Universitatea Tehnică din Cluj-Napoca

Proiectare cu microprocesoare

Mașină telecomandată prin semnal Bluetooth cu senzor anti-coliziune

---

Iliești Robert-Mircea

Cuprins

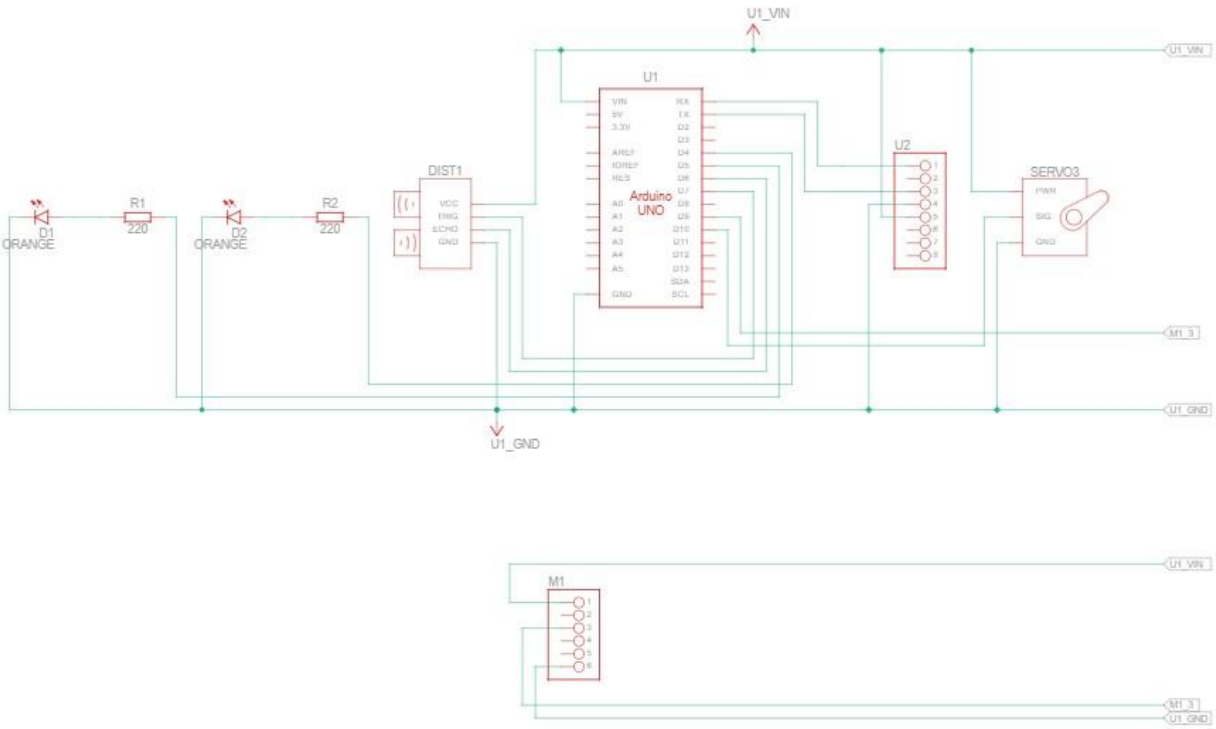
1. Introducere

Acest proiect controlează o mașină telecomandată prin Bluetooth folosind un modul HC-05. Mașina include un senzor ultrasonic HC-SR04 pentru a evita coliziunile, iar comenzile sunt primite și procesate pentru a gestiona mișcarea servomotoarelor de direcție și mișcare. LED-uri suplimentare sunt utilizate pentru semnalizare.

