TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Sander Tsõbulski 164826IASB

PROJEKTIÜLESANNETE VÄLJATÖÖTAMINE MOBIILSELE BOE-BOT ROBOTIPLATVORMILE

Bakalaureusetöö

Juhendaja: Priit Ruberg
PhD

Tallinn 2021

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Sander Tsõbulski

02.01.2021

Annotatsioon

Töö eesmärgiks oli välja mõelda ja luua erinevaid ülesandeid eriala Riistvara arendus ja programmeerimine esmakursuse tudengite jaoks aine Erialatutvustus (IAS0001) raames. Ülesannete loomine oli modifitseeritud Boe-Bot robotiplatvormile, mis baseerub Arduinol. Aine enda põhieesmärgiks oli üliõpilastele anda sissejuhatus robotite ülesehitusest ning nende programmeerimisest.

Lõputöö tegemine kujunes kolmeks etapiks. Esimesena tuli välja mõelda erinevaid ülesandeid tudengitele lahendamiseks. Teiseks pidi need ära jaotama omakorda kolmeks alamülesandeks. Kolmandaks oli katsetamine ja koodi kirjutamine.

Lõputöö teostamine kujunes kolmeks etapiks, kus esimeseks tuli välja mõelda ülesannete komplekt, mida tudengid hakkavad lahendama. Teiseks tuli ülesanded omakorda jaotada kolmeks etapiks. Viimasena toimus koodi kirjutamine ning katsetamine. Koodid on üleval GitLab repositooriumis, mille link on peatükis Lisa 6.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 32 leheküljel, 4 peatükki, 23 joonist, 3 tabelit.

Abstract

Project Tasks Development for the Mobile Boe-Bot Robot Platform

The purpose of this thesis is to create a set of tasks for freshman year students attending the subject Introduction to Speciality (in estonian Erialatutvustus) (IAS0001) in Hardware Development and Programming field of study. These tasks are for the modified Boe-Bot robot platform. The main intent for the assingments is to teach students the basics of robotic structure and how it works.

Firstly, a set of tasks had to be thought up for what the students could do with the Boe-Bot. Thinking of assigments to make was quite a hard task as it required a lot of thought process and experimenting with diferent modules. Also, composing exercises was limited as there were only a number of modules to work with.

Secondly, the tasks had to be seperated into standalone milestones, amount of three. For each excerice, students had two tweeks to accomplish. The purpose was to teach them that it is a good practice to cut the entire assingment into three different portions, so it would be easier to undertake.

Lastly, programs were written for the testing purposes to see if they were achiveable, which also took a very long time and a lot of effort. A link to the GitLab, wehere the codes are, is in Lisa 6 chapter.

All in all, thinking and creating excerices was a hard task. It requried an adequate amount of time to come up and test the tasks. As usually, problems accured, because there was an uncertainty as to what exact assignments can be done, because the number of diferent modules was limited. Also, not all the tasks in this thesis were adapted for the subject, reason being there was not enough time. The tasks had to be done in seven weeks and the preparations for it were not enough. Despite of that, a lesson was learned – communication and time management are very important.

The thesis is in estonian and contains 32 pages of text, 4 chapters, 23 figures, 3 tables.

Lühendite ja mõistete sõnastik

Mbps	*Megabit per second*, megabitti sekundis
PWM	*Pulse-width modulation*, pulsilaiusmodulatsioon
IDE	*Integrated Developement Environment*, integreeritud programmeerimiskeskkond
Creative Commons Attribution Share-Alike	*Creative Commons Attribution Share-Alike*, samalitsentsiliste alla kuuluv originaalteos
QTI	*Charge, Transfer, Infrared*, infrapunakiirguse emitter/vastuvõtja
SDA	*Data Signal/Serial Data*, andmete edastuse signaal
SCL	*Clock Signal/Serial Clock*, mikrokontrolleri seesmise kella muutmise signaal
SPI	*Serial Peripheral Interface*, sünkroonse järjestiksuhtluse liidese standard
ISM	*Industrial, scientific and medical*, raadiosagedusalad, mis on reserveeritud rahvusvaheliselt tööstuslikuks, teaduslikuks ning meditsiiniliseks otstarbeks
Joystick	Kursorhoob, liigutav hoob
CRC	*Cyclic Redundancy Check*, tsükkelkoodkontroll
Whitening	Võtipleegitus, krüptimine ja de- XOR šifferiga
XOR	Välistav disjunktsioon/või, binaarne tehe
PDU	*Protocol data unit*, protokolliandmeüksus juhtteabe või andmete terviklik üksus
I2C	*Inter-Integrated Circuit*, kahejuhtmeliides
Milestone	Eesmärk, agiilses meetodis kasutusel olev termin, mis tähistab progressiooni
Issue	Probleem, agiilses meetodis kasutusel olev termin, mis tähistab töö kurssi
MCU	*Microcontroller Unit*, mikrokontroller
EDR	*Enhanced Data Rate*, täiustatud andmeedastuskiirus
Põhiriba	Modulatsiooniga muutmata sagedusriba
AT	*Attention commands*, tähelepanu käsud, mis on loodud juhtmevaba seadmete lihtsamaks kontrollimiseks
MAC aadress	*Media Access Control*, meediumipöörduse juhtimise aadress, mis füüsilise seadme unikaalne identifitseerija.
BT	*Bluetooth*, Sinihammas, mis kujutab endast traadita andmesidet seadmete vahel

Sisukord

Jooniste loetelu

Tabelite loetelu

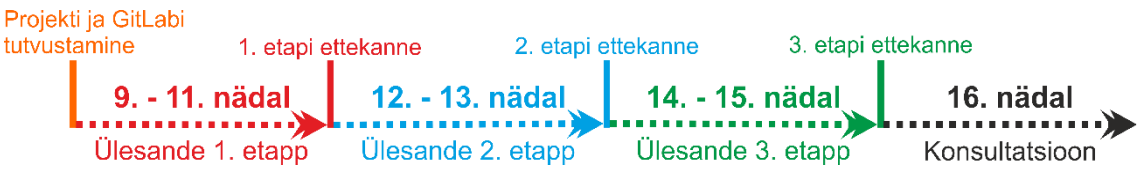
1 Sissejuhatus

Lõputöö eesmärgiks oli koostada Boe-Bot robotiplatvormile ülesandeid aine Erialatutvustus (IAS0001) raames, kus iga jaotati omakorda kolmeks etapiks. Lisaks osalesid aine ülesannete koostamisel Oskar Voorel ja Priit Ruberg. Lõputöö kujuneb kolmeks osaks, kus esimesena toimub ülesannete loomine, teiseks on programmide kirjutamine ja kolmandaks nende testimine.

Boe-Bot robotiplatvorm (Joonis 2) koosneb metallkerest, kahest tagarattast, ühest esiratta kuulist, Arduino Uno-st ja trükkplaadist ning maketeerimislauast, kuhu ühendatakse moodulid. Ülesannete enda loomine ning kirjutamine põhineb Arduino platvormil, mis baseerub C++ programmeerimiskeelel. Täpsem tutvustus asub peatükk kahes.

Aine enda põhieesmärgiks on tutvustada tudengitele Riistvara arendus ja programmeerimine õppeala olemust, mida hakkavad üliõpilased õppima ning millega tulevikus tegelema. Käesoleva aine osa pakub võimalust tundma õppida roboti ülesehitust ning algseid programmeerimisoskusi omandama. Lisaks on kasutusel GitLab repositooriumi keskkond, kuhu tudengid püstitavad oma koodid, probleemid ning edusammud. Samuti luuakse kahe- kuni kolmeliikmelised meeskonnad, kust igaüks saab olla tiimi vastutav kaheks nädalaks, mis õpetab neile koostööd ja vastutustundlikust.

Peale ülesande etappideks jaotamist, tuli iga jaoks välja mõelda veel kolm eri varianti, mis jaotati ära tudengitele 8 nädala peale (Joonis 1), kust ühe etapi lahendamise peale anti kaks nädalat, millest hiljem tuli koostada aruanne. See kujutas endast esitluse tegemist, kus räägiti, mis läks halvasti, hästi ning kus ollakse oma projektiga.



Joonis 1. Ülesannete tegemise jaotuse ajatelg.

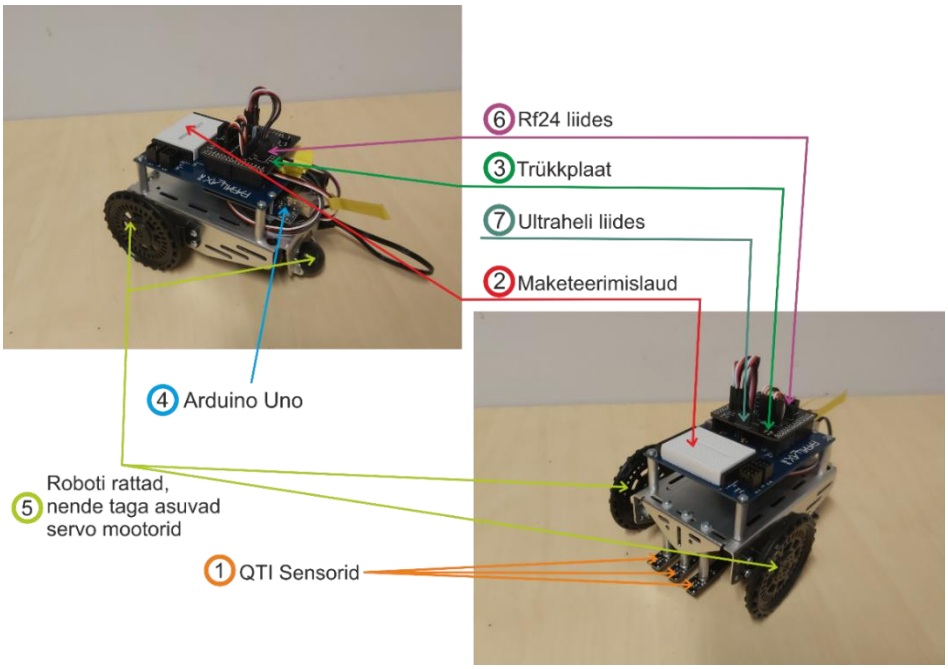
Ülesande jaotus kujunes kolme sektsiooni (Joonis 1), kus esimesel ajaperioodil toimus projekti ja GitLabi tutvustamine tudengitele ning ülejäänud 9. kuni 11. nädal esimese

etappiga tegelemine. Teisel perioodil (12. – 13. nädal) toimus teise ning kolmandal (14. – 15. nädal) kolmanda etapi tegemine. Lisaks iga ajaperioodi lõpus toimusid projektitööde ettekanded. 16. nädalal oli konsultatsiooni aeg tudengite jaoks, kus said tulla abi küsima. Eksamisessiooni ajal toimuvad projektide kaitsmised, kus enne kaitsmisele tulemist on vaja valmis saada ka aruanne.

2 Robotiplatvorm

Aine Erialatutvustus (IAS0001) tudengitel on võimalus tutvuda ning tegeleda Boe-Bot robotitega. See on robotiplatvorm, millega saavad üliõpilased tundma õppida roboti ülesehituse põhimõtteid, täpsemalt, mis paneb just selle tööle ja kuidas. Selleks on kasutusel erinevad moodulid ning Arduino programmeerimise keskkond (Joonis 4) koodide kirjutamiseks, mis ise baseerub C++ progammeerimiskeelel. Lisaks on internetist saadaval rohkelt materjali, mis aitavad kaasa programmide loomisel.

Boe-Bot robotil (Joonis 2) on peal maketeerimislaud (Joonis 2, punkt 2), trükkplaat (Joonis 2, punkt 3), Arduino Uno (Joonis 2, punkt 4), QTI sensorid (Joonis 2, punkt 1). ning robotil olevad servo mootorid (Joonis 2, punkt 5 ja Joonis 5). Lisaks on võimalus robotile külge panna raadioside moodul ja ultraheli sensor, mille jaoks on olemas spetsiaalsed kohad ning muid mooduleid, mida saab kinnitada maketeerimislauale. Robotil kokku on kolm trükkplaati, millest üks (Joonis 2, punkt 3) on disainitud Jürgen Soomi poolt.



