Šolski center Celje
Srednja šola za kemijo, elektrotehniko in računalništvo

ELEKTRIČEN AVTOMOBILČEK NA DALJINSKO UPRAVLJANJE

RAZISKOVALNA NALOGA

Avtorja:
Jakob Tomaž Plevnik, E4a
David Flander, E4a

Mentor:
Gregor Kramer, univ. dipl. inž. el.

Celje, marec 2020

1 KAZALO

1.1 KAZALO SLIK

2 POVZETEK

Raziskovala sva, če je možno izdelati električen daljinsko voden avtomobilček z arduino krmilniki in če bo zmožen doseči 80km/h.
Za izdelavo avtomobilčka na daljinsko vodenje sva se odločila, ker sva videla na spletu posnetke doma narejenih avtomobilčkov iz delov naročenih preko spletnih trgovin, kateri so bili zmožni doseči visoke hitrosti. Da bi izziv povečala in stroške zmanjšala, sva se odločila, da tovarniški krmilnik za avtomobilček zamenjava z arduinom (krmilnik, ki ga sam programiraš), ki ga bova sama programirala tako, da bo pošiljal informacije za hitrost in obračanje koles preko modulov nrf24l01.
Najprej sva naredila načrt, kako naj bi vse delovalo in katere komponente potrebujeva za izdelavo izdelka. Ko sva napisala spisek komponent, sva se pozanimala, ali jih imajo v lokalnih trgovinah in če jih niso imeli, sva jih naročila iz spletnih trgovin. Postopoma sva odpravljala težave s pomočjo mentorja in sestavljala avtomobilček.
Ko sva avtomobilček končala, sva ugotovila, da ni zmožen doseči 80 km/h. Možno ga je upravljati z arduino krmilniki, le da niso tako konstantni in zanesljivi kot tovarniški krmilniki.

3 UVOD

3.1 PREDSTAVITEV RAZISKOVALNEGA PROBLEMA

Ko sva gledala video posnetke doma narejenih avtomobilčkov, sva ugotovila, da so veliko bolj zmogljivi kot tisti, ki jih lahko kupiš v trgovini in tudi cenejši, zato sva se odločila, da tudi sama narediva avtomobilček in da ugotoviva ali so arduino krmilniki dovolj dobri, da lahko pošiljajo in računajo več spremenljivk hkrati in če bo avtomobilček zmožen doseči hitrost 80 km/h.

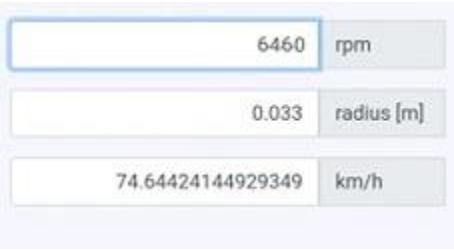
3.2 HIPOTEZE

Predvidevava, da bo avtomobilček:

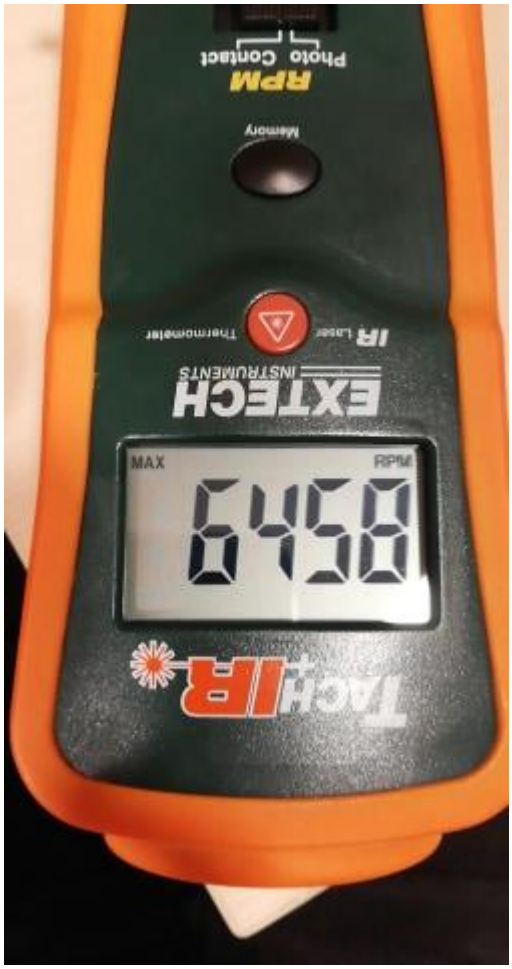
1. Sposoben doseči hitrost 80 km/h.
2. Lahko deloval z arduino krmilniki namesto tovarniškega krmilnika.

3.3 OPIS RAZISKOVALNIH METOD

Uporabila sva raziskovalne metode zbiranja statističnih podatkov, testiranje in analize dobljenih podatkov.
Pri zbiranju statističnih podatkov sva zbrala podatke o premeru koles avtomobilčka (0.066m) in vrtljaje koles avtomobilčka pri polni hitrosti (6460). Iz premera sva izračunala polmer (0.033m). Ko sva imela polmer in izmerjene vrtljaje koles, sva lahko izračunala najvišjo možno hitrost (75 km/h), ki jo lahko doseže najin avtomobilček s pomočjo spletne strani, ki je namenjena računanju hitrosti. Ker sva že dobila hitrost nižjo od hipotetične sva hipotezo ovrgla.
Pri testiranju sva samo vozila avtomobilček in opazovala, če se je obnašal kot sva predvidevala.



Slika 1: Računanje najvišje hitrosti



Slika 2: Merilec vrtljajev

4 OSREDNJI DEL NALOGE

4.1 PREDSTAVITEV REZULTATOV RAZISKOVANJA

4.1.1 Struktura raziskovalnega dela

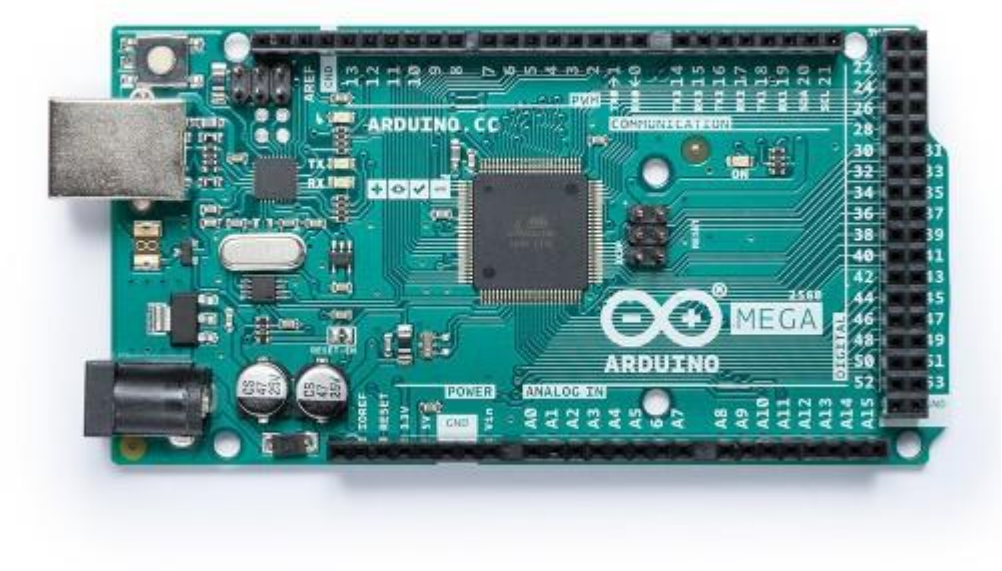
V prvem delu raziskovalne naloge sva si natančno zadala cilje in izdelala približni načrt ter spisek komponent, ki jih bova potrebovala. Večino komponent sva naročila preko spleta, arduino mega 2560 pa sva že imela od lanskoletne projektne naloge.
V drugem delu naloge sva delala na brezžični komunikaciji med arduinom nano in arduinom mega 2560. Za brezžično komunikacijo sva uporabljala modul nrf24l01 z antenami.
V tretjem delu naloge sva se osredotočila na pošiljanje vrednosti potenciometra za uravnavanje hitrosti iz arduina nano preko modula nrf24l01 na arduino mega 2560, kjer sva vrednosti preverjala tako, da sva prebrane vrednosti izpisovala v programu za programiranje arduina IDE na računalniku.
V četrtem delu naloge sva vgradila servo motor za usmerjanje avtomobilčka ter napisala program, da lahko motorček upravljamo daljinsko preko istih nrf24l01 modulov in arduinov nano in mega 2560.
V petem delu raziskovalne naloge sva vse komponente sestavila skupaj in izmerila vrtljaje.
V šestem delu naloge sva narisala, dala izdelati tiskano vezje za arduino nano, potenciometra in modem nrfl01 in zaspajkala vse te komponente na izdelano tiskano vezje.
V sedmem delu naloge sva zbirala podatke za potrditev hipotez. To vključuje testiranje vožnje avtomobilčka, kjer sva preverila ali sta krmilnika arduino zmožna zamenjati tovarniški krmilnik. Zbiranje podatkov vključuje tudi računanje hitrosti, da bi lahko potrdila ali ovrgla hipotezo, ki trdi, da bo avtomobilček sposoben doseči 80 km/h.

4.1.2 Opis komponent

4.1.2.1 Arduino mega 2560

Je mikrokrmilnik namenjen krmiljenju manjših enostavnejših vezji, ki ne potrebujejo visoke natančnosti. Večinoma se uporablja za učenje programiranja učencev elektrotehnike in mehatronike, uporabljajo ga tudi ljudje, ki se ukvarjajo z elektroniko kot konjičkom.
Mikrokrmilnik programiramo s programom imenovanim Arduino IDE, ki je narejen prav za ta mikrokrmilnik.
Vgrajen ima čip ATmega 2560, ki obratuje s frekvenco 16 MHz. Ima 54 digitalnih vhodno-izhodnih priključkov. 14 priključkov podpira pulzno širinsko modulacijo, 16 analognih vhodov in 9 od teh lahko deluje kot izhod.
Mikrokrmilnik obratuje na napetosti 5 V. Napajamo ga lahko z od 6 V do 20 V, ampak bolj priporočljivo je, da ga napajamo z od 7V do 12 V. Napajamo ga lahko preko USB

priključka ali DC priključka dimenzij 2.1 mm. Na voljo pa ima tudi dva napajalna priključka 5 V in 3,3 V.
Na posameznem digitalnem priključku je maksimalen tok 40 mA, pri priključku 3,3 V pa 50 mA. Zgornja meja skupnega izhodnega toka pa ne sme preseči 200 mA. Za nujne primere je vgrajena tudi varovalka, ki mikrokrmilnik izklopi pri 500 mA in vklopi nazaj šele, ko tok pade pod 500 mA.
Ima flash pomnilnik velikosti 256 KB (od tega 8 KB uporablja zagonski nalagalnik), SRAM 8 KB in EEPROM 4 KB.
Vgrajeno ima tudi tipko Reset, ki ustavi program in ga ponovno zažene, ter diodo za zaščito pred napačno polariteto.

