

## ASSIGNMENT IV

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.

### CODE:

sketch.ino

```
#include <WiFi.h>
#include<PubSubClient.h>
WiFiClient wifiClient;
String data3;
#define ORG "x7ay23"
#define DEVICE_TYPE "ESP32"
#define DEVICE_ID "assignment4"
#define TOKEN "1123581321"
#define speed 0.034
#define led 14
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/data/fmt/json";
char topic[] = "iot-2/cmd/command/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, wifiClient);

const int trigpin=5;
const int echopin=18;
String command;
String data="";
long duration;
float dist;

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

```

void loop() {
bool isNearby = dist < 100;
digitalWrite(led, isNearby);
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(500);
}
}
initManagedDevice();
Serial.println();
}
}

void initManagedDevice() {
if (client.subscribe(topic)) {
// Serial.println(client.subscribe(topic));
Serial.println("IBM 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){

```

```

String payload = "{\"Alert Distance\":\"";
payload += dist;
payload += "}";
Serial.print("\n");
Serial.print("Sending payload: ");
Serial.println(payload);
if (client.publish(publishTopic, (char*) payload.c_str()))
{ Serial.println("Publish OK");
}
}
if(dist>100){
String payload = "{\"Distance\":\"";
payload += dist;
payload += "}";
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");
}
}
}
}

```

diagram.json

```

{
  "version": 1,
  "author": "HARSHITHA S S 19CS057",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": -13.33, "left": -70,
"attrs": { } },
    { "type": "wokwi-hc-sr04", "id": "ultrasonic1", "top": -35.57, "left": 84.83,
"attrs": { } }
  ],
  "connections": [
    [ "esp:TX0", "$serialMonitor:RX", "", [ ] ],
    [ "esp:RX0", "$serialMonitor:TX", "", [ ] ],
    [ "ultrasonic1:VCC", "esp:VIN", "red", [ "v154.01", "h-243.35", "v-67.77" ] ],
    [ "ultrasonic1:GND", "esp:GND.1", "black", [ "v0" ] ],
    [ "ultrasonic1:TRIG", "esp:D5", "green", [ "v0" ] ],
    [ "ultrasonic1:ECHO", "esp:D18", "green", [ "v0" ] ]
  ]
}

```

## CONNECTION DIAGRAM:

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 WiFiClient wificlient;
4 String data3;
5 #define ORG "x7ay23"
6 #define DEVICE_TYPE "ESP32"
7 #define DEVICE_ID "assignment4"
8 #define TOKEN "1123581321"
9 #define speed 0.034
10 #define led 14
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/data/fmt/json";
13 char topic[] = "iot-2/cmd/command/fmt/String";
14 char authMethod[] = "use-token-auth";
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
17 PubSubClient client(server, 1883, wificlient);
18
19 const int trigpin=5;
20 const int echopin=18;
21 String command;
22 String data="";
23 long duration;
24 float dist;
25
26 void setup()
27 {
28   Serial.begin(115200);
29   pinMode(led, OUTPUT);
30   pinMode(trigpin, OUTPUT);
31   pinMode(echopin, INPUT);
32   wifiConnect();
33   mqttConnect();
34 }
35
```

On the right, the 'Simulation' window shows a 3D model of an ESP32 microcontroller connected to an HC-SR04 ultrasonic sensor. Below the simulation, the console output shows the following messages:

```
Publish OK
Sending payload: {"Alert Distance":41.94}
Publish OK
Sending payload: {"Alert Distance":41.94}
Publish OK
```

## IBM CLOUD SCREENSHOT:

The screenshot shows the IBM Watson IoT Platform dashboard. The 'Browse' tab is selected, and the 'assignment4' device is highlighted. The device status is 'Disconnected'. The 'Recent Events' tab is active, displaying a list of events:

| Event | Value                    | Format | Last Received     |
|-------|--------------------------|--------|-------------------|
| data  | {"Alert Distance":41.94} | json   | a few seconds ago |
| data  | {"Alert Distance":41.94} | json   | a few seconds ago |
| data  | {"Alert Distance":41.97} | json   | a few seconds ago |
| data  | {"Distance":123.98}      | json   | a few seconds ago |
| data  | {"Distance":123.98}      | json   | a few seconds ago |

**WOKWI LINK:** <https://wokwi.com/projects/348677162727899730>