TARTU ÜLIKOOL
Loodus- ja täppisteaduste valdkond
Füüsika instituut

Rimmo Rõõm

Digitaalsed mõõtmised füüsikaõppes platvormidel LEGO Mindstorm EV3 ja Arduino

Bakalaureusetöö (12 EAP)

Füüsika, keemia ja materjaliteaduse õppekava, füüsika eriala

Juhendaja: Kaido Reivelt, PhD

Tartu 2021

Digitaalsed mõõtmised füüsikaõppes platvormidel LEGO Mindstorm EV3 ja Arduino

Füüsika gümnaasiumi kursustes tutvutakse mitmete füüsikaliste mudelitega, mida ei ole võimalik ilma digivahenditeta eksperimentaalselt kontrollida. Antud bakalaureusetöös uuritakse võimalusi LEGO Mindstorms EV3 mikrokontrolleri ja Arduino mikrokontrolleri kasutamiseks füüsika praktikumides. LEGO Mindstorms EV3 sobivuse hindamiseks viidi läbi erinevaid mõõtesagedusi nõudvad testeksperimendid. Mikrokontrollerit Arduino Nano testiti vabalangemise aega mõõtvas katses. Töö tulemusena selgub, et LEGO Mindstorms EV3 ei ole üldiselt sobilik digitaalseteks mõõtmisteks füüsika eksperimentides. Samas Arduino Nanol on olemas kõik eeldused, et sellele üles ehitada odav, samas piisavalt täpne digitaalsete mõõtmiste süsteem.

Märksõnad: LEGO Mindstorms EV3, ajaline lahutus, Arduino, aja mõõtmine, füüsika praktikum

CERCS kood: T125 Automatiseerimine, robootika, juhtimistehnika

Digital measurements in physics learning on LEGO Mindstorm EV3 and Arduino platforms

In physics praxis we are faced with tasks and physical models that need digital tools to verify the experiments. In this bachelor's thesis, the suitability of using the LEGO Mindstorms EV3 microcontroller for this purpose is studied, as it is already widespread in Estonian schools. To assess suitability, the time resolution of the microcontroller is investigated, as many experiments require quite high sample speed. In the course of the work, a prototype based on Arduino Nano is also prepared and tested for physics praxis. The biggest advantage of a self-made device is flexibility in choosing the parameters and researching the results of the device. The results of these tests concludes that LEGO EV3 is not suitable for conducting digital observations in most of the cases, Arduino Nano has everything to build an inexpensive and efficient observation system.

Keywords: LEGO Mindstorms EV3, sample speed, Arduino, time measurements, physics praxis

CERCS code: T125 Automation, robotics, control engineering

Sisukord

Sissejuhatus

Üldhariduskoolide mehaanika kursuse dünaamikat käsitletavad teemad sisaldavad mitmeid füüsikalisi mudeleid, mida ei ole võimalik ilma digivahenditeta eksperimentaalselt kontrollida. Näiteks: impulsi jäävuse seadus põrgetes, viskekeha liikumine gravitatsiooniväljas, võnkumiste ja lainete uurimine ning teised [1] [2]. Üldjuhul kasutatakse koolides neis katsetes LabQuest [3] või Pasco Spark [4] andmelogijat, kui seda üldse tehakse.

Eelnimetatud teemade juures nõuavad paljud katsed üsna kõrgeid mõõtesagedusi. Näiteks üle laua serva veereva kuuli kukkumise aja mõõtmiseks peaksime suutma aega mõõta vähemalt sajandiksekundi täpsusega. Kiireid mõõtmisi on vaja teha ka põrgetes mõjuvate jõudude mõõtmiseks ja jäikade kehade võngete tuvastamiseks pärast kuuli põrget. Seega, igasugune füüsikakursustes kasutatav platvorm peab suutma mõõta füüsikalisi parameetreid sagedusega vähemalt 100 korda sekundis ning võimaldama neid andmeid ka lihtsalt töödelda.

Lisaks digitaalsetele mõõtevahenditele on mehaanika katseteks vaja ka kergesti käsitsetavaid mehaanilisi süsteeme. Üks võimalus on kasutada selleks ka Eestis laialt levinud robootika platvorme.

Robotitega ja teiste digitaalsete vahenditega on võimalik korraldada tõhusaid füüsikatunde, mis käsitlevad klassikalisi füüsika teemasid: arendada graafilise esitluse ja matemaatiliste võrrandite tundmaõppimist ning luua tingimusi muutujatega töötamiseks. Selline õppeviis aitab lahendada tüüpilisi probleeme, mis õpilastel tekivad mehaanika õppimisel. Õpilastel on katsete varal lihtsam aru saada füüsikalisest mudelist ja nähtuse olemusest. Katsete kasutamisel ei piirdu füüsika õppimine vaid matemaatiliste valemite pähe õppimisega [5]. On leitud, et robotitega tegelevad õpilased omandavad teadmisi lisaks robootikale ka arvutitehnikast, informaatikast, füüsikast ja inseneeriast. Samuti areneb nende keeleoskus [6].

Idee füüsikatundides roboteid kasutada tuleneb asjaolust, et füüsika katsetel on positiivne mõju füüsika õppimisele ning robotitega saab hõlpsasti füüsika katseid läbi viia. Vähetähtis ei ole ka fakt, et robootika on Eestis ennast tõestanud kui suurele hulgale õpilastele huvitav ja atraktiivne tegevus. Robotkatseid sisaldava efektiivse füüsika tunni ettevalmistamiseks on vaja saavutada mõistlik tasakaal, kus robootika ei varjuta ära füüsikat ja vastupidi. Lisaks peavad tunnis õppimise eesmärgid saama täidetud vähemalt sama edukalt, kui ilma robotiteta

läbi viidud füüsikatunnis. Selle saavutamiseks on konkreetsete robotkatsete kõrval samaväärselt tähtis sobivate õppemeetodite väljatöötamine.

Käesoleva töö eesmärk on uurida LEGO® Mindstorms EV3 [7] baaskomplekti ning Vernieri andurite ning Arduino mikrokontrollerite kasutamise sobivust põhikooli ja gümnaasiumi mehaanika teemade käsitlemiseks füüsika tundides.

Esimeses osas kirjeldan LEGO Mindstorms EV3 baaskomplekti mikrokontrollerit, Vernier andureid ja neid omavahel ühendavat NXT Sensor Adapterit ning annan ülevaate selle süsteemi tundmaõppimiseks tehtud eksperimentidest ja tulemustest. Teises osas kirjeldan Arduino mikrokontrolleri baasil valmistatud katseseadme prototüüpi, selle füüsikaliste parameetrite määramist ja sellega sooritatud eksperimente.

1. LEGO Mindstorms EV3

1.1 Ülevaade

1.1.1 Sissejuhatus

LEGO Mindstorms robotid on Eesti koolides laialt levinud ning komplekt annab paindlikkuse konstrueerida väga erinevaid mehaanilisi süsteeme. Füüsikaliste mõõtmiste tegemiseks on vaja põhjalikult tunda töös kasutatavaid mõõteriistu. LEGO Mindstorms EV3 baaskomplekti mikrokontrollerit, Vernier andureid ning neid omavahel ühendava NXT Sensor Adapteri [8] tehnilised spetsifikatsioonid on liialt üldsõnalised ning jätavad liiga palju tõlgendamisruumi. Seetõttu on nende parameetreid ja käitumist vaja eraldi uurida.

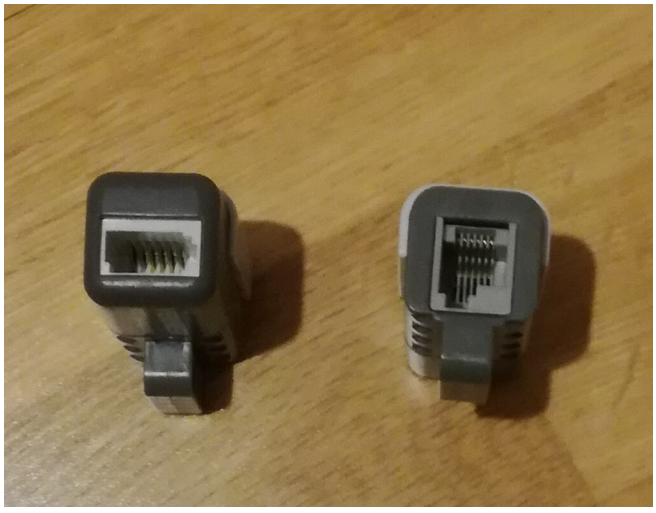
LEGO Mindstorms EV3 on edasiarendus varasematest versioonidest RCX ja NXT. Komplektis on programmeeritav mikrokontroller, üks keskmine ja kaks suurt servomootorit, erinevad andurid (värvi-, kaugus-, puute- ja güroandur) ning hulk LEGO® Technici klotse. EV3 on avatud platvormiga, nagu oli ka varasem versioon NXT. Avatud platvorm võimaldab ühendada mikrokontrolleriga ka paljude teiste anduritootjate andureid. Robotit on võimalik programmeerida nii graafiliselt kui ka kasutades programmeerimiskeelt Python [9].

Joonis 1. EV3 programmeeritav klots.

LEGO Mindstorms EV3 klotsis Joonisel 1 kasutatakse AM1808 mikrokontrollerit, mille sisendkanalid on 10-bitised ja väljundpinge anduritele on viis volti. Need parameetrid määravad anduritega mõõdetavate tulemuste täpsuseks 4,9 mV.

1.1.2 NXT Sensor Adapter

NXT Sensor Adapter Joonisel 2 on seade, mille abil on võimalik ühendada Vernieri analoogandurid LEGO Mindstorm EV3-ga. Adapter muudab RJ12 ühenduskaabli BTA ühenduskaabliks. Mõlemad kaablid on kuueviigulised. Viikudeks nimetatakse kiibi korpusest välja ulatuvaid metallkontakte, mis on ettenähtud kiibi elektriliseks ühendamiseks trükkskeemiga või kiibi pesaga. Vernieril on 40 andurit, mida on võimalik ühendada adapteri kaudu mikrokontrolleriga [8].



Joonis 2. NXT Sensor Adapter. Vasakul BTA pesa, paremal RJ12 pesa.

