Turbidimetru

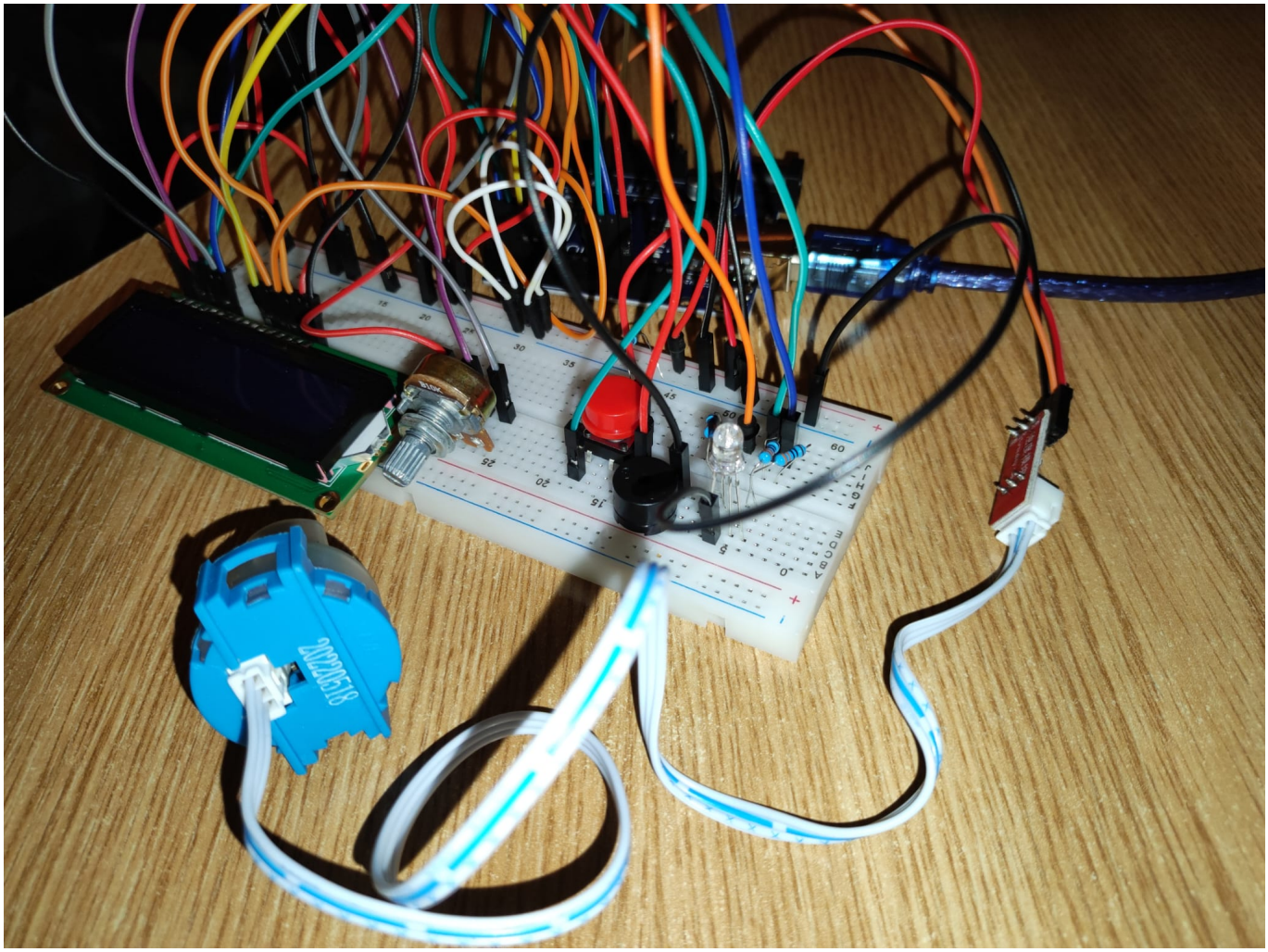
Introducere

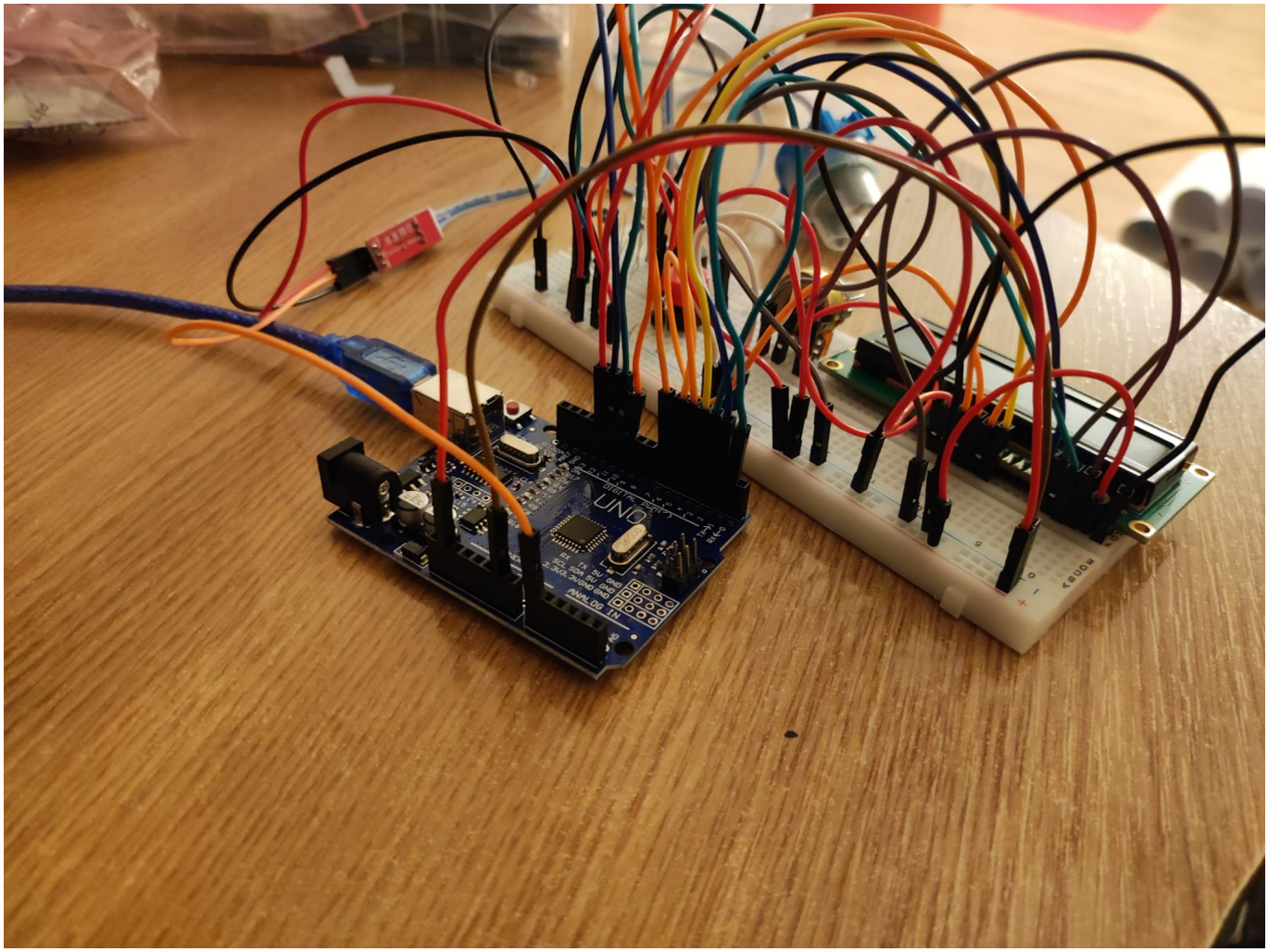
This project will measure the turbidity in the water and display on an LCD screen how pure the water is. Also, a green, yellow or red RGB LED will light up and a buzzer will sound. It is useful because it can detect if the water from the tap at home is safe to drink or it can detect how pure the water in a lake is.

