

## Assignment 4

### Wokwi Assignment

Assignment Date	29 October 2022
Student Name	Ms. Gayam Naveena
Student Roll Number	711119104023
Maximum Marks	2 Marks

Assignment 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.

Wokowi Link: <https://wokwi.com/projects/346504267904844371>

Program code:

```
#include <WiFi.h> //library for wifi
```

```
#include <PubSubClient.h> //library for MQTT
```

```
#define ECHO_GPIO 12
```

```
#define TRIGGER_GPIO 13
```

```
#define MAX_DISTANCE_CM 100 // Maximum of 5 meters
```

```
#include "Ultrasonic.h"
```

```
Ultrasonic ultrasonic(13, 12);
```

```
int distance;
```

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

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

```

#define ORG "ydw0ta"//IBM ORGANITION ID

#define DEVICE_TYPE "ESP32"//Device type mentioned in ibm watson IOT Platform

#define DEVICE_ID "ESP32"//Device ID mentioned in ibm watson IOT Platform

#define TOKEN "123456789" //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);

    delay(10);

    Serial.println();

    wificonnect();

    mqttconnect();

```

```
}
```

```
void loop()// Recursive Function
```

```
{
```

```
distance = ultrasonic.read(CM);
```

```
if (distance < 100) {
```

```
    Serial.print("Distance in CM: ");
```

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

```
    PublishData(distance);
```

```
    delay(1000);
```

```
    if (!client.loop()) {
```

```
        mqttconnect();
```

```
    }
```

```
}
```

```
delay(1000);
```

```
}
```

```
/*.....retrieving to Cloud.....*/
```

```
void PublishData(float temp) {
```

```
    mqttconnect();//function call for connecting to ibm
```

```
    /*
```

```
        creating the String in in form JSon to update the data to ibm cloud
```

```
    */
```

```
    String payload = "{\"Alert Distance\":\"";
```

```
payload += temp;
```

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

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

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

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

```
    Serial.println("Publish ok");// if it successfully 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(".");
```

```
            delay(500);
```

```
        }
```

```
        initManagedDevice();
```

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

```
    }
```

```
}
```

```
void wificonnect() //function definition 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];
}

Serial.println("data: " + data3);

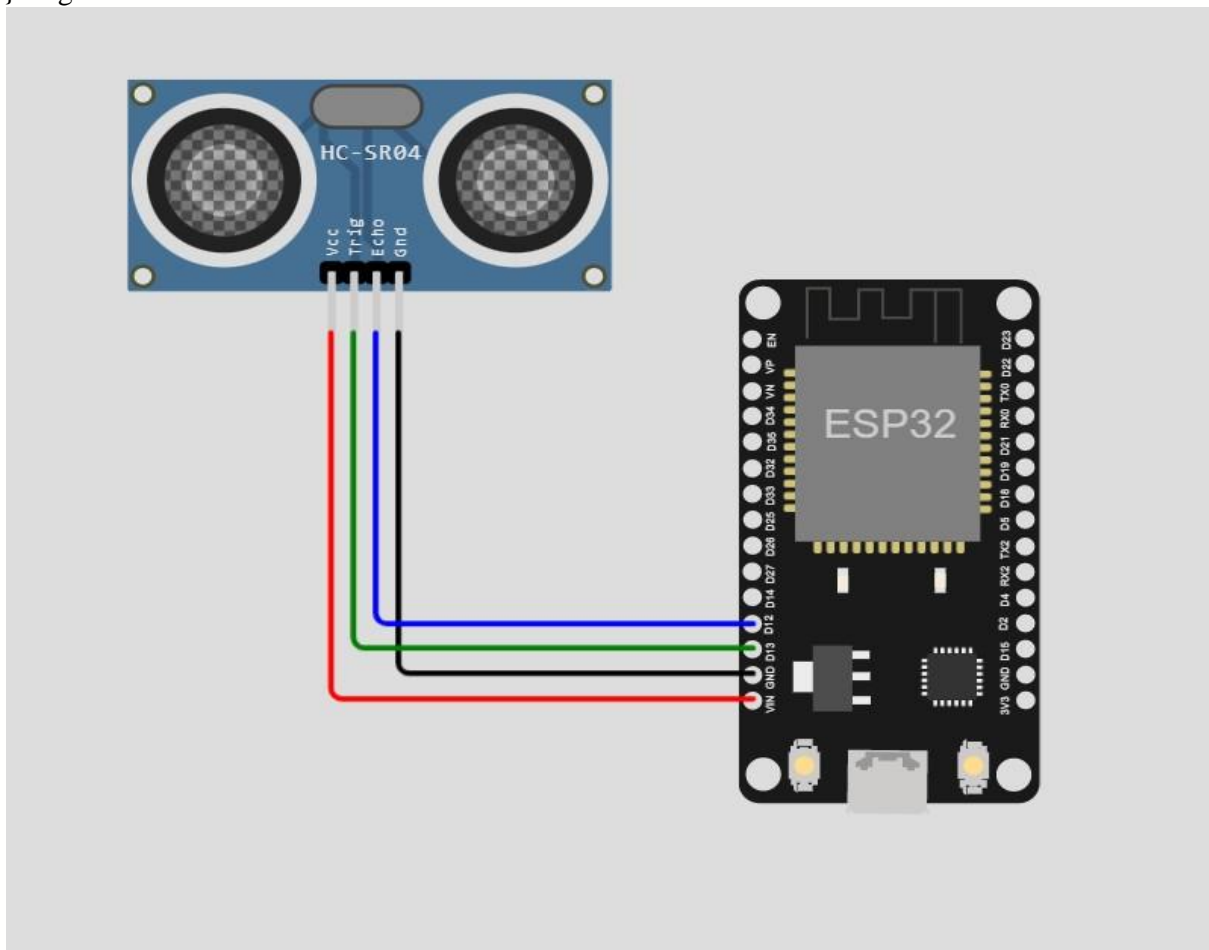
if (data3 == "lighton")
{
    Serial.println(data3);
}

else

{
    Serial.println(data3);
}

data3 = "";
}
}
Diagram:

```



When the distance is less than 100cm the alert is sent to the IBM cloud

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A table lists devices, with one ESP32 device shown as 'Connected'. Below the table, the 'Recent Events' tab is selected, displaying a list of events. The events are JSON payloads containing distance data, such as {"Alert Distance":91}. The dashboard also shows '0 Simulations running' and 'Items per page 50 | 1-1 of 1 item'.

Event	Value	Format	Last Received
Data	{"Alert Distance":91}	json	a few seconds ago
Data	{"Alert Distance":91}	json	a few seconds ago
Data	{"Alert Distance":91}	json	a few seconds ago
Data	{"Alert Distance":91}	json	2 minutes ago
Data	{"Alert Distance":91}	json	2 minutes ago

The screenshot shows the Wokwi IDE interface. On the left, the 'sketch.ino' file is open, displaying the Arduino code. The code includes headers for WiFi, PubSubClient, and the Ultrasonic library. It defines pins for the ultrasonic sensor and sets up the MQTT connection to the IBM Watson IoT Platform. The main loop reads the distance from the sensor and publishes an alert if the distance is less than 100cm. On the right, the 'Simulation' window shows a visual representation of the ESP32 and the ultrasonic sensor. The distance is displayed as 87cm. Below the simulation, the output console shows the MQTT messages being sent, including the payload {"Alert Distance":91.00}.

```
1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3
4 #define ECHO_GPIO 12
5 #define TRIGGER_GPIO 13
6 #define MAX_DISTANCE_CM 100 // Maximum of 5 meters
7 #include "Ultrasonic.h"
8
9 Ultrasonic ultrasonic(13, 12);
10 int distance;
11
12 void callback(char* topic, byte* payload, unsigned int payloadlength);
13
14 //-----credentials of IBM Accounts-----
15
16 #define ORG "ydw0ta" //IBM ORGANIZATION ID
17 #define DEVICE_TYPE "ESP32" //Device type mentioned in ibm watson IoT Platform
18 #define DEVICE_ID "ESP32" //Device ID mentioned in ibm watson IoT Platform
19 #define TOKEN "123456789" //token
20 String data3;
21 float h, t;
22
23 //----- Customise the above values -----
24
25 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
26 char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of event perform
27 char subscribeTopic[] = "iot-2/cmd/command/fmt/String"; // cmd REPRESENT command type AND
28 char authMethod[] = "use-token-auth"; // authentication method
29 char token[] = TOKEN;
30 char clientId[] = "dt:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
31
32 //-----
33
34 WiFiClient wificlient; // creating the instance for wificlient
35 PubSubClient client(server, 1883, callback, wificlient); //calling the predefined client
```

When the distance is more than 100cm the alert is not sent to the IBM cloud

IBM Watson IoT Platform

gn711119104023@t-lac.in  
ID: ydw0ta

Browse Action Device Types Interfaces

Add Device

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
ESP32	Connected	ESP32	Device	Oct 28, 2022 3:47 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
-------	-------	--------	---------------

Waiting for device events...

0 Simulations running

WOKWI

SAVE SHARE

Docs SIGN UP

sketch.ino diagram.json libraries.txt Ultrasonic.h Ultrasonic.cpp Library Manager

```
1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3
4 #define ECHO_GPIO 12
5 #define TRIGGER_GPIO 13
6 #define MAX_DISTANCE_CM 100 // Maximum of 5 meters
7 #include "Ultrasonic.h"
8
9 Ultrasonic ultrasonic(13, 12);
10 int distance;
11
12 void callback(char* topic, byte* payload, unsigned int payloadlength);
13
14 //-----credentials of IBM Accounts-----
15
16 #define ORG "ydw0ta" //IBM ORGANIZATION ID
17 #define DEVICE_TYPE "ESP32" //Device type mentioned in ibm watson IOT Platform
18 #define DEVICE_ID "ESP32" //Device ID mentioned in ibm watson IOT Platform
19 #define TOKEN "123456789" //Token
20 String data;
21 float h, t;
22
23
24 //----- Customise the above values -----
25 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
26 char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of event perform
27 char subscribeTopic[] = "iot-2/cmd/command/fmt/String"; // cmd REPRESENT command type AND
28 char authMethod[] = "use-token-auth"; // authentication method
29 char token[] = TOKEN;
30 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
31
32
33 //-----
34 WiFiClient wificlient; // creating the instance for wificlient
35 PubSubClient client(server, 1883, callback, wificlient); //calling the predefined client
```

Simulation

00:18.545 100%

Editing Ultrasonic Distance Sensor

Distance: 228cm

WiFi connected

IP address:  
10.10.0.2

Reconnecting client to ydw0ta.messaging.internetofthings.ibmcloud.com

iot-2/cmd/command/fmt/String

subscribe to cmd OK