Arduino Roboti ehitamine

Tellija: **HITSA**
Koostaja: **Robotimeister OÜ** (Alo Peets)
Toetaja: **Tartu Ülikool**, arvutiteaduse instituut

Kokkuvõttev lühikirjeldus:
Õppematerjali eesmärk on õppida tundma arendusplaati Arduino, selle tarkvara arenduskeskonda Arduino IDE ja eirnevaid elektroonikakomponente ja laiendusplaate. Lisaks niisama katsetamisele saab materjalide lõpuks valmis ehitada ka põneva anduritega roboti. Täiendavaks eeliseks on Arduino roboti 2x odavam hind Raspberry Pi robotiga võrreldes ~50 EUR.

Õpitulemus:
Materjali läbinu on võimeline iseseisvalt ehitama Arduino roboti

Maht: 1,5 EAP

Õppematerjal on suunatud järgmistele aineõpetajatele:
informaatika (programmeerimine), matemaatika (arvutused), füüsika (elektroonika, mehaanika)

Sisukirjeldus:
Õppematerjalide/koolituse läbinu teab ja oskab kasutada:

- sobiva Arduino või selle derivaadi valik (näiteks Idead Studio - võimekam ja odavam)
- Arduino arendusplaat
- Arduino IDE arenduskeskkond
- Näidiskoode käivitada
- RGB LEDi vilgutada
- lülitit kasutada
- Transistori ühendada ja valida
- ADC kasutada (valguse mõõtmiseks)
- Mootoreid juhtida
- joystick arduinoga ühendada
- kaugust mõõta ultraheli anduriga
- elektroonikakomponente ühendada
- Jadaliidese kaudu andmeid arvuti ja Arduino vahel liigutada
- erinevate laiendusplaatide (shield) sobivust hinnata ja neid ühendada
- 16x02 tähtekraani kasutada
- infrapuna LED saatja-vastuvõtja abil roboti juhtimine
- kõike eelpool mainitud omavahel koos toimima panna
- kõigest eelnevast 50 EUR maksva roboti ehitada ja selle intelligentselt sõitma panna.
- kõige eelneva juures C-programmeerimiskeel ja tervikkoodis kasutamine (IF; FOR, SWITCH, tekststitöötlus jne)

Täiendavalt kaetakse elektroonika põhitõdesid pinge/vool/takistus volt/amper/oom ja õpetatakse multimeetrit käsitlema. Koolitusel räägitakse juurde tausta, seletatakse täpsemalt riistvara võimalusi ja omandatakse professionaalse juhendajate abiga kõik õpitulemused ning valmistatakse õpetajad analoogseid ülesandeid lastega koolis läbi tegema.

HITSA Hariduse Infotehnoloogia Sihtasutus
ProgeTiiger
Euroopa Liit Euroopa Sotsiaalfond
Eesti tuleviku heaks
TARTU ÜLIKOOL arvutiteaduse instituut 1632

Sisukord

1 Arduino võimalused ... 4
1.1 Platvormi tutvustus ... 4
1.2 Arduino ajalugu ja populaarsus ... 6
1.3 Teiste tootjate Arduinol põhinevad plaadid ... 7
1.4 Erinevad Arduino tooted ... 8
1.4.1 Arduino arendusplaadid ... 9
1.4.2 Laiendusplaadid ... 12
1.4.3 Arduinol põhinevad komplektid ... 13
1.5 Arduino programmeerimine ... 15
1.6 Võrdlus Raspberry Pi-ga ... 16
2 Stardikomplekti sisu ja seadistamine ... 19
2.1 Stardikomplekti sisu ... 19
2.1.1 Kõikides projektides kasutatavad komponendid ... 21
2.1.2 Valgusdioodid ... 22
2.1.3 Takistid ... 23
2.1.4 Lülitid ... 24
2.1.5 Projekti juhtimiseks vajalikud tarvikud ... 24
2.1.6 Mootorid ... 25
2.1.7 Erkaanid ja näidikud ... 27
2.1.8 Keskkonnale reageerivad andurid ... 28
2.1.9 Teised komponendid ... 30
2.2 Arenduskeskkonna paigaldamine ja arendusplaadi ühendamine ... 34
2.3 Koodi kirjutamine ja plaadile laadimine ... 38
2.4 Maketeerimislaua ja protoplaadi kasutamine ... 40
3 Õppematerjalid ... 42
3.1 Erinevad võimalused LED vilgutamiseks ja FOR-tükkel ... 42
3.2 Lüliti ühendamine ja LEDi reageerimine vastavalt lüliti asendile vooluringis ja digitaalsignaali abil IF-lausega ... 46
3.3 RGB ledi ühendamine ja RGB koodide katsetamine analogWrite abil ... 49
3.4 Kolme nupu abil RGB LEDi lülitamine ... 52
3.5 Ekraani ühendamine ja teksti kuvamine 1602 LCD ekraanil ... 54
3.6 Helianduri ühendamine, LEDide vilgutamine vastavalt müratugevusele ... 58
3.7 Fototakisti abil valguse mõõtmine, Serial monitoriga tutvumine, valgustundlik LED 61

3.8 Juhtkangi ühendamine ja mootorite juhtimine 64
3.9 Infrapuna vastuvõtja, pult ja 7-segmendiline ühe numbriga LED 68
3.10 Temperatuuri kuvamine 4-kohalisel 7-segmendilisel näidikul 73
4 Arduino roboti ehitamine 78
4.1 Arduino ühendamine mootoritega 78
4.2 Mootorite liigutamine 82
4.3 Infrapuna sensorite ühendamine Arduinoga 84
4.4 Infrapuna sensorite abil roboti juhtimine 85
4.5 Ultraheli sensor 88
4.6 Laadimispunkti liikumise algoritm 89
4.7 Roboti ehitamine 90
4.8 Roboti ehitamiseks kasutatud asjade nimekiri ning soetuskoht: 92

1 Arduino võimalused

Käesolevas peatükis tutvustatakse Arduino arendusplaate, võrreldakse erinevaid Arduino komplekte, nende kasutusvõimalusi ja ajalugu. Tuuakse põhjendusi, miks antud platvorm on sobilik koolirobootikas kasutamiseks ning milliseid projekte on sellega võimalik teha. Kirjeldatakse ka Arduino eeliseid ja puudusi võrreldes teiste platvormidega. Samuti tutvustatakse Arduino programmide kirjutamiseks enim kasutatavat tarkvara - Arduino arenduskeskkonda.

1.1 Platvormi tutvustus

Arduino on avatud lähtekoodi ja -disainiga mikrokontrollerplatvorm elektroonika prototüüpimiseks. Arduinole saab külge ühendada erinevaid andureid (termo-, valgus-, vibratsiooni-, heli- jt andurid). Andurite abil saab Arduino sisendi ning edasi reageerib mikrokontroller vastavalt koodile ning käivitab mõne väljundseadme (mootor, valgusdiood, ekraan jne.

Arduinoga programmeerimiseks on algajatel kõige lihtsam kasutada Arduino integreeritud arenduskeskkonda (IDE - *Integrated development environment*) ning Arduino programmeerimiskeelt, mis tugineb *Wiring* keskkonnale. Arduino programmeerimiskeel baseerub C/C++ programmeerimiskeelele. Kompileerimisel tehakse koodis väikesed muudatused ning antakse edasi C/C++ (täpsemalt avr-g++) kompilaatorile. Erinevalt C-programmeerimiskeeles kirjutatud programmist tuleb töötava programmi jaoks kasutajal defineerida kaks funktsiooni: *setup()*, mis kutsutakse välja üks kord programmi alguses ja millega seadistatakse parameetrite algväärtused, ning *loop()*, mis kutsutakse korduvalt välja ja mis võimaldab oludel muutuda ja programmil vastavalt reageerida.

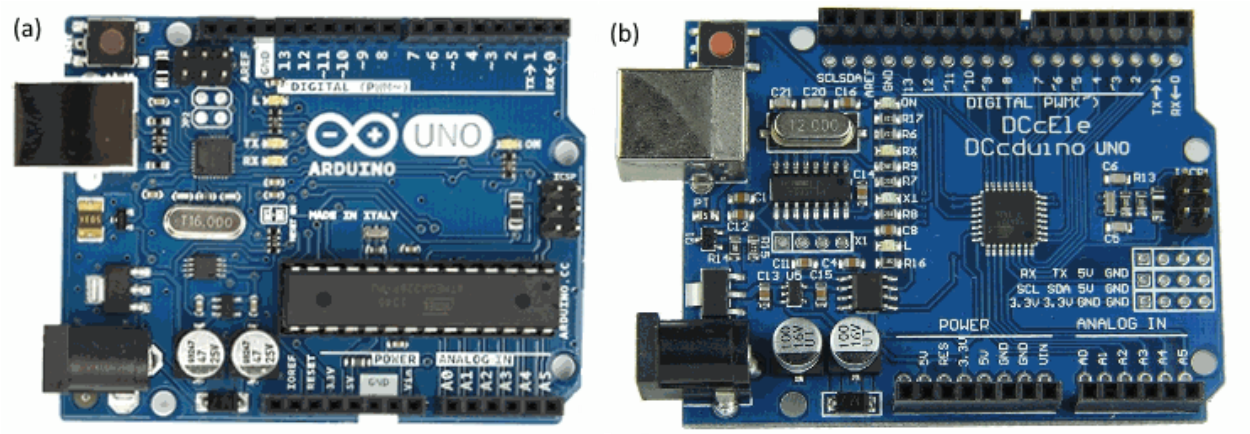
Arduino kaubamärgi all müüakse tooteid ainult Ameerika Ühendriikides. Teistes riikides müüakse samu tooteid Genuino nime all. Genuino nime all müüakse tooteid teistes riikides, kuna üks asutajatest Gianluca Martino registreeris Arduino nime Itaalias teiste asutajate teadmata. Jõudmata omavahel kokkuleppele Arduino kaubamärgi kasutamise üle Itaalias, kus Arduino on väga populaarne, lõpetati koostöö Gianlucaga ning loodi uus kaubamärk

Genuino, et saaks ka väljaspool Ameerika Ühendriike tooteid müüa. Lisaks Arduinole ja Genuinole esineb ka suur hulk võltsinguid, derivaate ja kloone. Kuna Arduino on avatud lähtedisainiga, on lubatud toota sarnase disaini ja võimalustega tooteid, kuid mitte Arduino kaubamärgi all. Võltsingud müüvad tihti ennast Arduino nime all ning kasutavad Arduino logo, kuid kvaliteet on tavaliselt madalam (vt joonis 1). Võltsingud on keelatud ning lähevad vastuollu autoriõigustega. Arduino tootja üritab neid tuvastada, et inimestele ei müüdaks praaktooteid Arduino nime all.

Originaal Võltsing Võltsing Võltsing

Joonis 1. Märk arendusplaadi tagaküljel Arduino originaaltootel ja võltsingutel

Derivaadid põhinevad Arduino toodetel, kuid neil on tihtipeale natuke erinev paigutus ja tunnused. Need tooted aitavad koguda Arduino toodetel populaarsust. Kloonid on tavaliselt väga sarnased päris Arduino toodetega, kuid müüvad end teise nime all (tavaliselt sisaldub nimes "Ardu" või "duino"). Käesolevas töös kasutatakse Arduino klooni DCcEle DccDuino UNO plaati (vt joonis 2 (b)). DCcEle DccDuino UNO on umbes neli korda odavam kui Arduino UNO ja Arduino Idead Studio derivaati Iteaduino UNO, mis omab täiendavaid väljaviike roboti mugavamaks ehitamiseks ja ühendamiseks.



Joonis 2. Arduino UNO (a) ja DCcEle DCcduino UNO (b)

Käesoleva töö teoreetiline osa põhineb Arduino originaaltootel, kuna on kõige populaarsem ning teiste versioonide aluseks olev tehnoloogia. Enamik analüüsi ja õpetusi on kehtivad ka kloonide ja derivaatide korral.

Arduino on sobiv platvorm nii algajatele kui ka kogenenud kasutajatele. Arduino IDE-ga on kaasas palju näidisprogramme, et alustamine oleks võimalikult lihtne. Samas on Arduino programmeerimiskeel paindlik, kuna sellega saab kogenenud kasutaja teha ka keerulisemaid projekte, kombineerides või muutes erinevaid näiteprogramme ning lisades väliseid teeke.

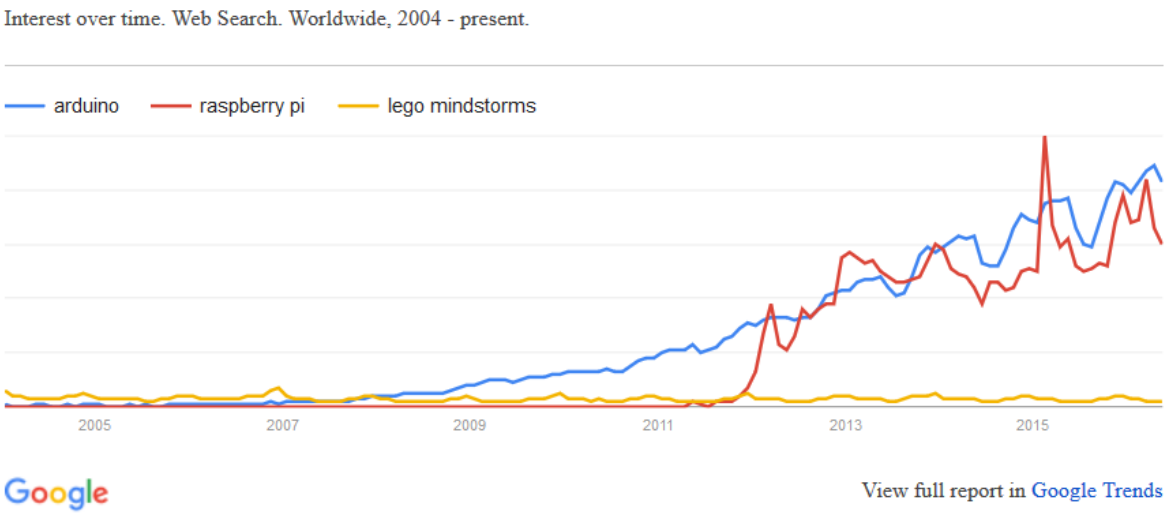
1.2 Arduino ajalugu ja populaarsus

Arduino asutati 2005. aastal Itaalias Ivera linnas ning firma põhiidee sai alguse vajadusest piisava võimsuse ja mõistlikku hinnaga arendusplaadi järele, mille abil õpilased saaksid luua elektroonikaprojekte. Arduino arendamise põhitiimi kuulusid David Cuartielles, Gianluca Martino, Tom Igoe, David Mellis ja Massimo Banzi.

Ivrea interaktsioondisaini instituudis *(Interaction Design Institute Ivrea* ehk IDII*)*, kus üks Adruino loojatest, M. Banzi, õppjõuna töötas, oli sel hetkel kasutusel mikrokontroller BASIC Stamp, mis oli selleks hetkeks juba pea kümnendi kasutusel olnud ning jäi ka hinna poolest õpilastele üsna kalliks. Põhiline puudus seisnes siiski selles, et BASIC Stamp-i jõudlusest jäi soovitud mõtete ellu viimiseks väheks. See ajendaski Banzi võtmaks ühendust kolleegiga Massachusettsi Tehnoloogiainstituudist *(Massachusetts Institute of Technology)*, kellel oli kogemus disainerisõbraliku programmeerimiskeele Processing arendamisega, tekkis idee luua sarnane tarkvara ka mikrokontrollerite programmeerimiseks.

Üks IDII tudengitest, Hernando Barragán, alustas soovitud platvormi arendamisega, tulemuseks oli Wiring, mis sisaldas nii arendusplaati kui ka arenduskeskkonda. Banzi võttis Wiringist küll eeskuju, kuid Arduino lõpplahendus oli veelgi intuitiivsem ja odavam, makstes ilmumise hetkel 30 dollarit. Kuna eesmärk oli pakkuda odavat, kiiresti ja võimalikult paljude inimesteni jõudvat toodet, siis otsustati avatud lähtekoodi kasuks. Riistvara on litsentsitud *Creative Commonsi Share-Alike 2.5* litsentsiga, mis lubab kasutamise nii isiklikel kui ka kaubanduslikel eesmärkidel, tingimusel, et järgi tehtud toode omab vabavaralist litsentsi ning on märgitud, et eeskuju on võetud Arduinost.

Innovaatiline lahendus jõudis kiiresti massidesse ning 2011 aastaks oli maailmas müüdud juba üle 250 000 arendusplaadi. Täpseid müüginumbreid on hetkel raske öelda, kuna vabavaralise riistvara tõttu on müügil palju Arduinol põhinevaid arendusplaate. Inimeste huvist Arduino vastu annab aimu Google otsingutulemuste statistika aastast 2004 kuni mai 2016 (Joonis 3). Graafikul on võrdluseks välja toodud ka koolirobootika peamiste konkurentide Raspberry Pi ja Lego Mindstorms otsinute statistika.



Joonis 3. Google Otsingutulemuste statistika Arduino kohta

Joonisel kujutatud info põhjal võib järeldada, et Arduino ei olnud vaid moetoode ning selle populaarsus maailmas jätkuvalt kasvab. Graafikul olevad langused jäävad suvekuudele, kui enamustes maailma koolides on vaheaeg, mis omakorda näitab haridusasutuste tõusvat huvi robootika õppetöösse kaasamise vastu.

1.3 Teiste tootjate Arduinol põhinevad plaadid

Genuino on Arduino tütarfirma, mis loodi Arduino asutajate Massimo Banzi, David Cuartielles, Tom Igoe ja David Mellis poolt. Plaadid on komponentide ja omaduste poolest identsed Arduinole, ainsaks erinevuseks ongi kaubamärgi nimi ja turunduspiirkond. Nimelt turustatakse Genuinot Euroopas, Aasias, Lõuna- Ameerikas, Aafrikas ja Kanadas, seevastu Arduino peamiseks sihtturuks on Ameerika Ühendriigid.

Joonis 4. Arduino UNO ja Genuino UNO arendusplaadid

Lisaks Genuinole, mille kvaliteet on võrdväärne Arduinoga, on müügil ka erinevate tootjate Arduinoga tarkvaraga ühilduvaid plaate. Mõned neist on oma plaadi arendamisel võtnud Arduinost eeskuju ning lisanud ka omaltpoolt erinevaid funktsionaalsusi - selliseid plaate nimetatakse derivaatideks. Samas on ka neid tootjaid, kes ise lisafunktsionaalsust lisamata kopeerivad Arduino lahendusi, kuid omavad vähemalt oma kaubamärki – neid kutsutakse kloonideks. Paraku ringleb turul ka järgi tehtud Arduino märgisega tooteid ehk võltsinguid. Avatud litsentsiga riistvara tõttu on derivaadid ja kloonid täiesti legaalsed, kuid võltsingud on keelatud.

Hiinast tellides saab Arduinoga samaväärse funktsionaalsusega plaadi vaid mõne dollariga, kuid peab arvestama võimalusega, et toode on kehvema kvaliteediga kui originaal ning suurema ohuga, et jõuab kohale defektiga. Lisaks tuleb mõelda ka moraalsetele küsimustele, mis sõltuvad konkreetsest tootjast, kas firma panustab ka klienditoele ja tehnoloogiate arendamisele ega tootmisel poleks kasutatud lapstööjõudu.

1.4 Erinevad Arduino tooted

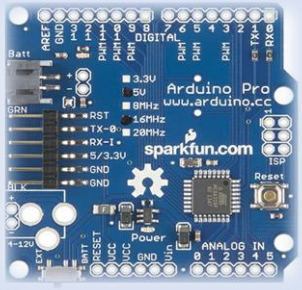
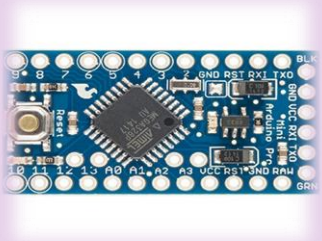
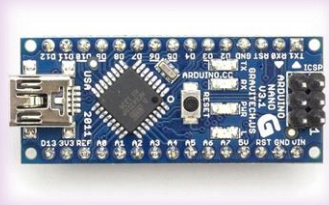
Käesolevas punktis antakse ülevaade enim kasutatavatest Arduino arendus – ja laiendusplaatide mudelitest ning nende võimsusest, funktsionaalsusest ning kasutusvõimalustest. Tutvustatakse ka robootikaga alustamiseks kõige mugavamat võimalust – stardikomplekti ning tuuakse välja võrdlus esindusliku originaalkomplekti ja odavama Arduino kloonil põhineva komplekti vahel ning ise valida sobivad komponendid

1.4.1 Arduino arendusplaadid

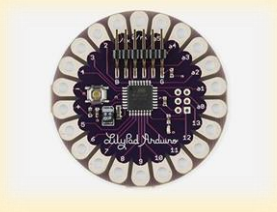
Arduino arendusplaatidest on saadaval palju erinevaid mudeleid, mille abil on võimalik koostada erineva eesmärgi ja raskusastmega projekte. Mõned populaarsemad arendusplaatide mudelid ja nende info on toodud välja tabelis 1. Arduinoga alustajatele on sobilikud odavama hinnaklassiga ning lihtsama disainiga UNO, 101, Pro (tabelis mängitud sinise taustaga) ja neile võimsuselt sarnased, kuid mõõtudelt väiksemad Mini, Micro ja Nano (tabelis märgitud lillaka taustaga). Keerukamate projektide jaoks on sobilikud Mega, Zero ja Due (tabelis märgitud punaka tasutaga), mis võimaldavad suuremat taktisagedust, ühilduvust keerulisemate seadmetega, omavad rohkem välkmälu, mis on vajalik koodi hoidmiseks ning sisaldavad rohkem analoog- ja digitaalpesasid. Ühendatud seadmetega seotud projektide (Internet of Things) jaoks on parim valik kahe protsessori, Wifi mooduli, USB-A pordi, micro SD kaardi pesa ja Linuxi operatsioonisüsteemiga Yún (tabelis märgitud rohelise taustaga). Tekstiilitöötluse projektide jaoks on saadaval mõõtmetelt väikesed ja kaalult kerged Gemma ja erinevad versioonid enim moekunstis kasutatavatest riiete sisse õmmeldavast arendusplaadist LilyPad (tabelis märgitud kollase taustaga).

Tabel 1. Arduino populaarsemad arendusplaadid

Arendusplaat	Mikrokontrolleri mudel	Üldandmed	Joonis
Arduino UNO	ATmega328P	Tööpinge: 5V Soovitatav sisendpinge: 7-12V Lubatud sisendpinge: 6-20V Taktisagedus: 16MHz Välkmälu: 32 KB	 **Joonis** 5. Arduino UNO
Arduino 101	Intel Curie	Tööpinge: 3.3V Soovitatav sisendpinge: 7-12V Lubatud sisendpinge: 7-20V Taktisagedus: 32MHz Välkmälu: 196 kB	 Joonis 6. Arduino 101

Arduino Pro	ATmega168 või ATmega328	Tööpinge: 3.3V Lubatud sisendpinge: 3.35 -12V Taktisagedus: 8 MHz Välkmälu: 16 KB või Tööpinge: 5 V Lubatud sisendpinge: 5 - 12 V Taktisagedus: 16 MHz Välkmälu: 32KB	 Joonis 7. Arduino Pro
Arduino Pro Mini	ATmega328	Tööpinge: 3.3V Lubatud sisendpinge: 3.35 -12 V Taktisagedus: 8 MHz Välkmälu: 32 kB või Tööpinge: 5V Lubatud sisendpinge: 5 - 12 V Taktisagedus: 16 MHz Välkmälu: 32 kB	 Joonis 8. Arduino Pro Mini
Arduino Micro	ATmega32U4	Tööpinge: 5V Soovitatav sisendpinge: 7-12V Lubatud sisendpinge: 6-20V Taktisagedus: 16MHz Välkmälu: 32 KB	 Joonis 9. Arduino Micro
Arduino Nano	Atmel ATmega168 või ATmega328	Tööpinge: 5V Soovitatav sisendpinge: 7-12V Lubatud sisendpinge: 6-20V Taktisagedus: 16MHz Välkmälu: 16 KB (ATmega168) või 32 KB (ATmega328)	 Joonis 10. Arduino Nano

Arduino Mega 2560	ATmega2560	Tööpinge: 5V Soovitatav sisendpinge: 7-12V Lubatud sisendpinge: 6-20V Taktisagedus: 16MHz Välkmälu: 256 KB	 Joonis 11. Arduino Mega 2560
Arduino Zero	ATSAMD21G18, 32-Bit ARM Cortex M0+	Tööpinge: 3.3V Soovitatav sisendpinge: 7-12V Lubatud sisendpinge: 6-20V Taktisagedus: 48 MHz Välkmälu: 256 KB	 Joonis 12. Arduino Zero
Arduino Due	AT91SAM3X8E	Tööpinge: 3.3V Soovitatav sisendpinge:7-12V Lubatud sisendpinge: 6-20V Taktisagedus: 84 MHz Välkmälu: 512 KB	 Joonis 13. Arduino Due
Arduino Yún	ATmega32U4 ja Atheros AR9331 (Linux)	Tööpinge: 5V Lubatud sisendpinge: 5V Taktisagedus: 16 MHz Välkmälu: 32 KB ja Tööpinge: 3.3V Taktisagedus: 16 MHz Välkmälu: 16 MB Ethernet: IEEE 802.3 10/100Mbit/s WiFi: IEEE 802.11b/g/n	 Joonis 14. Arduino Yún.

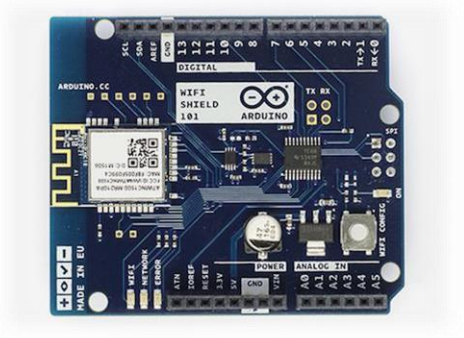
Arduino Gemma	ATtiny85	Tööpinge: 5V Soovitatav sisendpinge: 7-12V Lubatud sisendpinge: 6-20V Taktisagedus: 16MHz Välkmälu: 8 kB	 Joonis 15. Arduino Gemma
LilyPad Arduino Main Board	ATmega168 või ATmega328V	Tööpinge: 2.7-5.5 V Lubatud sisendpinge: 2.7-5.5V Taktisagedus: 8 MHz Välkmälu: 16 KB	 Joonis 16. LilyPad Arduino Main Board

Huvitava projekti jaoks ei piisa enamasti ainult arendusplaadist, nii läheb ka Arduinoga töötamiseks enamasti vaja ka lisatarvikuid. Suuremate lisafunktsionaalsuste katmiseks on võimalik kasutada laiendusplaate.

1.4.2 Laiendusplaadid

Laiendusplaadid (i.k *shield*) ühilduvad arendusplaatidega ning võimaldavad keerukamate projektide jaoks lisafunktsionaalsust, nagu näiteks internetiga ühendamise võimalus, või Bluetoothi tugi.

Laiendusplaatide kasutamiseks on vaja arendusplaati. Enim kasutatavad laiendusplaadid on Motor Shield, mis on mõeldud lisamootorite (nt. samm-mootor) ühendamise lihtsustamiseks, Ethernet Shield kaabliga internetiühenduse võimaldamiseks, Proto Shield, mis on abiks skeemide disainimisel ning annab juurde lisapesasid suuremate projektide jaoks, GSM Shield, mis koos GPRS toega SIM kaardiga annab Arduinole mobiiltelefoni funktsionaalsused ja WiFi Shield 101 (Joonis 17), mis võimaldab juhtmevaba internetiühendust.

