Antti Niiranen

PÖLYMITTARIN TOTEUTUS ARDUINO-ALUSTALLA

Opinnäytetyö
CENTRIA-AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma
Maaliskuu 2019

TIIVISTELMÄ OPINNÄYTETYÖSTÄ

Centria-ammattikorkeakoulu	**Aika** Maaliskuu 2019	**Tekijä/tekijät** Antti Niiranen
Koulutusohjelma Tieto- ja viestintätekniikan		
Työn nimi PÖLYMITTARIN TOTEUTUS ARDUINO-ALUSTALLA		
Työn ohjaaja Sakari Männistö		**Sivumäärä** 26 + 6
Työelämäohjaaja Jari Isohanni		

Opinnäytetyössä suunnitellaan siirrettävä pölymittari, joka lähettää mittaustiedot Sigfox-verkkoon. Laitteistona käytetään MKRFOX1200 Arduino-alustaa ja Sharpin valmistamaa pölyanturia, jonka toimintaan myös tutustutaan teoriassa ja käytännössä.
Kun testaukset on tehty ja ohjelman toiminta suunniteltu, niin anturia ja muita tarpeellisia laitteiston osia varten myös suunnitellaan sopivat kytkennät, joilla saadaan niille sopivat käyttöjännitteet ja virrankatkaisu silloin kun niitä ei tarvita. Laitetta varten on tehty mittausohjelma, joka mittaa pölyä, laittaa laitteiston lepotilaan ja myös herättää sen lepotilasta ja lähettää tulokset eteenpäin sopivassa muodossa, ja kalibrointiohjelma, jota tarvitaan laitteen käyttöönotossa. Laitteesta valmistetaan prototyyppi ja tätä varten suunnitellaan piirilevy ja 3D-tulostettava kotelo.

Asiasanat
Arduino, GP21010AU0F, MKRFOX1200, Pöly, Pölymittari, Sigfox,

ABSTRACT

Centria University of Applied Sciences	Date	Author
	March 2019	Antti Niiranen
Degree programme Information- and communication technology		
Name of thesis Arduino based dust sensor		
Instructor Sakari Männistö		**Pages** 26 + 6
Supervisor Jari Isohanni		

In this thesis a moveable dust meter that sends the measurement data to the Sigfox network was designed. Hardware used was MKRFOX1200 Arduino board and dust sensor made by Sharp, which was examined in theory and also in practice. When tests were done and function of the program was planned circuits for sensor and other equipment were also planned, so operating voltages were suitable, and they can be shut down when they are not needed. The program for the device was made, which puts the system to sleep and also wakes it up and sends measurements forward in correct format, and also the calibration program that is required in the setup of the device. Prototype was made and because of that circuit board was designed as well as 3D printable enclosure.

Key words
Arduino, GP21010AU0F, Dust, Dust sensor, MKRFOX1200, Sigfox

KÄSITTEIDEN MÄÄRITTELY

LED	Light emitting diode	Valodiodi
GND	Ground	Maataso
VCC	Voltage common connector	Käyttöjänniteliityntä
USB	Universal serial bus	
VIN	Input Voltage	Käyttöjännite
RX	Receive	Vastaanotto
TX	Transmit	Lähetys

1 JOHDANTO

Opinnäytetyön tarkoituksena oli suunnitella toteutettavaksi helposti siirrettävissä oleva langaton pölymittari, joka lähettäisi mittauspaikan pölypitoisuudet eteenpäin Sigfox-verkkoon. Laitteisto, jolla mittari toteutettiin, oli Arduino MKRFOX1200 sekä Sharp GP21010AU0F toimi pölyanturina.

Aluksi tutustutaan, kuinka laitteiston osia käytetään ja myöskin tehdään testimittauksia, joilla saadaan jotain tietoa laitteiston toiminnasta. Tämän jälkeen tutkitaan, miten laitteisto toimii erilaisilla käyttöjännitteillä ja valitaan tähän käyttöön parhaiten soveltuva vaihtoehto ja suunnitellaan tätä tarkoitusta varten sopivat kytkennät, jossa on myös mahdollisuus ohjelmalliseen virrankatkaisuun niiltä laitteiston osilta, jotka eivät ole tarpeellisia koko ajan. Ohjelmaa on kehitetty laitteiston ohessa sitä mukaa, kun laitteiston muutokset ovat vaatineet sitä. Samoin on tutustuttu lopullisessa laitteessa ja sen halutussa käyttötilanteessa tarpeellisiin ominaisuuksiin, kuten Sigfox-lähetykseen ja laitteiston nukkumiseen.

Kun laitteiston ohjelma ja kytkennät on suunniteltu, aletaan suunnitella ja valmistaa prototyyppiä suunnitelmien perusteella. Ensimmäisenä suunnitellaan piirilevy koekytkentäalustalla suunniteltujen kytkentöjen pohjalta, ja tämän jälkeen suunnitellaan laitteistoa ja piirilevyä varten 3D-ulostettava kotelo. Suunnittelun jälkeen kerrotaan, minkälainen prosessi prototyypin käyttöönotto on. Johtopäätöksissä käydään läpi suunnittelun eri vaiheissa ilmentyneitä ongelmia ja asioita, joiden pohjalta on tehty johtopäätöksiä ja ratkaisuja, joiden jälkeen on edetty seuraavaan vaiheeseen.

2 LAITTEISTO

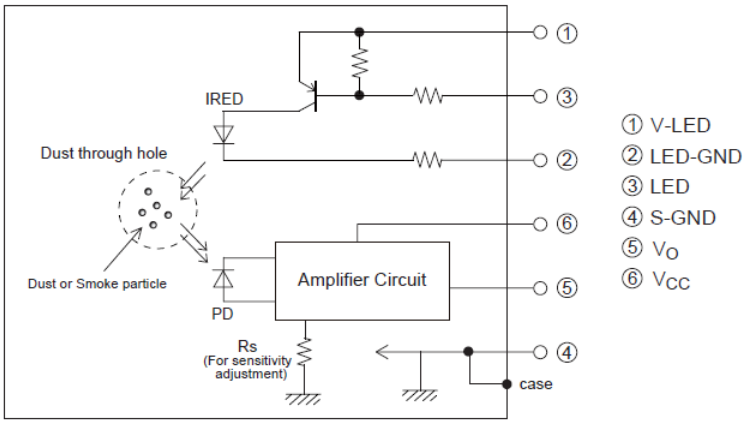
Laitteisto koostuu MKRFOX1200 Arduino-alustasta, jossa on mukana Sigfox-moduuli, joka voi lähettää ja vastaanottaa viestejä Sigfox-verkossa, sekä Sharp GP21010AU0F-pölyanturista, jota käytetään pölyn mittaukseen. MKRFOX1200 alustaa käytetään pölyanturin ohjaukseen ja tulosten lukuun ja lähetykseen. Lisäksi tarvittiin erilaisia elektroniikan komponentteja anturin toimintaa varten sekä myöhemmin myös muihin tarkoituksiin.

2.1 Pölyanturi

Pölyanturi on laite, jonka avulla voidaan mitata ilmassa olevan pölyn määrää. Anturi ei toimi yksinään, vaan se tarvitsee ympärilleen laitteistoa ohjaukseen ja mittaustuloksen lukemiseen ja esittämiseen. Erilaiset pölyanturit voivat toimia toisistaan poikkeavalla tavalla, joten seuraavissa luvuissa selvitetään tässä työssä käytetyn pölyanturin rakennetta, miten pölyanturi kytketään ja myöskin, miten mittaustapahtuman pitäisi tapahtua.

2.1.1 Pölyanturin rakenne

Pölyanturina käytetään Sharp:in valmistamaa GP21010AU0F-pölyanturia, joka koostuu laatikosta, jossa on kaksi reikää. Reikien läpi ilma voi virrata laatikon sisään ja ulos. Laatikon sisällä on infrapuna LED ja phototransistori sellaisessa kulmassa, että ledin lähettämä valo kimpoaa pölyhiukkasista ja jonka phototransistori rekisteröi. Lisäksi anturissa on muuta toiminnan kannalta tarpeellista elektroniikkaa, kuten vahvistinpiiri ja herkkyyden säätöön käytettävä säätövastus. (Sharp 2006, 2)



KUVA 1. Pölyanturin rakenne (mukaillen Sharp, 2006)

2.1.2 Pölyanturin kytkentä

Anturiin menee kuusi johtoa ja johtojen tarkoitukset ovat seuraavat:

1. V-LED (Käyttöjännite ledille)
2. LED-GND (ledin maa)
3. LED (Ledin ohjaus)
4. S-GND (Anturin maa)
5. VO (Anturin mittaustulos jännitteenä)
6. VCC (Anturin käyttöjännite)

Ledin ja anturin maat kytketään maahan. Käyttöjännitteenä otetaan Arduino-alustasta. Lediin ei kuitenkaan kytketä käyttöjännitettä suoraan, vaan väliin 150Ω kytketään vastus ja 220µF kondensaattori estämään jännitteen tippumista lyhyen ohjauspulssin aikana. Valmistaja suosittelee käyttöjännitteeksi 5V. Ledin ohjaus kytketään ohjattavaan lähtöön ja anturin mittaustulosta varten kytketään anturin lähtöjännite analogiseen tuloon jännitteen mittaamista varten. (Sharp, 2006, 5)

2.1.3 Pölyanturilla mittaaminen

Pölyn mittaus tapahtuu sytyttämällä ledi ja hetken kuluttua voidaan mitata jännite, joka kertoo pölypitoisuuden. Valmistajan suositusten mukaan mittaukseen käytetään 0,32ms ja mittauksia voidaan tehdä 10ms välein. Mittaustapahtuma etenee tällöin seuraavasti:

1. LED laitetaan päälle
2. odotetaan 0,28 millisekuntia että anturi saa pölyn määrän mitattua
3. luetaan jännite, jolloin saadaan tietoon anturin mittaama pölypitoisuus
4. sammutetaan ledi 0,04 ms päästä, jolloin koko tapahtuma on kestänyt 0,32 millisekuntia

(Sharp 2006, 5)

2.2 Arduino

Arduinot ovat avoimen lähdekoodin kehitysalustoja, joita voidaan käyttää erilaisissa elektroniikan projekteissa. Niissä on mikrokontrolleri, joka sisältää suorittimen, Ram-muistia ja ohjelmoitavaa Flash-muistia. Omalla sovellusohjelmalla voidaan kontrolloida ja lukea mikrokontrollerin sisään- ja ulostulonastoja ja myöskin kommunikoida muiden laitteiden kanssa laitteesta löytyvien porttien avulla. Mikrokontrollerin Arduino-piirilevyltä löytyy tarvittavat komponentit, jotka mahdollistavat laitteen toiminnan ja myöskin virta ja ohjelmointi liitäntöjä. Joissain malleissa voi olla lisäksi erilaisia sensoreita ja kommunikointiyhteyksiä. Arduinita ohjelmoidaan yleensä Arduinon omalla ohjelmankehitysympäristöllä. (Borchers 2015, 3.; Arduino 2018)

2.2.1 MKRFOX1200

MKRFOX1200 on Arduino-kehitysalusta, joka koostuu SAMD21-mikrokontrollerista ja ATA8520 Sigfox moduulista, joka mahdollistaa kommunikoinnin Sigfox-verkkoon. Laitetta voi käyttää kahdella 1,5V patterilla, akulla taikka ulkoisella 5V virtalähteellä, joko suoraan tai usb-liitännän kautta. Yleisimmistä Arduino-laitteista poiketen MKRFOX1200:n käyttöjännite on 3,3V, joten sisääntulopinneihin ei kannata tuoda yli 3,3V jännitteitä, ettei laite vaurioidu. (Arduino 2018)