1.2 Testide tulemused

1.2.1 EV3 klotsi reaalse mõõtesageduse hindamine

Eelkatsed näitasid, et ehkki tarkvaraliselt on võimalik LEGO Mindstorms EV3 mõõtesageduseks määrata kuni 1000 mõõtmist sekundis, ei suutnud see süsteem reaalselt nii kiirete sündmuste aegridasid registreerida. Samas on füüsikaliste protsesside registreerimisel seadme reaalse ajalise lahutuse teadmine hädavajalik. Kuna ei leidnud ühestki materjalist millise sagedusega suudab EV3 klots lugemeid võtta, ja ka EV3 programmeerimise

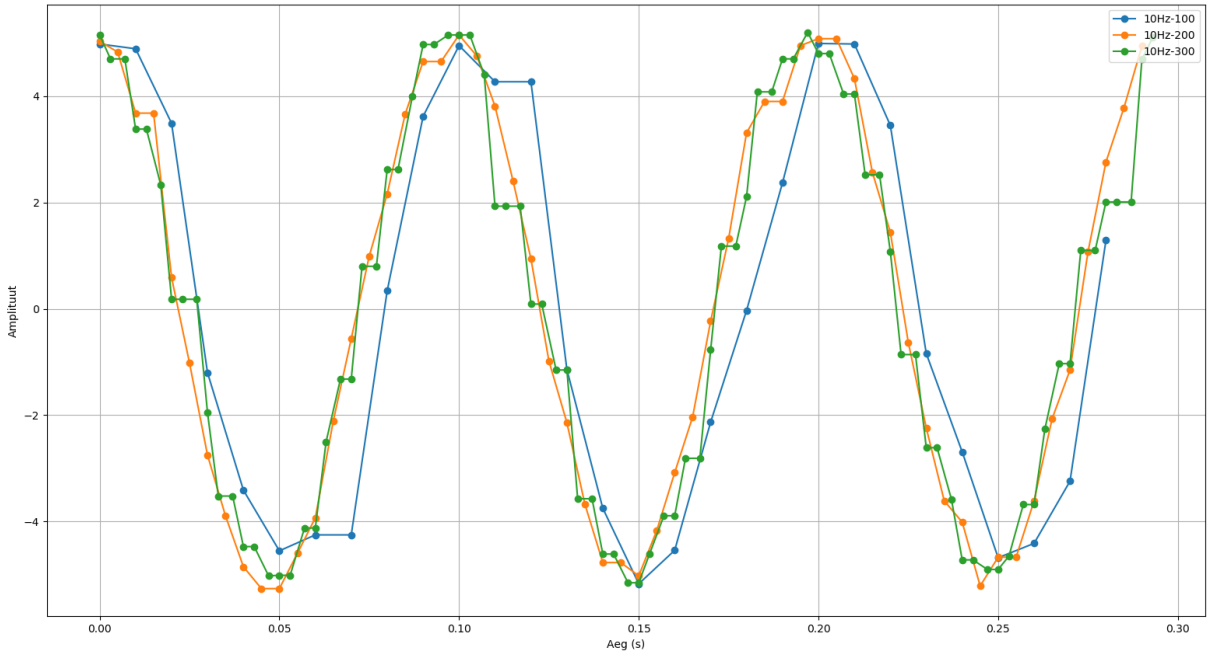
keskkonnas ei olnud piire sageduse määramiseks, siis konstrueerisin reaalse mõõtesageduse määramisks kontrollkatse.

Katses kasutasin signaali generaatorina Analog Discovery-t, Vernieri Differential Voltage Probe andurit koos NXT Sensor Adapteriga ning EV3 klotsi. EV3 klotsile kirjutasin programmi, mis loeb anduri väärtuseid erinevate sagedustega, iga tsükli pikkus oli viis sekundit. Programm alustas sajast lugemist sekundis ja iga tsükli järel tõstis lugemite arvu 100 võrra kuni 1000 lugemini sekundis. Saadud andmed salvestas programm ühtseks failiks. Programm koos selgitustega on leitav Lisas 1.

Testsignaaliks valisin sinusoidi amplituudiga 5 V ning kokku kasutasin 7 erinevat sagedust: 200 Hz, 100 Hz, 50 Hz, 10 Hz, 5 Hz ja 1 Hz. Iga test signaali jaoks käivitati programm eraldi. Nyquist–Shannoni teoreemi kohaselt peab olema ajaline lahutatavus rohkem kui kaks korda suurem kõrgeimast sagedusest signaalis. Valitud testsignaali sagedused jäävad selle kriteeriumi sisse.

Mõõtmisi alustasin 10 Hz sagedusega siinussignaalist. Katse tulemustest Joonisel 3 on korrektselt tuvastatav 10 Hz sinusoid. Jooniselt on välja jäetud kõrgema ajalise lahutusega mõõdetud signaal, et graafik oleks paremini loetav.

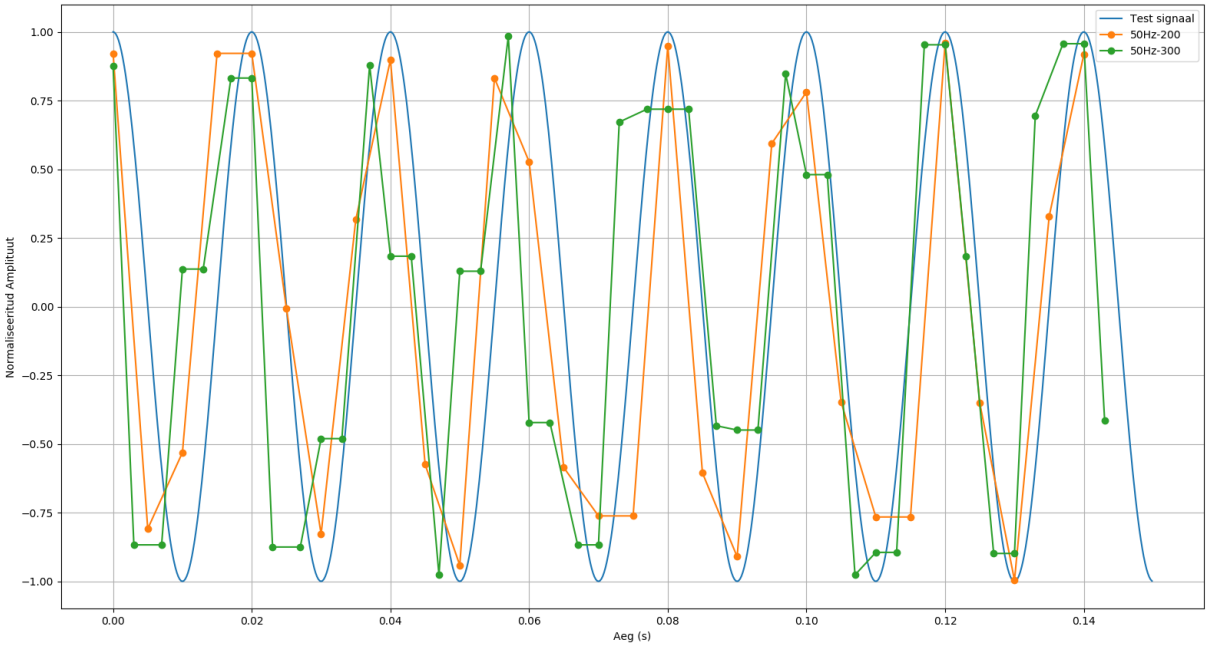
Joonisel torkab silma, et kõigi kolme madalaima ajalise lahutuse korral on signaal trepjas, mis viitab sellele, et kuigi EV3 kirjutas faili uue aja hetke olemas oleva väärtusega, pole siiski andurist uut väärtust jõutud veel pärida. Kõrgeima lugemiskiiruse juures see tendents süveneb. Kõikide lugemiskiirustega saadud andmete graafikud leiab Lisast 3.



Joonis 3. 10 Hz signaali mõõtmise tulemused. Sinine joon vastab 100, oranz 200 ja roheline 300 andmepunktile sekundis.

Tuginedes eelnevale oletusele leidsin, kui kaua aega kulub uue väärtuse tekkimiseks. Selleks kasutasin mõõtmiste andmeid, mis koguti kiirusega 1000 andmepunkti sekundis. Keskmine ajakulu andmepunkti muutumiseks oli 6.4 ms. Selle järgi arvutasin teoreetilise sageduse andmepunktide muutumise kohta, selleks sageduseks sain 156 Hz. Mis tähendab, et Nyquisti kriteeriumi järgi peaks olema EV3 klotsiga võimalik mõõta signaale, mis ei ületa 78 Hz piiri. See toetab ka oletust, miks on Joonisel 3 antud 200 Hz ja 300 Hz mõõtmissageduste jooned trepja kujuga. Kuid lahtiseks jääb veel, miks mõõtmissagedusega 100 Hz on samuti kohati trepja kujuga. Põhjuseks võib olla, et kohati kulub uue väärtuse võtmiseks siiski kauem aega kui 6.4 ms, aga põhjuseks võib ka olla, et tegelik andmete lugemiskiirus andurist on veel väiksem, kui minu leitud teoreetiline aeg. Kuna testimisel ei kasutanud madalamat lugemisagedust kui 100 Hz siis täpselt ei saa öelda, mis hetkest kaob trepjas kuju mõõdetud signaalist.

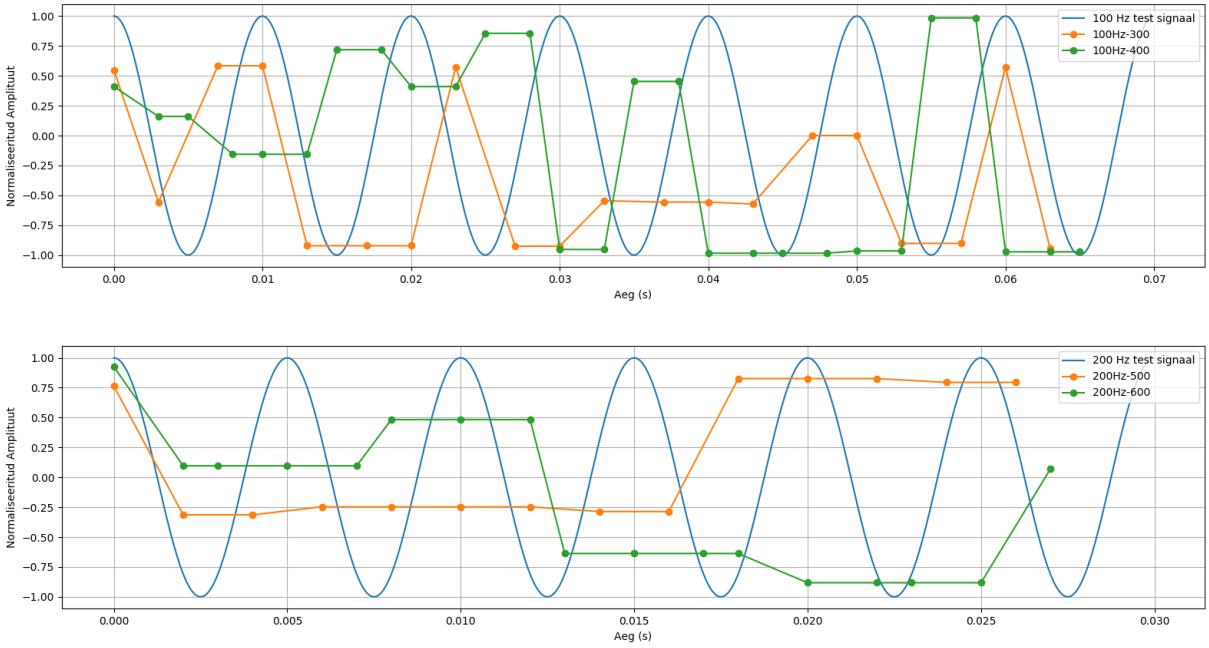
Järgmiseks vaatan 50 Hz sagedusega testsignaali, mis jääb veel Nyquist-i kriteeriumi sisse. Joonisel 4 on toodud testsignaali võrdlus kahe erineva ajalise lahutusega mõõdetud signaaliga. On näha, et 50 Hz signaal on veel äratuntav, kuid see on mõlemal lahutuse korral tugevalt moonutatud .



