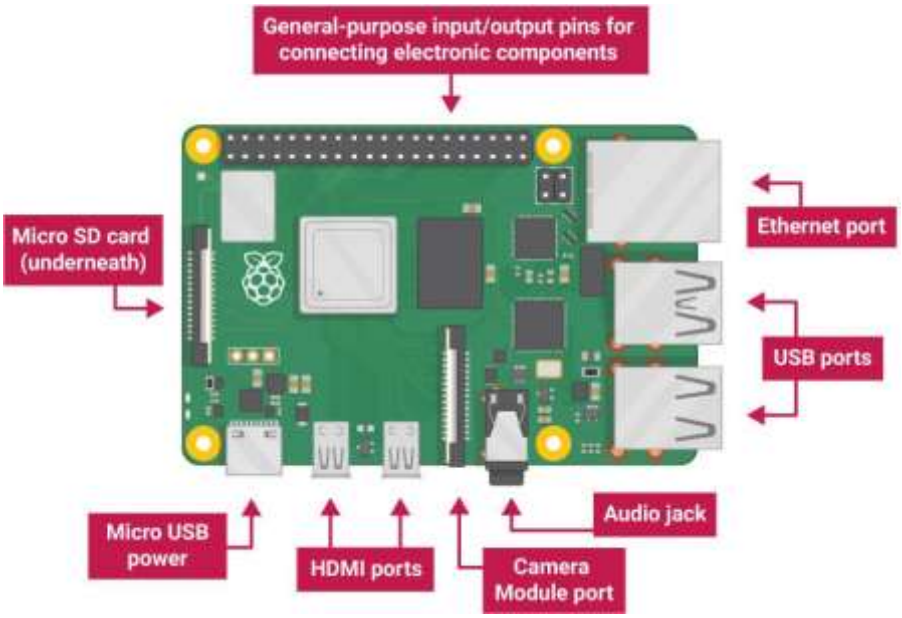
Lab Assignment - 4

Raspberry Pi

Course Outcome: CO4: Interface I/O devices and communication modules.

Raspberry Pi



- USB ports — these are used to connect a mouse and keyboard. You can also connect other components, such as a USB drive.
- SD card slot — you can slot the SD card in here. This is where the operating system software and your files are stored.
- Ethernet port — this is used to connect Raspberry Pi to a network with a cable. Raspberry Pi can also connect to a network via wireless LAN.
- Audio jack — you can connect headphones or speakers here.
- HDMI port — this is where you connect the monitor (or projector) that you are using to display the output from the Raspberry Pi. If your monitor has speakers, you can also use them to hear sound.
- Micro USB power connector — this is where you connect a power supply. You should always do this last, after you have connected all your other components.
- GPIO ports — these allow you to connect electronic components such as LEDs and buttons to Raspberry Pi.

Set up your SD card

If you have an SD card that doesn't have the Raspberry Pi OS operating system on it yet, or if you want to reset your Raspberry Pi, you can easily install Raspberry Pi OS yourself. To do so,

you need a computer that has an SD card port — most laptop and desktop computers have one.

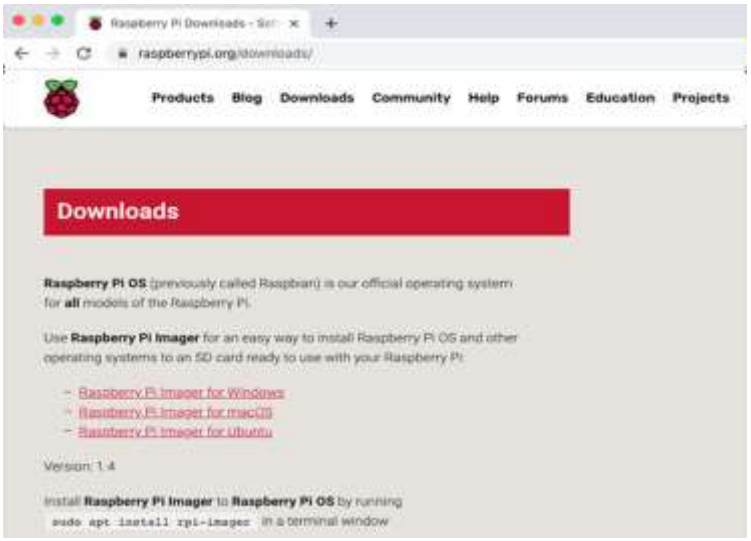
The Raspberry Pi OS operating system via the Raspberry Pi Imager

Using the Raspberry Pi Imager is the easiest way to install Raspberry Pi OS on your SD card.

