

ASSIGNMENT-4

DISTANCE DETECTION USING ULTRASONIC SENSOR

Date	31 October 2022
Team Member	01
Name	M.ABINEETHA
Student Reg.no.	710719106004
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.

CODE :

```
1 // include init.h/library for wifi
2 // include esp8266libnet.h/library for MQTT
3
4
5 void callback(char* topic, byte* payload, unsigned int payloadLength);
6
7 // ----- credentials of IoT Account -----
8
9 #define MQTT_USERNAME "root" // MQTT USERNAME ID
10 #define DEVICE_TYPE "ULTRASONIC" // Device type mentioned in the custom IoT Platform
11 #define DEVICE_ID "DISTANCE DETECT" // Device ID mentioned in the custom IoT Platform
12 #define TOKEN "a66c39424e9a6d8a" // Token
13 String data;
14 float dist;
15
16
17 // ----- Initialize the above values -----
18 char server[] = MQTT_MESSAGE_INTERFACE ThingsSpeak.com // Server Name
19 char publishTopic[] = "iot-dist/range" // topic name and type of event payload and format in which data to be sent
20 char subscribeTopic[] = "iot-dist/test/range" // and subscribe command type and format in test or remote action
21 char authMethod[] = "auth-token-auth" // authentication method
22 char token[] = TOKEN;
23 char clientID[] = "id:" MQTT_USERNAME "/" DEVICE_TYPE "/" DEVICE_ID // client id
24
25
26 //
27 #include <WiFiClient.h> // creating the instance for wifi client
28 #include <PubSubClient.h> // creating the instance for mqtt client
29
30 int led = A1;
31 int trig = A2;
32 int echo = A3;
33 void setup()
34 {
35   Serial.begin(115200);
```

```
36 pinMode(trig,OUTPUT);
37 pinMode(echo,INPUT);
38 pinMode(LED, OUTPUT);
39 delay(10);
40 wificonnect();
41 mqttconnect();
42 }
43 void loop()// Recursive Function
44 {
45
46     digitalWrite(trig,LOW);
47     digitalWrite(trig,HIGH);
48     delayMicroseconds(10);
49     digitalWrite(trig,LOW);
50     float dur = pulseIn(echo,HIGH);
51     float dist = (dur * 0.0343)/2;
52     Serial.print ("Distance in cm");
53     Serial.println(dist);
54
55
56     PublishData(dist);
57     delay(1000);
58     if (!client.loop()) {
59         mqttconnect();
60     }
61 }
62
63
64
65 /*.....retrieving to cloud.....*/
66
67 void PublishData(float dist) {
68     mqttconnect();//function call for connecting to ibm
69     /*
70     | creating the String in in form JSON to update the data to ibm cloud
```

```

70     // creating the string in in form json to update the data to the cloud
71     */
72     String object;
73     if (dist < 100)
74     {
75         digitalWrite(LED, HIGH);
76         Serial.println("object is near");
77         object = "Near";
78     }
79     else
80     {
81         digitalWrite(LED, LOW);
82         Serial.println("no object found");
83         object = "No";
84     }
85
86     String payload = "{\"distance\":";
87     payload += dist;
88     payload += ",";
89     payload += "\"object\":";
90     payload += object;
91     payload += "\"}";
92
93     Serial.print("Sending payload: ");
94     Serial.println(payload);
95
96
97
98

```

```

emp32@blink:~$ cat chapter_03.ino
100
101 if (client.publish(topic, (char*) payload.c_str())) {
102     Serial.println("publish ok"); // if it successfully send data to the cloud then it will print publish ok in serial monitor or else it will print publish failed
103 } else {
104     Serial.println("publish failed");
105 }
106
107
108 void mqttConnect() {
109     if (!client.connected()) {
110         Serial.print("reconnecting client to ");
111         Serial.println(server);
112         while (!client.connect(topic, username, password)) {
113             Serial.print(".");
114             delay(500);
115         }
116
117         digitalWrite(LED, HIGH);
118         Serial.println();
119     }
120 }
121
122 void mqttDisconnect() //function definition for mqttDisconnect
123 {
124     Serial.println();
125     Serial.print("disconnecting to ");
126
127     WiFi.disconnect(true); //passing the wifi credentials to establish the connection
128     while (WiFi.status() != WL_CONNECTED) {
129         delay(500);
130         Serial.print(".");
131     }
132
133     Serial.println();
134     Serial.println("WiFi disconnected");
135     Serial.println("IP address: ");
136     Serial.println(WiFi.localIP());
137 }

