TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Arvutisüsteemide instituut

IA40LT

Karl Laanemets 143078IASB

JUHTMEVABAL SENSORVÕRGUL PÕHINEV TAIMEDE SEIRESÜSTEEM

bakalaureusetöö

Juhendaja: Maksim Gorev
PhD

Tallinn 2017

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: Karl Laanemets

22.05.17

Annotatsioon

Tänapäevase kiire eluviisi puhul jääb tihti puudu ajast ning tahtejõust tegeleda meelepäraste tegevustega.

Töö eesmärgiks seati mulla niiskust ning õhutemperatuuri mõõtva sardsüsteemi loomine, sardsüsteemidest juhtmevaba prototüüpvõrgu tegemine ning veebisaidi arendamist mõõtetulemuste kuvamiseks.

Töö raskust tõstis süsteemide vahelise suhtluse viisi väljamõtlemine ning pooldupleks raadiomoodulitega selle realiseerimine. Keerukas oli ka harjumuspärasest arendusprotsessist teistsuguse viisi kasutamine andurseadme tarkvara loomisel.

Töö tulemusena loodi patareitoitel töötav sardsüsteem, mis on võimeline mõõtma mulla niiskust ning õhutemperatuuri. Valminud andurseadmest loodi kahe mõõtva seadmega ja ühe kommutaatoriga tähtvõrk. Kommutaatoriga ühendades loodi ka veebiserverit realiseeriv ja mõõtmisandmeid logiv andmekoguja.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 29 leheküljel, 5 peatükki, 14 joonist, ühte tabelit.

Abstract

Plant monitoring system based on a wireless sensor network

Today's fast-paced society often leaves people with no energy or willpower to make time for their beloved pot plants. As a solution, many systems based on wireless networks have emerged to keep track of the growing enviroment of plants. These systems are often capable of monitoring the enviroment beyond many kilometers and are appropriate for greenhouses or big fields. Such dimensions require expensive wireless solutions to be functional and thus makes them unsuitable to be used in a homely enviroment, such as appartments or private housing. Also, the aquired data often is not made suitable to be used by an ordinary user.

The aim of this thesis is to design and implement an embedded system which is capable of measuring soil humidity and air temperature to keep track of a plant's growing enviroment. Based on the system a wireless network is designed to handily keep track of a number of plants. To monitor the readings, a web page is implemented.

In the second chapter, the author proposes the requirements of the system and based on that, a design of the whole system. This includes the protocol to be used and message formats for communication.

The third chapter gives an overview of the hardware used for the realisation of the system and the reasons behind the choice of components.

The final chapter covers the software side of the system, how the works and how the communication and web site implementation looks like. It also describes the software development methodologies that are used.

As a result, the embedded sensor systems work on batteries and the wireless web works as intended, despite the problems that arose during development.

The thesis is in Estonian and contains 29 pages of text, 5 chapters, 14 figures, 1 table.

Lühendite ja mõistete sõnastik

ADC	*Analog-Digital Converter,* Analoog-digitaalmuundur
CCS	*Code Composer Studio,* Texas Instrumentals'i poolt loodud Eclipse'l põhinev arenduskeskkond
dBm	*decibel-milliwatts*, Detsibelli millivati suhtes
Draiver	*Draiver,* Välisseadet opsüsteemiga liidestav juhtimisprogramm
GPIO	*General Purpose Input/Output,* Üldotstarbeline sisens-väljund
IR	*Infra red,* Infrapuna
LAMP	*Linux, Apache, MySQL, PHP,* Vabavara komplekt veebisaitide loomiseks
LPM3	*Low Power Mode 3*, Mikrokontroller G2553 töörežiim
MOSFET	*metal–oxide–semiconductor field-effect transistor*, Isoleeritud paisuga väljatransistor
POSIX	*Portable Operating System Interface,* standartitekogum tagamaks ühilduvus operatsioonisüsteemide vahel
TDD	*Test-driven development,* tarkvara arendusmeetod, mis näeb ette testide kirjutamist enne programmikoodi
UART	*Universal Asynchronous Receiver/Transmitter,* universaalne asünkroontransiiver
USB	*Universal Serial Bus,* universaalne järjestiksiin
VLO	*Very-Low-Power Low-Frequency Oscillator,* madala võnkesagedusega ostsillaator
Wi-Fi	*Wireless Fidelity,* raadiokohtvõrk

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Tänapäevasele ühiskonnale iseloomuliku kiire eluviisi tõttu jääb inimestel tihti puudu ajast ja energiast, et pidevalt jälgida oma taimede kasvukeskkonda ning nende eest hoolt kanda. Suure hulga jälgimine muutub aga väga kiiresti tüütuks ja ajakulukaks. Rohkemaarvuliste taimede kasvukeskkonna jälgimiseks ning hindamiseks on välja kujunenud mitmeid juhtmevabadel andurvõrkudel põhinevad lahendusi, mis võimaldavad jälgida suure vaevata ja pikkade vahemaade tagant taimede keskonnanäitajaid. Mainitud võrgud on tihtilugu suuremajooneliste kasvuhoonete või põldude jälgimiseks ning võimaldavad tihti ka edasiarendust hoolduse automatiseerimiseks. Kallite juhtmevabade andmeedastuslahenduste kasutamine nendes süsteemides muudab nad aga sobimatuks koduseks ülesseadmiseks näiteks korterites või eramajades.

Kodused lahendused on see-eest aga enamasti juhtmestatud ja kohmakad, mistõttu on neid raske sobivasse kohta ümber tõsta. Juhtmestus tähendab tihti taimede andurseadmete toitestamist seinakontaktist [1]. Seega on süsteem enamasti mittemobiilne ja juhtmevaba suhtelahendus kaotab suuresti oma võlu.

Töö eesmärgiks on realiseerida patareitoitel töötav sardsüsteem ning neist koosnev juhtmevaba võrk, mis on sobilik väikesearvulise taimedekogumi monitoorimiseks koduses keskkonnas. Iseseisva toitega andursõlmede ja juhtmevaba suhtlusega soovitakse saavutada taime lihtsat ümberpaigutamisvõimalust. Taimede jälgimise mugavdamiseks luuakse süsteemile veebiliides.

Töös käsitletakse üldiselt süsteemi disaini, riistvara valikut, selle põhjal loodud lahendust ning kirjutatud tarkvara.

2 Süsteem

Peaükis kirjeldatakse süstemile seatud nõudeid, selle disaini ning tööd. Süsteemi võrk on ehitatud tähtvõrgu topoloogiat kasutades.

