Controlling an LED with Raspberry Pi GPIO Pins

Nano Version

Date Created April 2021

Reviewed

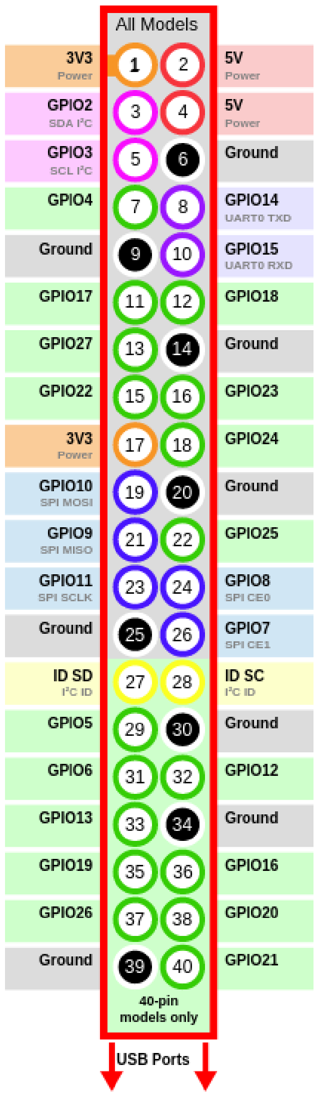
Version

Description

One of the biggest selling points of the Raspberry Pi is its GPIO, or General Purpose Input/Output ports. They are the little pins sticking out of the circuit board and allow you to plug various devices into your Raspberry Pi. With a little programming, you can then control them or detect what they are doing.

In this tutorial we are going to light an LED. In addition to your Raspberry Pi running Raspbian, what you will need is:

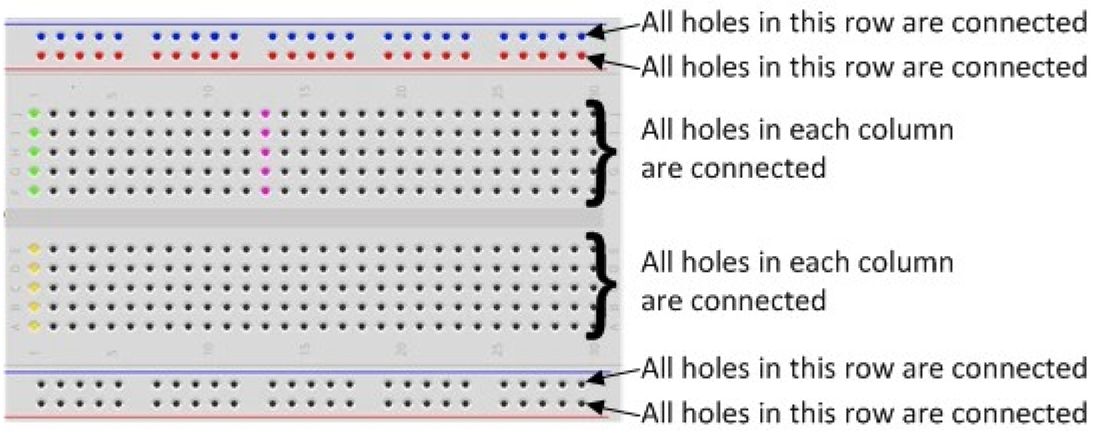
A Breadboard
An LED
A resistor between 50 and 500 ohm. (The bigger the resistor the dimmer the LED)
Two spike and hole jumper wires
Raspberry Pi GPIO

