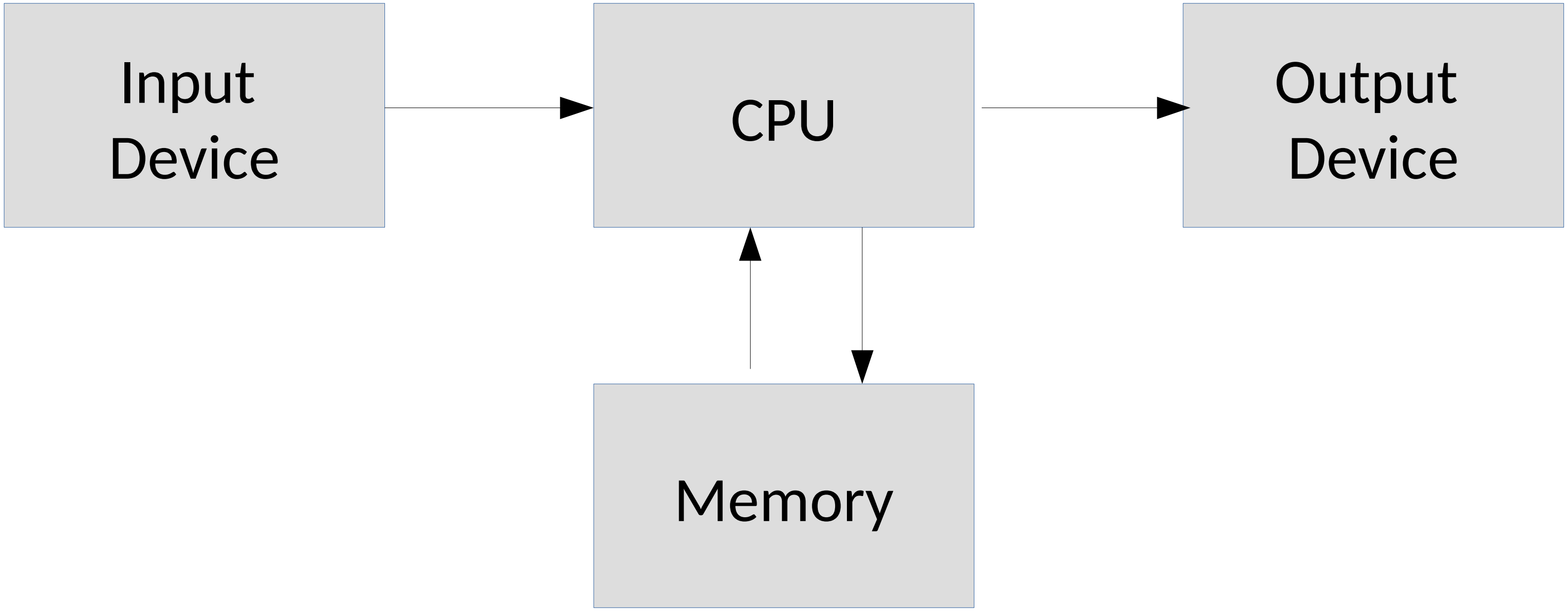
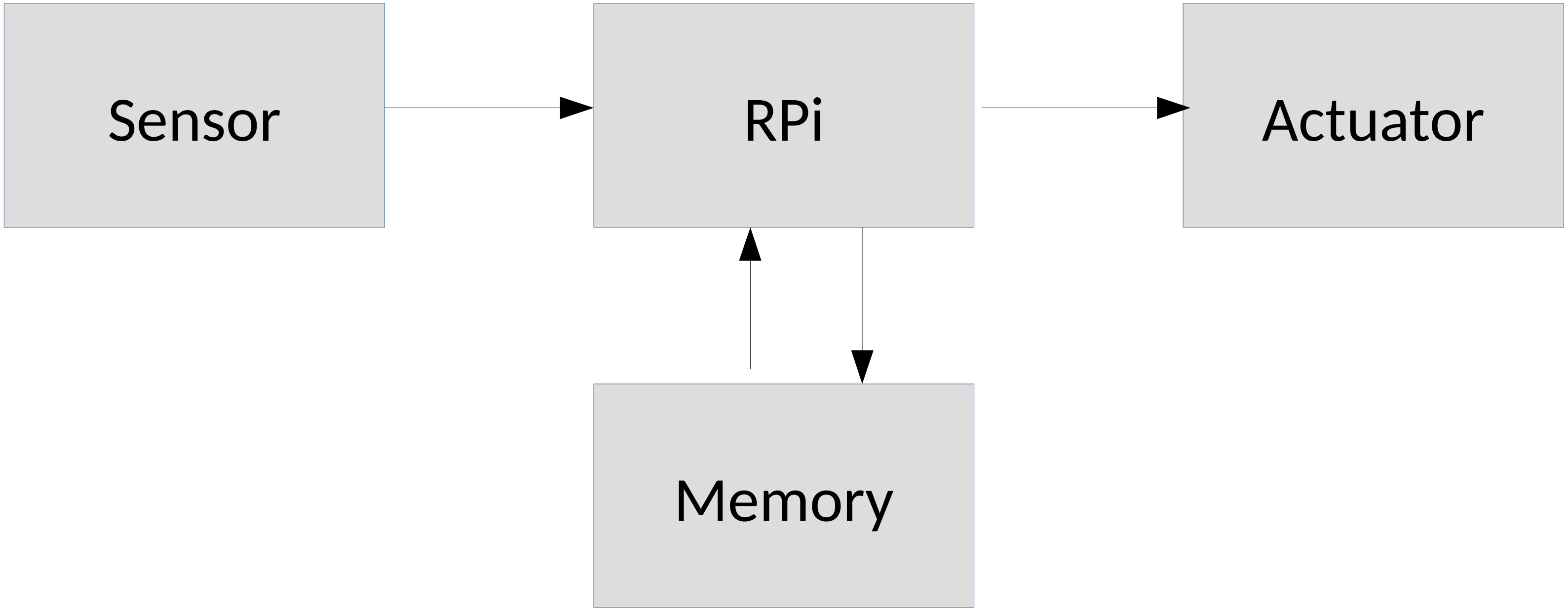
Sensors and Actuators using RPi

Tushar B. Kute,
http://tusharkute.com

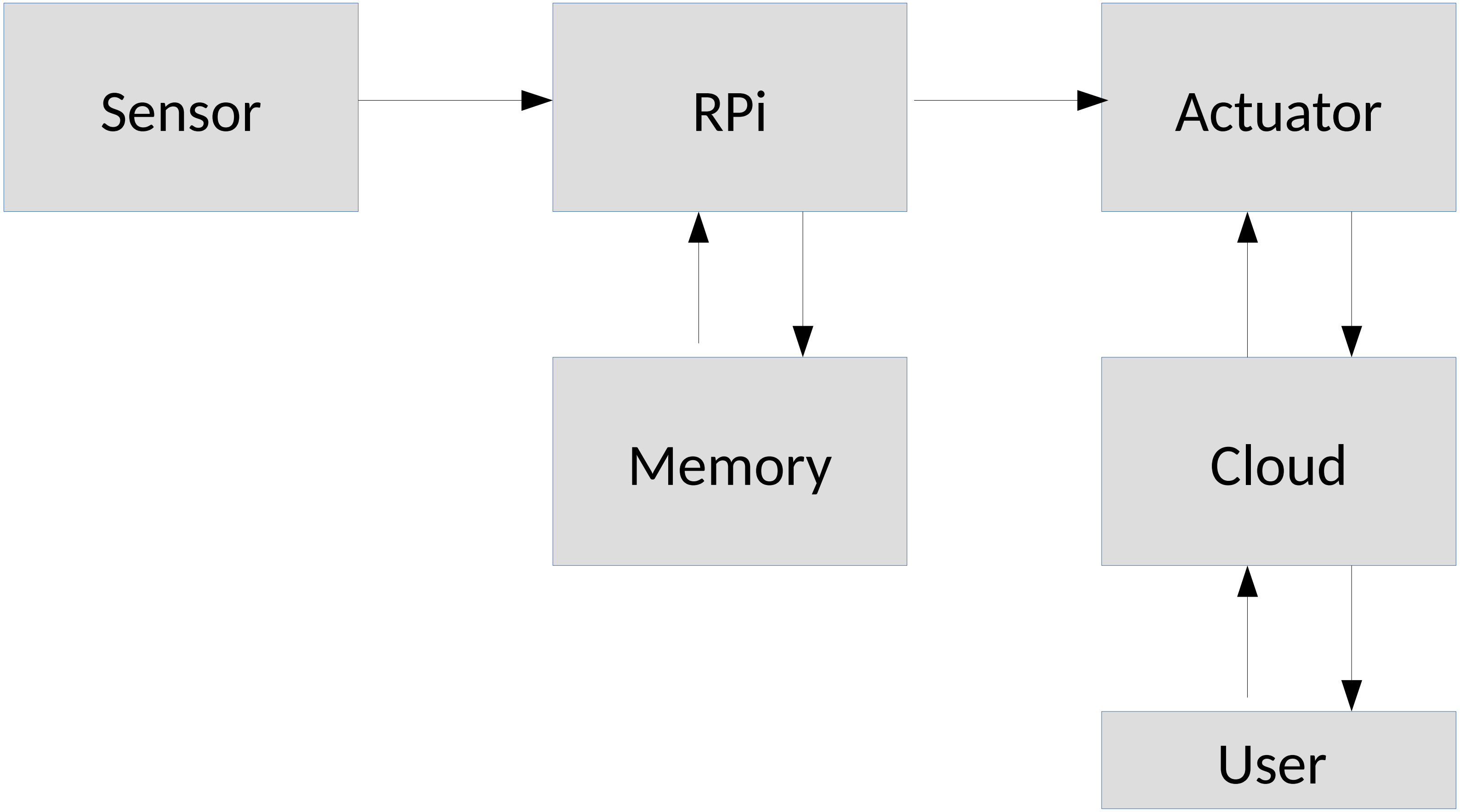
General Computer Architecture



RPi Embedded System Architecture



Simple IOT Architecture



Sensor

- A Sensor or a Detector is a device that is used to convert a physical parameter into a signal that can be measured or monitored.
- For example a GAS Sensor monitors gas concentration (PPM) and converts it into electrical signal that can be measured.
- The input parameter can be different like Light, Temperature, Humidity pressure etc but the output is generally a human readable or electrically monitorable.

Sensor Definition

- According to Oxford Dictionaries definition of Sensor is "A device which detects or measures a physical property and records, indicates, or otherwise responds to it."
- Meaning of Sensor as per Wikipedia "A sensor is an object whose purpose is to detect events or changes in its environment, and then provide a corresponding output"

Types of Sensors

- A sensor is classified based on various aspects such as
  - Application Based: Industrial Sensor, Automotive Sensor etc
  - Output Based : Resistive output, Differential Output, Differential output, voltage output etc
  - Parameter Sensing Based: Light, Temperature etc

Commonly used sensors

- Light Sensor
- Temperature Sensor
- Proximity Sensor
- Pressure Sensor
- GAS Sensor
- Current Sensor
- Bio-Medical Sensor
- Sound Sensor
- Tilt Sensor
- Hall effect Sensor
- Accelerometer Sensor
- Compass Sensor
- Flow Sensor
- Humidity Sensor
- Level Sensor
- Motion Sensor
- Speed Sensor
- RPM Sensor
- Force Sensor

Actuators

- An actuator is a component of a machine that is responsible for moving or controlling a mechanism or system.
- An actuator requires a control signal and a source of energy. The control signal is relatively low energy and may be electric voltage or current, pneumatic or hydraulic pressure, or even human power.
- When the control signal is received, the actuator responds by converting the energy into mechanical motion.