2.1 Süsteemi nõuded

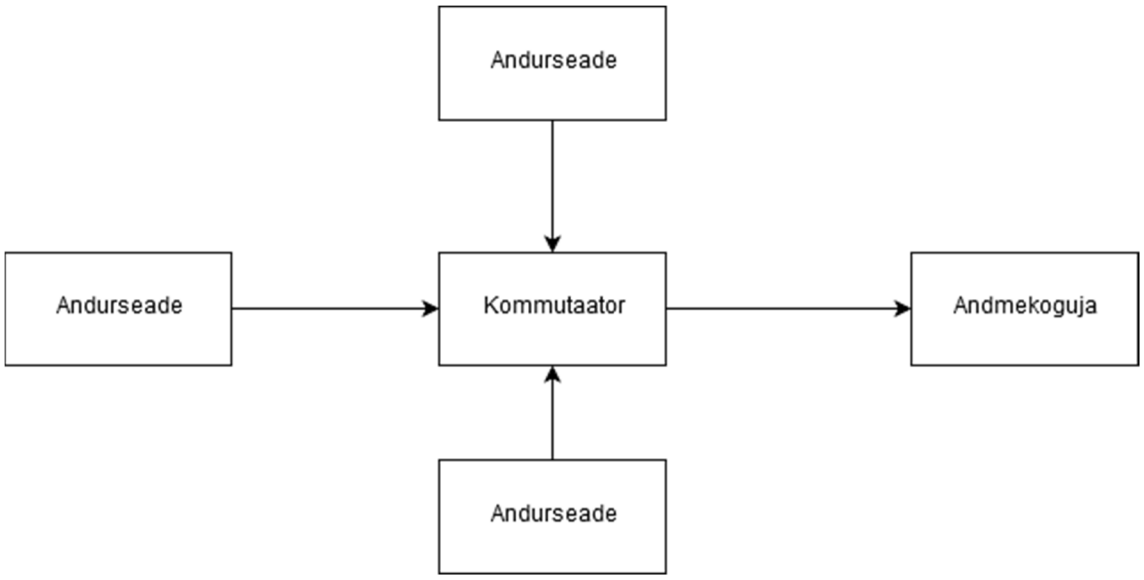
- Andureseade peab olema seadistatava uinumisperioodiga
- Andurseade peab kasutama patareitoidet
- Peab mõõtma mulla niiskust
- Süsteem peab mõõtma õhutemperatuuri
- Peab mõõtma patarei toitepinget
- Võrgus edastatakse ainult andurseadme poolt mõõdetud suurusi ja suhtluskanali müranäitajaid

Andurseadme tarkvarale on seatud Tallinna Tehnikaülikooli aine IXX0042 projekti raames järgnevad lisanõuded:

- Tarkvara peab olema disainitud modulaarselt
- Süsteemi disain peab olema hierarhiline

2.2 Süsteemi disain

Disainitav süsteem koosneb andursõlmedest, kommutaatorist ning andmekogujast (Joonis 1).



Joonis 1. Süsteemi plokkskeem 3 andurseadmega.

Süsteemi võrgu topoloogiana on kasutatud tähtvõrku ilma andurseadmete vahelise suhtluseta. Kommutaator käitub peamiselt kui andmeedastaja, saates andursõlmelt vastu võetud paketid edasi andmekogujasse. Alternatiivina oleks võimalik kommutaator eemaldada, kuid seda lahendust pole kasutatud, sest andmekoguja resurssivajadus ületab kasutatava mikrokontrolleri võimed ning ka seetõttu, et juhul kui ei kasutata antud mikrokontrollerit andmete vastuvõtmiseks, peab kasutatavale riistvarale looma raadiomooduli draiveri. Mikrokontrolleril realiseeritud draiveris on aga olemas UART liides, läbi mille on võimalik edastada infot andmekogujale, mis küll suurendab süsteemi mõõtmeid, kuid ei mõjuta mobiilsust.

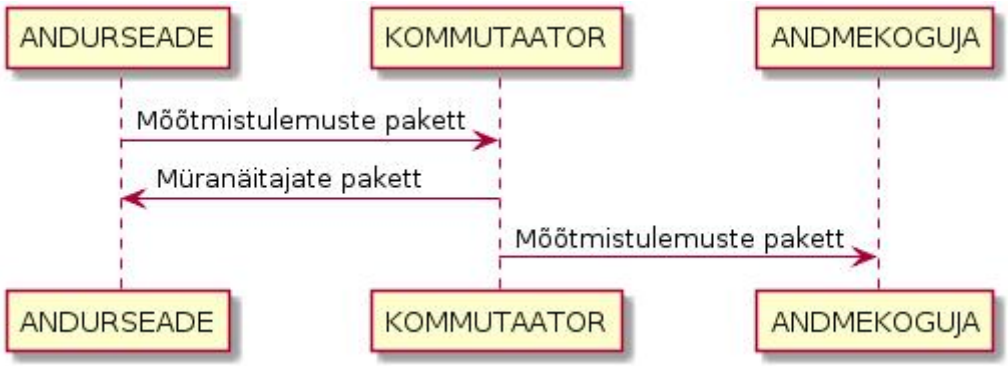
Võrgu topoloogia on valitud põhjusel, et lahenduse poolt katetava väikese ala tagajärjel ei ole vajalik andmeid edastada suurte vahemaade taha, seega ei ole võrgu ehitamiel mõistlik kasutada mesh või mõnda muud topoloogiat, mis näeb ette sensorsõlmede vahelülidena kasutamist, et andmete propageerimise vahemaad pikendada või hierarhilist struktuuri luua. Meshi kasutuselevõtt tähendaks suhtlusprotokolli keerukuse tõusu, mis omakorda tõstaks riistvara ressurssidele seatavaid nõudeid, vähendades sellega seadme patarei eluiga. See aga soosib tähtvõrku, kuna andmete edastamiseks loodavad ja

kasutatavad suhtlusprotokollid on disainitavad väga lihtsateks ja kiiretoimelisteks, kasutades nende implementeerimiseks vähem andurseadme resursse.

Valitud topoloogia miinusteks võib pidada võrgu talumatust kommutaatorsõlme töös tekkivatele viperustele. Kommutaatorsõlme töö osatähtsus süsteemi töös on kriitilise suurusega, mille töö lakkamisel ei jõua paketid andmekogujasse, mis põhjustab pealtnäha ka andmekoguja töö katkestuse. Kommutaatorsõlme töökindluse tagamiseks on vaja disainida too võimalikult lihtne, ehk kasutada seda ainult andmete edasisaatmiseks.

