TARTU ÜLIKOOL

Arvutiteaduse instituut

Informaatika õppekava

Alo Vals

Arduino LCD Lauatennis

Bakalaureusetöö (9 EAP)

Juhendajad: Alo Peets, Anne Villems, Taavi Duvin

Tartu 2017

Arduino LCD Lauatennis

Lühikokkuvõte:

Käesoleva bakalaureusetöö eesmärgiks on kirjeldada Arduino LCD Lauatennise mängu programmeerimist. Töö esimeses peatükis antakse ülevaade loodavast mängust ning sarnastest riist- ja tarkvaralistest lahendustest. Teises peatükis kirjeldatakse tööks vajaliku detailsusega riistvara komponente. Kolmandas peatükis kirjeldatakse vajalikke ettevalmistusi. Neljandas peatükis antakse ülevaade vajalikest programmeerimisvõtetest ning tutvustatakse tulemust.

Võtmesõnad:

Arduino Mega, LCD, mäng, lauatennis, juhtkang, juhtmevaba, Bluetooth

CERCS: P175 Informaatika, süsteemiteooria

Arduino LCD Table Tennis

Abstract:

The main purpose of this bachelor thesis is to describe how to program Arduino LCD Table Tennis. The first chapter of the thesis gives an overview of the game and also describes similar projects from other authors. Hardware components required for the programming of this thesis project will be described in the second chapter. The third chapter contains of necessary preparations. In the last chapter, the main attention is to give an overview of the appropriate programming techniques and to present the final outcome.

Keywords:

Arduino Mega, LCD, game, table tennis, joystick, wireless, Bluetooth

CERCS: P175 Informatics, systems theory

Sisukord

Sissejuhatus

Eesti Rakendusuuringute Keskuse CentAR poolt läbiviidud uuringust selgus, et IKT (info- ja kommunikatsioonitehnoloogia) kõrghariduse õpingute katkestajate hulgas oli 40. protsendil väidetavaks põhjuseks vale eriala valik. Vale eriala valiku tingis vähene või puuduv varasem kokkupuude erialaste praktiliste oskustega. [1]

Sarnasele järeldusele jõudis ka Kadri Mardo oma magistritöös. Tema leidis, et ülikoolieelne kokkupuude valitava eriala valdkonnaga on oluline. Tudengid, kellel on olemas varasem kogemus, suudavad alates esimesest semestrist paremini omandada uusi teadmisi ja nende jaoks ei ole õppimine ületamatu tegevus. Lisaks on tudengid oma valikus kindlamad, kuna huvi eriala vastu on sütitatud õppuris juba varem ja raskustesse sattumisel ei hääbu algne huvi nii lihtsasti, mis omakorda vähendab tudengite väljalangevust loodus- ja täppisteaduste valdkonna erialadelt. [2]

Loodus- ja täppisteaduste valdkonnast väljalangevuse vähendamiseks tuleb pöörata õpilastele rohkem tähelepanu. See tähendab, et valdkonna alaste kogemuste pakkumiseks tuleb võimaldada koolinoortele uute ja põnevate lahenduste realiseerimist. Käesolevas töös kasutatakse Arduino arendusplaati ja sellega ühilduvat värvilist vajutustundlikku ekraani, mille abil on võimalik visualiseerida virtuaalset maailma. Noortes peaks tekitama selline lahendus huvi, kuna enamasti viidavadki koolinoored arvuti- ja telefoni ekraane vaadates ja mitte lihtsalt vaadates vaid ekraanile kuvatavat mängumaailma jälgides. Arduino arendusplaadi ja ekraani koostööl on võimalik mängu loomine ja selle kuvamine. Probleemiks on aga eestikeelsete juhendite ja õppematerjalide puudumine, et rakendada Arduino arendusplaadiga ühilduvat värvilist vajutustundlikku ekraani.

Käesoleva bakalaureusetöö eesmärgiks on Arduino LCD Lauatennise mängu loomine olemasolevate ingliskeelsete juhendite ja projektide abil ning arendustöö põhjal luua eestikeelne juhend värvilise vajutustundliku ekraani kasutamiseks koos Arduino arendusplaadiga. Juhendi järgimisel suudab koolinoor iseseisvalt rakendada nimetatud ekraani sarnase mängu loomiseks. Eesmärgiks on juhendi abil pakkuda noortele

kokkupuudet loodus- ja täppisteaduste valdkonnaga ja seeläbi vähendada väljalangevust IKT erialadelt.

Lõputöö on liigendatud neljaks peatükiks. Esimeses peatükis antakse ülevaade Arduino Lauatennise mängust ning mängu loomiseks vajaminevast riistvarast. Teises peatükis kirjeldatakse detailselt riistvara komponente ja tuuakse välja olulised kohad mida nende valikul silmas pidada. Kolmandas peatükis tutvustatakse Arduino arenduskeskkonda ning kirjeldatakse kuidas teha vajalikud riistvara eelseadistused programmeerimiseks ja kirjeldatakse komponentide elektrilist ühendamist. Neljandas peatükis kirjeldatakse tarkvara ja tähtsamaid võtteid mängu programmeerimiseks.

1. Arduino Lauatennis

Järgnevas peatükis teostatakse töö üldine analüüs. Ühtlasi tutvustatakse lauatennise mängu, kirjeldatakse, millised on mängule ja riistvarale püstitatud nõuded ning põhjendatakse riistvara komponentide valikut.

1.1 Ajalugu – Ping Pong või Lauatennis

Käesoleva töö autor otsustas nimetada mängu eestipäraselt Lauatenniseks, sest arendatavas mängus kasutatakse laenatud elemente ja reegleid reaalses elus mängitavast Lauatennisest. Siinkohal võib tekkida küsimus: „Kas Lauatennis ja Ping Pong tähistavad ühte ja sama mängu?“. Järgnevalt tehakse ülevaade erinevate nimede kujunemise ajaloost.

Nime lahknemine pärineb 1901. aastast, kui nimi Ping Pong registreeriti kaubamärgina briti töösturi John Jaquesi poolt. Peatselt müüs aga perefirma J. Jaques & Son kaubamärgi ettevõttele Parker Brothers. Kuna kaubamärk oli seaduslikult fikseeritud, siis teised tootjad ei saanud enam sama nime kasutada ja sarnastele mängudele viidati nimega Lauatennis. [3]

Seega ei tasu sattuda segadusse erinevatest nimedest, kuna Ping Pong ja Lauatennis tähistavad tegelikult ühte ja sama mängu. Järgnevalt kirjeldatakse, kuidas mängitakse lauatennist tegelikus elus ja kuidas võiks mängimine välja näha seadmel mängides.

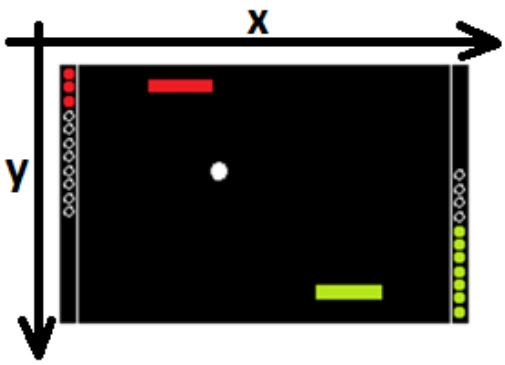
1.2 Reaalse mängu teisendus virtuaalseks

Lauatennis on sportlik ajaviide kahele inimesele, mida mängitakse ristkülikukujulisel laual. Mänguväljak on jagatud püstise võrguga kaheks võrdseks pooleks. Mängijatel on käes reket, millega servitakse ja lüüakse palli vastase poolele. Mängu sisuks on löökide vahetus ja mängijate ülesandeks on teha osavaid ja kavalaid lööke selliselt, et vastane eksiks: lööks pallist mööda või suunaks palli mängulaualt välja. Mängu sett kestab 11. punktini ja võiduks on vaja 2. punktist vahet ning ühes mängus on 3 setti. [4]

Käesolevas bakalaureusetöös muudeti ja lihtsustati reaalse lauatennise mängu reegleid selliselt, et mäng sobituks kasutatava riistvaraga, millest kirjutatakse täpsemalt peatükis 1.6. Virtuaalses, digitaalse seadme vahendusel mängitavas lauatennises mängivad üksteise vastu samamoodi kaks inimest, et tekitada põnevust vastase inimliku etteaimamatu käitumisega. Seadeldis paikneb reaalses keskkonnas mängijate vahel mingil alusel, selliselt, et mängijatel oleks mugav seadme ekraani jälgida. Alljärgnevalt selgitatakse mängu reegleid.

1.3 Arduino lauatennise reeglistik

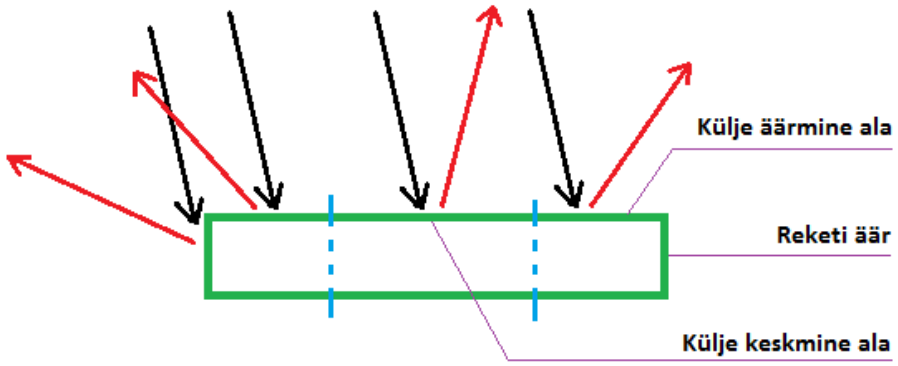
Virtuaalses lauatennises on võistlemas kaks mängijat. Enne mängima asumist on oluline, et mängijad jõuavad kokkuleppele, kas mängitakse settidega või mängitakse ainult üks mäng. Settidega mängu puhul peavad mängijad ise seti võite meeles hoidma. Mäng ei eelista kumbagi mängijat, kuna avaserv loositakse juhusliku suurima arvu põhjal. See tähendab, et mõlemal mängijal on võimalus endale genereerida suvaline arvu lõigust 0-500. Servimisõiguse saab mängija, kes saab genereerides suurema arvu. Mängijad on eristatavad värvi ja asetuse järgi. Mängusiseseid reketeid tähistavad ristkülikud, mis liiguvad x-telje suhtes (vaata joonis 1).



Joonis 1. Koordinaatteljestiku paiknemine mängu suhtes. Visand lauatennise mängust

Reketi juhtimiseks on igal mängijal kasutada pult. Mängijate ülesandeks on takistada palli liikumist mänguplatsilt välja ehk pall ei möödu reketist y-telje suhtes . Palli liikumine on lihtsustatud mänguväljal selliselt, et küljejoontest väljumine on võimatu ehk pall muudab

x-teljelist suunda küljejoonega kokku põrgates. Küljejooneks loetakse skoori ringe ülejäänust mänguplatsist eraldavat joont. Palli liikumissuund sõltub reketi tabamise asukohast (vaata joonis 2). Kui pall lendab vastu reketi äärt, siis muudab pall oluliselt oma liikumissuunda ja kiirust. Kui pall tabab x-teljega paralleelset reketi külje äärmisi alasid, siis muudab pall pisut vähem liikumissuunda ja kiirust. Kui pall tabab aga reketi külje keskmist ala, siis muudab pall ainult y-teljelist liikumissuunda.



Joonis 2. Palli liikumissuuna sõltuvus reketi tabamise asukohast. Musta noolega on märgitud palli liikumine reketi suunas. Punase noolega tähistatakse põrkamisjärgset liikumissuunda. Joonisel on kujutatud vasakpoolse langemisega palli. Autori koostatud

Küljejoonte kõrval asetsevad 11 tühja ringi skoori lugemiseks, mis täidetakse mängija värvusega, kui vastav mängija saavutab punkti. Skoori lugemine algab ühest ja maksimaalne skoor on 11 punkti. Skooride vahe peab olema vähemalt 2 punkti. Mäng loetakse lõppenuks juhul, kui üks mängijatest saavutab 11 punkti ja skooride vahe on vähemalt 2 punkti või, kui mängus pole enam võimalik 2 punktine vahe, st. lõppseis jääb 10:10. Sellisel juhul on tegemist viigiga. Mängu lõppedes on mängijate vaba valik, kas mängida veel või lõpetada.

Järgnevalt selgitatakse süsteemile ja seadmele püstitatud nõudeid.

1.4 Funktsionaalsed nõuded

Selles alapeatükis tuuakse välja funktsionaalsed ehk süsteemisiseseid nõuded. Paremaks mõistmiseks võib mõelda neist nõuetest, kui vastustest küsimustele: „Mida saab kasutaja süsteemis teha?“ ja „Kuidas süsteem kasutaja tegevust toetab ja sellele reageerib?“.

1. **Süsteem pöörab avakuva pealkirja ja ekraan on vajutustundlik -** Süsteem kuvab mängu pealkirja avakuval 2 sekundit algpositsioonis, seejärel pööratakse pealkiri pahupidi ning kuvatakse sellisena 2 sekundit. Seejärel korratakse protsessi. Samal ajal säilib kuva vajutustundlikkus.
2. **Kasutaja saab liikuda avakuvalt reketi juhtimist tutvustavale kuvale ja mängu info kuvale** – Süsteem kuvab avakuva igas nurgas ühe nupu, millest kaks diagonaalse asetusega viivad töö info kuvale ja teised kaks viivad reketi liigutamist tutvustavale kuvale.
3. **Süsteem kuvab avakuva nupud loetavalt mõlemale mängijale** – Mängija A on vasakul pool seadeldist ja mängija B on paremal pool seadeldist. Mängija A poolses ekraani servas kuvatakse 2 nuppu, mis on loetavad mängijale mängijale A. Mängija B poolses ekraani servas kuvatakse 2 nuppu, mis on loetavad mängijale B.
4. **Kasutaja saab liikuda info kuvalt tagasi avakuvale -** Info kuval on nupp, mida vajutades saab kasutaja liikuda tagasi avakuvale.
5. **Kasutaja saab liikuda reketi liigutamist tutvustavalt kuvalt tagasi avakuvale -** Reketi liigutamist tutvustaval kuval on nupp, mida vajutades saab kasutaja liikuda tagasi avakuvale.
6. **Kasutaja saab liikuda avakuva ja avaservi loosimise kuva vahel -** Süsteem tuvastab nupuvajutuse, mille järel suunatakse kasutaja avaservi loosimise kuvale. Tagasi liikumiseks on loosimise kuval nupp.
7. **Süsteem kuvab loosimise alustamiseks nupu ja annab mängijale tagasisidet, et loosinumbrit pole veel genereerima hakatud** - Loosikuval vilgub 3 nulli, kui mängija pole veel vajutanud loosinumbri genereerimise alustamise nuppu.
8. **Süsteem annab tagasisidet käimasoleva loosi arvu genereerimise protsessi kohta ja kuvab lõpetamiseks nupu, kasutaja saab oma suva järgi nuppu vajutades protsessi peatada** – Süsteem kuvab tagasisideks 3 millisekundiliste vahedega uue suvalise arvu. Kasutaja saab peatada numbri genereerimise protsessi vajutades protsessi peatavale nupule.

9. **Süsteem kuvab mängijale genereeritud suvalist arvu** – Kasutaja on vajutanud genereerimist lõpetavale nupule, seejärel peatab süsteem arvu genereerimise protsessi ja kuvab viimast suvalist arvu.
10. **Süsteem kuvab mõlema mängija genereeritud arvud ja annab tagasisidet võidu kohta** - Kui mõlema mängija numbrid on genereeritud ehk mõlemad mängijad on teinud läbi protsessi, mille käigus nad on vajutanud genereerimist alustavat nuppu ning seejärel genereerimist lõpetavat nuppu, siis vilgutab süsteem suuremat numbrit vastava mängija reketi värvuses 4 korda.
11. **Avaservi õigus on mängijal, kelle genereeritud suvalist arvu vilgutati loosikuval** - Pärast loosikuva kuvatakse mänguplats. Mäng algab siis, kui loosikuva võitja servib palli.
12. **Süsteem annab tagasisidet mängijatele servimise õiguse kohta** - Avaservi või punkti saavutamise järel annab süsteem mängijatele tagasisidet, kelle kord on servida. Süsteem vilgutab selle jaoks avaservi õiguse saanud või punkti saavutanud mängija reketi kohal valget joont.
13. **Mängija saab valida palli servimise hetke** – Iga servimine toimub ainult siis, kui servimisõigust omav mängija seda otsustab ja juhtimissüsteemiga seadmele sellest märku annab.
14. **Palli liikumissuund x-telje suhtes servimisel on mõjutatav juhtimissüsteemi abil** - Servimisõigust omav mängija liigutab reketit vasakule ning seejärel annab seadmele servimissoovist märku - toimub serv, mille tulemusena liigub pall reketi suhtes vasakule. Servimisõigust omav mängija liigutab reketit paremale ning seejärel annab seadmele servimissoovist märku - toimub serv ning pall liigub reketi suhtes paremale. Kui servimisõigust omav mängija ei liiguta reketit vasakule ega ka paremale vaid annab ainult servimissoovist märku - toimub serv, mille korral on palli liikumissuund sama nagu oli pallil enne servimist mängulaualt väljudes. Kui mängu avaservi puhul ei liiguta servimisõigust omav mängija reketit vasakule ega ka paremale vaid annab ainult servimissoovist märku - toimub serv, mille korral sõltub palli liikumissuund loosikuval genereeritud suvalise arvu suurusest. Kui arv oli väiksem kui 250, siis pall liigub servides vasakule, vastasel juhul paremale.
15. **Kasutaja saab seadeldist sisse ja välja lülitada** – Seadmel on lüliti, mille abil saab kasutaja seadeldise vooluring ühendada või katkestada. Selle tulemusena lülitatakse seadeldis sisse või välja.

1.5 Mittefunktsionaalsed nõuded

Järgnevalt selgitatakse süsteemi mittefunktsionaalseid nõudeid ehk omadusi, mis määratlevad, milline peab süsteem olema.

1. **Süsteem on kasutatav ilma kõrvalise abita** – Kasutajaliides on intuitiivne ja asjakohaste tähistustega.
2. **Süsteem võimaldab mängu mängimist vähemalt kolm korda järjest** – Mängijad saavad soovi korral mängu lõppedes mängu uuesti mängida vähemalt kolm korda järjest.
3. **Seadeldis on mõeldud kahele mängijale mängimiseks** – Seadeldisel on mängimiseks kaks pulti. Mäng ei ole mängitav üksinda.
4. **Seadeldise ekraaniga osa ja juhtkangidega osad on eraldiseisvad** – Kasutajatel on mugavam jälgida ekraani, mis on mängimise ajal paigal ehk juhtkangid ei ole füüsilises kontaktis ekraani osaga.
5. **Seadeldise ekraani osa energiaallikas on vahetatav** – Energiaallika vahetamine ei nõua kõrvalist abi.
6. **Seadeldise juhtkangiga komponendi energiaallikas on vahetatav** - Energiaallika vahetamine ei nõua kõrvalist abi.
7. **Ühenduse loomine ekraani osa ja juhtpultide vahel ei võta rohkem kui 10 sekundit** – Kasutaja saab kindlasti puldi juhtkangi liigutamisel pärast kümnendat sekundit reketit juhtida.
8. **Seadeldis on võimeline helilise tagasiside andmiseks kasutajale –** Menüü valikute tegemisel ja palli põrkamisel tekitab seadeldis helisignaali. Helisignaali väljalülitamine on võimaldatud.

Mittefunktsionaalsete nõuete kontrollimiseks tuleb katseisikutele mängukonsooliga mängida ning selle põhjal küsitlus teha. Järgnevalt tutvustatakse sarnased teostusi teistelt autoritelt.