Actuators

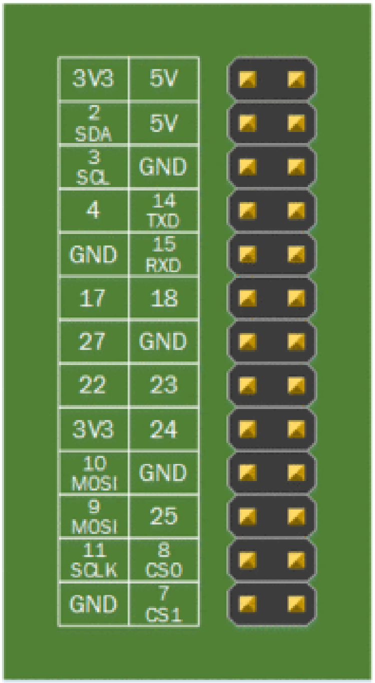
- Following basic actuators are used for signaling and output purpose:
  - LED
  - RGB LED
  - Buzzer
  - Servo Motor
  - DC Motor
  - Relay

Sensors

Joystick
Mercury Switch
Auto Flash LED
Sound Sensor
Flame Sensor
Touch Switch
Reed Switch
MPU6050

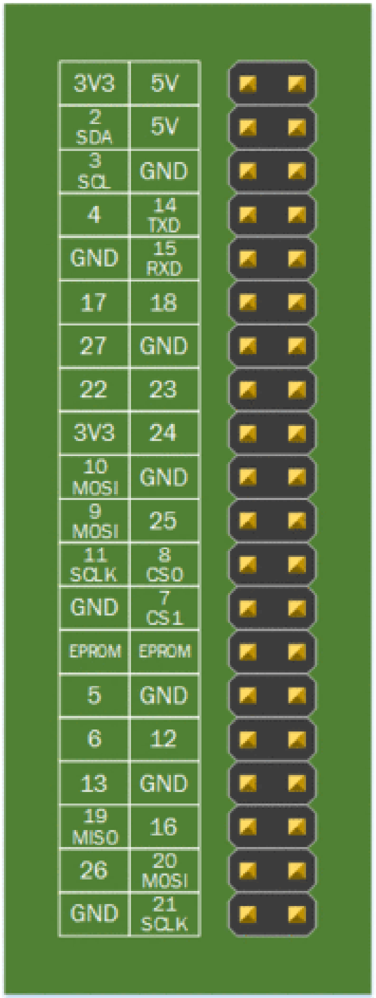
GPIO Pins

Models A & B



3V3	5V
2 SDA	5V
3 SCL	GND
4	14 TXD
GND	15 RXD
17	18
27	GND
22	23
3V3	24
10 MOSI	GND
9 MOSI	25
11 SCLK	8 CS0
GND	7 CS1

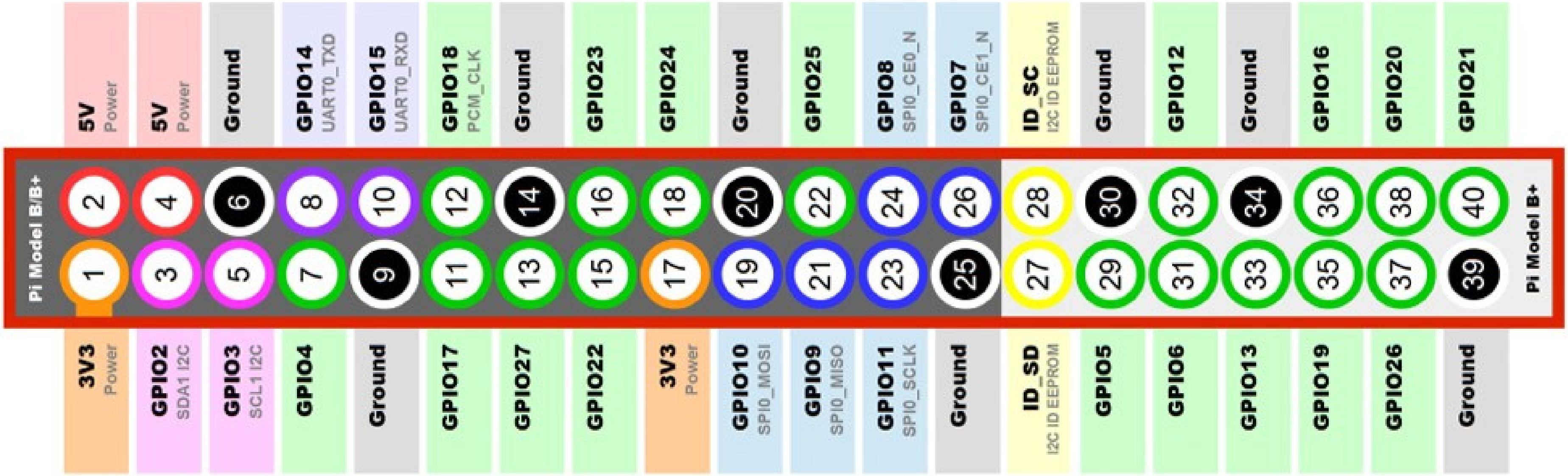
Models A+, B+ & Pi2



3V3	5V
2 SDA	5V
3 SCL	GND
4	14 TXD
GND	15 RXD
17	18
27	GND
22	23
3V3	24
10 MOSI	GND
9 MOSI	25
11 SCLK	8 CS0
GND	7 CS1
EPROM	EPROM
5	GND
6	12
13	GND
19 MISO	16
26	20 MOSI
GND	21 SCLK

GPIO Pins

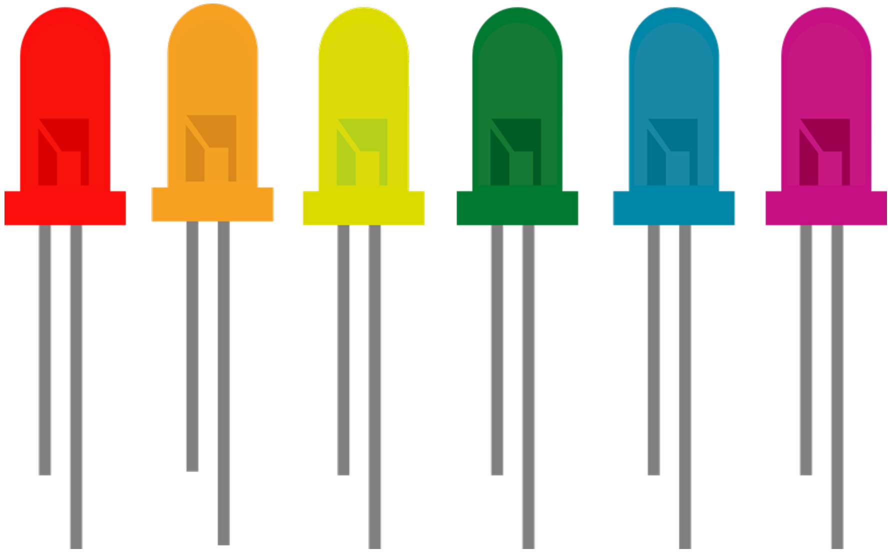
GPIO Pinout Diagram

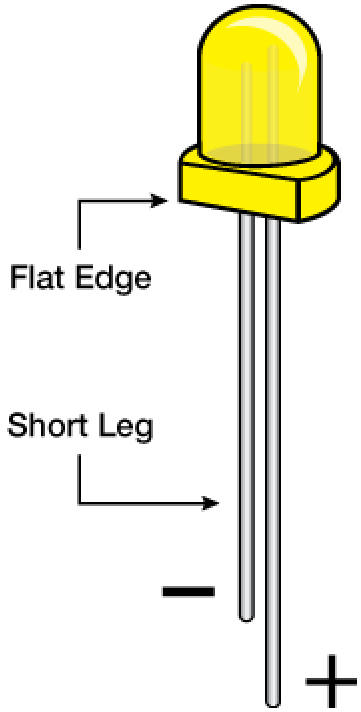


GPIO Modes

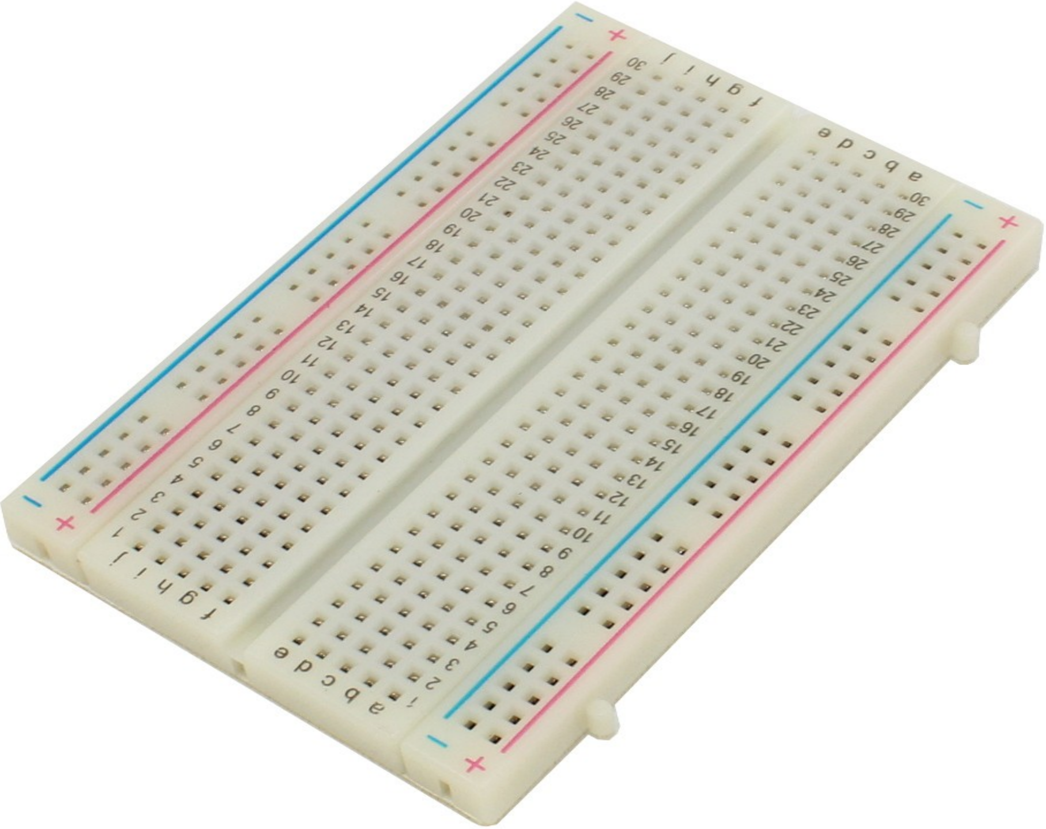
- The **GPIO.BOARD** option specifies that you are referring to the pins by the number of the pin the the plug - i.e the numbers printed on the board (e.g. P1) and in the middle of the diagrams below.
- The **GPIO.BCM** option means that you are referring to the pins by the "Broadcom SOC channel" number, these are the numbers after "GPIO" in the green rectangles around the outside of the below diagrams:
- Unfortunately the **BCM** numbers changed between versions of the Pi1 Model B.
  - The Model B+ uses the same numbering as the Model B r2.0, and adds new pins (board numbers 27-40).
  - The Raspberry Pi Zero, Pi 2B and Pi 3B use the same numbering as the B+.

