KAUNO TECHNOLOGIJOS UNIVERSITETAS

ELEKTROS IR ELEKTRONIKOS FAKULTETAS

Vytautas Tilinskas

SAVAEIGIO MODELIO NUOTOLINIO VALDYMO KŪRIMAS

Baigiamasis bakalauro projektas

Vadovas
Prof. Adas Gelžinis

KAUNAS, 2015

KAUNO TECHNOLOGIJOS UNIVERSITETAS
ELEKTROS IR ELEKTRONIKOS FAKULTETAS
ELEKTROS ENERGETIKOS SISTEMŲ KATEDRA

SAVAEIGIO MODELIO NUOTOLINIO VALDYMO KŪRIMAS

Baigiamasis bakalauro projektas
Elektros inžinerija 612H62001

Vadovas
Prof. Adas Gelžinis

Recenzentas

Projektą atliko
Vytautas Tilinskas

TVIRTINU:
KTU Elektros ir elektronikos fakulteto
_____katedros vedėjas
_____
201….. ....

BAKALAURO BAIGIAMOJO PROJEKTO UŽDUOTIS

Išduota studentui: *Vytautui Tilinskui* Grupė *EEVI-1*

1. Darbo tema:

Lietuvių kalba: *SAVAEIGIO MODELIO NUOTOLINIO VALDYMO KŪRIMAS*

Anglų kalba: DESIGN OF REMOTE CONTROL FOR AUTOMOTIVE MODEL

Patvirtinta 2015 m. balandžio mėn. 7 d. dekano potvarkiu Nr. *ST18-F-03-1*

2. Darbo tikslas: *Sukurti nuotolinio valdymo sistemą savaeigiam modeliui ir ją realizuoti.*

3. Reikalavimai ir sąlygos: *Sistema lengvai keičia judėjimo kryptį ir greitį, valdant du elektros variklius. Greitis keičiamas žingsniškai ir tolygiai.*

4. Projekto struktūra. *Turinys konkretizuojamas kartu su vadovu, atsižvelgiant į BBP pobūdį, pateiktą Metodinių reikalavimų 14 ir 15 punktuose.*

Projekto struktūrą turi sudaryti santrauka, įvadas, sistemų apžvalga, kuriamos sistemos aprašymas, mikrovaldiklio programavimas, eksperimentinė dalis ir išvados.

5. Ekonominė dalis. *Jei reikia ekonominio pagrindimo; turinys ir apimtis konkretizuojama darbo eigoje kartu su vadovu.*

6. Grafinė dalis. *Jei reikia, pateikiama schemos, algoritmai ir surinkimo brėžiniai; turinys ir apimtis konkretizuojama darbo eigoje kartu su vadovu.*

7. Ši užduotis yra neatskiriama bakalauro baigiamojo projekto dalis

8. Projekto pateikimo gynimui kvalifikacinėje komisijoje terminas *iki 2015-05-30*
(data)

Užduotį gavau: _____ _____
(studento vardas, pavardė, parašas) *(data)*

Vadovas: _____ _____
(pareigos, vardas, pavardė, parašas) *(data)*

KAUNAS, 2015

KAUNO TECHNOLOGIJOS UNIVERSITETAS

Elektros ir elektronikos fakultetas

(Fakultetas)

Vytautas Tilinskas

(Studento vardas, pavardė)

Elektros inžinerija 612H62001

(Studijų programos pavadinimas, kodas)

Baigiamojo projekto „Pavadinimas“

AKADEMINIO SĄŽININGUMO DEKLARACIJA

20 15 m. Gegužės 16 d.

Kaunas

Patvirtinu, kad mano **Vytauto Tilinsko** baigiamasis projektas tema „ Savaeigio modelio nuotolinio valdymo kūrimas “ yra parašytas visiškai savarankiškai, o visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.
Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

(vardą ir pavardę įrašyti ranka) *(parašas)*

V. Tilinskas. (2015) *Savaeigio modelio nuotolinio valdymo kūrimas*. Bakalauro baigiamasis darbas / vadovas prof. Adas Gelžinis; Elektros energetikos sistemų katedra,Elektros ir elektronikos fakultetas, Kauno technologijos universitetas. – Kaunas 2015. – 44 p.

SANTRAUKA

Šiame darbe sukuriama nuotolinio valdymo sistema savaeigiam modeliui. Sistema skirta per atstumą keisti važiavimo kryptį ir greitį, sistemos valdymui pritaikant išmanųjį telefoną, turintį ANDROID operacinę sistemą. Siekiant nustatyti sistemos valdymo tikslumą yra eksperimentiškai atliekami tyrimai. Mikrovaldiklyje keičiamas impulso pločio laikas, nuo kurio priklauso vidutinė išėjimo įtampa, keičianti variklio sukimosi greitį.

V. Tilinskas. (2015) Design of Remote Control for Automotive Model*:* Bachelor's work/ supervisor prof. A. Gelžinis; Department of Electrical Energy systems, Faculty of Electrical and Electronics, Kaunas University of Technology, -Kaunas, 2015. – 44 p.

SUMMARY

This work establishes a remote control system to self-propelled model. The system is designed to remotely change the direction of travel and speed. Management of the system is to adapt smartphone that has the Android operating system. In order to determine the accuracy of the control system is experimental investigations. Microcontrollers replaced pulse width defining the time, which determines the average output voltage that changes the motor rotation speed.

Turinys

Santrumpų žodynas

PWM (Pulse-width modulation) – impulso pločio moduliacija

RC(A resistor–capacitor circuit) – rezistoriaus ir kondensatoriaus grandinė

DC (Direct current) – nuolatinė srovė

H-bridg – H formos valdymo tiltelis

USB (Universal Serial Bus) – universali kompiuterinė jungtis

LiPo - Ličio polimerų baterija

Įvadas

Nuotolinių būdu valdomos sistemos yra šių dienų kasdienybė.Per atstumą yra valdomi lėktuvai, robotai, automobiliai, namų signalizacijos ir daugybė kitų įrenginių.

Norėdamas parodyti įgytas žinias universitete, nusprendžiau sukurti savaeigio modelio valdymo sistemą, kurioje per atstumą galima būtų keisti važiavimo kryptį ir greitį.Sistemos valdymas vyksta valdant nuolatinės srovės variklius. Vienas variklis valdo važiavimo greitį, kitas judėjimo trajektoriją. Greitis valdomas žingsniškai, iki pasirinkto iš valdymo pulto žingsnio. Variklius valdo užprogramuotas mikrovaldiklis, kurio valdymas susietas su mobiliuoju telefonu.

Eksperimento metu bus bandomas sistemos ryšio veikimas per blokines perdangas valdant sistemą namuose ir sistemai veikiant atviromis lauko sąlygomis, kurios neslopintų valdymo signalo. Taip pat bus tiriamas sistemos veikimo tikslumas keičiant impulsų plotį,nuo kurio priklauso variklius valdanti įtampa. Šie sistemos valdymo tyrimai yra reikalingi įvertinant sistemos veikimą, bei sprendžiant kaip būtų galima panaudoti šią valdymo sistemą.

Šį darbą sudaro savaeigio modelio sistemos apžvalga, kurioje yra aprašomos kokios dalys galėtų būti panaudotos joje. Toliau sudaroma principinė sistemos schema, kuri bus kuriama šiame darbe. Užprogramuojamas mikrovaldiklis, kuris valdys visos sistemos darbą. Eksperimentinė dalis, kurioje išbandomas sistemos veikimas ir išvados, kuriose apibendrinta kaip pavyko įgyvendinti projekto reikalavimus.

1. Savaeigio modelio sistemos apžvalga

Nepriklausomai nuo to ką norima valdyti, sistemos sandara išlieka labai panaši. Pagrindinėmis sistemos dalimis yra laikoma mikrovaldiklis, kuris apdoroja gautus signalus ir juos realizuoja, taip pat siųstuvas ir imtuvas. Siųstuvas dažnai būna sujungtas su valdymo pultu (dar vienu valdikliu) iš kurio gaunamos valdymo komandos, o imtuvas su mikrovaldikliu kuris tas gautas komandas apdoroja ir paverčia išėjimo signalais, skirtais atvaizduoti perduodamą informacija ar valdyti pasirinktą parametrą (apšvietimą, muziką, variklių valdymą ir t.t.).

Ryšio tipo pasirinkimas kaip ir mikrovaldiklio yra pakankamai didelis. Galima norimą informaciją perduoti radijo bangomis, palydoviniu GPS ryšiu, Wi-Fi arba Bluetuth bevieliu ryšio. Ryšio pasirinkimą turi nulemti kuriamos sistemos reikalavimui ir tai, kam ji bus skirta naudoti.

Kuriant sistemą, kuriai reikės perduoti duomenis, reikia rinktis mikrovaldiklį kuris turėtų duomenų siuntimo ir priėmimo kojeles (ne visi mikrovaldikliai jas turi). Prieš pasirenkant mikrovaldiklį reikia žinoti kokias funkcijas atliks sistema ir kiek išėjimų turi turėti mikrovaldiklis, tam kad kuriama sistema gebėtų atlikti visas norimas funkcijas.

1.1 Mikrovaldikliai

Mikrovaldiklis yra didelio integracijos lygio mikroschema, kurioje yra visos kontroleriui reikalingos dalys. Pagrindinės yra šios:

- Centrinis procesorius (CPU);
- Operatyvinė atmintis (RAM);
- Ištrinama programuojama pastovioji atmintis (EPROM/PROM/ROM);
- Nuoseklus ir lygiagretus įėjimo/isėjimo prievadai (I/O ports);
- LCD kontroleriai;
- Taimeriai;
- Pertraukimų kontroleris.

Pagrindiniai mikrovaldiklių pasirinkimo kriterijai:

- Mikrovaldikliai gaminami su 4, 8, 16 ir 32 bitų duomenų magistrale.
- Maksimalus taktinis dažnis. Paprastai mikrovaldiklių greitaveika pateikiama kaip atliekamų operacijų skaičiumi per sekundę, kuris gali būti mažesnis arba lygus taktiniam dažniui.

- Atminties tipas ir kiekis. Šiuolaikiniuose mikrovaldikliuose plačiausiai taikoma Flash ir EEPROM tipo atmintys kadangi joms programuoti užtenka nedidelės įtampos. ROM tipo atmintis turintys mikrovaldikliai naudojama masinėje gamyboje. Atminties kiekis gali svyruoti nuo 0,5 kB iki 2MB ar daugiau.
- Periferija - tai papildomi moduliai, integruoti į tą patį kristalą. Tai gali būti blokai palaikantys įvairius komunikacijos standartus, analoginių signalų keitikliai (ASK, SAK), kontroleriai palengvinantys LCD displėjų valdymą ir t.t.

Vienas iš populiariausių rinkoje esantis mikrovaldiklių gamintojų Arduino, pateikia pakankamai platų jų pasirinkimą, priklausomai nuo mikrovaldiklio išėjimų kiekio ir atminties poreikio. Šio tipo valdikliai pasižymi programavimo paprastumu ir aukšta greitaveika.

Arduino – platforma yra su integruotu Atmel firmos mikrokontroleriu. Tai mažas kompiuteris, kurio pagalba galima valdyti įvairius įrenginius. Ši platforma yra populiari, dėl to kad yra atviro kodo ir pasižymi didelėmis galimybėmis .Arduino platforma pristatyta 2005 metais. Prie šių mikrovaldiklių galima prijungti įvairius modulius, daviklius, ekranus ir juos valdyti programiniu kodu. Privalumas yra tas, kad jų nereikia lituoti, kurti ar projektuoti. Jie turi tik kontaktus, kuriuos reikia tinkamai prijungti prie platformos. Arduino skirtas ne tik pradedantiesiems.

Arduino ATMega 2560 mikrokontroleris yra naudojamas pirmajame lietuviškame palydove „LituanicaSAT-1“. Tai labai svarbi dalis, be kurios sistema negalėtų funkcionuoti. Apie kažkokius gedimus palydovo sistemoje neteko girdėti spaudoje, sistema neturėjo nesklandumų, todėl galima sutikti, kad šis mikrokontroleris yra patikimas ir svarbiose misijose.

Mikrokontrolerio programavimui naudojama speciali Arduino programinė įranga, kurią nemokamai galima parsisiųsti iš http://www.arduino.cc/ internetinio puslapio. Oficialioje svetainėje taip pat galima rasti daugybę programų pavyzdžių, kuriuos analizuojant daug greičiau yra išmokstama programuoti Arduino platformą. Svetainėje taip pat aktyvus forumas, kuriame galima rasti reikiamą informaciją ar konsultuotis su kitais vartotojais.

Arduino platformos privalumai:

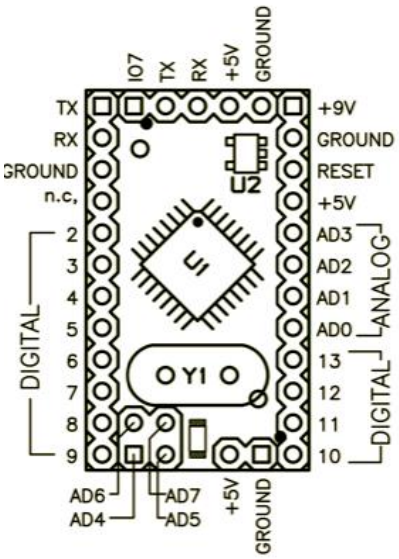
- didelės galimybės;
- patikimas;
- nereikia gerai išmanyti elektronikos;
- pigūs priedai;

- paprasta C ir C++ kalba;
- atviro kodo;
- daug programos pavyzdžių.

