Tampereen yliopisto

Roope Keskinen

KYLPYTYNNYRIN LÄMMITYSAVUSTIN

Informaatioteknologian ja viestinnän tiedekunta
Kandidaatintyö
Huhtikuu 2019

TIIVISTELMÄ

Roope Keskinen: Kylpytynnyrin lämmitysavustin
Kandidaatintyö
Tampereen yliopisto
Informaatioteknologian ja viestinnän tiedekunta
04 2019

Työn tavoitteena oli suunnitella ja rakentaa laite, joka avustaa kylpytynnyrin lämmittämisessä. Laite liitetään kylpytynnyrin kylkeen koukulla ja siitä menee kylpytynnyrin sisään kaksi anturia: vesianturi ja lämpötila-anturi. Laite avustaa lämmittäjää lähettämällä tekstiviestejä, kun veden pinta saavuttaa halutun korkeuden ja veden lämpötila saavuttaa halutun lämpötilan. Laite voidaan myös asettaa muistuttamaan halutun ajan päästä puiden lisäämisestä kylpytynnyrin pesään.

Työ koostuu kirjallisuus- ja rakenteluosuudesta. Kirjallisuusosuudessa tutustutaan käytettyjen komponenttien ja moduulien toimintaan sekä mitoittamiseen. Rakenteluosuudessa rakennetaan suunniteltu laite ja tutkitaan täyttääkö laite sille asetetut vaatimukset.

Laitteen ohjaamiseen käytetään ATmega328p-mikrokonrolleria, joka ohjelmointiin käyttäen Arduino-ohjelmointiympäristöä sen helppokäyttöisyyden ansiosta. Tekstiviestien lähettämiseen laitteessa käytetään valmista GSM-moduulia, joka käyttää SIM5320E-sirua. Laitteen tilan seurantaan ja ohjaamiseen käytetään LCD-näyttöä. GSM-moduulin ohjaamiseen hyödynnytetään UART-sarjaliikenneväylää ja LCD:n ohjaamiseen I^2C-väylää. Laitteen lämpötila-anturina käytetään termistoria ja vesianturina käytetään itse toteutettua anturia, jonka toiminta perustuu jännitteenjakoon. Työssä toteutettiin itse myös step-up-muunnin, jota käytetään nostamaan akun jännite laitteelle sopivaksi.

Laitteen rakentamisessa ei esiintynyt suurempia ongelmia ja suunniteltu laite toimi hyvin käytännössä. Laite täyttää kaikki sille luvussa 2 asetetut vaatimukset, joten laitteen toteuttaminen voidaan todeta onnistuneeksi.

Avainsanat: Elektroniikka, Sulautettujärjestelmä, GSM, LCD, Termistori, Mikrokontrolleri, UART, I^2C

ALKUSANAT

Idea tälle kandidaatintyölle syntyi, kun perheenjäsen ehdotti, että suunnittelisin laitteen, joka avustaa kylpytynnyrin lämmityksessä. Laitteen toteuttaminen oli mahdollista elektroniikkaharrastuksen ja elektroniikanopintojen kautta opittujen taitojen avulla.

Haluan kiittää kandidaatintyöni ohjaajaa Katja Lainetta työn ohjauksesta ja neuvojen antamisesta. Kiitän myös läheisiäni, jotka ovat tukeneet ja avustaneet työni aikana.

Tampereella, 19.4.2019

Roope Keskinen

SISÄLLYSLUETTELO

LYHENTEET JA MERKINNÄT

BMS	engl. Battery Management System, Akun suojapiiri.
CPU	engl. Central Processing Unit, Keskusyksikkö.
EEPROM	engl. Electronically Erasable Programmable Read-Only Memory, Flash-muistityyppi.
GSM	engl. Global System for Mobile Communications, Matkapuhelinjärjestelmä.
I^2C	engl. Inter-Integrated Circuit bus, Sarjaliikenneväylä.
IC	engl. Integrated Circuit, Mikropiiri.
ISP	engl. In-system programming, Mikrokontrollerin ohjelmointiväylä.
LCD	engl. Liquid crystal display, Nestekidenäyttö
PDU	engl. Protocol Data Unit, Yksittäinen datayksikkö, jota siirretään kahden laitteen välillä.
SRAM	engl. Static Random Access Memory, RAM-muistityyppi.
SCL	engl. Serial Clock Line, I^2C-väylän kellojohdin.
SDA	engl. Serial Data Line, I^2C-väylän datajohdin.
SIM	engl. Subscriber Identification Module. Kortti, joka sisältää mobiililiittymän tiedot.
UART	engl. Universal Asynchronous Receiver Transmitter, Sarjaliikenneväylä
USB	engl. Universal Serial Bus. Sarjaliikenneväylä.

1. JOHDANTO

Nykyään suuri osa laitteista on sulautettuja järjestelmiä eli elektronisia laitteita, jotka sisältävät mikroprosessorin. Tämä johtuu siitä, että laitteen toiminnallisuus voi olla monta kertaa yksinkertaisempaa toteuttaa mikroprosessorilla ohjelmallisesti kuin tekemällä koko laite analogisesti. Mikroprosessorien kehittyminen ja halpeneminen ovat mahdollistaneet sulautettujen järjestelmien yleistymisen. Laitteet, jotka eivät ole välttämättömiä, mutta helpottavat jonkin yksittäisen asian tekemistä, ovat yleistyneet. Esimerkkinä tällaisesta laitteesta on tämän kandidaatintyön aihe – kylpytynnyrin lämmitysavustin.

Kylpytynnyrin lämmittäminen on aikaa vievä projekti, jossa kylpytynnyrin tilaa on tarkkailtava jatkuvasti. Myös puiden lisääminen kylpytynnyrin pesään voi helposti unohtua tai veden haluttu lämpötila voi ylittyä helposti. Kylpytynnyrin jatkuva tarkkailu voi olla rasittavaa. Kandidaatintyön tarkoituksena on suunnitella ja toteuttaa laite, joka avustaa kylpytynnyrin lämmittämisessä. Laitteen on tarkoitus avustaa lämmittäjää lähettämällä lämmittäjälle viestejä kylpytynnyrin tilasta. Työssä asetetaan toteutettavalle laitteelle vaatimukset, käydään läpi valittujen komponenttien toiminta ja mitoittaminen sekä, kuinka komponentit yhdessä lopulta muodostavat toimivan kokonaisuuden. Kandidaatintyössä teoria on rajattu käsittelemään työn kannalta tärkeiden komponenttien toiminnan perusteita ja mitoittamista.

Tämä kandidaatintyö koostuu kahdesta osuudesta. Ensimmäisessä osuudessa eli luvussa 2 käydään läpi laitteelle asetetut vaatimukset, laitteen yleiskuva, käytettyjen komponenttien toiminta ja mitoittaminen. Toisessa osuudessa eli luvussa 3 käydään läpi laitteen piirilevyjen suunnittelu, laitteen rakentaminen, ohjelmoiminen ja testaaminen. Viimeisessä luvussa eli yhteenvedossa tarkastellaan lämmitysavustimelle asetettujen vaatimusten toteutumista ja jatkokehitysmahdollisuuksia.

2. LAITTEEN TOIMINTA

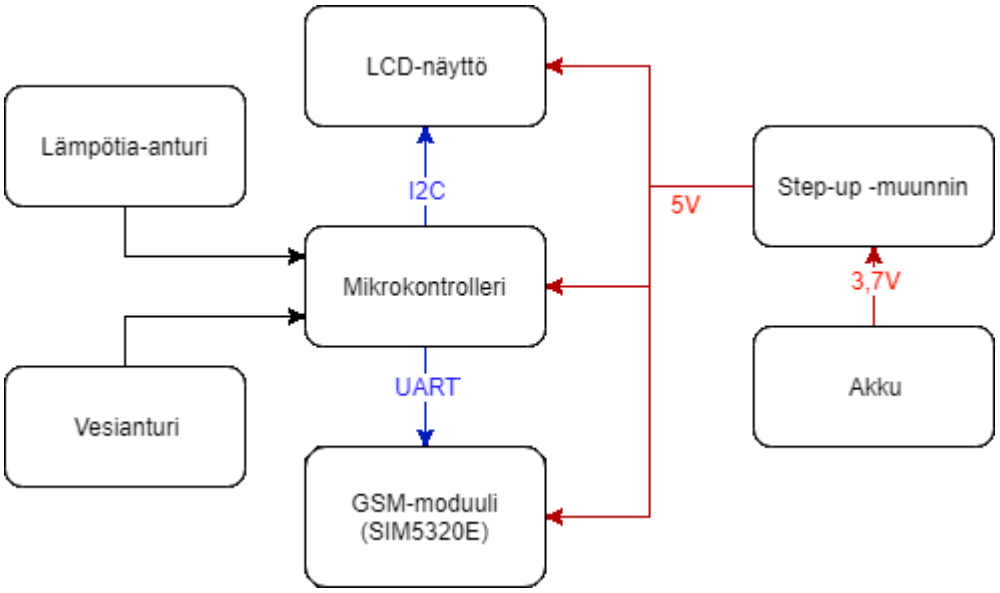
Suunniteltava kylpytynnyrin lämmitysavustin on laite, joka avustaa kylpytynnyrin lämmityksessä mittaamalla kylpytynnyrin tilaa ja ilmoittamalla tilan avustimen käyttäjälle. Alla on lista ominaisuuksista, jotka suunniteltavalla laitteella tulee olla, jotta sen toteuttaminen voidaan todeta onnistuneeksi.

- Ilmoittaa, kun vedenpinta saavuttaa halutun korkeuden.
- Ilmoittaa, kun veden lämpötila saavuttaa halutun arvon.
- Ajastin, joka muistuttaa puiden lisäämisestä halutun ajan päästä.
- Ilmoitukset toimitetaan tekstiviestinä.
- Laitteessa on näyttö, josta voi seurata laitteen ja kylpytynnyrin tilaa.
- Käyttöliittymä on yksinkertainen ja helppokäyttöinen.
- Laitteen on toimittava akun varassa useamman tunnin ajan (vähintään 5 h).
- Laitteen kotelo on roiskeenkestävä.

Laitteen mittaamien tietojen ilmoitustavaksi valittiin tekstiviesti, sillä kylpytynnyrin lämmitysolosuhteissa etäisyys kylpytynnyriin on todennäköisesti suuri. Tällaisessa tilanteessa summeri ei olisi riittävä keino tiedon ilmoittamiseen. Sähköpostia voitaisiin myös käyttää, mutta viestin vastaanottavat laitteet eivät välttämättä tarkista sähköpostia usein, jolloin viestin lähettämisen ja lukemisen välille voi syntyä suuria viiveitä.

2.1 Yleiskuva

Suunniteltu järjestelmä koostuu seitsemästä eri kokonaisuudesta ja kahdesta tiedonsiirtoväylästä. Järjestelmän rakennetta havainnollistaa kuvan 1 lohkokaavio.

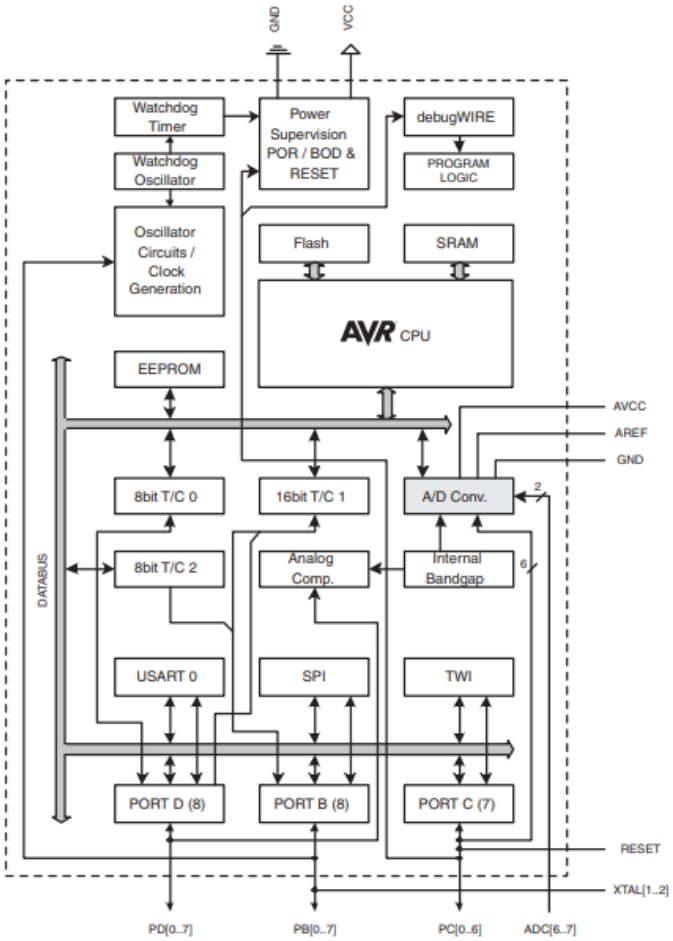


Kuva 1. *Järjestelmän lohkokaavio.*

Järjestelmä saa käyttöjännitteensä akusta, joka on nostettu halutulle tasolle step-up-muuntimella. Järjestelmää ohjaa mikrokontrolleri, joka saa tiedon kylpytynnyrin vedenpinnan korkeudesta ja veden lämpötilasta lämpötila- ja vesianturilla. Järjestelmää ohjataan ja sen tilasta viestitään käyttäjälle LCD:n (Liquid Crystal Display) avulla. LCD:n kanssa mikrokontrolleri kommunikoi I^2C-väylää (Inter-Integrated Circuit bus) käyttäen. Viimeinen osa järjestelmää on GSM-moduuli (Global System for Mobile Communications), jonka avulla järjestelmä lähettää tekstiviestejä käyttäjälle. GSM-moduulin kanssa mikrokontrolleri kommunikoi käyttäen UART-sarjaliikenneväylää (Universal Asynchronous Receiver Transmitter). Järjestelmän osien toimintaa ja mitoitusta kuvataan seuraavissa luvuissa tarkemmin.

2.2 Mikrokontrolleri

Mikrokontrollerit ovat pieniä yhden piirin tietokoneita, jotka sisältävät toiminnallensa tärkeät osat yhdelle piirille integroituna. Mikrokontrollerit sisältävät ainakin suorittimen ohjelman suorittamista varten, muistia ohjelman ja muuttujien tallettamista varten sekä I/O-portteja ympäröivän maailman kanssa vaikuttamiseen. [1] Kuva 2 havainnollistaa erään mikrokontrollerin rakennetta, ja siitä on löydettävissä edellä mainitut osat.



Kuva 2. ATmega328p-mikrokontrollerin lohkokaavio [2].

Mikrokontrollerin keskeisenä osana on sen CPU (Central Processing Unit) eli suoritin. Suoritin lukee konekieliset käskyt ohjelmamuistista ja suorittaa ne [1]. Suoritin myös käyttää mikrokontrollerin muita osia dataväylän kautta. Kuvassa 2 suorittimen yläpuolella on ohjelmamuisti eli Flash-muisti ja SRAM-muisti (Static Random Access Memory) eli dynaaminen muisti. Ohjelmamuisti on jaettu kahteen osaan: ohjelmamuistiksi ja käynnistyslataajan muistiksi. Ohjelmamuisti sisältää kontrollerille ladatun ohjelman, ja käynnistyslataajan muisti sisältää käynnistyslataajan eli ohjelman, jolla kontrolleri voidaan ohjelmoida ilman erillistä ohjelmointilaitetta. Flash-muisti on ei-haihtuvaa, ja se on jaettu lohkoiksi, joten muistista ei voi poistaa yksittäistä tavua vaan on poistettava lohko kerrallaan. Flash-muisti myös kestää vain rajallisen määrän kirjoituksia eli 10 000 kirjoitusta. Dynaamista muistia käytetään säilyttämään ajonaikaisia muuttujia. Dynaaminen muisti koostuu SRAM-soluista, jotka pitävät tietonsa tallessa ilman tarvetta virkistää soluja niin kauan kuin käyttöjännite on kytkettynä. SRAM-muisti on huomattavasti nopeampaa kuin Flash-muisti, mutta sitä on käytettävissä vain melko rajallinen määrä. Mikrokontrollerissa on myös EEPROM-muisti (Electronically Erasable Programmable Read-Only Memory), joka rakenteeltaan muistuttaa Flash-muistia, mutta siihen voi kirjoittaa tavu kerrallaan.

