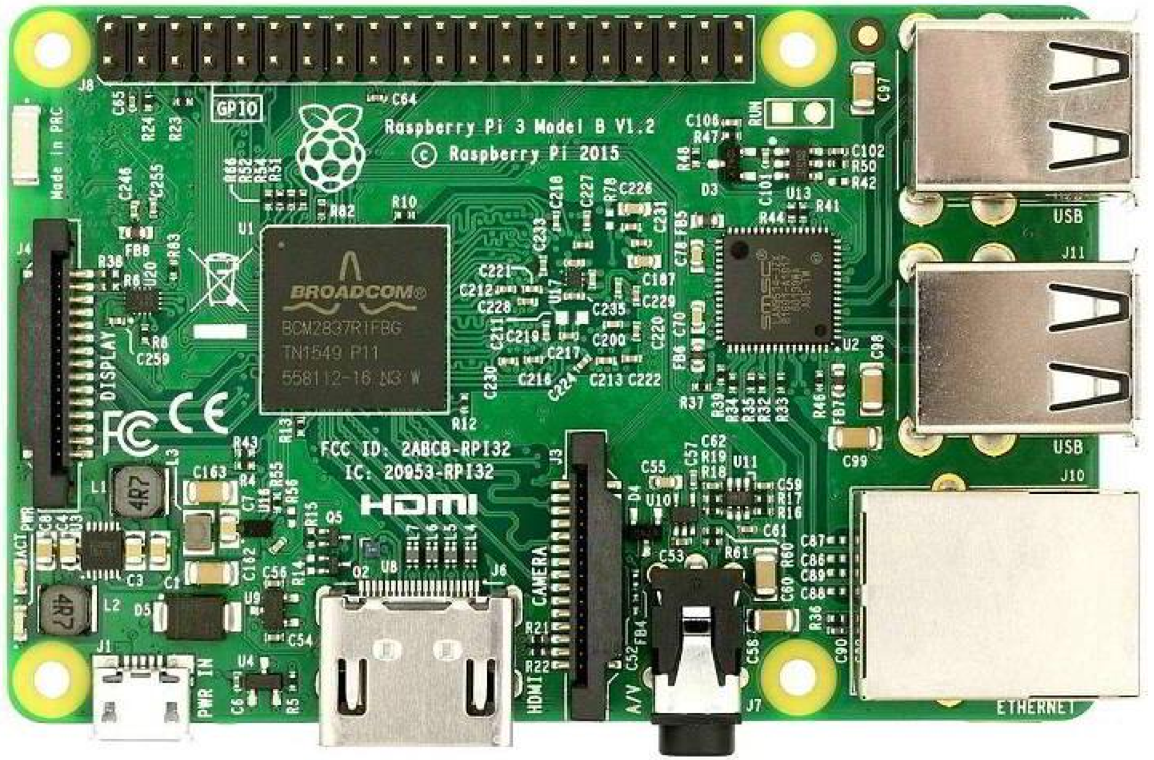
Introduction to Raspberry Pi and It's Programming



Resource Person

Dr. Vishal V. Panchbhai,

Assistant Professor,

Department of Electronics & Telecommunication Engg.

Priyadarshini College of Engineering, Nagpur

What is Raspberry Pi?

- Raspberry Pi is a **small single board computer**. By connecting peripherals like Keyboard, mouse, display to the Raspberry Pi, it will act as a **mini personal computer.**
- Raspberry Pi is popularly used for **real time Image/Video Processing, IoT based applications and Robotics applications.**
- Raspberry Pi **is slower than** laptop or desktop but is still a computer which can provide all the expected features or abilities, at a low power consumption.

OS for Raspberry Pi

- Raspberry Pi Foundation officially provides **Debian based Raspbian OS**. Also, they provide **NOOBS OS** for Raspberry Pi.

- We can install several **Third-Party versions** of OS like Ubuntu, Archlinux, RISC OS, Windows 10 IOT Core, etc.

- **Raspbian OS** is official Operating System available for free to use. This OS is efficiently optimized to use with Raspberry Pi. Raspbian have GUI which includes tools for Browsing,

Raspberry Pi Processor

- It has ARM based **Broadcom Processor** SoC along with **on-chip GPU** (Graphics Processing Unit).

- The CPU speed of Raspberry Pi varies from **700 MHz to 1.2 GHz**. Also, it has on-board SDRAM that ranges from **256 MB to 1 GB**.

