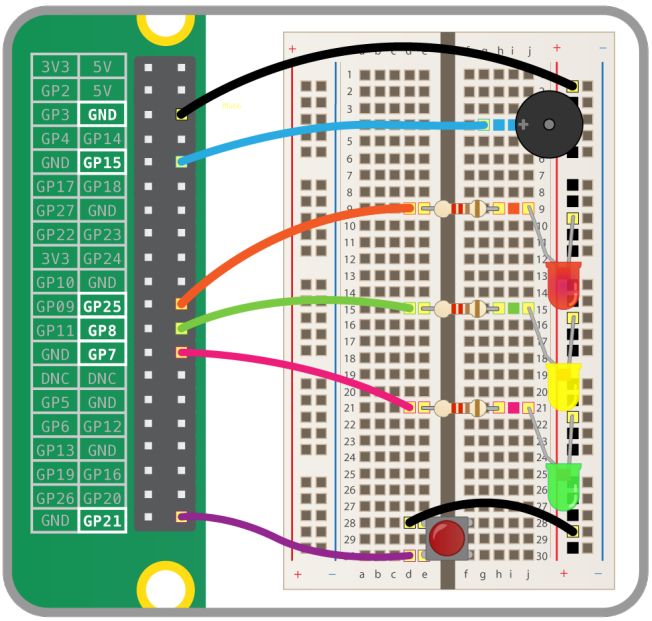
Interactive traffic lights with Python

Connect the LEDs

1 Connect your LEDs to the following pins:

LED	GPIO
Button	21
Red LED	25
Amber LED	28
Green LED	27
Buzzer	15



Control the LEDs and button

1 Open **Mu** from the main menu.

2 Enter the following code:

```
from gpiozero import LED, Button

red = LED(25)
button = Button(21)

while True:
    if button.is_pressed:
        red.on()
    else:
        red.off()
```

3 Save your code and run it with **F5**

4 Remove the while loop and add two more LEDs

```
from gpiozero import LED, Button

red = LED(25)
amber = LED(28)
green = LED(27)

button = Button(21)
```

5 Get them to come on when the button is pressed:

6 Run the code and press the button.

```
while True:
    if button.is_pressed:
        green.on()
        amber.on()
        red.on()
    else:
        green.off()
        amber.off()
        red.off()
```

Traffic lights

You can use the built-in `TrafficLights` class instead of three individual LEDs.

1 Amend the from `gpiozero import...` line to replace `LED` with `TrafficLights`:

```
from gpiozero import TrafficLights, Button
from time import sleep

button = Button(21)
lights = TrafficLights(25, 28, 27)

while True:
    button.wait_for_press()
    lights.on()
    button.wait_for_release()
    lights.off()
```

2 Try changing the lights to `blink`:

```
while True:
    lights.blink()
    button.wait_for_press()
    lights.off()
    button.wait_for_release()
```

Traffic lights sequence

As well as controlling the whole set of lights together, you can also control each LED individually. With traffic light LEDs, a button, and a buzzer, you can create your own traffic lights sequence, complete with pedestrian crossing!

1 Modify your loop to perform an automated sequence of LEDs being lit:

```
while True:
    lights.green.on()
    sleep(1)
    lights.amber.on()
    sleep(1)
    lights.red.on()
    sleep(1)
    lights.off()
```

2 Add a `wait_for_press()` so that pressing the button initiates the sequence:

```
while True:
    button.wait_for_press()
    lights.green.on()
    sleep(1)
    lights.amber.on()
    sleep(1)
    lights.red.on()
    sleep(1)
    lights.off()
```

Try some more sequences of your own.

3 Now try creating the full traffic lights sequence:

- Green on
- Amber on
- Red on
- Red and amber on
- Green on

Be sure to turn the correct lights on and off at the right time, and make sure you use `sleep` to time the sequence perfectly.

4 Try adding the button for a pedestrian crossing. The button should move the lights to red (not immediately), and give the pedestrians time to cross before moving the lights back to green until the button is pressed again.

5 Now try adding a buzzer to beep quickly to indicate that it is safe to cross, for the benefit of visually impaired pedestrians:

```
buzzer = Buzzer(15)

buzzer.on()
buzzer.off()
buzzer.beep(0.1, 0.1)
```

6 Your final interactive traffic lights code should start on a green light and then:

- Wait for the button to be pressed
- When pressed, change to red/amber, then green
- Beep for a while to say it's time to cross
- Go to amber and then green
- Repeat

RASPBERRY JAM