Tällöin se soveltuu tallentamaan muuttujia, joiden halutaan säilyvän käynnistysten välillä. EEPROM-muistisolu kestää 100 000 kirjoitusta, mutta sitä on saatavilla melko rajoitetusti ohjelmamuistiin verrattuna. [1][2]

Mikrokontrolleri saa käyttötaajuuden joko sisäisestä tai ulkoisesta oskillaattorista. Käyttötaajuus kuvaa sitä, kuinka monta käskyä kontrolleri voi suorittaa sekunnissa. Esimerkiksi 16 MHz kellolla kontrolleri suorittaa 16 000 000 käskyä sekunnissa. Kontrollerin sisäinen oskillaattori on 8 MHz RC-oskillaattori, joka mahdollistaa kontrollerin käytön ilman ulkoista oskillaattoria, mutta sen taajuus on herkkä lämpötilan- ja jännitteenvaihteluille. Sisäisessä oskillaattorissa voidaan käyttää myös esijakajaa, jolloin käyttötaajuutta saadaan pienennettyä, mikäli halutaan vähentää virrankulutusta. Käyttämällä ulkoista oskillaattoria voidaan käyttää jopa 20 MHz käyttötaajuutta, joka on vähemmän altis lämpötilan- ja jännitteenvaihteluille. [1][2]

Ajastin (kuvassa 2 T/C) on mikrokontrollerin laskuri, jonka arvoa voidaan kasvattaa sisäisillä tai ulkoisilla signaaleilla. Mikäli laskurin arvoa kasvatetaan kellotaajuudella, on kyseessä ajastin, joka voidaan asettaa aiheuttamaan keskeytys sen yli vuotaessa tai halutun arvon saavuttaessa. Ajastimeen voidaan asettaa myös esijakaja, jolloin laskurin arvoa ei kasvateta jokaisella kellosignaalilla, jolloin ajastin saadaan askeltamaan hitaammin. Ajastimia voidaan käyttää tahdistamaan aikakriittisiä tapahtumia. ATmega328p-mikrokontrolleri sisältää yhden 8-bittisen ja kaksi 16-bittistä ajastinta. 8-bittisellä ajastimella voidaan laskea arvot 255:een asti ja 16-bittisellä 65 535:een asti. Mikrokontrolleri sisältää myös Watchdog-ajastimen, joka on ajastin, jota käytetään tarkistamaan, toimiiko ohjelma vai onko se jäänyt jumiin. Watchdog-ajastin nollaa kontrollerin tai aiheuttaa keskeytyksen, mikäli sitä ei nollata ohjelmassa ennen kuin se saavuttaa huippuarvonsa. Aika, jossa Watchdog-ajastin nollaa kontrollerin, voidaan asettaa halutuksi asettamalla sopiva esijakaja ajastimelle. Aika voidaan asettaa välille 16 ms – 8 s. Watchdog-ajastin sisältää oman kellon, jolloin se on riippumaton kontrollerin kellosta. [1][2]

Mikrokontrollerin I/O-portteja (kuvassa 2 PORT) käytetään ympäristön kanssa vuorovaikuttamiseen. I/O-portit voivat olla joko digitaalisia tai analogisia. Digitaaliset I/O-portit voivat olla kolmessa eri tilassa, jotka ovat korkea ulostulo (Vcc), matala ulostulo (GND) ja korkeanimpedanssintila eli sisääntulo. Ulostulona I/O-portin impedanssi on asetettuna pieneksi ja ulostulo voidaan asettaa korkeaksi tai matalaksi, jolloin on mahdollista ohjata ulkoisia komponentteja. Ulostulon impedanssi on pieni, jotta se ei rajoittaisi ulostulon virtaa tai sen yli ei syntyisi suuria jännitehäviöitä. Sisääntulona ollessa I/O-portin impedanssi on asetettu korkeaksi ja sen tila voidaan lukea loogiseksi nollaksi tai ykköseksi. Suuren impedanssi ansiosta I/O-portti ei kuormita ulkoista piiriä, joten mikrokontrolleri ei vaikuta sen toimintaan. I/O-portit sisältävät myös ylösvetovastuksen, joka aktivoimalla

sisääntulo voidaan viedä käyttöjännitteeseen. Osa mikrokontrollerien I/O-porteista mahdollistavat myös analogisia toimintoja digitaalisten lisäksi. Toimiessaan sisäänmenona analogiset I/O-portit käyttävät AD-muunninta muuntamaan analogisen arvon digitaaliseksi, jota voidaan käsitellä ohjelmallisesti. AD-muuntimilla on eri suuruisia resoluutioita, jotka määrittävät kuinka pieniä jännitteeneroja ne kykenevät erottamaan. ATmega328p-mikrokontrolleri käyttää 10-bittistä AD-muunninta, joten se voi erottaa 1024 eri arvoa maan ja käyttöjännitteen väliltä. Analoginen ulostulo on tehty käyttäen DA-muuntimia, jotka voivat muuttaa ulostulon jännitettä käyttäen pulssinleveysmodulaatiota. Pulssinleveysmodulaatiossa luodaan kanttiaalto, jonka taajuus on vakio, mutta sen pulssisuhdetta eli kanttiaallon päällä- ja poissaoloaikaa voidaan säätää. Mitä suuremman osan jaksosta pulssi on päällä, sitä suurempi on ulostulon keskimääräinen jännite. Myös DA-muuntimilla on eri resoluutioita, jotka määräävät kuinka monta jännitteen arvoa ne voivat tuottaa maan ja käyttöjännitteen väliltä. ATmega328p-mikrokontrollerin DA-muuntimen resoluutio on 8-bittiä, jolloin se kykenee tuottamaan 256 diskreettiä jännitteen arvoa. [1][2] Osaa mikrokontrollerien I/O-porteista voidaan käyttää myös sarjaliikenneväylinä, joita käsitellään luvussa 2.3.

Työssä valittiin käytettäväksi mikrokontrolleriksi ATmega328p, sillä se tarjoaa riittävän määrän I/O-portteja ja sarjaliikenneväyliä ja minulla on valmiiksi kokemusta kyseisen mikrokontrollerin käytöstä. ATmega328p on myös tuettu Arduino-ympäristössä, jolloin käyttöön saa valmiita ohjelmistokirjastoja.

2.3 Sarjaliikenneväylät

Sarjaliikenneväylät ovat tiedonsiirtoväyliä, joissa dataa siirretään peräkkäin bitti kerrallaan. Sarjaliikenneväylät voidaan jakaa kahteen ryhmään sen mukaan pystyvätkö ne lähettämään ja vastaanottamaan dataa samaan aikaan. Mikäli samanaikainen lähettäminen on mahdollista, on kyseessä full-duplex-järjestelmä ja muussa tapauksessa on kyseessä half-duplex-järjestelmä. Sarjaliikenteessä laitteiden välinen tiedonsiirto voi olla synkronista tai asynkronista. Synkronisessa tiedonsiirrossa laitteet synkronoivat toisensa ja lähettävät jatkuvasti dataa pysyäkseen tahdissa tai dataväylän lisäksi laitteita yhdistää kellosignaali, joka määrää tiedonsiirtonopeuden. Asynkronisessa tiedonsiirrossa lähettävää ja vastaanottavaa järjestelmää ei ole tahdistettu toisiinsa, mutta jokaisen lähetettävän tavun yhteydessä on lähetettävä aloitus- ja lopetusbitti, jotta vastaanottava järjestelmä tietää milloin lähetys alkaa ja loppuu. [3][4]

Asynkronisen tiedonsiirron etuna on kellosignaalin puute, jolloin ei tarvitse yhtä montaa johdinta kuin synkronisessa tiedonsiirrossa. Synkroninen tiedonsiirto on puolestaan nopeampaa, sillä siinä ei tarvitse lähettää ylimääräisiä aloitus- ja lopetusbittejä. [4]

2.3.1 UART

UART on sarjaliikennepiiri, joka mahdollistaa asynkronisen sarjamuotoisen tiedon vastaanottamisen ja lähettämisen kahden laitteen välillä käyttäen RS-232-standardia. RS-232-standardissa kaksi laitetta on kytkettynä toisiinsa kolmella johtimella. Kaksi näistä johtimista on tarkoitettu tiedonsiirtoa ja yksi yhteistä maata varten. Yhdessä tiedonsiirtojohtimessa tieto voi kulkea vain yhteen suuntaan, joten tiedonsiirtojohtimia tarvitaan kaksi. [5]

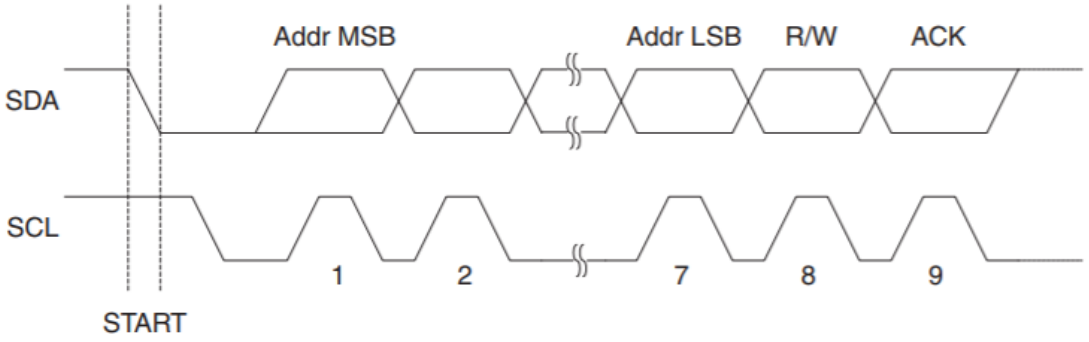
RS-232-standardin mukainen bittikehys koostuu aloitusbitistä, databiteistä, pariteettibitistä ja lopetusbitistä. Aloitusbitti on aina nolla, jolloin se vetää dataväylän nollaksi. Nollaksi vedetty dataväylä viestii vastaanottajalle, että lähetys on alkamassa. Databittejä voi olla viidestä yhdeksään, mutta yleensä lähetyksessä lähetetään kahdeksan databittiä eli yksi tavu. Pariteettibittiä käytetään lähetyksessä tapahtuneiden virheiden löytämiseen, mutta se ei kuitenkaan ole usein käytössä. Pariteettibittiä käytetään laskemalla yhteen databitit ja tarkistamalla onko tulos pariton vai parillinen. Pariton tulos vastaa lukua yksi ja parillinen lukua nolla. Vastaanottaja voi nyt tarkistaa, onko lähetys vastaanotettu oikein laskemalla databitit yhteen ja vertaamalla tulosta pariteettibittiin. Lopetusbitti koostuu yhdestä tai kahdesta bitistä, joiden arvo on yksi. Ylös asetettu dataväylä kertoo vastaanottajalle, että lähetys on päättynyt. [3][4][5] Työssä käytetään UART-sarjaliikennepiiriä GSM-moduulin kanssa kommunikointiin.

2.3.2 I^2C

I^2C on synkronoitu half-duplex sarjaliikenneväylä, joka toimii isäntä-orja-periaatteella eli isäntälaite aloittaa aina kommunikoinnin. I^2C-sarjaliikenneväylässä jokaisella orjalaitteella on seitsemänbittinen osoite eli yhteen väylään voi olla kytkettynä 128 orjalaitetta. Fyysisesti I^2C-väylä koostuu kahdesta johtimesta, SCL (Serial Clock Line) ja SDA (Serial Data Line), ja niihin liitetyistä ylösvetovastuksista eli vastuksista, joiden kautta johtimet on viety käyttöjännitteeseen. SCL-johdin on tarkoitettu kellosignaalia varten ja SDA on tarkoitettu dataa varten. [5]

I^2C-väylässä kommunikaatio alkaa viemällä SDA-johdin alas isäntälaitteen toimesta ja lopetus tapahtuu viemällä SDA-johdin ylös. Aloituksessa ja lopetuksessa SCL-johdin on jätetty ylös. Väylässä on myös mahdollisuus toistuvaan aloitukseen, jossa tehdään uusi aloitus ennen lopetusta. Toistuva aloitus mahdollistaa väylän pitämisen yhden isäntälaitteen hallinnassa, jolloin muut isäntälaitteet eivät pääse ottamaan väylää haltuunsa, jolloin on mahdollista lähettää useampi tavu ilman keskeytyksiä. [2]

I²C-väylässä jokainen lähetettävä paketti koostuu yhdeksästä bitistä ja jokainen bitti luetaan, kun kellosignaali on ylhäällä. Osoitepaketissa seitsemän ensimmäistä biittiä kertovat orjalaitteen osoitteen, kahdeksas R/W-bitti kertoo, kirjoitetaanko vai luetaanko orjalaitteesta ja viimeinen bitti on ACK-bitti eli kuittausbitti. ACK-bitin lähettää orjalaite yhdeksännellä kellosignaalilla vetämällä SDA-johtimen alas, joka viestii isäntälaitteelle, että viesti on saatu perille. [2] Kuvassa 3 on esitetty osoitepaketin rakenne. Datapaketit koostuvat kahdeksasta databitistä ja ACK-bitistä [2].

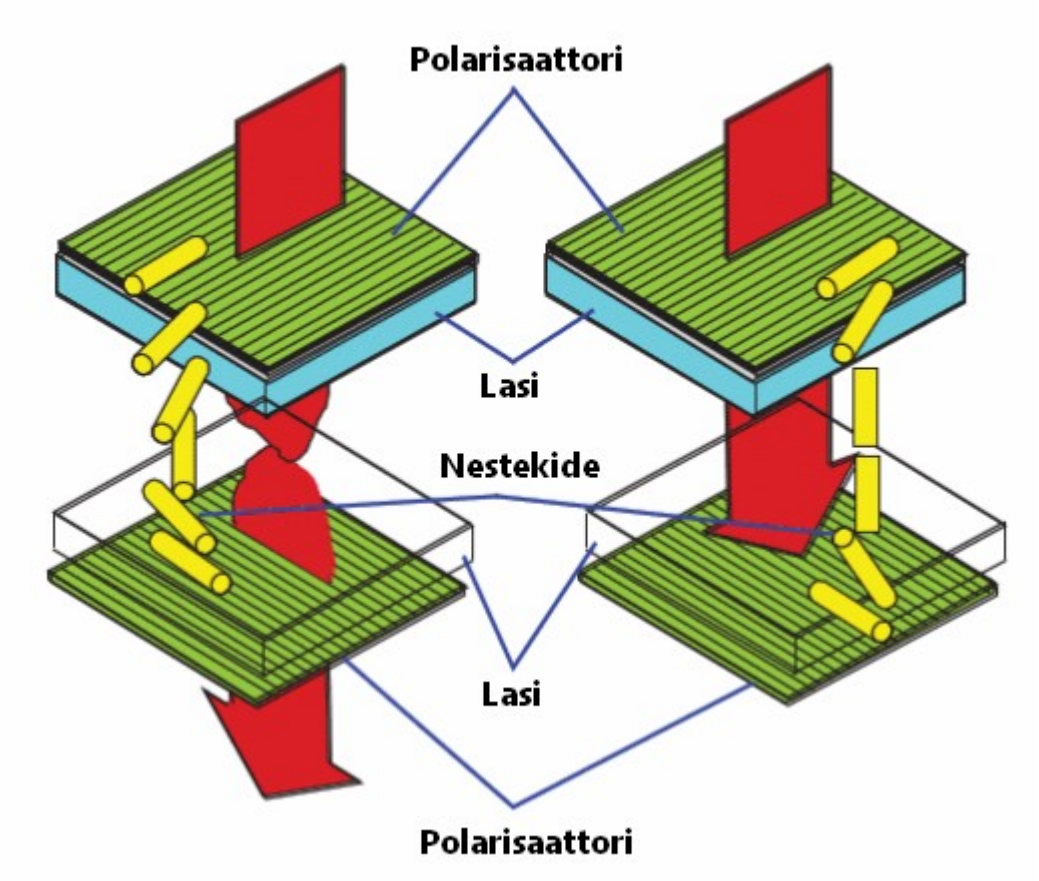


Kuva 3. *I2C-väylän osoitepaketin rakenne [2].*

I²C-väylän tukemat nopeudet ovat 100kbps, 400kbps ja 3,4Mbps, mutta sen kantamaa rajaavat väylän kapasitanssi ja lähetysnopeus [5]. I²C-väylä sopii ominaisuuksiltaan tapauksiin, joissa väylän pituus on lyhyt, mutta tarvitaan suurta tiedonsiirtonopeutta ja useampaa laitetta samassa väylässä. Työssä I²C-väylää käytetään LCD-näytön kanssa kommunikointiin.

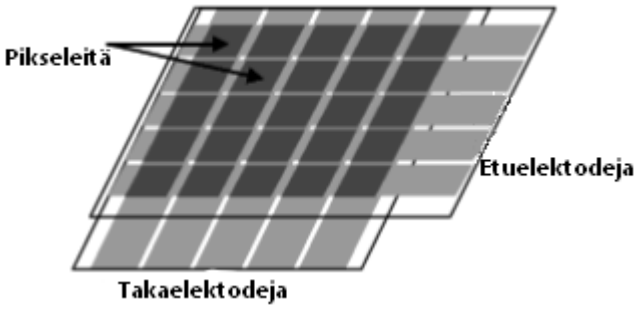
2.4 Nestekidenäyttö

LCD eli nestekidenäyttö on näyttötyyppi, jossa kuva muodostetaan nestemäisillä kiteillä, jotka polarisoivat valoa. LCD koostuu kahdesta polarisaattorista, jotka ovat suorassa kulmassa toisiinsa nähden, jolloin valo ei pääse läpäisemään niitä. Polarisaattoreiden välissä on kaksi lasipaneelia, joiden välissä on ohut kerros nestekidettä. Normaalissa tilassa nestekide on 90° kiertyneenä, jolloin se polarisoi valoa 90° ja valo pääsee läpäisemään paneelin. Kun nestekiteet altistetaan sähkökentälle ne suoristuvat, jolloin nestekiteet eivät polarisoi valoa eikä se pääse läpäisemään paneelia. Kuva 4 havainnollistaa nestekidenäytön rakennetta. Näin pystytään säätämään LCD:n yksittäisiä pikseleitä tai segmenttejä päälle tai pois päältä. Koska LCD ei kykene itsestään tuottamaan valoa, tarvitsee se toimiakseen joko taustavalon tai sen pitää heijastaa näytöstä läpi kulkeva valo takaisin katsojalle. [6]



Kuva 4. *Nestekidenäytön rakenne. Vasemmassa kuvassa ulkoista sähkökenttää ei ole ja oikeassa on. Muokattu lähteestä [6].*

LCD:t voivat koostua joko segmenteistä tai pikseleistä. Segmenttinäytöissä jokaiseen segmenttiin kytkeytyy oma elektrodi, jolla kyseistä segmenttiä ohjataan. Tämä kuitenkin käy ongelmalliseksi, kun pitäisi ohjata pikselinäyttöä, sillä pikselien määrä on erittäin suuri. Esimerkiksi työssä käytetty näyttö koostuu 3200 pikselistä, jolloin sen ohjaamiseen tarvittaisiin vähintään 3200 väylää. Kuitenkin kytkemällä näytön pikselit matriisiin eli kytketään rivien pikselien elektrodit toisiinsa ja sarakkeiden pikselien elektrodit toisiinsa kuten kuvassa 5, ohjaamiseen tarvitaan enää 132 väylää, mutta pikselit pitää multipleksata. Multipleksaaminen tarkoittaa sitä, että vain yhden rivin tai sarakkeen pikselit voivat olla päällä kerrallaan, mutta ohjattavia rivejä tai sarakkeita vaihdetaan niin nopeasti, että ihmiselle kaikki pikselit näyttävät olevan samaan aikaan päällä. Tällainen näyttö on nimeltään passiivimatriisinäyttö. On olemassa myös aktiivimatriisinäyttöjä, joissa pikselin tila säilyy muita päivittäessä. Passiivimatriisinäytöt ovat halvempia ja yleisimpiä pienessä mittakaavassa kuin aktiivimatriisinäytöt, mutta niiden kuvan laatu on huonompi. [6] Nestekidenäytöissä on tyypillisesti näytönohjainpiiri valmiiksi, jonka kanssa kommunikoidaan käyttäen sarja- tai rinnakkaisväylää, jolloin käyttäjän ei tarvitse huolehtia näytön ohjaamisesta vaan näytettävä tieto voidaan vain lähettää näytölle.



