TARTU ÜLIKOOL
Arvutiteaduse Instituut
Informaatika õppekava

Rauno Paluvee

Arduino LCD kasutamine kosmosemängu näitel

Bakalaureusetöö (9 EAP)

Juhendaja: Alo Peets
Kaasjuhendajad: Anne Villems
Taavi Duvin

Tartu 2017

Arduino LCD kasutamine kosmosemängu näitel

Lühikokkuvõte:

Käesoleva bakalaureusetöö eesmärgiks on luua interaktiivne ja lõbus mäng Arduino LCD-le, mis innustaks inimesi rohkem tegelema tehnika ja programmeerimisega. Töö sisu on jaotatud nelja peatükki. Esimeses peatükis tutvustatakse loodavat mängu ja kirjutatakse selle nõuetest. Teises peatükis tuuakse välja kõik lõputööks vajalik riistvara ja tarkvara. Kolmandas peatükis alustatakse mängu loomise protsessiga ja lugejale õpetatakse, kuidas teha sarnast mängu. Neljandas peatükis kirjutatakse ülevaatlikult 3D mudeli loomisest ja 3D printimisest ning näidatakse ka valminud mängu prototüüpi.

Võtmesõnad: Arduino, LCD, programmeerimine

Using Arduino LCD by Creating a Space Game

Abstract:

The aim of this bachelor thesis is to encourage people to feel more interest in technology and programming. The content of this thesis is divided into four chapters. The first chapter introduces the game and brings out its requirements. The second chapter contains all the necessary hardware and software for the game creation. The third chapter starts with the game creation process and teaching the reader how to make a similar game. The fourth chapter writes about 3D modeling and 3D printing. Finally the fourth chapter shows the completed prototype of the game.

Keywords: Arduino, LCD, programming

Sissejuhatus

Elame infoajastul, kus tehnoloogia on pidevalt uuenev ning tehnoloogiaõpet võiks koolides rohkem rakendada. Robootika on väga hea viis, kuidas arendada õpilaste teadmisi selles vallas. Täpsemalt arendab see tudengite STEM oskuseid, milleks on *science* - teadus, *technology* - tehnoloogia, *engineering* - insenerioskused ja *mathemathics* - matemaatika ning ka koostööd. Kõik eelnev mõjutas autorit valima lõputöö valdkonnaks tehnoloogia.

Erinevatest infokanalitest otsides nägi autor, kui vähe on eestikeelseid Arduino ja LCD ekraani õpetusi. Tekkis küsimus, kust leiaksid õpilased Arduino LCD mängu koostamiseks eesti keelset materjali. Sellest ajendatuna sai lõputöö teemaks valitud Arduino LCD mängu loomine.

Antud bakalaureusetöö eesmärgid on:

1. Luua interaktiivne ja lõbus mäng Arduino LCD-le.
2. Innustada inimesi tundma huvi tehnoloogia ja programmeerimise vastu, et arendada õpilaste teadmisi selles vallas.
3. Määrata täpsed nõuded mängu jaoks, et lugejal oleks selge pilt, milline see peab lõpuks olema.
4. Kirjeldada mängu loomise protsessi, et sellest saaks õppematerjal.
5. 3D printeril printida mängu jaoks ise disainitud korpus.
6. Hoida kõik kulud võimalikult madalad, et lugeja saaks ka ise valmistada sarnase mängu.

Lõputöö praktiline osa koosneb riistvara ülesseadmisest ja mängu tarkvara programmeerimisest. Seejärel teeb autor mängu jaoks korpuse disaini ning väljastab selle 3D printeris. Lõputöö tekstiline osa koosneb loodud süsteemi analüüsist, nõuetest (funktsionaalsed ja mittefunktsionaalsed), õpetusest, kasutatud riistvara, tarkvara ja tähtsamate koodiosade ning meetodite kirjeldustest.

Autor kirjutab algselt mängu ideest, semantikast ning nõuetest, mis peavad mängul olema.

Nõuete all mõeldakse mängust, kui tervikust, kus on riistvara ja tarkvara 3D prinditud korpuses üheskoos. Teises peatükis räägib autor lõputöös kasutatud riistvarast ja tarkvarast ning kirjeldab neid detailsemalt. Kolmandas peatükis antakse ülevaade, kuidas ühendada riistvara ja tarkvara ning toob ka mitmeid näiteid enda programmi lähtekoodist. Neljandas peatükis räägib autor mängule korpuse tegemisest ja 3D printimisest üldisemalt.

1. Mängu ülevaade

Mängu loomisel mõeldakse esmalt interaktiivsusele. Autor tahtis, et mängimisel saaks kasutada kahte nuppu, ekraani vajutustundlikkust ja liugurit. Mängija saab ise valida kahe erineva juhtkontrolleri vahel. Juhtkontrolleriteks on nupud ja liugur. Menüü on vajutustundlik ja enne mängu alustamist saab mängija valida, kas ta tahab kasutada liugurit või kahte nuppu. Autor võttis arvesse ka seda, et mäng peab tekitama võistlusmomendi. Selleks peab looma skoori ja tabeli, kust saab vaadata teiste saavutatud skoore. Skoori salvestamiseks on vaja salvestada andmeid mälukaardile ning nende kuvamiseks lugema neid mälukaardilt. Käesolev peatükk räägib täpsemalt mängust ning toob välja ka selle funktsionaalsed ja mittefunktsionaalsed nõuded.

1.1. Semantika

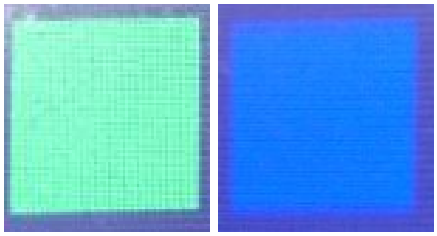
Mängu tegevus toimub aastal 3000 ja keskkonnaks on kosmos, kus mängija kontrollib kosmoselaeva. Kosmoselaeva piloodi ülesandeks on lennata läbi erinevat värvi udukogudest, välja arvatud punastest. Antud laev ei suuda sõita läbi punastest udukogudest ja tulemusena hävib, mis tähendab mängu lõppu. Sõites läbi teistest udukogudest, korjab laev mineraale. Kokku on mängus kolm erinevat värvi udukogu: sinised, rohelised ja punased. Sõites kosmoselaevaga läbi sinise või rohelise udukogu, suureneb mängija mineraalikogus ehk punktide arv, mida hiljem hakatakse kutsuma skooriks.

1.2. Graafika

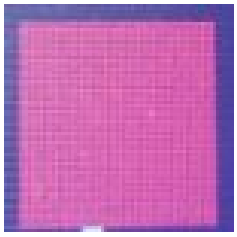
Selles peatükis näidatakse, milline näeb mäng välja ekraanil, et lugeja saaks parema ülevaate mängu nõuetest. Pildid on tehtud ekraanist ja lõigatud välja autori poolt. Graafika tegemisel peetakse silmas mängu kiirust, sest joonistades keerulisemaid objekte, nt kosmoselaev (vt joonis 3), muutub programm aeglasemaks. Eelpool mainitud probleem tekib, kuna iga erineva ala joonistamiseks on vaja erinevat käsku. Mida rohkem käske on vaja teha, seda aeglasem on ka programm. Keerulistel objektidel on rohkem detaile ja seega ka rohkem käske, mida programm

peab täitma. Sellel põhjusel on ainuke detailne objekt mängus kosmoselaev ning udukogud on kõik ruudukujulised.

Kõik udukogud on täpselt 40x40 pikslit suured ja paigutatakse ekraanile suvaliselt. Joonistel 1 ja 2 näidatakse kõiki mängus esinevaid udukogusid.



Joonis 1. Roheline ja sinine udukogu kuvatud ekraanil.

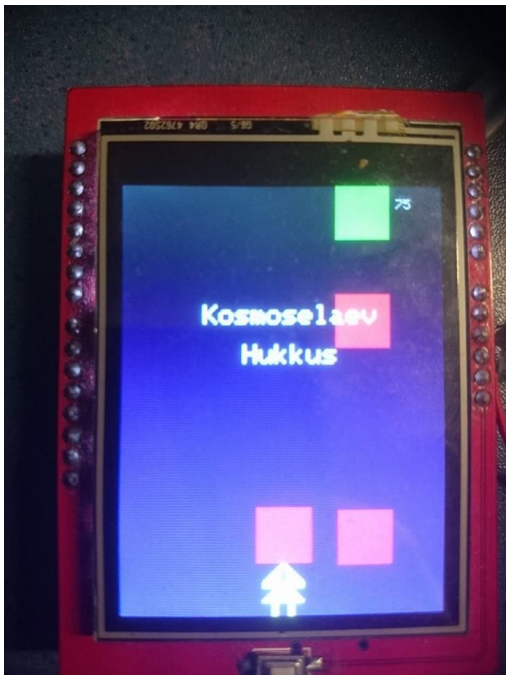


Joonis 2. Punane udukogu kuvatud ekraanil.

Joonis 3. Kosmoselaev kuvatud ekraanil.

Menüüdes navigeerimiseks kasutatakse vajutustundlikkust. Põhjus selleks on kasutaja mugavus. Joonistel 4, 5, 6, 7 ja 8 on toodud välja kõik mängus esinevad menüüd ja ekraanipildid.

Joonis 4. Peamenüü kuvatud ekraanil.



Joonis 5. Ekraan hetkel, kui kosmoselaev on sõitnud vastu punast udukogu.

Joonis 6. Skooritabel kuvatud ekraanil.

Joonis 7. Mäng läbi menüü kuvatud ekraanil.

Joonis 8. Skoori salvestamise menüü kuvatud ekraanil.

Menüüdes kasutatakse kokku kolme erinevat värvi, milleks on punane, roheline ja valge. Kõigi menüüde taust on musta värvi.

1.3. Reeglid

Mängu põhimõte on koguda võimalikult palju mineraale enne kokkupuudet punase udukoguga. Punaste udukoguga kokku puutudes lõppeb mäng ning ekraanile ilmub suurelt valget värvi kiri "Kosmoselaev hukkus" (vt joonist 5). Jätkamiseks on vaja mängijal vajutada ekraani. Kui mängija teeb ekraanivajutuse, kuvatakse menüü, kus on kirjas kogutud mineraalide arv ehk skoor, selle salvestamiseks nupp "SALVESTA SKOOR" ja tagasi menüüsse minemiseks "TAGASI". Seda menüüd on näha ka joonisel 7. "SALVESTA SKOOR" nupule vajutades saab mängija salvestada enda saavutatud skoori, kirjutades nime ja vajutades "SALVESTA". "TAGASI" nuppu vajutades saab mängija minna tagasi alguse ekraanile ning mängida uuesti.

Mängija saab liigutada kosmoselaeva vasakule ja paremale kahe nupu või liuguriga. Hoides all

paremat nuppu, liigub kosmoselaev paremale ning vasaku nupuga liigub vasakule. Liuguriga mängides asetseb kosmoselaev vastavalt liuguri potentsiomeetri asendile. Kosmoselaev ei liigu ekraani piiridest edasi. Kui kumbagi nuppu ei hoita hetkel all, siis kosmoselaev püsib seal, kuhu ta jäeti. Udukogud genereeritakse juhuslikult ning alustades uut mängu, kuvatakse need erinevalt. Skoori kuvatakse mängijale ekraani üleval paremal nurgas. Sinise udukogu skoori väärtus on 50 ja rohelise 25. Neid udukogusid on näha ka joonisel 1.

1.3. Nõuded

Selles peatükis loetletakse kõik funktsionaalsed ja mittefunktsionaalsed nõuded loodava mängu kohta. Mängu all mõeldakse siin mitte ainult tarkvara arendust, vaid tervikut, kus on riistvara ja tarkvara kõik üheskoos 3D printeris väljastatud korpuses.

1.3.1. Funktsionaalsed nõuded

Järgnevalt selgitatakse lõputöö mängu funktsionaalsed nõuded. Funktsionaalne nõue kirjeldab, mida kasutaja saab teha antud mängus. Tabel 1 esitab iga funktsionaalse nõude kohta lühikirjelduse.

Tabel 1. Mängu funktsionaalsed nõuded ja nende kirjeldused.

Nõue	**Kirjeldus**
Kasutaja peab saama skooritabelit vaadata.	Kasutaja saab vaadata skooritabelit, vajutades nupule “SKOORITABEL”. Seda nuppu on näha joonisel 4. Skooritabel on näha joonisel 5. Skooritabeli avamiseks peab kasutaja tegema selle nupu peale puute ning ekraanile ilmuvad maksimaalselt 10 parimat skoori, mis on eelnevalt salvestatud.
Kasutaja peab saama mängu alustada.	Kasutaja saab mängu alustada, puudutades ekraanil nuppu nimega “ALUSTA”. Nupp on näha joonisel 4. Kui kasutaja on vajutanud nuppu “ALUSTA”, siis mäng algab.
Kasutaja peab saama skoori salvestada.	Kasutaja saab pärast mängu lõppu skoori salvestada, vajutades nupule “SALVESTA

	SKOOR". See on näha ka joonisel 6. Kui kasutaja on vajutanud ekraanil antud nuppu, siis avaneb uus ekraan, kuhu saab sisestada 8-tähelise nime ja salvestada selle alla enda teenitud skoori. Skoori salvestamise menüüd on näha joonisel 8. Salvestatud skoori saab hiljem vaadata ka skooritabelist.
Kasutaja peab saama sisestada maksimaalselt 8- ja minimaalselt 1-tähelise kombinatsiooni, mille alla enda skoor salvestada.	Kasutaja saab skoori salvestamise menüüs kirjutada maksimaalselt 8- ja minimaalselt 1-tähelise kombinatsiooni, navigeerides mööda tähestikku nuppudega või liuguriga ja soovitud tähe juures vajutades "SISESTA". Nii saab kasutaja teha maksimaalselt 8 korda. Kasutaja saab salvestada sisestatud nime alla enda skoori, vajutades nupule "SALVESTA".
Kasutaja peab saama kosmoselaeva liigutada nuppudega.	Kasutaja saab kosmoselaeva liigutada kahe nupuga. Hoides all vasakut nuppu, liigub kosmoselaev ekraanil nii palju vasakule kui võimalik ning paremaga liigub paremale nii palju kui võimalik.
Kasutaja peab saama kosmoselaeva liigutada liuguriga.	Kasutaja saab kosmoselaeva liigutada liuguriga. Kosmoselaev asetseb vastavalt potentsiomeetri asendile.
Kasutaja peab saama valida liuguri ja nuppude vahel.	Vajutades lülitit, saab kasutaja valida kahe juhtkontrolleri vahel,
Kasutaja peab saama minna pärast mängu lõppu algmenüüsse tagasi.	Kui kasutaja on mängu kaotanud ja ekraanipuute teinud, siis on menüüs kaks valikut: "SALVESTA SKOOR" ja "TAGASI" (vt joonis 7). Vajutades "TAGASI", kuvatakse jälle algmenüü.
Kasutaja peab saama lülitada Arduino/Genuino Uno plaadi sisse.	Kui kasutaja vajutab toitelülitit korpuse paremas nurgas, siis lülitub Arduino/Genuino Uno plaat sisse ning kuvatakse peamenüü, mida on näha joonisel 4.

1.3.2. Mittefunktsionaalsed nõuded

Siin peatükis tuuakse välja kõik mängu mittefunktsionaalsed nõuded. Need nõuded määravad, milline peab mäng välja nägema. Tabel 2 esitab iga mittefunktsionaalse nõude kohta ka tema

lühikirjelduse.

Tabel 2. Mängu mittefunktsionaalsed nõuded ja nende kirjeldused.

Nõue	**Kirjeldus**
Skoori salvestamise lehel peab olema algselt nupp "SALVESTA" punast värvi, kuid sisestades ühe tähe, muutub nupp roheliseks.	Algselt on nupp "SALVESTA" punane, kuna kasutaja ei saa salvestada tühja nime alla enda skoori. Seda on näha ka joonisel 8. Kasutaja peab vähemalt ühe tähe sisestama, enne kui ta seda salvestada saab ning seda tehes muutub nupp "SALVESTA" roheliseks.
Mängu kaotades peab ilmuma ekraanile valget värvi kiri "Kosmoselaev hukkus" ja taustaks jääb mängu kaotusmoment.	Põrgates kokku punase udukoguga jääb mäng seisma ja ekraanile ilmub valget värvi tekst "Kosmoselaev hukkus". Seda sündmust on näha joonisel 5.
Skoori salvestamise lehel peab olema tähestikus hetkel valitud täht punane. Kõik teised tähed peavad olema valged.	Kasutaja saab endale valida sobiva tähe juhtkontrollerit liigutades või vajutades. Valitud tähe värv muutub punaseks. Eelnevalt valitud täht muutub valgeks. Ainult üks täht saab korraga olla punast värvi.
Mängu mängides peab olema üleval paremal nurgas valge värviga kuvatud mängija hetkeskoor.	Kasutaja saab mängides näha enda mineraalide arvu ehk skoori üleval paremal nurgas.
Kui lülitiga on valitud liugur juhtkontrolleriks, peab põlema sinine valgusdiood.	Kasutaja saab lülitiga valida kahe juhtkontrolleri vahel. Kasutaja teab, et kui põleb sinine valgusdiood, siis on hetkel vooluringis liugur.
Kui lülitiga on valitud nupud juhtkontrolleriks, siis peab põlema punane valgusdiood.	Kasutaja saab lülitiga valida kahe juhtkontrolleri vahel. Kasutaja teab, et kui põleb punane valgusdiood, siis on hetkel vooluringis nupud.
Kui toide on Arduino plaadil sees, siis peab põlema roheline valgusdiood.	Kasutaja saab lülitada mängu sisse, vajutades korpuse paremas pooles asuvat lülitit. Kasutaja teab, et kui põleb roheline valgusdiood, siis on Arduino plaat ühendatud toiteallikaga ning mäng läheb käima.
Arduino Uno plaadi sisselülitamisel peab	Kui kasutaja lülitab voolu sisse Arduino Uno

ilmuma LCD ekraanile taustaks pilt, mis loetakse SD-kaardilt.	plaadile, peab LCD ekraan käivituma ning taustaks ilmuma pilt. See pilt laetakse SD-kaardilt maksimaalselt kahe sekundiga.

