

# ASSIGNMENT 4

## CODE AND CONNECTIONS FOR ULTRASONIC SENSOR IN WOKWI

### QUESTION:

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.

### Code:

```
#include <WiFi.h>
#include <PubSubClient.h> void callback(char* subscribetopic,
byte* payload, unsigned int payloadLength);
//-----credentials of IBM Accounts-----
#define ORG "9f89z0"//IBM ORGANITION ID

#define DEVICE_TYPE "esp32"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "123098"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "1234509876" //Token
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);
const int trigPin = 5; const int echoPin = 18; #define
SOUND_SPEED 0.034
long duration; float
distance; void setup() {
Serial.begin(115200);
pinMode(trigPin, OUTPUT);
```

```

pinMode(echoPin, INPUT);
wificonnect();
mqttconnect();
} void loop() {
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW); duration
= pulseIn(echoPin, HIGH); distance =
duration * SOUND_SPEED/2;
Serial.print("Distance (cm): ");
Serial.println(distance);
if(distance<100)
{
Serial.println("ALERT!!");
delay(1000);
PublishData(distance);
delay(1000); if
(!client.loop()) {
mqttconnect();
} }
delay(1000);
}
void PublishData(float dist) {
mqttconnect();
String payload = "{\"Distance\":"; payload += dist;
payload += ", \"ALERT!!\":\"\"Distance less than 100cms\"";
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": "Anonymous maker",
  "editor": "wokwi",
  "parts": [
    { "type": "wokwi-esp32-devkit-v1", "id": "esp", "top": 8.67, "left":
115.33, "attrs": {} },
    { "type": "wokwi-hc-sr04", "id": "ultrasonic1", "top": -36.7, "left":
67.17, "attrs": {} }
  ],
  "connections": [