Kuva 5. *Passiivimatriisinäytön rakenne. Muokattu lähteestä [6].*

Tässä työssä näyttönä käytetään LCD:tä, joka on toteutettu passiivimatriisimenetelmällä. Näyttö koostuu neljästä rivistä ja kahdestakymmenestä sarakkeesta eli näyttö kykenee näyttämään 80 merkkiä kerrallaan. Jokaisen merkin korkeus on kahdeksan pikseliä ja leveys viisi pikseliä eli näyttö koostuu kaikkiaan 3200 pikselistä. Näytön kanssa kommunikoidaan käyttäen I^2C-väylää ja ohjelmallisesti kommunikointi tapahtuu käyttäen valmista kirjastoa, jotta ohjelma olisi yksinkertaisempi.

2.5 GSM-moduuli

Lämmitysavustimessa tekstiviestien lähettämiseen käytetään valmista GSM-moduulia. Moduuli on Keyestudion valmistama ja käyttää mobiiliverkon kanssa kommunikointiin SIM5320E-IC:tä (Integrated Circuit). Laitteessa käytetään valmista moduulia, sillä SIM5320E tarvitsee toimiakseen useita ulkoisia komponentteja kuten regulaattoreita, antenneja, liitännän SIM-kortille ja muita passiivisia komponentteja. Tästä syystä on kätevämpää käyttää valmista moduulia, jossa on kaikki tarvittavat komponentit, joten SIM5320E-IC:n ohjaamiseen tarvitaan vain yksi UART-väylä.

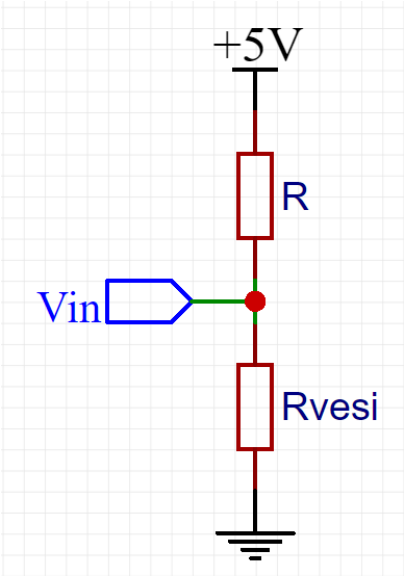
SIM5320E on IC, jolla voidaan liittyä GSM-verkkoon ja se mahdollistaa tekstiviestien lähettämisen ja vastaanottamisen, äänipuhelut sekä datan lähettämisen ja vastaanottamisen 3G nopeuksilla. IC mahdollistaa myös GPS-seurannan (Global Positioning System). IC:n ohjaaminen tapahtuu sarjaliikenneväylän kautta käyttäen AT-komentoja. AT-komennot ovat komentoja, joita käytetään ohjaamaan modeemeja. AT-komennot ovat merkkijonoja, joissa ensimmäiset merkit ovat aina ”AT” ja niitä seuraa haluttu komento. Esimerkiksi komennolla ”AT+CMGS="puhelinnumero”” asetetaan puhelinnumero tekstiviestin lähettämistä varten. [7][8]

Tässä työssä tekstiviestien lähettämiseen tarvitaan kahta eri komentoa. Ensimmäinen komento on ”AT+CMGF=<tila>”, jossa tila on 0 tai 1 ja sillä asetetaan tekstiviestin formaatti. Mikäli tilaksi asetetaan 0, niin tekstiviesti on kirjoitettava PDU (Protocol data unit) muodossa. Jos kuitenkin tilaksi asetetaan 1, niin tekstiviestit voidaan suoraan kirjoittaa tekstimuodossa. Valitaan käytettäväksi tilaksi 1, jolloin komento on ”AT+CMGF=1” ja haluttu viesti voidaan lähettää suoraan tekstinä. Toinen komento, jota tarvitaan, on ”AT+CMGS="puhelinnumero””, jossa puhelinnumero on tekstiviestin vastaanottajan puhelinnumero. Komennon syöttämisen jälkeen IC menee tilaan, jossa sille voidaan syöttää haluttu viesti lähettämällä se merkkijonona. IC lukee ja tallentaa merkkijonoa, kunnes sille lähetetään merkki ”0x1A”, joka tarkoittaa viestin päättymistä ja tällöin IC lähettää tekstiviestin. [8]

Jotta GSM-moduuli voisi liittyä matkapuhelinverkkoon tarvitsee se SIM-kortin ja puhelinliittymän. Lämmitysavustimeen päädyin ostamaan prepaid-liitttymän, sillä avustin ei lähetä paljon viestejä lyhyellä aikavälillä, joten ei ole kannattavaa ostaa kuukausittain maksettavaa liittymää. SIM5320E on myös suhteellisen kallis IC ja sen kaikkia ominaisuuksia ei hyödynnetä, joten työssä olisi voinut käyttää myös edullisempaa GSM-moduulia. Päädyin kuitenkin käyttämään SIM5320E-moduulia, sillä voin hyödyntää sen muita ominaisuuksia muissa projekteissa.

2.6 Vesianturi

Laitteessa vedenkorkeuden tunnistamiseen käytetään anturia, joka koostuu kahdesta johtimesta ja yhdestä vastuksesta R. Kummatkin johtimet menevät veteen, jolloin niiden välille syntyy veden aiheuttama resistanssi R_{vesi}. Toinen johtimista on kytkettynä maahan ja toinen on kytkettynä vastukseen R. Mikrokontrollerilla luetaan veden resistanssin yli oleva jännite, jolloin voidaan päätellä, onko anturi vedessä. Kuva 6 havainnollistaa vesianturin kytkentää.



Kuva 6. *Vesianturin toimintaperiaate.*

Anturin ollessa pois vedestä anturin piiristä puuttuu veden aiheuttama resistanssi, jolloin luettu jännite Vin on käyttöjännitteen suuruinen. Anturin ollessa vedessä luettava jännite perustuu jännitteenjakoon eli piiriin syötetty jännite jakaantuu sarjassa olevien vastuksien yli vastuksien resistanssien suhteessa. Vastuksen, joka on sarjassa toisen vastuksen kanssa, yli oleva jännite voidaan ratkaista kaavalla

$$U_1 = \frac{R_1}{R_1+R_2} U, \tag{1}$$

jossa R_1 on sen vastuksen resistanssi, jonka yli oleva jännite halutaan tietää, R_2 on toisen vastuksen resistanssi ja *U* on vastuksiin syötetty jännite.

Vesianturin lukemiseen käytetään digitaalista I/O-porttia, jolloin sisääntulon jännitteen on oltava suurempi kuin $0{,}7\,Vcc = 3{,}5\,V$ tai pienempi kuin $0{,}3\,Vcc = 1{,}5\,V$, jotta sisääntulo luetaan loogiseksi nollaksi tai ykköseksi [2]. Kun anturi ei ole vedessä, on sisääntulossa 5V, jolloin se luetaan loogiseksi ykköseksi. Anturin ollessa vedessä sisääntulossa on oltava alle 1,5 V, joten vastuksen *R* resistanssi on mitoitettava kaavalla 1, jossa U_1=5 V-1,5 V=3,5 V, *U*=5 V, R_1=*R* ja R_2 = R_{vesi}=200 kΩ, joka on veden mitattu resistanssi. Vastuksen *R* resistanssiksi saadaan *R*=467 kΩ. Laskettu arvo on raja-arvo, jolla sisääntulossa on 1,5 V, joten valitaan vastus, joka on suurempi kuin mitoitettu arvo eli valitaan 1 MΩ vastus.

2.7 Termistori

Laitteessa veden lämpötilan lukemiseksi käytetään termistoria eli vastusta, jonka resistanssi muuttuu merkittävästi lämpötilan muuttuessa vähän. Termistorit voidaan jakaa kahteen eri ryhmään: NTC- ja PTC-termistoreiksi. NTC-termistorien, (Negative Temperature Coefficient), resistanssi on negatiivisesti lämpötilariippuvainen eli sen resistanssi pienenee lämpötilan kasvaessa. PTC-termistorien, (Positive Temperature Coefficient), resistanssi on puolestaan positiivisesti lämpötilariippuvainen eli sen resistanssi kasvaa lämpötilan kasvaessa. NTC-termistorit ovat enemmän käytettyjä kuin PTC-termistorit, sillä ne omaavat lämpötilan mittaamisen kannalta suotuisat ominaisuudet, joten niitä käytetään paljolti lämpötilan mittaamiseen. [9] Yksi PTC-termistorien käyttökohteista on nollattavissa sulakkeissa, sillä ne toimivat sulakkeen tavoin lämpötilan noustessa ja palautuvat pieniresistanssiseen tilaan vian poistuessa.

Termistorin resistanssi ei ole lineaarisesti riippuvainen lämpötilasta, joten termistorin resistanssin ja lämpötilan suhdetta ei voi lineaarisesti approksimoida. Termistorin käyttäytymistä voidaan kuitenkin approksimoida Steinhart-Hart-yhtälöllä, joka on kolmannen asteen yhtälö, joka määrittää lämpötilan termistorin resistanssin funktiona. Steinhart-Hart-yhtälö on

$$\frac{1}{T} = A + B\ln(R) + C\,(\ln(R))^3, \tag{2}$$

jossa *T* on termistorin lämpötila Kelvineissä, *R* on termistorin resistanssi ja A, B ja C ovat termistorille ominaisia vakioita. [10] Termistorin valmistaja voi ilmoittaa vakioiden A, B ja C arvot tai ne voidaan ratkaista kaavoista

$$\frac{1}{T_1} = A + B\ln R_1 + C\,(\ln(R_1))^3 \tag{3}$$

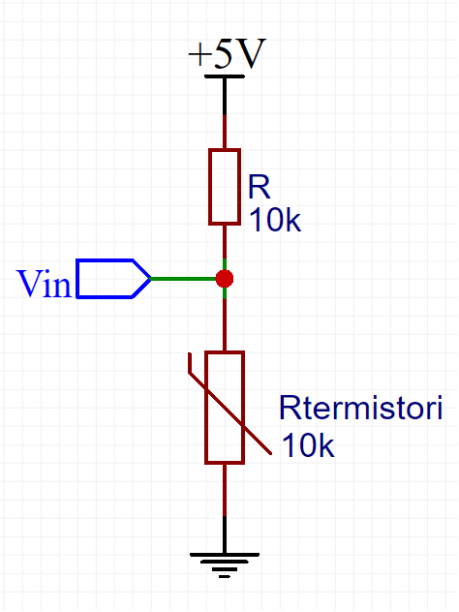
$$\frac{1}{T_2} = A + B\ln R_2 + C\,(\ln(R_2))^3 \tag{4}$$

$$\frac{1}{T_3} = A + B\ln R_3 + C\,(\ln(R_3))^3, \tag{5}$$

kun tiedetään termistorin resistanssi kolmessa eri lämpötilassa [9]. Termistorin resistanssin ja lämpötilan suhdetta voidaan kuitenkin mallintaa yksinkertaisemmalla β parametri -yhtälöllä, joka on Steinhart-Hart-yhtälö, josta vakiot A, B ja C on korvattu. β parametri -yhtälö on

$$\frac{1}{T} = \frac{1}{\beta} + \ln\frac{R}{R_0} + \frac{1}{T_0}, \tag{6}$$

jossa T on termistorin lämpötila Kelvineissä, β on termistorille ominainen vakio, T_0 on yleensä 298,15 K eli 25 °C, R_0 on termistorin resistanssi, kun lämpötila on 25 °C ja R on termistorin resistanssi lämpötilassa T. [10] Jotta termistorin mittaama lämpötila voitaisiin laskea, pitää mitata termistorin resistanssi. Mikrokontrollerilla ei kuitenkaan voi mitata suoraan resistanssia, kun taas jännitettä voi, joten termistori on liitettävä piiriin, jossa termistorin yli olevaa jännitettä voi mitata. Kun termistori liitetään sarjaan tunnetun vastuksen kanssa ja mitataan termistorin yli oleva jännite, voidaan termistorin resistanssi laskea käyttäen jännitteen jakoa eli kaava 1, jossa R_1 on ratkaistava termistorin resistanssi, R_2 on tunnetun vastuksen resistanssi, U on 5 V ja U_1 on termistorin yli oleva jännite. Termistorin kytkentä on havainnollistettu kuvassa 7.



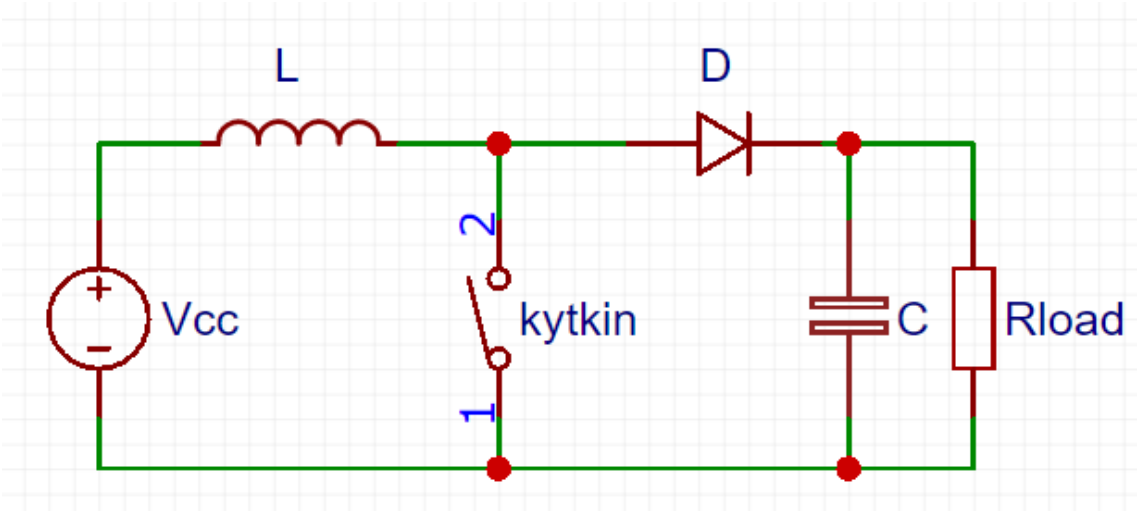
Kuva 7. *Termistorin kytkentä.*

Nyt lämpötila voidaan mitata mittaamalla termistorin ylioleva jännite ja muuntamalla se resistanssiksi kaavalla 1. Laskettu resistanssi voidaan sijoittaa kaavaan 6, jolloin saadaan laskettua termistorin mittaama lämpötila. Laitteessa käytetyn termistorin T_0 on 298,15 K, R_0 on 10 kΩ ja β on 3988 K [11].

2.8 Step-up -muunnin

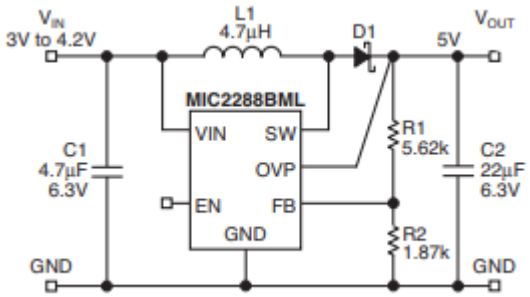
Step-up -muunnin on piiri, joka nostaa jännitettä eli muuntimen ulostulossa on suurempi jännite kuin sisääntulossa. Tyypillinen step-up -muunnin koostuu käämistä, kytkimestä, diodista ja kondensaattorista. Kuvassa 8 on esitetty step-up -muuntimen piirikaavio. Kun muuntimen kytkin on kiinni, kulkee käämin läpi virta, jota käämi pyrkii vastustamaan luoden magneettikentän ympärillensä. Kun kytkin avataan, pienenee käämin virta, jolloin

kelan magneettikenttä romahtaa aiheuttaen jännitepiikin. Nyt kela on sarjassa jännitelähteen kanssa, jolloin niiden yhteinen jännite on suurempi kuin sisääntulon jännite. Nyt suuremmalla jännitteellä ladataan ulostulon kondensaattori, jolloin ulostuloon on saatu aikaan suurempi jännite kuin sisääntulossa on. Kun kytkin taas suljetaan, alkaa käämi varastoida energiaa, mutta kondensaattori ylläpitää ulostulon jännitteen ja diodi estää sitä purkautumasta sisääntuloon. Kun kytkintä avataan ja suljetaan riittävän nopeasti, pysyy ulostulon jännite mahdollisimman vakiona. Säätämällä suhdetta kuinka kauan kytkin on päällä, voidaan säätää ulostulon jännitettä. [12]



Kuva 8. *Step-up -muuntimen piirikaavio.*

Tässä työssä käytetään step-up -muuntimen hallintaan MIC2288 IC:tä, sillä se tarvitsee toimiakseen vain muutaman ulkoisen passiivikomponentin. IC:n ulostulojännite voidaan myös asettaa halutuksi kahden vastuksen avulla ja takaisinkytkennän ansiosta ulostulojännite pysyy vakiona kuormankin vaihdellessa. IC:n datalehdeltä [13] saadaan tyypillinen kytkentä, kuva 9, mikäli halutaan ulostuloon viiden voltin jännite.



