

Assignment Date	22 October 2022
Student Name	Abishek R
Student Roll Number	2019504503
Maximum Marks	2

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 "90mku9"

#define DEVICE_TYPE "NodeMCU"

#define DEVICE_ID "9786450"

#define TOKEN "76453494567"

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 12

#define TRIG_PIN 13

#define led 14
```

```

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/346382091821253204>

OUTPUT:

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 void callback(char* subscribetopic,byte* payload,unsigned int payloadLength)
4 #define ORG "90mku9"
5 #define DEVICE_TYPE "NodeMCU"
6 #define DEVICE_ID "9786450"
7 #define TOKEN "76453494567"
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 WiFiClient wifiClient;
16 PubSubClient client(server,1883,callback,wifiClient);
17 #define ECHO_PIN 12
18 #define TRIG_PIN 13
19 #define led 14
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();
27 mqttConnect();
28 }
29 float readDistanceCM() {
30 float distance = 0;
31 // Send the trigger pulse
32 digitalWrite(TRIG_PIN, LOW); // Close the trigger
33 delay(2);
34 digitalWrite(TRIG_PIN, HIGH);
35 delay(10);
36 digitalWrite(TRIG_PIN, LOW);
37 // Receive the echo pulse
38 pinMode(ECHO_PIN, INPUT);
39 while(digitalRead(ECHO_PIN) == LOW)
40 delay(1);
41 while(digitalRead(ECHO_PIN) == HIGH)
42 delay(1);
43 distance = (micros() - startMicros) / 29 / 2;
44 return distance;
45 }
```

publish ok
Measured distance: 49.98
Sending payload: {"ALERT":49.98}
publish ok
Measured distance: 49.98
Sending payload: {"ALERT":49.98}
publish ok

Event	Value	Format	Last Received
distance	{"ALERT":49.98}	json	a few seconds ago
distance	{"ALERT":49.98}	json	a few seconds ago
distance	{"ALERT":49.98}	json	a few seconds ago
distance	{"ALERT":1.99}	json	a few seconds ago
distance	{"ALERT":1.99}	json	a few seconds ago

Items per page 20 | 1-1 of 1 item

0 Simulations running