

# **ASSIGNMENT-4**

## **DISTANCE DETECTION USING ULTRASONIC SENSOR**

Date	01 November 2022
Team ID	PNT2022TMID38782
Maximum Marks	2 Marks

### **Question:**

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

### **Code:**

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

WiFiClient wifiClient;

#define ORG "9tg03j"
#define DEVICE_TYPE "RaspberryPi"
#define DEVICE_ID "12345"
#define TOKEN "12345678"
#define speed 0.034

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/status1/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=19;
String command;
String data="";
String name="Alert";
String icon="";
long duration;
int dist;
void setup()
{
  Serial.begin(115200);
  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(".");
      Serial.print("*");
      delay(1000);
    }
    initManagedDevice();
    Serial.println();
  }
}

void initManagedDevice() {

```

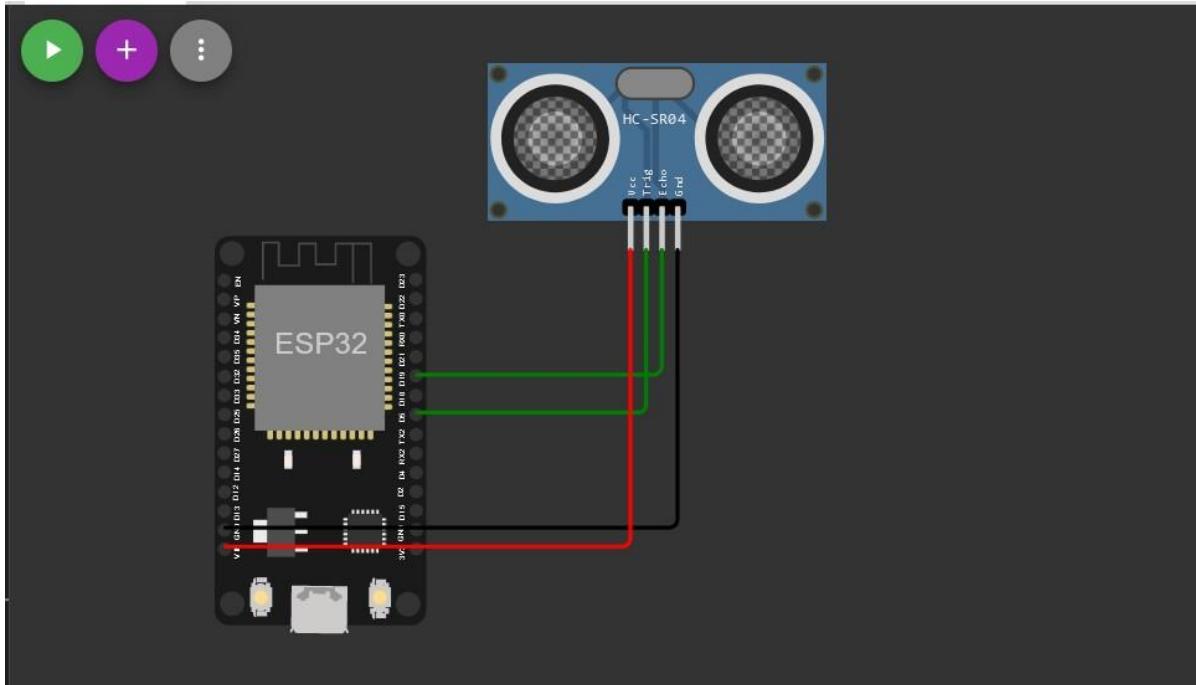
```

if (client.subscribe(topic)) {
    Serial.println(client.subscribe(topic));
    Serial.println("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){
        dist=100-dist;
        icon="Not-Crashed";
    }
    else{
        dist=0;
        icon="Crashed";
    }
    DynamicJsonDocument doc(1024);
    String payload;
    doc["Name"]=name;
    doc["Impact"]=icon;
    doc["Distance"]=dist;
    serializeJson(doc, payload);
    delay(3000);
    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");
    }
}

```

## DIAGRAM:



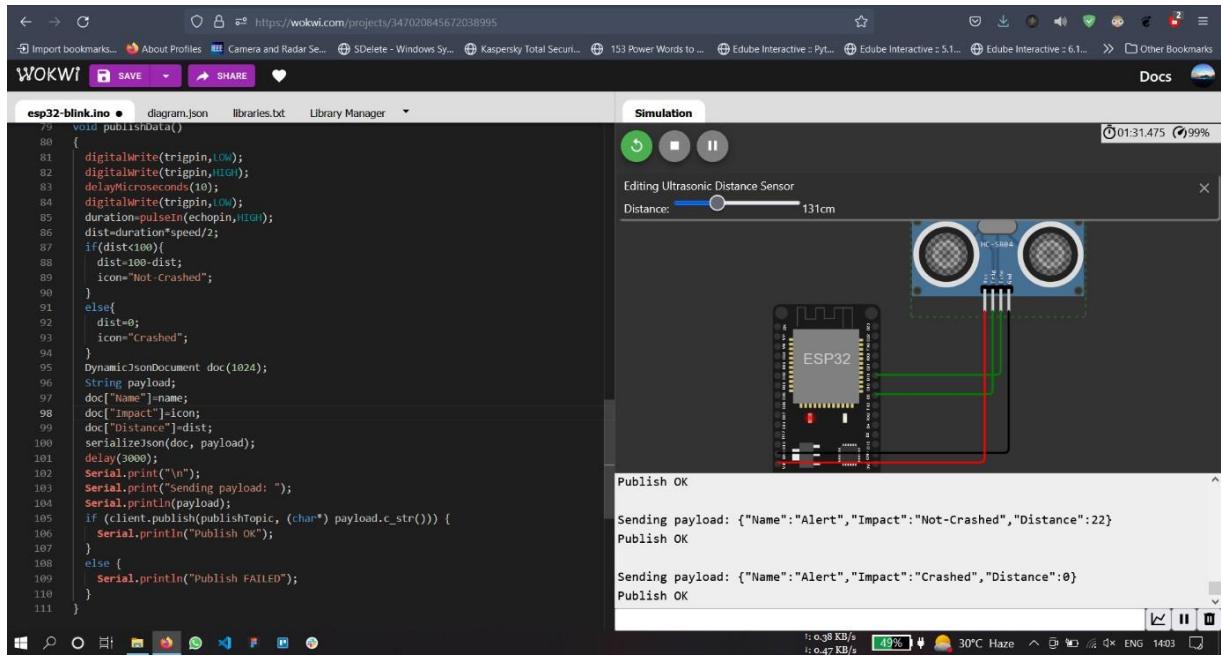
## OUTPUT:

The screenshot shows the Wokwi simulation interface. On the left, the code for `esp32-blink.ino` is displayed:

```
//> void publishData() {  
80  {  
81    digitalWrite(trigpin,LOW);  
82    digitalWrite(trigpin,HIGH);  
83    delayMicroseconds(10);  
84    digitalWrite(trigpin,LOW);  
85    duration=pulseIn(chopin,HIGH);  
86    dist=duration*speed/2;  
87    if(dist<100){  
88      dist=100-dist;  
89      icon="Not-Crashed";  
