Rev 1.0

*Nils Kr. Rossing og Astrid Johansen*

# MICRO:BIT

## Radiokommunikasjon – Forslag til undervisningsopplegg


Trådløs logging av temperatur i "solfanger"

NTNU
Trondheim

Institutt for fysikk

Skolelaboratoriet
for matematikk, naturfag og teknologi

April 2021

Denne siden er blank

# MICRO:BIT

## Radiokommunikasjon –
## Forslag til undervisningsopplegg

Nils Kr. Rossing og Astrid Johansen


**Micro:bit - Radiokommunikasjon – Forslag til undervisningsopplegg**

Trondheim 2020

Bidragsytere:
*Nils Kr. Rossing*, *(nils.rossing@ntnu.no)* Skolelaboratoriet, NTNU
Astrid Johansen, (astrid.johansen@ntnu.no) Skolelaboratoriet, NTNU

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

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

Faglige spørsmål rettes til:
**Skolelaboratoriet ved NTNU**
v/Nils Kr. Rossing, 73 55 11 91
nils.rossing@ntnu.no

Høgskoleringen 5
7491 Trondheim

Skolelaboratoriet ved NTNU
Telefon: 73 55 11 91
http://www.ntnu.no/Skolelab/

Rev 1.0 – 18.04.21

# Forord

Heftet er skrevet i forbindelse med et undervisningsopplegg knyttet til videreutdanningskurs i naturdag våren 2021. Alle deltagerne har noe kjennskap til micro:bit programmering, men er ikke kjent med bruk av radioen. Det er derfor lagt vesentlig vekt på å formidle hvordan sender-mottaker delen av micro:bit fungerer inntil et visst nivå (se avsnitt 2.3). Hensikten med å forsøke å beskrive radiokommunikasjonen omtrent slik den faktisk foregår er ikke at kursdeltakerne skal formidle dette videre til elevene, men heller gi deltagerne noe mer bakgrunn som de ev. kan bruke til å stimulere de av elevene som er interessert i å vite litt mer.

I forbindelse med et annet prosjekt ble radiodelen av micro:bit presentert omtrent slik den er omtalt i avsnitt 2.3, men i en noe forenklet form. Tilbakemeldingen var at de klarte å følge med og forsto det som ble presentert. Det var enighet om at det var fornuftig å gi elever flest en forenklet versjon, men at den interesserte elev burde kunne følge resonnementet i avsnitt 2.3. I en forenklet utgave kan det være lurt å fokusere på at man kan kommunisere trådløst parallelt innen forskjellige grupper og at dette ikke handler om ulike kanaler med forskjellig frekvens, men bruk av ulike adresser som gjør at micro:bit'ene overser de meldingene som ikke er adressert til dem.

Den jevne elev vil sannsynligvis være mer opptatt av hvilke muligheter toveis radiokommunikasjon gir enn hvordan kommunikasjonen faktisk fungerer på bit-nivå.

Ideen til forsøket "Klimaendringer og biologisk mangfold – mål temperaturen i solfangere" er hentet fra Johanna Sexe's hefte *Undervisningsopplegg med micro:bit for naturfag vgs* [1].

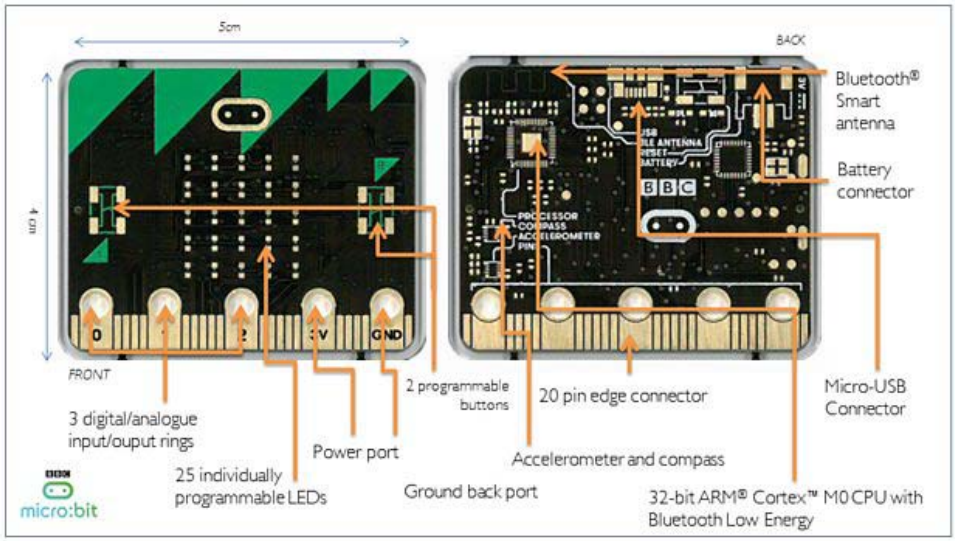
Trondheim
April 2021

Nils Kr. Rossing og Astrid Johansen
Skolelaboratoriet ved NTNU

# Innhold

# 1 Innledning

Micro:bit er blitt en svært populær krets for å komme raskt i gang med programmering og å realisere ideer. Årsaken til dette er flere, men de viktigste grunnene er at kretsen er utstyrt med sensorer og et enkelt diodedisplay, den kan kommunisere trådløst med andre micro:bits og den kan programmeres med ulike programmeringsverktøy 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.



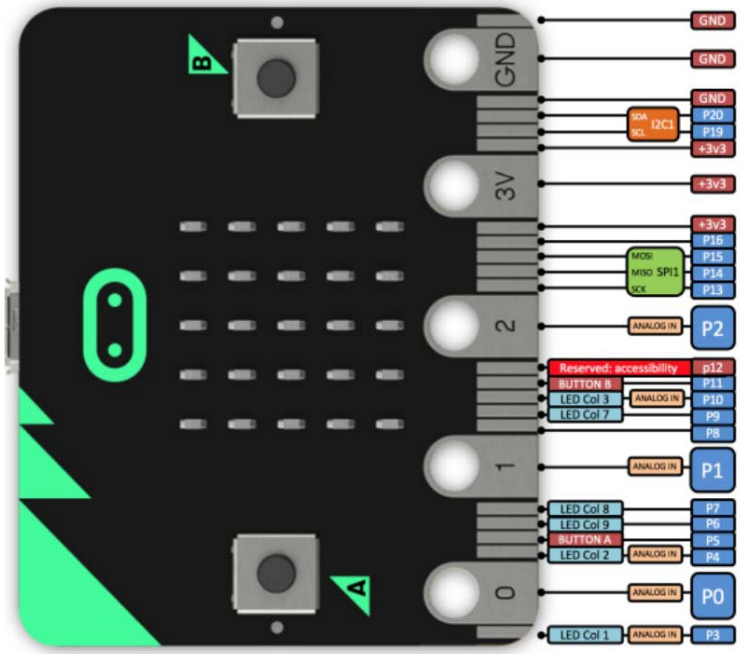

Kretsen har dessuten et 3 akse akselerometer og magnetometer slik at den kan detektere helningen til Micro:bit-kortet. Den kan dessuten fungere som kompass og har en innebygget temperatursensor. Videre kan diode-displayet brukes som lyssensor.

# 2 Litt teknisk småplukk om Micro:bit

I dette kapittelet skal vi se nærmere på Micro:bit-kretsen og løfte fram noen aktuelle programmeringsblokker. Det er ikke tanken at dette skal være noen komplett oversikt over blokkoding av Micro:bit.

## 2.1 Kantkontakten

Ved tilkobling av kantkontakten til Micro:bit får man tilgang til et langt større utvalg av porter, både inn- og utganger og analoge og digitale porter. Figuren under viser en oversikt over kantkontakten.

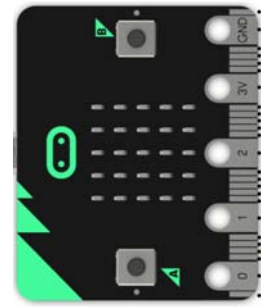



Som det framgår av figuren så har kretsen 20 porter hvorav 6 er analoge inn- eller utganger (P0, 1, 2, 3, 4 og 10), hvorav P0, 1 og 2 også kan brukes som berøringssensorer. De resterende 15 digitale portene (P5, 6, 7, 8, 9, 11, 12, 14, 15, 16, 17, 18, 19, 20) kan settes opp som inn- eller utganger etter behov så fremt de ikke er reservert til spesielle oppgaver som f.eks. P12 som er reservert til annet formål. Dessuten er P3, 4, 6, 7, 9, 10 brukt til å styre displayet, men kan frigjøres til andre formål etter behov. Vi legger også merke til at P3, 4 og 10, kombineres med analoge innganger. Hvilket betyr at lysdiodene også kan fungere som lyssensorer. Knappene A og B er koblet til P5 og P11.

I tillegg ser vi at drivspenningen er spesifisert til 3,3 V, men Micro:bit kan også kjøres på 3 V (dvs. ett CR2032 eller 2 x AA eller 2 AAA batterier). Forsyningsspenningen er tilknyttet tre av kontaktene langs kantkontakten og tre kontakter koblet til jord, se øverst på figuren over. Selv om mange

sensorer og *break out kort*[1] anvender 3,3 V, så er det også mange som trenger en arbeidsspenning på 5 V.

Pinne P19 (SCL) og P20 (SDA) kan også brukes til kommunikasjon via seriebuss ($I^2C$). Dette er en særdeles anvendelig seriekommunikasjonslinje for å kommunisere med andre digitale sensorer o.l. Det er også gitt mulighet for trelinje SPI-buss (P13, 14 og 15).




Jord (–)

3V (+)

P2

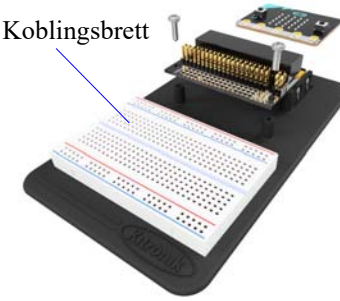
P1

P0

Fem av tilkoblingspunktene er utvidet og gjennomhullet slik at det skal være mulig å koble til bananstikkere eller krokodilleklemmer. Dette gjelder batterispenningen 3V (+ og –) og portene P0, P1 og P2. Dette gjør at en kan komme igang svært fort med et minimum av tilleggsutstyr. Samtidig som det ligger et betydelig utviklingspotensial i kretsen.

Det finnes kantkontakter som passer til Micro:bit og som selges til en overkommelig pris (typ. £4 montert på kretskort og dobbel stiftlist). Figuren over til venstre viser en kantkontakt uten kretskort. Bildet til høyre viser en kantkort med kretskort med dobbel stiftlist slik at man kan koble seg til kortet ved hjelp av jumper-ledninger.

For den som ønsker å gå videre og utforske mulighetene til kretsen, kan en anskaffe et Inventors kit som gjør det lett å koble til eksterne komponenter på et koblingsbrett.

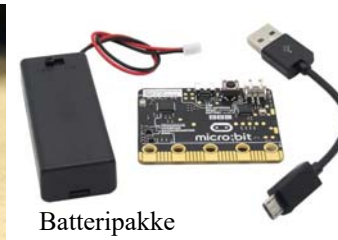



I tillegg til byggeplata som vist til høyre, så inneholder settet jumpere og en rekke sensorer og aktuatorer, men selges uten Micro:bit så det må kjøpes separat. Pris i Norge ca. kr. 500,- inkl. mva. Mens Micro:bit koster ca. kr. 250,- inkl. mva. Det er imidlertid store prisforskjeller også blant norske leverandører.

1. Små kretskort med sensorer eller andre elektriske komponenter, påmontert kontakter for tilkobling til andre elektroniske kretser

## 2.2 Batteripakke for Micro:bit

Dersom Micro:bit skal brukes uten å være tilkoblet PC, er det praktisk å montere den på et batteridekk (Power Back) som inneholder batteri (CR2032 - 3V), lydgiver (piezo elektrisk) og bryter. Dekket monteres til Micro:bit med tre maskinskruer med mutter og avstandsstykker. Alternativt kan en benytte ekstern batteripakke som gjerne følger med micro:bit.

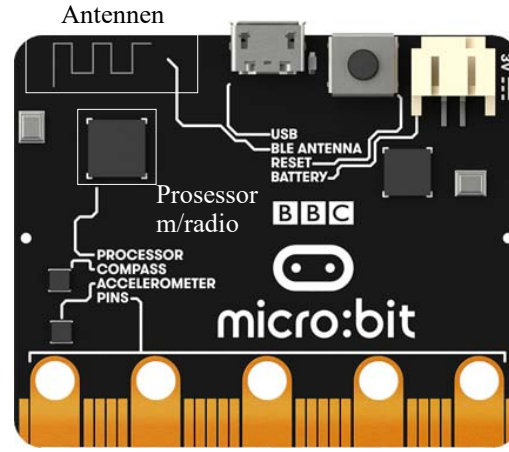



Batteripakke

## 2.3 Radioen i Micro:bit

La oss se litt nærmere på radiodelen av Micro:bit'en.