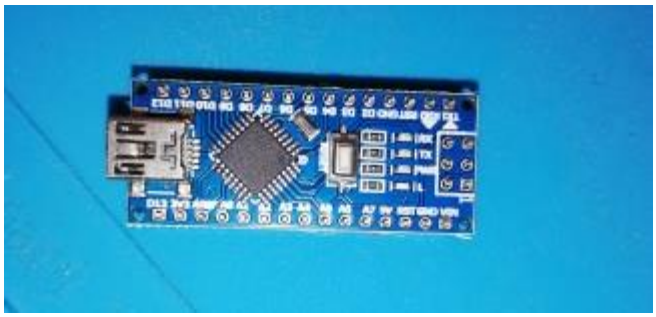


Slika 3: Arduino mega 2560

4.1.2.2 Arduino nano

Je mikrokrmilnik namenjen krmiljenju manjših enostavnejših vezji, ki ne potrebujejo visoke natančnosti. Večinoma se uporablja za učenje programiranja učencev elektrotehnike in mehatronike, uporabljajo ga tudi ljudje, ki se ukvarjajo z elektroniko kot konjičkom.
Mikrokrmilnik programiramo s programom imenovanim Arduino IDE, ki je narejen prav za ta mikrokrmilnik.
Vgrajen ima čip ATmega328, ki obratuje s frekvenco 16 MHz. Ima 14 digitalnih vhodno-izhodnih priključkov. 6 priključkov podpira pulzno širinsko modulacijo in 8 analognih vhodov.
Mikrokrmilnik obratuje na napetosti 5 V. Napajamo ga lahko z od 6 V do 20 V, ampak bolj priporočljivo je, da ga napajamo z od 7 V do 12 V. Napajamo ga lahko preko USB priključka ali DC priključka 27.

Ima flash pomnilnik velikosti 32 KB (od tega 2 KB uporablja zagonski nalagalnik), SRAM 2 KB in EEPROM 1 KB.
Vgrajeno ima tudi tipko Reset.



Slika 4: Arduino nano

4.1.2.3 Antena nrf24l01

Nrf24l01 je radijski sprejemnik z enim čipom za svetovni pas 2.4-2.5 GHz. Uporablja se za brezžično komunikacijo med krmilnikoma ali pri brezžični tipkovnici ali miški. Obratuje na napetosti 3,3 V in poraba toka se giba okoli 11,3 mA in 13,5 mA. Ima 8 vhodov/izhodov to so:

- GND - 0 V,
- VCC - 3,3 V,
- CE - vklopi RX (način prejemnika) ali TX (način pošiljatelja) način,
- CSN - izbira med čipi,
- SCK - Serijska ura,
- MOSI - Glavni izhod, podrejeni vhod,
- MISO - Glavni vhod, podrejeni izhod,
- IRQ - Zahteva za prekinitev brezžične komunikacije.

Slika 5: Nrf24l01 modul z anteno

4.1.2.4 Baterija

Uporablja se za napajanje elektromotorja za pogon in je narejena iz Li - litija, Fe - železa, P - fosforja in O_4 - kisika. Ima 4200 mAh in je 3S klase.

Slika 6: Baterija

4.1.2.5 Elektromotor

Elektromotor uporabljava za pogon avtomobilčka. Uporabila sva 4 polni, vodoodporni, 4300 KV, brezkrtačni elektromotor. Ima 360 A zagonskega toka in 60 A toka pri povprečni obremenitvi. Za napajanje potrebuje 12 V.

Slika 7: Elektromotor

4.1.2.6 Ogrodje

Večina ogrodja je narejena iz plastike in aluminija. Velikost ogrodja je 82 mm dolžine, 65 mm širine in 30 mm višine. Ogrodje sva morala sestavi sama, saj sva ga kupila ločeno po delih. Ogrodje je narejeno tako, da se elektromotor privije nanj in preko prenosa pogon pride na vsa štiri kolesa. Za baterijo je namenjen poseben prostor, saj je občutljiva na udarce in vbodnine.

Slika 8: Ogrodje

4.1.2.7 Servo motor

Uporabila sva ga za usmerjanje avtomobilčka levo in desno. Moč servo motorja je 4 kg in obratuje na 5 V. Upravljava ga z vrednostmi, ki jih prebereva z drugega potenciometra. Te vrednosti se gibajo med 0 in 1023, zato jih morava pretvoriti v vrednosti, ki jih potrebuje servo motor, kar so vrednosti od 0 do 180.
Servo motor je rotacijski aktuator ali linearni aktuator, ki omogoča natančno upravljanje kotnega ali linearnega položaja, hitrosti in pospeška. Servomotorji niso poseben razred motorjev, čeprav se izraz servomotor pogosto uporablja za označevanje motorja, ki je primeren za uporabo v krmilnem sistemu z zaprto zanko. Servomotorji se uporabljajo v aplikacijah, kot so robotika, CNC stroji ali avtomatizirana proizvodnja, kjer je potrebna natančnost in moč.

Slika 9: Servo motor

4.1.2.8 ESC

ESC uporabljava za uravnavanje hitrosti elektromotorja, vodiva pa ga s potenciometrom, katerega vrednosti bereva z arduinom nano. Te vrednosti pošljeva na arduino mega 2560 preko nrf24l01 modulov.

Elektronski nadzor hitrosti ali ESC je elektronsko vezje, ki nadzoruje in uravnava hitrost elektromotorja. Omogoča lahko tudi vzvratno gibanje motorja in dinamično zaviranje. Elektronski nadzor hitrosti se uporablja v radijsko krmiljenih modelih na električni pogon.

Slika 10: ESC

4.1.2.9 Potenciometer “joystick“

Uporabila sva potenciometra narejena posebej za izdelavo daljinskih upravljalnikov. Trakšne vrste potenciometrov se uporabljajo tudi na daljinskih upravljalnikih igralnih konzol. Lahko bi uporabila samo en potenciometer, ampak sva se zaradi lažje upravljivosti odločila, da uporabiva dva.
Potenciometer “joystick“ je komponenta, s katero vnašamo x in y mere ter 0/1 vrednosti. Lahko se šteje za kombinacijo potenciometrov in enega gumba. Tip podatkov dimenzije x, y so analogni vhodni signali, gumba pa digitalni vhodni signal. Ima 5 priključkov:

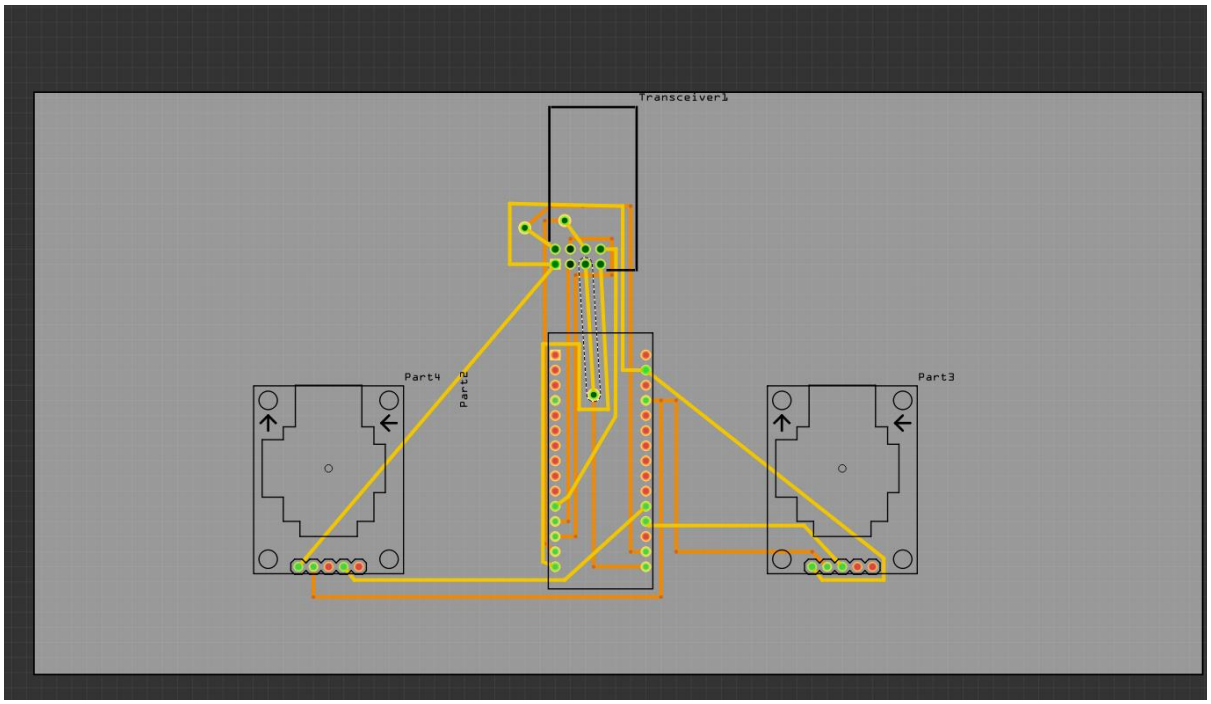
- Vcc (5V),
- Gnd (0V),
- Vrx (X vrednosti),
- Vry (Y vrednosti),
- SW (gumb).

Slika 11: Potenciometer

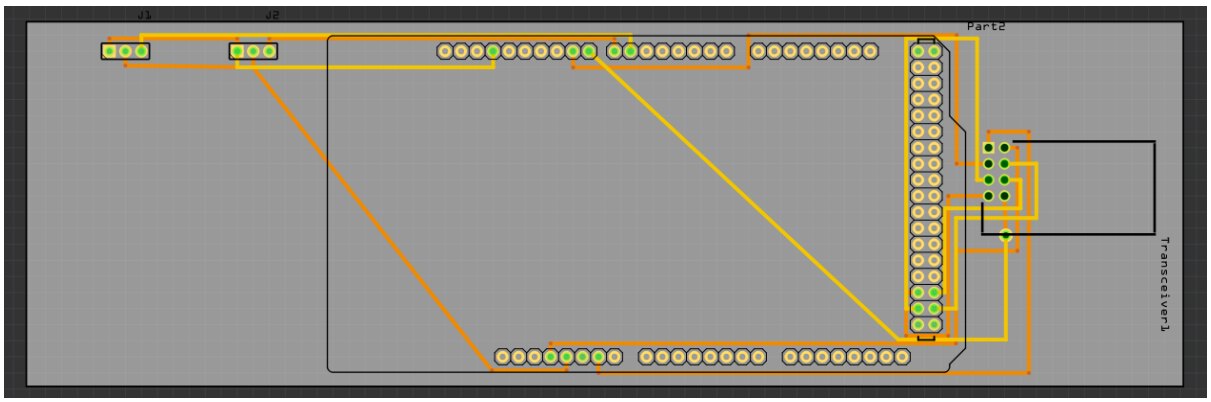
4.1.3 Izdelava tiskanih vezij

Potrebovala sva dve tiskani vezji, eno da poveževa arduino nano, dva potenciometra, modul nrf24l01 in baterijo in drugo vezje, da poveževa ESC, servo, modul nrf24l01, arduino mega 2560 in baterijo za napajanje krmilnika arduina mega 2560.

