## Universitatea Națională de Știință și Tehnologie POLITEHNICA București

Facultatea E.T.T.I.

Anul universitar 2023-2024

Sistem de monitorizare a locurilor de parcare

Proiect P2 2023-2024

Realizator:

Coordonator:

Prof. Dr. Ing. Vasile Mădălin Moise

* Balan Răzvan-Teodor

Cuprins

[Scopul 3](#_Toc134098698)

[Definirea obiectivelor și a cerințelor proiectului 3](#_Toc134098699)

[Definirea resursele preliminare 3](#_Toc134098700)

[Identificarea resurselor necesare 3](#_Toc134098701)

[Analiza nevoilor 3](#_Toc134098702)

[Elaborarea specificațiilor preliminare pentru software și hardware 4](#_Toc134098703)

[Materiale si echipamente necesare 4](#_Toc134098704)

[Dezvoltarea 4](#_Toc134098705)

[Examinare specificații preliminare software și hardware 4](#_Toc134098706)

[Elaborare specificații funcționale 4](#_Toc134098707)

[Dezvoltare prototip hardware 5](#_Toc134098708)

[Dezvoltare software 5](#_Toc134098709)

[Testarea și depanarea 5](#_Toc134098710)

[Elaborare plan de testare unitară folosind specificațiile produsului 5](#_Toc134098711)

[Testare module componente conform specificațiilor produsului 6](#_Toc134098712)

[Documente utile 6](#_Toc134098713)

[Elaborare manual de utilizare 6](#_Toc134098714)

[Documentarea Schemei electronice 6](#_Toc134098715)

[Documentarea placi de circuit imprimat 6](#_Toc134098716)

[Descrierea softului, instrucțiuni de asamblare și testare 7](#_Toc134098717)

# 

# Scopul

Proiectul nostru vizează implementarea unui sistem eficient de monitorizare a locurilor de parcare pentru a îmbunătăți gestionarea și utilizarea acestora. Prin acest proiect, ne propunem să reducem timpul petrecut în căutarea unui loc de parcare, să optimizăm ocuparea spațiului disponibil și să creștem satisfacția clienților

## Definirea obiectivelor și a cerințelor proiectului

1. **Optimizarea Utilizării Spațiului:** Asigurarea unei utilizări eficiente a locurilor de parcare disponibile pentru a maximiza capacitatea și a reduce congestia.
2. **Reducerea Timpului de Căutare a Locurilor de Parcare:** Furnizarea unei soluții care să permită șoferilor să găsească rapid locuri de parcare disponibile, reducând astfel timpul petrecut în căutare.
3. **Îmbunătățirea Experienței Utilizatorilor:** Crearea unei experiențe plăcute și eficiente pentru utilizatorii parcării, oferindu-le acces ușor la informații despre locurile de parcare disponibile.
4. **Monitorizarea și Gestionarea Parcării:** Furnizarea de date și rapoarte despre utilizarea parcării pentru a permite o gestionare mai eficientă a resurselor și pentru a lua decizii informate în ceea ce privește îmbunătățirea infrastructurii de parcare.
5. **Securitatea și Siguranța:** Asigurarea că sistemul de monitorizare oferă o securitate adecvată pentru datele și informațiile utilizatorilor și că ajută la prevenirea incidentelor de securitate în parcări.

## Definirea resursele preliminare

**Buget:**

- Bugetul pe care l-am avut si folosit in realizarea acestui proiect este de 20 lei.

**Personal:**

- Echipa proiectului este formată din 2 studenți din anul 3 dornici de descoperirea și perfecționarea unor noi abilități ce pot fi obținute in cadrul acestui proiect.

- Fiecare student contribuie în mod egal la realizarea obiectivelor proiectului, fie în dezvoltarea hardware, fie în dezvoltarea software. Ambii studenți completându-se unul pe celalalt în task-urile pentru realizarea acestui proiect

**Spațiu de Lucru:**

- Acest proiect dispune de un spațiu de lucru dedicat, echipat corespunzător pentru dezvoltarea hardware-ului și software-ului, astfel încât realizarea lui și punerea sa în funcțiune să se desfășoare în conformitate cu cerințele.

**Timp:**

- Estimăm că proiectul va dura aproximativ – saptamani (inclusiv toate etapele de cercetare, proiectare, dezvoltare, testare și implementare)

# Identificarea resurselor necesare

* Senzori de proximitate
* LED-uri
* Modul de Comunicație (Cablu USB)
* Componente de Montaj (fire, conectori)
* Alimentare (USB)
* Eventuale alte componente (rezistențe)

## Analiza nevoilor

Sistemul de monitorizare a locurilor de parcare trebuie să îndeplinească o serie de cerințe pentru a asigura funcționarea corectă și satisfacerea obiectivelor proiectului. Mai jos sunt detaliile cerințelor și necesităților sistemului:

* **Senzori de distanță:**

Senzorii de proximitate vor fi calibrați pentru a măsura distanța între vehicul și senzor, determinând astfel dacă locul de parcare este ocupat sau liber.

* **Interfețe de utilizator:**

Informațiile despre disponibilitatea locurilor de parcare vor fi afișate utilizatorilor prin intermediul LED-urilor. LED-ul de culoare verde semnificând că locul de parcare nu este ocupat, iar LED-ul de culoare roșie indică faptul că locul de parcare nu mai este ocupat.

* **Alimentare:**

Componentele sistemului vor fi alimentate prin intermediul unui cablu de alimentare USB conectat la o sursă de alimentare stabilă.

## Elaborarea specificațiilor preliminare pentru software și hardware

**Specificații preliminare pentru hardware:**

* **Senzori de distanță:**

*Tip:* Senzori de proximitate pentru detectarea prezenței vehiculelor în locurile de parcare. Senzorul de proximitate cuprinde toți senzorii care execută detecția fără contact în comparație cu întrerupătoarele de limitare care observă obiectele prin contactul fizic cu acestea. Aceste tipuri de senzori schimbă datele într-un semnal electric asupra mișcării sau apariției unei ținte.  Distanța maximă detectată de acest senzor se numește interval normal. Acest tip de senzor produce un câmp electromagnetic, altfel utilizează radiația IR pentru a genera un fascicul de lumină.

