

Nils Kr. Rossing

Måling av luftkvalitet og posisjon med Arduino – LÆRER og ELEVHEFTE



NTNU



Trondheim

Institutt for
fysikk

Skolelaboratoriet
for matematikk, naturfag
og teknologi

Mars 2020

Måling av luftkvalitet og posisjon med Arduino – LÆRER og ELEVHEFTE

Nils Kr. Rossing

Måling av luftkvalitet og posisjon med Arduino – LÆRER og ELEVHEFTE

Trondheim 2020

Layout og redigering: Nils Kr. Rossing, Skolelaboratoriet ved NTNU

Trykk: NTNU Grafisk senter

Tekst og bilder: Nils Kr. Rossing, Skolelaboratoriet ved NTNU

Faglige spørsmål rettes til:

Skolelaboratoriet for matematikk, naturfag og teknologi

v/ Nils Kr. Rossing nils.rossing@ntnu.no

Skolelaboratoriet ved NTNU

Realfagbygget,

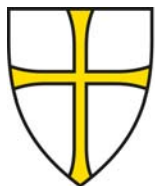
Høgskoleringen 5,

7491 Trondheim

Telefon: 73 55 11 43

<http://www.ntnu.no/skolelab/>

Rev 2.0 – 15.03.20



**Trøndelag
fylkeskommune**



Forord

Heftet beskriver oppbygging, uttesting og noe kalibrering og bruk av et hjemmelaget instrument for måling av luftkvalitet. Undervisningsopplegget bygger på et prosjekt som ble utviklet i forbindelse med Science Camp 2019 på oppdrag av **Trøndelag fylkeskommune** som bekostet utviklingen av tilbudet. Den gang ble det utviklet for et dagsopplegg for interesserte elever fra videregående skole. Opplegget som heftet beskriver er derimot ment som et større prosjekt og inviterer elevene til å bygge opp programvaren steg for steg med forståelse, og burde egne seg både som et elevhefte og en lærerveiledning. Undervisningsopplegget vil derfor gi en dypere forståelse av teknologien bak og burde også egne seg som en fordypning i programmering av Arduino for elever i videregående skole.

Tanken er at instrumentet skal kunne anvendes for måling av partikkeltetthet i luft i tillegg til at det måler fuktighet og temperatur med tilhørende GPS-posisjon. Dataene logges i en datafil på et SD-kort slik at de kan etterbehandles og analyseres i ettertid med passende programvare.

Ideen til prosjektet er hentet fra Air:bit som er utviklet ved Universitetet i Tromsø, se air-bit.uit.no. En takk til **Edvard Pedersen** som har latt oss bruke ideen. En takk også til **Tore Nordstad** ved Trondheim kommune, Miljøenheten som har bidratt med informasjon om luftkvalitetsdata fra Trondheim, se vedlegg H.

Det er vårt håp at opplegget kan gjennomføres som et prosjekt i forbindelse med fag som f.eks. teknologi og forskningslære (ToF). Det er ikke tvil om at stabiliteten til instrumentet ville ha blitt vesentlig bedre dersom sensorene hadde vært loddet opp på et kretskort. Vi tenker at dette ev. kan være fase 2 i et større prosjekt etter at man har lært instrumentet å kjenne ved bruk av koblingsbrett. Gjenbruk av komponenter er også viktig med tanke på bærekraft. En takk til **Bjørn Pedersen** ved Strinda videregående skole i Trondheim som tok sjansen på å starte opp prosjektet våren 2020 med sine ToF-elever

Trondheim 15.03.2020
Nils Kr. Rossing
Skolelaboratoriet ved NTNU
Institutt for fysikk



Innhold

1 Innledning	11
1.1 Kort beskrivelse av instrumentet	11
2 Montering, oppkobling og uttesting av luftfuktighet- og temperatursensor	13
2.1 Oppbygging av sensor for måling av luftfuktighet og temperatur	13
2.2 Oppkobling på koblingsbrett	15
2.3 Program for avlesing av temperatur og luftfuktighet	15
2.4 Kalibrering	17
3 Montering av display	19
3.1 Oppkobling av displayet	19
3.2 Programmet	19
3.2.1 Installasjon av biblioteker	20
3.2.2 Initiering av display	20
3.2.3 Skrive til displayet	21
3.2.4 Det komplette programmet ser da ut som følger:	22
4 Montering av SD-kort	25
4.1 Oppkobling av SD-kortet	26
4.2 Programmet	27
5 Montering av partikkelsensor	29
5.1 Oppkobling av partikkelsensoren	29
5.2 Programmet	30
5.2.1 Installasjon av biblioteker	30
5.2.2 Inkludering og initialisering av biblioteket og partikkelsensoren	31
5.2.3 Komplettest-programmet for partikkelmåler:	33
6 Inkludering av GPS-mottakeren	37
6.1 Oppkobling av GPS-mottakeren	37
6.2 Programmet	38
6.2.1 Installasjon av bibliotek	38
6.2.2 Inkludering og initialisering av biblioteket og GPS-mottakeren	38
6.2.3 Det komplette GPS-testprogrammet ser da slik ut:	43
6.2.4 Avleste data	47
7 Den komplette luftkvalitetsmåleren	49

8	Oppbygging av et kabinett	55
8.1	Design av boksen	55
8.2	Sett sammen boksen	55
8.3	Første sjekk	60
8.4	Installer programmet og test at det fungerer	60
8.5	Prosedyre for å observere data når man er utendørs eller nær et vindu	60
8.6	Displayet	61
8.7	Prosedyre for testing av instrumentet innendørs	62
8.8	Sjekk luftfuktighet og temperatur	62
8.9	Sjekk partikkeltetthet	62
8.10	Sjekk GPS-målinger og lagring av data	63
8.11	Feltmålinger	64
8.11.1	Dra ut til rundkjøring ved Lerkendal	65
8.11.2	Dra ned til krysset Elgeseter gata og gjør målinger	66
8.12	Les av data fra kommunal målestasjon og sammenlign med egne målinger	68
8.13	Behandling av data fra hjemmelaget instrument	68
8.14	Konklusjon og presentasjon	69
Vedlegg A	Tekniske detaljer om luftkvalitetsmåleren	70
A.1	Koblingsskjema og oppkobling	70
Vedlegg B	Oversikt over komponenter med pinning	73
B.1	Luftfuktighet og temperatur målinger	73
B.2	SDS011 - Støvmåler	74
B.3	NEO-6M - GPS	74
B.4	OpenLog	75
B.5	Arduino Nano	75
B.6	RGB LED	76
B.7	Komponent innkjøpliste	76
Vedlegg C	Kort innføring bruk av Arduino editoren	78
C.1	Prosedyre for å sette opp editoren for kortet Arduino Nano:	78
C.2	Prosedyre for å behandle programmet:	78
Vedlegg D	Komponentdata	80
D.1	Luftfuktighetssensor - HiH 4000	80
Vedlegg E	Komplett programkode	83



Vedlegg F Behandling av data	87
F.1 Behandling av data for plotting av ruta i Google Earth	87
F.2 Plotting i Google Earth	91
Vedlegg G BQ20 Partikkelmåler	95
G.1 Kortfattet brukerveiledning	96
Vedlegg H Måling av luftkvalitet med data fra Tr.heim	99
H.1 Hva består luftforurensningen av?	99
H.2 Kilder til luftforurensning	101
H.3 Hva bestemmer hvor mye luftforurensning det er der jeg er?	103
H.4 Forholdene i Trondheim	105



1 Innledning

I denne oppgaven skal dere bruke Arduino UNO (ev. NANO) til å lage et instrument som måler luftkvalitet. Vi bygger opp instrumentet bit for bit, og begynner med måling av luftfuktighet og temperatur. Deretter monterer vi et lite display som viser verdiene for fuktighet og temperatur.

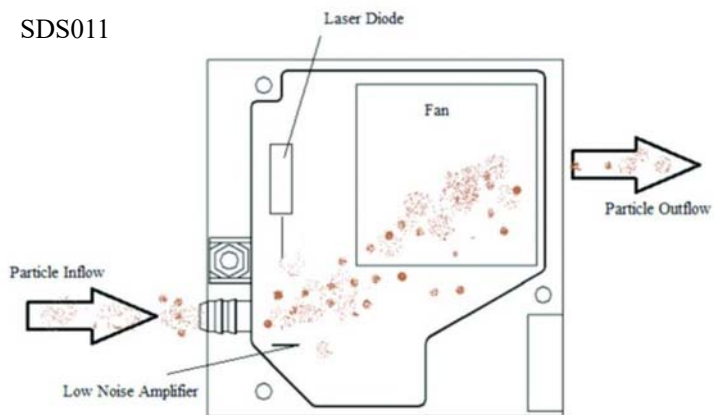
Det neste vi gjør er å montere partikkelmåleren og ser at vi er i stand til å måle partikkeltett i lufta. Dernest monterer vi en liten SD-kortleser slik at vi kan lagre dataene på et SD-kort, dermed er vi uavhengig av å ta med oss PC'en ut i terrenget.

Til sist skal vi montere en GPS-mottaker slik at vi også får logget posisjonen for målingene våre. Til slutt må vi se på hvordan vi kan presentere dataene våre.

Det vil også være aktuelt å bygge instrumentet inn i en boks for å beskytte det mot vær og vind.

1.1 Kort beskrivelse av instrumentet

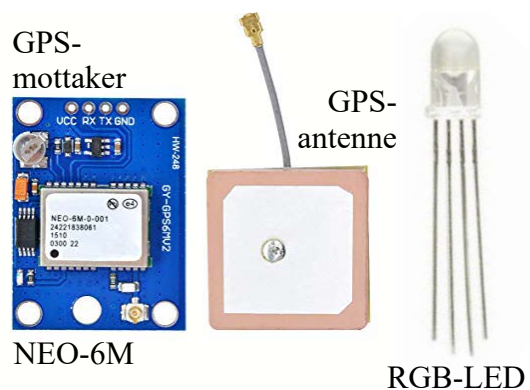
En **partikkelmåler** består av et mørkt kammer og en vifte som drar inn luft i kammeret gjennom et lite rør i siden. En laser belyser luftprøven som fyller kammeret. En lys-sensor er plassert til siden for laserstrålen slik at den kun fanger opp det lyset som spres på grunn av partiklene i kammeret. Dette signalet forsterkes, digitaliseres og regnes om til en partikkelkon-sentrasjon $\mu\text{g}/\text{m}^3$.



En **GPS-mottaker** med antenne etablerer kontakt med et antall satellitter. Disse sender ned tidssignaler som mottakerne benytter for å beregne posisjonen på jord-overflata. Det kan ta noen minutter før kommunikasjonen med satellittene er etablert og en lysdiode begynner å blinke blått på GPS-mottakeren.

En stor **RGB LED** vil lyse enten rødt eller grønt avhengig av om GPS-mottakeren klarer å beregne en gyldig posisjon.

- Rødt ved mislykket posisjonsberegning



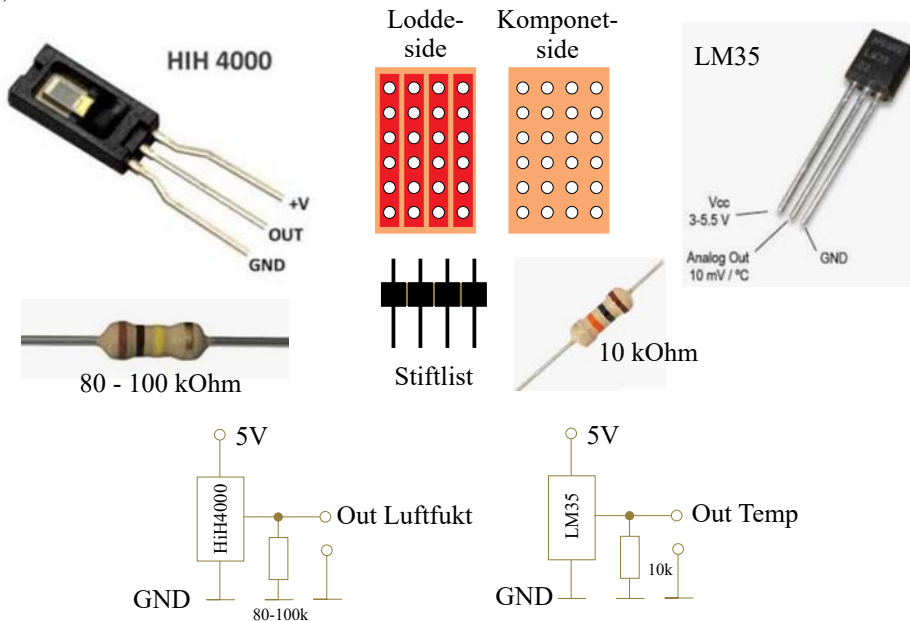


2 Montering, oppkobling og uttesting av luftfuktighet- og temperatursensor

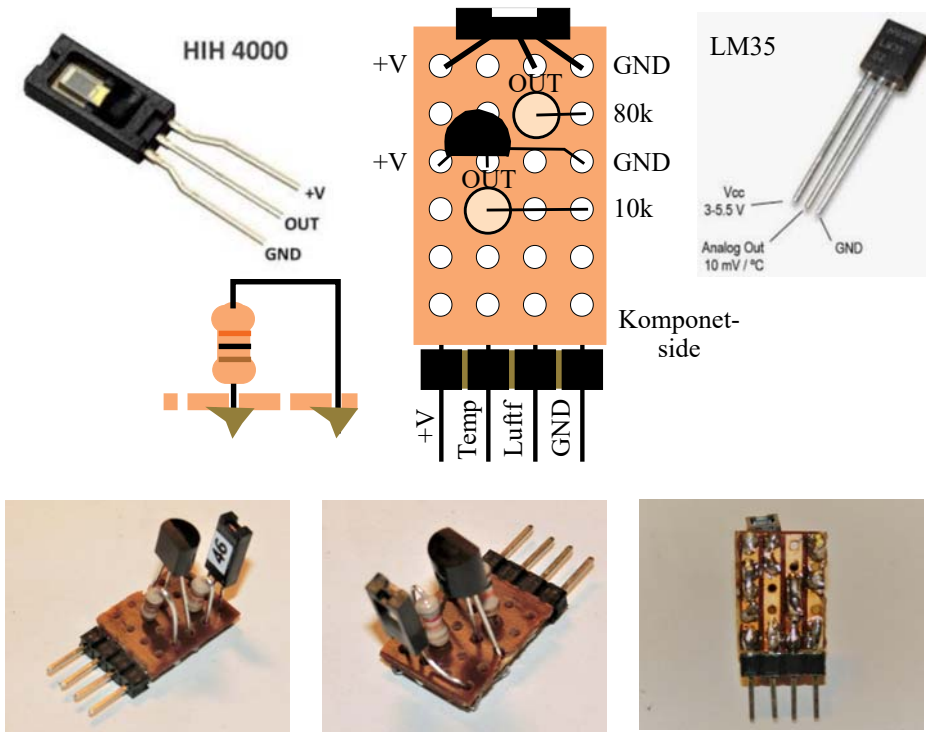
Temperatur og fuktighet er viktige parametere når man skal måle partikkeltetthet, da den ofte er tett forbundet med luftfuktigheten. Dessuten er relativ luftfuktighet sterkt forbundet med temperaturen.

2.1 Oppbygging av sensor for måling av luftfuktighet og temperatur

I denne oppgaven velger vi å kjøpe disse to separat og montere dem på et lite prototype-kort (vero board).



Monter komponentene som vist på figuren under:



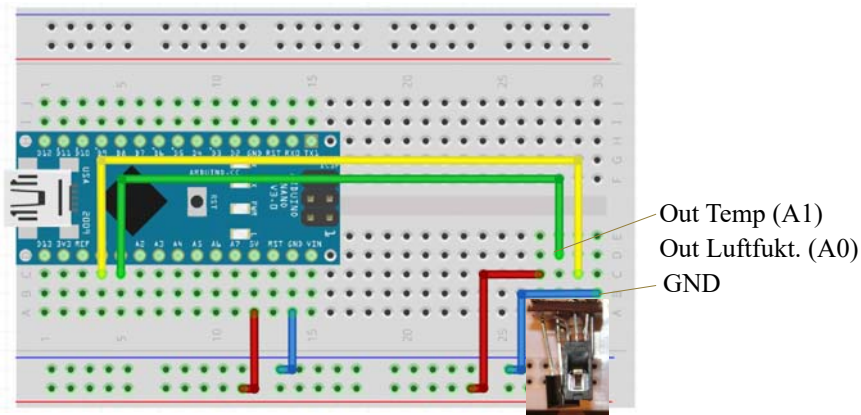
Komponentene monteres på komponentsiden og loddes på loddessiden der kobberstripene er.

- Pass på at LM35 og HiH4000 plasseres riktig vei og med beina i riktige hull
- Monter de to motstandene. Sett dem gjerne på høykant som vist på figuren over, sørg for at beina kommer i de riktige hullene og at motstandene kommer på riktig sted. Husk de har forskjellig verdi og må plasseres rett.
- Lodd beina til kobberbanene på undersiden. Pass på at det ikke kommer loddetinn mellom stripene.
- Monter stiftlisten på enden av kortet som vist. Lodd de korte stiftene til kobberstripene som vist på fotografiet over, nederst til høyre.
- Sjekk at loddningene er gode og at det ikke er kommet loddetinn mellom stripene (loddebroer).



2.2 Oppkobling på koblingsbrett

Dernest monterer vi vår temperatur- og luftfuktighetssensor på koblingsbrettet. Vi velger å sette den der den vil stå helt til slutt når alle komponentene er montert.



Vi legger merke til at A0 er koblet til måling av luftfuktighet og A1 er koblet til måling av temperatur.

2.3 Program for avlesing av temperatur og luftfuktighet

Vi skal nå laste opp et lite program for å teste ut de to sensorene. Videre skal vi legge inn omregningsformler slik at vi får ut verdiene i °C og relativ fuktighet i prosent.

Omregningsformlene for temperatur og luftfuktighet finner vi i databladet og er gjengitt her:

$$\text{Temperatur (}^\circ\text{C)} = T = V_{\text{Tout}} * 100 \quad (2.1)$$

$$\text{Relativ luftfuktighet (\%)} = (V_{\text{LFout}} - 0,16) \cdot (1,0546 - 0,00216 \cdot T) / 0,031 \quad (2.2)$$

V_{Tout} = Spenningen som leses av temperatursensoren

T = Temperaturen i grader C

V_{LFout} = Spenningen som leses av fuktighetssensoren

V_{Tout} = Spenningen som leses av temperatursensoren

Det er viktig å være klar over at når man leser av A0 og A1 så **får man ikke ut spenningen direkte**, men verdiene må regnes om fra et tall mellom 0 – 1023 til spenning i Volt. Spenningen beregnes på følgende måte:

$$\text{Avlest spenning} = (\text{Avlest verdi fra A0 eller 1} * 5.0 \text{ V}) / 1023 \quad (2.3)$$

Under er vist testprogrammet:

```
// Testprogram for uttesting av temperatursensor LM35 og
```

```

// Luftfuktighetssensor HiH4000.
// 05.03.20 Nils Kr. Rossing

int pinTemp = A1;
int pinFukt = A0;
float digTemp;      // Digitalt avlest temperatur
float digFukt;      // Digitalt avlest fuktighet
float temp;         // Beregnet temperatur i Celsius
float fukt;         // Beregnet relativ fuktighet i %
float tempCal=0.0;  // lineær kalibrering av temperatur
float fuktCal=0.0;  // lineær kalibrering av fukt

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  digTemp = analogRead(A1);
  digFukt = analogRead(A0);

  temp = digTemp*500.0/1023 + tempCal;      //LM35
  fukt = (digFukt/1023 - 0.16)/0.0062;
  fukt = fukt/(1.0546 -0.00216*temp)+fuktCal; // korrigerer fuktighetsmåling
med temp
// Skriver ut til monitoren
  Serial.print("Temp: ");
  Serial.print(temp);
  Serial.print("C");
  Serial.print(", Fukt: ");
  Serial.print(fukt,1);
  Serial.println("%");
  delay(1000);
}

```

Last inn programmet, åpne monitorvinduet og se om verdiene ser fornuftige ut.

NB! Ved opplasting av programmet til Arduino Nano velg under menyen *Verktøy/Prosesor/ATmega328P (old bootloader)*



Gjør en enkel test:

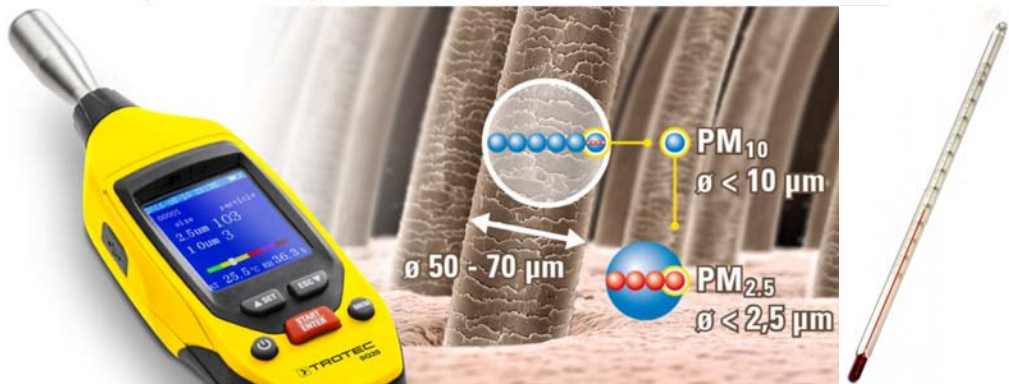
- Hold fingrene om temperatursensoren og se om temperaturen øker
- Pust fuktig luft på luftfuktighetssensoren og se om verdien endres.

Figuren til høyre viser en typisk utskrift til monitoren til Arduino IDE'en. Vi ser at verdien til luftfuktigheten stiger kraftig, fra ca. 18% til godt over 70 %. Dette skyldes et pust på sensoren. Målingene er tatt med 0,5 sek. mellomrom. Fuktighetssensoren reagerer fort og faller fort tilbake til normaltilstanden i rommet,

```
Temp: 25.39C, Fukt: 17.6%
Temp: 25.39C, Fukt: 18.3%
Temp: 25.39C, Fukt: 23.5%
Temp: 26.37C, Fukt: 37.0%
Temp: 23.93C, Fukt: 50.0%
Temp: 25.39C, Fukt: 58.9%
Temp: 25.88C, Fukt: 66.4%
Temp: 25.88C, Fukt: 70.8%
Temp: 25.39C, Fukt: 76.9%
Temp: 25.39C, Fukt: 77.7%
Temp: 24.41C, Fukt: 70.3%
Temp: 25.88C, Fukt: 63.7%
Temp: 25.39C, Fukt: 57.1%
Temp: 26.86C, Fukt: 50.4%
Temp: 24.90C, Fukt: 45.4%
Temp: 26.37C, Fukt: 41.1%
Temp: 23.93C, Fukt: 37.6%
Temp: 24.90C, Fukt: 34.6%
```

2.4 Kalibrering

Det er ikke uvanlig at det er avvik mellom de avleste verdiene og de "virkelige" verdiene. For å kunne korrigere for avvik må vi ha kalibrerte instrumenter, dvs. en luftfuktighetsmåler og et nøyaktig termometer. Dersom vi ikke har det, må vi nøye oss med å måle relative verdier. Bildet under viser en partikkelmåler som også måler fuktighet og temperatur (Trotec BQ20¹) og som kan brukes for kalibrering.



Når det gjelder å kalibrere temperatursensoren kan vi også komme langt med et vanlig termometer.

En enkel lineær kalibrering forutsetter en ren forskyvning langs skalaen og gjør at kalibreringen er enkel. Dvs. vi legger til eller trekker fra det avviket vi måler med det kalibrerte instrumentet, slik at vårt instrument og det kalibrerte instrumentet viser samme verdi.

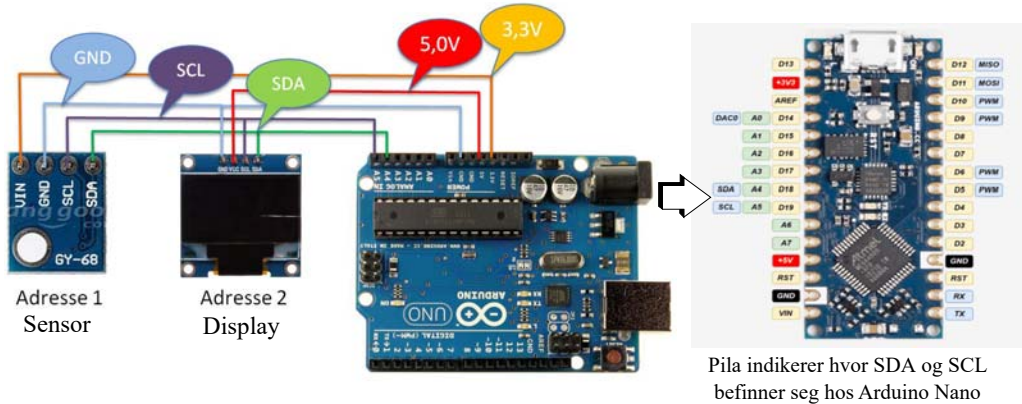
Det vil alltid være lurt å endre betingelsene og se om begge instrumentene følger hverandre.

1. <https://no.trotec.com/produkter-og-tjenester/maaleinstrumenter/luftkvalitet/partikkelteller/bq20-partikkelmaaler/>



3 Montering av display

Til dette formålet skal vi bruke et lite ensfarget OLED display hvor verdiene lastes inn fra Arduino'en til displayet langs en seriebuss, en såkalt I²C-buss. Siden signalene som sendes på bussen er adressert så kan bussen, forbinde mange ulike sensorer som vist på figuren under. Her bruker vi den kun til displayet.



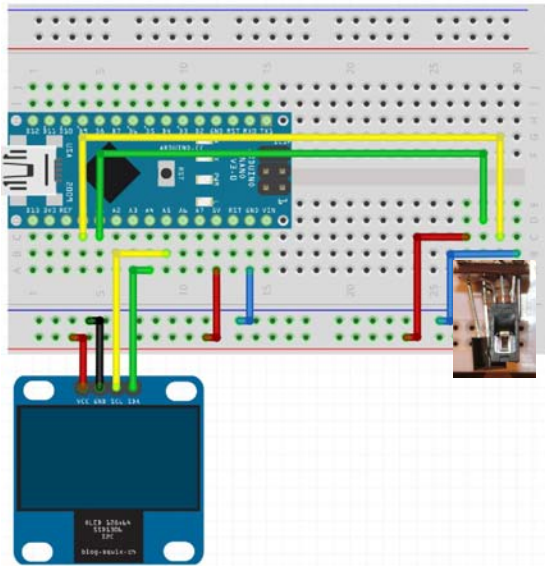
3.1 Oppkobling av displayet

For å koble opp displayet må man sannsynligvis bruke jumpere med han/hun kontakter slik at man kan koble ledninger mellom koblingsbrettet og displayet. Årsaken er at vi har valgt å montere displayet på utsiden av boksen i det ferdige instrumentet, derfor er det naturlig å montere ledninger mellom koblingsbrettet og displayet. Det er selvfølgelig fullt mulig å velge andre løsninger.

