

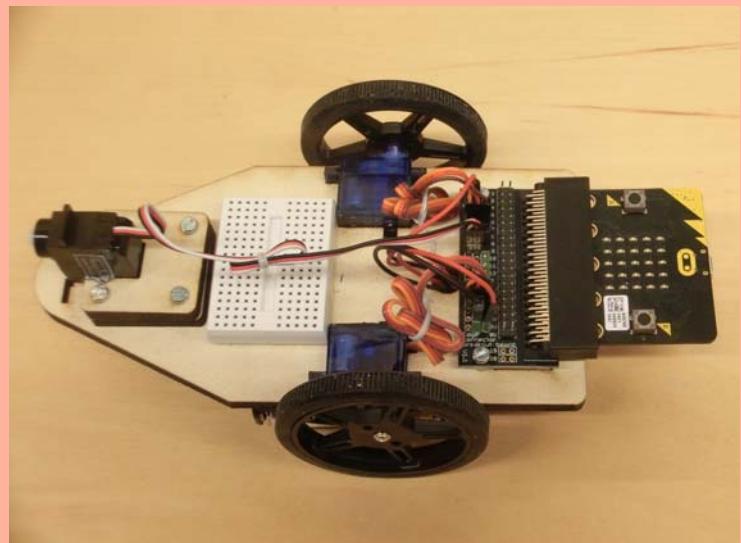


VITENSENTERET  
TRONDHEIM

*Nils Kr. Rossing*

# MICRO:BIT

Forslag til undervisningsopplegg



Laserkuttet og 3D-printet robot med Micro:bit

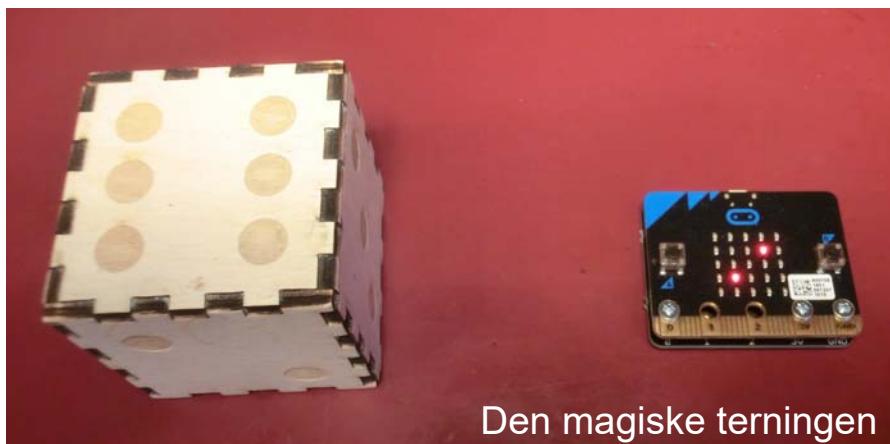
Oktober 2021

Denne siden er blank

# MICRO:BIT

Forslag til undervisningsopplegg

Nils Kr. Rossing



Den magiske terningen

## **Micro:bit - Forslag til undervisningsopplegg**

Trondheim 2021

ISBN 978-82-92088-59-3

Bidragsytere:

*Nils Kr. Rossing, (nkr@vitensenteret.com)* Vitensenteret i Trondheim

Layout og redigering: Nils Kr. Rossing, Vitensenteret i Trondheim

Tekst og bilder: Nils Kr. Rossing, Vitensenteret i Trondheim

Faglige spørsmål rettes til:

**Vitensenteret i Trondheim**

v/Nils Kr. Rossing, 73 59 77 23

[nils.rossing@vitensenteret.com](mailto:nils.rossing@vitensenteret.com)

Kongensgate 1

7011 Trondheim

Postboks 117

7400 Trondheim

Vitensenteret i Trondheim

Telefon: 72 90 90 07

<http://www.vitensenteret.com/>

Rev 6.1 – 16.10.21



---

## Forord

Utgangspunktet for dette arbeidet var en henvendelse fra Jon Haavie i mars 2017 med forespørsel om det var mulig å lage en enkel robot basert på Micro:bit og laserkuttede deler. Den gang kom dette noe ubeleilig, men i løpet av sommeren 2017 har det vært tid til å gjøre noen enkle forsøk med et laserkuttet chassis og noen 3D-printete deler. Høsten 2017 ble det laget en prototyp som ble prøvd ut under Skaperfestivalen ved Teknisk Museum (Oslo) 9. og 10. november. I november laget Joachim Haagen Skeie et utlegg for et kretskort for tilkobling av servoer med mере til Micro:bit. Disse leveres både som byggesett og som ferdig oppkoblet og gjør byggingen av robotten enklere og mer fleksibel. I løpet av jula 2017 ble det laget en ny prototyp som prøves ut under formidlersamlingen i Sandnes jan./feb. 2018. I tillegg vil robotten bli brukt som gjennomgående tema i Makerlærerkurset ved Vitensenteret i Trondheim våren 2018.

Vitensenteret i Trondheim har også klassesett av Micro:bit og Inventors kit og vil være interessert i å kunne tilby disse overfor elevgrupper eller besökende.

Utfordringen har primært vært å framskaffe et konsept som er enkelt nok og som samtidig gir muligheter for flinke elever til å utvikle robotten i de retningene de ønsker. En kan diskutere om et laserkuttet chassis gir den fleksibiliteten vi ønsker, siden det kreves at man må beherske det verktøyet som trengs for å lage underlaget og dessuten har tilgang til en laserkutter. Det samme gjelder de 3D-printede delene. Siden chassiset er relativt enkelt og billig å kutte ut i MDF, finér eller akryl, så kan en tenke seg å selge disse platene til en relativt lav pris. Videre kan en erstatte de 3D-printede delene med ferdig innkjøpt deler som heller ikke koster så mye.

Så det skal ligge en del muligheter i dette designet.

I 2020 ble det aktuelt å gå dypere inn i digital radiokommunikasjon som en del av Fagfornyelsen 2020. Det ble derfor utarbeidet et kapittel om hvordan micro:bit bruker radio for kommunikasjon. Dette ble eksemplifisert ved å beskrive hvordan radioen kan brukes til å fjernstyre bit:bot XL. Opplegget ble testet ut blant 8 lærere i ungdomsskolen i februar 2020 og senere ble et lignende opplegg testet ut under en DeKom-samling for lærere i videregående skole under fagdagen den 6. mars 2020 ledet av Johanna Sexe Skolelaboratoriet og Roy Even Aune Vitensenteret i Trondheim.

Trondheim  
Oktober 2021

Nils Kr. Rossing  
Vitensenteret i Trondheim



---

## Innhold

|   |           |
|---|-----------|
| <b>1 Innledning .....</b>   | <b>11</b> |
| 1.1 Micro:bit og roboter .....                                      | 11        |
| <b>2 Litt teknisk småplukk om Micro:bit .....</b>                   | <b>14</b> |
| 2.1 Kantkontakten .....   | 14        |
| 2.2 Batteripakke for Micro:bit .....                                | 16        |
| 2.3 Radioen i Micro:bit .....                                       | 16        |
| 2.3.1 nRF51822 – Nordic Semiconductor.....                          | 17        |
| 2.3.2 Radioen kan operere på to forskjellige måter.....             | 17        |
| 2.3.3 Peer to peer kommunikasjon .....                              | 18        |
| 2.3.4 Datapakkenes oppbygging .....                                 | 21        |
| 2.3.5 Modulasjon.....   | 22        |
| 2.3.6 Blokker som styrer radiokommunikasjon .....                   | 22        |
| 2.3.7 Overføring av informasjon og styring via radio.....           | 25        |
| <b>3 Programmering av Micro:bit – øvingsoppgaver .....</b>          | <b>27</b> |
| 3.1 Programvare, arbeidsflaten og lagring av prosjektfiler .....    | 27        |
| 3.2 Øvingsopplegg .....   | 30        |
| 3.2.1 Øvingsoppgaver – grunnleggende.....                           | 30        |
| 3.2.2 Forslag til andre øvingsoppgaver .....                        | 33        |
| 3.2.3 Oppgaver knyttet til styring av servoer .....                 | 34        |
| 3.3 Tips til blokkoding med Micro:bit - Noen sentrale blokker ..... | 35        |
| 3.3.1 Bruk av variabler.....  | 35        |
| 3.3.2 Akselerometer - Under katalogen Input.....                    | 36        |
| 3.3.3 Radio - Under katalogen Radio .....                           | 37        |
| 3.3.4 Servo - Under katalogen Pins.....                             | 40        |
| <b>4 Den magiske terningen - en innledende oppgave .....</b>        | <b>42</b> |
| 4.1 Problemstilling .....   | 42        |
| 4.2 Undervisningsopplegg .....                                      | 43        |
| 4.3 Framstilling av den tradisjonelle terningen .....               | 44        |
| 4.4 Programmet .....  | 45        |
| 4.4.1 Program Micro:bit 1 (mottaker) .....                          | 45        |
| 4.4.2 Program Micro:bit 2 (sender).....                             | 47        |
| <b>5 Turbidimeter – måling av partikkeltetthet i væsker .....</b>   | <b>48</b> |
| 5.1 Beskrivelse av prinsippene for et turbidimeter .....            | 48        |
| <b>6 Micro:bit – Forslag til et undervisningsopplegg</b>            |           |



---

|          |  |           |
|----------|--|-----------|
| 5.2      | Oppdraget .....  | 48        |
| 5.2.1    | Montering av lysdiode og lyssensor i kyvettehuset .....        | 49        |
| 5.2.2    | Montering av elektronikken.....                                | 51        |
| 5.3      | Lag programmet .....   | 52        |
| 5.3.1    | Program for å tenne lysdioden og for å avlese lyssensoren..... | 53        |
| 5.4      | Utfør målinger .....   | 56        |
| 5.4.1    | Gjør målinger på 10 beger og sett dem i “rett” rekkefølge..... | 57        |
| 5.4.2    | Skrive ut antall dråper.....                                   | 59        |
| <b>6</b> | <b>Fjernstyring av Bit:Bot .....</b>                           | <b>60</b> |
| 6.1      | Grunnleggende programmering av Bit:Bot .....                   | 61        |
| 6.1.1    | Programmering av sender-enheten .....                          | 61        |
| 6.1.2    | Programmering av mottaker-enheten.....                         | 63        |
| 6.2      | Tilleggsoppgaver .....   | 67        |
| 6.2.1    | Lys og lyd hos Bit:Bot.....                                    | 69        |
| <b>7</b> | <b>Barometrisk høydemåler .....</b>                            | <b>70</b> |
| 7.1      | Modellering av sammenheng mellom lufttrykk og høyde over havet | 70        |
| 7.2      | Måling av lufttrykk .....                                      | 71        |
| 7.3      | Display .....  | 72        |
| 7.4      | Biblioteker .....  | 74        |
| 7.5      | Beregning av eksponenter med micro:bit .....                   | 74        |
| 7.6      | Programmet skrevet i blokkode .....                            | 76        |
| 7.6.1    | Definisjon av variabler.....                                   | 76        |
| 7.6.2    | Innhenting av trykk fra BME280.....                            | 76        |
| 7.6.3    | Beregning av høyden .....                                      | 77        |
| 7.6.4    | Beregning av xn.....   | 77        |
| 7.6.5    | Utskrift .....   | 78        |
| <b>8</b> | <b>Beskrivelse av en selvlaget robot .....</b>                 | <b>79</b> |
| 8.1      | Innledende betraktninger om valg av løsning .....              | 79        |
| 8.1.1    | Chassie .....  | 79        |
| 8.1.2    | Sensorer og aktuatorer er aktuelle for robot .....             | 79        |
| 8.2      | Forslag til undervisningsopplegg .....                         | 80        |
| 8.2.1    | Hensikt:.....  | 80        |
| 8.2.2    | Enkelt eller avansert oppdrag .....                            | 80        |
| 8.2.3    | Anvendelse.....  | 81        |
| 8.3      | Betraktninger om realisering (“Tinkering”) .....               | 82        |



---

|                  |  |            |
|------------------|--|------------|
| 8.3.1            | Chassie.....   | 82         |
| 8.3.2            | Hjul.....  | 84         |
| 8.3.3            | Servoer.....   | 87         |
| 8.3.4            | Tegnefunksjon .....  | 88         |
| 8.3.5            | Batterier.....   | 91         |
| 8.3.6            | Kantkontakt- og Micro:bit kort .....   | 92         |
| 8.3.7            | Alternative sensorer.....  | 96         |
| 8.4              | Detaljerte byggebeskrivelser .....   | 98         |
| 8.4.1            | Byggebeskrivelse for Kodegenets Servo:bot-kort for Micro:bit...                    | 98         |
| 8.5              | Test av kortet og servoene .....   | 103        |
| 8.6              | Byggebeskrivelse robot .....   | 106        |
| 8.6.1            | Byggebeskrivelse – Grunnenhet med chassis .....                                    | 107        |
| 8.6.2            | Byggebeskrivelse – Tegneenhet.....   | 112        |
| 8.7              | Programmering av Micro:bit roboten .....   | 114        |
| 8.7.1            | Styring av roboten .....   | 115        |
| 8.7.2            | Programmering av senderenheten .....   | 116        |
| 8.7.3            | Programmering av mottakerenheten .....   | 117        |
| 8.7.4            | Program for styring av penna .....   | 117        |
| 8.7.5            | Kombinert program med kjørekontroll og penn.....                                   | 120        |
| <b>Vedlegg A</b> | <b>Komponentliste .....</b>  | <b>122</b> |
| A.1              | Komponent og innkjøpsliste .....   | 122        |
| <b>Vedlegg B</b> | <b>Løsningsforslag øvingsoppgaver .....</b>  | <b>123</b> |
| B.1              | Løsningsforslag øving 1 – Lag en animasjon .....                                   | 123        |
| B.2              | Løsningsforslag øving 2 – Bruk av variabler .....                                  | 124        |
| B.3              | Løsningsforslag øving 3 – Bruk av akselerometer .....                              | 124        |
| B.4              | Øving 4 Overføring av informasjon via radio .....                                  | 125        |
| B.5              | Løsningsforslag skritteller .....  | 126        |
| B.6              | Løsningsforslag lysteremin .....   | 126        |
| B.7              | Løsningsforslag til oppgave 7 –<br>Lag kompass som peker mot nord .....            | 126        |
| B.8              | Løsningsforslag til oppgave 8 –<br>Når Micro:bit ”forstår” hva som blir sagt ..... | 129        |
| B.9              | Styring av 360° servo .....  | 129        |
| B.10             | Løsningsforslag til Turbidimeter .....   | 130        |
| <b>Vedlegg C</b> | <b>OpenSCAD kode .....</b>   | <b>134</b> |