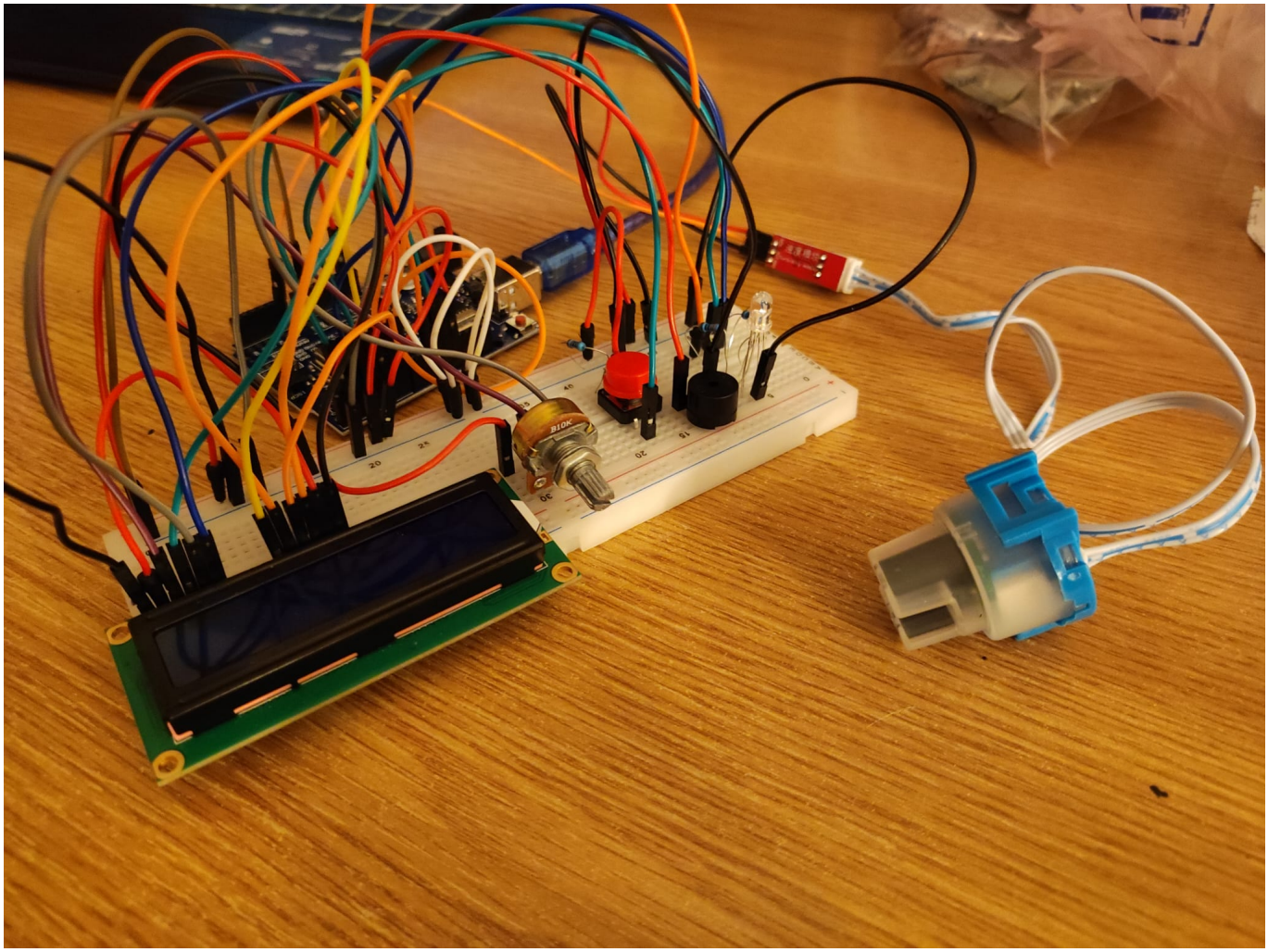
General Description

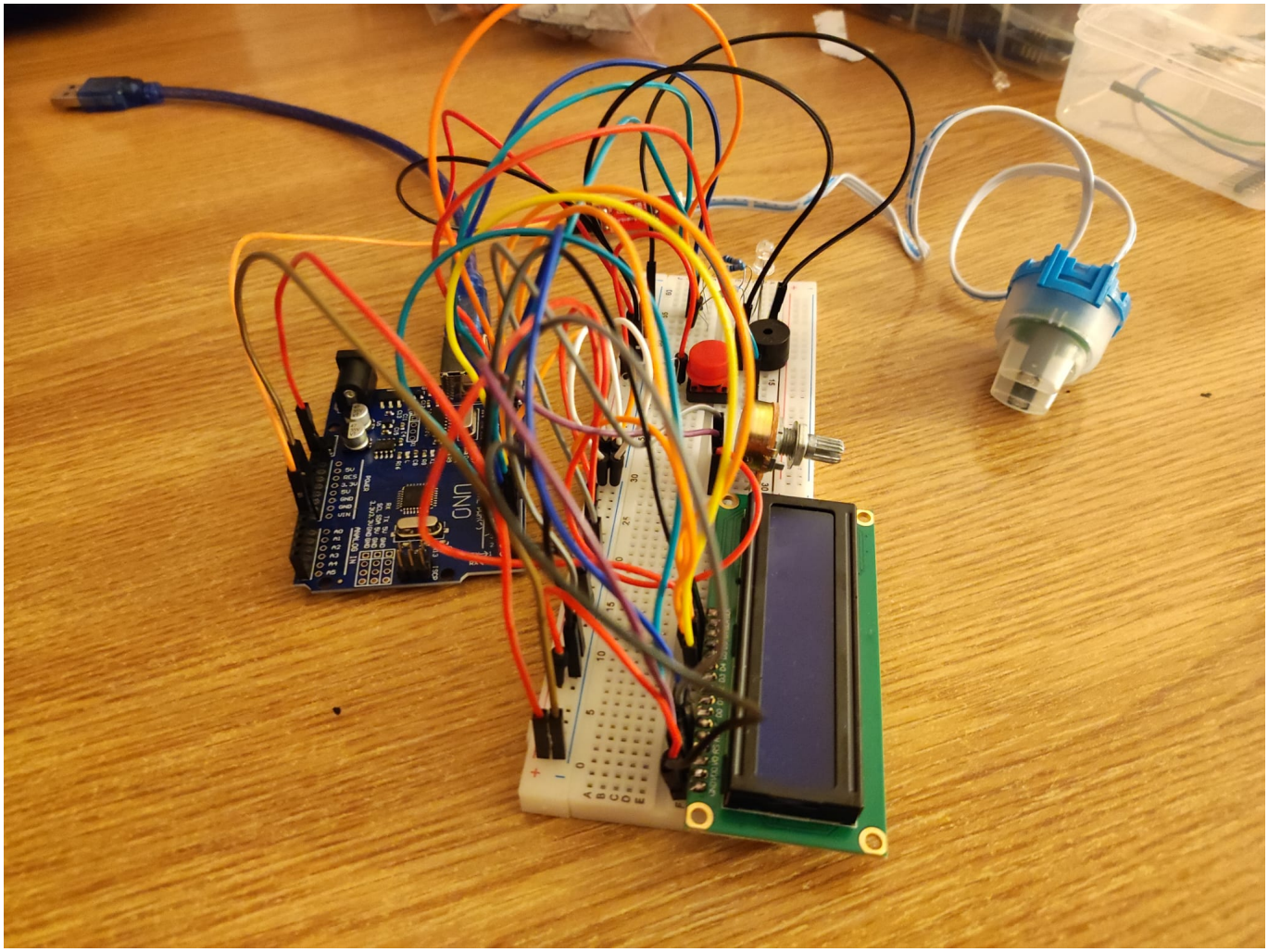
I will connect the turbidimeter to the Arduino and take the values that will be processed and transformed into a percentage from 0 to 100. Based on this percentage, the RGB led, the buzzer will light up and this percentage will be shown on the LCD.