Joonis 2. Boe-Bot robot.

2.1 Mikrokontroller Arduino Uno

Arduino Uno (Joonis 3) on avatud lähtekoodiga mikrokontrolleriplaat, mis põhineb ATmega328P mikrokontrolleril ja mille on välja töötanud Arduino.cc. Plaat on varustatud digitaalsete ja analoogsisendi/väljundi viikude komplektidega, mis võivad olla ühendatud erinevate laiendusplaatide (varjestuste) ja muude ahelatega. Plaadil on 14 digitaalset sisend-/väljundviiki (kuus *PWM*-väljundit), kuus analoogsisend-/väljundviiki, mis on programmeeritavad B-tüüpi (USB 2.0) USB-kaabli kaudu Arduino *IDE*-ga (integreeritud arenduskeskkond). Seda saab toita USB-kaabli või välise 9-voldise akuga, kuigi see aktsepteerib pingeid vahemikus 7 kuni 20 volti. Riistvara kujundust levitatakse *Creative Commons Attribution Share-Alike* 2.5 litsentsi alusel, mis on saadaval Arduino veebisaidil. [1]

Nimi "Uno" tähendab itaalia keeles ühte ja see valiti *Arduino Software (IDE)* 1.0 väljaandmise tähistamiseks. Uno plaat ja Arduino tarkvara (*IDE*) versioon 1.0 olid Arduino etalonversioonid, mis on nüüd arenenud uuemate väljalaseteni. Uno plaat on USB Arduino plaatide seerias esimene ja Arduino platvormi võrdlusmudel. [2]

Joonis 3. Arduino Uno. [2]

Ülesannete koostamisel oli kasutusel Arduino Uno. Sellel on piisavalt viike, et oleks võimalik ühendada nii Boe-Bot robotiplatvormil olevad ning ka lisa moodulid, kui peaks neid tarvis vaja minema. Samuti Boe-Botil olev trükkplaat on mõeldud täpselt Uno-ga kasutamiseks. Välkmälu suurus on 32 kilobaiti, mida ei ole palju, kuid väiksemate programmide puhul piisav. Samuti ka lõputööde raames tehtud programmid ei ole suuremahulised, et oleks tarvis suuremat välkmälu.

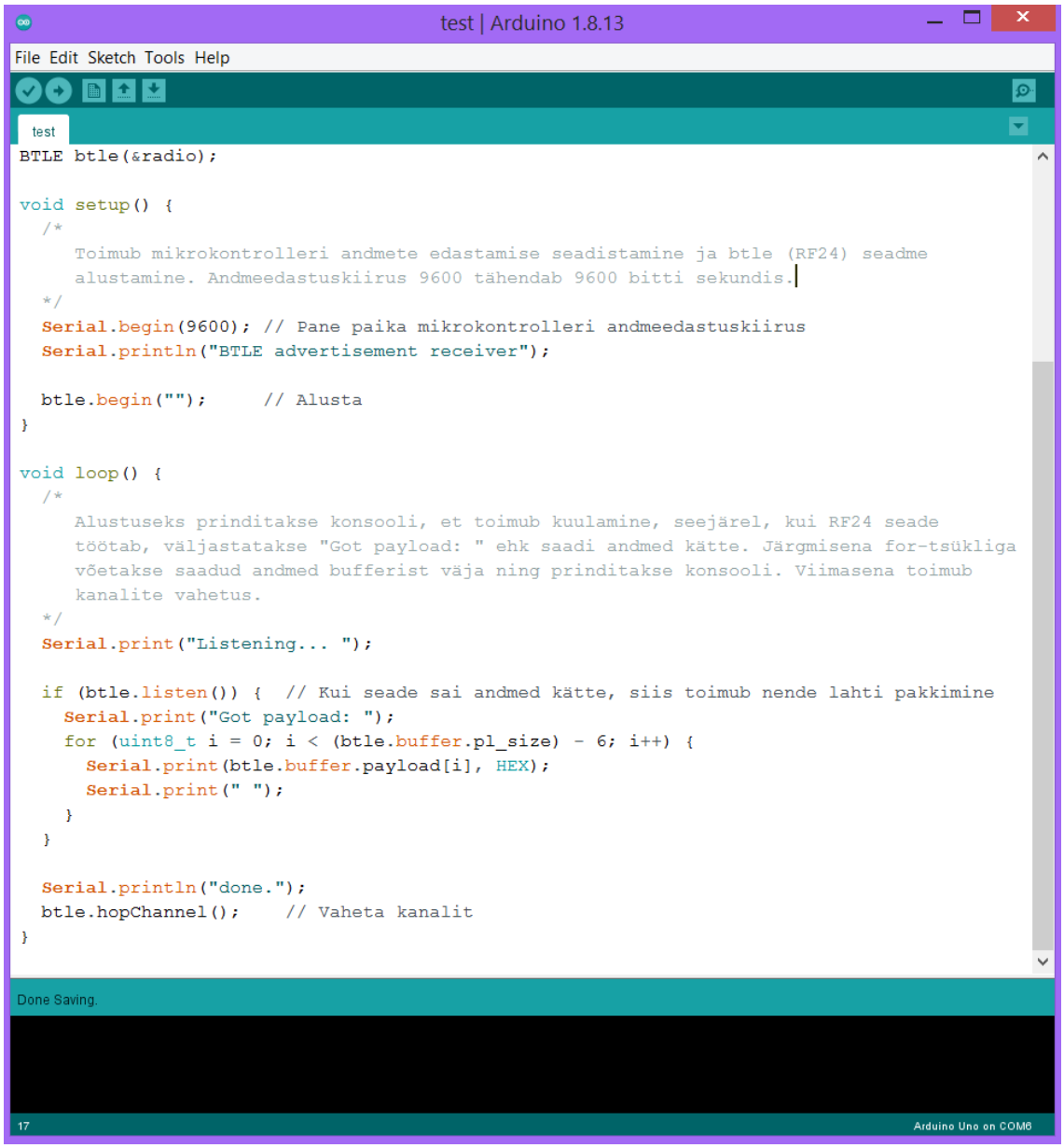
2.2 Arduino tarkvara

Arduino integreeritud programmeerimiskeskkond on tarkvaraplatvormist sõltumatu Javas kirjutatud rakendus, mis on tuletatud Processing-programmeerimiskeele ja *Wiring* projekti integreeritud programmeerimiskeskkondadest. Integreeritud programmeerimiskeskkond on disainitud nii, et seda saaksid kasutada tarkvaraarenduses vähekogenud inimesed. Keskkond (Joonis 4) sisaldab koodiredigeerijat, mille iseärasusteks on näiteks süntaksi esiletoomine ja sulgude kokkuviimine. Ka koodi kompileerimine ja mikrokontrolleri programmeerimine toimuvad ühe hiireklõpsuga. [3]

Arduino programmeerimiskeskkonnaga on kaasas C/C++ teek *Wiring*, mis muudab paljud tavalised sisend-väljundoperatsioonid palju lihtsamaks. Arduino programmid on kirjutatud C/C++ programmeerimiskeeles, kuid kasutajad peavad töötava programmi jaoks defineerima ainult kaks funktsiooni:

- setup() – funktsioon, mis töötab korra programmi alguses ning seadistab mikrokontrolleri parameetrid;
- loop() – funktsioon, mis kutsutakse korduvalt esile, kuni plaat välja lülitatakse. [3]

Joonisel 4 on välja toodud väikesemahulise programmi näide, mis kujutab endast *BLE* andmepakettide vastuvõtmist raadioside mooduliga. Esamlt deklareeritakse teegid käsuga „*#include*“, seejärel muutujad. Järgmisena ongi „*setup()*“, kus pannakse paika mikrokontrolleri parameetrid, ja „*loop()*“ funkstoonid. Programm töö käib nii kaua, kuni kasutaja selle lõpetab.



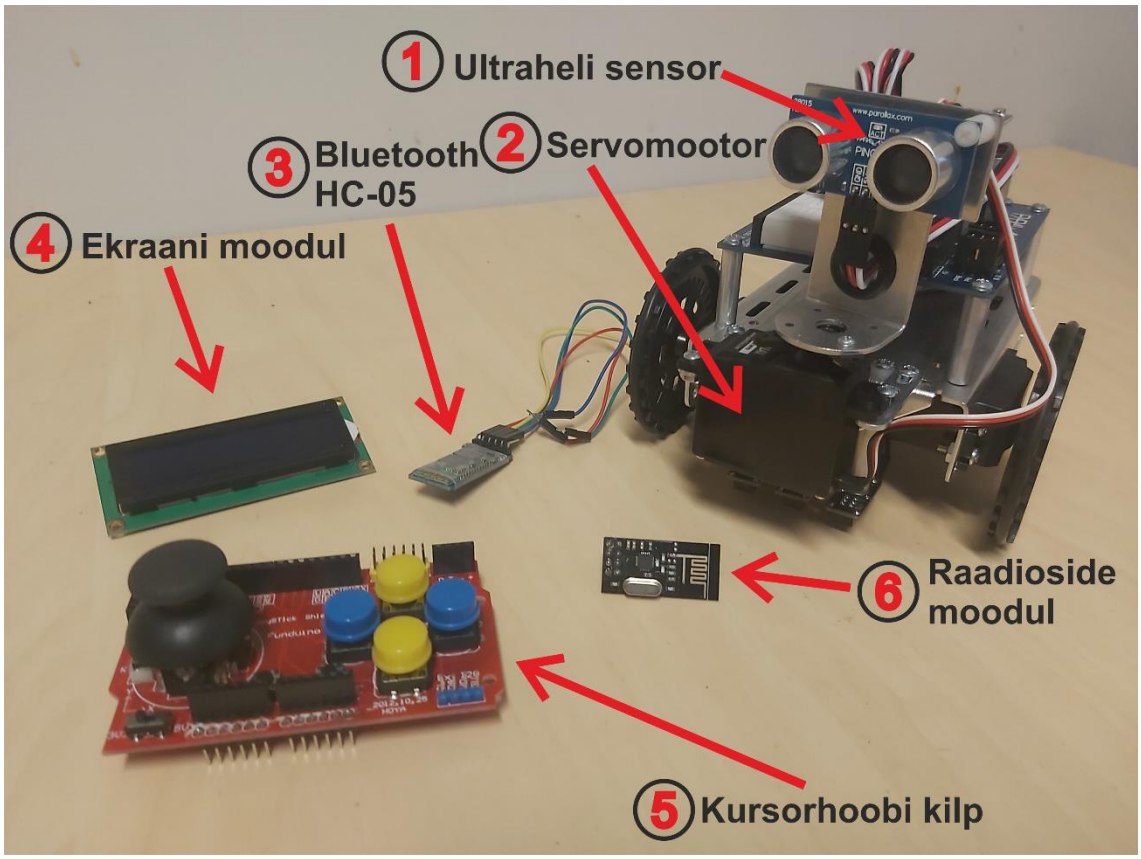
Joonis 4. Arduino kasutajaliides koos näidiskoodiga.

2.3 Kasutatud komponendid

Erinevates ülesannetes oli tudengitel võimalus kasutada mitmeid andureid. Need on elektroonilised komponendid, millega saab teha mõõtmisi, näiteks keskkonnast saadavad andmed muundatakse signaalideks, mida saab edastada arvutisse nende vaatamiseks või töötlemiseks. Lisaks kasutatakse mooduleid robotite loomiseks, millega moodustatakse erinevaid süsteeme, kus näiteks saab robotit panna sõitma, mõõtma või vedama.