Radioen er uten tvil den enkeltegenskapen som gjør Micro:bit'en til et så kraftfullt mikrokontrollerkort. Radioen gjør det mulig å kommunisere trådløst mellom to mikro:bits eller grupper av mikro:bits, og til andre enheter som f.eks. en PC. Radioen omtales gjerne som en BLE hvilket betyr Bluetooth Low Energi radio. Antennen er også integrert på kortet og kan sees som en sik-sak-bord øverst i venstre hjørne på baksiden. Selve radiomodulen er integrert i prosessoren som er den svarte kvadratiske kretsen under antenna.

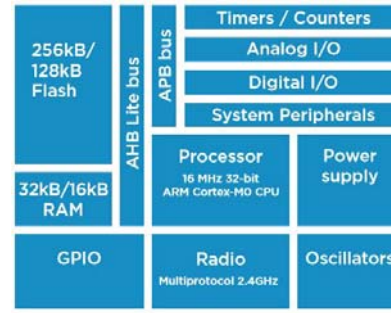



**Sendefrekvens og antennelengde**

Senderfrekvensene er lagt til 2,4 GHz området som er et lite frekvensbånd som brukes til mange ulike ting. Her finner vi bl.a. microbølgeovner[2] og andre kortdistanse kommunikasjonsystemer. Bølgelengden i dette frekvensbåndet er ca. 12,5 cm. Siden en vanligvis bruker antenner som har en lengde på en halv eller kvart bølgelengde, blir de ganske små. Antennelengden på mikro:bit'en er ca. 3 cm hvilket er ca. 1/4 bølgelengde. Dersom antennen er plassert på et kretskort, vil også lengden av antenne bli noe kortere enn om den hadde vært strekt ut i lufta. Dette skyldes at egenskapene til kretskortet som antennen er montert på, er annerledes enn for luft.

2. Mikrobølgeovner opererer normalt på 2,45GHz Mikrobølgeovener med lekkasje av mikrobølger kan derfor lett forstyrre kommunikasjonen mellom micro:bits.

### 2.3.1 nRF51822 – Nordic Semiconductor[3]



Blokkdiagram for nRF51822

Det norske firmaet *Nordic Semiconductor* med hovedkontor i Trondheim, har lagt et gullegg med den kombinerte mikroprosessoren og radioenheten nRF51822. Selve prosessoren er en 32-bit ARM-prosessor, med klokkefrekvens 16 MHz, som anvender en såkalt SoC teknologi (System on Chip). Dvs. at de fleste funksjonene i en kraftig mikrokontroller er plassert på samme brikke (substrat), gjerne også med en radioenhet (sender og mottaker - *transceiver*) på chip'en, som også er tilfelle for nRF51822. Fordelen med en slik løsning er at systemet kan gjøres ekstremt strømbesparende, da det ofte er effektkrevende å føre signaler fra en chip (krets) over til en annen.

Mikrokontrolleren nRF51822 har 31 generelle inn/utganger (GPIO[4]). En AD[5]-konverter på 10 bit gjør det mulig å koble til analoge signaler. Kretsen inneholder i tillegg en intern temperatursensor.

### 2.3.2 Radioen kan operere på to forskjellige måter

Radioen kan brukes på to forskjellige måter[6]:

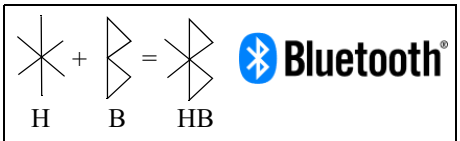
1. Den første omtales som en "peer to peer connectivity". Dvs. at kommunikasjonen skjer mellom to "likemenn" (peers). Et slikt system er uten "sjefer" som har kontroll over sendingene. Det gjøres heller ingen "avtaler" mellom sender og mottaker. En krets sender ut en melding og "håper" at noen fanger opp signalet. Det er denne varianten vi bruker når vi skal sette opp en enkel kommunikasjon mellom to micro:bits, eller mellom en micro:bit og flere andre. I tillegg kan flere grupper av micro:bits kommunisere med hverandre innen samme *gruppe* uten å forstyrre andre grupper. Micro:bits som skal kommunisere med hverandre må befinne seg innen samme gruppe som bestemmes av den som programmerer kretsene.

   Som all annen digital kommunikasjon overføres dataene i "pakker", dvs. et visst antall bit som er organisert på en bestemt måte. Skal større mengder data overføres så sendes flere pakker etter hverandre.

3. https://www.nordicsemi.com/-/media/Software-and-other-downloads/Product-Briefs/nRF51822-product-brief.pdf
4. GPIO - General Purpose In/Out
5. AD-konverter - Analog to Digital konverter. Omvandler en analog spenning til et digitalt tall.
6. https://www.littlebird.com.au/a/how-to/112/bluetooth-with-micro-bit

2. Den andre er en ordinær *bluetooth radiokanal*. Bluetooth er en kortholds radiostandard utviklet første halvdel av 90-tallet hos Ericsson Telecommunication for bl.a. å kunne kommunisere trådløst mellom elektroniske enheter som f.eks. mellom mobiltelefonen og øreproppper, mellom PC'en og musa eller tastaturet. Bluetooth opererer vanligvis i frekvensområdet 2,400 GHz – 2,485 GHz (hele ISM[7]-båndet er fra 2.4 – 2,5 GHz, en båndbredde på 100 MHz[8], eller 50 *kanaler* á 2 MHz, hvorav normalt kun 40 er for generell bruk). Micro:bit bruker en variant av bluetooth som kalles Bluetooth Low Energy (BLE). Den eneste forskjellen er at den sender med mindre effekt og er billigere å lage og å bruke. Den har derfor noe kortere rekkevidde enn tradisjonell bluetooth.

   Bluetooth standarden brukt hos micro:bits kan normalt kobles opp mot mobiltelefoner. For å oppnå kontakt mellom en micro:bit og en telefon utføres en "pairing". Dette kan gjøres på flere ulike måter som er godt beskrevet i "BBC micro:bit Bluetooth Profile"[9].




Hagall Bjarkan

Det fortelles at da de tre industrigigantene Intel, Nokia og Ericsson møttes i 1996 for bestemme seg for en kortdistanse radio, så foreslo ***Jim Kardach*** fra Intel og gi systemet arbeidstittelen Bluetooth etter den danske kongen Harald Blåtann siden han lyktes med å samle Skandinavia, slik de ønsket å forene PC og mobilverdenen. Dermed ble det slik. Symbolet for Bluetooth er også en sammensmelting av runebokstavene for H (Hagall) og B (Bjarkan), som er initialene til kongen.

Senderdelen av radioen kan programmeres til å sende med forskjellige effekter fra -30dBm til +4dBm i 8 trinn[10]. Hvilket betyr fra ca. 0,001 mW til 2 – 3 mW som er svært små effekter[11], men nok for kortdistansekommunikasjon (opp til ca. 70 m i fri sikt). Kretsen kan operere fra 1,8 – 3.6 V, dvs. på svært lave spenninger som også betyr lave effekter. Mottakerfølsomheten varierer fra -93 dBm til - 85 dBm, som er rimelig bra. Dataraten i radiokommunikasjonen kan settes til fra 250 kbps – 2 Mbps[12].

### 2.3.3 Peer to peer kommunikasjon[13]

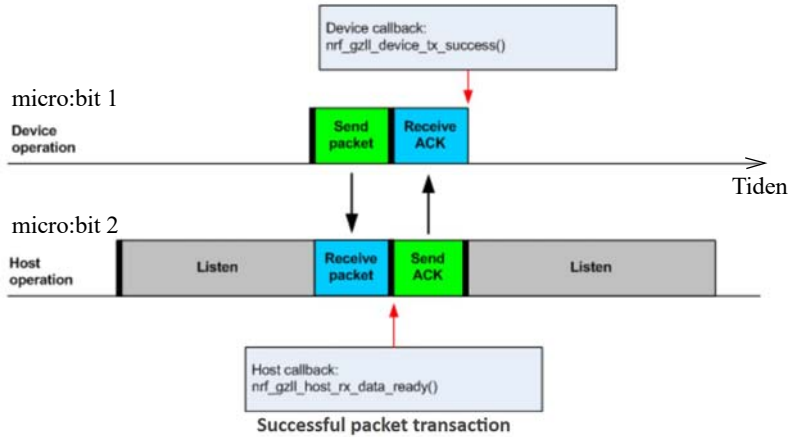
I dette avsnittet skal vi se nærmere på hvordan kommunikasjonen mellom to micro:bits foregår.

Den måten å overføre dataene på som brukes i dette tilfellet er spesiell for Nordic Semiconductor (og kan omtales som *proprietær*) og går under navnet Nordic Gazell. I denne måten å gjøre det på (protokollen) er hver pakke med data merket med en gruppekode ("group code") som inneholder en adresse og annen informasjon som er nødvendig for at dataene skal komme fram til adressaten

7. ISM - "Industrial, scientific and medical"
8. https://en.wikipedia.org/wiki/ISM_band
9. https://lancaster-university.github.io/microbit-docs/ble/profile/
10. https://makecode.microbit.org/reference/radio/set-transmit-power
11. En vanlig mobiltelefon kan sende på effekter opp til 2Watt, riktignok i kort pulser.
12. kbps - kilo bit pr. sekund. Mbps - Mega bit pr. sekund
13. https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.sdk51.v9.0.0%2Fgzll_02_user_guide.html&cp=4_0_7_5_0_2

på rett måte. Det er denne adressen som settes når vi velger "gruppe" når vi skal kommunisere med micro:bits.

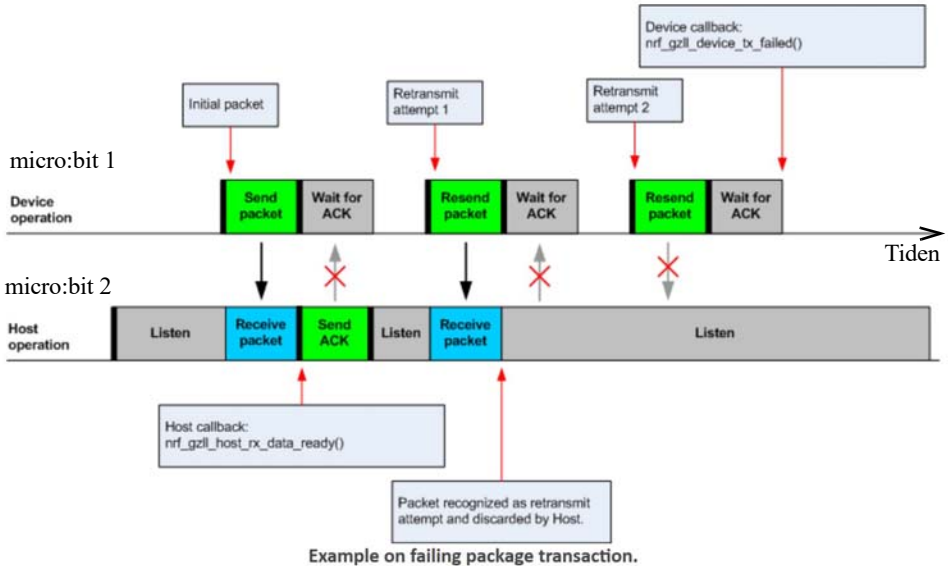
Figuren under viser hvordan en kan tenke seg at selve kommunikasjonen skjer.



Successful packet transaction

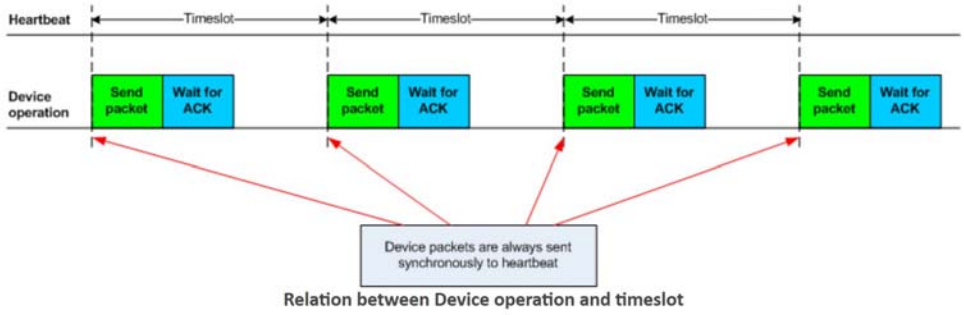
Vi tenker oss at micro:bit 1 (Device) "ønsker" å sende en melding til micro:bit 2 (Host). Micro:bit 1 sender en "pakke" med bit (0'er og 1'ere). Denne mottas av alle micro:bits som er innen rekkevidde, deriblant micro:bit 2. Denne sjekker adressen (gruppe) for å se om pakken er ment for den. Når alle bitene i pakken er mottatt, sendes det et svar tilbake til micro:bit 1 om at datapakken er korrekt mottatt (ACK – acknowledge) og at den er klar til å motta en ny datapakke.

