

## **PROBLEM STATEMENT :**

IoT Based Smart Solution for Railways

## **DOMAIN :**

Internet of Things

## **ASSIGNMENT 4:**

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

**By,**

S.SARANKUMAR( 623519106034)

S.RANJITHKUMAR( 623519106028)

J.NAVEENKUMAR (623519106021)

T.ILAIYAKANNU (623519106010)

## **Question-1:**

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.

## **WOKWI LINK:**

<https://wokwi.com/projects/347928538175767122>

## **CODE:**

```
#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 "66kfxd"//IBM ORGANITION ID
```

```
#define DEVICE_TYPE "rasberrypi"//Device type  
mentioned in ibm watson IOT Platform
```

```
#define DEVICE_ID "ranjithkumar"//Device ID mentioned  
in ibm watson IOT Platform
```

```
#define TOKEN "9150838645" //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]);

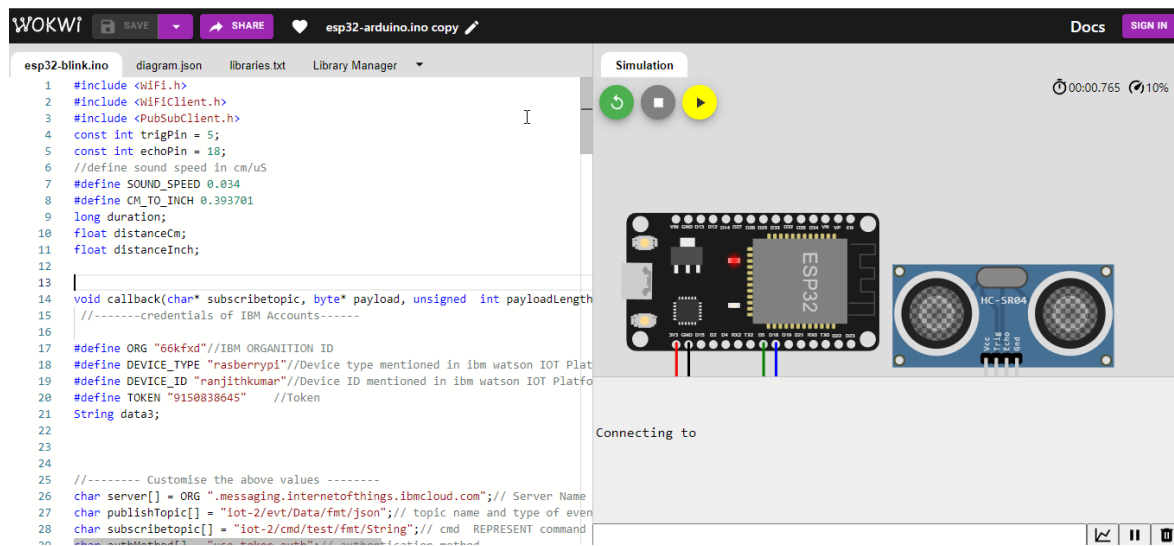
    data3 += (char)payload[i];

}

}

```

## OUTPUT:



WOKWI SAVE SHARE esp32-arduino.ino copy Docs

esp32-blink.ino diagram.json libraries.txt Library Manager

```
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 "66kfxd"//IBM ORGANIZATION ID
18 #define DEVICE_TYPE "rasberry"//Device type mentioned in ibm watson IOT Plat
19 #define DEVICE_ID "ranjithkumar"//Device ID mentioned in ibm watson IOT Platf
20 #define TOKEN "9150838645" //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 even
28 char subscribetopic[] = "iot-2/cmd/test/fmt/String"; // cmd REPRESENT command
29 char authMethod[] = "use-token-auth"; // authentication method
```

Simulation

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

00:40.702

**Watson iot connected:**

IBM Watson IoT Platform 623519106028@smartinternz.com ID: 66kfxd

Browse Action Device Types Interfaces Add Device +

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
Data	{"Distance (cm)":216.97}	json	a few seconds ago
Data	{"Distance (cm)":216.94}	json	a few seconds ago
Data	{"Distance (cm)":216.94}	json	a few seconds ago
Data	{"Distance (cm)":216.97}	json	a few seconds ago
Data	{"Distance (cm)":216.94}	json	a few seconds ago

Items per page 50 | 1-1 of 1 item 1 of 1 page 1