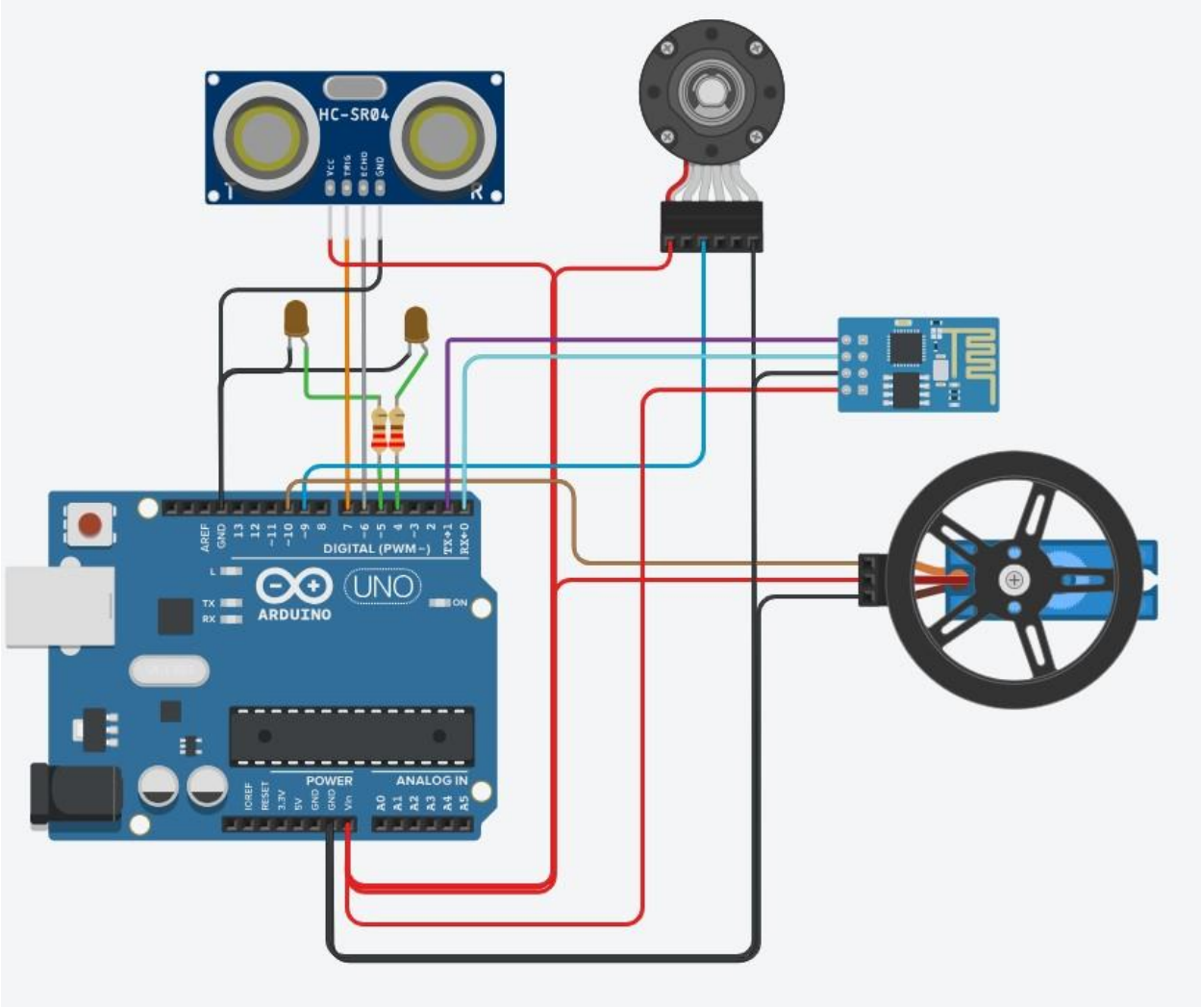
2. Componente utilizate

a. Microcontroller Arduino Jade M1
b. Servo 1: controlul mișcării (accelerației) mașini
c. Servo 2: controlul direcției mașini
d. HC-SR04: sensor ultrasonic pentru detectarea distanței
e. HC-05: modul Bluetooth
f. LED-uri: Indicatoare conectate pe pinii A4 și A5
g. Alte componente: fire de conectare, baterie, șasiu pentru mașină

