KONTROLA KRETANJA MALENOG AUTOMOBILA NA DALJINSKO UPRAVLJANJE RUČNIM GESTAMA

Kirasić, Anđelo

VELEUČILIŠTE U KARLOVCU

STROJARSKI ODJEL

PREDDIPLOMSKI STRUČNI STUDIJ MEHATRONIKE

ANĐELO KIRASIĆ

KONTROLA KRETANJA MALENOG AUTOMOBILA NA DALJINSKO UPRAVLJANJE RUČNIM GESTAMA

ZAVRŠNI RAD

Karlovac, 2024.

KARLOVAC UNIVERSITY OF APPLIED SCIENCES

MECHANICAL ENGINEERING DEPARTMENT

PROFESSIONAL UNDERGRADUATE STUDY OF MECHATRONICS

ANĐELO KIRASIĆ

ARDUINO NANO CONTROLLED HAND GESTURE TOY CAR

FINAL PAPER

Karlovac, 2024.

VELEUČILIŠTE U KARLOVCU

STROJARSKI ODJEL

PREDDIPLOMSKI STRUČNI STUDIJ MEHATRONIKE

ANĐELO KIRASIĆ

KONTROLA KRETANJA MALENOG AUTOMOBILA NA DALJINSKO UPRAVLJANJE RUČNIM GESTAMA

ZAVRŠNI RAD

Mentor: dr. sc. Adam Stančić, prof. struč. stud.

Karlovac, 2024.

PREDGOVOR

Izjavljujem da sam završni rad na temu „Kontrola kretanja malenog automobila na daljinsko upravljanje ručnim gestama“ izradio samostalno, koristeći znanje stečeno tijekom školovanja, samostalnim istraživanjem i koristeći navedenu literaturu.

Zahvaljujem se svojoj obitelji koja mi je bila najveća podrška tijekom studija. Zahvaljujem se mentoru dr. sc. Adamu Stančiću na savjetima i pomoći pri izradi ovog rada.

Karlovac, Potpis

SAŽETAK

Maleni automobil na upravljanje ručnim gestama je inovativna vrsta igračke (dimenzija 255 x 160 x 110 mm i mase 385 g) koja omogućuje korisniku upravljanje pomoću gesta ruku. Korištenjem senzora, Arduino Nano mikroupravljača i tehnologije prepoznavanja gesta, maleni automobil može reagirati na pokrete ruku korisnika. Geste ruku omogućavaju usmjeravanje malenog automobila u željenom smjeru: naprijed, unatrag, lijevo ili desno.

Korisnici mogu uživati u potpuno novom iskustvu igre, budući da geste omogućuju intuitivnije upravljanje. Nema potrebe za tradicionalnim upravljačem ili daljinskim upravljačem, što znači praktičnost i jednostavnost korištenja. Potiče na kreativnost i razvijanje motoričkih vještina, a i igra postaje dinamičnija.

Osim zabavnog aspekta, maleni automobil može imati i edukativnu vrijednost. Kroz interakciju s igračkom, korisnici mogu naučiti o osnovama tehnologije, senzora i prepoznavanja pokreta. Sveukupno, ova inovativna naprava spaja zabavu, edukaciju i tehnologiju na jedinstven način, čineći je privlačnom opcijom za djecu i roditelje koji traže napredne i interaktivne igračke.

Ključne riječi: Arduino Nano, maleni automobil, mikroupravljač, ručne geste, tehnologija

SUMMARY

The hand gesture-controlled car is an innovative type of toy that allows the user to control it using hand gestures. By utilizing sensors, Arduino Nano microcontroller and gesture recognition technology, the car can respond to the user's hand movements, enabling it to move forward, backward, left or right.

Users can enjoy a completely new gaming experience as gestures allow for more intuitive control. There's no need for traditional controller or remote, which means convenience and ease of use. It encourages creativity and the development of motor skills, making the game more dynamic.

In addition to the fun aspect, the car can also have educational value. Through interaction with the toy, users can learn about the basics of technology, sensors and motion recognition. Overall, this innovative device combines entertainment, education, and technology in a unique way, making it an attractive option for children and parents looking for advanced and interactive toys.

Key words: Arduino Nano, hand gesture, microcontroller,technology, toy car

SADRŽAJ

POPIS SLIKA

POPIS TABLICA

1. UVOD

U današnjem digitalnom dobu, tehnologija igra ključnu ulogu u gotovo svim aspektima naših života. Od komunikacije i poslovanja do zabave i obrazovanja, tehnološki napredak neprestano transformira naše svakodnevno iskustvo. U tom kontekstu, razvoj interaktivnih igračaka predstavlja značajan segment inovacija, posebno u području dječje zabave i edukacije. Jedna od najnovijih inovacija u tom području je maleni automobil na upravljanje ručnim gestama, koji predstavlja spoj napredne tehnologije i zabavnog iskustva za djecu i roditelje.

Prezentirani završni rad pokazuje proces razvoja i implementacije malenog automobila na upravljanje ručnim gestama, analizirajući korake, izazove i postignuća u stvaranju funkcionalnog prototipa. Kroz detaljan pregled korištenih tehnologija i softverskih implementacija, bit će pružen uvid u funkcionalnost ovog jedinstvenog uređaja.

Kroz sveobuhvatan pregled procesa stvaranja i postignuća, prezentirani završni rad nastoji pružiti uvid u osobni projekt malenog automobila na upravljanje ručnim gestama kao primjer vlastite inovacije i kreativnosti. Kroz iskustvo stvaranja i razvoja ovog prototipa, nastoji se potaknuti daljnje istraživanje, eksperimentiranje i stvaranje u području tehnoloških projekata i inovacija i u nekim drugim područjima gdje se tehnologija može na ovaj ili sličan način unaprijediti.

2. MIKROUPRAVLJAČI

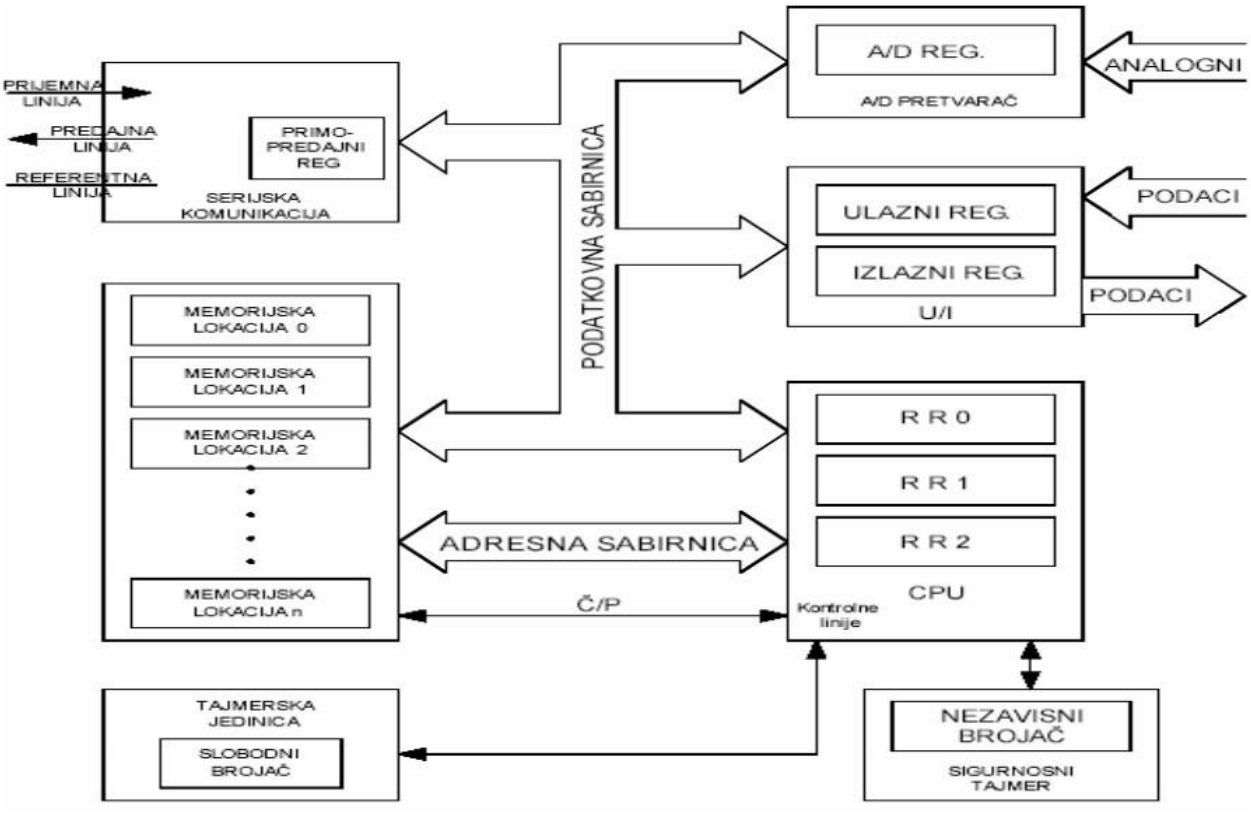
Mikroupravljači su mali, integrirani računalni sustavi koji se koriste za kontrolu različitih uređaja i sustava. Osnovna su komponenta mnogih elektroničkih uređaja, uključujući kućanske aparate, automobile, igračke, industrijske strojeve itd.

2.1. PRINCIP RADA MIKROUPRAVLJAČA

Mikroupravljači sadrže centralni procesor (engl. Central Processing Unit - CPU), memoriju, ulazno/izlazne međusklopove (ili portove) i dodatne komponente poput analogno – digitalnih pretvornika i serijskih komunikacijskih sučelja. Mikroupravljači rade na temelju programskog koda koji se pohranjuje u njihovu memoriju i koji definira funkcionalnost određenog uređaja.

Ključna karakteristika mikroupravljača je njihova sposobnost izvršavana zadataka u stvarnom vremenu. To znači da mogu brzo i precizno reagirati na vanjske podražaje ili promjene u okolini. Ova sposobnost čini ih idealnim za primjene poput upravljanja motorima, senzorima, svjetlosnim sustavima i drugim sustavima.

Mikroupravljači se proizvode u rasponu od 4 bita i s ograničenim mogućnostima, do vrlo brzih 32 bitnih procesora. Kako bi mikroupravljač mogao komunicirati s vanjskim parametrima, koriste se međusklopovi (engl. *port*). Pristup navedenim međusklopovima je dvosmjeran, a funkcionalnost sklopa se mijenja prema potrebi. Mikroupravljači se programiraju s pomoću osobnog računala preko ISP (engl. *In System Programming*) sustava. Razvoj softvera za mikroupravljače obično uključuje pisanje koda na programskim jezicima poput C, C++ i Python.



Slika 1: Blok shema mikroupravljača

Osnovni koraci koji opisuju rad mikroupravljača su:

1. **Inicijalizacija:** Nakon pokretanja mikroupravljača, prvo se izvršava inicijalizacija, definiraju se početni parametri, konfiguriraju se ulazno/izlazni pinovi i izvršavaju se druge pripremne radnje
2. **Izvršavanje programskog koda:** Nakon inicijalizacije, mikroupravljač počinje izvršavati programski kod koji je pohranjen u njegovoj memoriji. Programski koda definira logiku i funkcionalnost uređaja, uključujući način obrade ulaznih podataka i kontrolu izlaznih uređaja.
3. **Obrada ulaza:** Mikroupravljač neprestano čita ulazne podatke putem svojih ulazno/izlaznih pinova ili putem komunikacijskih sučelja poput serijske ili paralelne komunikacije. Podaci i informacije koje obrađuje proizlaze iz senzora, sklopki ili drugih vanjskih izvora.

