Robot cu sistem de parcare automat

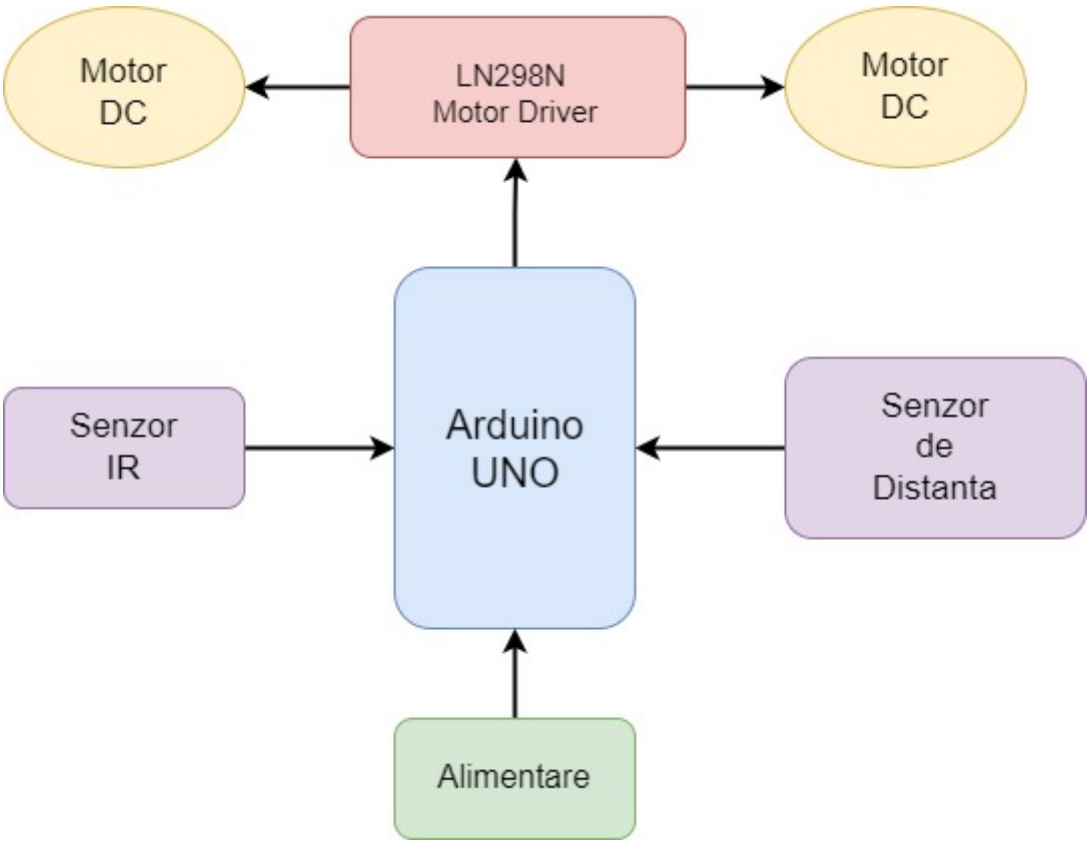
Introducere

- Acest proiect implementeaza o masinuta controlata prin telecomanda, echipata cu optiunile de controlare a directiei si de parcare automata.
- Scopul proiectului este acela de a facilita parcarea unei masini și de a preveni o eventuala coliziune.

Descriere generală

Functionalitatea de baza a proiectului este detectia de obstacole in spatele, cat si in partile laterale ale masinutei. Aceasta va fi dotata cu un senzor cu ultrasunete, montat pe un suport, pentru a determina distanta fata de obiecte. Sasiul se poate controla printr-un senzor de captare a luminii infrarosii de la o telecomanda comuna (de televizor) si un motor driver L298N folosit pentru a pune in miscare cele 2 roti. Datorita rotii ajutatoare din fata (cea de a treia roata), masinuta se va putea si rotii. De asemenea, apasand un buton ales arbitrar de pe telecomanda, masinuta va intra in modul de parcare automata si se va deplasa cu spatele pana cand va detecta o coliziune, caz in care placuta Arduino va solicita oprirea motoarelor.

