Parcare cu bariera

Sistem de Parcare Inteligenta cu Acces RFID si Monitorizare Locuri

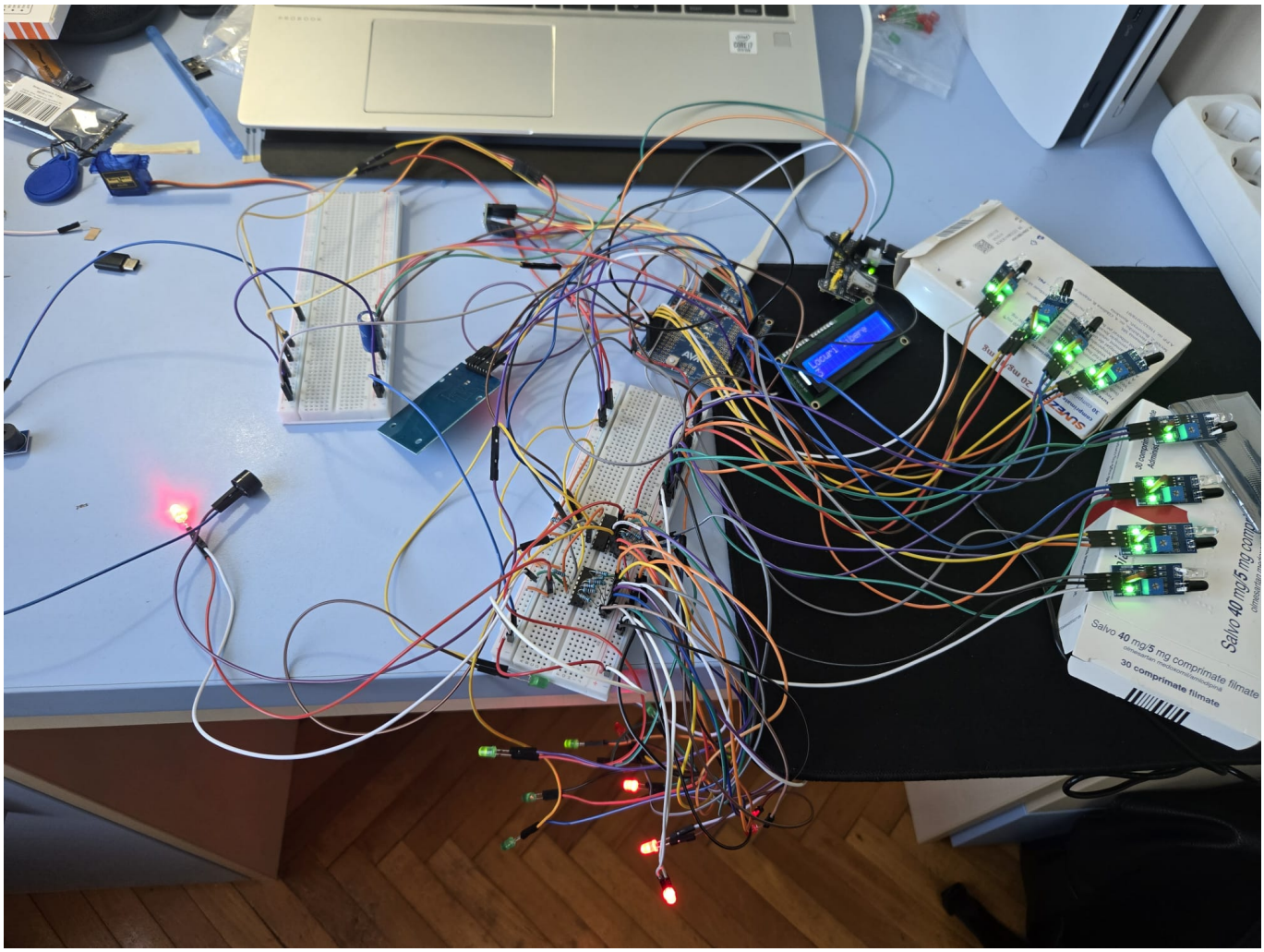
Introducere

Proiectul consta intr-un sistem de parcare automatizata care permite accesul controlat prin RFID si monitorizeaza in timp real disponibilitatea locurilor de parcare.

