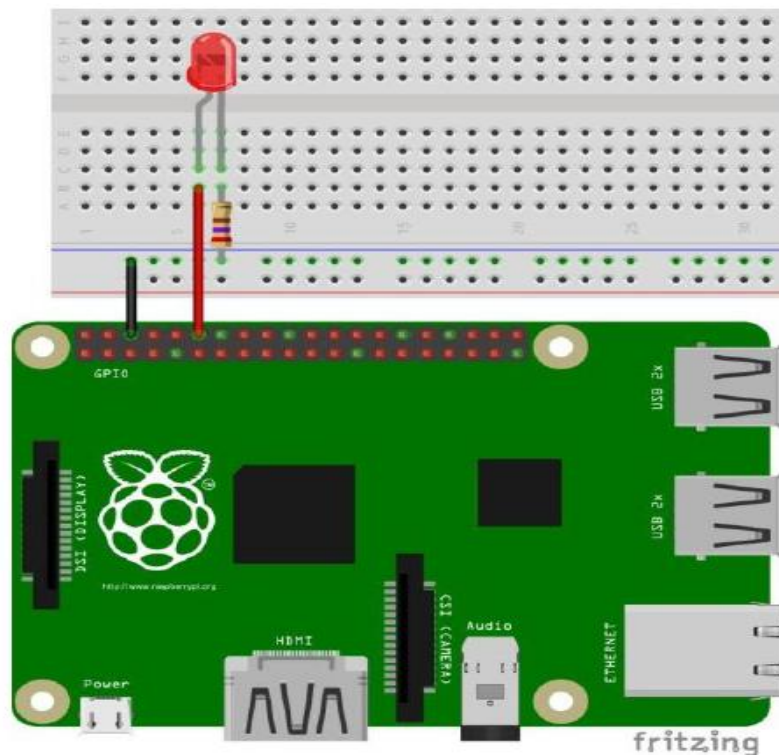


Controlling an LED with Node-RED

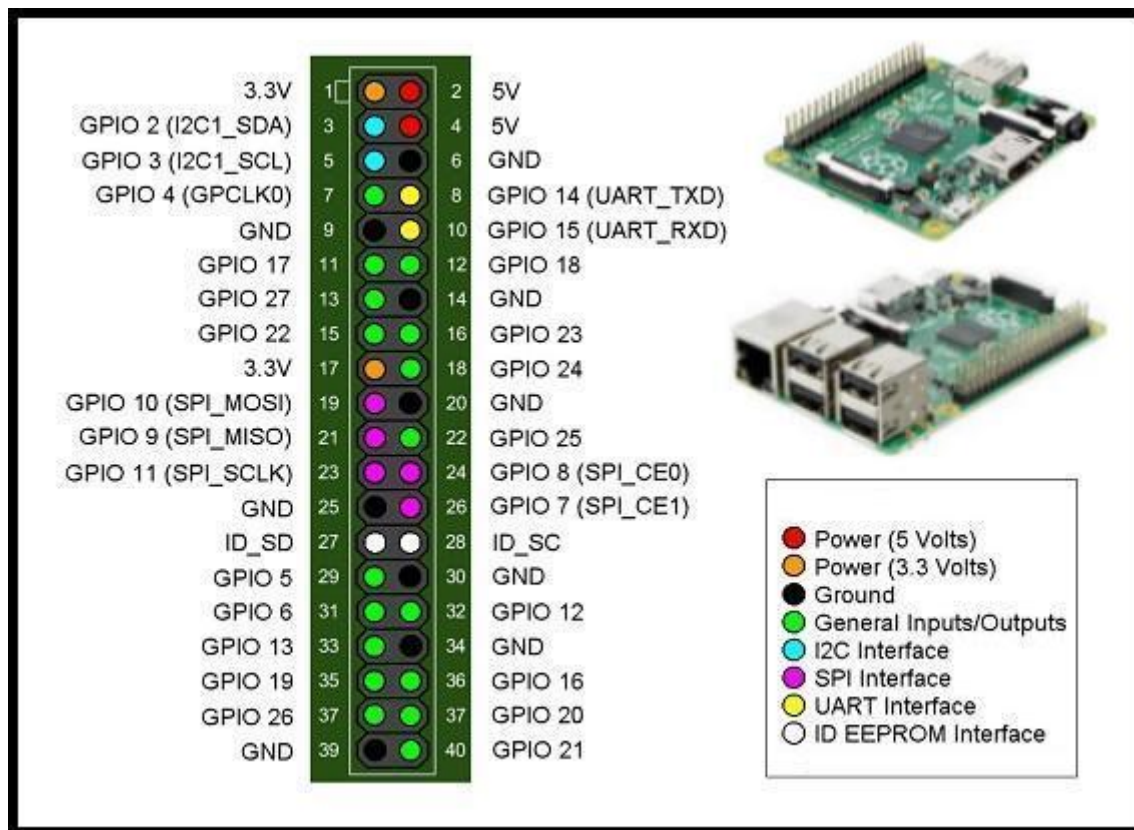
Here's the hardware required to complete this project:

- LED (5mm)
- 270Ω resistor
- Breadboard
- Wire cables



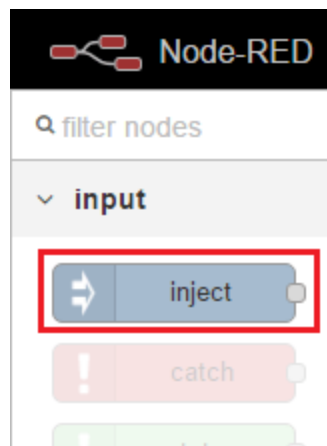
*

If you don't have a pin label, then this guide can help you to identify the pin numbers:

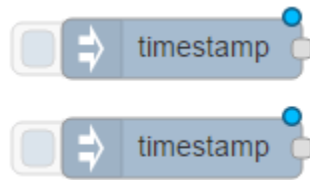


Preparing your flow

Drag two Inject nodes into your flow.



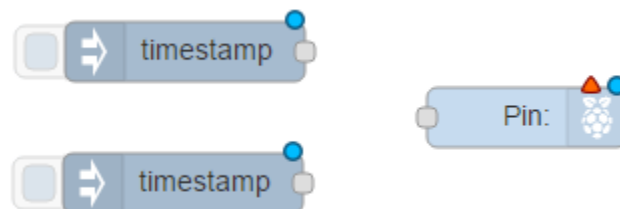
They will look like this in your flow:



Then, add the rpi-gpio out node to your flow



Here's what you should see:



Double-click the first Inject node to edit its properties. Select string and type 1. Press Ok.

Edit inject node

Select string

Type 1

✉ Payload ▼ a_z 1

☰ Topic

🔄 Repeat none ▼

☐ Inject once at start?

📌 Name Name

Note: "interval between times" and "at a specific time" will use cron. See info box for details.

Click Ok Ok Cancel

Edit Inject node number two. Select string and type 0. Press Ok.

Edit inject node

Select string

Type 0

✉ Payload ▼ a_z 0

☰ Topic

🔄 Repeat none ▼

☐ Inject once at start?

📌 Name Name

Note: "interval between times" and "at a specific time" will use cron. See info box for details.

Click Ok Ok Cancel

Lastly, edit the rpi-gpio out node. Set the type as a digital output and the GPIO1 – Pin 12 (that's where your LED is plugged in)

Edit rpi-gpio out node

12 - GPIO1

GPIO Pin: 12 - GPIO1 - BCM18 (Pi 2 Model B)

Type: Digital output

☐ Initialise pin state?

Name: Name

Pins in Use: 12

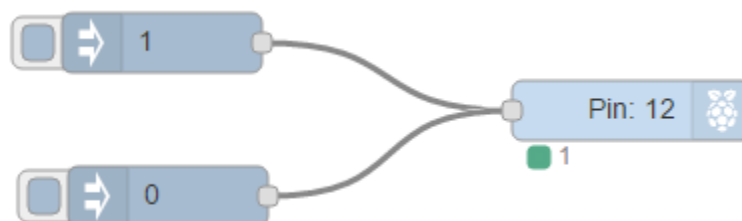
Tip: For digital output - input must be 0 or 1.

Click Ok

Ok Cancel

Deploying the application

After editing each node, your flow should look like this:



Click the Deploy button on the top-right corner to save your application.