Dersom micro:bit 2 "merker" at det er feil i dataene som den mottar, varsler den om at pakken ikke er mottatt korrekt og ber om at micro:bit 1 sender den på nytt. Ev. kan micro:bit 1 "erfare" at den ikke mottar noen kvittering (ACK), og sender pakken på nytt. En pakke vil kunne bli sendt om igjen flere ganger (3 ganger) før senderen gir opp.



Example on failing package transaction.

For at overføringen av data skal skje i ordnede former så sendes og mottas dataene i *tidsvinduer (Timeslot)* eller såkalte "hjerteslag" som vist i figuren under.



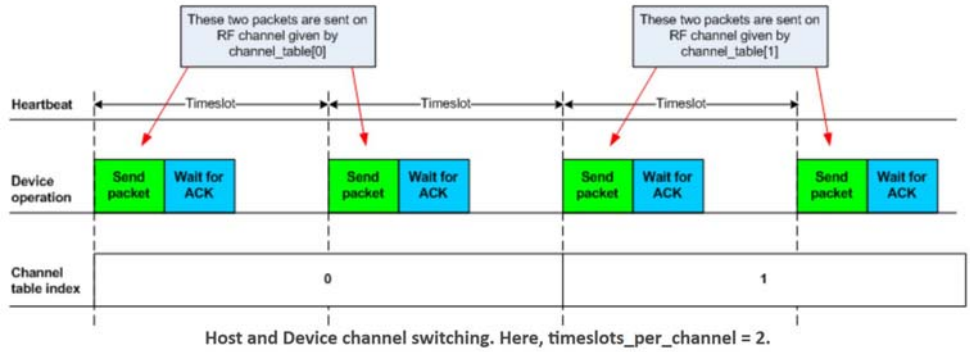
Relation between Device operation and timeslot

Dvs. at tidsrommet for hver ny pakke som sendes er hele tiden det samme (*Timeslot*). Dette krever at både sender og mottaker er synkronisert i forhold til hverandre. Lengden av et " timeslot" varierer med datahastigheten[14]:

- For 2 MBit/sek er timeslot perioden >= 600 µs.
- For 1 MBit/sek er timeslot perioden >= 900 µs.
- For 250 kBit/sek er timeslot perioden >= 2700 µs.

**Frekvenshopping**

Siden det kan være mye støy på de frekvensene som brukes gjør man bruk av *frekvenshopping*. Det vil si at med jevne mellomrom endres sendefrekvensen. Dersom det er støy på én frekvens så kan man håpe på at det er støyfritt på neste slik at en ev. gjentatt pakke vil komme fram til tross for at den mislyktes med første sending. Figuren under viser hvordan senderen skifter kanal etter bare å ha sendt to pakker



Host and Device channel switching. Here, timeslots_per_channel = 2.

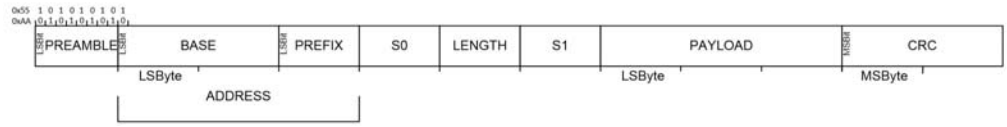
Når senderen skifter frekvens, så må også mottakeren skifte samtidig, ellers mister den pakken.

14.https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.sdk51.v9.0.0%2Fgzll_02_user_guide.html&cp=4_0_7_5_0_2

Både sender og mottaker må derfor bli enige om å bruke samme tabell over hvordan frekvenshoppingen skal utføres. Denne informasjonen må derfor overføres helt i starten av sendingene i det og kalles "Channel tabel index". Både sender og mottakere må slå opp tabellen som inneholder frekvensinformasjonen for å vite hvilke frekvenser de skal hoppe til, i tillegg til at frekvenshoppingen må skje synkront.

### 2.3.4 Datapakkenes oppbygging[15]

Figuren under viser et eksempel på hvordan en datapakke er satt sammen av mange deler med ulik funksjon.




**"Preamble" (innledning)**

Dette er en sekvens av 01010101 (0x55) eller 10101010 (0xAA)[16]. Hvilken som velges avhenger av den etterfølgende adressen. Dersom første bit i adressen er 0, så velger man 01010101. Tilsvarende velger man 10101010 dersom første bit i adressen er 1. Årsaken er at man ikke ønsker to like bit i overgangen mellom preamble og adresse.

**"Adressen"**

Adressen består av to deler BASE (2 byte) og PREFIX (1 byte). Adressen angir hvilken enhet eller enheter man ønsker å kommunisere med. Dersom den utsendte adressen stemmer med adressen hos mottakeren, så tas nytteinformasjonen (PAYLOAD) vare på i mottakeren, ellers forkastes den. Det er adressen som bestemmer hvilken gruppe micro:bit'ene skal tilhøre.

**"S0" og "S1"**

*S0* og *S1* er to byte (2x8 bit) hvor enkeltbitene gir mottakeren informasjoner om hvordan meldingen skal tolkes.

**"LENGTH"**

Angir lengden på den nyttige informasjon som skal overføres (PAYLOAD). Maksimal lengde på S0 + LENGTH + S1 + PAYLOAD er 254 byte, eller sagt på en annen måte ca. 250 tall og bokstaver.

**CRC (Cyclic Redundancy Check)**

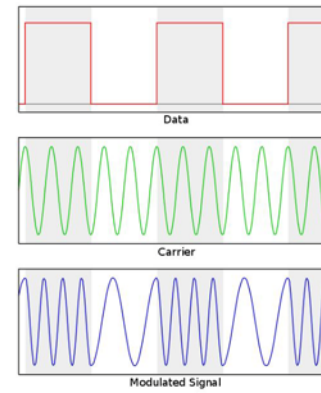
Dette er 2 byte (2x8 bit) som brukes til å sjekke om de mottatte dataene inneholder feil. Dersom noen få bit er feil så vil informasjonen i de to CRC-bytene til en viss grad kunne brukes til å rette

15.https://infocenter.nordicsemi.com/pdf/nRF51_RM_v3.0.1.pdf?cp=5_2_0 side 82

16.0xAA er tall uttrykt i det hexadesimale tallsystemet f.eks. 1010 1010 = AA hvor man benytter grunntall 16, tallrekken blir da 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Fire bit kan da uttrykkes med en bokstav 1010 = A. Dette skrives på formen (0xA)

feilene slik at meldingen blir riktig. Blir antallet feil for stort, klarer ikke disse to bytene å rette opp feilen, men de klarer å påvise *at den er feil* og ber om at meldingen sendes på nytt.

### 2.3.5 Modulasjon




Når dataene skal overføres så gjøres det ved å endre frekvensen på det utsendte signalet ørlite grann, slik at en digital 1'er og en digital 0'er har litt forskjellig sendefrekvens. Vi sier at signalet er *FSK modulert (Frequency Shift Keying)*. På figuren til høyre ser vi øverst det binære datasignalet med 0'ere og 1'ere. Midt i figuren ser vi *bærefrekvensen* som i vårt tilfelle er frekvensen på ca. 2,4 GHz. Nederst ser vi hvordan bærefrekvensen endres i takt med dataenes 0 og 1. Det må bemerkes at frekvensendringen er sterkt overdrevet i figuren.

*Bærefrekvensen* er den frekvensen som er valgt for overføringen av signalet og kan i prinsippet velges hvor det er plass eller hvor myndighetene har bestemt at denne typen sendinger skal skje.

I dette eksempelet skifter frekvensen brått som følge av endringen i det digitale signalet. En slik endring kalles for *Binært FSK* eller *BFSK*.

I mikro:bit brukes en modulasjonstype som kalles *GFSK*, eller *Gaussisk FSK*. Hvilket betyr at skiftet mellom frekvensene skjer langsommere og ikke brått som i BFSK. En langsommere endring gjør at det utsendte signalet ikke tar så mye plass i frekvensbåndet. Dvs. man kan overføre en større datamengde i en kanal med en gitt båndbredde.

På mottakersiden må en detektere disse små endringene i frekvens og gjøre dem om til et digitalt signal med 0'er og 1'ere. Helst med så få feil som mulig. Dette kalles å *demodulere* signalet og den komponenten som gjør det kalles en *demodulator*.

## 2.4 Programblokker som styrer radiokommunikasjon[17]

Micro:bit er tilrettelagt for å sende og motta beskjeder:

- ... mellom to micro:bits
- ... fra en til en gruppe micro:bits
- ... og å skille mellom ulike grupper som skal kommunisere internt, men ikke bli forstyrret av kommunikasjon i andre grupper.

La oss først se på noen grunnleggende kommandoer for å overføre informasjon mellom to eller flere micro:bits

17. Mye av dette stoffet er hentet fra Halfacree Gareth, The Official BBC Micro:bit User Guide, Wiley 2018, kapittel 8

## 2.5 Overføring av tekst og variabler

La oss se litt nærmere på kommandoene som overfører tekst og verdier (tall).

### 2.5.1 Etablering av en kommunikasjonsgruppe

Først må vi sette både sender og mottaker til samme gruppe. Det gjør vi ved å plassere blokken *radio sett gruppe* i *ved start* til samme i gruppenummer hos sender og mottaker.

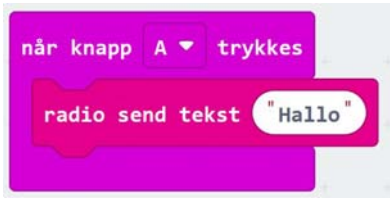
Sender


Mottaker

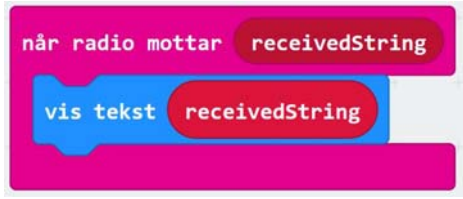

### 2.5.2 Send tekstmelding

Blokken i figuren under sender meldingen *Hallo* når knapp A trykkes. Mottakeren viser den mottatt teksten, som oppbevares i variabelen *receivedString,* i displayet (*vis tekst*). *receivedString* er en variabel som vi slipper å lage, men som er laget for oss av programmet. For å kunne vise innholdet i variabelen, må vi kopiere variabelen og legge den inn i *vis tekst* blokken. Det gjør vi ved å holde venstre musknapp nede og dra variabelen *receivedString* ut av plasseringen slik at vi får en kopi.

Sender




Mottaker

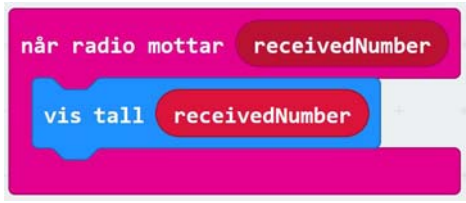



### 2.5.3 Sende tall

På tilsvarende måte kan vi sende et tall.

Sender


Mottaker

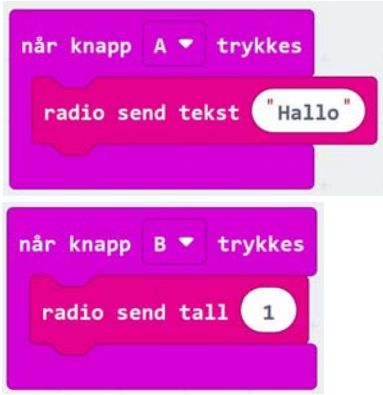



Programblokkene *radio send tall* i figuren over vil sende tallet "0" når knapp B trykkes. Når mottakeren mottar tallet så vises det i displayet.

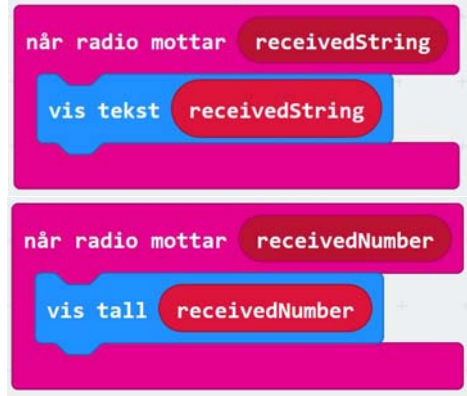
### 2.5.4 Sende tekst og tall

Vi kan også kombinere disse kommandoene ved at vi sender en melding når vi trykker på knappen A og et tall når vi trykker knappen B. Disse vil da mottas hos mottakeren og vises som henholdsvis tekst og tall.

