

# Masinuta cautatoare de comori

## Introducere

Proiectul “Mașinută Căutătoare de Comori” reprezintă o extensie a unei platforme robotice existente cu funcționalități avansate de detectare și interacțiune cu mediul înconjurător. Scopul principal este transformarea unui kit preconcept într-un sistem inteligent personalizat care poate identifica și colecta obiecte bazate pe culoare.

Ideea de la care am pornit a fost de a îmbunătăți o mașinută cu funcția predefinită de urmărire linie și de a adăuga un element de joc interactiv care să o transforme într-un sistem personalizat și mai complex.

Acest proiect este util atât pentru dezvoltarea personală, oferindu-mi posibilitatea de a aprofunda cunoștințele despre diferite protocoale de comunicație (GPIO, UART, I2C) și integrarea senzorilor, cât și pentru comunitate, demonstrând cum se poate transforma un kit comercial într-un sistem personalizat cu funcționalități extinse.

## Descriere generală

### Schema bloc:



### Protocoale de Comunicație

- GPIO: Control motoare, LED-uri, speaker
- I2C: Comunicare cu senzorul de culoare TCS34725 și ecranul OLED, precum și relație master-slave între Marble Pico, respectiv Arduino Uno
- UART: Comunicare cu DFPlayer Mini și modulul Bluetooth
- PWM: Control viteză motoare

### Modulele Sistemului

#### Modulul de Control:

- Microcontroller Arduino care gestionează mișcările mașinii, conexiunea Bluetooth cu aplicația mobilă

- Microcontroller Marble Pico pentru gestionarea Arduino (prin master-slave) la gasirea unei "comori" si al restului componentelor

### **Modulul de Senzori**

- Senzor culoare pentru identificarea "comorilor"
- Senzori IR pentru urmărirea liniei (hartii)

### **Modulul de Acționare**

- Motoare DC plus LED-uri pentru semnalizare vizuală.
- Modul convertor de tensiune de la 5V la 3.3V
- Ecran OLED și DFPlayer Mini pentru afișarea parametrilor și notificări audio.