1.6 Sarnaste lahenduste ülevaade

Käesolevas alapeatükis tutvustatakse ülevaatlikult teisi sarnaseid riist- ja tarkvaralisi lahendusi. Allpool toodud näidete puhul on kasutatud Arduino platvormi. Seejuures selgitatakse, mille poolest erineb käesoleva bakalaureusetöö lõpplahendus.

Michael Teeuw, Arduino Pong – Projektis kasutatakse arendusplaadina Arduino Unot, mängu kuvamiseks kasutatakse 1,3 tollist Adafruit OLED (Organic Light Emitting Diode) ekraani ning mängusiseste reketite juhtimiseks on pöördpotentsiomeetrid. Nimetatud projekt erineb arendusplaadi, ekraani ja juhtimissüsteemi valiku poolest ning mäng on oluliselt lihtsama ülesehitusega. Puudub vajutustundlikkus, menüü, värvid ning võimalus palli servimiseks. Lisaks sellele väärib mainimist, et seade on ühes tükis, mis on kahe mängijaga mängu puhul halb lahendus, kuna seadme käsitsemisel võib seade oluliselt liikuda ja ekraanil toimuva jälgimine on raskendatud. Välja toodud projekti *loop()* funktsiooni keha lihtsusest on võetud eeskuju käesoleva lõputöö programmeerimisel. Projekti detailne kirjeldus on leitav aadressilt *http://michaelteeuw.nl/post/87381052117/building-pong.*

Arduino, TFT Pong – Arduino meeskond on välja töötanud juhendi objektide kiireks liigutamiseks ja interaktiivseks juhtimiseks TFT (*Thin Film Transistor*) LCD (*Liquid Crystal Display*) ekraanil. Arendusplaadina kasutatakse Arduino Unot, kuvamiseks on Arduino TFT ekraan ning juhtimine on taaskord lahendatud pöördpotentsiomeetrite abil. Nimetatud projektis ei ole erilist mänguloogikat, tegemist on puhtalt interaktiivsuse ja objektide liigutamise tutvustuseks. Juhend on leitav aadressilt *https://www.arduino.cc/en/Tutorial/TFTPong.*

Tia’s Arduino Pong - Projektis on kasutatud Arduino Nano arendusplaati, 1,8 tollist TFT LCD ekraani ja juhtimiseks analoog juhtkangi moodulit. Nimetatud töö lahendus sarnaneb käesoleva töö teostusega, kuid siiski on tegemist oluliselt lihtsama lahendusega. Puudub menüü, puutetundlikkus, lihtsam juhtimissüsteemi lahendus ning mõeldud ainult ühele mängijale. Lisaks sellele puudub korralik dokumentatsioon. Projekt asub aadressil *http://myarduinoproject.weebly.com/past-projects.html*.

Varasemalt on kasutatud käesolevas töös rakendatud vajutustundlikku ekraani sarnase mängu kuvamiseks, kuid ingliskeelne dokumentatsioon puudub. Mängust on tehtud video, mida on näha aadressilt *https://www.youtube.com/watch?v=NSWKeX1sY0E*.

Käesoleva töö autor on teadlik, et sarnased lahendused eksisteerivad. Kuid eesmärgiks ei olegi millegi uue loomine vaid hariduslikel eesmärkidel terviklahenduse kokkupanek olemasolevatest inglisekeelsetest teostustest ja juhenditest. Arduino LCD Lauatennise nõuete kohaselt tuleb seadme terviklahenduse loomiseks kasutada erinevaid riistvara komponente. Järgnevalt tuuakse välja loodud mängukonsooli riistvaralised komponendid ja selgitatakse nende valikut.

2. Riistvara detailsem kirjeldus

Järgnevalt kirjeldatakse loodava mängukonsooli jaoks vajaminevaid komponente ja tuuakse välja olulisimad asjad, mida silmas pidada. Põhjalikum tutvumine komponentidega on vajalik, et mõista paremini vooluringide koostamist ja riistvara programmeerimist.

2.1 Arduino platvorm

Arduino on avatud lähtekoodiga elektroonika platvorm, mis põhineb lihtsasti kasutataval riist- ja tarkvaral. Nimetatud platvorm on suunatud riistvaralähedaste lahenduste loomiseks, milleks on ka mängukonsooli prototüübi loomine. Arduino arendusplaadil on mikrokontroller, mida on võimalik juhtida läbi Arduino IDE (Integrated Development Environment) kirjutatud programmide. Programmi põhjal suudab arendusplaat sisendsignaali lugeda ja sellele ka vastata. Seega võib Arduino arendusplaadil olevast mikrokontrollerist mõtelda kui ajust, mis juhib loodava seadme teisi osasid. [5][6]

Järgnevalt selgitatakse Arduino arendusplaadi valikut lähtudes kolmest parameetrist: jadaportide arv, pesade arv teiste komponentide ühendamiseks ja mäluruumi suurus programmi mahutamiseks. Käesolevas töös rakendatakse juhtmevaba juhtimissüsteemi ning selle teostamiseks on vajalik mitme jadapordi olemasolu. Pesade arv on oluline sellepärast, et kõik vajalikud lisadetailid saaksid vajalikud ühendused arendusplaadiga. Määrav tegur on ka mäluruumi suurus, sest mängu loomiseks tuleb kirjutada programmi palju koodi. Tabelis 1 on välja toodud kahe populaarseima Arduino arendusplaadi Uno ja Mega võrdlus [**7**].

Tabel 1. Arduino Uno ja Mega võrdlus [8]

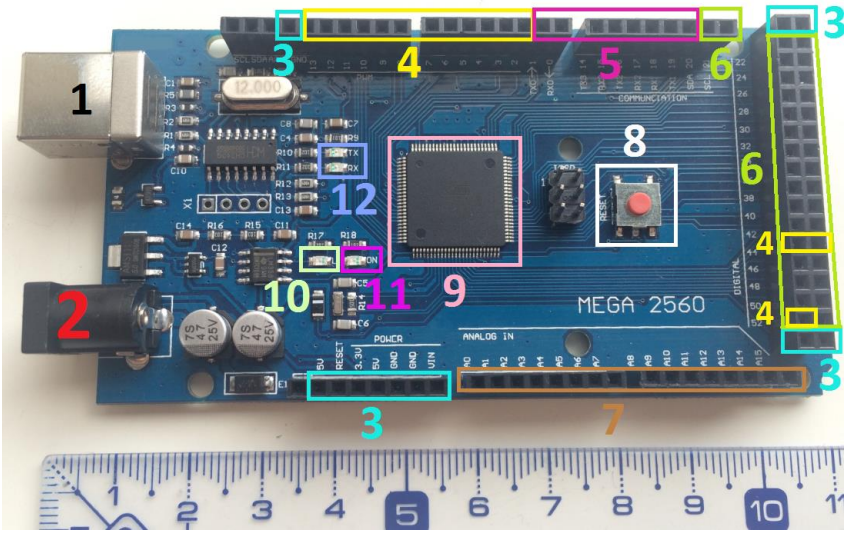
Arendusplaat	**Aruino Uno**	**Arduino Mega**
Mikrokontroller	ATmega328P	ATmega2560
Taktsagedus	16 MHz	16 MHz
Tööpinge / Sisendpinge	5 V / 7-12 V	5 V / 7-12 V
Digitaal sisend/väljund pesade arv	14	54
Analoog sisend pesade arv	6	16
Pulsilaiusmodulatsiooni pesade arv (PWM)	6	15
Jadaportide arv (paar RX, TX)	1	4
Pesade arv kokku	20	70
Välkmälu	32 kB, millest 0.5 kB kasutab alglaadur	256 kB, millest 8 kB kasutab alglaadur
Staatiline muutmälu	2 kB	8 kB
USB pesa tüüp	B	B

Lähtudes eelseatud parameetritest ja tabelis 1 välja toodud andmetest, siis käesoleva töö tegemiseks tuleks valida Arduino Mega arendusplaat. Järgmisena kirjeldatakse lähemalt Arduino Mega arendusplaati.

2.1.1 Arendusplaat Arduino Mega

Kõige olulisemaks komponendiks mängukonsoolis on Arduino Mega arendusplaat (vaata joonis 3), mis suudab signaalide abil suhelda nii teiste seadmetega, kui ka väliskeskkonnaga. Siinkohal on oluline mõista digitaal- ja analoogsignaalide erinevust.

Digitaalsignaalis eksisteerivad ainult kaks pingeväärtust, 1 (*HIGH*) ja 0 (*LOW*), mis üksteisele järgnedes moodustavad bitijada. Sellist bitijada nimetataksegi digitaalsignaaliks. Näiteks enne nupu vajutamist on bitijadas kõik väärtused 0-id, kuid nupu vajutamisel edastatakse bitijadasse pingeväärtus 1, mille tulemusena signaali vastuvõtja saab teada nupu vajutusest. Analoogsignaalis võib pingeväärtusi olla aga palju ning need võivad ajas sujuvalt muutuda. Näiteks juhtkangi liigutamisel küljelt küljele võivad väärtused signaalis varieeruda nullist kuni 1023ni. Seega on analoogsignaal kasulik väliskeskkonna muutlikkuse edastamiseks. [9]



Joonis 3. Arduino Mega arendusplaat ülalt vaadatuna. Joonisel on nummerdatud ja tähistatud Mega tähtsamad detailid arendaja jaoks

Järgnevalt seletatakse lahti joonisel 3 olev nummerdus [10];

1. USB (*Universal Serial Bus*) pesa, mida kasutatakse programmi üles laadimiseks ja arendusplaadi elektrivooluga toitmiseks.

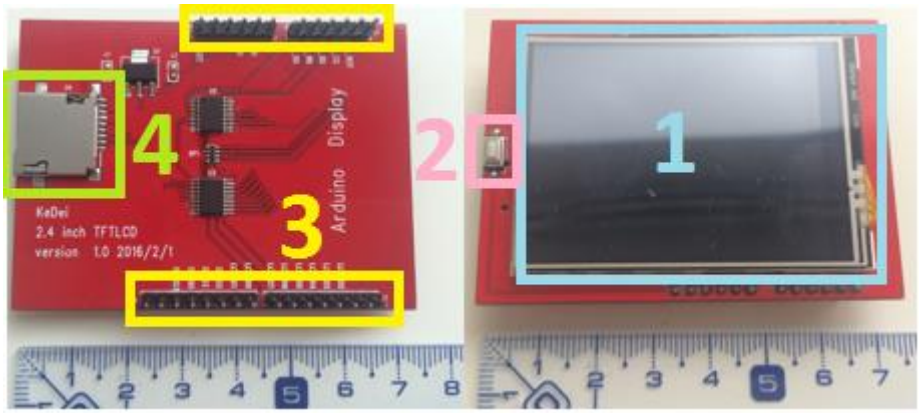
2. Toitepesa elektrivoolu tagamiseks. Sisendpinge on lubatud vahemikus 7-12 volti (edaspidi lühend V).
3. Arduino pesad toite tagamiseks teistele seadme komponentidele või Arduino enda toite tagamiseks. Aruino 5V ja GND (*ground*) ehk maanduse pesadesse võib ühendada stabiilselt 5V pinget pakkuva vooluallika. Arduino VIN ja GND pesadesse võib ühendada vooluallika, mis soovituslikult pakub pinget vahemikus 7-12V.
4. Pulsilaiusmodulatsiooni (*Pulse Width Modulation*, lühend PWM) võimekusega pesad, võimalik genereerida analoogsignaali. Selline võimekus on kasulik valgusdioodi (*Light-Emitting Diode*, lühend LED) heleduse muutmisel, audio signaalide genereerimisel, mootorite kiiruse kontrollimiseks jm.
5. Jadapordi pesad. Neid kasutatakse otseseks suhtluseks arendusplaadi ja arvuti või mõne muu seadme vahel. Käesolevas töös juhtpultidega suhtlemiseks läbi Bluetooth mooduli.
6. Digitaalsignaali pesad nii sisendi kui ka väljundi võimalusega. Digitaalsignaali pesade alla lähevad ka numbriga 4 ja 5 tähistatud pesad.
7. Analoogsignaali pesad, mis suudavad vastu võtta analoogsignaali. Vajadusel saab väljastada digitaalsignaali, kuid mitte analoogsignaali.
8. Arendusplaadi lähtestusnupp. Kui nupule puudub ligipääs, siis saab lähtestamist teostada ka Arduino RESET pesa madalaks (*LOW*) viimisel.
9. Peatüki alguses mainitud aju ehk programmeeritav mikrokontroller ATmega2560.
10. Sisseehitatud LED testimiseks, mida saab kontrollida programmis konstandiga *LED_BUILTIN* või pöördudes 13. viigu poole.
11. Sisseehitatud LED, mis näitab toite olemasolu arenduspladil.
12. Sisseehitatud LED-id näitamaks andmevahetuse toimumist või mittetoimumist jadapordi kaudu.

Arendusplaadi ehituses on veel osasid, kuid kuna käesolevas töös ei ole detailsem plaadi tundmine oluline, siis lihtsuse mõttes ei tooda tähistamata osasid välja. Järgmiseks oluliseks komponendiks on ekraan, millele kuvada mängumaailma.

2.2 Vajutustundliku ekraani moodul

Käesolevas töös on kesksel kohal ekraani abil virtuaalse maailma visualiseerimine kasutajale. Kuna ekraanile kuvatavat sisu ei ole palju ja arendatav mäng on minimalistliku väljanägemisega, siis 2,4 tolline vajutustundliku ekraani moodul on igati sobilik ja piisav. Tänu ekraani mooduli suurusele ja vajutustundlikkusele on see kasutatav ka paljudes teistes projektides.

Virtuaalse mängu visualiseerimiseks kasutatakse 2,4 tollist TFT LCD vajutustundliku ekraani moodulit, mis on üheks interaktsiooni viisiks mängu ja kasutaja vahel. Mooduli osad on näha joonisel 4.



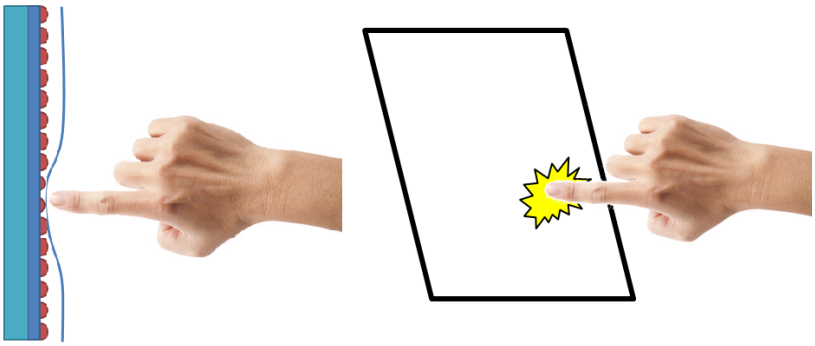
Joonis 4. 2,4 tolline TFT LCD puutetundlik moodul ülalt ja alt vaadatuna.

Joonisel 4 tähistab number 1 LCD ekraani, 2 plaadi lähtestamise nuppu, 3 viike Arduino arendusplaatidega ühendamiseks ja 4 *microSD* kaardi pesa. Mooduli sisendpingeks sobib 5 volti, mis tähendab, et mooduli viigud võib otse suruda Arduino arendusplaadi õigetesse pesadesse. LCD ekraani eraldusvõime on 240x320 pikslit ja ekraan suudab kuvada 262000. erinevat värvitooni [11]. Oluline on teada ekraani draiverit, mille põhjal saab internetist otsida sobiva teegi ekraani kasutamiseks. Konkreetse ekraani draiveriks on IL9341 [12]. Paigaldusest ja esimestest katsetustest kirjutatakse lähemalt peatükis 3.3.

Lugemise käigus võib tekkida segadus, kuna töös nimetatakse ekraani vajutustundlikuks. Järgnevalt selgitatakse Y. Lanceti veebimaterjali põhjal vajutustundliku ja puutetundliku ekraani sarnasusi ja erinevusi [13].

2.2.1 Vajutustundliku ekraani tööpõhimõte

Peale vaadates tunduvad vajutustundlik ja puutetundlik ekraan samasugused olevat, kuid tegelikult ei ole. Erinevate elektrooniliste seadmete puhul kasutatakse peamiselt kahte tüüpi ekraane. Igapäevase kasutuse põhjal on tuntuim mahutuvuslik (*capacitive*) ekraan (vaata joonis 5), mida kasutatakse enamasti nutitelefonides.



Joonis 5. Vasakul pildil on üldistatud kujul resistiivne ehk takistuslik (*resistive*) ekraan, paremale mahutuvuslik (*capacitive*) ekraan [13]

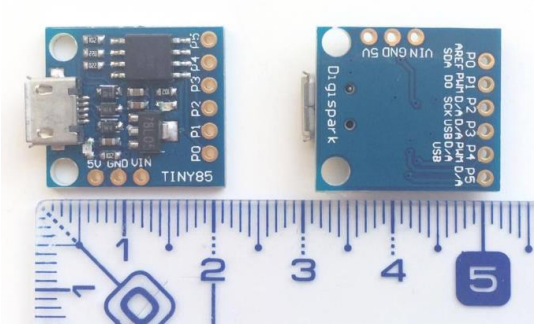
Mahutuvuslik ekraan koosneb LCD paneelist, elektroodidest, elektrit juhtivast kilest ja klaasist. Elektrit juhtiva kile abil on üle ekraani tekitatud ühtlane elektriväli. Kui näpuga puudutatakse ekraani, siis muudetakse ekraani elektrivälja jaotust. Nii saadaksegi teada puudutuse asukoht. Mahutuvusliku ekraani puhul on oluline, et ekraani puudutav ese oleks elektrit juhtiv (näiteks inimese sõrm). Käesolevas töös kasutatakse aga takistuslikku ekraani, mis on lihtsamini toodetav ja seeläbi ka odavam.

Takistuslikku ekraani kasutatakse tööstuslikes seadmetes (näiteks tööstuslikud õmblusmasinad) kui ka igapäevaselt kasutatavates seadmetes (näiteks auto keskkonsooli ekraan). Takistuslik ekraan koosneb LCD paneelist, elektroodidest ja kahest elektrit juhtivast kihist, mille vahel on õhuvahe. Ekraani tööpõhimõte seisneb vajutusel, mis

muudab elektriringi suletuks. Kui näpuga vajutada ekraanile, siis surutakse pealmine painduv ja elektrit juhtiv kiht vastu alumist elektrit juhtivat kihti. Selliselt tekitatakse elektriring ja nii suudetakse lugeda vajutuse asukohta. Vastupidiselt mahutuvuslikule ekraanile ei ole takistusliku ekraani puhul tähtis, millega ekraanile vajutatakse, sest oluline on kahe elektrit juhtiva pinna kokku vajutamine. Siiski on takistusliku ekraani kasutusmugavus väiksem, kuna kasutaja peab füüsiliselt vajutama ekraanile. Mahutuvusliku ekraani puhul piisab juba õrnast puudutusest reageerimiseks. Lisaks sellele ei saa takistusliku ekraaniga registreerida kahte samaaegset puudutust ning ekraani vaatamisnurk on väiksem, kui mahutuvuslikul ekraanil.

2.3 Arendusplaat Digispark

Käesolevas töös loeb Digispark arendusplaat juhtkangi mooduli analoogsignaali, töötleb seda ning seejärel suunab väljundi edasi Bluetooth moodulile. Digispark arendusplaat sarnaneb väga Arduino omaga, kuna ülesanne on mõlemal sama: reageerida sisendsignaalile ja vastavalt programmikoodile muuta see väljundsignaaliks. Lisaks sellele on mõlemal arendusplaadil sama tootja mikrokontroller, mõlemad on programmeeritavad Arduino IDE abil, neil on testimiseks LED, olemas on sisend- ja väljundviigud jne. Väiksemate mõõtude pärast on Digispark ka vähesemate võimalustega (vaata joonis 6). Kuid Digispark on suurepärane arendusplaat väiksemate ülesannete teostamiseks kitsamates tingimustes.