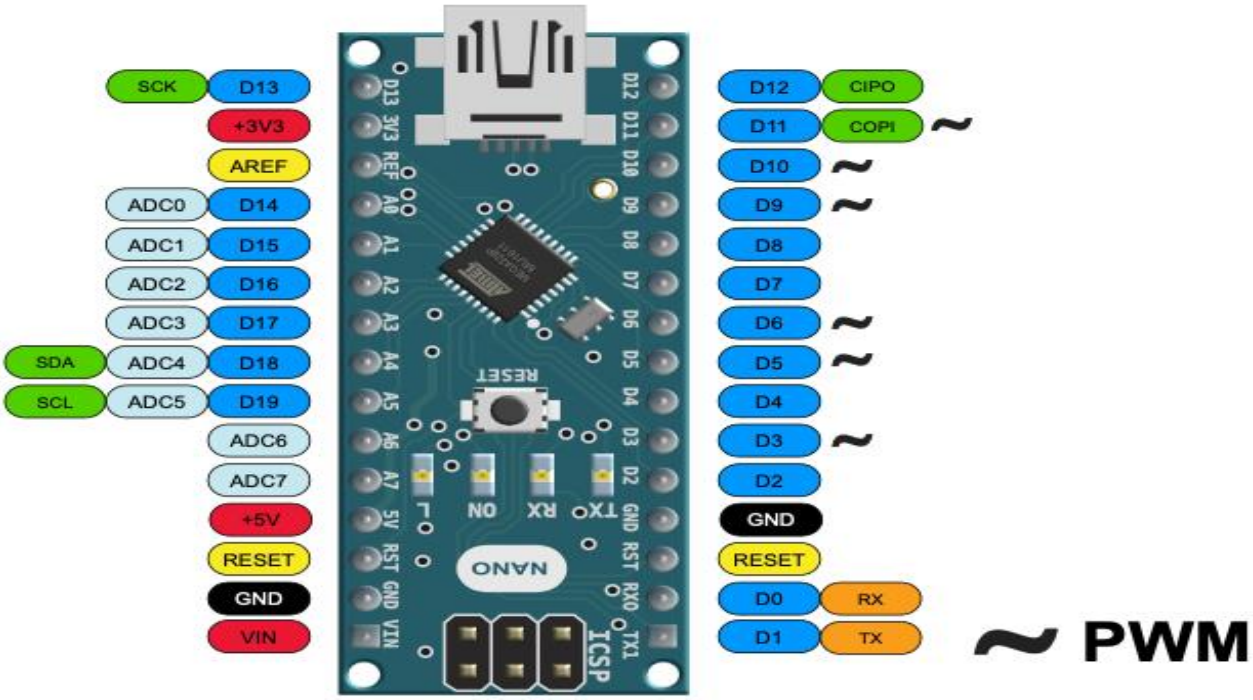
4. **Obrada podataka:** Nakon što dobije ulazne podatke, mikroupravljač izvršava odgovarajuće operacije prema programskom kodu (npr. računanje, usporedba, odlučivanje itd.)
5. **Generiranje izlaza:** Na temelju rezultata obrade ulaznih informacija, mikroupravljač generira odgovarajuće izlazne signale koji kontroliraju rad različitih uređaja ili sustava (npr. pokretanje motora, upravljanje svjetlosnim izvorima, prikazivanje informacija na zaslonu itd.)
6. **Ponavljanje procesa:** Ovaj ciklus obrade ulaza, izvršavanja programa i generiranje izlaza neprestano se ponavlja sve dok mikroupravljač ostaje uključen i pokrenut ili ako je u programskom kodu napisano drugačije.

2.2. ODABIR MIKROUPRAVLJAČA

Cilj pri odabiru mikroupravljača je izabrati onaj koji sadrži tražene parametre i funkcije za zadaće koje obavlja. Postoji mnogo različitih vrsta mikroupravljača koji se po svojem principu rada baš i ne razlikuju, ali svaki ima svoje prednosti i nedostatke.

Za ovaj projekt je korišten Arduino Nano. U odnosu na Arduino Uno, manjih je dimenzija, kompaktniji je i ima isti ATmega328 mikroupravljač. Što se tiče povezivanja na Internet (*Wi-Fi*), tu mogućnost Arduino Nano nema. Za povezivanje na Internet zahtijeva dodatni modul za *Wi-Fi* komunikaciju za razliku od ESP32 mikroupravljača koji već ima ugrađen *Wi-Fi* čip i pogodniji je za IoT (*Internet of* Things) projekte i bežično povezivanje. Raspberry Pi mikroupravljač je također jedan od popularnijih i korištenijih mikroupravljača, ali je pogodniji za složenije aplikacije koje zahtijevaju više obrade podataka, grafičko sučelje i povezanost s Internetom.

Kako za ovaj projekt nije bilo nužno imati povezanost s Internetom i koristiti grafičko sučelje, Arduino Nano je dobro odradio zadaće i ispunio očekivanja ovog projekta. Detaljnije pojašnjenje pristupa (engl. *pinout*) može se vidjeti na slici ispod (Slika 2):



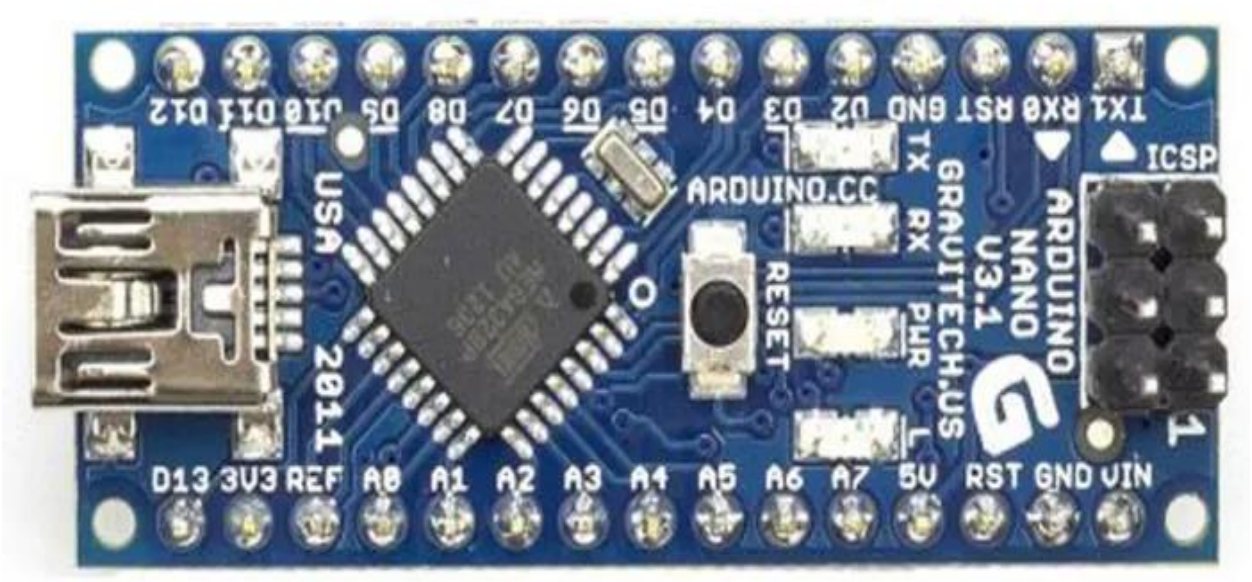
Slika 2: Arduino Nano izvedba pinova [2]

1. **Digitalni ulazno/izlazni pinovi (D2 – D13):** Arduino Nano ima ukupno 14 ulazno/izlaznih (I/O) pinova. Koriste se za povezivanje s vanjskim uređajima (senzori, LED diode, motori, prekidači itd.). Svaki od ovih pinova može biti konfiguriran kao ulaz ili izlaz putem programskog koda. Od 14 pinova, 6 ih ima PWM (engl. *Pulse Width Modulation*) mogućnost. PWM pinovi se koriste za komponente nad kojima želimo imati kontrolu (kontrola brzine motora, regulacija svjetla i sl). Na PWM pinovima se periodično mijenja širina impulsa signala, vrijednost napona ili jakosti struje, pa time dobivamo kontrolu nad određenim uređajem.
2. **Analogno ulazni pinovi (A0-A7):** 8 analognih pinova koji omogućuju priključivanje analognih senzora, mjere promjene napona ili jakosti struje. Npr. senzor temperature, svjetla i zvuka, pa te vrijednosti mogu pročitati putem programskog koda.
3. **Napajanje (VIN, VCC i GND):** Arduino mikroupravljač ima pinove za napajanje, VCC (+5V) (engl. *Voltage Common Collector*) i GND (engl. *ground*, zemlja). VCC pin dovodi pozitivan napon na Arduino mikroupravljač, a GND se spaja s negativnim polom napajanja ili zemlje. Također pored VCC pina, postoji i VIN pin. Ovi pinovi služe za napajanje pločice, jedina razlika je u tome što je na VCC pinu stabilizirani napon od 5V, dok je VIN (engl. *Voltage Input)* ulazni napon koji korisnik koristi za unos vanjskog napajanja na pločicu.

4. **USB (*type* C) priključak:** USB priključak kojim korisnik povezuje Arduino mikroupravljač i računalo u svrhu programiranja i prijenosa podataka s računala. Služi i kao izvor napajanja kada je pločica spojena na računalo.

5. **TX/RX pinovi:** TX (engl. *Transmit*) i RX (engl. *Receive*) pinovi su namijenjeni za serijsku komunikaciju s drugim uređajima. TX pin šalje podatke, a na drugom kraju RX pin prima podatke i izvršava naredbe.

6. **RESET:** Reset pin resetira mikroupravljač, kratko se spaja sa GND pinom čime će se mikroupravljač resetirati i ponovo pokrenuti.



Slika 3: Mikroupravljač Arduino Nano, ATmega328 [4]

3. PROJEKTIRANJE SUSTAVA

U ovom dijelu završnog rada bit će detaljnije opisani dijelovi i struktura malenog automobila pomoću shematskih prikaza i opisanih slika, te karakteristike komponenti korištenih za izradu ovog sustava. U narednim poglavljima bit će detaljnije objašnjeno kako sastaviti hardverske i softverske komponente sustava.

3.1. DIJELOVI SUSTAVA

Arduino Nano mikroupravljač

Mikroupravljač je već opisan u prethodnom odlomku, te će u nastavku biti prezentirana tablica tehničkih specifikacija mikroupravljača Arduino Nano.