```

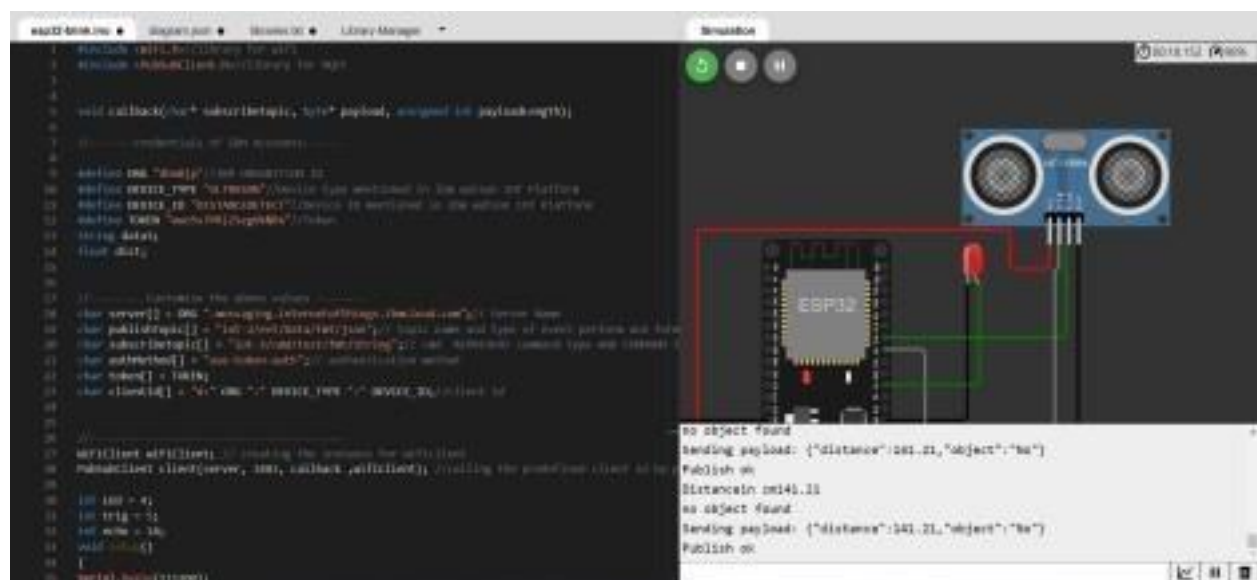
```
123
124   WiFi.begin("Wokwi-GUEST", ""); //passing the wifi credentials to establish the connection
125   while (WiFi.status() != WL_CONNECTED) {
126       delay(500);
127       Serial.print(".");
128   }
129   Serial.println("");
130   Serial.println("WiFi connected");
131   Serial.println("IP address: ");
132   Serial.println(WiFi.localIP());
133 }
134
135 void initManagedDevice() {
136     if (client.subscribe(subscribetopic)) {
137         Serial.println((subscribetopic));
138         Serial.println("subscribe to cmd OK");
139     } else {
140         Serial.println("subscribe to cmd FAILED");
141     }
142 }
143
144 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
145 {
146
147     Serial.print("callback invoked for topic: ");
148     Serial.println(subscribetopic);
149     for (int i = 0; i < payloadLength; i++) {
150         //Serial.print((char)payload[i]);
151         data3 += (char)payload[i];
152     }
153
154     // Serial.println("data: "+ data3);
155     // if(data3=="Near")
156     // {
157     // Serial.println(data3);
158     // }
159 }
```

```

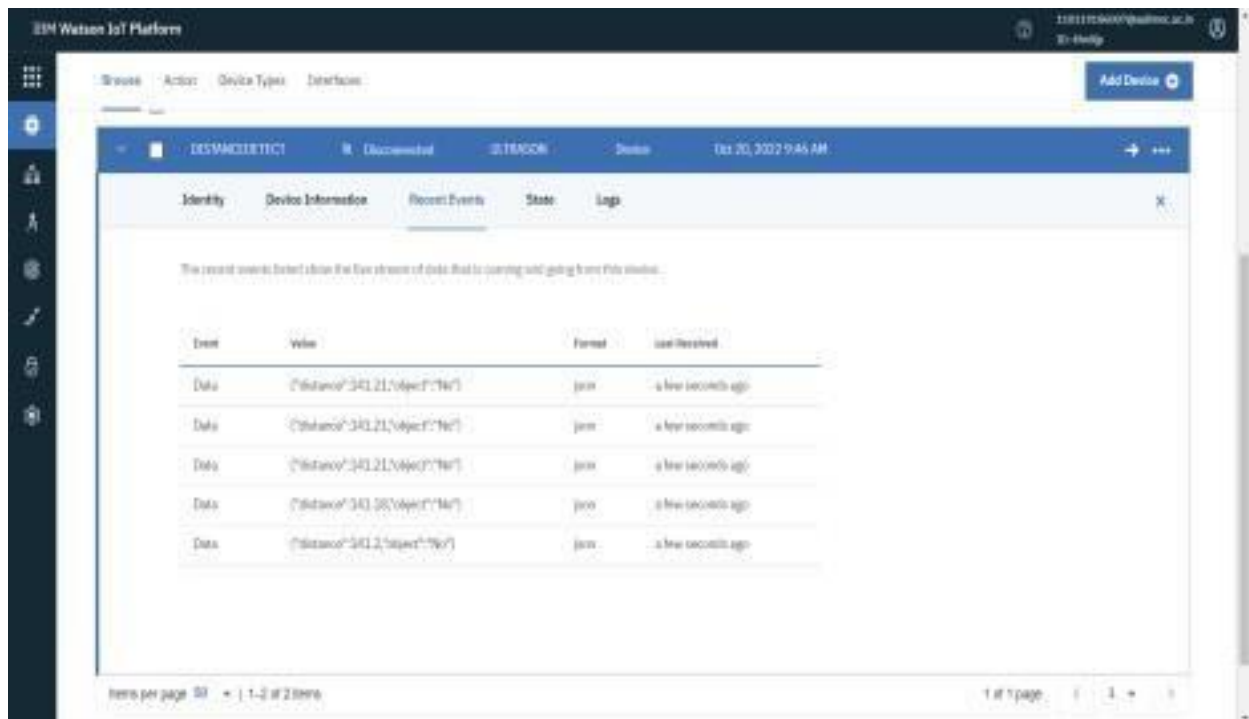
esp32-blink.ino • diagram.json • libraries.txt • Library Manager
142 }
143
144 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
145 {
146
147     Serial.print("callback invoked for topic: ");
148     Serial.println(subscribetopic);
149     for (int i = 0; i < payloadLength; i++) {
150         //Serial.print((char)payload[i]);
151         data3 += (char)payload[i];
152     }
153
154     // Serial.println("data: "+ data3);
155     // if(data3=="Near")
156     // {
157     // Serial.println(data3);
158     // digitalWrite(LED,HIGH);
159     // }
160     // }
161
162     // else
163     // {
164     // Serial.println(data3);
165     // digitalWrite(LED,LOW);
166     // }
167     data3="";
168 }
169
170
171