Schema bloc

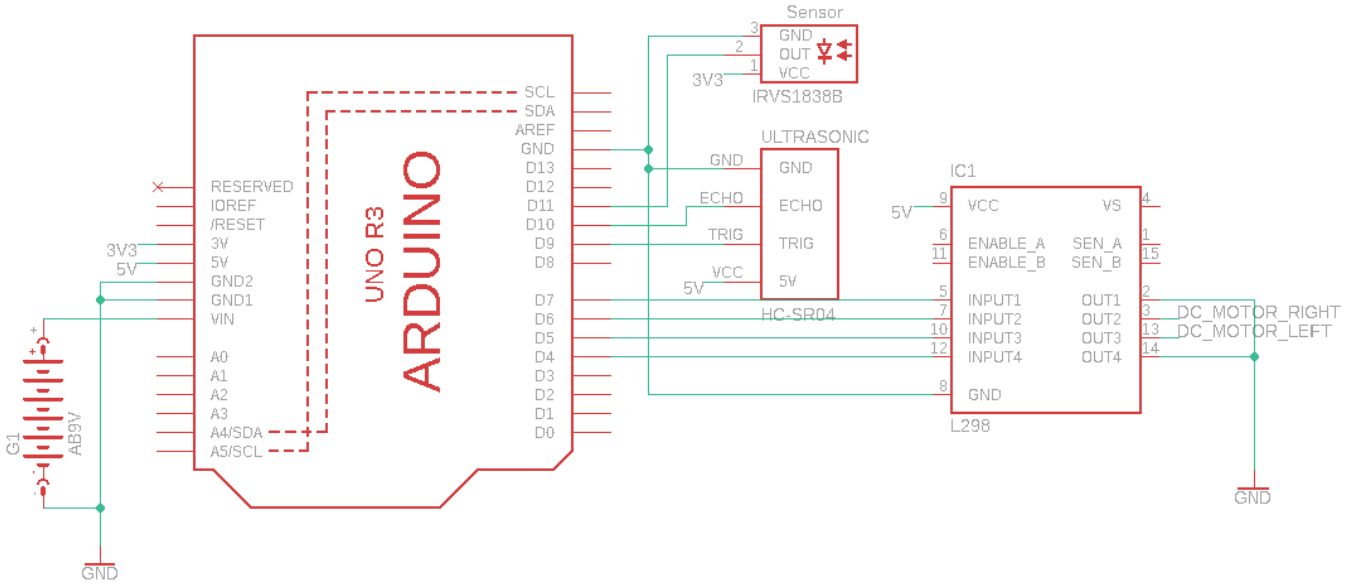


Hardware Design

Lista de piese:

- Arduino UNO
- Kit de Robot Premium din Aluminiu
- Doua Motoare DC
- Driver de Motoare Dual L298N
- Modul Receptor Telecomandă Infraroșu
- Senzor ultrasonic HC-SR04
- Suport pentru Senzorul Ultrasonic HC-SR04
- Jumper Cables
- Suport baterie 9V
- Baterie 9V

Schema Electrica



Software Design

Organizare

Odata ce circuitul masinutei este alimentat de la baterie, in **setup()** se vor initializa senzorul ultrasonic si de captare a luminii infrarosii, precum si pinii motoarelor DC care asteapta semnale de la telecomanda.

Pentru o organizare mai buna a logicii programului, am creat cateva functii ajutatoare:

```
bool detectCollision();
void controlMotors(int IN1_val, int IN2_val, int IN3_val, int IN4_val);
void stopMotors();
```

Detectarea coliziunilor

- Verificam daca masinuta este in miscare deoarece stand pe loc nu exista posibilitatea de lovire;
- Daca merge cu fata, masinuta se va indeparta de obiectul din spatele acesteia, deci ii permitem miscarea;
- Masinuta se va opri daca detecteaza vreun obiect la o distanta mai mica sau egala cu 10cm.

```
bool detectCollision()
{
  if (isRunning && (distance > 0 && distance <= 10) && !goFront)
    return true;

  return false;
}
```

Biblioteca **<NewPing.h>** implementeaza calcularea distantei in asa fel incat daca un obiect este mai departe decat distanta maxima la care este setat senzorul, acesta va intoarce distanta 0.

Controlul motoarelor

- Daca exista vreun obstacol in directia spre care urmeaza sa porneasca masinuta, motoarele nu vor porni deoarece vor fi imediat oprite in bucla **loop()**;
- In cazul in care motoarele pornesc, semnalizam faptul ca masinuta se afla in miscare.

```
void controlMotors(int IN1_val, int IN2_val, int IN3_val, int IN4_val)
{
  if (distance == 0 || distance > 10 || goFront) {
    digitalWrite(IN1, IN1_val);
    digitalWrite(IN2, IN2_val);
    digitalWrite(IN3, IN3_val);
    digitalWrite(IN4, IN4_val);

    isRunning = true;
  }
}
```

- Daca masinuta primeste comanda sa opreasca motoarele, nu se mai verifica nicio conditie si se transmite imediat semnalul.

```
void stopMotors()
{
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);
}
```

Program Flow

► **loop()** ⇒ aici se verifica daca se detecteaza vreo coliziune si se trateaza semnalele de la telecomanda.