Kuva 9. *MIC2288 IC:n tyypillinen kytkentä [13].*

Kuvan 9 kytkennästä selviää kaikkien tarvittavien komponenttien koot, mutta tarkistetaan kuinka lähellä viittä volttia ulostulo on annetuilla vastuksien arvoilla, kun datalehdeltä selviää, että takaisinkytkennän jännite on aina 1,24 V. Ulostulon jännite voidaan selvittää

käyttäen kaavaa 1, kun siinä U_1 on 1,24 V, R_1 on 1,87 kΩ ja R_2 on 5,62 kΩ. Ulostulon jännitteeksi saadaan nyt 4,96V. Jännite on lähellä halutta arvoa, mutta se voisi olla hieman suurempi, joten valitaan vastukseksi R_1 1,82 kΩ, jolloin ulostulojännite on 5,07 V. Vaihdetaan vielä kytkennän käämi 10 µH kokoiseksi ja sisäänmenon kondensaattori 10 µF kokoiseksi, jotta muunnin kykenee antamaan enemmän virtaa.

2.9 Akku

Litiumioniakku on akku, jossa varauksenkuljettajana toimii litiumioni. Litiumioniakku koostuu katodista, anodista ja ne yhdistävästä elektrolyytistä. Katodi on valmistettu litiumoksidista ja anodi hiilipohjaisesta aineesta, joka yleensä on grafiitti. Litiumioniakun purkautuessa anodilta vapautuu negatiivisesti varautuneita elektroneja, jotka kulkevat ulkoisen piirin kautta katodille. Anodilta vapautuu myös positiivisesti varautuneita litiumioneja, jotka kulkevat elektrolyytin kautta katodille. Katodilla positiiviset litiumionit vastaanottavat negatiiviset elektronit, jolloin katodilla tapahtuu pelkistymisreaktio ja katodille muodostuu litiumia. Kun litiumioniakkua ladataan, tapahtuu reaktio käänteisessä suunnassa. [14][15]

Litiumioniakut eivät ole käytössä yhtä anteeksiantavaisia kuin muut akkutyypit vaan niitä on käytettävä turvallisten käyttörajojen sisäpuolella, jotta vältytään vahingoilta. Käyttörajoista poikkeaminen voi johtaa käyttöiän lyhenemiseen tai pahimmissa tapauksissa litiumioniakun tuleen syttymiseen. Litiumioniakkuja ei tulisi ikinä ladata liian suuriin jännitteisiin, sillä ne saattavat syttyä tuleen. Liikaa purkamista tulisi myös välttää, sillä ne voivat hajota. Liian suuri virranotto ja liian nopea lataus saattavat myös aiheuttaa litiumioniakun vahingoittumisen. Jotta litiumioniakkuja voitaisiin käyttää turvallisesti, tarvitaan lisäksi BMS (Battery Management System) eli suojapiiri. Suojapiirin tehtävänä on estää akun yli- ja alijännitteet sekä liian suuret virrat. Suojapiiri tarkkailee akun tilaa ja mikäli jokin edellä mainituista tiloista toteutuu, niin suojapiiri katkaisee akun yhteyden muuhun piiriin. Taulukossa 1 on esitettynä litiumioniakkujen hyötyjä ja heikkouksia.[16]

Taulukko 1. *Litiumioniakkujen hyödyt ja heikkoudet [14].*

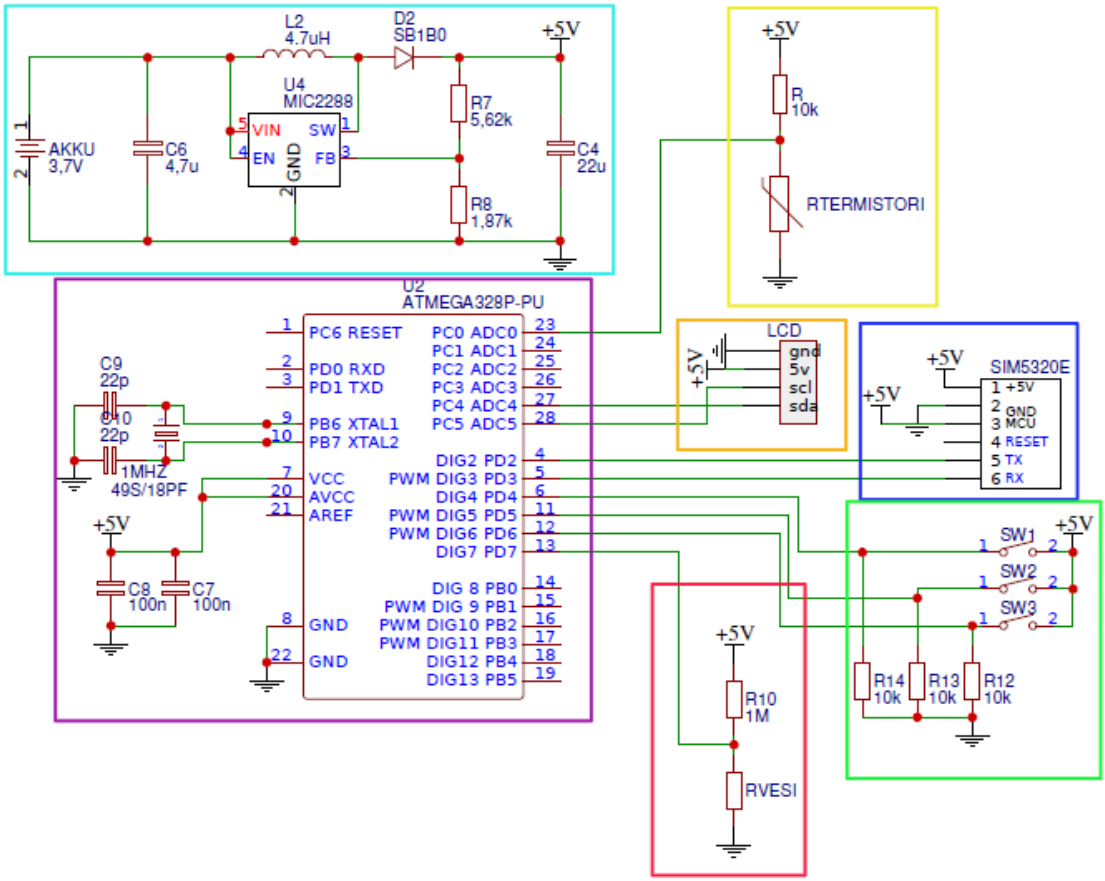
Hyödyt	Heikkoudet
Suuri kapasiteetti ja pieni sisäinen resistanssi	Tarvitsee suojapiirin.
Pitkä käyttö- ja säilytysikä.	Heikkenee suurissa lämpötiloissa ja varastoitaessa korkeissa jännitteissä.

Huoltovapaa.	Ei voi ladata nopeasti, mikäli lämpötila on nollan alapuolella.
Melko lyhyt latausaika ja yksinkertainen latausalgoritmi.	Kuljetusregulaatiot rajoittavat kuljettamista suurissa määrissä.
Vähäinen itsepurkautuminen.	

Lämmitysavustimeen valittiin käytettäväksi litiumioniakku, sillä se tarjoaa suuren kapasiteetin. Suuren kapasiteetin avulla voidaan saavuttaa vaadittu akunkesto mahdollisimman pienikokoisella akulla. Valitun litiumioniakun kapasiteetti on 2100 mAh ja sen kenno on malliltaan 18650 eli se on sylinteri, jonka halkaisija on 18 mm ja korkeus 65 mm [17]. Akun suojapiirinä käytetään geneeristä suojapiiriä, jonka valmistaja ei ole tiedossa, mutta myyjä lupaa, että piiri suojaa yli- ja alijännitteiltä ja liian suurelta virralta. Suojapiirin oikea toiminta todettiin kokeellisesti. Suojapiiri katkaisee yhteyden akkuun, mikäli jännite on alle 3,0 V, yli 4,2 V tai piirissä on oikosulku.

2.10 Kokonaisuuden toiminta

Kuvassa 10 on esitettynä kuinka aikaisemmissa luvuissa esitetyt komponentit yhdistyvät toisiinsa muodostaen toimivan kokonaisuuden. Kuvassa 10 vaaleansinisellä on rajattu step-up-muunnin, joka muuntaa akun 3,7 V jännitteen 5 V:ksi, jota käytetään muiden komponenttien käyttöjännitteenä. Violetilla merkittynä on mikrokontrolleri ja sen toimiakseen tarvitsemat kide ja kytkentäkondensaattorit. Mikrokontrolleri tarvitsee toimiakseen kytkentäkondensaattoreita, sillä mikrokontrolleri ottaa virtaa sykäyksissä, joita akku ei kykene antamaan toisin kuin kondensaattorit. Piirissä on kaksi kytkentäkondensaattoria eli yksi kullekin käyttöjännitteen sisäänmenolle. Mikrokontrolleriin on myös liitettynä 16 MHz kide, joka antaa kontrollerille käyttötaajuuden. Mikrokontrolleri sisältää myös oman oskillaattorin, mutta sitä ei käytetä, sillä ulkoinen kide tarjoaa nopeamman ja tarkemman taajuuden.



Kuva 10. *Laitteen komponentit ja kuinka ne yhdistyvät toisiinsa.*

Keltaisella kuvassa 10 on rajattuna lämpötilaa mittaava termistorikytkentä. Termistori on kytketty mikrokontrollerin analogiseen I/O-porttiin johtimella, jonka kautta mikrokontrolleri lukee termistorin yli olevan jännitteen ja muuntaa sen ohjelmallisesti lämpötilaksi. Laitteen toinen anturi eli vesianturi on merkittynä kuvassa 10 punaisella. Veden resistanssi aiheuttaa mikrokontrollerin digitaalisen I/O-portin jännitteen siirtymisen käyttöjännitteestä lähelle maata, jolloin mikrokontrolleri voi reagoida veden havaitsemiseen lähettämällä ilmoituksen. Vaaleanvihreällä on merkittynä kolme kytkintä, joita käytetään laitteen ohjaamiseen. Kukin kytkimistä on kytketty omaan I/O-porttiinsa, jotka on kytketty maahan alasvetovastuksilla. Kun kytkintä painetaan, muuttuu I/O-portin jännite käyttöjännitteeksi ja mikrokontrolleri tunnistaa korkean jännitetason napin painallukseksi.

Oranssilla kuvassa 10 on merkitty nestekidenäyttö. Nestekidenäyttö tarvitsee toimiakseen käyttöjännitteen ja se on liitetty mikrokontrolleriin I^2C-väylän kahta johdinta käyttäen. GSM-moduuli on toinen työssä käytetty moduuli ja se on merkittynä kuvassa 10 sinisellä. GSM-moduuli tarvitsee toimiakseen kaksi käyttöjännitettä, yksi moduulille ja toinen moduulin SIM5320E-mikrokontrollerille. Kumpikin käyttöjännite on 5 V, joten ne voidaan tuoda moduulille samalla johtimella. GSM-moduuli on kytketty mikrokontrolleriin

kahden digitaalisen I/O-portin kautta, jotka mallintavat ohjelmallisesti RS-232-sarjaliikenneväylää.

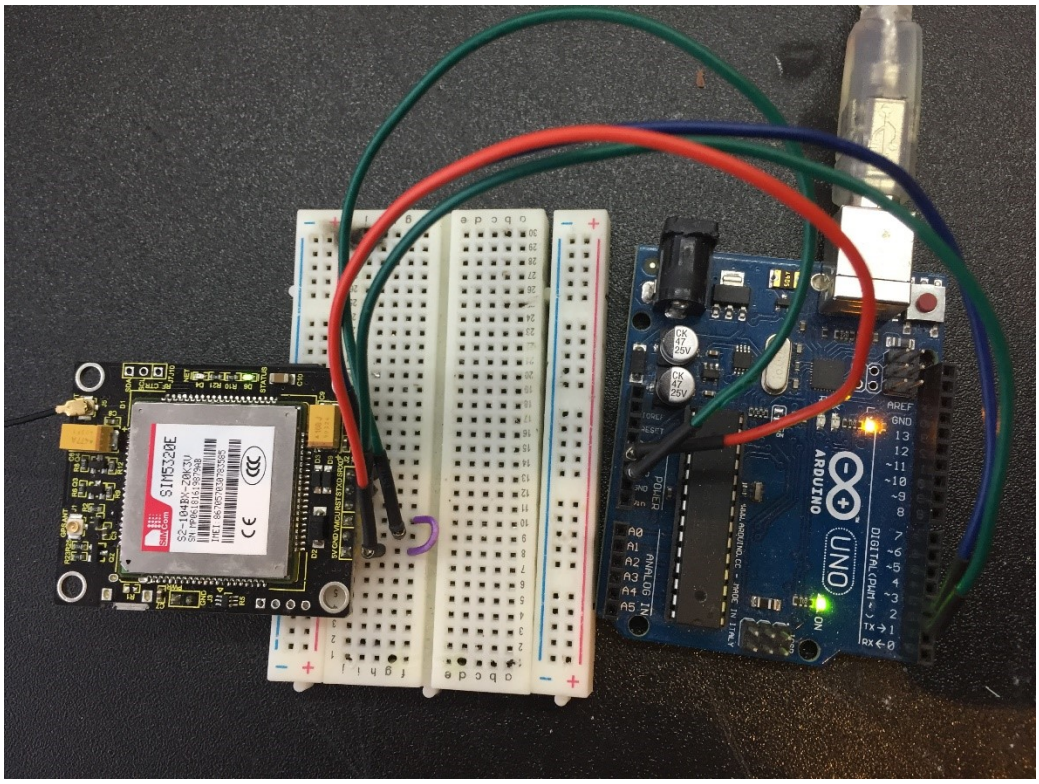
3. LAITTEEN RAKENNUSPROSESSI

Kylpytynnyrin lämmitysavustimen rakennusprosessi koostuu kolmesta vaiheesta: prototyypin rakentamisesta, piirilevyn suunnittelemisesta ja valmiin laitteen rakentamisesta. Seuraavissa luvuissa käsitellään myös käyttöliittymän ja laitteen käyttö sekä ohjelman toiminta. Lopuksi testataan, kuinka hyvin toteutettu laite toimii.

3.1 Prototyyppi

Järjestelmän prototyyppi toteutettiin käyttäen Arduino Uno -kehitysalustaa. Arduino valittiin käytettäväksi, sillä se on mahdollista ohjelmoida USB-väylää (Universal Serial Bus) käyttäen, jolloin komponenttien ja eri ohjelmaversioiden testaaminen on helppoa ja nopeaa.

Ensimmäinen vaihe oli testata ja tutustua SIM5320E-moduuliin, mikä tapahtui kytkemällä GSM-moduuli kiinni Arduinoon koekytkentälevyn ja hyppylankojen avulla. Käytetty kytkentä on esitettynä kuvassa 11. GSM-moduuli käyttää kommunikointiin UART-sarjaliikenneväylää, mutta sitä ei voi kytkeä Arduinon sarjaliikenneväylään, sillä sitä käytetään Arduinon ohjelmointiin. Moduuli kytkettiin digitaalisiin I/O-portteihin, jotka ohjelmallisesti mallintavat UART-sarjaliikenneväylää. GSM-moduulin oletusbaudinopeus on 115200 bit/s, joka on liian nopea ohjelmalliselle sarjaliikenneväylälle. GSM-moduulin baudinopeus muutettiin nopeuteen 4800 bit/s käyttäen Adafruitin FONA-kirjastoa [18].



Kuva 11. *GSM-moduulin testaaminen.*

Kun GSM-moduuli oli saatu toimimaan, seuraava vaihe oli kytkeä LCD kiinni Arduinoon ja alkaa kirjoittamaan lämmitysavustimen lopullista ohjelmaa. Ohjelmasta aluksi tehtiin sen päänäyttö, joka sisälsi vain valittavat asetukset ilman toimintoja. Kun päänäyttö toimi oikein, seuraavaksi laitteeseen lisättiin anturien lukeminen ja niiden oikea toiminta voitiin todeta päänäytöltä. Kuvassa 12 on esitettynä laitteen prototyyppi, jossa on kiinni LCD, jonka kautta ohjelman oikea toiminta voidaan todeta. Kuvasta myös huomataan, että käytetty LCD ei tue skandimerkkejä.