2.3 Süsteemi suhtlus ning pakettide vormindus

Süsteem suhtleb järgnevalt: andurseadme magamisperioodi lõppemisel tehakse süsteemi poolt vajalikud mõõtmised ning koostatakse nende põhjal pakett. Pakett saadetakse koheselt peale koostamist, kontrollimata kas kanal on kasutuses või mitte. Süsteemi osade vahelisel suhtlusel ei looda osapoolte vahelist kanalit ning ei tagata pakettide kohalejõudmist. Kommutaator saadab vastavalt vastuvõetud paketist loetud aadressile vastusena suhtluse jooksul mõõdetud kanali müranäitajad, mille järel saadetakse vastuvõetud info edasi andmekogujasse (Joonis 2).



Joonis 2. Süsteemi osade vaheline suhtlus.

Mõõtmistulemuste paketti lisatakse selle sõelumise lihtsustamiseks paketi kogupikkus, andurseadme lähteaadress, kommutaatori aadress ning mõõtetulemused. Andurseadme poolt saadetavad paketi tulemuslik osa koosneb peamiselt kahe välja kordusest: mõõtetulemuse identifikaator ning seejärel mõõtetulemus (Joonis 3). Mõõtetulemuse identifikaator on eristatud mõõtmistulemuste tüüpide järgi, näiteks temperatuuri identifikaator erineb niiskuseanduri omast. Süsteem saadab korraga 3 mõõtetulemust:

temperatuur, mulla niiskus ning patarei toitepinge. Mõõtetulemuste järel lisatakse paketti eelneva töötsükli suhtluse jooksul mõõdetud müranäitajad.

Paketi pikkus	Lähteaadress	Sihtaadress	ID	Mõõtetulemus	ID	Mõõtetulemus	ID	Mõõtetulemus

Joonis 3. Andurseadme andmepaketi vorming müranäitajateta.

Kommutaatori poolt loodav müranäitajate pakett omab sarnast ülesehitust. Paketi sõelumiseks on lisatud samad kolm esimest välja ning lõpetuseks kaks välja, mis koosnevad müranäitajale omasest identifikaatorist ning seejärel mõõdetud kanali müranäitajast.

3 Riistvara

Antud peatükk kirjeldab süsteemi eri osade komponendivalikut süsteemi realiseerimiseks ning andurseadme elektriskeemi. Andurseadme ning nendest koosneva võrgu realiseerimiseks on kasutatud mikrokontrollereid. Andmete hoiustamiseks on kasutatud Raspberry Pi nimelist miniarvutit.

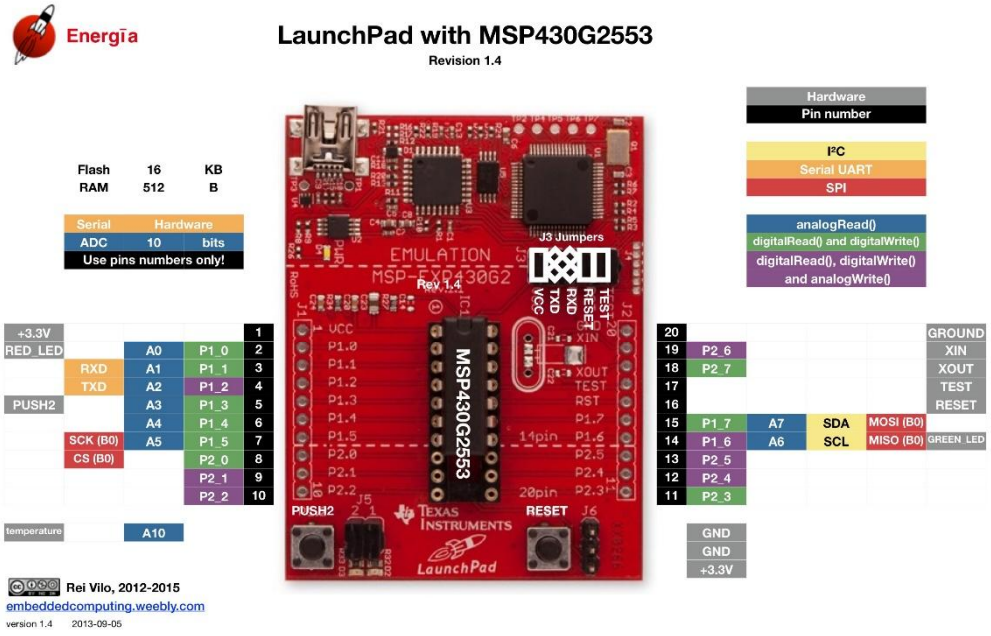
3.1 Andurseade

Andurseadme realisatsioon koosneb mikrokontrollerist, raadiomoodulist ning niiskusandurist. Peatükk annab ülevaate kasutatud komponentidest ning nende valikupõhjustest.

3.1.1 Mikrokontroller

Andurseadme juhtseadmena on kasutatud TI MSP430 perekonda kuuluvat mikronkontrollerit G2553 koos arendusplaadiga MSP-EXP430G2.

Arendusplaat võimaldab MSP430G2553 programmeerimist ja programmikoodi silumist läbi plaadil oleva emulaatori USB liidese, mis on ühendatud läbi J3 pistiku plaadi mikrokontrolleri osa külge (Joonis 4) [2] .



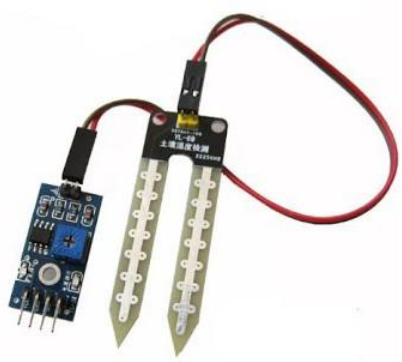
Joonis 4. MSP-EXP430G2 arendusplaadi väljaviikude skeem

Kuna mikrokontroller toimib operatiivpingel 1,8-3,6V ning omab erinevaid madala voolutarbega töörežiime on valitud mikrokontrollerit sobiv kasutada patareitoitel töötavate süsteemide loomiseks [2][3]. Kontrolleri valikut soodustas ka sisseehitud temperatuuriandur, analoog-digitaalmuundur ning selle kasutamiseks 1.5V ja 2.5V võimaldav referentspingegeneraator. Generaator lihtsustab analoog-digitaalmuunduri tulemuste kalibreerimist, sest tootjapoolsed kalibreerimismeetodite olemasolu tõttu [3].

3.1.2 Niiskusandur

Sensoranduri valikul on arvesse võetud analoogväljundi olemasolu ning mooduli toimimist sobilikul pingevahemikul.

