

Assignment - 4
ESP 32 – Ultrasonic Sensor

Assignment Date	01 November 2022
Student Name	Monish R
Student Roll Number	611219106047
Maximum Marks	2 Marks

Question-1:

Write code and Connection in wokwi for ultrasonic sensor.

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 "7dtxr4"//IBM ORGANITION ID
#define DEVICE_TYPE "monish"//Device type
mentioned in ibm watson IOT Platform
#define DEVICE_ID "monish123"//Device ID
mentioned in ibm watson IOT Platform
#define TOKEN "e)oFR*RTNM*NHbe2IM"
//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 Simulation:

The Wokwi Simulation interface displays the following code on the left and a virtual circuit diagram on the right.

```

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 "dtxr4"//IBM ORGANIZATION ID
18 #define DEVICE_TYPE "monish"//Device type mentioned in ibm watson IOT Platform
19 #define DEVICE_ID "monish212"//Device ID mentioned in ibm watson IOT Platform
20 #define TOKEN "eJofRRTNMWbeZIM" //Token
21 String data3;
22
23
24
25 //----- Customise the above values -----
26 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
27 char publishTopic[] = "iot-2/evt/data/fmt/json";// topic name and type of event perform a
28 char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT command type AND CO
29 char authmethod[] = "use-token-auth";// authentication method
30 char token[] = TOKEN;
31 char clientId[] = "di:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
32
33 WiFiClient wifiClient; // creating the instance for wifiClient
34 PubSubClient client(server, 1883, callback ,wifiClient);
35

```

The virtual circuit diagram on the right shows an ESP32 microcontroller connected to an HC-SR04 ultrasonic sensor. The sensor's VCC pin is connected to the ESP32's 5V pin, GND to GND, and the Trig pin to digital pin 5. The Echo pin is connected to digital pin 18. The simulation output on the right shows the following sequence of events:

- Distance (inch): 85.42
- Sending payload: {"Distance (cm)":216.95}
- Publish ok
- Distance (cm): 216.94
- Distance (inch): 85.41
- Sending payload: {"Distance (cm)":216.94}
- Publish ok

IoT Watson Platform:

The IBM Watson IoT Platform interface shows the following details for the device 'monish123':

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
monish123	Connected	monish	Device	Nov 1, 2022 12:29 PM	

The 'Recent Events' tab shows the following data:

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)":216.92}	json	a few seconds ago
Data	{"Distance (cm)":216.94}	json	a few seconds ago

Wokwi Project Link: <https://wokwi.com/projects/347113256143290963>