

## ASSIGNMENT-- 4

### Industry-specific intelligent fire management system

Team id : PNT2022TMID29717

#### 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.

#### ❖ Code :

```
#include <WiFi.h>
#include <PubSubClient.h>

#define ORG "0bm892"
#define DEVICE_TYPE "ESP32_Controller"
#define DEVICE_ID "Sensor"
#define TOKEN "1234567890"
#define trigpin 5
#define echopin 18
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/data/fmt/json";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
WiFiClient wifiClient;
PubSubClient client(server, 1883, wifiClient);
long duration;
float dist;
void setup()
{
    Serial.begin(9900);

    pinMode(trigpin, OUTPUT);
    pinMode(echopin, INPUT);
    wifiConnect();
    mqttConnect();
}
void loop() {

    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);
        }
        Serial.println();
    }
}

void publishData()
{
    digitalWrite(trigpin, LOW);
    digitalWrite(trigpin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigpin, LOW);
    duration=pulseIn(echopin, HIGH);
    dist=(duration*0.034) /2;
    if(dist<100)
    {
        String payload = "{\"Distance\":\"";
        payload += dist;
        payload += ",";
        payload += "\"Status\":\"";
        payload += "\"Alert\"}";
        Serial.print("\n");
        Serial.print("Sending payload: ");
        Serial.println(payload);
        if (client.publish(publishTopic, (char*) payload.c_str()))
    }
}

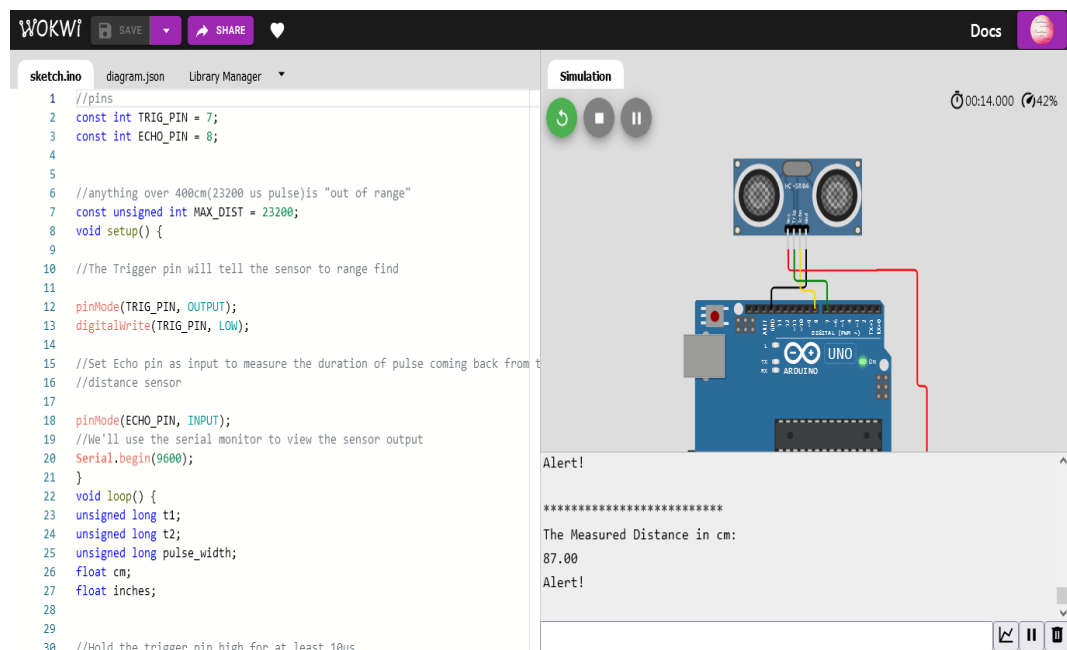
```

```

{
  Serial.println("Publish OK");
}
}
if(dist>100)
{
  String payload = "{\"Distance\":\"";
  payload += dist;
  payload += ",";
  payload += "\"Status\":\"";
  payload += "\"Normal\"}";
  Serial.print("\n");
  Serial.print("Sending payload: ");
  Serial.println(payload);
  if(client.publish(publishTopic, (char*) payload.c_str()))
  {
    Serial.println("Publish OK");
  }
  else
  {
    Serial.println("Publish FAILED");
  }
}
}
}

```

## ❖ Execution :



WOKWI
SAVE
SHARE
Docs

sketch.ino

diagram.json

Library Manager

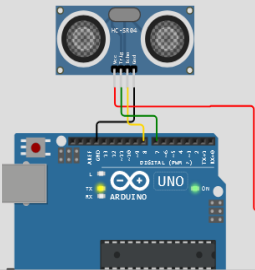
```

27 float inches;
28
29
30 //Hold the trigger pin high for at least 10us
31
32 digitalWrite(TRIG_PIN, HIGH);
33 delayMicroseconds(10);
34 digitalWrite(TRIG_PIN, LOW);
35
36 //wait for pulse on echo pin
37
38 while (digitalRead(ECHO_PIN) == 0);
39
40 //Measure how long the echo pin was held high (pulse width)
41 //note the micros() counter will overflow after ~70min
42
43 t1 = micros();
44 while (digitalRead(ECHO_PIN) == 1);
45 t2 = micros();
46 pulse_width = t2 - t1;
47
48
49
50 //calculate distance in centimeters and inches. The constants are found in the
51 //datasheet, and calculated from the assumed speed of sound in air at sea level
52
53 cm = pulse_width / 58;
54 inches = pulse_width / 148.0;
55
56 //print out results

