ASSIGNMENT-4
DISTANCE DETECTION USING ULTRASONIC SENSOR

Date	1st November 2022
Team ID	PNT2022TMID13493
Name	Arun .R
Mark	2Marks

Question1 :

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 <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 "4hn0jp"//IBM ORGANITION ID
10   #define DEVICE_TYPE "ULTRASON"//Device type mentioned in ibm watson IOT Platform
11   #define DEVICE_ID "DISTANCEDETECT"//Device ID mentioned in ibm watson IOT Platform
12   #define TOKEN "wuo5s7PR)ZSegVk&Rx"//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 and format in which data to be send
20   char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd  REPRESENT command type AND COMMAND IS TEST OF FORMAT STRING
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 id by passing parameter like server id,portand wificredential
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    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 ("Distancein 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 ibm 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 += "," "\"object\":\"";
89  payload += object;
90  payload += "\"}";
91
92
93  Serial.print("Sending payload: ");
94  Serial.println(payload);
95
96
97
98
```

esp32-blink.ino ●   diagram.json ●   libraries.txt ●   Library Manager ▾

```
98
99   if (client.publish(publishTopic, (char*) payload.c_str())) {
100    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
101  } else {
102    Serial.println("Publish failed");
103  }
104
105  }
106  void mqttconnect() {
107    if (!client.connected()) {
108      Serial.print("Reconnecting client to ");
109      Serial.println(server);
110      while (!!!client.connect(clientId, authMethod, token)) {
111        Serial.print(".");
112        delay(500);
113      }
114
115      initManagedDevice();
116      Serial.println();
117    }
118  }
119  void wificonnect() //function defination for wificonnect
120  {
121    Serial.println();
122    Serial.print("Connecting to ");
123
124    WiFi.begin("Wokwi-GUEST", "", 6);//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());
```

```
123
124     WiFi.begin("Wokwi-GUEST", "", 6);//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);
```