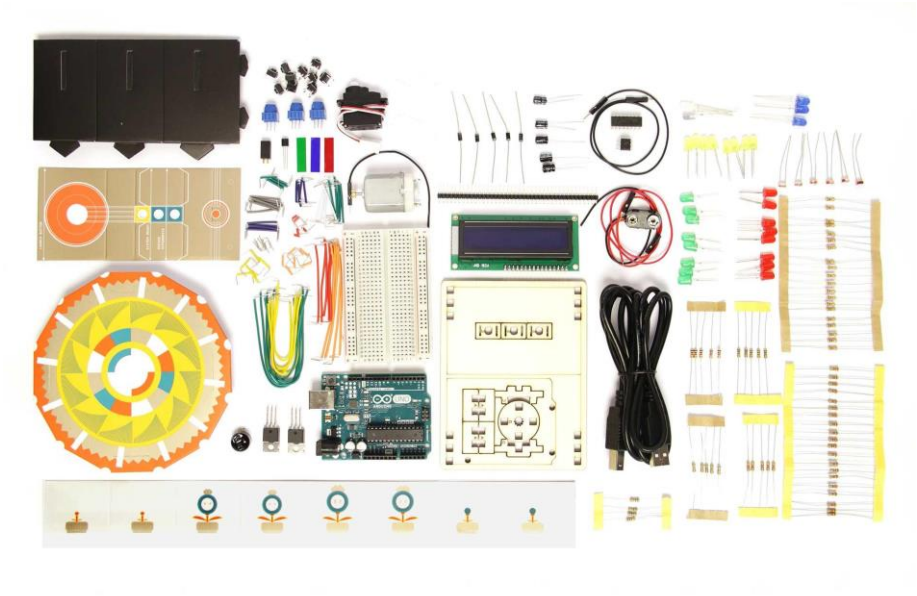


Joonis 17. Arduino WiFi Shield 101

Roboti ehitamiseks ega huvitavate katsete tegemiseks ei piisa ainult plaatidest, lisaks tulevad veel andurid, erinevad LED tuled ja kõik nende ühendamiseks vajalik. Seepärast on kõige parem alustada komplektiga, mis lisaks arendus- ja protoplaadile sisaldab ka vajalikke juhtmeid, maketeerimislauda, erineva värviga valgusdioode, takisteid ja nii alaloog kui ka digitaalandureid, mis tagavad hulgaliselt võimalusi erinevate projektide koostamiseks.

1.4.3 Arduinol põhinevad komplektid

Arduino ametlikud komplektid *Arduino Starter kit* ja *Arduino Basic Kit* (Joonis 18) on sisuliselt samad, ainuke erinevus on õpikus, *Starter kit* sisaldab 150-leheküljelist paberkandjal õpikut, *Basic kiti* ostjad saavad juurdepääsu sama sisuga e-materjalidele. Mõlemad sisaldavad UNO arendusplaati, üle 150 lisakomponendi, mille hulka kuuluvad erinevad andurid, LED-id, erineva takistusega takistid, nupud, juhtmed ja maketeerimislaud ning materjale 15 erineva projekti koostamiseks.



Joonis 18. Arduino ametlik komplekt (Basic Kit)

Originaalkomplektis sisalduvad juhendid on põhjalikud ning kõik komponendid kvaliteetsed ning luksuslikult pakendatud, mis teeb komplekti avamisest omaette elamuse.

Lisaks originaalkomplektidele on saadaval ka Arduino derivaatide ja kloonide komplekte, mille peamised komponendid sarnanevad originaalkomplektile, kuid mis sisaldavad odavama hinna eest rohkem erinevaid andureid ja lisakomponente. Sellised komplektid sobivad küll õpieesmärkide täitmiseks, kuid neis sisalduvad tooted pole üldiselt nii kvaliteetsed ja süstematiseeritult pakendatud. Peamised erinevused komplektide vahel on välja toodud tabelis 2.

Tabel 2. Arduinol põhinevate komplektide võrdlus

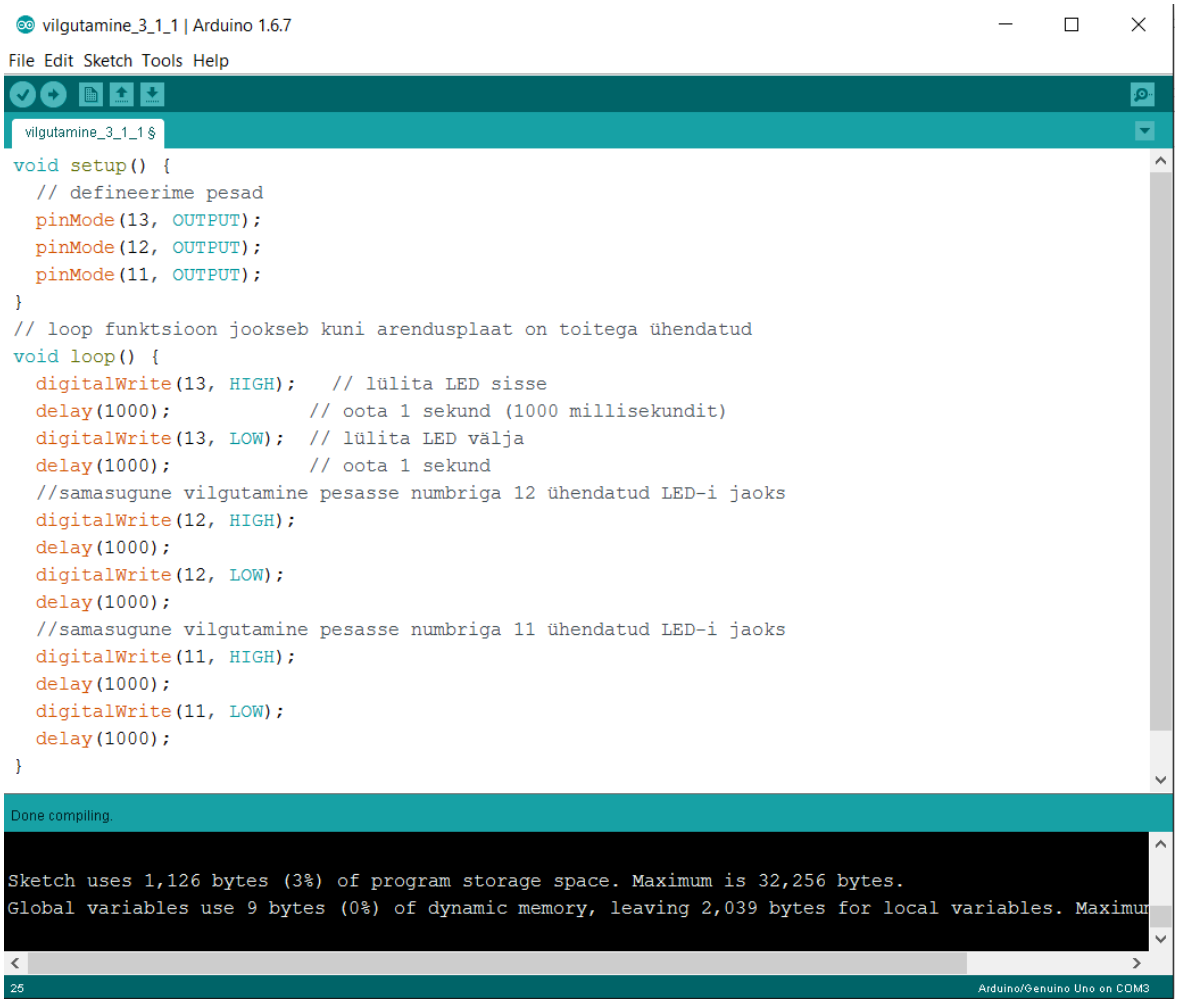
	Originaalkomplekt Arduino Basic kit	Klooni komplekt UNO R3 Starter Kit for Infiduino
Hind seisuga mai 2016 (ostulink)	73.64 EUR / $84 (Autodesk, Inc.)	46.05 EUR / $52.53 (ebay)
Aredusplaat	Arduino UNO board rev 3	Infiduino Uno R3 Mainboard
Lülitite arv	10 + 1 kaldlüliti	5 + 1 kaldlüliti
LED-ide arv	28 (5 eri värvi) + 1 RGB	30 (3 eri värvi) + 1 RBG
Takistite arv	60 (7 erineva tekistusega)	60 (3 erineva takistusega)
Maketeerimislaud	30- realine	60 - realine

Juhtmete arv	72	65 + 3 soontega kaablit
Mootorid	1 x servo + 1 x DC	1 x samm + 1 x servo
Ekraan	1602 LCD	1602 LCD
Andurite arv	8 (2 erinevat)	8 (kõik erinevad)
Laiendusplaat	-	Protoplaat koos väikse maketeermislauaga
Teiste lisavidinate arv	24 (10 erinevat)	16 (13 erinevat)

Võrdlusest tulenevalt sisaldab Infiduino stardikomplekt kõige madalama hinna eest enim erineva funktsionaalsusega andureid ja lisatarvikuid, seega kasutame seda oma edasises töös. Käesoleva õppematerjali teises peatükis kirjeldatakse täpsemalt antud stardikomplekti sisu ja kolmandas peatükis tuuakse välja erineva raskusastmega näidisprojekte, mis kasutavad komplektis olevaid komponente.

1.5 Arduino programmeerimine

Arduinot ja sellel põhinevaid teiste tootjate plaate on võimalik programmeerida kõigis keeltes, mida on võimalik kompileerida masinkoodi, kuid lihtsam on kasutada mõnda arenduskeskkonda. Arenduskeskkondi on erinevatele sihtrühmadele. Mõned kogenumad programmeerijad eelistavad AVR Studiot selle keerukama ja rohkemate võimalustega veahalduse poolest, kuid riistvaralähedast programmeerimist alustavatele inimestele on siiski kõige parem valik joonisel 19. kujutatud Adruino integreeritud arenduskeskkond koos Arduino programmeerimiskeelega. Selle eelisteks on aktiivne kasutajatugi, lihtne ülesehitus, põhjalikud kasutajajuhendid ning kasutatavus kõigi levinumate operatsioonisüsteemidega.



Joonis 19. Ekraanitõmmis Arduino arenduskeskkonnast.

Antud keskkond ise on kirjutatud Javas ning inspireeritud keskkonnast Processing ja teistest vabavaralistest rakendustest. Programmeerimiskeel põhineb raamistikul Wiring ning on väga sarnane C ja C++ keeltega. Arenduskeskkonna kasutamise eeliseks on koodi värvimine, sulgude ühendamine, samuti toimuvad nii kompileerimine kui ka koodi plaadile laadimine ühe hiireklõpsuga. Kood kirjutatakse kahte funktsiooni: *setup()*, kuhu läheb osa koodist, mida käivitatakse vaid korra - vajalik algseadete määramiseks. Teise funktsiooni *loop()* kirjutatakse peamine kood, mida kutsutakse korduvalt välja nii kaua kuni plaat on sisse lülitatud. Täpsema info arenduskeskkonna allalaadimise ja kasutamise kohta võib leida käesoleva materjali peatükist 2.2.

1.6 Võrdlus Raspberry Pi-ga

Lisaks Arduinole kasutatakse koolirobootikas ka teisi platvorme, millel kõigil on oma eelised ja kitsaskohad, enamused neist on märgatavalt kallimad (hind umbes 100 eurot või

üle). Seega on mõistlik võrrelda Arduinot sama hinnaklassi kõige populaarsema robootikaseadmega, milleks on Raspberry Pi (Joonis 20).

Joonis 20. Raspberry Pi Model 3 Model B

Raspberry Pi on sisuliselt iseseisev arvuti, Arduino seevastu arendusplaat, plaadi hind sõltub mudelist ja lisadest. Ainult plaadi saab Arduino UNO puhul kätte odavamalt kui Raspberry Pi 3. Mõlemale platvormile on saadaval lisaandureid, Arduinole ka laiendusplaate. Huvitavamate projektide jaoks vajab Arduino siiski rohkem lisasid, millest tingituna on ka lisasid sisaldava komplekti hind veidi kallim. Näitajate järgi, mis on täpsemalt välja toodud tabelis 2, on Raspberry Pi 3 kiirema taktisagedusega ja omab 512 000 korda rohkem mälu kui Arduino UNO. Lisaks sisaldab Raspberry Pi ka operatsioonisüsteemi Linux ja kaabliga internetiühenduse võimalust, mis Arduinol võimalik vaid laiendusplaadiga. Uuematel Raspberry Pi mudelitel on enamasti ka HDMI pesa, mis annab rohkem võimalusi teiste seadmetega ühendamiseks. Samas ei nõua Arduino stabiilset sisendpinget ning analoogpesade olemasolu teeb analoogandurite (näiteks heliandur) ühendamise tunduvalt lihtsamaks.

Tabel 3. Rasberry Pi Model 3 ja Arduino võrdlus

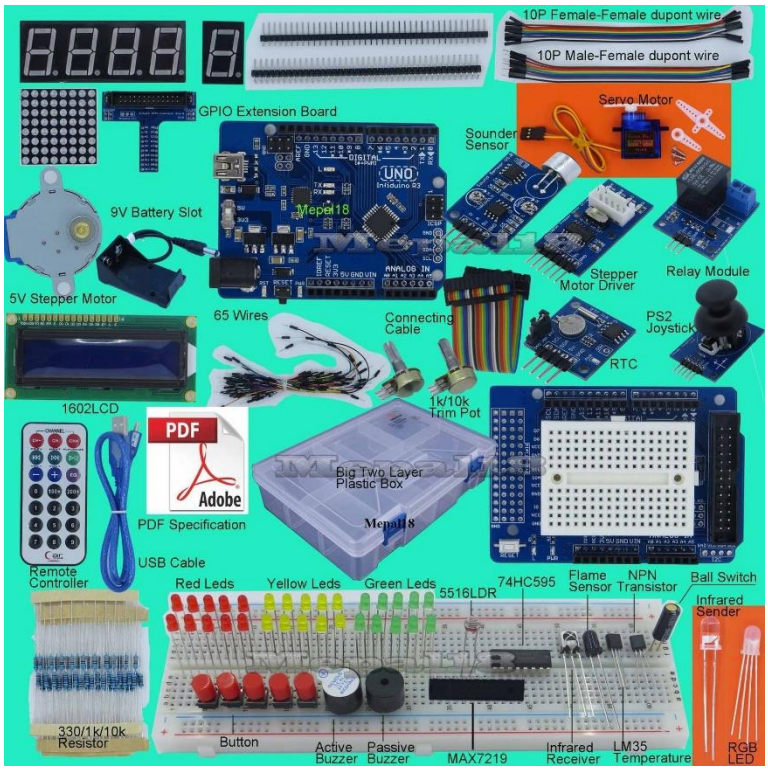
	Arduino UNO R3	Raspberry Pi 3 Model B
Mälu	0,002 MB	1 GB (lisaks SD kaardi võimalus)
Taktisagedus	16 Mz	1.2GHz
Lubatud sisendpinge	6- 20V	5 V

	(soovitatav 7 - 12 V)	
Ühendusvõimalused	14 digitaalpesa (millest 6 toetavad PWM väljundit) 6 analoogpesa	4 USB pesa 40 GPIO pesa 1 HDMI pesa 1 Ethernet pesa

Võrdlusest võib järeldada, et Raspberry Pi on sobilikum valik tarkvaraprojektide ja tarkvara testimise jaoks, kuid sellega on võimalik teha ka riistvara lähedasi lahendusi nagu näiteks nutirobot. Seevastu annab Arduino lihtsus ja huvitavate andurite rohkus sellele eelise riistvaraprojektide puhul, kuna võimaldab kergema vaevaga lahendada ülesandeid, mis vajavad väliskeskkonnale reageerimist. Paljude erineva funktsionaalsusega, kuid ehituselt lihtsate andurite tõttu on Arduino ka ideaalseks alguspunktiks riistvaralähedase programmeerimise õppimiseks.

2 Stardikomplekti sisu ja seadistamine

Käesolevas peatükis tutvustatakse ühes Arduino klooni stardikompletis (Sobilik alustamsieks kodukasutajal, Joonis 19) sisalduvaid tarvikuid ning nende kasutusvõimalusi, näidatakse kuidas lisasid arendusplaadiga ühendada, arenduskeskkonda kasutada ning koodi arendusplaadile laadida.



Joonis 21. UNO R3 Starter Kit for Arduino/Infiduino

2.1 Stardikomplekti sisu

Stardikoplekti kõige olulisem komponent on juba eelmisest peatükist tuttav UNO R3 mikrokontroller, sinna peale laaditakse soovitud kood ning külge ühendatakse kõik teised projektis kasutatavad komponendid. Erinevate tootjate komplektid võivad veidi varieeruda, antud töös on näidisprojektide tegemiseks kasutatud ebayst ostmishetkel 46.05 eurot maksnud Infiduino Ultimate UNO R3 stardikomplekti, kus on mõistliku hinna eest esindatud võimalikult paljud erinevad andurid. Ülevaate selle komplekti tarvikutest annab tabel 4.

Tabel 4. Arduino stardikomplekti komponendid

Nimetus	*ingliskeelne nimetus*	Kogus komplektis
Infiduino Uno R3 arendusplaat	*Infiduino Uno R3 Mainboard*	1
USB kaabel	*USB cable*	1
Juhtmed	*Wires*	2 x 10 soonega kaabel/ 65 x üksik
Maketeerimislaud	*Breadboard*	1 x suur ja 1x väike protoplaadil kasutamiseks
Takistid	*Resistors*	20 x 330 oom/ 20x 10Koom /20 x 1 Koom
Tavalised 3 mm valgusdioodid	*3mm Red LEDs*	10 x punane/ 10 x roheline / 10 x kollane
5 mm RGB valgusdiood	*5mm RGB LED*	1
Nupulülitid /Kaldlüliti	*Buttons with Cap / Ball switch*	5 x nupuga / 1 x kaldlüliti
Potentsiomeeter (muuttakisti)	*Potentiometer*	1 x 1K/ 1 x 10K
Infiduino prototüüpimise laiendusplaat + GPIO laiendusplaat + 26-soonega ühenduskaabel	*Infiduino Extension baord + GPIO Extension Board + Connecting Cable*	1
Juhtkang	*Joystick*	1
Pult	*Infrared Remote controller*	1
Infrapuna vastuvõtja	*Infrared Receiver*	1
Infrapuna saatja	*Infrared Sender*	1
1602A LDC ekraan	*1602A LCD Screen*	1
Heliandur	*Sound Sensor*	1
Fototakisti (valgusandur)	*Photoresistor*	1
Samm-mootor + draiverplaat	Stepper Motor + Driver Board	1
SG90 Servomootor	SG90 Servo motor	1
Sumistid	*Buzzers*	1 x aktiivne / 1x passiivne

Leegiandur	*Flame Sensor*	1
Jalgade komplektid	*Pin headers*	1 x küljega 40 pesa/ 1 x sirge 40 pesa
MAX7219 kiip	*MAX7219 Chip*	1
74HC595 8-bitine nihkeregister	*74HC595 8bit Shift Register*	1
LM35 temperatuuriandur	*LM35 Temperature Sensor*	1
SS8050 NPN bipolaartransistor	*SS8050 NPN Transistor*	1
7-segmendiline näidik	*Seven-segment Display*	1 x 1-numbriline /1 x 4-numbriline
8 * 8 LED maatriks	*8*8 Dot Matrix Led Display*	1
Kellaaja moodul	*RTC (Real Time Clock) module*	1
Relee moodul	*Relay Module (5V-10A)*	1
Patareihoidja (9 V patarei jaoks)	*9 V Battery Holder*	1

Täpsemad kirjeldused tabelis kujutatud toodete kohta on toodud järgnvates alampeatükkides.

2.1.1 Kõikides projektides kasutatavad komponendid

Mikrokontrollerit (Joonis 22) võib vaadata kui projekti südant, selle külge ühendatakse kõik ülejäänud projektis kasutatavad komponendid ning mälus hoitakse ülesannet täitvat koodi.

Joonis 22. Infiduino UNO mikrokontroller

USB kaabel on vajalik mikrokontrolleri ühendamiseks arvutiga nii toite eesmärgina kui ka koodi plaadile laadimise jaoks. Maketeerimislauale ühendatakse juhtmete abil andurid ja indikaatorid, selle olemasolu teeb võimalikuks komponentide ühendamise ilma jootmiseta, mistõttu on see hea lahendus prototüüpimiseks ja kooliro bootika projektide lahendamiseks.

2.1.2 Valgusdioodid

Valgusdioodid on pooljuhid, mis suudavad muuta elektrienergia valguseks. Pikemat jalga, millele rakendatakse positiivne pinge nimetatakse katoodiks, lühemat jalga, millele rakendatakse negatiivne pinge, anoodiks. Tavalised kahe jalaga valgusdioodid (Joonis 23) põlevad ainult ühte värvi.

Joonis 23. Eri värvi tavalised valgusdioodid

Ühel LED-il eri värvide kuvamiseks on välja töötatud 4-jalaga RBG valgusdiood (Joonis 24), kus üks jalg on kas anood või katood (sõltuvalt valgusdioodi tüübist) ja ülejäänud kolm tähistavad punast, rohelist ja sinist LED-i .

Joonis 24. RGB valgusdiood

Selline ehitus teeb võimalikuks ühe LED-i sees värvide vahetamise. Valgusdioode kasutatakse nii vooluringi ülesannetes vooluringi osana kui ka digitaalväljundina valgusmängu loomiseks ja erinevate andurite testimiseks.

2.1.3 Takistid

Takistid on vajalikud olukordades, kus tahame piirata komponendini jõudvat voolutugevust nagu näiteks valgusdioode kasutavates projektides, kuna liiga tugev elektrivool kahjustab LED-i, lühendades selle eluiga märgatavalt. Samas liiga tugeva takisti kasutamisel on valgusdioodid tuhmid ning projekt vähem efektne. Erineva takistusega takisteid on võimalik eristada värvitriipude järgi, väivikoodide lugemiseks on kõige lihtsam kasutada mõnda internetis leiduvat kalkulaatorit, näiteks G-Sound OÜ kodulehel asuv takistite värvikoodide kalulaator (link). Komplektis on takistid pakendatud 20 kaupa ning takistus on pakendiribale.

Joonis 25. Pakendis takistid

Sobiva takisti valimiseks peame välja arvutama, kui suurt takistust vaja on, seda võib teha käsitsi valemiga

$$R = \frac{(Arduino\ toide\ (5\ V) -\ valgusdioodi\ päripinge\ (keskmiselt\ 2\ V))}{valgusdioodi\ nimivool\ (0.02\ A)}$$

või kasutada mõnda kalkulaatorit, näiteks linear.org asuv LED kalkulaator (link). Saame vastuseks (sõltuvalt valgusdioodist) midagi 150Ω kanti, see on minimaalne väärtus, millega meie takisti tohib olla. Soovitatav on võtta umbes 20 Ω võrra tugevam. Kuna täpselt sobiva väärtusega takistit ilmselt pole, siis tuleb otsustada komplektist kõige lähema, suurema takistusega takisti kasuks.

2.1.4 Lülitid

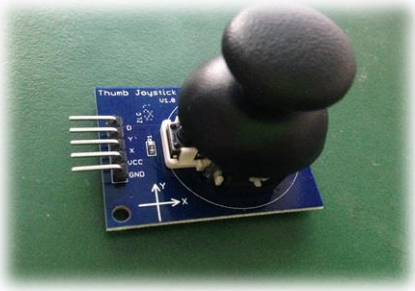
Lülitid on kasutusel vooluringi kontrollimiseks. Nupuga lülitite (Joonis 26) puhul ühendab lüliti vajutamine vooluringi. Kaldlüliti sõltub ühendus lüliti asendist, vooluring on ühendatud, kui kaldlüliti on püstises asendis ehk lüliti sees olev kuulike puutub ühendustega kokku.

Joonis 26. Tavalised nupuga lülitid

Lüliteid on lisaks vooluringi mehaanilisele osale võimalik ka vaadata kui andureid ja programmeerida digitaalsisenditena. Sellisel juhul tuleb kasutada takistit.

2.1.5 Projekti juhtimiseks vajalikud tarvikud

Projektide üle suurema kontrolli saavutamiseks kuuluvad komplekti ka põidlaga juhitav juhtkang (Joonis 27) ja koos infrapuna vastuvõtjaga kasutatav pult (Joonis 28).



Joonis 27. Juhtkang

Juhtkangi saab programmeerida vastavalt kangi asendile soovitud tegevust sooritama. Arduino Leonardo arendusplaat võimaldas ka juhtkangi kasutamist arvutihiirena, kuid selles töös kasutatav UNO R3 seda võimalust ei paku. Seega on antud komplekti puhul juhtkangi parim kasutusala erinevate mootorite liigutamine.

Joonis 28. Komplektis sisalduv pult

Parim viis projektide eemalt kontrollimiseks on puldi ja infrapuna vastuvõtja (Joonis 29) abil.

Joonis 29. Infrapuna vastuvõtja

Selleks tuleb tuvastada puldi nuppude signaalid ning seada need vastavusse soovitud tegevustega. Selliste ülesannete jaoks võib kasutada nii komplektis sisalduvat pulti või ka vabalt mõnda muud infrapuna saatjaga (puldi otsas olev väike läbipaistev andur) toodet, näiteks tavalist televiisoripulti. Erinevaid pulte kasutades tuleb arvestada, et programm tuleb kohandada kasutatava puldi signaalidele.

2.1.6 Mootorid

Servomootor on tagasisidega mootor, mida saab signaali abil soovitud asendisse juhtida. Signaali katkemisel naaseb mootor tasakaaluasendisse. Turul on saadaval eri pöördekraadiga servomootoreid. Poolpöörde servomootorid (Joonis 30) võimaldavad 180-kraadist liikumist ning nende kõige levinum kasutusala on robotite jäsemete

liigutamine. Täispöörde ehk 360–kraadiseid mootoreid saab kasutada robotite rataste veomootoritena.

Joonis 30. 180-kraadine ehk poolpöörde seromootor

Käesolevas komplektis sisaldus erineva pikkusega mootori külge ühendatavaid tiivikuid ning üks poolpöörde servomootor, mida saab juhtkangi abil näiteks lülitit kustutama juhtida.

Samm-mootor on mootor, mis pöörleb täisringe sammude kaupa. Erinevalt servost puudub samm-mootoril tasakaalupositsioon, kuid kuna võimaldavad määrata kiirust ning distantsi, siis on kindlaks määratud asukohta liikumine märgatavalt täpsem. Samm-mootoreid on nii unipolaarseid (Joonis 31) kui ka bipolaarseid. Erinevus seisneb mähiste ühendustes ja juhtimises, unipolaarset mootorit on kergem juhtida, kuna piisab võimalusest voolu välja lülitada, bipolaarse mootori juhtimiseks tuleb aga muuta polaarsust. Unipolaarset mootorit saab ühendada kui bipolaarset, kuid vastupidine ühendamine pole võimalik.

Joonis 31. Samm-mootor koos draiveriga

Samm-mootori kasutamiseks tuleb ühendada ka draiver, mis määrab sammu impulsid, liikumissuuna ning sammualgoritmi. Sobivad kasutamiseks kohtades, kus vaja täpsust, näiteks tindiprinterites tindi pea liigutamiseks. Samas ei ole samm-mootor parim valik roboti veomootorina, kuna võrreldes servomootoriga on nad väiksema erivõimsuse, suurema voolutarbega ning raskemini juhitavad.

2.1.7 Erkaanid ja näidikud

1602A LDC (Joonis 32) on kahest reast ja 16 veerust koosnev vedelkristallekraan, info, näiteks andurite abil saadud mõõtmistulemuste, kuvamiseks. Iga andmekuvamiskast on lahutusvõimega 2 x 7 pikslit. Ekraani kontrollimiseks saab kasutada Arduino arenduskeskkonnaga kaasas olevat teeki *LiquidCrystal library*. Ekraani saab kontrollida nii 4-biti kui ka 8 -biti režiimis. Ainult teksti kuvamiseks piisab 4 -bitist.

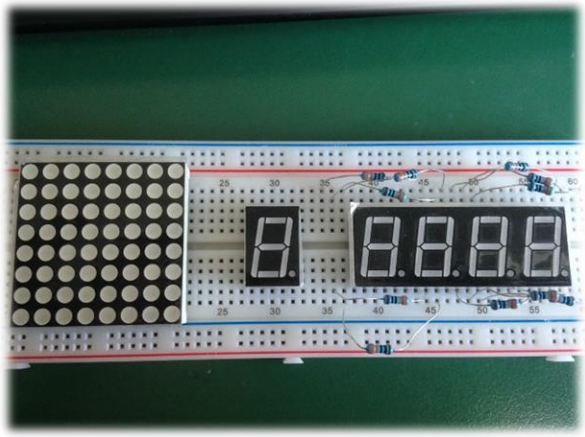
Joonis 32. 1602A LDC ekraan

Joonisel 30 kujutatud ekraani saab kontrollida 16 pesa abil, millest esimene, VSS tähistusega, on maandus, järgmisse, VDD märgiga ühendatakse toide (5 V), kolmas, V0, on kasutusel kontrasti muutmiseks, neljas (RS) tähistab registrivalikut andme- ja juhisteregistri vahel. Andmeregistrist tulev info kuvatakse ekraanil, juhisteregistri järgi valitakse järgmine tegevus. Viies, RW märgistusega, pesa määrab kas informatsiooni loetakse või kirjutatakse, mis ainult kirjutamiseks ühendatakse lihtsalt maandusega. Kuues (E), reguleerib, kas registritesse kirjutamine on lubatud või mitte. Järgmised kaheksa pesa, tähistusega D0-D7, on andmepesad, mille seisundid (*HIGH* või *LOW*) tähistavad registrisse kirjutatavaid ning sealt loetavaid bitte. Kaks viimast pesa on LED taustavalguse anood (A) ja katood (K).