Lahenduses on kasutatud imendumis-impedants-niiskusandurit, mis koosneb YL-69 sensorist ning YL-38 trükkplaadist. Sensor kasutab mulla niiskuse mõõtmiseks vaba vee leidumist mulla kapillaarsetes poorides [4]. Vee koguse põhjal muutub anduri takistus, muutes sellega skeemi väljundpinget (Joonis 5).



Joonis 5. YL-69 + YL-38 niiskusandur

Andurit on võimalik kasutada erinevatel tööpingetel, kasutamata trükkplaadi pealset komparaatorit. Kasutamine on võimalik arvutades vastavalt patarei hetkepingele mõõtmisvahemiku minimaal ja maksimaalväärtused ning selle põhjal tõlgendada kas tegemist on niiske või kuiva mullaga.

3.1.3 Raadiomoodul

Juhtmevaba suhtlusmooduli ning -tehnoloogia valikul on arvestatud moodulite voolutarvet, tööks vajalikku toitepinget, mooduli kasutuslihtsust ning kontrolli suhtluse protokollivaliku üle. Võrreldud on intituudist kättesaadavaid mooduleid tehnoloogiate järgi.

Wi-Fi ja Bluetooth lahenduste peamiseks tugevaks küljeks on suhtlusprotokollide standardiseeritus ning võrreldes ise valmistatud protokollidega just nende turvalisus. Formaadi poolest standardiseerimata moodulid annavad aga võimaluse muuta käitumine enda süsteemile kohasemaks, olles paindlikum realisiooni loomisel ning seega võimaldades lihtsat ja kiiret lahenduse prototüüpimist. Krüpteering ning kanalipõhine suhtlus on vajadusel võimalik välja jätta, muutes sellega süsteemi suhtlus kiiremaks.

Moodulite sobivus patareitoitega süsteemi on enamasti sõltuv erinevate töörežiimide olemasolust, mis võimaldaksid nende kasutust. Wi-Fi ja Bluetooth moodulite voolutarve ei õigusta ennast seetõttu, tarbides kohati kordades rohkem voolu ning tihti toimides ebasobival toitepingel (Tabel 1).

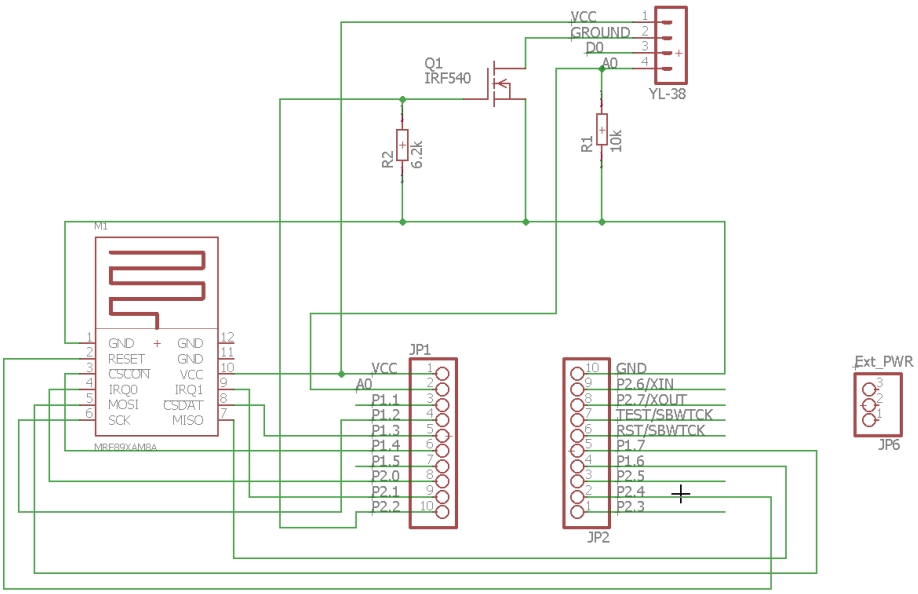
Tabel 1. Erinevate juhtmevaba moodulite näitajate võrdlus tehnoloogiate järgi.

Moodul	**TX (mA)**	**RX (mA)**	**Sleep(mA)**	**Toitepinge(V)**
ESP8266 (Wi-Fi)	135-215 [5]	60 - 62 [5]	0,01 [5]	3.3V [5]
hc04 (Bluetooth)	8	8	-	3.1~4.2
MRF89XAM8A (ISM sagedusvahemik)	25	3	0.001	2,1-3,6

Mikrokontrolleri toitevahemikuga sobivuse, madala voolutarbe ning kodukasutuseks piisava edastuskauguse tõttu on valitud välja MRF89XAM8A raadiomoodul, millele on Arvutitehnika instituudis realiseeritud kanalikihi (*data link*) arendajaliides [6]. Moodul valiti just olemasoleva arendajaliidese ja protokolli valimisvabaduse poolest, sest on võimalik luua oma protokoll andmete edastamiseks. Protokolli loomisega on võimalik muuta süsteemi kiiretoimelisemaks, vähendades sellega süsteemi töötsükli pikkust, mille tagajärjel pikeneb patarei eluiga. Valikut mõjutas ka mooduli lihtne kättesaadavus.

3.1.4 Süsteemi elektriskeem

Andurseadmes kasutatakse MRF89XAM8A raadiomoodulit, trükkplaadist YL-38 ning sensorist YL-69 koosnevat niiskusandurit (Joonis 6).



Joonis 6 Andurseadme elektriskeem.

Skeemil olevast pistikust JP6 varustatakse süsteemi tööks vajaliku pingega. Toitestamiseks on kasutatud kahte AAA patareid jadamisi. Niiskusanduri analoogväljundisse on lisatud takisti, et muuta väljundpinge mikrokontrolleri analoog-digitaalmuunduri sisemise referentspingele sobilikuks.

Süsteem on lisatud n-kanali MOSFET lülitamaks välja niiskusandur juhul kui mõõtmist ei toimu. Anduri väljalülitamisega pikendatakse selle eluiga, kuna sensor ei korrodeeru nii kiiresti. Väljalülitamisega vähendatakse märgatavalt ka süsteemi voolutarvet.

Raadiomooduli väljaviigud on jäetud ühendamata, järgides sellega teegi loojate nõuannet [6].

3.2 Kommutaatorsõlm ja andmekoguja

Kommutaatorsõlme realiseerimiseks on kasutatud Texas Instrumental'i poolt toodetavat G2553 mikrokontrollerit koos arendusplaadiga MSP-EXP430G2. Arendusplaadile on lisatud ka MRF89XAM8A raadiomoodul andmete vastu võtmiseks. Kommutaatorandmeid toitepinge tagatakse läbi arendusplaadi emulaatorosa USB liidese, mis on ühendatud Raspberry Pi USB liidesesse [2]. Raspberry Pi ning arendusplaat on

omavahel ühendatud UART jadaliidese kaudu läbi arendusplaadi viikude P1.1 ja P1.2 ning Raspberry GPIO viikude 8 ja 10.

