Priyadarshini College of Engineering, Nagpur

Department of Electronics & Telecommunication Engineering

Title : Lab Manual of Internet of Things Lab

Semester: VI Semester

By

Mr. Vishal Vasantrao Panchbhai[1]& Mr. Chandrashekhar Namdeo Bhoyar[2]

[1,2]Assistant Professor

Priyadarshini College of Engineering, Nagpur
Department of Electronics & Telecommunication Engineering
Experiment List of Lab Manual of Internet of Things
(Sixth Semester)

Index

Experiment Number - I

Aim: Interface a light emitting diode with arduino uno and write a code to turn on light emitting diode for one second and turn off light emitting diode for 2 seconds continuously.

Requirement: Arduino uno board, LED, Buzzer, 220 ohm resistor, breadboard, and connecting wires

Circuit Diagram:
Make the connection of circuit as shown in following figure

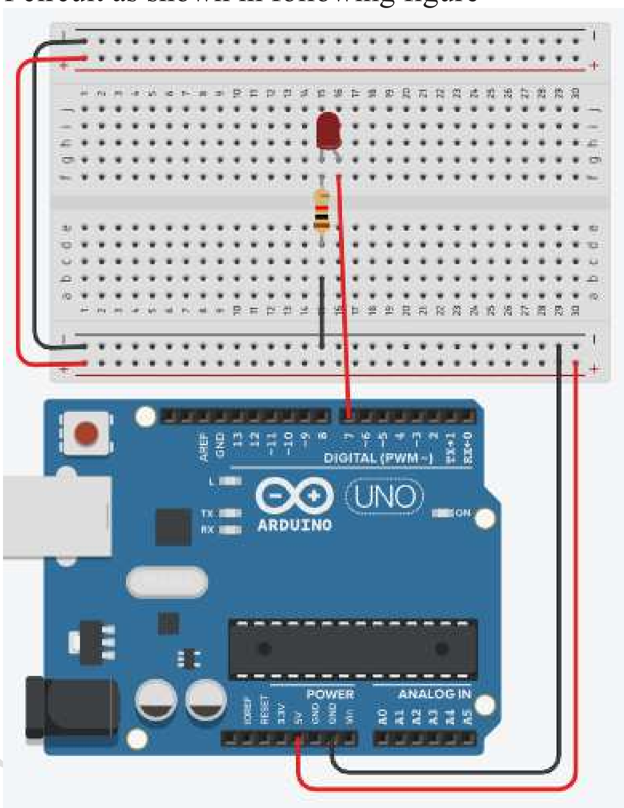

Figure 1: Circuit schematic of LED interfacing with Arduino Uno

Algorithm:
1) Initialize digital I/O pin 7 as output pin
2) Turn on LED for 1 second
3) Turn off LED for next 2 second
4) Jump to the step number 2

Program:

```
const int LED_digitalpin= 7;
void setup()
{
//define digital pin 7 of arduino uno as output
pinMode(LED_digitalpin, OUTPUT);
}

void loop()
{
 // turn on LED and generate delay of one second
digitalWrite(LED_digitalpin, HIGH);
delay(1000);

// turn off  LED and generate delay of two second
digitalWrite(LED_digitalpin, LOW);
delay(2000);
}
```


Result: We have successfully written and execute arduino program for LED blinking.

Questions

1) **How many digital input/ output pins and analog input pins available in arduino uno board?**
2) **What are different features of ATMEGA 328P controller?**

Experiment Number - II

Aim: Interface LM35 temperature sensor with arduino uno & write a code to monitor temperature.

Requirement: Arduino uno board, LM35 temperature sensor, connecting wires and breadboard

Circuit Diagram:
Make the connection of circuit as shown in following figure

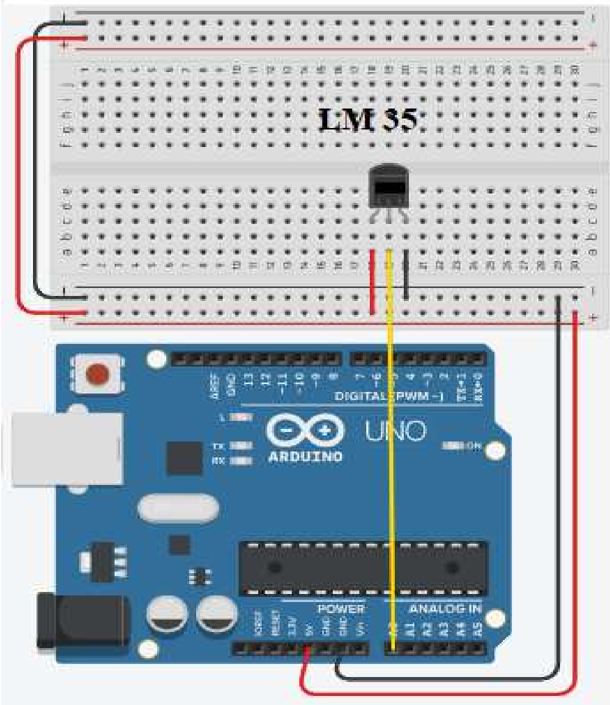

Figure 2: Circuit schematic of LM35 (temperature sensor) interfacing with Arduino Uno

Algorithm:
1) Define analog input pin A0 as temperature sensor pin
2) Initialize serial communication with baud rate 115200
3) Read the value of temperature sensor using analogRead( ) function and store it in variable
4) Convert the sensor reading in celsius format using formula (sensor reading x (5.0 / 1024.0))
5) Convert the voltage reading in Fahrenheit format using formula ((temperature in Celsius x 9.0) / 5.0) + 32.0
6) Generate delay of one second and jump to step no. 3

Program:

```
#define temp_sensor A0

void setup()
{
// Set the transmission rate of the serial port to 115200 bits per second.
  Serial.begin(115200);
 }

void loop()
{
 //store LM35 sensor voltage reading in variable
 int sensor_reading = analogRead(temp_sensor);

 // Convert LM35 sensor output to voltage
 float convert_to_volt = sensor_reading * (5.0 / 1024.0);

 // Convert the voltage reading into Celsius and Fahrenheit
 float temp_celcius = convert_to_volt * 100;
 float temp_fahre = (temp_celcius * 9.0 / 5.0) + 32.0;

 // display temperature reading in different format on serial port
 Serial.print("Temperature in Celsius = ");
 Serial.print(temp_celcius);
 Serial.print("");
 Serial.print("Temperature in Fahrenheit = ");
 Serial.println(temp_fahre);

//generate delay for one second
 delay(1000);
}
```


Result: We have successfully written and execute arduino program for monitoring temperature using LM35.