90    }  
91    else{  
92      dist=0;  
93      icon="Crashed";  
94    }  
95    DynamicJsonDocument doc(1024);  
96    String payload;  
97    doc["Name"]=name;  
98    doc["Impact"]=icon;  
99    doc["Distance"]=dist;  
100   serializeJson(doc, payload);  
101   delay(3000);  
102   Serial.print("\n");  
103   Serial.print("Sending payload: ");  
104   Serial.println(payload);  
105   if (client.publish(pubTopic, (char*) payload.c_str())) {  
106     Serial.println("Publish OK");  
107   }  
108   else {  
109     Serial.println("Publish FAILED");  
110   }  
111 }
```

On the right, the simulation window shows the ESP32 module and the HC-SR04 sensor. A slider labeled "Distance" is set to 79cm. The simulation status bar indicates "00:53.745" and "99%". The terminal window at the bottom shows the published payloads:

```
Publish OK  
Sending payload: {"Name":"Alert","Impact":"Not-Crashed","Distance":22}  
Publish OK  
Sending payload: {"Name":"Alert","Impact":"Not-Crashed","Distance":21}  
Publish OK
```



## Data uploaded to IoT Watson Platform

The screenshot shows the IBM Watson IoT Platform dashboard. The device list on the left shows a single device with ID 12345, which is connected and assigned to a Rasp Pi. The main view shows the device details for 12345, including its connection status, device type (Rasp), and last update time (Oct 14, 2022 9:55 AM). Below this, the 'Recent Events' tab is selected, displaying the following event log:

Event	Value	Format	Last Received
status1	{"Name": "Alert", "Icon": "trash", "FillPercent": 0}	json	a few seconds ago
status1	{"Name": "Alert", "Icon": "trash", "FillPercent": 0}	json	a few seconds ago
status1	{"Name": "Alert", "Icon": "trash", "FillPercent": 0}	json	a few seconds ago
status1	{"Name": "Alert", "Icon": "trash", "FillPercent": 0}	json	a few seconds ago

At the bottom, there are pagination controls for items per page (50) and a note indicating 1 of 1 page.