

PROBLEM STATEMENT:

IoT BASED GAS LEAKAGE MONITORING AND
ALERTING SYSTEM

DOMAIN:

INTERNET OF THINGS

ASSIGNMENT 4:

DISTANCE DETECTION USING ULTRASONIC
SENSOR

BY

SARANYA M (623519106035)

SRIPRIYA .S (623519106042)

SANTHOSH .S (623519106033)

SOUNDARYA .P (623519106037)

RANJITH KUMAR .L (623519106027)

QUESTION-1:

Write code and connections in wokwi for ultrasonic sensor. Whenever distance is less than 100 cm send "alert" to IBM cloud and display in device recent events.

WOKWI LINK:

<https://wokwi.com/projects/348028372173980243>

CODE:

```
#include <WiFi.h>
#include <WiFiClient.h>
#include <PubSubClient.h>
const int trigPin = 5;
const int echoPin = 18;
//define sound speed in cm/uS
#define SOUND_SPEED 0.034
#define CM_TO_INCH 0.393701
long duration;
float distanceCm;
float distanceInch;

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
//-----credentials of IBM Accounts-----
```

```

#define ORG "5lv9tl"//IBM ORGANITION ID
#define DEVICE_TYPE "Esp-32device"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "soundarya"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "9361356124" //Token
String data3;

//----- 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/test/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);

void setup() {
  Serial.begin(115200); // Starts the serial communication
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop() {
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);

```

```

// Calculate the distance
distanceCm = duration * SOUND_SPEED/2;

// Convert to inches
distanceInch = distanceCm * CM_TO_INCH;

// Prints the distance in the Serial Monitor
Serial.print("Distance (cm): ");
Serial.println(distanceCm);
Serial.print("Distance (inch): ");
Serial.println(distanceInch);

PublishData(distanceCm);
delay(1000);
if (!client.loop()) {
    mqttconnect();
}
}

void PublishData(float Cm) {
    mqttconnect();//function call for connecting to ibm
    /*
        creating the String in in form JSon to update the data to ibm cloud
    */
    String payload = "{\"Distance (cm)\":\"";
    payload += Cm;
    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(".");
        delay(500);
    }

    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];
    }
}

```

}

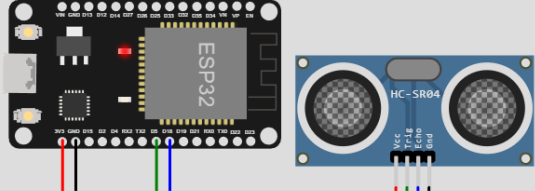
WOKWI SAVE SHARE esp32-arduino.ino copy Docs SIGN IN

esp32-blink.ino • diagram.json libraries.txt Library Manager

```
1 #include <WiFi.h>
2 #include <WiFiClient.h>
3 #include <PubSubClient.h>
4 const int trigPin = 5;
5 const int echoPin = 18;
6 //define sound speed in cm/uS
7 #define SOUND_SPEED 0.034
8 #define CM_TO_INCH 0.393701
9 long duration;
10 float distanceCm;
11 float distanceInch;
12
13
14 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
15 //-----credentials of IBM Accounts-----
16
17 #define ORG "51v9t1"//IBM ORGANITION ID
18 #define DEVICE_TYPE "Esp-32device"//Device type mentioned in ibm watson IOT Platform
19 #define DEVICE_ID "soundarya"//Device ID mentioned in ibm watson IOT Platform
20 #define TOKEN "9361356124" //Token
21 String data3;
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
27 char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT command
28 char authMethod[] = "use-token-auth";// authentication method
```

Simulation

00:24.609 35%



Distance (inch): 85.41
Sending payload: {"Distance (cm)":216.94}
Publish ok
Distance (cm): 216.94
Distance (inch): 85.41
Sending payload: {"Distance (cm)":216.94}
Publish ok

Activate Windows
Go to Settings to activate Windows.

IBM Watson IoT Platform 623519106037@smartinternz.com ID: 51v9t1

Browse Action Device Types Interfaces Add Device

Browse Devices

All Devices Diagnose

This table shows a summary of all devices that have been added. It can be filtered, organized, and searched on using different criteria. To get started, you can add devices by using the Add Device button, or by using API.

Search by Device ID Device Simulator 101

<input type="checkbox"/>	Device ID	Status	Device Type	Class ID	Date Added
> <input type="checkbox"/>	soundarya	Connected	Esp-32device	Device	11 Nov 2022 04:03

Items per page 50 | 1-1 of 1 item 1 of 1 page < 1 >

0 Simulations running

Activate Windows
Go to Settings to activate Windows.