

Assignment – 4
Distance Detection using Ultrasonic Sensor

Name	Kaiser A
Team ID	PNT2022TMID14815
Roll Number	7179KCTKCTKCTKCTKCTKCTKCT19BEC014

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.

Wokwi Link:

<https://wokwi.com/projects/346678740879671892>

Code:

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

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

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

```
//-----IBM Credentials-----//
```

```
#define ORG "3747gc"//IBM ORGANITION ID
```

```
#define DEVICE_TYPE "kaiser"//Device type mentioned in ibm watson IOT Platform
```

```
#define DEVICE_ID "esp32"//Device ID mentioned in ibm watson IOT Platform
```

```
#define TOKEN "12345678" //Token
```

```
String sub_data;
```

```
float distance;
```

```
//-----Server Setup-----//
```

```
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
```

```
//-----Main Code-----//
```

```
WiFiClient wificlient; //Ceating instance for WifiClient
```

```
PubSubClient client(server, 1883, callback , wificlient); //mqtt Client
```

```
int Led = 4;
```

```
int trig = 5;
```

```
int echo = 18;
```

```
void setup() {
```

```
  Serial.begin(115200);
```

```
  pinMode(trig, OUTPUT);
```

```
  pinMode(Led, OUTPUT);
```

```
  pinMode(echo, OUTPUT);
```

```
  delay(10);
```

```
  wificonnect();
```

```
  mqttconnect();
```

```
}
```

```
void loop() {
```

```
  digitalWrite(trig, LOW);
```

```
  digitalWrite(trig, HIGH);
```

```
  delayMicroseconds(10);
```

```
  digitalWrite(trig, LOW);
```

```
  float duration = pulseIn(echo, HIGH);
```

```
  float distance = (duration * 0.0343) / 2;
```

```
  Serial.print("Distance in Cm = ");
```

```
  Serial.println(distance);
```

```

Publish_Data(distance);

delay(1000);

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

```

```

void Publish_Data(float dist) {
    mqttconnect(); //Connect to Server

```

```

/* Creating the String in JSON format to send to the Cloud
   according to the distance from the Ultrasonic Sensor*/

```

```

String object;
if (dist < 100) {
    digitalWrite(Led, HIGH);
    Serial.println("Object is Near");
    object = "Near";
}

```

```

else {
    digitalWrite(Led, LOW);
    Serial.println("No Object Found");
    object = "No Object";
}

```

```

String payload = "{\"Distance\":";
payload += dist;
payload += "," "\"object\":";
payload += object;
payload += "\"}";

```

```

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

//Publish payload Message

if (client.publish(publishTopic , (char*) payload.c_str())) {
    Serial.println("Publish OK");
} else {
    Serial.println("Publish Failed");
}
Serial.println("");
}

//-----User Fuctions-----//

//Connect to Mqtt

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]);

```

```

sub_data += (char)payload[i];
}

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

sub_data="";
}

```

Output:

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

```

1 #include <WiFi.h>
2 #include <PubSubClient.h>
3
4 void callback(char* topic, byte* payload, unsigned int payloadLength);
5
6 //----- IBM Credentials-----//
7
8 #define ORG "3747gc"//IBM ORGANIZATION ID
9 #define DEVICE_TYPE "kaiser"//Device type mentioned in ibm watson IOT Platform
10 #define DEVICE_ID "esp32"//Device ID mentioned in ibm watson IOT Platform
11 #define TOKEN "12345678" //token
12 String sub_data;
13 float distance;
14
15 //-----Server Setup-----//
16
17 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
18 char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform
19 char subscribTopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT command type AND
20 char authMethod[] = "use-token-auth";// authentication method
21 char token[] = TOKEN;
22 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
23
24
25 //-----Main Code-----//
26 WiFiClient wificlient; //Creating instance for WiFi
27 PubSubClient client(server, 1883, callback, wificlient); //mqtt client
28
29 int Led = 4;
30 int trig = 5;
31 int echo = 18;
32
33 void setup() {
34   pinMode(trig, OUTPUT);
35   pinMode(echo, INPUT);
36 }
37
38 void loop() {
39   digitalWrite(trig, HIGH);
40   delayMicroseconds(1000);
41   digitalWrite(trig, LOW);
42   delayMicroseconds(5);
43   float distance = 0;
44   while (digitalRead(echo) == LOW) {
45     distance = 2 * pulseIn(trig, HIGH) / 58;
46   }
47   sub_data += (char)payload[i];
48   Serial.println("data: "+ sub_data);
49   sub_data="";
50 }