Joonis 4. 50 Hz signaali mõõtmise tulemused. Sinine joon kujutab testsignaali, oranz 200 ja roheline 300 andmepunktile sekundis.

Analüüsin ka 100 Hz ja 200 Hz testsignaali, et leida kinnitust teooriale, mille kohaselt ei õnnestu mul korrektset tulemust saada signaali sagedustega üle 78Hz piiri, kuna EV3 klotsil kulub ligikaudu 6.4 ms uue väärtuse saamiseks andurist. Joonisel 5 on kujutatud nii 100 Hz kui ka 200 Hz testsignaali koos mõõdetud tulemustega. On näha, et nendest signaalidest on juba võimatu öelda, millise kujuga oli algne signaal.

Kõik testsignaalid koos kõigi mõõtmissagedustega lähtudes Nyquist-i teoreemist on toodud Lisades 2 kuni 7.



Joonis 5. 100 Hz ja 200 Hz signaali mõõtmise tulemused koos testsignaali endaga.

1.3 EV3 katsetuste kokkuvõte

LEGO Mindstorms EV3 saab küll edukalt mõõtma signaale, mis sisaldavad kuni 10 Hz komponente, kui kasutada 200 Hz lugemi võtmise sagedust. Kuid tuleb arvesse võtta, et aeg-ajal esineb lugemi võtmisel viivitusi. Kõrgemate lugemi võtmise sageduste korral aga ei jõua andurist tulev info uueneda, seega ei ole mõistlik neid kasutada. Andurist tuleva info uuenemine võttis kõigi katsetuste põhjal aega 6.5 ms. 50 Hz testsignaal jääb küll nende kriteeriumite sisse, kuid ei anna väga head tulemust.

Testi tulemused ei rahulda aga ootusi, et kasutada LEGO Mindstorms EV3 komplekti mehaanika katsete läbiviimiseks füüsika praktikumides II ja III kooliastmes. Pärast sellele tulemusele jõudmist otsustasime alustada katseseeriat Arduino mikrokontrolleril põhinevate süsteemidega.

2. Digitaalsed mõõtmised Arduino mikrokontrolleriga

2.1 Ülevaade

2.1.1 Sissejuhatus

Arduino on avatud lähtekoodiga elektrooniline prototüüpimise platvorm, mille kõige suuremateks eelisteks on tarkvara ja riistvara lihtsus ning paindlikkus. Arduino on loodud eelkõige kunstnikke, disainereid ja hobielektroonikuid silmas pidades. Teiseks suureks eeliseks on Arduino enda ja selle andurite hind võrreldes olemasolevate seadmetega nagu Vernier LabQuest ja Pasco Spark. Viimaste andmelogijate hinnad jäävad 500$ piiresse ja seadmele lisanduvad veel andurid, mis tuleb eraldi juurde osta. Andurite hinnad jäävad 100$ piiresse. Tänu laiale levikule on kloonina Arduino hind umbes 3$ ja originaalina umbes 20$. Sarnased on ka andurite hinnaklassid. Olles laialt levinud, on Arduinol ka korralik tugi ja näidete süsteem olemas erinevate foorumite, õpetuste ja videote näol.

Käesoleva töö ühe alamülesande eesmärgiks oli luua mõõtmeseadme prototüüp, millega saaks läbi viia gümnaasiumi ja põhikooli füüsika praktikumide ülesandeid, mis aitaksid õpilastel paremini antud teemat mõista. Prototüübile said seatud järgmised nõuded.

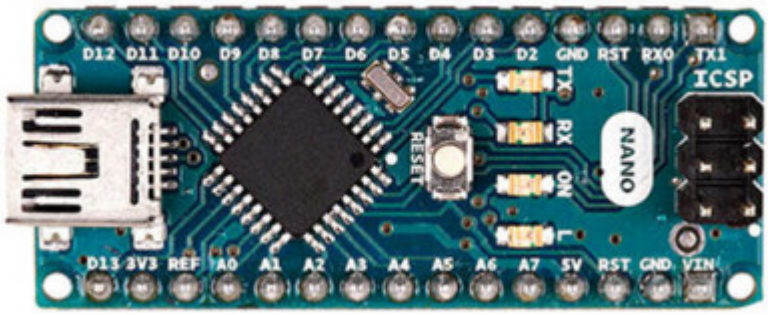
- Prototüübi komplekt peab olema lihtsasti soetatav ehk mõõtmeseade komplekti hind peab olema odavam kui antud hetkel turul olemasolevad komplektid.
- Seade peab olema lihtsasti kasutatav kõigil õpilastel.
- Võimeline registreerima kiireid sündmuseid.
- Seadmele peab olema võimalik külge panna erinevaid andureid vastavalt soovitud katsele.
- Seade peab olema võimeline kuvama mõõtetulemusi.

2.2 Prototüübi komponendid

Prototüübitud seade koosneb: arendusplaadist, mis teostab mõõtmiseid; ekraanist, millel kuvatakse kasutajale info; kahest potentsiomeetrist, millega määratakse nivood andurite jaoks; kahest nupust, mis on vajalikud nivoode ja mõõtmiste sooritamise jaoks; valgus- ja kiirendusandurist, mis on vajalikud katsete sooritamise jaoks. Prototüübi elektriskeemi leiab lisast 8.

2.2.1 Arduino Nano

Arduino Nano [10] on võrreldes teiste Arduino mudelitega väga kompaktne, aga sama võimekas kui teised Arduino mudelid. Mikrokontrolleriks on ATmega328P [11], tulenevalt sellest on Arduino Nano-l 14 digitaalset viiku ja 8 analoogviiku (vt joonis 6). Analoogviikudel mõõdetud väärtus esitatakse 10 bitilise arvuna ehk etalonpinge jagatakse 2^{10} osaks. Kiip on varustatud 16 MHz välise kristalliga, mis määrab kontrolleri töökiiruse.



Joonis 6. Arduino Nano [10].

2.2.2 ERM802-3 Seeria kuvar

ERM802-3 Seeria kuvar [12] on piisav, et mõõtmiste infot kuvada. Kuvar on kaherealine ja mõlemasse ritta mahub 8 tähemärki (joonis 7). See on piisav, et esialgu kuvada lühemaid teavitustekste, andurite väärtuseid ja mõõdetud tulemusi. Iga tähemärgi kuvamiseks on 8x5 piksli suurune ala. Suhtlus mikrokontrolleri ja kuvari vahel käib üle digitaalsete viikude. Programmeerides kasutasin LiquidCrystal teeki [13], mis tagab info saatmise kuvarile.

Joonis 7. ERM802-3 Seeria kuvar [10].

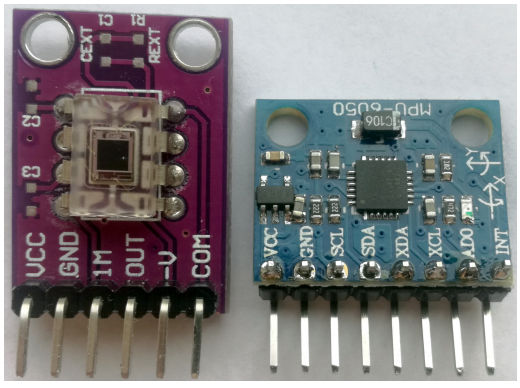
2.2.3 OPT101 Monoliitne fotodiood

OPT101 on monoliitne fotodiood, millele on sisseehitatud signaalivõimendi. Selline integreeritud kombinatsioon aitab vabaneda lekkevoolust ja vähendada tekkinud müra.

Mõõdetav väljundpinge kasvab lineaarselt valguse intensiivsusega [14], mille tõttu on seda andurit lihtne kasutada.

2.2.4 MPU6050 Kiirendus- ja güroskoopandur

MPU6050 Kiirendus- ja güroskoopandur koosneb kuuest väiksemast andurist, millest kolm mõõdavad kolmes teljes kiirendust ja teised kolm samades telgedes nurkkiirust [15]. Vastav väärtus esitatakse 16 bitise arvuna, mis annab parema kvantimise arvu kui Arduino analog-digitaal-konverter, ehk etalonpinge jagatakse 2^{16} osaks. Suhtlus anduri ja mikrokontrolleri vahel käib digitaalselt kasutades I2C [16] ehk kahesuunalist kahesoonelist jadasiini. Selline suhtlus seab süsteemile ka väikese piirangu: nimelt väheneb anduri ja mikrokontrolleri vaheline maksimaalne kaugus taktsageduse suurenedes. Madalaima taktsageduse juures, mis on 100 kHz, on usaldusväärne info edastamise kaugus varjestamata kaabliga 1 m . Programmeerides kasutasin I2Cdev ja MPU650 teeke [17].



Joonis 8. Vasakul - OPT101 Monoliitne fotodiood CJMCU-101 elektroonika plaadil, paremal - MPU6050 Kiirendus- ja güroskoopandur.

2.3 Prototüübi arendus

2.3.1 Palli veeremise katse kirjeldus

Testkatses veereb pall üle laua serva ja kukub põrandale, (joonis 9). Sellise katse ülesseadmine on lihtne, samas on vaja mõõtmisteks kasutada mitut andurit. Katses on võimalik tuvastada kolme ajahetke:

1) kui pall satub valgusanduri ette
2) kui pall lahkub valgusanduri eest

3) kui pall maandub põrandal.

Saadud tulemuste põhjal on võimalik leida näiteks palli horisontaalset kiirust ning selle järgi arvutada palli kukkumise kaugust laua servast. Samuti saab leida palli vabalangemise aega ja selle järgi hinnata raskuskiireduse väärtust. Joonisel 9 on näha, kuidas põrandale on asetatud kerge plaat, mille külge on õhukese kahepoolse teibiga kinnitatud kiirendusandur. Plaat oli vajalik, et anduril oleks võimalikult hea kontakt pinnaga, et tuvastada palli põrkel pinnaga materjali pidi levivat ristilainet. Kuna plaadi materjal oli pappkartong, siis oli plaat ka hea isolaator ruumi päris põrandal levivast mürast, mis võis potensiaalselt tekkida põrandal kõndides või mõne eseme mahakukkumisel. Kiirendusanduri juhe oli kinnitatud teibiga põranda külge, et vältida tahtmatul juhtme liigutamisel mõju anduri tööle. Fotodiood oli kruvitud otse puuklotsi külge, mis koos teise klotsiga hoidis palli veeremise renni alumist otsa. Fotodiood sai paigutatud palli tsentri kõrgusele, et fikseerida võimalikult täpselt ajahetked, mil pall veereb anduri ette ja omakorda lahkub sealt.

Joonis 9. Testkatse, kus pall veereb üle laua serva ja kukub põrandal olevale plaadile.