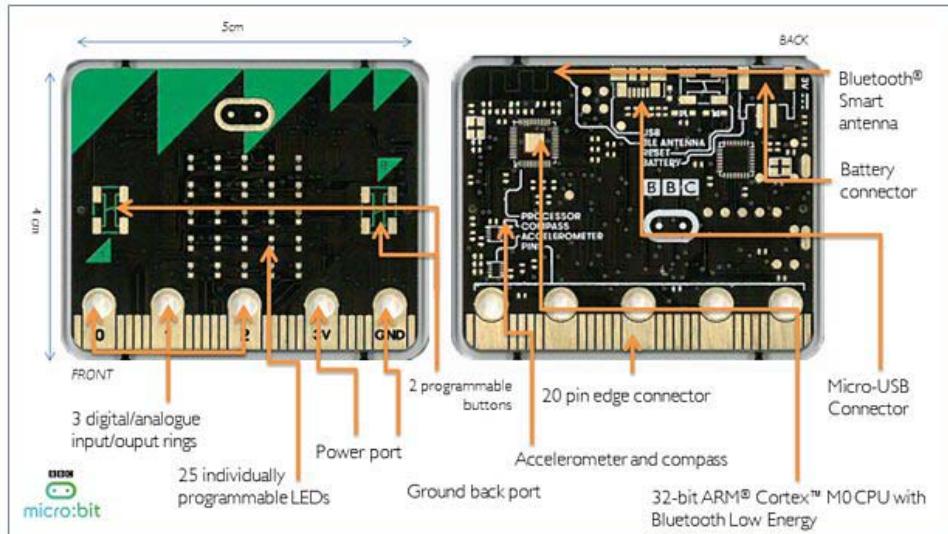
---

|                  |   |            |
|------------------|---|------------|
| C.1              | Holder for nesehjul .....                       | 134        |
| C.2              | Deler for løfting av penn .....                 | 134        |
| <b>Vedlegg D</b> | <b>Læreplaner .....</b>                         | <b>136</b> |
| D.1              | Teknologi og Forskningslære 1 .....             | 136        |
| D.2              | Teknologi og forskningslære 2 .....             | 137        |
| D.3              | Teknologi i Praksis (TiP) – ungdomsskolen ..... | 138        |
| D.4              | Programmering – ungdomsskolen .....             | 139        |
| <b>Vedlegg E</b> | <b>Filer for laserkutter .....</b>              | <b>142</b> |
| E.1              | Chassie for robot .....                         | 142        |
| E.2              | Magisk terning .....                            | 143        |



# 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, men gir samtidig store muligheter.

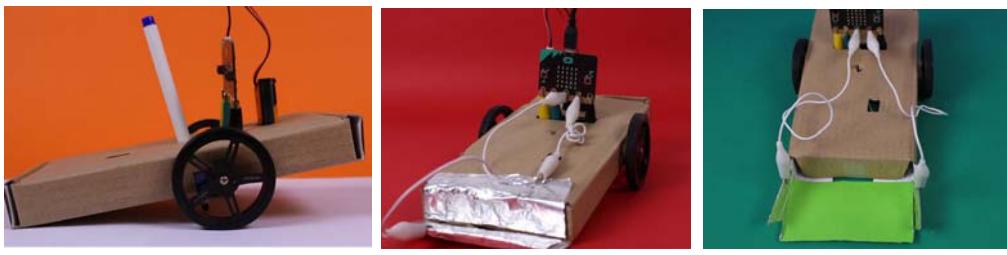


Kretsen har dessuten et 3 akse akselerometer og magnetometer slik at den kan detektere helningen til Micro:bit-kortet. Den kan dessuten fungere som kompass. Videre kan diode-displayet brukes som lysdetektorer.

## 1.1 Micro:bit og roboter

På markedet finnes det en mengde roboter bygget omkring Micro:bit, ikke minst roboter laget av enkle materialer. Her er noen eksempler som er i salg.

Bildet under viser en roboter bygget opp av meget billige materialer.



Artbot

Roombot

Golfbot



---

Artbot kan holde en pen og tegne på et papir mens den beveger seg. Roombot har en kollisjonsbryter foran laget av aluminiumsfolie som gjør den i stand til å detektere sammenstøt og på den måten snu og styre unna. Golfbot detekterer en ball som ruller inn i skuffen foran. Når ballen legger seg i skuffen sluttet en krets gjennom aluminiumsfolien<sup>1</sup>. Det fine med disse er at barna selv kan utforme og videreutvikle roboten selv med hjelpemidler de finner hjemme. Det forutsettes imidlertid at de har en Micro:bit. Pris for settet: ca. 20£.

### Kitronik Micro:bit buggy

Kitronik leverer også en linjefølgende robot: *Micro:bit buggy*<sup>2</sup>.

Denne leveres om byggesett og krever litt loddning. Den er basert på et DC-motor driverkort med sokkel for Micro:bit-kortet. To DC-motorer med utveksling driver hjulene. Tre lysensorer er festet foran på undersiden og fungerer som sensor for følging av en svart linje.

Pris € 32 Micro:bit-kort ikke inkludert.



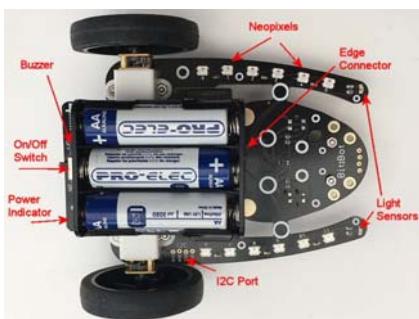
### 4tronix Robo:Bit Buggy<sup>3</sup>

har laget en interessant robotvariant med Micro:bit. Denne anvender akselerometeret for å detektere kollisjoner. Dessuten kan den styres trådløst via en annen Micro:bit. Med tillegg i prisen kan den utstyres med ultralyd avstandssensor og sensorer for å følge en linje.

Pris: £ 22,- hos Rapid electronics



### 4tronix Bit:Bot<sup>4</sup>



Denne roboten er basert på standard DC-motorer med utveksling som forsynes med 3 x AA batterier. 12 lysdioder (Neopixel) er montert på armer på hver side av roboten, og to digitale reflektanssensorer for å følge en linje er montert under foran. En lydgiver (buzzer) kan programmeres til å gi lyd. Det følger dessuten med en innretning for montering av en penn. Roboten kan dessuten utstyres med flere sensorer. En ulempe er at pennen er plassert så langt bak, dette gjør det utfordrende å lage programmer for å tegne spesifikke tegninger. Dette er rettet opp i

den neste utgaven bit:bot XL hvor pennen er plassert midt mellom hjulene. I tillegg er leveres den med en sokkel for montasje av en ultralyd avstandssensor. Pris bit:bot XL: 519,- + MVA.

Dette er et lite utvalg av roboter som finnes på markedet og som benytter Micro:bit.

- 
1. <https://www.techwillsaveus.com/shop/microbit-microbot-pack/>
  2. [https://www.stuff.tv/features/how-masterthe-bbc-microbit/master-1-microbit-crane?\\_format=amp](https://www.stuff.tv/features/how-masterthe-bbc-microbit/master-1-microbit-crane?_format=amp)
  3. <https://www.rapidonline.com/4tronix-robo-bit-buggy-for-bbc-micro-bit-75-0123>
  4. [https://www.rapidonline.com/pdf/75-0117\\_v1.pdf](https://www.rapidonline.com/pdf/75-0117_v1.pdf)  
<https://www.rapidonline.com/4tronix-robo-bit-buggy-for-bbc-micro-bit-75-0123>



---

Hva kan så vår robot tilføre dette mangfoldet?

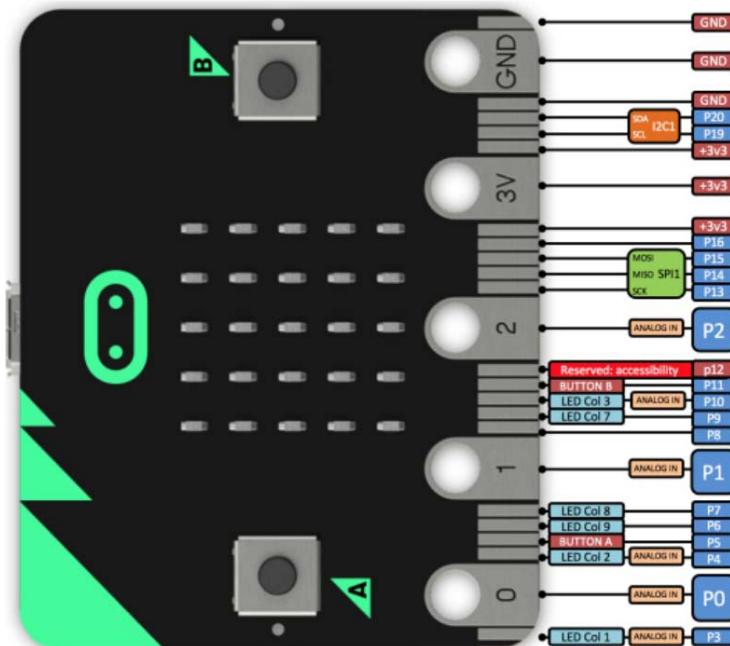


## 2 Litt teknisk småplukk om Micro:bit

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). Forsyningsspenningen er tilknyttet tre pinner. Dessuten er tre pinner koblet til jord. Selv om mange sensorer og *break out korts*<sup>5</sup> anvender 3,3 V, så er det også

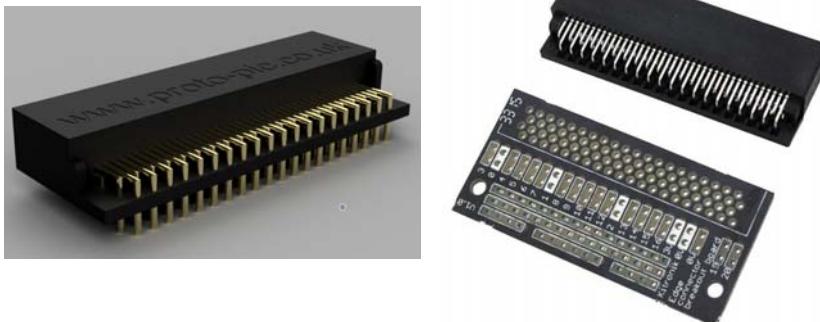
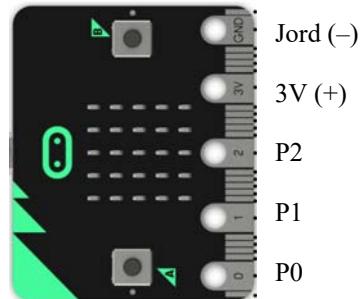


---

mange som trenger en arbeidsspenning på 5 V. Den roboten vi skal bygge har en 6 V kilde som kan senkes til 3,3 V om nødvendig.

Pinne P19 (SCL) og P20 (SDA) kan også brukes til kommunikasjon via seriebuss (I<sup>2</sup>C). 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).

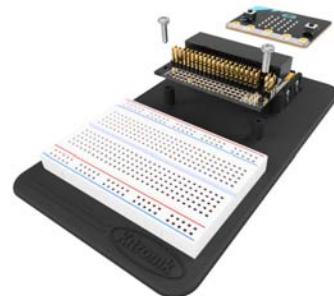
Fem av tilkoblingspunktene er utvidet og gjennomhullet slik at det skal være mulig å koble til bananstikkere eller krokodilleklemmer. Dette gjelder batterispenningen 3V (+ og -) og portene P0, P1 og P2. Dette gjør at en kan komme igang svært fort med et minimum av tilleggsutstyr. Samtidig som det ligger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



Det finnes kantkontakte 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 og kretskort som kan utstyret med dobbel stiftlist.

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 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.



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



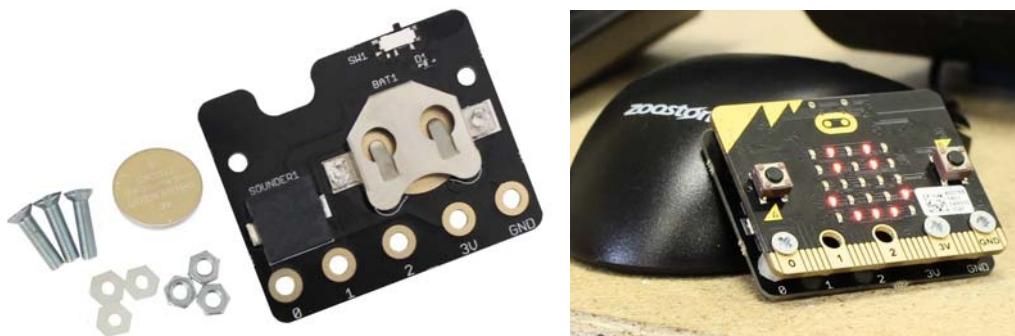
Nylig er det blitt utviklet et kort med kantkontakt av Joachim Hagen Skeie som driver Kodegenet AS. Kortet har fått navnet Micro:bit Servo:bot, og er spesielt laget for å kunne brukes for å bygge små roboter. Kortet kan fås som byggesett eller ferdig bygget.

Det er forberedt for å kunne koble til 4 servoer, avstandssensor og to Neopixler (fargete LED). Vi skal bruke kortet til vår robot.



## 2.2 Batteripakke for Micro:bit

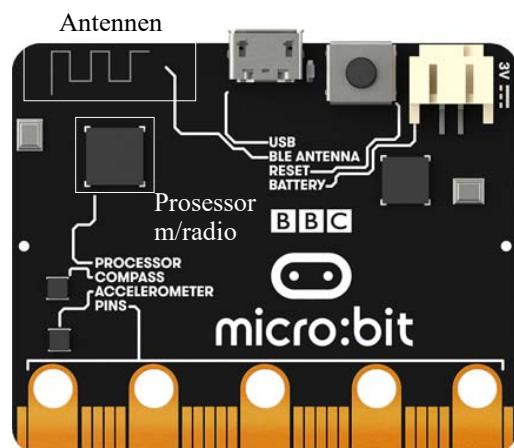
Dersom Micro:bit skal brukes uten å være tilkoblet PC, er det praktisk å montere den på et batteridekk (Power Back) som inneholder batteri (CR2032 - 3V), lydgiver (piezo elektrisk) og bryter. Dekket monteres til Micro:bit med tre maskinskruer med mutter og avstandsstykker.



## 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 radio-modulen er integrert i prosessoren som er den svarte kvadratiske kretsen under antennen.



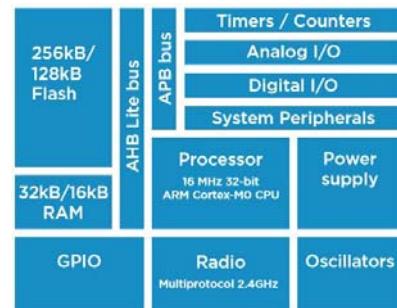
## Sendefrekvens og antennelengde

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<sup>6</sup> og andre kortdistanse sendere. 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 antennen bli noe kortere enn om den hadde vært strekt ut i rommet.

### 2.3.1 nRF51822 – Nordic Semiconductor<sup>7</sup>

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.



Blokkdiagram for nRF51822

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

### 2.3.2 Radioen kan operere på to forskjellige måter

Radioen kan operere på to forskjellige måter<sup>8</sup>:

- 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 befinner 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

- Mikrobølgeovner opererer normalt på 2,4GHz Mikrobølgeovner med lekkasje av mikrobølger kan derfor lett forstyrre kommunikasjonen mellom micro:bits.
- <https://www.nordicsemi.com/-/media/Software-and-other-downloads/Product-Briefs/nRF51822-product-brief.pdf>
- <https://www.littlebird.com.au/a/how-to/112/bluetooth-with-micro-bit>



som er organisert på en bestemt måte, dvs. en “pakke”. Skal større mengder data overføres så sendes flere pakker etter hverandre.

## 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. mobiltelefoner. Bluetooth opererer vanligvis i frekvensområdet 2,400 GHz – 2,485 GHz (hele ISM<sup>9</sup>-båndet er fra 2.4 – 2,5 GHz, en båndbredde på 100 MHz<sup>10</sup>, 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”<sup>11</sup>.

Senderdelen av radioen kan programmeres til å sende med forskjellige effekter fra -20dBm til +4dBm Hvilket betyr fra ca. 0,01 mW til 2-3 mW som er svært små effekter<sup>12</sup>, men nok for kortdistansekommunikasjon (opp til ca. 70 m med 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<sup>13</sup>.