Paljudes komplektides, kaasaarvatud käesolevas, on ekraan transpordi lihtsustamise eesmärgil pakendatud jalgadest eraldi. Sellisel juhul tuleb enne kasutama hakkamist jalad pesade külge joota. Järgneva peatüki näidisprojektides kasutatud ekraani külge on joodetud komplektis sisalduvad sirged jalad.

LCD ekraani kasutamine pole komplekti ainuke info kuvamise võimalus. Numbrilisi näitajaid saab hästi edasi anda ka 7- segmendilistel LED valgusega näidikutel. Samal

tehnoloogial põhineb ka 8x8 LED maatriks.



Joonis 33. 8x8 LED maatriks, 1- ja 4- numbrilised 7- segmendilised näidikud

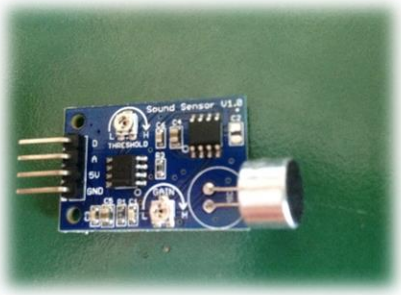
Ühest numbrist koosneva info, näiteks puldil vajutatud numbri, kuvamiseks sobib ka 1-numbriline näidik, kuid mõõtmisandmete nagu temperatuur või kellaaeg näitamiseks tuleb kasutada 4-numbrilist displeid. Maatriksil on võimalik kuvada näiteks üksikuid tähti või 64-punkti abil eristatavaid pildikesi.

2.1.8 Keskkonnale reageerivad andurid

Fototakisti (Joonis 34) on valgustundlik andur, mida kasutatakse valguse olemasolu ja intensiivsuse määramiseks. Mida pimedam on, seda suurem on takistus (kuni 1 megaoom), valguse peale takistus väheneb sõltuvalt valguse intensiivsusest. Ereda valguse puhul võib takistus langeda mõne oomini. Fototakisteid kasutatakse näiteks kaamerate valgusandurina, valgusvihu järgi liikuvates robotites ning tänavalampide valgustundlikuks muutmiseks.

Joonis 34. Fototakisti

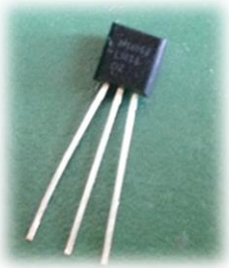
Heliandur (Joonis 33) on mikrofoni ja heliandmete töötlusplaadi kombineerimisel saadud andur, mis suudab mõõta müratugevust.



Joonis 35. Heliandur

Komplektis olevat andurit on võimalik kasutada nii analoog kui ka digitaalandurina. Digitaalsisendina kasutamine on mõttekas, kui soovime infot ainult müra olemasolu kohta. Olukordades, kus soovime saada andmeid ka müratugevuse kohta, tuleb ühendada andur analoogpesaga.

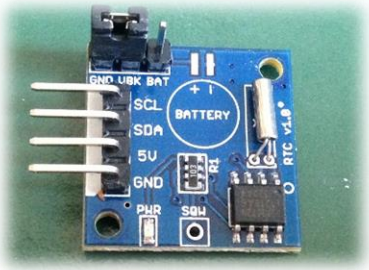
Temperatuuriandur (Joonis 34) on analoogandur, mis mõõdab ümbritseva keskkonna temperatuuri. Saadud vastuse saame kerge vaevaga teisendada endale sobivaks ühikuks (näiteks kraadideks Celsiuse järgi).



Joonis 36. Temperatuuriandur LM35

Temperatuuriandurit kasutatakse tavaliselt õhutemperatuuri mõõtmiseks, kuid võib ka proovida, kui adekvaatset tagasisidet annab kehatemperatuuri mõõtmine näiteks peopesast.

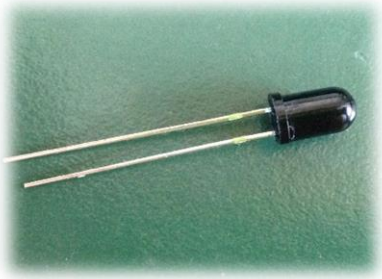
Kellaaja moodul (Joonis 35) ehk RTC (Real Time Clock) on vajalik praeguse ajahetke tuvastamiseks. Seda saab kasutada digitaalse kella loomisel või ajaarvestuse pidamiseks hetkedel, kus ülejäänud süsteem on toitest eemaldatud.



Joonis 37. Kellaaja moodul

Toitest sõltumatu ajaarvestuse pidamiseks tuleb RTC moodulil olevale patareikohale lisada patarei.

Leegiandur (Joonis 36) suudab tuvastada infrapuna lained lainepikkusega 760 nm ~ 1100 nm. Tänu sellele on see tundlik leegist tulevale kiirgusele, kuid nähtav valgus komponenti ei mõjuta.



Joonis 38. Leegiandur

Andur reageerib elavale tulele, mille tõttu koolides kasutamine ei ole praktiline. Samas on sel komponendil oma roll targa maja projektides ohutust tagava andurina.

2.1.9 Teised komponendid

Infrapuna saatja (Joonis 37) on seade, mille abil on võimaik saata infrapuna vastuvõtjale loetavat signaali elektroonikaseadmete juhtimiseks.

Joonis 39. Infrapuna saatja

Selle abil on näiteks võimalik ehitada televiisoripult või edastada signaale erinevate arendusplaatide vahel.

Potentsiomeeter on muuttakisti, mille abil on võimalik reguleerida näiteks valgusdioodide eredust või helitugevust. Välimuselt sarnaneb potentsiomeeter kruvile ning seda on saadaval

erinevate takistustega.

Joonis 40. Potentsiomeetrid

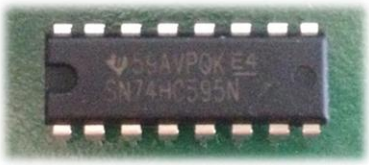
Käesolevas komplektis sisaldub kaks potentsiomeetrit, 10-kilo-oomine ja 1-kilo-oomine.

MAX7219 kiip (Joonis 39) on näidiku draiver, mis sobib kasutamiseks ühise katoodiga displeide puhul.

Joonis 41. MAX7219 kiip

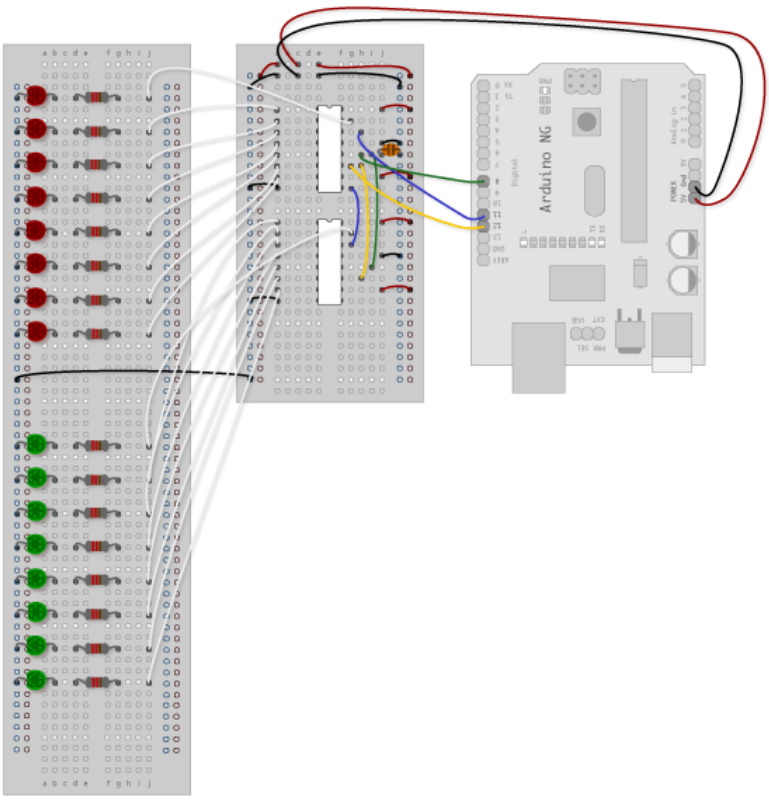
Selle abil on võimalik lihtsustada nii mitme 7-segmendiliste näidiku üheaegset ühendamist kui ka 8 x8 maatriksite kasutamist.

Nihkeregister on vajalik olukordades, kus arendusplaadil olevatest pesadest jääb soovitud ülesande täitmiseks väheks. Komplektis sisalduv 74HC595 8-bitine nihkeregister (Joonis 40) võimaldab kaheksa digitaalpesa asemel kasutada korraga kolme.



Joonis 42. 74HC595 8-bitine nihkeregister

Digitaalpesade puudus või esineda näiteks siis, kui soovime põneva valgusmängu jaoks ühendada korraga palju valgusdiode, veel võimsama tulemuse saame mitme nihkeregistri üheaegsel kasutamisel (Joonis 41).



Joonis 43. Näide kahe nihkeregistri üheaegsest kasutamisest

Nagu näha jooniselt 41, siis saame paralleelselt kasutada mitut nihkeregistrit nii, et arendusplaadil oleks endiselt hõivatud vaid kolm digitaalpesa.

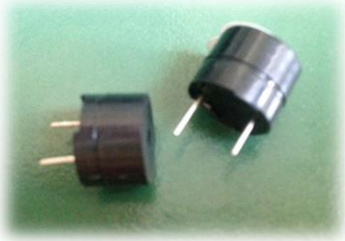
SS8050 NPN (Joonis 42) on bipolaartransistor ning võimaldab ühendada Arduinoga kõrgemate elektriliste nõuetega seadmeid.

Joonis 44. SS8050 NPN transistor

NPN on transistori tüüp, mille kasutamisel ühendatakse kollektor mootoriga ja emitter maandusega. Ilma transistorita ei ole digitaalpesadega võimalik ühendada komponente, mille elektrilised nõudmised on kõrgemad kui arendusplaadi poolt pakutav 40mA vool ja 5V pinge. Kõige enam on transistoritest kasu mootorite ühendamisel, kuna suuremate

robotite ehitamisel kasutatavatel võimsamatel mootoritel on kõrged tehnilised nõudmised. Käesoleva transistori abil on võimalik ühendada komponente, mis nõuavad kuni 1,5A voolu ja 40V pinget.

Sumistid on helisignaali tekitavad komponendid, mida saab kasutada näiteks äratuskellades, uksekellades või hoiatussignaali tekitamiseks ohutusandurites.



Joonis 45. Passiivne ja aktiivne sumisti

Komplektis sisaldub kaks erinevat sumistit – aktiivne, mis suudab ise signaali toota ning passiivne, millele tuleb Arduinoga soovitud signaal ette anda.

Relee moodul on elektriline lüliti, mille abil on võimalik kontrollida Arduinost tunduvalt kõrgema pingega töötavaid elektroonikaseadmeid. Relee kontrollimine Arduino toimub digitalWrite funktsiooni abil, antakse ette käsk sisse (HIGH) või välja (LOW) lülitamiseks. Projektis sisalduva relee abil on võimalik näiteks 9 V patarei kontrollimine. Komplektis sisaldub patareialus, kuid patarei tuleb ise juurde osta.

Joonis 46. Relee moodul ja patareihoidja

Releed on tähtsal kohal targa maja projektides erinevate elektroonikaseadmete lülitamiseks, näiteks kohvimasina käivitamine äratuskella helina peale.

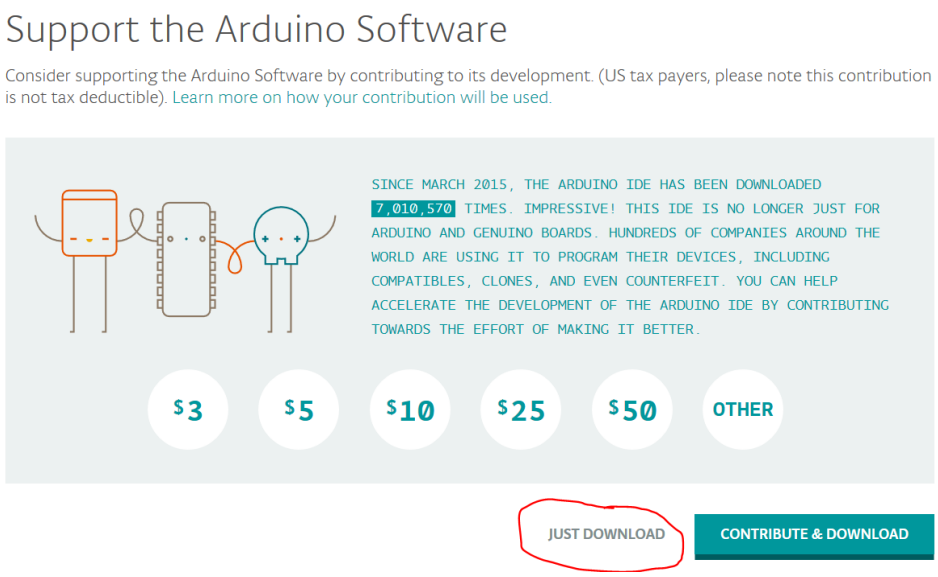
2.2 Arenduskeskkonna paigaldamine ja arendusplaadi ühendamine

Arduino kasutamiseks on kõigepealt vaja alla laadida koodi kirjutamiseks ja kompileerimiseks vajalik tarkvara, näiteks Arduino arenduskeskkond. Keskkonna leiab Arduino ametlikult kodulehet joonisel 47 kujutatud "*Download*" vaatest (link) ja selle allalaadimine on tasuta.

Joonis 47. Arenduskeskkonna alla tõmbamise esimene samm

Allalaadimise alustamiseks tuleb valida joonisel 47 punasega märgistatud menüüst kasutatavale operatsioonisüsteemile sobiv versioon. Windowsi kasutajatel on valida installeri, mis nõuab käivitamiseks administraatoriõigusi ja kokkupakitud programmi vahel. Kuna enamikes kooli ja koduarvutites on kasutusel Windows, siis antud peatükis näitame, kuidas installeerida Arduino arenduskeskkonda Windowsile.

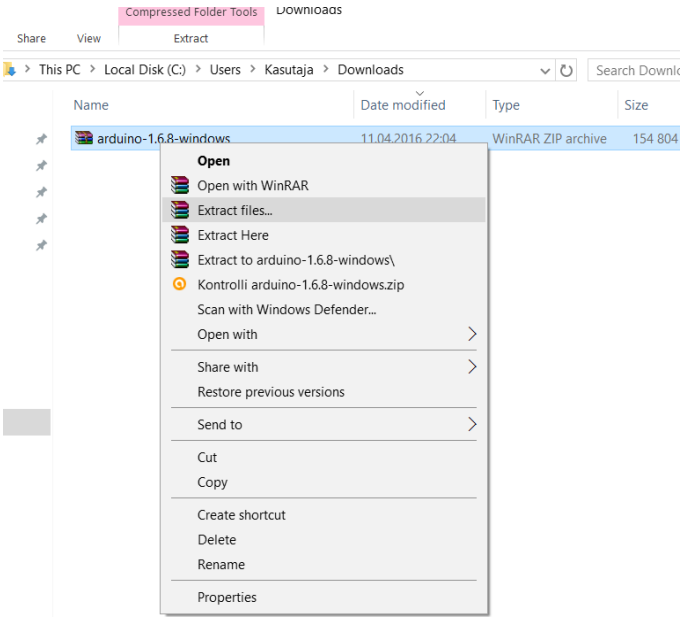
Pärast sobiva versiooni valimist avanevas vaates (Joonis 48) küsitakse, kas soovite Arduinole annetada, kui hetkel seda teha ei soovi, siis valige joonisel 48 punasega märgistatud "*JUST DOWNLOAD*".



Joonis 48. Arenduskeskkonna alla tõmbamise samm 2

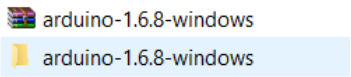
Kui allalaadimine on lõppenud, siis Windowsi installeri puhul topeltklõps installeri exe formaadis failile, nõustuge tingimustega, valige programmile meelepärane asukoht ja vajutage "*Install*". Pärast edukat installeerimist tuleb kiri "*Completed*", seejärel klõpsake ainukest aktiivset nuppu "*Close*". Installeerimine on edukalt lõppenud, arenduskeskkonda on võimalik avada töölauale ilmunud otseteest ja ka kaustas, kuhu programmi salvestasite, asuvast arduino.exe failist.

Kokkupakitud versiooni puhul avage kaust, kuhu zip faili salvestasite, klõpsake failile parema hiireklahviga ning valige "Extract files..." (Joonis 49).



Joonis 49. ZIP faili lahti pakkimine

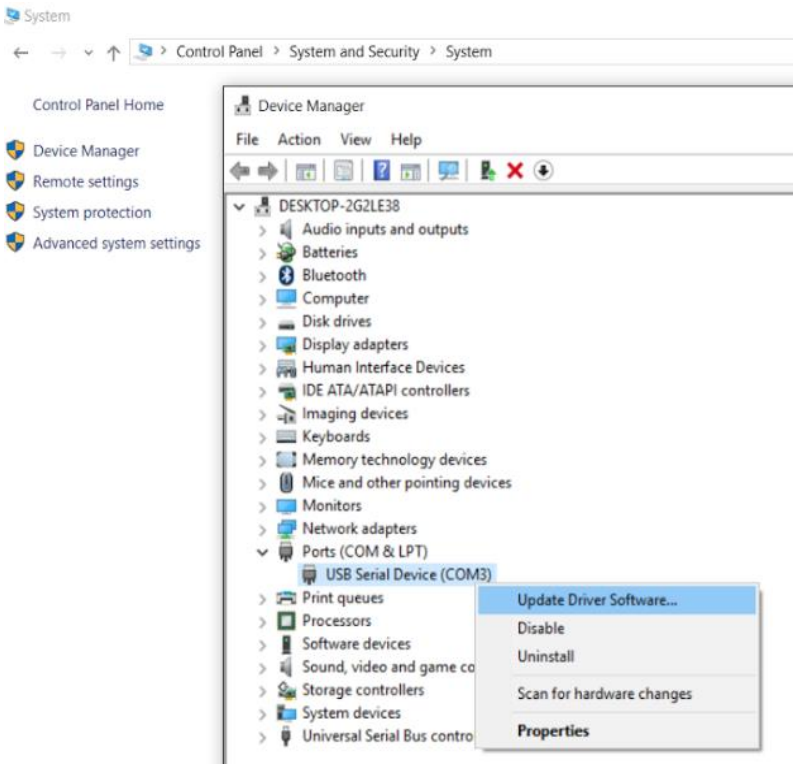
Seejärel klõpsake “Ok” ning samasse kausta ilmub zip failiga sama nimega kaust (Joonis 48), kus sees on soovitud tarkvara. Arenduskeskkonna käivitamiseks topeltklõps kaustas olevale arduino.exe failile.



Joonis 50. Lahtipaikitud kaust arenduskeskkonnaga

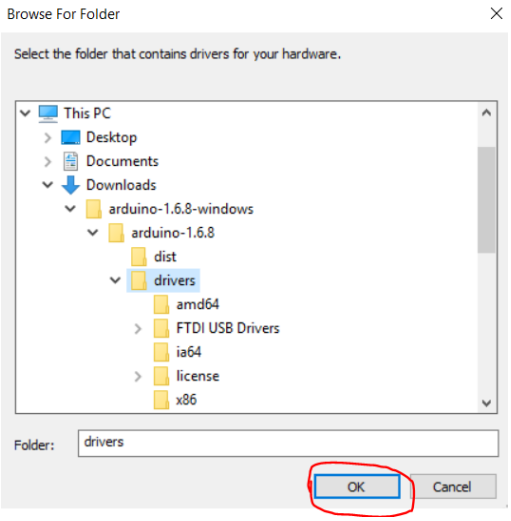
Nüüd kui vajalik tarkvara on olemas, võib ühendada arendusplaadi arvutiga. Selleks ühendada eelmises alampeatükis tutvustatud USB kaabli väiksem ots mikrokontolleri külge ja teine arvutiga. Ühenduse saamisel sütib mikrokontrolleril roheline PWR märgistusega tuluke. Esmakordsel ühendamisel käivitatakse automaatselt draiverite installatsioon (alates Windows XP kuni Windows 10 puhul), oodata kuni see lõpetab.

Kui mingil põhjusel draiverite installatsioon esmaühendamisel ei õnnestu, siis avada arvutis “*Control Panel*” (“Juhtpaneel”) -> “*System and Security*” (“Süsteem ja turve”) -> “System” (“Süsteem”). Avenenud vaates avada “*Device manager*” (“Seadmehaldur”). Leida üles rippoendist “*Ports (COM & LPT)*” (“Pordid (COM & LPT)”) ja selle all peaks olema meie arendusplaat, mille nimeks antud näites on USB Servial Device COM3, kuid mõnda teist plaati kasutades võib nimi olla veidi erinev. Juhul kui “*Ports (COM & LPT)*” peaks rippoendist puuduma, siis tuleks vaadata “*Other Devices*" (“Teised seadmed”) alla ning seal peaks olema "*Unknown Device*" (“Tundmatu seade”). Seejärel tuleb klõpsata seadmed parempoolse hiireklahviga ning valida draiverite tarkvara uuendamine (Joonis 51).



Joonis 51. Draiverite tarkvara uuendamine

Avanenud aknas valida draiverite tarkvara sirvimine oma arvutist ("*Browse my coputer for driver software*"). Nüüd tuleb otsida draiverite alamkaust kaustast, kuhu meil Arduino tarkvara on installeeritud, milleks tarkara paigaldamiseks installerit kasutades on tõenäoliselt (C:\Program Files (x86)\Arduino\drivers). ZIP failina installatsiooni puhul tuleb navigeerida kausta, kuhu ZIP sai lahti pakitud ning valida kaust “drivers”. Arduino draiverite kaust leitud, vajutada ok (mitte lasta end häirida sellest, et draiverite kaust sisaldab alamkaustu ja valida see mille nimi on lihtsalt “drivers”) (Joonis 52).



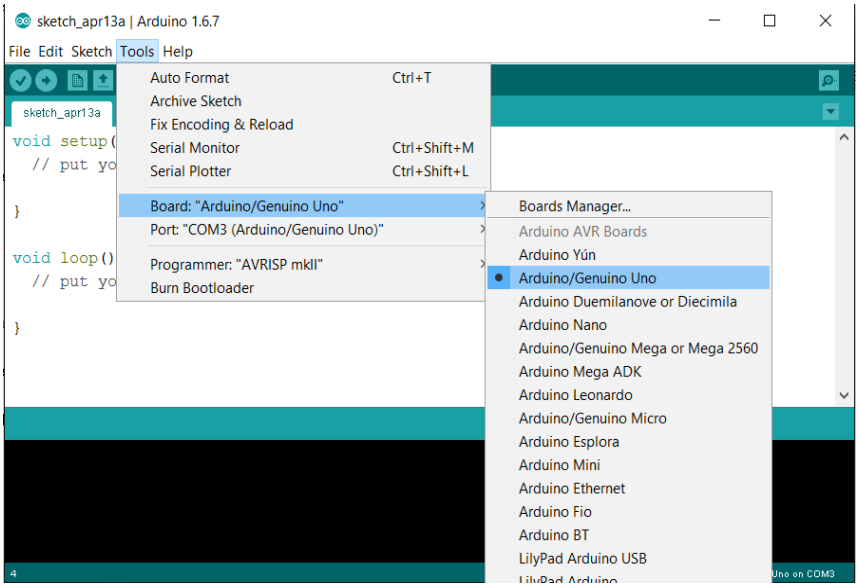
Joonis 52. Drainverite tarkvara leidmine enda arvutist

Nüüd tuleb vajutada lihtsalt “Next” ja oodata, et Windows lõpetaks draiverite installatsiooni.

Sellega on ettevalmistused tehtud, koodi testimisest, kirjutamisest ja plaadile laadimisest tuleb juttu järgmises alampeatükis.

2.3 Koodi kirjutamine ja plaadile laadimine

Koodi kirjutamiseks tuleb avada Arduino arenduskeskkond ning valida “*Tools*” alt millise arendusplaadiga tegu (Joonis 51), kuna käesoolevas töös vaadeldavas stardikomplektis sisalduv plaat on võrdväärne Arduino UNO-ga, siis selle seaded sobivad.

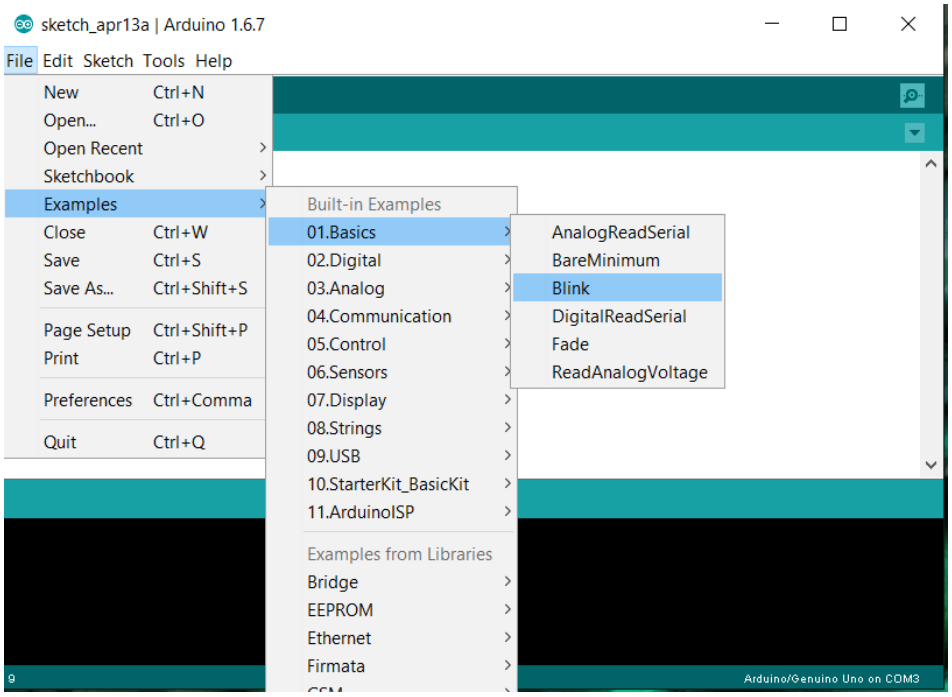


Joonis 53. Plaadi mudeli valimine

Kontrollida üle ka pordi valikud, mida saab teha Joonisel 53 kujutatud “Port” valikust (asub “Board” valiku all). Kui valikus on ainult üks port, siis ongi juba õige port valitud, kui porte on mitu ja tekib kahtlus, kas valitud on õige, siis jätta meelde millised pordid valikus on, ühendada Arduino arvuti küljest lahti ja avada sama menüü uuesti, nüüd on üks port kadunud ja see puuduolev port ongi selle arendusplaadi oma. Seejärel saab plaadi tagasi ühendada, õige port on taas tagasi ja saame selle valida.

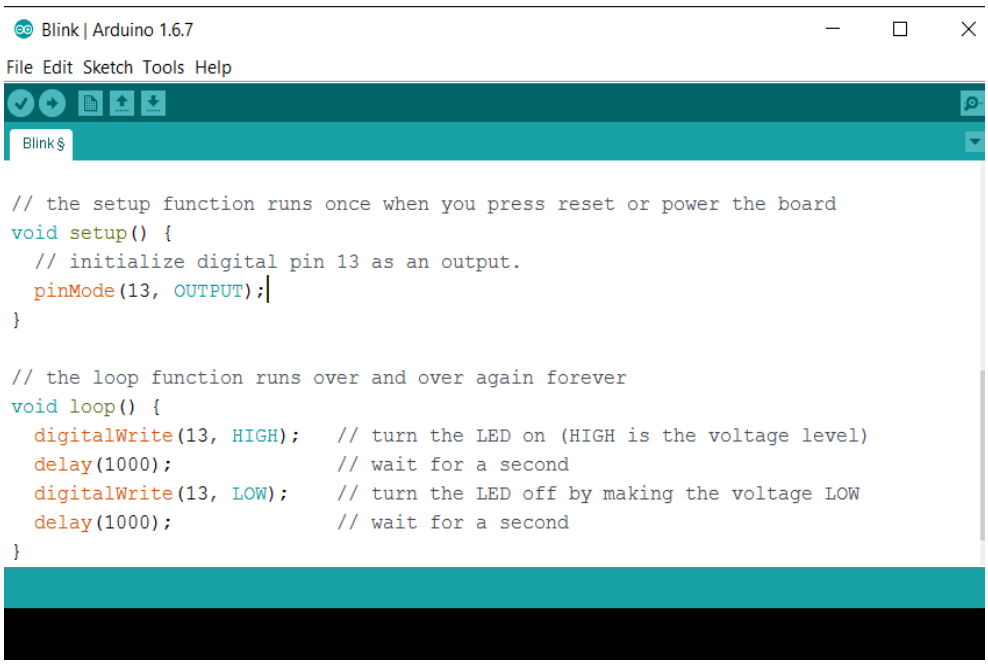
Esimeseks programmiks sobib väga hästi valgusdioodi vilgutamine, mis vilgutab plaadil olevat LED-i (tähistatud L tähega). Antud programm on juba arenduskeskkonnaga kaasas ning selle ja paljude teiste näitekoodide leidmiseks tuleb valida menüü “File” alt

“Examples”. Vilgutamiskood asub kaustas “01-Basics” ning selle nimi “Blink”, antud tee on kujutatud ka joonisel 54.