Nüüdseks on lugejale tutvustatud antud lõputöö eesmärke ja analüüsiti tarkvara ja riistvara nõudeid loodava mängu jaoks. Tehtavas analüüsis toodi välja mängu funktsionaalsed ja mittefunktsionaalsed nõuded ning näidati ka graafikat. Enne mängu loomise protsessi juurde liikumist tuuakse välja kõik lõputööks vajalik riistvara ja tarkvara.

2. Kasutatud riistvara ja tarkvara

Antud peatükis tutvustatakse lõputöös kasutatud riist- ja tarkvara. Iga komponendi korral seletatakse selle valiku põhjusi ning selle rolli käesoleva lõputöö loomisel. Riistvara valimisel on autor proovinud hoida kulud võimalikult madalad, et pidada kinni lõputöö kuuendast eesmärgist. Kogu riist- ja tarkvara on Eestis kättesaadav. Tabel 3 võrdleb iga riistvara komponendi hinda Eesti ja Hiina vahel.

Tabel 3. Lõputöös kasutatava riistvara hinnad Eestis ja Hiinas.

Riistvara	**Hind Eestis**	**Hind Hiinas**
Arduino/Genuino Uno	18,00 EUR	14,87 EUR
2,4 tolline vajutustundlik TFT LCD ekraan	18,00 EUR	4,49 EUR
2 nuppu	2,40 EUR	3,22 EUR
Liugur	13,90 EUR	2,00 EUR
2 lülitit	2,00 EUR	1,07 EUR
Micro-SD kaart	5,99 EUR	1,56 EUR
Micro-USB juhe	2,40 EUR	0,72 EUR
Ühendusjuhtmed	6,00 EUR	2,17 EUR

Maketeerimislaud	5,00 EUR	0,67 EUR
Takistid	0,30 EUR	2,31 EUR
Patareipesa	3,50 EUR	2,25 EUR
3 valgusdioodi	1,80 EUR	1,46 EUR

Tabelil 1 olevad hinnad on viimati vaadatud 09.05.2017. Eestist on kõik riistvara küll kättesaadav, kuid kokku läheb kallimaks, kui tellida Hiinast. Eestist tellides kõik komponendid läheks kokku 79,29 EUR ja Hiinast 36,79 EUR. See näitab, et lugejal on võimalik selline mäng valmistada ka iseseisvalt kodus 40 euro piires.

2.1. Arduino/Genuino Uno

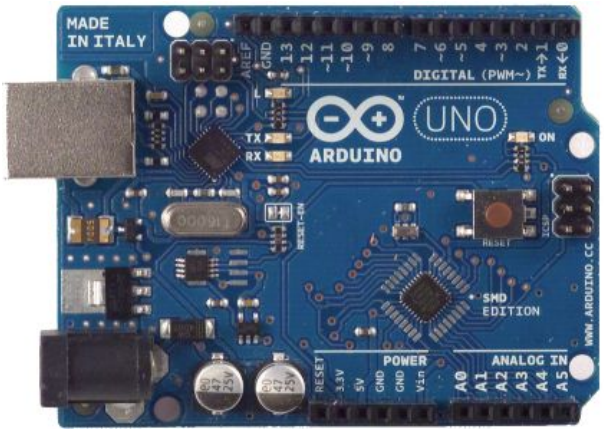
Antud lõputöös on tegu Genuino Uno plaadiga, mis on ostetud väljaspool Ameerika Ühendriike. Kõik plaadid, mis on Ameerika Ühendriikides, kannavad nime Arduino, ja kõik, mis on väljaspool seda, kannavad nime Genuino. See on tingitud Arduino ettevõttes esinevate konfliktide tõttu. Käesolevas lõputöös kirjutatakse ikkagi Arduino, sest valdav enamik kasutab pigem seda sõna. Edaspidi Arduinost rääkides mõeldakse tegelikult täpsemalt Genuinost.

Antud mängu tegemiseks kasutatakse Arduino/Genuino Uno plaati. Arduinol on valikus veel teisigi erinevaid plaate, millega saaks antud mängu realiseerida, näiteks Arduino/Genuino Leonardo ja Arduino/Genuino Mega. Külastades erinevaid Arduino foorumeid ja konsulteerides juhendajaga, leidis autor erinevaid põhjuseid, miks Arduino/Genuino Uno on alustajatele parim valik. Järgmisena tuuakse välja tehtud valiku põhjused:

1. Antud plaat on väga kergesti kättesaadav, ka Eestis.
2. Tegemist on suhteliselt odava riistvaraga, võrreldes teiste eelnevalt mainitud Arduino/Genuino plaatidega. Valitud plaat maksab Eestist ostes ligikaudu 18 eurot, kuid Arduino/Genuino Mega maksab umbes 45 eurot ning Arduino/Genuino Leonardo maksab 20 eurot [1].
3. Arduino/Genuini Uno jaoks leidub väga palju näidiskoode, projekte ja õpetusi, kuna seda

on kasutatud ja dokumenteeritud kõige rohkem tervest Arduino ja Genuino perest.

Arduino Uno on mikrokontrolleriga platvorm ehk riistvara koos operatsioonisüsteemi ja kompilaatoriga, millele rajatakse tegelik süsteem. Platvorm on natuke üldisem mõiste ning antud töös kasutatud Arduino Uno platvormi võib nimetada ka lihtsalt plaadiks, mida on näha joonisel 9. Antud plaat kasutab ATmega328 mikrokontrollerit, mis on sisuliselt terve plaadi süda. Mikrokontroller on kiip ehk väike elektroonikakomponent, milles on peale keskprotsessori veel mälu, mis salvestab koodi ning muud vajalikud andmed mikrokontrollerile [2]. Keskprotsessor on peamine vahend programmeeritud käskude täitmisel. [3]



Joonis 9. Arduino/Genuino Uno [4].

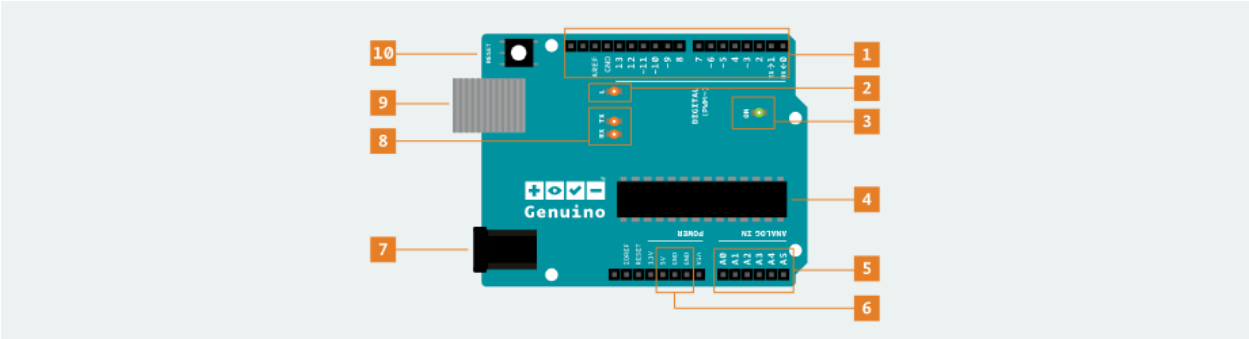
Enamik mikrokontrollerite arenduskeskkondi töötab ainult Windows operatsioonisüsteemiga. Arduino tarkvara töötab aga Mac, Windows ja Linuxi peal. Antud töös kasutatakse Windowsi, kuna see on kaasaegsete tarkvaraarenduste puhul kõige populaarsem operatsioonisüsteem [5].

Arduino plaadid on võimelised lugema sisendit, seda töötlema ja muutma selle väljundiks. Käesolevas lõputöös kasutatakse sisendina nuppudele ja ekraanile vajutamist ning väljund on vastavalt ekraanil kosmoselaeva liikumine või menüüs navigeerimine.

2.1.1. Arduino Uno plaadi anatoomia

Arduino Uno plaat on käesoleva lõputöö põhiline osa, kuna kõik käsud käivitatakse primaarselt

plaadil oleval mikrokontrolleril. Joonisel 10 on näidatud, mis elektroonikakomponendid on täpsemalt Arduino Uno plaadi peal.



Joonis 10. Arduino Uno plaadi anatoomia [6].

Joonisel 10 numbrite tähendused:

1. Digitaalsed viigud. Neid on plaadil kokku 14.
2. Viik 13 valgusdiood. See on ainuke sisseehitatud mehhanism, mis lülitab valgusdioodi sisse ja välja. Valgusdiood on väga kasulik esmaseks testimiseks.
3. Toite valgusdiood, mis näitab, kas plaat on ühendatud toiteallikaga.
4. ATmega mikrokontroller.
5. Analoogviigud. Neid kasutatakse antud lõputöös juhtkontrollerite ühendamiseks.
6. GND (maandusviik) ja 5V viik. Neid tuleb kasutada, et anda vooluringile +5V toidet ja maandust.
7. Toite ühendaja. Selle abil saab plaat toidet, kui ta ei ole USB'ga ühenduses. Aktsepteerib pinget vahemikus 7-12V.
8. TX ja RX LED'id. Need valgusdioodid vilguvad vastavalt plaadi ja arvuti vahelisele kommunikatsioonile.
9. USB pistikupesa. Kasutatakse plaadi toiteallikana ja visandite laadimiseks plaadile.
10. Lähtestusnupp, millele vajutamine põhjustab ATmega mikrokontrolleri taaskäivituse.

Arduino tarkvara käivitamist saab mõjutada, saates juhiseid Arduino plaadil olevale mikrokontrollerile, mille asukoht on tähistatud joonisel 10 numbriga 4. Tarkvara arendamiseks

on vaja kasutada Arduino programmeerimiskeskkonda, millest räägitakse täpsemalt järgmises peatükis. Selles keskkonnas toimub kõik koodi kirjutamine.

2.2. Arduino programmeerimiskeskkond

Avatud lähtekoodiga Arduino programmeerimiskeskkond ehk IDE nimega Arduino 1.8.1. teeb lihtsamaks koodi kirjutamise ja selle üleslaadimise Arduino plaadile. See keskkond on kirjutatud Javas ja põhineb *Processing* programmeerimiskeelel. *Processing* on avatud lähtekoodiga programmeerimiskeel, mis on loodud eesmärgiga õpetada programmeerimise baasteadmisi. Selle keele süntaks on väga sarnane C/C++ programmeerimiskeelele. [7]

"*Hello World"* programm näeb välja *Processing* keeles järgmine:

```
void setup() {
       println("Hello world.");
}
```

Arduino programmeerimiskeskkonnaga tuleb kaasa ka C/C++ teek *Wiring*, mis teeb sisend-väljundoperatsioonid lihtsamaks. *Wiring* põhineb eelnevalt mainitud *Processing* keelel ja lubab tarkvaral kontrollida plaadil olevat mikrokontrollerit ning luua igasuguseid interaktiivseid objekte. [8] Antud lõputöös peame kontrollima kahte nuppu, liugurit ja ekraani. Seda riistvara saame kontrollida Arduino keskkonnas loodud programmi käivitamisel. Seda programmi nimetatakse visandiks, mis salvestatakse arvutisse tekstifailina ja laiendusega .ino.

Arduino programmeerimiskeskkond toetab C ja C++ keeli, kasutades erilisi reegleid koodi struktureerimisel, näiteks loogelise sulu kirjutamisel lisab programm automaatselt pärast ENTER klahvi vajutamist lõpetava loogelise sulu ära. Antud keskkond sisaldab ka koodi redaktorit, mis lubab teksti lõigata, kleepida, otsida ja asendada. [9]

Minimaalne Arduino C/C++ programm vajab ainult kahte põhifunktsiooni. Esimene on programmi alustamiseks ja teine on programmi põhitsükkel. Need funktsioonid on juba

kompileeritud ja ühendatud programmilõiguga *main()*. [10] Need põhifunktsioonid on järgmised:

- *Setup()*: See funktsioon kutsutakse välja, kui visand algab pärast Arduino sisselülitamist või taaskäivitamist. Seda kasutatakse, et initsialiseerida vajalikud muutujad ja teegid. [11]
- *Loop()*: Pärast *setup* kutset läheb käiku funktsioon *loop*. Seda korratakse nii kaua põhiprogrammis, kuni plaat on välja lülitatud või taaskäivitatud. [12]

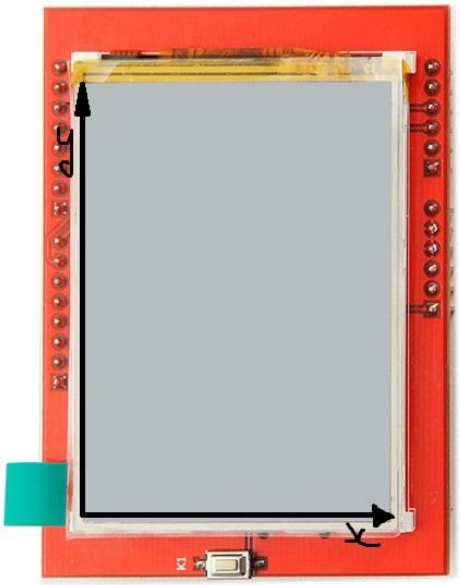
Nüüd on kirjutatud Arduino Uno plaadist ja millega arendada selle peale tarkvara. Edasi kirjutatakse täpsemalt Ekraanist, mida kasutatakse käesolevas lõputöös.

2.3. Ekraan

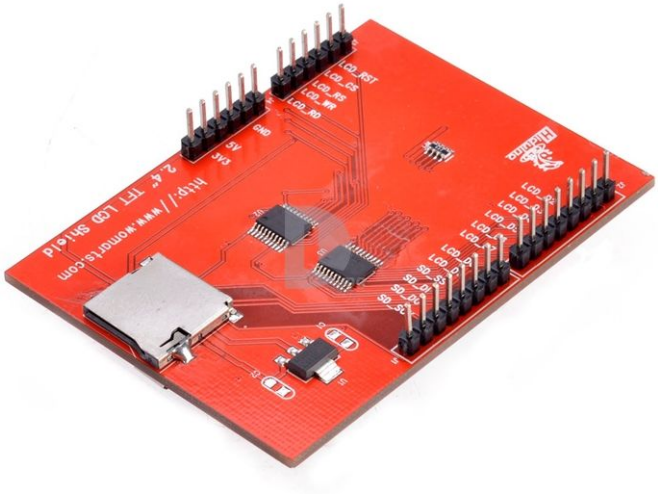
Lõputöö eesmärgiks on luua interaktiivne mäng Arduino LCD peale, mis tähendab, et lõputööks on vaja ka LCD ekraani. Mängu ülevaates seadsime ka tingimusteks, et mäng peab olema menüüs navigeerimiseks vajutustundlikku ekraaniga ja suuteline salvestama skoori. Kõik tingimused ekraani jaoks on järgmised:

- Ekraan peab olema puutetundlik, et kasutaja saaks menüüs navigeerida.
- Ekraan peab suutma värve kuvada, kuna tegemist on värvilise mänguga.
- Ekraanil peab olema microSD kaardi ühendus, et oleks võimalik salvestada ja lugeda skoori.
- Ekraan peab olema vähemalt 2 tolli suur.

Autor otsustas 2.4 tollise puutetundliku TFT LCD-ekraani kasuks. Tegemist on Arduino laiendusplaadiga, mis maksab Hiinas 7 EUR. Laiendusplaadi kasutamine on eelistatud, sest muidu peaks ekraani ise kokku jootma. Antud ekraan sobib ka eelnevalt mainitud mängu tingimustega. Seda ekraani on näha joonisel 11, kus on autori poolt peale joonistatud ka ekraani x ja y teljed, et edaspidi koordinaatidega tegelemisel oleks lugejal lihtsam aru saada.



Joonis 11. TFT LCD-kuvar (2,4 tolline) koos autori poolt peale joonistatud x ja y teljega [13].



Joonis 12. TFT LCD-kuvar tagantvaates [14].

TFT ehk *Think Film Transistor* on tehnoloogiatüüp, mida kasutatakse LCD pildi kvaliteedi suurendamiseks. Iga TFT-LCD piksel omab klaasi peal enda transistorit, mis pakub rohkem kontrolli piltide ja värvide kohta. [15]

Antud LCD'l on väga palju positiivseid külgi. Esiteks on sellel ekraanil väga hea värvikontrast, kuna tegemist on TFT tehnoloogia tüübiga. Ekraan on mängu jaoks piisava suurusega. Antud ekraan on suuteline kuvama 262 000 erinevat värvitooni. Piksleid on 240x320 ja igal pikslil on individuaalne pikslikontroll. Sellel ekraanil on palju suurem resolutsioon kui must-valgel 128x64 ekraanil. Olemas on ka takistuskihiga vajutustundlik ekraan, mis avastab sõrme vajutuse ükskõik missuguses ekraani asukohas. Kuvari sisse on ehitatud microSD kaardi ühendus, mida on näha jooniselt 12. Tänu sellele saab implenteerida piltide kuvamise, skoori salvestamise ja andmete lugemise. Kuvar kasutab digitaalseid viikusid D5-D13 ja analooge A0-A4. See tähendab, et kasutada saab digitaalseid viikusid D2 ja D3 ning analoogset viiku A5. Viik D12 on vaba, kui ei kasutata microSD mälukaarti, kuid antud töös kasutatakse seda. Kuvar töötab iga Arduino 328 või Megaga, välja arvatud Leonardo, mida pole veel toetatud). [14]

2.4. Juhtkontrollerid

Mängu reeglites kirjutati, et kasutaja saab liigutada kosmoselaeva kahe nupu või siis liuguriga. Neid seadmeid nimetatakse juhtkontrolleriteks. Juhtkontroller või siis lihtsalt kontroller on seade, millega juhitakse tegelast või mingit objekti mängus [16]. Antud lõputöös kasutatakse juhtkontrollerit, et liigutada kosmoselaeva, mida on näha joonisel 3, ja skoori salvestamiseks. Edasi kirjutatakse mõlemast juhtkontrollerist täpsemalt.