2.3 Sigfox

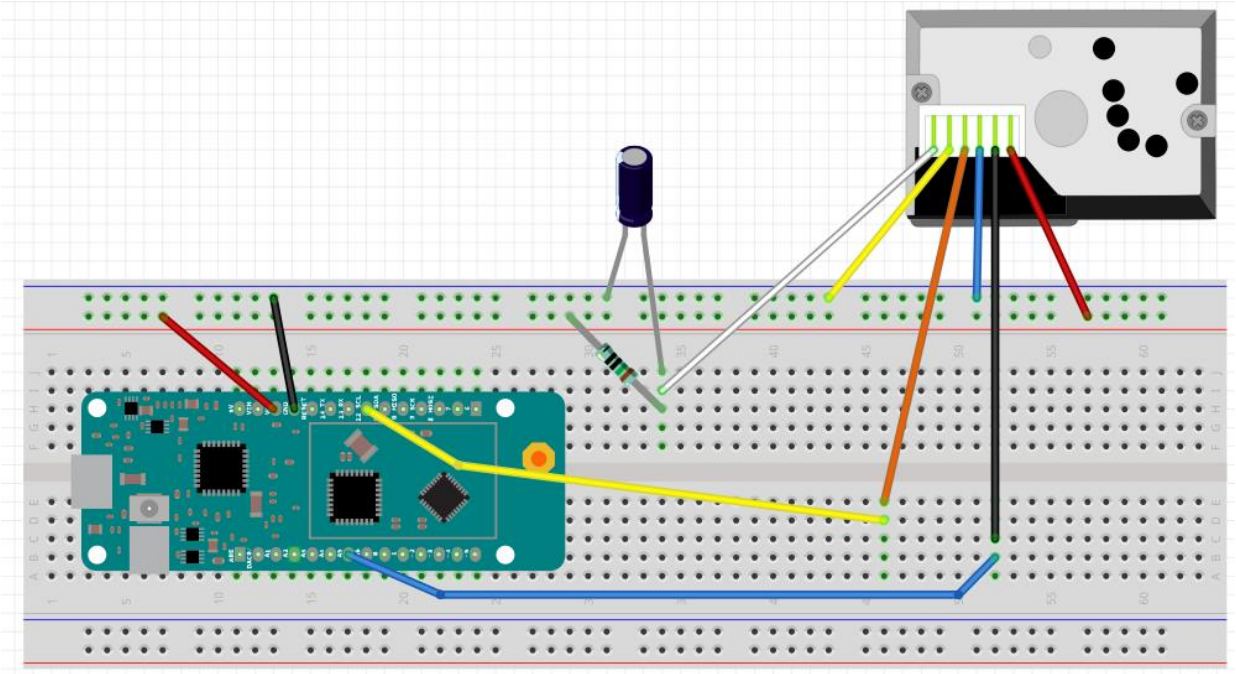
Sigfox on IoT-laitteita varten kehitetty langaton verkko, jossa pyritään edulliseen hintaan ja pieneen virrankulutukseen. Sigfox-protokolla on ohjelmisto, jonka avulla laite luo radioviesti modeemin lähetettäväksi. Ohjelmisto on rojaltivapaa ja modeemivalmistajien ilmaiseksi saatavilla, jotta syntyisi kilpailua ja Sigfoxia hyödyntävien tuotteiden hinnat tippuvat. Laite ei ole kytkettynä johonkin tiettyyn tukiasemaan, vaan sen lähettämä viesti vastaanotetaan kaikilla Sigfox-tukiasemilla, jotka ovat kantaman sisäpuolella. Tämän jälkeen viesti lähetetään Sigfox-pilvipalveluun, josta sen voi käydä lukemassa. Viestejä voidaan lähettää vuorokaudessa vain 140, ja ne voivat maksimissaan olla 12 tavua pitkiä. Laitteisiin voidaan myös lähettää viestejä, mutta ne voivat olla maksimissaan vain 8 tavua pitkiä ja tietoturvan ja virrankulutuksen vuoksi yhteys on ensin avattava laitteesta käsin. Laitteesta maksetaan vuosimaksua ja yhdellä vuosimaksulla saadaan käyttöön kaikkien operaattorien Sigfox-verkko eri maissa. (Connected Finland 2018; Isohanni 2017; Sigfox 2018)

3 KOEJÄRJESTELYT

Pölyanturin toimintaa selvitettiin tekemällä koemittauksia koulun neuvotteluhuoneessa ja rakennustyömaalla. Kytkentä ja ohjelma olivat mahdollisimman yksinkertaisia ja tulokset luettiin kannettavan tietokoneen näytöltä, johon laite oli yhdistetty USB-väylän kautta. Tulokset lähetettiin sarjaportin kautta laitteen mittaamana jännitearvona. Selvitettiin myös, miten tuuletin vaikuttaa mitattuihin arvoihin ja minkälaisia mittaustuloksia anturi oikein antaa.

3.1 Kytkennät

Testausta varten pölyanturi ja MKRFOX1200 kytkettiin koekytkentäalustalle kuvassa 3. esitetyin kytkennöin



KUVA 2. Havainnekuva pölyanturin kytkennöistä (mukaillen Arduino 2018; igorfonseca83 2016)

3.2 Ohjelmiston koodi

Testausta ja mittausta varten kirjoitettiin seuraavalla sivulla oleva koodi, joka ohjelmointiin MKRFOX1200:een. Koodi ottaa 10 näytettä vaaditulla tavalla ja tulostaa näytteistä minimi-, maksimi- ja keskiarvon ja tulostaa ne sarjaportin kautta ulos, joten mittaustulosta voi tarkkailla tietokoneella.

```
int ledOhj = 12;    //ledin ohjauspinni
int antMit = 5;     //anturin mittauspinni

int esko = 0;       //hassunniminen muuttuja
int samples = 10;   //kuinka monta näytettä otetaan

int mitMit = 0;     //mittaus muuttuja
int mitMin = 1024;  //minimimittaus muuttuja
int mitMax = 0;     //maksimimittaus muuttuja
int mitAvg = 0;     //mittauksen keskiarvo muuttuja

float mitVolt = (3.3 / 1024); //muuttaa mittauksen tuloksen volteiksi

void setup() {
Serial.begin(9600);
pinMode(ledOhj, OUTPUT);
digitalWrite(ledOhj, HIGH);
}

void loop() {
  mitMin = 1024;    //3 ekaa riviä alustaa muuttujat
  mitMax = 0;
  mitAvg = 0;

  for(esko = 0;esko<samples;esko++) //otetaan mittaustuloksia
  {
    digitalWrite(ledOhj,LOW); //ledi päälle
    delayMicroseconds(280);   //vaadittu tauko

    mitMit = analogRead(antMit);        //mitataan tulos
    delayMicroseconds(40);    //vaadittu tauko

    digitalWrite(ledOhj,HIGH);//sammutetaan ledi

    mitAvg += mitMit;         //lisätään tulos keskiarvon laskua varten
    if(mitMin>mitMit) mitMin = mitMit;  //minimin tarkistus
    if(mitMax<mitMit) mitMax = mitMit;  //maksimin tarkistus

    delay(10);  //vaadittu tauko
  }

  mitAvg /= samples;          //keskiarvon laskenta

  Serial.print("Min: ");      //tästä eteenpäin mittaustulosten
  Serial.print(mitMin*mitVolt);// tulostusta sarjaportin kautta
  Serial.print(" Avg: ");
  Serial.print(mitAvg*mitVolt);
  Serial.print(" Max: ");
  Serial.println(mitMax*mitVolt);
```

3.3 Mittaus

Koemittauksissa laitettiin pölyanturiin kiinni suppilo, johon oli yhdistetty kannettavan tietokoneen tuuletin. Pyöriessään tuuletin imi ilmaa pölyanturin läpi, jolloin ilma pölyanturin sisällä vaihtui nopeasti ja eikä tarvinnut odotella pitkään, että tulos vaihtui kun siirryttiin mittauskohteesta toiseen. Mittauksissa käytettiin edellisellä sivulla ollutta koodia ja tulokset luettiin Arduino-alustan sarjaporttimonitorilla, joka piirsi kuvaajan numeroarvoista. Kuvaajassa näkyy 10:n näytteen maksimiarvo, minimiarvo ja keskiarvo.

Mittauksissa saatiin seuraavissa luvuissa 3.3.1−3.3.4 kuvatut tulokset.

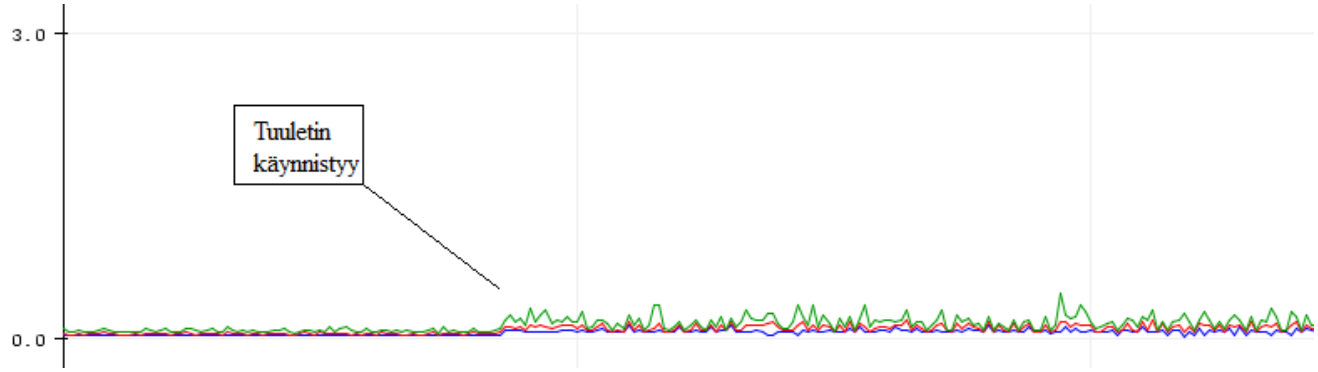
3.3.1 Koe 1. Anturi ilman tuuletinta

Mittaus suoritettiin neuvotteluhuoneessa, jossa on kokolattiamatto. Anturi sijoitettiin pöydälle toinen reikä pöytälevyä vasten eikä tuuletin pyörinyt. Ilma pölyanturin sisällä pysyi vakiona eikä liikkunut merkittävästi. Anturin mittaamat pölymäärät olivat hyvin pieniä ja vaihtelu minimi- ja maksimiarvon välillä ei ole kovinkaan suurta.

