

Assignment - 4

Distance Detection Using Ultra Sonic Sensor

Assignment Date	19 October 2022
Student Name	Mr. Syed Ali Novfal
Student Roll Number	110119106029
Maximum Marks	2 Marks

Question 1:

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/346228322074624596>

CODE:

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQTT

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

//-----credentials of IBM Accounts-----

#define ORG "f59trs"//IBM ORGANITION ID
#define DEVICE_TYPE "ultrasonicsensor"//Device type mentioned in ibm watson
IOT Platform
#define DEVICE_ID "distancedetection"//Device ID mentioned in ibm watson IOT
Platform
#define TOKEN "AlGMGaaF01nawa1QA3" //Token
String data3;
float dist;

//----- Customise the above values -----
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
```

```

//-----
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the
predefined client id by passing parameter like server id,portand
wificredential

int LED = 4;
int trig = 5;
int echo = 18;
void setup()
{
  Serial.begin(115200);
  pinMode(trig,OUTPUT);
  pinMode(echo,INPUT);
  pinMode(LED, OUTPUT);
  delay(10);
  wificonnect();
  mqttconnect();
}
void loop()// Recursive Function
{

  digitalWrite(trig,LOW);
  digitalWrite(trig,HIGH);
  delayMicroseconds(10);
  digitalWrite(trig,LOW);
  float dur = pulseIn(echo,HIGH);
  float dist = (dur * 0.0343)/2;
  Serial.print ("Distancein cm");
  Serial.println(dist);

  PublishData(dist);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}

/*.....retrieving to
Cloud.....*/

void PublishData(float dist) {
  mqttconnect();//function call for connecting to ibm
  /*
   creating the String in in form JSon to update the data to ibm cloud
  */

```

```

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";
}

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

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

if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");// if it sucessfully upload data on the cloud
    then it will print publish ok in Serial monitor or else it will print publish
    failed
} else {
    Serial.println("Publish failed");
}
}

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]);
        data3 += (char)payload[i];
    }

    // Serial.println("data: "+ data3);
    // if(data3=="Near")
    // {
    // Serial.println(data3);
    // digitalWrite(LED,HIGH);

    // }

    // else
    // {
    // Serial.println(data3);

```

```
// digitalWrite(LED,LOW);

// }
data3=" ";

}
```

OUTPUT:

When the object is not near to ultrasonic sensor

WOKWI

SAVE

SHARE

Docs

sketch.ino

diagram.json

libraries.txt

Library Manager