Questions

1) **Define sensor and explain types of sensors.**
2) **What are different temperature sensors available in market? Compare them**

Experiment Number - III

Aim: Interface HC-SR04 ultrasonic sensor with arduino uno and write a code to calculate distance of an object.

Requirement: Arduino uno board, ultrasonic sensor (HC-SR04), breadboard and connecting wires

Circuit Diagram:
Make the connection of circuit as shown in following figure

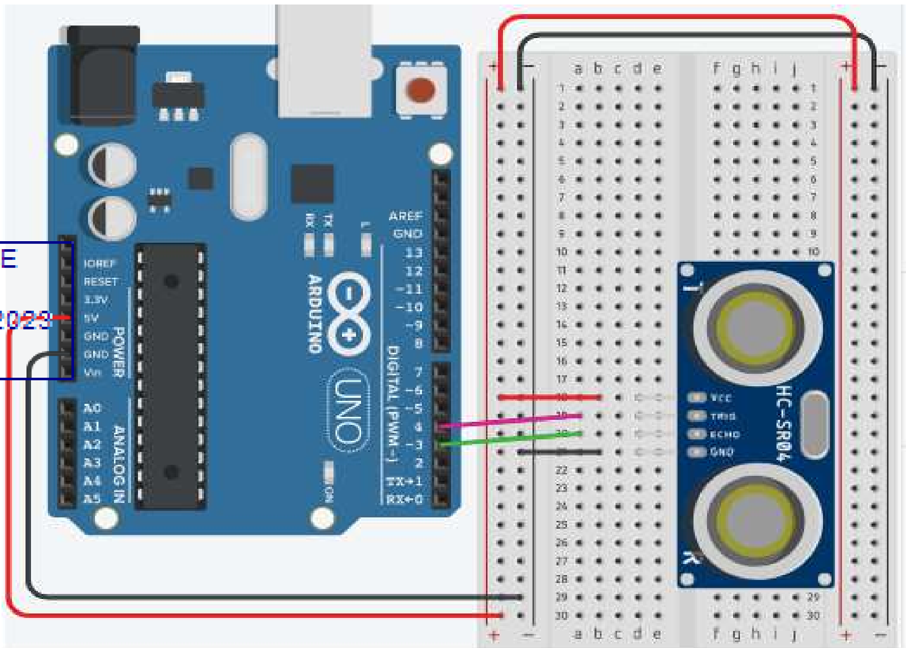

Figure 3: Circuit schematic of ultrasonic sensor interfacing with Arduino Uno

Algorithm:

1) Define digital I/O pin 4 and 3 of arduino as trigger and echo pin
2) Initialize serial port at 115200 baud rate
3) Initialize echo pin as input pin
4) Initialize trigger pin as output
5) Send logic low on trigger pin for 5 microsecond
6) Send logic high on trigger pin for 10 microsecond
7) Send logic low on trigger pin for 5 microsecond
8) Read the echo pin using pulseIn( ) function and store it in variable(eg. roundtrip_time)
9) Find the distance using formula (duration*.0343)/2 and store answer in variable (eg. obj_dist)
10) Send above reading on serial port
11) Jump to the step number 5.

Program:

```
const int trigger_digital_pin = 4;
const int echo_digital_pin= 3;

void setup()
{
 //initialize serial port at 115200 baud rate
 Serial.begin(115200);

// Set the echo pin as an input and the trigger pin as an output.
 pinMode(echo_digital_pin, INPUT);
 pinMode(trigger_digital_pin, OUTPUT);
 }

void loop()
{
//Generate 10 microsecond pulse on trigger pin
  digitalWrite(trigger_digital_pin, LOW);
  delayMicroseconds(5);
  digitalWrite(trigger_digital_pin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigger_digital_pin, LOW);
  delayMicroseconds(5);

  float roundtrip_time = pulseIn(echo_digital_pin, HIGH);
  float divisor= 0.0343 /(2.0*2.54);
  float obj_dist = roundtrip_time /divisor;

// display object distance from sensor in inches and generate delay of 1 second
  Serial.print("Object Distance = ");
  Serial.print(obj_dist);
  Serial.println(" inch");
  delay(1000);
  }
```


Result: We have successfully written and execute arduino program for calculating distance of an object using ultrasonic sensor.

Questions

1) **Define baud rate. What is the standard baud rate value generally used?**
2) **Compare ultrasonic sensor HC-SR04 and MaxBotix MaxSonar sensor.**

Experiment Number - IV

Aim: Interface bulb with arduino uno through relay circuit and write a code to turn on and off the bulb continuously for one second each.

Requirement: Arduino uno board, 5V relay circuit, bulb, bulb holder and connecting wires

