jamk.fi

Älykotijärjestelmä

Mikrokontrolleriohjattu älykoti

Thomas Kuronen

Opinnäytetyö
Marraskuu 2017
Tekniikan ja liikenteen ala
Insinööri (AMK), automaatiotekniikan tutkinto-ohjelma

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Kuronen, Thomas	Opinnäytetyö, AMK	Marraskuu 2017
	Sivumäärä 81	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: x

Työn nimi
Älykotijärjestelmä
Mikrokontrolleriohjattu älykoti

Tutkinto-ohjelma
Insinööri (AMK), automaatiotekniikan tutkinto-ohjelma

Työn ohjaaja(t)
Juho Riekkinen, Olli Väänänen

Toimeksiantaja(t)
Nodeon

Tiivistelmä

Opinnäytetyön tavoitteena oli selvittää kotiautomaatiojärjestelmän toteutusmahdollisuutta mikrokontrollereilla ja muilla halvoilla elektroniikkakomponenteilla. Mikrokontrolleripohjaisia kotiautomaatiojärjestelmiä ei vielä ole markkinoilla, joten työssä pyrittiin selvittämään ja toteuttamaan mahdollisimman laajasti järjestelmän mahdollisuuksia.

Älykotijärjestelmään halutuista toiminnoista tehtiin toimintakuvaukset, joiden pohjalta laadittiin laiteluettelo. Komponentit asennettiin talon, autotallin sekä piha-alueen pienoismalliin, missä työtä oli helppo simuloida. Älynä käytettiin Arduino-tuoteperheeseen kuuluvia mikrokontrollereita ja lisäksi ESP8266-pohjaista moduulia. Ohjelmakoodi kirjoitettiin laitteille Arduino IDE -ohjelmalla.

Opinnäytetyön tuloksena saatiin toimiva älykotijärjestelmä ja sitä varten rakennettu simulointiympäristö. Järjestelmä on halpa ja se on sovellettavissa hyvin erilaisiin kohteisiin. Toimeksiantaja sai työstä käyttöönsä piirikaaviot, ohjelmakoodit ja laiteluettelot. Näiden avulla kotiautomaatiojärjestelmän rakentaminen, tarkempi tutkiminen ja soveltaminen ovat mahdollisia.

Valmis älykotijärjestelmä täyttää työlle asetetut kriteerit ja vaatimukset. Järjestelmä on toteutettu halvoilla komponenteilla, se on sovellettavissa erilaisiin kohteisiin ja siinä on käytetty paljon erilaisia laitteita ja väylätekniikoita.

Avainsanat (asiasanat)
Arduino, mikrokontrolleri, älykotijärjestelmä, kotiautomaatiojärjestelmä, kotiautomaatio, väylätekniikka, ESP8266, pilvipalvelu, IoT.

Muut tiedot

jamk.fi

Description

Author(s)	Type of publication	Date
Kuronen, Thomas	Bachelor's thesis	November 2017
		Language of publication: Finnish
	Number of pages: 81	Permission for web publication: x

Title of publication

System of smart home

Smart home controlled by microcontroller

Degree programme

Automation engineering

Supervisor(s)

Riekkinen Juho, Väänänen Olli

Assigned by

Nodeon

Abstract

The aim of the thesis was to find out how to control the smart home by microcontroller and other cheap electrical components. There are not too many smart homes controlled by microcontroller on sale; hence, the aim of the thesis was to find out if it is possible to implement this kind of system.

A general description of the desired functions for the smart home was made followed by a catalogue of devices. The components were installed to a miniature smart home, garage and yard. The system was easy to simulate in the miniature format. The smart devices used were Arduino microcontrollers and ESP8266-module. The code was written with Arduino IDE -program.

The thesis result in a functional smart home and miniature. The system is cheap and it is applicable to different uses. The catalogue of devices, code and circuit diagram were given to the company that had assigned the project. Using the catalogue devices, code and circuit diagram it is possible to build the system of smart home, examine and apply it into practice.

The finished smart home system meets the set criteria and requirements. The smart home system is achieved using cheap components and it can be applied to different uses with various devices and bus technologies.

Keywords/tags (subjects)

Arduino, microcontroller, smart home, home automation, ESP8266, bus technology, IoT, cloud service.

Miscellaneous

Sisältö

Kuviot

Taulukot

Käsitteet

Arduino	Arduino on mikrokontrolleripohjainen kehitysalusta, jolla voidaan ohjata erinäisiä laitteita.
ArduinoIDE	Arduino-laitteille tarkoitettu kehitysympäristö.
Blynk	Mobiilikäyttöinen ohjelmisto.
ESP8266	Wifi-moduuli, jota voidaan käyttää kehitysalustoissa.
GND	GND on lyhenne sanasta ground.
HIGH-tila	Pinnin ollessa HIGH-tilassa sen jännite on yleensä 3 VDC- 5 VDC. Tämä tarkoittaa sitä, että laite on tunnistanut jotakin. HIGH-tila tarkoittaa loogista ykköstä.
IFTTT	Lyhenne sanoista If This Then That. IFTT on palvelu, joka mahdollistaa kytkemään toisiinsa erilaisia toimintoja.
Input	Syöte, joka voi olla esimerkiksi lämpötila-arvo tai tilatieto oven asennosta.
I2C	Tiedonsiirtoväylä, jota voidaan käyttää eri laitteiden väliseen kommunikointiin.
IO	Lyhenne sanoista Input ja Output. Input tarkoittaa syötettä ja output tarkoittaa tulostetta.
IoT	IoT- lyhenne tulee sanoista Internet Of Things.
LCD	Lyhenne sanoista Liquid Crystal Display eli nestekidenäyttö, jolla voidaan näyttää käyttäjälle tietoa.
LOW-tila	Pinnin ollessa LOW-tilassa sen jännite on 0 VDC. Se tarkoittaa yleisimmin sitä, että laite on pois päältä tai anturi ei ole tunnistanut mitään. LOW-tila tarkoittaa loogista nollaa.
Mikrokontrolleri	Mikrokontrolleri on periaatteessa pieni tietokone, jossa on mikroprosessori sekä muisti- ja liityntälohkoja.

NodeMCU	Kehitysalusta, jossa on sisäänrakennettu Wifi-moduuli.
NRF24L01	Lähetin-vastaanotin, jolla voidaan siirtää tietoa langattomasti 2,4 GHz:n taajuudella.
Output	Tuloste, jolla säädetään esimerkiksi moottorin pyörimisnopeutta.
PWM	Pulse Width Modulation, suomeksi pulssinleveysmodulaatio. Tämä on modulointitapa, jossa lähtevää ohjausjännitettä säädetään muuttamalla pulssisuhdetta.
Serial	Serial on Arduinon käyttämä väylätekniikka, jota käytetään Arduinon ja muiden laitteiden väliseen kommunikointiin.
Thinger	Thinger on pilvipalvelu, jonka avulla voidaan ohjata laitteita tai lukea tietoja etänä.
VCC/V+	VCC ja V+ lyhenteillä merkitään pinnit, joihin kytketään positiivinen jännitteensyöttö.

1 Johdanto

1.1 Opinnäytetyön lähtökohdat

Opinnäytetyön aiheeksi valittiin kotiautomaation tekeminen talon ja sen ympäristön pienoismalliin. Pienoismalli koostuu talosta, autotallista sekä piha-alueesta, jossa on muun muassa avautuva portti. Työn nimeksi muodostui osuvasti älykotijärjestelmä.

Markkinoilla ei ole vielä halpoja ratkaisuja toteuttaa älykotijärjestelmää, eikä ohjeita tämän kokoluokan järjestelmään ole saatavilla. Tämä herätti mielenkiinnon siitä, onko halvoilla ja rajoittuneilla kehitysalustoilla mahdollista tehdä automaatiojärjestelmä, joka vastaisi nykytalon automaation tarpeita. Työssä käytettiin Arduino- tuoteperheeseen kuuluvia kehitysalustoja, jotka ovat yksinkertaisia ja halpoja.

Toimeksiantajana työlle toimi Nodeon Oy. Nodeon on moderni teknologiatoimittaja, joka on perustettu vuonna 2005. Yksi tärkeimmistä osaamisalueista on teollisen internetin erilaiset ratkaisut ja erityisesti kriittisissä kohteissa Nodeon on vahvimmillaan. Kriittisiä kohteita ovat esimerkiksi erilaiset infrastruktuurikohteet, joissa järjestelmien varassa on jopa ihmishenkiä. Erityisosaaminen kohdistuukin teollisen internetin puolelta älyliikenteeseen ja niiden ratkaisuihin. (Nodeon, yritys lyhyesti 2017.)

Toimeksiantajan vaatimuksia työlle oli käyttää useita eri laitteita ja väylätekniikoita. Lisäksi työssä tuli selvittää pilvipalveluiden mahdollisuuksia ja niiden hyödyntämistä. Työn aikana valmistuneet toimintakuvaukset käytiin yhdessä toimeksiantajan kanssa lävitse ja ne hyväksytettiin.

Kyseisen aiheen valitseminen oli helppoa, koska olen harrastanut omatoimisesti elektroniikalla rakentamista jo pidemmän ajan. Lisäksi vastaavanlainen projekti olisi tarkoitus toteuttaa myös oikeaan ympäristöön tulevaisuudessa.

1.2 Opinnäytetyön tavoitteet ja rajaus

Opinnäytetyössä oli paljon erilaisia tavoitteita, joista tärkein oli saada selville, onnistuuko tällaisen kokonaisuuden tekeminen halvoilla ja yksinkertaisilla kehitysalus-

toilla. Kokonaisuudella tarkoitetaan taloon, autotalliin sekä piha-alueeseen rakennettua älykotijärjestelmää. Valmiin kotiautomaation piti olla sellainen, että sitä voitaisiin käyttää oikeassa ympäristössä.

Opinnäytetyössä käytettiin kartoittavaa tutkimusotetta. Kartoittavan tutkimusotteen tarkoituksena on selvittää ilmiöitä, joita ei tunneta kovin laajasti sekä etsiä uusia näkökulmia. Lisäksi kartoittavan tutkimusotteen tarkoituksena on kehittää aiheesta hypoteeseja. (Opinnäytetyön suunnitelman laatiminen n.d.) Opinnäytetyössä selvitettäviä asioita olivat esimerkiksi mikrokontrollerin muistin riittävyys, tehon rajoitteellisuus sekä erilaisten tiedonsiirtoväylien ja laitteiden yhteensopivuus. Muita tavoitteita oli valita komponentit siten, että kokonaisuudesta tulisi mahdollisimman halpa. Tämän vuoksi valmistuvan kotiautomaatiojärjestelmän voisi kuka tahansa asentaa esimerkiksi omaan taloon tai varastorakennukseen ilman suuria kustannuksia. Lisäksi rakennettavassa pienoismallissa olevien autotallin sekä portin ohjaukset tuli tapahtua erillisillä järjestelmillä, jotta maanalaista kaapelivetoa ei tarvinnut tehdä.

Työ oli siinä mielessä mielenkiintoinen, että siinä käytettävistä laitteista ja niiden toiminnasta on internetissä ja kirjoissa hyvin paljon erilaisia projekteja ja dokumentteja, mutta kotiautomaatiossa käytettävien laitteiden ja antureiden muodostamaa järjestelmää ei ole dokumentoitu tai tehty tällä tasolla. Työhön sisällytettiin yleisimpiä kotiautomaatiossa käytettäviä toimintoja, kuten valoisuuden ja lämpötilan mittausta, savun ja läsnäolon tunnistusta sekä erilaisien laitteiden ohjausta, kuten tuuletin ja askelmoottorit portilla ja autotallin ovessa. Lisäksi talon valaistusta ohjataan älykkäästi. Työssä ei otettu huomioon kotiautomaation monimutkaisempia järjestelmiä, kuten esimerkiksi ilmastoinnin ohjausta ja talon paineistusta lämpötilan, ilmanpaineen sekä kosteuden mukaan.

Työn lopputuloksena on kotiautomaatiojärjestelmä, joka on markkinoilla oleviin järjestelmiin verrattuna halpa ja järjestelmä joka vastaa kotiautomaatiolle asetettuja tarpeita. Näitä tarpeita ovat toimintakuvauksessa esitetyt toiminnallisuudet. Järjestelmä myös perustuu avoimeen lähdekoodiin, joten sen muokkaaminen ja laajentaminen omien tarpeiden mukaisesti onnistuu helposti. Järjestelmän lisäksi lopputuloksena on kodin ja sen ympäristön pienoismalli, jonka avulla järjestelmää pystyy testaamaan ja tulokset ovat havainnollistavia. Opinnäytetyö on onnistunut ja täyttänyt tavoitteet siinä vaiheessa, kun järjestelmä toimii sille määriteltyjen testauskriteereiden

mukaisesti. Lopputuloksena ei tule kuitenkaan olemaan kaupallinen tuote, vaan toimiva järjestelmä mitä voidaan kehittää myöhemmin myytäväksi tuotteeksi.

1.3 Tiedonhankinta

Tietoperustana toimi osittain oma aiemmin hankittu osaaminen elektroniikasta, koodaamisesta ja automaatiojärjestelmistä. Etsin koodin kirjoittamiseen tietoa kansainvälisiltä elektroniikkaharrastajien internetsivuilta, joissa on kootusti erilaisia projekteja. Näitä sivustoja ovat muun muassa instructables.com, hackter.io sekä Arduinon omat internetsivut, joilla on paljon yksityiskohtaista tietoa. Lisäksi käytin apuna alan kirjallisuutta ja opetusmateriaalia.

Työssä käytetyillä komponenteilla ja menetelmillä tätä järjestelmää voitaisiin tulevaisuudessa käyttää kotiautomaation lisäksi hyvinkin erilaisissa kohteissa. Kohteita voisivat olla esimerkiksi tekoaltaan kalanruokinta-automaatti, pihakanalan automaattinen ruoka- ja vesiautomaatti, kesämökin automatisointi ja sähkönkulutuksen minimointi sekä erilaisten laitteiden ohjaus ja seuranta. Laitteet, joita työssä käytettiin, ovat edullisia ja kaikkien saatavilla, joten pienempienkin ympäristöjen automatisointi on näillä komponenteilla helppoa ja järkevää toteuttaa. Lisäksi koodi on helposti muokattavissa ja koodin kirjoittamiseen löytyy paljon esimerkkejä ja apua.

Markkinoilta löytyy paljon valmiita ratkaisuja kotiautomaation toteuttamiseen. Ala on kasvanut viime vuosina räjähdysmäisesti ja kasvaa tulevinakin vuosina, minkä seurauksena käytettävissä on hyvin paljon eri käyttötarkoituksiin soveltuvia laitteita. Näissä kaikissa on kuitenkin muutamia yhteisiä piirteitä, joihin yritän opinnäytetyössä hakea ratkaisuja. Valmiit järjestelmät ovat hyvin kalliita sekä useasti myös rajoitteellisia. Tämä tarkoittaa sitä, että järjestelmät eivät ole räätälöityjä tiettyyn tarpeeseen, vaan ovat hyvin yleiskäyttöisiä. Laajennus- ja lisäosat ovat tehneet kuitenkin järjestelmistä paremmin soveltuvia. Jokainen lisätty komponentti järjestelmään maksaa kuitenkin todella paljon. Opinnäytetyössä kiinnitettiin huomiota kokonaisuuden hintaan sekä järjestelmän muokattavuuteen. Tästä syystä en ryhtynyt tekemään kotiautomaatiota valmiilla paketeilla, vaan yritin keksiä ratkaisun järjestelmän toteuttamiseen halvoilla elektroniikkakomponenteilla.

2 Internet Of Things

IoT tulee sanoista Internet Of Things, joka tarkoittaa vapaasti suomennettuna esineiden internetiä. Tämä tarkoittaa käytännössä sitä, että jokin laite on kytketty internetverkkoon ja sitä voidaan ohjata tai siitä voidaan lukea mittaustietoa etänä. IoT voi kuulostaa vieraalta termiltä, mutta useimmille sana älykello on kuitenkin tuttu. Älykellot ovat nykyään yhä enemmän IoT-laitteita, joten esimerkiksi urheilusuorituksista kerätty data tallennetaan automaattisesti verkossa olevalle palvelulle ja sitä voidaan myöhemmin tarkastella.

IoT:ssä on käytännössä kyse älyn lisäämisestä laitteisiin tai tuotteisiin. Nämä ovat yleensä myös yhteydessä internetiin. Voidaankin puhua siis älykkäistä tietolähteistä, jotka mahdollistavat täysin uudenlaisien liiketoimien kehittämisen. Keskiössä on lopulta kuitenkin asiakas tai käyttäjä. (Taanila 2016.)

Termiin liittyy myös ominaisuuksia, joista ei mielellään puhuta. Älykkäiden esineiden käyttö voi vaatia henkilökohtaisen käyttäjäprofiilin käyttämistä. Profiilit keräävät erilaista tietoa käyttäjästä, kuten viestintää, sijaintitietoja ja muita tekemisiä. Mahdollisuuksia on rajattomasti mikä johtaa siihen, että jokin arkinen laite kotona tietää käyttäjästä enemmän kuin käyttäjä itse. Laitteen lisäksi käyttäjän tiedot voi saada kuka tahansa, eikä tätä voida estää ilman laitteen käytön lopettamista. (Ryynänen 2017.)

3 Älykoti

Nykypäivänä lähes jokainen on kuullut sanan älykoti tai taloautomaatio. Jokaisella on omanlaisensa mielikuva älykodista, mutta mikä se todellisuudessa on? Älykoti muodostuu useista laitteista ja antureista ja tätä kokonaisuutta kutsutaan automaatiojärjestelmäksi. Kun automaatiojärjestelmä on asennettu taloon, sitä kutsutaan taloautomaatioksi.

Taloautomaatiolla tarkoitetaan yleisesti sellaisia ratkaisuja, joiden avulla voidaan ohjata ja seurata kodin laitteita ja järjestelmiä sekä hyödyntää näitä mahdollisimman kustannustehokkaasti. Yksinkertaisena esimerkkinä voidaan käyttää talon valaistusta. Sen sijaan, että jokaisessa huoneessa olisi vain tavalliset valokatkaisijat, niiden lisäksi olisi erilliset painikkeet erilaisia tilanteita varten. Aamu-painikkeesta voisi syttyä valot

esimerkiksi portaikkoon, keittiöön ja pihalle. Lisäksi voitaisiin ohjelmoida eri lämpötila-asetukset huoneisiin. Näitä valaistus-, lämpötila- ja paljon muita vaihtoehtoja asukas voisi määritellä haluamallaan tavalla. Järjestelmään voisi myös liittää palo- ja murtohälytykset, ovipuhelinjärjestelmän, turvakameran, kosteusvahdin, kodinteknikkalaitteet, ovien lukitukset sekä sähkötoimiset verhot ja markiisit. Auton lämmityksen voisi asettaa automaattiseksi tai sen voisi kytkeä heti herättyään makuuhuoneesta päälle. Kotona/poissa -kytkin lukitsee itsestään ovet, sulkee vedet, aktivoi murtohälytyksen sekä katkaisee sähköt tietyistä paikoista, kuten esimerkiksi liedestä. Lisäksi jos kotoa ollaan poissa pidemmän aikaa, lämpötila ja ilmastointi lasketaan tietylle tasolle, jolloin sähköä kuluu vähemmän. Jos kytkimen painaminen on lähtiessä unohtunut, poissa-tilan voi asettaa päälle myös tekstiviestillä. (Berghäll 2012.)

Koko järjestelmää voidaan ohjata kodissa sijaitsevalta kosketusnäytöltä, joka on yleensä kiinteästi eteisessä. Lisäksi sitä voidaan ohjata erillisellä ohjelmalla, joka asennetaan tablettiin tai tietokoneeseen. Lisäksi puhelinsovelluksella tai tekstiviestillä saadaan kytkettyä toimintoja päälle tai pois päältä. (Berghäll 2012.)

Pääasiassa järjestelmän tarkoitus on helpottaa elämää. Yhdellä napin painalluksella voidaan suorittaa useita eri toimintoja. Lisäksi järjestelmä huolehtii automaattisesti eri huoneiden ilmanvaihdon ja lämpötilan säätämistä optimaalisiksi. Järjestelmä seuraa myöskin eri laitteiden kuntoa ja antaa asukkaalle tietoa siitä, milloin esimerkiksi ilmanvaihtokoneen suodattimet tulisi vaihtaa tai milloin jokin laite on huollon tarpeessa. Tämä parantaa itsessään jo kodin turvallisuutta, mutta lisäksi automaatiojärjestelmä tarkkailee laitteita esimerkiksi vesivahinkojen varalta. Jos järjestelmä huomaa jonkin laitteen vuotavan, se sulkee tälle menevän vesiputken venttiilin ja estää vesivahingon. Lisäksi palo- ja murtohälytykset suojaavat omaisuutta. Murtohälytyksen sattuessa talon voi ohjelmoida vaikkapa vilkuttamaan valoja sekä lähettämään tapahtuneesta viestiä eteenpäin. (Berghäll 2012.)

Taloautomaatiojärjestelmän avulla voidaan ohjata esimerkiksi vapaa-ajankodille tai mökille lämmityksen ja ulkovalot päälle hyvissä ajoin ennen perille saapumista. Tällöin vapaa-ajankoti on valmiina odottamassa tulijoita. Lisäksi poissa ollessa vapaa-ajankotia voidaan pitää asutun näköisenä sytyttämällä valoja ja aukaisemalla verhoja automaattisesti. Murroista, tulipaloista sekä ovella kävijöistä saadaan myös reaaliaikaista tietoa ja kuvia. (Berghäll 2012.)

Taloautomaatiolla voidaan toteuttaa hienoja ja monimutkaisia asioita, mutta järjestelmän kääntöpuoli on sen kallis hinta. Pelkästään ohjausjärjestelmän kytkin maksaa 100-600 euroa. Lämmityksen, ilmanvaihdon ja valaistuksen ohjaukset sisältävä perusjärjestelmä noin 200 neliön omakotitaloon varovasti arvioituna maksaa 10 000 -15 000 euroa ilman asennustöitä. (Berghäll 2012.)

4 Älykotijärjestelmän suunnittelu

Älykotijärjestelmän suunnitteluun kuuluu yleisen toimintakuvauksen tekeminen, laitekohtaisen toimintakuvauksen tekeminen, laiteluettelon tekeminen sekä laitteiden topologiakuvauksen tekeminen. Yleinen toimintakuvaus kertoo selkeästi ja yleisesti älykodin toiminnallisuudet, eli sitä mitä järjestelmän tulisi tehdä. Laitekohtainen toimintakuvaus on yleiseen toimintakuvaukseen verrattuna tarkempi ja se kertoo laitekohtaisesti, mikä niiden rooli on järjestelmässä. Laiteluettelo kertoo laitteet mitä järjestelmä vaatii ja millä se voidaan toteuttaa. Laitteiden välinen topologiakuvaus kertoo eri laitteiden välisistä viestintäkanavista ja väylätekniikoista. Eli topologia kertoo miten eri laitteet kommunikoivat keskenään.

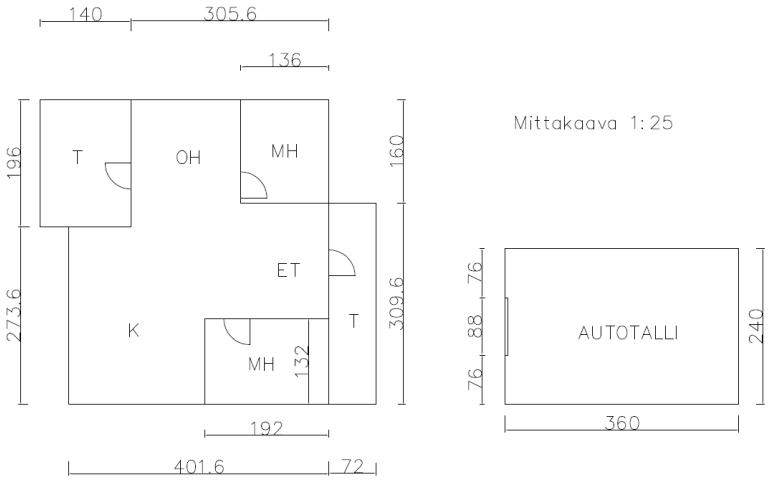
4.1 Toimintakuvaus

Toimintakuvaus on yleinen kuvaus siitä, mitä järjestelmällä halutaan saada aikaiseksi. Kuvauksessa kerrotaan järjestelmän toiminnasta kokonaisuutena, siitä kuinka eri toiminnot toimivat tietynlaisissa tilanteissa, mutta ei kuitenkaan kerrota toiminnallisuuksista yksityiskohtaisesti.

Toimintakuvaus tulee yleensä toimeksiantajalta, mutta tässä opinnäytetyössä työhön kuului tehdä myös toimintakuvaus ja hyväksyttää se toimeksiantajalla. Pienoismallin eri osa-alueet jaoteltiin erilleen ja jokaiselle tehtiin oma toimintakuvaus. Näitä osa-alueita oli itse pienoismallin rakentaminen, talon toiminnallisuudet, autotallin toiminnallisuudet, piha-alueen toiminnallisuudet sekä käyttöpaneelin toiminnallisuudet.

4.1.1 Pienoismallin rakenne

Pienoismallin talo ja autotalli tehtiin oikean talon pohjapiirustuksesta (ks. kuvio 1). Pienoismallin koon tuli olla sellainen, että sitä pystyy helposti liikuttamaan normaalilevyisistä ovista. Pienoismallin koko oikean talon pohjapiirustukseen suhteutettuna on 1:25. Talon sekä autotallin lisäksi pienoismallin ympärille tuli aita ja portti.



Kuvio 1. Talon ja autotallin pohjakuva

4.1.2 Yleiset toimintakuvaukset

Talon toiminnallisuudet

Valaistuksia ohjataan käyttöpaneelilta, jossa on kytkimet kaikille valaistusryhmille. Lisäksi terassien valot ohjautuvat automaattisesti hämärätunnistimen avulla. Kaikki valot ovat pois päältä, kun poissa-tila on aktiivinen. Optiona on, että eteisen valot ohjautuvat kytkimien lisäksi automaattisesti. Ohjauksen valo saa liiketunnistimelta.

Pienoismallin jäähdytys tapahtuu tietokoneen tuulettimella. Tuuletin on PID-säätimellä ohjattu ja sitä ohjataan lämpötila-antureiden avulla. Käyttäjä pystyy säätämään lämpötilan asetusarvoa käyttöpaneelilta.

Kotona ja poissa -tilat aktivoituvat, kun käyttäjä syöttää käyttöpaneelilla sijaitsevaan näppäinpaneelille oikean PIN-koodin. Poissa-tilan ollessa päällä varashälytin kytkeytyy automaattisesti päälle, ja jos sisällä tapahtuu liikettä tai jos ulko-ovet avautuvat, siitä aiheutuu hälytys. Hälytys aiheuttaa sireenin aktivoitumisen. Hälytyksen aktivoiduttua lähetetään automaattisesti sähköposti käyttäjälle hälytyksestä.

Tulipalo havaitaan savunilmaisimella. Tulipalo aiheuttaa sireenin aktivoitumisen sekä siitä lähetetään automaattisesti sähköposti käyttäjälle.

Autotallin toiminnallisuudet

Autotallin valaistusta ohjataan käyttöpaneelilta. Autotallin ovi aukeaa ja sulkeutuu askelmoottorilla. Askelmoottoria ohjataan kaukosäätimellä, joka rakennetaan itse 2,4 GHz:n radiomoduulitekniikkaa hyödyntäen.

Piha-alueen toiminnallisuudet

Terassien valaistusta ohjataan käyttöpaneelilta. Lisäksi valot ohjautuvat automaattisesti hämärätunnistimen avulla. Portti aukeaa ja sulkeutuu askelmoottorilla. Askelmoottoria ohjataan kaukosäätimellä autotallin oven tapaisesti. Portin välille asennetaan anturi, joka tunnistaa liikkeen portin sulkeutuessa. Jos anturi havaitsee jonkin esineen portin sulkeutuessa, portti pysähtyy ja avautuu uudelleen.

Käyttöpaneeli

Käyttöpaneelilta käyttäjä voi ohjata talon ja ympäristön valaistusta. Lisäksi käyttäjä voi asettaa taloon halutun lämpötilan. Talossa ei ole lämmitystä, mutta jäähdytys kytkeytyy päälle, jos asetusarvo on pienempi kuin talon lämpötila.

Käyttäjä voi kytkeä kotona ja poissa-tilat päälle näppäinpaneelin avulla. Lisäksi käyttöpaneelissa on LCD-näyttö, josta näkyy esimerkiksi lämpötila-arvoja, tuulettimen pyörimisnopeus prosentteina ja kotona ja poissa-tilojen aktivoinnit.

Muut toiminnot

Työssä pyrittiin käyttämään paljon erilaisia tekniikoita samalla pohtien sitä, miten lopputulos olisi sellainen, että sen voisi laajentaa oikeaan ympäristöön. Esimerkiksi eri rakennuksien välisiä kaapeleita ei tulisi olla ja laitteiden pitäisi toimia pienoismal-

lin lisäksi oikeassa ympäristössä. Lisäksi pienoismallista tulisi olla yhteys ulkoverkkoon. Kotiautomaation tietoja pitäisi pystyä lukemaan esimerkiksi pilvipalvelun kautta.

4.2 Laitteiden toiminnallisuudet

Laitteiden toimintakuvauksessa kerrotaan yleiseen toimintakuvaukseen verrattuna tarkemmin jokaisen laitteen toimintaa tietynlaisissa tilanteissa. Laitteiden toimintakuvaus tehtiin yleisen toimintakuvauksen pohjalta, ja samalla piti miettiä laitehankintoja sekä päivittää laiteluetteloa (ks. liite 1). Laitteiden toimintakuvaus ja laiteluettelo valmistuivat työssä hyvin pitkälle yhtä aikaa.

