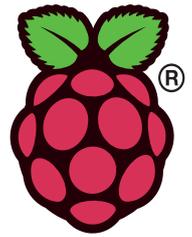


FLASHING LED RECIPE

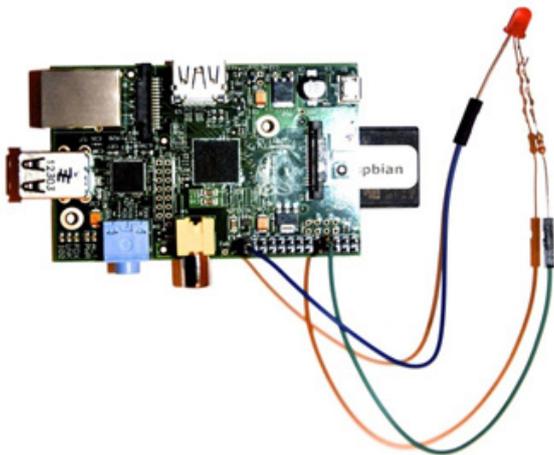
A PHYSICAL COMPUTING PROJECT FOR THE RASPBERRY PI – NO SOLDERING, TOOLS OR INTERNET ACCESS REQUIRED!



This recipe has been developed by the Raspberry Pi Foundation in conjunction with OCR

Difficulty: Very Basic

This recipe will allow you to turn an LED into an output device for your Raspberry Pi and will guide you through writing a program to make it flash.

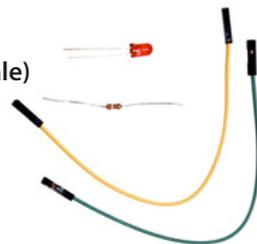


Ingredients needed in addition to your Raspberry Pi:

1x LED

1x 220Ω Resistor

2x Jumper Wire (female to female)



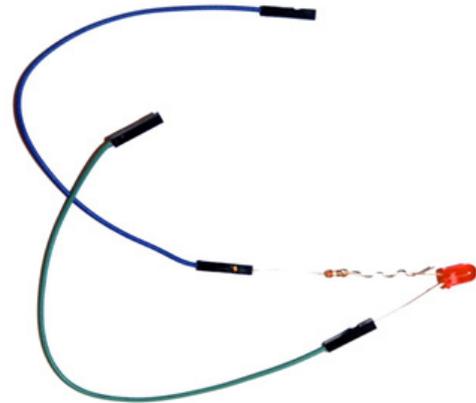
Method:

Turn an LED into an output for your program

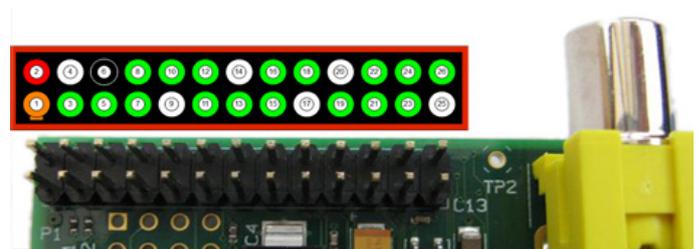
1. Take one end of the resistor and twist it around the cathode of the LED (nearest flat edge and the shorter lead) so that it forms a strong connection.



2. Push both the anode (longer lead) of the LED and the other end of the resistor into each of the jumper wires.



3. Take the other end of the jumper lead connected to the cathode of the LED (flat edge, shorter wire) and push onto pin 25 of the GPIO header which is connected to ground.
4. Take the end of the other jumper lead and push into onto pin 7 of the General Purpose Input-Output (GPIO) header which is connected to one of the GPIO channels.

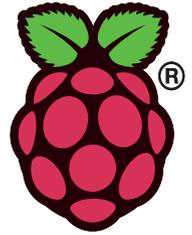


Raspberry Pi GPIO header pins. The diagram above the pins shows the pin numbers. You will be using pin 7 and pin 25. **Warning! You can damage your Raspberry Pi if you do not use the GPIO pins correctly!**

Congratulations! You have now attached an LED to your Raspberry Pi which can be used as an output in your programs.

... continued

WRITE A PROGRAM THAT LIGHTS UP THE LED



1. Open a command line text editor

```
nano FlashLED.py
```

2. Type in the code below (Pro Tip: Any lines beginning with a # symbol are comments so don't need to be included for the program to work - they will, however, help you to understand the code)

```
# First we need to import the libraries that
# we need

# Import the time library so that we can make
# the program pause for a fixed amount of time
import time

# Import the Raspberry Pi GPIO libraries that
# allow us to connect the Raspberry Pi to
# other physical devices via the General
# Purpose Input-Output (GPIO) pins
import RPi.GPIO as GPIO

# Now we need to set-up the General Purpose
# Input-Output (GPIO) pins

# Clear the current set-up so that we can
# start from scratch
GPIO.cleanup()

# Set up the GPIO library to use Raspberry Pi
# board pin numbers
GPIO.setmode(GPIO.BOARD)

# Set Pin 7 on the GPIO header to act as
# an output
GPIO.setup(7,GPIO.OUT)

# This loop runs forever and flashes the LED
while True:

    # Turn on the LED
    GPIO.output(7,GPIO.HIGH)

    # Wait for a second
    time.sleep(1)
```

- ```
Turn off the LED
```

```
GPIO.output(7,GPIO.LOW)
```

```
Wait for a second
time.sleep(1)
```

3. Run the program

```
sudo python FlashLED.py
```

**Congratulations!** Now when you run the program the LED will flash.

## MAKE IT YOUR OWN!

There are many simple ways that you could create your own variations of this recipe, some examples of which are listed below:

- Change the rate at which the LED flashes
- Require a user input to turn on the LED
- Connect multiple LEDs to the Pi