2.3.2 Ajalise lahutuse katse kirjeldus

Testitud sai ka ajalist lahutust. Selleks kasutasin TTi TG320 signaaligeneraatorit. Signaali kujuks valisin kastsignaal, mis imiteerib signaali, mis tekib palli veeremisest valgusanduri ette ja selle eest ära. Signaali pinge oli 5 V ± 0,25 V nihkega 2,5 V ± 0,06V. Aja mõõtmiseks kasutasin Arduino IDE sisseehitatud käsklust Millis ja Micros funktsiooni, mis annavad vastavalt millisekundites ja mikrosekundites möödunud aja Arduino käimapanemise hetkest.

Mõõdetud sai signaali tõusvat ja langevat hetke, kui väärtus oli tõusnud üle ja langenud alla vastavat nivoo, mis oli seatud miinimum- ja maksimumväärtuse vahele. Programm on leitav Lisast 9.

2.3.3 Prototüübi programm

Seadme programm on kirjutatud Arduino IDE keskkonnas. Seadmega mõõtmine algab mõlema anduri nivoo määramisest potentsiomeetriga, mida ületav pinge loetakse sündmuseks. Seadme ekraanil jookseb ka vastav instruktsioon. Esimene nivoo tuleb määrata valgusandurile. Ekraanile kuvatakse nii anduri signaal kui ka potentsiomeetriga seatud tulemuse nivoo väärtus reaalajas. Sobiva nivoo korral tuleb vajutada “Set/Start” nuppu. Järgmisena tuleb paika seada kiirendusanduri nivoo. Selleks kuvatakse ekraanile potentsiomeetriga seatud tulemuse nivoo väärtus ja kiirendusanduri signaali maksimaalne väärtus. Ekraanil uuendatakse potentsiomeetri ja kiirendusanduri maksimaalset väärtus siis kui on tuvastatud uus maksimaalne väärtus. Täpsema selgituse selle kohta leiab peatükist 2.3.4. Maksimaalset väärtust on võimalik nullida vajutades ”Reset” nuppu. Peale selle nivoo valimist väljub programm kalibratsiooni tsüklist ja siseneb mõõtmistsüklisse.

Mõõtmise alustamiseks tuleb vajutada “Set/Start” nuppu. Seejärel programm ootab järgmisi sündmuseid:

1. Pall tuleb fotodioodi ette.
2. Pall lahkub fotodioodi eest.
3. Pall maandub plaadile.

Sündmuseid oodatakse täpselt selles järjekorras. Järgmise sündmuse aega ei registreerita, kui eelnev pole toimunud, et kiirendada programmi mõõtmistsüklit ja vältida olukorda, kus kiirendusandur detekteerib nivoost kõrgema väärtuse enne, kui katse selleni jõudnud on. Mõõtetulemusena kuvatakse ekraanile sündmuste vahelised ajad millisekundites. Esimene aeg annab kuuli anduri eest möödumisaja ja teine aeg on vabalangemisele kuluv aeg. Katse taas sooritamiseks tuleb vajutada “Reset” nuppu. Antud hetkel uute nivoode sisestamiseks tuleb seade välja lülitada ja taaskäivitada. Programmi on leitav Lisast 10.

2.3.4 Ebaõnnestumised ja tulevane arendus

Prototüüpi valmistades ja seda katsetades oli mitmeid tagasilööke, osad neist jäävad edasise arendustöö parandamiseks. Prototüüpi sai seadistatud nii palju, et sellega oleks võimalik läbi viia testkatsed veendumaks, et sellega on võimalik reaalseid katseid läbi viia.

Seadmel oli omapära testimise päevadel esimesel käivitamisel mitte korrektselt tööle minna. See tähendas seda, et Arduinole tuli kogu programm uuesti peale laadida, mõnikord ei piisanud ka ühest korrast. Miks ta korrektselt tööle ei läinud jäi selgusetuks. Seade seisis kasutamiskordade vahelisel ajal puutumatult kapis. Kindlasti lõppseadme seisukohalt tahaks probleem täpsemat uurimist saada.

Mitmel korral oli probleemiks ekraanile info kuvamine. Nimelt ekraanile kuvatava info saatmine võttis üsnagi palju aega. Esimene probleem tekkis seadme kalibreerimise tsüklis kiirendusanduri väärtuse kuvamisel. Palli põrkamine vastu alust oli kordades kiirem, kui ekraanil info uuenemine ehk palli põrkamise moment võis toimuda hetkel, mil parasjagu saadeti uut infot ekraanile. Seega muutsin koodi nii, et ekraani väärtust uuendatakse juhul, kui kiirendusandur registreerib suurema kiirenduse väärtuse, kui varasemalt toimunud on. “Reset” nupuga oli võimalik maksimumtulemus nullida. Sellest hetkest oli ka selge, et mõõtmistsüklis tuleb aegade kuvamine tõsta tsükli lõppu. Ühe lahendusena võiks välja pakkuda teise Arduino Nano lisamist seadmesse, mis tegeleks ainult ekraanile info kuvamisega.

Järgmises arenguetapis tuleb tegeleda ka nähtusega, mida inglise keeles nimetatakse “debouncing” ja mis tekib nupu vajutamise hetkedel. Nimelt selgub, et nuppu vajutades nupu kontaktid põrkavad üksteise vastu mitu korda enne, kui saavutavad pideva kontaktse ühenduse, saates mikrokontrollerile mitu korda signaali, et nupp on alla vajutatud. See tähendas, et näiteks kalibratsiooni tsüklis oli võimalik ühe vajutusega läbida kogu tsükkel ja alustada isegi mõõtmist. Programmis on küll implementeeritud aeg, mille jooksul oodatakse, et pidev kontakt oleks tekkinud, kuid isegi praeguse implementatsiooniga juhtub olukordi, kus nupuvajutusega tehakse kaks sammu - juhtub harva, kuid siiski juhtub.

Kiirendusanduri võiks valida selliselt, mis ühendub otse Arduino analoogsisendisse. Antud töös kasutusel olnud anduri miinuseks on, et anduri väärtused saadetakse andurilt

mikrokontrollerile. See tähendab, et tekib viivitus tehes päringut andurile ja anduri vastamisega. Kuna I2C on mõeldud tegelikkuses ühe elektroonikaplaadi peal olevate seadmetega suhtlemiseks, siis anduri juhtme otsa panek, olenevalt juhtme pikkusest, mõjutab saadetava info korrektsust.

Programmi ülesehitust tuleks muuta selliselt, et oleks võimalik kasutada ka muid andureid. Teha anduri tüübile kas tuvastusmeetod või valida igale anduri tüübile vastav Arduino analoogsisend. Kalibratsioonitsükkel peaks vastama seadme küljes oleva anduri tüübile. Mõõtmistsüklis peaks olema võimalik valida sündmuste järjekorda ja mõõtmisrežiimi, et kas mõõdetakse sündmustevahelist aega või ühe anduri juhul pidevalt anduri väärtust. Mõõtmistsükli kiiremaks läbimiseks võiks kasutada mikrokontrollerile sisseehitatud katkestuste tabelit. See muudaks anduriväärtuse võrdlust nivooga automaatsemaks. Ja ebavajalikud katkestused tuleks tabelis välja lülitada, et programm oleks võimalikult vähe häiritud.

2.4 Katsete tulemused

Esimeseks katseks oli üle laua serva veereva palli kukkumine. Katses mõõtsin kahte aega: aega, mis kulub palli liikumiseks fotodioodi eest läbi ja aega, mis kulub palli vabalangemisele. Tegin kaks katseseeriat, mõlemas seerias oli 10 katset. Palli maandumisplaadi ja pallirenni vertikaalseks kauguseks oli 75 cm ± 1 mm ja palli veerema laskmise kõrgus renni pinnast 49 cm ± 1 mm ning palli läbimõõt 2,4 cm ± 1 mm.

Esimeses seerias andis fotodioodile maksimaalse väärtuse klassiruumi valgustus. Nivoo fotodioodi jaoks seadsin klassiruumi valgustuse ja palliga kaetud väärtuste keskele. Kiirendusanduri nivoo seadsin veidi madalamaks kui palli põrkes mõõdetav maksimaalne väärtus.

Teises katse seerias lisasin katsesse fotodioodi valgustama telefoni välklambi. Nii vähendasin klassiruumi valguse mõju, näiteks liikumisest tekkinud varjud fotodioodile. Selle käigus muutus ka maksimaalse valguse tase fotodioodile, seega reguleerisin uuesti ka katse nivoo fotodioodi jaoks.

Tabelist 1 on näha, et esimesel katse seerial on palli liikumise aeg Δt_1 fotodioodi eest olnud väga erinev võrreldes teise seeria tulemustega. Esimese seeria puhul on ka näha, et aja

väärtused erinevad suuresti. Põhjuseks võib olla madal nivoo tase ja ka klassiruumis liikumisest tekkinud varjatud. Teise katseseeria puhul on näha, et tulemused on palju ühtlasemad üle katseseeria. Lähtudes energia jäävuse seadusest ja võttes raskuskiirenduseks 9,81 m/s^2 leidsin arvutustest, et pall oleks pidanud veerema fotodioodi eest läbi ajaga 7,7 ms ± 0,37 ms. Esimese katseseeria tulemused erinesid sellest väärtusest suurel määral. Ka teise katseseeria tulemused erinevad sellest väärtusest rohkem kui 10 %.

Mõõdetud vabalangemise aeg on mõlemas seerias üsna sarnane. Kasutades liikumisvõrrandit, sain teoreetiliseks kukumise ajaks 391 ms ± 0,30 ms, mis on ka sarnane teises katseseerias saadud keskmisega, olles teoreetilisest väärtusest vaid 0.5 % suurem.

Teoreetiliselt leitud aegade määramatused on leitud 95% usaldusnivool.

Tabel 1. Prototüübi katse tulemused, laualt maha veerev pall.

Katse nr	1. Seeria		2. Seeria	
	Δt_1 (ms)	Δt_2 (ms)	Δt_1 (ms)	Δt_2 (ms)
1	2	400	12	402
2	1	402	12	392
3	2	414	11	394
4	3	406	12	390
5	1	398	13	384
6	1	394	12	389
7	6	393	11	391
8	5	396	12	407
9	0	398	11	394
10	1	401	12	391
Keskmine:	2	400	12	393

Kuna mõõdetud palli veeremise aeg fotodioodi eest ei rahuldanud ootuseid, sai üles seatud katse ajalise lahutuse kontrollimiseks. Katse kirjelduse leiab peatükist 2.3.2. Tabelis 2 on toodud testsignaalide sagedused, poolperioodide pikkused nii millisekundites kui mikrosekundites,

katsetulemuste keskmised ja keskmiste erinevus protsentides arvutuslikust poolperioodist. Sageduste täpsus on viimane numbrikoht. Kõik mõõtmistulemused leiab Lisast 11.

Tabel 2. Prototüübi ajalise lahutuse mõõtmis tulemused.