2.4.1. Nupud

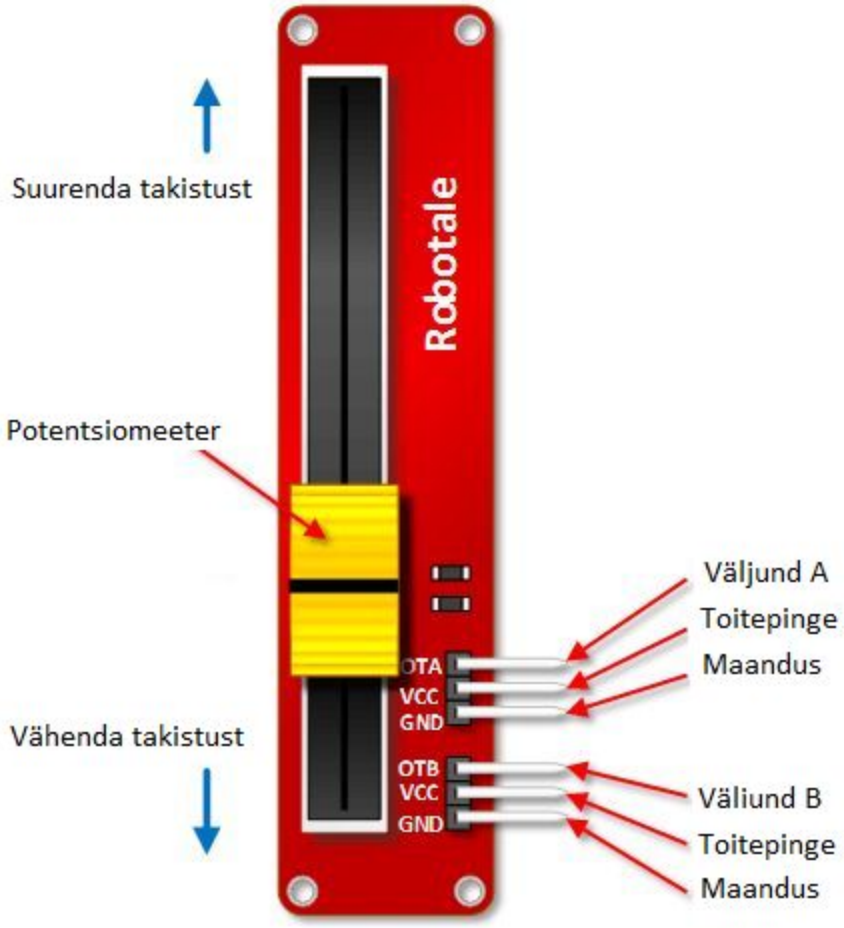
Nupp on komponent, mida vajutades ühendatakse kaks punkti vooluringis [17]. Käesolevas lõputöös kasutatakse kahte Arduinoga ühilduvat nuppu (vt joonist 13), mis kontrollivad kosmoselaeva liikumist ja skoori salvestamist.

Joonis 13. Arduino nupp.

Vasakule ja paremale liikumiseks on vaja eristada mõlemad nupud teineteisest. Selleks loetakse hiljem nupu poolt analoogviigule tulevat pinget. Käesolevas lõputöös kasutame 10K Ohm, 1K Ohm, 220K Ohm ja 330K Ohm takisteid. Need takistid on näidatud ka vastavalt joonisel 15. Kosmoselaev lendab paremale, kui hoitakse peal paremat nuppu, ja vasakule, kui hoitakse peal vasakut nuppu. Selleks, et teada, kumma nupuga on tegu, kontrollitakse pinget sisendis, kuhu nupp on ühendatud. Pinged peavad erinema, sest mõlema nupu korral kasutatakse sama sisendit, milleks on analoogviik A5. Sama sisendit kasutatakse, sest see on ainuke vaba sisend antud Arduino plaadil. Pinge muutmiseks antud sisendis peab muutma nuppude voolutugevust, lisades vooluskeemi takisteid.

2.4.2. Liugur

Liugur on kontroller, mida kasutatakse samuti selles lõputöös kosmoselaeva liigutamiseks ja ka skoori salvestamisel nime kirjutamiseks. Lõputöös kasutatav liugurit on näha joonisel 14.



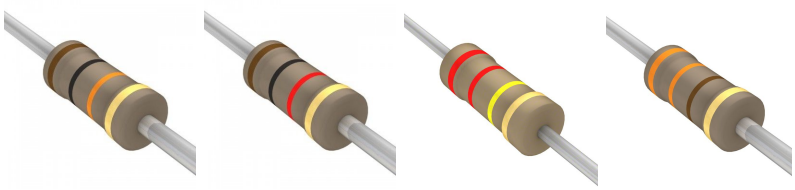
Joonis 14. Lõputöös kasutatava liuguri komponentide selgitus .

Liuguri liigutamiseks on liuguril peal potentsiomeeter, mida on näha ka joonisel 14. Potentsiomeeter on muuttakisti, mida liigutades muutub liuguri takistus. Takistust muutes muutub ka voolutugevus vastavale analoogviigule, kuhu liugur on ühendatud. Hiljem pinget lugedes saab kosmoselaeva liigutada sujuvalt ühest äärest teise.

2.5. Takistid

Käesolevas lõputöös kasutatakse mitmeid erinevaid takisteid. Takisti on elektroonikakomponent, mida kasutatakse mingi soovitava või kindla elektritakistuse tekitamiseks vooluringis. Takistitega saab voolutugevust piirata või siis tekitada pingelangust. [18] Joonisel 15 näidatakse

kõik kasutusele võetavad neli erinevat takistit, sest igaühel on erinev värvikood, millega neid saab eristada.

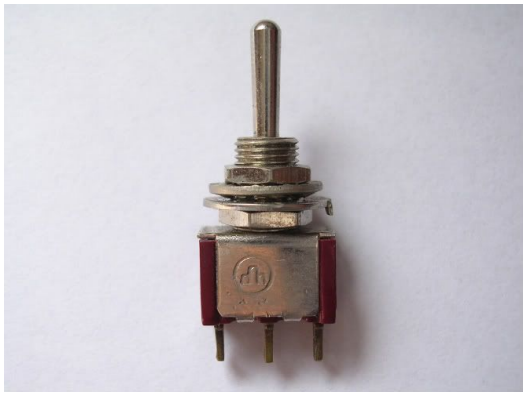


Joonis 15. a) 10K Ohm takisti b) 1K Ohm takisti c) 220K Ohm takisti d) 330K Ohm takisti.

Erinevaid takisteid kasutatakse, et saada täpselt soovitud voolutugevus mõlemale nupule vajutades. Nuppudel on vaja tekitada erinev voolutugevus, et hiljem eristada, kumba nuppu vajutatakse.

2.6. Lüliti

Käesolevas lõputöös tuleb kasutada lülitit, kuna see LCD kuvar hõivab 5 analoogviiku Arduino Uno plaadilt. See tähendab, et vabaks jääb vaid analoogviik A5. Selleks, et kasutada mõlemat kontrollerit sama plaadiga, kasutab autor lülitit, mis lülitaks ühel juhtkontrolleril voolu välja, et see ei segaks teist kontrollerit. Juhtkontrolleri vooluringluse katkestamiseks kasutatakse dioodi, mis laseb voolul liikuda vaid ühes suunas. Kasutatav lüliti on näha ka joonisel 16.



Joonis 16. Lüliti.

Lüliti erineb nupust, sest nupu toimimiseks on vaja seda peal hoida, kuid lülitit on vaja vajutada üks kord. Lülitit kasutatakse veel selles lõputöös Arduino plaadi ühendamisel toiteallikaga. Seda on vaja, et mängu saaks mängida ka USB juhtmega ühendamata.

2.7. Micro SD-kaart

Mängija skoori salvestamiseks on vaja kasutada mälukaarti. Selleks kasutatakse käesolevas lõputöös micro SD-kaarti, mida on näha ka joonisel 17. Sellele kaardile on vaja mahutada .txt formaadis fail, kuhu salvestatakse 10 parimat skoori. Tekstifail koosneb maksimaalselt kümnest reast, sest ekraanile kuvatakse maksimaalselt 10 skoori. Seega peab selle faili jaoks arvestama umbes 150 baiti ruumiga ehk mälukaardis peab olema vähemalt 150 baiti vaba ruumi.

Joonis 17. Micro SD-kaart.

SD on välkmäluga mälukaardi formaat, mis on mõeldud infotehnoloogilistes kandeseadmetes kasutamiseks. Välkmälu on säilmälu, mida saab elektriliselt kustutada ja programmeerida. Säilmälu tähendab seda, et andmed säilivad ka pärast toite väljalülitamist. Sellepärast sobibki see mälu käesoleva lõputöö jaoks väga hästi, kuna skoor peab olema alati salvestatud. [19]

2.8. Micro-USB juhe

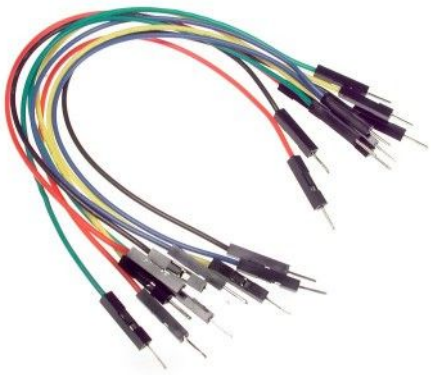
Arduino/Genuino Uno plaat saab toidet USB ühendusega või mingi välise toiteallika abil. Antud lõputöös kasutame ühendamiseks micro-USB juhet, mis on näha joonisel 18.

Joonis 18. Micro-USB juhe.

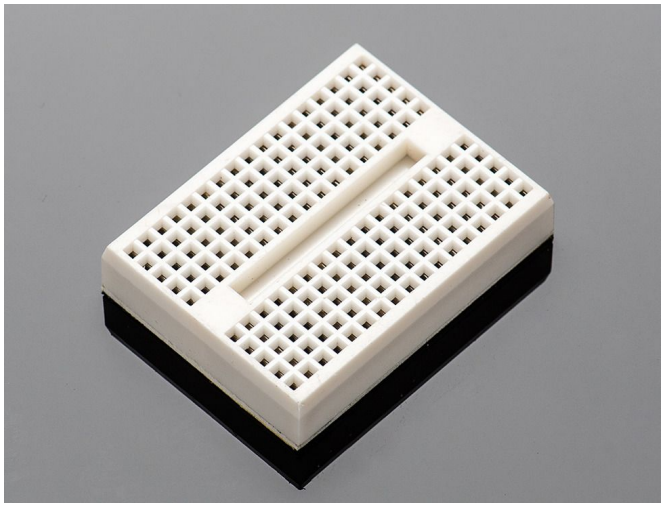
Selle juhtme abil saab ühendada Arduino plaadi arvutiga ning alustada programmeerimist.

2.9. Ühendusjuhtmed ja maketeerimislaud

Nuppude ühendamiseks Arduinoga kasutatakse algselt projektis ühendusjuhtmeid, mis on näha joonisel 19.



Joonis 19. Ühendusjuhtmed.

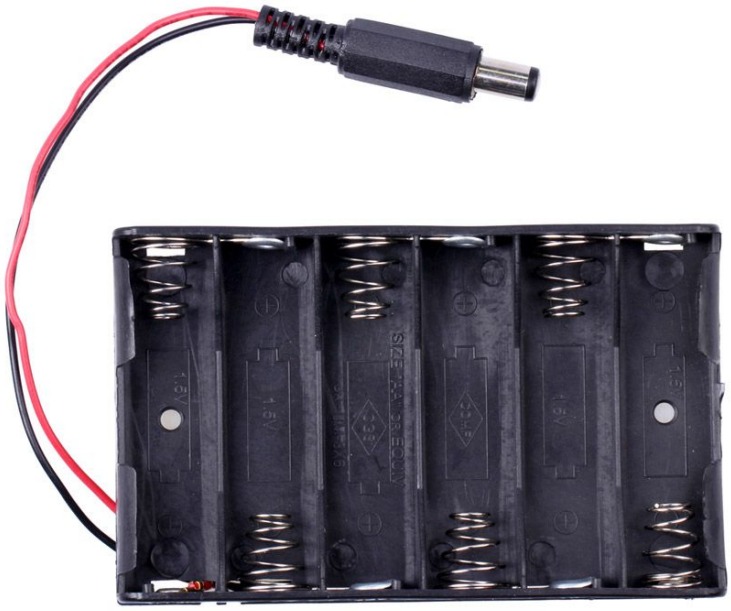


Joonis 20. Maketeermislaud.

Nuppude algseks ühendamiseks saab kasutatada maketeerimislauda. Antud töös kasutatakse väikest maketeerimislauda mõõtmetega 46mm x 36mm, mis on näidatud joonisel 20. Otsustati kasutada väikest maketeerimislauda, kuna pole liigselt palju ühendusi vaja teha. Hiljem, kui kõik riistvara on korrektselt ühendatud ja mäng töötab, saab alustada jootmisega. Siis võib eemaldada maketeerimislaua. Korpuse sisse maketeerimislaud ei mahu.

2.10. Patareipesa

Arduino Uno käivitamiseks on vaja anda plaadile 7-12V voolu [3]. Selleks kasutame patareipesa, kuhu mahub 6 AA patareid kõrvuti. Seda patareipesa on näha joonisel 21. See annab kokku 9V voolu, millest piisab umbes tunniks ajaks. Patareipesa tuleb ühendada juhtmega Arduino plaadi toitepesasse. Arduino plaadi toiteühendaja asukoht on joonisel 10 märgistatud numbriga 7.



Joonis 21. Patareipesa.

Lisaks patareipesale on vaja ka 6 AA patareid. Alternatiivina võib kasutada ka 5V akupanka ja ühendada see Arduino Uno plaadi USB pistikusse.

2.11. Valgusdiood

Valgusdiood on pn-siirdega diood, mis muundab elektrienergiat nähtavaks valguseks. Valfusdioodi kohta öeldakse ka lühendina LED (inglise keelest Light-Emitting Diode – valgust kiirgav diood). [20] Eesti keeles öeldakse ka lühendatuna leed. Käesolevas lõputöös kasutatavat valgusdioodi on näha joonisel 22.

Joonis 22. Valgusdiood

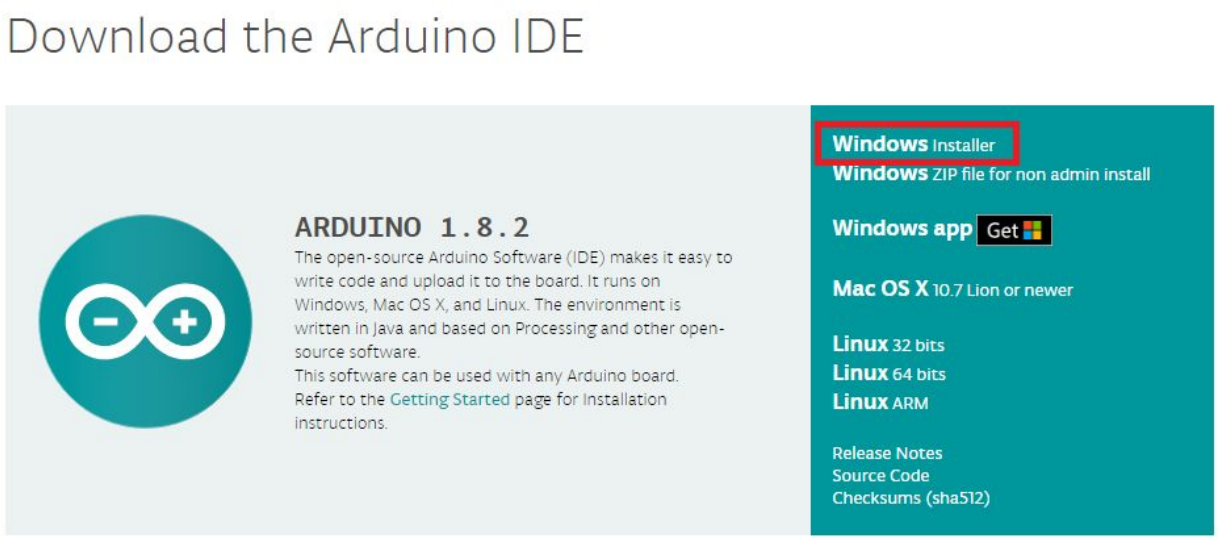
Mängu jaoks kasutatakse kolme leedi. Kaks neist näitavad kumb juhtkontroller on hetkel sisse lülitatud ja kolmas näitab, kas Arduino Uno plaat on ühendatud toiteallikaga. Nüüd on kõik vajalik riistvara ja tarkvara loetletud ning järgmisena kirjutatakse mängu loomise protsessist.

3. Mängu loomise protsess

Antud peatükis kirjutatakse, kuidas seada üles Arduino tarkvara ning ühendada eelnevas peatükis mainitud riistvara. Alustatakse Arduino programmeerimiskeskkonna ülesseadmisega, mida nimetatakse ka lihtsamalt Arduino IDE'ks.