KUVA 3. Visualisoidut mittaustulokset ilman tuuletinta

3.3.2 Koe 2. Anturi tuulettimen kanssa

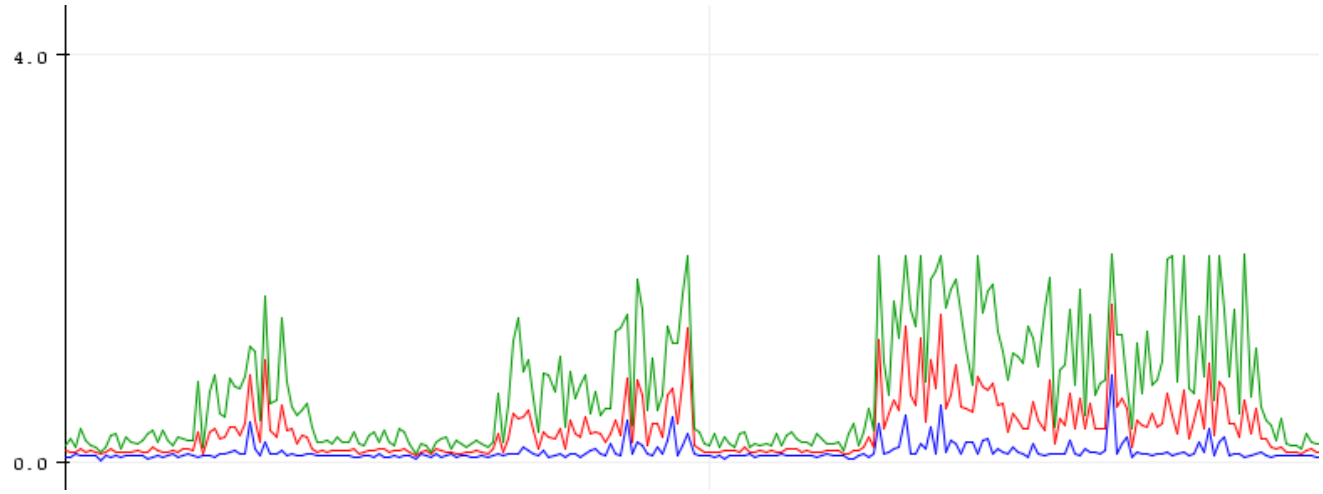
Mittaus suoritettiin samassa tilassa kuin edellinenkin mittaus. Anturia pidettiin ilmassa puolen metrin korkeudella molemmat reiät avoinna, ja tuuletin pyöri ja ilma virtasi anturin läpi. Tuulettimen käynnistys vaikutti selvästi mittaustuloksiin ja havaittavissa on selvä piikki tuulettimen käynnistyksestä. Tulokset ovat selvästi suuremmat kuin ilman tuuletinta, ja vaihtelua minimi- ja maksimiarvon välillä on myös enemmän.



KUVA 4. Visualisoidut mittaustulokset tuulettimen kanssa

3.3.3 Koe 4. Kokolattiamatto

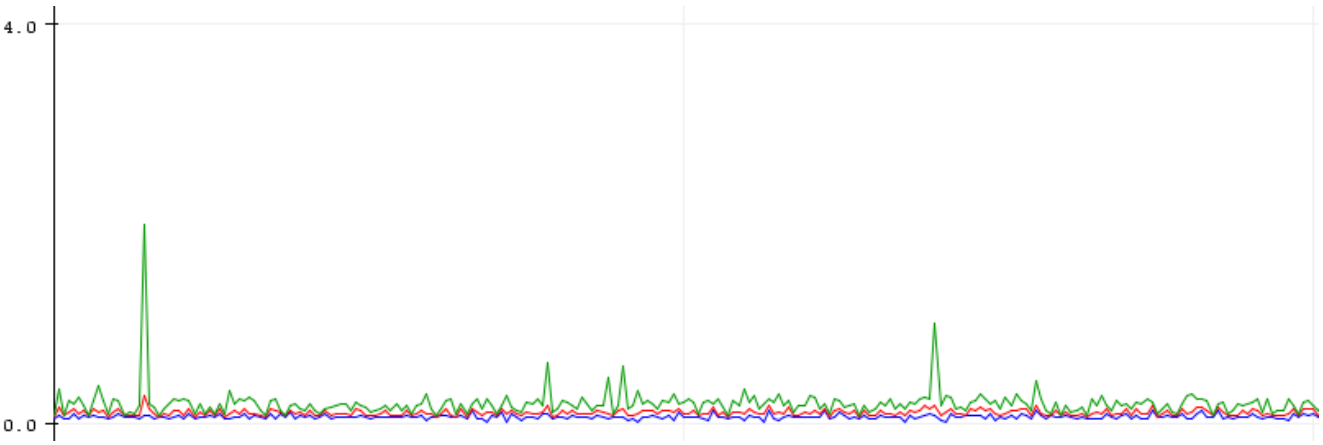
Mittaus suoritettiin samassa tilassa kuin edellinenkin mittaus. Anturista otettiin kiinni ja sitä hierottiin neuvottelutilan lattialla olevaan kokolattiamattoon tuulettimen pyöriessä. Tuloksista näkee selvästi, että mattoa hieromalla irtoaa paljon pölyä ja mittari näyttäisi menevän välillä maksimiarvoonsa, koska suurimman mitatun arvon piikit pysähtyvät usein samalle korkeudelle. Vaihtelu minimi- ja maksimiarvon välillä on myös todella suurta.



KUVA 5. Visualisoidut mittaustulokset kokolattiamatosta

3.3.4 Koe 4. Rakennustyömaa

Mittaus suoritettiin sisätiloissa loppuvaiheessa olevalla rakennustyömaalla, jossa oli betonilattia. Anturia pideltiin ilmassa tuulettimen ollessa päällä ja lattialla tömisteltiin anturin läheisyydessä siinä toivossa, että lattiasta irtoaisi pölyä, joka mahdollisesti näkyisi anturissa. Tuloksissa on havaittavissa yksittäisiä piikkejä ja tulokset näyttäisivät olevan hieman korkeammat kuin samalla tavalla mitatut tulokset toimistotilassa.



KUVA 6. Visualisoidut mittaustulokset rakennustyömaalta

4 ANTURIN TESTAUSTA JA KYTKENTÖJÄ

Päätettiin testata, millainen vaikutus pölyanturin käyttöjännitteellä on mittaustuloksiin, koska pölymittari on tarkoitus toteuttaa MKRFOX1200:lla, jonka käyttöjännite on 3,3V ja pölyanturi on taas suunniteltu toimimaan 5 V:n käyttöjännitteellä. Testauksen toteuttamiseen käytettiin Arduino unoa, joka toimii 5 V:n käyttöjännitteellä, jolloin siihen voidaan kytkeä pölyanturi ja käyttää siinä suunniteltua käyttöjännitettä ilman että riskinä on laitteen hajoaminen. Lisäksi siitä löytyy myös 3,3 V jännitteen ulostulo, jota voi käyttää antamaan pienempi käyttöjännite testejä varten. Kytkentää on hieman muokattu, että anturissa olevan ledin ja itse anturin käyttöjännitteet ovat helposti vaihdettavissa ja toisista erillään. Koodi taas on pysynyt lähes entisen kaltaisena. Suoritettiin pikainen testi, jossa selvitettiin käyttöjännitteen vaikutusta pienimpään ja suurimpaan helposti anturista saatavaan arvoon. Anturin pienin arvo on saavutettu, kun anturin sisällä ei ole mitään ja suurin arvo taas silloin kun anturin läpi on työnnetty jokin esine, esimerkiksi kynä.

Mittauksista voi havaita, että ledin käyttöjännitteen pienentäminen pienentää minimiarvoa ja anturin käyttöjännitteen pienentäminen maksimiarvoa. Mittarin datalehdessä on pölypitoisuuden vaikutus jännitteeseen ilmoitettu vain 5V käyttöjännitteellä ja maksimiarvo näyttäisi pitävän paikkansa, mutta minimiarvo on hiukan pienempi kuin datalehdessä ilmoitettu arvo. Alemman jännitteen käytöstä ei ole minkäänlaista tietoa datalehdessä, joten päätettiin tutkia hieman tarkemmin, minkälaisia vaikutuksia alemman jännitteen käytöllä on mittaustuloksiin ääripäiden välissä. Nopeat mittauksien tulokset on esitetty taulukossa 1.

Anturi V	Led V	Min	Max
5	5	0,35	3,75
5	3,3	0,2	3,75
3,3	5	0,35	2
3,3	3,3	0,2	2

TAULUKKO 1. Nopeat jännitemittaustulokset

4.1 Tarkempien mittausten suunnittelu ja toteutus

Koska mittauksilla saatiin tietoa vain ääripäistä, testattiin olisiko mahdollista saada jonkinlaisia tuloksia myös ääripäiden väliltä. Koska pölyä on vaikea kontrolloida, testattiin, onko erilaisilla esineillä vaikutusta. Mittauksissa käytössä ollut kynä oli lähes saman paksuinen kuin anturin reikäkin, joten kokeiltiin ohuempia esineitä kuten kolmea eripaksuista ruuvimeisseliä ja kuivaa spagettia.

Paksuudella ei ollut vaikutusta arvoihin, jos esine oli työnnetty kokonaan anturin läpi. Päätettiin työntää ohutta ruuvimeisseliä hyvin varovasti ja vähitellen yhä syvemmälle reikään ja huomattiin, että arvo alkoi hiljalleen nousta ja nyt oli menetelmä, jolla oli mahdollista saada arvoja minimin ja maksimin väliltä. Seuraavaksi mietittiin, miten olisi mahdollista pitää esine paikallaan, ettei reiässä oleva esine pääsisi vahingossa liikkumaan ja vaikuttamaan mitattaviin arvoihin. Erilaisista käytettävissä olevista vaihtoehdoista päädyttiin styroxpalaan anturin päällä, jota pidettiin paikallaan kiertämällä anturin ja palan ympärille alumiinifoliota. Palaan tehtiin reikä spagetilla, jota käytettiin myös mittauksissa. Mittaus suoritettiin työntämällä spagettia erittäin varovasti syvemmälle, kunnes arvossa huomattiin muutoksia, jonka jälkeen otettiin keskiarvot ylös eri jännitteillä ja sitten taas oli spagetin liikuttelun aika. Huomattiin, että spagetin ”vasarointi” erittäin varovasti jäykästä paperista tehdyllä käyntikortilla oli hyvä tapa liikutella spagettia kohti suurempaa arvoa ja jos meni vahingossa liian pitkälle, joten spagetti piti vetää takaisin ja koittaa uudelleen.

4.2 Tarkempien mittaustulosten tarkastelu

Mitatuissa tuloksissa merkittiin vain keskiarvot, mutta kiinnitettiin huomiota myös maksimiarvoihin. Koska huomattiin, että kun anturia käyttää alemmalla jännitteellä niin ulostulon jännite ei kohoa yli 2 voltin ja joissain tuloksissa keskiarvon lisäksi maksimiarvo on sama, koska alempi käyttöjännite estää arvon nousemisen korkeammalle. Mittauksien tulokset on esitetty taulukossa 2.

Anturin käyttöjännitteellä ei näyttänyt olevan vaikutusta tuloksiin ennen kuin se alkaa rajoittaa maksimiarvoa. Joten jos ei ole tarkoitus mitata suuria pölypitoisuuksia niin sen voi jättää huomiotta. Mutta ulostulojännite on erittäin helppo saada alemmaksi kahdesta vastuksesta toteutetulla jännitteenjakokytkennällä, jolloin anturilla voidaan käyttää suunniteltua 5 V:n jännitettä.

Alemmalla käyttöjännitteellä ledi antaa alempia arvoja kuin 5 V:n käyttöjännitteellä, mutta jos ne kertoo noin 1,75:llä niin, päästään aika lähelle samoja arvoja kuin 5V käyttöjännitteelläkin ja jos anturin käyttöjännite on 5 V niin silloin olisi mahdollista mitata suurempia pölymääriä, koska anturin antama maksimiarvo rajoittuu 3,75 volttiin, mutta alemmalla ledin käyttöjännitteellä tämä tulee rajoittavaksi tekijäksi vasta paljon myöhemmin.