4.2.1 Talon automaatiolaitteiden toiminnallisuudet

Valaistus

Eteisen valaistus (LED23) kytkeytyy päälle, kun kotona-tila on aktiivinen, liiketunnistin 1 (LOI21) havaitsee liikettä ja etuovi aukeaa. Tämän jälkeen eteisen valaistus sammuu itsestään 15 sekunnin kuluttua. Lisäksi valo kytkeytyy päälle, kun käännetään kytkin 4 (SW23) ON-asentoon. Poissa-tilan aktivoiduttua valaistus sammuu.

Olohuoneen valaistus (LED24) kytkeytyy päälle, kun kotona-tila on aktiivinen ja kytkin 5 (SW24) käännetään on-asentoon. Poissa-tilan aktivoiduttua valaistus sammuu.

Keittiön valaistus 6 (LED25) kytkeytyy päälle, kun kotona-tila on aktiivinen ja kytkin 6 (SW25) käännetään ON-asentoon. Poissa tilan aktivoiduttua valaistus sammuu.

Makuuhuoneiden valaistukset 1 ja 2 (LED20 ja LED21) kytkeytyvät päälle, kun kotona-tila on aktiivinen ja käännetään kytkimet 1 tai 2 (SW20/SW21) ON-asentoon. Poissa tilan aktivoiduttua valaistus sammuu.

Terassien valaistukset 7 ja 8 (LED26 ja LED27) kytkeytyvät päälle, kun kotona-tila on aktiivinen ja hämärätunnistin (XS20) muuttuu aktiiviseksi. Lisäksi valaistukset kytkeytyvät päälle, kun käännetään kytkimet 7 tai 8 (SW26/SW27) ON-asentoon. Poissa tilan aktivoiduttua valaistus sammuu.

Liiketunnistin

Havaitessaan liikettä, olohuoneessa sijaitseva liiketunnistin (LOI21) sytyttää eteisen valot päälle, kun kotona-tila on aktiivinen. Poissa-tilan ollessa aktiivinen ilmoitetaan liikkeestä sähköpostilla käyttäjälle.

Ovikytkin

Etuoven ovikytkin (IA30) aiheuttaa summerin (SIR20) aktivoitumisen sekä ilmoituksen varkaista käyttäjälle sähköpostilla, mikäli poissa-tila on aktiivinen.

Takaoven ovikytkin (IA31) aiheuttaa summerin (SIR20) aktivoitumisen sekä ilmoituksen varkaista käyttäjälle sähköpostilla, mikäli poissa-tila on aktiivinen.

Savunilmaisin

Olohuoneessa sijaitseva savunilmaisin (PI20) aiheuttaa summerin (SIR20) aktivoitumisen, mikäli havaitsee savua. Lisäksi tulipalosta ilmoitetaan käyttäjälle sähköpostilla.

Hämärätunnistin

Ulkoseinässä sijaitseva hämärätunnistin (XS20) aktivoituu, kun ulkona tulee pimeää. Lisäksi hämärätunnistin ohjaa terassien valaistusta.

Lämpötilamittaus

Lämpötilamittaus 1 (TE20) eteisessä ja lämpötilamittaus 2 (TE21) olohuoneessa mittaavat ilman lämpötilaa.

Tuuletin

Olohuoneessa sijaitseva tuuletin (M20) jäähdyttää ilmaa puhaltamalla lämmintä sisäilmaa ulos. Pyörimisnopeus riippuu lämpötilamittauksien TE20 ja TE21 arvoista. Pyörimisnopeus lasketaan PID-säätimen avulla.

Summeri

Summeri (SIR20) olohuoneessa hälyttää, kun järjestelmä on havainnut tulipalon tai varkaita.

4.2.2 Portin automaatiolaitteiden toiminnallisuudet

Askelmoottori

Porttia liikuttava askelmoottori (M40) liikuttaa porttia auki ja kiinni. Moottori saa liikkumiskäskyt kaukosäätimeltä. Mikäli anturi (P40) havaitsee esteen portin edessä sen sulkeutuessa, portti pysähtyy ja avautuu uudestaan.

Esteen indikointi

Portin edessä oleva anturi (P40) havaitsee portin välissä olevan esteen. Anturi toimii kytkimen tavoin.

4.2.3 Autotallin automaatiolaitteiden toiminnallisuudet

Valaistus

Autotallin valaistus 10 (LED60) kytkeytyy päälle, kun käännetään kytkin 9 (SW28) ON-asentoon.

Askelmoottori

Autotallin ovea liikuttava askelmoottori (M60) liikuttaa ovea auki ja kiinni. Moottori saa liikkumiskäskyt kaukosäätimeltä.

4.2.4 Paneelin automaatiolaitteiden toiminnallisuudet

Kytkimet kytkevät valaistuksia päälle ja pois päältä.

Näppäimistö

Kun käyttäjä kirjoittaa oikean PIN-koodin (11235#) näppäinpaneelilla (NP120), kotona tai poissa-tila kytkeytyy päälle. Jos käyttäjä kirjoittaa väärän koodin tai painaa *-merkkiä, muisti nollataan. Koodin ollessa oikea ilmoitetaan näytöllä (LCD20) ”Kotona-tila aktivoitunut”- tai ”Poissa-tila aktivoitunut”-tekstillä.

Potentiometri

Potentiometrillä (RC20) käyttäjä voi säätää talon lämpötilan asetusarvoa.

Näyttö

LCD-näyttö (LCD20) näyttää käyttäjälle kotona- ja poissa -tilojen aktivoinnit, lämpötilan asetusarvon, lämpötilat olohuoneessa sekä eteisessä, moottorin pyörimisnopeuden prosentteina, tulipalon sekä murtohälytyksen. Näytön alapuolella oleva painonappi 3 (P20) ohjaa näytön valikkoa. Nappia painamalla saadaan valikossa oleva seuraava tieto näkyville.

4.2.5 Kaukosäätimen automaatiolaitteiden toiminnallisuudet

Tilanvalitsin

Kaukosäätimessä on vaihtokoskettimilla varustettu 3-asentoinen katkaisija (SW100). Kun katkaisija on 1-asennossa, ohjataan ulkoporttia. Kun katkaisija on 2-asennossa, ohjataan autotallin ovea.

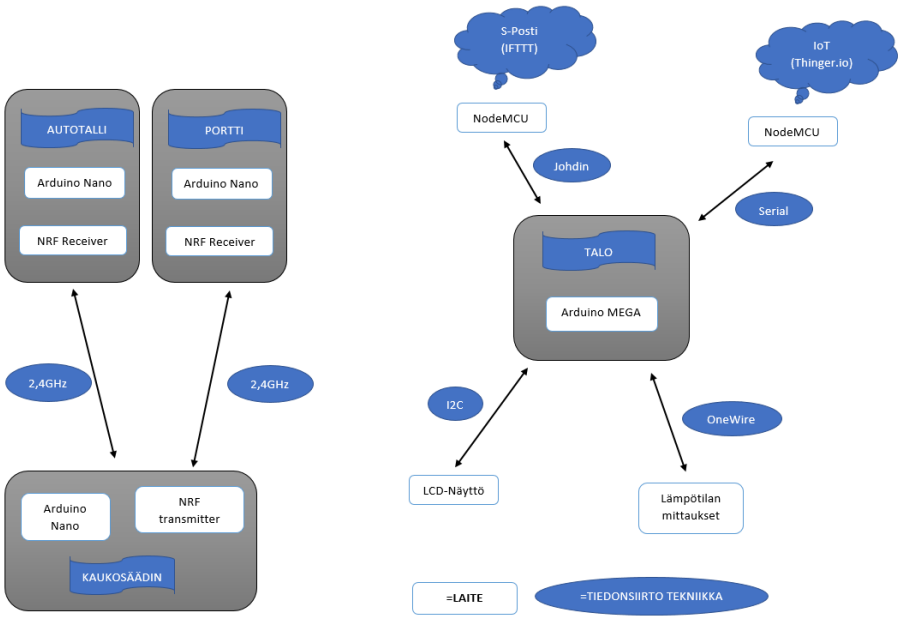
Painonapit

Painonapeilla 1 ja 2 (P100 ja P101) avataan ja suljetaan porttia ja autotallin ovea.

4.3 Laitteiden väliset yhteydet

Eri laitteiden väliset kommunikointimenetelmät ovat tärkeässä osassa suuremmassa järjestelmässä. Työssä käytetään paljon eri laitteita ja järjestelmiä, joten kommunikoinnin ja käskyttämisen tulee toimia saumattomasti. Autotallin ja portin järjestelmiin langaton tiedonsiirto toteutetaan kaukosäätimeltä 2,4GHz radiotekniikkaa käyttäen.

Talon järjestelmään liittyy useita eri väyliä ja kommunikointitekniikoita (ks. kuvio 2). Lämpötila-anturit kytketään sarjaan OneWire-väylään ja LCD-näyttö käyttää puolestaan I2C-väylää. Sähköpostipalvelua ohjataan omalla NodeMCU-laitteella. Tähän laitteeseen kommunikointi toteutetaan suoraan johtimella, joka voi olla high- tai low-tilassa. Myös tiedon lähettäminen ulkoverkkoon tapahtuu omalla NodeMCU-laitteella. Yhteys laitteeseen toteutetaan Arduinon serial-kommunikointipinnejä käyttämällä. Tätä tekniikkaa kutsutaan myös nimellä UART.



Kuvio 2. Väylien topologia

I2C-tiedonsiirtoväylä on suosittu väylätekniikka, koska se on yhteensopiva moniin laitteisiin. Väylä on myös helppo toteuttaa, sillä se tarvitsee toimiakseen vain kaksi johdinta viestintään. Laitteita voi olla väylällä 128, kun käytetään 7-bittisiä osoitteita. Niin monen laitteen kanssa kommunikointi kahdella johtimella on mahdollista, koska jokaisella laitteella on yksilöllinen osoite, jonka perusteella löydetään oikea laite. Väylän johtimia kutsutaan serial clock lineksi (SCL) ja serial data lineksi (SDL). SCL-johdin on kellosignaali, joka synkronoi tiedonsiirron eri laitteiden välillä. SDA-johdin kuljettaa tietoa. (Nedelkovski n.d.)

Arduinossa käytettyä serial-väylää kutsutaan sarjaliikenneväyläksi. Sarjaliikenne-väylää käytetään Arduinon ja tietokoneen tai muiden laitteiden väliseen kommunikointiin. Jokaisella Arduinolla on vähintään yksi serial-portti, josta käytetään myös nimityksiä UART tai USART. Väylä viestii digitaalisilla pinneillä TX ja RX. Sarjaliikenteen nopeus voidaan määrittää ohjelmasta. Yleisesti nopeus on 9600 baudia eli bittiä sekunnissa. (Arduino Serial 2017.)

5 Laitteet

Laitteet, jotka toimivat mittauksissa tai toimilaitteina, kytketään kiinni Arduino-kehitysalustaan, joka toimii järjestelmän ytimenä. Kehitysalustat perustuvat mikrokontrollereihin, jonka ympärille ne on rakennettu.

Inputtien ja outputtien, eli sisäänmenojen ja ulostulojen funktio voi olla monimutkainen, joten sitä ei kannata tai ei edes voi toteuttaa erillislogiikalla. Tällöin funktio voidaan toteuttaa prosessorien avulla. Mikrokontrollerit ovat ohjauspiirejä, joissa on sisäänmenojen, ulostulojen ja prosessorin lisäksi laskureita, komparaattoreita, AD-muuntimia ja muistia. (Aaltonen, Kousa, Stor-Pellinen 2000, 280.)

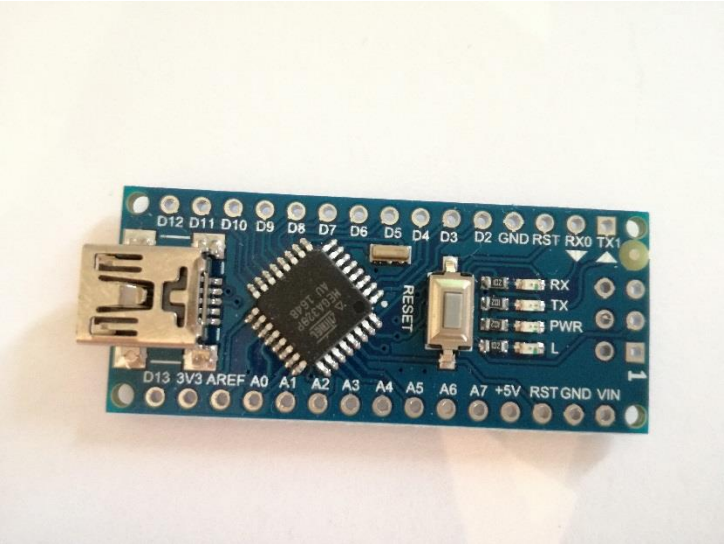
5.1 Arduino-kehitysalusta

Arduino on fyysisen tietojenkäsittelyn avoin alusta. Se koostuu yksinkertaisesta piirilevystä, jolla on ulos- ja sisääntuloliitännät. Arduinon ohjelmointi perustuu Processing-ohjelmointikieleen. Arduino IDE on integroitu kehitysympäristö, joka perustuu avoimeen lähdekoodiin. Tämän avulla laitteen ohjelmointi on aloittelijallekin helpompaa. (Banzi 2011, 1.)

Arduinoa käytetään erilaisten sovellusten ja projektien rakentamiseen ja ohjaamiseen. Ohjelmisto toimii tietokoneella ja sitä käytetään ohjelman kirjoittamiseen ja lähettämiseen Arduino-alustalle.

Arduinon sydämenä toimii 8-bittinen Atmel AVR- mikro-ohjain, jonka ympärille laitteisto rakentuu. Siinä on myös pinnejä, joihin voidaan kytkeä erilaisia ohjattavia ja mittaavia komponentteja kuten esimerkiksi valoja, moottoreita tai tunnistimia.

Arduinoja on erilaisia, kuten esimerkiksi Arduino Nano, -Uno, ja -Mega. Arduino Nano on yleisimmin käytettyjä alustoja. (Ks. kuvio 3.) Eri mallit eroavat toisistaan digitaalisten ja analogisten pinnien määrässä sekä keskus- ja flash-muistissa.

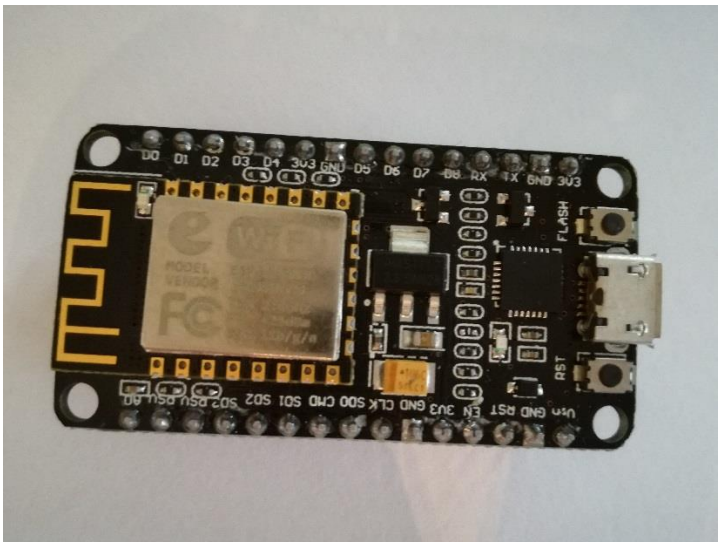


Kuvio 3. Arduino Nano

Laitteisiin on saatavilla myös suoraan kiinnitettäviä laitteita, joita kutsutaan shieldiksi. Se liitetään suoraan laitteiston pinneihin. Työssä käytettävät Arduinot ovat Arduino Nano ja Arduino Mega. Nanossa on ATmega328 mikrokontrolleri, 32KB flash-muistia, 8 analogista ja 22 digitalista porttia. Megassa on ATmega2560 mikrokontrolleri, 256KB flash-muistia, 16 analogista ja 54 digitalista porttia.

5.2 NodeMCU-kehitysalusta

NodeMCU on IoT-kehitysalusta, jonka ytimenä toimii ESP8266 WiFi-moduuli (ks. kuvio 4). Laitetta voidaan ohjelmoida Arduino IDE-ohjelmalla, kunhan NodeMCU-kirjasto on asennettuna. Lisäksi laitteessa on digitaalisia pinnejä, joiden toiminta voidaan ohjelmoida. Laite voidaan kytkeä langattomasti verkkoon ja sitä voidaan käyttää erilaisissa IoT-projekteissa lukemaan antureilta saatua tietoa ja ohjaamaan toimilaitteita.



Kuvio 4. NodeMCU

5.3 Liiketunnistin

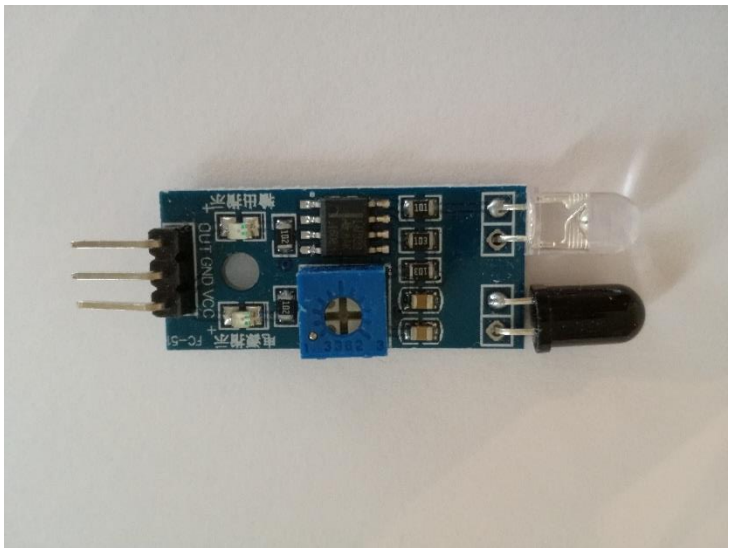
Työssä käytetty liiketunnistin on HC-SR501 passiivinen infrapunatunnistin. PIR-anturia käytetään liikkeen tunnistamiseen. Anturi tunnistaa liikkeen jonkin lämmönlähteen infrapunasäteilyn vaihtelun perusteella. Jotta kohde voidaan tunnistaa, sen täytyy liikkua sekä sen lämpötilan on oltava korkeampi, kuin kohteen taustan lämpötila.

PIR-anturissa on kolme pinniä, jotka ovat GND, VCC ja signaali. Anturi toimii 5V käyttöjännitteellä ja kun anturi tunnistaa liikkuvan kohteen, signaali-pinnin jännite muuttuu nollasta voltista viiteen volttiin. Tämä tarkoittaa sitä, että signaalipinni menee low-tilasta high-tilaan.

5.4 Ovikytkin

Ovien tilatietojen tunnistamiseen käytetään infrapunatekniikalla toimivia antureita (ks. kuvio 5). Anturit toimivat lähetin-vastaanotin -periaatteella, eli anturi lähettää infrapunasäteilyä ja tarkkailee säteilyn kimpoamista takaisin. Tällä periaatteella voidaan havaita esteitä, joista säteily kimpoaa takaisin. Anturissa on GND, VCC ja signaalipinnit ja se toimii 3.3 VDC tai 5 VDC jännitteellä. Anturissa on myös säätöruuvi, josta

voidaan säätää tunnistamisetäisyyttä. Kun laite tunnistaa kohteen, signaalipinni menee high-tilaan.



Kuvio 5. Infrapuna-anturi

5.5 Savunilmaisin

Savun tunnistamiseen käytetään MQ2 savunilmaisinta. Tunnistimessa on tinan ja oksidi-ionien muodostama orgaaninen yhdiste, jonka johtavuutta mitataan. Kun ilmassa on savua tai muita kaasuja, yhdisteen johtavuus muuttuu. Tämän johtavuuden muutoksen vuoksi pystytään havaitsemaan savua ja kaasuja. Tunnistin toimii 5 VDC jännitteellä ja siinä on GND, VCC, DO ja AO pinnit. AO-pinnistä voidaan mitata savun tai kaasun määrää tarkemmin, mutta tässä työssä käytetään vain DO-pinniä, joka muuttuu HIGH-tilaan, kun savua havaitaan. Tunnistamisherkkyyttä voidaan säätää säätöruuvista.

MQ2-savunilmaisin on herkkää materiaalia, jonka johtamiskyky on puhtaassa ilmassa pienempi. Mitä enemmän ilmassa on kaasua, sitä parempi on myös anturin johtumiskyky. Sähkövirran avulla voidaan mitata ilman kaasupitoisuus. Anturi on herkkä propaanille, vedylle, nestekaasuille, metaanille sekä muille höyryille. (MQ-2 Semiconductor Sensor for Combustible Gas n.d.)

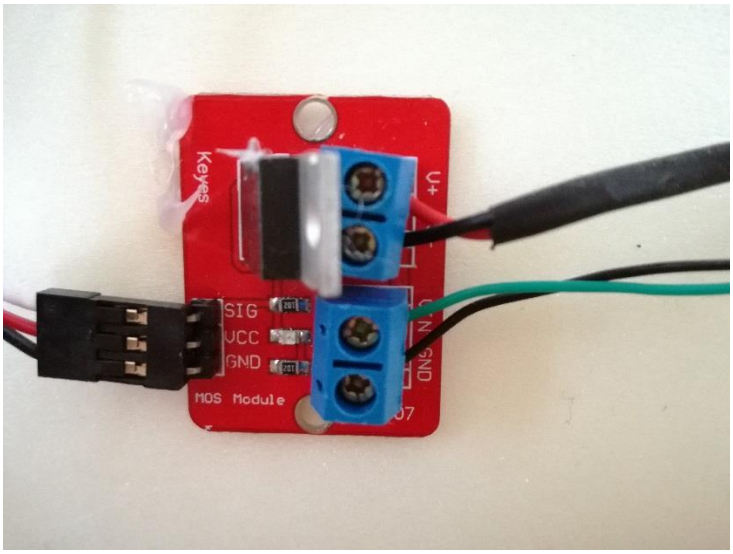
5.6 Lämpötila-anturi

Lämpötilan mittauksessa käytetään Dallas DS18B20 -anturia. Anturi tunnistaa lämpötilan ja ilmoittaa sen suoraan celsiusasteina. Anturi kommunikoi 1-wire-väylän kautta, joka sallii antureiden kytkimisen sarjaan samoilla kaapeleilla. Jokaisella anturilla on oma sarjanumero, minkä vuoksi jokaista sarjaan kytkettyä anturia voidaan lukea erikseen. Anturissa on GND, VDD ja DQ -pinnit. VDD-pinniin kytketään 5 VDC:n jännite ja DQ-pinnistä voidaan lukea lämpötilatieto. DQ-pinni täytyy kytkeä myös viiden voltin jännitteeseen 4,7 kΩ:n vastuksen kautta.

DS18B20 on 1-johtiminen digitaalinen lämpötila-anturi. Anturi mittaa lämpötilan välillä -55 °C ja +125 °C ja sen tarkkuus on ±0,5 °C. Jokaisella anturilla on yksilöllinen 64-bittinen sarjanumero, joka mahdollistaa hyvin suuren määrän antureita samalla väylällä. Antureiden virtalähdealue on 3 VDC- 5,5 VDC. Antureita voidaan käyttää erilaisien säätimien ohjaukseen, teollisuusjärjestelmiin, kulutustuotteisiin, lämpömittareisiin sekä kaikkiin lämpöherkkiin järjestelmiin. (Dimitrov 2016.)

5.7 Tuuletin

Tuulettimena käytetään tavallista tietokoneen tuuletinta, joka toimii 12 VDC-jännitteellä. Moottorin pyörimisnopeutta halutaan vaihdella, minkä vuoksi moottori on kytketty mosfet irf520-moduuliin (ks. kuvio 6). Moduulissa on ruuviterminaalit moottorin V+ ja GND -johtimiin sekä erilliset ruuviterminaalit syöttöjännitteelle johon voi kytkeä 12-24 VDC-jännitteen. Lisäksi moduulissa on pinnit, joihin kytketään ohjaavalta laitteelta VCC, GND sekä signaali. Signaali määrittää mosfet-transistorin johtavuuden ja se on PWM-tyyppinen ohjausjännite.



Kuvio 6. Mosfet IRF520-moduuli

Mosfet-transistorin nimitys tulee sanoista Metal-Oxide-Semiconductor Field-Effect-Transistor. Mosfet-transistoreita on sulku- sekä avaustyyppisiä. Avaustyypissä (enhancement mode) hilan ja sourcen välinen jännite-ero avaa drain-source-kanavan ja virta alkaa kulkea. Kun hilalle syötetään nolla volttia, transistori sulkeutuu eikä virta kulje. (Koskinen 2002, 103-104.)

5.8 Summeri

Työssä käytettävä summeri on piezosummeri. Summerin toiminta perustuu piezosähköiseen ilmiöön. Jännite saa summerissa aikaan mekaanista liikettä, joka synnyttää ääniaaltoja. Jännitteen eri taajuudet saavat aikaan erilaisia ääniä, mutta tässä työssä käytetään vain yhtä taajuutta, joka on 1 kHz. Summerin GND kytketään ohjaavan laitteen groundiin. VCC kytketään 100 Ω:n vastuksen kautta digitaaliporttiin, jonka taajuutta säädetään ohjelmallisesti erillisen kirjaston avulla.

Sana piezosähkö on peräisin Kreikan kielen sanasta piezein, joka tarkoittaa puristamista tai painamista. Piezosähköisen ilmiön yksi ainutlaatuinen ominaisuus on sen kyky toimia käänteisesti. Tämä tarkoittaa sitä, että piezosähköisen ominaisuuden omaava materiaali synnyttää jännitteen sitä puristaessa. Tämän lisäksi se toimii päinvastoin, eli kun siihen johdetaan jännite, se saa aikaan materiaalin venymisen tai pak-

kaantumisen. Ilmiö on hyödyllinen monissa sovelluksissa, joihin liittyy äänen tuottaminen ja havaitseminen, suurjännitteiden synnyttäminen, elektroninen taajuudenmuodostaminen, mikrobalanssit sekä tarkkojen kokoonpanojen hienosäätö. (The Piezoelectric Effective n.d.)

5.9 Hämärätunnistin

Hämärätunnistimena käytetty anturi on photoresistor eli valovastus. Valovastus on valmistettu puolijohdemateriaalista, minkä vastusarvo muuttuu valoisuuden muuttuessa. Valovastuksen käyttöjännite on 3.3-5 VDC ja siinä on sekä DO- että AO-lähdöt. DO-lähtö menee HIGH-tilaan, kun säätöruuvilla asetettu kynnysarvo ylittyy. AO-lähdöstä voidaan mitata sen hetkistä valoisuuden määrää. Työssä anturia käytetään kytkemään valaistusta päälle ja pois päältä, joten siinä on käytetty DO-lähtöä.

Valoa johtavia kennoja kutsutaan joskus myös valoresistoreiksi. Ne toimivat valoherkkyyden mukaisesti. Tiettyjen puolijohteiden vastusarvo vähenee, kun niihin osuu valoa. Esimerkkejä tällaisista puolijohteista on kadmiumsulfidi (CdS) ja kadmium selenidi (CdSe). (Cirovic 1979.)

5.10 LCD-näyttö

Tietojen näyttämiseen käyttäjälle on käytetty nestekidenäyttöä. Käytettävässä näytössä on sisäänrakennettu I2C-väylämahdollisuus ja sen väylä-osoitetta voidaan vaihtaa välillä 0X20-0X27. Näyttö on kooltaan 2 riviä ja 16 saraketta ja siinä on säätöruuvi, jolla voidaan säätää taustavalon voimakkuutta. I2C-väylä-adapteriin kytketään GND, VCC sekä ohjaavan laitteen SDA (data line) ja SCL (clock line).

5.11 Askelmoottori

Portin ja autotallin oven ohjaamiseen on valittu 5 VDC-jännitteellä toimiva 28BYJ-48 askelmoottori. Askelmoottoria ohjaa ULN2003 ohjain. Moottori kytketään ohjaimeen valmiiksi asennetulla liittimellä. Ohjaimeen tuodaan 5 VDC-käyttöjännite, sekä ohjaavan laitteen askel-signaalit, jotka kytketään pinneihin IN1-IN4. Askelmoottoria ohjataan pienillä askelilla, joten askelmoottorin pyöriessä yhden kierroksen, se on liikkunut useita askelia. Moottorissa on magneetteja, jotka reagoivat käämiin johdettuun

jännitteeseen ja saavat aikaan yhden askeleen. Kun käämeihin johdetaan jännitteitä oikeassa järjestyksessä, se saa aikaan moottorin pyörimisen. Moottorin pyörimissuuntaa voidaan myös vaihtaa syöttämällä käämeihin jännitettä päinvastaisessa järjestyksessä. Askelmoottorit ovat hyvin tarkkoja ja työssä käytetyllä askelmoottorilla yksi kierros tarvitsee 64 askelta mikä tarkoittaa sitä, että yksi askel on noin 5,7 astetta.

Askelmoottori on harjaton ja synkroninen sähkömoottori, joka pyörittää akselia askeleen kerralla saadessaan digitaalisen sähköpulssin. Akselin yksi pyörähdyskierros on jaettu useisiin askeliin, joita voi olla jopa 200. Moottori saa yhden sähköpulssin askeletta kohti, mikä aiheuttaa akselin pyörähtämisen tavallisesti noin 1,8°. Kun ohjain syöttää digitaalisia pulsseja tietyllä taajuudella, se saa akselin pyörimään tasaisesti. Pyörimisnopeus on suoraan verrannollinen taajuuteen, jolla ohjauspulsseja syötetään. Askelmoottoreita käytetään paljon niiden edullisuuden, luotettavuuden ja korkean vääntömomentin ansiosta. (Stepper Motor Basics n.d.)

