

ASSIGNMENT-4

WOKWI SIMULATOR AND IBM CLOUD

Assignment Date	19 November 2022
Student Name	PRIYADHARSHINI M
Student Roll Number	312319106124
Team ID	PNT2022TMID00340
Maximum Marks	2 Marks

QUESTION:

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

SOLUTION:

```
#include <WiFi.h> //library for wifi
#include <PubSubClient.h> //library for MQTT
#define ECHO_GPIO 12
#define TRIGGER_GPIO 13
#define MAX_DISTANCE_CM 100 // Maximum of 0.1 meters
#include "Ultrasonic.h"
Ultrasonic ultrasonic(13, 12);
int distance;
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
//-----credentials of IBM Accounts-----
#define ORG "4stdk7" //IBM ORGANITION ID
#define DEVICE_TYPE "ESP32_Controller" //Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "ibmA-4" //Device ID mentioned in ibm watson IOT Platform
#define TOKEN "iZaIta1phIzD!fwQR+" //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()// configureing 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 Centimeters: ");
    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 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];
    }
    Serial.println("data: "+ data3);
    if(data3=="lighton")
    {
        Serial.println(data3);
    }
    else

```

```

{
Serial.println(data3);
}
data3="";
}

```

WOKWI SIMULATION:

The screenshot displays the Wokwi simulation interface. On the left, the Arduino IDE shows a sketch for an ESP32 that uses the Ultrasonic library to measure distance and the PubSubClient library to send data to the IBM Watson IoT Platform. The code includes credentials for the device and a callback function to handle incoming data. On the right, the simulation shows an ESP32 board connected to an HC-SR04 ultrasonic sensor. The console output indicates that the sensor has detected a distance of 74 cm and has successfully sent a JSON payload to the IoT platform.

```

Publish ok
Distance in Centimeters: 74
Sending payload: {"Alert Distance:":74.00}
Publish ok
Distance in Centimeters: 74
Sending payload: {"Alert Distance:":74.00}
Publish ok

```

IBM Watson Platform-Device Event Log:

The screenshot shows the IBM Watson IoT Platform dashboard. The 'Recent Events' tab is selected, displaying a table of events received from the device. The table has four columns: Event, Value, Format, and Last Received. There are five entries, all of which are 'Data' events with a JSON payload containing an alert distance of 74. The events were received 'a few seconds ago'.

Event	Value	Format	Last Received
Data	{"Alert Distance:":74}	json	a few seconds ago
Data	{"Alert Distance:":74}	json	a few seconds ago
Data	{"Alert Distance:":74}	json	a few seconds ago
Data	{"Alert Distance:":74}	json	a few seconds ago
Data	{"Alert Distance:":74}	json	a few seconds ago

WOKWI SIMULATION LINK:

<https://wokwi.com/projects/348781206784967251>