Sender



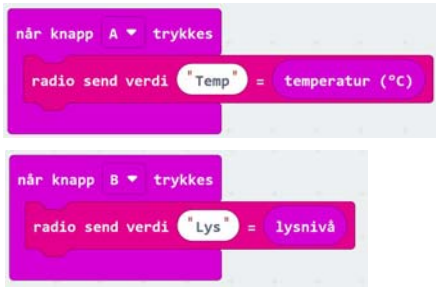

Mottaker



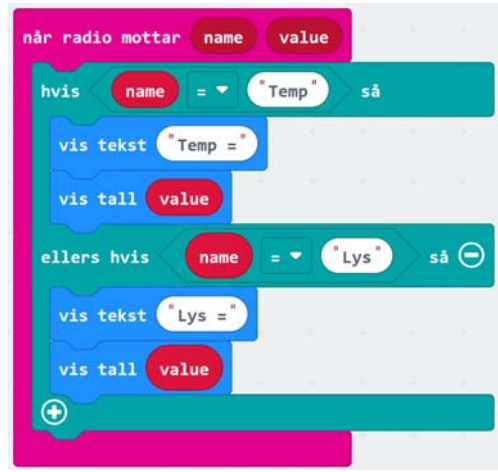

### 2.5.5 Send flere ulike verdier

Hva så om vi ønsker å sende flere verdier knyttet til ulike målinger. Hvordan skal vi da kunne skille de mottatte verdiene og vite hvilket tall som tilhører hvilken måling? I et slik tilfelle må vi gi verdiene et navn som vi sender over sammen med verdien. I praksis betyr dette at vi oppretter en variabel med et gjenkjennbart navn. La oss tenke oss at vi på sendersiden skal måle *temperatur* og *lysstyrke* som vi ønsker og sende over til mottakeren som igjen skal vise disse på displayet. For at vi skal vite hva som er hva må vi også angi hvilken måling som vises på displayet.

Sender



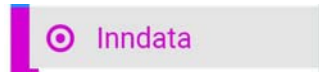

Mottaker



På sendersiden så har vi gitt *name* henholdsvis navnet *Temp* og *Lys* og knyttet dem til henholdsvis måling av *temperatur* og *lyssnivå* som vi leser av sensorene ved å bruke avlesningene:


og


Hentet fra menyen:

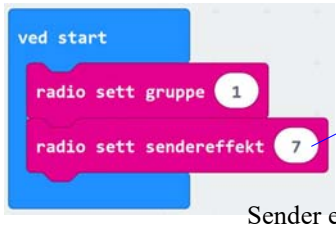



På mottakersiden må vi sjekke hvilken av de to målingene vi har mottatt ved å undersøke innholdet i variabelen *name*. Er denne *Temp* skriver vi "Temp =" på displayet i tillegg til verdien til temperaturen: *value*. Er denne *Lys* skriver vi "Lys =" på displayet i tillegg til verdien til lysstyrken: *value*. I dette tilfellet bruker vi displayet på sendersiden som en lysmåler i tillegg til at vi leser av temperatursensoren inne i mikroprosessoren.

### 2.5.6 Avlesning av signalstyrke og måling av rekkevidde

Micro:bit gir mulighet til å sette sendereffekten og å lese av styrken til det mottatte radiosignalet. Figuren under viser programmet for å sette sendereffekt og måle signalstyrken på henholdsvis sender- og mottakersiden.

Sender

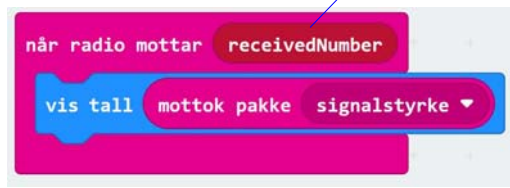



Setter sender-effekten til maks. (7)

Mottaker


Sender et tall hvert andre sekund


Leser av signalstyrken hver gang et tall mottas




Sendereffekten kan settes til verdiene 0 – 7 som tilsvarer ca. 0,001 mW til 2 – 3 mW. Signalstyrken leses av som et tall fra –128 som angir det svakest mulig målbare signalet, og til –42 som angir det sterkest mulig målbare signalet. Målingen skjer hver gang man sender et tall[18] eller også en tekstmelding.

18.https://makecode.microbit.org/reference/radio/received-signal-strength

### 2.5.7 Forandre frekvensbånd

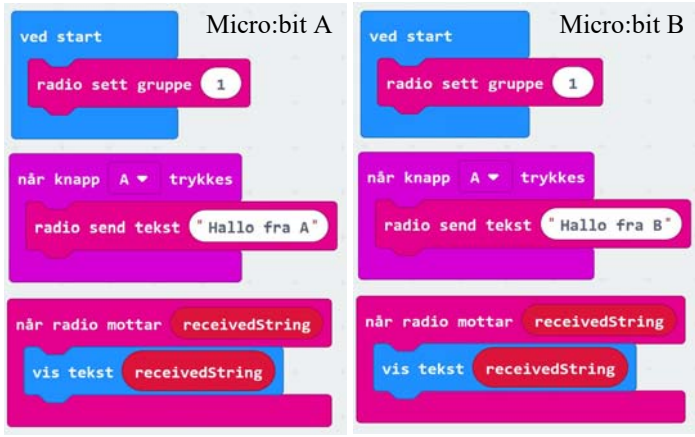

Vi har tidligere omtalt at micro:bit sender på frekvenser mellom 2,4 – 2,5 GHz som kan uttrykkes slik 2 400 – 2 500 MHz. Bredden til hver frekvenskanal er på 1 MHz. Om det er mye støy innenfor en kanal kan vi skifte kanal. Vi kan velge mellom frekvenskanalene 0 – 83. Dvs. at kanal 0 er frekvensen 2 400 – 2 401 MHz, mens kanal 1 er fra 2 401 – 2 402 MHz osv. til kanal 83 som strekker seg fra 2 483 – 2 484 MHz. Kommando-blokken som vi kan bruke for å sette frekvensen er *radio set frequency band*. Normalt vil dette gjøres i blokken *ved start*.

For hver frekvenskanal er det 256 grupper. Det er viktig at alle som skal kommunisere med hverandre bruker samme frekvenskanal og gruppe.

## 2.6 Kommunikasjon mellom flere micro:bit

Vi skal nå se hvordan vi kan sette opp toveis kommunikasjon mellom to eller flere micro:bits.

### 2.6.1 Å sette opp kommunikasjon mellom to micro:bits A og B (peer-to-peer)



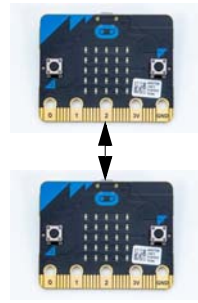

Figuren til venstre viser koden for hvordan man setter opp micro:bit A til å sende og motta innen gruppe 1: *ved start* sett *radio sett gruppe* til *1*. Dernest skal micro:bit A sende "Hallo fra A" når man trykker knapp A. Dersom micro:bit A mottar en tekst, så skal den vises på displayet som en tekst som ruller forbi.

Micro:bit B programmeres på samme måte, men denne sender en litt annen tekst: "Hallo fra B" når det trykkes på knapp A.

*Her programmeres begge micro:bits til å tilhøre gruppe 1. Det betyr både at de opererer med samme frekvensskjema (samme "kanal"), og at de opererer med samme adresse i adressefeltet i datapakken. I alt kan det settes opp 256 individuelle grupper (0 – 255) som kan operere uavhengig av hverandre til tross for at de opererer etter samme frekvenstabell. Dvs. at alle mottar alle meldinger, men at de siler ut de meldingene som ikke er merket med deres gruppeadresse. Dette kan bety at dersom mange sender samtidig så kan to meldinger forstyrre hverandre (kollidere) og må sendes på nytt.*

Eksempelprosjekt: *Dørklokke og døråpner*

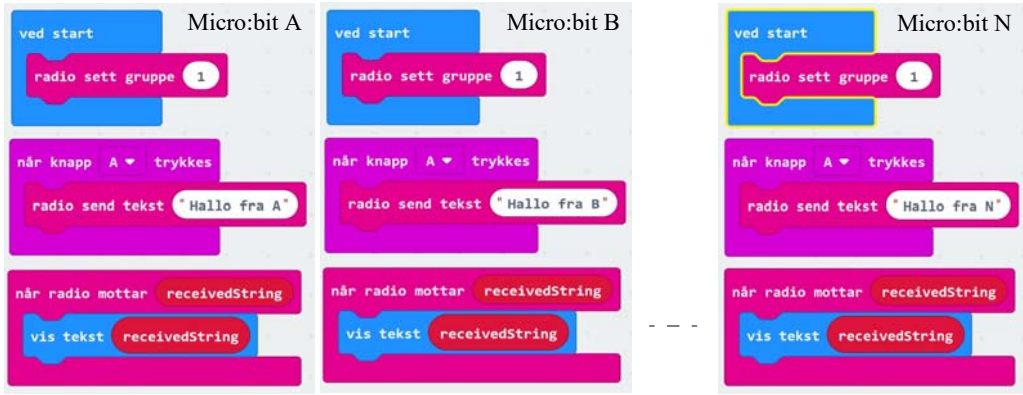
*Gruppe 1*



- *To micro:bits kan brukes som en toveis dørklokke. Når det ringer på sendes det en beskjed opp til kontorpulten din som varsler om at det står noen ved døren og vil inn. Du kan da svare at du er på vei for å åpne.*
- *Den mest nærliggende utvidelsen er å inkludere en ringelyd slik at du blir oppmerksom på at det er noen som vil inn, uten at du må se på displayet.*
- *Dernest vil det være en ide å ikke bare gi en beskjed om at du kommer å åpner ved å trykke knapp A, men rett og slett låse opp døra ved å trykke på knapp A (i praksis vil dette være et større prosjekt).*
- *Alternativt kan du varsle om at du kommer ved å trykke på knapp A. "JEG KOMMER" vises da på displayet ved ringeknappen. Eller du kan trykke knapp B som sørger for å låse opp døra og slippe inn gjesten.*

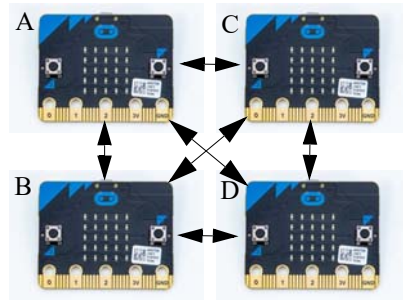
**Å sette opp kommunikasjon mellom mange micro:bits (fra en til mange)**

På samme måte kan man programmere flere til å operere i samme gruppe og dermed vil alle kunne kommunisere med mange som vist på figuren under.



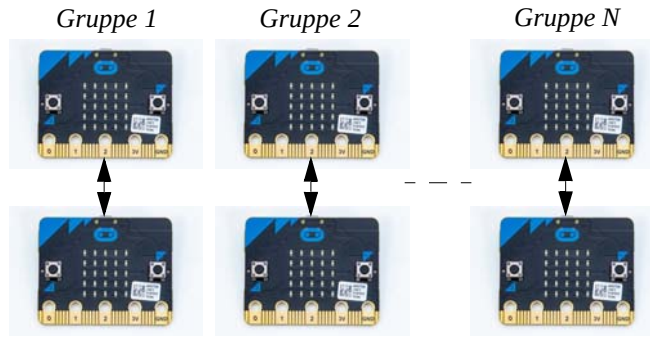

I dette tilfellet vil det når knapp A trykkes hos micro:bit A, gå ut melding til samtlige micro:bit som er koblet opp i nettverket. Det samme gjelder også for de andre. Dvs. at alle kan sende meldinger til alle.

*Gruppe 1*




Et slikt nettverk kalles *desentralisert* siden det ikke styres av *en sentral enhet*. Nettverket vil fungerer like godt til tross for at en eller ev. flere av micro:bitene er frakoblet. Eksempelvis kan A kommunisere med B selv om C er koblet fra. A kan kommunisere med C selv om B er koblet fra, og til sist B kan kommunisere med C selv om A er koblet fra.

Eksempelprosjekt: *Dørklokke og døråpner til ulike beboere*




- *En kan også koble opp flere micro:bits som opererer innen ulike grupper. Dette kan være aktuelt dersom man har flere beboere i huset, kanskje med en felles inngang. Da kan det være aktuelt med flere ringeknapper ved døra som kan kommunisere med én bestemt beboer.*

**Å sette opp gruppekommunikasjon**

Dette gir mulighet for en micro:bit til å kommunisere med ulike grupper, og til og med skifte gruppe under veis om det er ønskelig.

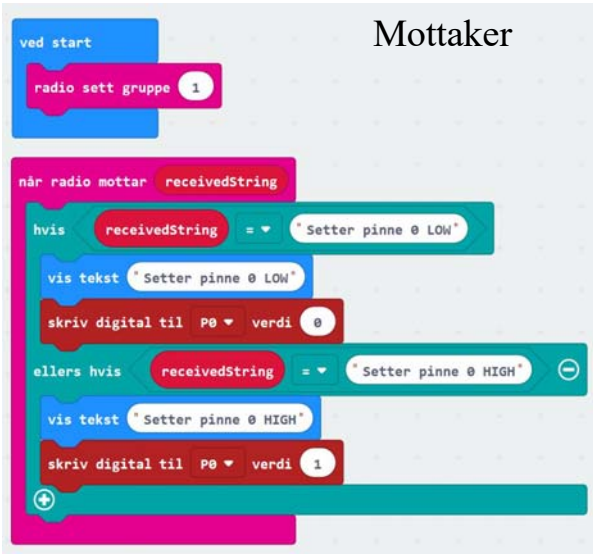
Vi legger nå inn et skifte av gruppe når vi trykker på knapp B. For å teste ut denne egenskapen så trengs tre mikro:bits, A, B og C.

