

## Assignment - 4

### Question-1:

Write code and Connection in wokwi for ultrasonic sensor. Whenever distance is less than 100cms send "alert" to the ibm cloud and display in device recent events.

### 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 Wokwi simulation interface displays a sketch of an ESP32 microcontroller connected to an ultrasonic sensor. The sketch code is as follows:

```

1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* topic, byte* payload, unsigned int
4 payloadLength);
5 //-----credentials of IBM Accounts-----
6 #define ORG "9lxobn"//IBM ORGANIZATION ID
7 #define DEVICE_TYPE "ESP32PROJECT"//Device type mentioned in ibm watson IOT Platform
8 #define DEVICE_ID "ESP32"//Device ID mentioned in ibm watson IOT Platform
9 #define TOKEN "ESP32PROJECT" //Token
10 String data3;
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/Data/fmt/json";
13 char subscribTopic[] = "iot-2/cmd/test/fmt/String";
14 char authMethod[] = "use-token-auth";
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
17 WiFiClient wifiClient;
18 PubSubClient client(server, 1883, callback, wifiClient);
19 const int trigPin = 5;
20 const int echoPin = 18;
21 #define SOUND_SPEED 0.034
22 long duration;
23 float distance;
24 void setup() {
25   Serial.begin(115200);
26   pinMode(trigPin, OUTPUT);
27   pinMode(echoPin, INPUT);
28   wifiConnect();
29   mqttConnect();
30 }
31 void loop()
32 {
33   digitalWrite(trigPin, LOW);
34   delayMicroseconds(2);
35   digitalWrite(trigPin, HIGH);

```

The simulation results on the right show a constant distance of 399.94 cm:

```

Distance (cm): 399.96
Distance (cm): 399.94
Distance (cm): 399.94
Distance (cm): 399.94
Distance (cm): 399.92
Distance (cm): 399.94
Distance (cm): 399.94

```

## IoT Watson Platform:

The IBM Watson IoT Platform interface displays the details for a device with ID 883855. The device is currently disconnected. The recent events table shows the following data:

Event	Value	Format	Last Received
event_1	{"randomNumber":22,"temp":13,"hump":80}	json	a few seconds ago
event_1	{"randomNumber":51,"temp":24,"hump":47}	json	a few seconds ago
event_1	{"randomNumber":56,"temp":8,"hump":32}	json	a few seconds ago
event_1	{"randomNumber":56,"temp":8,"hump":47}	json	a few seconds ago
event_1	{"randomNumber":34,"temp":29,"hump":79}	json	a few seconds ago

2 Simulations running