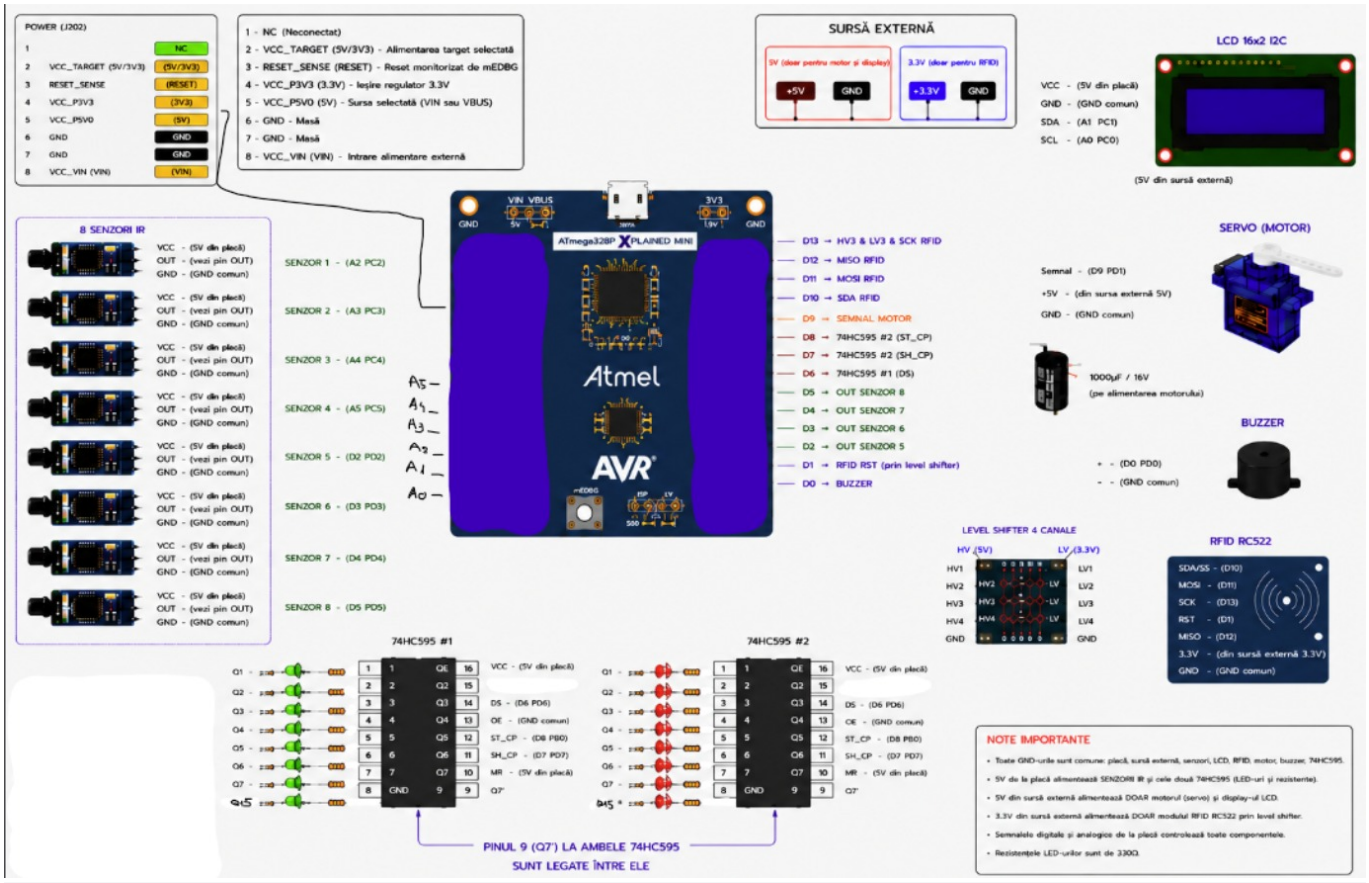
Sistemul detecteaza daca un utilizator are acces valid folosind un card RFID si permite deschiderea unei bariere automate doar daca exista locuri libere. In acelasi timp, fiecare loc de parcare este monitorizat cu senzori, iar starea acestuia este indicata prin LED-uri (verde pentru liber, rosu pentru ocupat). Un display afiseaza numarul de locuri disponibile sau mesajul “Parcare ocupata”.

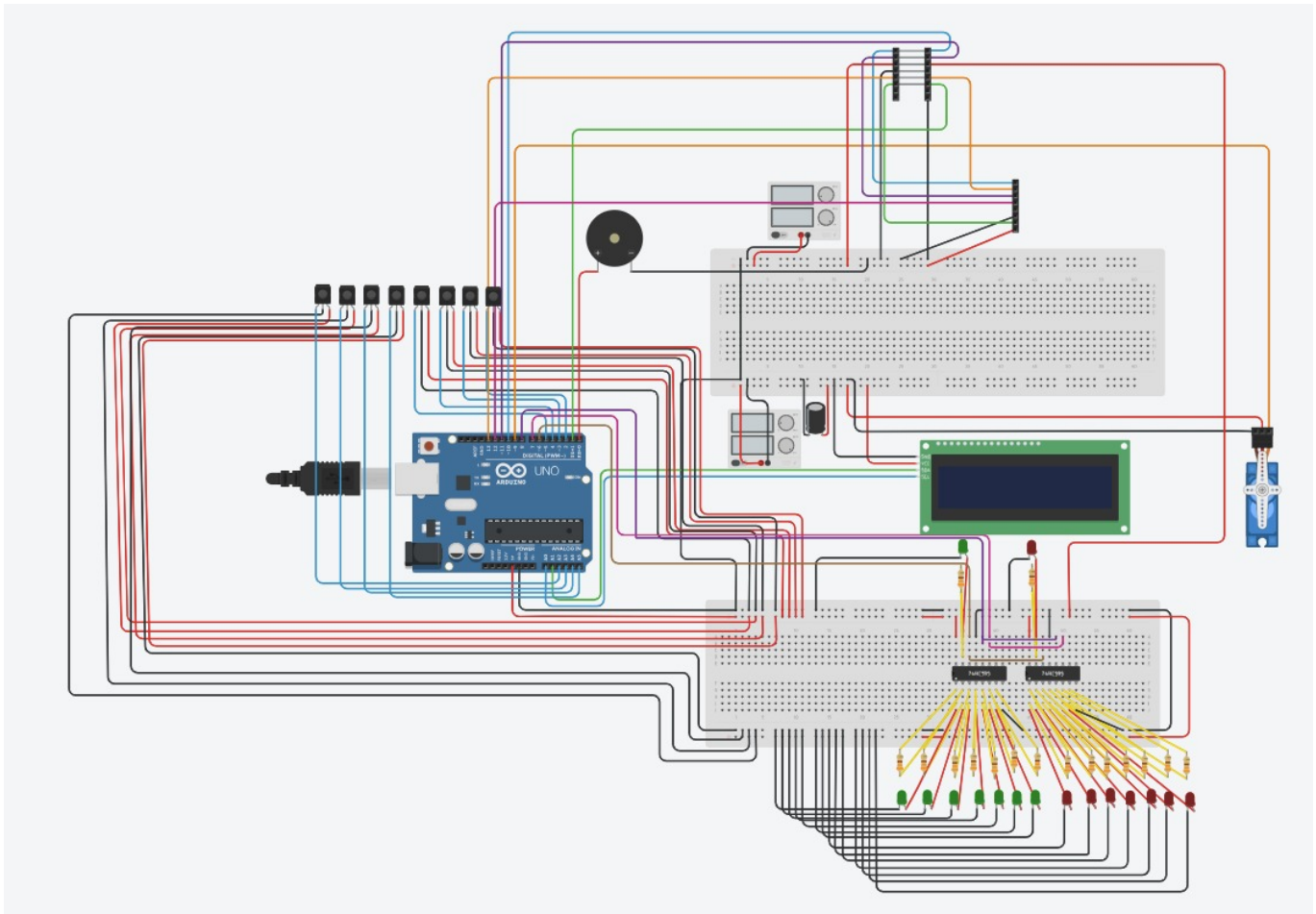
Ideea proiectului a pornit de la sistemele reale de parcare din centre comerciale si ansambluri rezidentiale.