Circuit Diagram:
Make the connection of circuit as shown in following figure

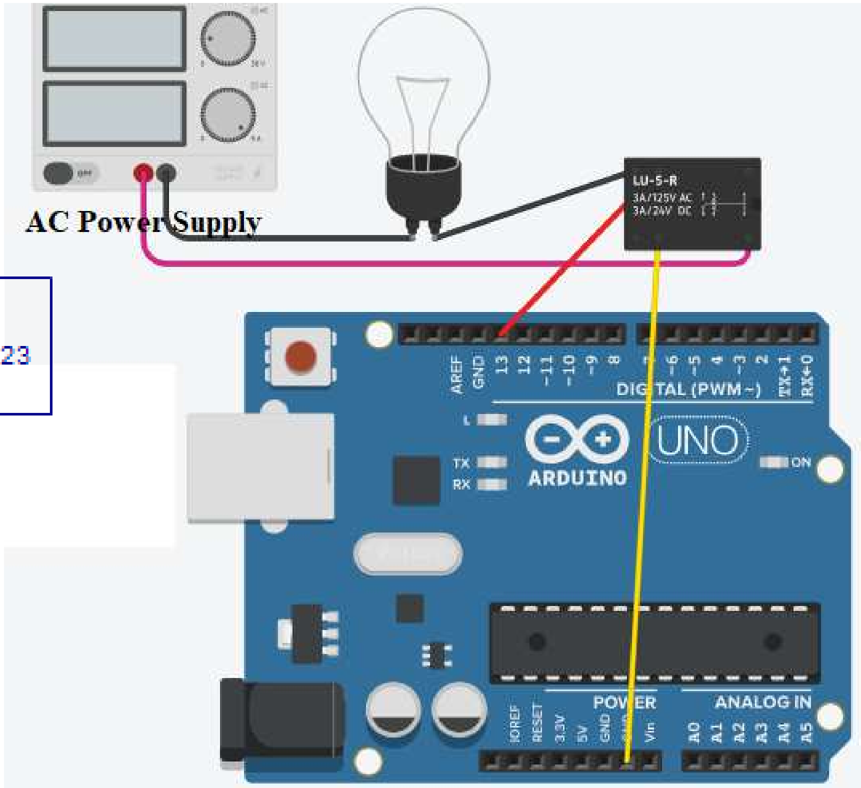

Figure 4: Circuit schematic of bulb interfacing using relay with Arduino uno

Algorithm:
1) Initialize relay to be activated through pin number 13 of board
2) Initialize digital I/O pin 13 as output pin
3) Turn on bulb which works on AC power supply for 1 second
4) Turn off bulb which works on AC power supply next 1 second
5) Jump to the step number 3

Program:

```
const int relay_digital_pin= 13;    // initialize relay to be activated through pin number 13 of
board

void setup()
{
// define digital pin 13 of arduino uno as output
    pinMode(relay_digital_pin, OUTPUT);
}

void loop()
{
// turn on bulb and generate delay of one second
  digitalWrite(relay_digital_pin, HIGH);
  delay(1000);

// turn off bulb and generate delay of one second
  digitalWrite(relay_digital_pin, LOW);
  delay(1000);
}
```


Result: We have successfully written and execute arduino program for turning on and off AC supply power bulb through relay switch

Questions

1) List the different types of relay.

2) What is optocoupler? List the application of optocupler.

Experiment Number - V

Aim: Interface EM 18 RFID module with arduino uno and write a code to read RFID tag number and display it on serial port.

Requirement: Arduino uno board, EM18 RFID reader, passive RFID tags and connecting wires

Circuit Diagram:

Make the connection of circuit as shown in following figure

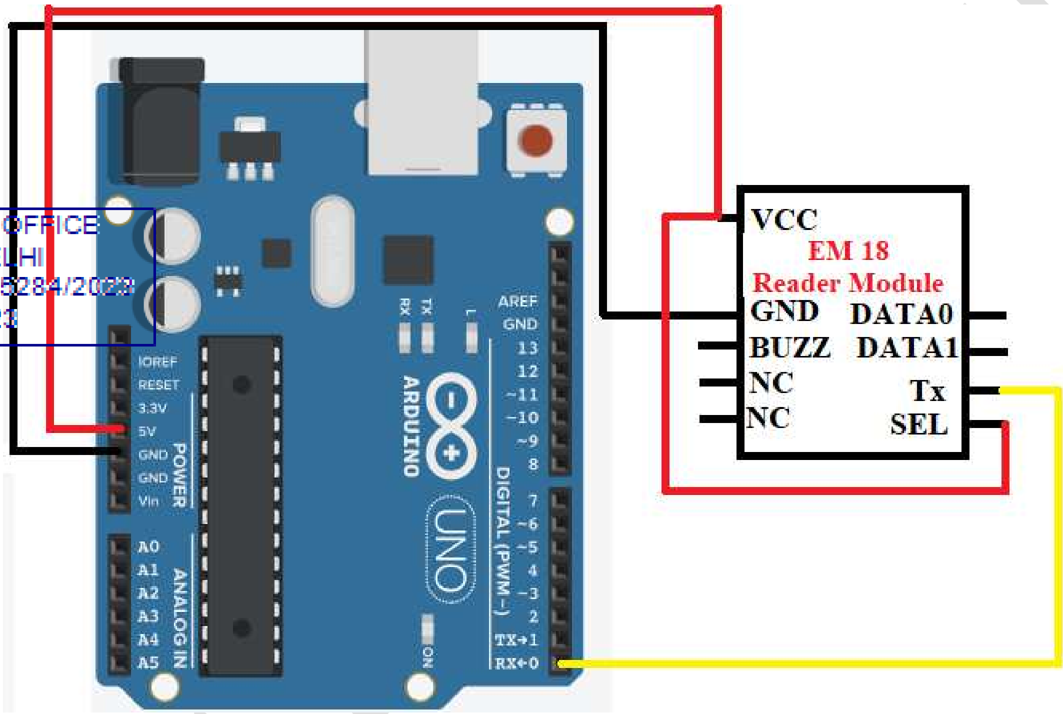

Figure 5: Circuit diagram for EM 18 with arduino uno

Algorithm:
1) Initialize serial port communication using 9600 baud rate
2) Initialize array variable ‘rfid_tag_no’ which will store 12 digit RFID tag number
3) Initialize ‘i’ variable which will count number of digits from RFID tag read by program.

4) While data is available on serial port of arduino board, read 12 digit RFID tag number and store it in variable. If serial port data is not available then jump to the step number 6.
5) After collection of 12 digit RFID tag number, display it on serial port.
6) Wait for 1 second and jump to the step number 4.

Program:

```
//connect RX of arduino board to TX of RFID module (EM18)

void setup()
{
//Initialize serial port communication at the baud rate 9600
  Serial.begin(9600);
}

void loop(){
    char rfid_tag_no[12];
    int i;

 // If serial data available then read RFID tag number
    while(Serial.available())
          {
          for(i=0; i < 12; )
                    {
                    rfid_tag_no[i] = Serial.read();
                    delay(5);
                    i++;
                    }
        Serial.println(rfid_tag_no);
          }

  delay(1000);
}
```

Result: We have successfully written and execute arduino program for reading RFID cards.

Questions

1) What are the different types of RFID available?

2) What are difference between EM18 and RC522?

Experiment Number - VI

Aim: Create local server using NodeMCU and control LED operation through browser.