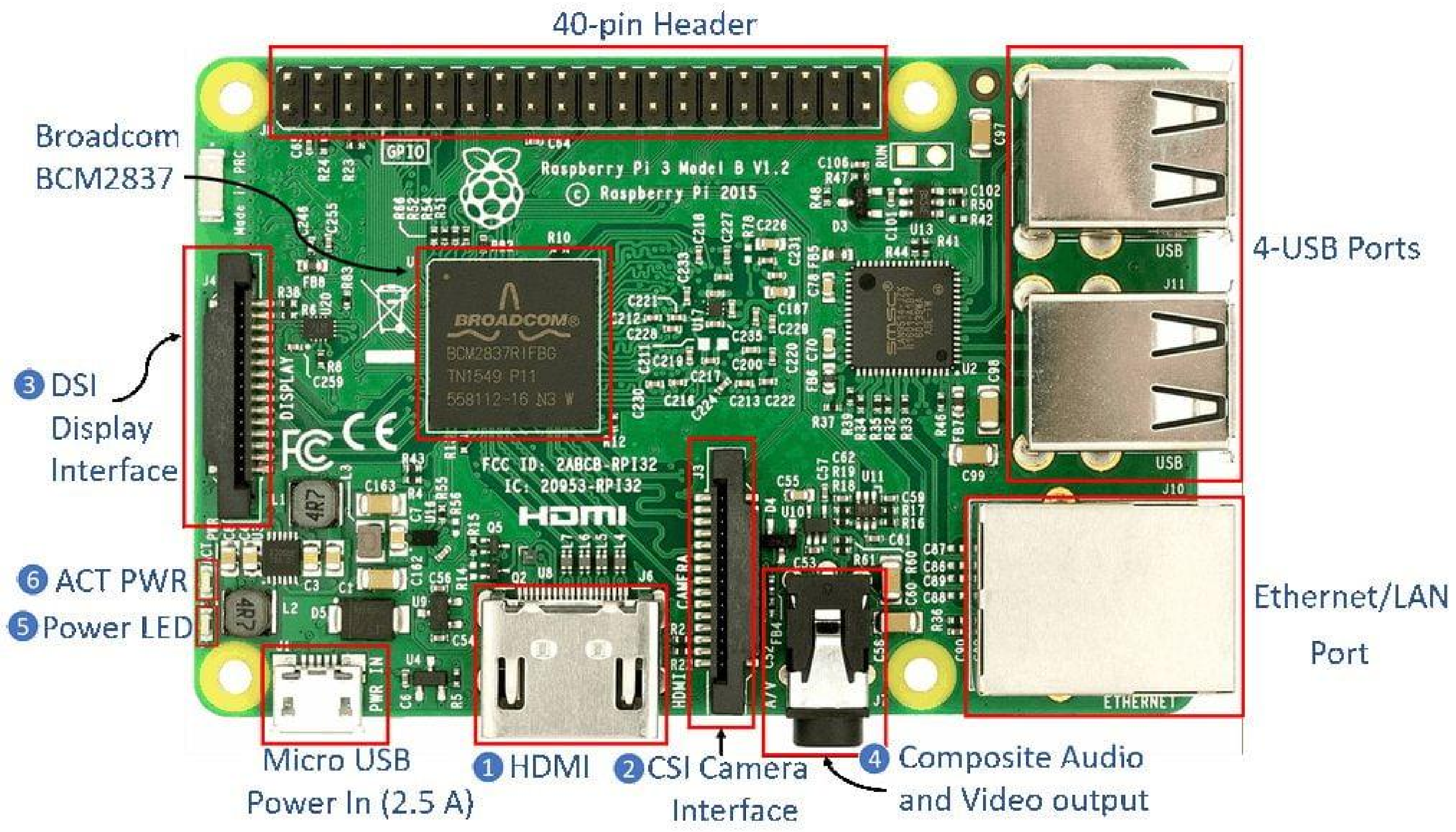
- Raspberry Pi also provides **on-chip SPI, I2C, and UART modules.**

Versions of Raspberry Pi Models

There are different versions of raspberry pi available as listed below:

- **Raspberry Pi 1 Model A**
- **Raspberry Pi 1 Model A+**
- **Raspberry Pi 1 Model B**
- **Raspberry Pi 1 Model B+**
- **Raspberry Pi 2 Model B**
- **Raspberry Pi 3 Model B**
- **Raspberry Pi 4 Model B**
- **Raspberry Pi Zero**

Raspberry Pi 3 Hardware Details



Raspberry Pi 3 Hardware Details

- **HDMI (High-Definition Multimedia Interface):** It is used for transmitting uncompressed video or digital audio data to the Computer Monitor, Digital TV, etc. Generally, this HDMI port helps to connect Raspberry Pi to the Digital television.

- **CSI Camera Interface:** CSI (Camera Serial Interface) interface provides a connection in between Broadcom Processor and Pi camera. This interface provides electrical connections between two devices.

- **DSI Display Interface:** DSI (Display Serial Interface) Display Interface is used for connecting LCD to the Raspberry Pi using 15-pin ribbon cable. DSI provides fast High-resolution display interface specifically used for sending video data directly from GPU to the LCD display.

