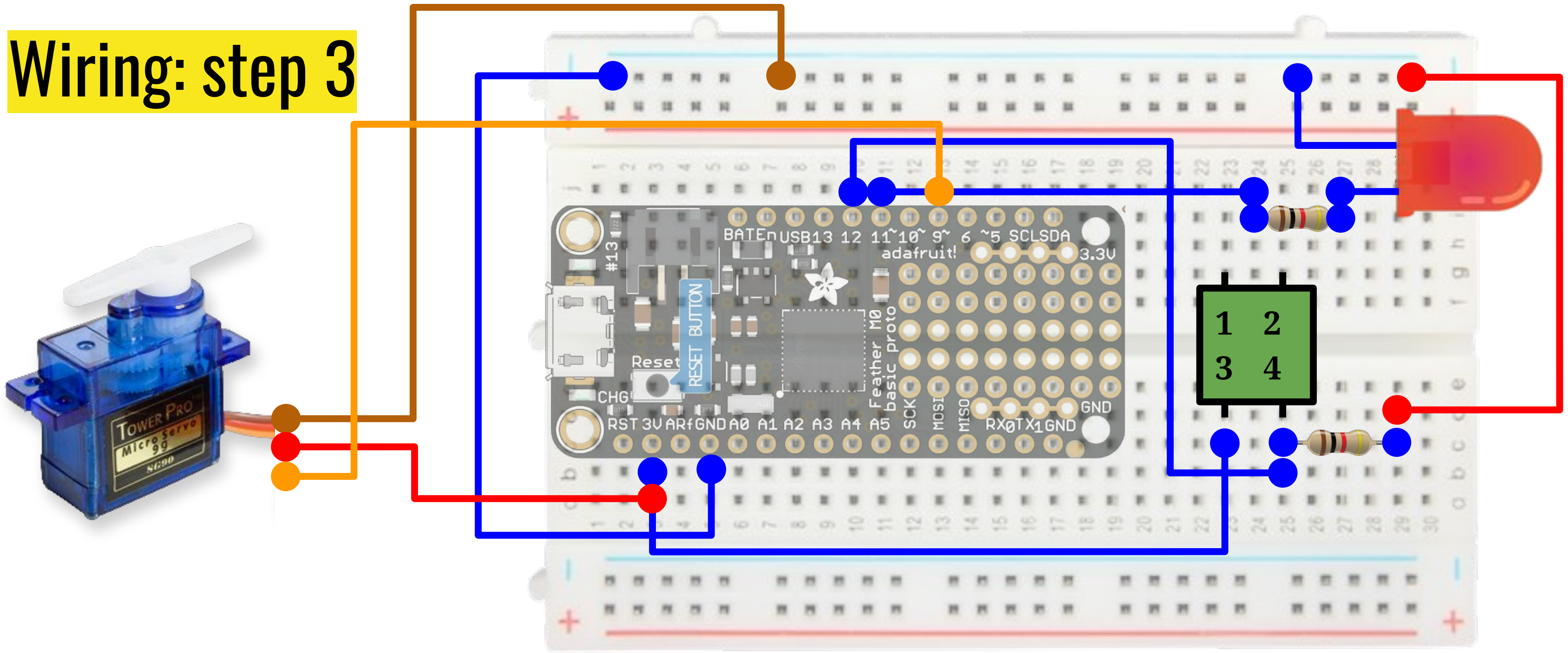
Connect the brown wire of servo motor to the ground column

Wiring: step 2



Connect the red wire of servo motor to 3V pin of Adafruit

Wiring: step 3



Tower Pro
Micro Servo 9g
SG90
BAT En USB 13 12 11 10 9 6 5 SCL SDA
adafruit!
3.3V
#13
RESET BUTTON
Reset
CHG
RST 3V ARf GND A0 A1 A2 A3 A4 A5 SCK MOSI MISO RX0 TX1 GND
GND
Feather M0 basic proto
1
2
3
4

Connect the orange wire of servo motor to pin 9 of Adafruit

Coding: part 1

```
#include <Servo.h>

const int buttonPin = 12;
const int ledPin = 11;

int buttonState = LOW;
int systemState = LOW;
int lastButtonState = LOW;
bool sweepBack = false;

unsigned long lastDebounceTime = 0;
unsigned long debounceDelay = 50;
```

```
Servo servo;
int angle = 0;

void setup() {
  pinMode(buttonPin, INPUT);
  pinMode(ledPin, OUTPUT);
  servo.attach(9);
}
```

Coding: part 2

```
void loop() {
  int reading = digitalRead(buttonPin);
  if (reading != lastButtonState) {
   lastDebounceTime = millis();
  }
  if ((millis() - lastDebounceTime) > debounceDelay) {
   if (reading != buttonState) {
    buttonState = reading;
    if (buttonState == HIGH) {
     systemState = !systemState;
    }
   }
  }
```

Coding: part 3

```
if (systemState) {
  digitalWrite(ledPin, HIGH);
  if (angle < 180 & !sweepBack) {
    angle++;
    delay(15);
  }
  else if(angle > 0 & sweepBack){
    angle--;
    delay(15);
  }
```

```
else if(angle == 180){
    sweepBack = true;
    angle--;
    delay(15);
  }
  else{
    sweepBack = false;
    angle++;
    delay(15);
  }
  servo.write(angle);
}
```

Coding: part 4

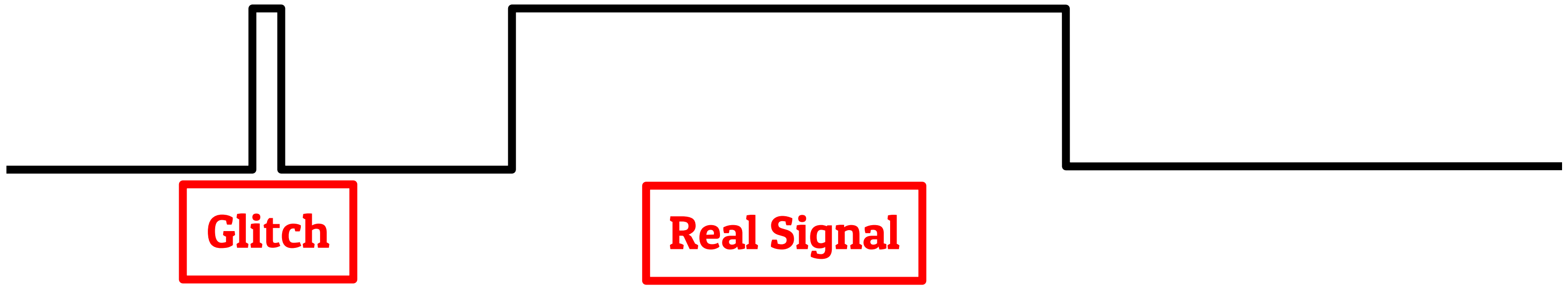
```
else {
    digitalWrite(ledPin, LOW);
    servo.write(90);
  }
  lastButtonState = reading;
}
```

Coding

Press the button, and the servo motor will start to sweep. Press again to turn it off.

The LED indicates whether the system is on or off

Coding: debouncing



Glitch signals may occur in any circuit. In our case, it may cause wrong status transition. We deal with the issue with software method, treat them as noises.