5.12 Näppäinpaneeli

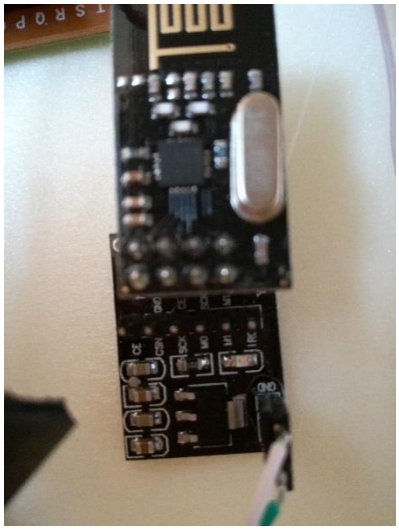
Käytetty näppäinpaneeli on kooltaan 4x3 merkkiä, eli siinä on yhteensä 12 merkkiä (ks. kuvio 7). Numeroiden 0-9 lisäksi siinä on *-ja #-merkit. Paneelissa on 7-johdinta, jotka on kytketty siten, että jokaiselle riville ja sarakkeelle on kytketty oma johdin. Kun yhtä näppäintä painetaan, saadaan painalluksesta tieto kahteen eri johtimeen. Johtimet kytketään ohjaavan laitteen digitaalisiin portteihin. Ohjaavaan laitteeseen on saatavilla erillinen kirjasto, jonka avulla saadaan tieto painalluksista.

Kuvio 7. Näppäinpaneeli

5.13 NRF24L01-lähetin/vastaanotin moduuli

Langaton yhteys on toteutettu käyttämällä NRF24L01-lähetin-vastaanotin paria. Yhteys laitteiden välillä on toteutettu 2,4Ghz taajuuskaistalla. Laite toimii 3,3 VDC jännitteellä, joten työssä on käytetty laitteille tarkoitettuja adaptereita mitkä mahdollistavat 5 VDC-jännitteen käytön (ks. kuvio 8). Adapterissa on 8-nastaa jotka ovat VCC, GND, MISO, MOSI, SCK, CS, CE ja IRQ. MISO-, MOSI-, SCK- ja CS-nastoja käytetään datan liikuttamiseen SPI-väylän avulla. CE-nasta mahdollistaa TX-ja RX-tilan käytön ja IRQ-nasta on keskeytysnasta. IRQ-nastaa ei työssä ole kytketty ollenkaan. Lähetin-vastaanotinparin kantama on ilman lisäantenneja muutamia satoja metrejä, jos välissä ei ole esteitä. Lisäantenneilla varustetut laitteet yltävät useiden satojen metrien päähän.

Lähetin- ja vastaanotinmoduuli käyttää 2,4Ghz taajuuskaistaa ja se toimii 250kbps-2Mbps baudinopeuksilla (bittiä sekunissa). Matalalla nopeudella ja avoimessa tilassa sen kantama voi olla 100 metriä. Moduulit voivat käyttää 125 eri kanavaa, jotka mahdollistavat verkon missä on 125 itsenäistä modeemia samassa paikassa. Jokaisella kanavalla voi olla kuusi eri osoitetta tai jokainen laite voi kommunikoida kuuden eri laitteen kanssa samanaikaisesti. (Nedelkovski n.d.)



Kuvio 8. NRF24L01 ja adapteri

5.14 Jännitemuunnin

Projektissa tarvitaan 12 VDC jännitettä, sekä 5 VDC jännitettä. Jännitesyöttö on 12 VDC, joten 5 VDC saadaan tästä muuntimen avulla. Työssä on käytetty XL4015 jännitteenalentajaa. Sisääntulojännite voi olla 5-32 VDC välillä ja lähtöjännite 0,8-24 VDC. Lähtöjännite ei voi olla kuitenkaan suurempi, kuin tulojännite. Lähtöjännite on säädettävissä säätöruuvilla.

6 Opinnäytetyön toteutus

6.1 Pienoismallin rakentaminen

Pienoismallin rakentaminen alkoi materiaalihankinnoilla. Pohjaksi valikoitui finnfoam-eristelevy sen kestävyyden ja keveyden vuoksi. Rakennelmien seinämateriaalit on tehty vanerista, jotka leikattiin sopivan kokoisiksi pohjapiirustuksen mukaisesti, tapetoitiin valmiiksi sekä asennettiin pohjalevyyn liiman ja puutikkujen avulla (ks. kuvio 9). Työstä haluttiin saada siistin näköinen, joten kaikki tapetoimattomat osat maalattiin valkoisella akryylimaalilla.

Kuvio 9. Pienoismallin rakentamista

Portti, autotallin ovi, talon ovet, aidat, terassit sekä puhaltimen suoja on rakennettu puutikuista, narusta sekä depron-levystä. Osat on viimeistelty akryylimaalilla ennen asennusta. Portti on asennettu pyörivien kiekkojen väliin (ks. kuvio 10).

Kuvio 10. Portin asennus

Käyttöpaneeli on leikattu vanerilevystä ja ennen asennusta siihen on porattu sopivat reiät kytkimille, potentiometrille ja näytölle. Paneeli on viimeistelty akryylimaalilla ja se on upotettu ja kiinnitetty pohjalevyyn. (Ks. kuvio 11.)

Kuvio 11. Käyttöpaneelin asennus

Pienoismallissa oleva tie on karkeaa hiekkaa, joka on kiinnitetty paksulla liimakerroksella. Muualle piha-alueelle on levitetty askartelussa käytettävää valkoista struktuuripastaa, joka tekee pohjan pinnasta hieman elävämmän näköisen. Pohja on viimeistelty vihreällä akryylimaalilla. Talon sekä autotallin lattiat on tehty dc-fix-muovilla, jota on helppo muotoilla ja asentaa. Valaistuksien, liiketunnistimen ja savunilmaisimen suojakotelot on tehty finnfoam-eristelevystä ja depron-levystä ja ne on viimeistelty akryylimaalilla.

6.2 Laitteiden asennus

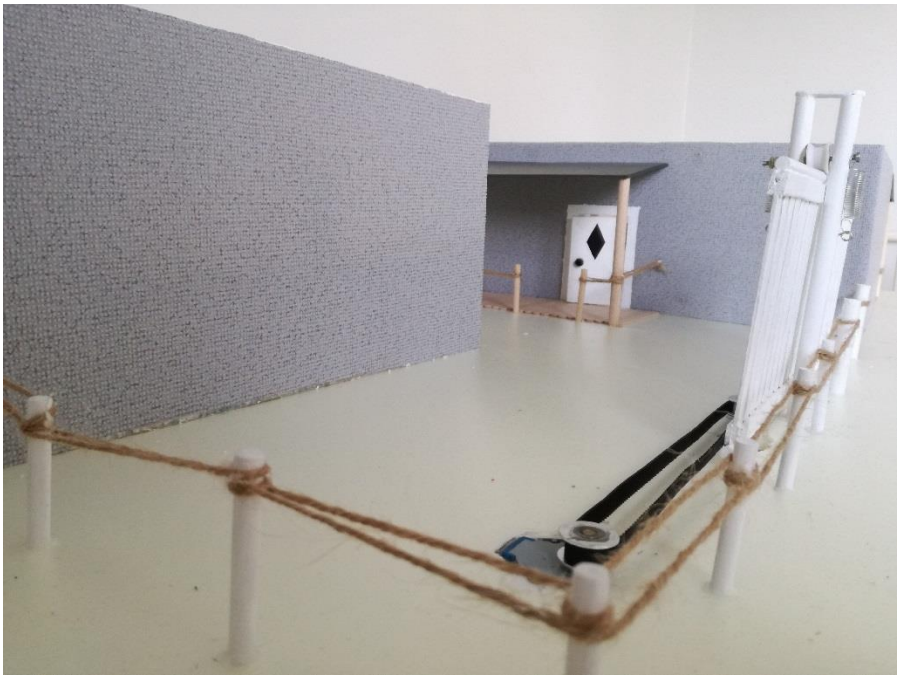
Kaikki laitteet ovat asennettu laitteiden toimintakuvauksen mukaisesti. Käyttöpaneeliin on asennettu yhdeksän katkaisijaa, joilla voidaan ohjata talon sekä autotallin valaistuksia. Lisäksi valikon painonappi, potentiometri sekä LCD-näyttä on upotettu

käyttöpaneeliin. Näppäinpaneeli on asennettu käyttöpaneeliin pinta-asennuksena tarrakiinnityksellä.

Valaistukset, lämpötila-anturit, liiketunnistin, summeri sekä savunilmaisin on asennettu talon sisäpuolelle seiniin kiinni ja johdotukset on viimeistelty käyttämällä pieniä johtokouruja. Valoisuusanturi sekä tuuletin on asennettu talon ulkopuolelle, jotta ne kuvaisivat parhaiten oikeaa ympäristöä (ks. kuvio 12).

Kuvio 12. Valoisuusanturi ja tuuletin

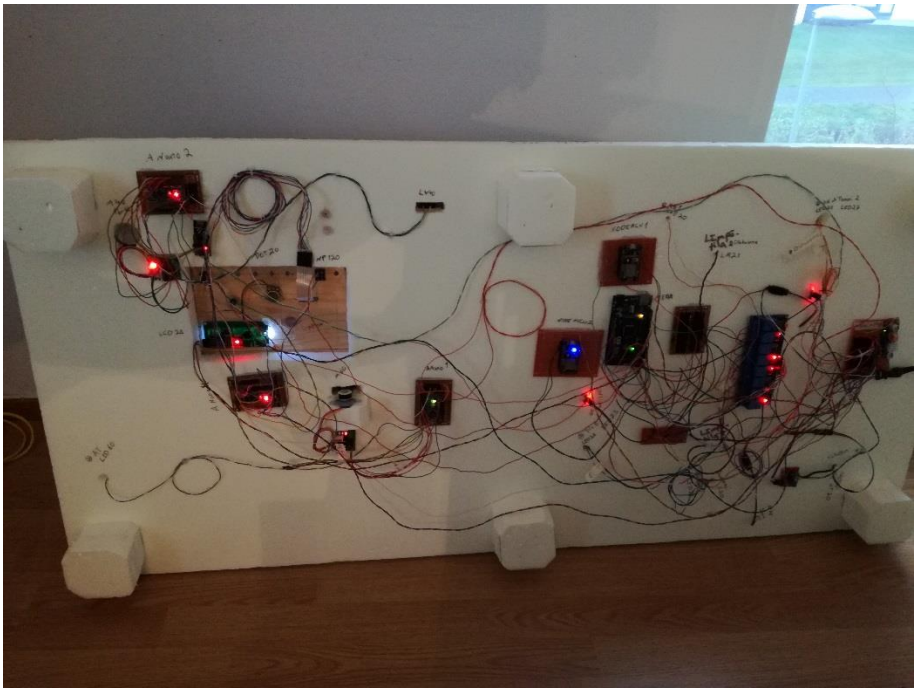
Autotallin ovea sekä porttia liikuttavat askelmoottorit on upotettu pienoismallin pohjaan ja liikkeet on johdettu moottoreilta portille ja ovelle käyttämällä hihnapyöriä ja hammashihnaa (ks. kuvio 13). Lisäksi portin eteen tielle on upotettu painelaatta, joka tunnistaa ajoneuvon liikkeen. Painelaatta on rakennettu painonapista sekä puutikusta.



Kuvio 13. Portin ohjaus

6.3 Laitteiden kytkentä

Kaikkien laitteiden kytkennät näkyvät piirikaavioista (ks. liite 2). Kytkennät sekä asennukset tehtiin pienoismallin alle, jotta työstä tulisi siisti (ks. kuvio 14). Kappaleessa esitellään kuitenkin tavallisesta poikkeavien laitteiden kytkennät erikseen ja tarkemmin.



Kuvio 14. Laitteiden kytkennät

Valaistus on toteutettu valkoisilla led-valoilla. Jokaisella eri värisellä led-valolla on oma kynnysjännite, eli jännite, joka voidaan led-valon palaessa mitata valon navoista. Led-valot eivät kestä paljoa virtaa, joten ne tarvitsevat vastuksen rajoittamaan virtaa. Tätä vastusta kutsutaan etuvastukseksi. Etuvastus voidaan laskea, kun tiedetään led-valon kynnysjännite Ud, syöttöjännite U sekä haluttu virta I. Vastuksen navoilla vaikuttava jännite saadaan vähentämällä syöttöjännitteestä led-valon navoilla vaikuttava jännite, eli $Ur = U - Ud$. Vastuksen resistanssi saadaan laskettua ohmin lain avulla kaavalla R= $\frac{Ur}{I}$. Työssä käytetään valkoisia led-valoja, joiden kynnysjännite on 3,6 V ja valon syöttöjännite on 5 V. Arduinon portit kestävät 20 mA virran, joten sitä käytetään laskukaavassa. Tästä saadaan vastuksen yli jääväksi jännitteeksi 1,4 V ja etuvastuksen suuruudeksi 70 Ω. Työssä on kuitenkin käytetty laskennalliseen arvoon verrattuna hyvinkin suuria etuvastuksia, joiden vastusarvo on 220 Ω. Ylimitoitetut vastukset saavat valon palamaan hieman himmeämmin. Led-valoja ohjataan releiden avulla, joten led-valot voitaisiin korvata tavallisilla 230 VAC lampuilla.

Lämpötila-anturit voidaan kytkeä sarjaan One wire -väylän avulla. Jokaisella anturilla on oma yksilöllinen osoite minkä vuoksi niiden arvoja voidaan lukea erikseen. Anturit tarvitsevat ylösvetovastuksen, joka kytketään väylän ja 5V väliin. Vastus on arvoltaan

4,7kΩ. Ylösvetovastus varmistaa, että silloin kun IO-porttiin ei tule muuta signaalia, se on high-tilassa.

Painonapin sekä kytkimen toinen napa kytketään 5 VDC-jännitteeseen ja toinen napa IO-porttiin. Nämä tarvitsevat kuitenkin toimiakseen alasvetovastuksen, joka on yleensä suuruudeltaan 10kΩ. Alasvetovastus kytketään IO-portin ja groundin välille. Alasvetovastus varmistaa, että silloin kun kytkimestä tai napista ei tule signaalia IO-portti on low-tilassa.

6.4 Ohjelmakoodin kirjoittaminen

Opinnäytetyön suurimpia kokonaisuuksia oli koodin kirjoittaminen. Koodin pitäisi olla sellainen, että se olisi helposti luettavaa eikä ristiriitaisuuksia syntyisi missään tilanteessa. Ohjelmakierron pitäisi olla lisäksi sulavaa, eikä se saisi jäädä yhteen kohtaan liian pitkäksi aikaa. Ohjelmakierto tarkoittaa laitteen koodin suorittamista alusta loppuun. Jos ohjelma pysähtyy esimerkiksi odottamaan tietoa joltakin anturilta, voi jäädä esimerkiksi tulipalo tai jokin ohjauskäsky huomaamatta. Ohjelmakierron syklin tulisi olla mahdollisimman nopea, jotta kaikki toiminnot ja tapahtumat huomioitaisiin, mutta ei kuitenkaan niin nopea, että laite kuormittuu liikaa. Sykli tarkoittaa aikaa joka kuluu yhden ohjelmakierron aikana.

Koodit on tehty Arduino IDE-ohjelmalla jokaiselle älylaitteelle erikseen. Näitä älylaitteita on yhteensä seitsemän joita ovat talossa oleva Arduino Mega, porttia ohjaava Arduino Nano, autotallissa oleva Arduino Nano, näppäinpaneelia tarkkaileva Arduino Nano, kaukosäätimessä oleva Arduino Nano sekä sähköpostipalvelua pyörittävä NodeMCU ja thinger-palveluun liitetty NodeMCU. Autotallin, portin ja kaukosäätimen laitteet kommunikoivat keskenään käyttämällä NRF24L01-laitteita, eikä ne ole yhteydessä muihin laitteisiin. Yhteyttä ei tehty, koska automaatiojärjestelmä jälkiasennuksena voisi tuottaa kaapelointiongelmia talon sekä autotallin ja portin välillä.

6.4.1 Talon ohjelmakoodi

Talossa sijaitseva Arduino Mega toimii masterina, joka antaa käskyjä sekä lukee tietoja muilta laitteilta. Masterin koodi on rakennettu tekemällä ohjelmallisesti eri mit-

taisia syklejä, joita suoritetaan (ks. liite 3). Eri mittaisissa sykleissä kutsutaan aliohjelmia jotka sisältävät varsinaiset toiminnot. Syklit ovat 20ms, 100ms, 500ms ja 1000ms mittaisia. Kaikki toiminnot suoritetaan kuitenkin 1000ms syklissä mikä on todettu riittäväksi. Kaikki toiminnot kuten esimerkiksi kytkimien asentotiedon tarkastaminen ja valojen sytyttäminen tapahtuvat siis yhden sekunnin välein. Tällä vähennetään tarpeettoman usein tapahtuvat toiminnot eikä laite kuormitu liikaa.

Masterin aliohjelmat, jotka suoritetaan 1000ms välein, on jaoteltu toiminnon mukaisesti. Aliohjelmia ovat lämpötilan mittaaminen, tuulettimen ohjaus, valojen ohjaus, summerin ohjaus, näytön ohjaus sekä monien eri antureiden, kytkimien ja säätimien tilatietojen lukeminen. Lisäksi koodissa on useita apualiohjelmia, jotka eivät suoraan ohjaa laitteita. Esimerkkinä apualiohjelmasta mainittakoon laskurit, jotka viivyttävät jonkin tiedon näyttämistä näytöllä.

6.4.2 Näppäinpaneelin ohjelmakoodi

Koska masterin ohjelmasykli on tietyissä tilanteissa yli sekunnin mittainen, ohjelma ei kerkeä rekisteröidä kaikkia näppäinpaneelin painikkeiden nopeita painalluksia. Tästä syystä näppäinpaneelille on asennettu oma Arduino Nano, joka seuraa painalluksia. Tässä laitteessa ohjelmasykliä ei ole rajoitettu, vaan se suorittaa koodia niin nopeasti kuin pystyy. Kun käyttäjä painaa näppäinpaneelissa olevia painikkeita nopeasti, laite kerkeää huomaamaan nämä painallukset nopean ohjelmakierron ansiosta. Näppäinpaneelia seuraavan Arduino Nanon koodiin on tehty muistipaikat mihin näppäimien painallukset tallennetaan (ks. liite 4). Jos pin-koodi on oikea, muuttuu laitteen digitaalinen portti numero 13 high- tai low-tilaan, mitä talossa sijaitseva Arduino Mega seuraa. Tästä saadaan kotona- ja poissa-tilatiedot.

6.4.3 Kaukosäätimen ohjelmakoodi

Kaukosäätimellä ohjataan porttia sekä autotallin ovea auki ja kiinni. Tämän kehitysalustana on käytetty Arduino Nanoa. Kaukosäätimessä olevalla 3-asentoisella kytkimellä valitaan NRF24L01-laitteiden välinen yhteys joko porttiin tai autotalliin ja painonapeilla tehdään itse ohjaus. Kytkimen asennosta riippuen aktivoidaan keskusteluputki kaukosäätimen ja portin välille, tai kaukosäätimen ja autotallin välille. Tällöin

ohjaukset eivät voi mennä ristiin ja ohjattava kohde tottelee ohjausta. Keskusteluputket ovat NRF24L01-laitteiden käyttämiä osoitteita, joilla erotellaan kanavat toisistaan. Esimerkkinä ihmisten välisestä kommunikoinnista tämä keskusteluputki on katsekontakti. Katsekontakti kertoo sekä puhujalle, että kuuntelijalle kenelle sanoma on tarkoitettu. Keskusteluputken muodostamisen jälkeen lähetetään vastaanottajalle yksinkertainen auki- tai kiinni-käsky. Nämä käskyt ovat koodissa toteutettu kokonaisluvuilla 1 ja 2. Kaukosäätimen ohjelmakoodi kommentoituna liitteenä (ks. liite 5).

6.4.4 Portin ohjelmakoodi

Portin ohjaukseen sekä esteen havaitsemiseen on käytetty kehitysalustana Arduino Nanoa. Ohjaus tapahtuu kaukosäätimellä, joten ohjausviestin vastaanottamiseen on käytetty NRF24L01-vastaanotinta. Vastaanotin tarkkailee omaa keskusteluputkea, jonka osoite on 0xF0F0F0F0E1LL. Kun vastaanotin havaitsee liikennettä putkessa, se lukee viestiä niin kauan, kunnes liikenne putkessa lakkaa. Viesti on joko auki- tai kiinni-käsky. Viestin saatuaan ohjataan porttia auki tai kiinni ja jatketaan putken kuuntelua. Koodissa tarkkaillaan myös painelaatan asentoa, mikä on sijoitettu portin eteen. Jos ajoneuvo on ajanut portin väliin, portti ei voi mennä kiinni. Portin ohjelmakoodi kommentoituna liitteenä (ks. liite 6).

6.4.5 Autotallin ohjelmakoodi

Autotallin ohjelmakoodi toimii samalla tavalla kuin portin ohjelmakoodi ilman painelaatan tarkkailua (ks. liite 7). Autotallin ohjaukseen on käytetty kehitysalustana Arduino Nanoa sekä viestin vastaanottamiseen NRF24L01-vastaanotinta. Kaukosäätimen ja autotallin välisen keskusteluputken osoite on 0xF0F0F0F0E2LL. Kehitysalustan avulla ohjataan lisäksi autotallin valaistusta.

6.4.6 Sähköpostin lähettämisen ohjelmakoodi

Sähköpostin lähettämiseen on käytetty erillistä NodeMCU-kehitysalustaa. Sähköpostin lähettämiseen on käytetty erillistä IFTTT-palvelua. Tämän seurauksena koodin rungon muodostaa poikkeuksellisesti Easycoding-palvelulla tehty ohjelmakoodi (ks. liite 8). Koodin tekemisestä Easycoding-palvelulla on kerrottu tarkemmin luvussa 7.5.1. Pieniä muutoksia koodiin on kuitenkin tehty, kuten sähköpostiviestin sisällön

muuttaminen tilanteesta riippuen. Lisäksi kehitysalusta tarkkailee digitaalisia portteja 5 ja 14, joihin se saa sähköpostiviestin lähetyskäskyn talon Arduino Megalta.

6.4.7 IoT-palvelun ohjelmakoodi

Erilaisten tietojen lähettäminen Thinger-palveluun on toteutettu erillisellä NodeMCU-kehitysalustalla. Kehitysalusta yhdistää koodissa määriteltyyn modeemiin josta on yhteys ulkoverkkoon. Lisäksi määritellään Thinger-palvelussa asetettu käyttäjänimi, laitteen tunnus ja salasana. Kehitysalusta tarkkailee itsensä ja talossa sijaitsevan Arduino Megan välistä serial-liikennettä. Serial-väylältä luetaan lämpötila-arvoja, moottorin pyörimisnopeutta sekä talon lämpötilan asetusarvoa. Nämä tiedot tallennetaan omiin muuttujiin ja lähetetään Thinger-palveluun liitteenä olevan koodin mukaisesti (ks. liite 9).

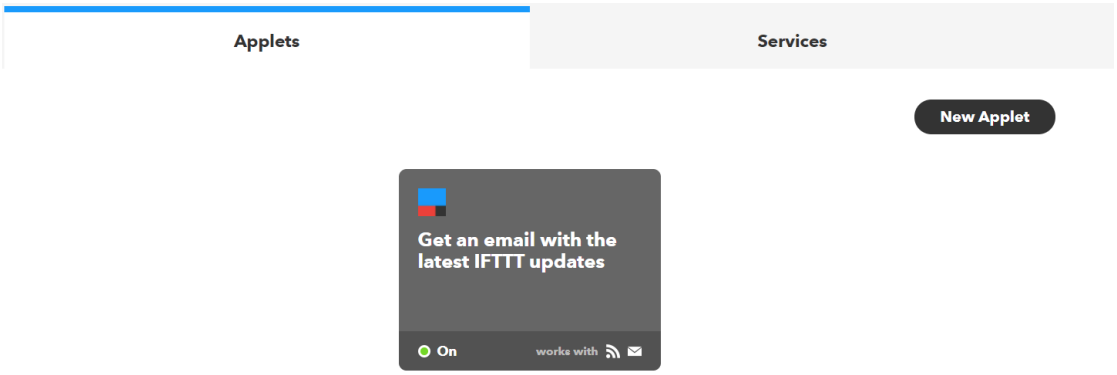
6.5 IoT-ominaisuudet

Kotiautomaatiojärjestelmästä halutaan lukea tietoja etänä ja lisäksi halutaan lähettää tulipalon sattuessa tai varkaita havaitessa sähköpostiviesti käyttäjälle. Näiden ominaisuuksien vuoksi järjestelmä pitää yhdistää internet-verkkoon. Verkkoon liittäminen on toteutettu käyttämällä NodeMCU-laitteita, joissa on sisäänrakennettu ESP8266 WiFi-moduuli. Kun NodeMCU yhdistetään paikalliseen WLAN-tukiasemaan, se toimii järjestelmän ja internet-verkon rajapintana. Sähköpostin lähettäminen ja kotiautomaatiojärjestelmän tietojen lähettäminen pilvipalveluun on toteutettu kahdella eri NodeMCU -laitteella ja kahdella eri palvelulla. Nämä voisi toteuttaa myös yhdellä laitteella ja yhdellä palvelulla, mutta opinnäytetyössä on haluttu käyttää eri palveluita vertailun vuoksi.

6.5.1 Sähköpostipalvelu

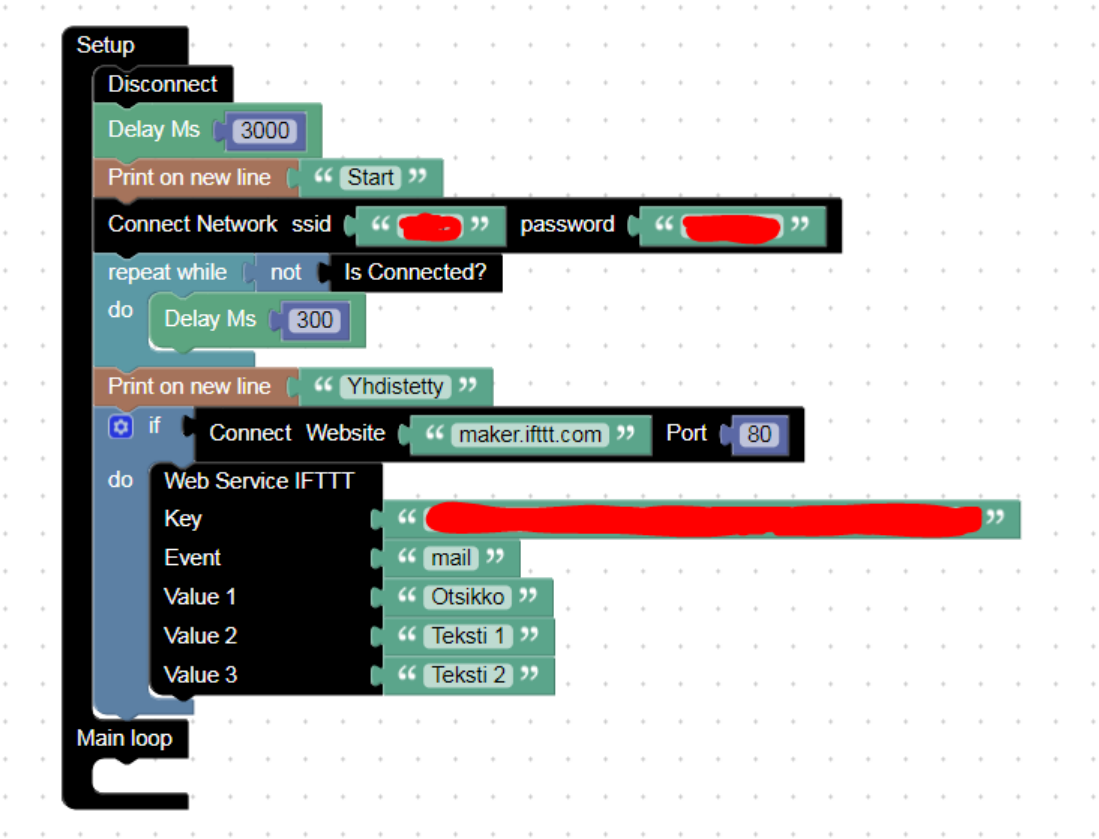
IFTTT (if this then that) -palvelu on keskittynyt pääasiassa sosiaalisen median eri tilien linkittämisen toisiinsa, mutta palvelulla on mahdollista tehdä paljon muutakin. Palvelussa tehdään jokin resepti (recipe), joka linkittää kaksi palvelua toisiinsa. Reseptissä olevista palveluista tarkastetaan yksinkertainen ehtolauseke (jos tämä sitten tuo). Ehtolauseke tarkastaa onko reseptissä oleva asia tapahtunut ja jos on, tehdään jokin ennalta määritelty toiminto. Opinnäytetyötä varten tehdyssä reseptissä tarkastetaan,

onko talossa varkaita tai onko talossa tulipaloa ja jos on, niin lähetetään käyttäjälle sähköposti. IFTTT-palveluun tehdään käyttäjätunnus minkä jälkeen voidaan tehdä ja tallentaa omia reseptejä (ks. kuvio 15).



Kuvio 15. IFTTT-resepti

Sähköpostipalvelua pyörittävän NodeMCU-laitteen koodi tehtiin Easycoding-palvelulla. Palvelussa voidaan rakentaa koodi käyttämällä yksinkertaisia blokkeja mitkä yhdistyvät toisiinsa (ks. kuvio 16).



Kuvio 16. Sähköpostipalvelun blokkirakenne

Blokkirakenteesta saatiin suoraan NodeMCU-yhteensopiva koodi. Koodiin piti tehdä pieniä muutoksia, koska sähköpostien lähetyskäskyt saadaan NodeMCU-laitteen digitaalisiin portteihin. Varkaita havaitessa NodeMCU-laitteen digitaaliseen porttiin D5 saadaan signaali. Palohälytyksestä saadaan signaali porttiin D1. Ohjelmakoodi on kommentoituna liitteenä (ks. liite 8).

IFTTT-palvelu on selkeä ja helppokäyttöinen minkä vuoksi se valittiin opinnäytetyöhön. Lisäksi palvelu on ilmainen. Palvelu on kuitenkin tarkoitettu pääasiassa sosiaalisen median eri palveluiden linkittämisen toisiinsa, joten se on hyvin rajoitteellinen.

