DOCUMENTAȚIE ROBOT ARDUINO

Profesor coordonator: Trăsnea Bogdan

Nume: Cosferenț Roxana-Adelina

Programul de studii: Automatică și informatică aplicată

Grupa: 4LF412

An: 2022-

OBIECTIVUL PROIECTULUI

Proiectul constă în programarea si asamblarea unui robot care să ocolească obstacolele.

COMPONENTELE UTILIZATE:

- Placa de extensie V5
- Roata universala
- Senzor ultrasonic
- Caroserie acrilica
- Motor servo (SG90)
- Placa de conducere a motorului L298N
- Fir Dupont cu 20 de pini
- Cutie pentru baterii
- Anvelope
- Motor DC
- Modulul receptor IR
- Sudor ultrasonic
- Kit de suruburi
- Acumulatoare 18650 3.7V

COMPONENTELE DE BAZĂ ALE ROBOTULUI:

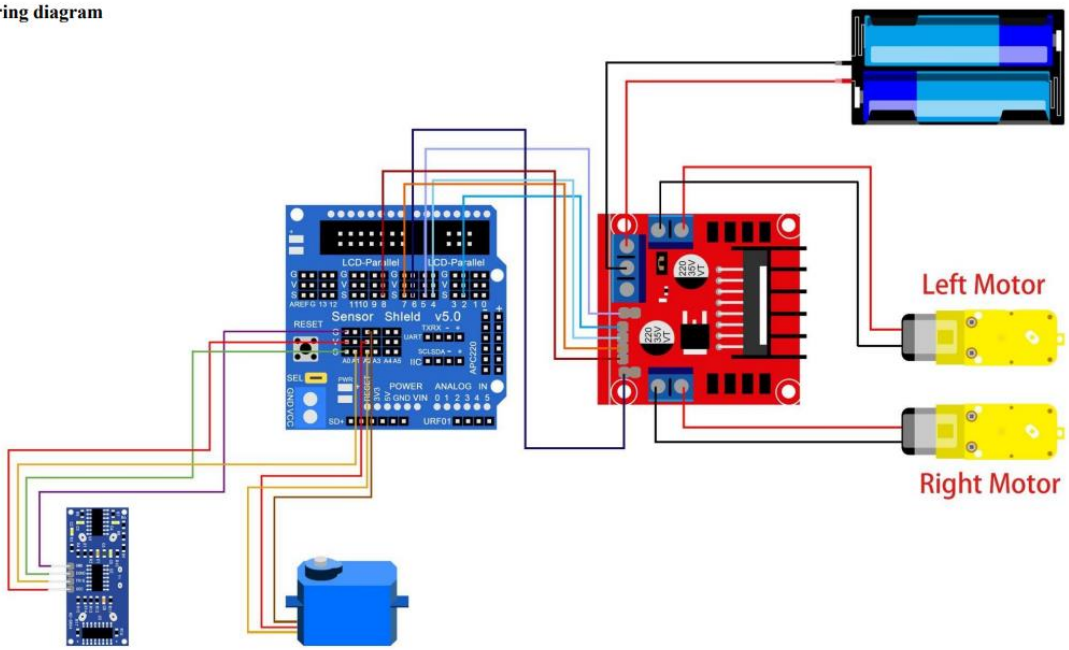
ASAMBLAREA ROBOTULUI:

Pentru asamblarea robotului, am folosit drept referință tutorialul venit cu robotul si totodată si tutoriale de pe youtube, am verificat pe rând fiecare piesă pentru a mă asigura ca functionează.