Vezje sva izdelala v programu fritzing, kjer sva našla vse komponente v njihovi knjižnici komponent in jih povezala. Tiskana vezja sva dala izdelati v naši šoli. Ko so bila vezja končana, sva vse komponente zaspajkala na tiskana vezja in jih pritrdila na ogrodje.



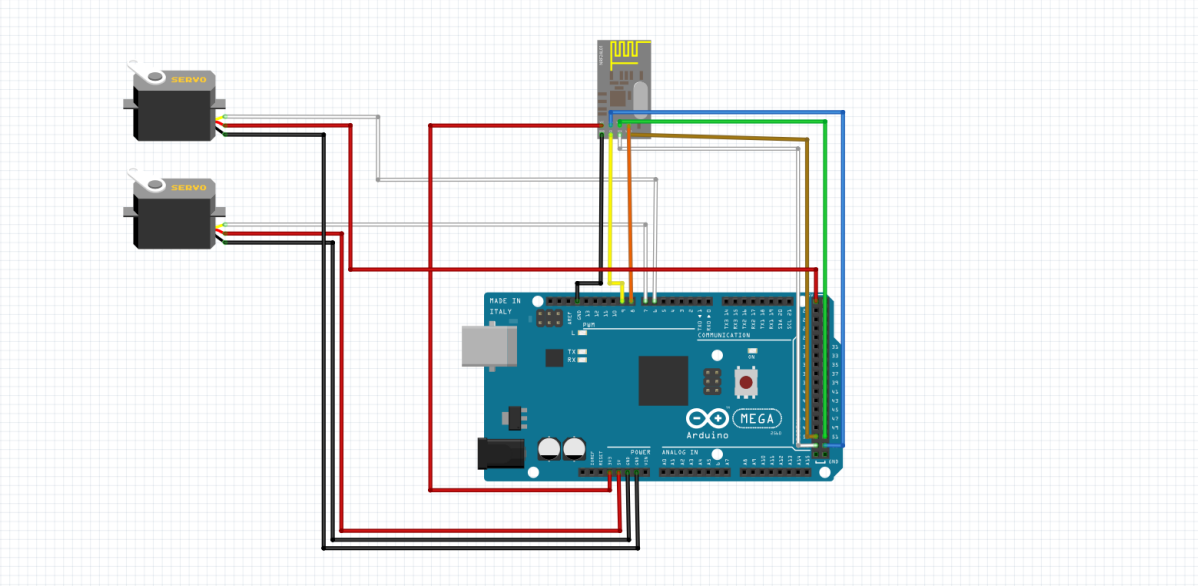
Slika 12: Načrt za tiskano vezje arduina nano, potenciometrov in nrf24l01 modula



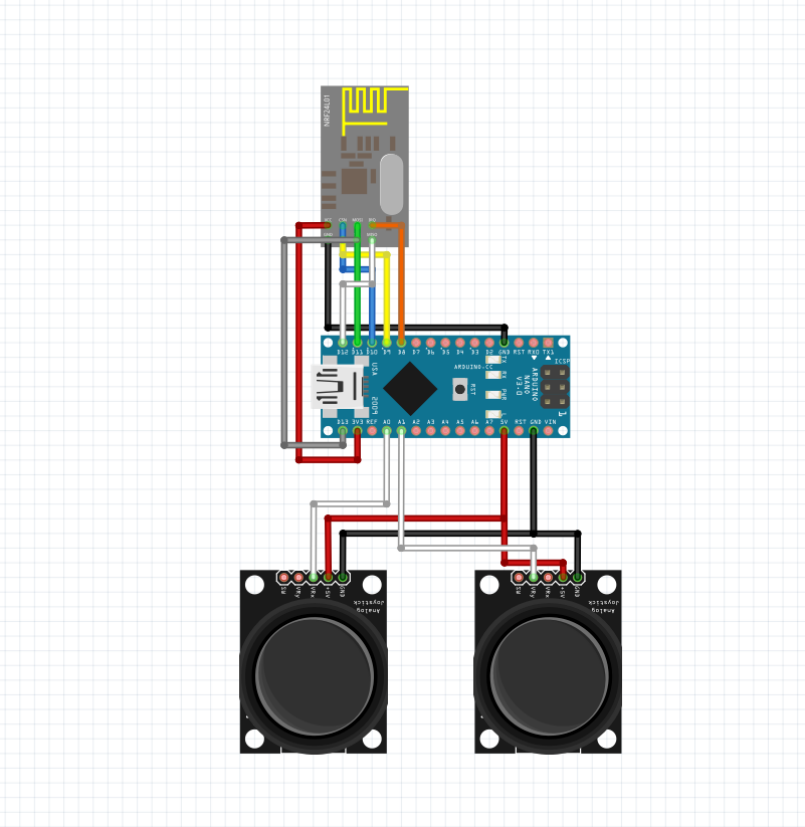
Slika 13: Načrt za tiskano vezje arduina mega 2560 in nrf24l01 modula

4.1.4 Vezalni načrt

Vezalni načrt sva narisala, da sva se bolje znašla pri vseh komponentah in da nama ni bilo potrebno razmišljati kateri vodnik sodi na določen priključek.



Slika 14: Vezalni načrt za arduina mega 2560



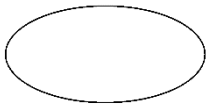
Slika 15: Vezalni načrt za arduino nano

4.1.4 Vizualni prikaz programa

Ko sva pripravljala načrt za programiranje, sva se odločila, da najprej napiševa vse postopke, za katere sva predvidevala, da jih bova potrebovala in tako sva dobila boljši pregled nad delom, ki ga je bilo potrebno opraviti. Da bi dobila še lažjo berljivost in

preglednost, sva se odločila, da celoten postopek napiševa v obliki diagrama poteka (flowchart), ker se nama je zdela ta oblika najpreglednejša. Ker imava dva različna programa (enega za arduino nano, ki pošilja in enega za arduino mega 2560, ki sprejema informacije), sva naredila dva diagrama poteka.
Diagram poteka grafično prikaže vsak korak, ki se odvije v programu. Sestavljen je iz simbolov (oblik), vsak simbol ima svoj pomen. Simboli so povezani s puščicami, da si lažje predstavljamo smer poteka programa.
Pri diagramu poteka uporabljamo 4 različne oblike: elipsa, pravokotnik, paralelogram in romb.

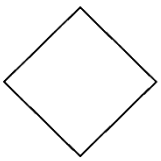
1. Za začetek in konec programa se uporablja elipsa.



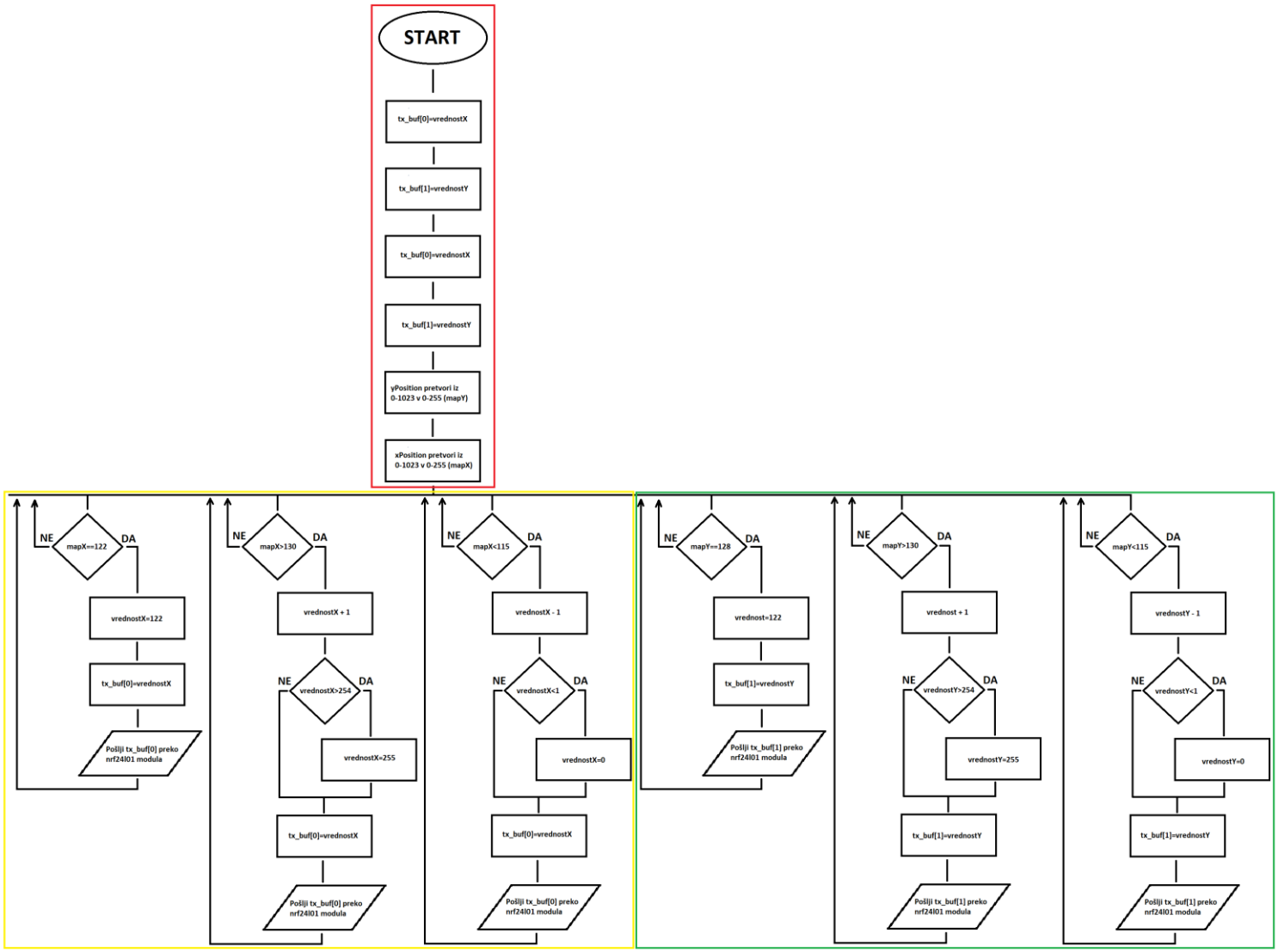
2. Izhodne in vhodne operacije kot so branje ali izpisovanje predstavljamo s paralelogramom.