Dar vienas Arduino privalumas – tai, kad jam yra sukurti papildomi skydeliai, kurie suteikia galimybę Arduino prijungti prie interneto, WiFi tinklo, Bluetooth ryšio. Taip pat gali suteikti galimybę naudoti atminties korteles ir dar daug visokių kitų galimybių.Visi minėti privalumai nulėmė šios platformos pasirinkimą.

1.1.1 Arduino MINI mikrovaldiklis

Arduino mini tai mažesnių išmatavimų su jungtimis Arduino plokštė be USB jungties. Minimalus dydis (30x18 mm) leidžia sutaupyti daug vietos, plokštė turi jungtis pajungimui. Šis mikrovaldiklis turi 14 skaitmeninių įvedimo jungčių iš kurių 6 gali būti naudojamos kaip kintamos įtampos vertės išėjimai ir 8 analoginius išėjimus(tai matyti 1.1.1.1 pav.). Norint prijungti prie plokštės kabelį, per kurį galima būtų įkelti programos kodą, reikia prijungti USB adapterį. Šių mikrovaldiklių trūkumas tas, kad jie dėl savo dydžio daug lengviau lūžta, palyginus su kitomis didesnėmis Arduino mikrovaldiklio platformomis. Skaitmeninės jungtys skirtos įvesčiai ir išvesčiai.. Šios jungtys yra kontroliuojamos mikrokontrolerio.Analoginės jungtys naudojamos tik įvesčiai.



1.1.1.1 pav. Arduino Mini mikrovaldiklio kanalų išėjimai.

Kaip matyti iš schemos, yra 14 išėjimų. RX ir TX kanalai naudojami informacijai gauti prijungiant UTP kabelį ar ryšio modulį.

Techniniai duomenys:

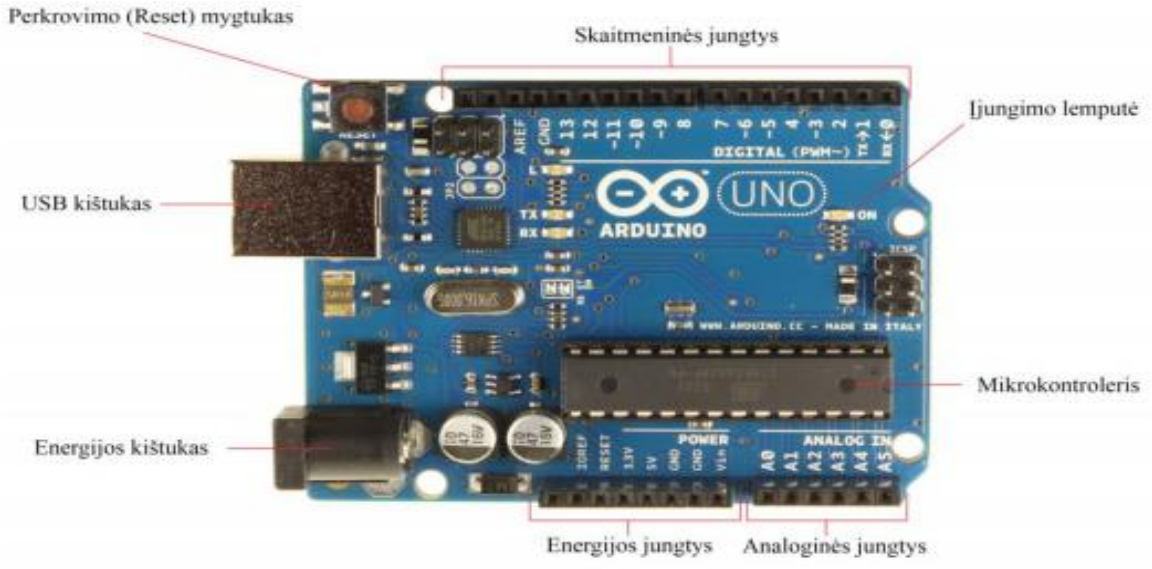
- Mikrovaldiklis Atmega328;
- Darbinė įtampa 5V;
- Įėjimo įtampa 7-9 V;
- Skaitmeniniai I / O įėjimai 14 (iš kurių 6 naudojami kaip PWM išėjimai);
- Analoginiai įėjimai 8 ;
- DC srovė per I / O įėjimą 40 mA;
- Flash atmintis 32 KB (iš kurių 2 KB naudojami bootloaderio);
- SRAM 2 KB;
- Laikrodžio greitis 16 MHz

1.1.2 Arduino Uno mikrovaldiklis

Arduino UNO plokštė su ATmega328 mikrokontroleriu . Ši plokštė turi 14 skaitmeninių įėjimų/išėjimų (iš kurių 6 gali būti naudojami PWM signalų išvedimui), 6 analoginius išėjimus, USB jungtį, maitinimo lizdą, ICSP jungtį ir RESET mygtuką. Plokštės dydis (68.6x53.4)

Techniniai duomenys:

- Mikrovaldiklis Atmega328;
- Darbinė įtampa 5V;
- Įėjimo įtampa 7-12V;
- Skaitmeniniai I / O įėjimai 14 (iš kurių 6 naudojami kaip PWM išėjimai);
- Analoginiai įėjimai 6 ;
- DC srovė per I / O įėjimą 40 mA;
- Flash atmintis 32 KB (iš kurių 0.5KB naudojami bootloaderio);
- SRAM 2 KB;
- Laikrodžio greitis 16 MHz



1.1.2.1 pav. Arduino Uno išėjimai/įėjimai

Išėjimai su bangelėmis (11,10,9,5,6,3) reiškia kad išėjimo signalas gali būti ir kintamos reikšmės. Perkrovimo mygtukas – skirtas Arduino prietaiso perkrovimui. Pasitaiko, kad dėl programos klaidų, netikslumų ar kitų priežasčių prietaisas užstringa ir nieko nedaro, tuomet perkrovimo mygtukas paleidžia veikiamą programą iš naujo.Iš esmės techniniai duomenys beveik identiški Uno ir Mini serijos mikrovaldiklių. Skiriasi tik dydis ir prijungimo prie kompiuterio būdas.

1.1.3 Arduino Mega mikrovaldiklis

Arduino Megamikrovaldiklis yra su USB sąsaja ir galimybe prisijungti prie išmaniojo telefono su ANDROID operacine sistema. Jis turi 54 skaitmenines įvedimo/išvedimo kanalusiš kurių 15 gali būti naudojamos kaip kintamos įtampos vertės išėjimai ir 16 analoginių išėjimų. Dėl didelio skaičiaus išėjimų ir vidinės atminties, mikrovaldiklis tinkamas sudėtingiems projektams, tačiau netinkamas ten kur reikia taupyti vietą ir viską daryti kompaktiškai.

Techniniai duomenys:

- Mikrovaldiklis Atmega2560-16AU;
- Darbinė įtampa 5V;
- Įėjimo įtampa 7-12V;
- Skaitmeniniai I / O įėjimai 54 (iš kurių 14 naudojami kaip PWM išėjimai);
- Analoginiai įėjimai 16 ;
- DC srovė per I / O įėjimą 40 mA;

- Flash atmintis 256 KB (iš kurių 8KB naudojami bootloaderio);
- SRAM 8 KB;
- Laikrodžio greitis 16 MHz

Šis mikrovaldiklis turi 8 informacijos perdavimo kanalus, tad vienu metu gali būti prijungta net kelios ryšio sistemos, ko negalima padaryti su prieš tai minėtais valdikliais.

1.1.3.1 pav. Arduino Mega mikrovaldiklio schema

Iš pateikto 1.1.3.1. paveikslėlio matosi 54 išėjimai, USB jungtis, perkrovimo mygtukas “Reset”. Visa sandara identiška Arduino Uno platformai.

1.2 Ryšio moduliai Arduino platformai

Norint sukurti gerai veikiančią nuotolinio valdymo sistemą, reikia išspręsti svarbų klausimą. Kaip bus perduodama informacija? Sakykim būtų nepatogu, jei kuriama sistema veiktų 50 metrų atstumu, bet paskui save vilktų laidus, kurias būtų valdoma. Tam kad nereikėtų laidų sistemai valdyti, galima prie mikrovaldiklio prijungti įvairius modulius, kurie skirti informacijos perdavimui beveliu tinklu. Kadangi kuriama sistema bus valdoma iš išmaniojo telefono, svarbu pasirinkti lengvai suderinamą ryšio modulį, kuriuo nesunkiai būtų galima perduoti įvairias komandas realiu laiku. GPS ir GSM-GPS ryšiai šiuo atveju sunkiai suderinami, kadangi jie labiau tinkami informacijos siuntimui dideliu atstumu, ir gali vėluoti kol pasiekia adresatą.

Galimi ryšio moduliai:

- WiFi ryšio modulis
- Bluetooth ryšio modulis.

1.2.1 WiFi ryšio modulis

Wi-Fi bevielio ryšio stotelės leidžia sukurti duomenų perdavimo tinklus panaudojant plačiajuostį radijo ryšį . Wi-Firyšio technologijos yra lengvai pritaikomos su labiausiai paplitusiomis personalinių, nešiojamų kompiuterių platformomis, išmaniaisiais telefonais bei periferiniais įrenginiais.Maksimalus patikimo bevielio ryšio atstumas tarp dviejų Wi-Fi įrenginių labai priklauso nuo pastarųjų techninių parametrų ir aplinkos sąlygų - jis gali skirtis nuo kelių metrų iki kelių šimtų kilometrų. Bevieliam ryšiui naudojama radijo bangų spektro juosta gali būti užgožtas pašalinių radijo triukšmų, dėl ko gali nukentėti duomenų perdavimo greitis.Wifi ryšį tarp telefono ir Arduino valdiklio galima sukurti panaudojus mikrovaldiklio Wi-Fi priedėlį

1.2.2 Bluetooth ryšio modulis.

Arduino "Bluetooth" nuoseklusis ryšys gali būti naudingas daugeliui programų, kurios valdytų LED apšvietimą,žingsninių variklių greitį, ar paprastų variklių apsisukimų dažnį. .Šio ryšio veikimo atstumas yra apie 30 metrų, tačiau atstumas priklauso nuo aplinkos kurioje jis yra naudojamas. Uždarose patalpose atstumas gali pakankamai smarkiai sumažėti. Mobiliojo telefono programos dažnai yra kuriamos taip kad jos galėtų komunikuoti su kitais prietaisais turinčiaisBluetooth ryšio galimybe. Taip galima keistis informacija ar atstumą, žaisti tą pati žaidimą keliems asmenims vienu metu ir t.t.

1.3 Variklių sukimosi greičio valdymo būdai:

Kuriant valdymo sistemą, svarbu žinoti kas bus valdoma ir kaip tai yra daroma. Šiuo atveju sistema valdys du elektros variklius, kurių valdymas keis savaeigės mašinėlės greitį ir trajektoriją. Reikia pasirinkti tokius variklių tipus kurių valdymas būtų greitas, paprastas ir patikimas. Tam reikia žinoti variklių privalumus ir trūkumus.

Sinchroninių ir asinchroninių variklių sukimosi dažnis reguliuojamas keičiant kintamosiosįtampos dažnį, o tam naudojami gana sudėtingi ir brangūs dažnio keitikliai. Sukimosi dažnis keičiamas reguliuojant raktų atidarymo dažnį. Tokiuose varikliuose dažniausiai būna įmontuoti holo jutikliai, kurie valdikliui praneša kada reikia perjungti maitinimą kitai fazei. Šis metodas taikomas tik mažos galios varikliams, nes statūs impulsai blogina naudingumo koeficientą ir skleidžia daug triukšmų į maitinimo tinklą ir eterį.

Nuolatinės srovės variklių sukimosi dažnis reguliuojamas keičiant maitinimo įtampą. Įtampągalima keisti pasyviai, reostatu, tačiau toks reguliavimo būdas turi didelių trūkumų: reostate išsiskiria dideli šilumos kiekiai (energijos nuostoliai), taip pat šis metodas tinkamas tik esant

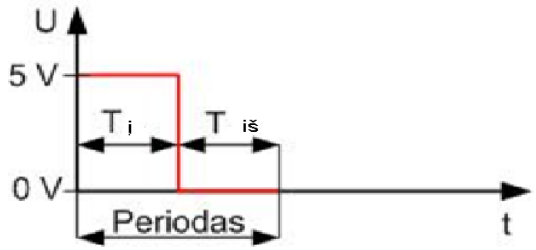
pastoviai apkrovai, kuriai padidėjus variklio maitinimo įtampa labai krenta, o apsukos žymiaisumažėja. Šiuo metu pats efektyviausias įtampos mažinimo būdas – PWM. Varikliui paduodama impulso, vienodo periodo įtampa, kurios efektyvioji vertė tiesiogiai priklauso nuo impulsų tankio:

$$U_{ef}=U\ t/T; \qquad (2)$$

čia *U*ef – efektinė įtampos vertė; *U* – valdiklio raktų maitinimo įtampa; *t/T* – impulso ir periodo laiko santykis, t.y. impulsų tankis. Taigi gauta sukimosi dažnio išraiška:

$$n=kU\ t/T; \qquad (3)$$

Suformuoti statūs impulsai elektros variklyje integruojami ir gaunama vidutinė kvadratinėvertė. Analogiški procesai vyksta integruojančioje RC grandinėje. Didelių galių variklių sukimosidažnį reguliuojant aukšto dažnio impulso pločio moduliacija, dažnai taikomi LC integratoriai, nesvariklio magnetolaidis netinkamas tokiam dažniui ir sudaro didelius energijos nuostolius (kaista).PWM integracijos pavyzdys pateiktas 8 pav., kur, vaizdesniam aiškinimui parinkta gana trumpapereinamųjų procesų konstanta τ. Čia statūs impulsai – filtro įėjime. Padidinus laiko konstantą τ amplitudę galima žymiai sumažinti.