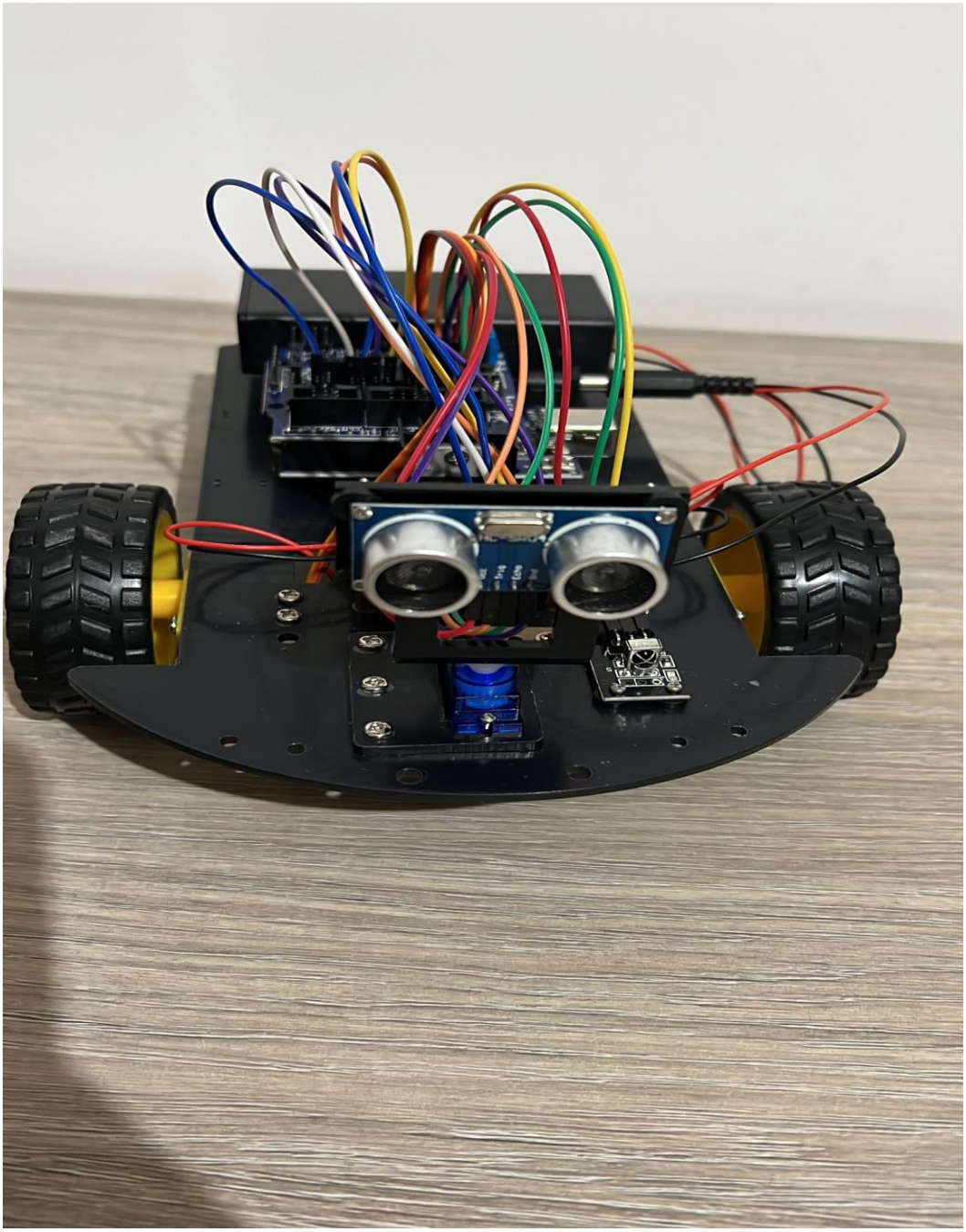
Schema inclusă in kit-ul arduino, a fost folositoare la montarea si conectarea componentelor responsabile de controlul robotului.

