# How to connect a push button or switch

Buttons and switches are a way of opening and closing a circuit, i.e. making and breaking a connection as one of the most rudimentary forms of sensor you can use with an Arduino.

There are dozens of different types of switches and buttons, but at their most basic is the momentary push button which we'll be focusing on in the wiring and getting started sections below. However the same approach applies to these as it does to any other type of button or switch.

## Different types

- Rocker switch
- Push button
- Tactile button
- Reed switch
- Tilt switch
- Key operated switch
- Rotary switch
- Slide switch
- Micro switch
- Toggle switch

There are many different types for different purposes:

## Push buttons

]

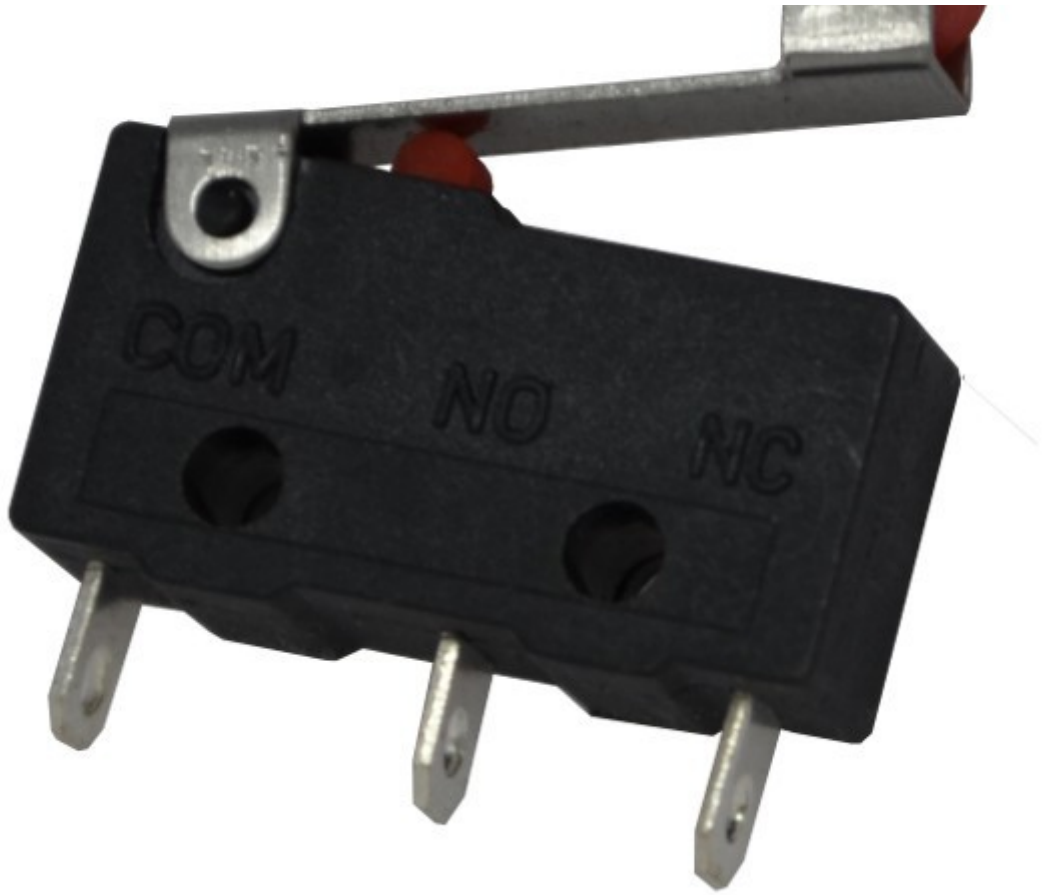
Push buttons like those found in a computer keyboard are really useful for activating an action, like a start video button.

## Rocker, slide and toggle switches


]

Rocker, slide and toggle switches work more like light switches holding their position, they can be a good way of indicating the mode of a device, such as playing video forward or backwards.

## Microswitches




]

Micro switches can with motors to detect when it has reached the end of movement, such as in a 3D printer to stop the motor going too far over the end, or to detect if a draw is open or closed.