Sagedus (Hz)	Arvutuslik poolperiood (ms)	Mõõtmiste keskmine (ms)	Vea protsent (%)	Arvutuslik poolperiood (us)	Mõõtmiste keskmine (us)	Vea protsent (%)
1,000	500	499	0,20	500000	499002	0,20
2,004	250	249	0,40	249501	249102	0,16
10,02	50	49	2,00	49900	49420	0,96
20,07	25	25	0,00	24913	24549	1,46
100,3	5	4	20,00	4985	4355	12,64
499,5	1	0	100,00	1001	364	63,64
1000	-	-	-	500	112	77,60

On näha, et millisekundites mõõdetud poolperioodide viga suureneb lineaarselt. Mikrosekundite poolperioodi viga suureneb samuti, kuid mitte nii lineaarselt. Mikrosekundites mõõdetud aegadel on tekkinud ka väiksemad vead kui millisekundites mõõdetud aegadel. Ajalise lahutuse kontrollkatse näitas, et peatükis 2.3.1 kirjeldatud palli veeremise katses olek tulnud prototüübi programmis kasutada aja mõõtmiseks funktsiooni Micros. Kuna mikrosekunites mõõtes on tekkinud väiksem ajaline viga kui millisekundites mõõtmisel. Katsest järeldub ka, et palli veeremise katses pole programm optimeeritud ja mõõtmisele kulub lisa aeg.

Kokkuvõte

Töö esimeses pooles uurisin LEGO Mindstorm EV3 mikrokontrolleri mõõtmiste ajalist lahutust. Testkatsetes programmeerisin mikrokontrolleri mõõtma erineva sagedusega siinussignaale. Töö teises pooles arendasin ja testisin Arduino mikrokontrolleril põhinevat mõõtmisseadme prototüüpi.

LEGO Mindstorms EV3-ga tehtud katsed näitasid, et see süsteem ei suuda sensorilt andmeid lugeda kiiremini kui 300 lugemit sekundis - sellele piirile lähenedes muutub aegrida astmeliseks, st järjestikused lugemid on ühesuguse väärtusega, ehkki sisendsignaal muutub. See viitab selgelt, et süsteem ei ole suuteline andurist kiiremini lugemeid võtma. Mõõtmised kiirustel 100 ja 200 lugemit sekundis andsid paremad tulemused, kuigi ka need tulemused jätsid graafikusse astmelise kujuga sälke. Seega Nyquist-i kriteeriumi järgi jäid proovitud testsignaalides mõõdetavaks 1 Hz, 5 Hz, 10 Hz ja 50 Hz. See aga ei rahuldanud algset töö eesmärki.

Töö teises osas tehtud katsetest Arduino Nanoga saab järeldada, et mõõdetud aeg erineb keskmiselt 1 ms võrra. Selline tulemus on sündmuste juures, mis juhtuvad väiksema sagedusega kui 20 Hz, aktsepteeritav, kuid lühema ajaperioodi juures kui 25 ms hakkab 1 ms erinevus suuremat rolli mängima. Tehtud katsetest tuli ka välja, et Arduino Nano mõõdab aega lühemalt kui tegelikult aega kulus, see vajaks täpsemalt uurimist, kui on soov lühemaid aegu mõõta kui 25 ms.

Kirjandus

[1] Mehaanika. “Mehaanika”. 2021. [Võrgumaterjal]. Kättesaadav: https://opik.fyysika.ee/index.php/book/section/8415#/section/8416. [Kasutatud 25 05 2021].

[2] Mehaanika: dünaamika, perioodilised liikumised. “Mehaanika: dünaamika, perioodilised liikumised”. 2021. [Võrgumaterjal]. Kättesaadav: https://opik.fyysika.ee/index.php/book/view/78#/section/35405. [Kasutatud 25 05 2021].

[3] LabQuest. “LabQuest”. 2021. [Võrgumaterjal]. Kättesaadav: https://www.vernier.com/product/labquest-2/. [Kasutatud 25 05 2021].

[4] Pasco Spark. “Pasco Spark”. 2021. [Võrgumaterjal]. Kättesaadav: https://www.pasco.com/products/interfaces-and-dataloggers. [Kasutatud 25 05 2021].

[5] G. Brockington, M. Schivani, C. Barscevicius, T. Raquel, M. Pietrocola. “Using robotics in kinematics classes: exploring braking and stopping distances,” Physics Education, volume 53-025012, 2018.

[6] O. Mubin, C. J. Stevens, S. Shahid, A. A. Mahmud,J.-J. Dong. “A review of the applicability of robots in education,” Technology for Education and Learning, 2013.

[7] LEGO. “LEGO Mindstorm EV3”. 2021. [Võrgumaterjal]. Kättesaadav: https://www.lego.com/en-us/product/lego-mindstorms-ev3-31313. [Kasutatud 25 05 2021].

[8] Vernier. “NXT Sensor Adapter”. 2021. [Võrgumaterjal]. Kättesaadav: https://www.vernier.com/products/interfaces/bta-nxt/. [Kasutatud 25 05 2021].

[9] Python. “Python”. 2021. [Võrgumaterjal]. Kättesaadav: https://www.python.org/. [Kasutatud 25 05 2021].

[10] Arduino Nano. “Arduino Nano”. 2021. [Võrgumaterjal]. Kättesaadav: https://store.arduino.cc/arduino-nano [Kasutatud 25 05 2021].

[11] ATmega328P andmeleht. “ATmega328P Data Sheet”. 2018. [Võrgumaterjal]. Kättesaadav: http://ww1.microchip.com/downloads/en/DeviceDoc/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061A.pdf [Kasutatud 25 05 2021].

[12] ERM802-3 Seeria kuvari andmeleht. “ERM802-3 Series Character Module Datasheet”. 2012. [Võrgumaterjal]. Kättesaadav: http://www.buydisplay.com/download/manual/ERM802-3_Series_Datasheet.pdf [Kasutatud 25 05 2021].

[13] LiquidCrystal teek. “LiquidCrystal Library”. 2019. [Võrgumaterjal]. Kättesaadav: https://www.arduino.cc/en/Reference/LiquidCrystal [Kasutatud 25 05 2021].’

[14] OPT101 Monoliitne fotodiood andmeleht. “OPT101 Monolithic Photodiode and Single-Supply Transimpedance Amplifier”. 2015. [Võrgumaterjal]. Kättesaadav: https://www.ti.com/lit/ds/symlink/opt101.pdf?ts=1590439954776 [Kasutatud 25 05 2021].

[15] MPU6050 andmeleht. “MPU-6000 and MPU-6050 Product Specification Revision 3.4”. 2013. [Võrgumaterjal]. Kättesaadav: https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf [Kasutatud 25 05 2021].

[16] I2C. “I2C-bus specification and user manual”. 2014. [Võrgumaterjal]. Kättesaadav: https://www.nxp.com/docs/en/user-guide/UM10204.pdf [Kasutatud 25 05 2021].

[17] MPU6050 ja I2Cdev teek. “MPU6050 Arduino Library”. 2021. [Võrgumaterjal]. Kättesaadav: https://github.com/ElectronicCats/mpu6050 [Kasutatud 25 05 2021].

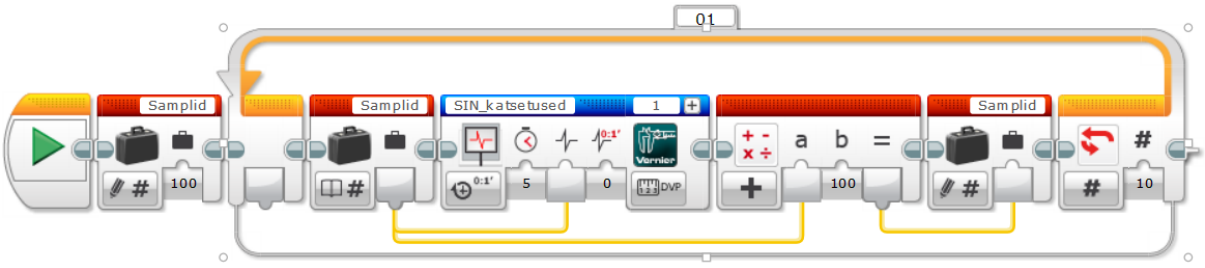
Lisad

Lisa 1.

LEGO Mindstorm EV3-le koostatud programm ajalise lahutuvuse testimiseks. Programm koosneb mitmest erinevast programmeerimise plokist:

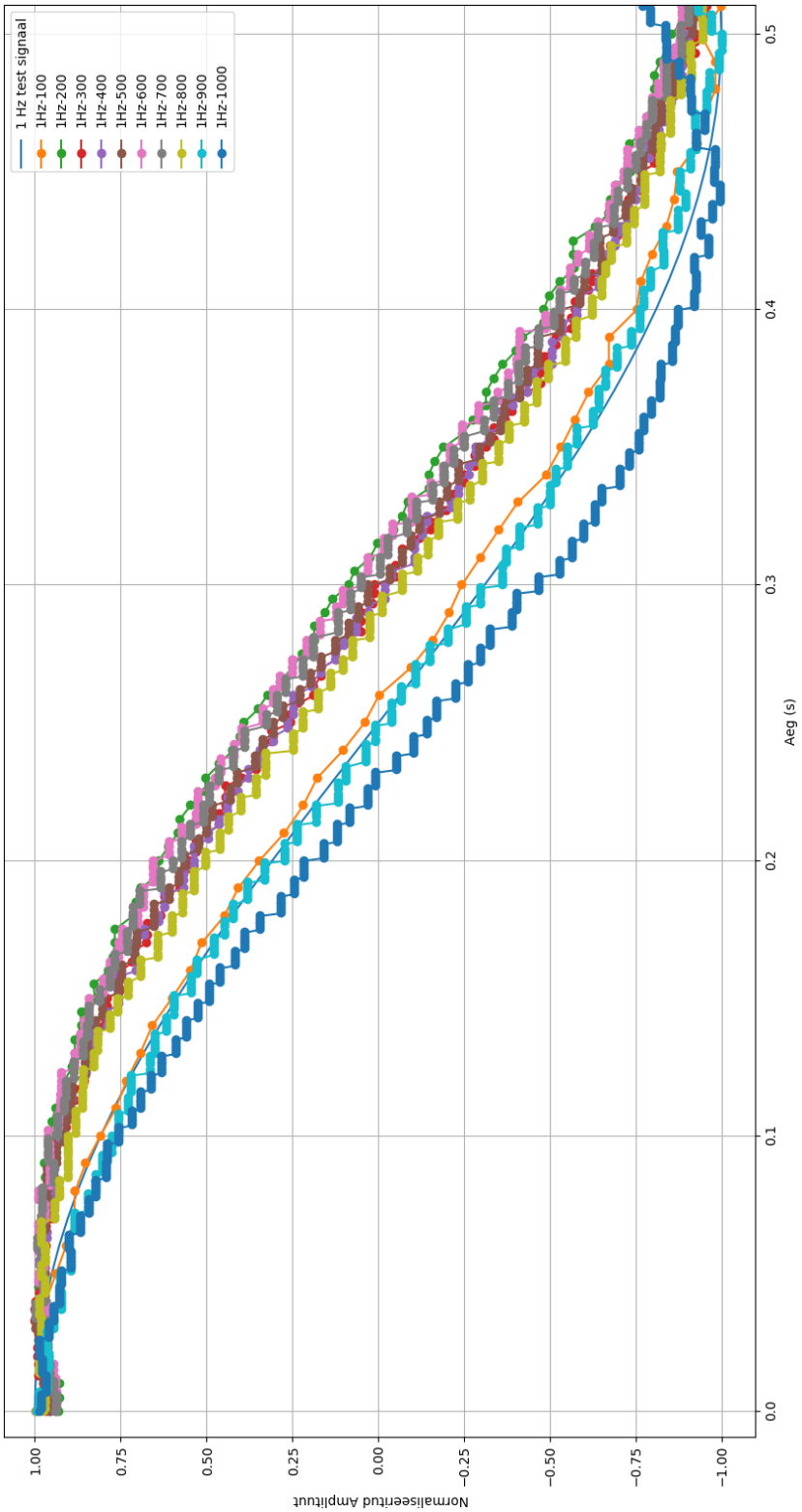
- Programmi algust tähistavast plokist
- Muutuja plokist
- Andmelogija plokist, mis võtab parameetriteks:
    - salvestusaja sekundites
    - andmepunktide arvu sekundis
    - kolmas väli tähistab töörežiimi “andmepunkti sekundi jooksul”