Joonis 54. Tee vilgutamisnäiteni

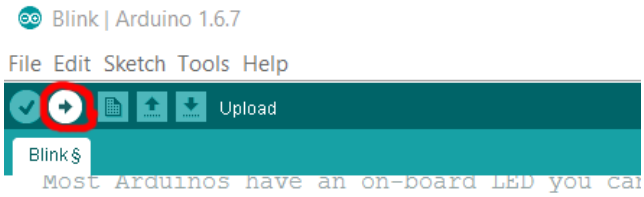
Seejärel avaneb uus arenduskeskkonna aken (Joonis 55), kus on lahti Vilgutamise projekti kood. Suur osa avanevas aknas olevast tekstist on kommentaarid, need read koodi toimimist ei mõjuta. Mitmerealine kommentaar paigutatakse märkide /* */ vahele ning üherealist kommentaari tähistab //.



Joonis 55. Vilgutamisnäite kood

Funktsioonis setup määratakse millisesse väljundiks digitaalpesa numbriga 13, kuna plaadil olev LED on ühendatud selle pesaga. Funktsioonis loop, mida täidetakse nii kaua kuni plaat on vooluallikaga ühendatud, lülitatakse LED sisse, oodatakse 1 sekund (1000 millisekundit), lülitatakse LED välja ja oodatakse uuesti 1 sekund.

Nüüd on teoorias selge mida programm tegema peaks ning saame selle plaadile laadida. Soovi korral võib koodi ka enne lihtsalt kompileerida, vajutades rohelisel ribal olevale linnukese märgile ("*Verify*"), see harjumus tuleb hiljem ise kirjutatud koodide puhul kasuks, kuna aitab vigu avastada. Kompileerimiseks ja plaadile laadimiseks valida programmi ülaääres asuvalt tegevusribalt noolemärk ehk "*Upload*" (Joonis 56).



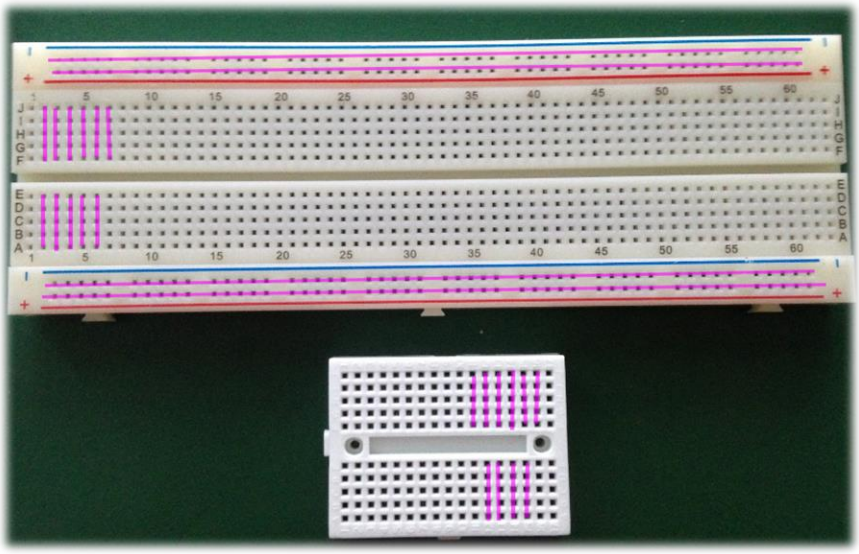
Joonis 56. Koodi plaadile laadimine

Kui kõik läks hästi, siis koodi all olevasse musta kasti tuleb valge kirjaga teade palju mälu kood võtab ning kasti kohal olevale rohelisele ribale ilmub väike kiri "*Done uploading*". Vea ilmnemisel läheb roheline riba oranžiks, selle ilmub veateade ning musta kasti kuvatakse punase kirjaga täpsem info vea kohta.

2.4 Maketeerimislaua ja protoplaadi kasutamine

Käesolevas töös on enamus projekte koostatud kasutades peatükis 2.1 tutvustatud maketeerimislauda, kuna see annab võimaluse mugavalt ja jootevabalt koostada erinevatest komponentidest koosnevaid vooluringe. Selline vajadus tekib valgusdioodide ühendamisel ning ka siis, kui arendusplaadiga tuleb ühendada rohkem kui üks andur.

Joonisel 57 on roosaga märgitud voolu liikumine maketeerimislaual.



Joonis 57. Voolu liikumine maketeerimislaual

Vool liigub suure maketeermislaua äärmistes ridades (tähistatud + ja -) mööda rida. Pluss märgiga tähistatud reale ühendatakse toide (arendusplaadil tähistatud 5 V), miinus märgiga reale maandus (arendusplaadil tähistatud GND). Kõigis keskmistes ridades liigub vool mööda veergu, need read on mõeldud komponentide paigutamiseks.

Väiksemal protoplaadil (joonsel 57 alumine) liigub vool mööda veerge, mille tõttu tuleb maandus ise pisikeste juhtmejuppide abil üle plaadi vedada. Samas on võimalik kompaktsuse tõttu mõned juhtmed ka ära jätta. Näiteks saame ära jätta valgusdioode kasutavas projektis takisti ja digitaalväljundi vahelise juhtme ning ühendada takisti otse sobivasse digitaalpesasse. Valgusdioodide ühendamist seletava ja teiste lihtsamate projektidega on võimalik tutvuda käesoleva töö järgmises kolmanda peatüki esimestes projektides.

Keerukamate projektide puhul, kus maketeerimislaual hakkab ruumi väheks jääma, on võimalik protoplaati ja maketeerimislauda koos kasutada. Selle teeb lihtsamaks samuti peatükis 2.1 tutvustatud GPIO laiendusplaat.

3 Õppematerjalid

Õppematerjal on suunatud informaatika (programmeerimine), matemaatika (arvutused), füüsika (elektroonika, mehaanika) aineõpetajatele. Käesolevas peatükis koostatakse juhendid, mille abil on võimalik lahendada nii Arduinoga alustavatele sobiliku raskusastmega projekte kui ka natuke keerulisemaid ülesandeid. Kõikides õppematerjalides on kasutatud eelmises peatükis lähemalt tutvustatud Arduino arendusplaate ja nendega sobituvaid komponente. Õppematerjalid koosnevad praktilistest näiteülesannetest, kus näidatakse, kuidas ühendada ja programmeerida erinevaid komplektis sisalduvaid andureid, LED tulesid, lüliteid ja teisi komponente. Materjalide läbimine annab ülevaate Arduino platvormi võimalustest, analoog ja digitaalandurite tööst, elektroonika ühendamisest ning põhilistest programmeerimises kasutatavatest võtetest. Joonised on koostatud vabavaralise prototüüpide koostamise programmiga fritzing. Projektide lõpuks valmib sõitev Arduino robot.

3.1 Erinevad võimalused LED vilgutamiseks ja FOR-tükkel

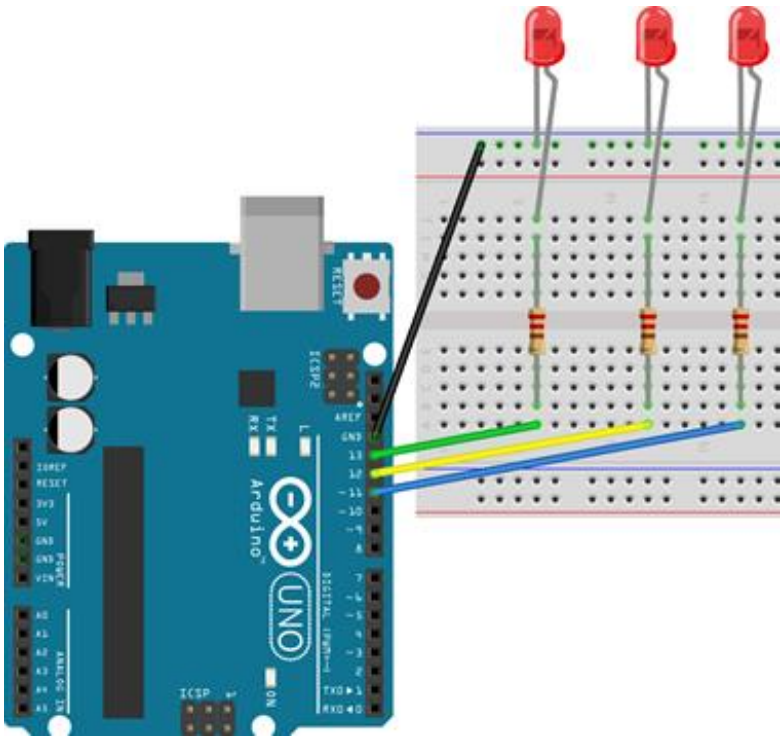
Käesolevas alampeatükis ühendatakse LED- tulesid maketeermislauale, proovitakse läbi erinevad vilgutamise võimalused ja tutvustatakse programmeerimises palju kasutatavat FOR-tsüklit.

Projekti koostamiseks läheb vaja:

1. 1 x UNO mikrokontroller
2. 1 x USB kaabel
3. 1 x maketeerimislaud
4. 3 x vabalt valitud värvi LED-id
5. 3x 330 Ω takistid
6. 4 x tavalised üksikud isane-isane juhtmed

Enne komponentide ühendamist tuleb veenduda, et arendusplaat oleks arvuti küljest (ja muudest toidetest) lahti ühendatud.

Ühendada kolm LED-i rööpühenduses maketeerimislauale nii, et katood (LED-i lühem jalg) oleks ühendatud maketeerimislaual miinusreale ning anood (LED-i pikem jalg) ühendatud maketeerimilaua keskel takistiga ning seejärel digitaalsiginaaliga, antud näites on kasutusel digitaalpesad numbritega 13, 12 ja 11. Miinusreale tuleb lisada ka maandusjuhe (ühedatud digitaalpesade kohal olevase GND pesasse). Ühendused võimalik üle kontrollida ka järgnevalt jooniselt 58.



Joonis 58. Kolme valgusdioodi ühendamine maketeerimislauale

Nüüd on aeg ühendada arendusplaat USB juhtme abil arvutiga ning avada arenduskeskkond. Kõigepealt võime uuesti proovida juba eelmisest peatükist tuttavat Blink näidet (*File -> Examples -> Basics -> Blink*). Laadime koodi kõigepealt muutmata kujul plaadile. Selle peale peaks vilkuma hakkama nii plaadil olev LED kui ka number 13 digitaalpesasse ühendatud valgusdiood.

Kahel ülejäänud lambil sellise koodiga otstarve puudub. Proovime kopeerida Blink projektis olevat koodi nii, et ka pesadega 12 ja 11 ühendatud LED-id kasutust leiaksid, näide selle kohta on toodud järgnevas koodiplokis (koodi kommentaarid muudetud eestikeelseks).

```
void setup () {
  // defineerime pesad
  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(11, OUTPUT);
}
// loop funktsioon jookseb kuni arendusplaat on toitega ühendatud
void loop() {
  digitalWrite(13, HIGH);   // lülita LED sisse
  delay(1000);            // oota 1 sekund (1000 millisekundit)
  digitalWrite(13, LOW);  // lülita LED välja
  delay(1000);            // oota 1 sekund
  //samasugune vilgutamine pesasse numbriga 12 ühendatud LED-i jaoks
  digitalWrite(12, HIGH);
  delay(1000);
  digitalWrite(12, LOW);
  delay(1000);
  //samasugune vilgutamine pesasse numbriga 11 ühendatud LED-i jaoks
  digitalWrite(11, HIGH);
  delay(1000);
  digitalWrite(11, LOW);
  delay(1000);
}
```

Joonis 59. Kood valgusdioodide järjest vilgutamiseks

Saime koodi mis küll töötab, aga on ebaefektiivne, kuna sarnast funktsionaalsust kutsutakse mitu korda uuesti välja ning mida rohkem LED-e kasutusel on, seda tüütumaks selline olukord muutuks. Selle vältimiseks on mõistlik panna kirja sama asi FOR- tsükli abil sarnaselt järgnevas koodiplokis kujutatud näitele.

```
void setup() {
// defineerime pesad
  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(11, OUTPUT);
}
//loop funktsioon jookseb kuni arendusplaat on toitega ühendatud
void loop() {
  // tsükkel, mis käib läbi pesad 11 kuni 13, paneb LED-id põlema ja
```

```
kustutab ära
  //suurendame pesa numbrit iga ringiga
  for (int pesa = 11; pesa <= 13; pesa++) {
      digitalWrite(pesa, HIGH); // lülita LED sisse
      delay(1000); // oota 1 sekund (1000 millisekundit)
      digitalWrite(pesa, LOW); // lülita LED välja
      delay(1000); // oota 1 sekund
  }
}
```

Joonis 60. Kood järjest vilgutamiseks kasutades FOR-tsüklit

Pauside eemaldamiseks enne järgmise LED-i süttimist tuleb eelnevast koodist eemaldada viimane FOR - tsükli sees olev rida (*delay(1000);*). Kõikide LED-ide korraga põlema jätmiseks tuleb eemaldada FOR - tsükli kehast kolm viimast rida (*delay(1000), digitalWrite(pesa, LOW);* ja *delay(1000);*). Kõigi lampide järjest põlema panemiseks ja kustutamiseks tuleb lisada kustutamise jaoks uus tsükkel, mida võib näha ka järgnevas koodiplokis.

```
void setup() {
  // defineerime pesad
  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(11, OUTPUT);
}
//loop funktsioon jookseb kuni arendusplaat on toitega ühendatud
void loop() {
  // tsükkel, mis käib läbi pesad 11 kuni 13
  //paneb LED-id põlema suurendame pesa numbrit iga ringiga
  for (int pesa = 11; pesa <= 13; pesa++) {
      digitalWrite(pesa, HIGH); // lülita LED sisse
      delay(1000); // oota 1 sekund (1000 millisekundit)
  }
  delay(1000); // oota veel üks sekund kui kõik LED-id põlevad
//tsükkel kustutamiseks
  for (int pesa = 11; pesa <= 13; pesa++) {
      digitalWrite(pesa, LOW); // lülita LED välja
      delay(1000); // oota 1 sekund
  }
}
```

Joonis 61. Kood valgusdioodide korraga süütamiseks ja kustutamiseks

Sellega oleme läbi proovinud mõned valgusdioodide erinevatest vilgutamisvõimalustest ja tutvunud FOR- tsükliga. Lisaks võib proovida veel näiteks kõigi korraga süütamist ja kustutamist ning vilkumiskiiruse muutmist (*delay* funktsioonile vähem millisekundeid ette andes).

3.2 Lüliti ühendamine ja LEDi reageerimine vastavalt lüliti asendile vooluringis ja digitaalsignaali abil IF-lausega

Käesolevas alampeatükis näidatakse esmalt lihtsa vooluringi koostamist, mis sisaldab LED- tulesid ning erinevaid lüliteid. Seejärel lüliti abil programmi kontrollimist, mille käigus tutvustatakse ka programmeerimises levinud IF-tingimust.

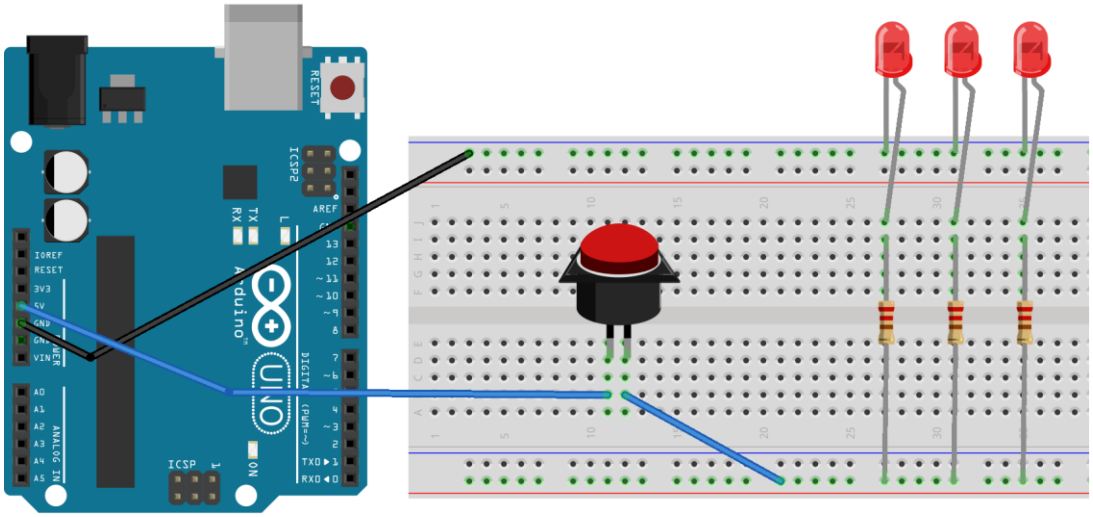
Projekti koostamiseks läheb vaja:

1. 1 x UNO mikrokontroller
2. 1 x USB kaabel
3. 1 x maketeerimislaud
4. 3 x vabalt valitud värvi LED-id
5. 3x 330 Ω takistid
6. 1 x 10k Ω takisti
7. 4 x tavalised üksikud isane-isane juhtmed
8. 1 x tavaline lüliti
9. 1 x kaldlüliti

Enne komponentide ühendamist tuleb veenduda, et arendusplaat oleks arvuti küljest (ja muudest toidetest) lahti ühendatud.

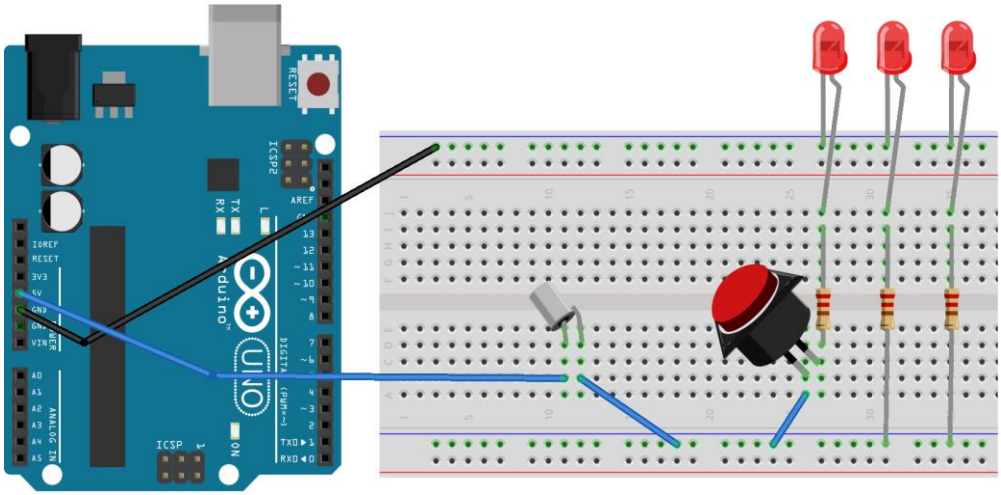
Algus sarnaneb peatükis 2.6.1. tehtud LED-ide vilgutamise projektile, tuleb ühendada kolm LED-i rööpühenduses maketeerimislauale nii, et katood (LED-i lühem jalg) oleks ühendatud maketeerimislaual miinusreale ning anood (LED-i pikem jalg) ühendatud maketeerimilaua keskel takistiga, seekord digitaalsignaaliga ühendamist pole vaja, selle asemel ühendame takisti ühe jala hoopis maketeerimislaua ääres olevale + reale.

Seejärel paigutada lüliti mugavasse kohta maketeerimislaua keskel nii, et lüliti jalad oleks paigutatud erinevatesse veergudesse. Komponendid paigas, lisada maandusjuhe (GND) LED-ide katoodidega samale miinusreale ning toitejuhe (5 V) lüliti ühe jalaga samasse veergu. Vooluringi lõpetamiseks lisada juhe lüliti teise jala ja + rea vahele sarnaselt joonisel 62 kujutatule.



Joonis 62. Lüliti ja valgusdioodide ühendamine

Testimiseks ühendada arendusplaat toite eesmärgil arvutiga, kuna tegu on lihtsa vooluringiga, kus arendusplaat on vaid vooluallikaks, siis arenduskeskkonda ega koodi vaja ei lähe. Kui kõik ühendused toimivad, siis peaks nupule vajutades kõik kolm valgusdioodi põlema minema ning nupu lahti laskmisel uuesti kustuma. Kui ühe lülitiga katse töötab, siis võib proovida sisuliselt sama asja mitme lülitiga, lisades LED-idele individuaalseid lüliteid. Joonisel 63 on kasutatud peamise lülitina kaldlülitit ning ühele LED-ile on lisatud tavalüliti.

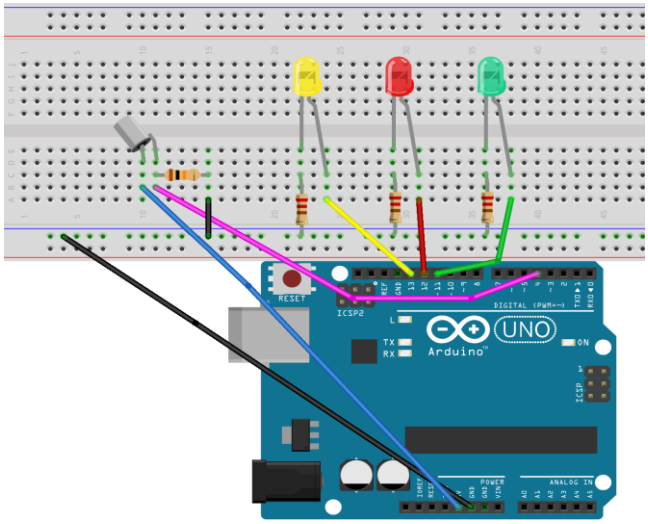


Joonis 63. Nupulüliti, kaldlüliti ja valgusdioodide ühendamine

Nüüd peaksid tavaasendis põlema kaks LED-i ja kolmas siis kui lüliti on alla vajutatud. Maketeerimislaua küljele kallutamisel peaksid kustuma kõik tuled.

Kui tavaline vooluring selge, siis võib proovida olukorda, kus lülitiga kontrollitakse signaali, näiteks lülitada vilkuvaid LED-e. Selleks ühendada arendusplaat aruti küljest lahti ning paigutada LED-id maketeerimislaual natuke ümber, nii et katood (lühem jalg) oleks

ühendatud läbi takisti maandusega ning anood (pikem jalg) digitaalsignaaliga, antud näites on LED-id ühendatud pesadesse 11, 12, ja 13. Lüliti (pole oluline, kas kasutada kaldlülitit või tavalist) toitejuhe võib jääda endisele kohale, kuid LED-ide kontrollimiseks tuleb lisada ühendus digitaalsignaaliga. Selleks lisada vahele maandusega ühendatud 10 kilo-oomine takisti ja sarnaselt joonisel 64 kujutatule võib digitaalsignaali ühendada näiteks pesasse numbriga 4.



Joonis 64. Lüliti ühendamine digitaalsisendina

Digitaalsignaali kontrollimiseks lüliti abil ühendada arendusplaat arvutiga ning laadida arenduskeskkonna abil plaadile järgnev kood, kus kontrollitakse IF-tingimuse abil, kas lüliti on sisse lülitatud, kui tingimus kehtib täidetakse eelmisest alampeatükist tuttav FOR-tsükkel. Tingimuse mittekehtimise korral kustutatakse kõik valgusdioodid.

```
//lülitamise algväärtus
int on_lylitatud = 0;
void setup() {
  // defineerime LED-ide pesad
  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(11, OUTPUT);
  // defineerime lüliti pesa sisendina
  pinMode(4, INPUT);
}
//loop funktsioon jookseb kuni arendusplaat on toitega ühendatud
void loop() {
  //loeme digitaalsignaali kas lüliti on vajutatud
 on_lylitatud = digitalRead(4);
 //kontrollime, kas signaal on olemas
 if (on_lylitatud == HIGH){
  //kui on, siis kordame eelmise alampeatüki vilgutamisülesannet
  //ainult, et natuke kiirema vilkumisega
  for (int pesa = 11; pesa <= 13; pesa++) {
```

```
      digitalWrite(pesa, HIGH); // lülita LED sisse
      delay(100); // (100 millisekundit)
      digitalWrite(pesa, LOW); // lülita LED välja
      }
  }
 else{
  //signaali puudumise korral, tuled kustu
      digitalWrite(13, LOW); //LED  välja
      digitalWrite(12, LOW); // LED välja
      digitalWrite(11, LOW); // LED välja
  }
}
```

Joonis 65. Kood lüliti kasutamiseks digitaalsisendina kõikide LED-ide korraga kontrollimiseks

Nüüd peaksid kaldlüliti puhul tavaasendis kõik LED-id vilkuma ning maketeerimislaua küljele kallutamisel kustuma. Tavalise lüliti puhul vilkuma vaid nuppu all hoides. Võib katsetada ka igale LED-ile oma digitaalsisendina defineeritud lüliti lisamist.

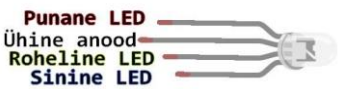
3.3 RGB ledi ühendamine ja RGB koodide katsetamine analogWrite abil

Käesolevas punktis näidatakse, kuidas ühendada peatükis 2.1.2 lähemalt tutvustatud värvimuutvat RGB valgusdioodi ning värvide muutmist, kasutades selleks ise defineeritud abifunktsiooni ja analogWrite käsku.

Projekti koostamiseks läheb vaja:

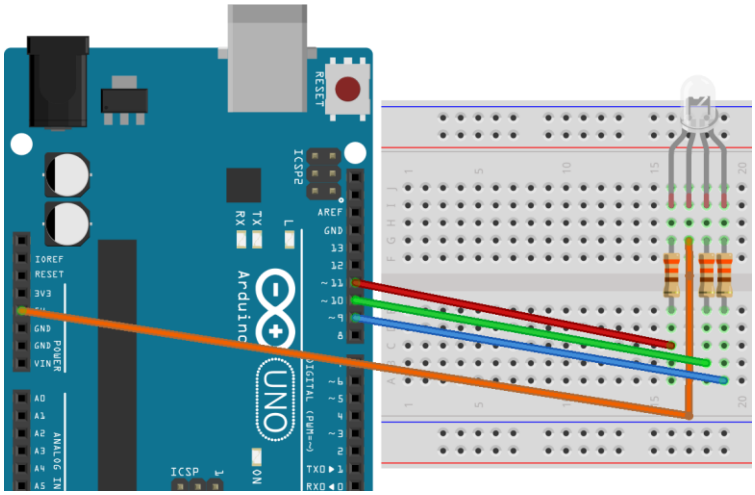
1. 1 x UNO mikrokontroller
2. 1 x USB kaabel
3. 1 x maketeerimislaud
4. 1 x RGB LED
5. 4 x tavalised üksikud isane-isane juhtmed
6. 3 x 330 Ω takisti

Enne komponentide ühendamist tuleb veenduda, et arendusplaat oleks arvuti küljest (ja muudest toidetest) lahti ühendatud.



