Touring Car

Introducere

- Proiectul constă în construirea unei mașini de tip rover 4×4, care poate fi controlată de la distanță printr-o aplicație pe telefon și permite manevrarea în toate cele 4 direcții. Mașina prezintă un mecanism inteligent de detecție a obstacolelor frontale: în cazul în care mașina s-ar lovi în partea din față de un obiect, aceasta blochează orice încercare a utilizatorului de a se mișca drept înainte, forțându-l să schimbe direcția de mers, pentru a evita impactul.
- Scopul proiectului este de a simula un prototip de mașină cu control la distanță, care să se folosească și de un senzor, asemenea mașinilor reale moderne.
- Ideea de la care am pornit a fost să creez un proiect practic, pe care l-aș putea folosi și în afara cursului de PM, având ca inspirație mașinile cu telecomandă pentru copii.
- Aș spune despre proiect că este util pentru mine și ceilalți pentru că îmbină design-ul și folosirea mai multor electronice cu ceva distractiv, putând fi folosit drept exemplu pentru cineva care nu are experiență cu microcontrollere, ca motivație pentru a realiza ceva practic.

Descriere generală

Mașina așteaptă comenzi prin Bluetooth de pe smartphone. Odată primită o comandă, aceasta ajunge prin HC-05 la microcontroller, care o traduce în semnale către driver-ul L298N (PWM pentru viteză și GPIO pentru direcție), punând cele 4 motoare în mișcare. Pentru viraj, motoarele de pe o parte se rotesc mai repede decât cele de pe cealaltă.

Senzorul ultrasonic rulează în permanență în fundal. Dacă detectează un obstacol prea aproape în față, microcontrollerul ignoră automat orice comandă de mers înainte, obligând utilizatorul să schimbe direcția. Când drumul e liber, mașina răspunde din nou normal la toate comenzile.

Hardware Design

Listă de componente

- 1 × Microcontroller ATMega328P Xmini Xplained
- 1 × Kit șasiu 4WD (plăci bază + 4 motoare DC cu reductoare + 4 roți)
- 1 × Driver motoare L298N
- 1 × Modul Bluetooth HC-05
- 1 × Senzor ultrasonic HC-SR04
- 6 × baterii AA
- 1 × Mini-breadboard
- n × Jumpere

Schema electrică

Pinout

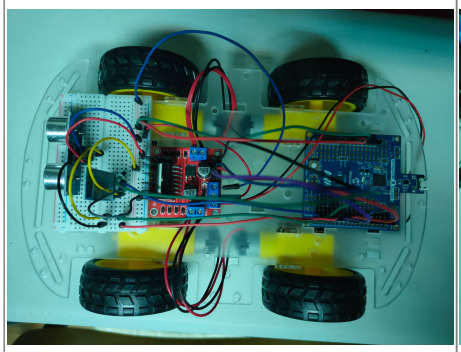
Periferic Conectat	Pin ATmega328P	Pin Periferic	Direcție Semnal
Modul Bluetooth HC-05	PD0 (RX)	TXD	Intrare
	PD1 (TX)	RXD	Ieșire
Senzor HC-SR04	PD2	Trig	Ieșire
	PD3	Echo	Intrare
Driver L298N	PD5	IN1	Ieșire
	PD6	IN2	Ieșire
	PB1	IN3	Ieșire
	PB2	IN4	Ieșire

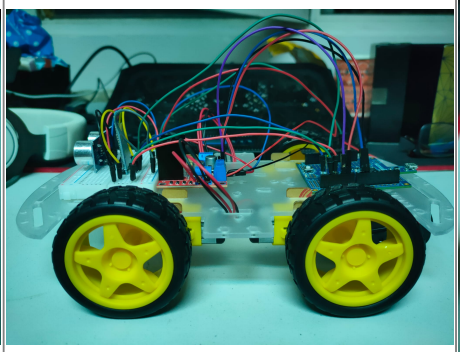
Schema electrică arată cum sunt legate toate componentele în jurul microcontrollerului ATmega328P.