6.5.2 Thinger-pilvipalvelu

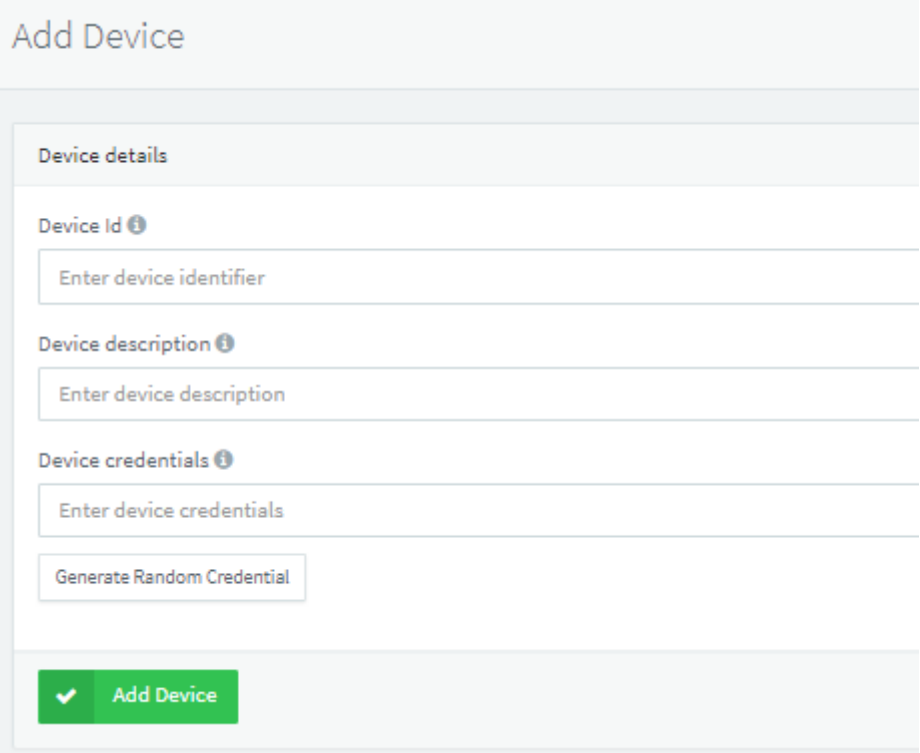
Thinger.io on verkossa oleva palvelu, joka tarjoaa laitteiden ja antureiden tietojen lukemisen ja ohjaamisen etänä. Palveluun on mahdollista liittää laitteita, jotka ovat

verkossa, kuten esimerkiksi Arduino, ESP8266, Raspberry Pi ja Intel Edison. Tämä palvelu valikoitui opinnäytetyöhön helppokäyttöisyytensä ja edullisuutensa vuoksi. Alkuun pääseminen on helppoa eikä se maksa käyttäjälle mitään.

Pienoismallista haluttiin lukea etänä lämpötilatietoja, lämpötilan asetusarvo sekä moottorin pyörimisnopeus. Nämä tiedot lähetetään palveluun esp8266-pohjaisella NodeMCU-laitteella. Laite tarvitsee yksinkertaisen koodin jonka avulla se yhdistää itsensä wlanin avulla sisäverkkoon ja Thinger.io-palvelun oikeaan laitteeseen.

Thinger.io-palveluun tehdään käyttäjätunnus, jolla voidaan kirjautua palveluun miltä tahansa päätelaitteelta, joka on kytkettynä internet-verkkoon. Ilman lisämaksuja käyttäjällä on mahdollista kytkeä palveluun kaksi eri laitetta, tehdä neljä käyttöliittymää, kerätä tietokantaan neljää erilaista tietoa, jotka säilyvät vuoden sekä tehdä halutessaan neljä eri toimintoa. Yksi niistä voi olla esimerkiksi sähköpostin lähettäminen. Lisämaksulla liitettävien laitteiden ja muiden mahdollisuuksien määrä kasvaa huomattavasti.

Laitteet, jotka halutaan liittää, pitää lisätä palveluun asettamalla laitteiden tunnisteet ja salasanat (ks. kuvio 17).



Kuvio 17. Thingerin laitemääritykset

Tunniste ja salasana lisätään myös laitteen koodiin, jotta se osaa linkittää itsensä palveluun. Lisäksi koodiin lisätään rivit niistä lähetettävistä tiedoista, jotka halutaan käyttöliittymässä näyttää (ks. kuvio 18).

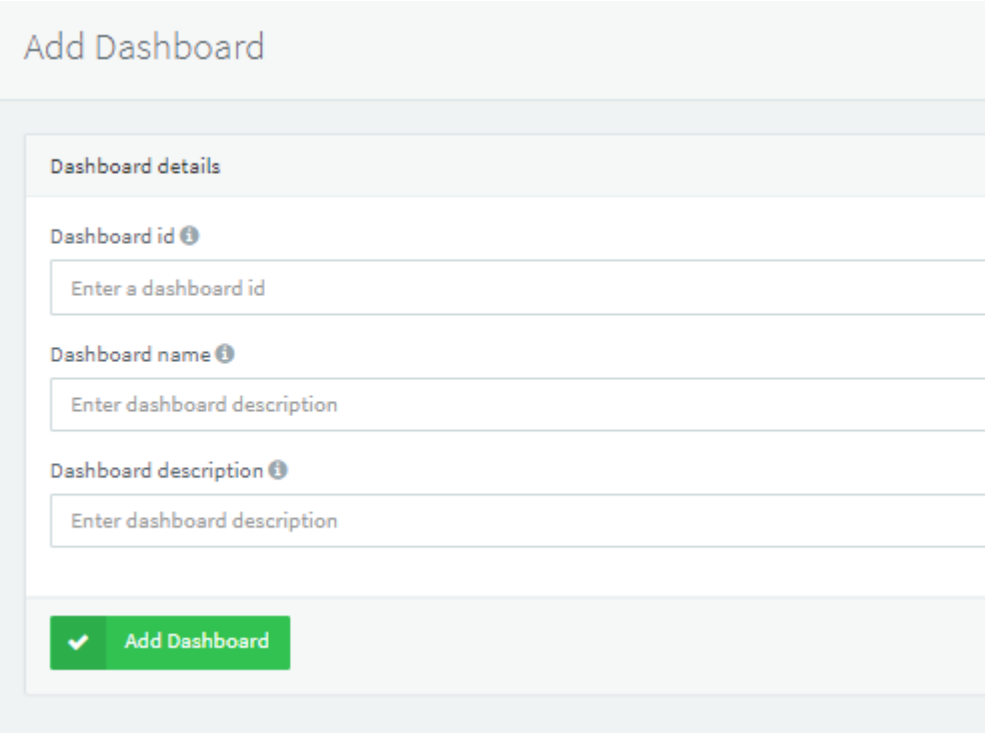
```
#define USERNAME "Käyttäjänimi"                    //Thinger.io palvelussa oleva käyttäjänimi.
#define DEVICE_ID "Tunniste"                       //Device ID, eli laitteen tunniste.
#define DEVICE_CREDENTIAL "salasana"               //Device Credential, eli laitteen salasana.

void setup()
{
  thing["lampotila1"] >> outputValue(arvo[0]);     //Eteisen lämpötilatiedon lähetys palveluun.
  thing["lampotila2"] >> outputValue(arvo[1]);     //Olohuoneen lämpötilatiedon lähetys palveluun.
  thing["prosentti"] >> outputValue(arvo[2]);      //Moottorin pyörimisnopeuden lähetys palveluun.
  thing["M20Setpoint"] >> outputValue(arvo[3]);    //Lämpötilan asetusarvon lähetys palveluun.
}
```

Kuvio 18. Thinger-laitteen tiedot ja näytettävät arvot koodissa

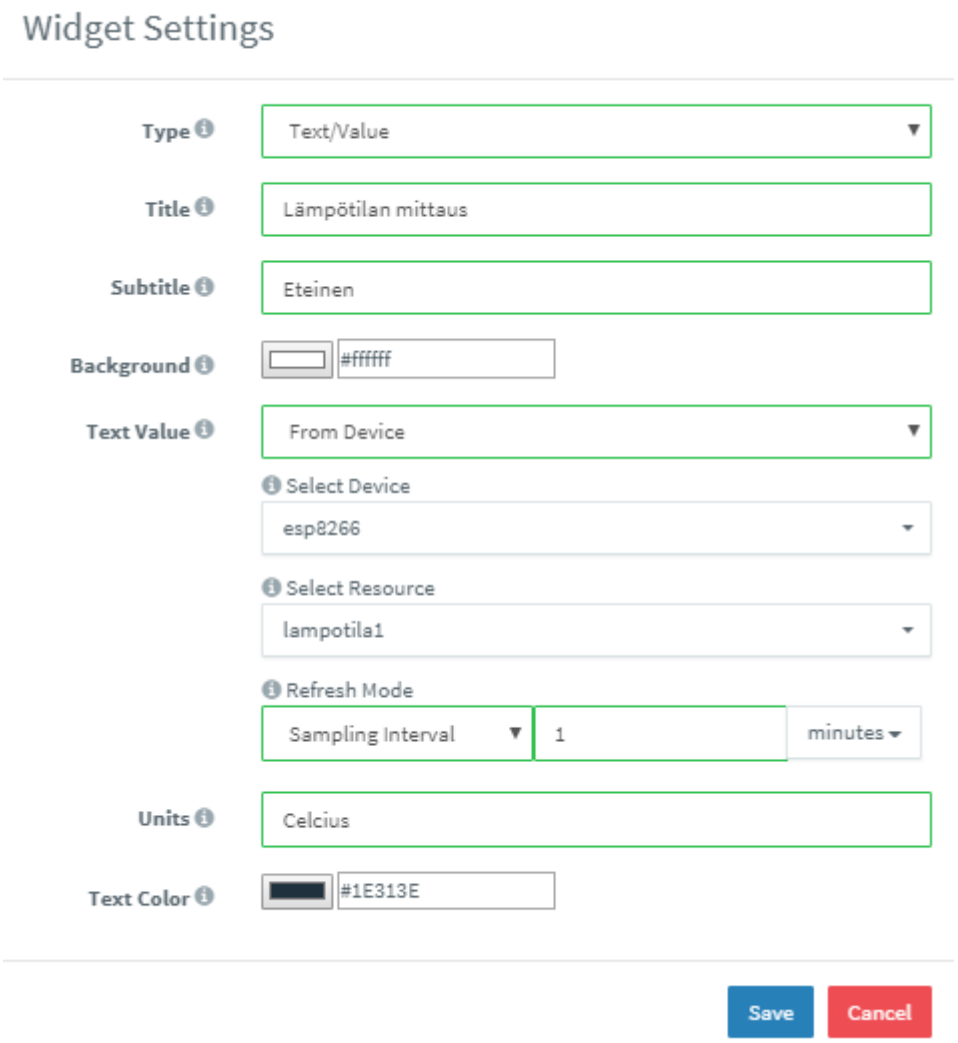
Jotta palvelusta olisi hyötyä, laitteiden lisäksi tehdään myös käyttöliittymä. Määritysvaiheessa käyttöliittymälle annetaan yksilöllinen tunniste, nimi sekä kuvaus (ks. kuvio

19). Tämän jälkeen käyttöliittymä on tehty, ja sinne voidaan liittää näytettäviä tietoja.



Kuvio 19. Thingerin-käyttöliittymän määritykset

Käyttöliittymään lisätään objekteja, joissa näkyvät halutut tiedot. Asetuksista voidaan valita näytettävän tiedon tyyppi, joka voi olla esimerkiksi pistekaavio, rengaskaavio, edistymispalkki tai pelkästään teksti/arvo-ikkuna. Tiedolle annetaan otsikot sekä paikka, mistä arvo luetaan. Esimerkkikuviossa 20 laite on nimetty nimellä esp8266 ja arvo on lampotila1. Päivitystyypiksi on valittu sampling interval ja valitaan haluttu näytteenottoväli. Tiedolle kirjoitetaan oikea yksikkö, joka esimerkkikuviossa on celcius.

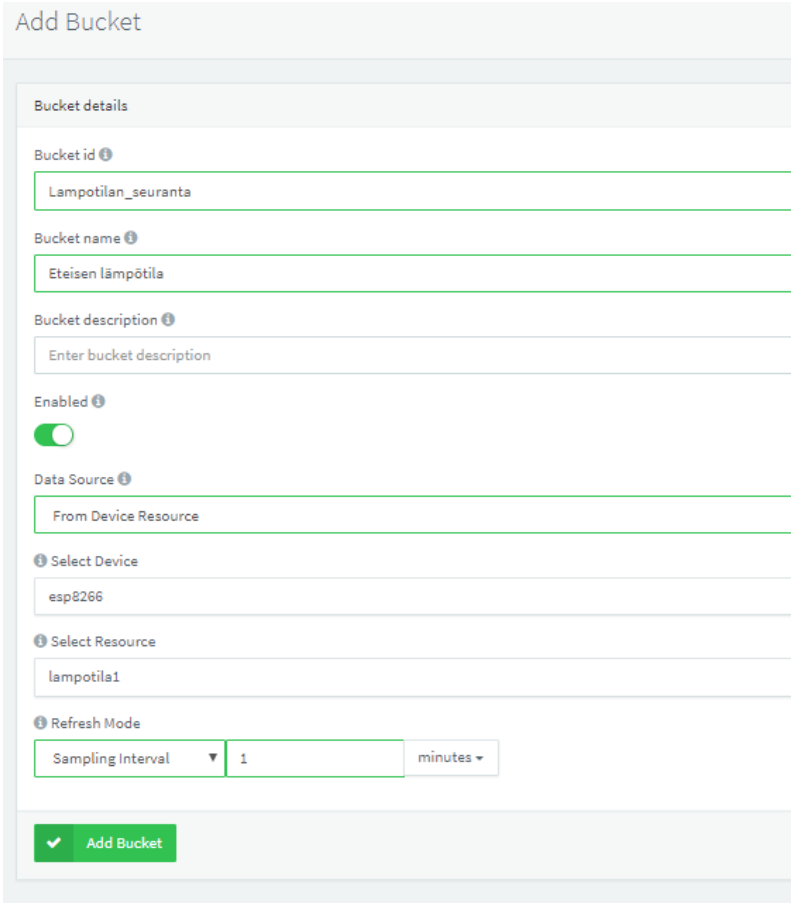


Kuvio 20. Thinger widget settingsin -määritykset

Opinnäytetyössä tehtiin eteisen sekä olohuoneen lämpötiloille tekstityyppinen ja pistekaaviotyyppinen näyttöruutu. Lisäksi moottorin pyörimisnopeudelle tehtiin rengaskaavio- ja pistekaaviotyyppiset näyttöruudut. Talon lämpötilan asetusarvolle tehtiin rengaskaaviotyyppinen näyttöruutu (ks. kuvio 21).

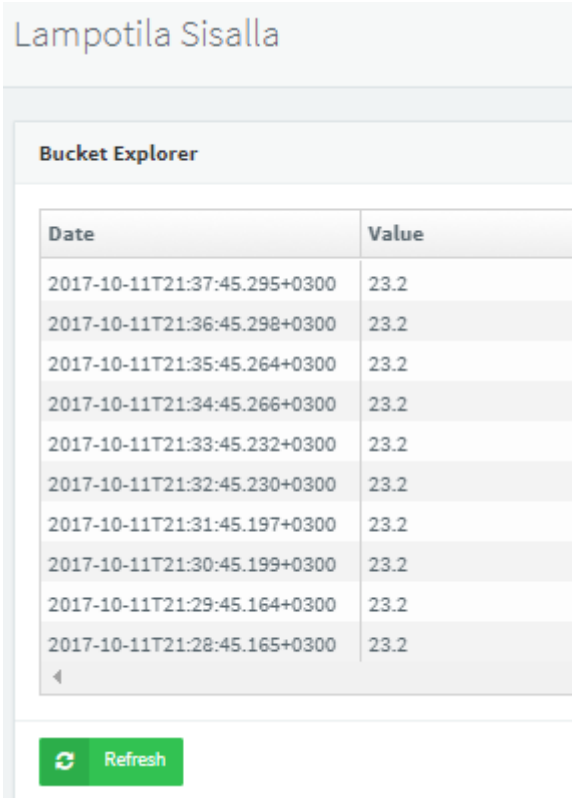
Kuvio 21. Thingerin-käyttöliittymä

Olohuoneen ja eteisen lämpötiloille on tehty lisäksi tietojen keruu. Tietojen keruu tehdään lisäämällä palveluun uusi tietopankki. Määrityksissä asetetaan tunniste, nimi ja kuvaus. Lisäksi valitaan kohde, josta tallennettava tieto saadaan ja valitaan haluttu tieto. Esimerkkikuviossa 22 tallennettava tieto luetaan esp8266-laitteelta ja tieto on eteisen lämpötila eli lampotila1. Lopuksi valitaan näytteenottoväli.



Kuvio 22. Thingerin tietojen keruu

Tallennettuja tietoja voidaan käydä lukemassa, sekä ne voidaan tuoda tietokoneelle eri tiedostomuodossa (ks. kuvio 23).

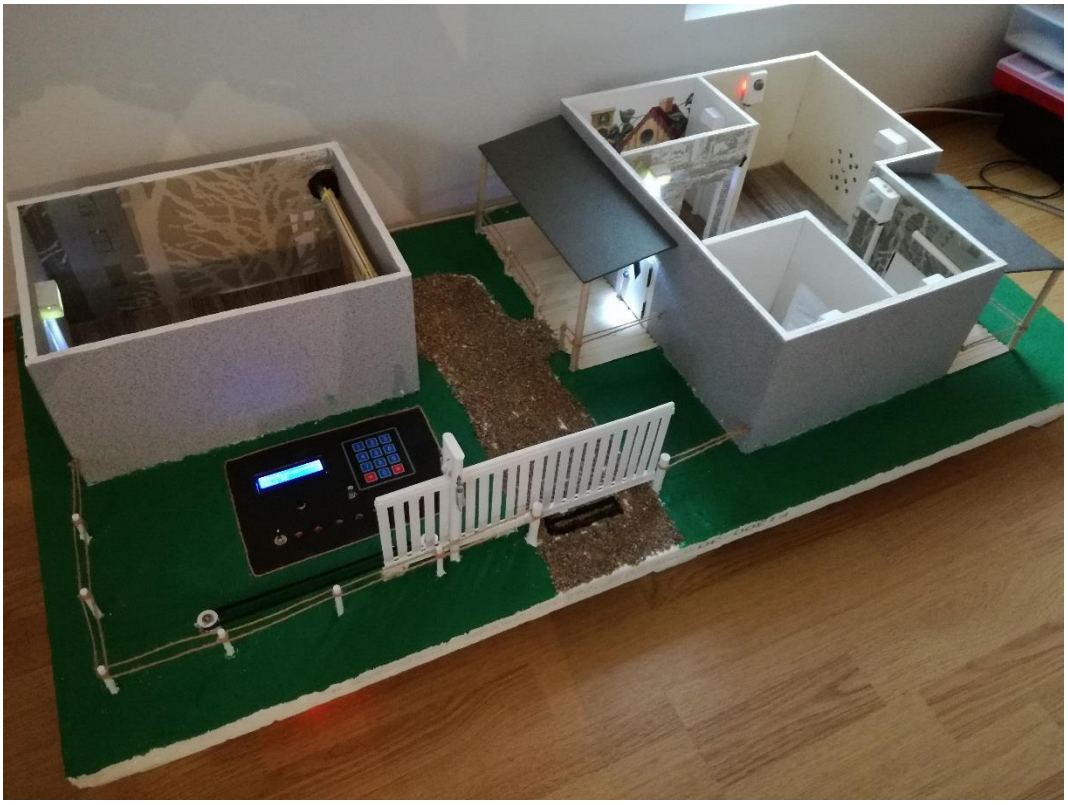


Kuvio 23. Thingeriin tallennettuja tietoja

Palvelu on hyvin monikäyttöinen eikä rajoitteita tule helposti vastaan. Laitteita on helppo liittää palveluun sekä reaaliaikainen tietojen seuraaminen käyttöliittymän avulla mahdollistaa kenen tahansa rakentaa omia IoT-laitteita. Graafisen käyttöliittymän tekeminen on tehty helppokäyttöiseksi, mikä vaikuttaa hyvin paljon sen näyttävyyteen. Muuten palvelu on osoittautunut erittäin hyväksi ja käytännölliseksi.

7 Tulokset

Opinnäytetyön lopputuloksena ja tuotoksena saatiin pienoismalli, johon on asennettu ne komponentit, joita mikrokontrolleriohjattu älytalo vaatii (ks. kuvio 24). Lisäksi saatiin kyseiseen älytalokokonaisuuteen tarvittavat koodit. Pienoismallin avulla voidaan helposti simuloida laitteiden ja koodin toimintaa.



Kuvio 24. Valmis pienoismalli

Työn alussa ja työn edetessä määriteltiin testauskriteerit, jotka työn tulisi täyttää (ks. taulukko 1).

Taulukko 1. Testauskriteerit

Tapahtuma	Toiminto	Seuraus	Kuittaus
TOTEUTUVAT AINA			
Käyttöpaneelille oikea pin-koodi	KOTI/POISSA-tila aktivoituu		KurTh
Palohälytys	Savunilmaisin tunnistaa savun	Summeri alkaa soida sekä ilmoitus sähköpostiin	KurTh
Avataan/suljetaan kauko-ohjauksella portti	Askelmoottori pyörii myötä- tai vastapäivään	Portti aukeaa/sulkeutuu	KurTh
Portin sulkeutuessa portin välissä tapahtuu liikettä	Painelaatta tunnistaa esteen	Portti pysähtyy ja avautuu tämän jälkeen itsestään.	KurTh
Avataan/suljetaan kauko-ohjauksella autotallin ovi	Askelmoottori pyörii myötä- tai vastapäivään	Ovi aukeaa/sulkeutuu	KurTh
Pilvipalvelu	Tiedot lähetetään pilvipalveluun	Tiedot voidaan lukea palvelusta	KurTh
Lämpötilan asetusarvon muutos	Säädetään asetusarvoa potentiometrillä	Asetusarvo muuttuu	KurTh
Näytön valikkorakenne	Painetaan näytön valikkonappia	Valikkorakenteen seuraava tieto näytetään näytöllä	KurTh
Sisälämpötila kasvaa	Tuuletin jäähdyttää 0...100% teholla	Lämpötila laskee	KurTh
KOTI-TILA AKTIIVINEN			
Valokatkaisija ON-asentoon	Ohjattu valo syttyy		KurTh
Ulkona hämärää	Ulkovalot syttyvät		KurTh
Eteisen valon automaattisytytys	Etuovi aukeaa ja liiketunnistin tunnistaa liikettä	Eteisen valo syttyy ja on päällä asetetun ajan verran, minkä jälkeen sammuu	KurTh
Poissa-tila aktivoituu		Valot sammuvat	KurTh
POISSA-TILA AKTIIVINEN			
Ovi avautuu	Tunnistin tunnistaa oven avautumisen	Ilmoitus sähköpostilla	KurTh
Sisällä tapahtuu liikettä	Anturi tunnistaa liikkeen	Ilmoitus sähköpostilla	KurTh
Koti-tila aktivoituu		Valot jotka olivat päällä poissa-tilan aktivoiduttua, syttyvät palamaan	KurTh

Työn edetessä kaikkien komponenttien toimivuus testattiin erikseen, mutta lopulliset testaukset työn toiminnoille tehtiin peräkkäin. Testaukset eri toiminnoille tehtiin useaan kertaan ja eri järjestyksessä. Näin pystyttiin tarkastamaan mahdollisimman monipuolisesti ristiriitaisuudet, joita ohjelmassa saattaa esiintyä. Kun jokainen laite ja ohjelma oli työn edetessä testattu toimivaksi, ongelmia ja ristiriitoja ei lopullisessa testauksessa tullut esille. Jokainen työlle asetettu kriteeri täyttyi.

Työn alussa tehdyt toimintakuvaukset oli määritelty todella tarkasti, mikä helpotti ohjelmakoodin kirjoittamista huomattavasti, eikä ristiriitaisuuksia syntynyt. Kom-

ponentit ja muut älykotijärjestelmään käytetyt tarvikkeet on valittu tavoitteiden mukaisesti. Käytetyt laitteet soveltuvat myös oikeaan ympäristöön ja ne ovat halpoja. Kokonaisuudesta tuli halpa verrattuna markkinoilla oleviin järjestelmiin ja se vastaa normaalin älykodin tarpeita. Näitä tarpeita on esimerkiksi valaistuksen ja lämpötilan ohjausta sekä lämpötilan, liikkeen ja ovien asentojen tunnistusta. Lisäksi järjestelmän muokattavuus on hyvin helppoa ja joustavaa erilaisia kohteita ajatellen avoimen lähdekoodin ja yleisien komponenttien ansiosta. Pienoismalli sisältää komponentit ja toiminnallisuudet, joita älykotijärjestelmä vaatii. Järjestelmässä on paljon eri mittauksia, ohjauksia sekä väyliä, jotka ovat liitettynä toisiinsa. Lisäksi järjestelmä on yhteydessä ulkoverkkoon mikä mahdollistaa eri tietojen lukemisen etänä.

Työssä tehty järjestelmä toteutti kaikki sille asetetut testauskriteerit, mikä on toiminut työn tulosten mittarina. Myös järjestelmän konkreettinen valmistuminen ja eri laitteiden ja väylien toimiminen yhdessä kertoo työn saavuttaneen asetetut kriteerit. Toimeksiantajan vaatimuksia työlle oli käyttää paljon erilaisia laitteita sekä väylätekniikoita. Lisäksi pilvipalveluiden hyödyntäminen oli kriteerinä. Työssä on käytetty paljon erilaisia antureita, jotka mittaavat tietoa ja lisäksi toimilaitteina on käytetty erilaisia laitteita. Opinnäytetyö on myös yhteydessä ulkoverkkoon kahteen eri palveluun. Myöskin toimeksiantajalta saadut kriteerit täyttyivät.

Työn suurimpia etuja markkinoilla oleviin järjestelmiin verrattuna on sen hinta. Työn loppuvaiheella saatiin tarjouslaskelma markkinoilla olevasta älykotijärjestelmästä, joka vastaa toiminnoiltaan ja laajuudeltaan opinnäytetyössä tehtyä järjestelmää. Tarjouslaskelma sisälsi laitteet, joita järjestelmä vaatii ja sen hinta-arvio oli 2000- 3000€. Opinnäytetyössä käytettyjen laitteiden hinnat ovat yhteensä 220€. Laitteiden hinnat on laskettu Suomalaisen verkkokaupan hintojen mukaan. Kummassakaan hinta-arviossa ei ole mukana johdotuksia eikä töitä. Älykotijärjestelmän lopullinen hinta riippuu toimilaitteista, kuten esimerkiksi porttia ja autotallin ovea liikuttavista moottoreista. Myöskään näitä toimilaitteita ei ole otettu hinta-arvioissa huomioon.

Työssä valmistunut kotiautomaatiojärjestelmä on markkinoilla oleviin järjestelmiin paljon halvempi ja laajentumiskykyisempi. Jos kotiautomaatiojärjestelmän omaavaan taloon halutaan jälkikäteen lisätä valaistuksien ohjausta, pistorasioiden ohjausta tai mittalaitteita, niiden lisääminen opinnäytetyössä valmistuneeseen järjestelmään on hyvin halpaa ja helppoa. Markkinoilla olevat järjestelmät myyvät valmiita paketteja,

minkä vuoksi yksittäisien laitteiden asennus jälkikäteen ei ole yhtä helppoa. Opinnäytetyössä käytetyt laitteet ovat hyvin yleisiä, joten niiden saatavuus on mahdollista myös vuosien päästä.

Opinnäytetyön tuloksena saatu järjestelmä ei ole kuitenkaan valmis tuote, jota voisi myydä sellaisenaan, vaan järjestelmässä käytetyt laitteet tulisi koteloida ja suunnitella helposti asennettavaksi. Lisäksi porttia ja autotallin ovea ohjaavat askelmoottorit ovat vain simulointia varten, eikä toimi oikeassa ympäristössä. Muutoin laitteet ja ohjelmakoodi ovat siirrettävissä sellaisenaan oikeaan ympäristöön. Tulevaisuudessa mikrokontrolleriohjattuja älykotijärjestelmiä tulevat hyödyntämään tavalliset ihmiset ja yritykset, jotka haluavat edullisen kotiautomaatiojärjestelmän. Tavalliset ihmiset eivät suoraan pysty asentamaan järjestelmää itselleen, kuten eivät pysty asentamaan myöskään markkinoilla olevia älykotijärjestelmiä, vaan he hyötyvät järjestelmästä yrityksiltä ostetun järjestelmän kautta. Lisäksi järjestelmää voi hyödyntää kuka tahansa asiasta kiinnostunut ja ohjelmoinnin osaava pienemmissäkin automaatiota vaativissa kohteissa.

Mikrokontrolleriohjatuissa kotiautomaatiojärjestelmissä on todella suuri bisnesmahdollisuus sen joustavuuden ja edullisuuden vuoksi. Mikrokontrollereilla voidaan toteuttaa markkinoilla oleviin järjestelmiin verrattuna paljon edullisemman ja joustavamman automaatiojärjestelmän. Toimeksiantaja saa käyttöönsä piirikaaviot, ohjelmakoodin sekä laiteluettelon. Näiden avulla järjestelmä voidaan rakentaa, soveltaa erilaisiin kohteisiin ja tutkia tarkemmin eri mahdollisuuksia.

8 Pohdinta

Opinnäytetyön alussa asetettiin tavoitteiksi selvittää ja kokeilla, onnistuuko halvoilla mikrokontrollereilla ja antureilla toteuttaa älytalo, joka vastaa nykyajan älytalon tarpeisiin. Älytalon perustarpeet ovat lämmityksen mittaus ja säätö, valaistuksen ohjaus sekä muita yksittäisiä elämää helpottavia toimintoja. Tuloksena saatu pienoismalli, joka simuloi älytaloa, sisältää nämä perustarpeet ja lisänä paljon muita ominaisuuksia. Lisäksi ohjelman ja järjestelmän muokattavuus on huomioitu laitteita valittaessa.

