

Assignment - 4
ESP 32 – Ultrasonic Sensor

Assignment Date	3 NOVEMBER 2022
Student Name	N.VARSHA
Student Roll Number	411719106059
Maximum Marks	2 Marks

Question-1:

Write code and Connection in wokwi for ultrasonic sensor. Whenever distance is less than 100cmssend “alert”to the ibm cloud and display in device recent events.

Solution:

Program:

```
#include <WiFi.h> #include
<WiFiClient.h> #include
<PubSubClient.h>const int
trigPin = 5; const int echoPin =
18;
//define sound speed in cm/uS
#define SOUND_SPEED 0.034
#define CM_TO_INCH 0.393701
long duration;
float distanceCm; float
distanceInch;

void callback(char* subscribetopic, byte* payload, unsigned intpayloadLength);
//----credentials of IBMAccounts----

#define ORG "b31tni"//IBM ORGANITION ID
#define DEVICE_TYPE "Assignment4"//Device type mentioned in ibm watson IOTPlatform
#define DEVICE_ID "assignment"//Device ID mentioned in ibm watson IOTPlatform#define TOKEN
"6r?TKCluy+okJ?9B+7" //Token
String data3;

//----- Customise the above values -----
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Namechar
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 commandtype 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 wificlientPubSubClient
client(server, 1883, callback, wifiClient);
```

```
void setup() {
  Serial.begin(115200); // Starts the serial communication pinMode(trigPin,
  OUTPUT); // Sets the trigPin as an Output pinMode(echoPin, INPUT); //
  Sets the echoPin as an Input Serial.println();
  wificonnect();
  mqttconnect();
}
```

```
void loop() {
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Reads the echoPin, returns the sound wave travel time in microseconds duration =
  pulseIn(echoPin, HIGH);

  // Calculate the distance
  distanceCm = duration * SOUND_SPEED/2;

  // Convert to inches
  distanceInch = distanceCm * CM_TO_INCH;

  // Prints the distance in the Serial Monitor
  Serial.print("Distance (cm): ");
  Serial.println(distanceCm); Serial.print("Distance (inch):
  "); Serial.println(distanceInch);

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

```
void PublishData(float Cm) {
```

```

mqttconnect();//function call for connecting to ibm
/*
    creating the String in in form JSon to update the data to ibm cloud
*/
String payload = "{\"Distance (cm)\":";payload +=
Cm;
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 cloudthen it will print publish
    ok in Serial monitor or else it will print publishfailed
} 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];
  }
}

```

Wokwi Simulation:

The screenshot displays the Wokwi simulation interface. On the left, the sketch code is shown, which includes the necessary libraries and defines an MQTT client for an IBM Watson IoT Platform. The code sets up an ESP32 microcontroller with an ultrasonic sensor (trigPin = 5, echoPin = 18) and configures it to publish distance measurements to an MQTT topic. The simulation on the right shows the physical components connected, with the sensor outputting a series of distance readings in centimeters.