Joonis 66. RGB LED-i jalgade seletus

Algus sarnaneb LED-ide vilgutamise projektile ainult seekord tuleb ühendada LED-i kolm lühemat jalga sarnaselt üksikutele valgusdioodidele, kuna need tähtistavadki ükskuid põhivärvidega LED-e (värviselgitused näidatud eelneval joonisel 64), mille üheaegsel erineva heledusega näitamisel saame luua erinevad värvikombinatsioone. Pesadena tuleb kasutada digitaalpesasid, kus juures on # või ~ märgistus, need pesad võimaldavad pinge muutmist analogWrite käsu abil, selles näites võtame kasutusele 9, 10 ja 11. Kuna tegemist on ühise anoodiga tüüpi RBG LED-iga siis tuleb anood (kõige pikem jalg) ühendada toitega (5 V). Ühendused peaksid sarnanema järgneval joonisel 65 kujutatule.



Joonis 67. RGB valgusdioodi ühendamine

Nüüd saab avada arenduskeskkonna ja hakata katsetama analogWrite käsu abil värvimängude loomist. Ideid värvide kombineerimiseks saab näiteks leheküljelt: http://www.rapidtables.com/web/color/RGB_Color.htm .

Järgnevas koodiplokis võib näha näidiskoodi, kus defineeritakse abifunktsioon, mis võtab argumentideks põhivärvide tugevused ning segab neid analogWrite käsu abil RGB valgusdioodil. Abifunktsiooni kutsutakse iga sekundi tagant välja erinevate argumentidega, mille tulemuseks on RGB LED-i värvi muutumine 1 sekundi tagant.

```
//defineerime pesad LED-i jalale vastava värvi järgi
int punanepesa = 11;
int rohelinepesa = 10;
int sininepesa = 9;
int sekund = 1000; //defineerime sekundi millisekundites

void setup()
{
  //seadistame väljundina
```

```
  pinMode(punanepesa, OUTPUT);
  pinMode(sininepesa, OUTPUT);
  pinMode(rohelinepesa, OUTPUT);
}

void loop()
{
  //kutsume välja abifunktsiooni ning punase/sinise/rohelise tugevust
reguleerides katsetame eri värve
  segaV2rve(255, 0, 0); //punane
  delay(sekund); //ootame sekundi
  segaV2rve(0, 255, 0); //roheline
  delay(sekund);
  segaV2rve(0, 0, 255); //sinine
  delay(sekund);
  segaV2rve(255,255,0); //kollane
  delay(sekund);
  segaV2rve(0,255,255); //aqua
  delay(sekund);
  segaV2rve(255,0,255); //fuksiaroosa
  delay(sekund);
   segaV2rve(0,128,128); //sinakasroheline
  delay(sekund);
  segaV2rve(255,255,255); //valge
  delay(sekund);
}
//abifunktsioon, millele anname argumendiks punase, rohelise ja sinise
tugevuse vastavalt RGB tabelile
void segaV2rve(int punane, int roheline, int sinine)
{
  //kuna tegu ühise anoodiga RGB LED-iga, siis tuleb RGB värvikoodide
abil soovitud värvi saamiseks argumentide väärtused vastupidiseks
muuta
  punane = 255 - punane;
  roheline = 255 - roheline;
  sinine = 255 - sinine;
  //erinevate värvide saamiseks määrame milline LED millise tugevusega
põleb, kasutades analogWrite käsku
  analogWrite(punanepesa, punane);
  analogWrite(rohelinepesa, roheline);
  analogWrite(sininepesa, sinine);
}
```

Joonis 68. Kood eri värvide kuvamiseks RBG valgusdioodil

Parema ettekujutuse parajasti põlevast värvist saab, kui LED-ile ümber panna valgest õhukesest A4 paberist toru. Hetkel tulevad värvid alati kindlas järjekorras, projekti huvitavamaks muutmiseks võib järgmise värvi valida juhuslikult, seades värvikombinatsioonid vastavusse kindlate arvudega ning genereerides random() käsu abil juhuslikke muutujaid. Ingliskeelne juhend Arduinoga juhuslike muutujate kasutamiseks asub siin.

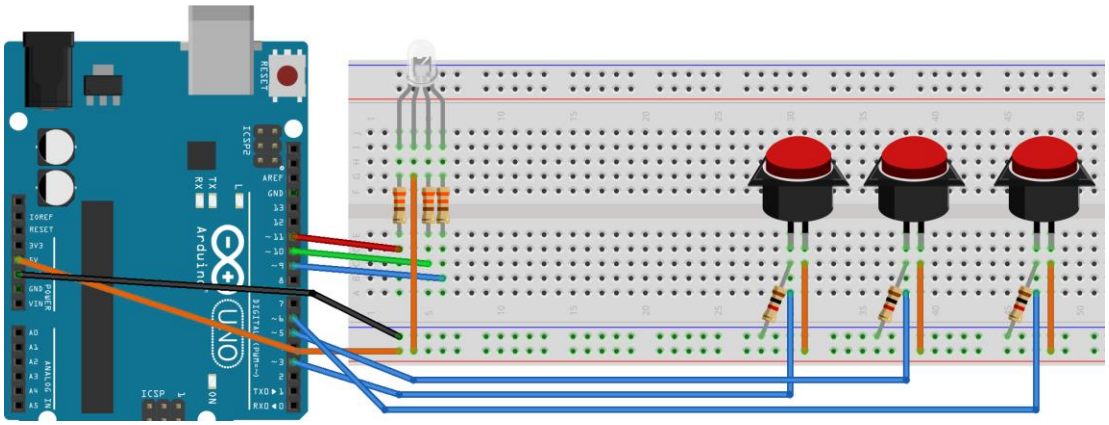
3.4 Kolme nupu abil RGB LEDi lülitamine

Eelmises alampeatükis tutvusime RGB LED-i ühendamise ja vilgutamisega, kuid meil polnud kontrolli parajasti põleva värvi üle. Selles alampeatükis selgitatakse, kuidas kontrollida RGB valgusdioodi kolme lüliti ja mitme IF-ELSE lause abil.

Projekti koostamiseks läheb vaja:

1. 1 x UNO mikrokontroller
2. 1 x USB kaabel
3. 1 x maketeerimislaud
4. 1 x RGB LED
5. 3 x 330 Ω takisti
6. 3 x 10 KΩ takisti
7. 3 x tavaline nupuga lüliti
8. 12 x tavalised üksikud isane-isane juhtmed

Ühendame RGB LED-i maketeerimislauale sarnaselt eelmisele alampeatükile. Kuna seekord on vaja paigutada ka kolm toidet vajavat lülitit, siis ühendame toitejejuhtme maketeerimislaua äärmisele + märgistusega reale ning lisame LED-i toitmiseks ühe väikse juhtme juurde. Lülitite ühendamine sarnaneb alampeatüki 3.2 viimasele osale, kus tuli samuti ühendada lüliti maanduse, voolu ja digitaalpesaga, kasutades 10 kilooomist takistit. Ühendused kujutatud ka järgneval joonisel 69.



Joonis 69. RGB LED-i ja kolme nupu ühendamine

Nüüd tuleb nupud programmeerida nii, et iga nupp kontrolliks ühte värvi. Selleks kasutame eelmise alampeatüki LED-i värvide muutmise koodi kombineerituna alampeatüki 3.2 lõpus tutvustatud lüliti ja *digitalRead* käsu abil LED-i kontrollimise võttega. Näide tulemusest järgevas koodiplokis.

```
//defineerime pesad LED-i jalale vastava värvi järgi
int punanepesa = 11;
int rohelinepesa = 10;
int sininepesa = 9;
//defineerime lülitite pesad
int lyliti1 = 3;
int lyliti2 = 5;
int lyliti3 = 6;
//lülitamiste algväärtused
int on_lylitatud1 = 0;
int on_lylitatud2 = 0;
int on_lylitatud3 = 0;
//defineerime värvide algväärtused
int pu= 0;
int ro = 0;
int si = 0;

void setup()
{
  //seadistame väljundina
  pinMode(punanepesa, OUTPUT);
  pinMode(sininepesa, OUTPUT);
  pinMode(rohelinepesa, OUTPUT);
  //seadistame lülitid sisendina
  pinMode(lyliti1, INPUT);
  pinMode(lyliti2, INPUT);
  pinMode(lyliti3, INPUT);
}

void loop(){
  on_lylitatud1 = digitalRead(lyliti1);
  on_lylitatud2 = digitalRead(lyliti2);
  on_lylitatud3 = digitalRead(lyliti3);
//kontrollime kas mõni nuppudest on vajutatud - kui on, siis lülitame
nupule vastama värvi sisse
  if (on_lylitatud1 == HIGH){
      pu = 255;
   }
  else{
      pu = 0;
   }
  if (on_lylitatud2 == HIGH){
      ro = 255;
   }
  else{
      ro = 0;
```

```
   }
  if (on_lylitatud3 == HIGH){
      si = 255;
   }
  else{
      si = 0;
   }
  //kutsume välja abifunktsiooni ning punase/sinise/rohelise tugevust
reguleerides katsetame eri värve
  segaV2rve(pu, ro, si);

}
//abifunktsioon, millele anname argumendiks punase, rohelise ja sinise
tugevuse vastavalt RGB tabelile
void segaV2rve(int punane, int roheline, int sinine)
{
  //kuna tegu ühise anoodiga RGB LED-iga, siis tuleb RGB värvikoodide
abil soovitud värvi saamiseks argumentide väärtused vastupidiseks
muuta
  punane = 255 - punane;
  roheline = 255 - roheline;
  sinine = 255 - sinine;
  //erinevate värvide saamiseks määrame milline LED millise tugevusega
põleb, kasutades analogWrite käsku
  analogWrite(punanepesa, punane);
  analogWrite(rohelinepesa, roheline);
  analogWrite(sininepesa, sinine);
}
```

Joonis 70. Kood RGB LED-il värvide muutmiseks kolme nupu abil

Nüüd peaks iga lüliti kontrollima erinevat RGB valgusdioodi värvi. Erinevaid nuppe üheaegselt all hoides saame värvikooslusi ise kontrollida. Hetkel põlevad LED-id ainult siis, kui nupud on all, edasiarendusena võib proovida nuppe programmeerida nii, et nupuvajutus jätab LED-i põlema nii kauaks kui seda värvi kontrollivat nuppu uuesti vajutatakse.

3.5 Ekraani ühendamine ja teksti kuvamine 1602 LCD ekraanil

Käesolevas punktis koostatakse juhend peatükis 2.1.7 tutvustatud 1602 LCD ekraani ühendamiseks, näidatakse ning kuidas sellel infot kuvada. Tausta ja teksti kontrastsuse reguleerimiseks kasutatakse potentsiomeetrit.

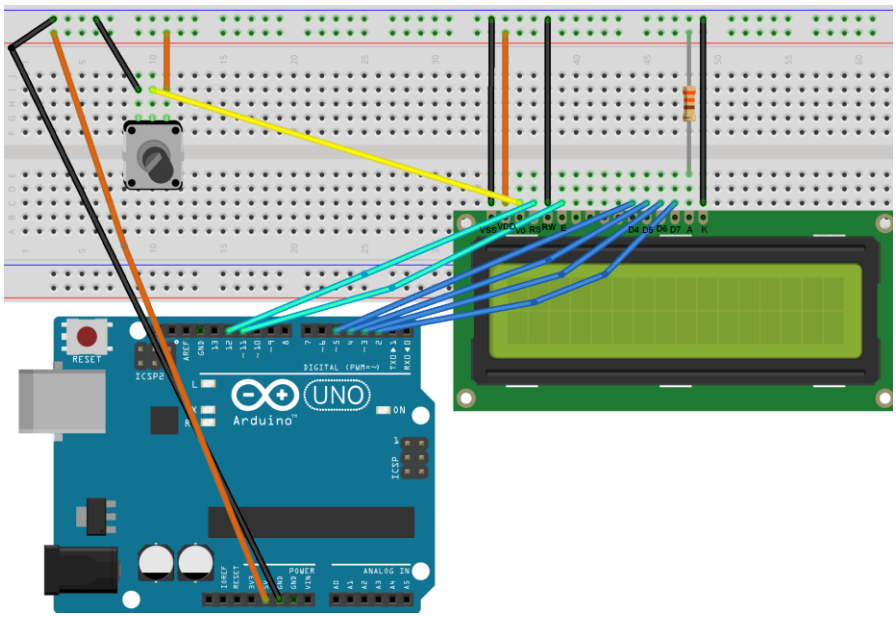
Projekti koostamiseks läheb vaja:

1. 1 x UNO mikrokontroller
2. 1 x USB kaabel

3. 1 x maketeerimislaud
4. 1 x 1602 LCD ekraan
5. 1 x potentsiomeeter
6. 15 x tavalised üksikud isane-isane juhtmed (6 lühemat ja 9 pikemat)
7. 1 x 330 Ω takisti

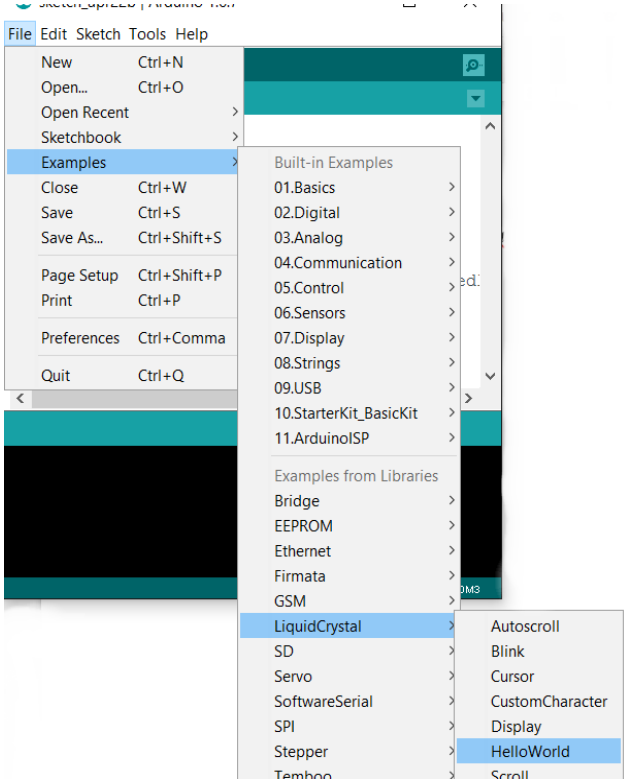
Enne komponentide ühendamist tuleb veenduda, et arendusplaat oleks arvuti küljest (ja muudest toidetest) lahti ühendatud.

Kõigepealt tuleb paigutada ekraan ja potentsiomeeter maketeerimislauale ja ühendada need, sarnaselt joonisel 69 kujutatule arendusplaadiga nii, et V0, mis määrab kontrasti oleks ühendatud potentsiomeeter, RS registrivalikuks digitaalpesaga 12, RW maandusega, kuna lugemist pole selles ülesandes vaja. Kirjutamise lubamiseks ühendada E digitaalpesaga 11. Ainult sümbolite ekraanile kuvamiseks 4-bitisest režiimist. Selleks ühendame pesad D4 -D7 digitaalpesadega 5-2. Mustad juhtmed tähistavad maandust, oranžid 5 V toidet. Taustavalgustuse anoodi (A) toidega ühendamisel kasutatakse takistit.



Joonis 71. LCD ekraani ja potentsiomeetri ühendamine.

Nüüd võib arendusplaadi USB juhtme abil arvutiga ühendada ning testida ekraani arenduskeskkonna näidisprogrammiga. Arenduskeskkonna LCD ekraani testimiseks mõeldud näited asuvad *File -> Exampes -> LyquidCrystal* valime nendest näiteks “Hello World” (Joonis 70).



Joonis 72. LCD ekraani näitekoodi avamine

Seejärel leida koodi setup () meetodist üles rida lcd.print("hello, world!"); ning asendada sõna *world* enda nimega. Peale koodi plaadile laadimist peaks ekraani ülemisele reale ilmuma kiri "hello, *teie nimi*" ning alumisele sekundiloendur. Keerata potentsiomeetrit nii palju vasakule kui võimalik, et tagada maksimaalne kontrastsus. Teksti mitteilmumise korral peaks üle kontrollima potentsiomeetri ühendused, taustavalguse probleemide korral kahe viimane pesa (A ja K) ühendused.

Esimene katsetus tehtud, saab proovida ekraanile joonistada endale meelepärast liikuvat pilti. Selleks avada erisümbolite näide *File -> Exampes -> LyquidCrystal ->CustomCharacter*, mille katsetamisel ilmub käsi üles-alla liigutav mehike. Ekraan ise koosneb 32 kastikesest (16 veergu 2 rida), millest igaüks on 5 x8 pikslit. Kujutiste kuvamine toimub 8-baitiste järjendite abil, kus iga baidi 5 parempoolset bitti defineerivad ühe pikslirea. Järgnevas koodiplokis on toodud CustomCharacker koodist inspireeritud liikuva pildi kood.

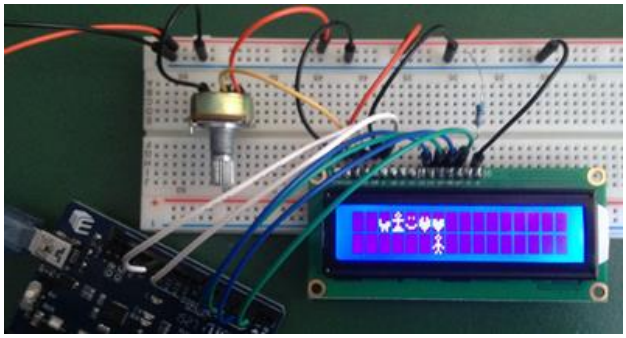
```
// lisame LiquidCrystal library
#include <LiquidCrystal.h>
// määrame, milliste digitaalpesadega ekraan ühendatud on
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
// loome erisümbolid
```

```
byte smiley[8] =
{0b00000,0b00000,0b01010,0b00000,0b00000,0b10001,0b01110,0b00000};
byte armsDown[8] =
{0b00100,0b01010,0b00100,0b00100,0b01110,0b10101,0b00100,0b01010};
byte armsUp[8] =
{0b00100,0b01010,0b00100,0b10101,0b01110,0b00100,0b00100,0b01010};
byte kass[8] =
{0b00000,0b00000,0b00000,0b10000,0b10011,0b11111,0b01110,0b01010};
byte tydruk[8] =
{0b00100,0b01010,0b00100,0b11111,0b00100,0b00100,0b01110,0b11111};
byte v2ikesyda[8] =
{0b00000,0b00000,0b00000,0b11011,0b11111,0b11111,0b01110,0b00100};

void setup() {
  // algväärtustatakse LDC ekraani read ja veerud
  lcd.begin(16, 2);
  // defineerime erisümbolid
  lcd.createChar(1, smiley);
  lcd.createChar(3, armsDown);
  lcd.createChar(4, armsUp);
  lcd.createChar(5, kass);
  lcd.createChar(6, tydruk);
  lcd.createChar(7,  v2ikesyda);
}

void loop() {
  // koht kuhu hakkame joonistama - 3. veerg, esimene rida
  lcd.setCursor(2, 0);
  // joonistame kassi
  lcd.write(5);
  // ja tüdruku
  lcd.write(6);
  //muudame kursori asukohta: 2. rida 7. veerg
  lcd.setCursor(6, 1);
  // joonistame poisi
  lcd.write(3);
  //ootame natuke
  delay(400);
  //muudame kursori asukohta: 1. rida 5. veerg
  lcd.setCursor(4, 0);
  //joonistame for tsükli abil  väikese pausiga 3 smaili
  for (int koht = 0; koht < 3; koht++) {
    lcd.write(1);
    delay(400);
  }
  //muudame kursori asukohta: 2. rida 7. veerg
  lcd.setCursor(6, 1);
  lcd.write(4);
  //loobime südameid
  for (int koht = 6; koht >= 4; koht--) {
  lcd.setCursor(koht, 0);
    lcd.write(7);
    delay(400);
```

```
  }
  delay(400);
}
```

Joonis 73. Kood LCD ekaanile väikse liikuva pildi loomiseks

Tulemuseks on liikuv pilt (Joonis 74), kus poiss ja tüdruk üksteiseke südameid ja naerunägusid saadavad.



Joonis 74. Hetk liikuvast pildist LCD ekraanil.

Näidet saab muuta nii, et tekiks soovitud temaatikaga lugu. Piltide ekraanil kuvatavaks koodiks muutmisel aitab näiteks selline lehekülg: https://omerk.github.io/lcdchargen/. Hetkel liigub pilt automaatselt ühtlase kiirusega kuni arenduslaadi toitest eemaldamiseni, lugu saaks huvitavamaks programmeerida, lisades reaktsioonid näiteks nupuvajutustele või anduritega saadavatele mõõtmistulemustele.

3.6 Helianduri ühendamine, LEDide vilgutamine vastavalt müratugevusele

Käesolevas alampeatükis näidatakse, kuidas ühendada heliandurit ja programmeerida valgusdioode müratugevusele reageerima.

Projekti koostamiseks läheb vaja:

1. 1x UNO mikrokontroller
2. 1 x USB kaabel
3. 1 x maketeerimislaud
4. 3 x vabalt valitud värvi LED-id
5. 3x 330 Ω takistid
6. 1 x heliandur (*Sound sensor*)
7. 4 x tavalised üksikud isane-isane juhtmed

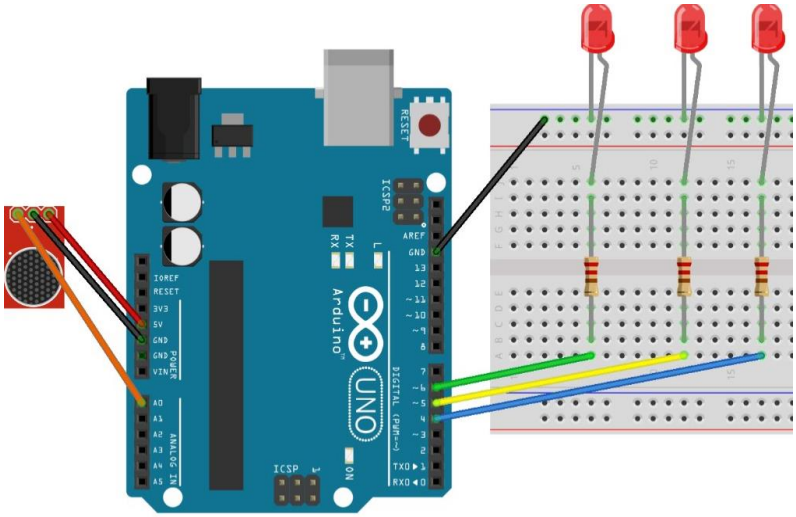
8. 3 x isane -emane juhtmed või 1x 10 soonega isane - emane kaabel, millest kasutame kolme pesa.

Enne komponentide ühendamist tuleb veenduda, et arendusplaat oleks arvuti küljest (ja muudest toidetest) lahti ühendatud.

Ühenda kolm LED-i rööpühenduses maketeerimislauale nii, et katood (LED-i lühem jalg) oleks ühendatud maketeerimislaual miinusreale ning anood (LED-i pikem jalg) ühendatud maketeerimilaua keskel takistiga ning seejärel digitaalsiginaaliga, antud näites on kasutusel digitaalpesad numbritega 4, 5 ja 6. Miinusreale tuleb lisada ka maandusjuhe (ühedatud digitaalpesade kohal olevase GND pesasse).

Nüüd kui LED-id ühendatud, tuleb paika panna peatükis 2.1 tutvustatud heliandur. Selleks tuleb vaadata helianduril olevaid märgistusi ja ühendada vastavalt sellele isane - emane juhtmed helianduri ja arendusplaadi vahel järgnevalt: 5 V toitega, GND maandusega, A alanoogsisendiga (antud näites kasutusel analoogpesa A0) ning D võib hetkel ühendamata jätta, kuna selles projektis me heliandurile digitaalühendiust ei vaja.

Kui kõik ühendused tehtud, siis võib need üle kontrollida jooniselt 73. Joonisel kujutatud helianduril on ainult kolm jalga, aga see ei muuda meie skeemi, kuna jooniselt puuduv digitaalväjund pole kasutusel.



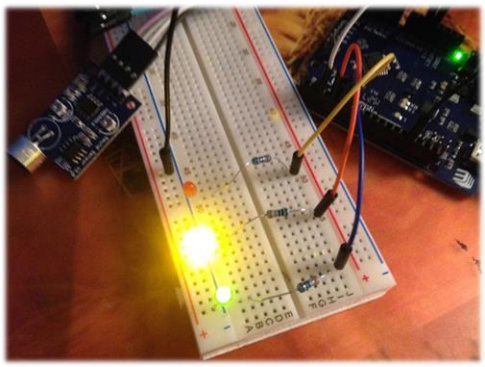
Joonis 75. Helianduri ja kolme LED-i ühendamine

Nüüd on aeg ühendada arendusplaat USB juhtme abil arvutiga ning avada arenduskeskkond, teha uus koodiaken ning kirjutada sinna järgnev kood (kommentaarid võib ära jätta).

```
// määrame algväärtused
int heliandur = A0;
int kynnis = 532;
int sensori_v22rtus = 0;

void setup() {
    //defineetime digitaalpesad, kuhu LED-id on ühendatud
    pinMode(4, OUTPUT);
    pinMode(5, OUTPUT);
    pinMode(6, OUTPUT);
}

void loop() {
    //loeme müra väärtust
    sensori_v22rtus = analogRead(heliandur);
    //kontrollime kas müra väärtus ületab konstandina antud künnise
    if (sensori_v22rtus >= kynnis) {
        //kui jah, siis süütame ühe LED-i ja jätame korraks põlema
        digitalWrite(4, HIGH);
        delay(200);
    }
    //kontrollime uusti kas müra väärtus ületab 1.1 kordse konstandina
antud künnise
    if (sensori_v22rtus >= (1.1*kynnis)) {
        //kui jah, siis süütame järgmise LED-i ja jätame korraks põlema
        digitalWrite(5, HIGH);
        delay(400);
    }
    //kontrollime uusti kas müra väärtus ületab 1.2 kordse konstandina
antud künnise
    if (sensori_v22rtus >= (1.2*kynnis)) {
    // kui jah, siis süütame järgmise LED-i ja jätame korraks põlema
        digitalWrite(6, HIGH);
        delay(600);
    }
    //vaastasel juhul kustutame LED-id
    else {
        digitalWrite(4, LOW);
        digitalWrite(5, LOW);
        digitalWrite(6, LOW);
    }
}
```

Joonis 76. Helile reageerivate valgusdioodide kood

Seejärel laadida kood plaadile ning testida projekti näiteks heliandurisse lauldes, muusikat lastes, puhudes või selle läheduses koputades (Joonis 77). Väiksema müra peale peaks süttima neljandasse pesase ühendatud LED, natuke suurema müra peale 5. ja veel suurema müra peale ka viimane.



Joonis 77. Helile reageerivate valgusdioodide testimine

Vaikuse korral peaksid kõik tuled korraga kustuma. Edasiarendusena võib proovida sama projekti suurema arvu valgusdioodidega.

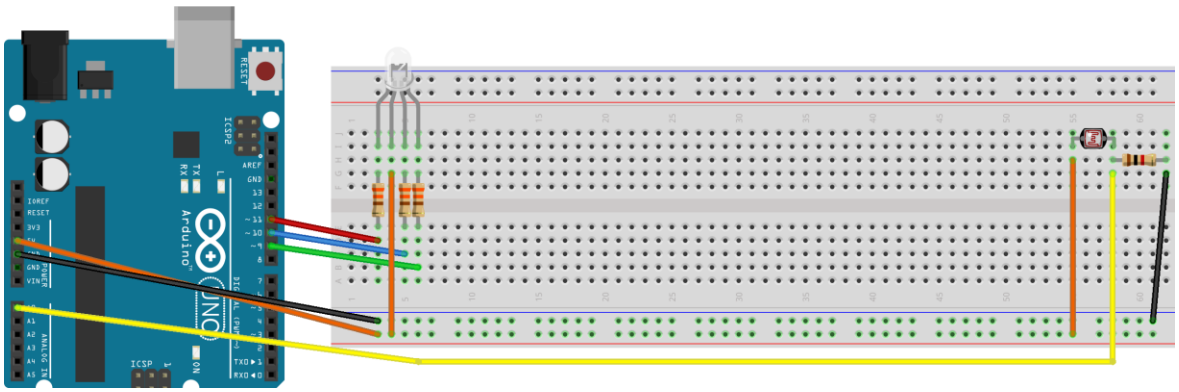
3.7 Fototakisti abil valguse mõõtmine, Serial monitoriga tutvumine, valgustundlik LED

Käesolevas punktis koostatakse juhend peatükis 2.1.8 tutvustatud fototakisti ühendamiseks. Näidatakse, kuidas mõõdetud analoogsignaali digitaalseks muuta ning Serial Monitori kaudu kuvada. Saadud tulemusi kasutades proovime seada RBG LED-i valgusele võimalikult silmasõbralikult reageerima.