Kuna mooduleid on palju erinevaid, siis nii on ka võimalusi. Arduino platvorm on seetõttu hea koht kust alustada elektroonika ja sardsüsteemide loomisega. Käesolevas lõputöös on

kasutusel näiteks ultraheli (Joonis 5, punkt 1), servomootori (Joonis 5, punkt 2), *BT* (Joonis 5, punkt 3), ekraani (Joonis 5, punkt 4) ja raadioside (Joonis 5, punkt 6) moodulid ning kursorhoobi kilp (Joonis 5, punkt 5). Lisaks oli kasutusel Arduino komplekt, mis kujutab endast Arduino Unot, kaablit, juhtmeid, resistore ja muid mooduleid



Joonis 5. Boe-Bot robot koos moodulitega.

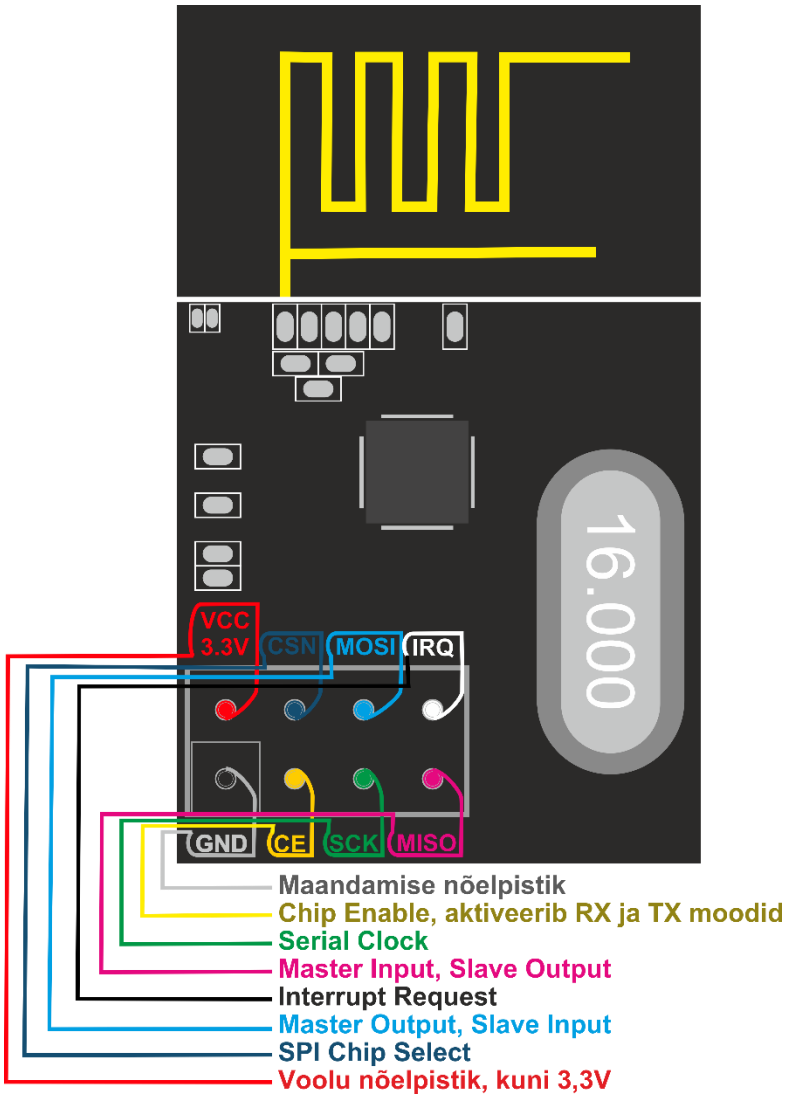
2.3.1 Servomootor

Servomootor (tuntud ka kui RC servomootor) (Joonis 6) on tagasisidega mootor, mis tähendab, et me saame signaali abil täpselt määrata mootori asendi. Tänu sisseehitatud ülekannetele, on selle jõud päris suur. Servomootoreid kasutakse tavaliselt mudelautode, -lennukite ja -laevade ning roboti liigeste juhtimiseks. 360-kraadi pöörlevaid servosid saab aga kasutada tavaliste mootoritena. Servomootoreid on lihtne Arduinoga ühendada, st ei pea olema mingit transistori või mootori draiverit ning Arduino IDE'ga on kaasas juba Servo.h teek, mis võimaldab koodi kirjutamist veelgi lihtsustada. [4]

Joonis 6. Parallax Continuous Rotation Servo. [5]

2.3.2 NRF24L01+ moodul

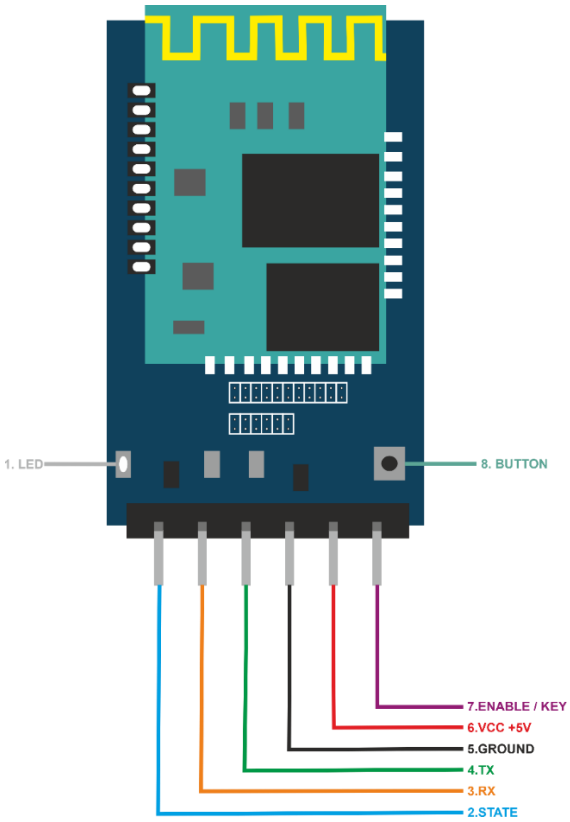
NRF24l01+ (edaspidi RF24) (Joonis 7) on sisseehitatud sageduse põhiriba protokolli mootoriga (*Enhanced ShockBurst* ™) ühe kiibiga 2,4 GHz transiiver, mis sobib ülimadala energiatarbega traadita rakenduste jaoks. RF24 on mõeldud töötamiseks ülemaailmses *ISM*-i sagedusalas sagedustel 2,400 - 2,4835GHz. RF24 abil raadiosüsteemi kujundamiseks vajatakse lihtsalt MCU-d ja mõnda välist passiivset komponenti. RF24-ja on võimalik juhtida ja konfigureerida sünkroonse järjestiksuhtluse liidesega (edaspidi *SPI*). *SPI* kaudu juurdepääsetav registrikaart sisaldab kõiki konfiguratsiooni registreid ja on juurdepääsetav kiibi kõigis töörežiimides (saatmise ja vastuvõtmise). [6]



Joonis 7. NRF24 moodul ja tiigid.

2.3.3 HC-05 Bluetooth moodul

HC-05 moodul (edaspidi HC-05) (Joonis 8) on lihtsasti kasutatav *BT* jadapordiprotokolliga moodul, mis on mõeldud traadita jadaühenduse seadistamiseks. HC-05-te saab kasutada *master* või *slave* konfiguratsioonis, mis on suurepärane lahendus traadita suhtlemiseks See jadapordi *BT* moodul on täielikult kvalifitseeritud *BT* versioon 2.0 ja *EDR* 3Mbps modulatsioon koos 2,4 GHz raadiosaatja ja põhiribaga (see on modulatsiooniga muutmata sagedusriba, mis hõivab üht või multipleksitud signaalide kogumit). [7]



Joonis 8. HC-05 BT moodul.

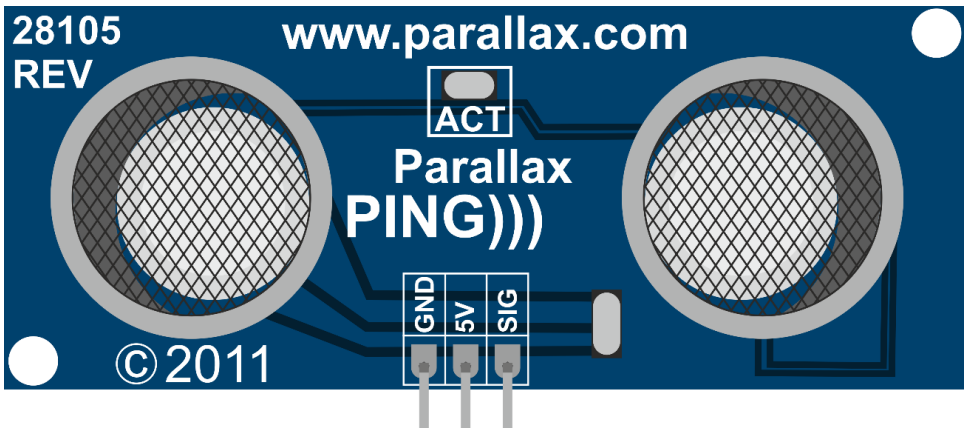
HC-05 on ülem-alluv moodul, mis vaikimisi on alluva režiimi peal. Mooduli rolli on võimalik konfigureerida ainult *AT* käskude abil, mis on Dennis Hayes'i poolt 1981 aastal loodud käskude keel, mis koosneb mitmetest lühikestest tekstistringidest. Kombineeritult moodustavad nad käsud erinevate tegevuste jaoks, näiteks ühenduse valimine, katkestamine ja ühenduse parameetrite muutmine. Olles alluva režiimis, on võimalik ainult ühendusi aktsepteerida, kuid mitte ise ühendusi teise seadmega luua. Ülema režiimis olles saab moodul nii ühendusi vastu võtta, kui ka luua. [8]

2.3.4 Ultraheli sensor

Parallax'i poolt valmistatud PING))) ultraheliandur (edaspidi ultraheli sensor) (Joonis 9) pakub kauguse mõõtmise meetodit. See andur sobib suurepäraselt igasuguste rakenduste jaoks, mis nõuavad mõõtmist liikuvate või seisvate objektide vahel.

Mikrokontrolleriga ühendamine on kiire, kus vaja läheb ainult ühte sisend- ja väljundviiku, mida kasutatakse ultraheli käivitamiseks ja seejärel kaja tagasipulsi "kuulamiseks". Andur mõõdab kaja tagasitulekuks vajalikku aega ja tagastab selle väärtuse mikrokontrollerile muutuva laiusega impulsina. [9]

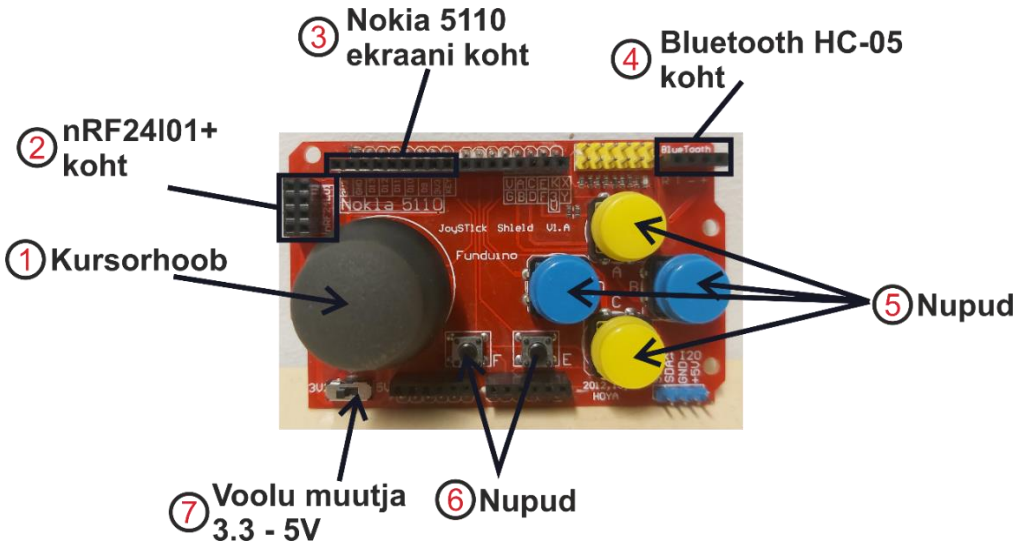
Andur suudab mõõta alates kolmest sentimeetrist kuni kolme meetrini. Lisaks töötab nii valges, kui ka pimedas keskkonnas.