# Wiring

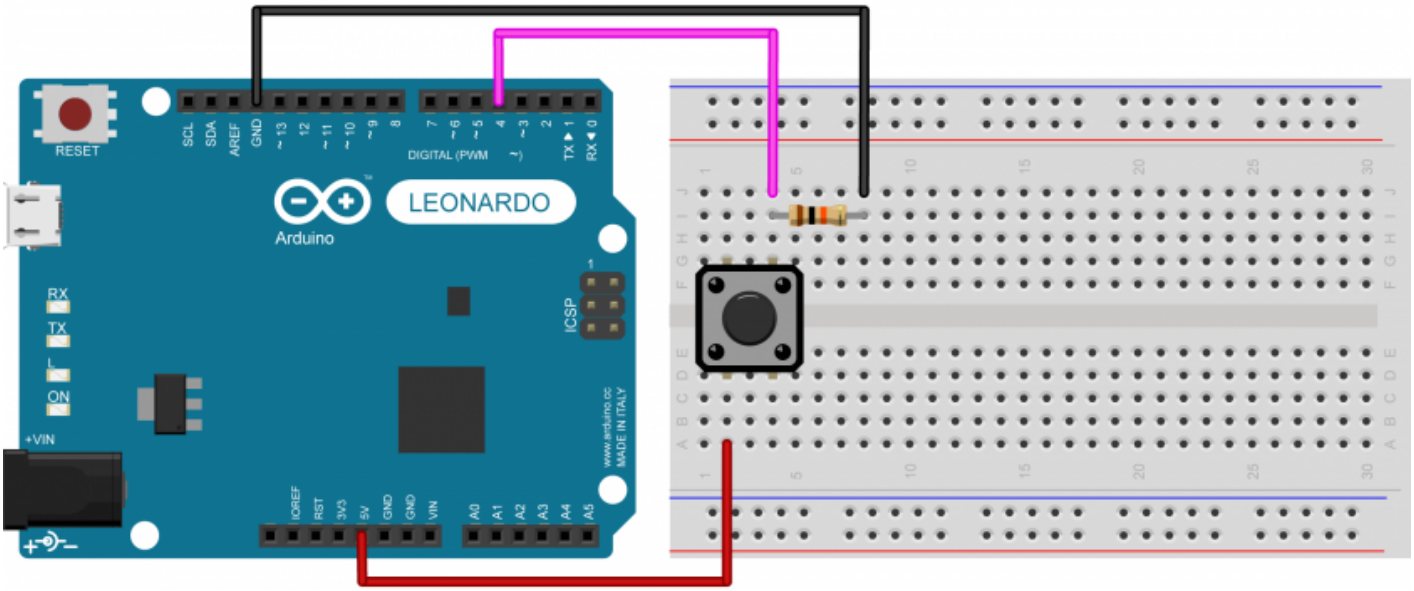
Wiring up buttons and switches is simple, however there is a complexity you might not have thought about.

Although a push button like that in the diagram only has two connections, which are closed by pressing the button, you have to add a resistor to make the circuit work properly.

When the button is pressed the current on one side is able to flow to the other, however when the button is released the circuit is broken and the wire to the Arduino is known as floating, the voltage is indeterminate, so we need to connect it to ground to ensure the Arduino reads 0V.

It's not possible however to do this otherwise when you apply 5V by closing the circuit you would create a short circuit, instead we connect the Arduino pin through a high value 10KΩ resistor to

ground, this allows the circuit to quickly reach 0V when the button is released but prevents large amounts of current flowing when the button is pressed.