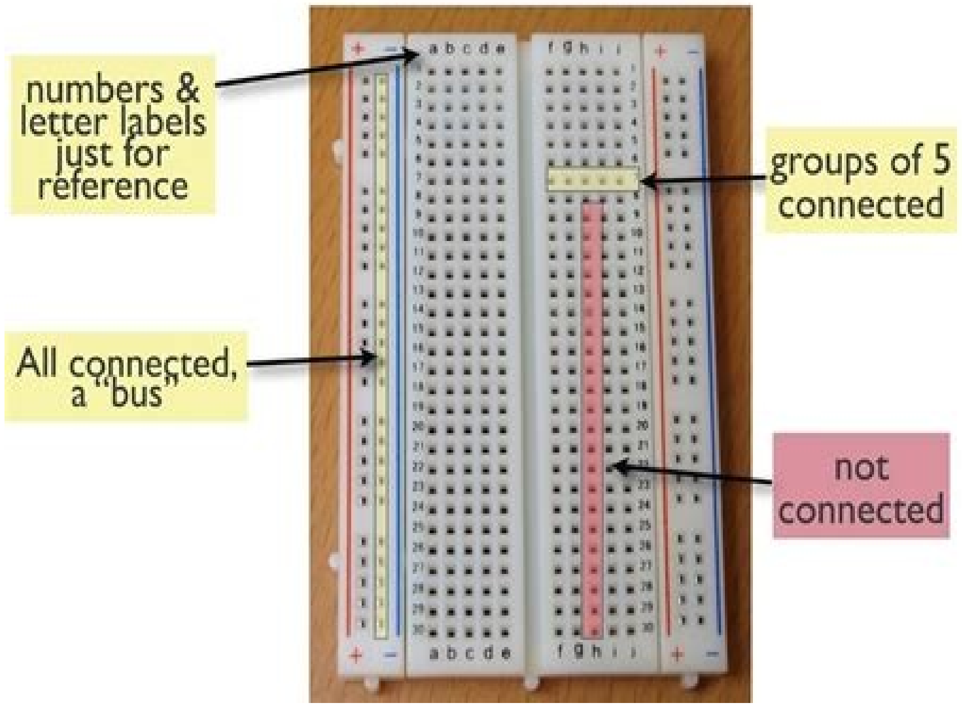


The Breadboard

The breadboard is a way of connecting electronic components to each other without having to solder them together. They are often used to test a circuit design before creating a Printed Circuit Board (PCB).

The holes on the breadboard are connected in a pattern.


All holes in this row are connected
All holes in this row are connected
All holes in each column are connected
All holes in each column are connected
All holes in this row are connected
All holes in this row are connected



The LED

When you pick up the LED, you will notice that one leg is longer than the other. The longer leg (known as the ‘anode’), is always connected to the positive supply of the circuit. The shorter leg (known as the ‘cathode’) is connected to the negative side of the power supply, known as ‘ground’.LED stands for Light Emitting Diode, and glows when electricity is passed through it.

LEDs will only work if power is supplied the correct way round (i.e. if the ‘polarity’ is correct). You will not break the LEDs if you connect them the wrong way round – they will just not light. If you find that they do not light in your circuit, it may be because they have been connected the wrong way round.

You must ALWAYS use resistors to connect LEDs up to the GPIO pins of the Raspberry Pi.

The Resistor

You must ALWAYS use resistors to connect LEDs up to the GPIO pins of the Raspberry Pi. The Raspberry Pi can only supply a small current (about 60mA). The LEDs will want to draw more, and if allowed to they will burn out the Raspberry Pi. Therefore putting the resistors in the circuit will ensure that only this small current will flow and the Raspberry Pi will not be damaged.

Resistors are a way of limiting the amount of electricity going through a circuit; specifically, they limit the amount of ‘current’ that is allowed to flow. The measure of resistance is called the Ohm (Ω), and the larger the resistance, the more it limits the current. The value of a resistor is marked with coloured bands along the length of the resistor body.

You will be using a 330Ω resistor. You can identify the 330Ω resistors by the colour bands along the body. The colour coding will depend on how many bands are on the resistors supplied:

If there are four colour bands, they will be Orange, Orange, Brown, and then Gold.
If there are five bands, then the colours will be Orange, Orange, Black, Black, Brown.

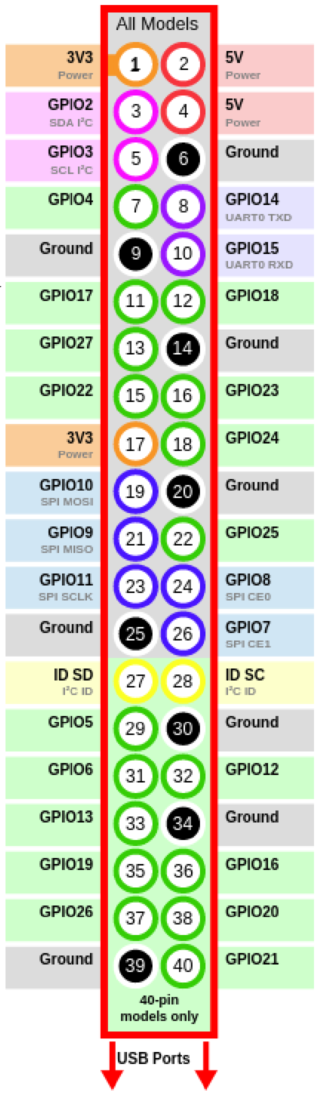
It does not matter which way round you connect the resistors. Current flows in both ways through them.

Jumper Wires

Jumper WiresJumper wires are used on breadboards to ‘jump’ from one connection to another. The ones you will be using in this circuit have different connectors on each end. The end with the ‘pin’ will go into the Breadboard. The end with the piece of plastic with a hole in it will go onto the Raspberry Pi’s GPIO pins.

The Raspberry Pi's GPIO Pins

GPIO stands for General Purpose Input Output. It is a way the Raspberry Pi can control and monitor the outside world by being connected to electronic circuits. The Raspberry Pi is able to control LEDs, turning them on or off, or motors, or many other things. It is also able to detect whether a switch has been pressed, or temperature, or light. In the CamJam EduKit you will learn to control LEDs and a buzzer, and detect when a button has been pressed. The diagram below left shows the pin layout for a Raspberry Pi Models A and B (Rev 2 - the original Rev 1 Pi is slightly different), looking at the Raspberry Pi with the pins in the top right corner. The new 40 pin Raspberry Pi’s shares exactly the same layout of pins for the top 13 rows of GPIO pins.