Joonis 9. Parallax'i ultraheli sensor PING))). [9]

2.3.5 Kursorhoobi kilp

Kursorhoobi kilp (Joonis 10) võimaldab ühendamist Arduino Unoga, et kergendada erinevate moodulite eraldi sidumist, mis omakorda vähendab juhtmete arvu. Lisaks pakub moodulite RF24, Nokia 5510 ekraani, HC-05, kui vajaminevate viikude arv ei ületa vabade oma, ühendamist.



Joonis 10. Kursorhoobi kilp.

Kursorhoobi kilpi on võimalik ka ühendada otse Arduino Uno külge ilma, et oleks juhtmeid vaja (Joonis 11). Kilbil endal on all olemas vastavad viigud, mida saab kinnitada täpselt Uno külge.

Joonis 11. Kursorhoobi kilp kinnitatuna Arduino Uno külge.

3 Ülesanded

Lõputöö raames sai koostatud modifitseeritud Boe-Bot robotiplatvormile ülesandeid aine Erialatutvustus (IAS0001) raames. Igast ülesandest loodi kolm eri versiooni, mis jaotati rühmade vahel ära, ning nendest omakorda kolmeks vaheetapiks. Edusammud pandi kirja GitLab repositooriumisse, samasse kohta, kuhu postitati programmide koodid. Ülesannete püstitused olid tehtud kui *milestone*-d, mille lõpetamiseks pidid tudengid looma *issue*-sid ja need siduma *milestone*-dega. *Issue*-de lahendamisel sulgeti need ning sellega märgiti *milestone* ehk vaheetapp lõpetatuks.

Tudengid rühmadesse jaotatud ja ülesanded käes, pidid seejärel määrama vastutava, kes jälgis ja andis tööülesandeid tiimiliikmetele – see kestis kaks nädalat, mille lõpus tuli teha ja esitada kokkuvõtlik esitlus. Iga tiimiliige pidi selle korra läbi tegema, et saada vajalikud punktid/märked kätte. Ettekanded ei olnud pikad, 5 minutit rääkimist ja näitamist, kus maal omadega ollakse ning mis läinud halvasti või hästi. Samuti said rääkida, kuidas toimus ülesannete jaotamine ning kes millega tegeles.

3.1 *BLE* ühenduse loomine RF24 ja nutitelefoni vahel

Ülesandeks oli luua *BLE* (*Bluetooth Low Energy*, edaspidi *BLE*) kommunikeerimine telefoniga, mitte ühenduse, vaid andmepakettide saatmise-vastuvõtmise teel. Kasutusel oli RF24 raadiosidemoodul, mille kaudu pidi suutma saata nutitelefonile (Android või iPhone) pakette ning neid ka vastu võtma.

Probleem, millega tudengid silmitsi seisid, oli võimaluse leidmine, kuidas saata *BLE* pakette üle moodulile, mis tegelikkuses *BLE*-d ei toetagi. Selleks oli neil kasutada spetsiaalne teek (*library*) tänu millele oli võimalik saata ja vastu võtta *BLE* pakette. Kuidas täpsemalt see toimis, kirjutatakse lähemalt peatükis 3.1.1 BLE ühendus RF24 mooduliga.

Ülesande enda eesmärgiks oli juhtida robotit nutiseadmest, kasutades selleks "*nRF Connect for Mobile*" aplikatsiooni Nordic Semiconductor ASA poolt ning RF24 moodulit. Ülesanne ise jaotus kolmeks etapiks.

3.1.1 Ülesande etapid

Esimene ülesanne oli edastada telefoni nimi robotile, levitades selleks aplikatsioonis (Semi) koostatud paketti. Selleks on olemas eraldi vaheleht „*Advertiser*“, kus seda teha saab. Luues uue paketi levitamiseks peab valima õiged seadistused ning mida saata. Kui robot sai andmed kätte andis ta sellega märku lülitades LED tuled sisse või sumistiga.

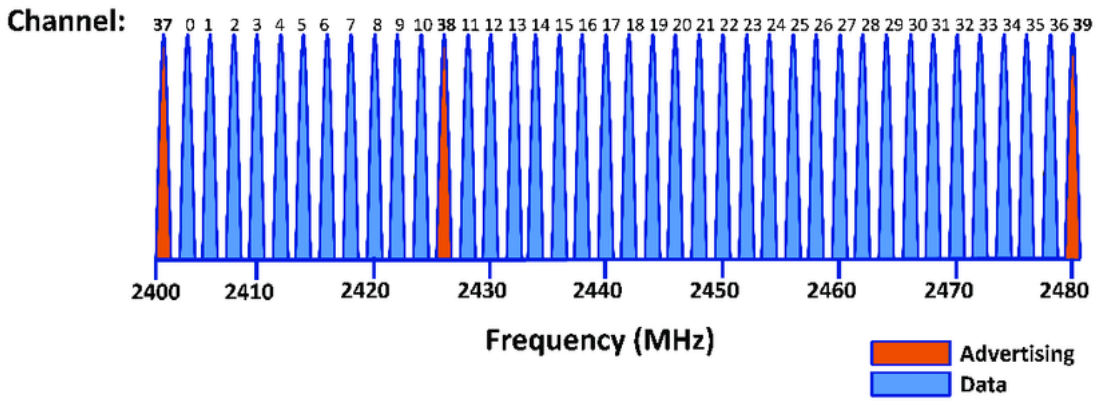
Teiseks ülesandeks oli suhtlemine nutitelefoni ja roboti vahel, kus mõlemad pooled pidid suutma andmeid vastu võtta, kui ka saata. Lisaks pidi olema võimalus saata robotile käske, millega saaks kontrollida servomootoreid.

Kolmandaks ülesandeks oli kauguse mõõtmine ultraheli sensoriga. Alustades mõõtmist, pidi robot sellest märku andma kasutades helisignaali või LED tulesid. Mõõtes ära kauguse, tuli see edastada telefoni, kus oli seda aplikatsioonis näha.

3.1.2 BLE ühenduse jäljendamine RF24 mooduliga

RF24 moodul on tegelikkuses mõeldud ainult raadiosidesignaalide edastamiseks ja vastuvõtmiseks 2.4GHz raadioribal, mis ei toeta *BT* ühendusi. Kuid sõltumata sellest suutis Dmitry Grinberg selle tööle saada ning samas kirjutada teegi, mis aitaks seda kasutada Arduino platvormil. Hiljem kirjutas Florian Echtler (kasutaja nimega „floe“) selle paremini ümber Arduino platvormile, mis on saadaval ka tema GitHub keskkonnas.

Grinberg lähtus asjaolust, et RF24 mooduli raadioside ning *BLE* ühendused (täpsemalt Lisa 3) on peaaegu sarnased, väljaarvatud mõnede eranditega. *BLE* puhul on vajalik esmalt ühenduse loomine kahe seadme vahel, mida aga RF24 luua ei saa, kuna eeldab 37-bitiste pakettide saatmist, mida aga ei toetata – maksimum suuruses paketid on 32-bitised. Sarnasused nende vahel on, et mõlemad töötavad 2,4GHz raadioribalaiusel - sagedusel, kus kanalid on üksteisest 1MHz kaugusel ning andmeedastuskiirus vähemalt 1 megabitt sekundis. Andmepakette saadetakse kolmel erineval raadiosidekanalitel 37, 38 ja 39 (Joonis 12), millede võrguside ribalaiused on vastavalt 2,402 GHz, 2,426 GHz ja 2,480 GHz. Kasutusel sellised kanalid, kuna 0 kuni 36 on mõeldud andmete edastamiseks. Vastav võrdlus *BLE* ja RF24 *BLE* versiooni vahel on välja toodud Tabel 1-s.



Joonis 12. BLE sageduse kanalid. [10]

Joonisel 12 on kanalite, kust toimub andmete edastamine, numbrid teistega võrreldes teistmoodi, mille põhjus seisneb sageduste õiges järjestuses. 0 kuni 36 (Joonis 12, *Channel*) on andmete (Joonis 12, *Data*) ja 37 kuni 39 teavitamise kanalid.

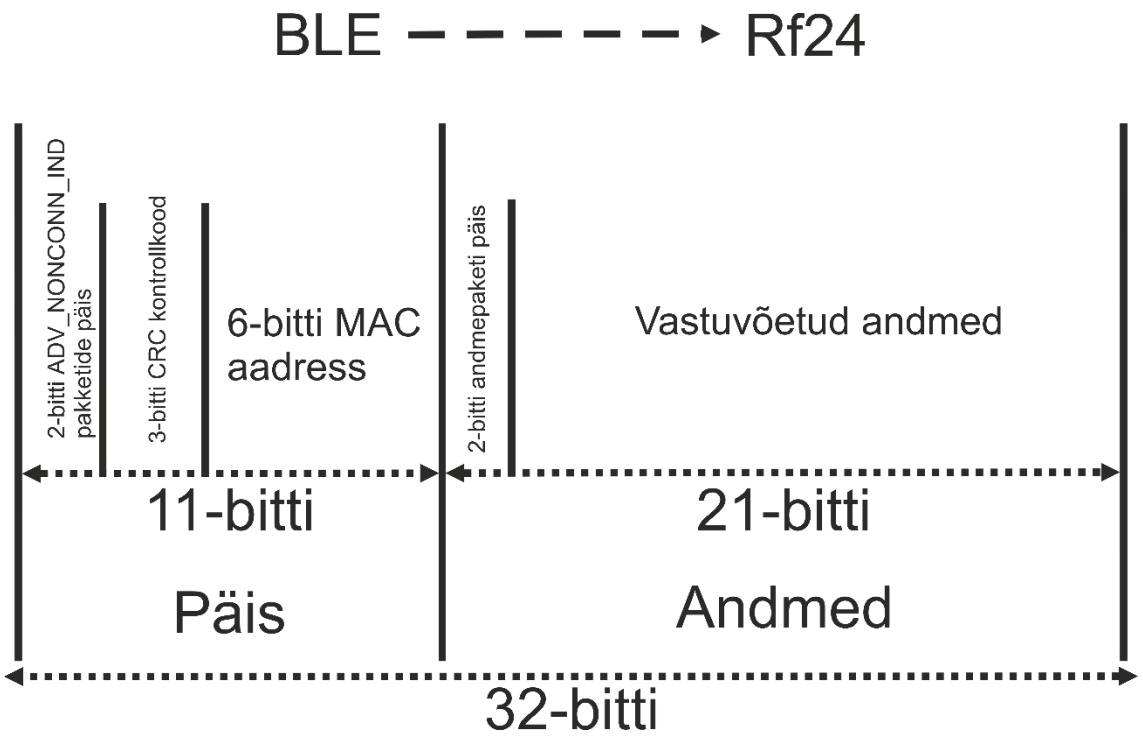
	BLE Ühendus	**BLE Ühendus RF24 mooduliga**
Seadmete ühendamine	Jah	Ei
Pakettide suurused	Kuni 265-baiti	Kuni 32-baiti
Kiirus	1Mbps	1Mbps
Kontrollkood	CRC	CRC ja võtipleegitus
Kanalid	2,400 – 2,480 GHz	2,402 GHz, 2,426 GHz ja 2,480 GHz

Tabel 1. BLE ja RF24 mooduliga ühenduste võrdlus.