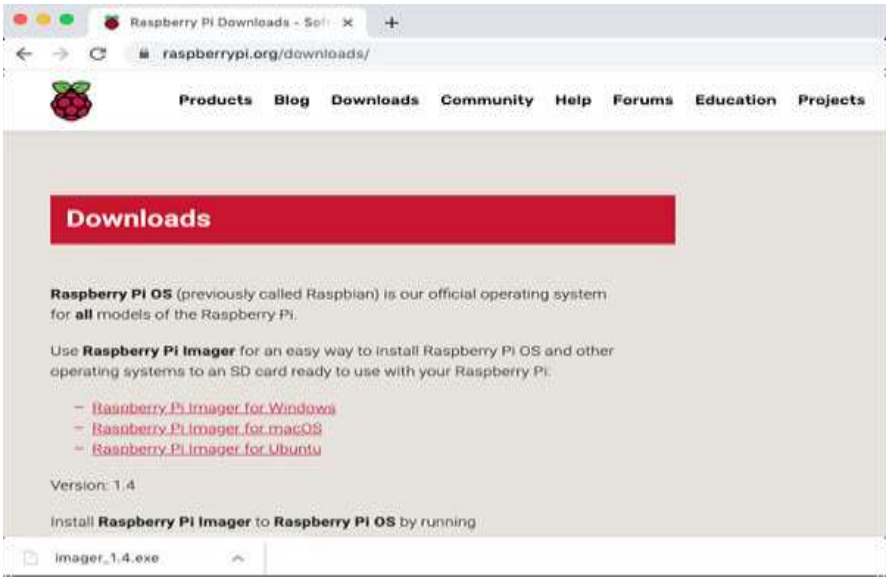
Note: More advanced users looking to install a particular operating system should use this guide to installing operating system images.

Download and launch the Raspberry Pi Imager

- Visit the Raspberry Pi downloads page



- Click on the link for the Raspberry Pi Imager that matches your operating system
- When the download finishes, click it to launch the installer



Using the Raspberry Pi Imager

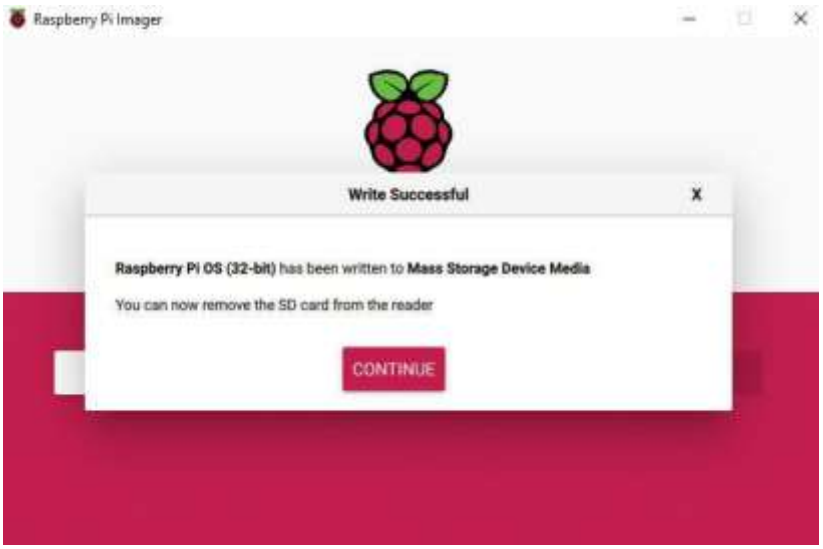
Anything that's stored on the SD card will be overwritten during formatting. If your SD card currently has any files on it, e.g. from an older version of Raspberry Pi OS, you may wish to back up these files first to prevent you from permanently losing them.

When you launch the installer, your operating system may try to block you from running it. For example, on Windows I receive the following message:

- If this pops up, click on More info and then Run anyway
- Follow the instructions to install and run the Raspberry Pi Imager
- Insert your SD card into the computer or laptop SD card slot
- In the Raspberry Pi Imager, select the OS that you want to install and the SD card you would like to install it on

Note: You will need to be connected to the internet the first time for the Raspberry Pi Imager to download the OS that you choose. That OS will then be stored for future offline use. Being online for later uses means that the Raspberry Pi imager will always give you the latest version.

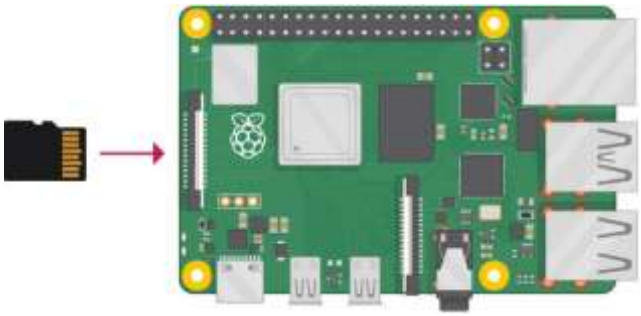
- Then simply click the WRITE button
- Wait for the Raspberry Pi Imager to finish writing
- Once you get the following message, you can eject your SD card



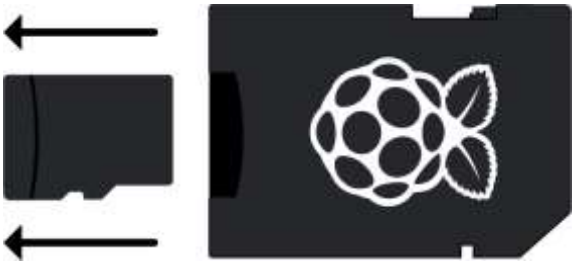
Connect your Raspberry Pi

Let's connect up your Raspberry Pi and get it running.