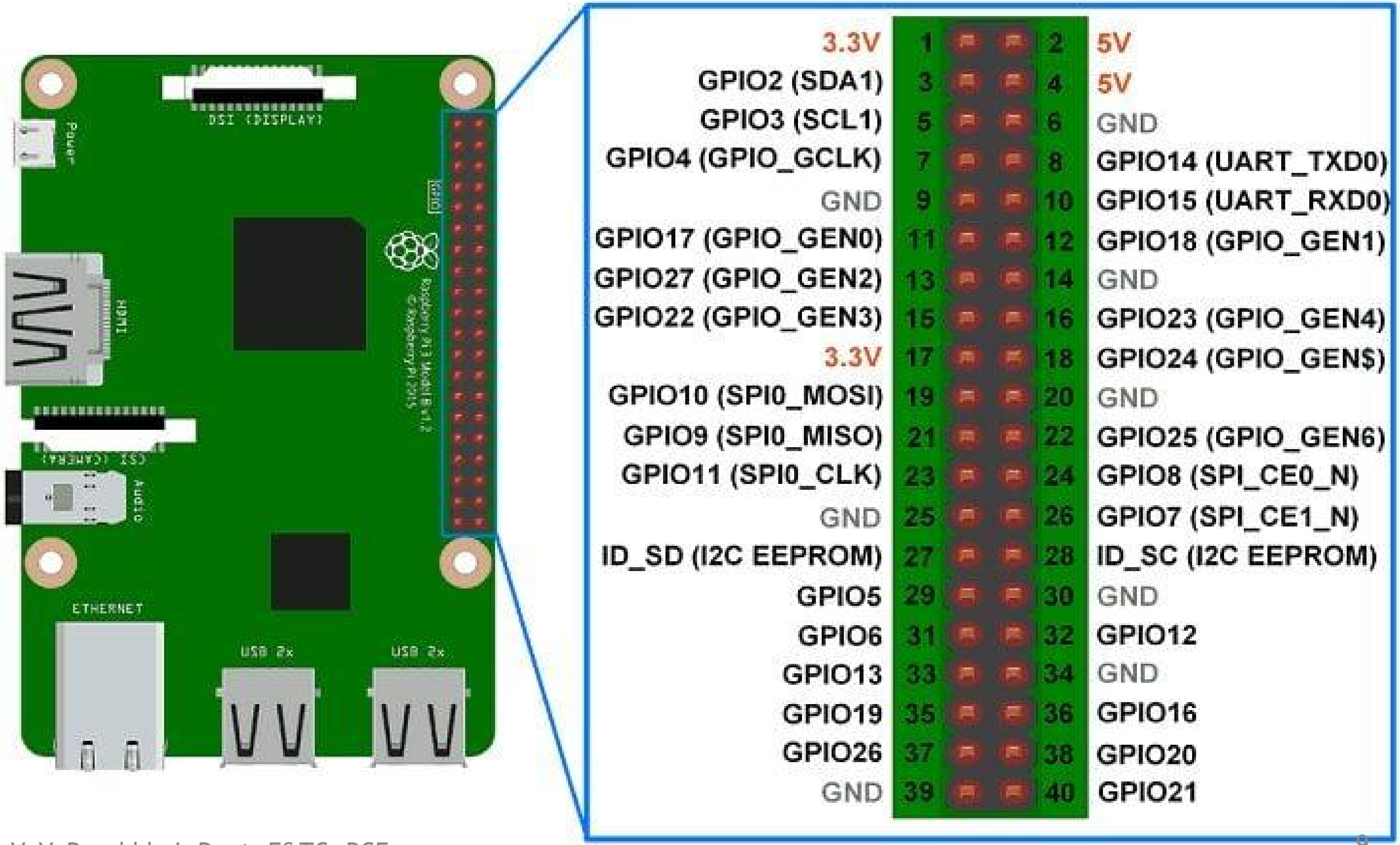
Raspberry Pi 3 Hardware Details

- **Composite Video and Audio Output:** The composite Video and Audio output port carries video along with audio signal to the Audio/Video systems.