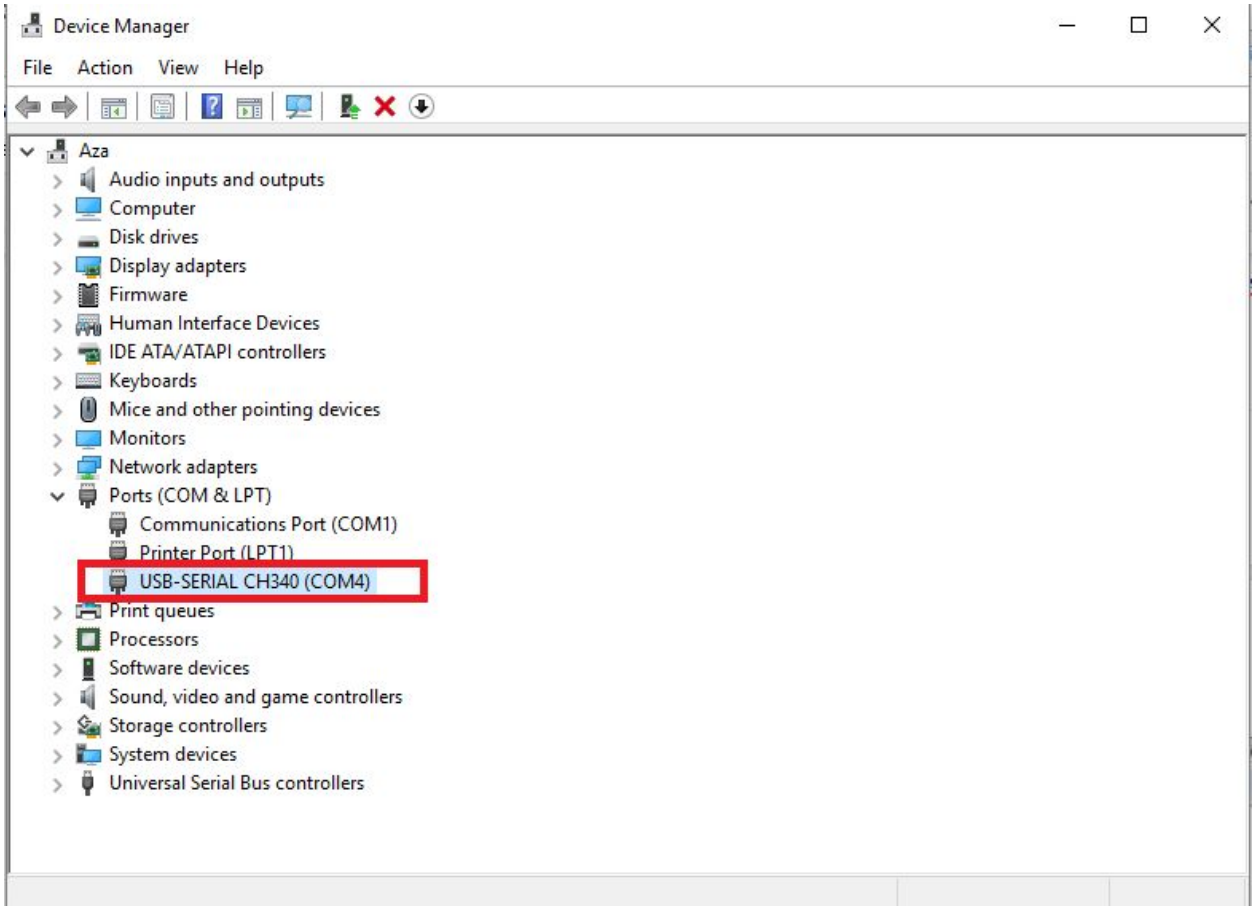
3.1. Arduino ühendamine arvutiga ja arenduskeskkond

Esimese asjana, et üldse Arduinoga saaks töötada, peab laadima alla Arduino IDE tarkvara. See on Arduino programmeerimiskeskkond, millest kirjutati ka täpsemalt eelmises peatükis. Allalaadimiseks on vaja minna aadressile http://www.arduino.cc/en/Main/Software. Sellel lehel on vaja navigeerida *Windows installer* juurde, mis on näidatud joonisel 23, ja seda vajutada.



Joonis 23. Abistav joonis Arduino IDE allalaadimiseks [7].

Kui Arduino tarkvara on edukalt arvutisse paigaldatud, siis ühendada Arduino plaat arvutiga USB juhtme abil. Meil on vaja teada, mis COM pordiga on plaat ühendatud ning edaspidi ühendada sama pordiga. Seadmehaldurist saab näha, millise COM pordiga on Arduino ühendatud (*Juhtpaneel -> Riistvara ja heli -> Seadmehaldur*). Joonisel 24 on näidatud, kus asub seadmehalduris otsitav koht.



Joonis 24. Seadmehaldur.

Kui on teada, mis COM pordiga on Arduino ühendatud, siis tuleks vaadata, kas Arduino keskkonnas on see ka korrektselt ära määratud. Kui ei ole, siis tuleb see ise ära teha. Samuti tuleb vaadata, kas on määratud õige Arduino plaat, milleks on Arduino/Genuino Uno. Mõlemat saab teha Arduino IDE alammenüüst *Tööriistad*.

Edasi tuleks kontrollida, kas kõik on õigesti ühendatud ja seadistatud. Selleks kasutame Arduino IDE mõnda näidisvisandit, navigeerides programmeerimiskeskkonnas järgmiselt: *Fail -> Näited -> 01.Basics -> Blink.* Sellele vajutades avaneb uus aken koos näidiskoodiga. Nüüd tuleb jooksutada seda programmi, vajutades *Laadi üles* nupule, mis on punaselt näidatud joonisel 25.

Joonis 25. Arduino IDE menüüriba.

Kui kood on ühendatud ja Arduino plaadile üles laetud, hakkab leed ehk valgusdiood vilkuma. See tähendab, et Arduino on korrektselt arvutiga ühendatud ja programmeerimiskeskkond on korrektselt seadistatud. Valgusdioodi asukoht Arduino/Genuino Uno plaadil on märgitud joonisel 9 numbriga 2.

Nüüd saab lugeja avada ka mängu lähtekoodi, mis on Bitbucket repositooriumist allalaaditav järgneval aadressil: https://bitbucket.org/RaunoP/arduino-lcd-kosmose-ralli/downloads/. Koodi allalaadimiseks on vaja vajutada "Download repository", mille tulemusena tekib arvutisse .zip fail, mis on vaja pakkida lahti. Lahti pakitud kaustas on kaust nimega "Arduino LCD mäng", kus sees paikneb .ino laiendusega fail "Kosmose Ralli lähtekood". Edasi hakatakse tegelema LCD ekraaniga ning selle ühendamisega Arduino Uno plaadiga.

3.2. LCD Ekraani seadistamine

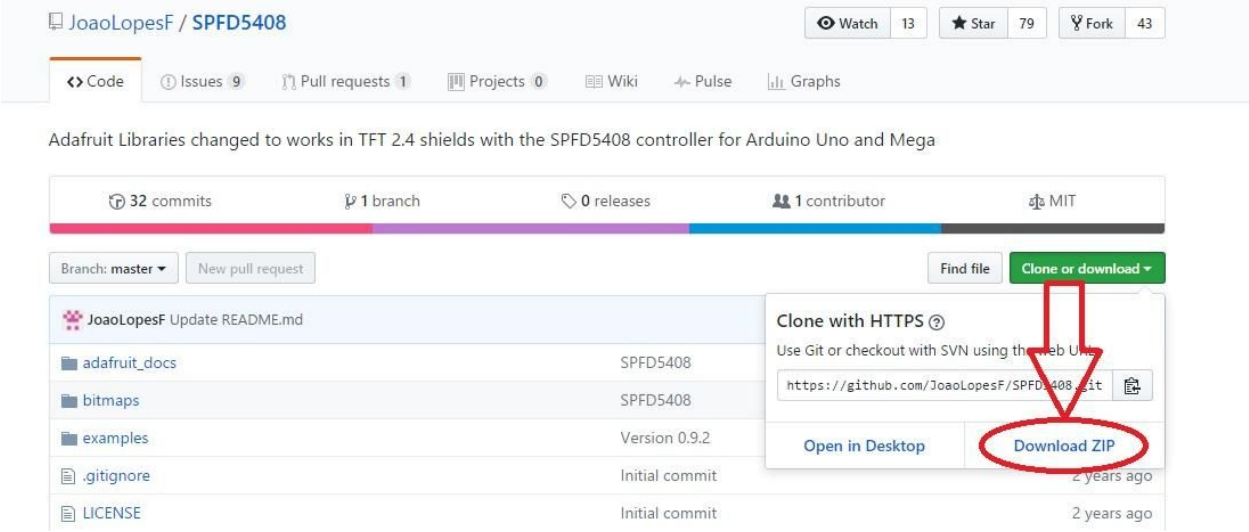
Selles peatükis räägitakse üldisemalt teekidest ja sellest, miks neid vaja on. Edasi saab lugeja teada, mis teeki on vaja, et antud lõputöös kasutatav LCD ekraan oleks programmeeritav. Autor otsustas kasutada olemasolevat teeki, kuna õppetöö eesmärk ei ole minna süvitsi teegi programmeerimisse, vaid teha mäng Arduino LCD-le. Lõpuks näidatakse, kuidas ühendada LCD ekraan Arduino Uno plaadiga ja kuidas jooksutada üks näidisprogramm.

3.2.1. Teegi allalaadimine

Programmeerimises nimetatakse teegiks kollektsiooni juba ette kompileeritud moodulitest. Programmid koosnevad moodulitest, mis koosnevad omakorda erinevatest funktsioonidest. Need moodulid on ladustatud objekti formaadis. Teegid on eriti kasulikud sageli kasutatud moodulite

hoiustamiseks, sest neid ei ole vaja eraldi kopeerida igasse programmi, mis neid kasutab. Ühesõnaga saab teekide abil kasutada juba ette kirjutatud funktsioone, nagu *drawRect(),* mis joonistab ekraanile antud argumentide põhjal vastava ristküliku. [21]

Igal LCD ekraanil on erinev teek, mida nad kasutavad. Kui teek on antud ekraani jaoks vale ja ei toeta seda LCD ekraani, siis ei ole võimalik kasutada ka selle funktsioone. Antud töös kasutatav LCD ekraani toetab teek nimega SPFD5408. Ekraani kasutamiseks ja programmeerimiseks on vaja tõmmata see teek Github`ist https://github.com/JoaoLopesF/SPFD5408. Teegi allalaadimiseks on vaja vajutada nupule *Download ZIP,* mis on näidatud joonisel 26.



Joonis 26. Teegi allalaadimine.

Allalaaditud *.zip* fail on vaja nüüd pakkida lahti Arduino kaustas olevasse *libraries* kausta, navigeerides järgmiselt: *See arvuti -> Dokumendid -> Arduino -> libraries*. Nüüd on olemas teek ning lugeja saab alustada programmeerimisega. Esimese asjana oleks vaja kontrollida, kas kõik töötab korrektselt. Selleks on Arduino programmeerimiskeskkonnas olemas testid, mida saab kohe jooksutada.

Teekide kasutamiseks on vaja esimese asjana nad programmikasutusele võtta. Seda saab teha käsuga *#include. Include* käsku kasutades saab programmeerija ligipääsu teegile ja seega välja

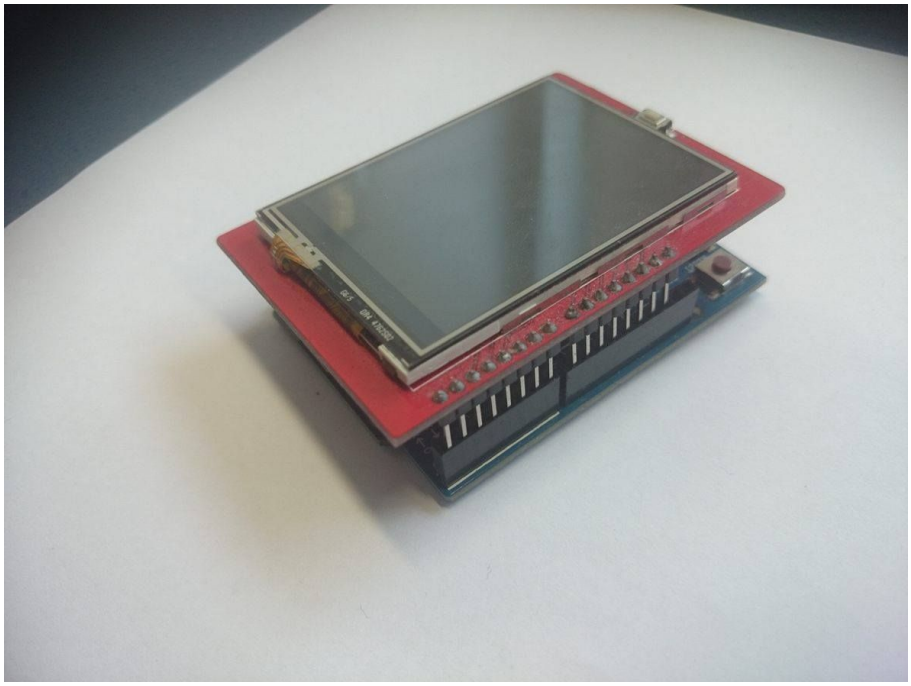
kutsuda selles ettekirjutatud funktsioone ja klasse. Lõputöös kasutatava ekraani programmeerimiseks on vaja kirjutada järgmised kolm rida programmi, kuna vajame funktsioone ja klasse nendest teekidest.

```
#include <SPFD5408_Adafruit_GFX.h>
#include <SPFD5408_Adafruit_TFTLCD.h>
#include <SPFD5408_TouchScreen.h>
```

Nüüd on teegid toodud programmi sisse ning seega saab ligi nende klassidele ja funktsioonidele. Edasi näidatakse, kuidas ühendada LCD ekraan Arduino Uno plaadiga.

3.2.2. Ekraani ühendamine Arduinoga

Nüüd ühendatakse LCD ekraani Arduino Uno plaadiga. Selleks on vaja ekraani viigud ühendada Arduino plaadi omadega. Antud ekraan kasutab digitaalseid viikusid D5-D13 ja analooge A0-A4. See tähendab, et vabaks jääb vaid üks analoogviik. Joonis 27 aitab lugejal aru saada, kuidas ta peab ühendama ekraani Arduino plaadiga.



Joonis 27. LCD ekraan ühendatud Arduino plaadiga.

Joonisel 27 on Arduino plaat lugeja poole digitaalsete viikudega. Digitaalsete viikude asukoht Arduino plaadil on näidatud ka joonisel 10 märgistatud numbriga 1.

Nüüd saab jooksutada mingi selle teegi poolt tagatud näidisprogrammi. Valime selleks lihtsa *graphicstest* programmi. Selle näidisprogrammi avamiseks tuleb Arduino IDE's navigeerida järgmiselt: *Fail -> Näited -> SPFD5408-master -> spfd5408_tftpaint.* Jooksutades seda programmi, peaksid ekraanile ilmuma erinevad visuaalid. See tähendab, et ekraan on korrektselt ühendatud Arduino Uno plaadiga ning Arduino programmeerimiskeskkond on edukalt üles seatud. Järgmisena näidatakse lugejale, kuidas kasutada ekraani vajutustundlikkust.

3.3. Ekraani vajutustundlikkuse seadistamine

Mängu ülevaates kirjutati, et tahetakse kasutada kindlasti vajutustundlikkust. Seda kasutatakse käesolevas lõputöös menüüs navigeerimiseks. Ekraani vajutustundlikkuse implementeerimiseks kasutatakse jällegi näidisprogrammi nagu eelmises peatükis. Seekord on näidisprogrammiks *tftpaint*, mis on sisuliselt ekraanile joonistamiseks loodud programm. Selle avamiseks peab navigeerima *Fail -> Näited -> SPFD5408-master -> spfd5408_tftpaint.* Programmis on esimese asjana näha, et on juba kaasatud ekraani jaoks vajalikud teegid. Järgmine tähtis koodilõik on viikude defineerimine. Arduino programmeerimiskeskkonnas saab defineerida väärtusi, kasutades käsku *#define.* See käsk tuleneb keelest C ja nagu eelnevalt kirjutati, kasutab Arduino IDE C/C++ hübriidset keelt. *#define* lubab programmeerijal anda nime konstantsele väärtusele enne programmi kompileerumist. Defineeritud konstandid ei võta programmi mälu, sest kompilaator asendab need eelprotsessi käigus fikseeritud bittide jadaga. [22] See tuleb käesolevas lõputöös kasuks, kuna ATmega328 mikrokontrolleril on 32KB mälu. Näidisprogrammis on viigud defineeritud järgmiselt:

```
#define YP A1
#define XM A2
#define YM 7
#define XP 6
```

Selle ekraaniga tekib siin probleem, kuna tegemist on kloonekraaniga Hiinast, kus aeti segamini

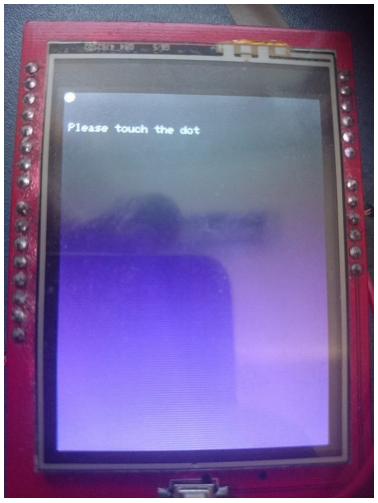
digitaalsed viigud D7 ja D5. See tähendab, et antud ekraaniga peab kasutama digitaalset viiku D5. Seega tuleb kirjutada eelmise rea asemel järgmise rea:

```
#define YM 5
```

Järgmisena on näidiskoodis näha koodilõiku, kus on kalibreeritud väärtused. Antud read näevad välja järgmised:

```
#define TS_MINX 125
#define TS_MINY 85
#define TS_MAXX 965
#define TS_MAXY 905
```

Kuna kalibreering on tehtud mingi erineva suuruse ekraaniga, siis tuleks see ise üle teha. Hetkel jooksutades programmi, saab joonistada ja valida värve, kuid vajutuse koordinaadid ei ole täpsed. Selleks on autor ise kalibreerinud väärtused korrektselt, kasutades näiteprogrammi *calibrate.* Näidisprogrammi *calibrate* avamiseks tuleb navigeerida Arduino programmeerimiskeskkonnas järgmiselt: *Fail -> Näited -> SPFD5408-master -> spfd5408_calibrate.* Programmi käivitamisel juhendatakse kasutajal teha ekraani ülesse vasakusse äärde üks vajutus (vt joonist 28) ja siis ekraani alla paremale üks vajutus. Järgmisena väljastatakse ekraanile *TS_MINX, TS_MINY, TS_MAXX ja TS_MAXY* väärtused sõltuvalt mõlema vajutuse asukohast.



Joonis 28. Näidisprogramm *calibrate* kuvatud LCD ekraanil.