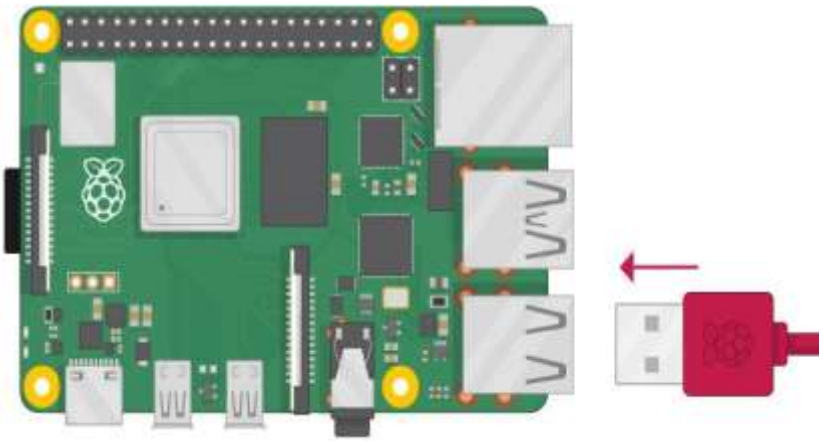
- Check the slot on the underside of your Raspberry Pi to see whether an SD card is inside. If no SD card is there, then insert an SD card with Raspbian installed (via NOOBS).



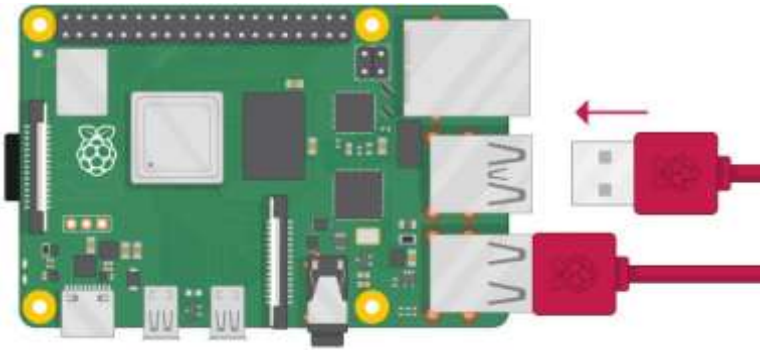
Note: Many microSD cards come inside a larger adapter — you can slide the smaller card out using the lip at the bottom.



- Find the USB connector end of your mouse's cable, and connect the mouse to a USB port on your Raspberry Pi (it doesn't matter which port you use).



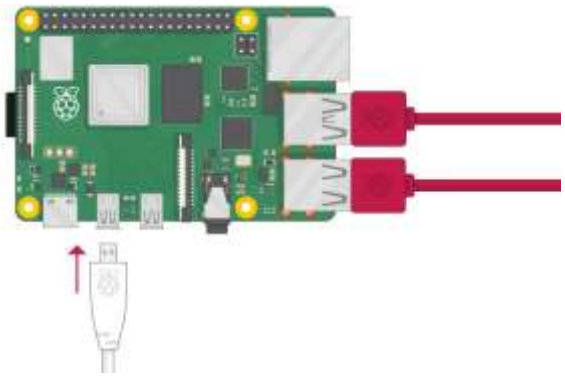
- Connect the keyboard in the same way.



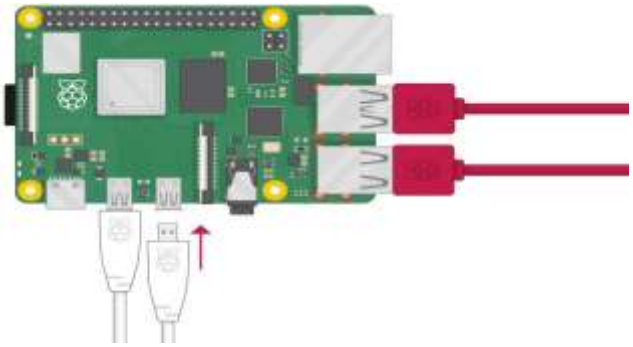
- Make sure your screen is plugged into a wall socket and switched on.
- Look at the HDMI port(s) on your Raspberry Pi — notice that they have a flat side on top.
- Use a cable to connect the screen to the Raspberry Pi's HDMI port — use an adapter if necessary.

