ASSIGNMENT 4

ULTRASONIC SENSOR SIMULATION IN WOKWI AND IBM CLOUD

Assignment Date	26 October 2022
Project ID	PNT2022TMID37828
Project Name	Project- Smart Solution For Railways
Maximum Marks	2 Marks

QUESTION:

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

Solution:

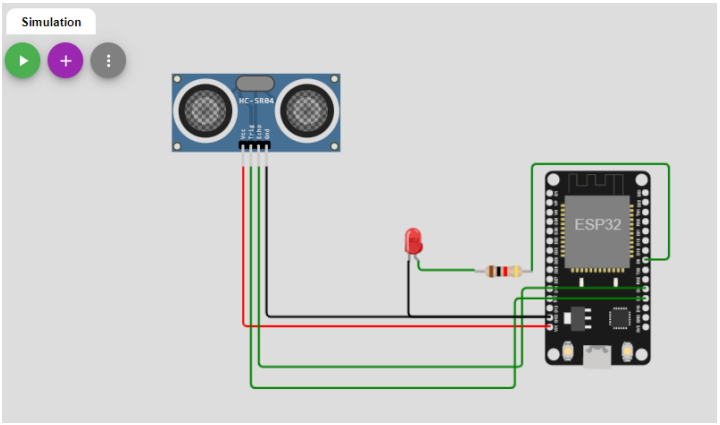
Introduction:

The HC-SR04 Ultrasonic Distance Sensor is connected to ESP32 and has 4 pins namely

Name	Description	Connection to ESP32
VCC	Voltage supply (5V)	Vin
TRIG	Pulse to start the measurement	D2
ECHO	Measure the high pulse length to get the distance	D4
GND	Ground	GND2

The distance is monitored by the ultrasonic sensor and if the distance is less than 100 cm, an "Alert Message" is sent to the IBM cloud. A led is made to glow if the distance is less than 100 cm.

Connection Diagram:



Code:

```
#include <WiFi.h>
#include <PubSubClient.h>
#include "Ultrasonic.h"
#define PIN_TRIG 2
#define PIN_ECHO 4
#define LED 5

void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);


#define ORG "iyxgzr"
#define DEVICE_TYPE "SensorNodes"
#define DEVICE_ID "97909150"
#define TOKEN "SEyIWOA(jsESMgHBaM"
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);

void setup()
{
  Serial.begin(115200);
  Serial.println("Distance Sensor");
  pinMode(PIN_TRIG, OUTPUT);
  pinMode(PIN_ECHO, INPUT);
  pinMode(LED, OUTPUT);
  wificonnect();
  mqttconnect();
}

float readDistanceCM() {
  digitalWrite(PIN_TRIG, LOW);
  delayMicroseconds(2);
  digitalWrite(PIN_TRIG, HIGH);
  delayMicroseconds(10);
  digitalWrite(PIN_TRIG, LOW);
  int duration = pulseIn(PIN_ECHO, HIGH);
```

```
  return duration * 0.034 / 2;
}

void loop()
{
  float distance = readDistanceCM();

  bool isNearby;
  if( distance < 100){
    digitalWrite(LED, HIGH);
  }
  else{
    digitalWrite(LED, LOW);
  }
  PublishData(distance);
    delay(1000);
    if (!client.loop()) {
      mqttconnect();
  }
  Serial.print("Measured distance: ");
  Serial.println(readDistanceCM());
  delay(100);
}
void PublishData(float dist) {
  mqttconnect();
  if(dist<100)
  {
  String payload = "{\"Distance\":";
  payload += dist;
  payload += ",\"Alert Message\":""\"The distance is less than 100 cm\"";
  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");
    }
  }
  else
  {
    String payload = "{\"Distance\":";
    payload += dist;
    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);
  if(data3=="lighton")
  {
Serial.println(data3);
digitalWrite(LED,HIGH);

  }

  else
  {
Serial.println(data3);
digitalWrite(LED,LOW);

  }
data3="";
}
```

Simulated output from Wokwi :

```
Distance Sensor

Connecting to .....
WiFi connected
IP address:
10.10.0.2
Reconnecting client to iyxgzr.messaging.internetofthings.ibmcloud.com
iot-2/cmd/test/fmt/String
subscribe to cmd OK

Sending payload: {"Distance":258.96}
Publish ok
Measured distance: 258.94
Sending payload: {"Distance":258.93}
Publish ok
Measured distance: 258.94
Sending payload: {"Distance":258.93}
Publish ok
```

```
Measured distance: 96.97
Sending payload: {"Distance":96.97,"Alert Message":"The distance is less than 100 cm"}
Publish ok
Measured distance: 96.97
Sending payload: {"Distance":96.97,"Alert Message":"The distance is less than 100 cm"}
Publish ok
Measured distance: 96.97
Sending payload: {"Distance":96.97,"Alert Message":"The distance is less than 100 cm"}
Publish ok
Measured distance: 96.97
Sending payload: {"Distance":96.97,"Alert Message":"The distance is less than 100 cm"}
Publish ok
Measured distance: 214.00
Sending payload: {"Distance":213.98}
Publish ok
Measured distance: 213.95
Sending payload: {"Distance":213.98}
Publish ok
```



```
Measured distance: 68.99
Sending payload: {"Distance":68.99,"Alert Message":"The distance is less than 100 cm"}
Publish ok
Measured distance: 68.99
Sending payload: {"Distance":68.99,"Alert Message":"The distance is less than 100 cm"}
Publish ok
Measured distance: 68.99
Sending payload: {"Distance":68.95,"Alert Message":"The distance is less than 100 cm"}
Publish ok
Measured distance: 68.99
Sending payload: {"Distance":68.99,"Alert Message":"The distance is less than 100 cm"}
Publish ok
```