Työn edetessä tuli uusia tai vaihtoehtoisia ideoita toteuttaa eri toimintoja, mikä vaikutti ohjelmakoodin muutoksiin sekä laitteiden vaihtoihin. Työssä joutui perehtymään hyvin paljon uusiin laitteisiin ja niiden toimintoihin. Lisäksi erilaisien pilvipalveluiden tutkiminen ja vertaileminen oli suuressa roolissa. Erilaisien vaihtoehtojen määrä järjestelmän toteuttamiseksi oli huikea, mikä vaati kompromissien tekemistä ja niissä pysymistä. Toisaalta taas uusien näkökulmien ja ideoiden ilmetessä oli oltava nöyrä ja joustaa omista päätöksistä. Kokonaisuutena työ onnistui loistavasti. Kaikki työlle asetetut tavoitteet onnistuivat ja lisäksi tuli paljon muita toimintoja.

Työssä käytetyt lähdemateriaalit ovat suurilta osin opetusmateriaalia ja opetuskirjoja. Lisäksi lähteinä on käytetty laitteiden keksijöiden ja rakentajien kirjoittamia kirjoja. Muita lähteitä ovat mielipidehaastattelut, joita on tarkasteltu kriittisesti ja vertailtu keskenään. Opinnäytetyössä saatua tulosta on myös tarkasteltu kriittisesti ja se on osoittautunut luotettavaksi ja käyttökelpoiseksi. Ohjelmallisia ristiriitoja ja ongelmia ei ole testauksien yhteydessä ilmennyt.

Pienoismalliin liitettyä älykotijärjestelmää voidaan hyödyntää oikeassa ympäristössä sellaisenaan. Lisäksi järjestelmä on muokattavissa erilaisiin kohteisiin. Älykotijärjestelmän laajentamismahdollisuudet ovat rajattomat, minkä vuoksi työn jatkokehittäminen on vain mielikuvituksesta kiinni. Järjestelmään voisi liittää esimerkiksi audiojärjestelmän, talon etähallintajärjestelmän, ovipuhelinjärjestelmän tai erillisen murto/palohälytysjärjestelmän. Eri järjestelmien yhteensopivuus ja niiden sulauttaminen yhteen antaa opinnäytetyölle lisäarvoa, sillä tulevaisuudessa valmiin kotiautomaatiojärjestelmän rinnalle voidaan asentaa jälkikäteen helposti myös muita edellä mainittuja järjestelmiä. Mikrokontrolleripohjaista järjestelmää voi hyödyntää jo olemassa olevat yritykset, jotka myyvät älykotijärjestelmiä. Esimerkiksi asiakas voi haluta jälkikäteen automaatiota myös ulkorakennuksiin, mikä on halpa toteuttaa opinnäytetyössä syntyneellä järjestelmällä.

Koko opinnäytetyön ajan olen pohtinut, että miksi tällaisia mikrokontrolleri-ohjattuja halpoja järjestelmiä ei vielä ole markkinoilla. En ole löytänyt suoraa vastausta kysymykseen, mutta uskon kysymyksen tulevan turhaksi muutamien vuosien kuluttua. Kun jotakin uutta ja vierasta esitellään ihmisille, jotka ovat tottuneet vanhoihin tapoihin, uusiin asioihin suhtaudutaan usein ennakkoluuloisesti. Tämä

ennakkoluuloisuus älykotijärjestelmiä ja automaattisia laitteita kohtaan on väistymässä ja se tuo laitteet yhä lähemmäksi ihmistä. Tämä taas lisää halpojen järjestelmien kysyntää, minkä jälkeen näitä aletaan tekemään.

Lähteet

Aaltonen, J., Kousa, S. & Stor-Pellinen, J. 2000. Elektroniikan perusteet. 2. korj. p. Helsinki: Limes.

Arduino Serial. 2017. Dokumentti. Viitattu 13.11.2017. https://www.arduino.cc/en/Reference/Serial

Banzi, M. 2011. Arduino perusteista hallintaan. Hämeenlinna: Robomaa.

Berghäll, L. 2012. Taloautomaatio ohjaa kodin laitteita. Älykäs koti helpottaa elämää. Anna-lehti 18.12.2012. Viitattu 20.9.2017. https://anna.fi/sisustus/sisustus/taloautomaatio-ohjaa-kodin-laitteita-alykas-koti-helpottaa-elamaa

Cirovic, M. 1979. Basic Electronics. Virginia: Reston.

Dimitrov, K. 2016. Arduino Thermometer With SD18B20. Arduino project hub-sivusto. Viitattu 15.11.2017. https://create.arduino.cc/projecthub/TheGadgetBoy/ds18b20-digital-temperature-sensor-and-arduino-9cc806

Koskinen, M. 2002. Analogia suunnittelu. Helsinki: Sanoma Magazines Finland.

MQ-2 Semiconductor Sensor for Combustible Gas. N.d. Dokumentti. Viitattu 13.11.2017. https://www.pololu.com/file/0J309/MQ2.pdf

Nedelkovski, D. N.d. Arduino Wireless Communication-NRF24L01 Tutorial. How To Mechatronics-sivusto. Viitattu 13.11.2017. http://howtomechatronics.com/tutorials/arduino/arduino-wireless-communication-nrf24l01-tutorial/

Nedelkovski, D. N.d. How I2C Communication Works and How To Use It with Arduino. How To Mechatronics-sivusto. Viitattu 13.11.2017. http://howtomechatronics.com/tutorials/arduino/how-i2c-communication-works-and-how-to-use-it-with-arduino/

Nodeon yritys lyhyesti. 2017. Nodeon Oy. Viitattu 20.9.2017. http://www.nodeon.com/fi/yritys/lyhyesti/

Opinnäytetyön suunnitelman laatiminen. N.d. Opinnäytetyön ohjeet. Viitattu 21.11.2017. https://oppimateriaalit.jamk.fi/yamk-kasikirja/opinnaytetyo-prosessina/opinnaytetyon-suunnitelman-laatiminen/

Ryynänen, T. 2017. Näin suojaat yksityisyytesi netisssä. Tekniikkaa ja taloutta -blogisivusto. Haaga-Helian ammattikorkeakoulu. Viitattu 18.10.2017. https://blogit.haaga-helia.fi/ryynanen/2017/03/16/nain-suojaat-yksityisyytesi-netissa/

Stepper Motor Basics. N.d. CircuitSpecialists-sivusto. Viitattu 15.11.2017. https://www.circuitspecialists.com/stepper-motor

Taanila, I. 2016. Internet of Things (IoT). IoT Finland 16.3.2016. Viitattu 18.10.2017. http://iotfinland.fi/internet-of-things-iot/

The Piezoelectric Effect. N.d. Nanomotion-sivusto. Viitattu 15.11.2017. http://www.nanomotion.com/piezo-ceramic-motor-technology/piezoelectric-effect/

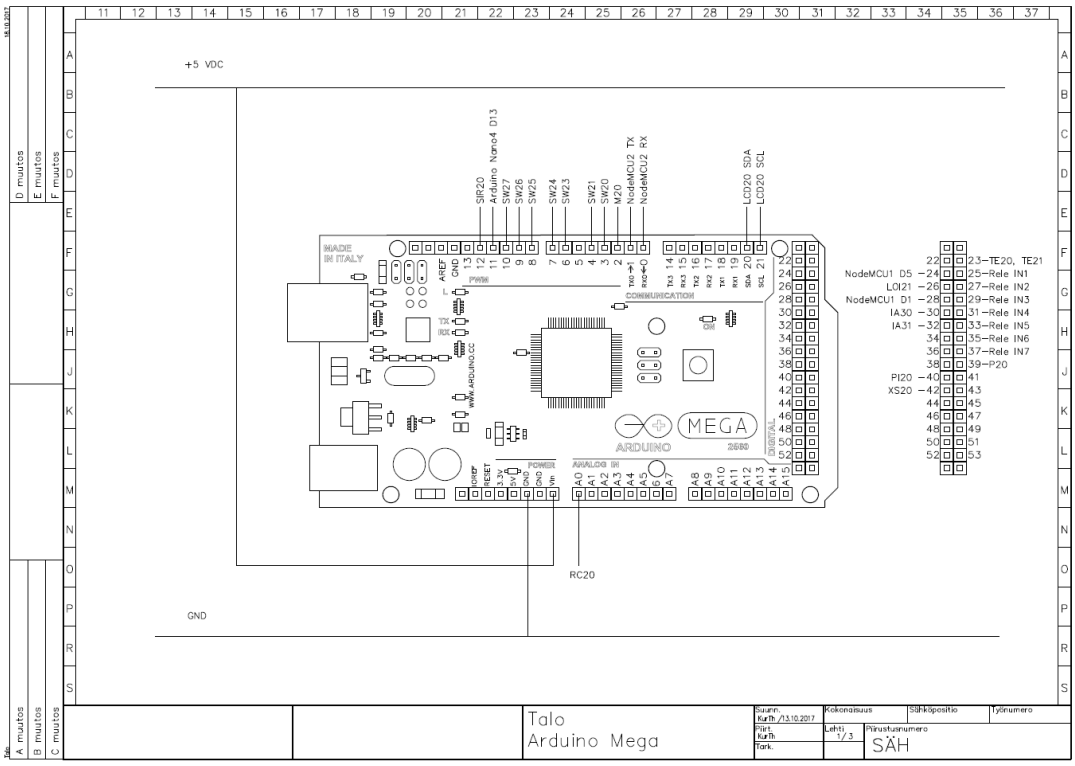
Liitteet

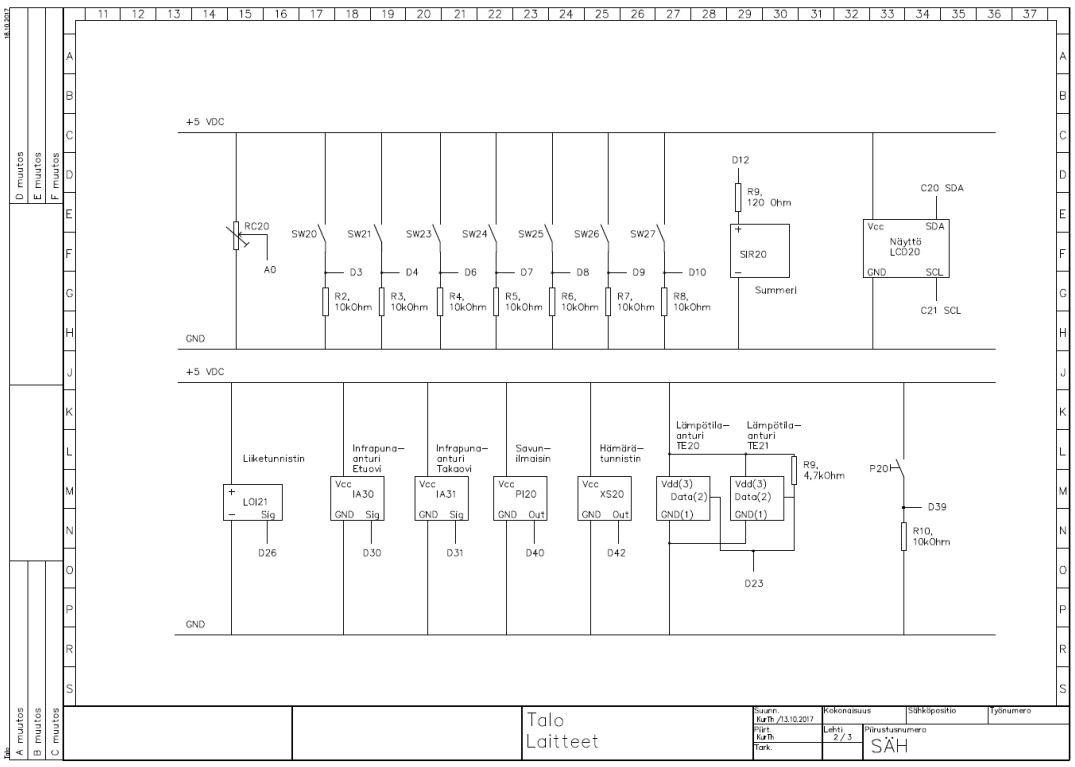
Liite 1. Laiteluettelo

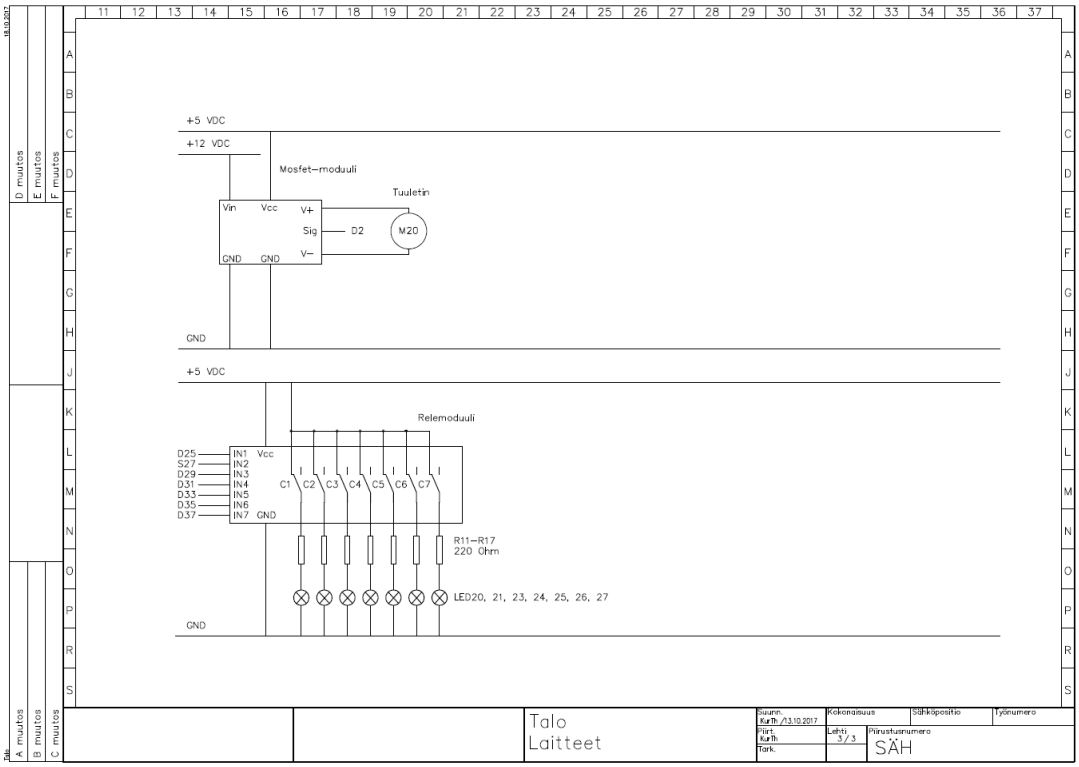
Laite	Tunnus/ Ryhmä	Ohjaus/Toiminta	Skaalaus	Kohde	Kuvaus
Talo	**20**				
Valaistus 1	LED 20	1/0		Makuuhuone 1	
Valaistus 2	LED 21	1/0		Makuuhuone 2	
Valaistus 4	LED 23	1/0		Eteinen	
Valaistus 5	LED 24	1/0		Olohuone	
Valaistus 6	LED 25	1/0		Keittiö	
Valaistus 7	LED 26	1/0		Terassi 1	
Valaistus 8	LED 27	1/0		Terassi 2	
Liiketunnistin 1	LOI 21	1/0		Olohuone	Tunnistaa liikkeen olohuoneessa
Ovikytkin 1	IA 30	1/0		Etuovi	Tunnistaa oven avauksen
Ovikytkin 2	IA 31	1/0		Takaovi	Tunnistaa oven avauksen
Savun ilmaisin 1	PI 20	1/0		Olohuone	Tunnistaa savun
Lämpötilamittaus 1	TE 20	0...5V	-10...+85C	Eteinen	Mittaa eteisen lämpötilaa
Lämpötilamittaus 2	TE 21	0...5V	-10...+85C	Olohuone	Mittaa olohuoneen lämpötilaa
Tuuletin	M 20	0...12V	0...100%	Olohuone	Jäähdyttää taloa
Summeri	SIR 20	1/0		Olohuone	Hälyttää varkaista sekä tulipalosta
Hämärätunnistin	XS20	1/0		Ulkoseinä	Tunnistaa hämärän
Käyttöpaneeli					
Kytkin 1	SW 20	ON/OFF		Makuuhuone 1	Valojen ohjaus
Kytkin 2	SW 21	ON/OFF		Makuuhuone 2	Valojen ohjaus
Kytkin 4	SW 23	ON/OFF		Eteinen	Valojen ohjaus
Kytkin 5	SW 24	ON/OFF		Olohuone	Valojen ohjaus
Kytkin 6	SW 25	ON/OFF		Keittiö	Valojen ohjaus
Kytkin 7	SW 26	ON/OFF		Terassi 1	Valojen ohjaus
Kytkin 8	SW 27	ON/OFF		Terassi 2	Valojen ohjaus
Kytkin 9	SW28	ON/OFF		Autotalli	Valojen ohjaus
Painonappi 3	P 20	1/0			Valikon selaus
LCD näyttö	LCD 20				Näyttää käyttäjälle tietoja
Potentiometri	RC 20	0...10000Ohm	18C...26C	Asetuslämpötila	Voi asettaa halutun lämpötilan taloon
Portti	**40**				
Askelmoottori	M 40			Portille	Avaa sekä sulkee porttia
Indikointi	P 40	1/0		Portille	Tunnistaa liikkeen portilla

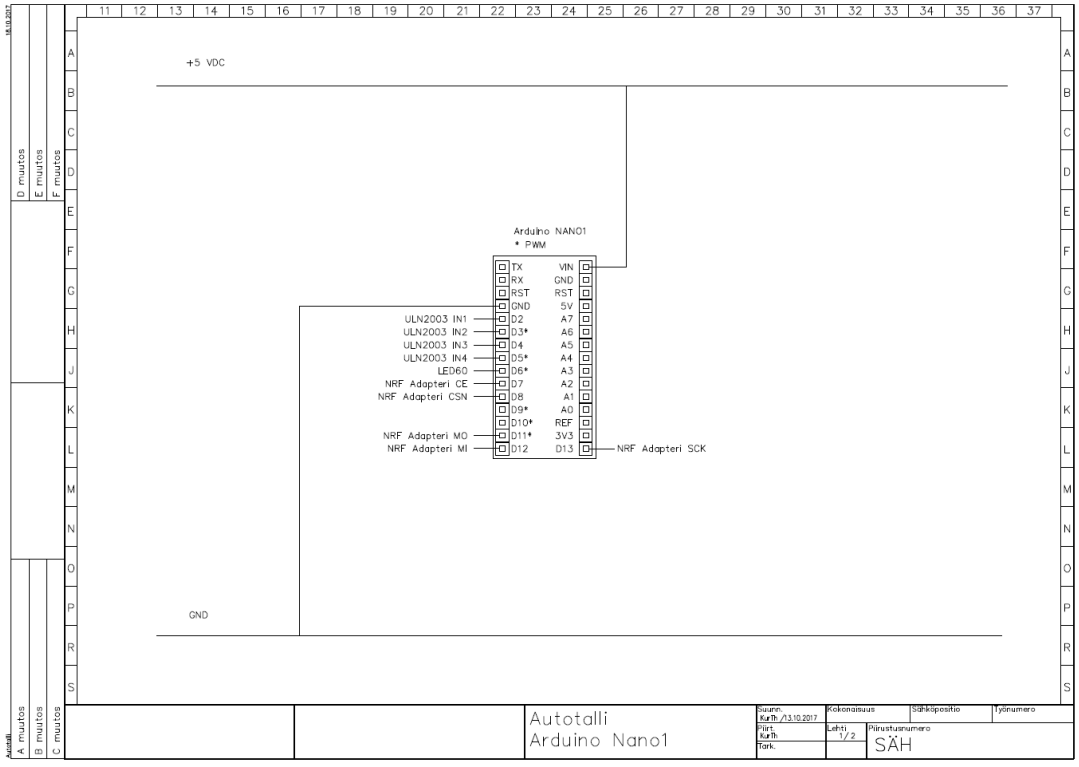
Autotalli	**60**				
Valaistus 10	LED 60	1/0		Autotalli	
Askelmoottori	M 60			Autotallin ovelle	Avaa sekä sulkee autotallin ovea
Kaukosäädin	**100**				
Painonappi 1	P 100	1/0		Moottori portti/autota	Moottori avaa portin/oven
Painonappi 2	P 101	1/0		Moottori portti/autota	Moottori sulkee portin/oven
Kytkin 10	SW 100	1/OFF/2		Moottori portti/autota	Kytkimellä valitaan ohjataanko autallin ovea vai porttia.
Näppäinpaneeli	**120**				
Näppäinpaneeli	NP 120				Kytkee kotona/poissa tilan.

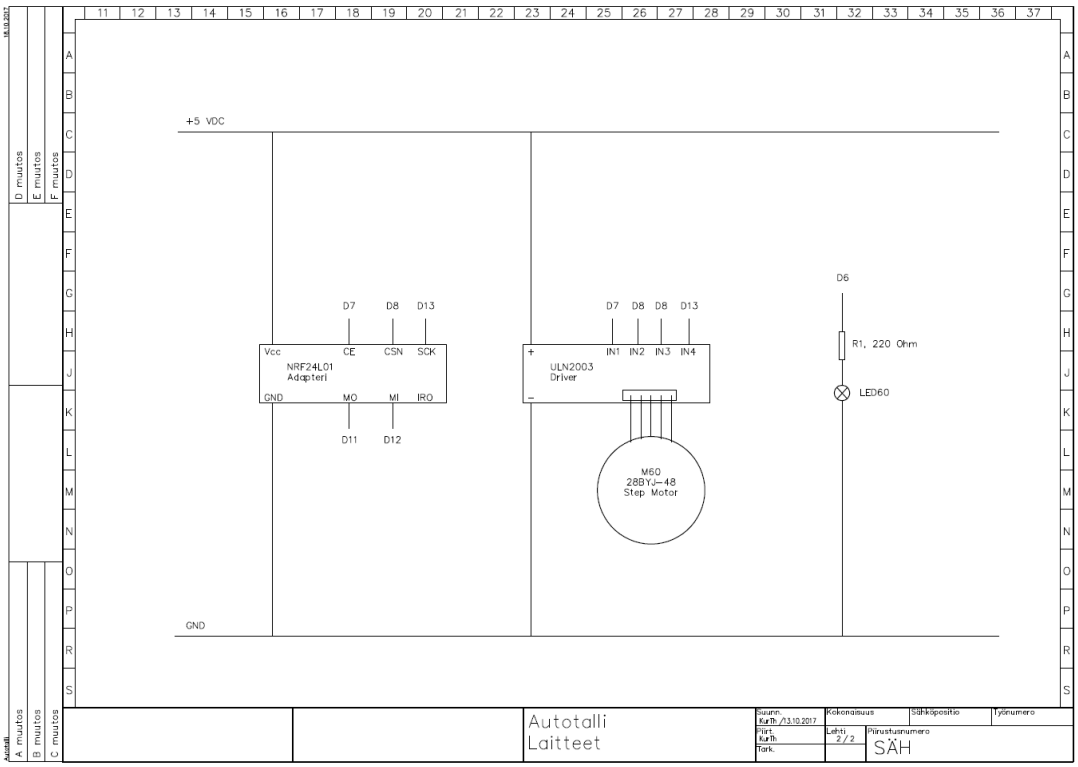
Liite 2. Piirikaaviot

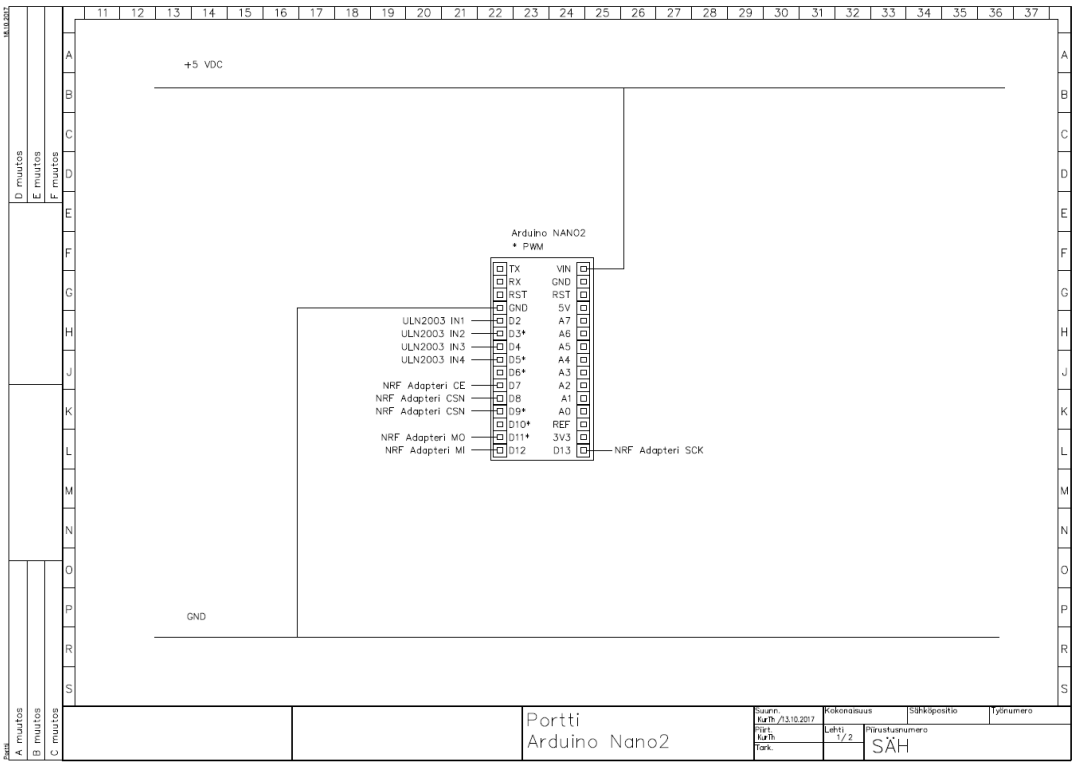


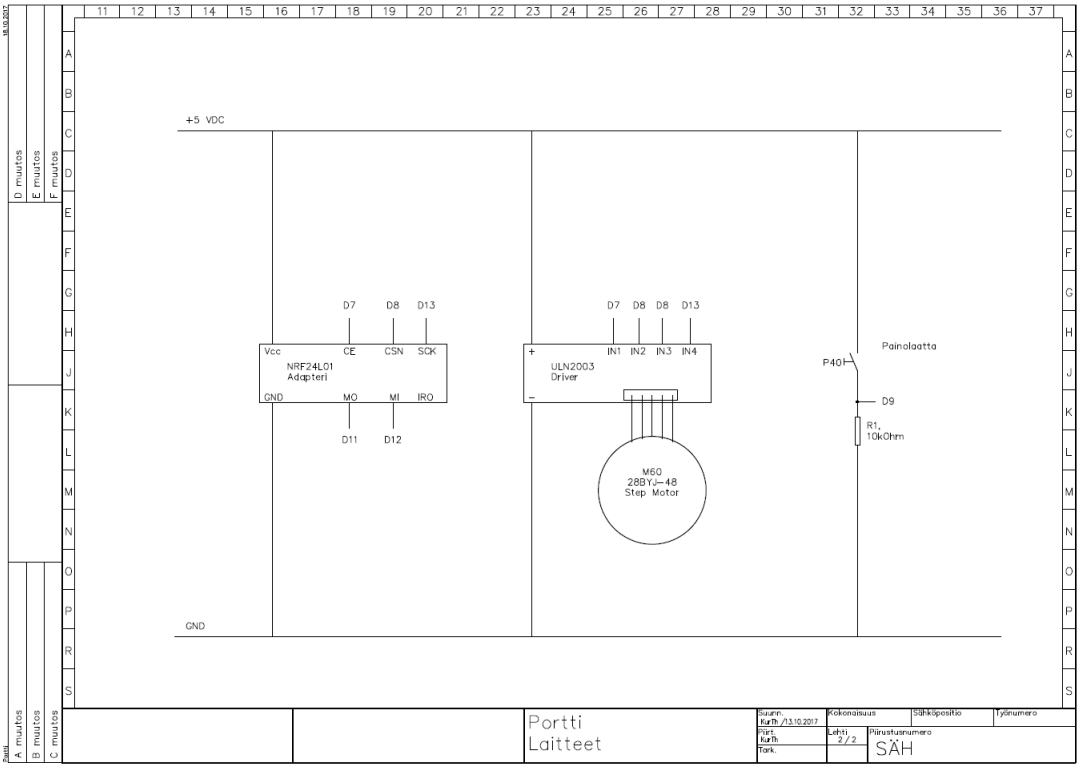


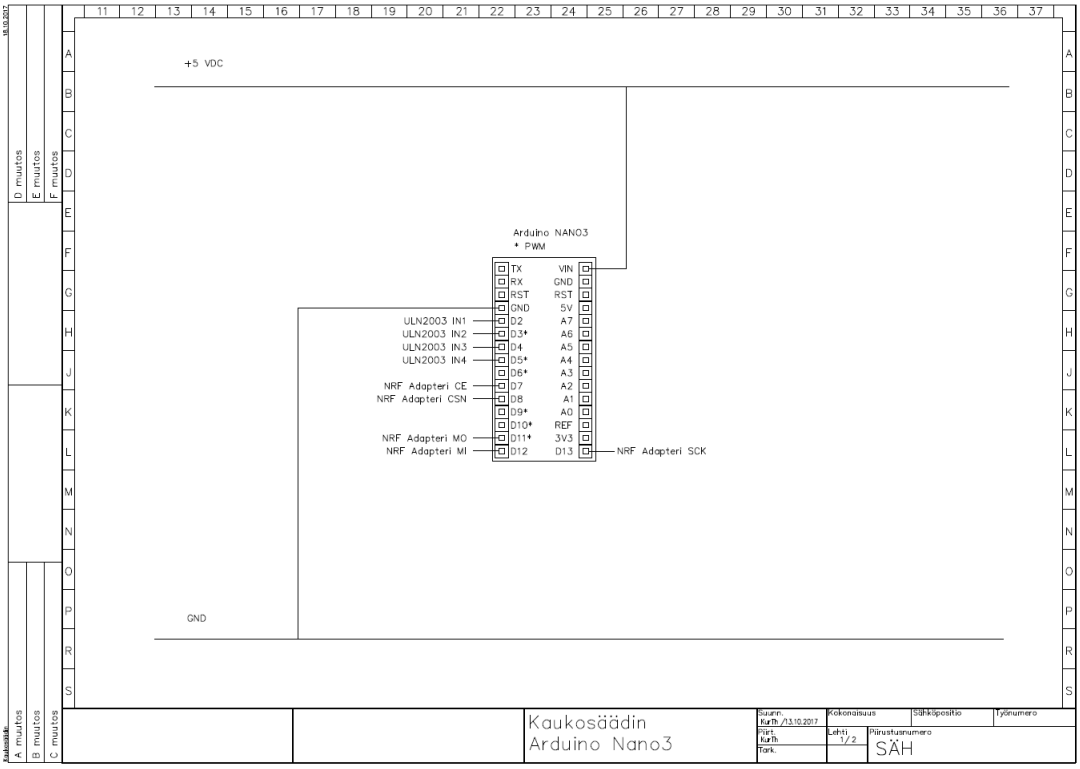

+5 VDC
+12 VDC
Mosfet–moduuli
Tuuletin
Vin
Vcc
V+
Sig
D2
M20
V–
GND
GND
+5 VDC
Relemoduuli
D25
S27
D29
D31
D33
D35
D37
IN1
IN2
IN3
IN4
IN5
IN6
IN7
Vcc
GND
C1
C2
C3
C4
C5
C6
C7
R11–R17
220 Ohm
LED20, 21, 23, 24, 25, 26, 27
GND
Talo
Laitteet
SÄH

