Parking senzor

Introducere

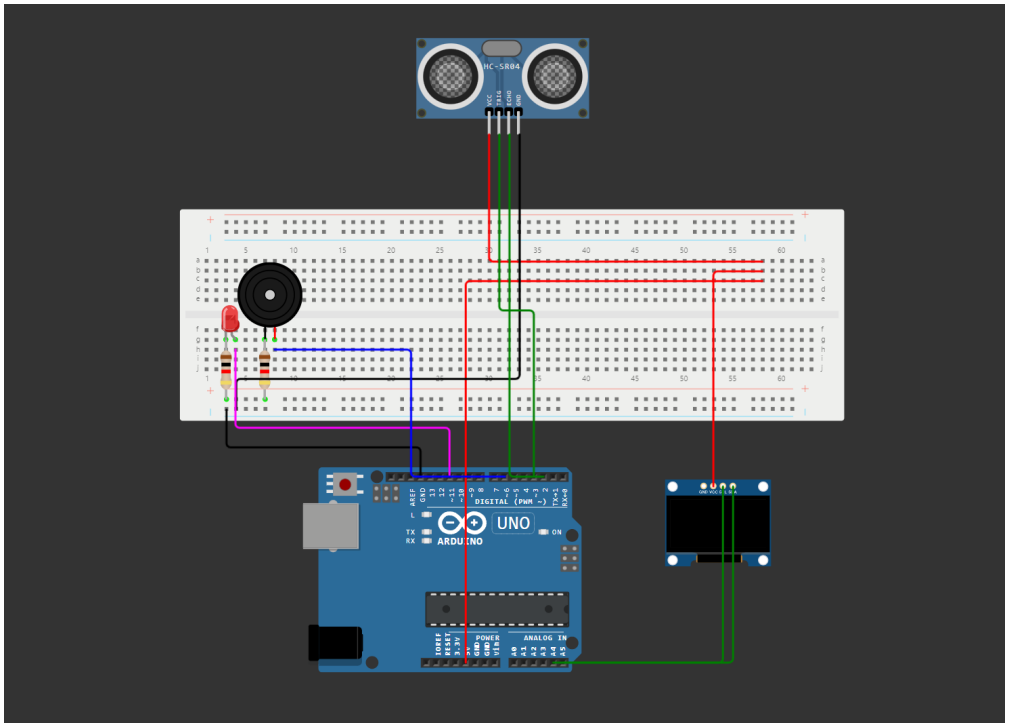
Descriere proiect:

- Senzor de parcare pentru masina
- Scopul senzorului este de a masura distanta de la un obiect la masina si de a afisa distanta
- Este util pentru persoanele care nu au senzor la masina

Descriere generală

Proiectul propus vizează dezvoltarea unui sistem de asistență pentru parcare care utilizează senzori ultrasunet pentru a determina distanța între vehicul și eventualele obstacole din apropiere. Sistemul daca in preajma senzorului se va afla un obiect acesta va transmite informatia si va afișa această distanță pe un ecran OLED, oferind astfel informații precise și ușor de citit.

Hardware Design



Arduino Uno R3

- Arduino UNO este o platforma de procesare open-source, bazata pe software si hardware flexibil si simplu de folosit ce are la baza microcontroller-ul ATmega 328.

HC-SR04

- Tensiune de operare: 5V DC
- Curent de operare: 15mA
- Unghi de masurare: 15 grade
- Distanta de variatie: 2cm - 4m

Monochrome 1.3" 128x64 OLED graphic display

- Acest bord / cip foloseste adresa I2C intre 0x3C-0x3D, selectabil cu jumperi
- Display-ul foloseste 40mA de la sursa de 3.3V.

LED

- Tip diodă LED
- Tensiune de lucru 1.8...2.4V

Buzzer activ

- Tensiune de lucru 5V

Software Design

Mediu de Dezvoltare:

Firmware-ul a fost dezvoltat folosind Arduino IDE, un mediu de dezvoltare integrat care oferă unelte și biblioteci esențiale pentru programarea și încărcarea codului pe plăcile Arduino.