```

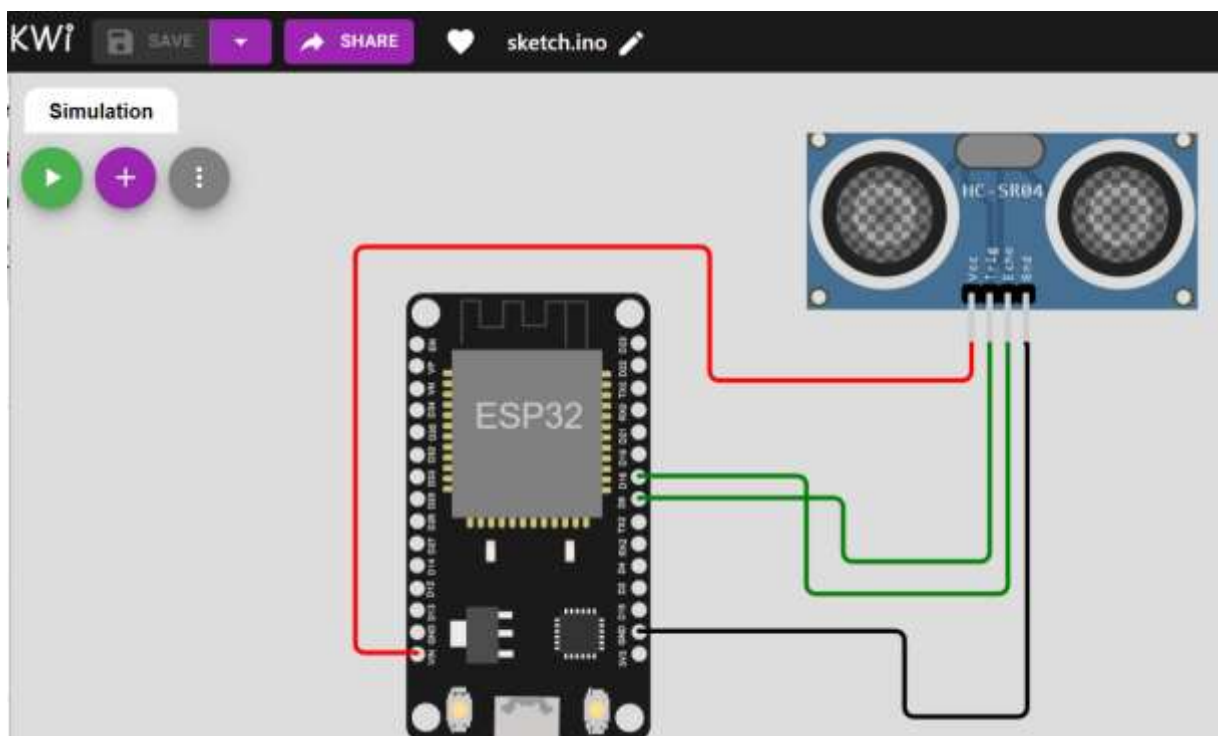
```

[ "esp:TX0", "$serialMonitor:RX", "", [] ],
[ "esp:RX0", "$serialMonitor:TX", "", [] ],
[ "esp:GND.1", "ultrasonic1:GND", "black", [ "h39.87", "v44.04", "h170" ]
],
[ "esp:D18", "ultrasonic1:ECHO", "green", [ "h77.87", "v80.01", "h110" ] ],
[ "esp:D5", "ultrasonic1:TRIG", "green", [ "h54.54", "v85.07", "h130.67" ]
],
[
  [
    "ultrasonic1:VCC",
    "esp:VIN",
    "red",
    [ "v15.24", "h-134.45", "v-80", "h-151.33", "v173.33" ]
  ]
]
]
}

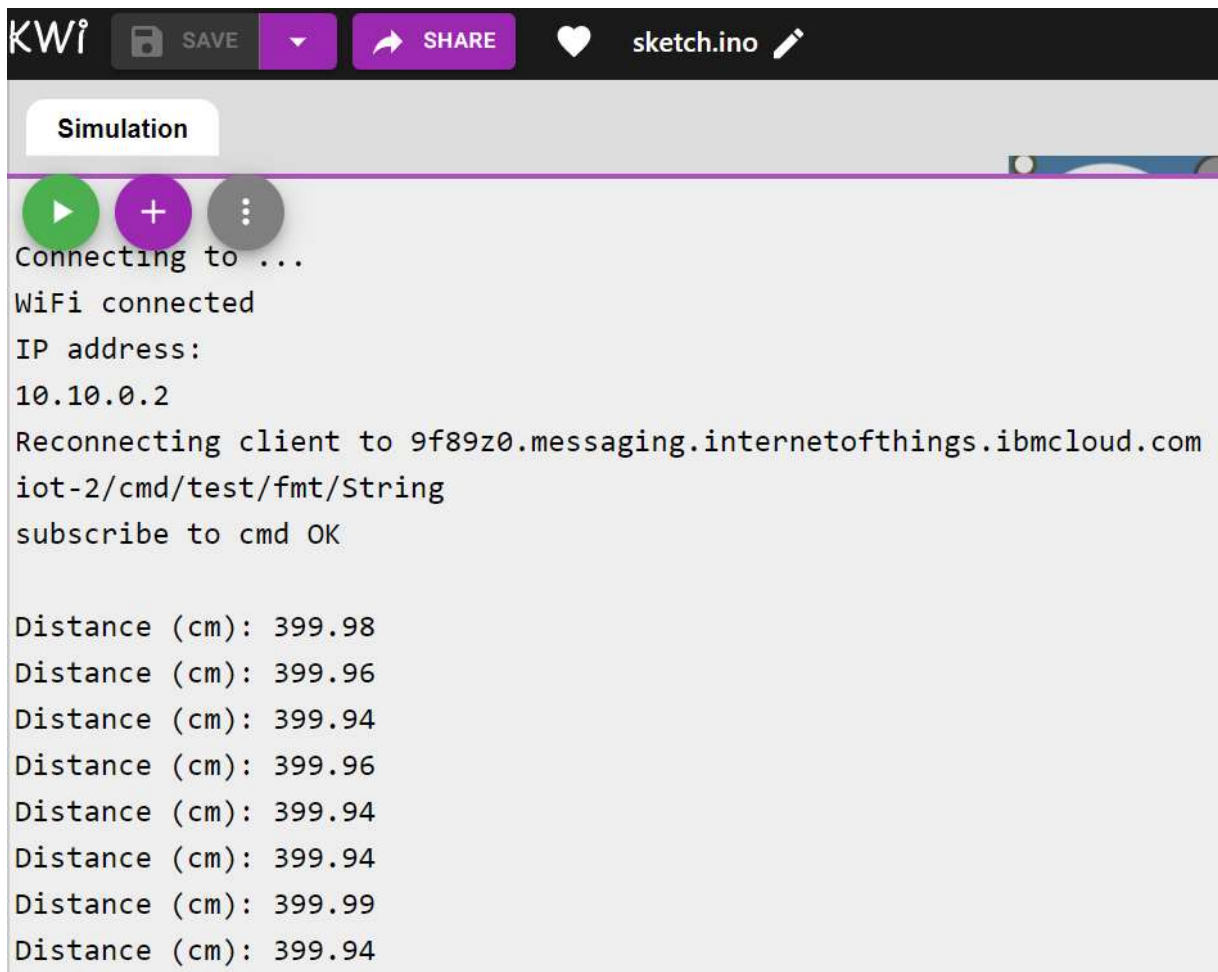
```