4 Tarkvara

Süsteemi tarkvaraline lahendus jaotub peamiselt kolmeks osaks: andurseadme, kommutaatori ning andmekoguja tarkvara. Andurseadme ning kommutaatori programm on kirjutatud C keeles. Andmekoguja jadaliidese suhtlus on realiseeritud samuti C keeles, andmete talletamiseks ning kuvamisels on kasutatud LAMP komplekti.

4.1 Andurseade

Seade on loodud Tallinna Tehnikaülikooli aine IXX0042 projekti raames, järgnevate nõuete põhjal:

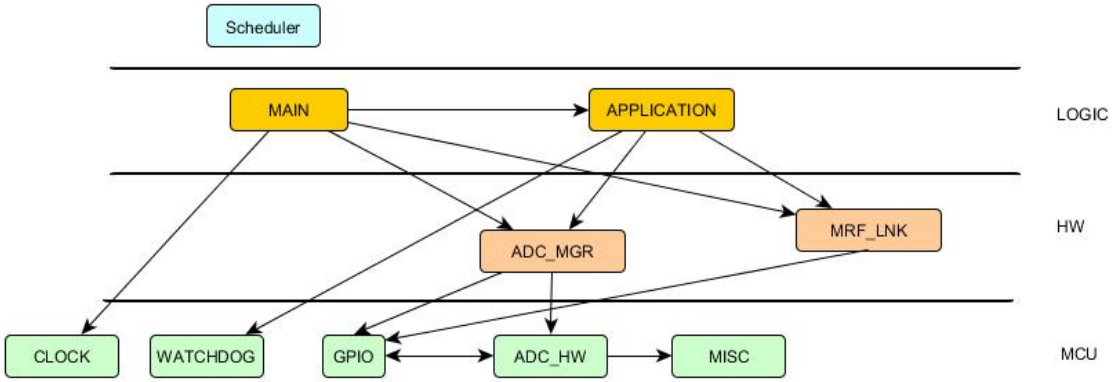
- Süsteemil peab olema hierarhilise disainiga
- Tarkvara peab olema modulaarne
- Vähemalt ühe mooduli loomisel tuleb kasutada TDD-d, kirjutades ühikteste

Tarkvara modulaarsus ning üleüldine kasutajaliidese loomine aitab muuta süsteemi töö jälgitavamaks. See muudab ka lihtsamaks tarkvarale moodulite lisamise.

Tarkvaras on kasutatud olekumasinaid andurseadme töö juhtimisel. Olekumasina kasutuse põhjus on nende poolt pakutav süsteemi tööjärje lihtne jälgimine, programmi silumise või mõnel muul eesmärgil. Lihtsustatud on ka programmeerimisel tehtud vigade leidmine, kuna igale olekule on vastavuses kindel programmi jupp. Testimise poolelt vähendab olekumasina kasutamine töömahtu, sest süsteemi täielikuks läbitestimiseks on vajalik vaid läbida kõik olekute vahelised siirded, mis on testimisraamistikega lihtsati automatiseeritav [8].

4.1.1 Ülevaade süsteemist

Tarkvara on struktureeritud hierarhiliselt, jagades süsteemi moodulid kihtide vahel. Kihiliselt on jaotatud süsteem 3 osasse: MCU, HW ja LOGIC (Joonis 7).



Joonis 7. Andurseadme tarkvarmoodulite struktuur.

MCU tasemel asuvad mikrokontrolleri funktsionaalsuse tarkvaraline abstraheerimine, süsteemi initsialiseerimine tööks ja sisend-ning väljundviikude seadistamine vastavalt ülessandele.

GPIO seadistab kasutamata väljaviigud väljunditeks ning võtab kasutusele *pull-down* takistid, lähtudes tootjapoolse madala voolutarbe saavutamise eeskirjadest ning CCS arenduskeskkonna komponendi *ULP advisor* antud soovitustest [3]. Moodul loob ka liidese niiskusanduri lülitamiseks vooluahelast lahti.

WATCHDOG mooduli puhul on tegemist kasutajaliidesega mikrokontrolleri uinumisperioodi seadistamiseks ning üleüldine eri töörežiimide vahelise lülitamise realisatsiooniga. Uinumisperioodi mõõtmiseks kasutatakse sisemist VLO taktigeneraatorit, mille sagedus kõigub 4-20kHz vahel [6]. Antud taktigeneraatorit on kasutatud kuna pole vajalik ülitäpne ajapidamine, mistõttu ei pidanud autor vajalikuks välise ning täpsma kristalli lisamist süsteemi. Uinumisperiood on seadistatav minimaaselt 1,6 minutit ning maksimaalselt 584 päeva, mis võib tõusta olenevalt taktigeneraatori tegelikust sagedusest.

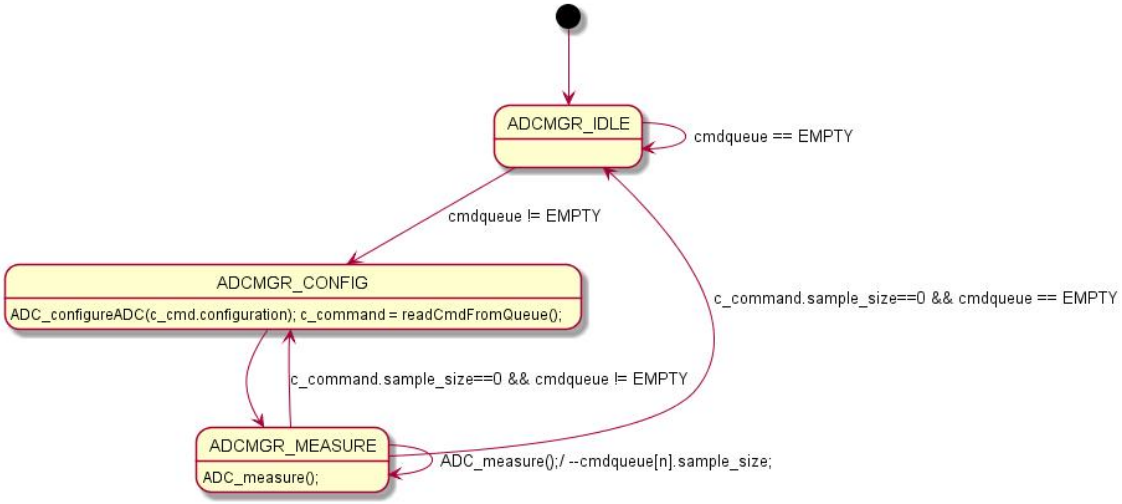
ADC_HW on realiseeritud loomaks liides mikrokontrolleri kümnekanalise analoog-digitaalmuunduri kasutamise lihtsustamiseks. Moodul liides võimaldab kontrollida muunduri sisselülitatust ning koormatust läbi liidese avalike funktsioonide ADC_checkIfADCON ja ADC_checkIfADCBusy. Samuti võimaldab liides