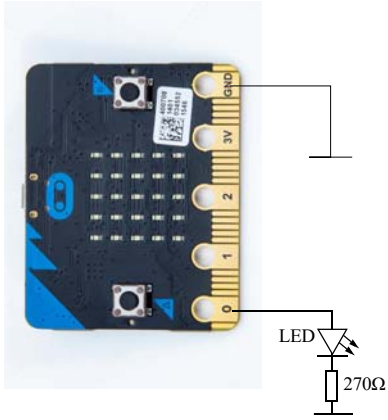

I utgangspunktet starter alle med å tilhøre gruppe 1. Når vi trykker på knapp A hos en av micro:bitene så mottar de to andre meldingen "Hallo fra A, B eller C". Dersom vi derimot trykker på knapp B, hos micro:bit A så vil det ikke skje noen ting, hos de to andre. B og C mottar ingen melding. Årsaken er at senderen hos A har skiftet til gruppe 2, mens de to andre, B og C, fortsatt befinner seg i gruppe 1. Trykker vi på knapp B hos micro:bit B, vil vi motta en melding hos A, men ikke hos C. Årsaken er at både A og B nå er i gruppe 2, mens C befinner seg forsatt i gruppe 1. Trykker vi på knapp B hos micro:bit C, så vil de to andre motta meldingen fra C. Nå er alle i samme gruppe igjen. De er i gruppe 2.

## 2.7 Styring av aktuator via radio

Det er mange måter å overføre informasjon mellom ulike micro:bits. I dette eksempelet skal vi se hvordan vi kan slå av og på en LED hos en micro:bit ved hjelp av en annen micro:bit. Figuren til høyre viser programmet i micro:bit'en som sender meldingene.


Figuren under viser programmet i micro:bit'en som mottar meldingen og tenner lysdioden.




Koden i figuren over bruker en tekststreng til å overføre informasjon i tillegg til at den slår av og på en lysdiode. En hvis-blokk (if-setning) sjekker om teksten er en kommando om å slå på ("Setter pinne 0 HIGH") eller slå av lyset ("Setter pinne 0 LOW"). Vi har valgt å bruke de engelske ordene for høy og lav, siden vi i denne sammenhengen ikke kan bruke norske bokstaver.

En mer elegant og mer fleksibel måte å gjøre dette på er å overføre informasjonen som et variabel navn f.eks. *Tenn LED* for deretter å la verdien fortelle hvilken LED som skulle tennes. Lag et program som tenner LED 1, 2 eller 3 (f.eks. rød, gul eller grønn).

## 2.8 Lagring av måledata i tabell

For å automatisere en måleserie som går over lengre tid så kan vi etter som målingene gjøres legge verdiene i en tabell i micro:bit'en. Når så måleserien er ferdig så må vi kunne lese av verdiene i tur og orden.

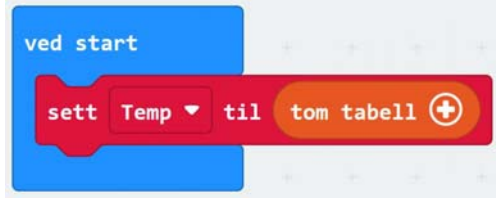
### 2.8.1 Opprett tabellen og gi den navn og dimensjon

1. Det første vi må gjøre er å opprette en tom tabell og gi den et navn når programmet starter, det gjør vi derfor i blokka *ved start*. Vi finner kommandoen knyttet til tabeller under meny-fanen *Avansert* og *Tabeller*.


2. Ved å trykke på pila ved *tabell* får vi anledning til å lage en tabell med et navn som vi ønsker. Vi oppretter en tabell med navnet *Temp*.


3. Dernest må vi bestemme oss for hvor mange ulike variabler vi ønsker å ta vare på i tabellen vår. Siden vi kun ønsker å ta vare på temperaturen så trenger tabellen vår bare en dimensjon. Vi trykker på (–) til høyre på blokka.


4. Dessuten vil vi at tabellen skal være tom når vi starter innsamlingen av data. Derfor henter vi oss blokka fra menyfanen som angir tom tabell og legger den inn i argumentet.



### 2.8.2 Skriv verdiene til tabellen

Når vi skriver verdier inn i tabellen gjør vi slik:

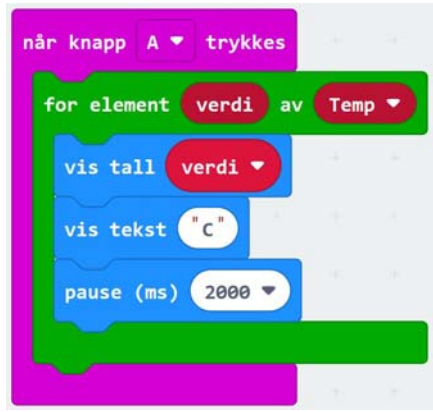
1. I dette eksempelet leser vi av temperaturen hvert 5 sekund og legger 20 målinger fortløpende inn i tabellen. Vi legger merke til at vi *legger til verdier.*


2. Dersom vi ønsker å legge til verdier som vi har mottatt via radioen, kan vi gjøre det på denne måten:


### 2.8.3 Les verdiene ut av tabellen

Dersom vi ikke lager en rutine som også kan lese ut av tabellen, gir det lite mening å bruke en tabell. Vi velger derfor å legge innholdet i tabellen fortløpende ut på displayet når vi trykker på knapp A.

1. I figuren til høyre leser vi tabellen *Temp* verdi for verdi ut på displayet med 2 sek. mellomrom. Vi må ev. legge inn tilstrekkelig tid slik at vi rekker å skrive ned verdiene etter som de kommer.



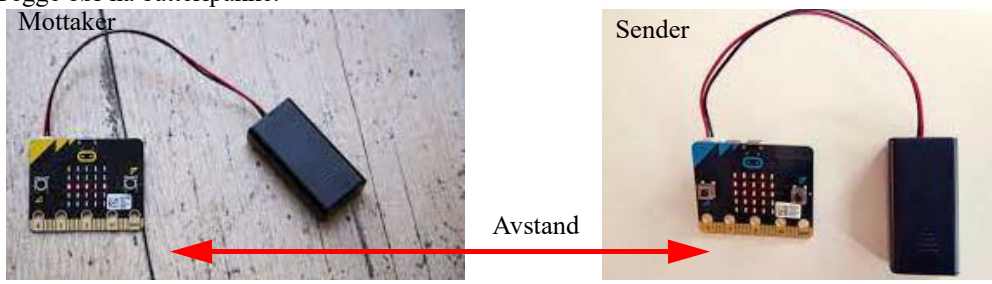
# 3 Oppgaver

I de følgende oppgavene skal vi bruke radio til å fjernovervåke en måleserie, men la oss først kartlegge rekkevidden til micro:bit under ulike betingelser.

## 3.1 Kartlegging av rekkevidde og signalstyrke

**Forberedende oppgave 1** *Undersøk hvordan det mottatte signalet hos microbit'en varierer med avstanden mellom sender og mottaker? Gjør praktiske forsøk og finn ut hvordan mottatt signalstyrke endrer seg med avstanden til senderen.*

For at vi skal kunne måle signalstyrken så må vi ha en micro:bit som fungerer som sender og en som fungerer som mottaker. Senderen kan vi f.eks. la ligge fast, mens mottakeren er bevegelig. Begge bør ha batteripakke.




**Programmering av senderen**

*La senderen sende en meldinger med jevne mellomrom. Marker gjerne på displayet med en hake hver gang det sendes en melding.*

**Programmering av mottakeren**

*Mottakeren skal motta meldingen fra senderen og vise signalstyrken på displayet. Husk å sett sender og mottaker på samme gruppe.*

For tips se avsnitt 2.5.6 på side 21.

**Måleresultater**

Noter signalstyrken i tabellen. Bruk meter eller skritt som måleenhet for avstand.

| Måling 1: | | Måling 2: | | Måling 3: | | Måling 4: | |
|---|---|---|---|---|---|---|---|
| Avstand | Signal | Avstand | Signal | Avstand | Signal | Avstand | Signal |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

| Måling 1: | | Måling 2: | | Måling 3: | | Måling 4: | |
|---|---|---|---|---|---|---|---|
| Avstand | Signal | Avstand | Signal | Avstand | Signal | Avstand | Signal |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

Det neste vi vil undersøke er hvilke andre ting som påvirker den mottatte signalstyrken.

> **Forberedende oppgave 2** *Kan microbit-signalet gå rundt hjørner? Undersøk om objekter som befinner seg mellom senderen og mottakeren påvirker styrken til det mottatte signalet?*

Her står dere fritt til å prøve ulike ting. Beskriv hva dere gjorde og hva resultatet ble. Vær mest mulig systematiske når dere gjennomfører forsøkene.

Forsøk 1:_____

Resultatet ble:_____

_____

_____

Forsøk 2:_____

Resultatet ble:_____

_____

_____

Forsøk 3:_____

Resultatet ble:_____

_____

_____

Forsøk 1:_____

Resultatet ble:_____

_____

_____

## 3.2 Undersøk nøyaktigheten til temperaturmåleren i micro:bit'en

Vi planlegger å bruke temperaturmåleren hos micro:bit'en til å måle temperaturen i et syltetøyglass når det belyses av sola eller en kraftig halogenlampe. Men først må vi kalibrere temperaturmåleren.

**Forberedende oppgave 3** *Undersøk hvor nøyaktig temperatursensoren i microbit'en er og om det er nødvendig å justere temperaturmålingen i dataprogrammet for å gi riktig verdi?*

*Diskuter hvordan dere kan finne og justere for eventuelle avvik. Utfør nødvendige målinger og foreta eventuelle justeringer.*

**Programmering**

*Lag et program som leser av temperatursensoren og skriver temperaturen ut på displayet.*

Vi fant at avviket var: _____°C

Følgende tiltak ble gjort:_____

_____

_____

**Forberedende oppgave 4** *For at temperaturmålingen microbit'en gjør skal være pålitelig, må microchipen på baksiden av micro:bit'en ikke utsettes for direkte sollys.*

*Hvordan kan dere lage et oppsett som sørger for dette?*

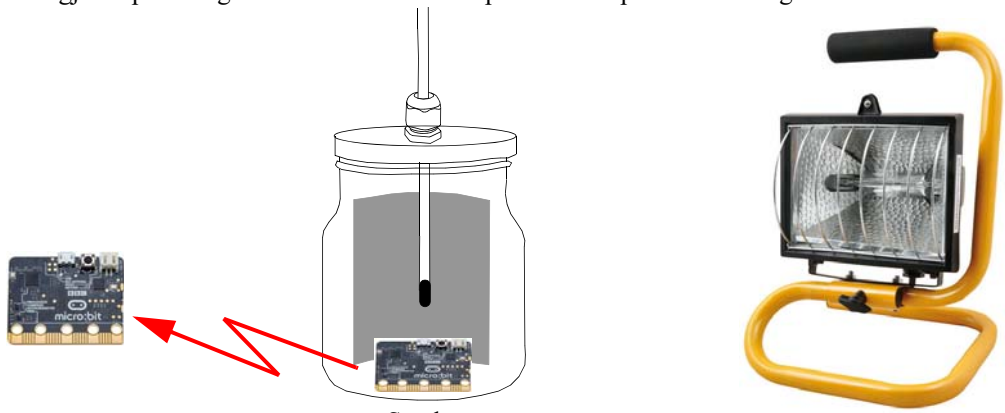
Følgende tiltak ble gjort:_____

_____

_____

## 3.3 Oppdrag: Mål temperaturutviklingen i et syltetøyglass

Endre gjerne på betingelsene for å se hva som påvirker temperaturutviklingen.



1. *Mål og les av temperaturutviklingen inne i et syltetøyglass som står ute i sola mens du selv befinner deg innendørs. Noter måledataene i en tabell og framstill disse grafisk. (Bruk gjerne Geogebra eller Excel.)***.**

**Programmering av senderen**

*Lag et program som leser av temperaturen med jevne mellomrom og sender resultatet i grader C til den utvendige micro:bit'en.*

**Programmering av mottakeren**

*Lag et program som mottar temperaturmålingen fra senderen og vise temperaturen i grader Celcius etterfulgt av C på displayet.*

For tips se avsnitt 2.5.3 på side 19.