1.3.1 pav. Impulso pločio laikinė diagrama.

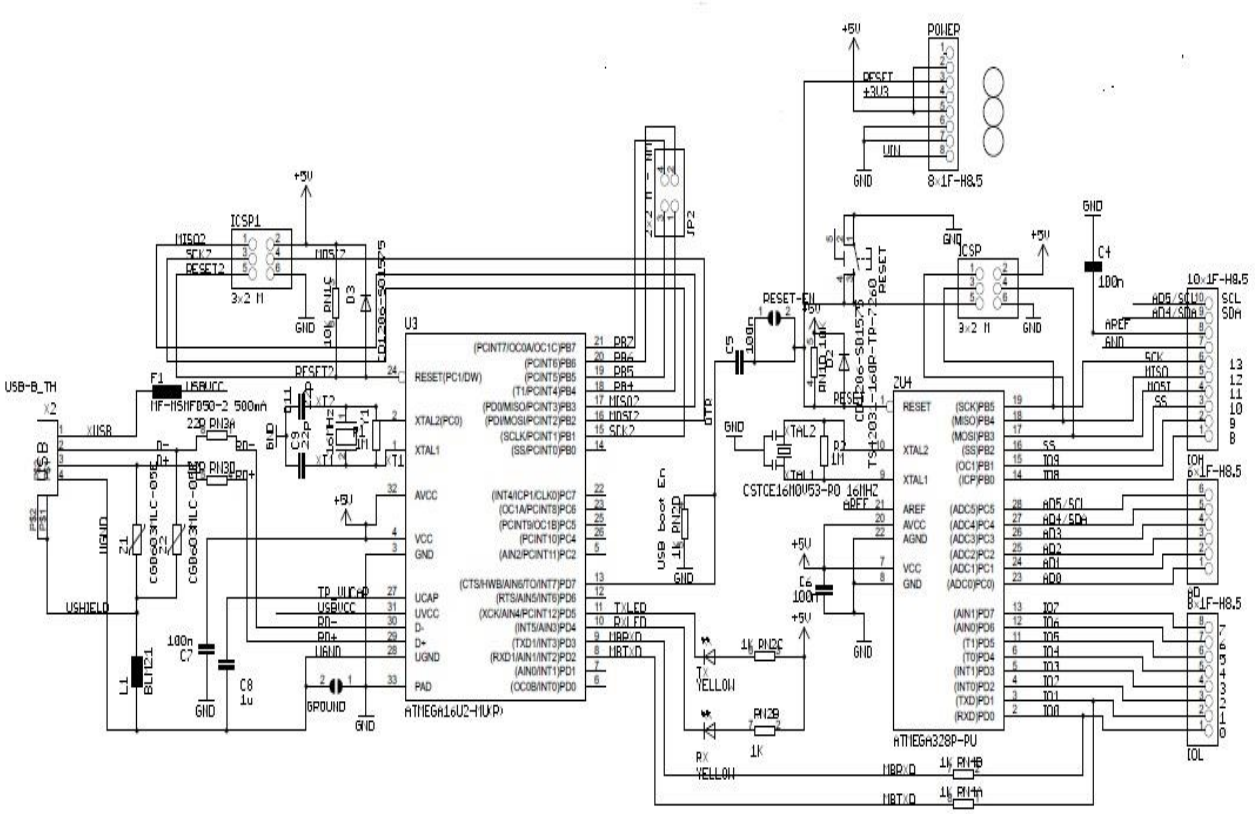
2 Savaeigio modelio realizavimas

Prieš pradedant kurti valdymo sistemą, reikia nuspręsti kokie komponentai bus naudojami, nes nuo jų priklauso visos sistemos realizavimas. Pagrindiniaikomponentai yra: variklių valdymo galios elektronika, duomenų ir valdymo sąsajos, mikrovaldiklis.

2.1 Mikrovaldiklio pasirinkimas

Valdyti kuriamam savaeigiam modeliui reikia valdiklio, kuris turėtų 4 kintamos įtampos vertės išėjimus skirtus valdyti varikliams. Todėl Arduino Mega platforma tikrai yra didelė, ir šiam projektui bus neišnaudojami šio mikrovaldiklio resursai. Rinktis reikėtų tarp Uno ir Mini mikrovaldiklių.Uno privalumas yra tas kad platformoje jau yra įmontuota USB ryšio sąsaja, kas leidžia mikrovaldiklį lengvai prijungti prie kompiuterio, iš kurio jis gauna maitinimą ir jau galima išbandyti įkeltos programos veikimą. Tai suteikia valdikliui lankstumo, greičio, nereikia kitų papildomų dalių norint įkelti programos kodą į valdiklį. Tiek Mini tiek Uno abi turi RX ir TX duomenų siuntimo kojeles, kas reiškia kad prie jų abiejų galima prijungti ryšio modulį.Įvertinus tai kad nereikia taupyti vietos ir viską daryti kuo kompaktiškiau, galima rinktis Uno platformą (abiejų platformų techniniai duomenys panašūs), kuri pagreitins ir palengvins darbą, nes reikės daug kartų prisijungti prie kompiuterio ir bandyti įvairias programas. Arduino Mini yra daug sudėtingiau prijungiama prie valdiklio, kai reikia ikelti programos kodą.

.



2.1.1 pav. Arduino Uno platformos sandara

Iš pateiktos sandaros matyti kad schema sudaro du skirtingi mikroprocesoriai. Matyti prie kurios kojos procesorių jungiami išėjimai.

2.2 Variklių pasirinkimas

Asinchroniniai varikliai, gaminami gana didelės galios, todėl mažam roboto modeliui netinka.

Sinchroniniai varikliai gaminami įvairių galių, būna ir labai mažų mikrovariklių. Tačiau kuriama mašinėlė dirba autonomiškai, t.y. jo maitinimo šaltinis yra labai ribotos talpos, taigi tokiojesistemojelabai svarbūs energijos nuostoliai. Dėl šios priežasties sistemai netinka varikliai su žadinimo apvijomis ir reikia naudoti tik variklius su pastoviuoju magnetu. Sinchroniniai varikliai su pastoviuoju magnetu kitaip vadinami žingsniniai varikliai, reikalauja sudėtingesnės valdymo schemos. Pačiu paprasčiausiu vienpolio 4 fazių žingsninio variklio atveju, jo valdymui reikalingi 4 tranzistoriai ir 4 valdymo signalai iš mikroprocesoriaus. Taip pat tokio variklio stabdymui įgyvendinti reikalingi dar 4 papildomi tranzistoriai. Dėl šių priežasčių sinchroninių variklių panaudojimo galimybė taip pat atmetama, nes tam reiktų didelės mikrovaldiklio platformos, turinčios didesnį skaičių išėjimų.

Pastoviosios srovės varikliai gaminami labai mažos – vidutinės galios. Tokie varikliai gali būti tokie maži, kad laisvai telpa į mobilų telefoną, kur atlieka vibratoriaus funkciją. Nuolatinės srovės varikliai turi padidintus energijos nuostolius dėl šepečių trinties, tačiau šią problemą nusveria kiti jo privalumai. Visų pirma – labai paprastas valdymas. Variklis pradeda suktis be jokių trūkčiojimų, o pačiu paprasčiausiu atveju jo valdymo schemą sudaro tik vienas tranzistorinis raktas.

Pakankamai akivaizdu jog nuolatinės srovės varikliai pasižymi lengviausiu valdymu, jų yra labai mažos formos ir jie tinkami naudoti automatikos įrenginiuose. Nuolatinės srovės varikliai naudotini ten, kur reikia tolygiai ir plačiose ribose reguliuoti sukimosi greitį. Pasirinkus variklių tipą, reikia pasirinkti konkretų variklį, kuris bus naudojamas sistemai realizuoti. Kuriama sistema siekiama pademonstruoti greičio ir krypties kitimą, (konkretus važiavimo greitis tam nėra svarbu) Todėl vienintelis keliamas reikalavimas varikliams yra kaina ir variklio dažnio valdymas.

2.2.1 Variklių valdymo sistemos pasirinkimas

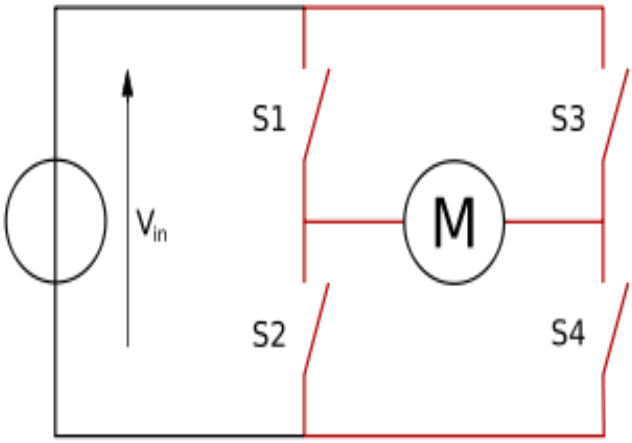
Viena iš kuriamos sistemos užduočių, yra variklio krypties kitimas. Keisti variklio greitį viena kryptimi yra daug lengviau nei priversti ji suktis priešinga kryptimi. Tam reikalingas maitinimo kontaktų apkeitimas vietomis. Visas procesas turi būti automatizuotas, nes kitaip nebus

išpildoma nuotolinio valdymo sąlyga. Tam reikalingas tranzistorinis valdymo raktas vadinamus tilteliu.

Reikalavimai variklio valdymo sistemai:

- Sukimosi dažnio keitimas ir stabilizavimas;
- Mažas užimamas spausdinto montažo plokštės plotas.

Tokios galios variklio apsisukimų dažniui keisti pakanka paties primityviausio anksčiau aptarto metodo naudojant impulso pločio moduliaciją (PWM). Mašinėlėje variklis turės suktis abejomis kryptimis, taigi reikalingas pilnas H tiltas, kurio struktūrinėschema pateikta 2.2.1.1 pav.



2.2.1.1 pav. Tiltelio schema

Čia Vin – maitinimo įtampos šaltinis, M – nuolatinės srovės variklis, o S1-S4 – jungikliai. H tiltas gali veikti keturiais režimais:

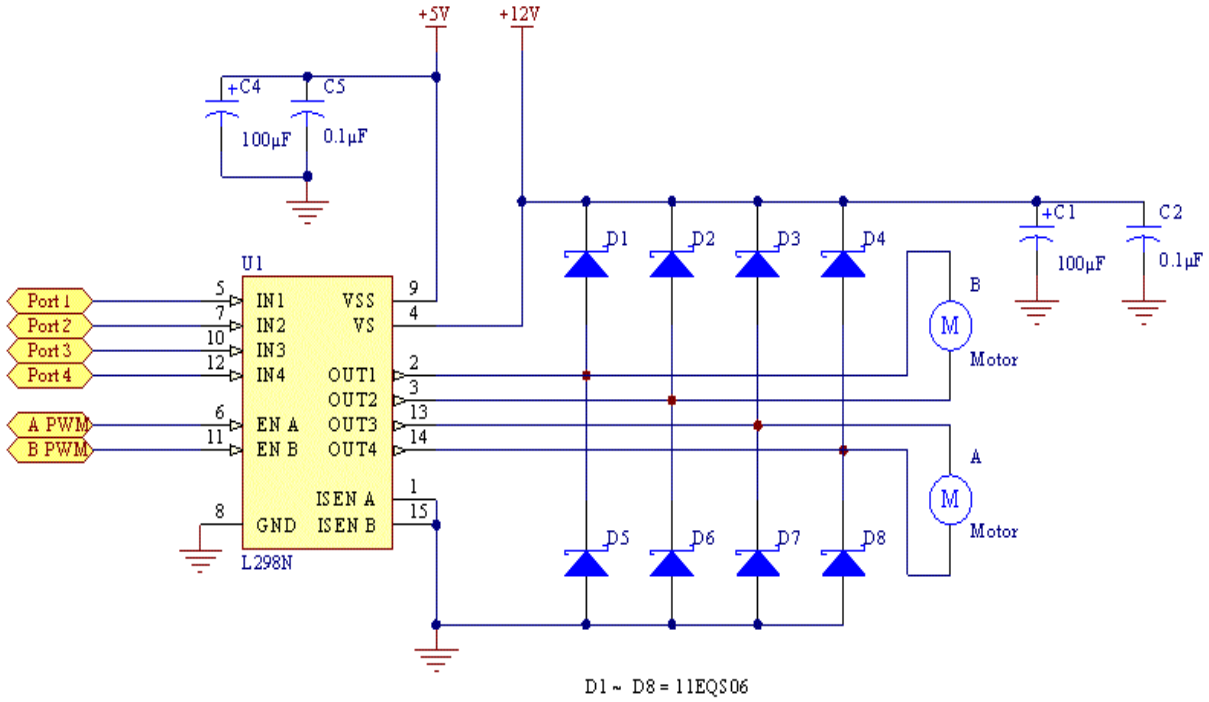
1. Išjungtas, variklis gali būti laisvai sukamas – visi jungikliai išjungti.
2. Įjungtas, variklis sukasi pagal laikrodžio rodyklę – jungikliai S1 ir S4 įjungti.
3. Įjungtas, variklis sukasi prieš laikrodžio rodyklę – jungikliai S2 ir S3 įjungti.
4. Įjungtas, variklis stabdymo režime – jungikliai S1 ir S3 arba S2 ir S4 įjungti.