konfigureerida muundurit vastavalt süsteemis tehtavatele mõõtmistele, kasutades ADC_configureADC funktsiooni. Hetkerealisatsioonis on võimalik seadistada muundurit mõõtmaks kolme parameetrit kolmel erineval kanalil: sisseehitatud temperatuurianduri väljundpinget, niiskusanduri väljundpinget ning patarei toitepinget. Kasutatav referentspinge seadistus nendel mõõtmistel on 2.5V. Liides annab võimaluse ka ühekordse mõõtmise elluviimiseks ADC_measure funktsiooniga, eelnevalt ADC_configureADC funktsiooniga muunduri konfigureerimisel. Moodul võimaldab ka privaatsete funkstioonide kasutamisel muunduri tuuma sisse-ja väljalülitamist ning mõõtmise alustamise ning lõpetamise funktsionaalsust.

MISC moodul võimaldab mikrokontrollerite süsteemiregistrite bittide seadistuse kontrolli, andes sisendparameetriteks kontrollitav süsteemiregister ning biti järgunumber.

HW kihis olevad moodulid peamistek ülessanneteks on mõõtmiste tegemise loogika implementeerimine ja sõnumite saatmine. MRF_LNK mooduli puhul on tegemist süsteemis kasutatava MRF89XAM8A raadiomooduli arendajaliidesega, mis võimaldab võrgusõlmede addresseerimist ning üleüldist andmeside pidamist. Teek on koostatud Tallinna tehnikaülikooli Arvutitehnika Instituudis [7].

ADCMGR moodul loob liidese mõõtmiste tegemiseks andurseadmes. Liides võimaldab mitmekordsete mõõtmiste tegemist, kasutades selleks mooduli ülesannete järjekorda. Mõõtmise tegemiseks tuleb vastavat tüüpi mõõtmise konstant viia ADCMGR'i mõõtmiste järjekorda ning tulemuse saamiseks siis liidese funktsioone kasutades mooduli tulemuste puhvrit pollides. Moodul on realiseeritud olekumasinana (Joonis 8), mis võimaldab mõõtmise tegemisel süsteemi teistel moodulitel oma tööd jätkata.



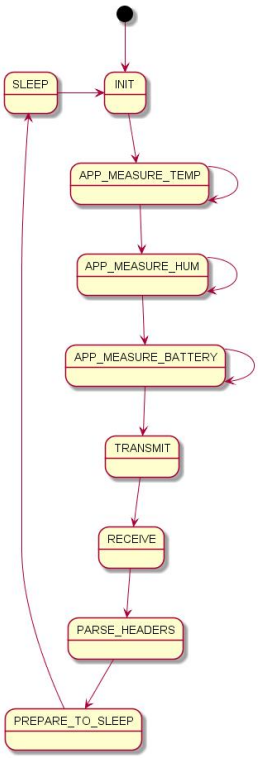
Joonis 8. ADCMGR mooduli olekudiagramm.

Ebasobiva mõõtmise tuvastamisel proovitakse antud mõõtmist teha uuesti 10 korda. Kui mõõtmine ei õnnestu, tagastatakse veateade.

Moodul MAIN koosneb süsteemi initsialiseerimiseks vajminevatest funkstioonide väljakutsetest ning lihtsast *while* tsüklist, mille sisuks on ADCMGR ja APPLICATION moodulite tsükliliste funktsioonide väljakutsumine.

4.1.2 Süsteemi töö

Süsteemi tööd juhib LOGIC taseme moodul APPLICATION. Moodul sisaldab olekumasinat (Joonis 9).



Joonis 9. APPLICATION mooduli olekugraaf.

Süsteemi olekus INIT initsialiseeritakse ADCMGR ehk lisatakse mõõtmiste järjekorda kolm mõõtmistegevust: temperatuur, niiskus ning seejärel patarei taseme mõõtmine. Samuti muudetakse raadiomooduli töörežiimi vähendamaks voolutarvet. Seejärel liigutakse edasi olekutesse, milles kontrollitakse pidevalt ADCMGR liidese kaudu, kas järjekorda lisatud mõõtmine on lõppenud. Mõõtmise lõppemisel lisatakse saadud mõõtmistulemust töötsüklil saadetavasse paketti ning liigutakse edasi järgmise suuruse mõõtmise juurde. Juhul kui mõõtmine pole lõppenud siirdutakse tagasi samasse olekusse kuniks mõõtmine on lõppenud. ADCMGR liidese poolt välja antud veateate korral täidetakse vastava mõõtmise andmebaidid nullidega ning siirdutakse järgmisesse olekusse. Mõõtmisolekute lõpul siirdutakse olekusse TRANSMIT, äratatakse raadiomoodul ning lisatakse saadetavasse paketti eelneval töötsüklil olekus RECEIVE vastuvõtmisel mõõdetud müranäitajad. Süsteemi esmasel käitamisel täidetakse selle mõõtmise sisubaidid nullidega. Pakett saadetakse ning siirdutakse edasi RECEIVE olekusse, kus jäädakse saadetud paketile vastust ootama. Ooteperiood on 5 millisekundit, mille jooksul tehakse uued kanali müranäitaja mõõtmised, mida järgneval töötsüklil saata. Raadiomooduli veateate korral alglähtestatakse ning lülitatakse voolusäästlikku režiimi peale mida liigutakse järgmisesse olekusse. Paketi kätte saamisel lülitatakse

raadiomoodul tagasi voolusäästlikku režiimi. PARSE_HEADER olekus sõelutakse vastuvõetud pakett, mille järel liigutakse PREPARE_TO_SLEEP olekusse, kus seadistatakse süsteemi uneperiood ja seejärel suundutakse uneolekusse, millest äratab süsteemi WATCHDOG moodul süsteemi katkestusega, mille järel algab töötsükkel algusest.