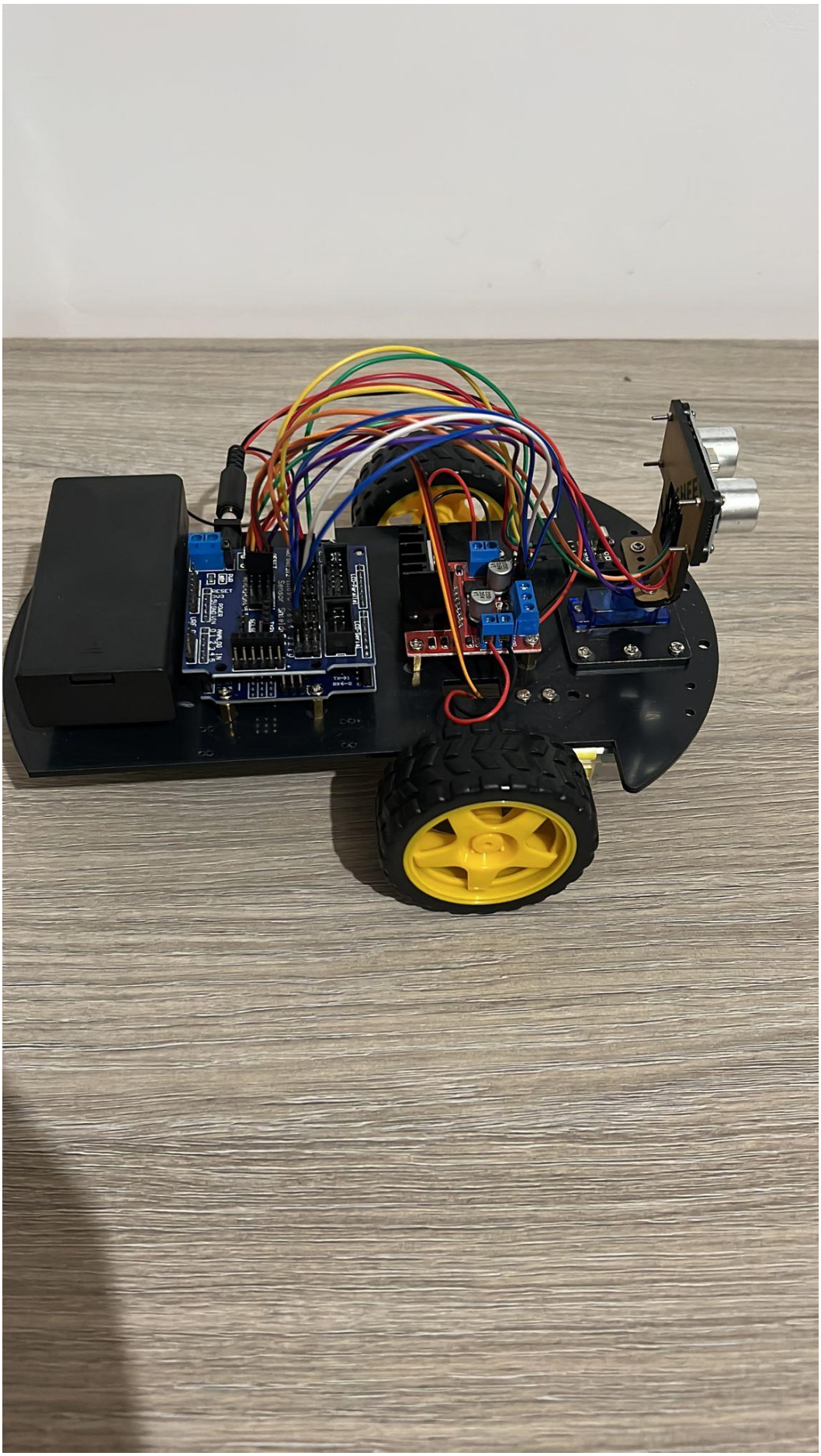
Schema prezentată în kit:

Wiring diagram



CÂTEVA POZE DUPĂ ASAMBLARE:





CODUL:

Acest cod este pentru un robot Arduino care folosește senzori ultrasonice și un motor servo pentru a evita obstacolele. Robotul are două roți, una stânga și una dreapta și este controlat de placa de conducere a motorului L298N.

Funcția principală pentru evitarea obstacolelor este funcția Ultrasonic_obstacle_avoidance(). În această funcție, robotul verifică întâi distanța din fața sa folosind funcția checkdistance(). Dacă distanța este mai mică de 20cm și mai mare de 0, se oprește, se întoarce motorul servo spre stânga și verifică distanța pe partea stângă, apoi se întoarce motorul servo spre dreapta și verifică distanța pe partea dreaptă. În funcție de distanță, alege partea cu mai mult spațiu și se întoarce în acea direcție. Dacă nu există obstacol în fața sa, continuă să se deplaseze înainte.