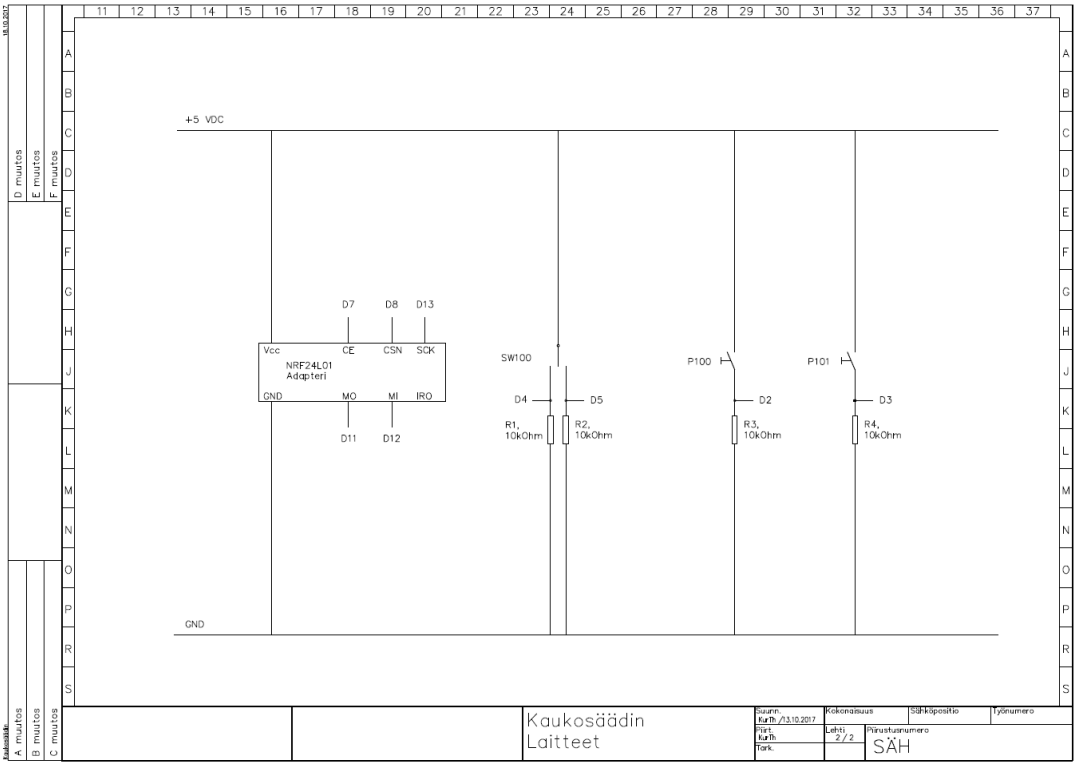

+5 VDC
Arduino NANO1
* PWM
TX
RX
RST
GND
D2
D3*
D4
D5*
D6*
D7
D8
D9*
D10*
D11*
D12
VIN
GND
RST
5V
A7
A6
A5
A4
A3
A2
A1
A0
REF
3V3
D13
ULN2003 IN1
ULN2003 IN2
ULN2003 IN3
ULN2003 IN4
LED60
NRF Adapteri CE
NRF Adapteri CSN
NRF Adapteri MO
NRF Adapteri MI
NRF Adapteri SCK
GND
Autotalli
Arduino Nano1
SÄH

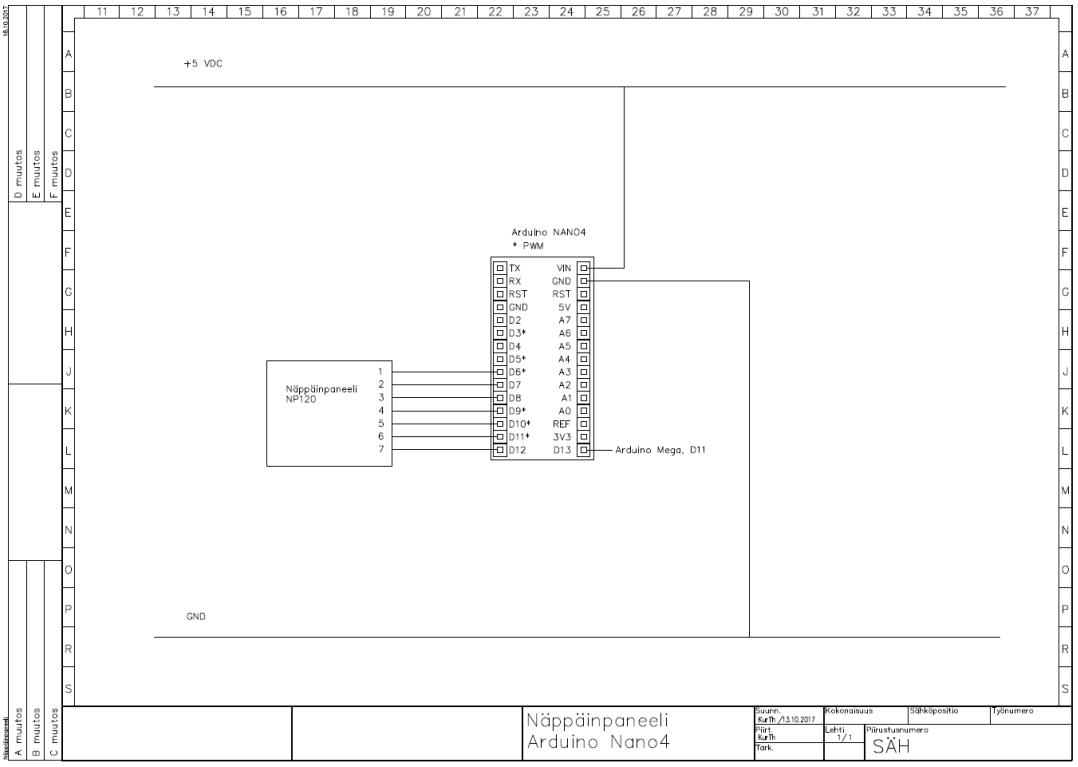

+5 VDC
D7
D8
D13
Vcc
CE
CSN
SCK
NRF24L01
Adapteri
GND
MO
MI
IRQ
D11
D12
D7
D8
D8
D13
+
IN1
IN2
IN3
IN4
ULN2003
Driver
–
M60
28BYJ-48
Step Motor
D6
R1, 220 Ohm
LED60
GND
Autotalli
Laitteet
Suunn. Kurth. /3.10.2017
Kokonaisuus
Sähköpositio
Työnumero
Piirt.
Kurth.
Tark.
Lehti 2/2
Piirustusnumero
SÄH

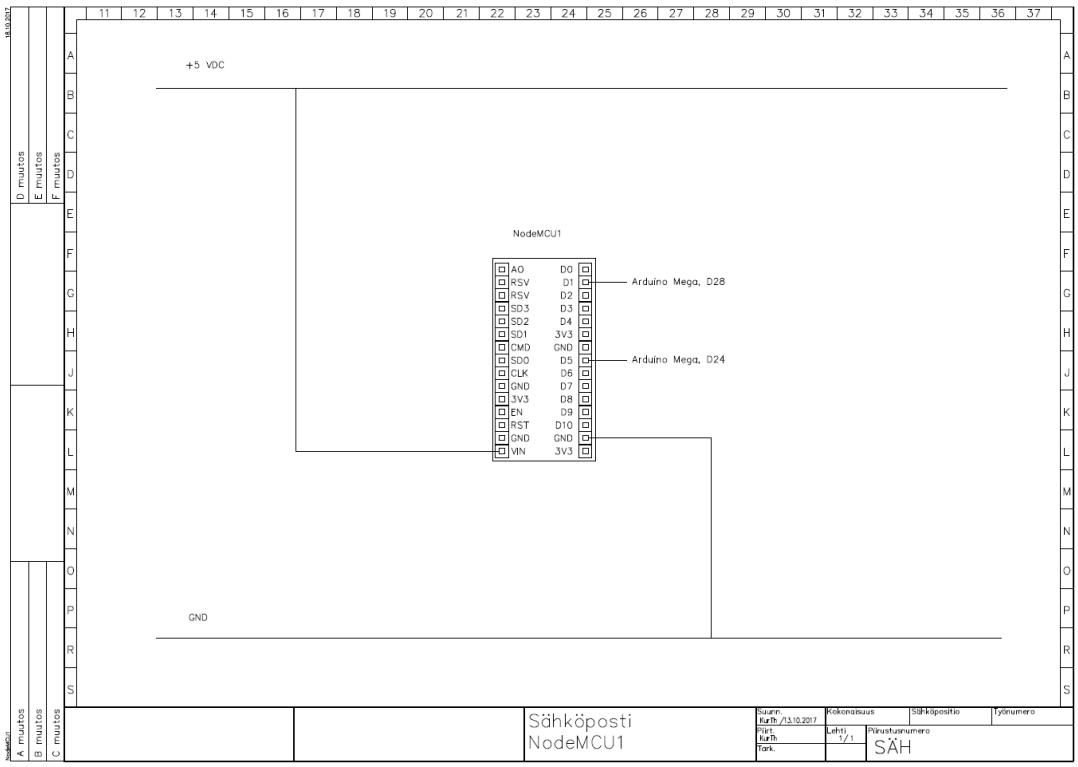

+5 VDC
Arduino NANO2
+ PWM
TX
RX
RST
GND
D2
D3*
D4
D5*
D6*
D7
D8
D9*
D10*
D11*
D12
VIN
GND
RST
5V
A7
A6
A5
A4
A3
A2
A1
A0
REF
3V3
D13
ULN2003 IN1
ULN2003 IN2
ULN2003 IN3
ULN2003 IN4
NRF Adapteri CE
NRF Adapteri CSN
NRF Adapteri CSN
NRF Adapteri MO
NRF Adapteri MI
NRF Adapteri SCK
GND
Portti
Arduino Nano2
Suunn. Kurth. /3.10.2017
Kokonaisuus
Sähköpositio
Työnumero
Piirt.
Kurth.
Tark.
Lehti 1/2
Piirustusnumero
SÄH

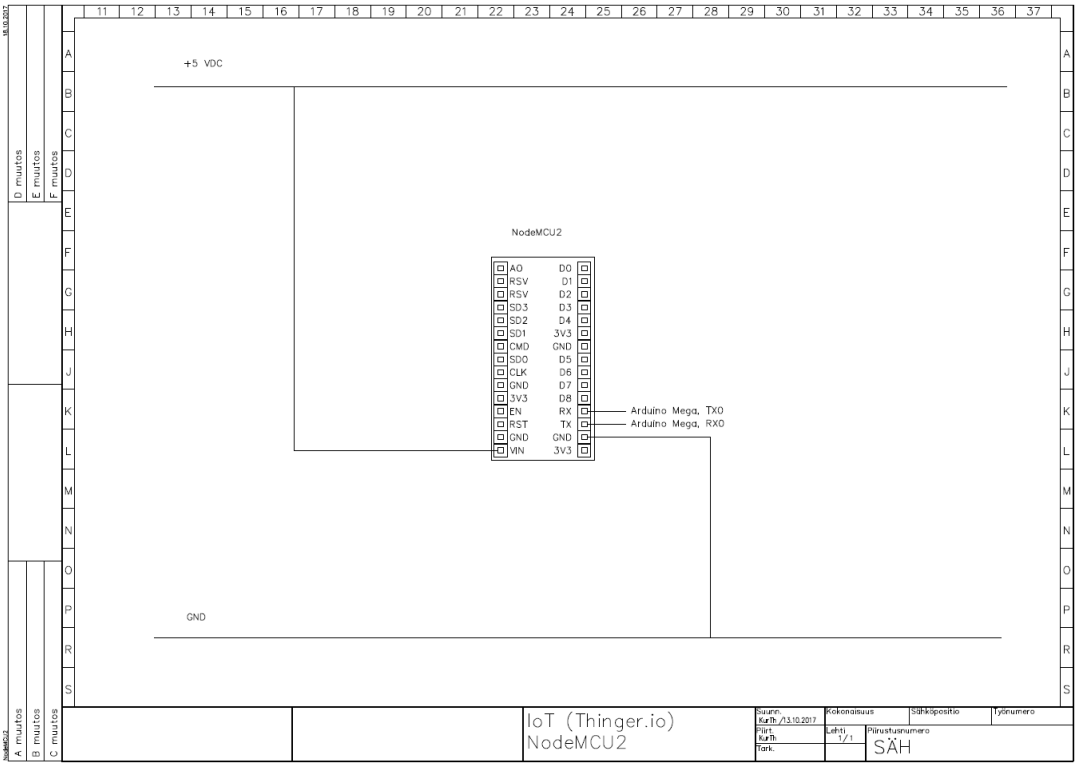

+5 VDC
D7
D8
D13
Vcc
CE
CSN
SCK
NRF24L01
Adapteri
GND
MO
MI
IRO
D11
D12
D7
D8
D8
D13
+
IN1
IN2
IN3
IN4
ULN2003
Driver
–
M60
28BYJ–48
Step Motor
Painolaatta
P40
D9
R1,
10kOhm
GND
Portti
Laitteet
SÄH


+5 VDC
Arduino NANO3
* PWM
TX
RX
RST
GND
D2
D3*
D4
D5*
D6*
D7
D8
D9*
D10*
D11*
D12
VIN
GND
RST
5V
A7
A6
A5
A4
A3
A2
A1
A0
REF
3V3
D13
ULN2003 IN1
ULN2003 IN2
ULN2003 IN3
ULN2003 IN4
NRF Adapteri CE
NRF Adapteri CSN
NRF Adapteri MO
NRF Adapteri MI
NRF Adapteri SCK
GND
Kaukosäädin
Arduino Nano3
SÄH


+5 VDC
D7
D8
D13
Vcc
CE
CSN
SCK
NRF24L01
Adapteri
GND
MO
MI
IRO
D11
D12
SW100
D4
D5
R1,
10kOhm
R2,
10kOhm
P100
D2
R3,
10kOhm
P101
D3
R4,
10kOhm
GND
Kaukosäädin
Laitteet
SÄH


+5 VDC
Arduino NANO4
* PWM
TX
RX
RST
GND
D2
D3*
D4
D5*
D6*
D7
D8
D9*
D10*
D11*
D12
VIN
GND
RST
5V
A7
A6
A5
A4
A3
A2
A1
A0
REF
3V3
D13
Arduino Mega, D11
Näppäinpaneeli
NP120
1
2
3
4
5
6
7
GND
Näppäinpaneeli
Arduino Nano4
SÄH


+5 VDC
NodeMCU1
AO D0
RSV D1
RSV D2
SD3 D3
SD2 D4
SD1 3V3
CMD GND
SD0 D5
CLK D6
GND D7
3V3 D8
EN D9
RST D10
GND GND
VIN 3V3
Arduino Mega, D28
Arduino Mega, D24
GND
Sähköposti
NodeMCU1
SÄH


+5 VDC
NodeMCU2
AO D0
RSV D1
RSV D2
SD3 D3
SD2 D4
SD1 3V3
CMD GND
SD0 D5
CLK D6
GND D7
3V3 D8
EN RX
RST TX
GND GND
VIN 3V3
Arduino Mega, TX0
Arduino Mega, RX0
GND
IoT (Thinger.io)
NodeMCU2
SÄH