4.1.3 Arendusprotsess

Eelnevalt välja toodud nõuetel toetudes on ADC tarkvaramoodul arendatud kasutades TDD-d. Arendusmeetod näeb ette mooduli käitumusliku mudeli kirjapanekut, kasutades selleks mooduli nõuetel põhinevate testide kirjutamist. Testid kirjutatakse enne programmi koostamist, mille tööd kontrollitakse. Seejärel kirjutatakse piisav programmi kood täitmaks testi poolt kontrollitavaid nõudeid.

TDD vähendab üleüldiselt vigade avastamiseks kuluvat aega, sest protsess on inkrementaalne ning lühiajaline. Lisaks ajakulu vähendamisele muudab TDD tarkvara modulaarsemaks ning liidese lihtsasti kasutatavaks. Sobiv kasutamiseks uute, eelneva koodibaasita projekti elluviimiseks.

TDD metoodika paremaks mõismiseks on siinkohal näitena toodud ADC mooduli stopConversion funktsiooni arendusprotsess kasutades TDD-d.

Kuna ADC moodul on riistvara abstraheerimiseks on vajalik süsteemi kasutajaliidest, antud juhul analoog-digitaalmuunduri töö juhtimiseks kasutatavate süsteemiregistreid, jäljendada. See tähendab süsteemi ja selle kasutajaliidese võltsingu loomist, kasutades selleks programmeerimiskeele muutujaid. Funktsiooni stopConversion puhul on vaja võltsida ADC10CTL0 registrit (Joonis 10).

15	14	13	12	11	10	9	8
SREFx			ADC10SHTx		ADC10SR	REFOUT	REFBURST
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
MSC	REF2_5V	REFON	ADC10ON	ADC10IE	ADC10IFG	ENC	ADC10SC
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Joonis 10. Süsteemiregistri ADC10CTL0 bitiväljad [3].

Funktsiooni testimise alustamiseks on esmalt vaja mõelda disainitava mooduli või siis antud juhul funktsiooni käitumise üle. Sellest tulenevalt on võimalik paika panna mida

testida. Funktsioon peab olema võimeline lõpetama mõõtmisprotsessi kui ka selle tööks vajalikud teised komponendit, seega referentspinge generaatori ning analoog-digitaalmuunduri tuuma. Bittide ENC, REFON ja ADC10ON seadistamine registris võimalik seda teha, seega on vaja testida nende väljade muutust. Selle põhjal on kirjutatud testimisjuhtum nimega *test_ADC_stopConversion* (Joonis 11).

```
void
test_ADC_stopConversion()
{

  ADC10CTL00 &= ~ENC; /* Stop conversion */
  ADC10CTL00 &= ~(REFON + ADC10ON); /* Disable voltage ref, ADC core */
  ADC_stopConversion();
  CU_ASSERT(ADC10CTL0 == ADC10CTL00);

}
```

Joonis 11. Testjuhtumi kood.

Testimisel kasutatud testimisraamistikuks on valitud CUnit, mis võimaldab hinnata muutujatest tehtavate muudatuste õigsust ning testide läbiviimise protsessi automatiseerida. Testi kirjutamisel on emuleeritud eelenvalt välja toodud bittide seadistamist võltseregistris ADC10CTL00, võrreldes uut väärtust raamistiku funktsiooni CU_ASSERT abiga ADC10CTL0 väärtusega, mis on peale muutmist oodatav väärtus.

ADC_stopConversion on testi esimesel jooksutamisel tühi, mistõttu testis kasutatav funkstioon annab raamistikule teada, et test on läbi kukkunud. Seejärel kirjutati funktsiooni vajalik funktsionaalsus (Joonis 12).

```
void
stopConversion()
/*!*****************************************************
**
** Stop ADC conversion, disable internal refrence generator and ADC core
**
*****************************************************/
{

  ADC10CTL0 &= ~ENC; /* Stop conversion */
  ADC10CTL0 &= ~(REFON + ADC10ON); /* Disable voltage refrence generator and
ADC core */

}
```

Joonis 12. Valmis stopConversion funktsioon peale testimisprotsessi.

Testi taaskordsel väljakutsumisel annab CU_ASSERT raamistikule märku õnnestunud testist.

4.2 Kommutaatorsõlm

Seadme tarkvara on kirjutatud C keeles, kasutades raadiomooduli teeki MRF_LNK. Tarkvaraline keerukus on hoitud madalana, tahtes sellelega vältida keeruka loogika kirjutamisel tekkivaid esinevaid ilmutamata vigu.

Süsteemi töötsükkel koosneb perioodilisest kuulamisest ning seejärel vastavalt sellele, kas on võetud andmeid vastu või mitte, seadmele vastamisest ja andmete edasisaatmisest või kanali kuulamise jätkamisest. Andmete edastamiseks kasutatakse teegis realiseeritud UART draiverit.

4.3 Andmekoguja

Andmekoguja tarkvaraline lahendust koosneb kolmest osast: kommutaatorsõlmelt saabuvate sõnumumite vastuvõtmine, vastuvõetud andmete talletamine andmebaasi ning kasutajale kuvamine.

Andmekoguja realiseerimiseks kasutatav Raspberry Pi'l töötab Raspbian Jesse Lite operatsioonisüsteem.

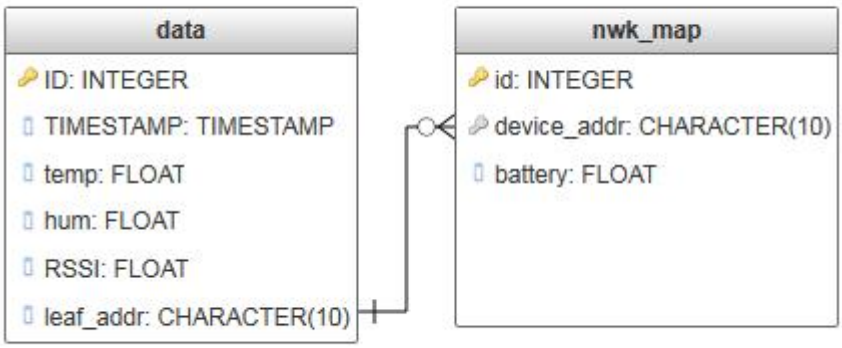
Kommutaatorsõlmel kasutatava raadiomooduli olemasoleva UART driveri liidese tõttu on kasutatud kommutaatorsõlme ja andmekoguja vaheliseks suhtluseks just seda

võimalust. Andmete vastuvõtmiseks jadaliideselt on kasutatud C keeles kirjutatud WiringPi teekide kogumit. Andmete lihtsaks faili kirjutamiseks on kasutatud C keele standardteekis leiduvaid POSIX teeke. Andmete pordilt lugemiseks ning faili kirjutamiseks on kirjutatud C keeles readingSerial.c programm, mis loeb infot Rasberry Pi viikudelt.