LED

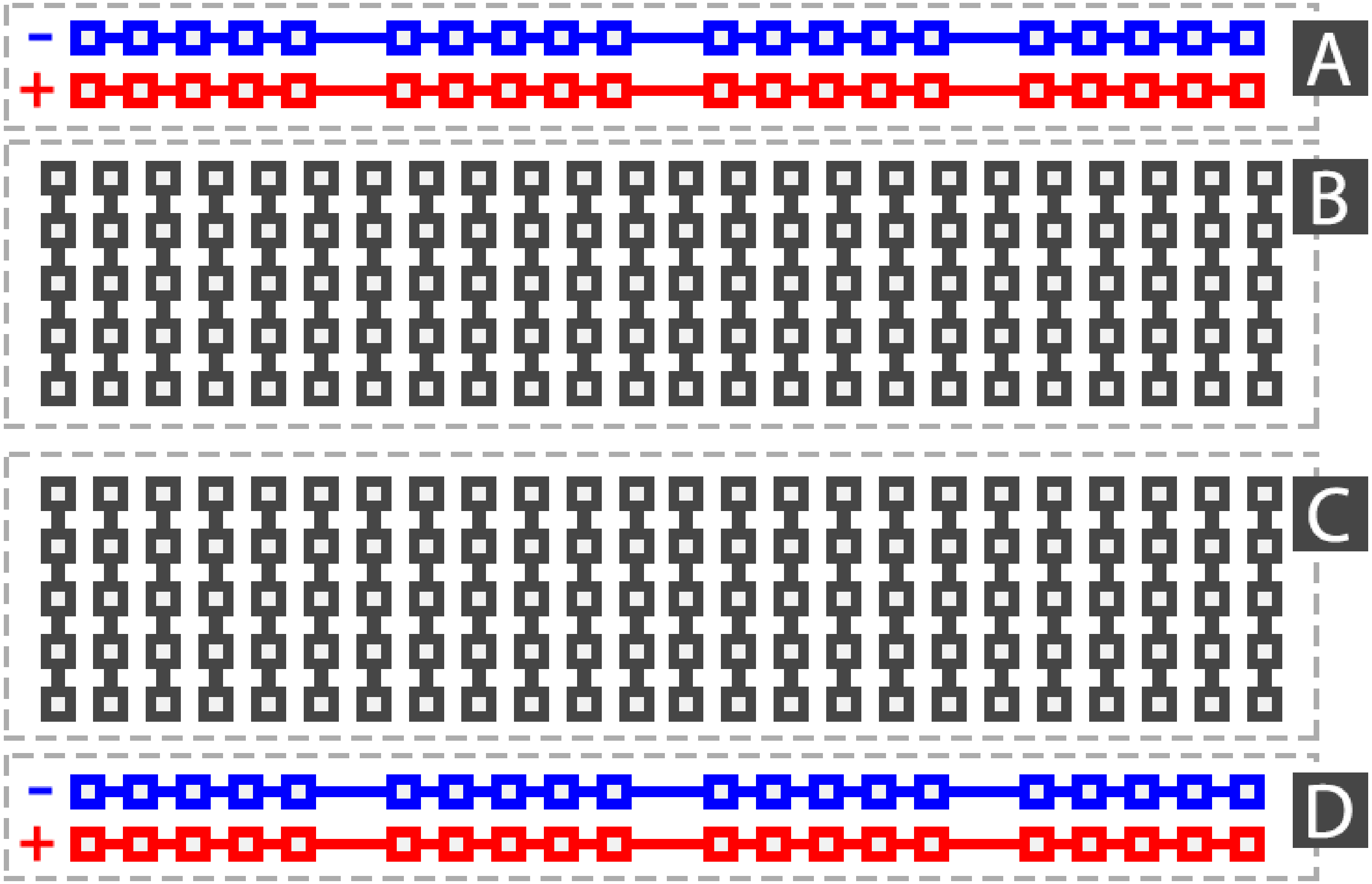




Breadboard


a b c d e
f g h i j
+
-
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

Breadboard internals



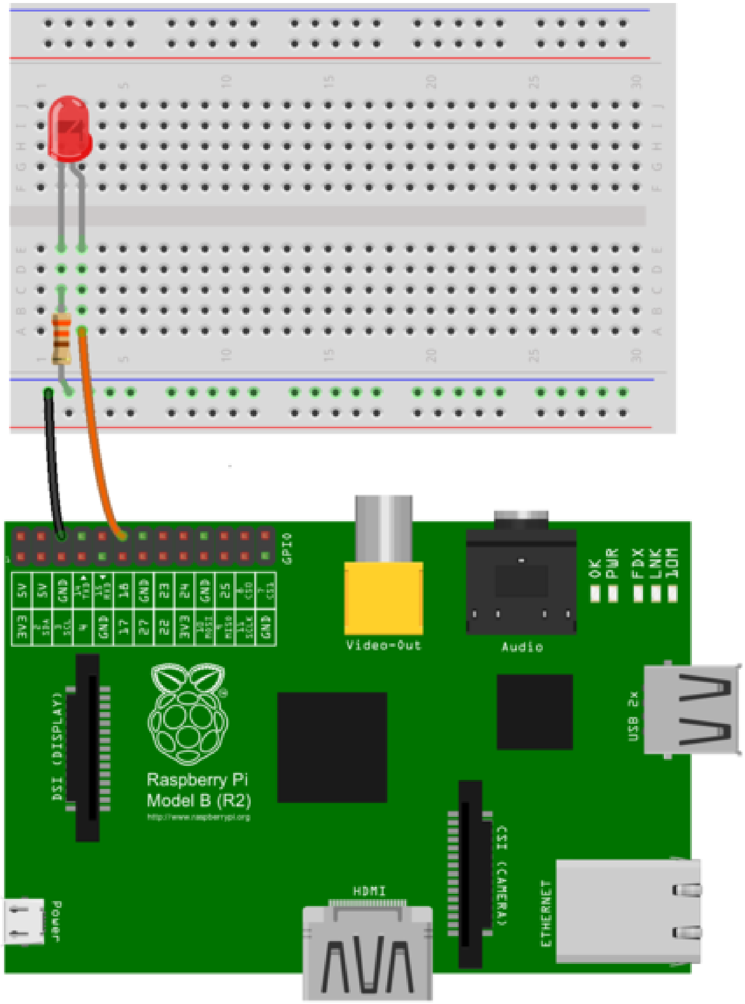
Resister

- You must ALWAYS use resistors to connect LEDs up to the GPIO pins of the Raspberry Pi. The Raspberry Pi can only supply a small current (about 60mA). The LEDs will want to draw more, and if allowed to they will burn out the Raspberry Pi. Therefore putting the resistors in the circuit will ensure that only this small current will flow and the Pi will not be damaged.
- Resistors are a way of limiting the amount of electricity going through a circuit; specifically, they limit the amount of 'current' that is allowed to flow. The measure of resistance is called the Ohm (Ω), and the larger the resistance, the more it limits the current. The value of a resistor is marked with coloured bands along the length of the resistor body.

Jumper Wires

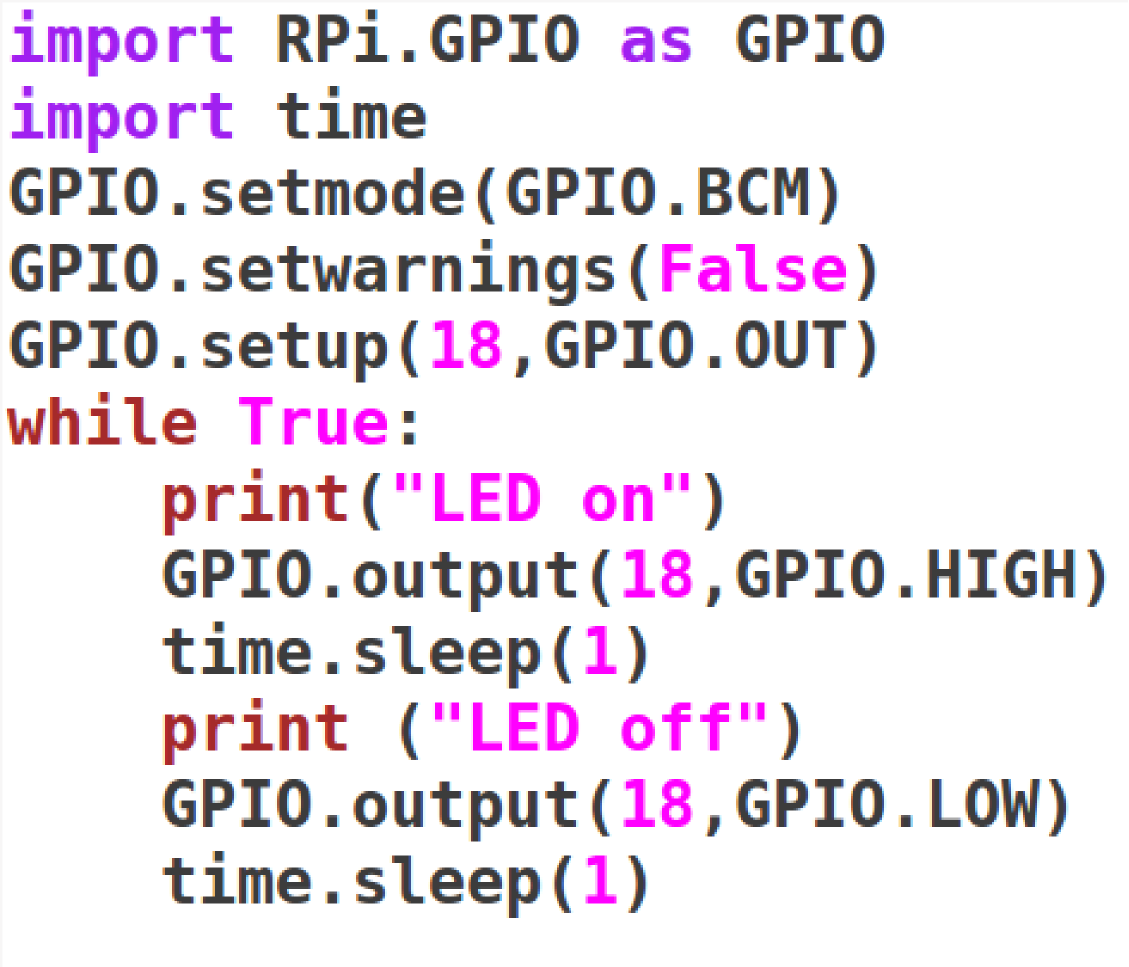
- Jumper wires are used on breadboards to ‘jump’ from one connection to another.
- The ones you will be using in this circuit have different connectors on each end.
- The end with the ‘pin’ will go into the Breadboard.
- The end with the piece of plastic with a hole in it will go onto the Raspberry Pi’s GPIO pins.

Lets start the connections



fritzing

Program:

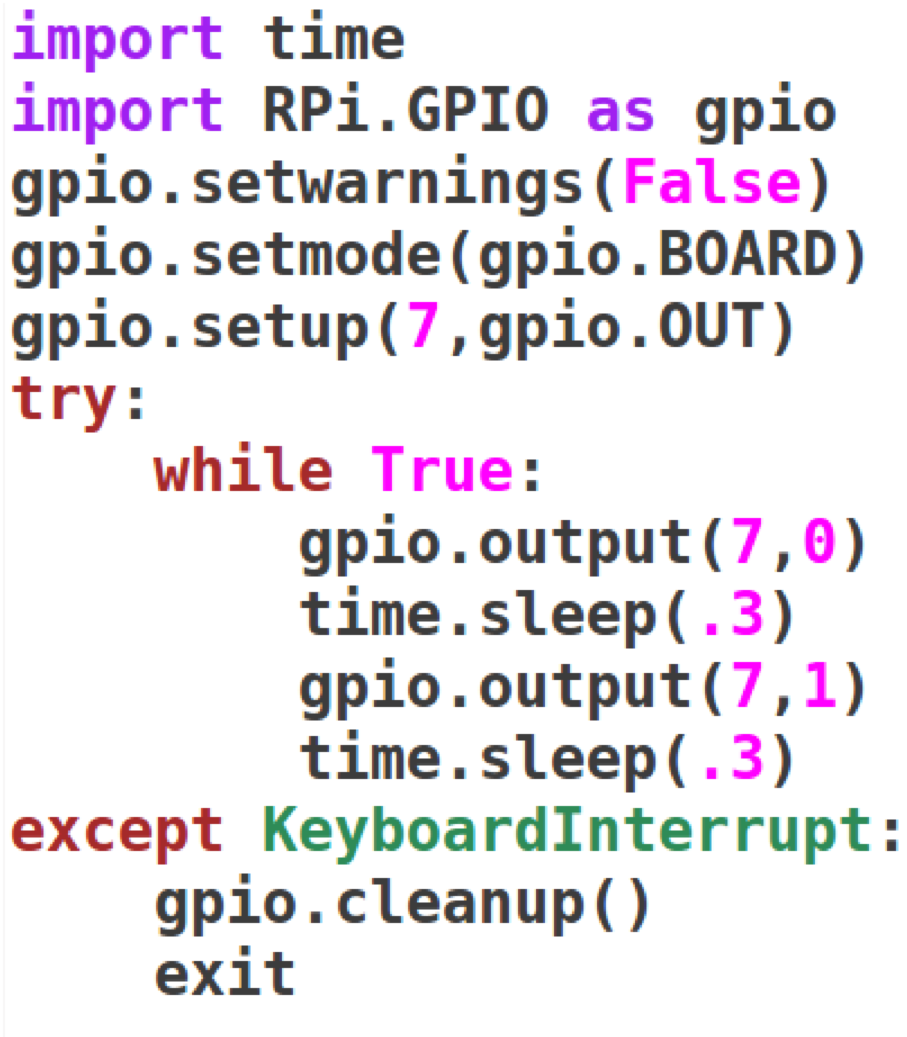


```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(18,GPIO.OUT)
while True:
    print("LED on")
    GPIO.output(18,GPIO.HIGH)
    time.sleep(1)
    print ("LED off")
    GPIO.output(18,GPIO.LOW)
    time.sleep(1)
```

Piezo Buzzer

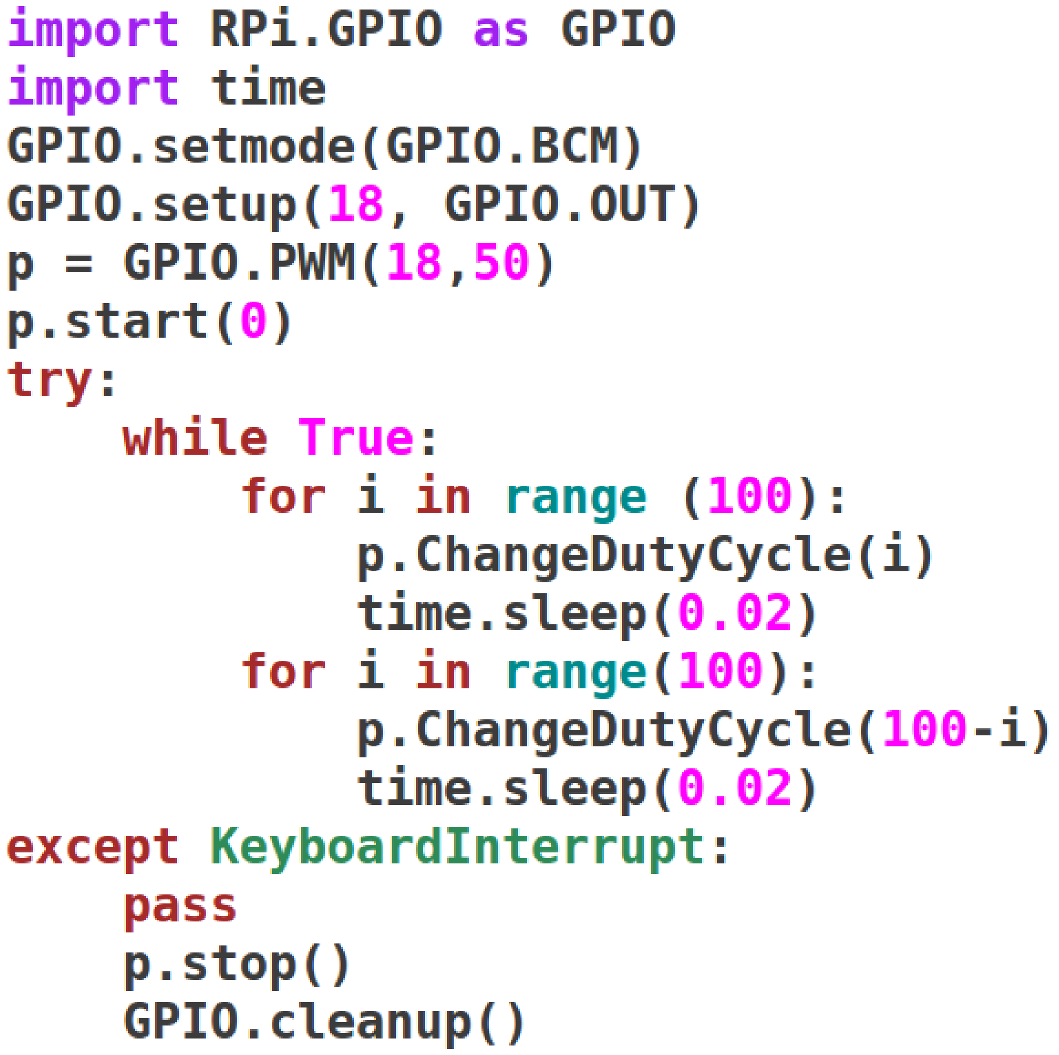
- Piezo buzzer is an electronic device commonly used to produce sound. Light weight, simple construction and low price make it usable in various applications like car/truck reversing indicator, computers, call bells etc.
- Piezo buzzer is based on the inverse principle of piezo electricity discovered in 1880 by Jacques and Pierre Curie.