Kuigi RF24 ei toeta *BLE* ühendust, leidis Grinberg siiski võimaluse seda võltsida viisil, et seade, mis toetab *BT* ühendust, suudab 32-bitiseid pakette vastu võtta. Esmalt tuleks pakettide kättesaamisel andmeid kontrollida, milleks on kasutusel *CRC* kontrollkood. Kuna *BLE* kasutab selle jaoks 24-baiti, mis on aga RF24 jaoks liiga suur, tuleb seega andmeid manuaalselt (mooduli sisse ehitatud kontrollkoodi funktsiooni ei kasutata siin) kontrollida, milleks on teegis olemas eraldi funktsioon. Selleks on *whitening* ehk võtipleegitus, mis kujutab endast andmete de- ning krüpteerimist kasutades *XOR* ehk välistava või šifferit.

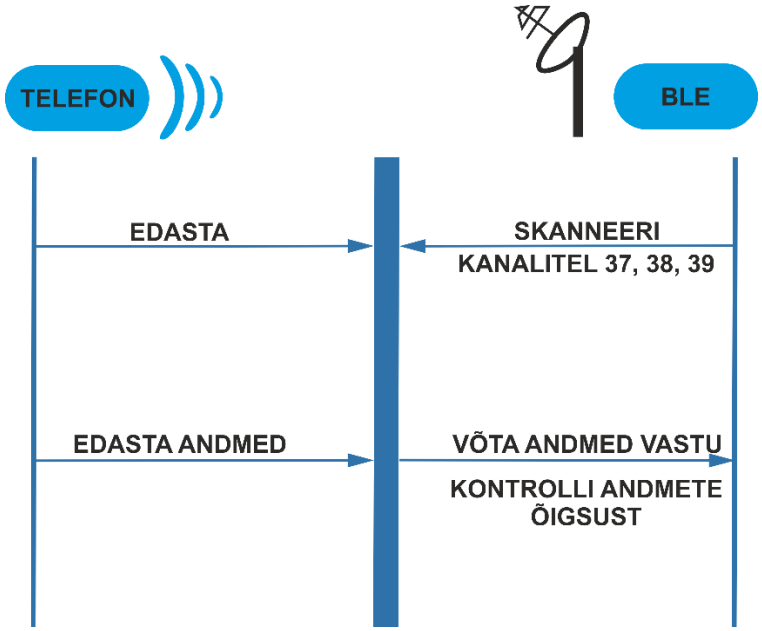
Iga vastuvõetud andmepaketi suurus on 32-bitti (Joonis 13), mis on piisav, et saata mõningaid üksikuid andmeid, nagu seadme nimi või muud soovitut. Joonisel 13 on ära näidatud, kuidas on andmepakett jaotunud. Kõige tähtsam osa aga paketist on esimesed 11-bitti, mis määravad ära kuhu saata ning et ikka õiged andmed jõuaksid kohale

(kontrollkoodiga kontrollimine). Ülejäänud 21-bitist on kasutada ainult 19-bitti andmete endi jaoks, millest 2-bitti on mõeldud päise jaoks.



Joonis 13. RF24 vastuvõetud andmepaketi näidis.

Andmepaketi päis on jaotatud kolmeks, kus esimesed kaks bitti on leviteavitamise protokolliandmeüksus (*advertising PDU*). *ADV_NONCONN_IND* tähendab siin, et tegemist on pakettidega, mis ei nõua seadmete omavahelist ühenduvust, see tähendab pakette saadetakse teadmata kes need kätte saab. Järgmised kolm bitti on mõeldud kontrollimiseks, kas saadetud andmed on õiged. Selleks on *BLTE* teegis olemas ekstra funktsioon, mis seda kontrollib. Kuigi RF24 endal on ka CRC kontrollkoodi võimalus, kuid seda kasutata, kuna kasutab liiga palju bitte. Viimased kuus bitti on seadme, kuhu saata, MAC aadress. Vaikimisi on selleks aadressiks määratud 0x8E89BED6 (mida kustutakse ka „*bed six*“ -na, viimased neli tähte aadressi lõpust), kuna võimalus, et saadakse kätte samasugusele aadressile saadetud pakett, on suhteliselt väike. Siin tuleb ka täheldada, et on määratud selline suvaline aadress ka põhjusel, sest *BLE* puhul ei toimi seadmete ühendamist (Joonis 14). Seega, kui mõlemad seadmed teavad juba määratud aadressi, oskavad ka pakette vastu võtta.



Joonis 14. BLE ühendus RF24 mooduli ja telefoni vahel.

Andmepaketi osas on esimesed kaks bitti andmete tüübi kohta. Juhul, kui tahetakse kaasa anda ka seadme nimi, kust andmed tulid, võtab see omakorda kaks bitti. Seega koos nimega jääb üle 17-bitti ning ilma 19-bitti andmeid, mida on võimalik saata ja vastu võtta.

3.1.3 Ülesande analüüs

Lõputööst ülesanne *BLE* ühenduse loomine RF24 ja nutitelefoni vahel kasutati 2020. aasta sügissemestri aine Erialatutvustus raames, kus kolm tudengi rühma said seda lahendada.

Ülesande plussideks oli RF24-ja tavapärasest teistsugune kasutamine, kuna ei ole mõeldud *BT* ühenduste jaoks, siis katsetamine ja tööle saamine oli huvitav. Lisaks erines teistest ülesannetest selle poolest, et töö sisaldas nutitelefoni kasutamist.

Miinusteks kujunes RF24, mis ei töötanud andmete vastuvõtmisel hästi. Nimelt probleem oli, et ühe andmepaketi vastuvõtmiseks kulus ligikaudu üks kuni viis minutit. Teiselt poolelt andmete saatmine nutitelefoni sujus kiirelt ning probleemideta. Täpset põhjust sellele ei leitud, kuid spekuleeriti, et tegu võis olla halva voolu regulatsiooniga RF24 ja roboti vahel. Katsetuste käigus leidsid tudengid ajutise lahenduse. Üheks oli RF24-ja näpuga hõõrumine ja teiseks tema kõigutamine liideses (Joonis 2, punkt 6).

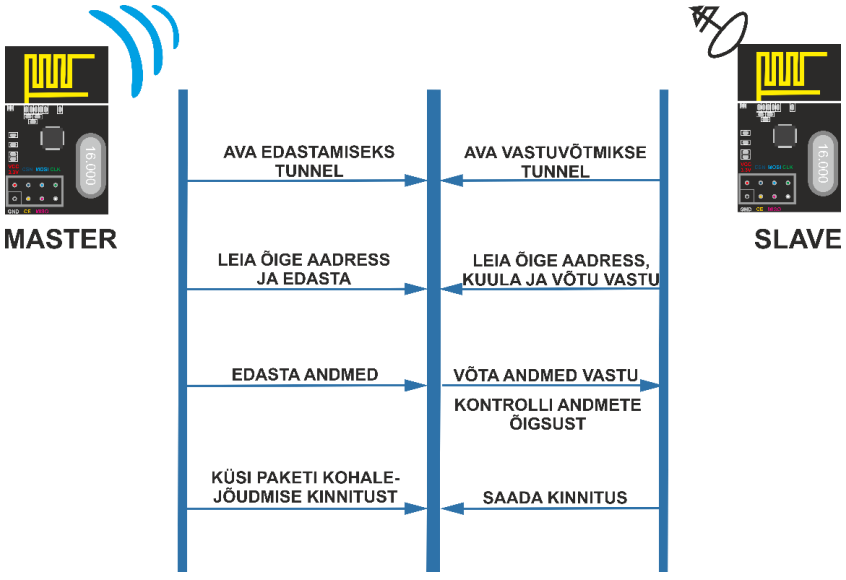
3.2 Roboti puldiga juhtimise ülesanne

Ülesandes on kasutada kursorhoobi kilpi (Joonis 10) (ing. kl. *joystick shield*), millel asetseb kursorhoob (Joonis 10, punkt 1) ning saab lisada teisi mooduleid külge ilma juhtmeteta. Ülesande eesmärgiks on tekitada raadioside roboti ja puldi vahel. Roboti juhtimiseks on tudengil olemas kursorhoobikilp, kuhu saab lisada RF24-ja ning selle kaudu saata käske robotile. Lisaks roboti juhtimiseks on kaks võimalust – kasutades nuppe või kursorhoopi.

Esiteks on vaja kursorhoobi kilp ära ühendada Arduino Uno platvormiga, mis on tehtud lihtsaks – kilp mahub täpselt Uno peale. Raadiosidemooduli lisamiseks on kilbil eraldi koht olemas, kuhu selle panna saab, samuti on robotil endal selle mooduli jaoks koht olemas. Siinpuhul tudengitel ei ole vaja mingit juhtmetega ühendusi tekitada.

Roboti juhtimiseks kursorhoobiga on vaja esmalt tudengitel aru saada, kuidas see töötab. Kursorhoop saab liikuda igas suunas – paremale, vasakule, tagasi, ette ning külgedele. Selle liikumist võib ette kujutada kui x- ja y-telge, mis vastavalt koordinaatidele leiab asukoha.

Teiseks tuleb tekitada kahe RF24 omavaheline raadioside (Joonis 15), kus oleks võimalik saata andmeid üksteisele roboti juhtimiseks. Selleks tuleks esmalt valida kumb moodulitest oleks *master* ja *slave*. Siinpuhul saadab *master* ainult käske ning *slave* võtab vastu.



Joonis 15. Andmeside kahe RF24 mooduli vahel.

3.2.1 Ülesande etapid

Ülesanne jaguneb kolmeks etapiks, kus esimeseks oleks saavutada RF24 moodulite vaheline ühendus. Nagu eelnevalt mainitud peab paika panema, milline oleks *master* ning *slave*. Üks RF24-jast läheb roboti külge ning teine puldi, kust hakatakse ka käske saatma ehk juhtima. Eduka ühenduse loomisel panna robotil LED tuli põlema või anda helisignaaliga märku.

Teiseks ülesandeks on roboti juhtimine edasi-tagasi ning paremale-vasakule kasutades selleks puldil olevat kursorhoopi (Joonis 10, punkt 1). Eelnevalt sai ka mainitud, et see töötab sarnaselt x- ja y-koordinaattelejestikule, kus suuna määramiseks on vaja teada vastavas ulatuses x- ja y-koordinaate. Järgmiseks oleks roboti juhtimine labürindis.

Kolmandaks ülesandeks on sõita robotiga labürint ühe korra läbi, kus teisel korral peab ta juba ise hakkama saama. Iseeneslik raja läbimiseks on mitu varianti. Esimene saaks võtta aega, kui palju läheb ühe kindla suuna sõitmiseks, näiteks sõidab otse kolm sekundit, seejärel üks sekund pöörab paremale ja nii edasi. Teisena oleks võimalik igale käsule panna kindel ajaviide, näiteks vahemikus 100 kuni 500 millisekundit. Nii ei tohiks tekkida probleeme, kuna ajavahemikud on üsna väikesed. Raja läbimisel salvestab robot samaaegselt käske, et hiljem saaks iseseisvalt raja läbida.

3.3 Roboti juhtimine *BT* ühenduse kaudu

Ülesandeks oli roboti juhtimine *BT* ühenduse kaudu. Kasutamiseks on HC-05 (Joonis 8), mis võimaldab ühenduse loomist kahe seadme vahel kasutades selleks *BT* ühendust. Lisaks on võimalus kasutada sarnast moodulit HC-06, mis on küll uuem, kuid vähemate funktsioonidega (võrdlus peatükis Lisa 4). Roboti juhtimine toimub läbi nutitelefoni. Google Play poest, kui ka Apple Appstore-ist on saadaval erinevad robotite juhtimiseks aplikatsioonid, läbi mille on võimalik luua robotiga ühendus ning käske saata. Selle ülesande eesmärgiks on õpetada tudengitele, kuidas toimib *BT* side seadmete ja moodulite vahel.