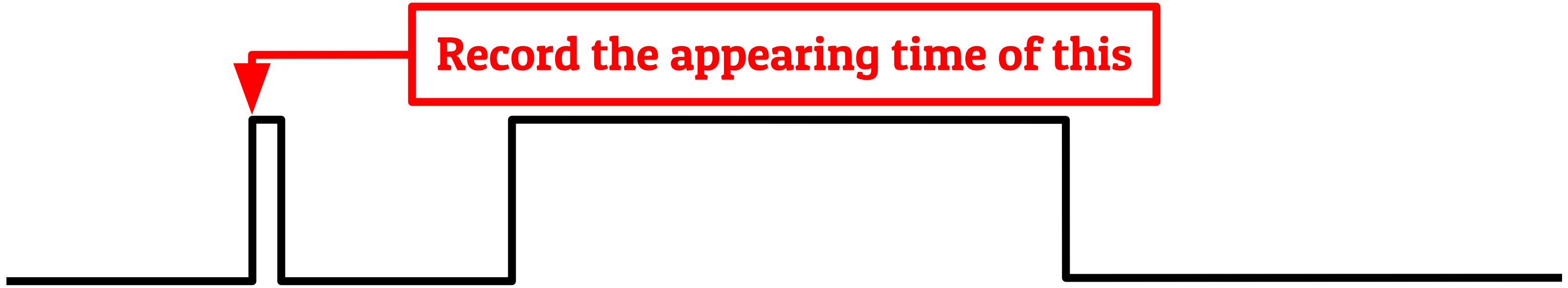
By hardware?
Add a load capacitor!

So, how does the code work?

Coding: debouncing

```
int reading = digitalRead(buttonPin);
if (reading != lastButtonState) {
   lastDebounceTime = millis();
}
```

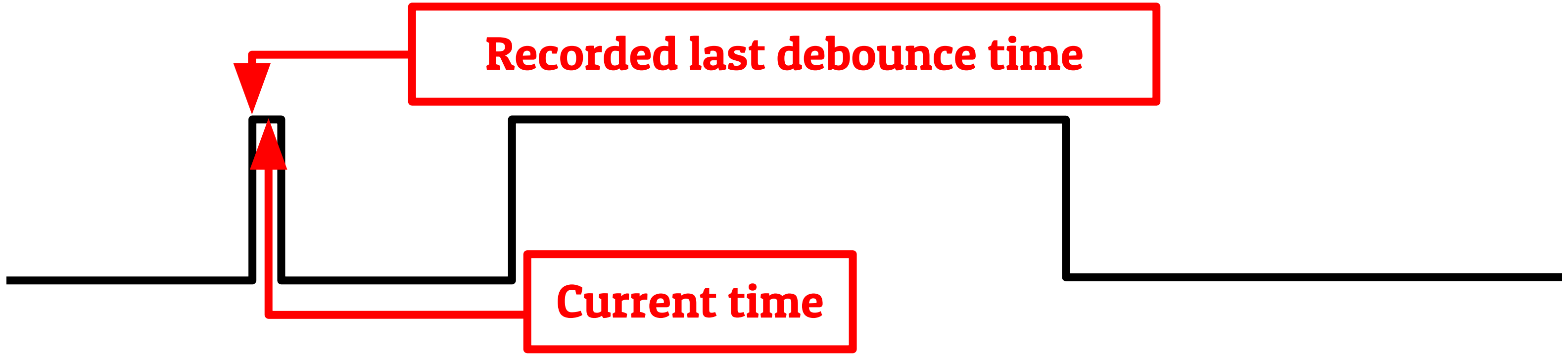
First, we read from the button pin each cycle. If this signal differs from the signal read in last cycle, we record the time



Coding: debouncing

```
if ((millis() - lastDebounceTime) > debounceDelay) {
  if (reading != buttonState) {
   buttonState = reading;
```

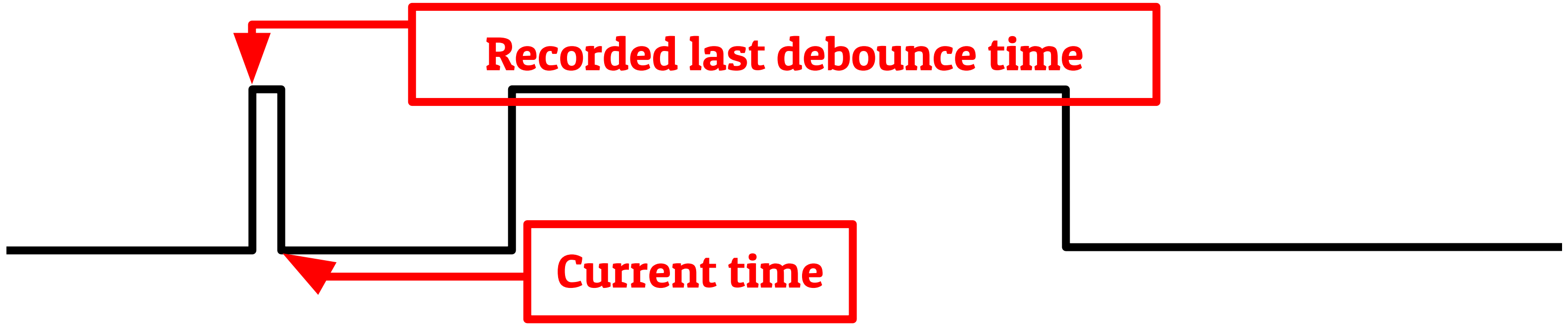
Suppose in the next cycle the timeline moves to the midpoint of this glitch. Since the time passed from pull up is shorter than preset delay, we move on to next cycle



Coding: debouncing

```
if ((millis() - lastDebounceTime) > debounceDelay) {
   if (reading != buttonState) {
     buttonState = reading;
```

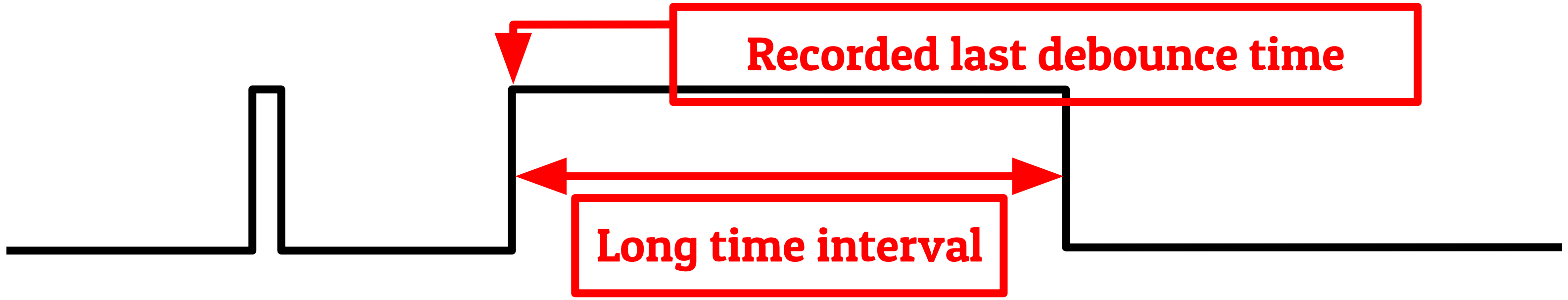
Keep executing until this cycle, where the glitch is going to pull down. Before updating the last debounce time, it has one final chance. Since the glitch signal didn't last for enough time, it does nothing to the button's state