Piezo Buzzer: Program



```python
import time
import RPi.GPIO as gpio
gpio.setwarnings(False)
gpio.setmode(gpio.BOARD)
gpio.setup(7,gpio.OUT)
try:
    while True:
        gpio.output(7,0)
        time.sleep(.3)
        gpio.output(7,1)
        time.sleep(.3)
except KeyboardInterrupt:
    gpio.cleanup()
    exit
```

Fading the LEDs

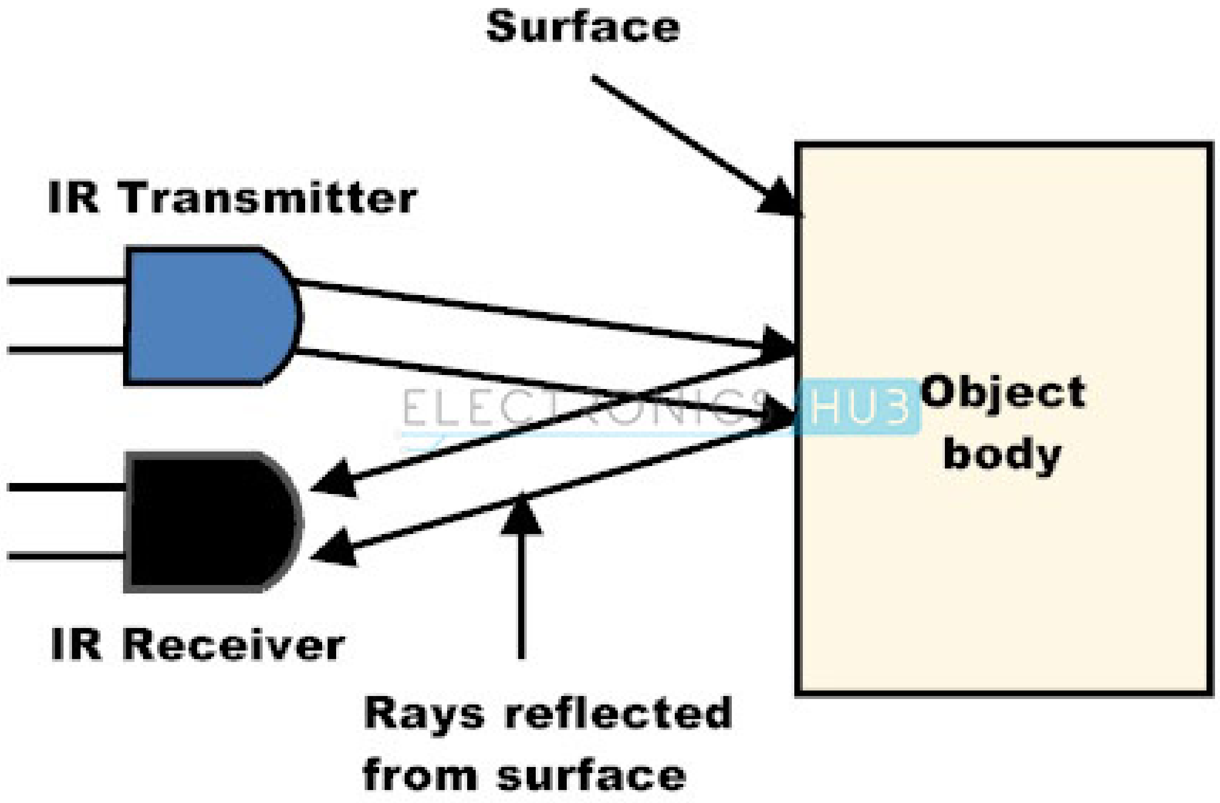


```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)
p = GPIO.PWM(18,50)
p.start(0)
try:
    while True:
        for i in range (100):
            p.ChangeDutyCycle(i)
            time.sleep(0.02)
        for i in range(100):
            p.ChangeDutyCycle(100-i)
            time.sleep(0.02)
except KeyboardInterrupt:
    pass
    p.stop()
    GPIO.cleanup()
```

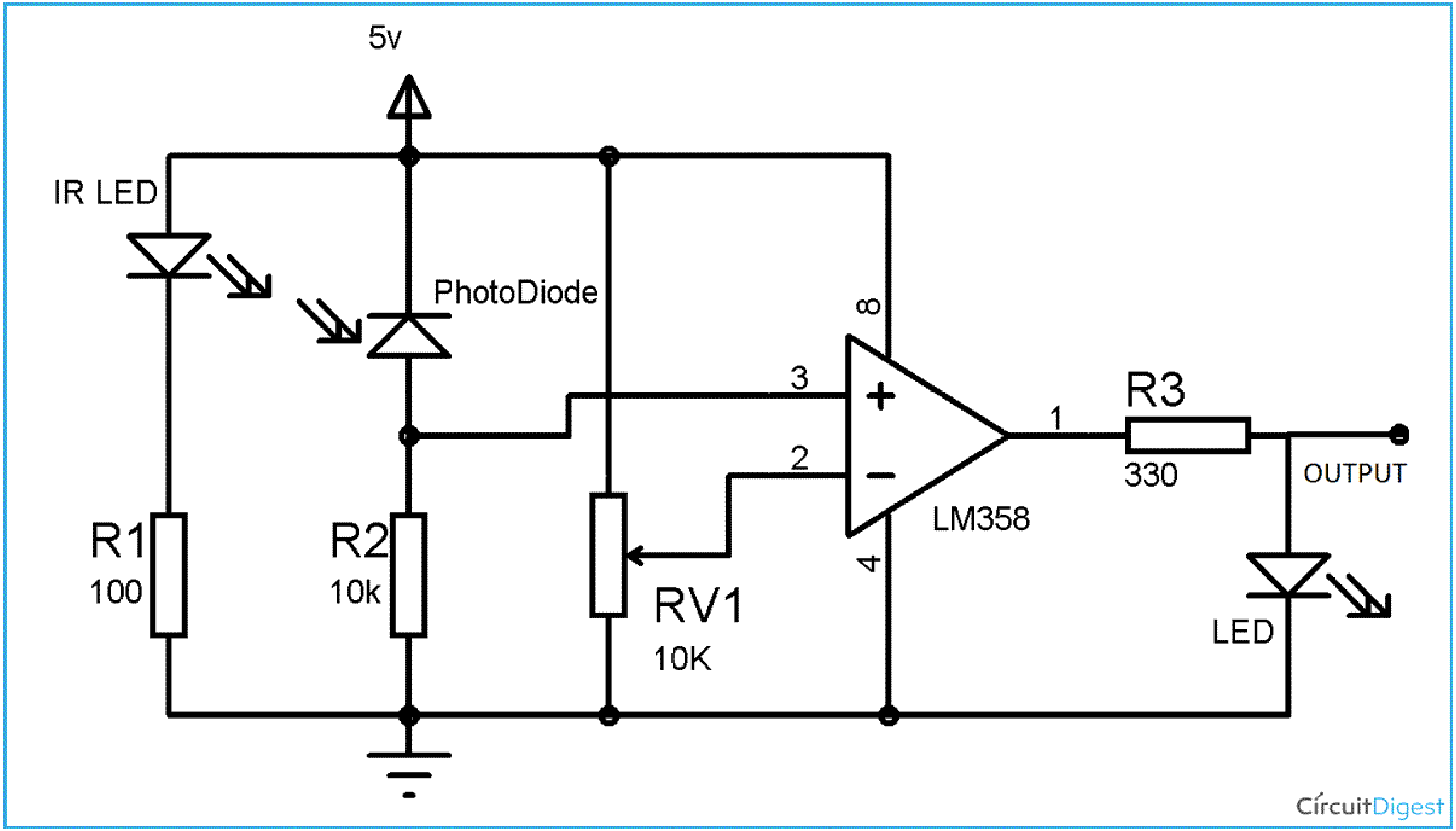
IR Sensor

- An infrared sensor is an electronic instrument which is used to sense certain characteristics of its surroundings by either emitting and/or detecting infrared radiation.
- Infrared sensors are also capable of measuring the heat being emitted by an object and detecting motion.
- Infrared waves are not visible to the human eye. In the electromagnetic spectrum, infrared radiation can be found between the visible and microwave regions.
- The infrared waves typically have wavelengths between 0.75 and 1000µm.

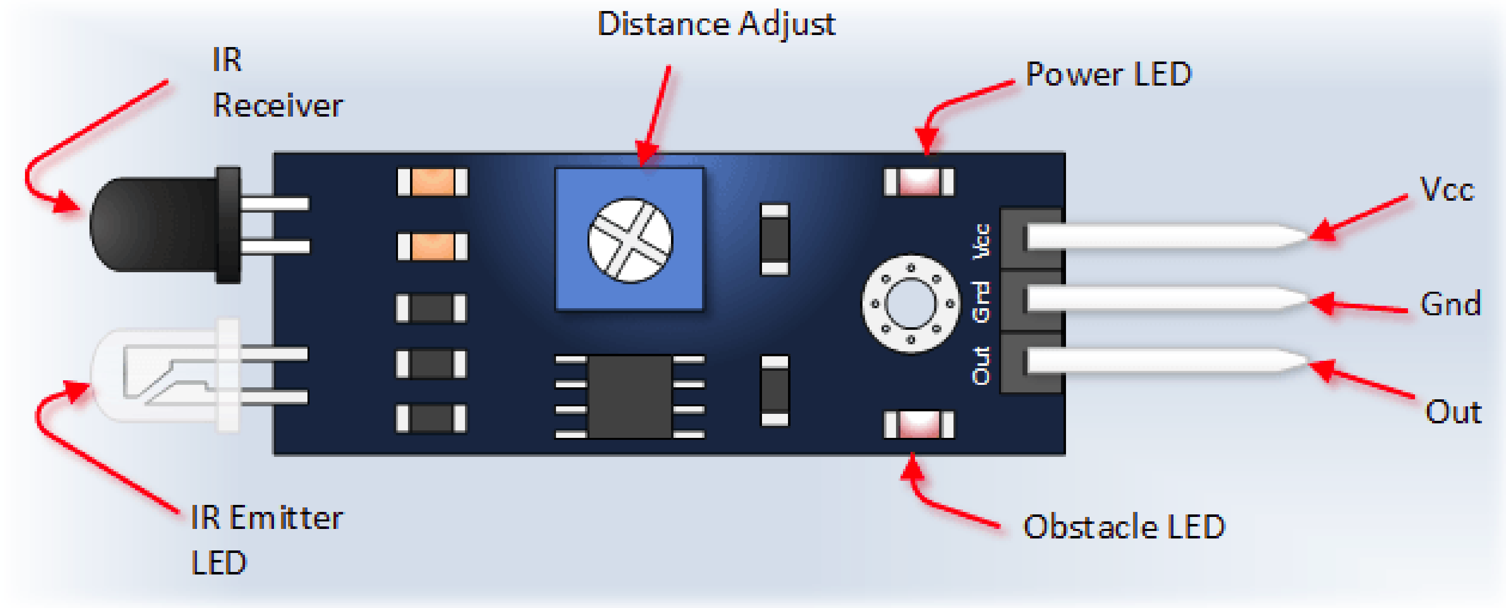
IR Sensor


Surface
IR Transmitter
Object body
IR Receiver
Rays reflected from surface
ELECTRONICS HUB