Näidisprogrammiga *calibrate* saadud väärtused antud ekraani korral on järgmised:

```
#define TS_MINX 130
#define TS_MINY 120
#define TS_MAXX 920
#define TS_MAXY 920
```

Vajutustundlikkuse rakendamiseks on vaja kasutada funktsioone TouchScreen klassist, mis on defineeritud teegi <*SPFD5408_TouchScreen.h*> poolt. Klass on kollektsioon funktsioonidest ja muutujatest, mis on kõik hoitud ühes kohas. Need funktsioonid ja muutujad võivad olla avalikud, mis tähendab, et kõik, kes kasutavad seda teeki, pääsevad nendele ligi, või privaatsed, mis tähendab, et nendele pääseb ligi vaid klassi sees. Igal klassil on spetsiaalne funktsioon nimega konstruktor, mida kasutatakse, et luua klassist isend. Konstruktor on erikujuline klassimeetod uute isendite loomiseks. Konstruktoril on klassiga sama nimi ja tal puudub tagastustüüp. [23]

Antud programmis peab *TouchScreen* klassi funktsioonide rakendamiseks looma selle klassi isendi. Selle loomiseks kasutame *TouchScreen* konstruktorit, mis võtab sisse 5 erinevat argumenti. Esimesed 4 argumenti on eelnevalt defineeritud viigud. Viimane argument on takistuse väärtus oomides. Seda on vaja vajutuse tugevuse täpsuseks. Käesolevas lõputöös määratakse selle väärtuseks 100 oomi. Koodis näeb see välja järgmiselt:

```
TouchScreen ts = TouchScreen(XP, YP, XM, YM, 100);
```

Autor seadis selle isendi nimeks *ts*. Seega saab hiljem kasutada funktsioone sellest klassist nii, et ette kirjutada *ts*, millele järgneb funktsiooni nimi. Näiteks funktsiooni *getPoint(),* mis tagastab punkti kaks koordinaati x ja y ning vajutuse tugevuse väärtuse z, kasutamiseks tuleb kirjutada *ts.getPoint()*. Järgmisena tuuakse ka näide selle funktsiooni kasutusest:

```
TSPoint waitOneTouch() {
  TSPoint p;
  do {
    p= ts.getPoint();
    pinMode(XM, OUTPUT);
    pinMode(YP, OUTPUT);
  } while((p.z < MINPRESSURE )|| (p.z > MAXPRESSURE));
  return p;
}
```

Antud funktsioon on võetud *tftpaint* näiteprogrammist. See funktsioon tagastab klassi *TSPoint* muutuja väärtuse. Algselt defineeritakse klassi *TSPoint* muutuja nimi täheks p. Järgmisena kasutatakse *do-while* tsüklit. *Do-while* tsükkel töötab samamoodi nagu tavaline *while* tsükkel, kuid ainuke erinevus on see, et tingimust kontrollitakse alles tsükli lõpus [20]. Seega tsükkel jookseb alati vähemalt ühe korra. Antud tsükkel jookseb nii kaua, kuni kasutaja teeb ühe puute ekraanile. Tingimuseks on seatud selles tsüklis

```
(p.z < MINPRESSURE )|| (p.z > MAXPRESSURE)
```

Selle tingimuse vääraks muutumisel lõpeb tsükkel. Eelnevalt mainiti, et funktsioon *getPoint()* tagastab 3 erinevat koordinaati. Nende koordinaatide tähendused on järgmised:

- x - Puute asukoht ekraani x-telje peal. Eelnevalt programmis defineeritud minimaalne x väärtus nimega *TS_MINX* on 130 ja maksimaalne *TS_MAXX* on 930. Seega x koordinaadi väärtus saab olla vahemikus [130, 930].
- y - Puute asukoht ekraani y-telje peal. Eelnevalt programmis defineeritud minimaalne y

väärtus nimega *TS_MINY* on 120 ja maksimaalne *TS_MAXY* on 930. Seega y koordinaadi väärtus saab olla vahemikus [120, 930].

- z - Puute tugevus ekraani peal. *MINPRESSURE* on seehulgas minimaalne puutetugevus ja *MAXPRESSURE* maksimaalne puutetugevus. Need väärtused on eelnevalt defineeritud *TouchScreen* klassis.

Käesoleva tsükli tingimuses kasutatakse ainult z koordinaati. Vaadatakse, kas puutetugevuse väärtus on väiksem, kui minimaalne puutetugevus või suurem kui maksimaalne puutetugevus. Seega on antud tingimus tõsi vaid siis, kui kasutaja ei puutu ekraani. Ekraani puudutades muutub tingimus vääraks ja tsükkel lõpetab tegutsemise. Tingimus on väär siis, kui mõlemad võrratused on väärad. Esimene võrratus muutub vääraks, kuna ekraani puudutades on *p.z* suurem *MINPRESSURE* väärtusest. Teine väärtus muutub vääraks, kuna ekraani puudutades ei saa olla puutetugevuse väärtus suurem kui *MAXPRESSURE* väärtus.

Sarnast funktsiooni rakendatakse ka mängu loomisel. Mängus menüüs navigeerimine on programmeeritud väga sarnaselt eelneva funktsiooniga. Erinevus on see, et lisaks z koordinaadile kasutatakse veel y koordinaati, et programm teaks, mis valikule on kasutaja vajutanud. Näiteks algmenüüs näeb puute kontrollimise kood välja järgmine:

```
bool vajutatudSKOORITABEL = false;
bool vajutatudALUSTA = false;
TSPoint p;
do {
    p= ts.getPoint();
    pinMode(XM, OUTPUT);
    pinMode(YP, OUTPUT);

     if (p.z > MINPRESSURE && p.z < MAXPRESSURE && p.y > 600 && p.y <
780) {
      vajutatudSKOORITABEL = true;
    }
     if (p.z > MINPRESSURE && p.z < MAXPRESSURE && p.y > 800 && p.y <
900){
      vajutatudALUSTA = true;
    }
  } while(vajutatudSKOORITABEL == false && vajutatudALUSTA == false);
```

```
if (vajutatudSKOORITABEL) {
  kuvaSkooritabel(skoor);
}
else {
  alustaMäng();
}
```

See tsükkel jookseb nii kaua, kuni vajutatakse menüü ühe valiku peale. Tsükli tingimus on tõene, kui mõlemad väärtused (*vajutatudSKOORITABEL* ja *vajutatudALUSTA*) on väärad. Tsükli tingimus muutub vääraks, kui kasutaja vajutab menüü valikule, mis asub ekraanil y koordinaatide 600 ja 780 vahel või 800 ja 900 vahel. Pärast tsükli lõppu kontrollitakse, kumba menüü valikut vajutati. Selleks kontrollitakse, kumb nendest väärtustest on tõene ning vastavalt sellele viiakse kasutaja ekraanis edasi. Järgmisena näidatakse lugejale, kuidas ühendada SD-kaart ekraaniga ning salvestada sellele andmeid.

3.4. Ekraanile joonistamine

Käesolevas lõputöös on vaja luua mitmeid erinevaid objekte. Objektide all mõeldakse udukogusid, kosmoselaeva, jooni, ristkülikuid ja teksti. Graafika peatükis näidati, millised need välja näevad. Esiteks luuakse udukogud, mis peavad liikuma ekraanil ülevalt alla. Udukogude joonistamiseks kasutatakse funktsioone <SPFD5408_Adafruit_TFTLCD.h> teegist. Nende funktsioonide kasutamiseks on vaja defineerida eelnevalt 5 analoogviiku, mida ekraan vajab. Viikudest, mida antud ekraan kasutab, kirjutati täpsemalt LCD ekraani peatükis. Ühesõnaga kasutab see kuvar 5 erinevat analoogviiku. Seega peab need programmis ka defineerima. Defineerime need viigud järgmiselt:

```
#define LCD_CS A3
#define LCD_CD A2
#define LCD_WR A1
#define LCD_RD A0
#define LCD_RESET A4
```

Edasi on vaja luua ekraani isend, mille kaudu saame ligi selle teegi funktsioonidele. Autor nimetas selle lihtsalt tft. See luuakse järgmiselt:

```
Adafruit_TFTLCD tft(LCD_CS, LCD_CD, LCD_WR, LCD_RD, LCD_RESET);
```

Nüüd on võimalik kasutada funktsioone sellest teegist, kirjutades ette tft. Näiteks udukogu loomisel on vaja kasutada *drawRect()* funktsiooni, mis joonistab antud parameetrite põhjal ristküliku. *DrawRect* võtab sisse 5 erinevat argumenti. Esimesed 2 argumenti tähistavad x ja y koordinaadi alguskohta, kust ristkülikku hakatakse joonistama. Teised 2 argumenti tähistavad x ja y koordinaatide lõppkohta. Viimane argument tähistab ristkülikku värvi. Näiteks rohelise udukogu loomiseks ekraani üles vasakusse nurka, mida on näha ka joonisel 1, tuleb kirjutada järgmine rida:

```
tft.fillRect (0, 0, 40, 40, ROHELINE);
```

Värvide kasutamiseks on vaja defineerida samuti varem ära ka kõik vajaminevad värvid. Väärtusteks on siin 16-bit värvid. Selleks kirjutas autor järgmise koodilõigu:

```
#define MUST     0x0000
#define SININE   0x001F
#define PUNANE   0xF800
#define ROHELINE 0x07E0
#define KOLLANE  0xFFE0
#define VALGE    0xFFFF
```

Udukogude liikuma panemiseks tehakse kõik 4 argumenti muutujateks ning iga mängutsükkel viib udukogusi pikslihaaval ekraani y teljel alla. Nüüd on ristkülikute joonistamine ekraanile näidatud. Edasi näidatakse, kuidas luua teksti ekraanile. Selleks tuuakse näide peamenüüst, mida on näha ka joonisel 4.

```
tft.fillScreen(MUST);
tft.setCursor (20, 30);
tft.setTextSize (4);
tft.setTextColor(ROHELINE);
tft.println("Kosmose");
tft.setCursor (80, 85);
tft.setTextSize (4.5);
```

```
tft.setTextColor(PUNANE);
tft.println("Ralli");
```

Esmalt kasutatakse funktsiooni *fillScreen(),* mis võtab argumendiks värvi. See funktsioon värvib terve ekraani antud värviga. Hetkel kasutatakse eelnevalt defineeritud väärtust *MUST* ehk musta värvi. Nüüd kasutatakse funktsiooni *setCursor()*, mis võtab argumentideks kaks koordinaati ekraani teljelt. See funktsioon määrab ära punkti, kust järgnev kiri hakkab pihta. Antud olukorras määratakse x koordinaadiks 20 ja y koordinaadiks 30. Edasi rakendatakse funktsiooni *setTextSize(),* mille argument määrab ära järgneva teksti suurus. Argumendi väärtust muutes muutub teksti kõrgus 10 piksli võrra. Näiteks, kui argumendiks antakse 1, siis on teksti kõrgus 10 pikslit ja argumendi 2 puhul on kõrguseks 20 pikslit. Edasi rakendatakse funktsiooni *setTextColor(),* mis võtab argumendiks 16-bit väärtuses värvi. Siin kasutatakse argumendina eelnevalt defineeritud rohelist värvi nimega *ROHELINE.* Teksti kirjutamiseks rakendatakse funktsiooni *println(),* mis võtab argumendiks sõne, mida kuvatakse eelnevalt määratud omadustega ekraanile ehk siis tekst algab x koordinaadilt 20 ja y koordinaadilt 30, teksti kõrguseks saab 40 pikslit ning värv on roheline. Järgmise teksti kuvamiseks peab nüüd liikuma uutesse koordinaatidesse ning tegutsema samalaadselt. Edasi näidatakase, kuidas kuvatakse kaks erinevat valikut, milleks on “SKOORITABEL” ja “ALUSTA”, ekraanile.

```
tft.drawRect (20, 180, 200, 60, VALGE);
tft.setCursor (55, 200);
tft.setTextSize (2);
tft.setTextColor(VALGE);
tft.println("SKOORITABEL");
tft.drawRect (20, 250, 200, 60, VALGE);
tft.setCursor (90, 270);
tft.setTextSize (2);
tft.setTextColor(VALGE);
tft.println("ALUSTA");
```

Rakendatakse funktsiooni *drawRect(),* mis on sarnane eelnevalt udukogude loomisel kasutatavale funktsioonile *fillRect().* Ainuke erinevus seisneb selles, et ristkülikut ei värvita seest ära. Viimane argument muudab vaid ristküliku servade värvi. Nagu näha jooniselt 4, on valikud valgelt joondatud ristkülikute sees. Edasi kirjutatakse, kuidas implementeerda mängule skoori

salvestamine ja selle kuvamine skooritabelis.

3.5. MicroSD-kaardile andmete salvestamine ja nende lugemine

Antud peatükk räägib skoori salvestamisest ja skooritablei koostamisest. Esimese asjana on vaja sisestada microSD-kaart ekraani laiendusplaati. Ekraanil asub SD-kaardi ühendus laiendusplaadi taga alumises servas. Seda asukohta on näha jooniselt 12. MicroSD-kaart on vaja sisestada korrektselt. Järgmisena seletatakse täpsemalt, kuidas saab microSD-kaardile luua .txt fail ja salvestada sinna andmeid ning hiljem neid lugeda ja kuvada skooritabelis.

Programmis SD-kaardiga töötamiseks tuleb jällegi kasutada teeki. Autor kasutab William Greimani poolt tehtud teeki <SD.h>. See teek lubab lugeda SD-kaardilt ja kirjutada SD -kaardile. Teegi sissetoomiseks programmi tuleb kirjutada järgnev rida:

```
#include <SD.h>
```

Nagu ekraanigi tööle saamiseks on ka siin vaja defineerida digitaalne viik. Selleks peab valima digitaalse viigu D10, kuna kõik teised on kinni. Viigu defineerimiseks on vaja kasutada *#define* käsku järgnevalt:

```
#define SD_CS 10
```

SD teegi ja kaardi lähtestamiseks on vaja mängu programmi *setup'*is rakendada *SD.h* teegi funktsiooni *begin()*. Antud funktsioonile saab anda ka ühe argumendi. Ilma argumendita alustaks see funktsioon ka digitaalse viigu D10 kasutamist, kuid argumendiga saab seda viiku ise valida. Autor defineeris eelnevalt selleks viiguks D10, seega pole vaja lisada argumenti *begin()* funktsioonile. Käesoleva mängu programmis näeb selle funktsiooni kasutamine välja järgmiselt:

```
SD.begin();
```

Skoori salvestamiseks microSD-kaardile on vaja luua algselt .txt fail. SD-kaardil oleva .txt faili avamine ja sellesse kirjutamine näeb välja järgmiselt:

```
minuFail = SD.open("uusSkoor.txt", FILE_WRITE);
minuFail.println(nimi + ":" + skoor);
minuFail.close();
```

Faili loomiseks kasutatakse SD teegist funktsiooni *open()*, mis avab faili SD-kaardil. Kui seda faili pole olemas, siis see luuakse. *Open()* funktsioon võtab sisse kaks argumenti. Esimene argument on faili asukoht. Antud programmis on määratud asukohaks tekstifail *uusSkoor.txt*. Teine argument tähistab seda, mis viisil tahetakse faili avada. Kokku on kaks erinevat viisi - *FILE_READ ja FILE_WRITE. FILE_READ* puhul avatakse fail lihtsalt lugemiseks. *FILE_WRITE* puhul avatakse fail lugemiseks ja kirjutamiseks. Hetkel kasutatakse *FILE_WRITE* viisi, kuna järgmisena on vaja kirjutada faili skoor. Skoori kirjutamiseks kasutame funktsiooni println(), mis sisestab ühe rea koos reavahetusega tekstifaili. Funktsioon võtab argumendiks teksti, mida soovitakse lisada. Kui vajalik tekst on lisatud, suletakse tekstifail.

Nüüd näidatakse, kuidas lugeda andmeid microSD-kaardilt. Selleks kasutatakse sama funktsiooni SD teegist, milleks on *open()*. Seekord aga *FILE_WRITE* asemel tuleb teiseks argumendiks *FILE_READ.* Antud koodirida näeb välja järgmine:

```
minuFail = SD.open("uusSkoor.txt", FILE_READ);
```

Nüüd saab kasutada funktsiooni *read(),* mis tagastab failis järgmise baiti või tähe. Autor loeb andmed tekstifailist ja kirjutab need skooritabelisse järgmiselt:

```
if (minuFail) {
  while (minuFail.available()) {
    tft.setCursor (x, y);
    char tähemärk = char(minuFail.read());

    if (tähemärk == '\n'){
      y += 20;
```

```
      x = 30;
    } else {
      x += 13;
      tft.print(tähemärk);
    }
  }
  //sulgeb faili
  myFile.close();
} else {
  // kui fail millegipärast ei avanud, väljastatakse error
  Serial.println("error faili avamisega");
}
```

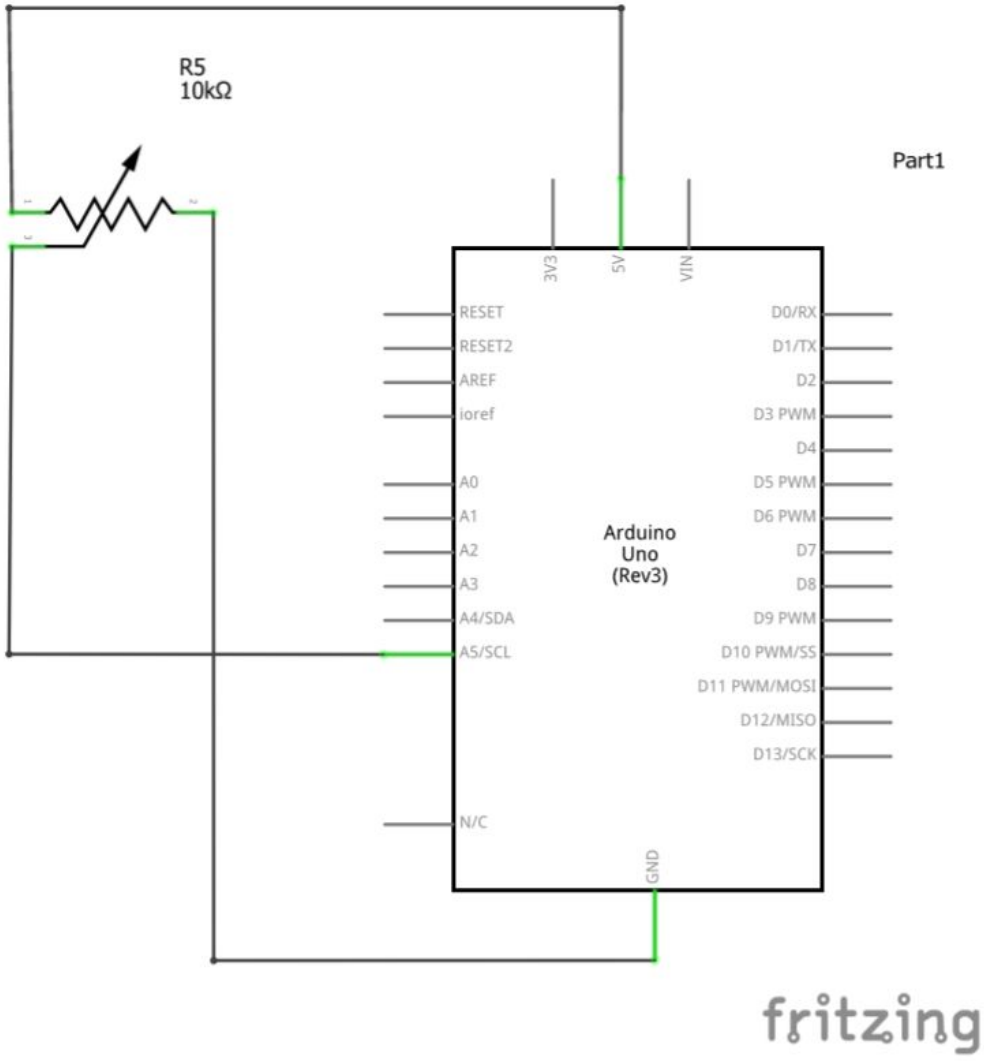
Esmalt kontrollitakse, kas see fail eksisteerib. Järgmisena kasutatakse *while* tsüklit. *While* tsükkel käib nii kaua, kuni tingimus selle sulgudes muutub vääraks. Tingimuseks on *minuFail.available(),* mis tähendab, et käsitletavale failile rakendatakse funktsioon *available().* See funktsioon kontrollib, kas selles failis on infoühikuid järele lugemiseks. Infoühikuteks selles failis on näiteks täht, number või tühik. Kui ei leidu ühtegi infoühikut, siis lõppeb ka *while* tsükkel. Tsükli sees kasutatakse *setCursor()* funktsiooni, millest räägiti täpsemalt peatükis 3.4. See määrab ära, kuhu skooritabelis hakatakse kirjutama. Nüüd luuakse muutuja *tähemärk,* mis muudab sisseloetud infoühiku tähemärgi tüüpi isendiks. Seega saab nüüd kasutada muutujat *tähemärk* kui tähte ning kirjutada funktsiooni *print()* abil ekraanile. Funktsioon *print()* on sarnane peatükis 3.4. mainitud funktsioonile *println().* Ainuke vahe seisneb selles, et *print()* ei lisa reavahetust ehk tähemärki '\n'. Järgmisena kontrollitakse, kas *tähemärk* on reavahetus. Sellel juhul tuleb seada y koordinaat funktsioonis *setCursor()* 20 pikslit allapoole, kuna see tähendab, et tekstifailis on tegemist rea vahetusega ehk ühe kasutaja nimi ja skoor on juba loetud. Kui *tähemärk* pole reavahetus, siis kirjutatakse see tähemärk ekraanile ning lisatakse x koordinaadile 13 pikslit juurde, et järgmine tähemärk tuleks eelmise kõrvale. Skooritabel on näha ka joonisel 6. Nüüd on näidatud, kuidas loetakse andmeid tekstifailist ja kirjutatakse need ekraanile. Edasi räägitakse juhtkontrollerite ühendamisest ja nende implementeerimisest mängu.

