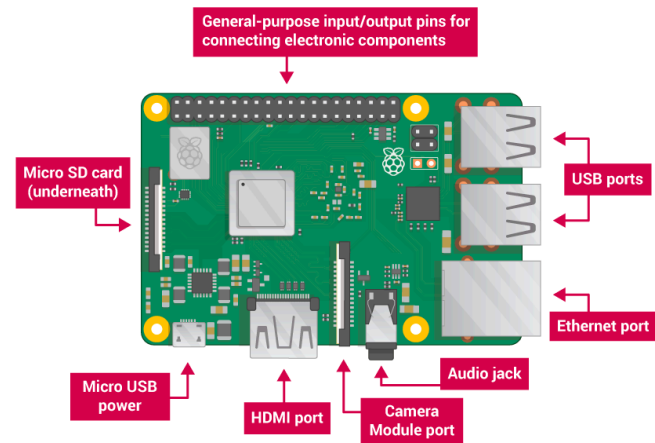




Set-up:

1. Install Pi into the case
2. Insert Micro SD into SD card holder on underside of board. (Gold teeth of SD card should face towards the board).
3. Connect keyboard, mouse and HDMI cable.
4. Connect power cable. Pi should load up.
5. Select Raspbian and click install
6. Connect ethernet cable.



Installing Modules:

Some modules will be useful when using the RPi. Additionally, changing some settings now, will be useful later. Type the commands into the terminal to make each change.

To change default python from version 2, to version 3 type the following commands:

```
cd usr/bin
```

```
sudo rm python
```

```
sudo ln -s python3 python
```

I²C is a very useful protocol for interfacing with the hardware, it is disabled by default, to enable it:

```
sudo raspi-config
```

Select option "5 Interfacing options" -> "P5 I2C" -> "Yes" -> "Finish"

Similarly, System Management Bus module is a useful module for interfacing, to install the module:

```
sudo apt-get install python-smbus
```

Terminal cheat sheet:

```
ls
```

Lists all the items in current directory.

```
pwd
```

Print working directory.

```
python filename.py
```

Runs the file with that name in current directory using python.

```
geany filename.py
```

Opens code into a text editor.

```
cd (location)
```

Changes the current directory of the terminal, e.g. 'cd Desktop' moves you to the Desktop.

```
sudo
```

Superuser do – completes command at the root level.



Pressing UP key will bring up previous commands.



Finishes what you are writing.

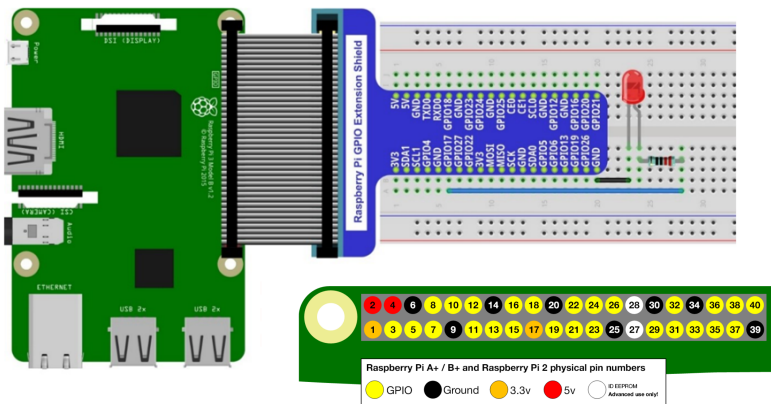
For the full instructions go to:

https://github.com/Freenove/Freenove_Super_Starter_Kit_for_Raspberry_Pi/blob/master/Tutorial.pdf



Hardware:

This program will take regular words as input, and output the words as Morse code by flashing the LED. The circuit is very simple, connect the GPIO extension shield using the grey cables, make sure it is connected in the orientation shown. The extension shield is not necessary but makes the circuits easier to understand as they get larger. Connect the LED's longest leg to a 220Ω resistor, this ensures it is at the correct voltage, this resistor should be connected to the pin labelled GPIO17. The short leg is connected to ground.



Software:

Create a new python file, you can use geany – a preinstalled text editor, or any other text editor.