Also, here are some pictures with the completed project:




UNO


B10K



Hardware Design

List of components:

- Arduino UNO R3
The Arduino IDE (Integrated Development Environment), a software tool made to make it simple to write, upload, and debug code for the board, can be used to program the Arduino UNO, which can be used to control a variety of electronic devices, including sensors, motors, lights, and other components.

- LCD 1602
A 16×2 LCD display module (LCD 1602) is a device that can be used to display alphanumeric characters, symbols, and even simple graphics. It has a display size of 16 columns and 2 rows.

- Passive Buzzer
A passive buzzer is a type of electronic component that is used to generate sound in electronic circuits. It is called passive because it does not have an integrated oscillator circuit and requires an external signal to produce a sound. The buzzer consists of a piezoelectric element that vibrates when an electric signal is applied to it, creating a sound wave.

- Turbidity Module (+Turbidity Sensor)
A turbidity module is a sensor used to measure the level of turbidity, or cloudiness, in a liquid. It

works by shining a light through the liquid and measuring the amount of light that is scattered by particles in the liquid. The more particles there are, the more the light will scatter, and the higher the turbidity measurement will be.

-RGB LED
Three distinct LED chips—one each for red, green, and blue—are housed inside an RGB LED. A variety of colors can be created by independently altering the brightness of each of these hues.

Other components:
- USB Cable
- 10k and 330 ohm resistors
- Potentiometer
- Wires
- Button

Software Design

I made this project in Arduino IDE.
Extra libraries: LiquidCrystal.
Sources 3-rd party: I also used Tinkercad to develop the project and to test various software/hardware segments.

Several variables are also used. A debugging mode that is normally commented out is also included. The debugging mode will transform the values of the turbidity module to values that steadily increase. This should normally only be used for developing or testing the project.

