

**Assignment - 4**  
ESP 32 – Ultrasonic Sensor

Assignment Date	31 October 2022
Student Name	Kishor G. K
Student Roll Number	611219106039
Maximum Marks	2 Marks

**Question-1:**

Write code and Connection in wokwi for ultrasonic sensor

**Solution:**

**Program:**

```
#include <WiFi.h>
#include <WiFiClient.h>
#include <PubSubClient.h>
const int trigPin = 5;
const int echoPin = 18;
//define sound speed in cm/uS
#define SOUND_SPEED 0.034
#define CM_TO_INCH 0.393701
long duration;
float distanceCm;
float distanceInch;

void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
//-----credentials of IBM Accounts-----

#define ORG "b31tni"//IBM ORGANITION ID
#define DEVICE_TYPE "Assignment4"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "assignment"//Device ID mentioned in ibm watson IOT
Platform#define TOKEN "6r?TKCIuy+okJ?9B+7" //Token
String data3;

//----- Customise the above values -----
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of
event perform and format in which data to be send
char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT command
type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
```

```

char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id

WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient);

void setup() {
    Serial.begin(115200); // Starts the serial communication
    pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
    pinMode(echoPin, INPUT); // Sets the echoPin as an Input
    Serial.println();
    wificonnect();
    mqttconnect();
}

void loop() {
    // Clears the trigPin
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    // Sets the trigPin on HIGH state for 10 micro seconds
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    // Reads the echoPin, returns the sound wave travel time in microseconds
    duration = pulseIn(echoPin, HIGH);

    // Calculate the distance
    distanceCm = duration * SOUND_SPEED/2;

    // Convert to inches
    distanceInch = distanceCm * CM_TO_INCH;

    // Prints the distance in the Serial Monitor
    Serial.print("Distance (cm): ");
    Serial.println(distanceCm);
    Serial.print("Distance (inch): ");
    Serial.println(distanceInch);

    PublishData(distanceCm);
    delay(1000);
    if (!client.loop()) {
        mqttconnect();
    }
}

void PublishData(float Cm) {

```

```

mqttconnect();//function call for connecting to ibm
/*
    creating the String in in form JSon to update the data to ibm cloud
*/
String payload = "{\"Distance (cm)\":";
payload += Cm;
payload += "}";

Serial.print("Sending payload: ");
Serial.println(payload);

if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");// if it sucessfully upload data on the cloud
    then it will print publish ok in Serial monitor or else it will print publish
    failed
} else {
    Serial.println("Publish failed");
}
}

void mqttconnect() {
    if (!client.connected()) {
        Serial.print("Reconnecting client to ");
        Serial.println(server);
        while (!client.connect(clientId, authMethod, token)) {
            Serial.print(".");
            delay(500);
        }

        initManagedDevice();
        Serial.println();
    }
}

void wificonnect() //function defination for wificonnect
{
    Serial.println();
    Serial.print("Connecting to ");

    WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish
    the connection
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.println("WiFi connected");
}

```

```

Serial.println("IP address: ");
Serial.println(WiFi.localIP());
}

void initManagedDevice() {
  if (client.subscribe(subscribetopic)) {
    Serial.println((subscribetopic));
    Serial.println("subscribe to cmd OK");
  } else
  {
    Serial.println("subscribe to cmd FAILED");
  }
}

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
{

  Serial.print("callback invoked for topic: ");
  Serial.println(subscribetopic);
  for (int i = 0; i < payloadLength; i++) {
    //Serial.print((char)payload[i]);
    data3 += (char)payload[i];
  }
}
}

```

## Wokwi Simulation:

The image shows the Wokwi simulation environment. On the left, the code editor displays the Arduino sketch. On the right, the simulation window shows a virtual ESP32 board with a red box highlighting the sensor module. Below the simulation window, the console output shows the results of the simulation.

```

1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 #include <Arduino.h>
4 const int trigPin = 5;
5 const int echoPin = 18;
6 //define sound speed in cm/us
7 #define SOUND_SPEED 0.034
8 #define CM_TO_INCH 0.393701
9 long duration;
10 float distanceCm;
11 float distanceInch;
12
13
14 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
15 // ... credentials of IOT Accounts
16
17 #define ORG "b3itml"//IOT ORGANIZATION ID
18 #define DEVICE_TYPE "Assignment-4"//device type mentioned in the wuon IOT Platform
19 #define DEVICE_ID "Assignment"//device ID mentioned in the wuon IOT Platform
20 #define TOKEN "677c7c0yok379b17" //Token
21 String data;
22
23
24
25 // ... to populate the above values
26 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // server name
27 // char publishTopic[] = "iot-2/ev/data/rml/json"; // topic name and type of event payload
28 char subscribetopic[] = "iot-2/cmd/test/rml/string"; // cmd REPRESENTER command type
29 char authMethod[] = "use-token-auth"; // authentication method
30 char token[] = TOKEN;
31 char clientId[] = "id" ORG "/" DEVICE_TYPE "/" DEVICE_ID //client id
32
33 WiFiClient wifiClient; // creating the instance for wifiClient
34 PubSubClient client(server, 1883, callback, wifiClient);
35
36 void setup() {

```

Simulation output:

```

Distance (inch): 53.92
Sending payload: {"Distance (cm)":136.95}
Publish ok
Distance (cm): 136.95
Distance (inch): 53.92
Sending payload: {"Distance (cm)":136.95}
Publish ok

```

## IoT Watson Platform:

IBM Watson IoT Platform

2K1Project19@Kod.ac.in  
ID: 831161

Browse Action Device Types Interfaces Add Device

Identity Device Information Recent Events State Logs

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
Data	("Distance (cm)":68.95)	json	a few seconds ago
Data	("Distance (cm)":68.95)	json	a few seconds ago
Data	("Distance (cm)":68.95)	json	a few seconds ago
Data	("Distance (cm)":119.97)	json	a few seconds ago
Data	("Distance (cm)":136.95)	json	a few seconds ago

Items per page 100 | 1-1 of 1 item 1 of 1 page < 1 >

1 Simulation running

Wokwi Project Link <https://wokwi.com/projects/347106404490805844>