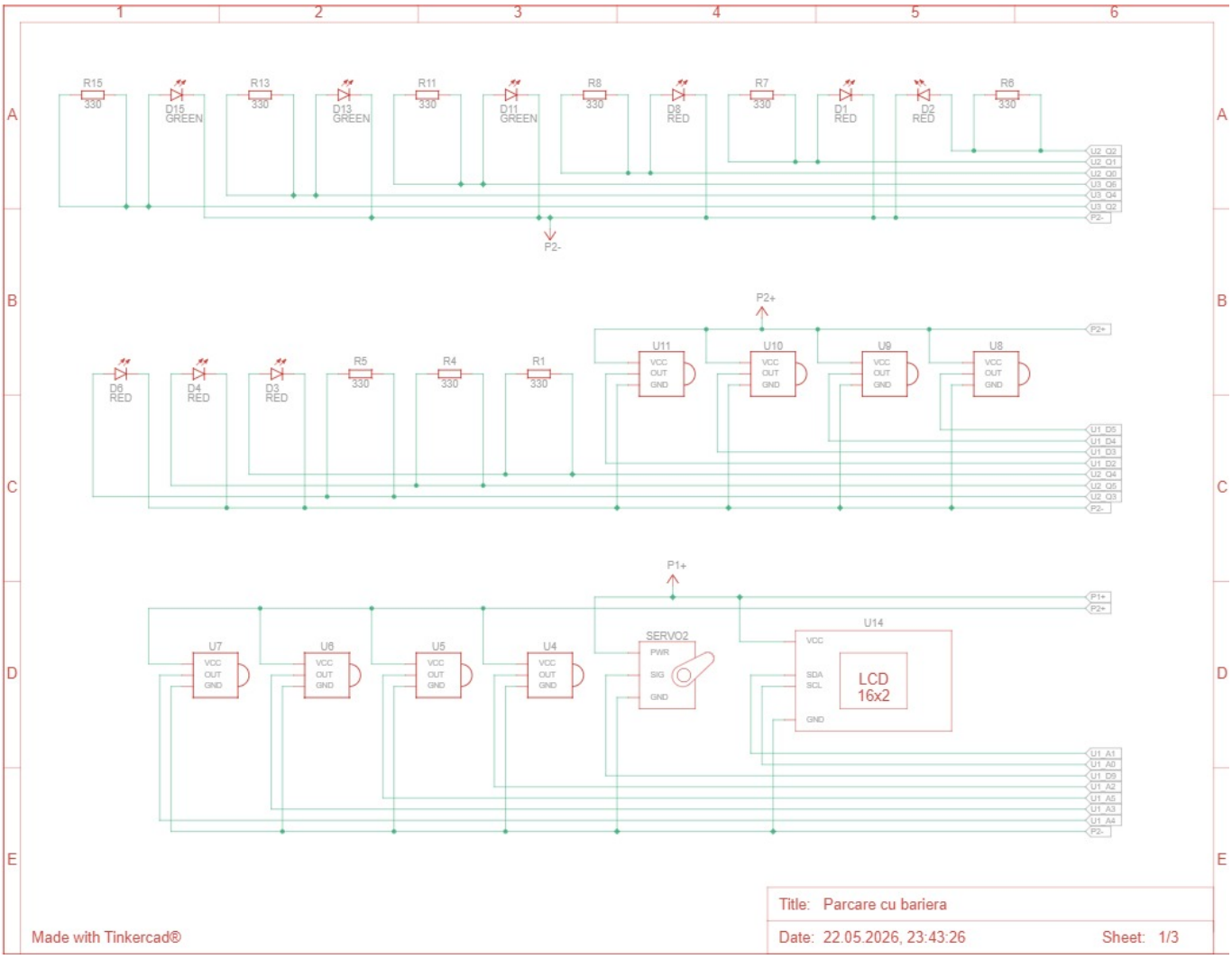
Proiectul este util deoarece demonstreaza integrarea mai multor componente hardware si software (RFID, senzori, control motor, afisaj) intr-un sistem embedded real.

