

## Assignment-4

Assignment date	24 <sup>th</sup> October, 2022
Student Name	Rajkumar S
Roll Number	2019504050
Maximum marks	2 Marks

### Question:

Write code and connections in wokwi for ultrasonic sensor. Whenever distance is less than 100cms send "alert" to ibm cloud and display in device recent events. Upload document with wokwi share link and images of ibm cloud.

### Code:

```
#include <WiFi.h> //library for wifi
#include <PubSubClient.h> //library for MQTT

#define ECHO_GPIO 12
#define TRIGGER_GPIO 14
#define MAX_DISTANCE_CM 100 // Maximum of 5 meters
#include "Ultrasonic.h"
Ultrasonic ultrasonic(14, 12);
int distance;

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

//-----credentials of IBM Accounts-----

#define ORG "84cem8" //IBM ORGANIZATION ID
#define DEVICE_TYPE "ESP32" //Device type mentioned in ibm watson IOT
Platform #define DEVICE_ID "RajEsp32" //Device ID mentioned in ibm watson IOT
Platform #define TOKEN "YKm?MdA47jor5uFPBT" //Token
String data3;
float h, t;

//----- 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/command/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); //calling the
predefined client id by passing parameter like server id,portand
wificredential

void setup()// configuring the ESP32
{
    Serial.begin(115200);
    delay(10);
    Serial.println();
    wificonnect();
    mqttconnect();
}

void loop()// Recursive Function
{

    distance = ultrasonic.read(CM);
    if(distance < 100){
        Serial.print("Distance in CM: ");
        Serial.println(distance);
        PublishData(distance);
        delay(1000);
        if (!client.loop()) {
            mqttconnect();
        }

    }

    delay(1000);

}

/* .....retrieving to
Cloud. .... */

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

```

```

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

if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");// if it successfully 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++) {
        data3 += (char)payload[i];
    }
    Serial.println("data: "+ data3);
    if(data3=="lighton")
    {
        Serial.println(data3);
    }
    else
    {
        Serial.println(data3);
    }
    data3="";
}

```

Output:

The screenshot displays the Arduino IDE interface. The left pane shows the 'sketch.ino' file with the following code:

```

1 #include <Ultrasonic.h>
2 Ultrasonic ultrasonic(14, 12);
3 int distance;
4 void callback(char* subscribetopic, byte* payload, unsigned int
5 payloadLength);
6 //-----credentials of IBM Accounts-----
7 #define ORG "84cem8"//IBM ORGANITION ID
8 #define DEVICE_TYPE "ESP32"//Device type mentioned in ibm watson IOT Platform
9 #define DEVICE_ID "RajEsp32"//Device ID mentioned in ibm watson IOT Platform
10 #define TOKEN "YKm?MjA447jor5uFPB1" //Token
11 String data3;
12 float h, t;
13 //----- Customise the above values -----
14 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
15 char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform and format
16 char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd REPRESENT command type AND COMMAND ID
17 char authMethod[] = "use-token-auth";// authentication method
18 char token[] = TOKEN;
19 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
20 //-----
21 WiFiClient wifiClient; // creating the instance for wificlient
22 PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined client id by pass
23 void setup()// configureing the ESP32
24 {
25     Serial.begin(115200);
26     delay(10);
27     Serial.println();
28     wifiConnect();
29     mqttConnect();
30 }
31 void loop()// Recursive Function
32 {
33     distance = ultrasonic.read(CM);
34     if(distance < 100){

```

The right pane shows a 'Simulation' window with a circuit diagram. It includes an 'HC-SR04' ultrasonic sensor and an 'ESP32' microcontroller. The sensor is connected to the ESP32 via four wires (red, yellow, green, and black). A 'Distance' slider is set to 68cm. Below the simulation, the output console shows the following log:

```

Publish ok
Distance in CM: 71
Sending payload: {"Alert Distance:":71.00}
Publish ok
Distance in CM: 71
Sending payload: {"Alert Distance:":71.00}
Publish ok

```

IBM Watson IoT Platform

ID: 84cem8

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	{"Alert Distance":71}	json	a few seconds ago
Data	{"Alert Distance":71}	json	a few seconds ago
Data	{"Alert Distance":71}	json	a few seconds ago
Data	{"Alert Distance":71}	json	a few seconds ago
Data	{"Alert Distance":71}	json	a few seconds ago

Items per page 50

1-1 of 1 item

1 of 1 page

1

WOKWI

SAVE

SHARE

Docs

sketch.ino

diagram.json

Ultrasonic.h

libraries.txt

Ultrasonic.cpp

Library Manager

Simulation

00:09.286 100%

Editing Ultrasonic Distance Sensor

Distance: 54cm

Publish ok

Distance in CM: 56

Sending payload: {"Alert Distance":56.00}

Publish ok

Distance in CM: 56

Sending payload: {"Alert Distance":56.00}

Publish ok

```

1  #include <Ultrasonic.h>
2  Ultrasonic ultrasonic(14, 12);
3  int distance;
4  void callback(char* subscribetopic, byte* payload, unsigned int
5  payloadLength);
6  //-----credentials of IBM Accounts-----
7  #define ORG "84cem8"//IBM ORGANTITION ID
8  #define DEVICE_TYPE "ESP32"//Device type mentioned in ibm watson IOT Platform
9  #define DEVICE_ID "RajEsp32"//Device ID mentioned in ibm watson IOT Platform
10 #define TOKEN "YKm?MdA47jor5uFPBT" //Token
11 String data3;
12 float h, t;
13 //----- Customise the above values -----
14 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
15 char publishTopic[] = "iot-2/evt/data/fmt/json";// topic name and type of event perform and format
16 char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd REPRESENT command type AND COMMAND ID
17 char authMethod[] = "use-token-auth";// authentication method
18 char token[] = TOKEN;
19 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
20 //-----
21 WiFiClient wifiClient; // creating the instance for wifiClient
22 PubSubClient client(server, 1883, callback, wifiClient); //calling the predefined client id by pas
23 void setup()// configuring the ESP32
24 {
25   Serial.begin(115200);
26   delay(10);
27   Serial.println();
28   wifiConnect();
29   mqttConnect();
30 }
31 void loop()// Recursive Function
32 {
33   distance = ultrasonic.read(CM);
34   if(distance < 100){

```

Device

1 item selected

Cancel

IBM Watson IoT Platform

ID: 84cem8

Browse

Action

Device Types

Interfaces

Add Device

Identity

Device Information

Recent Events

State

Logs

X

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

Event	Value	Format	Last Received
Data	{"Alert Distance":56}	json	a few seconds ago
Data	{"Alert Distance":56}	json	a few seconds ago
Data	{"Alert Distance":56}	json	a few seconds ago
Data	{"Alert Distance":56}	json	a few seconds ago
Data	{"Alert Distance":56}	json	a few seconds ago

Items per page 50 | 1-1 of 1 item

1 of 1 page

1

**Project Link:**

<https://wokwi.com/projects/346964753156932179>