3. Računske operacije in podprogrami so predstavljeni s pravokotniki.

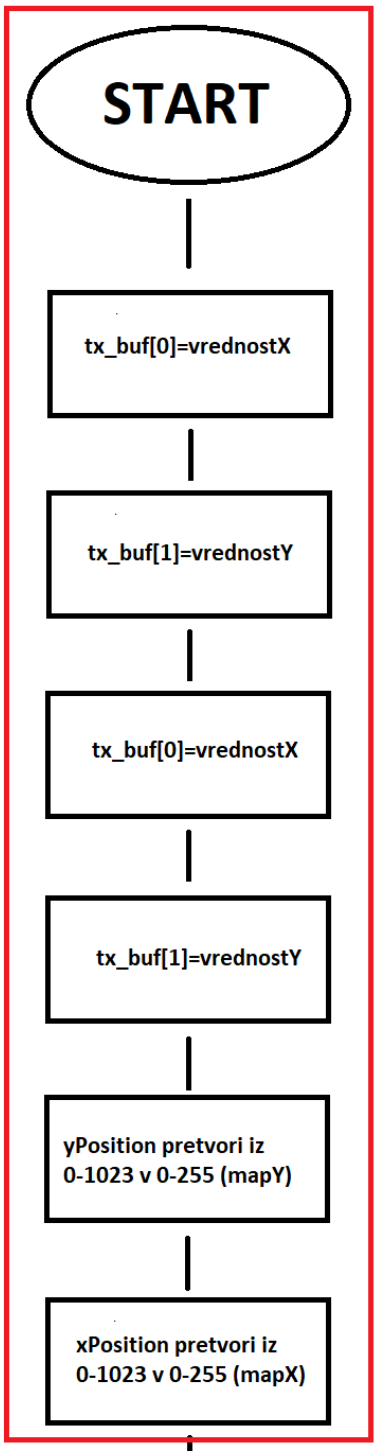
4. Odločitve, ki imajo dva možna izida so predstavljene z rombom.



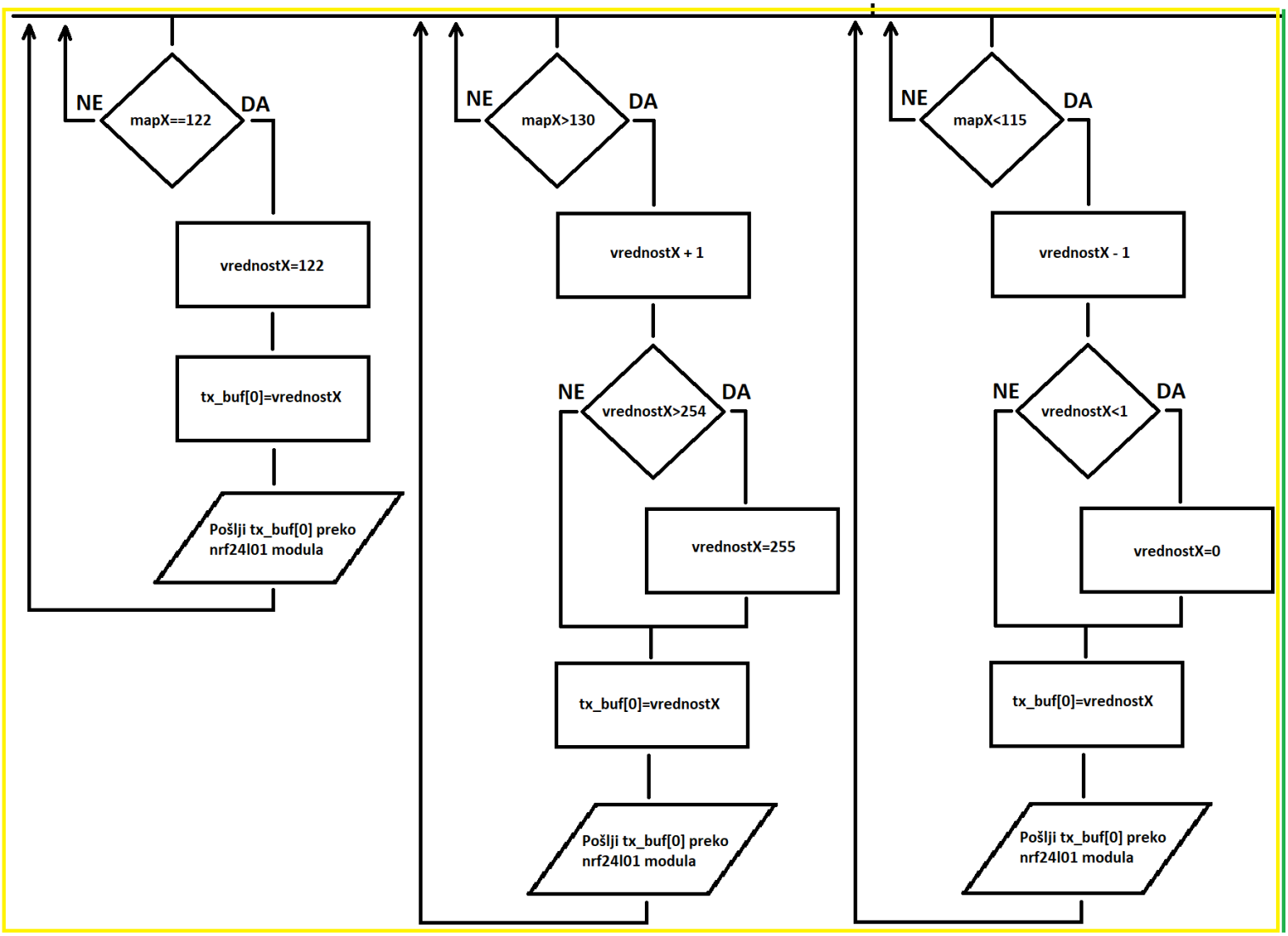
5. Smer poteka programa je predstavljena s puščico.



