Parcare Rezidentiala

Introducere

Marinescu Alexandru - 332CA

Poriectul reprezinta o Parcare rezidentiala, care permite accesul, printr-o cu cartela si care afiseaza locurile libere. Locurile libere fiind afisate pe un ecran, iar fiecare loc fiind evidentiat de un led daca este liber sau nu.

Descriere generală

Am gandit acest proiect ca sa evidentiez o utilizare reala a unui astfel de sistem, Arduino o sa se ocupe de fiecare senzor si de fiecare interacțiune intre LCD, senzori, servo-motor si RFID.

Utilizatorii o sa vina, o sa scaneze cardul RFID primit la cumpararea locuintei, daca acesta este valid, se aprinde beculetul si se ridica bariera. De asemenea, pentru eficientizare, gasirea mai simpla a unui loc liber, si evitarea incercarii de parcare cand nu sunt locuri se foloseste de Ecran si led-urile asignate fiecarui loc de parcare.

Aici este diagrama proiectului:

Hardware Design

Aici puneţi tot ce ţine de hardware design:

- listă de piese
- scheme electrice (se pot lua şi de pe Internet şi din datasheet-uri, e.g. http://www.captain.at/electronic-atmega16-mmc-schematic.png)
- diagrame de semnal
- rezultatele simulării

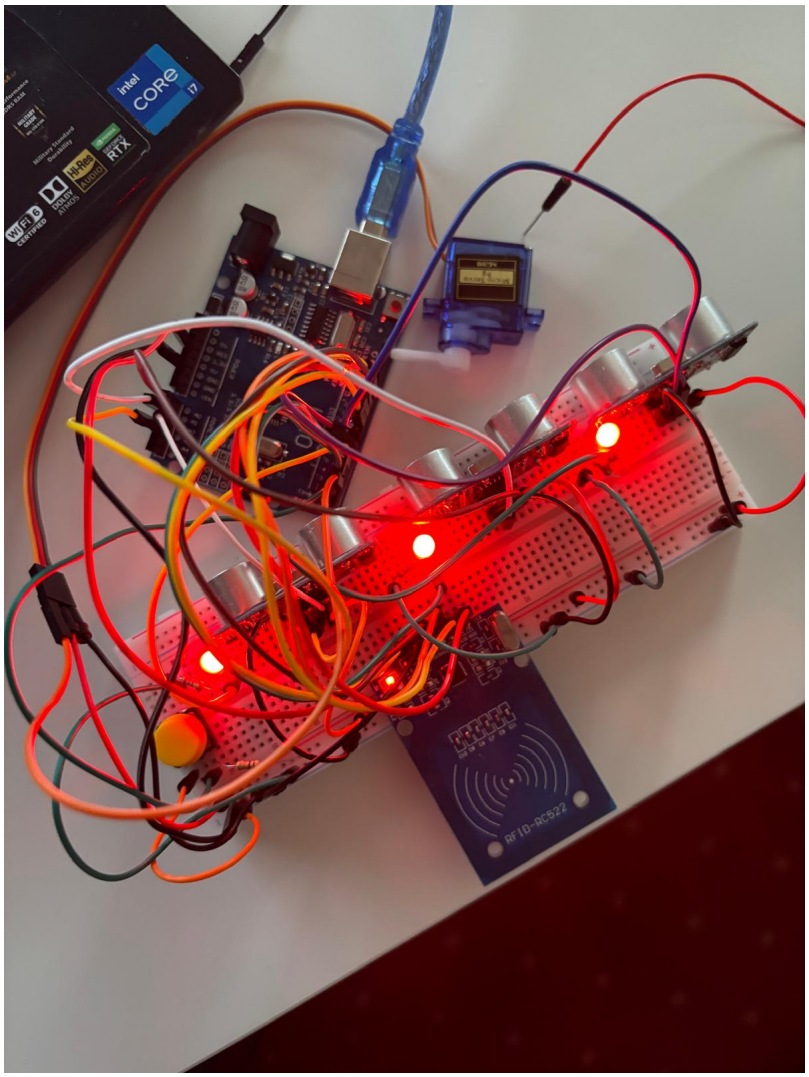
Lista de componente:

- Arduino UNO R3 - microcontrollerul principal care gestioneaza toate componentele și functiile

sistemului de parcare
- Ecran IIC 1602 LCD - afiseaza mesaj daca avem loc liber de parcare
- RFID MRFC-522 - detecteaza cartela folosita pentru a intra in parcarea rezidentiala
- Senzori ultrasonici HC-SR04 - decteaza prezenta unei masini in dreptul acestora, fiind montat in proximitatea locului de parcare
- IC Shift register SN74HC595N 74HC595 - controleaza multiple outputuri folosind doar 3 pini ai microcontrollerului. Am nevoie de el datorita numarului rrelativ mare de device-uri si numarului mic de porturi digitali ai microcontrollerului.
- ServoMotor SG90 - este folosit pentru a deschide si inchide bariera de intrare in parcare
- BYJ48 Stepper Motor - folosit pentru a ridica si inchide cealalta bariera de iesire din parcare
- Buton Pushbutton - folosit pentru a comanda ridicarea barierei de la iesire.

Schema electrica:

Stadiul actual:



Software Design

Link către repo-ul de Github cu codul sursă: https://github.com/AlexMari1234/Parcare-rezidentiala

Mediul de Dezvoltare

Codul este destinat a fi rulat pe o platformă Arduino, utilizând mediul de dezvoltare Arduino IDE. Arduino IDE oferă suport pentru scrierea, compilarea și încărcarea codului pe plăcile de dezvoltare Arduino. Mediul de dezvoltare este ales pentru simplitatea sa și pentru suportul extins al comunității pentru proiecte de automatizare și robotică.

Librării

Codul utilizează mai multe librării pentru a gestiona diverse componente hardware:

- **Servo.h**: Furnizată de Arduino, simplifică controlul servomotoarelor. Această librărie gestionează generarea semnalelor PWM necesare pentru a controla un servo motor standard.

- **MFRC522.h**: Librărie utilizată pentru interfațarea cu cititorul RFID MFRC522.

- **LiquidCrystal_I2C.h**: Utilizată pentru controlul unui ecran LCD cu interfață I2C, simplificând astfel conectarea și comunicarea cu ecranul.

Organizarea Codului

Codul este structurat în două secțiuni principale: `setup()` și `loop()`. Secțiunea `setup()` este destinată inițializării componentelor hardware și configurării inițiale a sistemului, în timp ce `loop()` conține logica principală care se repetă continuu în timpul funcționării dispozitivului. Codul include următoarele elemente:

1. **Include librării**: La început, codul include librăriile necesare pentru controlul servomotorului, cititorului RFID și ecranului LCD. Aceasta asigură că toate funcționalitățile necesare sunt disponibile pe parcursul programului.

1. **Defineste constante**: Utilizează directive `#define` pentru a denumi pinii senzorilor ultrasonici, LED-urilor, butonului și cititorului RFID. De asemenea, sunt definite constante pentru timpii de operare ai servo motorului, pentru a simplifica modificările ulterioare și a crește lizibilitatea codului.

1. **Inițializează variabile**: Declara variabile pentru a păstra starea inițială a servo motorului și pentru a gestiona temporizarea acțiunilor, cum ar fi timpul de ridicare și coborâre a barierei, precum și gestionarea stării de deschidere a barierei.

1. **Configurare inițială (`setup()`)**: Această secțiune inițializează comunicația serială, configurarea pinilor pentru servo motor, senzorii ultrasonici, LED-uri, buton și ecran LCD. Inițializarea componentelor hardware este esențială pentru a asigura că toate dispozitivele funcționează corect de la începutul execuției programului.

1. **Logica de control (`loop()`)**: Aceasta este secțiunea principală a codului care se repetă continuu și conține logica necesară pentru operarea sistemului de parcare. Include verificări pentru detectarea cardurilor RFID, citirea distanțelor de la senzorii ultrasonici, controlul LED-urilor și gestionarea afișării informațiilor pe ecranul LCD.

Algoritmi și Structuri Implementate

1. **Citirea senzorilor ultrasonici**: Valorile de la trei senzori ultrasonici sunt citite și utilizate pentru a determina dacă un loc de parcare este ocupat. Aceasta se face prin trimiterea unui puls de trigger și măsurarea timpului de răspuns al pulsului de ecou pentru a calcula distanța.

1. **Controlul LED-urilor**: LED-urile sunt aprinse sau stinse pe baza distanțelor măsurate de senzori. Dacă distanța este mai mică de 8 cm, LED-ul corespunzător se aprinde.

1. **Controlul servo motorului**: Servo motorul este controlat pentru a ridica și a coborî bariera de acces. Dacă un card RFID autorizat este detectat sau butonul de ieșire este apăsat, bariera se ridică pentru 5 secunde și apoi coboară automat.

1. **Afișarea pe ecranul LCD**: Ecranul LCD este utilizat pentru a afișa numărul de locuri de parcare libere și mesajul de status “Parking free spots”.

Rezultate Obținute

1. Sistemul permite deschiderea și închiderea barierei de acces în parcare la detectarea unui card RFID autorizat sau la apăsarea unui buton.
2. Ecranul LCD afișează în timp real numărul locurilor de parcare libere.
3. LED-urile indică vizual starea fiecărui loc de parcare.
4. Sistemul oferă o soluție eficientă și automatizată pentru gestionarea accesului și a disponibilității locurilor de parcare într-o parcare rezidențială.

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**.

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