```

1  #include <WiFi.h> //library for wifi
2  #include <PubSubClient.h> //library for MQTT
3
4
5  void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
6
7  //-----credentials of IBM Accounts-----
8
9  #define ORG "ocozu" //IBM ORGANITION ID
10 #define DEVICE_TYPE "ultrasonicsensor" //Device type mentioned in ibm watson IOT Platform
11 #define DEVICE_ID "distancedetection" //Device ID mentioned in ibm watson IOT Platform
12 #define TOKEN "TXGe6EJO0bmt&kwlnB" //Token
13 String data3;
14 float dist;
15
16
17 //----- Customise the above values -----
18 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
19 char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of event perform
20 char subscribetopic[] = "iot-2/cmd/test/fmt/String"; // cmd REPRESENT command type AND
21 char authMethod[] = "use-token-auth"; // authentication method
22 char token[] = TOKEN;
23 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
24
25
26 //-----
27 WiFiClient wificlient; // creating the instance for wificlient
28 PubSubClient client(server, 1883, callback, wificlient); //calling the predefined client
29
30 int LED = 4;
31 int trig = 5;
32 int echo = 18;
33 void setup()
34 {
35   Serial.begin(115200);
```

Simulation

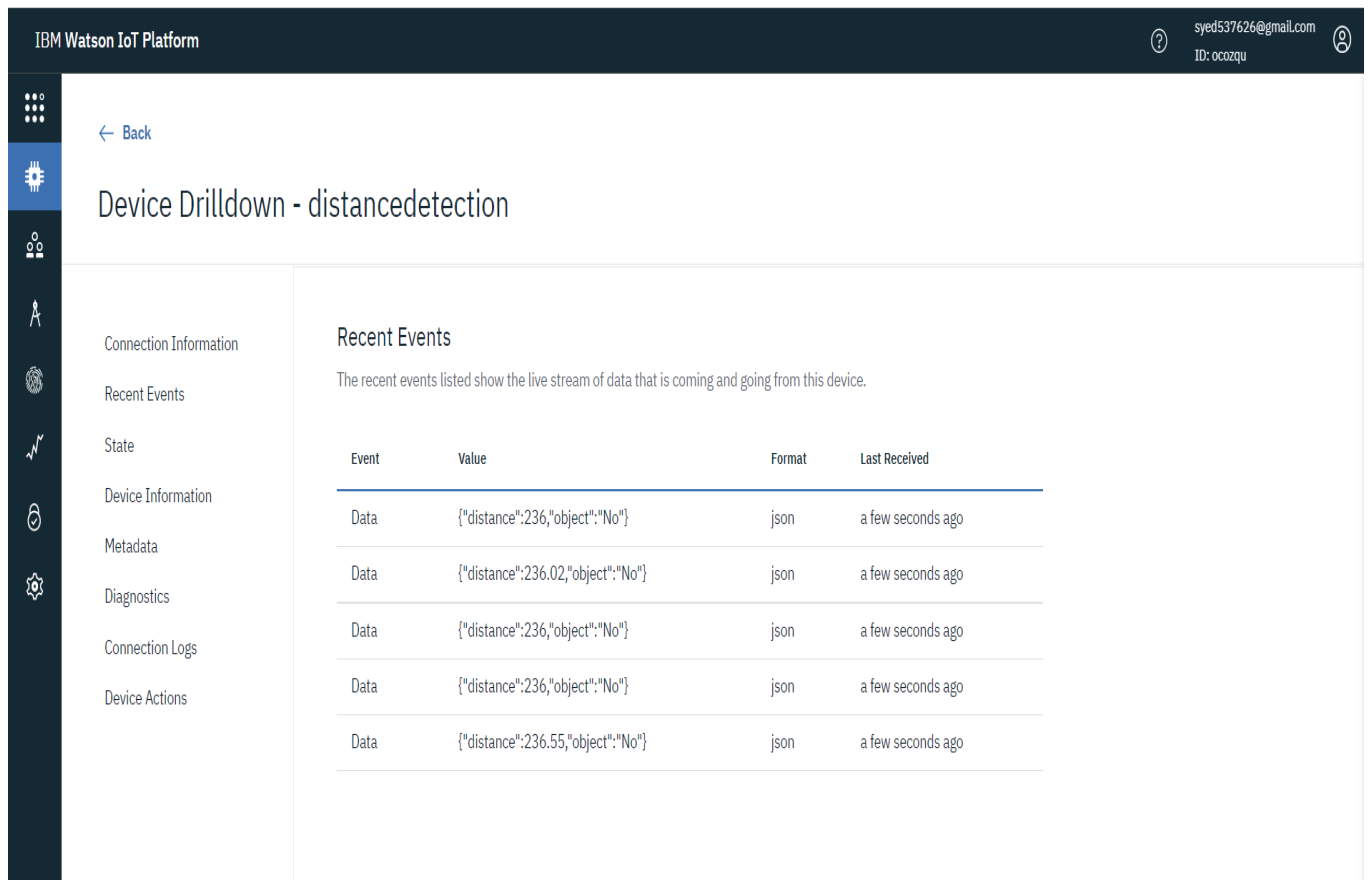
01:18.211

99%