Kuva 12. *Ohjelman testaaminen.*

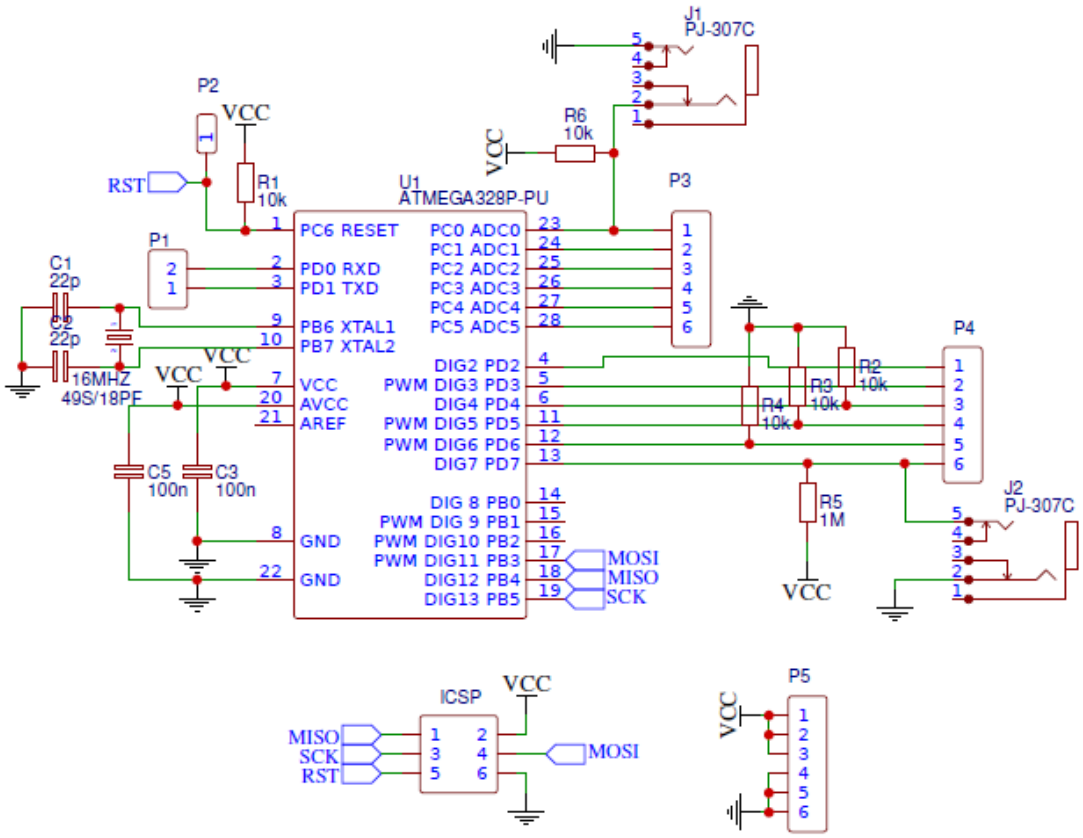
Viimeinen vaihe ohjelman kirjoittamisessa oli toteuttaa puhelinnumeronasetus ja akun varauksen lukeminen. Tämän jälkeen laitteen prototyyppi oli valmis ja ohjelma alustavasti valmis. Kun prototyyppi oli valmis ja sen oikea toiminta todettu, seuraava vaihe oli suunnitella laitteelle tarvittavat piirilevyt. Laitteen ohjelman ei tarvinnut olla vielä valmis tässä vaiheessa, sillä piirilevy suunnitellaan niin, että laitteen voi ohjelmoida myös jatkossa helposti.

3.2 Piirilevyjen suunnitteleminen ja rakentaminen

Piirilevyn suunnitteleminen toteutettiin käyttäen EasyEDA-ohjelmaa. Piirilevyn suunnitteleminen aloitettiin tekemällä kytkentäkaavio, joka sisältää kaikki tarvittavat komponentit ja niiden väliset kytkennät. Laitteeseen toteutettiin erikseen laitteen pääpiirilevy ja step-up-muuntimen piirilevy. Edellä mainitut osat toteutettiin erillisille piirilevyille, sillä nyt step-up-muuntimia voidaan käyttää muissa projekteissa ja mikäli toisessa piirilevyssä on suunnitteluvirhe, tarvitsee vain muokata toista piirilevyä.

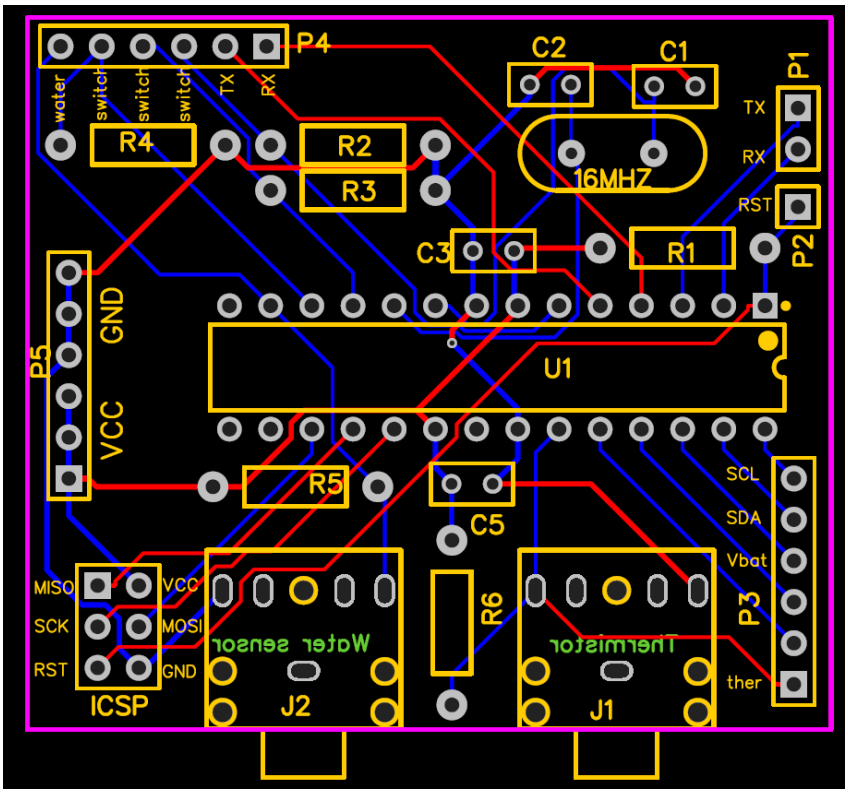
Laitteen pääpiirilevy sisältää mikrokontrollerin, mikrokontrollerin tarvitsemat kondensaattorit ja kiteen sekä liitännät piirilevyn ulkoisia komponentteja varten. Koska anturit ja kytkimet ovat piirilevyn ulkopuolisia, tulee piirilevylle vain niiden ylös- ja alasvetovastukset,

piikkirimat ja audioliitännät, joilla ulkoiset komponentit kiinnitettään piirilevylle. Lämpötila- ja vesianturin kiinnittämiseen käytetään audioliittimiä, sillä ne tarjoavat vankan ja helppokäyttöisen liitännän. Piirilevylle on lisätty myös ISP-liitäntä (In-system programming) ja piikkirimat UART-väylää varten, joten kontrollerin voi ohjelmoida sekä ohjelmointilaitteella, että käynnistyslataajaa käyttäen. Laitteen piirikaavio on esitettynä kuvassa 13.



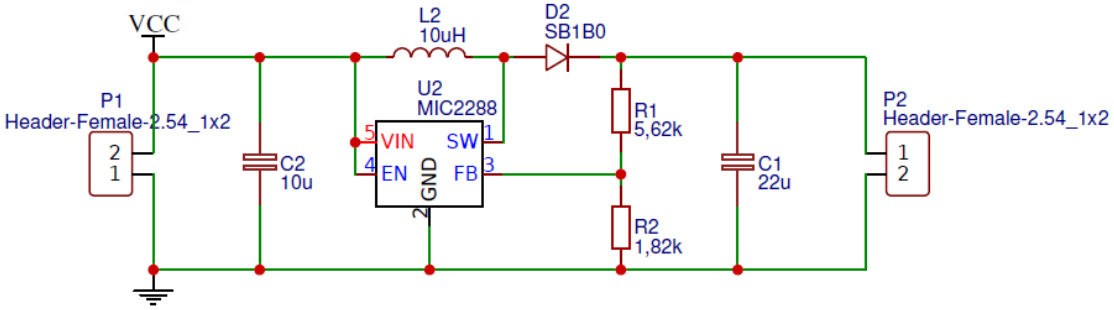
Kuva 13. *Laitteen piirikaavio.*

Seuraava vaihe oli luoda laitteen piirilevy, kuva 14, piirikaavion pohjalta. Audioliitännät asetettiin piirilevyn yhdelle reunalle, jotta ne voitaisiin tuoda laitteen kotelon läpi. Muut liitännät asetettiin piirilevyn toisille reunoille, jotta niihin pääsisi mahdollisimman helposti käsiksi piirilevyn alareunan ollessa kotelon pohjaa vasten. Mikrokontrolleri asetettiin piirilevyn keskelle, jotta siitä olisi mahdollisimman helppo vetää johtimet kaikkiin muihin komponentteihin. Lopuksi passiivikomponentit ja kide asetettiin mahdollisimman lähelle niitä paikkoja, joihin ne liitetään. Käytettyjen signaalijohtimien paksuus on 8 mil ja käyttöjännitejohtimien paksuus 12 mil.



Kuva 14. *Laitteen piirilevy.*

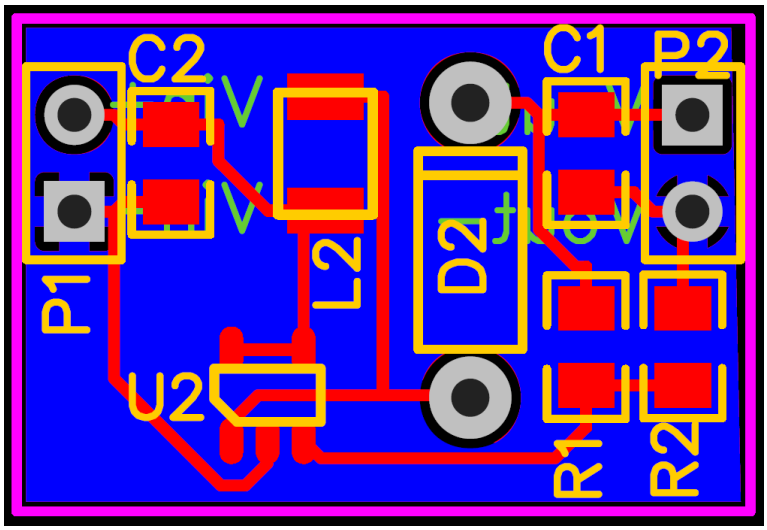
Step-up-muuntimen piirikaavio toteutettiin kuvan 9 mukaan, mutta osa komponenttien arvoista muutettiin luvun 2.8 mukaisiksi. Muuntimen ulos- ja sisääntulot on toteutettuina kuvan 15 piirikaaviossa piikkirimoina, mutta käytännössä piikkirimat jätetään asentamatta ja niiden tilalle liitetään johtimet.



Kuva 15. *Step-up-muuntimen piirikaavio.*

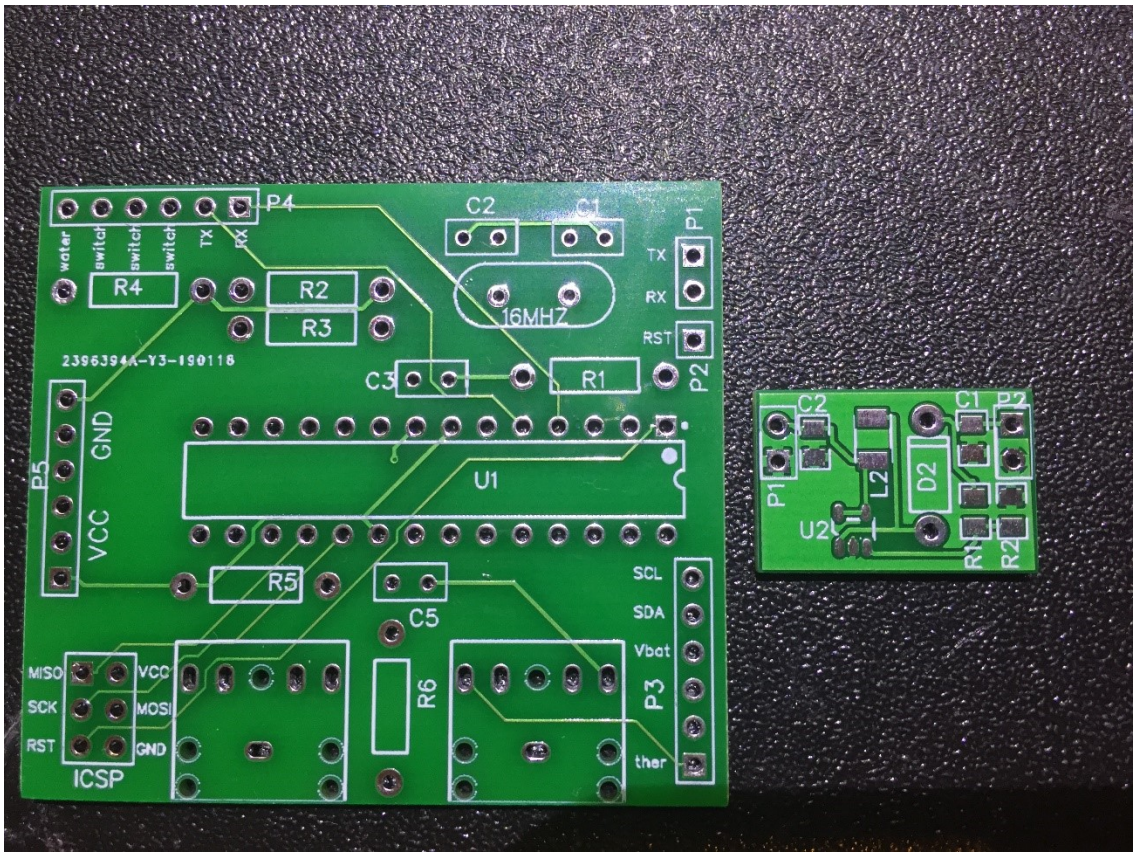
Seuraavaksi oli aika suunnitella step-up-muuntimen piirilevy. Piirilevy suunniteltiin niin, että sisään- ja ulostulot ovat vastakkaisilla puolilla piirilevyä, jotta se olisi mahdollisimman helppo asentaa ja käyttää. Piirilevy pyrittiin tekemään mahdollisimman kompaktiksi, sillä muuntimen IC:n valmistaja kehottaa pitämään johtimet mahdollisimman lyhyinä. Piirilevyn kompaktius saavutettiin käyttämällä pintaliitoskomponentteja. Muuntimen piirilevylle

lisättiin myös kummallekin puolelle maataso, jotta lämpö johtuisi IC:stä mahdollisimman hyvin pois. Maataso ei ole välttämätön kyseiselle muuntajalle, sillä sen antama teho ei ole kovin suuri, mutta maatasot lisättiin varmuuden vuoksi. Suunniteltu step-up-muuntimen piirilevy on esitettynä kuvassa 16.



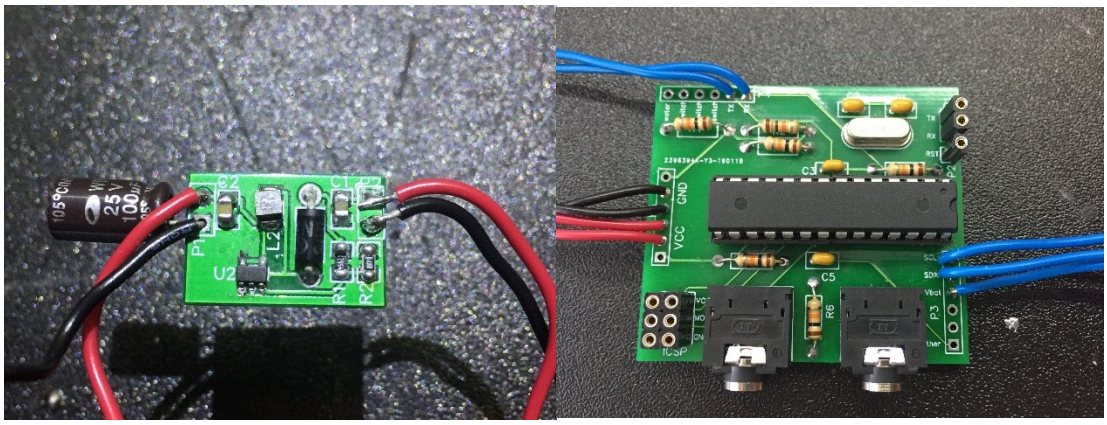
Kuva 16. *Step-up-muuntimen piirilevy.*

Suunnitellut piirilevyt tilattiin JLCPCB:ltä, joka on kiinalainen piirilevyjä valmistava yritys. Piirilevyt tilattiin itse tekemisen sijaan, jolloin niiden laatu on huomattavasti parempi. Saapuneet piirilevyt ovat esitettynä kuvassa 17 ja niiden laatu oli erittäin hyvä.