3.6. Juhtkontrollerite ühendamine

Mõlemad juhtkontrollerid ühendatakse analoogviiku A5, kuna see on ainuke vaba viik.

Ülejäänud analoogviigud on kasutusel LCD ekraani poolt. Siin tekkis ka autoril probleem, sest samas viigus kaks erinevat kontrollerit segavad teineteist. Seega on vaja kasutada lülitit, mis loob valiku kahe kontrolleri vahel, lülitades ainult ühte kontrollerisse +5 volti voolu sisse.

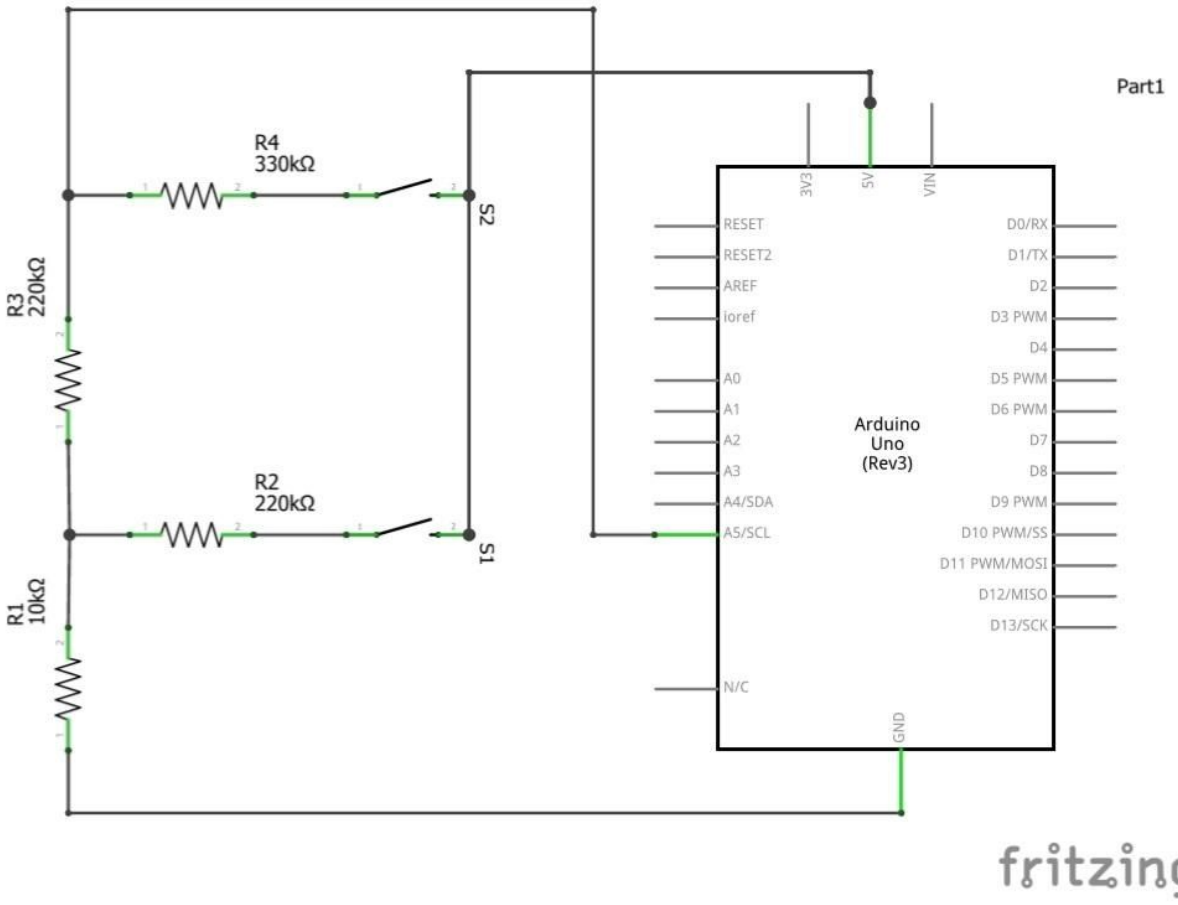
Esmalt ühendab autor liuguri analoogviiguga A5. Liuguri kasutamiseks on vaja kolm juhet ühendada Arduino plaadiga. Joonisel 29 on näidatud, kuidas need juhtmed ühendada.



Joonis 29. Vooluringi skeem liuguri ühendamisest Arduino Uno plaadiga. Joonis on koostatud autori poolt, kasutades Fritzing tarkvara.

GND ehk maandus tuleb ühendada Arduino plaadi maandusega. VCC ehk toitepinge tuleb ühendada Arduino plaadi viiguga 5V, mis väljastab 5V voolu. OTB ehk väljund B ühendatakse Arduino plaadil analoogviiguga A5. Nüüd on liugur ühendatud.

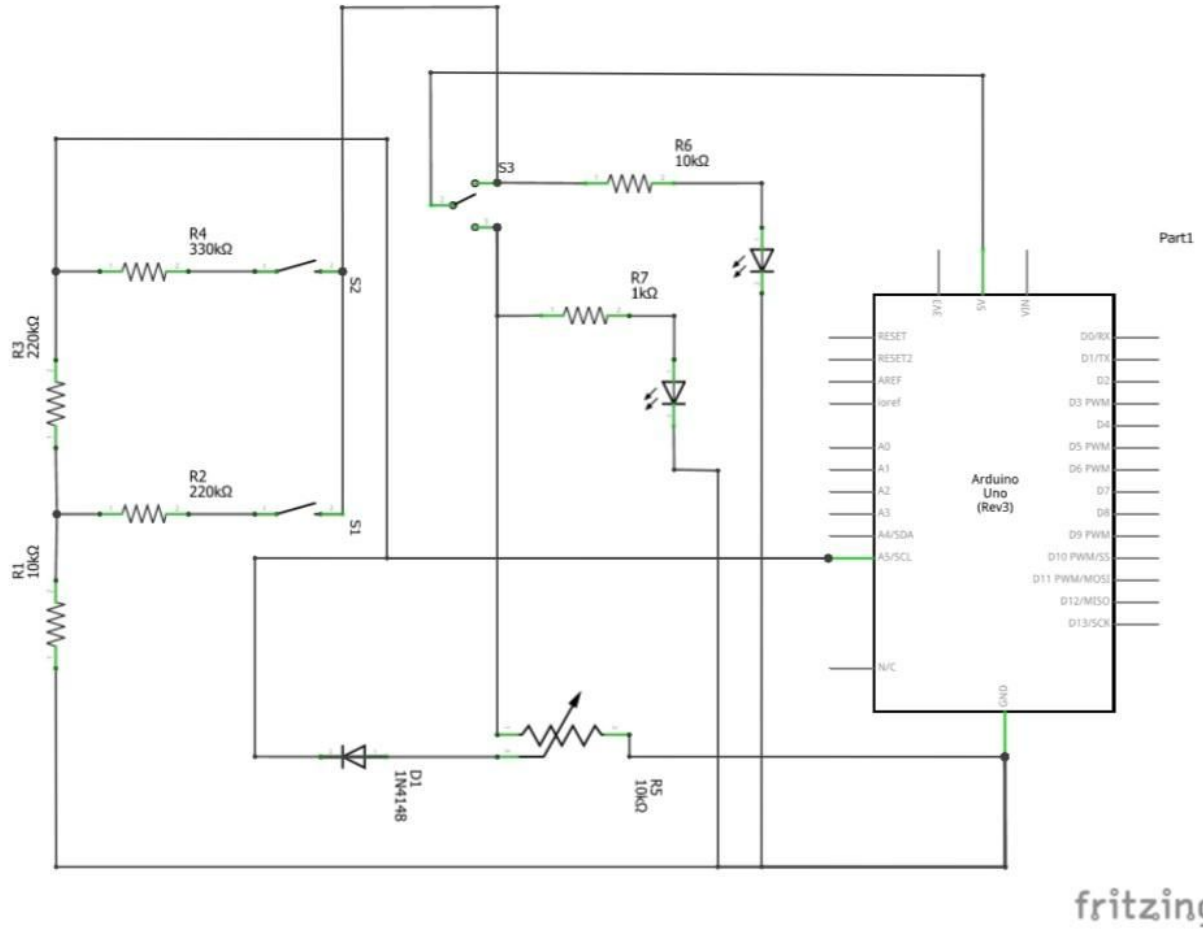
Teiseks ühendatakse nupud. Joonis 30 näitab, kuidas autor ühendas mõlemad nupud samuti analoogviiguga A5.



Joonis 30. Vooluringi skeem mõlema nupu ühendamisest Arduino Uno plaadiga. Joonis on koostatud autori poolt, kasutades Fritzing tarkvara.

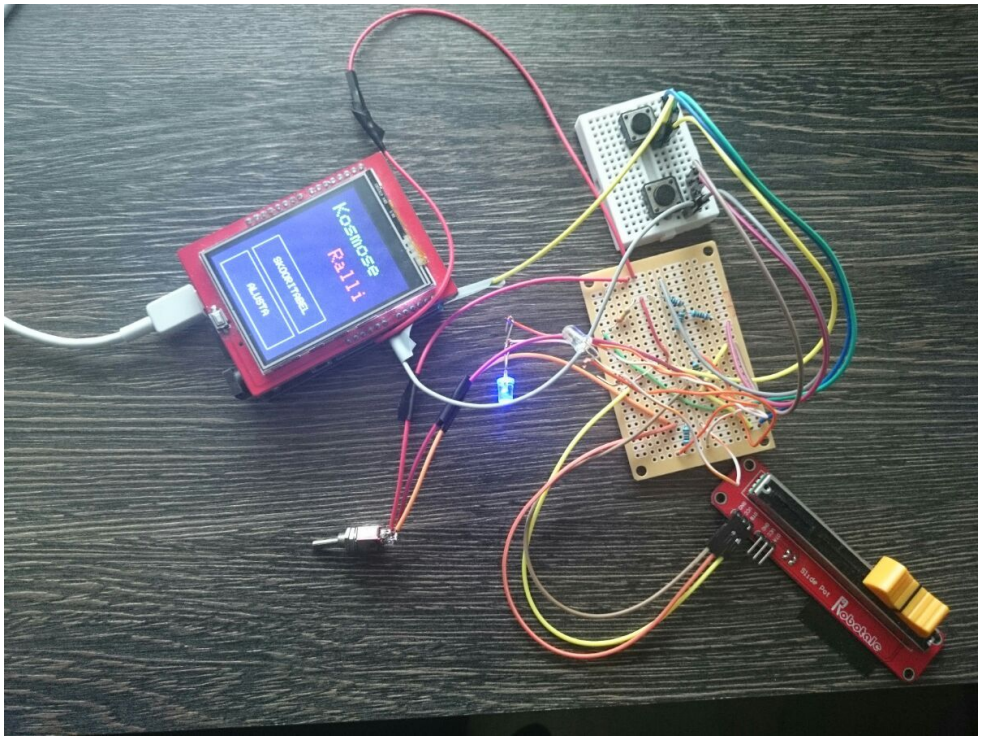
Jooniselt 30 on näha, et autor on määranud mõlemale nupule erineva takistuse. Selleks lisatakse skeemi takisteid. Takistid on tähistatud suure tähega R. Nupud on joonisel 30 tähistatud S1 ja S2.

Nüüd on mõlemad juhtkontrollerid ühendatud ning tuleb skeemi lisada lüliti, mis eemaldaks ühe kontrolleri vooluringlusest.



Joonis 31. Vooluringi skeem mõlema juhtkontrolleri ühendamisest Arduino Uno plaadiga koos lüliti ja kahe valgusdioodiga (toite valgusdiood puudub). Joonis on koostatud autori poolt, kasutades Fritzing tarkvara.

Joonise 31 vooluringi skeemi reaalset teostus on näha joonisel 32.



Joonis 32. Autori poolt kokku joodetud skeem, kus nupud on hetkel veel jootmata ja maketeerimislaual.

Autor kasutab väärtuste lugemiseks Arduino programmeerimiskeskkonnas olemasolevat funktsiooni *analogRead()*. See funktsioon võtab sisse ka ühe argumendi, milleks on analoogviigu number. Kuna antud lõputöös saab kasutada juhtkontrollerite jaoks vaid analoogviiku A5, siis selleks väärtuseks saab 5. Antud funktsioon loeb analoogviigu A5 pinget ning muudab selle vastavalt numbriks vahemikus 0 kuni 1023. [24] Funktsiooni *analogRead()* väärtusi analoogviigul A5 saab kontrollida järgmise näidisprogrammiga:

```
int analoogViik = 5;
int väärtus = 0;

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

void loop() {
  väärtus = analogRead(analoogViik);
  Serial.println(väärtus);
```

```
}
```

Jooksutades seda programmi, peab olema jadapordi monitorist näha liuguri takistuse väärtusi. Jadapordi monitor asub Arduino IDE üleval ribas. Selle avamiseks on vaja vajutada nuppu, mida on näha joonisel 33.

Joonis 33. Jadapordi monitor Arduino programmeerimiskeskkonnas.

Lülitades liugur vooluringlusesse ja liigutades selle peal olevat potentsiomeetrit muutuvad jadapordis väärtused. Vajutades aga nuppe ei juhtu midagi. Nüüd lülitades nupud vooluringlusesse on näha väärtuste muutumist, kui vajutada kummalegi nupule. Jadapordi monitorist on näha, et liuguri korral on väärtused vahemikus 0 kuni 1023. Vasakut nuppu vajutades on väärtused 24 ja 26 vahepeal ning parema nupu korral on väärtused 1020 ja 1024 vahepeal. Nende väärtuste abil saab kosmoselaeva liigutamiseks seada vastavad tingimused.