```

no object found
Sending payload: {"distance":236.00,"object":"No"}
Publish ok
Distancein cm236.02
no object found
Sending payload: {"distance":236.02,"object":"No"}
Publish ok
```

Data sent to the IBM Cloud device when the object is far



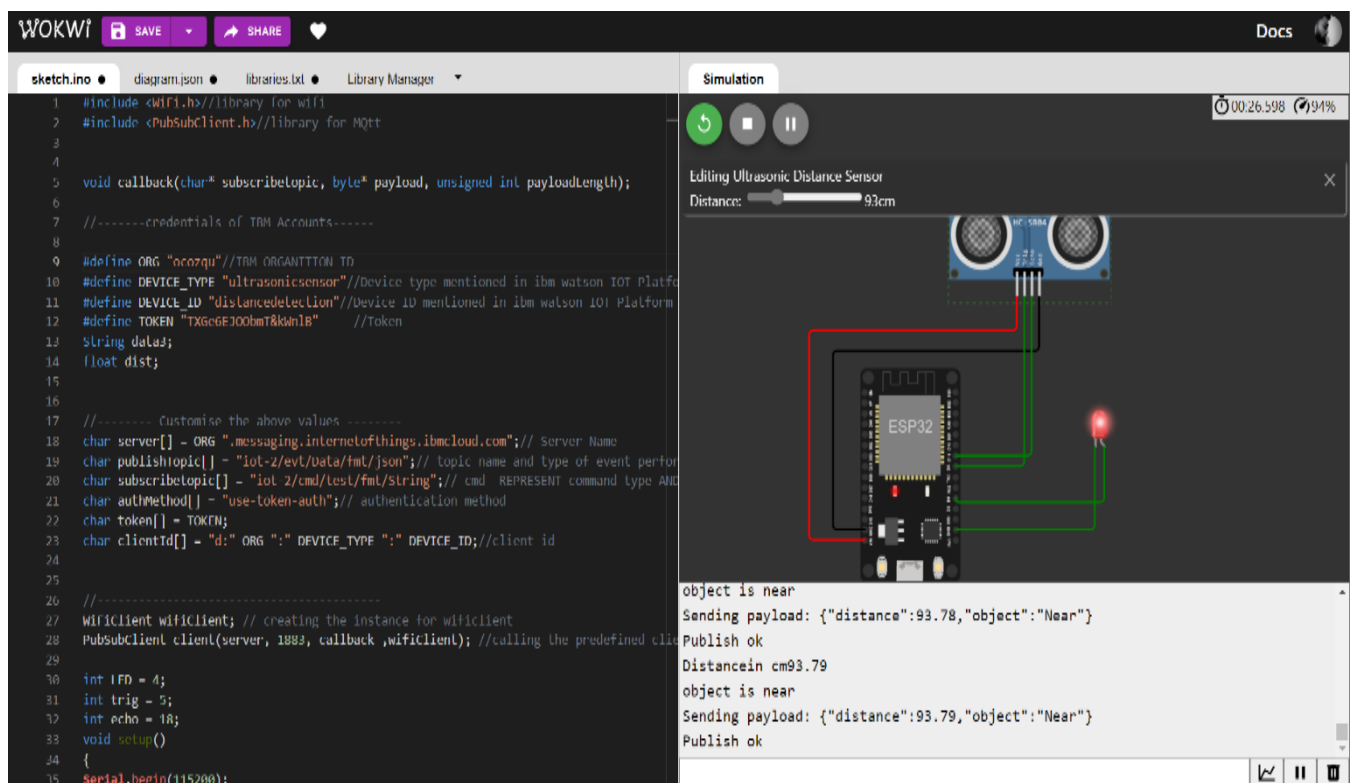
The screenshot shows the IBM Watson IoT Platform interface. The top navigation bar includes the IBM logo, the text "IBM Watson IoT Platform", and user information: "syeds37626@gmail.com" and "ID: ocozqu". A left sidebar contains icons for various functions. The main content area is titled "Device Drilldown - distancedetection". On the left, a vertical menu lists: "Connection Information", "Recent Events", "State", "Device Information", "Metadata", "Diagnostics", "Connection Logs", and "Device Actions". The "Recent Events" section is active, displaying a table of recent data events.

Recent Events

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":236,"object":"No"}	json	a few seconds ago
Data	{"distance":236.02,"object":"No"}	json	a few seconds ago
Data	{"distance":236,"object":"No"}	json	a few seconds ago
Data	{"distance":236,"object":"No"}	json	a few seconds ago
Data	{"distance":236.55,"object":"No"}	json	a few seconds ago

When the object is nearer to the ultrasonic sensor



The screenshot shows the WOKWI simulation environment. The top bar includes the "WOKWI" logo, "SAVE", "SHARE", and "Docs" buttons. The left sidebar shows "sketch.ino" selected. The main area is split into two panes. The left pane displays the Arduino code for the ESP32, which includes comments about IBM Cloud credentials and device configuration. The right pane shows a simulation of the hardware, including an ESP32 microcontroller, an ultrasonic sensor, and a red LED. A "Simulation" window is open, showing the sensor's distance as 93cm. Below the simulation, a console window displays the output of the program, showing the sensor detecting an object as "near" and sending a payload to the cloud.

