

Assignment -4

Assignment Date	02 November 2022
Student Name	Shreemathi R N
Student Roll Number	73771914173
Maximum Marks	2 Marks
Team ID	PNT2022TMID11664

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 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 "z22obn"//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 "12345678" //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 image shows the Wokwi simulation environment. On the left, the code editor displays the Arduino sketch. On the right, the simulation window shows a virtual circuit with an ESP8266 module and an HC-SR04 ultrasonic sensor connected by wires. Below the circuit, the serial output console shows the following log:

```

subscribe to cmd OK
Distance (cm): 100.01
Distance (inch): 39.37
Sending payload: {"Distance (cm)":100.01}
Publish ok
Distance (cm): 99.94
Distance (inch): 39.35
Sending payload: {"Distance (cm)":99.94}
Publish ok

```

IBM Cloud Alert:

Event	Value	Format	Last Received
Data	{"Distance (cm)":99.98}	json	a few seconds ago
Data	{"Distance (cm)":99.96}	json	a few seconds ago
Data	{"Distance (cm)":99.98}	json	a few seconds ago
Data	{"Distance (cm)":99.98}	json	a few seconds ago
Data	{"Distance (cm)":99.98}	json	a few seconds ago

Wokwi Share Link:

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