

Assignment -4

Assignment Date	1 October 2022
Student Name	SINDHU P
Student Roll Number	622119104101
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.

The image shows the Wokwi IDE interface with a C++ sketch for an ESP32 microcontroller connected to an HC-SR04 ultrasonic sensor. The sketch includes the following code:

```
1 #include<WiFi.h> //library for wifi
2 #include<PubSubClient.h> //library for MQTT
3 void callback(char* topic, byte* payload, unsigned int payloadlength);
4 //-----credentials of IBM Account-----
5 #define ORG "izyy6o" // IBM ORGANIZATION ID
6 #define DEVICE_TYPE "iotdeviceproject" //DEVICE TYPE MENTIONED IN IOT WATSON PLATFORM
7 #define DEVICE_ID "229714" //DEVICE ID MENTIONED IN IOT WATSON PLATFORM
8 #define TOKEN "24681012" //Token
9 String data3;
10 float dist;
11 //-----customize the above value-----
12 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; //server name
13 char publishTopic[] = "ultrasonic/evt/Data/fmt/json"; //topic name and type of event per
14 | and format in which data to be send"/
15 char subscribeTopic[] = "ultrasonic/cmd/test/fmt/String"; //cmd REPRESENT Command tupe a
16 COMMAND IS TEST OF FORMAT STRING"/
17 char authMethod[] = "use-token-auth"; //authentication method
18 char token[] = TOKEN;
19 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //CLIENT ID
20 //-----
21 WiFiClient wifiClient; // creating an instance for wifiClient
22 PubSubClient client(server, 1883, callback, wifiClient); //calling the predefined cl
23 by passing parameter like server id, port and wifi credential"/
24 int LED = 4;
25 int trig = 5;
26 int echo = 18;
27 void setup()
28 {
29   Serial.begin(115200);
30   pinMode(trig, OUTPUT);
31   pinMode(echo, INPUT);
32   pinMode(LED, OUTPUT);
```

The simulation window shows the physical connection of the HC-SR04 sensor to the ESP32. The sensor's VCC pin is connected to the ESP32's 5V pin, and its GND pin is connected to the ESP32's GND pin. The trig pin is connected to pin 5, and the echo pin is connected to pin 18. A red LED is also connected to the ESP32's 5V and GND pins.

WOKWI

SAVE

SHARE

Docs

sketch.ino

diagram.json

libraries.txt

Library Manager

Simulation

```

32 pinMode(LED,OUTPUT);
33 delay(10);
34 wifiConnect();
35 mqttconnect();
36 }
37 void loop()//recursive function
38 {
39   digitalWrite(trig,LOW);
40   digitalWrite(trig,HIGH);
41   delayMicroseconds(10);
42   digitalWrite(trig,LOW);
43   float dur=pulseIn(echo,HIGH);
44   float dist=(dur * 0.0343)/2;
45   Serial.print("distance in cm");
46   Serial.println(dist);
47   PublishData(dist);
48   delay(1000);
49   if (!client.loop()){
50     mqttconnect();
51   }
52 }
53 /*.....retriving to cloud.....*/
54 void PublishData(float dist){
55   mqttconnect();//function call for connecting to ibm
56   /*creating the string in form of JSON to update the data to ibm cloud*/
57   String object;
58   if(dist<100)
59   {
60     digitalWrite(LED,HIGH);
61     Serial.println("no object is near");
62     object="Near";
63   }

```

WOKWI

SAVE

SHARE

sketch.ino

diagram.json

libraries.txt

Library Manager

Simulation

```

63 }
64 else
65 {
66   digitalWrite(LED,LOW);
67   Serial.println("no object found");
68   object="No";
69 }
70 String payload="{\"distance\":";
71 payload +=dist;
72 payload +=",\" \"object\":\":";
73 payload += object;
74 payload += "\":";
75
76 Serial.print("Sending payload: ");
77 Serial.println(payload);
78 if(client.publish(publishtopic, (char*) payload.c_str())){
79   Serial.println("Publish ok");/* if its sucessfully upload data on the cloud then
80   publish ok in serial monitor or else it will print publish failed*/
81 } else{
82   Serial.println("Publish failed");
83 }
84 }
85 void mqttconnect(){
86   if(!client.connected()){
87     Serial.print("Reconnecting client to ");
88     Serial.println(server);
89     while(!client.connect(clientid,authMethod, token)){
90       Serial.print(".");
91       delay(500);
92     }
93     initManagedDevice();
94     Serial.println();

