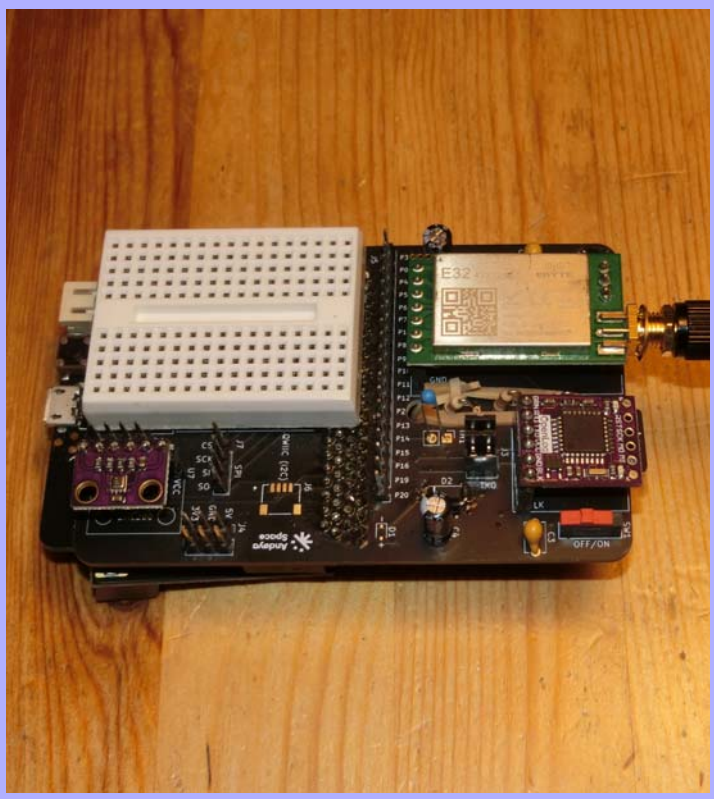
Rev. 3.6

Nils Kr. Rossing, Jørn Hafver, Fredrik Motland Kirkemo, Simen Bergvik og Kristian Enoksen

Micro:bit CanSat

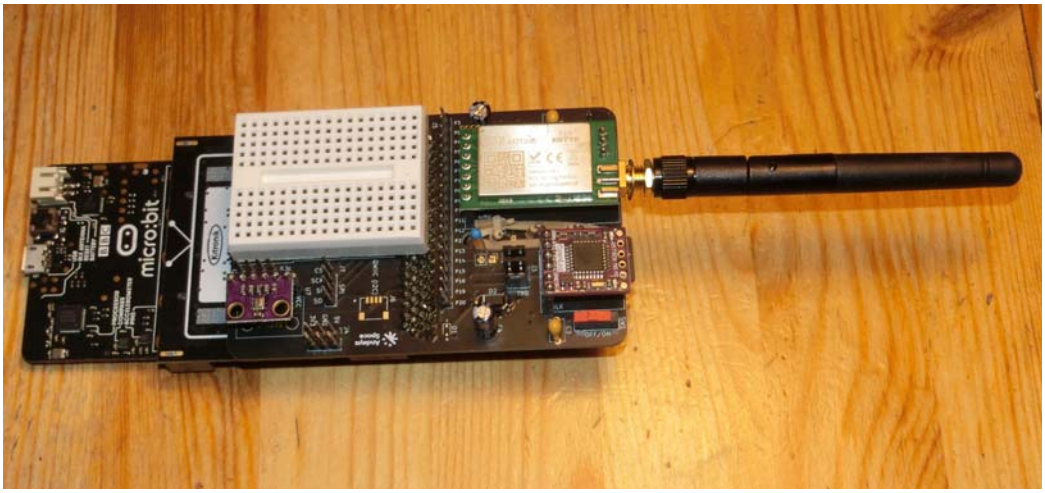


NTNU
Trondheim
Institutt for fysikk
Skolelaboratoriet
for matematikk, naturfag
og teknologi

April 2024

Micro:bit CanSat

Nils Kr. Rossing (NTNU, Skolelaboratoriet), Jørn Hafver og Fredrik Motland Kirkemo (Jærmuseet), Simen Bergvik og Kristian Enoksen (Andøya Space Education)



Micro:bit CanSat

Trondheim 2024

Layout og redigering:	Nils Kr. Rossing, Skolelaboratoriet ved NTNU
Tekst og bilder:	Nils Kr. Rossing, Skolelaboratoriet ved NTNU Fredrik Motland Kirkemo, Jærmuseet
Andre bidragsytere:	Jørn Hafver, Jærmuseet Simen Bergvik, Andøya Space Education Kristian Enoksen, Andøya Space Education
Trykk:	NTNU Grafisk senter
Forsidebilde:	Prototyp CanSat (Foto: Nils Kr. Rossing)

Faglige spørsmål rettes til:
Skolelaboratoriet for matematikk, naturfag og teknologi
Institutt for fysikk
v/ Nils Kr. Rossing nils.rossing@ntnu.no

Skolelaboratoriet ved NTNU
Realfagbygget,
Høgskoleringen 5,
7491 Trondheim

Telefon: 73 55 11 91
https://www.ntnu.no/skolelab/

Rev. 3.6 – 29.04.24

Undervisningsopplegget er utviklet i et samarbeid mellom Skolelaboratoriet, Jærmuseet, og Andøy Space Education.

Forord

Siden 2009 er det holdt CanSat-kurs ved rakettskytefeltet ved Andøya, opprinnelig av *NAROM* (NAsjonalt senter for ROMrelatert opplæring), senere under navnet *Andøya Space Education (ASE)*. Fra 2010 i samarbeid med Aalborg Universitet ved Jens Dalsgaard Nielsen og Skolelaboratoriet ved NTNU ved Nils Kr. Rossing.

Opp gjennom årene er det både holdt CanSat videreutdanningskurs (2011 og 2012) og etterutdannings kurs (2013–2022) for lærere i samarbeid med Skolelaboratoriet ved NTNU[1]. Videreutdanningskurset ble opprinnelige støttet økonomisk av Norsk Romsenter og Naturfagsenteret, og etter hvert da kurset ble holdt som etterutdanningskurs, kun av ESA (ESERO) og NTNU (m/timer). Kursene har hovedsakelig vært holdt ved Andøya Rakettsytefelt, men flere kurs er også holdt ved NTNU i Trondheim. I tillegg er det holdt en rekke CanSat-kurs av personell fra ASE flere steder i Norden. Alle kursene har dessuten vært tilbudt lærere fra hele Norden, hvilket har medført at noen er holdt på engelsk.

I starten gikk kurset over flere dager (4 – 5), men etter hvert ble kurset redusert til et 2-dagers kurs. For lærere med liten erfaring i tekstbasert programmering, ble dette svært krevende. Fra 2018 ble det derfor tilbudt et forkurs i grunnleggende Arduino-programmering, utviklet i et samarbeid mellom ansatte ved ASE og NTNU. Det viste seg snart at de aller fleste av deltagerne på CanSat-kurset også valgte å melde seg på forkurset, hvilket understreket behovet.

Opprinnelig har CanSat-kurset vært tilbudt lærere i videregående skole som underviser i Teknologi og forskningslære. Da valgfaget Programmering i ungdomskolen ble opprettet, ble CanSat-kursene også tilbudt ungdomsskolelærere som underviste valgfag. Etter at Fagfornyelsen kom i 2020, fikk vi et økende tilsig av lærere fra elektro yrkesfag i videregående utdanning, hvilket til dels skyldtes at læreverkene bruker Arduino i opplæringen, men også at enkelte fag i yrkesopplæringen kan gi rom for større prosjekter ala CanSat.

Fra 2009 og fram til i dag er det utviklet en rekke ulike varianter av CanSat-settet som har vært brukt under kursene. Det første settet som ble brukt var satt sammen av tre deler, en mikrokontrollerdel, en sensordel og en radiodel som ble koblet sammen med kabler. Til kurset i 2012 hadde Aalborg universitet utviklet et CanSat shield for Arduino UNO, med trykk- og temperatursensor, akselerometer, SD-kort terminal og telemetriradio. Dette kortet kom i stadig nye utgaver (Rev. 1 – 6) til Christoffer Stausland og Bente Jensen ved ASE lanserte en helt ny CanSat-versjon i 2018 bygget opp rundt mikrokontrollerkortet Teensy 3.5 med innebygget SD-kort terminal, det hele montert på et kretskort med magnet-, akselerasjon- og gyrometer, trykk- og temperatursensor i tillegg til en kraftig sender/mottaker. Dette ga en langt kraftigere CanSat, men krevde montering av overflatemonterte komponenter som var krevende å lodde for nybegynnere. Programmeringen var stort sett som ved bruk av Arduino.

Etter Fagfornyelsen i 2020 kom programmering inn som tema i alle fag, da spesielt i naturfag og matematikk, og Python ble etter hvert et foretrukket språk i både grunnskole, videregående skole og på universitet. Dessuten ble Micro:bit tatt i bruk på alle nivåer i skolen og med den også blok-

1. Det må her bemerkes at NAROM/ASE har holdt en rekke CanSat-kurs rundt om i Skandinavia uten Skolelaboratoriets medvirkning.

kode som gjorde det lettere å fokusere på programstruktur framfor syntaks. Flere har derfor spurt om ikke Micro:bit kan brukes i forbindelse med CanSat, siden mange skoler har disse settene på egen skole. Veien fra blokkode til Python og Java er heller ikke lang rent teknisk, editoren til micro:bit, MakeCode, tillater at man bytter mellom blokkode, Java skript og Python.

Med dette som bakgrunn ble det høsten 2022 bestemt at vi skulle starte utvikling av en Micro:bit CanSat i et samarbeid mellom Jærmuseet, Andøya Space Education og Skolelaboratoriet ved NTNU. En første prototyp var klar i oktober 2023 som ble testet ut overfor elever fra Talentsenteret ved Jærmuseet november 2023 – januar 2024. Deretter vil det bli holdt et tradisjonelt CanSat-lærerkurs i Sandvika nær Oslo 15. til 17. april 2023 i samarbeid med Skolelaboratoriet og Jærmuseet.

Gjennom alle disse årene har CanSat-kursene hatt som målsetning å forberede elevgrupper i grunn- og videregående skole til en CanSat-konkurranse på Andøya i mars/april, der målsetningen har vært å kåre både nasjonale og nordiske vinnere som kvalifiserer for å delta i den europeiske konkurransen. Konkurransen går ut på at de demonstrerer og dokumenterer sin egen hjemmebygde CanSat, konstruert etter internasjonale krav til kostnad, størrelse og vekt. Andøya har også vært vertskap for den europeiske konkurransen.

En takk til Skolelaboratoriet ved NTNU som har latt meg (Nils Kr. Rossing) arbeide med micro:bit CanSat-prosjektet gjennom hele 2023 uten ekstern finansiering. Også en takk til *Carl Angell* i redaksjonen i tidsskriftet *Fra fysikkens verden* (FFV), som oppmuntret meg til å skrive og publisere artikkelen *Lag en barometrisk høydemåler* i FFV 1/2024 som bl.a. er "spinn off" fra CanSat-prosjektet. Og en spesiell takk til *Øyvind G. Grøn* i den samme redaksjonen, som klarla og utledet den barometriske formelen som gjennom mange år er blitt brukt for å estimere høyde på bakgrunn av lufttrykk [3].

Skolelaboratoriet ved NTNU
April 2024

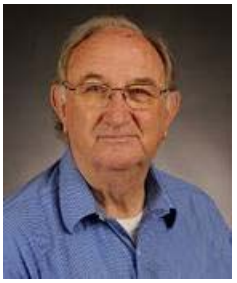
Nils Kr. Rossing (Skolelaboratoriet)
Jørn Hafver (Jærmuseet)
Fredrik M. Kirkemo (Jærmuseet)
Simen Bergvik (Andøya Space Education)
Kristian Enoksen (Andøya Space Education)

Innhold

1 Innledning

CanSat er et elektronikk og programmeringsprosjekt for bruk i undervisningen over hele verden. Prosjektet har pågått gjennom mange år.

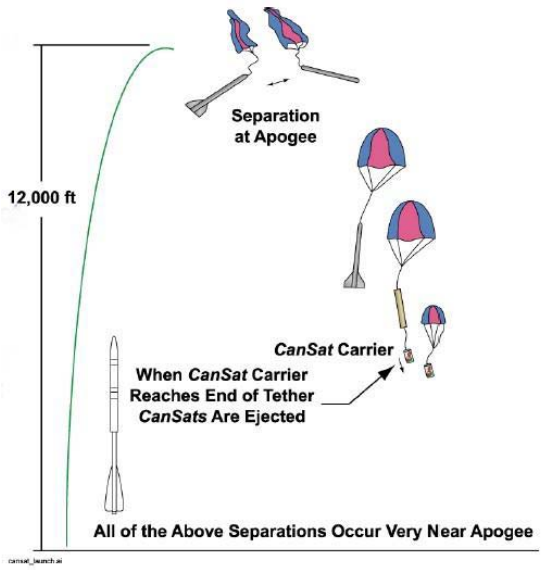
I 1998 møttes rundt 50 studenter og ansatte fra 12 universiteter i USA og Japan til et symposium på Hawaii. Det var det første "University Space Systems Symposium". Her møtte også Robert Twiggs (1935 –), professor emeritus ved Stanford University, opp med ideen til det som senere skulle bli nanosatellittprosjektene. Ideen var å sende en sonde på størrelse med en brusboks ut i "verdensrommet". Volumet skulle være rundt 350 milliliter og massen, omtrent 500 gram. Dette førte til et prosjekt som startet i 1999 kalt ARLISS og som hovedsakelig involverte studenter fra amerikanske og japanske universiteter. Prosjektet ble lansert den 11. september samme år og har vært gjennomført hvert år siden uten avbrudd.



Robert Twiggs (1935 -)

Den opprinnelige ideen var å skyte opp 3 satellitter på 350 milliliter, eller en satellitt med større volum. For å løfte satellitten opp til ca. 4000 meter, ble det benyttet en rakett som kunne løfte 1,8 kg til den nevnte høyden, noe som åpnet for målinger i nedre del av atmosfæren til en overkommelig pris (ca. kr. 4000,-).

I år 2000 var oppdragene betydelig endret. For eksempel var kravet at man skulle lage et landingssystem ved å bruke data fra en barometriske høydemåler eller GPS. Prosjektet ble vesentlig mer krevende i 2001 da "ComeBack"-kategorien ble lagt til, som krevde at satellitten også skulle returnere til et spesielt sted etter landing. Det nye oppdraget var svært vellykket. I 2002 gikk studentene ved Space Robotics Lab ved Tohoku University av med seieren ved å ende opp 45 meter fra målet, og i 2006 endte vinneren 6 meter fra målet.



Interessen for denne typen satellitter har vært økende og har spredt seg over hele verden. I 2003 plasserte University of Tokyo to satellitter av typen CubeSat i bane rundt jorda. CubeSat er satellitter formet som en kube litt større enn en CanSat. De siste årene har flere konkurranser blitt utviklet etter det samme konseptet foreslått av prof. Robert Twiggs og benyttet i forbindelse med ARLISS, både nasjonalt og internasjonalt.

I Norge ble CanSat lansert i 2009 av NAROM ved Andøya rakettskytefelt, nå Andøya Space Education (ASE). Senere er det gjennomført årlige lærerkurs med påfølgende undervisning i

klasserom over hele landet. I mars/april har de beste lagene i hvert land konkurrert om å få delta i den europeiske konkurransen. Det har også vært sendt norske lag til den amerikanske konkurransen (Heimdal vgs). Den nordiske konkurransen har flere ganger vært arrangert på Andøya.

Ved hjelp av et byggesett skal det bygges en liten sonde (CanSat) som skal skytes ut fra en rakett, slippes fra ballong eller drone, fra 300 – 1000 meters høyde. Sonden skal være på størrelse med en brusboks, derav navnet Can-Sat. og ha en maksimal vekt på 350 g. Sonden utstyres med en mikrokontroller og diverse sensorer. En radiosender sender (telemetrerer) de avleste målingene til en mottaker på bakken (bakkestasjon). Sendingen foregår i et ISM-bånd vanligvis omkring 434 MHz (ca. 70 cm bølgelengde). Sonden skytes ut eller slippes, og faller kontrollert mot bakken i fallskjerm. Sensorer samler inn måledata under fallet som kontinuerlig overføres til bakkestasjonen samtidig som dataene lagres på SD-kort ombord i sonden.

Minimumsspesifikasjoner

Det stilles følgende krav til sonden (Primary mission):

1. Sonden skal minimum måle lufttrykk og temperatur hvert 3. sekund.
2. Sonden skal overføre data til bakkestasjonen under fallet via radio.
3. Sonden skal bygges slik at den får plass i en 330 ml standard brusboks, eller slik at ingen deler av sonden stikker ut over omfanget til en slik boks. Selve boksen kan derfor bygges av andre materialer, evt. 3D-printes om ønskelig.
4. Boksen skal maksimalt veie det samme som en full brusboks, dvs. 350 g.
5. Sonden skal drives med batteri eller solcellepaneler.
6. Antennen skal være fleksibel og ikke stikke ut mer enn ca. 10 cm når den er stuvet sammen ved oppskyting.
7. En fallskjerm skal være forsvarlig festet til den ene enden av boksen.
8. Fallhastigheten skal være mellom 8 og 11 m/s.

Krav til bakkestasjonen

Bakkestasjonen skal...

1. ... kunne motta på senderfrekvensen til senderen i sonden (ca. 434 MHz)...
2. ... og kan være utstyrt med en rettningsantenne som kan følge sonden i fallet.
3. ... motta alle data fra sonden under fallet og lagres i en PC for senere analyse og presentasjon

Krav til bærerakett

Bæreraketten skal...

1. ... løfte sonden opp til toppunktet for ferden hvor en eller flere (to) sonder skytes ut fra hver rakett
2. ... en forsinket ladning inne i raketten skyter ut sondene slik at den kan falle fritt mot bakken
3. ... minimum kunne løfte sondene til ca. 800 meter

Under uttesting har det også vært vanlig å benytte værballong i snor eller drone for å løfte sondene opp til en passende høyde før den slippes. Under selve konkurranse har det vært brukt små faststoffraketter (ca. 1,5 m lang).

NB! Både oppskyting og slipp fra ballong må avtales med de lokale myndigheter, normalt nærliggende flyplasser.

1.1 Bakgrunn for valg av teknologi

Dersom man oppfyller kravene over, står man fritt til å velge teknologi. Helt fra begynnelsen i 2009 ble det brukt et mikrokontrollerkort bygget opp omkring ATMEGA168, en tidlig variant av Arduino. I tillegg hadde den en temperatur- (TMP37) og en lufttrykksensor (MPX4115A) på et eget sensorkort, et senderkort med en egen mikrokontroller (ATMEGA88) og en radiosender (ADF7012).

Allerede i 2012 ble det utviklet et nytt konsept som bygget på Arduino UNO med et spesialbygget shield-kort utviklet ved universitetet i Aalborg med trykk- og temperatursensor (LM35/NTC), en trykksensor (MPX4115A), et akselerometer (MMA7361L), en SD-kort terminal (OpenLog) og en liten radiosender (APC220). I løpet av de neste årene kom det stadig nye varianter av shield-kortet og temperatur- og trykksensoren ble byttet ut med en IMU (Inertial Measurement Unit) GY91 som inneholdt akselerometer gyrometer, magnetometer, temperatur- og trykkmåler.

I 2018 ble det utviklet et nytt kontrollerkort som var bygget opp omkring en langt kraftigere mikrokontroller, Teensy 3.5 med innebygget SD-kortterminal, senderen RFRFM-96-433 og sensorkortet GY91 (IMU), i tillegg var NTC-motstanden beholdt. Sistnevnte teknologiske løsning baserte seg på lodding av overflatemonterte komponenter, hvilket var krevende for utrente studenter.

Fra 2020 er programmering kommet inn i alle fag i skolen, hvilket har medført økt interesse for programmering, samtidig som det har vært en betydelig satsing på BBCs Micro:bit, hvilket har ført til at den sistnevnte har fått en betydelig utbredelse i både grunnskolen og i videregående skole. Det er derfor mange som spør hvorfor ikke Micro:bit kan brukes i forbindelse med CanSat, gjerne med mulighet til å programmere den med blokkode eller Python.

Dette er bakteppet for å lage en Micro:bit CanSat.

1.2 Kursets oppbygning

Det er utviklet et CanSat-byggesett med en Micro:bit versjon 2 (V2). Det er stort sett valgt hullmonterte komponenter slik at settet skal være lett å lodde sammen. Kantkontakten til Micro:biten kan evt. leveres ferdig montert i tillegg til en hylsekontakt eller stiftlist slik at alle porter er tilgjengelige. Ett mini koblingsbrett kan evt. limes fast på kortet. Dermed har man tilgjengelig et lite testlaboratorium. I tillegg lånes det ut et skjøtekort med OLED-display som kobles opp mellom Micro:biten og kantkontakten så det skal være enklere å lese av måleverdier. Dette er spesielt nyttig ved opplæring i programmering.

Man kan nå velge en av to alternative veier:

1. Bruke koblingsbrettet til å koble opp enkle kretser for å trene blokkprogrammering for deretter å lodde opp kretsen:
   - Tenning og slukking av lysdiode
   - Skriving til OLED-display
   - Bruk av temperatursensor (NTC)
   - Bruk av trykksensor (BMP/E280)
   - Omregning til relativ høyde
   - Nullpunktsjustering av høydemåleren
   - Montere og lodde opp kretsen
2. Lodde opp kretsen først slik at den er ferdig montert, for så å bruke den oppkoblede kretsen til å lære blokkprogrammering og samtidig kontrollere at kretsene virker som de skal.
   - Sensorkortet (BMP/E280), SD-kortterminalen (OpenLog) og radioen (E32) monteres i sokler slik at de kan inkluderes etter behov.
   - Gjennomføre et utvalg av punktene under punkt 1 med de monterte komponentene
   - På denne måten får man opplæring i blokkode samtidig som man får testet at oppkoblingen fungerer som den skal.

Deretter går man videre:

1. Koble opp radioen med antenne
2. Konfigurere og inkludere radioen
3. Opprette radiokontakt og skrive til SD-kort
4. Bruk av terminalprogram for mottak og lagring av data, evt. bruk av et ekstra CanSat-kort
5. Komplettere programmet
6. Lage fallskjermen
7. Gjennomføre et slipp og ta ned data
8. Analysere og presentere data

Forslag til tredagers kurs etter vei 2:

Dag 1 – Montering av CanSat

- Introduksjon til CanSat
- Montering (lodding) av CanSat
- Oppstart grunnkurs

Dag 2 – Grunnkurs

- Fortsettelse av grunnkurs
- Introduksjon til sensorer, bruk og virkemåte
- Måling av lufttrykk og temperatur
- Beregning av høyde

Dag 3 – Ferdigstilling og test av CanSat

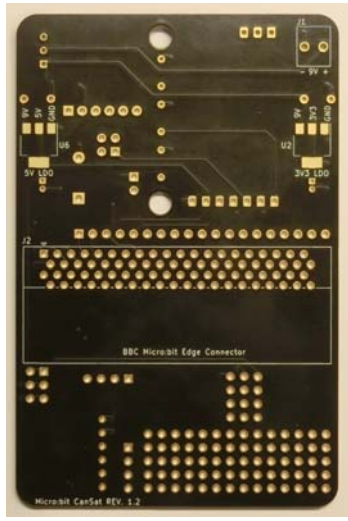
- Oppsett og test av radiokommunikasjon/lagring på SD-kort
- Slipp fra drone
- Kort om analyse av måledata
- Avslutning

2 Bygging av Micro:bit CanSat

Kapittelet beskriver hvordan vi monterer og lodder opp Micro:bit CanSat-en.

1. **Komponent- og loddesiden av kretskortet**
   Normalt har et kretskort en *komponentside,* der alle komponentene skal monteres og en *loddeside,* der all lodding foregår. Kort som er litt mer kompliserte kan ha komponenter på begge sider slik som i vårt tilfelle. Regelen er at komponentene plasseres på den siden som viser omrisset av komponenten og stort sett loddes på den andre siden, loddesiden.

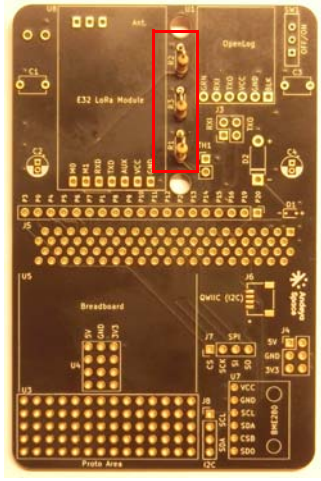
Kretskort – komponentside



Kretskort – loddeside

2. **Montering av tre like motstander (10kOhm)**
   Motstander kan ha litt forskjellig utforming. Er de tilstrekkelig små kan de plasseres liggende. Er de litt større kan de settes på høykant som vist under til venstre.


Brun, sort, oransje, gull 10kOhm

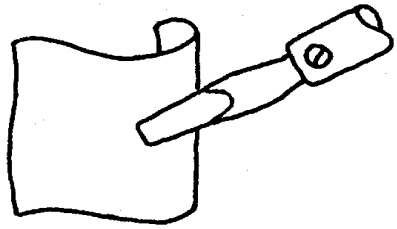


Alle motstandene vi bruker er på 10kOhm, med fargekoden: Brun, sort, brun, gull. Det spiller ingen rolle hvilken vei motstandene monteres.

3. **Loddekurs – Lodding av motstandene**
Etter at motstandene er montert og beina stukket gjennom kretskortet, skal de loddes til kobberbanene på loddesiden. Kobberbanene forbinder de enkelte komponentene slik at CanSat-en fungerer slik den skal. Å utføre riktig lodding er viktig. Dersom loddingene ikke blir gode nok, kan vi lett risikere at ingenting virker som det skal.

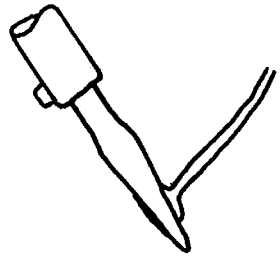
1.

Se til at loddebolten er ren for loddeslagg. Tørk av spissen med en fuktig klut e.l. mens den er varm.



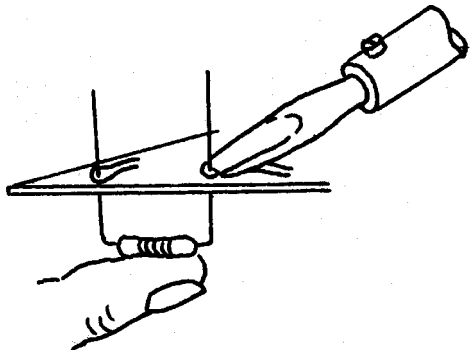
2.

Etter at loddebolten er rengjort, fortinnes begge sidene med litt tinn.



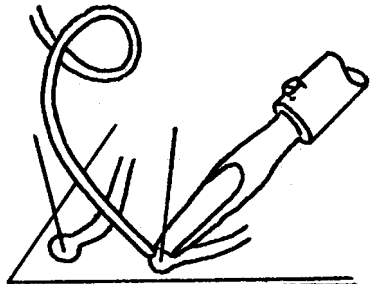
3.

Monter komponenten og varm opp loddestedet og beina på komponenten samtidig.

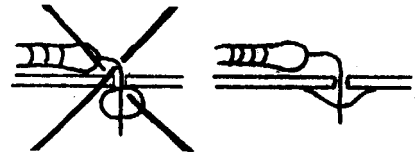


4.

Tilfør loddetinn der loddebolten berører kobberbanen og beinet på komponenten, slik at det smelter utover. Ikke varm for lenge.



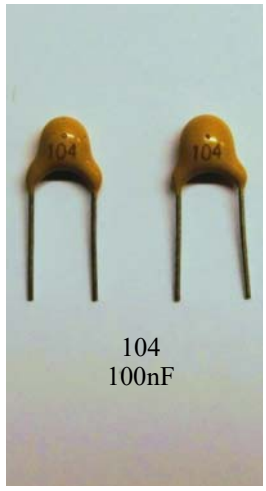
Se til at loddingen ikke er en kaldlodding. For at loddingen skal være god, bør loddetinnet ha flytt utover.

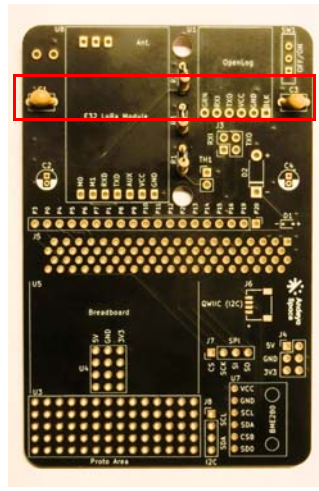


Etter at komponentene er loddet til kobberbanene, kan bena som er for lange klippes av inntil loddingen med en sideavbiter.

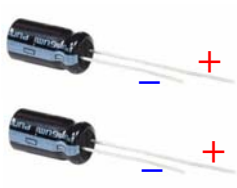
Monter og lodd alle motstandene.

4. **Montering av keramiske kondensatorer C1 og C3 (100nF)**
Kondensatorene er merket 104 (100nF). Det spiller ingen rolle hvilken vei de monteres. Beina stikkes gjennom kretskortet fra komponentsiden og loddes på loddesiden. Klipp av beina ved loddingen med sideavbiteren.

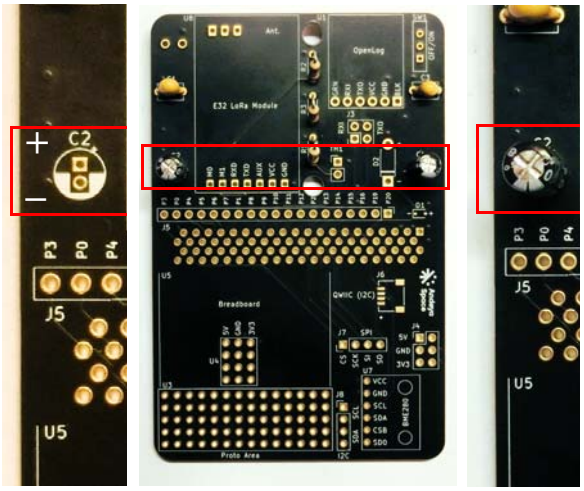




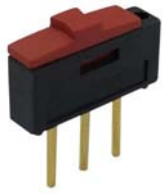
5. **Montering av elektrolyttkondensatorene, C2 og C4 (10µF)**
Elektrolyttkondensatorene er merket 10µF, og ***må monteres riktig vei***. Det er viktig at det lange beinet stikkes gjennom hullet merket med +. Beina stikkes gjennom kretskortet fra komponentsiden og loddes på loddesiden. Klipp av beinet ved loddingen med sideavbiteren.

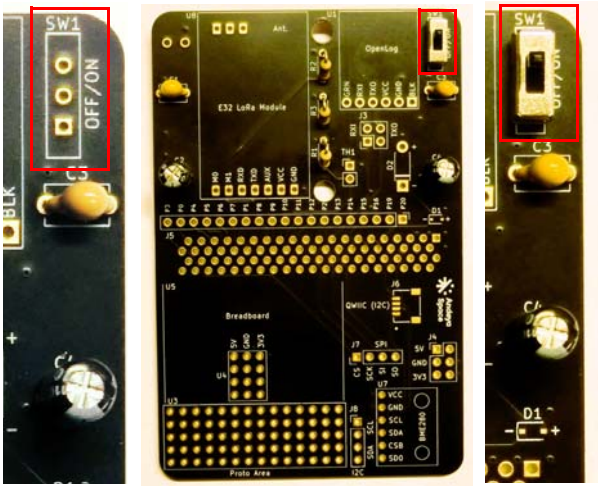


10µF

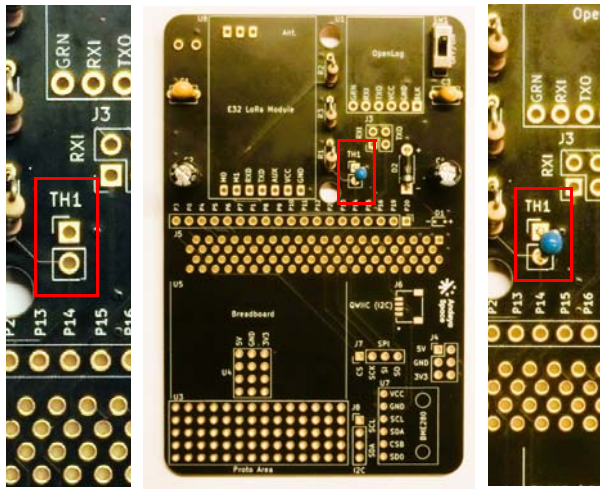


6. **Montering av Av/På-bryteren, SW1**
Det spiller ingen rolle hvilken vei bryteren settes. Stikk de tre beina gjennom kretskortet fra komponentsiden og lodd på loddesiden. Klipp av beina ved loddingen med sideavbiteren.

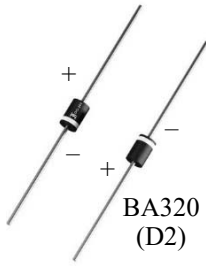

Av/På-bryter
(OFF/ON)

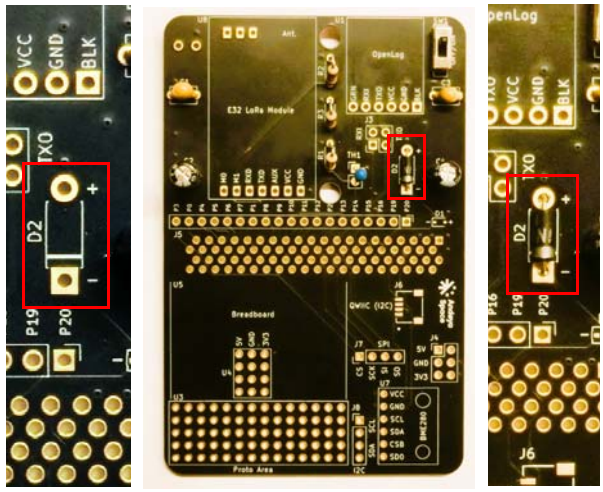


7. **Montering av NTC-motstanden (temperatursensoren), TH1**
Det spiller ingen rolle hvilken vei NTC-motstanden monteres. Stikk beina gjennom kretskortet fra komponentsiden og lodd på loddesiden. Klipp av beina ved loddingen med sideavbiteren.

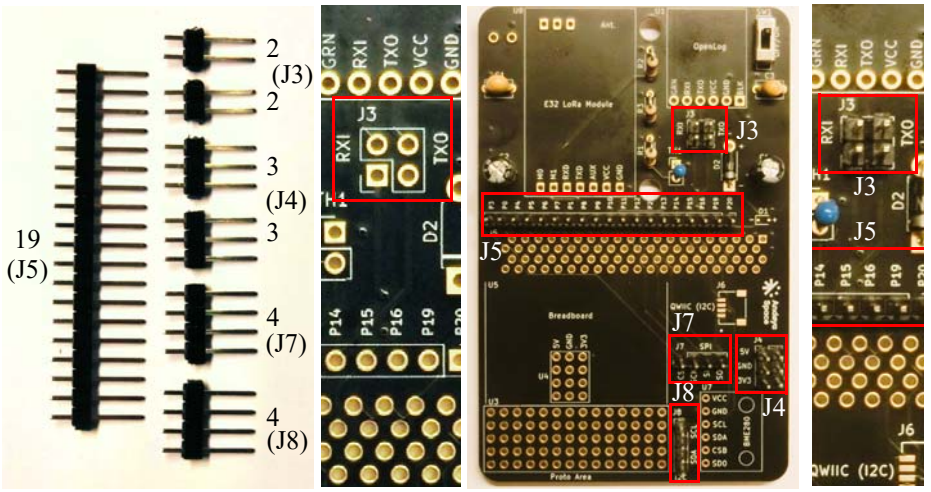


8. **Montering av Schottky-dioden, D2**
Schottky-dioden (D2) må plasseres rett vei. Den blanke ringen på dioden viser hvor katoden er, dvs. den enden som skal være nærmest minuspolen. På kretskortet er dioden merket med + og –. *Den blanke ringen skal plasseres nærmest – tegnet.* Stikk beina gjennom kretskortet fra komponentsiden og lodd på loddesiden. Klipp av beina ved loddingen med sideavbiteren.





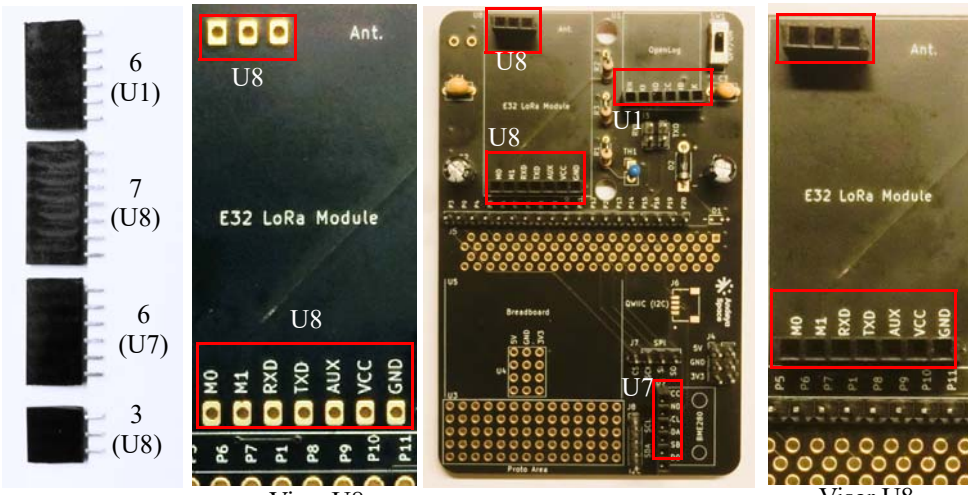
9. **Montering av stiftlistene, J3, J4, J5, J7 og J8**
Klipp opp siftlisten i riktige lengder (1 x 19, 2 x 2, 2 x 3, 2 x 4). Bruk sideavbiteren og klipp i hakket mellom to stifter. Stikk den korte enden av stiftene gjennom kretskortet fra komponentsiden og lodd på loddesiden.



Viser J3 før montering Viser J3 etter montering

Alterntivt kan man benytte hylsekontaker på alle kontaktpunkter unntat J3.

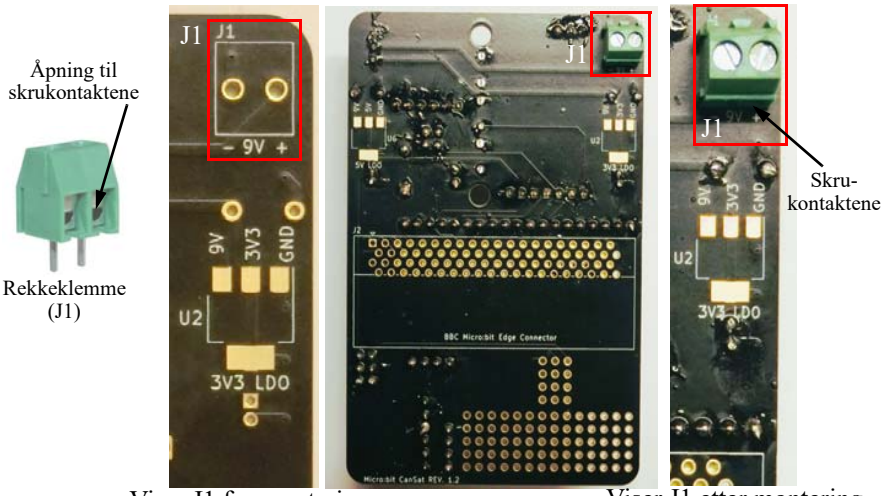
10. **Montering av hylsekontaktene, U1 (6), U7 (6) og U8 (3 + 7)**
Om hylsekontaktene ikke alt er klippet opp, så gjør det. Når du klipper opp hylsekontaktene, så må du klippe midt i en av kontaktene, som gjør at du vil miste en kontakt hver gang du klipper. Stikk loddepinnene gjennom kretskortet fra komponentsiden og lodd på loddesiden. Sørg for at kontaktene står loddrett på kortet.



Viser U8 før montering Viser U8 etter montering

11. **Montering av batterikontakt, J1**

Den grønne rekkeklemmen skal *monteres på loddesiden*. Denne kan monteres begge veier, men det kan være lurt å montere den slik at åpningene til skrukontaktene vender inn mot kortet siden det er gjort plass på kortet til batteriet. Stikk loddepinnene gjennom kretskortet fra komponentsiden og lodd på komponentsiden. Klipp av beina ved loddingen med sideavbiteren.

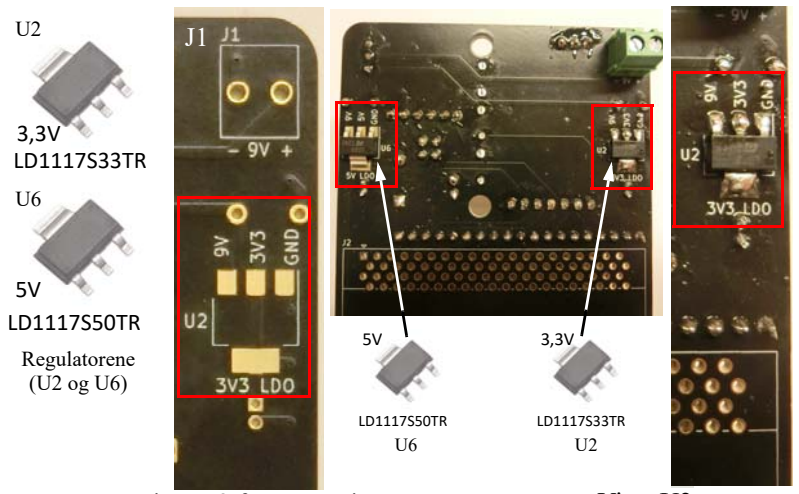


Viser J1 før montering

Viser J1 etter montering

12. **Montering av spenningsregulatorene, U2 og U6**

De to spenningsregulatorene skal *monteres på loddesiden.* Disse er overflatemonterte komponenter og har derfor ikke pinner som skal stikkes gjennom kretskortet, men tilkoblingspunkter som ligger på loddesiden. *De to komponentene er forskjellige og* ***må*** *ikke byttes om*. LD1117S33TR skal monteres på venstre side (U6) og LD1117S50TR på høyre side (U2) som på bildet. Se nøye på den trykte teksten på komponenten slik at det blir riktig.

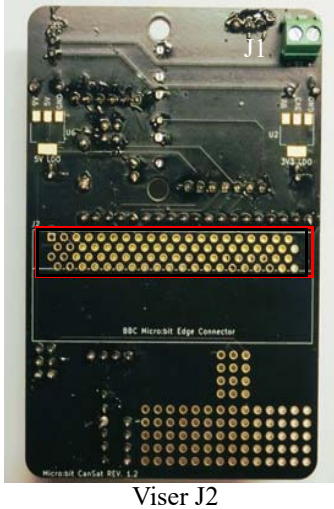


Viser U2 før montering

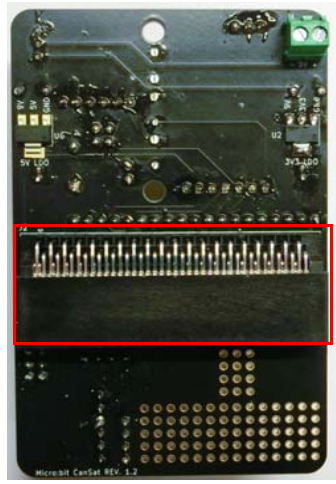
Viser U2 etter montering

13. **Montering av kantkontakten, J2**
Kantkontakten holder Micro:bit-en og skal monteres på loddesiden, men beina skal alle sammen gjennom kretskortet og loddes på komponentsiden. Pass på at kontakten ligger inn mot kretskortet. Lodd gjerne pinnene som sitter i hjørnene først slik at du er sikker på at den holder seg på plass. Pass på at det ikke blir loddebroer mellom kontaktene. Dersom det skulle skje, hold kortet slik at broa kommer på undersiden. Smelt broa med loddebolten slik at det overflødige tinnet renner ned på loddebolten.

Kantkontakt
(J2)



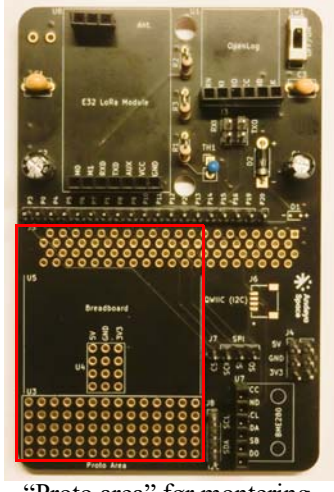
Viser J2
før montering



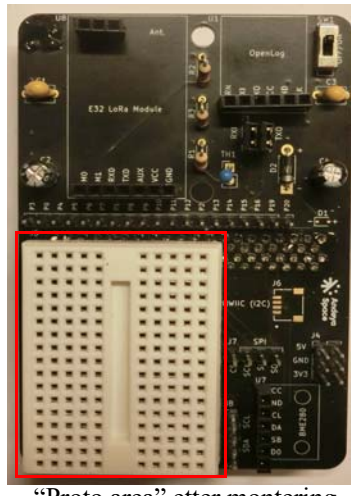
Viser J2
etter montering

14. **Montering av mini koblingsbrettet, "Proto area"**
Koblingsbrettet limes til "proto area" med den selvklebende tapen under eller en egen selvklebende tape. Alternativt kan man koble opp egen elektronikk i prototypområdet.

Koblingsbrettet
(Proto area)



"Proto area" før montering



"Proto area" etter montering

15. **Montering av batteri og batterikontakt**

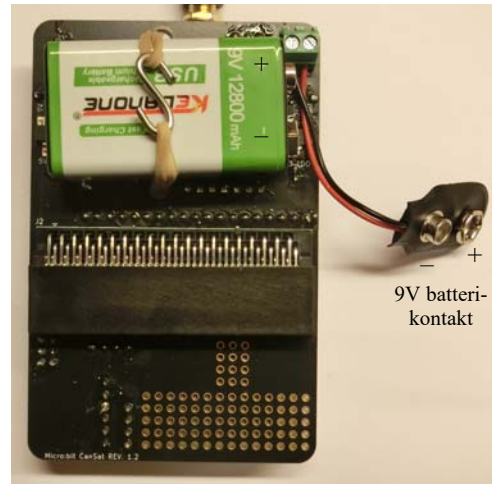
Vi har valgt å gjøre plass til et standard 9V batteri som kobles til kretsen vår med en batterikontakt. Et alternativ er å benytte et ladbart 9V LiPo-batteri. Pass på at den røde ledningen til batterikontakten kobles til + og den sorte til –. Vi har festet batteriet med en strikk som tres gjennom hullene i kretskortet og som evt. kan festes med en S-formet metallhake, se bildet under til høyre.

9V batteri-kontakt

9V batteri

9V LiPo-batteri

Batterikontakt før montering



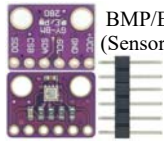
9V batteri-kontakt

Batterikontakt etter montering

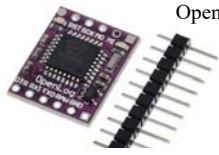
16. **Montering av stiftlister på E32 (radio) OpenLog (SD-kortterminal) og BMP/E280 (Sensorkort)**

Før vi monterer de tre kretsene må de utstyres med stiftlister slik at vi kan plugge dem inn i hylsekontaktene. Radioen E32 er vanligvis levert med stiftlist bak på kortet, men de tre stiftene foran er normalt ikke montert. BMP/E280 kan vanligvis greie seg med 4 stifter, men disse må da monteres som antydet på bildet under. OpenLog skal ha 6 stifter. Pass på at kortene vender riktig side opp.