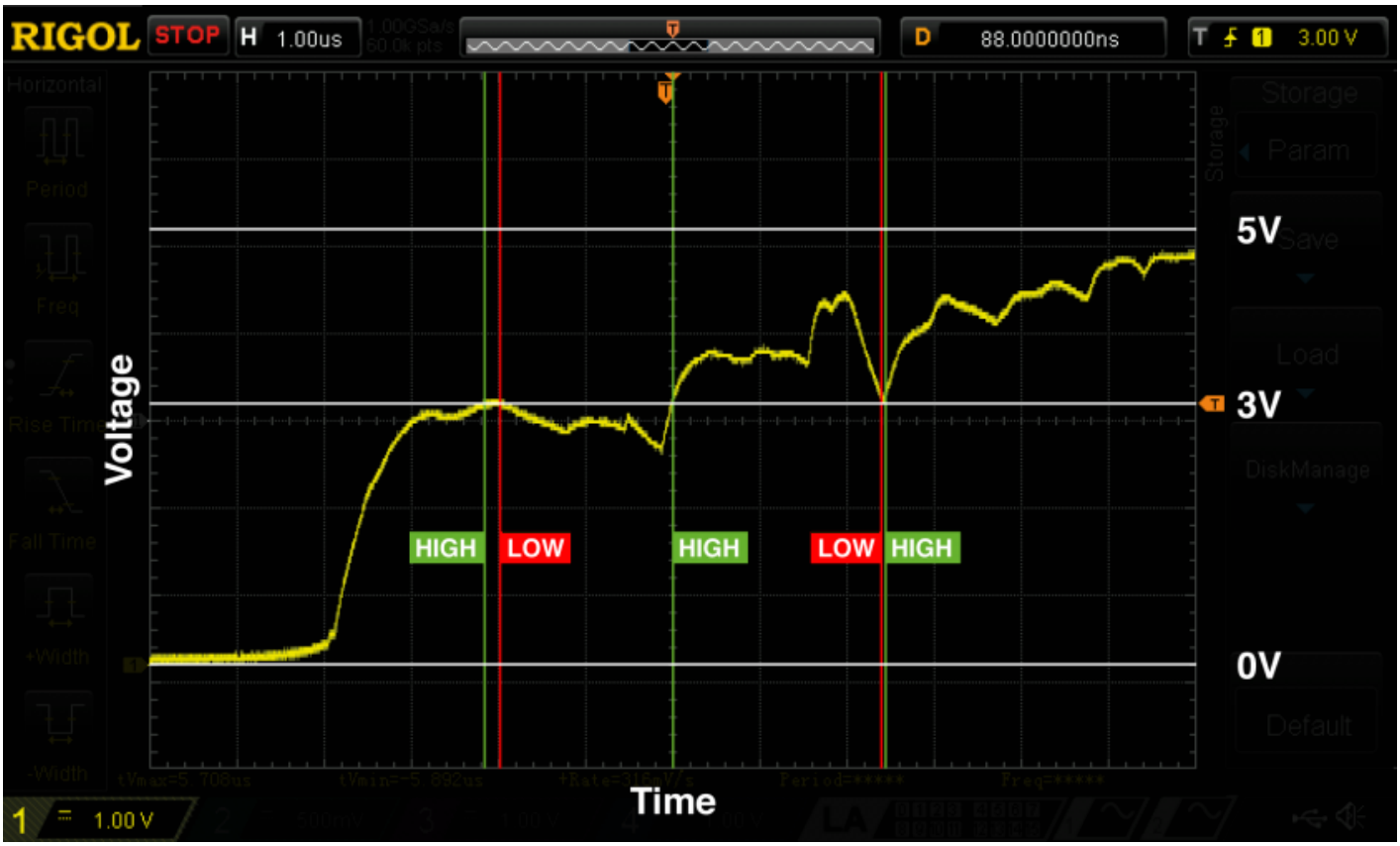

# Getting started

The following is a simple circuit that will get your button controlling the LED built into the Arduino.

```
#define ledPin 13
#define buttonPin 7

void setup() {
  pinMode( ledPin, OUTPUT );
  pinMode( buttonPin, INPUT );
}

void loop() {
  boolean btnState = digitalRead( buttonPin );

  if ( btnState == HIGH ) {
    digitalWrite( ledPin, HIGH );
  } else {
    digitalWrite( ledPin, LOW );
  }
}
```

If you want to add a toggle functionality such that one press causes the LED to come on, and another press then turns it off, so you don't have to hold the button down things get a little bit more complex.

The Arduino is a powerful computer and operates many times faster than human perception, as such when the mechanical push button is closed there is a small amount of 'bounce' where the circuit makes and breaks the connection a few times before it settles, this is detected by the Arduino as multiple presses.




] *This diagram shows the signal bouncing up and down over a period of 10µS (0.00001 seconds)*

In effect this means that each time you press the button to toggle just once it toggles multiple times, you can fix this either with a small capacitor, or modifications to your Arduino sketch.

The code here adds two major changes, first it tracks the current and previous button state through each loop meaning it can see if the button has changed from `LOW` to `HIGH`, and then adds a delay of 75ms to allow the button to settle but keep it fast enough that the user doesn't perceive this delay.

```
#define ledPin 13
#define buttonPin 7

boolean ledState = LOW;
boolean prevBtnState = LOW;

void setup() {
  pinMode( ledPin, OUTPUT );
  pinMode( buttonPin, INPUT );
}
```

```
void loop() {
  boolean btnState = digitalRead( buttonPin );

  if ( btnState == HIGH && prevBtnState == LOW ) {
    ledState = ! ledState;
    delay( 75 );
  }

  digitalWrite( ledPin, ledState );

  prevBtnState = btnState;
}
```

---

Revision #4
Created 10 July 2021 23:12:09 by Tom Lynch
Updated 29 April 2024 22:24:58 by Tom Lynch