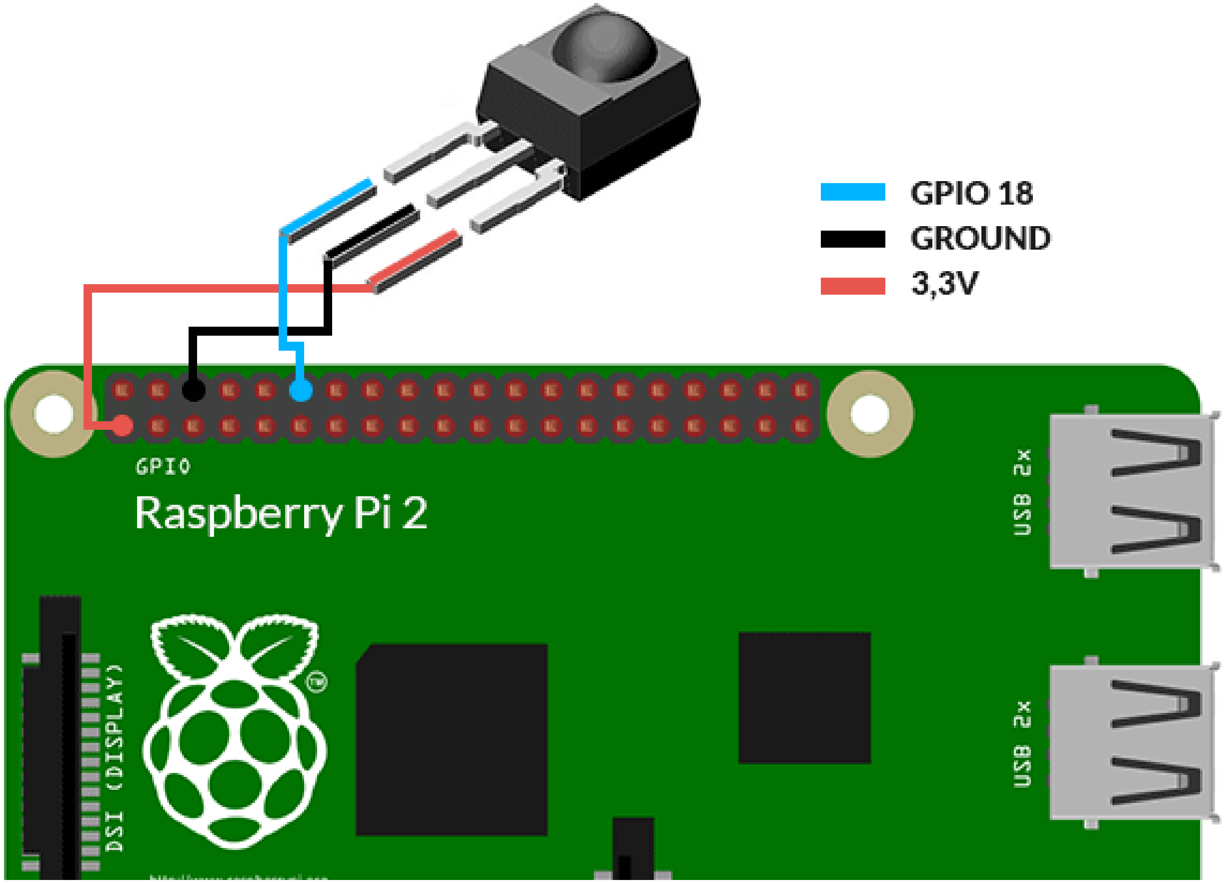
IR Sensor working



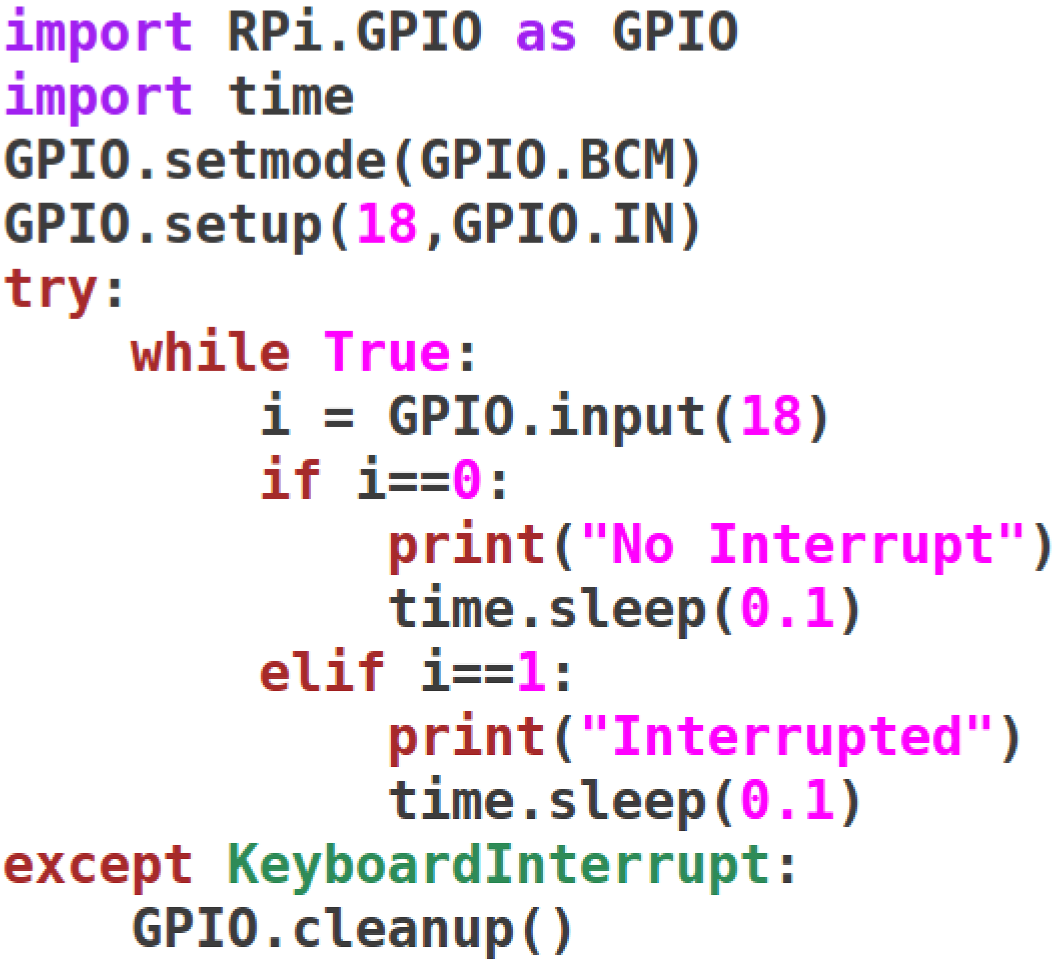
IR Sensor structure



IR Sensor connections



Program:



```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(18,GPIO.IN)
try:
    while True:
        i = GPIO.input(18)
        if i==0:
            print("No Interrupt")
            time.sleep(0.1)
        elif i==1:
            print("Interrupted")
            time.sleep(0.1)
except KeyboardInterrupt:
    GPIO.cleanup()
```

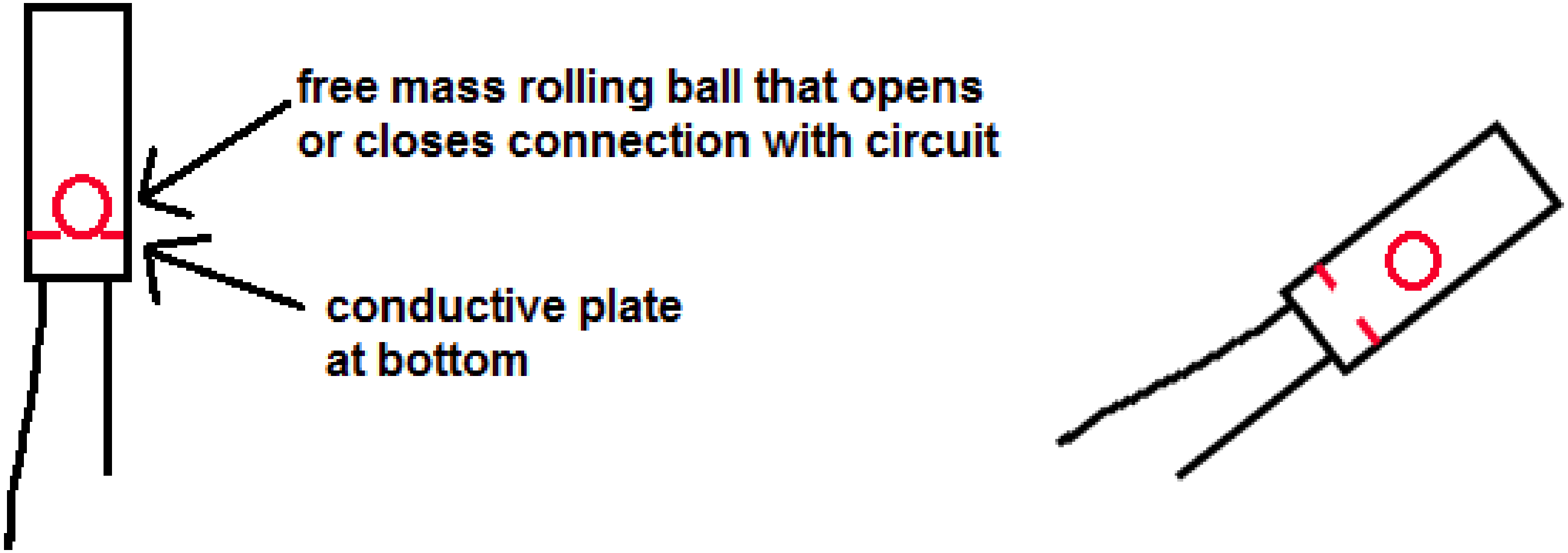
Tilt Sensor

- A tilt sensor is an instrument that is used for measuring the tilt in multiple axes of a reference plane.
- Tilt sensors measure the tilting position with reference to gravity, and are used in numerous applications.
- They enable the easy detection of orientation or inclination. Similar to mercury switches, they may also be known as tilt switches or rolling ball sensors.

Tilt Sensor Working

- A tilt sensor has a metallic ball that is designed to move the two pins of the instrument from the 'on' to the 'off' position, and vice versa, if the sensor reaches a pre-determined angle.
- Tilt sensors are the environment-friendly version of a mercury-switch.

Tilt Sensor Working



free mass rolling ball that opens or closes connection with circuit

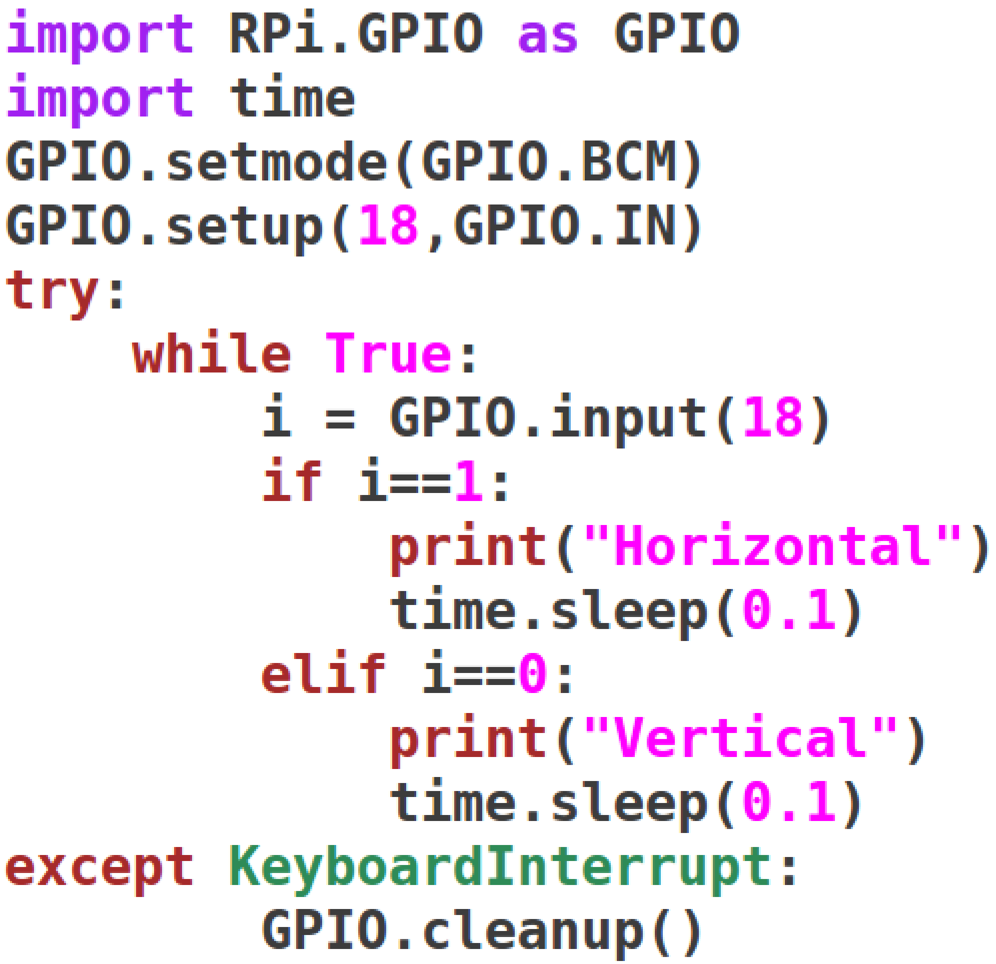
conductive plate at bottom

Closed circuit

Open circuit

Tilt Sensor Working

Program:

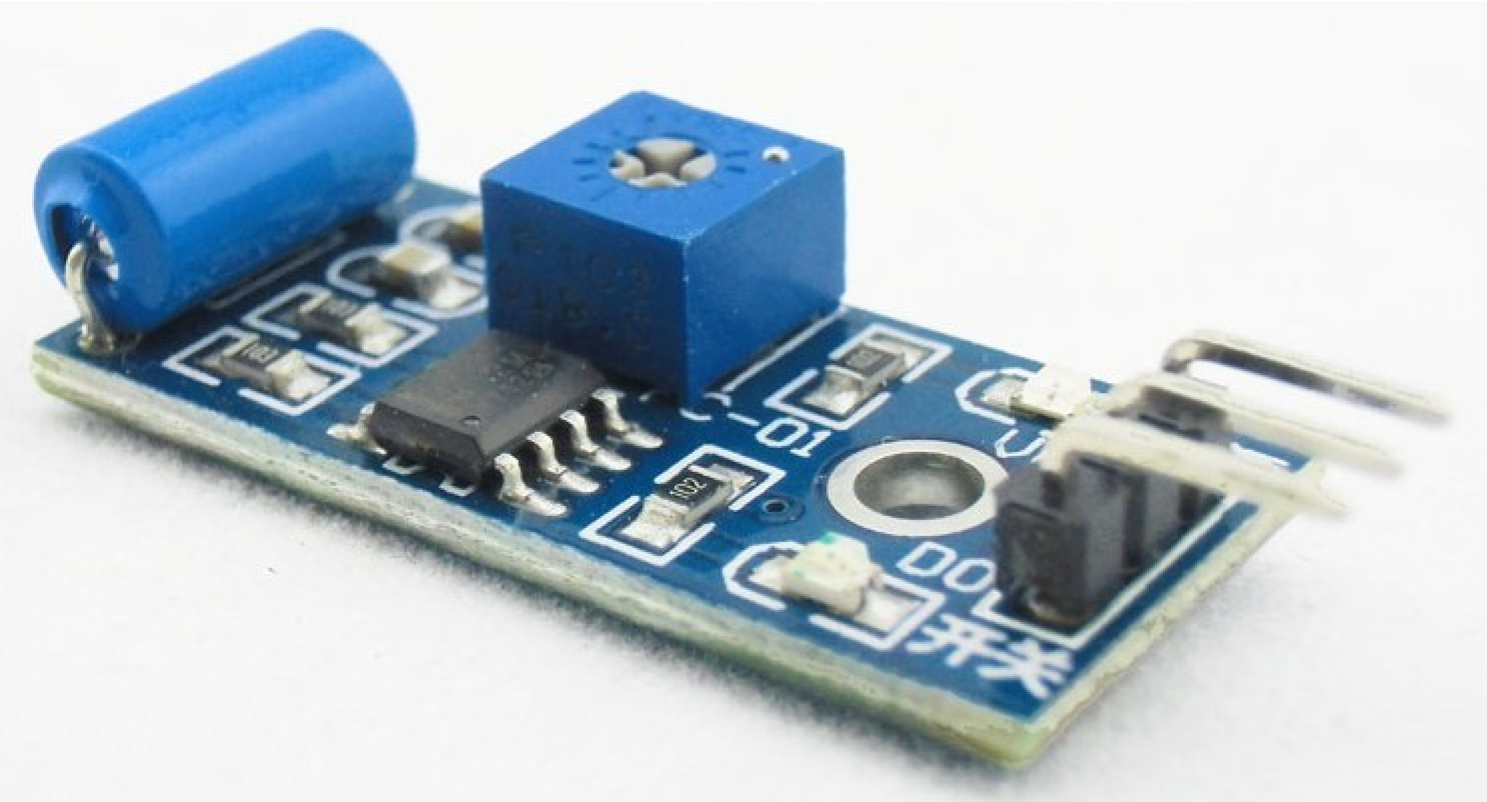


```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(18,GPIO.IN)
try:
    while True:
        i = GPIO.input(18)
        if i==1:
            print("Horizontal")
            time.sleep(0.1)
        elif i==0:
            print("Vertical")
            time.sleep(0.1)
except KeyboardInterrupt:
        GPIO.cleanup()
```

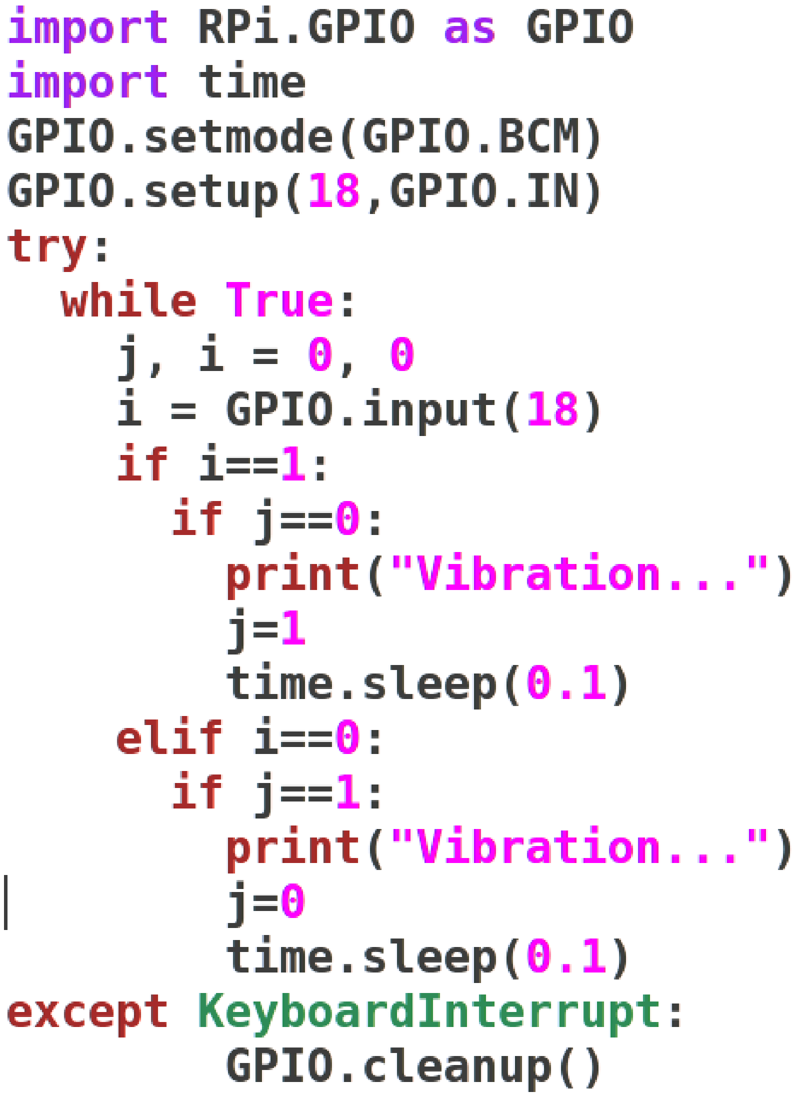
Applications

- To monitor the angle at which a mobile phone or tablet is held for the auto-rotate function
- To detect the position of hand-held game systems and in game controllers
- To indicate roll of boats, vehicles and aircraft
- To measure the angle at which a satellite antenna 'looks' toward a satellite
- To estimate the height of a tree or building using
- To measure the steepness of a ski slope
- As a warning system for the tilt angle of the surface cryogenic liquids during transportation
- To monitor laser levels and seismic activity.

Vibration Sensor



Program:



```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(18,GPIO.IN)
try:
  while True:
    j, i = 0, 0
    i = GPIO.input(18)
    if i==1:
      if j==0:
        print("Vibration...")
        j=1
        time.sleep(0.1)
    elif i==0:
      if j==1:
        print("Vibration...")
        j=0
        time.sleep(0.1)
except KeyboardInterrupt:
        GPIO.cleanup()
```

Temperature Sensor – DHT11

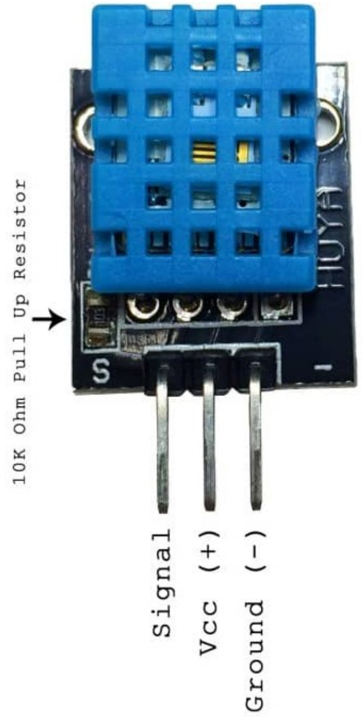
- The DHT11 temperature and humidity sensor is a nice little module that provides digital temperature and humidity readings. It’s really easy to set up, and only requires one wire for the data signal.
- These sensors are frequently used in remote weather stations, soil monitors, and home environment control systems. The programming is simple too, and many libraries and example code in both Python and C already exist.

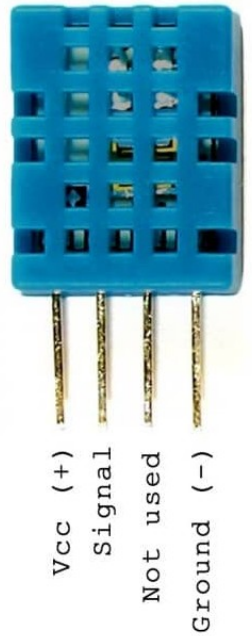
Temperature Sensor – DHT11

- The DHT11 contains a surface mounted NTC thermistor and a resistive humidity sensor. An IC on the back of the module converts the resistance measurements from the thermistor and humidity sensor into digital outputs of degrees Celsius and Relative Humidity.

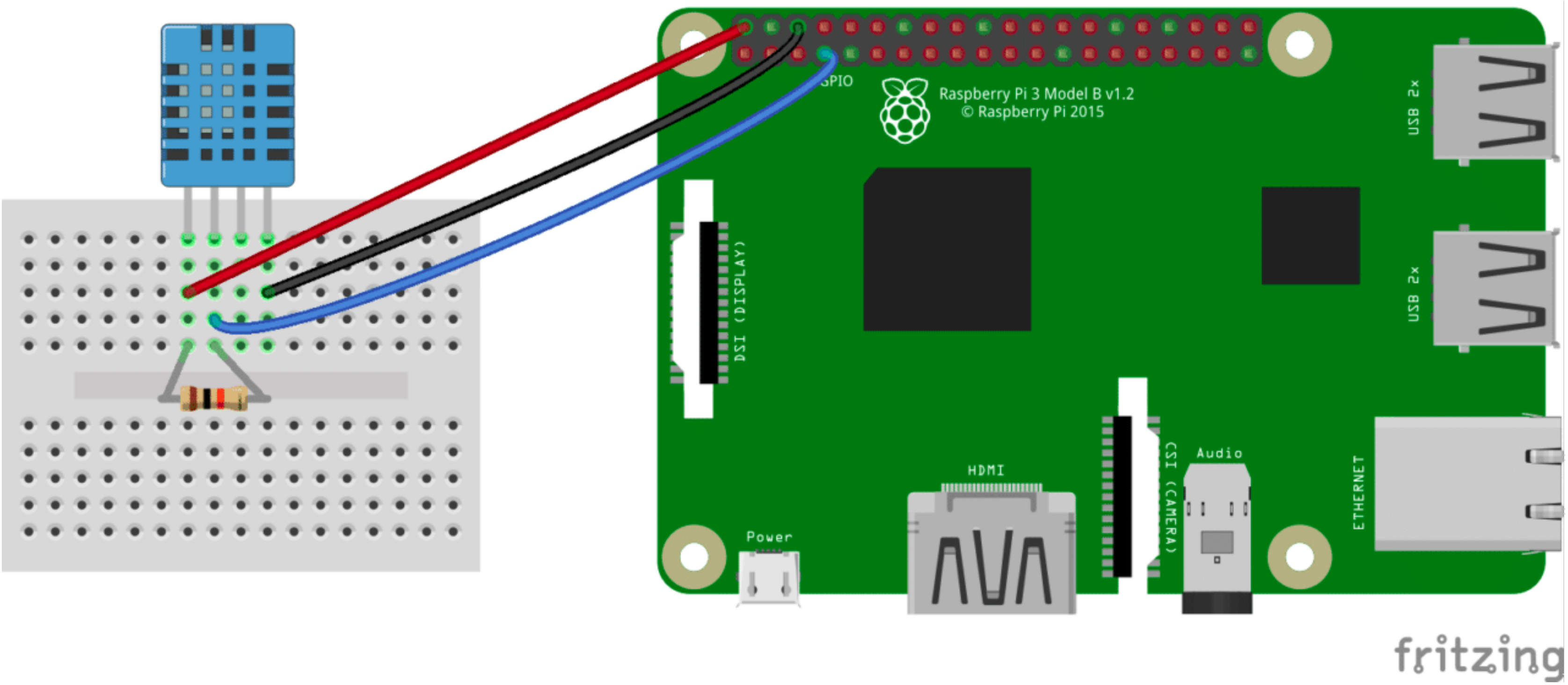
Temperature Sensor – DHT11

- There are two variants of the DHT11 you’re likely to come across. One is a four pin stand alone module, and the other is a three pin, PCB mounted module. The pinout is different for each one, so connect the DHT11 according to which one you have:





Temperature Sensor – DHT11



Programming the DHT11 in Python

- We will be using the Adafruit DHT11 Python library. We can download the library using Git, so if you don't have Git installed on your Pi already, enter this at the command prompt:

  sudo apt-get install git-core

- Note: If you get an error installing Git, run sudo apt-get update and try it again.