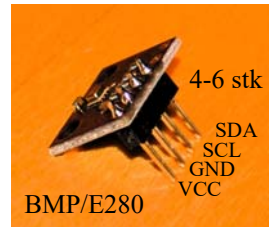
E32 (radio)

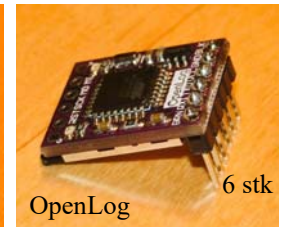


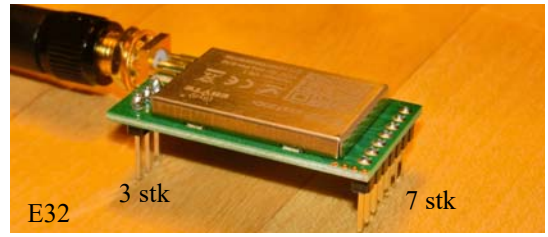
BMP/E280 (Sensorkort)



OpenLog

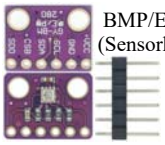


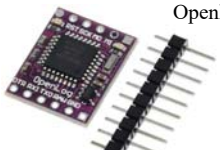


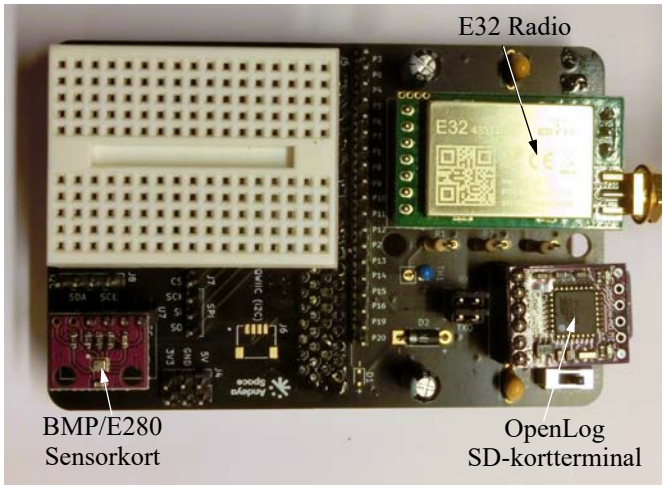


17. **Montering av E32** (radio) **OpenLog** (SD-kortterminal) **og BMP/E280 (Sensorkort)**
Vi kan godt vente med å montere disse kretsene til vi trenger dem i prosjektet. Her vil vi bare vise hvordan de skal monteres når vi kommer så langt.


E32
(radio)


BMP/E280
(Sensorkort)


OpenLog



18. **Visuell inspeksjon**
Før vi setter spenning på kretsen er det lurt å sjekke følgende:
    - Sjekk om noen loddinger mangler eller er dårlige, bruk gjerne et forstørrelsesglass
    - Sjekk om det er loddebroer
    - Sjekk om elektrolyttkondensatorene og Schottky-dioden står rett vei
    - Sjekk at spenningsregulatorene ikke er byttet om og ikke har loddebroer
    - Sjekk om batteriholderen er koblet rett vei
    - Sjekk at sensorkortet, OpenLog og radioen er koblet opp riktig
    - Mål med et Ohm-meter om + og – er kortsluttet eller har høy resistans

19. **Plassering av kortslutningsbøyle for oppkobling av OpenLog**
Ved hjelp av to kortslutningsbøyler kan vi koble til eller fra OpenLog. Ved innsamling av data trenger vi normalt kun å sende data fra Tx hos Micro:bit-en til RXD hos OpenLog. Dette gjør vi ved å plassere en kortslutningsbøyle på siden merket Rx.


Kortslutningsbøyler

Vi har nå montert kretsen og er klar for å begynne programmeringen.

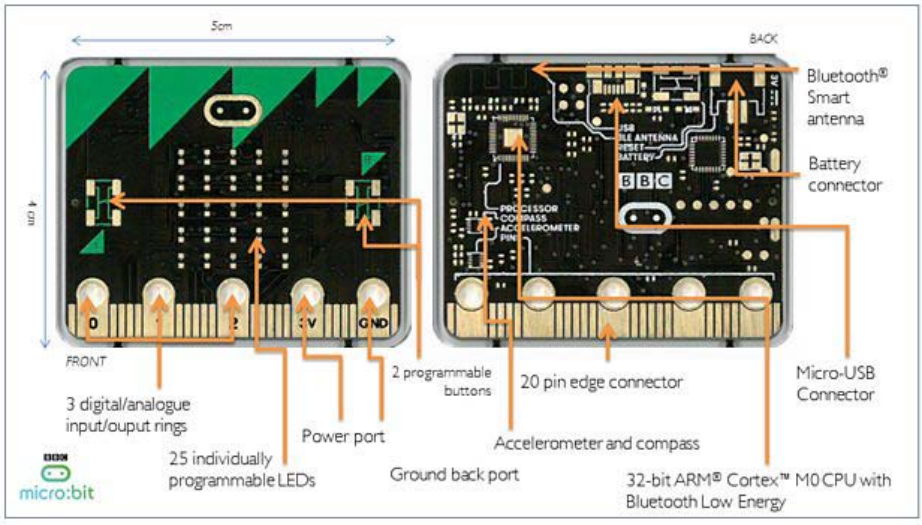
3 Grunnleggende om Micro:bit og makekode

I dette kapittelet skal vi gå gjennom noe grunnleggende fakta om micro;bit og bruk av programmeringsverktøyet MakeCode som er et nettbasert verktøy for programmering av mikro:bit. Den som er godt kjent med Micro:bit og programmering med blokkode i MakeCode, kan gå rett til avsnitt 4.6, side 47: “Oppdrag 6 – Lag et barometer”. Grunnkurset ender opp med et nulljustert barometrisk høydemåler (oppdrag 8), der vi bruker lufttrykket for å bestemme relative høydeforskjeller. Ved å nullpunktjustere instrumentet kan vi også bestemme den absolutte høyden over havet der vi befinner oss. Den barometriske høydemåleren inngår i CanSat-en.

Men la oss først se litt på mikrokontrollerkortet: Micro;bit.

3.1 Bakgrunn

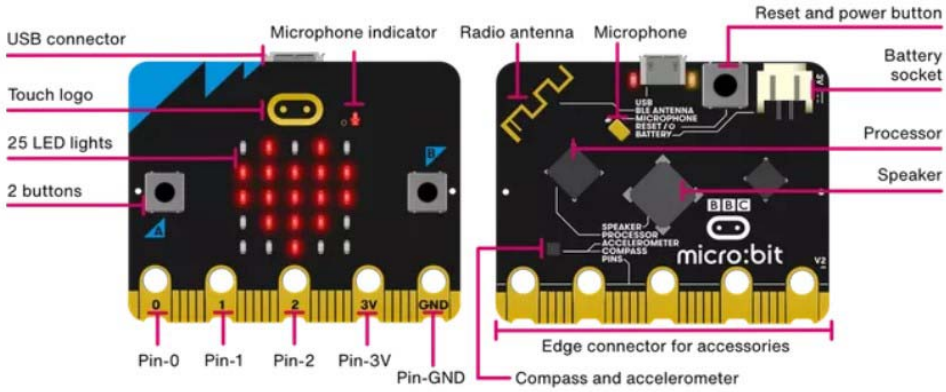
Micro:bit er blitt en svært populær krets for å komme raskt i gang med programmering og realisering av ideer. Årsaken til dette er flere, men de viktigste grunnene er at kretsen er utstyrt med sensorer og et enkelt diodedisplay, dessuten kan den kommunisere trådløst med andre Micro:bit-er og den kan programmeres med ulike programmeringsspråk, fra blokkprogrammering til Java og Python. Den kan også monteres i en sokkel (kantkontakt) slik at man kan få tilgang til mange av Micro:bits porter. Den er derfor lett å komme i gang med, samtidig som den gir store muligheter.



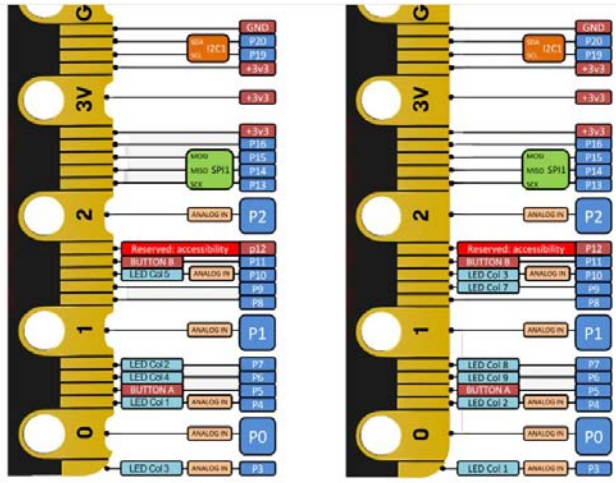
Kretsen har 3-akse akselerometer og magnetometer slik at den kan detektere helningen til Micro:bit-kortet og fungere som et enkelt kompass. 25 små lysdioder organsiert i en 5 x 5 matrise kan fungere som et primitivt display, man også kan bruke som lysdetektorer. Mikroprosessoren har også en temperatursensor. Denne registrerer imidlertid temperaturen i mikrokontrolleren og egner seg i liten grad til måling av temperaturen til omgivelsene.

Figuren over viser versjon 1 av Micro:bit. Det er også kommet en versjon 2^2 som har en del nyttige tilleggsfunksjoner.

I tillegg til å ha alt det som versjon 1 har av muligheter og sensorer, så har versjon 2 en liten mikrofon og en høyttaler. Disse gjør det mulig å måle lydnivå og spille melodier. En liten lysdiode viser at mikrofonen faktisk registrerer lyd. Den har dessuten en av/på knapp. Logoen på framsiden kan også brukes som touch-bryter (Touch logo).



Versjon 2 er også oppgradert på flere andre måter. Den har blant annet fått en kraftigere mikrokontroller fra Nordic Semiconductor (nRF52833), som har vesentlig mer lagerkapasitet (Flash og RAM)[3] enn versjon 1. Den kan også levere vesentlig mer strøm til utstyr som kobles til Micro:biten[4], og har fått oppgradert BlueTooth fra versjon 4.0 —> 5.0. Tilkoblingene er omtrent som tidligere.



Versjon 1 Versjon 2

2. https://kitronik.co.uk/blogs/resources/explore-micro-bit-v1-microbit-v2-differences
3. V1 - 256kB Flash 16kB RAM —> V2 - 512kB Flash, 128kB RAM
4. V1 - 90mA —> V2 - 200mA

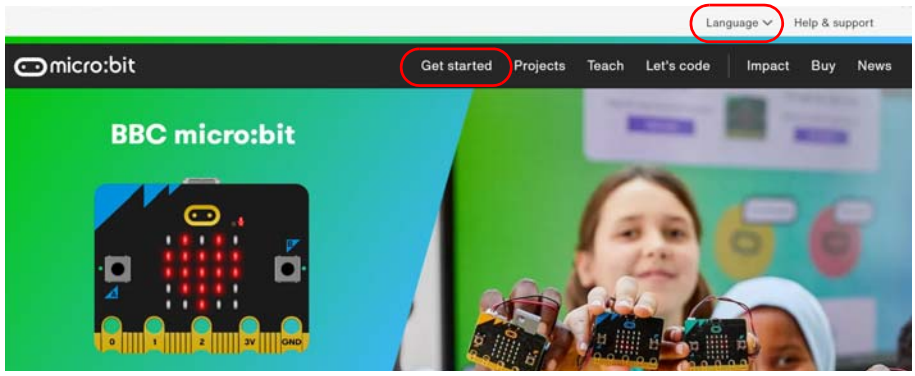
3.2 Programmeringsverktøyet, “MakeCode”

Micro:bit kan kodes med ulike språk bl.a. Python, Javascript eller *blokkode* som er basert på Java. Her skal vi konsentrere oss om å bygge opp programmet ved hjelp av *blokker*. Denne typen programmering kalles derfor *blokkoding*. En Micro:bit kan også programmeres med Skratch

Nettressursen

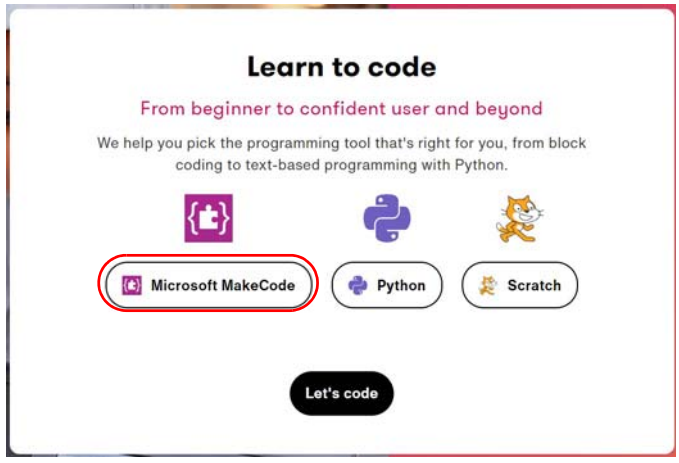
De fleste nettlesere kan brukes, men kan gi litt varierende prosedyrer for lagring og overføring av filer. For de som bruker Chromebook så er nettleseren gitt. Noen anbefaler bruk av Chrome, versjon 65 eller nyere, da fungerer webUSB som gjør overføring av filer fra datamaskin til Micro:bit enklere.

1. Gå til nettstedet: https://microbit.org/
2. Velg norsk som språk fra menylinjen, om det ikke alt er gjort.

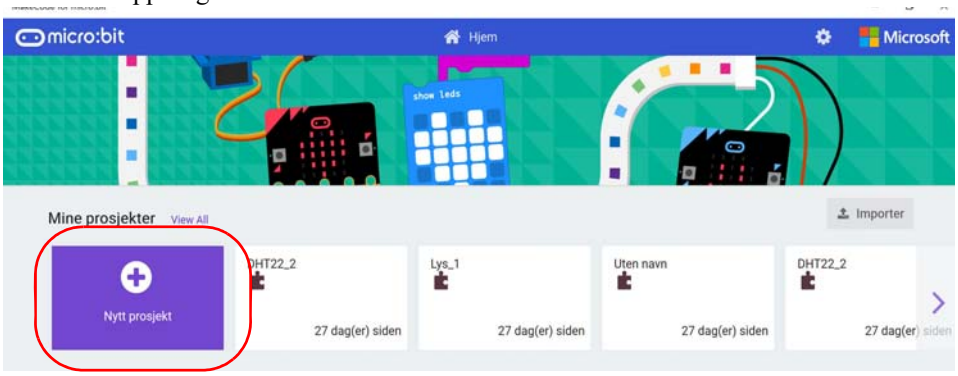


Start programmet

3. Velg *Start koding* fra menylinja eller gå ned til den delen av siden som er vist på bildet under, og velg programmeringsspråk. Vi velger å begynne med blokkoding dvs. *Microsoft MakeCode*:



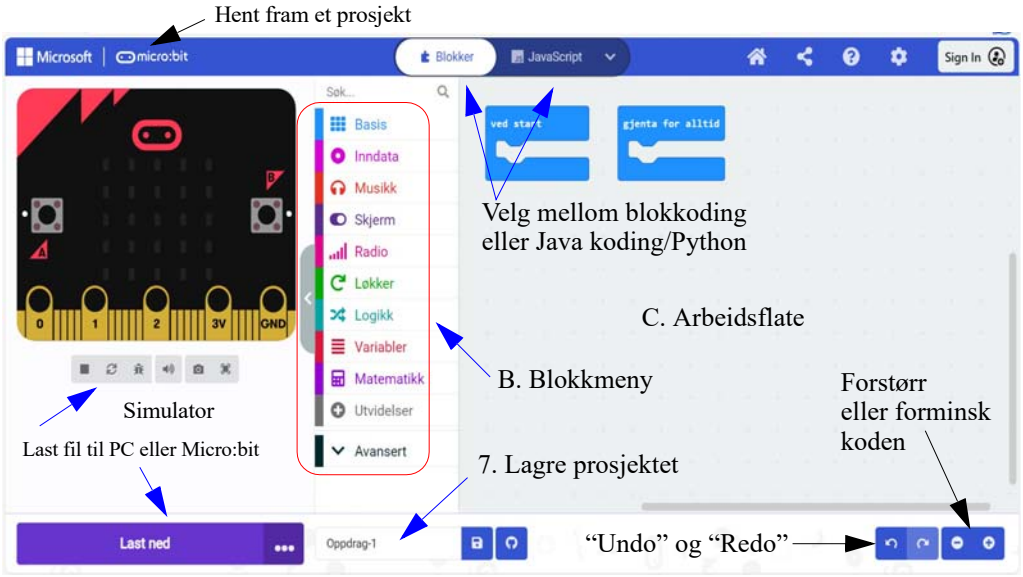
4. Du får da opp følgende vindu:



Velg *Nytt prosjekt*

5. Skriv inn navnet på ditt første prosjekt: *Oppdrag-1*. Vi ser for oss å løse mange oppdrag eller delprogrammer (Oppdrag-1,-2,-3 ...) som vi til sist kan sette sammen til et større program etter ønske.

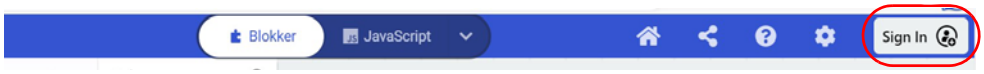
6. Du skal nå se et bilde omtrent som dette:



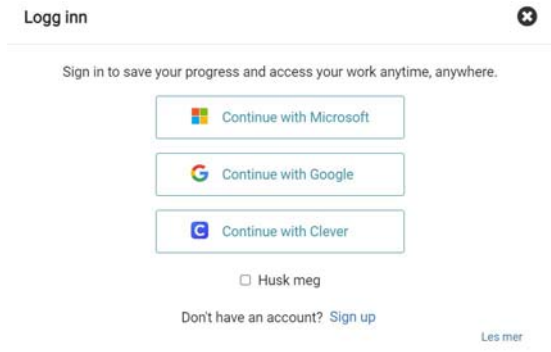
Her er en kort beskrivelse av gangen i arbeidet:

3.3 Lagre og overfør programfilen til Micro:bit-en

1. ***Opprett en konto***
   Du skrev inn prosjektnavnet når du gikk inn i programmet. For å kunne lagre programmene i skyen så er det lurt å opprette en konto. Velg *Sign in* øverst til høyre på menylinja.



   Da kommer følgende boks opp:



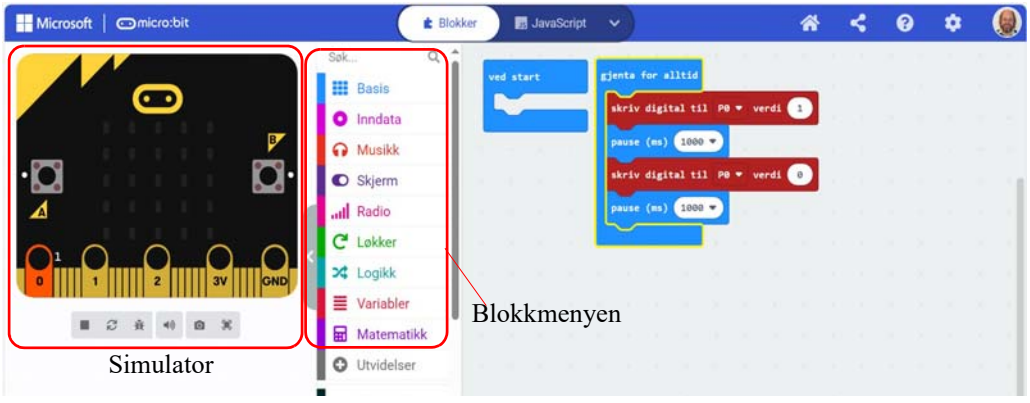
   Her kan du velge å bruke en konto hos Microsoft, Google eller Clever, eller opprette en konto hos MakeCode, i så fall velger du: *Don't have an account? Sign up.*

   Når man har opprettet en konto lagres programmene fortløpende i skyen.

2. ***Lagre prosjektet på egen maskin***
   Det kan også være lurt å lagre prosjektet på egen maskin. Du trykker da på disketten til høyre for prosjektnavnet og filen lastes ned til katalogen *Nedlastinger*. Flytt så filen over i en katalog slik at du lett finner den igjen senere.

   Før vi går videre må vi lage oss et lite program.

3. ***Mitt første program***
   Ved å velge blant fanene i blokkmenyen får vi opp lister over blokkommandoer. Disse dras fram av menyen og settes sammen på arbeidsflata slik at blokkene tilsammen blir det programmet vi ønsker oss, som i eksempelet vist på figuren under. Vi skal ganske snart se nærmere på hvordan programmet er bygget opp.

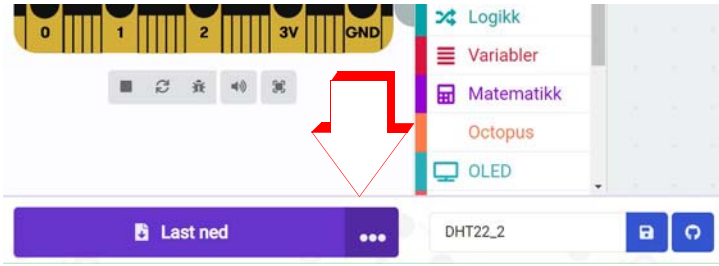


Legg spesielt merke til *simulatoren* til venstre i bildet. Denne vil vise hvordan programmet påvirker Mikro:bit-en eller hvordan bruk av knapper eller bevegelse av Micro:bit-en påvirker programmet. Det kan være praktisk å bruke simulatoren for å teste ut programmer før vi overfører det til den fysiske Micro:bit-en. Den oransje fargen lengst til venstre på figuren over, indikerer at denne kontakten har spenning (ca. 3V). Den skal kanskje slå på en lysdiode.

4. ***Lagring av programfilen***
   Filen lagres automatisk i skyen. Dette indikeres med haken i den vesle skyen til høyre for disketten. Ønsker du å lagre filen på egen maskin så må du laste den ned og flytte den til ønsket katalog.

5. ***Koble Micro:biten fysisk til PC-en***
   Det er på tide å koble til Micro:bit til USB-inngangen med den medfølgende USB-kabelen. En gul lysdiode på Micro:bit-en skal lyse når den får strøm. For å se at programmet vi har laget virker, må vi koble en lysdiode til Micro:bit-en. Dette kan vi gjøre på ulike måter. Vi velger å koble opp dioden på koblingsbrettet til CanSat-en, se side 36.

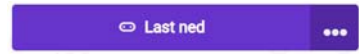
6. ***Koble Micro:biten til programverktøyet***
   Trykk på de tre prikkene til høyre for *Last ned* knappen



   Velger du *Connect device* får du opp flere informasjonsbokser, Klikk NESTE. I løpet av prosessen må du også velge den Micro:bit-en du ønsker å koble deg til (pair). Til slutt velger vi *Fullfør*.

   Dersom oppkoblingen er mislykket skyldes det sannsynligvis at micro:bit-ens "firmware" er for gammel (eldre enn versjon 0249).

7. ***Last ned programmet***
   Når vi nå trykker *Last ned* så overføres fila direkte til Micro:bit-en uten å lagres i den lokale PC-en. Den lagres imidlertid fortløpende i skyen.

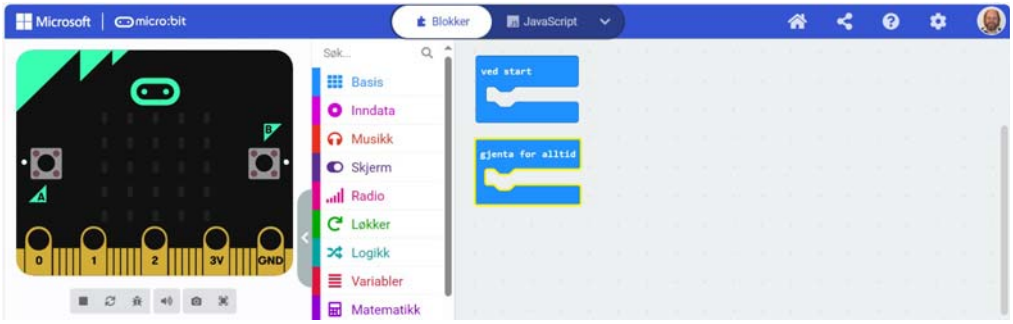


8. ***Manuell overføring av filen til Micro:bit***
   Du kan også bruke *Filutforskeren* i Windows til å flytte fila fra katalogen der du lagret den til micro:bit-en. Dette gjør du ved å slippe fila over symbolet av Micro:bit-en. Du vil se et blinkende gult lys på undersiden av Micro:bit'en idet filen overføres.

Programmet skal nå være overført.

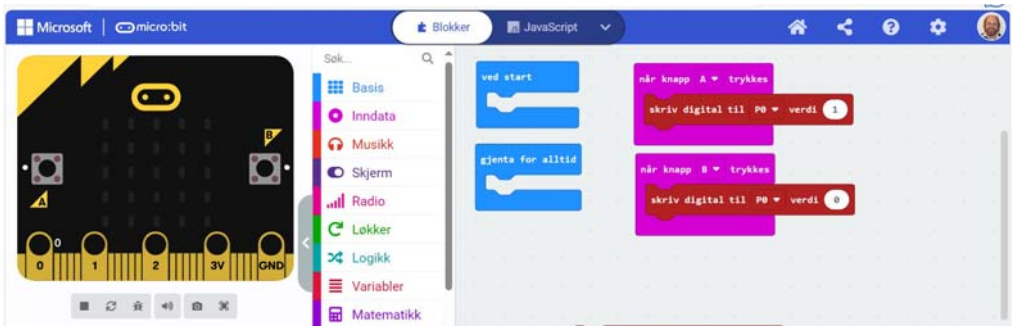
3.4 Programstruktur

Når vi åpner programmet så ligger det to blokker klar på arbeidsflata:



Det er blokkene "ved start" og "gjenta for alltid". Inne i "gapet" til disse to blokkene kan vi legge blokk-kommandoer. Det som legges i gapet til "ved start" gjøres bare ved oppstart av programmet. Det som legges i gapet til "gjenta for alltid" vil bli gjort om og om igjen til strømmen brytes eller batteriet er tomt. Vi sier at programmet går i en uendelig sløyfe.

Noen *hendinger* er imidlertid slik at programmet kan hoppe ut av sløyfa for å ta seg av hendingen. En slik hending er f.eks. at vi trykker på en av knappene til Micro:bit-en. Da vil vi gjerne at noe spesielt skal skje umiddelbart. Kanskje vil vi at en LED skal tennes når vi trykker på knapp A og at den samme LED'en skal slukkes når vi trykker på knapp B. I disse tilfellene så sier vi at sløyfa blir avbrutt, eller får et "interrupt", slik at den umiddelbart kan utføre det knappetrykket ber den om å gjøre. Med en gang jobben er gjort, hopper den tilbake i sløyfa og fortsetter der den slapp.

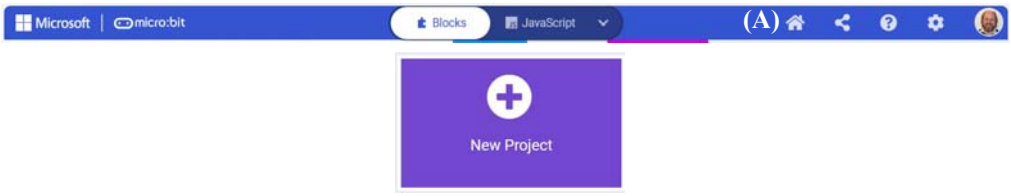


3.5 Praktisk kunnskap om redigering av filer

Det er noen triks det kan være greit å vite om når vi skal begynne å skrive programmer. Her er noen jeg har hatt nytte av:

Opprett nye programmer

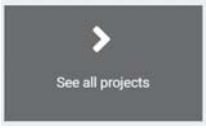
Dette er i og for seg enkelt ved at man forlater programmet man har arbeidet med ved å velge "Home"-knappen på menylinja. For så å velge "Nytt prosjekt" eller "New project".



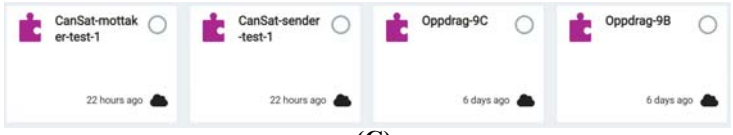
Det man må være klar over er at man da ikke får med seg installerte biblioteker, slik man f.eks. gjør i editoren til Arduino.

Ønsker vi å få med oss biblioteker og evt. bygge videre på tidligere arbeider, samtidig som vi tar vare på den gamle versjonen av programmet, så kan vi gjøre følgende:

Vi velger home (A) og får opp de siste programmene vi har redigert. Så går vi lengst til høyre i denne menyen og velger "See all projects" (B).

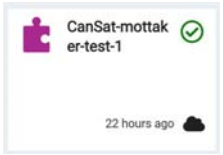


(B)



(C)

Så velger vi programmet vi ønsker å gå videre med, ved å klikke på ikonet slik at det kommer fram en grønn hake øverst i høyre hjørne (D). Vi klikker så på "Duplicate" på menylinja (E) og får opp en boks hvor vi kan skrive inn et nytt navn på kopien (F).



(D)

(F)

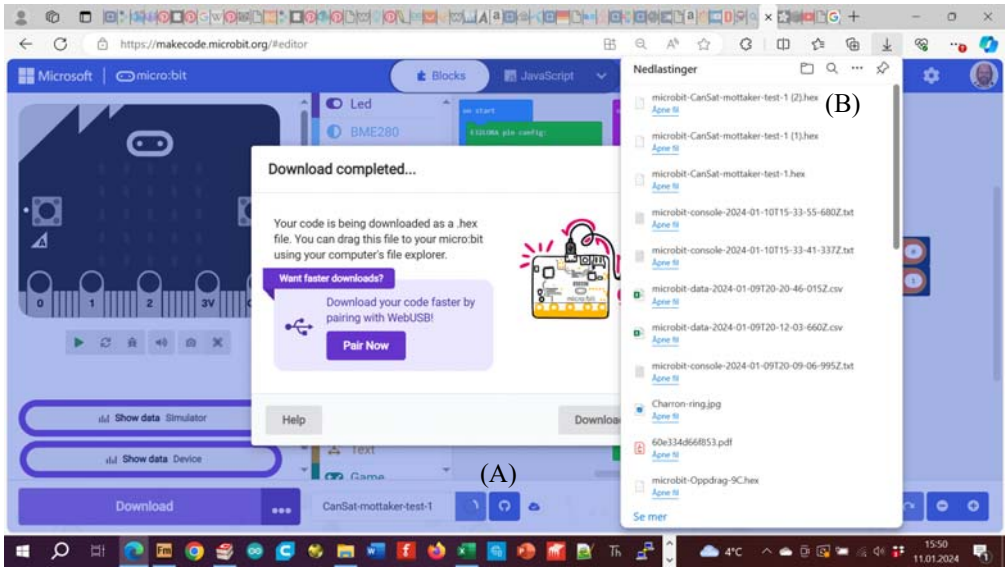
(E)

På denne måten lager vi en kopi av programmet med nytt navn og beholder bibliotekene fra den opprinnelige filen.

Lagring på egen PC

Det er alltid lurt å ha en backup av de viktigste programfilene på egen PC.

Dette får du til ved å klikke på disketten nederst midt på (A). Da overføres filen til katalogen "nedlastinger" på egen maskin (B). Derfra kan du så flytte filen videre over til ønsket katalog.



La oss så gå igang med oppdragene.

4 Oppdrag – Grunnkurs

I dette kapittelet skal vi bygge opp programmene steg for steg. Vi har kalt de enkelte delene for *Oppdrag*. La oss gi en kort oversikt over oppdragene før vi går i gang:

Oppdragene – en oversikt

Oppdrag 1
Koble opp en lysdiode på koblingsbrettet og skriv et program som får den til å blinke. 200 ms på og 200ms av.
Læringsutbytte: Bruk av porter, lagring og opplasting av programmer.

Oppdrag 2
Lag en teller som teller ned fra 9 – 1 og vis nedtelling på punktdisplayet. Tenn lysdioden når telleren har nådd til 0 og "stopp" programmet. Slukk lysdioden etter 5 sek.
Læringsutbytte: Anvendelse av for-løkke, if-setning, bruk av LED-display og bruk av variabler.

Oppdrag 3
Bruk Kitronik's OLED-display og skriv nedtellingen til displayet.
Læringsutbytte: Installere bibliotek og skrive til display.

Oppdrag 4
Lag et voltmeter og skriv spenningen til OLED-displayet.
Læringsutbytte: Les av analog inngang, forstå analog til digitalomforming (ADC) og omregning fra digital verdi til spenningsverdi.

Oppdrag 5
Lage et termometer med TMP36 og skriv avlest temperatur til displayet.
Læringsutbytte: Koble opp TMP36 på koblingsbrettet, omregning til °C og evt. kalibrering av termometeret.

Oppdrag 6
Monter BMP/E280 og les av temperatur, fuktighet og lufttrykk, og vis resultatet på OLED-displayet.
Læringsutbytte: Bruk av bibliotek. Bruk av I^2C-buss og adressering av komponenter langs bussen.

Oppdrag 7
Regne om fra barometrisk lufttrykk til relativ høyde.
Læringsutbytte: Bruk av matematikk og regneoperasjoner, forstå betydningen av parametere i omregningsformelen, matematisk modellering av et naturfenomen.

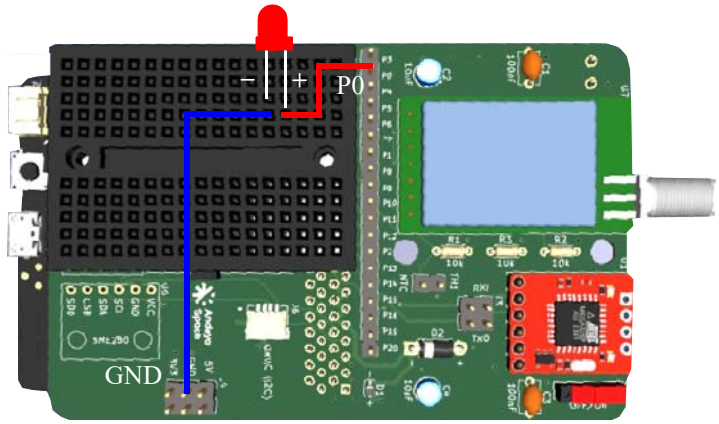
Oppdrag 8
Utføre nullpunktsjustering fra relativ til absolutt høyde over havet.
Læringsutbytte: Forstå betydningen av nullpunktjustering ved hjelp av en nærliggende referansestasjon.

4.1 Oppdrag 1 – Blinkende lysdiode

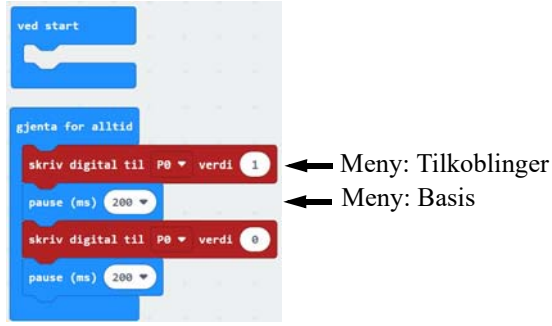
Oppdrag 1 – Blinkende lysdiode
Koble opp en lysdiode på koblingsbrettet og skriv et program som får den til å blinke. 200 ms på og 200ms av.
Læringsutbytte: Bruk av porter, lagring og opplasting av programmer.

1. **Klargjøring av kretsen**
   Fjern radioen (E32), SD-terminalen (OpenLog) og sensorkortet (BMP/E280) om disse er montert. Sett On/Off-bryteren i Off-posisjon. Til dette oppdraget skal vi hente forsynings-spenningen fra USB-kabelen.

2. **Oppkobling**
   Monter lysdioden på koblingsbrettet og koble anoden (+) til port P0 på Micro:bit-en og katoden (–) til GND. Vi vil at programmet skal slå av og på spenningen til porten P0. Når vi slår på spenningen danner vi en sluttet krets fra P0 (+) til jord (–). Vi kan se på P0 som den positive polen på "batteriet" og jord som den negative polen på "batteriet". Med programmet kan vi slå av og på "batterispenningen".



3. **Programmering**
   Oppdraget går ut på at vi skal tenne og slukke en lysdiode (LED).
   Hent blokkene fra menyene og skriv programmet i figuren under for å slå av og på lysdioden.
   Legg merke til at *verdien* til porten P0 kan settes til 1 (På) eller 0 (Av).



4. **Test programmet**
   Last programmet ned til Micro:bit-en og se hva som skjer med lyssdioden.

5. **Feilfinning:**

   - Sjekk at lysdioden er plassert rett vei, langt bein +, dvs. til P0
   - Sjekk at oppkoblingen er riktig
   - Sjekk at programmet skriver til riktig port (P0) og at det slår av og på lyset (0 og 1)

4.2 Oppdrag 2 – Nedtelling

Oppdrag 2 – Nedtelling
Lag en teller som teller ned fra 9 – 1 og vis nedtelling på punktdisplayet. Tenn lysdioden når telleren har nådd til 0 og "stopp" programmet. Slukk lysdioden etter 5 sek.
Læringsutbytte: Anvendelse av gjenta-løkke, hvis-setning, bruk av LED-display og bruk av variabler, lær å "stoppe" programmet.

1. **Oppkobling**
   Til dette oppdraget trenger vi ikke gjøre noen ekstra oppkobling. Vi kan la lysdioden stå oppkoblet.

2. **Programmering**
   For å lage en teller må vi lage oss en sløyfe med en variabel som teller ned fra en toppverdi (9) og ned til en nederste verdi (0). Vi kan betrakte en variabel som en "skuff" for å holde orden på telleren. Hver gang vi har talt én ned, så reduserer vi verdien av variabelen med 1.

   Først må vi sette et navn på variabelen, f.eks. Teller. Dernest må vi gi telleren en startverdi, og tilslutt må vi bestemme oss for hva som skal skje når vi kommer til null. Dessuten skal vi skrive verdien av telleren til LED-displayet for hver nedtelling.

   Lag variabelen "Teller": Gjør A, B og C som vist på figuren under

A 

Meny: Variabler

B 

C 

   Gi variabelen "Teller" en startverdi. Dette er noe vi bare gjør ved oppstart av programmet. Hent kommandoblokka (D) og plasser den i programmet.

D 

Meny: Variabler

Nedtelling og visning i LED-display: Når vi er i Gjenta-løkken (E), skal vi for hver runde redusere tellerverdien med 1 (E), og telleren skal vises på displayet. Sett sammen E – H.

E 

← Meny: Løkker

F 

← Meny: Variabler

G 

← Meny: Basis

H 

← Meny: Variabler

3. **Test programmet**
Gjør ferdig programmet, last det ned til Micro:bit-en og følg med på nedtellingen.

Vi erfarer at lysdioden ikke tenner når vi er på null og programmet fortsetter å telle med negative verdier. La oss se hvordan vi kan løse denne utfordringen.

4. **Tenn lysdioden og stans programmet.**
Tenn lysdioden: Til dette formålet bruker vi en *hvis-setning,* også kalt *betingelses-setning* (I). Hvis-setningen krever en betingelse som forteller når lysdioden skal tennes. Denne er rammet inn med gult i figuren under (I). Betingelsen er bygge opp av en sammenligning mellom variabelen *Teller* og verdien 0 (J, K). Hvis-setningen settes inn i Gjenta-løkka.

I 

← Meny: Logikk (“Logic”)

J 

← Meny: Logikk (“Logic”)

K 

← Meny: Variabler

Når betingelsen er oppfylt skal vi tenne lysdioden og slukke den etter 5 sekunder (L).

L 

← Fra Oppdrag 1

Stopp programmet: Å stoppe “Gjenta for alltid” lar seg egentlig ikke gjøre (M). Men vi kan legge inn en løkke der betingelsen alltid er oppfylt. Da vil den gå i denne løkka til “evig tid”, helt til vi slår av spenningen eller resetter programmet.

M 

← Meny: Løkker (“Loops”)

5. **Test programmet**
Gjør ferdig programmet, last det ned til Micro:bit-en og følg med på nedtellingen.

Fungerte det slik du tenkte?

6. **Feilfinning** (er selvfølgelig avhengig av feilen som oppstår):

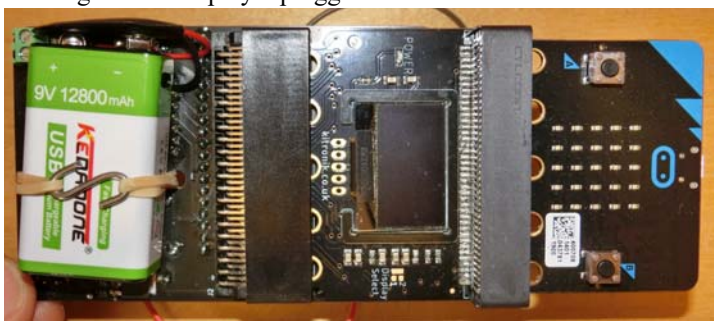
- Tenner ikke lysdioden når telleren kommer til null. Sjekk betingelsen i hvis-setningen.

4.3 Oppdrag 3 – Skriv til OLED-display

Oppdrag 3 – Skriv til OLED-display
Bruk Kitronik's OLED-display og skriv nedtellingen til displayet.
Læringsutbytte: Installere bibliotek og skrive til display.

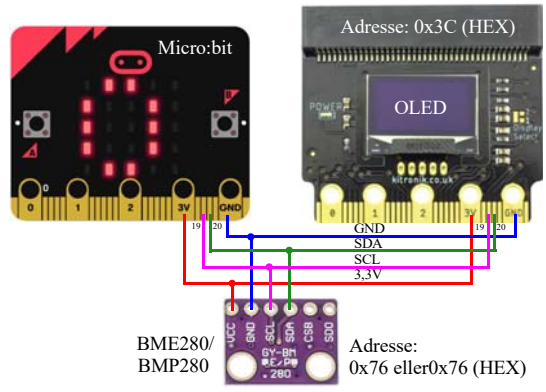
1. **Oppkobling:**
OLED-displayet: LED-displayet på Micro:bit har utvilsomt sine begrensninger. Vanligvis er det ikke behov for å bruke LED-displayet under normal bruke av en CanSat, men det kan være fint å ha under uttesting og alternativ bruk av CanSat-en.

   Oppkoblingen av OLED-displayet gjøres ved å plugge det inn mellom Micro:bit-en og kant-kontakten, for så å fjernes det når CanSat-en evt. skal sendes opp. Det viktigste er å passe på at Micro:bit-en og OLED-displayet plugges inn rett vei.



   Micro:bit-en og OLED-displayet kommuniserer via I^{2}C-bussen, så la oss se hvordan denne fungerer.

2. **I^{2}C-bussen:**
I^{2}C (IIC) står for Inter IC-bus, og var ment å være akkurat det da den ble utviklet av Philips Semiconductor tidlig på 80-tallet. Bussen er svært enkel med sine to linjer (klokke (SCL) og datalinje (SDA)) i tillegg til spenning og jord. Videre har hver krets langs bussen en unik adresse som enten er satt av fabrikken eller kan settes med strap'er på brikken. Bussen er dessuten utstyrt med kollisjonsdeteksjon. Dersom hver enhet (IC) har forskjellig adresse, kan man adressere informasjonen slik at den kommer til riktig enhet. I starten var den definert med en hastighet på 100 kbit/s. Senere, etter som en trengte raskere dataoverføring, er "Fast mode" – 400 kbit/s og "High speed" – 3,4 Mbit/s definert.

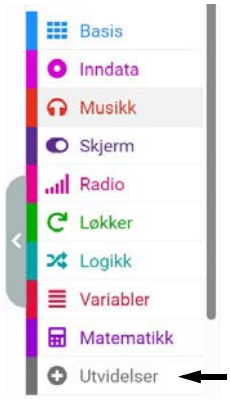


   På figuren over til høyre ser vi at to enheter er koblet på de samme to kommunikasjonslinjene (SDA (data), SCL (klokke)). Dette er displayet vårt og en trykksensor (BME/P280). I tillegg er kretsene forbundet til spenningskildene 3,3V og jord (GND). Det er også vanlig å koble

en motstand fra SDA- og SCL-linjene og til 3,3V, men det går også ofte bra uten når linjene er korte. I vårt tilfelle er linjene koblet til 3,3V på OLED-displayet.

Oppkoblingen mellom Micro:bit og OLED-displayet vil skje automatisk når de plugges sammen. I prinsippet kan mange slike enheter med ulik adresse kobles på bussen.

3. **Installasjon av bibliotek**
For på en enkel måte å kommunisere med OLED-displayet, så laster vi ned og installerer et bibliotek. Det gjør vi slik (A til F):



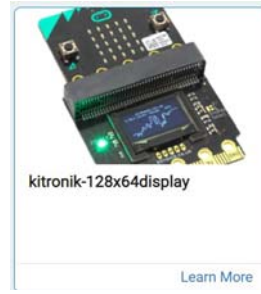
A: Velg - Utvidelser

C: Skriv - Kitronik display **D**: Velg - Søk

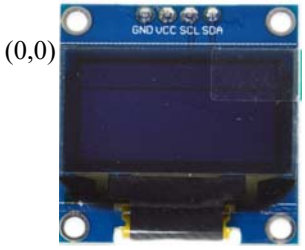
B: Velg - Lights and Display

E: Finn biblioteket i lista

F: Klikk på ikonet av *kitronik - 128x64 display*



4. **Kort beskrivelse av OLED-displayet**
OLED-displayet er egentlig et grafisk display som består av 128 x 64 punkter som kan lyse eller ikke og på den måten danne grafikk eller tekst. Figuren under viser hvordan punktene er organisert og kan adresseres.



Koordinater for plassering av markør

Ved hjelp av enkle blokk-kommandoer kan vi skrive til displayet. Her er noen viktig blokker:

Slå av og på displayet

Tøm displayet

Sett font-størrelse

Slett linje f.eks. linje 1

Skriv verdi av variabel Teller i starten på linje 1

5. **Programmering**
Vi skal nå legge inn programkoden og tar utgangspunkt i koden fra Oppdrag 3.

Forberedelser
Først fjerner vi blokken som skrev til LED-displayet

G

Initialisering av displayet
Vi setter opp displayet i "ved start" ved hjelp av blokkene:

H

← **H**: Meny: 128 x 64 Display

Her setter vi fontstørrelsen til "Big" og slår deretter på displayet ved å velge "PÅ" (utelater vi denne blokken så vil displayet være påslått, så den er kanskje unødvendig).

Skriv til displayet
Vi ønsker å skrive verdien til variabelen "Teller" til displayet. Siden vi bare ønsker å skrive ett tall som teller ned fra 9 – 0, lengst til venstre på linje 1, så må vi å tømme displayet før hver utskrift. Gjør vi ikke det så skriver den over det forrige tallet slik at utskriften blir ugjenkjennelig.

I

← **I**: Tøm og skriv til displayet

Legg inn en forsinkelse
Uten at vi gjør noe mer, vil tallene rase fra 9 – 0 før du vet ordet av det. Vi ønsker derfor å legge inn en forsinkelse på 1 sek, mellom hver utskrift. Legg inn forsinkelsen på et fornuftig sted i programmet.

6. **Test programmet**
Gjør ferdig programmet, last det ned til Micro:bit-en og følg med på nedtellingen.

Fungerte det slik du tenkte?

7. Feilfinning (er selvfølgelig avhengig av feilen som oppstår):

- *Displayet fungerer ikke, men starter å fungere når trykksensoren BME/P280 fjernes.*

Årsaken til dette er at displayet og trykksensoren bruker begge I^2C-bussen for å kommunisere med Micro:bit-en. Mens displayet bruker spenningen fra USB-en, så trenger trykksensoren spenning fra batteriet. Dersom batteriet kobles til, og Av/På-bryteren slås på, så virker trykksensoren og displayet sammen.

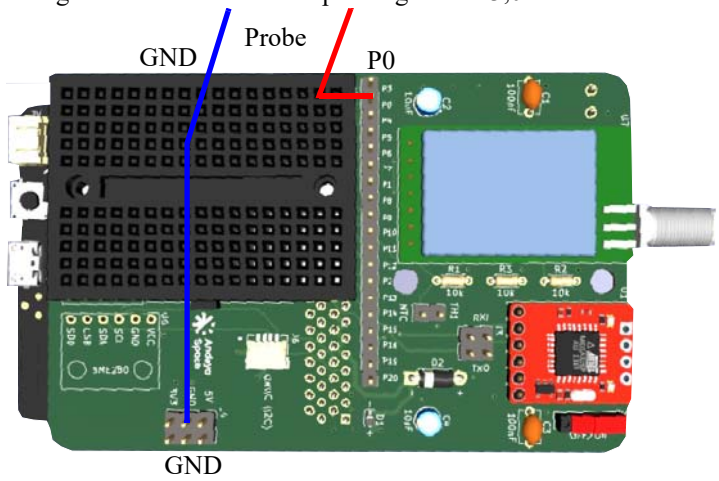
Husk å slå på displayet med kommandoen "Power on" i programmet ("ved start"), pkt. 4.

4.4 Oppdrag 4 – Lag et voltmeter

Oppdrag 4 – Lag et voltmeter
Lag et voltmeter og skriv spenningen til OLED-displayet.
Læringsutbytte: Les av analog inngang, forstå analog til digitalomforming (ADC) og regne om fra digital verdi til spenningsverdi.

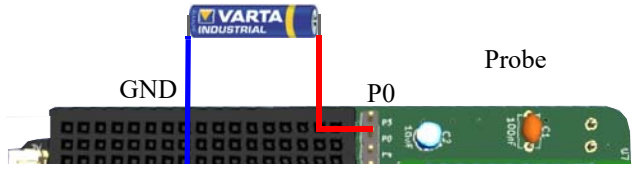
1. **Klargjøring av kretsen**
   Fjern lysdioden med ledninger om den ikke alt er fjernet, OLED-displayet skal beholdes.

2. **Oppkobling**
   Til dette oppdraget kobler vi en ledning til GND (jord) og en til P0. Dette er *proben* vår for å måle spenning. PASS PÅ – Ikke mål spenninger over 3,0V.