### 2.3.3 Peer to peer kommunikasjon<sup>14</sup>

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

Den pakken som brukes i dette tilfellet er spesiell for Nordic Semiconductor (og kan omtales som proprietær) og går under navnet Nordic Gazell. I denne protokollen er hver pakke merket med en gruppekode (“group code”) som inneholder en adresse og annen informasjon som er nødvendig

9. ISM - “Industrial, scientific and medical”

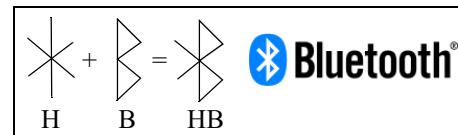
10.[https://en.wikipedia.org/wiki/ISM\\_band](https://en.wikipedia.org/wiki/ISM_band)

11.<https://lancaster-university.github.io/microbit-docs/ble/profile/>

12.Et vanlig mobiltelefon kan sende på effekter opp til 2Watt, riktig nok i kort pulser.

13.kbps - kilo bit pr. sekund. Mbps - Mega bit pr. sekund

14.[https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.sdk51.v9.0.0%2Fgzell\\_02\\_user\\_guide.html&cp=4\\_0\\_7\\_5\\_0\\_2](https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.sdk51.v9.0.0%2Fgzell_02_user_guide.html&cp=4_0_7_5_0_2)

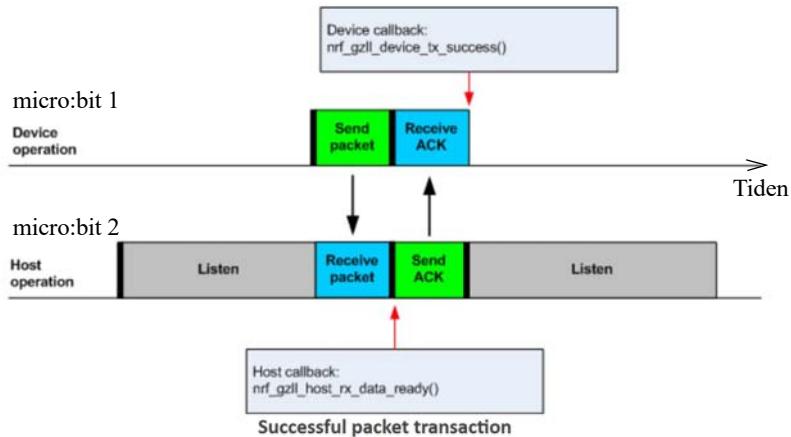


Hagall Bjarkan

Det fortelles at da de tre industrigiantene 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 forne 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.

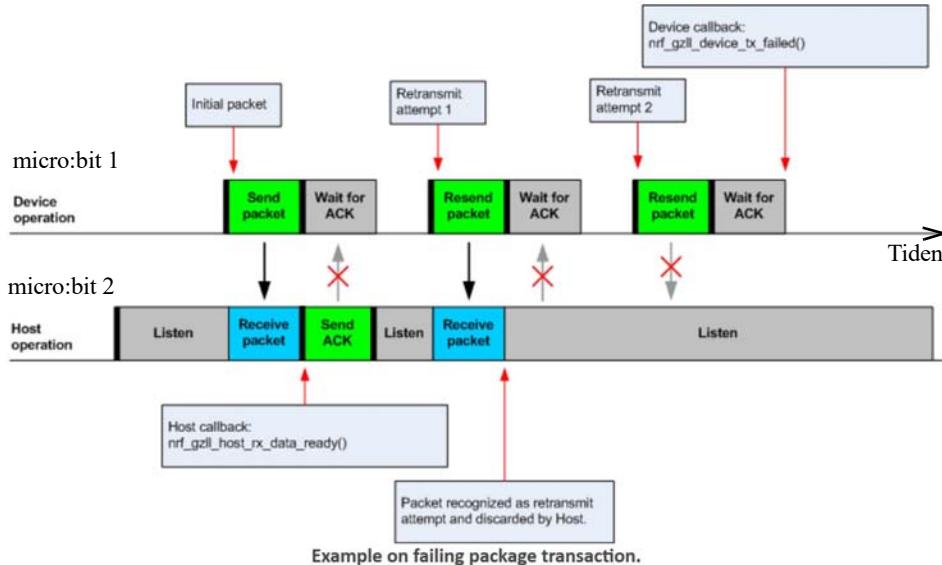
for at dataene skal komme fram til adressaten på rett måte. Det er denne adressen som settes når vi velger "gruppe" når vi skal kommunisere med micro:bits.

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



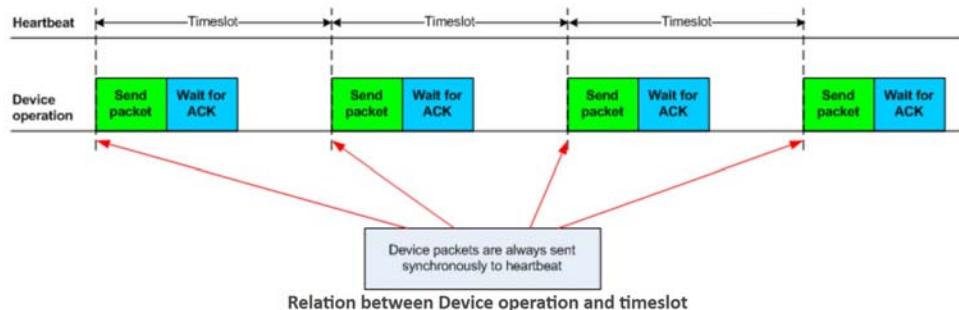
Vi tenker oss at micro:bit 1 (Device) "ønsker" å sende en melding til micro:bit 2 (Host). Micro:bit 1 sender en "pakke" med bit. Denne mottas av flere micro:bits deriblant micro:bit 2. Denne sjekker adressen (gruppe) for å se om pakken er ment for den. Når alle bitene i pakken er mottatt, sendes det 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 sende pakken på nytt. En pakke vil kunne bli sendt om igjen flere ganger (3 ganger) før senderen gir opp.





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.

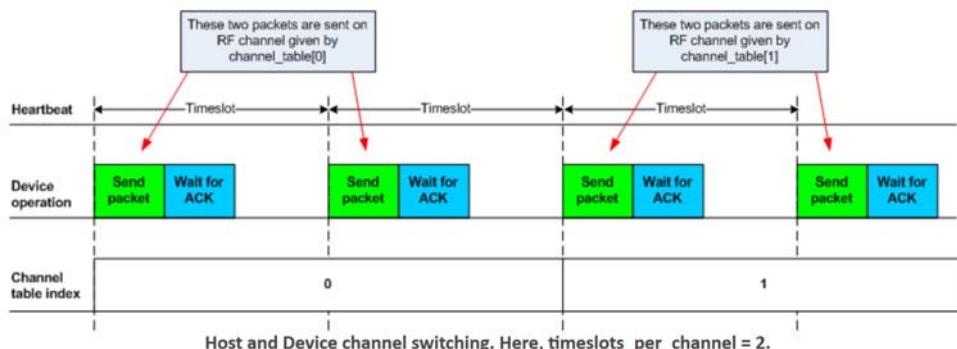


Dvs. at tidsrommet for hver ny pakke som sendes er hele tiden det samme (*Timeslot*). Dette krever at både sender og mottaker er synkronisert i forhold til hverandre. Lengden av et “timeslot” varierer med datahastigheten<sup>15</sup>:

- For 2 MBit/sek er timeslot perioden  $\geq 600 \mu\text{s}$ .
- For 1 MBit/sek er timeslot perioden  $\geq 900 \mu\text{s}$ .
- For 250 kBit/sek er timeslot perioden  $\geq 2700 \mu\text{s}$ .

### 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



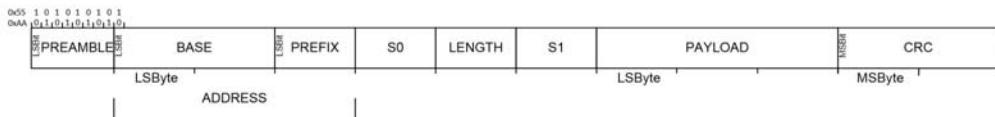
Når senderen skifter frekvens, så må også mottakeren skifte samtidig, ellers mister den pakken.

15. [https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.sdk51.v9.0.0%2Fg\\_zll\\_02\\_user\\_guide.html&cp=4\\_0\\_7\\_5\\_0\\_2](https://infocenter.nordicssemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.sdk51.v9.0.0%2Fg_zll_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 skjer synkront.

### 2.3.4 Datapakkene oppbygging<sup>16</sup>

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



#### “Preamble” (innledning)

Dette er en sekvens av 01010101 (0x55) eller 10101010 (0xAA)<sup>17</sup>. 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 nyttinformasjonen (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.

16.[https://infocenter.nordicsemi.com/pdf/nRF51\\_RM\\_v3.0.1.pdf?cp=5\\_2\\_0](https://infocenter.nordicsemi.com/pdf/nRF51_RM_v3.0.1.pdf?cp=5_2_0) side 82

17.0xAA er tall uttrykt i det hexadesimale tallsystemet f.eks. 1010 1010 = AA hvor man benytter grunn tall 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)



## 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 feilene slik at meldingen blir riktig. Blir antallet feil for stort, klarer ikke disse to bytene å rette opp feilen men 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 langsmmere 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*.

### 2.3.6 Blokker som styrer radiokommunikasjon<sup>18</sup>

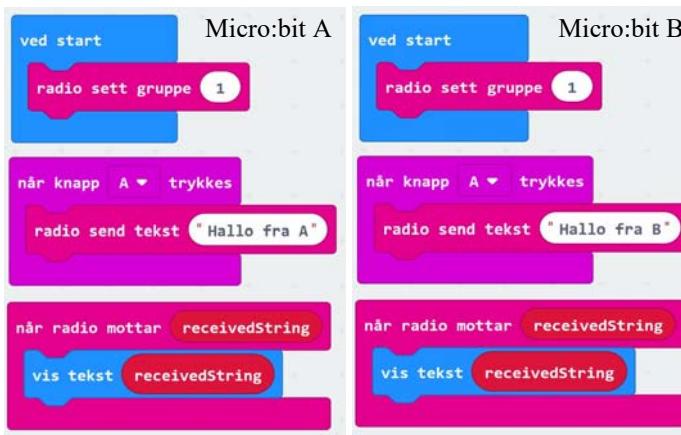
Micro:bit er tilrettelagt for å sende og motta beskjeder:

- ... 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.

<sup>18</sup>Mye av dette stoffet er hentet fra Halfacree Gareth, The Official BBC Micro:bit User Guide, Wiley 2018, kapittel 8



## 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 radio sett gruppe 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

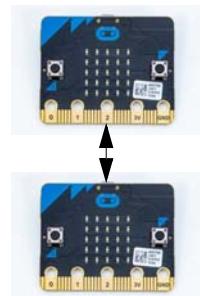
fra B" når det trykkes på knapp B.

*Her programmeres begge micro:bits til å tilhøre gruppe 1. Det betyr både at de opererer med samme frekvenskjema (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.*

### Prosjekt 1: Dørklokke og døråpner

- 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 når du trykker på knapp A.

Gruppe 1



## 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.

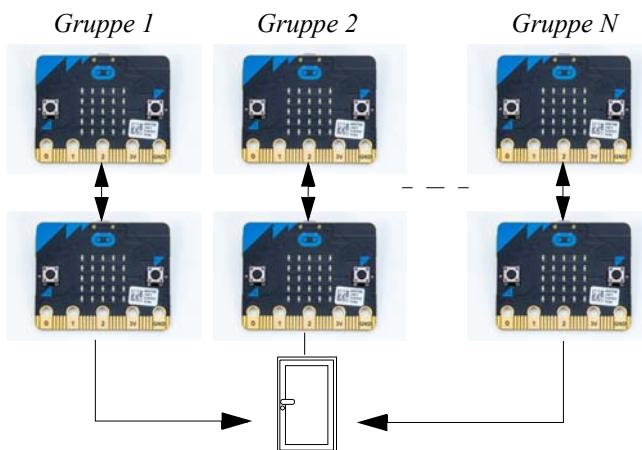
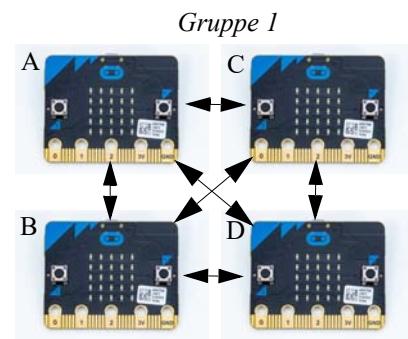


I dette tilfellet vil det når knapp A trykkes hos micro:bit A gå ut melding til samtlige micro:bit som er koblet opp i nettverket. Det samme gjelder også for de andre. Dvs. at alle kan sende meldinger til alle.

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

#### Prosjekt 2: 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. Dernest må alle micro:bit ved inngangen kunne åpne den samme døra.

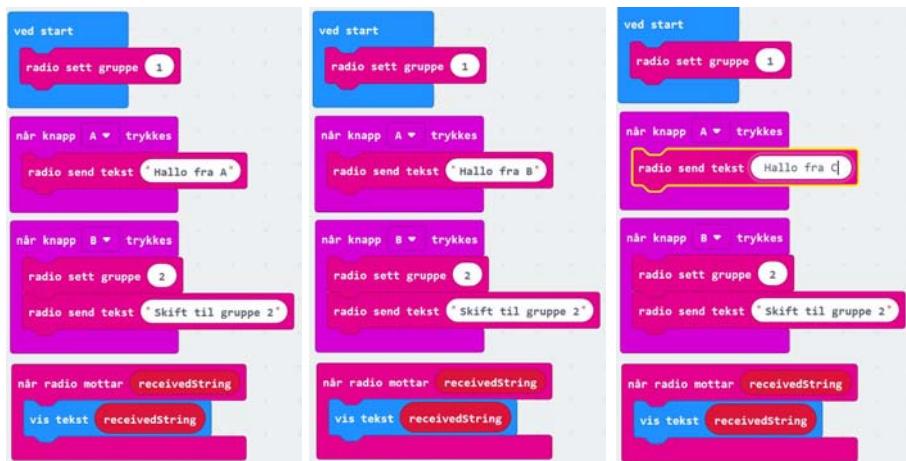


#### Å sette opp gruppekommunikasjon

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.



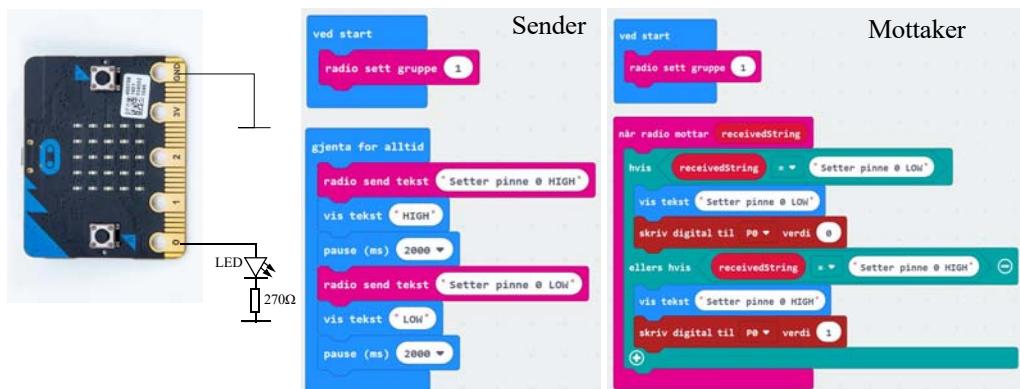
Vi legger nå inn et skifte av gruppe når vi trykker på knapp B. For å teste ut denne egenskapen så trenger vi tre mikro:bit, A, B og C.