Kuva 17. *Vasemmalla laitteen pääpiirilevy ja oikealla step-up-muuntimen piirilevy.*

Viimeinen vaihe piirilevyjen valmistuksessa oli koota piirilevyt eli juottaa kiinni tarvittavat komponentit ja testata niiden toiminta. Piirilevyihin juotettiin kiinni komponentit ja johtimet oikeille paikoilleen. Johtimilla piirilevyt liitetään toisiinsa kiinni ja pääpiirilevyyn saadaan liitettyä myös muut ulkoiset komponentit ja moduulit. Step-up-muuntimen sisääntuloon lisättiin vielä 100 µF kondensaattori, joka todettiin tarpeelliseksi myöhemmin, sillä GSM-moduuli otti käynnistyessään odotettua suurempia virtapiikkejä. Valmiit piirilevyt ovat esitettynä kuvassa 18. Step-up-muuntimen toiminta testattiin syöttämällä sen sisääntuloon 3,7 V ja mittaamalla sen ulostulo. Pääpiirilevyn testaaminen aloitettiin lataamalla mikrokontrolleriin Arduino-käynnistyslataaja ISP-liittimen kautta. Käynnistyslataajan lataaminen onnistui, mikä tarkoittaa, että ainakin mikrokontrollerin kannalta kriittiset osat piirilevyä toimivat. Seuraavaksi laitteen ohjelma ladattiin käyttäen UART-väylää, minkä jälkeen piirilevyn lopullinen toiminta voitiin todeta liittämällä LCD ja anturit piirilevyyn ja toteamalla niiden toiminta LCD:n kautta.



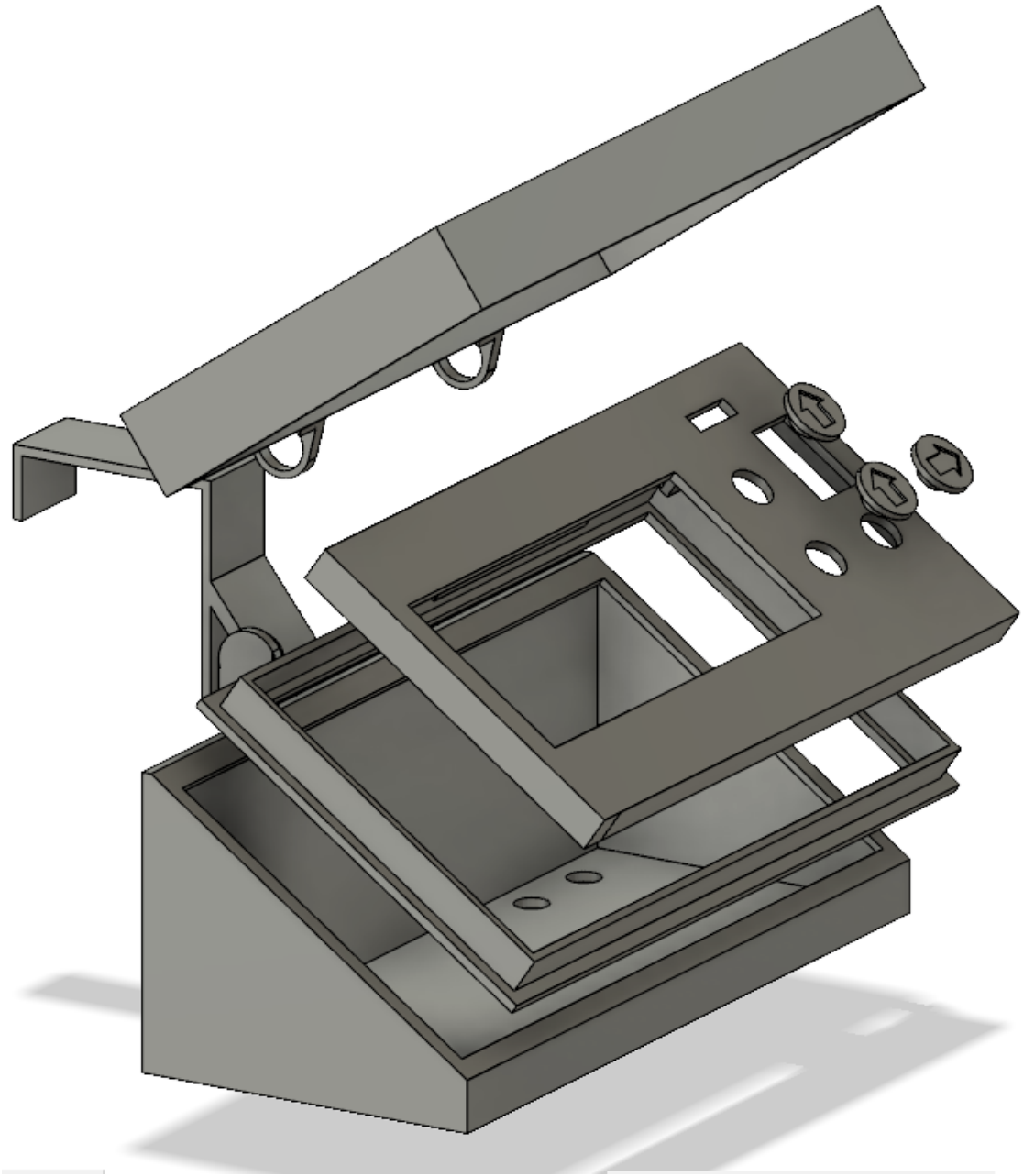
Kuva 18. *Valmiit piirilevyt komponentteineen. Vasemmalla step-up-muunnin ja oikealla laitteen pääpiirilevy.*

Kun piirilevyjen toiminta on todettu, voidaan siirtyä lämmitysavustimen rakentamisessa seuraavaan vaiheeseen eli valmiin laitteen rakentamiseen.

3.3 Valmis laite

Seuraava vaihe lämmitysavustimen rakentamisessa oli suunnitella ja rakentaa laitteelle kotelo. Kotelon valmistustavaksi valittiin 3D-tulostaminen, sillä se mahdollistaa monimutkaisten osien valmistamisen melko nopeasti ja edullisesti. Kotelo suunniteltiin käyttäen Autodesk Fusion 360 -ohjelmaa.

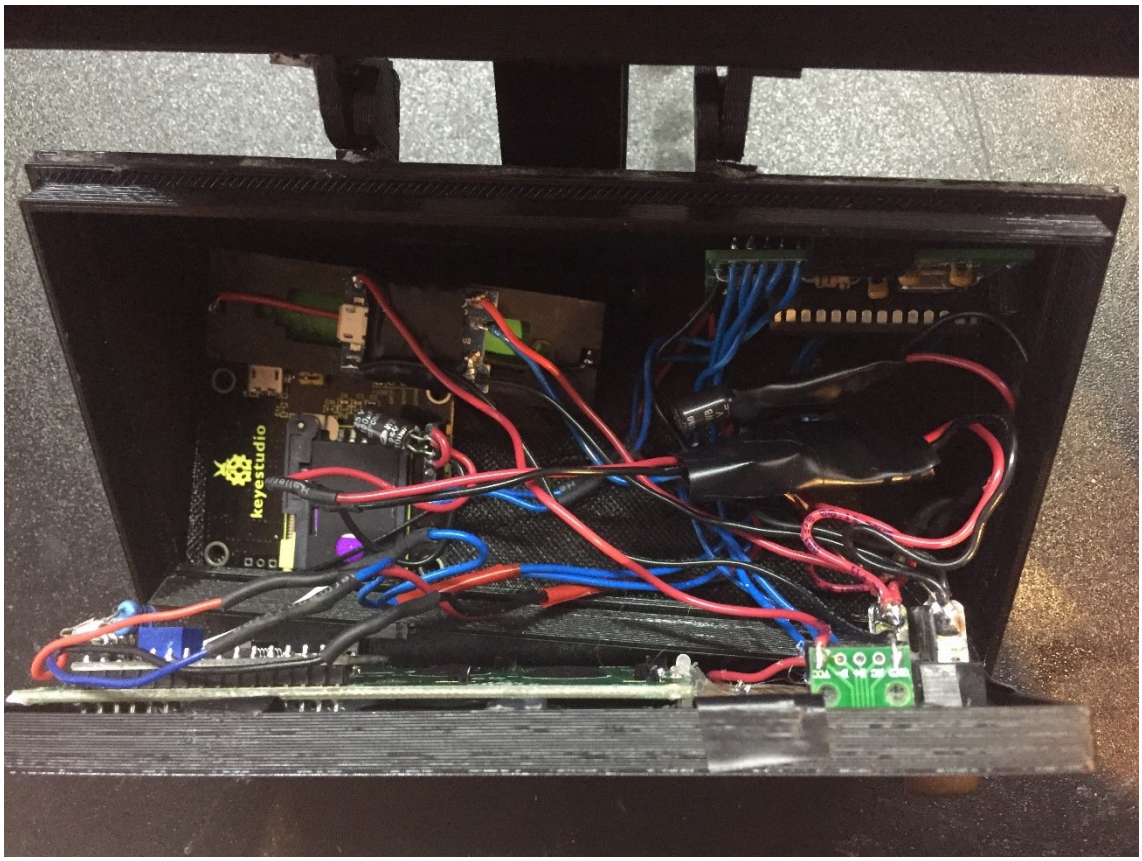
Suunniteltu kotelo koostuu 12 osasta, jotka ovat kansi, etupaneeli, etupaneelin kanta, runko, neljä saranaa, kolme nappia ja koukku. Laitteen kotelon räjäytyskuva on esitettynä kuvassa 19. Etupaneeli on suunniteltu niin, että se sisältää paikat kaikille siihen liitettäville käyttöliittymän osille. Tarvittavat osat vain liimataan kiinni etupaneeliin, joten niille ei tarvitse suunnitella erikseen kiinnityksiä. Etupaneeli liitetään etupaneelin kantaan vain painamalla, jolloin laitteen sisäosiin päästään käsiksi nostamalla etupaneeli pois. Etupaneelin kannan tehtävä on sovittaa etupaneeli runkoon ja samalla nostaa etupaneelia ylös, jotta vesi ei pääsisi kannen raoista käyttöliittymään. Etupaneelin kanta liimataan tiiviisti kiinni runkoon. Rungossa on kaksi reikää takakulmassa, joihin pääpiirilevyn audioliitännät asetetaan niin, että piirilevy tulee pystyyn takaseinämää vasten. Kotelon takana on reikä, johon voidaan kiinnittää erilaisia kiinnitysmenetelmiä. Koukku on suunniteltu niin, että se tulee kylpytynnyrin reunalle, jolloin lämmitysavustin roikkuu kylpytynnyrin ulkopuolella. Kotelon kansi tulee etupaneelin päälle ja sivuille, jolloin roiskeveden ei pitäisi päästä käyttöliittymään. Kansi on kiinnitettynä runkoon kahdella saranalla, jotka kummatkin koostuvat kahdesta osasta. Nappien pinnassa on nuolen kuva, joka kuvastaa napin toimintoa ja niiden pohjassa on kanta, jolla ne voidaan liittää kytkimiin.



Kuva 19. *Lämmitysavustimen kotelon räjäytyskuva.*

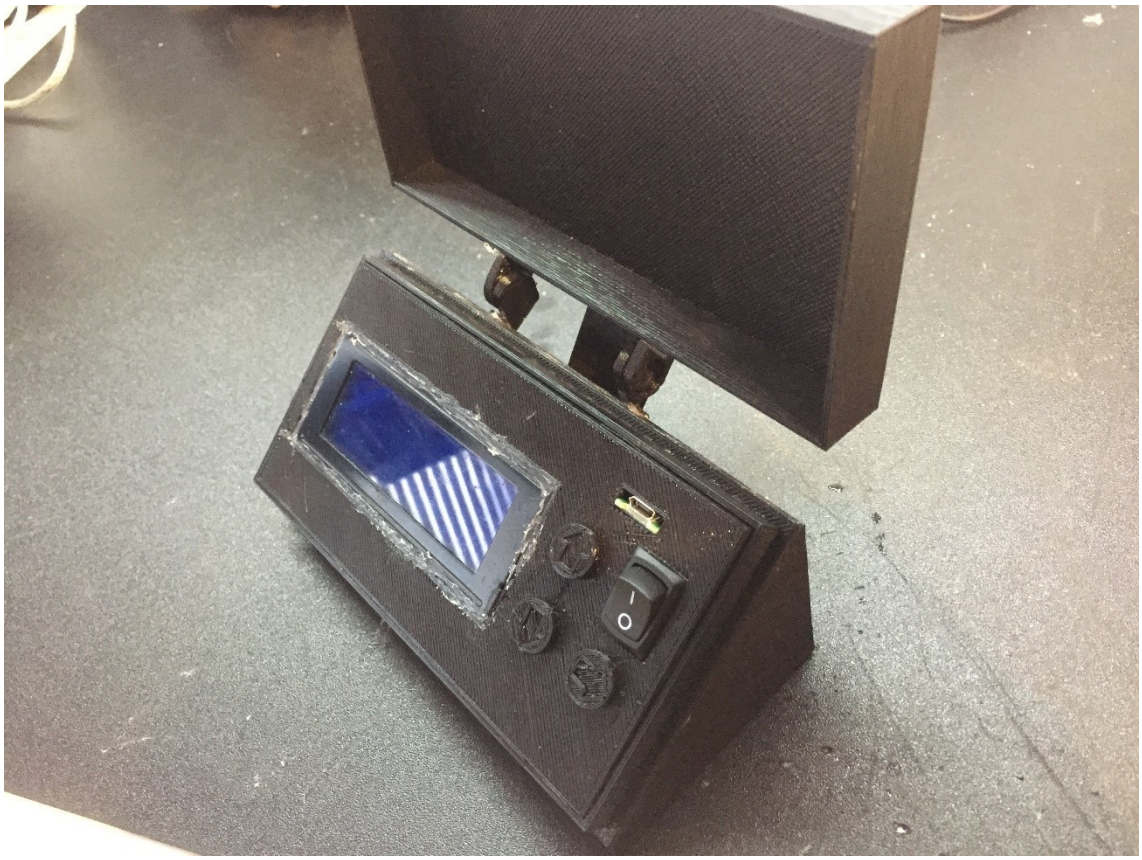
Kun laitteen kotelon osat oli 3D-tulostettu, laitteen kokoaminen aloitettiin liittämällä laitteen kanteen LCD, USB-liitäntä, virtakytkin ja yleiskäyttölevy, johon oli juotettu kiinni kolme kytkintä oikeille paikoillensa. Osien toisiinsa kiinnittämiseen käytettiin kuumaliimaa. Tämän jälkeen pääpiirilevy asennettiin paikallensa kotelon runkoon ja sen päälle liimattiin 3D-tulostettu pala, joka estää piirilevyä liikkumasta antureita liitettäessä. Rungon pohjalle myös asetettiin GSM-moduuli, step-up-muunnin ja akku suojapiireineen. Seuraavaksi kaikki osat juotettiin kiinni toisiinsa. Kun laite käynnistettiin, se toimi oikein, mutta noin 30 s päästä avustimen LCD tai koko laite saattoi jumittua. Ongelman syyksi osoittautui huonosti mitoitettu step-up-muunnin. GSM-moduuli tyypillisesti käyttää n. 200

mA virtaa, jonka mukaan muunnin oli mitoitettu, mutta hieman käynnistyksen jälkeen moduulin muodostaessa mobiiliverkkoon yhteys, moduuli ottaa jopa 1 A virtaa hetkellisesti. Virtapiikki aiheuttaa jännitteen alenemisen, joka saa laitteen jumiutumaan. Jo suunniteltua step-up-muunninta ei pystynyt enää jälkikäteen muokkaamaan tuottamaan tarpeeksi virtaa, joten päädyin lisäämään avustimeen toisen muuntimen. Nyt yksi muunnin tuottaa käyttöjännitteen GSM-moduulille ja toinen lopulle järjestelmälle. Muuntimille lisättiin myös sisäänmenoon 100 µF kondensaattorit auttamaan virtapiikkien kanssa. Kun avustimessa oli kaksi muunninta se toimi normaalisti. Kuvassa 20 on esitetty avustimen sisäosat.



Kuva 20. *Lämmitysavustimen sisäosat.*

GSM-moduulia ei kiinnitetty laitteen runkoon, jotta sen voi nostaa pois laitteesta SIM-kortin asettamista varten. Myöskään akkua tai step-up-muuntimia ei kiinnitetty runkoon, jotta akun voi vaihtaa tarvittaessa. Laitteen ollessa suljettuna komponentit puristuvat tiiviisti omille paikoillensa, eivätkä ne pääse juurikaan liikkumaan ulkoisen tärinän takia. Kohdat, joissa on oikosulkuvaara, kuten muuntimissa, on suojattu sähköteipillä. Muuntimet ovat suurimmassa oikosulkuvaarassa, sillä ne pääsevät liikkumaan kotelon sisällä johtimien varassa.