3. **Måling av spenning – Analog til digital konverteren (ADC)**
   Når vi skal måle spenning med en mikrokontroller må vi bruke en analog inngang som "leser" av spenningen og omformer den til et digitalt tall. De fleste av portene på Micro:bit kan også brukes som analoge innganger. Dvs. at portene registrerer nivåer mellom "0" og "1" (dvs. fra 0 til 3,0V), og angir disse med verdier fra 1 – 1023.

   Avlesning av spenning
   Når vi skal måle spenning, måler vi alltid spenning i forhold til noe, her i forhold til jord (GND). Dersom du leser av spenningen på porten og skriver tallet ut på displayet, så vil du se et helt annet tall enn 1,5V, som du kanskje hadde håpet.

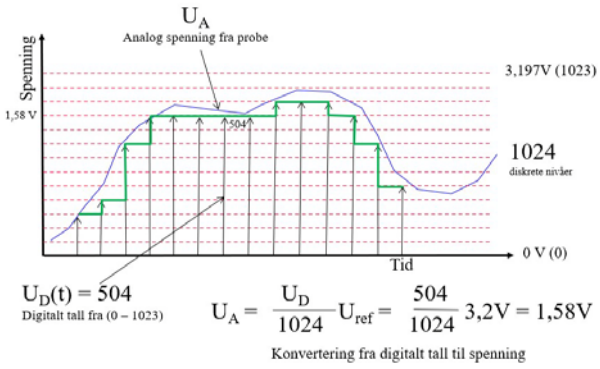


   Analog til digital omforming
   Årsaken til det overraskende resultatet, som kanskje viste seg å være på ca. 500, er at den målte batterispenningen gjøres om fra en spenning til et digitalt tall. Verdien til dette tallet er

bestemt av en intern spenningsreferanse (U_{ref}). Spenningsreferansen er vanligvis lik forsyningsspenningen, som er ca. 3,2V når vi bruker USB-kabelen som spenningskilde, og noe lavere når vi bruker batteri (ca. 3,0V). Dette er illustrert i figuren over til høyre.

Siden AD-konverteren opererer med 10 bit vil en spenning på inngangen lik referansespenning bli lik det digitale tallet $2^{10} - 1 = 1023$. For å få den målte spenningen (U_A) vist i Volt, må vi derfor regne oss tilbake til spenning med følgende formel, der U_D er den digitale verdien som kommer fra AD-konverteren:

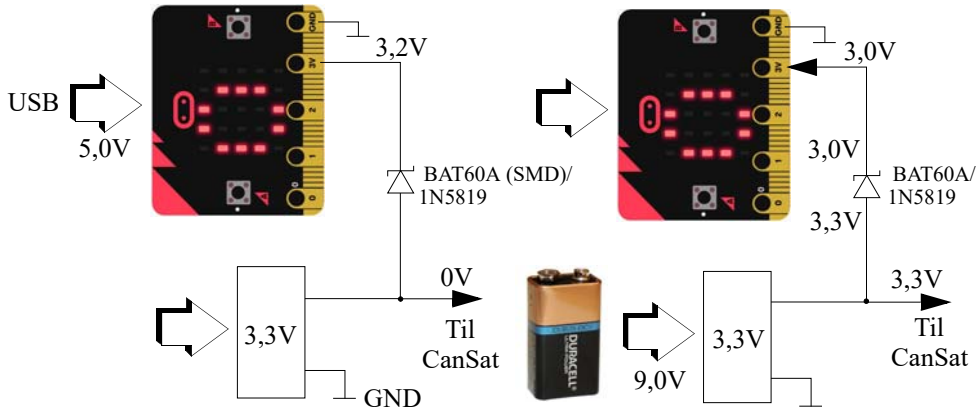
$$U_A = U_D/1024 * U_{ref} \quad (4.1)$$



Når vi måler vårt AA batteri til $U_D = 504$ så får $U_A = 1{,}58V$ i følge ligning (4.1).

Referansespenning – ved bruk av batteri og USB

Referansespenningen er ganske stabil så lenge vi bruker USB-kabelen som spenningskilde. Bruker vi derimot et batteri for å drive Micro:bit-en, kan denne variere etter som batterispenningen endrer seg. I vårt tilfelle vil spenningen til Micro:bit-en senkes en del pga. zenerdioden som er koblet mellom 3,3V regulatoren og Micro:bit-en. Spenningen vil derfor være stabil, men noe lavere enn når vi bruker USB-kabelen[5]. Spenningsfallet over zenerdioden vil også variere noe med strømtrekket. Resultatet av dette vil være at referansespenningen U_{ref}, vil variere avhengig om vi bruker batteriet eller USB-kabel som spenningskilde, og dermed også vår spenningsmåling.

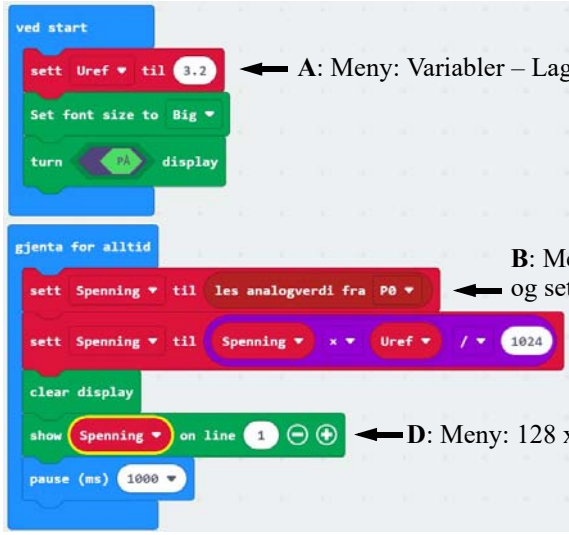


5. BAT60A har et typisk spenningsfall på 0,2V ved 100mA, men er overflatemontert og kan være vanskelig å montere. 1N5819 har et typisk spenningsfall på 0,28V ved 100mA og har aksielle ledninger som er langt enklere å montere.

NB! – *Vi merker oss at resten av CanSat-en er spenningsløs så lenge vi kun bruker USB-kabelen som spenningskilde og ikke har tilkoblet batteriet. Videre at når vi kun bruker batteri så har micro:bit-en kun 3,0 V spenningsforsyning, dette gjelder også OLED-displayet.*

4. **Programmering**
Vi skal lage et program som leser av spenningen f.eks. over et 1,5V batteri og skriver resultatet til OLED-displayet.

Vi velger å vise hele koden for voltmeteret vårt:



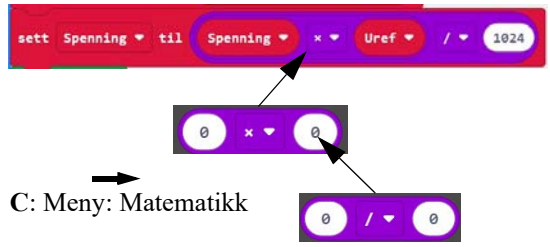
A: Meny: Variabler – Lag variabel U_{ref} og gi den verdien 3.2

B: Meny: Tilkobling – Les av analog port P0 og sett verdien inn i variabelen "Spenning"

C: Menyene: Variabler og Matematikk

D: Meny: 128 x 64 Display

Utfordringen vil være å deklarere to variable U_{ref} og *Spenning*, for så å tilordne verdien 3.2V til U_{ref} (A). Deretter leser vi av spenningen på port P0 med blokken "les analogverdi fra P0", denne finnes dels i menyen "Tilkobling" og dels i menyen "Variabler" (B). Så må vi regne om fra digital verdi til spenning i henhold til ligning (4.1), side 43. Her må vi benytte blokker fra menyen "Matematikk" (C). Legg merke til at regnestykket er satt sammen av to regneblokker av typen vist på figuren over til høyre. Til slutt så presenteres resultatet på OLED-displayet (D).



C: Meny: Matematikk

5. **Test programmet**
Gjør ferdig programmet, last det ned til Micro:bit-en. Finn et 1,5V batteri og bruk proben til å måle batterispenningen. Undersøk hvordan spenningen endres når spenningskilden er USB-kabelen og når den er 9V-batteriet.

6. **Feilfinning** (er selvfølgelig avhengig av feilen som oppstår):

- *Displayet virker ikke når CanSat-en kun bruker batteri*

Dette kan skyldes at displayet får for lite spenning. I så fall er løsningen å bruke både batteri og spenningstilførsel vi USB.

4.5 Oppdrag 5 – Lag et termometer

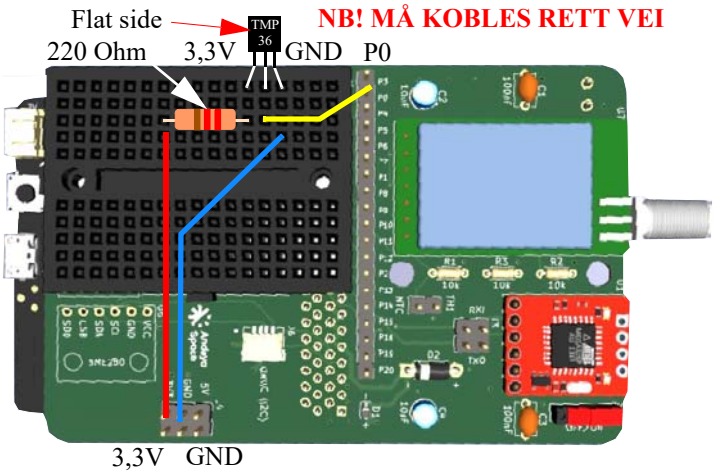
Oppdrag 5 – Lag et termometer

Lage et termometer med TMP36 og skriv avlest temperatur til displayet.

Læringsutbytte: Koble opp TMP36 på koblingsbrettet, omregning og evt. kalibrering av termometeret.

1. **Klargjøring av kretsen**
   Fjern ledningene (probene) fra Volt-meteret og lagre programmet for Oppdrag 4. Vi skal nå koble opp temperatursensoren TMP36.

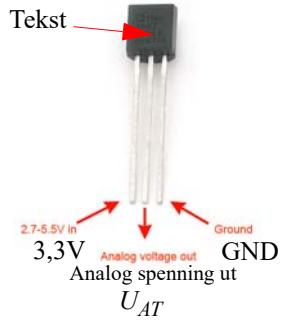
2. **Oppkobling**
   Til dette oppdraget kobler vi opp temperatursensoren TMP36 som vist på figuren under. Vi kobler en motstand på 220 Ohm i serie med spenningskilden



3. Omregning fra spenning til temperatur

TMP36 er en transistorlignende komponent som er en temperatursensor (bruk evt. et forstørrelsesglass for å lese teksten på komponenten). Denne skal kobles til 3,3V og 0V (GND). På senterpinnen kan vi ta ut en spenning som er proporsjonal med temperaturen i rommet i henhold til følgende sammenhenger:

- Spenningen er 0,5V ved 0°C
- Spenningen endrer seg med 10mV pr. K (eller °C)
- Spenningen øker med økende temperatur



Sammenhengen mellom temperatur og målt spenning er:

$$Temp = (U_{AT} - 0.5\ \text{V}) / 0{,}01\text{V/K} \quad (4.2)$$

Vi må også huske på at vi leser av den digitale verdien av spenningen slik at vi må regne om fra digital til analog spenning

$$U_{AT} = (U_{ref}/1024)U_{DT} \quad (4.3)$$

U_{DT} er tallet programmet leser av den analoge inngangen og som settes inn i ligning (4.3), som videre settes inn i ligning (4.2), som beregner temperaturen i °C. U_{ref} er referanse-spenningen som vil være noe forskjellig avhengig om energikilden er PC-en via USB-kabelen (U_{ref} = 3,2V) eller batteriet (U_{ref} = 3,0V) som omtalt foran.

4. **Programmering**

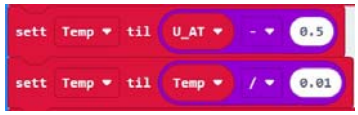
Det første vi må gjøre er å lese av spenningen (*U_DT*) på temperatursensoren som er koblet til P0.

← Menyene: Variabler og Tilkobling

Dernest beregner vi den analoge spenningen (*U_AT*) ved bruk av ligning (4.3).

← Menyene: Variabler og Matematikk

Så utfører vi omregningen fra spenning til temperatur i henhold til ligning (4.2). Legg merke til at vi gjør denne omregningen i to trinn og bruker variabelen *Temp* som mellomlagring av verdien.



← Menyene: Variabler og Matematikk

Så kan vi skrive ut temperaturen på displayet.

Redusere antall desimaler

Vi erfarer at variabler skrives ut med mange desimaler. Det er f.eks. lite hensiktsmessig å skrive ut temperaturen med mer enn en desimal. For å redusere antall desimaler bruker vi et triks:

← Menyene: Variabler og Matematikk

Vi multipliserer verdien av temperaturen med 10, for så å avkorte verdien, dvs. fjerne alle desimaler etter komma, for så å dividere med 10 før vi skriver ut verdien.

Skrive til display med tekst
For å kunne skrive til displayet med innledende og avsluttende tekst, må vi sette sammen en tekststreng av tekst og variabelen *Temp*:

Menyene: Display128x64 og Tekst

Legg merke til at vi konverterer variabelen *Temp* til tekst. Vi kan legge til flere variabler og tekstelementer dersom vi velger + lengst til høyre på tekstblokken.

5. **Test programmet**
Gjør ferdig programmet, last det ned til Micro:bit-en. Vi må koble til batteriet og slå på batterispenningen for å få spenning på uttaket for 3,3V (og 5,0V).

6. Feilfinning (er selvfølgelig avhengig av feilen som oppstår):

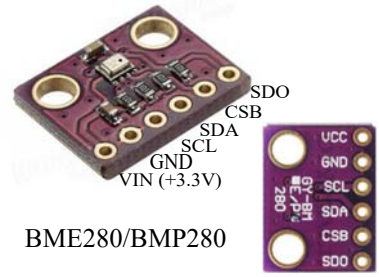
- *TMP36 blir svært varm*

  Slå av strømmen så fort som mulig. Sjekk om TMP36 er koblet riktig vei.

4.6 Oppdrag 6 – Lag et barometer

Oppdrag 6 – Lag et barometer
Monter BMP/E280 og les av temperatur, fuktighet og lufttrykk, og vis resultatet på OLED-displayet.
Læringsutbytte: Bruk av bibliotek. Bruke av I^2C-buss og adressering av komponenter langs bussen.

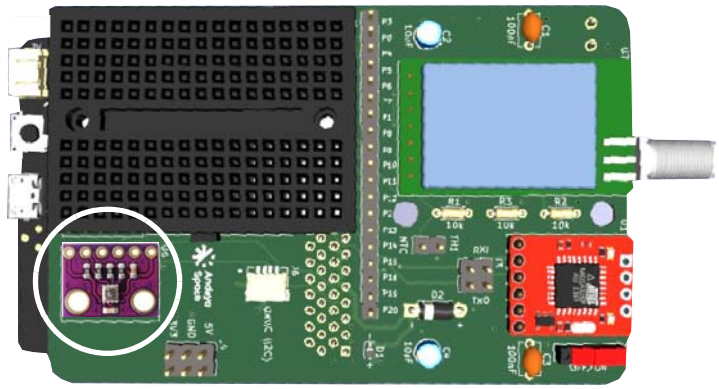
Vi skal nå koble opp BME280 (lufttrykk, temperatur og luftfuktighet) eller BMP280 (lufttrykk og temperatur). Pinningen for de to kretsene er like, men bibliotekene vi bruker kan være forskjellige.



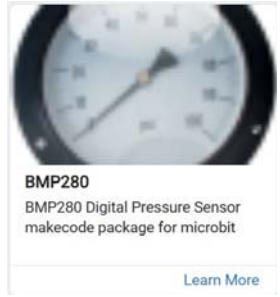
BME280/BMP280

1. **Klargjøring av kretsen**
Fjern ledninger og temperatursensoren fra koblingsbrettet. Vi skal nå koble opp BMP280/BME280 i sokkelen på CanSat-kretskortet.

2. **Oppkobling**
Til dette oppdraget kobler vi opp BMP280 eller BME280 som vist på figuren under.



3. **Installasjon av biblioteker**
Vi bruker det anbefalte biblioteket for *BMP280*. Dette installerer vi ved å velge *Utvidelser* fra menyen. Vi skriver BMP280 som søkeord og velger ikonet for BMP280, som vist på figuren til høyre.



Dersom det anbefalte Micro:bit biblioteket for BME280 ikke fungerer, kan man bruke følgende bibliotek[6]:

https://github.com/ElectronicCats/pxt-bme280

Vi velger *Utvidelser* som vanlig, men istedet for å søke i de anbefalte bibliotekene, skriver vi lenken til biblioteket inn i søkefeltet og trykker return.

I vårt eksempel har vi valgt å bruke BME280 og biblioteket https://github.com/ElectronicCats/pxt-bme280

4. **Programmering**
La oss se hvordan programmet kan bygges opp:

Initialisering
Vi bygger på tidligere programmer og inkluderer i seksjonen "ved start", blokker som gjelder initialisering av displayet og BME280 som vist på figuren under. Dette inkluderer også I^2C-adressen til BME280 som er 0x76 (heksadesimalt 76).

← Menyen: BME280 – Husk å skru på sensoren

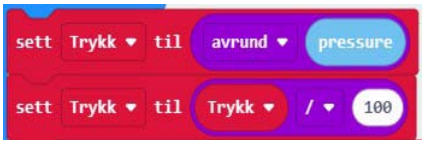
← Menyen: BME280

6. Oppdaget av Simen Bergvik ved Andøya Space Education

Les av sensorene for lufttrykk (pressure), temperatur (temperature) og luftfuktighet (humidity) som vist i figuren under.

← Menyen: BME280

Vi har dessuten valgt å forkorte verdiene slik at vi kan ha kontroll på antall desimaler. Vi legger merke til at lufttrykk leveres i Pa. og vi velger å vise det i mBar. Merk at mBar = hPa. Derfor har vi gjort det på følgende måte:



← Menyene: Variabler, Matematikk og BME280

Gjør tilsvarende med luftfuktighet og temperatur og velg å ha én desimal i visningen.

Midling av lufttrykk
Vi velger å lage en middelverdi av lufttrykket og å fjerne støy. Begynn med å ta middelverdien av 100 målinger og se om målingen blir mer stabil. Husk å nullstill variabelen *Trykk* for hver midling (måling).

← Menyene: Variabler, Løkker og BME280

5. **Test programmet**
Gjør ferdig programmet, last det ned til Micro:bit-en. Undersøk hvor mye lengre tid programmet bruker for å beregne trykket når verdien skal midles over 100 målinger. Er det en akseptabel forsinkelse?

6. **Feilfinning** (er selvfølgelig avhengig av feilen som oppstår):

- *Trykksensoren gir ikke verdier*

  Sjekk om den er slått på i programmet.

  Sjekk om bryteren er slått På og batteriet tilkoblet.

- *Trykksensoren runder av til hPa*

  Pass på at målingen gjøres i Pa og ikke hPa. Enkelte biblioteker gir mulighet til å velge.

4.7 Oppdrag 7 – Lag en barometrisk høydemåler[7]

Oppdrag 7 – Lag en barometrisk høydemåler
Regne om fra barometrisk lufttrykk til relativ høyde.
Læringsutbytte: Bruk av matematikk og regneoperasjoner, forstå betydningen av parametere i omregningsformelen.

Før vi går igang med å beregne høyden må vi ha litt bakgrunnskunnskap om sammenhengen mellom høyden over bakken og lufttrykket.

1. **Sammenheng mellom lufttrykk og høyde over hav**et

   Sammenhengen mellom lufttrykk og høyde over havet kan beskrives ganske godt matematisk. Skal vi måle absolutt høyde må vi dessuten passe på å nullpunktjustere høydemåleren siden den er sterkt avhengig av det generelle lufttrykket. I nedre del av atmosfæren er det ikke uvanlig å bruke ligning (4.4), som gir sammenhengen mellom lufttrykk og relativ høyde. Gitt en nærliggende referansehøyde (h_1), et referansetrykk (p_1) og en referanse-temperatur (T_1), vil ligningen gi absolutt høyde over havet:

$$h = \frac{T_1}{a}\left(\left(\frac{p}{p_1}\right)^{-\frac{aR}{g_0}} - 1\right) + h_1 \tag{4.4}$$

   Hvor:

h	Beregnet høyde i meter
h_1	Referansehøyde i meter
T	Temperatur i Kelvin
T_1	Referansetemperatur ved referansehøyden h_1
R	Den spesifikke gasskonstant 287,06 J/kg K
a	Temperaturgradient, foreslått verdi -0,0065 K/m
p	Målt trykk i Pa
p_1	Trykk i Pa ved referansehøyden
g_0	Tyngdeakselerasjonen 9,81 m/s^2

   Referansehøyde (h_1), trykk (p_1) og temperatur (T_1) er kalibrerte verdier på et sted i nærheten av der vi skal bruke høydemåleren, og som vi vet er riktige. Dette kan f.eks. være en meteorologisk målestasjon. Siden vi i første omgang er interessert i relative høydemål så kan vi sette $h_1 = 0$, $p_1 = p$ og T_1 lik temperaturen utendørs der vi er. Husk av T_1 skal være i Kelvin og er temperaturen utendørs. Vi skal senere se hvordan vi kan bruke h_1, p_1 og T_1 for å nullpunktjustere høydemåleren.

2. **Oppkobling**
   Oppkoblingen er som i oppdrag 6.

3. **Programmering**
   Vi ønsker å bygge på programmet som vi laget i oppdrag 6.

7. Se også *Fra fysikkens verden* – 1/2024 [2]

Forberedelser
Til dette oppdraget trenger vi ikke måle luftfuktigheten siden den ikke inngår i den barometriske høydemåleren. Så den målingen kan vi ta ut av programmet om vi vil forenkle programmet.

Beregning med eksponent
Vi legger merke til at omregningen fra trykk til høyde krever at vi regner med en eksponent. Dette er en regneoperasjon som MakeCode tilbyr (**). Vi vet også at eksponenten er et desimaltall og vi kan slå fast at skal beregningen bli nøyaktig nok, så må vi ha med flere desimaler.

$$h = \frac{T_1}{a}\left(\left(\frac{p}{p_1}\right)^{-\frac{aR}{g_0}} - 1\right) + h_1$$

Eksponent

Det viser seg imidlertid at eksponent-blokken i MakeCode ikke takler variabler med desimaltall, men kun heltall. Dermed kan vi ikke bruke denne blokken.

Vi skal etter hvert se hvordan vi kan omgå dette problemet.

Variabler og konstanter
Vi legger også merke til at formelen har mange variabler (p, p_1, h, h_1, T, T_1) og konstanter (a, R, g_0). Disse definerer vi i starten av programmet der vi legger inn konstantene og verdier for noen av variablene.

Oppsett av displayet (kjent fra før)

Oppsett av BME280 (kjent fra før)

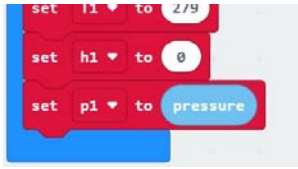
Antallet målinger som inngår i midlingen av trykkmålingene

Variabler og konstanter vi trenger

- Referansestasjonens temperatur i Kelvin
- Referansestasjonens høyde over havet
- Referansestasjonens lufttrykk når vi skal måle

I tillegg definerer vi variabelen *exp* som vi skal bruke til å ta vare på eksponenten.

Måling av relativ høyde
Ofte er det tilstrekkelig å beregne høyden relativt til der man starter å gjøre målinger. Det er spesielt aktuelt ved oppskyting av CanSat. En lur måte å nulljustere høydemåleren på er å sette p1 lik lufttrykket når man slår på spenningeni tillegg til at man setter referansehøyden h1 lik 0, som vist på figuren under.



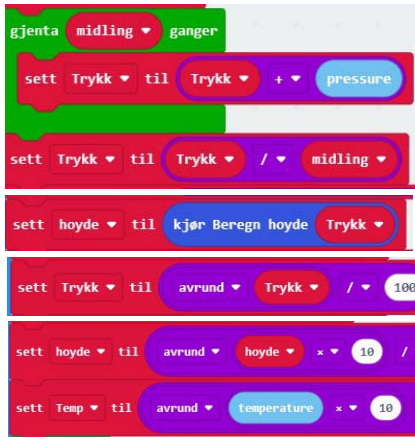
← Referansestasjonens temperatur i Kelvin

← Lokalt referansenivået (Startpunktet = 0)

← Referansenivåets lufttrykk

Det kan også være lurt å midle referansetrykket, p1, over mange målinger. Dermed reduseres usikkerheten i referansetrykket, som vil påvirke resultat av alle senere målinger.

Avlesning av sensorene og reduksjon av antall desimaler
En annen del av programmet som vi har sett før er selve avlesningen av sensorene, for luft-trykk og temperatur. Avrundingen bør imidlertid skje etter at høyden er beregnet av funksjonen "Beregn hoyde".



← Avleser trykksensoren "midling" antall ganger og summerer fortløpende

← Deler summen på antallet: "midling"

← Her kaller vi funksjonen som beregner "hoyde" Funksjonen har "Trykk" som argument

← Her gir vi "Trykk" bare to desimaler

← "hoyde" avrundes til én desimal

← "temperature" avleses og avrundes deretter til en desimal

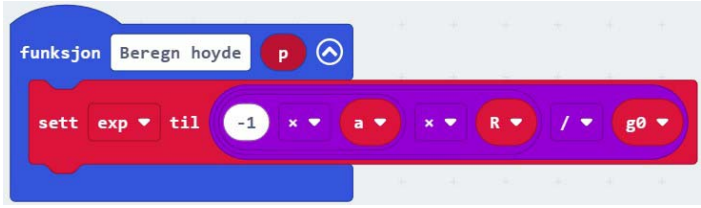
NB! *Det biblioteket vi har valgt å bruke leser av lufttrykket i Pa. Hos andre, blant annet biblioteket som er anbefalt av Makecode under "Utvidelser", gir mulighet til å lese av i hPa eller Pa. Leser man av sensoren i hPa, så rundes verdien av og man mister nøyaktighet.* ***Velg derfor å lese av i Pa.***

Funksjon som beregner høyden
Vi har lagt hele utregningen fra trykk til høyde i funksjonen: "Beregn høyde". En funksjon er en liten programbit som gjør en spesiell oppgave. Vi kan kalle funksjonen fra hoved-programmet når vi måtte ønske det. Idet vi kaller funksjonen legger vi ved variabelen *trykk*, som funksjonen trenger for å beregne høyden.

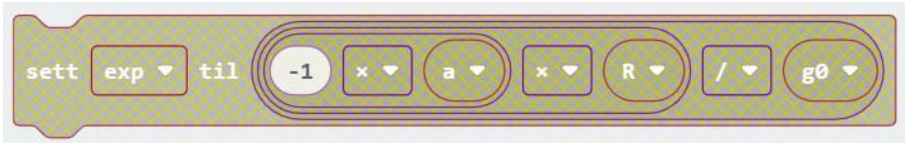
Vi skal nå lage funksjonen som vi kaller *Beregn høyde*.
Trykk for *Avansert* og velg *Funksjoner*

Skriv inn navnet på funksjonen: "*Beregn hoyde"* og legg til parameteren: *Nummer* som dukker opp som et felt hvor det står *tall*. I stedet for tall bruker vi variabelen *p* som er målt lufttrykk som skal inngå i formelen for å beregne høyden. Velg *Fullført* og du er tilbake i programmet som nå inkluderer funksjonsblokken.

Vi skal nå fylle gapet med det funksjonen skal gjøre. Vi begynner med å beregne eksponenten *exp = – a * R / g0,* dette gjør vi på vanlig måte som vist på figuren under.



$$\exp = -\frac{a \cdot R}{g0}$$



For at det skal være lettere å se hvordan vi bygger opp formelen for eksponenten, har vi valgt å ta utregningen ut av funksjonen slik at vi kan se de enkelte delene av utregningen. Når vi skal bygge opp en slik beregning, begynner vi med den ytterste som her vil være (? / g0), der "?" erstattes med (? * R) og der "?" til slutt erstattes av (-1 * a).

Beregning i JavaScript

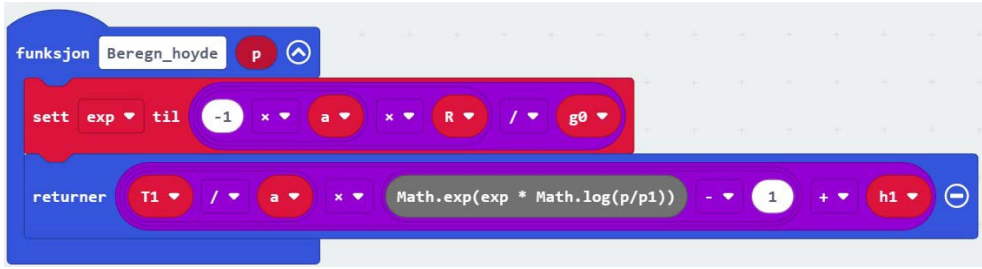
Resten av funksjonen skriver vi i JaveScript (alternativt i Python). Når vi velger JavaScript vil eksponenten, som vi alt har skrevet i blokkode, framstå som vist på figuren under.


```
1   function Beregn_hoyde (p: number) {
2       exp = -1 * a * R / g0
3   }
```

I denne funksjonen skriver vi inn en ny linje (3) som utfører beregning av høyden på grunnlag av lufttrykket p. Den ferdige funksjonen skrevet i JavaScript, blir da seende slik ut:

```
1   function Beregn_hoyde (p: number) {
2       exp = -1 * a * R / g0
3       return T1 / a * (Math.exp(exp * Math.log(p/p1)) - 1) + h1
4   }
```

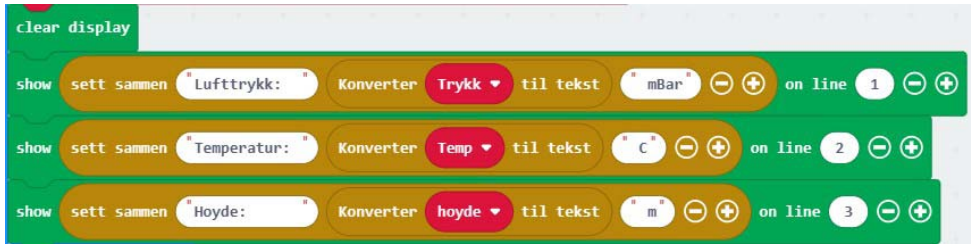
Vi legger merke til at vi skriver p/p_1^{exp} som *(Math.exp (exp * Math.log(p/p1)).* Grunnen er at vi vil tvinge MakeCode til å gjøre beregningen med JavaScipt biblioteket. Bruker vi JavaScipt funksjonen *Math.pow(a, b),* som vil være det naturlige, gjenkjenner MakeCode denne som eksponentfunksjonen og gjør den om til a**b (a^b) som vi fra før vet ikke fungerer. Bruker vi derimot logaritmer, unngår vi dette og får følgende kode når vi konverterer tilbake til blokkode:



Vi kan gjør det samme i Python om vi skulle ønske det.

Presentasjon av verdier til displayet

Vi har tidligere sett hvordan vi skriver resultatene ut til displayet.



Vi legger merke til at teksten og verdien for de tre parametrene: Lufttrykk, temperatur og høyde skrives ut på hver sin linje. I de hvite feltene skriver i teksten som er navnet på para-

meteren og etter verdien skriver vi benevningen.

4. **Test programmet**
   Gjør ferdig programmet, last det ned til Micro:bit-en og les av målt lufttrykk og relativ høyde. Beveg deg rundt på skolen i ulike etasjer og undersøke om høydeforskjellene mellom etasjene synes å stemme. Ta gjerne med høydemåleren ut og undersøk høydeforskjeller i nærområdet.

5. **Feilfinning** (er selvfølgelig avhengig av feilen som oppstår):

   - *Høydemålingen reagerer ikke eller gir svært grove utslag.*

     Dette kan skyldes at målingen gjøres i hPa hvilket gjør at verdien rundes av. Dermed vil man ikke se at høyden reagerer på små høydeforskjeller. Man må opp i nær 10 meter for at man skal se en respons.

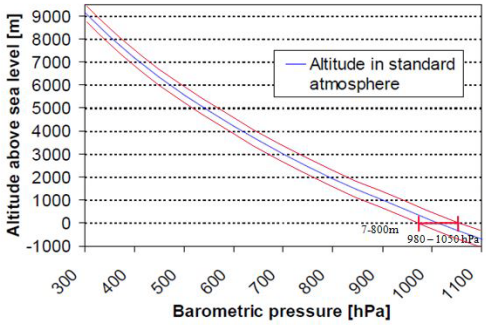
     Husk også å avrunde verdien for lufttrykket etter at verdien er brukt for å beregne høyden. Høydeberegningen trenger gjerne to desimaler etter komma for å gi tilfredsstillende nøyaktighet.

4.8 Oppdrag 8 – Nullpunktjuster høydemåleren til havnivået

Oppdrag 8 – Nullpunktjuster høydemåleren
Utføre nullpunktjustering fra relativ til absolutt høyde over havet.
Læringsutbytte: Betydningen av *nullpunktjustering*. Nullpunktjustering ved hjelp av en nærliggende referansestasjon.

Før vi kan bruke høydemåleren til å finne vår absolutte høyde over havet, må den nullpunktjusteres. Å bruke begrepet kalibrering blir for ambisiøst i denne sammenhengen.

1. **Behovet for nullpunktjustering**
   Diagrammet til høyre viser sammenhengen mellom trykk og høyde med økende høyde over havnivået. Normale variasjoner i lufttrykket ved havnivået kan typisk være fra 980 hPa til 1050 hPa (millibar). Denne naturlige variasjonen kan derfor gi en variasjon i beregningen absolutt hoh på 7 – 800 meter dersom vi ikke nullpunktjusterer målingene.

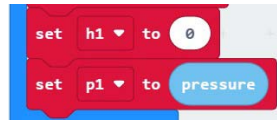


   Vi skjønner derfor at det er svært viktig med hyppig nullpunktjustering dersom vi skal kunne stole på instrumentets absolutte høydemålinger. Dette fikk Widerøes piloter føle på kroppen 10. februar 2020, da lufttrykket ble så lavt (ca. 940 mBar ved havnivå) at de barometriske høydemålerne i noen av de eldre Widerøe-flyene ikke kunne nullpunktjusteres, slik at flyene måtte settes på bakken til lufttrykket hadde steget.

   Det er imidlertid ikke så viktig å bestemme absolutt høyde over havet for vår CanSat. Da er det som oftest tilstrekkelig å nulljustere ved utskytningspunktet som omtalt over.

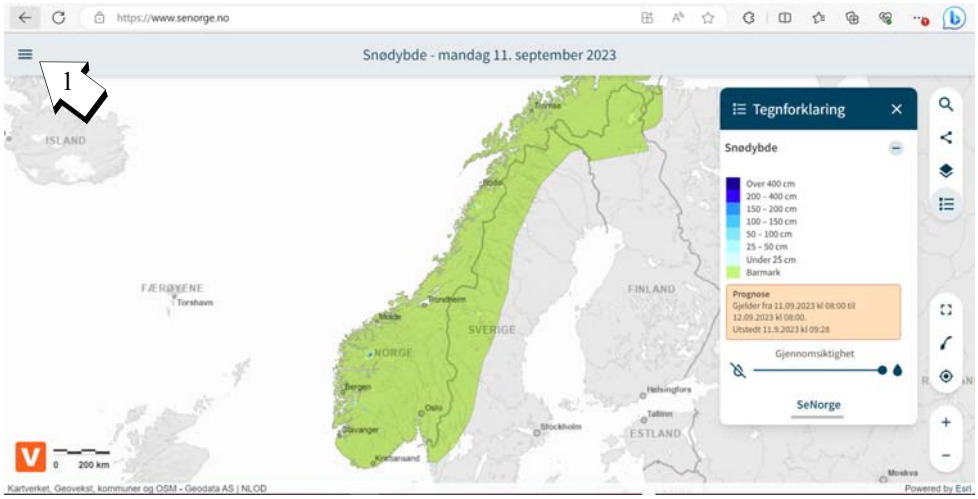
Måling av relativ høyde i stedet for absolutt høyde

Noen ganger ønsker vi å måle *høyden relativt til startstedet*, f.eks. relativt i forhold til stedet der vi sender opp CanSat-en. Dvs. at *p1*, referansetrykket på utskyningsstedet, settes lik lufttrykket når vi slår på strømmen ("*ved start*"), og referansehøyden *h1* = 0.



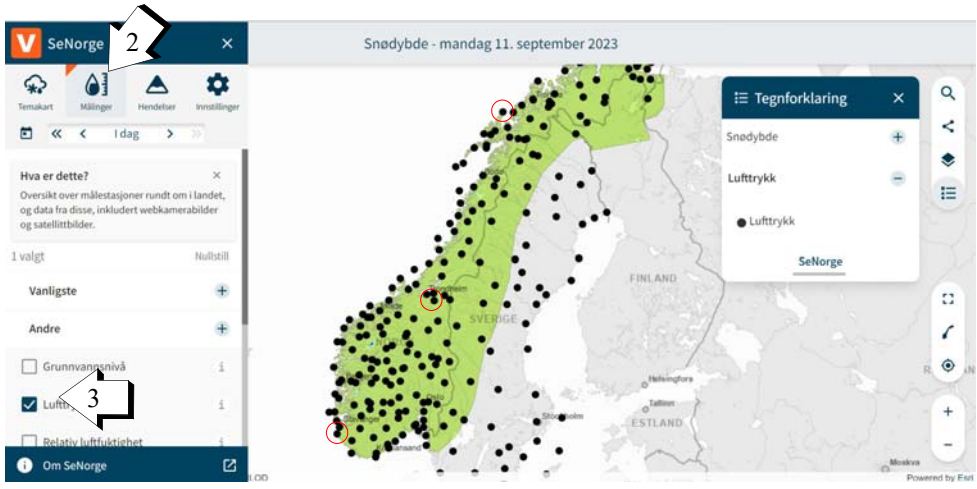
2. Kalibrerte verdier for lufttrykk finner vi på https://www.senorge.no/

www.senorge.no er en åpen portal på Internett, som viser daglig oppdaterte kart over snø-, vær- og vannforhold, og klima i Norge – og mye mer.



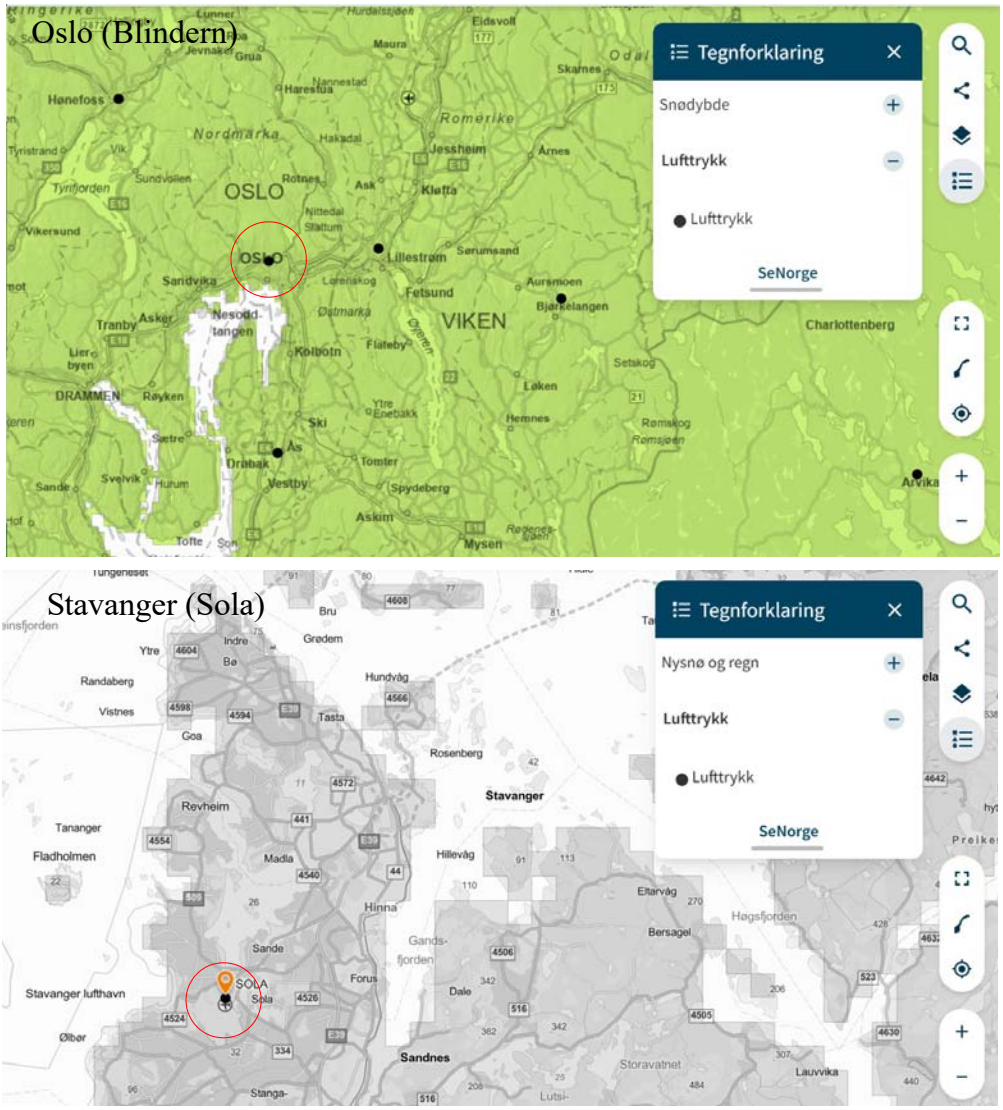
Velg menyen oppe i venstre hjørne (1). Siden vi ønsker å finne lufttrykket hos en nærliggende stasjon, trenger vi å få en oversikt over målestasjoner landet over, som måler lufttrykk.

3. **Velg *Målinger (2)***



4. **Velg å vise *lufttrykk***
   Kryss av for visning av målestasjoner som måler *Lufttrykk.* Da vil alle de offisielle målestasjonene i Norge vises på kartet som svarte prikker.

5. **Finn din nærmeste stasjon**
   Zoom inn på den delen av landet der du befinner deg. F.eks. på Andenes (Andøya flystasjon), Trondheim (Voll gård), Sandnes (Sola) eller Oslo (Blindern).



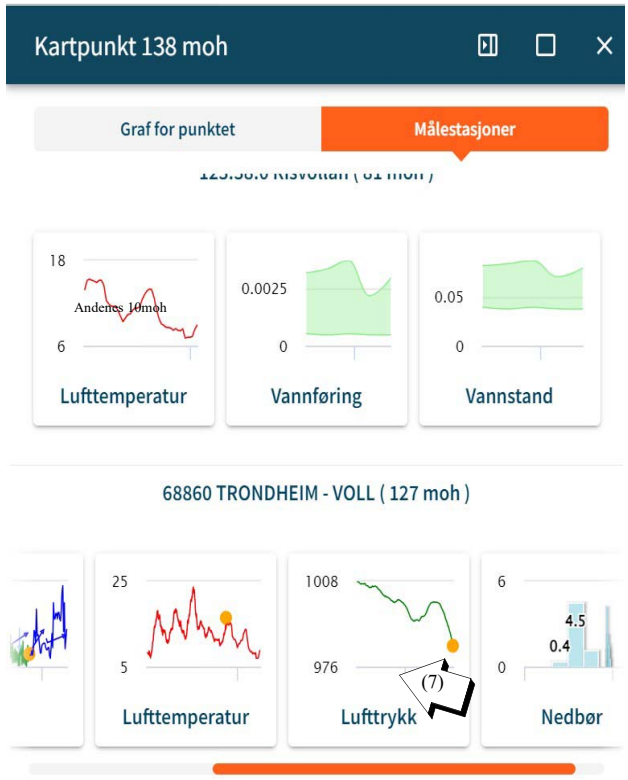
Vi kan f.eks. zoome inn på Trondheim og finne Voll gård, der målestasjonen ligger.

Dobbelklikk på punktet der målestasjonen ligger, og det kommer opp en oversikt over målinger denne stasjonen registrerer.

6. **Lufttrykk og høyde over havet (moh)**

Etter å ha klikket på den valgte målestasjon, får vi opp flere ikoner hvorav lufttrykket de siste 30 døgn er en av dem. Målingene oppdateres en gang i timen så avvikene burde være til å leve med. Les også av målestasjonens høyde over havet. For Andenes er dette 10 moh, for Voll i Trondheim 127 moh og for Blindern er det 94 moh.

Ved å klikke på ikonet for lufttrykk får vi opp en graf med lufttrykket over en periode. Vi er primært interessert i trykket gjort ved siste måling, som vises både grafisk og på en merkelapp ved grafen. Les av og noter siste oppdaterte verdi for lufttrykket.



Denne dagen, den 11.09.23 kl. 14:00, var lufttrykket 983,9 mBar (hPa) ved Voll gård i Trondheim. Vi har tidligere notert oss at den ligger 127 moh.

7. **Legg verdiene inn i programmet**
   Målestasjonens høyde over havet og lufttrykk, legges så inn som parametere h_1 og p_1.

← Temperatur i (Kelvin)

← Stasjonens høyde over havet (m)

← Stasjonens lufttrykk (mBar)

8. **Test programmet**
   Gjør ferdig programmet, last det ned til Micro:bit-en. Les av målt lufttrykk og høyden over havet. Ta med deg høydemåleren ut og undersøk høyden over havet rundt i nærområdet. Kontrollere de avleste høydene med et lokalt kart med høydekoter.

9. **Endring over tid**
   La høydemåleren ligge på samme sted over tid og se hvordan den målte høyden endrer seg. Dette skyldes sannsynligvis at det lokale lufttrykket endrer seg og dermed gjør målingen av høyden over havet unøyaktig eller ugyldig. En skjønner dermed fort at nullpunktjustering er "ferskvare".

   Dette kan skyldes at målingen gjøres i hPa hvilket gjør at verdien rundes av. Dermed vil man ikke se at høyden reagerer på små høydeforskjeller. Man må opp i nær 10 meter for at man skal se en respons.

10. **Feilfinning** (er selvfølgelig avhengig av feilen som oppstår):

    - *Lufttrykket kommer ikke fram når vi har funnet målestasjonen.*

      Dersom det er vanskelig å finne lufttrykket ved enkelte målestasjoner, så forsøk å klikke på overskriften, f.eks. 68860 TRONDHEIM – VOLL (127moh).

      Om dette ikke nytter, velg en annen nærliggende stasjon.

5 Oppdrag – CanSat-kurs

I dette kapittelet skal vi ferdigstille CanSat "primary mission" som skal inneholde følgende deler. Noen av oppdragene er alt løst i det innledende grunnkurset:

1. Montering av CanSat (kapittel 2, side 16)
2. Temperaturmåling (avsnitt 4.5, side 45), kan også bruke BMP/E280 (avsnitt 4.6 på side 47)
3. Trykkmåling (avsnitt 4.6, side 47)
4. Høydemåling (avsnitt 4.7, side 50)
5. Sender for overføring av data til bakkestasjon (avsnitt 5.3.1 på side 68)
6. Mottaker for mottak og lagring av data (avsnitt 5.3.2 på side 70)
7. Lagring av data på SD-kort (avsnitt 5.4 på side 73)
8. Sammenstilling og test av hele systemet (avsnitt 5.5 på side 75)
9. Analyse og presentasjon av data (kapittel 10, side 108)

Vi er ferdige med punktene 1 – 4 som er hovedinnholdet i oppdragene 6, 7 og 8 i grunnkurset. I dette kapittelet skal vi ta for oss punktene 5 – 9 (oppdrag 9 – 13).

5.1 Oppdragene 9 – 13

Oppdrag 9 – Sette opp og send data via radio
Installere biblioteket for radioen E32. Konfigurer radioen hos en sender og en mottaker. Send og motta testdata på ønsket kanal.
Læringsutbytte: Lære å sette opp E32, med riktig kanal og effekt slik at data sendes til mottakeren. Oppsett av E32 skal primært kunne gjøres i "*ved start"* og sekundært ved hjelp av program levert av fabrikanten (se vedlegg E.1, side 142).

Oppdrag 10 – Send og motta data via radio
Bruk fiktive eller virkelige måledata, og legg disse inn i en streng for overføring fra CanSat-en til bakkestasjonen, slik at dataene kan logges i en CSV-fil for senere analyse eller vises i real time i monitoren til MakeCode.
Læringsutbytte: Lære å legge data inn i en streng for senere å hente dem opp for analyse.

Oppdrag 11 – Skrive til SD-kort
Montere SD-kort terminalen og legge måledata ned på SD-kortet organisert som en CSV-fil (Comma Separated File).
Læringsutbytte: Administrere og legge data ned på SD-kortleseren.

Oppdrag 12 – CanSat – Systemdesign
I dette oppdraget skal vi sette sammen alle bitene av programmet slik at vi får et system som gjør målinger av tidspunkt, lufttrykk, temperatur (evt. luftfuktighet) og beregner høyden utfra målt lufttrykk. Høydemåleren skal nullpunktjusteres slik at den viser rett høyde i forhold til

havet. Dataene skal skrives til SD-kortet ombord i CanSat-en samtidig som data sendes ned til bakkestasjonen via radio. Dataene skal samles i en CSV-fil ved bakkestasjonen.
Læringsutbytte: Sette sammen programbiter til et system. Teste og evt. feilsøke systemet.