3.7. Kosmoselaeva liigutamine

Kasutaja peab saama liigutada kosmoselaeva vasakule ja paremale. Autor toob järgmisena näite enda koodist, kuidas ta liigutab kosmoselaeva vasakule.

```
void liiguV (int x, int y) {
         tft.fillRect(x+13-LIIKUMISPIKSEL,  y+6,  LIIKUMISPIKSEL,  8,
SININE);
    tft.fillRect(x+13-LIIKUMISPIKSEL, y , LIIKUMISPIKSEL, 2, VALGE);
        tft.fillRect(x+11-LIIKUMISPIKSEL,  y+2  ,  LIIKUMISPIKSEL,  2,
VALGE);
    tft.fillRect(x+9-LIIKUMISPIKSEL, y+4 , LIIKUMISPIKSEL, 2, VALGE);
    tft.fillRect(x+7-LIIKUMISPIKSEL, y+6, LIIKUMISPIKSEL, 2, VALGE);
    tft.fillRect(x+5-LIIKUMISPIKSEL, y+8 , LIIKUMISPIKSEL, 2, VALGE);
    tft.fillRect(x+3-LIIKUMISPIKSEL, y+10, LIIKUMISPIKSEL, 2, VALGE);
    tft.fillRect(x+1-LIIKUMISPIKSEL, y+12, LIIKUMISPIKSEL, 2, VALGE);
```

```
    tft.fillRect(x-LIIKUMISPIKSEL, y+14 , LIIKUMISPIKSEL, 5, VALGE);
    tft.fillRect(x+8-LIIKUMISPIKSEL, y+19, LIIKUMISPIKSEL, 2, VALGE);
    tft.fillRect(x+6-LIIKUMISPIKSEL, y+21, LIIKUMISPIKSEL, 2, VALGE);
    tft.fillRect(x+4-LIIKUMISPIKSEL, y+23, LIIKUMISPIKSEL, 2, VALGE);
    tft.fillRect(x+2-LIIKUMISPIKSEL, y+25, LIIKUMISPIKSEL, 2, VALGE);
    tft.fillRect(x-LIIKUMISPIKSEL, y+27 , LIIKUMISPIKSEL, 3, VALGE);
       tft.fillRect(x+17-LIIKUMISPIKSEL,  y+6  ,  LIIKUMISPIKSEL,  8,
VALGE);
           tft.fillRect(x+7-LIIKUMISPIKSEL,   y+30,LIIKUMISPIKSEL,10,
KOLLANE);
             tft.fillRect(x+19-LIIKUMISPIKSEL,y+30,LIIKUMISPIKSEL,10,
KOLLANE);
    tft.fillRect(x+17-LIIKUMISPIKSEL, y , LIIKUMISPIKSEL, 2, MUST);
    tft.fillRect(x+19-LIIKUMISPIKSEL, y+2, LIIKUMISPIKSEL, 2, MUST);
    tft.fillRect(x+21-LIIKUMISPIKSEL, y+4, LIIKUMISPIKSEL, 2, MUST);
    tft.fillRect(x+23-LIIKUMISPIKSEL, y+6, LIIKUMISPIKSEL, 2, MUST);
    tft.fillRect(x+25-LIIKUMISPIKSEL, y+8, LIIKUMISPIKSEL, 2, MUST);
    tft.fillRect(x+27-LIIKUMISPIKSEL, y+10, LIIKUMISPIKSEL, 2, MUST);
    tft.fillRect(x+29-LIIKUMISPIKSEL, y+12, LIIKUMISPIKSEL, 2, MUST);
    tft.fillRect(x+30-LIIKUMISPIKSEL, y+14, LIIKUMISPIKSEL, 5, MUST);
    tft.fillRect(x+22-LIIKUMISPIKSEL, y+19, LIIKUMISPIKSEL, 2, MUST);
    tft.fillRect(x+24-LIIKUMISPIKSEL, y+21, LIIKUMISPIKSEL, 2, MUST);
    tft.fillRect(x+26-LIIKUMISPIKSEL, y+23, LIIKUMISPIKSEL, 2, MUST);
    tft.fillRect(x+28-LIIKUMISPIKSEL, y+25, LIIKUMISPIKSEL, 2, MUST);
    tft.fillRect(x+30-LIIKUMISPIKSEL, y+27, LIIKUMISPIKSEL, 3, MUST);
    tft.fillRect(x+11-LIIKUMISPIKSEL, y+30, LIIKUMISPIKSEL,10, MUST);
    tft.fillRect(x+23-LIIKUMISPIKSEL, y+30, LIIKUMISPIKSEL,10, MUST);
  }
```

Antud funktsioon ei tagasta midagi ja võtab argumentideks väärtused x ja y. Need väärtused tähistavad kosmoselaeva hetkelist x ja y koordinaati. Ekraani koordinaatide telge on näha joonisel 10. Funktsiooni käivitamisel liigutatakse kosmoselaev vasakule *LIIKUMISPIKSEL* väärtuse võrra. Selleks väärtuseks on autor seadnud 4 ehk siis seesama kosmoselaev joonistatakse 4 pikslit vasakule. Varem seletati, et kosmoselaev on siin mängus ainuke detailsem objekt. Siit on nüüd näha, miks ei saa teha palju detailseid objekte. Põhjuseks on see, et liiga palju peab kasutama funktsiooni *fillRect()* ja mäng muutub seega liiga aeglaseks. Komsoselaeva paremale liigutamine on analoogne selle funktsiooniga. Funktsiooni *liiguP(mängijaX, mängijaY)* saab lugeja ise üles otsida autori koostatud mängu lähtekoodist.

Autor kasutab eelnevat funktsiooni, nii et algselt kontrollib *analogRead()* funktsiooni abil, mis

on hetkel selle tagastatud väärtus analoogviigus A5. Nende väärtuste muutmisest ja kuvamisest kirjutati täpsemalt peatükis 3.6. Kuna nuppudele vajutades on erinevad väärtused, siis saab kontrollida, kummale nupule vajutatakse vaid kahe tingimuslausega. Siin tuuakse selle kohta ka lugejale näide.

```
int mängijaX = 105;
int mängijaY = 280;
int pinge = analogRead(5);
if (pinge  == vasaku_nupu_pinge){
      if (mängijaX >= 45){
                  liiguV(mängijaX, mängijaY);
                  mängijaX -= LIIKUMISPIKSEL;
      }
} else if (pinge  == parema_nupu_pinge) {
      if (mängijaX <= 165){
                  liiguP(mängijaX, mängijaY);
                  mängijaX += LIIKUMISPIKSEL;
      }
}
```

Algselt määrab autor ära kosmoselaeva x ja y koordinaadid. Järgmisena loob muutuja *pinge,* mis loeb analoogviigult A5 pinget. Nüüd kontrollitakse, kas see loodud muutuja on võrdne vasaku nupu pingega. Kui see tingimus on tõsi, siis kontrollitakse, kas kosmoselaeva x koordinaat on suurem kui mängu vasaku piiri koordinaat, et laev ei lendaks ekraanist välja. Nüüd kasutatakse eelnevalt mainitud funktsiooni *liiguV (mängijaX, mängijaY)*. Pärast seda funktsiooni kasutust lahutatakse muutujast *mängijaX* väärtus *LIIKUMISPIKSEL*. Järgmine tingimus kontrollib, kas vajutatakse paremat nuppu ning see töötab analoogselt esimese kontrolliga. Nüüd saab kosmoselaeva juhtida nuppudega.

4. Korpuse loomine

Käesoleva lõputöö üheks eesmärgiks oli ka teha korpus mängu jaoks. Korpus valmistatakse 3D printeri abiga ja see disain tehakse programmis nimega Solid Edge Academic. Antud peatükis räägitakse pikemalt 3D mudeli tegemisest ja miks selline disain luuakse. Ülevaatlikult kirjutatakse ka 3D printimisest ja näidatakse prototüüpi valminud mängust.

4.1. 3D mudeli loomine

3D disainiks nimetatatakse protsessi, kus arvuti abil joonistatakse ja joonestatakse ruumiline mudel. Disaini loomiseks kasutatavd tarkvarad võib jagada nende funktsioneerimise põhimõttel kaheks:

- Ruumiliste kehadega disain - Kergem on alustada, aga tehnilised võimalused on piiratud. Siia alla käivad programmid nagu 3D Creationist, Google Sketchup, TinkerCad ja 3D crafter. [25]
- Professionaalsed CAD tarkvarad - Raskem on alustada, aga tehnilised võimalused on piiramatud. Siia alla käivad programmid nagu Solid Edge Academic, SolidWorks ja Autodesk Fusion 360. [25]

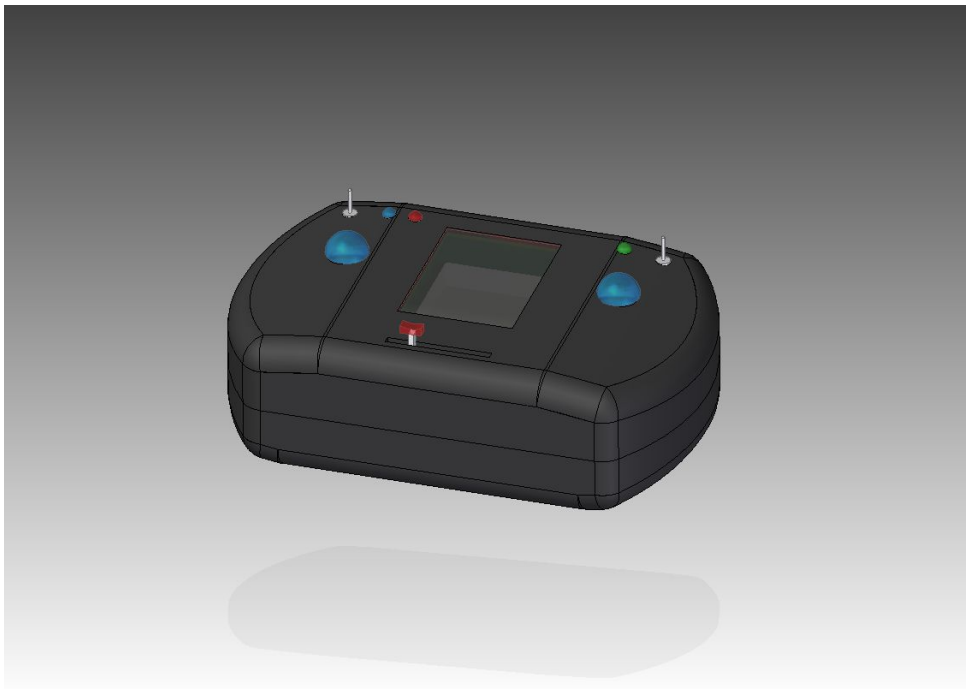
3D mudeli joonestamiseks kasutas autor proffesionaalset CAD tarkvara Solid Edge Academic. CAD ehk raalprojekteerimine on arvuti abil toodete, seadmete ja mudelite kujundamine [26]. Solid Edge Academic on akadeemiline versioon Solid Edge tarkvarast, millel on kõik pärisversiooniga sarnased funktsionaalsused. Ainukeseks erinevuseks on see, et joonistele märgitakse juurde, et tegu on akadeemilise versiooniga koostatud joonistega. Akadeemiline versioon on laialdaselt kasutatav akadeemilistes ringkondades, sest see on tasuta ja piiramatute tehniliste võimalustega CAD tarkvara. [25]

Antud 3D mudeli loomisel pidas autor silmas järgmisi nõudeid:

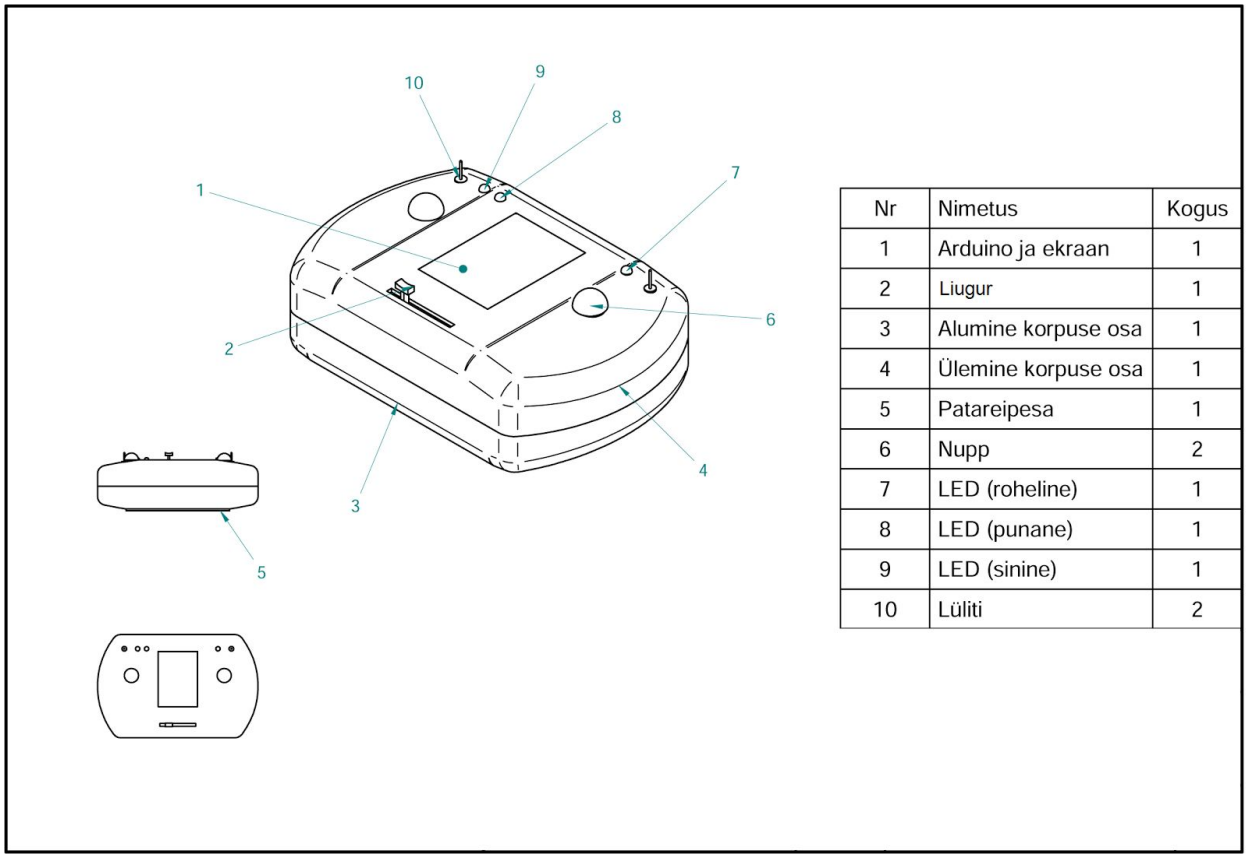
1. Mudel peab järgima kõiki esimeses peatükis seadistatud nõudeid.
2. Mudel peab olema kasutajasõbralik ja mugav käes hoida.

3. Mudel peab korralikult mahutama ära kõik riistvara.
4. Mudel peab esinduslik välja nägema.

Joonisel 34 on näidatud, milline näeb valminud 3D mudel välja Solid Edge Academic tarkvaras.



Joonis 34. 3D mudel korpusest koos riistvaraga kuvatud programmis Solid Edge. Joonis on koostatud autori poolt, kasutades Solid Edge Academic tarkvara.



Nr	Nimetus	Kogus
1	Arduino ja ekraan	1
2	Liugur	1
3	Alumine korpuse osa	1
4	Ülemine korpuse osa	1
5	Patareipesa	1
6	Nupp	2
7	LED (roheline)	1
8	LED (punane)	1
9	LED (sinine)	1
10	Lüliti	2

Joonis 35. Komponentide nimetused, kogused ja asukohad mudelil. Joonis on koostatud autori poolt, kasutades Solid Edge Academic tarkvara.

Jooniselt 35 on näha, kus enamik riistvara korpusel asub. Nüüd, kui disain on loodud, alustatakse 3D printimisega.

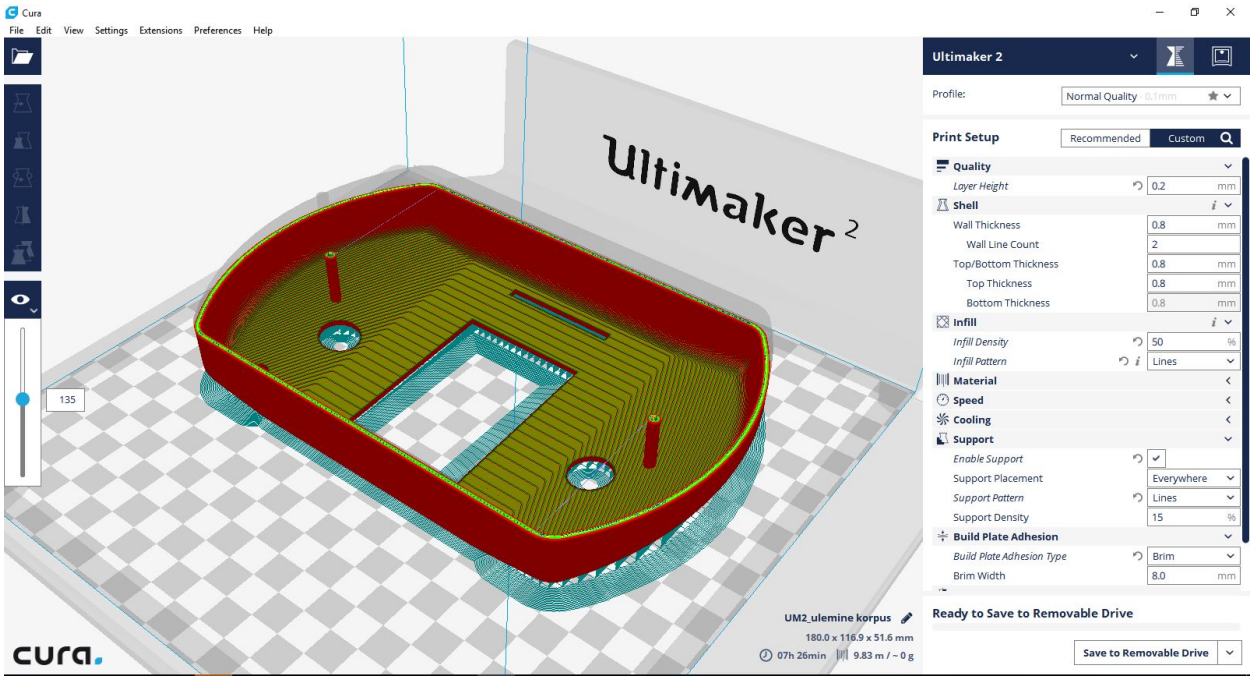
4.2. 3D printimine

3D printimine on protsess kolmemõõtmelise objekti loomiseks kiht kihilt, kus materjalikihid on vormistatud arvuti kontrolli all. Objektid võivad olla erineva kuju või geomeetriaga. Need on toodetud, kasutades digitaalse mudeli andmeid loodud 3D mudelilt. [27]

Printimiseks on vaja autoril väljastada projekt *.stl* failivormingus. STL ehk stereolitograafia on 3D Systemsi loodud failivorming, et kasutada stereolitograafia raalprojekteerimise programmides. Seda failivormingut toetavad paljud tarkvarapaketid ja see on laialt kasutatud

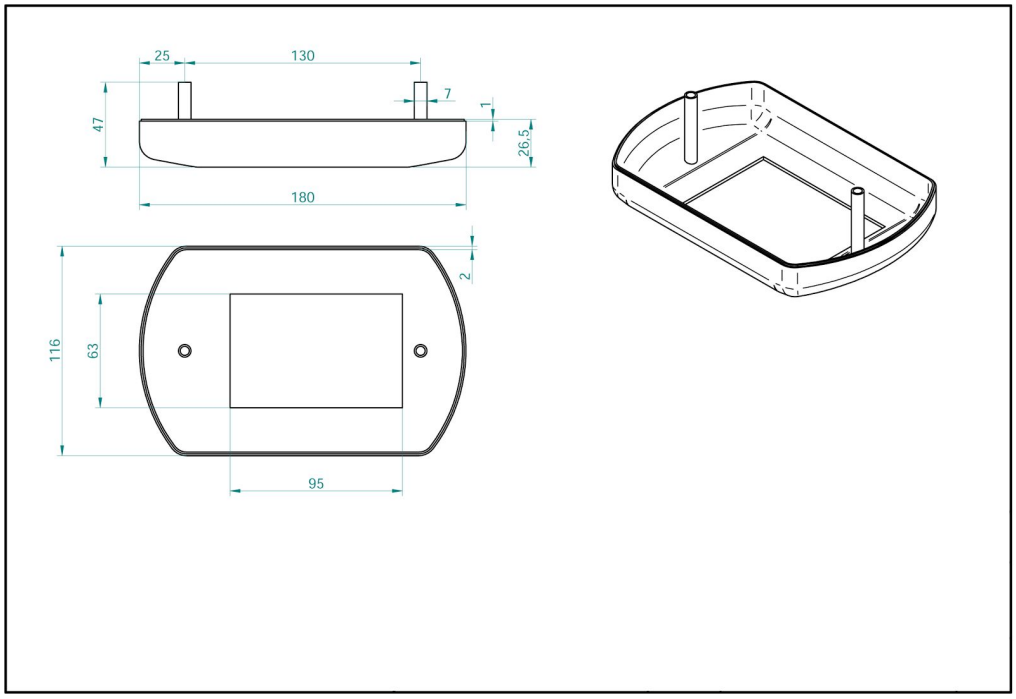
tööstusdisaini prototüüpide loomises [28].