Liite 3. Talon ohjelmakoodi

```
/*
   OPINNÄYTETYÖ 2017
   KOTIAUTOMAATIOJÄRJESTELMÄ

   TALO
   Thomas Kuronen
   10/2017

*/

//SERIAL
char lahtevaTieto[5];                       //Määritetään lahtevaTieto-taulukon koko.

//VARAS
boolean rosvo = false;                      //Määritetään rosvo-muuttuja booleaniksi ja alustetaan se FALSEksi.

//VALIKKONAPPI
int btn = 39;                               //Määritetään valikkonapin (btn, PN20) pinni.
int btnTila = 0;                            //Määritetään btnTila-muuttuja integeriksi ja alustetaan se arvolla 0.
int btnEdellinenTila = 0;                   //Määritetään btnEdellinenTila-muuttuja integeriksi ja alustetaan se arvolla 0.

//LIIKETUNNISTIN
int LT20 = 26;                              //Määritellään liiketunnistimen (LT20) pinni.
boolean liiketta;                           //Määritellään liikettä-muuttuja BOOLEANiksi.

//SÄHKÖPOSTI
int spostiInd = 24;                         //Määritellään spostiInd pinni.
int pHalyInd = 28;                          //Määritellään pHalyInd pinni.

//KOTONA/POISSA
int kotonaPoissa = 11;                      //Määritellään kotonaPoissa pinni.
boolean kotona = true;                      //Määritellään kotona-muuttuja BOOLEANiksi ja alustetaan se TRUEKSI.
boolean edellinenKotona = false;            //Määritellään edellinenKotona-muuttuja BOOLEANiksi ja alustetaan se FALSEKSI.
boolean kotonaApu = false;                  //Määritellään kotonaApu-muuttuja BOOLEANiksi ja alustetaan se FALSEKSI.
int kotonaApuLaskuri = 0;                   //Määritellään kotonaApuLaskuri-muuttuja INTEGERiksi ja alustetaan se arvolla 0.

//NÄYTTÖ
#include <Wire.h>                                       //LCD-Näytön tarvitsema kirjasto.
#include <LiquidCrystal_I2C.h>                          //LCD-Näytön tarvitsema kirjasto.
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);    //Rakennetaan näytön ominaisuudet ja annetaan sille nimi lcd.
int caseLuku = 1;                          //Määritellään casLuku-muuttuja INTEGERiksi ja alustetaan se arvolla 1.

//HÄMÄRÄTUNNISTIN
int HT20 = 42;                             //Määritellään hämärätunnistimen (HT20) pinni.
boolean hamaraa;                           //Määritellään hamaraa-muuttuja BOOLEANiksi.



//SAVUN ILMAISIN
int SI20 = 40;                              //Määritellään savunilmaisimen (SI20) pinni.
boolean savua;                              //Määritellään savua-muuttuja BOOLEANiksi.

//SUMMERI
const int S20 = 12;                        //Määritellään summerin (S20) pinni.

//OVET
int SW30 = 30;                             //Määritellään etuoven kytkimen (SW30) pinni.
int SW31 = 32;                             //Määritellään takaoven kytkimen (SW31) pinni.
boolean etuovi;                            //Määritellään etuovi-muuttuja BOOLEANiksi.
boolean takaovi;                           //Määritellään takaovi-muuttuja BOOLEANiksi.

//VALOT
int LED20 = 25;                            //Määritellään makuuhuoneen (LED20) valon pinni.
int LED21 = 27;                            //Määritellään makuuhuoneen (LED21) valon pinni.
int LED23 = 29;                            //Määritellään eteisen (LED23) valon pinni.
int LED24 = 31;                            //Määritellään olohuoneen (LED24) valon pinni.
int LED25 = 33;                            //Määritellään keittiön (LED25) valon pinni.
int LED26 = 35;                            //Määritellään terassin (LED26) valon pinni.
int LED27 = 37;                            //Määritellään terassin (LED27) valon pinni.
int apumuuttujaEteinen = 0;                //Määritellään apumuuttujaEteinen-muuttuja integeriksi ja alustetaan se arvolla 0.

//KATKAISIJAT
int SW20 = 3;                              //Määritellään makuuhuoneen (SW20) katkaisijan pinni.
int SW21 = 4;                              //Määritellään makuuhuoneen (SW21) katkaisijan pinni.
int SW23 = 6;                              //Määritellään eteisen (SW23) katkaisijan pinni.
int SW24 = 7;                              //Määritellään olohuoneen (SW24) katkaisijan pinni.
int SW25 = 8;                              //Määritellään keittiön (SW25) katkaisijan pinni.
int SW26 = 9;                              //Määritellään terassin (SW26) katkaisijan pinni.
int SW27 = 10;                             //Määritellään terassin (SW27) katkaisijan pinni.

//POTIKKA
int M20pot = A0;                           //Määritellään potentiometrin (M20pot) pinni.
double M20asetusarvo;                      //Määritellään M20asetusarvo-muuttuja DOUBLEksi.
double M20asetusarvoApu;                   //Määritellään M20asetusarvoApu-muuttuja DOUBLEksi.

//TUULETIN M20
#include <PID_v1.h>                        //PID-säätimen kirjasto
const int M20 = 2;                         //Määritellään moottorin (M20) pinni.
double prosentti;                          //Määritellään prosentti-muuttuja DOUBLEksi.
double Setpoint, Input, Output;            //Määritellään Setpoint-, Input- ja Output-muuttujat DOUBLEksi.
PID myPID(&Input, &Output, &Setpoint, 2, 4, 0, REVERSE);    //Rakennetaan PID-säädin ja annetaan sille nimi myPID.

/*Säädin viritetty kokeilemalla. Lopulliset arvot: Kp=2, Ki=4, Kd=0.
 * Suunta on asetettu REVERSE:ksi, jotta Inputin kasvaessa myös Output kasvaa.
 */
double M20lahto;                           //Määritellään M20lahto DOUBLEksi.
```

```
//LÄMPÖTILA-ANTURI
#include <OneWire.h>                    //Lämpötila-anturin tarvitsema kirjasto
#include <DallasTemperature.h>          //Lämpötila-anturin tarvitsema kirjasto
#define ONE_WIRE_BUS 23                 //Määritellään lämpötila-antureiden pinni.
OneWire oneWire(ONE_WIRE_BUS);          //Määritellään OneWire väylän pinni.
DallasTemperature sensors(&oneWire);    //Määritellään antureiden väylä ja annetaan sille nimeksi sensors.
double lampotila1;                      //Määritellään eteisen lämpötila-muuttuja (lampotila1) DOUBLEksi.
double lampotila2;                      //Määritellään olohuoneen lämpötila-muuttuja (lampotila2) DOUBLEksi.
double keskiLampotila;                  //Määritellään keskiLampotila-muuttuja DOUBLEksi.

//AJASTIMET
unsigned long edellinenMillis1 = 0;     //Määritellään eri muuttujat UNSIGNED LONGiksi ja alustetaan ne arvolla 0.
unsigned long edellinenMillis2 = 0;
unsigned long edellinenMillis3 = 0;
unsigned long edellinenMillis4 = 0;
const long aika20ms = 20;               //Määritellään aika20ms-muuttuja CONST LONGiksi ja alustetaan se arvolla 20.
const long aika100ms = 100;             //Määritellään aika100ms-muuttuja CONST LONGiksi ja alustetaan se arvolla 100.
const long aika500ms = 500;             //Määritellään aika500ms-muuttuja CONST LONGiksi ja alustetaan se arvolla 500.
const long aika1000ms = 1000;           //Määritellään aika1000ms-muuttuja CONST LONGiksi ja alustetaan se arvolla 1000.


void setup()
{

  Serial.begin(9600);                   //Asetetaan serialin datansiirtonopeus bittejä sekunissa (baud).

  //TUULETIN M20
  pinMode(M20, OUTPUT);                 //Määritellään pinni OUTPUTIKSI.
  myPID.SetMode(AUTOMATIC);             //Asetetaan myPID-säädin toimimaan automaattisesti.

  //LÄMPÖTILA-ANTURI
  sensors.begin();                      //Käynnistetään lämpötilan mittaus.

  //POTIKKA
  pinMode(M20pot, INPUT);               //Määritellään pinnit OUTPUTIKSI tai INPUTIKSI.

  //OVET
  pinMode(SW30, INPUT);
  pinMode(SW31, INPUT);

  //VALAISTUS
  pinMode(LED20, OUTPUT);
  pinMode(LED21, OUTPUT);
  pinMode(LED23, OUTPUT);
  pinMode(LED24, OUTPUT);
  pinMode(LED25, OUTPUT);
  pinMode(LED26, OUTPUT);
  pinMode(LED27, OUTPUT);

  //KYTKIMET
  pinMode(SW20, INPUT);
  pinMode(SW21, INPUT);
  pinMode(SW23, INPUT);
  pinMode(SW24, INPUT);
  pinMode(SW25, INPUT);
  pinMode(SW26, INPUT);
  pinMode(SW27, INPUT);

  //HÄMÄRÄTUNNISTIN
  pinMode(HT20, INPUT);

  //SAVUN ILMAISIN
  pinMode(SI20, INPUT);

  //SUMMERI
  pinMode(S20, OUTPUT);
```

```
  //NÄYTTÖ
  lcd.begin(16, 2);                         //Määritellään LCD-näytön sarakkeet ja rivit.
  lcd.backlight();                          //Käynnistetään LCD-näytön taustavalo.
  lcd.setCursor(0, 0);                      //Asetetaan LCD-näytön kursori 0-sarakkeelle ja 0-riville.
  lcd.print("Alykoti");                     //Kirjoitetaan näytölle teksti.
  delay(1000);                              //Odotetaan 1000ms.
  lcd.setCursor(0, 1);                      //Asetetaan LCD-näytön kursori 0-sarakkeelle ja 1-riville.
  lcd.print("-ToKu 2017");                  //Kirjoitetaan näytölle teksti.
  delay(2000);                              //Odotetaan 2000ms.
  lcd.clear();                              //Tyhjennetään näyttö.

  //KOTONA/POISSA
  pinMode(kotonaPoissa, INPUT);             //Määritellään pinnit OUTPUTIKSI tai INPUTIKSI.

  //Sähköposti
  pinMode(spostiInd, OUTPUT);
  pinMode(pHalyInd, OUTPUT);

  //LIIKETUNNISTIN
  pinMode(LT20, INPUT);

  //VALIKKONAPPI
  pinMode(btn, INPUT);
}


/*****************************************************PÄÄOHJELMAKIERTO*****************************************************************************/

void loop()
{
  //AJASTIMET
  unsigned long nykyinenMillis = millis();               //Kirjoitetaan ajastimen arvo millisekunteina nykyinenMillis-muuttujaan.
  /*****************************************************20ms*****************************************************************************/

  if (nykyinenMillis - edellinenMillis1 >= aika20ms)      //Jos nykyinenMillis - edellinenMillis1 >= aika20ms. (IF-rakenne toteutuu 20 ms välein)
  {
    edellinenMillis1 = nykyinenMillis;                    //Kirjoitetaan nykyinenMillis arvo muuttujaan edellinenMillis1.
    //TÄHÄN KOHTAA KOODI
  }
  /*****************************************************100ms****************************************************************************/
  if (nykyinenMillis - edellinenMillis2 >= aika100ms)    //Jos nykyinenMillis - edellinenMillis2 >= aika100ms. (IF-rakenne toteutuu 100 ms välein)
  {
    edellinenMillis2 = nykyinenMillis;                   //Kirjoitetaan nykyinenMillis arvo muuttujaan edellinenMillis2.
    //TÄHÄN KOHTAA KOODI
  }
  /*****************************************************500ms****************************************************************************/
  if (nykyinenMillis - edellinenMillis3 >= aika500ms)   //Jos nykyinenMillis - edellinenMillis3 >= aika500ms. (IF-rakenne toteutuu 500 ms välein)
  {
    edellinenMillis3 = nykyinenMillis;                  //Kirjoitetaan nykyinenMillis arvo muuttujaan edellinenMillis3.
    //TÄHÄN KOHTAA KOODI
  }
  /*****************************************************1000ms***************************************************************************************************/
  if (nykyinenMillis - edellinenMillis4 >= aika1000ms)    //Jos nykyinenMillis - edellinenMillis4 >= aika1000ms. (IF-rakenne toteutuu 1000 ms välein)
  {
    edellinenMillis4 = nykyinenMillis;                    //Kirjoitetaan nykyinenMillis arvo muuttujaan edellinenMillis4.

    //TÄHÄN KOHTAA KOODI

    Lampotilanmittaus();                                  //Suoritetaan aliohjelmat.
    Tuuletin();
    Potikka();
    Ovet();
    Hamaratunnistin();
    Savunilmaisin();
    Liiketunnistin();
    Summeri();
    Valot();
    KotiPoissa();
    Varas();
    Valikkonappi();
    Naytto();
    Thinger();
  }

}
```

```
/*********************************************************ALIOHJELMAT*************************************************************************************************/
void Lampotilanmittaus()
{
  sensors.requestTemperatures();
  lampotila1 = (sensors.getTempCByIndex(0));            //(Eteinen) Luetaan lämpötila-anturilta (Index 0) arvo ja talletetaan muuttujaan.
  lampotila2 = (sensors.getTempCByIndex(1));            //(Olohuone) Luetaan lämpötila-anturilta (Index 1) arvo ja talletetaan muuttujaan.
  keskiLampotila = (lampotila1 + lampotila2) / 2;       //Lasketaan lämpötilan keski-arvo ja talletetaan muuttujaan.
  keskiLampotila = (keskiLampotila - 15) * 17;          //(Keskilampotila = 0...255 (15C...30C)) Muutetaan keskiLampotila-muuttujan arvo PID-säätimelle sopivaksi.
}

/*
 * Lampotilanmittaus-aliohjelmalla mitataan lämpötila eteisestä sekä olohuoneesta.
 * Lämpötiloista lasketaan keskilämpötila joka on välillä 15C-30C ja se skaalataan
 * 0-255 välille jotta se on oikeassa asteikossa PID-säätimelle.
 */


void Tuuletin()
{
  Input = keskiLampotila;                               //Kirjoitetaan keskiLampotila PID-säätimen Inputiksi.
  myPID.Compute();                                      //Suoritetaan myPID-laskenta.
  M20lahto = map(Output, 0, 255, 100, 255);             //Skaalataan myPID-säätimeltä saatu (0...255) Output arvo sopivaksi (100...255) ja kirjoitetaan se muuttujaan M20lahto.

  if (M20lahto == 100)                                  //Tarkastetaan onko M20lahto 100 eli pienin mahdollinen arvo.
  {
    analogWrite(M20, 0);                                //Kirjataan M20 lähtöön arvo 0. (Tuuletin ei pyöri)
  }
  else
  {
    analogWrite(M20, M20lahto);                         //Jos M20lahto ei ole 100 vaan jotain muuta, niin kirjataan M20 lähtöön kyseinen M20lahto.
  }
  prosentti = map(M20lahto, 100, 255, 0, 100);          //Skaalataan pyörimisnopeus (100...255) prosentti-muuttujaan sopivaksi(0%...100%).
}

/*
 * Tuuletin-aliohjelma suorittaa myPID-säätimen laskennan ja ohjaa tuuletinta.
 * Jos myPID-säätimeltä saadaan ohjausarvoksi 0, niin tuulettimen asetusarvoksi asetetaan silloin 100. Tällöin tuuletin ei vielä pyöri.
 * Kun myPID-säätimeltä saadaan ohjausarvoksi 1, niin tuulettimen asetusarvoksi asetetaan silloin noin 102.
 * Tällä menetelmällä estetään tuulettimen hyvin pieni pyöriminen ja säästetään moottoria.
 * prosentti-muuttujaa käytetään näyttämään pyörimisnopeus prosentteina käyttäjälle.
 */



void Potikka()
{
  M20asetusarvo = analogRead(M20pot);                       //Luetaan potentiometrin (POT20) arvo ja kirjoitetaan se muuttujaan M20asetusarvo.

  M20asetusarvoApu = (M20asetusarvo / 127.875) + 18;        //Skaalataan M20asetusarvo (0...1023) sopivaksi muuttujaan M20asetusarvoApu. (18C...26C)
                                                            //M20asetusarvoApu-muuttujaa käytetään asetuslämpötilan näyttämiseen käyttäjälle.

  M20asetusarvo = map(M20asetusarvo, 0, 1023, 51, 187);     //Skaalataan M20asetusarvo (0...1023) sopivaksi muuttujaan M20asetusarvo. (51...187)
                                                            //M20asetusarvo- muuttujaa käytetään asetuslämpötilan kertomiseen PID-säätimelle.
                                                            //Luku 51 tarkoittaa 18 celciusta ja luku 187 takoittaa 26 celciusta.


  Setpoint = M20asetusarvo;                                 //Kirjoitetaan M20asetusarvo myPID-säätimen Setpointiksi.
}

/* Potikka-aliohjelma tarkastaa potentiometrin asennon mitä käytetään lämpötilan asetusarvon säätämiseen.
 * Potentiometrin arvo on välillä 0-1023 ja M20asetusarvoApu-muuttujaan se skaalataan välille 18C-26C. Skaalaus tapahtuu jakamalla
 * potentiometriltä saatu arvo luvulla 127.875 ja tämän jälkeen lisäämällä siihen luku 18. Tämä skaalattu arvo näytetään käyttäjälle.
 * M20asetusarvo pitää skaalata sopivaan muotoon myPID-säätimelle. Arvo skaalataan väliltä 0-1023 välille 51-187.
 * Nämä 51 ja 187 ovat raja-arvoja, jotka tarkoittavat lämpötilana 18 celciusta ja 26 celsiusta. Tälle välille asetusarvo halutaan.
 */

void Ovet()
{
  etuovi = digitalRead(SW30);                       //Luetaan SW30 (etuoven asentotunnistin) tilatieto etuovi-muuttujaan.
  takaovi = digitalRead(SW31);                      //Luetaan SW31 (takaoven asentotunnistin) tilatieto takaovi-muuttujaan.
}                                                    //Kun ovi on auki, niin muuttuja on TRUE, muutoin FALSE.
```

```
void Valot()
{

  if (kotona)                                   //Jos kotona-muuttuja on TRUE, eli ollaan kotona, niin menemme if-rakenteeseen.
  {                                             //Kaikki valaistukset toimivat samalla tavalla muutamaa poikkeusta lukuunottamatta
    //********MAKUUHUONE1***************/
    if (digitalRead(SW20) == HIGH)              //Jos kytkin SW20 (makuuhuone) on HIGH-tilassa, niin asetamme lähdön LED20 (makuuhuone)
    {                                           //HIGH-tilaan. Muutoin lähtö LED20 on LOW-tilassa.
      digitalWrite(LED20, HIGH);
    }
    else
    {
      digitalWrite(LED20, LOW);
    }
    //********MAKUUHUONE2***************/
    if (digitalRead(SW21) == HIGH)
    {
      digitalWrite(LED21, HIGH);
    }
    else
    {
      digitalWrite(LED21, LOW);
    }

    //********ETEINEN***************/
    if (digitalRead(SW23) == HIGH)              //Jos kytkin SW23 on HIGH-tilassa niin asetetaan lähtö LED23 HIGH-tilaan.
    {
      digitalWrite(LED23, HIGH);
    }
    else if (etuovi and liiketta)               //Jos etuovi on auki ja sisällä tapahtuu liikettä, niin menemme if-rakenteeseen.
    {
      apumuuttujaEteinen = 1;                   //Kirjoitetaan muuttujaan apumuuttujaEteinen arvo 1.
      digitalWrite(LED23, HIGH);                //Asetetaan lähtö LED23 HIGH-tilaan.
    }

    else if (apumuuttujaEteinen >= 1)           //Tässä if-lohkossa on suoritettu viivyttelytoiminto. Tämä toteutuu jos apumuuttujaEteinen >=1
    {
      apumuuttujaEteinen++;                     //Lisätään muuttujaan apumuuttujaEteinen jokaisella kierroksella +1.
      digitalWrite(LED23, HIGH);                //Pidetään lähtö LED23 HIGH-tilassa.
      if (apumuuttujaEteinen == 10)             //Kun apumuuttujaEteinen = arvo 10. Eli ohjelmakierto huomioon ottaen noin 10 sekuntia,
      {
        apumuuttujaEteinen = 0;                 //Kirjoitetaan apumuuttujaEteiseen arvo 0.
      }
    }

    else                                        //Jos mikään eteisen if-rakenne ei toteudu, niin valo sammutetaan.
    {
      digitalWrite(LED23, LOW);                 //Asetetaan lähtö LED23 LOW-tilaan.
    }
```

```
    //********OLOHUONE***************/
    if (digitalRead(SW24) == HIGH)
    {
      digitalWrite(LED24, HIGH);
    }
    else
    {
      digitalWrite(LED24, LOW);
    }
    //********KEITTIÖ***************/
    if (digitalRead(SW25) == HIGH)
    {
      digitalWrite(LED25, HIGH);
    }
    else
    {
      digitalWrite(LED25, LOW);
    }
    //********TERASSI 1***************/
    if (digitalRead(SW26) == HIGH or hamaraa)           //Jos kytkin SW26 on HIGH-tilassa tai hamaraa muuttuja = TRUE
    {
      digitalWrite(LED26, HIGH);                        //Lähtö LED26 asetetaan HIGH-tilaan
    }
    else
    {
      digitalWrite(LED26, LOW);
    }
    //********TERASSI 2***************/
    if (digitalRead(SW27) == HIGH or hamaraa)
    {
      digitalWrite(LED27, HIGH);
    }
    else
    {
      digitalWrite(LED27, LOW);
    }
  }
  else                                                  // Jos kotona-muuttuja on FALSE eli ei olla kotona, niin sammutamme kaikki valot
  {
    digitalWrite(LED20, LOW);
    digitalWrite(LED21, LOW);
    digitalWrite(LED23, LOW);
    digitalWrite(LED24, LOW);
    digitalWrite(LED25, LOW);
    digitalWrite(LED26, LOW);
    digitalWrite(LED27, LOW);
  }
```

```
void Hamaratunnistin()
{
  hamaraa = digitalRead(HT20);                    //Luetaan hämärätunnistimen (HT20) tilatieto hamaraa-muuttujaan.
}                                                 //Jos ulkona on hämärää, niin hamaraa muuttuja on TRUE.

void Savunilmaisin()
{
  savua = digitalRead(SI20);                      //Luetaan savunilmaisimen (SI20) tieto savua-muuttujaan.

  if (savua == 0)                                 //Ilmaisin toimii "väärinpäin", eli kun ilmaisin tunnistaa savua,
  {                                               //niin sen lähtö on FALSE. Jos savua, menemme if-rakenteeseen.
    digitalWrite(pHalyInd, HIGH);                 //Asetetaan lähtö pHalyInd HIGH-tilaan. Tällä kerrotaan NODEMCU-laitteelle palohälystä.
  }
  else
  {
    digitalWrite(pHalyInd, LOW);                  //Jos ei tunnisteta savua, niin asetetaan lähtö pHalyInd LOW-tilaan.
  }
}

void Liiketunnistin()
{
  liiketta = digitalRead(LT20);                   //Luetaan liiketunnistimen (LT20) tilatieto liiketta-muuttujaan.
}

void Summeri()
{
  if (savua == 0 or rosvo)                        //Jos tunnistetaan savua (savua = 0) tai talossa on varkaita (rosvo = 1), menemme if-rakenteeseen.
  {
    tone(S20, 1000);                              //Asetetaan tone-toiminnolla summerin (S20) lähtöön 1kHz taajuudella kanttiaaltoa.
  }
  else
  {
    noTone(S20);                                  //Lopetetaan kanttiaallon lähetys.
  }
}



void KotiPoissa()
{

  kotona = digitalRead(kotonaPoissa);             //Luetaan kotonapoissa-muuttujan tilatieto ja kirjoitetaan se kotona-muuttujaan.
                                                  //kotonaPoissa-tilatieto saadaan Arduino Nano 4:lta joka tunnistaa näppäinpaneelin (NP120) painallukset.
  if (kotona != edellinenKotona)                  //Tarkastetaan onko kotona-muuttuja erisuuri kuin edellinenKotona.
  {
    if ( kotonaApuLaskuri < 5)                    //Jos kotonaApuLaskuri-muuttujan arvo < 5, niin mennään if-rakenteeseen.
    {
      kotonaApu = true;                           //Kirjoitetaan kotonaApu-muuttujan tilaksi TRUE.
      kotonaApuLaskuri++;                         //Lisätään kotonaApuLaskuriin +1;
    }
    else                                          //Jos kotonaApuLaskuri = 5, menemme if-rakenteeseen.
    {
      kotonaApu = false;                          //Kirjoitetaan kotonaApu-muuttujaan FALSE.
      kotonaApuLaskuri = 0;                       //Kirjoitetaan kotonaApuLaskuri-arvoksi 0;
      edellinenKotona = kotona;                   //Kirjoitetaan edellinenKotona-muuttujaan kotona-muuttujan tila.
    }
  }
}

/*
 * KotiPoissa-aliohjelmalla tarkastetaan onko kirjauduttu kotia vai pois sieltä.
 * Aliohjelmassa on apumuuttujia ja laskuri, joiden avulla saadaan kirjautumiset näkymään käyttäjälle useamman sekunin ajan.
 * kotonaApuLaskuri-muuttujaa käytetään viive-toiminnossa joka ohjaa kotonaApu-apumuuttujaa.
 * kotonaApu-apumuuttuja taas ohjaa Näyttö()-aliohjelmassa tapahtuvaa kirjautumisnäyttöä.
 * Kun kirjaudutaan kotia tai pois kotoa, niin kotonaApu = TRUE noin 5 sekuntia.
 */

void Varas()
{
  if (kotona == LOW and (etuovi or takaovi or liiketta) )     //Jos ei olla kotona ja etuovi tai takaovi on auki tai sisällä tapahtuu liikettä,
  {
    digitalWrite(spostiInd, HIGH);                            //Asetetaan spostiInd-lähtö HIGH-tilaan.
    rosvo = true;                                             //Asetetaan rosvo-muuttuja TRUE-tilaan.
  }
  else                                                        //Jos if-ehto ei toteudu (ei ole varkaita)
  {
    digitalWrite(spostiInd, LOW);                             //Asetetaan spostiInd-lähtö LOW-tilaan.
    rosvo = false;                                            //Asetetaan rosvo-muuttuja FALSE-tilaan.
  }
}



void Naytto()                                                 //Jokaiselle prioriteetti 1-toiminnolle on oma ehto-lauseke.
{                                                             //Yleisille näytettäville asioille on CASE-rakenne.


  if (savua == 0)                                             //Jos savua-muuttuja = 0 (havaitaan savua).
  {
    lcd.setCursor(0, 0);                                      //Asetetaan LCD20-näytön kursori ensimmäiselle riville ja ensimmäiseen sarakkeeseen.
    lcd.print("*****Alykoti*****");                           //Kirjoitetaan näytölle teksti.
    lcd.setCursor(0, 1);                                      //Asetetaan LCD20-näytön kursori toiselle riville ja ensimmäiseen sarakkeeseen.
    lcd.print("***TULIPALO!*****");                           //Kirjoitetaan näytölle teksti.
  }

  else if (rosvo)                                             //Jos rosvo-muuttuja = TRUE (Varkaita talossa).
  {
    lcd.setCursor(0, 0);
    lcd.print("*****Alykoti*****");
    lcd.setCursor(0, 1);
    lcd.print("***VARKAITA!*****");
  }

  else if (kotona and kotonaApu)                              //Jos kotona- ja kotonaApu-muuttujat ovat TRUE-tilassa.
  {                                                           //"Kirjauduttu kotia"
    lcd.setCursor(0, 0);
    lcd.print("***Kotona-tila**");
    lcd.setCursor(0, 1);
    lcd.print("***aktiivinen***");
  }
  else if (kotona == false and kotonaApu)                     //Jos kotona-muuttuja on FALSE ja kotonaApu-muuttuja on TRUE
  {                                                           //"Kirjauduttu ulos kodista"
    lcd.setCursor(0, 0);
    lcd.print("***Poissa-tila**");
    lcd.setCursor(0, 1);
    lcd.print("***aktiivinen***");
  }
```

```
  else
  {
    switch (caseLuku)                                   //Toiminto (case) määräytyy caseLuku-muuttujan arvon mukaan.
    {                                                   //caseLuku-arvoa vaihdetaan erillisen napin painalluksen mukaan.
      case 1:
        lcd.clear();                                    //Tyhjätään näyttö
        lcd.setCursor(0, 0);                            //Asetetaan LCD20-näytön kursori ensimmäiselle riville ja ensimmäiseen sarakkeeseen.
        lcd.print("****Eteinen****");                   //Kirjoitetaan näytölle teksti.
        lcd.setCursor(0, 1);                            //Asetetaan LCD20-näytön kursori toiselle riville ja ensimmäiseen sarakkeeseen.
        lcd.print(lampotila1);                          //Kirjoitetaan näytölle lampotila1-muuttujan arvo (Eteisen lämpötila).
        lcd.setCursor(4, 1);                            //Asetetaan LCD20-näytön kursori toiselle riville ja neljänteen sarakkeeseen.
        lcd.print(" *C");                               //Kirjoitetaan näytölle teksti.
        break;                                          //Menemme pois switch-rakenteesta.

      case 2:                                           //Kts. case1
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("****Olohuone*****");
        lcd.setCursor(0, 1);
        lcd.print(lampotila2);
        lcd.setCursor(4, 1);
        lcd.print(" *C");
        break;

      case 3:                                           //Kts. case1
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("****Tuuletin****");
        lcd.setCursor(0, 1);
        lcd.print(prosentti);
        lcd.setCursor(4, 1);
        lcd.print(" %");
        break;

      case 4:                                           //Kts. case1
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("***Asetusarvo***");
        lcd.setCursor(0, 1);
        lcd.print(M20asetusarvoApu);
        lcd.setCursor(4, 1);
        lcd.print(" *C");
        break;
    }
  }


void Valikkonappi()
{

  btnTila = digitalRead(btn);                       //Luetaan btn-muuttujan tila (valikkorakenteen nappi) ja kirjoitetaan sen tila btnTila-muuttujaan.

  if (btnTila != btnEdellinenTila)                  //Jos bntTila on erisuuri kuin btnEdellinenTila menemme if-rakenteeseen.
  {
    if (btnTila == HIGH)                            //Jos bntTila on HIGH (painettu valikkorakenteen nappi pohjaan)
    {
      caseLuku++;                                   //Lisätään caseLuku-muuttujaan +1.
      if (caseLuku == 5)                            //Jos caseLuku-muuttuja = 5.
      {
        caseLuku = 1;                               //Kirjoitetaan caseLuku-muuttujaan arvo 1.
      }
    }
  }
}


void Thinger()
{
  lahtevaTieto[0] = lampotila1*5;                   //Kirjoitetaan lahtevaTieto-taulukon 0-paikkaan lampotila1-arvo *5 (Eteisen lämpötila)
  lahtevaTieto[1] = lampotila2*5;                   //Kirjoitetaan lahtevaTieto-taulukon 1-paikkaan lampotila2-arvo *5 (Olohuoneen lämpötila)
  lahtevaTieto[2] = prosentti;                      //Kirjoitetaan lahtevaTieto-taulukon 2-paikkaan prosentti-arvo (Moottorin M20 pyörimisnopeus 0...100%)
  lahtevaTieto[3] = M20asetusarvoApu*5;             //Kirjoitetaan lahtevaTieto-taulukon 3-paikkaan M20asetusarvoApu-arvo *5 (Lämpötilan asetusarvo)
  Serial.write(lahtevaTieto[0]);                    //Lähetetään edellä kirjoitetut tiedot Serial(9600) kautta NODEMCU-laitteelle.
  Serial.write(lahtevaTieto[1]);
  Serial.write(lahtevaTieto[2]);
  Serial.write(lahtevaTieto[3]);

/*
 * lahtevaTieto-taulukko on char-tyyppinen ja kun integer-tyyppinen tieto kirjoitetaan char-tyyppiseen muuttujaan,
 * se kirjoittaa vain kokonaisluvun ja leikkaa loput pois. Tämän vuoksi liukuluvut kerrotaan arvolla 5
 * ja serialin vastaanottopäässä luku jaetaan arvolla 5. Näin saadaan liukuluvut näytettyä käyttäjälle 0,2 yksikön tarkkuudella.
 */

}
```

Liite 4. Näppäinpaneelin ohjelmakoodi

```
/*
   OPINNÄYTETYÖ 2017
   KOTIAUTOMAATIOJÄRJESTELMÄ

   NÄPPÄINPANEELI
   Thomas Kuronen
   10/2017

*/

#include <Keypad.h>                       //Näppäinpaneelin käyttämä kirjasto.

const byte ROWS = 4;                      //Määritellään rivien määrä.
const byte COLS = 3;                      //Määritellään sarakkeiden määrä.


char hexaKeys[ROWS][COLS] = {             //Määritellään jokaiselle näppäimelle oma merkki.
  {'1', '2', '3'},
  {'4', '5', '6'},
  {'7', '8', '9'},
  {'*', '0', '#'}
};

byte rowPins[ROWS] = {6, 7, 8, 9};        //Määritellään rivipinnien kytkennät arduinolle.
byte colPins[COLS] = {10, 11, 12};        //Määritellään sarakepinnien kytkennät arduinolle.

Keypad customKeypad = Keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);      //Tehdään näppäinpaneeli asetusten mukaiseksi.

String koodi = "A";                       //Tehdään string-tyyppinen muuttuja ja alustetaan se merkkijonolla 'A'.
String koodioikea = "A11235#";            //Tehdään sting-tyyppinen muuttuja ja alustetaan se merkkijonolla 'A11235#'

boolean kotona = true;                    //Tehdään boolean-tyyppinen muuttuja ja alustetaan se olemaan TRUE.

int ind = 13;                             //Määritellään ind-muuttujan pinni.


void setup() {

Serial.begin(9600);                      //Asetetaan serialin datansiirtonopeus bittejä sekunissa (baud).
pinMode(ind, OUTPUT);                    //Määritellään pinni OUTPUTIKSI.
}



void loop() {

  char customKey = customKeypad.getKey();      //Luetaan painetun näppäimen merkki ja asetetaan se customKey-muuttujaan.

  if (customKey) {                             //Jos custonKey-muuttujassa on jokin arvo menemme if-rakenteeseen-
    koodi = String(koodi + customKey);         //Kirjoitetaan koodi-muuttujaan koodi-muuttujan sisältö ja lisätään siihen
                                               //customKey-muuttujan sisältö.
    if (customKey == '*')                      //Jos customKey = *, menemme if-rakenteeseen.
    {
      koodi = 'A';                             //Alustetaan koodi-muuttuja merkkijonolla 'A'.
    }

    if (customKey == '#')                      //Jos customKey = #, menemme if-rakenteeseen.
    {
      if (koodi == koodioikea)                 //Jos koodi-merkkijono on sama kuin koodioikea-merkkijono, menemme if-rakenteeseen.
      {
        koodi  = 'A';                          //Alustetaan koodi-muuttuja merkkijonolla 'A'.

        if (kotona)                            //Jos kotona-muuttuja on TRUE, menemme if-rakenteeseen.
        {
          digitalWrite(ind, HIGH);             //Asetetaan ind-lähtö HIGH-tilaan.
          kotona = false;                      //Asetetaan kotona-muuttujaan FALSE.
        }
        else                                   //Jos kotona-muuttuja ei ole TRUE, menemme if-rakenteeseen.
        {
          kotona = true;                       //Asetetaan kotona-muuttujaan TRUE.
          digitalWrite(ind, LOW);              //Asetetaan ind-lähtö LOW-tilaan.
        }

      }
      else                                     //Jos koodi-merkkijono ei ole sama kuin koodioikea-merkkijono,
      {
        koodi = 'A';                           //Alustetaan koodi-muuttuja merkkijonolla 'A'.
      }
    }
  }
}
```

Liite 5. Kaukosäätimen ohjelmakoodi

```
/*
   OPINNÄYTETYÖ 2017
   KOTIAUTOMAATIOJÄRJESTELMÄ

   KAUKOSÄÄDIN
   Thomas Kuronen
   10/2017

*/
//Kirjastot jotka NRF-lähetin tarvitsee.
#include "LowPower.h"
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

RF24 radio(7, 8);                                                   //Määritetään pinnit mihin radion CNS ja CE on kytketty.
int msg[1];                                                         //Määritetään msg-taulukon koko.
const uint64_t pipes[3] = {0, 0xF0F0F0F0E1LL, 0xF0F0F0F0E2LL};      //Määritetään pipes-taulukon kolme osoitetta.
const uint64_t pipe1 = pipes[1];                                    //Kirjoitetaan pipe1-muuttujaan osoite pipes-taulukon paikasta 1.
const uint64_t pipe2 = pipes[2];                                    //Kirjoitetaan pipe2-muuttujaan osoite pipes-taulukon paikasta 2.
int btnYlos = 2;                                                    //Määritellään painonapin (PN100) pinni.
int btnAlas = 3;                                                    //Määritellään painonapin (PN101) pinni.
int kytkinAsento1 = 4;                                              //Määritellään kytkimen (SW100) 1-asennon pinni.
int kytkinAsento2 = 5;                                              //Määritellään kytkimen (SW100) 2-asennon pinni.
boolean ylos;                                                       //Määritellään ylos-muuttuja booleaniksi.
boolean alas;                                                       //Määritellään alas-muuttuja booleaniksi.
int asento;                                                         //Määritellään asento-muuttuja integeriksi.

void setup() {

  Serial.begin(9600);                                               //Asetetaan serialin datansiirtonopeus bittejä sekunissa (baud).

  pinMode(btnYlos, INPUT);                                          //Määritellään pinnit INPUTIKSI tai OUTPUTIKSI.
  pinMode(btnAlas, INPUT);
  pinMode(kytkinAsento1, INPUT);
  pinMode(kytkinAsento2, INPUT);

  radio.begin();                                                    //Käynnistetään radio.
  radio.setPALevel(RF24_PA_LOW);                                    //Asetetaan radion tehoksi LOW.
}



void loop() {
  /*******************NAPPIEN ASENTOJEN TUNNISTUS*********************/
  if (digitalRead(kytkinAsento1) == HIGH)                           //Tarkastetaan valintakytkimen (SW100) asento.
  {
    asento = 1;                                                     //Asennossa 1 kirjoitetaan muuttujaan asento luku 1.
  }
  else if (digitalRead(kytkinAsento2) == HIGH)                      //Tarkastetaan valintakytkimen (SW100) asento.
  {
    asento = 2;                                                     //Asennossa 2 kirjoitetaan muuttujaan asento luku 2.
  }
  else                                                              //Kytkimen ollessa asennossa 0,
  {
    asento = 0;                                                     //Kirjoitetaan muuttujaan asento luku 0.
  }

  if (digitalRead(btnYlos) == HIGH)                                 //Luetaan painonapin (PN100) tilatieto. Jos nappia
  {                                                                 //on painettu, niin menemme if-rakenteeseen.
    ylos = 1;                                                       //Kirjoitetaan ylos-muuttujaan arvo 1.
  }
  else                                                              //Jos painonappia(PN100) ei paineta,
  {
    ylos = 0;                                                       //kirjoitetaan ylos-muuttujaan arvo 0.
  }

  if (digitalRead(btnAlas) == HIGH)                                 //Luetaan painonapin (PN101) tilatieto. Jos nappia
  {                                                                 //on painettu, niin menemme if-rakenteeseen.
    alas = 1;                                                       //Kirjoitetaan alas-muuttujaan arvo 1.
  }
  else                                                              //Jos painonappia(PN101) ei paineta,
  {
    alas = 0;                                                       //kirjoitetaan alas-muuttujaan arvo 0.
  }

  /*******************PUTKEN MÄÄRITYS*********************/
  if (asento == 1)                                                //Jos asento-muuttuja = 1,
  {
    radio.openWritingPipe(pipe1);                                 //avataan keskusteluputki pipe1.
  }
  else if (asento == 2)                                           //Jos asento-muuttuja = 2,
  {
    radio.openWritingPipe(pipe2);                                 //avataan keskusteluputki pipe2.
  }

  /*
   * Tällä if-rakenteella voidaan valita SW100 kytkimellä kohde mitä halutaan ohjata.
   * Autotallin ja portin ohjaukseen on eri keskusteluputket joita ohjataan erikseen.
   */
```

```
/******************YLÖS/ALAS*********************/
if (ylos == 1)                                  //Jos ylos-muuttuja = 1.
{

  msg[0] = 1;                                   //Kirjoitetaan msg-taulukon 0-paikkaan luku 1.
  radio.write(&msg, sizeof(msg));               //Lähetetään msg-taulukko vastaanottimelle.
  msg[0] = 0;                                   //Muutetaan msg-taulukon 0-paikkaan luku 0.

}


if (alas == 1)                                  //Jos alas muuttuja = 1.
{

  msg[0] = 2;                                   //Kirjoitetaan msg-taulukon 0-paikkaan luku 2.
  radio.write(&msg, sizeof(msg));               //Lähetetään msg-taulukko vastaanottimelle.
  msg[0] = 0;                                   //Muutetaan msg-taulukon 0-paikkaan luku 0.

}

/*
 * Vastaanottimelle lähetetään luku 1 tai luku 2.
 * Tämä kertoo pitääkö moottoria pyörittää myötäpäivään vai vastapäivään.
 * Kun ylös tai alas painonappia painetaan, lähetin lähettää lukua 1 tai lukua 2 useita kappaleita
 * nopealla syklillä.
 */

}
```