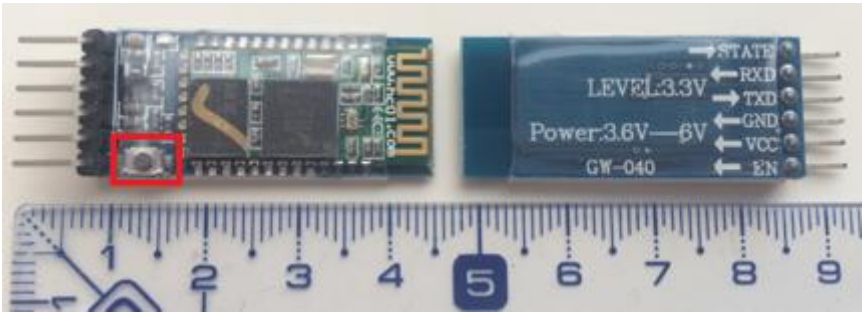
Joonis 6. Arendusplaat Digispark Attiny85 alt ja pealt vaates

Digispark arendusplaadi mikrokontrolleriks on Attiny85, millel on ainult 8 viiku. Kuid tegemist on odava mikrokontrolleriga, tänu millele on ka arendusplaadi üldine hind

soodne. Arendusplaadil on kasutamiseks kuus sisend ja väljund viiku. Silmas tuleb pidada, et kaks viiku, P3 ja P4 on kasutusel USB suhtlusel, mille tõttu võib tarvilik olla nende viikude vabastamine programmi üleslaadimiseks. Digispargil on 8 kB välkmälu, millest 2 kB kasutab alglaadur. Digisparki saab mitmel moel vooluga toita. Läbi USB pistiku võib ühendada vooluallika, mis pakub stabiilselt 5V või ühendades sama vooluallika 5V ja GND klemmiga. VIN ja GND klemmiga võib ühendada vooluallika, mis soovituslikult pakub pinget vahemikus 7-12V. Digispargi seadistamisest ja kasutamisest kirjutatakse peatükis 3.[14] [15]

2.4 Bluetooth moodul HC-05

Juhtmevaba suhtluse loomiseks juhtpultide ja ekraaniga osa vahel kasutatakse laialdaselt levinud Bluetooth tehnoloogiat [16]. Selle rakendamiseks käesolevas töös võetakse kasutusele HC-05 Bluetooth moodulid (vaata joonis 7). Kokkupuude antud riistvara komponendiga on kasulik, kuna Bluetooth mooduli abil saab järgnevates elektroonika projektides rakendada palju huvitavaid ideesid, mis hõlmavad endas arvuteid, nutitelefone või muid Bluetooth valmidusega seadmeid.



Joonis 7. Bluetooth moodul HC-05. Punase kastiga on tähistatud EN viiguga ühendatud nupp, mis on vajalik mooduli seadistamisolekusse viimiseks

HC-05 moodul on eelistatud, kuna moodulit on võimalik seadistada ülemaks (*master*) kui ka alluvaks (*slave*) ehk antud moodulil eksisteerib olek nii seadistamiseks kui ka andmevahetuseks. Välimuselt väga sarnane mudel HC-06 on alluva seadistuseg ja ainult ühe olekuga. Ülemseadistusega moodul suudab leida teisi seadmeid ja on paarumise (*pairing*) algatusvõimeline, samas alluva seadistusega moodul suudab olla ainult nähtav

ülemseadmetele ja paarumissoovi olemasolul sellele vastata. Seega on oluline, et moodulid oleksid seadistatavad, kuna juhtmevaba suhtlus hakkab toimuma kahe mooduli vahel, seda ühe puldi kontekstis. Teise puldi jaoks on teine paar mooduleid. [17]

HC-05 mudel on laialt levinud, tänu millele leidub internetis hulgaliselt materjale ja juhendeid mooduli rakendamiseks projektides. Moodul on hõlpsasti seadistatav ja kasutatav. Seadistamisest kirjutatakse lähemalt 3. peatükis.

2.5 Passiivne sumisti

Sumisteid (*buzzer*) kasutatakse tavaliselt alarmseadmetes (näiteks suitsuandur), olmetehnikas, äratuskellades, sporditabloodes jne. Lähtuvalt keskkonnast, kus sumistit kasutatakse, tuleks valida ka vastav sumisti tüüp. Sumistite valikust kasutatakse käesolevas töös piezoelektrilist (*piezoelectric*) sumistit, mis jaguneb omakorda veel kaheks tüübiks: passiivne ja aktiivne. Mängijatele helilise tagasiside andmiseks kasutatakse käesolevas töös passiivset sumistit (vaata joonis 8). Järgnevalt selgitatakse aktiivse ja passiivse sumisti erinevusi. [18]

Joonis 8. Lisaks sumistite erinevale kasutamisele võib neil vahet teha põhja järgi [19]. Vasakul on välja toodud passiivne sumisti ning paremal aktiivne sumisti

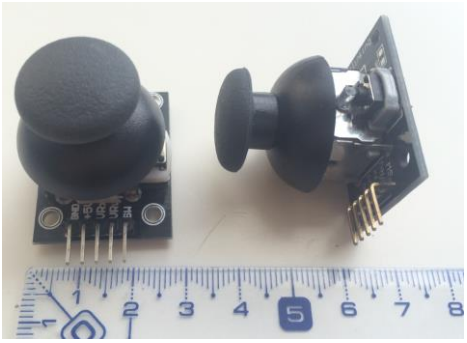
Kahe sumisti erinevus seisneb nende kasutamises. Aktiivse sumisti tööle rakendamiseks tuleb tekitada korrektse suunaga vooluring (sumistile on märgitud „+“, mis tähistab positiivse viiguga sumisti poolt. Sumisti rakendamisest on detailsemalt kirjutatud 3. peatükis) ning sumisti tekitab seejärel heli. Toimides passiivse sumistiga sarnaselt, siis

sumisti vaikib. Aktiivsesse sumistisse on sisse ehitatud võnkuvat elektrilainet tootev võnkering (*oscillating circuit*). Seega rakendades elektrivoolu aktiivsele sumistile hakkab laineallikas tööle, mille tulemusena tekitatab sumisti heli. Passiivsel sumistil aga puudub sisseehitatud laineallikas, tänu millele on passiivne sumisti aktiivsest ka odavam. Passiivse sumisti tööle rakendamiseks tuleb kasutada Arduino arendusplaadi pulsilaiusmodulatsiooni pesa (PWM tähisega plaadil) ja programmisiseselt sumistile saata võnkuvat elektrilainet. [20]

Eelneva põhjal saab paika panna aktiivse ja passiivse sumisti kasutatavuse projektides. Kui soovitakse tekitada seadme töö jooksul ainult ühes toonis helisignaali, siis sobib suurepäraselt aktiivne sumisti. Käesoleva töö puhul on oluline aga edastada erinevaid toone, mis iseloomustavad lauatennise mängus põrkava palli häält.

2.6 Analoog juhtkangi moodul

Peatükis 1.4 püstitatud funktsionaalsetes nõuetes on kirjas, et kasutajal peab olema võimalik nii reketit liigutada kui ka palli servida. Funktsionaalse nõude saab täita mugavalt analoog juhtkangi (*joystick*) mooduli abil (vaata joonis 9). Kasutades juhtkangi x-teljelist liikuvust reketi vasakule ja paremale liigutamiseks ning y-teljelist liikuvust palli servimiseks, saab mängukonsooli juhtimissüsteemi üles ehitada kahe analoog juhtkangi mooduli abil.

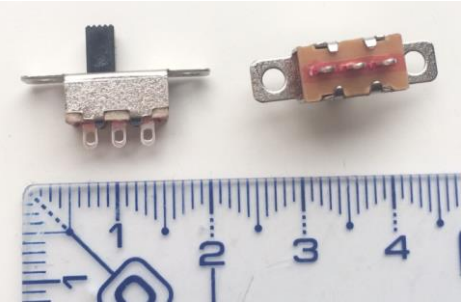


Joonis 9. Analoog juhtkangi moodul

Juhtkangi ehitus on lihtne - kaks potentsiomeetrit (üks x-telje jälgimiseks ja teine y-telje jälgimiseks) on ühendatud juhtkangiga. Potentsiomeeter ise on sujuvalt reguleeritav muuttakisti ja käitub nagu sensor edastades plaadile analoogsignaali. Juhtkangi liigutamisega muudetakse potentsiomeetrite takistust, mille põhjal saadaksegi aru, kuhu kasutaja juhtkangi parasjagu suunab. Juhtkangi kasutamiseks tuleb luua ühendus arendusplaadi analoogsignaali pesadega. [21]

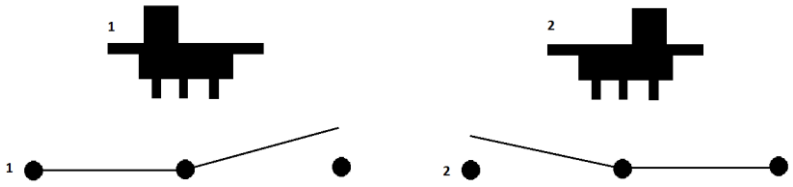
2.7 Liuglüliti

Kasutusmugavuse suurendamiseks tuleks mõtelda ka kiirele ja lihtsale seadme sisse ja välja lülitamisele. Lülituste tegemiseks kasutatakse käesolevas töös väiksemõõdulist liuglülitit (vaata joonis 10).



Joonis 10. Liuglüliti kolme kontaktiga.

Liuglüliti kontaktid kannatavad kuni 12 volti, tänu millele sobib vooluallikaks ka mugavalt kasutatav 9V patarei. Lüliti lülitustüübiks on ON-ON, mille tööpõhimõtte skeem on toodud joonisel 11. [22]



Joonis 11. ON-ON liuglüliti tööpõhimõtet illustreeriv skeem

ON-ON lüliti saab kergesti muuta ON-OFF lülitiks. Ühenduste tegemiseks tuleb kasutada lüliti keskmist ja ühte äärmist kontakti, sest nii muutub liuguri liigutamisel ühendamata kontakti suunas vooluring avatuks. Vooluringi koostamisest kirjutatakse peatükis 3.4.

2.8 Mängukonsooli vooluallikas

Järgnevalt kirjeldatakse Arduino Mega ja selle küljes olevate komponentide vooluallikat, milleks on akupank Colorovo PowerBox Slim 3000 (vaata joonis 12). Tegemist on õhukese ja kerge akupangaga, mis sisaldab endas 3000. milliampertunnise (M*illiamp Hour*, lühend mAh) mahutavusega liitium-polümeer akut.

Joonis 12. Taaslaetav akupank PowerBox Slim 3000.

Akupanga laius on 63 mm, pikkus 122 mm ja paksus 8 mm. Nimetatud akupanka on mugav kasutada Arduino Mega vooluallikana, kuna komponentide mõõtmed on üsna sarnased (Mega trükkplaadi pikkus on 101,52 mm ja laius on 53,3 mm) [*https://www.arduino.cc/en/Main/arduinoBoardMega2560*]. Akupangal on kaks pistikut: USB pistik teiste seadmete varustamiseks toitega (näiteks nutitelefon) ja Mikro-USB pistik akupanga laadimiseks. Nii sisendpingeks kui ka väljundpingeks on 5 volti, mis tähendab, et ilma täiendavaid pingemuundureid kasutamata võib akupanga väljund USB pistikust toita Arduino Megat tema 5V ja GND pesade kaudu või kasutades USB kaablit. [23]

2.9 Puldi toitesüsteem

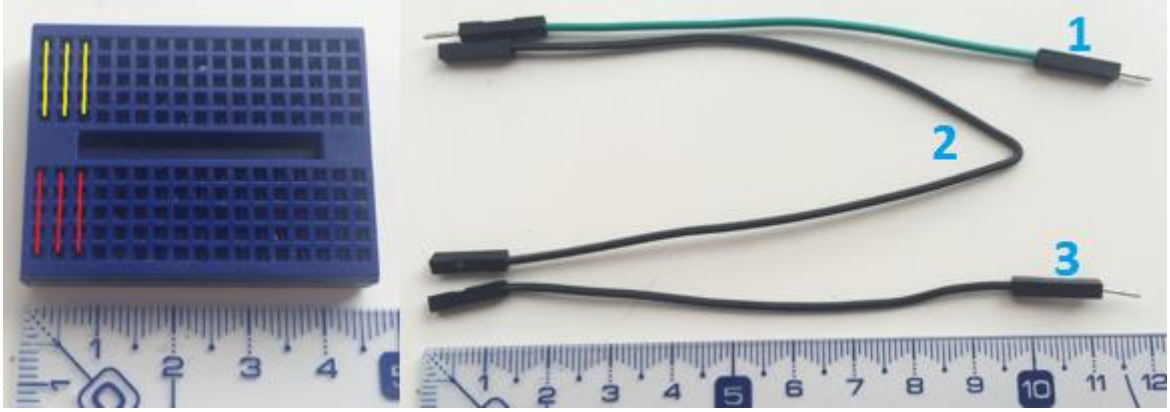
Kasutusmugavuse seisukohast vaadates peab olema ka juhtpultide vooluallikas kergesti vahetatav. Käesolevas töös kasutatakse juhtpuldi vooluallikaks 9 voldist kandilist patareid, mida saab lihtsasti ühendada spetsiaalse ühendusklemmi abil (vaata joonis 13).

Joonis 13. Ühendusklemm 9V patareile ja Panasonic 9V leelispatarei

Patarei on juhtpuldi jaoks mõõtmetelt piisavalt väike: pikkus 48,5 mm, laius 26,5 mm ja paksus 17,5 mm. Leelispatarei mahutavus on 520 milliampertundi ning väljundpingeks on 9 volti [24]. Seega on rangelt keelatud patarei ühendusklemmi juhtmete ühendamine arendusplaadi 5V ja GND viikudega või USB pistikuga. Käesolevas töös tuleb nimetatud patarei ühendusklemmi maandusjuhe (enamasti musta värvi) ühendada Digispark arendusplaadi GND viiguga ja pinge juhe (enamasti punast värvi) arendusplaadi VIN viiguga.

2.10 Maketeerimislaud ja ühendusjuhtmed

Katsetamisjärgus riistvara lahenduse puhul ei ole mõistlik jootmisega kiirustada, kuna üpris kindlasti tuleb varem või hiljem midagi vooluringis ümber tõsta. Järelikult tuleks kasutada ajutist komponentide kinnitamist maketeerimislaua ja spetsiaalsete juhtmetega (vaata joonis 14).



Joonis 14. Maketeerimislaud, millel tähistatud signaali liikumine. Paremal ühendusjuhtmd ja nende tüübid: 1 – isane isane, 2 - emane emane ja 3 – emane isane

Maketeerimislaua kasutamise juures on kõige olulisem selge arusaam signaali liikumisest. Joonisel 14 on tähistatud punaste joontega signaali rajad ühelpool maketeerimislauda ja kollaste joontega signaali rajad teiselpool. Nii eri värvi kui ka sama värvi rajad on üksteisest eraldatud. Järelikult on joonisel 14 välja toodud maketeerimislaual 34 erinevat rada.

Käesolevas peatükis kirjeldati piisava detailsusega seadme loomiseks vajaminevat riistvara ja selgitati ülevaatlikult nende tööpõhimõtteid. Lisaks sellele kirjeldati olulisi tähelepanekuid, et juhendi järgija ei kahjustaks riistvara. Järgmises peatükis kirjutatakse Arduino IDE-st ja selle kasutamisest ning tehakse vajalikud riist- ja tarkvara seadistused.

3. Ettevalmistused

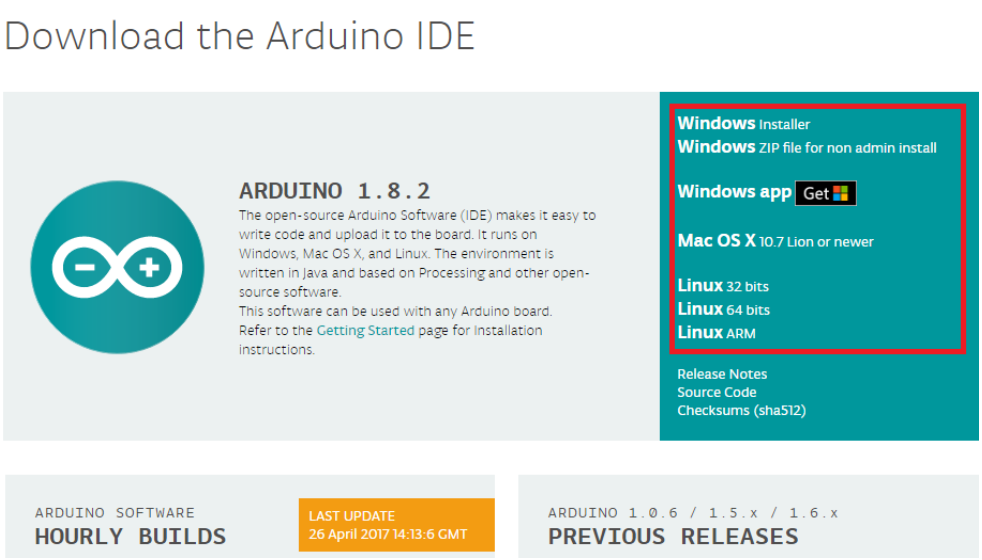
Käesolevas peatükis selgitatakse kuidas seada üles töökeskkond, kirjeldatakse kuidas teha esmased testid riistvaraga ja kuidas neid seadistada, et Lauatennise mängu programmeerimine kulgeks ladusalt.

3.1 Arduino arenduskeskkond

Arduino ametlikul lehel tutvustatakse arendustööks veebi- ja töölauaversiooni. Käesolevas töös kasutatakse Arduino IDE-t (inglise keeles *Integrated Developement Environment*) ehk integreeritud arenduskeskkonna töölauaversiooni. Programmeerimist saab teostada ka alternatiivsete tarkvaradega, kuid alustajale on kõige probleemivabam kasutada Arduino poolt pakutavaid keskkondasid. Järgnevalt kirjeldatakse Arduino IDE paigaldamist ja tutvustatakse arenduskeskkonda.

3.1.1 Paigaldamine

Esmalt tuleks minna Arduino ametlikule lehele *https://www.arduino.cc/en/Main/Software* ja olenevalt kasutatavast operatsioonisüsteemist valida endale sobiv versioon (vaata joonis 15).



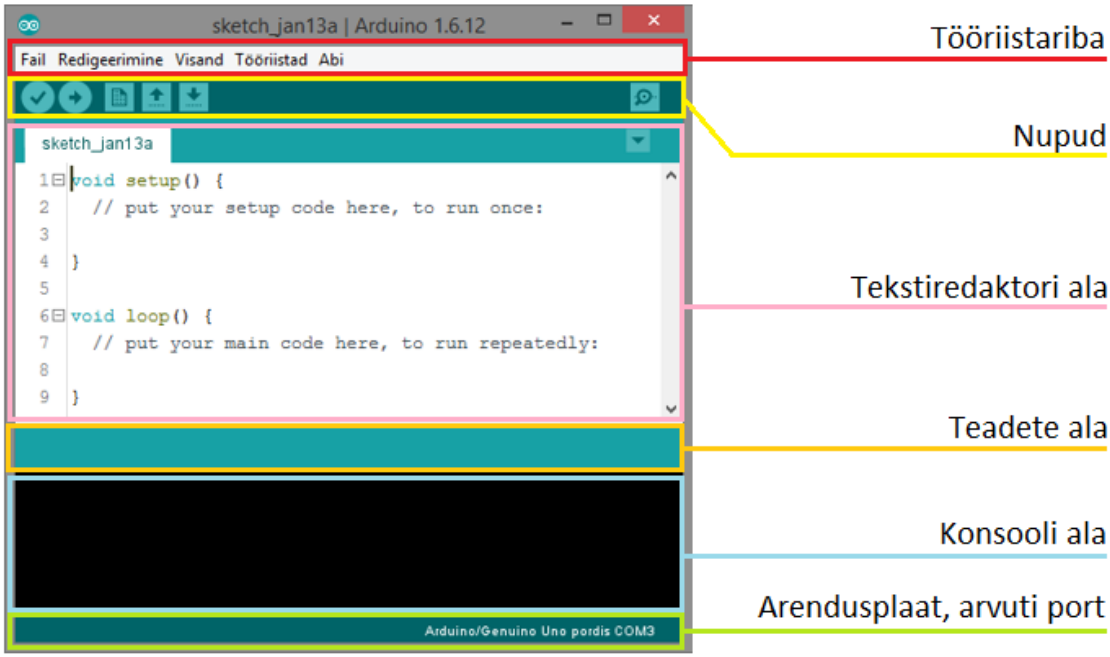
Joonis 15. Arduino IDE versioonide valik punases kastis.