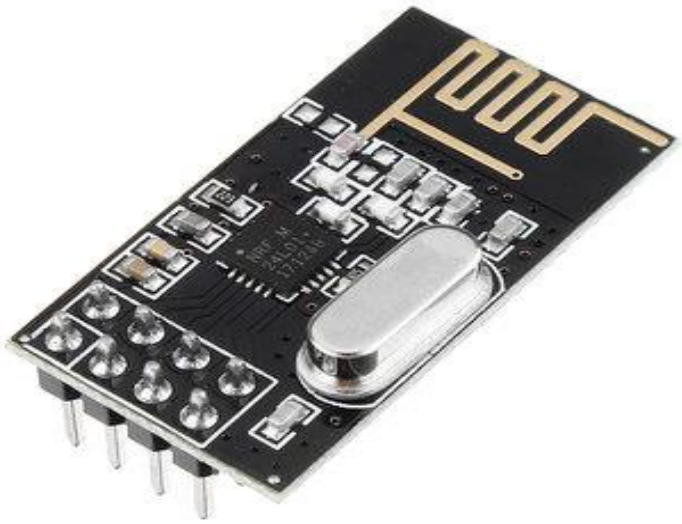
Tablica 1: Tehničke specifikacije mikroupravljača Arduino Nano [7]

Mikroupravljač	ATMega 328
Radni napon	5 V
Ulazni napon (preporučeni)	7 – 12 V
Ulazni napon (granični)	6 – 20 V
Digitalni I/O pinovi	22
Analogni ulazni pinovi	8
DC struja po I/O pinu	20 mA
PWM pinovi	6
Flash memorija	32 KB od kojih 2 KB za bootloader
SRAM	2 KB
EEPROM	1 KB
Radni takt	16 MHz

Kao razvojno okruženje Arduino mikroupravljači koriste Arduino IDE (engl. Integrated Development Environment). Arduino IDE temelji se na C programskom jeziku i izvorni kod je dostupan svim korisnicima koji ga mogu mijenjati, prepravljati i unaprijediti (tzv. „open source“). Arduino IDE podržava mnoge biblioteke (engl. Library), koje sadrže razne funkcije za određene komponente kojima Arduino mikroupravljač, u ovom slučaju Arduino

Nano šalje ili prima informacije. Osnovne biblioteke su već implementirane u sam sustav Arduino IDE-a, a po potrebi i zahtjevima komponenti one se mogu dodatno samostalno preuzeti. Arduino IDE je podržan od više operacijskog sustava kao što su MS Windows, Mac i Linux.

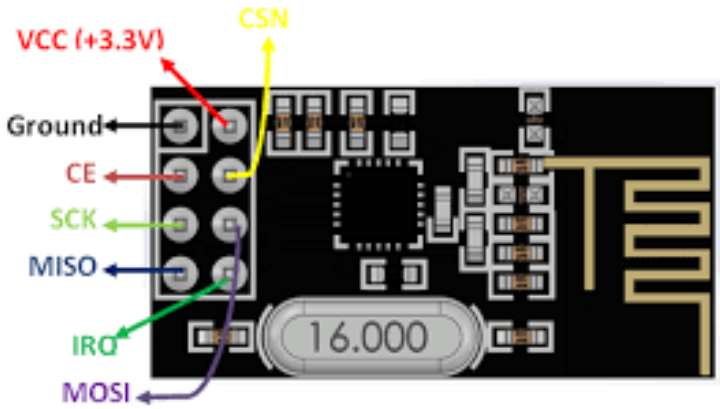
Bežični modul NRF24L01



Slika 4: Bežični modul NRF24L01 [6]

NRF24L01 je bežični modul koji se koristi za komuniciranje na kratkim udaljenostima, u IoT (engl. *Internet of Things*, Internet stvari) projektima, beskontaktnim sustavima i daljinskim upravljanjem, te senzorskim mrežama i sličnim aplikacijama. NRF24L01 bežični modul radi u frekvencijskom opsegu od 2.4 GHz, što mu omogućava komunikaciju i kompatibilnost s mnogim drugim bežičnim uređajima. Podržava brzinu prijenosa podataka do 2 Mbps (engl. *Megabits per second*). Niska potrošnja energije ga čini idealnim izborom za uređaje koji se napajaju preko baterija.

Njegova dimenzija od 15.5 mm x 29 mm olakšava integraciju u različite uređaje, a sučelje je jednostavno, razumljivo i lako upotrebljivo. Također je *„open source“* i ima različitih verzija i modula. *„Open source“* u odnosu na bežični modul NRF24L01 odnosi se na dostupnost i upotrebu softverskog koda koji je slobodno dostupan javnosti. Ovo je RF (engl. *Radio Frequency)* modul što znači da omogućuje bežičnu komunikaciju između različitih uređaja. U nastavku bit će detaljnije opisana izvedba pinova bežičnog modula NRF24L01.



Slika 5: Izvedba pinova na bežičnom modulu NRF24L01 [6]

1. VCC (+ 3.3 V): pin za napajanje modula, povezuje se na izvor napajanja od 3.3 V

2. GND (*Ground*): pin za uzemljenje, odnosno negativni pol izvora napajanja

3. CE (*Chip Enable): ovaj pin omogućava ili onemogućava čip. Aktiviranjem ovog pina pokreće se, odnosno zaustavlja se komunikacija sa NRF24L01

4. CSN (*Chip select)*: ovim pinom se odabire uređaj s kojim će mikroupravljač komunicirati i obično se koristi zajedno sa SPI (engl. *Serial Peripheral Interface*) protokolom. SPI protokol je serijski interfejs koji se koristi za komunikaciju između mikroupravljača i perifernih uređaja kao što su senzori, ekrani, memorijski čipovi i drugi integrirani krugovi. Omogućava brzu i pouzdanu komunikaciju među uređajima. Uređaj će komunicirati sa NRF24L01 samo ako su im adrese identične. Adresa uređaja koji šalje podatke, mora biti ista kao adresa uređaja koji prima podatke. Inače neće biti komunikacije.

5. SCK (*Serial Clock)*: prenošenje taktova za sinkronizaciju serijske komunikacije između mikroupravljača (Arduino Nano) i NRF24L01 zaslužan je SCK pin.

6. MOSI (*Master Out Slave In)*: pin za prenošenje podataka sa mikroupravljača do NRF24L01 u SPI komunikaciji.

7. MISO (*Master In Slave Out)*: ovaj pin radi obrnuto od MOSI pina, znači prenosi podatke sa NRF24L01 do mikroupravljača u SPI komunikaciji

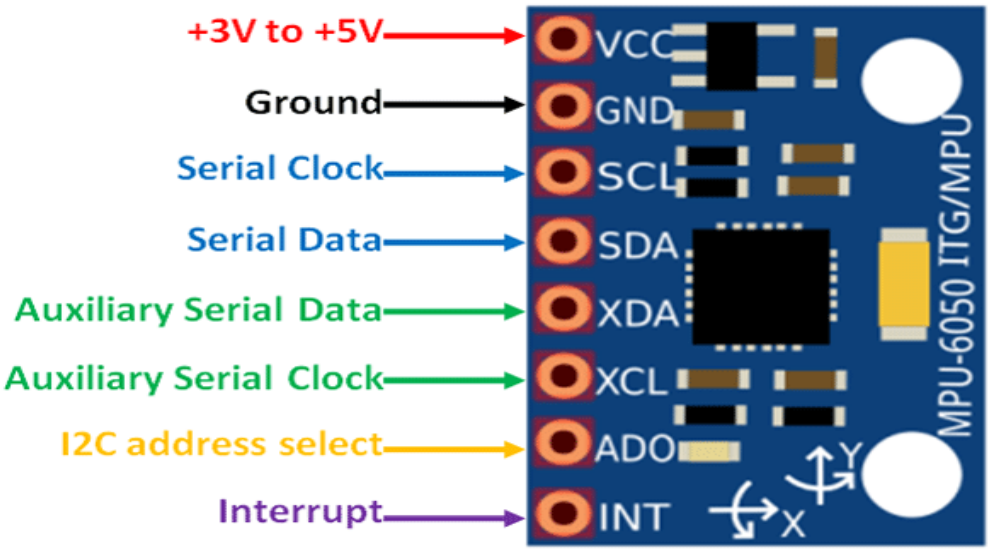
8. IRQ (*Interrupt Request)*: ovo je prekidni pin, tj. koristi se za prekid i obavještavanje mikroupravljača kada je primljena nova informacija ili kada se vrši neka druga radnja u NRF24L01 pločici.

Senzor MPU6050

Senzor MPU6050 što pruža informacije o poziciji sklopa u prostornom trodimenzionalnom sustavu. Kombinacija je troosnog (X,Y,Z) akcelerometra i žiroskopa. Koristi se u područjima robotike i pri stabilizaciji letjelica. U prezentiranom završnom radu senzor MPU6050 korišten je kao žiroskop. Uloga mu je prepoznati pokrete zgloba ruke. Pored pozicije i nagiba, ovaj sklop daje informacije o ubrzanju i kutnoj brzini.

Slika 6: Integrirani krug MPU6050 [10]

MPU6050 komunicira preko I2C (engl. *Inter – Integrated Circuit)* ili SPI digitalnih sučelja, što mu daje jednostavnost integracije u razne projekte. Omogućava programabilne opsege mjerenja za akcelerometar i žiroskop i time se prilagođava na osjetljivost senzora prema potrebama, tj. softverski je podesiv. Moguće mu je pormijeniti mjerne opsege i područja, proširiti ih ili smanjiti, ovisno o potrebama različitih sklopova.



Slika 7: Izvedba pinova na senzoru MPU6050 [10]

1. VCC: kao i u ostalim komponentama, VCC pin se povezuje na izvor napajanja od 3.3 V ili 5 V, ovisno o potrebi

2: GND: pin za uzemljenje, odnosno negativni pol izvora napajanja

3: SDA (*Serial Data*): dvosmjerni kanal za slanje i primanje podataka između mikroupravljača i MPU6050 senzora I2C komunikacijskim protokolom

4. SCL (*Serial Clock*): određuje brzinu prijenosa podataka i sinkronizira primanje i slanje podataka između mikroupravljača i MPU6050

5. XDA i XCL: XDA (engl. *Expandable Device Architecture)* i XCL (engl. *Extended Control Language)* su pinovi koje nisu uvijek prisutni na svim pločicama MPU6050, oni su samo dodatna serijska linija podataka (engl. *Data*) ili dodatna serijska linija sata (engl. *Clock*)

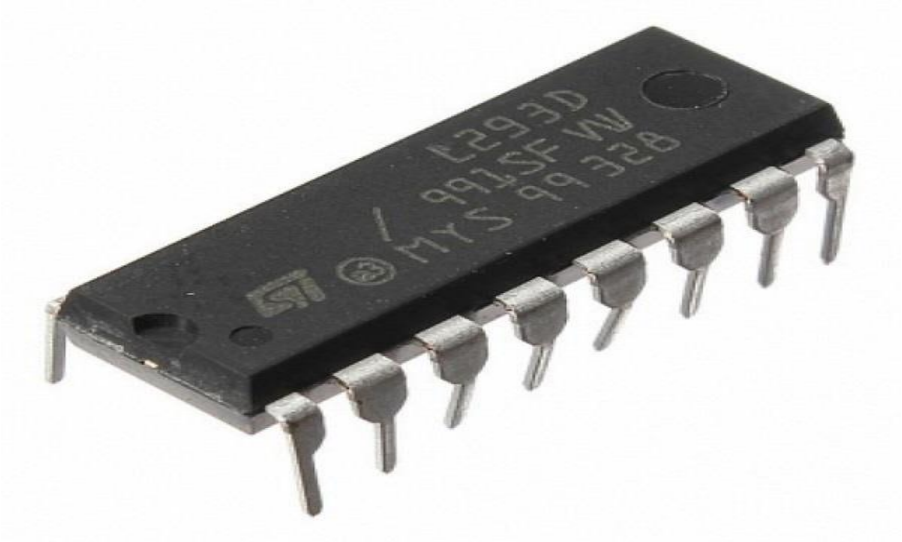
6. ADO: odabire adresu senzora

7. INT (*Interrupt*): pin za prekid

XDA pin obično služi kao izlazni podatkovni kanal, dok XCL pin služi kao izlazni kanal za sat. Oni pružaju dodatne mogućnosti za komunikaciju s drugim uređajima, osim standardne I2C komunikacije.

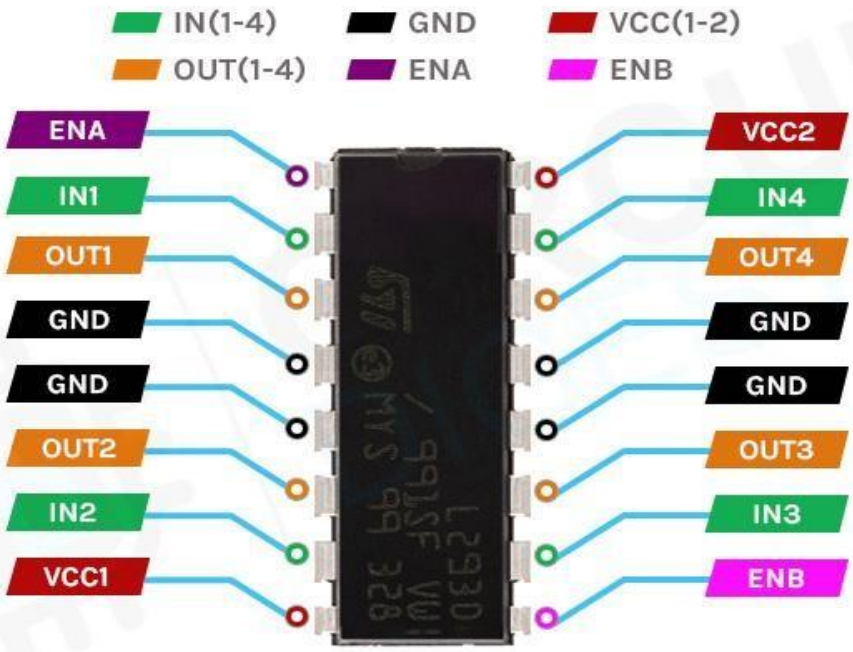
L293D modul za upravljanje motorima

U prezentiranom projektu korišten je modul za kontrolu i upravljanje motorima L293D koji omogućava kontrolu brzine i smjera rotacije jednog ili više istosmjernih motora. Dizajniran je kao dvostruki H – most, što znači da može kontrolirati rotaciju motora u oba smjera (naprijed i nazad) i omogućava kontrolu brzine.



Slika 8: Modul za kontrolu i upravljanje motorima L293D [12]

Ne može direktno skrenuti lijevo ili desno, ali ga je moguće isprogramirati preko mikroupravljača da prilikom skretanja u jednu od strana, jedan od motora se rotira sporije, a drugi brže. Time je postignuto skretanje u lijevo i u desno koristeći rotaciju dvaju motora koji se mogu rotirati samo unaprijed i unatrag. Koristan je jer može podnijeti visoke jakosti struje i naponske vrijednosti, a ima i zaštitu od povratne struje. Povratna struja se vraća u ulazni dio kruga kao rezultat povratne veze, a utječe na točnost, performanse i stabilnost elektroničnog kruga i gubitak linearitеta uz povećanje potrošnje energije. Ugrađena dioda zaštite štiti čip od povratnih struja. Modul sadrži i termalnu zaštitu od pregrijavanja i smanjuje gubitke energije. Pogodan za primjenu u CNC strojevima i bilo kojem sklopu gdje postoji mogućnost kontrole brzine. U ovom projektu, kontrola brzine će biti omogućena moduliranjem PWM signala na odgovarajućim pinovima.



Slika 9: L293D čip pinout [12]

1. ENA, ENB (*Enable A,B*): ovi pinovi služe za blokiranje i deblokiranje motora, kada se na njih primjeni visoki napon, tj. logička „1“, motori A i B su omogućeni za rad

2. IN1, IN2, IN3, IN4 (*Input 1,2,3,4*): ovo su pinovi koji se povezuju na mikroupravljač, to su ulazi (engl. *Input*) koji dovođenjem logičke „0“ ili „1“ određuju smjer rotacije, „0“ unatrag, „1“ unaprijed. U ovom projektu korištena su dva istosmjerna motora, odnosno ulazi IN1 i IN2 određuju rotaciju lijevog, a ulazi IN3 i IN4 rotaciju desnog motora.

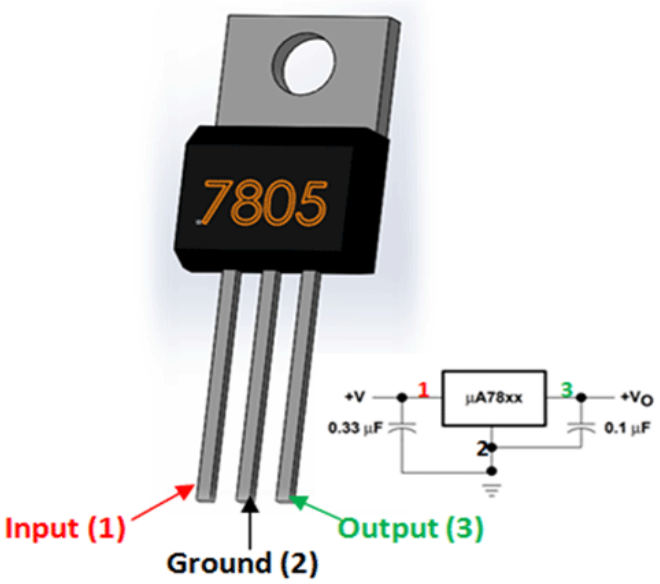
3. OUT1, OUT2, OUT3, OUT4 (*Output 1,2,3,4*): takozvani izlazni pinovi za spajanje motora. Na ove pinove spajaju se pozitivni i negativni pol motora. Na primjer, OUT1 je pozitivni pol motora, a OUT2 je negativni pol motora, kao i za OUT3 i OUT4 za drugi motor.

4. GND: pinovi uzemljenja (engl. *Ground*)

5. VCC1 i VCC2 (*Voltage at the common collector*): pinovi za napajanje internih logičkih dijelova čipa, a jedan od njih se koristi i spaja za napajanje motora kao Vs napon (izlazni napon)

7805 regulator napona

Regulator napona 7805 se koristi za stabilizaciju napajanja u raznim elektroničkim uređajima. Pruža stabilni izlazni napon od 5 V, neovisno o promjenama ulaznog napona ili opterećenja.



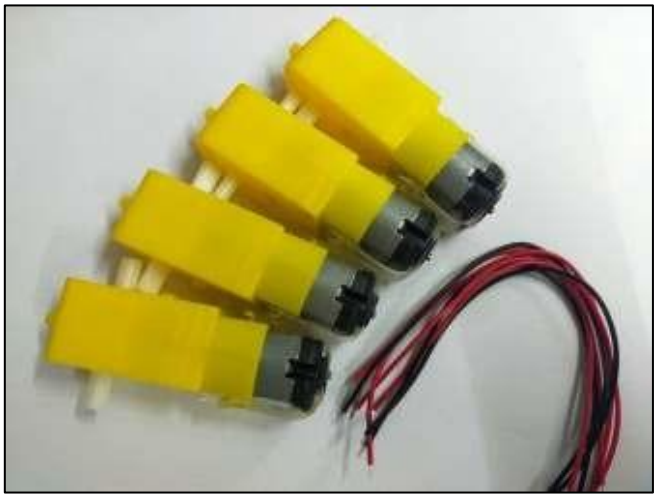
Slika 10: Modul za regulaciju napona 7805 [15]

Izlazni napon će ostati konstantan unatoč varijacijama u ulaznom naponu. Jednostavan je za integraciju u elektroničke sklopove. Maksimalna izlazna jakost struje je oko 1A, što ga čini pogodnim za napajanje manjih elektroničkih uređaja.

3.2. SASTAVLJANJE

U ovom poglavlju bit će objašnjene ostale komponente koje su sastavni dio konstrukcije, tj. karoseriju malenog automobila:

1. Motori za pogonske kotače sa žicama (prikazani na slici ispod):



Slika 11: Istosmjerni motori i žice za povezivanje

2. Kotačni malenog automobila (prikazani na slici ispod):

Slika 12: Kotači za maleni automobil

3. Pločice za fiksiranje motora, kako bi motori bili dobro fiksirani i ispravno postavljeni

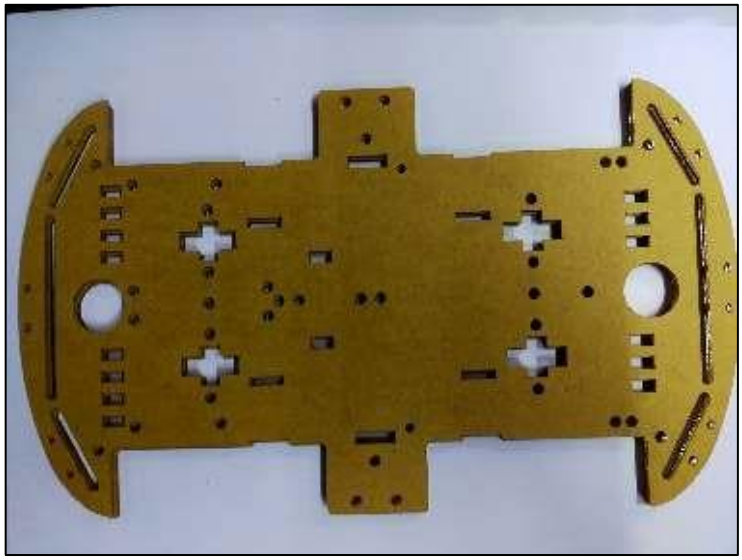
4. Bakrenih „stupova“ (dimenzija 6 x 35 mm / 7 x 6 mm) koji odvajaju gornji i donji dio šasije

5. Vijci duljine 3 mm za pritezanje i čvrstoću šasije sa mogućnošću rastavljanja šasije

Pločice za fiksiranje motora, bakreni stupovi i vijci prikazani su na slici ispod:

Slika 13: Vijci, bakreni stupovi, pločice za fiksiranje motora

6. Prozirna pločica od pleksiglasa (dimenzija 255x 160 x 3 mm), lagana, ali i jako izdržljiva, čine gornji i donji dio šasije (prikazani na slici ispod)



Slika 14: Pločica od pleksiglasa

Alati korišteni prilikom sastavljanja:

1. Lemilica, kombinirana kliješta, precizni odvijač, pištolj za ljepilo

a) Lemilica

b) Kombinirana kliješta

c) precizni odvijač

d) pištolj za ljepilo

Slika 15: Alati potrebni za sastavljanje malenog automobila

3.3. POSTUPAK SASTAVLJANJA

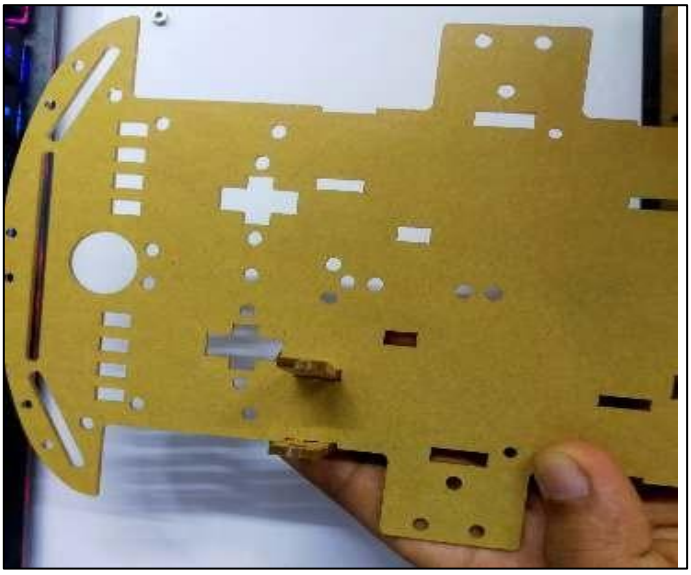
U ovom poglavlju biti će opisano kako i ispravno sastaviti i povezati fizičke (hardverske) dijelove malenog automobila. Potrebno je koristiti prikladan alat za svaki sastavni dio. Lemilica je korištena za postupak lemljenja komponenata na pločicu. Kombinirana kliješta su korištena za skidanje izolacije sa žica, kao i za prilagodbu dužine žica. Za pričvršćivanje vijaka korišten je precizni odvijač, a pištolj na vruće ljepilo za lijepljenje plastičnih dijelova malenog automobila.

1. Prije svega prvo se provjerava stanje motora i jesu li neke komponente oštećene. Nakon provjere uzimamo motore i na njih lemilicom lemimo crnu i crvenu žicu na njihove + i – polove, (crna žica je u većini slučajeva -, a crvena je + pol). Da bi se uvjerili da je

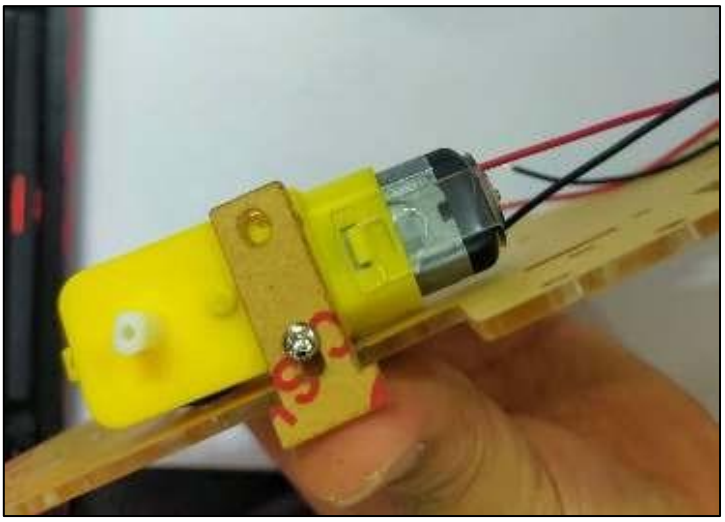
sve dobro spojeno, potrebno je kratko spojiti polove motora na bateriju i provjeriti je li se motor okreće.

Slika 16: Lemljenje žica na motor

2. Zalemljeni motori postavljaju se na pločicu pleksiglasa na predviđena mjesta i učvršćuju se pločicama za fiksiranje. Svaki motor se postavlja između dviju pločica i priteže vijcima.



Slika 17: Postavljanje pločica za fiksiranje motora



Slika 18: Fiksiranje motora

3. Postavljanje bakrenih stupova koji drže samu konstrukciju gornjeg i donjeg dijela šasije (Slika 18.)

Slika 19: Postavljanje bakrenih stupova

4. Postavljanje kotača na zupčanike motora i zatvaranje gornjim dijelom šasije. Bakreni stupići na slici iznad su pritegnuti s donje strane šasije, a nakon zatvaranja šasije s gornjim dijelom, također je potrebno pritegnuti bakrene stupiće.

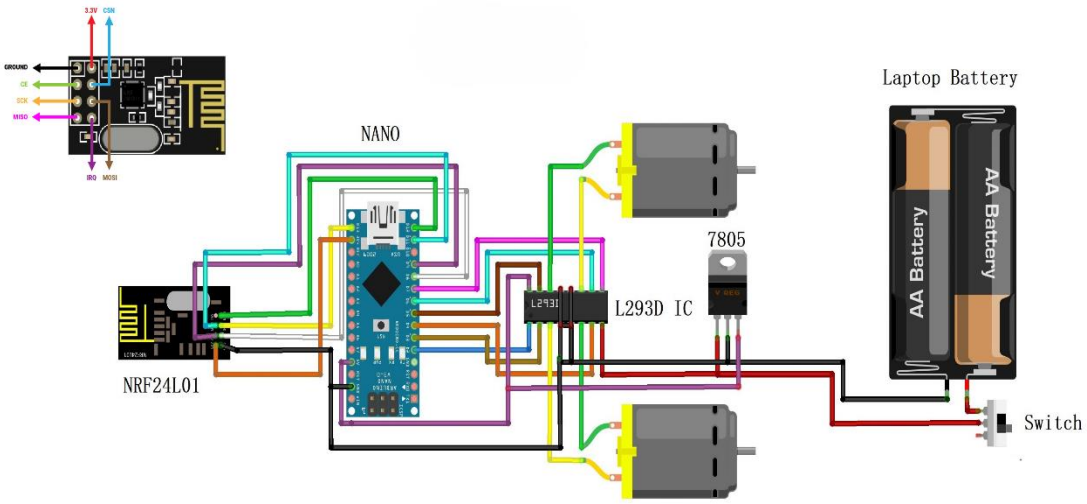
Slika 20: Sastavljena karoserija

3.4. LEMLJENJE TISKANE PLOČICE

Lemljenje je proces spajanja dvaju ili više metalnih dijelova korištenjem rastaljenog metala (lema) koji ima niže talište od materijala koji se spajaju. Postoji tvrdo i meko lemljenje, a u elektronici je naviše primijenjeno meko lemljenje. Lemljenje stvara pouzdane spojeve i vrlo je korisna tehnika bez potrebe za visokim temperaturama, što je pogodno za delikatne i osjetljive materijale i komponente. Za ovaj projekt korištene su vetronit pločice 10 x 10 mm za RX (engl. *Receiver* - prijemnik i 8 x 5 mm za TX (engl. *Transmitter* - predajnik) dio. Pločice su lemljene univerzalnom lemilicom ZD – 707NL od 40W.

SHEMA LEMLJENJA (*Receiver*)

Prilikom lemljenja treba biti oprezan da se slučajno lem ne prelije na neke druge kontakte komponenata, što može dovesti do neispravnog rada komponenti i samog završnog projekta. Na slici ispod se može vidjeti shematski prikaz prijemnika:



Slika 21: Shema lemljenja RX komponente [17]

RASPORED PINOVA KOD LEMLJENJA RX KOMPONENTE

U sljedećem poglavlju prezentiranog završnog rada, bit će opisan redoslijed lemljenja pinova i na koji način se spajaju kako bi cijeli sklop funkcionirao.

Povezivanje mikroupravljača i modula za bežičnu komunikaciju NRF24L01

U tablici ispod se nalazi raspored pinova za mikroupravljač Arduino Nano i raspored pinova za modul za bežičnu komunikaciju NRF24L01. Tablica prikazuje pravilno povezivanje pinova sa mikroupravljača Arduino Nano na modul NRF24L01.

Tablica 2: Povezivanje pinova između mikroupravljača i modula NRF24L01

ARDUINO NANO	NRF24L01
VCC	3.3 V
GND	GND
D13	SCK
D12	MISO
D11	MOSI
D9	CSN
D8	CE

Povezivanje mikroupravljača i modul za kontrolu i upravljanje motorima L293D

U tablici ispod se nalazi raspored pinova za mikroupravljač Arduino Nano i raspored pinova za modul za kontrolu i upravljanje motorima L293D. Tablica prikazuje pravilno povezivanje pinova sa mikroupravljača Arduino Nano na L293D.

Tablica 3: Povezivanje pinova između mikroupravljača i modula L293D

ARDUINO NANO	L293D
5V	VCC
GND	GND
D2	ENABLE A (1,2)
D3	INPUT 1
D4	INPUT 2
D5	INPUT 4
D6	INPUT 3
D7	ENABLE B (3,4)

Lemljenje ostalih komponenti

U sljedećem poglavlju bit će opisano povezivanje ostalih komponenti na pločicu i međusobno s drugim komponentama. Tablica ispod prikazuje lemljenje istosmjernih motora na modul za upravljanje i kontrolu motora L293D:

Polovi istosmjernih motora	Modul L293D
+	OUT 1
-	OUT 2
+	OUT 3
-	OUT 4

Tablica 4: Povezivanje pinova istosmjernih motora sa modulom L293D

Tablica ispod prikazuje spajanje pinova regulatora napona 7805 sa modulom upravljanja i kontrole motora L293D:

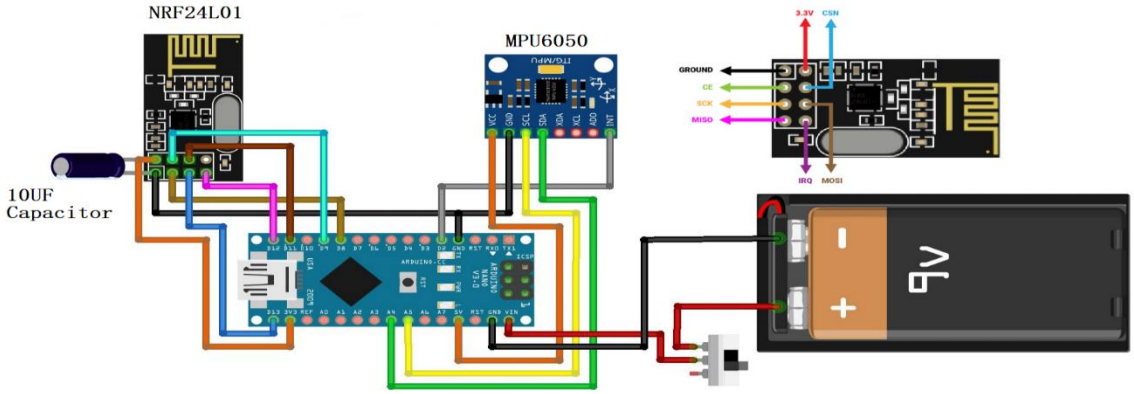
7805 regulator napona	Modul L293D
IN	VCC IN
GND	GND
OUT	VCC OUT

Tablica 5: Povezivanje pinova regulatora napona 7805 sa modulom L293D [14]

Obični *switch* (prekidač, sklopka) spojen je serijski između plus pola baterije i regulatora napona 7805. Prekidač (*ON/OFF*) služi za paljenje i gašenje sklopa.

SHEMA LEMLJENJA (Transmitter)

Shema lemljenja za predajnik (*Transmitter)* je vrlo slična shemi lemljenja za prijemnik (*Reciever)*. Bežični modul NRF24L01 se lemljenjem povezuje s mikroupravljačem na isti način kao i kod prijemnog (RX) dijela sklopa.