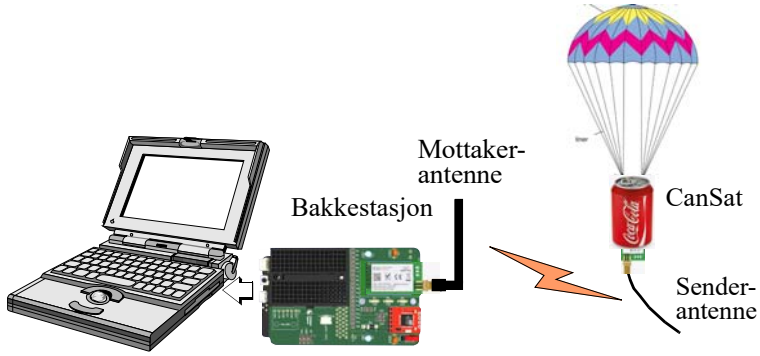
Oppdrag 13 – Analyse av data
Hente dataene fra fil, skrive enkel programvare for plotting og analyse av resultatene i Python eller Excel eller lignende programvare.
Læringsutbytte: Lære å lese dataene inn i den aktuelle programvaren for analyse og lagring av dataene.

5.2 Oppdrag 9 – Sette opp og send data via radioen E32-433T20DC

I dette oppdraget skal vi konfigurere radioen og sette den til riktig kanal.

Oppdrag 9 – Sette opp og send data via radio
Installere biblioteket for radioen E32. Konfigurer radioen hos en sender og en mottaker. Send og motta testdata på ønsket kanal.
Læringsutbytte: Lære å sette opp E32, med riktig kanal og effekt slik at data sendes til mottakeren. Oppsett av E32 skal primært kunne gjøres i "*ved start"* og sekundært ved hjelp av program levert av fabrikanten (se vedlegg E.1, side 142).

Radioen bruker vi til å sende data fra CanSat-en som faller gjennom atmosfæren og ned til mottakeren ved PC-en som står på bakken, også kalt bakkestasjonen.



Vi må først sørge for at senderen og mottakeren bruker samme kanal og sender og mottar like fort. Dette kalle vi å *konfigurere* eller *sette opp* radioene.

Det er flere måter vi kan konfigurere radioene på:

- Primært ved hjelp av en innledende kode i programmet som vi skal se nærmere på om litt
- Sekundært ved hjelp av et program utviklet av leverandøren av radioene, som er omtalt i vedlegg E.1, side 142 for den interesserte.

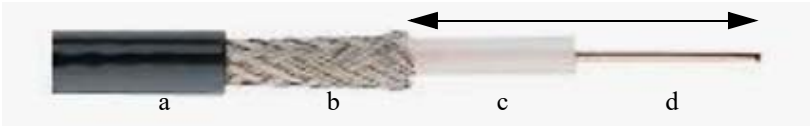
Men først må vi lage myke antenner til senderen og mottakeren. "Myke" for at de skal kunne pakkes inn i en rakett eller gjøre CanSat-en mer robust mot skade. Vi kan gjerne bruke stive antenner på bakkestasjonen, eller til og med håndholdte retningsantenner.

Lag en myk antenne

1. Lag antenna

For å kunne sende data må vi ha en antenne. Dersom vi kun ønsker å teste radioen i laboratoriet kan vi gjerne bruke en stiv antenne.

Dersom vi derimot skal bruke antenne i forbindelse med oppskyting, må vi lage en myk antenne av en *koaksialkabel* som er tilpasset til den frekvensen vi ønsker å sende på. I dette tilfellet ca. 434 MHz. En koaksialkabel er en kabel med en ledning innerst (d) med hvit plastisolasjon (c) og en vevd kappe av fortinnet kobber utenpå (b). Ytterst er det en ny plaststrømpe som kan være blank, rødlig, sort eller hvit (a).

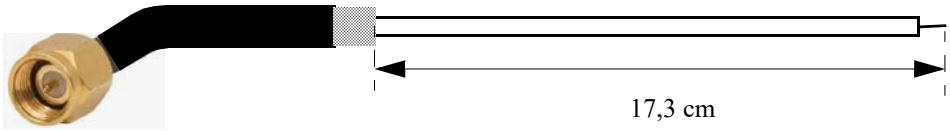


Vi ønsker å lage en såkalt "pisk"-antenne som kun er en ledning med en lengde tilpasset den *frekvensen* vi skal bruke, ca. 434MHz (434 000 000 Hz). En vanlig lengde på en slik antenne er 1/4 av bølgelengden. Dermed må vi finne ut hvor lang én bølgelengde er for frekvensen 434 MHz. Når vi vet at Hz = 1/sek (1/s), og fart måles i meter/sek (m/s) og bølgelengde måles i meter (m), så kan vi sette opp:

Bølgelengden = Farten til radiobølger / Frekvensen = 300 000 000 [m/s] / 434 000 000 [1/s]

Bølgelengden = 0,6912 meter

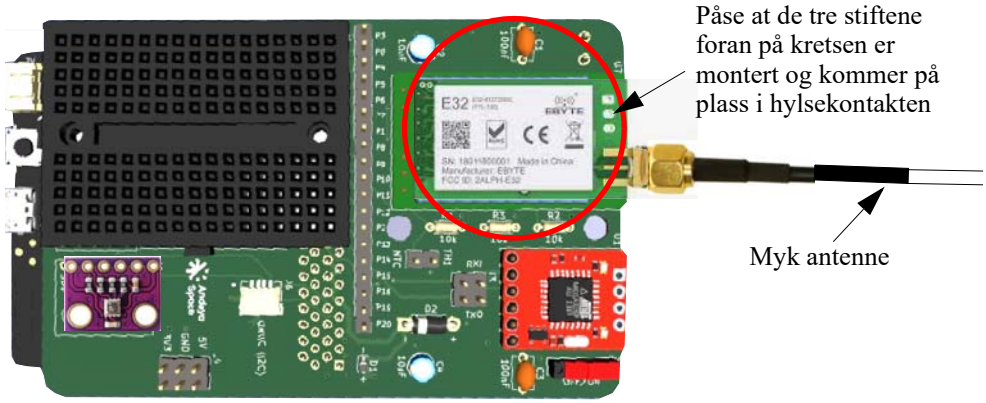
1/4 bølgelengde = 0,173 meter = 17,3 cm



Legg merke til at antennen begynner der hvor skjermen slutter og strekker seg til der ledningen slutter. Det er viktig at kontakten i enden er intakt og koblet til ledningen. Det er den som skal overføre signalet fra radioen til selve antenna.

Oppkobling

2. **Oppkoblingen** av radioen med antenne er særdeles enkel, vi plugger inn radioen E32 som vist på figuren under.



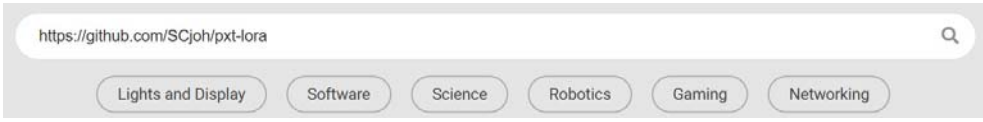
Konfigurer radioene ved hjelp av programkode:

3. **Last ned biblioteket**
   Til dette skal vi laste ned et litt modifisert bibliotek for radioen E32 for vår bruk[8]. Biblioteket finnes på følgende nettadresse:

   https://github.com/SCjoh/pxt-lora

   For å hente inn biblioteket velger vi "*Utvidelser*" fra menyen ...

   ... og skriver nettadressen til biblioteket i søkefeltet:



   Om litt får vi opp et ganske anonymt ikon som vi velger å installere. Dermed blir biblioteket installert og følgende mappe dukker opp i menyen:

For å teste kommunikasjonen trenger vi ha en sender og en mottaker. Dvs. vi må ha to CanSat-er. Dersom vi kun har bygget en så må vi alliere oss med en annen gruppe slik at vi får testet kommunikasjonen. Både senderen og mottakeren må konfigureres på samme måte:

8. Biblioteket er skrevet av Alexandre Frolov og modifisert av Jørn Hafver.

Program for konfigurering av radioene

4. **Skriv program for *konfigurering* av radioene**[9]
Vi skal nå sette opp radioene ved hjelp av et eget program som vi skal skrive i blokkode. Programmet forteller Micro:bit-en hvilke porter radioen er koblet til, hvor fort Micro:bit-en skal kommunisere med radioen og hvor fort radioen skal sende data til bakkestasjonen. Oppsettet skal være det samme for senderen i CanSat-en som for mottakeren i bakkestasjonen. Oppsettet gjøres i blokken *"ved start"* som vist i figuren under.

a)

b)

a) Først forteller vi programmet hvilke av radioens terminaler som er koblet til hvilke porter hos Micro:bit-en (M0: P9 og M1: P8, dessuten er AUX: P16, TX: P14 og RX: P15). Vi forteller den også hvilken datarate som skal brukes for å overføre data til radioen (BAUD: 9600). Vi legger merke til at TX hos radioen kobles til RX hos Micro:bit-en og RX hos radioen kobles til TX hos Micro:bit-en.

b) Så må vi fortelle radioen hvilken adresse og hvilken kanal den skal bruke for å overføre data (i vårt eksempel har vi valgt: ADDR: 1 og CHANNEL: 23[10]), og tilslutt hvor fort radioen må ta imot data fra Micro:bit-en (UART BAUD: 9600), og hvor fort den skal sende data "over lufta" (AIR BAUD: 2400). I tillegg må vi angi utgangseffekten til senderen (POWER: 0, 1, 2 eller 3), hvor 0 er maks effekt (100mW) og 3 er minimum effekt. I klasserommet kan det være greit å bruke lav effekt (3) siden det er kort avstand og mange som sender samtidig. Sendereffekten bør økes når vi skal bruke CanSat-en i felt.

c) Legg merke til at vi setter "sann" på både "CONFIGURATION MODE" og på "SAVE CONFIG", hvilket betyr at dette oppsettet gjelder kun for å konfigurere senderen og mottakeren og at konfigureringen skal lagres.

d) For å se om vi har fått satt opp radioen riktig kan vi f.eks. be om at den skriver ut oppsettet til LED-displayet når vi trykker på knapp B.

9. Oppsett programmet er skrevet av Jørn Hafver.
10. Det viser seg at det er lurt at hvert lag velger hver sin kanal, det ser ikke ut til å være nok å ha forskjellig adresse

Om alt er riktig, vil vi se følgende rulle over displayet: C0 00 00 1A 17 C7 der C0 00 (ADDR=) 00 1A (CHANNEL=) 17 (POWER=) C7.

d) Last opp og kjør programmet både i senderen og i mottakeren.
Trykk knapp B og sjekk om konfigureringsdatene ser fornuftige ut.

5. **Feilfinning** (er selvfølgelig avhengig av feilen som oppstår):

- *Ingen respons når vi ber om konfigureringsdata (trykker knapp B).*

Sjekk at koden er som beskrevet over og at batteriet er tilkoblet (det er ikke nok å ha tilkoblet USB-kabelen).

Sjekk også at det står "sann" i begge innboksene.

Sjekk at riktig bibliotek er benyttet: https://github.com/SCjoh/pxt-lora

Sjekk at spenningen på M0 og M1 er 5V, Vcc = 5V og at GND er forbundet med GND

Program for overføring av data via radio

6. **Skriv program for å *teste overføring* av data**
Vi skriver så et testprogram som kan sende og motta en enkel melding: "Hei". Det *samme programmet* kan brukes i både senderen og i mottakeren. Vi må imidlertid utvide programmet vårt når vi skal sende og motta ordentlige måledata, det kommer vi tilbake til[11].

Vi trenger ikke å sette opp radioene på nytt, men må fortelle programmet hvordan radioene er koblet til Micro:bit-en, det gjør vi som tidligere i *"ved start"*-blokken:

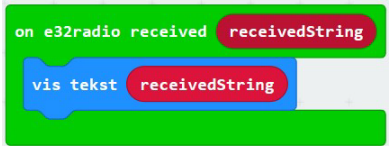
Nå vil vi bruke radioene til å sende data fra CanSat-en til bakkestasjonen og setter derfor CONFIGURATION MODE = "usann" for normal kommunikasjon. Dette legger M0 og M1 til 0V.

11. Testprogrammet er skrevet av Jørn Hafver

Dernest vil vi sende meldingen: “Hei” når vi trykker på knapp A:

Videre vil vi som sist, se hvordan radioen er konfigurert når vi trykker på knapp B:

og vise evt. mottatte meldinger på displayet:



7. **Kjør programmet**
   Bygg opp test-programmet. *Gå sammen med en annen og test sambandet begge veier.* Husk at begge må bruke samme adresse og kanal.

8. **Bruk OLED-display**
   Dersom dere har et OLED-display så kan dere montere displayet mellom Micro:bit-en og CanSat-kortet og skrive ut meldinger og evt. målinger som mottas, på OLED-displayet. Pass på at micro:bit-en får spenning fra USB-kabelen og at batteriet er tilkoblet.

9. Feilfinning (er selvfølgelig avhengig av feilen som oppstår):

- *Ingen respons på LED-displayet hos mottakeren når vi sender data (trykker knapp A).*

  Sjekk at koden er som beskrevet over og at batteriet er tilkoblet (det er ikke nok å ha tilkoblet USB-kabelen).

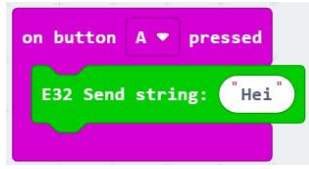
  Sjekk spesielt at det står “usann” i innboksen “ved start”.

  Trykk knapp B og sjekk at både senderen og mottakeren er konfigurert til samme adresse og kanal.

  Sjekk at M0 og M1 = 0V og at senderen og mottakeren har 5V på Vcc.

  Om dere har flere radioer, bytt begge radioene. Husk at også de må være konfigurert til samme adresse og kanal.

  Evt. forsøk å bytte ut *E32 Send string*-blokken med adresse og kanal, med blokken på figuren til høyre:



- *OLED-displayet synes ikke å virke.*

  Vi har erfart at dersom batteriet skal forsyne både micro;bit og OLED-display, kan spenningen bli for lav.

5.3 Oppdrag 10 – Send og motta måledata via radio E32

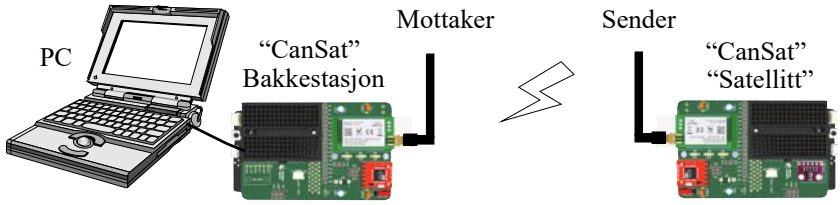
I dette oppdraget skal vi beskrive hvordan vi kan overføre måledata fra CanSat-en til bakkestasjonen (PC-en):

Oppdrag 10 – Send og motta data via radio
Bruk fiktive eller virkelige måledata, og legg disse inn i en streng for overføring fra CanSat-en til bakkestasjonen, slik at dataene kan logges i en CSV-fil for senere analyse eller vises i real time i monitoren til MakeCode.
Læringsutbytte: Lære å legge data inn i en streng for senere å hente dem opp for analyse.

Vi skal i dette oppdraget vise hvordan vi kan sende og motta måledata med to identiske Micro:bit CanSat-er. For den som ønsker å bruke en FTDI-krets og et terminalprogram se E.2, side 145.

For enkelthetsskyld kan vi, om vi ønsker det, først lage fiktive data som vi sender over til bakkestasjonen før vi skifter ut disse med virkelige måledata. Siden vi alt har montert trykk- og temperatursensoren BMP/E280 så velger vi å sende over data fra denne.

Send og motta data med to Micro:bit CanSat-er



5.3.1 Senderenheten (CanSat - Satellitt)

1. **Installasjon av biblioteker**
   Om vi ikke alt har gjort det så må vi installere biblioteker for radioen E32 og evt. BMP/E280 dersom vi ønsker å bruke virkelige data fra trykksensoren. Disse bibliotekene installerer vi på vanlig måte med å velge *Utvidelser* og skrive inn følgende i søkefeltet:

   Biblioteket til radioen E32, finner vi her:
   https://github.com/SCjoh/pxt-lora

   Biblioteket til BME280, finner vi her:
   https://github.com/ElectronicCats/pxt-bme280

2. **Konfigurering av radio**
   Vi forteller som sist, programmet hvilke porter hos Micro:bit-en radioen er tilkoblet. Dette gjør vi med blokken vist på figuren til høyre.

   I tillegg velger vi å slå på BME280 med å sette "set power" til "on".

3. **Sjekk konfigureringen av radioen**
   For å være sikker på at radioen er satt opp korrekt, inkluderer vi en kommando som når vi trykker på bryter B, viser konfigureringen av radioen. For å få til det leser vi parametrene som ligger i variabelen "*e32parametere*", og som følger med radiobiblioteket. Displayet kan f.eks. vise C0 00 00 1A 17 C7, avhengig av hvordan vi har konfigurert radioen.

4. **Send måleverdier**
   Så avleser vi og sender måleverdiene fra trykk- og temperatursensoren. På figuren under ser vi oppbyggingen av strengen som sendes over radioen. Figur A) viser kommandoen slik den ser ut i programmet. B) viser hvordan kommandoblokken er bygget opp.

   A)

   B)

   Vi legger merke til at tallverdiene gjøres om til tekst og det legges inn et skilletegn "-" som skal gjøre det lettere å skille verdiene når de mottas av bakkestasjonen.

   Alternativt kan man legge inn tilfeldige tall istedet for trykk og temperatur for å teste overføringen.

   NB! *Pass på at adressen og kanalen i sendekommandoen er i overensstemmelse med adressen og kanalen hos senderen og mottakeren (markert med piler på figuren over).*

5. **Lag programmet**
   Sett sammen programmet og overfør det til senderenheten til CanSat-en.

På sikt kan vi tenke oss å overføre følgende data:

- Identitet "-"
- Tid [msek] fra start "-"
- Temperatur [°C] "-"

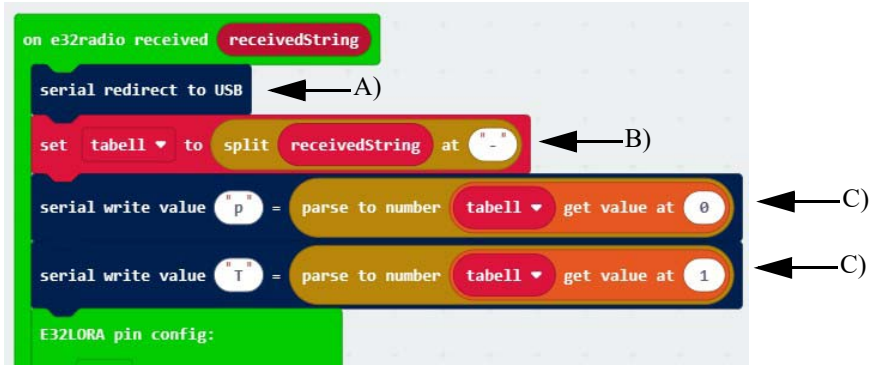
- Evt. fuktighet [%] “-”
- Lufttrykk [mBar] “-”
- Høyde [m]

Hver måling skal skilles med et skilletegn som f.eks. kan være en bindestrek eller et komma.

5.3.2 Mottakerenheten (CanSat - Bakkestasjonen)[12]

1. **Installasjon av biblioteker**
   Til bakkestasjonen trenger vi kun biblioteket til radioen E32:

   Biblioteket til radioen E32, finner vi her:
   https://github.com/SCjoh/pxt-lora
2. **Initialisering av mottakeren**
   Radioen på mottakersiden må konfigureres på samme måte som på sendersiden, med samme adresse og kanal. Dette gjør vi som sist med blokken som vist på figuren til høyre.
3. **Sjekk konfigureringen av radioen**
   Vi legger den samme konfigureringssjekken i mottakeren. Dermed kan vi evt. sjekke om konfigureringen av sender og mottaker er like.
4. **Motta data**
   Å motta måledatene fra CanSat-en er noe mer utfordrende enn å sende dem, som vist i figuren under:



A) Det første vi må gjøre når vi har mottatt data fra CanSat-en, er å endre serielinjen fra å motta fra radioen E32 (pin 15), til å overføre data til USB-en og videre til PC-en. Dernest må vi opprette variabelen “tabell” i “variabler”.

12. Programmet er skrevet av Jørn Hafver

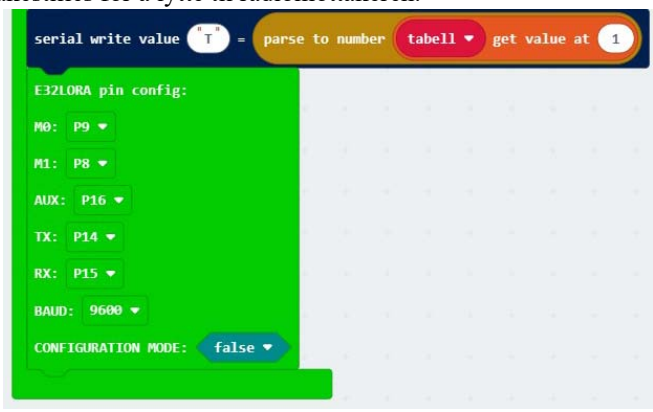
B) Etter at vi har mottatt en melding via radioen på Rx serielinjen, skal de mottatte dataene omorganiseres og sendes videre via USB-pluggen til PC-en for presentasjon og lagring. Omorganiseringen skjer ved at dataene splittes opp i to tekststrenger, en for lufttrykk og en for temperatur og legges inn i hver av kolonnene i tabellen "tabell". Oppsplittingen av tekststrengen gjøres der vi har plassert skilletegnet ("-").

C) Dernest hentes verdiene (som tekst) fra tabellen, fra kolonne "0" som merkes med "p", og fra kolonne "1"som merkes med "T". De to tekststrengene omformes deretter til et tall, som så sendes til PC-en via serielinjen og USB-pluggen.

Om nødvendig kan vi overføre flere tekststrenger som representerer verdier, bare adskilt med skilletegn. Tabellen får da flere kolonner som vi kan hente verdier fra for å sende over til PC-en.

5. **Omdiriger datastrøm for nye mottak**

Etter at vi har sendt dataene til PC-en, må vi sørge for å klargjøre mottakeren for å motta nye data fra CanSat-en via radioen. Det gjør vi med følgende blokk hvor serielinjen igjen tilbakestilles for å lytte til radiomottakeren.



6. **Presenter dataene i MakeCode**
Dataene skal nå ha ankommet PC-en. Vi går inn i MakeCode og programfilen (blokkoden) for mottakeren, der det har dukket opp et ikon nede til venstre (A), for "real time" presentasjon av dataene etter som de ankommer Micro:bite-en.

På figuren ser vi lufttrykket (p – øverst) og temperaturen (T – nederst) på skjermen. Øverst i høyre hjørne (B) ser vi symbolet for nedlasting til fil for videre behandling i andre programmer f.eks. Python eller Excel, se avsnittene 10.2, side 109 og 10.3, side 116.

7. **Feilfinning** (er selvfølgelig avhengig av feilen som oppstår):

- *Ingen mottatte data hos mottakeren.*

  Sjekk at koden hos sender og mottaker er som beskrevet over og at batteriet er tilkoblet hos senderen og power-bryteren er slått på.

  Sjekk at adresse og kanal er den samme hos sender og mottaker. Sammenlign kode når knapp B trykkes. Om disse ikke stemmer, konfigurer radioene på nytt.

  Sjekk at sensoren som skal levere data (BME/P280) står rett vei i sokkelen og har spenning på Vcc (3V).

  Evt. konfigurer et nytt sett med radioer og undersøk om det gir bedre resultat.

- *Ukjente data dukker opp hos mottakeren uten at vi sender.*

  Det er lurt å merke utsendte data med en identitet først i strengen, slik at det er lett å kjenne igjen den mottatt teksten, evt. at man kan identifisere hvem andre som har sendt den. Sjekk om dere har samme adresse og kanal. Evt. sørg for at en av dere skifter adresse og evt. kanal.

  Båndet er svært smalt. Det er egentlig bare kanal 23 og 24 som kan brukes i Norge. Så de ulike signalene bør hovedsakelig skilles ved ulike adresser.

  Forsøk om det er mulig å redusere sendereffekten hos den som forstyrrer.

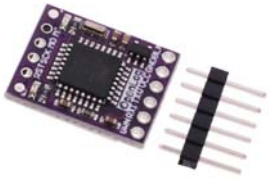
5.4 Oppdrag 11 – Skrive til SD-kort

Oppdrag 11 – Skrive til SD-kort
Monter SD-kort-terminalen og legg måledata ned på SD-kortet organisert som en CSV-fil (Comma Separated File).
Læringsutbytte: Administrere og lagre data på SD-kort.

Om vi vil kan vi inntil videre bruke simulerte data og vente med å lagre virkelige data til vi kommer til oppdrag 12, eller vi kan bruke virkelige data hentet fra BMP280 eller BME280 som vist i oppdrag 10.

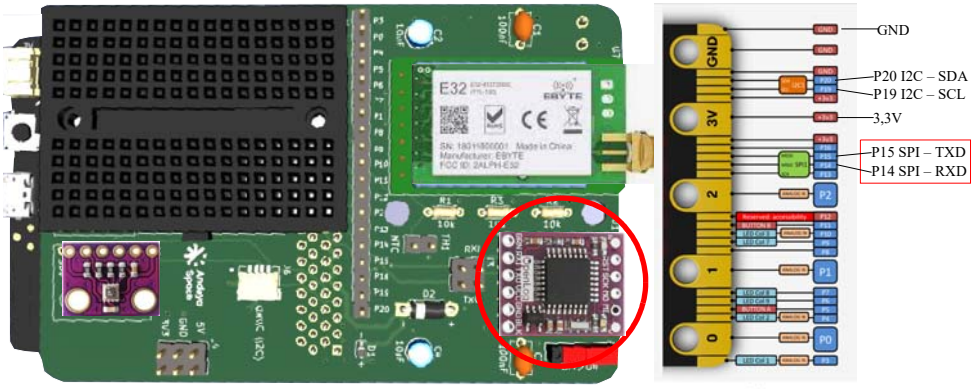
1. **Monter SD-kort terminalen**
   Normalt leveres OpenLog uten stiftlist, i så fall må denne loddes på først.

   Montering av stiftlist
   Det er særdeles viktig at den loddes inn med IC-kretsen på oppsiden og selve holderen for SD-kortet på undersiden, slik at rekkefølgen av pinnene passer til utlegget på kretskortet. Den korte siden av stiftlista skal stikkes opp fra undersiden slik kortet er vist på figuren til høyre. Den lange siden av stiftene skal stikkes ned i sokkelen (se også punkt 16., side 24).



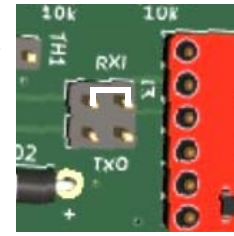
   Skulle man komme til å montere den på feil side. Kan man til nød montere kretsen motsatt vei i sokkelen, dvs. med kretsene stikkende inn over strappene J3 (RXI og TXO).

   Montering av SD-kort terminalen
   Plugg inn SD-kort terminalen i sokkelen som vist på figuren under.



   Den røde varianten fra SparkFun kan selvfølgelig også brukes, eller man kan velge å lagre data internt i minnet til Micro:bit-en, som er mulig på versjon 2 (se avsnitt 8.2 på side 102).

Montering av SD-kort



SD-kortet skyves inn i holderen på undersiden av kortet med kontaktpunktene av kortet på oversiden. OpenLog kan håndtere SD-kort på opp til 32GByte.

NB! For at OpenLog skal kunne ta imot data, må strappen RXI settes på, se bilde til høyre.

2. **Programmering**
Data overføres via UART'en som kommuniserer via pinnene Tx/Rx som er portene P15 og P14. Normalt vil vi kun sende data fra Micro:bit'en til OpenLog slik at det kun er P15 (Tx) som er i bruk og tilsvarende RXI hos OpenLog.

Siden radioen E32 og OpenLog begge bruker seriebussen (UART), så vil data som sendes til radioen automatisk også skrives ned på en fil hos OpenLog. Har vi først satt opp kommunikasjonen til E32, så har vi også gjort det for OpenLog.

Lagring av data på SD-kort
Hver gang spenningen blir slått på vil det bli opprettet en ny file, som automatisk får navnene:
LOG00000.txt
LOG00001.txt
LOG00002.txt
osv.

I tillegg opprettes en config.txt fil som inneholder konfigureringsinformasjon ved lagring av data (se avsnitt 8.1 på side 101).

Vi har erfart at ikke alle OpenLog uten videre lar seg skrive til. Vi har ikke funnet ut hva dette skyldes, men en mulighet er at billige versjoner kjøpt fra Kina ikke har installert firmeware.

3. **Skriv til SD-kort**
Send data til radioen og lagre data på SD-kortet. Sjekk at det opprettes en ny fil på kortet og at dataene lagres i filen.

4. Feilfinning (er selvfølgelig avhengig av feilen som oppstår):

- *Ingen data er skrevet til fil.*

  Sjekk at stiftlisten er montert på riktig side av kretsen, slik at koderen for SD-kortet havner på undersiden av kortet.

  Sjekk at alle stiftene kommer ned i hylsekontakten på kortet. Dvs. at kortet ikke er forskjøvet sideveis.

  Sjekk at SD-kortet er satt inn med kontaktpunktene opp, dvs. inn mot kortet.

  Sjekk at strappen er plassert på riktig sted og i riktig posisjon (se figuren over).

  Vi har også erfart at OpenLog-kloner fra Kina ikke fungerer som de skal, men at det går bedre med OpenLog fra SparkFun. Kanskje nødvendig å bytte terminal.

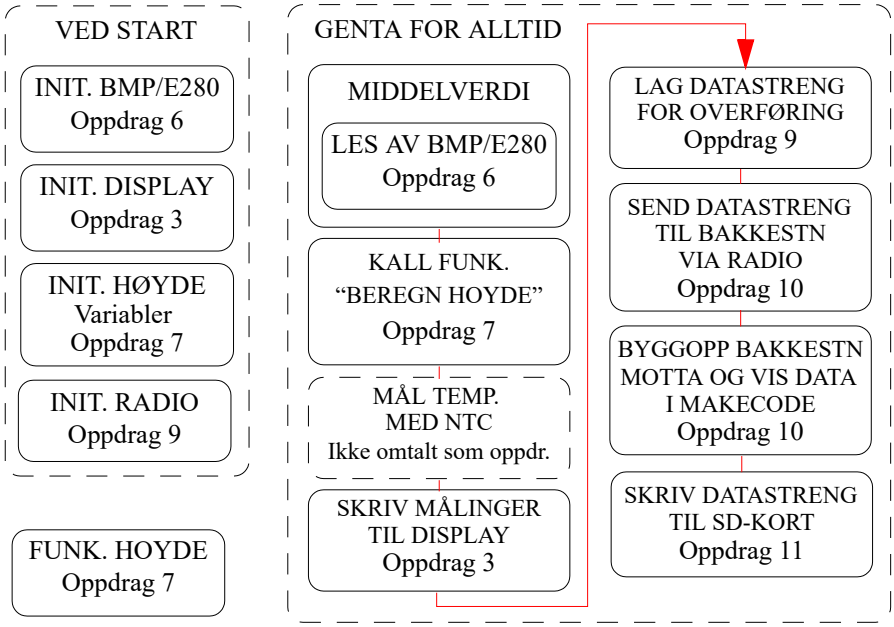
5.5 Oppdrag 12 – CanSat – Systemdesign

Oppdraget går ut på å sette sammen alle delene av prosjektet til en komplett CanSat.

Oppdrag 12 – CanSat – Systemdesign
I dette oppdraget skal vi sette sammen alle bitene av programmet slik at vi får et system som gjør målinger av tidspunkt, lufttrykk, temperatur (evt. luftfuktighet) og beregner høyden utfra målt lufttrykk. Høydemåleren skal nullpunktjusteres slik at den viser rett høyde i forhold til havet. Dataene skal skrives til SD-kortet ombord i CanSat-en samtidig som det sendes ned til bakkestasjonen via radio. Dataene skal samles i en CSV-fil ved bakkestasjonen.
Læringsutbytte: Sette sammen programbiter til et system. Teste og evt. feilsøke systemet.

Siden alle delene nå skal finnes så skal vi her kun sette opp et blokkdiagram av systemet. Oppgaven består dermed av å sette sammen de ulike delene til et komplett fungerende system.

1. **Blokkdiagrammet**
   Det er lurt å lage et blokkdiagram over programmet for å få oversikt over de enkelte delene og hvor de bør plasseres.



Vi legger merke til følgende:

- Beregning av høyden på grunnlag av lufttrykket gjøres i en egen funksjon (FUNK. HOYDE). Denne beregningen kan med fordel også gjøres i etterbehandlingen av dataene.
- Vi har inkludert displayet i denne systemoversikten. Vi må være forberedt på å fjerne utskrift til display under oppskytingen av CanSat-en. Evt. kan vi sende måledata via bluetooth over til en håndholdt Micro:bit med display om vi ønsker det. Dette er ikke beskrevet i dette heftet.

2. **Lag og test programmet**
   Bruk blokkdiagrammet til å skrive sammen hele CanSat-programmet og test om det fungerer som tiltenkt. Tenk gjennom hvordan hele programmet kan testes.

5.6 Oppdrag 13 – Analyse av data

I dette oppdraget skal vi primært presentere dataene slik at det også blir lettere å analysere dem.

> **Oppdrag 13 – Analyse av data**
> *Hente dataene fra fil, skrive enkel programvare for plotting og analyse av resultatene i Python eller Excel eller lignende programvare.*
> **Læringsutbytte:** Lære å lese dataene inn i den aktuelle programvaren for analyse og lagring av dataene.

Vi velger å benytte Python for i første omgang å presentere dataene. Vedlegg 10.1, side 108 beskriver kort hvordan vi kan lagre og presentere mottatte data i MakeCode for deretter å lagre dem i file for senere å hente dem inn i f.eks. Python eller Excel.

Bruk beskrivelsen i kapittel 10 til å gjøre følgende:

- Forbered datafila for lesing av analyseprogrammet
- Les inn dataene
- Lag plott av temperatur, lufttrykk og evt. beregnet høyde over havet som funksjon av tiden
- Beregn relativ høyde fra trykkmålinger, om det ikke alt er gjort, og plott høyden som funksjon av tiden
- Bruk MakeCode, Python, Excel, Geogebra eller et annet analyseprogram du er fortrolig med for videre analyse av dataene.

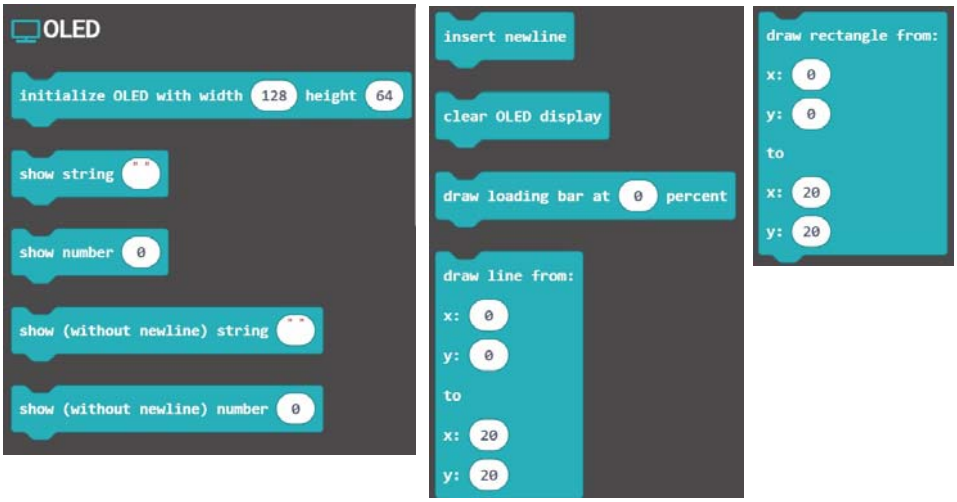
6 Displayer, radioer og aktuatorer

6.1 OLED – Display SSD1306

Kitronik leverer et lite OLED-display montert på et forlengelseskort for Micro:bit. Det grafiske displayet på 128 x 64 piksler, egner seg derfor ypperlig for å kobles midlertidig til micro:bit-en for å feilsøke og evt. vise måleresultater. Displayet henter data via I^2C-bussen.

For å bruke displayet må man installere et bibliotek ved å velge Utvidelser og søke etter displayet. Velg *Lights and Display* og bruk gjerne søkeordet: *Kitronik 128x64 Display* og velg ikonet vist på bildet over til høyre.

Figuren under viser tilbudet av blokker for å skrive til displayet.



Siden displayet trekker noe strøm så kan det være problematisk å bruke displayet når CanSat-en drives kun av 9V batteriet koblet til CanSat-kortet. Årsaken er at spenningsdroppet over Schottky-dioden som skal hindrekonflikt mellom intern spenningskilde (3,3V) og ekstern spenningskilde via USB-kanalen, gir for stort spenningsfall til å drive både Micro:bite-en og OLED-displayet.

Problemet løses ved å hente spenning fra USB-kontakten eller fra et ekstra batteri koblet til micro:bit-en når OLED-displayet skal brukes. I tillegg må 9V batteriet være tilkoblet ved bruk av sensorer, SD-kort terminal eller radio.

6.2 Radiomodemet E32 433T20DC[13]

Det finnes flere varianter av denne radioen både for 433Mhz, 900MHz og 2,4GHz. Likeså med ulik sendereffekt: 20dBm (T20DC, 100mW) eller 30dBm (T30D, 1Watt).

Radioen bruker *spredt spektrum* til forskjell fra APC220 som benytter GFSK[14]. Spredt spektrum er mer robust for støy og vil sannsynligvis kunne skille nabokanaler bedre fra hverandre. Istedet for å legge de enkelte radiosignalene i spesifikke frekvenskanaler som hos GFSK, så spres signalet ut over kanalbåndbredden ved at det "hakkes opp" med en spesiell kode. Når sender og mottaker benytter samme kode for å spre og samle signalene, så finner mottakeren tilbake til det opprinnelige utsendte signalet. "Kanalene" skilles fra hverandre ved at de bruker ulike koder for å spre og samle signalet.

Radioene kan også legges i lett dvale for så å la seg vekke av et signal fra en sender som etterspør data. Dermed kan man spare batteriene. Dette er kanskje ikke så viktig i forbindelse med CanSat som kun skal være operativ over korte perioder.

6.2.1 Grunnleggende spesifikasjoner for E32 433T20DC:

- Frekvensområde: 410 – 441MHz
- Sendereffekt (maks.): 20dBm (ca. 100mW)
- Antennekontakt: SMA-K
- Mottaker følsomhet: –146dBm (ved 0,3kbit datarate)
- Blokkering av inngang: –10dBm, inngangen blokkeres om signalet overstiger -10dBm[15]
- Datarate (luft): 0,3 – 19,2kb/sek, default 2,4kb/sek
- Kommunikasjon: UART
- Power supply: 3,3 – 5,2V
- Rekkevidde i fri sikt: 3km (med antenneforsterkning 5dBi)

13. Kretsenes manualer finnes her: https://www.ebyte.com/en/data-download.html?id=214&pid=211#load
14. Gaussisk frekvens skift nøkling (Gaussian Frequency Shift Keying)
15. Det kan være verdt å merke seg at mottakeren kan blokkeres av sterke signaler fra nabosendere. Det kan bety at man kan miste signalet fra egen sender dersom en kraftig nabosender sender et sterkt signal. Dette kan lett skje i en klasseromssituasjon.

E32 – varianter

Tabellen under viser ulike varianter av E32. Siden vi har valgt å holde oss til 433MHz[16] og en radio som er montert på et kort med antennekontakt (DIP), er imidlertid utvalget begrenset:

Model	Interface	IC	Frequency	Power	Distance	Air Data	Package	Size mm	Feature
E32-900T30D	UART	SX1276	868M 915M	30	8	0.3k~19.2k	DIP	24*43	LoRa spread spectrum
E32-900T20S	UART	SX1276	868M 915M	20	5.5	0.3k~19.2k	SMD	16*26	LoRa spread spectrum
E32-868T30S	UART	SX1276	868M	30	8	0.3k~19.2k	DIP	25*40.3	LoRa spread spectrum Continuous transmiss
E32-433T20S1	UART	SX1278	433M	20	3	0.3k~19.2k	SMD	17*25.5	LoRa spread spectrum
E32-433T20S2T	UART	SX1278	433M	20	3	0.3k~19.2k	SMD	17*30	LoRa spread spectrum
E32-170T30D	UART	SX1278	170M	30	8	0.3k~19.2k	DIP	24*43	LoRa Spread Spectrum
E32-433T30D	UART	SX1278	433M	30	8	0.3k~19.2k	DIP	24*43	LoRa spread spectrum
E32-433T20DC	UART	SX1278	433M	20	3	0.3k~19.2k	DIP	21*36	LoRa spread spectrum
E32-400M30S	SPI	SX1278	410~493 MHz	30	10	0.3k~19.2k	SMD	24*38.5	LoRa spread spectrum
E32-400T20S	UART	SX1278	410~525 MHz	10~20	3	0.3k~19.2kbps	SMD	16 * 26	LoRa spread spectrum
E32-900M30S	SPI	SX1276	850~930 MHz	30	10	0.3k~19.2kbps	SMD	24*38.5	LoRa spread spectrum

Vi legger merke til at vi har valget mellom en radio på maks. sendereffekt på 100mW (T20DC) og en på 1000mW (T30D) og begge har UART (Rx/Tx) grensesnitt. Det reglementet tillater imidlertid maks 100mW (20 dBm)[17].

Operasjonsmodi

Radioen kan opereres på fire ulike måter (modi). Disse fire modiene velges ved å sette verdiene på to digitale styreinnganger, M0 og M1.

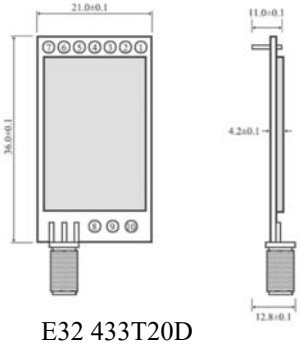
1. (M0 = 0, M1 = 0) Transparent sendermodus, som brukes ved vanlig dataoverføring
2. (M0 = 1, M1 = 0) Laveffekts sender-oppvekkingsmodus
3. (M0 = 0, M1 = 1) Laveffekts mottaker-oppvekkingsmodus
4. (M0 = 1, M1 = 1) Dyp dvale modus. Denne brukes også for å sette parametere hos radioen

16. Dette er et såkalt ISM-bånd for bruk til Industrielle, Vitenskapelige og Medisinske formål, og er i Europa avgrenset til 433,075 – 434,750 MHz i 69 kanaler á 25kHz. Vi bruker imidlertid frekvensområdet på en annen måte (spredt spektrum), slik at i praksis har vi kun mulighet til å bruke kanal 23 (410 + 23 MHz)
17. https://en.wikipedia.org/wiki/LPD433

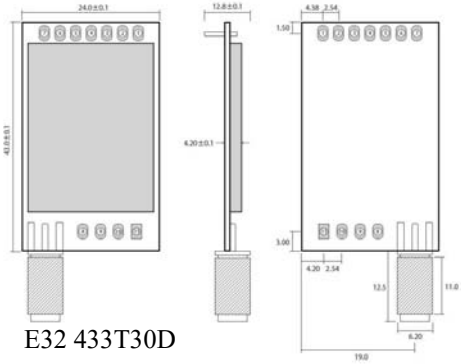
Dersom det ikke er noe stort poeng å spare energi, er transparent modus mest aktuell under normal drift.

Mål

Radioen er koblet opp på et lite kretskort med antennekontakt og har følgende mål: E32 433T20D er 21 x 36 mm pluss antennepluggen, hvilket ikke er så mye større enn for APC220. E32 433T30D er 24 x 43 mm pluss antennepluggen, altså 3 mm bredere og 7 mm lengre. Pinningen er lik hos de to:



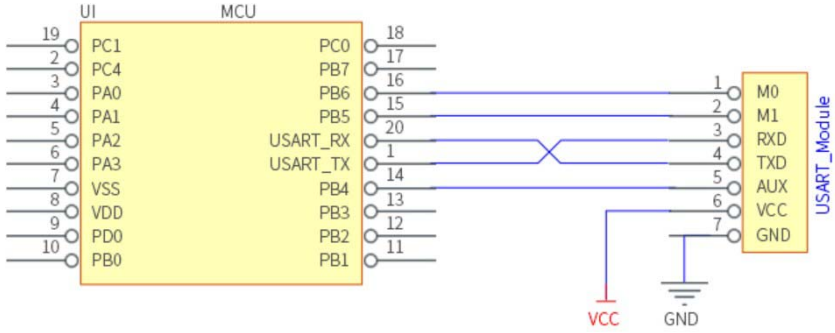
E32 433T20D
100mW (20dBm)



E32 433T30D
1000mW (30dBm)

Tilkoblinger

Kretsen har en rekke tilkoblinger. Ikke alle trenger å brukes eller styres av mikrokontrolleren.



Vi legger spesielt merke til at RXD og TXD er krysset inne på kretskortet mellom chipen (MCU) og kontakten. Når man kobler radiomodemet til USB-porten via FDMI adapteren, kobles RXD (E32) til TXO (FTDI) og TXD (E32) til TXI (FTDI). Tilsvarende vil gjelde for tilkobling til Micro:bit.

Tabellen under spesifiserer tilkoblingspunktene:

No.	Name	Direction	Function
1	M0	Input (weak pull-up)	Work with M1 to decide 4 working modes of module (not suspended, if not used, could be grounded).
2	M1	Input (weak pull-up)	Work with M0 to decide 4 working modes of module (not suspended, if not used, could be grounded).
3	RXD	Input	TTL UART inputs, connects to external (MCU, PC) TXD output pin. Can be configured as open-drain or pull-up input.
4	TXD	Output	TTL UART outputs, connects to external RXD (MCU, PC) input pin. Can be configured as open-drain or push-pull output
5	AUX	Output	To indicate module 's working status & wakes up the external MCU. During the procedure of self-check initialization, the pin outputs low level. Can be configured as push-pull output (suspending is allowed).
6	VCC	Input	Power supply ： 2.3~ 5.2V DC
7	GND	Input	Ground
8	Fixed orifice	-	Fixed orifice
9	Fixed orifice	-	Fixed orifice
10	Fixed orifice	-	Fixed orifice

Med dette i minne burde det være relativt enkelt å koble kretsen til Micro:bit-en via dens Rx/Tx (UART) port (se Vedlegg B).

6.2.2 Oppsett av radiomodemet E32[18]

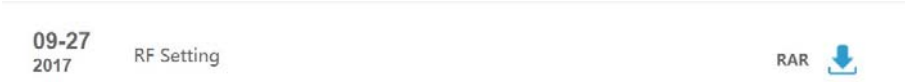
Dette kan vi gjøre på flere måter bl.a. i selve blokkoden og ved hjelp av programmet: *RF setting*.

Bruk av programmet: "RF Setting" for oppsett av radiomodemet

Vi henter programmet for oppsett av modemet fra leverandøren[19]. Gå til siden:

https://www.ebyte.com/en/data-download.html?page=3&id=199&cid=31#load

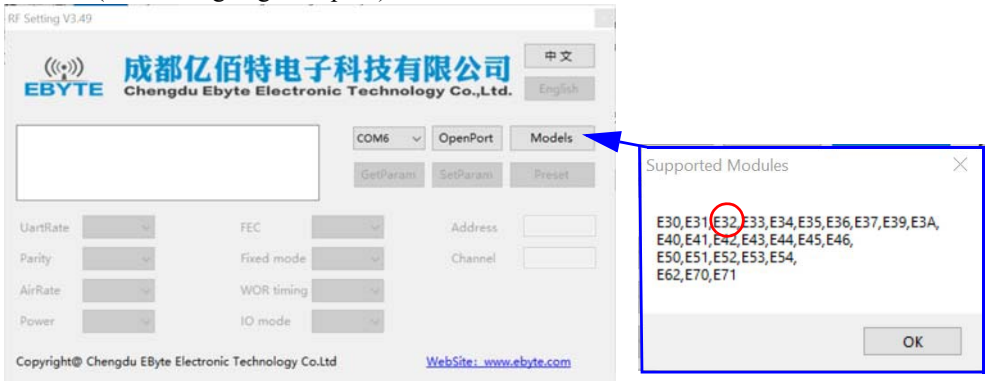
og bla ned til ark 3:.



Last ned, pakk ut og kjør programmet.