Valides nimekirjast endale sobiva versiooni, avaneb järgmine vaade, millel tuleks klõpsata nupul "*JUST DOWNLOAD*", seda juhul kui ei soovita teha rahalist annetust Arduino meeskonnale. Seejärel tuleb avada alla laaditud .exe fail ning seejärel järgida installeerija juhiseid. Pärast paigaldamist on arenduskeskkond kasutatav. Järgnevalt kirjeldatakse arenduskeskkonda lähemalt.

3.1.2 Arenduskeskkonna töölauaversioon

Käesolevas peatükis kirjeldatakse Arduino IDE tähtsamaid osasid, et oleks selgem arusaam tööriistast enne selle kasutamist.

Arenduskeskkonna üldilme on näha joonisel 16. Peale vaadates on eristatavad kuus ala. Alustades keskkonna uurimist alt võib kahe silma vahele jääda osa, kus näidatakse hetkel valitud arendusplaati ja arvuti porti, kuhu arendusplaat on USB juhtmega ühendatud. Järgmine teistest eristuv ala on konsooli ala. Konsooli alale kuvatakse programmi kontrollimise ja üleslaadimise protsessi kohta käivat infot. Kohe konsooli ala peal asub teadete ala. Sinna kuvatakse lühikesed teated mingi protsessi tulemuse kohta. Kas on koodis mingi viga, üleslaadimine õnnestus vms. Teadete alale järgneb tekstiredaktori osa, kuhu kirjutatakse programmikood. Programmi nimetatakse Arduino kogukonnas visandiks (*sketch*). Seejärel on näha nupurida, kus vasakult paremale on: kontrolli, laadi üles, uus, ava, salvesta ja jadapordi monitor. Viimase alana võib eristada tööriistariba.



Joonis 16. Arduino IDE alade eristatavus

Arenduskeskkonnas kasutatav keel põhineb Wiring tarkvara raamistikul ja on implementeeritud C ja C++ keele põhjal. Kui puudub varasem kokkupuude C keelega, siis kasuks tuleb ka kogemus Java keelega [25][26].

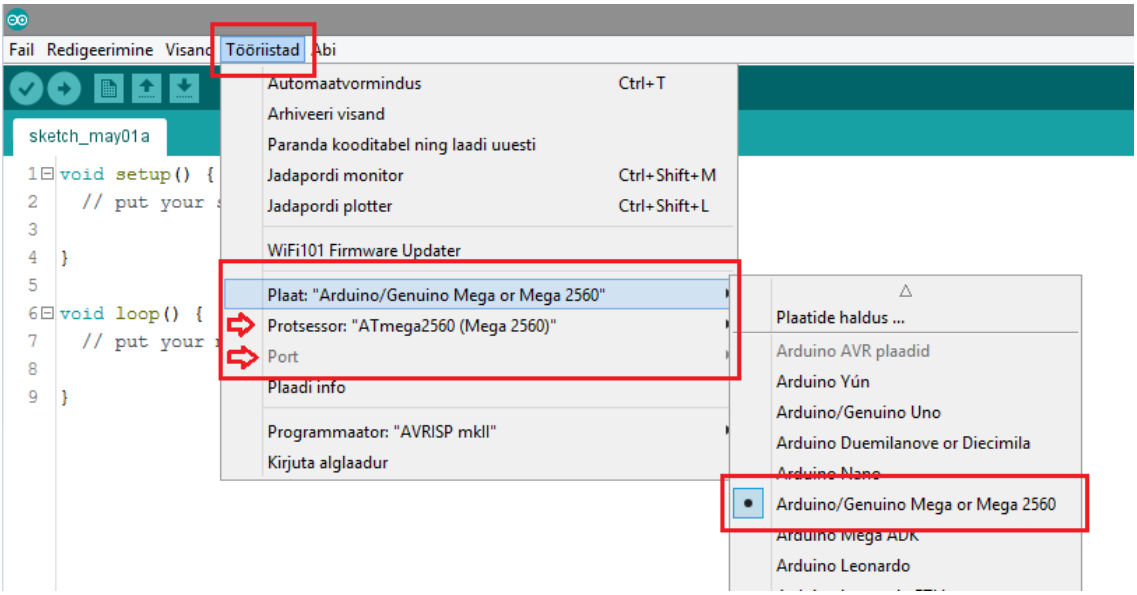
Joonisel 16 näeme, et kui arenduskeskkond avada, siis tekstiredaktori alas on juba mõned read kirjas. Tegemist on visandi esialgse struktuuriga. Funktsioon *void setup()* kutsutakse välja visandi käivitamisel üks kord. Visand käivitamiseks loetakse plaadi vooluringi ühendamist, uue visandi üleslaadimist või plaadi lähtestamise nupu vajutamist. [27]

Funktsioon *void loop()* kutsutakse välja pärast funktsiooni *setup()* täitmist. Koodi, mis on kirjutatud *loop()* funktsiooni sisse, korratakse lõpmatult. Seega kasutatakse *loop()* funktsiooni, et visandi töö saaks ajas muutuda vastavalt sisendsignaalidele. [28]

Käesolevas peatükis tutvustati ülevaatlikult arenduskeskkonda ja selgitati uue visandi automaatselt ilmunud koodiridasid. Järgmises peatükis kirjeldatakse visandi üles laadimist arendusplaadile.

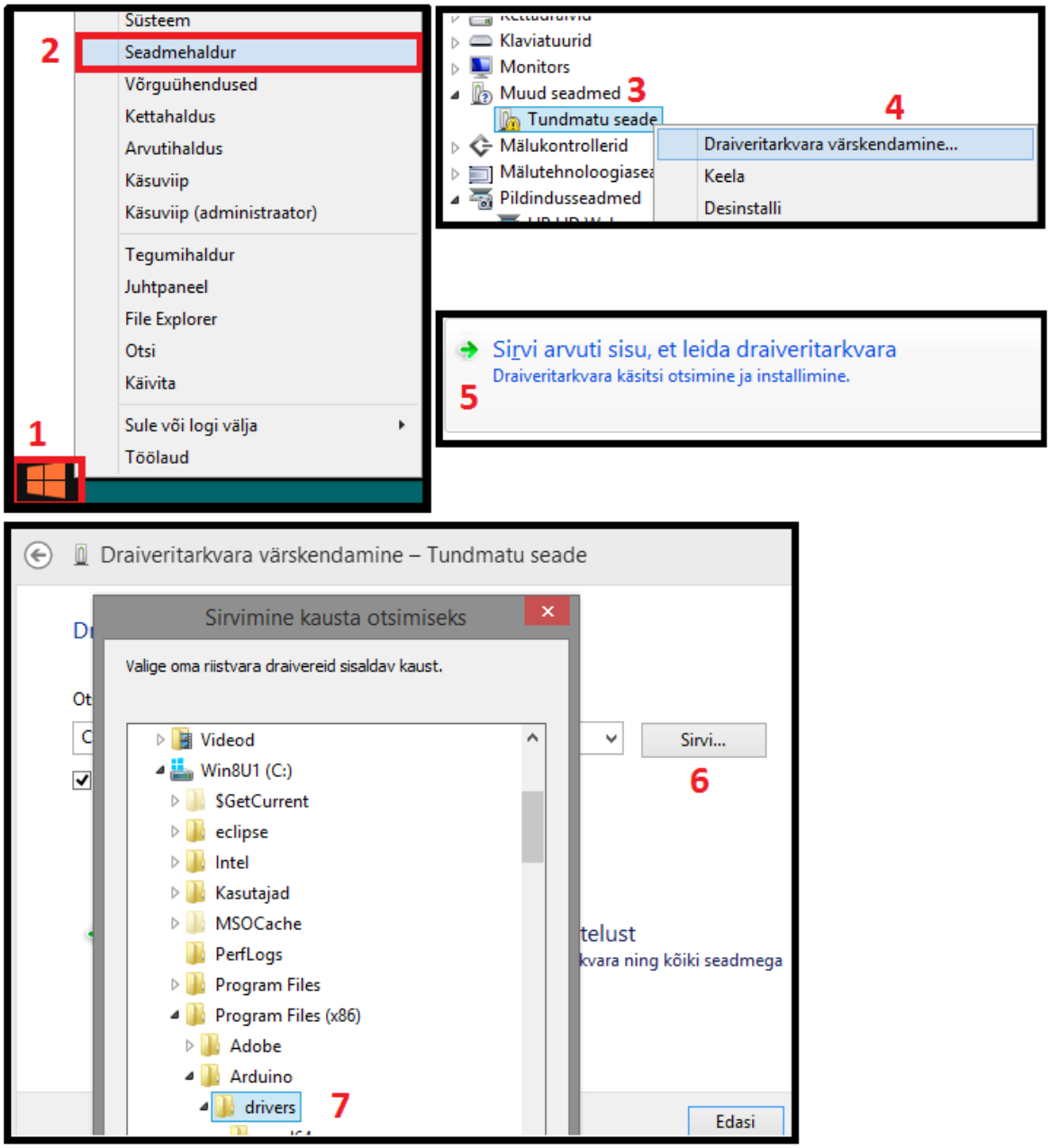
3.1.3 Visandi üles laadimine arendusplaadile

Järgnevalt kirjeldatakse Arduino IDE olulisimat omadust – visandeid on võimalik arendusplaadile üles laadida. Üles laadimiseks on vajalik arendusplaadi ühendus arvutiga. Seejärel tuleb tööriistaribalt „Tööriistad" menüüst valida õige plaat, protsessor ja port (vaata joonis 17).



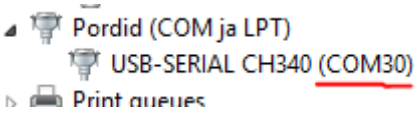
Joonis 17. Enne visandi üleslaadimist tuleb veenduda korrektses arendusplaadi mudeli, protsessori ja pordi valikus. Joonisel on märgistatud valikud Arduino Mega arendusplaadile üles laadimiseks

Pordi valiku tegemine võib osutuda kõige keerulisemaks, kuna on võimalus, et operatsioonisüsteem ei ole suutnud arendusplaadi draiverit korralikult arvutisse paigaldada. Sellisel juhul avaneb seadmehalduri avamisel vaatepilt „Tundmatust seadest" kollase kolmnurgaga ja vajalik on käsitsi draiveri paigaldus, mida on kirjeldatud joonisel 18.



Joonis 18. 1) parema hiireklahviga klõps „Start menüü“ ikoonil; 2) klõps valikul „Seadmehaldur“; 3) parem klõps valikul „Tundmatu seade“; 4) klõps valikul „Draiveritarkvara värskendamine“; 5) klõps kastil „Sirvi arvuti sisu...“; 6) klõps nupul „Sirvi“; 7) draiveri kausta valik

Kui kollast ohumärki pole seadmehalduri avamisel näha, siis tõenäoliselt on arendusplaadi pordi number leitav „Pordid (COM ja LPT)“ valikust (vaata joonis 19).



Joonis 19. Seadmehalduris kuvatav arendusplaadi port

Pärast korrektsete valikute tegemist ArduinoIDE-s tuleb klõpsata arenduskeskkonna nupul „Laadi üles“. Arenduskeskkond kuvab seepeale üles laadimise kohta käivat infot konsooli alale ja mõne aja möödudes peaks tulema teade „Üleslaadimine on lõpetatud“ teadete alale.

Erinevate riistvara komponentide kasutamiseks võib olla vajalik lisateekide installeerimine, kuna need ei pruugi kaasas olla Arduino IDE-ga. Järgnevalt selgitatakse teekide vajalikkust ja nende kasutamist töös.

3.1.4 Teegi mõiste tutvustus

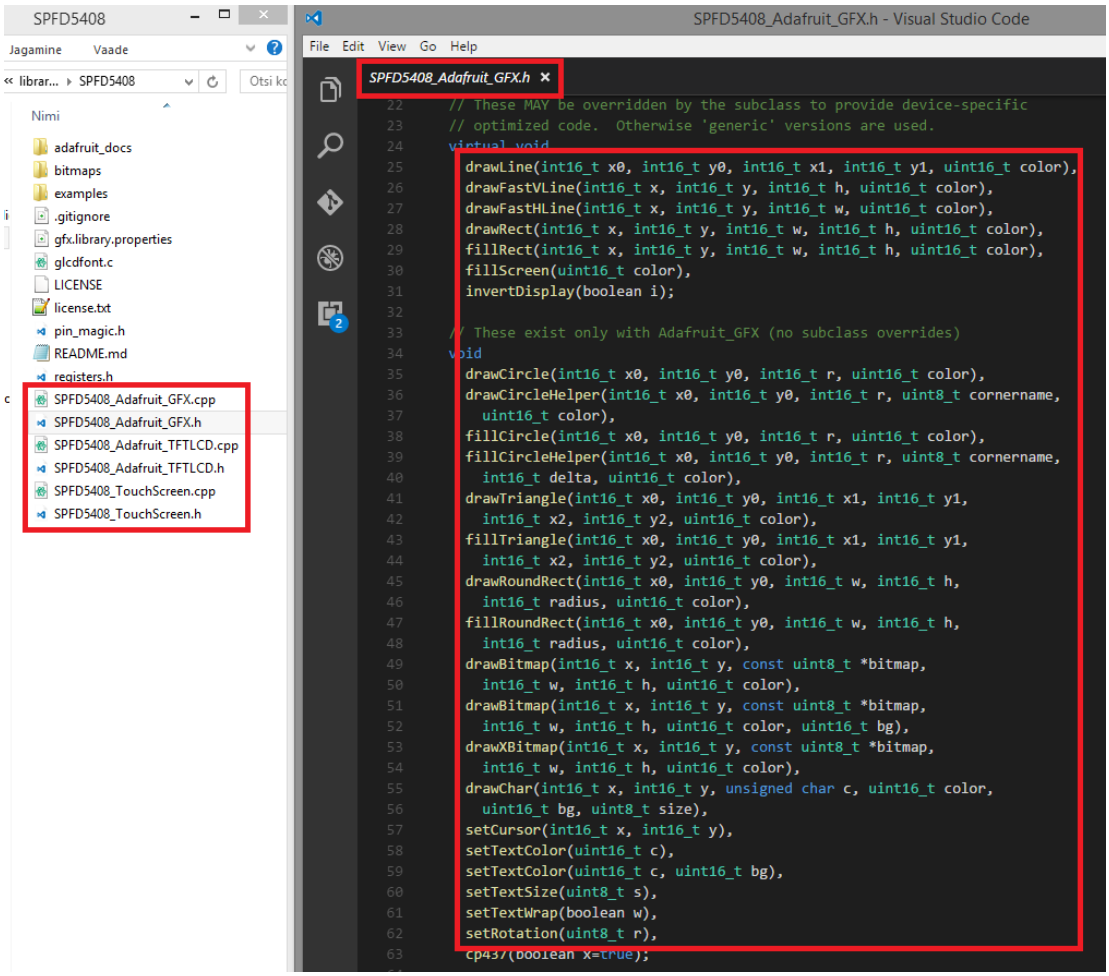
Selles alapeatükis selgitatakse, mis on teek ja milleks kasutatakse teeke Arduino arenduskeskkonnas.

Teegist võib mõtelda kui poest ostetud uuest tööriistakomplektist, kus on tööriistad, mida spetsialistil varem ei olnud. Programmeerimise mõttes on need tööriistad kellegi teise poolt loodud funktsioonid ja teek ise on funktsioonide kogumik [29]. Seega on teegid kasulikud, kuna nad lihtustavad oluliselt arendaja tööd ja hoiavad aega kokku.

Käesolevas lõputöös kasutatakse lisateeke riistvaraga suhtlemiseks ja protsesside ajaliseks juhtimiseks. Järgmisena kirjeldatakse ülevaatlikult teegi *SPFD5408* kausta, et välja tuua teegifailide uurimise olulisus.

Teegi *SPFD5408* kaustas on erinevad failid, millest leiame funktsioonid LCD kontrollimiseks. Failid, mille sisu tasub kindlasti uurida on ära märgistatud joonisel 20.

Joonisel 20 paremal on välja toodud faili *SPFD5408_Adafruit_GFX.h* sisalduv funktsioonide nimekiri, millest mõned funktsioonid on visandi koostamisel sagedaselt kasutusel. Teegi faile uurides saab teada, millised on kasutatavad funktsioonid, millised on vajalikud parameetrid ja nende tüübid ning näeb ka funktsiooni keha ülesehitust. Vajadusel saab ka teegi faile muuta (Näide: ekraani katsetamisel võib välja tulla, et ekraani pilt kuvatakse tagurpidi. Muutes vajalikus failis paar rida saab pildi õiget pidi ekraanile), kuid käesolevas töös ei ole muudatused vajalikud ning muudatuste tegemisest pikemalt ei räägita.



Joonis 20. Teegi failid, millest leiame riistvara kontrollimiseks kasulikud funktsioonid. Joonise tumedal alal on välja toodud ekraani mooduli juhtimiseks kasutatavad funktsioonid

Nimetatud teegi alla laadimisest ja paigaldamisest räägitakse lähemalt peatükis 3.3. Teekide kasutamisest kirjutatakse lähemalt 4. peatükis. Arduino IDE piisav kirjeldus ja

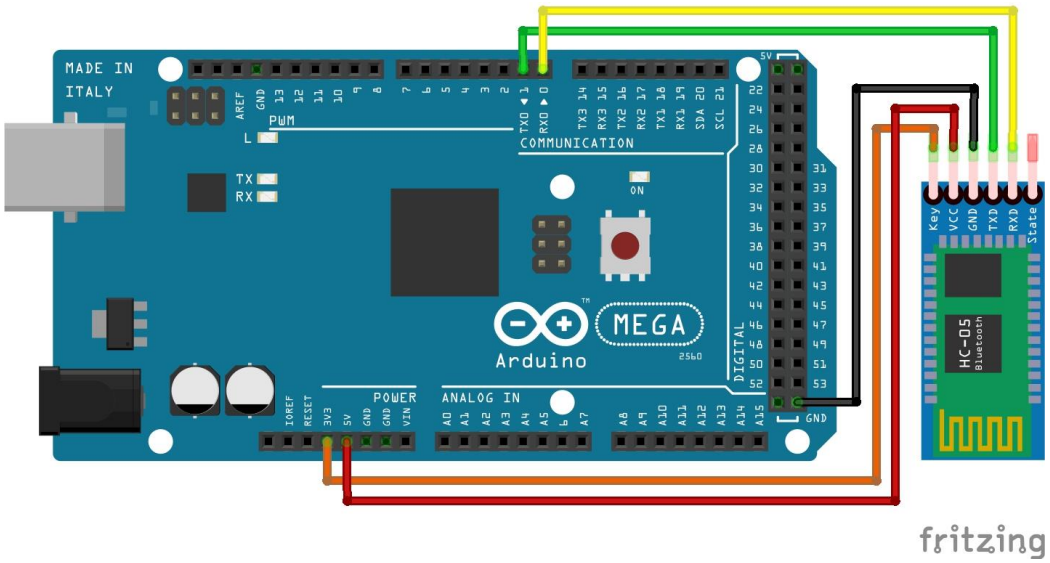
tutvustus tehti käesolevas peatükis. Edasi võetakse arenduskeskkond kasutusele ja alustatakse seadistustega.

