

Assignment -4

Assignment Date	27.10.2022
Student Name	Mr.W.Merun raj
Student Roll Number	821919104012
Maximum Marks	2 Marks

Question :

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.

Wokwi Link:

<https://wokwi.com/projects/322410731508073042>

Code:

```
#include <WiFi.h>
#include <PubSubClient.h>
WiFiClient wifiClient;
String data3;
#define ORG "r29hf1"
#define DEVICE_TYPE "merun"
#define DEVICE_ID "Assignment4"
#define TOKEN "123456789"
#define speed 0.034
#define led 14
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/data/fmt/json";
char topic[] = "iot-2/cmd/home/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
```

```

char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, wifiClient);
void publishData();
const int trigpin = 5;
const int echopin = 18;
String command;
String data = "";
long duration;
float dist;

void setup()
{
  Serial.begin(115200);
  pinMode(led, OUTPUT);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, INPUT);
  wifiConnect();
  mqttConnect();
}

void loop() {
  bool isNearby = dist < 100;
  digitalWrite(led, isNearby);
  publishData();
  delay(500);
  if (!client.loop()) {
    mqttConnect();
  }
}

void wifiConnect() {
  Serial.print("Connecting to "); Serial.print("Wifi");
  WiFi.begin("Wokwi-GUEST", "", 6);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.print("WiFi connected, IP address: ");
  Serial.println(WiFi.localIP());
}

void mqttConnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting MQTT client to "); Serial.println(server);
    while (!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }
  }
}

```

```

initManagedDevice();
Serial.println();
}
}
void initManagedDevice() {
if (client.subscribe(topic)) {
// Serial.println(client.subscribe(topic));
Serial.println("IBM subscribe to cmd OK");
} else {
Serial.println("subscribe to cmd FAILED");
}
}
void publishData()
{
digitalWrite(trigpin, LOW);
digitalWrite(trigpin, HIGH);
delayMicroseconds(10);
digitalWrite(trigpin, LOW);
duration = pulseIn(echopin, HIGH);
dist = duration * speed / 2;
if (dist < 100) {
String payload = "{\"Normal Distance\":\"";
payload += dist;
payload += "\"}";
Serial.print("\n");
Serial.print("Sending payload: ");
Serial.println(payload);
if (client.publish(publishTopic, (char*) payload.c_str())) {
Serial.println("Publish OK");
}
}
if (dist > 101 ) {
String payload = "{\"Alert distance\":\"";
payload += dist;
payload += "\"}";
Serial.print("\n");
Serial.print("Sending payload: ");
Serial.println(payload);
if (client.publish(publishTopic, (char*) payload.c_str())) {
Serial.println("Warning crosses 110cm -- it automatically of the loop");
digitalWrite(led, HIGH);
} else {
Serial.println("Publish FAILED");
}
}
}

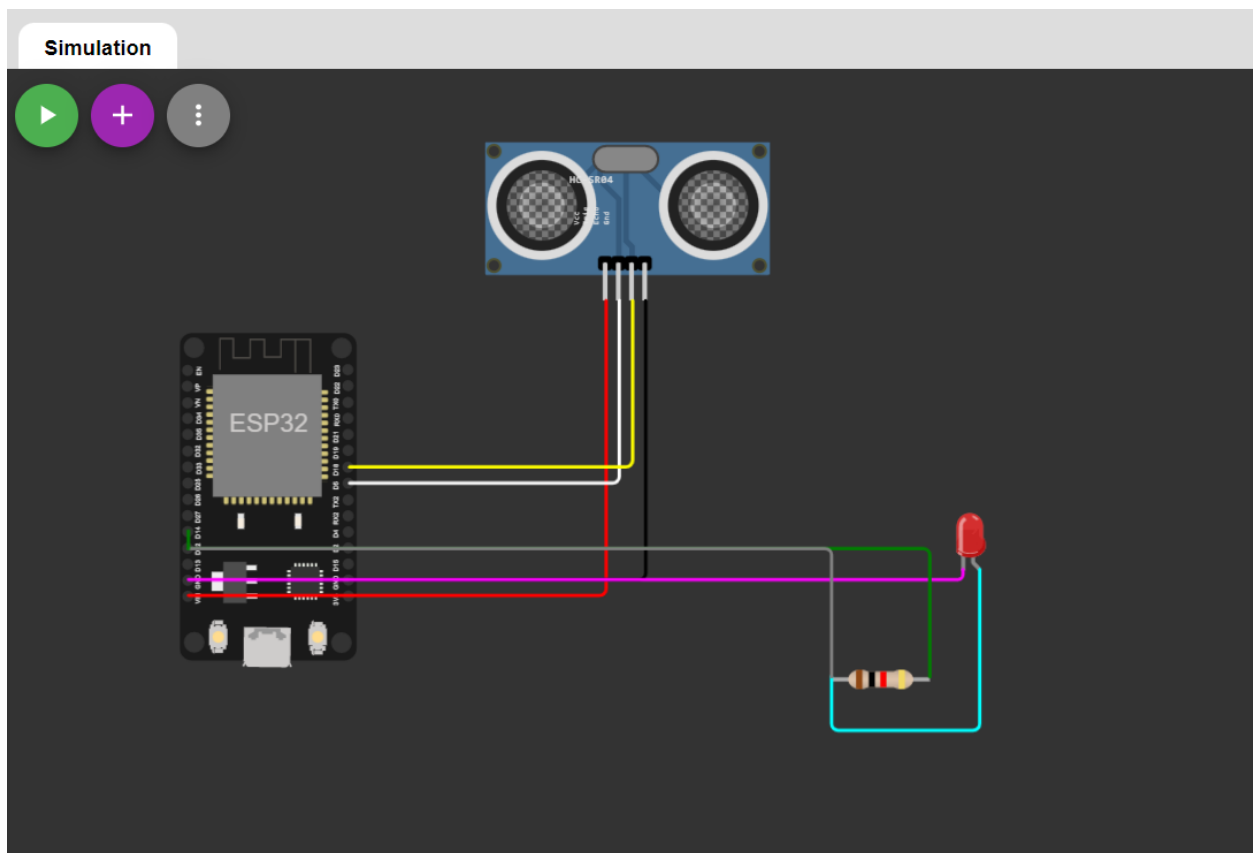
```

```

}
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++) {
    dist += (char)payload[i];
  }
  Serial.println("data:" + data3);
  if (data3 == "lighton") {
    Serial.println(data3);
    digitalWrite(led, HIGH);
  }
  data3 = "";
}
}

