

Parking Sensor

Introducere

Proiectul este un senzor de parcare care utilizeaza un ecran LCD, LED-uri si un buzzer pentru a afisa distanta pana la obstacole. Acesta măsoară distanța față de obstacole folosind un senzor ultrasonic, afișează distanța pe un ecran LCD, aprinde LED-uri de diferite culori în funcție de distanță (roșu, galben, verde) si emite sunete de avertizare printr-un buzzer. Am vrut sa fac o solutie ieftina pentru momentul in care parchez masina in garaj, pe bara din fata nu am senzori de parcare, iar aceasta solutie este una simpla, eficienta si cu un cost redus.

Descriere generală



Descrierea modulelor proiectului si a modului in care interactioneaza

1. Senzorul Ultrasonic:

Descriere: Utilizeaza unde sonore pentru a masura distanta pana la un obstacol. Include un emitator (trig) si un receptor (echo).

Interactiune: Trimite un puls ultrasonic prin pinul trig, apoi masoara timpul pana cand pulsul este receptionat prin pinul echo. Arduino calculeaza distanta pe baza acestui timp.

2. Arduino:

Descriere: Microcontroller-ul central care gestioneaza toate componentele proiectului.

Interactiune: Colecteaza date de la senzorul ultrasonic, proceseaza aceste date si controleaza LED-urile, buzzer-ul si afisajul LCD. Ruleaza codul scris in Arduino IDE care defineste comportamentul sistemului.

3. Afisajul LCD:

Descriere: Ecran cu 16x2 caractere pentru afisarea distantei pana la

obstacol.

Interactiune: Arduino trimite informatii de distanta catre LCD folosind pinii digitali. LCD-ul afiseaza aceasta distanta pentru a oferi un feedback vizual utilizatorului.

4. LED-uri (Rosu, Galben, Verde):

Descriere: Indicatoare vizuale pentru a semnaliza diferite intervale de distanta fata de obstacol.

Interactiune: Arduino controleaza starea LED-urilor pe baza distantei masurate. LED-ul rosu se aprinde pentru distante mici (pericol), galben pentru distante medii si verde pentru distante mari (siguranta).

5. Buzzer:

Descriere: Dispozitiv de avertizare sonora pentru a semnala proximitatea unui obstacol.

Interactiune: Arduino activeaza buzzer-ul la diferite frecvente si intervale in functie de distanta masurata. Frecventa sunetului creste pe masura ce distanta fata de obstacol scade.

Mod de interactiune:

Senzorul ultrasonic masoara distanta pana la obstacol si trimite datele catre Arduino.

Arduino proceseaza aceste date si decide ce informatii sa afiseze si ce actiuni sa intreprinda.

Afisajul LCD primeste datele de distanta de la Arduino si le afiseaza.

LED-urile sunt activate de Arduino in functie de distanta masurata pentru a oferi un feedback vizual clar.

Buzzer-ul este activat de Arduino pentru a oferi un avertisment sonor care devine mai intens pe masura ce distanta scade.

Potentiometrul permite ajustarea contrastului afisajului LCD pentru o vizibilitate optima.

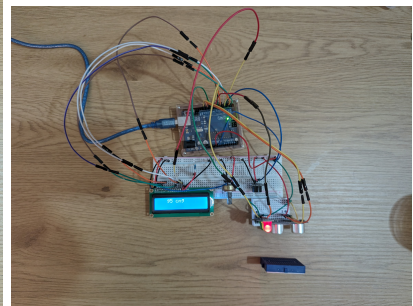
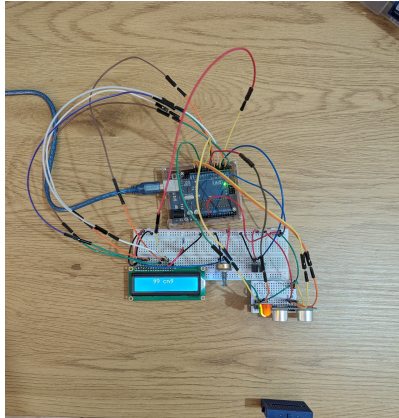
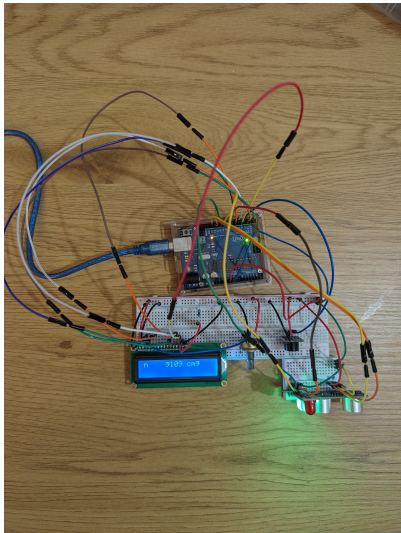
Toate componentele sunt alimentate printr-o sursa de alimentare comuna care asigura functionarea corecta a intregului sistem.