*Distanța de măsurare:* În intervalul specificat pentru a detecta prezența vehiculelor.

* **Interfețe de afișare:**

*Tip:* LED-urile reprezintă interfețe adecvate pentru afișarea stării locurilor de parcare. (Verde – loc liber / Roșu – loc ocupat)

*Metode de afișare:* Indicatori vizuali pentru locuri de parcare ocupate și libere.

* **Alimentare:**

Tensiune de alimentare: Specificațiile de tensiune necesare pentru componentele hardware.

**Specificații preliminare pentru software:**

* **Funcționalități:**

Afișarea stării locurilor de parcare (ocupate/libere).

Actualizare a informațiilor despre locurile de parcare.

**Algoritmi de monitorizare:**

Algoritmi pentru interpretarea datelor de la senzori și determinarea stării locurilor de parcare.

Algoritmi pentru gestionarea și actualizarea datelor despre locurile de parcare disponibile.

## Materiale si echipamente necesare

* Senzori de proximitate:

Cantitate: Două locuri de parcare.

Senzori ultrasonici de înaltă precizie conform specificațiilor preliminare.

* Interfețe de afișare:

Cantitate: Numărul de locuri de parcare sau de zone de afișare.

LED-uri conform specificațiilor preliminare.

* Modul de comunicare:

Modul USB conform specificațiilor preliminare.

* Alimentare:

Adaptor USB pentru fiecare dispozitiv.

Cabluri de alimentare USB.

Componente de montaj:

Cabluri pentru conectarea senzorilor și a interfețelor de afișare.

# Dezvoltarea

Dezvoltarea sistemului de monitorizare al locurilor de parcare a fost realizată printr-o abordare practică, utilizând componente hardware simple și soluții software eficiente. Sistemul include doi senzori de proximitate, LED-uri pentru semnalizare, fire de conectare și o sursă de alimentare.

* **Tehnologii și componente utilizate:**

***Senzori de proximitate:*** Acești senzori sunt responsabili pentru detectarea prezenței sau absenței vehiculelor în locurile de parcare.

***LED-uri:*** LED-urile sunt utilizate pentru a indica vizual starea locurilor de parcare (ocupat/liber).

***Fire de conectare:*** Firele sunt utilizate pentru a conecta senzorii și LED-urile la sursa de alimentare și la unitatea de control.

***Sursa de alimentare:*** Asigură energia necesară funcționării senzorilor și LED-urilor.

* **Etapele dezvoltării:**

***Proiectarea sistemului:*** Am început prin proiectarea schematică a sistemului, stabilind modul în care senzorii, LED-urile și sursa de alimentare vor fi conectate.

***Asamblarea hardware***: Senzorii de proximitate au fost montați și conectați la LED-uri și la sursa de alimentare utilizând fire de conectare.

***Programarea:*** Am dezvoltat codul necesar pentru a citi datele de la senzori și a controla LED-urile corespunzător. Programarea a fost realizată pe un microcontroller simplu (de exemplu, Arduino).

***Testarea:*** Sistemul a fost testat în condiții controlate pentru a verifica funcționarea corectă a senzorilor și a LED-urilor. Am realizat ajustări pentru a optimiza acuratețea și răspunsul sistemului.

***Implementarea:*** Sistemul a fost instalat în locurile de parcare desemnate și testat în condiții reale pentru a asigura fiabilitatea și eficiența în utilizare.

Dezvoltarea acestui sistem a fost orientată spre simplitate și eficiență, utilizând componente accesibile și tehnici de asamblare ușor de implementat. Echipa a colaborat îndeaproape pentru a asigura că toate părțile componente funcționează împreună armonios, rezultând într-un sistem fiabil și ușor de utilizat pentru monitorizarea locurilor de parcare.

## Examinare specificații preliminare software și hardware

* **Hardware:**

1. **Senzori de proximitate:**
   * **Model:** Modul Senzor PIR HC-SR501 (Senzor de mișcare)
   * **Gama de detectare:** 7 m
   * **Tensiunea de operare:** 3.3 V
   * **Interfață de conectare:** Analogică
2. **LED-uri:**
   * **Tip:** LED-uri de înaltă luminozitate
   * **Culoare:** Roșu (ocupat) și verde (liber)
   * **Tensiunea de operare:** 3.2-3.4 V
3. **Microcontroller:**
   * **Model:** ATmega164-DIL40
4. **Sursa de alimentare:**
   * **Tip:** Adaptor TTL-USB

**Software:**

1. **Platformă de dezvoltare:**
   * **Mediu de dezvoltare integrat :** CodeVision AVR
   * **Limbaj de programare:** C
2. **Funcționalități software:**
   * **Detectarea prezenței vehiculelor:** Citirea datelor de la senzori de proximitate pentru a determina ocuparea locurilor de parcare.
   * **Controlul LED-urilor:** Activarea LED-urilor corespunzătoare (roșu sau verde) pe baza datelor primite de la senzori.
   * **Diagnosticare și calibrare:** Funcții pentru diagnosticarea senzorilor și calibrarea acestora pentru a asigura acuratețea detectării.
3. **Interfață utilizator:**
   * **Tip:** Interfață fizică bazată pe LED-uri
   * **Indicatori:** LED roșu pentru loc ocupat, LED verde pentru loc liber

**Cerințe de performanță**

1. **Timp de răspuns:** Sistemul trebuie să răspundă la modificările stării locurilor de parcare în mai puțin de 7 secunde.
2. **Precizie:** Sistemul trebuie să aibă o rată de precizie de cel puțin 95% în detectarea ocupării locurilor de parcare.
3. **Durabilitate:** Componentele hardware trebuie să reziste la condiții de operare externe, inclusiv variații de temperatură și umiditate.

**Conectivitate**

1. **Tip:** Conexiuni cablate între senzori, LED-uri și microcontroller.
2. **Lungimea maximă a cablurilor:** 10 cm.

Aceste specificații preliminare stabilesc baza pentru dezvoltarea și implementarea sistemului de monitorizare al locurilor de parcare, asigurându-se că toate componentele hardware și software sunt compatibile și funcționează conform cerințelor stabilite.

## Elaborare specificații funcționale

Specificarea funcțiilor esențiale ale sistemului de monitorizare al locurilor de parcare este crucială pentru a asigura că toate cerințele utilizatorilor sunt îndeplinite eficient. Sistemul trebuie să fie capabil să detecteze, să monitorizeze și să semnalizeze starea locurilor de parcare în timp real. Specificațiile funcționale detaliate sunt următoarele:

**1. Detectarea ocupării locurilor de parcare:**

* ***Funcționalitate:*** Sistemul trebuie să detecteze prezența sau absența unui vehicul în fiecare loc de parcare utilizând senzori de proximitate.
* ***Condiții de activare:*** Senzorii trebuie să fie activi în mod continuu pentru a asigura monitorizarea în timp real.
* ***Rata de detecție:*** Sistemul trebuie să proceseze și să actualizeze starea locurilor de parcare în mai puțin de 1 secundă după orice schimbare detectată.

**2. Semnalizarea stării locurilor de parcare:**

* ***Funcționalitate:*** LED-urile vor indica vizual starea fiecărui loc de parcare. LED-urile roșii vor semnala că locul este ocupat, iar LED-urile verzi vor indica că locul este liber.
* ***Condiții de activare***: LED-urile trebuie să se aprindă sau să se stingă imediat ce senzorul detectează o schimbare în starea locului de parcare.
* ***Vizibilitate:*** LED-urile trebuie să fie vizibile dintr-o distanță de cel puțin 10 metri, atât ziua cât și noaptea.

**3. Calibrarea și diagnosticarea sistemului:**

* **Funcționalitate:** Sistemul trebuie să includă funcții pentru calibrarea senzorilor și diagnosticarea problemelor pentru a asigura acuratețea și fiabilitatea.
* ***Condiții de activare:*** Funcțiile de calibrare și diagnosticare trebuie să poată fi activate manual de către tehnicieni printr-o interfață dedicată.

**4. Alimentarea sistemului:**

* ***Funcționalitate:*** Sursa de alimentare trebuie să furnizeze energie constantă și suficientă pentru a asigura funcționarea continuă a senzorilor și a LED-urilor.
* ***Condiții de funcționare***: Sistemul trebuie să fie capabil să funcționeze în condiții variate de temperatură și umiditate, tipice unui mediu de parcare în aer liber.

**5. Monitorizarea și raportarea stării sistemului:**

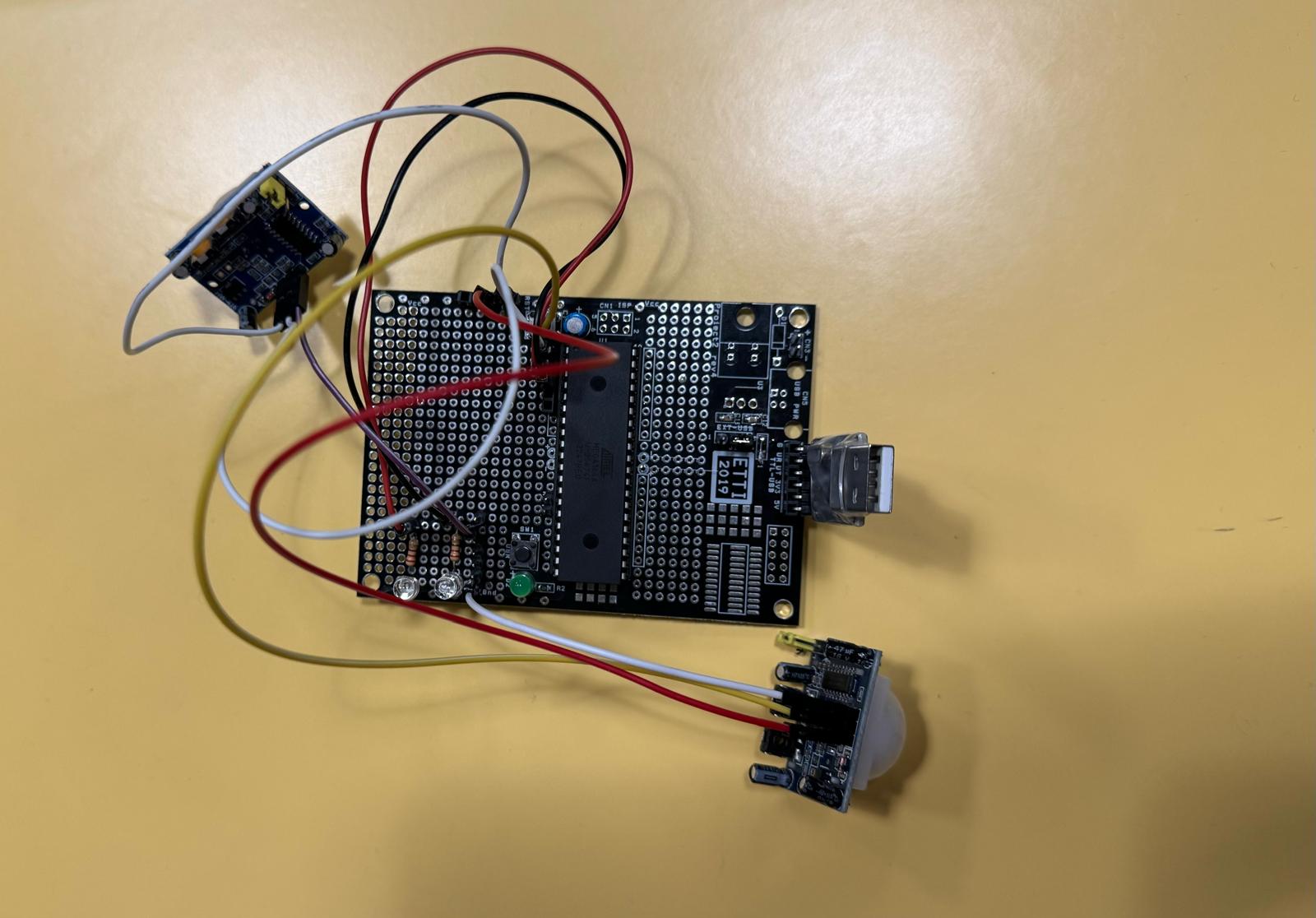
* ***Funcționalitate:*** Sistemul trebuie să fie capabil să raporteze starea fiecărui loc de parcare către o unitate centrală pentru monitorizare și analiză.
* ***Condiții de funcționare:*** Raportarea trebuie să fie în timp real, cu actualizări automate la fiecare schimbare de stare.

**6. Scalabilitatea sistemului:**

* ***Funcționalitate:*** Sistemul trebuie să fie proiectat astfel încât să permită adăugarea ușoară a mai multor locuri de parcare, fără a necesita modificări majore în infrastructura existentă.
* ***Condiții de implementare:*** Adăugarea de noi senzori și LED-uri trebuie să fie simplă și să nu interfereze cu funcționarea locurilor de parcare deja monitorizate.

Aceste specificații funcționale asigură că sistemul de monitorizare al locurilor de parcare va satisface nevoile utilizatorilor, va fi fiabil și eficient și va putea fi extins pe măsură ce cerințele se schimbă sau cresc.

## Dezvoltare prototip hardware

**Fig. 1** reprezintă partea superioară a PCB-ului ce conține următoarele component:

* 2 senzori de proximitate (PIR1, PIR2) amplasați în colțuri diferite;
* 2 LED-uri, alături de 2 rezintențe folosite pentru prevenirea supraîncărcării LED-urilor si a arderii lor;
* Fire de conexiune de la leduri la microcontroller + alimentări și masă, și fire de la PIR1, PIR2 la microcontroller + alimentări și masa.

Dezvoltarea prototipului hardware pentru sistemul de monitorizare al locurilor de parcare a implicat mai multe etape esențiale, de la selecția componentelor adecvate până la asamblarea și testarea inițială.

**1. Selectarea componentelor:**

* **Senzori de proximitate:** Am selectat senzori de proximitate cu o gamă de detectare adecvată (0-7 m) și cu o tensiune de operare de 3.3V DC. Senzorii aleși au fost suficient de sensibili pentru a detecta prezența vehiculelor.
* **LED-uri:** Am optat pentru LED-uri de înaltă luminozitate, roșii și verzi, pentru a semnaliza starea ocupată sau liberă a locurilor de parcare. LED-urile au fost selectate pentru vizibilitatea lor ridicată atât pe timp de zi, cât și noaptea.
* **Microcontroller:** Am utilizat un microcontroller ATmega164-DIL40 datorită flexibilității și ușurinței în programare. Acesta dispune de suficiente porturi I/O pentru a gestiona senzorii și LED-urile necesare.
* **Sursa de alimentare:** Am ales o sursă de alimentare TLL-USB cu o ieșire de 5V DC și un curent de 1A, pentru a asigura funcționarea continuă și stabilă a întregului sistem.

**2. Asamblarea prototipului:**

* **Montarea senzorilor și LED-urilor:** Senzorii de proximitate și LED-urile au fost montați pe o placă de test, conectându-le la microcontroller prin intermediul firelor de conectare. Configurația a fost aleasă pentru a permite modificări și ajustări ușoare în timpul testării.
* **Conectarea la microcontroller:** Senzorii și LED-urile au fost conectați la pinii corespunzători ai microcontroller-ului ATmega164-DIL40. Codul de control a fost scris pentru a citi datele de la senzori și pentru a controla starea LED-urilor.

**3. Programarea și testarea inițială:**

* **Scrierea codului:** Am dezvoltat un program simplu în CodeVision AVR pentru a testa funcționalitatea senzorilor de proximitate și a LED-urilor. Codul a fost scris în C și a inclus logica necesară pentru a aprinde LED-urile roșii sau verzi în funcție de datele primite de la senzori.
* **Testarea funcționalității:** Prototipul a fost testat într-un mediu controlat pentru a verifica acuratețea detecției și răspunsul rapid al LED-urilor. Testele au confirmat că senzorii detectează corect prezența vehiculelor și că LED-urile se aprind corespunzător.

**4. Ajustări și îmbunătățiri:**

* **Calibrarea senzorilor:** Am realizat ajustări pentru a calibra senzorii de proximitate, asigurându-ne că aceștia oferă date precise în toate condițiile de operare.
* **Optimizarea codului:** Codul a fost optimizat pentru a reduce latența și pentru a îmbunătăți eficiența energetică a sistemului.

Prin acest proces de dezvoltare a prototipului hardware, am reușit să creăm o bază solidă pentru sistemul de monitorizare al locurilor de parcare, asigurându-ne că toate componentele funcționează armonios și că sistemul este pregătit pentru implementare și testare în condiții reale.

## Dezvoltare software

* **Main.C :**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Project : Test software

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Chip type: ATmega164A

Clock frequency: 20 MHz

Compilers: CVAVR 2.x

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <mega164a.h>

#include <stdio.h>

#include <delay.h>

#include <string.h>

#include <stdlib.h>

#include "defs.h"

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\* BEGIN SERIAL STUFF (interrupt-driven, generated by Code Wizard) \*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#ifndef RXB8

#define RXB8 1

#endif

#ifndef TXB8

#define TXB8 0

#endif

#ifndef UPE

#define UPE 2

#endif

#ifndef DOR

#define DOR 3

#endif

#ifndef FE

#define FE 4

#endif

#ifndef UDRE

#define UDRE 5

#endif

#ifndef RXC

#define RXC 7

#endif

#define FRAMING\_ERROR (1<<FE)

#define PARITY\_ERROR (1<<UPE)

#define DATA\_OVERRUN (1<<DOR)

#define DATA\_REGISTER\_EMPTY (1<<UDRE)

#define RX\_COMPLETE (1<<RXC)

// USART0 Receiver buffer

#define RX\_BUFFER\_SIZE0 8

char rx\_buffer0[RX\_BUFFER\_SIZE0];

#if RX\_BUFFER\_SIZE0 <= 256

unsigned char rx\_wr\_index0,rx\_rd\_index0,rx\_counter0;

#else

unsigned int rx\_wr\_index0,rx\_rd\_index0,rx\_counter0;

#endif

// This flag is set on USART0 Receiver buffer overflow

bit rx\_buffer\_overflow0;

// USART0 Receiver interrupt service routine

interrupt [USART0\_RXC] void usart0\_rx\_isr(void)

{

char status,data;

status=UCSR0A;

data=UDR0;

if ((status & (FRAMING\_ERROR | PARITY\_ERROR | DATA\_OVERRUN))==0)

{

rx\_buffer0[rx\_wr\_index0++]=data;

#if RX\_BUFFER\_SIZE0 == 256

// special case for receiver buffer size=256

if (++rx\_counter0 == 0) rx\_buffer\_overflow0=1;

#else

if (rx\_wr\_index0 == RX\_BUFFER\_SIZE0) rx\_wr\_index0=0;

if (++rx\_counter0 == RX\_BUFFER\_SIZE0)

{

rx\_counter0=0;

rx\_buffer\_overflow0=1;

}

#endif

}

}

#ifndef DEBUG\_TERMINAL\_IO

// Get a character from the USART0 Receiver buffer

#define ALTERNATE\_GETCHAR

#pragma used+

char getchar(void)

{

char data;

while (rx\_counter0==0);

data=rx\_buffer0[rx\_rd\_index0++];

#if RX\_BUFFER\_SIZE0 != 256

if (rx\_rd\_index0 == RX\_BUFFER\_SIZE0) rx\_rd\_index0=0;

#endif

#asm("cli")

--rx\_counter0;

#asm("sei")

return data;

}

#pragma used-

#endif

// USART0 Transmitter buffer

#define TX\_BUFFER\_SIZE0 8

char tx\_buffer0[TX\_BUFFER\_SIZE0];

#if TX\_BUFFER\_SIZE0 <= 256

unsigned char tx\_wr\_index0,tx\_rd\_index0,tx\_counter0;

#else

unsigned int tx\_wr\_index0,tx\_rd\_index0,tx\_counter0;

#endif

// USART0 Transmitter interrupt service routine

interrupt [USART0\_TXC] void usart0\_tx\_isr(void)

{

if (tx\_counter0)

{

--tx\_counter0;

UDR0=tx\_buffer0[tx\_rd\_index0++];

#if TX\_BUFFER\_SIZE0 != 256

if (tx\_rd\_index0 == TX\_BUFFER\_SIZE0) tx\_rd\_index0=0;

#endif

}

}

#ifndef DEBUG\_TERMINAL\_IO

// Write a character to the USART0 Transmitter buffer

#define ALTERNATE\_PUTCHAR

#pragma used+

void putchar(char c)

{

while (tx\_counter0 == TX\_BUFFER\_SIZE0);

#asm("cli")

if (tx\_counter0 || ((UCSR0A & DATA\_REGISTER\_EMPTY)==0))

{

tx\_buffer0[tx\_wr\_index0++]=c;

#if TX\_BUFFER\_SIZE0 != 256

if (tx\_wr\_index0 == TX\_BUFFER\_SIZE0) tx\_wr\_index0=0;

#endif

++tx\_counter0;

}

else

UDR0=c;

#asm("sei")

}

#pragma used-

#endif

/\*

\* Timer 1 Output Compare A interrupt is used to blink LED

\*/

interrupt [TIM1\_COMPA] void timer1\_compa\_isr(void)

{

LED1 = ~LED1; // invert LED

}

/\*

\* main function of program

\*/

void debounce(void) {

delay\_ms(20); // Debounce de 20ms

}

unsigned short nr\_max = 2;

short locuri\_libere = 2;

void main (void)

{

Init\_initController(); // this must be the first "init" action/call!

#asm("sei") // enable interrupts

// initial state, will be changed by timer 1

green = 1;

red = 0;

while (TRUE) {

if (pir1 == 1) { // Senzorul PIR1 detecteaza miscare

debounce(); // Debounce pentru a evita detectii multiple

if (pir1 == 1) { // Verifica din nou după debounce

locuri\_libere--;

if (locuri\_libere < 0) locuri\_libere = 0;

while (pir1 == 1); // Asteapta ca senzorul sa revină la 0

delay\_ms(500); // Delay pentru stabilizare

}

}

if (pir2 == 1) { // Senzorul PIR2 detectează miscare

debounce(); // Debounce pentru a evita detectii multiple

if (pir2 == 1) { // Verifica din nou dupa debounce

locuri\_libere++;

if (locuri\_libere > nr\_max) locuri\_libere = nr\_max;

while (pir2 == 1); // Asteapta ca senzorul sa revina la 0

delay\_ms(500); // Delay pentru stabilizare

}

}

if (locuri\_libere <= 0) {

locuri\_libere = 0;

printf("PARCARE OCUPATA!!\n");

red = 1;

green = 0;

} else {

printf("PARCARE LIBERA!!\n");

red = 0;

green = 1;

}

printf("Nr. locuri ramase %d\n", locuri\_libere);

delay\_ms(1000); // Delay pentru actualizarea statusului

}

}// end main

# Testarea și depanarea

### **Testarea**

1. **Verificarea senzorilor PIR:**
   * Testează fiecare senzor PIR individual pentru a te asigura că detectează corect mișcarea. Poți folosi un LED de test conectat direct la ieșirea fiecărui senzor pentru a verifica activitatea acestora.
   * Verifică distanța și unghiul de detectare al senzorilor pentru a te asigura că sunt poziționați corect în raport cu locurile de parcare.
2. **Simularea scenariilor de utilizare:**
   * Simulează trecerea unei mașini prin fața PIR1 și PIR2 pentru a verifica dacă decrementarea și incrementarea locurilor libere funcționează corect.
   * Testează cum se comportă sistemul în diferite condiții, de exemplu, în cazul unui flux rapid de mașini sau al trecerii incomplete a mașinilor prin fața senzorilor.

### **Depanarea**

1. **Corectarea logicii de decrementare și incrementare:**
   * Asigură-te că senzorii PIR nu generează citiri false sau fluctuante. Poți adăuga un mic delay sau un filtru de software pentru a stabiliza citirile.
2. **Gestionarea limitelor locurilor libere:**
   * Verifică corectitudinea limitelor pentru locuri libere, astfel încât acestea să nu scadă sub zero sau să depășească numărul maxim de locuri de parcare.
3. **Îmbunătățirea codului:**
   * Modifică buclele **while** care așteaptă eliberarea senzorului PIR pentru a preveni blocarea codului. Înlocuiește bucla **while** cu o verificare condițională combinată cu un delay pentru a asigura citiri mai stabile.

## Elaborare plan de testare unitară folosind specificațiile produsului

#### **Obiectivul Testării**

Scopul testării unitare este de a verifica funcționarea corectă a fiecărei componente a sistemului de monitorizare a locurilor de parcare, care utilizează doi senzori de proximitate, două LED-uri, fire de conectare, un microcontroller ATmega164 și două rezistențe pentru LED-uri.

#### **Specificațiile Produsului**

1. **Senzori de proximitate**:
   * PIR1: detectează intrarea unei mașini.
   * PIR2: detectează ieșirea unei mașini.
2. **LED-uri**:
   * LED verde: indică locuri de parcare libere.
   * LED roșu: indică locuri de parcare ocupate.
3. **Microcontroller**:
   * ATmega164: gestionează logica sistemului și controlul LED-urilor.
4. **Fire de conectare**:
   * Conectează senzori, LED-uri și microcontroller.
5. **Rezistențe**:
   * Utilizate pentru a limita curentul prin LED-uri.

#### **Procedura de Testare Unitară**

##### **Testarea Senzorilor de Proximitate**

1. **Testul PIR1 (Senzor de intrare)**:
   * **Setup**: Conectează PIR1 la microcontroller.
   * **Acțiune**: Simulează trecerea unei mașini prin fața PIR1.
   * **Așteptări**: Verifică decrementarea variabilei **locuri\_libere** și rămânerea acesteia la valoarea zero dacă atinge limita inferioară.
2. **Testul PIR2 (Senzor de ieșire)**:
   * **Setup**: Conectează PIR2 la microcontroller.
   * **Acțiune**: Simulează ieșirea unei mașini prin fața PIR2.
   * **Așteptări**: Verifică incrementarea variabilei **locuri\_libere** și rămânerea acesteia la valoarea maximă dacă atinge limita superioară.

##### Testarea LED-urilor

1. **Testul LED verde**:
   * **Setup**: Conectează LED-ul verde la microcontroller prin intermediul unei rezistențe.
   * **Acțiune**: Setează condiția ca **locuri\_libere** să fie mai mare decât zero.
   * **Așteptări**: Verifică aprinderea LED-ului verde.
2. **Testul LED roșu**:
   * **Setup**: Conectează LED-ul roșu la microcontroller prin intermediul unei rezistențe.
   * **Acțiune**: Setează condiția ca **locuri\_libere** să fie zero.
   * **Așteptări**: Verifică aprinderea LED-ului roșu.

##### **Testarea Microcontroller-ului ATmega164**

1. **Testul inițializării sistemului**:
   * **Setup**: Conectează toate componentele conform schemei circuitului.
   * **Acțiune**: Inițializează sistemul și pornește microcontroller-ul.
   * **Așteptări**: LED-ul verde este aprins inițial indicând locuri de parcare libere.
2. **Testul decrementării și incrementării locurilor de parcare**:
   * **Setup**: Asigură-te că sistemul este inițializat corect.
   * **Acțiune**: Simulează trecerea unei mașini prin fața PIR1 și ieșirea acesteia prin fața PIR2.
   * **Așteptări**: Verifică corectitudinea decrementării și incrementării variabilei **locuri\_libere**.
3. **Testul limitării valorilor locurilor de parcare**:
   * **Setup**: Asigură-te că sistemul este inițializat corect.
   * **Acțiune**: Simulează repetat trecerea mașinilor prin fața PIR1 și PIR2 pentru a atinge și depăși limitele superioare și inferioare.
   * **Așteptări**: Verifică că **locuri\_libere** nu scade sub zero și nu crește peste valoarea maximă.

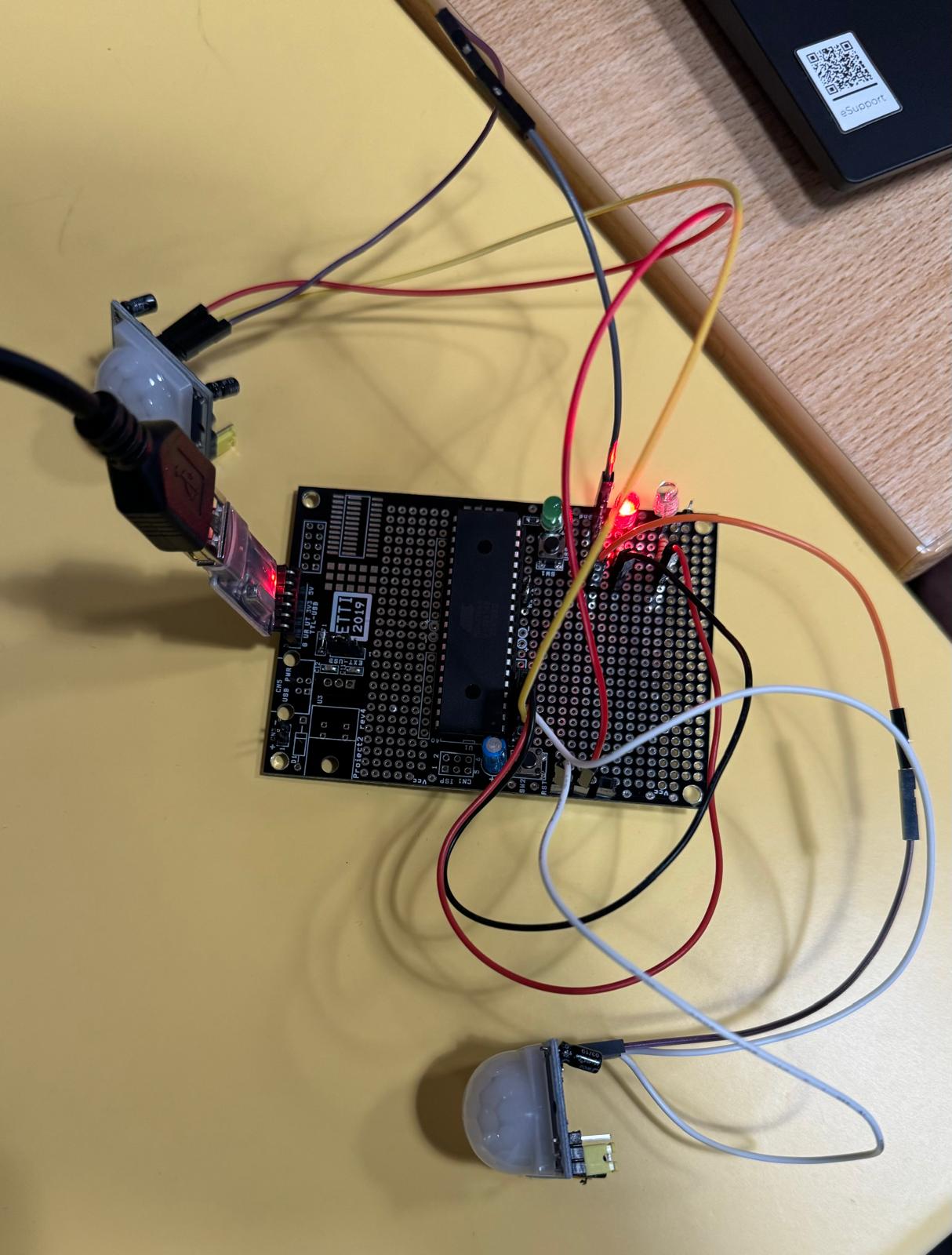
##### Testarea Integrată

1. **Testul funcționalității complete**:
   * **Setup**: Conectează toate componentele sistemului.
   * **Acțiune**: Simulează scenarii reale de parcare și eliberare a locurilor de parcare.
   * **Așteptări**: Verifică funcționarea corectă a întregului sistem, inclusiv schimbarea stării LED-urilor și actualizarea variabilei **locuri\_libere**.

#### Concluzii și Documentare

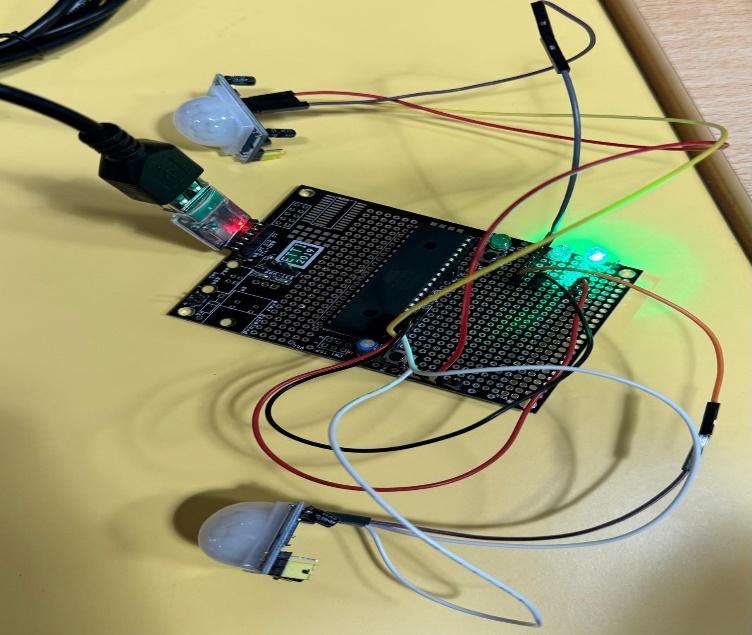
* Documentează fiecare test, incluzând setările inițiale, acțiunile efectuate, așteptările și rezultatele observate.
* Notează orice anomalii sau defecțiuni întâlnite și pașii pentru remedierea acestora.
* Asigură-te că toate componentele funcționează conform specificațiilor produsului înainte de a finaliza testarea unitară.

## Testare module componente conform specificațiilor produsului



**Fig. 2** *Circuitul alimentat si functional*

In **Fig. 2** se observă că atunci cand nu mai avem niciun loc de parcare liber, LED-ul corespunzator este de culoare roșie.



**Fig. 3** *Circuitul alimentat si functional*

In **Fig. 2** se observă că atunci cand avem un loc de parcare liber, LED-ul corespunzator este de culoare verde.

# Documente utile

1. **Specificațiile Produsului**
   * **Descriere**: Detaliază ce trebuie să facă sistemul și cum trebuie să funcționeze.
   * **Utilitate**: Ghid pentru a te asigura că proiectul respectă cerințele.
2. **Diagrama de Circuit**
   * **Descriere**: Schiță care arată cum sunt conectate componentele electrice.
   * **Utilitate**: Te ajută să conectezi corect senzorii, LED-urile și microcontroller-ul.
3. **Planul de Testare Unitară**
   * **Descriere**: Pașii și testele necesare pentru a verifica fiecare componentă a sistemului.
   * **Utilitate**: Asigură că fiecare parte a sistemului funcționează corect.
4. **Manualul Microcontroller-ului ATmega164**
   * **Descriere**: Informații despre cum să programezi și să folosești microcontroller-ul.
   * **Utilitate**: Ghid pentru programarea corectă a microcontroller-ului.
5. **Jurnalul de Testare**
   * **Descriere**: Înregistrarea tuturor testelor efectuate și rezultatele lor.
   * **Utilitate**: Documentează ce a funcționat și ce nu, pentru a îmbunătăți sistemul.

Aceste documente te vor ajuta să planifici, să implementezi și să testezi eficient sistemul de monitorizare al locurilor de parcare.

## Elaborare manual de utilizare

#### **Pornire**

1. **Alimentarea Sistemului**:
   * Conectați sistemul la un încărcător de 12V compatibil.
2. **Verificarea Indicatorilor Luminoși**:
   * Observați comportamentul LED-urilor: LED-ul verde ar trebui să fie luminat inițial, indicând locuri de parcare libere.
   * Atunci când toate locurile sunt ocupate, LED-ul roșu se va aprinde.

#### Siguranță și Întreținere

Pentru a asigura o utilizare sigură și durabilă a sistemului, urmați aceste recomandări:

a) **Protecția împotriva Factorilor Nocivi**:

* Evitați expunerea sistemului la umiditate, praf și șocuri fizice.

b) **Alegerea Corectă a Surselor de Alimentare**:

* Folosiți exclusiv un încărcător de 12V compatibil pentru a alimenta sistemul.

c) **Verificarea și Fixarea Conexiunilor**:

* Asigurați-vă că toate conexiunile și cablurile sunt conectate și izolate corespunzător pentru a evita deteriorările sau scurtcircuitările.

d) **Suport Tehnic și Depanare**:

* Pentru orice probleme sau întrebări legate de funcționarea sistemului, consultați secțiunea de suport tehnic a manualului sau contactați un specialist calificat.

#### **Contraindicații**

Pentru a evita deteriorarea și disfuncționalitatea sistemului, urmați aceste recomandări:

a) **Intervenții în Componentele Sistemului**:

* Nu încercați să demontați sau să reparați componentele sistemului de unul singur. Solicitați asistență de la personalul calificat pentru orice intervenție tehnică.

b) **Utilizarea Corectă a Tensiunii de Alimentare**:

* Nu aplicați tensiuni mai mari decât cele specificate (12V), deoarece acest lucru poate cauza daune ireversibile sistemului.

c) **Mediul de Utilizare**:

* Evitați utilizarea sistemului în medii cu temperaturi extreme sau umiditate ridicată, deoarece acest lucru poate afecta performanța și durabilitatea componentelor.

Respectând cu strictețe aceste recomandări, veți asigura o utilizare sigură și eficientă a sistemului de monitorizare al locurilor de parcare.

## Documentarea Schemei electronice

A picture containing text, diagram, plan, map

Description automatically generated

***Fig. 4****. Schema machetei*

## Documentarea placi de circuit imprimat

A close-up of a circuit board

Description automatically generated with medium confidence

A close-up of a circuit board

Description automatically generated with low confidence

***Fig. 5.*** *PCB: Top, Bottom, Silkscreen Top și vedere generala*

## Descrierea softului, instrucțiuni de asamblare și testare

#### **Descrierea Softului**

#### Softul este un sistem de monitorizare al locurilor de parcare, dezvoltat în limbajul de programare C utilizând mediul de dezvoltare CodeVision AVR. Acesta utilizează un microcontroller ATmega164 pentru a controla funcționarea sistemului. Principalele componente software includ:

1. **Gestionați funcționarea senzorilor**: Codul controlează citirea și interpretarea datelor provenite de la cei doi senzori de proximitate, responsabili de detectarea intrării și ieșirii mașinilor din locurile de parcare.
2. **Controlul LED-urilor**: LED-urile sunt utilizate pentru a indica starea locurilor de parcare (libere/ocupate). Softul controlează aprinderea și starea LED-urilor în funcție de informațiile primite de la senzori.
3. **Logica de decizie a sistemului**: Bazându-se pe datele primite de la senzori, softul decide când să comute între LED-ul verde (pentru locuri libere) și LED-ul roșu (pentru locuri ocupate).

#### **Instrucțiuni de Asamblare**

1. **Conectarea Componentelor**:
   * Conectați microcontroller-ul ATmega164 la senzorii de proximitate și la LED-uri utilizând cabluri de conexiune.
   * Asigurați-vă că fiecare LED este însoțit de o rezistență corespunzătoare pentru a preveni supraîncălzirea și arderea.
2. **Alimentare și Pornire**:
   * Alimentați sistemul cu un încărcător de 12V compatibil.
   * Porniți sistemul și observați comportamentul LED-urilor în funcție de informațiile primite de la senzori.

#### **Instrucțiuni de Testare**

1. **Verificarea Senzorilor**:
   * Aduceți obiecte în apropierea senzorilor pentru a simula intrarea și ieșirea mașinilor din locurile de parcare.
   * Verificați dacă LED-urile se aprind și se sting corect în funcție de detecțiile senzorilor.
2. **Testarea Logicii de Decizie**:
   * Simulați diverse scenarii de parcare și monitorizați comportamentul sistemului în funcție de starea locurilor de parcare.
   * Asigurați-vă că LED-urile se schimbă corect între verde și roșu în funcție de ocuparea locurilor.
3. **Testare Generală a Sistemului**:
   * Testați sistemul în diverse condiții de lumină și temperatură pentru a verifica funcționarea corectă a senzorilor și a LED-urilor.
   * Verificați stabilitatea și fiabilitatea sistemului în condiții diverse.

Urmând aceste instrucțiuni, veți asigura o asamblare corectă și o testare riguroasă a sistemului de monitorizare al locurilor de parcare, garantând funcționarea optimă a acestuia în mediu real.