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.

### 2.3.7 Overføring av informasjon og styring 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.





---

Koden i figuren over bruker en tekststreng til å overføre informasjon i tillegg til at den slår av og på en lysdiode. En enkel hvis-blokk (if-setning) sjekker om teksten sender 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.



### 3 Programmering av Micro:bit – øvingsoppgaver<sup>19</sup>

Siden blokkprogrammering for Micro:bit er ganske intuitiv, spesielt for de som har litt erfaring med programmering fra før, er det kun nødvendig med en meget enkel introduksjon.

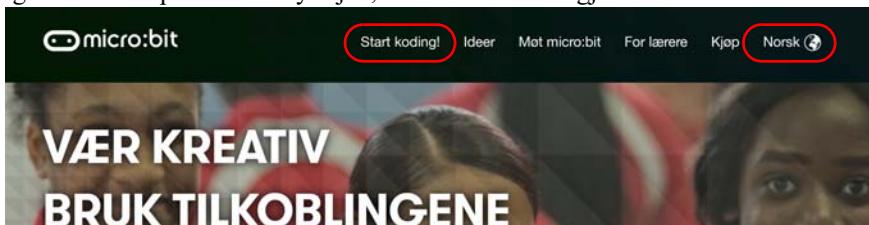
#### 3.1 Programvare, arbeidsflaten og lagring av prosjektfiler

Micro:bit kan kodes med ulike språk bl.a. ved hjelp av Python, Java eller *blokkoding* 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*.

##### Nettressursen

De fleste nettsesere kan brukes, men kan gi litt varierende prosedyrer for lagring og overføring av filer. For de som bruker Chromebook så er nettseseren gitt. Noen anbefaler bruk av Chrome, versjon 65 eller nyere, da fungerer webUSB som gjør overføring av filer fra datamaskin til microbit til en lek.

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. Gå ned til den delen av siden som er vist på bildet under, og trykk på *Start koding*:

##### MakeCode-editor

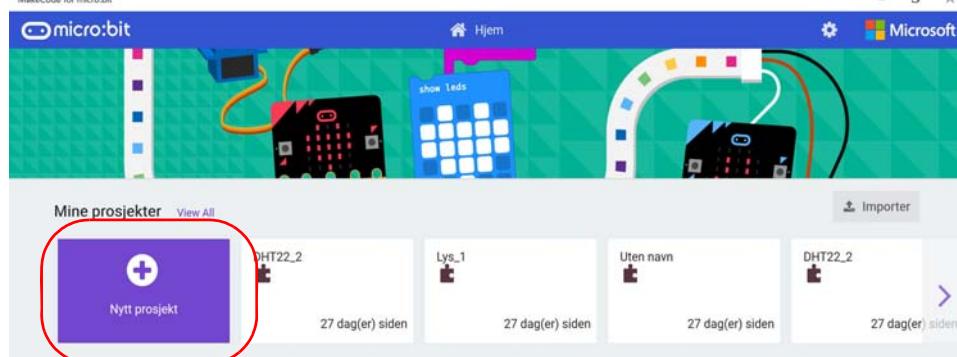
MakeCode-editoren fra Microsoft gjør det enkelt å programmere micro:bit-en din med klosser og JavaScript. Finn ut mer om [de nyeste oppdateringene i MakeCode](#).

Hvis du har problemer med å få tilgang til editoren, eller du vil bruke den uten Internett-tilkobling, sjekk "ofte stilte spørsmål".

19. Bygger på erfaringer som Jon Haavie fra Teknoteket ved Teknisk museum i Oslo



4. Du får da opp følgende vindu:



Velg *Nytt prosjekt*

5. Skriv inn navnet på ditt prosjekt:

*Mitt\_prosjekt-1*

Vi ser for oss å lage mange *Mitt\_prosjekt*-delprogrammer (*Mitt\_prosjekt-1,-2,-3 ...*) som vi til sist kan sette sammen til et større program etter ønske.

Create a Project 😊😊😊



Gi prosjektet ditt et navn.

*Mitt\_prosjekt-1*

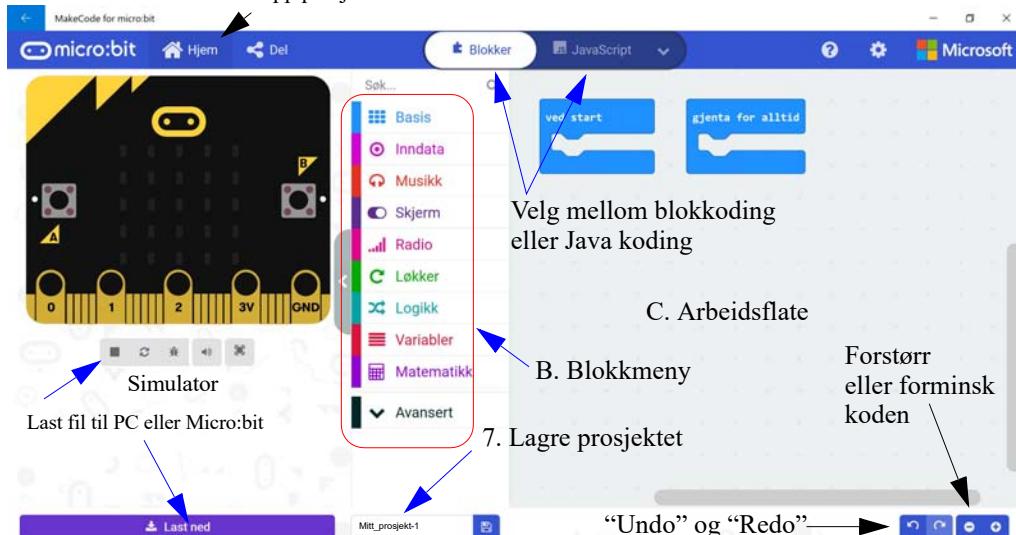


Kodealternativer

Create ✓

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

Last opp prosjekt



Her er en kort beskrivelse av gangen i arbeidet:



## Lagre programfilen

### 7. Lagre prosjektet

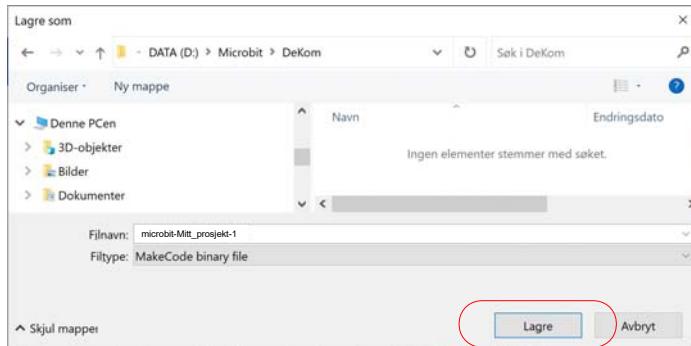
Du skrev inn prosjektnavnet når du gikk inn i programmet.

Mitt\_prosjekt-1



Det kan være lurt å lagre prosjektet først som sist. Du trykker

da på disketten til høyre for prosjektnavnet og får mulighet til å legge filen i en katalog. Pass på å opprett en katalog hvor du finner igjen filen senere. Trykk *Lagre* nederst til høyre.



## Skriv programkoden

### 8. Skriv programmet

Ved å velge blant fanene i blokkmenyen får du opp lister over blokkkommandoer. Disse dras fram av menyen og settes sammen på arbeidsflata slik at blokkene tilsammen blir det ønskede programmet som i eksempelet vist på figuren under. Vi skal ganske snart se nærmere på hvordan programmet er bygget opp

The screenshot illustrates the Microsoft MakeCode environment. On the left, the **Simulator** window shows a digital breadboard with pins labeled A, B, 0, 1, 2, 3V, and GND. In the center, the **Blokkmenyen** (Block menu) is open, displaying various categories like Basis, Inndata, Musikk, Skjerm, Radio, Lekker, Logikk, Variabler, Matematikk, Avansert, Funksjoner, and Tabeller. On the right, the code editor displays the following script:

```
gjenta for alltid
    skriv digital til P2 = verdi 0
    skriv digital til P0 = verdi 1
    pause (ms) 1000
    skriv digital til P0 = verdi 0
    skriv digital til P1 = verdi 1
    pause (ms) 1000
    skriv digital til P1 = verdi 0
    skriv digital til P2 = verdi 1
    pause (ms) 1000
```

At the bottom of the interface, there are buttons for **Last ned** (Download), **Mitt\_prosjekt-1** (Project Name), and **Lagre** (Save), with the **Lagre** button highlighted with a red circle.



---

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:biten påvirker programmet. Det kan være praktisk å bruke simulatoren for å teste ut programmer før man overfører det til den fysiske micro:biten. Den oransje fargen lengst til venstre indikerer at denne kontakten har spenning (3V). Den skal kanskje slå på en lysdiode.

## Overføring av programmet til micro:bit

### 9. Lagre filen

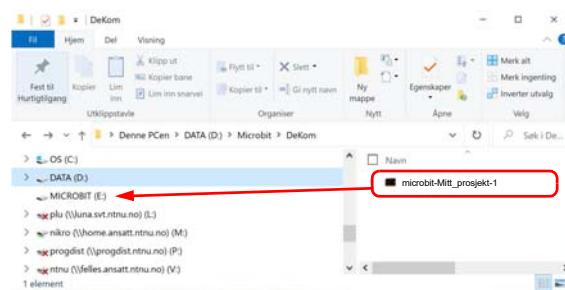
Før du flytter filen over til micro:bit, må du lagre de forandringene du har gjort. Dette gjør du ved å trykke på disketten nederst midt på skjermbildet (se figuren over). Du må gjerne skrive over forrige versjon av programmet eller lagre det under et nytt navn.

### 10. Koble til micro:bit

Det er nå på tide å koble til micro:bit til USB-inngangen. Bruk den medfølgende USB-kabelen.

### 11. Overfør filen til micro:bit

Du kan bruke *Filutforskeren* i Windows til å flytte fila fra katalogen der du lagret den og slipp den over symbolet av micro:biten. Du vil se et blinkende gult lys på undersiden av micro:bit idet filen overføres.



Programmet skal nå være overført.

Ønsker du mer informasjon velger du spørsmålsteget på menylinja øverst til høyre. Her har du mulighet til å stille spørsmål eller få opplæring.

## 3.2 Øvingsopplegg

Programmeringsverktøyet finnes på nett på følgende adresse:

<https://makecode.microbit.org/#>

Åpne gjerne programmet i Edge eller Internett Explorer<sup>20</sup>

### 3.2.1 Øvingsoppgaver – grunnleggende

Følgende oppgaver gi en grunnleggende innføring i programmering av Micro:bit. Siktemålet er hele tiden å legge grunnlaget for å kunne forstå programmet som skal brukes til å styre robotten med, både fra sender- og mottakersiden.

1. **Oppgave 1:** Lag en animasjon som beveger seg over displayet. Bruk minst tre bilder.

---

20.Chrome kan oppføre seg noe annerledes enn Internet Explorer



Tips:



Gå i endeløs loop



Skriv animasjon til display

Merk at fargen på blokkene viser hvilken meny de tilhører.

## 2. Oppgave 2A: Bruk av variabler

Lag et program som øker et tall på displayet med 1 hver gang du trykker på knapp A



## 3. Oppgave 2B

Lag et program som:

- ... viser økende antall på displayet for hvert trykk på knapp A og
- ... viser minkende antall på displayet for hvert trykk på knapp B og
- ... nullstilles når begge knappene trykkes samtidig

Tips:



## 4. Oppgave 3A: Lag et program som viser navnet til den ene deltageren på displayet når knapp A trykkes, og som viser navnet til den andre når knapp B trykkes.

Tips:



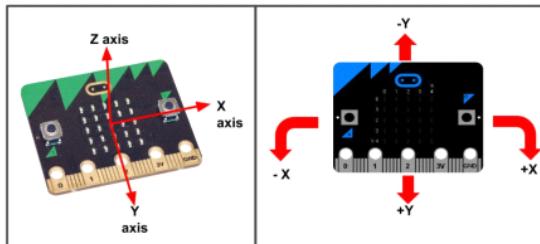
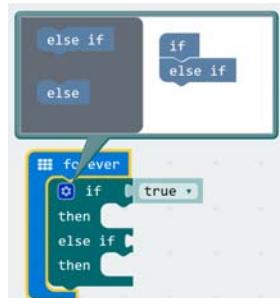
Skriv tekst til display

## 5. Oppgave 3B: Hva kan du gjøre for å vise mer enn to forskjellige navn?

## 6. Oppgave 3C: Bruk akselerometeret for å vise fire forskjellige navn avhengig av hvordan du holder Micro:biten



Tips:



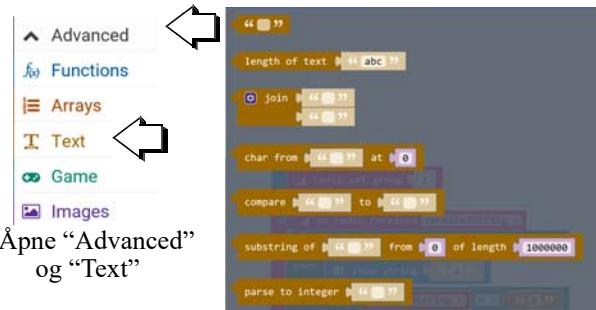
Akselerometerets funksjonalitet, graden av helning angis av et positivt tall i x- eller y-retning

7. **Oppgave 4A:** Micro:bit kan overføre informasjon fra en Micro:bit til en annen. Lag et program som overfører informasjon om hvilken knapp som trykkes på den ene Micro:biten til den andre, og samtidig vises på displayet til den andre.

Tips sender:



Tips mottaker:



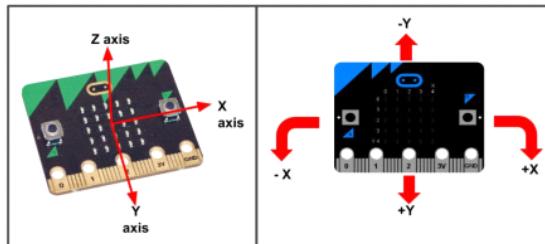


### 3.2.2 Forslag til andre øvingsoppgaver

#### 8. Oppgave 5A: Lag en skritt-teller som teller hvor mange skritt som tas og viser summen av skritt på forespørsel.

TIPS: Forsøk å utnytt at akselerometeret kan brukes til å finne Micro:bits retning i rommet  
Hvem lager den mest pålitelige skritt-telleren?

Tips:



#### 9. Oppgave 5B: La Micro:bit lage en truedelutt når den har talt 10 skritt.

Tips:



#### 10. Oppgave 6: Lag en lysteremin med en tone som varierer med lystyrken på lysdiodene

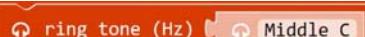
Tips:



Gir verdien 0 ved helt mørke og 255 ved kraftig lys



Omformer ett tallområde til et annet (f.eks. 0 - 1023 -> 0 - 4)



Lager en tone med en gitt frekvens

#### 11. Oppgave 7A: Lag et kompass som peker mot nord

Tips:



Leser av og legger kompassretningen inn i variabelen  
“Kompassretning i grader”



---

Lim neodym magneter på undersiden av bordet og bruk Micro:bit til å finne posisjonen til magnetene slik de framstår på oversiden av bordet.