Led jännite	5	3,3	3,3	5	
Anturi jännite	5	5	3,3	3,3	
	0,35	0,2	0,2	0,35	
	0,65	0,35	0,35	0,65	
	1,05	0,6	0,6	1,05	
	1,75	0,95	0,95	1,75	
	1,95	1,1	1,1	1,95!	
	2,4	1,35	1,35	2!	
	2,95	1,65	1,65	2!	
	3,3	1,85	1,85	2!	
	3,45	1,9	1,9!	2!	
	3,55	2	2!	2!	
	3,75	2,25	2!	2!	
	3,75	3,7	2!	2!	
! Tarkoittaa että maksimiarvokaan ei nouse yli 2v					

TAULUKKO 2. Mittaustulokset spagetilla

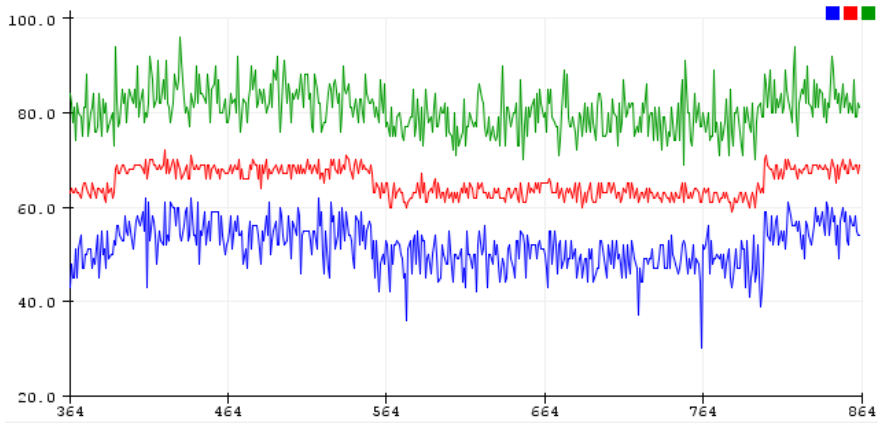
4.3 Anturin käyttöjännitteen korotus

Edellä tehtyjen mittausten perusteella päätettiin korottaa anturin käyttöjännite 5 volttiin. Koska MKRFOX1200 ei kestä yli 3,3 V:n jännitettä, käytettiin anturin ohjaamiseen transistoria ja mittaustulos luetaan jännitteenjakokytkennän kautta, jolloin vältettiin yli 3,3V jännitteen syöttäminen laitteeseen.

4.3.1 Ledin transistoriohjaus

Anturin lediä ohjataan vetämällä lediohjauspinni maihin. Pienemmällä jännitteellä se on voitu toteuttaa kytkemällä anturi suoraan Arduinoon ja ohjaamaan pinniä ohjelmallisesti. Nyt kuitenkin ohjataan transistoria, ettei laitteeseen pääse 5 V:n jännite. Transistorina käytetään saatavilla ollutta 2N 3904 H 331 transistoria, joka on NPN ja se on kytketty Arduinoon vastuksen kautta. Testattiin myös Arduino Unolla, minkälaisia vaikutuksia transistoriohjauksella on mitattuihin arvoihin ja tehtiin kytkennät ja ohjelmat, joissa oli mahdollista vaihtaa ohjaustapaa lennosta. Huomattiin, että transistorikytkentä antaa hiukan pienemmät arvot kuin suora ohjaus, mutta käytetyllä vastuksella voi olla hyvinkin suuri vaikutus mitattuihin arvoihin.

Mittauksissa käytettiin eriarvoisia vastuksia ja huomattiin, että jos vastus on liian pieni taikka suuri niin mittaustulos oli huomattavasti pienempi kuin ilman transistoriohjausta. 100 Ω:n vastus antoi käyttökelvottoman arvon ja samoin myös 68 KΩ:n vastus. 1 KΩ:n vastus antoi lähes saman arvon kuin ohjaus ilman transistoria ja 6KΩ vastus antoi hiukan paremman, joten oletettiin, että jokin vastus 1-10KΩ väliltä on sopiva, koska arvot eivät poikenneet paljoa suoraan ohjatusta anturista. MKRFOX kytkentään laitettiin 2,2 KΩ:n vastus, jota myös vertailtiin muihin saman suuruusluokan vastuksiin ja 2,2KΩ vastus antoi silti suurimman arvon. Ohjelmaa piti myös muuttaa siten että lediohjaus muutetaan nollasta positiiviseksi.

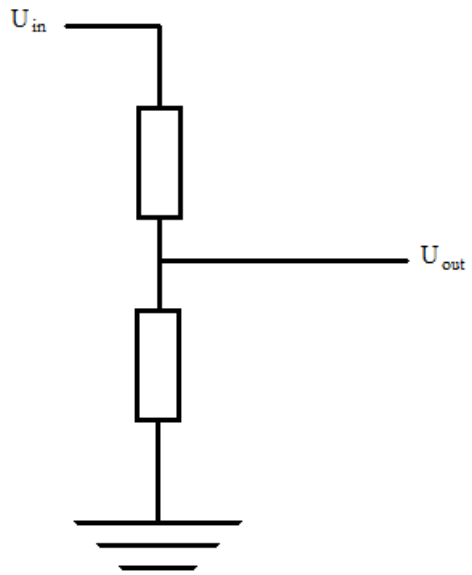


KUVA 7: Eri vastuksien testausta transistorin ohjaamiseen

4.3.2 Mittaustuloksen luku jännitteenjakokytkennän kautta

Anturin lähettämää jännitettä päätettiin pienentää kytkemällä peräkkäin kaksi vastusta ja mittaamalla jännitteen niiden keskeltä. Vastusten mitoitus kuitenkin täytyi suunnitella siten, että mittauskohdassa jännite voisi vaihdella mahdollisimman paljon, mutta ei kuitenkaan ylittää 3,3 volttia, jotta saataisiin mahdollisimman tarkka tulos. Kahdella identtisellä vastuksella jännite mittauskohdassa olisi 5 V:n jännitteellä 2,5 V, ja edellisessä testissä anturi antoi maksimissaan 3,75V, joten mitattava jännite ei voisi vaihdella kovinkaan paljoa ja tarkkuus kärsisi.

Datalehdessä (Sharp 2006, 4) mainittiin anturin virrankulutus, kun lähtöön oli kytketty 4,7 KΩ:n vastus, joten päätettiin käyttää vastuksia joiden yhteenlaskettu suuruus olisi lähellä kyseistä arvoa. Varauduttiin myös siihen, että vastuksien arvot vaihtelevat niihin merkatusta arvosta ja jännitteen pitäisi pysyä myös sallituissa rajoissa, vaikka vastuksien arvot olisivat vaihdelleet ääripäihin. Laskettiin pahin mahdollinen tilanne 4 V:n jännitteellä ja 10 %:n arvopoikkeamalla ja päädyttiin käyttämään 3,9 KΩ:n ja 1 KΩ:n vastuksia, joilla jännite oli kyseissä tilanteessa hieman yli 3,3V. Arvojen pitäessä paikkansa samassa tilanteessa jännite on 3,18V, joten vaihtelua ja tarkkuutta on riittävästi, mutta pitää kuitenkin varoa, ettei tuo vastuksienkaan kautta yli 5V jännitettä, koska vastukset on mitoitettu maksimissaan 4V jännitteelle, että saataisiin mahdollisimman suuri resoluutio mittaustuloksiin.



KUVA 8. Piirros jännitteenjakokytkennästä

4.4 Kalibrointiohjelma

Koska vastuksissa, transistoreissa ja antureissa on vaihtelua ja mittaustulos pitäisi saada myöskin sopimaan mahdollisimman hyvin lähetettyyn viestiin, päätettiin tehdä ohjelma, jonka avulla voitaisiin selvittää minkälaisia, arvoja anturi antaa kyseisessä kytkennässä ja myöskin sopivat arvot, joilla mittaustulos saadaan eteenpäin. Mittausohjelmasta muokattiin kalibrointiohjelma, jossa minimi ja maksimiarvoja ei nollata joka kierroksella, vaan ne muuttuvat vain silloin, jos jokin pienempi tai suurempi arvo tulee mitattua. Ohjelma käyttää näitä arvoja pienimmän ja suurimman mahdollisen mittaustuloksen selvittämiseen.

Suurin mahdollinen maksimitulos selviää, kun kerrotaan maksimiarvo mittausten lukumäärällä ja sama kertolasku tulee tehtyä myös minimitulokselle. Maksimiarvosta vähennetään minimitulos ja saatu arvo jaetaan 255:llä, että saadaan selvitettyä sopiva jakaja jakamaan mitattua tulosta siten, että voidaan lähettää se kahden tavun avulla. Ohjelma kertoo myös arvot, joiden avulla tulos on laskettu ja tarjoaa kolme vaihtoehtoa jakajaksi, joista voi valita parhaan vaihtoehdon esitettyjen tulosten perusteella. Kalibrointiohjelman antamista arvoista jakaja ja minimi on tarkoitus lisätä mittausohjelmaan ennen kuin se ohjelmoidaan mittarille ja jos mittauslaitteita on tarkoitus tehdä useampi kappale niin jokaisesta yksilöstä olisi hyvä selvittää siihen parhaiten sopivat arvot. Lisäksi saatiin idea lisätä Sigfox ID:n tulostuminen kalibrointiohjelmaan, joten käytetyn alustan tunnistukseen ei tarvitse käyttää toista ohjelmaa, vaan yksilöllinen tunniste saadaan selville kalibroinnin ohessa. Kalibrointiohjelman koodi löytyy liitteestä 1.

```
Sigfox ID  = 0018BC03
 Min: 3 Max: 918 Jakaja: 107.65
 Testijakaja: 107 Testitulos: 0 Jakotulos: 256.54
 Testijakaja: 108 Testitulos: 254 Jakotulos: 254.17
 Testijakaja: 109 Testitulos: 251 Jakotulos: 251.83
```

Kuva 9. Kalibrointiohjelman tulostusta

4.5 Virransyöttö patterilta

Koska laitteesta on tarkoitus tehdä helposti siirrettävä ja langaton, niin pitää tämä ottaa huomioon myös virtalähteessä. Lisäksi olisi hyvä, että anturi kuluttaisi mahdollisimman vähän virtaa, että toiminta aika olisi mahdollisimman pitkä. Koska ensimmäinen versio laitteesta oli lähinnä jonkinlainen prototyyppi, jota voisi sitten parantaa jälkikäteen, päätettiin toteuttaa virransyöttö 6AA-patterin avulla. Virransyöttö tapahtuisi L7805 regulaattorin kautta, joka tasaa jännitteen laitteelle sopivaksi ja pattereiden jännitteen tippumisesta ei ole hirveästi haittaa laitteen toiminnan kannalta. Virtaa oli syötetty koekytkennöissä MKRFOX:ille USB-väylän kautta 5V-nastasta eteenpäin tuulettimelle ja anturille, mutta regulaattorin kanssa 5V jännite tulee regulaattorilta ja se tulee sisään VIN-nastasta ja 5V-nastasta ei enää oteta jännitettä anturille ja tuulettimelle, vaan sekin tulee regulaattorin kautta. Koska virransyötön vaihto vaikuttaa jännitteisiin ja USB-väylän kautta kommunikointi ei onnistu, kun virran syöttää muualta, piti kalibrointiohjelmaa muuttaa siten, että sarjaliikenne lähetetään RX ja TX -pinnien kautta ulos ja luetaan sitten sopivalla laitteella.