Librării și Surse 3rd-Party:

Pentru acest proiect, am folosit următoarele librării standard disponibile în Arduino IDE:

“Adafruit_GFX.h”: Folosită pentru controlul ecranului OLED și afișarea informațiilor relevante. În plus, pentru a obține un control mai precis și eficient al unor componente hardware, am optat să lucrez direct cu registrele microcontroller-ului. Astfel, **senzorul ultrasonic HC-SR04** are patru pini : VCC pentru alimentare (5V), GND pentru ground, TRIG pinul de trigger pentru a inița măsurarea, ECHO pinul de echo pentru a citi durata impulsului de întoarcere:

```
DDRD |= (1 << DDD3);
DDRD &= ~(1 << DDD2);

PORTD |= (1 << PORTD3);
PORTD &= ~(1 << PORTD3);
```

Pentru led in loc de functia **pinMode(LED, OUTPUT)** din libraria Arduino am folosit:

```
DDRB |= (1 << DDB3); // configuram pinul 11 ca output
TCCR2A |= (1 << WGM20) | (1 << WGM21); //  Timer/Counter Control Register A
pentru Timer 2
TCCR2A |= (1 << COM2A1);
TCCR2A &= ~(1 << COM2A0);
TCCR2B |= (1 << CS22);
TCCR2B &= ~(1 << CS21);
TCCR2B &= ~(1 << CS20);
OCR2A = 254;
```

Pentru buzzer in loc de functia **tone(BUZZER,x)** din libraria Arduino am folosit:

```
DDRD |= (1 << DDD6);
TCCR0A |= (1 << WGM00) | (1 << WGM01) | (1 << COM0A1); // PWM
TCCR0B |= (1 << CS01);

unsigned int ocrValue = 16000000 / (2 * 8 * frequency) - 1;
// 16MHz - system clock frequency
// 8 - prescaler value
OCR0A = ocrValue;
```

Pentru afisajul senzorului pe ecranul oled am folosit functii din bibloteca arduino unde **epd_bitmap_sensor_x_on** este imaginea pentru care obiectul nu se afla in raza senzorului, iar **epd_bitmap_sensor_x_off** pentru obiectul care se afla in raza senzorului, **dist_step_x** este pragul de distanta:

```
dist_step_x = min_dist + round((max_dist - min_dist) / 4.0 * x);
```

```
display.drawBitmap(
  24,
  17,
  sensor.measured_distance_cm > dist_step_01 ? epd_bitmap_sensor_01_a_on :
epd_bitmap_sensor_01_a_off, 32, 14, 1);
display.drawBitmap(
  21,
  25,
  sensor.measured_distance_cm > dist_step_02 ? epd_bitmap_sensor_01_b_on :
epd_bitmap_sensor_01_b_off, 32, 16, 1);
display.drawBitmap(
  18,
  34,
  sensor.measured_distance_cm > dist_step_03 ? epd_bitmap_sensor_01_c_on :
epd_bitmap_sensor_01_c_off, 32, 17, 1);
display.drawBitmap(
  16,
  43,
  sensor.measured_distance_cm > dist_step_04 ? epd_bitmap_sensor_01_d_on :
epd_bitmap_sensor_01_d_off, 32, 18, 1);
```

Algoritmi și Structuri de Date:

Senzorul are urmatoarele caracteristici:

```
struct sensor_data{
 int echo_pin;
 int trig_pin;
 int measured_distance_cm;
 int label_xpos; // x position of the distance label
 int label_ypos; // y position of the distance label
```

```
 int label_width; // calculated width of the distance string
 int label_startpos_x; // start X position for the label
 int label_startpos_y; // start Y position for the label
 int label_endpos_x; // end X position for the label
 int label_endpos_y; // end Y position for the label
};
```

Principalele algoritmi implementate în firmware includ: Logica de pornire a ledului si buzzerului(ce este comentat este cu functii arduino avand aceeasi functionalitate):

```
  if(sensor.measured_distance_cm >= 2 && sensor.measured_distance_cm <= 25){
  //analogWrite(LED, 254);
  OCR2A = 254;
  tone(BUZZER,300);
  //setBuzzerFrequency(50);
  }
  else if(sensor.measured_distance_cm > 25 && sensor.measured_distance_cm <=
50) {
  //      analogWrite(LED, 127);
  OCR2A = 127;
  tone(BUZZER, 200);
  //setBuzzerFrequency(100);
  }
  else if(sensor.measured_distance_cm > 50 && sensor.measured_distance_cm <=
75) {
  //analogWrite(LED, 64);
  OCR2A = 64;
  tone(BUZZER, 100);
  //setBuzzerFrequency(200);
  }
  else {
  //analogWrite(LED, 0);
  OCR2A = 0;
  //noTone(BUZZER);
  TCCR0A &= ~((1 << COM0A1) | (1 << COM0A0));
  }
```