3.2 HC-05 Bluetooth mooduli seadistamine

Järgnevas alapeatükis kirjeldatakse HC-05 Bluetooth mooduli seadistamise protsessi. Seadistamist alustatakse Bluetooth moodulitest, et ekraani mooduli paigaldamist Arduino Mega pesadesse ei peaks mitu korda tegema.

HC-05 Bluetooth mooduli süsteemi tööpingeks on 3,3V. Kuna moodullahendusel on sisseehitatud pingemuundurid, mis kaitsevad süsteemi 5V pinge eest, siis lihtsuse mõttes ei ole järgnevalt kirjeldatavas vooluringis ega ka töö lahenduses kasutatud mooduli väliseid pingemuundureid. Väliste pingemuundurite kasutamine on soovituslik riistvara kaitsmiseks liiga kõrge pinge eest. Seega on nende mittekasutamine arendaja enda vastutusel. [30]

Nii ülema kui ka alluva mooduli seadistused tehakse Arduino Mega arendusplaadiga, kuna seadistuseks on vajalik 3,3 volti. Kõigepealt tuleb Arduinole laadida tühi visand, kus on ainult *setup()* ja *loop()* funktsioonid kirjas. Pärast edukat visandi üleslaadimist tuleks Arduino toitest lahti ühendada. Seejärel ühendada juhtmed vastavalt joonisel 21 näidatule. Key või En viik tuleb ühendada Arduino arendusplaadi 3V3 pesaga, mooduli VCC viik tuleb ühendada Arduino 5V pesaga, mooduli GND viik tuleb ühendada Arduino GND pesaga, mooduli TXD viik tuleb ühendada Arduino TX0 pesaga ja mooduli RXD viik tuleb ühendada Arduino RX0 pesaga. VCC ja GND viikude ühenduste tegemisel ei pea järgima rangelt skeemi vaid võib kasutada ka teisi Arduino 5V ja GND pesasid.

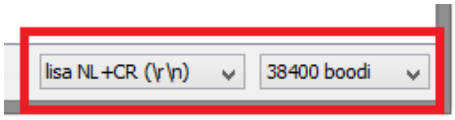


Joonis 21. HC-05 Bluetooth mooduli seadistamise skeem

Järgmisena tuleks panna HC-05 moodul seadistamise olekusse. Seadistamise olekusse sisenemiseks tuleb HC-05 moodulil vajutada pisikest nuppu (peatükk 2.4, joonis 7) ning seda all hoida, seejärel ühendada Arduino toitega ja umbes kahe sekundi möödudes võib nupu lahti lasta. Kui LED vilgub iga kahe sekundi järel, siis on moodul edukalt suunatud seadistamise olekusse.

Moodul seadistatakse läbi jadapordi monitori. Moodulile saadetakse käsklusi, mille üldkuju on järgmine. Käskluse algab tähistusega „AT”, sellele järgneb „+”, seejärel kirjutatakse parameetri nimi „parameeter”, edasi on kaks valikut. Kas tähistus „?”, millega küsitakse sisestatud parameetri olemasolevat väärtust, või tähistus „=”, millega seatakse sisestatud parameetrile uus väärtus. Kui tahetakse parameetrile seada uut väärtust, siis tuleb loomulikult ka uus väärtus pärast „=” tähistust kirjutada. [17]

Seadistamiseks tuleb lahti võtta jadapordi monitor, mis on ligipääsetav nupurealt viimasest nupust või tööriistaribalt „Tööriistad” menüü alt. Avanenud jadapordi monitoris tuleb teha seadistus, mis on näha joonisel 22.



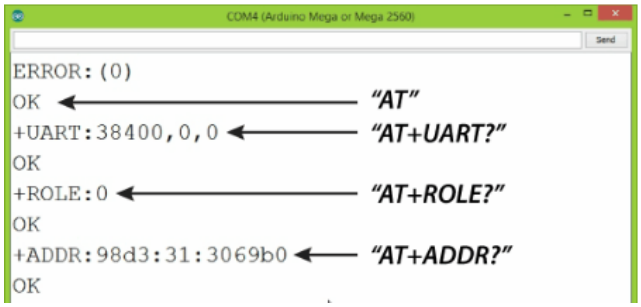
Joonis 22. Jadapordi monitoris vajalikud valikud enne Bluetooth mooduli seadistamist

Alljärgnevalt kirjeldatakse Bluetooth moodulite seadistamist vastavalt rollidele. Kirjeldused põhinevad Dejan Nedelkovski loodud veebimaterjalil [31].

3.2.1 Mooduli alluv seadistus

Järgnevas alapeatükis kirjeldatakse seadistamisolekus oleva HC-05 Bluetooth mooduli alluvaks seadistamise protsessi.

Moodul on algupäraselt juba alluva seadistusega, kuid sellegi poolest tuleb protsess läbi teha info kogumiseks ja seadistuse kontrollimiseks. Käesoleva bakalaureusetöö raames on olulised seadistamise käsud, mis on näha joonisel 23.



Joonis 23. Mooduli alluvaks seadistamise protsessi vajalikud käsud ja nende tagastused [31]

Seadistamist alustatakse testkäsuga, milleks on lihtsalt "AT", seejärel tuleb vajutada nuppu „Saada", joonisel 23 nupp „*Send*" üleval paremas nurgas. Joonisel 23 on näha esimesel real, et kui Bluetooth moodul ei ole veel suhtluseks valmis, tagastatakse käsule "AT" viga. Vea tagastamisel tuleb testkäsklus uuesti sisestada ja saata. Kui tagastuseks ei ole "OK" ka teist ega ka kolmandat korda, siis tuleb korrata peatüki 3.2 alguses kirjeldatud

mooduli seadistamisolekusse viimist. Kui testkäskluse vastuseks saadakse “OK”, siis saab seadistamise protsessiga edasi liikuda.

Järgmisena tuleks üle kontrollida sümboli edastamise kiirus ehk boodikiirus, mis peaks olema 38400 boodi ehk sümbolit sekundis. Kiiruse kontrollimiseks tuleb sisestada käsk “AT+UART?”. Joonisel 23 on näha, et tagastatakse kolm parameetrit. Lihtsuse mõttes huvitab meid käesolevas töös vaid esimene neist. Kui käsu “AT+UART?” tagastus on joonisel 23 olevast erinev, siis tuleks seadistada boodikiirus käsuga “AT+UART=38400,0,0”.

Kindlasti tuleb üle kontrollida mooduli roll käsuga “AT+ROLE?”, mille tagastuseks peaks olema “+ROLE: 0”. Vastasel juhul tuleks teha seadistus käsuga “AT+ROLE=0”. Number 0 tähistab käsus alluva rolli.

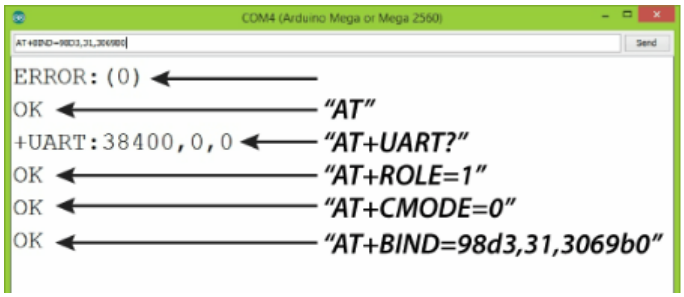
Kui rolli kontrollimine on teostatud, siis on vaja veel pärida ainult mooduli aadressi, mille poole ülemmoodul saab pöörduda. Seda tuleks teha käsuga “AT+ADDR?”, mille tagastuseks kuvatakse arv, mis sarnaneb joonisel 23 olevaga. Tagastuseks saadud arv tuleb kirja panna. Näiteks joonisel 23 kuvatud arv tuleks kirja panna viisil „98d3,31,3069b0”, seejuures tuleb tähele panna, et koolonid tuleb komadega asendada.

Mooduli alluvaks seadistamise protsess on lõppenud ja kindlasti tuleks moodul tähistada, et hilisemalt erinevate seadistustega moodulid sassi ei läheks. Arduino tuleb toitest lahti ühendada, et minna tööga edasi. Järgmises alapeatükis kirjeldatakse seadistamisolekus HC-05 Bluetooth mooduli ülemaks seadistamise protsessi.

3.2.2 Mooduli ülem seadistus

Järgnevas alapeatükis kirjeldatakse seadistamisolekus oleva HC-05 Bluetooth mooduli ülemaks seadistamise protsessi.

Protsessi eelduseks on peatükis 3.2.1 läbitud alluvaks seadistamise protsess teise Bluetooth mooduliga, kuna järgnevas protsessis vajatakse alluva mooduli aadressi. Ülemaks seadistamise käsud on näha joonisel 24.



Joonis 24. Bluetooth mooduli ülemaks seadistamise protsessis kasutatavad käsud ning nende tagastused [31]

Seadistamist alustatakse testkäsuga, milleks on lihtsalt "AT", seejärel tuleb vajutada saada, joonisel 24 nupp "Send". Joonisel 24 on näha esimesel real, et kui Bluetooth moodul ei ole veel suhtluseks valmis, tagastatakse käsule "AT" viga. Vea tagastamisel tuleb testkäsklus uuesti sisestada ja saata. Kui tagastuseks ei ole "OK" ka teist korda ega ka kolmandat korda, siis tuleb korrata peatüki 3.2 alguses kirjeldatud mooduli seadistamisolekusse viimist. Kui testkäskluse vastuseks saadakse "OK", saab seadistamise protsessiga edasi liikuda.

Järgmisena tuleks üle kontrollida sümboli edastamise kiirus ehk boodikiirus, mis peaks olema 38400 boodi ehk sümbolit sekundis. Kiiruse kontrollimiseks tuleb sisestada käsk "AT+UART?". Joonisel 24 on näha, et tagastatakse kolm parameetrit. Lihtsuse mõttes huvitab meid käesolevas töös vaid esimene neist. Kui käsu "AT+UART?" tagastus on joonisel 24 olevast erinev, siis tuleks seadistada boodikiirus käsuga "AT+UART=38400,0,0".

Edasi tuleks määrata antud moodul ülemaks, mida saab teha käsuga "AT+ROLE=1". Number üks tähistab käskluses ülemaks olemist. Tagastuseks käsklusele kuvatakse "OK".

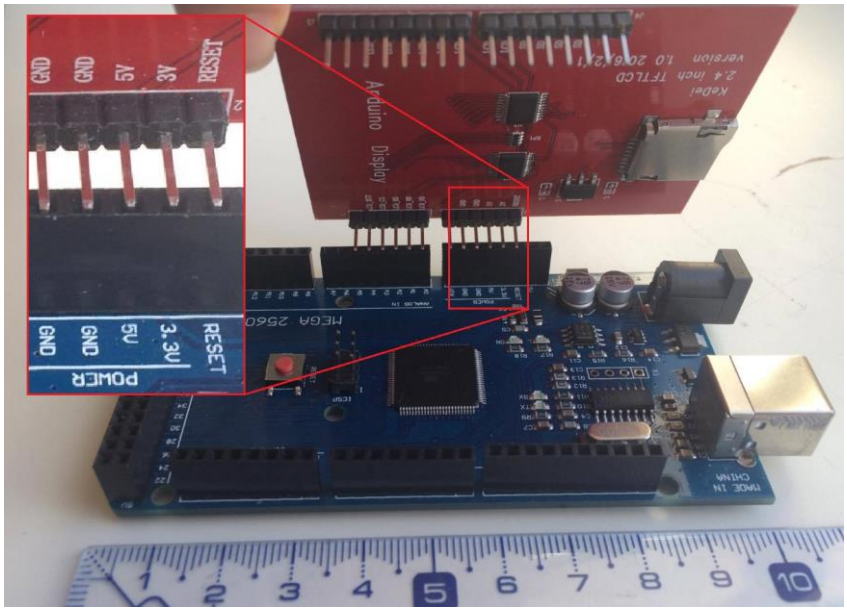
Käesolevas töös peab kindla Bluetooth moodulite paari vahel toimuma andmevahetus. Selle tagamiseks seadistatakse ülemmoodul ühenduma kindlal aadressil oleva mooduliga. Kõigepealt sisestatakse käsk "AT+CMODE=0", millega määratakse ülemmoodul ühenduma ainult ette antaval aadressil asuva mooduliga. Järgmise käsuga "AT+BIND=aadress" kirjeldatakse ülemale alamseadistusega mooduli aadressi, millega ühenduda.

Kui käsud on edukalt sisestatud, siis on HC-05 moodul ülemaks seadistatud. Seadistatud moodul tuleks kindlasti vastavalt tähistada. Arduino tuleb toitest lahti ühendada, et tööga edasi minna. Järgmiseks teostatakse vajutustundliku ekraani mooduli seadistamine.

3.3 Arduino Mega ja ekraani mooduli andmevahetuse loomine

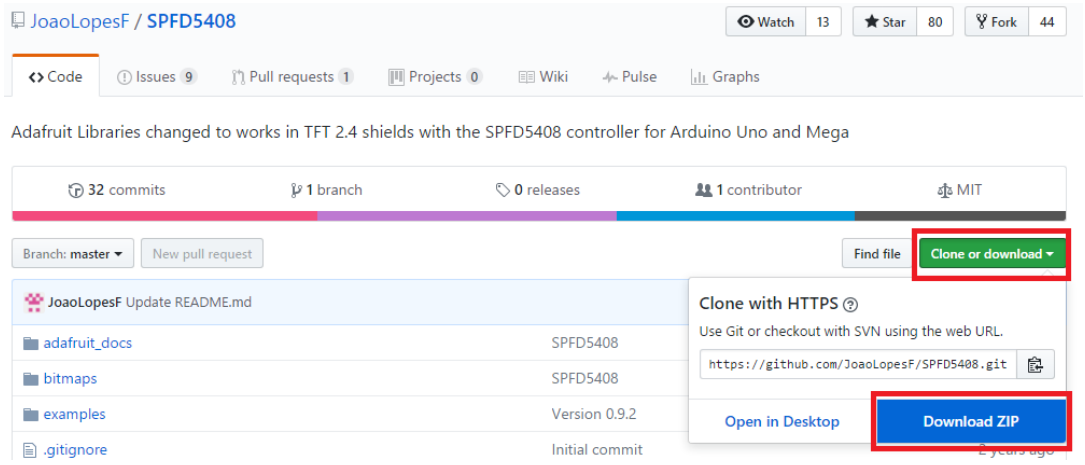
Järgnevalt kirjeldatakse ja selgitatakse Arduino Mega ja vajutustundliku ekraani mooduli andmevahetuse ülesseadmist.

Protsessi alustatakse Arduino MEGA arendusplaadi ja ekraani mooduli kokku sobitamisest. Juhinduda tuleks mooduli RESET viigu ja Arduino RESET pesa, mooduli 3,3V viigu ja Arduino 3V pesa, mooduli 5V viigu ja Arduino 5V pesa ühendamisest (vaata joonis 25). Kui nimetatud viigud ja pesad asetatakse vastavusse, siis klapivad ka ülejäänud ekraani viigud ja Arduino pesad omavahel. Eduka paigaldamise järel on tarvis alla laadida ekraaniga töötamiseks sobiv teek. Konkreetse ekraani puhul on sobivaks teegiks Jao Lopesi poolt modifitseeritud Adafruidi teekide kogumik, mis on kättesaadaval Githubi repositooriumist *https://github.com/JoaoLopesF/SPFD5408*.



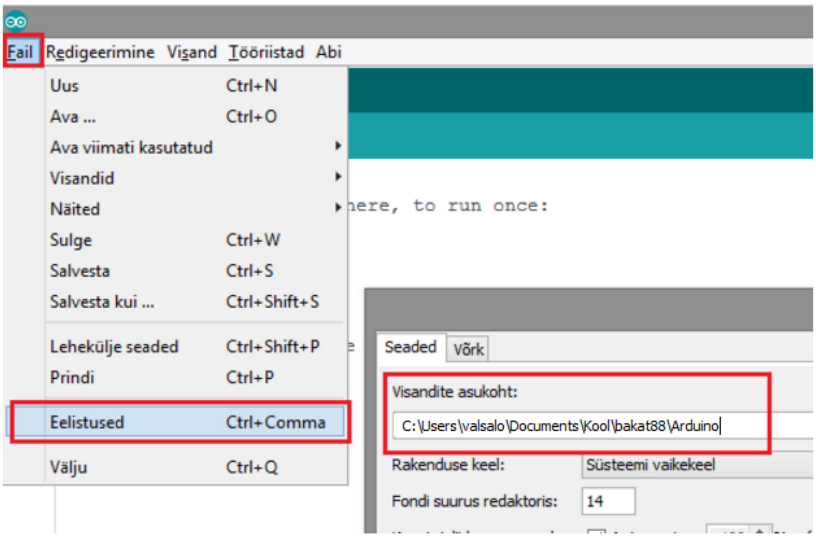
Joonis 25. Arduino Mega ja ekraani mooduli viikude joondus ühel küljel.

Teek on mõeldud 2,4 tolliste ekraanide jaoks. Järgmiseks tuleb alla tõmmata teek ZIP-failina (vaata joonis 26).



Joonis 26. J. Lopesi repositooriumist teegi alla laadimiseks vajalikud nupud tähistatult

Alla laaditud ZIP-fail tuleb lahti pakkida visandite asukohta, mida saab järgi vaadata klõpsates „Fail” ning seejärel „Eelistused” (vaata joonis 27).



Joonis 27. Visandite asukoha teada saamiseks tehtavad sammud

Visandite asukohta saab kasutaja vajadusel ka muuta, kui kirjutada teine teekond joonisel 27 märgitud tekstiväljale. Pärast teegi lahti pakkimist visandite kausta tuleb taaskäivitada Arduino IDE, et teeki kasutada. Esimene test tuleks sooritada teegi poolt pakutava näitevisandiga. Visandi leidmiseks tuleb liikuda „Fail" -> „Näited" -> „SPFD5408" -> „spfd5408_graphicstest". Avanevas aknas on näidisvisand, mille üleslaadimisel arendusplaadile kuvatakse ekraanile erinevaid kujundeid. Näitevisandisse on kirjutatud kood, mille abil luuakse ekraani isend, mille abil kuvatakse ekraanile kujutisi.

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

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

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

void setup() { tft.begin(0x9341); }
```

Võib juhtuda, et kujundeid ei kuvata, järelikult ei ole teek sobiv ekraanile. Uue teegi otsimiseks saab kasutada näitevisandi koodi algust.