- Verificam daca senzorul ultrasonic detecteaza vreun obiect, caz in care oprim motoarele si semnalam ca masinuta s-a oprit;
- Daca masinuta nu este in miscare sau merge cu fata, nu vom opri niciodata motoarele.

```
if (detectCollision()) {
    stopMotors();
    isRunning = false;
}
```

- Rutina de detectare si tratare a semnalelor de la telecomanda;
- Butoanele controleaza diferit motoarele masinutei, in functie de directia de mers pe care ne-o dorim

sau daca vrem sa ne oprim;
- Modificam variabila conditie daca masinuta porneste cu fata sau nu.

```
if (IrReceiver.decode()) {
    switch (IrReceiver.decodedIRData.decodedRawData) {
      case 3810328320:
        // Backwards ==> 'OK' BUTTON
        controlMotors(HIGH, LOW, LOW, HIGH);
        goFront = false;
        break;

      case 3877175040:
        // Go in front ==> '^' BUTTON
        controlMotors(LOW, HIGH, HIGH, LOW);
        goFront = true;
        break;

      case 4144561920:
        // Left ==> '<' BUTTON
        controlMotors(HIGH, LOW, HIGH, LOW);
        goFront = false;
        break;

      case 2774204160:
        // Right ==> '>' BUTTON
        controlMotors(LOW, HIGH, LOW, HIGH);
        goFront = false;
        break;

      case 3860463360:
        // STOP ==> '0' BUTTON
        stopMotors();
        goFront = false;
        isRunning = false;
    }
    IrReceiver.resume(); // Enable receiving of the next value
  }
```

Medii de dezvoltare

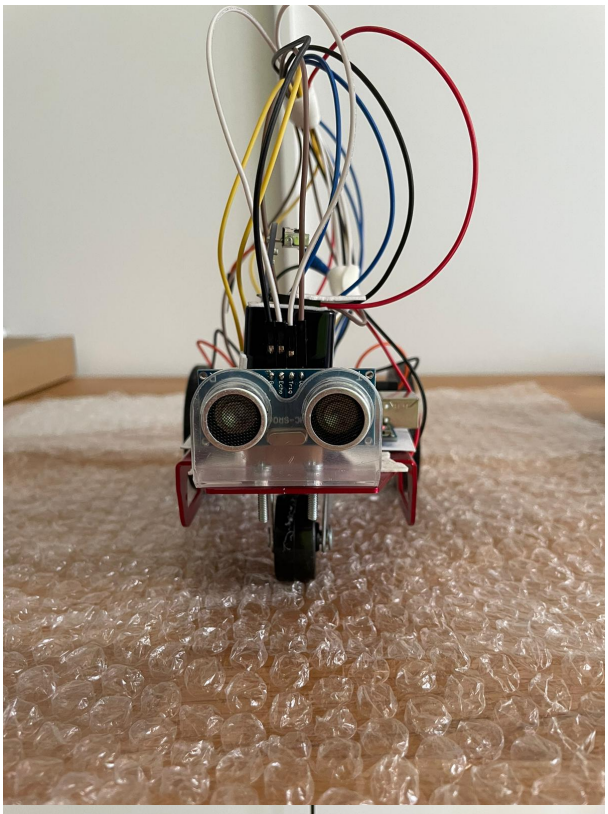
- Cod: ArduinoIDE
- Schema bloc: drawIO
- Schema electrica: EAGLE CAD