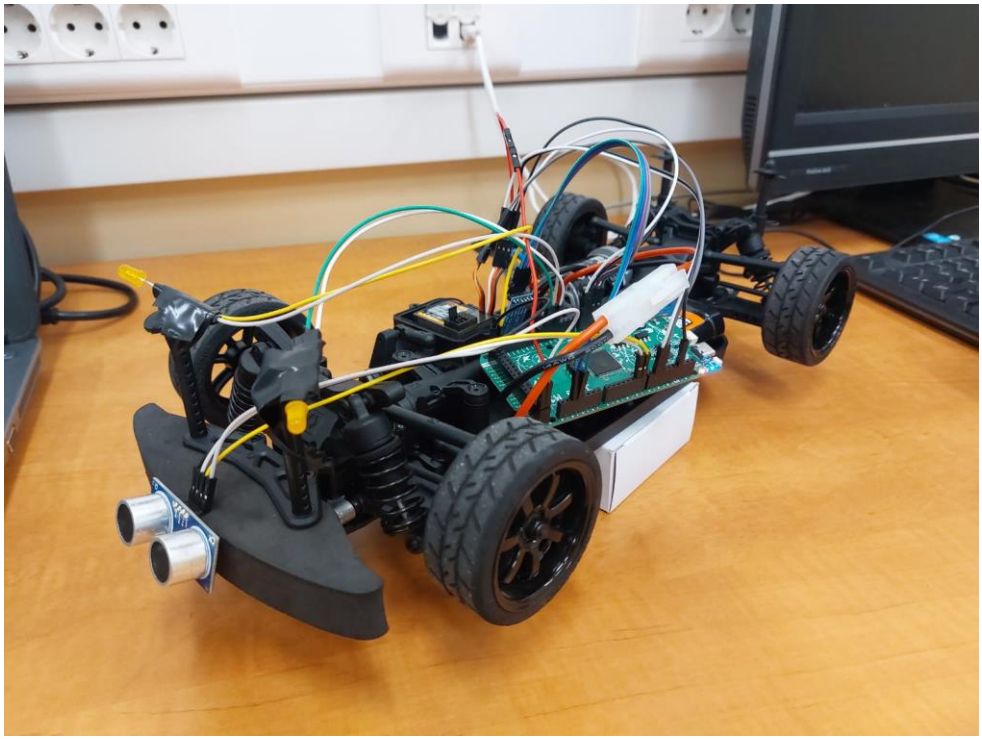
3. Schema electrică

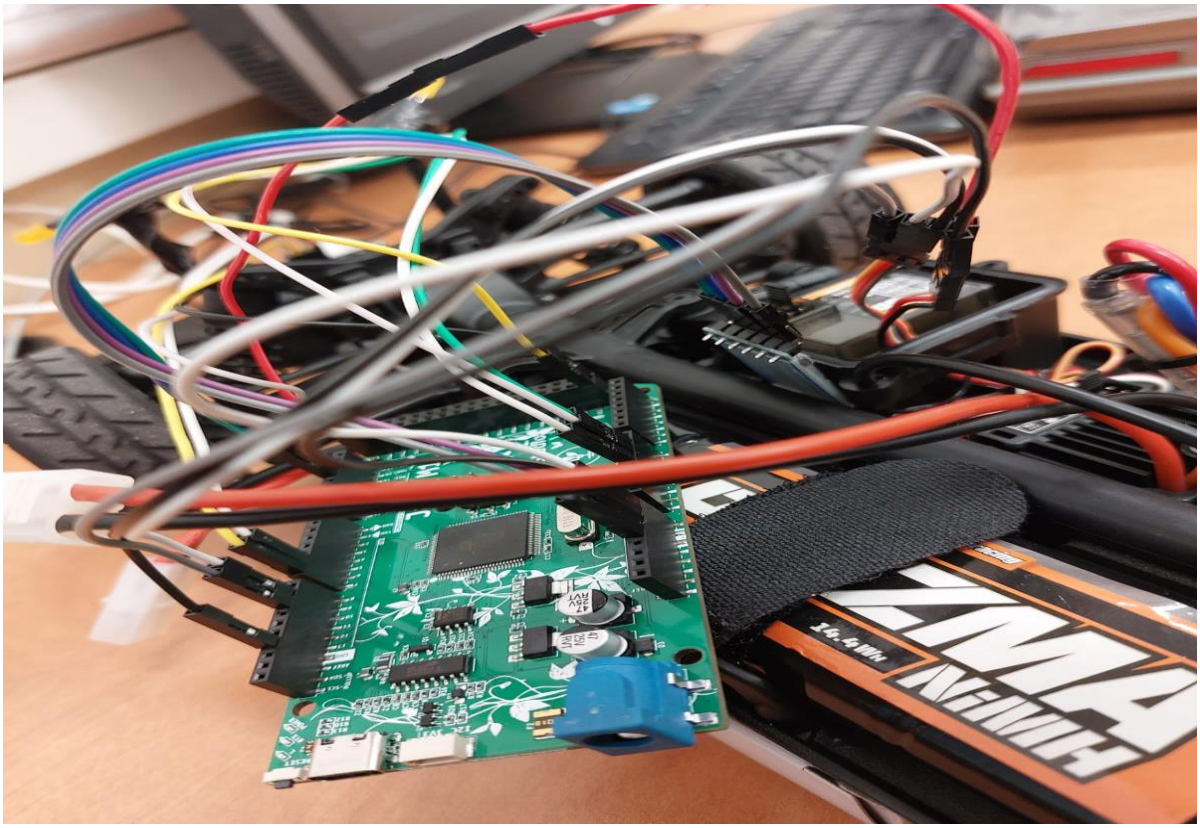