Coding: debouncing

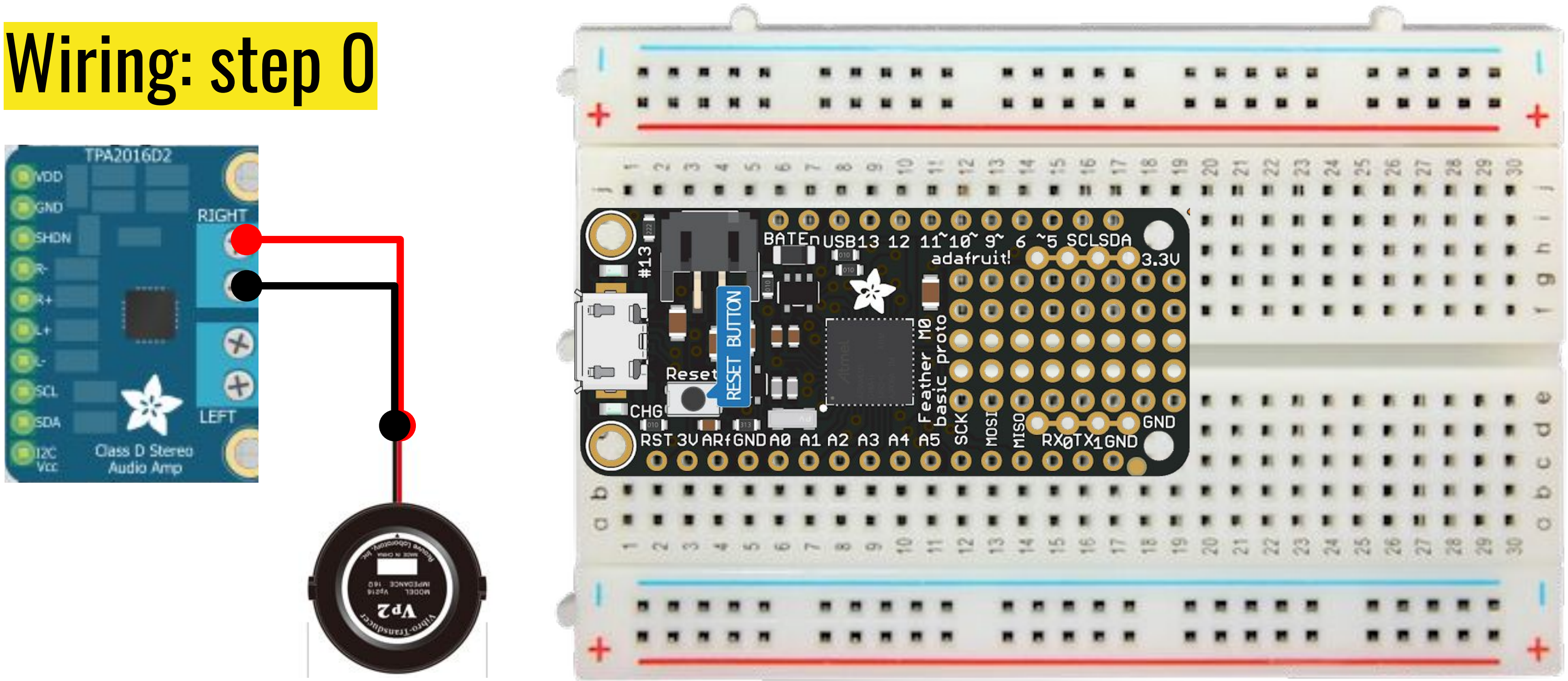
```
if ((millis() - lastDebounceTime) > debounceDelay) {
   if (reading != buttonState) {
     buttonState = reading;
```

Now consider this case. Since the pull up signal has last longer then the debounce delay, it is considered a valid input and thus changing the button's state