18.Detter er en nyttig nettside: https://moisen.eu/getting-started-with-lorawan-sx1278/
19.https://www.ebyte.com/en/pdf-down.aspx?id=199

Før vi kobler radiomodemet til USB-porten, vil vi få opp et brukergrensesnitt omtrent som på bildet under (husk å velg engelsk språk):



Trykker vi på *Models* så får vi opp informasjon som forteller oss at vi kan bruke dette programmet til å programmere E32 (se figur over til høyre). Ellers er det lite som kommer opp i de enkelte rubrikkene før vi kobler radiomodemet til USB-porten.

For å kunne sette opp modemet må M0 = 1 og M1 = 1 i henhold til tabellen under:

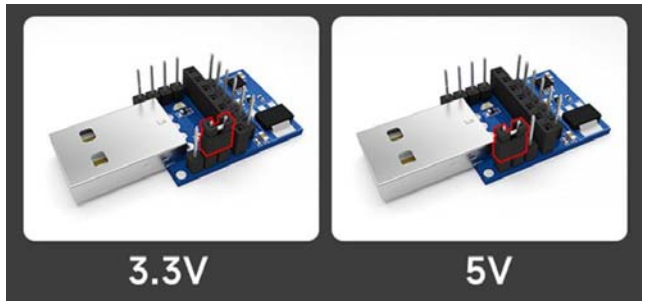
Mode(0-3)	M1	M0	Mode introduction	Remark
Mode 0 Normal	0	0	UART and wireless channel is open, transparent transmission is on.	The receiver must works in mode 0 or mode 1
Mode 1 Wake-up	0	1	UART and wireless channel is opened. The difference between normal mode and wake-up mode is it will add preamble code automatically before data packet transmission so that it can awaken the receiver works in mode 2.	The receiver can works in mode 0, mode 1 or mode 2.
Mode 2 Power-saving	1	0	UART is disabled. Wireless module works at WOR mode (wake on radio). It will open the UART and transmit data after receive the wireless data.	1,the transmitter must works in mode 1 2,transmitting is not allowed in this mode
Mode 3 Sleep	1	1	Parameter setting.	

Tilkoblingen kan gjøres ved hjelp av en FTDI[20] modul.

20. *Future Technology Devices International Limited* - Et skotsk firma som har spesialisert seg på overganger mellom ulike serie kommunikasjonsbusser som her mellom en USB og en UART - *Universal Asynchronous Receiver-Transmitter*

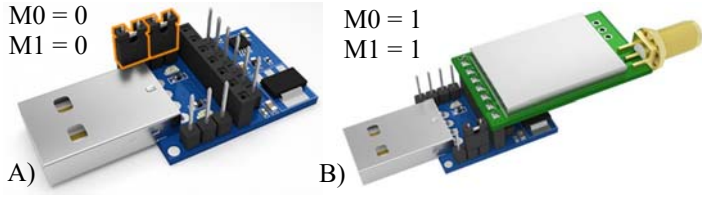
Oppkobling til PC'ens USB for konfigurering av radiomodemet

En god løsning er å anskaffe en FTDI som er skreddersydd for radioen E32. Denne kan kjøpes hos Aliexpress[21].



Ved hjelp av strapper kan man sette opp FTDI-en for å konfigurere radioen eller oppføre seg transparent som sender og mottaker. Figuren under til venstre viser oppsett for transparent oppførsel og til høyre for å konfigurere radioen. Ved å fjerne de to strappene i bakkant, settes M0 = 1 og M1 = 1 og radioen kan konfigureres.

I transparent modus settes de to strappene inn slik at M0 = 0 og M1 = 0. Legg merke til at her har vi ikke valgt spenning (A). Bildet til høyre (B) viser E32 433T20D plugget inn.

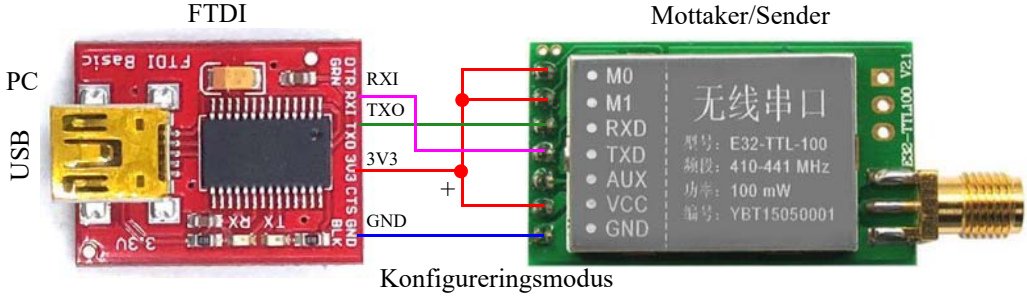


Transparent modus | Konfigureringsmodus

På denne måten kan vi sette parametrene for både mottaker- og senderenheten.

Bruk av generell FTDI

En alternativ oppkobling av radiomodemets Rx/Tx (UART) til PC-en for konfigurering, er beskrevet på Instructable[22]. Her kobles radiomodemet til PC'en via en generell FTDI (overgang mellom USB og RS232 (UART)) som vist på figuren under. Legg merke til at M0 = M1 = +3,3V (logisk høy), som er et oppsett for konfigurering av radioen i følge tabellen over.



Konfigureringsmodus

21.https://www.aliexpress.com/item/32800165617.html?spm=a2g0o.order_list.order_list_main.4.5ed91802zjLUsH
22.https://www.instructables.com/E32-433T-LoRa-Module-Tutorial-DIY-Breakout-Board-f/

Set parametere hos radio E32, sender og mottaker

Når radiomodemet er koblet opp kan vi koble FTDI-kretsen til USB inngangen på PC'en og velge riktig port. Vi får da opp et skjermbilde hos "RF Setting"-programmet som kan ligne på følgende:

Vi kobler til og velger først **OpenPort**, deretter velger vi riktig port og velger **GetParam** og radioens nåværende parametere vises.

I dette eksempelet velger vi å sette adressen til 85 og kanalen til 20 som kan være et greit sted å eksperimentere med ulike parametere. Vi velger også å sette effekten, Power, til rett nivå i vårt tilfelle 20 eller 21 dBm (legg merke til at her setter vi effekten direkte i dBm og ikke som 0,1,2 eller 3).

De nye parametrene overføres til radiomodemet ved å velge **SetParam.** Før vi kobler fra kretsen valger vi **ClosePort**.

Sett opp både sender og mottaker på samme måte.

Bruk av biblioteket: "E32LORA"

Her skal vi ganske kort beskrive bruken av biblioteket E32LORA som er en lett modifisert versjon av Alexander Frolovs utgave av biblioteket[23].

Biblioteket kan hentes fra github:

https://github.com/SCjoh/pxt-lora

Biblioteket er dokumentert i brukermanualen som kan lastes ned fra:

https://www.makerhero.com/img/files/download/E32_User+Manual_EN_v1.00.pdf

23.Modifikasjonen er gjort av Jørn Hafver

:

I blokken: *E32LORA pin config:* bestemmer vi hvilke pinner vi velger for å kommunisere med Micro:bit-en. Det er hensiktsmessig å velge P14 som TX og P15 som RX. De øvrige kan velges etter ønske, men for oss er de gitt av utlegget på kortet som angitt på figuren til venstre.

BAUD: Datahastigheten er hastigheten fra Micro:bit til radiomodemet og ikke på lufta.

AUX er en utgang som kan brukes til å registrere modemets status ved oppvåkning.

CONFIGURATION MODE: Når denne er sann så settes modemet i Parameter setting modus (M0 = 1, M1 = 1).

I blokken: *Set E32LORA module configurartion:* Settes ulike parametere som er vesentlig for overføring av informasjon:

ADDR: Adressen kan være et tall mellom 0 og 65535

CHANNEL: Her settes kanalnummeret. Dette er frekvensen mellom 410 og 441, hvor kanalnummeret (Ch.) angir frekvensen i MHz lik 410 + Ch. Ch. nr. 23 gir derfor 433MHz.

UART BAUD: Er datahastighet mellom Micro:bit og radiomodemet. Anbefalt hastighet er 9600 baud.

AIR BAUD: Er datahastigheten på lufta, anbefalt 2,4 kbaud

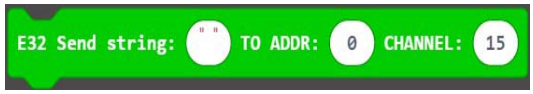
POWER: Angir utsendt effekt og kan settes til 3, 2, 1 eller 0. Der 0 angir maks effekt lik 20 dBm (ca. 100mW).
SAVE CONFIG: SANN angir at konfigurasjonen skal lagres.

SET SETUP MODE: Her kan en aktivt sette radiomodemet i setup modus, dvs. M0 = 1, M2 = 1.

SET NORMAL MODE: Her kan en aktivt sette radiomodemet i normal transparent modus, dvs. M0 = 0, M2 = 0.

E32 SEND STRING: (“ “): Her kan man sende en tekst streng av karakterer, f.eks. komma separerte måleverdier.

ON E32RADIO RECEIVED (*receivedString*): Når radiomodemet mottar en tekststreng av data, vil kommandoene i gapet utføres. Variabelen *receivedString* vil inneholde den overførte tekststrengen.



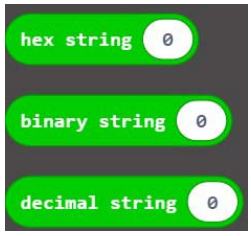
E32 SEND STRING: () TO ADDR: () CHANNEL: ( ). Kommandoblokken sender en streng til et bestemt adresse og kanal. Det anbefales å bruke denne for sending av tekststrenger.

AUX PIN: Variabelen angir verdien på utgangen AUX.

E32VERSION: Variabelen inneholder versjonsnummeret til radiomodemet.

E32RESET: Resetter radiomodemet.

E32PARAMETERS: Variabelen inneholder de aktuelle parametrene som modemet er satt opp med ved avlesningen.



HEX STRING (): Variabelen angir argumentet som en heksadesimal streng.

BINARY STRING (): Variabelen angir argumentet som en binær streng.

DESIMAL STRING (): Variabelen angir argumentet som en desimal streng.

7 Sensorer

I dette avsnittet skal vi se på et knippe sensorer som enten kan brukes uten biblioteker eller sensorer som det finnes biblioteker til for Micro:bit. Vi ønsker i utgangspunktet å se på individuelle sensorer og ikke pluggin-kort i tillegg til at vi ønsker å bruke blokkode. Dessuten må vi ikke glemme at Micro:bit har noen innebygde sensorer som kan brukes uten ekstra biblioteker. Micro:bit har akselerometer (x, y, z), magnetometer (x, y, z), en enkel lyssensor og temperatursensor.

7.1 Temperatur

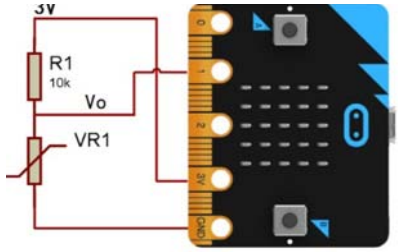
Her har vi flere mulige alternativer:

7.1.1 NTC-motstand

NTC-motstand er en motstand hvor motstandsverdien avtar med økende temperatur.

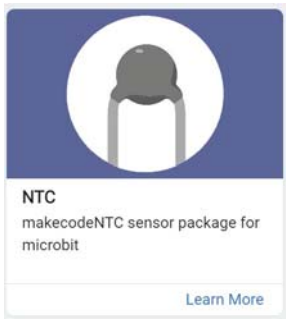
NTC-motstander er laget av et materiale hvis resistivitet varierer sterkt med temperaturen. Som navnet sier (Negative Temperature Coefficient – NTC) så avtar resistansen med økende temperatur.

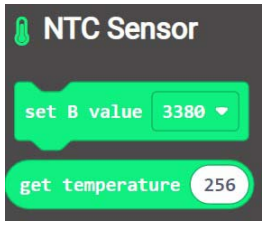
NTC-motstander er derfor vanligvis bygget opp som en polykrystalinsk *halvleder* som kan bestå av en blanding av krom, mangan, jern, kobolt og nikkel, som sintres[24] sammen med et plastisk bindemiddel.



Endringen i motstandsverdi kan lett konverteres til en endring i spenning ved å benytte en spenningsdeler. Sammenhengen mellom temperatur og motstandsverdi er svært ulineær, men spenningen ut av spenningsdeleren vil bli ganske lineær i et avgrenset temperaturområde.

Figuren til over til høyre viser et typisk oppkobling av sensoren. Vi antar at NTC-motstanden har en verdi lik 10kΩ (ved 20°C), det samme som seriemotstanden, R1. Dette gir en ganske lineær sammenheng mellom spenning og temperatur i området rundt 20°C.





Det finnes et standard bibliotek for bruk av NTC-motstander som temperaturmåling. Ved bruk av biblioteket[25] må vi legge inn B-verdien for den valgte NTC-motstanden, som normalt er oppgitt i databladet. Dernest mater vi biblioteksfunksjonen med avlest verdi fra

24. Sintring betyr at metallpulver knyttes sammen ved hjelp av oppvarming, men uten å smelte.
25. https://github.com/MakeCode-extensions/NTC

den analoge inngangen, som så returnerer temperatur i °C. Se figuren over til høyre.

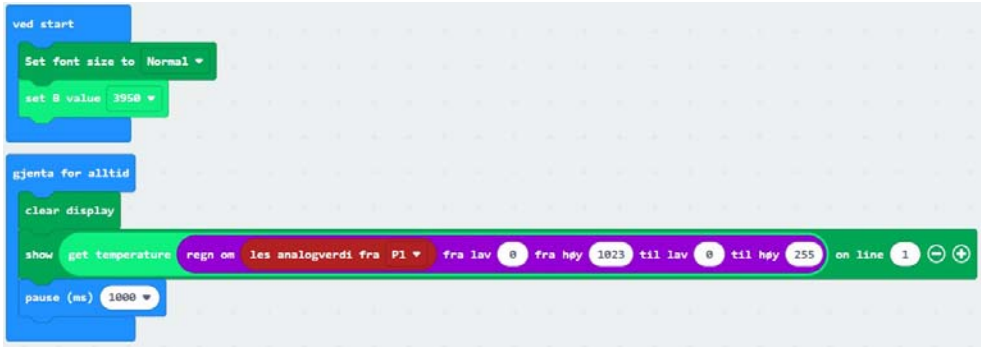
Biblioteket for NTC-motstanden har noen begrensninger:

- *Kommandoene gir ingen mulighet til å legge inn egne verdier for komponentkonstanten B, det er kun mulig å velge en av to verdier B = 3380 eller B = 3950. Det angis heller ingen konkret NTC-motstand i dokumentasjonen til biblioteket. En er derfor tvunget til å velge den nærmeste verdien for sin valgte NTC-motstand.*

- *Verdien til referansemotstanden for NTC-motstanden er også gitt lik 10kΩ. Det er dermed ikke mulig å velge alternative verdier.*

- *Det er også noe uklart hva som er sensorens dynamiske område. Det antydes at avleste verdier fra den analoge inngangen er fra 0 – 255, hvilket betyr at vi må skrumpe området for Micro:bit-ens innebygde AD-konverter som har et dynamisk område på 0 – 1023.*

Vi har derfor endt opp med følgende forslag til avlesning av sensoren og skriving av resultatet til Kitronik OLED displayet. I dette eksempelet er P1 valgt som analog inngang og oppkoblingen er i henhold til figuren over (på CanSat-kortet er NTC-sensoren koblet til port P10).



Linearisering og bruk av topunktsligning

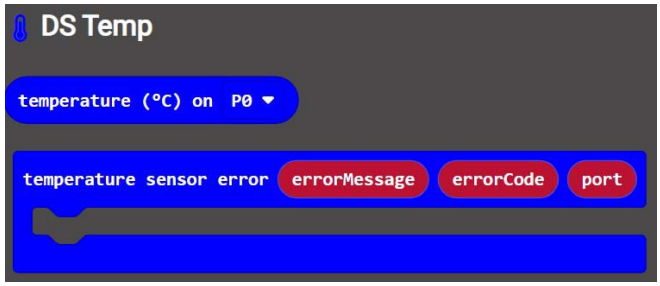
Siden spenningen ut av spenningsdeleren er ganske lineær som funksjon av temperaturen, kan man gjøre måliner ved to temperaturer og sette opp en topunktsligning for å beregne mellomliggende verdier. Dette er en fin oppgave som viser nytten av matematikk.

Se vedlegg for detaljer.

7.1.2 Temperatursensoren DS18B20

Dette er en mye brukt temperatursensor fra Dallas Semiconductor. Sensoren leveres både utformet som en transistor (TO92) eller som vanntett utgave om det er ønskelig. Sensoren krever en pullup-motstand på 4,7 kΩ mellom signalledningen (gul) og ledingen som tilfører spenning (rød). I tillegg kobles den sorte ledningen til jord.

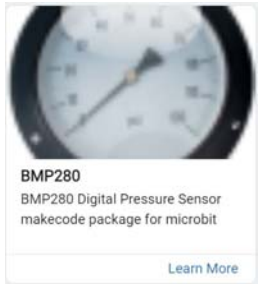
Biblioteket inneholder kun to blokker. Med den ene avleses temperaturen og med den andre kan man vise evt. feilmeldinger. I eksempelet er signalledningen (gul) koblet til P0.



7.2 Lufttrykk

Det finnes mange gode sensorer som måler lufttrykk. I denne sammenhengen velger vi og se på BMP280 og BME280 fra Bosch.

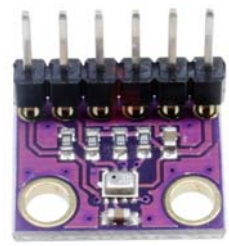
7.2.1 BMP280

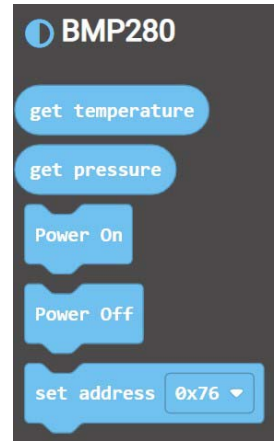


Dette er en svært populær sensor for måling av lufttrykk. Ved omregning til høyde er også temperaturen en viktig parameter, sensoren er derfor også utstyrt med en temperatursensor. Sensoren overfører målingene via I^2C-bussen.

Sensoren har et fåtall kommandoblokker, se figuren under til høyre. Foruten henting av lufttrykk og temperatur så gir den mulighet til å slå av spenningen på sensoren for å spare strøm. Den gir også mulighet til å velge mellom to adresser på bussen (0x76 og 0x77). Vanligvis benytter BMP280 adressen 0x76.

Kretsen leveres normalt som et breakout kort som vist til venstre på figuren under.





Temperaturen leveres i °C og lufttrykket leveres i Pa.

Måleområdet er fra 300 – 1100 hPa, som er det samme som mBar. Nøyaktigheten regnes å være 1 hPa, dvs. komponenten er kalibrert på fabrikken.

Kretsens arbeidsspenning er 3,3V.

7.2.2 BME280

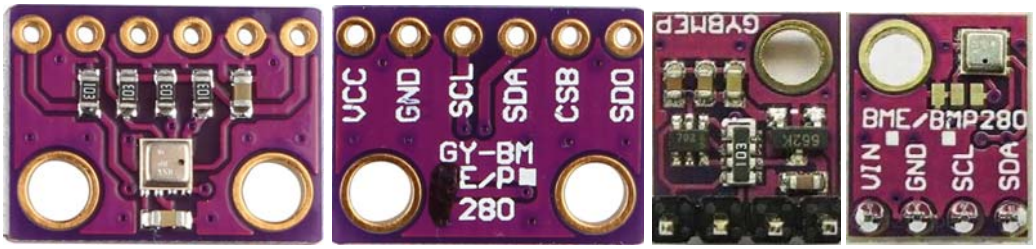
BME280 er en utvidelse av BMP280 hvor man har inkludert fuktighetsmåling i tillegg til lufttrykk og temperatur. Dette kvalifiserer tydeligvis for å bytte ut P (Pressure) med E (Environment). Hvilket gir løfter om en sensor som i større grad en BMP-serien, gir er mer fullstendig "bilde" av miljøet. Sensoren er spesielt beregnet for mobile anvendelser med sitt lave strømforbruk. Her er noen nøkkeldata for BME280[26]:

- *Spenning og energiforbruk:*
  Supplyspenning: 1,7 – 3,6 V
  Strømforbruk standby: 0,1 µA
  Strømforbruk med måling 1 x pr. sek.: 3,6 µA (H, P, T)
- *Grensesnitt:*
  I^2C eller SPI – I^2C adresse: 0x76
- *Luftfuktighetssensor:*
  Responstid: 1 sek.
  Nøyaktighet: ± 3% mellom 20 – 80% relativ fuktighet.
- *Lufttrykksensor:*
  Måleområde trykksensor: 300 – 1100 hPa
  Relativ nøyaktighet: ± 0,12 hPa
  Absolutt nøyaktighet: ± 1 hPa (0 – 65°C)
  Typisk responstid: 5,5 msek. (Typisk maks. samlingsrate: 182 Hz (garantert 157 Hz))
- *Temperatursensor:*
  Måleområde: – 40 – +65°C
  Nøyaktighet: ± 1°C (0 – 65°C)

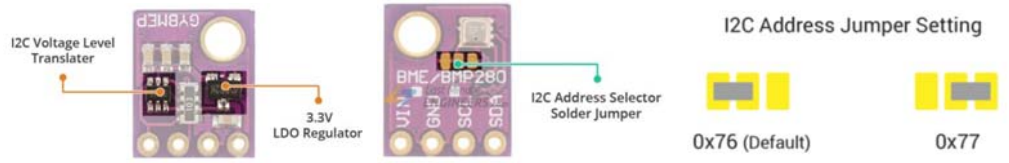
26. Informasjonen er hentet fra: https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bme280-ds002.pdf

Oppkobling

Figuren under viser pinningen til to utgaver av BME280:

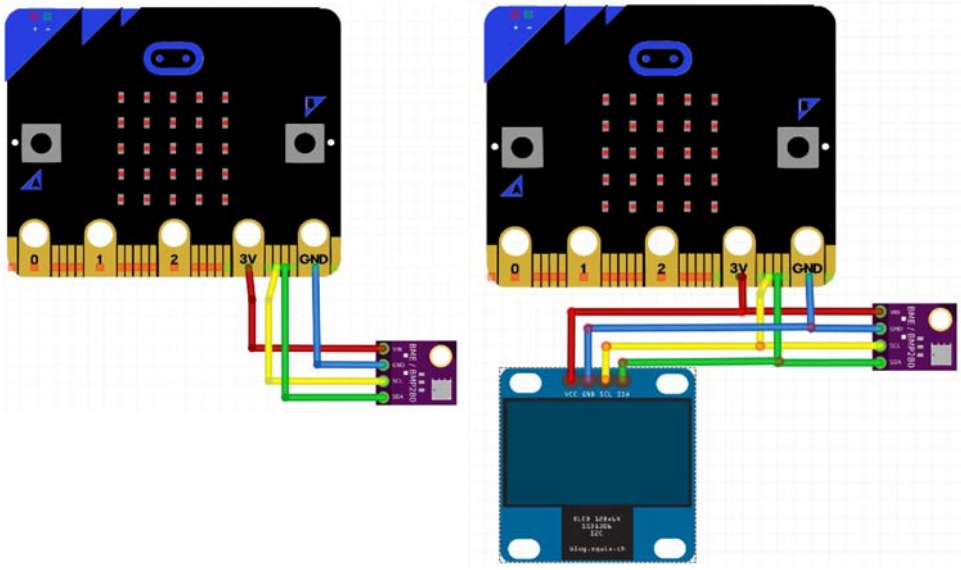


En viktig forskjell er at hos varianten til høyre kan man sette to ulike adresser i "adressefeltet". Default adresse er 0x76 HEX, men den kan endres til 0x77 HEX om man forbinder to pad'er som vist helt til høyre på figuren under. Dessuten har denne varianten en innebygget spenningsregulator slik at den kan arbeide på spenninger fra 3,3 – 5,0 V.



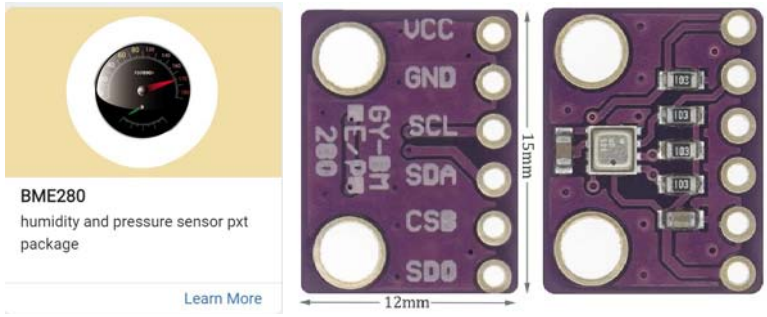
Spenningsregulator Adresse-setting

Siden BME280 normalt har en arbeidsspenning på 3,3V og kommuniserer med Micro:bit-en over I^2C-bussen, så kan vi koble oss direkte til kantkontakten, riktignok via en connector som ikke er vist på figuren under.



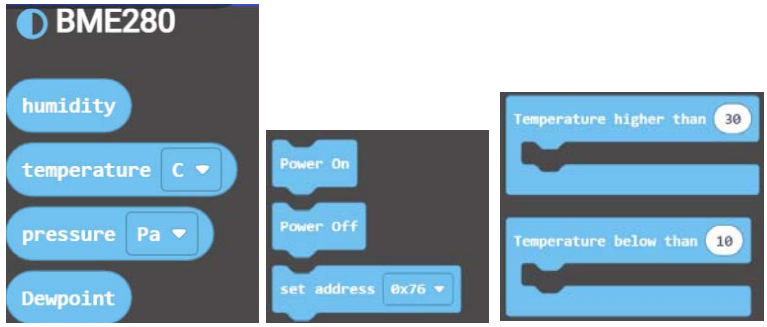
7.3 Luftfuktighet og dogpunkt – BME280

BME280 måler i tillegg til lufttrykk og temperatur også luftfuktighet.



BME280 kommuniserer på I^{2}C-bussen og biblioteket disponerer de to adressene 0x76 og 0x77. Normalt benyttes adressen 0x76.

I tillegg til å lese av temperatur, lufttrykk og luftfuktighet, så beregner biblioteket duggpunktet. Duggpunktet er den temperaturen som kreves for at fuktigheten i lufta skal begynne å felles ut som vann med den absolutte mengden fuktighet som finnes i lufta når målingen gjøres. I tillegg så kan man sette opp varslinger dersom en av parameterne kommer over eller under et gitt nivå. Dette er vist for temperatur på figuren under. Tilsvarende grenseverdier kan settes opp for de andre parameterne.



Dersom det anbefalte Micro:bit biblioteket for BME280 ikke fungerer, kan man bruke følgende bibliotek[27]:

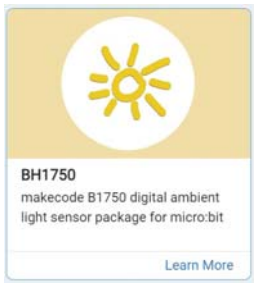
https://github.com/ElectronicCats/pxt-bme280

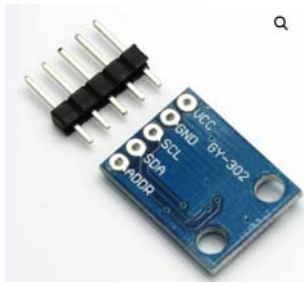
Man velger utvidelser som vanlig, men istedet for å søke i de anbefalte bibliotekene, skriver man lenken til biblioteket inn i søkefeltet og trykker return.

27. Oppdaget av Simen Bergvik ved Andøya Space Education

7.4 Lysintensitet – BH1750

Denne sensoren måler belysningsstyrke i Lux. Lux er lumen pr. kvadratmeter og forteller hvor opplyst en flate er, f.eks. et arbeidsområde. Kretsen BH1750 har en arbeidsspenning på fra 3,0 – 5,0V og typisk strømforbruk på 0,12 mA. Måleområde er fra 0 – 65535 Lux, med en måleoppløsning 1 Lux. Kretsen leverer måleresultatene via en I^2C-buss. Ved å legge ADDR-pinnen til GND har den adressen 0x23 (35), legges den til Vcc blir adressen 0x5C (92). Dvs. at vi i praksis kan koble to slike lyssensorer til bussen, med hver sin adresse.





Data hentes ut ved hjelp av blokkommandoen "*get intensity (lx)*". Dessuten kan man velge I^2C-adresse i tillegg til at man kan slå av og på kretsen som vist på figuren til høyre.

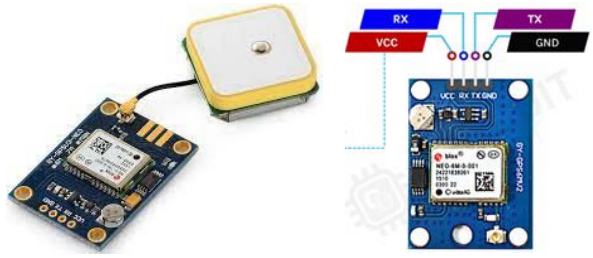
7.5 GPS-mottaker NEO-6M

En åpenbar nytte ved å utstyre CanSat-en med en GPS-mottaker, er at det gjør det lettere å gjenfinne den etter landing. I tillegg kan GPS-mottakeren supplere trykkmåleren i beregning av absolutt høyde.

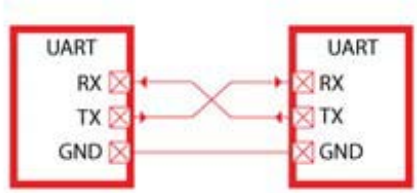
NEO-6M er en GPS-modul som kan følge opp til 22 satellitter over 50 kanaler, og måle posisjonen til mottakeren med en horisontal nøyaktighet på inntil 2.5 meter. Den er laget for en spenningsforsyning på 2,7 – 3,6V, men har en egen spenningsregulator som gjør at den tåler 5V spenningsforsyning. Enheten kommuniserer med Micro:bit-en gjennom UART[28] – den samme typen kommunikasjon som Micro:bit-en bruker til å sende data til OpenLog og radioen (E32). Presisjonen til enheten kan forbedres ytterligere med å bruke en aktiv GPS antenne med U.FL[29] kontakt.

28. UART - Universal Asynchronous Receiver-Transmitter
29. U.FL er en typebetegnelse på en miniatyr antennekontakt framstilt av firmaet Hirose, uten noen åpenbar betydning

Enheten kan oppdatere posisjonen sin opp til fem ganger i sekundet og trekker typisk 45 mA under vanlig drift. Den kan også settes i strømsparingsmodus som reduserer styrømtrekket til 11 mA.



Tilkobling med UART skjer gjennom en sendekanal, Tx, og en mottakerkanal, Rx. Det er viktig å legge merke til at når to UART-enheter snakker sammen så er senderen til den ene enheten koblet til mottakeren på den andre. (Se figuren under)



Nøkkeldata for NEO-6M

Tabellen under gir noen nøkkelparametere for GPS-mottakeren:

Power supply	5.0 V
Operating Current	45mA
Operating Temperature	-40°C ~ 85°C
Time-To-First-Fix: Cold Warm Start Hot Start Aided Starts	 27 s 27 s 1 s <3 s
Receiver sensitivity: Tracking & Navigation Reacquisition Cold Start (without aiding) Hot Start	 -161 dBm -160 dBm -147 dBm -156 dBm

Maximum Navigation update rate	1 Hz
Horizontal Position Accuracy	2,5 m

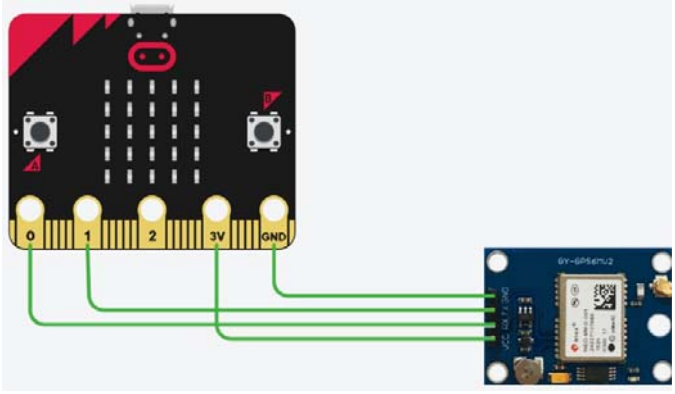
Oppstartstid

NEO-6M har et lite batteri som lades hver gang kortet får strøm. Batteriet brukes til å lagre data til BBR (Battery Backed RAM). Dersom batteriet er oppladet og det er data i BBR, så kan GPS-en startes med en såkalt "hot start". Denne oppstartsprosedyren tar ca. 1 sekund. Dersom det ikke er lagra noe data i BBR så trenger GPS-en en "cold start". Dette er en prosedyre som kan ta opptil 27 sekunder. Det er derfor viktig å ta hensyn til dette når man skriver koden, spesielt med tanke på oppskytning/droneslipp slik at man unngår tap av data ved oppstart.

På NEO-6M-kortet er det en indikator (LED) som brukes til å indikere om GPS-en har klart å koble seg til nok satellitter til at den kan beregne posisjonen sin. Denne skal blinke én gang i sekundet når GPS-en har kontakt med nok satellitter.

Oppkobling og programmering

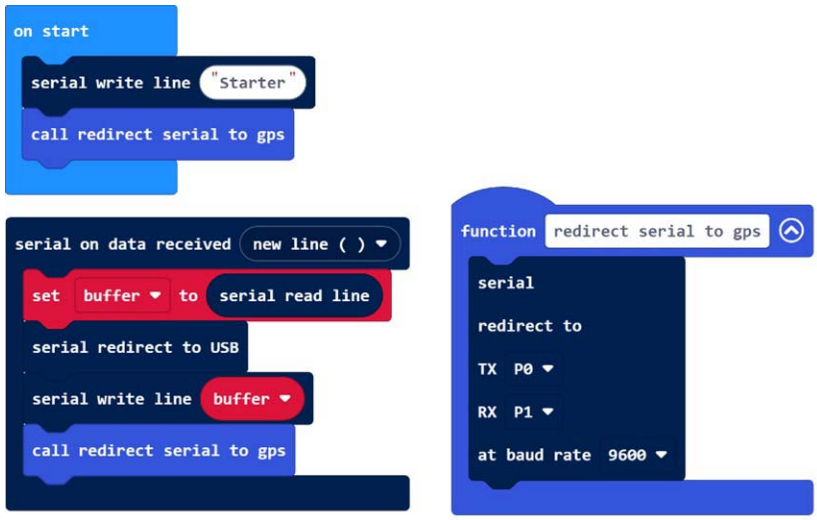
Når du kobler GPS-mottakeren til Micro:bit-en må du velge to porter til kommunikasjonen. Dette har blitt testet med P0 og P1, som vist i eksempelet under, men bør i prinsippet fungere like godt med to andre av portene som du kan rute seriekommunikasjonen til[30]. Du kan bruke prototype-området eller koblingsbrettet på CanSat-kortet til å koble opp GPS-modulen. Det enkleste da er å koble Vcc og GND til J4-kontakten på kortet.



Under er det vist et eksempel på et program som tar imot data fra GPS-modulen og sender dem videre som seriedata til MakeCode. Siden dataene fra GPS-en også er seriedata trenger vi å bytte mellom hvilken kilde seriedataene skal til og fra. Fra Micro:bit-en til PC-en brukes USB, men fra

30. Portene som er tilgjengelige for dette på micro:bit-en er P0, P1, P2, P8, P12, P13, P14, P15 og P16, men vær oppmerksom på at om du bruker P14 og P15 kan du forstyrre kommunikasjonen mellom micro:bit-en og E32-modulen og/eller openLog-modulen (se seksjon 8.6.3).

GPS-en til Micro:bit-en er det Tx = P0 og Rx = P1. Hvordan du kan programmere dette er vist i eksempelet under. For å gjøre koden litt mer kompakt er seriekonfigurasjonen for GPS-en flyttet til en egen funksjon, men dette er ikke strengt tatt nødvendig.



Merk at standard overføringsfart (baud rate) for NEO-6M er 9600 baud[31]. Legg også merke til at det er mottakerporten, Rx, på GPS-modulen som er koblet til P0, men for Micro:bit-en er dette en sendeport, Tx, og tilsvarende: Tx, på NEO-6M er koblet til P1 som defineres som Rx-pin for Micro:bit-en.

Dataene du mottar kan se noe kryptiske ut til å begynne med, men vi skal gå gjennom hva dataene betyr og hvordan du kan konfigurere GPS-modulen til å bare sende de dataene du er interessert i å motta.

NMEA-setninger

Organisasjonen *National Marine Electronics Association*, NMEA, har laget en standard for hvordan et GPS-signal ser ut. Hver linje med data GPS-en sender ut er en såkalt *NMEA-setning*. En slik setning kan for eksempel se slik ut:

$GPRMC,113537.000,A,5839.33664,N,00537.88488,E,0.000,194.78,020523,,,A*41

Tabellen gir en oversikt over hva de ulike delene av setningen betyr:

$	Markerer starten på en ny setning
GPRMC	Global Positioning Recommended Minimum Coordinates – viser hvilken type NMEA-setning det er.
113537.000	Tid i UTC – 12:35:37
A	Status A=active eller V=Void

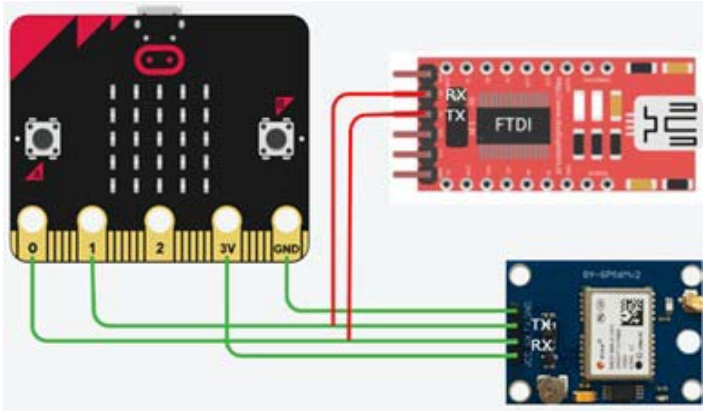
31. Baud - Tegn per sekund, eller hos oss Bit per sekund.

5839.33664,N	Breddegrad 58° 39.33664’ Nord
00537.88488,E	Lengdegrad 5° 37.88488’ Øst
0.000	Bakkefart i knop
194.78	Bevegelsesretning i antall grader fra nord
020523	Dato –2. mai, 2023
[tomme felt]	Disse brukes til magnetisk deklinasjon
A	Modus – A = autonom posisjonering
*41	Checksum. Begynner alltid med en *

Dette setningsformatet, GPRMC, er et format som ikke inneholder så mye annen nyttig informasjon enn koordinater og tid. Kobler du til GPS-en som beskrevet over så mottar du antakelig denne typen setning, eller i alle fall en del av denne typen setning. Ønsker du f.eks. å bruke høydedata trenger vi en setningstype som heter GPGGA. Den enkleste måten å konfigurere NEO-6M for kun å motta denne typen setninger, er å benytte en FTDI og gratisprogrammet: *u-center*.[32]

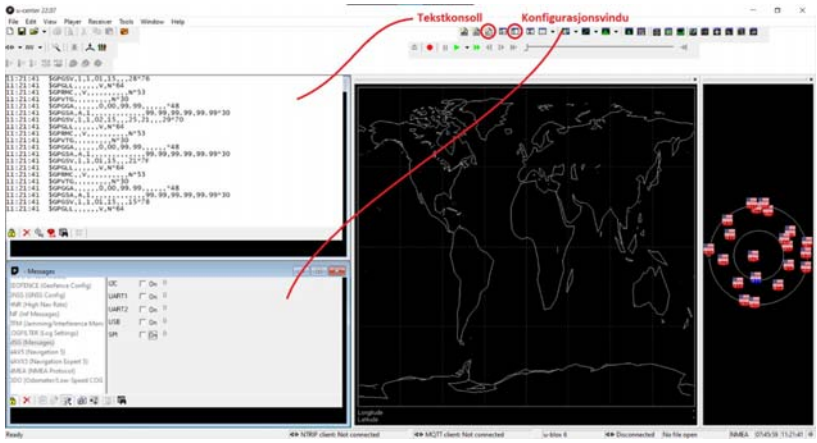
FTDI er beskrevet i vedlegg E.1 Kort fortalt er det en enhet som gjør det mulig å koble UART-enheter direkte til USB-porten til en PC.

Vi plasserer GPS-mottakeren NEO-6M på koblingsbrettet og kobler den opp til Micro:bit-en med jumpere som vist på figuren under. Dersom Micro:bit-en er tilkoblet PC-en via USB-pluggen, får GPS-mottakeren strøm fra Micro:bit-en. Vi monterer også FTDI-en på koblingsbrettet og forbinder Rx- og Tx-terminalene hos FTDI-en til Tx- og Rx-terminalene til NEO-6M, som vist på figuren. FTDI-en kobles så til en USB-kontakt hos PC-en. FTDI-en får dermed strøm og forbindes til felles jord via PC-en[33].



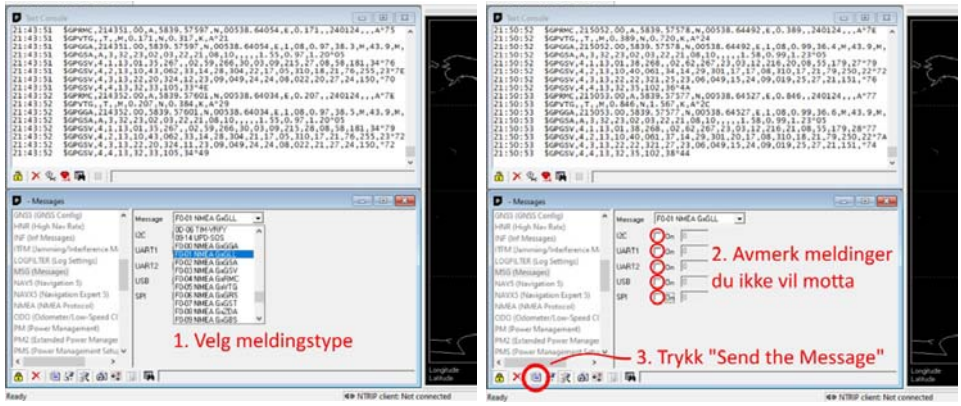
32. https://www.u-blox.com/en/product/u-center
33. Merk at dersom micro:bit-en og FTDI-en ikke er koblet til samme PC, må man koble sammen GND-pinnene på FTDI-en og GPS-modulen med en jumper. I teksten videre antar vi at micro:bit og FTDI begge er koblet til den samme PC-en.

Installer og åpne u-center. Når vi starter programmet kommer det sannsynligvis opp flere vinduer, men ikke alltid de du trenger. Åpne tekstkonsollen og konfigurasjonsvinduet.



I tekstkonsollen bør vi kunne se at det kommer inn linjer med data. Hvis det ikke gjør det kan det være at tilkoblingen må startes på nytt. På menylinjen øverst, velg Reciever – Connection – COMx, hvor x er porten på PC-en du har koblet FTDI-en til[34].

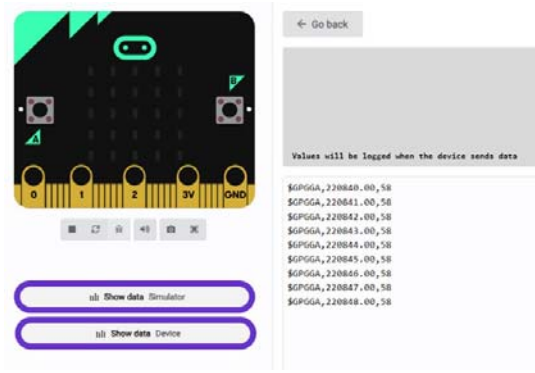
I konfigurasjonsvinduet blar vi ned til *MSG (Messages)*. Her kan vi velge mellom ulike *meldingstyper* som GPS-en kan tilby. (1) Bla ned til F0-01 NEMA GxGLL. Dette er den første meldingstypen du mottar. For hver meldingstype kan vi velge mellom fire ulike *tilkoblingstyper* (I^2C, UART, USB eller SPI) dit meldingen skal sendes (2). Vi velger UART1. Det er enklest å stille hver melding av eller på for alle kanaler samtidig. Når vi har avmerket en meldingstype vi ikke vil motta på Micro:bit-en, trykker vi på *Send the Message*-knappen (3) nede til høyre på figuren under.



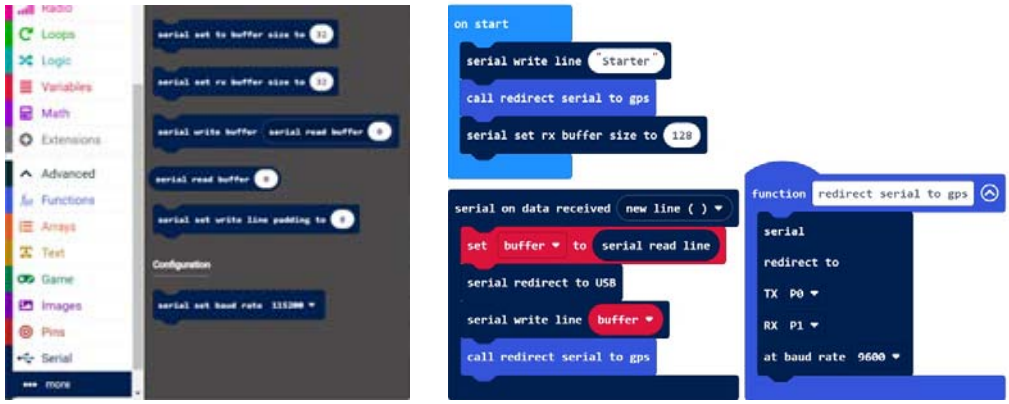
34. For å finne ut hvilken COM-port FTDI-en er koblet til åpner du enhetsbehandling på PC-en. Under fana COM-porter ser du én eller flere enheter. Om det er flere ting på lista så legg merke til hvilke porter som er i bruk, koble fra FTDI-en og se hvilken som forsvinner. Den er FTDI-porten vår.

Gjenta dette for alle meldingstypene, F0-02 NMEA GxGSA, og så videre, til du har konfigurert modulen til bare å sende den eller de setningstypene du ønsker. I tekstkonsollen bør det nå bare komme meldinger av denne typen.

Går vi til MakeCode-programmet og ser hva slags data vi mottar, bør vi se noe som ligner på figuren til høyre.



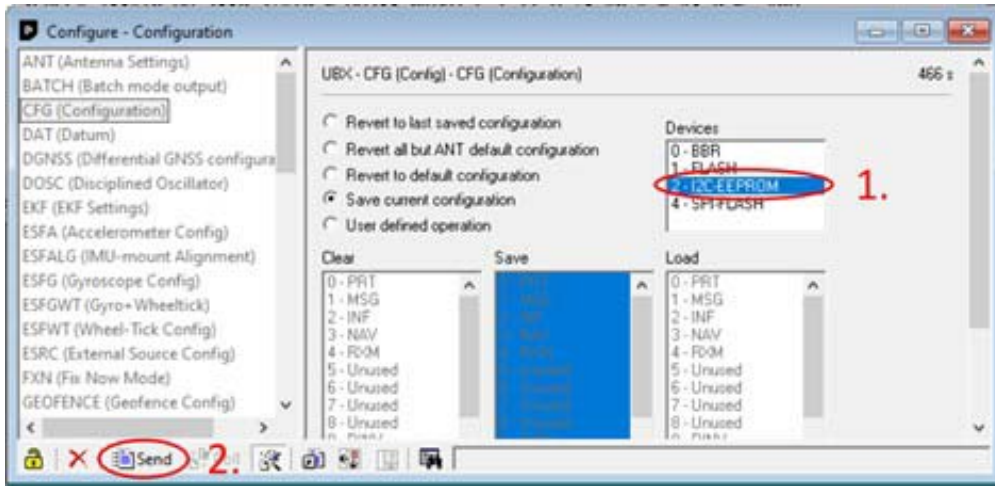
Som vi kan se av linjene med seriedata, så er ikke dette en fullstendig NMEA-setning. Det er fordi bufferen som strengen lagres i ikke er stor nok til å få plass til hele setningen. Dette kan vi heldigvis gjøre noe med i MakeCode. Ved å bruke menyen *Serial* og blokken *serial set rx buffer size to,* kan vi øke kapasiteten på bufferen til f.eks. 128 byte.



GPS-en er nå konfigurert til kun å sende én type NMEA-setning. NEO-6M har et eget batteri og en liten minneenhet (EEPROM [35]) som konfigureringen kan lagres i. Dette skjer *ikke* automatisk. For å gjøre det må vi i konfigurasjonsvinduet gå inn på *CFG (Configuration)*. Velg *Save current*

35.Electrically Erasable Programmable Read-Only Memory

configuration, og i enhetslista velger du I^{2}C-EEPROM (1) og trykker Send (2) som vist på figuren under.



