

Problem statement : IoT based safety gadget for child safety monitoring and notification.

Domain : Internet of Things

Assignment 4: Distance detection using ultrasonic sensor

By,

S.KARTHICK(623519106301)

## Question-1:

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.

### WOKWI LINK:

<https://wokwi.com/projects/348839716434477651>

### 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 "mi51ii"//IBM ORGANITION ID
#define DEVICE_TYPE "ESP"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "karthiks"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "123456789" //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];
    }
}
}

```

## OUTPUT:

**When object is not near to the ultrasonic sensor**

← ↻

https://wokwi.com/projects/348839716434477651

🔊 🔍 🏠 📁 📄

WOKWI

SAVE

SHARE

esp32-arduino.ino copy

Docs

SIGN IN

esp32-blink.ino

diagram.json

libraries.txt

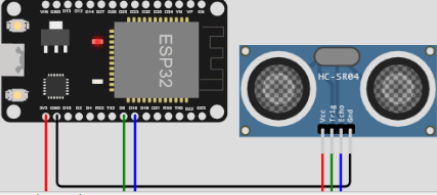
Library Manager

```
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 payloadLen
15 //-----credentials of IBM Accounts-----
16
17 #define ORG "mis111"//IBM ORGANITION ID
18 #define DEVICE_TYPE "ESP"//Device type mentioned in ibm watson IOT Platform
19 #define DEVICE_ID "karthiks"//Device ID mentioned in ibm watson IOT Platform
20 #define TOKEN "123456789" //Token
21 String data3;
22
23
24
25 //----- Customise the above values -----
26 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Na
27 char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of e
28 char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT comm
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 WiFiClient wificlient; // creating the instance for wificlient
```

Simulation

⏮ ⏪ ⏩ ⏭

00:05.396 1%



Distance (inch): 85.42

Sending payload: {"Distance (cm)":216.97}

Publish ok

Distance (cm): 216.94

Distance (inch): 85.41

Sending payload: {"Distance (cm)":216.94}

Publish ok

📈 ⏮ ⏪ ⏩ ⏭

# Data sent to the IBM cloud device when the object is far

The screenshot displays the IBM Watson IoT Platform interface. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A search bar labeled 'Search by Device ID' is present. The main content area shows a table of devices, with 'karthiks' selected. Below the device list, the 'Recent Events' tab is active, showing a stream of data events. The events table has columns for Event, Value, Format, and Last Received. The data shows distance measurements in centimeters, alternating between 216.94 and 217.38, with the most recent being 217.02.

Device ID	Status	Device Type	Class ID	Date Added
karthiks	Connected	ESP	Device	Nov 22, 2022 9:53 PM

Event	Value	Format	Last Received
Data	{"Distance (cm)":216.94}	json	a few seconds ago
Data	{"Distance (cm)":216.94}	json	a few seconds ago
Data	{"Distance (cm)":217.38}	json	a few seconds ago
Data	{"Distance (cm)":217.02}	json	a few seconds ago