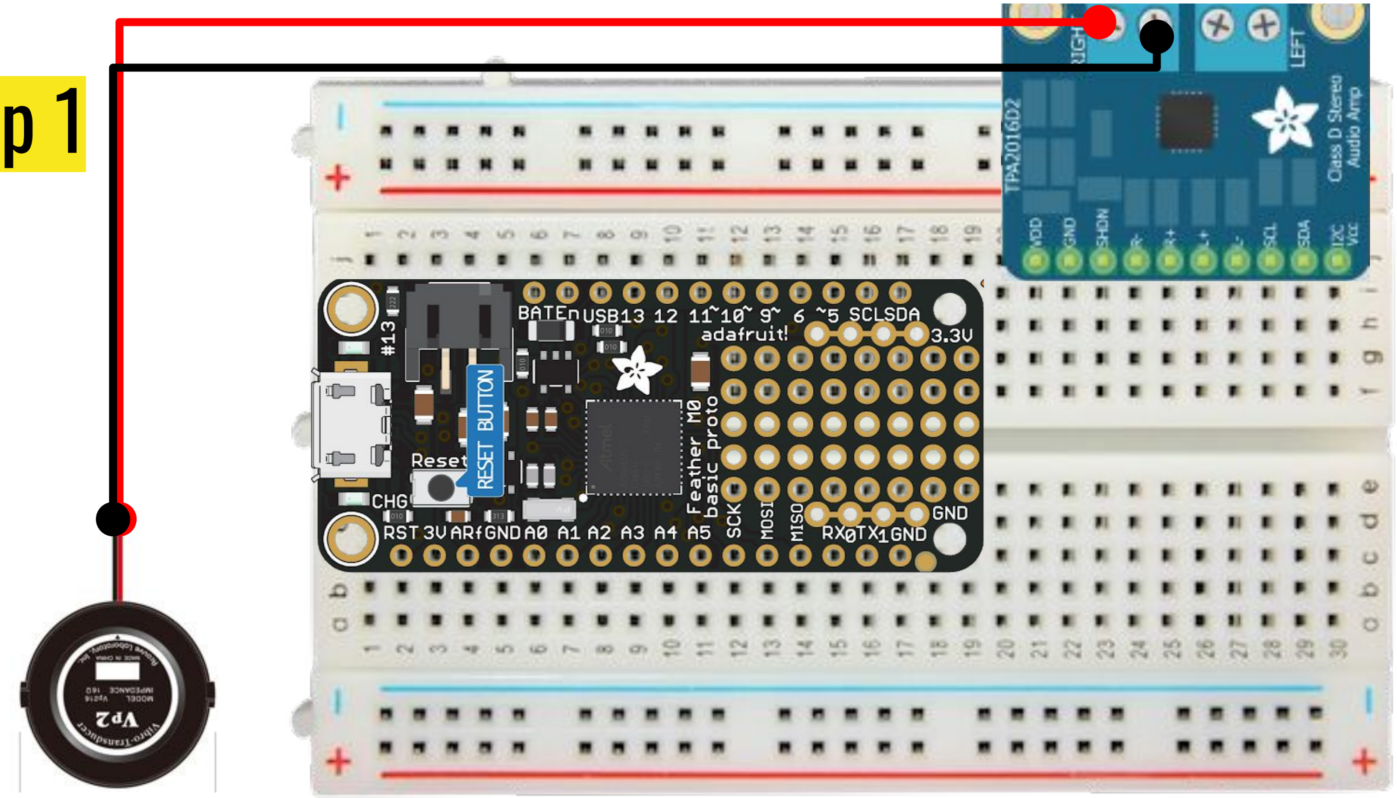
Section 4: Vibration

Wiring: step 0



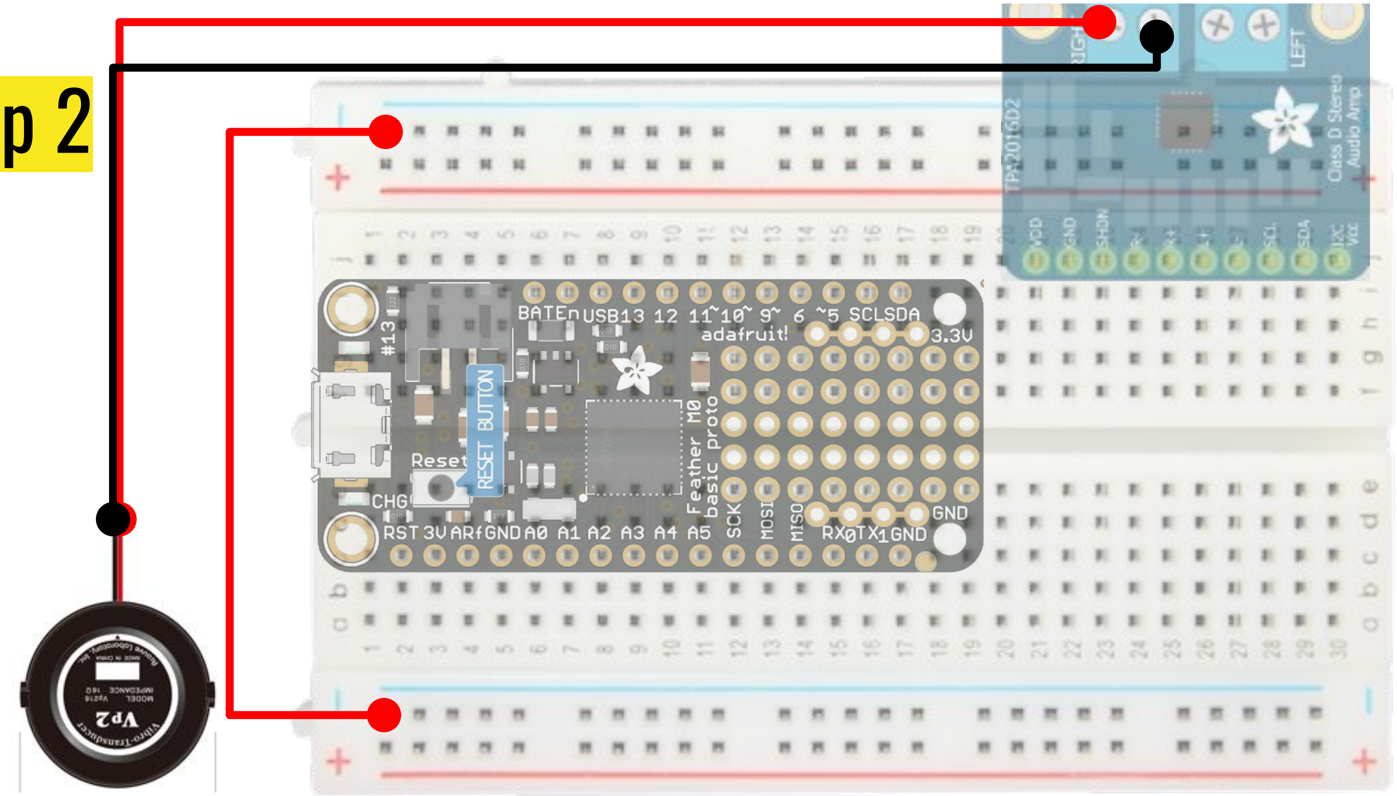
First, remove all components except Adafruit and plug the vibrator into the amplifier

Wiring: step 1



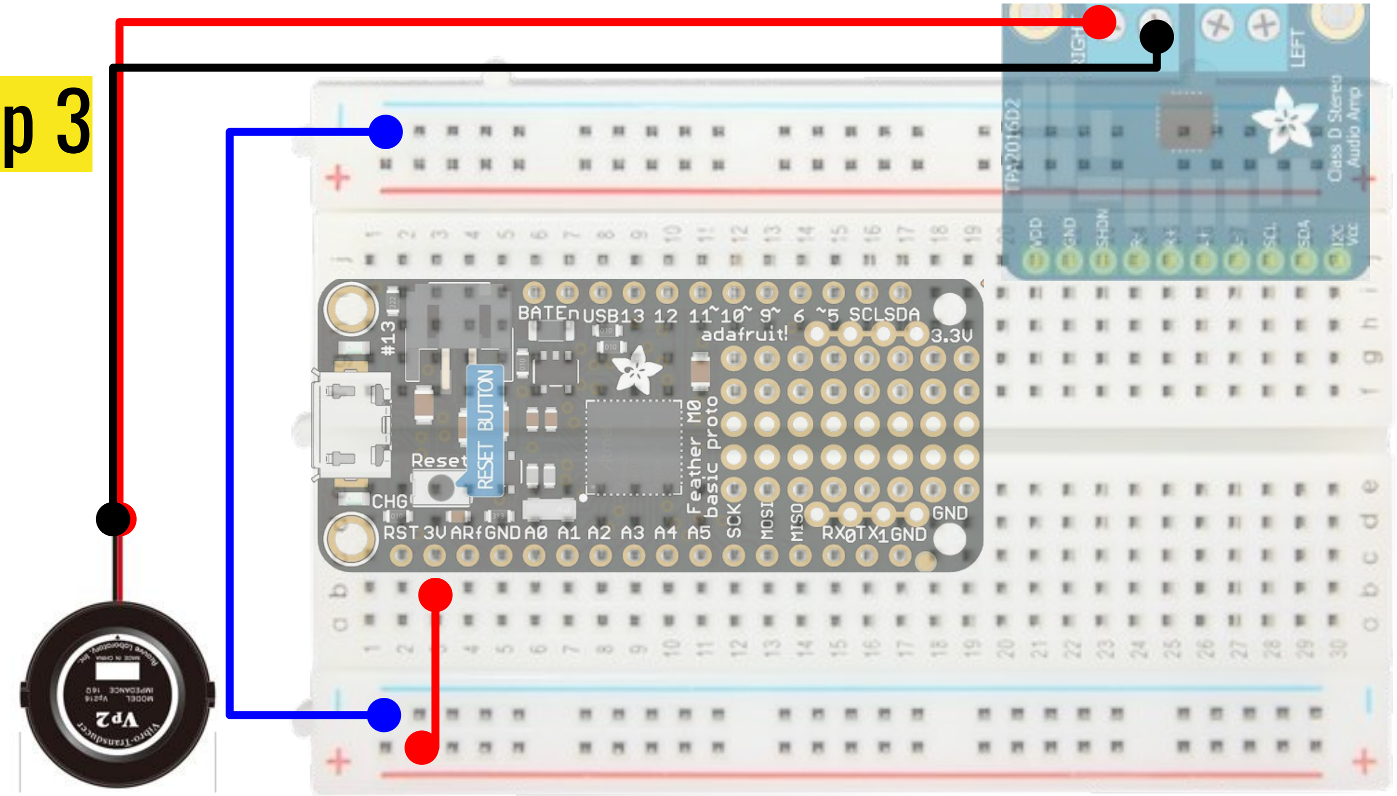
Plug the Adafruit TPA2016D2 board into the breadboard