Raspberry Pi 4

Connect your screen to the first of Raspberry Pi 4's HDMI ports, labelled HDMI0.



You could connect an optional second screen in the same way.

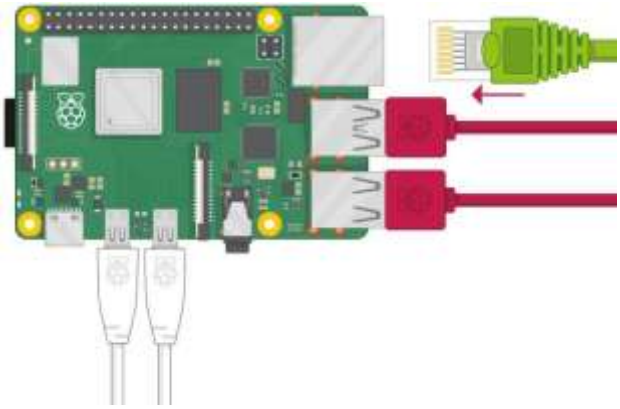


Raspberry Pi 1, 2, 3

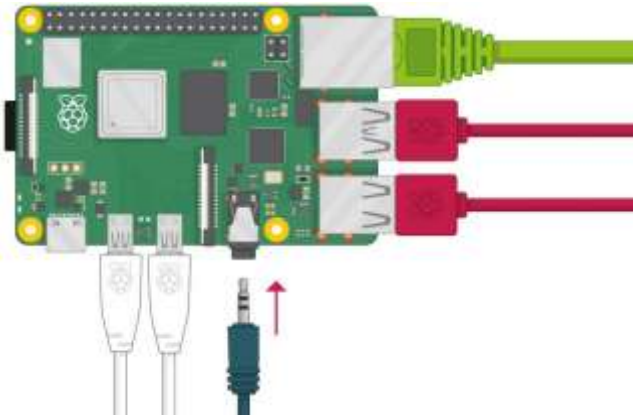
Connect your screen to the single HDMI port.

Note: nothing will display on the screen, because the Raspberry Pi is not running yet.

- If you want to connect the Pi to the internet via Ethernet, use an Ethernet cable to connect the Ethernet port on the Raspberry Pi to an Ethernet socket on the wall or on your internet router. You don't need to do this if you want to use wireless connectivity, or if you don't want to connect to the internet.



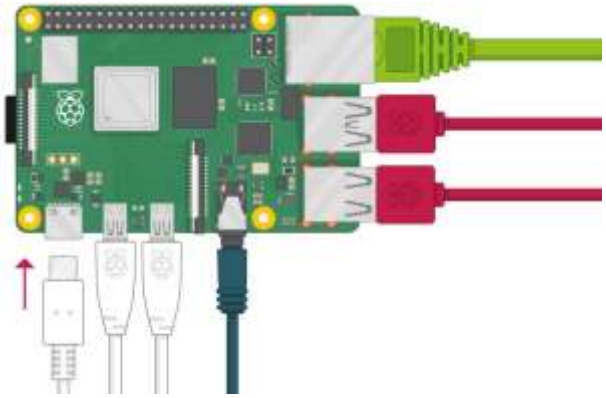
- If your screen has speakers, your Raspberry Pi can play sound through these. Or you could connect headphones or speakers to the audio port.



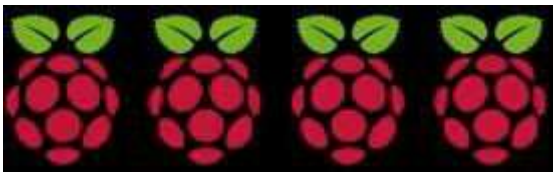
Start up your Raspberry Pi

Your Raspberry Pi doesn't have a power switch. As soon as you connect it to a power outlet, it will turn on.

- Plug the power supply into a socket and connect it to your Raspberry Pi's power port.



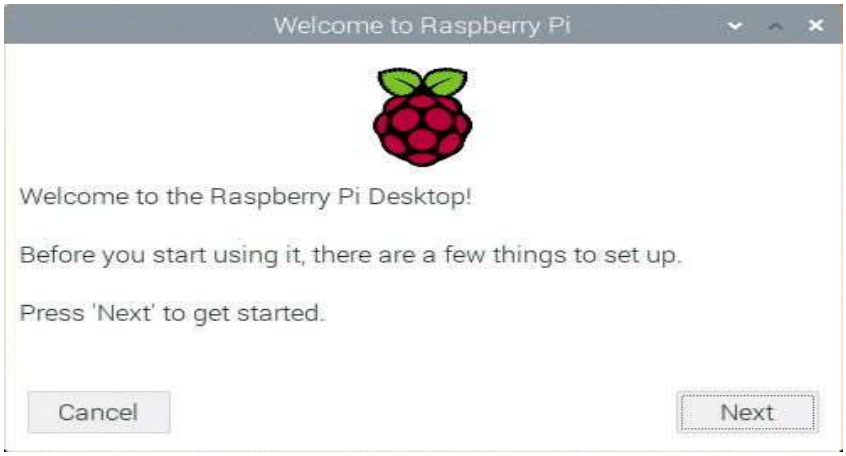
You should see a red LED light up on the Raspberry Pi, which indicates that Raspberry Pi is connected to power. As it starts up (this is also called booting), you will see raspberries appear in the top left-hand corner of your screen.