3.2 Programmet

Neste trinn er å skrive verdiene for temperatur og fuktighet til displayet i tillegg til monitoren på PC'en. For å få til dette må vi gjøre følgende:

1. Installere biblioteker for å bruke I²C-bussen og for å bruke displayet.
2. Deklarere og initialisere displayet
3. Skrive tekst og verdier til displayet.



3.2.1 Installasjon av biblioteker

For å kunne kommunisere med displayet (SSD1306), må vi installere to biblioteker. Disse bibliotekene er komprimert i zip-filer og kan lastes ned fra:

<https://www.ntnu.no/skolelab/bla-hefteserie>

under fanen: *Science Camp 2019 (vgs) - veiledningshefte*

Adafruit_SSD1306-master.zip (Display bibliotek (SSD1306))

Adafruit-GFX-Library-master.zip (Grafisk bibliotek (GFX.h))

Lagre bibliotekene i nedlastingskatalogen eller et annet sted der du finner dem igjen.

Bibliotekene installeres på følgende måte:

Åpen Arduino-editoren



Velg: *Skisse*

Velg: *Inkluder bibliotek*

Velg: *Legg til .zip bibliotek* (øverst i nedtrekksmenyen)

Finn biblioteket i katalogen Nedlasting

Velg: *Open*

... og biblioteket installeres

Gjenta det samme for begge bibliotekene.

3.2.2 Initiering av display

For at vi skal kunne skrive til displayet må vi:

- Inkludere bibliotekene i koden vår

For å kunne ta i bruk kommandoer (funksjoner) fra biblioteket må disse inkluderes øverst i koden som vist under:

```
// Inkludering av biblioteker  
#include <SPI.h>  
#include <Wire.h>  
#include <Adafruit_GFX.h>  
#include <Adafruit_SSD1306.h>
```



- Definere displayet som en klasse

Akkurat som vi må deklare variabler med en gitt type må vi deklare displayet vårt som en gitt *klasse* eller *instans*. Vi kaller det for enkelthet skyld kun for *display*. Dette gjør vi før `setup()`-funksjonen:

```
// Deklarasjon av klasser
Adafruit_SSD1306 display(4);
```

Hvorfor det står et 4-tall i parentes er ikke brakt på det rene.

- Initialisere displayet i `setup()` funksjon

Dernest må vi initialisere displayet vårt, f.eks. må vi fortelle biblioteket hvilken adresse displayet vårt har. Adressen er gjerne satt ved fabrikken, men kan også endres på mange komponenter ved å lodde over en eller flere forbindelse på baksiden av displayet. Dette er også tilfelle for vårt display som har to mulige adresser. Det betyr at vi kan koble to displayer på samme I²C-bussen og sende individuell informasjon til hver av dem.

```
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
```

Her er adressen angitt som 0x3C. Hva SSD1306_SWITCHCAPVCC betyr er ikke brakt på det rene.

3.2.3 Skrive til displayet

Når vi nå skal skrive til displayet så bruker vi et sett med funksjoner hentet fra biblioteket. Disse kommandoene skal plasseres i `loop()`-funksjonen og vil skrives ut for hver runde programmet gjør i `loop`'en.

Under er gjengitt et typisk eksempel på hvordan vi kan skrive ut tekst til displayet for temperatur og fuktighet:

```
// Klargjør display for utskrift
display.clearDisplay(); // Slett informasjon på display
display.setTextSize(1); // Sett størrelse på tekst
display.setTextColor(WHITE); // Hvit tekst på sort bakgrunn

// Skriver ut luftfuktighet med en desimaler
display.setCursor(0,0); // Plasser markør øverst til venstre
display.print("Luftfukt: "); // Skriv "Luftfukt".
display.print(fukt,1); // Skriv verdien til luftfukt med en desimaler
display.println(" %"); // Skriv benevnning %

// Skriver ut temperatur med en desimal
display.setCursor(0,10); // Flytt markør til andre linje
display.print("Temp.: "); // Skriv "Temp.:", Temperatur
display.print(temp,1); // Skriv den målte temperaturen i ...
```

```

display.println(" C");          // ... grader C

display.display();             // Send data over til displayet
delay(500);                    // Vent i 500 msek

```

Legg merke til at variabelen *temp* inneholder temperaturen i grader Celcius og variabelen *fukt* inneholder relativ luftfuktighet i prosent.

3.2.4 Det komplette programmet ser da ut som følger:

```

// Inkludering av biblioteker
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

int pinTemp = A1;
int pinFukt = A0;

float digTemp;           // Digitalt avlest temperatur
float digFukt;           // Digitalt avlest fuktighet
float temp;              // Beregnet temperatur i Celsius
float fukt;              // Beregnet relativ fuktighet i %
float tempCal=0.0;      // lineær kalibrering av temperatur
float fuktCal=0.0;     // lineær kalibrering av fukt

// Deklarasjon av klasser
Adafruit_SSD1306 display(4); // Deklarasjon av display som klassen
Adafruit_SSD1306

void setup()
{
  Serial.begin(9600);

  display.begin(SSD1306_SWITCHCAPVCC, 0x3C); //Initialiser displayet med adres-
  sen 0x3C
}

void loop()
{
  digTemp = analogRead(A1);
  digFukt = analogRead(A0);

```



```
temp = digTemp*500.0/1024 + tempCal;           //LM35
fukt = (digFukt/1024 - 0.16)/0.0062;
fukt = fukt/(1.0546 - 0.00216*temp)+fuktCal;   // korrigerer fuktighets-
måling med temp

// Skriv til monitoren
Serial.print("Temp: ");
Serial.print(temp);
Serial.print("C");
Serial.print(", Fukt: ");
Serial.print(fukt,1);
Serial.println("%");

// Skriv til displayet
// Klargjør display for utskrift
display.clearDisplay();           // Slett informasjon på display
display.setTextSize(1);          // Sett størrelse på tekst
display.setTextColor(WHITE);     // Hvit tekst på sort bakgrunn

// Skriver ut luftfuktighet med en desimaler
display.setCursor(0,0);          // Plasser markør øverst til venstre
display.print("Luftfukt: ");     // Skriv "Luftfukt."
display.print(fukt,1);           // Skriv verdien til luftfukt med en desimaler
display.println(" %");           // Skriv benevning %

// Skriver ut temperatur med en desimal
display.setCursor(0,10);         // Flytt markør til andre linje
display.print("Temp.: ");        // Skriv "Temp.:", Temperatur
display.print(temp,1);           // Skriv den målte temperaturen i ...
display.println(" C");           // ... grader C

display.display();
delay(500);
}
```




4 Montering av SD-kort

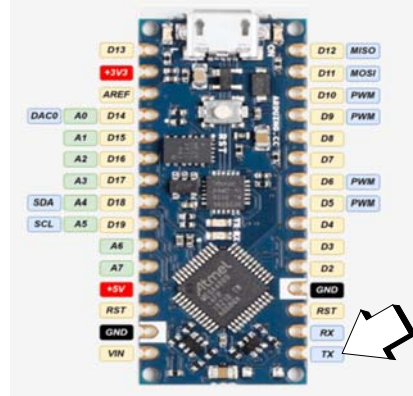
Dersom vi ønsker å samle data over tid og ikke har anledning til å koble instrumentet opp mot internett så er lagring av data på et SD-kort et godt alternativ. I avsnittet skal vi se hvordan vi kan koble opp og skrive til SD-kortet.

Vi velger OpenLog som er in datalogger som skriver til mini SD-kort, og som er lett å implementere siden kommuniserer på Rx/Tx linjene, de samme som brukes til å laste opp programmet til Arduino'en og de samme som brukes til å skrive til monitoren.

Siden vi ikke ønsker å lese fra SD-kortet er det kun nødvendig å koble oss opp til Tx-linjen hos Arduino Nano'en som er det samme som dataport D1 (se pila på figuren til høyre).

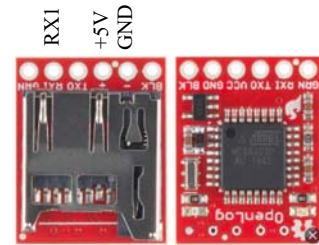
Dvs. at når vi skriver til monitoren så skriver vi også til SD-kortet. Hver gang vi slår på strømmen eller trykker reset så opprettes det en ny fil på kortet.

Fila er ei vanlig tekstfil med det formatet som våre Serial.print-kommandoer spesifiserer. Det er heller ikke nødvendig med noen ekstra biblioteker eller initiering av kretsen.

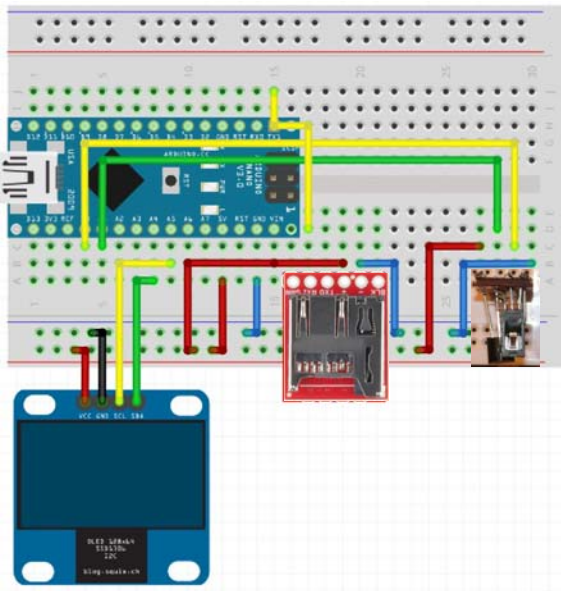


4.1 Oppkobling av SD-kortet

Merk at vi valgt å montere OpenLog SD-kort-leseren slik at selve holderen for SD-kortet ligger på oversiden av kretskortet når den er montert i kobling Brettet. Som det framgår fra monte-



ringsanvisningen under så skal kretsen bare ha +5V, jord (GND) og en forbindelse mellom TxI på Arduino Nano'en og pinnen kaldt RxI på OpenLog, noe som er logisk: Ut fra Arduino Nano TxI (Tx – Transmitt) og inn på OpenLogs RxI (Rx – Receive).



Når OpenLog er tilkoblet skal det blinke et gult eller blått lys på undersiden av kortet. Når det skrives til kortet.

4.2 Programmet

Det fine er at vi ikke trenger å gjøre noe med programmet da kommandoene som skriver til monitoren er de samme som skriver til OpenLog. Dataene på SD-kortet er også formatert på samme måte som i monitoren.

```
// Skriv til monitoren
Serial.print("Temp: ");
Serial.print(temp,1);
Serial.print("C");
Serial.print(", Fukt: ");
Serial.print(fukt,1);
Serial.println("%");
```

Det er på dette tidspunktet fornuftig å tenke på hvordan vi vil at teksten i datafilen skal se ut når vi f.eks. skal hente den inn i Excel. Da kan det være nyttig å droppe tekst og kun ha verdiene skilt med komma eller semikolon.

```
Temp: 25.39C, Fukt: 18.3%
Temp: 25.39C, Fukt: 18.3%
Temp: 25.39C, Fukt: 23.5%
Temp: 26.37C, Fukt: 37.0%
Temp: 23.93C, Fukt: 50.0%
Temp: 25.39C, Fukt: 58.9%
Temp: 25.88C, Fukt: 66.4%
Temp: 25.88C, Fukt: 70.8%
Temp: 25.39C, Fukt: 76.9%
Temp: 25.39C, Fukt: 77.7%
Temp: 24.41C, Fukt: 70.3%
Temp: 25.88C, Fukt: 63.7%
Temp: 25.39C, Fukt: 57.1%
Temp: 26.86C, Fukt: 50.4%
Temp: 24.90C, Fukt: 45.4%
Temp: 26.37C, Fukt: 41.1%
Temp: 23.93C, Fukt: 37.6%
Temp: 24.90C, Fukt: 34.6%
```



```
// Skriv til fil
Serial.print((float)millis()/1000,1);
Serial.print(";");
Serial.print(temp,1);
Serial.print(";");
Serial.print(fukt,1);
Serial.println(";");
```

Det kan også være lurt å skrive inn en tidsangivelse i begynnelsen av hver linje slik at vi kan tidfeste dataene som vist øverst i listingen over. funksjonen angir millisekunder fra vi startet programmet. Dersom vi deler denne på 1000 får vi tiden angitt i sekunder. Tallet 1 angir at det skal skrives med en desimal. Det er viktig å skrive inn (float) foran funksjonen slik at resultatet blir et desimaltall og ikke en heltall som det ellers ville ha blitt.

Figuren til høyre viser tidsangivelsen til venstre.

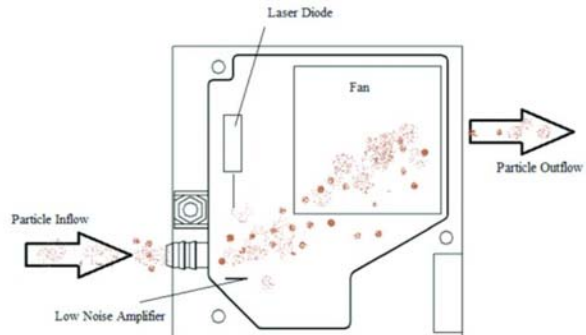
```
0.0;24.9;20.3;
0.7;24.4;19.5;
1.4;24.4;20.5;
2.1;25.4;20.0;
2.7;26.4;20.2;
3.4;25.4;20.8;
4.1;24.9;20.2;
4.8;24.9;19.5;
5.5;25.4;19.7;
6.1;24.4;19.0;
6.8;24.9;19.2;
7.5;23.4;18.7;
8.2;24.9;18.6;
8.8;26.4;18.3;
9.5;25.4;18.0;
10.2;26.9;17.6;
10.9;25.4;18.0;
11.5;25.9;18.3;
```




5 Montering av partikkelsensor

En partikkelmåler består av et mørkt kammer og en vifte som drar inn luft i kammeret gjennom et lite rør i siden. En laser belyser luftprøven som fyller kammeret. En lyssensor er plassert til siden for laserstrålen slik at den kun fanger opp det lyset som spres på grunn av partiklene i kammeret. Dette signalet forsterkes, digitaliseres og regnes om til en partikkelkonsentrasjon $\mu\text{g}/\text{m}^3$.

På figuren under ser vi for- og baksiden av sensoren.

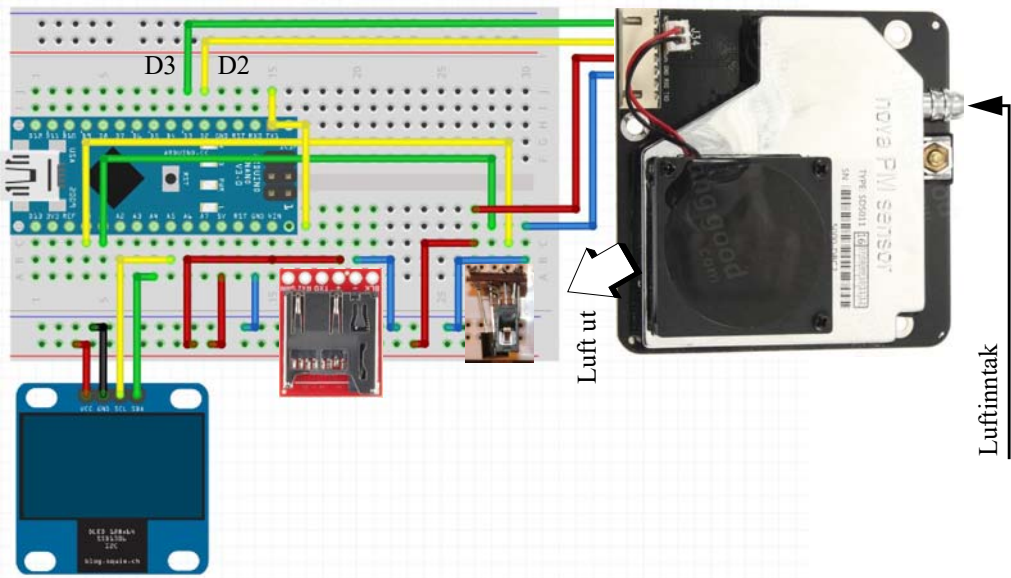


For enkelthetsskyld har vi valgt å lodde på jumperne på de fire terminalene TXD (grønn), RXD (gul), GND (sort) og Vcc (+5V, rød). Dette gjør det enkelt å koble sensoren til koblingsbrettet. Klipp jumperne i passe lengde, avisoler enden og lodd til terminalene. Litt flytelim over terminalene kan være lurt for å stabilisere jumper-ledningene.

5.1 Oppkobling av partikkelsensoren

Figuren under viser hvordan vi kan koble opp partikkelsensoren. Det er lurt å plassere den slik at luftinntaket vender ut foran på boksen, mens lufta blåses ut på siden av vifta som er plassert på

toppen av sensoren (se piler på figuren under).



Legg merke til at TXD og RXD kobles til henholdsvis D3 og D2 på Arduino Nano.

5.2 Programmet

La oss se hvordan vi kan inkludere partikkelsensoren i programmet. Metoden ligner mye på den vi brukte når vi monterte displayet:

1. Installer biblioteket for å bruke partikkelsensoren.
2. Deklarer og initialiser partikkelsensoren
3. Les av verdier fra partikkelsensoren
4. Skrive tekst og verdier til displayet og til monitor/fil.

5.2.1 Installasjon av biblioteker

For å kunne kommunisere med partikkelsensoren (SDS011), må vi installere et bibliotek til. Også dette biblioteket er komprimert i en zip-fil og kan lastes ned fra:

<https://www.ntnu.no/skolelab/bla-hefteserie>

under fanen: *Science Camp 2019 (vgs) - veiledningshefte*

Støvsensor bibliotek (SDS011)

Lagre biblioteket i nedlastingskatalogen eller et annet sted der du finner den igjen.



Biblioteket installeres som sist på følgende måte:

Åpen Arduino-editoren



Velg: *Skisse*

Velg: *Inkluder bibliotek*

Velg: *Legg til .zip bibliotek* (øverst i nedtrekksmenyen)

Finn biblioteket i katalogen Nedlasting

Velg: *Open*

... og biblioteket installeres

5.2.2 Inkludering og initialisering av biblioteket og partikkelsensoren

Det første som må gjøres er å inkludere biblioteket helt først i programmet:

```
#include <SDS011.h>
```

Dernest må vi fortelle hvilke to porter på Arduino Nano'en dataene skal overføres via. I vårt tilfelle er det portene *D2* (*PM_TX*) og *D3* (*PM_RX*) som defineres som to konstanter som vi skal bruke når vi senere skal initialisere partikkelsensoren. Dernest deklarerer instansen lik *sds* av klassen *SDS011*, også denne før *setup()*-funksjonen:

```
#define PM_TX 2
#define PM_RX 3
SDS011 sds;
```

Så kan partikkelsensoren initialiseres, noe som gjøres i *setup()*-funksjonen:

```
sds.begin(PM_TX, PM_RX);
```

Vi legger merke til at de to portene *PM_TX* og *PM_RX* legges inn som argumenter i initialiseringen.

Inne i *loop()*-funksjonen definerer vi to variabler (*pm25* og *pm10*) som skal holde avlesningen fra partikkelsensoren. En for 2,5 μm og en for 10 μm , begge deklarerer som desimaltall (float). I tillegg deklarerer en heltallsvariabel *error*, som tar vare på feilmeldingen som kan oppstå når vi forsøker å lese verdier fra partikkelsensoren (*sds.read()*). Programmet forsøker å lese partikkelsensoren helt til feilmeldingen forsvinner. Dette gjøres ved hjelp av en *do{sds.read()} while(<betingelse>);* helt til betingelsen ikke lenger er oppfylt, dvs. *error* = 0 og verdiene er lest feilfritt.

```
float pm25, pm10;
```

```

int error;
do
{
    error = sds.read(&pm25, &pm10);
} while (error != 0);    // Fortsett å les av støvmåleren helt til gyldige data

```

Så skrives verdiene av partikkelkonsentrasjonene ut til monitoren:

```

Serial.print("pm2,5: ");
Serial.print(pm25);           // Skriv ut støvkonsentrasjon 2,5 um
Serial.print("um");
Serial.print(", ");
Serial.print("pm10: ");
Serial.print(pm10);          // Skriv ut støvkonsentrasjon 10 um
Serial.print("um");
Serial.println(", ");

```

til SD-kortet:

```

Serial.print(pm25);           // Skriv ut støvkonsentrasjon 2,5 um
Serial.print(";");
Serial.print(pm10);          // Skriv ut støvkonsentrasjon 10 um
Serial.print(";");

```

og til displayet:

```

display.setCursor(0,20);     // Flytt markør til neste linje
display.print("pm2.5: ");    // ... ug/m3
display.print(pm25,1);       // Skriv den målte pm 2,5 u
display.print(" pm10: ");    // ... ug/m3
display.println(pm10,1);     // Skriv den målte pm 10 u

```

Legger vi inn dette i programmet vil vi få en utskrift i monitoren som vist på figuren under:

```

0.3, Temp: 22.46C, Fukt: 32.3%, pm2,5: 2.00um, pm10: 5.20um,
1.3, Temp: 22.95C, Fukt: 31.5%, pm2,5: 2.00um, pm10: 5.10um,
2.3, Temp: 22.95C, Fukt: 32.2%, pm2,5: 2.00um, pm10: 5.30um,
3.3, Temp: 22.95C, Fukt: 32.0%, pm2,5: 2.10um, pm10: 5.50um,
4.3, Temp: 22.95C, Fukt: 32.3%, pm2,5: 2.10um, pm10: 5.50um,
5.3, Temp: 22.95C, Fukt: 32.6%, pm2,5: 2.10um, pm10: 5.30um,
6.3, Temp: 22.46C, Fukt: 32.0%, pm2,5: 2.20um, pm10: 5.30um,
7.3, Temp: 22.46C, Fukt: 31.5%, pm2,5: 2.10um, pm10: 5.20um,
8.3, Temp: 21.97C, Fukt: 31.9%, pm2,5: 2.10um, pm10: 5.20um,
9.3, Temp: 22.46C, Fukt: 32.0%, pm2,5: 2.10um, pm10: 5.40um,
10.3, Temp: 22.46C, Fukt: 31.8%, pm2,5: 2.10um, pm10: 5.30um,
11.3, Temp: 22.95C, Fukt: 31.5%, pm2,5: 2.10um, pm10: 5.10um,
12.3, Temp: 23.44C, Fukt: 31.9%, pm2,5: 2.10um, pm10: 5.10um,
13.3, Temp: 21.97C, Fukt: 31.8%, pm2,5: 2.10um, pm10: 5.10um,

```




På SD-kortet vil det se slik ut.

```
0.7;22.9;32.3;2.20;6.10;  
1.7;22.9;31.7;2.20;6.10;  
2.7;23.9;32.8;2.20;5.90;  
3.7;22.9;32.2;2.10;5.80;  
4.7;22.5;31.6;2.20;5.90;  
5.7;23.9;31.3;2.00;5.40;  
6.7;22.9;31.8;2.10;5.50;  
7.7;22.9;31.7;2.00;5.20;  
8.7;22.9;32.3;2.00;5.30;  
9.7;22.9;31.7;2.00;5.10;  
10.7;22.9;32.0;2.00;5.00;  
11.7;22.9;31.5;2.00;5.10;  
12.7;22.5;32.0;2.00;5.10;
```

5.2.3 Komplettest-programmet for partikkelmåler:

Det komplette test-programmet for partikkelmåleren, luftfuktighets- og temperatursensoren blir da seende slik ut:

```
// Inkludering av biblioteker  
#include <SPI.h>  
#include <Wire.h>  
#include <Adafruit_GFX.h>  
#include <Adafruit_SSD1306.h>  
#include <SDS011.h>  
  
int pinTemp = A1;  
int pinFukt = A0;  
  
float digTemp; // Digitalt avlest temperatur  
float digFukt; // Digitalt avlest fuktighet  
float temp; // Beregnet temperatur i Celsius  
float fukt; // Beregnet relativ fuktighet i %  
float tempCal=0.0; // lineær kalibrering av temperatur  
float fuktCal=0.0; // lineær kalibrering av fukt  
  
// Deklarasjon av klasser  
// Displayet  
Adafruit_SSD1306 display(4); // Deklarasjon av display som klassen  
Adafruit_SSD1306  
  
// Partikkelsensoren  
#define PM_TX 2  
#define PM_RX 3
```

```

SDS011 sds;

void setup()
{
  Serial.begin(9600);
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C); //Initialiser displayet med adres-
sen 0x3C
  sds.begin(PM_TX, PM_RX); // Initialisering av partikkelsensoren
}

void loop()
{
  // Les av temperatur og fuktighet
  digTemp = analogRead(A1);
  digFukt = analogRead(A0);

  temp = digTemp*500.0/1024 + tempCal; //LM35
  fukt = (digFukt/1024 - 0.16)/0.0062;
  fukt = fukt/(1.0546 -0.00216*temp)+fuktCal; // korrigerer fuktighet-
småling med temp

  // Les av partikkeltetthet
  float pm25, pm10;
  int error;
  do
  {
    error = sds.read(&pm25, &pm10);
  } while (error != 0); // Fortsett å les av støvmåleren helt til gyldige data

  // Velg den utskriften som passer: Til monitor eller til fil, kommenter ut
den som ikke passer

  // Skriv til monitoren

  Serial.print((float)millis()/1000,1); // Angir tid fra start i
sekunder med en desimal
  Serial.print(", ");
  Serial.print("Temp: ");
  Serial.print(temp);
  Serial.print("C");

```



```
Serial.print(", ");
Serial.print("Fukt: ");
Serial.print(fukt,1);
Serial.print("%");
Serial.print(", ");
Serial.print("pm2,5: ");
Serial.print(pm25); // Skriv ut støvkonsentrasjon 2,5 um
Serial.print("um");
Serial.print(", ");
Serial.print("pm10: ");
Serial.print(pm10); // Skriv ut støvkonsentrasjon 10 um
Serial.print("um");
Serial.println(", ");

/*
// Skriv til fil
Serial.print((float)millis()/1000,1);
Serial.print(";");
Serial.print(temp,1);
Serial.print(";");
Serial.print(fukt,1);
Serial.print(";");
Serial.print(pm25); // Skriv ut støvkonsentrasjon 2,5 um
Serial.print(";");
Serial.print(pm10); // Skriv ut støvkonsentrasjon 10 um
Serial.println(";");
*/

// Skriv til displayet

// Klargjør display for utskrift
display.clearDisplay(); // Slett informasjon på display
display.setTextSize(1); // Sett størrelse på tekst
display.setTextColor(WHITE); // Hvit tekst på sort bakgrunn

// Skriver ut luftfuktighet med en desimaler
display.setCursor(0,0); // Plasser markør øverst til venstre
display.print("Luftfukt: "); // Skriv "Luftfukt."
```

```

display.print(fukt,1);          // Skriv verdien til luftfukt med en desimaler
display.println(" %");        // Skriv benevning %

// Skriver ut temperatur med en desimal
display.setCursor(0,10);      // Flytt markør til andre linje
display.print("Temp.: ");     // Skriv "Temp.:", Temperatur
display.print(temp,1);        // Skriv den målte temperaturen i ...
display.println(" C");        // ... grader C

// Skriver ut partikkelkonsentrasjonen
display.setCursor(0,20);      // Flytt markør til neste linje
display.print("pm2.5: ");     // ... ug/m3
display.print(pm25,1);        // Skriv den målte pm 2,5 u
display.print(" pm10: ");     // ... ug/m3
display.println(pm10,1);      // Skriv den målte pm 10 u

display.display();
delay(500);
}

```

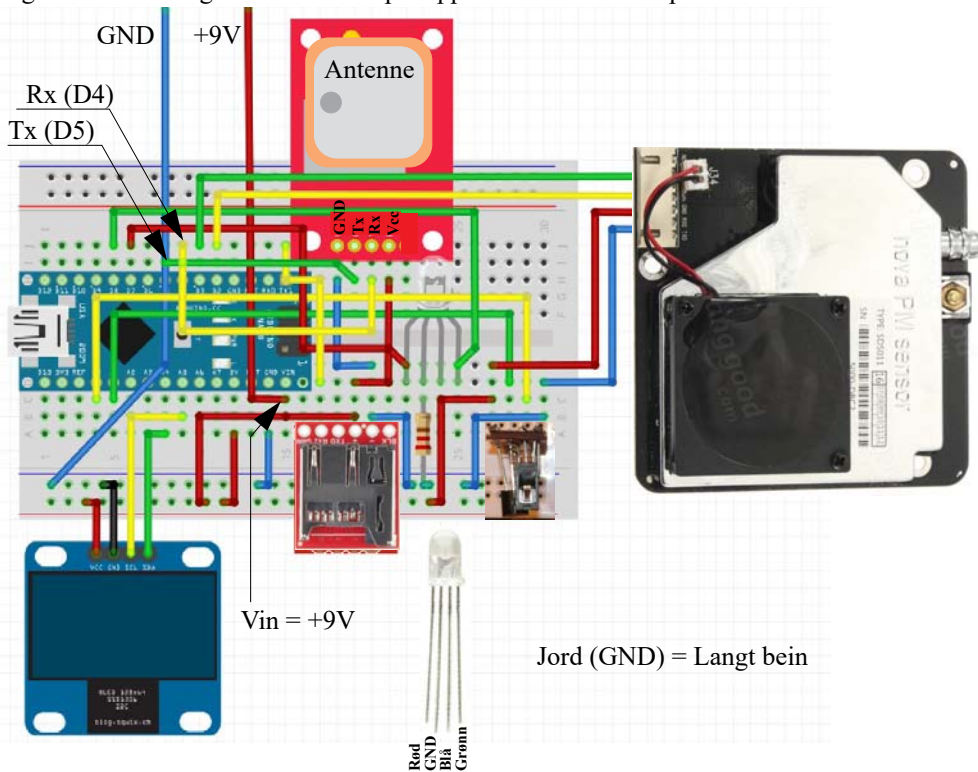


6 Inkludering av GPS-mottakeren

GY-NEO6MV2 er en GPS som passer godt for bruk med Arduino. Den er også billig² og har en relativt “stor” antenne i forhold til flere lignende modeller. Dette gjør at den er lettere å holde låst til satellitter mens mottakeren er i bevegelse. Den krever 5V og er dermed lett å koble til en Arduino UNO eller Nano som i vårt tilfelle.

