

ASSIGNMENT-4

Date	16 October 2022
Team ID	PNT2022TMID46033
Project Name	IoT Based Safety Gadget for Child Safety Monitoring & Notification
Maximum Marks	4 Marks

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.

```
/*
 * Ultrasonic.cpp
 *
 * Library for Ultrasonic Ranging Module in a minimalist way
 *
 */

#if ARDUINO >= 100
  #include <Arduino.h>
#else
  #include <WProgram.h>
#endif

#include "Ultrasonic.h"

Ultrasonic::Ultrasonic(uint8_t trigPin, uint8_t echoPin, unsigned long timeOut) {
  trig = trigPin;
  echo = echoPin;
  threePins = trig == echo ? true : false;
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  timeout = timeOut;
}

unsigned int Ultrasonic::timing() {
  if (threePins)
    pinMode(trig, OUTPUT);

  digitalWrite(trig, LOW);
  delayMicroseconds(2);
  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);
```

```

        if (threePins)
            pinMode(trig, INPUT);

        previousMicros = micros();
        while(!digitalRead(echo) && (micros() - previousMicros) <= timeout); // wait for
the echo pin HIGH or timeout
        previousMicros = micros();
        while(digitalRead(echo) && (micros() - previousMicros) <= timeout); // wait for
the echo pin LOW or timeout

        return micros() - previousMicros; // duration
    }

    /*
    * If the unit of measure is not passed as a parameter,
    * sby default, it will return the distance in centimeters.
    * To change the default, replace CM by INC.
    */
    unsigned int Ultrasonic::read(uint8_t und) {
        return timing() / und / 2; //distance by divisor
    }

    /*
    * This method is too verbal, so, it's deprecated.
    * Use read() instead.
    */
    unsigned int Ultrasonic::distanceRead(uint8_t und) {
        return read(und);
    }
}

/*
* Ultrasonic.h
*
* Library for Ultrasonic Ranging Module in a minimalist way
*
*/

#ifndef Ultrasonic_h
#define Ultrasonic_h

/*
* Values of divisors
*/
#define CM 28
#define INC 71

```

```

class Ultrasonic {
public:
    Ultrasonic(uint8_t sigPin) : Ultrasonic(sigPin, sigPin) {};
    Ultrasonic(uint8_t trigPin, uint8_t echoPin, unsigned long timeOut = 20000UL);
    unsigned int read(uint8_t und = CM);
    unsigned int distanceRead(uint8_t und = CM) __attribute__((deprecated ("This
method is deprecated, use read() instead.")));
    void setTimeout(unsigned long timeOut) {timeout = timeOut;}
    void setMaxDistance(unsigned long dist) {timeout = dist*CM*2;}

private:
    uint8_t trig;
    uint8_t echo;
    boolean threePins = false;
    unsigned long previousMicros;
    unsigned long timeout;
    unsigned int timing();
};

#endif // Ultrasonic_h
{
    "version": 1,
    "author": "Rozen Berg",
    "editor": "wokwi",
    "parts": [
        {
            "type": "wokwi-arduino-uno",
            "id": "uno",
            "top": 259.31,
            "left": 31.06,
            "rotate": 0,
            "hide": false,
            "attrs": {}
        },
        {
            "type": "wokwi-hc-sr04",
            "id": "ultrasonic",
            "top": 86.99,
            "left": 109.89,
            "rotate": 0,
            "hide": false,
            "attrs": { "distance": "100" }
        }
    ]
}

```

```

    ],
    "connections": [
      [ "uno:GND.1", "ultrasonic:GND", "black", [ "v-8", "*", "v8" ] ],
      [ "uno:13", "ultrasonic:ECHO", "green", [ ] ],
      [ "uno:12", "ultrasonic:TRIG", "purple", [ "*", "v4" ] ],
      [ "uno:5V", "ultrasonic:VCC", "red", [ "v16", "h-96", "*", "v12" ] ]
    ]
  }
}
/*

```

Ultrasonic Simple

Prints the distance read by an ultrasonic sensor in centimeters. They are supported to four pins ultrasound sensors (like HC-SC04) and three pins (like PING))) and Seeed Studio sensors).

The circuit:

* * Module HC-SC04 (four pins) or PING))) (and other with three pins), attached to digital pins as follows:

```

-----
| HC-SC04 | Arduino |   3 pins | Arduino |
-----
| Vcc  | 5V  | | Vcc  | 5V  |
| Trig | 12  | OR | SIG | 13  |
| Echo | 13  | | Gnd  | GND  |
| Gnd  | GND | -----
-----

```

*/

```
#include "Ultrasonic.h"
```

/*

Pass as a parameter the trigger and echo pin, respectively, or only the signal pin (for sensors 3 pins), like:

```
Ultrasonic ultrasonic(13);
```

*/

```
Ultrasonic ultrasonic(12, 13);
int distance;
```

```
void setup() {
  Serial.begin(9600);
}
```

```
void loop() {
```

```
// Pass INC as a parameter to get the distance in inches
```

```
distance = ultrasonic.read(CM);
```

```
Serial.print("Distance in CM: ");  
Serial.println(distance);
```

```
distance = ultrasonic.read(INC);
```

```
Serial.print("Distance in Inches: ");  
Serial.println(distance);
```

```
delay(1000);
```

```
}
```

Link: <https://woki.com/projects/346782026684170836>

The screenshot displays the WOKWI IDE interface. On the left, the 'sketch.ino' file is open, showing a C++ sketch for an ESP32 connected to an HC-SR04 ultrasonic sensor. The sketch includes libraries for WiFi and MQTT, defines pins for Trig (15) and Echo (4), and sets a minimum distance of 100. It includes a callback function for MQTT messages and a main loop that reads the distance in centimeters and inches, prints it to the serial monitor, and delays for 1000ms. On the right, the 'Simulation' window shows a 3D model of the ESP32 and the HC-SR04 sensor connected by wires. Below the IDE, a screenshot of the Woki IoT platform shows the device 'assign_4' with a 'Recent Events' tab. The events table shows a stream of 'MESSAGE:ALERT' data points.

```
1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3 #define TrigPIN 15
4 #define EchoPIN 4
5 #define MINDIST 100
6
7
8
9 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
10
11 //-----credentials of IBM Accounts-----
12
13 #define ORG "phsvnq" //IBM ORGANITION ID
14 #define DEVICE_TYPE "assign" //Device type mentioned in ibm watson IOT Platform
15 #define DEVICE_ID "assign_4" //Device ID mentioned in ibm watson IOT Platform
16 #define TOKEN "123456789" //Token
17 String data3;
18 float h, t;
19
20
21 //----- Customise the above values -----
22 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
23 char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of event
24 char subscribetopic[] = "iot-2/cmd/command/fmt/String"; // cmd REPRESENT comma
25 char authMethod[] = "use-token-auth"; // authentication method
26 char token[] = TOKEN;
27 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
28
29
30 //-----
```

Event	Value	Format	Last Received
Data	("MESSAGE:ALERT")	json	a few seconds ago
Data	("MESSAGE:ALERT")	json	a few seconds ago
Data	("MESSAGE:ALERT")	json	3 minutes ago
Data	("MESSAGE:ALERT")	json	3 minutes ago
Data	("MESSAGE:ALERT")	json	4 minutes ago