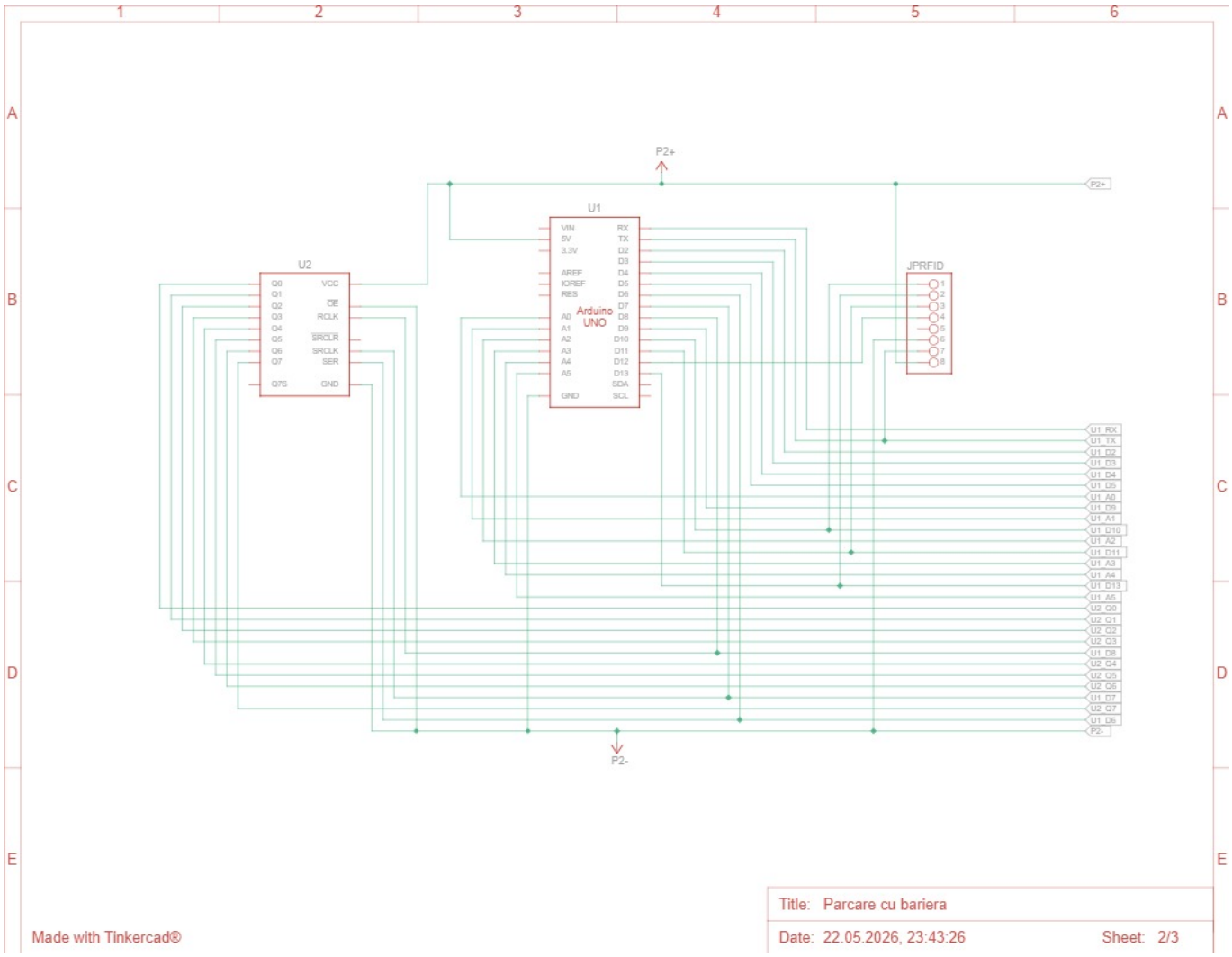


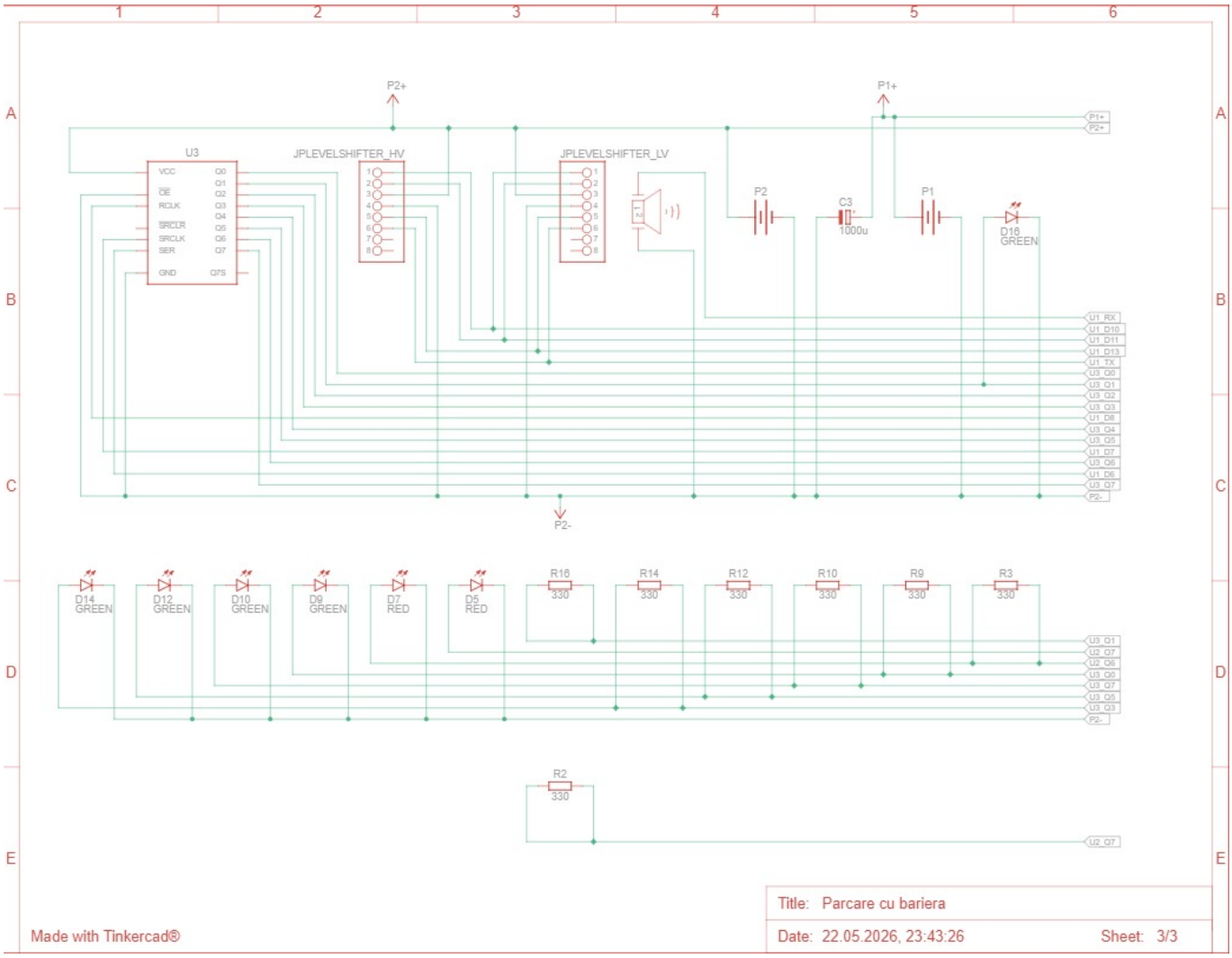
Schema electrica


POWER (J202)
1 NC
2 VCC_TARGET (5V/3V3)
3 RESET_SENSE
4 VCC_P3V3
5 VCC_P5V0
6 GND
7 GND
8 VCC_VIN (VIN)
1 - NC (Neconectat)
2 - VCC_TARGET (5V/3V3) - Alimentarea target selectată
3 - RESET_SENSE (RESET) - Reset monitorizat de mEDBG
4 - VCC_P3V3 (3.3V) - Ieșire regulator 3.3V
5 - VCC_P5V0 (5V) - Sursa selectată (VIN sau VBUS)
6 - GND - Masă
7 - GND - Masă
8 - VCC_VIN (VIN) - Intrare alimentare externă
SURSĂ EXTERNĂ
5V (doar pentru motor și display)
+5V
GND
3.3V (doar pentru RFID)
+3.3V
GND
LCD 16x2 I2C
VCC - (5V din placă)
GND - (GND comun)
SDA - (A1 PC1)
SCL - (A0 PC0)
(5V din sursă externă)
8 SENZORI IR
VCC - (5V din placă)
OUT - (vezi pin OUT)
GND - (GND comun)
SENZOR 1 - (A2 PC2)
SENZOR 2 - (A3 PC3)
SENZOR 3 - (A4 PC4)
SENZOR 4 - (A5 PC5)
SENZOR 5 - (D2 PD2)
SENZOR 6 - (D3 PD3)
SENZOR 7 - (D4 PD4)
SENZOR 8 - (D5 PD5)
ATmega328P XPLAINED MINI
Atmel
AVR
D13 → HV3 & LV3 & SCK RFID
D12 → MISO RFID
