

ASSIGNMENT-4

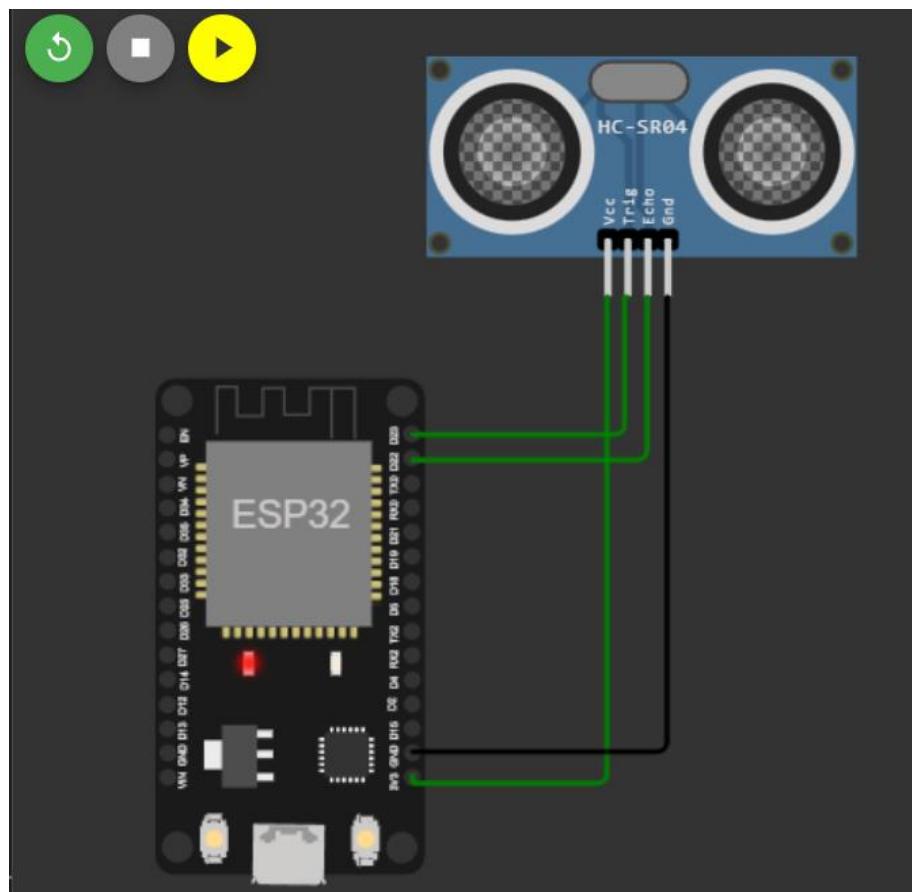
WOKWI PROGRAMMING

ASSIGNMENT DATE:	20-10-2022
STUDENT NAME:	NEHA D
STUDENT ROLL NUMBER:	2019504555
MAXIMUM MARKS:	

QUESTION:

Write code and connections in wokwi for ultrasonic sensor. Whenever distance is less than 100 cms send "alert" to IBM cloud and display in device recent events. Upload document with wokwi share link and images of IBM cloud.

CONNECTIONS:



WOKWI LINK:

<https://wokwi.com/projects/345955152667083348>

SOLUTION:

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include <HCSR04.h>
HCSR04 hc(23, 22);

#define TRIG_PIN 23 // ESP32 pin GIOP23 connected to Ultrasonic Sensor's
TRIG pin

#define ECHO_PIN 22 // ESP32 pin GIOP22 connected to Ultrasonic Sensor's
ECHO pin

float distance_cm;

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

//-----credentials of IBM Accounts-----
#define ORG "epfefm"//IBM ORGANITION ID
#define DEVICE_TYPE "abcd"//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "123"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678" //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 wifilegent  
PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined  
client id by passing parameter like server id, port and wificredential  
void setup()  
{  
    // begin serial port  
    Serial.begin (9600);  
    // configure the trigger pin to output mode  
    pinMode(TRIG_PIN, OUTPUT);  
    // configure the echo pin to input mode  
    pinMode(ECHO_PIN, INPUT);  
    delay(10);  
    Serial.println();  
    wificonnect();  
    mqttconnect();  
}  
void loop()  
{  
    // generate 10-microsecond pulse to TRIG pin  
    digitalWrite(TRIG_PIN, HIGH);  
    delayMicroseconds(10);
```

```
digitalWrite(TRIG_PIN, LOW);
distance_cm=hc.dist();

// print the value to Serial Monitor
Serial.print("Distance: ");
Serial.print(distance_cm);
Serial.println(" cm");

PublishData(distance_cm);

if (!client.loop()) {
    mqttconnect();
}

delay(5000);

}

/*
.....retrieving to Cloud....*/
void PublishData(float dis) {
    mqttconnect();//function call for connecting to ibm
    /*
        creating the String in in form JSon to update the data to ibm cloud
    */
    String payload = "{\"Distance\":";
    payload += dis;
    payload += "}";
    Serial.print("Sending payload: ");
    Serial.println(payload);
    if(dis<100)
```

```
{  
    Serial.println("ALERT DISTANCE LESS THAN 100 CM");  
}  
  
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];
    }
}
```

```
}

Serial.println("data: "+ data3);

}
```

OUTPUT:



```
00:17.866 102%  
Connecting to ...  
WiFi connected  
IP address:  
10.10.0.2  
Reconnecting client to epfefm.messaging.internetofthings.ibmcloud.com  
iot-2/cmd/test/fmt/String  
subscribe to cmd OK  
  
Distance: 332.98 cm  
Sending payload: {"Distance":332.98}  
Publish ok  
Distance: 120.99 cm  
Sending payload: {"Distance":120.99}  
Publish ok  
Distance: 36.99 cm  
Sending payload: {"Distance":36.99}  
ALERT DISTANCE LESS THAN 100 CM  
Publish ok  
Distance: 380.97 cm  
Sending payload: {"Distance":380.97}  
Publish ok
```

IBM CLOUD:

The screenshot shows the IBM Cloud Device View interface. At the top, there's a header bar with a dropdown arrow, a square icon, the ID '123', a green dot indicating 'Connected', the string 'abcd', the word 'Device', and the timestamp 'Oct 18, 2022 8:45 PM'. Below the header is a navigation bar with tabs: 'Identity', 'Device Information', 'Recent Events' (which is underlined, indicating it's the active tab), 'State', and 'Logs'. A descriptive message below the tabs states: 'The recent events listed show the live stream of data that is coming and going from this device.' The main content area displays a table of recent events:

Event	Value	Format	Last Received
Data	{"Distance":380.97}	json	a few seconds ago
Data	{"Distance":36.99}	json	a few seconds ago
Data	{"Distance":120.99}	json	a few seconds ago
Data	{"Distance":332.98}	json	a few seconds ago