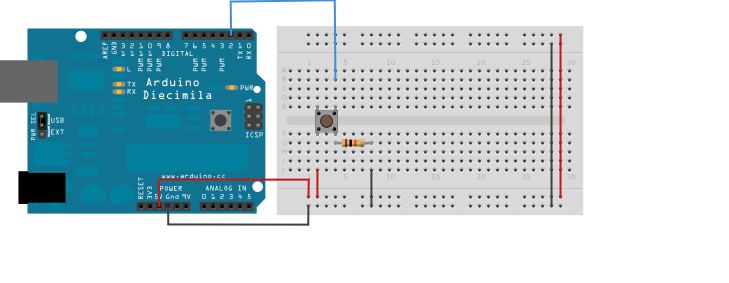
Button / Digital Read (use external resistor)

Pushbuttons or switches connect two points in a circuit when you press them. This example turns on the built-in LED on pin 13 when you press the button.

Hardware

* Arduino or Genuino Board
* Momentary button or Switch
* 10K ohm resistor
* hook-up wires
* breadboard

Circuit



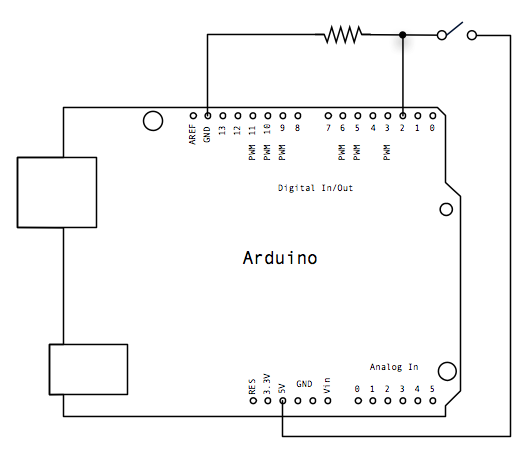
Connect three wires to the board. The first two, red and black, connect to the two long vertical rows on the side of the breadboard to provide access to the 5 volt supply and ground. The third wire goes from digital pin 2 to one leg of the pushbutton. That same leg of the button connects through a pull-down resistor (here 10K ohm) to ground. The other leg of the button connects to the 5 volt supply.

When the pushbutton is open (unpressed) there is no connection between the two legs of the pushbutton, so the pin is connected to ground (through the pull-down resistor) and we read a LOW. When the button is closed (pressed), it makes a connection between its two legs, connecting the pin to 5 volts, so that we read a HIGH.

You can also wire this circuit the opposite way, with a pullup resistor keeping the input HIGH, and going LOW when the button is pressed. If so, the behavior of the sketch will be reversed, with the LED normally on and turning off when you press the button.

If you disconnect the digital I/O pin from everything, the LED may blink erratically. This is because the input is "floating" - that is, it will randomly return either HIGH or LOW. That's why you need a pull-up or pull-down resistor in the circuit.

Schematic

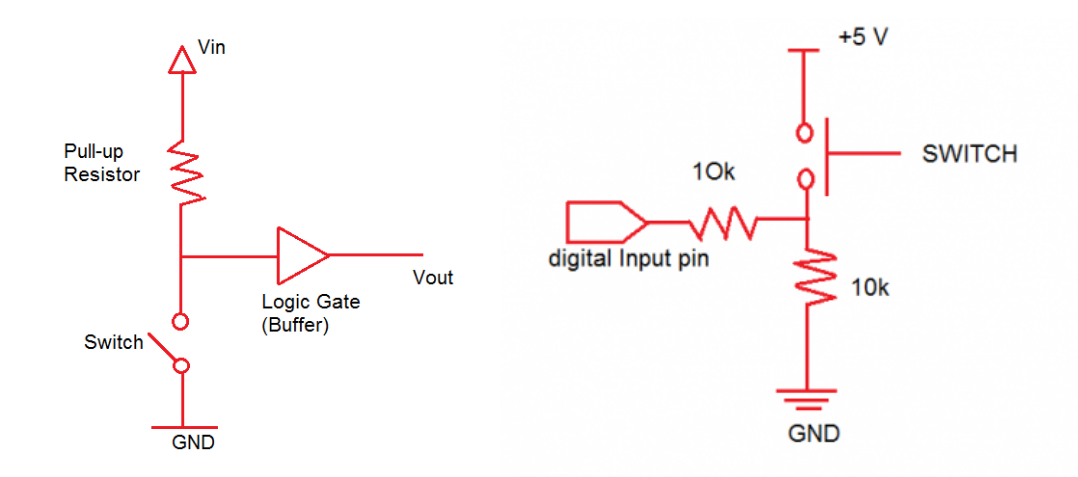


Code

*// constants won't change. They're used here to set pin numbers:*  
const int buttonPin = 2;     *// the number of the pushbutton pin*  
const int ledPin =  13;      *// the number of the LED pin*  
  
*// variables will change:*  
int buttonState = 0;         *// variable for reading the pushbutton status*  
  
void **setup**() {  
  *// initialize the LED pin as an output:*  
  pinMode(ledPin, OUTPUT);  
  *// initialize the pushbutton pin as an input:*  
  pinMode(buttonPin, INPUT);  
}  
  
void **loop**() {  
  *// read the state of the pushbutton value:*  
  buttonState = digitalRead(buttonPin);  
  
  *// check if the pushbutton is pressed. If it is, the buttonState is HIGH:*  
  if (buttonState == HIGH) {  
    *// turn LED on:*  
    digitalWrite(ledPin, HIGH);  
  } else {  
    *// turn LED off:*  
    digitalWrite(ledPin, LOW);  
  }  
}