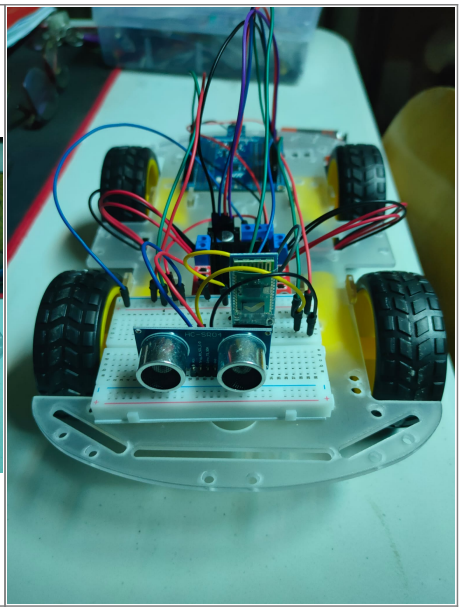
Pentru alimentare, am folosit 4 baterii AA (6V) conectate direct la pinul de 12V al driverului L298N, de unde își iau curent motoarele. Regulatorul intern de pe driver reduce această tensiune la 5V pentru a alimenta microcontrollerul, modulul Bluetooth și senzorul ultrasonic. Toate componentele din circuit împart aceeași masă comună (GND).

Pe partea de date, legăturile respectă exact tabelul de pinout. Modulul Bluetooth comunică serial prin pinii PD0 și PD1 (conectați la TXD/RXD). Senzorul de distanță folosește pinii PD2 (Trig) și PD3 (Echo) pentru a trimite și primi semnalele folosite la evitarea obstacolelor. Driverul L298N primește comenzile de direcție prin pinii IN1-IN4 (legați la pini de pe plăcuță ce permit folosirea PWM, pentru a controla manual viteza), iar ieșirile lui (OUT1-OUT4) controlează cele 4 motoare ale șasiului, fiind legate în paralel câte două pentru fiecare lateral al mașinii.

Mașina asamblată arată ca în imaginile de mai jos. Un demo în care se poate observa că funcționează toate componentele se poate găsi aici.







Software Design

Mediu de dezvoltare și biblioteci

Codul a fost scris în C++ folosind extensia PlatformIO din Visual Studio Code. Singura bibliotecă third-party folosită este **Arduino.h**, care oferă funcțiile de bază pentru interacțiunea cu pinii microcontrollerului (digitalWrite, analogWrite, pinMode, millis etc.), cât și pentru comunicarea cu modulul HC-05 prin portul UART hardware al ATmega328P.

Structura codului

Codul este organizat în jurul unui automat de stări simplu, bazat pe variabila `currentCommand`, care reține ultima comandă validă primită prin Bluetooth. În fiecare iterație a `loop()`, microcontrollerul procesează orice comandă nouă sosită, citește distanța față de obstacole și execută acțiunea corespunzătoare sau declanșează evitarea autonomă dacă e cazul.

Funcțiile principale sunt:

- `setForwardSpeed()`, `setBackwardSpeed()`, `turnLeft()`, `turnRight()`, `stopCar()` — controlează direcția și viteza motoarelor prin semnale PWM și GPIO către driverul L298N
- `readDistance()` — trimite un puls pe TRIG și măsoară durata echo-ului pe ECHO_PIN folosind `pulseIn()`, convertind rezultatul în centimetri
- `readDistanceStable()` — apelează `readDistance()` de 3 ori și returnează media citirilor valide, ignorând citirile eronate
- `handleAcceleration()` — incrementează `currentSpeed` progresiv la fiecare iterație folosind `millis()`, fără a bloca execuția cu `delay()`
- `autonomousEvasion()` — oprește mașina, merge înapoi până când distanța față de obstacol depășește `SAFE_DISTANCE` sau se atinge un timeout de siguranță, după care apelează

`scanAndAlign()`
- `scanAndAlign()` — rotește mașina spre stânga, măsoară distanța, apoi spre dreapta, măsoară din nou, și se orientează spre direcția cu mai mult spațiu disponibil
- `serialEvent()` — funcție apelată automat de framework după fiecare iterație a `loop()` când au sosit date pe UART, validează comanda primită și o pune într-un buffer volatil

Element de noutate

Elementul care diferențiază acest proiect de o mașină RC simplă este mecanismul de evitare autonomă a obstacolelor. Când mașina detectează un obstacol la mai puțin de 15cm în față în timp ce primește comanda de mers înainte, preia controlul autonom: se oprește, merge înapoi până ajunge la o distanță sigură, scanează stânga și dreapta și se orientează spre direcția cu mai mult spațiu. După finalizarea manevrei, returnează controlul utilizatorului. Un alt element notabil este accelerarea progresivă la pornire, care simulează comportamentul unei mașini reale.

