

Assignment -4

Assignment Date	02 November 2022
Student Name	Ms. Megavarshini S
Student Roll Number	611219106043
Maximum Marks	2 Marks
Team ID	PNT2022TMID30278

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 = 27;
const int echoPin = 26;
//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 "hgcyjg"//IBM ORGANITION ID
#define DEVICE_TYPE "Distancesensor"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "Ultrasonic"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "ultrasonicsensor" //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 cm\":";
    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];
  }
}

```

Wokwi Output:

The screenshot displays the Wokwi web IDE interface. On the left, the 'sketch.ino' file contains the following code:

```

71  mqttconnect();
72  }
73  }
74
75  void PublishData(float centimeter) {
76    mqttconnect();//function call for connecting to ibm
77    /*
78     * creating the String in in form JSON to update the data to ibm cloud
79     */
80    String payload = "{\"Distance in cm\":";
81    payload += centimeter;
82    payload += "}";
83
84    Serial.print("Sending payload: ");
85    Serial.println(payload);
86
87
88    if (client.publish(PublishTopic, (char*) payload.c_str())) {
89      Serial.println("Publish ok");// if it successfully upload data on the cloud then it will
90    } else {
91      Serial.println("Publish failed");
92    }
93  }
94
95  }
96  void mqttconnect() {
97    if (!client.connected()) {
98      Serial.print("Reconnecting client to ");
99      Serial.println(server);
100      while (!client.connect(clientId, authMethod, token)) {
101        Serial.print(".");
102        delay(500);
103      }
104    }
105    initManagedDevice();
106    Serial.println();

```

On the right, the 'Simulation' window shows a visual representation of the hardware: an ESP32 microcontroller board connected to an HC-SR04 ultrasonic sensor. The sensor's VCC pin is connected to the ESP32's 5V pin, GND to GND, and the trigger pin to a digital pin. The output window below the simulation shows the following log:

```

Publish ok
Distance : 100.03
Sending payload: {"Distance in cm":100.03}
Publish ok
Distance : 99.99
Sending payload: {"Distance in cm":99.99}
Publish ok

```

IBM Cloud Alert:

The screenshot displays the IBM Watson IoT Platform interface. At the top, the header shows 'IBM Watson IoT Platform' and a user profile with email '2k19ece043@klot.ac.in' and ID 'hgyljg'. Below the header, a navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains icons for various IoT functions. The main content area shows a device named 'Ultrasonic' with status 'Connected'. The 'Recent Events' tab is active, displaying a table of events. The table has columns for 'Event', 'Value', 'Format', and 'Last Received'. The events are listed as 'Data' with values like '(*Distance in cm*:99.94)' and '(*Distance in cm*:100.03)', all in 'json' format, received 'a few seconds ago' or 'a minute ago'. A '0 Simulations running' status is shown at the bottom right.

Event	Value	Format	Last Received
Data	(*Distance in cm*:99.94)	json	a few seconds ago
Data	(*Distance in cm*:99.98)	json	a minute ago
Data	(*Distance in cm*:99.98)	json	a minute ago
Data	(*Distance in cm*:99.99)	json	a minute ago
Data	(*Distance in cm*:100.03)	json	a minute ago

Wokwi Share Link:

<https://wokwi.com/projects/347097236554908242>