Requirement: NodeMCU board, 5mm LED, 330Ω resistor,breadboardand connecting wires

Algorithm:

1) Prepares a server that waits for requests on port 80.
2) Initialize serial port with baud rate 115200
3) Connect Nodemcu to WiFi network
4) Start local created server (in step 1) after successful WiFi connection establishment.
5) Print IP address of local server on serial port
6) Initialize D0 pin of NodeMCU as output pin
7) Listen for any incoming request from clients if request from client is not received then program will wait for that. If request is received then program will proceed to next step.
8) Read client request until carriage return
9) Check index of ‘ /on’, if it is other than -1 then switch on LED which is connected to D0 pin of NodeMCU. If index value of ‘/on’ is -1 then it should check index of ‘ /off’ , if it is other than -1 then switch off LED which is connected to D0 pin of NodeMCU.
10) Write a program for local web server page
    i) Initialize title of project is LED control which will be display on tab
    ii) Provide heading to the web page as Priyadarshini College of Engineering, Department of Electronics & Telecommunication Engineering, WiFi Controlled LED.
    iii) Create two buttons with name LED ON and LED OFF and assigned them actions.

Program:

```
#include <ESP8266WiFi.h>
const char* ssid1 = "Redmi Note XX";
const char* pass1 = "k12345XXXX";

//Prepares a server that waits for requests on port 80.
WiFiServer server(80);

void setup()
{
// Set the transmission rate of the serial port to 115200 bits per second.
  Serial.begin(115200);

// Display message Connecting to WiFi on serial port
  Serial.println("Connecting to WiFi");

// begin connection of Nodemcu to WiFi network
  WiFi.begin(ssid1,pass1);

// wait until NodeMCU connection establishment with WiFi network
  while(WiFi.status()!=WL_CONNECTED)
          {
            erial.print("- ");
            lay(600);
```

```
// Display message NodeMCU connected to WiFi Network on serial port
  Serial.println("NodeMCU connected to WiFi Network");

// Start local created server after internet connection
 server.begin();

// Display IPv4 address of local server on serial port
 Serial.println(WiFi.localIP());

// Initialize D0 pin of NodeMCU as output pin
pinMode(D0,OUTPUT);
}
void loop()
{
// listen for any incoming request from clients
  WiFiClient client = server.available();

//if request from client is not received then program does not execute following statements
  if(!client){
    return;
  }

// Read client request until carriage return
    String client_req = client.readStringUntil('\r');

// Print request message on serial port
Serial.println(client_req);

// check index of ' /on'
if(client_req.indexOf(" /on")!= -1)
  {
      Serial.println("Current LED State = ON");
      digitalWrite(D0,LOW);
    }

// check index of ' /off'
else if(client_req.indexOf(" /off")!= -1)
 {
      Serial.println(" Current LED State = OFF");
      digitalWrite(D0,HIGH);
    }
// Program for Server Web Page
 client.println("<html><title>LED Control</title><center>");
  client.println("<h1>Priyadarshini College of Engineering </h1>");
  client.println("<h1>Department of Electronics & Telecommunication Engineering </h1>");
  client.println("<h1>WiFi Controlled LED</h1>");
  client.println("<a href='/on'><button>LED ON</button></a>");
         ntln("<a href='/off'><button>LED OFF</button></a>");
         ntln("</center></html>");
```

Output:

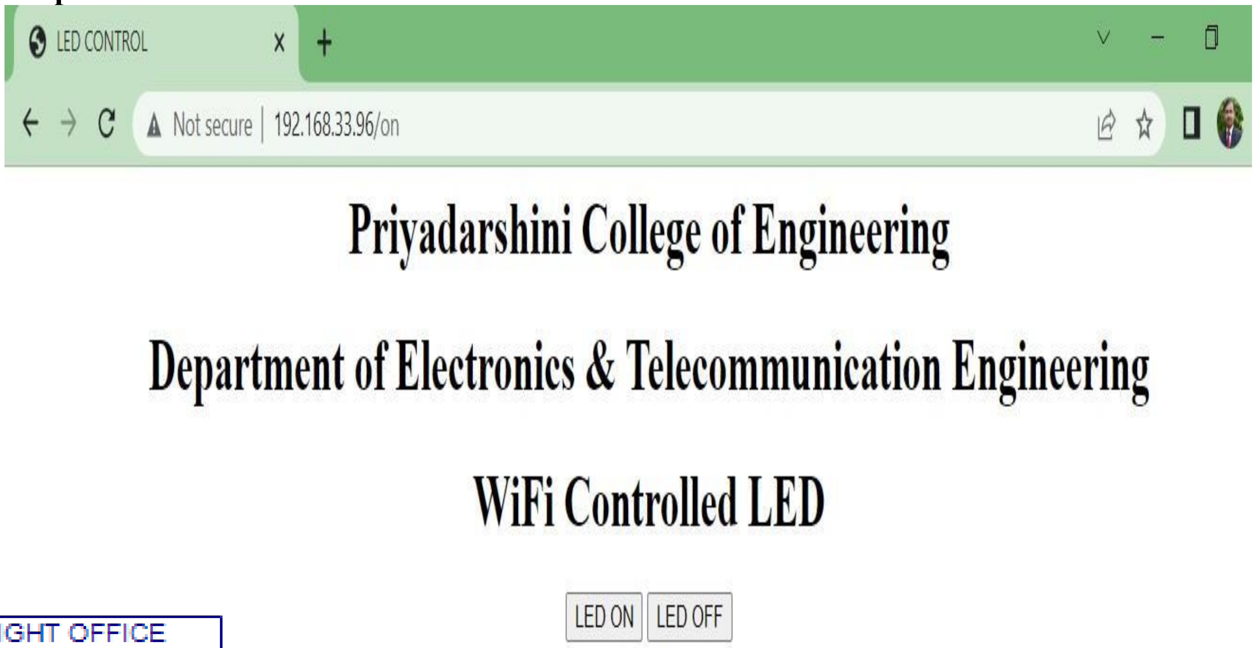

Figure 6a: Screenshot of local server web page

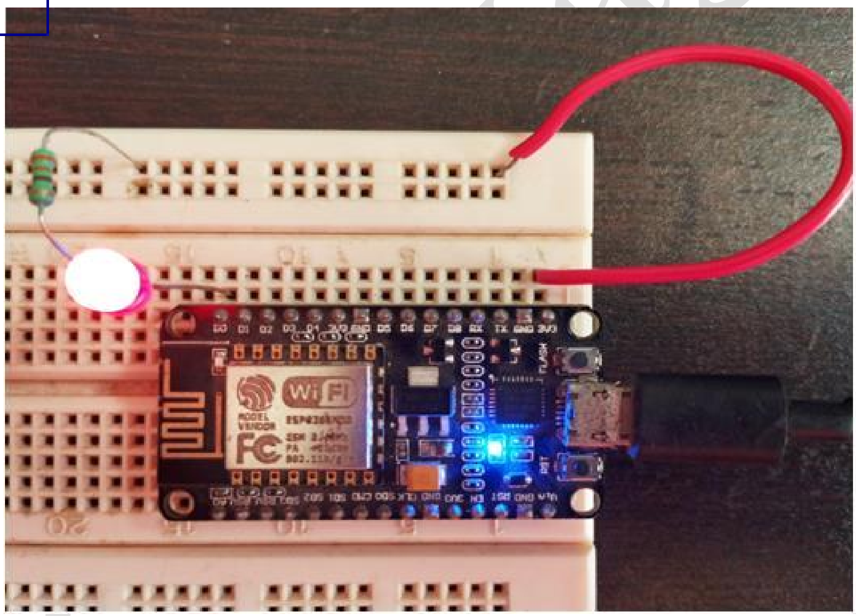

Figure 6b: LED ON State

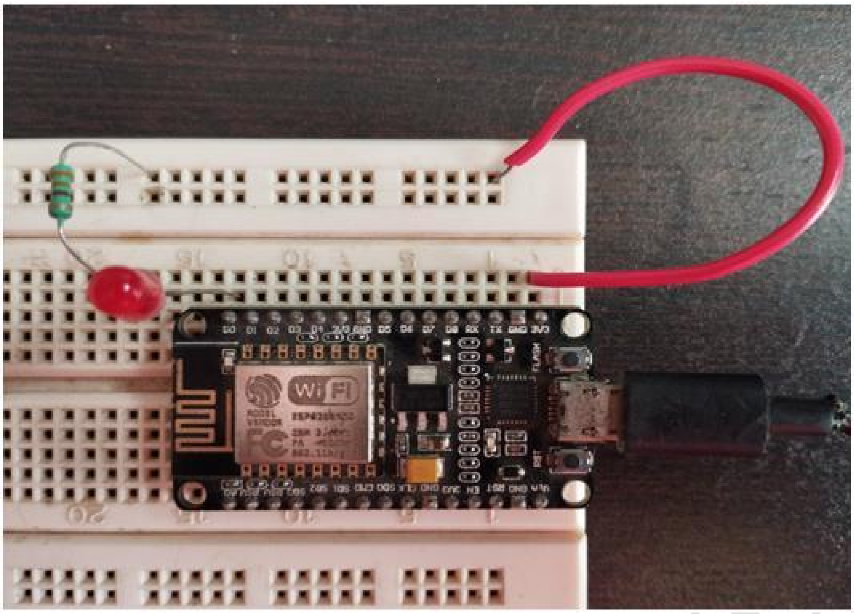

Figure 6c: LED OFF State

Result: We have successfully written and execute program for creation of local server using NodeMCU and control LED operation through browser.

Questions

1) What are features of NodeMCU?

2) What are the limitations of local server created using NodeMCU?

Experiment Number - VII

Aim: Interface DHT 11 humidity and temperature senor with nodeMCU and write a code to send temperature and humidity reading to remote server continuously.

Requirement: NodeMCU board, DHT11 sensor, breadboard, connecting wires and account on thingspeak remote server

Circuit Diagram:
Make the connection of circuit as shown in following figure

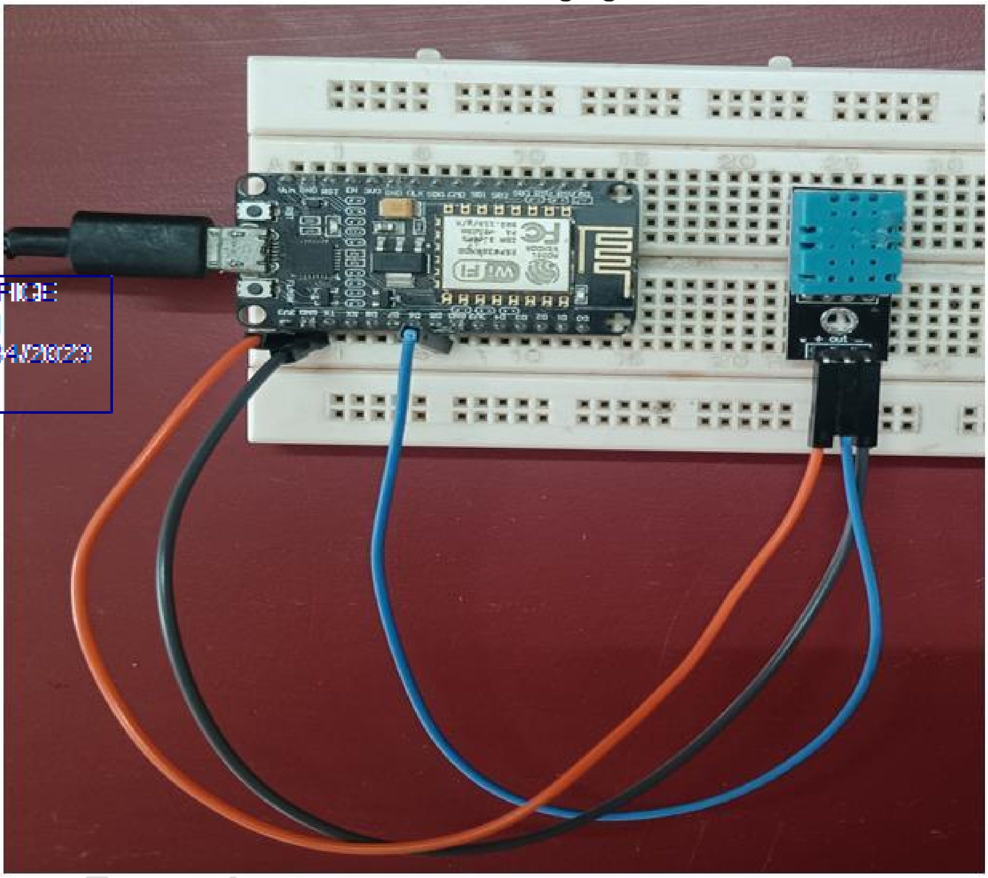

Figure 7: Circuit schematic of DHT11 interfacing with NodeMCU

Pre-requisite
1) ESP8266 board package must be installed
2) ThingSpeak library must be installed
3) DHT11 library must be installed
4) Programmer should have account on thingspeak.com
5) Create one channel (channelID and writeAPIkey we will get there)