Sett opp utstyret, belys glasset og noter temperaturen i tabellen under:

| Måling 1: | | Måling 2: | | Måling 3: | | Måling 4: | |
|---|---|---|---|---|---|---|---|
| Tid | Temp. | Tid | Temp. | Tid | Temp. | Tid | Temp. |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

2. *Diskuter hvilke faktorer som kan påvirke temperaturutviklingen inne i glasset og undersøk betydningen av dem ved å registrere temperaturutviklingen over en gitt tid. Finn støtte for antakelsene dine ved å endre betingelsene***.**
3. *Lag et oppsett som gir høyest (og lavest?) mulig temperatur i syltetøyglasset i løpet av f.eks. 10 minutter. Registrerer temperaturutviklingene og presenter dem grafisk***.**
4. *Diskuter om det praktiske dere har gjort her kan relateres til klimaproblematikken og om det vil være egnet å inkludere i klasserommet.*

## 3.4 Mulige utvidelser av oppdraget

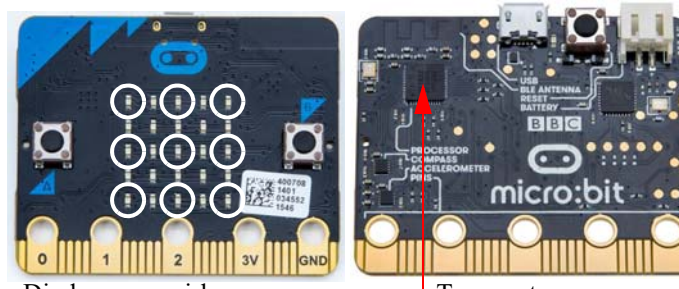
I dette avsnittet skal vi se på noen mulige utvidelser av oppdragene over.

### 3.4.1 Måling av temperatur og lysstyrke

Lys og varme er jo knyttet tett til hverandre. Siden micro:bit'en også har en lysmåler så ønsker vi å måle lysstyrken i tillegg til temperaturen.

> *Utvid oppsettet ditt til å måle lysstyrke i tillegg til temperaturen. Da må du passe på å merke målingene siden det er to parametere som skal overføres.*

Vi vet at temperaturmåleren sitter på baksiden av micro:biten og lysmåleren sitter på framsiden. For å måle lysstyrken så innser vi at framsiden må belyses[19]. Mens temperatursensoren måler temperaturen i grader Celcius så vil lyssensoren kun måle relativ lysstyrke. Sol er dessuten en svært lyssterk lyskilde så det er rimelig å anta at den i direkte sollys vil mål maksimalverdien hele tiden.




Diodene som virker som lyssensorer er ringet inn

Temperatursensor

> *Diskuter hvordan dere kan bruke lysmåleren på en slik måte at det i minst mulig grad påvirker temperaturmålingen. Hvordan vil dere unngå at lyssensoren måler maksimalverdien hele tiden?*

**Programmering av senderen**

> *Lag et program som leser av temperaturen og lysstyrken med jevne mellomrom og sender resultatet til den utvendige micro:bit'en.*

**Programmering av mottakeren**

> *Lag et program som mottar temperatur og lysstyrke fra senderen og viser temperaturen i grader Celcius og relativ lysstyrke på displayet.*

For tips se avsnitt 2.5.5 på side 20.

19.https://lancaster-university.github.io/microbit-docs/extras/light-sensing/

### 3.4.2 Automatisering av temperaturmålingen

Vi innser at det kan være kjedelig å måle temperaturøkningen på denne måten. Vi ønsker derfor å automatisere målingene:

> *Endre programmet slik at temperaturmålingene midlertidig blir lagret i en tabell i micro:biten i stedet for å vise dem etter tur på displayet og du slipper å sitte å følge med. Programmet skal lagres slik at hele måleserien kan leses av i ettertid.*

For tips se avsnitt 2.8 på side 26.

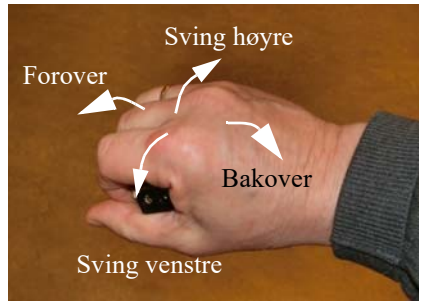
# 4 Fjernstyring av Bit:Bot

For de som ønsker litt større utfordringer, skal vi i dette kapittelet beskrive hvordan vi kan fjernstyre en robot av typen Bit:Bot XL ved hjelp av en håndholdt micro:bit. Vi ønsker å bruke Bit:Bot sitt bibliotek som gjør det lettere å programmere den.

Vi har tatt utgangspunkt i et undervisningsopplegg utviklet av Roy Even Aune ved Vitensenteret i Trondheim. En nettbasert utgave av opplegget finnes på viki-siden til Vitensenteret: http://wiki.trigger.vitensenteret.com/doku.php?id=introduksjon_til_bitbot

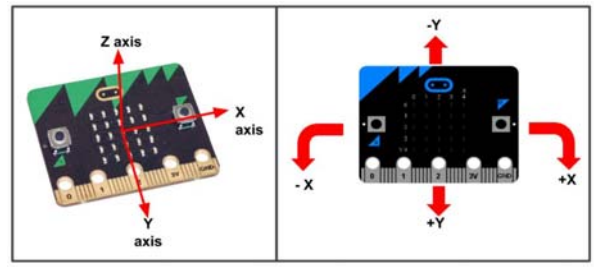
Vi ønsker å fjernstyre roboten ved hjelp av enkle håndbevegelser. For å utføre dette oppdraget trenger vi derfor to micro:bits, en batteripakke og en Bit:Bot XL. Den ene micro:bit'en, sender-enheten, bruker vi til å sende kommandoer til micro:bit'en som er montert på Bit:Bot'en, mottaker-enheten.

Vi ønsker at oppdragets fokus skal være radiokommunikasjon og bruk av sender- og mottaker-enhetene hos micro:bit'ene.




For å kunne styre og regulere farten til roboten, skal vi bruke *akselerometeret* i sender-enheten. Siden det kan være mest praktisk å holde kortet på tvers inne i hånda, så velger vi å øke hastigheten framover ved å bøye hånden framover, og tilsvarende øker vi hastigheten bakover ved å bøye hånden bakover (opp). I tillegg ønsker vi å kunne svinge roboten til venstre og høyre, ved å dreie hånden mot henholdsvis venstre og høyre. Dette er mulig når vi vet hvordan akselerometeret i micro:bit'en fungerer.

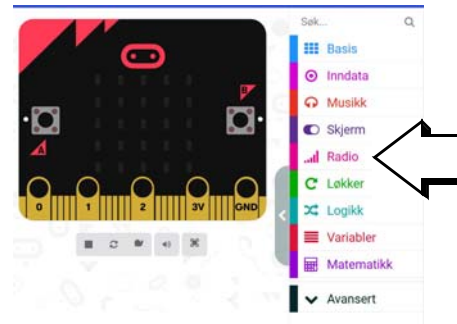
Figuren til høyre viser hvordan akselerometerets retninger er definert (se figuren til høyre). Siden akselerometeret forholder seg til tyngdeakselerasjonen, som er 1 g i vertikal retning, vil avleste verdier i x- og y-retning bli som i tabellen under avhengig av hvordan vi holder micro:bit'en. Legg merke til at den avleste verdien har benevningen mg (milli g):




| Handling | x-retning | y-retning | z-retning |
|---|---|---|---|
| Flatt på bordet, display opp | 0 mg | 0 mg | - 1000 mg |
| Flatt på bordet, display ned | 0 mg | 0 mg | + 1000 mg |
| Tilting mot høyre om y-akse, display opp | økende positiv verdi | 0 mg | minkende negativ verdi |
| Tilting mot venstre om y-akse, display opp | økende negativ verdi | 0 mg | minkende negativ verdi |
| Tilting framover om x-aksen, display opp | 0 mg | økende positiv verdi | minkende negativ verdi |
| Tilting bakover om x-aksen, display opp | 0 mg | økende negativ verdi | minkende negativ verdi |

## 4.1 Vi starter programmeringen

La oss begynne med å programmere senderenheten (radiosenteren) som er den enheten som skal holdes i hånda. Vi bruker kommandoblokker fra radio-menyen (se figuren til høyre)



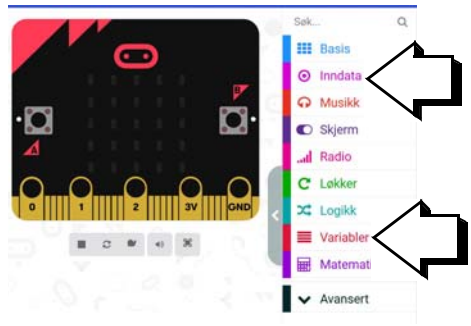

## 4.2 Programmering av senderenheten

### 1. Velg radiogruppe

Startblokken hentes fra *Basis-menyen* (blå) og *radio sett gruppe* hentes fra Radio-menyen (rød). Velg en radiogruppe som skiller seg fra de andre om det er flere i rommet som gjør det samme.

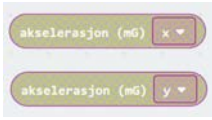
I vårt eksempel har vi valgt *gruppe 10*. For at vår sender-enhet skal kunne kommunisere med vår mottaker-enhet hos roboten, må også den settes til samme gruppe, dvs. gruppe 10 i vårt eksempel.

2. **Les av akselerometrene**



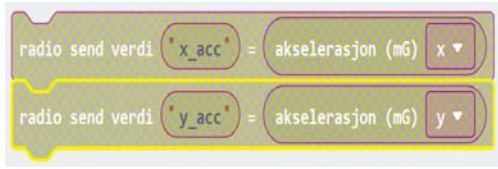

Vi skal lese av akselerometeret i x- og y-retning og sende verdiene til mottakerenheten. Verdiene for g leses av i milli g (her betegnet mG). Vi finner kommandoblokken for å lese av akselerometeret i menygruppen *Inndata*.

Siden vi skal lese av akselerometeret i både x- og y-retning så må vi gjøre to slike avlesninger, en for hver retning, x og y (se figuren under). Retning velges med den vesle pila til høyre).




Dernest må vi knytte avlesningene til *identfikatorer* slik at mottakeren vet hvilken måleverdi den mottar. Vi velger å kalle disse for *x_acc* og *y_acc*. De to identifikatorene lager vi ved å skrive dem inn mellom hermetegnene som vist på figuren under.

Senderkommandoen finner i vi Radio-menyen og vi velger *radio send verdi* med identifikatorer og avleste verdier:




3. **Legg inn i loop**

Vi gjentar sendingen hele tiden, gjerne med en liten tidsforsinkelse (*Pause*) mellom hver sending. *Pause* kommandoen finnes i menyen *Basis*. Vi velger å legge inn 50 millisekunder mellom sending av de to verdiene x*_acc* og *y_acc.*


Da er programmet for sender-enheten ferdig. Før vi sender programmet over til micro:bit'en gir vi programmet et navn og lagrer det. Bruk menylinjen for lagring nederst. Vi har kalt programmet for *Bit-bot Sender 1*

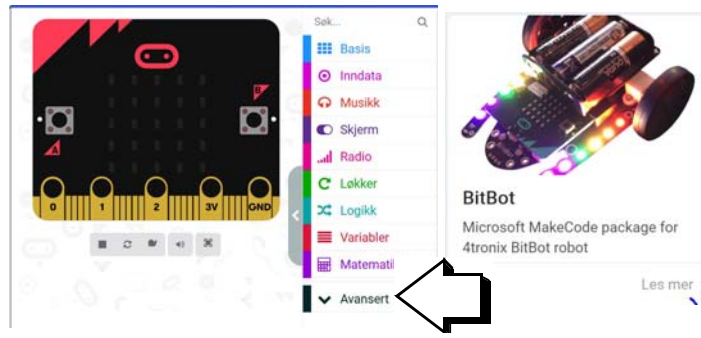

Man velger selv om man vil lagre i skyen (default) eller på egen PC.

Programmet legges over til sender-enhetens micro:bit, ved f.eks. å dra fila over til micro:bit'en som er koblet til PC'en med en USB-kabel.

## 4.3 Programmering av mottaker-enheten

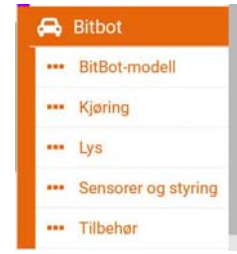
Mottaker-enheten er den micro:bit'en som er plugget i Bit:Bot'en og som mottar signalene fra sender-enheten.

**4. Installer bibliotek for styring av Bit:Bot**



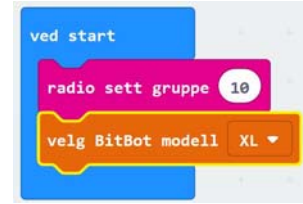

For at det skal være lettere å programmere Bit:Bot'en har noen laget et bibliotek med et ekstra sett av kommandoer. For å kunne ta i bruk disse kommandoene må vi installere biblioteket til Bit:Bot. Vi installerer biblioteket på følgende måte:

Velg menyen *Avansert* og deretter *Utvidelser.* Du kommer da til en meny hvor du finner en rekke utvidelser deriblant Bit:Bot. Velg Bit-Bot ved å trykke på ikonet (figur over til høyre).