After a few seconds the Raspberry Pi OS desktop will appear.

Finish the setup

When you start your Raspberry Pi for the first time, the Welcome to Raspberry Pi application will pop up and guide you through the initial setup.



- Click Next to start the setup.
- Set your Country, Language, and Timezone, then click Next again.

- Enter a new password for your Raspberry Pi and click Next.

- Connect to your WiFi network by selecting its name, entering the password, and clicking Next.

Note: if your Raspberry Pi model doesn't have wireless connectivity, you won't see this screen.

- Click Next let the wizard check for updates to Raspbian and install them (this might take a little while).
- Click Done or Reboot to finish the setup.

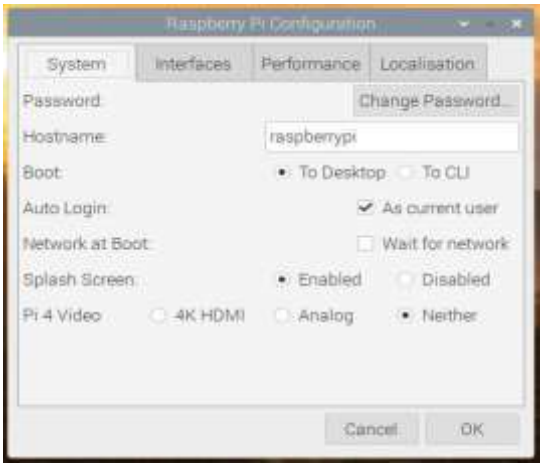
Note: you will only need to reboot if that's necessary to complete an update.

Configuring your Raspberry Pi

You can control most of your Raspberry Pi's settings, such as the password, through the Raspberry Pi Configuration application found in Preferences on the menu.

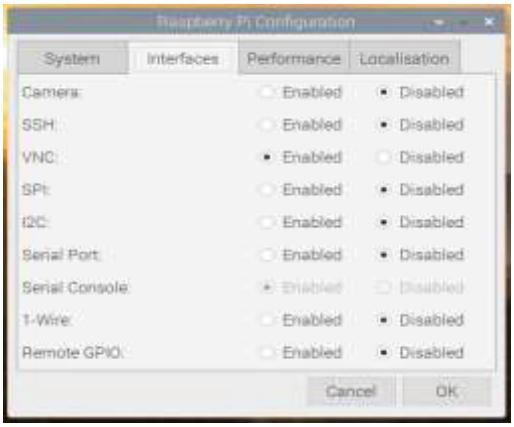
- System

In this tab you can change basic system settings of your Raspberry Pi.



- Password — set the password of the pi user (it is a good idea to change the password from the factory default 'raspberry')
- Boot — select to show the Desktop or CLI (command line interface) when your Raspberry Pi starts
- Auto Login — enabling this option will make the Raspberry Pi automatically log in whenever it starts
- Network at Boot — selecting this option will cause your Raspberry Pi to wait until a network connection is available before starting
- Splash Screen — choose whether or not to show the splash (startup) screen when your Raspberry Pi boots
- Interfaces

You can link devices and components to your Raspberry Pi using a lot of different types of connections. The Interfaces tab is where you turn these different connections on or off, so that your Raspberry Pi recognises that you've linked something to it via a particular type of connection.



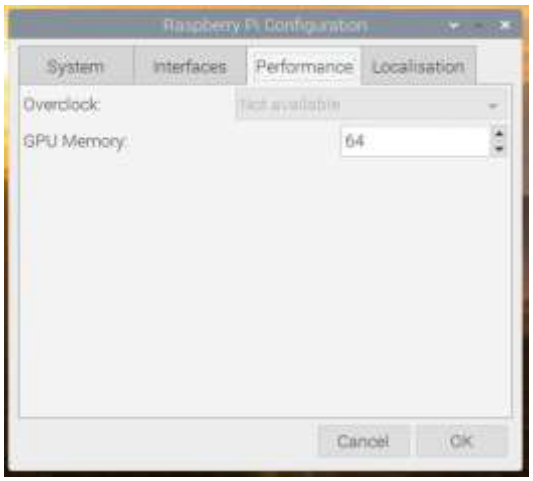
- Camera — enable the Raspberry Pi Camera Module
- SSH — allow remote access to your Raspberry Pi from another computer using SSH
- VNC — allow remote access to the Raspberry Pi Desktop from another computer using VNC

- SPI — enable the SPI GPIO pins
- I2C — enable the I2C GPIO pins
- Serial — enable the Serial (Rx, Tx) GPIO pins
- 1-Wire — enable the 1-Wire GPIO pin
- Remote GPIO — allow access to your Raspberry Pi's GPIO pins from another computer

- Performance

If you need to do so for a particular project you want to work on, you can change the performance settings of your Raspberry Pi in this tab.