Wokwi link: <https://wokwi.com/projects/347208525764624979>

## Circuit



## Output

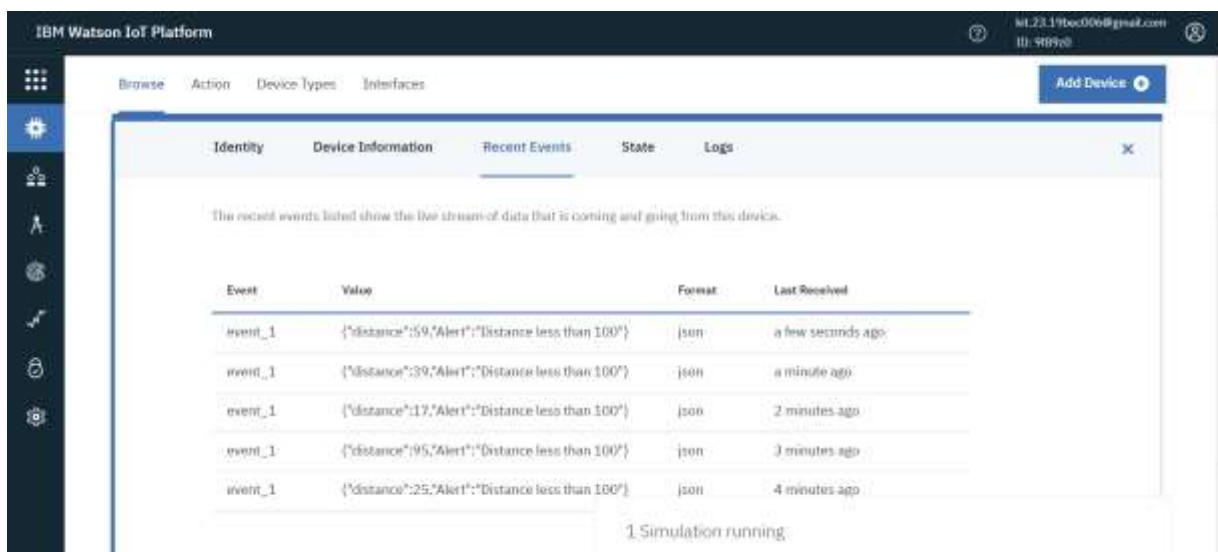


The screenshot shows the Arduino IDE interface with the Serial Monitor open. The title bar includes 'KW', 'SAVE', 'SHARE', and 'sketch.ino'. The Serial Monitor is set to 'Simulation' mode. The output text shows the device connecting to WiFi, receiving an IP address of 10.10.0.2, and connecting to an IBM Cloud IoT endpoint. It then reports a series of distance measurements in centimeters.

```
KW SAVE SHARE sketch.ino
Simulation
Connecting to ...
WiFi connected
IP address:
10.10.0.2
Reconnecting client to 9f89z0.messaging.internetofthings.ibmcloud.com
iot-2/cmd/test/fmt/String
subscribe to cmd OK

Distance (cm): 399.98
Distance (cm): 399.96
Distance (cm): 399.94
Distance (cm): 399.96
Distance (cm): 399.94
Distance (cm): 399.94
Distance (cm): 399.99
Distance (cm): 399.94
```

## IBM Cloud Output



The screenshot shows the IBM Watson IoT Platform interface. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. The 'Recent Events' tab is selected, showing a table of events. The table has columns for 'Event', 'Value', 'Format', and 'Last Received'. The events are listed in descending order of time, with the most recent event at the top. The 'Value' column contains JSON objects representing the event data.

Event	Value	Format	Last Received
event_1	{"distance":59,"Alert":"Distance less than 100"}	json	a few seconds ago
event_1	{"distance":39,"Alert":"Distance less than 100"}	json	a minute ago
event_1	{"distance":17,"Alert":"Distance less than 100"}	json	2 minutes ago
event_1	{"distance":95,"Alert":"Distance less than 100"}	json	3 minutes ago
event_1	{"distance":25,"Alert":"Distance less than 100"}	json	4 minutes ago

1 Simulation running