```

Circuit Diagram:



Output:

The screenshot displays the Wokwi IoT Platform interface for a project named "esp32-dht22.ino". The code on the left is an Arduino sketch that configures an ESP32 to act as an IoT device. It includes the necessary libraries, defines the device name, ID, and token, and sets up the MQTT client to publish distance data to a specific topic on the IBM Watson IoT Platform. The simulation on the right shows the hardware components: an ESP32 microcontroller, an ultrasonic sensor, and an LED. The console output shows the device successfully sending distance data (19.98) to the cloud.

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 WiFiClient wificlient;
4 String data3;
5 #define ORG "r29hf1"
6 #define DEVICE_TYPE "merun"
7 #define DEVICE_ID "Assignment4"
8 #define TOKEN "123456789"
9 #define speed 0.034
10 #define led 14
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/data/fmt/json";
13 char topic[] = "iot-2/cmd/home/fmt/String";
14 char authMethod[] = "use-token-auth";
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
17 PubSubClient client(server, 1883, wificlient);
18 void publishData();
19 const int trigpin = 5;
20 const int echopin = 18;
21 String command;
22 String data = "";
23 long duration;
24 float dist;
25
26 void setup()
27 {
28   Serial.begin(115200);
29   pinMode(led, OUTPUT);
30   pinMode(trigpin, OUTPUT);
31   pinMode(echopin, INPUT);
32   wificlient.begin();
33 }
```

Simulation

00:48.509 99%

PUBLISH OK

Sending payload: {"Normal Distance":19.98}
Publish OK

Sending payload: {"Normal Distance":19.98}
Publish OK

Sending payload: {"Normal Distance":19.98}
Publish OK

Sending payload: {"Normal Distance":19.98}

esp32-dht22.iino - Wokwi Arduino

IBM Watson IoT Platform

← → ↻ r29hf1.internetofthings.ibmcloud.com/dashboard/devices/browse

IBM Watson IoT Platform

821919104012@smartinternz.com
ID: r29hf1

Browse

Action

Device Types

Interfaces

Search by Device ID

Device Simulator ☒

ID

Device ID

Status

Device Type

Class ID

Date Added

Descriptive Location

Assignment4

Connected

merun

Device

Nov 6, 2022 11:18 AM

→ ...

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	{"Normal Distance":19.98}	json	a few seconds ago
data	{"Normal Distance":19.98}	json	a few seconds ago
data	{"Normal Distance":19.98}	json	a few seconds ago
data	{"Normal Distance":19.98}	json	a few seconds ago
data	{"Normal Distance":19.98}	json	a few seconds ago