  Antud plokk teeb ise faili salvestamise.
- Liitmistehte plokk, et suurendada lugemiskiirust järgmisel mõõtmisel
- Kordustsükkli plokk



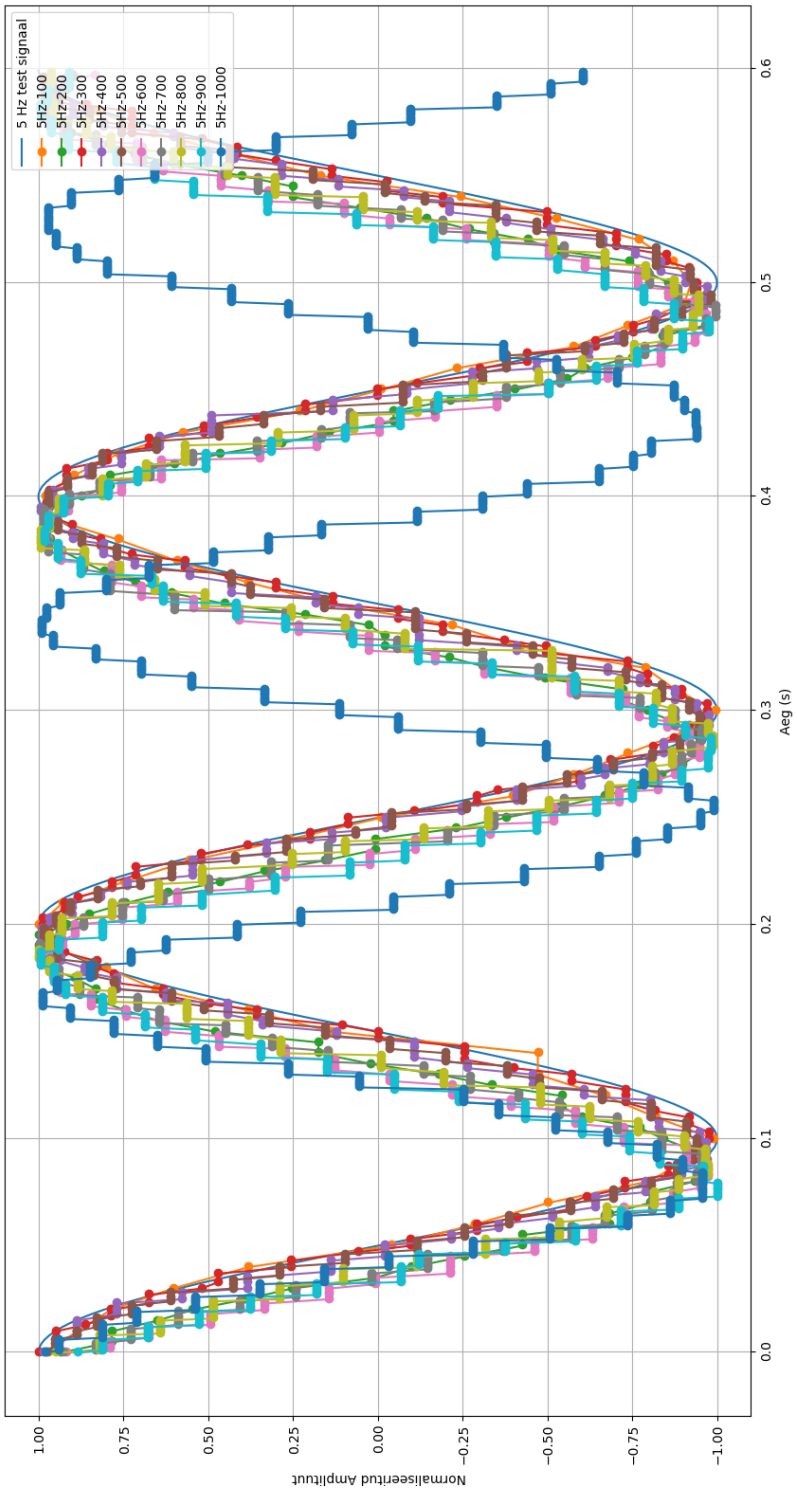
Lisa 2.

EV3-ga mõõdetud 1 Hz testsignaal kõigi testitud lugemise kiirustega.



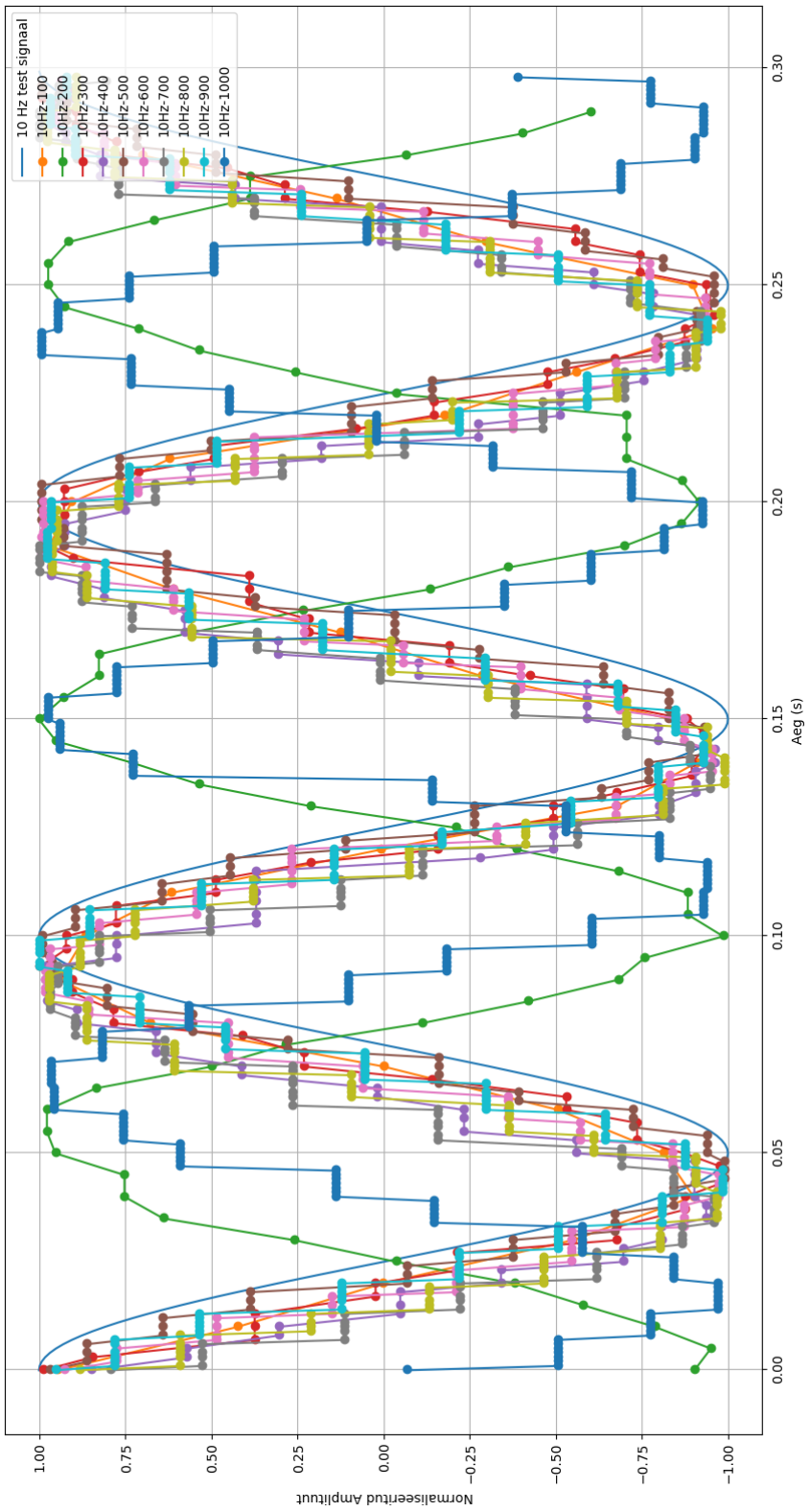
Lisa 3.

EV3-ga mõõdetud 5 Hz testsignaal kõigi testitud lugemise kiirustega.