Reiktų atkreipti dėmesį, kad H tiltas turi būti valdomas gana preciziškai, nes vienu metu įjungus jungiklius S1 ir S2 arba S3 ir S4 įvyksta trumpasis maitinimo šaltinio jungimas, o tokios situacijos pasekmės reikalauja remonto.

Kadangi varikliai mašinėlėje yra du (vienas skirtas greičiui valdyti, kitas krypčiai), todėl reikia ir dviejų tiltelių. Dėl dviejų tiltelių poreikio varikliams valdyti, išsirinkau „L298N Dual H-Bridge Motor Controller" modulį.

Pagrindiniai integrinio grandyno L298N parametrai :

- 2 H tilteliai
- Maitinimo įtampa 7-12V
- Korpuso dydis 4x4,5x3 mm



2.3.2.2 pav. Strukturinė L298N Dual H-Bridge modulio schema

2.4.2pav. pateikta integrinio grandyno L298N struktūrinėschema, kurioje matome du atskirus H tiltus ir loginį valdymo bloką. Pirmojo H tilto (ir pirmojo variklio) valdymui skirti loginiai įėjimai IN1 ir IN2, o antrojo – IN3 ir IN4.

2.2.2 Ryšio sąsajos modulio pasirinkimas

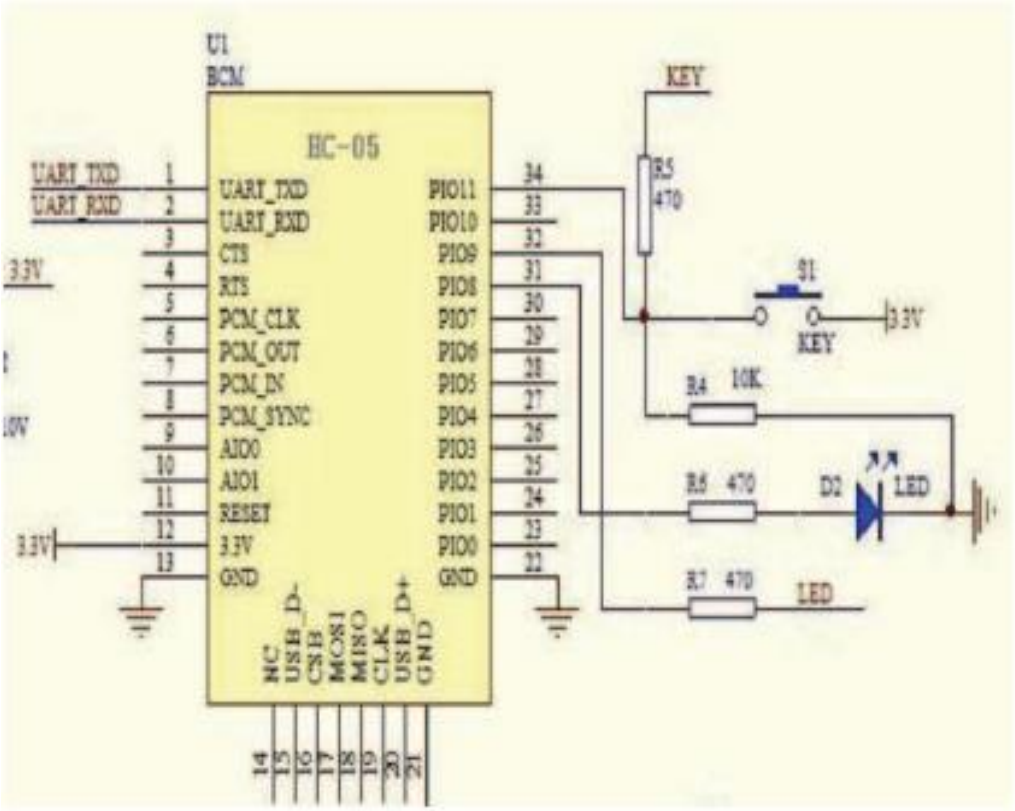
Pasirinkau kaip pultą panaudoti mobilųjį telefoną, nes darosi populiaru valdyti viską išmaniuoju telefonu. Apsvarsčius alternatyvas (GSM tinklą, Wifi ryšį ir bleutooth ryšį), nusprendžiau valdymą sieti su mašinėlę per Bluetooth HC-05 modulį. Bluetooth ryšys dažnai jau yra susietas su programomis mobiliajame telefone, todėl šio ryšio pasirinkimas leidžia išvengti programos konfigūravimų, kurie yra pakankamai sudėtingi. Taip pat šis ryšio modulis lengviausiai užprogramuojamas ir prijungiamas prie mikrovaldiklio.

Pagrindiniai parametrai:

- Maitinimo įtampa 5V;
- Palaikomos duomenų perdavimo greitis: 1200, 2400, 4800, 9600, 19200, 38400, bps

2.2.2.1 pav.Bluetooth HC-05 modulis

Kaip matyti iš 2.2.2.1 pav. Šis modulis turi tik 4 kojeles, kurias reikia prijungti prie mikrovaldiklio. Pažiūrėjus į principinę schemą, esančia apačioje, matome kad 1 koja (TXD) skirta siusti informaciją iš valdiklio, o 2 koja (RXD) skirta informacijai gauti. Taip pat matyti jog maitinimo įtampa jungiama +3,3 V į 12 koją, o žemės kontaktas prijungiamas prie 21 kojos



2.2.2.2 pav.Bluetooth HC-05 modulio schema.

Bluetooth HC-05 modulis išsiskiria tuo kad gali veikti kaip siųstuvas ir kaip imtuvas.

2.2.3 Baterijos pasirinkimas

Baterijai keliami reikalavimai neapibrėžia nei baterijos talpos nei tipo. Svarbiausias aspektas baterijos įtampa, ir daugkartinis panaudojimas. Ličio polimerų baterijos yra populiariausios tarp radijotechnikos megėjų, nes šio tipo baterijos šiuo metu turi didžiausią pranašumą prieš senojo tipo baterijas, tokias kaip NiCad (nikelio-kadmio) ar NiMH (nikelio-metalo hibrido).

Privalumai:

- LiPo baterijos yra lengvos.
- LiPo baterijos turi didžiulę talpą.
- LiPo baterijos turi didžiulį iškrovos koeficientą, dėl ko gali būti naudojami net stipriausiems elektros varikliams sukti.
- Tinkamai prižiūrint savo bateriją ji atlaikys iki 300-400 krovimo ciklų.

Kitaip tariant LiPo baterijos pasižymi geru svorio ir talpos santykiu, be to gali būti įvairios formos ir dydžio. Visi šie išvardinti privalumai yra svarbūs bet kokiai per atstumą valdomai mažai sistemai.

Tačiau šios baterijos turi ir trūkumų:

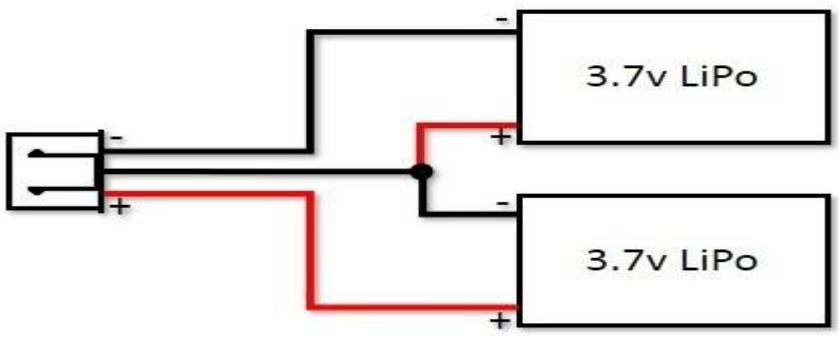
- LiPo baterijos yra pakankamai brangios, lyginant su NiCad ar NiMH baterijomis.
- Dėl lakiųjų elektrolitų naudojimo šiose baterijose, jos gali greitai užsidegti ar net sprogti.

Norint įkrauti šio tipo baterijas, reikia turėti specialų įkrovimo bloką. Pasirinkus netinkamą voltražą, baterija sugenda, galimas net baterijos užsidegimas nes jo yra labai degios.

Baterijos pasirinkimą įtakojo tai kad tiltelio maitinimui reikia 7 - 12 V įtampos maitinimo. Baterijos matmenys turi buti kaip įmanoma kompaktiškesni. Todėl rinkausi didelės talpos 2 celių greitai įkraunamą ličio bateriją, kaip matyti iš 2.2.3.1 pav, kiekviena celė turi 3,.7 V ir yra sujungtos nuosekliai viena kitai. 7,4 V potencialų skirtumas gaunamas tik tarp abiejucelių. Dėl plataus pasirinkimo, kritiniu veiksniu tapo kainos ir talpos santikis.

Pagrindiniai pasirinktos baterijos parametrai:

- Talpa: 1300mAh
- Įtampa: 7.4V
- Svoris: 76g
- Matmenys: 72x36x13mm

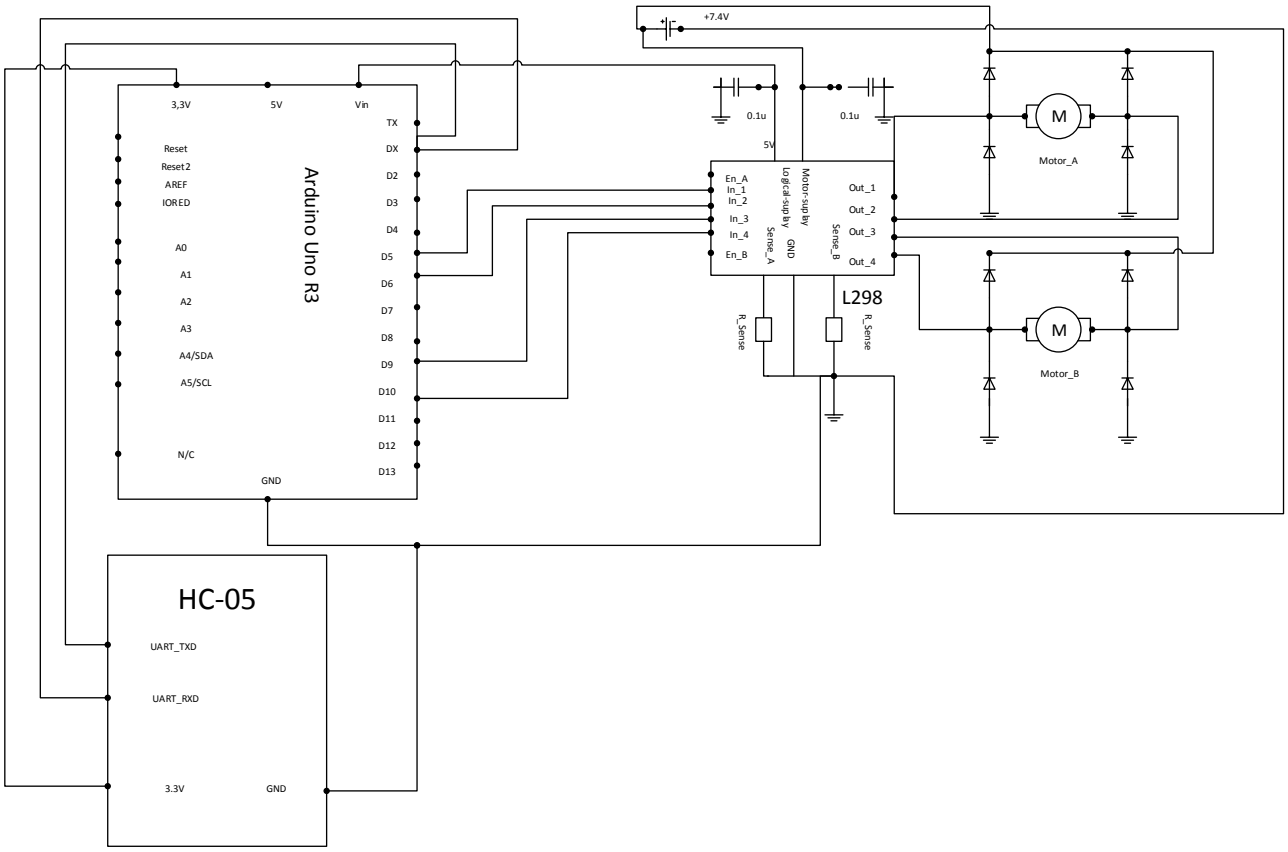


2.2.3.1 pav. Dviejų celių baterijos schema.