6.1 Oppkobling av GPS-mottakeren

Figuren under viser hvordan vi kan koble opp GPS-mottakeren. Det er lurt å plassere antenne mest mulig fritt. Vi har valgt å montere den på toppen av RF-modulen på “break out” kortet.



GPS-modulen kommuniserer ved hjelp av en Rx og en Tx linje, disse kobles til to digitale porter som defineres som kommunikasjonskanal i programmet. I dette tilfellet har vi valgt D4 (Rx) og D5 (Tx).

Figuren viser også de to ledningene som går til batteriet GND (blå) og +9V (rød). Merk at +9V er tilknyttet V_{in} på Arduino Nano'en. V_{in} blir regulert ned til +5,0 V på kortet.

2. https://www.kultogbillig.no/index.php?_route_=GY-NEO6MV2-Flight-Controller-GPS-Modul-For-Arduino&search=GPS Pris.: 119,- inkl. MVA

6.2 Programmet

La oss se hvordan vi kan bygge opp programmet som leser GPS-mottakeren. Denne gangen velger vi først kun å konsentrere oss om GPS-mottakeren for deretter, når vi har testet den, og inkludere den i resten av programmet.

Metoden ligner mye på den vi brukte når vi monterte displayet og partikkelmåleren:

1. Installer biblioteket for å bruke GPS-mottakeren.
2. Deklarer og initialiser GPS-mottakeren
3. Les av verdier fra GPS-mottakeren
4. Skriv tekst og verdier til displayet og til monitor/fil.

6.2.1 Installasjon av bibliotek

For å kunne kommunisere med GPS-mottakeren (GY-NEO6M), må vi installere et bibliotek til. Også dette biblioteket er komprimert i en zip-fil og kan lastes ned fra:

<https://www.ntnu.no/skolelab/bla-hefteserie>

under fanen: *Science Camp 2019 (vgs) - veiledningshefte*

GPS++ bibliotek (.zip)

Lagre biblioteket i nedlastingskatalogen eller et annet sted der du finner den igjen.

Biblioteket installeres som sist på følgende måte:

Åpen Arduino-editoren



Velg: *Skisse*

Velg: *Inkluder bibliotek*

Velg: *Legg til .zip bibliotek* (øverst i nedtrekksmenyen)

Finn biblioteket i katalogen Nedlasting

Velg: *Open ...* og biblioteket installeres

6.2.2 Inkludering og initialisering av biblioteket og GPS-mottakeren

Før setup()-funksjonen:

Det første som må gjøres er å inkludere to biblioteker helt først i programmet:



```
#include <SoftwareSerial.h> //  
#include <TinyGPS++.h>
```

Vi har behov for to biblioteker, både det som angår selve GPS-mottakeren, men også et bibliotek som gjør det mulig å sende over informasjon på seriell form via generelle digitale porter.

Dernest må vi fortelle hvilke to porter på Arduino Nano'en dataene skal overføres via, i vårt tilfelle er det portene *D4* (som går til *Rx på GPS'en*) og *D5* (som går til *Tx på GPS'en*) som defineres som to konstanter som vi bruker når vi senere skal initialisere GPS-mottakeren³:

```
#define GPS_RX 5  
#define GPS_TX 4
```

Dernest deklarerer instansen *gpsCom* av klassen *SoftwareSerial* og instansen *gps* av klassen *TinyGPSPlus*, begge før *setup()*-funksjonen:

```
SoftwareSerial gpsCom(GPS_RX, GPS_TX);  
TinyGPSPlus gps;
```

Vi legger merke til at de to portene *GPS_RX* og *GPS_TX* legges inn som argumenter i deklarasjonen av *gpsCom*.

Videre deklarerer hvilke digitale porter lysdiodene kobles til:

```
#define LED_RED 7  
#define LED_GREEN 8
```

Her ser vi at den røde lysdioden er koblet til *D7* og den grønne til *D8*. Legg også merke til at etter definisjonen av konstanter skal det ikke være semikolon.

I *setup()*-funksjonen:

Innenfor *setup()*-funksjonen må vi initialisere seriekommunikasjon, dette gjøres med følgende kommando som forteller hvor fort kommunikasjon skal gå, her er datahastigheten satt til 9600 baud (bit/sek):

```
gpsCom.begin(9600);
```

Vi må heller ikke glemme at vi må fortelle port *D7* og *D8* at de skal være utganger siden de skal slå av og på lysdiodene som indikerer at GPS-mottakeren er i kontakt med satellittene (grønn) eller ikke (rød):

```
pinMode(LED_RED, OUTPUT);  
pinMode(LED_GREEN, OUTPUT);
```

I *loop()*-funksjonen:

Innenfor *loop()*-funksjonen skjer innhenting av data fra GPS-mottakerne og utskrift av data. I vårt tilfelle handler det om lengdegrad, breddegrad og høyde. Flere andre parametere er også tilgjengelig og kan hentes inn, men foreløpig nøyer vi oss med disse:

3. Her er det lett å bli forvirret men *GPS_RX (5)* er den pinnen på Noano'en som mottar data fra *Tx på GPS'en*, og *GPS_TX (4)* er den pinnen på Nano'en som sender dataen til *Rx på GPS'en*, *Rx* kobles til *Tx* og *Tx* kobles til *Rx* hvilket er logisk nok.

```

////////// Les fra GPS-enhet //////////

if(gpsOn)          // gpsOn=True Leser GPS-data
{
    // For testing innendørs kan GPS slås av dvs. gpsOn=False
    gpsCom.listen();
    bool gpsEncodeComplete = false;

    do
    {
        if (!gpsCom.available()) // Ingen GPS-data tilgjengelig. Les på nytt
        {
            continue;
        }
        gpsEncodeComplete = gps.encode(gpsCom.read());
        if (!gpsEncodeComplete) // Ufullstendig GPS-data tilgjengelig. Les på nytt
        {
            continue;
        }
    } while (!gpsEncodeComplete); // Gjenta lesning til gyldige data oppnås

    bool gpsValid = gps.location.isValid();
    bool gpsUpdated = gps.location.isUpdated();
    bool isUseful = gpsValid && gpsUpdated;

    if (!isUseful) // Ugyldige data tenn rød lysdiode
    {
        digitalWrite(LED_RED, HIGH);
        delay(500);
        digitalWrite(LED_RED, LOW);
        return;
    }
    else // Gyldige data tenn grønn lysdiode
    {
        digitalWrite(LED_GREEN, HIGH);
        delay(500);
        digitalWrite(LED_GREEN, LOW);
    }
}

```




Hva betyr kodelinjene over?

Dette kan se ganske komplisert ut, men det er ikke så vanskelig dersom vi deler opp koden i biter. Det første vi har gjort er å legge inn en test (*if(gpsOn)*) som omslutter hele GPS-koden:

```
if (gpsOn)
{
    .... //kode
}
```

Denne er lagt inn for å kunne hoppe over lesingen fra GPS-mottakerne dersom vi befinner oss innendørs. Innendørs har GPS for liten dekning slik at ingen GPS-data er å oppdrive. I dette tilfellet vil programmet strande her og aldri komme videre. Noen ganger er det derfor lurt å ha mulighet til å hoppe over denne delen av programmet under uttesting. *gpsOn* settes under deklarasjonen av variabelen før *setup()*-funksjonen. Dersom:

- *gpsOn = true* (1) så søker GPS-mottakeren etter data fra satellitter, men dersom ...
- *gpsOn = false* (0) så hopper programmet over hele data innhentingen

Merke at konstantene *true* og *false* skjuler verdiene 1 og 0.

La oss se hva som befinner seg innenfor den omsluttende betingelsen.

```
gpsCom.listen();
```

Kommandoen *gpsCom.listen()*; etterspør informasjon fra GPS-mottakeren. Fra et slikt kall kommer det mye informasjon som lagres i variabler skjult av GPS-biblioteket. Disse vil bli hentet inn etter behov.

```
bool gpsEncodeComplete = false;
```

Her deklarerer en boolsk variabel, dvs. en variabel som kan være sann (*true*) eller usann (*false*). Navnet indikerer at det handler om de dekodete GPS-dataene er komplette eller ikke.

Så følger en

```
do
{
    ...
} while (!gpsEncodeComplete);
```

Programmet vil fortsette å være i *do - while*-sløyfa så lenge betingelsen i parenteser etter *while* er oppfylt. Denne er oppfylt dersom den er 1. Vi husker at *gpsEncodeComplete* ble satt lik *false* (0), om den ikke er blitt endret så er den fortsatt *false* (0). Utropstegnet (!) foran *gpsEncodeComplete* betyr at innholdet inverteres, dvs. at 0 blir til 1 og omvendt. Dvs. betingelsen er *true* og programmet vil fortsette å gå i ring i *do - while*-sløyfa helt til *gpsEncodeComplete* blir *true* (1).

La oss derfor se hva som befinner seg inne i `do {} while()`-sløyfa:

```
if (!gpsCom.available()) // Ingen GPS-data tilgjengelig. Les på nytt
{
    continue;
}
gpsEncodeComplete = gps.encode(gpsCom.read());
if (!gpsEncodeComplete) // Ufullstendig GPS-data tilgjengelig. Les på nytt
{
    continue;
}
```

Først kalles funksjonen `gpsCom.available()` denne returnerer verdien 1 dersom det er tilgjengelige data fra satellittene, og 0 dersom det ikke er det. Legg merke til at funksjonen kalles inne i argumentet til `if()`-funksjonen.

Dersom `gpsCom.available()` returnerer en `false` (0) som blir til (1) pga (!), vil den behandle innholdet i `if`-funksjonen hvor det kun står `continue`, som betyr at den skal hoppe over resten av `do {} while()`-sløyfa og sjekke betingelsen i loopen. Om denne fortsatt er oppfylt skal den begynne på nytt igjen.

Dersom `gpsCom.available()` returnerer en `true` (1) så vil den bli invertert til en (0) pga. (!) og den går videre uten å gå innom innholdet i `if()`-funksjonen. Dermed har man detektert at GPS-data er tilgjengelig.

Den går så videre og leser data med funksjonen `gpsCom.read()`. Resultatet settes inn i funksjonen `gps.encode()` som dekode de leste dataene. Det er først nå at en finner ut om de mottatte dataene er komplette. Dette indikeres ved at `gps.encode()` returnerer `true` (1) eller `false` (0). Om den er `false` så går den innom innholdet i `if()`-funksjonen og hopper over resten av `do {} while ()`-sløyfa og sjekker om betingelsen er oppfylt.

Om `gps.encode()` returnerer `true` (1) så vil den sette `gpsEncodeComplete = 1` gå utenom `if()`-funksjonen og hoppe ut av `do {} while ()`-sløyfa og gå videre i programmet. Den har altså funnet data og de er komplette.

Vi legger merke til om den ikke finner komplette data så vil den gå i `do {} while ()`-sløyfa til “evig tid” og blokkere resten av programmet.

Det neste som nå skjer er at den kaller funksjoner som kan fortelle om dataene er valide (`gps.location.isValid()`) og oppdaterte (`gps.location.isUpdated()`).

```
bool gpsValid = gps.location.isValid();
bool gpsUpdated = gps.location.isUpdated();
bool isUseful = gpsValid && gpsUpdated;
```

Om begge disse er oppfylte så settes verdien til variabelen `isUsefull = true` (1), hvis en eller begge ikke er oppfylt så settes `isUsefull = false` (0). Utfallet av disse testene er avgjørende for om lysdioden skal lyse grønt (har brukbare data) eller rødt (har ikke brukbare data):

```
if (!isUseful) // Ugyldige data tenn rød lysdiode
```



```
{
  digitalWrite(LED_RED, HIGH);
  delay(500);
  digitalWrite(LED_RED, LOW);
  return;
}
else // Gyldige data tenn grønn lysdiode
{
  digitalWrite(LED_GREEN, HIGH);
  delay(500);
  digitalWrite(LED_GREEN, LOW);
}
```

Lengre ned i programmet skrives de aktuelle dataene ut til monitoren:

```
Serial.print(gps.location.lng(), 6); // Lengdegrader, 6 desimaler
Serial.print(",");
Serial.print(gps.location.lat(), 6); // Breddegrader, 6 desimaler
Serial.print(",");
Serial.print(gps.altitude.meters(), 1); // Høyde i meter, 1 desimal
```

Legg merke til at utskriften inneholder kallene til de aktuelle dataene *gps.location.lng()* (lengdegrader med seks desimaler), *gps.location.lat()* (breddegrader med seks desimaler) og *gps.altitude.meters()* (høyden i meter).

Senere i programmet skrives de samme parameterne ut på det vesle OLED-displayet:

```
display.setCursor(0,24); // Flytt markør til neste linje
display.print("L:");
display.print(gps.location.lng(),5); // Skriv lengdegrad
display.print(" B:");
display.println(gps.location.lat(),5); // Skriv breddegrad
display.display(); // Overfør informasjonen til displayet
```

Vi har sløffet høyde pga. plasshensyn på displayet.

Da har vi laget oss et testprogram for å teste ut GPS-mottakeren og skrive ut resultatene i monitoren og på det vesle displayet.

6.2.3 Det komplette GPS-testprogrammet ser da slik ut:

Legg merke til at vi i testprogrammet for GPS-mottakeren har beholdt bibliotekene og utskriften til displayet slik at det skal være lett å se at data kommer ut.

```
// Programvare for testing av GPS-mottakeren
// av Nils Kr. Rossing 13.03.20 2020
```

```

// Skolelaboratoriet, Institutt for fysikk, NTNU

// Inkluder biblioteker
#include <SoftwareSerial.h> //
#include <TinyGPS++.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

//////////////////// Display //////////////////////
Adafruit_SSD1306 display(4); // Deklarasjon av enheten "display" av klassen
Adafruit_SSD1306

//////////////////// GPS //////////////////////
#define GPS_RX 5 // Går til Tx på GPS'en
#define GPS_TX 4 // Går til Rx på GPS'en

SoftwareSerial gpsCom(GPS_RX, GPS_TX);
TinyGPSPlus gps;

//////////////////// LED //////////////////////
#define LED_RED 7
#define LED_GREEN 8

///////// Testing og kalibrering ///////////
int intervall = 5; // Angir ca. intervall i sek. mellom hver måling
bool gpsOn = true; // True = GPS-måling på False = GPS-måling avslått

//////////////////// setup() //////////////////////
void setup()
{
  // Aktiver LEDpinner
  pinMode(LED_RED, OUTPUT);
  pinMode(LED_GREEN, OUTPUT);

  // Initialisering av seriekommunikasjon
  Serial.begin(9600); // Til PC monitor
  gpsCom.begin(9600); // Til GPS mottaker
}

```



```
display.begin(SSD1306_SWITCHCAPVCC, 0x3C); // Initialiser display med
adresse 0x3C

Serial.println("Test_GPS_1");
Serial.println("Tid (s), Lengdegrad, Breddegrad, Hoyde");

// Klargjør display for utskrift
display.clearDisplay(); // Slett informasjon på display
display.setTextSize(1); // Sett størrelse på tekst
display.setTextColor(WHITE); // Hvit tekst på sort bagrunn

// Skriver ut en initialiseringstekst til displayet
display.setCursor(26,6); // Plasser markør
display.println("Init. V5"); // Skriv Initialisering
display.setCursor(8,16); // Plasser markør øverst til venstre
display.println("Kan ta minutter");
display.display();
}

void loop() {

////////////////////// Les fra GPS-enhet ////////////////////////

if(gpsOn) // gpsOn=True Leser GPS-data - For testing
innendørs
{ // kan GPS slås av dvs. gpsOn=False
gpsCom.listen();
bool gpsEncodeComplete = false;

do
{
if (!gpsCom.available()) // Ingen GPS-data tilgjengelig. Les på nytt
{
continue;
}
gpsEncodeComplete = gps.encode(gpsCom.read());
if (!gpsEncodeComplete) // Ufullstendig GPS-data tilgjengelig. Les
på nytt
{
```

```

        continue;
    }
} while (!gpsEncodeComplete);    // Gjenta lesning til gyldige data oppnås

bool gpsValid    = gps.location.isValid();
bool gpsUpdated  = gps.location.isUpdated();
bool isUseful    = gpsValid && gpsUpdated;

if (!isUseful)           // Ugyldige data tenn rød lysdiode
{
    digitalWrite(LED_RED, HIGH);
    delay(500);
    digitalWrite(LED_RED, LOW);
    return;
}
else                       // Gyldige data tenn grønn lysdiode
{
    digitalWrite(LED_GREEN, HIGH);
    delay(500);
    digitalWrite(LED_GREEN, LOW);
}
}

///// Skriv ut data til PC monitor og SD-kort /////

Serial.print(long(millis()/1000));    // Skriv ut hele sekunder
Serial.print(",");
Serial.print(gps.location.lng(), 6); // Lengdegrader, 6 desimaler
Serial.print(",");
Serial.print(gps.location.lat(), 6); // Breddegrader, 6 desimaler
Serial.print(",");
Serial.print(gps.altitude.meters(), 1); // Høyde i meter, 1 desimal
Serial.println();

// Klargjør display på toppen av instrumentet for utskrift
display.clearDisplay();    // Slett informasjon på display
display.setTextSize(1);    // Sett størrelse på tekst
display.setTextColor(WHITE);    // Hvit tekst på sort bagrunn

```



```
// Skrive ut Lengde- og Breddegrad
display.setCursor(0,0);           // Flytt markør til neste linje
display.print("L:");
display.print(gps.location.lng(),5); // Skriv lengdegrad
display.print(" B:");
display.println(gps.location.lat(),5); // Skriv breddegrad
display.display();               // Overfør informasjonen til displayet

delay(intervall*1000);          // Intervall mellom hver måling
}
```

6.2.4 Avleste data

Slik vi har skrevet programmet så kan man forvente at følgende vil skje:

- Displayet vil ganske raskt vise

Init. V5

Kan ta minutter

- Dessuten vil det komme røde blink i lysdioden, dvs. den finner ikke gyldige data
- Etter ett eller to minutter begynner den blå lysdioden på GPS-mottakeren å blinke blått. Dette betyr at den har låst til tilstrekkelig antall satellitter og begynner å motta data.
- Så kommer de første grønne blinkene innimellom mange røde. Normalt vil det gjerne komme 2 – 3 røde blink mellom hvert grønne blink.
- Åpner vi monitoren vil vi kunne se noe som ligner på det som er vist på figuren under:

```
Test_GPS_1
Tid (s), Lengdegrad, Breddegrad, Hoyde
2,10.453933,63.426277,0.0
9,10.453945,63.426273,0.0
16,10.453948,63.426265,0.0
23,10.453955,63.426258,0.0
30,10.453967,63.426254,0.0
37,10.453980,63.426250,0.0
44,10.453990,63.426246,0.0
51,10.453992,63.426246,0.0
58,10.453986,63.426239,0.0
65,10.453986,63.426231,0.0
72,10.453982,63.426223,0.0
79,10.453975,63.426219,0.0
86,10.453965,63.426216,0.0
93,10.453961,63.426216,0.0
```

Vi legger merke til at høyde datamangler hvilket ikke er uvanlig i starten. Det kan gå flere titalls minutter for høydedataene kommer. Vi ser også at det går ca. 7 sek. mellom hver måling, 5 sek. er lagt inn som forsinkelse.



7 Den komplette luftkvalitetsmåleren

Vi har bygget opp og testet ut elektronikken og programmet bit for bit samtidig som vi har forsøkt å forstå koden. Det skulle nå bare være å laste opp det komplette programmet og teste instrumentet i sin helhet.

Listingen under viser det komplette programmet:

```
// Programvare for måling av luftkvalitet
// Bygget på programvare utviklet til Air:bit UiT
// Endret og tilpasset av Nils Kr. Rossing 13.03.20
// Skolelaboratoriet, Institutt for fysikk, NTNU

// Inkluder biblioteker
#include <SoftwareSerial.h> //
#include <TinyGPS++.h>
#include <SDS011.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

//////////////////// Display //////////////////////
Adafruit_SSD1306 display(4); // Deklarasjon av enheten "display" av klassen
Adafruit_SSD1306

//////////////////// GPS //////////////////////
#define GPS_RX 5
#define GPS_TX 4

SoftwareSerial gpsCom(GPS_RX, GPS_TX);
TinyGPSPlus gps;

////////// Støvpartikler SDS011 //////////
#define PM_TX 2
#define PM_RX 3
SDS011 sds;

//////////////////// LED //////////////////////
#define LED_RED 7
#define LED_GREEN 8

////////// Fuktighet og temperatur //////////
```

```

int pinTemp = A1;
int pinFukt = A0;
float digTemp;      // Digitalt avlest temperatur
float digFukt;      // Digitalt avlest fuktighet
float temp;         // Beregnet temperatur i Celsius
float fukt;         // Beregnet relativ fuktighet i %

//////// Testing og kalibrering //////////
int intervall = 5;  // Angir ca. intervall i sek. mellom hver måling
float tempCal=-4.0; // lineær kalibrering av temperatur
float fuktCal=0.0;  // lineær kalibrering av fukt
bool gpsOn = true;  // True = GPS-måling på False = GPS-måling avslått

////////////////////// setup() ////////////////////////
void setup()
{
  // Aktiver LEDpinner
  pinMode(LED_RED, OUTPUT);
  pinMode(LED_GREEN, OUTPUT);

  // Initialisering av seriekommunikasjon
  Serial.begin(9600);           // Til PC monitor
  sds.begin(PM_TX, PM_RX);     // Til støvsensor
  gpsCom.begin(9600);         // Til GPS mottaker

  display.begin(SSD1306_SWITCHCAPVCC, 0x3C); // Initialiser display med
  adresse 0x3C

  Serial.println("Luftforurensning_SC_nkr_5 13.03.20");
  Serial.println("Tid (s), Lengdegrad, Breddegrad, Hoyde, pm2.5, pm10, Temp(C),
  Fukt(%)");

  // Klargjør display for utskrift
  display.clearDisplay();      // Slett informasjon på display
  display.setTextSize(1);     // Sett størrelse på tekst
  display.setTextColor(WHITE); // Hvit tekst på sort bagrunn

  // Skriver ut en initialiseringstekst til displayet
  display.setCursor(26,6);     // Plasser markør

```



```
    display.println("Init. V5");      // Skriv Initialisering
    display.setCursor(8,16);        // Plasser markør øverst til venstre
    display.println("Kan ta minutter");
    display.display();
}

void loop() {

///// Avlesning av temperatur og fuktighet //////////////////////////////////

    // Midling over 100 målinger for å fjerne tilfeldig støy
    digTemp = 0;
    for(int i = 0; i<100; i++)
    {
        digTemp = digTemp + analogRead(A1);
    }
    digTemp = digTemp/100;

    temp = digTemp*500.0/1024 + tempCal;      //Omregning fra spenning til C

    // Midling over 100 målinger for å fjerne tilfeldig støy
    digFukt = 0;
    for(int i = 0; i<500; i++)
    {
        digFukt = digFukt + analogRead(A0);
    }
    digFukt = digFukt/500;

    fukt = (digFukt/1024 - 0.16)/0.0062;      // Omregning fra spenning til
fuktighet RH%
    fukt = fukt/(1.0546 -0.00216*temp)+fuktCal; // Korrigererer fuktighetsmåling
med temp

///////////////////////////////// Les fra GPS-enhet //////////////////////////////////

    if(gpsOn)                                // gpsOn=True Leser GPS-data - For testing
    innendørs
    {
        // kan GPS slås av dvs. gpsOn=False
    }
}
```

```

gpsCom.listen();
bool gpsEncodeComplete = false;

do
{
  if (!gpsCom.available()      // Ingen GPS-data tilgjengelig. Les på nytt
  {
    continue;
  }
  gpsEncodeComplete = gps.encode(gpsCom.read());
  if (!gpsEncodeComplete)     // Ufullstendig GPS-data tilgjengelig. Les
  på nytt
  {
    continue;
  }
} while (!gpsEncodeComplete); // Gjenta lesning til gyldige data oppnås

bool gpsValid  = gps.location.isValid();
bool gpsUpdated = gps.location.isUpdated();
bool isUseful  = gpsValid && gpsUpdated;

if (!isUseful)           // Ugyldige data tenn rød lysdiode
{
  digitalWrite(LED_RED, HIGH);
  delay(500);
  digitalWrite(LED_RED, LOW);
  return;
}
else                     // Gyldige data tenn grønn lysdiode
{
  digitalWrite(LED_GREEN, HIGH);
  delay(500);
  digitalWrite(LED_GREEN, LOW);
}
}

////////// Les fra støvmåleren //////////

float pm25, pm10;