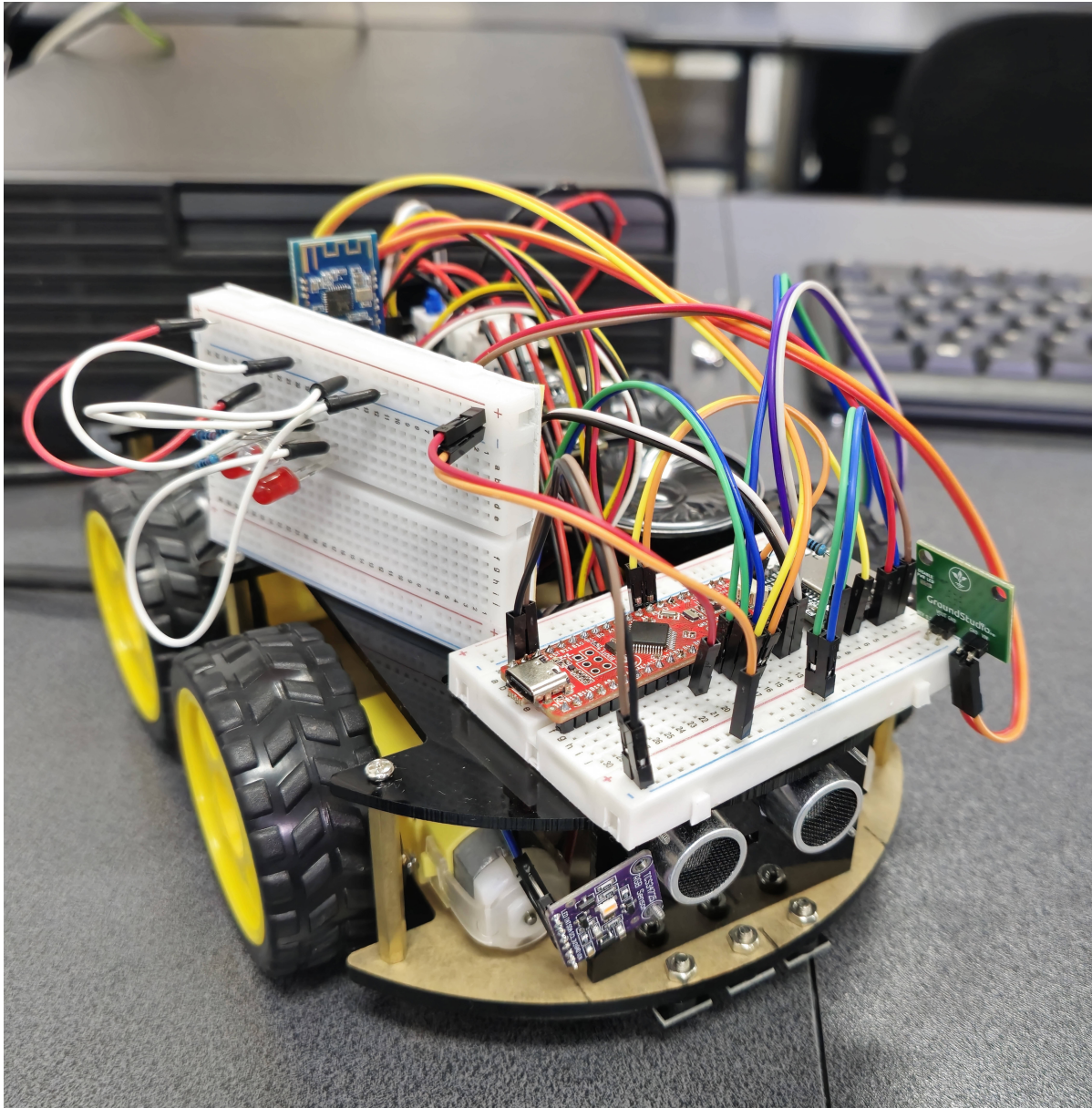
### **Funcționare**

Folosind o aplicatie mobila, mașinuta poate fi controlată manual, insa poate intra in modul Hunting in care fie merge inainte in continuu, fie urmareste un traseu cu linie neagra, detectând și alarmând utilizatorul in cazul găsirii unor obiecte de culori specifice ("comori", eu am ales de culoare neagra). Sistemul oferă feedback vizual și auditiv intr-un mod pe care nu il voi dezvalui inca ;).

## **Hardware Design**

### **Lista preliminară de piese:**

- Kit mașinută robotică (bază, motoare, roți)
- Microcontrollere (Arduino Uno, Marble Pico)
- Senzor de culoare TCS34725
- Ecran OLED I2C
- Modul DFPlayer Mini
- Card microSD
- Speaker 2W 32 Ohm
- Senzori IR pentru urmărirea liniei
- Baterii și regulator de tensiune
- Cabluri și componente electronice auxiliare



## Software Design

**Mediu de dezvoltare:** Arduino IDE, Flutter

**Librării planificate:**

- Adafruit\_TCS34725 pentru senzorul de culoare
- Adafruit\_SSD1306 pentru ecranul OLED
- DFRobotDFPlayerMini pentru modulul audio
- Raspberry Pi Pico RP2040
- Librării predefinite in kit-ul masinii

## **Algoritmi și structuri planificate:**

### **Algoritm de detecție a culorii:**

- Citirea valorilor RGB de la senzorul TCS34725
- Compararea cu valorile predefinite pentru identificarea “comorilor”
- Declansarea comportamentului de alertare

### **Algoritm de urmărire a liniei:**

- Citirea valorilor de la senzorii IR
- Ajustarea direcției motoarelor pentru a menține traseul

### **Interfață utilizator:**

- Afișarea parametrilor și a mesajelor de feedback pe ecranul OLED
- Generarea de feedback audio pentru evenimente specifice (gasire obiect, cautare obiect etc.)

### **Mașină de stare pentru modurile de funcționare:**

- Urmărire linie (Cautare comori)
- Control manual
- Raportare statistici

### **Proces de interpretare a comenzilor primite prin bluetooth de la o aplicatie mobila:**

- Dezvoltarea unei aplicatii de Android care permite conectarea prin adresa MAC a modului și trimiterea de caractere encodeate către microcontroller
- Folosirea unei librării externe de interpretare a caracterelor encodeate venite ca input
- Executarea unei funcții (deplasare, schimbare viteză, schimbarea modului) în funcție de aceste caractere

### **Realizarea conexiunii master-slave între Marble Pico, respectiv Arduino:**

- Trimiterea de comenzi din Marble Pico la adresa Arduino prin I2C de executare a funcțiilor din codul slave

## **Rezultate Obținute**

Am implementat aproape în totalitate tot ce am menționat la Software Design.