D11 → MOSI RFID
D10 → SDA RFID
D9 → SEMNAL MOTOR
D8 → 74HC595 #2 (ST_CP)
D7 → 74HC595 #2 (SH_CP)
D6 → 74HC595 #1 (DS)
D5 → OUT SENZOR 8
D4 → OUT SENZOR 7
D3 → OUT SENZOR 6
D2 → OUT SENZOR 5
D1 → RFID RST (prin level shifter)
D0 → BUZZER
SERVO (MOTOR)
Semnal - (D9 PD1)
+5V - (din sursa externă 5V)
GND - (GND comun)
1000µF / 16V
(pe alimentarea motorului)
BUZZER
+ - (D0 PD0)
- - (GND comun)
LEVEL SHIFTER 4 CANALE
HV (5V)
LV (3.3V)
RFID RC522
SDA/SS - (D10)
MOSI - (D11)
SCK - (D13)
RST - (D1)
MISO - (D12)
3.3V - (din sursă externă 3.3V)
GND - (GND comun)
74HC595 #1
74HC595 #2
PINUL 9 (Q7') LA AMBELE 74HC595
SUNT LEGATE ÎNTRE ELE
NOTE IMPORTANTE

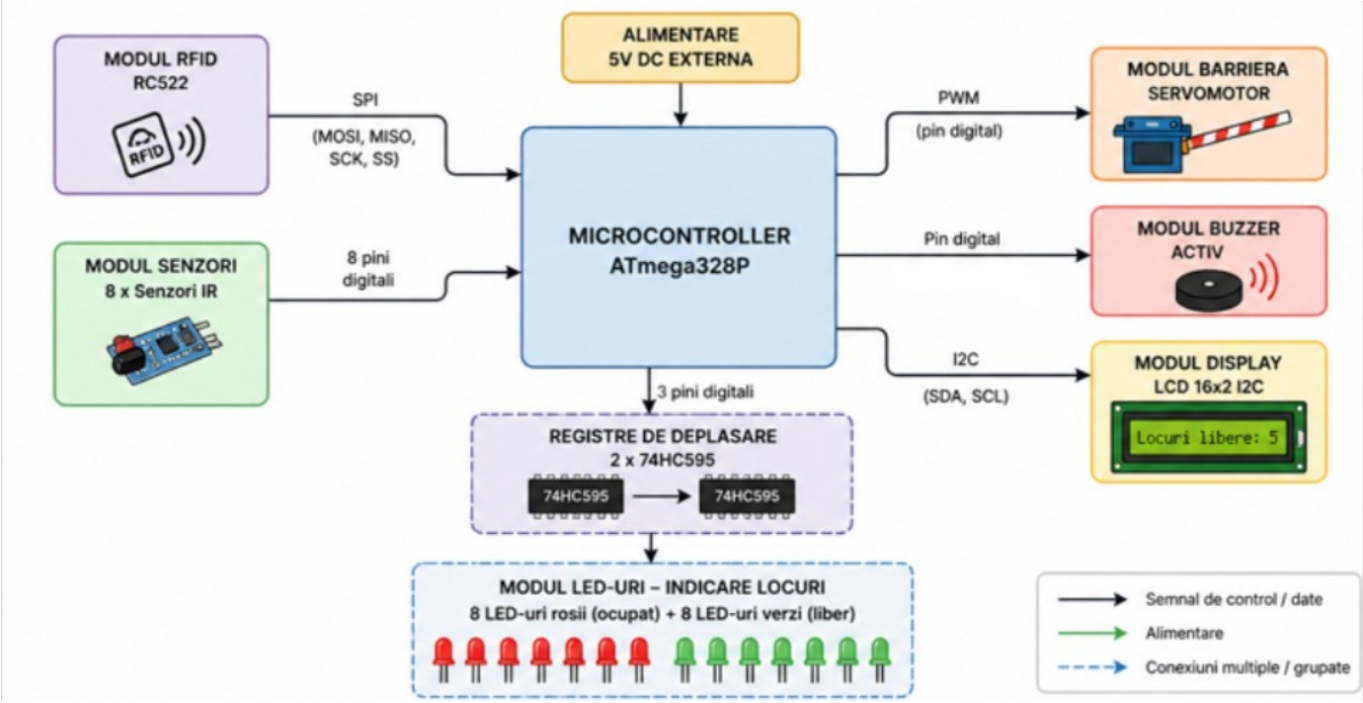

UNO


R15
330
D15
GREEN
R13
330
D13
GREEN
R11
330
D11
GREEN
R8
330
D8
RED
R7
330
D1
RED
D2
RED
R6
330
P2-
P2+
D6
RED
D4
RED
D3
RED
R5
330
R4
330
R1
330
U11
U10
U9
U8
VCC
OUT
GND
P1+
U7
U6
U5
U4
SERVO2
PWR
SIG
GND
U14
SDA
SCL
LCD
16x2
Made with Tinkercad®
Title: Parcare cu bariera
Date: 22.05.2026, 23:43:26
Sheet: 1/3


