Senzor de proximitate
cu afișaj digital și alertă sonoră

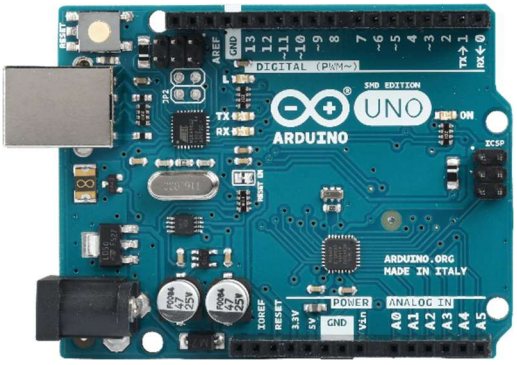
Bargaoanu Bogdan Alexandru

Afilierea: Facultatea de Automatica si Calculatoare, Romana, Anul 2, Specializare Automatica si Informatica Aplicata, Grupa 30125, Universitatea Tehnica din Cluj-Napoca , Strada George Baritiu , Nr. 26-28 , 400027 , Cluj-Napoca , Romania.

e-mail: Bargaoanu.Re.Bogdan@student.utcluj.ro

Principala idee a proiectului este folosirea în condiții de mediu nefavorabile , iar ca pentru acest lucru sa fie realizabil, s-a folosit un senzor de proximitate bazat pe tehnologia ultrasunetelor al carui avantaj este insensibilitatea la lumina,praf,fum,vapori,puf si aburi. Domeniul de aplicare este industria automobilistică avand o eroare de ~0.035cm/cm (conform producatorului).

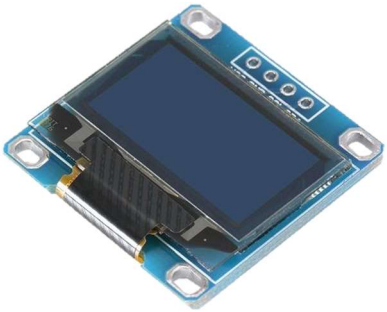
1. **Componente folosite in realizare**
   - Placa de dezvoltare Arduino Uno



   - Buzzer MH-FMD

   - HC-SR04 – Senzor de distanta cu ultrasunete

   - Ecran OLED SSD1306



Observatie: Viteza sunetului: 343m/s (in cazul nostru 0,343cm/µs).

- 3 LED (1 rosu, 1 portocaliu, 1 verde)

- 3 rezistente 220Ω

2. **Rezumat functionare**

Cu ajutorul unui semnal PWM se citeste periodic senzorul de proximitate, convertindu-se durata de timp pe care o parcurge sunetul pana la intoarcere in distanta cu ajutorul formulei:

$$distanta = \frac{durata}{2} \times 0{,}343 \text{ cm}$$

Conform producătorului HC-SR04 poate măsură distanta fară o eroare semnificativa până la aproximativ 450 cm, așa că au fost definite 3 cazuri. Primul caz îl reprezintă o distanta care aparține intervalului [x+450,50) oricare ar fi x, LED-ul verde și buzzerul fiind pornite pentru 200ms respectiv oprite pentru 800ms. În al doilea caz, distanta aparține intervalului [50,15), iar LED-ul portocaliu și buzzerul sunt pornite pentru 200ms respectiv oprite pentru 400ms. Ultimul caz este caracterizat de o distanta din intervalul [15,0] și vă porni LED-ul roșu și buzzerul pentru 200ms respectiv opri pt 100ms. Indiferent de caz, rezultatul în cm vă fi afișat pe ecranul OLED, alaturi de un dreptunghi care se vă umple direct proporțional cu distanta citita, până la pragul de 100cm, care reprezintă umplerea completa a dreptunghiului. Orice distanta mai mare de 100cm vă afișă un dreptunghi plin, acest lucru fiind realizat pentru a ușura testarea.