Reusesc prin intermediul aplicatiei mobile să mă conectez la microcontroller, iar mașinuta răspunde

pozitiv la comenzile pe care i le dau prin intermediul acesteia (se deplaseaza, isi schimba viteza, poate intra in modul hunting)

Ecranul OLED functioneaza bine, afiseaza informatiile care trebuie, iar led-urile raspund la schimbarile de stare oferite din codul Marble Pico. Senzorul de culoare are downside-ul de a lumina slabut, iar pentru a detecta un obiect de culoarea care trebuie, masinuta, mai precis led-ul senzorului, trebuie sa fie foarte aproape de acesta. Cu toate acestea, am reusit sa il calibrez in asa fel incat sa detecteze cum trebuie rosu si sa genereze comportamentul de alerta al masinii.

Cea mai mare problema pe care am intampinat-o si pe care nu am mai avut timp sa o rezolv este redarea sunetului. Am folosit initial un dfplayer cumparat dintr-o parte, am conectat pinii unde trebuie, iar la diagnosticare primeam mereu eroare de citire a cardului sd. dupa asta, am folosit un dfplayer cumparat din alta parte si a functionat cu fix aceeasi configuratie (am dovada video). pe masura ce am mai adaugat functii inasa, dfplayer-ul s-a oprit din functionare si nu am reusit sa imi dau seama care e problema. Am inceput sa primesc aceleasi mesaje de eroare ca la primul dfplayer. Am comandat unul nou prea tarziu si nu am mai avut cand sa il testez (nu a ajuns pana la ziua de prezentare).

## Download

### Cod pentru Arduino Uno

<https://drive.google.com/file/d/1uDA5Elq3zFg71mTdAhzrtgXJGqO79v4O/view?usp=sharing>

### Cod pentru Marble Pico

<https://drive.google.com/file/d/1J-Zo0cVsG1i93x4mLIZHQvctaU53GIMD/view?usp=sharing>

### README

<https://drive.google.com/file/d/1DFzQl-jZ2bUoNoqddRgVWwXMsvUDopd5/view?usp=sharing>

### Cod aplicatie mobila

<https://github.com/lulySquadyt/flutter-car-remote-app.git>

## Bibliografie/Resurse

### Resurse Hardware

- Datasheet Marble Pico:

[https://github.com/GroundStudio/GroundStudio\\_Marble\\_Pico/blob/main/Documentation/REV0.0.3/Data sheet%20GroundStudio%20Marble%20Pico%20REV%201.1.pdf](https://github.com/GroundStudio/GroundStudio_Marble_Pico/blob/main/Documentation/REV0.0.3/Data%20sheet%20GroundStudio%20Marble%20Pico%20REV%201.1.pdf)

- Datasheet TCS34725: <https://cdn-shop.adafruit.com/datasheets/TCS34725.pdf>
- Datasheet DFPlayer Mini: <https://picaxe.com/docs/spe033.pdf>
- Datasheet HC-05:  
[https://components101.com/sites/default/files/component\\_datasheet/HC-05%20Datasheet.pdf](https://components101.com/sites/default/files/component_datasheet/HC-05%20Datasheet.pdf)
- Datasheet KY-033: <https://download.kamami.pl/p587610-KY-033-Joy-IT.pdf>

## Resurse Software

- I2C Bus Specification: <https://www.nxp.com/docs/en/user-guide/UM10204.pdf>
- Arduino I2C Slave Implementation Guide: <https://deepbluembedded.com/arduino-i2c-slave/>
- Adafruit\_SSD1306 Library Documentation:  
[https://adafruit.github.io/Adafruit\\_SSD1306/html/class\\_adafruit\\_\\_\\_s\\_s\\_d1306.html](https://adafruit.github.io/Adafruit_SSD1306/html/class_adafruit___s_s_d1306.html)
- Adafruit\_TCS34725 Library Documentation:  
[http://adafruit.github.io/Adafruit\\_TCS34725/html/class\\_adafruit\\_\\_\\_t\\_c\\_s34725.html](http://adafruit.github.io/Adafruit_TCS34725/html/class_adafruit___t_c_s34725.html)
- SoftwareSerial Library Documentation: <https://docs.arduino.cc/learn/built-in-libraries/software-serial/>
- Wire Library Documentation: <https://www.arduino.cc/en/Reference/wire>

[Export to PDF](#)

From:  
<http://ocw.cs.pub.ro/courses/> - **CS Open CourseWare**

Permanent link:  
<http://ocw.cs.pub.ro/courses/pm/prj2025/rnedelcu/iulian.popa3007>

Last update: **2025/05/28 00:07**