```

On the right, the simulation window shows an ESP32 board connected to an Ultrasonic Distance Sensor. The sensor's distance is 290cm. The console output shows the following messages:

```

Publish OK
Distance in Cm = 292.53
No Object Found
Sending payload: {"Distance":292.53,"object":"No Object"}
Publish OK

```

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

```

1 #include <WiFi.h>
2 #include <PubSubClient.h>
3
4 void callback(char* topic, byte* payload, unsigned int payloadLength);
5
6 //----- IBM Credentials-----//
7
8 #define ORG "3747gc"//IBM ORGANIZATION ID
9 #define DEVICE_TYPE "kaiser"//Device type mentioned in ibm watson IOT Platform
10 #define DEVICE_ID "esp32"//Device ID mentioned in ibm watson IOT Platform
11 #define TOKEN "12345678" //token
12 String sub_data;
13 float distance;
14
15 //-----Server Setup-----//
16
17 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
18 char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform
19 char subscribTopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT command type AND
20 char authMethod[] = "use-token-auth";// authentication method
21 char token[] = TOKEN;
22 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
23
24
25 //-----Main Code-----//
26 WiFiClient wificlient; //Creating instance for WiFi
27 PubSubClient client(server, 1883, callback, wificlient); //mqtt client
28
29 int Led = 4;
30 int trig = 5;
31 int echo = 18;
32
33 void setup() {
34   pinMode(trig, OUTPUT);
35   pinMode(echo, INPUT);
36 }
37
38 void loop() {
39   digitalWrite(trig, HIGH);
40   delayMicroseconds(1000);
41   digitalWrite(trig, LOW);
42   delayMicroseconds(5);
43   float distance = 0;
44   while (digitalRead(echo) == LOW) {
45     distance = 2 * pulseIn(trig, HIGH) / 58;
46   }
47   sub_data += (char)payload[i];
48   Serial.println("data: "+ sub_data);
49   sub_data="";
50 }

```

On the right, the simulation window shows an ESP32 board connected to an Ultrasonic Distance Sensor. The sensor's distance is 50cm. The console output shows the following messages:

```

Publish OK
Distance in Cm = 50.40
Object is Near
Sending payload: {"Distance":50.40,"object":"Near"}
Publish OK

```

IBM Watson IoT Platform

7179kctkctkctkctkctkct19bec014@smartinternz.com
ID: 3747gc

?

8

Browse

Action

Device Types

Interfaces

Add Device

esp32

Connected

kaiser

→ ...

Identity

Device Information

Recent Events

State

Logs

×

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
Data	{"Distance":292.53,"object":"No Object"}	json	a few seconds ago
Data	{"Distance":292.53,"object":"No Object"}	json	a few seconds ago
Data	{"Distance":292.53,"object":"No Object"}	json	a few seconds ago
Data	{"Distance":295.53,"object":"No Object"}	json	a few seconds ago
Data	{"Distance":281.43,"object":"No Object"}	json	a few seconds ago

Items per page 50 | 1-1 of 1 item

1 of 1 page

<

1

>

IBM Watson IoT Platform

7179kctkctkctkctkctkct19bec014@smartinternz.com
ID: 3747gc

?

8

Browse

Action

Device Types

Interfaces

Add Device

esp32

Connected

kaiser

→ ...

Identity

Device Information

Recent Events

State

Logs

×

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
Data	{"Distance":50.42,"object":"Near"}	json	a few seconds ago
Data	{"Distance":50.42,"object":"Near"}	json	a few seconds ago
Data	{"Distance":50.42,"object":"Near"}	json	a few seconds ago
Data	{"Distance":50.42,"object":"Near"}	json	a few seconds ago
Data	{"Distance":50.39,"object":"Near"}	json	a few seconds ago

Items per page 50 | 1-1 of 1 item

1 of 1 page

<

1

>