Det finnes to versjoner av Bit:Bot: Classic og XL. Sjekk hva du har og velg riktig robot (Skolelaboratoriet har Bit:Bot XL). Dette gjør du når du skal til å bruke kommandoene. Figuren til høyre viser en rekke tilleggsmenyer med kommandoer spesiallaget for Bit:Bot.

**5. Velg radiokanal og modell**




Her velger vi samme radiokanal (gruppe) som senderen. Dessuten velger vi Bitbot modell fra Bitbot menyen: *BitBot-modell,* og deretter *velg BitBot modell*. Bruk den vesle pila til å velge modell, *classic* eller *XL*. Ved å velge *Auto* så vil Bit:Bot'en selv velge riktig bibliotek ved opplasting, dersom micro:bit'en er plugget inn i roboten når programmet lastes opp. Det er viktig å velge riktig modell siden det er brukt litt forskjellige porter på micro:bit'en for å styre de to typene robot. Skolelaboratoriets Bit:Bot'er er av typen XL.

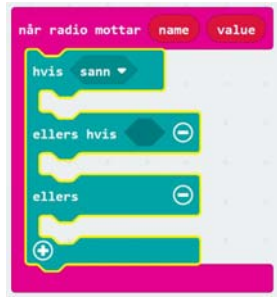
## 6. Motta akselerometerverdiene og legg dem i to variabler, X og Y

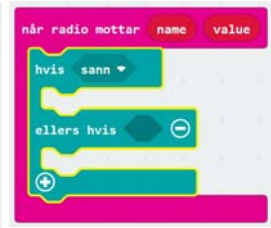

Vi oppretter de to variablene X og Y, som skal holde de mottatte verdiene fra *x_acc* og *y_acc*. Først oppretter vi de to variablene. Dette gjør vi i menyen *Variabler* og skriver inn de to variablene i innboksen *Lag ny variabel*.

Vi velger å sette de to variablene X og Y til 0 ved oppstart av programmet og legger dem i blokka *ved start*.

Dernest skal vi lytte etter radiomeldinger på vår radiogruppe (kanal). Til dette bruker vi kommandoblokken: *når radio mottar "name" "value"*. Her kan man enten bruke default navnene *"name"* og *"value"*, eller man kan definere sine egne navn.


Denne blokka sjekker om det mottas informasjon innen den aktuelle radiogruppen (10), om så er tilfelle utføres de kommandoene som legges inn i "funksjons-gapet". Når når det mottas informasjon innen den aktuelle gruppen, vil *name* og *value* inneholde henholdsvis navnet på den overførte parameteren (*x_acc* eller *y_acc*) og verdien (0 – 1023). Vi legger de to variabel-verdiene inn i henholdsvis variablene *X* og *Y*.

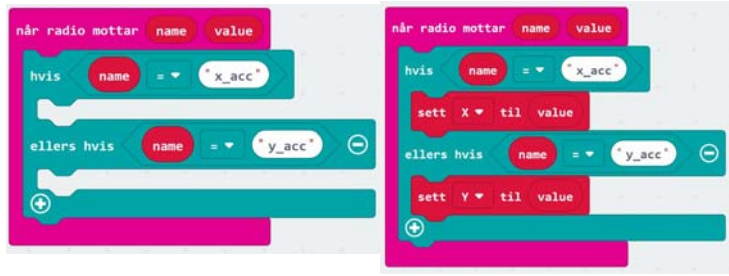




Velg *hvis-ellers-blokken* fra *Logikk-menyen.* Ved å trykke på + nederst i venstre hjørne av blokken, får man opp et ekstra *ellers-hvis-felt* som vi ønsker i å bruke her. Vi ønsker imidlertid ikke *ellers-feltet* så denne fjerner vi ved å trykke – til høyre for ellers.

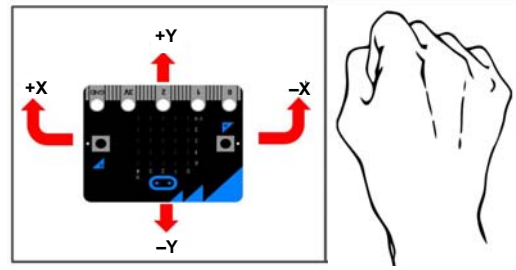
Dernest legger vi inn betingelsene ved å hente en sammenligning fra Logikk-menyen. Vi velger den som sammenligner tekst (med hermetegn, se figuren til høyre).

Så legger vi inn *name* og skriver inn *x_acc* og *y_acc* mellom hermetegnene for å sjekke hvilken variabel som er oversendt. Variabelen *name* kan vi kopiere fra den ytre ramma ved kun å dra den over.



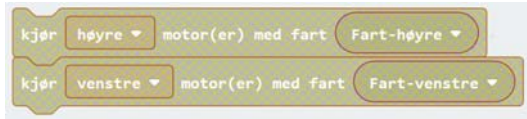

Nå har vi de aktuelle *x_acc* og *y_acc* liggende i henholdsvis variablene *X* og *Y*.

## 7. Styring av motorene




Vi skal bruke akselerometerverdiene mottatt fra sender-enheten til å styre roboten. Som vi har omtalt tidligere så mottar vi positive og negative verdier fra *x_acc* (*X*) og *y_acc* (*Y*) i henhold til figuren til høyre. Neven lengst til høyre viser i hvilken retning den holder micro:bit'en.

Roboten har to motorer, en motor til hvert av hjulene. Ved å kjøre de to hjulene med forskjellig fart, vil roboten svinge. For å kjøre motorene med en gitt fart bruker vi kommando-blokkene *kjør høyre/venstre motor(er) med fart*. Det er viktig at vi kan styre de to motorene uavhengig av hverandre.




I tillegg definerer vi to variabler "Fart-høyre" og "Fart-venstre".

Vi vet at *X*-verdien som kan være både positiv og negativ, skal kontrollere forskjellen mellom hastigheten til motorene, og *Y*-verdien, som også kan være positiv eller negativ, skal styre framdriften. Positive verdier vil drive roboten framover, og negative verdier vil drive roboten bakover. La oss sette opp noen ligninger som beskriver robotens bevegelser framover og bakover:

$$\text{Fart-høyre} = Y \tag{4.1}$$

$$\text{Fart-venstre} = Y \tag{4.2}$$

Dersom roboten skal svinge til høyre, må farten på venstre hjul øke og farten på høyre hjul avta, og omvendt om den skal svinge til venstre. Denne variasjonen styres av X-verdien. Vi kan da modifisere formlene våre slik:

$$\text{Fart-høyre} = Y - X \quad (4.3)$$

$$\text{Fart-venstre} = Y + X \quad (4.4)$$

Vi ønsker også å kunne justere *følsomheten* til styrefunksjonen. Dette kan vi gjøre ved å multiplisere *Y* og *X* med to følsomhetsfaktorer, *fx* og *fy*. Disse kan f.eks. ha verdier fra 0 til 2. Verdier mellom 0 og 1 vil redusere følsomheten, mens verdier mellom 1 og 2 vil øke følsomheten i forhold til verdien 1 som ikke gir noen justering av følsomheten.

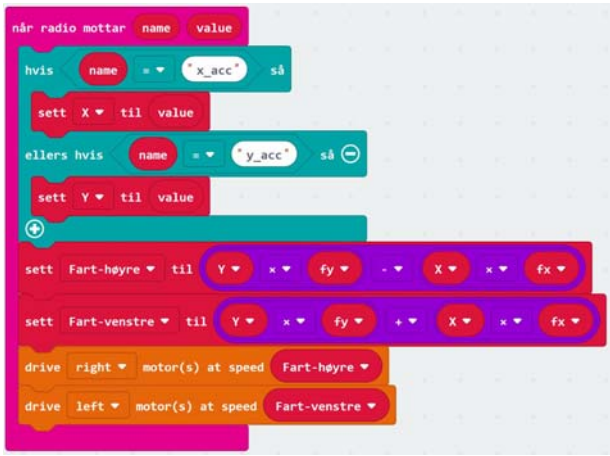
Økt følsomhet betyr at håndbevegelsen får større konsekvenser for farten, mens redusert følsomhet betyr at håndbevegelsen for mindre konsekvenser for farten.

$$\text{Fart-høyre} = Y*fy - X*fx \quad (4.5)$$

$$\text{Fart-venstre} = Y*fy + X*fx \quad (4.6)$$

I tillegg kan vi legge inn startverdier for *fx* og *fy* i oppstartsrutinen *ved start*. Husk og definer variablene *fx* og *fy* under menyen *Variabler*.

I utgangspunktet har vi gitt *fx* og *fy* verdien 1 hvilket betyr at vi hverken har redusert eller økt følsomheten.



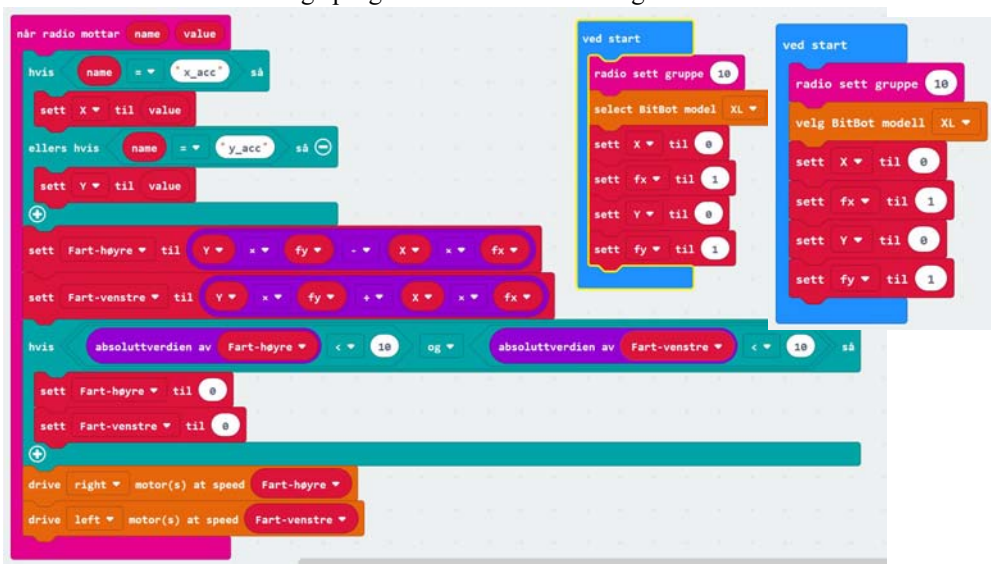


### 8. Sett opp et stoppvindu

Noen ganger kan det være vanskelig å få roboten til å stå helt stille. For å gjøre det lettere å holde roboten i ro når vi ønsker det, så kan vi sette *Fart-høyre* = 0 og *Fart-venstre* = 0 dersom verdien av de to variablene er under en viss *terskelverdi*, f.eks. 10. Siden dette gjelder både framover (+ verdi) og bakover (– verdi), kan vi bruke den matematiske funksjonen *absoluttverdi* når vi skal undersøke om verdiene er under terskelverdien.

For å teste om farten er mindre enn 10 bruker vi en *hvis-blokk som vi finner under menyen Logikk*.

Dermed skulle det ferdige programmet bli som vist i figuren under.




### 4.3.1 Overføring av programmet til micro:bit på BitBot

***NB!*** *Pass på at Bit:Bot XL løftes fra bordet når den slås på. Dersom den står på bordet vil lyssensorene (reflektanssensorene) på undersiden av roboten vanligvis registrere mørke og gå inn i parringsmodus for bluetooth hvilket vi ikke ønsker i denne omgangen.*

Dernest kan programmet flyttes over til micro:bit'en.

## 4.4 Tilleggsoppgaver

En kan se for seg en rekke forskjellige utvidelser av funksjonen til roboten. Her er noen forslag:

1. La roboten lage lyd når du trykker på knapp A.
2. Slå på lys ved å trykke på knapp B.
3. Skift mellom ulike farger på lyset når du trykker gjentatte ganger på knapp B. I løpet av sekvensen skal lysene være avslått.
4. Skriv et program som gjør at roboten følger en svart linje på gulvet.
5. La roboten skifte mellom å følge en linje og bli fjernstyrt når knapp A trykkes.