4.5.1 Transistori ohjaus tuulettimelle ja anturille

Virrankulutuksen vähentämiseksi on hyvä katkaista virta niiltä laitteilta mitä ei käytä ja koska tuuletin vie eniten virtaa koko laitteistosta on siitä hyvä aloittaa. Koska 2N 3904 ei pysty ohjaamaan yli 200mA virtamääriä, niin laitettiin se ohjaamaan BC327 PNP transistoria. Ohjaus olisi ollut helpompi toteuttaa NPN transistorilla, mutta tarkoituksena oli ohjata myös pölyanturille menevää virtaa, joten sekin olisi mahdollista kytkeä pois päältä silloin kun mittaustapahtumaa ei ole ja koska nollat oli tarkoitus pitää maassa niin ei ohjaus olisi NPN transistorilla onnistunut. Alustavissa testikytkennöissä oli vastus ainoastaan NPN transistorissa ja huomattiin, että NPN transistori lämpeni, joten päätettiin laittaa vastus myös transistorien välille viemään vähentämään transistorin lämpenemistä ja toimimaan samalla myös hyppylankana. Testikytkennöissä myös havaittiin, että tuuletin vei paljon virtaa, joten ohjaustransistorin ylle muodostui noin 0,4V jännite ja tämä tarkoitti sitä, että jos anturin olisi kytkenyt saman transistoriohjauksen taakse olisi sille tullut vain 4,6V jännite halutun 5V sijasta. Anturin käyttöjännitteen ohjaus päätettiin toteuttaa erillään tuulettimen ohjauksesta identtisellä kytkennällä ja vastusarvojen laskua varten päätettiin mitata anturin virrankulutus ja laskea vastusarvot siten, että virtaa on varmasti riittävästi. Laskuissa päädyttiin käyttämään 120KΩ vastusta NPN transistorin ohjauksessa ja transistorien välille laitettiin joku pieni alle 1KΩ vastus ja tähän tuli laitettua sitten 120Ω vastus. Samoja vastuksia testattiin myös tuulettimen ohjauksessa ja tuuletin pyöri, joten tuulettimelle ei laskettu erikseen omia vastuksia. Anturin virrankulutus oli huomattavasti pienempi kuin tuulettimen ja transistorin ylle muodostui vain 0,02V jännite, joten oltiin riittävän lähellä toivottua 5V jännitettä, kun jännitettä oli 4,98V. Lisäksi ohjelmia piti muuttaa siten että niihin lisättiin ohjaus. Kuva kytkennöistä löytyy liitteestä 3.

5 MITTAUSOHJELMA

Laitteen kehittyessä yksinkertainen alussa esitetty mittausohjelma on kehittynyt laitteen mukana, ja siihen on tehty tarvittavia muutoksia testitilanteita varten, sekä laitteiston muutoksien johdosta tarpeellisia muutoksia, kuten ledin transistoriohjauksesta johtuva 0 ohjauksen muuttuminen 1 ohjaukseksi. Lisäksi siihen on pitänyt lisätä toimintoja laitteen lopullista käyttöä ajatellen, koska alussa esitelty ohjelma oli tarkoitettu lähinnä anturin ja mittaustapahtuman testaukseen. Nykyisessä ohjelmassa on kolme tehtävää, jotka käydään seuraavissa luvuissa läpi yksitellen. Lopullinen ohjelma on liitteessä 2.

5.1 Mittaustuloksen käsittely

Anturin ohjauslogiikka on pysynyt lähes samana kuin anturin testaukseen käytetyssä ohjelmassa, mutta näytteitä otetaan 10:n sijaan 30 ja tarkastellaan pelkkää keskiarvoa. Näytteet tulee 10-bittisinä kokonaislukuina ja ne lasketaan yhteen ja kalibrointiohjelman avulla selvitettyjen lukujen avulla muutetaan saatu mittaustulos 8-bittiseksi kokonaisluvuksi. Saatu tulos tallennetaan listaan, johon on tarkoitus kerätä 6 muutakin näytettä, ennen kuin ne lähetetään eteenpäin.

5.2 Lepotila

Lepotilaa käytetään laitteen virrankulutuksen pienentämiseksi ja myöskin ajoittamaan näytteenottohetket 2 minuutin välein, jotta Sigfox viestien vuorokautista rajaa ei tulisi ylitettyä. Lepotilaa testattiin aluksi erillisellä ohjelmalla ja kun sitä opittiin käyttämään ja se saatiin toimimaan, lisättiin se osaksi mittausohjelmaa. Virrankulutuksen vähentämiseksi on edellisessä kappaleessa mainitut ohjaukset myös otettu käyttöön ja ennen lepotilaan siirtymistä sammutetaan tuuletin ja anturin käyttöjännite ja ne myös laitetaan päälle, kun herätään lepotilasta ennen mittaustapahtuman alkamista ja varmuuden vuoksi odotetaan pieni hetki ennen mittaustapahtuman käynnistystä, että anturin kondensaattori on ehtinyt latautua ja tuuletin lähtenyt pyörimään.

5.3 Sigfox-verkkoon lähetys

Viestien lähetystä testattiin aluksi erillisellä ohjelmalla, kuten lepotilaakin. Ohjelmassa selvitettiin laitteen tunnus ja sitten koitettiin lähettää viestejä, ja katsoa tuleeko ne perille. Kun lähetys oli todettu toimivaksi, tehtiin siitä osa ohjelmaa. Laitteen tunnuksen selvitys lisättiin kuitenkin kalibrointiohjelmaan. Vuorokautinen viestien maksimimäärä on 140 viestiä ja viestin maksimipituus on 12 tavua, joten mittaustapahtuman yhteydessä kerätyt 6 8-bittistä näytettä lähetetään samalla kertaa, jolloin saadaan koko viestin maksimipituus käytettyä ja näytteet on otettu 2 minuutin välein, joten vuorokaudessa tulee lähetettäväksi 120 viestiä. Mittaustapahtuman päätteeksi tarkistetaan näytteiden määrä listassa ja kun niitä on 6, lähetetään lista eteenpäin ja aloitetaan alusta. Lähetys tapahtuu käynnistämällä Sigfox-yhteys ja aloittamalla paketti. Pakettiin lisätään jokainen arvo erikseen ja sitten paketti lähetetään eteenpäin. Kun paketti on lähtenyt matkaan, suljetaan Sigfox-yhteys ja laite menee nukkumaan.

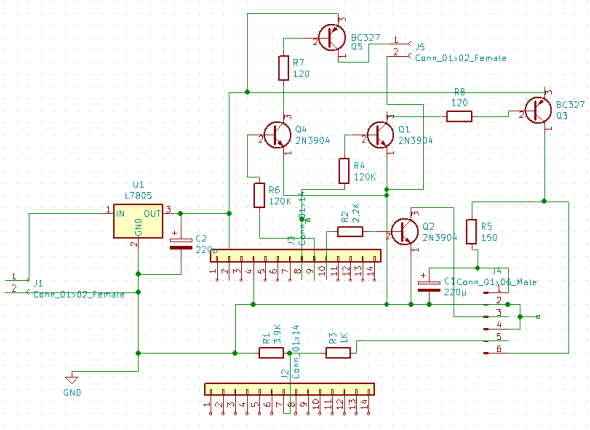
6 PROTOTYYPIN SUUNNITTELU

Laitteesta päätettiin rakentaa prototyyppi, että voitaisiin testata, miten laite toimii todellisessa käyttöympäristössä. Koekytkentäalustalla suunniteltuja ja testattuja kytkentöjä varten piti suunnitella piirilevy, johon kytkennät tehtäisiin ja laitteen osille varten piti suunnitella kotelo. Piirilevy päätettiin suunnitella piirilevyjen suunnitteluohjelmistolla ja valmistaa syövyttämällä. Kotelo taas suunniteltiin tulostettavaksi 3D-tulostimella.

6.1 Piirilevyn suunnittelu

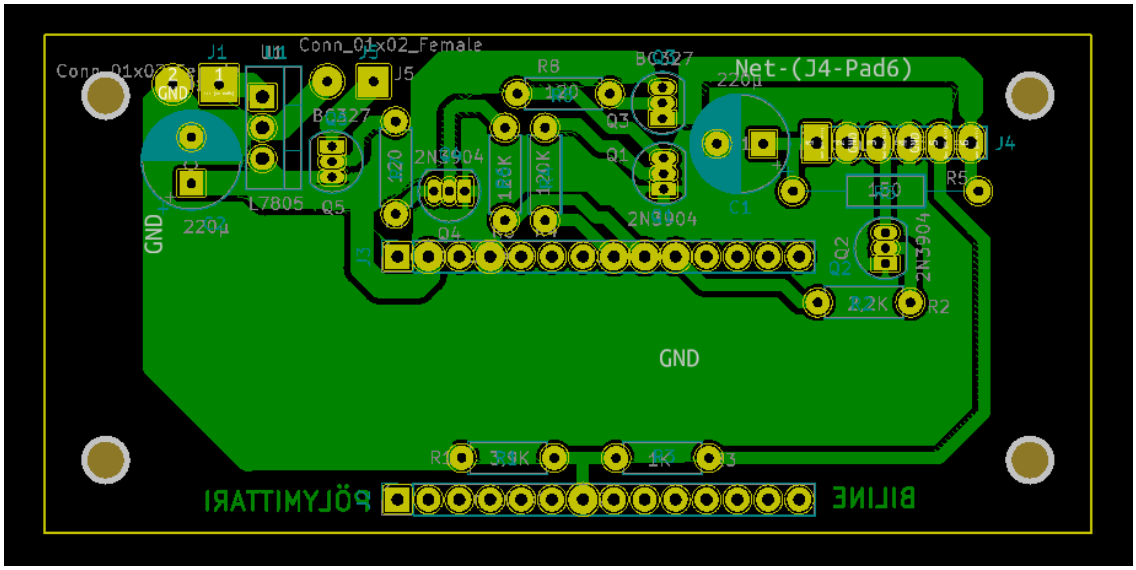
Piirilevyn suunnitteluun käytettiin KiCad ohjelmistoa, ja suunnittelu aloitettiin luomalla uusi projekti. Seuraavaksi luotiin kytkennät avaamalla .sch-päätteinen tiedosto, joka on kytkentätiedosto. Tämän jälkeen haettiin ja laitettiin tarvittavien komponenttien symbolit ja maa place symbol -työkalulla. Suurin osa komponenteista löytyi komponenttikirjastosta, mutta tuuletin, virtajohto ja pölyanturin johdossa käytettiin sopivaan määrän kytkentöjä omaavia liitinsymboleja. MKRFOX1200:n kytkemiseen päätettiin käyttää kahta 14 kytkentää omaavaa liitintä. Tämän jälkeen komponentteja voitiin siirrellä ja pyöritellä, että niiden kanssa olisi helpompi ja selkeämpi työskennellä. Lisäksi niille pystyi antamaan arvot, jolloin oli helpompi tietää mistä komponentista oli kyse. Kun komponentit oli siirrelty sopiville paikoille, muodostettiin kytkennät place wire -työkalun avulla.

Kun kytkentä oli muodostettu, täytyi symboleille etsiä sopiva kotelointi (footprint) Assing footprints -työkalun avulla ja koteloinnin pitäisi olla sellainen, johon todellisen komponentin voi kytkeä. Kotelointeja pystyy katsomaan ja mittaamaan näytöllä ja kotelojen nimistäkin voi päätellä jotain. Kun kytkennät oli tehty ja koteloinnit määritelty, luotiin vetolista (netlist), jonka jälkeen kaikki tarvittava kytkentöjen osalta oli tehty.