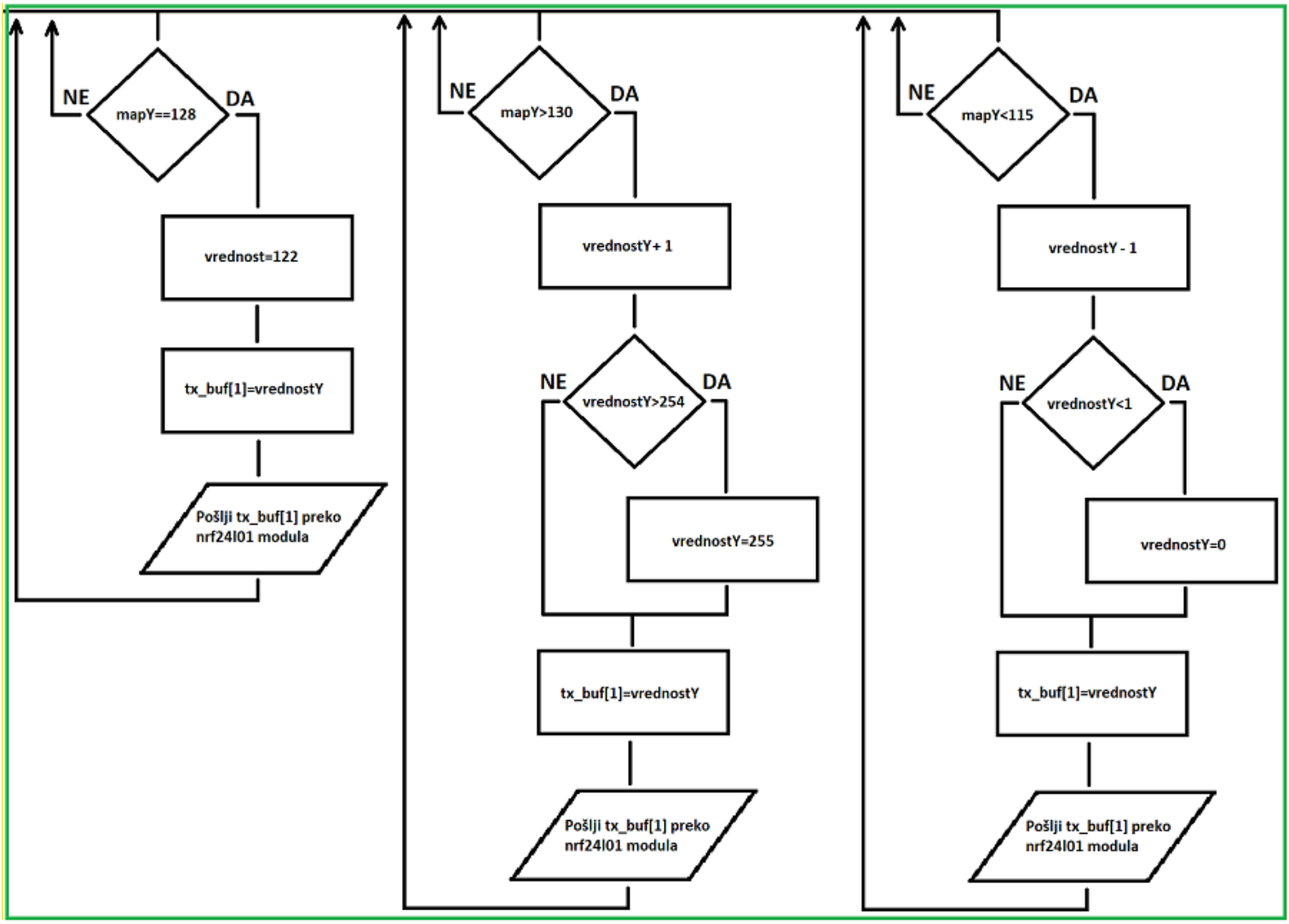
Slika 16: Diagram poteka arduino nano



Slika 17: Diagram poteka arduino nano 1

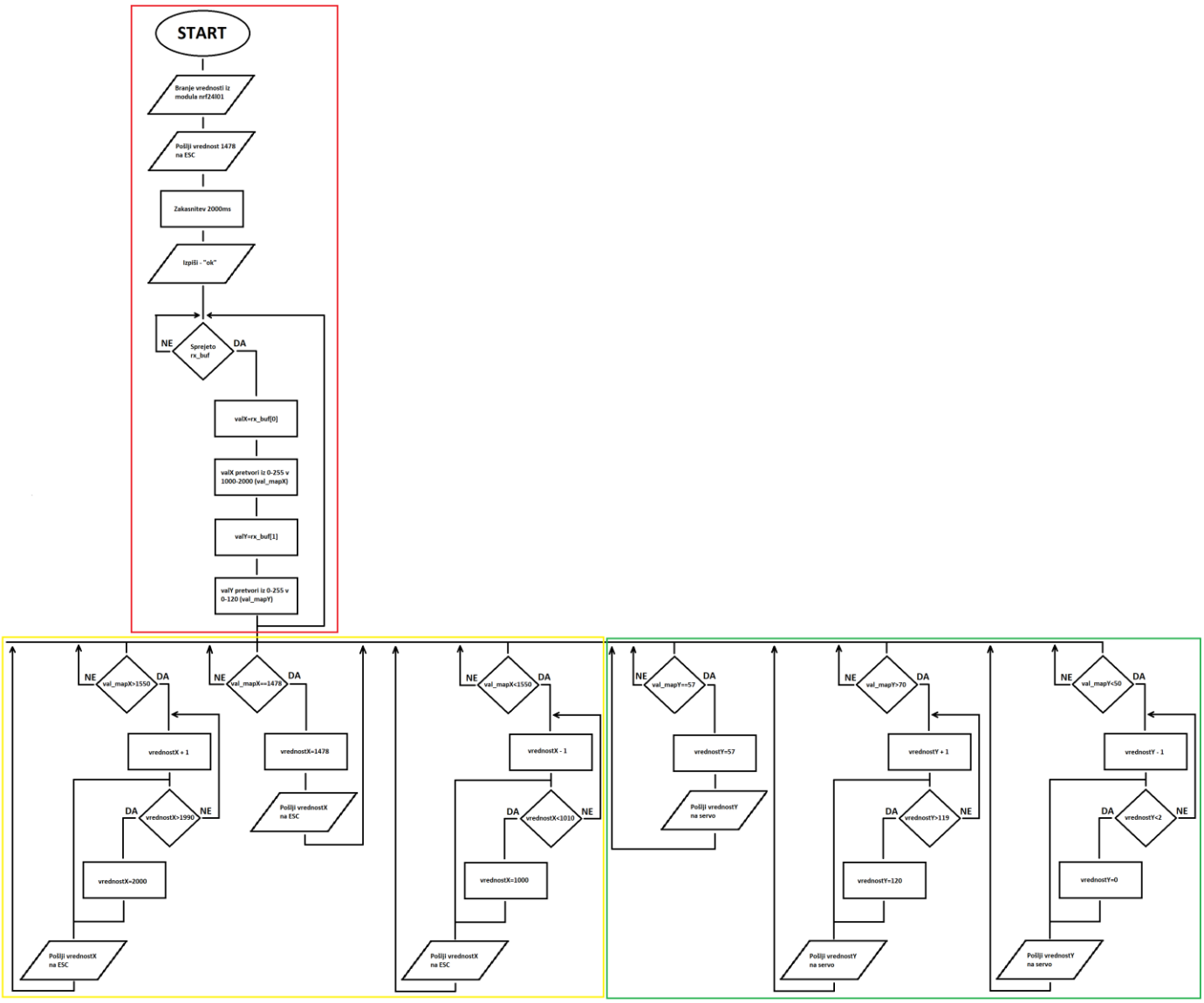


Slika 18: Diagram poteka arduino nano 2

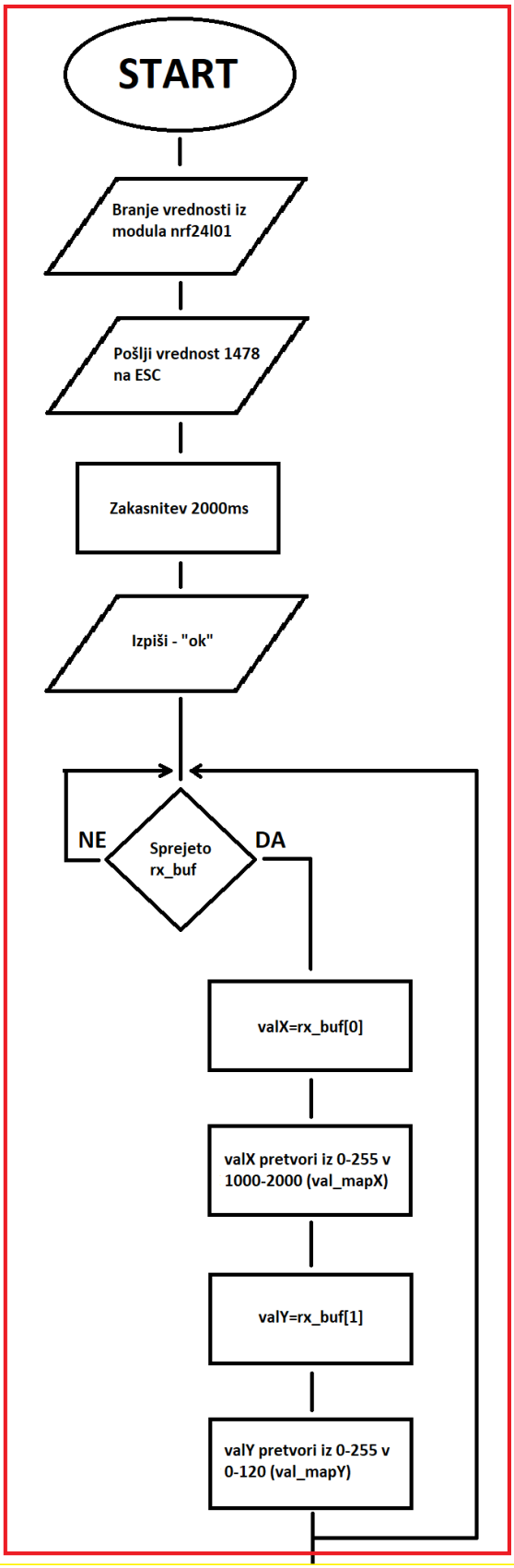


Slika 19: Diagram poteka arduino nano 3

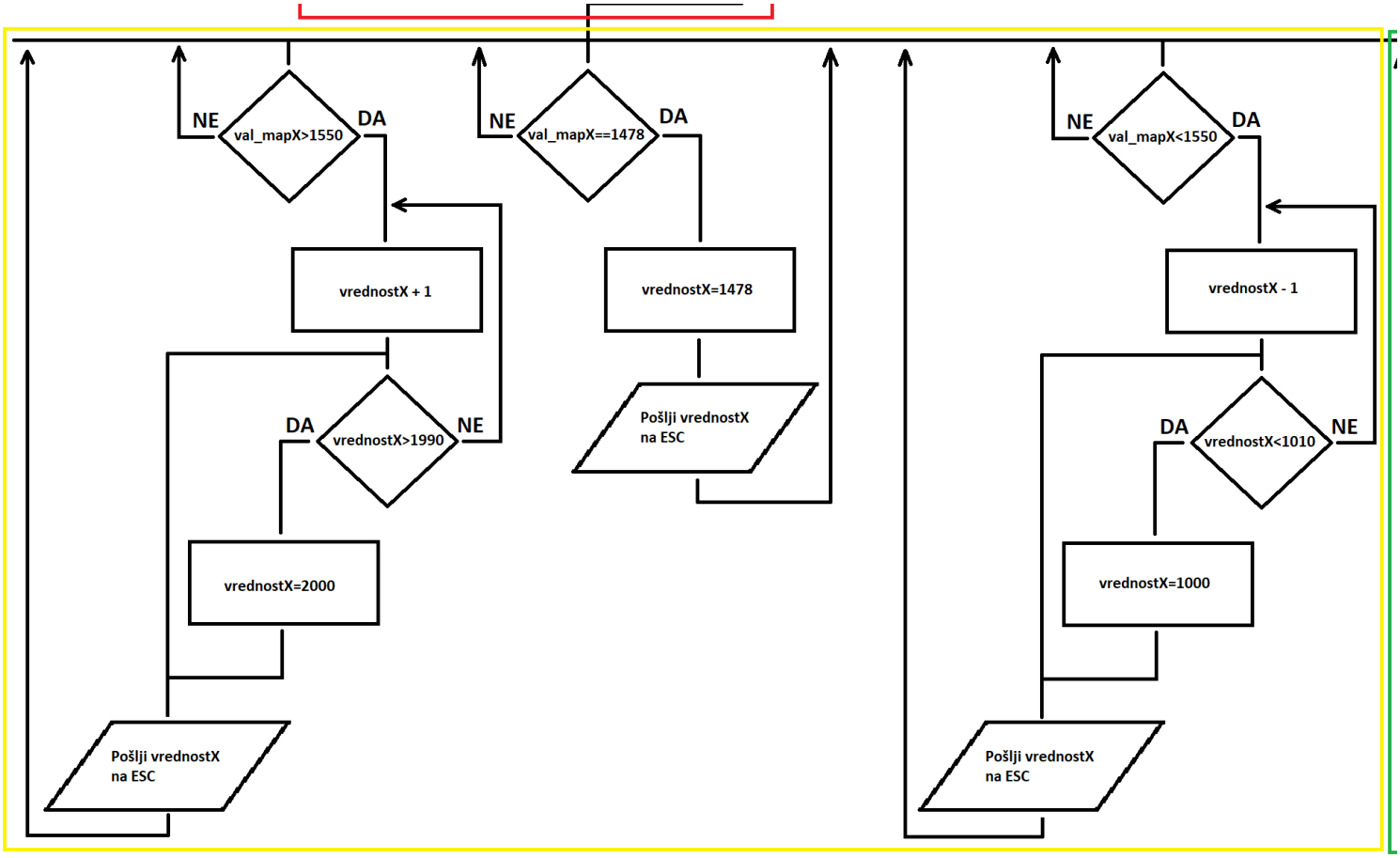
Program arduina nano (pošiljatelj) na začetku pretvori vrednost vrednostX v tx_buf[0], vrednost vrednostY v tx_buf[1] in nato v void loopu oba ukaza ponovi, v void loopu pretvori še vrednosti yPosition iz vrednosti med 0 in 1023 v sorazmerne vrednosti med 0 in 255 in novo spremenljivko poimenuje mapY in xPosition iz vrednosti med 0 in 1023 v sorazmerne vrednosti med 0 in 255 in novo spremenljivko poimenuje mapX. V void lupu se nato zvrsti šest enakovrednih if zank.
Prva zanka preverja, če je vrednost mapX enaka 122. Če je to res, v spremenljivko vrednostX zapiše 122 in vrednost vrednostX prepiše v tx_buf[0] in to vrednost pošlje preko nrf24l01 na arduino mega 2560, če pa ni pa program nadaljuje na naslednjo if zanko.
Druga zanka preverja, če je vrednost mapX večja od 130. Če je to res, se začne izvajati for zanka, ki vsak krog prišteje 1 vrednosti vrednostX. Znotraj te if zanke je še ena if zanka, ki preverja, če je vrednostX večja od 254, če je v spremenljivko vrednostX zapiše 255, novo vrednost vrednostX prepiše v vrednost tx_buf[0] in pošlje vrednost tx_buf[0] preko nrf24l01 modula na arduino mega 2560, če pa vrednost vrednostX ne izpolnjuje teh pogojev program pošlje preko nrf24l01 modula vrednost vrednostX, ki smo ji prišteli 1.
Tretja zanka preverja, če je vrednost mapX manjša od 115. Če je to res, se začne izvajati for zanka, ki vsak krog odšteje 1 vrednosti vrednostX. Znotraj te if zanke je še ena if zanka, ki preverja, če je vrednostX manjša od 1, če je v spremenljivko vrednostX zapiše 0, novo vrednost vrednostX prepiše v vrednost tx_buf[0] in pošlje vrednost tx_buf[0] preko nrf24l01 modula na arduino mega 2560, če pa vrednost vrednostX ne izpolnjuje teh pogojev program pošlje preko nrf24l01 modula vrednost vrednostX, ki smo ji odšteli 1.
Četrta zanka preverja, če je vrednost mapY enaka 128. Če je to res, se v spremenljivko vrednostY zapiše 122 in vrednost vrednostX prepiše v tx_buf[1] in to vrednost pošlje preko nrf24l01 na arduino mega 2560, če pa ni pa program nadaljuje na naslednjo if zanko.
Peta zanka preverja, če je vrednost mapY večja od 130. Če je to res, se začne izvajati for zanka, ki vsak krog prišteje 1 vrednosti vrednostY. Znotraj te if zanke je še ena if zanka, ki preverja, če je vrednostY večja od 254, če je v spremenljivko vrednostX zapiše 255, novo vrednost vrednostY prepiše v vrednost tx_buf[1] in pošlje vrednost tx_buf[1] preko nrf24l01 modula na arduino mega 2560, če pa vrednost vrednostY ne izpolnjuje teh pogojev program pošlje preko nrf24l01 modula vrednost vrednostY, ki smo ji prišteli 1.
Šesta zanka preverja, če je vrednost mapY manjša od 115. Če je to res, se začne izvajati for zanka, ki vsak krog odšteje 1 vrednosti vrednostY. Znotraj te if zanke je še ena if zanka, ki preverja, če je vrednostY manjša od 1, če je v spremenljivko vrednostY zapiše 0, novo vrednost vrednostY prepiše v vrednost tx_buf[1] in pošlje vrednost tx_buf[1] preko nrf24l01 modula na arduino mega 2560, če pa vrednost vrednostY ne izpolnjuje teh pogojev program pošlje preko nrf24l01 modula vrednost vrednostY, ki smo ji odšteli 1.