```

WOKWI
SAVE
SHARE
Docs

sketch.ino

diagram.json
libraries.txt
Library Manager

```

93   initManagedDevice();
94   Serial.println();
95 }
96 }
97 void wificonnect()//function definition for wificonnect
98 {
99   Serial.println();
100   Serial.print("connecting to ");
101   Wifi.begin("Mokwi.GUEST", "",6);//PASSING THE WIFI CREDENTIALS TO ESTABLISH CONNE
102   while (Wifi.status() !=WL_CONNECTED){
103     delay(500);
104     Serial.print(".");
105   }
106   Serial.println("");
107   Serial.println("Wifi connected");
108   Serial.println("IP address");
109   Serial.println(Wifi.localIP());
110 }
111 void initManagedDevice(){
112   if(client.subscribe(subscribetopic)){
113     Serial.println((subscribetopic));
114     Serial.println("subscribe to cmd OK");
115   }else{
116     Serial.println("subscribe to cmd failed");
117   }
118 }
119 void callback(char* subscribetopic,byte*payload,unsigned int payloadlength)
120 {
121   Serial.print("callback invoked for topic: ");
122   Serial.println(subscribetopic);
123   for(int i=0; i< payloadlength; i++){
124     //Serial.print((char)payload[i]);
125     data3 +=(char)payload[i];

```

Simulation

WOKWI
SAVE
SHARE
Docs

sketch.ino

diagram.json
libraries.txt
Library Manager

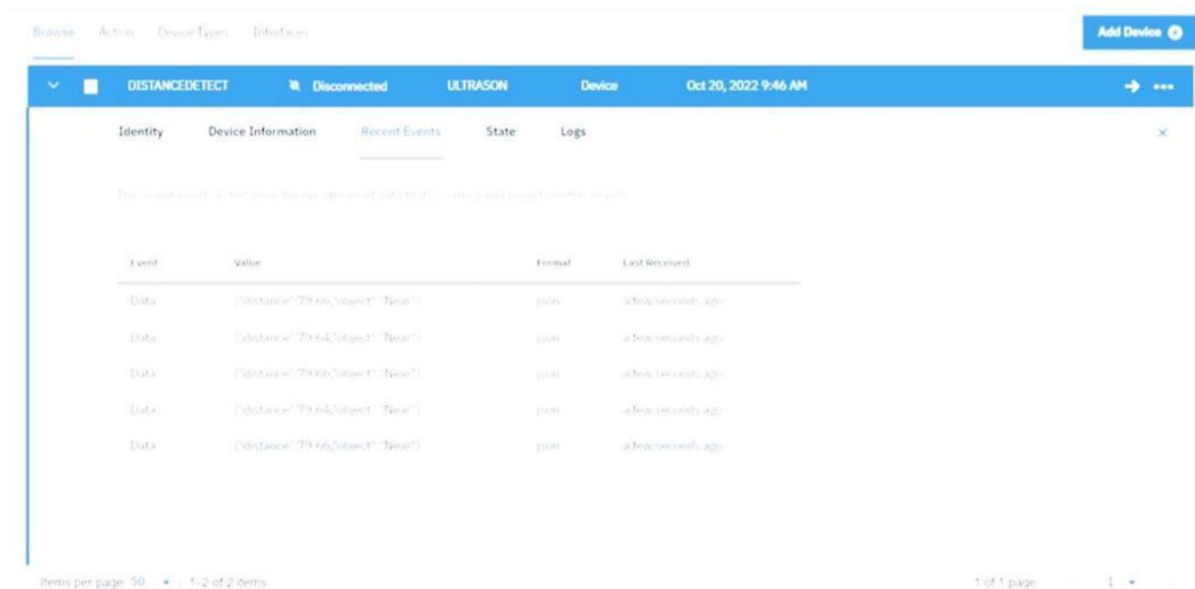
```

111 void initManagedDevice(){
112   if(client.subscribe(subscribetopic)){
113     Serial.println((subscribetopic));
114     Serial.println("subscribe to cmd OK");
115   }else{
116     Serial.println("subscribe to cmd failed");
117   }
118 }
119 void callback(char* subscribetopic,byte*payload,unsigned int payloadlength)
120 {
121   Serial.print("callback invoked for topic: ");
122   Serial.println(subscribetopic);
123   for(int i=0; i< payloadlength; i++){
124     //Serial.print((char)payload[i]);
125     data3 +=(char)payload[i];
126   }
127   //Serial.println("dta: "+ data3);
128   //if(data3=="Near")
129   //{
130   //Serial.println(data3);
131   //digitalWrite(LED,HIGH);
132   //}
133   //else
134   //{
135   //Serial.println(data3);
136   //digitalWrite(LED,LOW);
137   //}
138   data3="";
139 }

