

ASSIGNMENT-4

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.

CODE:

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

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

#define ORG "b4o8hv"//IBM ORGANITION ID
#define DEVICE_TYPE "ultrasonic"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "ultrasonicsensor"//Device ID mentioned in ibm watson IOT
Platform
#define TOKEN "1234567890" //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
    distance = duration * Speed/2;

    // Convert to inches
    distanceInch = distance * cm_to_inch;

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

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

void PublishData(float centimeter) {
    mqttconnect();//function call for connecting to ibm
    /*
        creating the String in in form JSON to update the data to ibm cloud
    */
    String payload = "{\"Distance in Centimeter\":\"";
    payload += centimeter;
    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... ");

    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];
    }
}
}

```

OUTPUT:

The screenshot displays the Wokwi IDE interface. On the left, the 'sketch.ino' file is open, showing the following code:

```

1 #include <WiFi.h>
2 #include <WiFiClient.h>
3 #include <PubSubClient.h>
4 const int trigPin = 5;
5 const int echoPin = 18;
6 //define sound speed in cm/uS
7 #define Speed 0.034
8 #define cm_to_inch 0.393701
9 long duration;
10 float distance;
11 float distanceInch;
12
13
14 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
15 {
16     //-----credentials of IBM Accounts-----
17     #define ORG "b4o9hv"//IBM ORGANITION ID
18     #define DEVICE_TYPE "ultrasonic"//Device type mentioned in ibm watson IOT Plat
19     #define DEVICE_ID "ultrasonicsensor"//Device ID mentioned in ibm watson IOT P
20     #define TOKEN "1234567890" //Token
21     String data3;
22
23
24
25 //----- Customise the above values -----
26 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
27 char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of even
28 char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT command
29 char authMethod[] = "use-token-auth";// authentication method
30 char token[] = TOKEN;

```

On the right, the 'Simulation' window shows a visual representation of the hardware: an ESP8266 microcontroller board connected to an ultrasonic sensor module. Below the simulation, the output log shows the following sequence of events:

```

Publish ok
Distance : 399.96
Sending payload: {"Distance in Centimeter":399.96}
Publish ok
Distance : 399.96
Sending payload: {"Distance in Centimeter":399.96}
Publish ok
Distance : 399.98
Sending payload: {"Distance in Centimeter":399.98}
Publish ok

```

The screenshot displays the IBM Watson IoT Platform interface. The top navigation bar includes tabs for 'Service Details - IBM Cloud', 'IBM Watson IoT Platform', and 'Wokwi - Wokwi Arduino and'. The main content area shows a list of devices. The 'ultrasonicsensor' device is selected, and its details are shown in a modal window. The 'Recent Events' tab is active, displaying a table of events.

Event	Value	Format	Last Received
Data	{"Distance in Centimeter":399.98}	json	a few seconds ago
Data	{"Distance in Centimeter":399.92}	json	a few seconds ago
Data	{"Distance in Centimeter":399.94}	json	a few seconds ago
Data	{"Distance in Centimeter":399.96}	json	a few seconds ago
Data	{"Distance in Centimeter":399.96}	json	a few seconds ago

WOKWI SHARE LINK

<https://wokwi.com/projects/347826727643251282>