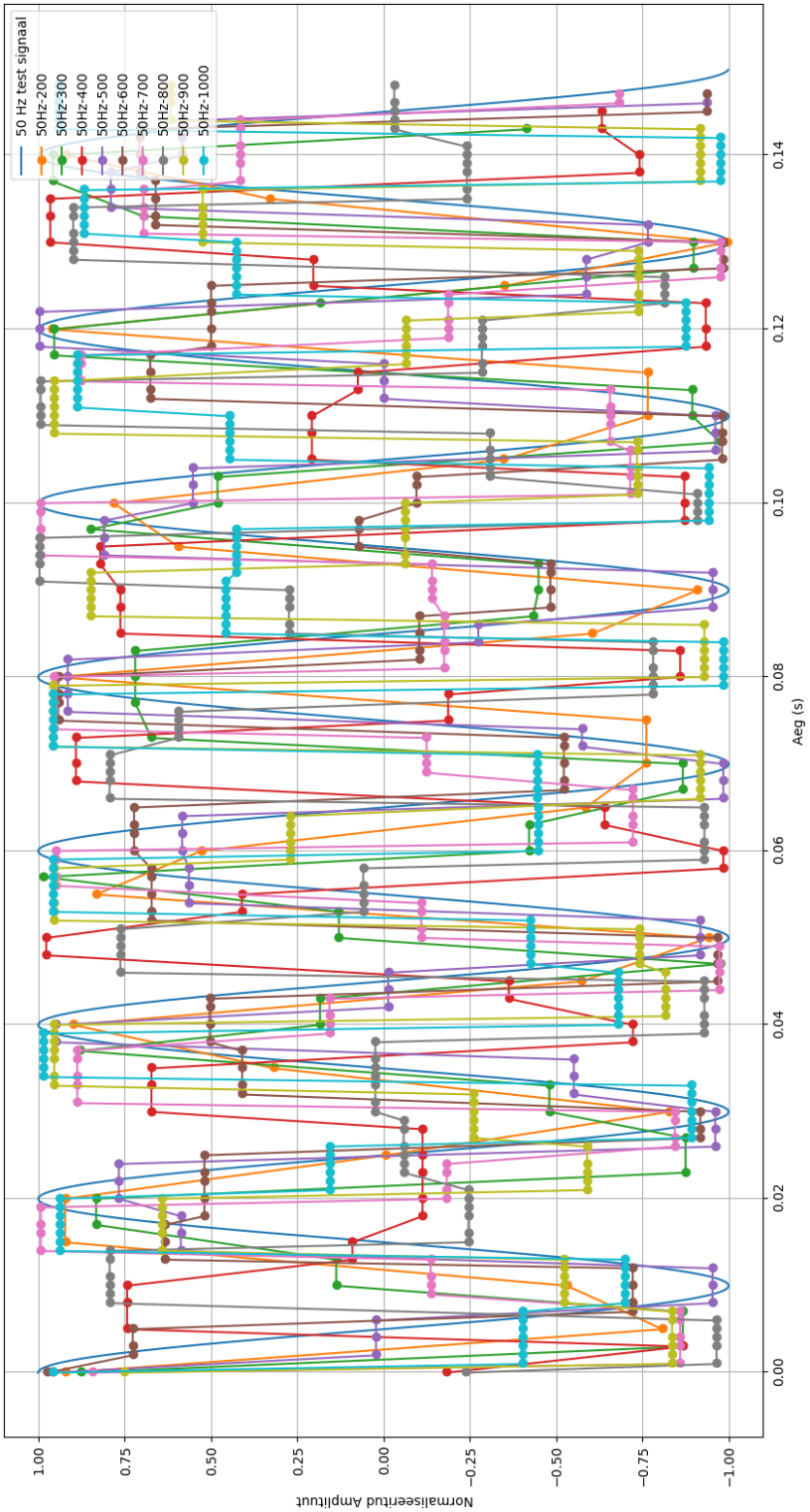
Lisa 4.

EV3-ga mõõdetud 10 Hz testsignaal kõigi testitud lugemise kiirustega.



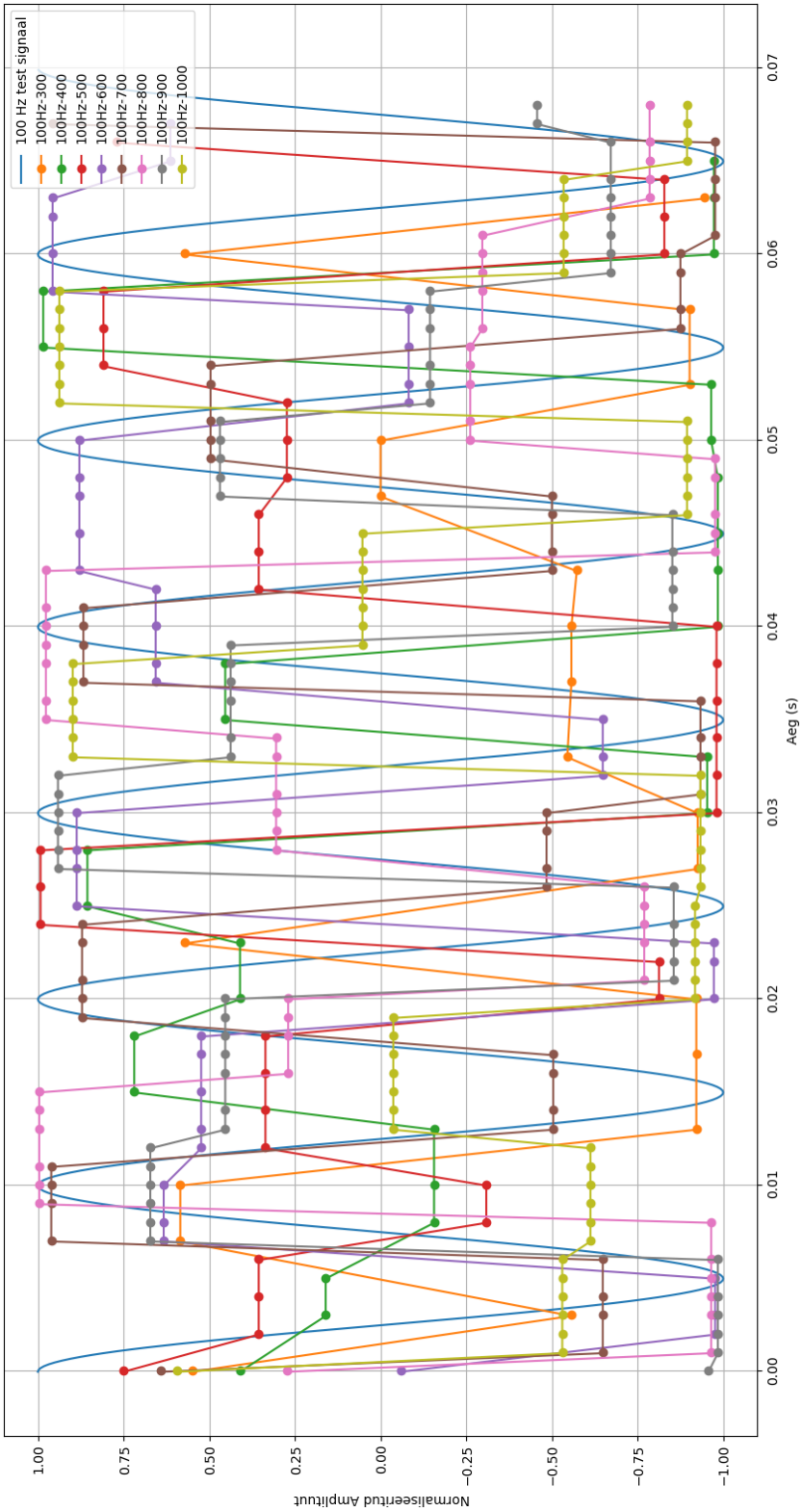
Lisa 5.

EV3-ga mõõdetud 50 Hz testsignaal kõigi testitud lugemise kiirustega.



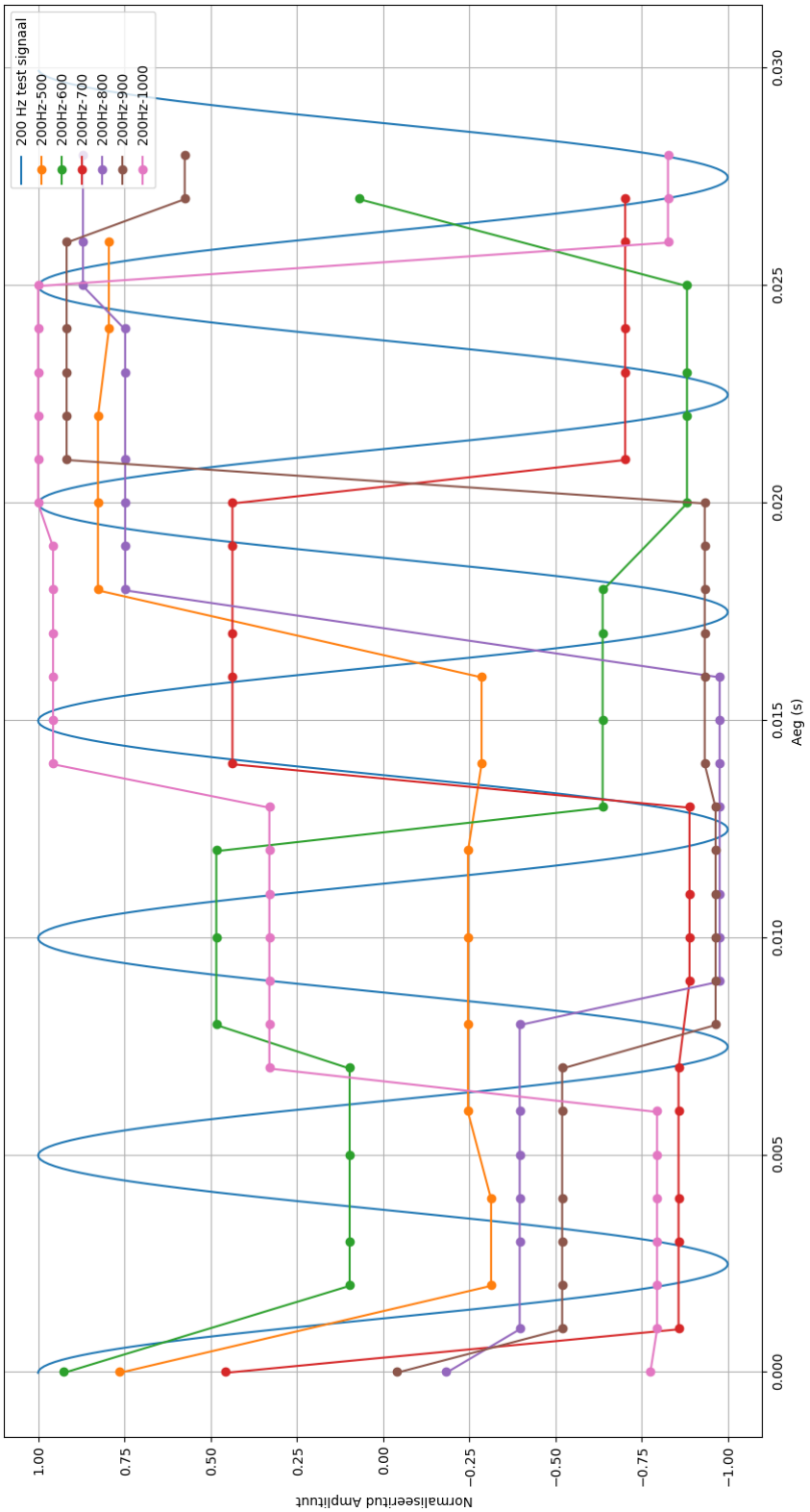
Lisa 6.

EV3-ga mõõdetud 100 Hz testsignaal kõigi testitud lugemise kiirustega.