-inițializarea si definirea pinilor

```
#include <Servo.h>
#include <Servo.h>
Servo myservo;
int Echo_Pin=A0;
int Trig_Pin=A1;
#define Lpwm_pin  5
#define Rpwm_pin  6
int pinLB=2;
int pinLF=4;
int pinRB=7;
int pinRF=8;
volatile int D_mix;
volatile int D_mid;
volatile int D_max;
volatile int Front_Distance;
volatile int Left_Distance;
volatile int Right_Distance;
volatile int Right_IR_Value;
volatile int Left_IR_Value;
```

-verificarea distantei

```
float checkdistance() {
  digitalWrite(Trig_Pin, LOW);
  delayMicroseconds(2);
  digitalWrite(Trig_Pin, HIGH);
  delayMicroseconds(10);
  digitalWrite(Trig_Pin, LOW);
  float distance = pulseIn(Echo_Pin, HIGH) / 58.00;
  delay(10);
  return distance;
}

void Detect_Left_and_Right__distance() {
  myservo.write(180);
  delay(400);
  Left_Distance = checkdistance();
  delay(600);
  Serial.print("Left_Distance:");
  Serial.println(Left_Distance);
  myservo.write(0);
  delay(400);
  Right_Distance = checkdistance();
  delay(600);
  Serial.print("Right_Distance:");
  Serial.println(Right_Distance);
  myservo.write(90);
}

void Ultrasonic_obstacle_avoidance()
{
  Front_Distance=checkdistance();
  if((Front_Distance < 20)&&(Front_Distance > 0))
{
    stopp();//stop
    delay(100);
    myservo.write(180);
    delay(500);
    Left_Distance=checkdistance();
    delay(100);
    myservo.write(0);
    delay(500);
    Right_Distance=checkdistance();
    delay(100);
if(Left_Distance > Right_Distance)
    {
      rotate_left(180);
      myservo.write(90);
```

```
      delay(300);
    }
    else
    {
      rotate_right(180);
      myservo.write(90);
      delay(300);
    }
  }
  else
  {
    go_forward(100);
  }
}
```

```
void Obstacle_Avoidance_Main()
{
  Ultrasonic_obstacle_avoidance();
}
```

```
void setup(){
  myservo.attach(A2);
  Serial.begin(9600);
  D_mix = 10;
  D_mid = 20;
  D_max = 100;
  Front_Distance = 0;
  Left_Distance = 0;
  Right_Distance = 0;
  myservo.write(90);
  pinMode(Echo_Pin, INPUT);
  pinMode(Trig_Pin, OUTPUT);
  pinMode(pinLB,OUTPUT); // /pin 2
  pinMode(pinLF,OUTPUT); // pin 4
  pinMode(pinRB,OUTPUT); // pin 7
  pinMode(pinRF,OUTPUT);  // pin 8
  pinMode(Lpwm_pin,OUTPUT);  // pin 5 (PWM)
  pinMode(Rpwm_pin,OUTPUT);  // pin 6(PWM)
}

void loop(){
  Obstacle_Avoidance_Main();
```

```
}


void go_forward(unsigned char speed_val)
    {digitalWrite(pinRB,LOW);
     digitalWrite(pinRF,HIGH);
     digitalWrite(pinLB,LOW);
     digitalWrite(pinLF,HIGH);
     analogWrite(Lpwm_pin,speed_val);
     analogWrite(Rpwm_pin,speed_val);


    }

void go_backward(unsigned char speed_val)
    {
     digitalWrite(pinRB,LOW);
     digitalWrite(pinRF,HIGH);
     digitalWrite(pinLB,LOW);
     digitalWrite(pinLF,HIGH);
     analogWrite(Lpwm_pin,speed_val);
     analogWrite(Rpwm_pin,speed_val);
    }

void rotate_left(unsigned char speed_val)
    {digitalWrite(pinRB,HIGH);
     digitalWrite(pinRF,LOW );
     digitalWrite(pinLB,LOW);
     digitalWrite(pinLF,HIGH);
     analogWrite(Lpwm_pin,speed_val);
     analogWrite(Rpwm_pin,speed_val);


    }
void rotate_right(unsigned char speed_val)
    {
      digitalWrite(pinRB,LOW);
     digitalWrite(pinRF,HIGH);
     digitalWrite(pinLB,HIGH);
     digitalWrite(pinLF,LOW);
     analogWrite(Lpwm_pin,speed_val);
     analogWrite(Rpwm_pin,speed_val);

    }
void stopp()        //stop
    {
     digitalWrite(pinRB,HIGH);
```

```
 digitalWrite(pinRF,HIGH);
 digitalWrite(pinLB,HIGH);
 digitalWrite(pinLF,HIGH);
}
```

Funcția setup() este locul unde are loc configurarea inițială pentru program. Ea configură comunicarea serială, modurile pin-urilor pentru diferitele componente și inițializează motorul servo la o poziție neutră.

Funcția loop() este locul unde este executat codul principal.