```
//  uint16_t identifier = tft.readID();
//
//  if(identifier == 0x9325) {
//    Serial.println(F("Found ILI9325 LCD driver"));
//  } else if(identifier == 0x9328) {
//    Serial.println(F("Found ILI9328 LCD driver"));
//  } else if(identifier == 0x7575) {
//    Serial.println(F("Found HX8347G LCD driver"));
//  } else if(identifier == 0x9341) {
//    Serial.println(F("Found ILI9341 LCD driver"));
//  } else if(identifier == 0x8357) {
//    Serial.println(F("Found HX8357D LCD driver"));
//  } else {
//    Serial.print(F("Unknown LCD driver chip: "));
//    Serial.println(identifier, HEX);
//     Serial.println(F("If using the Adafruit 2.8\" TFT Arduino
shield, the line:"));
//    Serial.println(F("  #define USE_ADAFRUIT_SHIELD_PINOUT"));
//       Serial.println(F("should appear in the library header
(Adafruit_TFT.h)."));
//    Serial.println(F("If using the breakout board, it should NOT
be #defined!"));
//     Serial.println(F("Also if using the breakout, double-check
that all wiring"));
//    Serial.println(F("matches the tutorial."));
//    return;
//  }
//  tft.begin(identifier);
```

Kood tuleb sisse kommenteerida ning uuesti üles laadida arendusplaadile. Seejärel tuleb avada jadapordi monitor, seadistuseks 9600 boodi ja reavahetuseta, kus kuvatakse LCD

draiveri (*driver*) kiibi kuueteistkümnendarv. Selle alusel otsida veebist ekraani kiibile vastavat teeki. Käesolevas töös keskendume tutvustatud teegi kasutamisele.

Järgmisena tuleb tegeleda ekraani vajutustundlikusega. Taaskord leiab näitevisandite alt vajutustundlikuse kalibreerimiseks visandi spfd5408_calibrate. Enne koodi üleslaadimist tuleb asendada visandis real 29 olev järgnevaga: *#define YM 5*, et visand töötaks käesoleva ekraani mooduliga. Seejärel tuleb kood üles laadida arendusplaadile ja järgida ekraanile kuvatavaid juhiseid. Kalibreerimise tulemused kuvatakse ekraanile ja kirja tuleks panna TS_MINX, TS_MINY, TS_MAXX ja TS_MAXY väärtused. Neid arve läheb hilisemas töökäigus vaja, mida on kirjeldatud lähemalt 4. peatükis. Kalibreerimise näitevisandist on kirjas kood vajutustundlikkuse tarvis.

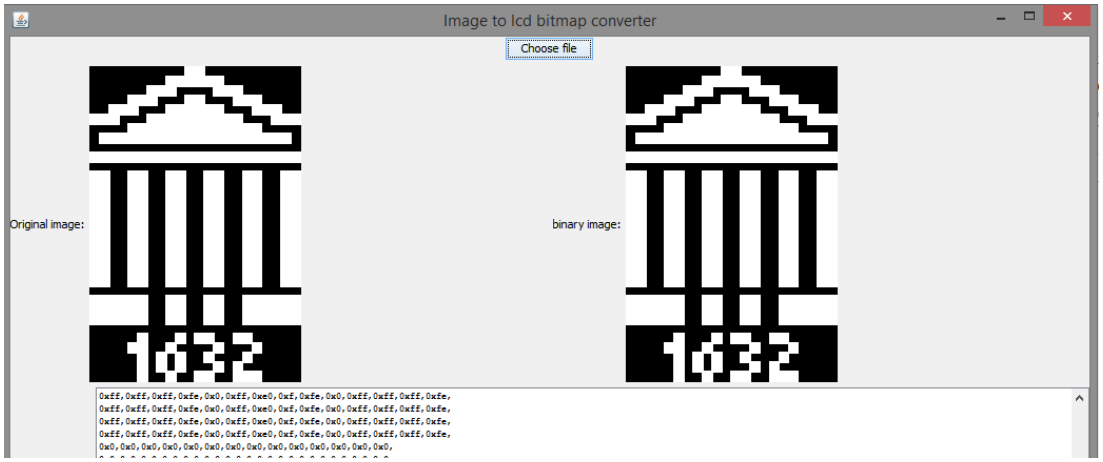
```
#include <SPFD5408_TouchScreen.h>

#define SENSIBILITY 300
#define YP A1
#define XM A2
#define YM 5
#define XP 6

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

Ekraani mooduliga on ettevalmistustöö lõpetatud. Arduino tuleks toitest lahti ühendada enne järgmise punkti juurde liikumist. Alljärgnevalt kirjeldatakse riistvara komponentidest vooluringi koostamist.

3.4 Pisipildi baidimassiiviks muutmine

Käesolev alapeatükk põhineb Nick Koumarise koostatud veebimaterjalil [32]. Järgnevalt kirjeldatakse kuidas teha logo eeltöötlus, et seda saaks ekraanile kuvada.

Alustuseks on vaja leida sobiv logo, mis peaks olema mustvalge värvilahendusega (pilditöötlusprogrammi abil on võimalik mustvalgeks muuta) ja logo kujutis peaks olema võimalikult selgete piiridega. Järgnevas näites on kasutatud Tartu Ülikooli pikselmärki [33]. Tähele tuleb panna, et logo kuvatav osa peab olema valget värvi ja taust must (vaata joonis 27).



Joonis 27. Img2Code tööriist töötleb rastergraafikas pildi baidimassiiviks, mis kuvatakse tööriistas valgele tekstialale

Valitud logo suurus tuleb muuta selliseks, millisena soovitakse teda ekraanil kuvada ning muutmistulemus tuleb salvestada .bmp faililaiendiga. Pildi laius ja kõrgus tuleb meelde jätta, sest seda tuleb programmeerimisel teada. Toimingu saab edukalt sooritada programmis MS Paint. Edasi tuleb alla tõmmata programm Img2Code, millega saab rastergraafikas pildi muuta 16-bitiseks baidi massiiviks. Tööriista leiab aadressilt *https://github.com/ehubin/Adafruit-GFX-Library/tree/master/Img2Code***.** Seejärel tuleb avada pilt programmis. Genereeritud baidimassiivi saab kasutada mugavalt logo kuvamiseks, millest kirjutatakse lähemalt peatükis 4 (vaata joonis 28).

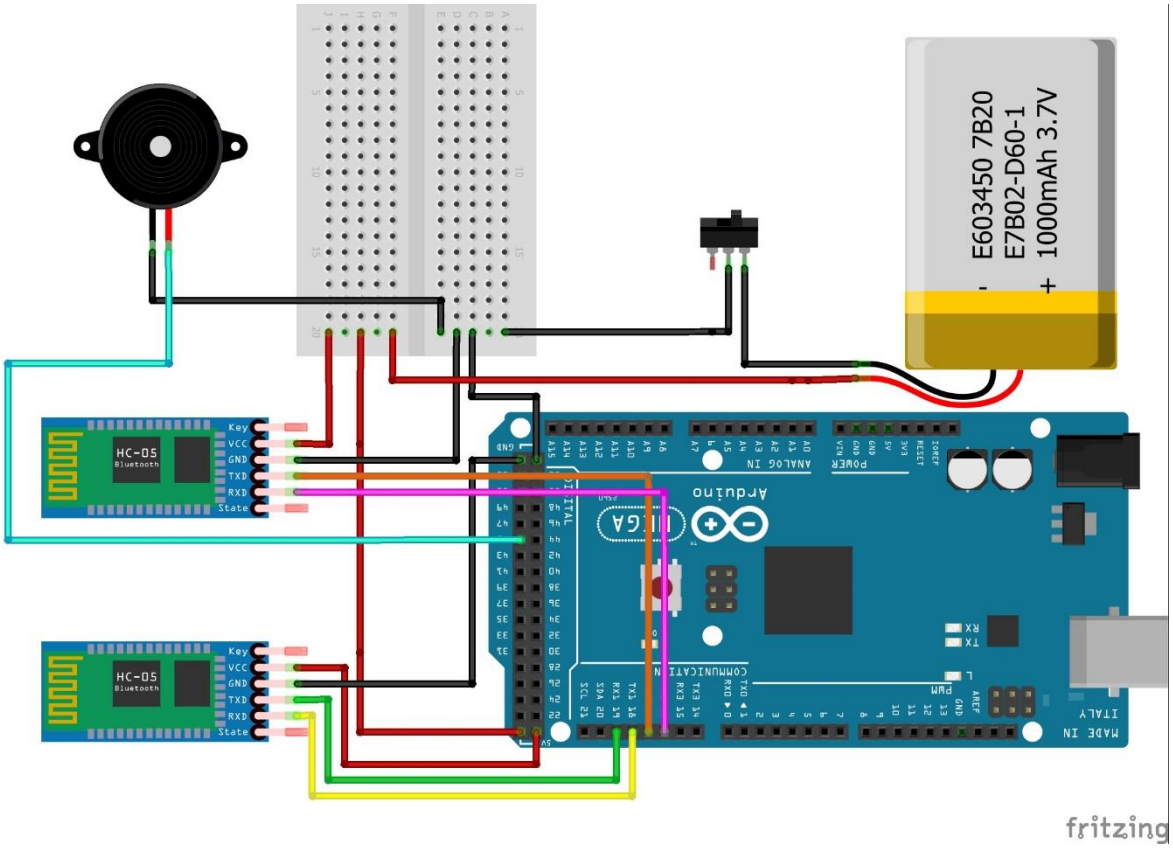
Joonis 28. Baidimassiivi kuvamine ekraani moodulil

Baidimassiivide abil saab kuvada mustvalgeid rastergraafikas pilte terve ekraani ulatuses. Kuid baidimassiivide kasutamine visandis hõivab märgatavalt mäluruumi. Mida suurem on kuvatav pilt, seda rohkem kasutatakse mälu.

Riistvara komponentide korrektseks toimimiseks on vaja teha vastavad ühendused. Järgnevalt kirjeldatakse vooluringide koostamist.

3.5 Ekraaniga komponendi vooluring

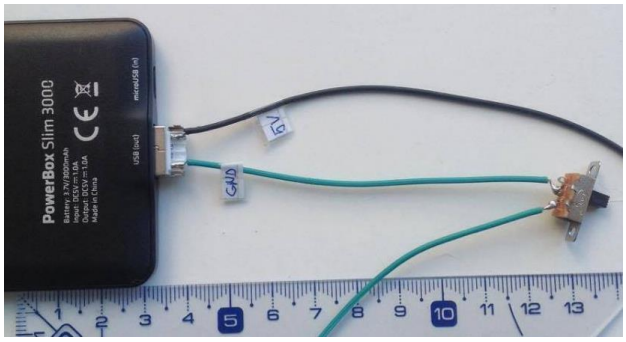
Järgnevalt kirjeldatakse vooluringi koostamise skeemi ja olulisemaid detaile, mida silmas pidada. Seadme keskse osa vooluring, mille kaudu saavad mängijad tagasisidet mängus toimuva kohta, on välja toodud joonisel 29.



Joonis 29. Ekraaniga komponendi skeem

Kollaste kastidega on tähistatud ekraani moodulile vajalikud Arduino viigud. Sumisti juhtimiseks kasutatakse Arduino pesa number 45. Oluline on silmas pidada, et joonisel 29 näidatud vooluringis osaleb kaks HC-05 Bluetooth moodulit: ülem ja ülem või alluv ja alluv. Olenevalt sellest kuidas arendaja ette näeb. Moodulite ühendamiseks on Arduino Mega arendusplaadil selleks puhuks veel kolm paari RX ja TX viikusid, tänu millele saame luua segamatu suhtluse kahe puldi ja Arduno vahel. Ühe Bluetooth mooduli TXD viik tuleb ühendada Arduino RX1 viiguga ja mooduli RXD viik tuleb ühendada Arduino TX1 viiguga. Teise mooduli ühendamine tuleb teostada sarnaselt, kuid Arduino RX1 ja TX1 viikude asemel tuleb kasutada RX2 ja TX2 viikusid.

Joonisel 29 on illustreerivalt lisatud ka akupanga kujutis, mis vastab tegelikule juhtmete paigutusele (vaata joonis 30).



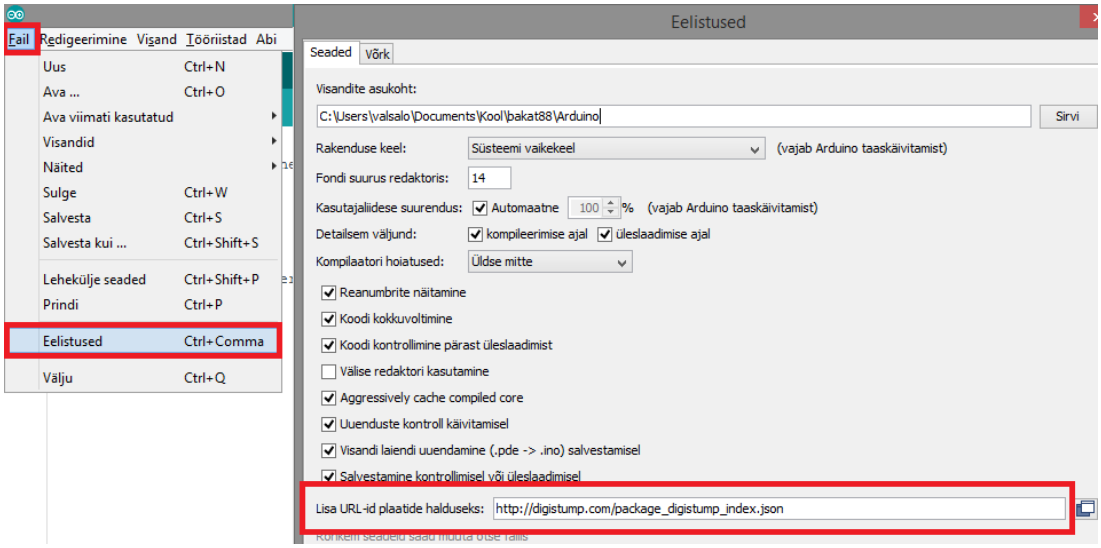
Joonis 30. Akupanga sisestatud modifitseeritud USB pistik koos liuglülitiga.

Joonisel 30 on maketeerimislaud ära toodud selleks, et näidata kuidas enne jootmist vooluring oli üles seatud. Maketeerimislauda kasutatakse maanduse ja pinge jagamiseks teistele seadmetele. Jootmisjärgselt ei kasutata seadmes maketeerimislauda.

3.6 Arenduskeskkonna seadistamine Digispargi programmeerimiseks

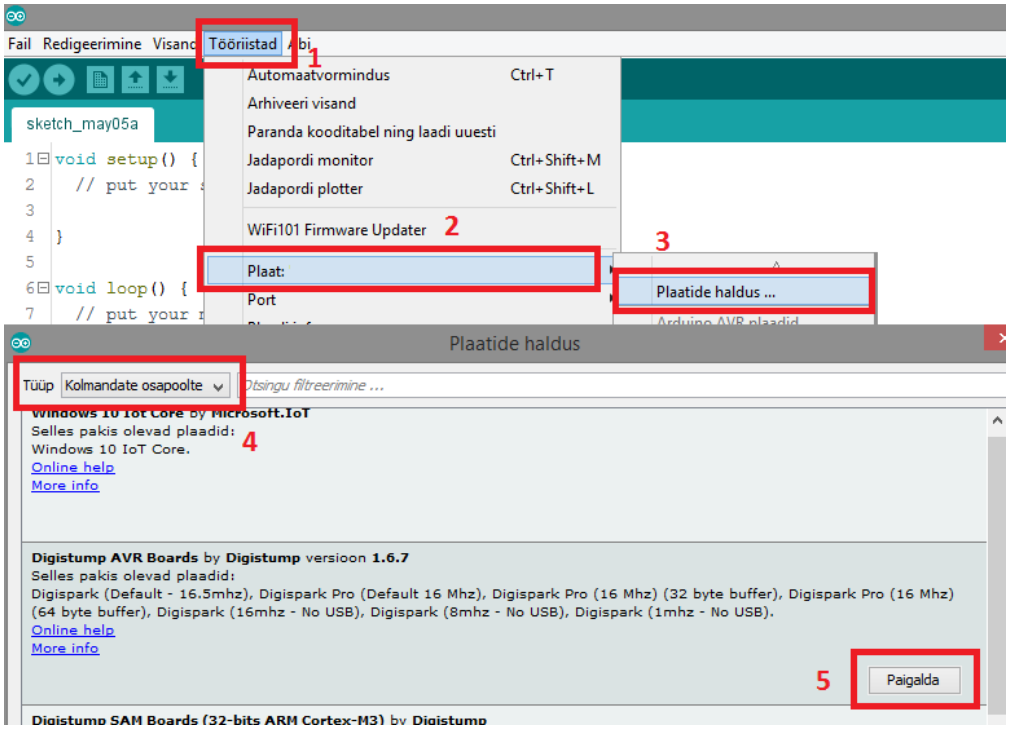
Järgnev peatükk põhineb Digispargi ametlikul juhendil, kus kirjeldatakse Arduino arenduskeskkonna seadistamist Digispargi arendusplaadi programmeerimiseks [34].

Alustuseks tuleb liikuda “Fail”->”Eelistused” ja avanevas aknas otsida üles tekstiväli, kuhu saab sisestada lisa internetiaaddresi plaatide halduse jaoks (vaata joonis 31).



Joonis 31. Lisa internetiaadressi lisamise tekstiväli

Nagu joonisel 31 näha, siis tekstiväljale „Lisa URL -id plaatide halduseks” tuleb kirjutada internetiaadress *http://digistump.com/package_digistump_index.json* ning seejärel klõpsata nupul „OK”. Järgmisena tuleb liikuda plaatide haldamise aknasse ning paigaldada Digispargile sobiv plaadi pakett (vaata joonis 32).



Joonis 32. Arendusplaadi Digispark sobiliku paketi paigaldamiseks vajalikud sammud

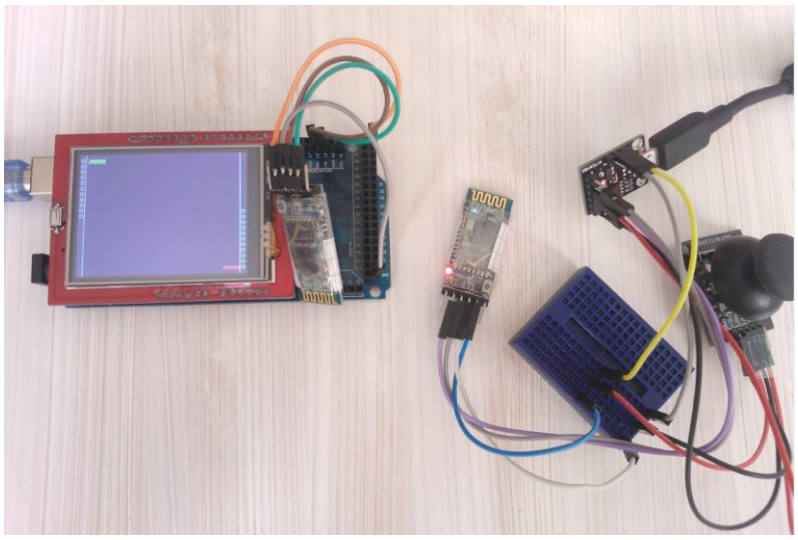
Pärast paigaldamise lõppu võib arvuti operatsioonisüsteem teada anda draiverite paigaldamise soovist, aknas tuleks klõpsata nupul “Edasi”.

Visandi üleslaadimise protsess Digispargi puhul näeb välja pisut teistmoodi, kui Arduino puhul. Sarnaselt peatükis 3.1.3 joonisel 17 nähtavale tuleb valida plaadi tüüp, milleks Digispargi puhul on “*Digispark (Default - 16.5 Mhz)*”. Erinevuseks on see, et nüüd ei ole oluline protsessori ega ka pordi valik. Digispark peab olema lahti ühendatud arvuti küljest enne, kui vajutatakse visandi üles laadimise nuppu. Digispark tuleb ühendada arvutiga, kui arenduskeskkonna konsooli alale tuleb vastav teade. Seejärel on Digispark arendusplaadi ühendamiseks 60 sekundit.

