ASSIGNMENT 4

**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.

# Code:

#include <WiFi.h>

#include <PubSubClient.h>

#define TRIGGER 2

#define ECHO 15

#define sound\_speed 0.034

int distance;

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

//-------credentials of IBM Accounts------

#define ORG "xk3vsk"

#define DEVICE\_TYPE "esp32"

#define DEVICE\_ID "12345"

#define TOKEN "12345678"

String data3;

//-------- Customise the above values --------

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);

pinMode(TRIGGER, OUTPUT);

pinMode(ECHO, INPUT);

delay(10);

**Serial**.println();

wificonnect();

mqttconnect();

}

void loop()

{

digitalWrite(TRIGGER, HIGH);

delayMicroseconds(10);

digitalWrite(TRIGGER, LOW);

int duration=pulseIn(ECHO,HIGH);

distance=(duration\*sound\_speed)/2;

**Serial**.print(distance);

**Serial**.println(" cms.");

if(distance<100){

PublishData(distance);

}

delay(1000);

if (!client.loop()) {

mqttconnect();

}

}

/\*.....................................retrieving to Cloud...............................\*/

void PublishData(int d) {

mqttconnect();

String payload = "{\"message\":\"alert\"";

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": "19EC078- PRAVEEN V",

"editor": "wokwi",

"parts": [

{ "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": -7.33, "left": -95.33, "attrs": {} },

{ "type": "wokwi-hc-sr04", "id": "ultrasonic1", "top": -40.71, "left": 79.84, "attrs": {} }

],

"connections": [

[ "esp:TX0", "$serialMonitor:RX", "", [] ],

[ "esp:RX0", "$serialMonitor:TX", "", [] ],

[ "ultrasonic1:VCC", "esp:VIN", "red", [ "v137.24", "h-271.79", "v-81.33" ] ],

[ "ultrasonic1:GND", "esp:GND.1", "black", [ "v0" ] ],

[ "ultrasonic1:TRIG", "esp:D5", "green", [ "v0" ] ],

[ "esp:D18", "ultrasonic1:ECHO", "green", [ "h0" ] ]

]

}

# Wokwi simulation link:

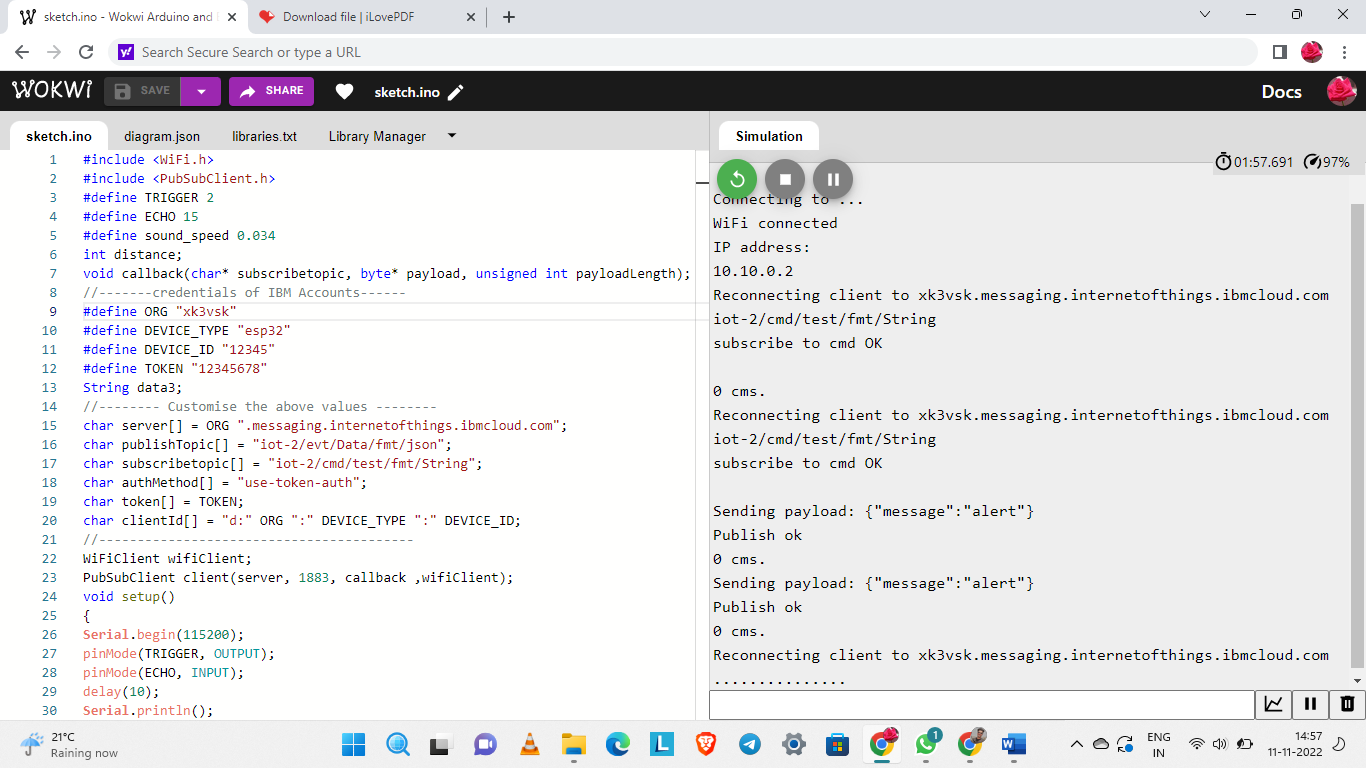
[**https:/wokwi.com/projects/348017165182435923**](https://wokwi.com/projects/348017165182435923)

# Circuit Diagram:



**Output:**

Wokwi output:



# IBM cloud output:

