UNIVERSITATEA POLITEHNICA BUCUREȘTI

FACULTATEA DE ELECTRONICĂ, TELECOMUNICAȚII ȘI TEHNOLOGIA INFORMAȚIEI

ROBOT MOBIL CU FUNCȚIE DE PARCARE AUTONOMĂ

# PROIECT DE DIPLOMĂ

Prezentat ca cerință parțială pentru obținerea titlului de Inginer în domeniul Electronică și Telecomunicații programul de studii: Microelectronică, Optoelectronică și Nanotehnologii

**Conducător științific:**

Conf. Dr. Ing. Horia CUCU

**Student:**

Marius-Cristian MAREȘ

București

2020

Universitatea "Politehnica" din Bucureşti
Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei
Departamentul **TEF**

**Anexa 1**

**TEMA PROIECTULUI DE DIPLOMĂ**

a studentului **MAREŞ S.V. Marius-Cristian , 441E-MON.**

**1. Titlul temei:** Robot mobil cu funcție de parcare autonomă

**2. Descrierea contribuţiei originale a studentului (în afara părţii de documentare) și specificații de proiectare:**
Proiectul are drept scop crearea unui robot mobil care să parcheaze autonom, într-o parcare inteligentă. Aplicațiile acestui robot pot fi extinse în domeniul autoturismelor, la ajutarea persoanelor cu handicap sau pur și simplu pentru o parcare mai eficientă.

Se va realiza robotul mobil cât și o zona special amenajată pentru parcarea acestuia. Cele două entități vor avea la bază câte un microcontroler prin intermediul căruia vor fi procesate datele provenite din mediul exterior, date preluate de către senzorii echipați pe robotul mobil cât și de la senzorii plasați în parcare. În urma procesării datelor preluate din parcare, masina va primi informații prin intermediul modulului nRF24L01
despre disponibilitatea locurilor de parcare și va efectua parcarea automata, daca acest lucru este posibil. Totdată, utilizatorului îi va fi oferită posibilitatea de a controla robotul mobil și manual, prin telecomandă.

**3. Resurse folosite la dezvoltarea proiectului:**
Ca și resurse necesare realizării proiectului voi folosi diferiți senzori( senzori de distanță, senzori de temperatură), trei module ce au la bază microcontrolere din familia ATMEGA, software de proiectare CAD necesar realizării PCB, iar limbajul de programare de bază utilizat este C.

**4. Proiectul se bazează pe cunoştinţe dobîndite în principal la următoarele 3-4 discipline:**
Microcontrolere, Circuite Electronice Fundamentale, Dispozitive electronice, Programarea Calculatoar

**5. Proprietatea intelectuală asupra proiectului aparţine:** U.P.B.

**6. Data înregistrării temei:** 2019-11-25 11:20:24

**Conducător(i) lucrare,**
Conf. dr. ing. Horia CUCU

semnătura: .....

**Student,**

semnătura:.....

**Director departament,**
Conf. dr. ing. Marian VLĂDESCU

semnătura:.....

**Decan,**
Prof. dr. ing. Cristian NEGRESCU

semnătura:.....

Cod Validare: **db8d94a1ae**

Anexa 5

## Declarație de onestitate academică

Prin prezenta declar că lucrarea cu titlul *„Robot mobil cu funcție de parcare autonomă"*, prezentată în cadrul Facultății de Electronică, Telecomunicații și Tehnologia Informației a Universității "Politehnica" din București ca cerință parțială pentru obținerea titlului de *Inginer* în domeniul *Inginerie electronică, telecomunicații și tehnologii informaționale*, programul de studii *Microelectronică, optoelectronică și nanotehnologii* este scrisă de mine și nu a mai fost prezentată niciodată la o facultate sau instituție de învățământ superior din țară sau străinătate.

Declar că toate sursele utilizate, inclusiv cele de pe Internet, sunt indicate în lucrare, ca referințe bibliografice. Fragmentele de text din alte surse, reproduse exact, chiar și în traducere proprie din altă limbă, sunt scrise între ghilimele și fac referință la sursă. Reformularea în cuvinte proprii a textelor scrise de către alți autori face referință la sursă. Înțeleg că plagiatul constituie infracțiune și se sancționează conform legilor în vigoare.

Declar că toate rezultatele simulărilor, experimentelor și măsurătorilor pe care le prezint ca fiind făcute de mine, precum și metodele prin care au fost obținute, sunt reale și provin din respectivele simulări, experimente și măsurători. Înțeleg că falsificarea datelor și rezultatelor constituie fraudă și se sancționează conform regulamentelor în vigoare.

București, *17.06.2020*

Absolvent *Marius-Cristian MAREȘ*

(semnătura în original)

# CUPRINS

# Listă de figuri

# Listă de tabele

# Listă de ecuații

# LISTĂ DE ACRONIME

A

AMR= Anisotropic Magnetoresistance Effect (Magnetorezistență anizotropă)

C

CS = Chip select
CE = Chip Enable

I

ISM = Industrial, Scientific and Medical bands
$I^2C$ = Inter-Integrated Circuit
IDE = Integrated Development Environment (Mediu de dezvoltare)

L

LED = Light-emitting Diode (Diodă electroluminiscentă)
LCD = Liquid Crystal Display (Afișaj cu cristale lichide)
LSB = Least significant bit (Bitul de semnificație minimă)

M

MOSI = Master Out/ Slave In
MISO = Master In/ Slave Out

P

PWM = Pulse Width Modulation (Modulația lățimii pulsului)

PCB = Printed Circuit Board (Placă cu cablaj imprimat)

R

RAS = Registru de apriximări succesive

S

SPI = Serial Peripheral Interface

SS = Slave select

SDA = Serial Data (Linie de date serială)

SCL = Serial Clock (Linie de ceas serială)

SRAM = Static random-access memory (Memorie statică cu acces aleator)

U

UART = Universal Asynchronous Receiver/Transmitter (Receptor-transmițător asincron universal)

USB = Universal Serial Bus (Magistrală Serială Universală)

# Capitolul 1 Introducere

## 1.1 Motivație

Automobilul reprezintă cel mai utilizat mijloc de transport, atât pentru deplasarea oamenilor, cât și a bunurilor. Acesta a reprezentat un pilon foarte important în procesul de dezvoltare al umanității, datorită avantajelor pe care le prezintă. Printre aceste avantaje se numără: posibilitatea de a deplasa bunuri sau persoane între două locații într-un mod sigur și rapid sau comoditatea pe care acest mijloc de transport o oferă. De-a lungul timpului, automobilul a suferit numeroase ajustări ce au dus la o îmbunătățire a calității acestuia, iar această evoluție continuă să existe și în zilele noastre. În prezent, direcția de dezvoltare se îndreaptă către producerea de automobile electrice, reducerea gradului de poluare, eficientizarea timpului petrecut în trafic, conducerea autonomă. Societatea Inginerilor de Autovehicule (SAE), a clasificat viitorul autovehiculelor în șase niveluri de automatizare. Aceasta clasificare se definește în figura 1.1.

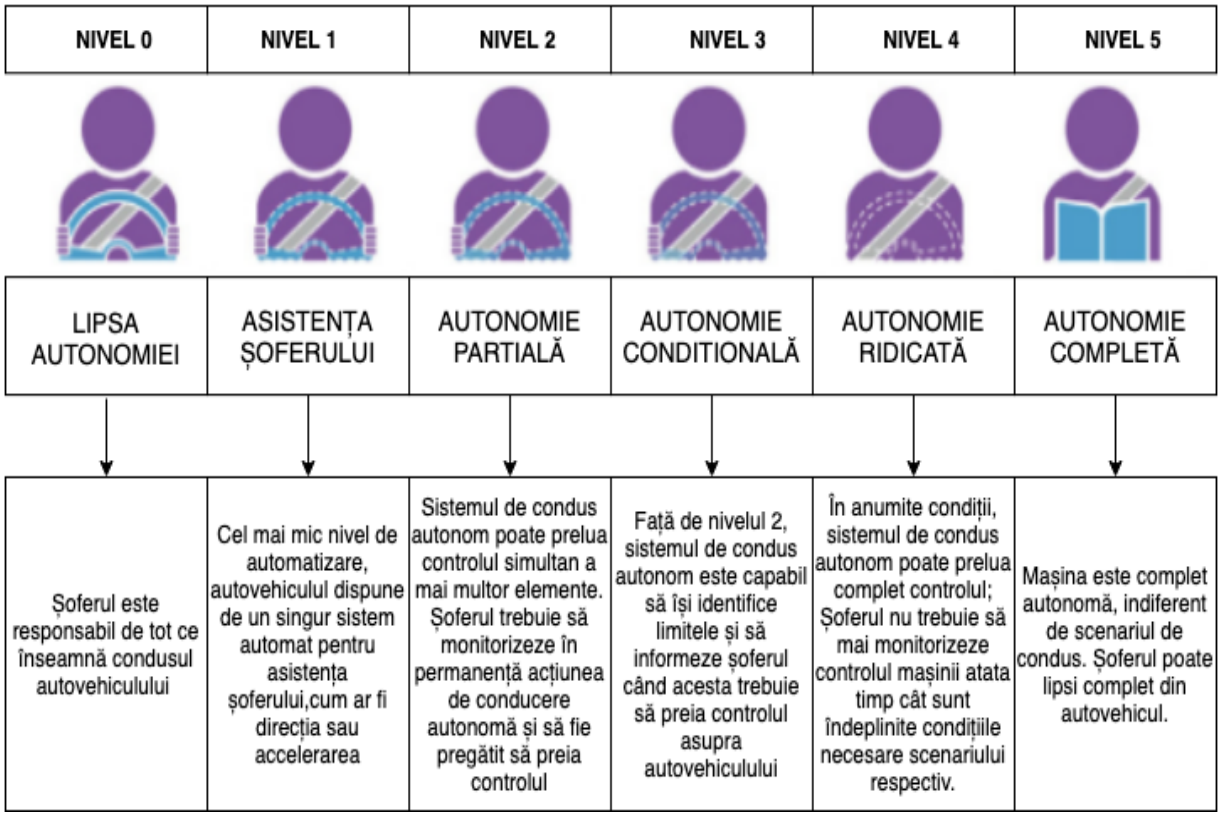



*Figura 1.1: Viitorul autovehiculelor în funcție de nivelul de automatizare[1]*

Cu o astfel de clasificare în minte, nivelul de dezvoltare este evident. În zilele noastre, accentul se pune pe ultimele două niveluri, mai exact, pe crearea unui sistem autonom de conducere a autovehiculului, sistem ce poate funcționa fără monitorizarea permanentă a șoferului, acesta putând lipsi complet din interiorul autovehiculului. De asemenea, odată cu implementarea sistemului de condus autonom, elementele clasice de control ale mașinii (volanul, pedalele de control, semnalizările) nu mai sunt necesare. Așadar, această direcție a reprezentat punctul central în jurul căruia a luat naștere motivația mea în alegerea prezentei lucrări.

## 1.2 Obiective

Având în vedere direcția de dezvoltare menționată la punctul anterior (vezi Figura 1.1), lucrarea prezentă are ca scop realizarea unui robot mobil care să parcheze autonom. Se va construi parcarea fizică împreună cu un sistem capabil să colecteze date de la mediul exterior, date privind disponibilitatea locurilor de parcare, și totodată, aceste date se vor transmite prin intermediul undelor radio către mașină. În urma recepționării datelor de la parcare, mașina va prelucra aceste date și va parca în zona identificată ca fiind liberă. Mașina va fi echipată cu o serie de senzori cu ajutorul cărora aceasta va prelua date de la mediul exterior, date ce vor fi utilizate pentru deplasarea corectă către locul de parcare.

Senzorii utilizați sunt: senzorul ultrasonic HC-SR04, utilizat pentru măsurarea distanțelor și pentru detecția unui potențial obstacol, senzor fotoelectric infraroșu cu ajutorul căruia s-a determinat numărul de rotații ale motorului, modulul nRF24L01 prin intermediul căruia se realizează comunicația și magnetometrul HMC5883L cu ajutorul căruia se monitorizează coordonatele inițiale și coordonatele finale în urma unui viraj. Așadar, pe baza datelor primite de la senzorii menționați anterior, mașina va realiza manevra de parcare fără intervenția utilizatorului. Conectarea senzorilor la microcontroler se va realiza utilizând o placă cu cablaj imprimat, proiectată în cadrul proiectului.

De asemenea, mașina va putea fi controlată de către utilizator prin intermediul unei telecomenzi. Modulul telecomenzii conține un LCD, pe care se vor afișa temperatura și umiditatea, date ce vor fi transmise de la mașină către telecomandă. Trecerea de la controlul manual la modul autonom de parcare se va realiza la comanda utilizatorului, prin acționarea unui anumit buton de pe telecomandă.

## 1.3 Structura lucrării

În cadrul Capitolului 2 se vor prezenta concepte teoretice privind componentele utilizate pentru realizarea robotului mobil. Se vor prezenta noțiunile de bază privind funcționalitatea acestora pentru a se oferi o imagine de ansamblu asupra utilității în cadrul proiectului.

În Capitolul 3 se vor prezenta conceptele teoretice privind componentele utilizate pentru realizarea parcării inteligente. Vor fi abordate noțiuni de bază privind funcționalitatea componentelor și rolul utilizării acestora în cadrul parcării.

Capitolul 4 este dedicat descrierii noțiunilor teoretice privind componentele utilizate pentru realizarea telecomenzii folosite pentru controlul robotului mobil.

În cadrul Capitolului 5 se vor prezenta detaliile implementării hardware. În subcapitolul 5.1 vor fi prezentate detaliile implementării hardware pentru robotul mobil, detalii privind schema electrică și detalii privind proiectarea circuitului cu cablaj imprimat utilizat pentru integrarea cât mai compactă a elementelor utilizate în cadrul mașinii. În subcapitolele 5.2 și 5.3 vor fi prezentate detaliile privind implementarea hardware pentru parcarea inteligentă, respectiv pentru telecomandă.

În Capitolul 6 se vor prezenta detalii privind implementarea software a celor trei programe utilizate pentru controlul robotului mobil, pentru controlul parcării inteligente, respectiv pentru controlul telecomenzii. Vor fi prezentate organigramele software ale codurilor sursă și se va explica funcționalitatea anumitor funcții implementate.

# Capitolul 2 Robot mobil - Noțiuni teoretice

## 2.1 Motor cu reductor și roată

Motoarele de curent continuu sunt dispozitive electromagnetice care transformă energia electrică în energie mecanică rotativă utilizând interacțiunea câmpurilor magnetice și a conductorilor și se utilizează într-o mulțime de aplicații precum: ventilatoare, roboți industriali, electrocasnice, roboți mobili de jucărie etc. Motorul de curent continuu este cel mai utilizat actuator (ansamblu ce convertește o formă de energie în energie mecanică) pentru producerea mișcărilor de rotație, iar prin controlul acestor mișcări, motoarele se dovedesc a fi indispensabile în aplicații în care controlul vitezei sau poziționarea sunt necesare. În cadrul proiectului, la realizarea mașinii, am utilizat patru motoare cu reducție de 1:48, la care am atașat patru roți cu diametrul de 65mm și anvelopă de cauciuc. Elementul de reducție este realizat din roți dințate realizate din materiale plastice, iar raportul de 1:48 semnifică faptul că 48 de rotații ale motorului sunt echivalente cu o singură rotație a axului extern la care este atașat roata. Utilizarea roților dințate are ca și avantaje reducerea zgomotului și mărirea forței de acționare, iar ca și dezavantaj se poate menționa faptul că acest ansamblu poate fi folosit doar în anumite condiții de temperatură, și totodată, utilizarea mecanismului de reducție reduce viteza de rotație a motorului. Ansamblul utilizat este ilustrat în figura 2.1:

*Figura 2.1: Motor cu reductor și roată; Sursa[2]*

Un motor de curent continuu este realizat dintr-o structură, ce constă, în principal, din două părți. Partea de la exterior a unui motor este prevăzută cu o carcasă metalică, ce reprezintă corpul staționar, numit stator. La un capăt al carcasei metalice se găsește un capac din plastic prevăzut cu două terminale, iar la celălalt capăt este prezent un ax metalic ce iese din carcasă, ax ce are rolul de a transfera energia mecanică produsă și pe care se pot atașa diferite elemente externe precum: roți, elemente de reducție realizate din diferite materiale, ventilatoare, etc.

În partea din interior, atașat axului metalic, se găsește un ansamblu care se rotește producând mișcarea, numit rotor și doi magneți permanenți de polaritate diferită ce produc un câmp magnetic puternic prin rotor. Rotorul este realizat dintr-un număr de discuri laminate în formă de T, în jurul cărora se găsesc conductori înfășurați ce poartă curentul electric de la baterie. Această înfășurătoare este conectată prin intermediul unui ansamblu de comutator cu perii, de aici

și denumirea, des întâlnită, de motor cu perii. În imaginea următoare sunt ilustrate elementele constructive ale unui motor, elemente ce au fost prezentate anterior:

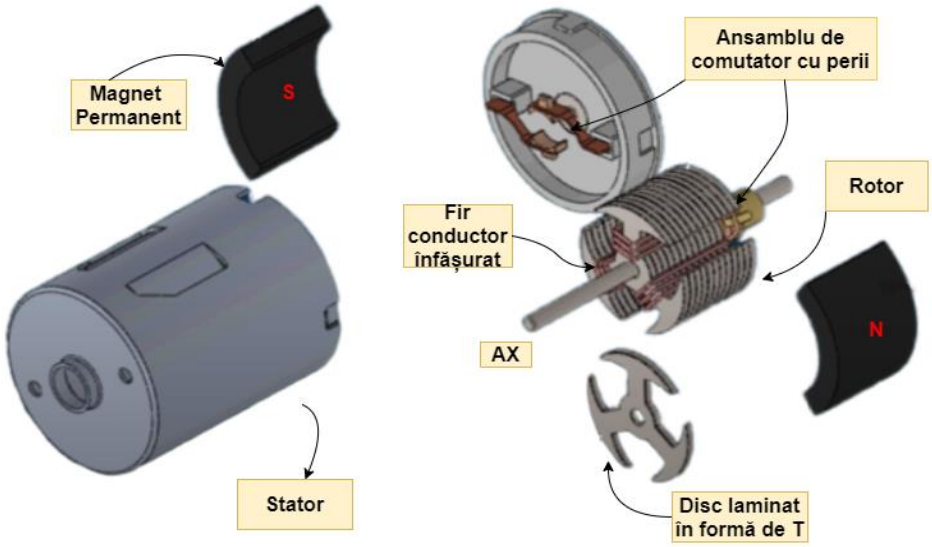



*Figura 2.2: Elementele constructive ale unui motor; Sursa [3]*

La trecerea curentului prin rotorul cu înfășurare, se creează un câmp electromagnetic, ce creează mișcarea de rotație a rotorului. Această mișcare de rotație are la bază următoarele principii. Într-un conductor de metal se găsesc atomi în jurul cărora se mișcă sarcinile electrice libere, numite electroni. Electronii sunt sarcini electrice negative ce se mișcă liber între atomii materialului de metal. Dacă la bornele conductorului de metal se aplică o diferență de potențial, de exemplu, firul conductor este legat la o baterie, electroni sunt puși în mișcare, aceștia circulând de la borna negativă la cea pozitivă prin intermediul conductorului. Astfel, prin circuit se stabilește un curent de sens opus, de la borna pozitivă la cea negativă. Totodată, la trecerea sarcinilor electrice printr-un conductor se generează câmpuri magnetice în jurul acestuia. Aceste câmpuri create au valori scăzute și prin intermediul lor nu se poate pune în mișcare rotorul. Astfel, pentru a intensifica acest efect și pentru a crea un câmp electromagnetic mai puternic, se înfășoară conductorul metalic în jurul acelor discuri laminate ce formează rotorul. Fiecare înfășurare creează un câmp electromagnetic și toate acestea se combină într-un câmp electromagnetic mai puternic, ce duce la generarea mișcării de rotație a rotorului din interiorul motorului.

Pentru a determina direcția de orientare a liniilor de câmp generate de trecerea electronilor prin înfășurarea rotorului, practic, pentru determinarea direcției de mișcare a rotorului, se folosește regula mâinii stângi. Această regula este importantă, deoarece cu ajutorul ei se poate determina în ce direcție se va deplasa rotorul atunci când câmpul magnetic produs de cei doi magneți permanenți va interacționa cu câmpul electromagnetic produs la trecerea electronilor prin conductorul metalic. În figura 2.3 este indicat un exemplu tipic în care este necesar utilizarea corectă a regulii mâinii stângi pentru determinarea direcției de rotație a motorului.

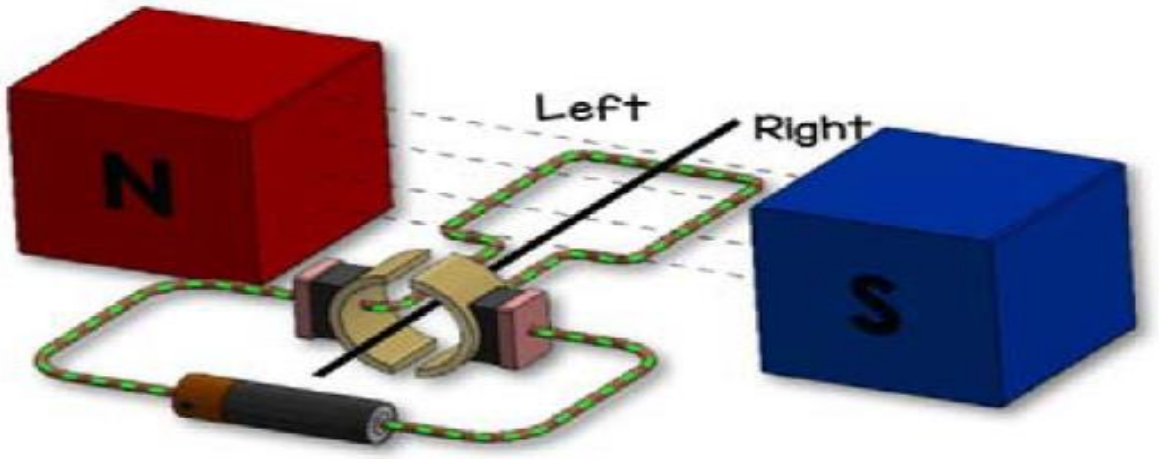



*Figura 2.3: Determinarea direcției de mișcare a motorului; Sursa[3]*

În acest exemplu, se aplică regula mâinii stângi pentru ambele părți ale firului conductor. Pentru partea din stânga, curentul prin conductor circulă de la baterie spre circuit(iese din baterie), direcția de curgere a curentului fiind indicată de degetul mijlociu, iar direcția câmpului magnetic este de la nord la sud. Astfel, direcția în care se va deplasa firul conductor va fi indicată de poziția degetului mare, care în acest caz va fi în jos. Această reprezentare este ilustrată în figura 2.4 a). Pentru partea dreapta, curentul circulă spre baterie (direcția indicată de către degetul mijlociu), iar direcția de curgere a câmpului magnetic rămâne aceeași (direcția indicată de către degetul arătător), de la nord la sud. Astfel, direcția forței de deplasare este indicată de către degetul mare, aceasta fiind ilustrată în figura 2.4 b).

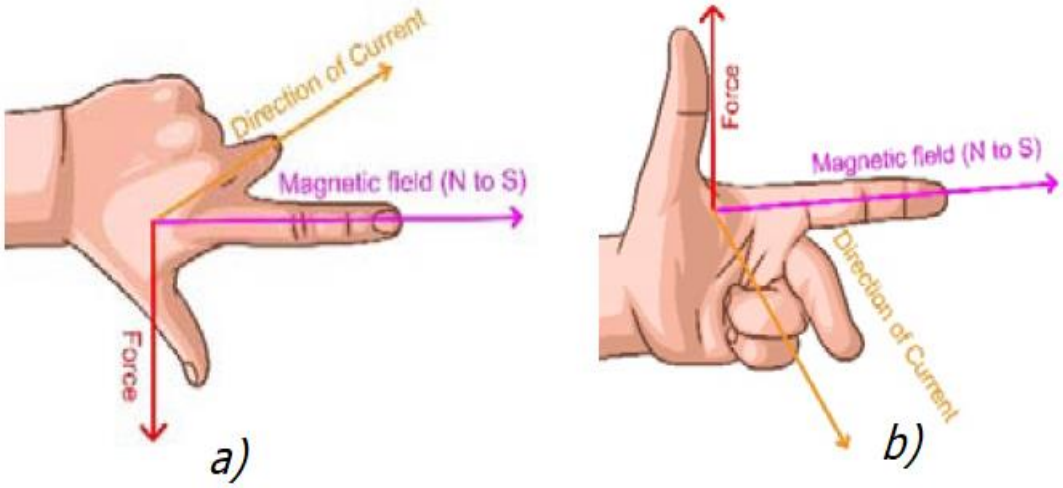



*Figura 2.4.: Regula mâinii stângi; Sursa [3]*

Aceste două forțe a căror direcție de deplasare au fost determinate anterior, se combină și vor genera mișcarea de rotație spre stânga a rotorului din interiorul motorului [3].

## 2.2 Modul cu Driver de Motoare Dual

Am menționat anterior, că motoarele de curent continuu sunt componente adecvate pentru includerea lor în aplicații precum roboții mobili. Totuși, pentru a crea un robot mobil ce se poate deplasa cu precizie, este nevoie de integrarea unui sistem de control al motorului. Motoarele de curent continuu pot fi acționate prin aplicarea unei diferențe de potențial la bornele acestora. Acest lucru va duce la mișcarea de rotație într-o anumită direcție, iar prin schimbarea polarității semnalului de la bornele motorului, mișcarea de rotație va fi inversată. Cel mai simplu mod prin care se poate controla acționarea unui motor este prezentat în figura 2.5.

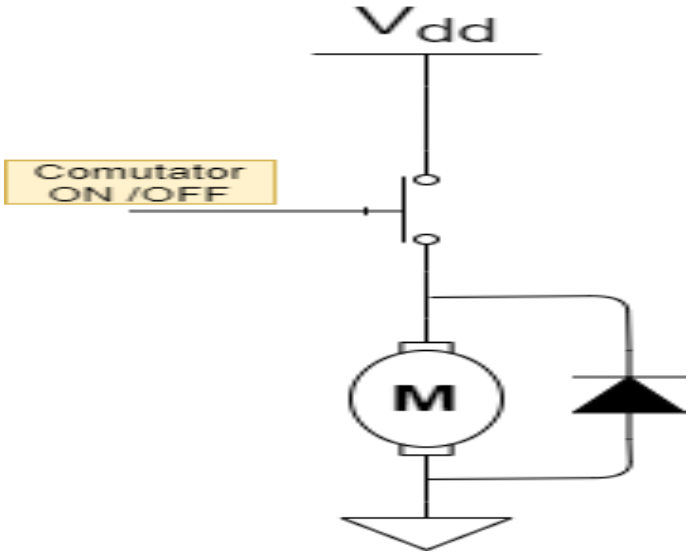



*Figura 2.5: Controlul unidirecțional al motorului*

În figura 2.5, motorul de curent continuu este conectat la sursa de alimentare prin intermediul unui comutator. Acest comutator este acționat, de obicei prin intermediul unui semnal de control de la microcontroler și în general, este constituit dintr-un circuit realizat din tranzistoare. De exemplu, acest comutator poate fi reprezentat de un singur tranzistor bipolar sau de un ansamblu de două tranzistoare aflate în configurație Darlington, atunci când motorul are nevoie de o valoare mai mare a curentului. În momentul când comutatorul a fost închis pentru o perioadă de timp, prin acesta va curge un curent I. Atunci când comutatorul se deschide, curentul prin motor nu va mai avea pe unde să circule, astfel că acesta se va descărca instantaneu la masă. Aceast lucru va genera un arc electric la bornele comutatorului, ceea ce poate produce daune componentelor electrice din circuit. Pentru a evita acest lucru, în circuitul din figura 2.5 se conectează o diodă în paralel cu motorul, astfel când comutatorul va fi deconectat, aceasta va oferi o cale prin care curentul va circula, evitând astfel crearea arcului electric, ceea ce duce la protejarea circuitului, iar când comutatorul este închis, dioda blochează curgerea curentului prin aceasta, neschimbând funcționalitatea motorului. Această diodă este denumită adesea, diodă de tip „flyback”.

Așadar, cu un astfel de circuit poate controla direcția de mișcare a motorului într-o singură direcție. Mai mult, prin comutarea continuă cu o anumită frecvență a comutatorului între cele două stări „ON/OFF”, putem varia viteza de rotație a motorului între limita inferioară (starea staționară) și limita superioară (viteza maximă). Acest mod de control este cunoscut sub numele de „Pulse

Width Modulation” (Modulația lățimii pulsului) și se bazează pe modificarea proporției de timp „ON” față de timpul „OFF”.

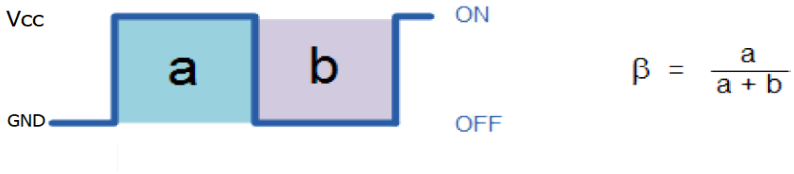



*Figura 2.6: Control PWM: Sursa [4]*

În figura 2.6, factorul β reprezintă factorul de umplere al semnalului, iar prin variația parametrilor a și b, corespunzători timpilor de „ON și OFF”, valoarea acestui factor de umplere se va modifica corespunzător relației ilustrate în figură. Dacă valoarea factorului de umplere este mare, durata în care comutatorul va fi închis este mai mare decât durata în care comutatorul va fi deschis, ceea ce va genera o viteză de rotație mare pentru motor. Daca valoarea factorului de umplere este mică, motorul va fi pentru o scurtă perioadă de timp pornit. Astfel, prin utilizarea semnalului de tip PWM, se poate controla cu precizie viteza de rotație a motorului.

Totuși, acest mod de control unidirecțional al motorului nu este suficient pentru aplicația propusă în cadrul acestui proiect. În cadrul proiectului, robotul mobil realizat trebuie să fie capabil să se deplaseze în toate direcțiile, ceea ce presupune posibilitatea controlului bidirecțional al motoarelor utilizate. Pentru a putea realiza controlul mișcării de rotație în ambele direcții ale unui motor de curent continuu se utilizează adesea un circuit numit circuit de control „în punte H”. Un astfel de circuit este ilustrat în figura 2.7:

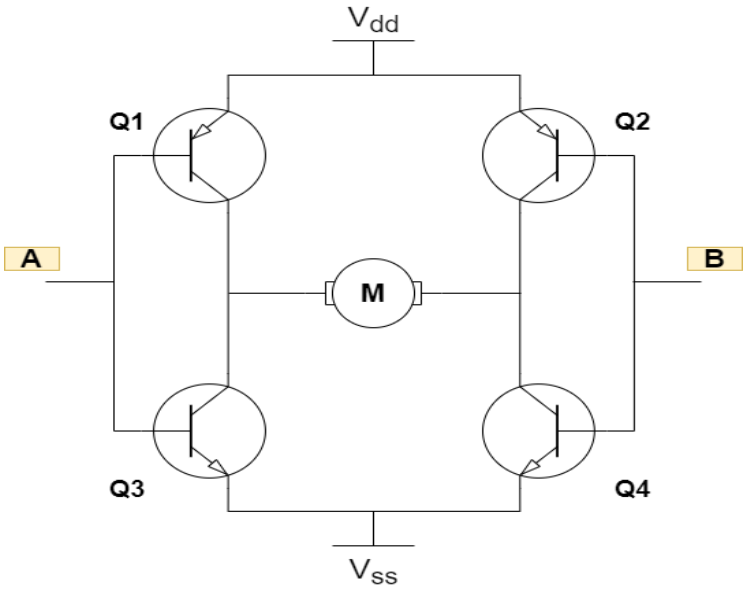



*Figura 2.7: Puntea H*

Circuitul din figura 2.7 este realizat din patru tranzistoare bipolare, două tranzistoare de tip PNP, notate cu Q1 și Q2 și două tranzistoare de tip NPN notate cu Q3 și Q4. Intrările circuitului sunt notate cu A și B, iar alimentarea se realizează prin intermediul surselor constante de tensiune notate cu $V_{dd}$, respectiv $V_{ss}$. Motorul este conectat în centrul circuitului, într-o conexiune în forma literei H, de unde și denumirea circuitului. Pentru a analiza funcționalitatea acestui circuit, trebuie să identificăm cele patru cazuri în care intrările A și B se pot afla. Astfel, cazul în care intrările A și B sunt conectate la un potențial corespunzător nivelului logic „0”, va genera același efect ca și cazul în care intrările A și B sunt conectate la un potențial corespunzător nivelului logic „1” și anume, nu va determina mișcarea de rotație a motorului, deoarece în cele două cazuri tranzistoarele Q1 și Q2 vor fi deschise simultan, respectiv tranzistoarele Q3 și Q4 vor fi deschise simultan, ceea ce va genera același potențial la bornele motorului, deci motorul va fi oprit. Celelalte două situații în care se pot găsi intrările A și B sunt: A conectată la potențialul corespunzător nivelului logic „0” și B conectată la potențialul corespunzător nivelului logic „1”, respectiv A conectată la potențialul corespunzător nivelului logic „1” și B conectată la potențialul corespunzător nivelului logic „0”.

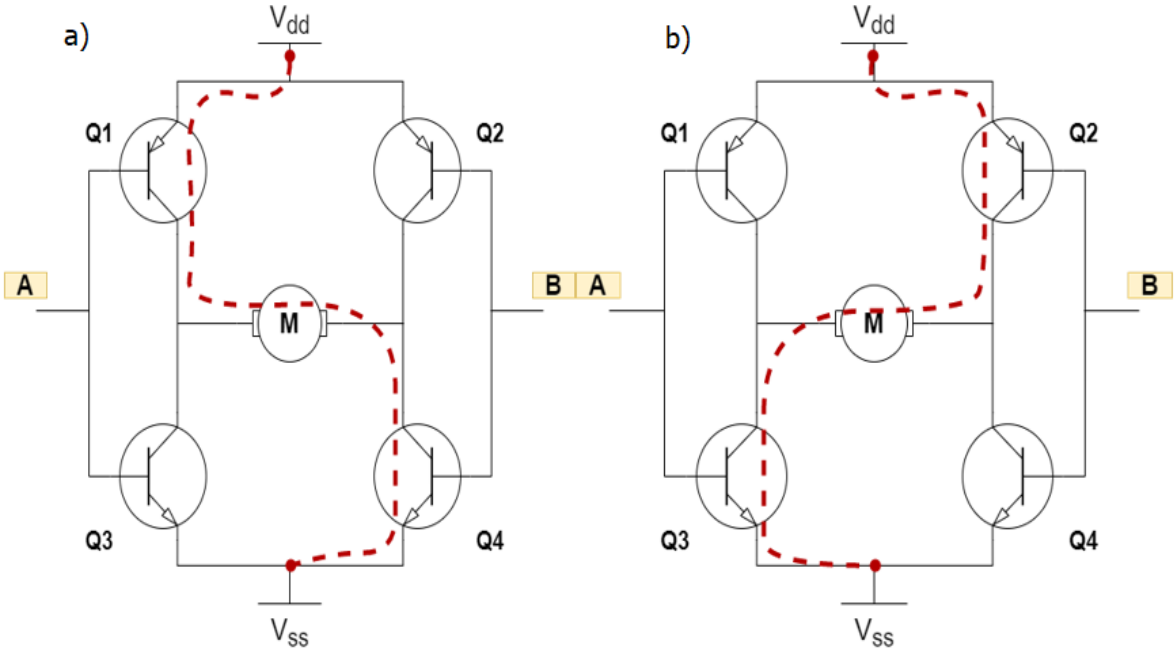



*Figura 2.8: Funcționarea punții H*

În circuitul din figura 2.8 sunt ilustrate cele două cazuri menționate anterior. În figura 2.8.a) este ilustrat cazul în care punctul A e conectat la potențialul corespunzător nivelului logic „0” și B este conectat la potențialul corespunzător nivelului logic „1”. În acest caz, tranzistoarele Q3 și Q2 vor fi blocate, iar tranzistoarele Q1 și Q4 vor fi deschise, oferind o cale de curgere a curentului în direcția ilustrată cu linie punctată. Așadar, în acest caz motorul va fi polarizat generând mișcarea de rotație a axului în direcția respectivă.

În figura 2.8.b) este ilustrat cazul în care punctul A este conectat la potențialul corespunzător nivelului logic „1” și B este conectat la potențialul corespunzător nivelului logic „0”. În această situație, tranzistoarele Q1 și Q4 vor fi blocate, iar tranzistoarele Q2 și Q3 vor fi deschise. Curentul va curge corespunzător liniei punctate, prin tranzistorul Q2, prin motor și prin tranzistorul Q3. Acest lucru va duce la generarea mișcării de rotație în direcția opusă situației din 2.8.a). Așadar, prin utilizarea unei punți H, putem schimba polaritatea motorului prin schimbarea direcției de curgere a curentului prin acesta, ceea ce va duce la generarea mișcării de rotație într-un sens și în celălalt, oferind astfel un control precis al direcției de deplasare a robotului mobil.

Pentru aplicația dezvoltată am utilizat modulul L298N, prin intermediul căruia se pot controla simultan două motoare de curent continuu în ambele direcții, totodată folosind semnale de tip PWM pentru controlul vitezei de rotație a acestora. Modulul driver L298N are patru pini de intrare(IN1 - IN4) , doi pini de activare(EN A și EN B) și patru pini de ieșire la care se pot conecta cele două motoare. Modulul poate fi alimentat în gama de 5-35V, ceea ce îl face potrivit pentru diverse aplicații. Acesta conține și un stabilizator de tensiune 78M05 ce reprezintă o sursă constantă de 5V cu ajutorul căreia poate fi alimentat extern microprocesorul sau alte elemente din cadrul proiectului. Dacă tensiunea de alimentare a modulului L298N are o valoare mai mică de 7V, stabilizatorul intern ce oferă o tensiune constantă de 5V nu va mai funcționa. În acest caz, dacă se dorește folosirea stabilizatorului, se va îndepărta comutatorul și se va alimenta de la o sursă externă. De asemenea, pentru a nu distruge stabilizatorul integrat, atunci când tensiunea de alimentare a modulului este mai mare de 12V, trebuie ca jumperul de la bornele pinului de 5V să fie îndepărtat.

Modalitatea de control a modulului este exemplificată în tabelul 2.1

| INPUT | | | OUTPUT | |
|---|---|---|---|---|
| ENA | IN1 | IN2 | OUT1 | OUT2 |
| H | H | L | H | L |
| H | L | H | L | H |
| L | \ | \ | Z | Z |

*Tabelul 2.1: Controlul motorului utilizând modulul L298N; Sursa [5]*

Când pinul EN A al modulului este un semnal corespunzător nivelului logic „1” („High”), iar intrările modulului sunt IN1 conectat la nivel logic „0” și IN2 conectat la nivel logic „1” , motorul se va roti în sensul acelor de ceasornic. În cazul în care pinul de activare, EN A va fi conectat la nivel logic „1”, iar intrările IN1 și IN2 conectate la „1” logic , respectiv „0” logic, motorul va avea o mișcare de rotație în sens invers acelor de ceasornic. Dacă EN A va fi conectat la „1” logic, iar ambele intrări sunt conectate la „0” logic sau la „1” logic, starea în care se va găsi motorul va fi una de frânare. În final, ultima situație în care poate funcționa motorul este reprezentată de legarea pinului de EN A la nivelul logic „0”, stare în care motorul va fi oprit.

Deoarece robotul mobil are patru motoare, iar driverul de motoare dual L298N este capabil de controlul a două motoare, am decis să conectez motoarele din partea stângă împreună la OUT 3 și OUT4, iar motoarele din partea dreaptă la OUT1 și OUT2. Astfel, spre exemplu, controlând ieșirea corespunzătoare pinilor OUT1 și OUT2, sunt controlate ambele motoare din partea stângă a robotului mobil. Modulul L298N este alimentat prin intermediul unui ansamblu realizat din două baterii de tip Li-Ion, ansamblu ce va fi dezvoltat ulterior în subcapitolul 2.9.

# 2.3 Senzor Fotoelectric Infraroșu

În cadrul proiectului, simplul control al direcției de rotație pentru cele patru motoare utilizate nu este suficient, deoarece pentru a reuși o parcare corectă, trebuie să se cunoască cu precizie date privind deplasarea robotului mobil. O primă soluție pentru monitorizarea corectă a deplasării robotului mobil, o reprezintă utilizarea senzorului fotoelectric infraroșu.

Senzorul fotoelectric infraroșu este un ansamblu realizat dintr-un emițător și un receptor, de obicei aliniate unul către celălalt, cu scopul detectării distanței dintre acestea sau cu scopul detectării prezenței sau absenței unui obiect aflat între cele două elemente. Emițătorul constă într-un LED a cărui lungime de undă emisă se află în domeniul infraroșu, iar receptorul este reprezentat de către un fototranzistor. În cadrul lucrării prezente, am dorit utilizarea a câte unui modul de senzor fotoelectric infraroșu în miniatură în formă de U, pentru fiecare parte a motoarelor din spatele robotului mobil (motorul din stânga și motorul din dreapta). Utilizând acest modul, împreună cu o roată pentru „encoder”, am putut determina numărul de rotații ale motorului.

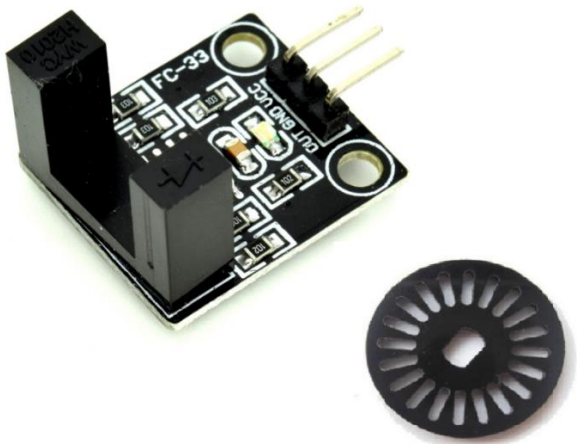



*Figura 2.9: Modul Senzor Fotoelectric Infraroșu[6]*

Modulul prezentat în figura 2.9 conține un senzor fotoelectric în formă de U, ITR9608-F [7] și un comparator LM393, utilizat pentru a obține un output digital. Modulul ITR9608F este realizat dintr-o diodă electroluminiscentă realizată din GaAlAs, a cărei lungime de undă emisă se află în domeniul infraroșu și un fototranzistor NPN aliniate într-o carcasă termoplastică neagră, ce

permite transferul optic între cele două elemente. Structura senzorului fotoelectric în formă de U este reprezentată în figura următoare:

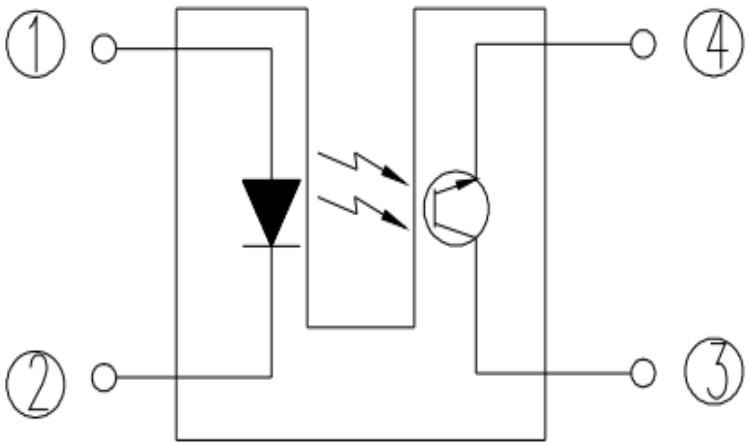



*Figura 2.10: Structura senzorului fotoelectric infraroșu(ITR9608-F) [7]*

Principiul de funcționare al acestei structuri este următorul: atunci când dioda electroluminiscentă este polarizată, căderea de tensiune tipică fiind de 1,2V, aceasta are o emisie spectrală corespunzătoare lungimii de undă centrale de 940 nm, emisie ce este receptată de către fototranzistor. În cazul fototranzistorului, se observă că terminalul din bază lipsește, deoarece nu este necesar să se injecteze un curent de bază. Fotocurentul generat de către emisia diodei LED, în joncțiunea C-B, devine curent de bază pentru fototranzistor. Astfel, dacă tranzistorul este conectat întru-un circuit care îi asigura polarizarea corectă, atunci fotocurentul B-E, amplificat de $\beta$ ori, se va regăsi sub formă de curent de colector în acest circuit. Valoarea tipică pentru curentul de colector de la ieșirea ansamblului este de 20 mA.

Conectarea acestui ansamblu într-un circuit ce conține un comparator LM393, va oferi la ieșire un output digital. Modulul returnează „LOW” atunci când nu se află niciun obiect între cele două brațe sau „HIGH”, în caz contrar. Roata pentru „encoder”, prezentată în figura 2.9 și folosită în cadrul acestui proiect este construită astfel încât să prezinte douăzeci de zone ce permit trecerea radiației emise de către dioda electroluminiscentă și respectiv, alte douăzeci de zone ce se opun trecerii radiației. Astfel, la ieșirea modulului vom avea un semnal digital al cărui nivel „HIGH” va fi generat de prezența zonei ce se opune trecerii radiației emise de către LED, respectiv nivelul „LOW” generat de prezența zonei ce permite trecerea radiației emise de către LED. Așadar, urmărind tranzițiile dintr-un nivel „HIGH” întru-un nivel „LOW”, sau dintr-un nivel „LOW” întru-un nivel „HIGH” și știind că pentru fiecare patruzeci de astfel de tranziții s-a realizat cu succes o rotație completă a roții se va putea calcula cu precizie distanța de deplasare a mașinii.

# 2.4 Senzor de distanță ultrasonic

Până acum, pentru controlul deplasării robotului mobil s-au utilizat cele patru motoare de curent continuu, acționate prin intermediul modulului L298N, iar ca și ansamblu pentru monitorizarea distanței parcurse s-a utilizat senzorul fotoelectric infraroșu. Deoarece mașina este proiectată pentru a reuși să parcheze corect, precizia în ceea ce privește distanțele parcurse de către aceasta este un lucru esențial în cadrul proiectului. Astfel, pentru îmbunătățirea preciziei, s-a mai introdus un alt element de monitorizare a distantelor parcurse, un senzor ultrasonic pentru detecția obstacolelor.

Spectrul sunetele acustice este împărțit în trei intervale. Aceste trei intervale sunt următoarele: infrasonic, audio, ultrasonic. Gama corespunzătoare sunetelor infrasonice corespunde frecvențelor joase, sub 20 Hz. Sursele ce generează aceste sunete infrasonice pot fi, de exemplu, vulcanii, cutremurele, anumite vibrații ce provin de la utilaje de mari dimensiuni. Sunetele din gama audio, se află în spectrul cuprins între 20 Hz și 20 kHz, spectru ce este corespunzător frecvențelor ce pot fi auzite de către urechea umană. Această gama poate varia de la persoana la persoana. Pentru frecvențe mai mari de 20 kHz, domeniul corespunzător este reprezentat de către frecvente corespunzătoare ultrasunetelor.[8] Acest domeniu nu poate fi auzit de către om, dar ultrasunetele pot fi detectate de către anumiți senzori sau de către anumite animale, spre exemplu, liliecii.

În cadrul proiectului, am utilizat senzorul ultrasonic HC-SR04, ce reprezintă un ansamblu realizat dintr-un transmițător ultrasonic, un receptor ultrasonic și un circuit de control. Senzorul ultrasonic HC-SR04 este un senzor de distanță ce poate detecta obiecte aflate la distanțe de până la patru metri, cu un consum redus de curent și care poate fi alimentat la o tensiune de 5 V. Este un senzor ce poate fi achiziționat la un preț redus, ideal pentru aplicații precum sisteme de alarmă, uși cu deschidere automată, detecția obstacolelor, senzor de parcare etc. În cadrul proiectului, senzorul ultrasonic este folosit pentru a măsura distanța între mașină și un anumit obstacol. În funcție de această distanță robotul mobil va ști cu precizie distanta pe care trebuie să se deplaseze.

Traductoarele sunt dispozitive care convertesc un semnal de o anumită natură într-un alt semnal de natura diferită. Transmițătoarele și receptoarele ultrasonice sunt traductoare care convertesc un semnal electric în ultrasunete, respectiv ultrasunetele într-un semnal electric. În cadrul senzorului ultrasonic utilizat, elementele de construcție ale acestuia sunt reprezentate de către cristale piezoelectrice, care produc o oscilație corespunzătoare frecvențelor ultrasonice când se aplică un semnal electric în cazul transmițătoarelor ultrasonice, respectiv, în receptoarelor ultrasonice, cristalul piezoelectric va genera un semnal electric când suprafața acestuia este expusă incidenței undelor ultrasonice. Pentru a putea detecta un obstacol se poate realiza un ansamblu dintr-un transmițător ultrasonic și un receptor ultrasonic. Transmițătorul ultrasonic emite unde, care la întâlnirea unui obstacol se vor reflecta și vor fi depistate de către receptor. Cunoscând viteza undelor sonore prin mediul de transmisie se poate calcula distanța până la obiectul depistat.

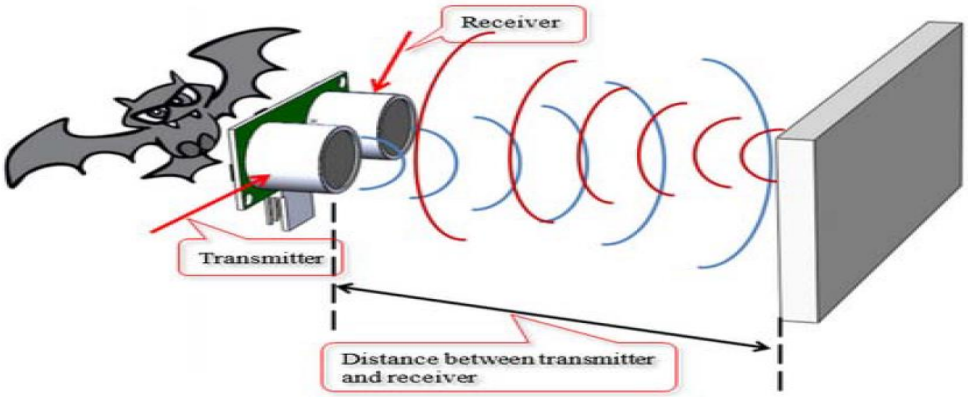



*Figura 2.11: Detecția unui obstacol utilizând senzorul ultrasonic; Sursa [9]*

Principiul de funcționare al senzorului ultrasonic utilizat este următorul: transmiterea unui puls de 10ms pe intrarea „Trig“ a senzorului este necesară pentru începerea măsurătorii. După aceea, modulul va transmite oscilații ultrasonice în cicluri de câte opt impulsuri la o frecvență de 40 kHz. Pinul „Echo“ se află conectat la un potențial corespunzător nivelului logic „1”. Undele ultrasonice se propagă prin mediul de transmisie, iar în momentul când depistează un obstacol, acestea vor fi reflectate de suprafața acestuia. Undele reflectate vor fi recepționate de către receptorul ultrasonic, iar starea pinului „Echo“ va fi una corespunzătoare nivelului logic „0“. Cu alte cuvinte, durata de timp în care pinul „Echo” se află în starea corespunzătoare nivelului logic „1”, corespunde duratei totale de timp necesare undelor ultrasonice să parcurgă distanța de la transmițător la obiect și înapoi de la obiect la receptor. Astfel, semnalul de ieșire de la pinul „Echo“, va fi un semnal corespunzător timpului parcurs de către undele ultrasonice. În figura 2.12 este ilustrat modul de funcționare al modulului ultrasonic utilizat.

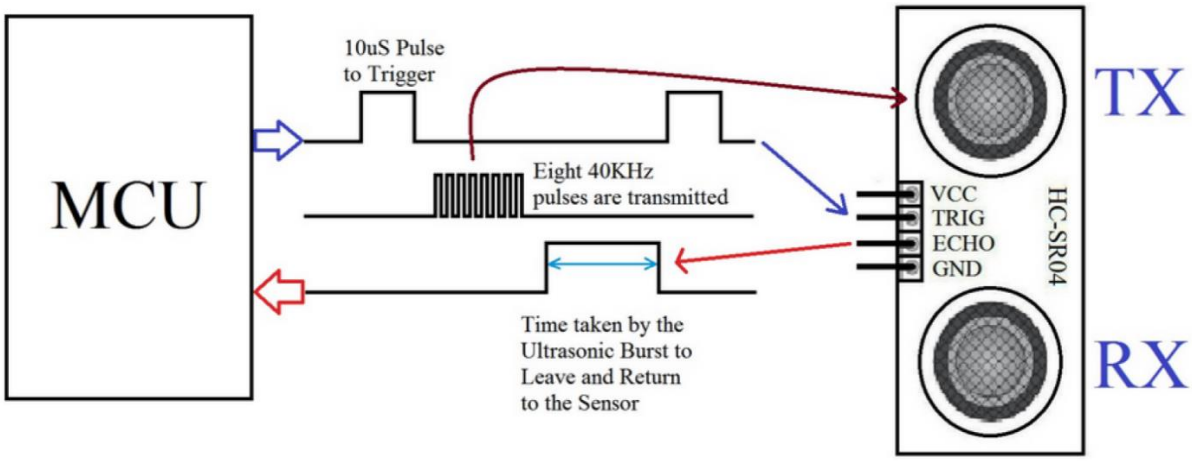



*Figura 2.12: Principiul de funcționare al senzorului ultrasonic; Sursa [9]*

Mediul de transmisie prin care se propagă undele ultrasonice este aerului, astfel, viteza de propagare a undelor ultrasonice, va fi viteza sunetului care este aproximativ 340 m/s. Astfel, pentru a determina distanța până la obiect, se va folosi formula vitezei. Se cunoaște că viteza este egală cu distanța parcursă în unitatea de timp. Astfel, distanța poate fi calculată ca produsul dintre viteza de propagare a sunetului în aer și timpul corespunzător undelor să parcurgă distanța dintre transmițător și obiect , respectiv dintre obiect și receptor. Deoarece distanța parcursă de undele ultrasonice între transmițător și receptor reprezintă dublul distanței între modulul ultrasonic și obstacol, formula finală pentru calculul distanței până la obiect este dată de ecuația 2.1.

$$D = \frac{t * v}{2}$$

*Ecuaţia 2.1: Distanța până la obiect*

Undele ultrasonice emise de către transmițătorul ultrasonic devin tot mai slabe pe măsură ce se îndepărtează de transmițător. De asemenea, puterea semnalului este mai mare în zona centrală din fața transmițătorului, iar pe măsură ce unghiul crește spre exterior, puterea semnalului este atenuată.

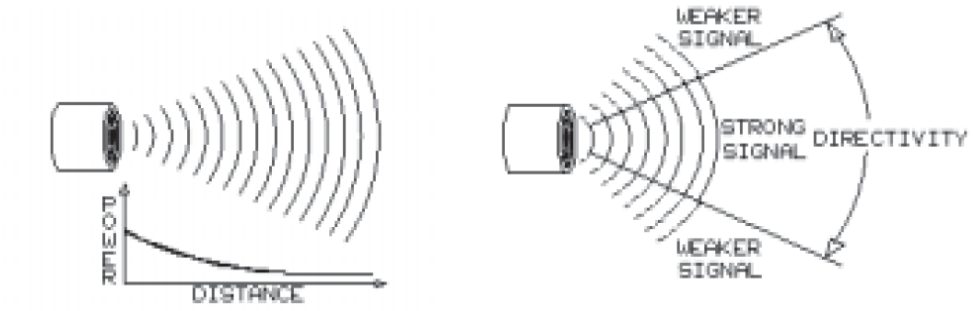



*Figura 2.13: Transmisia undelor ultrasonice; Sursa [8]*

În urma analizei figurii 2.13, se constată că modulul ultrasonic HC-SR04, utilizat în cadrul proiectului, va detecta cu precizie un obstacol care se află în centrul direcției sale de emisie. În cazul obstacolelor poziționate la un unghi mai mare de 15º, undele ultrasonice pot fi reflectate în afara zonei de acțiune a modulului, astfel vor apărea probleme de detecție a obstacolelor. Totodată, utilizând un astfel de modul pot apărea probleme în încercarea de a depista obstacole a căror formă este curbată, mică sau subțire. O altă serie de dezavantaje pe care le prezintă acest modul ar putea fi reprezentate de faptul că modulul este sensibil la variația temperaturii, deoarece viteza de propagare a sunetului depinde de temperatură, sau de faptul că modulul nu poate depista culoarea obstacolului. Cu toate astea, în comparație cu modulul infraroșu de obstacole utilizat în cadrul parcării inteligente și descris în paragraful 3.3, senzorul ultrasonic HC-SR04 poate depista obiecte la distanțe mult mai mari, gama de acțiune fiind între 2cm și 400cm, prezintă o eroare la măsurarea distanței de doar 3mm, este insensibil la interferențe precum fumul sau diferite condiții de iluminare și totodată, poate poate fi achiziționat la un preț foarte scăzut.

# 2.5 Magnetometrul

Până acum, în cadrul proiectului, pentru o deplasare corectă a robotului mobil am utilizat patru motoare de curent continuu, controlate prin intermediul modulului L298N, ce conține două circuite în configurație de punte H, iar ca și sistem pentru monitorizarea corectă a deplasării am utilizat senzorul ultrasonic de distanță și modulul fotoelectric infraroșu prezentate în secțiunile anterioare. Integrarea acestor elemente a dus la deplasarea corectă a mașinii pe direcția înainte, însă, în ceea ce privește virajul acesteia, ansamblul senzorilor utilizați nu oferă performanțe satisfăcătoare. Astfel, pentru a realiza un viraj corect, în cadrul proiectului a mai fost adăugat modulul HMC5883L, cu ajutorul căruia se monitorizează poziția mașinii. Modulul HMC5883L este un dispozitiv cu ajutorul căruia se poate măsura câmpul magnetic din vecinătatea acestuia, putându-se astfel determina orientarea acestuia față de polii magnetici ai pământului. Știind astfel poziția mașinii înainte de viraj și poziția acesteia după realizarea manevrei, se poate afla unghiul în care mașina s-a mișcat și astfel se pot face corecțiile necesare. Magnetometrul are la bază modulul HMC5883L, realizat de firma Honeywell. Având dimensiuni foarte reduse și un preț scăzut, modulul poate fi integrat cu ușurință în diferite aplicații precum telefoanele mobile, sistem de navigație auto, busolă digitală, etc.

Modulul utilizează tehnologia AMR, tehnologie ce are la bază magnetorezistoare realizate din metale feromagnetice anizotrope. Efectul magnetorezistiv este prezent în aproape toate metalele, însă, pe baza structurilor bazate pe Ni-Fe sau pe alte materiale feromagnetice, s-au găsit cele mai multe aplicații. [10] Un material anizotrop reprezintă un material cu o distribuție regulată a atomilor sau a ionilor din interiorul acestuia. Cu alte cuvinte, proprietățile unui material anizotrop depind de direcția exercitată din exterior. Magnetorezistoarele sunt rezistoare a căror rezistență se modifică la interacțiunea cu un câmp magnetic exterior.

Magnetorezistivitatea este definită ca variația relativă a rezistivității materialelor în prezența unui câmp magnetic H. Variația concretă a rezistivității depinde de material, de unghiul dintre câmpul electric și câmpul magnetic aplicat. Astfel, materialele feromagnetice utilizate au o puternică anizotropie structurală, care influențează dependența rezistivității de câmpul magnetic.[10]

În figura 2.14 este prezentat efectul magnetorezistiv al materialelor anizotrope. Se observă cum, la aplicarea unui câmp magnetic extern $\vec{H}$, în planul rezistorului, vectorul de magnetizație $\vec{M}$ se orientează pe direcția câmpului aplicat. Între vectorul de magnetizație $\vec{M}$ și direcția de curgere a curentului $\vec{J}$ se formează unghiul $\alpha$. Variația rezistenței materialului magnetorezistiv este o funcție de $\cos^2 \alpha$, unde unghiul $\alpha$ este dependent de mărirea câmpului magnetic aplicat. Așadar, rotirea vectorului magnetizație $\vec{M}$, spre vectorul densitate de curent $\vec{J}$ , duce la creșterea rezistenței materialului magnetorezistiv, iar rotirea vectorului magnetizație $\vec{M}$, contrar direcției vectorului densitate de curent $\vec{J}$ , duce la scăderea rezistenței.[10]

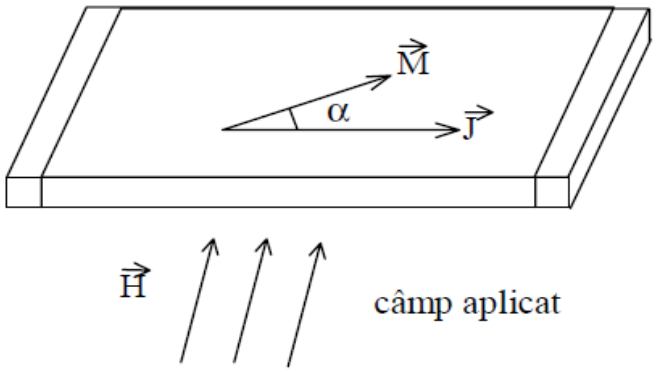



*Figura 2.14: Efectul magnetorezistiv de anizotropie (AMR); Sursa[10]*

De obicei, magnetorezistoarele se utilizează într-o configurație de punte. Pentru fiecare latură a punții se utilizează materiale magnetorezistive cu aceeași rezistență R, însă, pentru brațele opuse ale punții magnetizația este aleasă astfel încât să se producă micșorarea, respectiv mărirea rezistenței. În figura 2.15 este prezentată structura de tip punte a magnetorezistoarelor. Astfel, tensiunea dintre punctele B și D, tensiunea de ieșire a punții va fi proporțională cu variația relativă a rezistenței $\frac{\Delta R}{R}$. [10] Această variație relativă a rezistenței materialului magnetorezistiv este egală cu produsul dintre câmpul magnetic aplicat $\vec{H}$ și sensibilitatea senzorului magnetorezistiv.

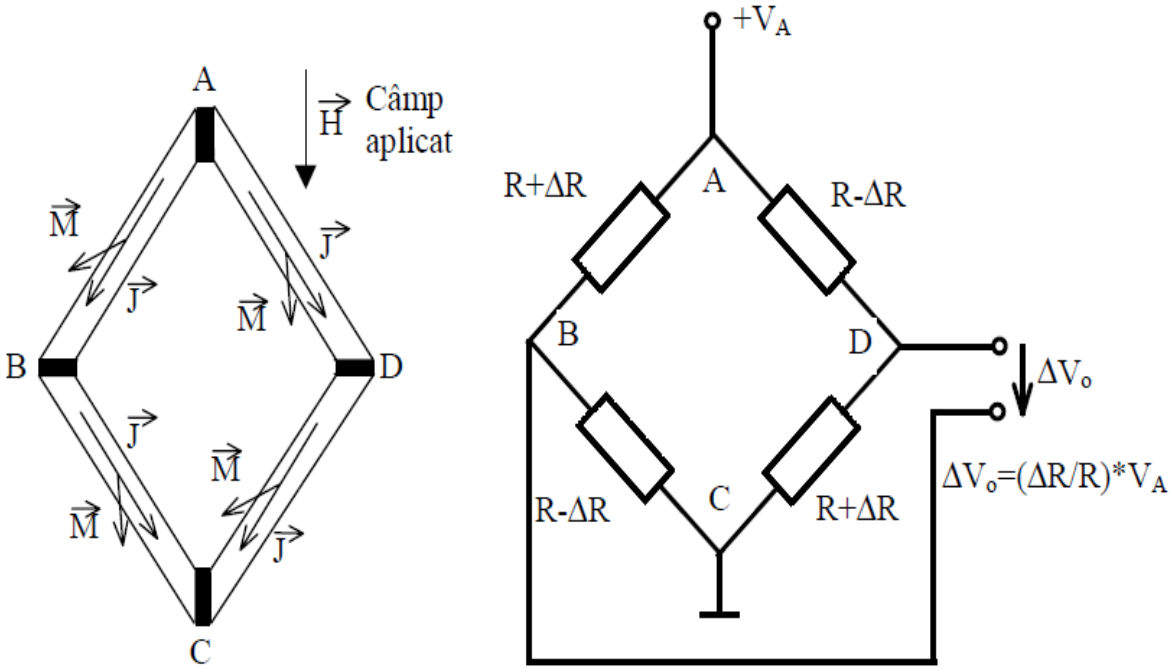



*Figura 2.15: Utilizarea în configurație de tip punte a magnetorezistoarelor; Sursa[10]*

Magnetometrul comunică cu microcontrolerul prin intermediul protocolului de comunicație $I^2C$. Protocolul de comunicație $I^2C$ este utilizat pentru comunicația dintre unul sau mai multe dispozitive de tip „master” cu unul sau mai multe dispozitive de tip „slave”. Este un protocol de comunicație serială ce utilizează doar două fire pentru transmiterea datelor între dispozitive. Aceste două magistrale de comunicație sunt SDA („Serial Data”) și SCL („Serial Clock”). Magistrala SDA este utilizată pentru transmiterea și pentru recepția datelor atât de către dispozitivele de tip „master”, cât și de către dispozitivele de tip „slave”. Magistrala SCL este utilizată pentru semnalul de ceas, care este controlat de către dispozitivele de tip „master”. Schimbul de date dintre dispozitivele de tip „master” și dispozitivele de tip „slave” pe magistrala seriala de date, SDA, se realizează sincron în funcție de semnalul de ceas.

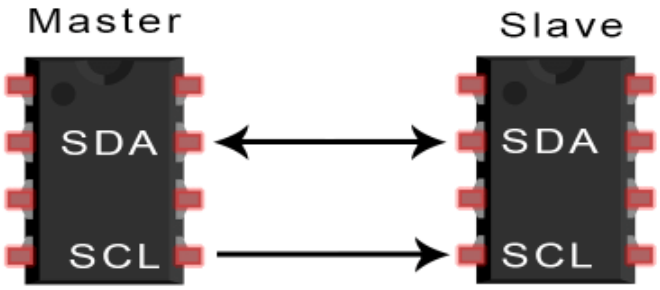



*Figura 2.16: Protocolul de comunicație $I^2C$; Sursa [11]*

Utilizând protocolul de comunicație $I^2C$, datele sunt transmise sub formă de mesaje, ce sunt împărțite în cadre de date. Fiecare mesaj conține: adresa dispozitivului de tip „slave” cu care dispozitivul de tip „master” vrea să comunice, unul sau mai multe cadre de date ce conțin informația ce se dorește a fi transmisă, biți de start și stop, biți de scriere sau citire și biți de confirmare sau neconfirmare.

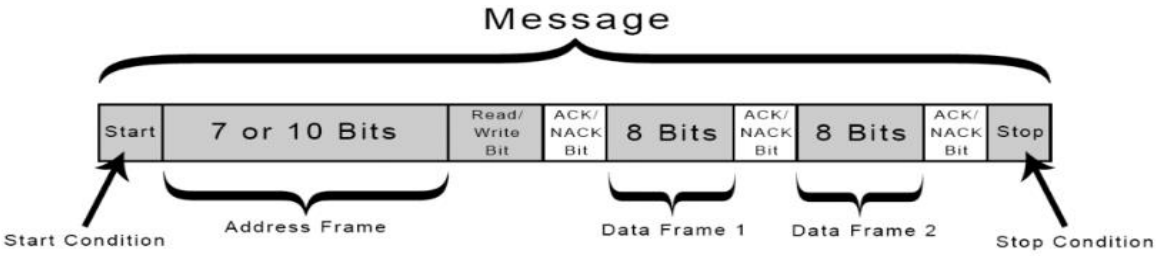



*Figura 2.17: Structura mesajului transmis utilizând protocolul de comunicație $I^2C$; Sursa [11]*

Pentru a începe comunicația, dispozitivul de tip „master” va transmite bitul de start către fiecare dispozitiv de tip „slave” conectat, schimbând semnalul transmis pe magistrala SDA, dintr-un semnal de nivel logic „1”, într-un semnal de nivel logic „0” , înainte de a schimba semnalul de ceas din nivelul logic „1” în nivelul logic „0”. După trimiterea bitului de start, dispozitivul de tip „master” va transmite fiecărui dispozitiv de tip „slave”, adresa dispozitivului cu care dorește să

comunice, împreună cu bitul de scriere sau citire. Bitul de scriere sau citire are rolul de a informa dispozitivul de tip „slave” dacă dispozitivul de tip „master” dorește să trimită sau să primească date de la acesta. Dacă dispozitivul de tip „master” dorește să transmită date, bitul de scriere va fi corespunzător nivelului logic „0”, iar dacă dispozitivul de tip „master” dorește să citească date de la dispozitivul de tip „slave”, bitul va fi corespunzător nivelului logic „1”. Fiecare dispozitiv de tip „slave” va compara adresa trimisă cu propria sa adresă, iar dacă adresa trimisă se potrivește cu adresa proprie, dispozitivul de tip „slave” va returna un bit de confirmare pe magistrala SDA. După identificarea dispozitivului de tip „slave” cu care se dorește a se inițializa comunicația, dispozitivul de tip „master” va transmite sau va recepționa cadrele de date. După fiecare cadru de date transmis, dispozitivul receptor va returna un bit de confirmare pentru a marca transmiterea cu succes a informației. Pentru a opri transmiterea datelor, dispozitivul de tip „master” va transmite un bit de stop, schimbând semnalul de pe magistrala SCL în nivelul logic „1”, înainte de a schimba semnalul de pe magistrala SDA într-un semnal corespunzător nivelului logic „1”.[11]

Utilizând o adresă de 7 biți, un dispozitiv de tip „master” poate controla până la 128 dispozitive de tip „slave” ($2^7$ = 128 adrese unice), iar utilizând o adresă pe 10 biți, un dispozitiv de tip „master” poate comunica cu până la 1024 de dispozitive de tip „slave” ($2^{10}$ = 1024 adrese unice). Totodată, utilizând protocolul de comunicație $I^2C$, mai multe dispozitive de tip „master” pot comunica cu unul sau mai multe dispozitive de tip „slave”, ceea ce reprezintă un avantaj atunci când se dorește accesarea unei memorii de unul sau mai multe microcontrolere sau atunci când se dorește afișarea unor date pe un display LCD.

Așadar, prin integrarea magnetometrului în cadrul proiectului, am reușit să cresc performanțele robotului mobil în ceea ce privește virajul acestuia.

# 2.6 Senzor de temperatură și umiditate

În cadrul proiectului, robotul mobil a fost echipat cu senzorul de temperatură și umiditate DHT11, pentru a colecta date de la mediul exterior, urmând ca aceste date să fie transmise către utilizator și afișate pe ecranul LCD de la telecomandă. Senzorul DHT11 este un senzor de temperatură și umiditate, ce poate fi alimentat în gama de tensiuni cuprinse între 3,3 V – 5 V, ce consumă un curent de maxim 2,5 mA. Cu ajutorul acestui senzor se pot măsura temperaturi cuprinse între 0º C și 50º C, cu o eroare de măsurare de ± 2º C. Modulul nu poate fi folosit pentru temperaturi sub 0º C. Gama de măsurare a umidității este cuprinsă între 20% - 95% RH („relative humidity”), eroarea de măsurare a temperaturii fiind de ± 5% RH. [12] Conform datelor menționate în fișa de catalog, modulul are dimensiuni reduse, 16 mm x 5,5 mm x12 mm, este prevăzut cu 4 pini: pinul de alimentare, pinul de masă, pinul de date și un pin ce nu este utilizat, este încapsulat într-o carcasă de plastic prevăzută cu orificii, proces ce este realizat în laborator. Coeficienții de calibrare sunt de asemenea, elemente ce au fost calibrate în cadrul procesului de fabricație. Astfel, datorită dimensiunilor sale reduse, a costurilor scăzute, a compatibilității cu o gamă largă de microcontrolere, a fiabilității oferite, senzorul DHT11 este ideal pentru aplicații precum sisteme de monitorizare utilizate în industria agricolă sau sisteme de monitorizare a propriei grădini, realizarea unei stații meteorologice mobilă, sisteme de control pentru temperatura și umiditatea din mediul casnic, echiparea pe roboți mobili ce explorează diferite medii, etc.

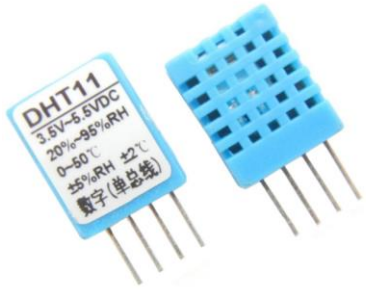



*Figura 2.18: Senzorul de temperatură și umiditate DHT11; Sursa: Oprimus Digital*

Senzorul de temperatură și umiditate DHT11, are la bază un ansamblu realizat dintr-o componentă ce măsoară umiditatea, o componentă cu ajutorul căreia este măsurată temperatura și un circuit integrat specializat ce prelucrează datele primite de la cele două elemente menționate anterior, făcând ulterior posibilă comunicația serială pe un singur fir cu microcontrolerul ce va primi informațiile de la senzor. Determinarea umidității din aer se realizează utilizând o componentă de măsurare a umidității de tip rezistiv. Această componentă detectează vaporii de apă din aer prin măsurarea rezistenței electrice dintre doi electrozi. Ansamblul traductor de tip rezistiv pentru determinarea umidității este realizat din următoarele elemente: un substrat, de regulă realizat din materiale ceramice, pe care este atașat un material ce are proprietatea de a absorbi vaporii de apă din aer, în urma căreia sunt generați ioni ce duc la creșterea conductivității dintre cei doi electrozi atașați ansamblului.

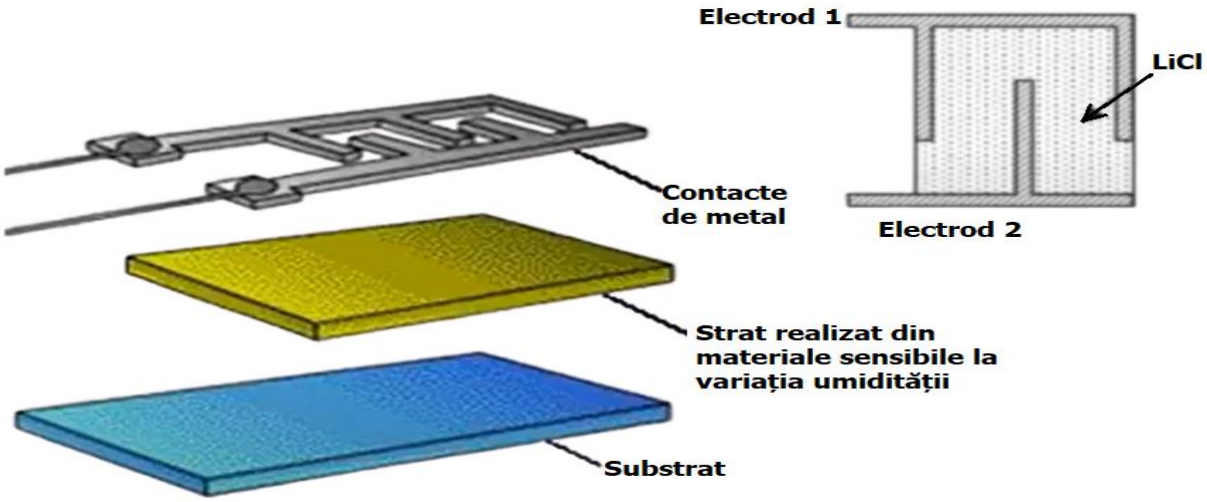



*Figura 2.19: Traductor de tip rezistiv pentru determinarea umidității; Sursa: [13]*

În figura 2.19 este ilustrată structura unui astfel de element pentru determinarea umidității. Așadar, principiul de funcționare este următorul: atunci când vaporii de apă sunt absorbiți de stratul sensibil la umiditate, ionii sunt eliberați ceea ce duce la creșterea conductivității dintre cei doi

electrozi. Creșterea conductivității dintre cei doi electrozi duce la scăderea rezistenței dintre acestea. Modificarea rezistenței dintre cei doi electrozi este proporțională cu umiditatea relativă, astfel, când umiditatea relativă a aerului crește, duce la creșterea conductivității dintre cei doi electrozi, deci la scăderea rezistenței dintre aceștia, iar când umiditatea relativă este scăzută, crește rezistența dintre electrozi. Umiditatea relativă a aerului reprezintă raportul dintre moleculele de apă aflate în aer și numărul maxim de molecule de apă care pot fi în aer la o temperatură dată. Umiditatea relativă este exprimată în procente. De exemplu, 60% umiditate relativă înseamnă că în aer se găsesc 60% molecule de apă dintr-un procent total de 100%. Umiditatea relativă nu poate depăși valoarea de 100%, deoarece apare fenomenul de condensare. Astfel, după cum se poate observa în foaia de catalog a senzorului, gama umidității relative ce poate fi măsurată diferă în funcție de temperatură. Astfel, la 0º C, gama umidității este cuprinsă între 30% RH și 90% RH, la 25º C se află între 20% RH și 90% RH, iar la o temperatură de 50 ºC, gama de măsurare a umidității relative este cuprinsă între 20% RH și 80% RH.

Pentru măsurarea temperaturii, modulul are la bază un termistor NTC, componentă a cărei rezistență se modifică odată cu variația temperaturii. Fiind un termistor NTC, coeficientul de variație cu temperatura este negativ, ceea ce înseamnă că rezistența termistorului scade atunci când valoarea temperaturii crește. Legea după care rezistența unui termistor cu coeficient negativ de variație a temperaturii își modifică valoarea este dată de ecuația 2.2.

$$R_T = A * e^{\frac{\beta}{T}}$$

*Ecuația 2.2: Legea de variație a rezistenței termistoarelor NTC; Sursa: [14]*

În cadrul ecuației 3.1 mărimile utilizate sunt următoarele:

- $R_T$ reprezintă rezistența termistorului la temperatura T , temperatură ce este exprimată în Kelvin
- β reprezintă o constantă de material ce caracterizează sensibilitatea termistorului NTC. Valorile uzuale ale constantei β sunt cuprinse între 2000ºK și 5000ºK.
- Factorul multiplicativ A, reprezintă o constantă ce depinde de tipul termistorului, unitatea de măsura a acesteia este **Ω** și reprezintă valoarea rezistenței termistorului când temperatura tinde către valori infinite (ipotetic) [14]

Caracteristica variației rezistenței unui termistor cu coeficient de variație negativ al temperaturii este prezentată în figura 2.20. Se observă faptul că variația rezistenței termistorului în funcție de temperatură este o funcție exponențială negativă.

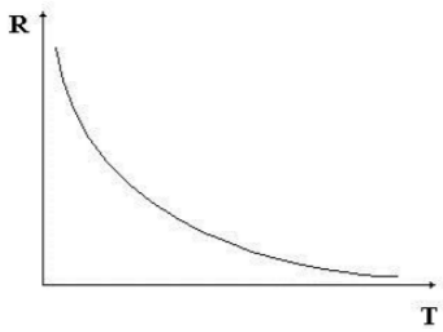



*Figura 2.20 :Caracteristica termică a termistorului de tip NTC; Sursa: [14]*

Un circuit integrat specializat cu o ieșire pe 8 biți este utilizat în cadrul modulului DHT11 cu rolul de a prelucra informațiile colectate de la cele două elemente cu ajutorul cărora se realizează măsurătorile. De asemenea, circuitul specializat stochează coeficienții de calibrare stabiliți în procesul de fabricație și controlează modalitatea de transmisie între senzorul DHT11 și microcontrolerul cu care acesta comunică; în cazul proiectului prezent, senzorul DHT11 este conectat la placa de dezvoltare ce folosește un microcontroler ATmega328P. Transmisia datelor între senzorul DHT11 și microcontroler se realizează prin intermediul interfeței seriale, prin pinul de date ce este conectat la microcontroler. În cadrul proiectului am conectat un rezistor de „pull-up" la pinul date, acesta având rolul de a conecta pinul de date la nivelul logic "1", atunci când pe pinul de date nu se transmite un semnal, evitând astfel apariția de valori aleatorii datorate zgomotului.

Pentru a inițializa comunicația între senzorul DHT11 și microcontroler, acesta din urmă va trimite un semnal de pornire, urmând ca senzorul DHT11 să transmită pe firul de date un cadrul de 40 de biți ce reprezintă valorile umidității și temperaturii citite de acesta. Fără ca senzorul să primească un semnal de activare, acesta nu va transmite informații către microcontroler și se va afla în starea de așteptare. Pașii realizați în cadrul procesului de comunicație sunt prezentați în figura 2.21.

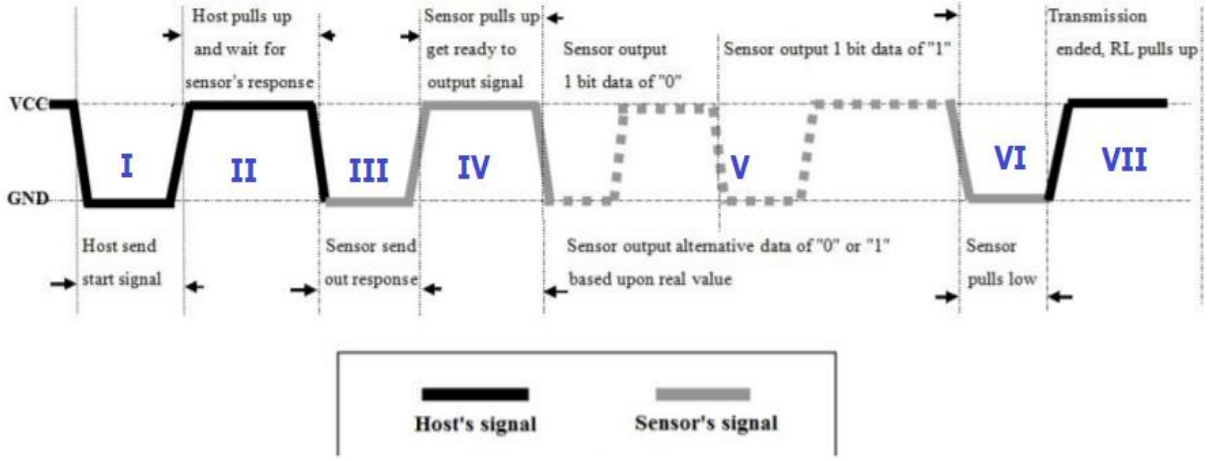



*Figura 2.21: Procesul de comunicație dintre DHT11 și microcontroler; Sursa: [15]*

Așa cum se poate observa în figura 2.21, microcontrolerul, cu denumirea generică utilizată în figură de „host” (gazdă), trimite în etapa I, un semnal de activare către senzor, trăgând semnalul din nivel logic „0” în nivel logic „1”, semnal ce este numit semnal de start și are rolul de a activa comunicația între cele două elemente. În etapa a II-a, microcontrolerul setează semnalul în nivelul logic „1”, așteptând răspunsul de la senzorul DHT11. În etapa a III-a și a IV-a, senzorul confirmă primirea cererii de la microcontroler și setează pinul de date la nivelul „1” logic pentru a informa microcontrolerul că urmează să-i transmită datele privind temperatura și umiditatea. În etapa a V-a sunt transmise informațiile într-un cadru de 40 de biți, primii 16 biți reprezentând informația privind umiditatea relativă măsurată, următorii 16 biți reprezentând informația despre valoarea temperaturii măsurate de către senzor, iar ultimii 8 biți reprezentând biții de control pentru a se verifica ca toate informațiile au fost transmise fără pierderi. În etapele VI și VII este marcată finalizarea comunicației între cele două dispozitive, senzorul trecând în modul de așteptare.

Așadar, senzorul DHT11 este un modul ușor de folosit, prin intermediul căruia se pot achiziționa date privind mediul exterior, date ce prezintă informații utile pentru utilizator. Am dorit să integrez un astfel de senzor în cadrul proiectului, deoarece aceste date pot fi importante atunci când utilizatorul dorește să cunoască temperatura sau umiditatea din parcare sau când acesta dorește să primească informații despre mediul înconjurător atunci când folosește robotul mobil pentru a explora o zonă greu accesibilă.

## 2.7 Modul de comunicație wireless

Partea de comunicație între cele trei module utilizate în cadrul proiectului se realizează prin utilizarea modulului nRF24L01. Acesta este un modul wireless ce are la bază circuitul integrat nRF24L01 proiectat de către firma Nordic Semiconductor. Modulul este un dispozitiv a căror tensiuni de alimentare funcționează în gama 1,9 - 3,6V, ce utilizează interfața de comunicație SPI, operează la frecvența de 2,4 GHz, capabil să transmită cu viteze de până la 2Mbps și totodată, ce poate fi achiziționat la un cost redus. Acest modul este ideal pentru integrarea în aplicații precum telecomenzi fără fir, tastatură wireless, rețele de comunicație fără fir, mouse wireless, controlere pentru jocuri video, sisteme automate pentru case inteligente, diverse jucării, etc. Utilizând acest modul se pot transmite date la distanțe de până la 80 metri în câmp deschis, ceea ce îl face ideal pentru aplicațiile menționate anterior.

În cadrul proiectului am utilizat trei astfel de dispozitive, fiecare având rolul de a transmite și primi informații specifice. În cazul parcării, modulul nRF24L01 are rolul de a transmite informațiile primite de la senzorii instalați în parcare, privind disponibilitatea locurilor de parcare, către robotul mobil, iar în cazul robotului mobil și în cazul telecomenzii, modulele au atât rolul de transmițător, cât și de receptor. De exemplu, modulul utilizat în cadrul robotului mobil joacă rolul de transmițător, atunci când sunt transmise către telecomandă informații privind temperatura și umiditatea mediului ambiant și rolul de receptor atunci când acesta primește comenzi de deplasare de la telecomandă.

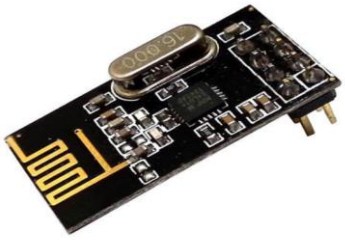


*Figura 2.22: Modulul nRF24L01; Sursa: Optimus Digital*

Modulul este proiectat pentru operarea în banda de frecvențe ISM cuprinsă între 2,400 – 2,4835 GHz, bandă ce este oferită gratuit la nivel mondial și este utilizată în principal pentru aplicații industriale, științifice și medicale. Modulul se conectează prin intermediul a opt pini: pinii GND și VCC sunt utilizați pentru alimentarea modulului, un pin folosit pentru întreruperi, pinul CE utilizat pentru activarea modulului și alți patru pini utilizați în cadrul comunicației SPI.

Modulul nRF24L01 poate fi configurat prin intermediul interfeței seriale SPI („Serial Peripheral Interface”), astfel se pot configura diferiți parametrii precum: frecvența canalului, puterea de ieșire sau rata de transmisie a datelor. Interfața de comunicație SPI reprezintă un protocol de comunicație sincron, ceea ce înseamnă că utilizează două linii separate pentru transmisia și recepția datelor și un semnal de ceas pentru sincronizarea acestora. O altă caracteristică a interfeței de comunicație serială SPI o reprezintă faptul că datele pot fi transferate fără întrerupere, astfel acest tip de comunicație nu necesită transmiterea biților de start și de stop. Dispozitivele ce comunică prin protocolul SPI, se află într-o relație de master-slave. Dispozitivul de tip „master” este reprezentat, de obicei, de către microcontroler, iar dispozitivul de tip „slave” poate fi reprezentat de către senzori, afișaje LCD, memorii, etc. În cadrul comunicației seriale SPI, semnalul de ceas este generat de către dispozitivul de tip „master”. Există un singur dispozitiv de „master” ce poate comunica cu unul sau mai multe dispozitive de tip „slave”. Cea mai simplă configurație fiind reprezentată de către o rețea cu un singur dispozitiv de tip „master” și un dispozitiv de tip „slave”. Această configurație este ilustrată în figura următoare:

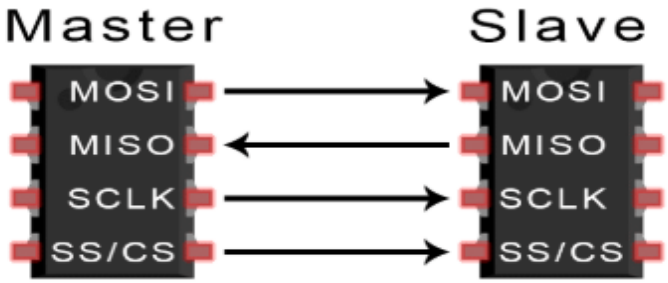



*Figura 2.23: Configurația Master-Slave [16]*

Când data este transmisă de la dispozitivul de tip „master” către dispozitivul de tip „slave”, este folosită magistrala de date numită MOSI („Master Output / Slave Input”), iar magistrala folosită de dispozitivul de tip „slave” pentru a transmite date către dispozitivul de tip „master” se cheamă MISO („Master Input / Slave Output”). Magistrala SCLK este folosită pentru semnalul de ceas, iar magistrala SS/CS este folosită pentru selecția dispozitivului de tip „slave”. Modul de funcționare este următorul: dispozitivul de tip „master” generează semnalul de ceas pe baza căruia se vor sincroniza transmisia și recepția datelor, apoi se transmite un semnal corespunzător nivelului logic „0” pe magistrala de date SS/CS pentru a selecta dispozitivul de tip „slave”, cu care se va comunica. Dispozitivul de tip „master”, trimite datele prin intermediul magistralei MOSI, câte un bit la fiecare front pozitiv de ceas, iar acestea sunt recepționate de către dispozitivul de tip „slave”. Dacă un răspuns este necesar, dispozitivul de tip „slave” v-a transmite datele către „master” pe magistrala MISO, transmisie ce va fi, de asemenea, sincronă cu semnalul de ceas.

Modulul nRF24L01, poate opera la frecvențe cuprinse între 2.4 GHz și 2.525 GHz. Conform foii de catalog[17], modulul poate folosi până la 125 de canale diferite, ce ocupă o lățime de bandă mai mică de 1 MHz, atunci când rata de transmisie a datelor este setată la 250 kbps sau 1 Mbps. Putând folosi până la 125 de canale diferite, modulul oferă posibilitatea de a crea 125 rețele independente. Fiecare modul poate comunica cu până la alte șase module.

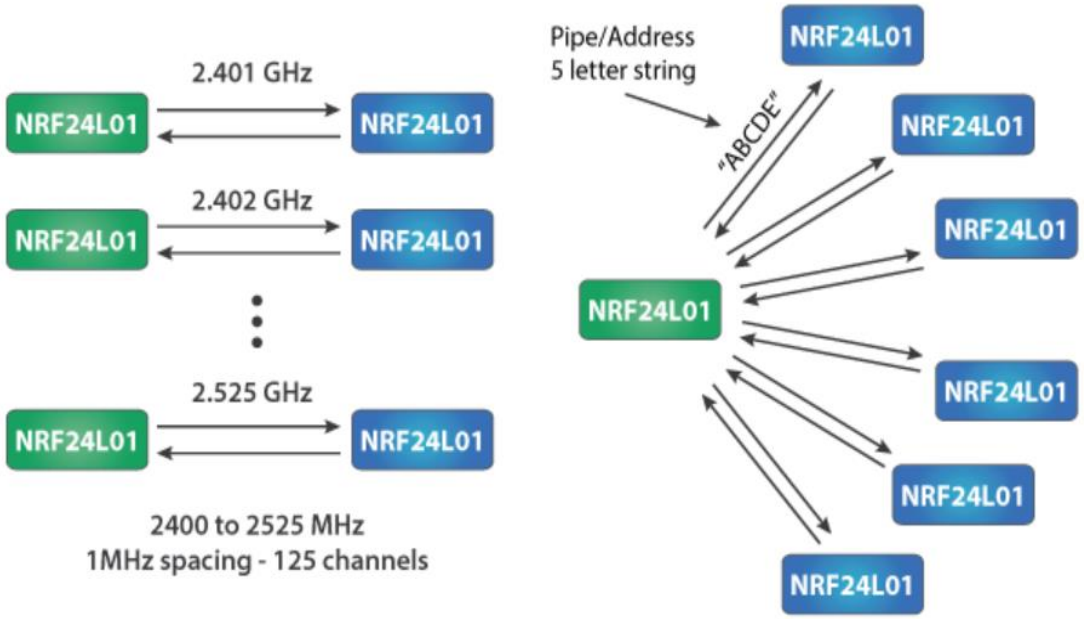



*Figura 2.24: Distribuția canalelor utilizând modulul nRF24L01; Sursa: [18]*

# 2.8 Placa de dezvoltare compatibilă cu Arduino Nano

Arduino Nano este o placă de dezvoltare proiectată de către Arduino.cc, ce are la bază microcontrolerul Atmega328p. Având dimensiunile de 45 mm x 18 mm și o greutate de doar 7g, placa de dezvoltare Arduino Nano este ideală pentru proiecte de dimensiuni mici, fiind ușor integrabilă în orice tip de proiect. Placa este prevăzută cu doisprezece pini digitali, numerotați de la D2 la D13 și opt pini analogici numerotați de la A0 la A7. Toți acești pini pot fi folosiți ca pini de intrare sau ieșire, dar pe lângă asta, aceștia pot prezenta diferite funcții. Spre exemplu, pinii D3, D5, D6, D9, D10, D11 pot fi folosiți ca și pini de ieșire ce pot genera un semnal PWM, pinii analogici numerotați de la A0 la A7, pot fi folosiți pentru măsurarea unui semnal analogic cuprins între 0V și 5V, pinii digitali D2 și D3 pot fi folosiți pentru a genera întreruperi, pinii D10, D11, D12 și D13 pot fi utilizați în cadrul protocolului de comunicație SPI, pinul D13 poate fi folosit pentru acționarea ledului atașat pe placă, pinii analogici A4 și A5 pot fi folosiți în cadrul comunicației $I^2C$ sau pinii D0 și D1 pot fi folosiți pentru comunicația serială. Placa este prevăzută cu doi pini de masă, doi pini de resetare, un pin de 5V pentru alimentarea microcontrolerului și a altor componente de pe placă, un pin de 3,3V a cărui tensiune este generată de stabilizatorul intern, un pin $V_{in}$ pentru alimentarea plăcii Arduino Nano de la o sursă externă și patru LED-uri montate pe suprafața plăcii ce sunt utilizate pentru marcajul alimentării, pentru transmiterea și recepționarea datelor prin comunicația serială și pentru uz general, LED ce este conectat la pinul D13.

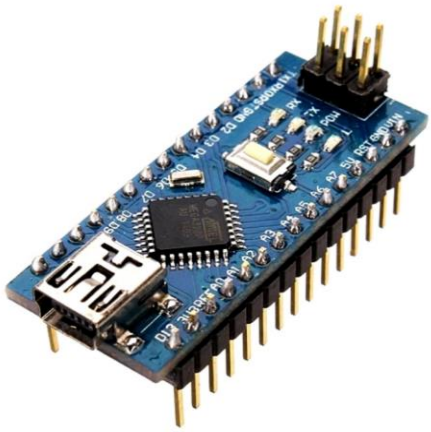


*Figura 2.25: Placa de dezvoltare Arduino Nano; Sursa: Optimus Digital*

Alimentarea plăcii Arduino Nano se poate realiza în trei moduri. O primă variantă ar fi prin conectarea mufei USB Jack Mini tip B printr-un cablu la computerul personal, ceea ce va oferi alimentarea necesară plăcuței pentru a funcționa. O a doua variantă pentru alimentarea plăcuței ar fi conectarea unei surse externe de tensiune la pinul $V_{in}$ . Gama de tensiuni ce se poate aplica la intrarea $V_{in}$ este cuprinsă între 7 V - 12 V, iar stabilizatorul intern va oferi o tensiune constantă de 5 V necesară pentru alimentarea microcontrolerului și a componentelor. O ultimă variantă prin care se poate alimenta placa de dezvoltare Arduino Nano este prin conectarea la pinul de 5V a unei

surse externe constante de tensiune de 5V. Pentru programarea microcontrolerului Atmega328p prin intermediul cablului cu mufă USB Jack Mini tip B, se folosește circuitul integrat CH340G, ce are rolul de a converti semnalele primite prin cablul USB la comunicație serială UART, cea cu care microcontrolerul principal Atmega328p este compatibilă.

UART („Universal Asynchronous Receiver/Transmitter") este un circuit integrat, care are rolul de a controla comunicația serială, de la microcontroler către un alt microcontroler sau un dispozitiv extern. Un prim avantaj pe care îl prezintă comunicația serială UART, se datorează faptului că necesită doar două fire de conexiune între dispozitivele ce schimbă informații, spre deosebire de comunicația prin intermediul protocolului de comunicație SPI, unde sunt necesare patru conexiuni. Prin utilizarea comunicației seriale UART, nu este necesară transmiterea semnalului de ceas, comunicația fiind astfel una asincronă. Pentru a comunica între două entități, spre exemplu, două microcontrolere, se va realiza conversia datelor paralele, pe care le procesează un microcontroler, într-o formă serială, apoi vor fi transmise sub această formă prin intermediul pinului de transmisie Tx și recepționate de către entitatea receptoare pe pinul Rx, unde vor fi convertite la loc sub formă paralelă pentru a fi procesate. În figura 2.26, este ilustrată conexiunea pinilor în cazul comunicației seriale asincrone. Pentru a fi posibilă realizarea comunicației între două entități hardware, pinul Tx al primului UART trebuie să fie legat la pinul receptor Rx al celuilalt, iar transmițătorul celui de-al doilea dispozitiv UART trebuie să fie legat la pinul Rx, al primei unități receptoare UART.

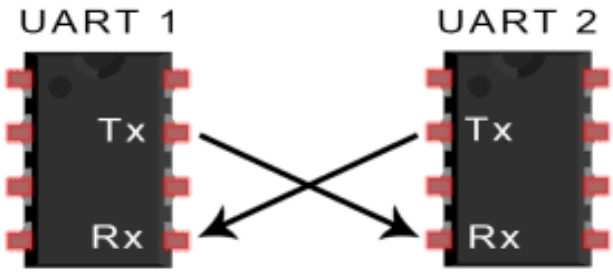



*Figura 2.26: Comunicația serială asincronă; Sursa: [19]*

Fiind o comunicație asincronă, ce se realizează fără a utiliza un semnal de ceas, ce sincronizează transmisia de date între două entități, controlul datelor se va realiza prin adăugarea unor biți suplimentari, pentru marcarea începerii transmiterii, finalizarea acesteia, dar și controlul transmiterii corecte a acesteia. Transmisia și recepția datelor se realizează la aceeași frecvență, cunoscută sub denumirea de rată de transmisie/recepție a biților. Datele ce sunt transmise prin intermediul comunicației asincrone seriale sunt organizate în pachete de date. Fiecare pachet conține un bit de start, un cadru de date, un bit de paritate (opțional) și un bit de stop. Pentru a începe transmisia serială asincronă , pe linia corespunzătoare pinului Tx a primei unități UART este transmis un bit de start, iar când acesta este recepționat de pinul Rx a celei de-a doua unități UART, aceasta începe citirea la frecvența corespunzătoarea ratei de transmisie a datelor. Cadrul de date transmis poate conține un număr de la 5 la 9 biți. Opțional, pentru a verifica dacă transmisia

s-a realizat fără erori se poate transmite și un bit de paritate. Pentru a marca sfârșitul transmisiei, unitatea UART care a inițiat comunicația va transmite bitul de stop.

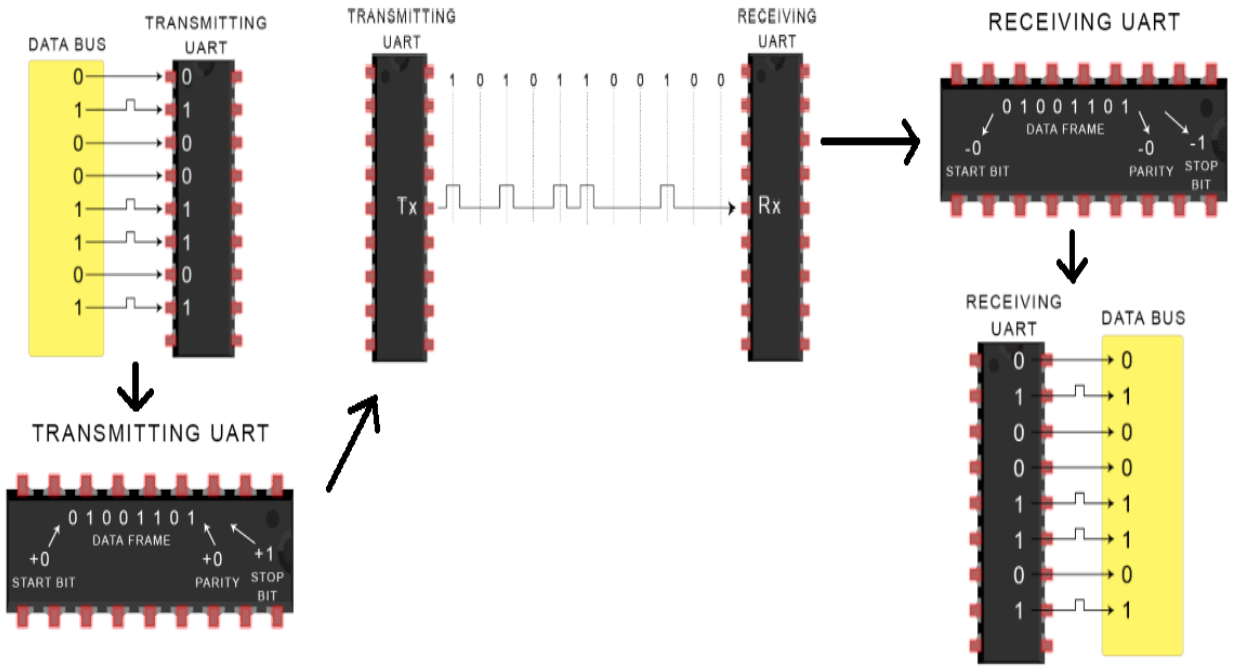



*Figura 2.27: Modul de funcționare al comunicației seriale asincrone; Sursa: [19]*

În figura 2.27 sunt ilustrați pașii realizați în cadrul transmisiei seriale asincrone. Inițial, unitatea UART primește datele în paralel de pe magistrala de date, urmând ca aceasta să le convertească într-un pachet de date serial, adăugând biții de start, de paritate și de stop. Pachetul de date este transmis de către unitatea UART ce inițiază transmisia către unitatea receptoare, prin intermediul unei conexiuni între portul Tx al transmițătorului și portul Rx al receptorului. Unitatea de la recepție va primi datele la o frecvență prestabilită, urmând procesul invers de la transmisie, înlăturând biții de control și convertind cadrul de date serial într-unul paralel pe care îl transferă mai departe pe magistrala de date a celei de-a doua entități.

Principalul dezavantaj al comunicației seriale asincrone este faptul că nu suportă comunicația de tip „multi-master” sau „multi-slave”.

Curentul maxim suportat de pinii de intrare și ieșire este de 40 mA. Memoria flash, memoria unde programul scris de utilizator este stocat are o capacitate de 32KB, memoria SRAM („static random access memory”), unde sunt create și stocate variabilele create este de 2KB, iar memoria EEPROM este de 1KB. Plăcuța de dezvoltare Arduino Nano, operează la frecvența de 16MHz, frecvență ce este generată de un oscilator cu cristal de cuarț. Oscilatorul cu cristal de cuarț funcționează pe principiul piezoelectric invers, care se manifestă prin deformarea materialelor piezoelectrice la aplicarea unor tensiuni electrice din exterior.

Pentru conversia semnalelor analogice în semnale digitale, microcontrolerul ATmega328p conține un convertor analog-numeric cu aproximații succesive pe 10 biți. Convertorul analog-numeric este un circuit care are rolul de a transforma un semnal analogic într-un semnal digital, adică într-o mărime numerică. Această mărime numerică este o aproximare a semnalului analogic.

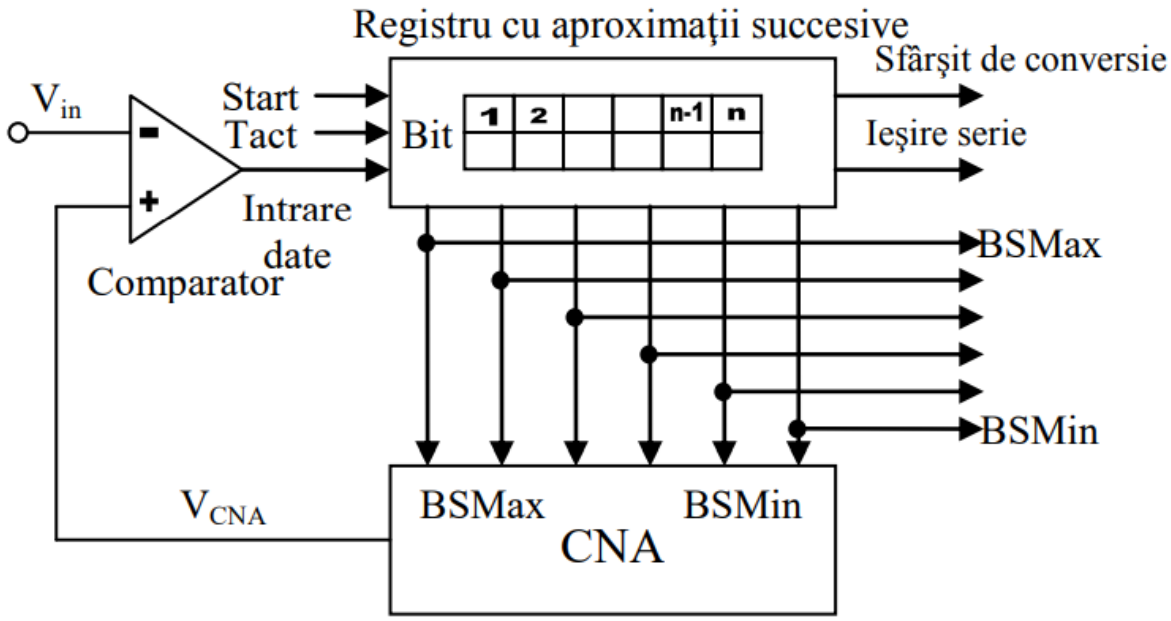



*Figura 2.28: Convertorul analog-digital cu aproximații succesive; Sursa: [20]*

În figura 2.28 este ilustrată schema de bază a convertorului analog-digital cu aproximații succesive. Schema conține un convertor numeric-analog în bucla de reacție care este comandat de un registru cu aproximații succesive (RAS), cu denumirea în engleză de Succesive Approximation Register (SAR). La momentul începerii conversiei, registrul cu aproximații succesive conține toți biții 0, mai puțin bitul cel mai semnificativ (BSMax) care este setat 1. Ieșirea convertorului numeric-analog este conectată la intrarea comparatorului, fiind comparată cu semnalul de intrare.

Modul de funcționare este următorul: la început, prestabilirea impusă de registrul cu aproximații succesive face ca semnalul analogic de la intrarea $V_{in}$, să fie comparat cu jumătate din tensiunea maximă de intrare, $\frac{Vmax}{2}$. Dacă semnalul de intrare va fi mai mare decât jumătate din tensiunea maxima , al doilea bit semnificativ va fi setat 1. În schimb, dacă semnalul de intrare va fi mai mic decât jumătate din tensiunea maximă de intrare, bitul cel mai semnificativ din registrul cu aproximații succesive va fi setat 0, iar cel de-al doilea bit semnificativ va fi setat 1. Astfel, tensiunea echivalentă produsă de convertorul digital-analog va fi echivalentă pentru $\frac{Vmax}{4}$, fiind din nou comparată cu semnalul de la intrare. Acest proces continuă, semnalul de intrare fiind comparat cu semnalul provenit din bucla de reacție, care reprezintă un semnal din ce în ce mai apropiat de valoarea semnalului de la intrare, până când se ajunge la bitul cel mai puțin semnificativ, notat cu BSMin sau LSB.

În continuare voi realiza un exemplu de calcul pentru a vedea cum funcționează un convertor analog-numeric cu aproximații succesive. Voi alege un convertor analog-numeric pe 4 biți pentru simplitate. Astfel, pentru tensiunea de intrare voi alege o valoarea de 1.2 V, iar pentru tensiunea de referință, valoarea de 2.56 V. Inițial, în registrul cu aproximații succesive va fi setat cel mai semnificativ bit cu 1, iar ceilalți biți cu 0. Astfel, tensiunea de corespunzătoare de la ieșirea convertorului numeric-analogic va fi egală cu $V_{CNA} = \frac{V_{REF}}{2}\left(\frac{b1}{2^0} + \frac{b2}{2^1} + \frac{b3}{2^2} + \frac{b4}{2^3}\right) = \frac{V_{REF}}{2} = 1{,}28V$. Deoarece $V_{in} < V_{CNA}$ bitul cel mai semnificativ va fi schimbat în 0, iar următorul cel mai semnificativ bit va fi setat 1, astfel în registrul de aproximații succesive se va găsi 0100. Cu această nouă valoare, se stabilește tensiunea buclei de reacție ce va fi comparată cu tensiunea de intrare la valoarea $V_{CNA} = \frac{V_{REF}}{2}\left(0 + \frac{1}{2} + 0 + 0\right) = \frac{V_{REF}}{4} = 0{,}64V$. La această iterație semnalul de la intrare $V_{in} > V_{CNA}$, ceea ce înseamnă ca procesul continuă prin setarea următorului bit 1. Astfel, pentru următoarea iterație în registrul cu aproximații succesive se va găsi valoarea 0110, corespunzătoare tensiunii din bucla de reacție $V_{CNA}=\frac{V_{REF}}{2}\left(0 + \frac{1}{2} + \frac{1}{4} + 0\right) = \frac{3*V_{REF}}{4}$ = 1,92V. În cadrul acestei iterații $V_{in}<V_{CNA}$, deci bitul setat anterior în 1 va fi schimbat în 0, iar ultimul bit rămas va fi setat 1. Astfel, în registrul cu aproximări succesive se va găsi valoarea 0101, ceea ce corespunde valorii $V_{CNA}= \frac{V_{REF}}{2}\left(0 + \frac{1}{2} + 0 + \frac{1}{8}\right) = \frac{5*V_{REF}}{8}$=1,6V. Pentru această ultimă iterație $V_{in}<V_{CNA}$ , ceea ce înseamnă că ultimul bit va fi 0 , astfel, în final, în registrul de aproximații succesive se va găsi valoarea 0100 corespunzătoare tensiunii 1.28V, care reprezintă cea mai bună aproximație a tensiunii $V_{in}$ de 1,2V realizată cu convertorul analog-numeric de 4 biți. Cu cât se folosește un convertor analog-numeric pe mai mulți biți, cu atât aproximația realizată va fi mai bună.

# 2.9 Blocul de alimentare

Pentru alimentarea robotului mobil am folosit doi acumulatori Li-Ion de la Sony. Tensiunea nominală a acumulatorului folosit este de 3,7V, ceea ce înseamnă că ansamblul serie realizat cu cei doi acumulator oferă o tensiune de 7,4V. Am conectat ansamblul realizat printr-un comutator ce este poate fi acționat mecanic de către utilizator atunci când dorește să pornească alimentarea robotului mobil. Ansamblul de alimentare se conectează pentru a alimenta modulul L298N, cu ajutorul căruia se controlează motoarele mașinii. Modulul L298N conține un stabilizator de tensiune de 5V, ceea ce oferă tensiunea necesară alimentării plăcii Arduino Nano. Pentru a nu avea probleme de instabilitate, tensiunea generată la intrarea modulului driver de motoare trebuie să nu fie mai mică de 7V pentru ca stabilizatorul intern să poată genera la ieșire tensiunea constantă de 5V necesară alimentării plăcii Arduino Nano și a celorlalte elemente utilizate. În realitate, cele două acumulatoare pot fi încărcate cu ajutorul unui încărcător special până la tensiunea de 4,1V, ceea ce oferă un total de 8,2V, tensiune ce este suficientă astfel încât stabilizatorul intern al modulului driver de motoare să furnizeze tensiunea de 5V constantă necesară alimentării plăcii Arduino Nano și a celorlalte componente atașate în cadrul robotului mobil descris în acest capitol. Comutatorul ON/OFF utilizat este unul cu menținere, adică este un comutator ce rămâne în starea în care a fost acționat ultima oară, până la următoarea acționare din exterior.

*Figura 2.29: Acumulatorul Li-Ion și comutatorul utilizat; Sursa: Optimus Digital*

# Capitolul 3 Parcarea inteligentă Noțiuni teoretice

În cadrul proiectului, mi-am propus ca robotul mobil să poată parca autonom în cadrul unui spațiu special amenajat. Astfel, am conceput și implementat o parcare inteligentă, capabilă să recepționeze date de la mediul exterior privind disponibilitatea locurilor de parcare, să le prelucreze, să afișeze aceste date pe un afișaj LCD și totodată, să le comunice robotului mobil. În cadrul realizării parcării inteligente, am utilizat patru senzori infraroșii ce au rolul de identifica prezența sau absența unei mașini atât la intrarea în parcare cât și în locurile de parcare disponibile, leduri de diferite culori pentru marcajul accesului în parcare, un LCD pentru afișarea informațiilor privind disponibilitatea locurilor de parcare, un servomotor pentru acționarea barierei ce permite accesul în parcare, o placă de dezvoltare Arduino UNO R3, ce are rolul de a prelucra datele primite de la senzori și de a controla comportamentul celorlalte elemente utilizate, un modul de comunicație wireless nRF24L01 prin intermediul căruia datele vor fi transmise către robotul mobil, o placă prototip compatibilă cu placa de dezvoltare Arduino UNO R3, utilizată pentru a putea integra firele de conexiune ale elementelor utilizate în cadrul parcării într-o formă cât mai compactă și un bloc de alimentare. Cu excepția modulului wireless de comunicație care este dezvoltat în subcapitolul 2.7 și al afișajului LCD care este dezvoltat în subcapitolul 4.2, toate celelalte componente vor fi prezentate în cele ce urmează.

## 3.1 Servomotorul

Servomotorul reprezintă un ansamblu realizat dintr-un motor, controlat prin intermediul unui servomecanism. Servomotoarele sunt dispozitive electronice cu o mare arie de aplicabilitate ce sunt utilizate cu rolul de a controla cu precizie poziția arborelui și unghiul în care acesta se deplasează. Servomotoarele se folosesc în aplicații industriale de mare precizie, dar totodată, există și servomotoare de uz general, ce pot fi achiziționate la prețuri reduse, utilizate în diferite aplicații ce au ca scop controlul precis al distanței sau unghiul în care se mișcă anumite elemente. Acestea pot fi folosite la construcția roboților de jucărie, la acționarea mecanismelor de deschidere și închidere, brațe mecanice, CD-playere, etc.

În cadrul proiectului prezent, am utilizat un astfel de servomotor pentru acționarea barierei de la intrarea în parcarea inteligentă construită. Servomotorul are în principal următoarele componente: un motor de curent continuu, un potențiometru și un circuit de control. În general, un servomotor conține o cutie de viteze încorporată (sistem de reducție), realizată din roți dințate, ce au rolul de a reduce viteza de rotație a motorului de curent continuu din interiorul ansamblului servomotor. Brațul de la ieșirea servomotorului, numit și arbore de ieșire nu are o mișcare continuă ca în cazul motoarelor de curent continuu despre care am discutat în secțiunea 2.1, ci acestea se deplasează în pași ceea ce reprezintă principalul avantaj oferit de aceste dispozitive. Potențiometrul din interiorul ansamblului este utilizat ca un senzor de feedback pentru ajustarea poziției arborelui de ieșire. Acesta este cuplat cu arborele motorului(axul motorului) prin intermediul roților dințate.

Circuitul de control din interiorul unui servomotor  are rolul de monitoriza și de a ajusta poziția servomotorului până când axul extern (arborele) se află exact în poziția dorită. Pentru a realiza acest lucru, circuitul de control utilizează semnalul primit prin intermediul potențiometrului și semnalul de control. Structura clasică a unui potențiometru este prezentată în figura 3.1:

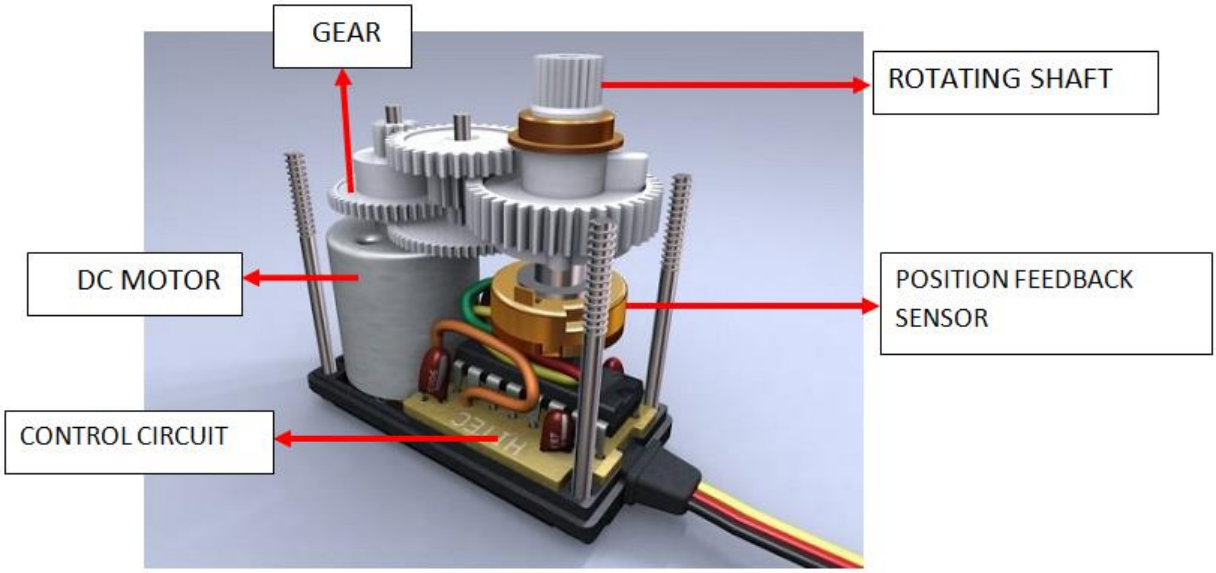



*Figura 3.1: Structura unui servomotor; Sursa [21]*

În figura 3.1 se observă că servomotoarele se conectează prin intermediul a trei fire, firul negru fiind conectat la masă, firul roșu reprezentând conexiunea de alimentare, iar firul galben reprezintă firul de comandă. Tot în figura 3.1 se observă ca acest ansamblu de trei fire este conectat la circuitul de comandă („control circuit”), circuit ce se conectează la motorul de curent continuu („DC motor”) și la senzorul de feedback reprezentat de potențiometru („position feedback sensor”). De asemenea, se observă că potențiometrul și motorul de curent continuu se conectează prin intermediul roților dințate la axul rotativ de la exterior, pe care îl întâlnim adesea cu denumirea de arbore.

Modul de funcționare ce stă la baza servomotoarelor este următorul: circuitul de control compară cele două semnale primite prin intermediul firului de comandă și prin intermediul senzorului de feedback, reprezentat de către potențiometru. Acestea sunt interpretate ca și tensiuni, iar dacă diferența de tensiune între cele două este zero, circuitul de control interpretează că semnalul primit prin intermediul firului de comandă coincide cu poziția curentă a arborelui (axului rotativ); în acest caz nu se acționează motorul de curent continuu. Dacă diferența de tensiune are o valoare pozitivă sau negativă, motorul de curent continuu va fi acționat, acesta învârtindu-se spre dreapta sau spre stânga. Acest proces se cheamă proces de corectare a erorilor. Viteză de rotație depinde de diferența dintre cele două tensiuni, cu cât valoarea tensiunii este mai mare cu atât motorul se va roti mai rapid. Motorul de curent continuu va acționa în cadrul procesului de rotație mecanismul de roți dințate, care vor pune în mișcare axul rotativ de la exterior, adică arborele. Pe măsură ce arborele își schimbă poziția va modifica valoarea potențiometrului din bucla de reacție, iar această modificare va fi trimisă către circuitul de control pentru a putea fi monitorizată de către

acesta. În momentul când diferența de tensiune va fi zero, circuitul de control va opri semnalul de acționare pentru motorul de curent continuu, deoarece brațul arborelui a ajuns în poziția dorită.

Semnalul trimis către servomotor este un semnal PWM, adică un semnal cu lungime variabilă a impulsului. În funcție de lungimea pulsului, circuitul de control va ști direcția în care trebuie să mute arborele. De exemplu, pentru un impuls de 1,5 ms, arborele va fi direcționat către poziția de 90 grade. Pentru un impuls de 1ms, arborele va fi rotit în sens contrar acelor de ceasornic, către poziția de 0 grade, iar pentru un semnal cu durata impulsului de 2ms, arborele va fi rotit în sensul acelor ce ceasornic către poziția de 180 grade. [21]

Pentru acționarea barierei de la intrarea în parcare am utilizat un servomotor, SG90, pe care l-am alimentat la un ansamblu de patru baterii de 1,5V. Am ales acest servomotor datorită dimensiunilor de aproximativ 22.2 x 11.8 x 31mm, datorită greutății reduse și totodată, datorită costului redus. Modulul utilizat este prezentat în figura 3.2.


*Figura 3.2: Micro Servomotor; Sursa: Optimus Digital*

# 3.2 Dioda electroluminiscentă

Dioda electroluminiscentă (LED) este o diodă semiconductoare ce emite lumină la polarizarea directă a joncțiunii p-n. Acestea emit fotoni ca urmare a recombinării purtătorilor de sarcină. Fotonul reprezintă o particulă emisă de către LED a cărei lungime de undă ($\lambda$) depinde de diferența de energie dintre cele două nivele energetice (banda de conducție și banda de valență). Electronii din banda de conducție sar în banda de valență și se recombină cu un gol. Procesul de recombinare poate să fie de două tipuri: recombinare radiativă sau recombinare neradiativă. Recombinarea în urma căreia s-a emis un foton poartă numele de recombinare radiativă. Fenomenul de recombinare neradiativă nu are ca rezultat emisia unui foton, ci energia este absorbită în material ducând la încălzirea structurii. Raportul dintre numărul de recombinări radiative și numărul total de recombinări (recombinări radiative + recombinări neradiative) se numește eficiență cuantică. Astfel, folosindu-se diferite combinații de materiale(GaAs, GaInAs, GaAs, etc.) rezultă diferite energii, în funcție de diferența dintre nivelele energetice, ceea ce duce la obținerea anumitor lungimi de undă.

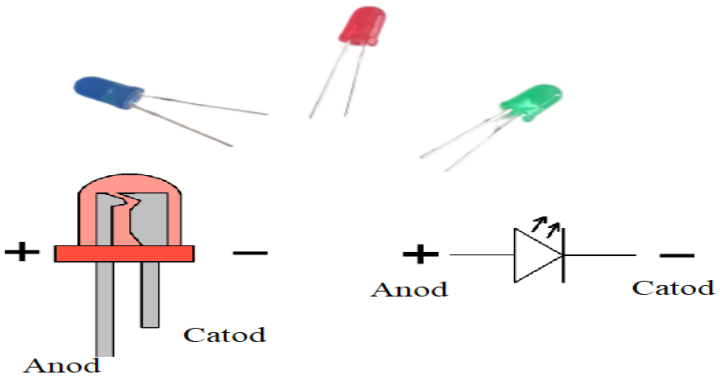



*Figura 3.3: Dioda electroluminiscentă (LED)*

În cadrul lucrării, la realizarea parcării am utilizat patru leduri, cu diametru de 5mm, de diferite culori. Aceste dispozitive au fost utilizate cu scopul de iluminare a parcării sau cu scopul marcării permisiunii sau interzicerii accesului în parcare. Astfel, pentru iluminare s-au utilizat două leduri, un LED albastru și unul alb, pentru marcajul permisiunii accesului în parcare s-a utilizat un LED de culoare verde, iar pentru marcajul interzicerii accesului în parcare s-a utilizat un LED roșu.

Ledurile sunt diode care emit radiație necoerentă. Este evident faptul că lungimea de undă nu va avea o singură valoare, ci se va situa într-o anumită plajă de valori, deoarece fotonii ca lungime de undă provin din recombinări electron-gol, cu electronii având energii ușor diferite. Astfel, lungimea de undă emisă de către un LED are o distribuție de tip Gaussian. O astfel de reprezentare poate fi văzută în figura 3.4.

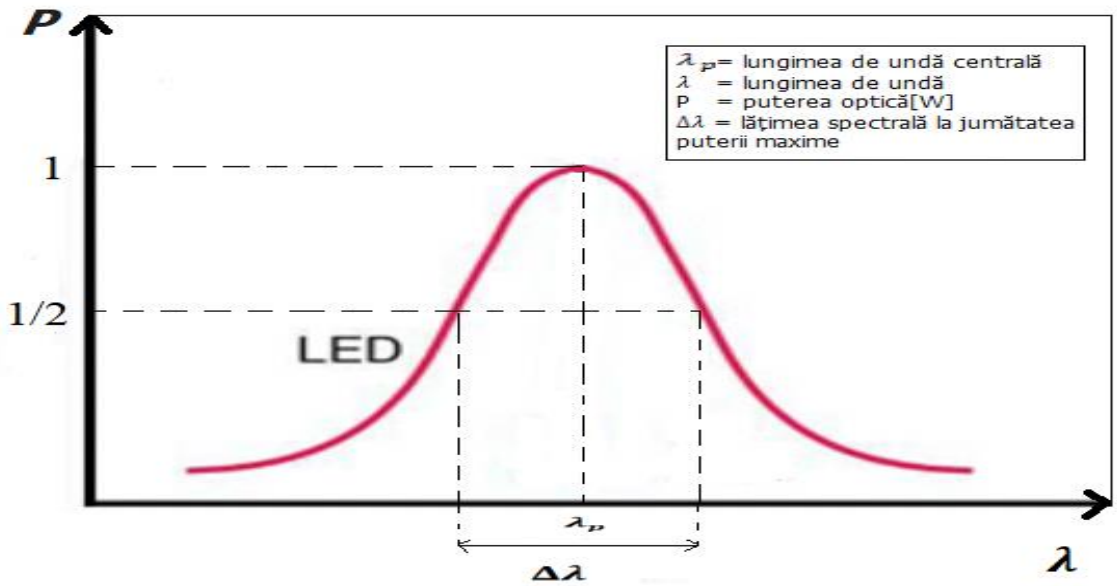



*Figura 3.4: Spectrul lungimilor de undă emis de LED*

Parametrul caracteristic cel mai important al dispozitivelor optoelectronice de tip LED este lungimea de undă centrală, notată cu $\lambda_p$. Aceasta reprezintă lungimea de undă corespunzătoare puterii optice maxime. Pentru LED-ul roșu, valoarea lungimii de undă centrală se află în jurul valorii de 625 nm, pentru LED-ul verde în jur de 525 nm, iar pentru LED-ul albastru în jur de 470 nm. În cazul ledului alb, acesta nu este un LED cu o structură simplă ca a celor prezentate anterior, ci el provine dintr-un LED albastru ce are o structură care emite fotoni cu o lungime de undă corespunzătoare culorii albastre, iar pe deasupra avem o depunere de fosfor galben, ceea ce aduce o emisie secundară, astfel, în ansamblu, emisia totală să fie alb.

În cazul fiecărui LED, pentru o utilizare corespunzătoare am luat în considerare amplasarea unui rezistor în serie cu acesta, rezistor ce are rolul de a limita curentul ce străbate structura LED. În lipsa utilizării unui rezistor, ledul poate suferi modificări ireversibile ale structurii, modificări ce pot duce chiar la distrugerea completă a acestuia. Valoarea rezistorului se va determina în funcție de valoarea curentului direct ce străbate structura LED. În continuare, voi arata un exemplu de calcul pentru determinarea valorii rezistorului utilizat în cazul conectării diodei LED.

În circuitul din figura 3.5, aplicând legea a doua a lui Kirchhoff, rezultă ecuația 3.1 a) și 3.1 b):

$$V = I * R + V_{LED} \qquad (a)$$

$$R = \frac{V - V_{LED}}{I} \qquad (b)$$

*Ecuaţia 3.1: Determinarea valorii rezistorului utilizat*

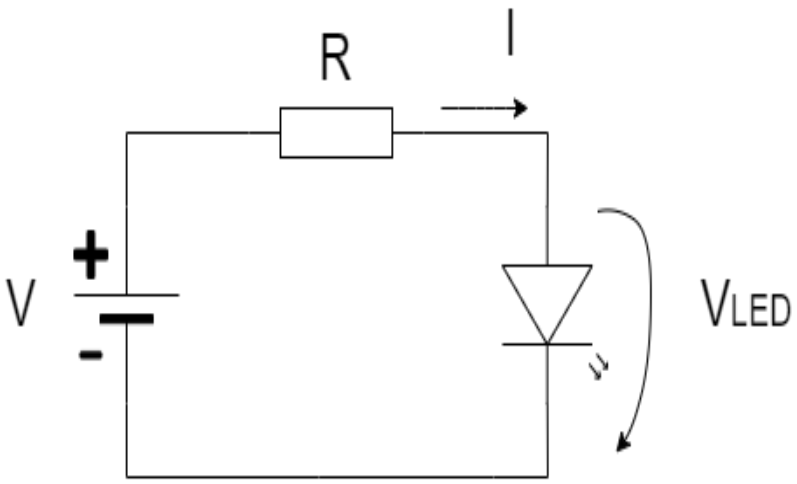



*Figura 3.5: Determinarea valorii rezistorului utilizat*

Pentru o diodă LED a cărui spectru emis corespunde radiații luminoase de culoare roșie, valoarea tensiunii de la bornele acesteia are valori tipice în jurul a 2,2V și un curent direct ce străbate structura în jurul valorii de 25mA. Ținând cont că tensiunea de alimentare se presupune constantă la valoarea de 5V, valoarea determinată pentru rezistor, aplicând ecuația 2.2 este de aproximativ 112 Ω. Așadar, am ales un rezistor cu o valoare nominală a rezistenței de 220 Ω, valoare tipică întâlnită pe piață, mai mare decât valoarea determinată anterior. Pentru conectarea ledului alb și pentru conectarea ledului albastru, datorită valorilor mai ridicate pentru tensiunea de deschidere, am utilizat rezistoare a căror rezistență nominală este de 100 Ω.

# 3.3 Senzor Infraroșu de obstacole

În cadrul parcării am realizat un sistem capabil să transmită informații robotului mobil, informații referitoare la disponibilitatea locurilor de parcare. Colectarea datelor privind prezența sau absența unei mașini am realizat-o utilizând modulul senzor infraroșu de obstacole prezentat în figura 3.6:

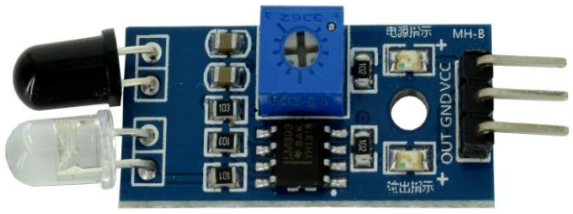



*Figura 3.6: Modul senzor infraroșu de obstacole; Sursa [22]*

Am ales utilizarea modulului senzor infraroșu de obstacole datorită compatibilității sale cu sistemul creat. Acesta prezintă o serie de avantaje precum: gama tensiunilor de alimentare cuprinsă între 3V și 5V, posibilitatea de detecție a obstacolelor pe o distanță cuprinsă între 2cm și 30 cm, unghiul de observare al obstacolului de 35°, output digital, precum și un cost redus de achiziționare. În cadrul proiectului, am folosit patru astfel de module, câte unul pentru a monitoriza prezența sau absența unui obstacol în cadrul fiecărui loc de parcare și un modul la intrarea în parcare. Modulul este construit pe baza a două elemente principale: elementul activ pe baza căruia este depistat un posibil obstacol și elementul de control ce oferă la ieșirea circuitului un output digital, reprezentat de nivel logic „0”, atunci când este întâlnit un obstacol sau nivel logic „1” , atunci când modulul nu detectează obstacole. Totodată, modulul conține două diode electroluminiscente cu rolul de a indica prezența unui obstacol și alimentarea corespunzătoare a modulului și un potențiometru, cu ajutorul căruia se poate modifica nivelul de referință, astfel încât să se ajusteze sensibilitatea, ceea ce duce la modificarea distanței de detecție a obstacolelor în gama cuprinsă între 2cm și 30 cm.

Elementul activ al modulului senzor infraroșu de obstacole este reprezentat de ansamblul format din transmițătorul infraroșu și receptorul infraroșu. Sursa de emisie a radiației infraroșii este reprezentată de o diodă electroluminiscentă cu diametrul de 5mm. Lungimile de undă emise de către dioda electroluminiscentă se află în domeniul infraroșu apropiat, domeniu caracterizat de lungimile de undă cuprinse între 700nm și 1400nm. Valoarea tipică a tensiunii de alimentare pentru dioda utilizată este de 1,2 V , iar lungimea de undă centrală emisă de dioda LED are o valoare tipică de 940nm [23]. Receptorul infraroșu este reprezentat de către o fotodiodă, ce are rolul de a detecta radiația emisă de către emițător. Fotodioda, ca și dispozitiv, reprezintă un senzor de radiație optică, ce are rolul de a prelua radiația luminoasă incidentă pe suprafața acesteia și a o transforma prin intermediul efectului fotovoltaic în energie electrică. Acestea sunt realizate dintr-o joncțiune de siliciu p-n și sunt încapsulate într-o capsulă transparentă pentru a permite radiației luminoase recepționate să cadă direct pe joncțiunea p-n. Joncțiunea p-n este formată din materialul din stratul p de la suprafața activă și materialul din stratul n din substrat, aceasta funcționând ca un convertor fotoelectric. Lungimea de undă a radiației luminoase de detectat este impusă de grosimea stratului p. [24] Astfel, modulul conține o fotodiodă de 5mm, a cărei sensibilitate spectrală are o valoare tipică de 940nm și poate fi alimentată la o tensiune inversă de maxim 60V. Valoarea tipică a curentului de întuneric pentru aceasta diodă este de ordinul nanoamperilor. [25]

Principiul de funcționare pentru depistarea unui obstacol este următorul: transmițătorul reprezentat de către dioda electroluminiscentă emite radiație infraroșie în mediul exterior. O parte din această radiație, la întâlnirea unei suprafețe (obstacol) se reflectă, urmând să fie incidentă pe suprafața fotodiodei. În funcție de intensitatea radiantă pe suprafața fotodiodei, prin structura acesteia se va stabili un fotocurent ce va fi preluat mai departe de blocul de control. În figura următoare este ilustrat modul de funcționare al modulului senzorului infraroșu de obstacole prezentat anterior.

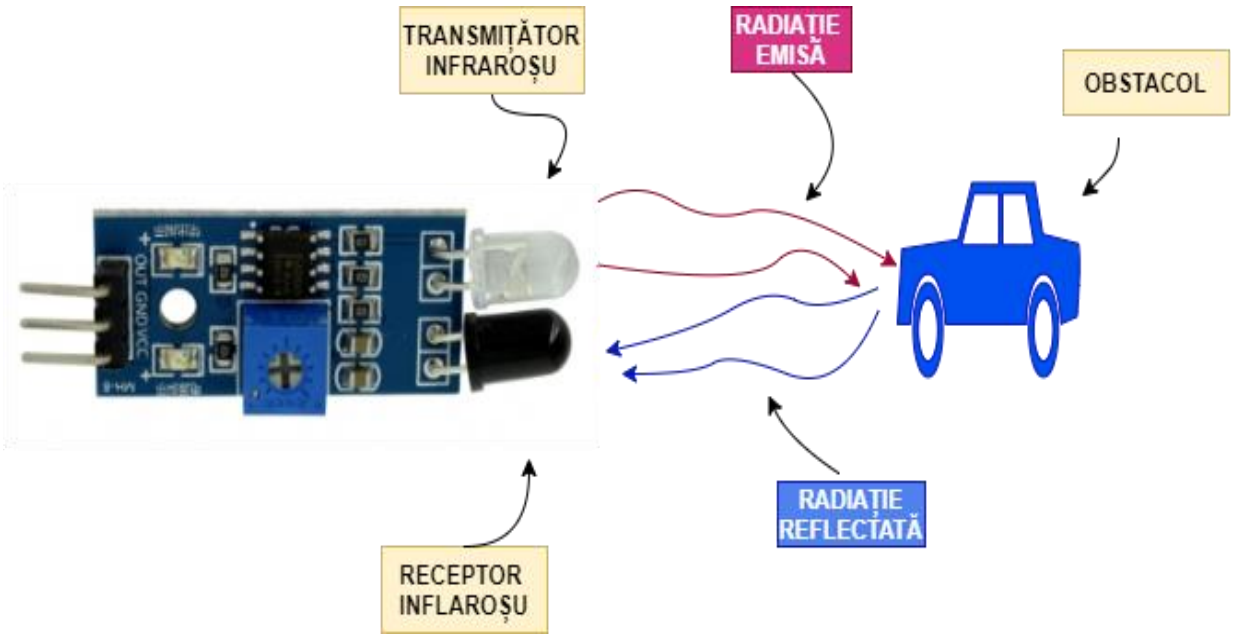



*Figura 3.7 Principiul de funcționare pentru depistarea obstacolelor*

Trebuie menționat faptul că suprafețele negre nu pot fi detectate de către modulul senzorului infraroșu de obstacole, datorită faptului că aceste suprafețe absorb radiația incidentă pe acestea, astfel procesul de reflexie va fi aproape inexistent și pe structura fotodiodei nu va mai exista radiație incidentă rezultată din reflexia de pe suprafața obstacolului. Dacă suprafața pe care este incidentă radiația emisă de transmițătorul infraroșu este una reflexivă( suprafețe albe sau de culori deschise), va exista reflexie la nivelul suprafeței, reflexie ce va fi depistată de către fotodiodă. În cadrul proiectului se va folosi un material reflexiv atașat pe suprafața mașinii, deoarece șasiul utilizat are o culoare neagră, făcând astfel imposibilă reflexia radiației incidente la nivelul acestuia.

Schema tipică a modulului senzorului infraroșu de obstacole este prezentată în figura 3.8:

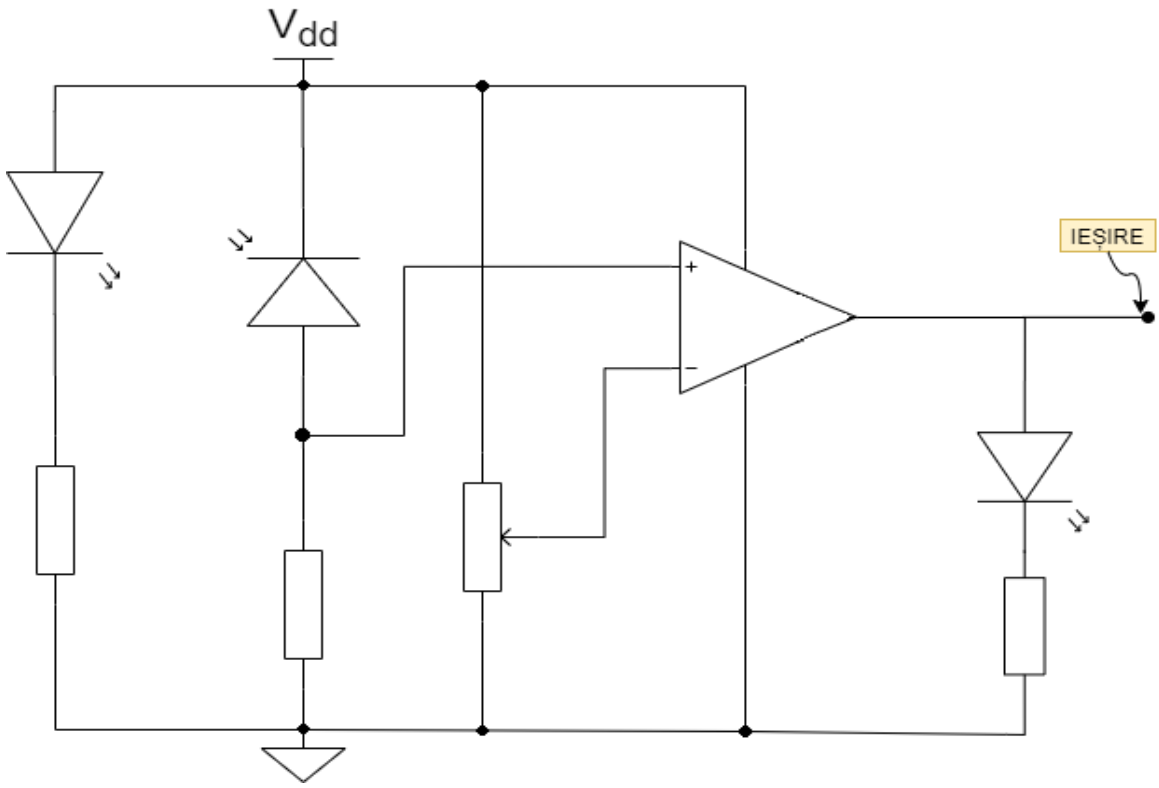



*Figura 3.8: Schema tipică a modulului senzor infraroșu de obstacole; Sursa [26]*

Schema tipică conține elementele menționate anterior: dioda electroluminiscentă al cărui spectru de emisie se află în domeniul corespunzător radiațiilor infraroșii, fotodioda, comparatorul LM393 și un potențiometru pentru ajustarea distanței de detecție. Pe prima ramură a circuitului este prezentă dioda semiconductoare înseriată cu un rezistor ce are rolul de a limita curgerea curentului prin dioda electroluminiscentă. Pe ramura a doua a circuitului se află fotodioda înseriată cu un rezistor, ansamblu ce este conectat la intrarea comparatorului. A treia ramură a circuitului conține un potențiometru cu ajutorul căruia va fi reglată tensiunea de referință de la intrarea comparatorului. În final, la ieșirea comparatorului este prezent ansamblul realizat dintr-un rezistor înseriat cu o diodă LED, diodă ce emite radiație luminoasă atunci când modulul a detectat un obstacol.

Așadar, funcționalitatea completă a modulului este următoare: radiația emisă de către transmițătorul infraroșu fiind reflectată de suprafața obstacolului este indecentă pe fotodiodă și generează un curent prin ramura a doua a circuitului, determinând o cădere de tensiune la bornele rezistorului înseriat cu fotodioda. Comparatorul LM393 preia această variație a tensiunii din ramura a doua și o compara cu tensiunea de referință. Dacă căderea de tensiune pe rezistorul din ramura a doua a circuitului este mai mare decât tensiunea de referință, atunci la ieșirea comparatorului se va genera un semnal corespunzător nivelului "1" logic și dioda de la ieșirea comparatorului va emite radiație luminoasă, semnalizând prezența unui obstacol. În caz contrar, dacă căderea de tensiune pe rezistorul înseriat cu fotodioda este mai mică decât tensiunea de referință, ieșirea comparatorului va fi un semnal corespunzător nivelului "0" logic, iar dioda LED de la ieșirea circuitului nu va emite radiație luminoasă. Distanța de detecție a modulului poate fi variată reglând potențiometrul, ceea ce va genera o nouă tensiune de referință. [26]

## 3.4 Placă de dezvoltare compatibilă cu Arduino UNO R3

Procesarea datelor provenite de la senzorii din parcare și controlul celorlalte elemente ale parcării precum ledurile, servomotorul sau afișajul LCD se realizează cu ajutorul plăcii de dezvoltare Arduino UNO R3. Aceasta are la bază microcontrolerul ATmega328p. Aceasta este dispusă cu 14 pini digitali de intrare sau ieșire, dintre care șase pot genera semnal PWM la ieșire, conține șase pini analogici, o memorie flash de 32KB, memorie SRAM de 2KB și memorie EEPROM de 1KB. Memoria EEPROM și memoria flash sunt memorii nevolatile, informația stocată pe acestea persistă și după ce alimentarea modulului a fost oprită, iar memoria SRAM este de tip volatil, adică informația stocată pe aceasta va fi ștearsă atunci când alimentarea este oprită. Placa poate fi alimentată utilizând o conexiune USB direct la computer, sau prin intermediul unei surse externe de tensiune ce poate fi conectată fie prin intermediul unei mufe DC de diametru interior de 2.1mm fie direct prin conectarea la pinul $V_{in}$ al plăcii. Tensiunea de alimentare externă recomandată este cuprinsă în gama de 7V-12V, cu toate că aceasta suportă o alimentare externă cuprinsă între 6V și 20V. Dacă tensiunea de alimentare externă este mai mică de 7V, pinul de 5V poate deveni instabil, iar dacă tensiunea de alimentare externă depășește 12V, regulatorul intern de tensiune se poate supraîncălzi generând astfel deteriorări la nivelul plăcii. [27]

Având același microcontroler ca și Arduino Nano, doar că diferă capsula acestuia, rămân valabile informațiile prezentate în cadrul subcapitolului 2.8. Față de plăcuța prezentată în cadrul subcapitolului 2.8 și utilizată în realizarea robotului mobil, placa de dezvoltare Arduino UNO R3, utilizează pentru conversia datelor, primite prin USB, microcontrolerul ATmega16U2. Acesta convertește datele primite prin USB de la computer în date seriale și comunică microcontrolerului principal ATmega328p aceste date prin intermediul comunicației seriale asincrone, UART. De asemenea, placa de dezvoltare Arduino UNO R3 prezintă pini suplimentari pentru magistrala de date (SDA) și pentru magistrala de ceas (SCL), folosite în cadrul protocolului de comunicație $I^2C$. Totodată, placa mai prezintă și un pin suplimentar, IOREF, care are rolul de a furniza referința de tensiune cu care funcționează microcontrolerul.

Pentru a conecta toate firele distribuite la nivelul parcării într-o formă cât mai compact, am utilizat o placă de prototip compatibilă cu Arduino UNO R3, ce conține pin de tip mamă, pentru a putea conecta componentele, dar și un spațiu unde se pot face lipituri pentru diferite componente direct la nivelul plăcii. Plăcuța conține și două leduri cu două butoane, pe care nu le-am folosit în cadrul proiectului. Modul în care am folosit această plăcuță va fi dezvoltat în subcapitolul 5.2.

*Figura 3.9: Placa Arduino UNO R3 și placă prototip compatibilă cu aceasta ; Sursa: Optimus Digital*

# 3.5 Blocul de alimentare

Blocul de alimentare folosit în cadrul parcării este alcătuit dintr-o baterie Li-Ion conectată prin intermediul unui comutator la un modul ridicător de tensiune, boost DC-DC, ce oferă la ieșire tensiunea constantă de 5V necesară alimentării plăcii de dezvoltare Arduino UNO R3 și a celorlalte elemente utilizate în cadrul parcării. Comutatorul este unul cu reținere, acesta putând fi acționat manual de către utilizator atunci când se dorește alimentarea parcării. Totodată, am folosit și un ansamblu serie format din patru baterii de tip AA de 1,5V, pentru alimentarea separată a servomotorului ce acționează bariera de la intrarea în parcare.

În figura 3.10 este prezentat modulul DC-DC utilizat pentru ridicarea tensiunii primite de la acumulatorul Li-Ion la valoarea de 5V necesară alimentării plăcii de dezvoltare Arduino UNO R3 și a celorlalte componente utilizate în parcare.


*Figura 3.10:Modul DC-DC Boost; Sursa: Optimus Digital*

Tensiunea de intrare a modulului prezentat în figura 3.10 se poate afla în gama cuprinsă între 0,9 V și 5 V, iar tensiunea de la ieșirea acestuia este de 5V. Curentul de la ieșirea modulului poate ajunge la valori cuprinse între maxim 800 mA – 1000 mA. Acest modul este ideal pentru realizarea alimentării componentelor electrice și a plăcilor de dezvoltare cu microcontroler utilizate în cadrul proiectelor. Putând fi achiziționat la un cost redus și având în vedere faptul că prezintă o mufă USB, cu ajutorul modulului se poate realiza chiar și un încărcător pentru telefonul mobil

# Capitolul 4 Telecomanda Noțiuni teoretice

Pentru a oferi utilizatorului posibilitatea de a controla robotul mobil manual, în cadrul proiectului, am realizat o telecomandă ce are la bază un „shield” joystick compatibil cu placa de dezvoltare Arduino UNO R3. Totodată, ansamblul realizat conține și un afișaj LCD pe care vor fi afișate informațiile primite de la mașină, privind temperatura și umiditatea mediului ambiant. Telecomanda comunică cu robotul mobil prin intermediul modulului wireless descris în paragraful 2.7. Totodată, placa de dezvoltare a fost deja discutată în paragraful 3.4. În cadrul capitolului 4, vor fi descrise celelalte elemente utilizate pentru realizarea telecomenzii.

## 4.1 Shield Joystick

Pentru realizarea telecomenzii am achiziționat un „shield” joystick compatibil cu Arduino UNO, ce este ideal pentru construirea unui controler pentru robotul mobil. „Shield-ul” joystick conține un modul joystick cu două axe, șase butoane fără reținere (patru pentru direcțiile de deplasare și două pentru selecție), un comutator de selecție al alimentării între 3,3V și 5V, pini mamă pentru a facilita conexiunea diferitelor elemente precum modulul nRF24L01 sau pini tată pentru conexiunea privind comunicația $I^2C$. În figura 4.1 este prezentat modulul achiziționat.

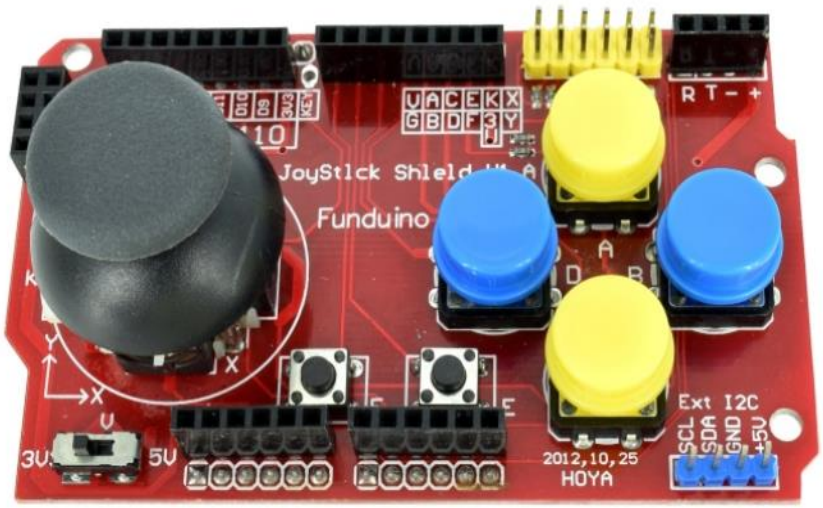



*Figura 4.1: Shield Joystick; Sursa: Optimus Digital*

Modulul joystick utilizat este un modul cu două axe, având practic la bază două potențiometre pentru citirea valorilor pentru cele două axe. Potențiometrul pentru citirea valorilor pe axa orizontală X, este conectat la pinul A0, iar potențiometrul utilizat pentru citirea valorilor pe axa Y este conectat la pinul A1. Valorile citite de la aceste două potențiometre se află în gama 0-1023 deoarece, microcontrolerul plăcii de dezvoltare cu care este compatibil acest modul conține un convertor analog-numeric pe 10 biți. Astfel, atunci când asupra modulului joystick nu se exercită nicio forță externă, acesta se va găsi în poziția de echilibru corespunzătoare valorii 512 pentru ambele axe. De asemenea, modulul joystick conține și un comutator ce poate fi acționat prin apăsarea acestuia.

Cele șase butoane atașate pe placă sunt numerotate cu litere mari de la A la F și pot fi atribuite pentru activarea diferitor funcții dorite de către utilizator. Toate cele șase butoane au rezistențe de „pull-up" asociate. Astfel, butonul A este conectat la pinul D2, butonul B este conectat la pinul D3, butonul C este conectat la pinul D4, butonul D este conectat la pinul D5, butonul E este conectat la pinul D6, iar butonul F este conectat la pinul D7. În cadrul proiectului, utilizatorul poate acționa butonul C pentru a acționa frâna, oprind astfel deplasarea mașinii, poate acționa butonul A pentru cel mult 4 secunde pentru a primi informații de la mașină privind temperatura și umiditatea mediului înconjurător, sau poate acționa butonul F pentru a opri comanda manuală a robotului mobil, inițializând totodată, comunicația între parcare și mașină.

Modulul nRF24L01 poate fi conectat direct la placa de dezvoltare prin intermediul acestui modul, deoarece shiled-ul joystick prezintă un conector compatibil cu modulul de comunicație wireless.

## 4.2 Afișaj LCD

Datele primite de la mașină prin intermediul modulului wireless, privind informațiile despre temperatura și umiditatea mediului înconjurător vor fi afișate pe un ecran LCD. LCD înseamnă „liquid-crystal display", ceea ce se traduce prin afișaj cu cristale lichide. Denumirea vine de la faptul că afișajul cu cristale lichide are la bază o matrice realizată din cristale lichide care au proprietatea de a influența direcția de polarizare a luminii atunci când asupra lor este acționată o tensiune electrică. În cadrul proiectului am utilizat un modul LCD 1602 cu interfață $I^2C$ și lumină de contrast albastră, ce are o tensiune de alimentare de 5V și un consum de curent de doar 1,1mA. Tensiunea de alimentare pentru lumina de contrast albastră este de 4,2V, iar consumul de curent este de 100mA.[28] Afișajul cu cristale lichide, oferă posibilitatea de a putea afișa 2 linii a câte 16 caractere, de aici și denumirea modulului LCD 1602.

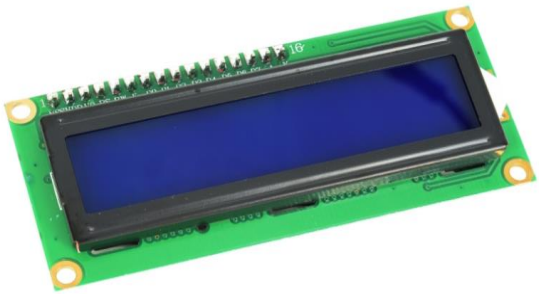


*Figura 4.2: Afișaj LCD ; Sursa: Optimus Digital*

Afișajul cu cristale lichide ilustrat în figura 4.2 este prevăzut cu 16 pini. Pinii sunt numerotați de la 1 la 16 și prezintă următoarele funcții: pinii $V_{ss}$ și $V_{dd}$ sunt utilizați pentru alimentarea LCD-ului, pinul $V_o$ este utilizat pentru ajustarea contrastului, pinul $R_s$ este utilizat pentru selecția registrului din memorie în care se vor scrie datele, pinul R/W este utilizat pentru selecția între modul de citire sau scriere, pinul E reprezintă pinul de activare, următorii 8 pini, numerotați de la D0 la D7, reprezintă pinii de date, iar ultimii doi inscripționați cu A, respectiv K, sunt utilizați pentru alimentarea luminii de fundal.

Deoarece modulul achiziționat conține un adaptor $I^2C$, pentru conectarea ansamblului la placa de dezvoltare Arduino sunt necesare doar patru conexiuni, conexiunea SDA, corespunzătoare magistralei de date, conexiunea SCL corespunzătoare semnalului de ceas, conexiunea pentru 5V și conexiunea pentru masă. Modulul conține și un potențiometru pentru reglarea contrastului afișajului cu cristale lichide.

# 4.3 Blocul de alimentare

Pentru alimentarea telecomenzii am utilizat un ansamblu realizat dintr-un acumulator Li-Po a cărui tensiune nominală este de 3,7V, conectat prin intermediul unui comutator cu reținere la un modul convertor ridicător de tensiune DC-DC. Ieșirea modulului DC-DC furnizează 5V constanți utilizați la alimentarea plăcii Arduino și totodată, la alimentarea celorlalte elemente utilizate. În sourfigura 4.3 este ilustrat acumulatorul utilizat.

*Figura 4.3: Baterie Li-Po; Sursa: [29]*

Convertorul ridicător de tensiune (engl. „boost DC”) este un circuit ce oferă la ieșire o valoare mai mare a tensiunii față de tensiunea de la intrarea acestuia. În cadrul proiectului am utilizat un astfel de modul atât pentru alimentarea telecomenzii cât și pentru realizarea alimentării în cadrul parcării inteligente. Un astfel de convertor ridicător de tensiune conține următoarele elemente: un inductor, o diodă, un circuit comutator și un condensator. Structura de bază a unui astfel de circuit este prezentată în figura 4.4.

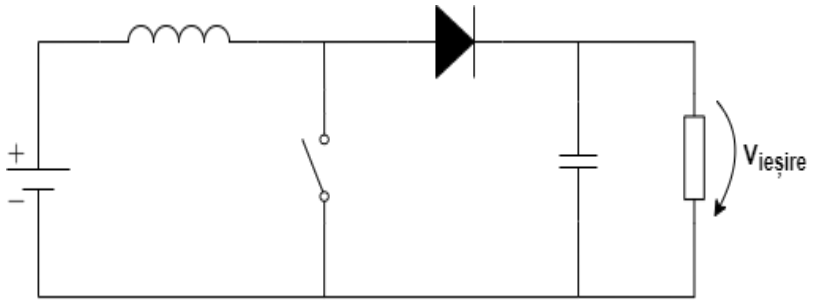



*Figura 4.4: Schema de bază a convertorului DC-DC; Sursa: [35]*

Principiul de funcționare al unui astfel de circuit este următorul: atunci când comutatorul este închis, curentul circulă prin bobină, determinând creșterea energiei înmagazinate la nivelul acesteia. În situația în care comutatorul este deschis, curentul din bobină circulă spre ieșire prin diodă. Astfel, tensiunea la ieșire va fi egală cu tensiunea de la intrare plus tensiunea generată de către bobină. Tensiunea de la ieșire va fi mai mare decât tensiunea de la intrare, de aici și denumirea de circuit ridicător de tensiune. În momentul când comutatorul este închis, dioda nu va mai conduce, iar tensiunea la ieșire este asigurată de către condensator. Pentru a se genera o tensiune constantă la ieșirea circuitului , timpul în care comutatorul este închis trebuie să fie mai mic decât timpul în care condensatorul este descărcat. [35]

# CAPITOLUL 5 Detalii de implementare hardware

## 5.1 Implementarea hardware a robotului mobil

Pentru realizarea hardware a robotului mobil am utilizat un șasiu prevăzut cu patru motoare de curent continuu, pe care am poziționat și conectat puntea de control pentru motoare, senzorii utilizați pentru deplasarea corectă a mașinii, placa de dezvoltare Arduino Nano și bateriile ce alimentează robotul mobil. În figura 5.1 este prezentată diagrama bloc a robotului mobil.

Partea de alimentare a robotului mobil se realizează de la două baterii Li-Ion. Modul de conectare al acestora este în serie, ansamblu ce generează o tensiune cuprinsă în gama 7,2 V și 8,2 V în funcție de procentul de încărcare al acumulatorilor folosiți. Ansamblul realizat cu cele două baterii înseriate se conectează la blocul de control printr-un comutator cu reținere.

Controlul robotului mobil se realizează cu ajutorul a patru motoare de curent continuu, conectate la driverul de motoare L298N. Cu ajutorul modulului L298N se pot controla doar două motoare de curent continuu, astfel, am conectat ambele motoare de pe partea dreaptă în paralel la o ieșire a punții, respectiv celelalte două motoare de pe partea stângă, în paralel la cealaltă ieșire a punții. Tensiunea de alimentare a punții și respectiv a celor patru motoare se realizează prin intermediul blocului de alimentare. Pe lângă alimentarea motoarelor, puntea de control asigură o ieșire constantă de 5 V, ce poate fi folosită extern pentru alimentarea altor componente electrice.

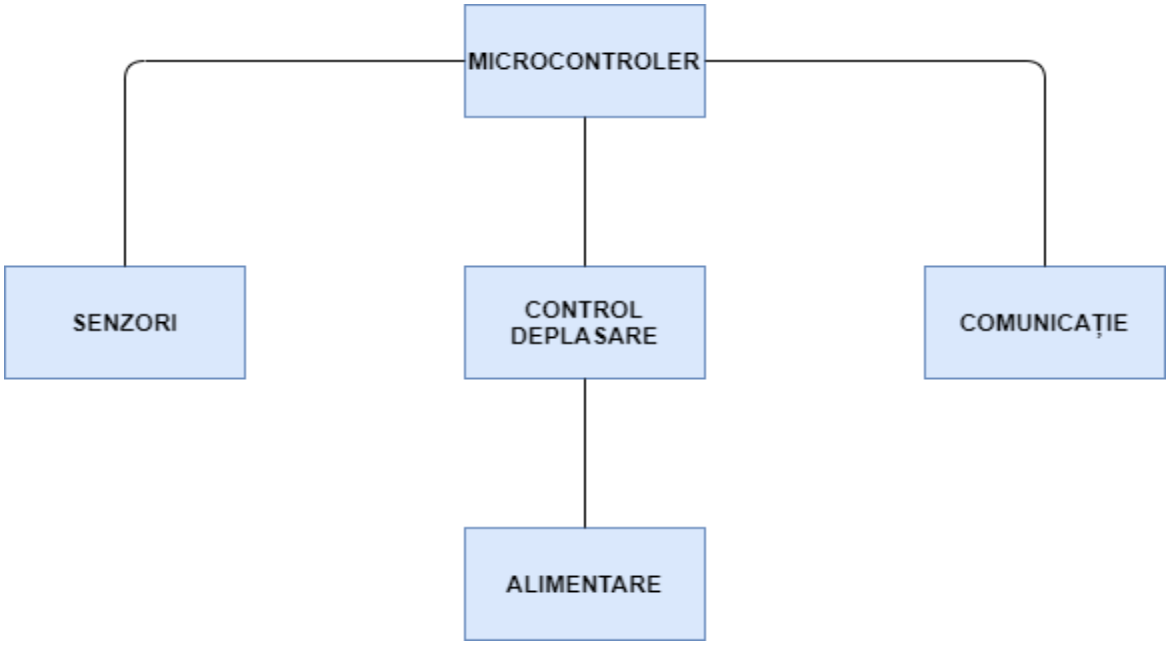



*Figura 5.1: Diagrama bloc a robotului mobil*

Blocul de senzori este reprezentat de către ansamblul senzorilor utilizați pentru colectarea datelor de la mediul exterior, date ce sunt necesare atât pentru orientarea robotului mobil cât și pentru utilizator. Astfel, senzorii folosiți sunt următorii: un senzor DHT11 ce are rolul de a colecta datele privind temperatura și umiditatea mediului exterior, senzorul ultrasonic HC-SR04 cu ajutorul căruia se poate determina distanța până la obstacol, doi senzori fotoelectrici cu infraroșu care împreună cu două roți pentru „encoder" ajută la determinarea distanței de deplasare a robotului mobil și senzorul QMC5883L, cu ajutorul căruia se determină poziția curentă a robotului mobil și se poate urmări realizarea unui viraj corect la 90 de grade a acestuia.

Blocul de comunicație este reprezentat de către modulul wireless(engl. wireless = rețea fără fir) nRF24L01, ce joacă rolul atât de receptor cât și de transmițător. Prin intermediul modulului se primesc date de la telecomandă sau de la parcare, date privind direcțiile de deplasare, respectiv disponibilitatea locurilor de parcare și totodată, se trimit informații de la mașină către telecomandă privind temperatura și umiditatea mediului exterior.

Procesarea datelor se realizează prin intermediul plăcii de dezvoltare Arduino Nano, ce are la bază microcontrolerul ATmega328p. Placa de dezvoltare Arduino Nano și celelalte componente utilizate sunt alimentate de la pinul de 5V al punții de control a motoarelor. În cazul componentelor ce necesită o alimentare de 3,3 V, acestea sunt alimentate la pinul de 3,3V disponibil pe placa de dezvoltare Arduino Nano.

Schema electrică a robotului mobil este prezentată în Anexa 1. Aceasta a fost realizată în mediul de proiectare Eagle. Deoarece este un mediu de proiectare ce oferă acces restrictiv la biblioteci și aflându-mă în situația în care nu am găsit simboluri pentru toate componentele utilizate, am realizat diferite simboluri pentru componentele utilizate, precum cele pentru motoare sau cele pentru senzorii fotoelectrici cu infraroșu.

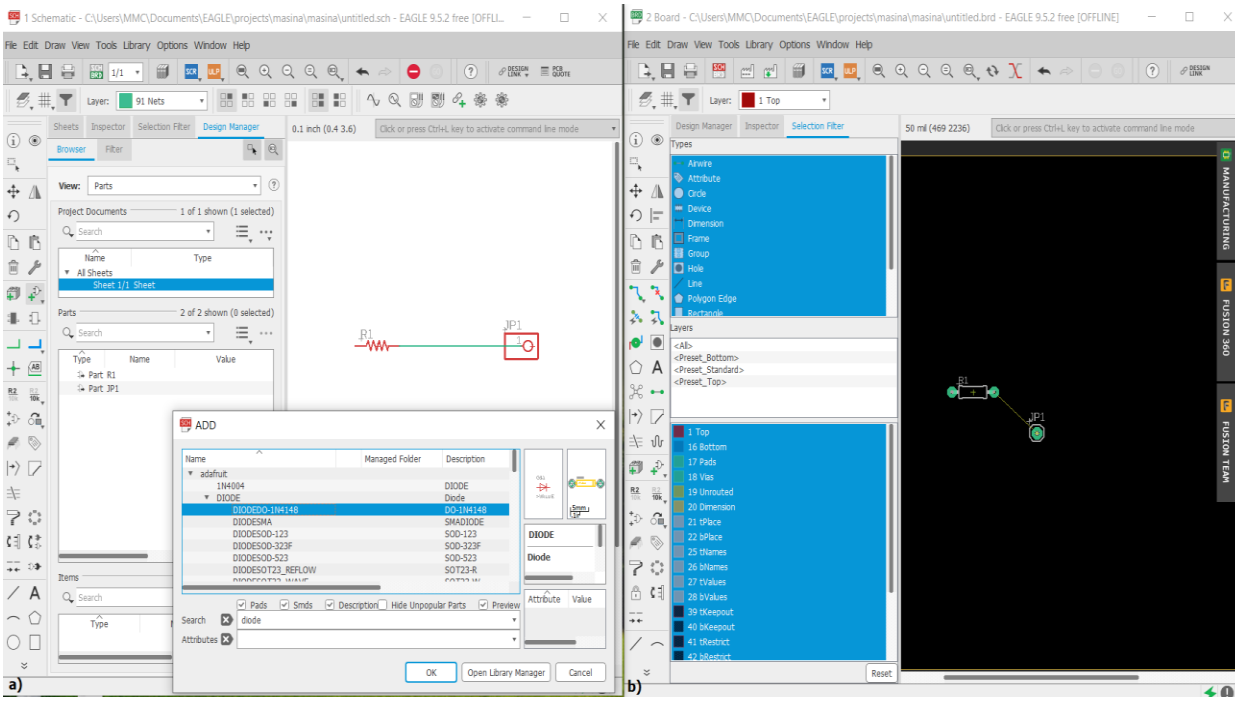



*Figura 5.2: Interfața programului Eagle*

În figura 5.2 a) este prezentată interfața editorului schematic din cadrul programului Eagle, unde au fost proiectate și editate schemele realizate în cadrul programului, iar în figura 5.2 b) este prezentată interfața pentru proiectarea și editarea layoutului, în cadrul căruia s-a proiectat circuitul cu cablaj imprimat realizat în cadrul dezvoltării hardware a robotului mobil.

În cadrul proiectului, la realizarea robotului mobil, am dorit să utilizez cât mai puține fire de legătură pentru a conecta toate elementele utilizate, deoarece la deplasări ale acestuia sau la eventuale impacte nedorite, pot foarte ușor apărea probleme de întreruperi ale conexiunilor. Pentru a evita această problemă, dar și pentru a realiza o așezare cât mai precisă și mai compactă a componentelor am realizat o placă de cablaj imprimat. Schema circuitului este prezentată în Anexa 2. În schema circuitului, în plus față de elementele prezente în circuitul propriu-zis, am adăugat șase pini de masă, șase pini de 5V și patru pini de 3,3V. Totodată, am adăugat doi pini ce oferă posibilitatea conectării a două leduri. Fiecare dintre cei doi pini sunt conectați prin intermediul unui rezistor a cărei rezistență are valoarea 220Ω la pinul de 5V.

Circuitul cu cablaj imprimat proiectat (prezentat în Anexa 3), respectă următoarele reguli de proiectare:

- Dimensiunile PCB sunt de 75 mm lungime și 65,25 mm lățime
- Componentele utilizate și pinii tată folosiți pentru conectarea din exterior a celorlalte elemente sunt așezate numai pe fața superioară (TOP) a plăcii cu cablaj imprimat.
- Grosimea stratului de cupru este de 0,035 mm , iar dimensiunea stratului izolant FR4 este de 1,5mm
- S-a păstrat față de marginea plăcii cu cablaj imprimat o distanță de 1,016mm
- Spațierea în toate cazurile este de 0,3mm
- Găurile de trecere au diametrul de 0,7mm
- Lățimea traseelor de semnal este de 0,4064 mm
- PCB-ul prezintă patru găuri de prindere cu diametrul de 4mm

În Anexa 3 este ilustrat layout-ul circuitului, în figura 5.3 este ilustrată fața superioară a plăcii cu cablaj imprimat (vedere TOP) , iar în figura 5.4 este ilustrată fața inferioară a plăcii cu cablaj imprimat (vedere BOTTOM).

În Anexa 6 este prezentată macheta robotului mobil dezvoltată în cadrul proiectului.

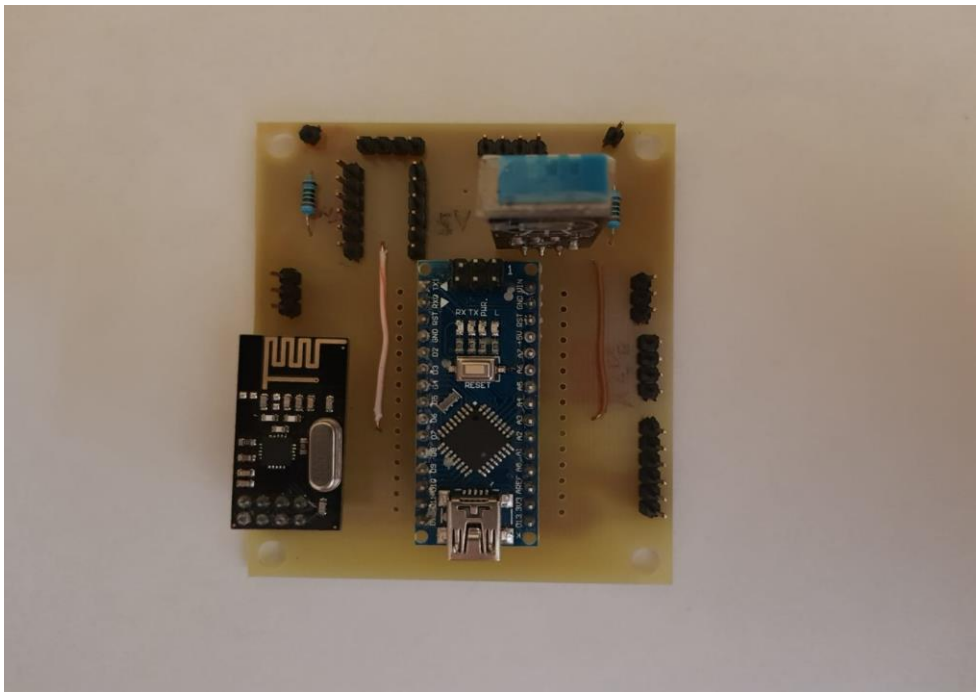


*Figura 5.3: Vedere TOP a plăcii cu cablaj imprimat*

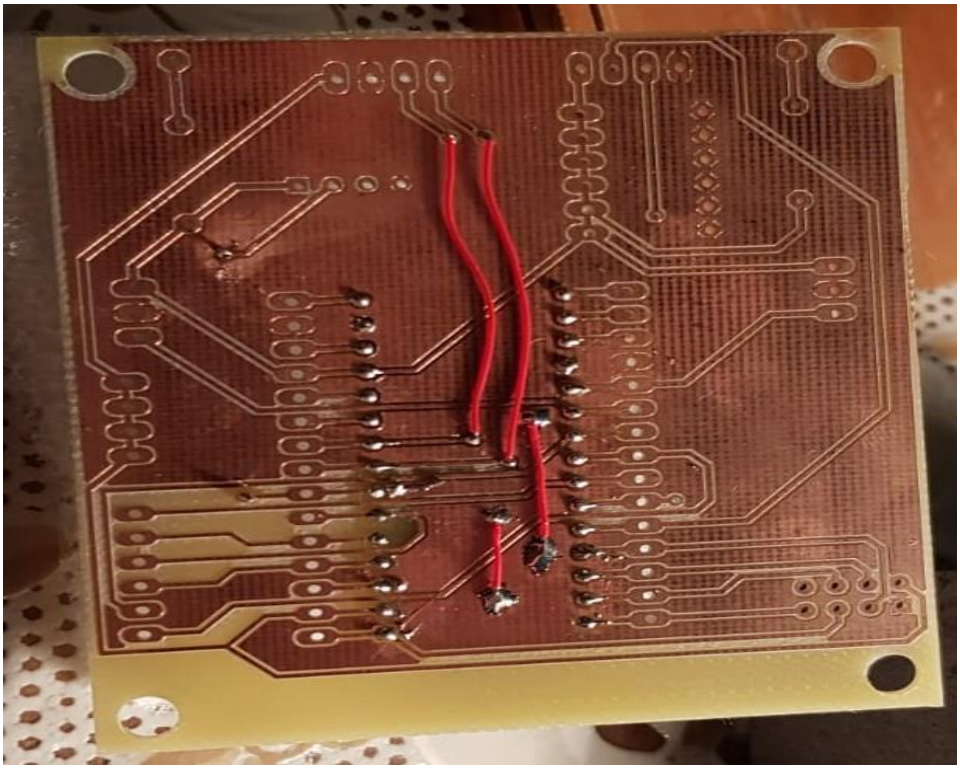


*Figura 5.4: Vedere BOTTOM a plăcii cu cablaj imprimat*

# 5.2 Implementarea hardware a parcării inteligente

Pentru construirea parcării am folosit două plăci de polistiren extrudat, cu ajutorul cărora am realizat modelul parcării și am integrat componentele utilizate. Firele de legătură au fost mascate în interiorul polistirenului, oferind astfel, un aspect special. Totodată, am utilizat diferite semne de marcaj și bandă pentru marcaj pentru a delimita zonele parcării. În ceea ce privește partea electrică a parcării, diagrama bloc a acesteia este prezentată în figura 5.5.

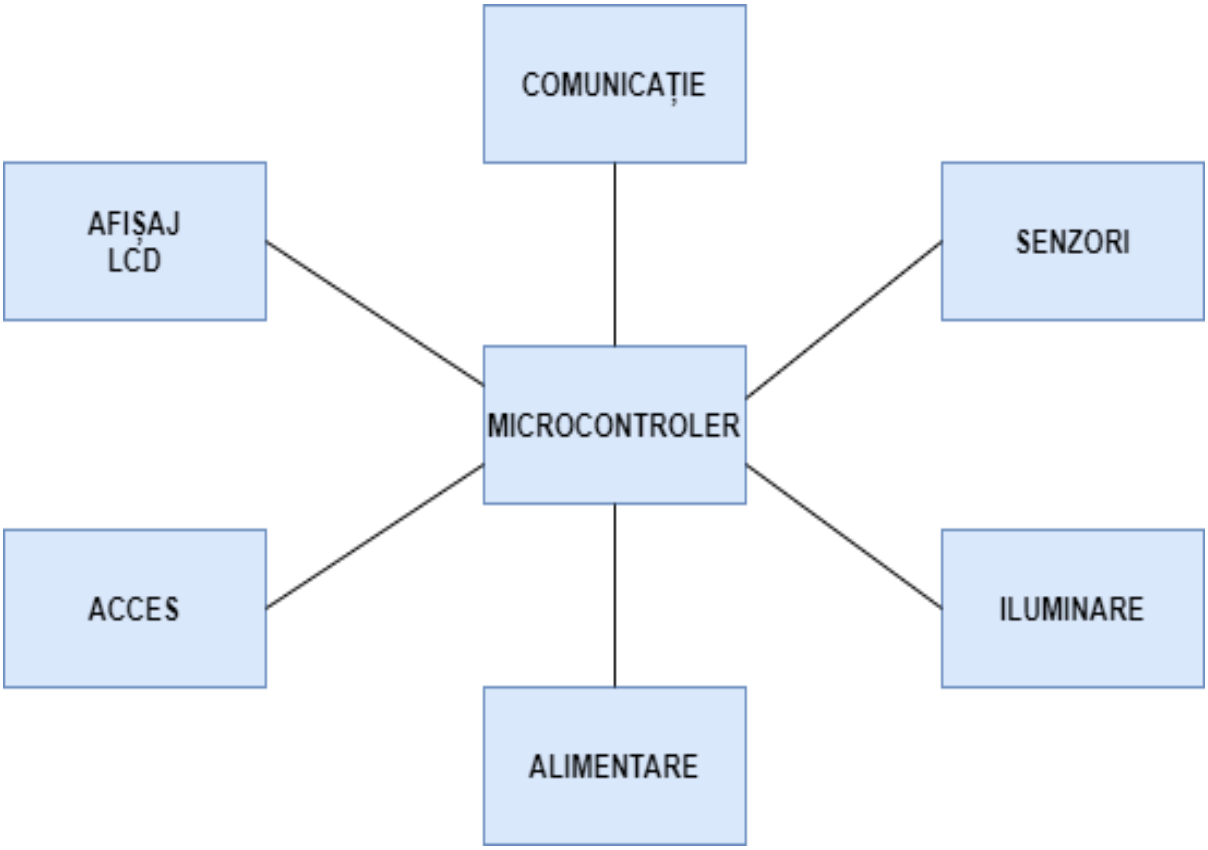



*Figura 5.5: Diagrama bloc a parcării inteligente*

Blocul de alimentare al parcării este constituit cu ajutorul unui acumulator Li-Ion conectat la un convertor DC-DC prin intermediul unui comutator cu reținere. Convertorul DC-DC oferă la ieșire o tensiune constantă de 5V, ce asigură alimentarea microcontrolerului și a celorlalte componente utilizate.

Blocul de iluminare este reprezentat de către diodele electroluminiscente de diferite culori ce au rolul de a marca accesul în parcare. La intrarea în parcare sunt dispuse două diode electroluminiscente de culoare roșie și verde, ce marchează permisiunea accesului sau lipsa

permisiunii acestuia în parcare, iar în interiorul parcării două diode electroluminiscente de culoare albastră și albă, ce au rol de iluminare. În serie cu fiecare diodă electroluminiscentă am legat un rezistor a cărui valoare este de 220Ω. Valoarea acestuia a fost aleasă în funcție de tensiunea de deschidere a diodei electroluminiscente și de curentul direct prin acestea.

Senzorii utilizați în cadrul parcării au rolul de a depista prezența sau absența unui obstacol. S-au utilizat patru senzori infraroșii de obstacole, amplasați în diferite puncte cheie ale parcării. Unul dintre senzori este amplasat la intrare, iar ceilalți trei sunt amplasați în fiecare loc de parcare. Firele de conexiune sunt mascate în materialul cu ajutorul căruia s-a construit parcarea, astfel, senzorii nu oferă un aspect nedorit parcării. Modalitatea de conexiune a senzorilor infraroșii de obstacole este ilustrată în Anexa 4.

Blocul de comunicație este reprezentat de către modulul wireless nRF24L01, care are rolul de a transmite către mașină informațiile privind disponibilitatea locurilor de parcare.

La intrarea în parcare este atașat un ecran LCD, pe care este afișată disponibilitatea locurilor de parcare. Astfel, dacă un loc de parcare este liber, pe ecranul LCD se va afișa „v”, iar dacă un loc este ocupat, pe ecranul LCD se va afișa „x”. Dacă cel puțin un loc de parcare este liber, dioda electroluminiscentă de culoare verde va fi polarizată direct, iar aceasta va marca permiterea accesului în parcare. Dacă toate cele trei locuri de parcare vor fi ocupate, dioda electroluminiscentă de culoare roșie se va aprinde, iar pe ecranul LCD-ului va fi afișat mesajul „PARCAREA ESTE OCUPATA!”. Conexiunea afișajului LCD este prezentată în schema din Anexa 4.

Partea de procesare a datelor se realizează cu ajutorul plăcii de dezvoltare Arduino Uno R3, ce are la bază microcontrolerul ATmega328p. Așa cum am menționat anterior, aceasta este alimentată de la ieșirea constantă de 5V a convertorului DC-DC menționat în cadrul blocului de alimentare.

Permiterea accesului în parcare se realizează prin acționarea unei bariere de către un servomotor. Alimentarea servomotorului se realizează de la un ansamblu de patru baterii de tip AA, ce furnizează o tensiune maxima de 6V. Acestea sunt legate la servomotor prin intermediul unui comutator cu reținere. Atunci când cel puțin un loc de parcare este liber și la intrarea în parcare se află o mașină, bariera va fi ridicată de către servomotor, permițând accesul în parcare. Dacă niciun loc nu este liber sau dacă nicio mașină nu se află la intrarea în parcare, bariera nu va fi ridicată și astfel, accesul în parcare va fi restricționat.

Schema electrică a parcării este ilustrată în Anexa 4, unde LED1 este de culoare albastră, LED2 este alb, LED3 este de culoare roșie, LED4 este de culoare verde, SENZOR_IR0 este senzorul de la intrarea în parcare, iar senzorii SENZOR_IR1, SENZOR_IR2, SENZOR_IR3 sunt senzorii amplasați în locurile de parcare corespunzătoare (loc1,loc2,loc3).

În Anexa 7 este prezentată parcarea realizată în cadrul proiectului.

# 5.3 Implementarea hardware a telecomenzii

Schema bloc a telecomenzii realizate în cadrul proiectului este prezentată în figura 5.6.

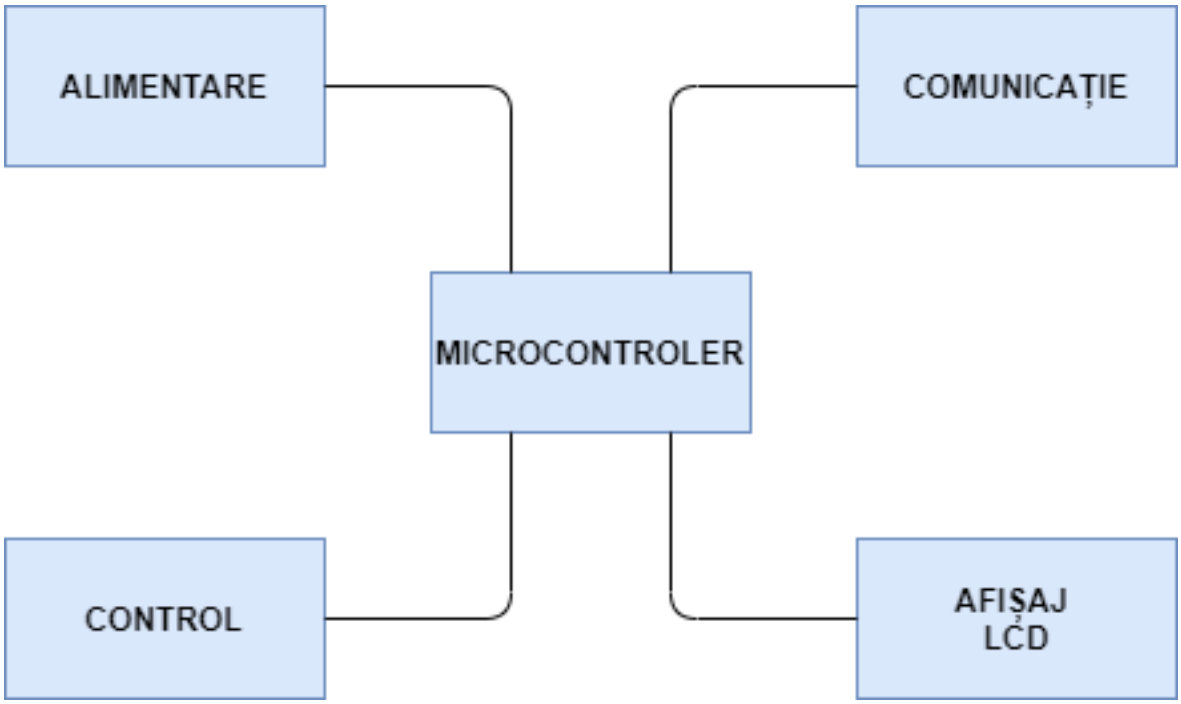



*Figura 5.6: Schema bloc a telecomenzii*

Blocul de alimentare constă într-un acumulator Li-Ion ce este conectată prin intermediul unui comutator la convertorul DC-DC, a cărui ieșire furnizează cei 5V constanți necesari alimentării microcontrolerului și a celorlalte elemente utilizate.

Partea de control, de interacțiune dintre utilizator și microcontroler se realizează prin intermediul unui modul joystick compatibil cu placa de dezvoltare utilizată. Controlul propriu-zis se realizează printr-un joystick cu două axe, ce reprezintă direcțiile de deplasare ale robotului mobil și prin alte trei butoane, unul pentru acționarea frânei, unul pentru setarea modului de receptor al telecomenzii, pentru a primi informații despre mediul înconjurător de la mașină și unul pentru a comuta între comunicația dintre mașină și telecomandă, în comunicația dintre mașină și parcare. Dacă utilizatorul dorește ca mașina să treacă în modul de parcare autonom, acesta va acționa butonul F, de pe modulul joystick, ceea ce va duce la realizarea comunicației dintre mașină și parcare. După acționarea butonului F, utilizatorul nu mai poate direcționa robotul mobil, manual, prin intermediul telecomenzii.

Comunicația se realizează, ca și în celelalte cazuri, cu ajutorul modulului wireless nRL24L01. Setat în modul de transmisie, prin intermediul acestuia sunt trimise informațiile preluate de la modulul joystick privind direcțiile de deplasare, către robotul mobil, iar dacă

modulul este setat în modul de recepție, va primi date de la mașină privind informații despre temperatura și umiditatea mediului înconjurător.

Datele sunt prelucrate cu ajutorul plăcii de dezvoltare Arduino UNO R3 ce are la bază microcontrolerul ATmega328p. Modulul joystick este compatibil cu placa de dezvoltare utilizată, ceea ce duce la un aspect compact al telecomenzii, deoarece modulul joystick se atașează direct în pinii mamă prezenți pe placa de dezvoltare Arduino UNO R3.

Afișajul LCD utilizat prezintă informațiile primite de la robotul mobil, informații despre temperatură și umiditate. Acesta este atașat de ansamblul realizat de către placa de dezvoltare și modulul joystick utilizate.

În Anexa 5 este prezentată schema electrică a telecomenzii, iar în Anexa 8 este prezentată telecomanda fizică realizată.

# CAPITOLUL 6 Detalii de implementare software

## 6.1 Implementarea software a robotului mobil

Implementarea software s-a realizat în Arduino IDE, mediu de dezvoltare compatibil cu limbajele de programare C și C++. În cadrul proiectului, au fost utilizate atât funcții prestabilite cât și funcții proprii ce au fost dezvoltate utilizând limbajul C. În mediul de dezvoltare Arduino IDE, programul este structurat în trei părți: zona globală, unde sunt incluse bibliotecile utilizate, variabilele folosite, funcțiile proprii create; funcția „setup", ce se execută o singură dată, folosită în general, pentru configurarea pinilor sau pentru configurarea modulelor folosite și funcția principală „loop", ce se execută la infinit, cât timp alimentarea microcontrolerului este pornită. De obicei, în funcția principală se apelează funcții, se prelucrează datele primite de la senzori, se configurează diferite elemente conectate la microcontroler, etc. În cadrul robotului mobil, codul sursă este prezentat în Anexa 9. În partea de început a codului sursă sunt declarate variabilele globale utilizate, sunt incluse bibliotecile utilizate [30], [31], [32], [33] și sunt dezvoltate funcții ce vor fi folosite în cadrul programului. În funcția „setup" se configurează pinii și modulele utilizate, iar în cadrul funcției principale „loop" este dezvoltată funcționalitatea mașinii. Diagrama bloc a funcționalității codului sursă este prezentată în figura 6.1.

Inițial, se verifică dacă robotul mobil comunică cu parcarea sau cu telecomanda. Acest lucru se face prin intermediul variabilei „schimba", a cărei valoare predefinită este 0. Astfel, pentru prima iterație a funcției „loop", se va executa ramura corespunzătoare blocului de comunicație realizat între mașină și telecomandă. Robotul mobil este configurat ca fiind receptor, iar telecomanda ca fiind transmițător. Astfel, se verifică dacă se recepționează date de la telecomandă. În cazul afirmativ, se prelucrează informațiile primite. Dacă utilizatorul a apăsat butonul F de pe telecomandă, valoarea constantei „schimba" va fi modificată, iar la următoarea iterație a funcției principale, „loop", se va executa ramura corespunzătoare comunicației dintre mașină și parcare. Ulterior, se verifică dacă utilizatorul a acționat butonul de frână de pe telecomandă. Dacă butonul a fost acționat, motoarele vor fi oprite, iar dacă butonul de frână nu a fost acționat, robotul mobil va executa mișcările de deplasare corespunzătoare datelor primite de la telecomandă (deplasarea înainte, deplasarea înapoi, deplasarea la stânga sau la dreapta). În urma acestei etape se va verifica dacă trebuie transmise date privind temperatura și umiditatea mediului exterior. Transmiterea datelor privind informațiile despre mediul exterior se face o dată la șase secunde, din momentul alimentării robotului mobil, respectiv de la ultima transmisie efectuată. Dacă condiția de transmisie este îndeplinită, se citesc datele, privind informații despre temperatura și umiditatea mediului ambiant, de la senzorul DHT11, acestea fiind apoi transmise către telecomandă. Ulterior acestei etape, procesul se va relua de la capăt.

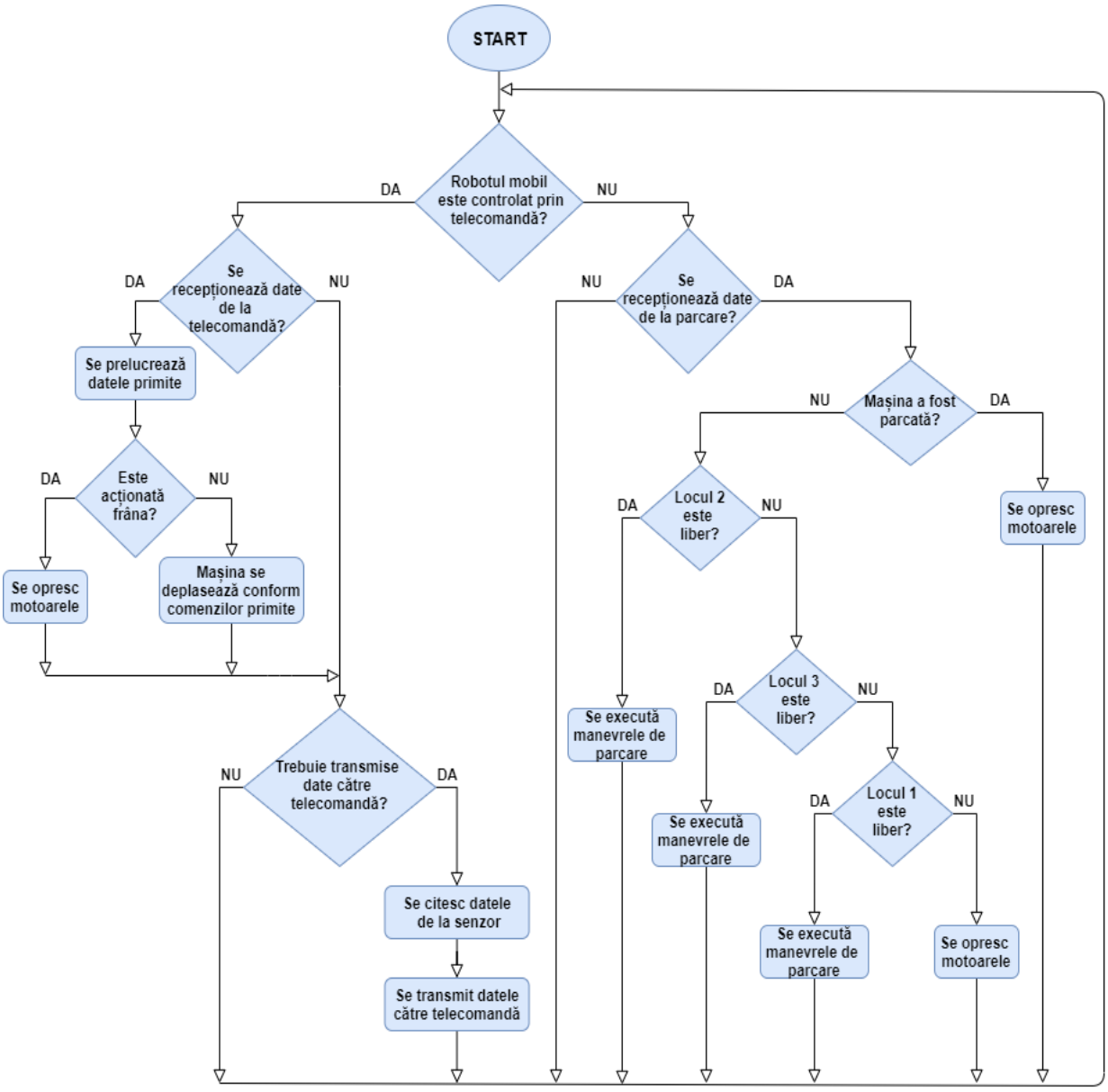



*Figura 6.1: Organigrama software a robotului mobil*

Dacă în urma procesării ultimelor date primite de la telecomandă, reiese că utilizatorul a acționat butonul pentru schimbarea modului de control al robotului mobil, din modul de control prin intermediul telecomenzii, în modul de control autonom, se va executa ramura corespunzătoare comunicației dintre mașină și telecomandă. Astfel, adresa de comunicație va fi schimbată și robotul mobil va recepționa date privind disponibilitatea locurilor de parcare. Dacă robotul nu primește informații de la parcare, procesul se va relua până când acesta va primi informațiile dorite. În momentul în care robotul mobil va primi informații de la parcare, acesta le va procesa, identificând locul disponibil în care trebuie să parcheze. Acesta va executa manevrele necesare locului respectiv, iar în urma încheierii procesului de parcare autonom, motoarele robotului vor fi oprite, iar valoarea variabilei „ok” va fi egală cu 1, ceea ce înseamnă că manevrele de parcare au fost executate și robotul mobil este parcat.

Pentru realizarea manevrelor de deplasare corespunzătoare fiecărui loc de parcare, în cadrul proiectului, am dezvoltat trei funcții: una pentru deplasarea înainte, una pentru deplasarea la dreapta și alta pentru deplasarea la stânga. În figura 6.2 este ilustrată descrierea funcției utilizate pentru deplasarea înainte.

```
void inainte (float distanta , int viteza) {

  digitalWrite(s_inainte , HIGH);
  digitalWrite(s_inapoi  , LOW);
  digitalWrite(d_inainte , HIGH);
  digitalWrite(d_inapoi  , LOW);

  numarator_ISR1 = 0;
  numarator_ISR2 = 0;

  float numar_de_rotatii = (distanta *10) / circumferinta ;
  unsigned long pasi = numar_de_rotatii * tranzitii_pe_rotatie;

  while ( (numarator_ISR1 < pasi) && ( numarator_ISR2 < pasi) ){
   if(numarator_ISR1<pasi){
                            analogWrite( en_stanga , viteza) ;
                           } else {
                                   analogWrite( en_stanga, 0);
                                  }
   if(numarator_ISR2 < pasi ){
                            analogWrite( en_dreapta, viteza);
                           }else {
                                   analogWrite( en_dreapta, 0);
                                 }
  }
   oprire_motoare();
   numarator_ISR1 = 0;
   numarator_ISR2 = 0;
}
```

*Figura 6.2: Funcția utilizată pentru deplasarea pe direcția înainte*

Funcția pentru deplasarea robotului mobil pe direcția înainte primește la intrare doi parametrii. Primul parametru este un număr real, ce reprezintă distanța exprimată în centimetri pe care se dorește deplasarea robotului, iar al doilea parametru reprezintă viteza de deplasare al acestuia. Al doilea parametru poate fi un număr întreg cuprins între 0 și 255; valoarea 0 corespunde tensiunii nule aplicată la bornele motoarelor, iar valoarea 255 corespunde tensiunii maxime, ceea ce va duce la mișcarea de rotație la viteză maximă a motoarelor. Inițial, în cadrul funcției sunt setate direcțiile de rotație ale motoarelor, prin setarea polarității celor două borne ale motoarelor. Ulterior, este calculat numărul de rotații ale roților necesar deplasării pe distanța respectivă, ca fiind distanța exprimată în milimetrii împărțită la circumferința roții. Circumferința roții este calculată anterior, în cadrul programului, ca fiind diametrul roții înmulțit cu valoarea numărului PI. După determinarea numărului de rotații ale roților, se determină numărul de pași, ca fiind produsul între numărul de tranziții pe rotație și numărul de rotații determinat anterior. Numărul de tranziții pe rotație este 40, deoarece există 40 de tranziții posibile identificate de către senzorul fotoelectric infraroșu atașat roții „encoder” utilizată. Roata „encoder” atașată pe axul motorului, are douăzeci de spații ce permit trecerea radiației infraroșii de la senzor și douăzeci de spații ce blochează trecerea acesteia. Variabila „pași”, reprezintă deci, numărul total de tranziții corespunzătoare deplasării robotului pe distanța respectivă.

Variabilele „numarator_ISR1” și „numarator_ISR2” reprezintă numărătoare ce se vor incrementa în cadrul unei alte funcții dezvoltate anterior, la fiecare tranziție depistată de către senzorul fotoelectric infraroșu. În cadrul funcției „while” motoarele vor fi activate, până când se va atinge numărul de tranziții corespunzătoare distanței respective. Ulterior, funcția se încheie prin apelarea funcției „oprire_motoare”, descrisă în cadrul Anexei 9 și prin resetarea valorilor variabilelor „numarator_ISR1” și „numarator_ISR2”.

```
void dreapta (int viteza, int grade) {

int PozInit,PozCurenta,target,diferenta;
int FlagOprire = 0;
int indice_LD = -1, indice_LS = -1;
int LS[5]= {0, 0, 0, 0, 0}; // limita stanga
int LD[5]= {0, 0, 0, 0, 0}; // limita dreapta
  digitalWrite(s_inainte , HIGH);
  digitalWrite(s_inapoi  , LOW);
  digitalWrite(d_inainte , LOW);
  digitalWrite(d_inapoi  , HIGH);
  numarator_ISR1 = 0;
  numarator_ISR2 = 0;
  int pasi = 10;
  PozInit = compass.readHeading();
  target = PozInit + grade;
  if(target > 360) {
    target = target - 360;
  }
for(int i=0;i<5;i++){
  LS[i]= target+i-5;
  if(LS[i] <= 0) {
    LS[i] = LS[i] + 360;
  }
}
for(int j=0;j<5;j++){
  LD[j]= target+j+1;
  if(LD[j] > 360) {
    LD[j] = LD[j] - 360;
  }
}
 do{
   numarator_ISR1 = 0;
   numarator_ISR2 = 0;

  while ( (numarator_ISR1 < pasi) && ( numarator_ISR2 < pasi) )
{
  if(numarator_ISR1<pasi){
      analogWrite( en_stanga , viteza) ;
                          }
                           else {
                                analogWrite( en_stanga, 0);
                                }
 if(numarator_ISR2 < pasi ){
      analogWrite( en_dreapta, viteza);
                           }
                            else {
                                   analogWrite( en_dreapta, 0);
                                 }
}
   numarator_ISR1 = 0;
   numarator_ISR2 = 0;
    PozCurenta = compass.readHeading();
  for(int k=0; k<5;k++){
if( (PozCurenta == LS[k]) || (PozCurenta == LD[k])
{
     if(PozCurenta == LS[k])
               {
                 indice_LS = k;
               }
                else {
                     indice_LD = k;
                      }
     FlagOprire = 1;
     break;
   }

  }
  }while(FlagOprire == 0);

 oprire_motoare();
 delay(2000) ;

// ~~~~~~~~~~~~~~~~~~AJUSTARE POZIȚIE ~~~~~~~~~~

if( !(indice_LS == -1) ) {

  diferenta = 5 - indice_LS;

  digitalWrite(s_inainte , HIGH);
  digitalWrite(s_inapoi  , LOW);
  digitalWrite(d_inainte , LOW);
  digitalWrite(d_inapoi  , HIGH);

  analogWrite(en_stanga,255);
  analogWrite(en_dreapta,255);
  delay(diferenta * 15);
  oprire_motoare();

} else if( !(indice_LD == -1) ) {

  diferenta = indice_LD + 1 ;

  digitalWrite(s_inainte , LOW);
  digitalWrite(s_inapoi  , HIGH);
  digitalWrite(d_inainte , HIGH);
  digitalWrite(d_inapoi  , LOW);

  analogWrite(en_stanga,255);
  analogWrite(en_dreapta,255);
  delay(diferenta * 15);
  oprire_motoare();
}
}
```

*Figura 6.3: Funcția utilizată pentru virajul la dreapta*

În figura 6.3 este ilustrată funcția pentru realizarea virajului la dreapta. Aceasta primește ca și parametrii de intrare două variabile întregi, „viteza” și „grade”, ce reprezintă variabila

proporțională cu viteza de rotație a motoarelor, respectiv parametrul „grade” ce reprezintă unghiul de rotație al robotului mobil. La început sunt declarate și inițiate variabilele utilizate în cadrul funcției și sunt configurate direcțiile de rotație ale motoarelor corespunzătoare deplasării la dreapta. Ulterior, este memorată în variabila „PozInit”, poziția inițială a robotului mobil și este calculată valoarea variabilei „target”, care reprezintă poziția în care se dorește să se deplaseze robotul mobil. Dacă poziția „target” depășește valoarea 360º, se va scădea valoarea 360º din variabila „target”, deoarece s-a realizat o depășire a valorii maxime. Mașina se poate găsi într-o poziție corespunzătoare valorilor cuprinse între 0º și 360º. Din cauza surselor de eroare, în urma virajului, robotul nu se va găsi fix în poziția dorită, astfel se va alege o marjă de eroare de ± 5º. Vectorii „LS” și „LD” sunt inițializați cu pozițiile corespunzătoare acestor valori acceptate. Funcția „do while” se va executa cât timp poziția finală obținută în urma virajului va fi diferită de pozițiile acceptate, în care se dorește să se ajungă. Ulterior, se va verifica dacă robotul mobil a virat prea mult sau prea puțin față de poziția „target”. Se va memora în variabila „indice_LS” sau în variabila „indice_LD” indicele corespunzător poziției în care se află robotul mobil și se va face o ajustare a poziției acestuia, proporțională cu valoarea indicelui. În final, motoarele vor fi oprite prin apelarea funcției „oprire_motoare()”.

Funcția creată pentru executarea manevrei de viraj la stânga este dezvoltată asemănător cu funcția prezentată anterior pentru realizarea virajului la dreapta. Aceasta nu va mai fi dezvoltată în cadrul acestui subcapitol, putând fi vizualizată în Anexa 9.

## 6.2 Implementarea software a parcării inteligente

Implementarea software a parcării inteligente s-a realizat, ca și în cadrul implementării soft a robotului mobil, în mediul de dezvoltare Arduino IDE. În partea de început a programului sunt declarate și inițiate variabilele folosite în cadrul programului și sunt incluse bibliotecile [31], [34], utilizate pentru interacțiunea cu componentele din cadrul parcării. În funcția „setup” sunt configurați pinii utilizați, ecranul LCD și modulul de comunicație wireless. Structura părții de cod descrisă în cadrul funcției principale „loop” este descrisă în organigrama prezentată în figura 6.4.

Inițial, se preiau informațiile privind disponibilitatea locurilor de parcare de la senzorii amplasați la nivelul acesteia. În cazul în care toate locurile de parcare sunt ocupate, accesul în parcare va fi interzis. Bariera atașată la intrarea în parcare va rămâne coborâtă, iar dioda electroluminiscentă de culoare roșie amplasat la intrarea în parcare va lumina, marcând interzicerea accesului în parcare. Ulterior acestei etape, se va afișa pe ecranul LCD amplasat la intrarea în parcare, mesajul „PARCAREA ESTE OCUPATĂ!” și se vor transmite către robotul mobil informațiile privind disponibilitatea locurilor de parcare. În cazul în care cel puțin un loc de parcare este liber, se va verifica dacă la intrarea în parcare se află o mașină. Dacă la intrarea în parcare se află o mașină, se va permite accesul în parcare: bariera va fi ridicată, dioda electroluminiscentă de culoare roșie va fi stinsă, iar dioda electroluminiscentă de culoare verde se va aprinde marcând permisiunea accesului în parcare. În caz contrar, dacă la intrarea în parcare nu se află o mașină, cu toate că cel puțin un loc de parcare este liber, bariera va rămâne coborâtă, interzicând accesul în parcare. În urma acestei etape, pe ecranul LCD vor fi afișate informațiile privind disponibilitatea locurilor de parcare, astfel, pentru a marca un loc de parcare disponibil, pe ecranul LCD se va afișa litera „v” în dreptul locului de parcare respectiv, iar pentru a marca

indisponibilitatea unui anumit loc, în dreptul acestuia se va afișa litera „x”. Procesul continuă cu transmiterea informațiilor către robotul mobil, după care acesta se reia de la început, conform organigramei din figura 6.4.

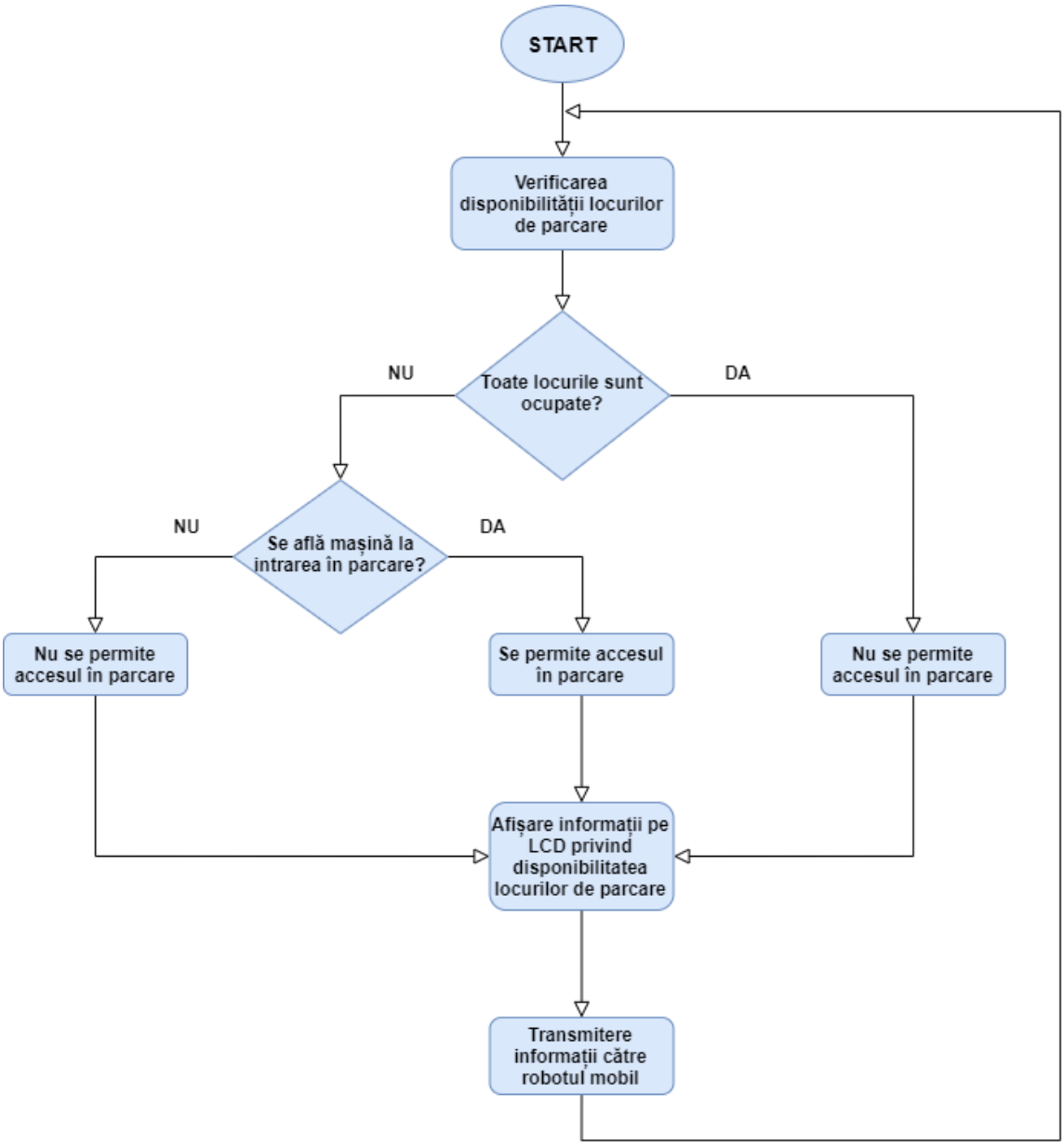



*Figura 6.4: Organigrama software a parcării inteligente*

Codul sursă, dezvoltat în cadrul proiectului, privind implementarea software a parcării inteligente este atașat în Anexa 10.

# 6.3 Implementarea software a telecomenzii

Codul sursă, privind implementarea software a telecomenzii, dezvoltat în cadrul proiectului este atașat în Anexa 11. Ca și în cazul celorlalte două coduri sursă dezvoltate pentru controlul mașinii, respectiv pentru controlul parcării, și în cazul telecomenzii, în partea de început a codului sursă sunt incluse bibliotecile [31], [34], utilizate în cadrul proiectului și sunt definite variabilele utilizate pe parcurs. În cadrul funcției „setup” se configurează pinii utilizați, se configurează ecranul LCD și totodată, se configurează modulul de comunicație wireless. Configurarea pinilor constă în definirea tipului acestora, pini de tip ieșire sau intrare, iar configurarea modulului de comunicație wireless constă în setarea canalului de comunicație, setarea adresei de comunicație, setarea puterii de transmisie și setarea ratei de transmisie. Organigrama software a telecomenzii este ilustrată în figura 6.5.