3.7 Juhtpuldi vooluring

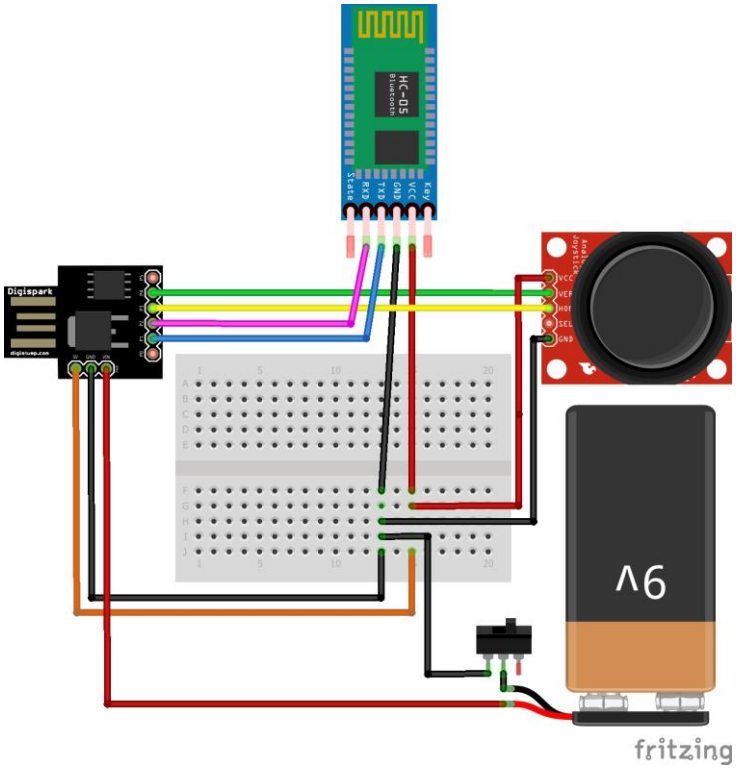
Järgnevas alapeatükis kirjeldatakse juhtmevaba Bluetooth juhtpuldi vooluringi skeemi koostamist.

Arendamisjärgus ja enne jootmist koosneb juhtpult Digsipark arendusplaadist, HC-05 Bluetooth moodulist, analoog juhtkangi moodulist ja maketeerimislauast (vaata joonis 33).



Joonis 33. Juhtmevaba juhtimise katsetus Bluetooth andmeedastuse abil

Joonisel 34 on kujutatud teistsuguse ühendamissüsteemiga Digispark arendusplaati, kuid arendusplaadi ülejäänud ehitus on sama nagu peatükis 2.3 välja toodud Digispark arendusplaadil. Joonisele 34 on lisatud ka liuglüliti ja toiteallikas, et ilmestada juhtpuldi komponentide lõpplahendust. Katsetamise ja arendamis vältel võib juhtpulti toita siiski USB kaabli abil.



Joonis 34. Juhtpuldi skeem. Autori koostatud

Kogu vooluring saab toidet peatükis 2.9 kirjeldatud 9 voldisest leelis patareist, mille positiivne ühendusklemmi juhe on ühendatud Digispark arendusplaadi VIN viiguga ja negatiivne juhtme ots on ühendatud liuglüliti keskmise klemmiga. Joonise gg keskel on näha maketeerimislauda, mida kasutatakse maanduse ja toite jagamiseks seadmete vahel. Suhtlemist Digispargi ja HC-05 mooduli vahel teostatakse järgmiste ühenduste abil: Digispargi P1 viik on ühendatud Bluetooth mooduli TXD viiguga ja Digispargi P2 viik on ühendatud mooduli RXD viiguga. Suhtlemist Digispargi ja analoog juhtkangi mooduli vahel teostatakse järgmiste ühenduste abil: Digispargi P3 viik on ühendatud juhtkangi HOR (või VRx, olenevalt tähistusest) viiguga ja Digispargi P4 viik on ühendatud VER (või VRy) viiguga.

Ettevalmistused programmeerimiseks on tehtud. Edasine töö on seotud tihedalt Arduino IDE kasutamisega ja sellest antakse ülevaade järgmises peatükis.

4. Lauatennise mängu programmeerimine

Järgnevas peatükis antakse ülevaade mängu loogikast, selgitatakse teekide kasutamist, kirjeldatakse käesolevas töös kasutatud olulisemaid funktsioone ja võtteid visandi koostamisel.

Enne programmeerimisvõtete kirjeldamist ja selgitamist kirjeldatakse valminud mängukonsooli (vaata joonis 35).

Joonis 35. Arendustöö tulemuseks on juhtmevabade pultidega Arduino LCD Lauatennis

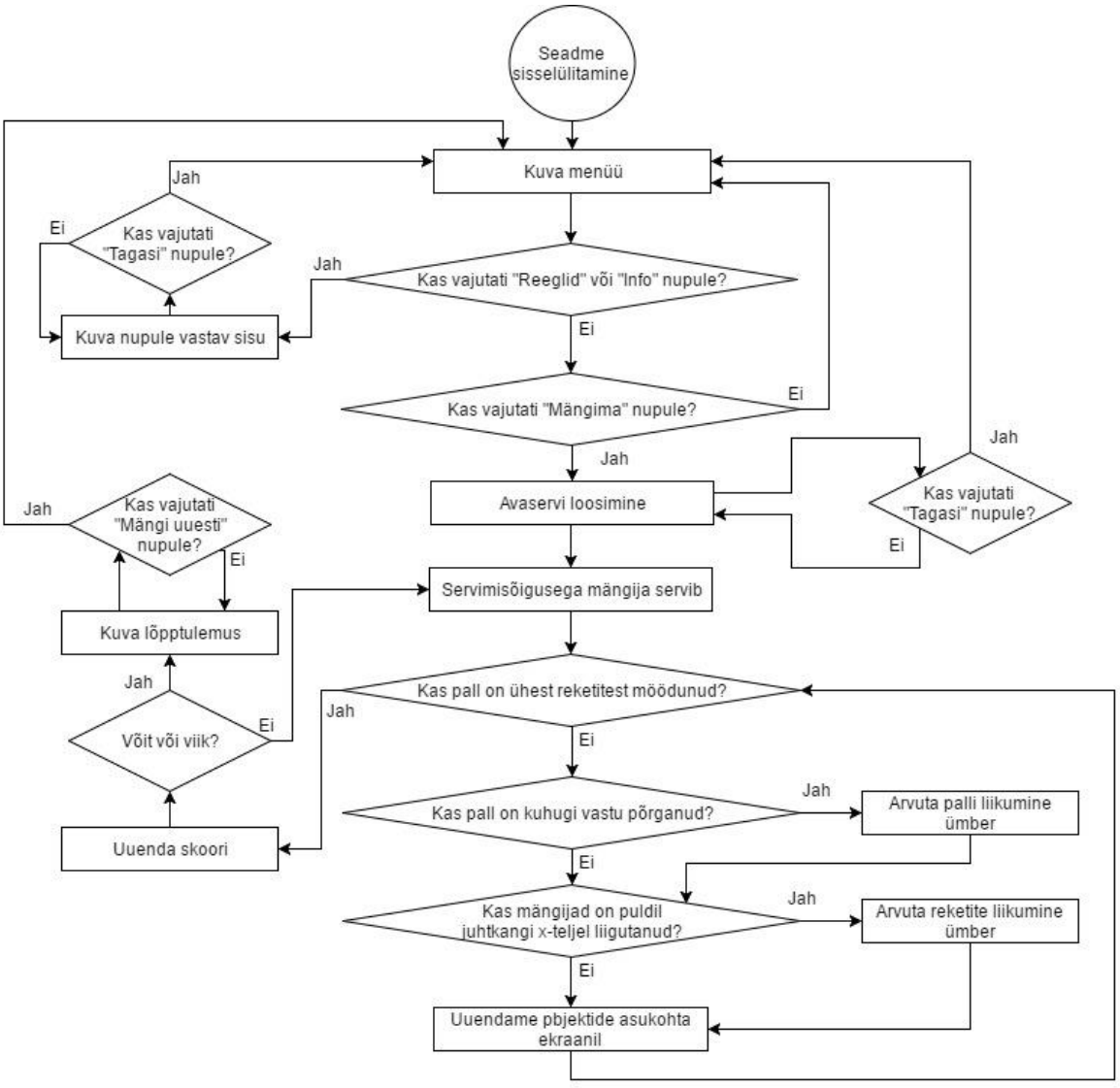
Mängukonsoolile saab luua korpuse, et kasutuskogemus oleks parem. Nõuete kohaselt on ekraaniga osa eraldiseisev pultidest. Ekraaniga osa saab asetseda mängijate vahel ja mõlemad mängijad saavad liigutada juhtkangidega mängusiseseid reketeid. Mängus töötab avaservi loosimise süsteem juhusliku numbri alusel. Servimisõigust omav mängija saab vaba tahte alusel servida ja mõjutada tehtavat servi juhtkangi abil. Mäng on üles ehitatud selliselt, et mängijatel on võimalus mängu lõppedes uuesti alustada. Mänguks vajaminev kood on koodihoidlas Github aadressil *https://github.com/alovals/arduino-lcd-lauatennis*.

Töös kasutatud riistvara komponentide nimekiri koos hinna ja viitega veebipoele on välja toodud lisas 4.

Järgnevalt kirjeldatakse esimest sammu mängu arendamisel, milleks on vooskeemi loomine.

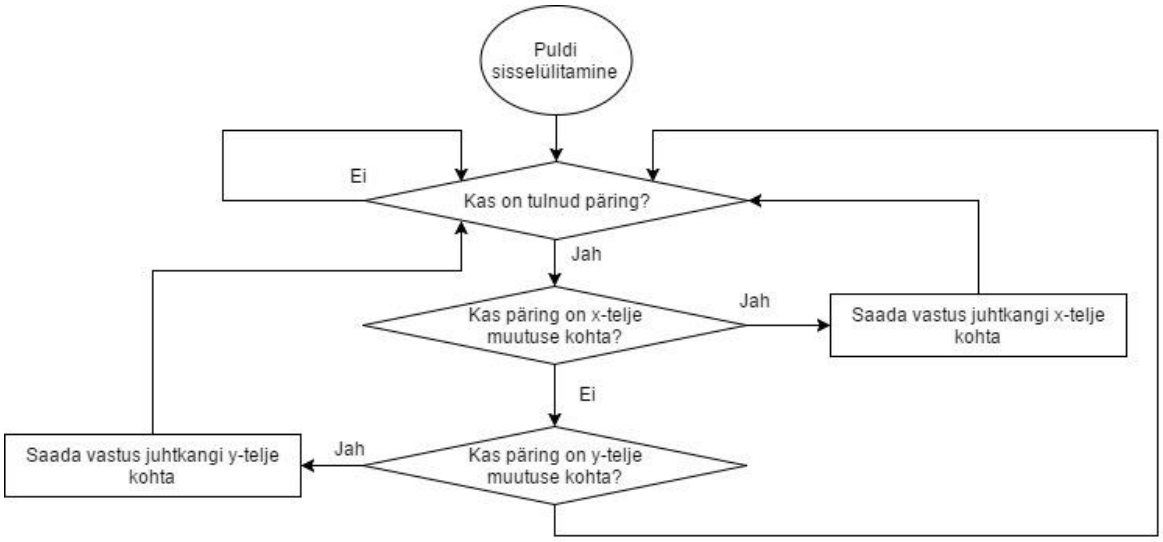
4.1 Tarkvara vooskeem

Enne mängu programmeerimist oleks mõistlik koostada vooskeem programmi loogilise kulgemise kohta, et oleks selgem arusaam eesseisvast arendustööst (vaata joonis 36).



Joonis 36. Mängukonsooli ekraaniga osa programmi vooskeem

Joonisel 36 oleval vooskeemil on märgitud tähtsaimad otsustuskohad programmi täitmise ajal. Ekraaniga osa detailsem vooskeem on välja toodud lisas 1. Käesoleva töö vooskeem saab jaotada kaheks: ekraaniga osa ja puldi vooskeem. Jagunemine on tingitud mitme arendusplaadi olemasolust ning igaühel on oma kindel loogiline töö ülesehitus (vaata joonis 37).



Joonis 37. Mängukonsooli juhtpuldi programmi vooskeem

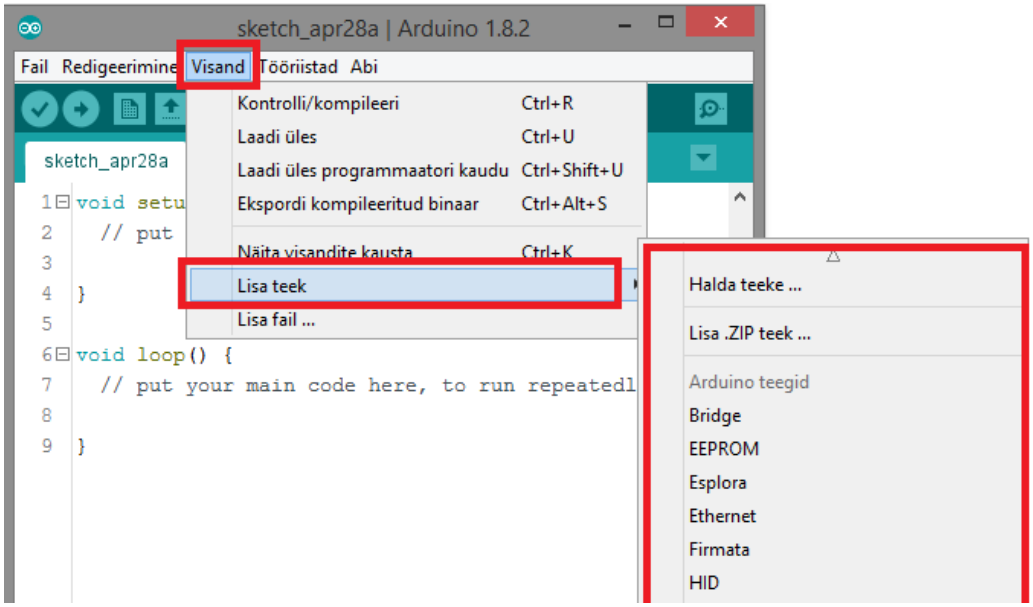
Vooskeemidel ei ole programmi töö lõppseisund tähistatud, kuna programmi töö võib katkeda suvalisel hetkel.

Kui arendaja on selgeks mõtelnud, mida tegema hakatakse, siis võib asuda programmeerima. Järgnevalt kirjeldatakse teegi lisamist ning kasutamist visandis.

4.2 Teekide lisamine ja kasutamine

Järgnevalt selgitatakse kuidas lisada mängu realiseerimiseks vajaminevaid teeke visandisse ja kuidas teeke kasutada. Peatükis 3.3 lisatud teegile kasutatakse veel lisaks teeke *TimedAction* ja *SoftSerial*.

Teegi visandis kasutamiseks tuleb kasutada võtmesõna *#include*, mille järele kirjutatakse kasutatava teegi nimi. Teegi lisamine visandisse on aga lihtsam, kui seda teostada tööriistariba kaudu (vaata joonis 38).



Joonis 38. Teekide lisamine tööriistariba kaudu

Teegi lisamisel tööriistaribalt hoitakse aega kokku, välditakse näpuvigu ja arendaja saab keskenduda olulisele - programmeerimisele. Kui teek on lisatud visandisse, siis on visandis kasutatavad teegis sisalduvad funktsioonid.

Nüüdseks on kirjeldatud kõike vajaminevat, et hakata midagi ekraanile kuvama. Järgmises alapeatükis kirjeldatakse ekraanile erinevate objektide kuvamist.

4.3 Ekraanile kuvamine

Järgnevalt kirjeldatakse ekraanile kuvamise erinevaid viise ning funktsioone kuvamise teostamiseks.

Kasutusmugavuse poole püüeldes on esimeseks sammuks selgesti loetav tekst. Kontrastsuse loomiseks defineeritakse mõned värvid. Värve kirjeldatakse visandis 16-

bitiliste värvikoodidena, näiteks kood 0xFFFF tähistab valget. Teiste värvide värvikoode saab mugavalt järgi vaadata aadressilt *http://www.barth-dev.de/online/rgb565-color-picker/*. Defineerimiseks tuleks kirjutada rida *#define WHITE 0xFFFF*. Peatükis 3.3 toodi välja kood ekraani objekti loomiseks. Kui vastav objekt on loodud, siis saab mugavalt kutsuda välja funktsioonid teegist SPFD5408, mille abil kuvatakse tekst ekraanile:

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

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

Adafruit_TFTLCD tft(LCD_CS, LCD_CD, LCD_WR, LCD_RD, LCD_RESET);
//...
tft.begin(0x9341);              //käivitame ekraani objekti
tft.fillScreen(WHITE);          //tausta värvimine valgeks
tft.setCursor(10, 20);          //kirja vasaku ülemise nurga
                                //punkt
tft.setTextColor(BLACK);        //teksti värviks must
tft.setTextSize(3);
tft.print("Tere!");             //Kuvame teksti ekraanile
```

Mängus kuvatakse lisaks tekstile ka kujundeid. Kujundite kuvamist saab ekraani teegi abil mugavalt lahendada. Liikuvate objektide kuvamine on mõistlik kirjutada mingi funktsiooni kehasse, mille parameetriteks on objekti vana asukoht ja uus asukoht, et kujundi kustutamine ekraanilt oleks tehtav sama funktsiooniga.

```
//...
void liigutaRingi(int vana_x, int vana_y, int uus_x, int uus_y){
```

```
tft.fillCircle(vana_x, vana_y, <raadius>, BLACK);//taustavärv
tft.fillCircle(uus_x, uus_y, <raadius>, WHITE);
}
//...
```

Kui võimalik, siis tasuks objekti liigutada piksel haaval funktsiooni *drawPixel()* abil, sest nii kulutab mikroprotsessor suurte kujundite joonistamisele vähem aega ja mäng on sujuvam.

Menüü loomiseks on teegis *SPFDF5408* funktsioonid *initButton(), drawButton()* ja *contains()*. Nupu loomiseks on vajalik eelnevalt ekraani objekti ja vajutustundlikuse objekti loomine, ning peatükis 3.3 tehtud kalibreerimine, mille tulemusena saadi väärtused TS_MINX, TS_MINY, TS_MAXX ja TS_MAXY. Järgnevalt on ära näidatud nupu funktsioonide kasutamine üldistatud kujul.

```
//nupu objekt
Adafruit_GFX_Button nupp;
//..
void setup()
      //...
      //algväärtustame nupu objekti
      nupp.initButton(&tft, 10, 10, 30, 20, BLACK, WHITE, BLACK,
„nupp“, 3);

//joonistame nupu ekraanile
nupp.drawButton(false)

void loop(){
      //...
```

```
      //kontroll vajutuse tuvastamiseks
If (vajutuskoht.z > MINPRESSURE &&
nupp.contains(vajutuskoht.x, vajutuskoht.y)){
//...
```

Nupu kuvamise kood on liiga pikk ja seda ei tooda käesolevas peatükis välja. Koodi saab uurida lisast 2.

Kuna peatükis 1.4 püstitatud funktsionaalsete nõuete kohaselt saama mängida kaks inimest, siis programmi teatud tööhetkedel on oluline ka ekraani pööramine. Ekraani pööramist saab teostada funktsiooni *setRotation()* abil, mille argumendiks on kas 0, 1, 2 või 3. Iga number tähistab ekraani ühte külge. Katsetades funktsiooni erinevate argumentidega, saab õige pea selgeks, kuidas ekraani vastavalt vajadusele pöörata. Meeles tuleb pidada, et ekraani pööramisel ei pöörata kõigest ekraanil kuvatavat vaid ka ekraani koordinaatteljetikku.

Lisaks teksti ja kujundite kuvamisele saab ekraanile kuvada ka rastergraafika pilte, mis on varasemalt baidimassiiviks teisendatud. Teisendamise protsessi kirjeldati peatükis 3.4. Järgnevalt on välja toodud näide visandi jupist logo kuvamiseks.