```
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    //    }
168    data3="";
169
170
171    }
```
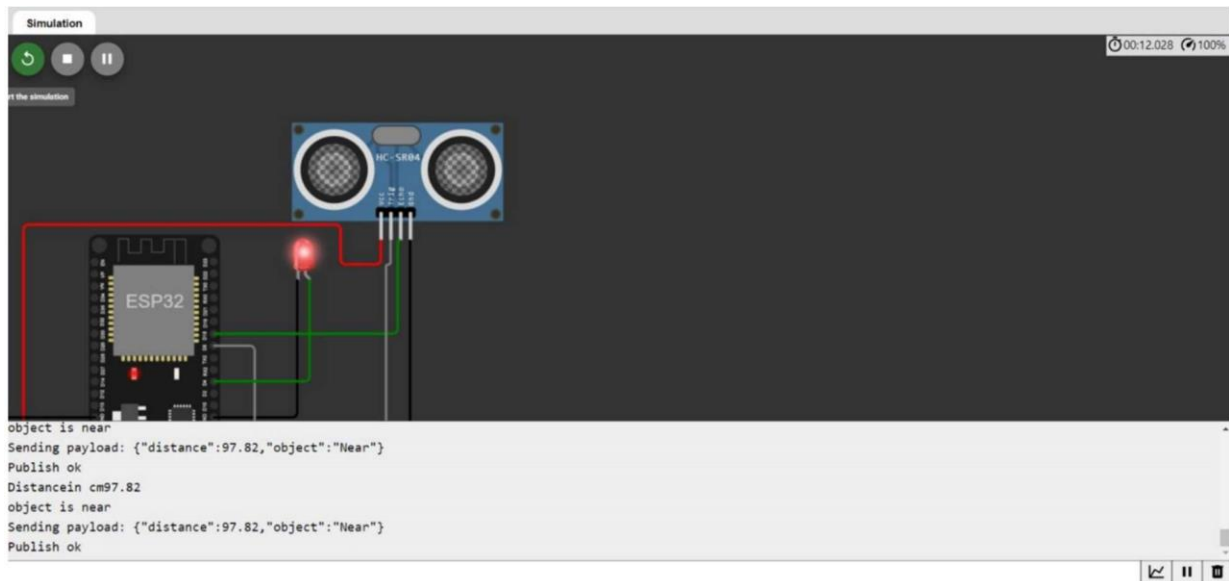
OUTPUT:
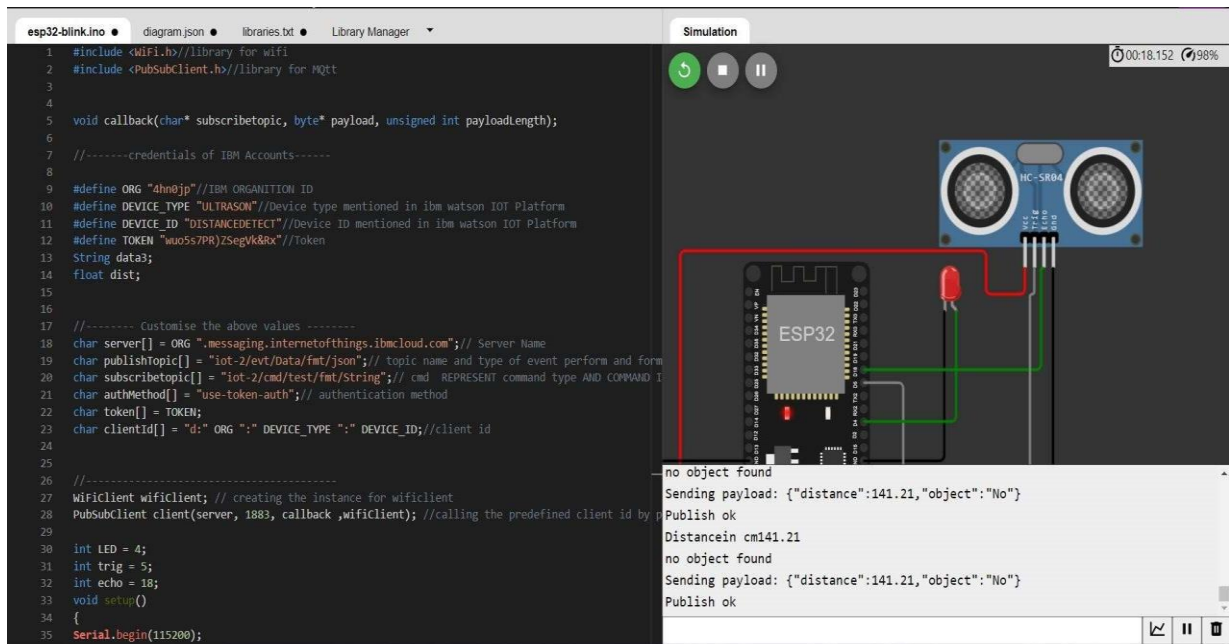
Data send to the IBM cloud device when the object is far
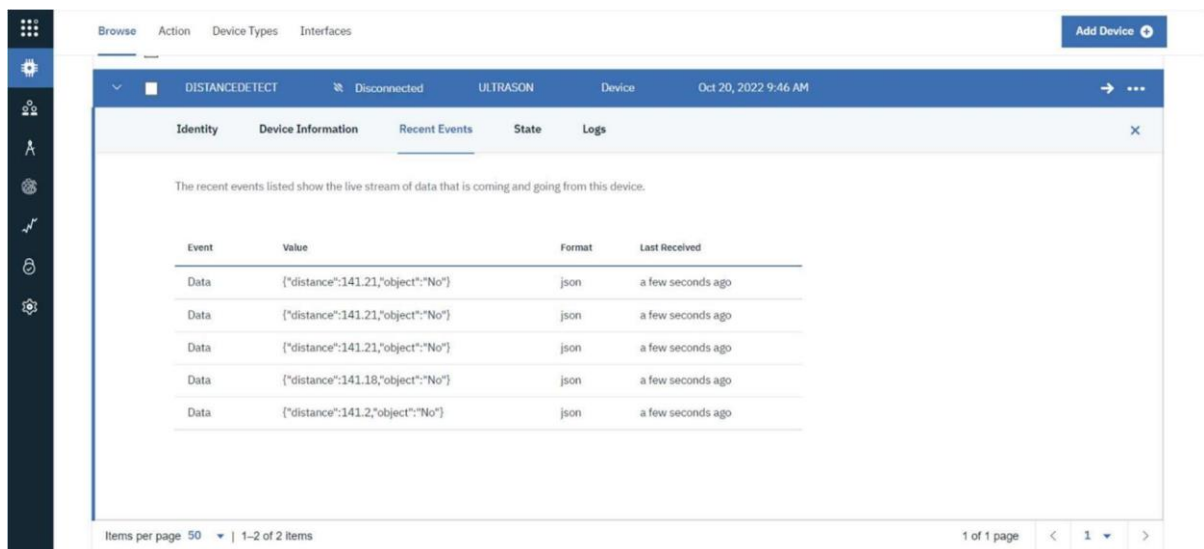
When object is near to the ultrasonic sensor

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