Projekti koostamiseks läheb vaja:

1. 1x UNO mikrokontroller
2. 1 x USB kaabel
3. 1 x maketeerimislaud
4. 1x fototakisti
5. 1x 10 KΩ takisti
6. 1 x RGB LED
7. 3 x 330 Ω takisti
8. 8 x tavalised üksikud isane-isane juhtmed

Ühendada kõigepealt maketeerimislaua ühte äärde sarnaselt järgneva joonisega fototakisti nii et üks jalg oleks ühendatud toite ning teine läbi maandusesse mineva takisti analoogsignaaliga. Nüüd lisada maketeerimislaua teise otsa, et LED-i valgus ei mõjutaks andurit, alampeatükkidest 3.2 ja 3.3 tuttav RGB LED.



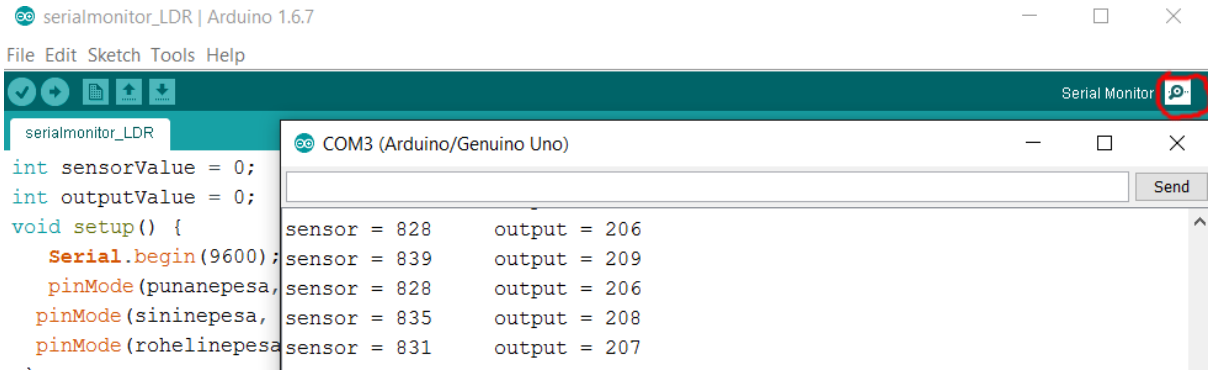
Joonis 78. RGB valgusdioodi ja fototakisti ühendamine

Kuna me ei tea milliste tulemustega fototakisti valgusele reageerib, siis testime esialgu ainult fototakistit, kasutades selleks järgnevas koodiplokis kujutatud AnalogInOut (File- > Examples -> Analog -> AnalogInOut) näitel põhinevat koodijuppi, kus kirjutame väärtused iga sekundi tagant Serial Monitori.

```
int LDR = 0; //defineerime analoogpesa, kuhu fototakisti on ühendatud
//algväärtused
int sensorValue = 0;
int outputValue = 0;
void setup() {
   Serial.begin(9600); //Serial monitori seadistamine
 }

 void loop() {
   sensorValue = analogRead(LDR); //loeme mõõdetud analoogväärtuse
   outputValue = map(sensorValue, 0, 1023, 0, 255); //kaardistame
selle analoogväljundi vahemikku

  // prindime tulemused Serial Monitori
  Serial.print("sensor = ");
  Serial.print(sensorValue);
  Serial.print("\t output = ");
  Serial.println(outputValue);
  delay(1000); // ootame sekundi
 }
```

Joonis 79. Kood fototakisti väärtuste Serial Monitori kirjutamiseks

Monitori jälgimiseks avada see arenduskeskkonna ülemise rohelise riba paremas ääres asuvast luubi ikoonist (Joonis 80).



Joonis 80. Serial Monitori avamine

Alampeatükkides 3.2 ja 3.3 selgitatud värvide segamist ja fototakisti abil saadud andmete analüüsi kombineerides saame koodi, mis reguleerib RGB LED-i värvust valguse vähenedes eredalt sinakasrohekalt soojemaks punakamaks ning prindib info valguse tugevuse kohta Serial Monitori.

```
const int LDR = 0; //defineerime analoogpesa, kuhu fototakisti on
ühendatud
int punanepesa = 11;
int rohelinepesa = 10;
int sininepesa = 9;

//algväärtused
int sensorValue = 0;
int outputValue = 0;
void setup() {
  Serial.begin(9600); //Serial monitori seadistamine
  pinMode(punanepesa, OUTPUT);
  pinMode(sininepesa, OUTPUT);
  pinMode(rohelinepesa, OUTPUT);
 }
 void loop() {
   sensorValue = analogRead(LDR); //loeme mõõdetud analoogväärtuse
   outputValue = map(sensorValue, 0, 1023, 0, 255); //kaardistame
selle analoogväljundi vahemikku
  // prindime tulemused Serial Monitori
  Serial.print("sensor = ");
  Serial.print(sensorValue);
  Serial.print("\t output = ");
  Serial.println(outputValue);
  segaV2rve((255-(outputValue/2)),outputValue,outputValue);
//reguleerib värvide kontrasti vastavalt signaalile
  delay(1000); // ootame sekundi
 }
 void segaV2rve(int punane, int roheline, int sinine)
{
  punane = 255 - punane;
  roheline = 255 - roheline;
```

```
  sinine = 255 - sinine;
  analogWrite(punanepesa, punane);
  analogWrite(rohelinepesa, roheline);
  analogWrite(sininepesa, sinine);
}
```

Joonis 81. Kood RBG LED-i värvi muutmiseks vastavalt valgustugevusele

Testimiseks on hea hoida andurit korraks pimedas ning seejärel proovida otsese ereda valguse peale laskmist näiteks taskulambi abil. Edasiarendusena võib proovida mõõtmistulemuste kuvamist LDC ekraanil nii et ekraani taustavalgus reageeriks nähtava valguse eredusele.

3.8 Juhtkangi ühendamine ja mootorite juhtimine

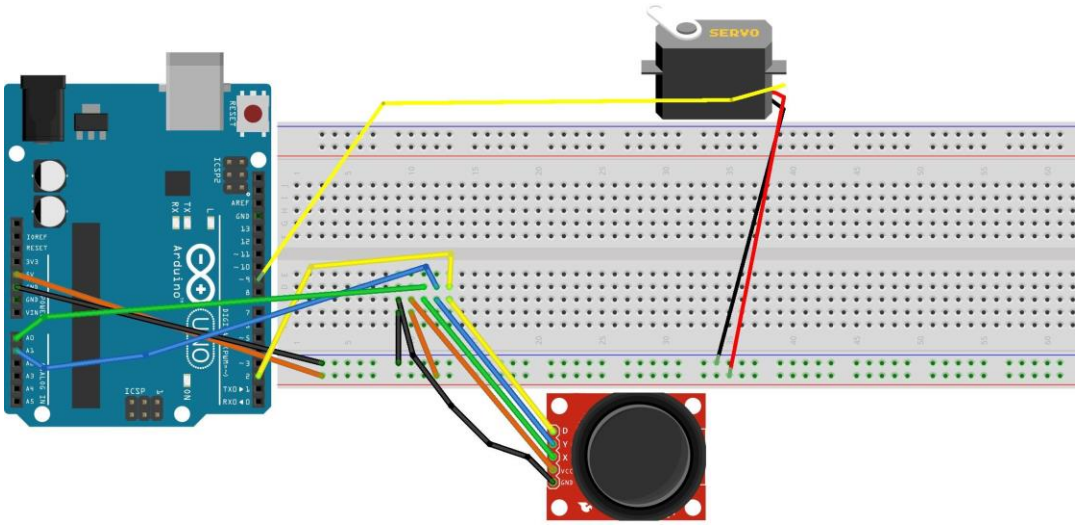
Käesolevas alampeatükis ühendatakse peatükis 2.1.6 tutvustatud mootoreid ning näidatakse, kuidas neid juhtkangi abil juhtida. Kasutusel on 180 kraadi pöörav servo mootor, mida saab juhtida näiteks lülitile vajutama ning samm-mootor, mille liikumiskiiruse üle omame kontrolli.

Projekti koostamiseks läheb vaja:

1. 1x UNO mikrokontroller
2. 1 x USB kaabel
3. 1 x maketeerimislaud
4. 1x juhtkang
5. 1x 180 kraadine servo mootor
6. 1 x sammmootor
7. 1 x sammmootori draiver
8. 15 x tavalised üksikud isane-isane juhtmed

Kõgepealt ühendame mootori ja juhtkangi sarnaselt joonisel 80 kujutatule makereerimislaua vahendusel arendusplaadiga. Kuna meie juhtkang on jalgadega, siis võib selle otse maketeerimislauale asetada, joonisel on jalgade asemel juhtmed, mis ei muuda ühenduste loogikat. Juhtkangi ühendamisel tuleb jälgida pesade märgistusi, kuna pesade järjekord võib erinevate tootjate toodetel olla erinev. Käesoleva komplekti tootel on maandus alumine (GND), järgmine 5 V toide (VCC), x- koordinaati tähtistav X ja y- koordinaat Y ühendatakse

analoogpesadega ning digitaalsignaal (D) tuleb ühendada digitaalpesaga. Servomootoril kirjalikud märgistused puuduvad, kuid sellest tulevad juhtmed on otstarbele kohaste värvidega (must maandus, punane toide ning kollakasorantžikas digitaalsignaal).



Joonis 82. Servo mootori ja juhkangi ühendamine

Katsetame areduskeskkonnaga kaasas oleva näitekoordi abil ainult mootori liikumist, selleks avame ja laadime plaadile näidiskoodi Sweep (File - >Examples ->Servo ->Sweep), mis liigutab mootorit ühele ja teisele poole.

Nüüd, kui nähtud kuidas mootor töötab, saab juurde kirjutada kangist juhtimise osa. Järgnevas koodiplokis kasutame 180 kraadise servo liigutamiseks ainult juhtkangi x-telge, alguses on meil defineeritud ka y pesa, see leiab kasutust järgmises ülesandes. Telgede väärtuste vaatamiseks kasutame alampeatükis 3.7 lähemalt tutvustatud Serial Monitori.

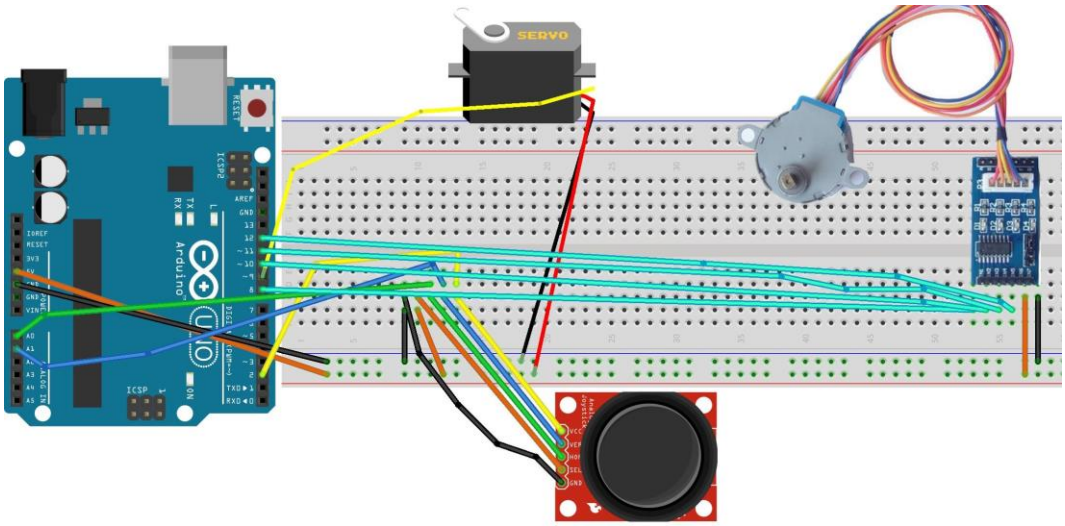
```
// By Explore Labs

#include <Servo.h> //servo lisad, mis lihtsustavad juhtimist
Servo servo; // servoobjekt servomootori kontrollimiseks

int joyX = A0; // juhtkangi X-teljega ühendatud analoogpesa
int joyY = A1; // juhtkangi Y-teljega ühendatud analoogpesa
int x, y; // muutujad analoogpesadest vastuste lugemiseks

void setup()
{
  servo.attach(9); // defineerime servo pini
  Serial.begin(9600); //defineerime Serial Monitori, et näha infot
liikumise kohta
}

void loop()
{
```

```
  x = map(analogRead(joyX), 0, 1023, 900, 2100); // loeme analoogpesast
x -telje väärtuse ja paneme selle servo jaoks sobivale skaalale
  //prindime vastuse kontrollimise eesmärgil serial monitorile
  Serial.print("x-telg: ");
  Serial.println(x);

  servo.write(x); // anname servole x telje väärtuse
  delay(15); // ootame servo liikumist
}
```

Joonis 83. Kood servo mootori juhtimiseks juhtkangiga

Nüüd peaks võimalik olema juhtkangist servo mootorit edasi-tagasi liigutada, selline fonktsionaalsus on kasulik näiteks eemalt lüliti või mõne nupu klõpsamiseks. Hetkel oli y-telg defineeritud, aga väärtusi sealt ei loetud, kuna ühe mootori juhtimiseks piisab ühest teljest.

Kasutamaks suuremat osa juhtkangi võimalustest, ühendame maketeerimislauale lisaks ühe samm-mootori. Ühendamine on mingil määral sarnane servo mootori ühendamisega, kuid erinevalt servost tuleb vahele lisada ka draiver nii, et mootorist tulevad juhtmed ühendatakse sarnaselt joonisel 82 kujutatule draiveri külge ning draiver omakorda maketeerimislauale. Juhtmed tuleb ühendada vastavalt draiveril olevatele märgistustele: GND maandusega, 5V toitega ning ülejäänud digitaalpesadega.



Joonis 84. Juhtkangi, sevo ja samm-mootori korraga ühendamine

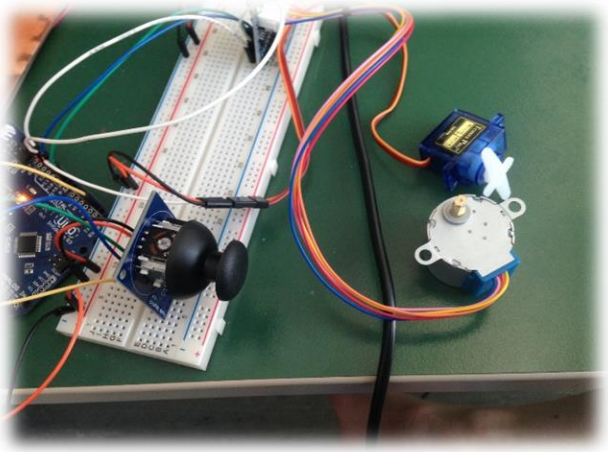
Täiustame eelmist koodi nii, et sammmootori kiirus sõltuks juhtkangi y-teljest.

```
/*
 Stepper Motor Control - speed control
 Created 30 Nov. 2009
 Modified 28 Oct 2010
 by Tom Igoe
Servo osa- By Explore Labs

 */
```

```
//servo ja stepperi lisad, mis lihtsustavad nende mootorite juhtimist
#include <Servo.h>
#include <Stepper.h>

const int stepsPerRevolution = 200;  //sõltub mootori tüübist, antud
mootor töötab 200 spr
Stepper myStepper(stepsPerRevolution, 8, 10, 11,12); //defineerime
sammmootori pinid, pinide valik selline, kuna 9. oli juba servo jaoks
kasutusel
Servo servo; // servoobjekt servomootori kontrollimiseks

int joyX = A0; // juhtkangi X-teljega ühendatud analoogpesa
int joyY = A1; // juhtkangi Y-teljega ühendatud analoogpesa
int x, y; // muutujad analoogpesadest vastuste lugemiseks

void setup()
{
  servo.attach(9); // defineerime servo pini
  Serial.begin(9600); //defineerime Serial Monitori, et näha infot
liikumise kohta
}

void loop()
{
  x = map(analogRead(joyX), 0, 1023, 900, 2100); // loeme analoogpesast
x -telje väärtuse ja paneme selle servo jaoks sobivale skaalale
  y = map(analogRead(joyY), 0, 1023, 0, 100); // loeme analoogpesast y -
telje väärtuse ja paneme selle sammmootori jaoks sobivale skaalale
  //prindime vastused kontrollimise eesmärgil serial monitorile
  Serial.print("x-telg: ");
  Serial.print(x);
  Serial.print("y-telg: ");
  Serial.println(y);

  if (y > 0) {
  myStepper.setSpeed(y/2); //muudame sammmootori kiirust vastavalt
juhtkangi y telje väärtusele
  myStepper.step(stepsPerRevolution / 100); // liikutame sammmootorit
  }

  servo.write(x); // anname servole x telje väärtuse
  delay(15); // ootame servo liikumist
}
```

Joonis 85. Servo ja samm-mootori korraga juhtimine

Nüüd saab juhtkangi abil korraga juhtida mõlemat mootorit (Joonis 86).



Joonis 86. Juhtkang ja mootorid

Parema ettekujutuse sammmootori liikumiskiirusest saab, kui selle külge kinnitada sobiv ratas, ratta puudumisel võib liikuva osa külge paberist väikese lipukese meisterdada.

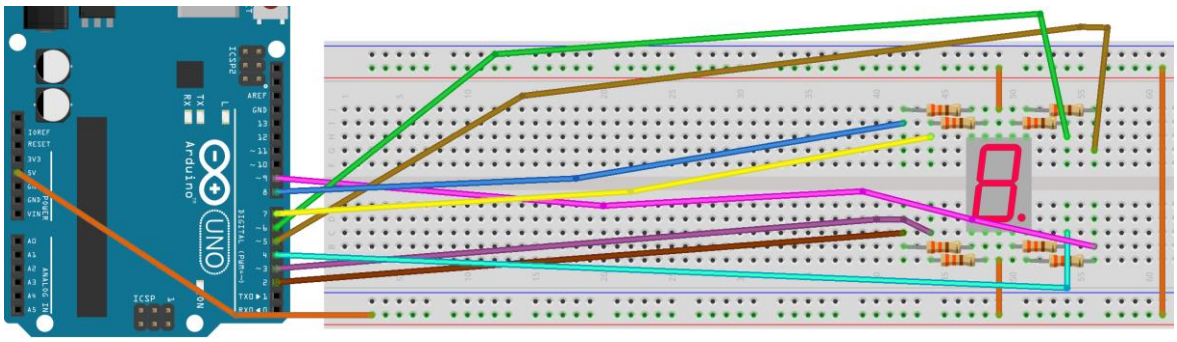
3.9 Infrapuna vastuvõtja, pult ja 7-segmendiline ühe numbriga LED

Käesolevas alampeatükis näidatakse, kuidas ühendada peatüki 2.1 alampunktides tutvustatud infrapuna vastuvõtjat ja 7- segmendilist 1-numbrilist valgusdioodi. Tänu infrapuna vastuvõtjale saame LED-i eemalt puldiga kontrollida, kas käivitada LED-il jooksvat numbrite tsüklit või kuvada valgusdioodil hoopis puldil vajutatud numbrit. Puldilt tulevate signaalide vahetamiseks sobib hästi programmeerimises tuntud SWITCH - lause kasutamine.

Projekti koostamiseks läheb vaja:

1. 1x UNO mikrokontroller
2. 1 x USB kaabel
3. 1 x maketeerimislaud
4. 1x 7-segmendiline ühise anoodiga numberLED
5. 1x pult
6. 1x infrapuna vastuvõtja
7. 8x 330 oom takisti
8. 16 x tavalised üksikud isane-isane juhtmed

Ühendame kõigepealt 7-segmendilise LED-i sarnaselt joonisel 85 kujutatule maketeerimislauale nii, et iga digitaalpesaga ühenduse vahel oleks takisti. Kuna tegu on ühise anoodiga LED-iga, siis tuleb pesad 3 ja 8 ühendada toitega.



Joonis 87. 7-segmendilise valusdioodi ühendamine

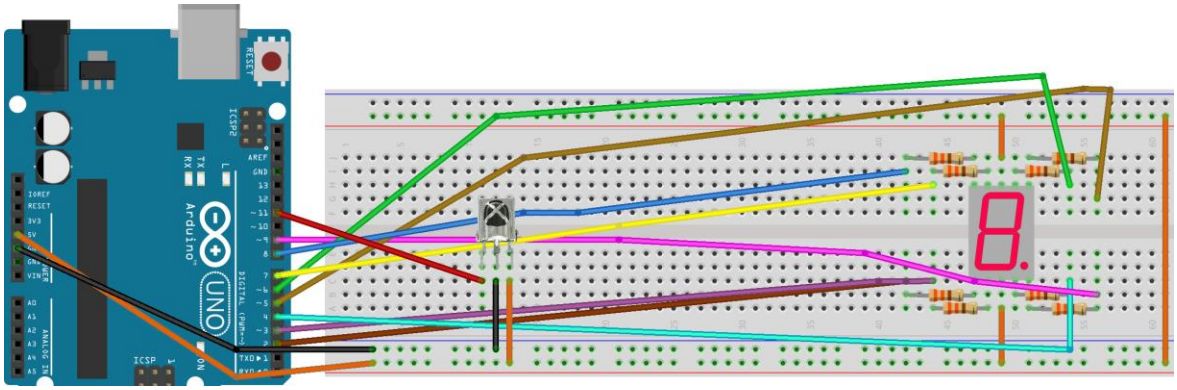
Proovime sellel järgneva koodi abil numbreid kujutada ja eelmistest alampeatükkidest tuttava FOR tsükli abil neid üheksast nullini lugema panna.

```
// Arduino 7 segment display example software
// http://www.hacktronics.com/Tutorials/arduino-and-7-segment-led.html
// License: http://www.opensource.org/licenses/mit-license.php
// defineeritakse numbrite koostamiseks vajalikud bitikombikatsioonid
//kuna tegu ühise anoodiga, siis 1 tähendab välja ja 0 sisse lülitatud
byte seitsme_seg_numbrid[10][7] = { { 0,0,0,0,0,0,1 },  // = 0
                                    { 1,1,0,0,1,1,1 },  // = 1
                                    { 0,0,1,0,0,1,0 },  // = 2
                                    { 1,0,0,0,0,1,0 },  // = 3
                                    { 1,1,0,0,1,0,0 },  // = 4
                                    { 1,0,0,1,0,0,0 },  // = 5
                                    { 0,0,0,1,0,0,0 },  // = 6
                                    { 1,1,0,0,0,1,1 },  // = 7
                                    { 0,0,0,0,0,0,0 },  // = 8
                                    { 1,0,0,0,0,0,0 }   // = 9
                                                 };


void setup() {          //seadistame inid 2-8 väljunditena
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  lisaPunkt(0); //lülitame punkti ka sisse
}
//funktsioon punkti sisse lülitamiseks
void lisaPunkt(byte punkt) {
  digitalWrite(9, punkt); //punkt on ühendatud digitaalpesasse 9
}
```

```
//numbrite kirjutamise funktsioon, argumendiks võetakse kirjutatav
number
void kirjutaNumbrid(byte arv) {
  byte pesa = 2; //esimene pesa, mis digitaalpesadest LED-il kasutusel
on
//tsükkel numbrite kirjutamiseks eelnevalt defineeritud muutuja abil
  for (byte segLoendur = 0; segLoendur < 7; ++segLoendur) {
  digitalWrite(pesa, seitsme_seg_numbrid[arv][segLoendur]);
  ++pesa;
  }
}

void loop() {
//tsükkel numbrite 9 kuni 0 lugemiseks 1 sekundise vahega
  for (byte loendur = 10; loendur > 0; --loendur) {
   delay(1000);
   kirjutaNumbrid(loendur - 1);
  }
  delay(4000); //tsükki lõppedes tehakse 4 sekudine paus
}
```

Joonis 88. Kood numbrite kuvamiseks 7-segmendilisel valgusdioodil

Hetkel töötab programm kohe arendusplaadi toitega ühendamise hetkest ning tsüklina pidevalt uuesti kuni toite eemaldamiseni. Loenduri üle suurema kontrolli saavutamiseks ühendame lisaks ka infrapuna vastuvõtja, mis annab meile võimaluse koodi puldi abil käivitada või lihtsalt enda soovitud numbreid kuvada. Vastuvõtja tuleb ühendada, sarnaselt järgneval jonisel kujutatule, maanduse toite ja ühe digitaalpesaga, selles näites kasutatud pesa number 11.



Joonis 89. Infrapuna vastuvõtja ühendamine koos 7-segmendilise LED-iga

Lisame puldiga juhitavuse osa eelnevale koodile juurde. Kõigepealt peame ära asendama ka ühe teegi, kuna Arduinoga kaasas olev RobotIRremote library tekitab konflikti. Selleks tuleb tõmmata *Infrared remote library* aadressilt: https://github.com/z3t0/Arduino-IRremote ning asendada sellega Arduino *libraries* kaustas (tavaliselt *C:\Program Files (x86)\Arduino\libraries*) asuv RobotIRremote kaust.

Nüüd peaks IRremote teek ilusti töötama ja saame proovida 7-segmendilist LED-i puldist lülitatavaks muuta. Selleks lisame eelmisele koodile IR vastuvõtja seadistamiseks vajalikud muutujad ja puldist tulevatele signaalidele reageerimise osa. Selleks kasutame *SWITCH*-lauset, kus ühele sündmusele vastava iga erineva juhu (*case*) korral tehakse mingi tegevus, kui ükski defineeritud juht ei esine, siis saadetakse korda *default* plokis olev tegevus.

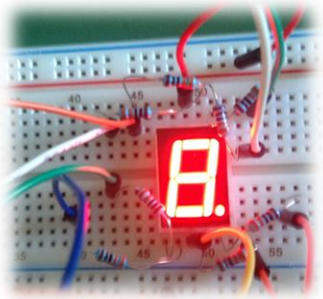
```
// LED-i osa: Copyright Arduino 7 segment display example software
// http://www.hacktronics.com/Tutorials/arduino-and-7-segment-led.html
// License: http://www.opensource.org/licenses/mit-license.php
// IR vastuvõtja osa: Copyright https://brainy-bits.com/tutorials/ir-
remote-arduino/
#include "IRremote.h" //teek infrapunasignaaliga toimetamiseks

int RECV_PIN = 11; //defineerime digitaapesa, millega vastuvõtja on
ühendatud

IRrecv irrecv(RECV_PIN);   // loome vastuvõtja muutja
decode_results results;   // vastuse muutuja

//                                pesad: 2,3,4,5,6,7,8
byte seitsme_seg_numbrid[10][7] = { { 0,0,0,0,0,0,1 },  // = 0
                                    { 1,1,0,0,1,1,1 },  // = 1
                                    { 0,0,1,0,0,1,0 },  // = 2
                                    { 1,0,0,0,0,1,0 },  // = 3
                                    { 1,1,0,0,1,0,0 },  // = 4
                                    { 1,0,0,1,0,0,0 },  // = 5
                                    { 0,0,0,1,0,0,0 },  // = 6
                                    { 1,1,0,0,0,1,1 },  // = 7
                                    { 0,0,0,0,0,0,0 },  // = 8
                                    { 1,0,0,0,0,0,0 }   // = 9
                                                 };

void setup() {
  Serial.begin(9600); //käivitame serial monitori
  irrecv.enableIRIn(); // käivitame vastuvõtja
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(9, OUTPUT);
  writeDot(0);
}
void writeDot(byte dot) {
  digitalWrite(9, dot);
}

void kirjutaNumbrid(byte arv) {
```

```
  byte pesa = 2; //esimene pesa, mis digitaalpesadest LED-il kasutusel
on
//tsükkel numbrite kirjutamiseks eelnevalt defineeritud muutuja abil
  for (byte segLoendur = 0; segLoendur < 7; ++segLoendur) {
  digitalWrite(pesa, seitsme_seg_numbrid[arv][segLoendur]);
  ++pesa;
  }
}

void loop() {
  if (irrecv.decode(&results)) // kui saame infrapuna signaali
  {
  translateIR(); //kutsume välja funktsiooni, kus tuvastame signaali
  Serial.println(results.value, HEX); //prindime saadud signaali koodi
serial monitori
  irrecv.resume(); // ootame uut signaali
  }
}
//numbtire 9-0 lugemise funktsioon
void loe(){
  for (byte loendur = 10; loendur > 0; --loendur) {
   delay(1000);
   kirjutaNumbrid(loendur - 1);
  }
}
 // IR käskue tõlkimise funktsioon
void translateIR() {
 //kasutame switch käsku, et vahetada tegevusi vastavalt puldi
signaalile, vastuse kirjutame ka serial monitori
  switch(results.value)
  {
  case 0xFF629D: Serial.println(" FORWARD"); break;
  case 0xFF22DD: Serial.println(" LEFT"); break;
  case 0xFF02FD: Serial.println(" -OK-");   loe(); break; //käivitame
numbrite lugemise 9-0
  case 0xFFC23D: Serial.println(" RIGHT");   break;
  case 0xFFA857: Serial.println(" REVERSE"); break;
  //numbriklahvide vajutamisel kuvame soovitud numbri
  case 0xFF6897: Serial.println(" 1");  kirjutaNumbrid(1);  break;
  case 0xFF9867: Serial.println(" 2");  kirjutaNumbrid(2);  break;
  case 0xFFB04F: Serial.println(" 3");  kirjutaNumbrid(3); break;
  case 0xFF30CF: Serial.println(" 4");  kirjutaNumbrid(4);  break;
  case 0xFF18E7: Serial.println(" 5");  kirjutaNumbrid(5);  break;
  case 0xFF7A85: Serial.println(" 6");  kirjutaNumbrid(6); break;
  case 0xFF10EF: Serial.println(" 7");  kirjutaNumbrid(7); break;
  case 0xFF38C7: Serial.println(" 8");  kirjutaNumbrid(8);  break;
  case 0xFF5AA5: Serial.println(" 9");  kirjutaNumbrid(9);  break;
  case 0xFF42BD: Serial.println(" *");  break;
  case 0xFF4AB5: Serial.println(" 0");  kirjutaNumbrid(0);  break;
  case 0xFF52AD: Serial.println(" #");  break;
  case 0xFFFFFFFF: Serial.println(" REPEAT");break;
  default:
  Serial.println(" muu nupp  "); //tundmatu nupu korral
```

```
  }
  delay(500);
}
```

Joonis 90. 7-segmendilise LED-i juhtimine puldi abil

Nüüd peaks LED puldil iga erineva numbri peale kuvama valitud numbri ning "OK" klõpsamise peale alustama numbrite üheksast alla lugemise tsüklit.