HC-05-l on 6 erinevat viiku ja lisaks LED lamp, mis näitab, kas moodul töötab või mitte, ning ka hetke töörežiimi. Veel on nupp (Joonis 8, punkt 8), millega saab lülituda ümber

kahe erineva režiimi vahel – käsu- ja andmerežiimi. Moodul töötab pingel 3,6V kuni 6V, kuna Arduino on ainult 3,3V ja 5V viigud, siis tuleb ühendada 5V-sse pinni.

Roboti juhtimiseks on võimalik alla laadida, kas Google Play või Apple Appstore poest, rakenduse, mis võimaldaks *BT* ühenduse kaudu käske saata. Selliseid rakendusi on mõlemas poes päris palju, millel on veel lisaks igasugused muud funktsiooni lisaks kaasas, näiteks tulede põlema panemine, helide tegemine ja muud, mis kõik käib käskude andmisel robotile. Siinpuhul ei ole ka probleemiks vanem *BT* versioon 2.0. Sellega on võimalik ühildada ka kõige uuemad *BT* versiooni toetavad seadmed, ainuke erinevus, mis sisse tuleb, on kiirus ja energiakasutus, kuid need selle ülesande puhul rolli ei mängi.

3.3.1 Ülesande etapid

Esimeseks ülesande variandiks on tudengitel luua ühendus nutitelefoni ja roboti vahel. HC-05-l on kaks erinevat režiimi – andme- ja käsurežiim. Vaikimisi on seade alati „*AT Command Mode*“ ehk andmerežiimis, kust saab andmeid saata. Ühenduse loomiseks teise seadmega on vaja üle minna käsurežiimi. Selleks peab esmalt vajutama moodulil olevat nuppu (Joonis 8, punkt 8), seda all hoidma kaks sekundit ning siis voolu andma (ühendades juhtme voolusisendisse). Kui tuli hakkab vilkuma kahe sekundiliste intervallidega, siis moodul on käsurežiimis. Seejärel soovitud seadmega ühendus luua käsuga "AT+PAIR=XXXX,XX,XXXXXX,20" (ilma jutumärkideta), kus „X“ tähistab MAC aadressi ühte karakterit. „AT+PAIR“ on käsk, millega teavitatakse seadet, et on soov ühendust luua seadmega. Järgmise osana tuleb üles otsida seadme *BT* aadress ning lisada see käsule. Näiteks kui *BT* aadress on „A5:AB:CD:ED:E5:K7“, siis tuleb see kirjutada selliselt: „A5AB,CD,EDE5K7“. Viimane komaga eraldatud koht (…,20) on ajalimiit sekundites, mil ta proovib luua ühendust. Kui ühendus loodud, vastab ta tulemusega "OK“ Arduino *Serial Monitoris* (edaspidi terminal). Juhul, kui mingit tulemust ei ilmu, tähendab see, et ühendus ei ole loodud ning peab uuesti proovima.

Kui ühendus seadme ja *BT* mooduli vahel on saavutatud, on järgmiseks etapiks saata käske robotile. Nagu eelnevalt mainitud on nutitelefonidele saadaval erinevaid rakendusi, mis suudavad *BT* ühenduse kaudu saata käske. Samuti siin ülesandes võib kasutada omal valikul aplikatsiooni, peaasi on see, et see suudaks saata käske, mida moodul ära tunneb. Selleks, et käsku ära tunda, peab selle esmalt välja printima terminalis. Tavaliselt on

käsud, kas numbritena või üksikute tähtedena. Järgmisena on tudengitel vaja leida viis, mis käsuga robotit liikuma panna.

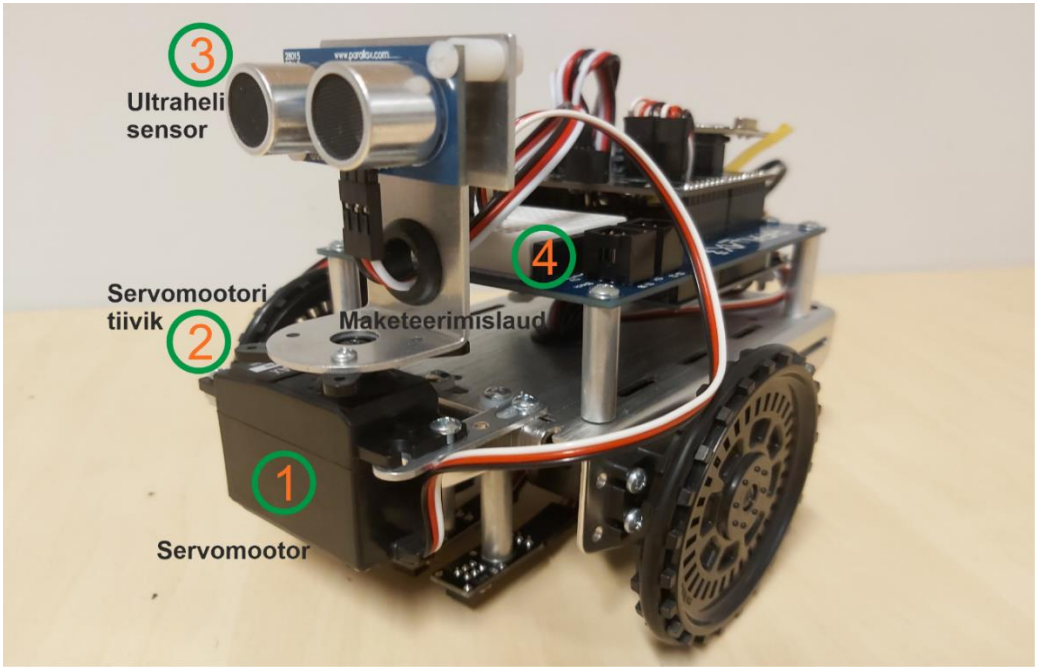
Kolmandaks etapiks on luua *BT* ühendus kahe mooduli vahel (nutiseadet enam ei kasutata). Siin tulebki sisse funktsioon, mida HC-05 toetab, kuid mitte HC-06, milleks on kahe seadme vaheline kommunikeerimine. Kuna HC-06 saab ainult andmeid vastu võtta, kuid mitte saata, siis selles ülesandes seda kasutada ei saa. Eesmärgiks on luua suhtlus kahe mooduli vahel ning ühest saata käske teise. Kui käsk on edukalt roboti poolt kätte saadud, vastab juhile, et käsk vastu võetud (roheline tuli põleb) ning hakkab seda täitma.

3.4 Labürindi läbimine liigutatava ultraheli sensoriga

Ülesande eesmärgiks on luua robotile ultraheli sensori süsteem, millega suudetakse takistustest mööda põigata. Parallaxil on selleks otstarbeks olemas lahendus, „*PING))) Ultrasonic Sensor + Mounting Bracket*“ [11], mis sisaldab servomootorit, kuhu otsa on kinnitatud ultrahelisensor. Robot hakkab sõitma ning takistuseni jõudes jääb robot seisma ning hakkab ultrahelisensoriga otsima, kumbalt poolelt saaks mööduda ning kas see üldse on võimalik.

Ülesandes on kasutusel ultraheli sensor ning servomootor (Joonis 6 ja Joonis 16, punkt 1). Lisa servomootori paigutamine võib osutuda keeruliseks. Põhjus selles, et trükkplaadil on juba HC-05-e jaoks koht olemas, kuid mitte lisa servomootori jaoks. Kuigi koht on olemas HC-05 jaoks, on see 4-kohaline, kuid vaja läheb ainult kolme viiku, mistõttu peab teadma täpselt kuhu sisendisse mis väljund käib. Täiendavalt peab hakkama uurima, milliste viikudega saaks ühendada servomootorit, kus on vaja esmalt teada, millised tiigud üldse sobivad ning kas need on vabad.

Servomootor ja HC-05 paigas, peab hakkama mõtlema, kuidas sellega liikuda, et tuvastada vabat teekonda, kas paremalt või vasakult. HC-05-e nihutamiseks on servomootoril tiivik (Joonis 16, punkt 2), mida saab liigutada 360-ne kraadi ulatuses.



Joonis 16. Boe-Bot robot esiotsast koos ultraheli sensori ja servomootoriga.

3.4.1 Ülesande etapid

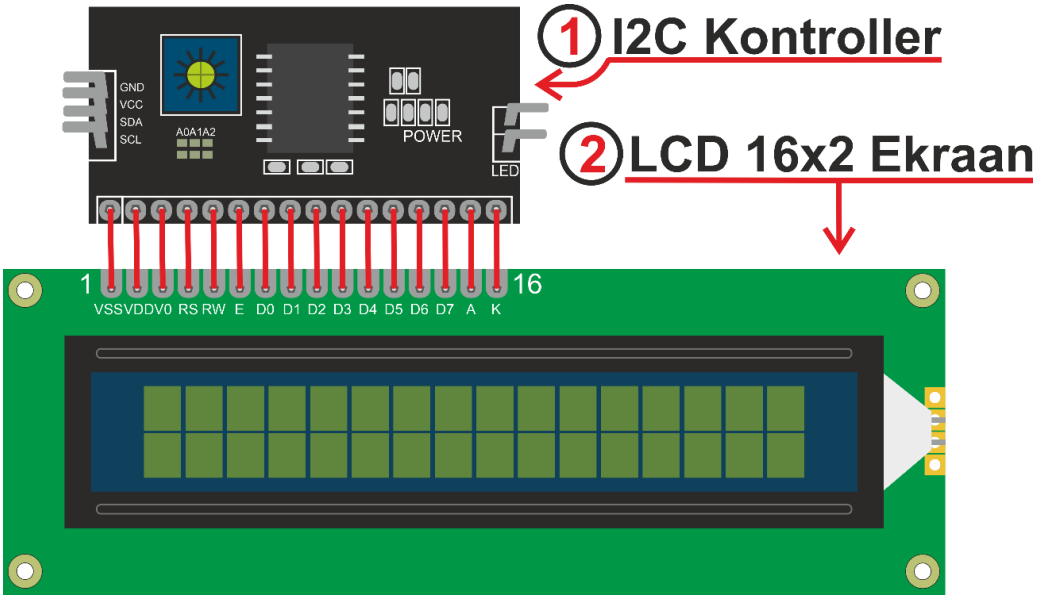
Esimeseks ülesandeks tuleks ühendada telefon HC-05-ga ning alla laadida vastav rakendus, millega oleks võimalik robotit juhtida. Siin peab tähele panema, et erinevad rakendused võivad käske saata robotile isemoodi, näiteks otse sõites võidakse saata tähemärke või hoopis numbreid. *BT* ühendus loodud, peab robotit suutma juhtida.

Teiseks ülesandeks on lisaks HC-05-le kasutusele võtta ka ultrahelisensori moodul. Juhtides peab robot jälgima ette tulevaid takistusi, mille puhul jääb seisma ning annab sellest märku, kas punase LED tulega või helisignaaliga. Järgmiseks peab robot iseennast viima asendisse, kus takistus ei jää tee peale ning sealt saab teda edasi sõidutada.

Kolmandas ülesandes peab robot suutma ise sõita otse ning takistuste ette tulemisel nendest põikama. Sarnane teise ülesandega, kuid antakse ainult üks käik, mille järel hakkab robot sõitma seni kuni tuleb stopp käsk, mis pannakse ise paika. Näiteks saates paremale liikumise käsu, pöörab robot end esmalt õiges suunas ning seejärel hakkab liikuma, samas ka takistustest põikama.

3.5 Raja kaardistamine ekraanile

Esmalt tuleks saada robot sõitma, kas raadioside või *BT* ühenduse kaudu (erinevad ülesannete variandid) ning seejärel teekonna joonestamine ekraanile. Kasutatakse Arduino komplektist saadaval LCD 16x2 I2C (Joonis 17) ekraani moodulit.