2.2.4 Sistemos principinė schema

Norint kad sistema veiktų, reikia visas sistemos dalis tvarkingai sujungti su mikrovaldiklio platforma. 2.2.4.1 pav atvaizduoja kaip prie Arduino Uno platformos yra prijungiamas valdymo tiltelis, ryšio modulis, baterija. Šias pasirinktas dalis jau aptariau prieš tai. Baterijos maitinimas, jungiamas prie tiltelio maitinimo, kuris dirba maitinant jį 7-12 V įtampa. Tiltelis savo struktūrinėje schemoje viduje turi stabilizatorių, kuris skirtas stabilizuoti +5V įtampą, skirtą maitinti mikrovaldiklį ar kitą prietaisą, taip tarsi papildomai jį apsaugant nuo sudegimo užsitrumpinus grandinei. Todėl iš tiltelio maitinamas mikrovaldiklis ir Bluetooth ryšio modulis. Baterijos žemės laidas prijungiamas prie tiltelio, o nuo tiltelio prijungimą prie kurio nors iš mikrovaldiklio žemės kontaktų, taip įžeminant visą mikrovaldiklį, kadangi visos mikrovaldiklio žemės sujungtos kartu. Ryšio modulis, kaip jau buvo kažkur minėta, su mikrovaldikliu jungiasi TXD ir RXD kontaktas. TXD kontaktas turėtų būti suprantamas kaip siunčiantis duomenis, RXD kaip priimantis. Bet reikia nepamiršti kad mikrovaldiklio ryšio kontaktai irgi skirti priimti ir siųsti duomenis. Tik jeigu iš mikrovaldiklio duomenys yra siunčiami iš TX kontakto, tada logiška kad tie duomenys turėtų būti gaunami į ryšio modulio RXD kontaktą, skirtą priimti duomenis. Svarbu nesumaišyti kontaktų, nes juos sumaišius galimas ryšio modulio schemos sugedimas. Taip pat dar liko mikrovaldiklio kintamos įtampos išėjimai, kurie prijungiami prie tiltelio. Kintamos įtampos išėjimai (vadinami pin). Pasirinktus išėjimus reikės užprogramuoti programoje, todėl nėra skirtumo kuriuos pasirinkti. Visi išėjimai yra vienodi.

.



2.2.4.1 pav. Principinė savaeigės mašinėlės elektrinė schema

3 Mikrovaldiklio programavimas

Pirmiausia buvo parsisiųsta programavimo programa skirta „Arduino“mikroprocesoriams programuoti: „Arduino 1.0.5“ programa iš www.arduino.cc oficialaus internetinio puslapio. Vienas iš programos privalumų yra tai, kad jos atmintyje jau galima rasti nemažai skirtingų pavyzdžių, skirtų susipažinti ir išmokti dirbti su šia programa. Taip galima ne tik įsitikinti jog mikroprocesorius veikia, bet ir sužinoti kokia programos struktūra ir kaip veikia atskiros funkcijos.

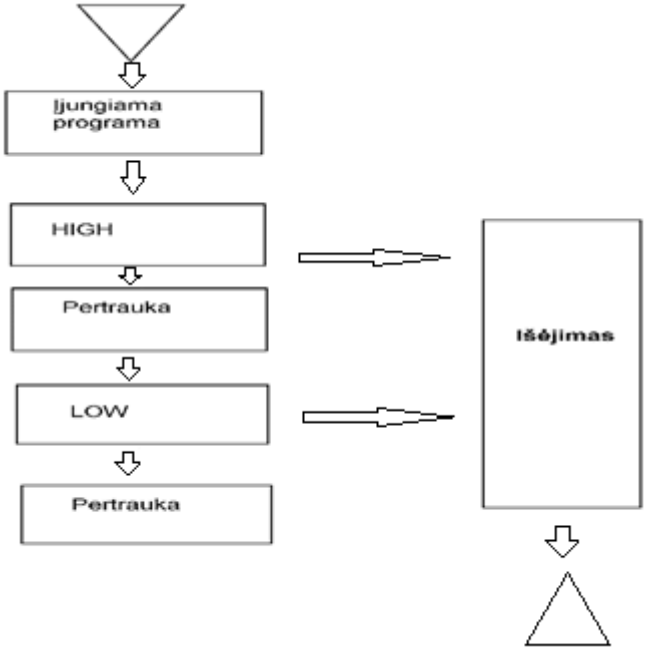
Bet kokia rašoma programa turi atitikti tam tikrą struktūrą. Šiuo atveju programuojant bet kurį Arduino platformos mikrovaldiklį, programa yra sudaryta iš kintamųjų ir konstantų aprašymo, po to seka void setup(){} ir void loop(){} funkcijos. Void setup(){} funkcija paleidžia aprašomas funkcijas tik įsijungus programai. Void loop(){} funkcijoje aprašomos norimos funkcijos, kurios veiks visa laiką kol programa bus įjungta.

```
int ledpin=13;
```

```
void setup()
{
 pinMode(ledpin,OUTPUT);
}
void loop()
{
  digitalWrite(ledpin,HIGH);
  delay(1000);
  digitalWrite(ledpin,LOW);
  delay(1000);
```

Tai pačios paprasčiausios programos kodas, kuriame galima aiškiai matyti minėta programos struktūrą. Visų pirma priskiriama ledpin reikšmė (13). Toliau nustatoma kad bus naudojama išėjimo reikšmė jau pasirinktame 13 pin išėjime. Ir nustatoma kad bus naudojama skaitmeninė reikšmė, tai reiškia kad išėjimo būsena gali būti įjungta arba išjungta, paduodant aukštą arba žemą signalus. Void loop(){} funkcijoje įrašytos komandos visą laiką veikia, tai reiškia kad įvykdžius pirmą funkciją, tai yra įjungus ledą, jis degs musų nurodytą laiko tarpą.(šiuo atveju 1000ms=1s). Vėliau vieną sekundę bus išjungtas ir ciklą vėl kartosis. Ši kodas yra labiau mokomojo pobūdžio. Konkrečiu šiuo atveju keisis į Arduino Uno mikrovaldiklį jau integruotos led lemputės mirgėjimo dažnis. Ši lemputė yra sujungta su 13 išėjimu (pinu). Pakeitus išėjimo reikšmę į kitą išėjimą, tektų prie to išėjimo prijungti lemputę.Kaip jau žinom Led lemputes darbui reikalingi 2V. Padavus didesnę įtampą lemputė greitai perdega. Todėl prijungiat led lempute, visada reikia prie jos prijungti papildomą varžą, kuri stabilizuotų įtampą, kuri tenka led lemputei. Tai reikalinga tam kad HIGH (aukštas signalas) išėjime atitinka +5V įtampą. LOW atitinka 0V ir yra vadinamus žemu signalu. Tokia paprasta programa nesunkiai galima suprogramuoti šviesoforo darbą. Tereikėtų prie mikrovaldiklio prijungti tris lempas, programoje nustatyti išėjimus ir lempų mirgėjimo ar įjungimo išjungimo laikus.

Trumpai tariant programa įsijungia ir pradeda veikti. Signalas paduodamas į led lemputę aukšo lygio, tada atsijungia per tam tikrą laiko tarpą ir paduodamas žemo lygio signalas, kuris po nustatyto laiko atsijungia. Viskas vyksta ciklišKai kol programa yra įjungta. Veikimo algoritmas, pavaizduojantis schemos veikimą, atvaizduotas 3.1 pav.



3.1 pav. lemputės mirgėjimo algoritmas

3.1 Bluetooth modelio užprogramavimas

Bluetooth modelis savyje turi mikroprocesorių, kuris valdo jo darbą ir saugo informaciją. Norint suderinti jį su mikrovaldikliu, reikia atskirai jį užprogramuoti. Bluetooth modelis neturi USB ar kažkokios kitos sąsajos prie kompiuterio. Prie kompiuterio jis yra prijungimas per Arduino mikrovaldiklį, panaudojant jį kaip portą perduoti informacijai. Tuo metu mikrovaldiklis neatlieka jokių funkcijų, jis veikia tuščios eigos rėžimu. Modulio programavimas vyksta ta pačia programa „Arduino 1.0.5“. Prie mikrovaldiklio prijungiamas ne į ryšio kanalus, bet į paprastus kintamos įtampos išėjimus, kurias bus programuojama modulio vardas, bei failų priėmimo ir siuntimo funkcijos.

```
#include <SoftwareSerial.h>
SoftwareSerial BTSerial(10, 11);
void setup()
{
  pinMode(9, OUTPUT);
  digitalWrite(9, HIGH);
  Serial.begin(9600);
  Serial.println("Enter AT commands:");
  BTSerial.begin(9600);
}
void loop()
{
  if (BTSerial.available())
```

```
 Serial.write(BTSerial.read());
  if (Serial.available())
    BTSerial.write(Serial.read());
}
```

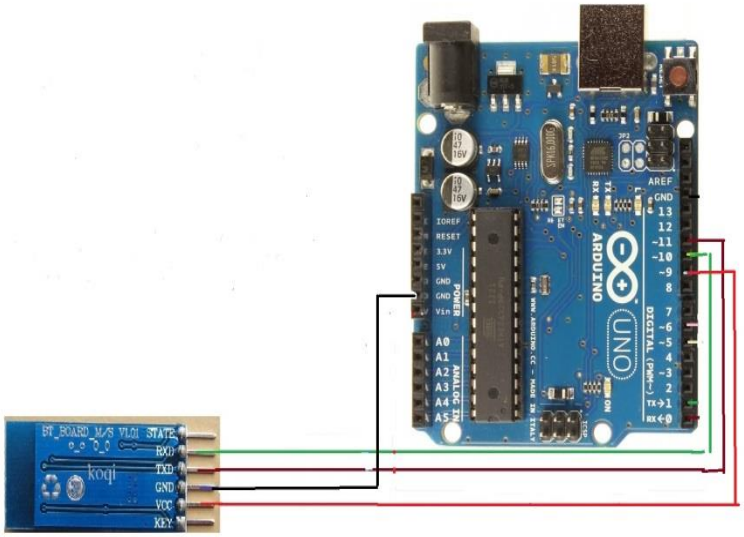
Šis kodas naudojamas nustatyti savo ryšio pavadinimą ir įrašyti programą į bluetooth modeli, kad kai šis įtaisas gaus maitinimą, jis nuolatos perduotų duomenis iš ryšio modulio į valdiklį ir iš programavimo programos nuosekliojo porto monitoriaus nuskaitytų rašomas reikšmes ir nusiųstų jas į ryšio kontrolerį, kurioje bus gauta informacija išsaugoma. Bus gautas ryšio pavadinimas, taip pat nustatyta kad bet kuris įrenginys turintis bluetooth ryšio galimybę, galėtų keistis informacija tarpusavyje.

```
#include <SoftwareSerial.h>
```

Eilutė kurioje nurodome funkcijų biblioteką, kurios funkcijas naudosim kuriant programą. Toliau nusakome iš kurių išėjimų bus perduodama programinės įrangos informacija.(10,11 išėjimai.).Paprastai kalbant programa yra užprogramuojama kad veiks prisijungta prie Bluetooth modulio. Kai bus prie jo prisijungta, modulis nuskaitys duomenis, tada perduos juos į mikrovaldiklį. Tai vyks ciklišKai. Perdavus vienus duomenis, po pasirinkto laiko tarpo vėl bus nuskaitomi kiti ir vėl perduoti valdikliui.

Prijungimo schema atvaizduota 3.1.1 pav. Nuo 9 išėjimo prijungiamas maitinimas ryšio moduliui, ir nustatoma sistemos veikimo sparta kuri yra sulyginama su ryšio modulio sistemos greičiu. Toliau norint kažką programuoti, reikia įrankių juostoje išsikviesti nuoseklioje porto monitorių. Tai programos langas į kurį galima rašyti iš kompiuterio įvairius simbolius, reikšmes kurios perduodamas į sistemą. Norint pakeisti ar nustatyti savo ryšio pavadinimą atsidariusiame nuosekliojo porto lange reikia įrašyti komandą „AT+namevardenispavardenis“ ir paspausti enter klaviatūros klavišą. Taip pat yra funkcijos kurias įvedus galima nustatyti kad ryšio modulis visada veiktų, būtų prieinamas bet kuriam vartotojui, kad ryšys būtų tik gaunamas arba tik siunčiamas, priklausomai nuo to, ka reikia gauti norint realizuoti kuriama sistemą.

Daugiau apie komandas galima pasižiūrėti internete esančiam priede skirtam būtent įvairioms funkcijoms aprašyti. Priedas yra pakankamai didelis, todėl tarp priedų spausdinti jį būtų netikslinga. Ji galima rasti adresu [http://www.linotux.ch/arduino/HC0305_serial_module_AT_commamd_set_201104_revised.pdf].



3.1.1 pav. Ryšio modelio prijungimas prie mikrovaldiklio, skirtas programuoti patį ryšio modulį.

3.2 Galimų važiavimo krypčių aprašymas

Reikėjo susipažint su tiltelio veikimu varikliams valdyti. Prijungus tiltelį sukūriau programą kuri leistų keisti variklio sukimosi greitį vienam varikliui. Bandymai buvo atliekami su dviem nuolatinės srovės varikliais, taip imituojant mašinos greičio ir krypties kitimą. Nustatomi 4 išėjimai, kurias valdomi abu varikliai. Vienas iš Arduino platformos valdiklių trūkumų yra tai kad vienu metu gali veikti tik viena jo komanda. Parašiau programą kurioje lengvai galėčiau keisti norimos kryptys komandos veikimą, kadangi kuriant pilną kodą reikės aprašyti visas įmanomas ar bent jau pagrindines kryptis kuriomis galės judėti kuriamas savaeigis modelis.(važiavimas į priekį, atgal, stovėjimas vietoje, sukimas į kairę, į dešiną, važiavimas į dešiną ir t.t.)