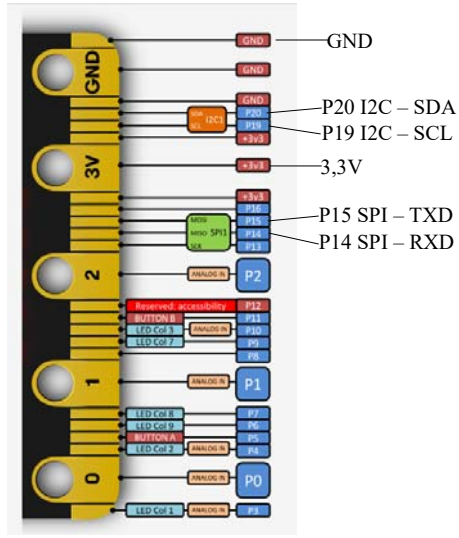
Test at det fungerer ved å koble micro:bit-en fra PC-en, slå av og på strømmen til CanSat-en, og så koble PC-en til på nytt. Hvis du fortsatt mottar bare den setningstypen du ønsker i seriemonitoren, så virker alt som det skal.

8 Lagring av data

I dette avsnittet skal vi se hvordan vi kan lagre data på minnekort og i internminnet til Micro:bit versjon 2. Micro:bit versjon 1 mangler denne funksjonen.

8.1 Skriving av data til OpenLog[36] (SD-kort)

OpenLog er en datalogger som skriver til et micro SD-kort og krever en arbeidsspenning fra 3,3V – 12,0V. Det ser derfor ut til at vi kan benytte spenningen fra Micro:bit-en. Imidlertid krever radioen E32 5V, dermed kan det være greit å kjøre både OpenLog og E32 på 5V. OpenLog kommuniserer på en serielinje (UART) slik at vi kan benytte Micro:bits innebygde seriebuss (P14 og P15). OpenLog trekker ca. 2 – 3 mA når den står og venter på data, og mellom 20 – 23 mA når data skrives til et SD-kort.



Dersom vi studerer kantkontakten til Micro:bit-en, finner vi både I^{2}C-bussen (P19/20) og Rx/Tx-bussen (P14/15).

Vi kobler P15 (TXD) hos Micro:bit-en, til RXI hos OpenLog, og tilsvarende kobler vi P14 (RXD) hos Micro:bit til TXO hos OpenLog.

Vi velger for anledningen å koble OpenLog til en 5V-spenningskilde mens Micro:bit opererer på 3,3 V, da vi har erfart at en marginal spenning kan føre til at OpenLog stadig oppretter nye filer[37]. Begge kretsene har felles jord (GND).

Config.txt fil

På SD-kortet opprettes en config.txt fil i tillegg til datafilene. Denne kan f.eks. se slik ut:

9600, 26, 3, 0, 1, 1, 0

Parametrene har følgende betydning:

baud, escape, esc#, mode, verb, echo, ignoreRX

som betyr:

baud (9600) – Datahastighet ved skriving av data til OpenLog og kan ha følgende verdier: 2400, 4800, 9600, 19200, 38400, 57600 og 115200

escape (26) – Angir hvilken ASCII karakter som er escape

36.https://www.sparkfun.com/products/13712
37.Erfaringer rapportert fra Andøya Spece Education (NAROM)

esc# (3) – Antall ganger man må sende esc for å gå inn i konfigurasjonsmodus

mode (0) – Angir type systemmodus. OpenLog starter som default i *New Log mode (0)*
Akseptable moduser er 0 = New Log, 1 = Sekvensiell Log,
2 = Kommandomodus

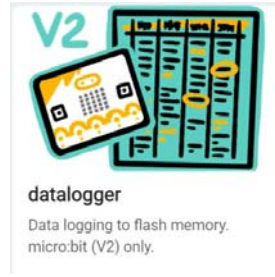
verb (1) – Ved å sette den til 1 så gis feilmeldinger, settes den 0 vises ikke feilmeldinger

echo (1) – Ved å slå på denne (1) vil kommandoer sendt til OpenLog mens den er i kommandomodus, returneres. Dette er praktisk når en er i kommandomodus.

ignoreRX (0) – Normalt vil OpenLog resetts dersom RXI linjen holdes lav over en viss tid. Dersom ignoreRX settes lik 1 vil ikke dette inntreffe.

8.2 Lagring av data i internt minne[38]

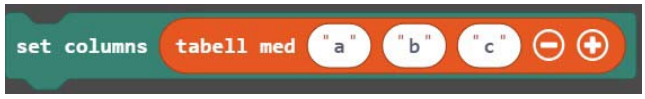
Verson 2 av Micro:bit har et intern flash minne på 256kbyte. Deler av dette minne kan brukes til å lagre data permanent (*none volatile*) over tid uavhengig av om spenningen brytes. Dette kan være praktisk for datalogging slik vi har behov for i en CanSat. For relativt korte sekvenser av data så vil slik lagring kunne erstatte SD-kort.



Micro:bit v2 gir mulighet til å lagre data i filer i et format som lett kan hentes inn i Excel. La oss se hvordan dette kan gjøres.

For å få tilgang til programvareblokker som egner seg til datalogging. Henter vi ned en utvidelse kalt “Datalogger”.

Vi får da opp en nye meny med navnet “Datalogger”. Her finner vi følgende kommandoer:



- Her lager vi en tabell med et ønsket antall kolonner tilordnet måledatene a, b og c (bør gis mer beskrivende navn). Dataene lagres “permanent” (dvs. tåler strømbrudd).

- Med denne programblokken kan vi legge til tidspunkt for når dataene er hentet inn. Vi kan velge mellom millisekunder, sekunder, minutter, timer og dager eller ingen tidsangivelse.

- Denne programblokken lar oss legge til en linje til i tabellen, der vi både spesifiserer hvilken kolonne dataene skal legges i (a, b eller c) og verdien.

38. Stoffet til dette avsnittet er hentet fra: https://microbit.org/get-started/user-guide/data-logging/

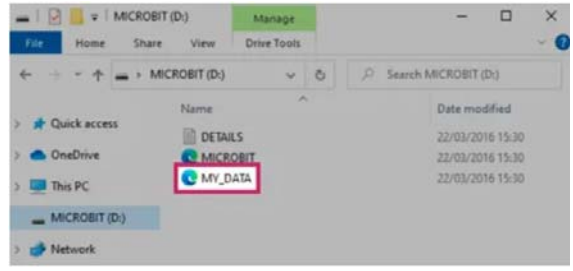
- Denne kommandoblokken vil slette hele datasettet, dvs. den vi gjøre at lagercellene kan overskrives av nye data.

- Her kan vi bestemme hva som skal skje dersom lageret går fullt.

- Dersom denne er på, så sendes dataene ut på den serielle linjen, hvilket vil være nyttig i vår sammenheng som ønsker å sende data til bakkestasjonen samtidig som vi lagrer data i lageret vårt.

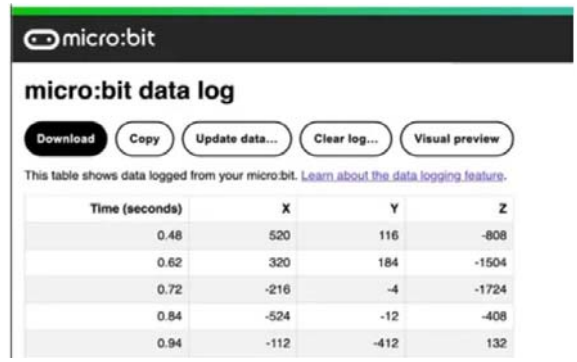
8.3 Opplasting og behandling av lagrede data

Når CanSat-en er landet og vi ønsker å hente ut data fra lageret, så gjøres dette ved å koble Micro:bit-en til en PC og åpne filutforskeren. Der finner man som forventet et drev med navnet MICROBIT (E), klikker man på dette drevet får man også opp filen: MY_DATA.



Dersom vi klikker på MY_DATA får vi opp et vindu som viser tabellen med måledata som antydet på figuren til høyre.

I MY_DATA kan vi forhåndsvise dataene på tabellform eller som grafer. Vi kan også isolere individuelle grafer ved å klikke på tegnforklaringen.



Time (seconds)	X	Y	Z
0.48	520	116	-808
0.62	320	184	-1504
0.72	-216	-4	-1724
0.84	-524	-12	-408
0.94	-112	-412	132

Vi kan bruke COPY knappen til å kopiere hele tabellen og legge den rett over i f.eks. Excel eller vi kan laste ned dataene som en CSV-file som også lett kan hentes inn i et regneark.

Det beskrevne verktøyet vil bli værende på din PC og vil ikke bli lastet opp på nettet, dermed vil ingen andre få tilgang til dataene.

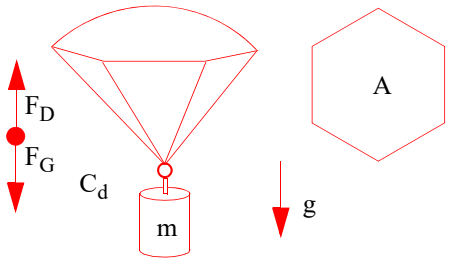
9 Beregning av fallskjermens størrelse og utforming

I dette kapittelet skal vi se nærmere på hvordan vi kan konstruere fallskjermen slik at vi oppnår ønsket fallhastighet på mellom 8 – 11 m/s.

9.1 Beregning av fallskjermens areal ut fra ønsket fallhastighet

Fallhastigheten bestemmes av en rekke ulike parametere.

Vi antar at sonden inntar terminalhastigheten etter svært kort tid, slik at den faller med konstant fart omtrent fra slipptidspunktet. Når dette inntreffer vil *drag'et*, F_D, på grunn av luftmotstanden, være lik tyngdekrafta, F_G:



$$F_D = F_G \qquad (9.1)$$

$$F_G = m \cdot g \qquad (9.2)$$

Hvor:

g = Tyngde akselerasjonen – 9,81 m/s^2
m = Massen av CanSat – maks. 0,350 kg

En kan også finne en formel for *drag'et*, F_D:

$$F_D = 1/2 \cdot C_d \cdot \rho \cdot v_t^2 \cdot A \qquad (9.3)$$

Hvor:

C_d = *Drag*-faktor varierer med blant annet formen på fallskjermen (kule-kalott $C_d = 1{,}5$)
ρ = Tettheten til luft (typisk $\rho = 1{,}22$ kg/m^3) varierer med lufttrykket
v_t = Terminalhastigheten
A = Tverrsnittsarealet av fallskjermen projisert ned på horisontalplanet.

Setter vi ligning (9.2) og (9.3) inn i ligning (9.1) og løser denne mht. terminalhastigheten, får vi:

$$v = \sqrt{\frac{2mg}{C_d \rho A}} \qquad (9.4)$$

Drag-faktoren er avhengig av formen på fallskjermen. Dersom den kan settes tilnærmet lik en kuleflate vil den i teorien ha en *drag*-faktor nær $C_d = 1{,}5$. Dersom fallskjermen tilnærmet er en sirkulær flate med samme tverrsnitt som kula, vil en C_d nær 0,75 være nærmere sannheten. En fallskjerm som vist på figuren over, vil ha en *drag*-faktor som ligge et sted mellom disse verdiene[39].

39.http://my.execpc.com/~culp/rockets/descent.html

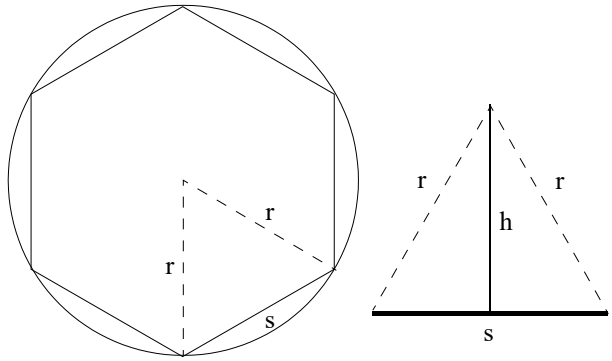
Bestem *drag*-faktoren:

Drag-faktoren, C_d, for en gitt fallskjerm kan måles ved følgende metode:

1. Mål og beregn arealet, A, til fallskjermen (projisert areal)
2. Heng CanSat-en under fallskjermen og finn den totale massen, m
3. Slipp CanSat-en i fallskjerm fra en avsats fra 5–10 meter over bakken
4. Mål tida til fallet over en gitt fallhøyde. Eller film slippet med et kamera med et kjent antall bilder i sekundet. Plasser en målestav i bildet for å bestemme dimensjonene.
5. Løs ligningen (9.4) mht C_d. Sett inn måleverdier og beregn C_d.

9.2 Beregning av fallskjermens utforming ut fra ønsket areal

For å finne det totale arealet for en fallskjerm, kan vi bruke illustrasjonen til høyre som viser en 6-kantet fallskjerm innskrevet i en sirkel.



Av figuren ser vi at fallskjermen består av 6 like trekanter. Det betyr at det totale projiserte arealet for denne fallskjermen er $A = 6 \cdot A_T$ hvor A_T er arealet til hver av trekantene.

For å finne arealet for den enkelte trekanten, A_T, kan vi benytte formelen:

$$A_T = \frac{sh}{2} \tag{9.5}$$

Vi kan måle lengdene s og h direkte på fallskjermen. Det totale projiserte arealet av den 6-kantede fallskjermen blir dermed:

$$A = \frac{6sh}{2} \tag{9.6}$$

Det er mulig å finne et mer generelt utrykk for beregning av arealet for en fallskjerm formet som en regulær mangekant med et vilkårlig antall kanter. For mer informasjon om beregning av arealet til fallskjermer, se:

http://www.sunward1.com/imagespara/The%20Mathematics%20of%20Parachutes%28Rev2%29.pdf

Når fallskjermens totale areal A er bestemt, kan fallhastigheten bestemmes av ligning: (9.4)

På kurset vil det bli gjennomført slipp av CanSat med fallskjerm uten modifikasjoner. Fallhastig-

heten kan dermed bestemmes dersom slipphøyden er kjent (for eksempel ved hjelp av data fra trykksensoren) og tiden CanSat-en bruker på fallet til bakken (hvordan kan falltiden bestemmes?). Ved å anta konstant fart, kan formelen $v = s/t$ brukes for å beregne fallhastigheten.

Sammenlign målingene av fallhastigheten med beregningene.

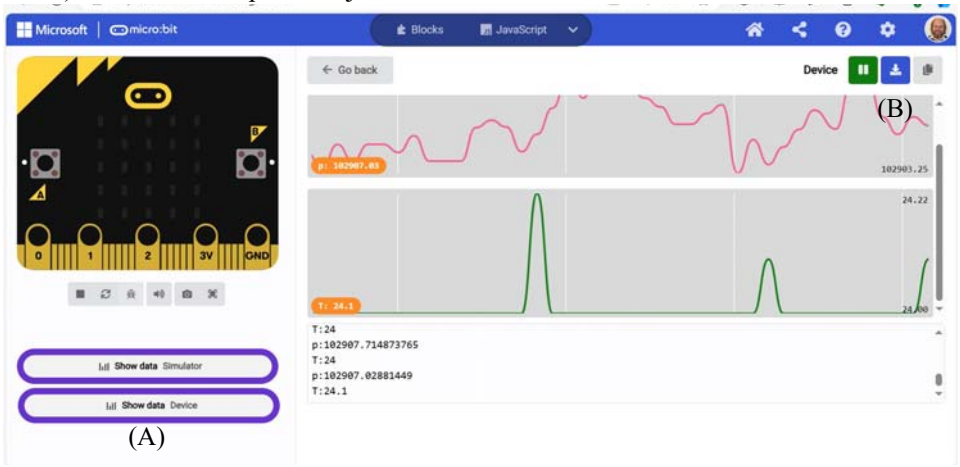
10 Presentasjon og behandling av måledata

I dette kapittelet skal vi gi eksempler på hvordan vi henter inn data til Excel og Google Earth for visning av resultatene. I tillegg skal vi se hvordan vi kan bruke Python[40] til å lese, plotte og analysere data. Men la oss først se hvordan vi kan bruke MakeCode til å presentere data og lage en datafil vi evt. kan hente inn i andre programmer som f.eks. Excel.

10.1 Bruk av MakeCode for presentasjon og lagring av data

Vi har tidligere sett hvordan vi kan sende data fra Micro:bit til PC-en (avsnitt 5.3, side 68).

Presenter dataene i MakeCode
Vi antar at dataene kommer inn til PC-en via en USB-kontakt. Vi går da inn i MakeCode og programfilen (blokkoden) for mottakeren. Det vil da dukke opp et ikon (A) nede til venstre ("Show data device"), for "real time" presentasjon av dataene etter som de ankommer Micro:bite-en.

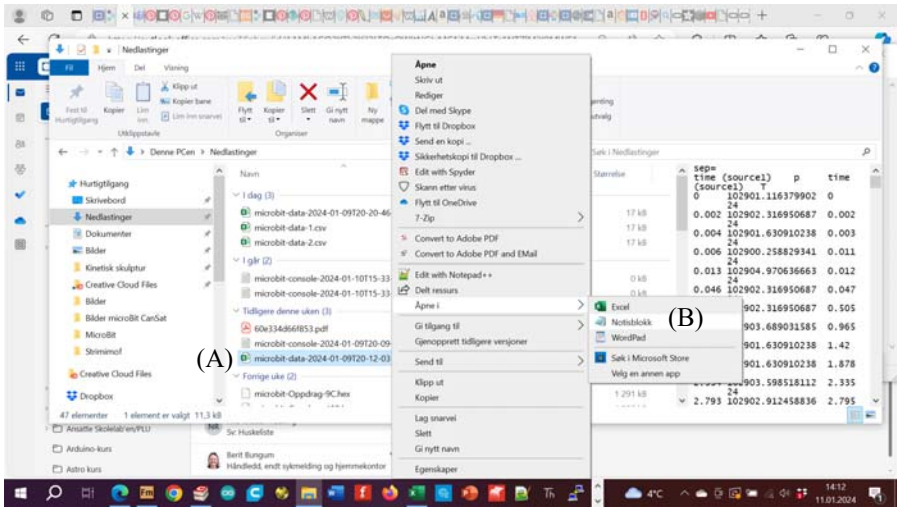


På figuren ser vi lufttrykk (p – øverst) og temperatur (T – nederst) på skjermen. Øverst i høyre hjørne ser vi symbolet for nedlasting til file (B) for videre behandling i andre programmer f.eks. Python eller Excel, se avsnittene 10.2, side 109 og 10.3, side 116. Vi velger nedlasting som CSV-file. Denne blir da lagret i katalogen "nedlastinger".

40. Python programmene er utviklet av Thor Inge Hansen ved Skien vgs.

Klargjøring av datafil

Selv om vi har oversatt tekststrengene til tall og skulle forvente at f.eks. Excel oppfatter dataene som tall, så viser det seg at dette ofte ikke er tilfelle. For å bøte på dette går vi inn på "nedlastinger" og åpner filen (A) i et enkelt redigeringsprogram som f.eks. notepad++ eller notebook (B).



Her kan vi bruke "Erstatt" og erstatte alle "." i tallene med ",", for så å lagre filen som en .csv-fil. Når vi nå åpner den i Excel vil verdiene i kolonnene oppføre seg som tall og ikke som tekst. Vi er nå klare til å behandle målingene, evt. presentere dem som grafer. Vi har for eksempel også muligheten til å ta målt lufttrykk og regne om til høyde om vi evt. ikke har gjort det som en del av målingen ombord i CanSat-en.

Plotting og videre behandling av dataene

Se avsnittene 10.2, side 109 og 10.3, side 116 for evt. plotting, videre behandling og analyse.

10.2 Bruk av Python for å presentere og analysere dataene[41]

I dette avsnittet skal vi gi et eksempel på hvordan vi kan presentere dataene ved hjelp av Python.

10.2.1 Installasjon og oppstart med Python

La oss først se hvordan vi kan etablerer programmeringsmiljøet for bruk av Python. Dette er blant annet hentet fra boka *Python for realfag* [1].

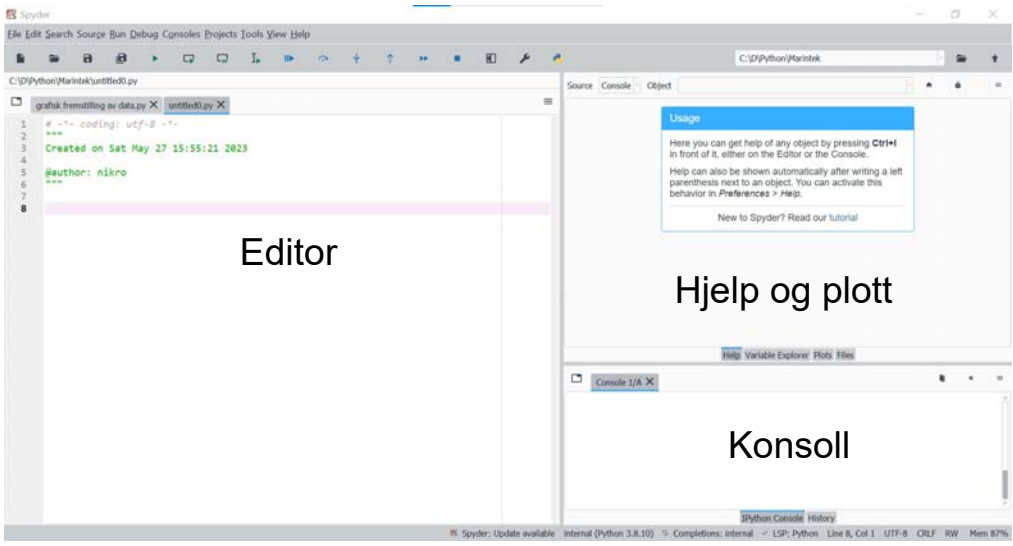
41. Dette avsnittet bygger på arbeidet til Thor Inge Hansen ved Skien videregående skole som har utviklet presentasjon og analyseverktøyet.

For å installere Python-pakken går vi til hjemmesiden til firmaet Anaconda og laster ned og installerer:

https://www.anaconda.com/download/

I tillegg til Spyder som er Python editoren, får man med flere bibliotekspakker og andre mer eller mindre nyttige programmer.

Når vi starter Spyder får vi opp en side som ligner på den som er vist i figuren under, med tre felter eller vinduer med ulike funksjoner: *Editor*, *hjelp* og *plott*, og et *konsoll* som viser responsen som programmet gir, inkludert advarsler og feilmeldinger.



I "Editoren" skriver vi inn kommandoene, evt. programkoden vi ønsker å kjøre. I vinduet "Hjelp og plott" kan vi få tips og hint, evt. oppsøke hjelpefunksjoner, vi kan se innhold i variabler, vise plott og utskrifter eller oversikt over filer. I vinduet "Konsoll" får vi også svar fra programmet, feilmeldinger, advarsler ("warnings") og kommandohistorien.

I stedet for å liste opp alle mulighetene til Spyder så skal vi gjennomgå et aktuelt program-eksempel, men først la oss se litt på et typisk filformat for dataene våre slik de kan framstå etter at de er lagt ned på filen.

10.2.2 Organisering av data i filen

La oss ta utgangspunkt i måledata organisert linje for linje i en fil. Vi tenker oss at vi ønsker å gjøre målinger til gitte tidspunkter, f.eks. en gang hvert sekund. Når vi først gjør en måling ønsker vi å måle flere ulike parametere samtidig. En datalinje kan begynne med et navn som identifiserer dataene (ID), tidspunktet fra strømmen ble slått på, temperatur, lufttrykk, beregnet høyde, med

mer. Dataene legges etter hverandre på linjen med komme mellom. Når så neste måling skal gjøres, legges de på neste linje slik at det etter hvert dannes en tabell av data som vist i eksempelet under.

```
Fil  Rediger  Format  Vis  Hjelp
Terminal log file
Date: 12.12.2023 - 09:42:33
-----------------------------------------
ID,Tid,Temp,Trykk,Hoyde,
CanSat-NTNU,1777,24.2,1004.45,19.3,
CanSat-NTNU,4529,24.3,1004.4,19.7,
CanSat-NTNU,7287,24.4,1004.47,19.1,
CanSat-NTNU,10043,24.4,1004.46,19.2,
CanSat-NTNU,12799,24.4,1004.45,19.3,
CanSat-NTNU,15555,24.4,1004.42,19.5,
CanSat-NTNU,18317,26.3,1004.44,19.4,
CanSat-NTNU,21078,27.6,1004.38,19.8,
CanSat-NTNU,23837,28.6,1004.39,19.8,
CanSat-NTNU,26596,29,1004.4,19.7,
```

```
Fil  Rediger  Format  Vis  Hjelp
ID,Tid,Temp,Trykk,Hoyde,
CanSat-NTNU,1777,24.2,1004.45,19.3,
CanSat-NTNU,4529,24.3,1004.4,19.7,
CanSat-NTNU,7287,24.4,1004.47,19.1,
CanSat-NTNU,10043,24.4,1004.46,19.2,
CanSat-NTNU,12799,24.4,1004.45,19.3,
CanSat-NTNU,15555,24.4,1004.42,19.5,
CanSat-NTNU,18317,26.3,1004.44,19.4,
CanSat-NTNU,21078,27.6,1004.38,19.8,
CanSat-NTNU,23837,28.6,1004.39,19.8,
```

Terminalprogrammet som vi her har brukt, lager også en egen topptekst med tre linjer som vi velger å ta bort (se innramming). Fila på SD-kortet vil ikke få en slik ekstra topptekst.

I dette eksempelet er mikrokontrollen og sensorene aktive under hele måleperioden, dermed kan vi trykt legge inn en tekstlinje øverst i fila ved programstart. En slik tekstlinje vil være nyttig for både Python og Excel for å identifisere de ulike kolonnene.

Lengst til venstre ser vi navnet som identifiserer måleserien vår, deretter følger tiden siden oppstart i millisekunder, temperatur, lufttrykk og beregnet høyde over havet (I dette eksemplet har vi gjort all omregning og nullpunktjustering i programvaren i CanSat-en. En må vurdere om denne omregningen og nullpunktjusteringen bør gjøres i analyseprogrammet[42].

Komma er et greit skilletegn når vi skal bruke Python til å analysere dataene siden desimaltegnet er et punkt. Slike filer kalles CSV-filer (Comma Separated Values).

La oss se hvordan vi kan hente utvalgte måledata inn i Python for presentasjon.

10.2.3 Lesing og presentasjon av data fra fil[43]

La oss se på et grunnleggende eksempel. Eksempelet leser data fra filen vist i forrige avsnitt. Eksempelet burde være enkelt å modifisere og utvide. Programeksempelet finnes i sin helhet i vedlegg G.

Her går vi gjennom programmet del for del slik at det skal være mulig å forstå kommandolinjene og dermed være lettere å gjøre endringer og bygge ut programmet etter egne behov. Noen vil kanskje synes at gjennomgangen er i grundigste laget, men kan være nyttig for de som ikke kjenner Python fra før.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

42. Vi valgte denne løsningen da vi tidligere ønsket å lage en liten håndholdt høydemåler med eget display.
43. Programmet er skrevet av Thor Inge Hansen ved Skien videregående skole

Først importeres diverse biblioteker som brukes i forbindelse med utregningene og presentasjonene mm. `numpy` er et bibliotek som håndterer arrayer av data som vi innledningsvis ikke bruker, `matplotlib.pyplot` del av et plottebibliotek og `pandas` er et kraftig bibliotek for dataanalyse som vi her i første omgang bruker for å hente inn data fra CSV-fila.

I Python er det dessuten vanlig å lage forkortelser av biblioteksnavnene slik at det skal være lettere å hente ut funksjoner fra bibliotekene. `numpy` forkortes f.eks. np (`numpy as np`) og `matplotlib.pyplot` forkortes plt (`matplotlib.pyplot as plt`) osv.

Henting og strukturering av data fra file på PC'en

```
#Henter inn fila som en dataframe i pandas
data = pd.read_csv("Data/CanSat_5.log", delimiter=',')
```

Måleverdiene hentes ut av datafila ved hjelp av biblioteksfunksjonen `pd.read_csv` fra pandas-biblioteket (pd.). Biblioteksfunksjonen har som vi ser, to argumenter. Det første er *filstien og filnavnet*: `Data/` i fila `CanSat_5.log`. og det andre argumentet angir skilletegnet mellom verdiene, i dette tilfellet er det "," (vi har tidligere også brukt "-"). Verdiene legges inn i dataarrayet `data`. Vi legger merke til alle linjer som begynner med # er kommentarer.

```
Time = data['Tid'].values/1000
temperatur = data['Temp'].values
lufttrykk = data['Trykk'].values
hoyde = data['Hoyde'].values
```

Det neste som gjøres er å plukke ut verdiene i utvalgte kolonner og legge disse i egne array eller lister som har navnene `Time`, `temperatur`, `lufttrykk` og `hoyde`. Disse henter ut samtlige verdier i de angitte kolonnene (unntatt overskriften) og legger dem inn i lista. Siden vi har et kolonnenavn, bruker vi dette for å identifisere kolonnene. Dersom vi ikke har det, kan vi benytte første dataverdien i hver kolonnen for å angi kolonnen, da må vi gjøre det for hver ny fil vi skal analysere eller visualisere. Hver dataliste er gitt navn som skulle gjøre det lett å identifisere hva data det handler om.

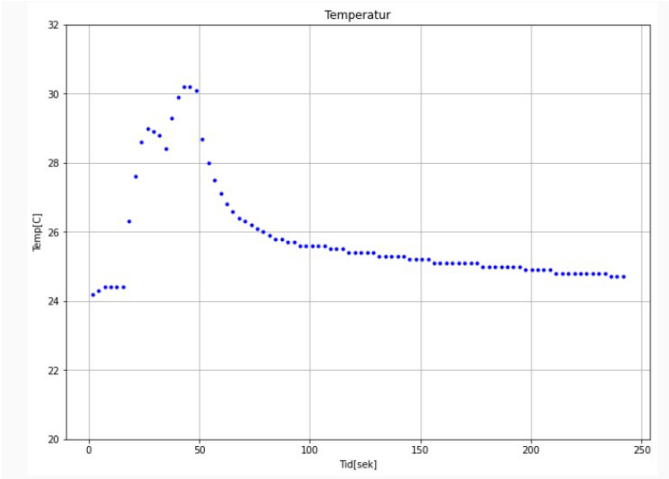
Vi kan nå gjøre en rekke analyser og beregninger på de innsamlede dataene. Her skal vi inntil videre kun plotte dataene som funksjon av tiden.

Plotting av data

Vi er nå klare for plotting av dataene våre. I første omgang lager vi tre plott.

Først plotter vi temperaturen som funksjon av tiden.

Her ser vi hvordan temperaturen varierer fra 24C og opp til drøyt 30C. Årsaken til temperaturøkningen skyldes at det ble pustet på sensoren.



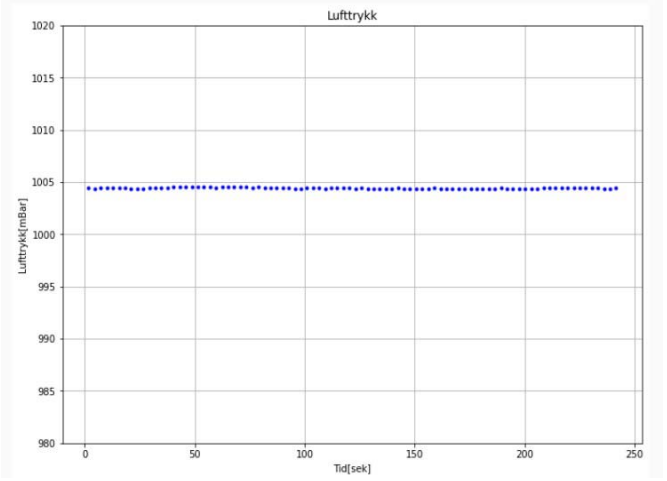
Programbiten under viser hvordan dette plottet framkommer:

```
plt.figure(1,figsize=(11,8)) # Angir figurnummer og figur størrelse (cm)
plt.plot(Time,temperatur,'b.') # Plotter spenning som funk. av tid med blå .
plt.ylabel('Temp[C]') # Angir tekst på den vertikale aksen
plt.xlabel('Tid[sek]') # Angir tekst på den horisontale aksen
plt.title('Temperatur') # Angir figuroverskrift
plt.ylim(20.0,32.0) # Angir skalaen på den vertikale aksen
plt.grid() # Angir at det skal tegnes opp et rutenett
```

I `plt.figure` vil normalt figuren bli angitt i inch, men dersom den generelle måleenheten i programmet er satt til cm er det denne som gjelder. I `plt.plot` vil måleverdiene angis med blå punkter `'b.'`.

Dernest plotter vi måleverdiene for lufttrykk som funksjon av tiden.

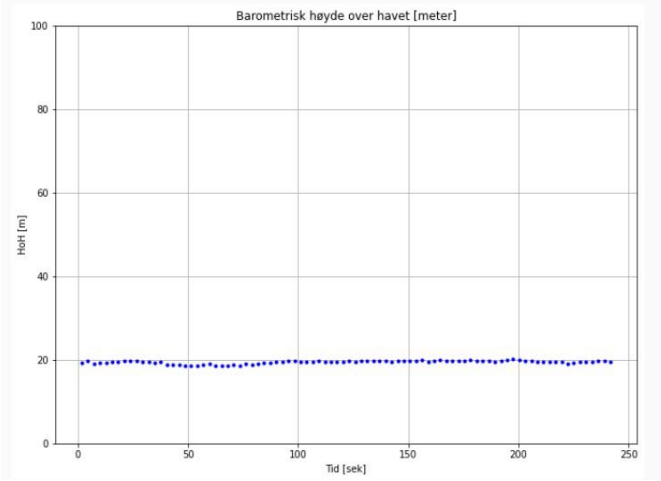
Her ser vi hvordan lufttrykket forandrer seg over et tidsintervall på 240 sekunder. Ikke overraskende er endringen neglisjerbar.



Her er det ingen overraskelser mht. programmeringen:

```
plt.figure(2,figsize=(11,8))
plt.plot(Time,lufttrykk,'b.')
plt.ylabel('Lufttrykk[mBar]')
plt.xlabel('Tid[sek]')
plt.title('Lufttrykk')
plt.ylim(980,1020)
plt.grid()
```

Til sist plotter viser den beregnede høyden over havet ut fra lufttrykket. Høyden er i dette tilfellet ikke nullpunktjustert, og programmet er som forventet.




```
plt.figure(3,figsize=(11,8))
plt.plot(Time,hoyde,'b.')
plt.ylabel('HoH [m]')
plt.xlabel('Tid [sek]')
plt.title('Barometrisk høyde over havet [meter]')
plt.ylim(0,100)
plt.grid()
```

Dersom vi ønsker å plotte flere kurver i samme diagram som vist i figuren til høyre, føyer vi til en ekstra linje med plt.plot().

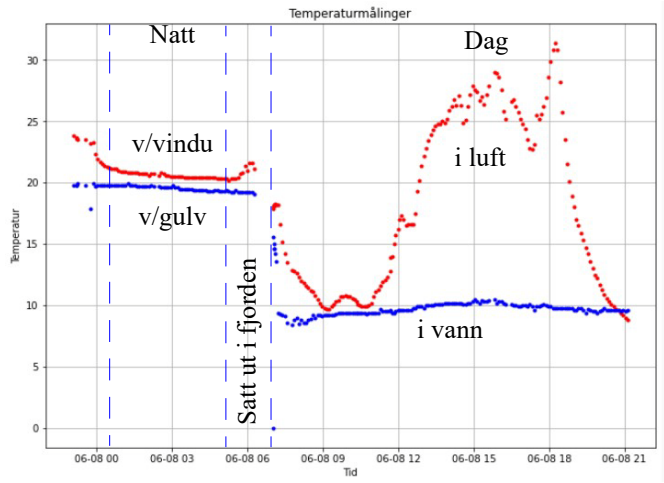
Her har vi plottet temperaturen i vann (blå kurve) og lufttemperaturen (rød kurve).

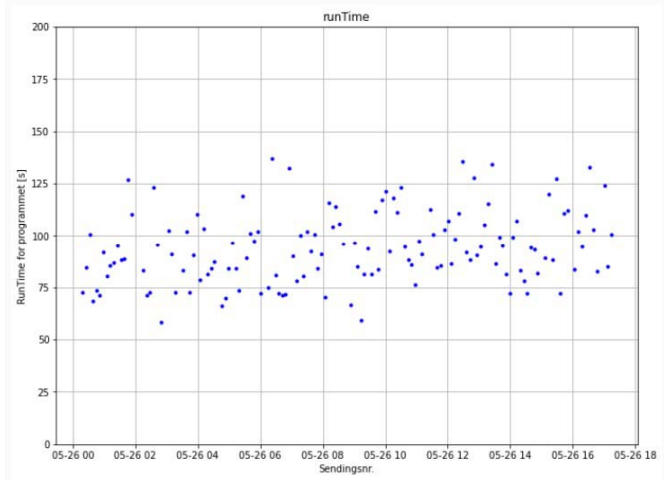
Vi legger merke til at de to sensorene ikke er helt samstemte når begge er innendørs. Den ene er plassert ved gulvet den andre foran vinduet.

Målingene er hentet fra et annet prosjekt der vi både måler temperatur i sjø og luft.

Til slutt plotter vi *på*-tiden som funksjon av tiden.

Her ser vi hvordan på-tiden varierer over et tidsintervall på ca. 18 timer den 26. mai 2023. Vi ser at på-tiden varierer fra ca. 60 til 140 sekunder.





Her er heller ingen overraskelser mht. programmeringen:

```
plt.figure(3,figsize=(11,8))
plt.plot(timelist,runTime,'b.')
plt.ylim(0,200)
plt.title('runTime')
plt.xlabel('Sendingsnr.')
plt.ylabel('RunTime for programmet [s]')
plt.grid()
```

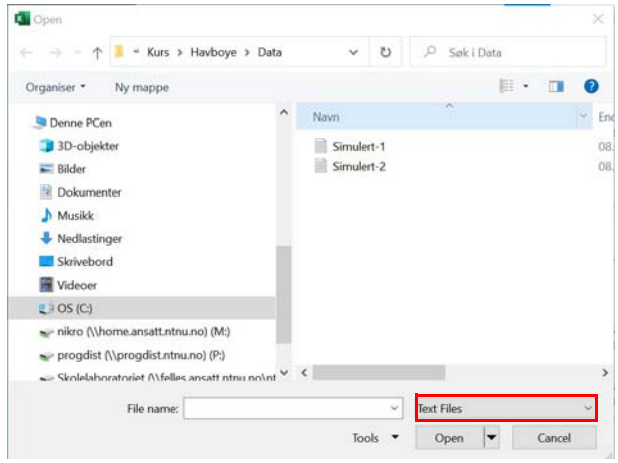
10.3 Bruk av Excel for visualisering av data

Et praktisk og vanlig verktøy for analyse av data er Excel. I dette avsnittet skal vi vise hvordan vi i prinsippet kan importere og tegne ut grafer av innsamlede data fra bøya. I dette eksempelet har vi følgende data: Måling nr., tidsangivelse (Epoketid), lengde- og breddegrad hentet fra GPS, luft- og vanntemperatur, luftfuktighet, lufttrykk og lysintensitet på overflata. Dataene for dette eksempelet er simulert og er hentet fra et annet prosjekt, men prinsippene burde kunne brukes.

10.3.1 Importer data fra en tekst-fil til Excel

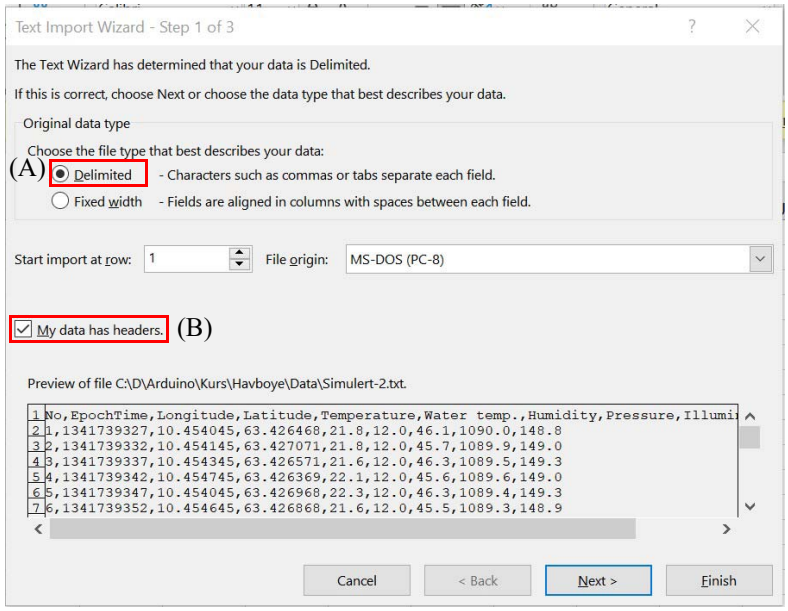
Importering av data i Excel kan gjøres på ulike måter. En måte som fungerer er å hente inn filen med "Open". Da vil man ledes gjennom følgende vinduer for at dataene skal bli formatert på ønsket måte.

Velg ønsket fil:

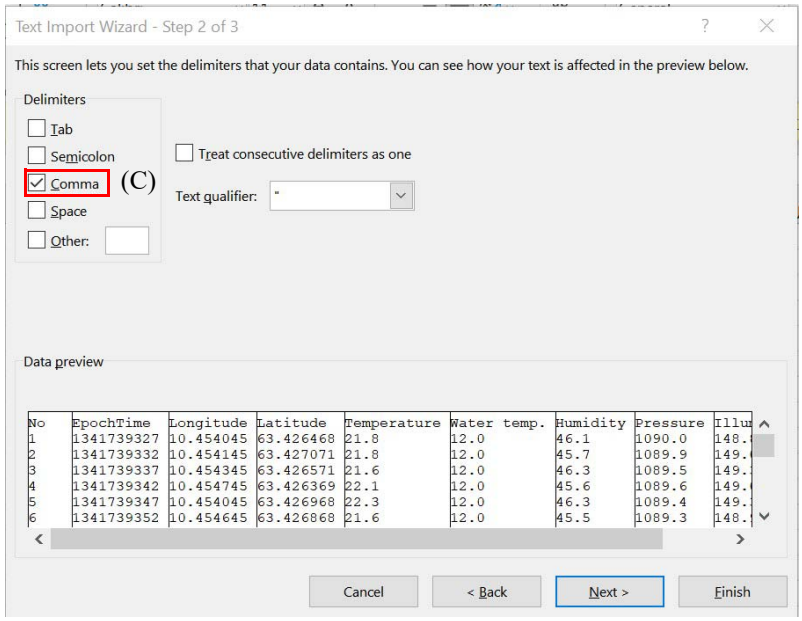


Siden det er en tekst-fil så velg denne typen format i innboksen nederst i høyre hjørne, dermed blir slike filer synlige i fil-oversikten.

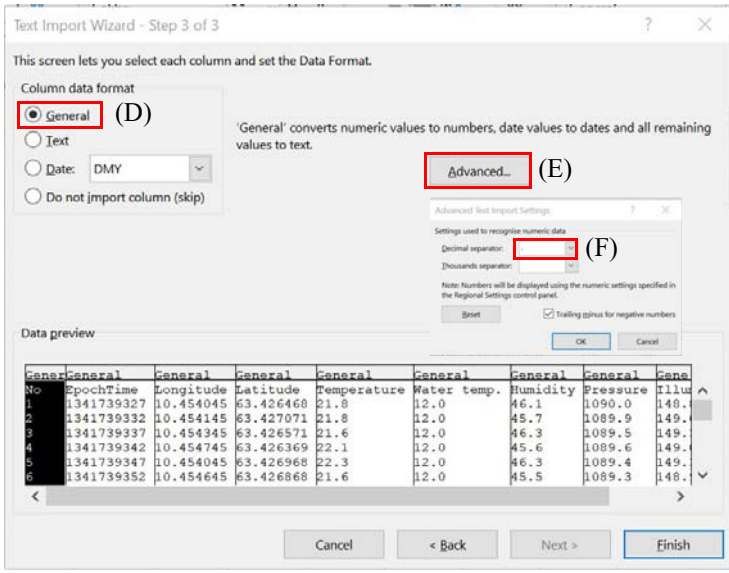
Velg “Data med skilletegn” (Delimited) (A) og bestem fra hvilken rad du ønsker å starte å importere data. Likeså er det viktig å merke av om det brukes topptekst (B), som vi har gjort her. Trykk så “Neste”:



Velg hvilken type skilletegn som er benyttet. I vårt tilfelle har vi benyttet “komma” (C).



Tilslutt velges hvilken type data det gjelder, i vårt tilfelle er det standard tallformat (General) (D). Dersom vi skal importere flyttall (med komma), velges desimalpunktet (F), “,” eller “.” ved å velge “Avansert” (E). Det må vi gjøre i vårt tilfelle, ellers har dataene en tendens til å bli til datoer eller annen tekst.



Dermed skal dataene være importert i regnearket og plassert i kolonner som vist på figuren under.

	A	B	C	D	E	F	G	H	I	J
1	No	EpochTim	Longitude	Latitude	Temperati	Water ten	Humidity	Pressure	Illuminance	
2	1	1,34E+09	10,45405	63,42647	21,8	12	46,1	1090	148,8	
3	2	1,34E+09	10,45415	63,42707	21,8	12	45,7	1089,9	149	
4	3	1,34E+09	10,45435	63,42657	21,6	12	46,3	1089,5	149,3	
5	4	1,34E+09	10,45475	63,42637	22,1	12	45,6	1089,6	149	
6	5	1,34E+09	10,45405	63,42697	22,3	12	46,3	1089,4	149,3	
7	6	1,34E+09	10,45465	63,42687	21,6	12	45,5	1089,3	148,9	
8	7	1,34E+09	10,45435	63,42697	22,1	12	45,9	1089,6	148,8	
9	8	1,34E+09	10,45485	63,42657	21,5	12	46,2	1089,8	149	
10	9	1,34E+09	10,45455	63,42637	22,2	12	45,7	1089,8	148,8	
11	10	1,34E+09	10,45485	63,42627	21,7	12	46,3	1089,6	149,1	
12	11	1,34E+09	10,45485	63,42707	21,6	12	45,7	1089,8	148,7	
13	12	1,34E+09	10,45455	63,42667	21,9	12	45,6	1089,9	148,9	
14	13	1,34E+09	10,45415	63,42697	22,1	12	46,2	1090,1	149,3	
15	14	1,34E+09	10,45405	63,42637	21,9	12	45,8	1089,9	148,7	
16	15	1,34E+09	10,45415	63,42667	21,8	12	46,1	1089,9	148,5	
17	16	1,34E+09	10,45475	63,42657	21,5	12	45,5	1089,8	149	
18	17	1,34E+09	10,45455	63,42697	21,6	12	46,1	1089,5	148,8	
19	18	1,34E+09	10,45475	63,42697	22,1	12	45,8	1089,7	148,7	
20	19	1,34E+09	10,45465	63,42697	22,3	12	46	1089,9	149	

Siden vi ikke trenger å gjøre beregninger på noen av dataene, så kan vi illustrere verdiene i hver kolonne som grafer om vi skulle ønske det.

10.3.2 Lage grafer i Excel

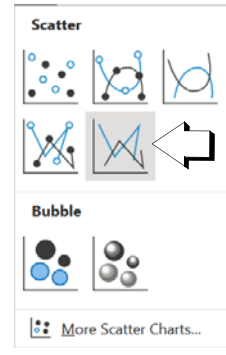
Slik kan vi lage grafer:

1. Merk de to kolonnene som skal representeres av en graf.
   Det kan være lurt å lage en overskrift på kolonnene om det ikke alt er gjort. Overskriften vil automatisk tilordnes seriene i grafen. Merk de kolonnene som skal utgjøre datasettet. I vårt tilfelle kan det være fornuftig å bruke Epoketiden langs x-aksen. Vi får da tegnet grafene som funksjon av tiden, f.eks. temperatur som funksjon av tiden.