Joonis 17. Ekraani moodul LCD 16x2 (1) koos I2C kontrolleriga (2).

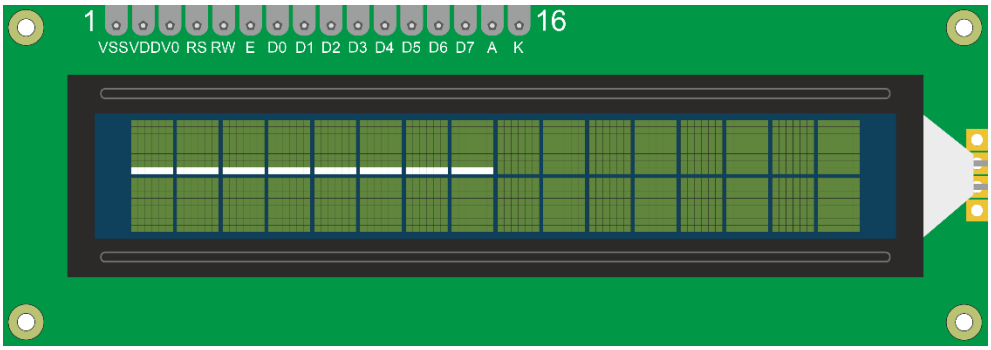
Ülesandes kasutatakse LCD 16x2 ekraani moodulit koos I2C kontrolleriga (Joonis 17, punkt 2), mis aitab vähendada ühenduste arvu. Samuti pakub moodulite kontrollimist kahejuhtmelise ja -suunalise I2C-siini seeriakella (SCL) ja jadaandmete (SDA) kaudu. Tänu sellele on vaja ainult nelja viiku ühendada – maandamise, voolu, seeriakella ja jadaandmete. Lisaks ei pea muretsema vabade viikude pärast, kuna nii SCL kui ka SDA on vabad.

Tuleb märkida, et esmalt ekraanile kirjutamiseks või joonistamiseks on vaja teada algusaadressi, mis tuleb leida *lcd scanneri* funktsiooni abil, milleks on vaja kasutada „*Wire*“ teeki. Igal seadmel, mida saab MCU külge ühendada, on aadress. Konkreetse seadmega suhtlemiseks peame teadma selle mooduli aadressi. Selleks ongi meil „*Wire*“ teek, mis sisaldab funktsioone, mille abiga saab ekraani algusaadressi üles leida.

3.5.1 Ülesande etapid

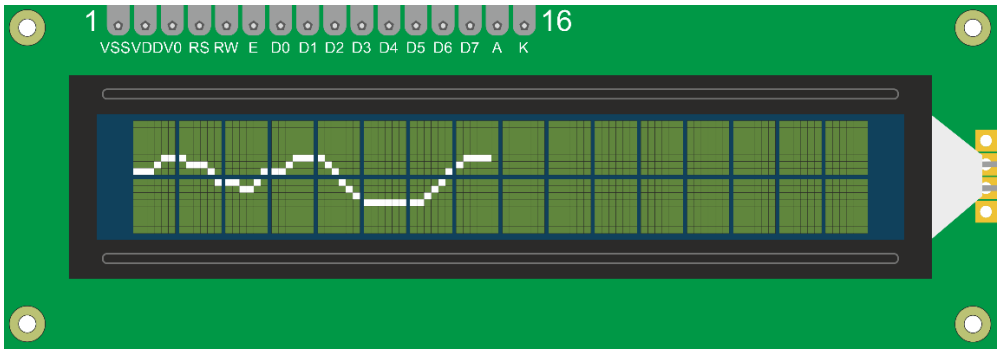
Ülesanne on jaotatud kaheks versiooniks, kus esimeses peab robotit juhtima RF24 mooduli abil ning teises *BT* ühenduse kaudu. Mõlemad versioonid jaotuvad veel alamülesanneteks. Siin tuleb kasutada ka eraldi teeki „*LiquidCrystal-I2C-library*“ [12].

Esimeses ülesandes peab roboti ära ühendama RF24-ga ja ekraani mooduliga. Roboti juhtimine toimub läbi Arduino Uno, mis on ühendatud RF24-ga. Ekraanile peab suutma joonistada ainult otse läbitud rada (Joonis 18). Nagu ekraani mooduli nimestki, on see jaotatud 16 korda 2 segmendiks (Joonis 17, punkt 2), kust iga on 8 korda 5 (pikkus korda laius) pikslit, kokku 40. Nende täitmine toimub 5-kohaliste kahendarvude liitmisel kahend astmetega. Põhjus selles, et ühtede ja nullide lugemine toimub paremalt vasakule. Näiteks esimese piksli täitmiseks on esmalt kahendarv 00000_2 vaja liitmis bititehe 2^4-ga (ehk 16-ga) ning tulemuseks saame 10000_2.



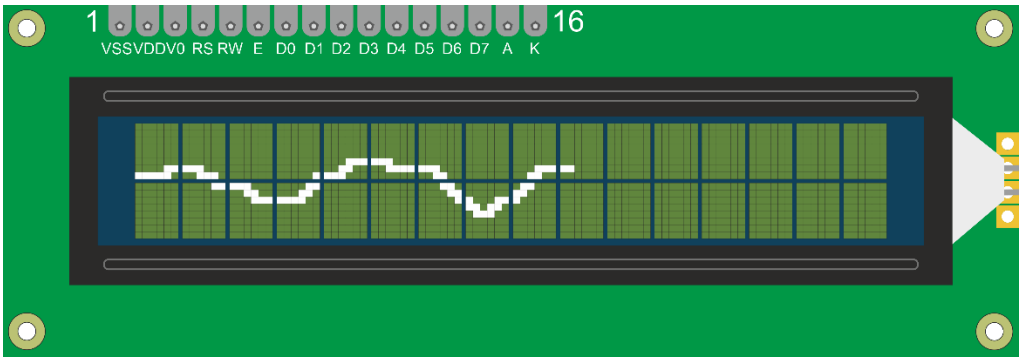
Joonis 18. Roboti otse liikumise trajektoor.

Teiseks etapiks on vaja joonistada roboti teekond ekraanile (Joonis 19). Robot hakkab sõitma, kus samas toimub joonestamine ekraanile. Eelnevalt mainitud on ekraan jaotatud segmentideks ja need omakorda piksliteks, mis tõttu on joonestamine ekraanile piiratud ning tuleb arvestada roboti teekonnaga.



Joonis 19. Roboti liikumise trajektoor joonistatud ekraanile.

Kolmas etapp on eelnevaga sarnane, kuid siin raja joonistamisel tuleb kinni pidada ekraani piiridest. Kui robot peaks ületama kindlat piiri, siis robot jääb seisma ning põlema läheb punane LED tuli. Samuti peab tekkima pidev joon (Joonis 20).



Joonis 20. Roboti liikumise pidev trajektoor ekraanil.

4 Kokkuvõte

Lõputöö eesmärgiks oli luua aine Erialatutvustus (IAS0001) raames ülesannete komplekt, mida tudengid saavad hakata rühmatöö projektina lahendama. Komplekt koosneb ülesannetest, mis on jaotatud omakorda etappideks, milleks on igaühe jaoks aega kaks nädalat.

Esmalt toimus ülesannete välja mõtlemine, mis aga kujunes arvatust keerulisemaks. Põhjuseid oli palju, millest üks ainult kindlate moodulite saadavus, mille peal ülesandeid koostada on aega nõudev. Ülesanded peatükkides 3.2 ja 3.4 on koostatud moodulite baasil, mis ei olnud tol hetkel saadaval. Ülesanded, mis olid koostatud saadaval moodulitega, võeti aines kasutusele, teised alles järgneval aastal.

Teiseks oli ülesannete etappideks jaotamine. Põhimõte seisnes õpetamisest tudengitele, et tehes projekti on parem jaotada terve ülesanne osadeks, kui kõike korraga teha. Lisaks abiks oli neil internetist sarnaste projektide otsimine, mille järgi teha, kuid tudengid pidid nendest aru saama ning vajadusel seletama, kuidas kood toimib, juhul kui taheti kasutada neid enda töödes.

Kolmandaks etapiks oli ülesannetele koodi kirjutamine, mis postitati GitHub keskkonda üles, mille juurde käis lisaks roboti ehitamine ning sellele vajalike moodulite ühendamine ja nendega katsetamine.

Kokkuvõttes oli ülesannete väljamõtlemine üsna keeruline ning aega nõudev töö, eriti katsetamine. Alguses ei osanud ka mõelda kõikide probleemide peale, mis võiksid esineda, näiteks moodulite või robotil vabade viikude puudumine. Kuid see omakorda õpetas, et alati tuleb ette mõelda ning kommunikeerida inimestega, kes on rohkem tuttavad robotite teemadega, segaduste ja aja kulutamise vähendamiseks.

Kasutatud kirjandus

[1] „What is an Arduino?,“ [Võrgumaterjal]. Available: https://learn.sparkfun.com/tutorials/what-is-an-arduino. [Kasutatud 17 12 2020].

[2] „ARDUINO UNO REV3,“ [Võrgumaterjal]. Available: https://store.arduino.cc/arduino-uno-rev3. [Kasutatud 17 12 2020].

[3] „Arduino,“ [Võrgumaterjal]. Available: https://github.com/arduino/Arduino. [Kasutatud 28 12 2020].

[4] „Arduino – Projekt 10: servomootorid,“ [Võrgumaterjal]. Available: https://www.metshein.com/unit/arduino-projekt-10-servomootorid/. [Kasutatud 28 12 2020].

[5] „Parallax Continuous Rotation Servo,“ [Võrgumaterjal]. Available: https://www.parallax.com/product/parallax-continuous-rotation-servo/. [Kasutatud 02 01 2021].

[6] „nRF24L01+,“ [Võrgumaterjal]. Available: https://www.digikey.com/catalog/en/partgroup/nrf24l01/44008. [Kasutatud 23 12 2020].

[7] „Bluetooth Module HC-05,“ [Võrgumaterjal]. Available: https://wiki.eprolabs.com/index.php?title=Bluetooth_Module_HC-05. [Kasutatud 23 12 2020].

[8] J. mega-das, „Bluetooth AT Commands Settings (HC05 HC06),“ 10 08 2020. [Võrgumaterjal]. Available: https://create.arduino.cc/projecthub/mega-das/bluetooth-at-commands-settings-hc05-hc06-305257. [Kasutatud 23 12 2020].

[9] „PING))) Ultrasonic Distance Sensor,“ [Võrgumaterjal]. Available: https://www.parallax.com/product/ping-ultrasonic-distance-sensor/. [Kasutatud 25 12 2020].

[10] J. Tosi, F. Taffoni, M. Santacatterina ja R. Sannino, „Performance Evaluation of Bluetooth Low Energy: A Systematic Review,“ 2007. [Võrgumaterjal]. Available: https://www.researchgate.net/figure/BLE-frequency-channels-It-can-be-noticed-that-channels-from-0-36-are-assigned-to-data_fig3_321800210. [Kasutatud 22 12 2020].

[11] „PING))) Ultrasonic Sensor + Mounting Bracket,“ [Võrgumaterjal]. Available: https://www.parallax.com/product/ping-ultrasonic-sensor-mounting-bracket/. [Kasutatud 04 01 2021].

[12] F. d. Brabander ja J. P. S. G. Silva, „Arduino-LiquidCrystal-I2C-library,“ [Võrgumaterjal]. Available: https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library. [Kasutatud 03 01 2020].

[13] D. Grinberg, „Faking Bluetooth LE,“ [Võrgumaterjal]. Available: http://dmitry.gr/index.php?r=05.Projects&proj=11.%20Bluetooth%20LE%20fakery. [Kasutatud 25 11 2020].

[14] „Bluetooth low energy,“ [Võrgumaterjal]. Available: https://et.wikipedia.org/wiki/Bluetooth_low_energy. [Kasutatud 23 12 2020].

