

Assignment - 4

Signs with Smart Connectivity for Better Road Safety

Assignment Date	20 Oct 2022
Student Name	Gayathri B
Student Roll num	910619106015
Maximum Marks	2 Marks

Question :

Write code and connections in wokwi for the ultrasonic sensor. Whenever the distance is less than 100 cms send an "alert" to the IBM cloud and display in the device recent events. Upload document with wokwi share link and images of IBM cloud.

Code in Wokwi:

```
#include <WiFi.h>
#include <PubSubClient.h>

void callback(char* subscribtopic, byte* payload, unsigned int
payloadLength);

//-----credentials of IBM Accounts-----
#define ORG "confidential"//IBM ORGANITION ID
#define DEVICE_TYPE "gaya"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "0605"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "confidential" //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char subscribtopic[] = "iot-2/cmd/test/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
WiFiClient wifiClient;
PubSubClient client(server, 1883, callback ,wifiClient);
const int trigPin = 5;
const int echoPin = 18;
#define SOUND_SPEED 0.034
long duration;
float distance;
void setup() {
```

```

Serial.begin(115200);
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
wificonnect();
mqttconnect();
}
void loop()
{
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distance = duration * SOUND_SPEED/2;
Serial.print("Distance (cm): ");
Serial.println(distance);
if(distance<100)
{
Serial.println("ALERT!!");
delay(1000);
PublishData(distance);
delay(1000);
if (!client.loop()) {
mqttconnect();
}
}
delay(1000);
}
void PublishData(float dist) {
mqttconnect();
String payload = "{\"Distance\": ";
payload += dist;
payload += ", \"ALERT!!\": \"\" \"Distance less than 100cms\"";
payload += "}";
Serial.print("Sending payload: ");
Serial.println(payload);

if (client.publish(publishTopic, (char*) payload.c_str())) {
Serial.println("Publish ok");
} 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()
{
  Serial.println();
  Serial.print("Connecting to ");
  WiFi.begin("Wokwi-GUEST", "", 6);
  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);
  data3="";
}

```

Wokwi sketch and Simulation:

The screenshot displays the Wokwi IDE interface. On the left, the 'sketch.ino' file is open, showing a C++ sketch for an ESP32. The sketch includes libraries for WiFi, PubSubClient, and MQTT. It defines constants for the IBM Watson IOT Platform, including the organization ID, device type 'gaya', device ID '0602', and a token. The setup function initializes the serial port, pins, and MQTT client. The loop function reads the distance from an ultrasonic sensor (HC-SR04) and publishes the data to the MQTT broker. If the distance is less than 100 cm, it also publishes an alert message.

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* topic, byte* payload, unsigned int
4 payloadLength);
5 //-----credentials of IBM Accounts-----
6 #define ORG " " //IBM ORGANITION ID
7 #define DEVICE_TYPE "gaya" //Device type mentioned in ibm watson IOT Platform
8 #define DEVICE_ID "0602" //Device ID mentioned in ibm watson IOT Platform
9 #define TOKEN " " //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);
```

On the right, the 'Simulation' window shows a visual representation of the ESP32 and the HC-SR04 sensor. Below the simulation, the console output shows the following sequence of events:

```
Publish ok
Distance (cm): 1.99
ALERT!!
Sending payload: {"Distance":1.99,"ALERT!!":"Distance less than 100cms"}
Publish ok
Distance (cm): 1.99
ALERT!!
```

IBM Watson IOT Platform :

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes the 'Browse' tab, which is currently selected. The main content area displays a table of devices. The first device, with ID '0602', is shown as 'Connected' and has a 'gaya' device type. Below the table, the 'Recent Events' tab is selected, showing a list of events received from the device. Each event contains a JSON payload with distance and alert information.

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
0602	Connected	gaya	Device	1 Nov 2022 12:31	

Event	Value	Format	Last Received
Data	{"Distance":1.99,"ALERT!!":"Distance less than 100cms"}	json	a few seconds ago
Data	{"Distance":1.99,"ALERT!!":"Distance less than 100cms"}	json	a few seconds ago
Data	{"Distance":1.99,"ALERT!!":"Distance less than 100cms"}	json	a few seconds ago
Data	{"Distance":1.99,"ALERT!!":"Distance less than 100cms"}	json	a few seconds ago
Data	{"Distance":1.99,"ALERT!!":"Distance less than 100cms"}	json	a few seconds ago

0 Simulations running

*Whenever the distance is less than 100 cms send an "alert" to the IBM cloud and display in the device **Recent Events**.*