Funcționalități din laborator

- **Lab 1 — UART**: comunicarea cu modulul Bluetooth HC-05 se face prin portul UART hardware al ATmega328P (PD0/PD1), la 9600 bauds
- **Lab 2 — Întreruperi**: recepția comenzilor Bluetooth este gestionată prin `serialEvent()`, care este apelată automat de framework pe baza întreruperii hardware USART_RX, imediat după fiecare iterație a `loop()`; comenzile sunt validate și puse într-un buffer volatil, decuplând recepția de execuția propriu-zisă
- **Lab 3 — Timere/PWM**: `analogWrite()` este folosit pe pinii PD5, PD6, PB1, PB2 pentru a controla viteza motoarelor; `millis()` este folosit în `handleAcceleration()` pentru a incrementa viteza non-blocant

Calibrarea senzorului

Pragurile de distanță au fost determinate experimental. Threshold-ul de oprire a fost setat la 15cm — suficient de aproape ca mașina să nu se oprească prea devreme, dar suficient de departe ca motoarele să aibă timp să se oprească înainte de impact. Distanța de siguranță pentru terminarea manevrei de evitare a fost setată la 30cm, oferind suficient spațiu după ce mașina a ieșit din zona de pericol. Citirea distanței se face ca medie a 3 măsurători consecutive pentru a elimina ecourile false ale senzorului HC-SR04.

Optimizări

- Citirea distanței este plasată după procesarea comenzii Bluetooth în `loop()`, astfel încât comenzile să fie preluate imediat, fără să aștepte finalizarea celor 3 citiri ale senzorului.

- Accelerarea este implementată non-blocant cu `millis()` în loc de `delay()`, astfel încât senzorul și UART-ul să fie în continuare procesate în timp ce viteza crește.
- Comenzile primite în timpul manevrei autonome sunt ignorate — buffer-ul este golit la finalul manevrei, pentru a evita executarea unor comenzi acumulate în timp ce mașina era în control autonom.
- Comenzile Bluetooth sunt validate în `serialEvent()` înainte de a fi acceptate — doar caracterele din setul FBLRS sunt procesate, ignorând zgomotul de pe linie.

Demo

Un demo video al proiectului poate fi găsit aici.

Concluzii

- Proiectul și-a atins obiectivul principal: o mașină 4WD funcțională, controlabilă de la distanță prin Bluetooth, cu detecție și evitare autonomă a obstacolelor.
- Mecanismul de evitare autonomă funcționează fiabil, oprind mașina și alegând automat direcția cu mai mult spațiu disponibil.
- Accelerarea progresivă adaugă un comportament mai natural la pornire, apropiind mașina de comportamentul unui vehicul real.
- Proiectul poate servi ca punct de plecare pentru oricine vrea să înceapă cu microcontrollere, combinând mai multe concepte (UART, PWM, întreruperi, senzoristica) într-un rezultat concret și distractiv.

Download

Pentru testare, codul proiectului poate fi gasit în acest repo.

Bibliografie/Resurse

Resurse Software

- Tutorial YouTube — Arduino 4WD Robot Car — punct de plecare pentru structura proiectului
- Arduino Bluetooth Control — aplicația Android folosită pentru controlul mașinii
- PlatformIO — mediu de dezvoltare
- Arduino Reference — documentație funcții Arduino

Resurse Hardware

- ATmega328P Datasheet — Microchip
- L298N Motor Driver Datasheet
- HC-SR04 Ultrasonic Sensor Datasheet
- HC-05 Bluetooth Module Datasheet

Export to PDF

From:
https://ocw.cs.pub.ro/courses/ - **CS Open CourseWare**

Permanent link:
https://ocw.cs.pub.ro/courses/pm/prj2026/ciprian.popescu0411/235141

Last update: **2026/05/27 03:01**