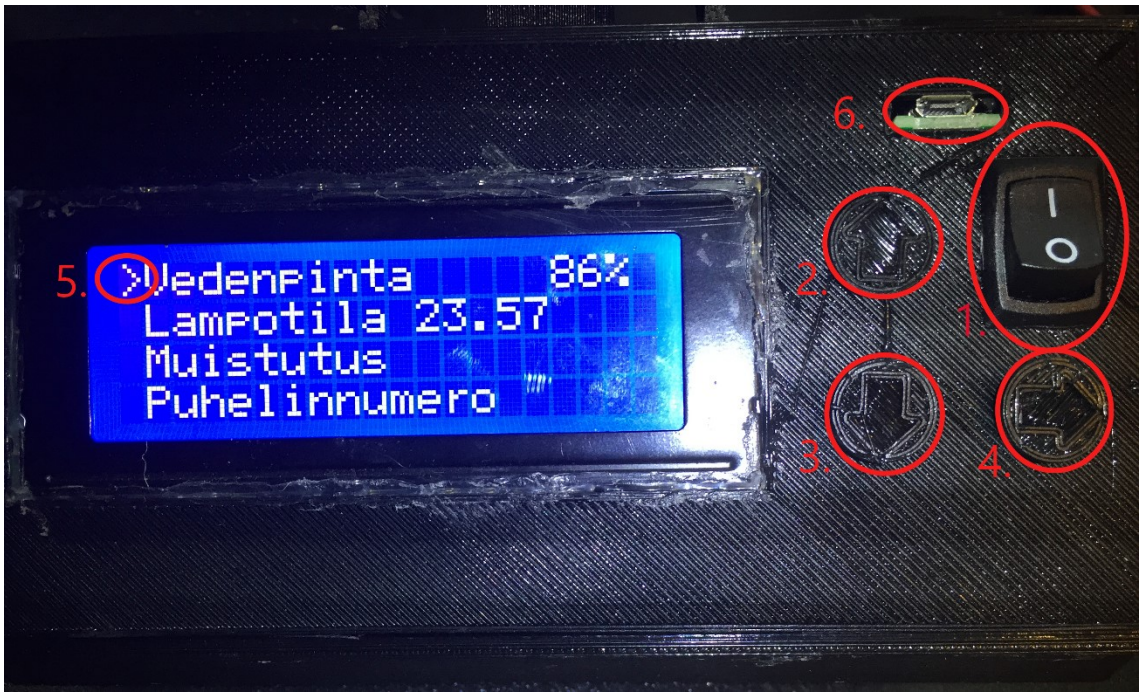


Kuva 21. *Valmis lämmitysavustin.*

Kuvassa 21 on esitettynä valmis laite. Laitteen rakentaminen vaikutti onnistuneen, joten seuraava vaihe on ottaa laite käyttöön ja selvittää mikäli se täyttää sille asetetut vaatimukset.

3.4 Käyttöliittymä ja laitteen käyttö

Lämmitysavustimen käyttöliittymä on suunniteltu mahdollisimman helppokäyttöiseksi, joten se koostuu vain yhdestä virtakytkimestä, kolmesta napista ja LCD:stä. Kuvassa 22 virtakytkin on merkitty numerolla yksi ja numerolla kuusi on merkitty USB-liitäntä, jonka kautta laitetta voi ladata. Kuvassa 22 numerolla viisi on merkitty kursori, jota ohjataan ylös ja alas liikkumiskytkimillä kaksi ja kolme. Kytkin kaksi liikuttaa kursoria ylös ja kytkin kolme alas. Kursoria käytetään valitsemaan haluttu toiminto näytöltä. Kytkin neljä on valintakytkin, jolla valitaan haluttu toiminto. Laitteen näytön oikeassa yläreunassa on myös esitettynä laitteen akun varaus prosentteina.



Kuva 22. *Lämmitysavustimen käyttöliittymä.*

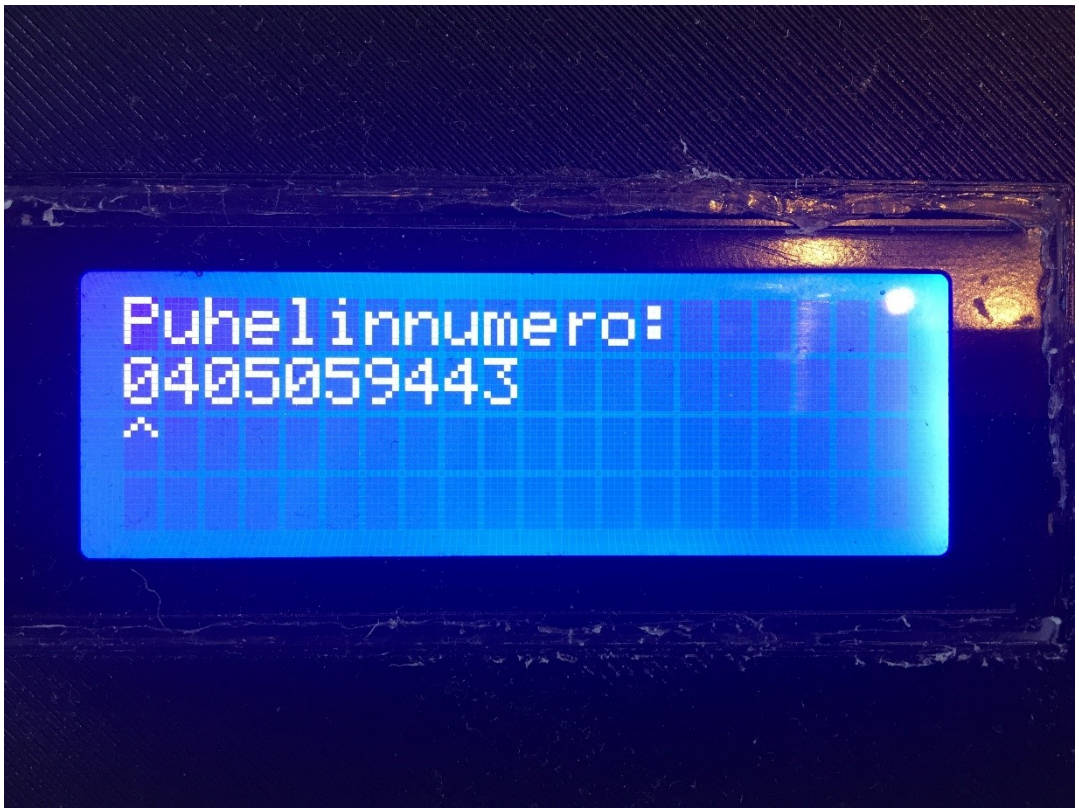
Vedenpinnan korkeuden tarkkailu otetaan käyttöön liikuttamalla kursori kohdan ”Vedenpinta” kohdalle ja painamalla valintakytkintä. Näytölle ilmestyy merkki ”<” ilmoittamaan, että vedenpinnan korkeuden tarkkailu on käytössä. Mikäli valintakytkintä painetaan uudelleen, niin vedenpinnan korkeuden tarkkailu poistuu käytöstä.

Lämpötilan tarkkailu asetetaan päälle viemällä kursori kohtaan ”Lampotila” ja painamalla valintakytkintä. Nyt laite menee tilaan, jossa haluttu kohdelämpötila voidaan asettaa painamalla alas- ja ylöskytkimiä. Laite ehdottaa kohdelämpötilaksi 36 ˚C ja se voidaan asettaa välille 20…50 ˚C. Kun haluttu lämpötila on valittu, painamalla valintakytkintä lukitaan asetettu lämpötila ja poistutaan lämpötilan asetustilasta. Lämpötilan tarkkailu saadaan pois käytöstä jälleen painamalla valintakytkintä lämpötilavalinnan kohdalla. Kun lämpötilan tarkkailu on käytössä, näytöllä näkyvät kohdelämpötila ja tämän hetkinen veden lämpötila samaan aikaan.

Muistutus puiden lisäämisestä asetetaan samalla tavalla kuin lämpötilan tarkkailu. Muistutus voidaan asettaa aikavälille 1…60 min, mutta ajan asettaminen aloitetaan aina kohdasta 20 min, jotta käyttäjä välttyisi turhilta kytkimien painalluksilta. Kun muistutus on asetettuna, jäljellä oleva aika näytetään näytöllä. Muistutusajastimen saavuttaessa arvon 0 min lähetetään tekstiviesti ja muistutusajastin otetaan pois käytöstä, joten muistutus on asetettava manuaalisesti uudelleen. Kuvassa 23 on esitettynä lämmitysavustimen näyttö, kun kaikki asetukset on otettu käyttöön.

Kuva 23. *Lämmitysavustimen käyttöliittymä, kun kaikki asetukset on otettu käyttöön.*

Lämmitysavustimesta valittaessa valinta ”Puhelinnumero”, siirtyy laite puhelinnumeron asetustilaan. Puhelinnumeron asetustila on esitettynä kuvassa 24. Puhelinnumerotilassa käyttäjälle esitetään tämänhetkinen puhelinnumero ja kursori, joka osoittaa yhteen numeroon kerrallaan. Kursorin osoittamaa numeroa voidaan vaihtaa alas- ja ylöskytkimillä ja valintakytkintä painaessa kursori siirtyy seuraavaan numeroon. Jokainen numero voidaan asettaa välille 0…9. Kun kursori on viimeisen numeron kohdalla ja valintakytkintä painetaan, asetettu puhelinnumero tallennetaan ja laite poistuu puhelinnumeron asetustilasta alkuvalikkoon. Laitteeseen voidaan asettaa vain kymmennumeroinen puhelinnumero.



Kuva 24. *Lämmitysavustin puhelinnumeron asetustilassa.*

Laitteen käyttöliittymä päivitetään viiden sekunnin välein tai aina kun jotakin kytkintä painetaan. Laite myös tarkistaa anturit ja päivittää muistutusajastimen vain käyttöliittymän päivittyessä, joten tilan näyttämisessä ja viestien lähettämisessä voi esiintyä pieniä viiveitä. Koska anturien lukeminen on sidottu päänäytön päivittämiseen, niin antureita ei lueta, kun laite on asetustilassa. Tästä syystä laite on aina jätettävä päänäyttöön, jotta se toimisi oikein.

Lämmitysavustinta käytetään asettamalla se roikkumaan kylpytynnyrin kylkeen. Tämän jälkeen laitteen pohjaan liitetään lämpötila- ja vesianturit. Lämpötila-anturin kärki asetetaan veteen ja vesianturi laitetaan roikkumaan kylpytynnyrin reunalta niin, että sen kärki on korkeudella, joka veden pinnan halutaan saavuttavan. Tämän jälkeen laitteesta valitaan toiminnot, mitä laitteen halutaan seuraavan ja laite alkaa valvomaan kylpytynnyrin tilaa.

3.5 Ohjelma

Järjestelmän ohjelma on kirjoitettu Arduino-ohjelmointiympäristössä, koska se yksinkertaistaa mikrokontrollerin ohjelmointia ja tarjoaa valmiita funktiota ja kirjastoja käytettäväksi. Ohjelma koostuu 12 funktiosta, jotka on esiteltynä taulukossa 2. Koko ohjelma on esitettynä liitteessä A.

Taulukko 2. *Ohjelman funktiot ja niiden tehtävät.*

Funktio	Funktion tehtävä
setup()	Asettaa I/O-portit ja alustaa sarjaliikenneväylät.
loop()	Ohjelman päätoistorakenne. Pyörittää ohjelman käyttöliittymää ja tarkistaa ovatko anturien arvot saavuttaneet asetetut arvot.
tulostaNaytto(String tekstit[4])	Tulostaa ohjelman päävalikon näytölle. Ottaa parametrina listan, joka sisältää tulostettavan tiedon.
lueNapit()	Lukee tiedon, onko nappeja painettu. Päivittää globaaleja muuttujia painettujen nappien mukaan. Palauttaa tiedon siitä, painettiinko yhtäkään nappia.
asetaPuhNum()	Tulostaa näytön ja lukee käyttäjän syötteitä, joiden avulla asetetaan haluttu puhelinnumero.
lahetaViesti(String viesti)	Lähettää tekstiviestin. Ottaa parametrinä lähetettävän viestin.
lueLampotila()	Lukee veden lämpötilan. Lukee lämpötilan kolme kertaa ja palauttaa niiden keskiarvon. Funktio on saatu Adafruitin termistoriohjeesta [19].

lueVedenpinta()	Lukee, onko vesianturi kosketuksissa veteen. Palauttaa true mikäli anturi koskettaa vettä.
asetaLampotila()	Asettaa kohdelämpötilan käyttäjän syötteiden mukaan. Kohdelämpötilan voi asettaa välille 20…50 ˚C.
lueAkku()	Lukee akun jännitteen ja muuntaa sen prosenteiksi. $U_{akku} \geq 4{,}1V -> 100\%$ $U_{akku} \leq 3V -> 0\%$
asetaMuistutus()	Asettaa muistutuksen halutun ajan päähän käyttäjän syötteiden mukaan. Muistutuksen voi asettaa välille 1…60min.
lueEdelliset()	Lukee aikaisemmin asetetut valinnat EEPROM-muistista ja ottaa ne käyttöön.

Ohjelmassa kaikki asetukset tallennetaan EEPROM-muistiin ja luetaan käynnistyksen yhteydessä. Tämä mahdollistaa sen, että mikäli laite sammuu tai resetoituu Watchdog-ajastimen toimesta, kykenee ohjelma jatkamaan toimintaa edellisestä pisteestä käynnistyksen yhteydessä.

3.6 Laitteen testaaminen

Kylpytynnyrin lämmitysavustin testattiin käytännössä lämmittämällä kylpytynnyri ja tarkkailemalla lämmitysavustimen toimintaa. Kuvassa 25 on lämmitysavustin käytössä. Vesianturi tunnisti veden oikein sen noustessa anturin korkeudelle ja lämpötilamittari mittasi veden lämpötilaa oikein. Lämpötilan mittauksessa on kuitenkin otettava huomioon anturin syvyys, sillä pintavesi lämpiää huomattavasti nopeammin kuin vesi kylpytynnyrin pohjalla. Tekstiviestit lämmitysavustin lähetti oikein ilman minkäänlaisia ongelmia. Laitteen testaaminen tapahtui n. 3 ˚C lämpötilassa, jolloin LCD:n merkkien päivitysnopeus oli hidastunut selvästi, mutta se ei aiheuta ongelmia laitteen oikealle toiminnalle. Laitteen mitattu tyypillinen virrankäyttö oli n. 220 mA, jolloin laite kestää n. 9,5 h 2100 mAh akulla, joten laiteen akunkesto täyttää hyvin sille asetetun viiden tunnin vaatimuksen.

Kuva 25. *Lämmitysavustin käytössä.*

Laitteen roiskeenkestävyyttä testattiin kaatamalla laitteen päälle vettä. Laite oli pääasiassa roiskeenkestävä, mutta kannen läpi pääsi hieman tihkumaan vettä näytölle. Tämä johtuu siitä, että 3D-tulostetun kannen rakenne on huokoinen, jolloin hieman vettä pääsee kulkeutumaan sen läpi. Ongelma saatiin kuitenkin korjattua tulostamalla kansi uudelleen 0,2 mm kerroskorkeudella 0,4 mm sijaan, jolloin kannen rakenne on tiiviimpi eikä vesi pääse enää kulkeutumaan sen lävitse.

Laitteen käyttöliittymän helppoutta testattiin antamalla laite henkilölle, joka ei ole ennen käyttänyt laitetta. Testihenkilö oppi käyttämään laitetta nopeasti minimaalisella opastuksella, joten voidaan todeta laitteen olevan helppokäyttöinen. Laitteen käyttäminen oli helppoa myös hanskat kädessä, sillä laitteessa on suuret napit.

Edellä tehtyjen testien perusteella voidaan todeta, että rakennettu lämmitysavustin täyttää kaikki sille luvun kaksi alussa asetetut vaatimukset, joten laitteen suunnitteleminen ja rakentaminen olivat onnistuneita.

4. YHTEENVETO

Tämän työn tarkoituksena oli suunnitella ja rakentaa laite avustamaan kylpytynnyrin lämmityksessä. Työn tuloksena syntyi kylpytynnyrin lämmitysavustin, joka avustaa mittaamalla kylpytynnyrin tilaa ja välittämällä tietoa sekä muistutuksia tekstiviesteinä lämmittäjälle. Laite täyttää kaikki sille luvussa kaksi asetetut vaatimukset, joten laitteen toteutus voidaan katsoa onnistuneeksi.

Taulukossa 3 on esitetty kaikki käytetyt komponentit ja niiden hinnat, joista voidaan laskea koko laitteen kustannukset. Laitteen hinnaksi tuli loppujen lopuksi 94,99 €, joka on hieman korkea. Laitteen suurin yksittäinen kustannus oli käytetty GSM-moduuli. Vaihtamalla GSM-moduuli halvempaan, joka sisältää vähemmän ei tarpeellisia ominaisuuksia, voidaan säästää jo paljon laitteen kustannuksissa. Laitteen kustannuksissa ei ole otettu huomioon 3D-tulostetun kotelon hintaa.

Taulukko 3. *Lämmitysavustimessa käytetyt komponentit ja niiden hinnat.*

Komponentti	Määrä (kpl)	Hinta (€/kpl)
GSM-moduuli	1	42,99
Ternistori	1	5,75
LCD	1	18,4
Kytkin	3	0,15
Virtakytkin	1	2,06
Li-ion akku	1	5,99
Atmega328p-mikrokontrolleri	1	1,67
3,5 mm audioliitäntä	2	0,64
16 MHz kide	1	0,47
100 nF kondensaattori	2	0,09
22 pF kondensaattori	2	0,08
10 kΩ vastus	5	0,007
1 MΩ vastus	1	0,03
1,5 m audiojohdin	1	3,65
MIC2288	2	0,49
10 µH kela	2	0,92
10 µF kondensaattori	2	0,29
22 µF kondensaattori	2	0,85
5,62 kΩ vastus	1	0,03
1,82 kΩ vastus	1	0,06
1N5819 diodi	2	0,26
BMS	1	0,82
Pääpiirilevy	1	0,35
Step-up-muuntimen piirilevy	2	0,89
Kokonaishinta		94,99

Lämmitysavustinta voitaisiin vielä kehittää integroimalla kaikki laitteen osat yhdelle piirilevylle, jolloin laitteen kokoa saataisiin huomattavasti pienennettyä ja sisäosia siistittyä. Mikäli lämmitysprosessia haluttaisiin automatisoida, voitaisiin laitteeseen lisätä portteja, joiden kautta voitaisiin ohjata ulkoisia moduuleja, kuten venttiilejä. Myös laitteen step-up-

muunninta voisi vielä kehittää antamaan lisää virtaa, jolloin laitteeseen riittäisi vain yksi muunnin.