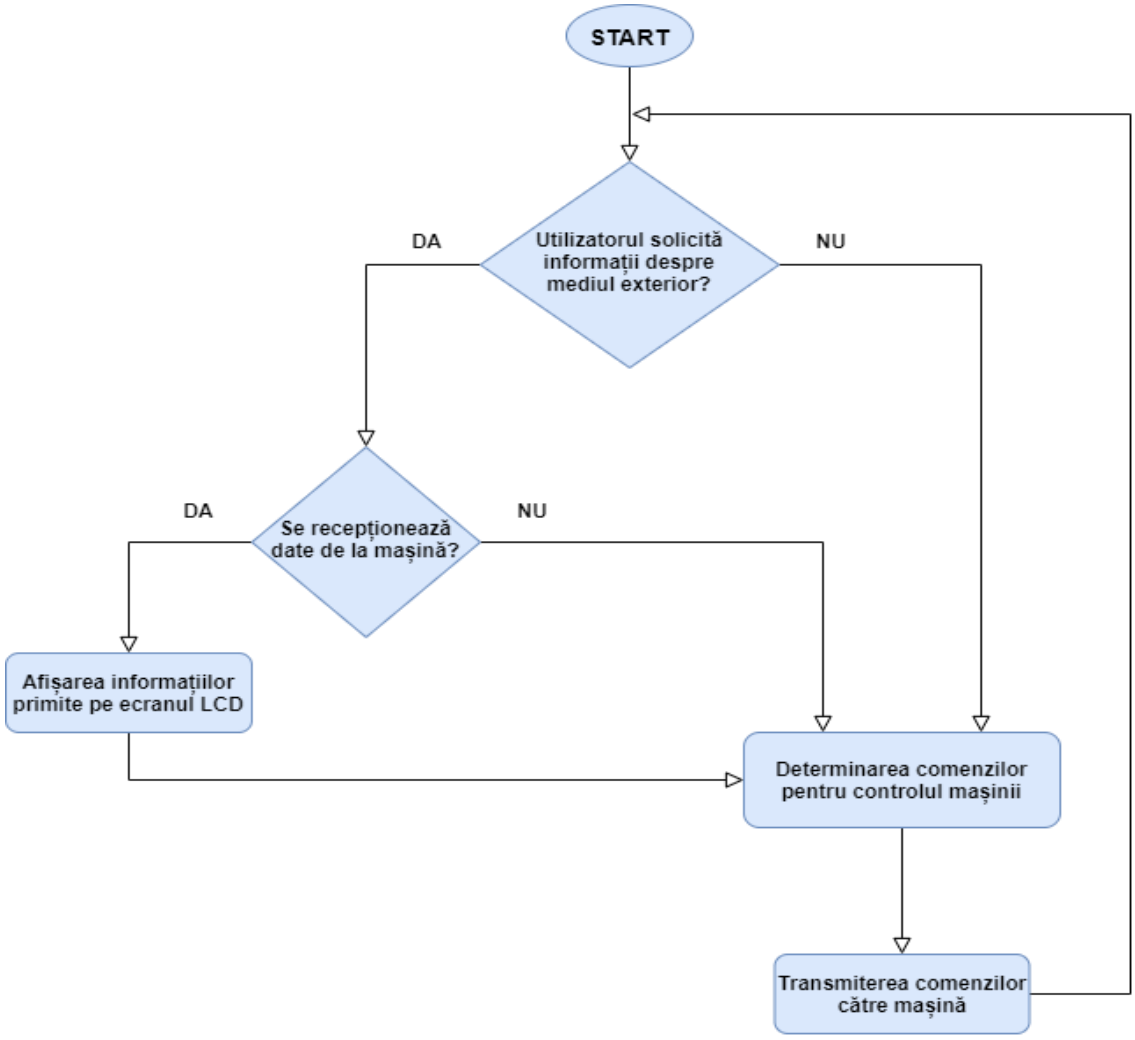



*Figura 6.5: Organigrama software a telecomenzii*

Dacă se dorește recepționarea informațiilor privind temperatura și umiditatea mediului exterior, date primite de la robotul mobil, utilizatorul va acționa butonul F de pe telecomandă. În momentul în care butonul este apăsat, telecomanda este configurată ca și receptor și se așteaptă primirea informațiilor de la mașină. Deoarece robotul mobil transmise o dată la șase secunde informațiile către telecomandă, pentru a fi sigur că va primi aceste informații, utilizatorul trebuie să țină apăsat butonul F timp de maxim șase secunde. În urma primirii informațiilor, acestea se prelucrează urmând să fie afișate pe ecranul LCD atașat telecomenzii. În urma terminării procesului de afișare a datelor pe ecranul LCD, în cazul în care nu au fost recepționate datele, ori în cazul în care nu este acționat butonul F de pe telecomandă, programul continuă cu determinarea comenzilor privind controlul deplasării robotului mobil. Acest lucru se face prin citirea stării butoanelor sau a valorilor potențiometrelor atașate joystick-ului din cadrul modulului utilizat. Pentru deplasarea pe direcția y, corespunzătoare direcției de deplasare înainte și înapoi, dacă valoarea citită de la potențiometru este mai mică sau egală decât 450 direcția de deplasare va fi înapoi, dacă valoarea este mai mare sau egala cu 550, direcția de deplasare va fi înainte, iar dacă valoarea este cuprinsă între cele două valori, înseamnă că modulul joystick nu a fost acționat de către utilizator pe direcția respectivă. Un raționament asemănător se utilizează și în cadrul determinării deplasării pe direcția x, reprezentând direcția de deplasare stânga, respectiv dreapta. În urma determinării informațiilor privind controlul robotului mobil, acestea vor fi transmise prin intermediul modulului de comunicație wireless către mașină. În urma transmiterii, procesul explicat anterior se va relua de la început, așa cum este evidențiat și în cadrul organigramei din figura 6.5.

# Concluzii

## Concluzii generale

În cadrul proiectului s-a realizat un robot mobil care poate parca autonom într-o parcare inteligentă. În cadrul proiectului a fost construit atât robotul mobil, cât și parcarea inteligentă. Totodată, a fost realizată și o telecomandă prin intermediul căreia utilizatorul poate controla manual robotul mobil. Pentru o integrare cât mai compactă a elementelor utilizate în cadrul robotului mobil s-a realizat și o placă cu cablaj imprimat. Pentru a se executa o deplasare cât mai precisă a robotului mobil, în cadrul proiectului, s-au folosit date primite de la mai mulți senzori, precum: senzor ultrasonic de distanță, senzor fotoelectric infraroșu, magnetometru. Pentru a evidenția eroarea la deplasarea pe direcția înainte, s-au efectuat mai multe măsurători pentru a vedea distanța reală parcursă de robotul mobil atunci când se dorește deplasarea acestuia pe o distanță de 20 cm. Datele obținute în urma măsurătorilor sunt prezentate în figura 7.1.

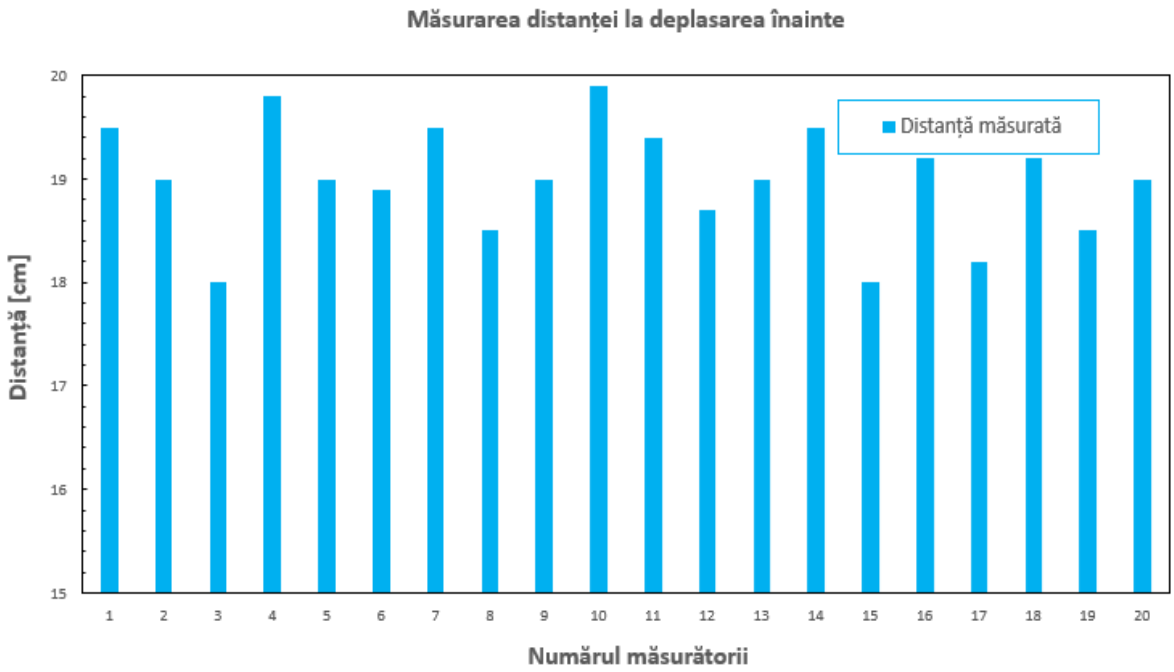



*Figura 7.1: Distanța reală parcursă de robotul mobil*

Eroarea medie obținută la deplasarea robotului pe direcția înainte utilizând funcția creată în cadrul proiectului este de aproximativ 5,4%. Consider că este o eroare acceptabilă deoarece, sursele de eroare ce duc la obținerea acesteia se pot datora, atât calității componentelor utilizate,

erorilor de calcul apărute în cadrul implementării funcției de deplasare pe direcția înainte, dar și erorii introduse de factorul uman, în cadrul realizării măsurătorilor.

Consider că, în urma realizării proiectului am dobândit cunoștințe teoretice privind modulele și componentele electronice utilizate, am căpătat experiență privind partea de proiectare și editate a unui circuit cu cablaj imprimat, am dobândit cunoștințe de programare, dar și experiență practică în procesul de creare a celor trei module: robotul mobil, parcarea inteligentă și telecomanda.

# Contribuții personale

Contribuțiile personale în cadrul dezvoltării acestui proiect sunt următoarele:

- Am creat robotul mobil ce se poate deplasa autonom, sau care poate fi controlat manual de către utilizator prin intermediul telecomenzii
- Am creat o parcare inteligentă capabilă identifice detalii despre disponibilitatea locurilor de parcare și să transmită aceste informații către robotul mobil
- Am creat o telecomandă prin intermediul căreia utilizatorul poate controla robotul mobil și pe care poate vedea informațiile privind mediul exterior, informații colectate de către mașină
- Am proiectat o placă cu cablaj imprimat pentru integrarea elementelor utilizate în cadrul dezvoltării robotului mobil
- Am realizat programarea robotului mobil pentru a executa manevrele de deplasare necesare parcării sau pentru executarea comenzilor primite de la telecomandă
- Am realizat programarea parcării inteligente pentru a colecta date privind disponibilitatea locurilor de parcare și pentru a transmite aceste informații către robotul mobil sau de a le prezenta pe ecranul LCD amplasat la intrarea în parcare
- Am realizat programarea telecomenzii pentru a transpune comenzile primite de la utilizator în manevre de deplasare executate de către robotul mobil
- Prin dezvoltarea celor trei module am realizat un sistem compact ce poate fi implementat și la o scară mai largă, oferind utilizatorilor posibilitatea de a avea un autovehicul capabil să parcheze autonom.

# Dezvoltări ulterioare

Ca și dezvoltări ulterioare proiectul poate fi îmbunătățit prin folosirea unor senzori ce oferă o precizie mai bună privind măsurătorile efectuate. O altă direcție de dezvoltare a proiectului o reprezintă adăugarea de noi senzori atât în cadrul parcării, cât și în cadrul robotului mobil. Spre exemplu, dacă parcarea ar fi implementată într-o locație închisă se poate crea un sistem de

ventilație, cu ventilator și senzori de temperatură și umiditate, sau senzori de gaze. Robotului mobil i se pot adăuga următoarele elemente: sistem de iluminare, sistem de acționare a unui claxon, sistem de semnalizare, sistem capabil să identifice autonom locurile de parcare disponibile sau poate fi echipat cu mai mulți senzori pentru a crește precizia manevrelor de deplasare.

Proiectul poate fi îmbunătățit și prin dezvoltarea funcțiilor deja create sau prin adăugarea de noi funcții privind controlul robotului mobil, al parcării sau al telecomenzii. Spre exemplu, în cadrul funcțiilor realizate pentru deplasarea robotului mobil se pot adăuga secvențe de cod pentru verificarea corectitudinii parametrilor introduși sau se poate modifica marja de eroare acceptată pentru realizarea manevrelor de viraj.

Proiectul se poate dezvolta prin adăugarea unei noi funcții ce realizează manevrele necesare ieșirii din parcare a robotului mobil. Totodată, se poate dezvolta o rețea de roboti mobili care să comunice între ei, pentru a realiza manevrele de parcare independent, fără a se intercala în procesul de execuție al manevrelor de parcare.

# Bibliografie


[1] *Automated Vehicles for Safety, https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety , accesat la data: 16.01.2020*

[2] *https://www.optimusdigital.ro/ro/motoare-altele/139-motor-cu-reductor-si-roata.html?search_query=motor+cu+reductie&results=3 ,accesat la data: 25.01.2020*

[3] *DC Motor Explained, https://www.youtube.com/watch?v=GQatiB-JHdI&t=635s , accesat la data: 16.04.2020*

[4] *Dr. Ing. Vlad-Cristian Georgescu, Senzori și circuite de condiționare a semnalelor, Curs 4 – Actuatoare, pagina 13*

[5] *Chen, L., Zhang, J., & Wang, Y. (2018, May). Wireless Car Control System Based on ARDUINO UNO R3. In 2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC) (pp. 1783-1787). IEEE.*

[6] *https://www.optimusdigital.ro/ro/senzori-senzori-optici/599-senzor-fotoelectric-in-miniatura-in-forma-de-u.html?search_query=senzor+fotoelectric+&results=10 , accesat la data: 18.02.2020*

[7] *https://www.electrodragon.com/w/images/6/60/ITR9608.pdf , accesat la data: 18.02.2020*

[8] *https://wiki.dcae.pub.ro/images/8/83/Ultrasonic_Transducer.pdf , accesat la data: 01.05.2020*

[9] *https://www.handsontec.com/dataspecs/HC-SR04-Ultrasonic.pdf , accesat la data: 01.05.2020*

[10] *http://www.cetti.ro/v2/curs_ccp/p2_13.pdf , accesat la data: 02.05.2020*

[11] *https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/ , accesat la data 03.05.2020*

[12] *https://www.optimusdigital.ro/ro/senzori-senzori-de-temperatura/584-senzor-de-temperatura-dht11.html?search_query=dht11&results=14 , accesat la data 04.05.2020*

[13] *https://www.youtube.com/watch?v=g-54itFQ-i8&t=128s , accesat la data 04.05.2020*

[14] *http://www.cetti.ro/v2/curs_ccp/materiale_ccp/TERMISTOARE/Termistoare_Varistoare_2013_revG.pdf , accesat la data 04.05.2020*

[15] *https://www.electronicoscaldas.com/datasheet/DHT11_Aosong.pdf , accesat la data 04.05.2020*

[16] https://www.circuitbasics.com/basics-of-the-spi-communication-protocol/ , accesat la data: 03.06.2020

[17] https://datasheet.octopart.com/NRF24L01-Nordic-Semiconductor-datasheet-10541936.pdf , accesat la data 05.05.2020

[18] https://howtomechatronics.com/tutorials/arduino/arduino-wireless-communication-nrf24l01-tutorial/ , accesat la data 05.05.2020

[19] https://www.circuitbasics.com/basics-uart-communication/ , accesat la data 07.05.2020

[20] https://mail.uaic.ro/~ftufescu/Circuite%20de%20conversie%20AD.pdf , accesat la data 07.05.2020

[21] https://ettron.com/what-is-a-servo-motor-how-a-servo-motor-works-control/, accesat la data:01.06.2020

[22] https://www.optimusdigital.ro/ro/senzori-senzori-optici/4514-senzor-infrarosu-de-obstacole.html?search_query=MODUL+SENZOR+OBSTACOL&results=18 , accesat la data: 01.06.2020

[23] https://www.tme.eu/Document/51467c482a9b32b37fc96070c60e59ba/l-53f3c.pdf , accesat la data: 01.06.2020

[24] Lucrarea numărul 4 de laborator, Simularea comportamentului fotodiodei într-un circuit optoelectronic, disciplina Senzori și traductori fotonici

[25] https://www.tme.eu/Document/570789529f5d40589506cb0d271e526a/BPV10NF.pdf , accesat la data: 01.06.2020

[26] https://www.electronicshub.org/ir-sensor/ , accesat la data 02.06.2020

[27] https://store.arduino.cc/arduino-uno-rev3 , accesat la data de 03.06.2020

[28] https://www.optimusdigital.ro/ro/optoelectronice-lcd-uri/2894-lcd-cu-interfata-i2c-si-backlight-albastru.html?search_query=lcd&results=204 , accesat la data 05.06.2020

[29] https://www.tme.eu/en/details/accu-lp803450_cl/rechargeable-batteries/cellevia-batteries/lp803450/ , accesat la data 06.06.2020

[30] https://github.com/dthain/QMC5883L , accesat la data 07.06.2020

[31] https://github.com/nRF24/RF24 , accesat la data 09.06.2020

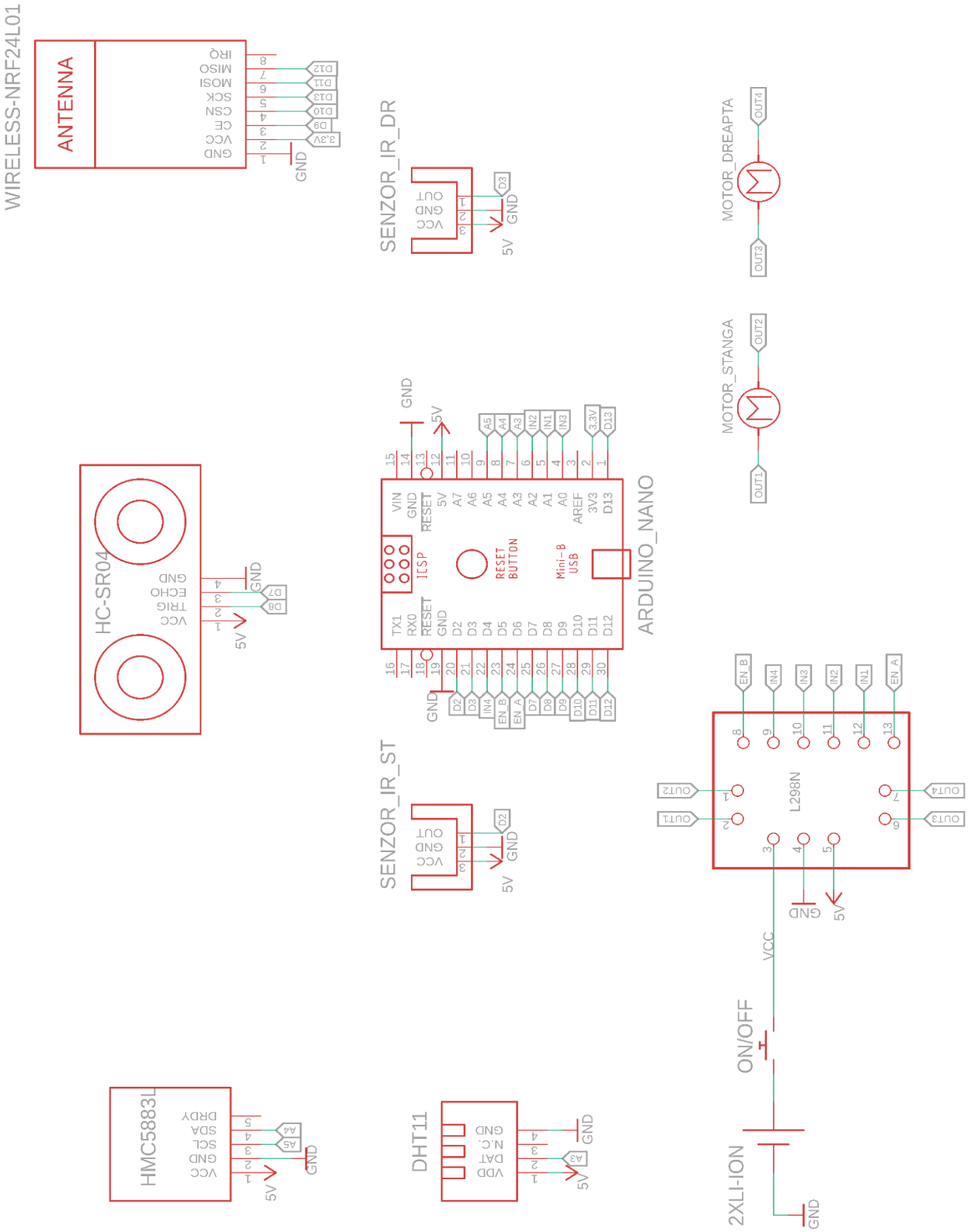
[32] https://github.com/winlinvip/SimpleDHT , accesat la data 09.06.2020

[33] https://github.com/mrRobot62/Arduino-ultrasonic-SR04-library , accesat la data 11.06.2020

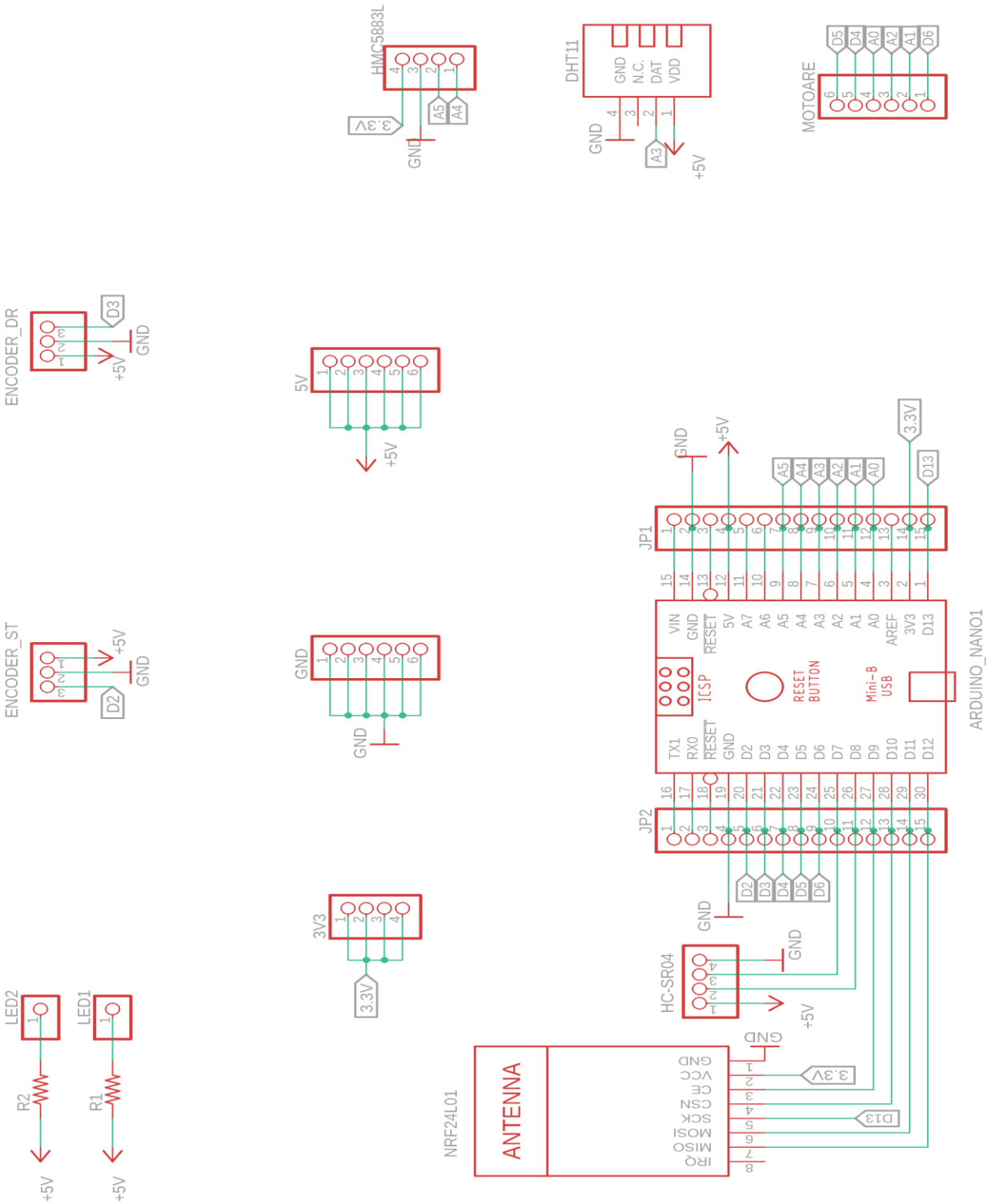
[34] https://github.com/johnrickman/LiquidCrystal_I2C , accesat la data 11.06.2020

[35] http://ep.etc.tuiasi.ro/site/Electronica%20Industriala/referate%20laborator/ciclul_3/L11%20-%20Convertor%20boost%202007.pdf ,accesat la data 18.06.2020