P2+
U1
U2
JPRFID
Arduino UNO
P2-
Made with Tinkercad®
Title: Parcare cu bariera
Date: 22.05.2026, 23:43:26
Sheet: 2/3



Descriere generala



Proiectul realizat reprezinta un sistem de parcare inteligenta bazat pe microcontrolerul ATmega328P. Sistemul monitorizeaza in timp real ocuparea locurilor de parcare utilizand senzori IR si permite accesul automat prin autentificare RFID. Informatiile privind disponibilitatea locurilor sunt afisate pe un display LCD 16×2 cu interfata I2C, iar semnalizarea vizuala este realizata cu LED-uri verzi si rosii controlate prin registre de deplasare 74HC595. Accesul in parcare este controlat printr-un servo motor care actioneaza bariera dupa validarea unui card RFID. Sistemul utilizeaza si un buzzer pentru feedback sonor la ridicarea si coborarea barierei.

Pentru stabilitatea alimentarii, proiectul foloseste surse externe separate: -sursa externa de 5V pentru servo motor si display LCD; -sursa externa de 3.3V pentru modulul RFID RC522.

Toate componentele utilizeaza masa comuna (GND comun).

Hardware Design

Lista de piese:

Nr.	Componenta	Cantitate	Rol in proiect	Interfata folosita
1	ATmega328P Xplained Mini	1 buc	Unitatea centrala de procesare	-
2	Display LCD 16×2 cu modul I2C	1 buc	Afisarea locurilor disponibile si a mesajelor sistemului	I2C (SDA, SCL)
3	Modul RFID RC522	1 buc	Citirea cardurilor RFID pentru acces	SPI
4	Servo motor SG90	1 buc	Actionarea barierei de acces	PWM
5	Buzzer pasiv 5V	1 buc	Feedback audio la acces	GPIO / PWM
6	Registru de deplasare 74HC595	2 buc	Controlul extins al LED-urilor	Serial GPIO
7	LED-uri verzi	8 buc	Semnalizarea locurilor libere	GPIO prin 74HC595

8	LED-uri rosii	8 buc	Semnalizarea locurilor ocupate	GPIO prin 74HC595
9	Rezistoare 330 ohm	16 buc	Limitarea curentului pentru LED-uri	-
10	Senzori IR	8 buc	Detectarea ocuparii locurilor de parcare	GPIO
11	Level Shifter bidirectional 4 canale	1 buc	Conversia nivelelor logice 5V ↔ 3.3V pentru RFID	SPI
12	Condensator electrolitic 1000uF / 16V	1 buc	Stabilizarea alimentarii servo motorului	-
13	Sursa externa 5V	1 buc	Alimentarea servo motorului si display-ului LCD	Alimentare
14	Sursa externa 3.3V	1 buc	Alimentarea modulului RFID RC522	Alimentare
15	Breadboard MB-102	1 buc	Realizarea conexiunilor electrice	-
16	Fire jumper tata-tata si mama-tata	1 set	Realizarea conexiunilor intre componente	-

Conexiuni:

Pin ATmega328P	**Rol**
PC0 (A0)	Display LCD I2C - SCL / Clock
PC1 (A1)	Display LCD I2C - SDA / Data
PC2 (A2)	Senzor IR 1 - semnal OUT
PC3 (A3)	Senzor IR 2 - semnal OUT
PC4 (A4)	Senzor IR 3 - semnal OUT
PC5 (A5)	Senzor IR 4 - semnal OUT
PD0 (D0)	Buzzer pasiv
PD1 (D1)	RFID RC522 - pin RST
PD2 (D2)	Senzor IR 5 - semnal OUT
PD3 (D3)	Senzor IR 6 - semnal OUT
PD4 (D4)	Senzor IR 7 - semnal OUT
PD5 (D5)	Senzor IR 8 - semnal OUT
PD6 (D6)	74HC595 - pin DS / Data
PD7 (D7)	74HC595 - pin SH_CP / Clock
PB0 (D8)	74HC595 - pin ST_CP / Latch
PB1 (D9)	Servo motor - semnal PWM
PB2 (D10)	RFID RC522 - SDA / SS
PB3 (D11)	RFID RC522 - MOSI
PB4 (D12)	RFID RC522 - MISO
PB5 (D13)	RFID RC522 - SCK prin level shifter

Software Design

Stadiul actual al implementarii software

Implementarea software este functionala si integreaza toate modulele principale ale proiectului: citirea celor 8 senzori IR, afisarea numarului de locuri libere pe LCD, controlul LED-urilor prin doua registre 74HC595, citirea cardului RFID RC522, actionarea barierei prin servo motor si semnalizarea sonora cu buzzer.

Codul este scris in C pentru AVR, fara framework Arduino, folosind direct registrele microcontrollerului ATmega328P. Programul initializeaza perifericele necesare, citeste periodic senzorii, actualizeaza LED-urile si display-ul, iar la detectarea unui card RFID decide daca permite sau refuza accesul in functie de numarul de locuri disponibile.

Biblioteci folosite

In varianta finala a proiectului nu sunt folosite biblioteci externe Arduino. Am ales implementarea directa in C AVR pentru a avea control complet asupra registrelor microcontrollerului si pentru a demonstra utilizarea directa a notiunilor parcurse la laborator.

Sunt folosite doar headerele standard AVR:

- avr/io.h - pentru acces direct la registrele microcontrollerului;
- util/delay.h - pentru intarzieri controlate;
- stdint.h - pentru tipuri de date fixe, precum uint8_t si uint16_t;
- stdbool.h - pentru tipul boolean.

Comunicatiile I2C si SPI sunt implementate manual prin configurarea registrelor TWI si SPI ale microcontrollerului. Controlul servo motorului este realizat prin Timer1, iar LED-urile sunt controlate prin semnale digitale trimise catre registrele 74HC595.

