

Assignment Date	22 October2022
Student Name	Arun kumar S
Student Roll Number	2019504009
Maximum Marks	2 Marks

ASSIGNMENT-4

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.

Source Code:

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic,byte* payload,unsigned int payloadLength);
#define ORG "liyiqr"
#define DEVICE_TYPE "NodeMCU"
#define DEVICE_ID "1515"
#define TOKEN "123456789"
String data3;
char server[]= ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[]="iot-2/evt/distance/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);
#define ECHO_PIN 14
#define TRIG_PIN 12
#define led 27
void setup() {
// put your setup code here, to run once:
Serial.begin(115200);
pinMode(led, OUTPUT);
pinMode(TRIG_PIN, OUTPUT);
pinMode(ECHO_PIN, INPUT);
wificonnect();
mqttconnect();
}
float readDistanceCM() {
digitalWrite(TRIG_PIN, LOW);// Clear the trigger
delayMicroseconds(2);
```

```

digitalWrite(TRIG_PIN, HIGH); // Sets the trigger pin to HIGH state for 10 microseconds
delayMicroseconds(10);
digitalWrite(TRIG_PIN, LOW);
int duration=pulseIn(ECHO_PIN, HIGH);
//Serial.println(duration);
//duration = pulseIn(ECHO_PIN, HIGH);
return duration*0.017;
//Serial.println(duration);
}
void loop() {
float distance = readDistanceCM();
//Serial.println(distance);
bool isNearby = distance < 100;
digitalWrite(led, isNearby);
Serial.print("Measured distance: ");
Serial.println(distance);
if(distance<100){
PublishData2(distance);
}else{
PublishData1(distance);
}
//PublishData(distance);
delay(1000);
if(!client.loop()){
mqttconnect();
}
//delay(2000);
}
void PublishData1(float dist){
mqttconnect();
String payload= "{\"distance\":";
payload += dist;
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 PublishData2(float dist){
mqttconnect();
String payload= "{\"ALERT\":";
payload += dist;
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 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++){
data3 += (char)payload[i];}
Serial.println("data:"+ data3);
if(data3=="lighton"){
Serial.println(data3);
digitalWrite(led,HIGH);
}else{
Serial.println(data3);
digitalWrite(led,LOW);}
data3="";
}
}

```

WOKWI LINK:

<https://wokwi.com/projects/346469452322177619>

OUTPUT:

NORMAL CASE:

The screenshot displays the Wokwi web IDE interface. On the left, the 'sketch.ino' file is open, showing the following code:

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* topic, byte* payload, unsigned int payloadLength) {
4   #define ORG "liyiqr"
5   #define DEVICE_TYPE "NodeMCU"
6   #define DEVICE_ID "1515"
7   #define TOKEN "123456789"
8   String data;
9   char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
10  char publishTopic[] = "iot-2/evt/distance/fmt/json";
11  char subscribeTopic[] = "iot-2/cmd/test/fmt/String";
12  char authMethod[] = "use-token-auth";
13  char token[] = TOKEN;
14  char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
15  WiFiClient wifiClient;
16  PubSubClient client(server, 1883, callback, wifiClient);
17  #define ECHO_PIN 14
18  #define TRIG_PIN 12
19  #define led 27
20  void setup() {
21    // put your setup code here, to run once:
22    Serial.begin(115200);
23    pinMode(led, OUTPUT);
24    pinMode(TRIG_PIN, OUTPUT);
25    pinMode(ECHO_PIN, INPUT);
26    wifiConnect();
```

On the right, the 'Simulation' tab is active, showing a visual representation of the ESP32 and the Ultrasonic Distance Sensor. A slider indicates the measured distance is 156cm. Below the simulation, the serial output shows the following sequence of events:

```
publish ok
Measured distance: 155.94
Sending payload:{"distance":155.94}
publish ok
Measured distance: 152.97
Sending payload:{"distance":152.97}
publish ok
```

ALERT CASE:

The screenshot displays the Wokwi IDE interface. On the left, the Arduino sketch is visible, featuring the following code:

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* topic, byte* payload, unsigned int length) {
4   #define ORG "liyiqr"
5   #define DEVICE_TYPE "NodeMCU"
6   #define DEVICE_ID "1515"
7   #define TOKEN "123456789"
8   String data3;
9   char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
10  char publishTopic[] = "iot-2/evt/distance/fmt/json";
11  char subscribeTopic[] = "iot-2/cmd/test/fmt/String";
12  char authMethod[] = "use-token-auth";
13  char token[] = TOKEN;
14  char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
15  PubSubClient wificlient;
16  PubSubClient client(server, 1883, callback, wificlient);
17  #define ECHO_PIN 14
18  #define TRIG_PIN 12
19  #define led 27
20  void setup() {
21    // put your setup code here, to run once:
22    Serial.begin(115200);
23    pinMode(led, OUTPUT);
24    pinMode(TRIG_PIN, OUTPUT);
25    pinMode(ECHO_PIN, INPUT);
26    wificlient.connect();
```

On the right, the simulation window shows an Ultrasonic Distance Sensor connected to an ESP32. The sensor's distance is set to 61 cm. The console output shows the following sequence of events:

```
publish ok
Measured distance: 60.94
Sending payload:{"ALERT":60.94}
publish ok
Measured distance: 60.94
Sending payload:{"ALERT":60.94}
publish ok
```

IBM CLOUD STORAGE:

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. The main content area displays the 'Recent Events' tab for a device named 'liyiqr'. The events are listed in a table with columns: Event, Value, Format, and Last Received.

Event	Value	Format	Last Received
distance	{"ALERT":60.94}	json	a few seconds ago
distance	{"ALERT":60.94}	json	a few seconds ago
distance	{"distance":155.94}	json	a few seconds ago
distance	{"distance":155.94}	json	a few seconds ago
distance	{"distance":155.99}	json	a few seconds ago

At the bottom of the dashboard, a status bar indicates '2 Simulations running'.