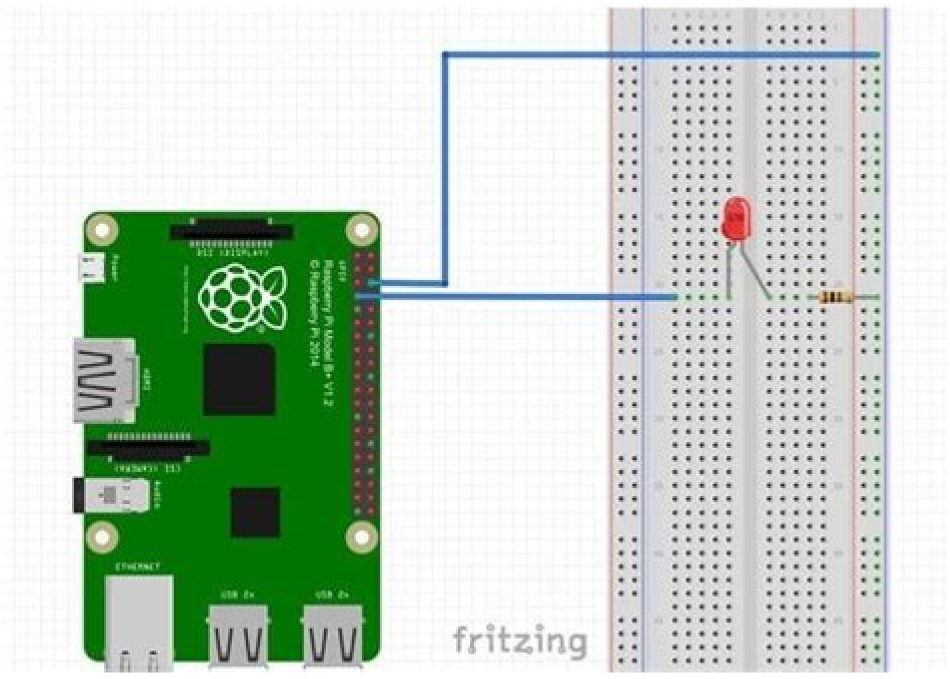


Building the Circuit

The circuit consists of a power supply (the Raspberry Pi), an LED that lights when the power is applied, and a resistor to limit the current that can flow through the circuit.

You will be using one of the ‘ground’ (GND) pins to act like the ‘negative’ or 0 volt ends of a battery. The ‘positive’ end of the battery will be provided by a GPIO pin. Here we will be using pin 4. When they are ‘taken high’, which means it outputs 3.3 volts, the LED will light. Now take a look at the circuit diagram below.

A Single LED



You should turn your Raspberry Pi off for the next bit, just in case you accidentally short something out.

Use one of the jumper wires to connect a ground pin to the rail, marked with blue, on the breadboard. The female end goes on the Raspberry Pi's pin, and the male end goes into a hole on the breadboard.
Then connect the resistor from the same row on the breadboard to a column on the breadboard, as shown above.
Next, push the LEDs legs into the breadboard, with the long leg on the right.
Lastly, complete the circuit by connecting pin 4 to the right hand leg of the LED. This is shown here with the orange wire.

Note: Any gpio pin could be used to connect to the LED positive (anode) leg. If you were to use a diferent gpio pin you would need to use the correct number of that pin in your code.

The Code

You are now ready to write some code to switch the LED on. Turn on your Raspberry Pi and open the terminal window.

Create a new text file "led_expt.py" by typing the following:

```
nano led_expt.py
```

Type in the following code:

```
import gpiozero # importing the whole gpiozero library
import time
led4 = LED(4)  # the led is connected to gpio pin 4
print "LED on"
led4.on() #turns the led on gpio pin 4 on
time.sleep(1)
print "LED off" # turns the led on gpio pin 4 off
```

Once all the code has been entered and checked it, save and exit the text editor with "Ctrl + x" then "y" then "enter".
Running the Code

To run this code type:

```
sudo python3  led_expt.py
```

You will see the LED turn on for a second and then turn off.

If your code does not run and an error is reported, edit the code again using nano LED.py.

Explanation of the code

So, what is happening in the code? Let's go through it a line at a time:

```
import RPi.GPIO as GPIO
```

The first line tells the Python interpreter (the thing that runs the Python code) that it will be using a 'library' that will tell it how to work with the Raspberry Pi's GPIO pins. A 'library' gives a programming language extra commands that can be used to do something different that it previously did not know how to do. This is like adding a new channel to your TV so you can watch something different.

```
import time
```

Imports the Time library so that we can pause the script later on.

```
led4=LED(4)
```

Each pin on the Raspberry Pi has a gpio number you need to tell the program which pin is to be used.

```
print "LED on"
```

This line prints some information to the terminal.

```
led4.on()
```

This turns the GPIO pin ‘on’. What this actually means is that the pin is made to provide power of 3.3volts. This is enough to turn the LED in our circuit on.

```
time.sleep(1)
```

Pauses the Python program for 1 second

```
print "LED off"
```

This line prints some information to the terminal.

```
led4.off()
```

This turns the GPIO pin ‘off’, meaning that the pin is no longer supplying any power.

And that's it! You are now able to turn an LED on and off.