Slika 22: Shema lemljenja TX komponente [17]

Jedina razlika je u tome što predajni (TX) dio sklopa ulazi još i modula MPU6050 (akcelerometar i žiroskop), baterija od 9V u odnosu na shemu lemljenja prijemnog (RX) dijela sklopa i elektrolitski kondenzator kapaciteta 10μF koji se lemi između GND i 3.3V pinova na NRF24L01 modulu. Prekidač se povezuje na plus pol baterije s jedne strane i na VIN pin napajanja mikroupravljač Arduino Nano s druge strane.

U tablici ispod se nalazi raspored pinova za mikroupravljač Arduino Nano i raspored pinova za modula MPU6050. Tablica prikazuje pravilno povezivanje pinova sa mikroupravljača Arduino Nano na MPU6050.

Tablica 6: Povezivanje pinova između mikroupravljača i modula MPU6050 [10]

ARDUINO NANO	**MPU6050**
5 V	VCC
GND	GND
A5	SCL
A4	SDA
D2	INT

4. PROGRAMIRANJE MIKROUPRAVLJAČA

U ovom dijelu završnog rada bit će detaljnije opisan postupak programiranja, testiranja zalemljenih kontaktnih površina te spremnost svih povezanih komponenti za rad.

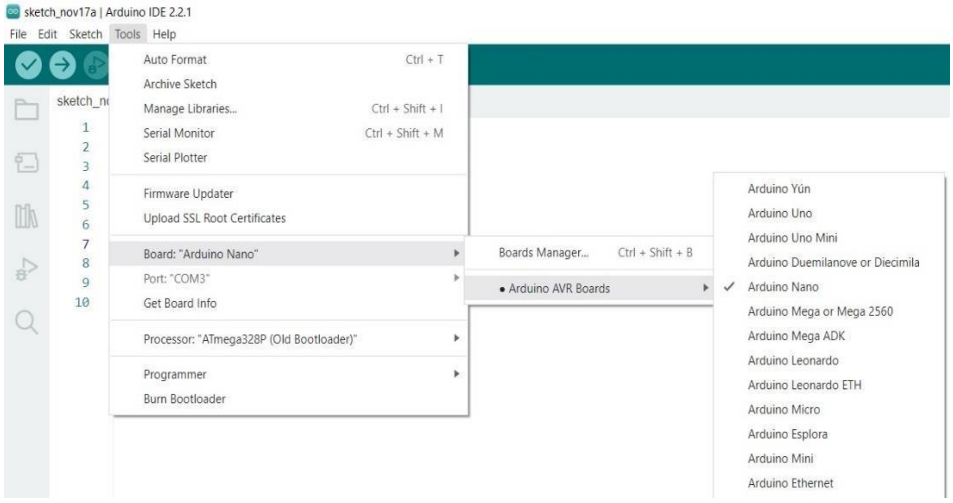
Program koji je korišten prilikom programiranja i softverske izvedbe je Arduino IDE verzija 2.2.1. (engl. *Integrated Development Environment)*. Arduino IDE je besplatno programsko rješenje koje omogućava razvoj projekata za Arduino mikroupravljače. Ovo razvojno okruženje je popularno među korisnicima jer je jednostavno za korištenje i dolazi sa već instaliranim dodatnim programskim bibliotekama koje se koriste u radu (engl. *Library)*.

4.1. POVEZIVANJE MIKROUPRAVLJAČA S RAČUNALOM

Da bi mikroupravljač mogao komunicirati s računalom potrebno ga je pravilno povezati. U ovom slučaju koristi se USB tip C kabel koji vrši napajanje i prijenos podataka od računala do mikroupravljača. Provjera povezanosti i odgovarajućeg pogonskog programa (engl. *Driver*) se obavlja automatski putem MS Windows 11 OS-a (Operacijski Sustav). U nekim slučajevima računalo automatski ne prepoznaje da je kabel povezan, te je potrebno provesti postupak ažuriranja pogonskog programa (za instaliran operativni sustav na računalu) i odabrati odgovarajuće serijsko komunikacijsko sučelje (engl. *COM Port)*.

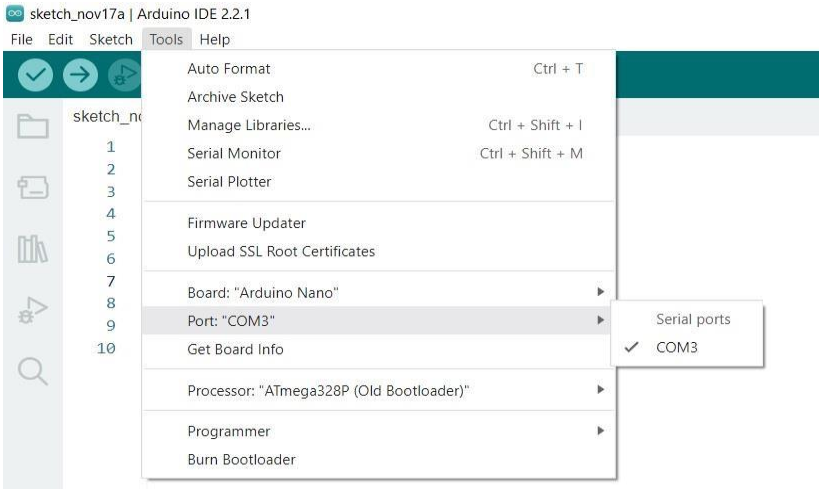
Slika 23: USB Type C, kabel korišten za povezivanje računala i mikroupravljača [8]

Nakon povezivanja mikroupravljača i računala kabelom sa slike iznad pokreće se Arduino IDE aplikaciju na računalu. U aplikaciji je potrebno odabrati odgovarajući model, tj. tip mikroupravljača koji se koristi. Različiti Arduino mikroupravljači su slični, ali nisu iste izvedbe. Male su razlike u programiranju različitih Arduino mikroupravljača, pa ako se dogodi da mikroupravljač ne funkcionira kako bi trebao, moguće je da je odabran krivi model mikroupravljača. Model odabiremo tako da pratimo padajuće izbornike sa gornjeg lijevog kuta aplikacije odabirom opcija: *Tools* ► *Board* ► *Arduino AVR Boards* ► *Arduino Nano*.



Slika 24: Odabir modela mikroupravljača

Nakon odabira mikroupravljača, potrebno je odabrati odgovarajuće serijsko komunikacijsko sučelje ako ga računalo nije prepoznalo automatski. Ako postupak odabira nije automatski proveden, serijsko komunikacijsko sučelje se odabire odabirom opcija padajućeg izbornika: *Tools* ► *Ports* ► *COM.* Bit će naveden veći broj opcija, recimo npr. COM1, COM2… COM8 (Slika 24.) U slučaju da nije poznato komunikacijsko sučelje na koje je spojen USB kabel za povezivanje, potrebno ga je odspojiti i ponovno spojiti na računalo i ponovno provjeriti oznaku komunikacijskog sučelja na računalu. Nakon toga potrebno je provjeriti ispravnost koda odabirom opcije *Verify.* Ako provjera ne pokaže nikakve greške, program se unosi na pločicu mikroupravljača odabirom opcije *Upload*.



Slika 25: Odabir serijskog komunikacijskog sučelja

4.2. PROGRAMSKA PODRŠKA ZA RAD S MIKROUPRAVLJAČEM

Programiranje Arduino mikroupravljača se djelomično razlikuje od klasičnog C/C++ programiranja. Osnovna razlika je što svaki program napisan u Arduino IDE aplikaciji mora imati dvije osnovne funkcije koje izvršavaju programski kod. Te funkcije su:

– **void setup** ( ) { }

– **void loop** ( ) { }

Setup i *loop* su dvije osnovne funkcije za pisanje Arduino koda. *Void* znači da funkcije ne vraćaju vrijednosti. U funkciji *setup* definiraju se izrazi koji se očitavaju samo jednom u kodu, npr. ako je potrebno definirati neke parametre, varijable, stalne vrijednosti ili

postaviti pin mikroupravljača kao ulaz ili izlaz. U funkciji *loop* izvršava se tzv. beskonačna petlja. Petlja predstavlja glavni dio programskog koda, npr. izvršavanje zadanih funkcija poput if, while, else, if else, itd. Arduino programska podrška ima prethodno definirane funkcije koje nije potrebno kreirati, te pomažu u jednostavnijem izvršavanju koda. Osnovne funkcije su prikazane u tablici ispod:

Tablica 7: Osnovne funkcije Arduino IDE aplikacije [4]

FUNKCIJA
digitalRead ( )
digitalWrite ( )
pinMode ( )
delay( )
analogRead ( )
analogWrite ( )
Serial.print ( )
Serial.begin ( )
Serial.read ( )

DEFINICIJA OSNOVNIH ARDUINO IDE FUNKCIJA

Funkcija digitalWrite

Funkcija digitalWrite postavlja stanje određenog digitalnog pina. Postavlja pin u jedno od dva stanja:

– **HIGH** (visoko) – postavlja stanje pina na visoki napon tj. 5V ili 3.3V, ovisno o ploči

– **LOW** (nisko) – postavlja stanje pina na niski napon tj. 0V.

Prije korištenja funkcije, potrebno je postaviti odgovarajući pin kao izlaz, koristeći funkciju „pin mode“ unutar funkcije „setup ( )“. DigitalWrite se koristi samo za digitalne pinove, odnosno na analognim pinovima ne funkcionira. Ova funkcija je ključna za kontrolu digitalnih izlaza na Arduino ploči, omogućavajući jednostavno upravljanje LED diodama, relejima, motorima i drugim digitalnim komponentama. Na slici ispod je prikazan primjer jednostavnog koda za treperenje LED diode.

```
void setup() {
  // Postavljanje digitalnog pina 13 kao izlaznog
  pinMode(13, OUTPUT);
}

void loop() {
  // Uključivanje LED-ice
  digitalWrite(13, HIGH);
  delay(1000); // Čekanje 1 sekundu
  // Isključivanje LED-ice
  digitalWrite(13, LOW);
  delay(1000); // Čekanje 1 sekundu
}
```

Slika 26: Primjer koda za funkciju digitalWrite

Funkcija digitalRead

Funkcija digitalRead očitava stanje (napon) određenog digitalnog pina. Ova funkcija vraća jedno od dva stanja:

– **HIGH** (visoko) – pin je na visokom naponu (5V, 3.3V)

– **LOW** (nisko) – pin je na niskom naponu (0V)

Korištenjem ove funkcije, moguće je implementirati razne ulazne kontrole u projektima, npr. očitavanje tipkala, senzora i drugih uređaja. Na slici ispod je primjer rada funkcije. Očitavanje stanje tipkala na pinu 2 i uključivanje i isključivanje LED diode sa digitalnog pina 13.

```
void setup() {
  pinMode(2, INPUT); // Postavljanje digitalnog pina 2 kao ulaznog
  pinMode(13, OUTPUT); // Postavljanje digitalnog pina 13 kao izlaznog
}

void loop() {
  int buttonState = digitalRead(2); // Očitavanje stanja pina 2
  if (buttonState == HIGH) {
    digitalWrite(13, HIGH); // Uključivanje LED-ice
  } else {
    digitalWrite(13, LOW); // Isključivanje LED-ice
  }
}
```

Slika 27: Primjer koda za funkciju digitalRead

Funkcija pinMode

Funkcija pinMode postavlja način rada određenog digitalnog pina. Ovaj način rada može biti:

– **OUTPUT** (izlaz) – pin koji šalje signale od Arduino mikroupravljača do ostalih komponenata. Radi zajedno sa funkcijom digitalWrite, prvo se aktivira pin, a zatim se postavlja na visoki ili niski napon.
– **INPUT** (ulaz) – pin koji prima signala od ostalih komponenata na Arduino mikroupravljaču
– **INPUT_PULLUP** (ulaz s unutarnjim pull-up otpornikom) – pin koji se koristi za primanje signala s aktiviranim unutarnjim pull-up otpornikom. Pull-up otpornik je pasivna komponeta koja osigurava da logički ulaz uvijek ima definirano stanje.