```

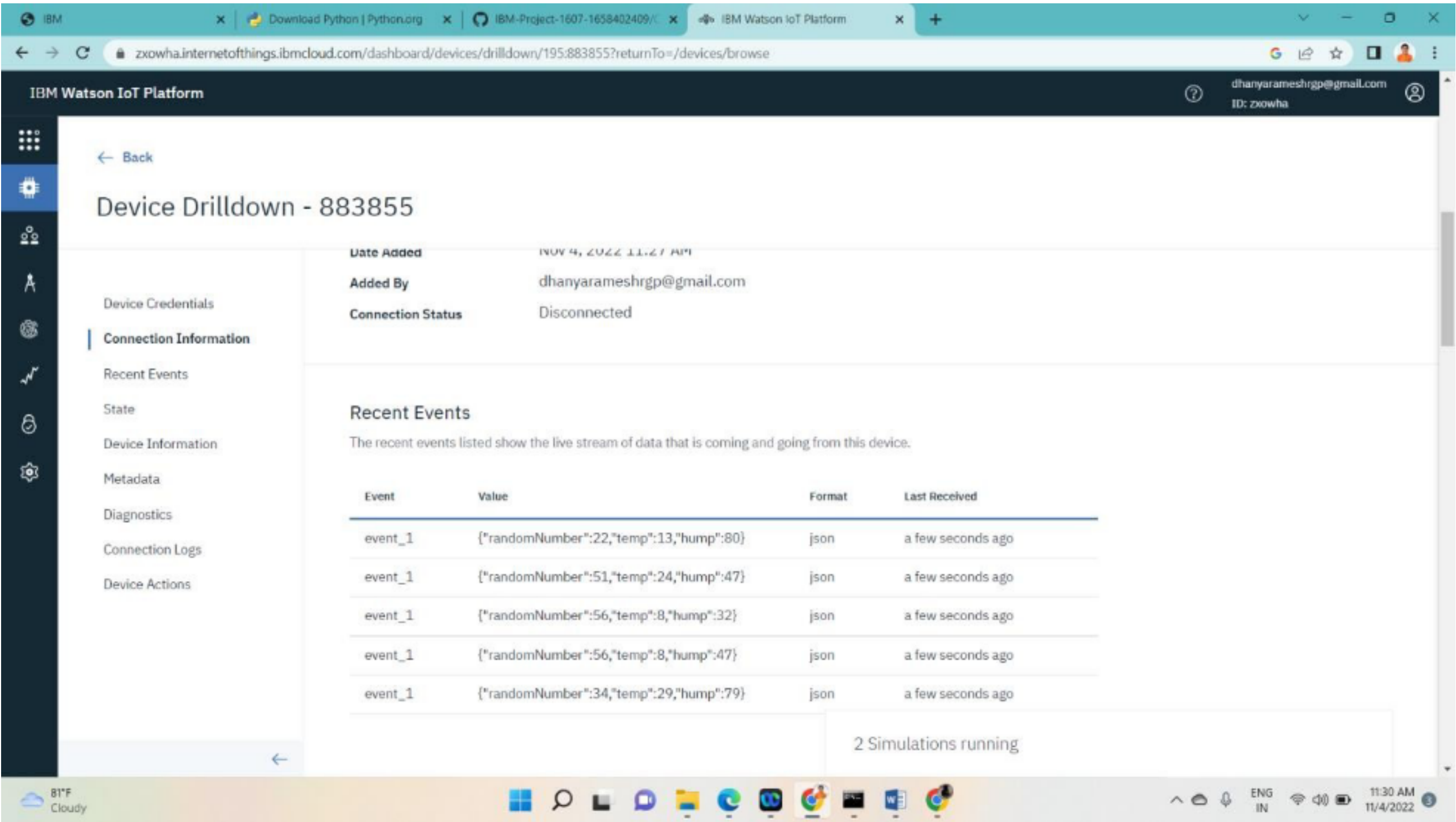
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* subscribetopic, byte* payload, unsigned int
4 payloadLength);
5 //-----credentials of IBM Accounts-----
6 #define ORG "9lxobn"//IBM ORGANITION ID
7 #define DEVICE_TYPE "ESP32PROJECT"//Device type mentioned in ibm watson IOT Platform
8 #define DEVICE_ID "ESP32"//Device ID mentioned in ibm watson IOT Platform
9 #define TOKEN "ESP32PROJECT" //Token
10 String data3;
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/Data/fmt/json";
13 char subscribetopic[] = "iot-2/cmd/test/fmt/String";
14 char authMethod[] = "use-token-auth";
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
17 WiFiClient wificlient;
18 PubSubClient client(server, 1883, callback ,wificlient);
19 const int trigPin = 5;
20 const int echoPin = 18;
21 #define SOUND_SPEED 0.034
22 long duration;
23 float distance;
24 void setup() {
25   Serial.begin(115200);
26   pinMode(trigPin, OUTPUT);
27   pinMode(echoPin, INPUT);
28   wificlient.connect();
29   mqttconnect();
30 }
31 void loop()
32 {
33   digitalWrite(trigPin, LOW);
34   delayMicroseconds(2);
35   digitalWrite(trigPin, HIGH);

```

Simulation controls: 00:22.296 99%

Distance (cm): 399.96
 Distance (cm): 399.94
 Distance (cm): 399.94
 Distance (cm): 399.94
 Distance (cm): 399.92
 Distance (cm): 399.94
 Distance (cm): 399.94

IoT Watson Platform:



<https://wokwi.com/projects/347372760316510804>