# Anexa 1 Schema electrică a robotului mobil

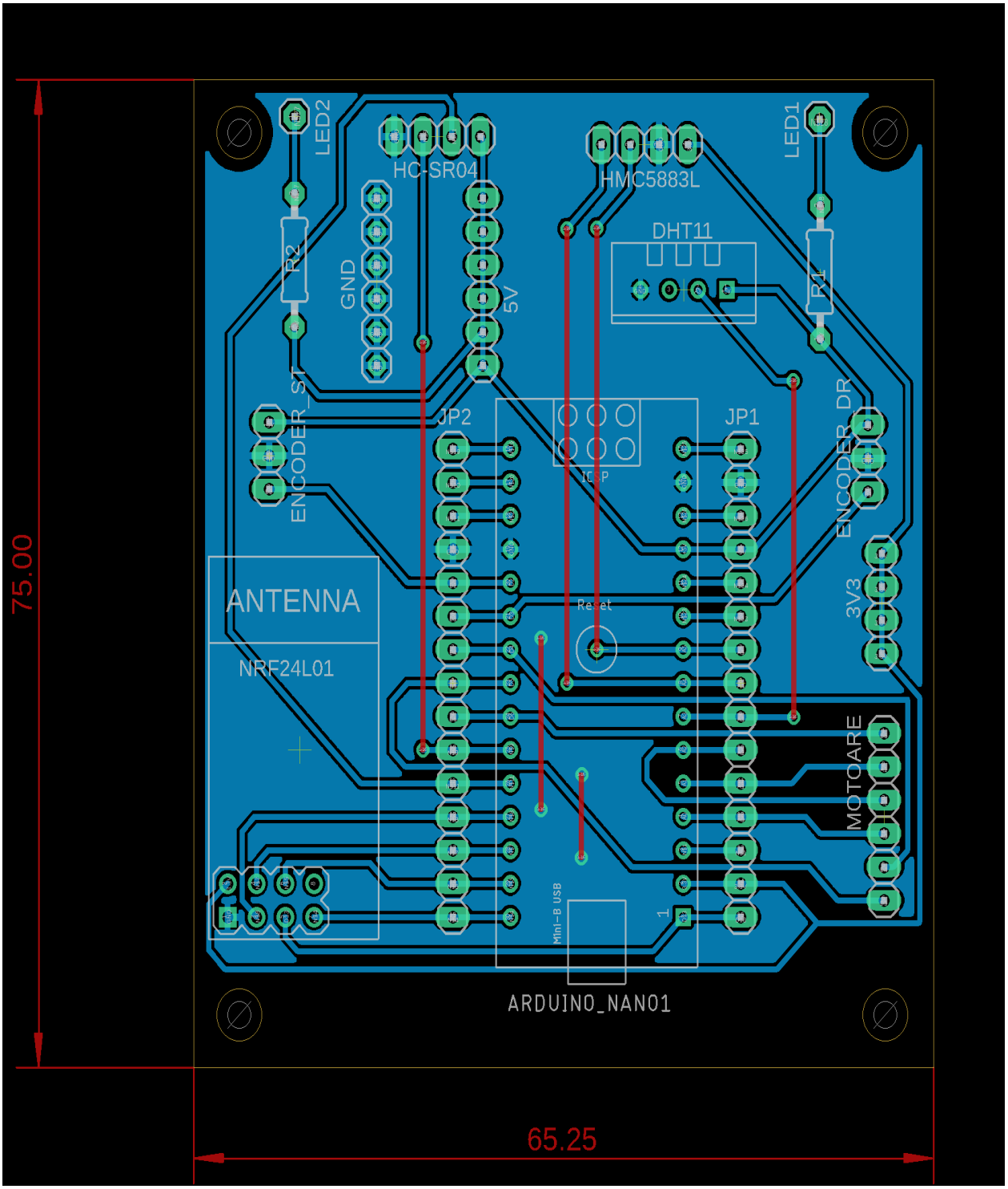


# Anexa 2 Schema electrică a plăcii cu cablaj imprimat

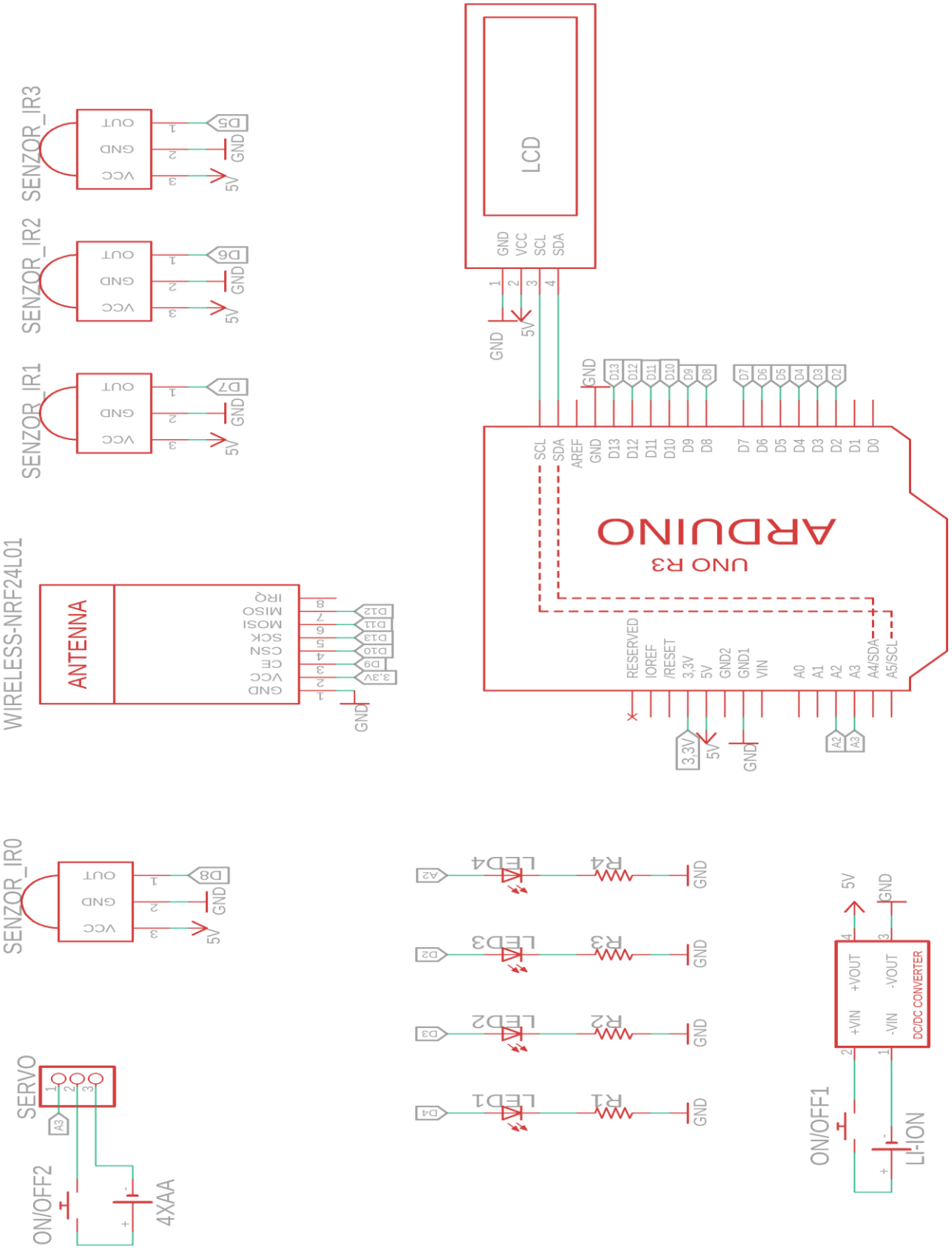


# Anexa 3 Vedere layout a plăcii cu cablaj imprimat

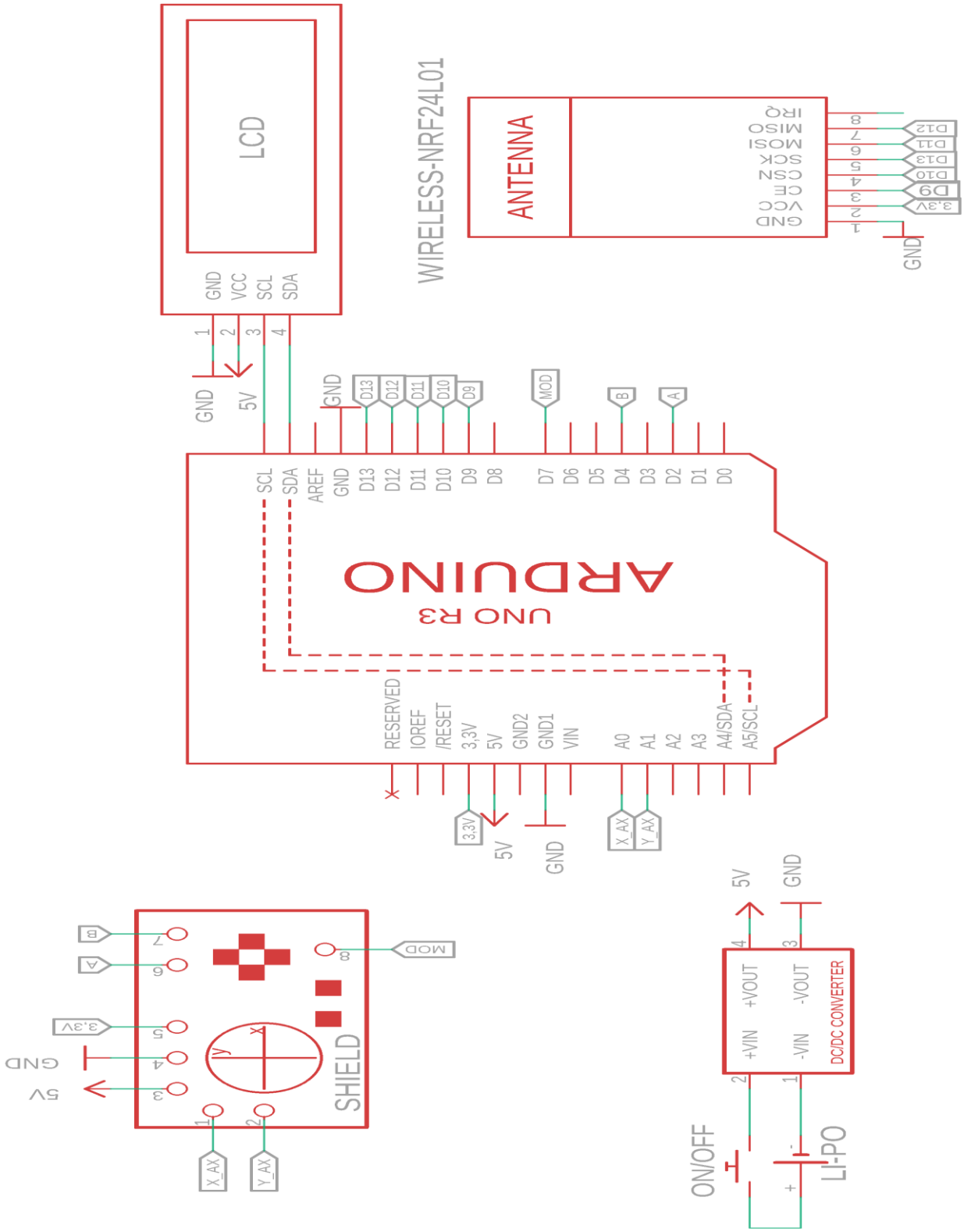
`



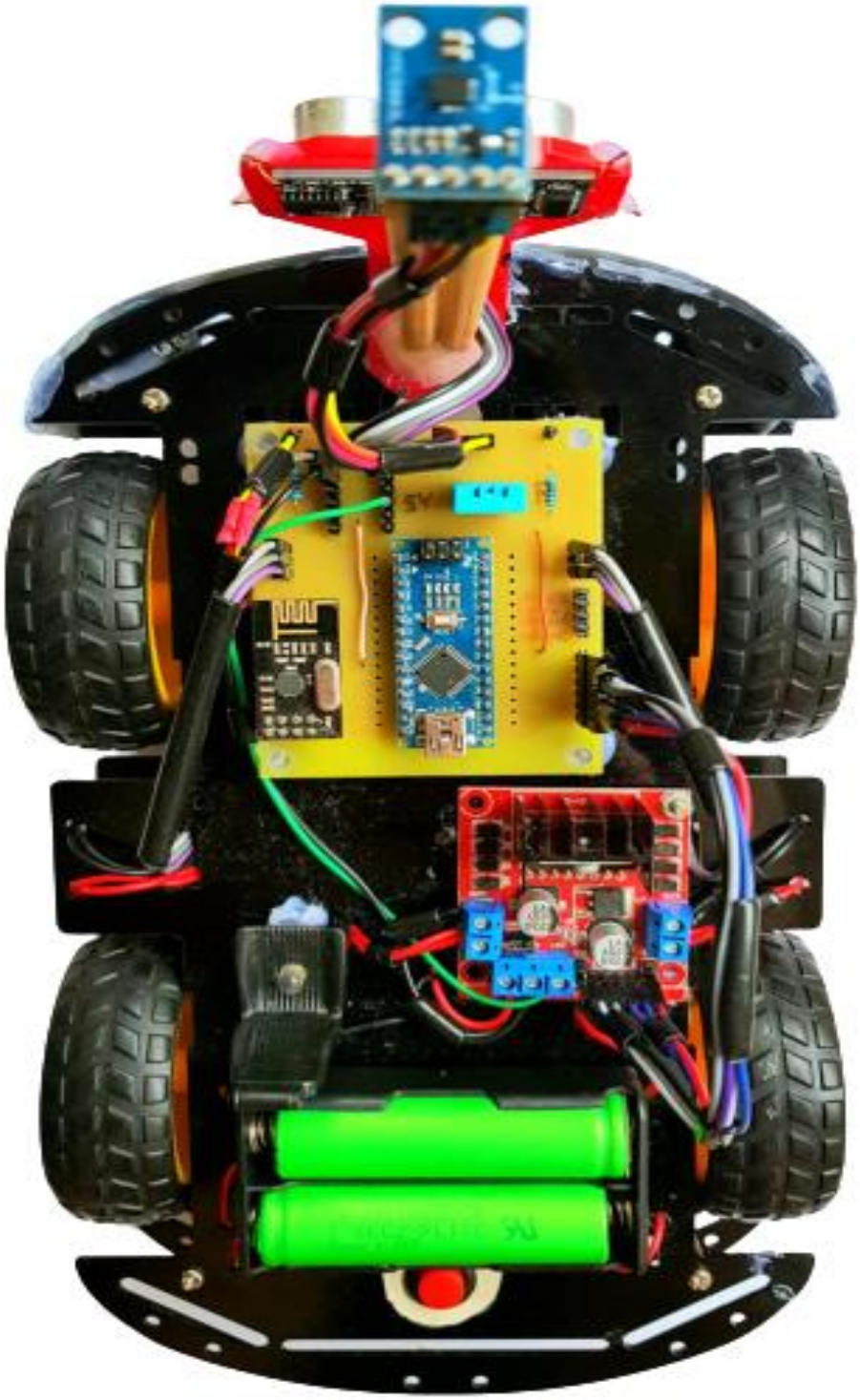
# Anexa 4 Schema electrică a parcării



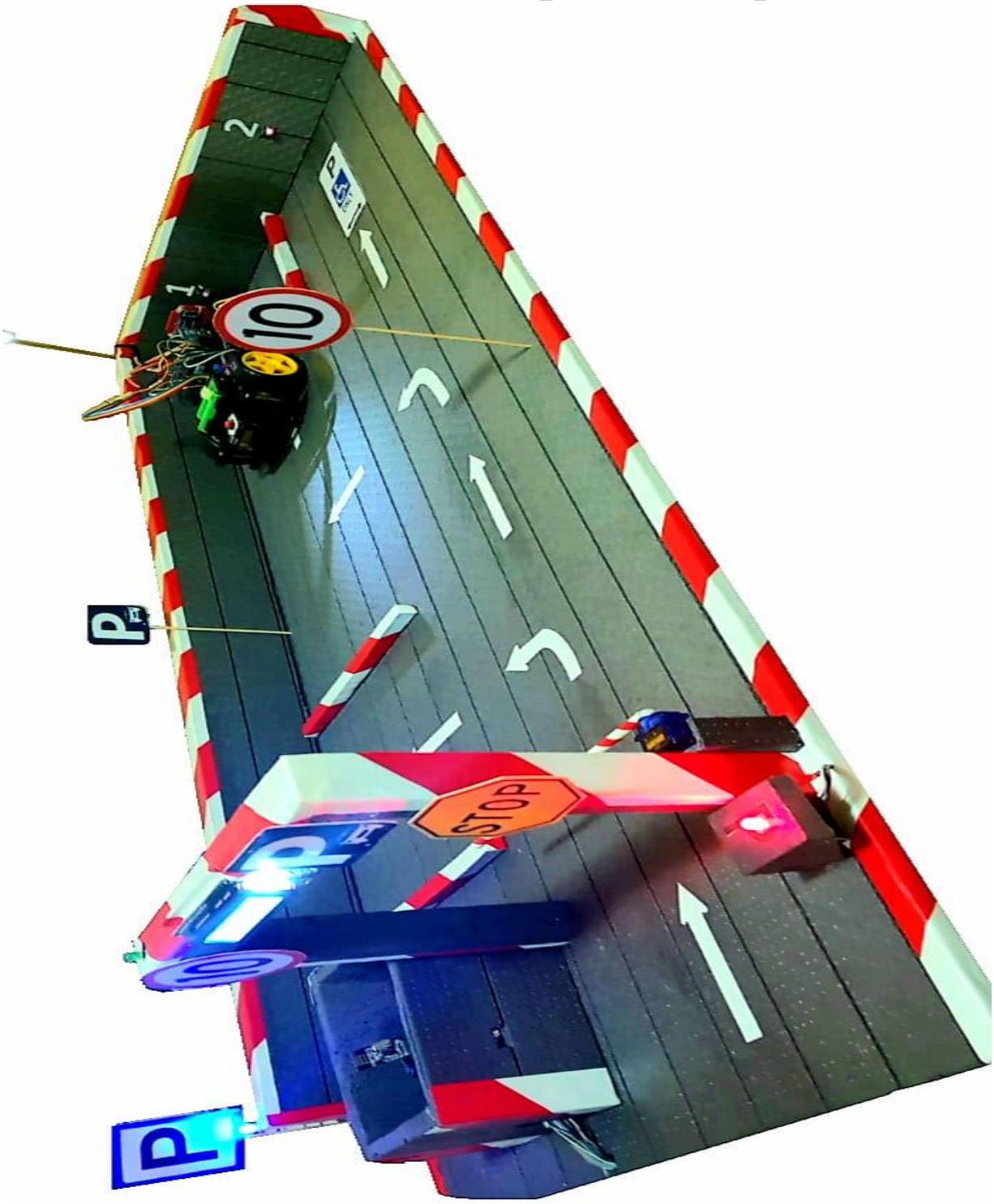
# Anexa 5 Schema electrică a telecomenzii



# Anexa 6 Realizarea practică a robotului mobil



# Anexa 7 Realizarea practică a parcării



# **Anexa 8** Realizarea practică a telecomenzii

# Anexa 9 Codul sursă al robotului mobil

```
#include <nRF24L01.h>
#include <printf.h>
#include <RF24.h>
#include <RF24_config.h>
#include <SimpleDHT.h>
#include <QMC5883L.h>
#include<Wire.h>
QMC5883L compass;
int pozitie;
int schimba = 0;
// daca schimba = 0 comunicația se realizează
// între telecomanda- masina
// daca schimba = 1 comunicația se realizează
// între masina - parcare
int ok=0;

//~~~~~~~~~~~~~~~~~~HC_SR04~~~~~~~~~~~~~~~~~~
#include "SR04.h"
#define TRIG_PIN   8
#define ECHO_PIN   7
long distanta;
SR04 sr04 = SR04(ECHO_PIN,TRIG_PIN);

//~~~~~~~~~~~~~~~~~~MOTOARE~~~~~~~~~~~~~~~~~~
const byte encoder_stanga  = 2;
const byte encoder_dreapta = 3;
const int diametru_roata = 67; //mm
const float circumferinta = diametru_roata *
PI ;  //valoare în mm
const float tranzitii_pe_rotatie = 40;
volatile int numarator_ISR1 = 0;// encoder
//stanga
volatile int numarator_ISR2 = 0;// encoder
//dreapta.
#define en_stanga    6 // enable motoare
#define en_dreapta   5 // enable motoare
// ~~~~~~motoarele din stanga~~~~~~~
#define s_inainte  A1 // in1 driver
#define s_inapoi   A2 // in2 driver
//~~~~~~motoarele din dreapta~~~~~~
#define d_inainte A0 // in3 driver
#define d_inapoi  4 // in4 driver

//~~~~~~~~~~~~~~~~~~NRF~~~~~~~~~~~~~~~~~~~~~~
RF24 radio(9,10);//CE, CSN
// adresa de comunicație cu mașina
const byte adresa[6] = "00001";
// adresa de comunicație cu parcarea
const byte adresa_parcare[6] = "00011";

//~~~~~~~~~~~~~~~~~~DHT11~~~~~~~~~~~~~~~~~~~~
unsigned long timp;
int pin_dht11 = A3;
SimpleDHT11 dht11;
byte temperatura = 0;
byte umiditate   = 0;

//~~~Rutina de răspundere la întreruperi~~~
void ISR1 ()
       { //  MOTOR STANGA
        numarator_ISR1 ++ ;
       }
void ISR2 ()
        {//  MOTOR DREAPTA
        numarator_ISR2 ++ ;
         }
//~~~~~Funcție pentru oprirea motoarelor~~~~
```

```
void oprire_motoare(){
  analogWrite(en_stanga  , 0);
  analogWrite(en_dreapta , 0);
  digitalWrite(s_inainte , LOW);
  digitalWrite(s_inapoi  , LOW);
  digitalWrite(d_inainte , LOW);
  digitalWrite(d_inapoi  , LOW);
}

//~~~~~~Funcţia pentru mers înainte~~~~~~~
void inainte (float distanta , int viteza) {
// setez directia de deplasare
  digitalWrite(s_inainte , HIGH);
  digitalWrite(s_inapoi  , LOW);
  digitalWrite(d_inainte , HIGH);
  digitalWrite(d_inapoi  , LOW);
  numarator_ISR1 = 0;
  numarator_ISR2 = 0;

  float numar_de_rotatii = (distanta *10) /
circumferinta ;
  unsigned long pasi = numar_de_rotatii *
tranzitii_pe_rotatie;
    while ( (numarator_ISR1 < pasi) && (
numarator_ISR2 < pasi) ) {
   if(numarator_ISR1<pasi){
      analogWrite( en_stanga , viteza) ;
                          } else
{    analogWrite( en_stanga, 0);
                          }
   if(numarator_ISR2 < pasi ){
      analogWrite( en_dreapta, viteza);
                          }else {
analogWrite( en_dreapta, 0);
                          }
  }
   oprire_motoare(); // opresc motoarele
   numarator_ISR1 = 0;
   numarator_ISR2 = 0;
}

//~~~~~~~~~~FUNCTIA PENTRU DREAPTA~~~~~~~~
void dreapta (int viteza, int grade) {
int PozInit,PozCurenta,target,diferenta;
int FlagOprire = 0;
int indice_LD = -1, indice_LS = -1;
int LS[5]= {0, 0, 0, 0, 0};// limita stanga
int LD[5]= {0, 0, 0, 0, 0};// limita dreapta
  digitalWrite(s_inainte , HIGH);
  digitalWrite(s_inapoi  , LOW);
  digitalWrite(d_inainte , LOW);
  digitalWrite(d_inapoi  , HIGH);
  numarator_ISR1 = 0;
  numarator_ISR2 = 0;
  int pasi = 10;
//citesc pozitia initiala
  PozInit = compass.readHeading();
 // calculez target-ul
 target = PozInit + grade;
  if(target > 360) {
    target = target - 360;
  }
for(int i=0;i<5;i++){
// inițializarea vectorului LS
  LS[i]= target+i-5;
  if(LS[i] <= 0) {
    LS[i] = LS[i] + 360;
  }
}
for(int j=0;j<5;j++){
// inițializarea vectorului LD
  LD[j]= target+j+1;
  if(LD[j] > 360) {
    LD[j] = LD[j] - 360;
```

```
  }
}

 do{

   numarator_ISR1 = 0;
   numarator_ISR2 = 0;
  while ( (numarator_ISR1 < pasi) && (
numarator_ISR2 < pasi) )
      {
        if(numarator_ISR1<pasi){
       analogWrite( en_stanga , viteza) ;
                  }  else {
       analogWrite( en_stanga, 0);
                          }
        if(numarator_ISR2 < pasi ){
             analogWrite( en_dreapta,
viteza);
                                    }
                                     else {

analogWrite( en_dreapta, 0);
                                          }
       }

   numarator_ISR1 = 0;
   numarator_ISR2 = 0;

PozCurenta = compass.readHeading();

for(int k=0; k<5;k++){
  if( (PozCurenta == LS[k]) || (PozCurenta
== LD[k]) )
  {
    if(PozCurenta == LS[k])
              {
                indice_LS = k;
              }
                  else {
                       indice_LD = k;
                        }
    FlagOprire = 1;
    break;
  }
}
 }while(FlagOprire == 0); // inchid do while
 oprire_motoare(); // opresc motoarele
 delay(2000) ;
// ~~~~~~~~~~~AJUSTARE POZIȚIE ~~~~~~~~
if( !(indice_LS == -1) ) {
 //prea mult stanga,poziția trebuie ajustată
  diferenta = 5 - indice_LS;
// setarea deplasarii spre dreapta
  digitalWrite(s_inainte , HIGH);
  digitalWrite(s_inapoi  , LOW);
  digitalWrite(d_inainte , LOW);
  digitalWrite(d_inapoi  , HIGH);
  analogWrite(en_stanga,255);
  analogWrite(en_dreapta,255);
  delay(diferenta * 15);
  oprire_motoare();
} else if( !(indice_LD == -1) ) {
//prea mult dreapta,poziția trebuie ajustată
  diferenta = indice_LD + 1 ;
  // setarea deplasarii spre stanga
  digitalWrite(s_inainte , LOW);
  digitalWrite(s_inapoi  , HIGH);
  digitalWrite(d_inainte , HIGH);
  digitalWrite(d_inapoi  , LOW);
  analogWrite(en_stanga,255);
  analogWrite(en_dreapta,255);
  delay(diferenta * 15);
  oprire_motoare();
}
```

```
}
//~~~~~~~FUNCTIA PENTRU STANGA~~~~~~~~
void stanga(int viteza, int grade) {

int PozInit,PozCurenta , target,diferenta;
int FlagOprire = 0;
int indice_LD = -1, indice_LS = -1;
int LS[5]= {0, 0, 0, 0, 0}; //limita stanga
int LD[5]= {0, 0, 0, 0, 0}; //limita dreapta
// setez directia de deplasare pentru stanga
  digitalWrite(s_inainte , LOW);
  digitalWrite(s_inapoi  , HIGH);
  digitalWrite(d_inainte , HIGH);
  digitalWrite(d_inapoi  , LOW);
  numarator_ISR1 = 0;
  numarator_ISR2 = 0;
  int pasi = 10;
//citesc pozitia initiala
  PozInit = compass.readHeading();
  target = PozInit - grade;
// calculez target-ul
  if(target < 360)
      {
       target = target + 360;
      }
for(int i=0;i<5;i++)
{
//initializerea vectorului LS
  LS[i]= target+i-5;
  if(LS[i] <= 0)
      {
       LS[i] = LS[i] + 360;
      }
}
for(int j=0;j<5;j++)
{
//initializarea vectorului LD
  LD[j]= target+j+1;
  if(LD[j] > 360)
      {
       LD[j] = LD[j] - 360;
      }
}
  do{
   numarator_ISR1 = 0;
   numarator_ISR2 = 0;
        while ( (numarator_ISR1 < pasi) && (
numarator_ISR2 < pasi) ){
        if(numarator_ISR1<pasi){
         analogWrite( en_stanga , viteza) ;
                                }
                                  else {
        analogWrite( en_stanga, 0);
                                       }
        if(numarator_ISR2 < pasi ){
          analogWrite( en_dreapta, viteza);
                                    }
                                     else {
        analogWrite( en_dreapta, 0);
                                          }
        }
   numarator_ISR1 = 0;
   numarator_ISR2 = 0;
PozCurenta = compass.readHeading();
for(int k=0; k<5;k++){
if( (PozCurenta == LS[k]) ||
                        (PozCurenta == LD[k]))
   {
    if(PozCurenta == LS[k])
                  {
                   indice_LS = k;
                  }
```

```
                    else {
                         indice_LD = k;
                          }
    FlagOprire = 1;    // setez flag ul de
oprire 1
    break;
  }
}
 }while(FlagOprire == 0);
oprire_motoare(); // opresc motoarele
delay(2000) ;

//~~~~~~~~~~~~~~AJUSTARE ~~~~~~~~~~~~~~
if( !(indice_LS == -1) ) {
//prea mult stanga, poziția trebuie ajustată
 diferenta = 5 - indice_LS;
//Setarea deplasarii la dreapta
  digitalWrite(s_inainte , HIGH);
  digitalWrite(s_inapoi  , LOW);
  digitalWrite(d_inainte , LOW);
  digitalWrite(d_inapoi  , HIGH);

  analogWrite(en_stanga,255);
  analogWrite(en_dreapta,255);
  delay(diferenta * 15);
  oprire_motoare();
} else if( !(indice_LD == -1) ) {
//prea mult dreapta,poziția trebuie ajustată
  diferenta = indice_LD + 1 ;
//Setarea deplasarii la stanga
  digitalWrite(s_inainte , LOW);
  digitalWrite(s_inapoi  , HIGH);
  digitalWrite(d_inainte , HIGH);
  digitalWrite(d_inapoi  , LOW);

  analogWrite(en_stanga,255);
  analogWrite(en_dreapta,255);
  delay(diferenta * 15);
  oprire_motoare();

}
}

void setup() {
 // ~~~~~~~~~~~~FUNCTIA SETUP~~~~~~~~~~~
  pinMode(encoder_stanga ,INPUT_PULLUP);
  pinMode(encoder_dreapta,INPUT_PULLUP);
  pinMode(en_stanga , OUTPUT);
  pinMode(en_dreapta, OUTPUT);
  pinMode(s_inainte , OUTPUT);
  pinMode(s_inapoi  , OUTPUT);
  pinMode(d_inainte , OUTPUT);
  pinMode(d_inapoi  , OUTPUT);
  Wire.begin();
  compass.init();
  compass.setSamplingRate(50);
 // Serial.begin(9600);
  radio.begin();
  radio.setChannel(115);
  radio.setPALevel(RF24_PA_MAX);
  radio.setDataRate( RF24_250KBPS );
  radio.openReadingPipe(1,adresa);
  radio.startListening();
  attachInterrupt(0 ,ISR1, CHANGE);
  attachInterrupt(1 ,ISR2, CHANGE);
  timp=millis();

}
void loop() {
 // ~~~~~~~~~~~~~~FUNCTIA LOOP~~~~~~~~~~~~~
  pozitie = compass.readHeading();
if ( schimba == 0 ) {
// comunic cu telecomanda
```

```
  if(radio.available())
  {
    char dataReceptionata[4] = {0};
    radio.read(&dataReceptionata, 4);
  // Serial.println(dataReceptionata);

  // ~~~~~~~~CU CINE COMUNIC?~~~~~~~~~
  if(dataReceptionata[2] == 't')
              {
              schimba= 1;
// comunicare intre masina-parcare
              }
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
if(dataReceptionata[3] == 'f')
{
 // Daca nu este apasata frana
//~~~~~~~~~~~~~~~~~INAINTE~~~~~~~~~~~~~~~~
    if(dataReceptionata[0]== 'f')
    {
      digitalWrite(s_inainte , HIGH);
      digitalWrite(s_inapoi  , LOW);
      digitalWrite(d_inainte , HIGH);
      digitalWrite(d_inapoi  , LOW);
      analogWrite(en_stanga ,255);
      analogWrite(en_dreapta,255);
    }
    else if(dataReceptionata[0] == 'd')
    {
   //~~~~~~~~~~~~~~~~INAPOI~~~~~~~~~~~~~~~~
      digitalWrite(s_inainte , LOW);
      digitalWrite(d_inainte , LOW);
      digitalWrite(s_inapoi  , HIGH);
      digitalWrite(d_inapoi  , HIGH);
      analogWrite(en_stanga ,255);
      analogWrite(en_dreapta,255);
    } else if(dataReceptionata[0]=='x' &&
dataReceptionata[1]=='x' )
    { // MOTOARELE SUNT OPRITE
      analogWrite(en_stanga , LOW);
      analogWrite(en_dreapta, LOW);
    }
    //~~~~~~~~~~~~~~~STANGA~~~~~~~~~~~~~~~
    if(dataReceptionata[1]== 'r')
    {
      digitalWrite(s_inainte , HIGH);
      digitalWrite(d_inainte , LOW);
      digitalWrite(s_inapoi  , LOW);
      digitalWrite(d_inapoi  , HIGH);
      analogWrite(en_stanga ,255);
      analogWrite(en_dreapta,255);
    }
    else if(dataReceptionata[1] == 'l')
    {
    //~~~~~~~~~~~~~~~DREAPTA~~~~~~~~~~~~~~~
      digitalWrite(s_inainte , LOW);
      digitalWrite(d_inainte , HIGH);
      digitalWrite(s_inapoi  , HIGH);
      digitalWrite(d_inapoi  , LOW);

      analogWrite(en_stanga ,255);
      analogWrite(en_dreapta,255);

    }
}
  else{
      //~~~~~~~~~~~~~~~FRANA~~~~~~~~~~~~~~~
      oprire_motoare();
      }
}

if((millis()-timp>=6000) ){
```

```
//odata la 6 secunde se transmit catre
//masina informatii privind temperatura si
//umiditatea

   dht11.read(pin_dht11,&temperatura,
&umiditate, NULL) ;

   String informatie;

   String informatie1;

   informatie1=String(int(umiditate));

   informatie=String(int(temperatura));

   String totalinf = informatie +
informatie1 ;

   char transmit_inf[5];

   totalinf.toCharArray(transmit_inf,5);

  radio.stopListening();

  radio.openWritingPipe(adresa);

  for(int k=0;k<20;k++){

 // transmit informatia catre telecomanda

radio.write(&transmit_inf,

sizeof(transmit_inf));

  //Serial.println(transmit_inf);

  }

  timp=millis(); // resetez timpul

// setez masina ca receptor pentru a primi
//comenzi de la telecomanda

  radio.openReadingPipe(1,adresa);

  radio.startListening();

  }

  }

   else  {//schimba=1

  //~~~~~~ COMUNIC CU PARCAREA ~~~~~~~~

  //---------------------------------------

  radio.stopListening();

  delay(50);

  radio.openReadingPipe(1,adresa_parcare);

  radio.startListening();

 if( radio.available() )

   {

      delay(1000);

      char date_locuri[3] = {0};

      radio.read(&date_locuri, 3);

//  Serial.println(String(date_locuri));
//  o = ocupat; l = liber

// Serial.println(date_locuri[0]);//locul 1

// Serial.println(date_locuri[1]);//locul 2

 // Serial.println(date_locuri[2]);//locul 3


// ~~~~~~~~~~~ UNDE PARCHEZ? ~~~~~~~~~~~~

if(ok==0)

{

if(date_locuri[1] == 'l' )

{// daca locul 2 este liber

  //~~~~~~~~~~~~~~~ TRASEU 2~~~~~~~~~~~~~~

  do{

    distanta=sr04.Distance();

    Serial.println(distanta);

    inainte(5,128);

    }while(distanta>10);

    ok=1; // Mașina a fost parcată

  //~~~~~~~~~~~~~FINAL TRASEU2~~~~~~~~~~

} else if ( date_locuri[2] == 'l' )

          { //daca locul 3 este liber


//~~~~~~~~~~~~~~~~TRASEU3~~~~~~~~~~~~~~~~

        do{

              distanta=sr04.Distance();

              Serial.println(distanta);

              inainte(5,128);

           }while(distanta>55);// primul pas

          delay(2000);

          stanga(150,80); // al 2 lea pas

          delay(2000);

           do{

           distanta=sr04.Distance();

           inainte(5,120);

          }while(distanta>10);// al 3lea pas

            ok=1;//Mașina a fost parcată
```

```
          //~~~~~~~~~~FINAL TRASEU3~~~~~~~                        {// daca locul 1 este liber
          }
           else if ( date_locuri[0] == 'l' )
//~~~~~~~~~~~~ TRASEU 1 ~~~~~~~~~~~~
  inainte(80,255); // Primul pas
  delay(2000);
  stanga(160,80); // Al 2 lea pas
    do{
      distanta=sr04.Distance();
      inainte(5,128); // Al 3 lea pas
       }while(distanta>10);
       delay(2000);
       dreapta(160,80); // Al 4 lea pas
       delay(2000);
 do{
distanta=sr04.Distance();// Al 5 lea pas
    inainte(5,128);
   }while(distanta>10);
  ok=1; // Mașina a fost parcată
//~~~~~~~~~ FINAL  TRASEU 1 ~~~~~~~~
           } else {
//daca toate locurile sunt ocupate,
//motoarele vor fi oprite
        oprire_motoare();
       }
}
else {
// daca am parcat, opresc motoarele
  oprire_motoare();
      }
    }
  }
}
```

# Anexa 10 Codul sursă al parcării inteligente

```
#include <Servo.h>
#include <RF24.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <nRF24L01.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
int led_albastru = 4; // led albastru
int led_alb = 3; // led galben
int led_r = 2; // led rosu
int led_v = A2; // led verde
Servo bariera;
String loc1,loc2,loc3;
RF24 radio(9,10); //CE, CSN
const byte adresa_parcare[6] = "00011";

//~~~~~~~~~~~~~~~~~~SETUP~~~~~~~~~~~~~~~~~~~
void setup() {
//Serial.begin(9600);
pinMode(8,INPUT); // Senzor IR_bariera
pinMode(7,INPUT); // Senzor IR_loc_1
pinMode(6,INPUT); // Senzor IR_loc_2
pinMode(5,INPUT); // Senzor IR_loc_3

pinMode(led_albastru,OUTPUT);
pinMode(led_alb,OUTPUT);
pinMode(led_r,OUTPUT);
pinMode(led_v,OUTPUT);

bariera.attach(A3);
bariera.write(0);
radio.begin();
radio.setChannel(115);
radio.setPALevel(RF24_PA_MAX);
radio.setDataRate(RF24_250KBPS);
radio.openWritingPipe(adresa_parcare);

lcd.begin();
lcd.backlight();

// SE APRIND LED-URILE
digitalWrite(led_albastru,HIGH);
digitalWrite(led_alb,HIGH);
delay(1000);
}

//~~~~~~~~~~~~~~~~~~LOOP~~~~~~~~~~~~~~~~~~~~
void loop() {

 int senzor_bariera = digitalRead(8);
 int senzor_loc1    = digitalRead(7);
 int senzor_loc2    = digitalRead(6);
 int senzor_loc3    = digitalRead(5);
 int contor=0;
//~~~~~~VERIFIC PRIMUL LOC DE PARCARE~~~~~
 if(senzor_loc1 == LOW){
    loc1='o'; // ocupat
    contor=contor+1;
       }else{
             loc1='l'; //liber
            }
//~~~~~VERIFIC AL 2 LEA LOC DE PARCARE~~~~~
```

```
 if(senzor_loc2 == LOW){
    loc2='o'; // ocupat
    contor=contor+1;
      }else{
            loc2='l'; //liber
           }
//~~~~~VERIFIC AL 3 LEA LOC DE PARCARE~~~~~
 if(senzor_loc3 == LOW){
    loc3='o'; // ocupat
    contor=contor+1;
      }else{
            loc3='l'; //liber
           }
//~~~~~~~~~~~ ACȚIONAREA BARIEREI ~~~~~~~~


if (contor == 3)
    {
// daca toate locurile sunt ocupate
  digitalWrite(led_v ,LOW);
  bariera.write(0); // bariera ramâne
coborâtă
    } else if(senzor_bariera == LOW)
            {
 // daca se află mașină la intrare și este
cel putin un loc liber
              digitalWrite(led_v,HIGH);
              delay(100);
              bariera.write(90); // se
permite accesul
              delay(500);
            } else
                 {
                  delay(3000);
                  bariera.write(0); //
bariera coborata
                  delay(100);
                  digitalWrite(led_v,LOW);
                  }
//AFIȘAREA INFORMAȚIILOR PE LCD
if(loc1 == "o" && loc2 == "o" &&
loc3 == "o")
  {
  digitalWrite(led_r,HIGH);


  lcd.clear();
  lcd.setCursor(1,0);
  lcd.print("PARCAREA ESTE");
  lcd.setCursor(4,1);
  lcd.print("OCUPATA!");

  } else {

    digitalWrite(led_r,LOW);
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Loc1");
    lcd.setCursor(6,0);
    lcd.print("Loc2");
    lcd.setCursor(12,0);
    lcd.print("Loc3");

    if (loc1 == "o")
        {
          lcd.setCursor(2,1);
          lcd.print("x");
        } else {
            lcd.setCursor(2,1);
            lcd.print("v");
               }
    if (loc2 == "o")
        {
          lcd.setCursor(7,1);
```

```
            lcd.print("x");                                    }
          } else {                                    if (loc3 == "o")
               lcd.setCursor(7,1);
               lcd.print("v");
         {
            lcd.setCursor(13,1);
            lcd.print("x");
          } else {
              lcd.setCursor(13,1);
              lcd.print("v");
                 }
           }
// TRANSMIT CATRE MASINA INFORMATIILE
String text;
text = loc1 + loc2 + loc3;
char text1[4];
text.toCharArray(text1,4);
radio.write(&text1,sizeof(text1));

}
```

# Anexa 11 Codul sursă al telecomenzii

```
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
LiquidCrystal_I2C lcd(0x27, 16 ,2);
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

RF24 radio(9, 10);  // CE, CSN
const byte adresa[6] = "00001";

int POT_X= A0;
int POT_Y= A1;
int date_pin = 2;
int frana_pin = 4;
int mod = 7 ;
int x_val,y_val;
char schimba;
String dir0,dir1,frana;
// DIR1 = STANGA/DREAPTA; DIR0 = FATA/SPATE
int temperatura, umiditate;

//~~~~~~~~~~~~~~~~~~SETUP~~~~~~~~~~~~~~~~
void setup() {

  pinMode(POT_X, INPUT);
  pinMode(POT_Y, INPUT);
  pinMode(mod  , INPUT);
  pinMode(date_pin, INPUT);
  pinMode(frana_pin , INPUT);
  //Serial.begin(9600);
  radio.begin();
  radio.setChannel(115);
  radio.setPALevel(RF24_PA_MAX);
  radio.setDataRate(RF24_250KBPS);
  radio.openWritingPipe(adresa);
  lcd.begin();
  lcd.backlight();
}

void loop() {
//~~~~~~~~~~~~~~~LOOP~~~~~~~~~~~~~~~~
//~~~~~~~~~date de la masina~~~~~~~~
 if(digitalRead(date_pin) == LOW){
//daca butonul de date este apasat, initiez
//modul de recepţie
     radio.openReadingPipe(1, adresa);
     radio.startListening();


  if(radio.available())
  {
// daca receptionează date
  char dataReceptionata[5] = {0};
  radio.read( &dataReceptionata, 5);
  // Serial.println(dataReceptionata);
  String informatie, informatie1;
  informatie = String( dataReceptionata[0])
+ String(dataReceptionata[1]);
    temperatura = informatie.toInt();

  informatie1 = String( dataReceptionata[2])
+ String(dataReceptionata[3]);
    umiditate = informatie1.toInt();

   // afisez informatiile pe LCD
    lcd.setCursor(0, 0);
    lcd.print("Temp: ");
    lcd.print(temperatura);
    lcd.print("(grade)");
    lcd.setCursor(0, 1);
    lcd.print("Umiditate: ");
    lcd.print(umiditate);
    lcd.print(" %");
    }
    }
       else {
// daca nu este apăsat butonul, initiez
//modul de transmisie
radio.stopListening();
radio.openWritingPipe(adresa);
```

```
            }
//~~~~~~~ este frâna acţionată ? ~~~~~~~~
 if(digitalRead(frana_pin) == HIGH)
 {
  frana = 'f';
  //f= neapasat
 }
 else
{
 frana ='t';
 //t= apasat
}

 if( digitalRead (mod) == HIGH )
 {
   schimba = 'f';
 }
 else {
// daca este apăsat, comunicaţia se va
//realiza între parcare şi maşină
      schimba = 't';
      }

 x_val = analogRead(POT_X);
// citesc valoarea de la joystick
  y_val = analogRead(POT_Y);
// citesc valoarea de la joystick



//~~~~AFLU DIRECTIA Y (inainte/inapoi)~~~~
  if (y_val <= 450)
  {
    dir0 = 'd'; //inapoi
  }
  if (y_val >= 550)
  {
    dir0 = 'f'; //inainte
  }
  if (y_val > 450 && y_val < 550)
  {
    dir0 = 'x'; // stop
  }
//~~~AFLAM DIRECTIA X (dreapta/stanga)~~~~
  if (x_val <= 200)
  {
    dir1 = 'l'; //left
  }
  if (x_val >= 800)
  {
    dir1 = 'r'; //right
  }
  if (x_val > 200 && x_val < 800)
  {
    dir1 = 'x'; // stop
  }


// ~~~Transmit informaţiile către maşină~~~
String data;
data = dir0 + dir1 + schimba + frana ;
char dataTransmisa[5];
data.toCharArray(dataTransmisa,sizeof(dataTr
ansmisa));
radio.write(&dataTransmisa,sizeof(dataTransm
isa));
//Serial.println(String(dataTransmisa));


}
```