Primjer korištenja funkcije na slici ispod:

```
void setup() {
  pinMode(13, OUTPUT); // Postavljanje digitalnog pina 13 kao izlaznog
  pinMode(7, INPUT); // Postavljanje digitalnog pina 7 kao ulaznog
  pinMode(2, INPUT_PULLUP); // Postavljanje digitalnog pina 2 kao ulaznog
}

void loop() {
  // Vaš glavni kod ovdje
}
```

Slika 28: Primjer rada funkcije pinMode

Funkcija delay

Funkcija „delay“ se koristi za zaustavljanje izvođenja programa na određeni vremenski interval izražen u tisućinkama sekunde. Preciznost ove funkcije može varirati, ovisno o programskom jeziku. U slučaju programiranja u Arduino IDE aplikaciji, preciznost je visoka i teško da će biti ikakvih odstupanja prilikom izvođenja. Preciznost ovisi o osnovnom taktu procesora, interupcije i preopterećenje procesora. Nedostatak ove funkcije je ta, što blokira kod. Dok je „delay“ aktivan, mikroupravljač neće moći izvršavati druge zadatke. Zato treba promišljeno koristiti ovu funkciju i u manje složenijim programima gdje se izvršava manje zadataka istovremeno.

Primjer korištenja funkcije prikazan je na slici ispod:

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);  // Postavljanje ugrađene LED diode kao izlaz
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // Uključivanje LED diode
  delay(1000);  // Čekanje 1 sekunde (1000 milisekundi)
  digitalWrite(LED_BUILTIN, LOW);  // Isključivanje LED diode
  delay(1000);  // Čekanje 1 sekunde
}
```

Slika 29: Primjer rada funkcije "delay"

Funkcije analogRead i analogWrite

Funkcije analogRead i analogWrite su ključne za rad s analogno-digitalnim i digitalno analognim pretvaračima.

Funkciju **analogRead** koristimo za čitanje analognog signala sa pina koji korisnik dodijeli. Kod Arduina to omogućava čitanje napona na analognom pinu i pretvara očitani napon u digitalnu vrijednost. Na Arduinu su analogni pinovi označeni sa slovom A. Funkcija analogRead se isključivo koristi za očitavanje analognih vrijednosti.

Funkcija **analogWrite** koristi se za postavljanje analogne vrijednosti (PWM signala na digitalni pin koji podržava PWM). Funkcija analogWrite koristi se kod simulacije analognog izlaza.

```
analogWrite(9, 128);
```

Slika 30: Primjer sintakse za analogWrite funkciju

Na slici iznad dan je primjer postavljanja PWM signala na digitalnom pinu 9 s radnim ciklusom od približno 50% (vrijednost 128 je približno 50% od maksimalne vrijednosti 255). Na slici ispod dan je primjer očitavanja vrijednosti s potenciometra spojenog na analogni pin A0 koji se koristi za kontrolu inteziteta osvjetljenja LED diode spojene na digitalni pin 9.

```
void setup() {
}

void loop() {
  int sensorValue = analogRead(A0);
  int outputValue = map(sensorValue, 0, 1023, 0, 255);
  analogWrite(9, outputValue);
  delay(10);
}
```

Slika 31: Primjer korištenja analogRead i analogWrite funkcija

Funkcija Serial.begin

Svrha funkcije Serial.begin je inicijalizacija serijske komunikacije Arduino mikroupravljača. Ključna je za postavljanje brzine prijenosa podataka (*baud rate*) između mikroupravljača i računala prilikom slanja i prijenosa informacija i kodova. Na slici ispod prikazana je sintaksa funkcije:

```
Serial.begin(speed);
```

Slika 32: Sintaksa funkcije Serial.begin

Parametar „speed“ označava brzinu prijenosa podataka u bitovima po sekundi (bps). Najčešće korištena vrijednost za veliku većinu aplikacija ja 9600 bps. Funkcija Serial.begin se mora pozvati unutar funkcije „setup()“ kako bi se serijska komunikacija inicijalizirala prije nego što započne glavni program.

Funkcije Serial.print i Serial.read

Funkcije Serial.print i Serial.read su bitne za serijsku komunikaciju, ali imaju različite svrhe. Funkcija Serial.print se koristi za prikaz podataka putem serijske veze u ASCII (engl. *American Standard Code for Information Interchange)* formatu, a funkcija Serial.read se koristi za čitanje podataka..

Funkcija **Serial.print** šalje podatke s mikroupravljača na drugi uređaj putem serijske veze. Korisna je za ispis podataka na serijski monitor (*Serial monitor*) u svrhu debugiranja (engl. *debug,* traženje problema u kodu) ili za komunikaciju s drugim serijskim uređajima. Ova funkcija sadrži dva parametra: „data“ i „format“.

- „data“ su poslani podaci u različitim formatima, tekstualno (*string*), cijele brojeve (*int*), decimalne brojeve (*float*)
- „format“ je specifikacija brojevnog sustava u kojem je broj zapisan. Ovaj parametar je nije obavezan jer aplikacije samostalno prepoznaju i preoblikuju formate brojeva, ali u slučaju da ne mogu prepoznati, korisnik definira brojevni sustav (BIN = binarni, OCT = oktalni, DEC = dekadski i HEX = heksadekadski). Na slici ispod primjer sintakse za funkciju Serial.print:

```
Serial.print(data);
Serial.print(data, format);
```

Slika 33: Sintaksa funkcije Serial.print

Funkcija **Serial.read** se koristi za čitanje podataka primljenih preko serijske veze. Ova funkcija čita jedan bajt podataka koji su stigli putem serijske veze. Komunikacija ide prema mikroupravljaču iz vanjskog izvora (npr. računalo) i funkcijom Serial.read se podaci mogu pročitati u mikroupravljaču. Na slici ispod prikazana je sintaksa za funkciju Serial.read i rad u kombinaciji sa funkcijom Serial.print i Serial.begin:

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  if (Serial.available() > 0) {
    int incomingByte = Serial.read();
    Serial.print("Primljeni bajt: ");
    Serial.println(incomingByte);
  }
}
```

Slika 34: Kombinacija funkcija Serial.print, Serial.read i Serial.begin

Na početku programskog koda prvo je potrebno navesti dodatne programske biblioteke koje se koriste u radu. Na slici ispod prikazano je navođenje knjižara za ovaj projekt:

```
#include <SPI.h>        // Potrebna za nRF24L01
#include <RF24.h>       // Biblioteka za nRF24L01

#include <Wire.h>       // Potrebna za MPU6050
#include <MPU6050.h>    // Biblioteka za MPU6050
```

Slika 35: Navođenje korištenih dodatnih programskih biblioteka koje se koriste u projektu

4.3. TESTIRANJE RADA MOTORA I KOMPONENTI

U ovom poglavlju bit će testirane komponente i motori sastavljenog malenog automobila. Provjera ispravnosti svih komponenti vrši se pokretanjem odgovarajućeg testnog programskog koda kojim će se lakše pratiti tijek događanja i redom korigirati i dorađivati kod po potrebi.

TESTIRANJE MOTORA

Motori su prema shemama ispravno spojeni i potrebno je provjeriti njihovu funkcionalnost. Provjeru je moguće provesti na dva načina:

– kratko spojiti svaki motor na bateriji (na plus i minus polove baterije sa polovima motora)

– napisati probni programski kod u Arduino IDE aplikaciji, poslati ga mikroupravljaču i korigirati po potrebi

Za ovaj projekt isprobana su oba načina radi sigurnosti i smanjenja sumnje u neispravnost komponenti ili zbog kratko ili neispravno povezanih spojeva prilikom lemljenja.

U nastavku je naveden programski kod za testiranje motora:

```
// Motor A /

const int ENA = 10;  // PWM pin za kontrolu brzine motora A
const int IN1 = 4;  // Motor A input 1
const int IN2 = 2;  // Motor A input 2

// Motor B
const int ENB = 7;  // PWM pin za kontrolu brzine motora B
const int IN3 = 6;  // Motor B input 1
const int IN4 = 5;  // Motor B input 2
```

U ovom dijelu koda inicijalizirani su input pinovi motora koji dolaze sa modula za kontrolu i upravljanje motorima L293D i njihovih aktivacijskih (enable, ENA, ENB) pinova.

```
void setup() {
  // Pinovi postavljeni kao OUTPUT
  pinMode(ENA, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(ENB, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
```

```
  // Inicijlazacija serijske komunikacije
  Serial.begin(9600);
}

void loop() {
  // Motor A i B rotiraju naprijed
  digitalWrite(ENA, HIGH);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);

  digitalWrite(ENB, HIGH);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);

delay (1000);

// Motor A i B rotiraju natrag
  digitalWrite(ENA, HIGH);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);

  digitalWrite(ENB, HIGH);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);

delay (1000);

// Motor A i B rotiraju suprotnim smjerovima za omogućavanje skretanja u desno
  digitalWrite(ENA, HIGH);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);

  digitalWrite(ENB, HIGH);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);

delay (1000);

// Motor A i B rotiraju suprotnim smjerovima za omogućavanje skretanja u lijevo
 digitalWrite(ENA, HIGH);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);

  digitalWrite(ENB, HIGH);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
```

```
delay (1000);

 }
```

Između svakog bloka izvršavanja radnji ubačena je pauza („delay“) u trajanju jedne sekunde između svake kretnje.

Maleno vozilo je uspješno testirano – sve komponente i cjelokupan sustav ispravno funkcionira. Dodatna serija testova odnosi se na sklopove predajnika (TX) i prijemnika (RX). Za uspješnu komunikaciju između prijemnog i predajnog modula za bežičnu komunikaciju NRF24L01, potrebno je u testnom programskom kodu koristiti identičnu adresu.