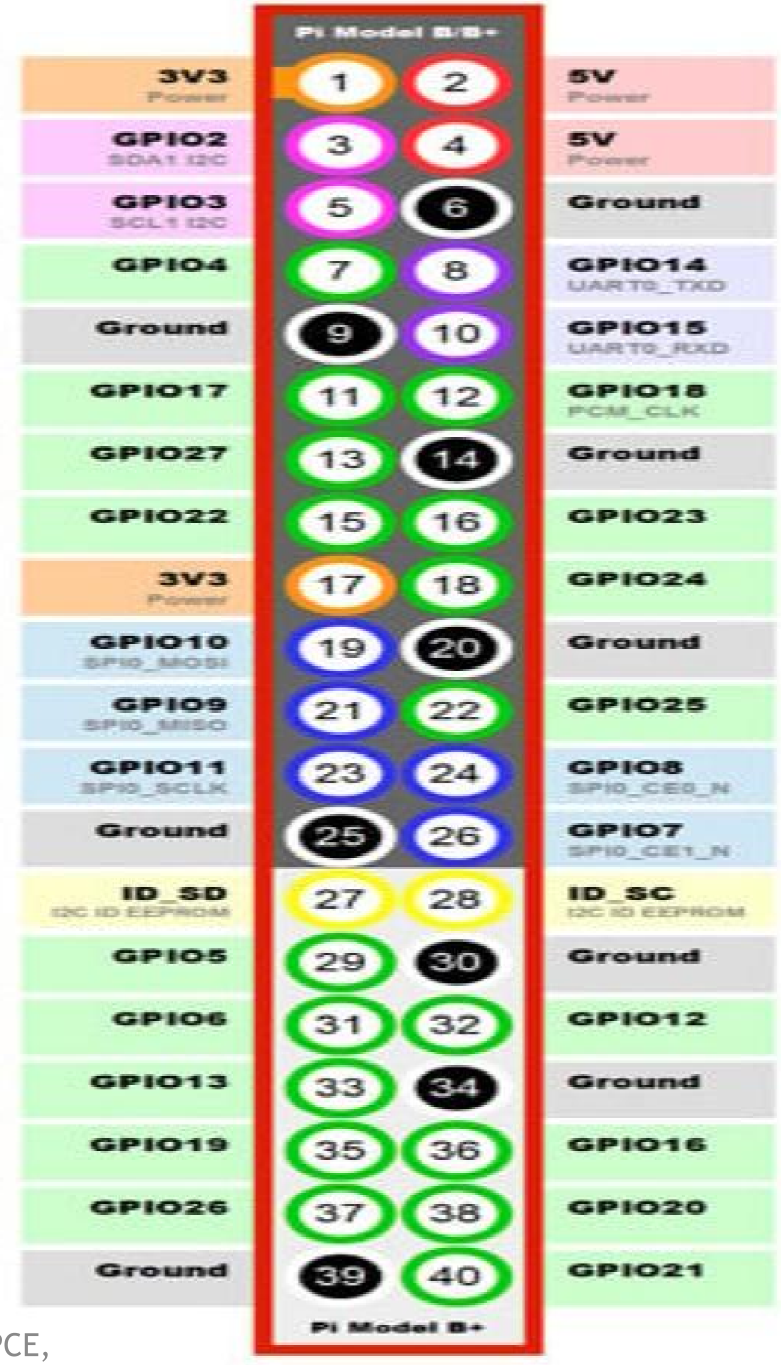
- **Power LED:** It is a RED colored LED which is used for Power indication. This LED will turn ON when Power is connected to the Raspberry Pi. It is connected to 5V directly and will start blinking whenever the supply voltage drops below 4.63V.

- **ACT PWR:** ACT PWR is Green LED which shows the SD card activity.

Raspberry Pi GPIO Access



3.3V	1	2	5V
GPIO2 (SDA1)	3	4	5V
GPIO3 (SCL1)	5	6	GND
GPIO4 (GPIO_GCLK)	7	8	GPIO14 (UART_TXD0)
GND	9	10	GPIO15 (UART_RXD0)
GPIO17 (GPIO_GEN0)	11	12	GPIO18 (GPIO_GEN1)
GPIO27 (GPIO_GEN2)	13	14	GND
GPIO22 (GPIO_GEN3)	15	16	GPIO23 (GPIO_GEN4)
3.3V	17	18	GPIO24 (GPIO_GEN$)
GPIO10 (SPI0_MOSI)	19	20	GND
GPIO9 (SPI0_MISO)	21	22	GPIO25 (GPIO_GEN6)
GPIO11 (SPI0_CLK)	23	24	GPIO8 (SPI_CE0_N)
GND	25	26	GPIO7 (SPI_CE1_N)
ID_SD (I2C EEPROM)	27	28	ID_SC (I2C EEPROM)
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21



Pi Model B/B+
3V3 Power
1
2
5V Power
GPIO2 SDA1 I2C
3
4
5V Power
GPIO3 SCL1 I2C
5
6
Ground
GPIO4
7
8
GPIO14 UART0_TXD
Ground
9
10
GPIO15 UART0_RXD
GPIO17
11
12
GPIO18 PCM_CLK
GPIO27
13
14
Ground
GPIO22
15
16
GPIO23
3V3 Power
17
18
GPIO24
GPIO10 SPI0_MOSI
19
20
Ground
GPIO9 SPI0_MISO
21
22
GPIO25
GPIO11 SPI0_SCLK
23
24
GPIO8 SPI0_CE0_N
Ground
25
26
GPIO7 SPI0_CE1_N
ID_SD I2C ID EEPROM
27
28
ID_SC I2C ID EEPROM
GPIO5
29
30
Ground
GPIO6
31
32
GPIO12
GPIO13
33
34
Ground
GPIO19
35
36
GPIO16
GPIO26
37
38
GPIO20
Ground
39
40
GPIO21
Pi Model B+

Raspberry Pi GPIO Access

GPIO (**General Purpose Input Output**) pins can be used as input or output and allows raspberry pi to connect with general purpose I/O devices.