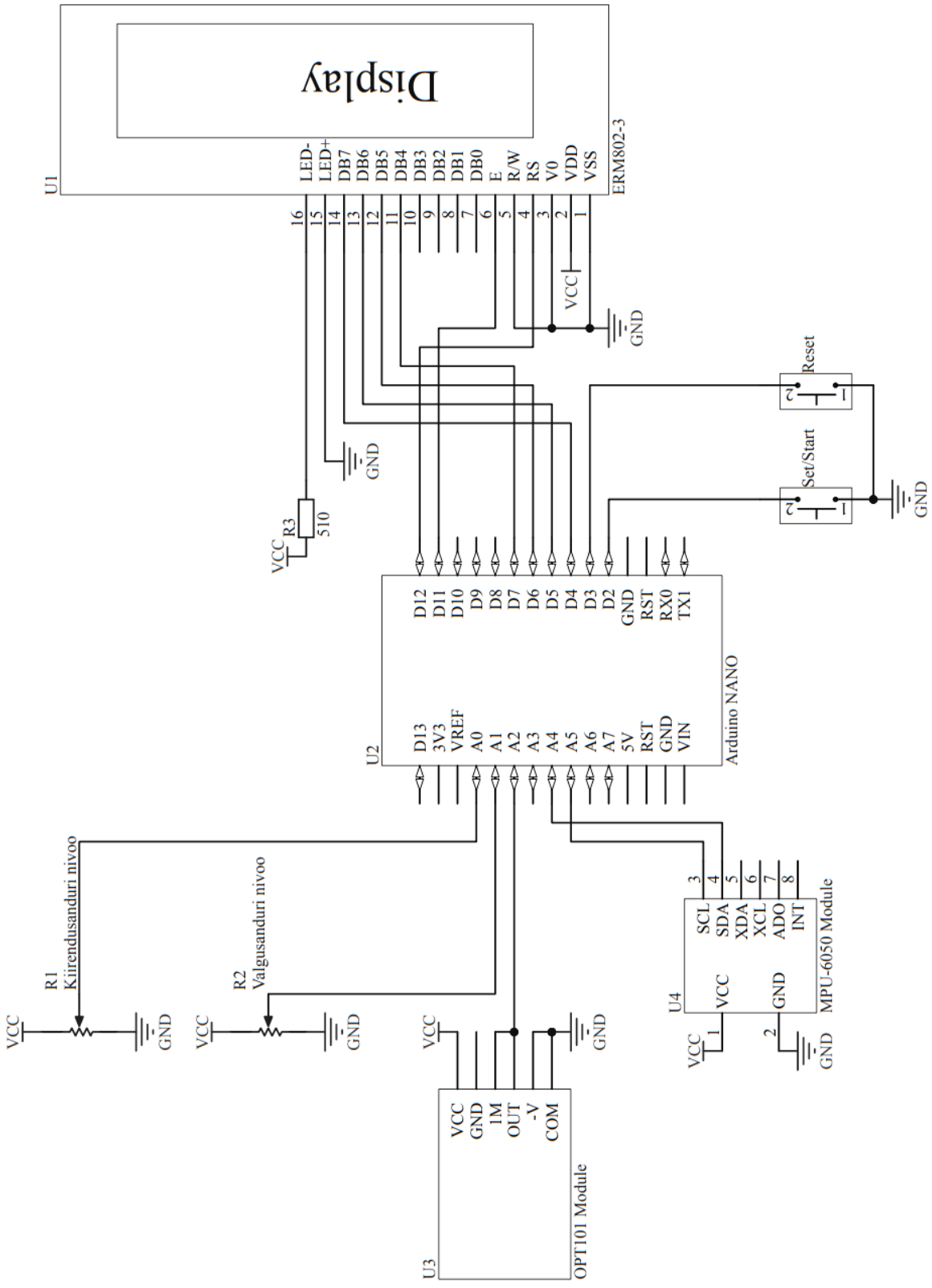
Lisa 7.

EV3-ga mõõdetud 200 Hz testsignaal kõigi testitud lugemise kiirustega.



Lisa 8.

Prototüübitud seadme elektriskeem.



Lisa 9.

Programm prototüübi ajalise lahutuse mõõtmiseks.

```
#define photo A2 //photogate pin

boolean t1 =false;
long time1;
long time2;

void setup() {
  Serial.begin(9600);
  Serial.println("Setup is started.");
}

void loop() {
  //Serial.println(analogRead(photo));
  if((analogRead(photo) > 512) & !t1){
      //time1 = millis();
      time1 = micros();
      t1 = true;
      //Serial.println(time1);
    }
    if((analogRead(photo) < 512) & t1){
      //time2 = millis();
      time2 = micros();
      t1 = false;
      //Serial.println(time2);
      Serial.println(time2-time1);
    }
}
```

Lisa 10.

Prototüübi programm palli veeremise katsele.

```
#include <LiquidCrystal.h> // include the display library
#include "MPU6050.h" // include the accelerometer library

#define pot A0 // potentiometer pin
#define pot1 A1
#define photo A2 //photogate pin
#define but 2 //button pin for set/start
#define but2 3 //button pin for reset

LiquidCrystal lcd(12, 11, 7, 6, 5, 4); //display pin settup
MPU6050 accelgyro; //accelometer setup

enum Task {
  Calib,
  Main
}; //Creating tasks

//setting variables
Task task = Calib;
int16_t acc;
int16_t max_acc;
int baseline;
volatile boolean state = false;
volatile boolean cali = false;
volatile boolean measure = false;
int l_sensor;
int acc_sensor;
boolean t1 =false;
boolean t2 =false;
boolean t3 =false;
long time1;
long time2;
long time3;

//setting functions
void calibration();
void measureTask();
int readAnalog(int pin);
void pressed();
void pressed2();
boolean debounced();
void setHeading(String text);
void setLHeading(String text);
void setValues(int value, int x = 0, int y = 1);

void setup() {
```

```
  Wire.begin();      // join I2C bus
  Serial.begin(9600);
  Serial.println("Setup is started.");
  lcd.begin(8, 2);// set up the LCD's number of columns and rows:
  pinMode(photo,INPUT);
  pinMode(pot,INPUT);
  pinMode(but,INPUT_PULLUP);
  pinMode(but2,INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(but), pressed, FALLING);
  attachInterrupt(digitalPinToInterrupt(but2), pressed2, FALLING);

  accelgyro.initialize();  // initialize the accelerometer
  for (int i = 0; i < 10; i++){
    acc += abs(accelgyro.getAccelerationX());
  }
  baseline = acc/10;
  max_acc = acc/10;
  Serial.println("Setup is finished.");
}

void loop() {
  switch(task) {
    case Calib:
      calibration();
      break;
    case Main:
      if (measure){
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("t1 ");
        lcd.print(0);
        lcd.setCursor(0, 1);
        lcd.print("t2 ");
        lcd.print(0);

        measureTask();

        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("t1 ");
        lcd.print(time2-time1);
        lcd.setCursor(0, 1);
        lcd.print("t2 ");
        lcd.print(time3-time2);
        Serial.println("uuendatud");
      }
      break;
  }
}
```

```
void measureTask(){
  while(measure){
    if((analogRead(photo) < l_sensor) & !t1){
      time1 = millis();
      t1 = true;
    }
    if((analogRead(photo) > l_sensor) & t1 & !t2){
      time2 = millis();
      t2 = true;
    }
    acc = abs(accelgyro.getAccelerationX());  // read measurements from
device
    if(((acc-baseline)*0.05 > acc_sensor) & t2 & !t3){
      time3 = millis();
      t3 = true;
      measure = false;
    }
  }
}

void setHeading(String text){
  lcd.clear();
  lcd.setCursor(8, 0);
  lcd.autoscroll();

  for (int thisChar = 0; thisChar < text.length(); thisChar++) {
    lcd.print(text[thisChar]);
    delay(200);
  }

  for (int thisChar = 0; thisChar < 8; thisChar++) {
    lcd.print(" ");
    delay(200);
  }

  lcd.noAutoscroll();
  delay(500);
  lcd.clear();
}

void setLHeading(String text){
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print(text);
}

void setValues(int value, int x, int y){
  lcd.setCursor(x, y);
  lcd.print("    ");
  lcd.setCursor(x, y);
```

```
  lcd.print(value);
  delay(100);
}

boolean debounced(int dbtime){
  static unsigned long last_interrupt_time = 0;
  unsigned long interrupt_time = millis();
  if (interrupt_time - last_interrupt_time > dbtime)
  {
    return true;
  }
  last_interrupt_time = interrupt_time;
}

void pressed() {
  switch(task) {
    case Calib:
      state = debounced(200);
      Serial.println("Nupp 1 pressed. ");
      break;
    case Main:
      measure = true;
      break;
  }
}

void pressed2() {
  switch(task) {
    case Calib:
      max_acc = 0;
      break;
    case Main:
      if(debounced(200)){
        t1 =false;
        t2 =false;
        t3 =false;
      }
      break;
  }
}

int readAnalog(int pin){
  int analog = 0;
  for (int i = 0; i < 10; i++){
    analog += analogRead(pin);
  }
  return (analog/10);
}

void calibration(){
```

```
  int photosensor;
  int pot1sensor;
  int potsensor;

  setHeading("Kalibreerimine");
  delay(100);

  setLHeading("MAX val.");
  state = false;
  while(true){
    photosensor = readAnalog(photo);
    pot1sensor = readAnalog(pot1);
    setValues(photosensor, 0, 1);
    setValues(pot1sensor, 4, 1);
    if(state){
      l_sensor = pot1sensor;
      break;
    }
  }
  delay(100);

  setLHeading("TRIG ACC");
  state = false;
  while(true){
    potsensor = readAnalog(pot);
    acc = abs(accelgyro.getAccelerationX());
    if (acc > max_acc){
      max_acc = acc;
      setValues((max_acc-baseline)*0.05, 0, 1);
      setValues(potsensor, 5, 1);
    }
    if(state){
      acc_sensor = potsensor;
      break;
    }
  }
  delay(100);

  task = Main;
  setHeading("Kalibreerimine on lõppenud");
  delay(100);
}
```

Lisa 11.

Arduino ajalise lahutuse määramise katse tulemused.

Sagedus (Hz)	Mõõtmis andmed (ms)																			
1.000	500	499	500	500	499	498	500	498	499	499	499	498	500	499	499	500				
2.004	249	249	247	248	249	248	249	249	249	249	249	249	248	249	249	249				
10.02	49	50	50	49	51	49	49	50	49	50	49	49	49	49	50	49				
20.07	25	25	25	25	24	24	25	25	25	24	24	24	25	25	25	25				
100.3	5	4	4	4	6	5	4	4	5	5	4	4	4	5	4	4				
499.5	1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	1				
	Mõõtmis andmed (us)																			
1.000	499184	498960	498960	498960	498960	498960	499184	498960	498960	498960	498960	498960	498960	499184	498960	498960				
2.004	249200	248976	249200	249200	248976	249200	249200	248976	249200	249200	248976	248976	249200	248976	248976	249200				
10.02	49392	49616	49392	49616	49392	49392	49392	49392	49392	49392	49392	49392	49392	49392	49392	49392				
20.07	24304	24528	24304	24528	24528	24528	24528	24722	24722	24722	24722	24528	24528	24528	24528	24528				
100.3	4144	4368	4368	4592	4368	4368	4368	4368	4368	4368	4144	4368	4368	4376	4368	4376				
499.5	336	336	112	336	560	336	336	336	336	336	336	560	336	336	336	560				
1,000	112	112	112	112	112	112	112	112	112	112	112	112	112	112	112	112				

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Rimmo Rõõm,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose "Digitaalsed mõõtmised füüsikaõppes platvormidel LEGO Mindstorm EV3 ja Arduino", mille juhendaja on Kaido Reivelt, reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.

2. Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.

3. Olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile.

4. Kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Rimmo Rõõm

28.05.2021