KUVA 9. KiCad kytkennät

Varsinaisen piirilevyn suunnittelu aloitettiin avaamalla .kicad_pcb-päätteinen tiedosto, joka on piirilevytiedosto. Suunnittelu aloitetiin tuomalla kytkennät, read netlist -työkalun avulla. Tämän jälkeen ruudulle ilmestyi komponentit ja kytkennät näkyvät viivoina komponenttien jalkojen välillä. Nyt komponentteja voitiin siirrellä ja pyöritellä ja tarkoituksena oli saada komponentit järjestettyä siten että kytkennät ovat mahdollisia. Lisäksi tässä täytyi huomioida, että MKRFOX1200:n jalat tulee sopivalle etäisyydelle toisistaan, koska kytkennöissä oli käytetty kahta erillistä osaa. Kytkennät tehtiin route tracks -työkalut avulla ja välillä kytkentöjä ei ollut mahdollista tehdä, joten osaa komponenteista piti siirrellä ja pyöritellä. Tarvittaessa myös kytkentöjä taikka kotelointeja piti muuttaa, jolloin palattiin takaisin kytkentäohjelman pariin ja sitten päivitettiin piirilevy vastaamaan kytkentää. Tämmöisiä muutoksia oli mm eri ohjauspinnien vaihdot ja yhden vastuksen koteloinnin muuttaminen pidemmäksi, jolloin sitä voitiin käyttää samalla myös hyppylankana, ja osa kytkentäjohdoista voitiin vetää sen ali. Kun kaikki kytkennät oli saatu vedettyä, niin päätettiin täyttää tyhjiä kohtia add filler zones -työkalulla ja suurin osa täytetystä alueesta on kytketty maihin, joten samalla saatiin myös hyvä maataso, johon ei muodostu yhtä helposti potentiaalieroja piirilevyn eri kohdissa, mutta myös 5V jännitteelle on oma hieman pienempi täytetty kohta. Kun suunnittelu oli valmis, voitiin vielä lisätä teksti, että näkee mikä laite kyseessä ja testattiin osien sopivuutta tulostamalla piirilevy paperille ja vertailemalla todellisia komponentteja. Tämän jälkeen voitiin tehdä maski ja syövyttää sen avulla piirilevy. Piirilevykuva ja komponenttien paikat ja arvot on liitteessä 6.



KUVA 10. Piirilevy KiCad ohjelmassa

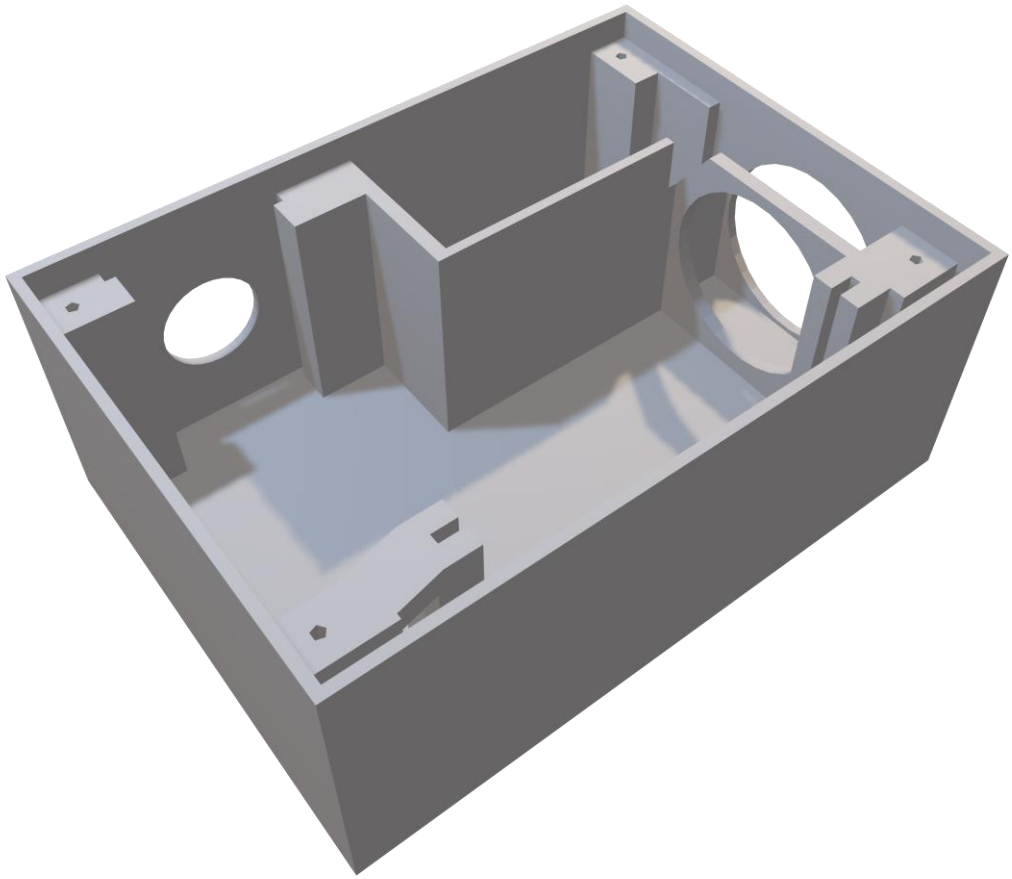
6.2 Kotelon suunnittelu

3D-tulostusta varten täytyi kotelo ensin suunnitella ja toteuttaa suunnitelmien perusteella tulostettava malli. Suunnittelu aloitettiin sommittelemalla osia pöydällä eri asentoihin samalla pohtien osien kiinnitystä ja johtojen vetoa. Kun jonkinlainen idea kotelosta oli mietitty päätettään toteuttaa tämän perusteella 3D-malli. Ennen kuin siirryttiin toteuttamaan mallia tietokoneella, niin mitattiin osista tarpeelliset mitat ja kirjattiin ne ylös paperille.

Mallin toteuttamiseen käytettään OpenSCAD -nimistä ohjelmaa, jossa voidaan luoda 3D-malleja geometristen kappaleiden pohjalta. Mallin luominen muistuttaa hyvin paljon koodausta, koska kappaleiden luonti, koko, kääntö ja liikuttaminen tehdään kirjoittamalla. Lisäksi kappaleita voi yhdistää toisiinsa, taikka poistaa kappaleesta jonkin toisen kappaleen muotoinen pala. Ohjelmassa olisi myös lisää ominaisuuksia, mutta tämän kotelon suunnitteluun edellä mainitut työkalut olivat riittävät.
Mallin luonti aloitettiin luomalla kappaleryhmiä, jotka vastasivat kooltaan ja muodoltaan käytettyjä osia, mutta mittoja oli varmuuden vuoksi hieman kasvatettu, että osat menisivät helpommin paikoilleen. Tämän jälkeen osia liikuteltiin omille paikoilleen, nostettiin 3 milliä ylöspäin ja luotiin iso laatikko, josta osien muodot vähennettiin. Tämän jälkeen luotiin laatikon sisälle tyhjää tilaa leikkaamalla laatikosta pois pienempiä laatikoita. Kiinnitys oli suunniteltu toteutettavaksi siten, että osille oli juuri sopivan kokoiset reiät käytössä ja kun kansi laitetaan päälle niin se pitäisi osat paikoillaan, joten leikkaukset suunniteltiin siten että kotelon sisälle jäi seiniä pitämään osat paikoillaan. Seuraavaksi mietittiin kannen kiinnitystä ja päädyttiin siihen, että kansi kiinnitettäisiin ruuveilla, joten koteloon lisättiin ruuvien reiät.

Tässä vaiheessa lähetettiin tarvittavat tiedostot tulostusta varten ja tulostettiin koeversio kotelosta. Kotelon tulostuksessa oli ongelmia, joten kotelo hieman kärsi, mutta koteloa tutkimalla saatiin tietoa mitä olisi korjattava. Piirilevyä ei oltu suunniteltu ensimmäisessä versiossa kiinnitettäväksi samalla tavoin kuin muita osia, joten piirilevyn kiinnitys muutettiin samanlaiseksi kuin muillakin osilla. Piirilevy oli onneksi samankokoinen kuin mittari ja patterikotelo, joten kotelon korkeutta ei tarvinnut muuttaa, mutta piirilevyä käännettiin 90 astetta siten, että se ei enää ollut samansuuntainen lattian kanssa, vaan se nostettiin pystyyn ja menee nyt lattiasta kattoon. Lisäksi joissain kohdissa materiaali oli liian ohutta ja murtumia oli syntynyt ja seinämät liian joustavia, joten kyseisiä kohtia vahvistettiin ja tuettiin. Aiemmassa kotelosuunnitelmassa oli myös tehty paksujen kohtien sisälle tyhjää tilaa, mutta tulostinohjelmassa olisi ollut mahdollista myös täyttöprosentin valintaan, joten tyhjät tilat päätettiin poistaa ja antaa tulostusohjelmiston lisätä tyhjät tilat. Osia pystyi sovittamaan koteloon vaurioista huolimatta ja

osat sopivat paikoilleen suunnitellusti antennia lukuun ottamatta, koska se oli vahingossa tullut 5 senttiä liian pitkäksi ja mittari hölskyi turhan paljon, joten mittarista tehtiin puoli milliä pienempi. Seuraavan kotelon tulostus onnistui paremmin. Samalla tulostettiin myös kansi, joka oli suunniteltu siten että ruuvien sijainnit oli kopioitu kotelosta ja ruuvin reikiä tehty isommiksi ja lisätty kolo uppokantaa varten, että ruuvi uppoaisi sopivasti kanteen eikä jää koholle. Kotelon ja kannen koodit löytyy liitteistä 4 ja 5.



KUVA 11: Kotelon 3D-malli

7 PROTOTYYPIN KÄYTTÖÖNOTTO

Laitteen käyttöönotto aloitettiin kalibroimalla. Kalibrointi tapahtui ohjelmoimalla kalibrointiohjelma MKRFOX1200:n ja kun ohjelmointi oli suoritettu, irrotettiin usb-kaapeli ja kiinnitettiin MKRFOX1200 piirilevylle sille varattuun paikkaan. Koska kalibrointiohjelman tiedot tulostetaan sarjaportin kautta, kytkettiin sarjaporttiin laite, joka luki lähetetyt viestit ja tulosti ne tietokoneen näytölle. Kun kytkennät oli tehty, yhdistettiin patteri ja odoteltiin jonkin aikaa, että minimiarvo ei enää laskenut ja tämän jälkeen työnnettiin kynä anturin läpi, jolloin saatiin selville maksimiarvo. Kalibrointiohjelman jakajista valittiin sopiva ja tässä tapauksessa kuvassa näkyvistä jakajista valittiin 108, koska 107 oli liian pieni ja tulos voi pyörähtää ympäri, jolloin tuloksena näkyy 0, vaikka arvo on maksimissa ja 109 oli ehkä vähän turhan iso. Lisäksi otettiin muistiin myös Sigfox ID, että tiedetään mistä laitteesta on kyse. Kalibrointiohjelman tulostus näkyy alla olevassa kuvassa. Kun kalibrointi oli saatu suoritettua lisättiin saadut arvot mittausohjelmaan ja ohjelmoitiin se MKRFOX1200:n. Tämän jälkeen osat laitettiin omille paikoilleen koteloon, kytkettiin virta ja laitettiin kansi päälle, jonka jälkeen laite oli toiminnassa.

```
Sigfox ID  = 0018BC03
 Min: 3 Max: 918 Jakaja: 107.65
 Testijakaja: 107 Testitulos: 0 Jakotulos: 256.54
 Testijakaja: 108 Testitulos: 254 Jakotulos: 254.17
 Testijakaja: 109 Testitulos: 251 Jakotulos: 251.83
```

Kuva 9: Kalibrointiohjelman tulostusta

8 JOHTOPÄÄTÖKSET

Tässä työssä oli hyvin monta erilaista työvaihetta ennen kuin annettu tehtävä oli muuttunut valmiiksi laitteeksi, ja jokaisessa vaiheessa piti tehdä omat johtopäätökset ennen kuin jatkoi eteenpäin. Anturin toimintaa testattaessa huomasin, että vaihtelu minimi- ja maksimiarvon välillä oli hyvin suurta, joten pölyn määrää oli parempi kuvata usean mittauksen keskiarvolla, kuin yksittäisellä mittaustuloksella ja myöskin tuuletin koettiin tarpeelliseksi. Anturin toimintaa alijännitteellä myös tutkittiin ja huomattiin että, mittausalue pienenee samalla, kun anturin jännite pienenee ja anturin ledin käyttöjännitteen pienentäminen taas pienentää anturin antamaa tulosta.