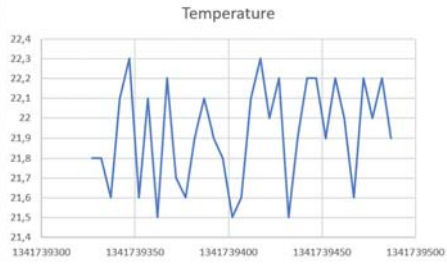
	A	B	C	D	E	F	G	H	I	J
1	No	EpochTim	ongitude	Latitude	Temperati	Water ten	Humidity	Pressure	Illuminance	
2	1	1,34E+09	10,45405	63,42647	21,8	12	46,1	1090	148,8	
3	2	1,34E+09	10,45415	63,42707	21,8	12	45,7	1089,9	149	
4	3	1,34E+09	10,45435	63,42657	21,6	12	46,3	1089,5	149,3	
5	4	1,34E+09	10,45475	63,42637	22,1	12	45,6	1089,6	149	
6	5	1,34E+09	10,45405	63,42697	22,3	12	46,3	1089,4	149,3	
7	6	1,34E+09	10,45465	63,42687	21,6	12	45,5	1089,3	148,9	
8	7	1,34E+09	10,45435	63,42697	22,1	12	45,9	1089,6	148,8	
9	8	1,34E+09	10,45485	63,42657	21,5	12	46,2	1089,8	149	
10	9	1,34E+09	10,45455	63,42637	22,2	12	45,7	1089,8	148,8	
11	10	1,34E+09	10,45485	63,42627	21,7	12	46,3	1089,6	149,1	
12	11	1,34E+09	10,45485	63,42707	21,6	12	45,7	1089,8	148,7	
13	12	1,34E+09	10,45455	63,42667	21,9	12	45,6	1089,9	148,9	
14	13	1,34E+09	10,45415	63,42697	22,1	12	46,2	1090,1	149,3	
15	14	1,34E+09	10,45405	63,42637	21,9	12	45,8	1089,9	148,7	
16	15	1,34E+09	10,45415	63,42667	21,8	12	46,1	1089,9	148,5	
17	16	1,34E+09	10,45475	63,42657	21,5	12	45,5	1089,8	149	
18	17	1,34E+09	10,45455	63,42697	21,6	12	46,1	1089,5	148,8	
19	18	1,34E+09	10,45475	63,42697	22,1	12	45,8	1089,7	148,7	
20	19	1,34E+09	10,45465	63,42697	22,3	12	46	1089,9	149	

2. Velg type representasjon fra menyen "Sett inn" (Insert) og velg f.eks. punktdiagram med linjer mellom punktene i diagrammet:

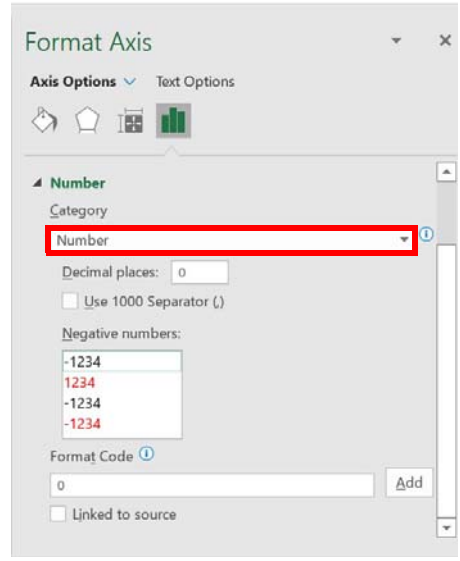
3. Velg linjediagram fra nedtrekksmenyen som vist på figuren under til høyre.



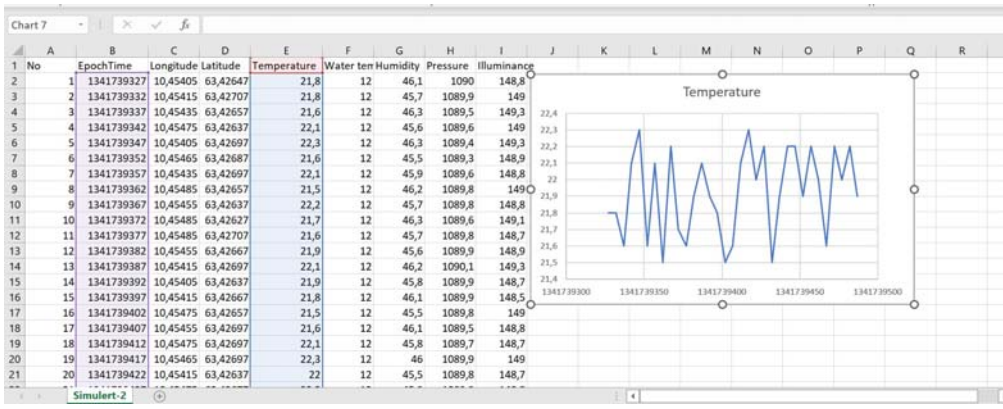
4. Når dette er gjort vil diagrammet tegnes ut som vist på figuren under.



5. Ønsker vi å endre på formatet på aksene, så klikker vi på den akse-benevningen vi ønsker å endre og vi får opp vinduet vist til høyre. Her kan vi f.eks. endre på tallformatet til akse-benevningen, f.eks. fra vitenskapelig notasjon til vanlig tallnotasjon for tidsangivelsen langs x-aksen.



6. Vi kan dermed ende opp med en presentasjon som vist under:

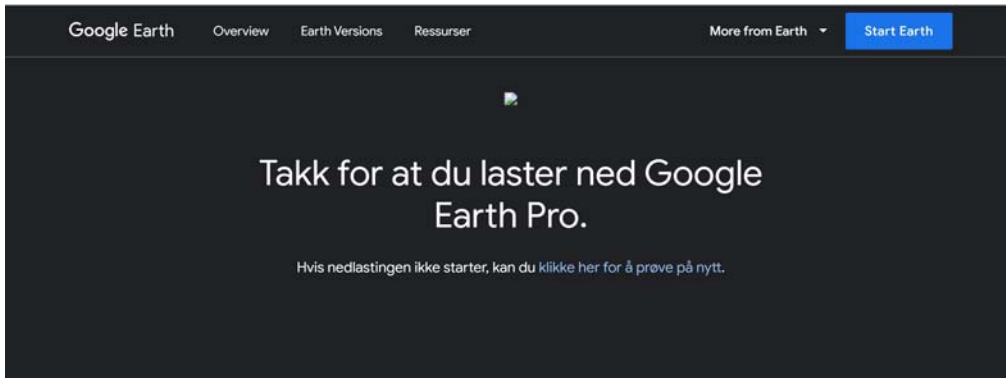


	No	EpochTime	Longitude	Latitude	Temperature	Water ten	Humidity	Pressure	Illuminance
2	1	1341739327	10,45405	63,42647	21,8	12	46,1	1090	148,8
3	2	1341739332	10,45415	63,42707	21,8	12	45,7	1089,9	149
4	3	1341739337	10,45435	63,42657	21,6	12	46,3	1089,5	149,3
5	4	1341739342	10,45475	63,42637	22,1	12	45,6	1089,6	149
6	5	1341739347	10,45405	63,42697	22,3	12	46,3	1089,4	149,3
7	6	1341739352	10,45465	63,42687	21,6	12	45,5	1089,3	148,9
8	7	1341739357	10,45435	63,42697	22,1	12	45,9	1089,6	148,8
9	8	1341739362	10,45485	63,42657	21,5	12	46,2	1089,8	149
10	9	1341739367	10,45455	63,42637	22,2	12	45,7	1089,8	148,8
11	10	1341739372	10,45485	63,42627	21,7	12	46,3	1089,6	149,1
12	11	1341739377	10,45485	63,42707	21,6	12	45,7	1089,8	148,7
13	12	1341739382	10,45455	63,42667	21,9	12	45,6	1089,9	148,9
14	13	1341739387	10,45415	63,42697	22,1	12	46,2	1090,1	149,3
15	14	1341739392	10,45405	63,42637	21,9	12	45,8	1089,9	148,7
16	15	1341739397	10,45415	63,42667	21,8	12	46,1	1089,9	148,5
17	16	1341739402	10,45475	63,42657	21,5	12	45,5	1089,8	149
18	17	1341739407	10,45455	63,42697	21,6	12	46,1	1089,5	148,8
19	18	1341739412	10,45475	63,42697	22,1	12	45,8	1089,7	148,7
20	19	1341739417	10,45465	63,42697	22,3	12	46	1089,9	149
21	20	1341739422	10,45415	63,42637	22	12	45,5	1089,8	148,7

10.4 Bruk av Google Earth for visning av posisjon fra GPS

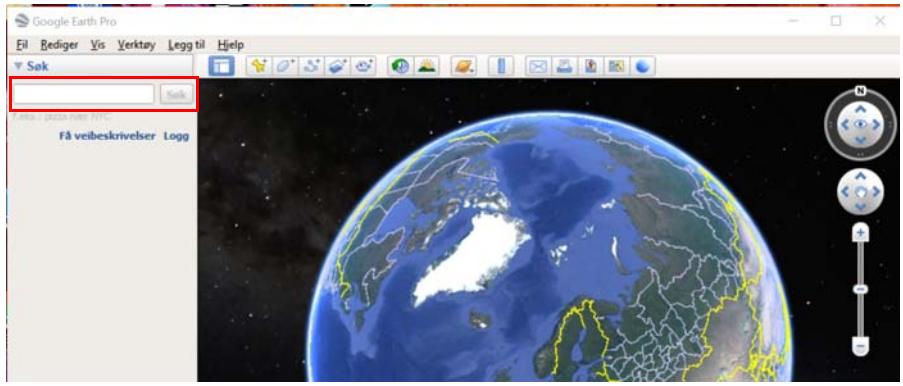
I dette avsnittet skal vi gi en oppskrift for hvordan vi kan vise en posisjon eller en trase i Google Earth.

Google Earth kan lastes ned og installeres fra følgende adresse: https://www.google.com/earth/versions/download-thank-you/



10.4.1 Plotting av en enkeltposisjon

Når man starter Google Earth får man opp følgende vindu.



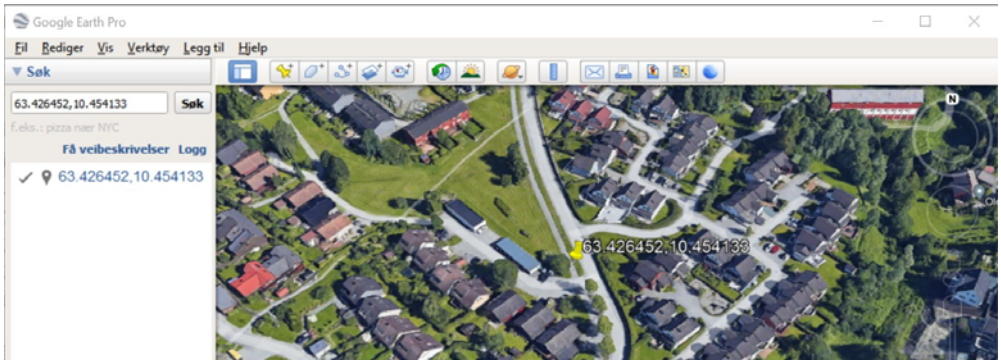
Ved å skrive inn bredde- og lengdegrader i innboksen øverst til venstre i vinduet, vil vi kunne "forflytte oss" til det angitte stedet på kloden.

Lengde- og breddegrader legges inn som vist under:

63.426452,10.454133
Breddegrader[°],Lengdegrader[°]

Her er selvfølgelig rekkefølgen viktig.

Legg merke til at koordinatene minst bør ha med 4 siffer etter komma, gjerne 5 eller 6, høyde godtas ikke.



10.4.2 Plotting av en trase i Google Earth

Dersom vi ønsker å plotte en trase i Google Earth kan vi benytte et programmeringsspråk kalt "Keyhole Markup Language" (KML) som er utviklet for visualisering av to- og tredimensjonale strukturer knyttet til kartdata.

Siden språket er temmelig omfattende og vi kun trenger å bruke en beskjeden del av det, så benytter vi en ferdige programkode og klipper inn våre data for lengde-, breddegrad og høyde. Disse legges inn som en liste med data i kml-koden (se under), før kml-fila lagres med et ønsket navn.

Koordinater og høyde data legges inn som vist under:

-112.2550785337791,36.07954952145647,2357
Lengdegrader[°],Breddegrader [°],Høyde [m]

Legg merke til at rekkefølgen på koordinatene er omvendt av det vi la inn i innboksen på forsiden av Google Earth.

Programkode skrevet i kml (legg merke til at et sett med eksempelkoordinater og høyde er klippet inn – rød kode). Her er det lagt inn 11 posisjoner. Du må gjerne legge inn flere eller færre.

```
<?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>
```

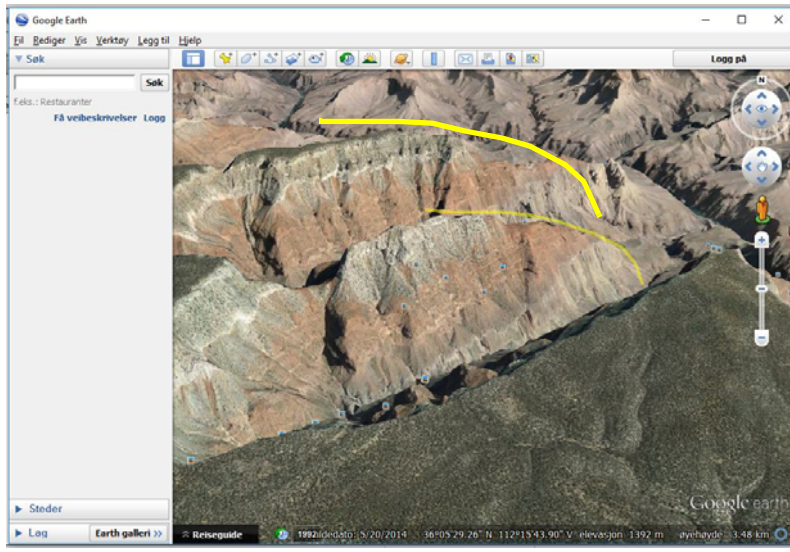
```
    </LineStyle>
    <PolyStyle>
      <color>7f00ff00</color>
    </PolyStyle>
  </Style>
  <Placemark>
    <name>Absolute Extruded</name>
    <description>Transparent green wall with yellow outlines</description>
    <styleUrl>#yellowLineGreenPoly</styleUrl>
    <LineString>
      <extrude>0</extrude>
      <tessellate>0</tessellate>
      <altitudeMode>absolute</altitudeMode>
      <coordinates>
        -112.2550785337791,36.07954952145647,2357
        -112.2549277039738,36.08117083492122,2357
        -112.2552505069063,36.08260761307279,2357
        -112.2564540158376,36.08395660588506,2357
        -112.2580238976449,36.08511401044813,2357
        -112.2595218489022,36.08584355239394,2357
        -112.2608216347552,36.08612634548589,2357
        -112.262073428656,36.08626019085147,2357
        -112.2633204928495,36.08621519860091,2357
        -112.2644963846444,36.08627897945274,2357
        -112.2656969554589,36.08649599090644,2357
      </coordinates>
    </LineString>
  </Placemark>
</Document>
</kml>
```

De koordinatene som pr. i dag ligger i eksempelet, angir en helikopterflyvning over Grand Canyon i Colorado, USA.

Eksempeldataene byttes ut med de aktuelle dataene ***og kodefila lagres under et ønsket filnavn som ender med .kml.***

Filen hentes opp i Google Earth ved å dobbelklikke på filnavnet. Siden fila ender på .kml, så skal den automatisk bli gjenkjent av Google Earth, starte programmet og laste inn datafilen.

En vil da få tegnet inn traseen som angitt av lista med koordinater og vist på riktig sted på jorda. Eksempelet under viser traseen til helikopteret over Grand Canyon (gul linje).



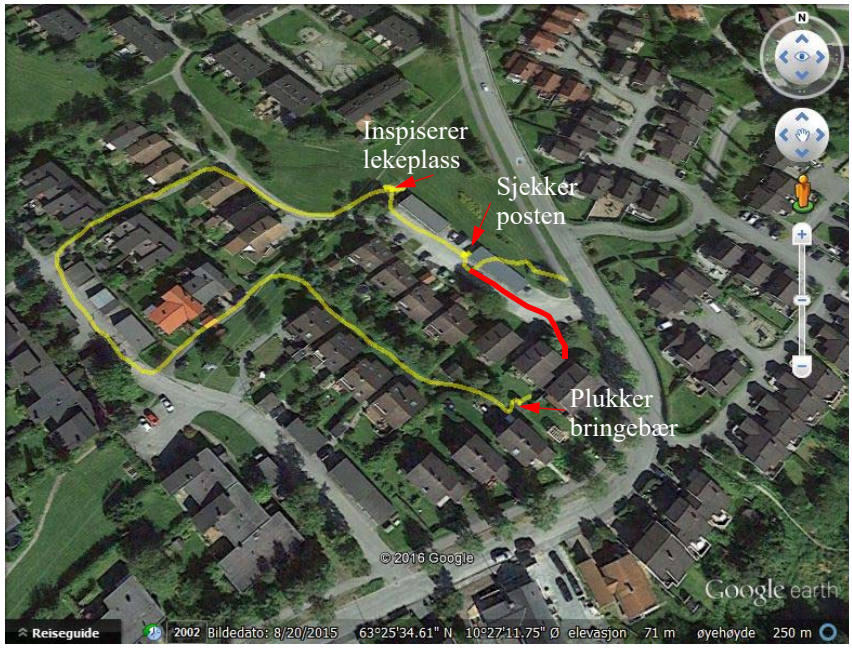
Dersom vi ønsker å vise hvor vi har gått en tur, så kan det være upraktisk å måtte vise absolutt høyde. Høydemålinger kan ha store avvik slik at vi kan oppleve at traseen går mange meter over eller forsvinner under bakken, til tross for at man hadde begge beina på jorda. I slike situasjoner kan det være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

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

med kommandoen:

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

så vil kurven følge bakken som vist på traseen på bildet under.



Vi legger imidlertid merke til at mottakeren har problemer i starten. I dette området er avviket stort før den tar seg inn. Den røde kurven angir den riktige ruta. Deretter er den svært nøyaktig. Posisjonsdata ble samplet hvert 3. sekund på denne turen.

10.4.3 Editering av kml-fila

Hvilket program skal vi så bruke for å legge inn koordinater og høyde? Et nyttig editeringsverktøy til dette formålet er Notepad++. Dette programmet er en avansert teksteditor som ikke gjør noen endringer med filformatet såfremt man ikke ønsker det. Notepad++ kan lastes ned fra:

https://notepad-plus-plus.org/downloads/

For å forenkle prosessen med å klippe inn data i kml-koden, er det praktisk å skrive koordinat- og høydedata i det formatet som programmet ønsker: Lengdegrad, breddegrad, høyde. Husk å bruke punktum som desimalpunkt og komma mellom hver verdi. ***NB! Det skal ikke være mellomrom etter kommaene.***

11 Referanser

[1] Haugen, F.A. og Lysaker, M., *Python for REALFAG*, Utgave 1 Fagbokforlaget , Bergen 2020.

[2] Rossing, N.K., *Lag en barometrisk høydemåler*, Fra fysikkens verden – 1/2026, Side 26 – 32.

[3] Grøn, Ø.G., *Relasjoner mellom trykk, høyde og temperatur i atmosfæren*, Fra fysikkens verden – 1/2026

Vedlegg A Komponentliste

A.1 Komponent- og bestillingsinformasjon

Tabellen under inneholder de fleste av komponentene som er brukt og hvor de kan skaffes fra. Prisen angir stykk-pris, men kjøpt i et visst antall som står i parentes. Prisene er ferskvare og vil ha endret seg siden denne lista ble oppdatert:

Nummer	Type komponent	Betegnelse	Verdi	Leverandør	Stykk pris eks. MVA
J1	Koblingsplint, Batterikontakt		2 skrukontakter	ELFA 300-94-873	Kr. 5,33 (10 stk)
J2	Kantkontakt Micro:bit			Kitronik	Kr. 24,07 (1 stk)
J3	Stiftlist m/strap	Header Pin single raw male 1x40 pin	2 x 2 stifter	ELFA 300-24-547	Kr. 8,36 (3x24stk)
J4	Stiftlist, Power supply Breakout		2 x 2 stifter		
J5	Stiftlist		1 x 19 stifter		
J6	QWIIC-connector		4 pin	ELFA 301-60-894	Kr. 7,42 (3 stk)
	Kortslutningsbøyler		2	ELFA 301-12-300	Kr. 2,37 (10 stk)
U1	SD-kort terminal	OpenLog		ELFA 301-73-922	Kr. 207,0 (5 stk)
U2	3,3V regulator	LD1117S33TR	3,3V	ELFA 301-70-563	Kr. 5,51 (5 stk)
U3	Prototyp areal	-	-	-	-
U4	Mini koblingsbrett			ELFA 301-15-118	Kr. 9,35 (2 stk)
U5	5,0V regulator	LD1117S50TR	5,0V	ELFA 301-70-565	Kr. 5,47 (5 stk)
U6	Sensor (Trykk, Fukt og temp)	BME280	3,3V	AliExpres	Kr. 43,44 (1 stk)
U7	Radiomodul	E32 433T20D		AliExpress	Kr. 67,69 (1 stk)
	Hylsekontakt (BME280)	Header Female	6	ELFA 148-78-295	Kr. 10,80 (3 stk)
	Hylsekontakt (E32 433T20D)	Header Female	7	ELFA 148-78-303	11,41 (10 stk)
	Hylsekontakt (E32 433T20D)	Header Female	3	ELFA 148-78-261	5,37 (10 stk)
	Hylsekontakt (OpenLog)	Header Female	6	ELFA 148-78-295	Kr. 10,80 (3 stk)

	Hylsekontakt (Micro:bit)	Header Female (alternativ til siftlist)	16+3 = 19	ELFA 148-78-386 148-78-261	Kr. 14,80 + Kr. 5,37 (10 stk)
D1	Schottky-diode, Batteri beskyttelse	IN5817	10mA, 20V, 300 mV	ELFA 300-31-456	Kr. 0,94 (25 stk)
SW1	Power Switch, en polt, glide	Skyvebrytere, 1CO, PÅ-PÅ, Hullmontering		ELFA 301-61-239	Kr. 14,20 (10 stk)
TH1	Termistor_NTC	NTCLE203E3103FB0	10 k	ELFA 301-53-871	Kr. 10,10 (10 stk)
R1	Motstand (NTC seriemotstand)		10 k	ELFA? 300-88-533	1 x Kr. 0,4312 (100 stk)
R2	Motstand (Pull up)		10 k	ELFA? 300-88-533	1 x Kr. 0,4312 (100 stk)
R3	Motstand (Pull up)		10 k	ELFA? 300-88-533	1 x Kr. 0,4312 (100 stk)
C1	Keramisk kondensator		100nF (beinavst. 5,08 mm)	ELFA 165-71-681	Kr. 3,01 (10 stk)
C2	Elektrolytt kondensator		10uF 25V (beinavst. 1,5mm)	ELFA 301-07-927	Kr. 2,48 (10 stk)
C3	Keramisk kondensator		100nF (beinavst. 5,08 mm)	ELFA 165-71-681	Kr. 3,01 (10 stk)
C4	Elektrolytt kondensator		10uF 25V	ELFA 301-07-927	Kr. 2,48 (10 stk)
	Batterikontakt		9V	ELFA 169-14-329	Kr. 4,83 (10 stk)
	Minnekort		16Gbyte	ELFA 300-19-838	Kr. 29,90 (10 stk)
	Mini koblingsbrett			ELFA 301-15-118	Kr. 9,75 (2 stk)
	Kretskort			ASE[a]	0,-

a. Andøya Space Education

A.2 Kostnadsoverslag bygget på priser høsten 2023

Dersom vi regner sammen prisen på enkelt komponenter fra lista foran, så ender vi opp med en kostnad pr sett på kr. 612,- (eks. MVA) som er stykkprisen ved innkjøp til 10 CanSat-sett. I tillegg kommer en Micro:bit v2 for hver CanSat, antenne, boks og batteri. Vi trenger også en ekstra CanSat for å ta imot data ved PC'en (bakkestasjon). Alternativt kan vi ha bruk en FTDI som ikke er tatt med i regnstykket foran.

En vi se at en vesentlig del av kostnaden er SD-terminalen OpenLog. Denne kan man droppe eller man kan kjøpe den fra f.eks. AliExpress. Man må imidlertid være klar over at ikke alle OpenLog kjøpt fra AliExpress lar seg skrive til uten videre.

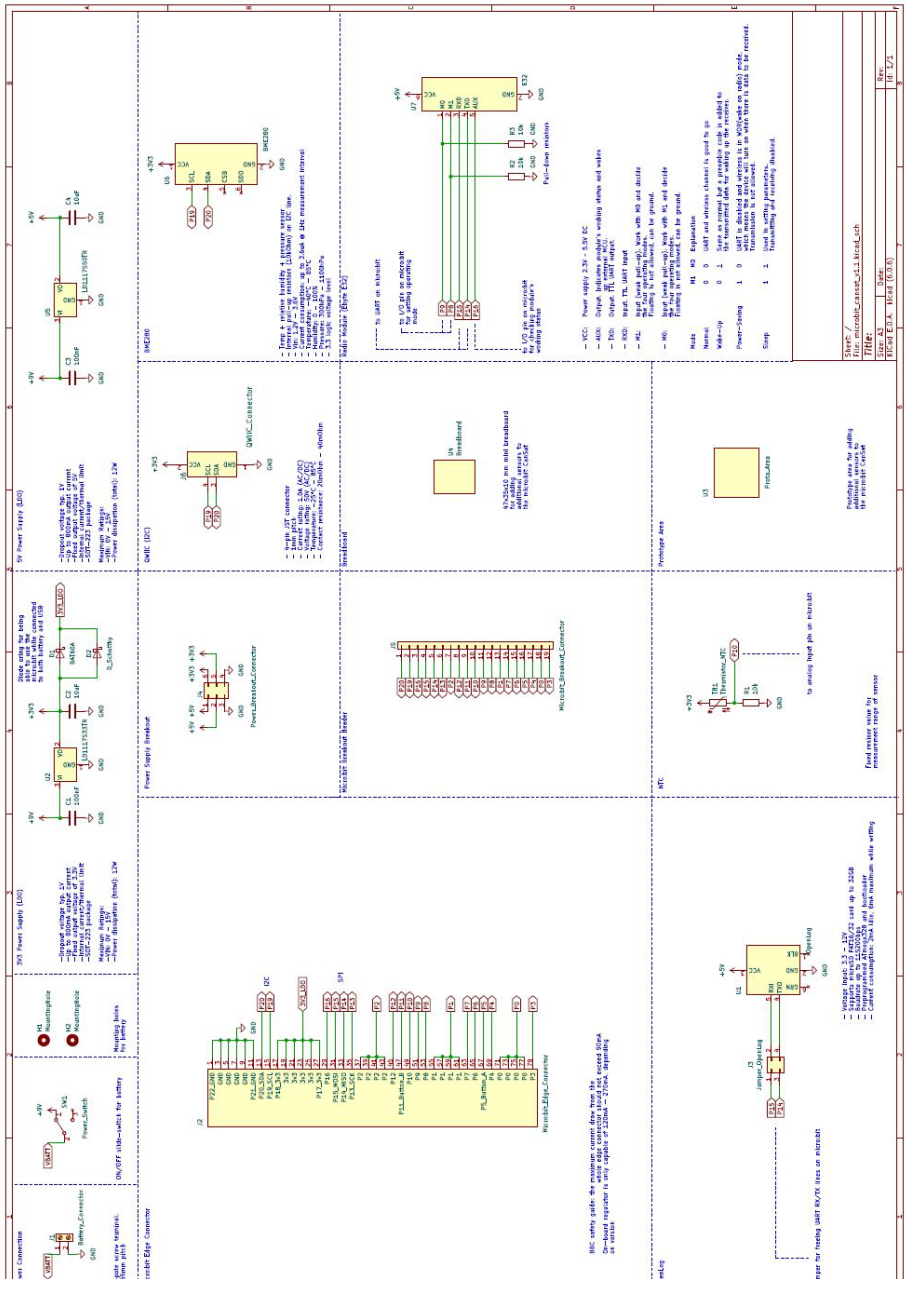
A.3 Oversikt over nettadresse til mulige leverandører av komponenter

Lista gir nettadresse til mulige leverandører:

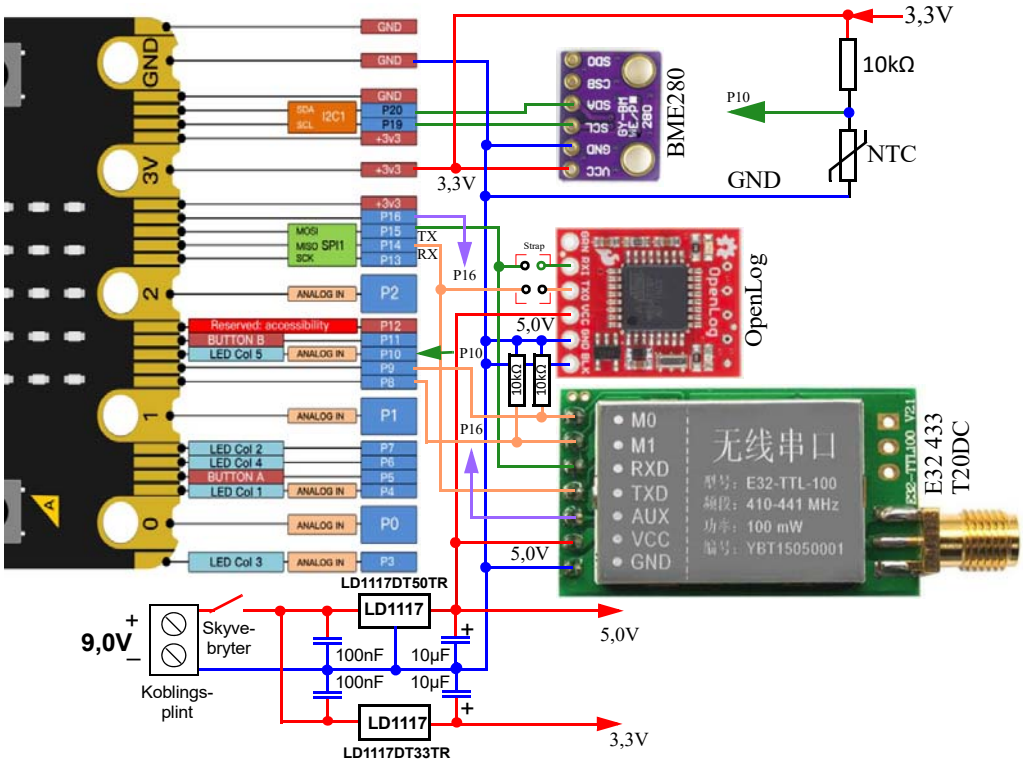
- LD1117DT50TR – https://www.elfadistrelec.no/no/ldo-spenningsregulator-5v-800ma-sot-223-st-ld1117s50tr/p/30170565
- LD1117DT33TR – https://www.elfadistrelec.no/no/ldo-spenningsregulator-3v-800ma-dpak-st-ld1117dt33tr/p/17305387
- E32 433T20DC – https://www.aliexpress.com/item/1005002668147892.html
- OpenLog – https://www.aliexpress.com/item/1005003739790712.html
- BME280 – https://www.aliexpress.com/item/1005001621866431.html
- NTC – https://www.elfadistrelec.no/no/ntc-termistor-ntc-10kohm-3977k-40mm-vishay-ntcle203e3103gb0/p/30153872
- Micro:bit kantkontakt – https://kitronik.co.uk/products/4148-edge-connector-header-for-bbc-microbit
- Motstander – 5 x 10kΩ for hullmontasje – https://www.elfadistrelec.no/no/metalloksidfilm-motstand-250mw-10kohm-rnd-components-rnd-155mor0w4j0103a50/p/30088533
- El.lytt kondensator – 2 x 10µF 16 V – https://www.elfadistrelec.no/no/radiell-elektrolyttkondensator-10uf-3ua-16v-28ma-rnd-components-rnd-150ksm016m100c07s/p/30146141
- Keramiske kondensatorer 1 x 100nF – https://www.elfadistrelec.no/no/keramisk-kondensator-100pf-100v-05-rnd-components-rnd-150mnpo101j2ar15ds706/p/30128736
- Koblingsplint – 2 x 2 polt – https://www.elfadistrelec.no/Web/Downloads/_t/ds/RND%20205-EK254-2.54_eng_tds.pdf
- En polt skyvebryter – https://www.elfadistrelec.no/no/skyvebrytere-1co-pa-pa-hullmontert-rnd-components-rnd-210-00662/p/30161239

Vedlegg B Kretsskjema, blokkdiagram og kretskort

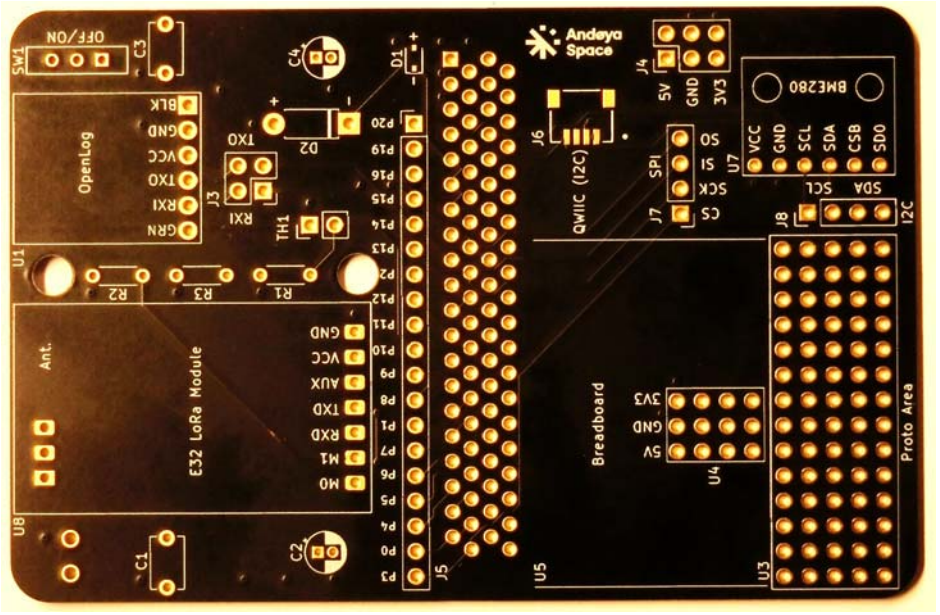
Figuren under viser kretsskjemaet som er tegnet og lagt ut av Kristian Enoksen (ASE).



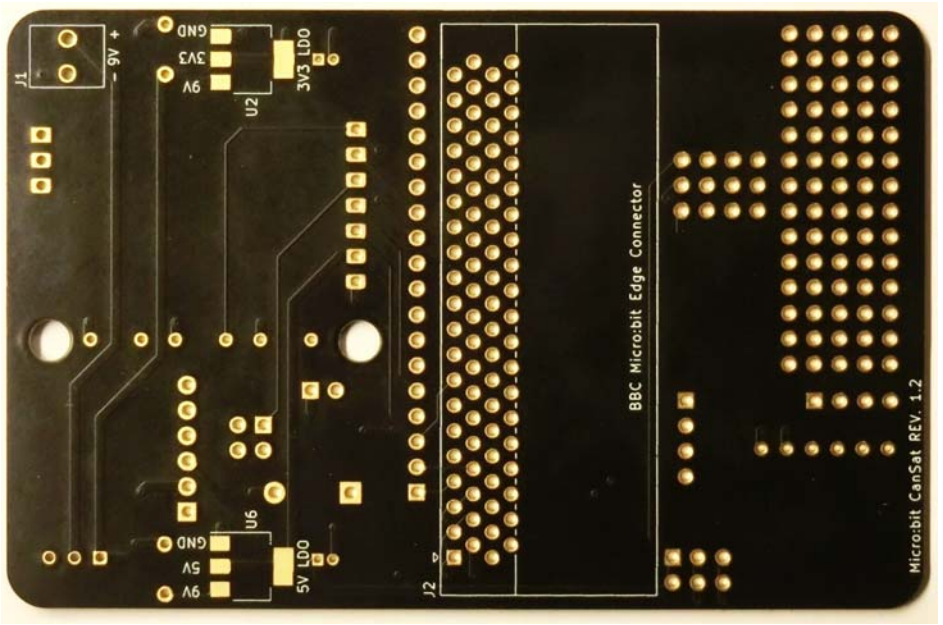
Figuren under vise blokkdiagramet for Micro:bit CanSat.



Figuren under vise det endelige kretskortet.



Kretskort – komponentside



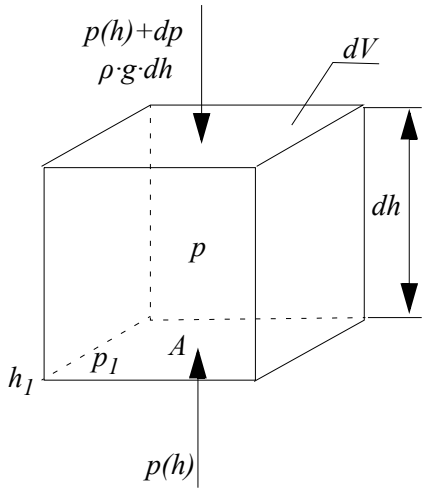
Kretskort – loddeside

Vedlegg C Barometrisk høydemåling

I dette vedlegget skal vi første se på en utledning av den barometriske formelen for beregning av høyde som funksjon av lufttrykk. Deretter skal vi se på en alternativ formel for å beregne høyde som funksjon av lufttrykk for så å sammenligne de to formlene.

C.1 Utledning av den barometriske ligningen

Denne utledningen bygger på professor emeritus Øyvind G. Grøn ved universitetet i Oslo artikkel i "Fra fysikkens verden", nr. 1, 2024 [3].



Han tar utgangspunkt i en luftsøyle som vist på figuren til høyre, med et tverrsnittsareal på A. Vi tar ut en skive av luftsøylen med høyde dh. Lufta inne i skiva har tetthet ρ.

På grunn av at lufta inne i skiva har en vekt, så vil vekta av lufta trykke mot arealet A nederst i volumet av skiva, være litt større enn øverst i volumet av skiva, selv om arealet er det samme. Forskjellen i lufttrykk over tykkelsen av skiva, er dp og vi kan sette opp følgende sammenheng:

$$dp = -\rho g dh \tag{C.1}$$

der ρ er tettheten av lufta, g er tyngdeakselerasjon og dh er tykkelsen av skiva.

Tettheten ρ av luft kan uttrykkes ved hjelp av tilstandsligningen for ideelle gasser. Tettheten vil være proporsjonal av trykket p, og omvendt proporsjonal med temperaturen T, og proporsjonaliteten uttrykkes ved hjelp av den spesifikke gasskonstanten R_s. Tilstandsligningen for ideelle gasser kan uttrykkes slik:

$$\rho = \frac{p}{R_s T} \tag{C.2}$$

Dersom vi setter inn uttrykket for ρ fra ligning (C.1) i ligning (C.2) får vi det som kalles den *hydrostatiske* ligningen.

$$\frac{dp}{p} = \frac{g}{R_s T} dh \tag{C.3}$$

Vi ser at den absolutte temperaturen, T, inngår i uttrykket, og er i denne sammenhengen konstant. Vi vet imidlertid at temperaturen endres som funksjon av høyden. Vi må derfor etter hvert finne et uttrykk for temperaturen som funksjon av høyden.

Dersom vi likevel, inntil videre, antar at temperaturen er den samme, uavhengig av høyden, og løser ligning (C.3) ved å integrere fra en referansehøyde, h_1, og opp til en vilkårlig høyde h, vil vi få:

$$h = h_1 - \frac{RT}{g} \ln \frac{p}{p_1} \tag{C.4}$$

Denne formelen kalles den *hypsometriske formelen* for sammenhengen mellom trykk og høyde i atmosfæren, antatt at temperaturen er konstant mht. høyden, hvilket er temmelig urealistisk. Det er derfor nødvendig å finne en mer realistisk sammenheng mellom høyde og temperatur. En akseptabel antagelse for nedre del av atmosfæren er at temperaturen faller lineært med høyden. Man har funnet ut at en rimelig antakelse er at temperaturen faller med en gradient a, tilnærmet lik 0,0065K/m. Vi kan da uttrykke temperaturen som funksjon av høyden h på følgende måte:

$$T(h) = T_1 - a(h - h_1) \tag{C.5}$$

Hvor T_1 er temperaturen ved referansehøyden h_1.

Vi setter denne sammenhengen inn i den hydrostatiske ligningen (C.3) og får uttrykket:

$$\frac{dp}{p} = \frac{g}{R_s} \frac{dh}{a(h - h_1) - T_1} \tag{C.6}$$

Løser vi denne differensialligningen mht p/p_1, får vi:

$$\ln\left(\frac{p}{p_1}\right) = \ln\left[1 - \frac{a}{T_1}(h - h_1)\right]^{g/aR_s} \tag{C.7}$$

Løser vi denne ligningen mht. h får vi:

$$h = h_1 + \frac{T_1}{a}\left[1 - \left(\frac{p}{p_1}\right)^{aR_s/g}\right] \tag{C.8}$$

Denne formelen kalles *den barometrsike formelen*, eller også den *barometriske nivelerings-formelen*[44]. Her er a > 0.

I vår formelen har vi tatt utgangspunkt i at når temperaturen faller med økende høyde så ønsker vi at dette angis med a < 0. Siden formelen skal bli lik enten vi bruker negativ eller positiv a, må vi bytte ut a med –a.

Dvs. vi må skifte fortegne foran alle a'ene i ligning (C.8).

44.https://no.frwiki.wiki/wiki/formule_du_nivellement_barom%C3%A9trique

$$h = h_1 + \frac{T_1}{-a}\left[1 - \left(\frac{p}{p_1}\right)^{(-a)R_s/g}\right] \quad \text{(C.9)}$$

Omorganiserer vi denne ligningen får vi nettopp vår ligning:

$$h = \frac{T_1}{a}\left[\left(\frac{p}{p_1}\right)^{(-a)R_s/g} - 1\right] - h_1 \quad \text{(C.10)}$$

C.2 Barometrisk høydemåling – to formler

Vi har i vedlegg C.1, side 133 sett at den barometriske ligningen for beregning av høyden ut fra lufttrykket (lign. C.11), er relativt komplisert:

$$h = \frac{T_1}{a}\left(\left(\frac{p}{p_1}\right)^{-\frac{aR}{g_0}} - 1\right) + h_1 \quad \text{(C.11)}$$

Hvor:

h	Beregnet høyde i meter
h_1	Starthøyde i meter
T_1	Starttemperatur i høyden h_1
a	Temperaturgradient, foreslått verdi -0,0065 K/m
p	Målt trykk i Pa
p_1	Trykk i Pa ved starthøyden
g_0	Tyngdeakselerasjonen 9,81 m/s^2
R	Den spesifikke gasskonstant 287,06 J/kg K

Det finnes alternative omregningsformler[45] som er noe enklere. Men også her støter vi på problemer dersom vi ønsker å bruke ren blokkode:

$$h = \ln(p/p_1) \cdot (-18012{,}4) + h_1 \quad \text{(C.12)}$$

Betydningen av parameterne er de samme som i lign. (C.11).

Som vi ser så har man antatt at temperaturen betyr lite i nedre luftlag og dermed latt den inngår i konstanten. I den enkle formelen har vi lagt inn en korreksjonfaktor k som gjør at vi kan få de to kurvene mest mulig like. Omregningsformelen blir da:

$$h = k \cdot \ln(p/p_1) \cdot (-18012{,}4) + h_1 \quad \text{(C.13)}$$

45. Beregnet av Stian Wannebo, Vitensenteret i Trondheim

Det kan være av interesse å sammenligne de to omregningsformlene, både med og uten korreksjon. I tabellen til venstre, på figuren under, er de to omregningsformlene sammenlignet uten korreksjon (k). Kolonnen til venstre er ligning (C.11) og kolonnen til høyre er ligning (C.13). Tabellen til høyre viser de samme beregningene, men nå er den forenklede omregningsformelen multiplisert med en faktor k = 1,0865. Vi ser at de to tallrekkene blir nesten like.

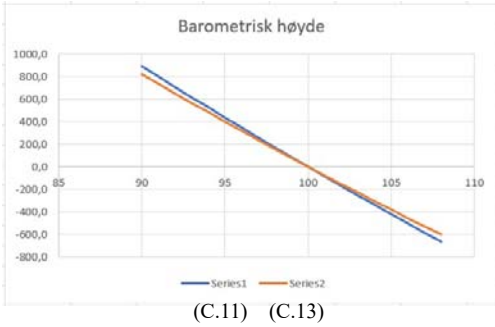
Lufttrykk i kPa	Høyde (m)	Høyde (m)			
90	894,3	824,2	Starttemperatur i K (T1)	293	K
91	801,4	737,8	Starthøyde i m (h1)	0	m
92	709,3	652,3	Starttrykk i Pa (p1)	100	kPa
93	617,9	567,7	a (Temperaturgradient)	-0,0065	K/m
94	527,4	484,0	R Spesifikk gasskonstant)	287,06	J/K kg
95	437,6	401,3	g0 (Tyngdeakselerasjon)	9,81	m/sek2
96	348,6	319,3	k	1	
97	260,4	238,3			
98	172,9	158,0			
99	86,1	78,6			
100	0,0	0,0			
101	-85,4	-77,8			
102	-170,1	-154,9			
103	-254,1	-231,2			
104	-337,5	-306,8			
105	-420,3	-381,7			
106	-502,4	-455,8			
107	-583,8	-529,3			
108	-664,7	-602,0			

(C.11) (C.13)

Lufttrykk i kPa	Høyde (m)	Høyde (m)			
90	894,3	895,5	Starttemperatur i K (T1)	293	K
91	801,4	801,6	Starthøyde i m (h1)	0	m
92	709,3	708,7	Starttrykk i Pa (p1)	100	kPa
93	617,9	616,8	a (Temperaturgradient)	-0,0065	K/m
94	527,4	525,9	R Spesifikk gasskonstant)	287,06	J/K kg
95	437,6	436,0	g0 (Tyngdeakselerasjon)	9,81	m/sek2
96	348,6	347,0	k	1,0865	
97	260,4	258,9			
98	172,9	171,7			
99	86,1	85,4			
100	0,0	0,0			
101	-85,4	-84,6			
102	-170,1	-168,3			
103	-254,1	-251,2			
104	-337,5	-333,4			
105	-420,3	-414,7			
106	-502,4	-495,2			
107	-583,8	-575,1			
108	-664,7	-654,1			

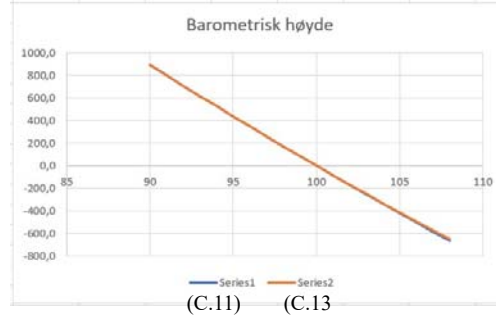
(C.11) (C.13)

I figuren under er de samme resultatene vist som grafer.



(C.11) (C.13)

Uten korreksjonsfaktor k



(C.11) (C.13

Med korreksjonsfaktor k

Selv om ligning (C.13) er vesentlig enklere enn (C.11) så er den ikke noe enklere dersom vi insisterer på å bruke blokkode. Bruker vi derimot Java eller Python blir det hipp som happ hvilken omregningsformel vi bruker siden disse språkene har et betydelig bedre utbygd matematisk bibliotek.

Vedlegg D Temperaturmåling med NTC – Linearisering og bruk av topunktsligningen

D.1 Modellering av NTC-motstand som funksjon av temperaturen

En forenklet sammenheng mellom resistansen (R) og temperaturen (T) kan uttrykkes som:

$$R = Ae^{B/T} \tag{D.1}$$

hvor A og B er "konstanter" bestemt av materialet og temperaturen. Konstantene kan likevel betraktes som *tilnærmet konstante* innen begrensede temperaturområder.

I datablader for NTC-motstander oppgis gjerne resistansen (R_r) for en referansetemperatur (T_r). I et temperaturområde rundt denne referansetemperaturen antas B-verdien å være tilnærmet konstant ($B_{25/85}$ – B-verdien er tilnærmet konstant innen området 25°C til 85°C).

Vi kan da sette opp følgende to ligninger:

$$R = Ae^{\frac{B_{25/85}}{T}} \tag{D.2}$$

$$R_r = Ae^{\frac{B_{25/85}}{T_r}} \tag{D.3}$$

Ved å eliminere A fra disse uttrykkene, kommer vi fram til følgende sammenheng, løst med hensyn til resistansen R:

$$R = R_r \cdot e^{\left(\frac{B_{25/85}}{T} - \frac{B_{25/85}}{T_r}\right)} \tag{D.4}$$

Dette uttrykket går under betegnelsen *Beta-formelen*.

Når vi skal beregne verdien for en NTC-motstand ved en gitt temperatur, slår vi opp B-verdien, R_r og T_r i databladet, sørger for at de aktuelle temperaturene ligger innenfor området til B-verdien, og beregner R ved å sette inn ønsket temperatur T. Temperaturen angis i grader Kelvin.