12. **Oppgave 8:** Når Micro:bit "forstår" hva som blir sagt

La en forsøksperson velge et tall fra 1 til 4

Legg Micro:bit med displayet ned

Når Micro:biten snus rundt skal displayet vise det tallet som forsøkspersonen har sagt

Det er du som snur Micro:bit'en

### 3.2.3 Oppgaver knyttet til styring av servoer

I de etterfølgende oppgaver kreves to 360° servoer og kortet Servo:bot

13. **Oppgave 9A Programmering av 360° servo - Styring**

Lag et program som kjører høyre servo forover når knapp A trykkes og venstre servo forover når knapp B trykkes. Når begge trykkes stopper begge motorene.

Tips:



```
servo skriv pulslengde på P0 til (μs) [1600]
```

P0 - høyre servo

P1 - venstre servo

Pulslengde 1525 – 1800, servo går framover, fart bestemmes av pulslengde

Pulslengde 1475 – 1200, servo går bakover, fart bestemt av pulslengde

Pulslengde 1475 – 1525, servo står i ro

14. **Oppgave 9B Programmering av 360° servo - Styring**

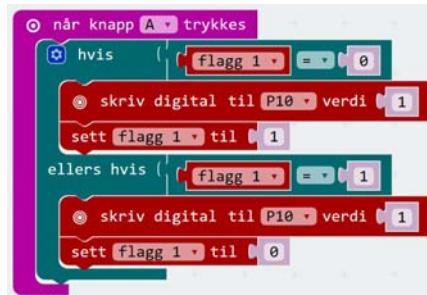
Lag et program som kjører høyre servo framover når knapp A trykkes første gang og bakover når knapp A trykkes 2. gang, og venstre servo forover når knapp B trykkes første gang og bakover når knapp B trykkes 2. gang. Når både A og B trykkes samtidig, stopper begge motorene.



Tips:



Sett flaggene  
innledningsvis til 0



Velg en av to aksjoner avhengig av  
hva som ble gjort sist gang

For å la samme bryter gi to forskjellige responser for annet hvert trykk kan man benytte "flagg". Et flagg er en merkelapp som kan slås av og på ("heises" eller "fires"), avhengig av flaggets tilstand velges en av to responser.

## 15. Oppgave 9C Programmering av 360° servo - Fjernstyring

Lag en fjernstyringsprogram slik at en servoene kan styres som i oppgave A, men via en radio-forbindelse med en Micro:bit nr. 2.

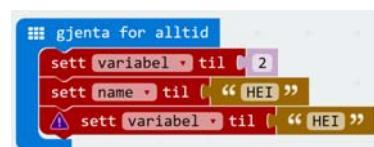
### 3.3 Tips til blokkoding med Micro:bit - Noen sentrale blokker

Her skal vi se på noen blokker i et par undermenyer som er litt spesielle for Micro:bit og nyttige for de formål vi skal bruke dem til.

#### 3.3.1 Bruk av variabler

Når man programmerer med blokkode så har man to typer variable, det er *heltall* og *tekst*. Hvilken som blir hva bestemmes av om man fyller en variabel med tall eller tekststrenger. Det er ikke nødvendig å deklarere variable innledningsvis. Men trenger heller ikke innledningsvis å sette en variabel til 0 eller en tekst-variabel til "ingen tekst", det vil programmet gjøre automatisk ved oppstart.

Tilordning av variabler er vist i figuren over til høyre. Her settes variablene "variabel" til verdien 0, dvs. at variablene "variabel" vil være en tall-variabel hvor det ikke er mulig å skrive inn en bokstav eller tekst. Variablen "name" derimot settes imidlertid til en tekst her lik "HEI" og vil dermed bli en tekstvariabel (tekststregng), og kan da ikke senere i programmet tilordnes et tall. Gjør man det, får man en advarsel som vist nederst på figuren over.





## Endre variabelnavn

Det anbefales ikke å bruke de navnene som automatisk kommer opp ved når man trykker på pilen til høyre for "variabel".

Velger man menyen "Variabler" får man opp listen av "default" variabler (variabel, name, item osv.) i tillegg til de man har definert selv (berøring). Ved å velge "Lag en variabel"

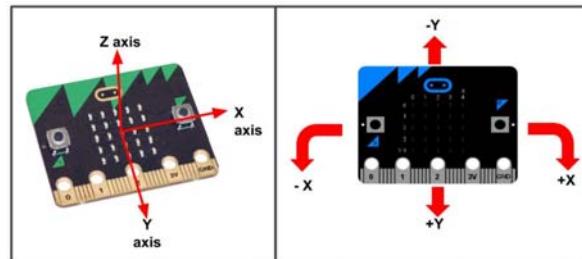
(øverst i lista, kan man definere egne variabler. Dette anbefales sterkt. Bruk navn som er forståelige og umiddelbart gir mening for den som leser programmet. Disse vil da legges til lista.



### 3.3.2 Akselerometer - Under katalogen Input

Denne variablene inneholder løpende avlest verdi av akselerometeret i Micro:bit. Akselerometeret måler akselerasjon i alle tre akseretningene x-, y- og z-retningen og leverer en verdi lik  $\pm 2048$  med benevningen milli-g (mg), hvor  $g = 9.81 \text{ m/s}^2$ . Dvs. at med denne innstillingen (område satt lik 2g) kan Micro:bit måle verdier fra  $-2 \text{ g}$  til  $+2 \text{ g}$ . Området for akselerometeret kan sette til ett av fire områder, se under for nærmere beskrivelse.

Et akselerometer som ligger i ro eller beveger seg i en konstant rettlinjet bevegelse, vil her på jorda kun påvirkes av gravitasjonen. Når Micro:bit ligger flatt på bordet vil vi ideelt sett kun få en verdi i z-retning  $\text{acc}_z = -1000 \text{ milli-g} = -1000 \text{ mg} = -1 \text{ g}$ . Verdien er negativ fordi den positive retningen til z-aksen er rettet oppover fra Micro:bit, mens gravitasjonen virker nedover. Figuren over til høyre<sup>21</sup> viser hvordan aksene til akselerometeret er definert.



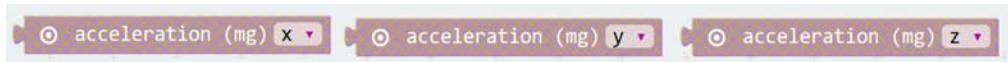
Følgende bevegelser vil gi ulike verdier for x-, y- og z-retningene:

| 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 |

21. <http://microbit-challenges.readthedocs.io/en/latest/tutorials/accelerometer.html>



Med andre ord så kan akselerometeret fortelle oss orienteringen av Micro:bit i rommet. Følgende tre blokkvariabler gjør det mulig å avlese de tre verdiene.



Disse er avleste verdier som kan tilordnes variabler som vist i eksempelet under.



Her tilordnes akselerometerverdien i x-retning til variabelen "x".

Men akselerasjon kan også skje på andre måter enn ved tyngdeakselerasjonen. For eksempel ved at Micro:bit gis en akselererende bevegelse, dvs. at farten øker eller minker.

Kommandoen "Set accelerometer range" kan måleområdet for akselerometeret endres til verdiene: 1, 2, 4 eller 8 g. Avlesningen vil alltid være i milli-g (mg) slik at verdien kan gå opp til  $\pm 8000$  mg.



### 3.3.3 Radio - Under katalogen Radio

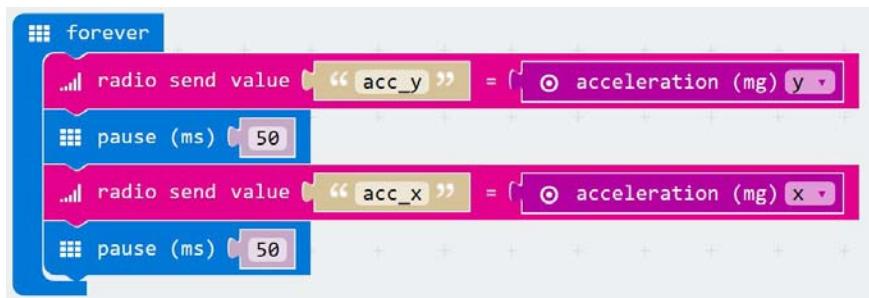
Denne skal vi bruke for å opprette radioforbindelse mellom to eller flere Micro:bits. La oss anta at det er ønskelig å opprette en radioforbindelse mellom to Micro:bits. Det skal overføres et tallverdi, og en tekst som kan knyttes til dette tallet. Dermed er det mulig å overføre forskjellige parametere (verdier) og det er fortsatt mulig å skille verdiene fra hverandre ved hjelp av den "medsendte" tekststrengen.

1. I "on start" funksjonen bestemmes hvilken kanal vi ønsker å opprette radioforbindelsen i ("radio set group ch"). I alt kan man velge mellom 256 kanaler fra 0 – 255.
2. Dernest må man velge et navn og en parameter man ønsker å sende:





Figuren under viser et eksempel der vi sender to parametere knyttet til hver av parametrene "acc\_x" og "acc\_y".

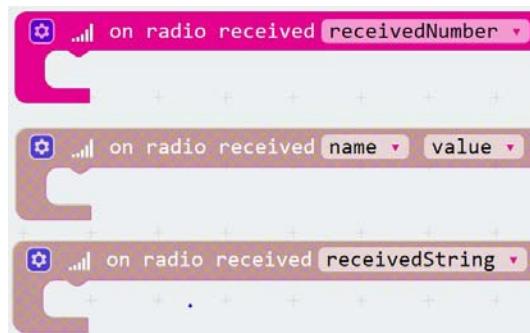


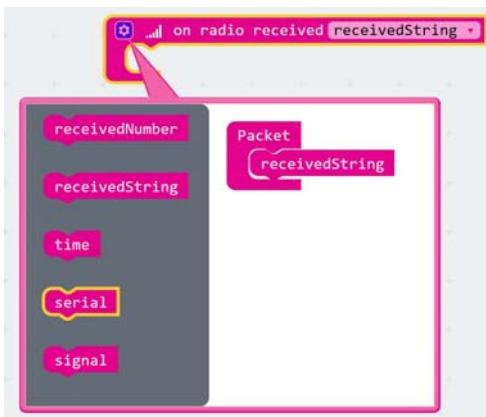
Her ser vi at vi avleser verdien til akselerometeret i x-retning med blokken "acceleration (mg) z" og sender verdien sammen med navnet acc\_x. Tilsvarende sender vi akselerometerverdien i y-retning under navnet acc\_y. La oss se hvordan disse kan skilles igjen på mottakersiden.

3. Også mottakeren må settes opp med riktig kanal. Dette gjøres på samme måte som på sendersiden i "on start" blokken..



4. Vi har ulike kommandoer som kan brukes til å motta og skille ut de ulike parametrene. Under ser vi alle de som ligger i menyen. I virkeligheten er alle disse den samme funksjonen, man har bare valgt ut litt forskjellige parametere.





Dersom vi trykker på det vesle hjulet i venstre hjørne får vi opp samtlige valgmuligheter som vist i figuren under.

Her er en komplett oversikt<sup>22</sup> over parametere som kan velges inn i datapakken som sendes over radiokanalen (Bluetooth):

**receivedNumber:** Er en variabel som inneholder den *verdien* som er knyttet til datapakken (akselerometerverdien i eksempelet foran)

**receivedString:** Er teksten som er knyttet til datapakken. Det kan også være navnet

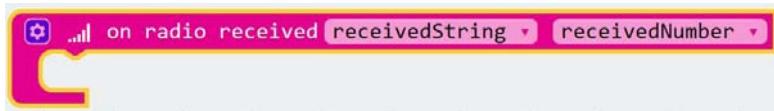
til variablene viss verdi ønskes overført.

**time:** Er systemtidspunktet til Micro:bit som sendte meldingen, dvs. tid fra påslag.

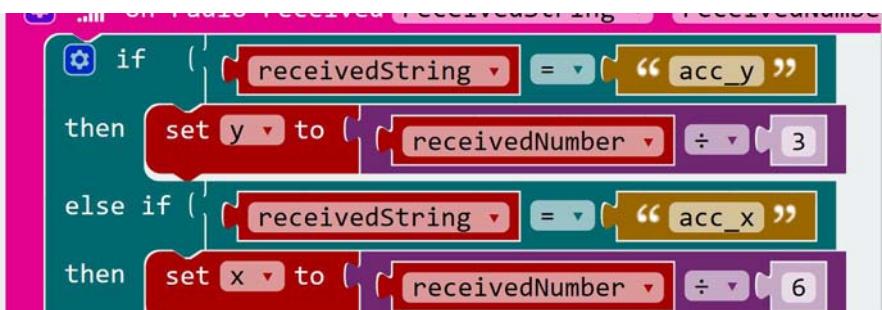
**serial:** Serienummeret til den Micro:bit som sendte meldingen, om den er merket med et serienummer fra fabrikken.

**signal:** Angir signalstyrken til radiosignalet, fra en svakeste verdi på -128 til en sterkeste verdi på -42.

Vi velger å ta med variablene "receivedNumber" og "receivedString" og får følgende argumenter i funksjonen. Her knytter vi "receivedString" ("default" navn) til parameterens navn f.eks. acc\_y og receivedNumber" til en verdi mellom +/- 0 – 1000 som angir akselerometerverdien.



- I figuren under ser vi hvordan vi anvender innholdet i receivedString for å skille de to variablene x\_acc og y\_acc.



22. <https://makecode.microbit.org/reference/radio/on-data-packet-received>



Ved hjelp av betingelsen til if-setningen sjekkes hvilke av de to medfølgende tekststrengene sendingen gjelder slik at de to tilhørende verdiene kan tilordnes de riktige variablene.

6. Dernest kan vi la det etterfølgende programmet bruke de overførte verdiene som nå er tilordnet variabler.

I stedet for å bruke "receivedString" og "receivedNumber" kan definere sine egne variabler som gir mer mening.

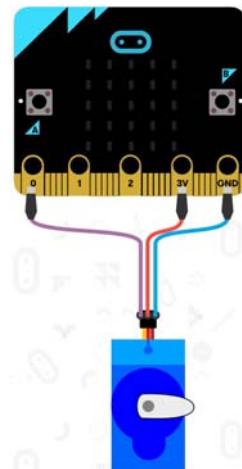
### 3.3.4 Servo - Under katalogen Pins

Denne skal vi bruke for å styre ulike servoer, både 180° og 360° servoer.

Hos en 180° servo kan akslingen dreies 0 – 180°. Slike egner seg for å dreie en arm eller dra i en sleide. En 360° servo roterer kontinuerlig 360° rundt og vil fungere som en motor. Slike egner seg til å kjøre en robot bortover gulvet. Figuren til høyre viser oppkoblingen for en 180° servo. Siden vi bruker P0 som styreutgang kan vi bruke en bananstikker eller en krokodilleklemme.

Vi skal her se nærmere både på blokker som kan styre ut 180° og 360° servoer. Disse blokkene finner vi under blokkfamilien "Pins".

Blokken vist under er primært for styring av en 180° servo.



Her velger vi hvilken pinne som vi anvender til styring og hvor mange grader vi vil at akslingen skal dreie. I eksempelet over vil servoen som er koblet til port P0, dreie 180°.

Når man trykker den vesle pila ved siden av pinnenummeret, vil man få opp en komplett oversikt over de portene som er valgbare.