Wiring: step 2



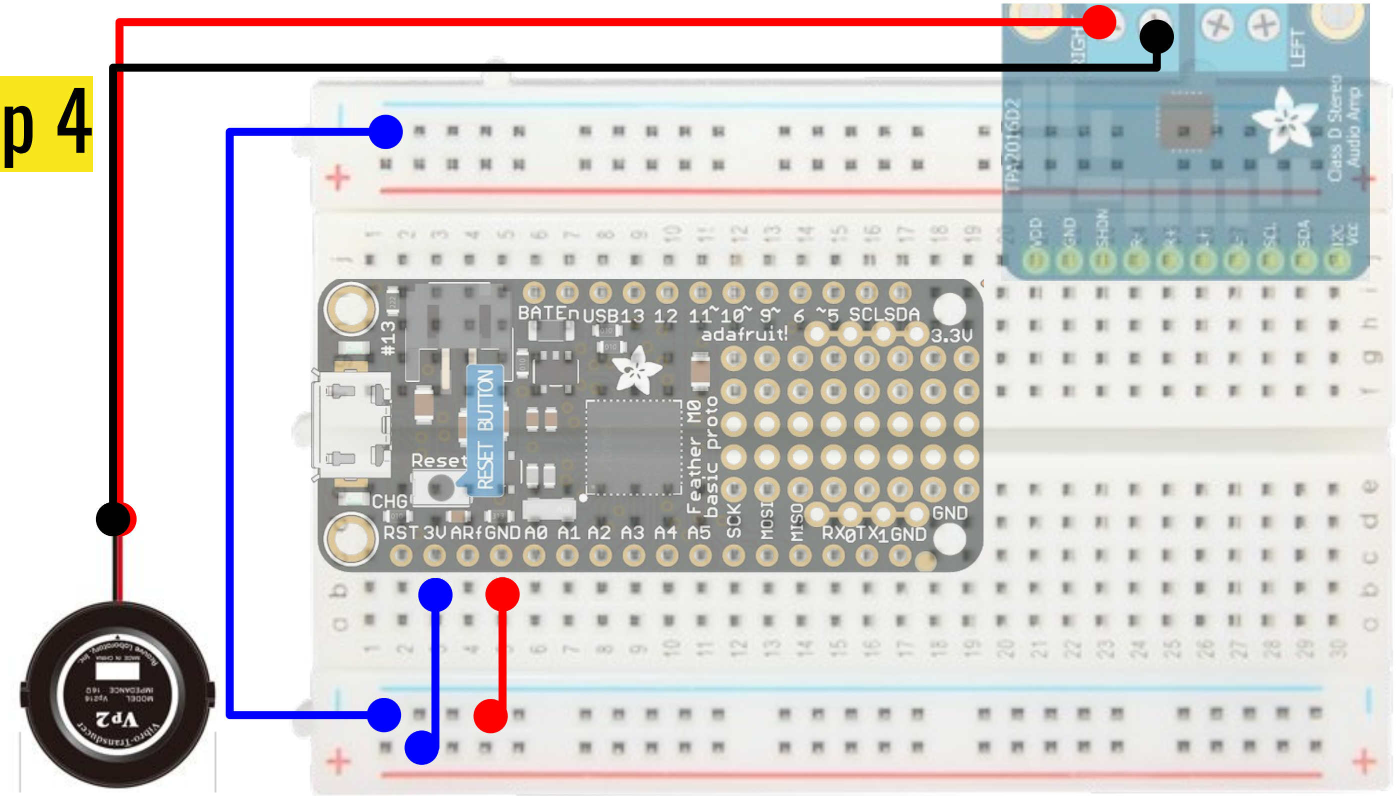
Connect both ground columns

Wiring: step 3



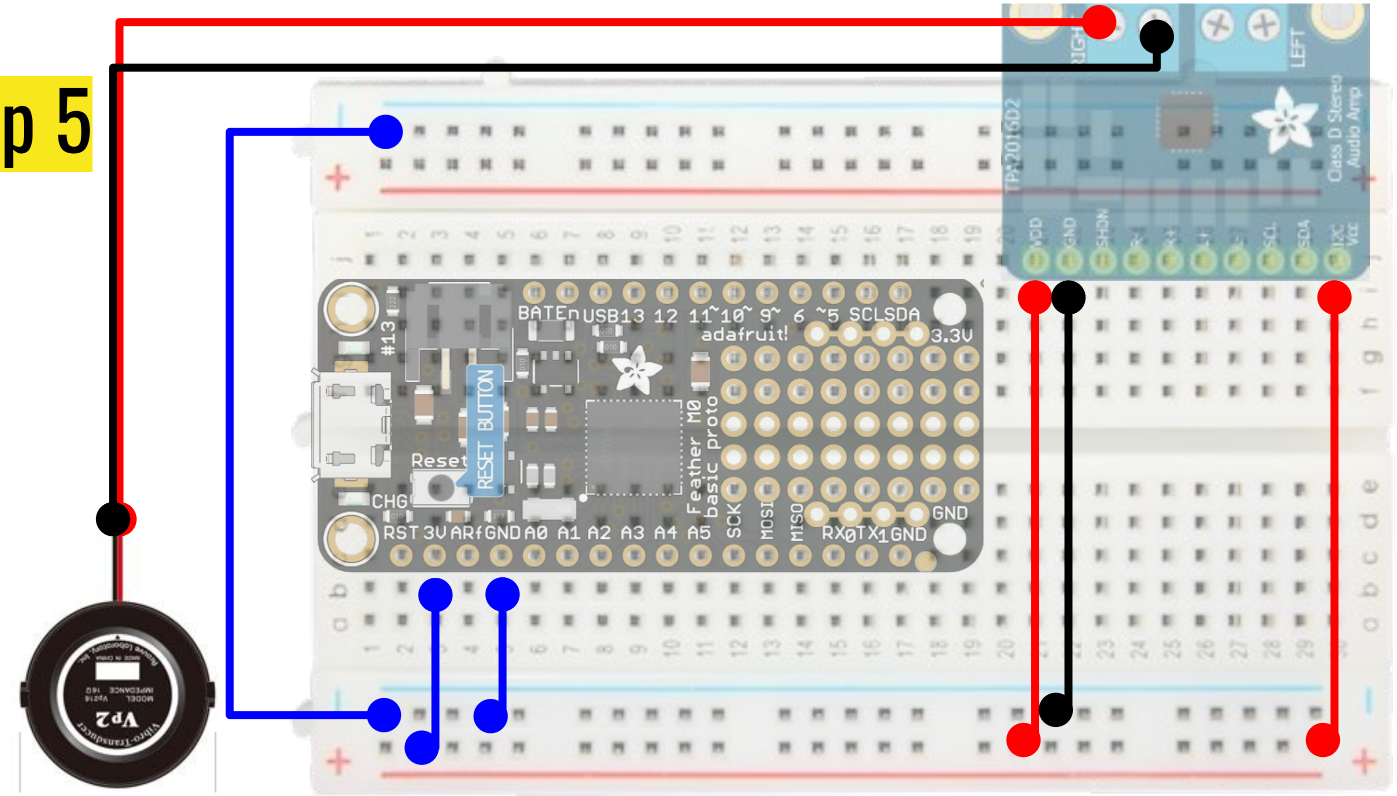
Connect 3V pin to the power column

Wiring: step 4



