

ASSIGNMENT 4

Team Id : PNT2022TMID16239

Name : Vasugi.G

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

Sketch.ino

```
#include <WiFi.h>

#include <PubSubClient.h>

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

//-----credentials of IBM Accounts-----

#define ORG "zqkiyb"//IBM ORGANITION ID

#define DEVICE_TYPE "ESP"//Device type mentioned in ibm watson IOT Platform

#define DEVICE_ID "Assignment4"//Device ID mentioned in ibm watson IOT Platform

#define TOKEN "1 87654321" //Token

String data3;

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";

char publishTopic[] = "iot-2/evt/Data/fmt/json";

char subscribetopic[] = "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="";  
}
```

Diagram.json

```
{  
  "version": 1,  
  "author": " 6169_Vasugi G",  
  "editor": "wokwi",  
  "parts": [  
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 17.33, "left": -98.67, "attrs": {} },  
    { "type": "wokwi-hc-sr04", "id": "ultrasonic1", "top": -11.41, "left": 50.44, "attrs": {} }  
  ],  
  "connections": [  
    [ "esp:TX0", "$serialMonitor:RX", "", [] ],  
    [ "esp:RX0", "$serialMonitor:TX", "", [] ],  
    [ "ultrasonic1:VCC", "esp:VIN", "red", [ "v0" ] ],  
    [ "ultrasonic1:GND", "esp:GND.1", "black", [ "v0" ] ],  
    [ "ultrasonic1:TRIG", "esp:D5", "green", [ "v0" ] ],  
    [ "ultrasonic1:ECHO", "esp:D18", "green", [ "v0" ] ]  
  ]  
}
```

Wokwi link:

<https://wokwi.com/projects/347193895229063762>

Output :

WOKWI

sketch.ino

```

1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* subscribetopic, byte* payload, unsigned int
4 payloadLength);
5 //-----credentials of IBM Accounts-----
6 #define ORG "zqkiyb"//IBM ORGANITION ID
7 #define DEVICE_TYPE "ESP"//Device type mentioned in ibm watson IOT Platform
8 #define DEVICE_ID "Assignment4"//Device ID mentioned in ibm watson IOT Platform
9 #define TOKEN "87654321" //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();

```

Simulation

Distance (cm): 80.94
 ALERT!!
 Sending payload: {"Distance":80.94,"ALERT!!":"Distance less than 100cms"}
 Publish ok
 Distance (cm): 80.99
 ALERT!!

zqkiyb.internetofthings.ibmcloud.com/dashboard/devices/browse

IBM Watson IoT Platform

111719106169@smartinternz.com
 ID: zqkiyb

Browse Action Device Types Interfaces

Add Device

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":80.94,"ALERT!!":"Distance less than ...	json	a few seconds ago
Data	{"Distance":80.94,"ALERT!!":"Distance less than ...	json	a few seconds ago
Data	{"Distance":80.94,"ALERT!!":"Distance less than ...	json	a few seconds ago

Items per page 50 | 1-1 of 1 item

1 of 1 page