

ASSIGNMENT – 4

Name : Pradeepika S R

Date : 08-11-2022

Team ID : PNT2022TMID42240

Register Number : 710019106034

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:

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribtopic, byte* payload, unsigned int
payloadLength);
//-----credentials of IBM Accounts-----
#define ORG "ss3zo8"//IBM ORGANITION ID
#define DEVICE_TYPE "pradeepikadevice"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "2207"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "j@EvVhgnsUYLDBZkIm" //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 LINK:

<https://wokwi.com/projects/347574893885260372>

WOKWI OUTPUT:

The screenshot displays the Wokwi IoT simulator interface. On the left, a code editor shows the following C++ code for an ESP32:

```
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 "ss3zo8"//IBM ORGANITION ID
7 #define DEVICE_TYPE "pradeepikadevice"//Device type mentioned in ibm wat
8 #define DEVICE_ID "2207"//Device ID mentioned in ibm watson IOT Platform
9 #define TOKEN "j@EvVhgnSUyLDBZkIm" //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   wifiConnect();
29   mqttConnect();
30 }
31 void loop()
32 {
```

On the right, the simulation window shows a visual representation of the ESP32 and the HC-SR04 ultrasonic sensor connected by wires. Below the visual, a console output displays the following distance readings:

```
Distance (cm): 399.94
Distance (cm): 399.98
Distance (cm): 399.94
Distance (cm): 399.94
Distance (cm): 399.94
Distance (cm): 399.94
Distance (cm): 399.96
Distance (cm): 399.94
```

The interface also includes a top navigation bar with 'SAVE' and 'SHARE' buttons, and a bottom status bar showing the time as 5:36 PM on 11/8/2022.

Browse
Action
Device Types
Interfaces

Add Device

☐
Device ID
Status
Device Type
Class ID
Date Added

2001
Connected
salman_device
Device
7 Nov 2022 14:58

Identity
Device Information
Recent Events
State
Logs

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
Data	{"Distance":98.94,"ALERT!!":"Distance less than 1...	json	a few seconds ago
Data	{"Distance":98.94,"ALERT!!":"Distance less than 1...	json	a few seconds ago
Data	{"Distance":81.97,"ALERT!!":"Distance less than 1...	json	a minute ago

1 Simulation running