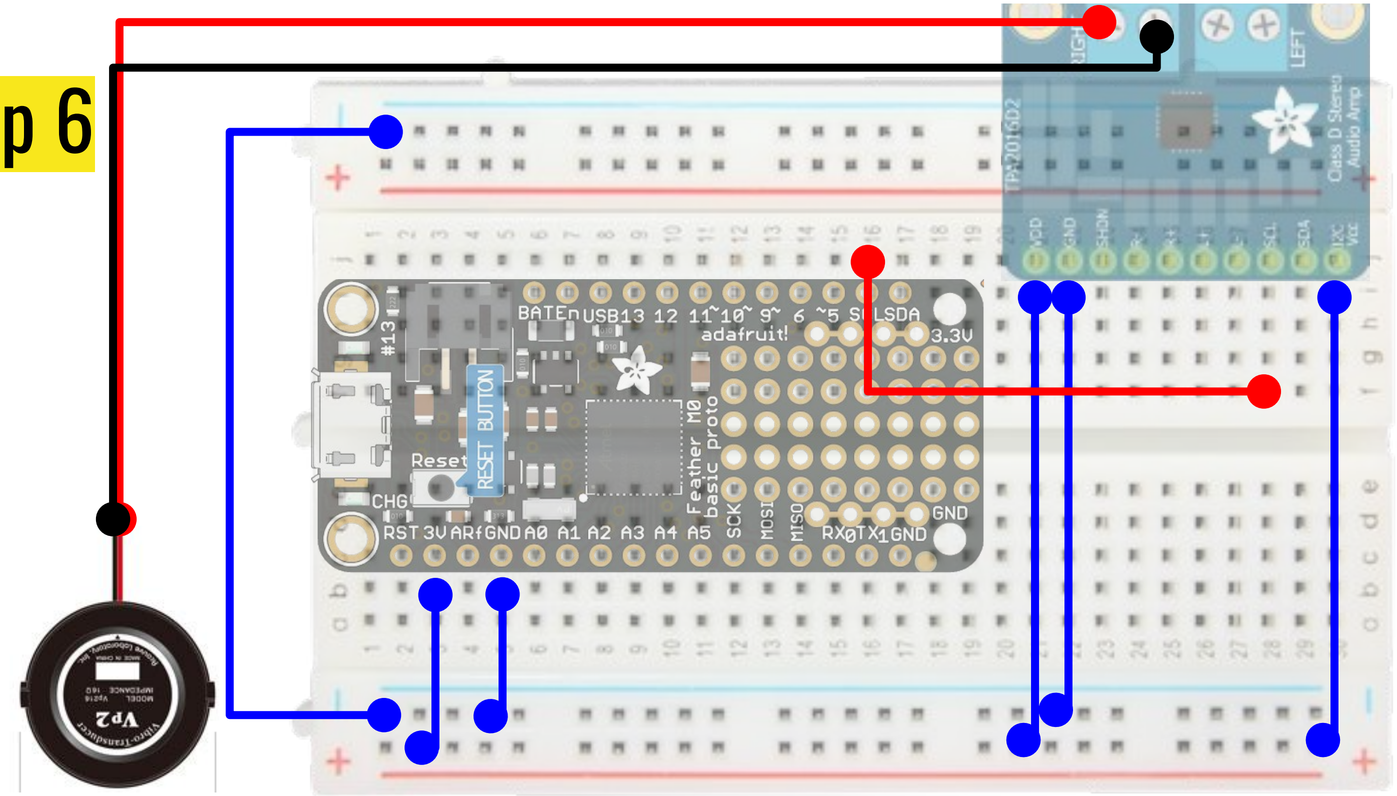
Connect ground pin to the ground column

Wiring: step 5



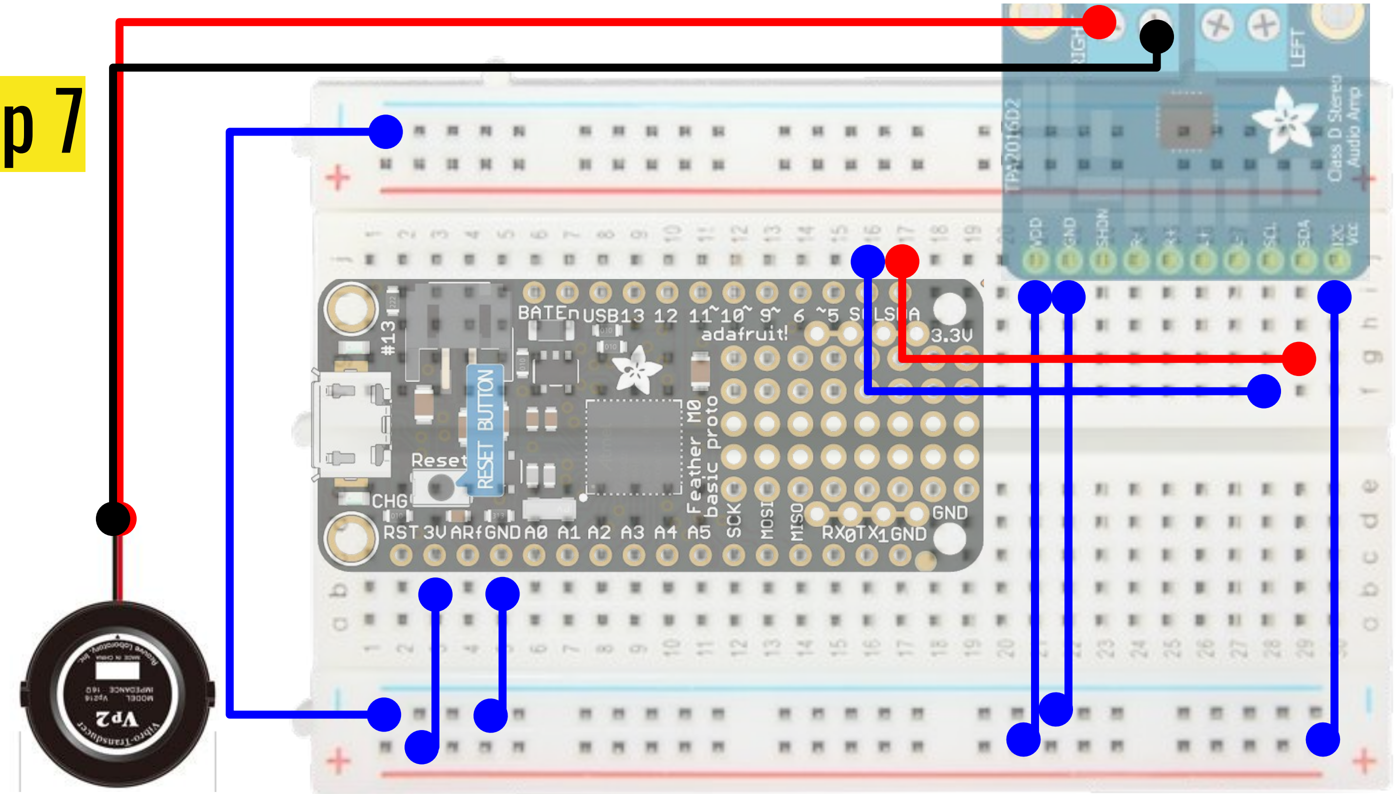
Connect the ground pin and VDD, I2C VCC pins of TPA2016D2 to correct columns