Algorithm:
1) Create object of DHT class
2) Create object of WiFiClient class
3) Initialize serial port communication using 115200 baud rate
ze and connect with WiFi network with known service set identifier and password
ze the DHT11 sensor module
ze client connection with thingspeak server

7) Measure temperature and humidity using predefined functions from DHT library and store it in variables.
8) Send measured value to thingspeak server using channeled and writeAPIkey
9) Display temperature and humidity value on serial port
10) Wait for 15 seconds and jump to step 7

Program:

```
#include <ESP8266WiFi.h>
#include<ThingSpeak.h>
#include<DHT.h>

const char *ssid1 =  "Redmi Note XX";
const char *pass1 =  "kb768p38i5XXXX";

DHT dht(D5, DHT11);
 WiFiClient client;

long channelID = 2063xxx;
const char writeAPIkey[] = "86B0ZZ2YNDxxxxxx";

void setup()
{
        //Initialize serial port communication at 115200 baud rate
          Serial.begin(115200);

        // begin connection of Nodemcu to WiFi network
          WiFi.begin(ssid1, pass1);

        // wait until NodeMCU connection establishment with WiFi network
          while(WiFi.status() != WL_CONNECTED)
                {
                Serial.print("*");
                delay(600);

                }
          Serial.println();

        //Display NodeMCU connected to network on serial port
          Serial.println("NodeMCU connected to WiFi network");

        // Begin the DHT11 sensor module
          dht.begin();

        // Initialize client connection with Thingspeak server
          ThingSpeak.begin(client);
}

()

Measure temperature and humidity using predefined functions from DHT library
```

```
// and store it in variables.
  float temp_reading = dht.readTemperature();
  float humi_reading = dht.readHumidity();

// Set field value of thingspeak server and write it
  ThingSpeak.setField(1, humi_reading) ;
  ThingSpeak.setField(2, temp_reading) ;
  ThingSpeak.writeFields(channelID, writeAPIkey);

// Display temperature and humidity value on serial port
  Serial.print("Current Temperature of Environment in Celsius =  ");
  Serial.print(temp_reading);
  Serial.print(" Current Humidity of Environment in Percentage =  ");
  Serial.print(humi_reading);

//As server value is updated after every 15 seconds, generate delay of 15 seconds
  delay(15000);
}
```


Result: We have successfully written and execute the program for fetching humidity and temperature using DHT 11 sensor and sent it to remote server using NodeMCU.

Questions

1) Compare DHT11 & DHT22 sensor

2) Write the name of different remote server available for IoT application apart from ThingSpeak.

Experiment Number - VIII

Aim: Interface LM35 with NodeMCU and write a code to fetch and send temperature reading to remote server.

Requirement: NodeMCU board, LM35 sensor, breadboard, connecting wires and account on thingspeak remote server

Circuit Diagram:

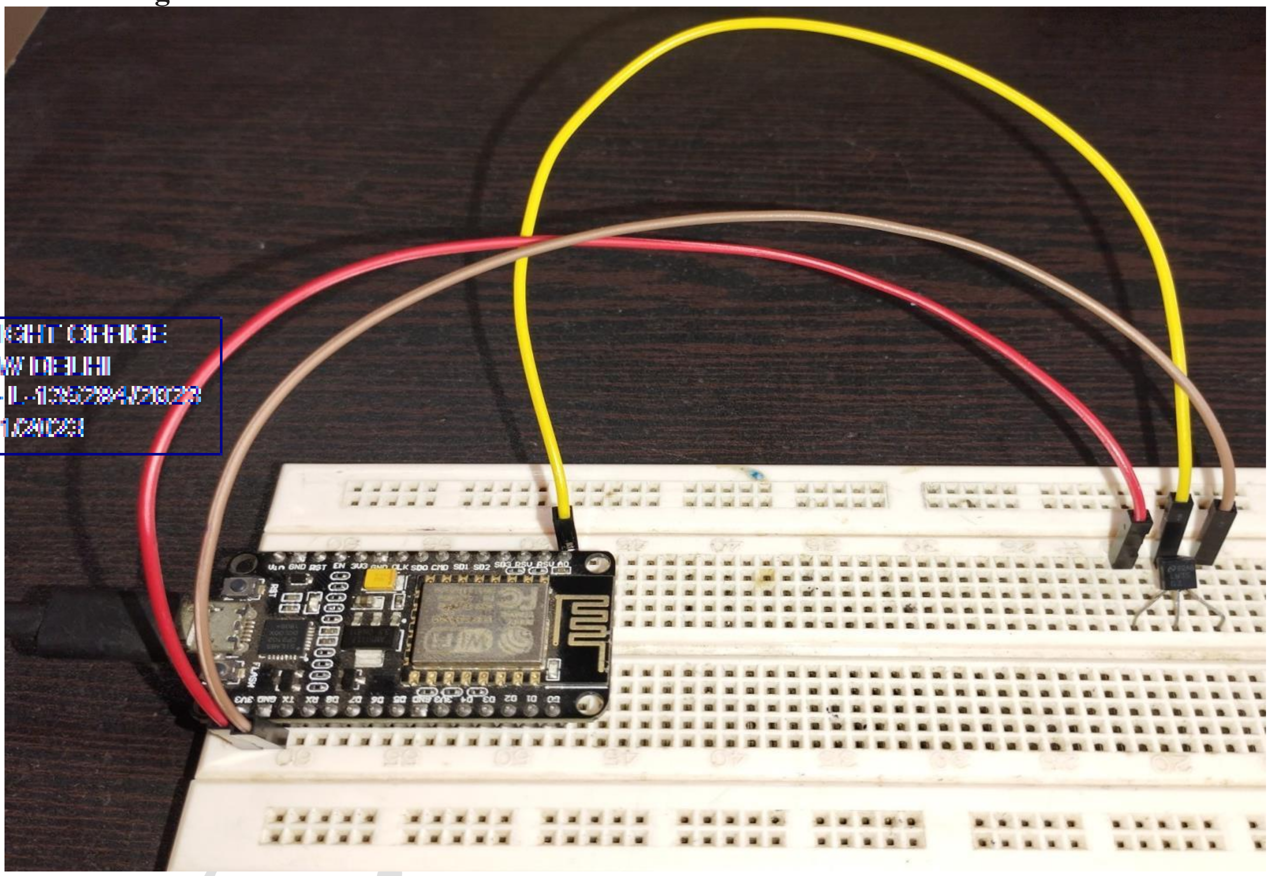

Figure 8: Circuit schematic of LM35interfacing with NodeMCU

Pre-requisite

1) ESP8266 board package must be installed
2) ThingSpeak library must be installed
3) Programmer should have account on thingspeak.com
4) Create one channel (channelID and writeAPIkey we will get there)