Kiekvienas variklis valdomas dviejų išėjimų logika. Labai svarbu kad abu išėjimai tenkantys varikliui vienu metu nepaduotų aukšto signalo, nes taip sudegintų variklio apsukas ir sugadintų variklį. Sakykim 5 ir 6 išėjimo signalai vienu metu abu negali būti aukšti, kaip ir 9,10 išėjimų signalai. Vienu metu tik vienas iš jų gali būti HIGH reikšmės ir tai nulems variklio sukimąsi į vieną pusę. Pakeitus HIGH reikšmę į kito išėjimo signalą, variklis jau suksis į kitą pusę. Taip sukuriamas variklių valdymas.

```
void setup() {
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(9, OUTPUT);
  pinMode(10, OUTPUT);
}
```

```
void loop() {
   digitalWrite(5, HIGH); // desine
   analogWrite(6, 0);
   digitalWrite(10, LOW);
   analogWrite(9, HIGH);
 delay (1000);
}
```

Čia 5, 6, 9, 10 yra nurodyti pin‘ai (išėjimai) į kuriuos sujungti abu varikliai. 5 ir 6 pin‘ai - pirmas variklis, o 9 ir 10 pin‘ai – antras variklis. 5-6 išėjimai atsakingi už važiavimo į priekį arba atgal kryptį, kitaip sakant greitį. 9-10 išėjimai valdoma priekinius ratus, nuo kurių priklauso mašinėlės važiavimo kryptis. Norint imituoti krypti “mašinėlė važiuoja į priekį sukdama į dešinę” variklis atsakingas už važiavimą pirmyn-atgal turi dirbti važiavimo į priekį rėžimu(5-HIGH), o variklis atsakingas už krypties keitimą, turi suktis į dešinę(9-HIGH). Taip keičiant šias reikšmes gaunamos įvairios važiavimo trajektorijos.

3.3 Pilnas sistemos kodas

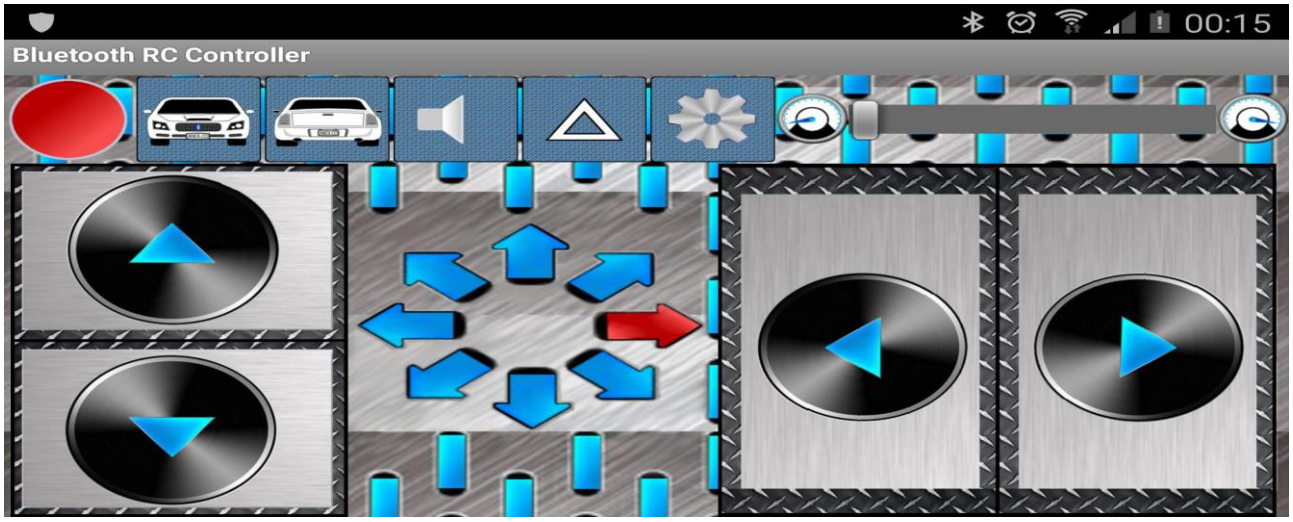
Visų pirma norint sukurti pilną sistemos kodą, reikia turėti valdymo programą telefone, iš kurios bus valdoma sistema. Yra du variantai, kaip gauti tokią programą. Galima kurti savo programą arba iš “Google Play” parduotuvės parsisiųsti jau sukurtą programą, kurią vistiek reikės suderinti su valdoma savaeige mašinėle. Išanalizavęs esamas programas, nusprendžiau kad mano lūkesčiams įgyvendinti turėtų užtekti jau esamos programos. Iš “Google Play” internetinės programinės įrangos parduotuvės parsisiunčiau nemokama “Bluetooth RC Controller” programėlę, kurios darbalaukį galima pamatyti 3.3.1 pav, kuriame matyti norimos važiuoti važiavimo kryptys ir akselerometras, kurį panaudosiu greičiui valdyti. Turint programą reikia ją išsianalizuoti, ir sužinoti rodyklių, kurios imituoja važiavimo krypties pasirinkimą, reikšmes. Tai daroma įsijungus programą ir joje susiradus Bluetooth ryšio modelį, susiejus jį su telefonu. Atsidarius Arduino programavimo langą užtenka įvesti dvi komandas, kurios reikštų kad visa informacija gauta paspaudus krypties rodyklę yra nuskaitoma ir atvaizduojama nuosekliojo porto monitoriaus lange kompiuteryje.

```
command += (char)Serial.read();
delay(50);
Serial.println(command);
```

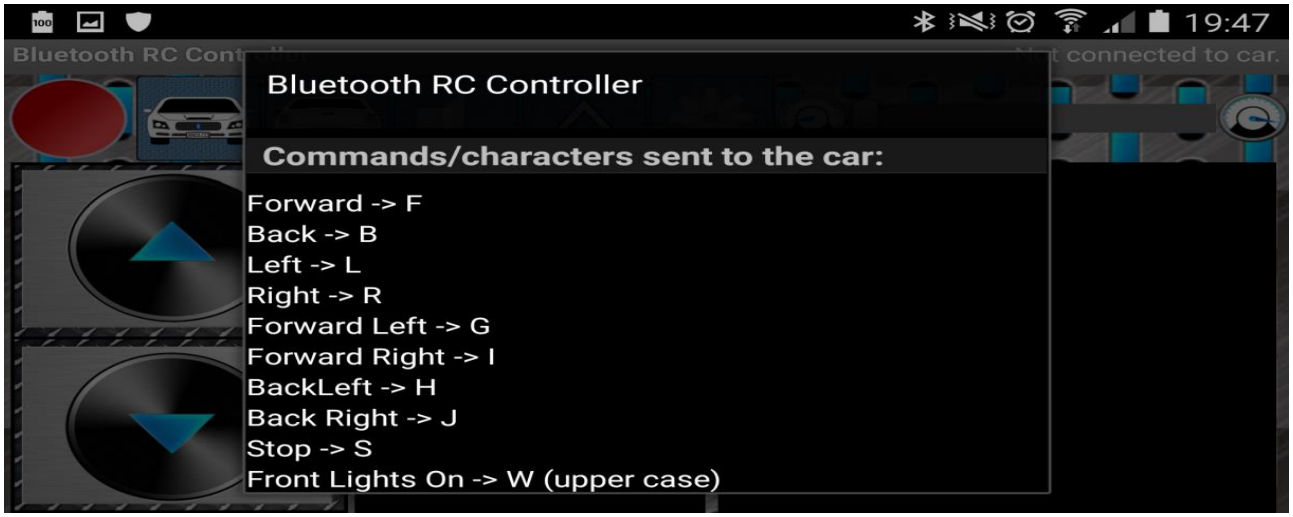
Šios komandos atsakingos už duomenų nuskaitymą ir atvaizdavimą.

Kitas būdas kaip galima sužinoti krypčių reikšmes, yra atsidaryti programa, atsidaryti programos nustatymus ir ten susirasti skyrelį Commands/Characters (3.1.2 pav). Jame aprašomos

naudojamos reikšmės. Išskirtina tai kad didžiosios ir mažosios raidės yra atskiras simbolis, turintis kitą reikšmę.



3.1.1 pav. Bluetooth RC Controller progromos darbalaukis



3.1.2 pav. Bluetooth RC Controller progromoje esančios raidžių reikšmės

Taip susižinojus visas reikšmes, kurios atsakingos už modelio valdymą, galima pradėti rašyti programą. Programos struktūra turi išlikti tokia pati kaip jau minėjau anksčiau, tik programa, kadangi bus didesnė, turės dagiau funkcijų kintamųjų ir bus žymiai sudėtingesnė.

Programa prasideda aprašant reikiamus kintamuosius ir nustatant reikiamas bibliotekas, kurių funkcijos bus naudojamos.

```
#include <SoftwareSerial.h>
#include <Serial.h>

long lastincrease = 0;
long now = 0;
int spd = 65; // galiniu ratu greitis
```

```
int spd1 = 180; // priekiniu ratu pasisukimo greitis
String command = "";
```

SoftwareSerial.h yra būtina biblioteka kad veiktų nuskaitymo ir duomenų perdavimo funkcijos. Serial.h biblioteka apibrėžia naudojamas if funkcijas. Toliau reikia nusistatyti pradines konstantas, tam kad būtų galima sudarinėti funkcijas, kurios apdorotų konstantų reikšmes. “spd” ir “spdl” variklių valdymui naudojamos konstantos kurios apibrėžia paduodamos varikliams įtampos santykį, keičiant impulso trukmę. Arduino mikrovaldiklis gali keisti impulse trukmę 255 skirtingom reikšmėm. Sakykim pasirinkus 65 reikšmę, tai reiškia kad maksimali įtampa varikliui yra 65/255 dalys maksimalios įtampos kurią gali perduoti valdymo grandinė. “lasincrease” konstanta naudojama nuosekliam variklių galios didinimui, kas leis varikliui įsibėgėti tam tikrais intervalais, o ne iš karto paduodant maksimalią reikšmę. Tai yra naudinga varikliui ir galios perdavimo elementui (grandinei), nes jie ne taip greitai sugenda.

Sekanti dalis void setup() funkcija, kurioje reikia aprašyti komandas, kurios įsijungs įsijungus varikliui.

```
void setup()
{
  pinMode(5, OUTPUT); // vairas desine
  pinMode(6, OUTPUT); // vairas kaire
  pinMode(9, OUTPUT); // greitis i prieki
  pinMode(10, OUTPUT); // greitis atgal
  Serial.begin(9600);
}
```

Šioje dalyje aprašomi išėjimai, kurie bus naudojami. Keičiant jų reikšmes, bus gaunamos skirtingos važiavimo kryptys, kurias reikia aprašyti toliau programoje. Kryptys kurias as sugalvojau panaudoti:

- Tiesiai
- Tiesiai ir į kairę
- Tiesiai ir į dešinę
- Atgal
- Atgal ir į kairę
- Atgal ir į dešinę
- Pasuka tik į dešinę
- Pasuka tik į kairę

- Sustoja

```
void kryptis7() // sustoja
{
  digitalWrite(5, LOW); // nesuka i desine
  digitalWrite(6, LOW); // nesuka i kaire
  digitalWrite(9, LOW); // nevaziuoja i prieki
  digitalWrite(10, LOW); // nevaziuoja atgal
}
```

Kad stovėtų visi išėjimo signalai nustatyti į žemos įtampos reikšmes. Norint gauti kryptį tiesiai, reikia į 9 išėjimą paduoti aukšto lygio signalą. Analogiškai su kitomis kryptimis.

Toliau pereinama prie funkcijos void loop() realizavimo. Šioje dalyje rašomos funkcijos, kurias bus galima atlikti visą laiką, kai sistema bus įjungta. Visų pirma svarbu kad sistema pradėtų dirbti tik tuo atveju, kai prie jos prisijungs valdymo prietaisas.

```
void loop()
{
  if (Serial.available())
  {
command += (char)Serial.read();
    //delay(50);
    Serial.println(command);
```

Šis algoritmas leis, kad programa pradėtu veikti tik prisijungus prie bluetooth ryšio. Kai bus prisijungia, programa nuskaitys spaudžiamas reikšmes, kurios bus priskiriamos vėliau, ir jas perduos mikrvaldikliui. Delay(50) nusako kad reikšmių nuskaitymo dažnis bus perduodamas kas 50ms. Tai reiškia kad per sekunde sistema galės perduoti 20 reikšmių. Aišku šitą dažnį dar galima keisti, didinti arba mažinti, bet sunkiai įmanoma per sekundę prispausti tiek skirtingų reikšmių.

Toliau reikia kiekvienai reikšmei valdymo pulte priskirti po atitinkama komandą iš prieš tai išvardintų komandų (tiesiai, atgal ir t.t.)

```
if (command == "F") // vaziavimo i prieki siuntimas
    {
      kryptis1();
    }
```

F reikšmė valdymo pulte atitinka rodyklę į priekį, todėl šia reikšmę priskiriame komandai ‚kryptis1()‘, kuri sukonfigūruota taip kad valdymo impulsai būtų paduoti atitinkamam varikliui, gauti važiavimą norima kryptimi. Taip analogiškai priskiriamos visos valdymo pulto reikšmės.