- Raspberry pi 3 model B took out **26 GPIO pins** on board.
- Raspberry pi can **control many external I/O devices** using these GPIO’s.
- These pins are a **physical interface** between the Pi and the outside world.
- We can program these pins according to our needs to interact with external devices.
- Raspberry Pi can **connect to the Internet** using Ethernet port or on-board Wi-Fi adapter.

Pin Numbering

We should define GPIO pin which we want to use as an output or input. But Raspberry Pi has **two ways of defining pin number** which are as follows:

- **GPIO Numbering**
- **Physical Numbering**

GPIO Numbering

- In **GPIO Numbering**, pin number refers to number on Broadcom SoC (System on Chip) channel **(BCM).** So, we should always consider the pin mapping for using GPIO pin. GPIO Numbering (BCM) is defined as

```
GPIO.setmode(GPIO.BCM)
```

Physical Numbering

- While in **Physical Numbering**, pin number refers to the pin of 40-pin P1 header on Raspberry Pi **Board.** The above physical numbering is simple as we can count pin number on P1 header and assign it as GPIO. Physical Numbering (Board) is defined as

```
GPIO.setmode(GPIO.BOARD)
```

Libraries Generally Used

1) RPi.GPIO

- **Overview:** Provides low-level control of GPIO pins.
- **Key Features:**
  - Configure pin modes (input/output).
  - Control digital signals (high/low).
  - Event detection (button presses).
- **Use Case:** Low-level hardware control (LEDs, sensors).
- Code Example:

```
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)
GPIO.output(18, GPIO.HIGH)
```

Libraries Generally Used

2) time

- **Overview:** The time library provides various time-related functions.
- **Key Functions:**
  - time.time(): Returns current time in seconds since the epoch.
  - time.sleep(): Pauses execution for a specified number of seconds.
  - time.strftime(): Formats time into a readable string.
  - time.localtime(): Returns the current local time.
- **Use Case Example:** Delay execution to control timing of hardware events.
- Eg. **time.sleep(1)**

Libraries Generally Used

3) picamera

- **Overview:** picamera allows for interfacing with the Raspberry Pi camera module.
- **Key Features:**
  - Capture still images and video.
  - Real-time video stream handling.
  - Controlling camera properties like brightness, contrast, etc.
- **Use Case Example:** Capturing images for a security system or time-lapse photography.
- Code Example:

```
from picamera import PiCamera
camera = PiCamera()
camera.capture('image.jpg')
```

Libraries Generally Used

4) GPIOZero Library

- **Overview:** gpiozero is used for controlling the GPIO pins on the Raspberry Pi.
- **Key Features:**
  - Control LEDs, buttons, motors, and other components.
  - Provides a simple interface for hardware control.
- **Use Case Example:** Automating home appliances using GPIO pins.
- Code Example:

```
from gpiozero import LED
led = LED(17)
led.on()
```

Libraries Generally Used

5) Adafruit_DHT Library

- **Overview:** The Adafruit_DHT library is used to interface with DHT11/DHT22 humidity and temperature sensors.
- **Key Features:**
  - Retrieve temperature and humidity data from the sensor.
  - Supports multiple sensor types (DHT11, DHT22).
- **Use Case Example:** Home automation and weather stations.
- Code Example:

```
import Adafruit_DHT
sensor = Adafruit_DHT.DHT22
humidity, temperature =
```

Libraries Generally Used

6) Serial Library

- **Overview:** The serial library enables serial communication with devices connected to the Raspberry Pi via UART or USB.
- **Key Features:**
  - Send and receive data over serial communication.
  - Configure baud rate, timeout, and port settings.
- **Use Case Example:** Communicating with Arduino or other microcontrollers.
- Code Example:

```
import serial
ser = serial.Serial('/dev/ttyUSB0', 9600)
ser.write(b'Hello!')
```

Libraries Generally Used

7) Math Library

- **Overview:** The math library provides mathematical functions.
- **Key Functions:**
  - math.sqrt(): Square root.
  - math.pi: Constant Pi.
  - math.sin(), math.cos(), math.tan(): Trigonometric functions.
- **Use Case Example:** Perform calculations in robotic control systems.
- Code Example:

```
import math
result = math.sqrt(16)
```

Functions Used

1) GPIO.setmode (Pin Numbering System)

- This function is used to define Pin numbering system i.e. GPIO numbering or Physical numbering.