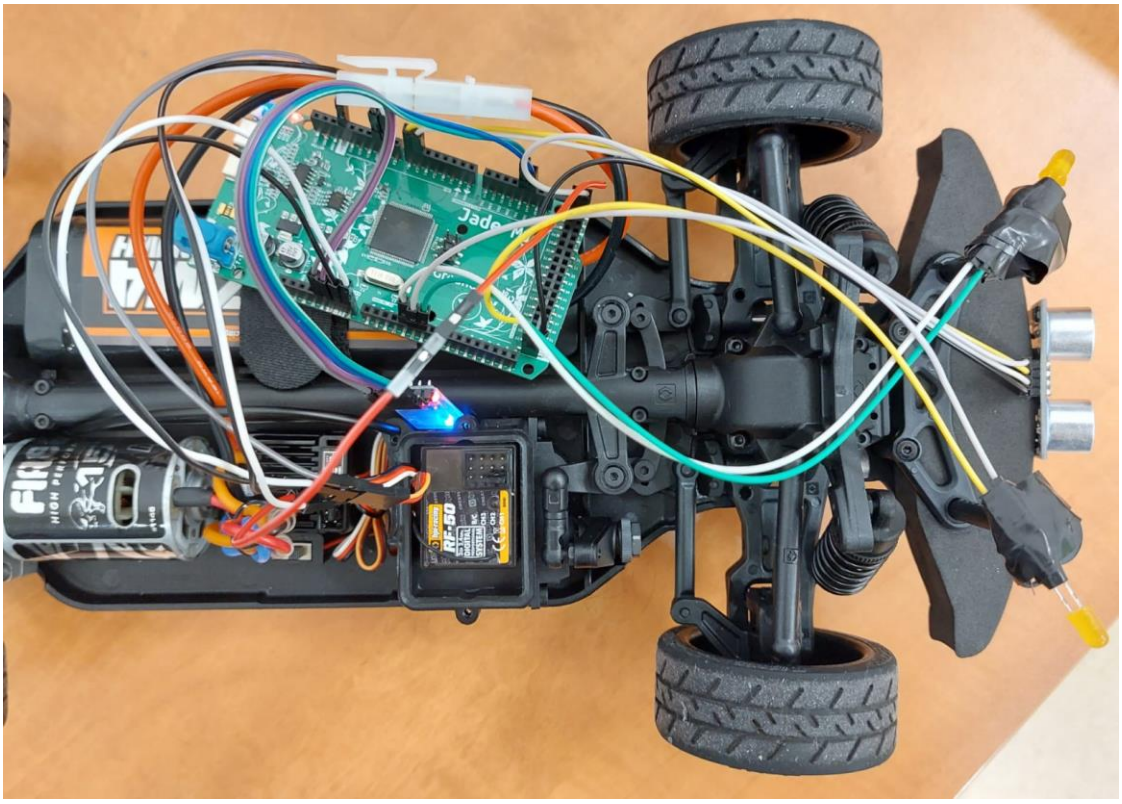
4. Schema de montaj

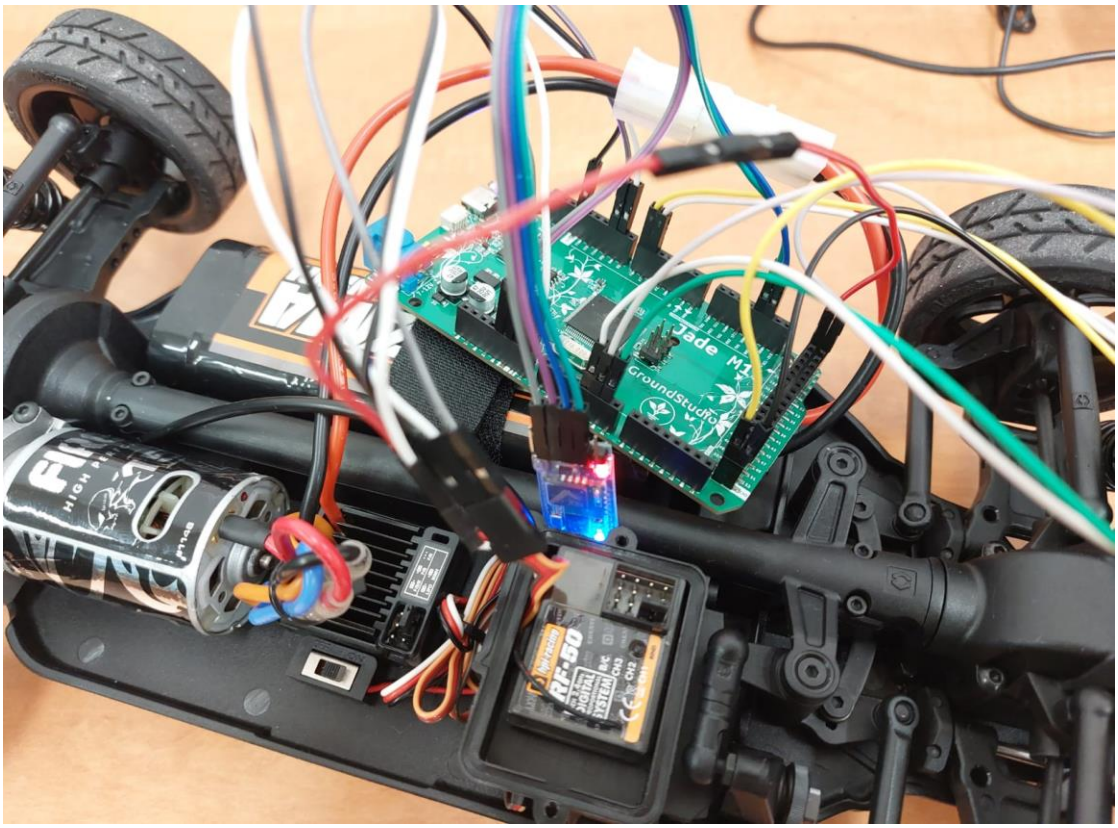


Universitatea Tehnică din Cluj-Napoca






RF-50


Jade M1
GroundStudio
RF-50

5. Funcții principale

Configurare hardware:

- Pini și comunicare:
  - TRIG_PIN (pin 7): Trimiterea semnalului de declanșare pentru senzorul ultrasonic.
  - ECHO_PIN (pin 6): Recepționarea semnalului de la senzorul ultrasonic.
  - A4 și A5: LED-uri pentru semnalizare.
  - Servo 1 (pin 9) și Servo 2 (pin 10): Controlul mișcării și direcției.
- Bluetooth: Comunicarea este realizată prin portul Serial1 la 9600 baud rate.

Funcții implementate

Controlul mașinii:

- idle(): Oprește mișcarea mașinii și centrează direcția.
- forward(), backward(): Mișcarea înainte și înapoi folosind motorul electric.
- forwardLeft(), forwardRight(), backwardLeft(), backwardRight(): Mișcarea într-o anumită direcție combinând rotațirea roților și direcția de deplasare.
- left(), right(): Schimbarea direcției la stânga sau la dreapta.

Interacțiunea cu senzorul:

- measureDistance():
  - Trimte un impuls ultrasonic folosind TRIG_PIN.
  - Măsoară timpul în care semnalul revine folosind ECHO_PIN.
  - Calculează distanța bazată pe timp (viteză sunetului).

Comunicarea prin Bluetooth:

- processCommands():
    - Primește comenzi în format text de la modulul HC-05.
    - Comanda se încheie cu caracterul “\n”.
    - Comenzile valide sunt interpretate și salvate în variabila currentCommand.
- executeCurrentCommand():
    - Verifică și execută comenzile stocate în currentCommand.

Semnalizare cu LED-uri:

- blink(int led): Blink rapid pentru unul dintre LED-uri (de 5 ori cu pauze de 500ms).

Lista comenzilor Bluetooth:

1. F: Înainte
2. B: Înapoi
3. L: Stânga
4. R: Dreapta
5. FL, LF: Înainte și la stânga
6. FR, RF: Înainte și la dreapta
7. BL, LB: Înapoi și la stânga
8. BR, RB: Înapoi și la dreapta
9. S: Oprire
10. Q: Blink pe LED-ul A4
11. E: Blink pe LED-ul A5