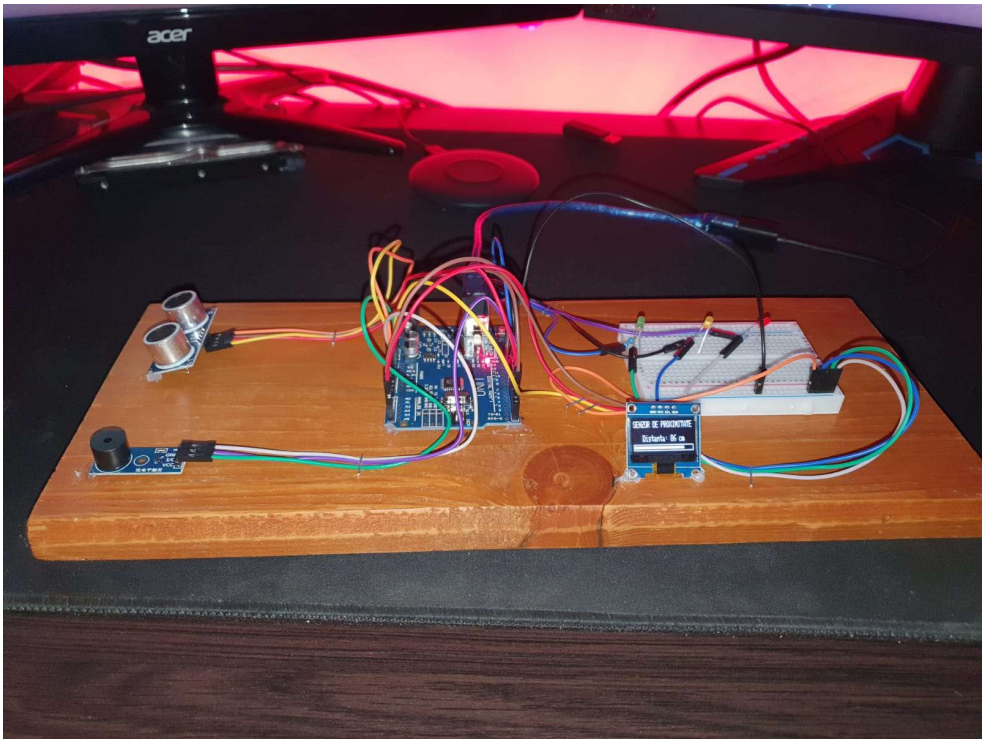


Figure 1- Testare

3. **Montaj și specificații tehnice**

- **Arduino Uno –** *VERSIUNE* – R3.
  MICROCONTROLER – Atmega328P la o frecventa de 16MHz.
  MEMORIE – 32KB flash, 2KB SRAM, 1KB EEPROM.
  COMUNICARE – UART, I2C (folosita in realizarea proiectului) , SPI.
  ALIMENTARE – USB sau conector pentru baterie de 9V.
  PINI – 6 PWM, 6 analog, 14 digital, 13 pentru led-uri incorporate.
  TENSIUNE/CURENT – 7-12V (nominal) și 20mA.
  DIMENSIUNI – lungime 68.6mm, latime 53.4mm și greutate 25g.

  **Observație:proiectul a fost realizat folosind platofrma de dezvoltare Arduino IDE , specifica pentru produsele de tip Arduino și care se bazează pe limbajul C/C++.*

- **HC-SR04** – *TRIG* – pin de ieșire pentru transmițătorul senzorului, pe care se trimite un PWM de la pinul digital 3 de pe Arduino Uno.
  ECHO – pin de intrare pentru pentru receptorul senzorului cu ajutorul căruia se citește durata transmiteri ultrasunetelor. Pin conectat la pinul digital 2 de pe Arduino Uno.
  GND – ground.
  VCC – alimentare, pin conectat la o ieșire de 5V de pe Arduino Uno.