Actualizarea și afișarea informațiilor pe ecranul OLED, inclusiv distanța față de obstacol. Astfel, o imagine pe ecranul OLED va arata asa:

```
// 'car_image', 56x15px
const unsigned char epd_bitmap_car_image [] PROGMEM = {
 0xc0, 0x40, 0x00, 0x00, 0x00, 0x02, 0x03, 0xc0, 0x8f, 0xff, 0xff, 0xff,
0xf1, 0x03, 0xe0, 0x70,
 0x00, 0x00, 0x00, 0x0e, 0x07, 0xff, 0x00, 0xff, 0xff, 0xff, 0x00, 0xff,
0xc0, 0xff, 0x00, 0x00,
 0x00, 0xff, 0x03, 0xc0, 0x00, 0x05, 0x75, 0xc0, 0x00, 0x03, 0x47, 0xf8,
0x05, 0x51, 0x40, 0x1f,
 0xe2, 0x47, 0xfc, 0x05, 0x65, 0x80, 0x3f, 0xe2, 0x40, 0x78, 0x06, 0x45,
0x40, 0x1e, 0x02, 0x30,
 0x00, 0x00, 0x00, 0x00, 0x00, 0x0c, 0x1e, 0x01, 0xff, 0xff, 0xff, 0x80,
```

```
0x78, 0x07, 0xe2, 0x00,
 0x00, 0x00, 0x47, 0xe0, 0x00, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00, 0x00,
0xbf, 0xff, 0xff, 0xff,
 0xfd, 0x00, 0x00, 0x60, 0x00, 0x00, 0x00, 0x06, 0x00
};
```

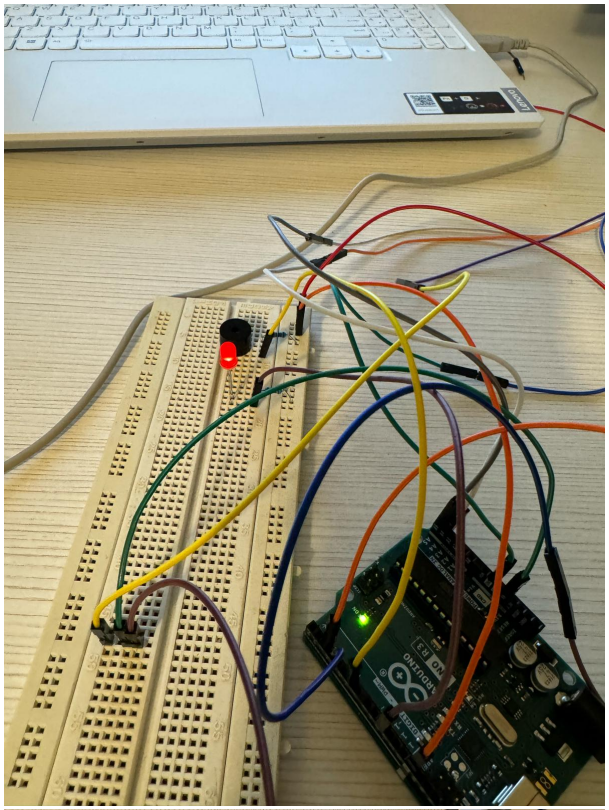
Surse și Funcții Implementate:

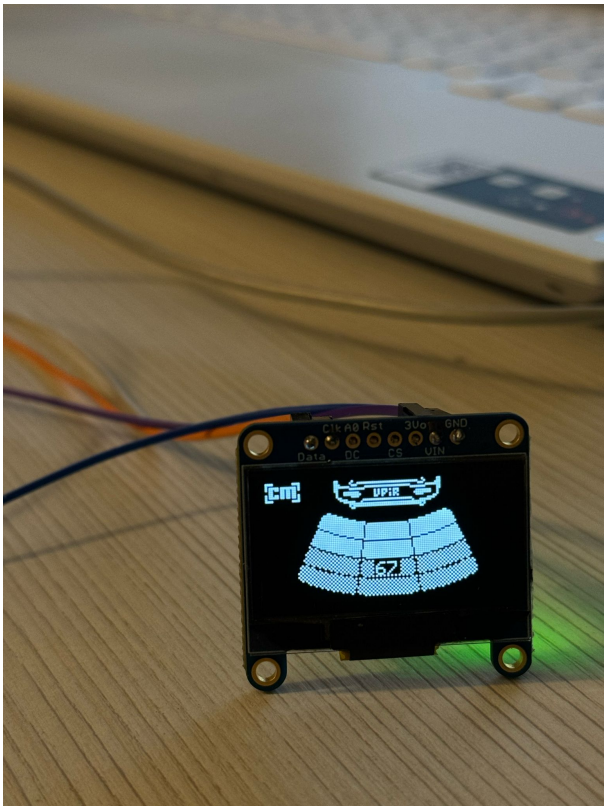
setup(): Inițializează configurările necesare pentru comunicarea cu senzorul cu ultrasunete și ecranul OLED. unsigned long measurePulseHigh(uint8_t pin): Realizează măsurarea distanței cu senzorul ultrasonic.

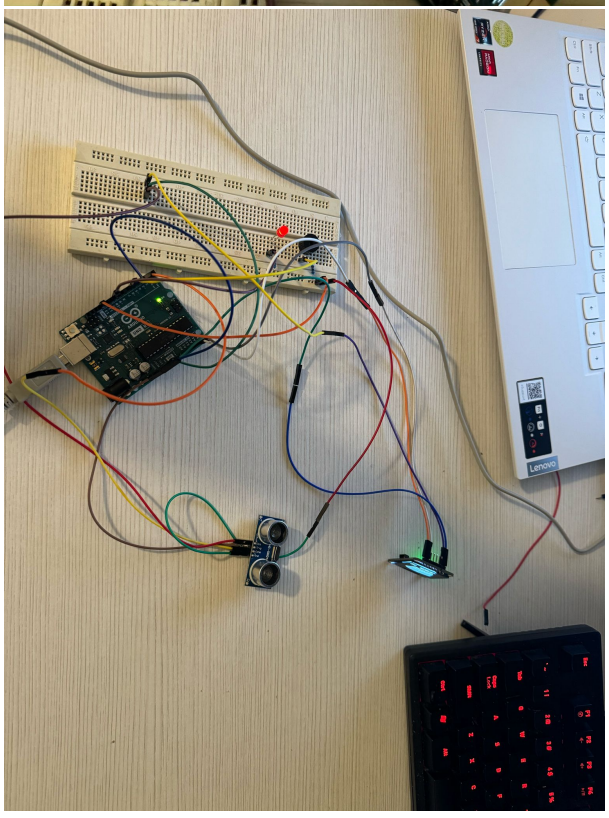
```
unsigned long measurePulseHigh(uint8_t pin) {
  unsigned long startTime = 0;
  unsigned long endTime = 0;

  while (!(PIND & (1 << pin)));
  // Record the start time
  startTime = micros();
  // Wait for the pulse to end
  while (PIND & (1 << pin));
  endTime = micros();

  return endTime - startTime;
}

  unsigned long duration = measurePulseHigh(PIN_ECHO);
  float distance_cm = duration * 0.034 / 2; // se inmulteste cu viteza
sunetului (cm/us) and se imparte la 2(pentru distanta undei inainte si
inapoi)
```

loop(): Implementează bucla principală a programului, care: Procesează datele și actualizează afișajul pe ecranul OLED. Gestionează alte acțiuni sau funcționalități specifice proiectului, cum ar fi semnalarea sonoră în caz de apropiere excesivă de obstacol.

Rezultate Obţinute







Link pentru video: videos.rar

Download

O arhivă (sau mai multe dacă este cazul) cu fişierele obţinute în urma realizării proiectului: surse, scheme, etc. Un fişier README, un ChangeLog, un script de compilare şi copiere automată pe uC crează întotdeauna o impresie bună 😉.

Fişierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fişierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul). **Exemplu:** Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

Bibliografie/Resurse

Resurse Hardware

https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf
https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf
https://html.alldatasheet.com/html-pdf/1179026/ETC2/SSD1306/111/1/SSD1306.html

Resurse Software

https://projecthub.arduino.cc/Isaac100/getting-started-with-the-hc-sr04-ultrasonic-sensor-7cabe1
https://docs.arduino.cc/built-in-examples/basics/Blink/

Export to PDF