Joonis 91. Number 8 kuvamine 7-segmendilisel näidikul

Erinevatel pultidel on signaali vastused erinevad, puldilt saadud signaal kirjutatakse ka alampeatükis 3.7 lähemalt tutvustatud Serial Monitori ning soovi korral on võimalik *case*-tingimusi muutes antud kood mõne teise puldi jaoks ümber programmeerida.

3.10 Temperatuuri kuvamine 4-kohalisel 7-segmendilisel näidikul

Käesolevas punktis näidatakse, kuidas ühendada peatükis 2.1 alampeatükkides tutvustatud temperatuuriandurit ja 7- segmendilist 4-numbrilist valgusdioodi. Koostatakse juhend temperatuuri mõõtmiseks LM35 temperatuurianduri abil ning kuvatakse saadud tulemused näidikule.

Projekti koostamiseks läheb vaja:

1. 1x UNO mikrokontroller
2. 1 x USB kaabel
3. 1 x maketeerimislaud
4. 1x 7-segmendiline 4-kohaline valgusdiood
5. 1x temperatuuriandur
6. x x tavalised üksikud isane-isane juhtmed
7. X x 330 oom takisti

Ühendame kõigepealt 7-segmendilise valgusdioodi sarnaselt joonisel xx kujutatule maketeerimislauale nii, et iga digitaalpesaga ühenduse vahel oleks takisti ning pesadevahelised ühendused vastaksid tabelile xx.

http://duinos.net/files/2015/common_anode_4digit_7_segment_led_display.ino

```
/*
  Showing number 0-9 on a Common Anode 7-segment LED display
  Displays the numbers 0-9 on the display, with one second inbetween.
    A
   ---
F |   | B
  | G |
   ---
E |   | C
  |   |
   ---
    D
  This example code is in the public domain.
 */

// Pin 2-8 is connected to the 7 segments of the display.
int starttime =0;
int endtime =0;
int pinA = 2;
int pinB = 3;
int pinC = 4;
int pinD = 5;
int pinE = 6;
int pinF = 7;
int pinG = 8;
int D1 = 9;
int D2 = 10;
int D3 = 11;
int D4 = 12;

// the setup routine runs once when you press reset:
void setup() {
  Serial.begin(9600);
    pinMode(0, INPUT);
  // initialize the digital pins as outputs.
  pinMode(pinA, OUTPUT);
  pinMode(pinB, OUTPUT);
  pinMode(pinC, OUTPUT);
  pinMode(pinD, OUTPUT);
  pinMode(pinE, OUTPUT);
  pinMode(pinF, OUTPUT);
  pinMode(pinG, OUTPUT);
  pinMode(D1, OUTPUT);
  pinMode(D2, OUTPUT);
  pinMode(D3, OUTPUT);
  pinMode(D4, OUTPUT);
```

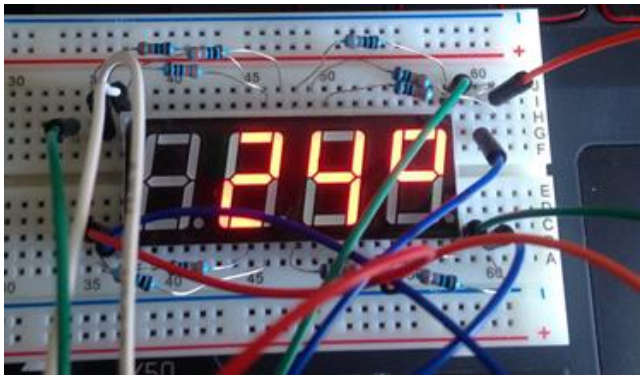
```
}

// the loop routine runs over and over again forever:
void loop() {
 int data = analogRead(0);
  float millivolts = (data / 1024.0) * 5000;
  int celsius = millivolts / 10;

byte num =celsius % 10;
Serial.print(num);

  Serial.print(celsius);
  Serial.println(" kraadi");
 starttime = millis();
endtime = starttime;
while ((endtime - starttime) <=1000) // do this loop for up to 1000mS
{

  digitalWrite(D1, HIGH);
  digitalWrite(D2, LOW);
  digitalWrite(D3, LOW);
  digitalWrite(D4, LOW);
  //0
if (celsius <0){
  digitalWrite(pinA, HIGH);
  digitalWrite(pinB, HIGH);
  digitalWrite(pinC, HIGH);
  digitalWrite(pinD, HIGH);
  digitalWrite(pinE, HIGH);
  digitalWrite(pinF, HIGH);
  digitalWrite(pinG, LOW);
  delay(1);               // wait for a second
  }
  digitalWrite(D1, LOW);
  digitalWrite(D2, HIGH);
  digitalWrite(D3, LOW);
  digitalWrite(D4, LOW);


  //1
  if (celsius > 9 and celsius < 20 ){
  kirjutaNumber(1);
  delay(1);               // wait for a second
  }
  else if (celsius > 20 and celsius < 30){
     kirjutaNumber(2);
  delay(1);               // wait for a second
    }
  digitalWrite(D1, LOW);
  digitalWrite(D2, LOW);
  digitalWrite(D3, HIGH);
  digitalWrite(D4, LOW);
  //2
  kirjutaNumber(num);
```

```
  delay(1);               // wait for a second

  digitalWrite(D1, LOW);
  digitalWrite(D2, LOW);
  digitalWrite(D3, LOW);
  digitalWrite(D4, HIGH);
  //3
  digitalWrite(pinA, LOW);
  digitalWrite(pinB, LOW);
  digitalWrite(pinC, HIGH);
  digitalWrite(pinD, HIGH);
  digitalWrite(pinE, HIGH);
  digitalWrite(pinF, LOW);
  digitalWrite(pinG, LOW);
  delay(1);               // wait for a second

endtime = millis();
}


}

void kirjutaNumber(byte arv) {
  if (arv ==0){
     digitalWrite(pinA, LOW);
  digitalWrite(pinB, LOW);
  digitalWrite(pinC, LOW);
  digitalWrite(pinD, LOW);
  digitalWrite(pinE, LOW);
  digitalWrite(pinF, LOW);
  digitalWrite(pinG, HIGH);
    }
   else if(arv ==1){
     digitalWrite(pinA, HIGH);
  digitalWrite(pinB, LOW);
  digitalWrite(pinC, LOW);
  digitalWrite(pinD, HIGH);
  digitalWrite(pinE, HIGH);
  digitalWrite(pinF, HIGH);
  digitalWrite(pinG, HIGH);
    }
    else if(arv ==2){
       digitalWrite(pinA, LOW);
  digitalWrite(pinB, LOW);
  digitalWrite(pinC, HIGH);
  digitalWrite(pinD, LOW);
  digitalWrite(pinE, LOW);
  digitalWrite(pinF, HIGH);
  digitalWrite(pinG, LOW);
    }
    else if(arv ==3){
      digitalWrite(pinA, LOW);
  digitalWrite(pinB, LOW);
```

```
  digitalWrite(pinC, LOW);
  digitalWrite(pinD, LOW);
  digitalWrite(pinE, HIGH);
  digitalWrite(pinF, HIGH);
  digitalWrite(pinG, LOW);
    }
    else if(arv ==4){
       digitalWrite(pinA, HIGH);
  digitalWrite(pinB, LOW);
  digitalWrite(pinC, LOW);
  digitalWrite(pinD, HIGH);
  digitalWrite(pinE, HIGH);
  digitalWrite(pinF, LOW);
  digitalWrite(pinG, LOW);
    }
    else if(arv ==5){
  digitalWrite(pinA, LOW);
  digitalWrite(pinB, HIGH);
  digitalWrite(pinC, LOW);
  digitalWrite(pinD, LOW);
  digitalWrite(pinE, HIGH);
  digitalWrite(pinF, LOW);
  digitalWrite(pinG, LOW);
    }
    else if(arv ==6){
      digitalWrite(pinA, LOW);
  digitalWrite(pinB, HIGH);
  digitalWrite(pinC, LOW);
  digitalWrite(pinD, LOW);
  digitalWrite(pinE, LOW);
  digitalWrite(pinF, LOW);
  digitalWrite(pinG, LOW);
    }
    else if(arv ==7){
      digitalWrite(pinA, LOW);
  digitalWrite(pinB, LOW);
  digitalWrite(pinC, LOW);
  digitalWrite(pinD, HIGH);
  digitalWrite(pinE, HIGH);
  digitalWrite(pinF, HIGH);
  digitalWrite(pinG, HIGH);
    }
    else if(arv ==8){
       digitalWrite(pinA, LOW);
  digitalWrite(pinB, LOW);
  digitalWrite(pinC, LOW);
  digitalWrite(pinD, LOW);
  digitalWrite(pinE, LOW);
  digitalWrite(pinF, LOW);
  digitalWrite(pinG, LOW);
    }
    else if(arv ==9){
      digitalWrite(pinA, LOW);
  digitalWrite(pinB, LOW);
```

```
  digitalWrite(pinC, LOW);
  digitalWrite(pinD, HIGH);
  digitalWrite(pinE, HIGH);
  digitalWrite(pinF, LOW);
  digitalWrite(pinG, LOW);
    }

  }
```

Joonis 92. Kood termomeetri koostamiseks



Joonis 93. Termomeeter

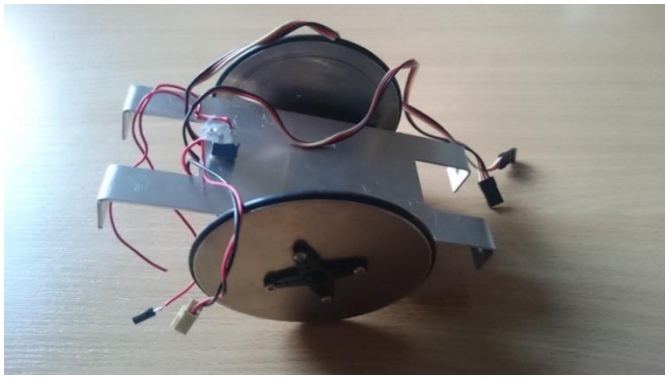
Edasiarendusena võib proovida viimase punkti sisse lülitamist ning temperatuuri kuvamist ühe komakoha täpsusega. Samuti võib mõõtmistulemust täpsemaks muuta, lisades õiged ümardamisreeglid. Nüüdseks on meil piisavalt kogemust Arduinoga et asuda edasi roboti ehitamise juurde.

4 Arduino roboti ehitamine

Arduino roboti mehhaanika ehitamiseks on mitmeid erinevaid viise. Käesolevas dokumendis kasutame mootoriteks digitaaljuhtimisega servo mootoreid, mida võib ühendada otse Arduino sisend-väljundviikude külge. Roboti kere võib ehitada erinevatest materjalidest näiteks plastkarbist (pikemalt käsitletud NUTIROBOT nutirobot.ut.ee projektis), 3D printida (http://www.thingiverse.com/thing:18264 või http://www.thingiverse.com/thing:4954) või kasutada mõnda muud olemasolevat platvormi (nagu käesolevas juhendis).

4.1 Arduino ühendamine mootoritega

Käesolevas töös kasutatakse juba eelnevalt metallist valmis tehtud kahe rattaga alust, millele saab peale paigutada Arduino arendusplaadi (vt joonis 94).

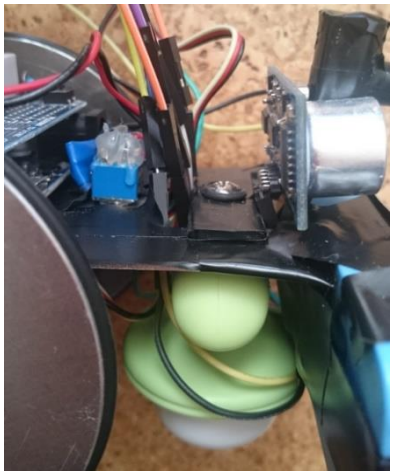


Joonis 94. Kahe rattaga alus robotile

Rattaid juhivad kaks servomootorit, mille saab Arduinoga ühendada. Kuna kahe rattaga ei ole robot tasakaalus, on lisatud ette veel kolmas ratas. Kuna see ei ole mootoriga ühendatud, on oluline, et see liiguks kergesti igas suunas. Seetõttu on ratta alumises osas kuul. Ratas selle töö jaoks on ostetud poest Tiger (kuid iga nn. Kolmas ratas peaks sobima). Et ratas paremini fikseeritud oleks, on ratta plastmassist osal (vt joonisel 23 roheline osa) ümarad kohad lamedamaks lihvitud.

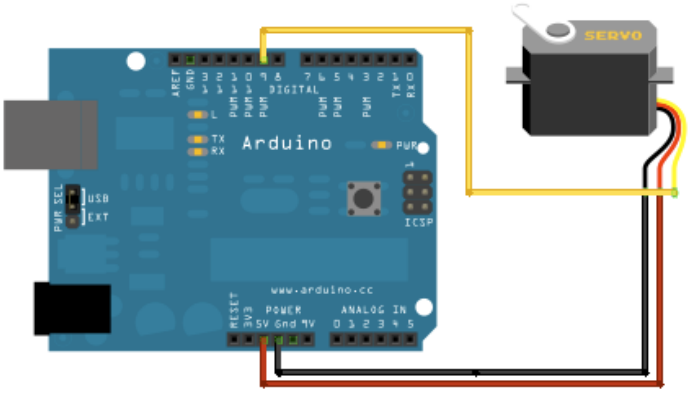
Joonis 95. Lihvitud plastmassist osaga ratas

Ratta roboti aluse külge kinnitamiseks on kasutatud isoleerteibiga kaetud alumiiniumist plaati. Plaadi ja ratta plastmassist osa sisse on tehtud augud. Ratas on roboti alusega ühendatud nii, et alus on alumiiniumplaadi ja ratta vahel ning plaat on ühendatud ratta plastmassist osaga kruvi abil (vt joonis 96)



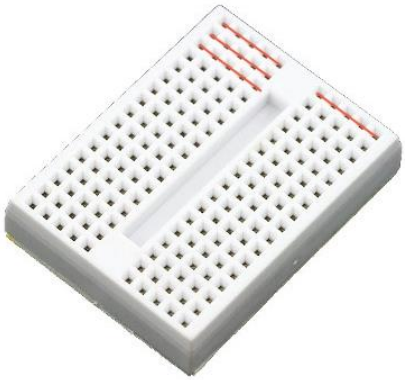
Joonis 96. Ratta ühendamine robotiga

Servomootori ühendamiseks Arduinoga on vaja kolme juhet: VCC (toide), GND (maandus) ja signaal. VCC juhe on punane, GND must ning signaal kollane (vt joonis 97).



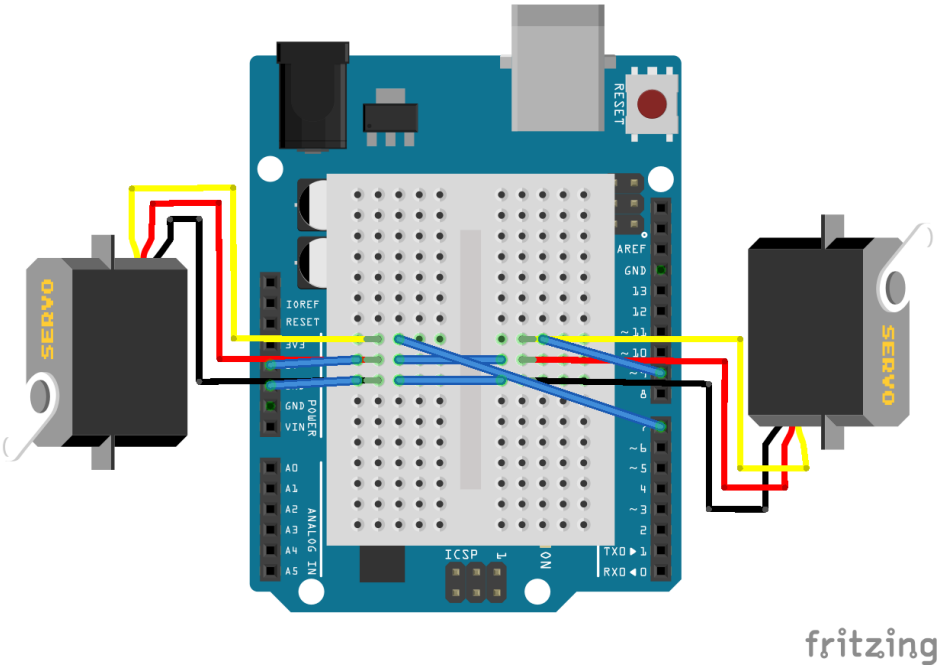
Joonis 97. Arduino ühendamine servomootoriga

Ühendada on vaja kaks servomootorit. Arduino arendusplaadil pole selleks piisavalt pesasid. Puudu on üks VCC (joonisel 97 on selle pesa tähis 5V). Signaali edasi kandmiseks kasutame prototüüpimise laiendusplaati, millega saab Arduino peale lisada pisikese maketeerimislaua. Maketeerimislaual on rajad ühenduses maketeerimislaua keskel oleva osa suhtes risti. Signaali katkestab maketeerimislaua keskel olev osa (vt joonis 98).



Joonis 98. Signaali kandumine maketeerimislaual (punased jooned joonisel)

Kuna maketeerimislaua keskel olev osa katkestab signaali, on joonisel 99 toodud VCC (punane) ja GND (must) signaal maketeerimislaua vasakult poolelt üle paremale poolele. Vasakpoolne servomootor on ühendatud pesaga 7 ning parempoolne pesaga 9. Neid kasutame hiljem programmi kirjutamises servomootorite ühendamiseks, et saaks neid vajadusel liigutada.



Joonis 99. Kahe servomootori ühendamine Arduino maketeerimisaluse ja arendusplaadiga

Servomootorite liigutamiseks tuleb kirjutada programm, mis tuvastab servomootorid ning neid vajadusel liigutab.

4.2 Mootorite liigutamine

Servomootorite kasutamiseks tuleb kasutada välist teeki, mis on Arduino IDE-s juba olemas. Selleks tuleb valida ülemisest menüüst Visand, seejärel *Include Library* ning sealt valida Servo. Nüüd peaks programmi tekkima rida *#include <Servo.h>*. Enne *setup()* funktsiooni kirjutamist defineerime kaks Servo objekti, *servoLeft* ja *servoRight*. Selleks tuleb kirjutada lihtsalt read *Servo servoLeft* ning *Servo servoRight*. Nüüd hakkame *setup()* funktsiooni kirjutama.

Selleks, et programm teaks, milliste füüsilise pesadega servomootorid ühendatud on, tuleb kirjutada read *servoLeft.attach(7)* ning *servoRight.attach(9)*. Nüüd on servomootorid ühendatud ning neid saab hakata liigutama.

Antud töös kasutatakse 360-kraadi servomootoreid. Nende liigutamiseks tuleb servo objekti kirjutada arv vahemikus 0-180. Arv 0 tähendab, et mootor liigub ühes suunas täiskiirusel, arvuga 180 liigub teises suunas täiskiirusel ning 90 tähendab, et mootor jääb seisma. Arvu 90 asemel võib olla sellest veidi erinev arv ning seda tuleb lihtsalt proovida. Antud töös kasutatud servomootoritel on selleks 93.

Servomootorite katsetamiseks tuleb teha programm, mis paneb roboti ruudus sõitma (vt joonis 28). Selleks tuleb sõita natuke aega edasi, siis pöörata ning korrata tegevust. Edasi sõitmiseks, pööramiseks ja roboti peatamiseks tuleb teha eraldi meetodid, mida hakata meetodis *loop()* välja kutsuma. Edasi sõitmiseks tuleb parem servomootor panna asendisse 180 ning teine asendisse 7, et mõlemad mootorid liiguksid samal kiirusel. Ka neid arve tuleb katsetada, et robot liiguks otse. Paremale pööramiseks tuleb mõlemad servomootorid panna asendisse 180, kuna servomootorid on ühendatud Arduinoga nii, et üks on suunaga roboti ette ja teine taha. Roboti peatamiseks tuleb mõlemad servomootorid panna asendisse 93.

Asendi määramiseks servomootoril tuleb kirjutada vasaku servomootori puhul rida *servoLeft.write(x)*, kus x on asendi väärtus, ning analoogne parema servomootori jaoks. Pärast asendite määramist tuleb hetkeks programm peatada reaga *delay(x)*, kus x on peatamise aeg millisekundites. Programmi peatamine on vajalik selleks, et programm ei läheks edasi uue käsu täitmisega enne, kui eelmine on täidetud. Näiteks paremale pöörates tuleb hoida servomootorites asendeid seni, kuni on keeratud 90 kraadi. Selleks kuluva aja saab teada katsetades. Antud töös kasutatud servomootorite puhul on selleks 400 millisekundit.

Edasi sõitmise meetod (*forward(int n)*) ja paremale sõitmise meetod (*turnRight(int n)*) võtavad argumendiks täisarvu (*int n*), mitu millisekundit tuleb oodata enne järgmise käsu täitmist. Meetodi välja kutsumisel antakse meetodile kaasa argument kujul *forward(2000)*. Meetodi sees asendatakse *delay(n)* sees olev *n* argumendi väärtusega (2000).

```
#include <Servo.h>  // Arduino IDE-s olemas

Servo servoLeft;    // Loob servo objekti
Servo servoRight;   // Loob servo objekti

void setup() {
  servoLeft.attach(7);  // Ühendab servo objekti pesaga 7
  servoRight.attach(9); // Ühendab servo objekti pesaga 9
}

void loop() {
  // Tsükkel, mida läbitakse 4 korda
  for (int i = 0; i < 4; i++) {
    forward(2000);    // Käivitab forward() meetodi
    turnRight(400);  // Käivitab turnRight() meetodi
  }
  stopRobot();    // Käivitab stopRobot() meetodi
  delay(2000);    // Peatab programmi töö kaheks sekundiks
}
// Robot liigub edasi
void forward(int n) {
  servoLeft.write(180);
  servoRight.write(7);
  delay(n);
}
// Robot pöörab paremale
void turnRight(int n) {
  servoLeft.write(180);
  servoRight.write(180);
  delay(n);
}
// Robot peatub
void stopRobot() {
  servoLeft.write(93);
  servoRight.write(93);
}
```

Joonis 100. Roboti ruudus sõitmise programm

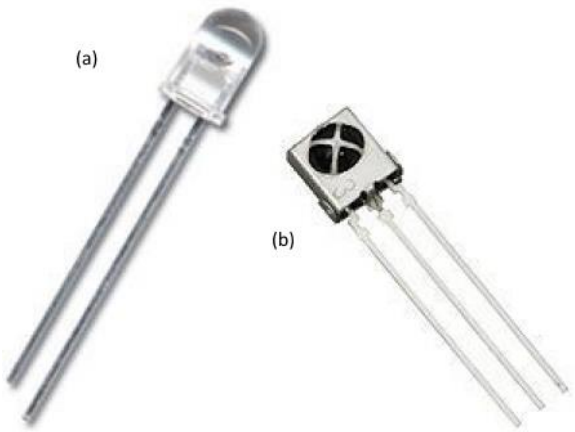
Pärast kolme meetodi loomist saab need meetodis *loop()* välja kutsuda. Ruudus hakkab robot sõitma nii, et meetodid *forward(int n)*, mis paneb roboti edasi liikuma, ja meetod *turnRight(int n)* käivitatakse 4 korda järjest *for*-tsükli abil. Seejärel peatatakse robot meetodi *stopRobot()* abil programmi töö kaheks sekundiks ning siis käivitatakse meetod *loop()* uuesti.

Robot suudab nüüd liikuda. Järgmiseks võiks saada roboti kaugjuhtida. Selleks kasutatakse infrapuna sensoreid, mida tutvustatakse järgmises punktis.

4.3 Infrapuna sensorite ühendamine Arduinoga

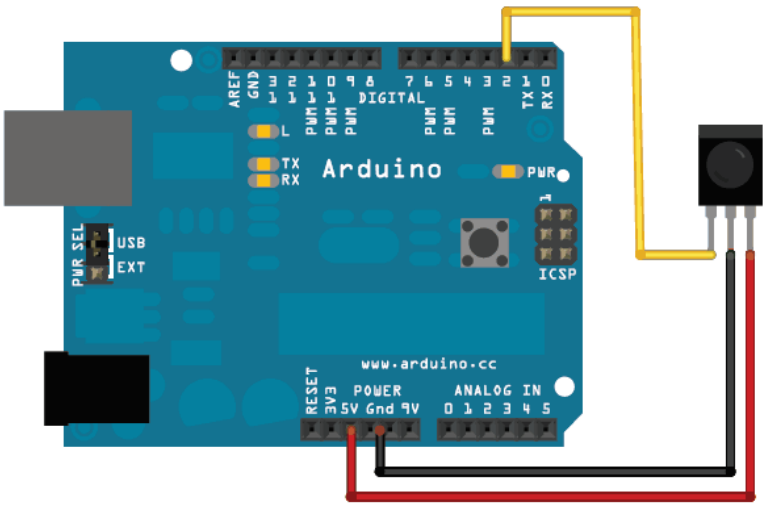
Infrapunavalgus ehk infrapunakiirgus on valgus, mis pole inimese silmale nähtav. Infrapunakiirgust leidub ka looduses. Kõik, mis kiirgab soojust, kiirgab ka infrapunakiirgust. Selleks, et looduslik infrapunakiirgus ei segaks infrapunakiirgust kasutavaid seadmeid, on saatja signaal moduleeritud teatud kandevsagedusele. Tavaliselt kasutatakse sagedust 38kHz. See tähendab, et signaal lülitatakse sisse ja välja 38 000 korda sekundis.

Infrapuna puldi sees on infrapuna saatja (vt joonis 101 (a)) ning robotil oleva Arduino peal on infrapuna vastuvõtja (vt joonis 101 (b)).



Joonis 101. Arduino infrapuna saatja (a) ja -vastuvõtja (b)

Infrapuna vastuvõtja ühendamiseks tuleb ühendada kolm juhet: signaal (kollane), baas (must) ja toide (punane) (vt joonis 102). Juhtmete järjekord sõltub sensorist.



Joonis 102. Infrapuna vastuvõtja ühendamine Arduinoga

4.4 Infrapuna sensorite abil roboti juhtimine

Roboti juhtimiseks tuleb kirjutada infrapuna sensorile programm. Programmi kirjutamiseks tuleb kasutada teeki, mida Arduino IDE-s olemas ei ole, et infrapuna sensoreid kasutada. Selleks tuleb lehelt *GitHub* alla laadida uus teek *Arduino-IRremote*. Uue teegi kasutamiseks tuleb eelnevalt kustutada või tõsta teise kausta Arduino IDE-s olev RobotIRRemote teek, kuna vastaval juhul tekivad teekide vahel konfliktid. RobotIRRemote teek asub *Windowsis* kaustas *C:/Program Files (x86)/Arduino/libraries*. Uue teegi installimisega tulevad kaasa ka näiteprogrammid, mis on kättesaadavad Arduno IDE-s valides ülemisest menüüst *File*, seejärel *Examples* ning tekkinud menüüst *IRremote*. Uute teekide kasutamisest on rohkem kirjutatud punktis 1.5.