- **Ecran OLED SSD1306** – *SCL* – Serial Clock, pin prin care se transmite semnalul de ceas, conectat la pinul analogic A5 de pe Arduino Uno.
  SDA – Serial Data, pin prin care se transmit datele pentru afișare, conectat la pinul analogic A4 de pe Arduino Uno.
  GND – ground.
  VCC – alimentare, pin conectat la o ieșire de 3.3V de pe Arduino Uno.

  **Observație: ecranul a fost configurat cu ajutorul librăriei Adafruit_GFX și Adafruit_SSD1306, specifice pentru acest model de ecran și comunicare I2C.*

- **MH-FMD** – *I/O* – pin de iesire folosit pentru pornirea si oprirea buzzerului, conectat la pinul digital 6 de pe Arduino Uno.
  GND – ground.
  VCC – alimentare, pin conectat la o ieșire de 5V de pe Arduino Uno.

- **LED-**uri – fiecare legat in serie cu o rezistenta de 220 Ω.
  ROSU – conectat pe pinul digital 8 al Arduino Uno.
  GALBEN – conectat pe pinul digital 12 al Arduino Uno.
  VERDE – conectat pe pinul digital 13 al Arduino Uno.

4. **Cod**

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define echoPin 2
#define trigPin 3
#define ledvPin 13
#define ledgPin 12
#define ledrPin 8
#define buzzerPin 6
#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);
long duration;
int distance;
void setup() {
  Wire.begin();
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(ledvPin, OUTPUT);
  pinMode(ledgPin, OUTPUT);
  pinMode(ledrPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
  Serial.begin(9600);
}
void drawPercentbar(int x, int y, int width, int height, int progress) {
  progress = progress > 100 ? 100 : progress;
  progress = progress < 0 ? 0 : progress;
  float bar = ((float)(width - 4) / 100) * progress;
  display.drawRect(x, y, width, height, WHITE);
  display.fillRect(x + 2, y + 2, bar, height - 4, WHITE);
  if (height >= 15) {
    display.setCursor((width / 2) - 3, y + 5);
    display.setTextSize(1);
    display.setTextColor(WHITE);
    if (progress >= 50)
      display.setTextColor(BLACK, WHITE); }
}
void loop() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
```

```
  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.034 / 2;
  Serial.print("Distanta: ");
  Serial.print(distance);
  Serial.println(" cm");
  if (distance > 50) {
    digitalWrite(buzzerPin, LOW);
    digitalWrite(ledvPin, HIGH);
    delay(200);
    digitalWrite(buzzerPin, HIGH);
    digitalWrite(ledrPin, LOW);
    digitalWrite(ledgPin, LOW);
    digitalWrite(ledvPin, LOW);
    delay(800);
  } else if (distance <= 50 && distance > 15) {
    digitalWrite(buzzerPin, LOW);
    digitalWrite(ledgPin, HIGH);
    delay(200);
    digitalWrite(buzzerPin, HIGH);
    digitalWrite(ledrPin, LOW);
    digitalWrite(ledgPin, LOW);
    digitalWrite(ledvPin, LOW);
    delay(400);
  } else {
    digitalWrite(buzzerPin, LOW);
    digitalWrite(ledrPin, HIGH);
    delay(200);
    digitalWrite(buzzerPin, HIGH);
    digitalWrite(ledrPin, LOW);
    digitalWrite(ledgPin, LOW);
    digitalWrite(ledvPin, LOW);
    delay(100); }
  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.setCursor(0, 0);
  display.print("SENZOR DE PROXIMITATE");
  display.setCursor(20, 15);
  display.print("Distanta: ");
  display.print(distance);
  display.print(" cm");
drawPercentbar(0, 25, 128, 7, distance);
  display.display();
}
```

5. **Bibliografie**

- https://www.arduino.cc/
- https://projecthub.arduino.cc/
- https://en.wikipedia.org/wiki/Ultrasonic_transducer
- https://docs.arduino.cc/learn/communication/wire
- https://docs.arduino.cc/hardware/uno-rev3
- https://www.arduino.cc/en/software