Algorithm:

1) Initialize serial port communication using 115200 baud rate
2) Initialize and connect with WiFi network with known service set identifier and password
3) Display the IP address of connected client device on serial port
4) Initialize client connection with thingspeak server
5) Read the temperature of the surroundings using the LM35 sensor, which is interfaced with ut pin of the NodeMCU i.e. A0, and save it in a variable.
t measured temperature value in Celsius and Fahrenheit unit
easured value to thingspeak server using channeled and writeAPIkey

8) Display temperature value in Celsius and Fahrenheit unit on serial port
9) Wait for 15 seconds and jump to step 5

Program:

```
#include <ESP8266WiFi.h>
#include <ThingSpeak.h>

const char *ssid1 =  "Redmi Note xx";
const char *pass1 =  "kb768p38i5uxxxx";

WiFiClient client;

long channelID = 20632xx;
const char writeAPIKey[] = "LVQW9KEI3GKxxxxx";
#define tempsensor_pin A0

void setup()
{
  // Initialize serial port communication at baud rate 115200
  Serial.begin(115200);

// begin connection of Nodemcu to WiFi network
  WiFi.begin(ssid1, pass1);

// wait until NodeMCU connection establishment with WiFi network
  while(WiFi.status() != WL_CONNECTED)
  {
     Serial.print("- ");
    delay(600);
  }
  Serial.println();

//Display NodeMCU connected to WIFI network
Serial.println("NodeMCU connected to WiFi network");

// Initialize client connection with Thingspeak server
ThingSpeak.begin(client);
}

void loop()
 {
//store LM35 sensor voltage reading in variable
  int sensor_reading = analogRead(tempsensor_pin);

 // Convert LM35 sensor output to voltage
  float convert_to_volt = sensor_reading * (5.0 / 1024.0);

rt the voltage reading into Celsius and Fahrenheit
p_celcius = convert_to_volt * 100;
p_fahre = (temp_celcius * 9.0 / 5.0) + 32.0;
```

```
// Set field value of thingspeak server and write it
  ThingSpeak.setField(1,temp_celcius) ;
  ThingSpeak.setField(2,temp_fahre) ;
  ThingSpeak.writeFields(channelID, writeAPIKey);

  // display temperature reading in different format on serial port
  Serial.print("Temperature in Celsius = ");
  Serial.print(temp_celcius);
  Serial.print(" ");
  Serial.print("Temperature in Fahrenheit = ");
  Serial.println(temp_fahre);

//generate delay for 15 seconds
  delay(15000);

}
```


Result: We have successfully written and execute the program for fetching temperature using LM35 sensor and sent it to remote server using NodeMCU.

Questions

1) Compare DHT11 & LM35 sensor

2) List different types of temperature sensor available in market which can be interfaced with arduino?

Experiment Number - IX

Aim: Interface a light emitting diode with Raspberry Pi board & write a code to switch on light emitting diode for one second and switch off light emitting diode for two seconds continuously.

Requirement: Raspberry Pi board 3B, LED, 330Ω resistor, and connecting wires

Circuit Diagram:
Make the connection of circuit as shown in following figure

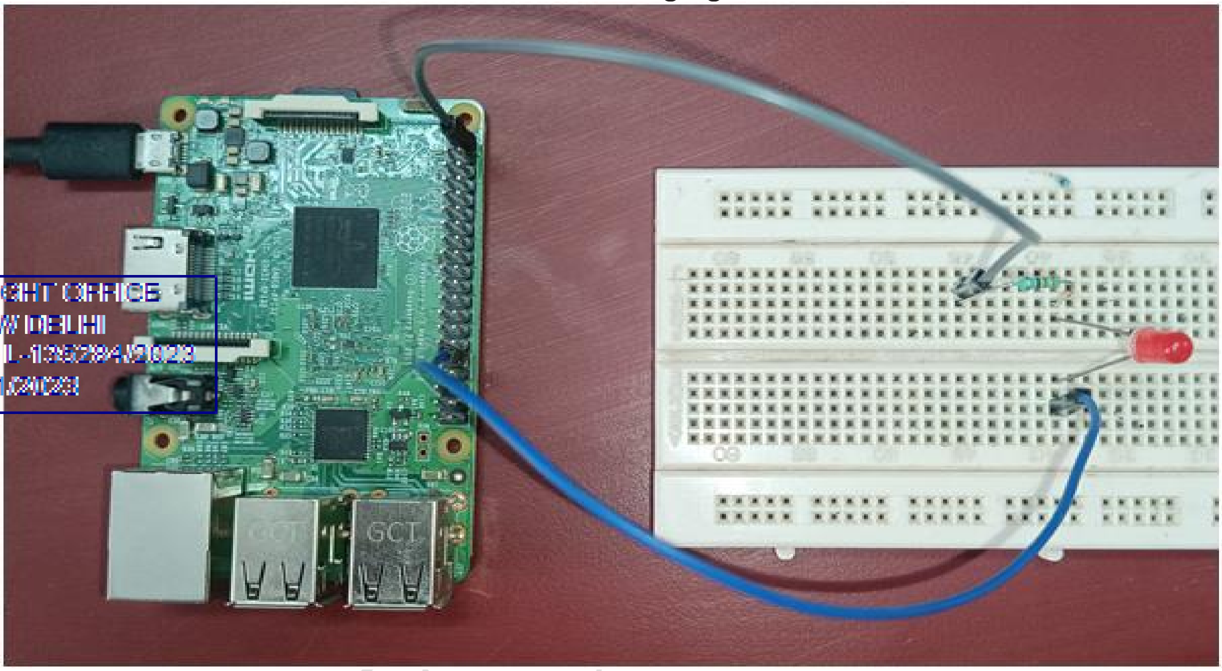

Figure 9: Circuit schematic of LED interfacing with Raspberry Pi

Algorithm:
1) Import Raspberry Pi general purpose input/output (GPIO) library for controlling GPIO pins using software
2) Import time library for delay generation purpose
3) Assign name led_pin to GPIO12 pin
4) Define Raspberry Pi board pin numbering system as GPIO numbering
5) Disable all warnings
6) Initialize GPIO12 pin as output pin
7) Initialize infinite loop using while statement
8) Switch on LED for 1 second
9) Switch off LED for 2 seconds
10) Jump to step 8