[15] F. Echtler, „BTLE,“ [Võrgumaterjal]. Available: https://github.com/floe/BTLE. [Kasutatud 25 11 2020].

[16] „Bluetooth® Low Energy Packet Types,“ [Võrgumaterjal]. Available: https://microchipdeveloper.com/wireless:ble-link-layer-packet-types. [Kasutatud 20 12 2020].

[17] „HC-05 - Bluetooth Module,“ [Võrgumaterjal]. Available: https://components101.com/wireless/hc-05-bluetooth-module. [Kasutatud 29 11 2020].

[18] M. Currey, „HC-05 and HC-06 zs-040 Bluetooth modules,“ [Võrgumaterjal]. Available: http://www.martyncurrey.com/hc-05-and-hc-06-zs-040-bluetooth-modules-first-look/. [Kasutatud 11 12 2020].

[19] „AT Command Set,“ [Võrgumaterjal]. Available: https://www.techopedia.com/definition/575/at-command-set. [Kasutatud 23 12 2020].

[20] A. Joseph, „Interface I2C 16x2 LCD with Arduino Uno (Just 4 wires),“ [Võrgumaterjal]. Available: https://create.arduino.cc/projecthub/akshayjoseph666/interface-i2c-16x2-lcd-with-arduino-uno-just-4-wires-273b24. [Kasutatud 25 12 2020].

[21] „Arduino For Beginners,“ [Võrgumaterjal]. Available: https://www.makerspaces.com/arduino-uno-tutorial-beginners/. [Kasutatud 20 12 2020].

Lisa 1 – Lihtlitsents

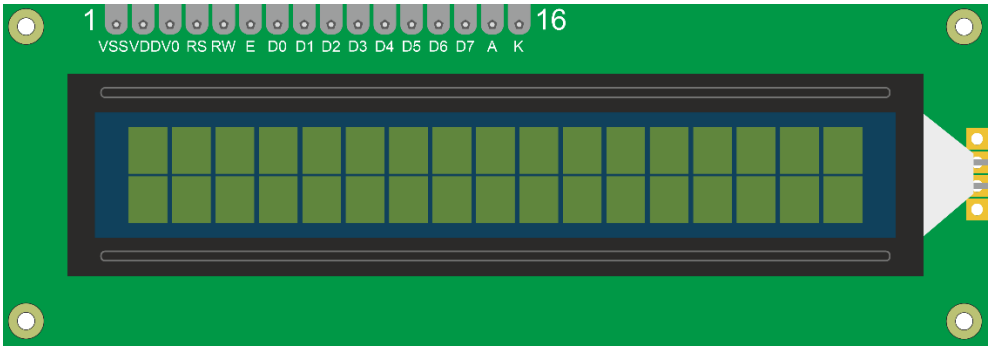
Mina, Sander Tsõbulski

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Projektiülesannete väljatöötamine mobiilsele Boe-Bot robotiplatvormile“, mille juhendaja on Priit Ruberg
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

05.01.2020

Lisa 2 – Olemasolevate ülesannete täiendus ekraani mooduliga

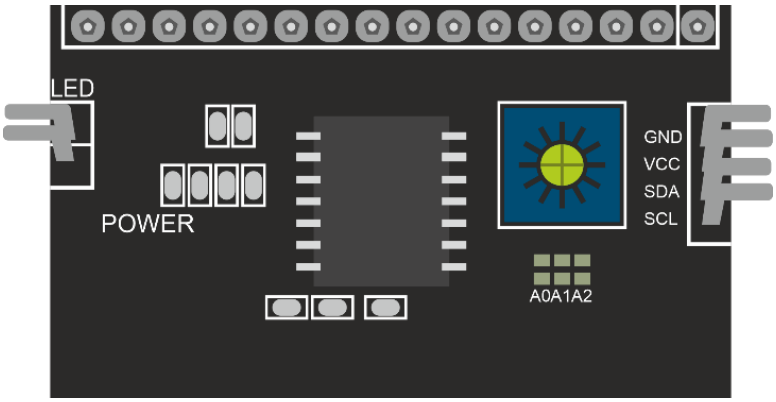
Kuna tänavu aasta (2020) aine Erialatutvustus (IAS0001) raames ei olnud kasutusel ekraani moodulit, millega oleks saanud tudengitele ülesannetele veel lisa tegevusi juurde mõelda, mis oleks aitanud neid programmide loomisel ja silumisel. Üheks selleks on ekraani moodul LCD 16x2 I2C (Joonis 21).



Joonis 21. LCD 16x2 I2C ekraani moodul.

Üheks kasutusalaks oleks ülesannete täiendamisel, kus robot saaks saata sõnumeid või veateateid sõitmise ajal. Näiteks oli üks ülesanne, kus kaks robotit pidid sõitma ühe raja peal ning üksteise kommunikeeruma, et teisele mitte otsa sõita. Seal oleks saanud saata ekraani peale, kas tee on vaba või sõidab hetkel seal keegi.

Teiseks saaks lasta ekraanil erinevaid kujukesi. Näiteks ühes ülesandes, kus tuli lugeda sisse noote, saaks samuti neid näha ka ekraani peal ja tuvastada, mis noodiga tegu. Teine näide oleks roboti sõitmise või labürindi läbimisel näidata suunda noolekestena.



Joonis 22. LCD ekraani kontrolleri moodul.

Lisaks on võimalus ühendada ekraani moodulile kontroller ICM1602 (Joonis 22), millega on võimalik vajaminevate viikude arvu vähendada neljani. Järgi jäävad maandamise, voolu, *SDA* ja *SCL* viigud, mille puhul kaks viimast on kontrollplaadil kasutamata, seega ei teki probleemi, et viikude vabadest kohtadest oleks puudus.

Lisa 3 – Bluetooth mooduli HC-05 viikude tabel

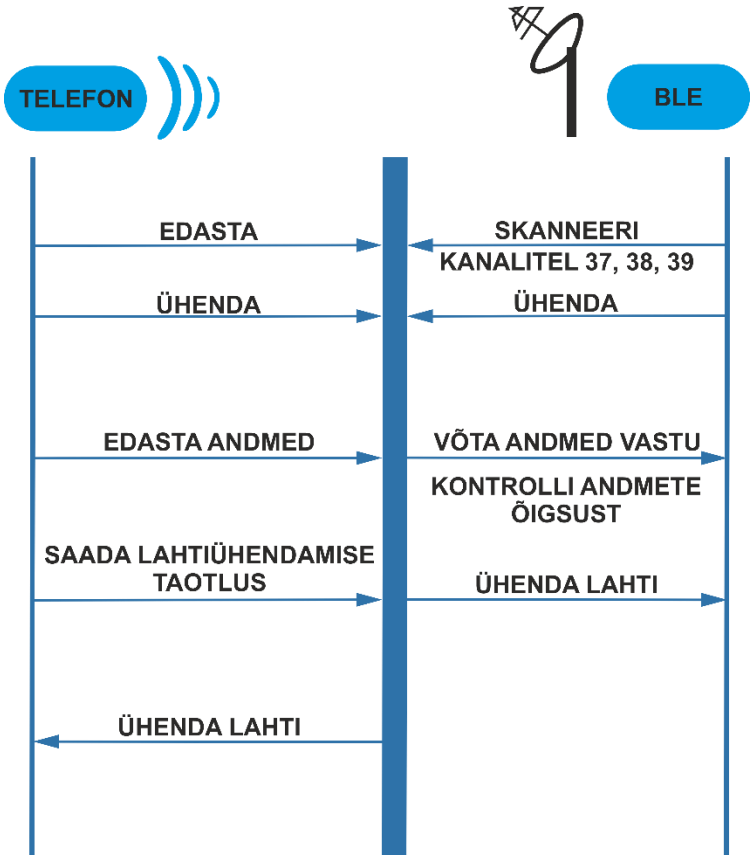
HC-05 kasutab *BT* versiooni 2.0, mis on küll suhteliselt vana versioon, kuid sellegipoolest väiksemate projektide puhul hea kasutada. Lisaks on ta ka ühildatav teiste, uuemate, versioonidega. Tabelis 2 on välja toodud HC-05 viigud, nende nimed ning kirjeldused.

NIMI	**KIRJELDUS**
1. LED	Näitab mooduli olekut: • vilgub üks kord 2 sekundi jooksul: moodul on sisenenud käsurežiimi; • korduv vilkumine: ühenduse ootamine andmesiderežiimis; • vilgub kaks korda 1 sekundi jooksul: ühenduse loomine andmerežiimis õnnestus.
2. STATE	Olekutihvt on ühendatud plaadil oleva LED-iga, mida saab kasutada kontrollimaks, kas *BT* töötab korralikult.
3. RX	Receiver – Võtab andmeid vastu.
4. TX	Transceiver – Saadab andmeid.
5. GROUND	Maandamise punkt.
6. VCC +5V	Voolupunkt. Tahab vähemalt 5 volti.
7. ENABLE / KEY	Kasutatakse ümberlülitamiseks andme- ja AT käsurežiimi jaoks. Vaikimisi on andme režiimil.
8. BUTTON	Kasutatakse ENABLE / KEY pinni jaoks, et lülitada kahe režiimi vahel.

Tabel 2. Mooduli HC-05 viikude kirjeldused.

Lisa 4 - BLE (Bluetooth Low Energy) ühendus

Erinevalt *BT*-st, mis teeb sagedushüppeid täpse ajakava alusel hoolimata kõigest, hüppab *LE* (*Low Energy*) pärast teatud hulga pakettide saatmist, mille jaoks pole vaja ärkvel olla hoides *SCL*-i, et teada saada, kuhu edasi hüpata. Tegelikkuses lubab *LE* seade ühenduse säilitamiseks täielikult raadio välja lülitada. Lisaks on üks omadustest, et seadmed saavad saata soovimatut ülekannet väikeste andmepakketidena. Erinevalt *BT*-st võib *LE*-maailmas seadmete otsimine olla passiivne - kuulatakse lihtsalt õigete kanalite edastuspakette ja kõiki edastusi (Joonis 23). [13]



Joonis 23. BLE ühendus.

Lisa 5 - Moodulite HC-05 ja HC-06 võrdlemine

Ülesandes on kasutusel HC-05, kuna sellel on eelis seoses HC-06 mooduliga. Nimelt HC-05 saab olla nii „*master*“, kui ka „*slave*“ rollis ehk võtta vastu ja saata andmeid teisele seadmele, HC-06 saab olla ainult „*slave*“ rollis. Seadistamise poole pealt on jällegi HC-06 moodul parem tema ühe rolli tõttu ning ka vähemate väljundviikude poolest, kuue asemel neli, millest tulenevalt on seda kergem ühendada ja vajab vähem juhtmeid.

Olles mõlemal erinevad *BT* moodulid peale, on nende riistvara atribuudid samad (Tabel 3). Ainuke erinevus siin on mudelite versioonid, kus HC-06-l on uuem tarkvara viikude 31-34 puudumise tõttu. Siit tuleb ka teine erinevus, et HC-05-l on neli lisa viiku, ehk siis kokku 34 30-ne asemel.

ERINEVUS	**HC-05**	**HC-06**
Pinnide arv	6	4
Master ja *Slave*	*Master* ja *slave*	Ainult *slave*
Režiimid	Käsu- ja andmerežiim	Andmerežiim
Režiimi vahetus nupp	Jah	Ei
BT moodulite mudelid	EGBT-045MS	EGBT-046S
Viikude arv	34	30

Tabel 3. Moodulite HC-05 ja HC-06 võrdlemine.

Lisa 6 – Kood

Lõputöö raames kirjutatud kood asub GitLab repositooriumis lingil https://gitlab.pld.ttu.ee/satsob/Robotite_ylesannete_projekt . Ülesannete koodid asuvad kaustas „Lõputöö ülesanded“.