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

```
ledPin = 11
```

```
def setup():
    GPIO.setmode(GPIO.BOARD)
    GPIO.setup(ledPin, GPIO.OUT)
    GPIO.output(ledPin, GPIO.LOW)
```

```
GPIO.output(ledPin, GPIO.HIGH)
time.sleep(1)
```

```
GPIO.output(ledPin, GPIO.LOW)
GPIO.cleanup()
```

You need the GPIO module. GPIO stands for general purpose input/output and this is how python can send outputs to the LED. The time module allows you to add pauses – useful for Morse code.

Set the LED pin to number 11.

A setup function is useful. The 3rd line sets the led pin to be used as an output. The 4th line sets the initial output to LOW, in this case it corresponds to LED off. 'Setmode' changes the numbering of the outputs. Instead of the confusing GPIO numbers, the pins are numbered in order (as above). You may have guessed that pin number 11 corresponds to GPIO17.

To turn the LED on, set the GPIO output to HIGH. And use the line `time.sleep()` with the number of seconds you want the program to pause in the brackets. This way you can create a series of flashes.

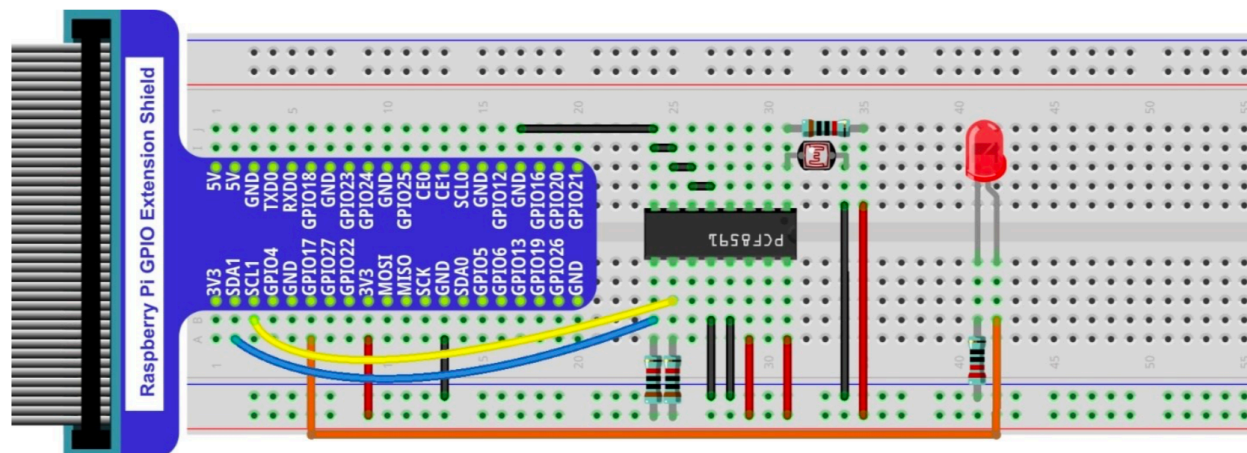
At the end of your program, run this code. It will turn off the LED to make sure it does not stay on, as well as releasing any GPIO resources. If you don't do this you may get errors when you try to reload the program.


To download a sample finished code, type the following command into the terminal. This will create a new folder with the python files.

```
git clone https://github.com/BenKinchin/rpi_code.git
```



Hardware:



This project requires a few more components. This component:  is a Light Dependent resistor. It decreases in resistance when light is shone upon it. The integrated circuit in the centre is PCF8591. It can be used for analogue to digital conversion. In this circuit it is being used to read the voltage from the LDR. The LED is not required in this circuit but is useful for testing. It is connected with a 220Ω resistor (the others are 10kΩ).

Software:

```
import smbus
```

For this project you will need to import the smbus module.

```
address = 0x48
```

```
bus=smbus.SMBus(1)
```

```
cmd=0x40
```

Define the following variables at the beginning, these are needed to use the PCF8591 to read the voltages.

The function analogRead will take the value from the chip. This is not the true voltage (although that can be calculated, it is a relative scale. Now calling the function will return the voltage.

```
def analogRead(chn):
    """Reads volage of LDR"""
    value = bus.read_byte_data(address,cmd+chn)
    return value
```

To call the function just use analogRead(0).

```
analogRead(0)
```

```
def turn_off():
    """Closes bus and cleans GPIO data"""
    bus.close()
    GPIO.cleanup()
    sys.exit()
```

The function time.time() returns the time. This will be useful for timing things. Like length of Morse code flashes?

```
time.time()
```

For your turn off function, add 'bus.close()'.