Wiring: step 6



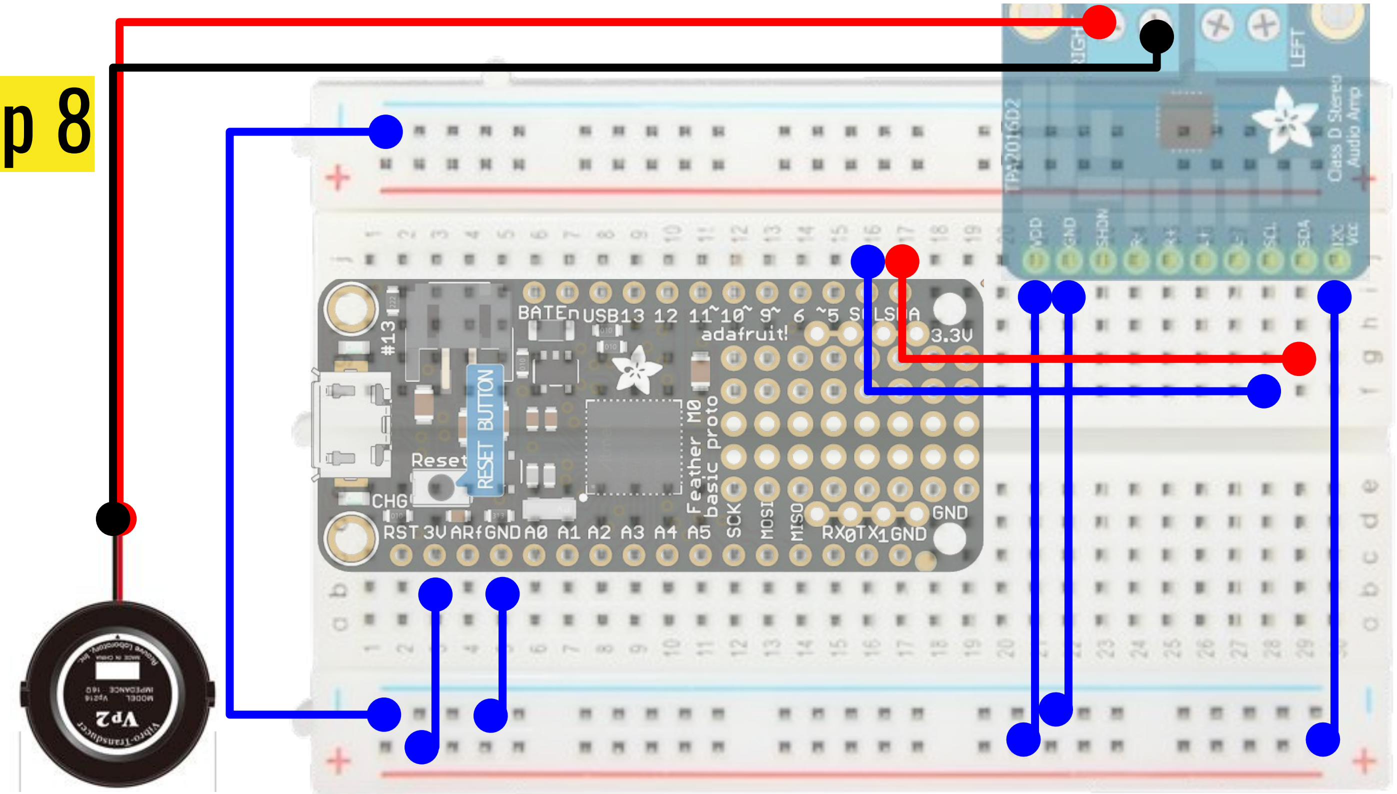
Connect the SCL pins together

Wiring: step 7



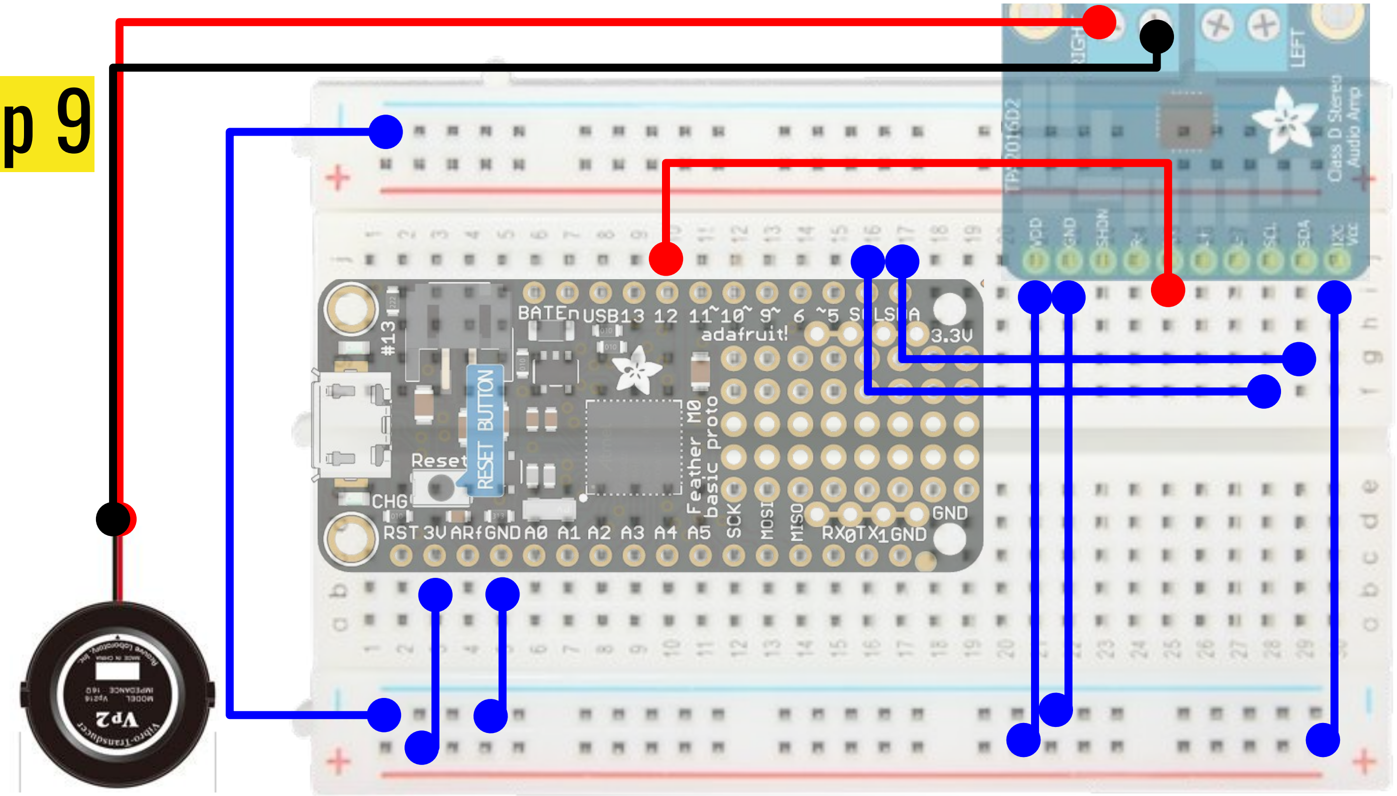
Connect the SDA pins together

Wiring: step 8



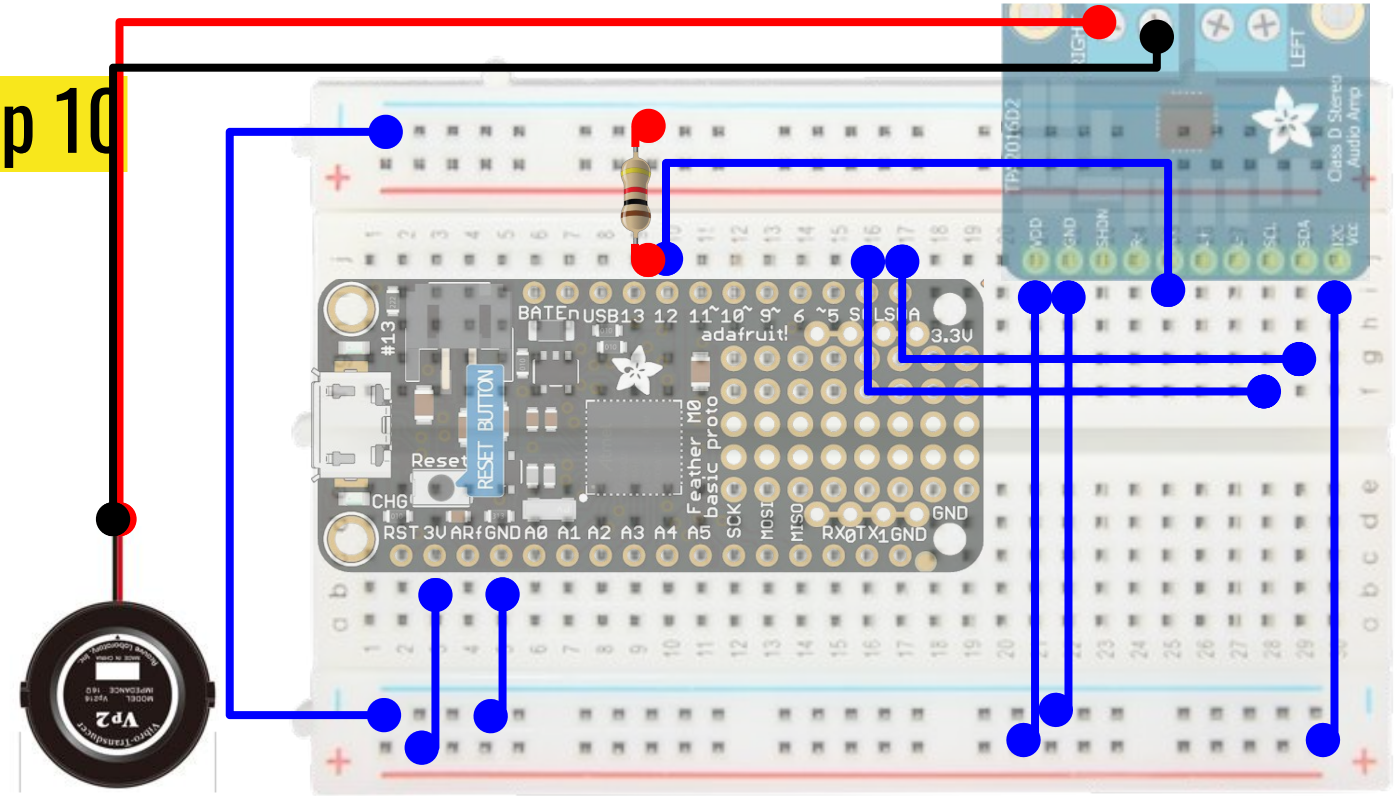
Connect the SDA pins together

Wiring: step 9



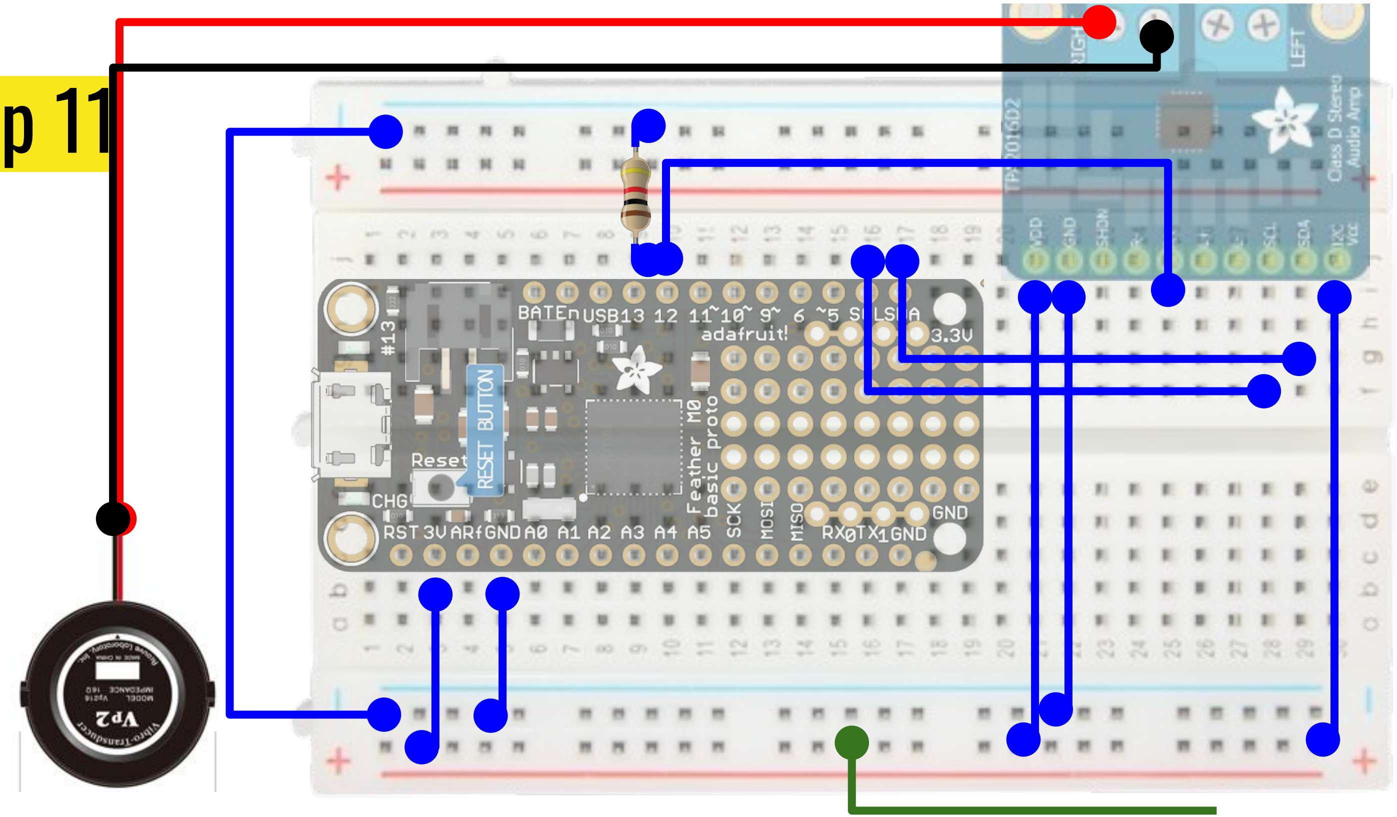
Connect the R+ pin with pin 12

Wiring: step 10



Plug a resistor between pin 13 and the ground column

Wiring: step 11



Plug a wire into the power column, and leave the other end empty for now

Coding: part 1

```
#include <Wire.h>
#include "Adafruit_TPA2016.h"

#define CPU_HZ 48000000
#define TIMER_PRESCALER_DIV 1024

void startTimer(int frequencyHz);
void setTimerFrequency(int frequencyHz);
void TC3_Handler();
void vibrate(int pinNum, int half_wavelength,int vibrateTime);

int timer = 0;
```

Coding: part 2

```
int isVibrate = 0;
int vibrateFreq = 0;
int vibrateTime = 0;
Adafruit_TPA2016 audioamp01 = Adafruit_TPA2016();
void setup() {
  pinMode(12, OUTPUT);
  pinMode(13, INPUT);
  startTimer(1000);
  audioamp01.begin();
  audioamp01.enableChannel(true, true);
  audioamp01.setLimitLevelOff();
  audioamp01.setAGCCompression(TPA2016_AGC_OFF);
}
```

Coding: part 3

```
void loop() {
  if (digitalRead(13)) {
    isVibrate = 1;
    vibrateFreq = 25;
    vibrateTime = 1000;}
  else {
    isVibrate = 0;
    vibrateFreq = 0;
    vibrateTime = 0;
  }
  audioamp01.setGain(30);
  vibrate(12,vibrateFreq,vibrateTime);
}
```

Coding: part 4

```
void setTimerFrequency(int frequencyHz) {
  int compareValue = (CPU_HZ / (TIMER_PRESCALER_DIV * frequencyHz))-1;
  TcCount16* TC = (TcCount16*) TC3;
  // Make sure the count is in a proportional position to where it was
  // to prevent any jitter or disconnect when changing the compare value.