Install the Adafruit DHT11 library

1. Enter this at the command prompt to download the library:

```
git clone https://github.com/adafruit/Adafruit_Python_DHT.git
```

2. Change directories with:

```
cd Adafruit_Python_DHT
```

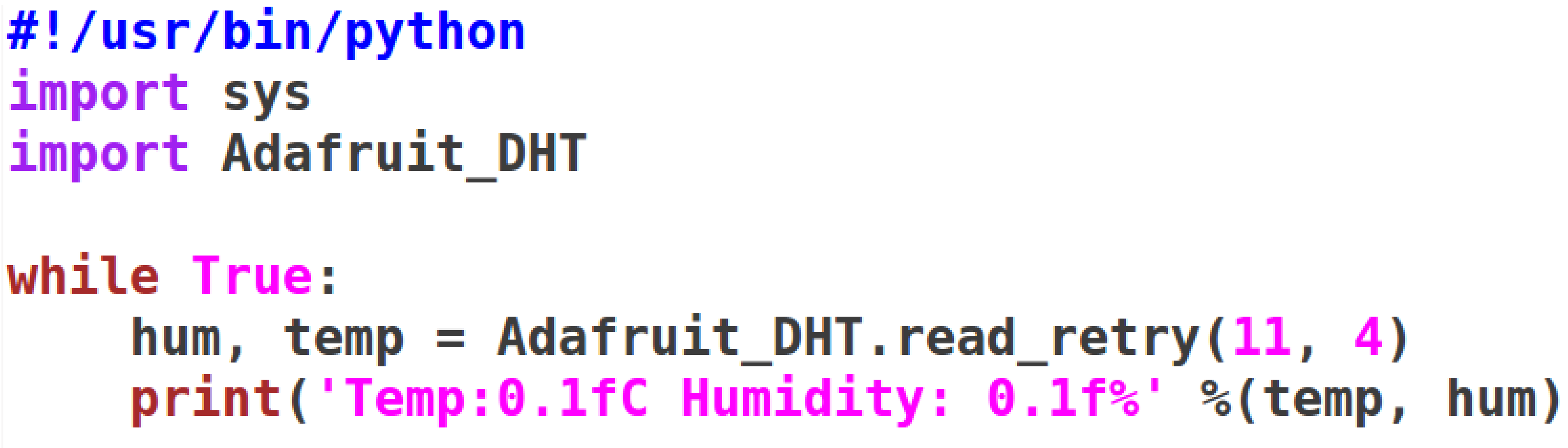
3. Enter this:

```
sudo apt-get install build-essential python-dev
```

4. Install the library with:

```
sudo python setup.py install
```

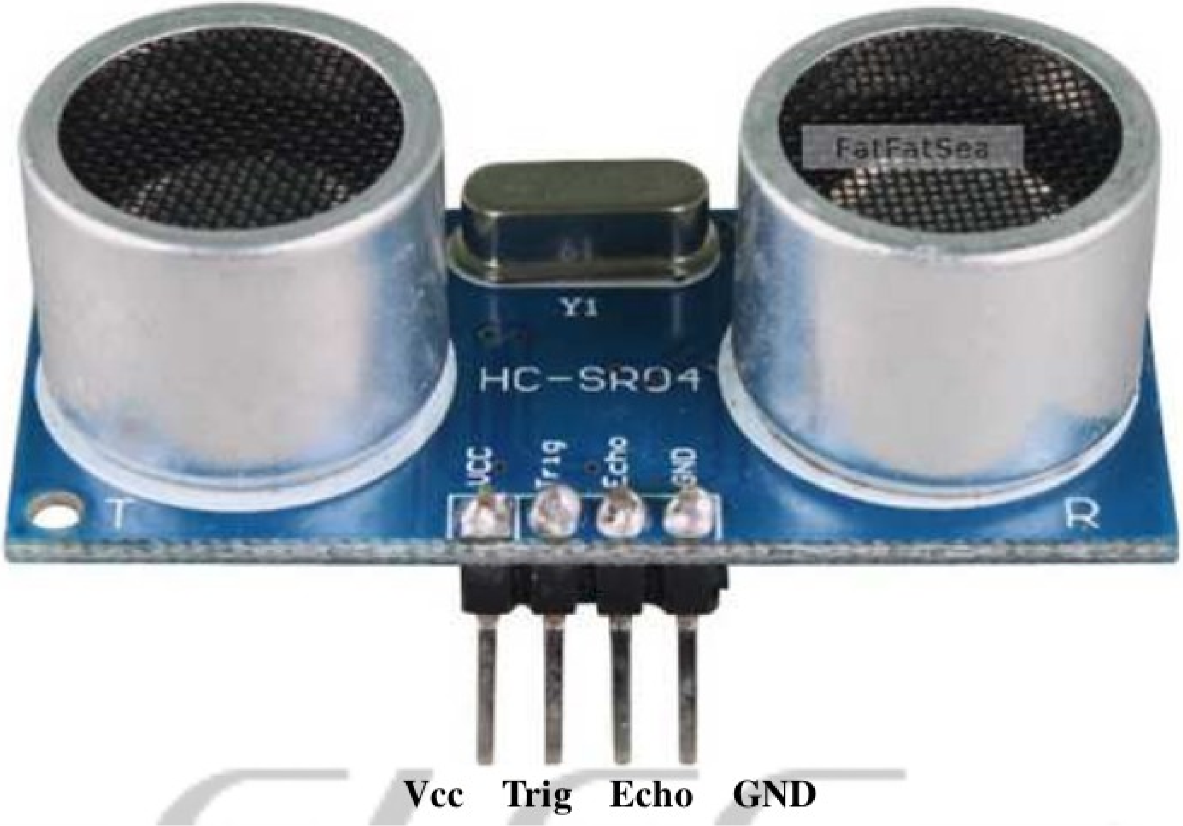
Program:



```
#!/usr/bin/python
import sys
import Adafruit_DHT

while True:
    hum, temp = Adafruit_DHT.read_retry(11, 4)
    print('Temp:0.1fC Humidity: 0.1f%' %(temp, hum)
```

Ultrasonic Distance Sensor



Ultrasonic Distance Sensor

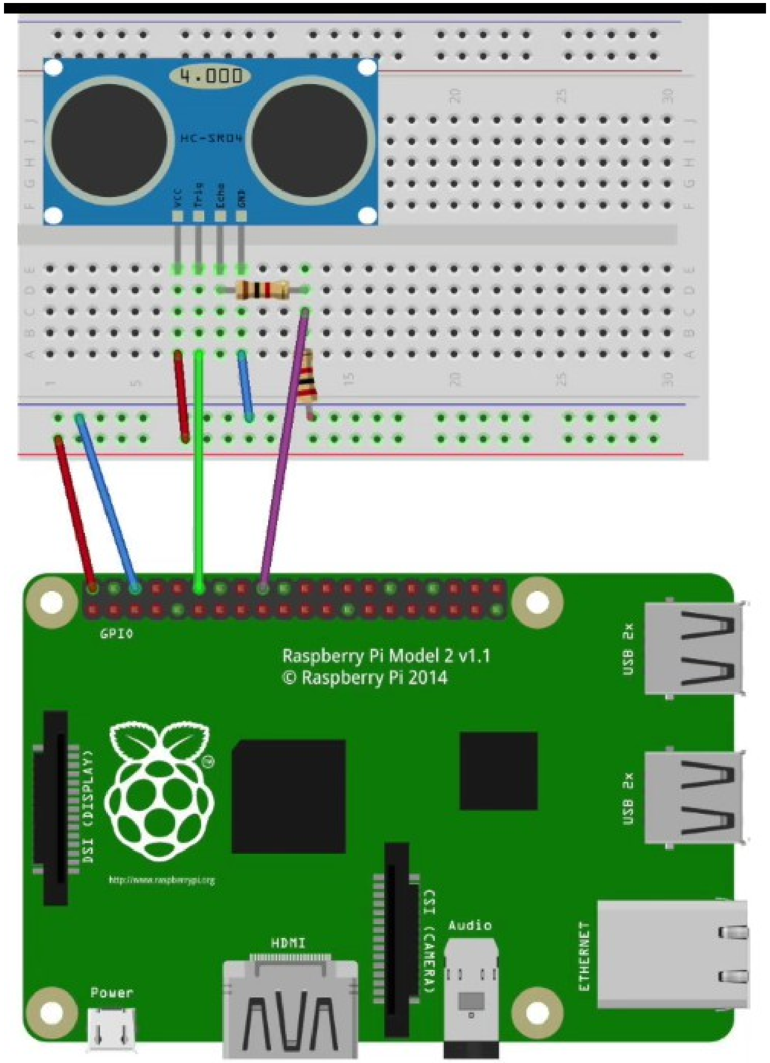
- Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:

  (1) Using IO trigger for at least 10us high level signal,

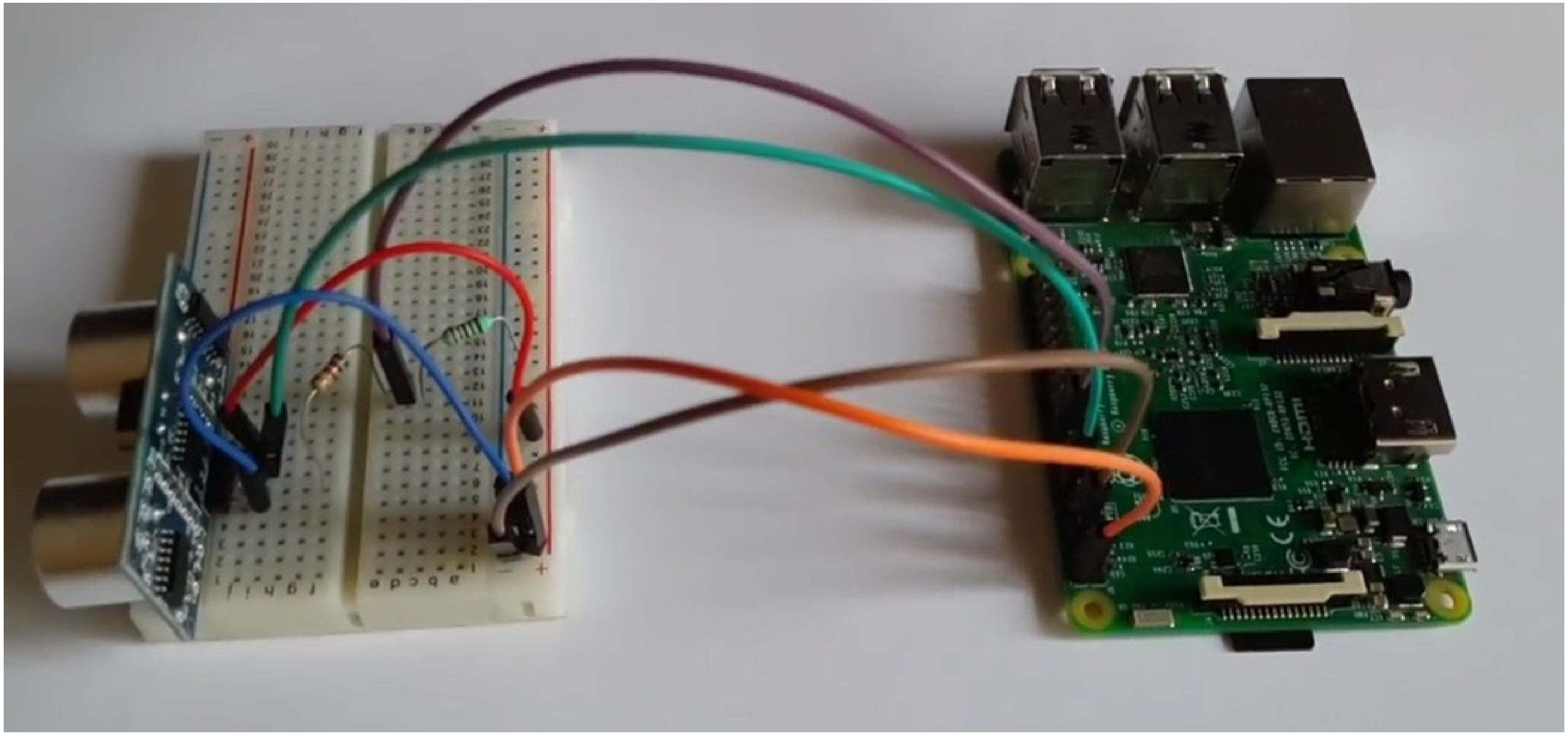
  (2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.

  (3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.
- Test distance = (high level time×velocity of sound (340M/S) / 2