Greičio kitimas taip pat aprašomas panašiai. Akselerometras jau yra suskirstytas į 10 skirtingų padėčių, kurių raidinės reikšmės randamos programoje. Per bandymus buvo nustatyta kad mašinėlė pradeda judėti padavus jai 65/255 dalies maksimalios įtampos. Maksimalią įtampą atitinka 255/255. Vertė. Tačiau visos kuriamos sistemos dažnai neveikia maksimaliu galingumu, dėl to kad būtų užtikrintas ilgesnis sistemos darbas. Todėl palikdamas rezervą nusprendžiau maksimalią galią sumažinti iki 235, kas leistų į lygias 10 dalių padalinti akcelerometro reikšmes, taip sukuriant skirtingą valdymo greitį. 235-65=170; 170/10=17. 17- minimalus greičio kitimo žingsnis.

Akselerometro ribojamos maksimalaus greičio reikšmės:

- 65/255
- 82/255
- 99/255
- 116/255
- 133/255
- 150/255
- 167/255
- 184/255
- 201/255
- 218/255
- 235/255

```
if (command == "0")
   {
    spd = 65;
   }
if (command == "1")
   {
    spd = 82;
   }
```

Tokiu principu valdomas greitis. Esant 0 akselerometro padėčiai, minimalus greitis 65/255 maksimalaus galimo greičio.

Ir paskutinis programos dalykas, kuris labai svarbus ir saugo sistemą, yra tai, kad greitis neaugtų šuoliškai. Kad nebūtų laužomos rotoriaus dalys, kurios perduos sukimosi greitį,

greitis turi kisti ne tik iki maksimalios pasirinktos reikšmės, bet tą pasirinktą reikšmę turi pasiekti įsibėgėdamas.

```
if (speedfincreasing < spd)
   {
     speedfincreasing = speedfincreasing + 5;
   }
 }
 delay(50);
```

Greitis auga +5/255 dalimis 20 kartų per sekundę greičiu. Norint kad augtų greičiau, tereikia pakeisti +5 reikšmę į didesnės vertės skaičių. Bet kokiu atveju pavara bus ne taip laužoma ir ne taip greitai nusidėvės, panaudojus tokį programos kodą.Dauguma sistemos derinimo darbų buvo gaunami išbandant sistemą. Rašant programos kodą, teko atlikti įvairius bandymus, siekiant parinkti tinkamiausias reikšmes sistemai valdyti. Pilną programos kodą su visomis komandomis galima rasti prieduose.

4 Eksperimentinis tyrimas

Eksperimento tikslas – išbandyti sukurtos mašinėlės valdymo ribas uždaromis ir atviromis patalpų sąlygomis bei nustatyti įtampos keitimo veikimą, naudojant PWM.

Eksperimentinių tyrimų metu buvo fiksuojamas impulso pločio moduliacijos veikimas, prie skirtingų valdymo reikšmių ir tos reikšmės lyginamos su apskaičiuotomis teoriškai. Taip pat buvo matuojamas sistemos veikimo nuotolis.(atstumas nuo kurio nutrūksta ryšys tarp valdymo pulto ir savaeigio modelio)

Impulso pločio moduliacija arba PWM, yra naudojama gauti analoginiam signalui, iš skaitmeninio valdymo signalo. Keičiant impulso laiką, keičiasi ir vidutinė įtampos vertė, nes keičiasi impulso plotis. Kiekvienas funkcijos analogWrite () intervalas gali būti bet koks imtinai nuo 0 iki 255 reikšmės (Arduino Uno R3 mikrovaldiklis tokiu santykiu leidžia keisti intervalo reikšmes). Ši reikšmė, tai santykis impulso pločio su visu periodu. Padavus reikšmę analogWrite (255), išėjime gausim maksimalią reikšmę. Padavus analogWrite (0), gausim minimalią reikšmę (0V). Padavus analogWrite (127), gausim 50% maksimalios reikšmės. Taip parinkus analogWrite() reikšmę yra keičiamas PWM, kuris yra skirtas varikliams valdyti.

4.1.1 Bandymai uždarose patalpose

Buvo 10 kartų skirtingose patalpos vietose matuojamas atstumas, nuo kurio dingdavo sistemos valdymas. Bandymai atliekami gyvenamojo daugiabučio blokinio tipo name. Valdymo signalui laisvai trukdo sklisti betoninės buto pertvaros. Valdymas vykdomas iš kitų kambarių, ir stebimas kokiu atstumu nutrūko valdymo ryšys.

4.1.1.1 lentelė. Ryšio nutraukimo atstumas gyvenamajo namo patalpose.

Bandymo nr.	Atstumas, m
1	7,4
2	11,2
3	9,8
4	9,3
5	10,9
6	8,1
7	7,8
8	10,2
9	9,4
10	8,6

Skaičiuojamas visų reikšmių aritmetinis vidurkis. Taip nustatoma vidutinis sistemos veikimo atstumas uždaroje patalpoje. Apskaičiuota 9,27m valdymo zona.

4.1.2 Bandymai lauke

Taip pat 10 kartų buvo matuojamas atstumas lauke valdant sistemą. Lauke sistemos perdavimo signalas nėra slopinamas jokių kliūčių ar pertvarų.

4.1.2.1 lentelė. Ryšio nutraukimo atstumas atviroje erdvėje.

Bandymo nr.	Atstumas, m
1	23,6
2	30,9
3	28,4
4	32,1
5	25,3
6	27,7
7	30,1
8	32,3
9	28
10	30,8

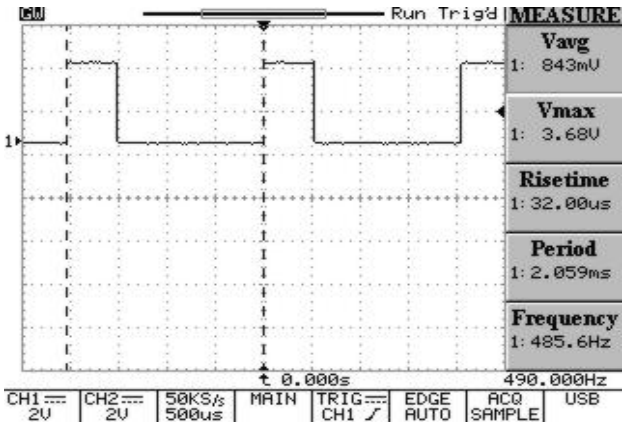
Skaičiuojamas visų reikšmių aritmetinis vidurkis. Taip nustatoma vidutinis sistemos veikimo atstumas atviroje aplinkoje. Apskaičiuota 28,92m valdymo zona.

4.1.3 Eksperimentinis tyrimas osilografu

Atliekant bandymus buvo išmatuota ir nustatyta kad viena baterijos celė yra išsikrovusi, todėl bandymus atlikau su kita cele, kuri buvo pilnai įkrauta.(gali būti baterijos gedimas arba tiesiog išsikrovusi baterija) Bandymui panaudojau veikiančia mašinėlės valdymo schema, vietoje variklio prijungus osilografo kontaktus.

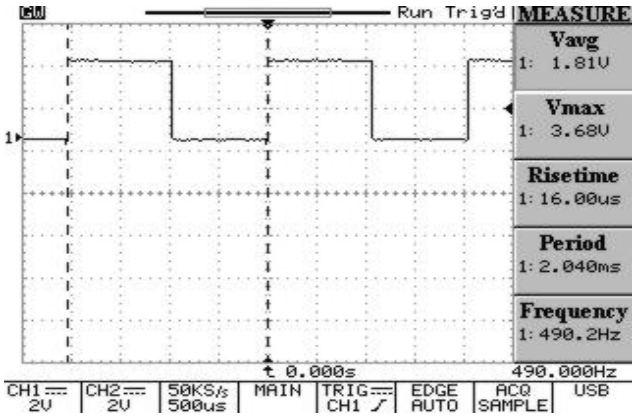
Išmatuota baterijos įtampos reikšmė su multimetru U =3,74 V

Toliau atlikinėjau įtampų nustatymus keisdamas analogWrite () reikšmes, parinkdamas jas tokias kaip ir programavimo metu. (65,82,99,116,133,150,167,184,201,218,235). Šių reikšmių nustatymas aprašomas 24 puslapyje, skyrelyje apie greičio kitimo algoritmą.



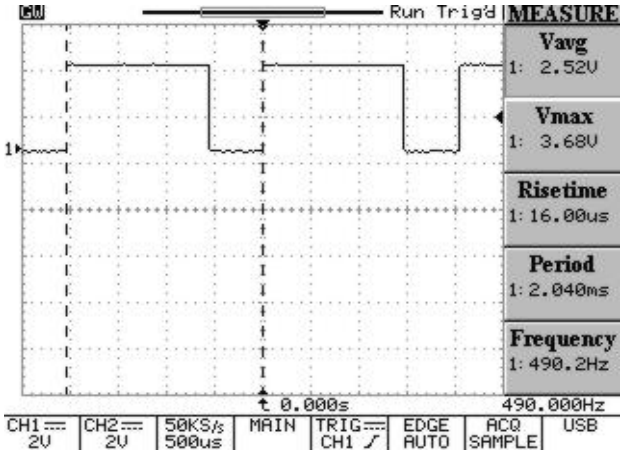
4.1.3.1 pav. analogWrite (65) įtampos osilograma

Vidutinė įtampos reikšmė, nustačius analogWrite (65) yra U=0,843V. Periodas T=2.059ms. Teoriškai periodas turėtų būti 2ms, gauti netikslumai dėl netiksliai nustatytų kursorių.



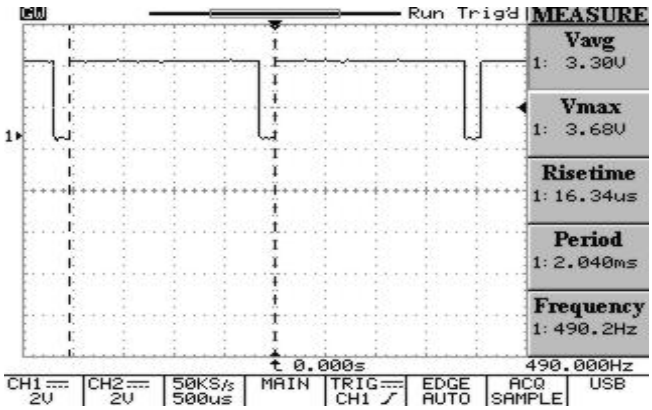
4.1.3.2 pav. analogWrite (116) įtampos osilograma

Vidutinė įtampos reikšmė, nustačius analogWrite (116) yra U=1,81V. Periodas T=2.040ms.



4.1.3.3 pav. analogWrite (167) įtampos osilograma

Vidutinė įtampos reikšmė, nustačius analogWrite (167) yra U=2,52V. Periodas T=2.040ms.



4.1.3.4 pav. analogWrite (235) įtampos osilograma

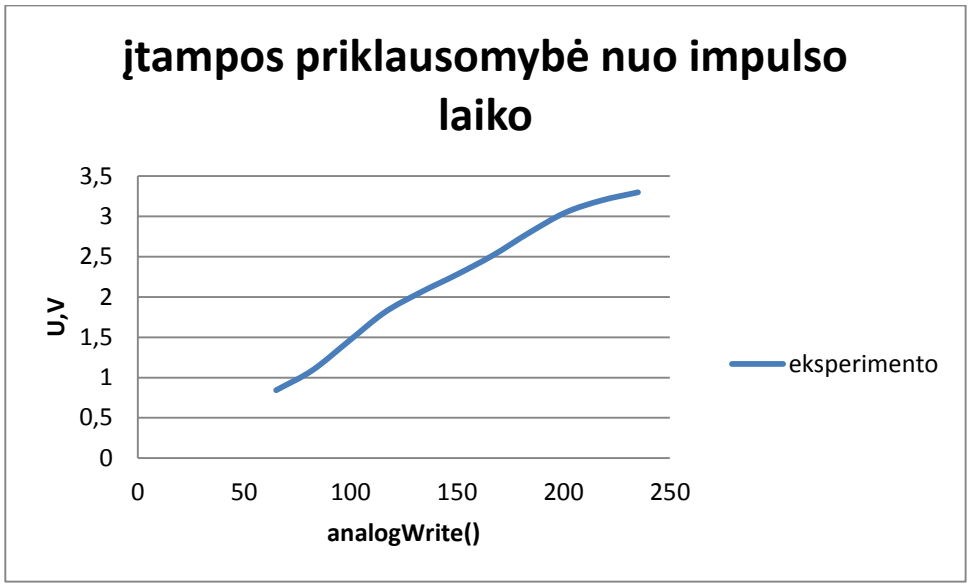
Vidutinė įtampos reikšmė, nustačius analogWrite (235) yra U=3,30V. Periodas T=2.040ms.

Vizualiai iš gautų grafikų galima matyti kaip keičiasi impulso plotas, keičiant impulso laiką.

Visos eksperimento metu nustatytos vertės surašomos į lentelę ir atvaizduojamos grafiškai.