Infrapuna vastuvõtja programmi kirjutamiseks võetakse aluseks näiteprogramm *IRrecvDemo* (vt joonis 34). Tekitatakse *irrecv* objekt, mis on võimeline võtma vastu infrapuna signaali. Kui signaal leitakse, tegutsetakse edasi vastavalt signaali väärtusele. Selleks kasutatakse tingimuslauset *switch*, mis saab argumendiks vastuvõetud signaali väärtuse (*results.value*). Tingimuslause *switch* sees on mitu *case*-lauset, mis määravad ära, mida tuleb teha teatud väärtuse korral. Näiteks *case 0xFD8877* blokki liigutakse siis, kui saadakse väärtus 0xFD887. Joonisel 28 olev programm on kirjutatud nii, et puldil (vt joonis 33) olevate noolenuppude vajutamisel liigub robot vastavasse suunda. Selleks kutsutakse iga noolenupu vajutuse korral välja teatud meetod, mis on kirjeldatud pärast meetodit *loop()* (vt joonis 34).

Joonis 103. Arduino juhtpult

Kui signaali pole kirjeldatud *case*-lausega ning robot on arvutiga ühendatud, väljastatakse signaali väärtus jadapordi monitorile (*default*-lause). Pärast signaali otsimist oodatakse enne uue signaali otsimist 100 millisekundit (*delay(100)*).

```
/*
 * IRremote: IRrecvDemo - demonstrates receiving IR codes with IRrecv
 * An IR detector/demodulator must be connected to the input RECV_PIN.
 * Version 0.1 July, 2009
 * Copyright 2009 Ken Shirriff
 * http://arcfn.com
 */

#include <IRremote.h>
#include <Servo.h>  // Arduino IDE-s olemas

Servo servoLeft;    // Loob servo objekti
Servo servoRight;   // Loob servo objekti

int RECV_PIN = 11; // Määrab ära sensori pesa
IRrecv irrecv(RECV_PIN); // Teeb irrecv objekti
decode_results results; // Teeb results objekti

void setup() {
  servoLeft.attach(7);  // Ühendab servo objekti pesaga 7
  servoRight.attach(9); // Ühendab servo objekti pesaga 9
  Serial.begin(9600); // Alustab jadapordi monitori
  irrecv.enableIRIn(); // Alustab infrapuna vastuvõtja
}

void loop() {
  if (irrecv.decode(&results)) {  // Kontrollib, kas saab signaali
    // Valib tegevuse vastavalt results.value väärtusele
    switch(results.value) {
      case 0xFD8877: // Kui väärtus on 0xFD8877
        forward(500); // Robot läheb edasi
        stopRobot(); // Robot peatatakse
        break; // Lõpetatakse case plokk
      case 0xFD6897:
        turnRight(400); // Robot pöörab paremale 90 kraadi
```

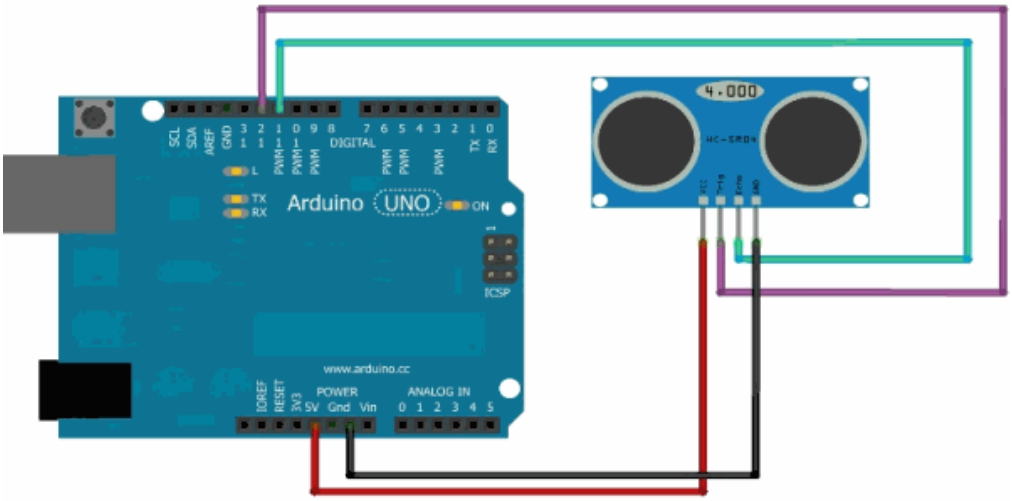
```
        stopRobot();
        break;
      case 0xFD28D7:
        turnLeft(400); // Robot pöörab vasakule 90 kraadi
        stopRobot();
        break;
      case 0xFD9867:
        reverse(500); // Robot läheb tagasi
        stopRobot();
        break;
      default: // Kui tuleb teine signaal
        // Väärtus väljastatakse jadapordi monitorile
        Serial.println(results.value, HEX);
    }
    irrecv.resume(); // Võtab vastu järgmise väärtuse
  }
  delay(100);   // Peatab programmi töö 100 millisekundiks
}

// Võtab argumendiks täisarvu ning
// sõidab nii kaua edasi (millisekundites)
void forward(int n) {
  servoLeft.write(180);
  servoRight.write(7);
  delay(n);
}

// Võtab argumendiks täisarvu ning
// sõidab nii kaua tagasi (millisekundites)
void reverse(int n) {
  servoLeft.write(7);
  servoRight.write(180);
  delay(n);
}

// Võtab argumendiks täisarvu ning
// pöörab nii kaua paremale (millisekundites)
void turnRight(int n) {
  servoLeft.write(180);
  servoRight.write(180);
  delay(n);
}

// Võtab argumendiks täisarvu ning
// pöörab nii kaua vasakule (millisekundites)
void turnLeft(int n) {
  servoLeft.write(0);
  servoRight.write(0);
  delay(n);
}

// Peatab roboti
void stopRobot() {
  servoLeft.write(93);
  servoRight.write(93);
}
```

Praegu on programm tehtud nii, et robot reageerib ainult puldil olevate noolenuppude signaalile ja siis liigub neile vastavalt. Algoritmi kirjeldatakse rohkem punktis 4.5. Selleks, et robot liikudes seintele vastu ei sõidaks, on vaja ühendada robotile ultraheli sensor. Sensori ühendamise ja kasutamisega tutvutakse järgmises punktis.

4.5 Ultraheli sensor

Ultraheli sensor mõõdab anduri ees oleva vahemaa takistuseni. Liiga väikse vahemaa korral robot enam edasi ei liigu, et vältida kokkupõrget takistusega. Ultraheli sensor koosneb saatjast ja vastuvõtjast. Saatja saadab heliimpulsi ja vastuvõtja võtab selle vastu. Aeg, mis kulub saatmisest vastuvõtmiseni teisendatakse sentimeetriteks.

Ultraheli sensori kasutamiseks tuleb Arduinoga ühendada 4 juhet: toide (punane), baas (must) ja kaks signaali (Trig (lilla) ja Echo (roheline)) (vt joonis 104).



Joonis 104. Ultraheli sensori ühendamine Arduinoga

Ultraheli sensori kasutamiseks tuleb kirjutada programm (vt joonis 105). Programmis tuleb esmalt enne *setup()* meetodit määrata ära õiged *Echo* ja *Trig* pesad ning *EchoTime* pikkus, milleks on 0. *EchoPin* väärtus on 12 ning *TrigPin* väärtus 13. Meetodis *setup()* initsialiseeritakse *EchoPin* ja *TrigPin* ning alustatakse jadapordi monitor, et saaks sinna väärtusi väljastada. Meetodis *loop()* saadab *TrigPin* signaali (*digitalWrite(TrigPin, HIGH)*), ootab 10 mikrosekundit ning siis lõpetab signaali saatmise (*digitalWrite(TrigPin, LOW)*). Seejärel mõõdetakse meetodi *pulse()* abil aeg, mis kulub, et *EchoPin* signaali vastu võtaks (*EchoTime = pulseIn(EchoPin, HIGH)*). Seejärel arvutatakse kulunud aeg ümber sentimeetriteks (*(EchoTime/2)/29*) ning väljastatakse väärtus jadapordi monitorile.

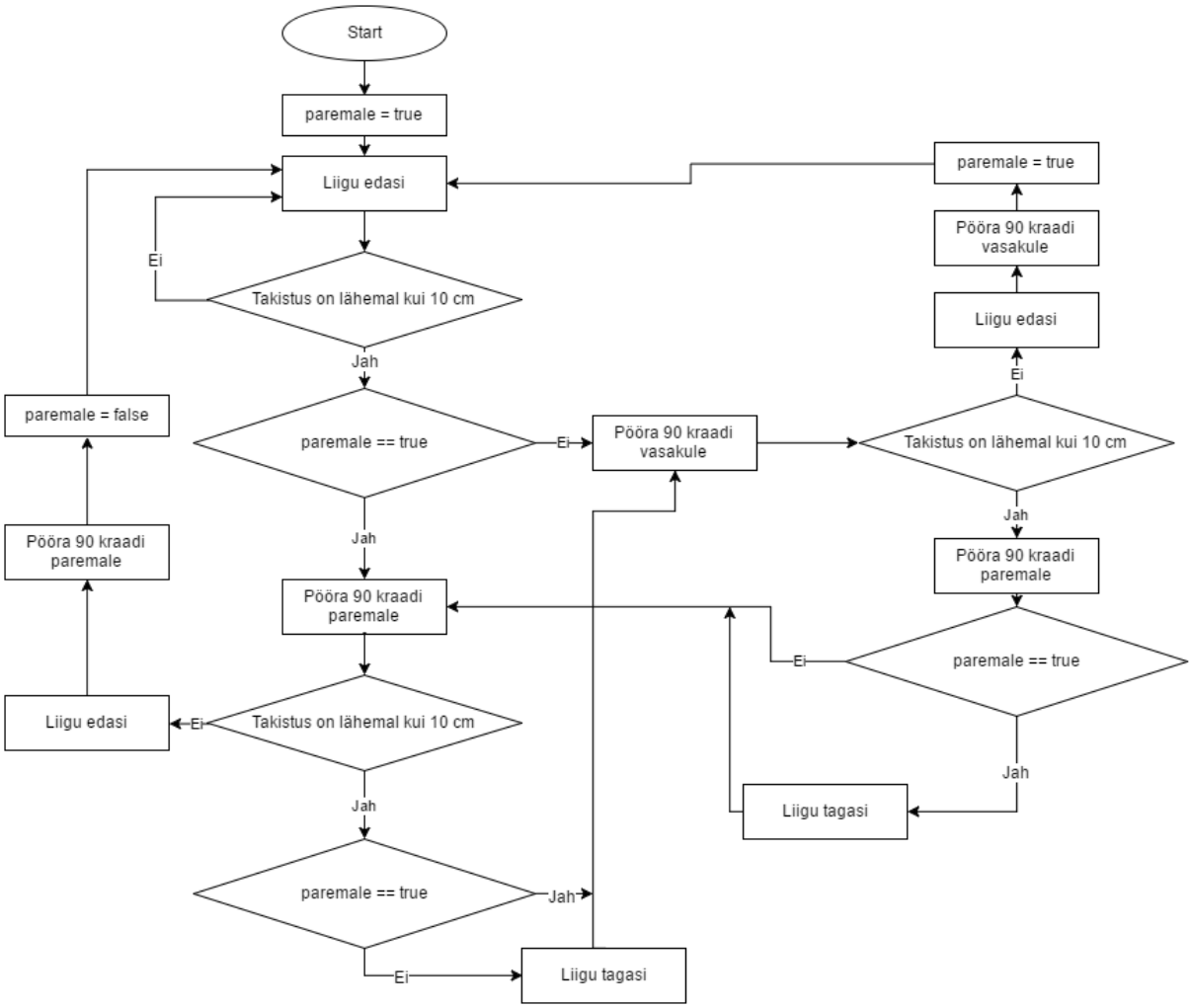
```
// Määrab õigete pesad
int EchoPin = 12;
int TrigPin = 13;
long EchoTime = 0;

void setup() {
  Serial.begin(9600); // Alustab jadapordi monitori
  // Initsialiseerib Echo ja Trig pesad
  pinMode(EchoPin, INPUT);
  pinMode(TrigPin, OUTPUT);
}

void loop() {
  digitalWrite(TrigPin, HIGH); // Trig saadab signaali
  delayMicroseconds(10);
  digitalWrite(TrigPin, LOW); // Lõpetab signaali saatmise

  EchoTime = pulseIn(EchoPin, HIGH);  // Echo võtab signaali vastu
  // Kulunud aeg teisendatakse sentimeetriteks
  int dist = (EchoTime/2)/29;
  Serial.print(dist, DEC);  // Väljastab jadapordi monitorile
  Serial.println("cm");
  // Ootab 1 sekundi
  delay(1000);
}
```

Joonis 105. Ultraheli sensori kasutamise programm

Lõplikus programmis kasutatakse mõõdetud vahemaad eesolevast takistusest, et mitte vastu seina sõita. Seda saab kontrollitakse *if*-tingimuslausega. Vahel annab ultraheli sensor tulemuseks 0, kuigi takistus on tegelikult kaugemal. Programmi kirjutades tuleb sellega arvestada kirjutades *if*-tingimuslausesse, et kui tulemus on 0, tuleb seda eirata.

4.6 Laadimispunkti liikumise algoritm

Robot liigub toas, kus pole põrandal alla 5 sentimeetrilaiuseid objekte, näiteks juhtmed ja toolijalad. Vabas olekus liigub robot toas edasi-tagasi. Algoritm on kirjeldatud joonisel 106 oleva vooskeemiga. Üks kord pöörab paremale ning järgmine kord vasakule. Pööre tehakse siis, kui edasi sõites tuleb takistus ette. Kui pööret tehes tuleb sein ette, pööratakse teisele poole. Võib juhtuda, et mõlemale poole pöörates on sein liiga lähedal. Vältimaks olukorda, kus robot jääbki pöörama, kontrollitakse, kummale poole pidi robot alguses pöörama. Kui mõlemal pool on takistus liiga lähedal, liigub robot natuke tagasi ning proovib uuesti pöörata.



Joonis 106. Roboti liikumise algoritm

Joonisel 106 olev skeem algab joonise üleval olevast ovaalist "Start". Edasi liigub programm noolte suunas. Ristkülikutes on tegevused ning rombides tingimuslaused, kust edasi liigutakse vastavalt sellele, kas tingimuslause oli tõene või väär.

4.7 Roboti ehitamine

Antud punktis antakse ülevaade, kuidas on ühendatud robot erinevad osad. Robot koosneb kahe servomootoriga alusest, kolmandast rattast roboti esiosas, Arduino arendus- ja laiendusplaadist, maketeerimislauast, LCD-ekraanist ning sensoritest. Robotil on infrapuna- ning ultraheliandur. Lisaks on roboti ette kinnitatud akupank.

Arduino arendusplaadile on ühendatud prototüüpimise laiendusplaat koos maketeerimislauaga, et saaks rohkem juhtmeid ühendada. Näiteks on arendusplaadil ainult üks 5V pesa. Maketeerimislaua abil saab ühest pesast signaali edasi kanda mitmesse pesasse.

Arduino arendusplaat on kinnitatud kahepoolse kleeplindiga alumiiniumist alusele. Alus on eelnevalt kaetud isoleerteibiga. Alusel on kaks ratast, mida liigutavad servomootorid. Rattaid on robotil vaja, et liikuda laadimispunkti. Alusele on kinnitatud ka kolmas ratas, et robot oleks tasakaalus.

LCD-tähtekraani ühendamiseks on kasutatud I2C-kontrollerit. Ilma kontrollerita tuleks ühendada rohkem juhtmeid ning väikesel maketeerimislaual ei ole selleks piisavalt vabu pesasid. Ekraan on kinnitatud roboti tagumisse osasse. LCD-tähtekraani peale väljastatakse mõõdetud aku pinge.

Robotil on kaks andurit. Infrapuna sensor on vajalik, et võtta vastu infrapuna signaali. Ringi sõites võib robot sõita vastu seina. Selle vältimiseks on robotile ühendatud ultraheliandur, mis mõõdab, kui kaugel on roboti ees olev objekt. Ultraheli sensori abil muudab robot liikumise suunda, kui takistus tuleb ette. Sensor ei mõõda alla 5-sentimeerise laiusega objekte, näiteks toolijalgu või juhtmeid. Seetõttu tuleb sellised esemed roboti liikumisalast enne eemaldada. Roboti ette on kinnitatud akupank.

Joonisel 107 on näha lõplik robot koos anduritega.

Joonis 107. Näide valminud robotist

4.8 2.4“ puutetundliku ekraani kasutamine

Puutetundliku LCD ekraani kasutamine Arduinoga on arvatavasti üks keerulisemaid tegevusi mida te Arduinoga kunagi teete. Olukorra teeb eriti segaseks asjaolu, et Arduino LCD laiendusplaadid pole standardiseeritud ja iga tootja (loe: veebileht/vahendaja) müüb veidi erinevat ekraani, mille jaoks on vaja teistsugust *library* ja seadistust kui see mida siinkohal kirjeldatakse. Antud juhend käsitleb Banggood.com lehel perioodil Juuni 2016 kuni kirjutamise hetkeni (oktoober 2016) müüdavaid ekraane http://www.banggood.com/2_4-Inch-TFT-LCD-Shield-Touch-Board-Display-Module-For-Arduino-UNO-p-940735.html

Esimese asjana enne LCD kasutamist peaksime alla laadima library ehk tarkvarakogumi, mis sisaldab näidiskoodi ehk riistvara draivereid LCD ekraani kasutamsieks. Selleks otsige GOOGLE lehelt „**SPFD5408**“ märksõna ja saadud vastustest valida GitHub leht, kust saate vastava tarkvara alla laadida. Selleks minge lehele https://github.com/JoaoLopesF/SPFD5408 ja vajutage keskel paremas servas rohelisele nupule kirjaga „**Clone or Download**“ ja sealt valida „**Download ZIP**“. Nüüd peaks antud failid lahti pakkima ning kopeerima kausta nimega „**SPFD5408-master**“ Arduino täiendava tarkvara kausta „**C:\Program Files (x86)\Arduino\libraries**“. Nüüd peaksite tegema **Arduino IDE**-le restardi selleks sulgege Kõik Arduino IDE aknad ja avage Arduino IDE uuesti.

Nüüd saate valida
File → Examples → SPFD5408-master → spfd5408_graphicstest

Ühendage LCD laiendusplaat Arduinoga UNO-ga ja laadige SPFD5408_graphicstest näidiskoodi Arduinosse. Ekraanil peaks kuvatama erinevaid jooni ja teksti. Siit edasi saate hakata näidiskoodi modifitseerima ja muutma vastavalt oma soovidele ja eesmärkidele.

Puutetundlikkuse töölesaamsieks tuleb aa täiendavalt vaeva näha ja näiteks lihtsa joonistusprogrammi tegemiseks tuleks valida
File → Examples → SPFD5408-master → spfd5408_tftpaint

Avanenud programmis tuleks teha järgmised muudatused:
Otsida üles read mis seadistavad YP, XM, YM ja XP parameetrid (ehk kuhu jala külge on ühendatud puutetundlikkus) ja muuta neid vastavalt sinu kasutada olevale ekraanile.
Antud juhendis kasutada oleva Banggood.com 2,4 LCD puhul siis read

```
#define YP A1  // must be an analog pin, use "An" notation!
#define XM A2  // must be an analog pin, use "An" notation!
#define YM 7   // can be a digital pin
#define XP 6   // can be a digital pin
```

Asendada järgmiste ridadega

```
#define YP A2  // must be an analog pin, use "An" notation!
#define XM A1  // must be an analog pin, use "An" notation!
#define YM 6   // can be a digital pin
#define XP 5   // can be a digital pin
```

Täiendavalt veenduda et kood sisaldaks ridu, mis määrab millist graafika *library* kasutame ja LCD ekraani tüübi koodi, milleks on **0x9341**

```
#include <SPFD5408_Adafruit_GFX.h>    // Core graphics library
#include <SPFD5408_Adafruit_TFTLCD.h> // Hardware-specific library
#include <SPFD5408_TouchScreen.h>
```

ning

```
  tft.begin(0x9341); // SDFP5408
  tft.setRotation(0);
```

Näiteks saab sellisel juhul realiseerida Arduino peale kalkulaatori https://www.youtube.com/watch?v=0gjTIMJKs4c , mis on hea ja õpetlik ülesanne ka näiteks analüüsimiseks kuidas ekraani ja selle erinevaid funktsioone kasutada.

Täiendavalt võib osutuda vajalikuks redigeerida faili „**C:\Program Files (x86)\Arduino\libraries\SPFD5408-master\SPFD5408_TouchScreen.cpp**“ täpsemalt sealt rida 163 pöörata y ja x telge ning muuta neid vastassuunaliseks selleks saate vahetada x ja y omavahel või asenda „y“ väärtusega „1023-y“ ehk tulem oleks midagi sellist:

```
  return TSPoint(1023-y, x, z); // for TFT paint
```

4.9 Roboti ehitamiseks kasutatud asjade nimekiri ning soetuskoht:

Nagu eelnevalt kirjeldatud siis kodukasutajatel soovitatakse alustada Arduino stardikomplekti soetamisega ja siis vajalikud lisad roboti ehitamsieks ükshaaval juurde osta. Koolidele soovitame järgmist nimekirja osadest, mis on 10+ koguse jurues soodsam ja praktilisem lahendus, sest soetatakse ainutl osad mida on vaja konkreetse roboti ehitamsieks.

Idead Studio:

- Iteaduino Uno – 6,4 USD
- ITEAD 1602 LCD Keypad – 5 USD
- Ultrasonic Module HC-SR04 Range Finder – 2 USD
- Joystick breakout board – 2 USD

Robotshop EU:

- 2x Micro servo – 5 EUR

Ebay:

- Micro Roller Lever Arm Limit Switch – 0,2 USD
- Photo Light Sensitive Resistor Photoresistor Optoresistor 5mm GL5539 – 0,1 USD
- 5mm 4pin RGB Tri-Color Diffused Common Anode LED Lamp Emitting Diode – 0,1 USD
- 5mm difused LED-s: GREEN; RED; YELLOW; BLUE, WHITE – 0,5 USD
- Mini Breadboard Solderless Protoboard PCB Test Board 400 Contacts Tie Points – 2 USD
- New Infrared IR Wireless Remote Control Module Kits for Arduino – 2 USD
- Details about Microphone Sensor AVR PIC High Sensitivity Sound Detection Module For Arduino – 1 USD
- DHT11 Temperature and Relative Humidity Sensor Module for arduino – 1 USD
- Jumper wire Male-Male 10tk 20cm – 0,3 USD
- Jumper wire Male-Female 10tk 20cm – 0,3 USD
- Jumper wire Female-Female 10tk 20cm – 0,3 USD
- 3x Breadboard comptaible 2pin micro switch – 0,5 EUR
- 5v comptaible micro DC motor – 1 USD
- Väljatransistor mootori jaoks – 0,2
- Digispark Kickstarter ATTINY85 Arduino General Micro USB Development Board – 2 USD
- 2.4 Inch TFT LCD Shield Touch Board Display Module For Arduino UNO – 7 USD

ORDI:

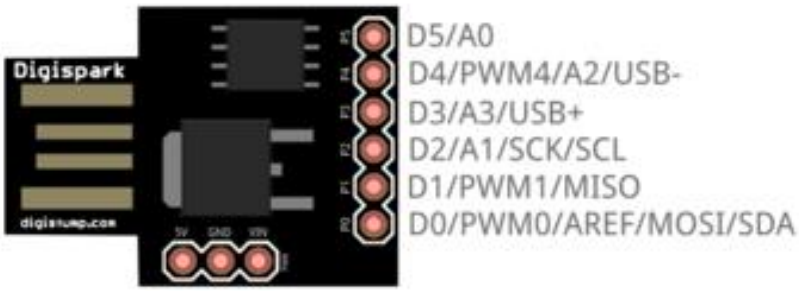
- 5V Akupank PB-100 2600mAh – 8 EUR (ordi)
- USB kaabel Type A (plug)- mini type B (plug) 1m või pikem – 1 EUR

Eeldatav hind ~50 EUR

DIGISPARK

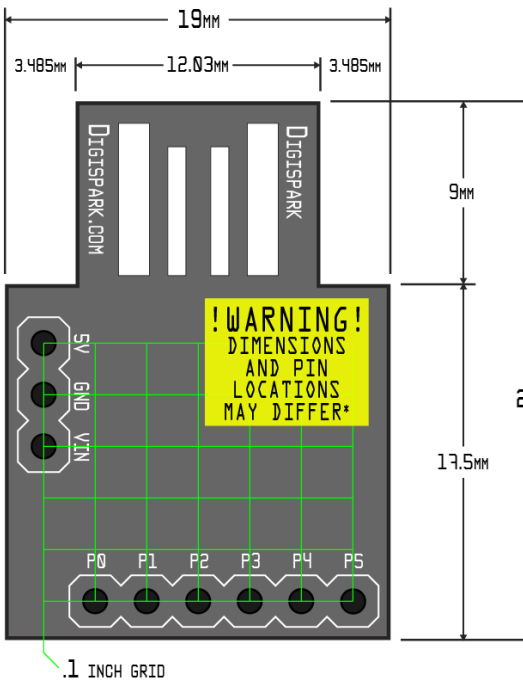
Väike ja odav kuid võimekas Arduinoga ühilduv arendusplaat

- **Programmeerimiskeskkond:** Arduino IDE (C-keel), vajab eraldi seadistust
- **Vaba tarkvara ja vaba riistvara** (kogu tarkvara, skeemid jne on avalikud)
- **Protsessor:** Attiny85 (16,5 MHz, 8 bit, 8 KB flash, low power)
- **Sisend-väljund :** 6 digitaalset I/O viiku
  4 Analoog sisendit
- **Tööpinge:** 5V
- **Väline toide:** 7-35V
- **Mõõtmed:** 1,8 x 2,6 cm



D: Digital Read/Write
A: Analog Read (ADC)
PWM: Analog Write
MOSI/MISO/SCK: SPI
SDA/SCL: I2C
USB+/-: USB Interface
AREF: Analog Reference

mena (hiir,
test ja riistvarast



*DIMENSIONS AND PIN LOCATIONS BASED ON CURRENT VERSION FOUND AT
HTTP://WWW.KICKSTARTER.COM/PROJECTS/DIGISTUMP/DIGISPARK

DigiSpark kasutamine:

DigiSpark on Arduino derivaat ja selle kasutamine on veidi erinev klassikalise Arduino-ga võrreldes. Esimese asjana nagu ka Arduinoga tuleb arvutisse paigaldada Arduino IDE, mille saab alla laadida www.arduino.cc lehelt. Erinevalt tavalisest Arduino/Genuino-st tuleb lisaks paigaldamisele ka enne DigiSpark kasutamist Arduino IDE-le lisada DigiSparki tugi. Vastava juhendi leiab lehelt. http://digistump.com/wiki/digispark/tutorials/connecting

Pärast edukat DigiSpark installeerimist saate seda testida järgmise koodiga. Pööra tähelepanu asjaolule et erinevalt tavalisest Arduinost tuleb DigiSPark ühendada alles pärast programm nupu vajutamist 60s jooksul. Antud nõue tuleneb asjaolust et seadme ühendamisel arvutiga (toitega) on seade programmeerimise resiimis ja pärast 5s möödumist tuntakse see ära kui USB seade.

```
// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(0, OUTPUT); //LED on Model B
  pinMode(1, OUTPUT); //LED on Model A
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(0, HIGH);   // turn the LED on (HIGH is the voltage level)
  digitalWrite(1, HIGH);
  delay(1000);               // wait for a second
  digitalWrite(0, LOW);    // turn the LED off by making the voltage LOW
  digitalWrite(1, LOW);
  delay(1000);               // wait for a second
}
```

Väga lõbus harjutus on seadistada DigiSpark käituma klaviatuuri või hiirena. Vastavalt
File → Examples → DigisparkKeyboard → Keyboard
File → Examples → DigisparkMouse → Mouse

Kui nüüd seade ühendada arvuti taha siis hakkab 5 sekundi pärast kas hiir liikuma või tekib ekraanile tekst nagu keegi oleks klaviatuuril klahve vajutanud.