Hardware Design

Componente hardware folosite:

Arduino Uno
Modul LCD 1602
Senzor ultrasonic distanta HC-SR04

Modul LED semafor
 Modul Buzzer Activ
 Breadboard și fire pentru conexiuni
 Potentiometru 100



Pinii folositi pentru fiecare componenta si justificarea alegerii lor

Senzor Ultrasonic HC-SR04:

TrigPin (Pin 12): Configurat ca OUTPUT pentru a trimite impulsuri ultrasonice. Pinul 12 este folosit deoarece este un pin digital care poate genera un semnal precis necesar pentru declansarea impulsului ultrasonic.

EchoPin (Pin 13): Configurat ca INPUT pentru a receptiona ecoul undelor ultrasonice reflectate. Pinul 13 este folosit pentru a masura cu precizie durata semnalului de eco, necesar pentru calculul distantei.

LCD 16x2:

RS (Register Select) (Pin 1): Selecteaza intre modul de comanda si modul de scriere de date. Pinul 1 este folosit deoarece este configurat pentru a comuta intre aceste moduri esentiale de operare a LCD-ului.

E (Enable) (Pin 2): Activeaza scrierea datelor la bordul LCD. Pinul 2 este ales pentru a activa transferul de date intre Arduino si LCD in momente precise.

D4, D5, D6, D7 (Pinii 7, 6, 5, 4): Folositi pentru transmiterea datelor in modul 4-bit. Pinii 7, 6, 5 si 4 sunt utilizati pentru a reduce numarul de pinii necesari pentru conectarea LCD-ului, permitand totodata transferul eficient de date.

LED-uri:

LED Rosu (Pin 8): Indica distante foarte mici (pericol de coliziune). Pinul 8 este utilizat deoarece este un pin digital care poate controla starea ON/OFF a LED-ului.

LED Galben (Pin 11): Indica distante medii. Pinul 11 este ales pentru a controla starea LED-ului galben fara a interfera cu ceilalti pinii.

LED Verde (Pin 10): Indica distante mari (zona sigura). Pinul 10 este

folosit pentru a controla starea LED-ului verde, avand un rol similar cu ceilalti pini pentru LED-uri.

Buzzer:

Buzzer Pin (Pin 3): Utilizat pentru a genera sunete in functie de distanta pana la obstacol. Pinul 3 este un pin PWM (Pulse Width Modulation), capabil sa genereze semnale de frecventa variabila, esential pentru controlul sunetelor emise de buzzer.

Justificarea alegerii pinilor:

Compatibilitate si functionalitate: Fiecare pin a fost selectat pentru a asigura compatibilitatea cu functiile necesare (OUTPUT pentru semnale si INPUT pentru masuratori).

Evitarea conflictelor: Pinilor li s-au atribuit roluri specifice pentru a evita conflictele si interferentele intre componentele multiple conectate la Arduino.

Utilizarea eficienta a resurselor: Alegerea pinilor digitali si PWM optimizeaza utilizarea resurselor Arduino, asigurand performanta si eficienta maxima in functionarea proiectului.

Simplicitate si claritate: Pinii au fost selectati si aranjati pentru a mentine cablarea simpla si clara, facilitand asamblarea si depanarea sistemului.

Software Design

Mediu de dezvoltare: Arduino IDE
Biblioteci folosite: LiquidCrystal

Rezultate Obținute

Un senzor functional, ce imi afiseaza distanta pe un LCD 1602, aprinde unul din cele trei leduri in functie de distanta obstacolului, si emite un sunet printr-un buzzer asemenator celor de la senzorii de parcare ai unei masini.

Concluzii

Download

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

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

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** și **Resurse Hardware**.

[Export to PDF](#)

From:
<http://ocw.cs.pub.ro/courses/> - **CS Open CourseWare**

Permanent link:
<http://ocw.cs.pub.ro/courses/pm/prj2024/apredescu/alexandru.dumenica> 

Last update: **2024/05/27 11:18**