I start off the software by defining all the variables, including the library and setting up the components. In the loop section of the code i have the Buzzer playing a melody for the first loop and showing a short welcome on the LCD. At the very start of the loop we can also see the Arduino reading the turbidity module's values and transforming them to values from 0 to 100. Afterwards, we test the turbidity. If the turbidity is 20 or lower, the RGB led turns red and the buzzer sings 3 lower tones. If the turbidity is 50 or lower, the RGB led turns yellow and the buzzer sings another 3 tones. If the turbidity is higher than 50, the RGB led turns green and the buzzer sings 3 other corresponding tones. During this, the LCD also shows the procentage of turbidity and a short message saying wheter it's dirty, cloudy or clean. The loop will reset every 3 seconds. I have set a variable to stop repeating the same buzzing if the value is in the same category as the last loop as it can get annoying fast.

```
void loop() {
int sensorValue = analogRead(A0);
int turbidity = map(sensorValue, 0,640, 100, 0);
if(turbidity < 0) turbidity = 0;
//for debugging only:
//if(i == 0) turbidity = 30;
//if(i != 0) {turbidity = debug_int*10; debug_int++;}
//if(i != 0 && debug_int>=10){turbidity = 0; debug_int=0;}
//end of debugging section
```

```
  if(i==0){

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Stan Alex's");
  tone(13, 261, 200); //Middle C
  setColor(luminozitate_led, luminozitate_led, luminozitate_led);
  delay(200);
  tone(13, 277, 200); //C#
  delay(200);
  tone(13, 294, 200); //D
  delay(200);
  tone(13, 311, 200); //D#
  delay(200);
  setColor(luminozitate_led, luminozitate_led, 0);
  tone(13, 370, 200); //F#
  delay(400);
  setColor(0, luminozitate_led, luminozitate_led);
  tone(13, 370, 200); //F#
  delay(400);

  lcd.clear();
  lcd.setCursor(0, 1);
  lcd.print("Turbidity Meter");
  setColor(0, luminozitate_led, 0);
  tone(13, 349, 200); //F
  delay(200);
  tone(13, 311, 200); //D#
  delay(200);
  tone(13, 294, 200); //D
  delay(200);
  setColor(0, 0, luminozitate_led);
  tone(13, 370, 200); //F#
  delay(400);
  setColor(luminozitate_led, 0, 0);
  tone(13, 370, 200); //F#
  delay(400);
  setColor(0, 0, 0);
  i=1;
  }
  delay(3000);

  if(turbidity < 21){
    setColor(luminozitate_led, 0, 0);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("It's dirty");
    lcd.setCursor(0, 1);
    lcd.print("Procent: "); lcd.print(turbidity);lcd.print(" %");

    //red
```

```
      if(buzz==1 && stop_repetition != 40){
    tone(13, 330, 200); //E
    delay(200);
    tone(13, 294, 200); //D
    delay(200);
    tone(13, 261, 200); //Middle C
    delay(200);
    stop_repetition = 40;
    }

  } else if(turbidity < 51){
  setColor(luminozitate_led, luminozitate_led, 0);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("It's cloudy");
  lcd.setCursor(0, 1);
  lcd.print("Procent: "); lcd.print(turbidity);lcd.print(" %");

  //yellow:
        if(buzz==1 && stop_repetition != 20){
    setColor(luminozitate_led, luminozitate_led, 0);
    tone(13, 349, 200); //F
    delay(200);
    tone(13, 311, 200); //D#
    delay(250);
    tone(13, 311, 200); //D#
    delay(150);

    stop_repetition = 20;
    }
  setColor(luminozitate_led, luminozitate_led, 0);
  } else{
  setColor(0, luminozitate_led, 0);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("It's clean");
  lcd.setCursor(0, 1);
  lcd.print("Procent: "); lcd.print(turbidity);lcd.print(" %");
  setColor(0, luminozitate_led, 0);
  //green:
      if(buzz==1 && stop_repetition != 30){
    setColor(0, luminozitate_led, 0);
    tone(13, 392, 200); //G
    delay(200);
    tone(13, 370, 200); //F#
    delay(200);
          tone(13, 392, 200); //G
    delay(200);

    stop_repetition = 30;
    }
```

```
      setColor(0, luminozitate_led, 0);
    }

}
```

We will now talk about the functions used:
I used a function to set the color of the RGB led to whatever is received through 3 parameters, the Red Value, the Green Value and the Blue Value. This function then sets the intensity of each pin accordingly.

```
void setColor(int redValue, int greenValue, int blueValue) {
  analogWrite(led_Rosu, 255-redValue);
  analogWrite(led_Verde, 255-greenValue);
  analogWrite(led_Albastru, 255-blueValue);
}
```

I used another function to interrupt the normal ongoing processes of the loop and set a variable used to activate/deactivate the buzzer when the button is pressed.

```
void ISR_button(){
  if(buzz == 1){
        buzz =0;
  }
  else{buzz = 1;}
}
```

Obtained Results

After completing this project, i managed to create a functional turbidity meter with a bunch of add-ons. I have also understood a lot better how all the components in my project can work together to achieve something great. I have managed to create an algorithm that will take all these separate components and make them work perfectly together.

Concluzii

We can measure the purity of water with a nice interface and a friendly environment

Download

turbidity_meter_v2.zip

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

Listă cu documente, datasheet-uri, resurse Internet folosite, eventual grupate pe **Resurse Software** şi **Resurse Hardware**.

Export to PDF