

#### Assignment -4

Assignment Date	02 November 2022
Student Name	Ms.Darsini B
Student Roll Number	611219106010
Maximum Marks	2 Marks
Team ID	PNT2022TMID30241

#### Question:

Write code and connections in wokwi for the ultrasonic sensor. Whenever the distance is less than 100 cms 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.

#### Code:

```
#include <WiFi.h>
#include <WiFiClient.h>
#include <PubSubClient.h>
const int trigPin = 12;
const int echoPin = 14;
//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 "mps2km"//IBM ORGANITION ID
#define DEVICE_TYPE "Assignment-ibm"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "Sensor"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "uLF(HDd!OVrC43&_wj" //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 Output:

The screenshot displays the Wokwi online IDE interface. On the left, the code for 'wifi-scan.ino' is visible, which includes libraries for WiFi, PubSubClient, and defines constants for an ultrasonic sensor (HC-SR04). The code implements a callback function that publishes distance data to an MQTT topic. On the right, the simulation window shows the physical components: an ESP32 microcontroller and an HC-SR04 sensor. The output console at the bottom right shows the sensor's readings and the MQTT publishing process.

**Simulation Output:**

```

Distance (cm): 399.91
Distance (inch): 157.44
Sending payload: {"Distance (cm)":399.91}
Publish ok
Distance (cm): 399.92
Distance (inch): 157.45
Sending payload: {"Distance (cm)":399.92}
Publish ok

```

## IBM Cloud Alert:

The screenshot shows the IBM Cloud IoT Platform interface. On the left is a dark sidebar with various icons. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. An 'Add Device' button is in the top right. The main content area has tabs for 'Identity', 'Device Information', 'Recent Events' (selected), 'State', and 'Logs'. Below the 'Recent Events' tab, a message states: 'The recent events listed show the live stream of data that is coming and going from this device.' A table follows with the following data:

Event	Value	Format	Last Received
Data	{"Distance (cm)":399.96}	json	a few seconds ago
Data	{"Distance (cm)":399.92}	json	a few seconds ago
Data	{"Distance (cm)":399.96}	json	a few seconds ago
Data	{"Distance (cm)":399.92}	json	a few seconds ago
Data	{"Distance (cm)":400.01}	json	a few seconds ago

## Wokwi Share Link:

<https://wokwi.com/projects/305569599398609473>