

## Assignment -4

### WOWKI SIMULATION WITH IBM WATSON

Assignment Date	19 November 2022
Student Name	HARI PRASATH A S
Student Roll Number	111519106501
Maximum Marks	2 Marks

#### Question-1:

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

#### Solution :

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

WiFiClient wifiClient;

#define ORG "nhpwjc"
#define DEVICE_TYPE "raspberrypi"
#define DEVICE_ID "12345"
#define TOKEN "123456789"
#define speed 0.034

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char topic[] = "iot-2/cmd/home/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, wifiClient);
void publishData();

const int trigpin=5;
const int echopin=18;
String command;
String data="";

long duration;
int dist;

void setup()
{
  Serial.begin(115200);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, INPUT);
  wifiConnect();
  mqttConnect();
}

void loop() {
```

```

publishData();
delay(500);

if (!client.loop()) {
    mqttConnect();
}
}

void wifiConnect() {
    Serial.print("Connecting to "); Serial.print("Wifi");
    WiFi.begin("Wokwi-GUEST", "", 6);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.print("WiFi connected, IP address: "); Serial.println(WiFi.localIP());
}

void mqttConnect() {
    if (!client.connected()) {
        Serial.print("Reconnecting MQTT client to "); Serial.println(server);
        while (!client.connect(clientId, authMethod, token)) {
            Serial.print(".");
            delay(1000);
        }
        initManagedDevice();
        Serial.println();
    }
}

void initManagedDevice() {
    if (client.subscribe(topic)) {
        Serial.println(client.subscribe(topic));
        Serial.println("subscribe to cmd OK");
    } else {
        Serial.println("subscribe to cmd FAILED");
    }
}

void publishData()
{
    digitalWrite(trigpin, LOW);
    digitalWrite(trigpin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigpin, LOW);
    duration=pulseIn(echopin, HIGH);
    dist=duration*speed/2;

    if(dist<100){
        DynamicJsonDocument doc(1024);
        String payload;
        doc["AlertDistance:"]=dist;
        serializeJson(doc, payload);
        delay(3000);
        Serial.print("\n");
        Serial.print("Sending payload: ");
        Serial.println(payload);
        if (client.publish(publishTopic, (char*) payload.c_str())) {
            Serial.println("Publish OK");
        } else {
            Serial.println("Publish FAILED");
        }
    }
}
}

```

## IBM CLOUD OUTPUT

The screenshot displays the IBM Cloud IoT Platform interface. On the left, a sidebar contains navigation icons. The main area shows a table of devices with columns: Device ID, Status, Device Type, Class ID, and Date Added. A device with ID 12309 is selected, showing a status of 'Connected' and a device type of 'ThisDevice'. Below the table, the 'Recent Events' tab is active, displaying a list of events with columns: Event, Value, Format, and Last Received. The events are JSON payloads containing 'AlertDistance' values. On the right, a simulation panel shows '1/50 Simulations Running' and a 'New Simulation' button. Below this, a dropdown menu for 'Device Type' is open, showing 'ThisDevice' and a '1 Event' button. At the bottom, a status bar indicates '777 events sent' and '22.12 KB sent'.

Event	Value	Format	Last Received
event_hand	{"AlertDistance":44}	json	a few seconds
event_hand	{"AlertDistance":86}	json	a few seconds
event_hand	{"AlertDistance":52}	json	a few seconds
event_hand	{"AlertDistance":23}	json	a few seconds
event_hand	{"AlertDistance":40}	json	a few seconds

## WOWKI OUTPUT

The screenshot shows the Arduino IDE interface with a simulation window. The code in the main editor is for an ESP32 device connected to an HC-SR04 ultrasonic sensor. The code defines the sensor's pin configuration and sets up an MQTT client to publish distance data to the IBM Cloud IoT Platform. The simulation window on the right shows a 3D model of the ESP32 and the HC-SR04 sensor. Below the simulation, a console window displays the output of the simulation, showing the device sending a payload of {"AlertDistance":0} to the MQTT broker.

```

1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 #include <ArduinoJson.h>
4
5 WiFiClient wificlient;
6
7 #define ORG "luicm6"
8 #define DEVICE_TYPE "ThisDevice"
9 #define DEVICE_ID "12309"
10 #define TOKEN "V20lr-jK48p00Rrte"
11 #define speed 0.034
12
13
14 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
15 char publishTopic[] = "iot-2/evt/Data/fmt/json";
16 char topic[] = "iot-2/cmd/home/fmt/string";
17 char authMethod[] = "use-token-auth";
18 char token[] = TOKEN;
19 char clientId[] = "ds:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
20 PubSubClient client(server, 1883, wificlient);
21 void publishData();
22
23 const int trigpin=5;
24 const int echopin=18;
25 String command;
26 String data="";
27
28 long duration;
29 int dist;
30
31 void setup()
32 {
33   Serial.begin(115200);
34   pinMode(trigpin, OUTPUT);
35   pinMode(echopin, INPUT);
  
```

Simulation output:

```

Sending payload: {"AlertDistance":0}
Publish OK
Sending payload: {"AlertDistance":0}
Publish OK
  
```