```

Simulation

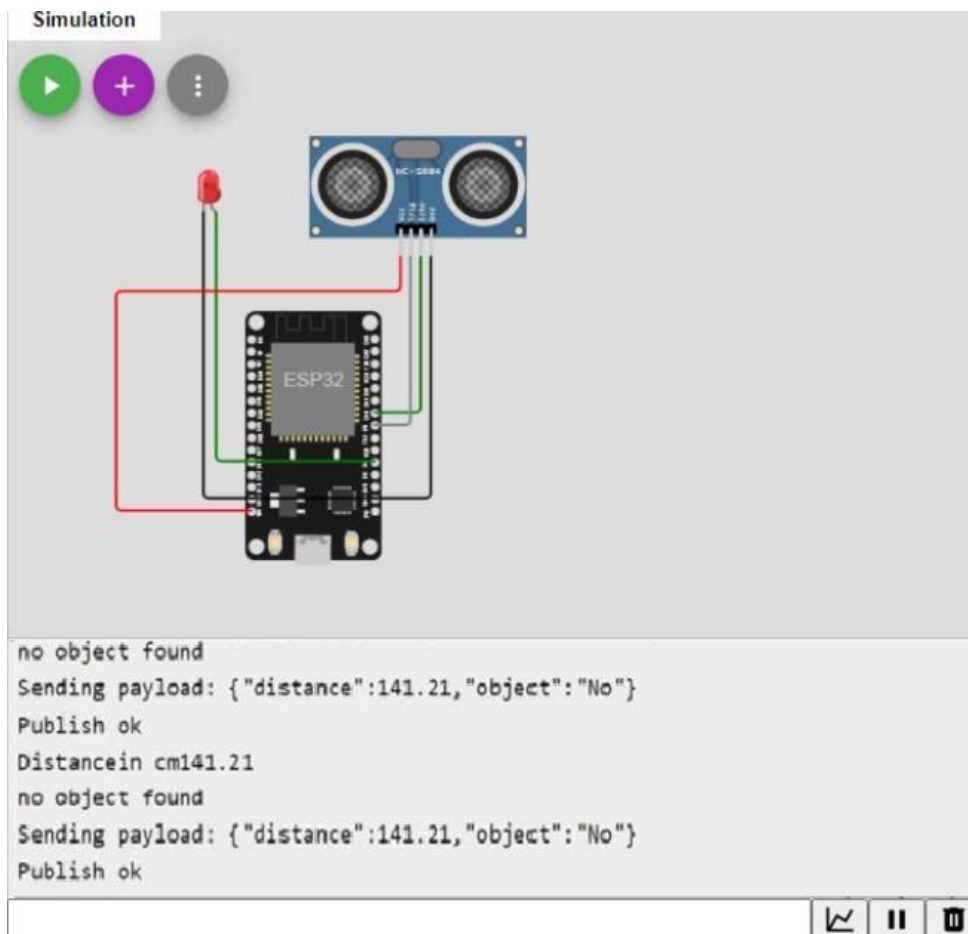
DATA SENT TO IBM CLOUD ON NO OBJECT DETECTED



The screenshot shows the IBM Cloud IoT Platform console. At the top, there are tabs for 'Browse', 'Actions', 'Device Types', and 'Interfaces'. A blue header bar contains the device name 'DISTANCEDETECT', its status 'Disconnected', the type 'ULTRASON', and the location 'Device'. The date and time 'Oct 20, 2022 9:46 AM' are also displayed. Below the header, there are tabs for 'Identity', 'Device Information', 'Recent Events', 'State', and 'Logs'. The 'Recent Events' tab is selected, showing a table of events. The table has four columns: 'Event', 'Value', 'Format', and 'Last Received'. There are five rows of data, all with the same value: '{"distance":79.66,"object":"None"}'. The 'Format' column shows 'json' and the 'Last Received' column shows 'a few seconds ago'. At the bottom, there is a pagination bar showing 'Items per page: 50' and '1 of 1 page'.

Event	Value	Format	Last Received
Data	{"distance":79.66,"object":"None"}	json	a few seconds ago
Data	{"distance":79.66,"object":"None"}	json	a few seconds ago
Data	{"distance":79.66,"object":"None"}	json	a few seconds ago
Data	{"distance":79.66,"object":"None"}	json	a few seconds ago
Data	{"distance":79.66,"object":"None"}	json	a few seconds ago

WHEN OBJECT DETECTED BY ULTRASONIC DETECTOR SENSOR



The simulation shows an ESP32 microcontroller connected to an HC-SR04 ultrasonic sensor. The sensor is connected to the ESP32 via a red wire (VCC), a green wire (GND), and a blue wire (Trig). The sensor is also connected to a red LED. The simulation output shows the following sequence of events:

```
no object found
Sending payload: {"distance":141.21,"object":"No"}
Publish ok
Distancein cm141.21
no object found
Sending payload: {"distance":141.21,"object":"No"}
Publish ok
```