D.2 NTCLE201E3103S

Fra databladet[46] for *NTCLE201E3103S* finner vi følgende: R_{25} er referansemotstand (R_r) ved 25 °C (T_r = 298 K):

ELECTRICAL DATA AND ORDERING INFORMATION				
R_{25} (Ω)	R_{25}-TOL. (± %)	$B_{25/85}$ (K)	$B_{25/85}$-TOL. (± %)	SAP MATERIAL AND ORDERING NUMBER
3000	2.18	3977	0.75	NTCLE201E3302SB
5000	2.18	3977	0.75	NTCLE201E3502SB
10 000	2.18	3977	0.75	NTCLE201E3103SB
3000	2.18	3977	0.75	NTCLE300E3302SB
5000	2.18	3977	0.75	NTCLE300E3502SB
10 000	2.18	3977	0.75	NTCLE300E3103SB

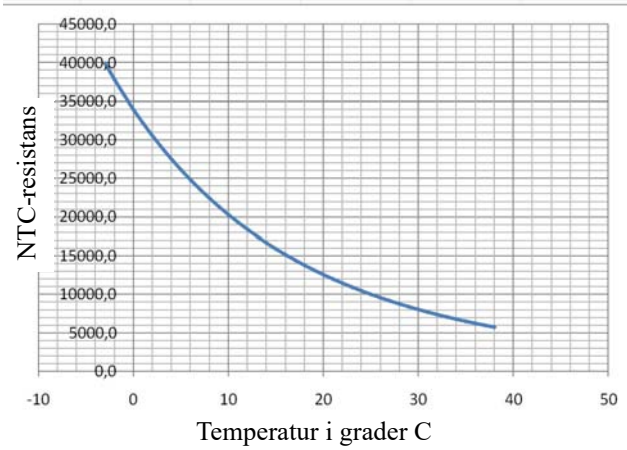
Figur D.1 Datablad for NTC-motstand NTCLE201E3103SB, 3–10 kΩ

Med disse dataene kan vi skrive:

$$R = 10\text{k} \cdot e^{\left(\frac{3977}{T} - \frac{3977}{298}\right)} \tag{D.5}$$

hvor $B_{25/85}$ = 3977 (NTCLE201E3103SB – 10 kΩ) og referansetemperaturen T_r = 298 K.

Dersom vi beregner verdier for R i temperaturområdet 25° – 85°C, får vi følgende graf:



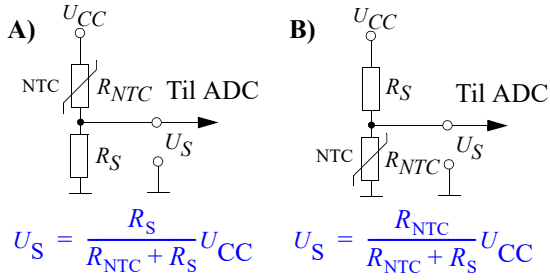
Figur D.2 NTC motstand som funksjon av temperaturen NTCLE201E3103SB – 10 kΩ

En annen viktig parameter for NTC-motstander, er hvor raskt resistansen endrer seg med temperaturen. Denne parameteren betegnes *NTC-motstandens tidskonstant* (τ), og angir den tiden det tar for resistansen og endre seg til 63,2% av den nye resistansen etter at temperaturen har endret seg 1 K (Kelvin) over omgivelsestemperaturen. En antar at temperaturendringen ikke er forårsaket av indre oppvarming på grunn av elektrisk strøm som flyter gjennom motstanden.

46. Databladet er hentet fra: http://www.elfa.se/pdf/60/06027916.pdf

Vi var ikke istand til å finne en verdi for tidskonstanten for den aktuelle NTC-motstanden. Men det er rimelig å anta at den er under 10 sek. avhengig av omgivelsene.

D.3 Oppkobling mot ADC



Siden grensnittet til kontrolleren krever en spenning, kobles NTC-motstanden i serie med en motstand som vist i figuren til høyre. Velg verdien på seriemotstanden lik den nominelle verdien på NTC-motstanden (R_{25}). Spenningsnivået, U_S, beregnes fra formlene som antydet på figuren. Legg merke til at oppkoblingen på tegning A gir økende spenning, U_S, med økende temperatur, mens oppkoblingen i tegning B gir fallende spenning, U_S, med økende temperatur.

I vårt tilfelle er NTC-motstanden plassert nærmest GND som vist i figur B over. Vi får da en fallende spenning som funksjon av økende temperatur.

D.4 Optimal seriemotstand

Vi har registrert at motstandsverdien til NTC-motstanden er svært ulineær. Nå er det ikke motstanden vi måler, men spenningen på utgangen av spenningsdeleren. Så la oss se hvordan spenningen avhenger av temperaturen og hvordan lineariteten varierer med verdien til seriemotstanden.

På bakgrunn av ligningene foran kan vi sette opp et uttrykk for temperaturen som funksjon av spenningen som evt. kan legges inn i programmet i mikrokontrolleren.

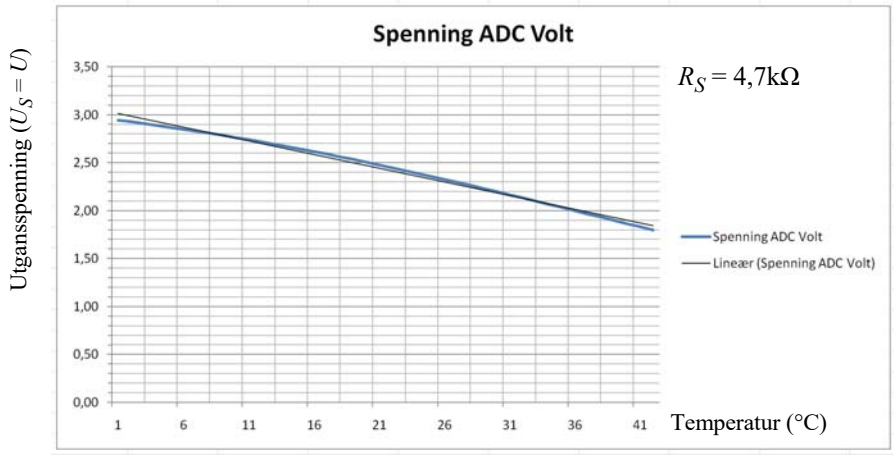
Vi setter:

$$R_{NTC} = 10\text{k} \cdot e^{\left(\frac{3977}{T} - \frac{3977}{298}\right)} \tag{D.6}$$

inn i ligningen:

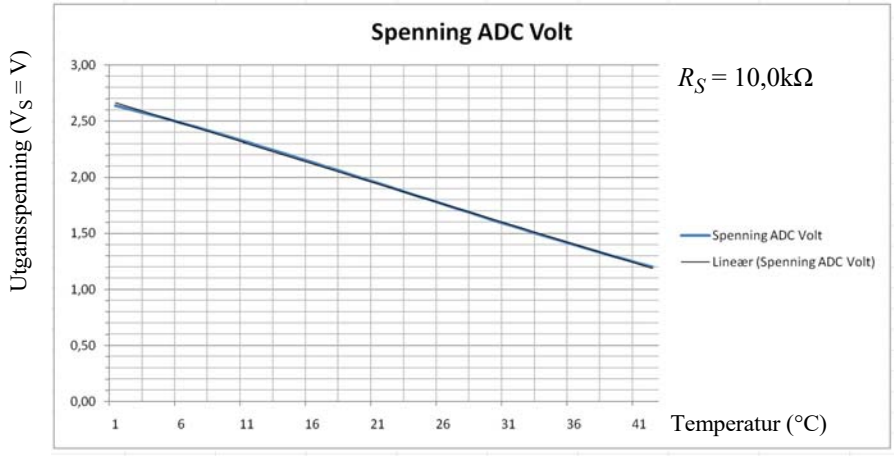
$$U_S = \frac{R_{NTC}}{R_{NTC} + R_S} U_{CC} \tag{D.7}$$

... og vi kan beregne U_S som funksjon av temperaturen for ulike verdier av seriemotstanden, R_S. I figuren under har vi modellert spenningen V_S som funksjon av temperaturen i området 0 – 42°C med en seriemotstand på R_S = 4,7 kΩ:



Sammen med spenningskurven har vi lagt inn den best tilpassede lineære sammenhengen (tynn rett linje). Vi legger merke til at avvikene er betydelig i endene av området. I temperaturområdet 0 – 42°C er spenningssvinget lik U_{diff} = 0,82 V.

Dersom vi imidlertid velger en seriemotstand R_S = 10kΩ ser kurven lang mer lineær ut.



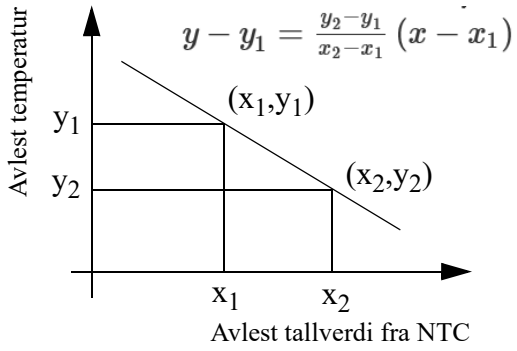
Vi legger merke til at avvikene ved endene av området er langt mindre med 10kΩ. Dessuten er spenningssvinget i området 0 – 42°C lik U_{diff} = 1,08 V. Dvs 10kΩ gjør ikke bare kurven mer lineær, men men vi utnytter det dynamiske området til ADC bedre.

Vi anbefaler derfor å bruke en seriemotstand på 10 kΩ for *NTCLE201E3103SB.*

Kalibrering av temperatursensoren

Denne beregningen antar at det er en omtrent lineær sammenheng mellom måleverdi og temperaturer i det aktuelle temperaturområdet. Dette er ikke langt fra sannheten i et avgrenset område (f.eks. 0 – 42°C). Under denne betingelsen kan følgende metode benyttes med tilstrekkelig nøyaktighet:

1. Bruk et termometer å mål "virkelig" temperatur f.eks. innendørs (y_1), les av tallverdien (gjerne den digitale verdien) som kommer fra NTC-motstanden (x_1).
2. Ta så Teensy'en med ut samtidig som du logger data (vi antar at det er kaldere ute). Mål "virkelig" temperatur med et termometer (y_2) og les av tallverdien fra CanSat (x_2).



Figuren over viser hvordan topunktsformelen kan brukes for å finne et uttrykk for sammenhengen mellom målte verdier og temperatur.

Topunktsformelen for lineære ligninger kan også skrives slik:

$$y = ((y_2 - y_1)/(x_2 - x_1)) * (x - x_1) + y_1 \quad \text{(D.8)}$$

Hvis $k = (y_2 - y_1)/x_2 - y_2)$ (stigningskoeffisienten), kan vi skrive ligningen slik:

$$y = k\,(x - x_1) + y_1 \quad \text{(D.9)}$$

Sett verdiene inn i formelen over og finn et uttrykk for y (temperatur i C) som funksjon av tallverdien hentet fra AD-konverteren som måler spenningen fra NTC-motstanden (x).

Vedlegg E Bruk av FTDI og terminalprogram

Vedlegget beskriver hvordan man istedet for å bruke en ekstra CanSat og en Micro:bit som bakkestasjon, kan bruke en FTDI direkte koblet til radioen og USB-porten til PC-en.

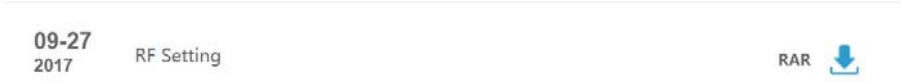
E.1 Konfigurer og test radioene ved hjelp av program fra leverandøren av E32

Avsnittet beskriver hvordan vi setter opp radioen ved hjelp av programmet RF-setting fra leverandøren av radioen E32. Vi kobler radioen til USB-kontakten på PC'en ved hjelp av en FTDI-krets[47] (overgang mellom radioen og USB'en på PC'en). Vi trenger derfor ingen ekstra CanSat for å ta imot og overfør data ved bakkestasjonen, det holder med en radio av typen E32 og en FTDI. Dette er derfor en vesentlig billigere løsning enn ved bruk av en ekstra CanSat.

1. **Last ned programmet: "RF Setting"**
   Gå til nettsiden:

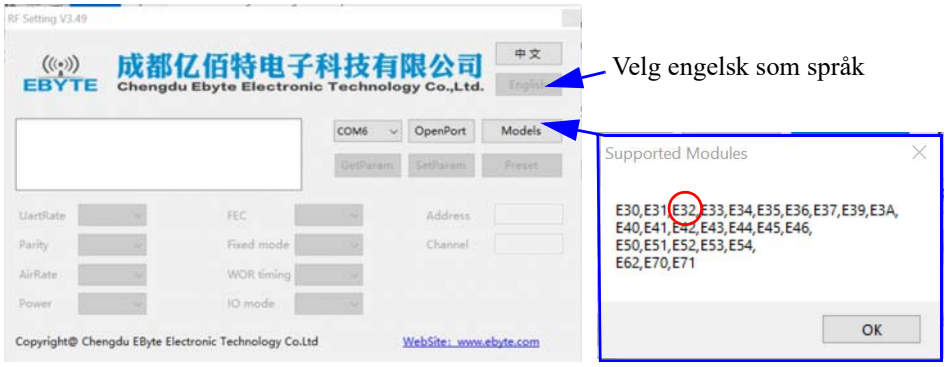
   https://www.ebyte.com/en/data-download.html?page=3&id=199&cid=31#load

   og bla ned til ark 3:

   Last ned, pakk ut og kjør programmet fra oppsett av radioen.

   Før vi kobler radiomodemet til USB-porten, vil vi få opp et brukergrensesnitt omtrent som på bildet under:

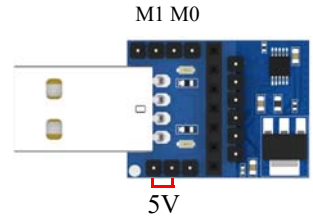
   

   Sørg for å velg engelsk tekst.

47. Er en krets som knytter sammen UART'en (Universal Asyncronous Rx/Tx) hos Micro:biten med USB'en hos PC'en slik at data kan mottas, vises eller lagres på PC'en. FTDI står for "Future Technology Devices International Limited" som er et lite skotsk firma som har spesialisert seg på denne typen seriekommunikasjon. Kretsen kan kjøpes hos AliExpress: https://www.aliexpress.com/item/1005002355529299.html

Trykker vi på *Models* så får vi opp informasjonen som vist over til høyre, som forteller oss at vi kan bruke dette programmet til å programmere radioen E32. Ellers er det lite som kommer opp i de enkelte rubrikkene før vi kobler radiomodemet til USB-porten.

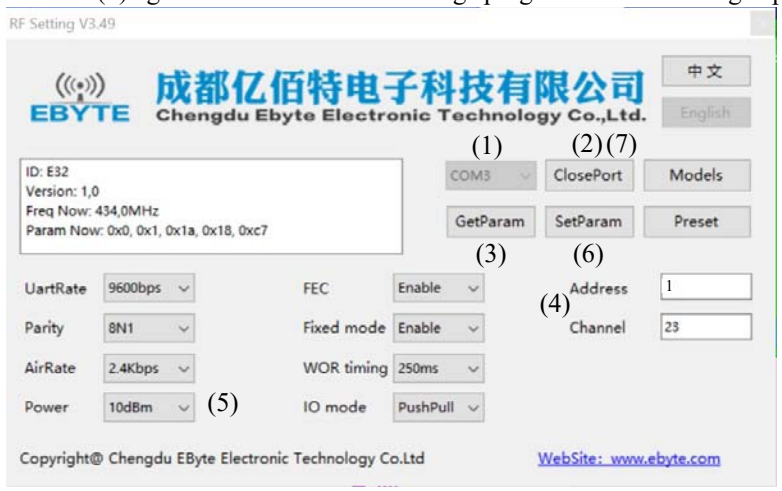
2. **Oppkobling av radioen E32 til USB-porten til PC-en**
Vi kobler E32 til USB-porten ved hjelp av FTDI-kretsen som er skreddersydd for radioen E32.



Vi setter opp kretsen ved hjelp av en strapp (5V) slik at den får spenningen 5V. M0 og M1 skal settes til "1" ved å *unnlate å sette opp strapper*. Med M0 = M1 = "1" gjør det mulig å sette og lese av konfigurasjonen til radioen[48].

Vi plugger i USB-kontakten og kan nå lese av og sette konfigurasjonen til radioen.

3. **Konfigurer radioene**
Når radioen er koblet opp kan vi velge riktig port (1). Deretter velger vi *OpenPort* (2) og deretter *GetParam* (3) og vi får et bilde av "RF Setting" programmet som kan ligne på følgende:



Vi velger å sette adressen til 1 og kanalen til 23. Det er viktig å velge en adresse og en kanal som er forskjellig fra de andre som befinner seg i nærheten (4), men lik for den vi ønsker å kommunisere med.

Vi velger å sette effekten **Power** til ønsket nivå, i vårt eksempel, 10 dBm[49] (10mW). Dersom vi skal bruke radioen utendørs kan vi øke effekten til f.eks. 20dBm (100mW) (5).

De nye parametrene overføres til radiomodemet ved å velge **SetParam (6).** Før vi kobler fra kretsen velger vi **ClosePort (7)**.

Så gjør vi akkurat det samme med den andre radioen som skal stå ved bakkestasjonen.

48. Ved å sette på begge strappene, for M0 og M1, så vil M0=M1="0", vil radioen oppfører seg transparent og sender data over radiokanalen på vanlig måte.
49. dBm er et antall dB relativt til 1mW

4. **Lag testprogrammet**

Vi skriver et enkelt testprogram for å teste kommunikasjonen. Denne gangen vil vi sende data til mottakeren som er koblet direkte til USB-inngangen til PC'en ved hjelp av FTDI-kretsen. Denne gangen må vi sette opp FTDI-kretsen slik at M0 og M1 legges til "0", dvs. normal sending og mottaking. Det gjør vi med strappene M0 og M1 som vist på figuren til høyre og koble radioen med antenne til FTDI-kretsen som vist på figuren over.

Vi trenger ikke å sette opp radioen på nytt, men må fortelle programmet hvordan radioen er koblet til Micro:bit-en, det gjør vi i *"ved start"*-blokken:

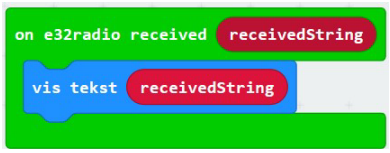
Vi ønsker å bruke radioene til å sende data fra CanSat-en til bakkestasjonen og setter derfor CONFIGURATION MODE = "usann" for normal kommunikasjon.

Dernest vil vi sende en enkel melding når vi trykker på knapp A:

NB! *Pass på at adressen i kommandoen over er i overensstemmelse med adresse og kanal hos sender og mottaker.*

Videre vil vi se hvordan radioen er konfigurert når vi trykker på knapp B:

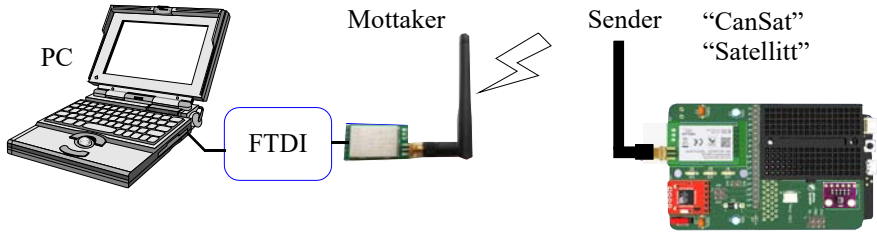
og vise evt. mottatte meldinger på displayet:



5. **Kjør testprogrammet**
   Skriv test-programmet, gå sammen med en annen og test sambandet begge veier.

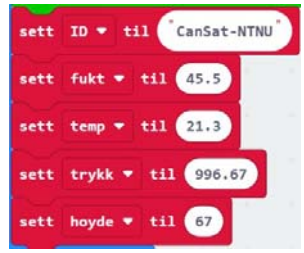
E.2 Send data med CanSat og motta med radio tilkoblet PC'en med FTDI-krets

Figuren under illustrerer systemet med to radioer, sender- og mottaker og bruk av FTDI:

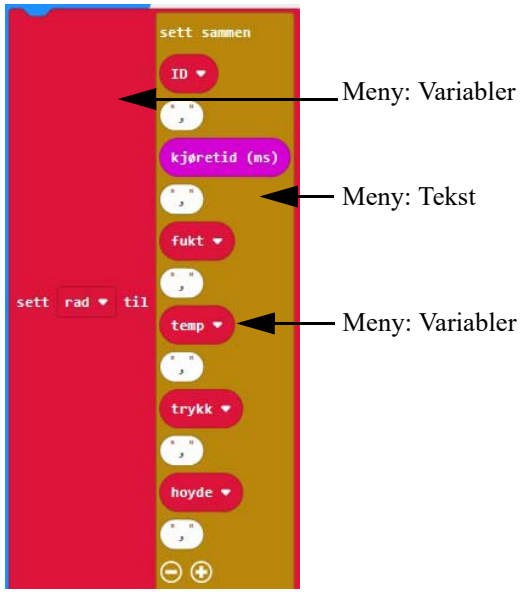


Først kan vi for enkelthetsskyld lage fiktive data som vi sender over til bakkestasjonen før vi skifter ut disse med virkelige måledata.

1. **Deklarasjon av variabler og fiktive data**
   Først definerer vi oppkoblingen til Micro:bit-en for så å deklarere og tilordne verdier til følgende variabler:



2. **Kommaseparert rad med fiktive data**
I tillegg deklarerer vi variabelen *rad* som skal holde raden med kommaseparerte måleverdier. Dette gjør vi under blokken **"***gjenta for alltid"*. Variabelen *rad* bygger vi opp som vist på figuren under:



Det er viktig å ha med det siste kommaet. Det viser seg at dette gjør lesingen med Python enklere.

3. **Send variabelen *rad***
Raden sender vi med blokkommandoen fra LoRaES32 menyen:.

Vi velger å bruke blokken der vi kan spesifisere adresse og kanal. Husk å legg inn en forsinkelse før neste overføring av data.

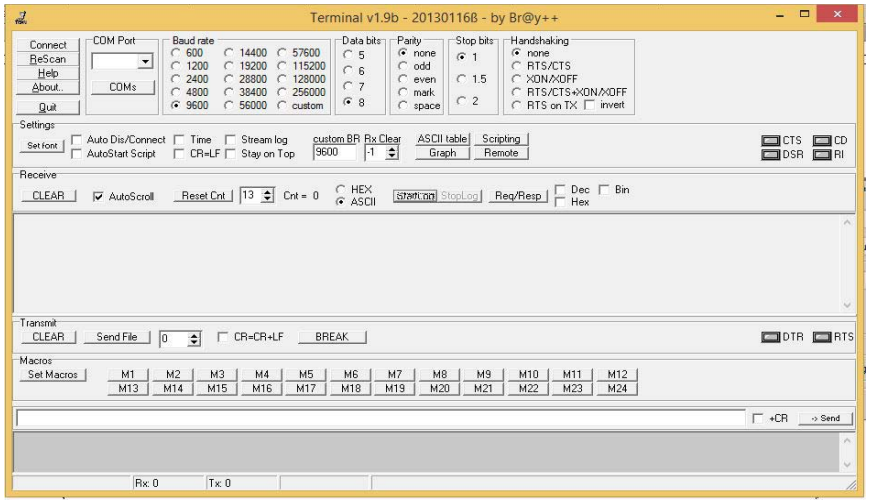
4. **Send linjeskift**
Det viser seg at det sendes intet linjeskift etter at tekststrengen "rad" er sendt, som medfører at alt kommer ut på samme linje hos terminalprogrammet hos PC'en. En enkel måte å få lagt til linjeskift på er å legge inn en ekstra tekstmelding med kun et mellomrom som inkluderer linjeskift. Husk å legge inn et delay mellom de to sendingene for å unngå feilmelding.

5. **Presentasjon og lagring av data på PC**
I dette eksempelet velger vi å motta og presentere dataene med et terminal-program.

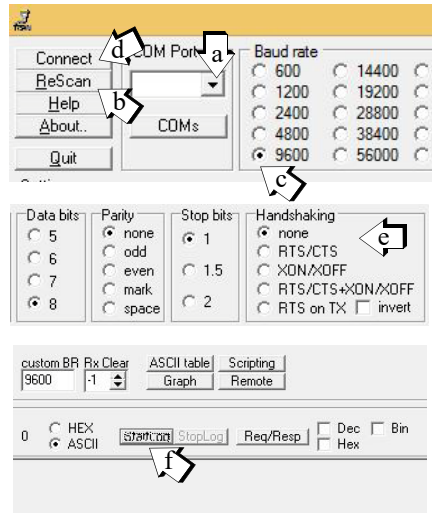
Vi har valgt å bruke terminalprogrammet: *Terminal.* Last ned terminal-programmet:

https://sites.google.com/site/terminalbpp/

Programmet *Terminal* trenger ikke å installeres, men kan kjøres direkte fra exe-filen. Programmet startes ved å klikke på programikonet, og man får opp et vindu som vist på figuren under.

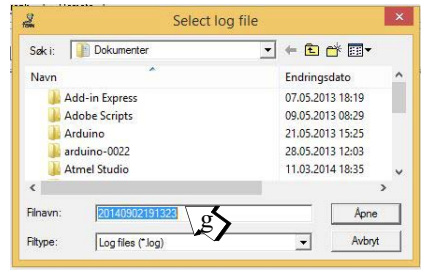


Oppsett av programmet:

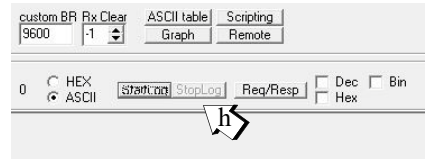


- *Velg COM-porten til radioen (a)*
  Om du ikke finner den rette porten, prøv å trykke *"ReScan"* (b). Da vil alle tilgjengelige porter leses inn på nytt.
- *Velg datahastighet*
  Velg 9600 Baud fra listen (c)
- *Opprett kontakt*
  Trykk "*Connect"* (d) for å koble opp forbindelsen mot COM-porten.
- *Oppsett av kommunikasjon (e)*
  Oppsett av kommunikasjonen gjøres ved å velge antall "*Data bits"*, "*Parity"* (none), "*Stop bit"* og "*Handshaking"* (none). Sett valgene som vist på figuren til høyre.

Lagring av data
Trykk "*StartLog"* (f) for å starte logging av data. Du vil da få spørsmål om å oppgi et filnavn (g). Oppgi filnavn og katalog og trykk "*Åpne"*.

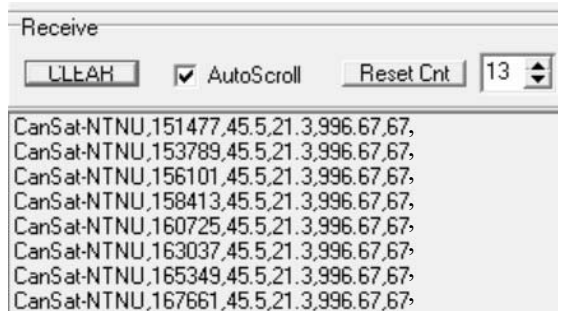


Avslutt lagring (h)
Trykk *StopLog* for å avslutte logging av data.



Filformat
Dersom dataene er hensiktsmessig organisert (CSV-file), kan filen leses rett inn i Excel eller Python..

6. **Visning av fiktive data**
Dersom vi er heldige kan resultatet i terminalprogrammet bli omtrent som vist på skjerm-bildet til høyre. Siden vi bruker fiktive data så er de temmelig statiske.



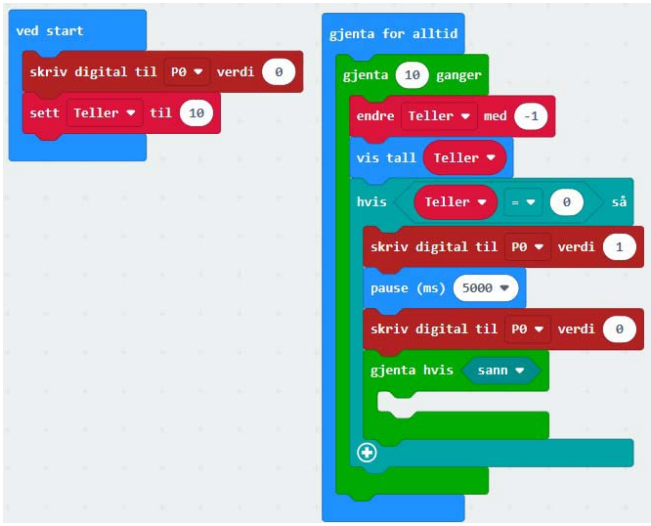
Vedlegg F Løsningsforslag

F.1 Løsningsforslag grunnkurs 1 - 8

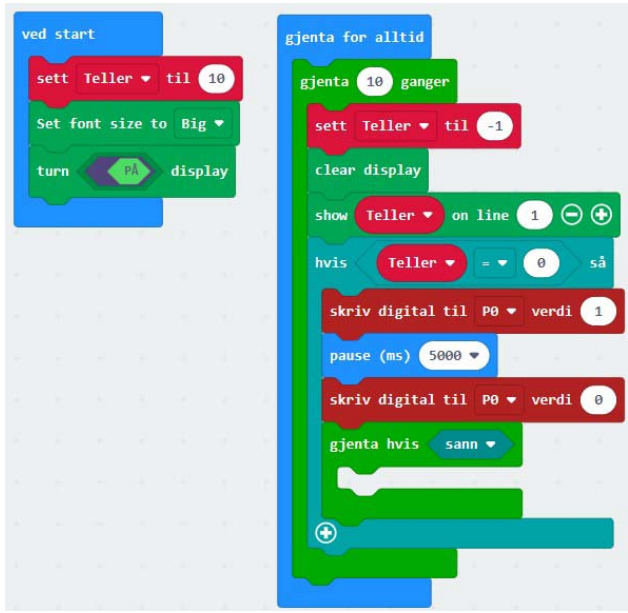
I de neste avsnittene vil vi vise forslag til løsninger til oppdragene 1 – 8, knyttet til grunnkurset. Vi må presisere at de viste forslagene ikke er en endelig fasit, men en av flere måter å løse oppdragene på.

F.1.1 Løsningsforslag – Oppdrag 1

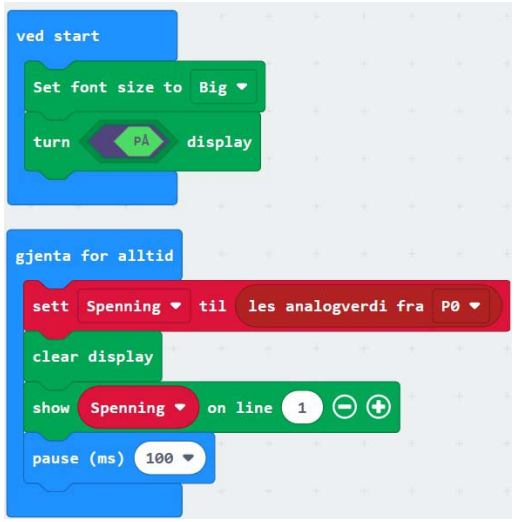
F.1.2 Løsningsforslag – Oppdrag 2



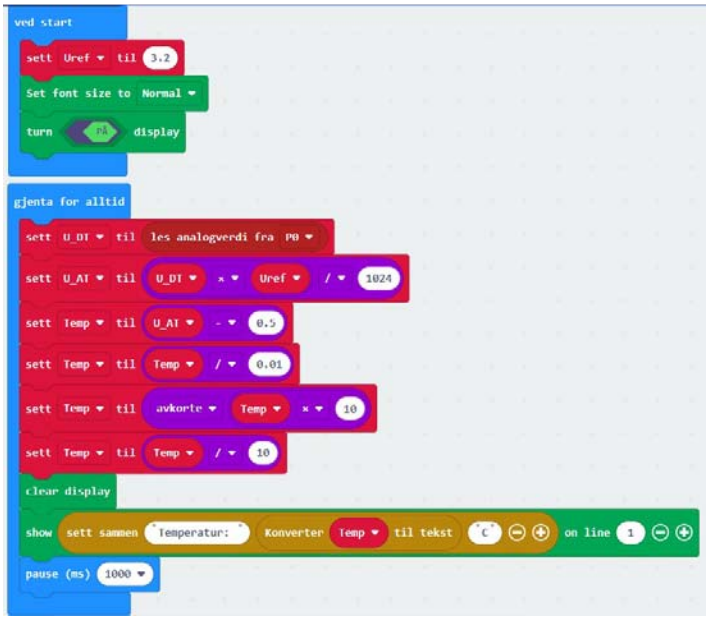
F.1.3 Løsningsforslag – Oppdrag 3



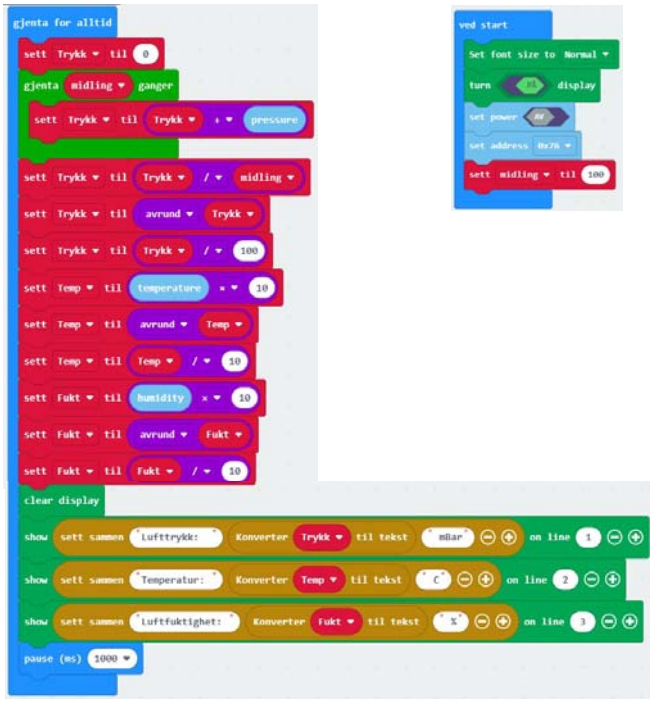
F.1.4 Løsningsforslag – Oppdrag 4



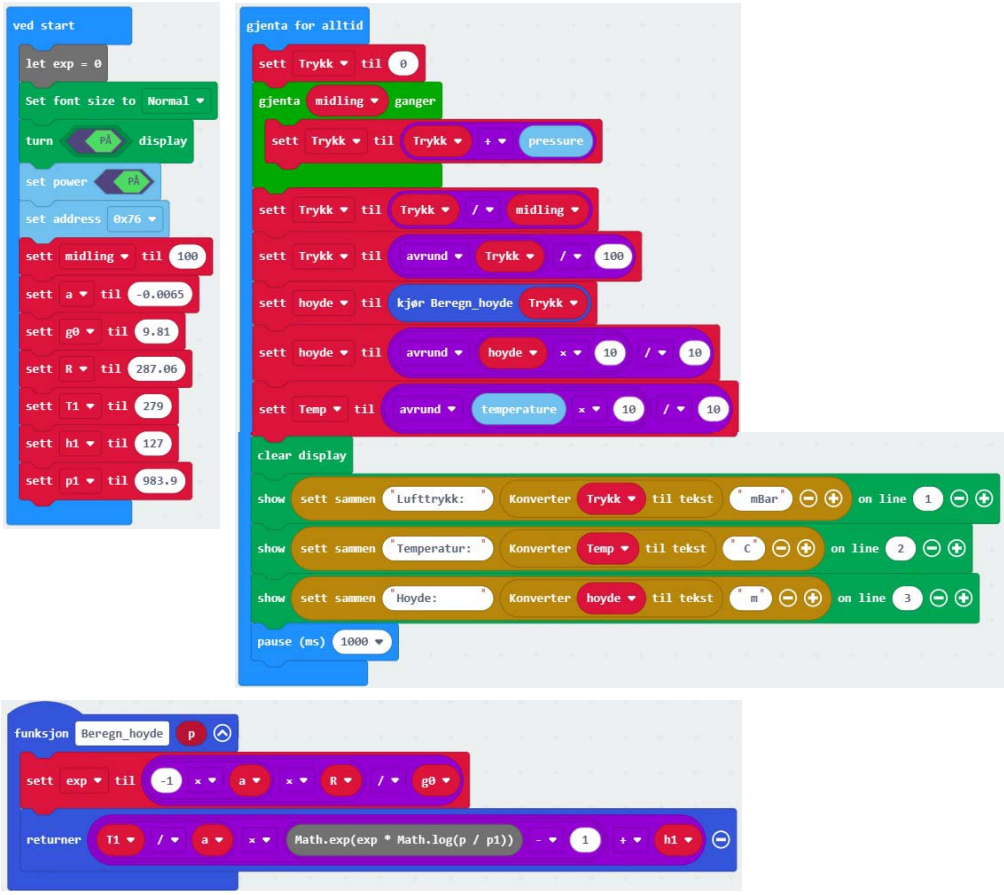
F.1.5 Løsningsforslag – Oppdrag 5



F.1.6 Løsningsforslag – Oppdrag 6



F.1.7 Løsningsforslag – Oppdrag 7



F.1.8 Løsningsforslag – Oppdrag 8

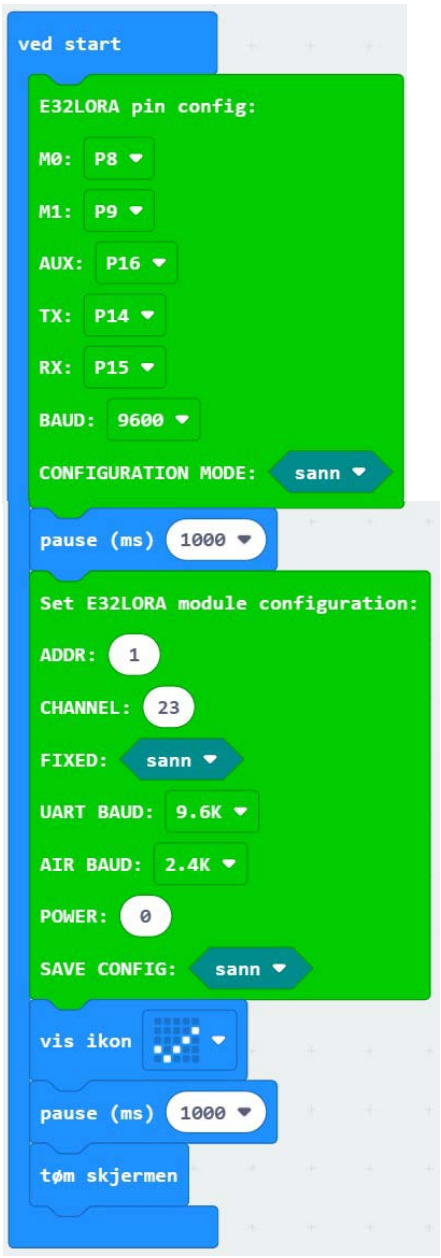
Til dette oppdraget er det ingen programmessige endringer i forhold til oppdrag 7.

F.2 Løsningsforslag CanSat-kurs, oppdrag 9 – 13

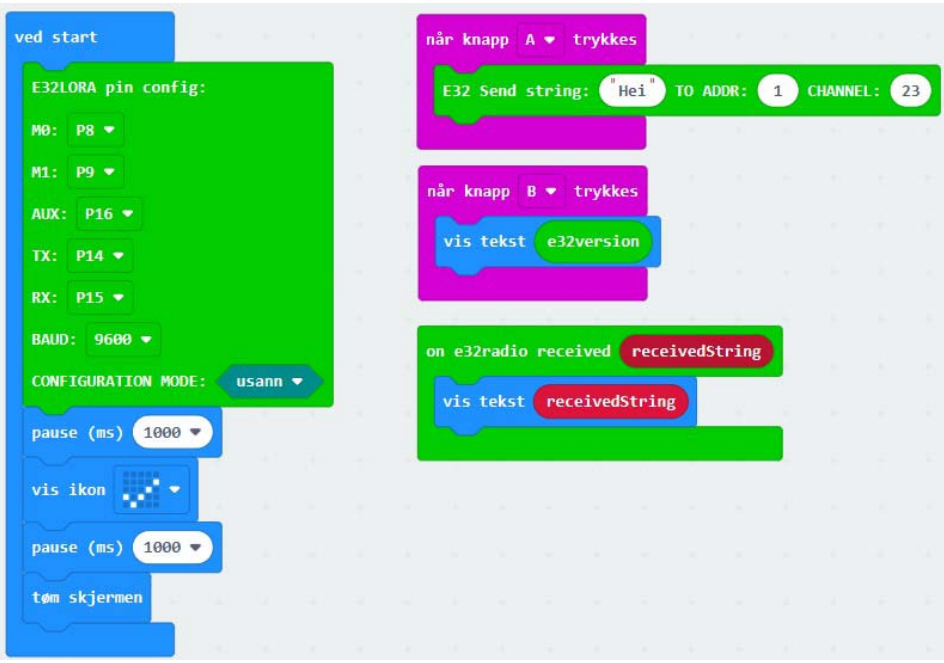
I de neste avsnittene vil vi vise forslag til løsninger til oppdragene 9 – 13, knyttet til CanSat-en. Vi må presisere at de viste forslagene ikke er en endelig fasit, men en av flere måter å løse oppdragene på. Vi må også ta forbehold om at det kan forekomme feil i løsningene.

F.2.1 Løsningsforslag – Oppdrag 9

Oppdrag 9A – Program for oppsett av radioene med E32-bibliotek



Oppdrag 9B – Program for testing av radiokommunikasjonen med E32-bibliotek

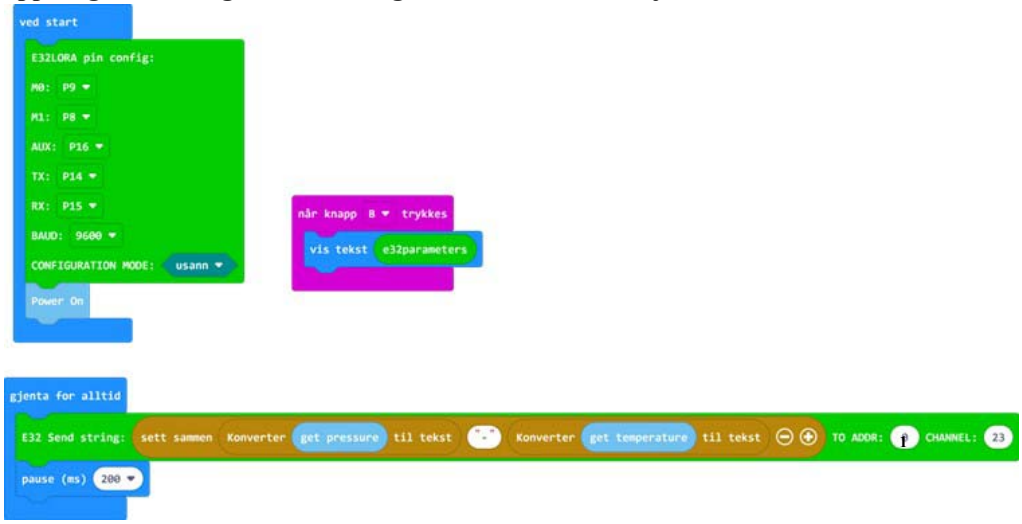


Vi legger merke til at dette programmet vil fungere både på sender- og mottakersiden.

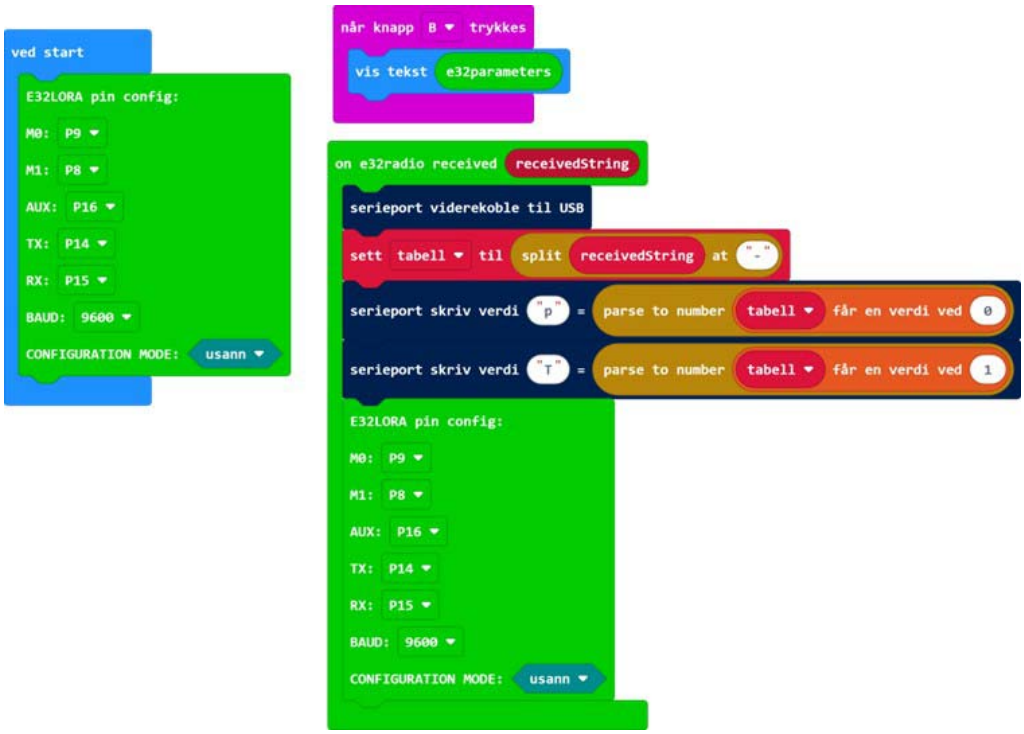
F.2.2 Løsningsforslag – Oppdrag 10

Program for testing av kommunikasjon mellom CanSat senderenhet og en CanSat bakkestasjon:

Oppdrag 10A – Program for testing av radiokommunikasjonen fra CanSat senderenhet



Oppdrag 10B – Program for testing av radiokommunikasjonen til CanSat mottakerenhet – bakkestasjon



F.2.3 Løsningsforslag – Oppdrag 11

Til dette oppdraget er det ingen programmessige endringer i forhold til oppdrag 10.

F.2.4 Løsningsforslag – Oppdrag 12

Figuren under viser løsningsforslag til sammenstilling og utsendelse av virkelige data med radioen E32. Inntil videre er dette programmet basert på en FTDI-løsning ved bakkestasjonen.

F.2.5 Løsningsforslag – Oppdrag 13

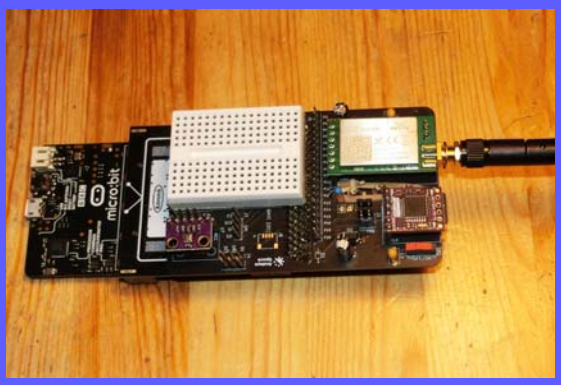
Her presenteres et forslag til hvordan dataene kan behandles med programverktøyet Python. Inntil videre se Vedlegg G, side 158.

Vedlegg G Eksempel på Python-program for presentasjon av data.

Vedlagt et enkelt Python-program for presentasjon av data.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd


#%% import og laging av arrays


#Henter inn fila som en dataframe i pandas
data = pd.read_csv("Data/CanSat_5.log", delimiter=',')


# Tar ut de kolonnonene som jeg vil bruke fra data og lager egne arrays (numpy lister) av dem
data blir lagret som float
# Kolonnene hentes ut ved overskrift (første data i kolonnen).


Time = data["Tid"].values/1000
temperatur = data['Temp'].values
lufttrykk = data['Trykk'].values
hoyde = data['Hoyde'].values


#%% Plotfunk


plt.figure(1,figsize=(11,8))
plt.plot(Time,temperatur,'b.')
plt.ylabel('Temp[C]')
plt.xlabel('Tid[sek]')
plt.title('Temperatur')
plt.ylim(20.0,32.0)
plt.grid()
```

```
plt.figure(2,figsize=(11,8))
plt.plot(Time,lufttrykk,'b.')
plt.ylabel('Lufttrykk[mBar]')
plt.xlabel('Tid[sek]')
plt.title('Lufttrykk')
plt.ylim(980,1020)
plt.grid()

plt.figure(3,figsize=(11,8))
plt.plot(Time,hoyde,'b.')
plt.ylabel('HoH [m]')
plt.xlabel('Tid [sek]')
plt.title('Barometrisk høyde over havet [meter]')
plt.ylim(0,100)
plt.grid()
```



Heftet gir en komplett beskrivelse av en micro:bit CanSat, "primary mission", og er ment å være en veiledning for læreren som underviser teknologi og forskningslære eller yrkesfag elektro, og kurshefte for micro:bit CanSat-kurs.

Nils Kr. Rossing
Dosent emeritus ved Skolelaboratoriet, NTNU
Tilknyttet Vitensenteret i Trondheim
E-post: nils.rossing@ntnu.no

Jørn Hafver
Realfagsformidlar ved Jærmuseet
E-post: joh@jaermuseet.no

Fredrik Motland Kirkemo
Realfagsformidlar ved Jærmuseet
E-post: fmk@jaermuseet.no

Simen Bergvik
Science Teacher ved Andøya Space Education
E-post: simen.bergvik@andoyaspace.no

Kristian Enoksen
Science Teacher ved Andøya Space Education
E-post: kristian.enoksen@andoyaspace.no

ISBN
Rev. 3.6– 29.04.24

Trykk: NTNU Grafisk senter

NTNU
Trondheim

Skolelaboratoriet
for matematikk, naturfag og teknologi

Tlf. 73 55 11 43
https://www.ntnu.no/skolelab

Institutt for fysikk