```



```
int error;
do
{
    error = sds.read(&pm25, &pm10);
} while (error != 0);    // Fortsett å les av støvmåleren helt til gyldige data

///// Skriv ut data til PC monitor og SD-kort /////

Serial.print(long(millis()/1000));    // Skriv ut sekunder
Serial.print(",");
Serial.print(gps.location.lng(), 6); // Lengdegrader i desimalgrader, 6
desimaler
Serial.print(",");
Serial.print(gps.location.lat(), 6); // Breddegrader i desimalgrader, 6
desimaler
Serial.print(",");
Serial.print(gps.altitude.meters(), 1); // Høyde i meter, 1 desimaler
Serial.print(",");
Serial.print(pm25);                    // Skriv ut støvkonsentrasjon 2,5 um
Serial.print(",");
Serial.print(pm10);                    // Skriv ut støvkonsentrasjon 10 um
Serial.print(",");
Serial.print(temp,1);                  // Skriv ut temperatur
Serial.print(",");
Serial.print(fukt,1);                  // Skriv ut fuktighet
Serial.println();

// Klargjør display på toppen av instrumentet for utskrift
display.clearDisplay();                // Slett informasjon på display
display.setTextSize(1);                // Sett størrelse på tekst
display.setTextColor(WHITE);          // Hvit tekst på sort bagrunn

// Skriver ut temperatur og fuktighet
display.setCursor(0,0);                // Plasser markør øverst til venstre
display.print("T: ");                  // Skriv "Luftrykk
display.print(temp,1);                 // Skriv verdien til luftrykk med to desimaler
display.print(" C");                   // Skriv benevning mbar (mb)
display.print(", RF: ");                // Skriv "Temp.:", Temperatur
display.print(fukt,1);                 // Skriv den målte temperaturen i ...
```

```

display.println(" %");          // ... grader C

// Skrive ut støvkonsentrasjon
display.setCursor(0,12);       // Flytt markør til neste linje
display.print("pm2.5: ");      // ... ug/m3
display.print(pm25,1);         // Skriv den målte pm 2,5 u
display.print(" pm10: ");      // ... ug/m3
display.println(pm10,1);       // Skriv den målte pm 10 u

// Skrive ut Lengde- og Breddegrad
display.setCursor(0,24);       // Flytt markør til neste linje
display.print("L:");
display.print(gps.location.lng(),5); // Skriv lengdegrad
display.print(" B:");
display.println(gps.location.lat(),5); // Skriv breddegrad
display.display();             // Overfør informasjonen til displayet
og vis

delay(intervall*1000);        // Intervall mellom hver måling
}

```

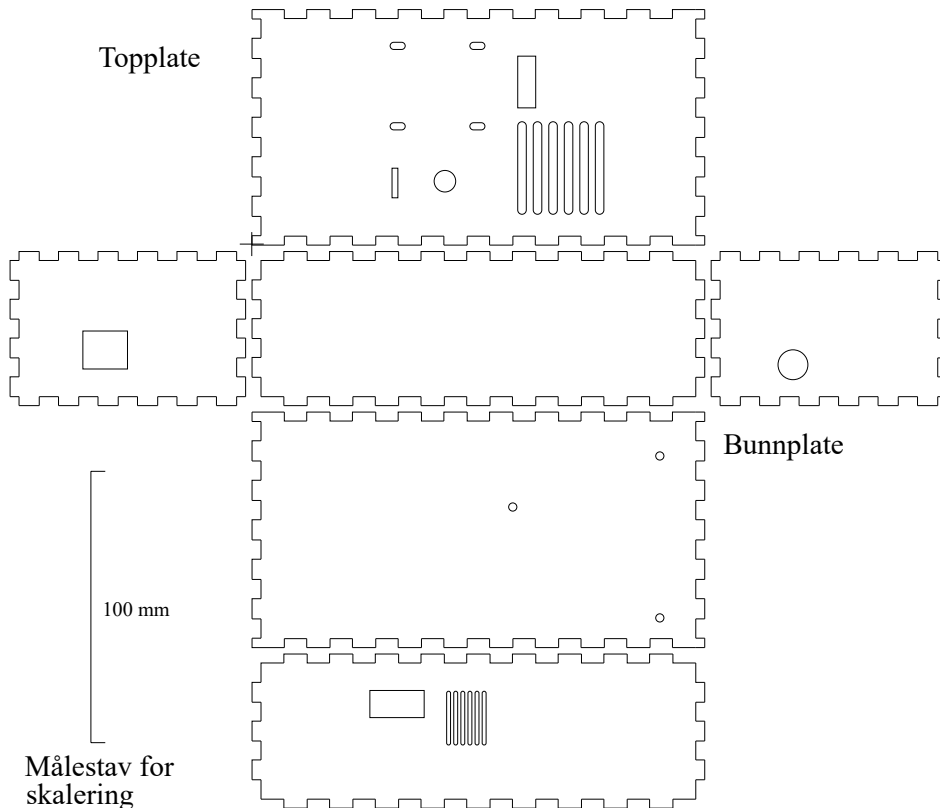


8 Oppbygging av et kabinett

Boksen er designet ved hjelp av et standard “boksprogram” på nettet og modifisert med hull og spalter slik at det skal passe til vårt formål. I dette avsnittet skal vi se hvordan vi kan sette sammen boksen.

8.1 Design av boksen

Selve boksen er tegnet med programmet: <https://makeabox.io/> . Figuren under viser en nedskalert versjon av boksen med inntegnede hull og spalter. Før den kan skrives ut så må den skaleres opp slik at målestaven nede til venstre blir akkurat 100 mm. Topplata er laget i gjennomsiktig 3 mm akryl, resten i 3.3 mm MDF.

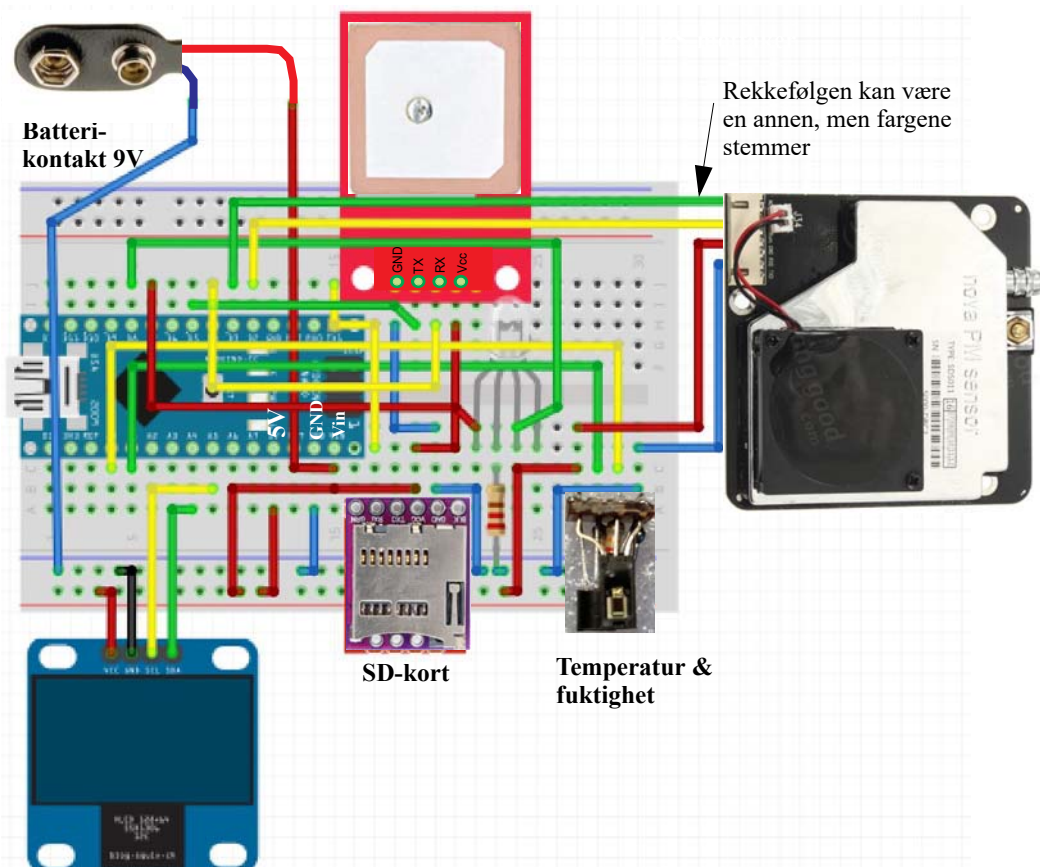


8.2 Sett sammen boksen

Til å beskytte elektronikken bruker vi en spesiallaget laserkuttet boks i MDF/akryl. Det er viktig at alle sideplatene plasseres på rett side da det er laget hull som passer til elektronikken på inn-

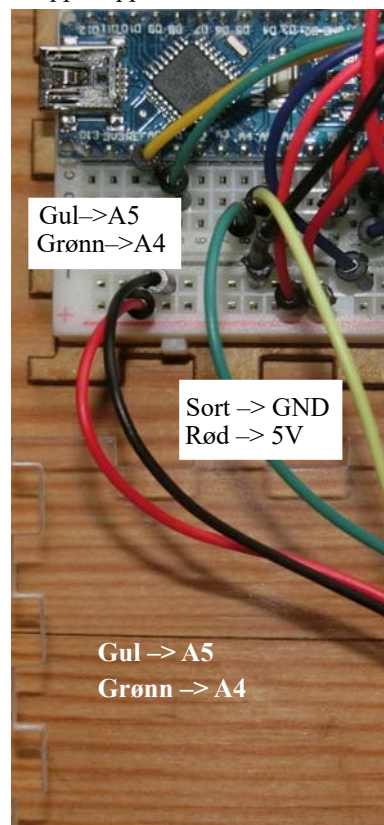
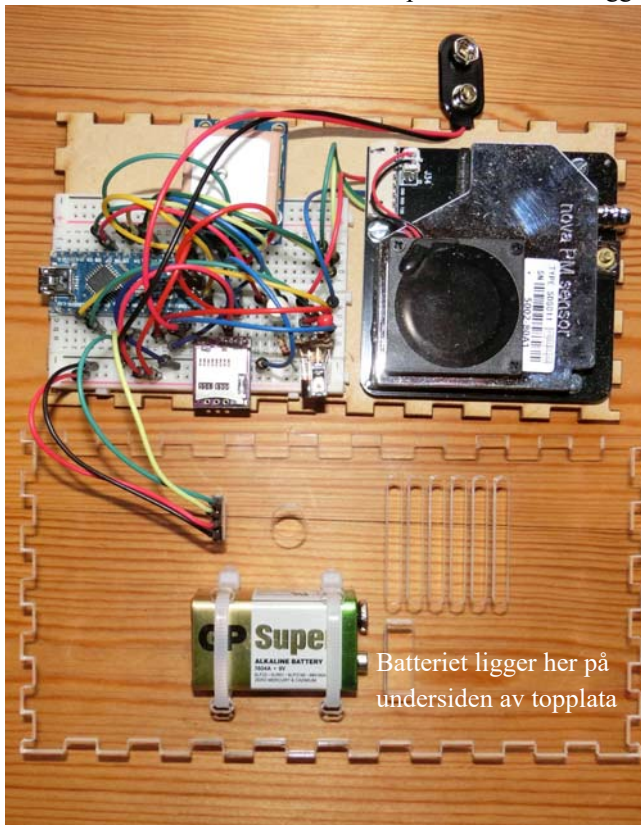
siden. Vi skal ikke lime sammen boksen, men kun presse sideplater og lokk sammen. Om de ikke sitter godt nok, bruker vi en strikk eller litt tape til å holde dem på plass.

- Det første vi gjør er å koble til ledningene fra displayet som er montert på topplate. Displayet har fire ledninger med ulike farge som vist nederst på figuren under.



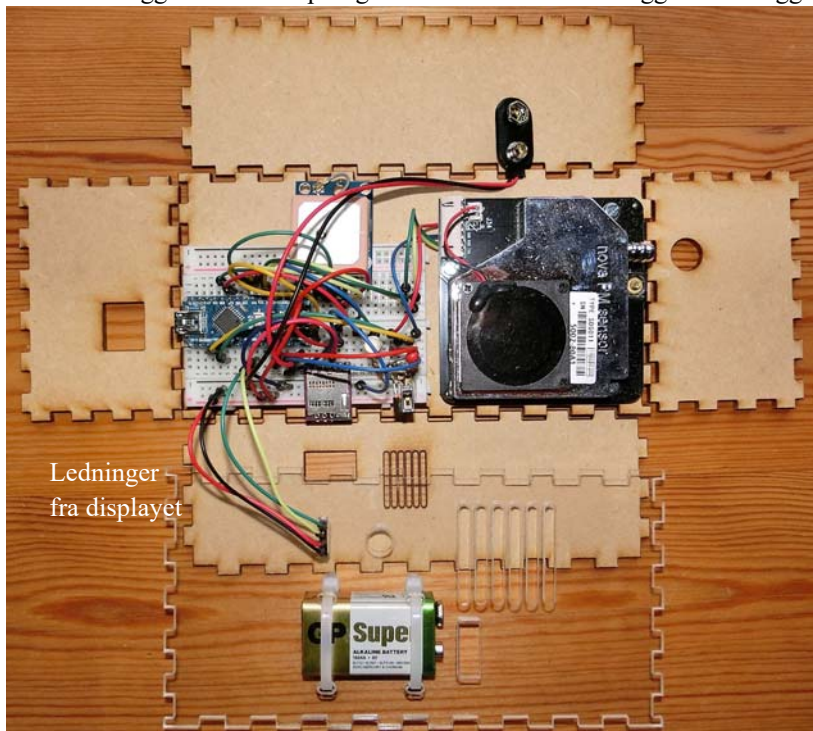


Vi legger toppлата opp-ned ved siden av bunnplata og kobler til ledningene som vist på bildet under. Monter batteriet med to strips slik at det blir liggende oppå toppлата.

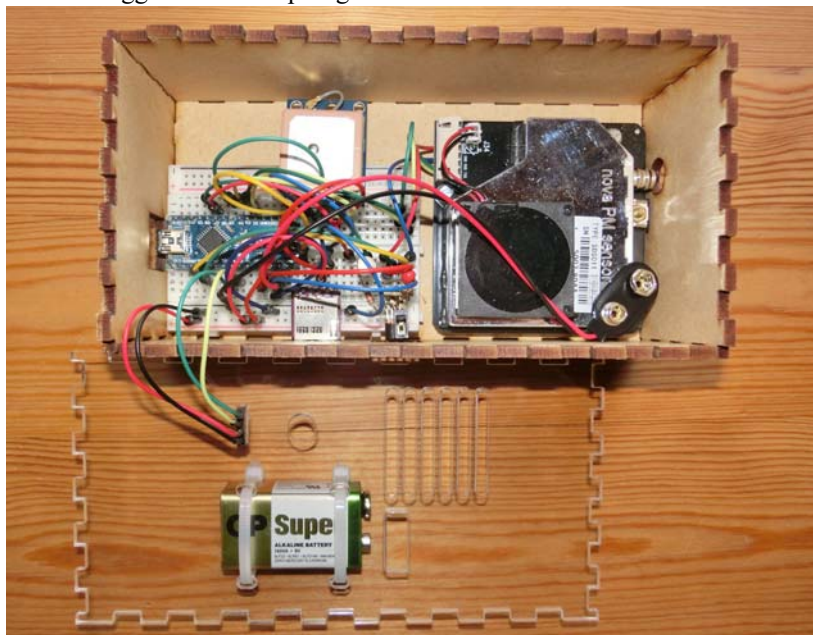


- Gå over og sjekk at alle ledningene er koblet opp riktig i henhold til skjemaet.

- Legg utover sideveggene som vist på figuren under. Merk hvor veggene skal ligge.

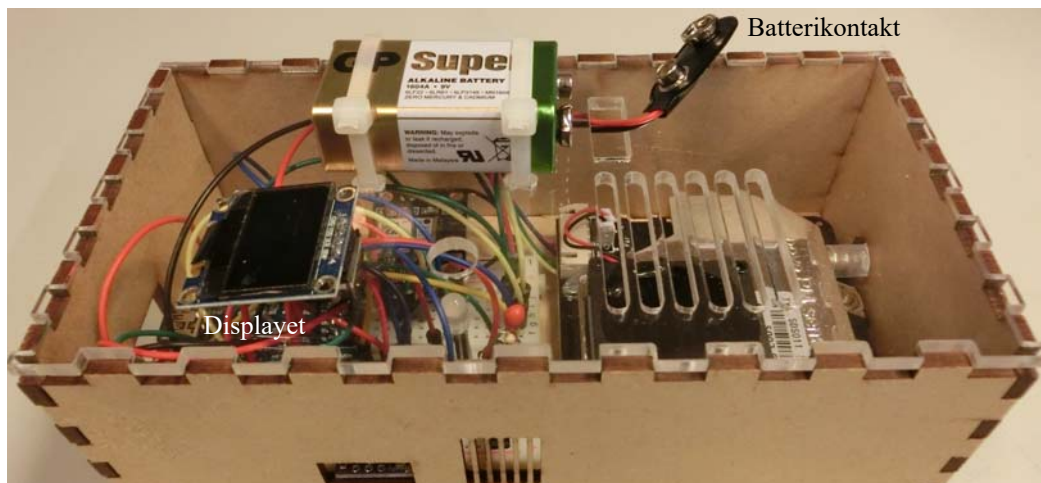


- Monter de fire veggene som vist på figurene under.

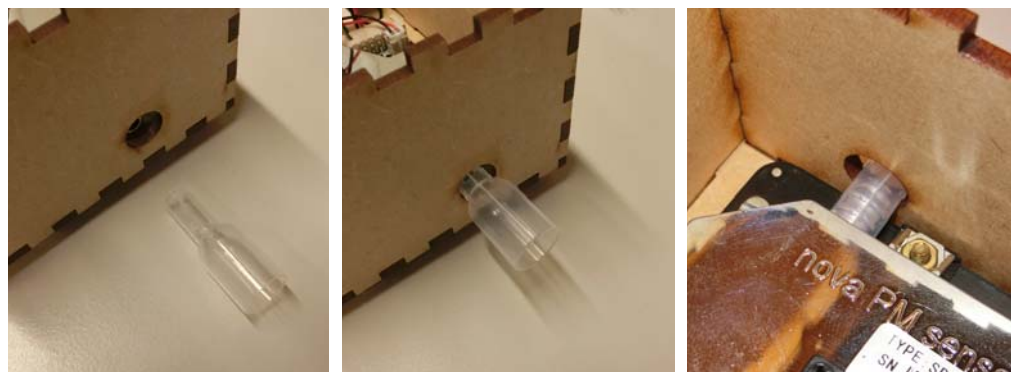




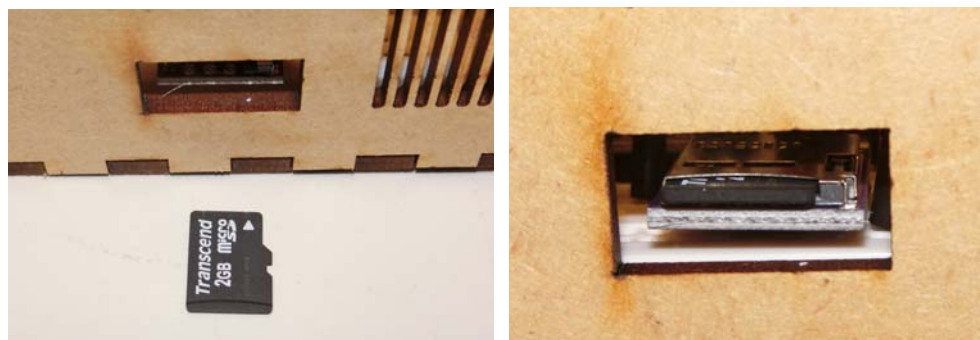
- Tre batterikontakten gjennom hullet i lokket og sett på lokket. Derne st plasseres displayet i sokkelen.



- Monter ei hette (avklipt plast pipette) over støvinntaket som vist på figuren under.



- Sett inn SD-kortet. Dette presses inn i SD-kortholderen til man merker et lite klikk. Kortet tas ut med å trykke det inn til man hører et lite klikk og det kommer ut. Påse at kortet settes inn rett vei.



- Vent med å koble til batteriet.

8.3 Første sjekk

Den første sjekken av kortet gjøres ved å koble USB-kabelen mellom PC'en og Arduino Nano'en ved hjelp av USB-kabelen.

Sjekk følgende:

1. En rød lysdiode skal lyse på Arduino Nano-kortet
2. Om denne ikke lyser. **NB! Ta ut kabelen og sjekk oppkoblingen.** Sjekk spesielt at de røde, blå og sorte ledningene er koblet rett.
3. Om den lyser. Sjekk at det blinker gult en gang i blant på undersiden av SD-kortholderen.



8.4 Installer programmet og test at det fungerer

- Last opp programmet som leser av alle sensorverdiene og skriver dem til SD-kortet.

Hent programmet fra nettsiden: <https://www.ntnu.no/skolelab/bla-hefteserie>

Gå ned til fanen: *Science Camp 2019*


Hent fila: *luftkval_1.0.ino*

Legg den et sted på PC'en hvor dere finner den igjen

- Start Arduino editoren

8.5 Prosedyre for å observere data når man er utendørs eller nær et vindu⁴

Man har flere måter å observere data på, enten ved å åpne monitoren eller ved å studere displayet. La oss først bruke monitoren:

4. Ytterst til høyre på menylinja finner man monitorsymbolet 



Når man åpner monitoren kommer det opp et nytt vindu hvor man kan studere data som skrives til SD-kortet

5. Åpne monitoren på PC'en og studer dataene som kommer.

4. For hjelp til å bruke Arduino editoren se vedlegg C



Dersom man befinner seg innendørs er det lite sannsynlig at GPS'en vil finne noen satellitter. Det er derfor nødvendig å gå utendørs for å teste GPS-mottakeren. Når GPS-kretsen låser til et tilstrekkelig antall satellitter vil den grønne lysdioden begynne å blinke, ellers vil den røde blinke. Det er normalt at det kommer et par røde blink mellom hvert grønt.

De utskrevne dataene vil ha formen:

```
COM8
Luftforurensning_SC_nkr_4C
Tid (s), Lengdegrad, Breddegrad, Hoyde, pm2.5, pm10, Temp(C), Fukt(%)
631,10.453965,63.426319,0.0,0.50,4.40,27.3,38.9
642,10.453982,63.426338,0.0,0.70,3.80,25.9,39.3
649,10.453991,63.426296,0.0,0.70,4.10,26.4,38.2
656,10.453999,63.426273,0.0,1.00,7.10,26.4,38.1
663,10.454012,63.426280,0.0,1.00,7.80,25.4,38.2
```

Tid i hele sekunder siden start, lengdegrad (°), breddegrad (°), høyde (m) over havet, partikkeltetthet pm2.5 ($\mu\text{g}/\text{m}^3$) og pm10 ($\mu\text{g}/\text{m}^3$), temperatur (°C) og relativ fuktighet (%).

8.6 Displayet

Bildet under viser displayet:



Bildet til venstre vises før GPS'en har låst til satellitter.

Bildet til høyre når alt fungerer som det skal:

- **T**: Temperatur i grader (°C),
- **RF**: Relativfuktighet (%),
- **pm2,5**: Konsentrasjon ($\mu\text{g}/\text{m}^3$) av 2,5 μm partikler,
- **pm10**: Konsentrasjon ($\mu\text{g}/\text{m}^3$) av 10 μm partikler inkluderer også alt aom er mindre,
- **L**: Lengdegrad (°),
- **B**: Breddegrad (°)

8.7 Prosedyre for testing av instrumentet innendørs

Programmet er laget slik at dersom det ikke mottar GPS-data så vil utstyret heller ikke skrive ut partikkeltetthet, temperatur og fuktighet. Under den innledende testingen hopper vi derfor over GPS-målingene og venter med dem til vi kommer utendørs.

- Slik endrer du programmet slik at det hopper over GPS-målingene

Gå inn i programmet og finn følgende programlinjer

```
//////// Testing og kalibrering //////////  
int intervall = 5; // Angir ca. intervall i sek. mellom hver måling  
float tempCal=0.0; // lineær kalibrering av temperatur  
float fuktCal=0.0; // lineær kalibrering av fukt  
bool gpsOn = true; // True = GPS-måling på False = GPS-måling avslått
```

Sett:

```
bool gpsOn = false;
```

Dermed vil programmet inntil videre hoppe over GPS-målingene og vise de øvrige måleverdier slik at instrumentet kan kontrolleres og ev. kalibreres innendørs.

8.8 Sjekk luftfuktighet og temperatur

Utfør følgende prosedyre:

- Slå på instrumentet
- Les av og skriv ned verdien for temperatur og luftfuktighet på *selvbygget instrument*:

Avlest luftfuktighet: _____ % relativ fuktighet

Avlest temperatur: _____ °C

Les av og skriv ned temperatur og luftfuktighet fra et *kalibrert instrument*

Avlest luftfuktighet: _____ % relativ fuktighet

Avlest temperatur: _____ °C

Sammenlign verdiene og kommenter:

Kalibrert instrument



TroTec BQ20 måler luftfuktighet, temperatur og pm 2.5 og pm 10.0

8.9 Sjekk partikkeltetthet

- Slå på instrumentet
- Les av og skriv ned verdier for partikkeltetthet på *selvbygget instrument*:



Avlest partikkeltetthet 2,5µm: _____ µg/m³

Avlest partikkeltetthet 10µm: _____ µg/m³

Les av og skriv ned verdier for partikkeltetthet *kalibrert instrument*

Avlest partikkeltetthet 2,5µm: _____ µg/m³

Avlest partikkeltetthet 10µm: _____ µg/m³

- Sammenlign målingene med grenseverdiene for krav til partikkeltetthet i lufta:

Alarmgrenseverdier for partikkelkonsentrasjon¹⁾

Luftkvalitet	Verdi i µg/m ³	Indikatorskala
Utmerket	0 til 10 µg/m ³	Grønn
Bra	10 til 35 µg/m ³	Gul
Lav belastning	35 til 75 µg/m ³	Oransje
Middels belastning	75 til 150 µg/m ³	Rød
Kraftig belastning	150 til 250 µg/m ³	Fiolett
Svært kraftig belastning	> 250 µg/m ³	Brun

Sammenlign verdiene og kommenter (se også vedlegg H):

8.10 Sjekk GPS-målinger og lagring av data

Vi skal nå sjekke at GPS-mottakeren fungerer som den skal. Denne testen må gjøres utendørs eller nær et vindu.