Anturia päädyttiin käyttämään sille suunnitellulla 5 V:n jännitteellä, ja tämä johti uusien kytkentöjen suunnitteluun, koska anturista tuleva jännite olisi ollut liian suuri MKRFOX1200:lle, joten piti laittaa jotain väliin. Jännitteenjakokytkennässä mielestäni oli hyvä saada mahdollisimman laaja mittausalue, joten vastukset valittiin sen perusteella. Anturin ledin ohjauksessa transistoriohjaus antoi pienempiä arvoja kuin suora ohjaus, joten pohdin, onko transistorin ohjausvirran määrällä jotain vaikutusta ja päätin rajoittaa sitä erikokoisilla vastuksilla, koska vaikutti siltä, että mitä enemmän ohjausvirtaa transistorille niin sitä kauemmas kollektori siirtyy nollasta ja anturin antama tulos pieneni. Jos taas rajoitti ohjausvirtaa liikaa suurella vastuksella, anturin antama tulos pieneni, joten päädyin testaamaan erikokoisia vastuksia ja ottamaan sen jossa tulos pieneni kaikista vähiten.

Aluksi oli tarkoitus tehdä ohjaus tuulettimelle ja anturille saman transistorin kautta, mutta tuuletin vei paljon virtaa, joten ohjaustransistorin ylimenevä jännite oli merkittävä. Päätin ottaa ohjaukset erilleen, jotta saatiin anturille mahdollisimman suuri jännite. Tuulettimella ei ollut niin paljoa väliä, koska riittää, että se pyörii jotenkin. Rajoittamalla tuulettimen saamaa virtaa voidaan myös vähentää laitteen virrankulutusta.

Kytkentöjen suunnittelun jälkeen lisättiin ohjelmaan muutokset, joita kytkennät olivat aiheuttaneet ja koska vastuksissa on vaihtelua, päätin tehdä sitä varten kalibrointiohjelman, ettei arvot muuttuisivat laitteiden välillä johtuen eriarvoisista vastuksista ja samalla myös antaisi jotain lukuja, joiden avulla mittaustulos saatiin sopimaan 8-bittiselle arvolle. Testasin ohjelmaa ja kalibroinnissa käytin virtalähteenä tietokoneen usb-paikkaa, ja kalibroinnin jälkeen laitteen virta tuli usb-laturista. Vaikka kalibroinnissa oli valittu sopivat arvot, pyörähtivät arvot ympäri. Tähän syynä oli se, että usb-laturin jännite poikkesi tietokoneen usb-paikan jännitteestä, joten kalibroinnin kommunikaatio siirrettiin usb-portista erilliseen

sarjaporttiin, jotta voitiin syöttää virtaa samasta virtalähteestä sekä kalibroinnissa että myöskin laitteen toimiessa ja jotta usb-portin virransyötöllä ei olisi mitään vaikutusta arvoihin, koska se olisi irti.

Ennen prototyypin valmistusta olin huolissani myös siitä, onko 8-bittiset luvut riittävän tarkkoja ja näkyykö vaihtelua todellisissa tilanteissa, mutta prototyypin valmistumisen jälkeen asia voitiin testata ja testasin prototyyppiä mittaamalla pölyä 10 sekunnin välein aivan kuten mittausohjelmassakin, mutta arvo ei lähtenyt Sigfox:in kautta ulos, vaan luin sen sarjaportin kautta. Arvo oli suurimman osan ajasta joko 10 taikka 11, mutta kun käännyin tuolilla, se taisi nostattaa hieman pölyä ilmaan ja arvo oli tämän jälkeen hetken aikaa 12, joten pienet muutokset pölyn määrässä näkyy mittaustuloksissa ja mittarin antamat arvot sopivat todellisiin tilanteisiin. Myöhemmin prototyyppi laitettiin koulun sisätiloihin lähettämään tietoa Sigfox-verkkoon ja siinäkin pölyn arvot vaihteli kolmen eri arvon välillä, aivan kuten sarjaportinkin kautta testatussa huoneilmassa, mutta nyt kuitenkin pölyä oli mitattu kolmen tunnin ajalta ja mittauksia oli tehty 2 minuutin välein. Pölyn määrä näyttäisi olevan huoneilmassa aika lailla vakio, joten laite olisi hyvä laittaa johonkin, missä pölyn määrä vaihtelisi paljon. Puhaltamalla pölynimurin pölypussista pölyistä ilmaa mittaria kohden nosti mitattua tulosta todella paljon hetkellisesti ja sen jälkeen pölyn määrä oli sen jälkeen koholla, mutta laskeutui vähitellen, kun pölykin laskeutui. Pölymittaria olisi mielenkiintoista testata ulkona pidempikestoisilla mittauksilla esimerkiksi kadun läheisyydessä, jolloin voisi nähdä onko vuorokauden ajalla vaikutusta mittaustuloksiin esimerkiksi mitatussa liikennepölyn määrässä tai kun teitä lakaistaan talven jälkeen lakaisukoneella, tai vaikuttaako siitepölyn määrä mitattuun ilman pölyisyyteen.

Piirilevyn suunnittelussa olisi ollut hyvä korvata pienten transistorien kotelot hieman suuremmilla, jolloin juottaminen olisi ollut helpompaa. Lisäksi olin valinnut suurimpaan osaan vastuksista hiukan lyhyen koteloinnin, joten niitä oli vaikea saada paikoilleen piirilevyä vasten, ja osa jäi hieman koholle, joten nekin olisi hyvä muuttaa. Kondensaattorit hankittiin suunnittelun jälkeen ja niiden kotelointi poikkesi hiukan suunnittelussa käyttämästäni kondensaattorista, joten nekään eivät menneet paikoilleen niin kuin olin suunnitellut, mutta toimivat kuitenkin. Kotelon suunnittelu onnistui hyvin ensimmäisen yrityksen ja erehdyksen kautta, mutta kotelon reunat jäivät turhan korkeiksi, joten kansi ei mennyt ihan alas asti ja osat jäivät hölskymään. Tämä ei haittaa laitteen toimintaa, ja hölskyminen saadaan kuriin laittamalla jotain kannen ja osien väliin, mutta tätä voisi parantaa seuraavassa koteloversiossa esimerkiksi tekemällä reunoista matalammat.

LÄHTEET

Arduino, 2018. What is Arduino?. Saatavissa:
https://www.arduino.cc/en/Guide/Introduction
Viitattu 21.6.2018.

Arduino, 2018. Arduino MKRFOX1200. Saatavissa:
https://www.arduino.cc/en/Main.ArduinoBoardMKRFox1200
Viitattu 25.6.2018.

Borchers J. ,2015. Arduino in a Nutshell. Saatavissa:
http://hci.rwth-aachen.de/arduino
Viitattu 21.6.2018.

ConnectedFinland. 2018. Saatavissa:
http://www.connectedfinland.fi/
Viitattu 25.12.2018

igorfonseca83. 2016. Saatavissa:
http://forum.fritzing.org/t/gp2y1010au0f-sharp-dust-sensor/2165
Viitattu: 26.6.2018

Isohanni. J. Sigfox – maailmanlaajuinen IoT-verkkoliittymä. 2017, Saatavissa:
https://centriabulletin.fi/sigfox-maailmanlaajuinen-iot-verkkoliittyma
Viitattu: 25.12.2018

Sharp, 2006. GP2Y1010AU0F Compact Optical Dust Sensor. Saatavissa:
https://www.sparkfun.com/datasheets/Sensors/gp2y1010au_e.pdf
Viitattu 20.6.2018.

Sigfox. 2018. Technology Overview. Saatavissa:
https://www.sigfox.com/en/sigfox-iot-technology-overview
Viitattu 25.12.2018

Sigfox. 2018. Radio Technology Keypoints. Saatavissa:
https://www.sigfox.com/en/sigfox-iot-radio-technology
Viitattu 25.12.2018

LIITE 1 - KALIBROINTIOHJELMA

```
#include <SigFox.h>

int tuuletin = 11;  //tuulettimen ohjauspinni
int anturi = 12;    //anturin käyttöjännitteen ohjauspinni
int ledOhj = 10;    //ledin ohjauspinni
int antMit = 5;     //anturin mittauspinni
int mitMit = 0;     //mittaus muuttuja
int mitMin = 1024;  //minimimittaus muuttuja
int mitMax = 0;     //maksimimittaus muuttuja

int samples = 30;   //30 näytettä, kuten mittausohjelmassakin
unsigned int mittaukset;
float jakaja;
byte testiarvo;
float testijaettu;
int testijakaja;
int kierros = 0;

String sigfoxID;

void setup() {
  Serial1.begin(9600);    //sarjaportti päälle rx ja tx nastoista

  pinMode(ledOhj, OUTPUT);  //asetetaan ohjauspinnit ulostuloiksi
  pinMode(anturi, OUTPUT);
  pinMode(tuuletin, OUTPUT);
  digitalWrite(ledOhj, LOW);    //anturin led pois päältä
  digitalWrite(tuuletin, LOW);  //tuuletin pois päältä
  digitalWrite(anturi, HIGH);   //anturille virrat
  SigFox.begin(); //käynnistetään sigfox
  sigfoxID = ("Sigfox ID  = " + SigFox.ID()); //otetaan selville sigfox ID
mittarin tunnistusta varten
  SigFox.end(); //suljetaan sigfox

delay(1000);  //riittävän pitkä tauko että
}

void loop() {

    digitalWrite(ledOhj,HIGH); //ledi päälle
    delayMicroseconds(280); //vaadittu tauko
    mitMit = analogRead(antMit);  //mitataan tulos
    delayMicroseconds(40);  //vaadittu tauko
    digitalWrite(ledOhj,LOW);//sammutetaan ledi
    delayMicroseconds(9680);//vaadittu tauko
    if(mitMin>mitMit) mitMin = mitMit;  //minimin tarkistus
```