Elementul de noutate al proiectului

Elementul de noutate al proiectului consta in integrarea mai multor functionalitati intr-un sistem complet de parcare inteligenta. Sistemul nu doar afiseaza locurile disponibile, ci coreleaza informatia primita de la senzori cu semnalizarea vizuala, controlul accesului RFID si actionarea automata a barierei.

Un alt element important este utilizarea registrelor 74HC595 pentru controlul celor 16 LED-uri folosind doar 3 pini ai microcontrollerului. Astfel, proiectul extinde numarul de iesiri disponibile si permite controlul individual al LED-urilor pentru fiecare loc de parcare.

Functionalitati din laborator utilizate

Proiectul foloseste mai multe concepte studiate in laboratoare:

- GPIO - pentru citirea senzorilor IR, controlul buzzerului si controlul registrelor 74HC595;
- Pull-up intern - pentru stabilizarea intrarilor digitale ale senzorilor;
- Timere si PWM - pentru generarea semnalului de control al servo motorului;
- SPI - pentru comunicatia cu modulul RFID RC522;
- I2C / TWI - pentru comunicatia cu display-ul LCD 16×2;
- Lucrul direct cu registre - pentru configurarea porturilor, a comunicatiilor si a Timer1;
- Interactiunea cu senzori si actuatori - pentru citirea senzorilor IR si controlul LED-urilor, buzzerului si barierei.

Aceste functionalitati sunt legate direct de continutul laboratoarelor: GPIO si lucrul cu registrele sunt introduse in laboratorul 0, timerele si intreruperile in laboratorul 2, PWM-ul in laboratorul 3, SPI-ul in laboratorul 5, iar I2C-ul in laboratorul 6.

Fragmente relevante de cod

Configurarea comunicatiei I2C pentru LCD

Display-ul LCD foloseste magistrala I2C/TWI. Comunicatia este initializata prin configurarea registrelor TWSR, TWBR si TWCR.

```
void twi_init() {
  TWSR = 0x00;
  TWBR = 72;
  TWCR = (1 << TWEN);
}
```

Registrul TWBR stabileste viteza magistralei I2C, iar bitul TWEN activeaza perifericul TWI al microcontrollerului. Aceasta functionalitate este folosita pentru transmiterea comenzilor si caracterelor catre display.

Trimiterea datelor catre cele doua registre 74HC595

Pentru controlul celor 16 LED-uri am folosit doua registre 74HC595. Astfel, toate LED-urile pot fi controlate folosind doar 3 pini ai microcontrollerului.

```
void shift595_send(uint16_t value) {
  PORTB &= ~(1 << LATCH_PIN);

  for (int i = 15; i >= 0; i--) {
    if (value & (1 << i)) PORTD |= (1 << DATA_PIN);
    else PORTD &= ~(1 << DATA_PIN);

    PORTD |= (1 << CLOCK_PIN);
```

```
    _delay_us(2);
    PORTD &= ~(1 << CLOCK_PIN);
  }

  PORTB |= (1 << LATCH_PIN);
}
```

Functia trimite pe rand cei 16 biti catre registre. Primii 8 biti controleaza LED-urile rosii, iar ceilalti 8 biti controleaza LED-urile verzi. La final, semnalul latch actualizeaza iesirile registrelor.

Calcularea locurilor libere si actualizarea LED-urilor

Senzorii IR sunt cititi periodic. In functie de starea fiecarui senzor, se aprinde LED-ul verde pentru loc liber sau LED-ul rosu pentru loc ocupat.

```
for (uint8_t i = 0; i < 8; i++) {
  if (senzori_liber[i]) {
    locuri++;
    leduri |= (1 << i);
  } else {
    leduri |= (1 << (i + 8));
  }
}

shift595_send(leduri);
```

Variabila leduri contine starea tuturor celor 16 LED-uri. Bitii 0-7 sunt folositi pentru LED-urile verzi, iar bitii 8-15 pentru LED-urile rosii.

Configurarea PWM pentru servo motor

Servo motorul este controlat cu Timer1, folosind iesirea OC1A de pe pinul PB1 / D9.

```
void servo_init() {
  DDRB |= (1 << PB1);

  TCCR1A = (1 << COM1A1) | (1 << WGM11);
  TCCR1B = (1 << WGM13) | (1 << WGM12) | (1 << CS11);

  ICR1 = 39999;
}
```

Timer1 este configurat in mod Fast PWM, cu perioada de aproximativ 20 ms, potrivita pentru controlul unui servo motor. Pozitia barierei este setata prin modificarea registrului OCR1A.

```
void servo_write(uint8_t angle) {
```

```
  if (angle == 0) {
    OCR1A = 2000;
  } else {
    OCR1A = 4000;
  }
}
```

Valoarea OCR1A determina durata impulsului PWM. Pentru proiect, valorile au fost calibrate astfel incat bariera sa aiba doua pozitii clare: coborata si ridicata.

Initializarea comunicatiei SPI pentru RFID

Modulul RC522 comunica prin SPI. Pinii MOSI, SCK si SS sunt configurati ca iesiri, iar MISO ca intrare.

```
void spi_init() {
  DDRB |= (1 << PB2) | (1 << PB3) | (1 << PB5);
  DDRB &= ~(1 << PB4);

  PORTB |= (1 << RFID_SS);

  SPCR = (1 << SPE) | (1 << MSTR) | (1 << SPR0);
}
```

Bitul SPE activeaza perifericul SPI, iar MSTR configureaza microcontrollerul ca master. RC522 functioneaza ca dispozitiv slave.

Citirea UID-ului cardului RFID

Pentru detectarea cardului RFID se trimite o comanda REQA, apoi se foloseste comanda anticollision pentru citirea UID-ului.

```
bool rfid_read_uid(uint8_t *uid) {
  rfid_write(0x09, 0x26);
  rfid_write(0x01, 0x0C);
  rfid_setBit(0x0D, 0x80);

  _delay_ms(10);

  if (!(rfid_read(0x04) & 0x30)) return false;

  rfid_write(0x09, 0x93);
  rfid_write(0x09, 0x20);

  _delay_ms(10);
```

```
  for (uint8_t i = 0; i < 5; i++) {
    uid[i] = rfid_read(0x09);
  }

  return true;
}
```

Daca nu este detectat niciun card, functia intoarce false. Daca un card este prezent, UID-ul este salvat intr-un vector si ulterior afisat pe LCD.