================================

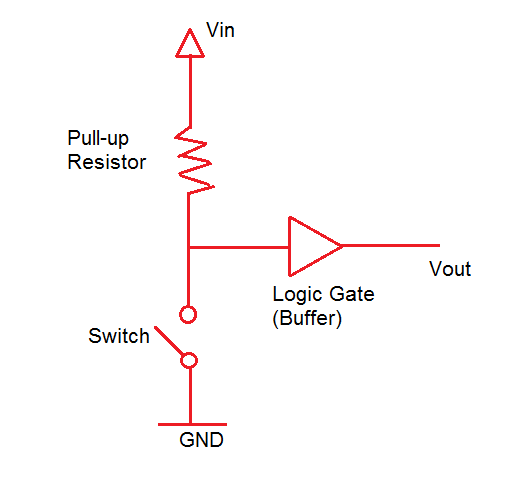
# PULL-UP AND PULL-DOWN RESISTORS



If you check out any digital electronic circuit, you’ll mostly find pull-up and pull-down resistors in them. They are used to correctly bias the inputs of digital gates to stop them from floating around randomly when there is no input condition. For any microcontroller in an embedded system such as Arduino, pull-up and pull-down resistors utilize input and output signals for communication with external hardware devices, the General Purpose Input Output (GPIO). Implementing the pull-up and pull-down resistors in the circuit will let you achieve either ‘high’ or ‘low’ states. If you don’t implement it and there’s nothing connected to your GPIO pins, your program will read a “floating” impedance state.

## ****PULL-UP RESISTORS****

A pull-up resistor is used to establish an additional loop over the critical components while making sure that the voltage is well-defined even when the switch is open. It is used to ensure that a wire is pulled to a high logical level in the absence of an input signal. It is not a special kind of resistor. They are simple fixed-value resistors connected between the voltage supply and the appropriate pin that defines the input or output voltage in the absence of a driving signal. When the switch is open, the voltage of the gate input is pulled up to the level of the input voltage. When the switch is closed, the input voltage at the gate goes directly to the ground. You need to use a pull-up resistor when you have a low default impedance state and wish to pull the signal to ‘high’.

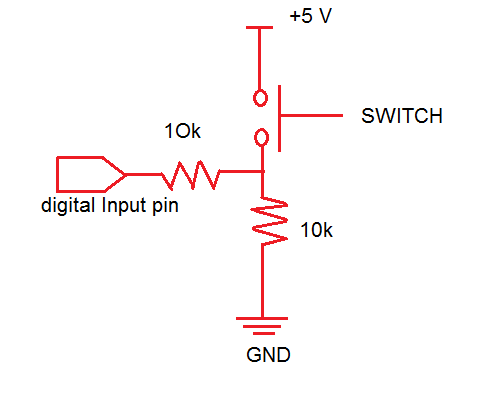


Pull-up Resistor Schematic

In the figure above, a pull-up resistor with a fixed value was used to connect the voltage supply and a particular pin in the digital logic circuit. The pull-up resistor is paired with a switch to ensure that the voltage between Ground and VCC is actively controlled when the switch is open. At the same time, it will not affect the state of the circuit. If we do not use a pull-up resistor, it will result in a short circuit. This is because the pin cannot be directly shorted to the ground or VCC as this will eventually damage the circuit. Following the principle of Ohm’s law, if there is a pull-up resistance, a small amount of current will flow from the source to the resistors and to the switch before reaching the ground.

## ****PULL-DOWN RESISTORS****

On the other hand, a **pull-down resistor** is used to ensure that inputs to logic systems settle at expected logic levels whenever external devices are disconnected or of high impedance. It ensures that the wire is at a defined low logic level even when there are no active connections with other devices. The pull-down resistor holds the logic signal near to zero volts (0V) when no other active device is connected. It pulls the input voltage down to the ground to prevent an undefined state at the input. It should have a larger resistance than the impedance of the logic circuit. Otherwise, it will make the input voltage at the pin on constant logical low value no matter the position of the switch. When the switch is open, the voltage of the gate input is pulled down to the level of the ground. When the switch is closed, the input voltage at the gate goes to Vin. Without the resistor, the voltage levels would virtually float between the two voltages.



Pull-down Resistor Schematic

Just like the pull-up resistor in the first figure, the pull-down resistors in this circuit also ensures that the voltage between VCC and a microcontroller pin is actively controlled when the switch is open. Unlike the pull-up resistor, the pull-down resistor pulls the pin to a low value instead of high value. The pull-down resistor which is connected to the ground or 0V sets the digital logic level pin to default or 0 until the switch is pressed and the logic level pin becomes high. Therefore, the small amount of current flows from the 5-V source to the ground using the closed switch and pull-down resistor preventing the logic level pin from getting shorted with the 5-V source.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Your full name | Tel. | Email | Group | Su23 |
| Nguyễn Xuân Trung | 0944353601 | [Trungnxqe170172@fpt.edu.vn](mailto:Trungnxqe170172@fpt.edu.vn) | 2 – SE17B02 |  |
|  |  |  |  |  |
| Screen shot:    https://www.tinkercad.com/things/6817DFT2JNS-9abuttondigitalread/editel?sharecode=OdcAG8NxjtVZ11-1mF\_YpiBkiznXNeEWfXmdIfsYdm8 | | | | |
| How it works?  Pin2 is pulled down (the line goes through the button). When button is pressed, pin2 read HIGH. The microcontrolled is programmed to light the board led whenever this happened. | | | | |