Skal vi styre en 360° servo så bruker vi kommandoen:



Her velger vi hvilken pinne som vi ønsker å bruke til styring og hvor fort servoen skal rottere. Blokken setter opp den valgte porten (f.eks. P0) som en analog utgang som sender et pulstog med pulser som har en lengde gitt i mikrosekunder (f.eks. 1500μS). Ved å trykke pila ved siden av P0, så vil man få opp alle de mulig valgbare utgangene.

---

360° servoer er ofte konstruert slik at de står i ro for pulslengder på omkring 1500 µs, går med full fart framover ved en pulslengde på 1800 µs og full fart bakover med en pulslengde på ca. 1200 µs.

### Justering av 360° servoer<sup>23</sup>

Det finnes en rekke ulike 360° servoer på markedet. Vi har valgt å bruke *FS90R Continuos servo*. Denne har en liten trimmeskrue på baksiden som gjør det mulig å justere nullpunktet for servoen. Dvs. justere den slik at servoen står stille i et 45µs vindu omkring 1500 µs pulslengden.



---

23.<https://www.pololu.com/product/2820>



## 4 Den magiske terningen - en innledende oppgave

I denne oppgaven skal vi trenere på programmering av noen sentrale egenskaper med Micro:bit:

1. Bruk av akselerometeret
2. Bruk av radiokommunikasjon
3. Bruk if-setninger og betingelser

I tillegg skal vi bruke:

4. Tilstfeldighetsfunksjonen

### 4.1 Problemstilling

Tenk deg følgende: Du har en tradisjonell sekskantet terning med øyne og en elektronisk terning laget ved hjelp av en Micro:bit.



Når man rister på den elektroniske terningen vil den gi tilfeldige antall øyne fra 1 til 6 på displayet.

Så blir man bedt om å trykke inn trykknapp B og holde denne inne et par sekunder. Så rister man på nytt. Det som nå skjer er at resultatet av ristingen gir samme antall øyne hele tiden, nemlig det antall øyne som den tradisjonelle terningen viser. Helt konsekvent.





---

Dersom man snur den tradisjonelle terningen og rister den elektroniske, så vil den vise den nye verdien. Slik er det hver gang, den elektroniske terningen ser ut til å kopiere den tradisjonelle terningen.

Forsök å analyser problemstillingen og kom opp med ulike forslag til hvordan dette er mulig.

## 4.2 Undervisningsopplegg

Vis frem terningen å la elevene utføre eksperimentet som beskrevet foran:

La dem eksperimentere med terningene og se hva som skjer:

- La dem diskutere ulike løsninger for hvordan dette kan være mulig.  
Det vil selvfølgelig være stor forskjell for elever som er ukjent med, og de som er kjente med muligheten i Micro:bit. Oppgaven vil sannsynligvis passe best for de som har litt kjennskap til Micro:bit på forhånd.
- Når de har avslørt hemmelighet, la dem beskrive funksjonen til programmene med ord. Utfordre dem så til å lage en grov skisse av et flytdiagram for programmene. Diskuter løsningene med dem på grunnlag av flytdiagrammene.
- Be dem sette opp et forslag til hvordan de vil gå fram for å designe terningene. Planen bør inneholde en gradvis oppbygging av programmet med milepæler som inkluderer testing av programmet med hårdvare.
- Har de tilgang til en laserkutter kan de jo også designe den tradisjonelle terningen.

### Forslag til plan for programmering

Test programmet i Micro:bit mellom hvert av punktene. Oppgaven krever at man har to stykk Micro:bit: Micro:bit 1 (mottaker) og Micro:bit 2 (sender) og at hver av dem er utstyrt med batteri.

Det kan være lurt å lage flere testprogrammer for å teste ut ulike funksjoner om man er ukjent med hvordan funksjonene fungerer.

#### Lag en riste-terning

1. Lag en elektronisk terning (Micro:bit 1) som skriver ut tilfeldige antall øyne når man trykker på A-knappen  
Tips 1: Bruk “on button A pressed” under *Input*  
Tips 2: Bruk “Pick a random ...” under *Math*
2. La terningen bli styrt av risting. Terningen gir en ny verdi hver gang man rister Micro:bit  
Tips: Bruk “on shake ...” under *Input*

#### Lag en terning som avhenger av orienteringen i rommet

3. Lag en terning (Micro:bit 2) som viser ulike antall øyne avhengig av hvilken side som peker opp. Bruk en tradisjonell terning til å bestemme hvordan øynene skal plasseres.  
Tips: Bruk “acceleration (mg)” under *Input*



## Opprett radioforbindelse

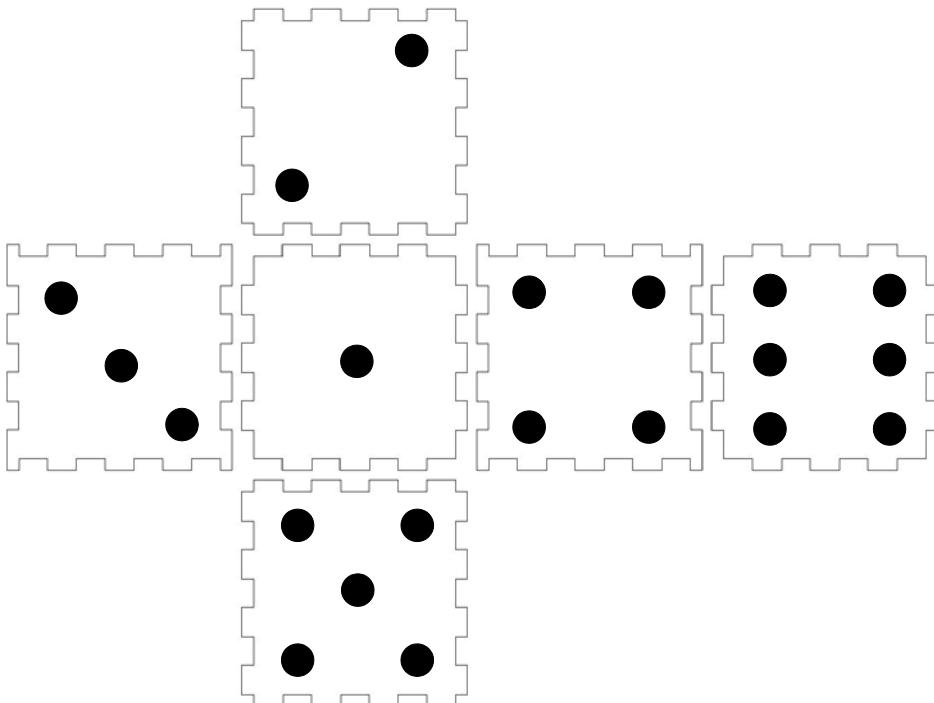
4. Sett opp radiokanalen mellom Micro:bit 2 og Micro:bit 1  
Tips: Bruk “radio set group” under *Radio*
5. Legg inn en send-kommando i Micro:bit 2 som sender terningverdien  
Tips: Bruk “radio send <name> value” under *Radio*
6. Lag et nytt program for Micro:bit 1 som mottar og viser terning-verdien fra Micro:bit 2  
Tips: Bruk “on radio received <mottatt verdi>” under *Radio*

## Lag fjernstyrт terning

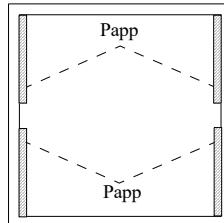
7. Kombiner programmet som mottar terning-verdier fra Micro:bit 2 med den tilfeldige terningsfunksjonen. Bruk A-knappen og B-knappen til å gå mellom vanlig terning til radiostyrт terning (trykk B-knappen) og tilbake (trykk A-knappen).  
Tips 1: Bruk “if then” og “if then else” under *Logic*  
Tips 2: Bruk flagg-variabler for å huske siste trykk på bryterne

### 4.3 Framstilling av den tradisjonelle terningen

Terningen kan lett framstilles ved å bruke et av mange boksprogrammer. Det kan være praktisk å bruk et program som gir mulighet til å angi innvendige mål da disse må passe til bredden til en Micro:bit.



Til dette designet har vi brukt: Maker Case:  
<http://www.makercase.com/> og øyne er skrevet på senere i programmet FlexiDesign. Fire små pappbiter er benyttet for å lage spor til Micro:bit inne i terningen slik at den skal ligge i ro.



## 4.4 Programmet

Programmet består av to programmer, ett for hver av de to Micro:bit. Det ene er for Micro:bit 1 som ligger inne i terningen og som skal registrere terningens posisjon og sende ut verdier. Det andre programmet skal dels kunne generere tilfeldige tallverdier fra 1 – 6 eller motta og vise de mottatte verdiene fra Micro:bit 2.

### 4.4.1 Program Micro:bit 1 (mottaker)

Figuren under viser det blokkprogrammerte programmet.

```

on start
    radio set group [1]
    set [A-flag v] to [true v]
    set [B-flag v] to [false v]

when [radio received v] [mottatt verdi]
    set [terning verdi v] to [mottatt verdi v]
    if [button A v] is pressed
        then
            set [A-flag v] to [true v]
            set [B-flag v] to [false v]
        else if [button B v] is pressed
            then
                set [B-flag v] to [true v]
                set [A-flag v] to [false v]

```



The image displays two Scratch scripts side-by-side, both utilizing the Micro:bit extension.

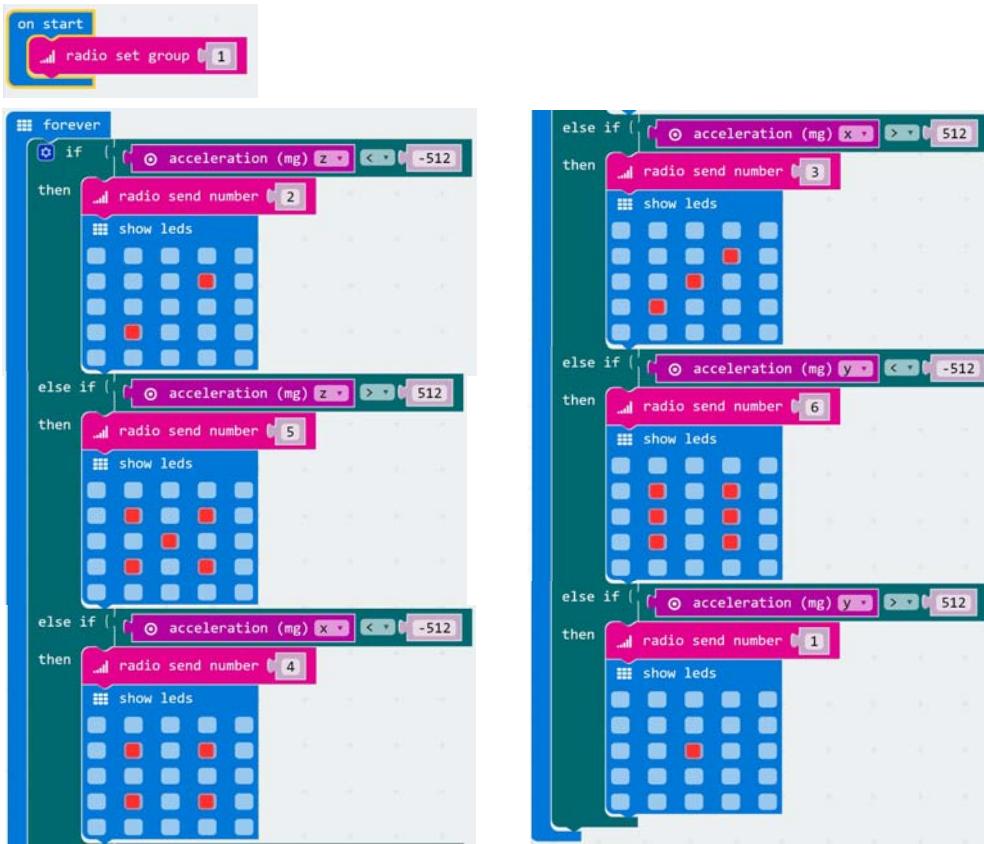
**Left Script:** Triggers when the micro:bit is shaken. It sets a variable `Tilfeldig` to a random value from 0 to 6. Then it uses an `if...then` loop to check the value of `Tilfeldig`. If `Tilfeldig = 2`, it shows a pattern where the top-left LED is red. If `Tilfeldig = 5`, it shows a pattern where the middle-left LED is red. If `Tilfeldig = 4`, it shows a pattern where the bottom-left LED is red. If `Tilfeldig = 3`, it shows a pattern where the middle-right LED is red. If `Tilfeldig = 6`, it shows a pattern where the bottom-right LED is red. If `Tilfeldig = 1`, it shows a pattern where the middle LED is red.

**Right Script:** Triggers when the micro:bit is shaken. It sets a variable `B-flag` to 1. Then it uses an `else if...then` loop to check the value of `terningsverdi`. If `terningsverdi = 2`, it shows a pattern where the top-left LED is red. If `terningsverdi = 5`, it shows a pattern where the middle-left LED is red. If `terningsverdi = 3`, it shows a pattern where the bottom-left LED is red. If `terningsverdi = 6`, it shows a pattern where the bottom-right LED is red. If `terningsverdi = 1`, it shows a pattern where the middle LED is red.



#### 4.4.2 Program Micro:bit 2 (sender)

Figuren under viser det blokkprogrammerte programmet for senderen. Denne detekterer retningen til terningen og sender denne til mottakeren som viser riktig terningverdi.





## 5 Turbidimeter – måling av partikkeltetthet i væsker

I dette måleoppsettet har vi valgt å benytte en Micro:bit og la displayet på Micro:biten vise måleresultatene.

### 5.1 Beskrivelse av prinsippene for et turbidimeter

Et turbidimeter mäter *spredningen* av lyset som sendes inn i væskeren, derfor må lyskilden og lyssensoren plasseres på vegger som står vinkelrett på hverandre. Graden av spredning avhenger av størrelsen og konsentrasjonen av partiklene.

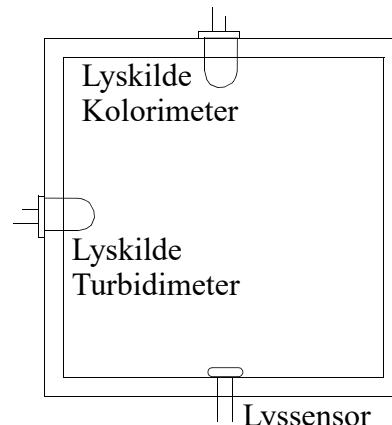
Samtidig kan vi mæle *absorbansen* til væskeblandinga, dvs. hvor mye av det utsendte lyset som absorberes i væskeren. For å mæle absorbans plasseres lyskilden rett overfor sensoren. Dette kan vi kalle et *kolorimeter*.

Både absorbans og spredning vil være avhengig av fargen til lyset som sendes ut. Det er ikke uvanlig å bruke IR-LED som kilde for turbidimetermålinger, og ulike farger tilpasset absorbjonsspekteret til oppløsningen. Her har vi foreløpig valgt å bruke hvitt lys.

For å hindre strølys fra å forstyrre målingene våre, monteres lyskilder og sensor i et lystett kammer, også kalt *kyvetthus*. Kammeret er så stort at det kan romme et plastkrus med plass til 1,5 dL vann. På sikt kan man også tenke seg et mindre oppsett tilpasset *dramsglass* eller lignende.

Turbidimeteret kan også brukes til å mæle innholdet av partikler i drikke- og avløpsvann o.a. Det er også mulig å sammenligne målinger gjort med vårt hjemmelagde instrument med et profesjonelt måleinstrument og om mulig kalibrere turbidimeteret vårt. Vurdering av kvaliteten til instrumentet og ev. forslag til forbedringer vil være et mål med utforskningen.