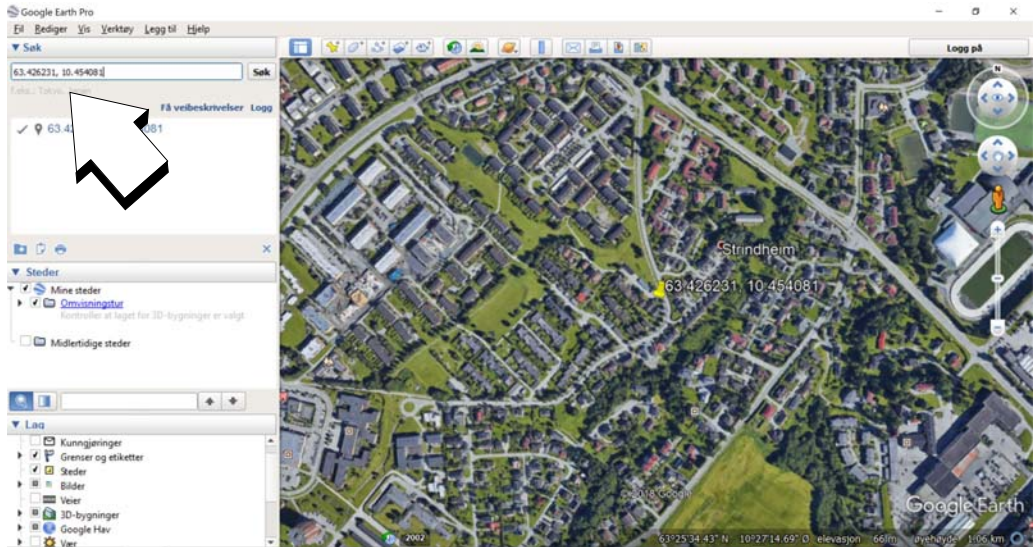
- Gå inn i programmet og slå på GPS-målingene:
`bool gpsOn = true;`
- Sett et SD-mikro kort inn i SD-kortholderen
- Slå på instrumentet. Det er sannsynlig at den lille røde lysdioden på GPS-mottakeren lyser kontinuerlig (ikke alle har denne), mens den store røde blinker, hvilket betyr at GPS-mottakerne ennå ikke har låst seg til satellittene og ingen målinger er tilgjengelig.
- Gå utendørs og vent til GPS-kretsen har låst seg til satellittene, det kan ta flere minutter. Dere vil da se at den lille røde (eller blå) lysdioden på GPS-mottakeren begynner og etter en stund at den store grønne dioden også begynner å blinke.
- Bli stående og vent et par minutter, før dere slår av instrumentet og går inn for å sjekke data.
- Ta ut SD-mikro kortet og sett det inn i adapteren og sett det så inn i PC'ens kortleser. Klikk på fila og studer resultatet. Les av koordinatene og skriv ned:



Breddegrad: _____ °

Lengdegrader: _____ °

- Åpne Google Earth og skriv inn koordinatene fra målingene i søkefeltet (pila på figuren under). Husk at breddegraden (63.--°) skal stå først, deretter lengdegraden adskilt med komma (10.--°), desimaltegnet skal være “.”. Bruk gjerne 5 eller 6 desimaler. Hvordan stemmer dette med forventningene?



8.11 Feltnålinger

Dersom alt ser ut til å fungere er vi klare for å dra ut på måleoppdrag i felten.

Siden dette opplegget opprinnelig ble laget for å gjennomføres på Skolelaboratoriet på Gløshaugen har vi valgt nærliggende lokasjoner for målingene. Dette må selvfølgelig tilpasses der dere befinner dere.

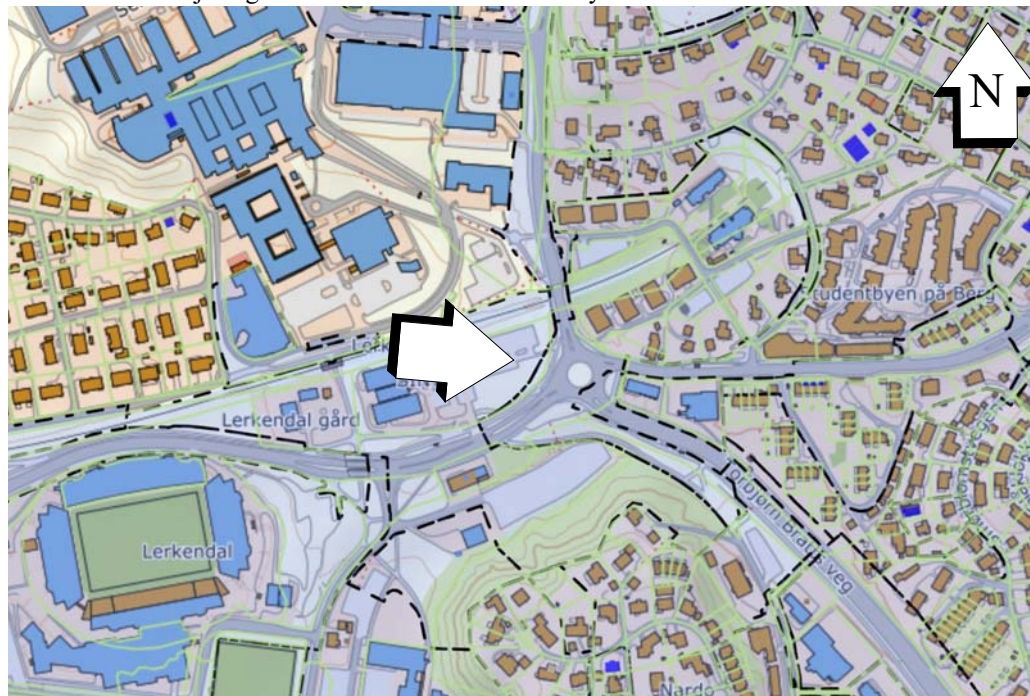
Ønsker man høye konsentrasjoner av partikler bør man velge dager da det er tørt i været og ganske støvete. Vinterdager med klar himmel og bare veier kan gi høye verdier, spesielt dersom mange bruker piggdekk.

Nå er det jo ingen ønskesituasjon med høy partikkeltetthet, så lav tetthet er egentlig godt nytt.



8.11.1 Dra ut til rundkjøring ved Lerkendal

Trafikkerte rundkjøringer kan være steder der det er mye svevestøv når det er tørt i været.



- **Gjør målinger med det hjemmelaget instrument**

Ta med dere instrumentet og dra ut til rundkjøringa nord for Lerkendal, plassere dere ved det angitte stedet og gjøre 2 min. opptak. Derneft skal dere gå rundt Rundkjøringa og gjøre målinger mens dere går. Instrumentet slås på i det dere forlater Realfagbygget og slås av idet dere er tilbake ved Realfagbygget.

- **Bestem vindretningen ved rundkjøringa**

Forsøk å bestemme hvilken kant vinden kommer, bruk kartet for å orientere dere. *Tegn inn vindretningen på kartet.*

- **Gjør målinger rundt rundkjøringa med profesjonelt instrument**

Gjør også målinger med profesjonelt instrument og noter verdier på fire ulike plasser rundt rundkjøringa. **Merk av på kartet.** Les også av fuktighet og temperatur.

Fuktighet: _____% Temperatur: _____ °C

1. Måling med profesjonelt måleinstrument 2,5µm: _____ µg/m³
Måling med profesjonelt måleinstrument 10µm: _____ µg/m³
2. Måling med profesjonelt måleinstrument 2,5µm: _____ µg/m³
Måling med profesjonelt måleinstrument 10µm: _____ µg/m³

3. Måling med profesjonelt måleinstrument 2,5 μm : _____ $\mu\text{g}/\text{m}^3$
Måling med profesjonelt måleinstrument 10 μm : _____ $\mu\text{g}/\text{m}^3$
4. Måling med profesjonelt måleinstrument 2,5 μm : _____ $\mu\text{g}/\text{m}^3$
Måling med profesjonelt måleinstrument 10 μm : _____ $\mu\text{g}/\text{m}^3$

8.11.2 Dra ned til krysset Elgeseter gata og gjør målinger

Vi har valgt å gjøre målinger i Elgeseter gate siden det er en kommunal målestasjon der. Dermed er det mulig å sammenligne egne målinger med kommunale målinger.



- **Gjør målinger med det hjemmelagde instrumentet**

Ta med dere instrumentet og dra ut til det angitte krysset i Elgeseter gate. Plasser dere ved det angitte stedet og gjøre 2 min. opptak. Dermed skal dere gå rundt krysset og gjøre målinger mens dere går. Instrumentet slås på i det dere forlater Realfagbygget og slås av når dere er tilbake på Realfagbygget.

- **Bestem vindretningen ved krysset**

Forsøk å bestem hvilken kant vinden kommer, bruk kartet for å orientere dere. *Tegn inn vindretningen på kartet.*



- **Gjør målinger rundt rundkjøringen med profesjonelt instrument**

Gjør også målinger med profesjonelt instrument og noter verdier på fire ulike plasser rundt rundkjøringen. **Merk av på kartet under.** Les av fuktighet og temperatur.

Fuktighet: _____ % Temperatur: _____ °C

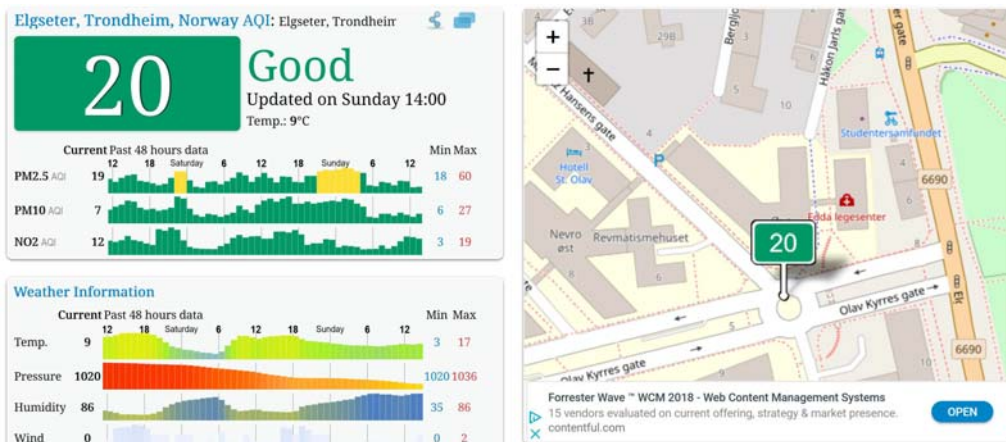
1. Måling med profesjonelt måleinstrument 2,5 μ m: _____ μ g/m³
Måling med profesjonelt måleinstrument 10 μ m: _____ μ g/m³
2. Måling med profesjonelt måleinstrument 2,5 μ m: _____ μ g/m³
Måling med profesjonelt måleinstrument 10 μ m: _____ μ g/m³
3. Måling med profesjonelt måleinstrument 2,5 μ m: _____ μ g/m³
Måling med profesjonelt måleinstrument 10 μ m: _____ μ g/m³
4. Måling med profesjonelt måleinstrument 2,5 μ m: _____ μ g/m³
Måling med profesjonelt måleinstrument 10 μ m: _____ μ g/m³

Merk av målepunktene på kartet under



8.12 Les av data fra kommunal målestasjon og sammenlign med egne målinger

Gå inn på nettstedet: <http://aqicn.org/city/norway/norway/trondheim/elgseter/> og ta ut data fra målestasjonen ved Elgeseter gate for den aktuelle perioden når det gjelder temperatur, luftfuktighet og partikkeltetthet.



- Noter følgende parametere fra målestasjonen for det aktuelle tidsrommet:

Temperatur: _____ C

Luftfuktighet: _____ % relativ fuktighet

PM2.5: _____ $\mu\text{g}/\text{m}^3$

PM10: _____ $\mu\text{g}/\text{m}^3$

Sammenlign med egne målinger utført med det hjemmelagde måleinstrumentet og den håndholdte profesjonelle måleren.

Kommenter resultatene og ev. avvik mellom de tre målingene:

8.13 Behandling av data fra hjemmelaget instrument

Sammenlign de dataene dere målte

Dere skal nå analysere dataene dere har funnet. Følgende skal gjøres

- Plott temperatur, luftfuktighet og partikkeltetthet for $2,5\mu\text{m}$ og $10\mu\text{m}$ som funksjon av tiden. Excel kan gjerne brukes for å plote disse dataene.
- Om dere ønsker å plott ruta dere gikk i Google Earth, så er det forklart i vedlegg F.



-
- Forsøk å knytte de to kurvene sammen slik at dere kan bestemme hvor de ulike målingene ble gjort.

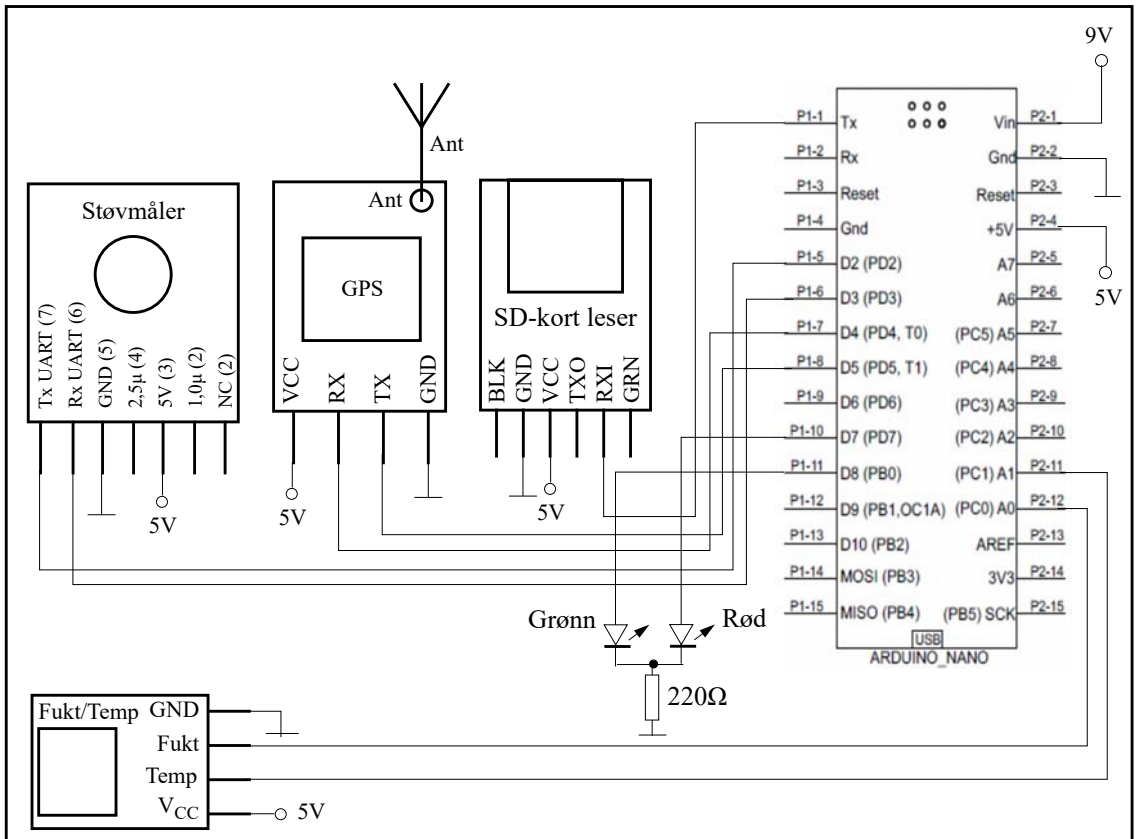
8.14 Konklusjon og presentasjon

- Oppsummer resultatene og vurder:
 - ... luftkvaliteten på de stedene dere har gjort målinger
 - ... kvaliteten på deres egne målinger med hjemmelaget utstyr
- Lag en 4 min. lang presentasjon av prosjektet dere har gjennomført. Følgende bør være med:
 - ... Beskrivelse og vurdering av byggeprosjektet og uttesting
 - ... Resultater fra målinger, hva og hvor dere gjorde målinger
 - ... Vurdering av luftkvaliteten

Vedlegg A Tekniske detaljer om luftkvalitetsmåleren

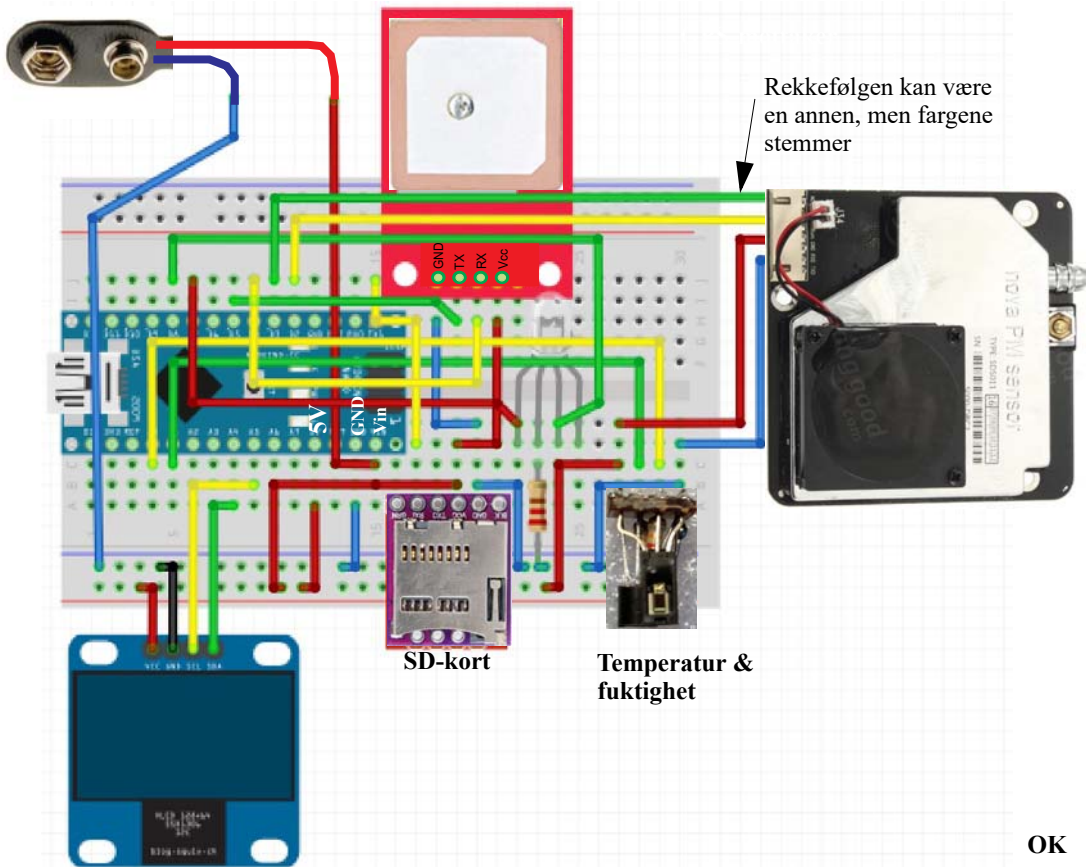
A.1 Koblingsskjema og oppkobling

Figuren under viser skjema for oppkobling av de ulike komponentene.





Kretsen kan kobles opp på et halvkort som vist på figuren under.



Tabellen under angir hvilke pinner som er koblet til hvilke ut- og innganger.

Tabell 8.1 Oversikt over tilkoblinger

Komponent	Pinne Komp	Pinne Nano	Funksjon
LED 1 (Rød)	anode	A1	Strøm
LED 2 (Blå)	anode	A0	Strøm

Tabell 8.1 Oversikt over tilkoblinger

Komponent	Pinne Komp	Pinne Nano	Funksjon
Temperatur/Fuktighets sensor LM35 og HiH4000	1	GND	GND
	2	A0	V _{LFout}
	3	A1	V _{Tout}
	4	Vcc	+5V
Støvmåler SDS011	3	Vcc	+5V
	5	GND	GND
	Rx UART (6)	D3	Data
	Tx UART (7)	D2	Data
SD-kort - OpenLog	+	+5V	+5V
	-	GND	GND
	RXI	D0 ->Rx	Data
	TXO	D1 ->Tx	Data
GPS NEO-6M	VCC	+5V	+5V
	GND	GND	GND
	RXD	D5	Data
	TXD	D4	Data

B.2 SDS011 - Støvmåler

Figuren viser pinningen til komponenten, “bottum view”.



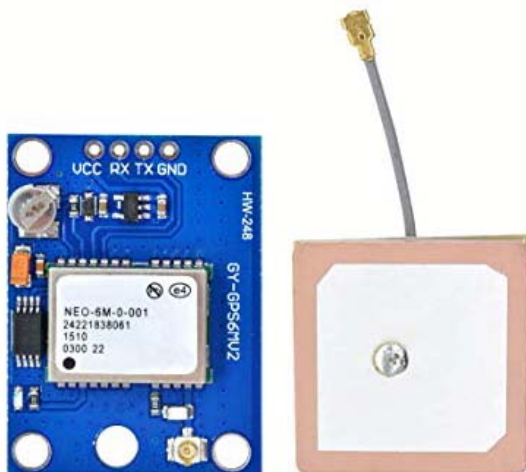
Målingene kommer ut som:

Mengde partikler med størrelse $2,5 \mu\text{m}$ angis i $\mu\text{g}/\text{m}^3$

Mengde partikler med størrelse $10 \mu\text{m}$ angis i $\mu\text{g}/\text{m}^3$.

B.3 NEO-6M - GPS

Figuren viser pinningen til komponenten, “bottum view”



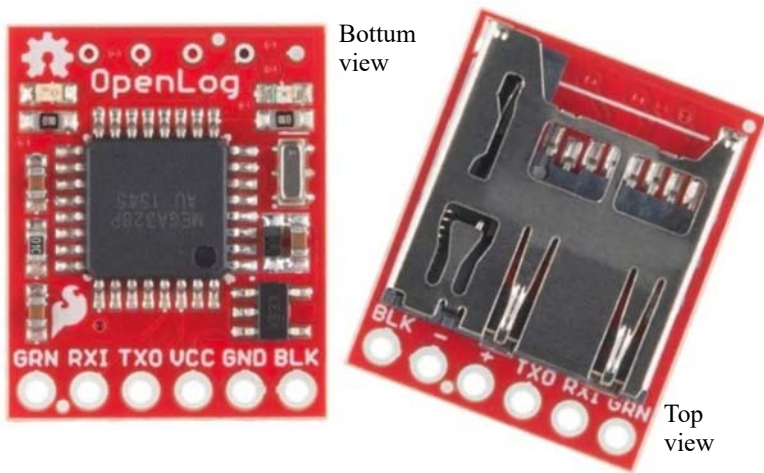
Her kan vi lese ut en rekke ulike data, men nøyer oss med følgende:

Tiden i timer, minutter og sekunder, lengdegrad i desimalgrader med seks desimaler, og breddegrad i desimalgrader med seks desimaler



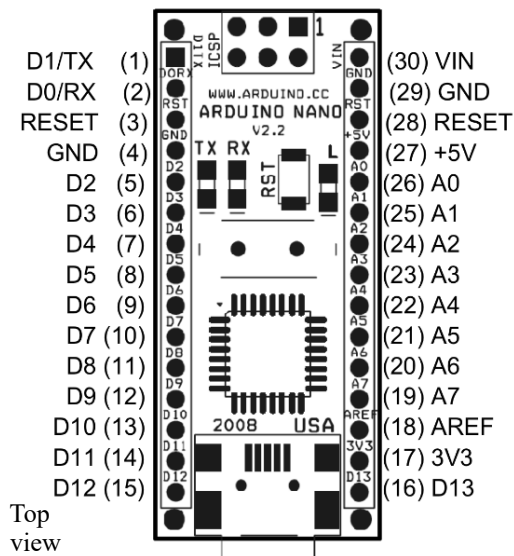
B.4 OpenLog

Figuren viser pinningen til komponenten



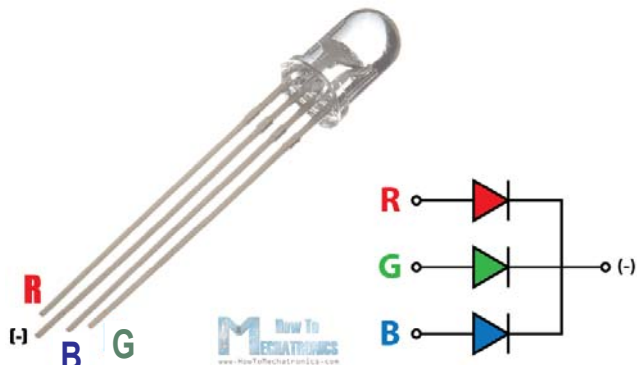
B.5 Arduino Nano

Figuren viser pinningen til komponenten



B.6 RGB LED

Figuren viser pinningen til komponenten



B.7 Komponent innkjøpliste

Lista under gir en oversikt over de viktigste komponentene og hvor de kan kjøpes. Det er også antydnet en pris.