```

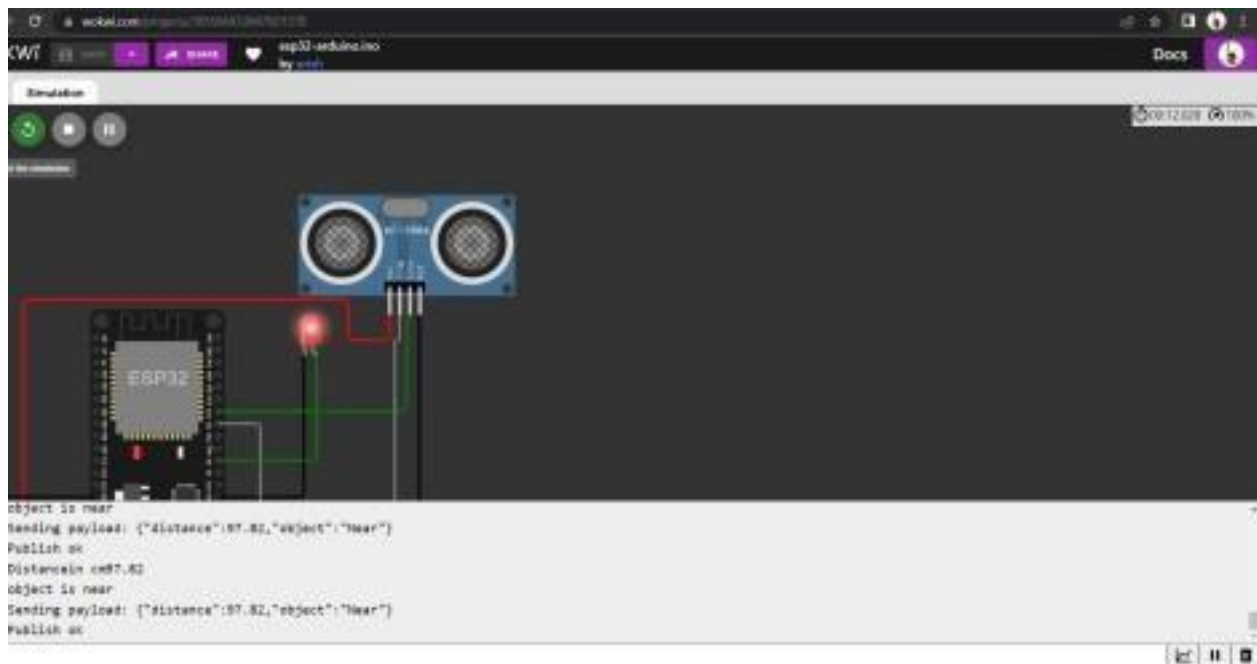
OUTPUT:



Data send to the IBM cloud device when the object is far



When object is near to the ultrasonic sensor



Data sent to the IBM Cloud Device when the object is near

[illegible]