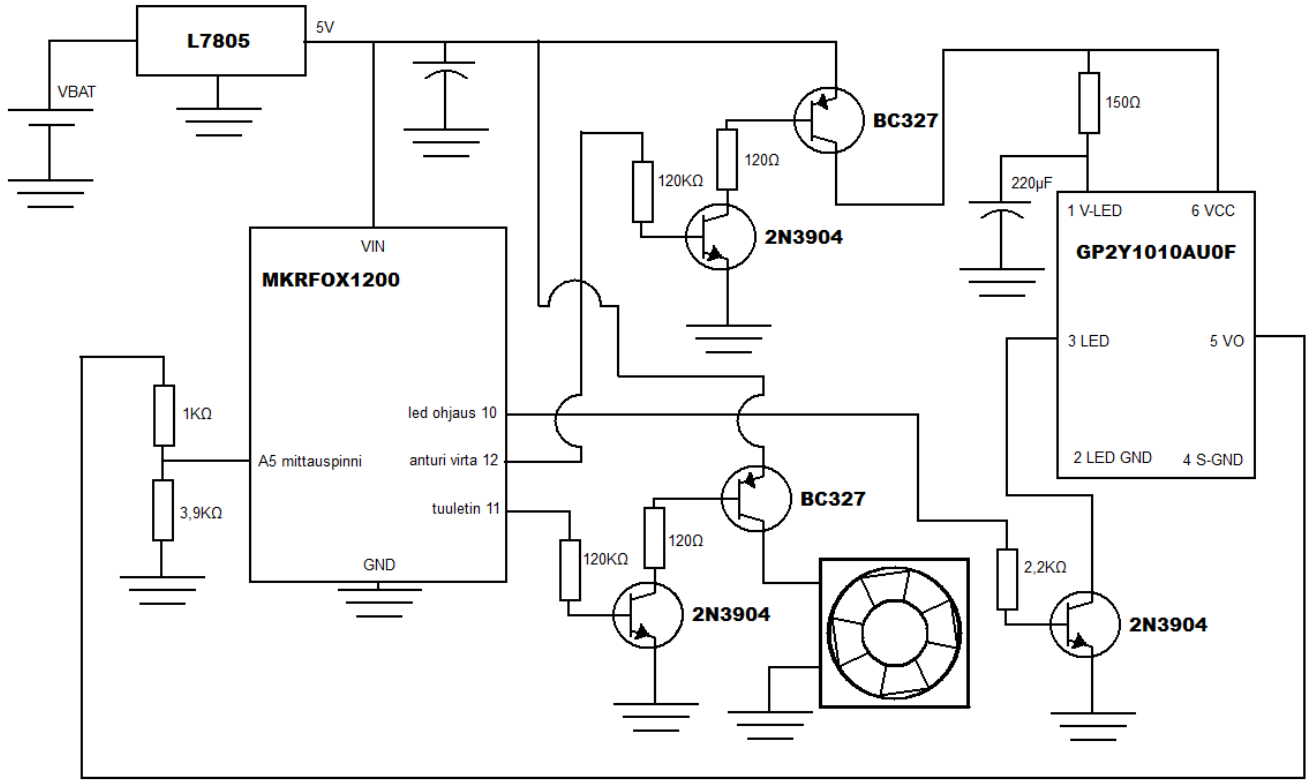
```
    if(mitMax<mitMit) mitMax = mitMit;  //maksimin tarkistus
    kierros++;
  if((kierros%50)==0) //tulostetaan kalibroinnin tulokset 50 kierroksen vä-
lein
  {
    mittaukset = mitMax*samples;    //lasketaan suurin mittaustulos
    mittaukset -= (mitMin*samples); //vähennetään suurimmasta pienin mit-
taustulos
    jakaja = mittaukset/255.0;      //selvitetään jakaja

    Serial1.println(sigfoxID);       //tulostetaan sigfox ID

    Serial1.print(" Min: "); //tulostetaan arvot kalibrointia varten
    Serial1.print(mitMin);
    Serial1.print(" Max: ");
    Serial1.print(mitMax);

    Serial1.print(" Jakaja: "); //tulostetaan alustava jakaja liukulukuna
    Serial1.println(jakaja);
    testijakaja=int(jakaja); //muutetaan se testejä varten kokonaisluvuksi

    for(int i = 0;i<3;i++) //testataan kolmea erisuuruista jakaajaa
    {
    testiarvo=byte(mittaukset/testijakaja); //testaus 8-bittiseen lukuun
    testijaettu=mittaukset/(testijakaja*1.0); //edellinen mutta liukulukuna
    Serial1.print(" Testijakaja: ");   //jakajana käytetty jakaja
    Serial1.print(testijakaja);
    Serial1.print(" Testitulos: ");    //mitä tulee ulos
    Serial1.print(testiarvo);
    Serial1.print(" Jakotulos: "); //minkälainen liukuluku
    Serial1.println(testijaettu);
    testijakaja++;         //kasvatetaan jakajaa seuraavalla kierrokselle
    }
}
}
```

LIITE 2 - MITTAUS JA LÄHETYSOHJELMA

```
#include "ArduinoLowPower.h"
#include <SigFox.h>

int tuuletin = 11;  //tuulettimen ohjauspinni
int anturi = 12;    //anturin käyttöjännitteen ohjauspinni
int ledOhj = 10;    //ledin ohjauspinni
int antMit = 5;     //anturin mittauspinni

////////////////////////////////////////////////////////////
int jakaja = 108;   //SELVITÄ SOPIVA JAKAJA KALIBROINTIOHJELMALLA
////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////
int minimi = 3;   //SELVITÄ MINIMIARVO  KALIBROINTIOHJELMALLA
////////////////////////////////////////////////////////////

int mitMit;                 //mittausmuuttuja
int samples = 30;           //näytteiden määrä
unsigned int mitYht = 0;    //mittaustulos yhteensä

int kierros = 0;  //kierroslaskuri
byte mittaukset[] = {0,0,0,0,0,0}; //mittaustuloksien tallennus

void setup() {
LowPower.attachInterruptWakeup(RTC_ALARM_WAKEUP, dummy, CHANGE);  //nuk-
kumista varten
minimi *= samples;

  pinMode(ledOhj, OUTPUT);  //asetetaan ohjauspinnit ulostuloiksi
  pinMode(anturi, OUTPUT);
  pinMode(tuuletin, OUTPUT);

  digitalWrite(ledOhj, LOW);    //anturin led pois päältä
  digitalWrite(tuuletin, LOW);  //tuuletin pois päältä
  digitalWrite(anturi, LOW);   //anturin virrat pois päältä
}

void loop() {

  digitalWrite(anturi, HIGH);   //anturin virrat päälle mittausta varten
  delay(100);                  //annetaan kondensaattorin latautua
  digitalWrite(tuuletin, HIGH);  //tuuletin päälle mittausta varten
  delay(100);                  //odotellaan että tuuletin pyörii kunnolla
ja mahdolliset jännitepiikit ei haittaa mittausta
  mitYht = 0; //nollataan mittaustulos
```

```
  for(int i = 0;i<samples;i++)  //mittaustapahtuma jossa otetaan mittauksia
10 millisekunnin välein 30 kpl, eli yhteensä 300ms
  {
    digitalWrite(ledOhj,HIGH);    //ledi päälle

    delayMicroseconds(280);       //vaadittu tauko
    mitMit = analogRead(antMit);  //mitataan tulos

    delayMicroseconds(40);      //vaadittu tauko
    digitalWrite(ledOhj,LOW);   //sammutetaan ledi

    delayMicroseconds(9680);    //vaadittu tauko
    mitYht += mitMit;           //tuloksen yhteenlasku
  }

  digitalWrite(tuuletin, LOW);  //tuuletin pois päältä mittauksen lopuksi
  digitalWrite(anturi, LOW);   //anturin virrat pois päältä mittauksen lo-
puksi

  mittaukset[kierros] = byte((mitYht-minimi)/jakaja); //lasketaan sopiva
arvo lähetettäväksi eteenpäin ja laitettaan se muistiin
  kierros++;  //kasvatetaan kierrosta että seuraava mittaustulos tulee eri
muistipaikkaan

  if(kierros==6)  //6 kierroksen välein lähetetään tulokset eteenpäin
  {
    SendPacket();
    kierros=0;
  }
  LowPower.sleep(119500); //nukutaan 2 minuutttia - 500ms(300ms mittausta-
pahtuma 200ms tuuletin ja anturin käynnistys tauko)
}

void SendPacket()   //mittaustulosten lähetys eteenpäin
{
  SigFox.begin();   //sigfox käyntiin
  SigFox.beginPacket(); //uuden paketin teko
  for(int i=0; i<6;i++) //lisätään mittaustulokset yksitellen sigfox paket-
tiin
  {
  SigFox.write(mittaukset[i]);  //tuloksen lisäys
  }
  SigFox.endPacket(); //paketti valmis, joten lähetetään paketti
  SigFox.end();  //sammutetaan sigfox
}
void dummy() {} //nukkumista varten varmuuden vuoksi
```

LIITE 3 - KYTKENNÄT



LIITE 4 – KOTELO

```
difference()
{
cube([135,97,55]); //kotelon ulkoseinät

union()
    {
//leikkaus
translate([12,9,3])
cube([107,53,50]);
translate([2,17,3])
cube([60,68,50]);

//kansileikkaus
translate([2,2,49])
cube([131,93,10]);

//ruuvit
translate([7,12,28])
cylinder(r1=1.25,r2=1.25, h=22);
translate([7,90,28])
cylinder(r1=1.25,r2=1.25, h=22);
translate([128,12,28])
cylinder(r1=1.25,r2=1.25, h=22);
translate([128,90,28])
cylinder(r1=1.25,r2=1.25, h=22);

//mittari
translate ([47,95,3])
rotate([0,0,180])
{
    //mittarin paikka ja reikä
    cube([31,13,47]);
    //reunat
    translate([-2,0,0])
    cube([35,2.5,47]);

    translate([15.5,3,23.5])
    rotate([90,90,0])
    cylinder(r1=10,r2=10,h=15,center=true);
}

//tuuletin
translate([127.5,40,27]){rotate([90,0,90])
    {
   //runko
    translate([-20.5,-20.5,-5.5])
```

```
    cube([41,45,11]);
    //johtoreikä
    translate([16,7,-5.5])
    cube([10,19,11]);
    //reikä
    translate([0,0,0])
    cylinder(r1=19.5,r2=19.5,h=30,center=true);
}}

//antenni
translate([2,2,33])
cube([131,5,17]);
translate([17,2,3])
cube([102,15,53]);

//patterikotelo
translate([65,64,3])
rotate([90,0,90])
{   cube([31,47,59]);
    //virtapistoke
    translate([-2,0,58])
    cube([19,50,7.1]);
    translate([5,25,52])
    rotate([0,0,35])
    cube([20,30,6]);
}}}
//piirilevyn pidikkeet
translate([25,10,3])
rotate([0,0,0])
difference()
{
union(){
cube([94,10,46]);

//tukipilari
translate([-13,-3,41])
rotate([0,0,12])
cube([15.6,10,5]);
    }
union()
    {
translate([3,4,0])
        {
cube([88,2,47]);

translate([5,-5,0])
cube([78,12,47]);
        }}}
```

LIITE 5 – KANSI

```
difference()
{
    union(){
        translate([0,0,5])
cube([135,97,2]);
    translate([2,2,0])
cube([131,93,6]);
    }
union()
    {
//ruuvit
translate([7,12,0])
        {
cylinder(r1=2,r2=2, h=22);
translate([0,0,4.1])
cylinder(r1=1,r2=3.5, h=3);
        }
translate([7,90,0])
        {
cylinder(r1=2,r2=2, h=22);
translate([0,0,4.1])
cylinder(r1=1,r2=3.5, h=3);
        }
translate([128,12,0])
        {
cylinder(r1=2,r2=2, h=22);
translate([0,0,4.1])
cylinder(r1=1,r2=3.5, h=3);
        }

translate([128,90,0])
        {
cylinder(r1=2,r2=2, h=22);
translate([0,0,4.1])
cylinder(r1=1,r2=3.5, h=3);
        }
}
}
```

LIITE 6 – PIIRILEVYKUVA JA KOMPONENTIT