Warning: Changing your Raspberry Pi's performance settings may result in it behaving erratically or not working.



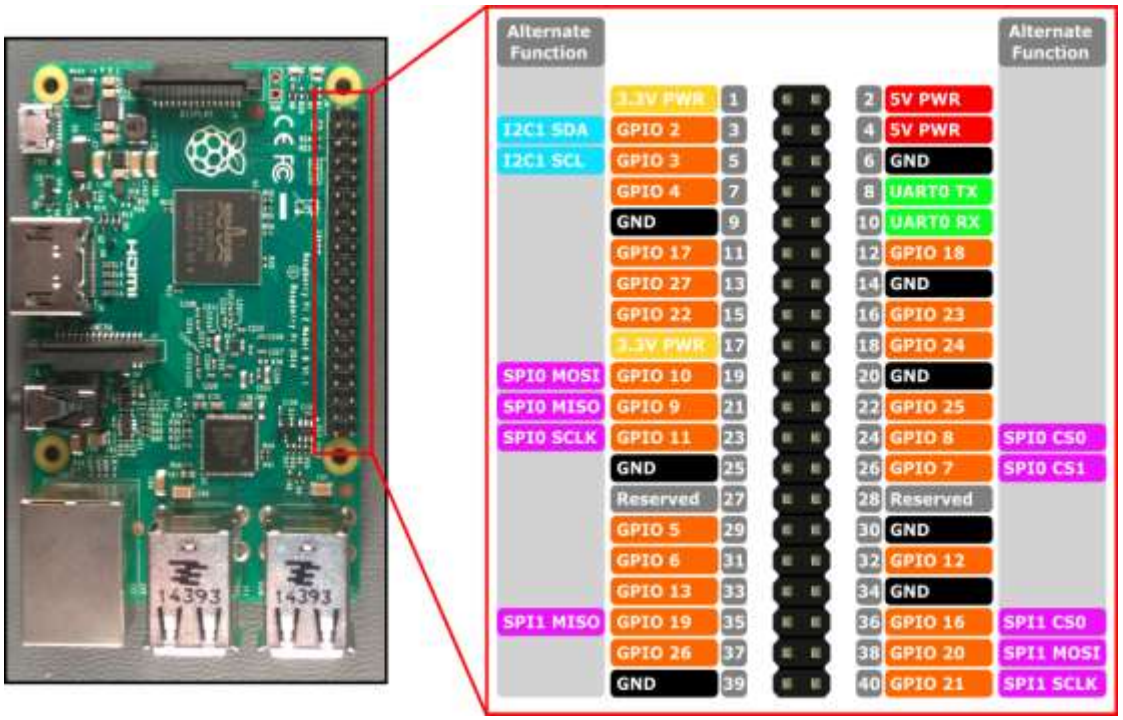
- Overclock — change the CPU speed and voltage to increase performance
- **GPU Memory** — change the allocation of memory given to the GPU

- Localisation

This tab allows you to change your Raspberry Pi settings to be specific to a country or location.

- Locale — set the language, country, and character set used by your Raspberry Pi
- Timezone — set the time zone
- Keyboard — change your keyboard layout
- WiFi Country — set the WiFi country code

GPIO Pins :



There are 40output pins for the PI. Not all 40 pin out can be programmed to our use. There are only 26 GPIO pins which can be programmed. These pins go from **GPIO2 to GPIO27**. These **26 GPIO pins can be programmed** as per need. With special GPIO put aside, we have 17 GPIO remaining

Each of these 17 GPIO pins can deliver a maximum of **15mA current**. And the sum of currents from all GPIO cannot exceed 50mA. So we can draw a maximum of 3mA in average from each of these GPIO pins. So one should not tamper with these things unless you know what you are doing.