```

Simulation

00:43.783 20%



The Measured Distance in cm:  
87.00  
Alert!  
\*\*\*\*\*  
The Measured Distance in cm:  
87.00

WOKWI
SAVE
SHARE
Docs

sketch.ino

diagram.json

Library Manager

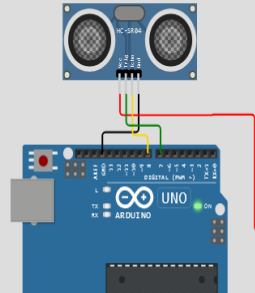
```

56 //print out results
57 if (pulse_width > MAX_DIST) {
58   Serial.println("Out of range");
59 }
60 else
61 {
62   Serial.println("*****");
63   Serial.println("The Measured Distance in cm:");
64   Serial.println(cm);
65   if (cm < 100)
66   {
67     //while (true)
68     {
69       Serial.println("Alert!");
70     }
71   }
72   Serial.println("");
73 }
74
75 //wait at least 1000ms before next measurement
76 delay(1000);
77 }
78

```

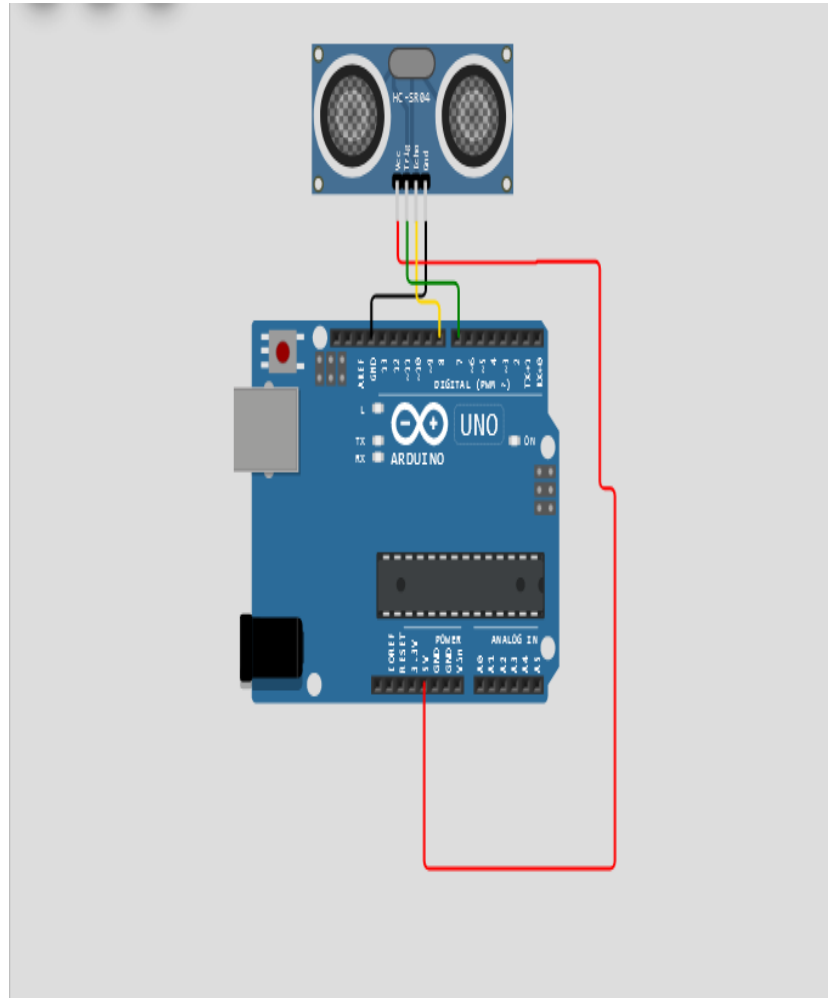
Simulation

00:52.350 42%



The Measured Distance in cm:  
87.00  
Alert!  
\*\*\*\*\*  
The Measured Distance in cm:  
87.00

❖ circuit diagram :



## IBM cloud output :

The screenshot displays the IBM Cloud IoT Platform interface. At the top, there are tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. A blue 'Add Device +' button is located in the top right corner. Below the tabs, a header bar shows the device name 'ESP32\_Controller', its status 'Connected', and the time 'Oct 30, 2022 4:42 PM'. A dropdown menu is open, showing options: 'Identity', 'Device Information', 'Recent Events' (which is selected), 'State', and 'Logs'. Below the dropdown, a message states: 'The recent events listed show the live stream of data that is coming and going from this device.' A table follows, listing recent events with columns for 'Event', 'Value', 'Format', and 'Last Received'. The table contains five rows of data, all with a 'json' format and 'a few seconds ago' as the last received time. At the bottom right, a box indicates '0 Simulations running'.

Event	Value	Format	Last Received
data	{"Distance":1.99,"Status":"Alert"}	json	a few seconds ago
data	{"Distance":125.95,"Status":"Normal"}	json	a few seconds ago
data	{"Distance":125.95,"Status":"Normal"}	json	a few seconds ago
data	{"Distance":34.97,"Status":"Alert"}	json	a few seconds ago
data	{"Distance":34.97,"Status":"Alert"}	json	a few seconds ago

0 Simulations running

❖ Wokwi URL : <https://wokwi.com/projects/346948920308400723>