

## ASSIGNMENT- 4

### DISTANCE DETECTION USING ULTRASONIC SENSOR

#### Question:

Write code and connections in wokwi for ultrasonic sensor. Whenever distance is less than 100 cms send "alert" to IBM cloud and display in device recent events.

#### CODE:

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

WiFiClient wifiClient;

#define ORG "1bklkq"
#define DEVICE_TYPE "abcd"
#define DEVICE_ID "rasp"
#define TOKEN "12345678"
#define speed 0.034

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/abcd_1/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="";
String lat="14.167589";
String lon="80.248510";
String name="point2";
String icon="";
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 {

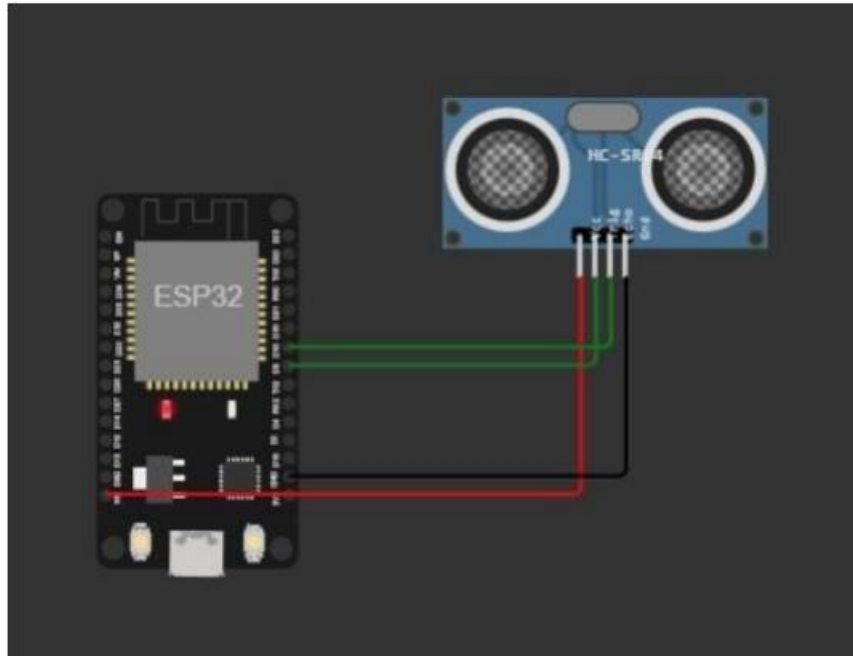
```

```

        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){
        dist=100-dist;
        icon="fa-trash";
    }else{
        dist=0;
        icon="fa-trash-o";
    }
    DynamicJsonDocument doc(1024);
    String payload;
    doc["Name"]=name;
    doc["Latitude"]=lat;
    doc["Longitude"]=lon;
    doc["Icon"]=icon;
    doc["FillPercent"]=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");
    }
}
}

```

## CONNECTIONS:



## OUTPUT:

```

1  sketch.ino  diagram.json  libraries.txt  Library Manager
2  #include <Arduino.h>
3  #include <ArduinoJson.h>
4
5  WiFiClient wificlient;
6
7  #define ORG "ibklkq"
8  #define DEVICE_TYPE "abcd"
9  #define DEVICE_ID "rasp"
10 #define TOKEN "12345678"
11 #define speed 0.034
12
13 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
14 char publishTopic[] = "iot-2/evt/abcd_1/fmt/json";
15 char topic[] = "iot-2/cmd/home/fmt/String";
16 char authMethod[] = "use-token-auth";
17 char token[] = TOKEN;
18 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
19 PubSubClient client(server, 1883, wificlient);
20 void publishData();
21
22 const int trigpin=5;
23 const int echopin=18;
24 String command;
25 String data="";
26 String lat="14.167589";
27 String lon="80.248510";
28 String name="point2";
29 String icon="";
30
31 long duration;
32 int dist;
33
34 void setup()
35 {
36   Serial.begin(115200);

```

Simulation
00:40.678

o", "FillPercent": 0}

Publish OK

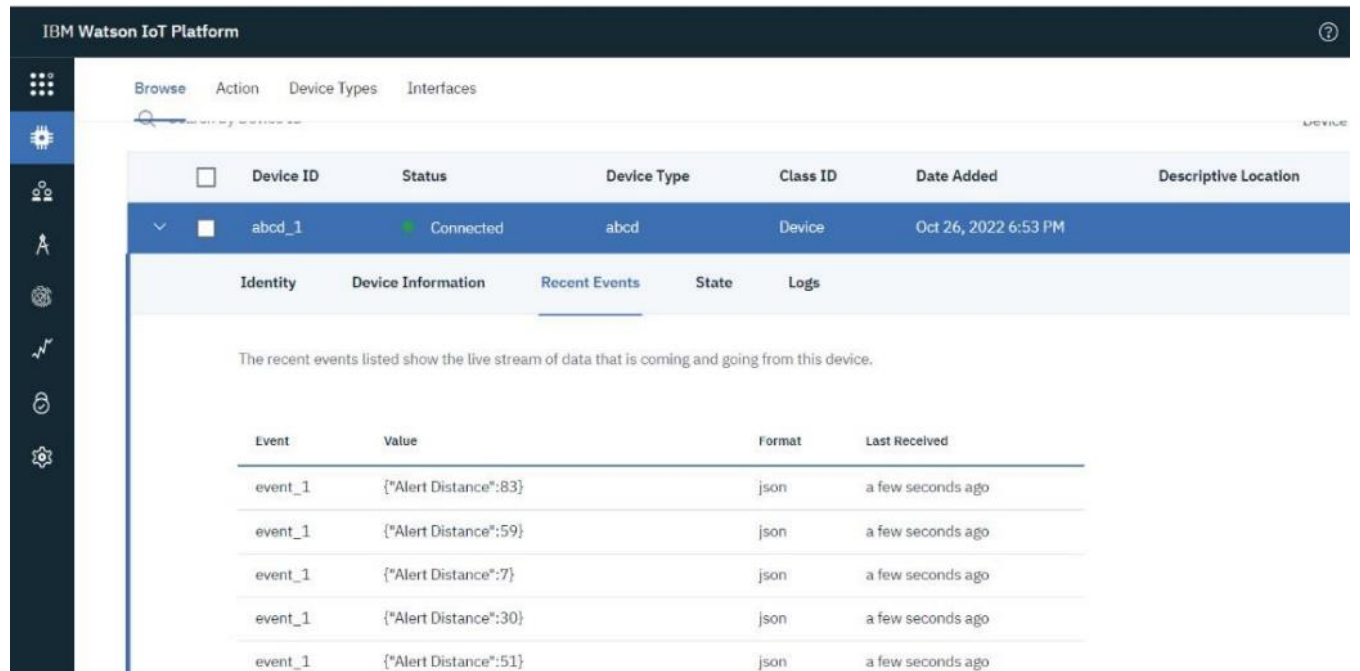
Sending payload:

```
{
  "Name": "point2",
  "Latitude": "14.167589",
  "Longitude": "80.248510",
  "Icon": "fa-trash-o",
  "FillPercent": 0
}
```

Publish OK

26 October

## Data sent to the IBM Cloud Device when the object is near



The screenshot displays the IBM Watson IoT Platform interface. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. The left sidebar contains various icons for navigation. The main content area shows a table of devices with columns: Device ID, Status, Device Type, Class ID, Date Added, and Descriptive Location. A device named 'abcd\_1' is highlighted, showing a status of 'Connected'. Below the device list, there are tabs for 'Identity', 'Device Information', 'Recent Events', 'State', and 'Logs'. The 'Recent Events' tab is active, displaying a message: 'The recent events listed show the live stream of data that is coming and going from this device.' Below this message is a table of recent events with columns: Event, Value, Format, and Last Received.

| Event   | Value                 | Format | Last Received     |
|---------|-----------------------|--------|-------------------|
| event_1 | {"Alert Distance":83} | json   | a few seconds ago |
| event_1 | {"Alert Distance":59} | json   | a few seconds ago |
| event_1 | {"Alert Distance":7}  | json   | a few seconds ago |
| event_1 | {"Alert Distance":30} | json   | a few seconds ago |
| event_1 | {"Alert Distance":51} | json   | a few seconds ago |