Liite 6. Portin ohjelmakoodi

```
/*
   OPINNÄYTETYÖ 2017
   KOTIAUTOMAATIOJÄRJESTELMÄ

   PORTTI
   Thomas Kuronen
   10/2017

*/

//Kirjastot jotka NRF-vastaanotin tarvitsee.
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

RF24 radio(7, 8);                                   //Määritetään pinnit mihin radion CNS ja CE on kytketty.

int msg[1];                                         //Määritetään msg-taulukon koko.

const uint64_t pipe = 0xF0F0F0F0E1LL;               //Määritetään pipe-muuttujan osoite.

int hairio;                                         //Määritetään hairio-muuttuja integeriksi.

int Pin1 = 2;                                       //Määritellään askelmoottorin johtimen pinni.
int Pin2 = 3;                                       //Määritellään askelmoottorin johtimen pinni.
int Pin3 = 4;                                       //Määritellään askelmoottorin johtimen pinni.
int Pin4 = 5;                                       //Määritellään askelmoottorin johtimen pinni.
int _step = 0;                                      //Määritellään _step-muuttuja integeriksi ja alustetaan se luvulla 0.


boolean suunta = true;                              //Määritellään suunta-muuttuja booleaniksi ja alustetaan se TRUEKSI.
boolean edellinenSuunta = false;                    //Määritellään edellinenSuunta-muuttuja booleaniksi ja
                                                    //alustetaan se FALSEKSI.
int luku = 15000;                                   //Määritellään luku-muuttuja integeriksi ja alustetaan se arvolla 15000.
int askeleet = 15000;                               //Määritellään askeleet-muuttuja integeriksi ja alustetaan se arvolla 15000.
int laatta = 9;                                     //Määritellään painolaatan pinni.

void setup() {

  Serial.begin(9600);                               //Asetetaan serialin datansiirtonopeus bittejä sekunissa (baud).
  pinMode(Pin1, OUTPUT);                            //Määritellään pinnit INPUTIKSI tai OUTPUTIKSI.
  pinMode(Pin2, OUTPUT);
  pinMode(Pin3, OUTPUT);
  pinMode(Pin4, OUTPUT);
  pinMode(laatta, INPUT);

  radio.begin();                                    //Käynnistetään radio.
  radio.setPALevel(RF24_PA_LOW);                    //Asetetaan radion tehoksi LOW.
  radio.openReadingPipe(1, pipe);                   //Avataan keskusteluputki jonka osoite on muuttujan pipe arvo.
  radio.startListening();                           //Aloitetaan putken kuuntelu.


void loop() {

  /***********RADIO**************/

 if (digitalRead(laatta == LOW))                    //Jos laatta-muuttuja on LOW-tilassa. (Portin edessä ei ole estettä)
 {
  while (radio.available())                         //Suoritetaan while-looppia niin kauan kun keskusteluputkesta
  {                                                 //on tulossa tietoa.
    radio.read(&msg, sizeof(msg));                  //Luetaan putkesta tuleva tieto ja tallennetaan se msg-taulukkoon.
  }
 }

  /***********SUUNNAN VAIHTO**************/

  if (msg[0] == 1 and edellinenSuunta == false and luku == askeleet) //PORTTI MENEE KIINNI
  {                                                 //Jos msg-taulukon 0-paikan arvo on 1 ja edellinensuunta = false ja luku = 15000
    suunta = true;                                  //Kirjoitetaan suunta-muuttujaksi TRUE
    luku = 0;                                       //Alustetaan luku-muuttuja nollaksi
    msg[0] = 0;                                     //Alustetaan msg-taulukon 0-paikka nollaksi
    edellinenSuunta = true;                         //Kirjoitetaan edellinenSuunta-muuttujaksi TRUE
  }

/*
 * EdellinenSuunta-muuttujalla voidaan varmistaa, että portti on auki. Näin estetään portin liikkuminen kiinni jos se on jo kiinni.
 * Lisäksi luku-muuttuja sisältää askeleet mitä portti liikkuessaan ottaa. Jos askeleet on erisuuri kuin 15000, niin portti on silloin vielä matkalla.
 * Kun portti on matkalla, niin se ei ota liikkumiskäskyjä vastaan. 15000 askelta on määrä minkä moottori ottaa portin mennessä kiinni tai auki.
 * Suunta-muuttujaa käytetään myöhemmin koodissa kertomaan portin suunta.
 */

  else if (msg[0] == 2 and edellinenSuunta == true and luku == askeleet) //PORTTI AVAUTUU
  {                                                 //Jos msg-taulukon 0-paikan arvo on 2 ja edellinensuunta = true ja luku = 15000
    suunta = false;                                 //Kirjoitetaan suunta-muuttujaksi false
    luku = 0;                                       //Alustetaan luku-muuttuja nollaksi
    msg[0] = 0;                                     //Alustetaan msg-taulukon 0-paikka nollaksi
    edellinenSuunta = false;                        //Kirjoitetaan edellinenSuunta-muuttujaksi FALSE
  }

/*
 * EdellinenSuunta-muuttujalla voidaan varmistaa, että portti on kiinni. Näin estetään portin liikkuminen auki jos se on jo auki.
 * Lisäksi luku-muuttuja sisältää askeleet mitä portti liikkuessaan ottaa. Jos askeleet on erisuuri kuin 15000, niin portti on silloin vielä matkalla.
 * Kun portti on matkalla, niin se ei ota liikkumiskäskyjä vastaan. 15000 askelta on määrä minkä moottori ottaa portin mennessä kiinni tai auki.
 * Suunta-muuttujaa käytetään myöhemmin koodissa kertomaan portin suunta.
 */

  stepit();                                         //Suoritetaan aliohjelma stepit(). Tämä aliohjelma liikuttaa moottoria yhden askeleen kerralla.
```

```
  /**********ASKELEET JA EHDOT**************/

  if (luku < askeleet and suunta == false and hairio == 0)    //Jos luku < asekeleet ja suunta = false ja hairio = 0, menemme if-rakenteeseen.
  {
    _step--;                      //Vähennetään _step-muuttujasta luvun yksi.
    if (_step < 0)                //Jos _step-muuttuja on pienempi kuin luku 0,
    {
      _step = 7;                  //kirjoitetaan _step-muuttujan arvoksi 7.
    }
    delay(1);                     //Odotetaan 1 millisekunti.
    luku++;                       //Lisätään luku-muuttujaan luku 1.
  }
/*
 * Luku-muuttujan arvo pitää olla pienempi kuin askeleet. Tämä estää moottorin pyörimisen jos se on jo saavuttanut määränpään.
 * Suunta-muuttuja pitää olla false, eli portti on saanut avautumiskäskyn. Lisäksi ei saa olla häiriötä. Häiriö tulee, kun portin edessä on este.
 * Jos nämä toteutuvat, if-rakennetta suoritetaan niin kauan, kunnes portti on täysin auki. Toisin sanoen luku-muuttuja on saavuttanut arvon 15000.
 */

  else if (luku < askeleet and suunta == true and digitalRead(laatta) == LOW and hairio == 0)
  {                               //Jos luku < askeleet ja suunta = true ja laatta = low ja hairio = 0.
    _step++;                      //Lisätään _step-muuttujaan luku yksi.
    if (_step > 7)                //Jos _step-muuttuja > 7,
    {
      _step = 0;                  //kirjoitetaan _step-muuttujan arvoksi 0.
    }
    delay(1);                     //Odotetaan 1 millisekunti.
    luku++;                       //Lisätään luku-muuttujaan luku 1.
  }
  /*
   * Portti on saanut kiinni-käskyn, koska suunta-muuttuja on true. Portti menee kiinni jos luku < askeleet, suunta = true,
   * painolaatta ei ole tunnistanut estettä sekä järjestelmässä ei ole häiriötä. Häiriö tulee, jos painolaatta tunnistaa esteen.
   * Jos nämä toteutuvat, else if-rakennetta suoritetaan niin kauan, kunnes portti on täysin kiinni. Eli luku-muuttuja on saavuttanut arvon 15000.
   */

  else if (digitalRead(laatta) == HIGH and suunta == true and hairio == 0 and luku != askeleet)
  {                               //Jos painolaatta = HIGH, suunta = true, hairio = 0 ja luku on erisuuri kuin askeleet.
    hairio = 1;                   //Kirjoitetaan hairio-muuttujaan arvo yksi.
  }
  /*
   * Else if-rakenne toteutuu, kun painolaatta tunnistaa esteen sekä portti on menossa kiinni.
   * Luku-muuttujan pitää olla erisuuri kuin askeleet, mikä tarkoittaa että portti on liikkeessä.
   * Häiriötä ei tehdä turhaan, eli kun portti on aukeamassa ja painolaatta tunnistaa esteen,
   * tämä ei saa vaikuttaa toimintaan. Portti ei voi auetessa vahingoittaa estettä.
   */


  else if (hairio == 1)           //Jos hairio = 1. Eli portti on ollut menossa kiinni. Tällöin luku-muuttujassa askeleet voisivat olla vaikkapa 8000.
  {
    _step--;                      //Vähennetään _step-muuttujasta luku 1.
    if (_step < 0)                //Jos _step-muuttujan arvo on pienempi kuin 0.
    {
      _step = 7;                  //Kirjoitetaan _step-muuttujan arvoksi 7.
    }
    delay(1);                     //Odotetaan 1 millisekunti.
    luku--;                       //Vähennetään luku-muuttujasta arvo 1.

    if (luku <= 0)                //Jos luku-muuttuja on 0 tai pienempi.
    {
      luku = askeleet;            //Kirjoitetaan luku-muuttujaan askeleet arvo (15000)
      edellinenSuunta = false;    //Kirjoitetaan edellinenSuunta-muuttujaan false.
      msg[0] = 0;                 //Kirjoitetaan msg-taulukon kohtaan 0 arvo 0.
      hairio = 0;                 //Kirjoitetaan hairio-muuttujaan arvo 0.
    }
  }
  /*
   * Jos järjestelmässä on tapahtunut häiriö, eli painolaatta on tunnistanut esteen portin mennessä kiinni,
   * porttia avataan yhtä monta askelta mitä se oli kerennyt ottamaan mennessä kiinni.
   * Tämä palauttaa portin täysin samaan kohtaan mistä se oli lähtenytkin, eli täysin auki.
   * Vasta sitten kun portti on täysin auki, se poistaa häiriön ja jatkaa normaalia toimintaa.
   * Häiriön aikana myöskään kaukosäätimen käskyillä ei ole merkitystä.
   */

  else                            //Jos mikään edellä olevista if-rakenteista tai else if-rakenteista ei toteudu,
  {
    digitalWrite(Pin1, LOW);      //Kirjoitetaan moottoria ohjaavien pinnien lähtojen arvoiksi LOW.
    digitalWrite(Pin2, LOW);
    digitalWrite(Pin3, LOW);
    digitalWrite(Pin4, LOW);
  }

}
```

```
void stepit()                   //Stepit-aliohjelma pyöräyttää moottoria myötä- tai vastapäivään yhden askeleen kerralla.
{                               //_step-muuttujan arvon perusteella se tietää missä kohtaa se on edellisen kerran ollut menossa.
                                //Aliohjelma suoritetaan jokaisella ohjelmakierrolla, mutta jos _step-muuttuja pysyy samana,
                                //moottori ei liiku.
  switch (_step)
  {
    case 0:
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, LOW);
      digitalWrite(Pin4, HIGH);
      break;
    case 1:
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, HIGH);
      digitalWrite(Pin4, HIGH);
      break;
    case 2:
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, HIGH);
      digitalWrite(Pin4, LOW);
      break;
    case 3:
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, HIGH);
      digitalWrite(Pin3, HIGH);
      digitalWrite(Pin4, LOW);
      break;
    case 4:
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, HIGH);
      digitalWrite(Pin3, LOW);
      digitalWrite(Pin4, LOW);
      break;
    case 5:
      digitalWrite(Pin1, HIGH);
      digitalWrite(Pin2, HIGH);
      digitalWrite(Pin3, LOW);
      digitalWrite(Pin4, LOW);
      break;


    case 6:
      digitalWrite(Pin1, HIGH);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, LOW);
      digitalWrite(Pin4, LOW);
      break;
    case 7:
      digitalWrite(Pin1, HIGH);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, LOW);
      digitalWrite(Pin4, HIGH);
      break;
    default:
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, LOW);
      digitalWrite(Pin4, LOW);
      break;
  }
}
```

Liite 7. Autotallin ohjelmakoodi

```
/*
   OPINNÄYTETYÖ 2017
   KOTIAUTOMAATIOJÄRJESTELMÄ

   AUTOTALLI
   Thomas Kuronen
   10/2017
*/

//Kirjastot jotka NRF-vastaanotin tarvitsee.
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

RF24 radio(7, 8);                                   //Määritetään pinnit mihin radion CNS ja CE on kytketty.
int msg[1];                                         //Määritetään msg-taulukon koko.
const uint64_t pipe = 0xF0F0F0F0E2LL;               //Määritetään pipe-muuttujan osoite.
int Pin1 = 2;                                       //Määritellään askelmoottorin johtimen pinni.
int Pin2 = 3;                                       //Määritellään askelmoottorin johtimen pinni.
int Pin3 = 4;                                       //Määritellään askelmoottorin johtimen pinni.
int Pin4 = 5;                                       //Määritellään askelmoottorin johtimen pinni.
int _step = 0;                                      //Määritellään _step-muuttuja integeriksi ja alustetaan se luvulla 0.

boolean suunta = true;                              //Määritellään suunta-muuttuja booleaniksi ja alustetaan se TRUEKSI.
boolean edellinenSuunta = false;                    //Määritellään edellinenSuunta-muuttuja booleaniksi ja
                                                    //alustetaan se FALSEKSI.
int luku = 6000;                                    //Määritellään luku-muuttuja integeriksi ja alustetaan se arvolla 6000.
int askeleet = 6000;                                //Määritellään askeleet-muuttuja integeriksi ja alustetaan se arvolla 6000.
                                                    //(Oven avautuessa tai sulkeutuessa askelmoottori ottaa 6000 askelta.)
int LED60 = 6;                                      //Määritellään valaistuksen pinni.
int SW28 = 9;                                       //Määritellään kytkimen pinni.

void setup() {

  Serial.begin(9600);                               //Asetetaan serialin datansiirtonopeus bittejä sekunissa (baud).
  pinMode(Pin1, OUTPUT);                            //Määritellään pinnit INPUTIKSI tai OUTPUTIKSI.
  pinMode(Pin2, OUTPUT);
  pinMode(Pin3, OUTPUT);
  pinMode(Pin4, OUTPUT);
  pinMode(SW28, INPUT);
  pinMode(LED60, OUTPUT);

  radio.begin();                                    //Käynnistetään radio.
  radio.setPALevel(RF24_PA_LOW);                    //Asetetaan radion tehoksi LOW.
  radio.openReadingPipe(1, pipe);                   //Avataan keskusteluputki jonka osoite on muuttujan pipe arvo.
  radio.startListening();                           //Aloitetaan putken kuuntelu.
}



void loop() {
  /***********RADIO*************/
  while (radio.available())                         //Suoritetaan while-looppia niin kauan, kun keskusteluputkesta
  {                                                 //on tulossa tietoa.
    radio.read(&msg, sizeof(msg));                  //Luetaan putkesta tuleva tieto ja tallennetaan se msg-taulukkoon.
  }
  /***********SUUNNAN VAIHTO*************/
  if (msg[0] == 1 and edellinenSuunta == false and luku == askeleet) //OVI MENEE KIINNI
  {                                                 //Jos msg-taulukon 0-paikan arvo on 1 ja edellinensuunta = false ja luku = 6000
    suunta = true;                                  //Kirjoitetaan suunta-muuttujaksi TRUE
    luku = 0;                                       //Alustetaan luku-muuttuja nollaksi
    msg[0] = 0;                                     //Alustetaan msg-taulukon 0-paikka nollaksi
    edellinenSuunta = true;                         //Kirjoitetaan edellinenSuunta-muuttujaksi TRUE
  }
  else if (msg[0] == 2 and edellinenSuunta == true and luku == askeleet) //OVI AVAUTUU
  {                                                 //Jos msg-taulukon 0-paikan arvo on 2 ja edellinensuunta = true ja luku = 6000
    suunta = false;                                 //Kirjoitetaan suunta-muuttujaksi false
    luku = 0;                                       //Alustetaan luku-muuttuja nollaksi
    msg[0] = 0;                                     //Alustetaan msg-taulukon 0-paikka nollaksi
    edellinenSuunta = false;                        //Kirjoitetaan edellinenSuunta-muuttujaksi FALSE
  }
  stepit();                                         //Suoritetaan aliohjelma stepit(). Tämä aliohjelma liikuttaa moottoria yhden askeleen kerralla.
  /***********ASKELEET JA EHDOT*************/
  if (luku < askeleet and suunta == false)          //Jos luku < asekeleet ja suunta = false, menemme if-rakenteeseen.
  {
    _step--;                                //Vähennetään _step-muuttujasta luku yksi.
    if (_step < 0)                          //Jos _step-muuttuja on pienempi kuin luku 0,
    {
      _step = 7;                            //kirjoitetaan _step-muuttujan arvoksi 7.
    }
    delay(1);                               //Odotetaan 1 millisekunti.
    luku++;                                 //Lisätään luku-muuttujaan luku 1.
  }
  else if (luku < askeleet and suunta == true)      //Jos luku < askeleet ja suunta = true, menemme if-rakenteeseen.
  {
    _step++;                                //Lisätään _step-muuttujaan luku yksi.
    if (_step > 7)                          //Jos _step-muuttuja > 7,
    {
      _step = 0;                            //kirjoitetaan _step-muuttujan arvoksi 0.
    }
    delay(1);                               //Odotetaan 1 millisekunti.
    luku++;                                 //Lisätään luku-muuttujaan luku 1.
  }
  else                                      //Jos mikään edellä olevista if-rakenteista tai else if-rakenteista ei toteudu,
  {
    digitalWrite(Pin1, LOW);                //Kirjoitetaan moottoria ohjaavien pinnien lähtojen arvoiksi LOW.
    digitalWrite(Pin2, LOW);
    digitalWrite(Pin3, LOW);
    digitalWrite(Pin4, LOW);
  }
```

```
  if (digitalRead(SW28) == HIGH)           //Jos kytkin SW28 asento on HIGH-tilassa,
  {
    digitalWrite(LED60, HIGH);             //Asetetaan lähtö LED60 HIGH-tilaan.
  }
  else
  {
    digitalWrite(LED60, LOW);              //Muutoin asetetaan lähtö LED60 LOW-tilaan.
  }

}

void stepit()                              //Stepit-aliohjelma pyöräyttää moottoria myötä- tai vastapäivään yhden askeleen kerralla.
{                                          //_step-muuttujan arvon perusteella se tietää missä kohtaa se on edellisen kerran ollut menossa.
                                           //Aliohjelma suoritetaan jokaisella ohjelmakierrolla, mutta jos _step-muuttuja pysyy samana,
                                           //moottori ei liiku.
  switch (_step)
  {
    case 0:
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, LOW);
      digitalWrite(Pin4, HIGH);
      break;
    case 1:
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, HIGH);
      digitalWrite(Pin4, HIGH);
      break;
    case 2:
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, LOW);;
      digitalWrite(Pin3, HIGH);
      digitalWrite(Pin4, LOW);
      break;
    case 3:
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, HIGH);
      digitalWrite(Pin3, HIGH);
      digitalWrite(Pin4, LOW);
      break;


    case 4:
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, HIGH);
      digitalWrite(Pin3, LOW);
      digitalWrite(Pin4, LOW);
      break;
    case 5:
      digitalWrite(Pin1, HIGH);
      digitalWrite(Pin2, HIGH);
      digitalWrite(Pin3, LOW);
      digitalWrite(Pin4, LOW);
      break;
    case 6:
      digitalWrite(Pin1, HIGH);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, LOW);
      digitalWrite(Pin4, LOW);
      break;
    case 7:
      digitalWrite(Pin1, HIGH);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, LOW);
      digitalWrite(Pin4, HIGH);
      break;
    default:
      digitalWrite(Pin1, LOW);
      digitalWrite(Pin2, LOW);
      digitalWrite(Pin3, LOW);
      digitalWrite(Pin4, LOW);
      break;

  }
}
```

Liite 8. Sähköpostipalvelun ohjelmakoodi

```
/*
   OPINNÄYTETYÖ 2017
   KOTIAUTOMAATIOJÄRJESTELMÄ

   SÄHKÖPOSTI
   Thomas Kuronen
   10/2017

*/

#include <ESP8266WiFi.h>            //Kirjasto minkä ESP8266 laite tarvitsee
WiFiClient client;                  //Alustaa client kirjaston

//Koodi mitä on saatu easycoding-palvelusta.
String MakerIFTTT_Key ;
;String MakerIFTTT_Event;
char *append_str(char *here, String s) {int i = 0; while (*here++ = s[i]) { i++; }; return here - 1;}
char *append_ul(char *here, unsigned long u) {char buf[20]; return append_str(here, ultoa(u, buf, 10));}
char post_rqst[256]; char *p; char *content_length_here; char *json_start; int compi;

//Sähköpostin lähetysmerkki

int VarasInd = 14;                      //Määritellään pinni mihin saadaan talon Arduino Megalta tieto varkaasta.
int LahetaVarasSposti = 0;              //Määritellään LahetaVarasSposti-muuttuja ja annetaan sille arvo 0.
int edellinenLahetaVarasSposti = 0;     //Määritellään edellinenLahetaVarasSposti-muuttuja ja annetaan sille arvo 0.

int PaloInd = 5;                        //Määritellään pinni mihin saadaan talon Arduino Megalta tieto tulipalosta.
int LahetaPaloSposti = 0;               //Määritellään LahetaPaloSposti-muuttuja ja annetaan sille arvo 0.
int edellinenLahetaPaloSposti = 0;      //Määritellään edellinenLahetaPaloSposti-muuttuja ja annetaan sille arvo 0.



void setup()
{
  Serial.begin(9600);                   //Asetetaan serialin datansiirtonopeus bittejä sekunissa (baud).
  WiFi.disconnect();                    //Poistutaan nykyisestä verkosta jos sellaisessa ollaan.
  delay(3000);                          //Odotetaan 3 sekuntia.
  Serial.println("Start");              //Kirjoitetaan Serial-monitoriin teksti.
  WiFi.begin("[redacted]", "[redacted]");       //Määritellään verkon SSID ja salasana
  while ((!(WiFi.status() == WL_CONNECTED))) {          //Pyöritetään tyhjää while-looppia niin kauan,
    delay(300);                                         //kunnes yhteys verkkoon on muodostunut.
  }

  pinMode(VarasInd, INPUT);             //Määritellään pinnit INPUTEIKSI.
  pinMode(PaloInd, INPUT);
}
```

```
void loop()
{

 LahetaVarasSposti = digitalRead(VarasInd);       //Luetaan talon Arduino Megalta tulevaa varastietoa ja kirjataan
                                                  //sen arvo muuttujaan.
  if (LahetaVarasSposti != edellinenLahetaVarasSposti)    //Jos LahetaVarasSposti on erisuuri kuin edellinenLahetaVarasSposti
  {                                               //if-rakenteeseen mennään vain kerran yhden hälytyksen aikana.
    if (LahetaVarasSposti == 1)                   //Jos LahetaVarasSposti = 1. (Varas on huomattu)
    {
      Serial.println("Connected");                //If-rakenteen sisällä oleva koodi saatu easycoding-palvelusta.
      if (client.connect("maker.ifttt.com", 80)) {        //Yhdistetään palveluun
        MakerIFTTT_Key = "[redacted]";
        MakerIFTTT_Event = "mail";
        p = post_rqst;
        p = append_str(p, "POST /trigger/");
        p = append_str(p, MakerIFTTT_Event);
        p = append_str(p, "/with/key/");
        p = append_str(p, MakerIFTTT_Key);
        p = append_str(p, " HTTP/1.1\r\n");
        p = append_str(p, "Host: maker.ifttt.com\r\n");
        p = append_str(p, "Content-Type: application/json\r\n");
        p = append_str(p, "Content-Length: ");
        content_length_here = p;
        p = append_str(p, "NN\r\n");
        p = append_str(p, "\r\n");
        json_start = p;
        p = append_str(p, "{\"value1\":\"");
        p = append_str(p, "Sahköposti kotiautomaatiolta");
        p = append_str(p, "\",\"value2\":\"");
        p = append_str(p, "Varkaita");
        p = append_str(p, "\",\"value3\":\"");
        p = append_str(p, " talossa!!!");
        p = append_str(p, "\"}");

        compi = strlen(json_start);
        content_length_here[0] = '0' + (compi / 10);
        content_length_here[1] = '0' + (compi % 10);
        client.print(post_rqst);

      }
    }
    delay(50);                                    //Odotetaan 50 millisekuntia.
    edellinenLahetaVarasSposti = LahetaVarasSposti;       //Kirjataan edellinenLahetaVarasSposti-muuttujaan
  }                                               //LahetaVarasSposti-muuttujan arvo.

/*
 * Kun talon Arduino Mega ilmoittaa varkaasta, niin sähköpostiviesti lähetetään kerran.
 * edellinenLahetaVarasSposti-muuttujan avulla viestiä ei lähetetä turhaan.
 */



/*
 * Palohälytyksestä aiheutuva hälytys ja sähköpostin lähetys toimii
 * samalla tavalla kuin varkaasta aiheutuva hälytys. Vain muuttujien nimet
 * ja viesti ovat erilaisia.
 */

   LahetaPaloSposti = digitalRead(PaloInd);

     if (LahetaPaloSposti != edellinenLahetaPaloSposti)
  {

    if (LahetaPaloSposti == 1)
    {
      Serial.println("Connected");
      if (client.connect("maker.ifttt.com", 80)) {
        MakerIFTTT_Key = "[redacted]";
        MakerIFTTT_Event = "mail";
        p = post_rqst;
        p = append_str(p, "POST /trigger/");
        p = append_str(p, MakerIFTTT_Event);
        p = append_str(p, "/with/key/");
        p = append_str(p, MakerIFTTT_Key);
        p = append_str(p, " HTTP/1.1\r\n");
        p = append_str(p, "Host: maker.ifttt.com\r\n");
        p = append_str(p, "Content-Type: application/json\r\n");
        p = append_str(p, "Content-Length: ");
        content_length_here = p;
        p = append_str(p, "NN\r\n");
        p = append_str(p, "\r\n");
        json_start = p;
        p = append_str(p, "{\"value1\":\"");
        p = append_str(p, "Sahköposti kotiautomaatiolta");
        p = append_str(p, "\",\"value2\":\"");
        p = append_str(p, "Tulipalo ");
        p = append_str(p, "\",\"value3\":\"");
        p = append_str(p, "talossa!!!");
        p = append_str(p, "\"}");

        compi = strlen(json_start);
        content_length_here[0] = '0' + (compi / 10);
        content_length_here[1] = '0' + (compi % 10);
        client.print(post_rqst);

      }
    }
    delay(50);
    edellinenLahetaPaloSposti = LahetaPaloSposti;
  }
}
```

Liite 9. IoT-palvelun ohjelmakoodi

```
/*
   OPINNÄYTETYÖ 2017
   KOTIAUTOMAATIOJÄRJESTELMÄ

   IoT, Thinger
   Thomas Kuronen
   10/2017
*/

#include <ESP8266WiFi.h>                        //Kirjasto mitä esp8266 vaatii.
#include <ThingerESP8266.h>                     //Kirjasto mitä thinger-palvelu vaatii.

#define USERNAME "[redacted]"                   //Thinger-palvelun käyttäjänimi
#define DEVICE_ID "[redacted]"                  //Thinger-palveluun tehty laitteen tunnus.
#define DEVICE_CREDENTIAL "[redacted]"          //Thinger-palveluun tehty laitteen salasana.

#define SSID "[redacted]"                       //Modeemin SSID
#define SSID_PASSWORD "[redacted]"              //Modeemin salasana

ThingerESP8266 thing(USERNAME, DEVICE_ID, DEVICE_CREDENTIAL);     //Asetetaan käyttäjätiedot

float arvo[4];                                  //Määritellään float-tyyppinen arvo-niminen taulukko missä on 4 saraketta.
int i;                                          //Määritellään integer-tyyppinen i-niminen muuttuja.

void setup() {

  Serial.begin(9600);                           //Asetetaan serialin datansiirtonopeus bittejä sekunissa (baud).
  pinMode(BUILTIN_LED, OUTPUT);                 //Määritellään sisäinen LED-valaisin OUTPUTIKSI, jotta sitä voidaan ohjata.

  thing.add_wifi(SSID, SSID_PASSWORD);          //Yhdistetään thinger-palveluun modeemin kautta.

  thing["lampotila1"] >> outputValue(arvo[0]);  //Lähetetään lämpötilatieto1 palveluun.
  thing["lampotila2"] >> outputValue(arvo[1]);  //Lähetetään lämpötilatieto2 palveluun.
  thing["prosentti"] >> outputValue(arvo[2]);   //Lähetetään moottorin pyörimisnopeustieto palveluun.
  thing["M20Setpoint"] >> outputValue(arvo[3]); //Lähetetään lämpötilan asetusarvotieto palveluun.
}


void loop() {
  thing.handle();                               //Suoritetaan thinger-toiminto. (Yhdistetään palveluun ja lähetetään tiedot)

  if (Serial.available())                       //Jos talon Arduino Megalta on tulossa serial-väylältä tietoa niin menemme if-rakenteeseen.
  {
    for (i = 0; i < 4; i++)                     //For silmukka pyörähtää 4 kertaa.
    {
      arvo[i] = Serial.read();                  //Luetaan väylältä tulevaa tietoa ja tallennetaan se arvo-muuttujan sen hetkiseen sarakkeeseen.
      if(i != 2)                                //Jos väylältä tuleva tieto on jotai muuta kuin moottorin pyörimisnopeus,
      {
      arvo[i] = arvo[i]/5;                      //Jaamme väylältä tulevan tiedon arvolla 5.
      }
      delay(100);                               //Odotetaan 100ms. (Tällä on ajoitettu lähetys ja vastaanottopää pyörähtämään samassa syklissä.)

    }
    i = 0;                                      //Kirjoitetaan muuttujaan i arvo 0.
  }
/*
 * Lämpötila-,moottorin pyörimisnopeus- ja lämpötilan asetusarvotiedot tulevat talon Arduino Megalta serial-väylää pitkin.
 * Tietoja on 4 kappaletta, joten for silmukka myös pyörähtää 4 kertaa. Arduino Mega kertoo tiedot (poislukien moottorin pyörimisnopeus) luvulla 5,
 * ennen kuin se lähettää tiedot. Tiedot tulevat perille tavuina (kokonaislukuna) minkä jälkeen ne pitää vastaanottopäässä jakaa luvulla 5.
 * Luku 5 valikoitui kertoimeksi siksi, koska tavun suuruus voi olla maksimissaan 255. Tämä tarkoittaa että suurin mahdollinen lämpötila
 * mikä serial-väylä onnistuu lähettämään ilman ongelmia on 255/5=51. 51 celcius astetta riittää työssä hyvin.
 *
 * Tietojen lähettäminen serialia pitkin on hitaampaa kuin tietojen lukeminen, joten lukemista on hidastettu 100 millisekunin viiveellä.
 * Tällä ajoitukset osuvat täydellisesti lähetys ja vastaanottopään kanssa yhteen ja oikeat tiedot saadaan tallennettua oikeisiin muuttujiin.
 * Tietojen lukeminen olisi voinut suorittaa myös tarkkailemalla Serial.available()-arvoa. Serial.available() palauttaa lukemattomien tavujen
 * määrän serial-väylällä.
 */
}
```