Antall	Komponent	Leverandør	Pris
1	Partikkelmåler SDS011	BangGood https://www.banggood.com/search/sds011-sensor.html	Kr. 187,17 + Frakt MVA Toll
1	Luftfuktighetssensor HiH4000	ELFA Distrelec https://www.elfadistrelec.no/no/fuktighetssensor-honeywell-hih-4000-001/p/17305675	Kr. 180,00+ Frakt MVA
1	Temperatursensor LM35	ELFA Distrelec https://www.elfadistrelec.no/no/temperatursensor-to-92-lm35-texas-instruments-lm35dz-nopb/p/17309057	Kr. 16,30+ Frakt MVA
1	GPS-mottaker GY-NEO6MV2	Kult og Billig https://www.kultogbillig.no/index.php?_route_=GY-NEO6MV2-Flight-Controller-GPS-Modul-For-Arduino&keyword=GPS	Kr. 119,00+ Frakt
1	RGB-diode	Kult og Billig https://www.kultogbillig.no/index.php?_route_=20-stk-5mm-RGB-LED-Diode-rod-gronn-bla-l&keyword=RGB	Kr. 2,5 ved kjøp av 20 stk + Frakt
1	Arduino Nano	Kult og Billig https://www.kultogbillig.no/index.php?_route_=ATmega328P-Arduino-Nano-V3-kompatibelt-kort-USB-kabel-medfølger&keyword=Arduino%20nano	Kr. 76,00+ Frakt
1	OpenLog	BangGood https://www.banggood.com/Openlog-Blackbox-Flight-Data-Recorder-Cleanflight-Betaflight-Naze32-F3-for-FPV-Racing-Drone-p-1218836.html	Kr. 84,57 + Frakt MVA Toll
1	SD-kort mikro Kingston MicroSDHC 8GB	Komplett.no https://www.komplett.no/product/362738/foto-video/minnekort-til-foto-video/kingston-microsdhc-8gb	Kr. 33,0 + Frakt
1	Display 0.96 Inch I2C 128 x 64 OLED	Kult og Billig https://www.kultogbillig.no/index.php?_route_=Elektronikk/arduino/0-96-Inch-I2C-IIC-Serial-128-x-64-OLED-LCD-LED-Skjerm-Modul-For-Arduino	Kr. 59,00+ Frakt



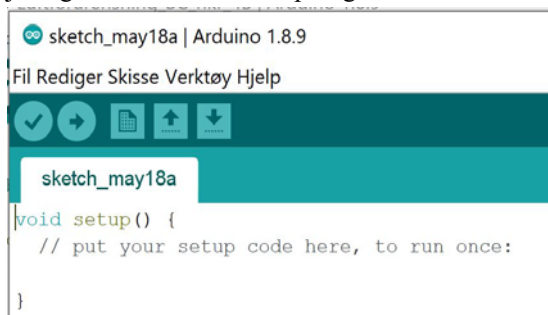
Antall	Komponent	Leverandør	Pris
1	Motstand 220 Ohm	ELFA Distrelec https://www.elfadistrelec.no/no/kullfilm-fastmotstand-220ohm-250mw-rnd-components-rnd-155rd14jn221t52/p/30156064	Kr. 0,20 + Frakt
1	Koblingsbrett	ELFA Distrelec https://www.elfadistrelec.no/no/experimenting-plate-hvit-400-koblingspunkter-84x55mm-rnd-components-rnd-255-00005/p/30115103	Kr. 24,80 + Frakt
Div.	Jumperkabler	Kult og Billig https://www.kultogbillig.no/index.php?_route_=Elektronikk/elektronikk-kabler-og-kontakter/65-X-Breadboard-Jump-Wires	Kr. 34,0 for 60 stk forskjellig lengde + Frakt
1	Hylselist 1 x 40-pins 2.54mm Hylselist	Kult og Billig https://www.kultogbillig.no/index.php?_route_=Elektronikk/elektronikk-kabler-og-kontakter/1-x-40-pins-2-54mm-Hylselist-Header	Kr. 3,40 ved kjøp av 10 stk + Frakt
1	Plast pipette	VWR ev. apotek	
1 plate	3,0 mm MDF	Nilsson Trelast AS (Trondheim) http://www.nilssontrelast.no/Default.aspx?id=28042010045706	Kr. 96,73 3,0 mm 122 x 244 cm
1 plate	3,0 mm klar Akryl	Alvøen (Bergen) https://www.alvoen.no/skiltemner-og-plater/akryl-for-laser/akryl-2-8-mm/akryl-klar-stoept-1010-x-710-x-3-mm	Ukjent 3 mm 1000 x700 mm
1	9V-batterikontakt	ELFA Distrelec https://www.elfadistrelec.no/no/batterikontakt-9v-key-stone-233/p/16914329	Kr. 6,68 + MVA, Frakt
1	9V batteri	ELFA Distrelec	Kr. 13,40 ved kjøp av 10 stk + MVA, Frakt

Vedlegg C Kort innføring bruk av Arduino editoren

Når du åpner editoren ser du følgende bilde:



Det viktige er menylinjen og ikonene som er vist på figuren under





1. Koble til Luftkvalitetsinstrumentet ved hjelp av USB-kabelen
2. Påse at SD-kortet er montert i SD-kort holderen

C.1 Prosedyre for å sette opp editoren for kortet Arduino Nano:

3. Velg "Verktøy" fra menylinjen. Velg "Kort" og velg "Arduino Nano" fra lista
4. Velg "Verktøy" fra menylinjen. Velg "Port" og velg den porten med høyest nummer fra lista
5. Velg "Verktøy" fra menylinjen. Velg "Prosesor" og velg "ATmega328 (Old bootloader)" fra lista

C.2 Prosedyre for å behandle programmet:

6. Hent opp programmet dere lastet ned ved å velge: 
Finn programmet og velg "Åpne" finn fila dere lastet ned
7. Sjekk at programmet er uten syntaksfeil ved å velge 



8. Kompiler og last opp programmet til Arduino Nano i instrumentet ved å velge



Programmet skal nå befinne seg i kontrolleren i instrumentet.

Vedlegg D Komponentdata

I dette vedlegget skal vi gjengi noen av databladene for noen av kretsene som brukes.

D.1 Luftfuktighetssensor - HiH 4000⁷

Fra databladet leser vi følgende:

Table 1. Performance Specifications (At 5 Vdc supply and 25 °C [77 °F] unless otherwise noted.)

Parameter	Minimum	Typical	Maximum	Unit	Specific Note
Interchangeability (first order curve)	–	–	–	–	–
0% RH to 59% RH	-5	–	5	% RH	–
60% RH to 100% RH	-8	–	8	% RH	–
Accuracy (best fit straight line)	-3.5	–	+3.5	% RH	1
Hysteresis	–	3	–	% RH	–
Repeatability	–	±0.5	–	% RH	–
Settling time	–	–	70	ms	–
Response time (1/e in slow moving air)	–	5	–	s	–
Stability (at 50% RH)	–	1.2	–	% RH	–
Voltage supply	4	–	5.8	Vdc	2
Current supply	–	200	500	µA	–
Voltage output (1 st order curve fit)	$V_{OUT} = (V_{SUPPLY})(0.0062(\text{sensor RH}) + 0.16)$, typical at 25 °C				
Temperature compensation	True RH = (Sensor RH)/(1.0546 – 0.00216T), T in °C				
Output voltage temperature, coefficient at 50% RH, 5 V	–	-4	–	mV/°C	–
Operating temperature	-40[-40]	See Figure 1.	85[185]	°C[°F]	–
Operating humidity	0	See Figure 1.	100	% RH	3
Storage temperature	-50[-58]	–	125[257]	°C[°F]	–
Storage humidity	See Figure 2.			% RH	3

Specific Notes:

- Can only be achieved with the supplied slope and offset. For HIH-4000-003 and HIH-4000-004 catalog listings only.
- Device is calibrated at 5 Vdc and 25 °C.
- Non-condensing environment.

General Notes:

- Sensor is ratiometric to supply voltage.
- Extended exposure to ≥90% RH causes a reversible shift of 3% RH.
- Sensor is light sensitive. For best performance, shield sensor from bright light.

Den avleste spenningen fra den analoge porten er:

$$V_{fukt} = (V_s)(0,0062 * (\text{sensor RH}) + 0,16) \text{ ved temp. typ. } 25 \text{ °C} \quad (\text{D.1})$$

Hvor *sensor RH* er målt relativt luftfuktighet i %, og V_s er batterispenningen.

Dersom vi løser denne mht. *sensor RH* og setter $V_s = 5V$ får vi:

$$\text{sensor RH} = (V_{fukt}/5V - 0,16) / 0,0062 \quad (\text{D.2})$$

Vi må også regne om fra et tall mellom 0 – 1023 til en spenning. Det gjør vi på følgende måte:

$$V_{fukt} = V_s * V_{fuktdig}/1023 = 5V * V_{fuktdig}/1023 \quad (\text{D.3})$$

Hvor $V_{fuktdig}$ er målt verdi på den analoge porten (0 – 1023). Setter vi ligning D.3 inn i D.2 får vi:

$$\text{sensor RH} = (V_{fuktdig}/1023 - 0,16) / 0,0062 \quad (\text{D.4})$$

7. <https://sensing.honeywell.com/honeywell-sensing-hih4000-series-product-sheet-009017-5-en.pdf>



Dersom vi ønsker å korrigere for temperaturen T slik at vi får *sann RH* så oppgir databladet denne sammenhengen:

$$\text{Sann RH} = (\text{sensor RH}) / (1,0546 - 0,00216 * T) \text{ hvor } T = \text{temp. i } ^\circ\text{C} \quad (\text{D.5})$$

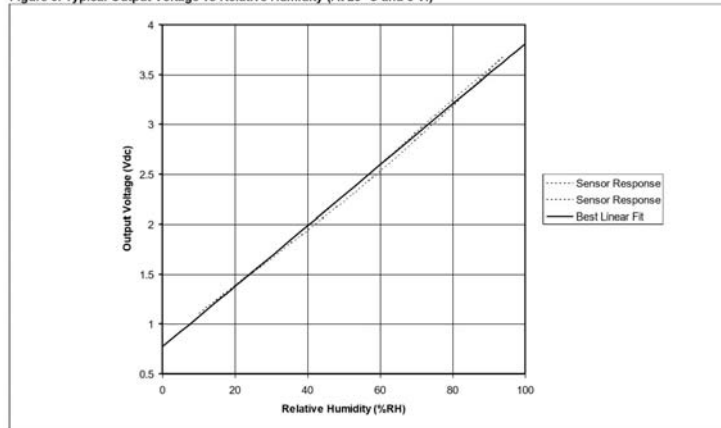
Det enkleste er nå å gjøre beregningen i to trinn:

1. Finne *sensor RH* fra ligning D.4
2. Sette verdien for *sensor RH* inn i ligning D.5

I programmet er avlest spenning fra porten kalt *digFukt* (int), og den korrigerte er kalt *fukt* (float).

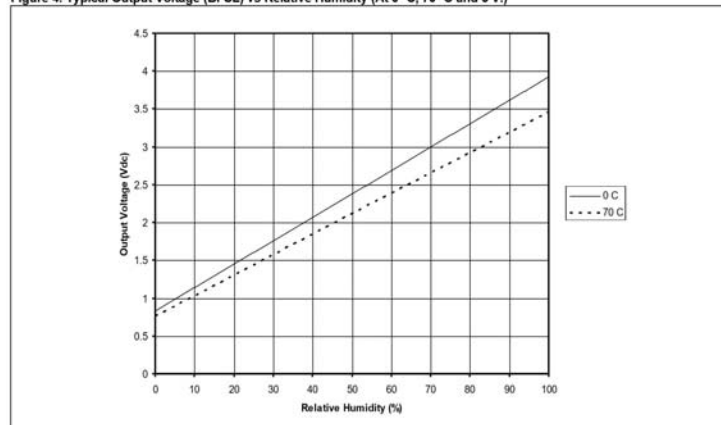
Så hvordan framkommer disse merkelige sammenhengene? Se det, sier databladet lite om. Men studerer vi figuren til høyre så ser vi at den målte sensorresponsen avviker noe fra den lineære sammenhengen. Det er rimelig å anta at den anbefalte formelen uttrykker den beste lineære tilpasningen.

Figure 3. Typical Output Voltage vs Relative Humidity (At 25 °C and 5 V.)



Vi ser også at denne sammenhengen endrer seg med temperaturen, så en korreksjon mht. temperaturen kan være på sin plass.

Figure 4. Typical Output Voltage (BFSL) vs Relative Humidity (At 0 °C, 70 °C and 5 V.)



Det er også mulig å bestille sensorer med kalibreringsdata, ala det som er vist på figuren til høyre. Dette kan være fornuftig dersom man krever høy nøyaktighet, men det koster.

Table 2. Example Data Printout

Model	HIH-4000-003
Channel	92
Wafer	030996M
MRP	337313
Calculated values at 5 V V _{OUT} at 0% RH V _{OUT} at 75.3% RH	0.826 V 3.198 V
Linear output for 3.5% RH accuracy at 25 °C Zero offset Slope RH	0.826 V 31.483 mV/%RH (V _{OUT} - zero offset)/slope (V _{OUT} - 0.826)/0.0315
Ratiometric response for 0% RH to 100% RH V _{OUT}	V _{SUPPLY} (0.1652 to 0.7952)



Vedlegg E Komplet programkode

Følgende er koden så langt den er utviklet (08.03.20):

```
// Inkludering av biblioteker
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <SDS011.h>

int pinTemp = A1;
int pinFukt = A0;
float digTemp; // Digitalt avlest temperatur
float digFukt; // Digitalt avlest fuktighet
float temp; // Beregnet temperatur i Celsius
float fukt; // Beregnet relativ fuktighet i %
float tempCal=0.0; // lineær kalibrering av temperatur
float fuktCal=0.0; // lineær kalibrering av fukt

// Deklarasjon av klasser
// Displayet
Adafruit_SSD1306 display(4); // Deklarasjon av display som klassen
Adafruit_SSD1306

// Partikkelsensoren
#define PM_TX 2
#define PM_RX 3
SDS011 sds;

void setup()
{
  Serial.begin(9600);
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C); //Initialiser displayet med adres-
sen 0x3C
  sds.begin(PM_TX, PM_RX); // Initialisering av partikkelsensoren
}

void loop()
{
```

```

// Les av temperatur og fuktighet
digTemp = analogRead(A1);
digFukt = analogRead(A0);

temp = digTemp*500.0/1023 + tempCal;           //LM35
fukt = (digFukt/1023 - 0.16)/0.0062;
fukt = fukt/(1.0546 -0.00216*temp)+fuktCal;   // korrigerer fuktighets-
måling med temp

// Les av partikkeltetthet
float pm25, pm10;
int error;
do
{
    error = sds.read(&pm25, &pm10);
} while (error != 0);    // Fortsett å les av støvmåleren helt til gyldige data

// Velg den utskriften som passer: Til monitor eller til fil, kommenter ut
den som ikke passer

// Skriv til monitoren

Serial.print((float)millis()/1000,1);         // Angir tid fra start i
sekunder med en desimal
Serial.print(", ");
Serial.print("Temp: ");
Serial.print(temp);
Serial.print("C");
Serial.print(", ");
Serial.print("Fukt: ");
Serial.print(fukt,1);
Serial.print("%");
Serial.print(", ");
Serial.print("pm2,5: ");
Serial.print(pm25);                          // Skriv ut støvkonsentrasjon 2,5 um
Serial.print("um");
Serial.print(", ");
Serial.print("pm10: ");
Serial.print(pm10);                          // Skriv ut støvkonsentrasjon 10 um

```



```
Serial.print("um");
Serial.println(", ");

/*
// Skriv til fil
Serial.print((float)millis()/1000,1);
Serial.print(";");
Serial.print(temp,1);
Serial.print(";");
Serial.print(fukt,1);
Serial.print(";");
Serial.print(pm25);           // Skriv ut støvkonsentrasjon 2,5 um
Serial.print(";");
Serial.print(pm10);           // Skriv ut støvkonsentrasjon 10 um
Serial.println(";");
*/

// Skriv til displayet

// Klargjør display for utskrift
display.clearDisplay();      // Slett informasjon på display
display.setTextSize(1);     // Sett størrelse på tekst
display.setTextColor(WHITE); // Hvit tekst på sort bakgrunn

// Skriver ut luftfuktighet med en desimaler
display.setCursor(0,0);     // Plasser markør øverst til venstre
display.print("Luftfukt: "); // Skriv "Luftfukt."
display.print(fukt,1);      // Skriv verdien til luftfukt med en desimaler
display.println(" %");      // Skriv benevning %

// Skriver ut temperatur med en desimal
display.setCursor(0,10);    // Flytt markør til andre linje
display.print("Temp.: ");   // Skriv "Temp.:", Temperatur
display.print(temp,1);      // Skriv den målte temperaturen i ...
display.println(" C");      // ... grader C

// Skriver ut partikkelkonsentrasjonen
display.setCursor(0,20);    // Flytt markør til neste linje
```

```
display.print("pm2.5: "); // ... ug/m3
display.print(pm25,1); // Skriv den målte pm 2,5 u
display.print(" pm10: "); // ... ug/m3
display.println(pm10,1); // Skriv den målte pm 10 u

display.display();
delay(500);
}
```

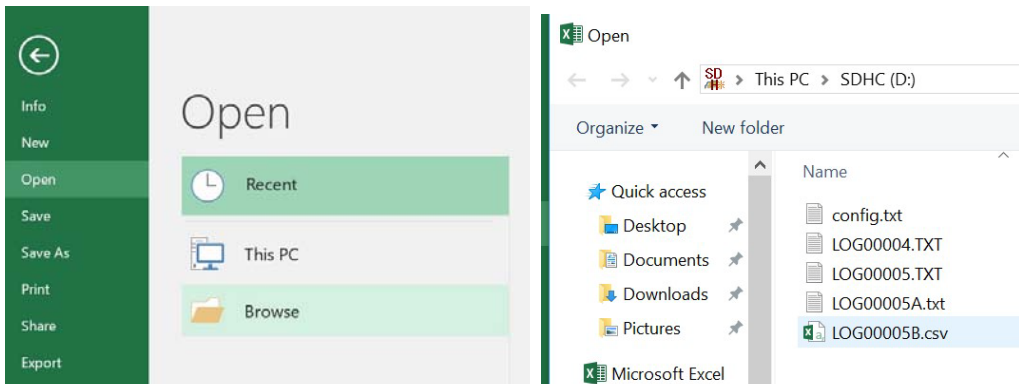


Vedlegg F Behandling av data

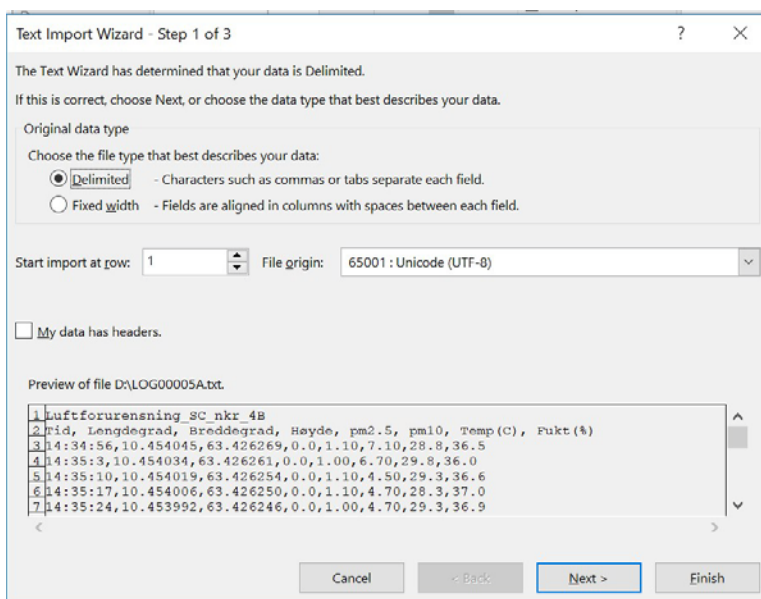
Dette vedlegget beskriver ulike måter å behandle dataene på.

F.1 Behandling av data for plotting av ruta i Google Earth

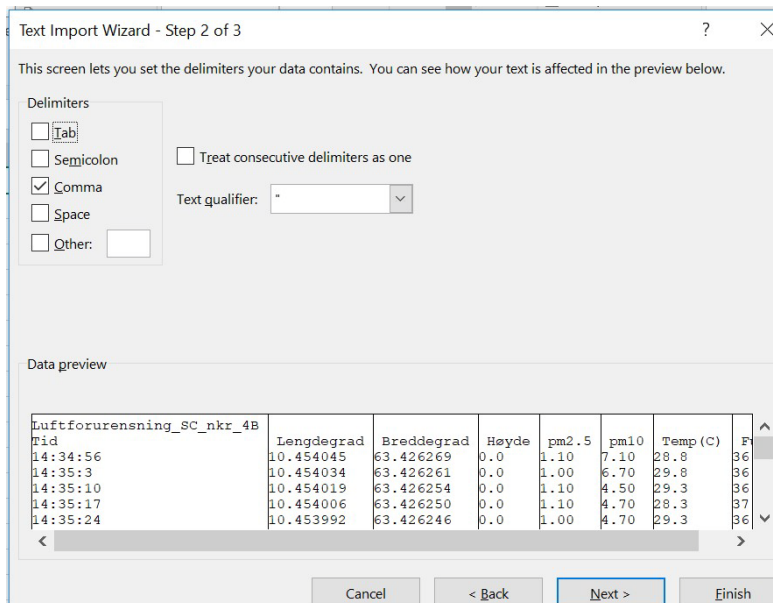
1. Start Excel
2. Velg «Open file» (under til venstre)



3. Velg «Browse» og velg SD-kort og den aktuelle filen, f.eks. LOG0005.TXT (over til høyre)
4. Du kommer da til «Text import Wizard» (under). Huk av på «Delimited», som betyr at vi kan skille kolonnene med skilletegn, f.eks. komma. Trykk «Next».



5. Fjern haken ved «Tab» og velg i stedet «Comma». Trykk så «Next» (under)





6. I dette vinduet kan du velge hvilke kolonner du vil ha med inn i regnearket. F.eks. tar vi bort tiden i første kolonnen. Vi velger også å ta bort pm2.5, pm10, Temp (C) og Fukt (%).

Text Import Wizard - Step 3 of 3

This screen lets you select each column and set the Data Format.

Column data format

General
 Text
 Date: DMY
 Do not import column (skip)

'General' converts numeric values to numbers, date values to dates, and all remaining values to text.

Advanced...

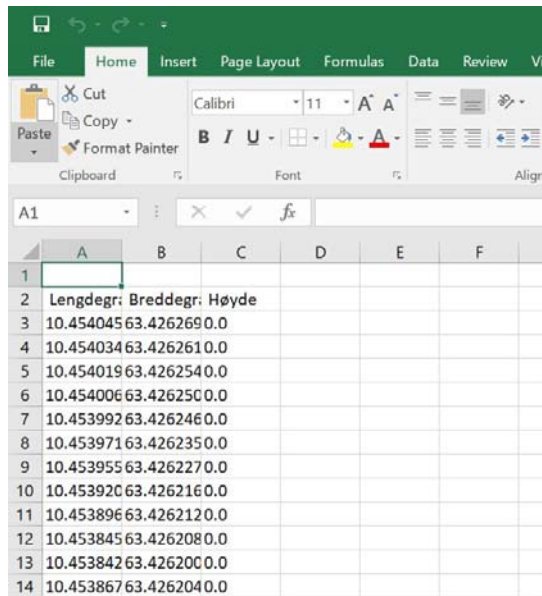
Data preview

Skip Column	General	General	General	General	General	General	General
Luftforurensning_SC_nkr_4B	Lengdegrad	Breddegrad	Høyde	pm2.5	pm10	Temp (C)	F
Tid	10.454045	63.426269	0.0	1.10	7.10	28.8	36
14:34:56	10.454034	63.426261	0.0	1.00	6.70	29.8	36
14:35:3	10.454019	63.426254	0.0	1.10	4.50	29.3	36
14:35:10	10.454006	63.426250	0.0	1.10	4.70	28.3	37
14:35:17	10.453992	63.426246	0.0	1.00	4.70	29.3	36
14:35:24							

Cancel < Back Next > Finish

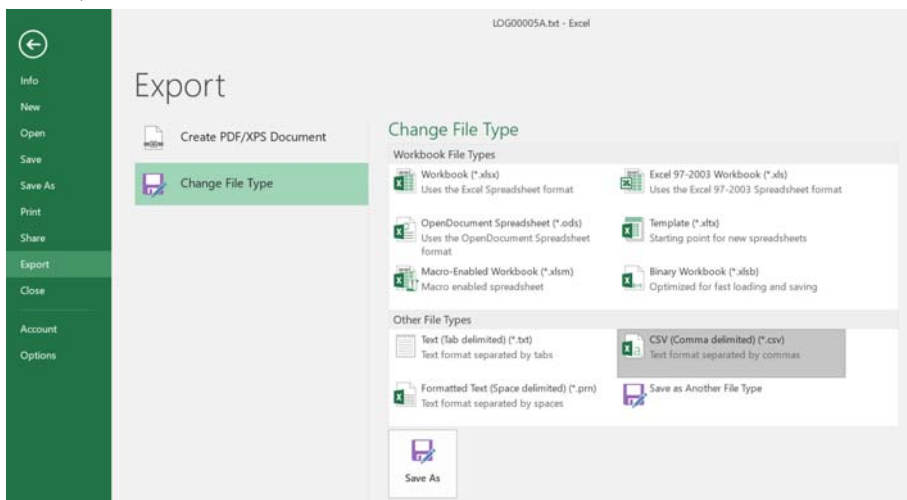
Disse skal vi behandle senere og velger å hake av på «Do not import column (skip)» for disse. Vi velger å ta med «Lengdegrad», «Breddegrad» og «Høyde» som vi merker med «General». Velg til slutt «Finish»

7. Vi skal nå få opp følgende bilde i Excel:



	A	B	C	D	E	F
1						
2	Lengdegr:	Breddegr:	Høyde			
3	10.454045	63.426269	0.0			
4	10.454034	63.426261	0.0			
5	10.454019	63.426254	0.0			
6	10.454006	63.426250	0.0			
7	10.453992	63.426246	0.0			
8	10.453971	63.426235	0.0			
9	10.453955	63.426227	0.0			
10	10.453920	63.426216	0.0			
11	10.453896	63.426212	0.0			
12	10.453845	63.426208	0.0			
13	10.453842	63.426200	0.0			
14	10.453867	63.426204	0.0			

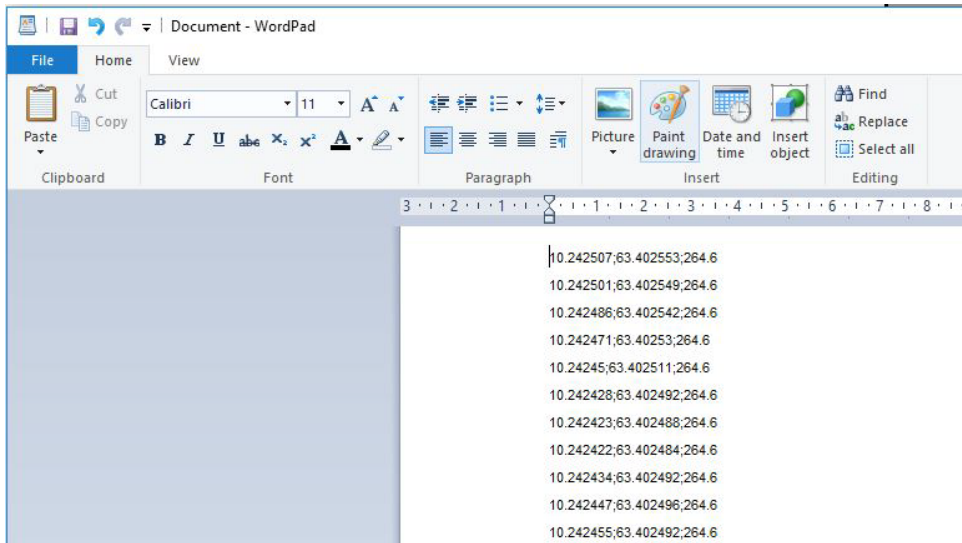
8. Vi har nå Lengdegrad, Breddegrad og Høyde i hver sin kolonne. Denne filen skal vi nå lagre som en CSV-fil, dvs. en fil hvor dataene er separert med komma («Comma Separated Values»).



9. Velg «Save as», velg så Export) og deretter CSV (Comma delimited). Filen lagres nå som en lang rekke data separert med komma.



10. Åpne fila i WordPad og sjekk om skilletegner er “;” eller “,”. Om det er koloen. Velg “Replace” og endre alle “;” til “,”.