6. Codul sursă

```
#include <Servo.h>

Servo motion;     // Motion servo
Servo directie;    // Direction servo

String command = "";         // Buffer for reading full commands
String currentCommand = "";  // Currently active command
bool stop = false;           // Stop flag

// Constants for HC-SR04
const int TRIG_PIN = 7;  // Ultrasonic sensor TRIG pin
const int ECHO_PIN = 6;  // Ultrasonic sensor ECHO pin

// LEDs
const int ledA4 = A4;
const int ledA5 = A5;
const int ledA6 = A6;

// Variables for distance measurement
float distance = 0;

void setup() {
  // Initialize Serial Communication for Debugging
  Serial.begin(9600);

  // Initialize Bluetooth Communication (Serial1 for HC-05 on Mega)
  Serial1.begin(9600);

  // Configure ultrasonic sensor pins
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);

  // Configure LEDs
  pinMode(ledA4, OUTPUT);
  pinMode(ledA5, OUTPUT);
  pinMode(ledA6, OUTPUT);
  digitalWrite(ledA4, LOW);
  digitalWrite(ledA5, LOW);
  digitalWrite(ledA6, LOW);

  // Attach servos
  motion.attach(9);
  directie.attach(10);

  // Set initial servo positions
  motion.writeMicroseconds(1500);  // Neutral position (stop)
  directie.write(90);              // Center the direction servo
  delay(7000);                     // Wait for initialization
}

void loop() {
  // Measure distance continuously
  distance = measureDistance();

  // Update the stop condition based on distance threshold
  stop = (distance <= 20.0);

  // If a stop condition is met, halt all motion
  if (stop || currentCommand == "S") {
    idle();
    currentCommand = "";  // Clear the active command when stopping
  } else {
    // Execute the current active command if no stop condition
    executeCurrentCommand();
  }

  // Process new commands from Bluetooth
  processCommands();
}

void processCommands() {
  while (Serial1.available() > 0) {
    char incomingChar = Serial1.read(); // Read one character from Bluetooth

    if (incomingChar == '\n') {        // Command delimiter
      command.trim();                  // Remove extra spaces
      if (command == "S") {            // Stop command
        currentCommand = "S";
      } else {
        currentCommand = command;      // Update the current command
      }
      command = "";                    // Clear the command buffer
    } else {
      command += incomingChar;         // Append character to buffer
    }
  }
}
```

```
void executeCurrentCommand() {
  if (currentCommand == "FL" || currentCommand == "LF") {
    forwardLeft();
  } else if (currentCommand == "FR" || currentCommand == "RF") {
    forwardRight();
  } else if (currentCommand == "BL" || currentCommand == "LB") {
    backwardLeft();
  } else if (currentCommand == "BR" || currentCommand == "RB") {
    backwardRight();
  } else if (currentCommand == "F") {
    forward();
  } else if (currentCommand == "B") {
    backward();
  } else if (currentCommand == "L") {
    left();
  } else if (currentCommand == "R") {
    right();
  } else if (currentCommand == "Q") {
    blink(ledA4);
  } else if (currentCommand == "E") {
    blink(ledA5);
  }
}

float measureDistance() {
  long duration;
  float distance;

  // Send a 10-microsecond pulse to the TRIG pin
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  // Read the ECHO pin to calculate the duration
  duration = pulseIn(ECHO_PIN, HIGH);

  // Calculate the distance in cm
  distance = (duration * 0.034) / 2;
  return distance;
}

void idle() {
  motion.writeMicroseconds(1500);  // Neutral position (stop)
  directie.write(90);              // Center the direction servo
}

void forwardLeft() {
  motion.writeMicroseconds(1600);  // Forward motion
  directie.write(120);             // Steer left
}

void forwardRight() {
  motion.writeMicroseconds(1600);  // Forward motion
  directie.write(60);              // Steer right
}

void backwardLeft() {
  motion.writeMicroseconds(1400);  // Backward motion
  directie.write(120);             // Steer left
}

 void backwardRight() {
   motion.writeMicroseconds(1400);  // Backward motion
   directie.write(60);              // Steer right
 }

 void forward() {
   motion.writeMicroseconds(1600);  // Forward motion
   directie.write(90);              // Center direction
 }

 void backward() {
   motion.writeMicroseconds(1400);  // Backward motion
   directie.write(90);              // Center direction
 }

 void left() {
   directie.write(120);             // Steer left
 }

 void right() {
   directie.write(60);              // Steer right
 }

 void blink(int led) {
   for (int i = 0; i < 5; i++) {
     digitalWrite(led, HIGH);  // Turn the LED on
     delay(500);               // Wait for half a second
     digitalWrite(led, LOW);   // Turn the LED off
     delay(500);               // Wait for half a second
   }
 }
```

7. Posibile extinderi

1. Evaziune automata a obstacolelor: Implementația unor funcții suplimentare pentru ocolirea automata.
2. Feedback către utilizator: Trimiterea distanței măsurate prin Bluetooth.
3. Control prin aplicație: Dezvoltarea unei aplicații dedicate pentru control prin Bluetooth.

8. Bibliografie

1. https://users.utcluj.ro/~razvanitu/teaching.htm (Cursuri)
2. https://biblioteca.utcluj.ro/files/carti-online-cu-coperta/336-3.pdf
3. https://www.arduino.cc/en/Guide
4. https://www.tinkercad.com