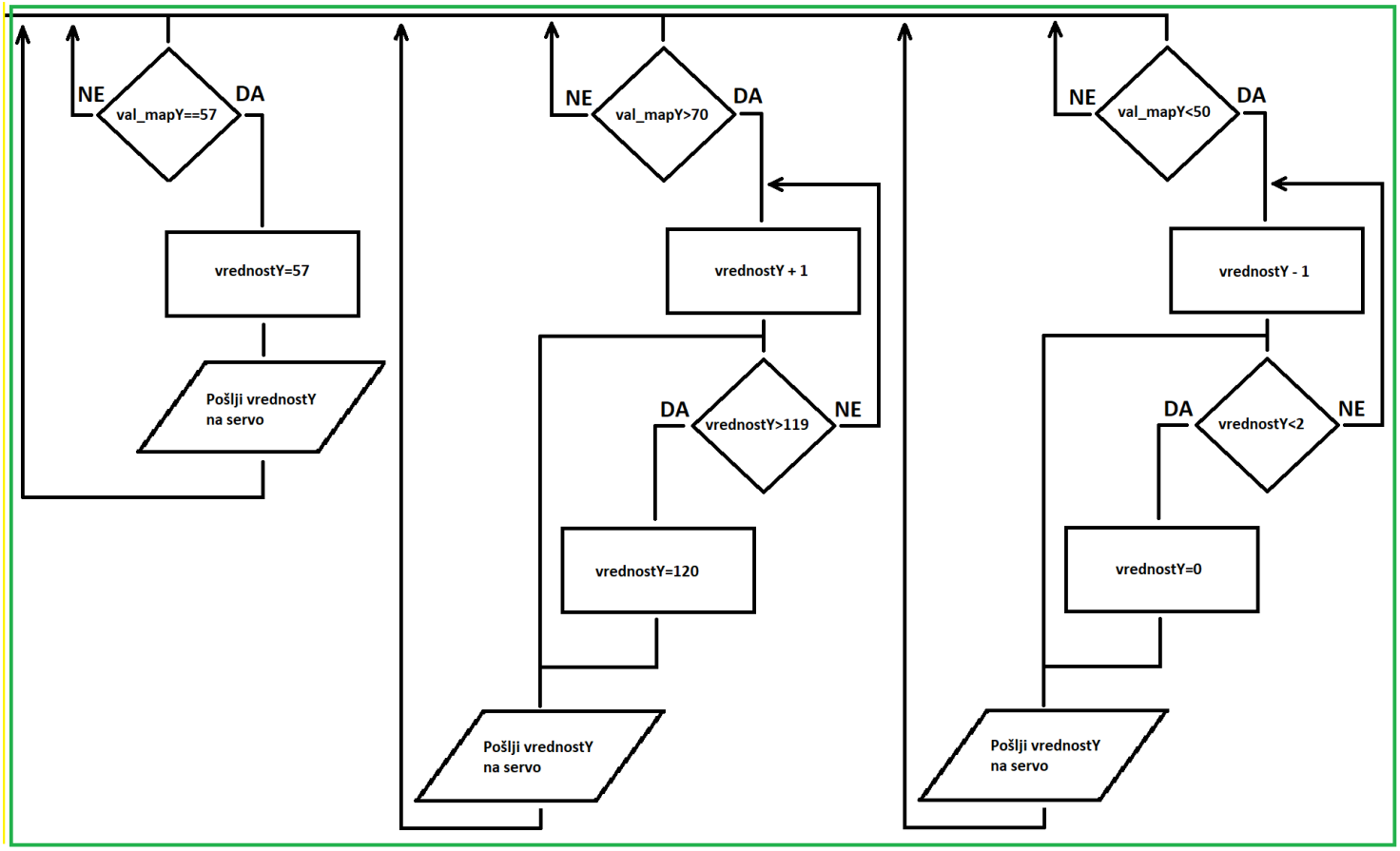
Slika 20: Diagram poteka arduino mega 2560



Slika 21: Diagram poteka arduino mega 2560 1



Slika 22: Diagram poteka arduino mega 2560 2



Slika 23: Diagram poteka arduino mega 2560 3

Programiranje arduina mega 2560 (sprejemnika). Program začne z branjem vrednosti iz nrf24l01 modula, nato pošlje vrednost 1478 na ESC in izpiše “ok“ na ekranu v programu Arduino IDE. Sledi if zanka, ki preverja če je bila sprejeta kakšna rx_buf vrednost. Če je vrednost bila sprejeta, vrednost rx_buf[0] prepiše v valX in to vrednost pretvori iz vrednosti med 0 in 255 v sorazmerne vrednosti med 1000 in 2000 in novo spremenljivko poimenuje val_mapX. Takoj za tem vrednost rx_buf[1] prepiše v valY in to vrednost pretvori iz vrednosti med 0 in 255 v sorazmerne vrednosti med 0 in 120 in novo spremenljivko poimenuje val_mapY. Za tem se začne izvajati void loop, kjer si sledi šest enakovrednih if zank.
Prva zanka preverja, če je vrednost val_mapX večja od 1550. Če je to res, se začne izvajati for zanka, ki vsak krog prišteje 1 vrednosti vrednostX. Za tem sledi še ena if zanka, ki preverja, če je vrednost vrednostX večja od 1990, če je to vrednost spremeni v 2000 in jo pošlje na ESC, če pa ni pa pošlje vrednost vrednostX, ki smo ji prišteli 1 in program nadaljuje na naslednji od enakovrednih if zank.
Druga zanka preverja ali je val_mapX enak 1478. Če je to res, 1478 prepiše v vrednost vrednostX in jo pošlje na ESC, če se pa val_mapX ne ujema s 1478 pa program nadaljuje na naslednjo if zanko.
Tretja zanka preverja, če je vrednost val_mapX manjša od 1550. Če je to res, se začne izvajati for zanka, ki vsak krog odšteje 1 vrednosti vrednostX. Za tem sledi še ena if zanka, ki preverja, če je vrednost vrednostX manjša od 1010, če je to vrednost spremeni v 1000 in jo pošlje na ESC, če pa ni pa pošlje vrednost vrednostX, ki smo ji prišteli 1 in program nadaljuje na naslednji od enakovrednih if zank.
Četrta zanka preverja ali je val_mapY enak 57. Če je to res, 57 prepiše v vrednost vrednostY in jo pošlje na servo motor, če se pa val_mapY ne ujema s 57 pa program nadaljuje na naslednjo if zanko.
Peta zanka preverja, če je vrednost val_mapY večja od 70. Če je to res, se začne izvajati for zanka, ki vsak krog prišteje 1 vrednosti vrednostY. Za tem sledi še ena if zanka, ki preverja, če je vrednost vrednostY večja od 119, če je to vrednost spremeni v 120 in jo pošlje na servo motor, če pa ni pa pošlje vrednost vrednostY, ki smo ji prišteli 1 in program nadaljuje na naslednji od enakovrednih if zank.
Šesta zanka preverja, če je vrednost val_mapY manjša od 50. Če je to res, se začne izvajati for zanka, ki vsak krog odšteje 1 vrednosti vrednostY. Za tem sledi še ena if zanka, ki preverja, če je vrednost vrednostY manjša od 2, če je to vrednost spremeni v 0 in jo pošlje na servo motor, če pa ni pa pošlje vrednost vrednostY, ki smo ji prišteli 1 in program nadaljuje na naslednji od enakovrednih if zank.

4.1.5 Programiranje

4.1.5.1 Programiranje arduina nano

Program s katerim sva programirala arduino nano je arduino IDE, kjer se uporablja programski jezik C++. Arduino nano je namenjen pošiljanju informacij preko nrf24l01 anten na arduino mega 2560 in branju vrednosti (yPosition in xPosition) iz 1. in 2. potenciometra. S potenciometrom 1 določamo hitrost in smer vrtenja elektromotorja s potenciometrom 2 pa zasuk servo motorja.