Logica principala a proiectului

Bucla principala actualizeaza permanent starea parcarii si verifica daca a fost citit un card RFID.

```
while (1) {
  int locuri = actualizeaza_parcarea();

  if (rfid_read_uid(uid)) {
    afiseaza_uid(uid);

    if (locuri == 0) {
      lcd_print("Acces refuzat");
    } else {
      deschide_bariera();
    }

    rfid_halt();
    _delay_ms(500);
    rfid_init();
  }

  _delay_ms(150);
}
```

Aceasta este partea care leaga toate modulele intre ele: senzorii determina disponibilitatea locurilor, RFID-ul declanseaza cererea de acces, iar servo motorul actioneaza bariera daca exista locuri libere.

Actionarea barierei si feedback-ul sonor

La acces permis, bariera este ridicata, buzzerul este activat, apoi dupa o pauza bariera este coborata automat.

```
void deschide_bariera() {
  lcd_clear();
  lcd_print("Acces permis");
```

```
  servo_write(90);
  buzzer_on_2s();

  _delay_ms(2000);

  lcd_clear();
  lcd_print("Bariera jos");

  servo_write(0);
  buzzer_on_2s();
}
```

Buzzerul functioneaza pe durata miscarii barierei, oferind feedback sonor utilizatorului.

Structura software

Codul este impartit in mai multe sectiuni functionale:

- LCD I2C - contine functiile pentru initializarea si controlul display-ului;
- 74HC595 - contine functia de trimitere a celor 16 biti catre registrele de deplasare;
- Servo - initializeaza Timer1 si seteaza pozitia barierei;
- Buzzer - genereaza un semnal sonor timp de aproximativ 2 secunde;
- SPI / RFID - initializeaza SPI-ul si permite citirea UID-ului cardului RFID;
- Parcare - citeste senzorii, calculeaza numarul de locuri libere si actualizeaza LED-urile;
- main - initializeaza toate modulele si ruleaza bucla principala.

In bucla principala, sistemul actualizeaza constant starea parcarii. Daca este detectat un card RFID, UID-ul este afisat pe LCD. Daca exista cel putin un loc liber, bariera se ridica, buzzerul porneste pe durata miscarii, apoi bariera coboara automat. Daca parcarea este plina, accesul este refuzat si mesajul este afisat pe display.

Validarea functionalitatilor

Functionalitatile au fost validate incremental:

- LCD-ul a fost testat separat prin afisarea unui mesaj simplu;
- senzorii IR au fost testati individual, apoi impreuna, prin afisarea starilor pe display;
- LED-urile au fost testate separat prin registrele 74HC595;
- RFID-ul a fost testat prin citirea UID-ului cardului;
- servo motorul a fost testat prin miscarea barierei intre pozitia inchisa si deschisa;
- buzzerul a fost testat impreuna cu miscarea servo motorului;
- in final, toate modulele au fost integrate intr-un singur program.

Validarea finala a constat in simularea ocuparii locurilor de parcare si apropierea unui card RFID de

cititor. Sistemul a actualizat corect numarul de locuri libere, LED-urile corespunzatoare si a actionat bariera doar atunci cand existau locuri disponibile.

Calibrarea senzorilor

Senzorii IR au fost calibrati individual folosind potentiometrul de pe fiecare modul. Pentru fiecare senzor am ajustat pragul de detectie astfel incat acesta sa isi schimbe starea atunci cand un obiect se afla in dreptul locului de parcare.

Procesul de calibrare a fost realizat astfel:

- fiecare senzor a fost conectat individual;
- iesirea OUT a fost citita de microcontroller;
- s-a apropiat un obiect de senzor pentru a verifica schimbarea starii;
- potentiometrul senzorului a fost ajustat pana cand detectia a devenit stabila;
- dupa calibrare, senzorii au fost testati impreuna in sistemul complet.

Pentru servo motor, calibrarea a constat in ajustarea valorilor OCR1A folosite pentru pozitiile barierei. Pozitia inchisa foloseste un impuls de aproximativ 1 ms, iar pozitia deschisa foloseste un impuls mai mare, ajustat astfel incat bariera sa ajunga in pozitia dorita.

Optimizari realizate

Principala optimizare a fost utilizarea celor doua registre 74HC595. Fara acestea, cele 16 LED-uri ar fi necesitat 16 pini separati ai microcontrollerului. Prin folosirea registrelor, toate LED-urile sunt controlate folosind doar 3 pini: DS, SH_CP si ST_CP.

O alta optimizare a fost renuntarea la bibliotecile Arduino si trecerea la cod C AVR cu acces direct la registre. Astfel, codul este mai apropiat de arhitectura hardware si permite control mai precis asupra perifericelor.

Au fost realizate si optimizari hardware/software pentru stabilitate:

- servo motorul si display-ul sunt alimentate din sursa externa de 5V;
- RFID-ul este alimentat din sursa externa de 3.3V;
- toate masele sunt comune;
- un condensator de 1000uF este folosit pentru stabilizarea alimentarii servo motorului;
- RFID-ul este reinitializat dupa fiecare citire pentru a permite citiri repetate ale cardurilor.

Demo video

Demo-ul video al proiectului va prezenta urmatoarele etape:

1. pornirea sistemului si afisarea numarului de locuri libere pe LCD;
2. activarea senzorilor IR si schimbarea LED-urilor din verde in rosu;

3. actualizarea automata a numarului de locuri libere;
4. citirea unui card RFID;
5. ridicarea barierei si activarea buzzerului;
6. coborarea automata a barierei;
7. cazul in care parcarea este plina si accesul este refuzat.

Link demo video: https://youtube.com/shorts/kGitde-WedE?is=PO4sWVbsrNmVk2R9

Github

https://github.com/EduardG05/ProiectPM

Bibliografie

https://www.pcbasic.com/ro/blog/ir_sensors.html

https://ocw.cs.pub.ro/courses/pm/proiect/xplainedmini

https://components101.com/sites/default/files/component_datasheet/SG90%20Servo%20Motor%20Datasheet.pdf

https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf

Export to PDF