Andmete lugemiseks välja valitud port dev/ttyAMA0. Selle pordi kasutamine vähendab keerukust andmete interpreteerimisel, sest USB kasutamisel jadasuhtluseks vajaks kahepoolset suhlust seadme tuvastamiseks, mis suurendaks kommutaatorsõlme ressursikasutust ja vähendaks sellega süsteemi jõudlust. Kuna ttyAMA0 on vaikeseadistusena seotud plaadil oleva Bluetooth mooduli külge, on muudetud /boot/blabla, ühendamaks lahti ta sealt ning võimaldades sellega GPIO viikude kaudu UART ühenduse loomist [8].

Kuna ttyAMA0 on seotud Linuxi konsooliga ei ole võimalk lugeda seda samaaegselt WiringPi teegiga rikkumata loetavaid andmeid. Selleks, et WiringPi teeki saaks kasutada on ttyAMA0 lahti ühendatud linuxi konsoolist, kustutades /boot/cmdline.txt failist seotus konsooliga.

Programm *readingSerial* logib andmed faili *serial.log*, millest hiljem loeb *db.php* skript andmed andmebaasi. Andmebaas koosneb kahest tabelist, milles talletatakse vastuvõetud info (Joonis 13). Tabelis *data* hoitakse kõiki vastu võetud mõõtmistulemusi ning *nwk_map* tabelis võrku ühendatud seadmete kirjeid.



Joonis 13. Andmebaasi struktuur.

Skript *db.php* teeb ka vajalikud andmete tõlgendused enne andmebaasi viimist, mis hõlmab *Texas Instrumentals'i* poolt ettenähtud arvutused arvutamaks analoog-digitaalmuundurilt saadud tulemus ümber Celsiuse kraadidesse ja niiskusanduri mõõtetulemuste kalibreerimist olenevalt seadme hetkesest toitepingest [3].

Andmebaasis talletatakse temperatuur Celsiuse kraadides, niiskus analoogväljundist saadud pinge kujul. Kanalimüra talletatakse RSSI veergu dBm-ides.

Veebisaidil kuvatakse kõik *nwk_map* tabelis leitavad seadmed ning nende värskeimad mõõtmised (Joonis 14).

Joonis 14. Veebisaidi ekraanitõmmise näide 3 andurseadmega.

5 Kokkuvõte

Bakalaureusetöö tulemusena loodi potitaimede mullaniiskust ning kasvukeskkonna õhutemperatuuri mõõtev andurseade, mis toimib patareitoitel. Loodi ka seadmetest koosnev juhtmevaba võrgu prototüüp. Mõõdetud mõõtmistulemuste kuvamiseks on realiseeritud ka lihtne veebisait.

Andurseadme ning kommutaatorsõlme loomisel kasutati tarkvara tegemiseks C keelt. Riistvaralise realisatsiooni elluviimisel mikrokontrollereid, niiskusandureid ning raadiomooduleid. Andmekoguja realiseeriti Raspberry Pi miniarvutil, millel on kasutatud Raspbian Jesse Lite operatsioonisüsteemi ning LAMP tarkvarakomplekti.

Töö raskust tõstis süsteemide vahelise suhtluse viisi väljamõtlemine ning pooldupleks raadiomoodulitega selle realiseerimine. Keerukas oli ka harjumuspärasest arendusprotsesst erineva kasutamine andurseadme tarkvara loomisel.

Loodud prototüüpi on võimalik arendada edasi nii tarkvaraliselt kui ka riistvaraliselt. Andurseadme riistvaralise lahenduse arendamisel on kasulik süsteemi mõõtmete vähendamiseks luua eraldi trükkplaat, eemaldamaks süsteemist mikrokontrolleri poolt kasutatava arendusplaadi vajaduse.

Tarkvaralise poolelt on süsteemi kasulik lisada keerukam suhtlusprotokoll, mis kontrolliks andmepakettide kohalejõudmist. Veebisaidi edasiarendamisel ning suhtluse krüpteerimisel võimaldaks andurseadmele konfigureerimispakettide saatmist, muutest süsteemi kasutust lihtsamaks. Tarkvaralisel on võimalik süsteemi veel edasi arendada, kohandades see mõne targa kodu standardiseeritud lahendusse.

6 Kasutatud kirjandus

[1] P. H. Tarange, R. G. Mevekari ja P. A. Shinde, „Web based Automatic Irrigation System using wireless sensor network and Embedded linux board,“ %1 *International Conference on Circuit, Power and Computing Technologies [ICCPCT]*, 2015.

[2] „MSP430G2x53, MSP430G2x13 Mixed Signal Microcontroller (Rev. J),“ mai 2013. [Võrgumaterjal]. Available: http://www.ti.com/lit/ds/symlink/msp430g2553.pdf. [Kasutatud 21 mai 2017].

[3] „MSP430x2xx Family User's Guide,“ [Võrgumaterjal]. Available: http://www.ti.com/lit/ug/slau144j/slau144j.pdf. [Kasutatud 22. mai 2017].

[4] „YL-38 Soil Moisture Sensor With Arduino,“ [Võrgumaterjal]. Available: http://www.diyspacepk.com/yl-38-soil-moisture-sensor-with-arduino/. [Kasutatud 22. mai 2017].

[5] „ESPRESSIF SMART CONNECTIVITY PLATFORM: ESP8266,“ Espressif Systems, 12 oktoober 2013. [Võrgumaterjal]. Available: https://nurdspace.nl/images/e/e0/ESP8266_Specifications_English.pdf. [Kasutatud 21 mai 2017].

[6] M. Leier ja M. Gorev, „868MHz wireless connection,“ TTÜ Arvutitehnika Instituut, [Võrgumaterjal]. Available: http://ati.ttu.ee/embsys/index.php/868MHz_wireless_connection. [Kasutatud 21 mai 2017].

[7] „MSP-EXP430G2 Development Kit User's guide,“ [Võrgumaterjal]. Available: http://www.ti.com/lit/ug/slau318g/slau318g.pdf. [Kasutatud 21 mai 2017].

[8] J. W. Grenning, Test-Driven Development for Embedded C, Dallas: The Pragmatic Programmers, LLC, 2011.

[9] „Raspberry Pi Documentation,“ Raspberry Pi Foundation, [Võrgumaterjal]. Available: https://www.raspberrypi.org/documentation/. [Kasutatud 21 mai 2017].

[10] „Esquematico YL-38,“ [Võrgumaterjal]. Available: https://d29h7ql7qnxkqx.cloudfront.net/pix/ebayimage2012/26215-8.jpg. [Kasutatud 22. mai 2017].