F.2 Plotting i Google Earth

Vi har nå laget txt-filen som inneholder lengdegrad, breddegrad og høyde over havet som vist i figuren under. Vi velger å hente opp fila ved hjelp av Notisblokk eller Notepad (En)

Dersom vi skal plote en trase i Google Earth må vi benytte et programmeringsspråk kalt “Keyhole Markup Language” (KML). Dette er et “markup language” utviklet for visualisering av to- og tredimensjonale traseer knyttet til kartdata.

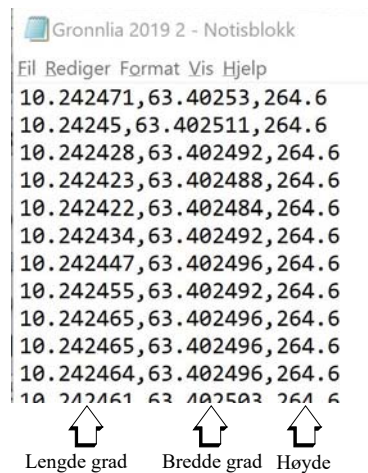
Vi skal nå benytte en ferdig programkode og klippe inn våre data for lengde-, breddegrad og høyde. Den ferdige programkoden ligger i mappa “Science Camp 2019” under “Luftkvalitet”, “Google Earth filer” og heter: “**Streknings-Mal.kml**”. Posisjonsdataene legges inn som en liste med i kml-koden som vist under, før kml-fila lagres med et ønsket navn.

NB! Ved lagring skriver man inn .kml etter filnavnet

Koordinater og høyde data legges inn som vist under:

10.454045,63.426269,65.0

Lengdegrader [°],Breddegrader [°],Høyde [m]



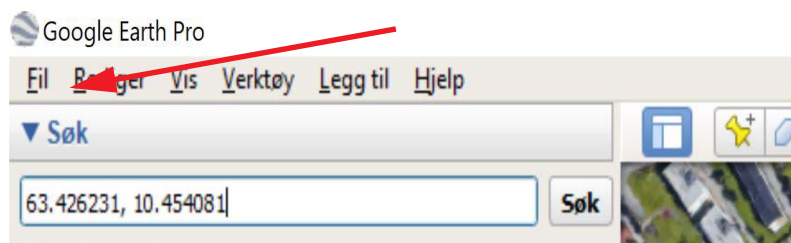
Programkode skrevet i kml (legg merke til at et sett med eksempelkoordinater og høyde er klippet inn).

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Document>
    <name>Paths</name>
    <description>Examples of paths. Note that the tessellate tag is by default
      set to 0. If you want to create tessellated lines, they must be authored
      (or edited) directly in KML.</description>
    <Style id="yellowLineGreenPoly">
      <LineStyle>
        <color>7f00ffff</color>
        <width>4</width>
      </LineStyle>
      <PolyStyle>
        <color>7f00ff00</color>
      </PolyStyle>
    </Style>
    <Placemark>
      <name>Absolute Extruded</name>
      <description>Transparent green wall with yellow outlines</description>
      <styleUrl>#yellowLineGreenPoly</styleUrl>
      <LineString>
        <extrude>0</extrude>
        <tessellate>0</tessellate>
        <altitudeMode>clampToGround</altitudeMode>
        <coordinates>
10.454045,63.426269,0.0
10.454034,63.426261,0.0
10.454019,63.426254,0.0
10.454006,63.426250,0.0
10.453992,63.426246,0.0
10.453971,63.426235,0.0
10.453955,63.426227,0.0
10.453920,63.426216,0.0
10.453896,63.426212,0.0
10.453845,63.426208,0.0
10.453842,63.426200,0.0

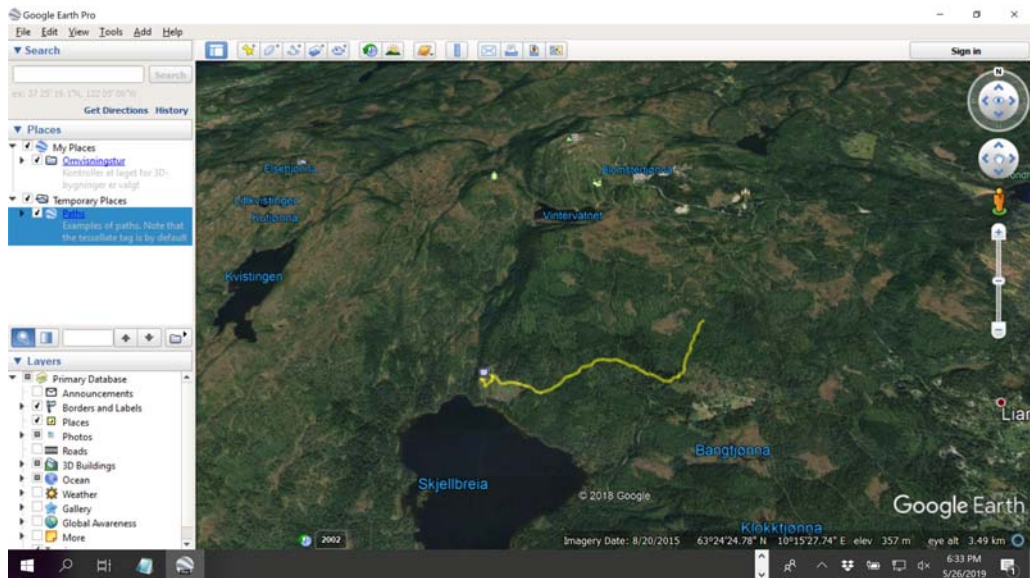
```



```
10.453867,63.426204,0.0
10.453885,63.426235,0.0
10.453924,63.426231,0.0
10.453936,63.426231,0.0
10.453932,63.426242,0.0
10.453831,63.426189,0.0
10.453737,63.426120,0.0
10.453672,63.426021,0.0
10.453612,63.425949,0.0
10.453562,63.425895,0.0
10.453533,63.425907,0.0
10.453561,63.425937,0.0
10.453541,63.425930,0.0
    </coordinates>
  </
LineSt-
ring>
  </
Placemark>
</Document>
</kml>
```



Filen hentes opp i Google Earth ved å velge *Fil* og *Åpne* for å laste opp den filen hvor koden og dataene ligger (se figuren over). Ev. bare klikke på filnavnet. Figuren under viser ruta i Google Earth.



Dersom man ønsker å vise hvor man har godt en tur så kan det være upraktisk å måtte vise absolutt høyde. Høydemålinger kan ha store avvik slik at en kan oppleve at traseen går mange meter over bakken eller forsvinner under bakken til tross for at man hadde begge beina på jorda. I slike situasjoner kan det være greit å bytte ut kommandoen:

```
<altitudeMode>absolute</altitudeMode>
```

med kommandoen:

```
<altitudeMode>clampToGround</altitudeMode>
```



Vedlegg G BQ20 Partikkelmåler

Kompakt miljømåleenhet for registrering av finstøvelastning, lufttemperatur og luftfuktighet

Med BQ20 har du en kompakt miljømåler som er enkel å bruke og gjør det mulig å registrere helserelevante romklimaparametre med kun ett apparat: Lufttemperatur, luftfuktighet og fremfor alt støvkonsentrasjonen i romluften:



BQ20 registrerer kvantitative konsentrasjoner av luftbåren E-støv (PM10) og A-støv (PM2.5) samtidig. Partikkelmassene til disse vises direkte som masseandel av luften i mikrogram pr. kubikkmeter på displayet. Dessuten informerer en søyleindikator med fargede søyler om statusen til partikkelbelastningen akkurat der og da.

I tillegg kan individuelle grenseverdier defineres i apparatet. Overskridelse av disse signaliseres umiddelbart med en akustisk alarm – en spesielt nyttig funksjon ved bruk av BQ20 for kontinuerlig overvåking av rom innendørs. For slik permanent bruk kan strømtilførselen også skje direkte med USB-kabel for å skåne

batteriet til den ellers mobile, batteridrevne BQ20.

På grunn av kombinasjonen av registrerbare miljøparametre som er helt unik for denne apparatklassen, anbefales BQ20 som et uunnværlig hjelpemiddel for rask overvåking av luftkvaliteten på arbeidsplassen, ved offentlige institusjoner eller hjemme.

Optisk partikkelmåler for kvantitativ registrering av konsentrasjonen av E- og A-støv.



Et menneskehår har en diameter på ca. 50 til 70 μm . Til sammenligning: diameteren til støvdelen som pustes inn (E-støv, PM10) og partiklene som trenger inn i alveolene (A-støv, PM2.5).

Det finnes nå i mange land lovlige grenseverdier for konsentrasjoner av finstøv for å beskytte nasjonens helse – for eksempel på arbeidsplassen.

Finstøv består av en kompleks blanding av ulike partikler og deles inn i ulike fraksjoner etter størrelse. Støv som kan pustes inn og trenger inn i nesehulen, kalles E-støv og defineres som støvstørrelsesfraksjon PM10 der alle støvpartikler har en aerodynamisk diameter på under 10 mikrometer.

Støvfraksjonen PM2.5 omfatter dessuten alle finpartikler som kan nå inn til alveolene – også kalt A-støv. Disse kan trenge inn i bronkiene og lungeblærene.

BQ20 kan registrere disse støvfraksjonene iht. den internasjonale PM-standarden (Particulate Matter) som PM10 og PM2.5. Måleren viser disse andelene numerisk per kubikkmeter romluft på fargedisplayet til BQ20. Dessuten fremstilles luftens belastningsgrad grafisk med søyler. På den måten kan du til enhver tid raskt komme med pålitelige kvantitative uttalelser om den rådende konsentrasjonen av finstøv med BQ20.

G.1 Kortfattet brukerveiledning

Dette er en meget kortfattet brukerveiledning og et utdrag av manualen som ligger på følgende nettadresse:

<https://no.trotec.com/fileadmin/downloads/Messgeraete/Luftqualitaet/BQ20/TRT-BA-BQ20-002-NO.pdf>

G.1.1 Vanlig bruk av BQ20



1. Slå på apparatet ved å holde PÅ-tasten inne til displayet begynner å lyse. Det høres et signal og apparatet er klart til bruk så snart skjermen begynner å lyse.
2. Ta av støvhetta foran luftinntaket.
3. Sikt med apparatet mot måleområdet.
4. Trykk på START-knappen og innsugingen av luft starter.



Apparatet teller ned ca. 5 sekunder før målingen starter. Deretter fortsetter målingen over det innstilte måleintervallet (30 sek. 1 min. 2. min. 5. min.) Målerresultatene vises på displayet (se over)

5. Trykk MEM-tasten for å lage data. Alle dataene lagres sammen med tidspunktet da prøven ble tatt.

G.1.2 Innstillinger

Ved å holde SET-tasten inne og en meny følgende innstillinger vil komme opp på skjermen.



1. Hold SET-tasten inn og SYSTEM-menyen kommer opp (til høyre)
2. **Data/Time**
Sett dato og klokke
3. **Alarm Set**
Sett terskel for alarm
4. **Sample Time**
Sett lengden på prøvetiden (30 sek, 1 min., 2 min. 5 min.)
5. **Unit (°C/°F)**
Velg grader C eller grader F
6. **Memory View**
Se en liste over lagrede måleverdier
7. **Mass/Particle**
Velg å vise måleverdien i $\mu\text{g}/\text{m}^3$ eller antall pr. m^3 .

8. **Auto Power off**
Sett tidsrom for automatisk “strømmen av”

G.1.3 Faregrenser

Tabellene under viser faregrenser for partikkelbelastning ...

Alarmgrenseverdier for partikkelbelastning¹⁾

Kanal	Grønn	Gul	Oransje	Rød	Fiolett	Brun
2,5 μm	0 til 545	546 til 1235	1236 til 2470	2471 til 3300	3301 til 4950	> 4950
10 μm	0 til 68	69 til 170	171 til 340	341 til 454	455 til 680	> 680

og partikkelkonsentrasjon.

Alarmgrenseverdier for partikkelkonsentrasjon¹⁾

Luftkvalitet	Verdi i $\mu\text{g}/\text{m}^3$	Indikatorskala
Utmerket	0 til 10 $\mu\text{g}/\text{m}^3$	Grønn
Bra	10 til 35 $\mu\text{g}/\text{m}^3$	Gul
Lav belastning	35 til 75 $\mu\text{g}/\text{m}^3$	Oransje
Middels belastning	75 til 150 $\mu\text{g}/\text{m}^3$	Rød
Kraftig belastning	150 til 250 $\mu\text{g}/\text{m}^3$	Fiolett
Svært kraftig belastning	> 250 $\mu\text{g}/\text{m}^3$	Brun

Alarmgrenseverdiene som er oppgitt her refererer til en gjennomsnittsverdi for partikkelkonsentrasjon PM_{2.5} over et tidsrom på 24 timer og er basert på de globale retningslinjene fra Verdens helseorganisasjon (WHO) for luftkvalitet.



Vedlegg H Måling av luftkvalitet med data fra Tr.heim

Vedlegget gir en kortfattet oversikt over hva som ligger i begrepet luftkvalitet. Beskrivelsen er bl.a. hentet fra nettstedet Luftkvalitet i Norge (<https://luftkvalitet.miljostatus.no/artikkel/170>).

H.1 Hva består luftforurensningen av?

Luftforurensning kan defineres som partikler, gasser og stoffer i lufta som er skadelige for mennesker og/eller økosystemer. Lokal luftforurensning har effekter på lokalt nivå og kan gi skader på:

- Menneskers helse
- Økosystemer og vegetasjon. Både nitrogendioksid (NO₂) og svoveldioksid (SO₂) bidrar til forurensning og overgjødning av vann og vassdrag. Bakkenær ozon kan skade vegetasjon.
- Materialer og bygninger. SO₂ medfører korrosjon og nedbryting av materialer i bygninger og kulturminner. Bakkenær ozon kan skade materialer.

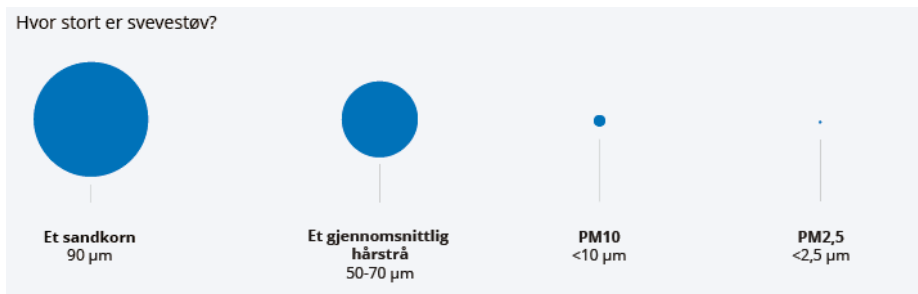
H.1.1 Svevestøv (PM)

Svevestøv, eller partikulært materiale (PM), er små, luftbårne partikler som varierer i størrelse og sammensetning. De viktigste størrelsesgruppene angitt i mikrometer (µm) er:

- PM_{0,1} (ultrafin fraksjon)
- PM_{2,5} (finfraksjonen)
- PM_{10-2,5} (grovfraksjonen)
- PM₁₀ (grovfraksjon og finfraksjon)

De største partiklene avsettes i øvre luftveier. Mindre partikler, med en diameter på under 2,5 µm, kan følge med lufta vi puster inn og helt ned i lungene.

Svevestøv består av en blanding av mange ulike forbindelser, og kan dannes ved forbrenningsreaksjoner og mekanisk slitasje, virvles opp av vind, eller dannes direkte i atmosfæren ved kondensering av gasser. Forbrenningspartikler dominerer i fin-/ ultrafin fraksjon, mens mekanisk genererte partikler som oftest dominerer i grovfraksjonen. De viktigste kildene til svevestøv (PM₁₀ og PM_{2,5}) skyldes veitrafikk, vedfyring og langtransportert forurensning.



Tabellen⁸ under gir en oversikt over hva som regnes for å være akseptable og uakseptable verdier. Partikler angis gjerne som $\mu\text{g}/\text{m}^3$.

AQI	Air Pollution Level	Health Implications	Cautionary Statement (for PM2.5)
0 - 50	Good	Air quality is considered satisfactory, and air pollution poses little or no risk	None
51 - 100	Moderate	Air quality is acceptable; however, for some pollutants there may be a moderate health concern for a very small number of people who are unusually sensitive to air pollution.	Active children and adults, and people with respiratory disease, such as asthma, should limit prolonged outdoor exertion.
101-150	Unhealthy for Sensitive Groups	Members of sensitive groups may experience health effects. The general public is not likely to be affected.	Active children and adults, and people with respiratory disease, such as asthma, should limit prolonged outdoor exertion.
151-200	Unhealthy	Everyone may begin to experience health effects; members of sensitive groups may experience more serious health effects	Active children and adults, and people with respiratory disease, such as asthma, should avoid prolonged outdoor exertion; everyone else, especially children, should limit prolonged outdoor exertion
201-300	Very Unhealthy	Health warnings of emergency conditions. The entire population is more likely to be affected.	Active children and adults, and people with respiratory disease, such as asthma, should avoid all outdoor exertion; everyone else, especially children, should limit outdoor exertion.
300+	Hazardous	Health alert: everyone may experience more serious health effects	Everyone should avoid all outdoor exertion

H.1.2 Nitrogendioksid (NO_2)

Nitrogenmonoksid (NO) og nitrogendioksid (NO_2) er reaktive gasser som dannes ved høy temperatur i forbrenningsprosesser, og disse har fellesbetegnelsen NO_x . I nærvær av ozon omdannes NO til NO_2 . Hovedkilden til NO_2 er veitrafikk.

H.1.3 Svoveldioksid (SO_2)

Svoveldioksid (SO_2) er en fargeløs gass som er lett løselig i vann. Den dominerende menneskeskapte utslippskilden er svovelholdige fossile brennstoffer som kull og tungolje. Hovedkilden til SO_2 er industri.

H.1.4 Bakkenær ozon (O_3)

Ozon er en reaktiv gass som finnes både nær bakken og høyere opp i atmosfæren. Bakkenært ozon dannes i nærvær av NO_x , flyktige organiske forbindelser (VOC) og sollys. Hovedkilden til bakkenært ozon i Norge er langtransportert ozon fra Europa.

8. <https://aqicn.org/sensor/sds011/>



H.1.5 Metaller

Ulike metaller kan forekomme i luft, hovedsakelig bundet til svevestøv. Eksempler på forurensende metaller er bly, nikkel, kadmium og arsen. Kilder er lokal industri, trafikk og langtransportert luftforurensning.

H.1.6 Polysykliske aromatiske hydrokarboner (PAH)

Forbrenning av organisk materiale fører til dannelse av PAH, også kjent som tjærestoffer. PAH inkluderer flere hundre forskjellige stoffer med ulike egenskaper.

PAH forekommer i forurenset luft som damp og bundet til svevestøv. Benzo(a) pyren er en av de mest helse- og miljøskadelige PAH-forbindelsene, og finnes i forurenset luft. Biltrafikk, industri og vedfyring er viktige kilder til PAH.

H.1.7 Benzen (C₆H₆)

Benzen er en fargeløs væske som fordamper raskt ved romtemperatur. I uteluft kan benzen forbli i dampform fra noen timer til flere dager, avhengig av miljø, klima og annen forurensning. Benzen er en naturlig komponent i olje- og kullprodukter, som frigjøres ved kull-, olje- og vedfyring. Trafikk er den viktigste utslippskilden.

H.1.8 Karbonmonoksid (CO)

CO er en fargeløs gass som hovedsakelig dannes ved ufullstendig forbrenning av organisk materiale. Naturlige prosesser gir betydelige CO-utslipp, men det er likevel de menneskeskapte utslippene som har størst betydning for helseeffektene. Trafikk og vedfyring er de viktigste.

Det er viktig å presisere at selv om CO₂ er en alvorlig trussel for det globale klimaet, så er ikke CO₂ noen forurensende gass lokalt. Den er derfor ikke omtalt under forurensende gasser.

H.2 Kilder til luftforurensning

Det finnes ulike kilder til lokal luftforurensning. Veitrafikk og vedfyring er de viktigste i Norge.

H.2.1 Veitrafikk

Veitrafikk er den viktigste kilden til lokal luftforurensning i Norge

Veitrafikk bidrar til luftforurensning både gjennom eksosutslipp og slitasjepartikler. Utslipp fra veitrafikk bidrar mye til konsentrasjonene av luftforurensning fordi de skjer på bakkenivå.

Eksos fra kjøretøy med forbrenningsmotor kan bestå av blant annet små svevestøvpertikler og nitrogendioksid (NO₂). Diesel-kjøretøy er den viktigste kilden til NO₂-utslipp i Norge, men utslippene kan reduseres med renseteknologi.

En annen viktig kilde til lokal luftforurensning er slitasjepartikler fra vei, bildekk og bremsler. Disse partiklene kalles ofte veistøv. Alle typer kjøretøy bidrar til slitasjepartikler, men det dannes langt mer veistøv når kjøretøy bruker piggdekk. Svevestøv fra veitrafikk kan virvles opp igjen om det ikke skylles bort av nedbør eller fjernes ved renhold.

Les mer om veitrafikk og luftforurensning på Miljostatus.no⁹.

H.2.2 Vedfyring og forbrenning

Vedfyring og forbrenning er en viktig kilde til luftforurensning i Norge

Ved forbrenning dannes det mange ulike stoffer som slippes ut i røyken. Spesielt utslipp av små svevestøvpertikler (PM_{2,5}) som i stor grad kommer fra vedfyring. I tillegg slippes mange andre stoffer ut, blant annet PAH.

Utslipp fra vedfyring bidrar til konsentrasjoner av luftforurensning over større områder fordi de skjer via piper som er høyere opp enn bakkeplan, noe som fører til større spredning.

Les mer om vedfyring og luftforurensning på Miljostatus.no.

H.2.3 Skip og havn

Skipstrafikk og havneaktivitet kan bidra til luftforurensning lokalt

Utslipp fra skip og havn kan være en viktig kilde til luftforurensning lokalt. Selv om skip kan ha store totale utslipp av luftforurensning er det hovedsakelig det som slippes ut når skip ligger ved land med motorene i gang som påvirker lokal luftkvalitet i byer.

Nitrogendioksid, svoveldioksid og små svevestøvpertikler er de viktigste utslippene fra skip. Som for alle andre utslippskilder vil bidraget til lokal luftforurensning variere med lokale forhold som topografi og meteorologi. Det er for eksempel viktig om den dominerende vindretningen er fra eller mot land.

H.2.4 Industri

Industri er en viktig kilde til forurensning noen steder i Norge

På enkelte steder i Norge er industri en vesentlig og viktig kilde til lokal luftforurensning. På landsbasis er industri generelt en mindre viktig kilde enn veitrafikk og vedfyring på grunn av strenge krav til rensing av utslipp, og fordi utslippene vanligvis slippes ut via høye skorsteiner.

Utslippene kan bestå av mange ulike stoffer, og varierer med type industri og hva som produseres. Svevestøv, nitrogendioksid, svoveldioksid, metaller og PAH er blant stoffene som kan slippes ut fra industri. I tillegg påvirkes luftkvaliteten i Norge av industriutslipp i andre land som transporteres til Norge med vinden.

Les mer om industri og luftforurensning på Miljostatus.no.

9. <https://www.miljostatus.no/tema/luftforurensning/utslipp-fra-veitrafikk/>



H.2.5 Langtransportert luftforurensning

Langtransportert luftforurensning kommer med vinden til Norge

Det er små svevestøvpartikler (PM10 og PM2,5) og gasser som ozon som lettest transporteres med vinden til Norge. Langtransportert luftforurensning kan stamme fra industriutslipp i andre land og fra naturlige kilder som skogbranner.

Les mer om langtransportert luftforurensning på Miljøstatus.no.

H.2.6 Naturlige kilder

Naturlige kilder kan også bidra til lokal luftforurensning

Naturlige kilder kan være utslipp fra vulkaner i utbrudd og skogbranner. Skogbranner og vulkanutslipp vil gi lignende utslipp som vedfyring, og fordi det ofte er store områder som brenner kan skogbranner gi store utslipp på relativt kort tid. Utslipp fra både lokale kilder og fra andre land kan påvirke luftforurensningen i Norge.

H.2.7 Bygg- og anlegg

Bygg- og anleggsvirksomhet kan bety mye lokalt

Bygg- og anleggsvirksomhet kan være en betydelig kilde til utslipp av luftforurensning lokalt. For eksempel ved utslipp fra anleggsmaskiner, graving og transport av masser.

H.2.8 Andre kilder

Andre kilder som kan påvirke luftkvaliteten

Jordbruk kan bidra med utslipp av partikler, ammoniakk og utslipp fra kjøretøy som traktorer. Flytrafikk har utslipp som kan påvirke den lokale luftkvaliteten ved flyplasser. Slitasjepartikler fra skinner, hjul og bremses fra tog, trikk og t-bane kan bidra til høye nivåer av svevestøv på innendørs perronger.

H.3 Hva bestemmer hvor mye luftforurensning det er der jeg er?

Sted: Hvor du befinner deg har stor betydning. Om du befinner deg i nærheten av en forurensningskilde, for eksempel en vei med mye trafikk, og hvordan vær- og spredningsforholdene er (for eksempel mye eller lite vind) vil avgjøre hvor utsatt du er for å puste inn forurenset luft.

Nærhet: Avstanden til forurensningskildene har betydning for i hvilken grad du blir eksponert for luftforurensning. Jo lengre vekk du beveger deg fra forurensningskilden (for eksempel fra veikanten), jo mindre luftforurensning blir du eksponert for.

Skjer forurensende utslipp nær bakken får utslippet større betydning for mennesker som oppholder seg her, enn hvis det skjer høyere oppe. Utslipp fra piper skjer høyere oppe og blir uttynnet før det kommer ned til bakken.

Noen forurensninger har også lengre rekkevidde enn andre. Grove svevestøvpartikler faller ras-

kere til bakken enn finstøv. Veistøv består mest av grove partikler. Det grove støvet når kortere ut fra veien, enn finstøv som kan fordele seg over større områder.

Tid: Luftkvaliteten varierer mye med årstidene og gjennom dagen. Luftkvaliteten er generelt på sitt verste om vinteren og er verre på dagtid enn om natten. Årsakene er spesielle værforhold om vinteren og høyere forurensningsaktivitet på dagtid.

Vær: Værforholdene påvirker hvordan luftforurensningen spres. Vind vil spre forurensningen, for eksempel eksosgass og veistøv fra biler. Nedbør vasker luften for støv og binder det. Perioder uten nedbør fører til oppvirvling av veistøv fra tørre veibaner, slik at støvet bidrar til forverret luftkvalitet. Når veiene er våte blir det mindre oppvirvling av støv.

I noen tilfeller oppstår et værphenomen som kalles inversjon. Normalt synker temperaturen med høyden, men ved inversjon er det omvendt. Dette fører til at kald luft ligger under varm luft. Siden kaldluft er tyngre enn varmluft vil den kalde luften virke som et lokk, der luften under lokket ikke slipper ut. Forurensningen vil da hope seg opp og bli høy.

Omgivelser: Topografi, bygninger og vegetasjon påvirker spredningen av luftforurensning. I noen tilfeller kan omgivelsene fungere som en barriere mot forurensningskilden, og føre til mindre luftforurensning der man befinner seg. Også elver og åpent terreng vil kunne lede forurensningen til områder lenger unna. I andre tilfeller kan omgivelsene føre til mer luftforurensning, for eksempel kan bygninger hindre at forurensningen spres.

Vegetasjon: Vegetasjon kan både ha en filtrerende effekt på luftforurensningen, og kan påvirke luftstrømmene. Vegetasjon langs veier kan for eksempel bidra til å binde partikler og redusere forurensningen, i tillegg til å skjerme bakenforliggende områder. Om vinteren mangler mye av vegetasjonen bladverk og mister dermed sine filtrerende egenskaper.



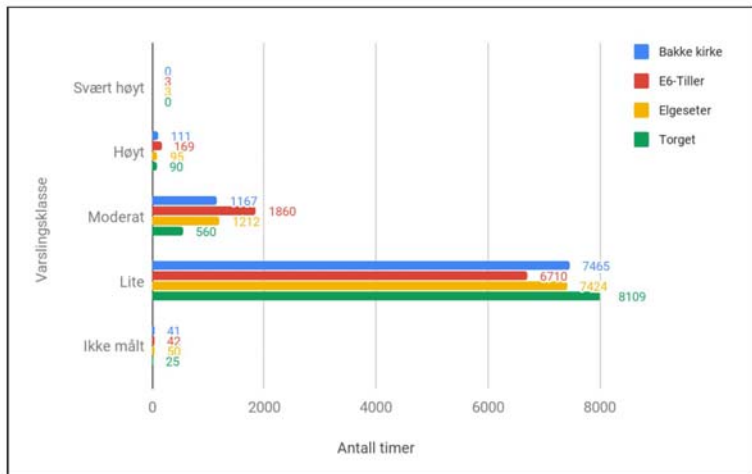
H.4 Forholdene i Trondheim¹⁰

Tabellen under viser varslingsgrenser for partikkeltetthet og NO₂.

TABELL 1 VISER DE NYE VARSLINGSKLASSENE FOR LOKAL LUFTKVALITET I NORGE (2015). TALLENE ER I MIKROGRAM PER KUBIKKMETER (µG/M³). MERK AT DE NYE VARSLINGSKLASSENE ER ENDRET, SLIK AT FORURENSNINGSNIVÅET LETTERE BLIR "HØYT", MENS DET SKAL MYE TIL FØR DET BLIR "SVÆRT HØYT".

Varslingsklasse	PM ₁₀ døgn	PM _{2,5} døgn	PM ₁₀ Time	PM _{2,5} Time	NO ₂	Helsevirkninger	Helseråd
Lite	<30	<15	<50	<25	<100	Liten eller ingen helseisiko	Utendørs aktivitet anbefales
Moderat	30-50	15-25	50-80	25-40	100-200	Moderat helseisiko: Helseeffekter kan forekomme hos enkelte astmatikere og personer med andre luftveissykdommer, samt alvorlige hjertekarsykdommer	Utendørsaktivitet kan anbefales for de aller fleste, men enkelte bør vurdere sin aktivitet i områder med mye trafikk eller høye andre utslipp
Høyt	50-150	25-75	80-400	40-150	200-400	Betydelig helseisiko: Helseeffekter kan forekomme hos astmatikere og personer med andre luftveissykdommer, samt alvorlige hjertekarsykdommer	Barn med luftveislidelser (astma, bronkitt) og voksne med alvorlige hjertekar- eller luftveislidelser bør redusere utendørsaktivitet og ikke oppholde seg i de mest forurenkede områdene
Svært høyt	>150	>75	>400	>150	>400	Alvorlig helseisiko: Følsomme grupper i befolkningen kan få helseeffekter. Luftveisiritasjoner og ubehag kan forekomme hos friske personer	Personer med hjertekar- eller luftveislidelser bør redusere utendørsaktivitet og ikke oppholde seg i de mest forurenkede områdene

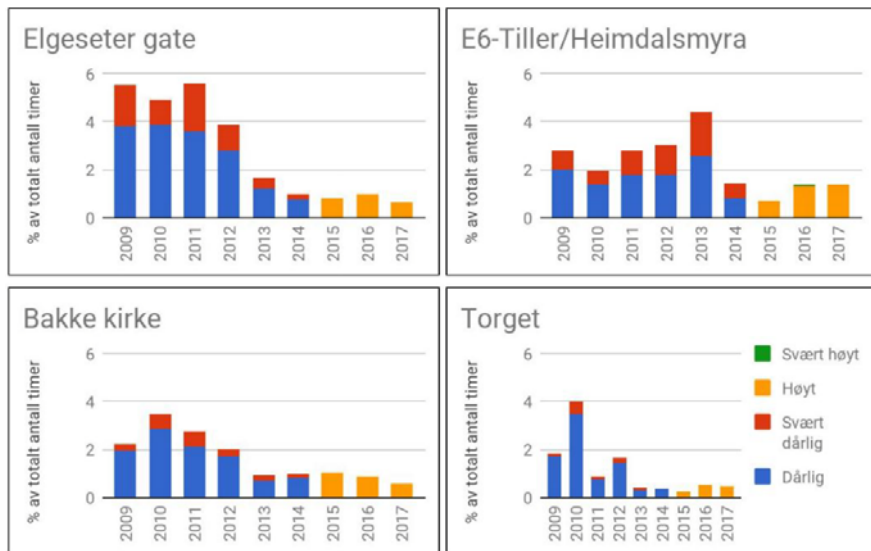
Luftkvaliteten i Trondheim er vanligvis god sommerstid. Om vinteren gir bruk av piggdekk på bar asfalt svevestøv, mens kalde motorer og fyring gir noe svevestøv, NO₂ og andre gasser. Det er mest luftforurensning i kaldt, vindstille, tørt vær i rushtidene og på utfartsdager. Luftkvaliteten i Trondheim overvåkes på fire målestasjoner; E6-Tiller, Elgeseter, Bakke kirke og Torvet (se figuren høyre).



Figur 6: Varslingsklasser gir en samlet vurdering av luftkvalitet basert på alle timeverdiene i hele 2018, for svevestøv (PM₁₀, PM_{2,5}) og NO₂. Merk at antall timer "moderat" luftforurensning ikke nødvendigvis er reelt, jamfør figur 1.

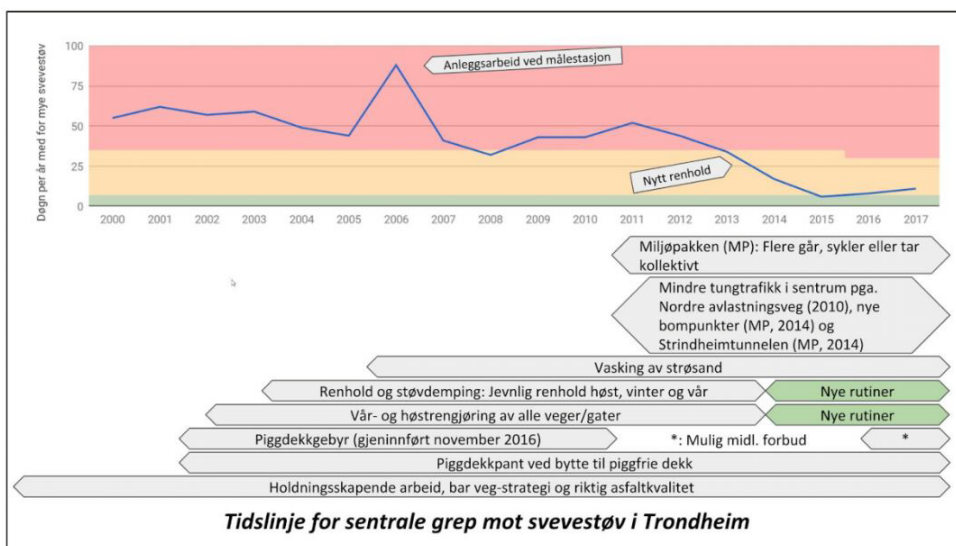
10. Informasjonen er hentet fra: Luftkvalitet i Trondheim 2017 Nettadresse: <https://sites.google.com/trondheim.kommune.no/luftrapport/start>

En av årsakene til de høye verdiene for moderat varslingsklasse var perioder med manglende NO₂ data. Av figuren til høyre ser vi også at forholdene har bedret seg betydelig siden 2009. Det skyldes hovedsakelig økt renhold og bruk av støvdempende midler på veier og gater.



FIGUR 4 VISER ANDELEN " (SVÆRT) DÅRLIG" LUFT ELLER " (SVÆRT) HØYT" FORURENSNINGSNIVÅ (JFR. TABELL 1) FOR DE ULIKE MÅLESTASJONENE. FOR SAMMENLIGNING AV NYE OG GAMLE VARSLINGSKLASSER: GRENSEN FOR "HØYT" FORURENSNINGSNIVÅ MTP. PM ER 20 % LAVERE ENN GRENSEN VAR FOR "DÅRLIG" LUFT.

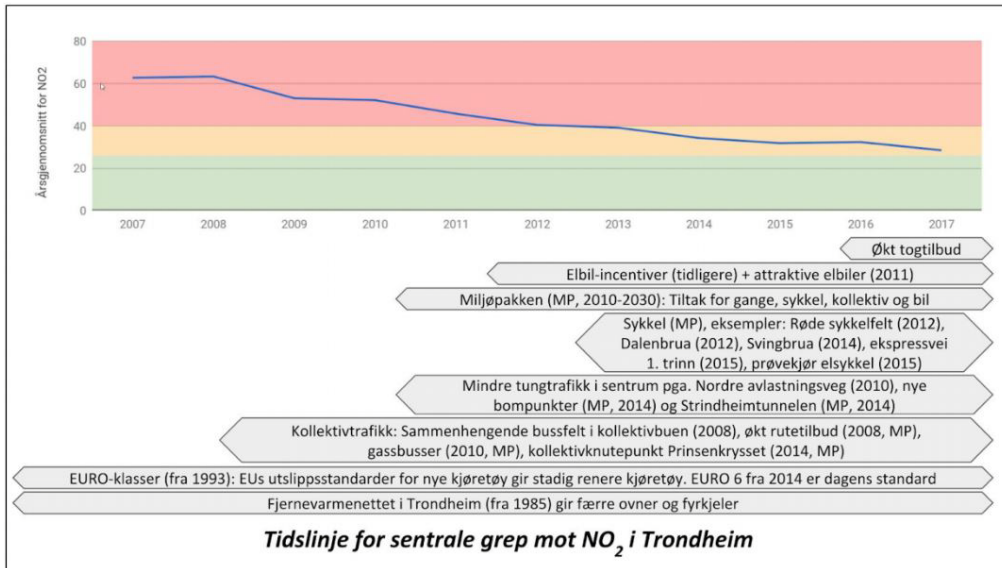
Det er gjort en rekke tiltak for å senke utslippet av svevestøv, NO₂ og klimagasser i Trondheim (se figur 5 og 6). Siden 2008, er mange av disse tiltakene innført eller intensivert i regi av Miljøpakken.



Figur 5 viser generelt trafikkreduserende tiltak og spesifikke tiltak rettet mot svevestøv (PM₁₀), samt målinger fra verste målestasjon.



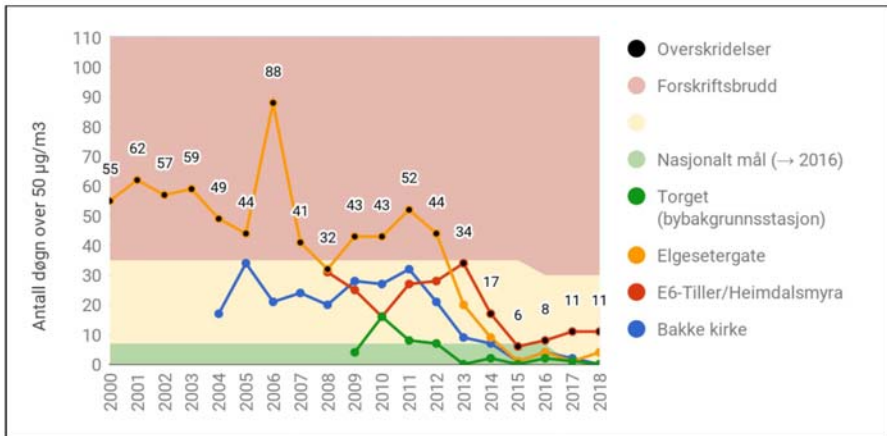
Tilsvarende tiltak er gjort for å redusere utslipp av NO₂.



Figur 6 viser generelt trafikkreduserende tiltak og spesifikke tiltak rettet mot NO₂, samt målinger fra verste målestasjon.

Kilder til svevestøv i Trondheim

I Trondheim er det støv fra veislitasje som følge av utstrakt bruk av piggdekk på bar asfalt som vanligvis er hovedårsaken til overskridelsene



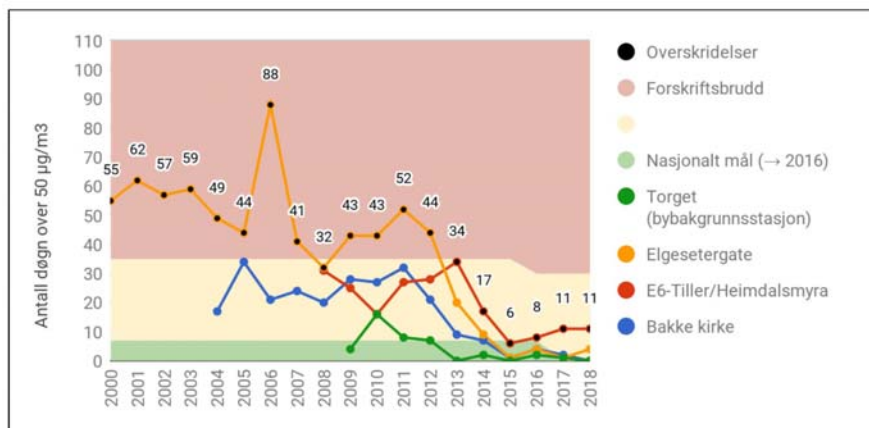
Figur 9 viser antall døgn med overskridelser av grenseverdien for svevestøv (PM₁₀) i perioden 2000 - 2018. Det var anleggsarbeid ved Elgeseter i 2006, Heimdalsmyra i 2013 og i nærheten av E6-Tiller i 2017 og 2018.

av grenseverdikrav for PM₁₀. Lokalt eller i perioder kan bygge- og anleggsvirksomhet være den viktigste kilden - som ved E6-Tiller i 2017, i forbindelse med utbyggingen av E6-sør. Bruk av piggdekk på bar asfalt var medvirkende årsak. Andre kilder til svevestøv er eksos og fyring. Disse gir mest fint svevestøv (PM_{2,5}). Hovedvegene (Elgeseter og E6-Tiller) har mest PM_{2,5} fra eksos, mens målestasjonene på Bakke kirke og Torget (bybakgrunnsstasjon) kan ha betydelig innslag av

PM_{2,5} fra fyring i kalde vintre. Støvmengden i lufta påvirkes av været. Særlig regn har en effekt siden det "vasker" lufta og hindrer oppvirvling av støv. Vind påvirker også ved å tørke opp vege og dermed spre støvet.

Antall døgn over forurensningsforskriftens grenseverdi er vist i figur 9. Folkehelseinstituttets (FHI) luftkvalitetskriterier – eller anbefaling ut i fra et helseperspektiv – er at døgnmiddel PM₁₀ er maksimalt 30 µg/m³. Frem til 2015 var kriteriet 35 µg/m³.

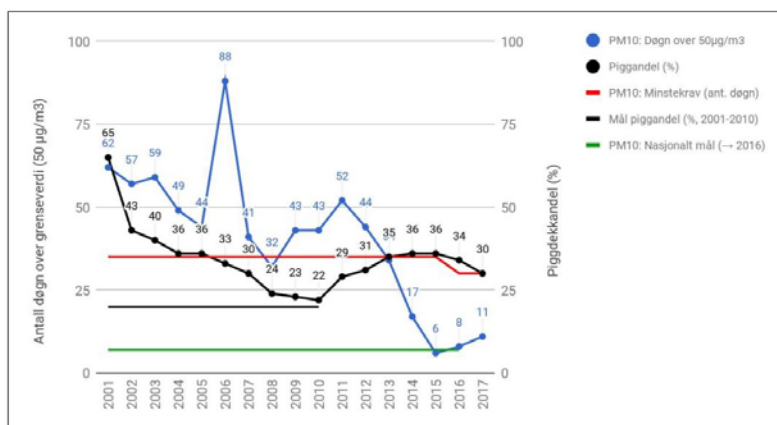
Oppvirvling av vegstøv var hovedårsaken til samtlige overskridelser av døgnmiddelkravet for PM₁₀ (se figur 9). Hovedkilden til



Figur 9 viser antall døgn med overskridelser av grenseverdien for svevestøv (PM₁₀) i perioden 2000 - 2018. Det var anleggsarbeid ved Elgeseter i 2006, Heimdalsmyra i 2013 og i nærheten av E6-Tiller i 2017 og 2018.

vegstøvet på den mest foruren- sede målestasjonen, E6-Tiller, var omfattende massetransport i forbindelse med utbyggingen av E6-sør i 2017–18. Bruk av piggedekk på bar asfalt var medvirkende årsak. På de andre målestasjonene kom overskridelsene i løpet av to dager med kaldt vær og høytrykk, som ga oppblomstring av vegstøv fra bruk av piggedekk på bar asfalt.

Boligoppvarming bidro ikke nevneverdig til overskridelser på noen av målestasjonene i 2017. Figur 11 viser forholdet mellom piggedekkkandel og svevestøv i perioden 2000–2017. Piggedekkkandelen sank og lufta ble bedre da vi hadde piggedekkgbeyr (2001–2010).



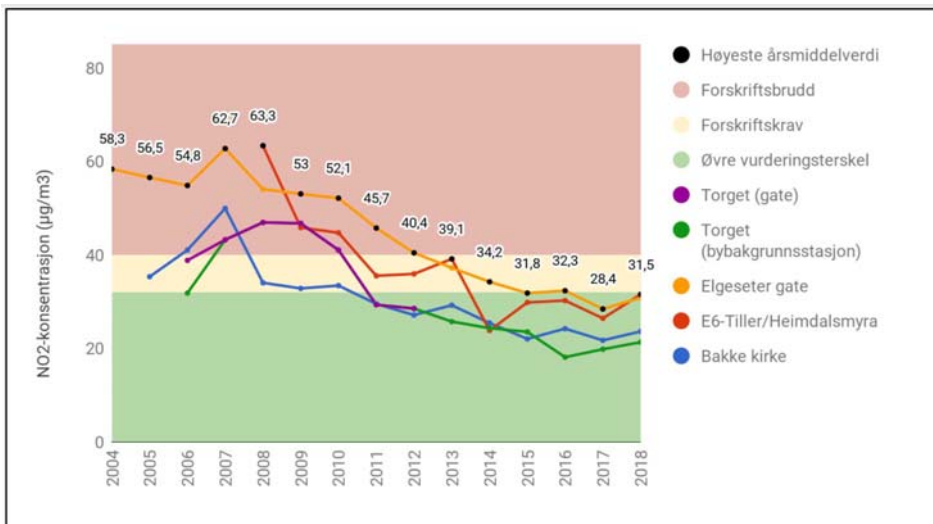
FIGUR 11: PIGGEDEKKANDEL OG OVERSKRIDELSER AV DØGNMIDDELKRAVET FOR PM₁₀. ELGESETER HAR HATT FLEST OVERSKRIDELSER FREM TOM. 2012, MENS E6-TILLER/HEIMDALSMYRA HAR FLEST OVERSKRIDELSER FRA 2013. RENERE LUFT FRA 2013 SKYLDES I HOVEDSAK RENHOLD OG STØVDEMPING.



Bedre asfaltkvalitet og andre tiltak antas også å ha hatt effekt på luftkvaliteten. Den absolutt største forbedringen kom imidlertid etter nytt renhold fra 2013: Sammenhengen mellom piggdekkandel og antall døgn med for mye svevestøv forsvinner med det nye renholdet. Økt piggdekkandel vil imidlertid kunne øke utgiftene til renhold og støvdemping, og gi dårligere luftkvalitet langs veger uten jevnlig renhold og støvdemping gjennom vinteren.

Nitrogenoksider i Trondheim

Trondheim hadde dessverre for lav datadekning på de mest forurensede målestasjonene i 2017. Målingene fra 2018, samt trend siste år tilsier likevel at grenseverdiene for NO_2 ble overholdt også i 2017. Forurensningsforskriften stiller krav om maks 18 timer over $200 \mu\text{g}/\text{m}^3$ per år, og maks $40 \mu\text{g}/\text{m}^3$ som gjennomsnittskonsentrasjon gjennom året (årsmiddelverdi). Det har ikke vært målt noen timer over $200 \mu\text{g}/\text{m}^3$ siden 2011. Høyeste årsmiddelverdi i 2018 ble målt på E6-Tiller, med $31,5 \mu\text{g}/\text{m}^3$ (se figur 8 under). I tillegg har Folkehelseinstituttet et luftkvalitetskriterium på $100 \mu\text{g}/\text{m}^3$ målt med en times midlertidstid.



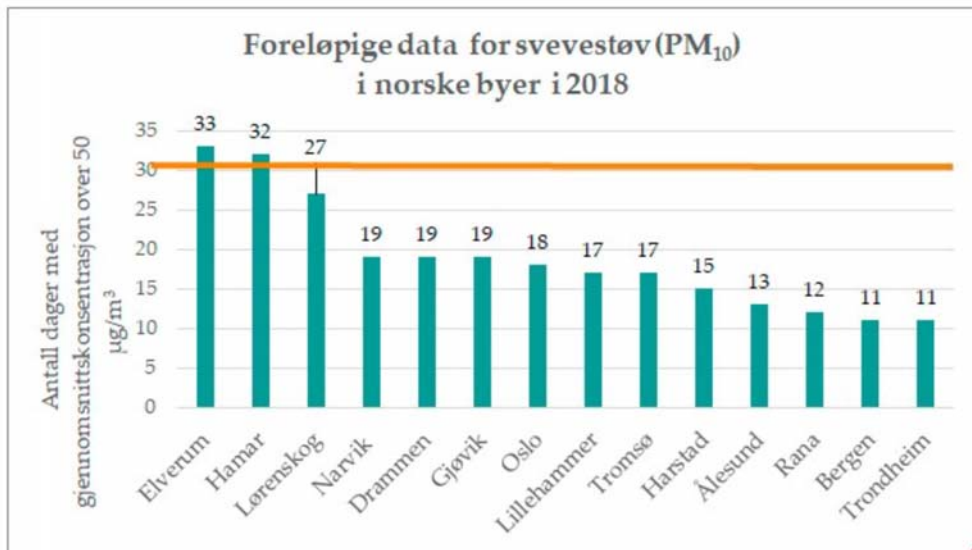
Figur 8: Årsmiddelverdi for NO_2 på alle målestasjoner. Merk at verdiene for 2017 er for lave som følge av manglende data fra november og desember. 2018 hadde dermed den laveste årsmiddelverdien som er målt i Trondheim, tross en relativt kald vinter.

Figur 8 viser årsmiddelverdien for NO_2 -konsentrasjonen i Trondheim har gått sterkt ned i perioden 2004–2015. Reduserte utslipp fra personbiltrafikken er trolig hovedårsaken: Veksten i personbiltrafikken har stagnert; bilene har lavere utslipp; kollektivtrafikken har økt kraftig; busene har skiftet drivstoff fra diesel- til gassbuss. Ikke minst har satsingen på gange, sykkel og elbiler gitt mange utslippsfrie trafikanter, mindre kø og dermed lavere utslipp fra andre trafikanter. Samtidig har det blitt mindre tungtrafikk gjennom sentrum etter bygging av Nordre avlastningsvei og differensierte bomtakster som leder trafikken via omkjøringsvegene. Utbygging av fjernvarmenettet har også gitt mindre fyring, sammen med stadig færre oljefyrer.

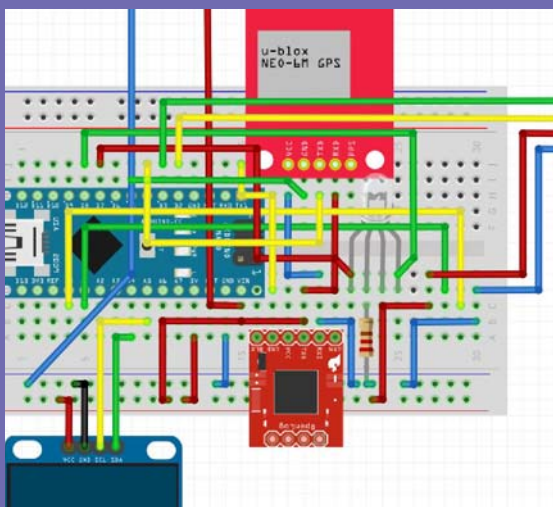
Utviklingen ser etter 2015 ut til å ha stabilisert seg. Hovedfartsårene fra sør og øst er fortsatt utsatte områder, sammen med innfartsårene til sentrum, Singsaker-ringen, samt ved hovedveiene på Heimdal og i Ila. Våre tiltak mot NO₂ er listet opp i kapitlet om tiltak mot luftforurensning (over).

Hvordan ligger Trondheim an i forhold til andre norske byer?

Trondheim gått fra å være en “værsting” til å ble omtrent best i klassen de seneste årene. Dette gjelder både for svevestøv og for NO₂. Diagrammet under viser måleresultater for antall døgn med en gjennomsnittlig partikkelkonsentrasjon over 50 µg/m³ for noen norske byer



Svevestøv: Grafen viser det høyeste antallet døgn målt med en konsentrasjon av svevestøv (PM₁₀) over 50 mikrogram per kubikkmeter luft i ulike kommuner i Norge i 2018. Den oransje linjen viser grenseverdien. Hamar og Elverum brøt grenseverdien i fjor. | *entral database for lokal luftkvalitet (SDB)*



Opplegget ble laget i forbindelse med Science Camp 2019 i Trondheim. Heftet er et forsøk på å legge opplegget til rettet for ToF-faget. Vi har derfor gått vesentlig grundigere til verks med å beskrive framgangsmåte og å bygge opp programmet.

For å lette uttesting og oppbygging så tar vi for oss del for del og tester ut individuelt før det hele settes sammen. Vi antar at det er gunstig for bedre å forstå hvordan kretsene fungerer.

Opprinnelig var det Trøndelag fylkeskommune som kostet utviklingen av prosjektet som bygger på et lignende prosjekt, Air:bit utviklet ved Universitet i Tromsø.

Nils Kr. Rossing

Dosent ved Skolelaboratoriet

E-post: nils.rossing@ntnu.no

Prosjektleder ved Vitensenteret

E-post: nkr@vitensenteret.com

NTNU



Trondheim

Institutt for
fysikk

Skolelaboriet
for matematikk, naturfag
og teknologi

Tlf. 73 55 11 43

<https://www.ntnu.no/skolelab>