  TC->COUNT.reg = map(TC->COUNT.reg, 0, TC->CC[0].reg, 0, compareValue);
  TC->CC[0].reg = compareValue;
  Serial.println(TC->COUNT.reg);
  Serial.println(TC->CC[0].reg);
  while (TC->STATUS.bit.SYNCBUSY == 1);
}
```

Coding: part 5

```
        void startTimer(int frequencyHz) {
         REG_GCLK_CLKCTRL = (uint16_t) (GCLK_CLKCTRL_CLKEN |
GCLK_CLKCTRL_GEN_GCLK0 | GCLK_CLKCTRL_ID_TCC2_TC3) ;

         while ( GCLK->STATUS.bit.SYNCBUSY == 1 ); // wait for sync
         TcCount16* TC = (TcCount16*) TC3;
         TC->CTRLA.reg &= ~TC_CTRLA_ENABLE;
         while (TC->STATUS.bit.SYNCBUSY == 1); // wait for sync
         TC->CTRLA.reg |= TC_CTRLA_MODE_COUNT16;
         while (TC->STATUS.bit.SYNCBUSY == 1); // wait for sync
         TC->CTRLA.reg |= TC_CTRLA_WAVEGEN_MFRQ;
         while (TC->STATUS.bit.SYNCBUSY == 1); // wait for sync
```

Coding: part 6

```
  TC->CTRLA.reg |= TC_CTRLA_PRESCALER_DIV1024;
  while (TC->STATUS.bit.SYNCBUSY == 1); // wait for sync
  setTimerFrequency(frequencyHz);
  // Enable the compare interrupt
  TC->INTENSET.reg = 0;
  TC->INTENSET.bit.MC0 = 1;
  NVIC_EnableIRQ(TC3_IRQn);

  TC->CTRLA.reg |= TC_CTRLA_ENABLE;
  while (TC->STATUS.bit.SYNCBUSY == 1); // wait for sync
}
```

Coding: part 7

```
void TC3_Handler() {
 TcCount16* TC = (TcCount16*) TC3;
 if (TC->INTFLAG.bit.MC0 == 1) {
  TC->INTFLAG.bit.MC0 = 1;
  if(isVibrate == 1)
   timer ++;
  else
   timer =0;
 }
}
```

Coding: part 8

```
void vibrate(int pinNum, int half_wavelength,int vibrateTime){
 if(timer < vibrateTime && isVibrate == 1){
  if((timer/half_wavelength)%2 ==0)
   digitalWrite(pinNum,HIGH);
  else
   digitalWrite(pinNum,LOW);
 }
 else{
  isVibrate = 0;
 }
}
```

Coding

Touch pin 13 of Adafruit with the empty end of the green wire….

The vibrator will start to vibrate!