```
//programmimälusse kirjutamiseks
#include <avr/pgmspace.h>
//...
const unsigned char ut []PROGMEM ={...} //Img2Code baidimassiiv
//..
void setup(){
      //...
      //logol kindlad mõõtmed!
```

```
tft.drawBitmap(80, 40, ut, 111, 166,BLACK);
//...
```

4.4 Ülesannete ajaline käsitlus

Peatükis 1.4 püstitatud funktsionaalsete nõuete kohaselt peab Arduino suutma teha teatud töö hetkedel ülesandeid, mille kestvus on ajaliselt määratud, kuid ülesande täitmise vältel peab säilima ka vajutustundlikus. Probleemi lahendamiseks kasutatakse teeki TimedAction. Teek on kättesaadav aadressilt *http://playground.arduino.cc/Code/TimedAction#Download*. Enne teegi kasutamist tuleb failis *TimedAction.*cpp muuta rida *#include WProgram.h* reaks *#include Arduino.h*. Antud teek aitab rakendada Arduinol protokollilist täitmist (*protothreading*) [35]. Järgnev koodilõik ilmestab teegi rakendamist.

```
//lisame visandisse vajaliku teegi
#include <TimedAction.h>

//objekt ülesande küsitlemiseks
//ülesannet teostatakse iga 2 sekundi järel
TimedAction ylesanne = TimedAction(2000, ylesande_funktsioon);
//...
void loop(){
     //küsitlus töö kriitilisuse kohta
     ylesanne.check();
//...
```

Pikem ja selgem koodinäide on leitav lisast 3. Protokollilise täitmise tööpõhimõte seisneb pideval küsitlemisel: „Kas mõni ülesanne vajab täitmist?“ [36]. Positiivse vastuse korral asutakse ülesannet lahendama. Seega on teek sobilik, kui programmis on ülesandeid, mis vajavad täitmist iga kindla arvu millisekundite järel. Teek tuleb paigaldada tuginedes peatükis 3.3 kirjutatule.

4.5 Juhtmevaba suhtluse teostus

Käesolevas töös toimub juhtmevaba suhtlus Arduino ja Digispargi arendusplaatide vahel, milleks kasutatakse HC-05 Bluetooth mooduleid. Peatükis 3.5 kirjutatu põhjal peab Bluetooth moodul olema ühendatud jadaporti ehk kasutatakse pesasid TX ja RX, et päringute ja vastuste saatmine oleks võimalik. Digispargil paraku puuduvad jadapordi klemmid, mille tõttu on tingitud teegi *SoftSerial* kasutamine. Antud teegi abil saab emuleerida jadapordi olemasolu ja vastu võtta päringuid ning nendele vastata. Järgnevas koodinäites on ära toodud jadapordi objekti loomine.

```
#include <SoftSerial.h>
int digiRx = 1;
int digiTx = 2;
SoftSerial BtSerial(digiRx, digiTx);
//::
```

Loodud jadapordi objekti saab kasutada nagu tavalist jadaporti. Emuleerimise korral peab arvestama, et suhtlus seadmete vahel ei ole nii kiire, kui kasutada riistvara poolt pakutavaid jadapordi pesasid. Teek *SoftSerial* paigaldatakse peatükis 3.5 tehtavate seadistuste käigus ja eraldi paigaldus ei ole vajalik.

Kokkuvõte

Käesoleva bakalaureusetöö põhieesmärgiks oli koostada eestikeelne juhend värvilise vajutustundliku ekraani kasutamiseks. Arendustöö vältel analüüsiti ja dokumenteeriti nimetatud ekraani kasutatavust Arduino Mega arendusplaadiga. Bakalaureusetöö lõpuks jõuti terviklahenduseni, milleks on juhtmevabade pultidega juhitav Arduino LCD Lauatennise mäng.

Lõputöös valminud juhendi järgi on algaja huviline võimeline programmeerima ka teisi sarnaseid mänge, sest teadmised ekraani ja muu riistvara programmerimise kohta on samad. Muuta tuleb ainult mängu loogilist ülesehitust.

Eesmärgi täitmise muutis keeruliseks varasema kogemuse puudumine riistvara komponentidega ning nende programmeerimisega. Lahenduse välja töötamiseks tuli läbida hulgaliselt inglisekeelseid juhendeid ja proovida erinevad lähenemisi probleemi lahendamiseks. Töö autor õppis bakalaureusetöö tegemisel riistvara programmeerimist ja käsitlemist. Töö autor loodab, et käesolevast tööst on abi õpitubade läbiviimisel ja noortele valdkonna tutvustamisel.

Kasutatud materjalid

[1] Eesti Rakendusuuringute Keskus CentAR, „Tudengite õpingute katkestamise põhj used IKT kõrghariduses“ http://www.centar.ee/uus/wp-ontent/uploads/2015/07/IKT-katkestajate-uuringu-l%C3%B5ppraport-veebi.pdf (01.01.2017)

[2] M. Kadri, „Info- ja kommunikatsioonitehnoloogia erialade tudengite õpingud kõrgkoolis: esimesel aastal väljalangemine ja õpingute jätkamine“ http://193.40.4.3/handle/10062/53218 (01.01.2017)

[3] The English PingPong Association, „History of Ping Pong“ http://www.pingpongengland.co.uk/about_ping_pong/history-of-ping-pong/ (03.03.2017)

[4] Eesti Lauatenniseliit, „Regulatsioonid“ http://www.lauatennis.ee/web/node/17 (03.03.2017)

[5] Arduino, „What is Arduino?“ https://www.arduino.cc/en/Guide/Introduction (12.01.2017)

[6] Codeduino, „Arduino vs Raspberry Pi Comparsion“ https://codeduino.com/tutorials/arduino-vs-raspberry-pi/ (12.01.2017)

[7] Tested, „Know Your Arduino“ http://www.tested.com/tech/robots/456466-know-your-arduino-guide-most-common-boards/ (12.01.2017)

[8] Arduino, „Compare board specs“ https://www.arduino.cc/en/Products/Compare (14.02.2017)

[9] Diffen, „Analog vs. Digital“ http://www.diffen.com/difference/Analog_vs_Digital (14.02.2017)

[10] Arduino, „Arduino Mega 2560“ http://www.mantech.co.za/datasheets/products/A000047.pdf (14.02.2017)

[11] Miscellaneous Web Stuff, „Touch Screen Shield for Arduino UNO“ http://misc.ws/2013/11/08/touch-screen-shield-for-arduino-uno/ (14.02.2017)

[12] Cyan Infinite, „2.4″ Touchscreen LCD Shield“ http://cyaninfinite.com/tutorials/2-4-touchscreen-lcd-shield/ (14.02.2017)

[13] Yaara Lancet, „What Are the Differences Between Capacitive & Resistive Touchs creens?“ http://www.makeuseof.com/tag/differences-capacitive-resistive-touchscreens-si/ (14.02.2017)

[14] Digistump, „Digispark USB Development board“ http://digistump.com/products/1 (14.02.2017)

[15] Digistump Wiki, „Quick Reference“ http://digistump.com/wiki/digispark/quickref (14.02.2017)

[16] Open Electronis, „Top 5 Wireless Ways to Communicate with your Controller“ https://www.open-electronics.org/top-5-wireless-ways-to-communicate-with-your-controller/ (14.02.2017)

[17] Arduino Info, „BlueTooth-HC05-Modules-How-To“ https://arduino-info.wikispaces.com/BlueTooth-HC05-HC06-Modules-How-To (14.02.2017)

[18] Future Electronics, „What is a buzzer?“ http://www.futureelectronics.com/en/passives/buzzers.aspx (14.02.2017)

[19] Sunfounder Maker Education, „Lession 3 Buzzer“ https://www.sunfounder.com/learn/universal-kit-for-Arduino/lesson-3-buzzer-universal-kit.html (14.02.2017)

[20] Sunfounder Maker Education, „Lesson 12 Buzzer“ https://www.sunfounder.com/learn/sensor_kit_v1_for_Arduino/lesson-12-buzzer-sensor-kit-v1-for-arduino.html (14.02.2017)

[21] Arduino, „Interfacing a Joystick“ https://www.arduino.cc/en/Tutorial/JoyStick (15.01.2017)

[22] Oomipood, „Liuglüliti ON-ON“ https://www.oomipood.ee/product/10170_gby_liugluliti_on_on_spdt_100ma_19_5_11_4mm_s103 (15.01.2017)

[23] Colorovo, „Colorovo PowerBox Slim 300“ http://www.colorovo.eu/products-en/portable-chargers-en/colorovo-powerbox-slim-3000.html?product_id=2464 (15.01.2017)

[24] 1000Bulbs, „Panasonic - 9V Size - Alkaline Battery“ https://www.1000bulbs.com/product/4504/PAN-C37879V.html (15.01.2017)

[25] Arduino, „Arduino/Processing Language Comparision“ https://www.arduino.cc/en/Reference/Comparison (13.01.2017)

[26] Wiring, „Cover“ http://wiring.org.co/ (14.01.2017)

[27] Arduino, „setup()“ https://www.arduino.cc/en/Reference/Setup (14.01.2017)

[28] Arduino, „loop()“ https://www.arduino.cc/en/reference/loop (14.01.2017)

[29] FreeTDS User Guide, „What is a C library?“ http://www.freetds.org/userguide/rtl.define.library.htm (14.01.2017)

[30] Instructables, „Modify the HC-05 Bluetooth Module Defaults Using AT Commands“ http://www.instructables.com/id/Modify-The-HC-05-Bluetooth-Module-Defaults-Using-A/ (14.01.2017)

[31] Dejan Nedelkovski, „How To Configure and Pair Two HC 05 Bluetooth Modules as Master and Slave | AT Commands“ http://howtomechatronics.com/tutorials/arduino/how-to-configure-pair-two-hc-05-bluetooth-module-master-slave-commands/ (14.01.2017)

[32] Nick Koumaris, „Arduino Bitmap Graphics Tutorial“ http://educ8s.tv/arduino-bitmap-graphics-tutorial/ (14.01.2017)

[33] Kalle Toompere, „Tartu Ülikooli tunnusgraafika“ http://www.ut.ee/sites/default/files/ut_files/b9dcbf563c0c53f2c5516f7103a48ebb.pdf (14.01.2017)

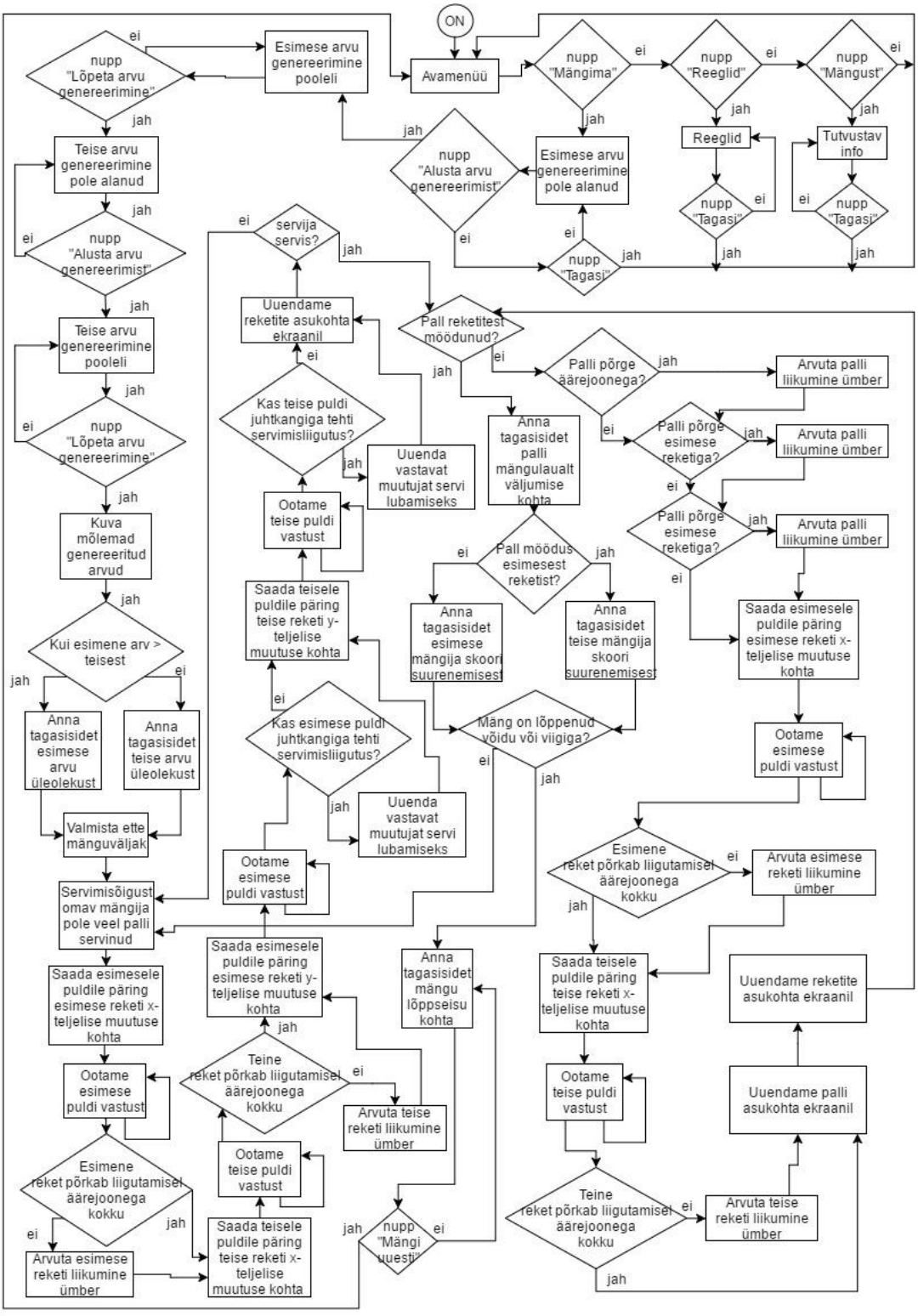
[34] Digistump Wiki, „Connecting and Programming Your Digispark“ https://digistump.com/wiki/digispark/tutorials/connecting (14.01.2017)

[35] Arduino, „TimedAction“, http://playground.arduino.cc/Code/TimedAction#Download (14.01.2017)

[36] Adam Dunkels, „About Protothreads“ *http://dunkels.com/adam/pt/about.html* (14.01.2017)

Lisad

I. Ekraaniga komponendi üldisem vooskeem



II. Vajutustundliku nupu rakendamiseks vajalik kood

```
//teekide lisamine visandisse
#include <SPFD5408_Adafruit_GFX.h>
#include <SPFD5408_Adafruit_TFTLCD.h>
#include <SPFD5408_TouchScreen.h>

//kalibreerimise näidisvisandist
#define YP A1
#define XM A2
#define YM 5
#define XP 6
#define SENSIBILITY 300

//vajutustundlikkuse objekt
TouchScreen ts = TouchScreen(XP, YP, XM, YM, SENSIBILITY);

//kalibreerimisel saadud väärtused
#define TS_MINX 133
#define TS_MINY 139
#define TS_MAXX 900
#define TS_MAXY 920

//kalibreerimise näidisvisandist
#define LCD_CS A3
#define LCD_CD A2
#define LCD_WR A1
#define LCD_RD A0
#define LCD_RESET A4

//ekraani objekt
Adafruit_TFTLCD tft (LCD_CS, LCD_CD, LCD_WR, LCD_RD, LCD_RESET);

//minimaalne surve ekraanile võimaliku müra eemaldamiseks
#define MINPRESSURE 3

//objekt sisaldab vajutuse x, y ja z(surve) väärtusi
```

```
TSPoint vajutuskoht;

//muutujad ekraani mõõtmete hoidmiseks
int laius;
int korgus;

//nupu objekt
Adafruit_GFX_Button nupp;

void setup(){

 //alustame ekraani tööga
      tft.begin(0x9341);

 //päring ekraani objektile ekraani laiuse kohta
 laius = tft.width() - 1;

      //päring ekraani objektile ekraani korguse kohta
 korgus = tft.height() - 1;

 //algväärtustame nupu objekti
 nupp.initButton(&tft,  10,  10,  30,  20,  BLACK,  WHITE,  BLACK,
„nupp“, 3);

      //joonistame nupu ekraanile
      nupp.drawButton(false)
}
void loop(){

 //iga iteratsioon uuendame vajutuspunkti
 vajutusFunk();

 //kontroll vajutuse tuvastamiseks
      If (vajutuskoht.z > MINPRESSURE &&
      nupp.contains(vajutuskoht.x, vajutuskoht.y)){
       //nuppu vajutati
       //siia programmi reaktsioon vajutusele
```

```
      }
}
//funktsioon vajutuskoha töötlemiseks
void vajutusFunk(){

 //päring vajutustundlikkuse objektile vajutuse kohta
      vajutuskoht = ts.getPoint();

      //TFT elektriliseks juhtimiseks ptk. 2.2.1
      pinMode(XM, OUTPUT);
      pinMode(YP, OUTPUT);

      //vahemuutujad algsete väärtuste säilitamiseks
      int x_koordinaat = vajutuskoht.x;
      int y_koordinaat = vajutuskoht.y;

      //vajutuse koordinaatide töötlus selliselt, et oleks
      //kooskõla peatükis 1.3 joonis 1. välja toodud teljestikuga
      vajutuskoht.x = map(y_koordinaat, TS_MINY, TS_MAXY, 0,
laius);
      vajutuskoht.y = map(x_koordinaat, TS_MINX, TS_MAXX, korgus,
0);
}
```

III. Ülesannete protokolliline täitmine

```
//lisame visandisse vajaliku teegi
#include <TimedAction.h>

//kalibreerimise näidisvisandist
#define LCD_CS A3
#define LCD_CD A2
#define LCD_WR A1
#define LCD_RD A0
#define LCD_RESET A4

//ekraani objekt
Adafruit_TFTLCD tft (LCD_CS, LCD_CD, LCD_WR, LCD_RD, LCD_RESET);

//objekt ülesande küsitlemiseks
TimedAction pooraEkraani = TimedAction(2000, poora);

void setup(){

 //alustame ekraani tööga
      tft.begin(0x9341);

      //ekraani vaatamisnurk selliselt laiem
      tft.setRotation(1);
}
void loop(){

      //küsitlus töö kriitilisuse kohta
 pooraEkraani.check();
}
void poora(){

 //pöörame ekraani pahupidi
 if (tft.getRotation == 1){
```

```
 //kuvatud tekst tuleb kustutada
 title(BLACK);

          //pööramine
 tft.setRotation(3);

 //kuvatakse tekst pööratult
 title(WHITE);
 }

     //pöörame ekraani tagasi, kui on pahupidi algpositsiooni
     //suhtes
 else{
 title(BLACK);
 tft.setRotation(1);
 title(WHITE);
 }
}
```

IV. Riistvara komponentide nimekiri

(vajaminev kogus)	Maksumus Eestist ostes	Maksumus välismaalt ostes
Arduino Mega (1tk)	47 € http://bit.ly/2q6PGdO	9,18 € http://ebay.to/2pCVWaw
Vajutustundliku ekraani moodul (1tk)	-	5,59 € http://bit.ly/2qXMwuc
Analoog juhtkangi moodul (2tk)	6,2 € http://bit.ly/2q8JLDt	1,3 € http://bit.ly/2r3JKQC
Bluetooth moodul HC-05 (4tk)	12 € http://bit.ly/2pBh7cr	3,98 € http://bit.ly/2qvGr8i
Digispark Attiny85 (2tk)	4,5 € http://bit.ly/2q60Z6p	1,31 € http://ebay.to/2qXFWDY
Akupank Colorovo PowerBox 3000 (1tk)	10,9 € http://bit.ly/2qXNM0c	14 € http://ebay.to/2q8p3DN
Passiivne sumisti (1tk)	-	1,82 € http://ebay.to/2q5Pk7w
Liuglüliti ON-ON (4tk)	0,5 € http://bit.ly/2qvIbym	0,82 € (5tk) http://bit.ly/2pBxhST

V. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, **Alo Vals**,

(*autori nimi*)

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose
Arduino LCD Lauatennis,
(*lõputöö pealkiri*)

mille juhendaja on Alo Peets, Anne Villems, Taavi Duvin,

(*juhendaja nimi*)

   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**