U nastavku je naveden programski kod za testiranje predajnika (TX):

```
#include <SPI.h>
#include "RF24.h"
#include "Wire.h"
#include "I2Cdev.h"
#include "MPU6050.h"


MPU6050 mpu;
int16_t ax, ay, az;
int16_t gx, gy, gz;
int data[2];

//8 i 9 su digitalni pinovi koji su spojeni na CE i CSN od NRF24L01 modula
RF24 radio(8,9);

//Kreirana adresa odašiljača
const uint64_t pipe = 0xE8E8F0F0E1LL;

void setup(void){
  Serial.begin(9600);
  Wire.begin();
  mpu.initialize();              //Inicijalizacija MPU-a
  radio.begin();                 //Početak komunikacije
  radio.openWritingPipe(pipe);
}

void loop(void){

  //Sljedeća funkcija kontrolira tj. prepoznaje i odašilje signale po x, y, z
osima, ovisno o pokretima ruke
  mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);

  //Po X osi, funkcija data [0] služi za naprijed i natrag
```

```
  //Po Y osi, funkcija dana [0] služi za lijevo i desno
  data[0] = map(ax, -17000, 17000, 300, 400 ); //Send X axis data
  data[1] = map(ay, -17000, 17000, 100, 200);  //Send Y axis data
  radio.write(data, sizeof(data));
}
```

Programski kod očitava mjerenja digitalnog akcelerometra i žiroskopa, kako god se pokreti ruke budu mijenjali, tako se mijenjaju i očitanja vrijednosti pozicije i nagiba objekta u odnosu na trodimenzionalni koordinatni sustav. Okretanjem ruke, očitava se promjena položaja ruke u odnosu na trodimenzionalni koordinatni sustav, pa se prema tome maleni automobil ponaša kako u kodu piše. Ako se ruka nagne u lijevo, kretat će se u lijevo itd.

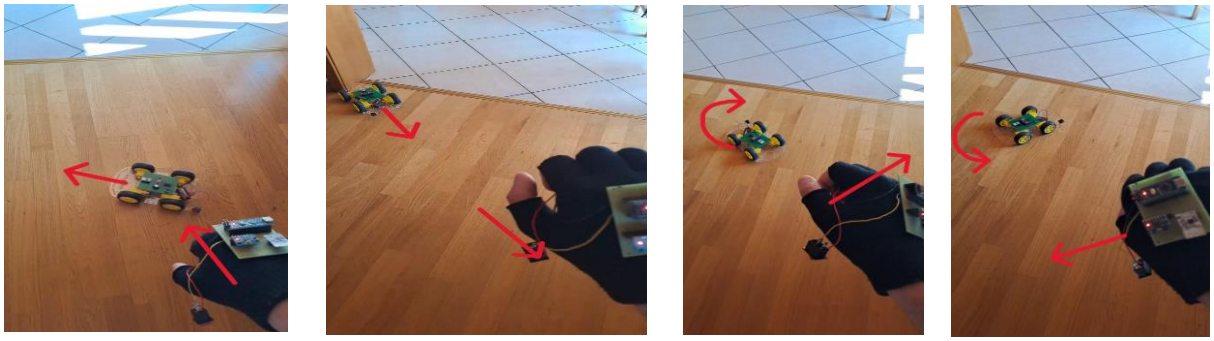
U nastavku je naveden programski kod za testiranje prijemnika (RX):

```
#include <SPI.h>
#include "RF24.h"

//Definirani pinovi za aktivaciju motora
const int enbA = 10;
const int enbB = 7;

//Definirani pinovi za plus i minus pinove motora
const int IN1 = 4;    //Right Motor (-)
const int IN2 = 2;    //Right Motor (+)
const int IN3 = 6;    //Left Motor (+)
const int IN4 = 5;    //Right Motor (-)


//Definirane varijable za brzine motora
int RightSpd = 130;
int LeftSpd = 150;

//Definirana varijabla za kretanje po X i Y osi
int data[2];

//Definirani digitalni pinovi za NRF24L01 modul
RF24 radio(8,9);

// Adresa za komunikaciju sa NRF24L01 na TX dijelu
const uint64_t pipe = 0xE8E8F0F0E1LL;

void setup(){
  //Pinovi motora definirani kao OUTPUT
  pinMode(enbA, OUTPUT);
  pinMode(enbB, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
```

```
  pinMode(IN4, OUTPUT);

  Serial.begin(9600);
  radio.begin();                      //Početak NRF24 koumunikacije
  radio.openReadingPipe(1, pipe);
  radio.startListening();
  }

void loop(){
  if (radio.available()){
    radio.read(data, sizeof(data));

    if(data[0] > 380){
      //Naprijed
      analogWrite(enbA, RightSpd);
      analogWrite(enbB, LeftSpd);
      digitalWrite(IN1, LOW);
      digitalWrite(IN2, HIGH);
      digitalWrite(IN3, LOW);
      digitalWrite(IN4, HIGH);
    }

    if(data[0] < 310){
      //Unatrag
      analogWrite(enbA, RightSpd);
      analogWrite(enbB, LeftSpd);
      digitalWrite(IN1, HIGH);
      digitalWrite(IN2, LOW);
      digitalWrite(IN3, HIGH);
 digitalWrite(IN4, LOW);
    }

    if(data[1] > 180){
      //lijevo
      analogWrite(enbA, RightSpd);
      analogWrite(enbB, LeftSpd);
      digitalWrite(IN1, LOW);
      digitalWrite(IN2, HIGH);
      digitalWrite(IN3, HIGH);
      digitalWrite(IN4, LOW);
    }
    if(data[1] < 110){
      //desno
      analogWrite(enbA, RightSpd);
      analogWrite(enbB, LeftSpd);
      digitalWrite(IN1, HIGH);
      digitalWrite(IN2, LOW);
      digitalWrite(IN3, LOW);
      digitalWrite(IN4, HIGH);
    }
```

```
    if(data[0] > 330 && data[0] < 360 && data[1] > 130 && data[1] < 160){
      //zaustavljanje / kočenje
      analogWrite(enbA, 0);
      analogWrite(enbB, 0);
    }
  }
}
```

Na kraju se vrši provjera ispravnosti smjera kretanja malenog vozila. Na ispod prikazanim slikama testiranja malenog vozila, dodatnu su naznačeni smjerovi kretanja redoslijedom (s lijeva na desno): kretanje naprijed, natrag, lijevo i desno.



Slika 36: a) naprijed, b) natrag, c) desno, d) lijevo

Testiranjem je utvrđeno da maleni automobil reagira na pokrete ruke, ali ne reagira u određenom u vremenu i u određenom položaju. Reakcija malenog automobila je prespora u svim položajima ruke. Nakon provedenih testiranja i manjih korekcija programskog koda, uspješno je povećana brzina reakcije sustava na promjenu položaja ruke uz održavanje brze i stabilne bežične veze između komponenti sustava.

5. ZAKLJUČAK

Zadatak završnog rada je izrada malenog automobila na upravljanje ručnim gestama uz pomoć mikroupravljača Arduino Nano. Kombinacijom dijelova, stečenog i primijenjenog znanja autora na ovaj završni rad, izrađen je prototip sa zadovoljavajućim rezultatima s obzirom na svrhu za koju je namijenjen, ali i kao osnova za neke buduće projekte. Prezentirani projekt uz određene dorade i preinake može biti primijenjen u različitim sustavima koji se mogu na isti ili sličan način doraditi i izgraditi. S obzirom na to da u današnje vrijeme postoje tzv. „automobili bez vozača“ ili „taxi bez vozača“ tako bi se i slično dalo napraviti i s ovim projektom. Primijeniti na automobilima normalne veličine u svrhu lakšeg parkiranja ili izlaza iz uskih prostora pokretom ruke, tj. ručnim gestama umjesto ulaženja u automobil.

Dodatno unaprjeđenje bi bilo spajanje manjih solarnih panela, korištenje samo jedne baterije bez potrebe kupovanja novih i odlaganja starih. Tako bi se baterije uvijek mogle obnavljati, što je na neki način i korisno s obzirom na okoliš. Također je tu i mogućnost spajanja i nekih drugih senzora na maleni automobil, npr. infracrveni senzor, senzor pokreta, pa bi se mogao zaustaviti sam, ako se dogodi da nailazi na nešto što mi ne vidimo s naše točke gledišta, s obzirom na to da je komunikacija bežična i nema određen domet kao sa žičanom komunikacijom.

6. LITERATURA

[1] TechTarget, Microcontroller (MCU), URL:
https://www.techtarget.com/iotagenda/definition/microcontroller

[2] Edis Techlab, Arduino Nano pinout
https://edistechlab.com/arduino-nano-pinout/?v=fd4c638da5f8

[3] Geeksforgeeks, Microcontroller and it's types, URL:
https://www.geeksforgeeks.org/microcontroller-and-its-types/

[4] Arduino Nano, URL:
https://www.conrad.hr/p/arduino-board-nano-core-nano- atmega328-1172623

[5] „ATMEGA328P pdf, ATMEGA328P Description, URL:
https://pdf1.alldatasheet.com/datasheet-pdf/view/241077/ATMEL/ATMEGA328P.html

[5] Lekšić K. *Daljinsko upravljanje robotskom rukom s NRF24,*
Osijek 2022.

[6] AranaCorp, modul NRF24L01
https://www.aranacorp.com/en/using-a-nrf24l01-module-with-arduino/

[7] Arduino Nano karakteristike, URL: https://store.arduino.cc/products/arduino-nano

[8] StarTech.com, Type C cable, URL: https://www.startech.com/en-eu/cables/rusb2ac1mb

[9] Components101, MPU6050 Accelerometer and Gyroscope Module, URL:
https://com_ponents101.com/sensors/mpu6050-module

[10] HardwareLibre, MPU6050
https://www.hwlibre.com/en/mpu6050/

[11] LastMinuteEngineers, Control DC Motors with L293D Motor Driver IC & Arduino, URL: https://lastminuteengineers.com/l293d-dc-motor-arduino-tutorial/?utm_content=cmp-true

[12] Grobotronics, L293D https://grobotronics.com/motor-driver-l293d-1a.html?sl=en

[13] KoungShun, L293D, URL: https://hr.szks-kuongshun.com/uno/uno-board-shield/l293d-motor-driver-board.html

[14] HardwareLibre, LM7805 regulator napona, URL: https://www.hwlibre.com/hr/lm7805/

[15] Components 101, LM7805 Voltage Regulator https://www.google.com/url?sa=i&url=https%3A%2F%2Fcomponents101.com%2Fics%2F7805-voltage-regulator-ic-pinout-datasheet&psig=AOvVaw1JDlu0GwobML8JzyP0wu_i&ust=1714744628903000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCID8ntKP74UDFQAAAAAdAAAAABAO

[16] 4WD Car kit, URL: https://www.amazon.in/Robocraze-4-wheel-Drive-Chassis-Encoder/dp/B077GNR6WQ

[17] Autodesk Instuctables, Schematic https://www.instructables.com/Autodesk-Tutorials/

[17] Osnovne Arduino funkcije, URL: https://www.arduino.cc/reference/en/,

[18] NYU Tandon School of Engineering, Arduino programming notebook, URL: https://www.google.com/search?q=arduino+ide+software+functions+pdf&rlz=1C1GCEA_enHR936HR936&oq=A&gs_lcrp=EgZjaHJvbWUqCAgAEEUYJxg7MggIABBFGCcYOzIOCAEQRRgnGDsYgAQYigUyBggCEEUYOzIGCAMQRRg8MgYIBBBFGDwyBggFEEUYPDIGCAYQRRg8MgYIBxBFGDzSAQgxMzY0ajBqOagCALACAQ&sourceid=chrome&ie=UTF-8

[19] Arduino Software, URL: https://docs.arduino.cc/learn/starting-guide/the-arduino-software-ide/

[20] RandomNerdTutorials, RF 433Mhz Transmitter/Receiver Module with Arduino, URL: https://randomnerdtutorials.com/rf-433mhz-transmitter-receiver-module-with-arduino/