

ASSIGNMENT – 4

ULTRASONIC SENSOR SIMULATION IN WOKWI AND IBM CLOUD

Assignment Date	22 October 2022
Student Name	Kishore R
Student Roll Number	212219040062
Maximum Marks	2 Marks

QUESTION:

Write code and connections in wokwi for the ultrasonic sensor. Whenever the distance is less than 100 cm send an “alert” to the IBM cloud and display in the device recent events.

CODE:

```
#include <WiFi.h>
#include <PubSubClient.h>

void callback(char* subscribetopic,byte* payload, unsigned int payloadLength);

#define ORG "i8xseg"
#define DEVICE_TYPE "ESP"
#define DEVICE_ID "2679"
#define TOKEN "9688811163"

String data3;

char server[]= ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[]="iot-2/evt/distance/fmt/json";
char subscribeTopic[]="iot-2/cmd/test/fmt/String";
char authMethod[]="use-token-auth";
char token[]=TOKEN;
char clientID[]="d:ORG:DEVICE_TYPE:DEVICE_ID;
```

```
WiFiClient wifiClient;

PubSubClient client(server,1883,callback,wifiClient);

#define ECHO_PIN 14
#define TRIG_PIN 12
#define led 27

void setup() {
    // put your setup code here, to run once:

    Serial.begin(115200);

    pinMode(led, OUTPUT);

    pinMode(TRIG_PIN, OUTPUT);

    pinMode(ECHO_PIN, INPUT);

    wificonnect();

    mqttconnect();
}

float readDistanceCM() {
    digitalWrite(TRIG_PIN, LOW);

    delayMicroseconds(2);

    digitalWrite(TRIG_PIN, HIGH);

    delayMicroseconds(10);

    digitalWrite(TRIG_PIN, LOW);

    int duration=random(1,200);

    //Serial.println(duration);

    //duration = pulseIn(ECHO_PIN, HIGH);

    return duration ;

    //Serial.println(duration);

}
```

```
void loop() {

    float distance = readDistanceCM();

    //Serial.println(distance);

    bool isNearby = distance < 100;
    digitalWrite(led, isNearby);

    Serial.print("Measured distance: ");
    Serial.println(distance);

    if(distance<100){

        PublishData2(distance);

    }else{

        PublishData1(distance);

    }

    //PublishData(distance);

    delay(1000);

    if(!client.loop()){

        mqttconnect();

    }

    //delay(2000);
}

void PublishData1(float dist){

    mqttconnect();

    String payload= "{\"distance\".";

    payload += dist;

    payload+="}";

    Serial.print("Sending payload:");
```

```

Serial.println(payload);

if(client.publish(publishTopic,(char*)payload.c_str())){

    Serial.println("publish ok");
} else{

    Serial.println("publish failed");
}
}

void PublishData2(float dist){

    mqttconnect();

    String payload= "{\"ALERT\":";

    payload += dist;

    payload+="}";

    Serial.print("Sending payload:");

    Serial.println(payload);

    if(client.publish(publishTopic,(char*)payload.c_str())){

        Serial.println("publish ok");
    } else{

        Serial.println("publish failed");
    }
}

void mqttconnect(){

    if(!client.connected()){

        Serial.print("Reconnecting to ");

        Serial.println(server);

        while(!!!client.connect(clientID, authMethod, token)){

            Serial.print(".");

            delay(500);

        }
    }
}

```

```

    initManagedDevice();

    Serial.println();
}
}

void wificonnect(){

    Serial.println();

    Serial.print("Connecting to");

    WiFi.begin("Wokwi-GUEST","",6);

    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++){

    data3 += (char)payload[i];

}

Serial.println("data:"+ data3);

if(data3=="lighton"){

    Serial.println(data3);

    digitalWrite(led,HIGH);

}else{

    Serial.println(data3);

    digitalWrite(led,LOW);

}

data3="";

}

```

Output:

Normal Case:

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

```

1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* subscribeTopic,byte* payload,unsigned int payloadLength){
4 #define ORG "i8xseg"
5 #define DEVICE_TYPE "ESP"
6 #define DEVICE_ID "2679"
7 #define TOKEN "9688811163"
8 String data3;
9
10 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
11 char publishTopic[] = "iot-2/evt/distance/fmt/json";
12 char subscribeTopic[] = "iot-2/cmd/test/fmt/String";
13 char authMethod[] = "use-token-auth";
14 char token[] = TOKEN;
15 char clientId[] = "d:" + ORG + ":" + DEVICE_TYPE + ":" + DEVICE_ID;
16
17 WiFiClient wifiClient;
18 PubSubClient client(server,1883,callback,wifiClient);
19
20 #define ECHO_PIN 14
21 #define TRIG_PIN 12
22 #define led 27
23
24 void setup() {
25 // put your setup code here, to run once:
26 Serial.begin(115200);
27 pinMode(led, OUTPUT);
28 }

```

The simulation window on the right shows the hardware setup and the execution log. The log output is as follows:

```

publish ok
Measured distance: 54.00
Sending payload:{"ALERT":54.00}
publish ok
Measured distance: 165.00
Sending payload:{"distance":165.00}
publish ok

```

Alert Case:

The screenshot shows the Wokwi IoT simulator interface. On the left, the 'sketch.ino' file is open, displaying the following code:

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* topic, byte* payload, unsigned int length) {
4   #define ORG "i8xseg"
5   #define DEVICE_TYPE "ESP"
6   #define DEVICE_ID "2679"
7   #define TOKEN "968881163"
8   String data;
9
10  char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
11  char publishTopic[] = "iot-2/evt/distance/fmt/json";
12  char subscribeTopic[] = "iot-2/cmd/test/fmt/String";
13  char authMethod[] = "use-token-auth";
14  char token[] = TOKEN;
15  char clientId[] = "d:" + ORG + ":" + DEVICE_TYPE + ":" + DEVICE_ID;
16
17  WiFiClient wifiClient;
18  PubSubClient client(server, 1883, callback, wifiClient);
19
20  #define ECHO_PIN 14
21  #define TRIG_PIN 12
22  #define led 27
23
24  void setup() {
25    // put your setup code here, to run once:
26    Serial.begin(115200);
27    pinMode(led, OUTPUT);
28  }
```

On the right, the 'Simulation' window shows a virtual circuit with an ESP32 microcontroller, an HC-SR04 ultrasonic sensor, and an LED. The console output shows the following sequence of events:

```
publish ok
Measured distance: 62.00
Sending payload:{"ALERT":62.00}
publish ok
Measured distance: 95.00
Sending payload:{"ALERT":95.00}
publish ok
```

IBM CLOUD STORAGE

The screenshot shows the IBM Watson IoT Platform dashboard. The 'Recent Events' tab is selected, displaying a table of events for the device 'i8xseg'. The table has the following columns: Event, Value, Format, and Last Received.

Event	Value	Format	Last Received
distance	{"ALERT":2}	json	a few seconds ago
distance	{"distance":115}	json	a few seconds ago
distance	{"ALERT":45}	json	a few seconds ago
distance	{"ALERT":57}	json	a few seconds ago
distance	{"ALERT":35}	json	a few seconds ago

Below the table, it states '0 Simulations running'.