

## ASSIGNMENT 4

**Team ID** :- PNT2022TMID33628

**Project name** :- IOT based smart crop protection for agricultural system

**Name** :- Vignesh R

### QUESTION:

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. Upload document with wokwi share link and images of ibm cloud.

### SOLUTION:

#### CODE:

```
#include <WiFi.h> //library for wifi
#include <PubSubClient.h> //library for MQTT
// creating the instance by passing pin and typr of dht connected
float distance;
#define sound_speed 0.034
int trigpin=18;
int echopin=19;
int led=5;
int LED=9;
long duration;
```

```

String message;// creating the instance by passing pin and typr of dht connected

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

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

#define ORG "93oivx"//IBM ORGANITION ID
#define DEVICE_TYPE "NodeMCU"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "12345"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678" //Token
String data3;
float h, t;

//----- Customise the above values -----
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event
perform and format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd REPRESENT
command type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id

//-----

```

```

WiFiClient wifiClient; // creating the instance for wificlient

PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined client
id by passing parameter like server id,portand wificredential

void setup()// configureing the ESP32
{
    Serial.begin(115200);
    pinMode(trigpin,OUTPUT);
    pinMode(echopin,INPUT);
    pinMode(led,OUTPUT);
    delay(10);
    Serial.println();
    wificonnect();
    mqttconnect();
}

void loop()// Recursive Function
{

    digitalWrite(trigpin,LOW);
    digitalWrite(trigpin,HIGH);
    delay(1000);
    digitalWrite(trigpin,LOW);
    duration=pulseIn(echopin,HIGH);
    distance=duration*sound_speed/2;
    Serial.println("distance"+String(distance)+"cm");
    if(distance<100)
    {

```

```

    message="Alert";
    digitalWrite(led,HIGH);
} else
{
    message="No problem";
    digitalWrite(led,LOW);
}
delay(1000);
PublishData(distance,message);
if (!client.loop()) {
    mqttconnect();
}
}

```

/.....retrieving to Cloud...../

```

void PublishData(float d, String a) {
    mqttconnect();//function call for connecting to ibm
    /*
        creating the String in in form JSon to update the data to ibm cloud
    */
    String payload = "{\"distance\":\"";
    payload += d; payload += "\"";
    payload += "," " {"\"message\":\"";

```

```
payload += a;  
payload += "}";
```

```
Serial.print("Sending payload: ");  
Serial.println(payload);
```

```
if (client.publish(publishTopic, (char*) payload.c_str())) {
```

```
    Serial.println("Publish ok");// if it sucessfully upload data on the cloud then it will  
    print publish ok in Serial monitor or else it will print publish failed
```

```
    } 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(".");
```

```
        initManagedDevice();
```

```
        Serial.println();
```

```
    }
```

```
}
```

```
void wificonnect() //function defination for wificonnect
```

```

{
  Serial.println();
  Serial.print("Connecting to ");

  WiFi.begin("Wokwi-GUEST", "", 6); //passing the wifi credentials to establish the
connection
  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];

}

data3="";

}

```

## DISTANCE IS LESS THAN 100 cms:

The screenshot shows the WOKWI IoT simulator interface. On the left, the sketch.ino file is open, displaying C++ code for an ESP32 microcontroller. The code includes libraries for WiFi, PubSubClient, and DHT. It defines pins for a trigpin (18), echopin (19), and an LED (5). The code sets up a WiFi connection and a PubSubClient instance. It defines a callback function for the subscribed topic and sets up the MQTT connection with IBM Watson IoT Platform credentials. The code also defines a data3 variable and a loop to process the payload. On the right, the simulation window shows the ESP32 microcontroller connected to an Ultrasonic Distance Sensor. The sensor's distance is set to 82cm. The console shows the sensor reading 81.97cm and the ESP32 sending a JSON payload: {"distance":81.97},{"message":"Alert"}.

## DISTANCE IS GREATER THAN 100 cms:

WOKWI

SAVE

SHARE

sketch.ino

Docs

M

sketch.ino

diagram.json

libraries.txt

Library Manager

```

1 #include <WiFi.h>//library for wifi
2 #include <PubSubClient.h>//library for MQTT
3 // creating the instance by passing pin and type of dht connected
4 float distance;
5 #define sound_speed 0.034
6 int trigpin=18;
7 int echopin=19;
8 int led=5;
9 int LED=0;
10 long duration;
11 String message;// creating the instance by passing pin and type of dht connected
12
13 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
14
15 //-----credentials of IBM Accounts-----
16
17 #define ORG "93oivx"//IBM ORGANIZATION ID
18 #define DEVICE_TYPE "NodeMCU"//Device type mentioned in ibm watson IOT Platform
19 #define DEVICE_ID "12345"//Device ID mentioned in ibm watson IOT Platform
20 #define TOKEN "12345678" //Token
21 String data3;
22 float h, t;
23
24
25 //----- Customise the above values -----
26 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
27 char publishTopic[] = "iot-2/evt/data/fmt/json";// topic name and type of event perform a
28 char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd REPRESENT command type AND
29 char authMethod[] = "use-token-auth";// authentication method
30 char token[] = TOKEN;
31 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
32
33

```

Simulation

04:10.869 100%

Editing Ultrasonic Distance Sensor

Distance: 223cm

distance222.96cm

Sending payload: {"distance":222.96},{"message":"No problem"}

distance222.96cm

Sending payload: {"distance":222.96},{"message":"No problem"}

distance222.96cm

## DEVICE RECENT EVENTS IN IBM WATSON:

IBM Watson IoT Platform

2019ecd219@swce.ac.in ID: 93oivx

Browse

Action

Device Types

Interfaces

Device ID

Status

Device Type

Class ID

Date Added

Descriptive Location

12345

Connected

NodeMCU

Device

Sep 24, 2022 3:36 PM

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	{"d":{"distance":222.96,"message":"No problem"}}	json	a few seconds ago
Data	{"d":{"distance":222.96,"message":"No problem"}}	json	a few seconds ago
Data	{"d":{"distance":81.97,"message":"Alert"}}	json	a few seconds ago
Data	{"d":{"distance":81.97,"message":"Alert"}}	json	a few seconds ago
Data	{"d":{"distance":81.97,"message":"Alert"}}	json	a few seconds ago



