

**Assignment -4**  
Distance Detection Using Ultrasonic Sensor

Assignment Date	19 October 2022
Student Name	Mr. Mohammed Ishaq
Student Roll Number	110119106013
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/346017500771648082>

**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 "9a7os9" //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 "r8*G0FF!4miEvwlQ7Q" //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 object is not near to the ultrasonic sensor**

The screenshot displays the Wokwi web IDE interface. On the left, the code editor shows the following code:

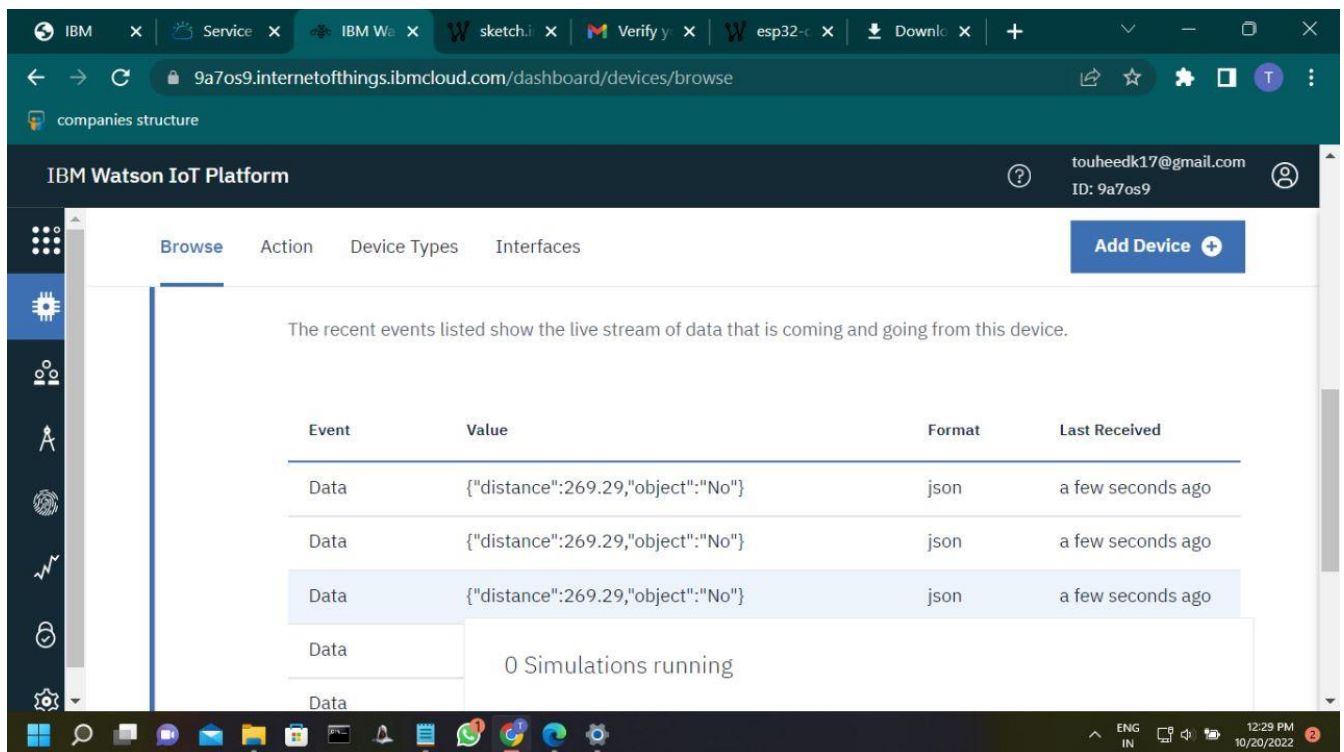
```
22 char token[] = TOKEN;  
23 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;  
24  
25  
26 //-----  
27 WiFiClient wifiClient; // creating the instance for wifi  
28 PubSubClient client(server, 1883, callback ,wifiClient)  
29  
30 int LED = 4;  
31 int trig = 5;  
32 int echo = 18;  
33 void setup()  
34 {  
35   Serial.begin(115200);  
36   pinMode(trig,OUTPUT);  
37   pinMode(echo,INPUT);  
38   pinMode(LED, OUTPUT);  
39   delay(10);  
40 }
```

On the right, the simulation window shows a circuit diagram of an ESP32 microcontroller connected to an ultrasonic sensor. The output log displays the following messages:

```
no object found  
Sending payload: {"distance":269.29,"object":"No"}  
Publish ok
```

The simulation window also shows a timer at 00:57.764 and a battery level at 98%.

## Data sent to the IBM cloud device when the object is far



The screenshot displays the IBM Watson IoT Platform dashboard. The browser address bar shows the URL `9a7os9.internetofthings.ibmcloud.com/dashboard/devices/browse`. The dashboard header includes the IBM Watson IoT Platform logo and the user's email `touheedk17@gmail.com` with ID `9a7os9`. The main content area is titled "Browse" and shows a table of recent events. The table has four columns: Event, Value, Format, and Last Received. The data shows three identical rows of "Data" events with the value `{"distance":269.29,"object":"No"}` in "json" format, received "a few seconds ago". A tooltip indicates "0 Simulations running".