V programu sva najprej določila konstanti X_pin=0 in Y_pin=1. Takoj zatem določimo še 9 spremenljivk: VRx=A0, VRy=A1, tester=0, vrednostX=122, vrednostY=122, xPosition=0, yPosition=0, mapX=0, mapY=0.
V void setupu določiva Serial begin na 9600, priključka VRx in VRy nastaviva kot vhoda, vrednosti vrednostX in vrednostY prepiševa v spremenljivki tx_buf[0] in tx_buf[1] v tem zaporedju, na koncu izpiševa še "TX_Mode Start" na zaslon v programu Arduino IDE.
V void loopu ponovno vrednosti vrednostX in vrednostY prepiševa v spremenljivki tx_buf[0] in tx_buf[1] v tem zaporedju in nastaviva, da bere iz priključkov VRx in VRy in vrednosti iz VRx vpisuje v spremenljivko xPosition, VRy pa v spremenljivko yPosition. Vrednosti xPosition in yPosition obsegata vrednosti med 0 in 1023. Ker so te vrednosti prevelike, da bi jih lahko poslala preko modulov, jih v naslednjem koraku pretvoriva xPosition v vrednosti med 0 in 255 (mapX) in yPosition v vrednosti med 0 in 255 (mapY).
Za tem si sledi šest enakovrednih if zank, ki sva jih dodala zaradi boljše natančnosti potenciometrov in da sva preprečila hitre skoke vrednosti pri nepazljivem ravnanju s potenciometri.
Prva zanka preverja, če je vrednost mapX enaka 122. Če je, v spremenljivko vrednostX zapiše 122 in vrednost vrednostX prepiše v tx_buf[0] in to vrednost pošlje preko nrf24l01 na arduino mega 2560, če pa ni pa program nadaljuje na naslednjo if zanko.
Druga zanka preverja, če je vrednost mapX večja od 130. Če je, se začne izvajati for zanka, ki vsak krog prišteje 1 vrednosti vrednostX. Znotraj te if zanke je še ena if zanka, ki preverja, če je vrednostX večja od 254, če je v spremenljivko vrednostX zapiše 255, novo vrednost vrednostX prepiše v vrednost tx_buf[0] in pošlje vrednost tx_buf[0] preko nrf24l01 modula na arduino mega 2560, če pa vrednost vrednostX ne izpolnjuje teh pogojev program pošlje preko nrf24l01 modula vrednost vrednostX, ki smo ji prišteli 1.
Tretja zanka preverja, če je vrednost mapX manjša od 115. Če je, se začne izvajati for zanka, ki vsak krog odšteje 1 vrednosti vrednostX. Znotraj te if zanke je še ena if zanka, ki preverja, če je vrednostX manjša od 1, če je v spremenljivko vrednostX zapiše 0, novo vrednost vrednostX prepiše v vrednost tx_buf[0] in pošlje vrednost tx_buf[0] preko nrf24l01 modula na arduino mega 2560, če pa vrednost vrednostX ne

izpolnjuje teh pogojev program pošlje preko nrf24l01 modula vrednost vrednostX, ki smo ji odšteli 1.
Četrta zanka preverja, če je vrednost mapY enaka 128. Če je, v spremenljivko vrednostY zapiše 122 in vrednost vrednostX prepiše v tx_buf[1] in to vrednost pošlje preko nrf24l01 na arduino mega 2560, če pa ni pa program nadaljuje na naslednjo if zanko.
Peta zanka preverja, če je vrednost mapY večja od 130. Če je, se začne izvajati for zanka, ki vsak krog prišteje 1 vrednosti vrednostY. Znotraj te if zanke je še ena if zanka, ki preverja, če je vrednostY večja od 254, če je v spremenljivko vrednostX zapiše 255, novo vrednost vrednostY prepiše v vrednost tx_buf[1] in pošlje vrednost tx_buf[1] preko nrf24l01 modula na arduino mega 2560, če pa vrednost vrednostY ne izpolnjuje teh pogojev program pošlje preko nrf24l01 modula vrednost vrednostY, ki smo ji prišteli 1.
Šesta zanka preverja, če je vrednost mapY manjša od 115. Če je, se začne izvajati for zanka, ki vsak krog odšteje 1 vrednosti vrednostY. Znotraj te if zanke je še ena if zanka, ki preverja, če je vrednostY manjša od 1, če je v spremenljivko vrednostY zapiše 0, novo vrednost vrednostY prepiše v vrednost tx_buf[1] in pošlje vrednost tx_buf[1] preko nrf24l01 modula na arduino mega 2560, če pa vrednost vrednostY ne izpolnjuje teh pogojev program pošlje preko nrf24l01 modula vrednost vrednostY, ki smo ji odšteli 1.
Na koncu if zank se program vrne na začetek, prebere novo vrednost in ponovi cel krog znova. Razlika je lahko samo v vrednostih.

```
#define debug 1

#include "NRF24L01.h"

#define TX_ADR_WIDTH    5  // 5 unsigned chars TX(RX) address width
#define TX_PLOAD_WIDTH  3   // 3 unsigned chars TX payload
unsigned char TX_ADDRESS[TX_ADR_WIDTH]  =
{
    0x00,0x00,0x00,0x00,0x01
};
unsigned char rx_buf[TX_PLOAD_WIDTH] = {0}; // initialize value
unsigned char tx_buf[TX_PLOAD_WIDTH] = {0};

const int X_pin = 0; // analog pin connected to X output
const int Y_pin = 1; // analog pin connected to Y output

int VRx = A0;
int VRy = A1;
int tester = 0;
int vrednostX = 122;
int vrednostY = 122;
int xPosition = 0;
int yPosition = 0;
int mapX = 0;
int mapY = 0;

void setup()
{

  Serial.begin(9600);

  pinMode(VRx, INPUT);
  pinMode(VRy, INPUT);

   tx_buf[0]=vrednostX;
   tx_buf[1]=vrednostY;


     NRF_Init();
    NRF_SeTxMode();                         // Initialize IO port
    Serial.println("TX_Mode Start");
}
```

Slika 24: Program pošiljatelj (nano) 1

```
void loop()
{
  tx_buf[0]=vrednostX;
  tx_buf[1]=vrednostY;
  xPosition = analogRead(VRx);
  yPosition = analogRead(VRy);
  mapX = map(xPosition, 0, 1023,0 ,255 );
  mapY = map(yPosition, 0, 1023,0 ,255 );



   if(mapX==122){

      vrednostX=122;
      tx_buf[0]=vrednostX;
      do
      {
          NRF_Send(tx_buf[0]);
      }
      while(!NRF_CheckAck());
}



   if(mapX>130){

      for(vrednostX<=255; vrednostX++;){
         if(vrednostX>254)
         {
            vrednostX=255;
         }
      }
        tx_buf[0]=vrednostX;
        do
        {
           NRF_Send(tx_buf[0]);
        }
        while(!NRF_CheckAck());
        break;
    }
```

Slika 25: Program pošiljatelj (nano) 2

```
  if(mapX<115){

     for(vrednostX>=0; vrednostX--;) {
        if(vrednostX<1)
        {
          vrednostX=0;
        }

        tx_buf[0]=vrednostX;
        do
        {
          NRF_Send(tx_buf[0]);
        }
        while(!NRF_CheckAck());
        break;
    }
}



    if(mapY==128){

       vrednostY=122;
       tx_buf[1]=vrednostY;
       do
       {
          NRF_Send(tx_buf[1]);
       }
       while(!NRF_CheckAck());
   }
```

Slika 26: Program pošiljatelj (nano) 3

```
    if(mapY>130){

       for(vrednostY<=255; vrednostY++;){
          if(vrednostY>254)
          {
             vrednostY=255;
          }

          tx_buf[1]=vrednostY;
          do
          {
             NRF_Send(tx_buf[1]);
          }
         while(!NRF_CheckAck());
         break;
     }
  }



   if(mapY<115){

      for(vrednostY>=0; vrednostY--;){
          if(vrednostY<1)
          {
           vrednostY=0;
          }

          tx_buf[1]=vrednostY;
          do
          {
             NRF_Send(tx_buf[1]);
          }
          while(!NRF_CheckAck());
          break;
       }
    }
}
```

Slika 27: Program pošiljatelj (nano) 4

4.1.5.2 Programiranje arduina mega 2560

Program s katerim sva programirala arduino nano je arduino IDE, kjer se uporablja programski jezik C++. Arduino mega 2560 služi upravljanju servo motorja in elektromotorja za pogon. Vrednosti za ta dva motorja prejme preko modulov nrf24l01 od arduina nano.

Na začetku sva določila 7 spremenljivk: vrednostY, vrednostX, potpin=0, potpin1=0, valY=0, val_mapY=0, valX=0 in val_mapX=0.
V void setupu sva določiva vrednost serial begin na 9600. ESCju sva določila priključek 7 in razpon vrednosti med 1000 in 2000. Servo motorju sva določila priključek 6. Na ESC sva 2 sekundi pošiljala vrednost 1478, ki je sredina vrednosti (motor se ne vrti). Na koncu v void setapu izpiševa besedo “ok” na zaslon v programu Arduino IDE.
V void loopu sva na začetku napisala if zanko, ki vedno preverja, če je arduino mega 2560 sprejel kakšno vrednost na rx_buf. Če je, vrednosti valX pretvori iz vrednosti med 0 in 255 v vrednosti med 1000 in 2000 in novo spremenljivko poimenuje val_mapX in vrednosti valY pretvori iz vrednosti med 0 in 255 v vrednosti med 0 in 120 in novo spremenljivko poimenuje val_mapY. Za tem si sledi šest enakovrednih if zank.
Prva zanka preverja, če je vrednost val_mapX večja od 1550. Če je, se začne izvajati for zanka, ki vsak krog prišteje 1 vrednosti vrednostX. Za tem sledi še ena if zanka, ki preverja, če je vrednost vrednostX večja od 1990. Če je, to vrednost spremeni v 2000 in jo pošlje na ESC. Če pa ni, pa pošlje vrednost vrednostX, ki smo ji prišteli 1. In program nadaljuje na naslednji od enakovrednih if zank.
Druga zanka preverja ali je val_mapX enak 1478. Če je, 1478 prepiše v vrednost vrednostX in jo pošlje na ESC. Če se pa val_mapX ne ujema s 1478, pa program nadaljuje na naslednjo if zanko.
Tretja zanka preverja, če je vrednost val_mapX manjša od 1550. Če je, se začne izvajati for zanka, ki vsak krog odšteje 1 vrednosti vrednostX. Za tem sledi še ena if zanka, ki preverja, če je vrednost vrednostX manjša od 1010. Če je, to vrednost spremeni v 1000 in jo pošlje na ESC. Če pa ni, pa pošlje vrednost vrednostX, ki smo ji prišteli 1. Program nadaljuje na naslednji od enakovrednih if zank.
Četrta zanka preverja ali je val_mapY enak 57. Če je, 57 prepiše v vrednost vrednostY in jo pošlje na servo motor, če se pa val_mapY ne ujema s 57, pa program nadaljuje na naslednjo if zanko.
Peta zanka preverja, če je vrednost val_mapY večja od 70. Če je, se začne izvajati for zanka, ki vsak krog prišteje 1 vrednosti vrednostY. Za tem sledi še ena if zanka, ki preverja, če je vrednost vrednostY večja od 119. Če je, to vrednost spremeni v 120 in jo pošlje na servo motor, če pa ni, pa pošlje vrednost vrednostY, ki smo ji prišteli 1. Program nadaljuje na naslednji od enakovrednih if zank.
Šesta zanka preverja, če je vrednost val_mapY manjša od 50. Če je, se začne izvajati for zanka, ki vsak krog odšteje 1 vrednosti vrednostY. Za tem sledi še ena if zanka, ki preverja, če je vrednost vrednostY manjša od 2. Če je, to vrednost spremeni v 0 in jo pošlje na servo motor, če pa ni, pa pošlje vrednost vrednostY, ki smo ji prišteli 1.