# 5 Referanser


[1] Sexe, J., Undervisningsopplegg med micro:bit for naturfag VGS, Juni 2020, Skolelaboratoriets blå hefteserie, https://www.ntnu.no/documents/2004699/12108297/Undervisningsopplegg+med+micro-bit+for+naturfag+vgs.pdf/f633c301-3cb4-f2e7-e179-bb7b3f3a2960?t=1597748114115

[2] Rossing N.K., Micro:bit - Forslag til undervisningsopplegg, Rev. 5.0 21.02.2020, Vitensenteret, Trondheim

[3] Nordic Gazell protokoll https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.sdk51.v9.0.0%2Fgzll_02_user_guide.html&cp=4_0_7_5_0_2

[4] Micro:bit radio https://infocenter.nordicsemi.com/pdf/nRF51_RM_v3.0.1.pdf?cp=5_2_0

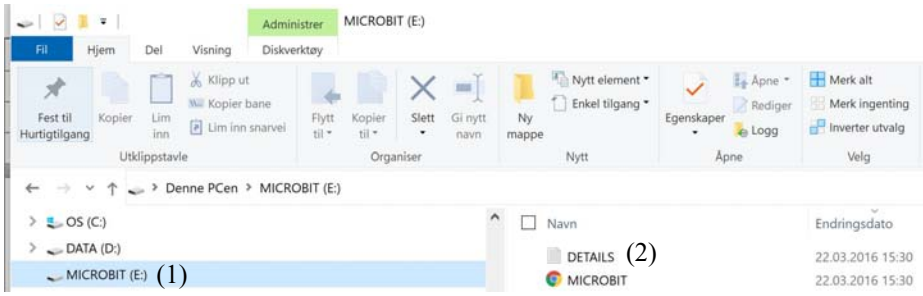
# Vedlegg A Installasjon av ny firmware

I tillegg til programmet som vi lager og legger inn i micro:bit, så trenger den noe ekstra programvare som ligger fast i micro:bit og som den blant annet bruker for å kommunisere med PC'en. Denne er installert i kretsen når den leveres. Denne programvaren kalles *firmware*.

Etter som nye funksjoner kommer til kan det være nødvendig å legge inn ny firmware, dette kan spesielt gjelde for gamle micro:bit.

Hvilken versjon av firmware du har kan sjekkes på denne måten:

1. Koble micro:bit til PCen og bruk file-utforskeren




   Klikk på MICROBIT drevet (1) og velg å klikk på DETAILS (2). Da får man opp en liste over informasjon, deriblant versjonsnummeret for firmware'en.

2. Oversikt over installert firmware

```
Fil Rediger Format Vis Hjelp
# DAPLink Firmware - see https://mbed.com/daplink
Unique ID: 9900000049734e45001680190000003f0000000097969901
HIC ID: 97969901
Auto Reset: 1
Automation allowed: 0
Overflow detection: 0
Daplink Mode: Interface
Interface Version: 0249
Git SHA: 9c5fd81e6545d00b7f7c21ca9d8577dbd6a5fed2
Local Mods: 0
USB Interfaces: MSD, CDC, HID, WebUSB
Interface CRC: 0xcdb7b2a3
Remount count: 0
URL: https://microbit.org/device/?id=9900&v=0249
```

   Vi ser at i denne micro:bit'en så er *Interface Versjon: 0249* som er siste utgave pr. 11.04.21.

Dersom du har en eldre utgave av firmware så kan det være greit å installere siste versjon. I så fall går du fram slik:

3. Last ned siste versjon av firmware
   Gå til: https://microbit.org/get-started/user-guide/firmware/
   Gå et stykke ned på siden å finn knappen for å laste ned den nye versjonen av firmware:

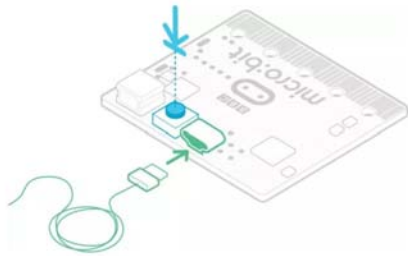

   Her er det to versjoner å velge mellom. Det skyldes at det i 2020 ble lansert en ny versjon av micro:bit – V2. De aller fleste har fortsatt V1 så det er vel sannsynlig at det er denne du har. Last ned firmware og legge den på et sted du finner igjen den.


Versjon 1

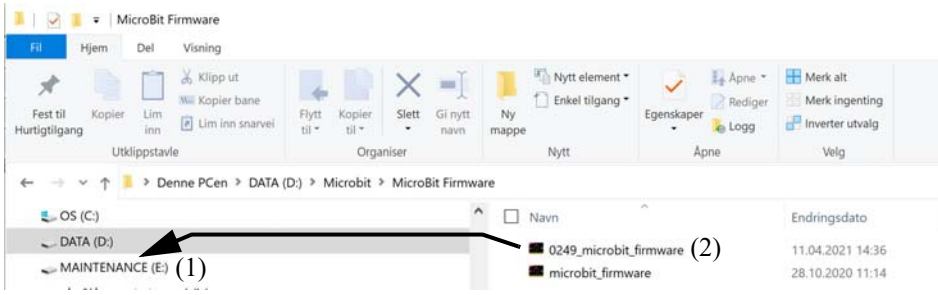

Versjon 2

   Versjon er utstyrt med innebygget mikrofon og høyttaler.

4. Koble micro:bit'en fra PC'en og fra eventuelle batteri.
5. Hold så inn Reset-knappen på micro:bit'en idet du på nytte plugger til kabelen til PC'en.



6. File-utforskeren vil nå vise følgende. Vi ser at vi har fått et nytt drev: MAINTENANCE (1).




   I tillegg har vi funnet fram den nye versjonen av firmwaren (2)

7. Installer den nye firmware ved å dra den over til drevet MAINTENANCE.
   Etter en liten stund vil den nye firmwaren være installert og det tradisjonelle drevet til micro:bit vises i file-utforskeren.

# Vedlegg B Erfaringer gjort under uttesting

## B.1 Målinger utført 12.04.21

### Måleoppstilling

Senderen, en standard micro:bit, ble lagt ned i et sylteagurkglass ca. 12 cm høyt. Temperaturen ble målt av micro:bit sin interne temperatursensor i tillegg til et ordinært glasstermometer stukket gjennom lokket og gjort lufttett med en kabelgjennomføring. Utendørs micro:bit ble forsynt med 2 x AA batterier.



### Erfaringer

- Svart papir ble langt på innsiden av glasset med micro:biten (senderen) plassert på høykant i bunnen av glasset bak papiret.
- Termometeret stakk ca. 5 cm ned i glasset på innsiden, dvs. det målte i øvre halvdel av glasset. Dersom termometeret ble stukket lengre ned i glasset var det ikke mulig å lese av temperaturen på utsiden. Sensoren hos micro:biten var ca. 5 cm levere en termometeret.
- Senderen tok målinger hvert 5. sek.
- Avviket mellom glasstermometer og måling med micro:bit viste +3 grader i micro:bitens favør når glasset sto innendørs. En annen micro:bit viste + 5 grader avvik. Det er med andre ord forskjell mellom ulike micro:bits. Det ble korrigerte for avviket før glasset ble flyttet utendørs.
- Glasset ble plassert i sola slik at det svarte arket skygget for glasstermometeret og mikrobiten som sto vertikalt mot bunnen av glasset.
- Temperaturforskjell mellom glasstermometer (ca. 34C) og micro:bit temp. (ca. 25C), er da 9 grader. Utetemperaturen er målt til +3 grader i skyggen.

Med andre ord det kan se ut til at det er store temperaturforskjeller inne i glasset uten omrøring.

Det anbefales at glasset settes ute i skyggen i god tid før målingen i sola starter. Dermed skulle en kunne se en solid temperaturøkning når det flyttes ut i sola. Det ble ikke målt hvor lang tid det tar til temperaturen inne i glasset stabiliserer seg etter at det ble flyttet ut i sola.

Det var ikke noe problem med en avstand mellom sender og mottaker på ca. 10 meter gjennom glasset og en yttervegg, med vindu. Det ble kjørt med maksimal sendereffekt (7).

# Vedlegg C Løsningsforslag

## C.1 Måling av signalstyrke

Figuren viser programmet for senderen og for mottakeren. Vi har valgt å vise ikoner idet senderen sender ut signaler.

Senderen

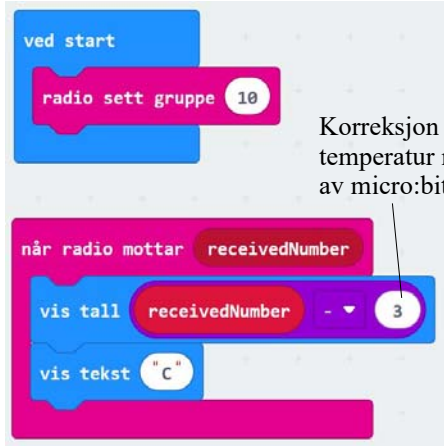

Mottakeren

## C.2 Måling og overføring av temperaturmåling

Programmet måler temperaturen i solfangeren ved hjelp av en micro:bit (sender) og overfører verdien trådløst til en annen micro:bit (mottaker). Legg merke til korreksjonen på 3C i eksempelet her. Avviket er funnet ved å sammenligne måling av temperaturen ved hjelp av micro:bit og et vanlig glasstermometer. Korreksjonen er gjort etter at begge målerne har stått i romtemperatur i lang tid.

Senderen


Mottakeren



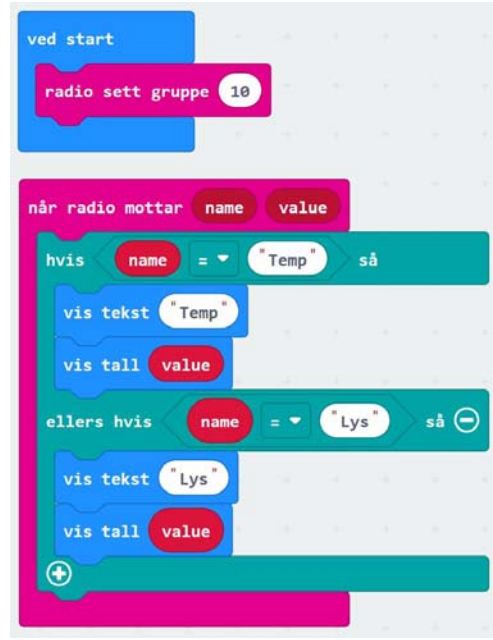

Korreksjon av temperatur målt av micro:bit

## C.3 Måling av temperatur og lysstyrke

Dette programmet måler temperatur og lysstyrke og sender resultatet over til mottakeren.

Senderen


Mottakeren

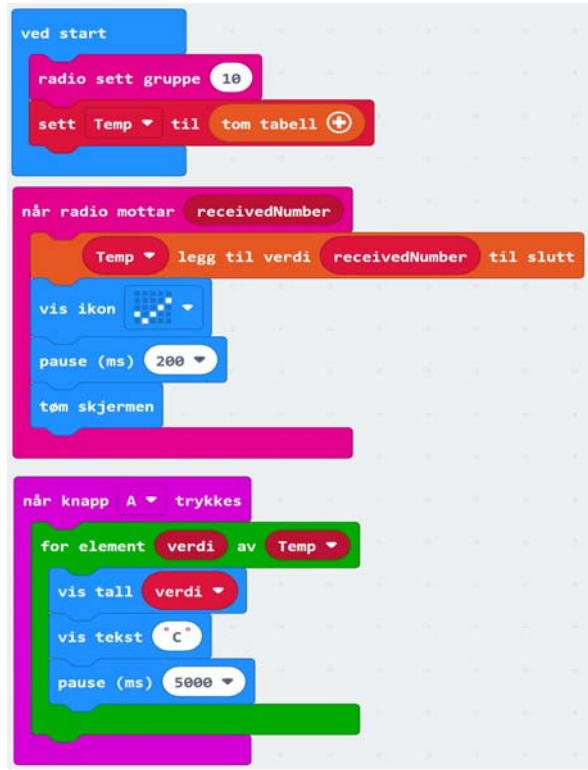


## C.4 Automatisering av temperaturmålingen

Programmet måler temperaturen og sender verdiene over til mottakeren som lagrer dem i en tabell som kan leses ut i ettertid.

Senderen


Mottakeren



Heftet beskriver radiodelen av BBCs micro:bit og hvordan den kan brukes til trådløs logging av temperatur i en solfanger ev. for styring av en Bit:Bot med en håndholdt micro:bit og håndbevegelser. Heftet omtaler også sentrale blokkommandoer i forbindelse med bruk av radioen hos micro:bit.

Oppskriften er ment å være en introduksjon til temaet om sender og mottaker og hvordan trådløs kommunikasjon kan anvendes på en nyttig måte. Vi håper beskrivelsen kan inspirere til andre lignende prosjekter hvor man anvender radioen.

**Nils Kr. Rossing**
Prosjektleder ved Vitensenteret i Trondheim
Dosent ved Skolelaboratoriet ved NTNU
e-post: nils.rossing@ntnu.no

**Astrid Johansen**
Universitetslektor ved Skolelaboratoriet, NTNU
e-post: astrid.johansen@ntnu.no

ISBN
Rev. 1.0 - 18.04.21