Program:

```
import RPi.GPIO as GP
import time as t

led_pin=12
GP.setmode(GPIO.BCM)     # Select physical pin numbering as GPIO numbering
GP.setwarnings(False)          # Disable all warnings

GP.setup(led_pin, GPIO.OUT)   # set GPIO12 pin as output pin

while True:
    GP.output(led_pin, GPIO.LOW)  # Send logic 1 on GPIO12 pin
    print('LED ON')                              # Display message
    t.sleep(1)                                # Generate delay of 1 second
    GP.output(led_pin, GPIO.HIGH)   # Send logic 0 on GPIO12 pin
    print('LED OFF')
    t.sleep(2)                                 # Generate delay of 2 seconds
```


Result: We have successfully written and execute Raspberry Pi program for LED on and off.

Questions

1) How many operating system available for Raspberry Pi boards? Which are those?
2) What will you get if you run the pinout command on Pi?
3) How is Raspberry Pi different from Arduino?

Experiment Number - X

Aim: Interface ultrasonic sensor HC-SR04 with Raspberry Pi board & write a code to calculate distance of an object.

Requirement: Raspberry Pi board 3B, ultrasonic sensor HR SC-04, resistors and connecting wires

Circuit Diagram:
Make the connection of circuit as shown in following figure

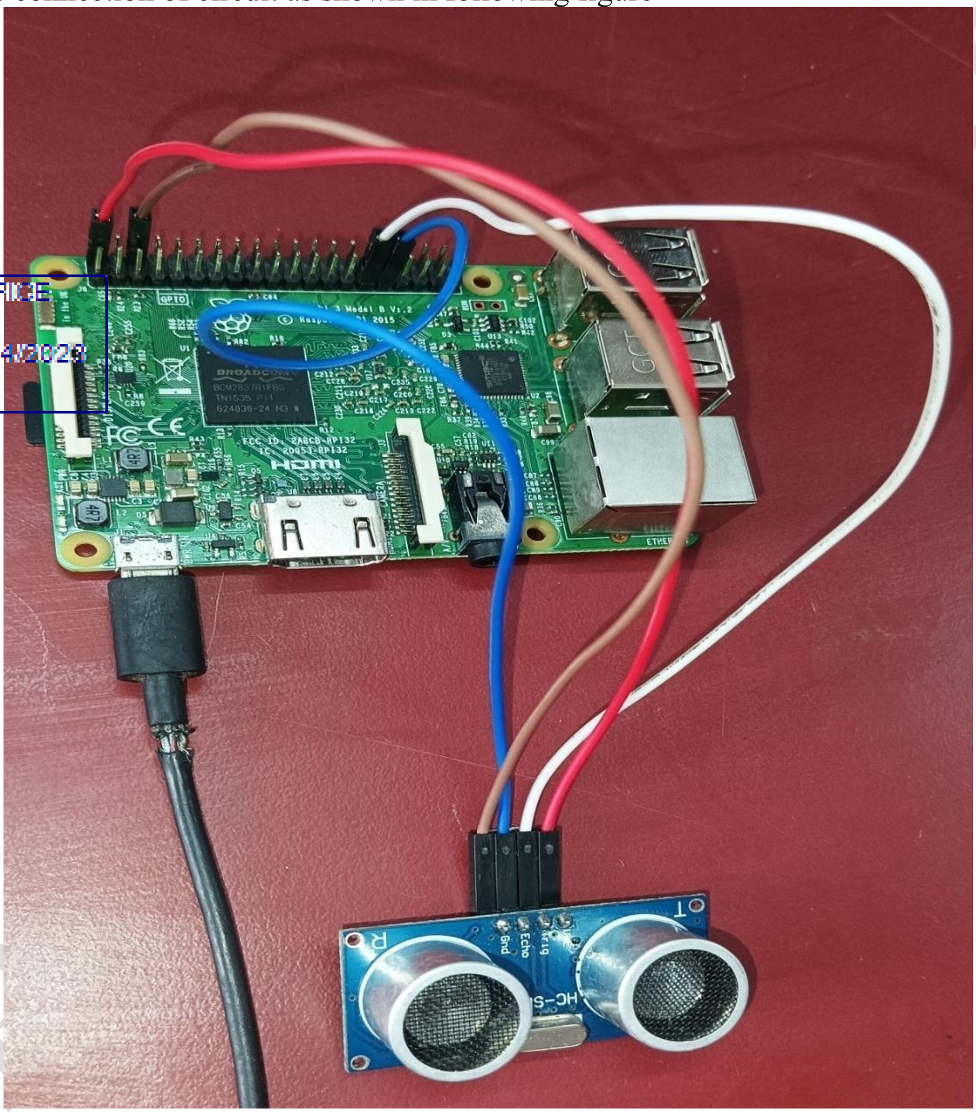

Figure 10: Circuit schematic of ultrasonic sensor interfacing with Raspberry Pi

Algorithm:
1) From gpiozero library import DistanceSensor function to calculate distance of an object using ultrasonic sensor
2) Import time library for delay generation purpose
numbering format as GPIO pin instead of physical pin numbering
warnings

5) Initialize GPIO6 pin (output pin) as trigger pin and GPIO5 pin (input pin) as echo pin & create one object with name ultrasonic_sensor
6) Initialize infinite loop using while statement
7) Get the object distance using distance method. The output of this method is in meter then converts it into centimeter by multiplying answer by 100
8) Display the object distance reading
9) Wait for one second then jump to step 7.

Program:

```
# In gpiozero library, default pin numbering is GPIO numbering
from gpiozero import DistanceSensor
import time as t

ultrasonic_sensor = DistanceSensor(echo=5, trigger=6)  # Call DistanceSensor function

while True:
    distance= (ultrasonic_sensor.distance*100)          # Convert distance in centimeter
    print('distance = {0:0.2f} cm'.format (distance))   # Display the object distance reading
    t.sleep(1)                                          # Generate delay of 1 second
```


Result: We have successfully written and execute Raspberry Pi program for calculating distance of an object using ultrasonic sensor.

Questions
1) What is NOOBS?
2) What is a GPIO pin? How many GPIO pins available in Raspberry PI 4 Model?