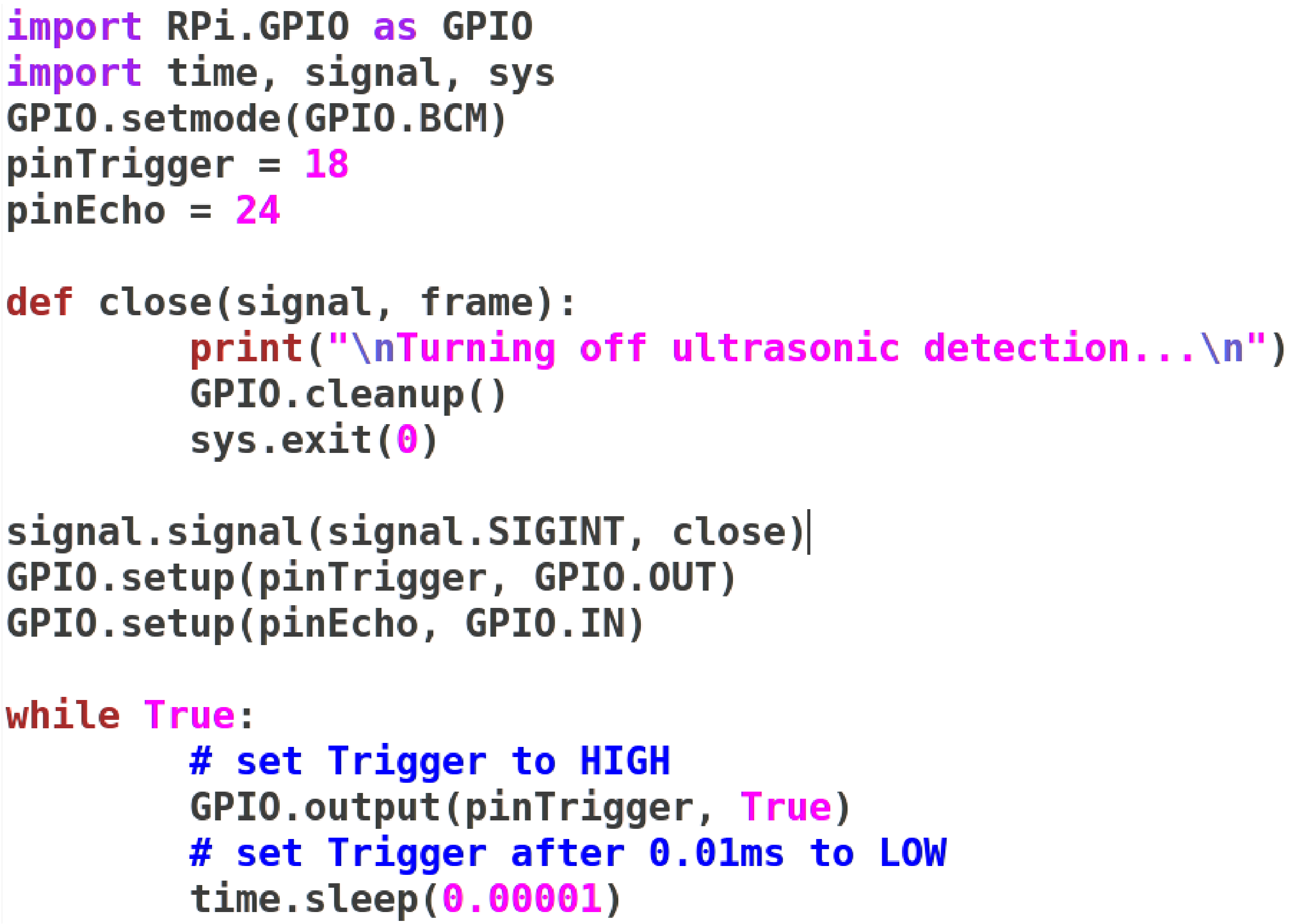
Connections



Connections



Program: part-1



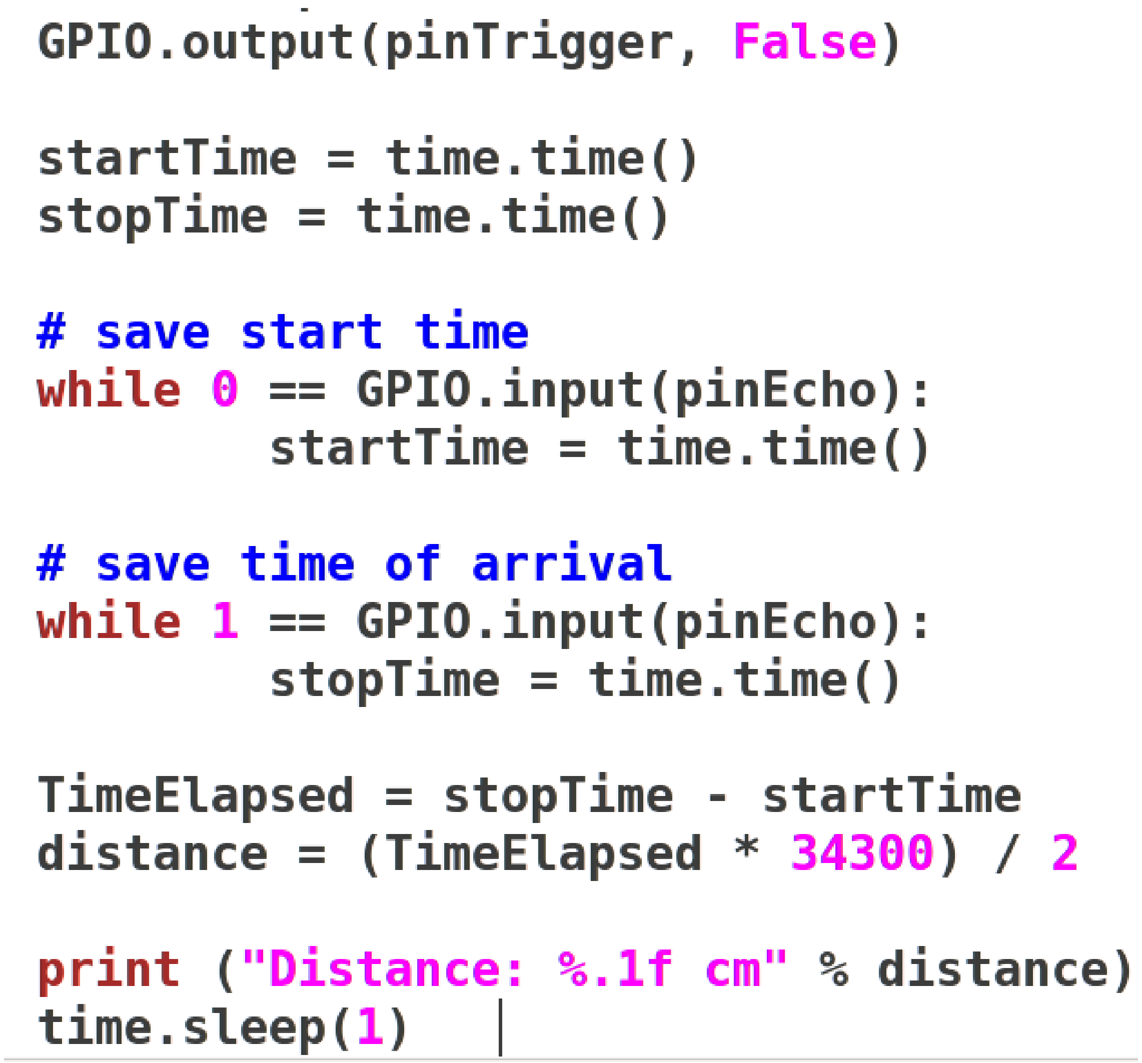
```
import RPi.GPIO as GPIO
import time, signal, sys
GPIO.setmode(GPIO.BCM)
pinTrigger = 18
pinEcho = 24

def close(signal, frame):
        print("\nTurning off ultrasonic detection...\n")
        GPIO.cleanup()
        sys.exit(0)

signal.signal(signal.SIGINT, close)
GPIO.setup(pinTrigger, GPIO.OUT)
GPIO.setup(pinEcho, GPIO.IN)

while True:
        # set Trigger to HIGH
        GPIO.output(pinTrigger, True)
        # set Trigger after 0.01ms to LOW
        time.sleep(0.00001)
```

Program: part-2



```
GPIO.output(pinTrigger, False)

startTime = time.time()
stopTime = time.time()

# save start time
while 0 == GPIO.input(pinEcho):
        startTime = time.time()

# save time of arrival
while 1 == GPIO.input(pinEcho):
        stopTime = time.time()

TimeElapsed = stopTime - startTime
distance = (TimeElapsed * 34300) / 2

print ("Distance: %.1f cm" % distance)
time.sleep(1)
```

Servo Motor

- A Servo Motor is a combination of DC motor, position control system and gears. Servos have many applications in the modern world and with that, they are available in different shapes and sizes. We will be using SG90 Servo Motor which is one of the popular and cheapest one. SG90 is a 180 degree servo. So with this servo we can position the axis from 0-180 degrees.
- A Servo Motor mainly has three wires, one is for positive voltage, another is for ground and last one is for position setting. The Red wire is connected to power, Brown wire is connected to ground and Yellow wire (or WHITE) is connected to signal.

Servo Motor

Servo Motor

- In servo, we have a control system which takes the PWM signal from Signal pin. It decodes the signal and gets the duty ratio from it.
- After that, it compares the ratio to the predefined positions values. If there is a difference in the values, it adjusts the position of the servo accordingly.
- So the axis position of the servo motor is based on the duty ratio of the PWM signal at the Signal pin.

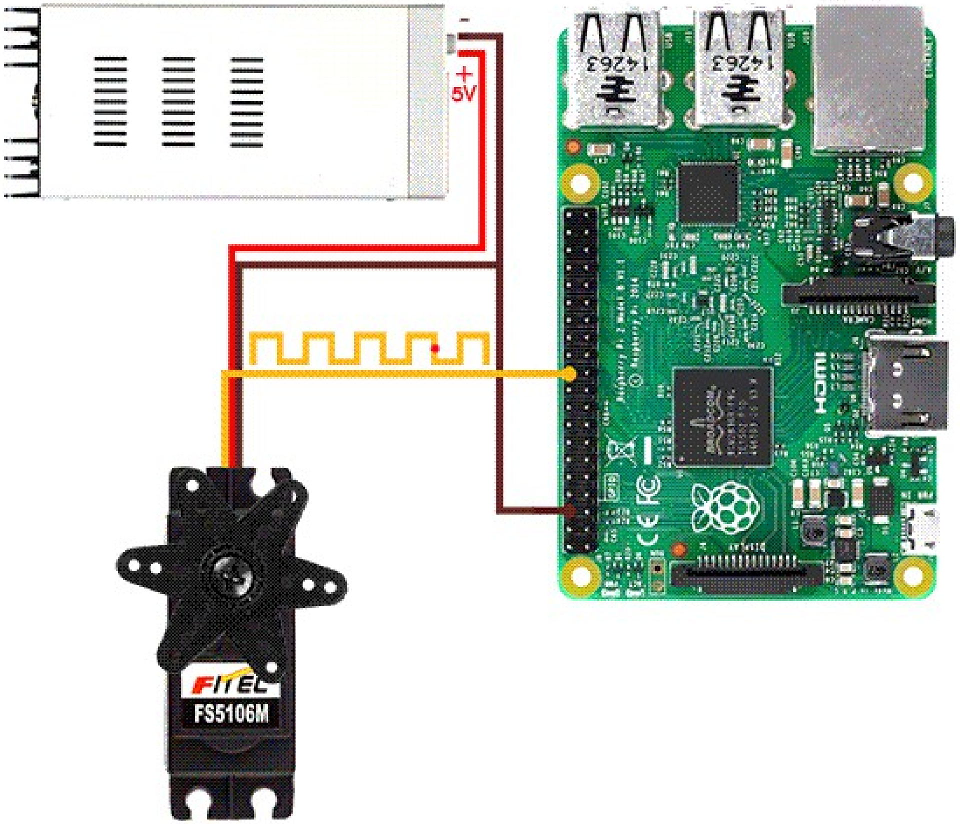
Servo Motor

- A Servo Motor is a combination of DC motor, position control system and gears. Servos have many applications in the modern world and with that, they are available in different shapes and sizes. We will be using SG90 Servo Motor which is one of the popular and cheapest one. SG90 is a 180 degree servo. So with this servo we can position the axis from 0-180 degrees.
- A Servo Motor mainly has three wires, one is for positive voltage, another is for ground and last one is for position setting. The Red wire is connected to power, Brown wire is connected to ground and Yellow wire (or WHITE) is connected to signal.

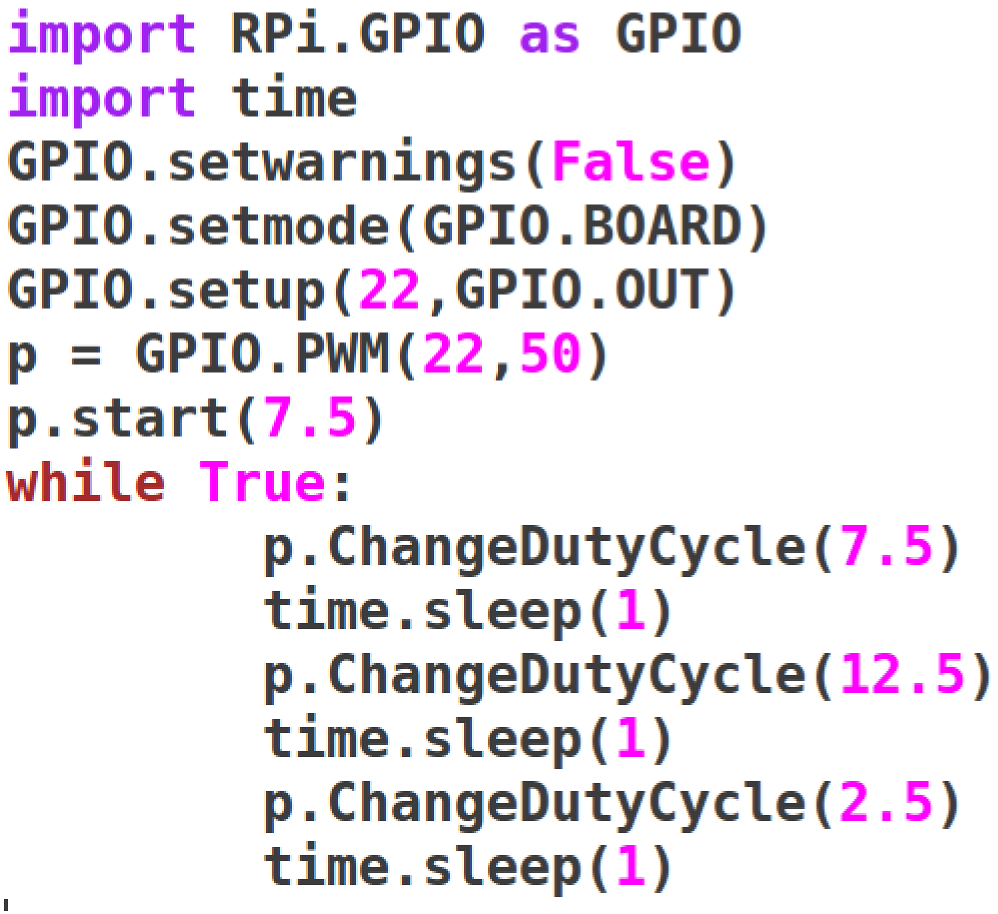
Duty cycle

Position	Duty cycle
0 degrees	2.5
90 degrees	7.5
180 degrees	12.5

Servo Motor



Servo Motor Program



```
import RPi.GPIO as GPIO
import time
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(22,GPIO.OUT)
p = GPIO.PWM(22,50)
p.start(7.5)
while True:
        p.ChangeDutyCycle(7.5)
        time.sleep(1)
        p.ChangeDutyCycle(12.5)
        time.sleep(1)
        p.ChangeDutyCycle(2.5)
        time.sleep(1)
```

Thank you

Web Resources
http://mitu.co.in
http://tusharkute.com

Blogs
http://digitallocha.blogspot.in
http://kyamputar.blogspot.in

tushar@tusharkute.com