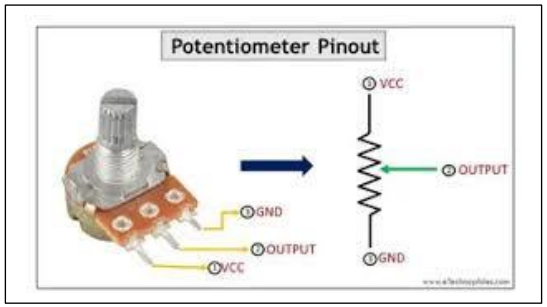
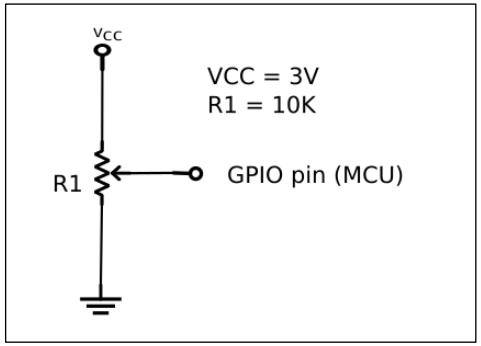
Sensor Implementation – Python (MicroPython)

A. Analog

a. **Potentiometer**

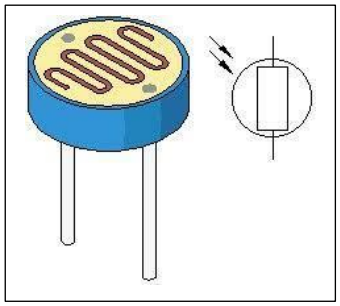
i. Connecting a potentiometer to the MCU.
ii. Reading the values from the potentiometer.
iii. Store the values in a data structure.
iv. Python Library : microbit.pin.read_analog()

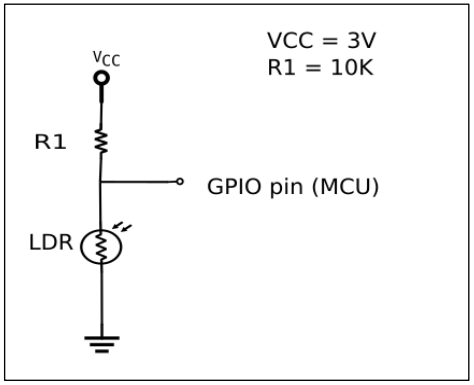




b. **LDR**

i. Connecting a LDR to the MCU.
ii. Reading the values from the LDR.
iii. Store the values in a data structure.
iv. Python Library : microbit.pin.read_analog()

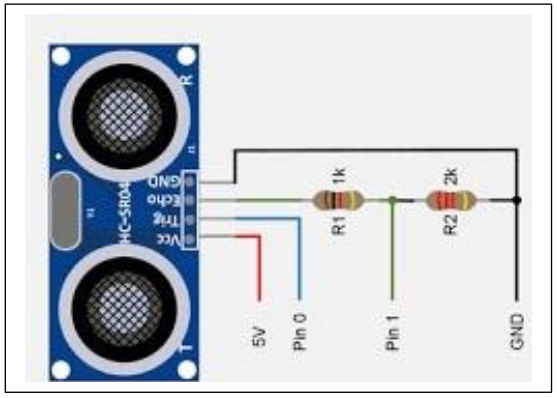


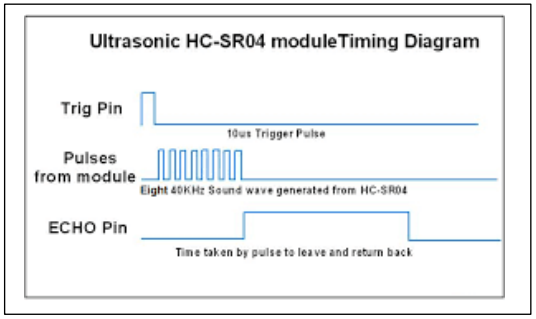


B. Digital

a. **HC_SR04**

i. Connecting a HC_SR04 to the MCU.
ii. Reading the values from the HC_SR04.
iii. Store the values in a data structure.
iv. Python Library : microbit.write_digital()
microbit.read_digital()





POT – Python Code

```
# Import libraries
from microbit import *


#Configure circuit/pins
led1 = pin16
pot = pin0


# Preping circuit/pins
led1.write_digital(0)
sleep(500)


#Reading Potentiometer
while True:
    potVal = pot.read_analog()
    v = potVal/1023*3.2
    print(potVal)
    print(v)
    sleep(100)
```

LDR – Python Code

```
# Import libraries
from microbit import *


## Configuring circuit/pins
led_w = pin16
ldr = pin0
buzz = pin15
led_w.write_digital(1)


## Analog read
while True:
    potVal = ldr.read_analog()
    print(potVal)
    if potVal >= 295:
        print("Opstruction")
        buzz.write_digital(1)
        sleep(150)
        buzz.write_digital(0)
    else:
        print("Clear")
    sleep(50)
```

HC_SR04 – Python Code

```
# Import libraries
from microbit import *
from machine import time_pulse_us


# Continuous Ultra Sonic Ranging
while True:
    # Output a pulse to trigger ultrasonic burst
    trig.write_digital(1)
    sleep(10)
    trig.write_digital(0)
    # Masure the input echo pulse in microseconds
      convert to seconds
    micros = time_pulse_us(echo, 1)
    t_echo = micros / 1000000
dist_cm = (t_echo / 2) * 34300
```