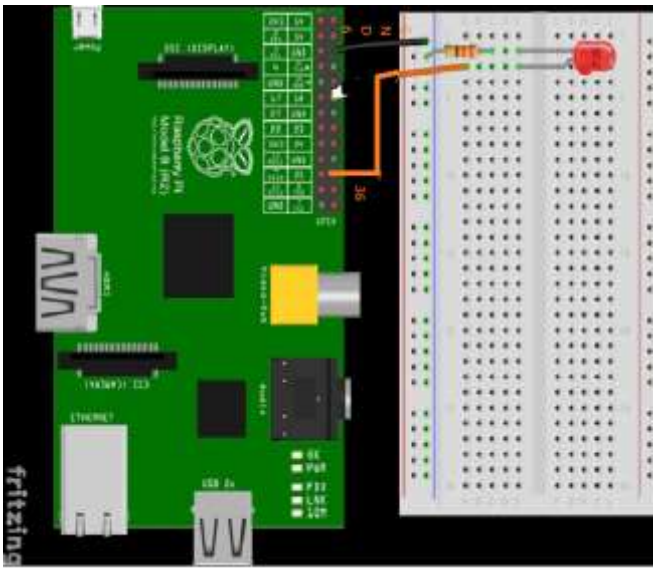
Exercise

1. Interfacing pi with LED

Components Required

- Raspberry Pi 3
- Connecting pins
- 220Ω or 1KΩresistor
- LED
- Bread Board

Circuit Diagram:



Program:

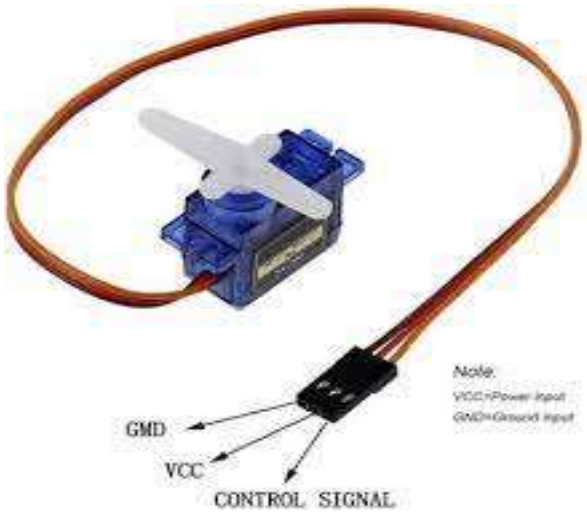
a. Infinite loop (while loop) blinking an led with a delay of 1 second.

```
import RPi.GPIO as GPIO
import time
ledpin = 36  # Define the GPIO pin number (BOARD numbering)
# Suppress warnings
GPIO.setwarnings(False)
# Set up the GPIO mode
GPIO.setmode(GPIO.BOARD)
# Set up the GPIO pin as an output
GPIO.setup(ledpin, GPIO.OUT)
# Blink the LED in an infinite loop
try:
    while True:
        GPIO.output(ledpin, True)   # Turn on LED
        time.sleep(1)               # Wait for 1 second
        GPIO.output(ledpin, False)  # Turn off LED
        time.sleep(1)               # Wait for 1 second
except KeyboardInterrupt:
    # Clean up GPIO settings before exiting
    GPIO.cleanup()
```

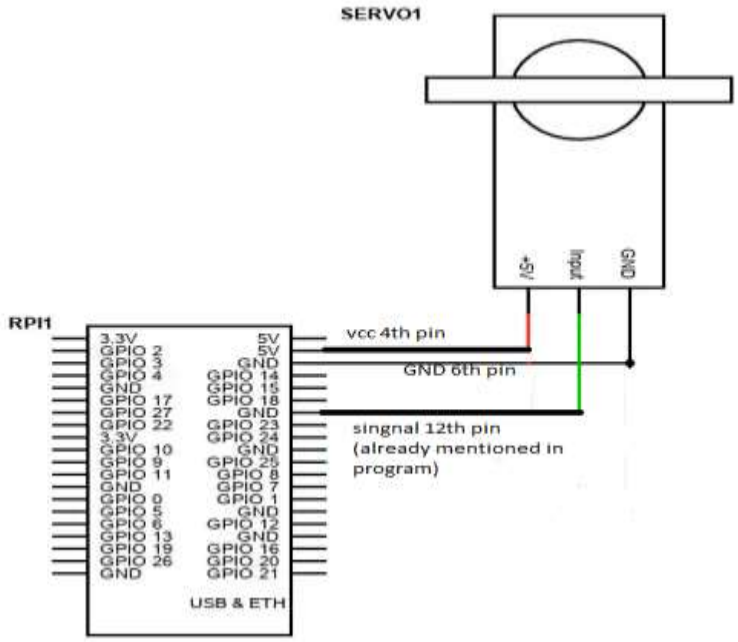
b. Blink LED 10 times(using for loop).

```
import RPi.GPIO as GPIO
import time

ledpin = 40  # Define the GPIO pin number (BOARD numbering)

# Suppress warnings
GPIO.setwarnings(False)

# Set up the GPIO mode
GPIO.setmode(GPIO.BOARD)

# Set up the GPIO pin as an output
GPIO.setup(ledpin, GPIO.OUT)

# Blink the LED on and off 10 times
for i in range(10):
   GPIO.output(ledpin, True)  # Turn on LED
   time.sleep(1)              # Wait for 1 second
   GPIO.output(ledpin, False) # Turn off LED
   time.sleep(1)              # Wait for 1 second

# Clean up GPIO settings
GPIO.cleanup()
```

