

Assignment - 4
ESP 32 – Ultrasonic Sensor

Assignment Date	3 NOVEMBER 2022
Student Name	Parthipan M
Student Roll Number	411719106301
Maximum Marks	2 Marks

Question-1:

Write code and Connection in wokwi for ultrasonic sensor. Whenever distance is less than 100cms send "alert" to the ibm cloud and display in device recent events.

Solution:

Program:

```
#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 "b31tni"//IBM ORGANITION ID
#define DEVICE_TYPE "Assignment4"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "assignment"//Device ID mentioned in ibm watson IOT
Platform#define TOKEN "6r?TKCluy+okJ?9B+7" //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 successfully 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 definition 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 Simulation:

The screenshot displays the Wokwi web-based simulation environment. The interface includes a browser window at the top with tabs for IBM Watson IoT Platform, Wokwi Library List, and the current project. The main workspace is divided into three sections:

- Code Editor (Left):** Contains the Arduino sketch for an ESP32. It includes headers for `WiFi.h` and `PubSubClient.h`, and defines MQTT credentials for IBM Watson IoT Platform. The `initManagedDevice` function initializes the MQTT client and subscribes to a topic. The `callback` function processes the received payload.
- Simulation (Right):** Shows a 3D model of the ESP32 microcontroller connected to an ultrasonic sensor module. The sensor is positioned to measure distance. A console output at the bottom right of the simulation window shows a series of distance measurements in centimeters: 399.96, 399.94, 399.94, 399.94, 399.92, 399.94, and 399.94.
- Library Manager (Bottom Left):** Shows the installed libraries for the project.

The bottom of the screen shows a Windows taskbar with various application icons and system status information, including the time (10:51 AM) and date (11/4/2022).

IoT Watson Platform:

The screenshot displays the IBM Watson IoT Platform interface. The browser address bar shows the URL: `zowha.internetofthings.ibmcloud.com/dashboard/devices/drilldown/195:883855?returnTo=/devices/browse`. The user is logged in as `dhanyarameshrgp@gmail.com` with ID `zowha`.

The main content area is titled "Device Drilldown - 883855". It features a sidebar with navigation options: Device Credentials, Connection Information (selected), Recent Events, State, Device Information, Metadata, Diagnostics, Connection Logs, and Device Actions.

Under "Connection Information", the following details are shown:

- Date Added: NOV 4, 2022 11:27 AM
- Added By: dhanyarameshrgp@gmail.com
- Connection Status: Disconnected

The "Recent Events" section displays a table of live stream data:

Event	Value	Format	Last Received
event_1	{"randomNumber":22,"temp":13,"hump":80}	json	a few seconds ago
event_1	{"randomNumber":51,"temp":24,"hump":47}	json	a few seconds ago
event_1	{"randomNumber":56,"temp":8,"hump":32}	json	a few seconds ago
event_1	{"randomNumber":56,"temp":8,"hump":47}	json	a few seconds ago
event_1	{"randomNumber":34,"temp":29,"hump":79}	json	a few seconds ago

At the bottom right, a status bar indicates "2 Simulations running".

<https://wokwi.com/projects/347372760316510804>