3D printimise tööpõhimõtteid on mitmeid, kui käesolevas lõputöös kasutatakse 3D printimisel *Fuse Filament Fabrication* tehnoloogiat. See tehnoloogia põhineb ideel suruda materjal läbi väikese kõrgel temperatuuril oleva avause. 3D objektide kiht kihilt ehitamiseks liigutatakse samal ajal väga täpselt seda avaust. [29] 3D printeritest kasutati antud lõputöös *Ultimaker* printerit. *Ultimaker* tarkvara on ka näha joonisel 36.

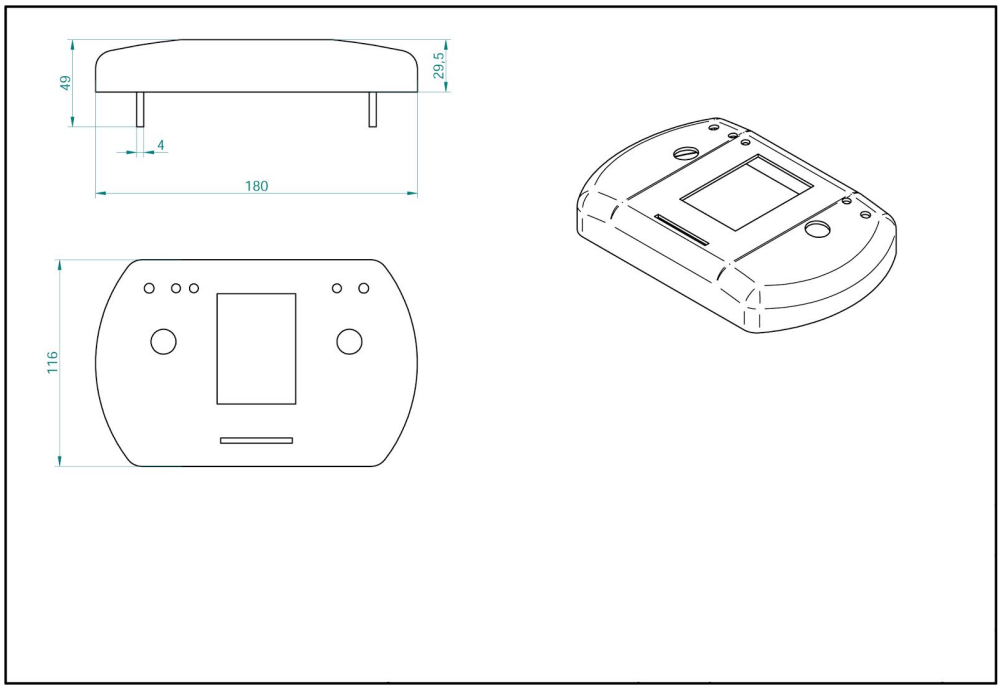


Joonis 36. Korpuse ülemise osa printimine kasutades *Ultimaker* tarkvara [25].

Joonisel 36 on näidatud, kuidas prinditakse korpuse ülemist osa. Korpuse ülemist osa koos mõõtmetega on näha joonisel 38 ja alumist osa joonisel 37.



Joonis 37. Korpuse alumine osa koos mõõtmetega. Joonis on koostatud autori poolt, kasutades Solid Edge Academic tarkvara.



Joonis 38. Korpuse ülemine osa koos mõõtmetega. Joonis on koostatud autori poolt, kasutades Solid Edge Academic tarkvara.

Mudel prinditakse välja kahel erineval korral. Esiteks prinditakse korpuse ülemine osa ja teiseks prinditakse korpuse alumine osa. Korpuste ühendamiseks tegi autor kaks ühenduspulka. Nende abil saab mõlemad korpused omavahel kinnitada. Esmast välja prinditud prototüüpi on näha joonisel 39.

Joonis 39. Esmane prototüüp valminud mängust. Vasakul pool on mängu pealmine vaade ja paremal tagumine vaade.

Esmase prototüübi printimiseks läks 12 tundi aega. Korpuse ülemise osa printimiseks läks 6 tundi ja alumise osa printimiseks 6 tundi. Lõpuks on valminud esmane prototüüp loodud mängust.

Kokkuvõte

Käesoleva bakalaureusetöö põhiliseks eesmärgiks oli luua hariv ja lõbus mäng Arduino LCD ekraanile, mis innustaks inimesi rohkem tegelema tehnika ja programmeerimisega. Autor ei leidnud väga palju õpetusi, mis räägiksid sellest eesti keeles ning seetõttu otsustaski võtta selle teema ise käsile. Autoril ei olnud varasemat kokkupuudet Arduinoga, kuid valis selle teema, kuna tehnoloogia ja programmeerimine olid alati huvi pakkunud. See bakalaureuse lõputöö teema võimaldab just tegeleda mõlemaga samal ajal ning see on ka üks põhjustest, miks käesolev lõputöö sai valitud.

Töö jagunes neljaks osaks. Esimeses osas kirjutati tehtavast mängust, selle semantikast ja reeglitest. Lugejale toodi välja ka pilte mängust, et tekiks parem arusaam, mida hakatakse looma. Peatüki lõpus kirjutas autor üles kõik funktsionaalsed ja mittefunktsionaalsed nõuded tehtava mängu kohta. Seega leidis lahenduse sissejuhatuses loodud eesmärkide loetelus kolmas punkt.

Teises peatükis tõi autor välja hinnatabeli, kus oli näidatud kõik lõputöös kasutatava riistvara hinnad Eestist ja Hiinast tellides. Jõuti järelduseni, et Hiinast riistvara tellides tuleks mängu loomine suhteliselt odav. Seega sai ka kuues eesmärk lahendatud, milleks oli kulude kokkuhoid mängu loomisel. Edasi kirjutas autor pikemalt kõigist vajalikest komponentidest.

Töö kolmandas osas kirjutas autor tähtsamatest protsessidest, mida kasutati mängu loomisel. Sellega sai lahendatud sissejuhatuses seatud neljas eesmärk, milleks oli õppematerjali loomine. Tervet mängu koodi autor aga ei seletanud, sest see on kättesaadav Bitbucket repositooriumis, mida saab leida lõputöö lisades.

Lõputöö neljandas osas kirjutas autor korpuse loomisest, kuna see oli sissejuhatuses seatud viies eesmärk. Autor kirjeldas, miks ja kuidas ta tegi sellise disaini. Seejuures mainiti ka erinevaid tarkvarasid 3D mudeli loomiseks. Kirjutati ülevaatlikult 3D printimisest ja kuidas valmis korpus mängu jaoks. Esimene eesmärk oli luua interaktiivne mäng Arduino LCD peale ja lõpuks näidati

ka valminud prototüüpi sellest mängust.

Abstract

The aim of this bachelor thesis was to create an educating and fun game on Arduino LCD which would encourage people to be more involved in engineering and programming. Author did not find many teachings about this subject in estonian and therefore decided to take this matter in his own hands. The author had no previous experience with Arduino, but chose this topic because technology and programming always interested him. This bachelor thesis enabled to deal with both technology and programming at the same time, which was one of the reasons why this thesis was chosen.

The content of this thesis was divided into four chapters. The first chapter introduced the game and brought out its requirements. The author pointed out some pictures of the game, to give a better understanding of what will be created. The chapter concluded with all of the functional and non-functional requirements for the game.

The third chapter started with the game creation process and taught the reader how to make a similar game. The author did not explain the entire game code because it is available in the Bitbucket repository that can be found in the Extras chapter.

The fourth chapter wrote about 3D modeling and 3D printing. The author describes how and why he made such a design. Author also mentioned the different softwares that can be used to create a 3D model and wrote briefly about 3D printing. Finally the fourth chapter shows the completed prototype of the game.

Kasutatud kirjandus

1. Arduino arendusplaadid, http://www.ittgroup.ee/en/47-arduino-boards (viimati vaadatud 10.03.2017).
2. Microcontroller, https://en.wikipedia.org/wiki/Microcontroller (viimati vaadatud 20.12.2016).
3. Overview, https://www.arduino.cc/en/Main/ArduinoBoardUno (viimati vaadatud 20.12.2016).
4. Overview, http://www.geeetech.com/wiki/index.php/Arduino_Uno (viimati vaadatud 20.12.2016)
5. Today's most popular operating systems, http://www.zdnet.com/article/todays-most-popular-operating-systems/ (viimati vaadatud 20.02.2017).
6. Arduino/Genuino Uno Board Anatomy, https://www.arduino.cc/en/Guide/BoardAnatomy (viimati vaadatud 10.04.2017).
7. Arduino 1.8.2, https://www.arduino.cc/en/Main/Software (viimati vaadatud 05.01.2017).
8. What will YOU do with the W?, http://wiring.org.co/ (viimati vaadatud 10.01.2017).
9. Software development, https://en.wikipedia.org/wiki/Arduino#cite_note-47 (viimati vaadatud 10.01.2017).
10. Bare Minimum code needed, https://www.arduino.cc/en/Tutorial/BareMinimum (viimati vaadatud 10.01.2017).
11. setup(), https://www.arduino.cc/en/Reference/Setup (viimati vaadatud 10.01.2017).
12. loop(), https://www.arduino.cc/en/Reference/Loop (viimati vaadatud 10.01.2017).
13. Product Description, http://alexnld.com/product/5pcs-2-4-inch-tft-lcd-shield-touch-board-display-module-for-arduino-uno/ (viimati vaadatud 15.01.2017).
14. 2.4 inch TFT touch Screen LCD Arduino Shield, http://artofcircuits.com/product/2-4-inch-tft-touch-lcd-module-lcd-screen-module-for-ard

uino (viimati vaadatud 15.01.2017).
15. What is TFT-LCD? Definition of TFT-LCD, https://www.lifewire.com/what-is-tft-lcd-578664 (viimati vaadatud 05.12.2016).
16. Game controller, https://en.wikipedia.org/wiki/Game_controller (10.01.2017).
17. Pushbutton, https://www.arduino.cc/en/tutorial/pushbutton (13.01.2017).
18. Resistor, http://whatis.techtarget.com/definition/resistor (15.01.2017).
19. Secure Digital, https://et.wikipedia.org/wiki/Secure_Digital (viimati vaadatud 20.01.2017).
20. Light-emitting diode (LED), http://whatis.techtarget.com/definition/light-emitting-diode-LED (viimati vaadatud 22.01.2017).
21. Library, http://searchsqlserver.techtarget.com/definition/library (viimati vaadatud 24.01.2017).
22. Define, https://www.arduino.cc/en/reference/define (viimati vaadatud 14.02.2017).
23. Writing a Library for Arduino, https://www.arduino.cc/en/Hacking/libraryTutorial (viimati vaadatud 18.03.2017).
24. Circuit, https://www.arduino.cc/en/Tutorial/AnalogReadSerial (viimati vaadatud 14.04.2017).
25. 10 aastat hobi – 3D printimist, http://etu.ut.ee/2017/3d-printimine/ (viimati vaadatud 05.05.2017).
26. Raalprojekteerimine, https://et.wikipedia.org/wiki/Raalprojekteerimine (viimati vaadatud 28.04.2017).
27. 3D printing process (Basic Principle), http://www.createitreal.com/index.php/technology/process (viimati vaadatud 29.04.2017).
28. STL (file format), https://en.wikipedia.org/wiki/STL_(file_format)#cite_note-1 (viimati vaadatud 29.04.2017).
29. Mis on 3D printimine?, http://etu.ut.ee/2017/3d-printimine/ (viimati vaadatud 09.05.2017).

Lisad

I. Loodud kosmosemängu lähtekood

Käesolevas lõputöös valminud mängu lähtekood on võimalik leida aadressil https://bitbucket.org/RaunoP/arduino-lcd-kosmose-ralli/overview.

II. Terminid

Viik - kiibikorpusest välja ulatuv metallkontakt, mis on ette nähtud kiibi elektriliseks ühendamiseks trükkskeemi või kiibipesaga.

III. Käesolevas lõputöös kasutatava riistvara hinnad Eestis ja Hiinas

Tabel 4. Lõputöös kasutatava riistvara hind Eestis ja Hiinas koos viidetega.

Riistvara	**Hind Eestis**	**Hind Hiinas**
Arduino/Genuino Uno	18,00 EUR [http://www.ittgroup.ee/en/arduino-boards/85-arduino-uno.html?search_query=Uno&results=38]	14,87 EUR [http://www.ebay.com/itm/Original-Official-Genuino-Uno-USB-Microcontroller-Rev-3-ATMega328P-Arduino-/191979075181?hash=item2cb2d8366d:g:ihkAAOSwbsBXmfyy]
2,4 tolline vajutustundlik TFT LCD ekraan	18,00 EUR [http://www.ittgroup.ee/en/arduino-shields/960-8-touch-lcd-shield-for-arduino.html]	4,49 EUR [http://www.ebay.com/itm/2-4-TFT-LCD-Display-Shield-Touch-Panel-ILI9341-240X320-for-Arduino-UNO-MEGA-/192163706724]
2 nuppu	2,40 EUR [https://www.oomipood.ee/product/pbs_33b_bl_nupp_off_on_18mm_sinine_2a_250v?q=nupp&page=2]	3,22 EUR [http://www.ebay.com/itm/50pcs-6-6-7mm-Tactile-Push-Button-Switch-4pin-Tact-Switch-4P-DIP-for-Arduino-/162212052435?hash=item25c4977dd3:g:8X0AAOSwYIxX4r6b]
Liugur	13,90 EUR [https://www.oomipood.ee/product/650_0110_liugpotentsiomeeter_littlebits_slide_dimmer?q=liug&category_id=414]	2,00 EUR [http://www.ebay.com/itm/Electronic-Block-10K-Sliding-Slider-Potentiometer-Module-for-Arduino-New-GK-/351670018046?hash=item51e12a9ffe:g:QlUAAOSwv9hW2mLP]
2 lülitit	2,00 EUR [https://www.oomipood.ee/en/product/smts102_2a1_tumbler_on_on_125v_3a_ts_01?q=l%C3%BCliti&page=4]	1,07 EUR [http://www.ebay.com/itm/5-Pcs-AC-ON-OFF-SPDT-2-Position-Latching-Toggle-Switch-LW-/182050318772?hash=item2a630b69b4:g:aP8AAOSwr7ZW4cL2]

Micro-SD kaart	5,99 EUR [https://www.photopoint.ee/salvestusmeedia/4492-silicon-power-malukaart-microsdhc-4gb-class-4]	1,56 EUR [http://www.ebay.com/itm/128MB-Micro-SD-TF-Memory-Card-For-Samsung-Galaxy-S5-S4-S3-Note-4-3-2-Android-LW-/172366666937?hash=item2821da9cb9:g:7IsAAOSwpLNX9vou]
Micro-USB juhe	2,40 EUR [http://www.ittgroup.ee/en/wires/214-usb-cable-a-microb.html?search_query=micro+usb+cable&results=14]	0,72 EUR [http://www.ebay.com/itm/Spiral-Coiled-USB-2-0-A-Male-to-Micro-USB-B-5Pin-Adaptor-Spring-Cable-Cord-/121362056266?hash=item1c41be044a:g:Gj0AAOSwsLtVfTy2]
Ühendusjuhtmed	3,00 EUR [http://www.ittgroup.ee/en/wires/806-extension-cable-female-female-20-cm-40-pc.html] + 3,00 EUR [http://www.ittgroup.ee/en/wires/805-extension-cable-male-female-20-cm-40-pc.html]	2,17 EUR [http://www.ebay.com/itm/120Pcs-Good-Male-to-Female-Dupont-Wire-Jumper-Cable-for-Arduino-Breadboard-New-/162255494591?var=&hash=item25c72e5dbf:m:moC0vqmne1vNBmY7wQJfIpQ]
Maketeerimislaud	5,00 EUR [https://www.oomipood.ee/product/pps0270_maketeerimislaud_84_46mm_270_punkti?q=maketeerimislaud]	0,67 EUR [http://www.ebay.com/itm/Useful-170-Tie-points-Mini-Solderless-Prototype-Breadboard-For-Arduino-Shield-/121789303105?hash=item1c5b354941:g:0vUAAOSwYHxWINap]
Takistid	0,30 EUR [https://www.oomipood.ee/product/search?q=resistor]	2,31 EUR [http://www.ebay.com/itm/600PCS-Assorted-Resistors-30-VALUES-1-4W-Metal-Film-Kit-Set-Pack-Arduino-PI-/262935451781?hash=item3d382c9885:g:ZfMAAOSwBahVSH49]
Patareipesa	3,50 EUR [https://www.oomipood.ee/pr	2,25 EUR [http://www.ebay.com/itm/Bl

	oduct/48170_gby_patareipesa_6_aa_korvuti_suvistatav_pesa_avatava_kaanega?q=patareipesa&page=1]	ack-AA-Batteries-Holder-Box-Case-Making-Battery-Pack-for-Arduino-monolithic-/121087294434?hash=item1c315d7be2:g:F-cAAMXQCZ1TcenY]
3 valgusdioodi	1,80 EUR [http://www.ittgroup.ee/en/displays-and-indicators/798-led-5-mm-rgb.html?search_query=led&results=91]	1,46 EUR [http://www.ebay.com/itm/10pcs-Ultra-Bright-5mm-4-pin-RGB-Diffused-Common-Anode-LED-for-Arduino-/252093190584?hash=item3ab1ecd1b8:g:U3YAAOSwAYtWG7dw]

IV. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, **Rauno Paluvee**,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose
   Arduino LCD kasutamine kosmosemängu näitel, mille juhendajad on Alo Peets, Anne Villems ja Taavi Duvin,
   1.1. reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace´i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, **11.05.2017**