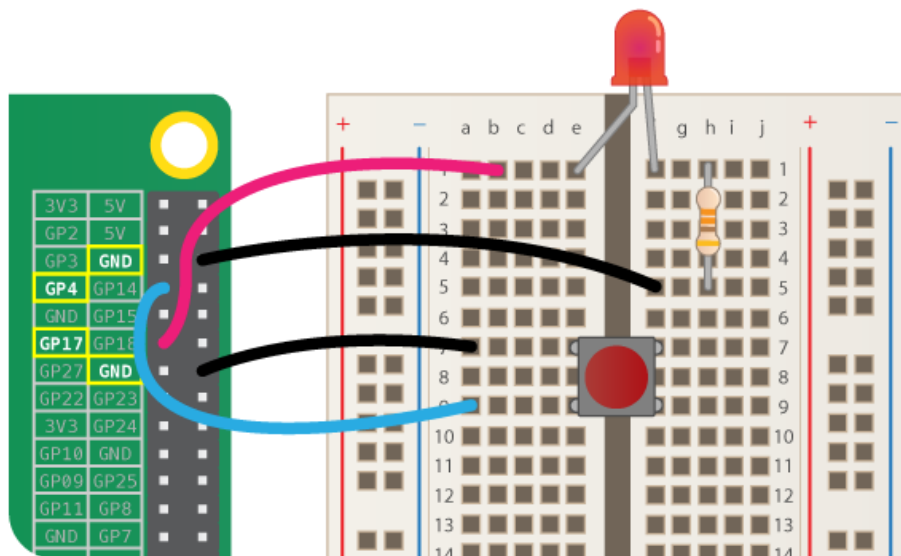
Testing the application

Now, press the square of the Inject 1 node



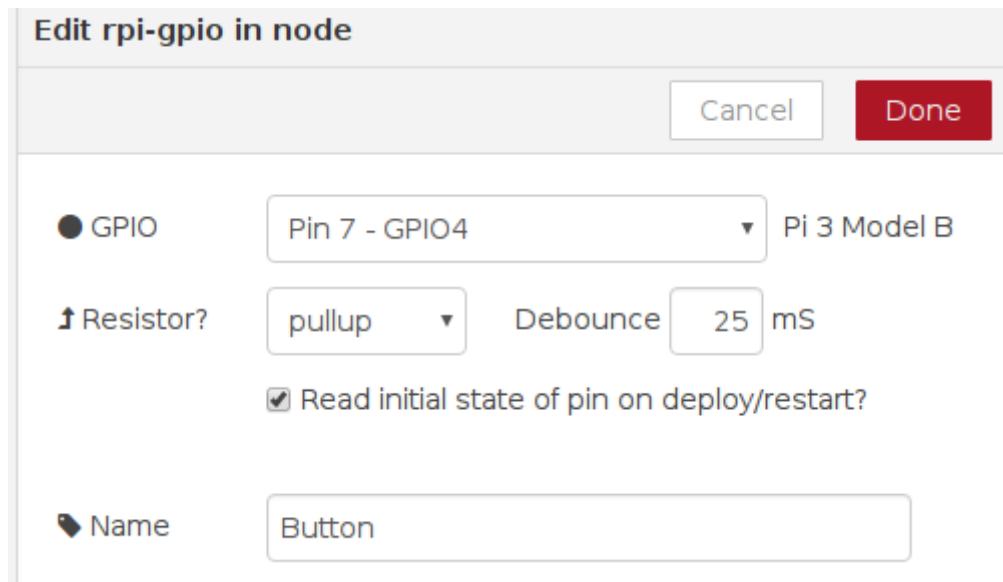
Adding a button

Now let's add a button to control the LED. Wire up your button to the Raspberry Pi as shown in the image below, so that your LED is still connected to GPIO pin 17, and your button is connected to GPIO pin 4:



Remove your Turn on and Turn off inject nodes by clicking the node and pressing Delete on the keyboard. We no longer need these, as we will be controlling the LED using a physical button.

Now we need to add a Raspberry Pi GPIO input node. This is the node with the raspberry icon on the left. Set up this node to receive input from your physical button as follows:



Specifying **pullup** means that GPIO pin 4 will be set **to** HIGH, and pressing the button will cause it to go LOW.

✚ You will notice that the LED is lit to start with, and that pressing the button switches it off. That's not quite right! This is because we are using pullup as explained in the previous step, so the button pin will be HIGH by default. HIGH generates the message **1**, and this turns the LED on. When we press the button, we cause the pin to go LOW, generating a **0** message which turns off the LED. We need to reverse the values. We want the LED to receive the message **0** by default and the message **1** when the button is pressed.

✚ Add a switch node. This can be found in the function section. This node is similar to the **if/elif/else** type constructs you may have seen in Scratch or Python. You can configure it to have multiple output paths (outlined in red on the screenshot) depending on the value passed in. In this case we will set up the node so that if the property `msg.payload` is equal to 1, the first path will be followed. Click the small Add button at the bottom to add a second path, and for this path select otherwise in the drop down. This path will be followed if the input was anything other than 1. Click Done when you are finished.

Edit switch node

Cancel

Done

Name

If input is 1

Property

▼ msg.payload

≡

== ▼

▼ a₂ 1

→ 1 ✕

≡

== ▼

▼ a₂

→ 2 ✕

==

!=

<

<=

>

>=

is between

contains

matches regex

is true

is false

is null

is not null

otherwise

✚ You should see your newly created switch node, with two dots on the right side for outputs. Note that the title **If input is 1** is simply a description of what the node does, and has no effect on its function.

- Join up the GPIO input button node to the input (left side) of the switch node.
- Now drag in a yellow change node from the functions section and double-click on it to configure it. We will use this node to change the message being sent. Remember: when we created the switch node, the first output was set to be followed if the input message was **1**. We will use the change node to change the message to **0**.

8

Edit change node

Cancel
Done

Name

Rules

Set
▼

▼ msg.payload

to

▼ a_z 0

- Press Done, then draw a line from first output of the switch node to the change node. Then connect the output of the change node to the LED node:
- Now add another change node to set the msg.payload to **1**. Connect this node to output 2 of the switch node and then to the LED node. This tells the flow that when the otherwise branch is followed (i.e. the msg.payload is not already **1**), we would like to change it to **1**. When you are ready, deploy your flow, and then push the physical button to confirm that it works properly.