S tem je program zaključil zadnjo if zanko in začel s ponovnim branjem vrednosti iz modula nrf24l01.

```
#include "NRF24L01.h"
#include <Servo.h>
#define debug 0
#define TX_ADR_WIDTH    5   // 5 unsigned chars TX(RX) address width
#define TX_PLOAD_WIDTH  3  // 3 unsigned chars TX payload
unsigned char RX_ADDRESS[TX_ADR_WIDTH]  =
{
  0x00,0x00,0x00,0x00,0x01
};
/*unsigned char RX_ADDRESS[TX_ADR_WIDTH]  =
{
  0x34,0x43,0x10,0x10,0x01
};*/
unsigned char rx_buf[TX_PLOAD_WIDTH] = {0};
unsigned char tx_buf[TX_PLOAD_WIDTH] = {1};

Servo ESC;
Servo myservo;
int vrednostY;
int vrednostX;
int potpin = 0;
int valX=0;
int val_mapX=0;
int valY=0;
int val_mapY=0;
void setup()
{


    Serial.begin(9600);
    NRF_Init();                         // Initialize IO
    NRF_SetRxMode();
    //Serial.println("RX_Mode start...");
    myservo.attach(6);
    ESC.attach(7,1000,2000);
    ESC.writeMicroseconds(1478);
    delay(2000);

    Serial.println("ok");
}
```

Slika 28: Program prejemnik (mega 2560) 1

```
void loop()
{
  if(NRF_Receive(rx_buf)){
    valX = rx_buf[0];
    val_mapX = map(valX, 0, 255, 1000, 2000);
    valY = rx_buf[1];
    val_mapY = map(valY, 0, 255, 0, 120);




    if(val_mapX==1478)
    {
      vrednostX=1478;
      ESC.write(vrednostX);
    }



    if(val_mapX>1550){
        for(vrednostX<=2000; vrednostX++;){

          if(vrednostX>1990)
          {
            vrednostX=2000;
          }

          ESC.write(vrednostX);
          break;
      }
    }
```

Slika 29: Program prejemnik (mega 2560) 2

```
if(val_mapX<1550){
    for(vrednostX>=1000; vrednostX--;) {

      if(vrednostX<1010)
      {
        vrednostX=1000;
      }

      ESC.write(vrednostX);
      break;
  }
}



 if(val_mapY==57)
 {
   vrednostY=57;
   myservo.write(vrednostY);
 }




 if(val_mapY>70){
    for(vrednostY<=120; vrednostY++;){

      if(vrednostY>119)
      {
         vrednostY=120;
      }

      myservo.write(vrednostY);
      break;
   }
 }
```

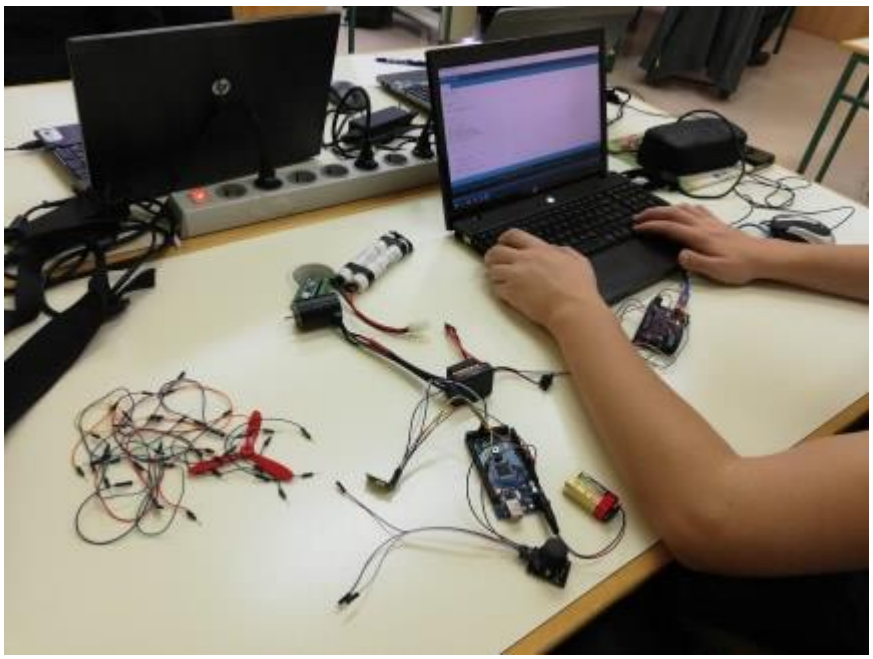
Slika 30: Program prejemnik (mega 2560) 3

```
      if(val_mapY<50){
          for(vrednostY>=0; vrednostY--;) {

            if(vrednostY<2)
            {
               vrednostY=0;
            }

            myservo.write(vrednostY);
            break;
       }
     }
   }
 }
```

Slika 31: Program prejemnik (mega 2560) 4

4.1.6 Testiranje

Na začetku sva testirala, če imava delujoči ESC, elektromotor, baterijo in mikrokrmilnika. To sva testirala z osnovnim programom (branje potenciometra, pretvarjanje v pravilne vrednosti in pošiljanje na ESC) in se prepričala, da imava delujoče komponente. Ko sva se prepričala, da vse deluje, sva začela testirati nrf24l01 module. Ko nama je uspelo, da so nrf24l01 moduli pošiljali informacije preko anten iz enega mikrokrmilnika na drugega, sva začela vse skupaj povezovati.
Na koncu sva testirala še delovanje servo motorja in pogonskega motorja preko modulov nrf24l01.
Najino zadnje testiranje je bila vožnja avtomobilčka. Ta test sva morala opraviti zato, da sva lahko ovrgla ali potrdila hipotezo, ki pravi da bo avtomobilček deloval z arduino krmilniki namesto tovarniškimi krmilniki.



Slika 32: Testiranje ESC-ja

4.2 RAZPRAVA

Rezultate, ki sva jih dobila, nam kažejo, da bo hipoteza, ki trdi, da je mogoče uporabiti arduino mikrokrmilnike namesto tovarniškega krmilnika delno potrjena. Hipoteza, ki trdi, da bo avtomobilček zmožen doseči 80 km/h, ne bo potrjena.
Arduina pa imata občasno probleme z obdelavo vseh podatkov in sta delovala s občasnimi prekinitvami. Za še večjo natančnost sva program priredila tako, da postopoma lovi vrednost, ki jo zahtevamo s potenciometrom. Tako ni mogoče povzročiti nenadnih skokov vrednosti. S tem sva sicer odpravila mehanske težave elektromotorja, a arduino krmilniki so še vedno imeli občasne težave z občasnimi prekinitvami. Tako je hipoteza delno potrjena.
Hitrost sva izračunala tako, da sva uporabila podatka, ki sva ju izmerila: vrtljaje koles in obseg koles in jih vnesla v spletno stran, ki je namenjena temu računanju. Rezultat je bil manjši od 80 km/h, zato je bila najina druga teza ovržena.

5 ZAKLJUČEK

Med izdelovanjem sva se veliko naučila glede programiranja z dvema različnima krmilnikoma in kako uporabljati brezžične module. Pri odpravljanju težav nama je bil v največjo pomoč mentor in splet, kej sva odkrila veliko rešitev za težave na katere sva naletela.

Najini hipotezi nista bili popolnoma potrjeni :

- Avtomobilček ni sposoben doseči hitrost 80 km/h.
- Avtomobilček lahko delno deluje z arduino krmilniki namesto tovarniškega krmilnika.

6 VIRI

Nrf24l01 modul. Pridobljeno 5.3.2020 s strani:
http://wiki.sunfounder.cc/index.php?title=NRF24L01_Wireless_Transceiver_Module

Krmilnik arduino nano. Pridebljeno 5.3.2020 s strani:
https://www.geeetech.com/wiki/index.php/Arduino_Nano

Krmilnik arduino mega 2560. Pridobljeno 5.3.2020 s strani:
https://www.geeetech.com/wiki/index.php/Arduino_Mega_2560

Servo motor. Pridobljeno 5.3.2020 s strani:
https://www.aliexpress.com/digit-servo_reviews.html

Potenciometer. Pridobljeno 5.3.2020 s strani:
https://www.brainy-bits.com/arduino-joystick-tutorial/

7 ZAHVALA

Zahvaljujeva se vsem, ki so kakorkoli pripomogli k izdelavi najine raziskovalne naloge. Še posebej bi se zahvalila mentorju Gregorju Kramerju, univ. dipl. inž. el. in Davorju Zupanu, inž. el. za vso podporo, nasvete in prosti čas, ki sta nama ga namenila.

IZJAVA*

Mentor **Gregor Kramer** v skladu z 20. členom Pravilnika o organizaciji mladinske raziskovalne dejavnosti »Mladi za Celje« Mestne občine Celje, zagotavljam, da je v raziskovalni nalogi z naslovom **Električni avtomobilček na daljinsko upravljanje**, katere avtorji so **Jakob Tomaž Plevnik, David Flander**:

- besedilo v tiskani in elektronski obliki istovetno,
- pri raziskovanju uporabljeno gradivo navedeno v seznamu uporabljene literature,
- da je za objavo fotografij v nalogi pridobljeno avtorjevo dovoljenje in je hranjeno v šolskem arhivu,
- da sme Osrednja knjižnica Celje objaviti raziskovalno nalogo v polnem besedilu na knjižničnih portalih z navedbo, da je raziskovalna naloga nastala v okviru projekta Mladi za Celje,
- da je raziskovalno nalogo dovoljeno uporabiti za izobraževalne in raziskovalne namene s povzemanjem misli, idej, konceptov oziroma besedil iz naloge ob upoštevanju avtorstva in korektnem citiranju,
- da smo seznanjeni z razpisni pogoji projekta Mladi za Celje.

Celje, 5. 6. 2020 žig šole Podpis mentorja

Podpis odgovorne osebe

ŠOLSKI CENTER CELJE
ŠOLA ZA KEMIJO, ELEKTROTEHNIKO IN RAČUNALNIŠTVO
SREDNJA 3 CELJE, Pot na Lavo 22

*
POJASNILO

V skladu z 20. členom Pravilnika raziskovalne dejavnosti »Mladi za Celje« Mestne občine Celje je potrebno podpisano izjavo mentorja (-ice) in odgovorne osebe šole vključiti v izvod za knjižnico, dovoljenje za objavo avtorja (-ice) fotografskega gradiva, katerega ni avtor (-ica) raziskovalne naloge, pa hrani šola v svojem arhivu.