For å utføre målingene bruker vi Micro:bit som mikrokontroller. Denne styrer lyskildene, leser av sensorverdiene og skriver ut måleverdiene på diode displayet på framsiden av Micro:biten.



Turbidimeterhuset (kyvethuset) kan være prismeformet boks som her eller et papprør

### 5.2 Oppdraget

Oppdraget krever at det bygges et hensiktsmessig kyvettehus for plastkoppen.

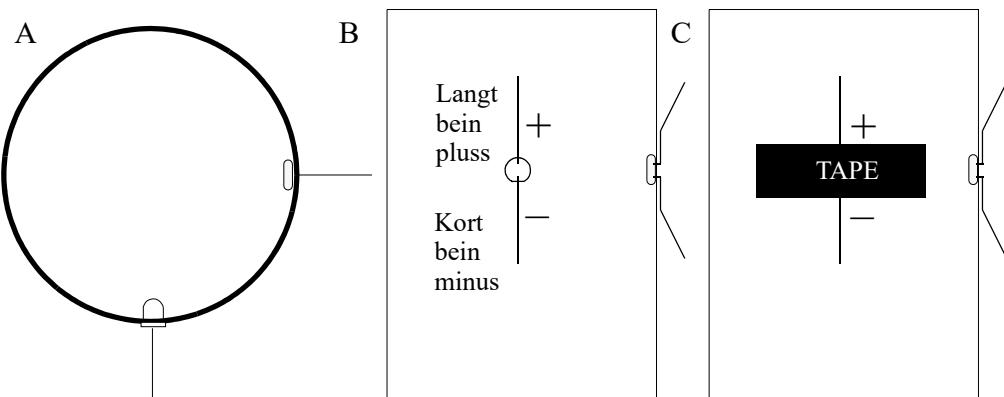
Til dette oppdraget trenger vi følgende komponenter:

- Pappsyylinder med div hull
- Lokk som holder lyset ute

- Krympestrømpe for å gjøre lyssensoren direktiv (mer følsom rett forfra)
- Litt svart elektrikertape
- En eller to lysdioder som avgir hvitt lys
- En lysfølsom motstand, lyssensor (LDR)
- Tilpasset stiftlist
- Fire labledninger med isolerte krokodilleklemmer
- Micro:bit Inventors kit (kantkontakt og koblingsbrett)
- Micro:bit mikrokontrollerkort
- Motstand 220kOhm (rød – rød – sort – oransje)
- Motstand 220Ohm (rød – rød – brun)

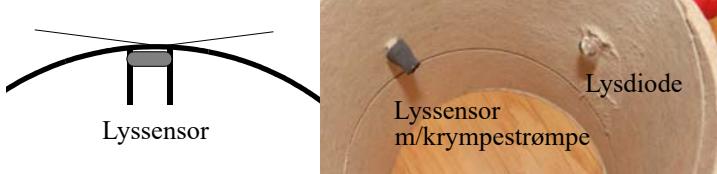
### 5.2.1 Montering av lysdiode og lyssensor i kyvettehuset

1. **Monter lysdioden og den lysfølsomme motstanden** som vist på figuren under. Vær konsekvent ved å **monter plussledningen til lysdioden opp (det lengste beinet)**. Det spiller ingen rolle hvilken vei den lysfølsomme motstanden (lyssensor) monteres. Bøy ledningene slik at de peker henholdsvis oppover og nedover som antydet på figuren under (B). NB! Det er viktig å sette en svart tape på tvers over baksiden av lysdioden og lyssensoren (C) for å hindre lyslekkasje.



2. **Monter krympestrømpe på lyssensoren.**

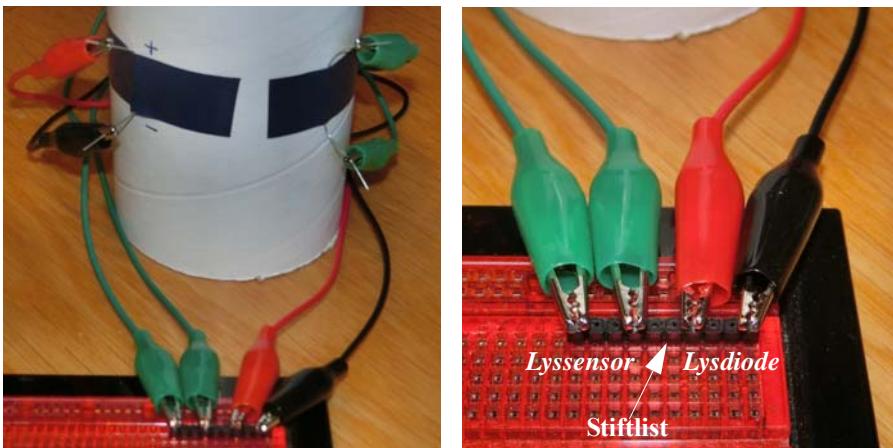
Siden vi kun er interessert i lys som er reflektert fra partiklene, må vi sørge for å hindre at direkteleys fra dioden treffer lyssensoren. Dette gjør vi ved å skyve en ca. 6 mm krympestrømpe inn på lyssensoren før vi monterer den i boksen.





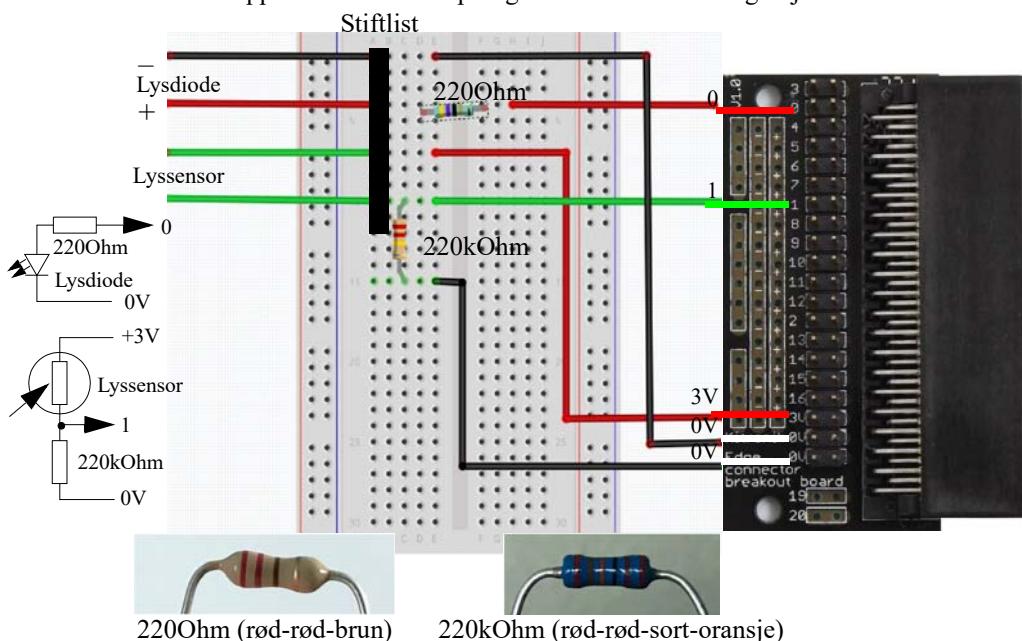
### 3. Monter stiftlisten og krokodilleledningene

Stiftlisten monteres ytterst til høyre på koblingsbrettet som vist på bildet under. Bruk krokodilleledninger til å koble opp forbindelsen mellom kyvettehuset og koblingsbrettet som vist. Om tilgjengelig, bruk rød ledning til + på lysdioden og sort til – på lysdioden. Bruk en annen farge til å koble opp lyssensoren.



#### 5.2.2 Montering av elektronikken

- Vi skal nå koble lysdiode og lyssensor til Micro:biten. Bruk *Micro:bit Inventors kit* pluss tilbehør for å koble opp kretsen som vist på figuren under. Se koblingsskjema nede til venstre.





Vi bruker to motstander. En på 220 Ohm (rød – rød – brun) i serie med lysdioden for å begrense strømmen, og en 220 kOhm (rød – rød – sort – oransje) i serie med lyssensoren for å kunne konvertere endring i motstand til endring i spenning.

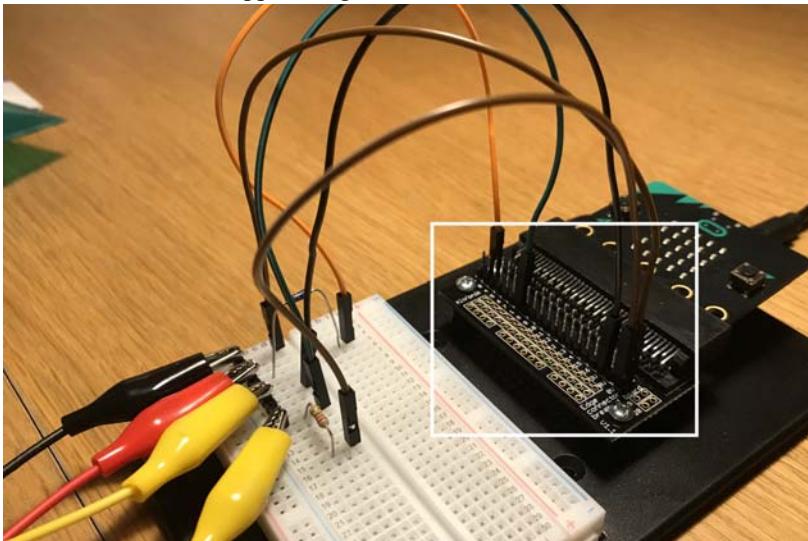


220Ohm (rød-rød-brun)

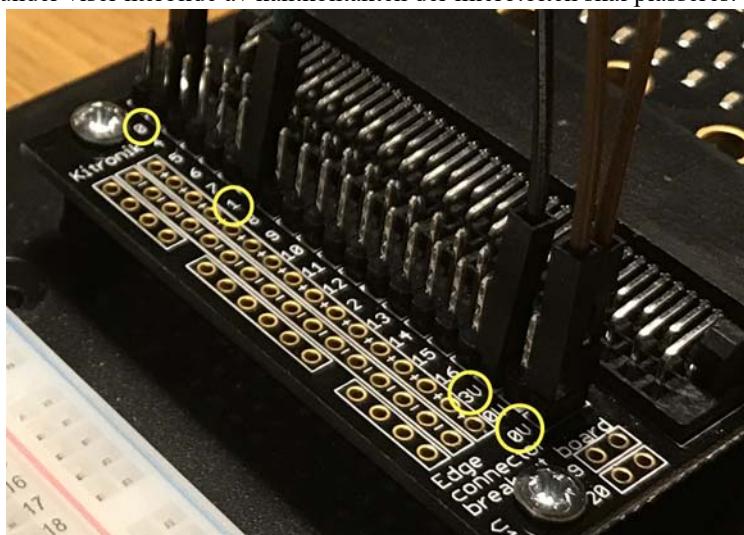


220kOhm (rød-rød-sort-oransje)

Bildet under viser hvordan oppkoblingen kan bli seende ut.



Figuren under viser nærbilde av kantkontakten der micro:biten skal plasseres.



Viser nærbilde av tilkoblingen til stiftlisten på kantkontakten.



Vi er nå klar til å begynne å lage programmet. For den som er usikker på MakeCode-editoren til Micro:bit, se avsnitt 3.3 på side 35.

## 5.3 Lag programmet

La oss bygge opp programmet gradvis.

### 5.3.1 Program for å tenne lysdioden og for å avlese lyssensoren

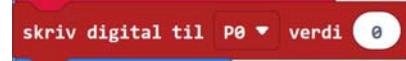
Når vi åpner MakeCode så ligger det to blokker klar til bruk. Dette er blokken *ved start* og *gjenta for alltid*. Det vi kun ønsker skal skje ved oppstart legger vi i *ved start*, og det vi vil skal gjenta seg hele tiden, legger vi i *gjenta for alltid*.



#### 1. Test lysdioden: Tenn og slukk lysdioden

Vi har koblet lysdioden til terminal P0 (pinne 0).

- For å kunne slå av og på spenningen på denne utgangen bruker vi kommandoblokken: *skriv digital til P0 verdi 1*, som vi finner under *Avansert* og menyfanen *Tilkobling*. Vi har valgt port “0” (P0) og verdi logisk “1”, som her betyr at vi setter en spenning på 3V på terminal P0, som nå er en utgang, slik at lyset i lysdioden slås på.
- Tilsvarende kan vi slå av lyset i dioden ved å sette verdien i blokken lik “0”. se figuren til høyre.
- I tillegg ønsker vi at lysdioden skal være påslått, eller ev. avslått en viss tid. Til dette bruker vi kommandoene: *pause (ms)*, som stopper programmet i et ønsket antall millisekunder, her 500 ms.

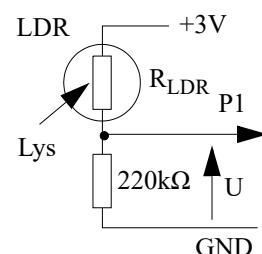


**NB!** Under uttestingen av turbidimeteret har vi funnet ut at lysdioden bruker tid til å slå seg på, dersom tiden fra påslag til måling blir for kort (f.eks. 100 msec) så vil dette påvirke målingen i betydelig grad. Undersøk gjerne dette.

Vurder derfor gjerne å la lysdioden være på hele tiden.

#### 2. Avlesning av den lysintensiteten

Motstandsverdien i den lysfølsomme motstanden (LDR – Light Dependent Resistor) endres med lysstyrken som treffer den. Mye lys gir lav motstand (høy ledningsevne), og lite lys gir høy motstand (lav ledningsevne). Når vi kobler opp LDR'en som vist på figuren til høyre så vil variasjonen i motstandsverdi ( $R_{LDR}$ ) gjøres om til en tilsvarende variasjon i spenning (U)





som vi kan måle på terminal P1. Økt lysstyrke → Redusert motstand ( $R_{LDR}$ ) → Økt spenning (U). Tilsvarende vil Redusert lysstyrke → Økt motstand ( $R_{LDR}$ ) → Redusert spenning (U). Sammenhengen er ikke nødvendigvis lineær.

Når programmet leser av spenning på inngang A0, leser den ikke av spenningen direkte, men et måleverdi mellom 0 – 1023 hvor 0 tilsvarer 0V og 1023 tilsvarer 5 V. Dersom vi ønsker å regne om fra avlest måleverdi til spenning, må vi benytte følgende omregningsformel:

$$\text{Avlest spenning} = \text{Avlest måleverdi} * 5V / 1023$$

Hvor *måleverdien* er det tallet vi leser av.

Strengt tatt trenger vi ikke å finne spenningen, men kun holde oss til måleverdien. Jo høyere måleverdi jo sterkere lys.

- **Definer en variabel som holder den målte spenningen**

Dette kan f.eks. gjøres slik:

Velg fanen *Variabler* og *Lag en variabel* og velge et passende navn til variabelen. Vi velger *Avlest\_lys*. Navnet skal være kort og beskrivende slik at det er lett å vite hva slags verdi variabelen holder.

Når vi har definert variabelen så kommer det opp tre blokker som vist på figuren til høyre. Først selve variabelen *Avlest\_lys*, og så en blokk som gir variabelen en verdi, sett *Avlest\_lys til <0>*, f.eks. verdien 0. Den nederste blokken legger en verdi til den verdien som alt ligger i blokken, *endre Avlest\_lys med <1>*. Denne kan være en variabel eller et hvilket som helst tall.