2. Interfacing servo motor with pi



RED: VCC(PINS 1,2,4),BROWN: GND(6,9,34,39,30,14,20) YELLOW: 12



Program:

```
import RPi.GPIO as GPIO
import time
motor = 12  # Define the GPIO pin number (BOARD numbering)
# Suppress warnings
GPIO.setwarnings(False)
# Set up the GPIO mode
GPIO.setmode(GPIO.BOARD)
# Set up the GPIO pin as an output
GPIO.setup(motor, GPIO.OUT)
# Set up PWM on the motor pin at 50Hz
servo = GPIO.PWM(motor, 50)
# Start PWM with a duty cycle of 0
servo.start(0)
try:
    while True:
      for i in range(0, 101, 5):  # Change range to 0-100 inclusive
        servo.ChangeDutyCycle(i)
        time.sleep(0.2)
except KeyboardInterrupt:
   print("Motor stopped")
   servo.stop()  # Correct this line
   GPIO.cleanup()
```

b. CONTROL SERVO USING PI BY ENTERING ANGLE BY USERS

Program:

```
# Import libraries
import RPi.GPIO as GPIO
import time

# Set GPIO numbering mode
GPIO.setmode(GPIO.BOARD)

# Set pin 11 as an output, and define as servo1 as PWM pin
GPIO.setup(11,GPIO.OUT)
servo1 = GPIO.PWM(11,50) # pin 11 for servo1, pulse 50Hz

# Start PWM running, with value of 0 (pulse off)
servo1.start(0)

# Loop to allow user to set servo angle. Try/finally allows exit
# with execution of servo.stop and GPIO cleanup

try:
while True:
#Ask user for angle and turn servo to it
angle = float(input('Enter angle between 0 & 180: '))
servo1.ChangeDutyCycle(2+(angle/18))
time.sleep(0.5)
servo1.ChangeDutyCycle(0)

finally:
#Clean things up at the end
servo1.stop()
GPIO.cleanup()
print("Goodbye!")
```

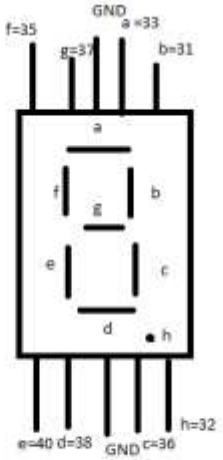
c. ROTATE SERVO FROM 0 TO 180 DEGREE

Program:

```
# Import libraries
import RPi.GPIO as GPIO
import time

# Set GPIO numbering mode
GPIO.setmode(GPIO.BOARD)

# Set pin 11 as an output, and set servo1 as pin 11 as PWM
GPIO.setup(11, GPIO.OUT)
servo1 = GPIO.PWM(11, 50)  # Note 11 is pin, 50 = 50Hz pulse

# Start PWM running, but with a value of 0 (pulse off)
servo1.start(0)
print("Waiting for 2 seconds")
time.sleep(2)
```

```
# Let's move the servo!
print("Rotating 180 degrees in 10 steps")

# Define variable duty
duty = 2

# Loop for duty values from 2 to 12 (0 to 180 degrees)
while duty <= 12:
    servo1.ChangeDutyCycle(duty)
    time.sleep(0.3)
    servo1.ChangeDutyCycle(0)
    time.sleep(0.7)
    duty += 1

# Wait a couple of seconds
time.sleep(2)

# Turn back to 90 degrees
print("Turning back to 90 degrees for 2 seconds")
servo1.ChangeDutyCycle(7)
time.sleep(0.5)
servo1.ChangeDutyCycle(0)
time.sleep(1.5)

# Turn back to 0 degrees
print("Turning back to 0 degrees")
servo1.ChangeDutyCycle(2)
time.sleep(0.5)
servo1.ChangeDutyCycle(0)

# Clean things up at the end
servo1.stop()
GPIO.cleanup()
print("Goodbye")
```

3. Seven Segment Display



Display seven segment with yes or no

Program:

```
import RPi.GPIO as GPIO
import time

# Suppress warnings
GPIO.setwarnings(False)

# Set up the GPIO mode
GPIO.setmode(GPIO.BOARD)

# Define the GPIO pins
pins = [33, 31, 36, 38, 40, 37, 35]

# Set up each pin as an output
for i in pins:
    GPIO.setup(i, GPIO.OUT)

choice = 'y'
while choice != 'n':
    for i in pins:
        GPIO.output(i, True)  # Turn on the pin
        time.sleep(1)         # Wait for 1 second
        GPIO.output(i, False) # Turn off the pin
    choice = input('Want to continue [y/n]? ')

# Clean up GPIO settings
GPIO.cleanup()
```