In BCM,

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(21, GPIO.OUT)
```

In BOARD,

```
GPIO.setmode(GPIO.BOARD)
GPIO.setup(40, GPIO.OUT)
```

Functions Used

2) GPIO.setwarnings(False)

- When we initialize any GPIO pin it shows warning while compiling because of previous initialization of same pin.
- To avoid this warning, we should disable warnings using command

GPIO.setwarnings(False)

Functions Used

3) GPIO.setup (channel, direction, initial value, pull up/pull down)

This function is used to set the direction of GPIO pin as an input/output.

- channel – GPIO pin number as per numbering system.
- direction – set direction of GPIO pin as either Input or Output.
- initial value – can provide initial value
- pull up/pull down – enable pull up or pull down if required

Functions Used

3) GPIO.setup (channel, direction, initial value, pull up/pull down)

Few examples are given as follows,

- GPIO as Output

  GPIO.setup(channel, GPIO.OUT)
- GPIO as Input

  GPIO.setup(channel, GPIO.IN)
- GPIO as Output with initial value

  GPIO.setup(channel, GPIO.OUT, initial=GPIO.HIGH)
- GPIO as Input with Pull up resistor

  GPIO.setup(channel, GPIO.IN, pull_up_down = GPIO.PUD_UP)

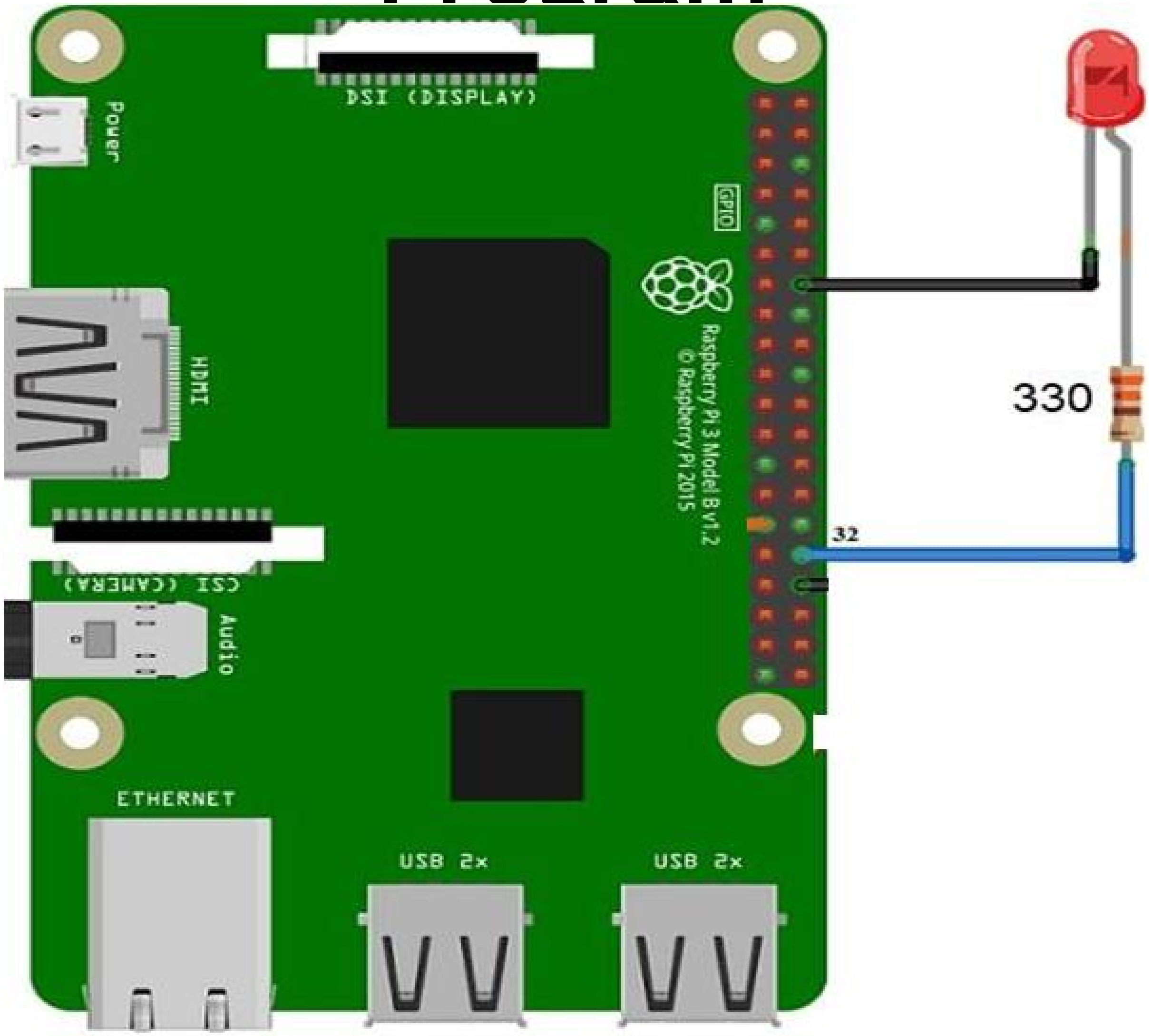
Functions Used

4) GPIO.output(channel, state)

- This function is used to set the output state of GPIO pin.
- channel – GPIO pin number as per numbering system.
- state – Output state i.e. HIGH or LOW of GPIO pin

e.g. **GPIO.output(7, GPIO.HIGH)**

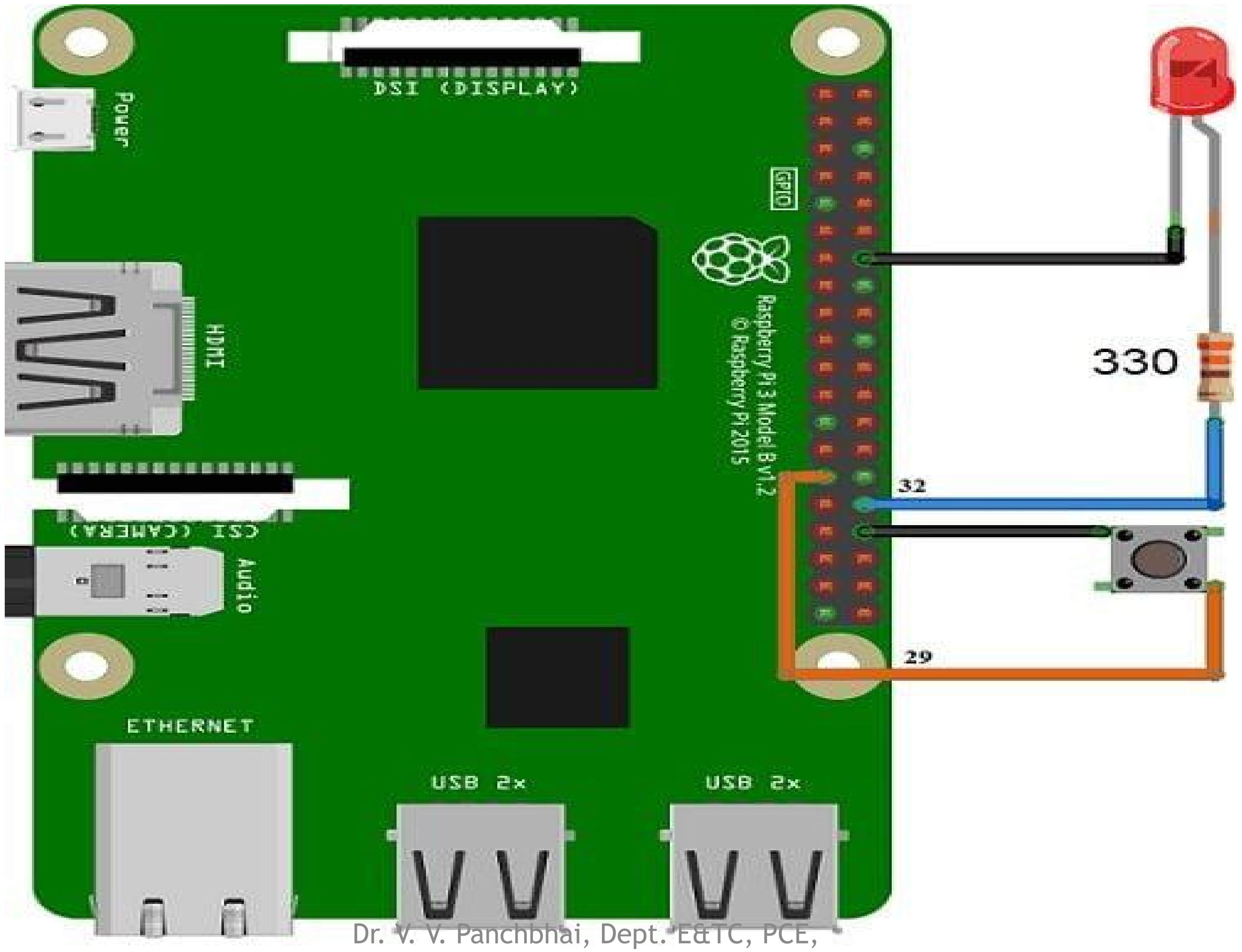
Interface LED and Write Blink Program



Program