Event	Value	Format	Last Received
Data	<code>{"distance":269.29,"object":"No"}</code>	json	a few seconds ago
Data	<code>{"distance":269.29,"object":"No"}</code>	json	a few seconds ago
Data	<code>{"distance":269.29,"object":"No"}</code>	json	a few seconds ago

0 Simulations running

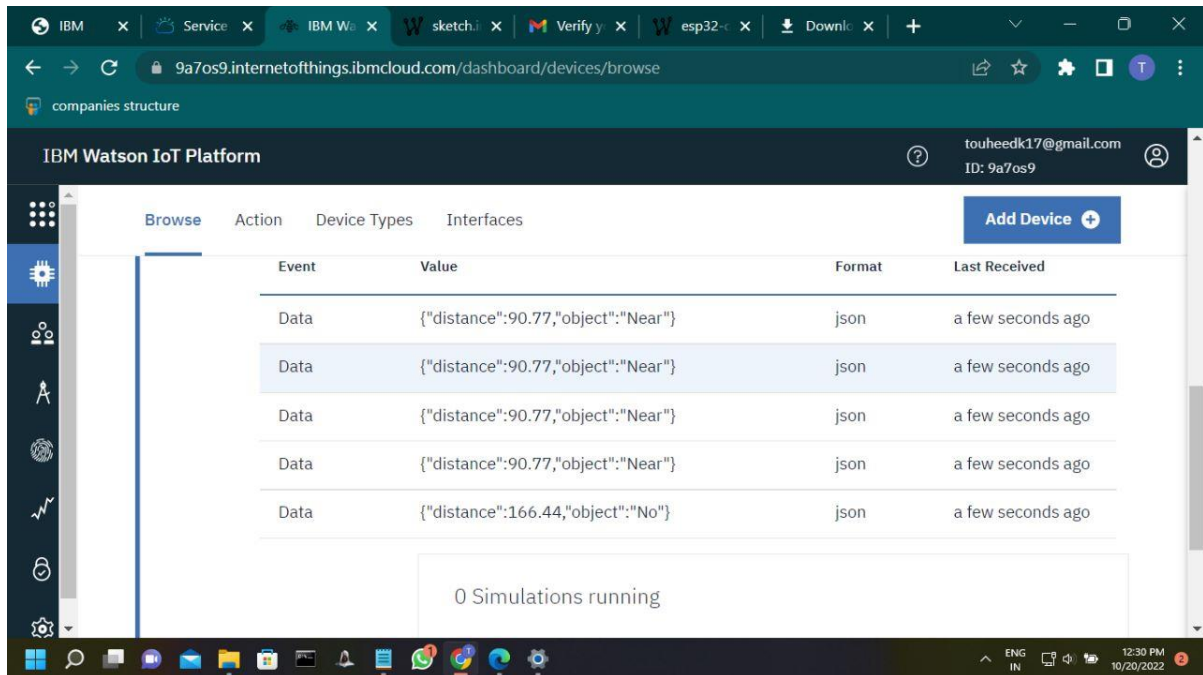
## When object is nearer to the ultrasonic sensor

The screenshot displays the Wokwi online IDE interface. On the left, the code for `esp32-dht22.ino` is visible, showing the setup for an ESP32 microcontroller connected to an ultrasonic sensor and an LED. The code includes pin definitions, library includes, and logic to trigger the LED when an object is near.

```
22 char token[] = TOKEN;
23 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_I
24
25
26 //-----
27 WiFiClient wifiClient; // creating the instance for wifi
28 PubSubClient client(server, 1883, callback ,wifiClient)
29
30 int LED = 4;
31 int trig = 5;
32 int echo = 18;
33 void setup()
34 {
35   Serial.begin(115200);
36   pinMode(trig, OUTPUT);
37   pinMode(echo, INPUT);
38   pinMode(LED, OUTPUT);
39   delay(10);
40 }
```

The right side of the interface shows a simulation of the hardware. An ESP32 board is connected to an ultrasonic sensor (HC-SR04) and a red LED. The simulation is running, with a timer at 00:43.798 and 92% battery. The console output shows the message "object is near" and the JSON payload: `{"distance":82.70,"object":"Near"}`. The status "Publish ok" is also displayed.

## Data sent to the IBM cloud device when the object is near



The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes the IBM logo and several open tabs: Service, IBM W..., sketch.i..., Verify y..., esp32-c..., and Downlo... The address bar displays the URL `9a7os9.internetofthings.ibmcloud.com/dashboard/devices/browse`. The dashboard header shows the user's email `touheedk17@gmail.com` and ID `9a7os9`. The main content area has a sidebar with icons for various functions. The central table displays data events with the following columns: Event, Value, Format, and Last Received. The table contains five rows of data, all with the format 'json' and 'a few seconds ago' as the last received time. The first four rows have a value of `{"distance":90.77,"object":"Near"}`, and the fifth row has a value of `{"distance":166.44,"object":"No"}`. Below the table, a status box indicates '0 Simulations running'.

Event	Value	Format	Last Received
Data	<code>{"distance":90.77,"object":"Near"}</code>	json	a few seconds ago
Data	<code>{"distance":90.77,"object":"Near"}</code>	json	a few seconds ago
Data	<code>{"distance":90.77,"object":"Near"}</code>	json	a few seconds ago
Data	<code>{"distance":90.77,"object":"Near"}</code>	json	a few seconds ago
Data	<code>{"distance":166.44,"object":"No"}</code>	json	a few seconds ago

0 Simulations running

<https://wokwi.com/projects/346017500771648082>