LÄHTEET

[1] D.V. Gadre, Programming and customizing the AVR microcontroller, McGraw-Hill Education, 2000, 336 p.

[2] ATmega328p-mikrokontrollerin datalehti, 2018, 662 p. Saatavilla (Viitattu 5.3.2019): http://ww1.microchip.com/downloads/en/DeviceDoc/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061A.pdf

[3] RS232 and Serial Communications, TAL Technologies, 2019. Saatavilla (Viitattu 5.3.2019): http://www.taltech.com/datacollection/articles/serial_intro

[4] Serial Communication, SparkFun Electronics, Saatavilla (Viitattu 5.3.2019): https://learn.sparkfun.com/tutorials/serial-communication/all#serial-intro

[5] J. Patrick, Serial Protocols Compared, 2002. Saatavilla (Viitattu 5.3.2019): https://www.embedded.com/design/connectivity/4023975/Serial-Protocols-Compared

[6] D.K. Yang, S.T. Wu, Fundamentals of Liquid Crystal Devices. New York: John Wiley & Sons, Incorporated; 2014, 565 p.

[7] N. Agnihotri, AT Commands, GSM AT command set, Saatavilla (Viitattu 7.4.2019): https://www.engineersgarage.com/tutorials/at-commands

[8] SIM5320 AT Command Set, 2012, 498 p. Saatavilla (Viitattu 7.4.2019): http://www.mt-system.ru/sites/default/files/simcom_sim5320_atc_en_v1.23.pdf

[9] D. Ibrahim, Microcontroller-Based Temperature Monitoring and Control, Newnes, 2002, 235 p.

[10] P. Scherz; S. Monk, Practical Electronics for Inventors, Fourth Edition. Temperature, McGraw-Hill Professional, 2016, AccessEngineering

[11] NTC termistorin datalehti, 2013, 9 p. Saatavilla (Viitattu 2.3.2019): http://www.farnell.com/datasheets/2015777.pdf?_ga=2.80747031.1272710398.1551193595-590799133.1547120500&_gac=1.18233675.1547724464.Cj0KCQiA7IDiBRCLARIsABIPohge-mazXyRgv4ZQ2G2kVoInGXF77igucL1dC4oEaeHdS9sjT8CfrSfkaAvlUEALw_wcB

[12] M.K. Kazimierczuk, Pulse-Width Modulated DC-DC Power Converters. New York: John Wiley & Sons, Incorporated, 2015, 1457 p.

[13] MIC2288-muuntimen datalehti, 2018, 24 p. Saatavilla (Viitattu 7.4.2019): http://ww1.microchip.com/downloads/en/DeviceDoc/MIC2288-1A-1-point-2MHz-PWM-Boost-Converter-DS20006034B.pdf

[14] How do Lithium Batteries Work?, Battery University, 2018, Saatavilla (Viitattu 7.4.2019): https://batteryuniversity.com/learn/article/lithium_based_batteries

[15] G.L. Plett, Battery Management Systems, Volume 1 - Battery Modeling, Artech House, 2015, 325 p.

[16] A.D. Battery Management Systems for Large Lithium Ion Battery Packs, Norwood: Artech House, 2010, 271 p.

[17] Litiumioniakun datalehti, 2012, 9 p. Saatavilla (Viitattu 7.4.2019): https://www.tme.eu/fi/Document/115c0c91a21fa2415574c0d861498cd6/SONY_US18650VTC4.pdf

[18] Adafruit FONA -kirjasto. Saatavilla (Viitattu 7.4.2019): https://github.com/adafruit/Adafruit_FONA

[19] Using a Thermistor, Adafruit, 2012, Saatavilla (Viitattu 7.4.2019): https://learn.adafruit.com/thermistor/using-a-thermistor

LIITE A: OHJELMA

```
#include <Wire.h>
#include <LCD.h>
#include <LiquidCrystal_I2C.h>
#include <EEPROM.h>
#include <SoftwareSerial.h>
#include <avr/wdt.h>

#define I2C_ADDR    0x27
#define BACKLIGHT_PIN  3
#define En_pin  2
#define Rw_pin  1
#define Rs_pin  0
#define D4_pin  4
#define D5_pin  5
#define D6_pin  6
#define D7_pin  7

#define vesianturi 7
#define SERIESRESISTOR 10000
#define BCOEFFICIENT 3988
#define THERMISTORNOMINAL 10000
#define TEMPERATURENOMINAL 25
#define NUMSAMPLES 3

LiquidCrystal_I2C  lcd(I2C_ADDR, En_pin, Rw_pin, Rs_pin, D4_pin,
D5_pin, D6_pin, D7_pin);
SoftwareSerial SIM(2, 3);

String menu[4] = {"Vedenpinta", "Lampotila", "Muistutus", "Puhelinnu-
mero"};

bool valikkoValinnat[4] = {false, false, false, false};
bool valintaPainettu = false;
bool varoitus1 = false;
bool varoitus2 = false;

unsigned long viimeisinVesiHavainto = 0;
unsigned long ajastin = 0;
unsigned long paivitysAjastin = 0;

int lampotila = 0;

byte puhelinNum[10] = {0, 4, 0, 5, 0, 5, 8, 1, 9, 1};
byte muistutusAika;
byte kursori = 0;

void setup() {
 delay(1000);
  //Asetetaan watchdog resetoimaan mikrokontrolleri, mikäli watchdog
ajastinta ei resetoida
  wdt_enable(WDTO_500MS);
  //käynnistetään ohjelmallinen sarjaportti
  SIM.begin(4800);
  //Alustetaan LCD-näyttö
  lcd.begin (20, 4);
  lcd.setBacklightPin(BACKLIGHT_PIN, POSITIVE);
  lcd.setBacklight(HIGH);
```

```
  lcd.home();

  //Asetetaan sisään- ja ulostulot
  pinMode(vesianturi, INPUT);
  pinMode(A0, INPUT);
  pinMode(A3, INPUT);
  for (int i = 4; i < 7; i++) {
    pinMode(i, OUTPUT);
  }

  //Luetaan puhelinnumero EEPROM muisista
  for (int i = 0; i < 10; i++) {
    puhelinNum[i] = EEPROM.read(i);
  }
  //Luetaan edelliset asetukset
  lueEdelliset();
  tulostaNaytto(menu);
}



//Ohjelman pääloopi
void loop() {
  //resetoidaan watchdog ajastin
  wdt_reset();
  //Luetaan kytkimet
  if (lueNapit()) {

    //varmistetaan, että kursorin arvo ei ole liian suuri tai pieni
    if (kursori == 4) {
      kursori = 3;
    }
    else if (kursori == 255) {
      kursori = 0;
    }

    if (valintaPainettu) {
      valintaPainettu = false;

      //Asetetaan haluttu lämpötila
      if (valikkoValinnat[1] == false && kursori == 1) {
        asetaLampotila();
        kursori = 1;
      }

      //Asetetaan haluttu muistutusaika
      if (valikkoValinnat[2] == false && kursori == 2) {
        asetaMuistutus();
        ajastin = millis();
        kursori = 2;
      }

      valikkoValinnat[kursori] = !valikkoValinnat[kursori];

      //Tallennetaan EEPROM muistiin valittu arvo, mikäli se muuttui
      if(valikkoValinnat[0] == true){
        if(EEPROM.read(10) != 1){
          EEPROM.write(10,1);
        }
      }
      else if(valikkoValinnat[0] == false){
```

```
        if(EEPROM.read(10) != 0){
          EEPROM.write(10,0);
        }
      }
      if(valikkoValinnat[kursori] == false && EEPROM.read(10+kursori)
!= 0 && kursori > 0){
        EEPROM.write(10+kursori,0);
      }
    }
    tulostaNaytto(menu);
  }

  //Asetetaan puhelinnumero
  if (valikkoValinnat[3]) {
    valintaPainettu = false;
    asetaPuhNum();
    kursori = 3;
    valikkoValinnat[3] = false;
    tulostaNaytto(menu);
  }

  //Vähennetään ajastinta minuutin välein
  if (valikkoValinnat[2] == true && millis() - ajastin >= 60000) {
    muistutusAika--;
    EEPROM.write(12,muistutusAika);
    ajastin = millis();
    tulostaNaytto(menu);
  }

  //Tarkastaa onko haluttu vedenpinnan korkeus saavutettu
  if (lueVedenpinta() == true && valikkoValinnat[0] == true) {

    unsigned long aikaero = millis() - viimeisinVesiHavainto;
    //Vedenpinnan on oltava halutulla tasolla 3 sekuntia
    if (aikaero >= 3000 && aikaero <= 3100) {
      valikkoValinnat[0] = false;
      tulostaNaytto(menu);
      lahetaViesti("Vedenpinta on saavuttanut halutun korkeuden");
      EEPROM.write(10,0);
    }
    else if (aikaero >= 3100) {
      viimeisinVesiHavainto = millis();
    }
  }

  //Tarkastetaan onko haluttu lämpötila saavutettu
  else if (valikkoValinnat[1] == true && lueLampotila() > lampotila) {
    valikkoValinnat[1] = false;
    lahetaViesti("Vesi on saavuttanut lampotilan " + String(lampotila)
+ "C");
    EEPROM.write(11,0);
    tulostaNaytto(menu);
  }

  //Tarkastetaan onko asetettu muistutusaika loppunut
  else if (valikkoValinnat[2] == true && muistutusAika == 0) {
    valikkoValinnat[2] = false;
    lahetaViesti("Muistathan lisata puita");
    tulostaNaytto(menu);
  }
```

```
  //Päivitetään näyttö 5 sekunnin välein
  if (millis() - paivitysAjastin >= 5000) {
    paivitysAjastin = millis();
    tulostaNaytto(menu);
  }

}


//Funktio tulostaa päänäytön
void tulostaNaytto(String tekstit[4]) {
  lcd.clear();
  lueAkku();
  for (int i = 0; i < 4; i++) {
    if (kursori == i) {
      lcd.setCursor(0, i);
      lcd.print(">");
    }

    lcd.setCursor(1, i);
    lcd.print(tekstit[i]);

    if (i == 1) {
      lcd.print(" ");
      float lampotila = lueLampotila();
      if (lampotila == -273.15) {
        lcd.print("       ");
      }
      else{
        lcd.print(lampotila);
      }
    }

    if (valikkoValinnat[i] == true) {
      if (i == 1) {
        lcd.print("/");
        lcd.print(lampotila);
      }
      else if ( i == 2) {
        lcd.print(" ");
        lcd.print(muistutusAika);
        lcd.print("min");
      }
      else {
        lcd.print(" <");
      }
    }
  }
}


// Luetaan napit
bool lueNapit() {
  wdt_reset();
  if (digitalRead(5) == HIGH) {
    delay(30);
    while (digitalRead(5) == HIGH) {
      wdt_reset();
    }
    kursori++;
    return true;
```

```
  }
  else if (digitalRead(4) == HIGH) {
    delay(30);
    while (digitalRead(4) == HIGH) {
      wdt_reset();
    }
    kursori--;
    return true;
  }
  else if (digitalRead(6) == HIGH) {
    delay(30);
    while (digitalRead(6) == HIGH) {
      wdt_reset();
    }
    valintaPainettu = true;
    return true;
  }
  return false;
}


//Funktion avulla asetetaan puhelinnumero
void asetaPuhNum() {
  lcd.clear();
  byte kursori2 = 0;
  while (true) {
    wdt_reset();
    kursori = puhelinNum[kursori2];
    if (lueNapit()) {
      if (kursori == 10) {
        kursori = 9;
      }
      else if (kursori == 255) {
        kursori = 0;
      }
      puhelinNum[kursori2] = kursori;
    }

    if (valintaPainettu) {
      valintaPainettu = false;
      kursori2++;
      if (kursori2 == 10) {
        break;
      }
      lcd.clear();
    }
    lcd.setCursor(0, 0);
    lcd.print("Puhelinnumero:");
    lcd.setCursor(0, 1);
    for (int i = 0; i < 10; i++) {
      lcd.print(puhelinNum[i]);
    }
    lcd.setCursor(kursori2, 2);
    lcd.print("^");
  }
  //Talletetaan asetettu puhelinnumero EEPROM muistiin
  for (int i = 0; i < 10; i++) {
    if (EEPROM.read(i) != puhelinNum[i]) {
      EEPROM.write(i, puhelinNum[i]);
    }
  }
```

```
  lcd.clear();
}

//Lähettää parametrinä annetun viestin tekstiviestinä
void lahetaViesti(String viesti) {
  wdt_reset();
  String puhelinNumString;
  String komento = "AT+CMGS=";

  for (int i = 0; i < 10; i++) {
    puhelinNumString += String(puhelinNum[i]);
  }
  komento += '"';
  komento += puhelinNumString;
  komento += '"';

  delay(25);
  SIM.println("AT+CMGF=1");
  delay(25);
  SIM.println(komento);
  delay(25);
  SIM.print(viesti);
  delay(25);
  SIM.write(0x1A);
  wdt_reset();
}


//Lukee termistorin lmpötilan kolme kertaa ja palauttaa niiden
//keskiarvon. Saatu Adafruitin termistoriohjeesta, Saatavilla:
//https://learn.adafruit.com/thermistor/using-a-thermistor
float lueLampotila() {
  wdt_reset();
  uint8_t i;
  float average;
  uint16_t samples[NUMSAMPLES];

  // take N samples in a row, with a slight delay
  for (i = 0; i < NUMSAMPLES; i++) {
    samples[i] = analogRead(A0);
    delay(10);
  }
  // average all the samples out
  average = 0;
  for (i = 0; i < NUMSAMPLES; i++) {
    average += samples[i];
  }
  average /= NUMSAMPLES;

  // convert the value to resistance
  average = 1023 / average - 1;
  average = SERIESRESISTOR / average;

  float steinhart;
  steinhart = average / THERMISTORNOMINAL;     // (R/Ro)
  steinhart = log(steinhart);                  // ln(R/Ro)
  steinhart /= BCOEFFICIENT;                   // 1/B * ln(R/Ro)
  steinhart += 1.0 / (TEMPERATURENOMINAL + 273.15); // + (1/To)
  steinhart = 1.0 / steinhart;                 // Invert
  steinhart -= 273.15;                         // convert to C
```

```
  return steinhart;
}


//Tarkistaa onko vedenpinta saavuttanut sensorin
bool lueVedenpinta() {
  if (digitalRead(vesianturi) == HIGH) {
    return false;
  }
  else {
    return true;
  }
}


//Funktio asettaa halutun kohdelämpötilan
void asetaLampotila() {
  wdt_reset();
  lampotila = 36;
  lcd.setCursor(17, 2);
  lcd.print("^^");
  lcd.setCursor(16, 1);
  lcd.print("/");
  lcd.print(lampotila);
  kursori = lampotila;
  while (true) {
    wdt_reset();
    if (lueNapit()) {

      if (kursori < 20) {
        kursori = 20;
      }
      else if ( kursori > 50) {
        kursori = 50;
      }

      lcd.setCursor(17, 1);
      lcd.print("  ");
      lcd.setCursor(17, 1);
      lcd.print(kursori);
    }
    if (valintaPainettu) {
      valintaPainettu = false;
      lampotila = kursori;
      break;
    }

  }
  //talletetaan lämpötila EEPROM muistiinabcdefghijklmnopqrstuvwzyx
  EEPROM.write(11, byte(lampotila));
}


// Lukee akun jännitteen ja muuntaa sen prosenteiksi
// >= 4,1V -> 100%
// <= 3V -> 0%
void lueAkku() {
  wdt_reset();
  lcd.setCursor(16, 0);
  int lukema = analogRead(A3);
  double jannite = lukema * (5.0 / 1023.0);
```

```
  if (jannite >= 4.1) {
    jannite = 4.1;
  }
  else if ( jannite <= 3) {
    jannite = 3;
  }

  int prosentti = (jannite - 3) * (100 - 0) / (4.1 - 3) + 0;

  if(prosentti <= 15 && not varoitus2){
    lahetaViesti("Akun varaus alle 15%");
    varoitus2 = true;
  }
  else if(prosentti <= 30 && not varoitus1){
    lahetaViesti("Akun varaus alle 30%");
    varoitus1 = true;

  }

  lcd.print(prosentti);
  lcd.print("%");
}


//Funktio asettaa halutun muistutusajan
void asetaMuistutus() {
  wdt_reset();
  muistutusAika = 20;
  lcd.setCursor(11, 2);
  lcd.print(muistutusAika);
  lcd.print("min");
  kursori = muistutusAika;
  while (true) {
    wdt_reset();
    if (lueNapit()) {

      if (kursori < 1) {
        kursori = 1;
      }
      else if (kursori > 60) {
        kursori = 60;
      }

      lcd.setCursor(11, 2);
      lcd.print("  ");
      lcd.setCursor(11, 2);
      lcd.print(kursori);

    }
    if (valintaPainettu) {
      valintaPainettu = false;
      muistutusAika = kursori;
      break;
    }
  }
  //talletetaan asetettu aika EEPROM muistiin
  EEPROM.write(12, byte(muistutusAika));
}
```

```
//Lukee muistista edelliset valinnat ja ottaa ne käyttöön, mikäli ne
olivat käytössä aikaisemmin.
void lueEdelliset(){
  for(int i=10; i < 13; i++){
     if(EEPROM.read(i) != 0){
        valikkoValinnat[i-10] = true;
        if(i == 11){
          lampotila = EEPROM.read(i);
        }
        else if(i == 12){
          muistutusAika = EEPROM.read(i);
        }
     }
  }
}
```