

ASSIGNMENT 4

Name: A.Sugapriya

TEAM ID: PNT2022TMID06149

Ultrasonic sensor simulation in Wokwi

Question: Write a code and connections in wokwi for the ultrasonic sensor. Whenever the distance is less than 100cms send an "Alert" to IBM cloud and display in the device recent events.

```
#include <WiFi.h>
#include <PubSubClient.h>
void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);
//-----credentials of IBM Accounts-----
#define ORG "91xobn"//IBM ORGANITION ID
#define DEVICE_TYPE "ESP32PROJECT"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "ESP32"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "ESP32PROJECT" //Token
String data3;
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/Data/fmt/json";
char subscribetopic[] = "iot-2/cmd/test/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
WiFiClient wifiClient;
PubSubClient client(server, 1883, callback ,wifiClient);
const int trigPin = 5;
const int echoPin = 18;
#define SOUND_SPEED 0.034
long duration;
float distance;
void setup() {
  Serial.begin(115200);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  wificonnect();
  mqttconnect();
}
void loop()
{
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
```

```

delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distance = duration * SOUND_SPEED/2;
Serial.print("Distance (cm): ");
Serial.println(distance);
if(distance<100)
{
Serial.println("ALERT!!");
delay(1000);
PublishData(distance);
delay(1000);
if (!client.loop()) {
mqttconnect();
}
}
delay(1000);
}
void PublishData(float dist) {
mqttconnect();
String payload = "{\"Distance\": ";
payload += dist;
payload += ", \"ALERT!!\": \"\" \"Distance less than 100cms\"";
payload += "}";
Serial.print("Sending payload: ");
Serial.println(payload);
if (client.publish(publishTopic, (char*) payload.c_str())) {
Serial.println("Publish ok");
} 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()

```

```

{
  Serial.println();
  Serial.print("Connecting to ");
  WiFi.begin("Wokwi-GUEST", "", 6);
  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];
  }
  Serial.println("data: "+ data3);
  data3="";
}

```

Diagram json

```

{
  "version": 1,
  "author": "Sugapriya A",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 23.33, "left": -106,
      "attrs": {} },

```

```

{ "type": "wokwi-hc-sr04", "id": "ultrasonic1", "top": -15.04, "left": 86.5,
  "attrs": {} }
],
"connections": [
[ "esp:TX0", "$serialMonitor:RX", "", [] ],
[ "esp:RX0", "$serialMonitor:TX", "", [] ],
[ "ultrasonic1:VCC", "esp:VIN", "red", [ "v168.58", "h-279.11", "v-66" ] ],
[ "ultrasonic1:GND", "esp:GND.1", "black", [ "v0" ] ],
[ "ultrasonic1:TRIG", "esp:D5", "green", [ "v0" ] ],
[ "ultrasonic1:ECHO", "esp:D18", "green", [ "v0" ] ]
]
}

```

libraries.txt:

```

# Wokwi Library List
# See https://docs.wokwi.com/guides/libraries

```

PubSubClient

Library Manager:

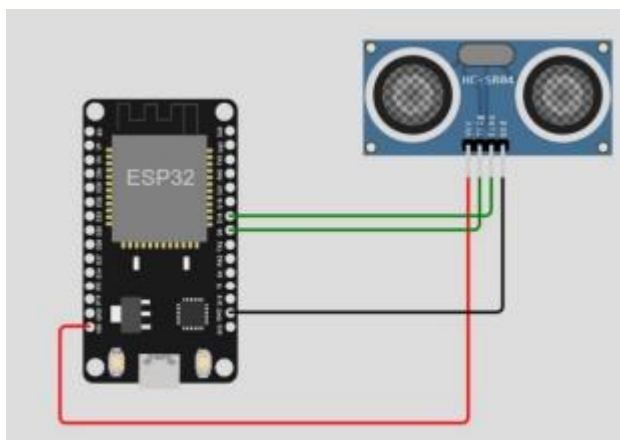
Installed Libraries:

PubSubClient

Wokwi simulation link:

<https://wokwi.com/projects/347584419737895506>

Circuit diagram:



Output:

```
Distance (cm): 399.94
Distance (cm): 399.94
Distance (cm): 142.95
Distance (cm): 72.96
ALERT!!
Sending payload: {"Distance":72.96,"ALERT!!":"Distance less than 100cms"}
Publish ok
```

IBM cloud output



The screenshot shows the IBM Cloud IoT Platform interface. On the left is a dark sidebar with icons for various functions. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A blue 'Add Device' button is on the right. Below the navigation bar, there are tabs for 'Identity', 'Device Information', 'Recent Events', 'State', and 'Logs'. The 'Recent Events' tab is selected. Below the tabs, a message states: 'The recent events listed show the live stream of data that is coming and going from this device.' Below this message is a table with four columns: 'Event', 'Value', 'Format', and 'Last Received'. The table contains four rows of data, all with 'event_1' in the 'Event' column and 'a few seconds ago' in the 'Last Received' column. The 'Value' column contains JSON strings: [{"distance":7,"Alert":"","Distance less than 10"}], [{"distance":9,"Alert":"","Distance less than 10"}], [{"distance":8,"Alert":"","Distance less than 10"}], and [{"distance":9,"Alert":"","Distance less than 10"}]. The 'Format' column contains the value 'json' for all rows.

Event	Value	Format	Last Received
event_1	[{"distance":7,"Alert":"","Distance less than 10"}]	json	a few seconds ago
event_1	[{"distance":9,"Alert":"","Distance less than 10"}]	json	a few seconds ago
event_1	[{"distance":8,"Alert":"","Distance less than 10"}]	json	a few seconds ago
event_1	[{"distance":9,"Alert":"","Distance less than 10"}]	json	a few seconds ago