```
import RPi.GPIO as GPIO
import time
ledpin=12
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(ledpin, GPIO.OUT)
while True:
    GPIO.output(ledpin, GPIO.HIGH)
    print('LED ON')
    time.sleep(1)
    GPIO.output(ledpin, GPIO.LOW)
    print('LED OFF')
    time.sleep(2)
```

Interface LED and Push Button

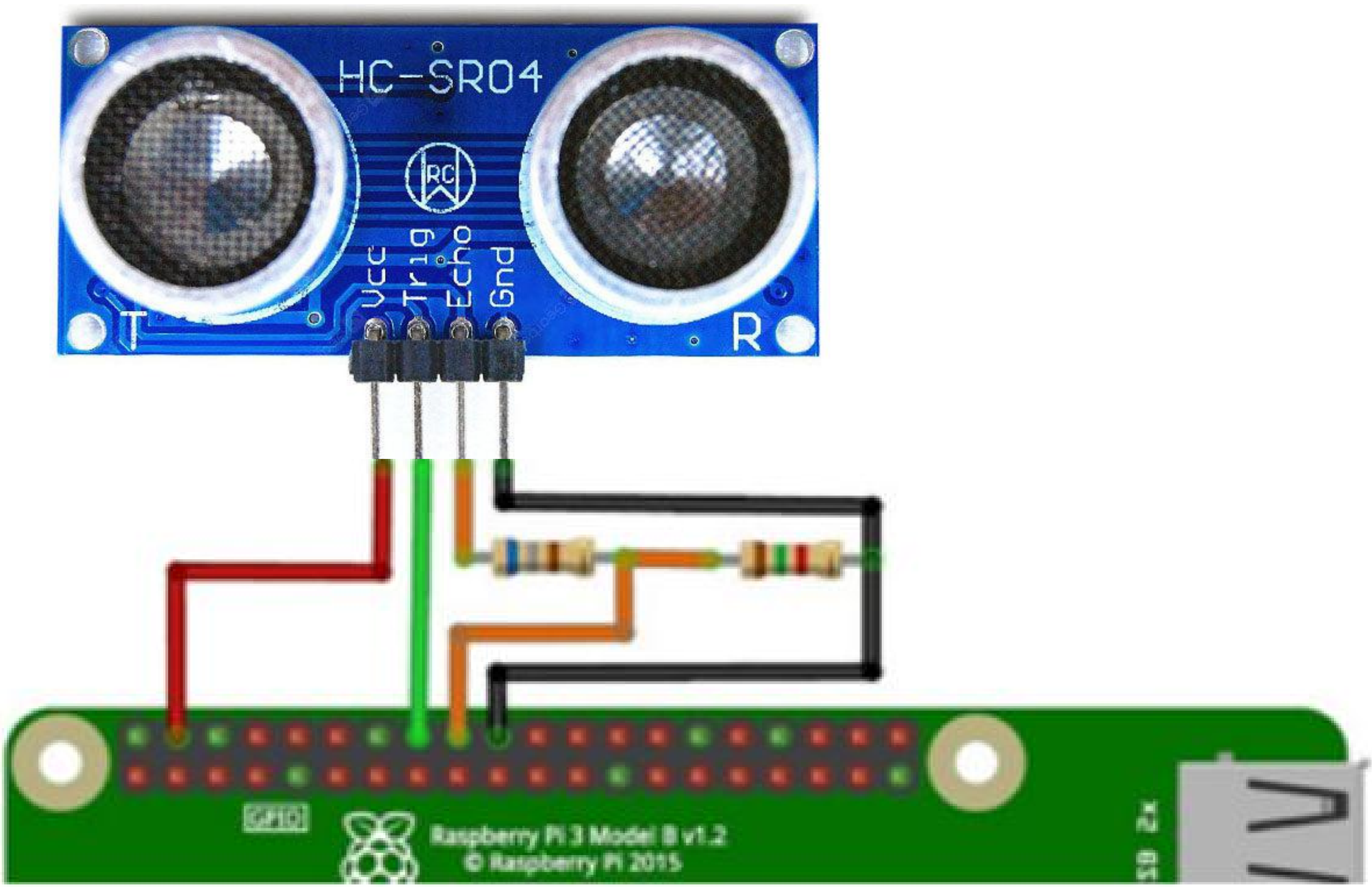


Program

```
import RPi.GPIO as GPIO  #import RPi.GPIO module

LED = 32                 #pin no. as per BOARD, GPIO18 as per
   BCM
Switch_input = 29   #pin no. as per BOARD, GPIO27 as per
   BCM

GPIO.setwarnings(False)  #disable warnings
GPIO.setmode(GPIO.BOARD)      #set pin numbering format
GPIO.setup(LED, GPIO.OUT)      #set GPIO as output
GPIO.setup(Switch_input, GPIO.IN,
   pull_up_down=GPIO.PUD_UP)

while True:
    if(GPIO.input(Switch_input)):
        GPIO.output(LED,GPIO.LOW)
    else:
```

Interface Picamera and Write Program

Interface Ultrasonic sensor



Program

```
from gpiozero import DistanceSensor
from time import sleep

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

ultrasonic_sensor = DistanceSensor(echo=5,
   trigger=6)
while True:
    distance= (ultrasonic_sensor.distance*100)
    print('distance = {0:0.2f} cm'.format (distance))
    sleep(1)
```

References

[1] https://www.electronicwings.com/raspberry-pi

[2] https://www.w3schools.com/nodejs/nodejs_raspberrypi.asp

[3] https://magpi.raspberrypi.com/issues/143/contributions/new

Thank You