```
1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3
4
5 void callback(char* subscribeTopic, byte* payload, unsigned int payloadLength);
6
7 //-----credentials of IBM Accounts-----
8
9 #define ORG "ocozqu" //TRM ORGANIZATION ID
10 #define DEVICE_TYPE "ultrasonicsensor" //Device type mentioned in ibm watson IoT Platform
11 #define DEVICE_ID "distancedetection" //Device ID mentioned in ibm watson IoT Platform
12 #define TOKEN "TXGGE700bmt&kin18" //Token
13
14 String data;
15 float dist;
16
17 //----- Customise the above values -----
18 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
19 char publishTopic[] = "iot-2/evt/data/fmt/json"; // topic name and type of event performed
20 char subscribeTopic[] = "iot-2/cmd/test/fmt/string"; // cmd REPRESENT command type AND
21 char authMethod[] = "use-token-auth"; // authentication method
22 char token[] = TOKEN;
23 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
24
25
26 //-----
27 WiFiClient wifiClient; // creating the instance for wifiClient
28 PubSubClient client(server, 1883, callback, wifiClient); //calling the predefined client
29
30 int IFD = 4;
31 int trig = 5;
32 int echo = 18;
33 void setup()
34 {
35   Serial.begin(115200);
```

Simulation

Editing Ultrasonic Distance Sensor
Distance: 93cm

object is near
Sending payload: {"distance":93.78,"object":"Near"}
Publish ok
Distance in cm 93.79
object is near
Sending payload: {"distance":93.79,"object":"Near"}
Publish ok

Data sent to the IBM Cloud device when the device is near

The screenshot displays the IBM Watson IoT Platform interface. At the top, the header shows 'IBM Watson IoT Platform' and user information: 'syed537626@gmail.com' and 'ID: ocozqu'. A sidebar on the left contains navigation icons and labels: 'Connection Information', 'Recent Events', 'State', 'Device Information', 'Metadata', 'Diagnostics', 'Connection Logs', and 'Device Actions'. The main content area is titled 'Device Drilldown - distancedetection' and features a 'Recent Events' section. This section includes a descriptive text: 'The recent events listed show the live stream of data that is coming and going from this device.' Below the text is a table with four columns: 'Event', 'Value', 'Format', and 'Last Received'. The table contains six rows of data, all showing 'Data' events with JSON values indicating a distance of 93.78 and a 'Near' object status, all in 'json' format, and received 'a few seconds ago'.

Event	Value	Format	Last Received
Data	{"distance":93.79,"object":"Near"}	json	a few seconds ago
Data	{"distance":93.78,"object":"Near"}	json	a few seconds ago
Data	{"distance":93.78,"object":"Near"}	json	a few seconds ago
Data	{"distance":93.78,"object":"Near"}	json	a few seconds ago
Data	{"distance":93.78,"object":"Near"}	json	a few seconds ago
Data	{"distance":93.78,"object":"Near"}	json	a few seconds ago

<https://wokwi.com/projects/346228322074624596>