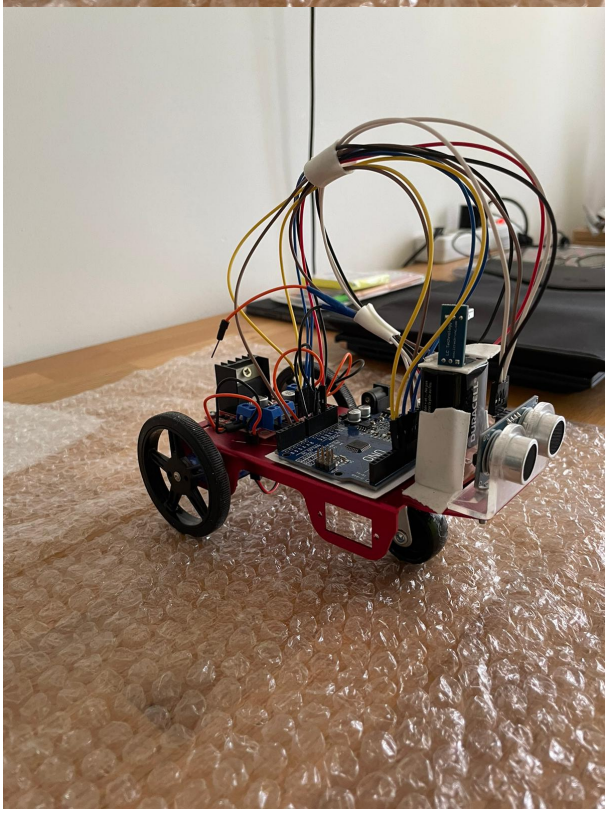
Biblioteci

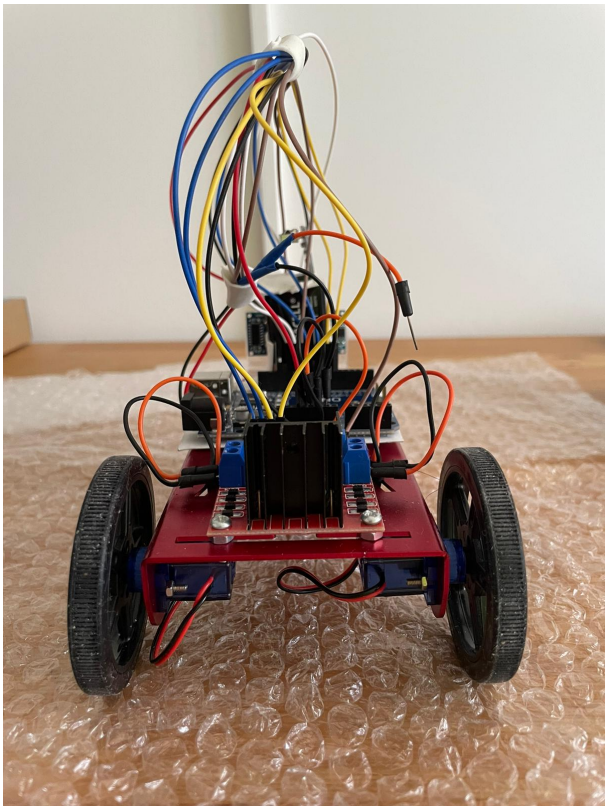
- *IRremote.hpp* ⇒ utilizarea senzorului cu infrarosu IRVS1838B pentru controlarea masinutei

- *NewPing.h* ⇒ utilizarea senzorului ultrasonic HC-SR04 pentru detectarea coliziunilor

Rezultate Obţinute







Concluzii

Mi-a placut foarte mult sa lucrez la acest proiect, deoarece am avut sansa sa aprofundez cunostintele acumulate la curs si la laborator pentru a realiza un prototip interesant si foarte util. Sunt constient ca functionalitatile sale nu sunt extrem de avansate, dar asa cum a fost gandit proiectul, isi indeplineste intocmai atributiile.

Am intampinat cateva probleme pe parcursul implementarii, cum ar fi: lipsa de spatiu pe sasiul masinutei si spatii foarte mici de prins suruburile, prea putini pini de alimentare si de GND pe placuta Arduino pentru componentele pe care le aveam de controlat sau roata ajutatoare (cea de a 3-a) care a trebuit lipita pentru ca se rotea in jurul axului (asa a fost gandita de producator), iar masinuta nu mergea drept.

Pe viitor, implementarea se poate extinde adaugand eventual un alt senzor si pe fata sau prin incarcarea unor noi tipuri de miscari controlate prin telecomanda.

Download

Sursele proiectului pot fi vizualizate aici: proiect.zip

Jurnal

- 10.05.2022 → Alegere tema proiect
- 15.05.2022 → Completare Milestone 1: Introducere, Descriere, Schema Bloc și Componente
- 29.05.2022 → Completare Milestone 2: Schema electrică, Software Design, Rezultatele Obținute, Concluzii, Arhivă

Bibliografie/Resurse

Resurse Hardware

- https://www.youtube.com/watch?v=GPVC84D5ULw&ab_channel=MRMS-WORKSHOP
- https://www.youtube.com/watch?v=E2sTbpFsvXI&ab_channel=RyanChan
- http://www.handsontec.com/dataspecs/L298N%20Motor%20Driver.pdf

Resurse Software

- http://www.handsontec.com/dataspecs/L298N%20Motor%20Driver.pdf
- https://bitbucket.org/teckel12/arduino-new-ping/wiki/Home
- https://github.com/Arduino-IRremote/Arduino-IRremote

Export to PDF