4.1.3.1 lentelė. Eksperimento metu gautos reikšmės

analogWrite ()	U2,V eksperimento
65	0,843
82	1,09
99	1,45
116	1,81
133	2,06
150	2,28
167	2,52
184	2,8
201	3,05
218	3,2
235	3,3



4.1.3.5 pav. Eksperimentiškai gautos įtampos priklausomybė nuo impulso laiko

4.2 Teorinė dalis (skaičiavimai)

Vidutinės įtampos skaičiavimas. Kadangi bandymai buvo atliekami tik su viena baterijos cele, tai skaičiavimams taip pat naudojama vienos celes įtampa U=3,7V. Norint teoriškai apskaičiuoti efektinęįtampą, reikia maitinimo įtampą padauginti iš impulso įjungimo laiko ir padalinti iš pilno periodo laiko.

$$U_{ef}=U\ t/T; \qquad (4)$$

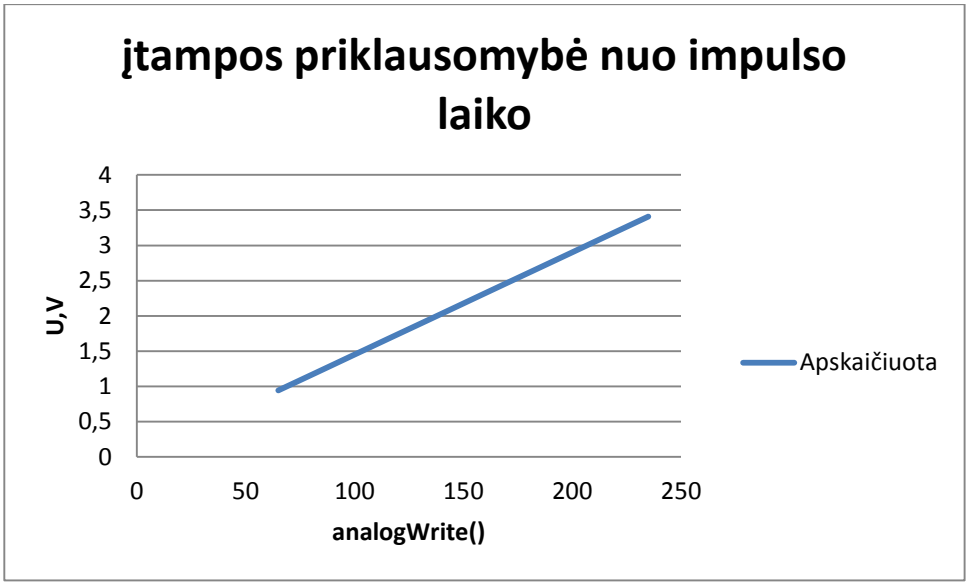
t – impulso įjungimo laikas periode ; T – periodas.

$$Uef := \frac{3.765}{255} = 0.943 \ V$$

Analogiškai skaičiuojamos visos kitos reikšmės. Gauti rezultatai surašomi į rezultatų lentelę ir atvaizduojami grafiškai.

4.2.1 lentelė. Teoriškai gautos reikšmės

analogWrite ()	U1,V apskaičiuota
65	0,943
82	1,189
99	1,436
116	1,683
133	1,929
150	2,176
167	2,423
184	2,669
201	2,916
218	3,163
235	3,409



4.2.1 pav. Teoriškai gautos įtampos priklausomybė nuo impulso laiko

Iš grafiko matyti, kad teoriškai priklausomybė yra tiesinė. Eksperimento metu gauta priklausomybe nėrą visiškai tieses formos. Vadinasi schemos veikimas nėra tikslus 100%.

4.3 Rezultatų palyginimas

Rezultatai lyginami tarp gautų teoriškai ir eksperimentiškai. Įtampų skirtumas apskaičiuojamas iš apskaičiuotos atėmus įtampą, gautą eksperimento metu.

$$\Delta U = U1 - U2 ; \quad (5)$$

U1- apskaičiuota vertė; U2- gauta eksperimento metu.

Toliau norint nustatyti sistemos tikslumą, skaičiuojamas gautos įtampos skirtumo ir teorinės vertės santykis, kuris išreiškiamas procentine verte. Tai apskaičiuojama:

$\Delta U*100\%/U1$;(6)

5.3.1 lentelė. Įtampų palyginimas

analogWrite ()	U1,V apskaičiuota	U2,V eksperimento	ΔU, V skirtumas	%
65	0,943	0,843	0,1	10,6
82	1,189	1,09	0,099	8,3
99	1,436	1,45	0,014	1
116	1,683	1,81	0,127	7,6
133	1,929	2,06	0,131	6,8
150	2,176	2,28	0,104	4,8
167	2,423	2,52	0,097	4
184	2,669	2,8	0,131	4,9
201	2,916	3,05	0,134	4,6
218	3,163	3,2	0.037	1,2

Norint nustatyti vidutinę tikslumo reikšmę, skaičiuojamas aritmetinis vidurkis.

Gauta vidutinę tikslumo reikšmę 5,38 %.

Gautas vidutinis netikslumas keičiant išėjimo įtampą nuo teorinio skaičiavimo skiriasi 5,38%. Eksperimento rezultatus galėjo įtakoti ir tai kad baterijos celės įtampa išmatuota buvo 3,74 V, teoriniams skaičiavimams naudota 3,7V. Taip pat schemos kokybė, kadangi naudojamas tiltelis nėra originalios gamybos, tai pigesnė originalo kopija. O kopijos dažnai būna padaromos iš mažiau kokybiškų dalių.

5 Išvados

- Darbe aptartos dalys,reikalingos kuriant savaeigį modelį.
- Užprogramuotas ryšio modulis, norint susieti valdymo įrenginį su sistema.
- Panaudota speciali programa mobiliajam telefonui, kuri leidžia valdyti sistemą.
- Sukurtas valdymo algoritmų programos kodas, kuris valdo visos sistemos darbą. Kodas apjungia važiavimo krypties ir greičio keitimą.
- Sukurtas variklio greičio valdymo algoritmas, kuris valdomas iš telefono. Tai valdymą daro praktišku, nes leidžia pasirinkti tinkamiausią greitį. Greitis keičiamas žingsniškai. Tamsukurta funkcija, kuri suteikia galimybę iki pasirinktos maksimalios vertės didinti greitį palaipsniui. Tai užtikrina ilgesnį sistemos darbą.
- Visos dalys perkeltos ant nuotoliniu būdu valdomos mašinėlės platformos, panaudojus du jos variklius ir taip realizuojant kuriamą valdymo sistemą savaeigiam modeliui.
- Eksperimento metu buvo siekiama nustatyti kaip tiksliai keičiasi impulso plotis, nuo kurio priklauso vidutinės įtampos vertė, valdantis variklio sukimosi greitį. Atlikus bandymus buvo nustatytas vidutinis 5,38% nuokrypis nuo teorinių skaičiavimų, esant idealiam sistemos valdymui.
- Taip pat skirtingose aplinkose buvo tiriamas sistemos valdymo atstumas. Uždarose patalpose, kuriose valdymo signalas buvo slopinamas mūrinių sienų, vidutinės valdymo ribos buvo 9,27 m. Lauko sąlygomis sistemą buvo galima valdyti 28,92 m atstumu.

6 Literatūros sąrašas

1. http://www.bitstoc.com/index.php?route=product/product&product_id=62(Žiūrėta 2015-04-27)
2. http://www.hobbyking.com/hobbyking/store/__6469__ZIPPY_Flightmax_1300mAh_2S1P_20C.html(Žiūrėta 2015-04-27)
3. http://en.pudn.com/downloads182/doc/detail849113_en.html(Žiūrėta 2015-04-28)
4. http://en.wikipedia.org/wiki/H_bridge(Žiūrėta 2015-04-28)
5. http://pcwortex.byethost9.com/?p=105(Žiūrėta 2015-04-30)
6. Elektrotechnikos pagrindai (Valentinas Zaveckas, 2012m.) [70-86 psl.]
7. Elektros energetikos pagrindai mokomoji knyga (Svinkūnas Gytis, Navickas Algimantas)[]
8. http://www.arduino.cc/#(Žiūrėta 2015-04-30)
9. https://www.youtube.com/watch?v=fCxzA9_kg6s&list=PLA567CE235D39FA84 (Žiūrėta 2015-05-8)
10. http://www.scritub.com/limba/lituaniana/Mikrovaldikli-apibrimas-ir-sav111982314.php (Žiūrėta 2015-05-10)
11. http://www.linotux.ch/arduino/HC-0305_serial_module_AT_commamd_set_201104_revised.pdf(Žiūrėta 2015-05-10)
12.
13. http://abc-rc.pl/templates/images/files/995/1425483439-hc-06-datasheet.pdf(Žiūrėta 2015-05-10)
14. http://anodas.lt/ (Žiūrėta 2015-04-19)
15. http://www.zodziai.lt/reiksme&word=Modelis&wid=12967 (Žiūrėta 2015-05-14)
16. http://www.personalas.ktu.lt/~ignmart/rpt/RPT-LD4.pdf (Žiūrėta 2015-05-05)

Priedai

1 priedas

```
#include <SoftwareSerial.h>
#include <Serial.h>

long lastincrease = 0;
long now = 0;
int spd = 65; // galiniu ratu greitis
int spd1 = 180; // priekiniu ratu pasisukimo greitis
String command = "";
int speedfincreasing = 60;
void setup()
{
  pinMode(5, OUTPUT); // vairas desine
  pinMode(6, OUTPUT); // vairas kaire
  pinMode(9, OUTPUT); // greitis i prieki
  pinMode(10, OUTPUT); // greitis atgal
  pinMode(8, OUTPUT);
  Serial.begin(9600);
}
void kryptis1() // tiesiai
{
  digitalWrite(5, LOW); // nesuka i desine
  digitalWrite(6, LOW); // nesuka i kaire
  analogWrite(9, speedfincreasing); // VAZIUOJA I PRIEKI
  digitalWrite(10, LOW); // nevaziuoja atgal
}
void kryptis2() // tiesiai-kaire
{
  digitalWrite(5, LOW); // nesuka i desine
  analogWrite(6, spd1); // PASUKA I KAIRE
  analogWrite(9, speedfincreasing); // VAZIUOJA I PRIEKI
  digitalWrite(10, LOW); // nevaziuoja atgal
  // delay(500);
}
void kryptis3() // tiesiai-desine
{
  analogWrite(5, spd1); // PASUKA I DESINE
  digitalWrite(6, LOW); // nesuka i kaire
  analogWrite(9, speedfincreasing); // VAZIUOJA I PRIEKI
  digitalWrite(10, LOW); // nevaziuoja atgal
  //delay(500);
}
void kryptis4() // atgal
{
  digitalWrite(5, LOW); // nesuka i desine
```

```
  digitalWrite(6, LOW); // nesuka i kaire
  digitalWrite(9, LOW); // nevaziuoja i prieki
  analogWrite(10, speedfincreasing); // VAZIUOJA ATGAL
}
void kryptis5() // atgal-kaire
{
  digitalWrite(5, LOW); // nesuka i desine
  analogWrite(6, spd1); // PASUKA I KAIRE
  digitalWrite(9, LOW); // nevaziuoja i prieki
  analogWrite(10, speedfincreasing); // VAZIUOJA ATGAL
  // delay(500);
}
void kryptis6() // atgal-desine
{
  analogWrite(5, spd1); // PASUKA I DESINE
  digitalWrite(6, LOW); // nesuka i kaire
  digitalWrite(9, LOW); // nevaziuoja i prieki
  analogWrite(10, speedfincreasing); // VAZIUOJA ATGAL
  //delay(500);
}
void kryptis8() // pasuka tik i desine
{
  analogWrite(5, spd1); // PASUKA I DESINE
  digitalWrite(6, LOW); // nesuka i kaire
}
void kryptis9() // pasuka tik i kaire
{
  analogWrite(6, spd1); // PASUKA I KAIRE
  digitalWrite(5, LOW); // nesuka i desine
}
void kryptis7() // sustoja
{
  digitalWrite(5, LOW); // nesuka i desine
  digitalWrite(6, LOW); // nesuka i kaire
  digitalWrite(9, LOW); // nevaziuoja i prieki
  digitalWrite(10, LOW); // nevaziuoja atgal

}
void loop()
{
  if (Serial.available())
  {
    command += (char)Serial.read();
    //delay(50);
    Serial.println(command);
    if (command == "F") // vaziavimo i prieki siuntimas
    {
      kryptis1();
    }
    if (command == "B") // vaziavimo atgal siuntimas
```

```
{
  digitalWrite(8, HIGH);
  kryptis4();
}
if (command == "S") // stabdymo siuntimas
{
  digitalWrite(8, LOW);
  kryptis7();
  speedfincreasing = 60;
}
if (command == "R") // sukimo i desine siuntimas
{
  speedfincreasing = 60;
  kryptis8();
}
if (command == "L") // sukimo i kaire siuntimas
{
  speedfincreasing = 60;
  kryptis9();
}
if (command == "I") // tiesiai-desine siuntimas
{
  kryptis3();
}
if (command == "G")// tiesiai-kaire siuntimas
{
  kryptis2();
}
if (command == "H")// atgal-kaire siuntimas
{
  kryptis5();
}
if (command == "J") // atgal-desine siuntimas
{
  kryptis6();
}
if (command == "0")
{
  spd = 65;
}
if (command == "1")
{
  spd = 82;
}
if (command == "2")
{
  spd = 99;
}
if (command == "3")
{
```

```
      spd = 116;
    }
    if (command == "4")
    {
      spd = 133;
    }
    if (command == "5")
    {
      spd = 150;
    }
    if (command == "6")
    {
      spd = 167;
    }
    if (command == "7")
    {
      spd = 184;
    }
    if (command == "8")
    {
      spd = 201;
    }
    if (command == "9")
    {
      spd = 218;
    }
    if (command == "q")
    {
      spd = 235;
    }
    if (speedfincreasing < spd)
    {
      speedfincreasing = speedfincreasing + 5;
    }
  }
  delay(50);
  command = "";
}
```