Vi har nå definert en variabel *Avlest\_lys* og trenger å fylle denne med verdien av spenningen som vi leser av på terminal P1. Denne blokken finner vi under *Avansert* og menyfanen *Tilkobling*

**les analogverdi fra P0**



Siden vi ønsker å lese av terminal P1 og ikke P0, så må vi endre terminalen ved å trykke på den vesle pila til høyre for P0.

Den leste verdien skal nå legges inn i variabelen *Avlest\_lys*. Dette gjør vi med følgende blokk:

Legger verdien inn      Leser av verdi

**sett Avlest\_lys til les analogverdi fra P1**

Det neste vi må gjøre er å skrive ut den avleste verdien til displayet.

### 3. Skriv til displayet



Vi skal nå skrive verdien til displayet på micro:biten. Til dette bruker vi blokken *vis tall <n>*. I stedet for å vise tallet 0, legger vi inn vår variabel *Avlest\_lys*:



Vi har nå alle blokker som skal til for å slå av og på lyset i lysdioden, lese av verdien til LDR'en og skrive ut verdien til displayet.

Skriv programmet og test at det fungerer som ønsket.

Forslag til løsning finnes i vedlegg B.10.1 på side 130.

#### 4. Midling av avlest verdi

Verdiene kan av ulike årsaker variere en del fra måling til måling og kan skyldes tilfeldig støy på den målte spenningen. Vi kan fjerne tilfeldig støy fra målingene ved å *midle måleverdiene*. Dvs. vi tar mange målinger som summeres sammen før vi deler på antall målinger vi har gjennomført, dermed får vi middelverdien.

For å akkumulere avleste verdier bruker vi følgende blokk som vi har sett tidligere: *endre Avlest\_lys med <1>*:



Denne blokken legger verdien 1 til innholdet i variabelen

*Avlest\_lys*. Dersom vi istedet for 1 setter inn en ny måling så vil vi kunne akkumulere mange målinger. For å få til det må vi bruke en *gjenta <n> ganger* blokk. Denne vil sørge for å gjenta det som står i gapet et ønsket antall ganger, f.eks. 100 ganger. Gjenta-blokken finner vi i menyfanen *Løkker*



Ved midling så vil verdien i variabelen kunne bli ganske stor. Skal vi finne middelverdien må vi så dele på antall målinger, f.eks. 100. Dette gjør vi ved å bruke en aritmetikkblokk hentet fra menyfanen *Matematikk*:



Aritmetikkblokk

Verdien delt på 100 legges så tilbake i variabelen *Avlest\_lys* som så kan skrives ut.

**NB!** Til slutt er det viktig at variabelen *Avlest\_lys* nullstilles før vi går i gang med en ny måle-serie for å finne middelverdien.

Løsningsforslaget for turbidimeteret med midling finnes i vedlegg B.10.2 på side 131.

#### 5. Visning av verdien på displayet



Når programmet kjører vil måleverdien rulle over displayet med noen sekunders mellomrom. Normalt vil resultatet oppgis med to desimaler.



#### 6. Lag en variabel som holder antall målinger i midlingen

Under målingene ønsker vi å se betydningen av antallet målinger som ligger til grunn for beregningen av middelverdien. Det er derfor praktisk å definere en variabel som holder denne verdien. Dermed trenger vi kun å endre ett tall for å endre antallet målinger. Vi kaller denne variablene for *Antall*.

Gjør følgende:

- Opprett variablene *Antall* og sett verdien til 100 i blokken *ved start*.
- Erstatt verdiene for antallet som ligger til grunn for midlingen, med variablene *Antall* over alt i programmet hvor denne verdien forekommer.

Løsningsforslag finnes i vedlegg B.10.3 på side 132.

### 5.4 Utfør målinger

Først skal vi gjøre noen målinger uten midling. Vi setter derfor antallet målinger, *Antall* = 1.

#### 1. Testing med tomt og fylt beger

Sett et tomt beger ned i kammeret, sett lokket på og les av verdien på displayet. Du vil sannsynligvis få en verdi i nærheten av **4 – 500**. Fullstendig mørke vil være ca. 0 og fullt opplyst ca. 1000.

Sett så inn et beger fylt med rent vann. Sørg for at vannet og kanten av begeret er fritt for bobler. Siden du sender lyset vinkelrett inn i kammeret i forhold til sensoren, (LDR) måler du nå lys som er spredt i vannet og som treffer sensoren.

Gjør følgende forsøk:

- Les av verdien med tomt beger og med lokket på: \_\_\_\_\_
- Ta av lokket og les av verdien: \_\_\_\_\_
- Ta ut begeret og sett på lokket og les av verdien: \_\_\_\_\_
- Fyll begeret opp til streken med vann og vent noen minutter til vannet har klart seg. Sett på lokket og les av verdien: \_\_\_\_\_



Drøfte resultatene av de fire første målingene: \_\_\_\_\_  
\_\_\_\_\_

- Dreier begeret  $90^\circ$  og  $180^\circ$  rundt, sett på lokket å se om målingene er avhengig av hvordan begeret er orientert:

Hva ble resultatet? \_\_\_\_\_  
\_\_\_\_\_

## 2. Gjør målinger av to vannprøve

Vi skal nå teste måleinstrumentet med to vannprøver:

- Fyll to plastbeger med rent vann. Sørg for at de ikke har bobler som svever i vannet eller at det er bobler langs innsiden av begrene. Bruk gjerne en pipette til å fjerne luftbobler på innsiden.
- Bruk pipetten til å ha i 5 dråper lettmelk i det ene begeret og rør godt rundt
- Senk begeret med det rene vannet ned i kyvettehuset og legg på lokket.  
Les av måleverdien: \_\_\_\_\_
- Senk begeret med 5 dråper melk ned i kyvettehuset og legg på lokket.  
Les av måleverdien: \_\_\_\_\_
- Forskjellen mellom de to målingene er typisk: \_\_\_\_\_

Gjør mange målinger. Legg merke til at gjentatte målinger varierer noe.

Anslå forskjellen i måleverdi mellom de to målingene:  
Variasjonen i målingene på hver av begrene skyldes hovedsakelig støy. Vi ønsker å redusere variasjonen ved å gjøre en midling på 500 målinger (sett *Antall* = 500).

## 3. Gjør målinger på to vannprøver ved hjelp av midling

Vi skal nå teste måleinstrumentet med våre to vannprøver med midling av 500 målinger:

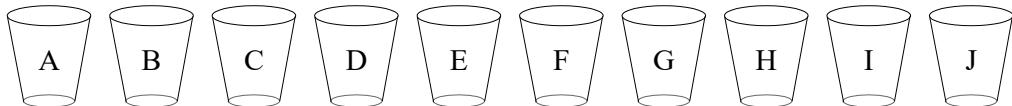
- Bruk de samme vannprøvene som tidligere:
- Senk begeret med det rene vannet ned i kyvettehuset og legg på lokket. La målingene stabilisere seg i ca. 30 sek. og les av måleverdien: \_\_\_\_\_
- Senk begeret med 5 dråper melk ned i kyvettehuset og legg på lokket. La målingene stabilisere seg i ca. 30 sek. og les av måleverdien: \_\_\_\_\_

Ble målingene mer stabile?  
\_\_\_\_\_  
\_\_\_\_\_

## 5.4.1 Gjør målinger på 10 beger og sett dem i “rett” rekkefølge

### Bestem rekkefølgen til begrene

Det er satt frem 10 plastbeger merket A til J. I tillegg til 1,5 dL vann inneholder disse fra 1 til 10 dråper melk.

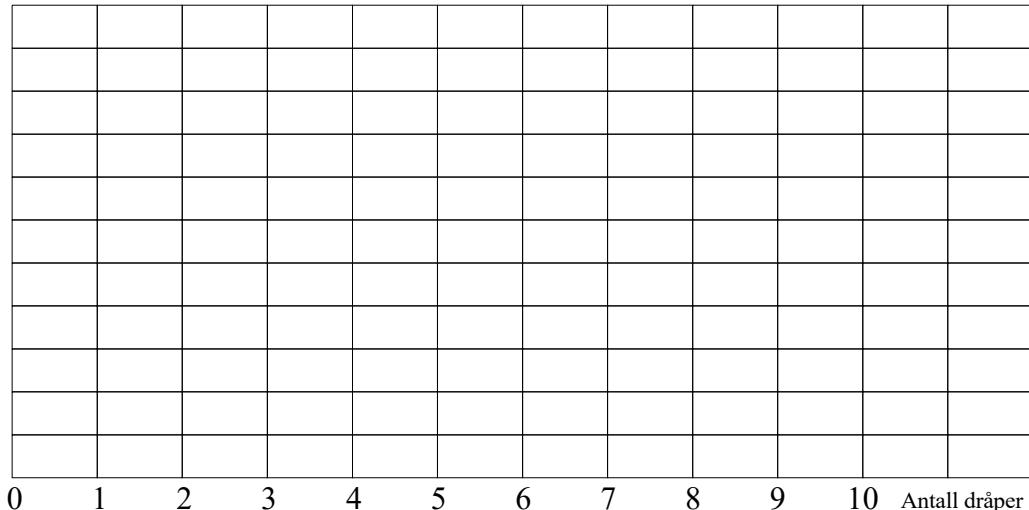


- Bruk instrumentet som *turbidimeter* og bestem rekkefølgen til begrene slik at begeret med én dråpe står lengst til venstre og det med ti står lengst til høyre. Før måleverdiene inn i tabellen:

**Tabell 5.1 Målinger av turbiditet**

| Målinger   | A | B | C | D | E | F | G | H | I | J |
|------------|---|---|---|---|---|---|---|---|---|---|
| Verdi      |   |   |   |   |   |   |   |   |   |   |
| Rekkefølge |   |   |   |   |   |   |   |   |   |   |

Tegn grafene for turbiditet med antall dråper langs x-aksen og de avleste måleverdiene langs y-aksen. Skaler aksene slik at de passer til måleserien.



- Legg inn en best tilpasset kurve og bestem følsomheten til instrumentet, dvs. endring i måleverdi som funksjon av antall dråper melk i oppløsningen for turbidimeteret.

Foreslå mulige forbedringer av måleinstrumentet: \_\_\_\_\_



---

---

---

---

**NB!** Unngå å komme borti lysdioden eller lyssensoren når begrene settes ned i kyvettekameret slik at betingelsene ikke endres fra måling til måling. Det er også viktig at begeret settes ned omtrent på samme måte for hver måling.

#### 5.4.2 Skrive ut antall dråper

I denne oppgaven skal vi bruke *hvis-blokken* for å skrive ut antall dråper i det begeret vi måler på. Dvs. vi må gjøre en omregning slik at måleverdien gjøres om til antall dråper.

##### 1. Finn grenseverdiene for ulike antall dråper melk

Ta utgangspunkt i tabell 5.1 på side 58 og velg grenseverdier for intensitetsmålingene for ulike antall dåper melk.

Tabell 5.2 Grenseverdier for ulike antall dråper melk

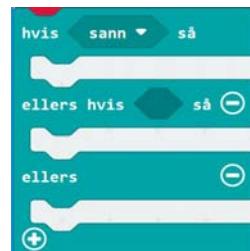
| # dråper      | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |  |
|---------------|---|---|---|---|---|---|---|---|---|---|----|--|
| Grenseverdier |   |   |   |   |   |   |   |   |   |   |    |  |

Avhengig av den målte verdien skal vi nå skrive ut antall dråper i begeret på displayet. Til dette kan vi bruke *hvis ellers – blokken*.

##### 2. Hvis ellers-blokken

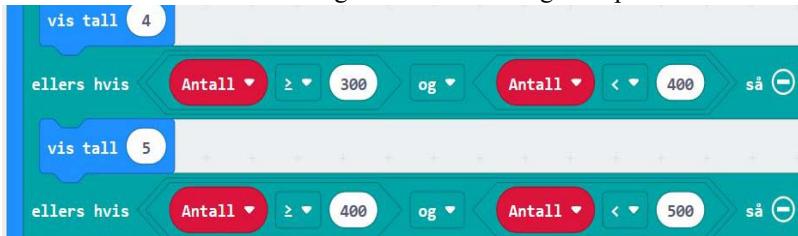
*Hvis ellers-blokken* finnes under menyfanen *Logikk* og den ser ut som vist i figuren til høyre. Dersom betingelsen er sann så utføres kommandoblokken i gapet under. Dersom den er usann går man til neste *ellers hvis* og sjekker om den betingelsen er sann osv. Ved å velge + nederst til venstre i blokken, legges det til flere *ellers hvis*.

For enkelhets skyld har vi i dette eksempelet valgt grenseverdiene 100, 200, 300 osv. Slik vil det ikke være i virkeligheten, grenseverdiene vil sannsynligvis ligge mye tettere. I vår eksempel vil 0 til 100 tilsvare 1 dråpe som

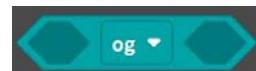
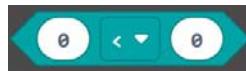




skrives ut til displayet, fra 100 til 200, 2 dråper til displayet osv. Dere må finne de riktige grenseverdiene å sette inn disse i betingelsene for *hvis ellers*-blokkene. Et utsnitt av hvis ellers-blokker kan f.eks. se ut som vist i figuren under for 4 og 5 dråper.



Merk at blokkene for sammenligning og for logiske funksjoner finnes under menyfanen *Logikk*.



### 3. Skriv og test ut programmet

Ta utgangspunkt i programmet du alt har laget og lag tilstrekkelig antall *hvis ellers*-blokker slik at du får ut antall dråper for samtlige beger.

Beskriv hvilke utfordringer dere møtte:

---

---

---

Forslag til løsning på utfordringer:

---

---

---

Løsningsforslag finnes i vedlegg B.10.4 på side 133.

## 6 Fjernstyring av Bit:Bot

I dette kapittelet skal vi 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 wiki-siden til Vitensenteret: [http://wiki.trigger.vitensenteret.com/doku.php?id=introduksjon\\_til\\_bitbot](http://wiki.trigger.vitensenteret.com/doku.php?id=introduksjon_til_bitbot)

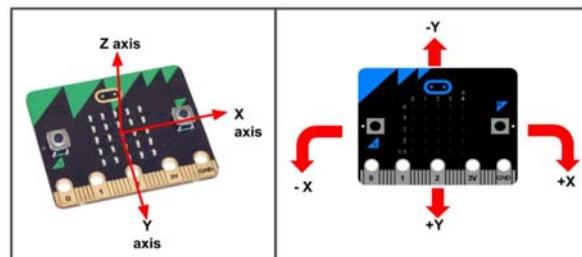
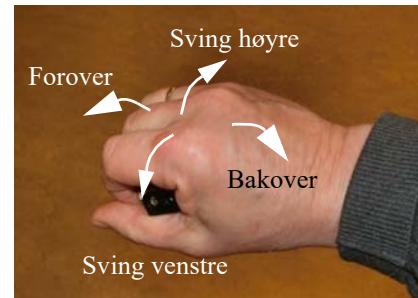


Vi ønsker å fjernstyre roboten ved hjelp av enkle håndbevegelser. For å utføre dette oppdraget trenger vi derfor to micro:bit'er, en batteripakke og en Bit:Bot XL. Den ene micro:bit'en, *sender-enheten*, bruker vi til å sende kommandoer til micro:bit'en som er montert på Bit:Bot'en, *mottaker-enheten*.

Vi ønsker at oppdragets fokus skal være *radio-kommunikasjon* 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 (ned), og tilsvarende øker vi hastigheten bakover ved å bøye hånden bakover (opp). I tillegg ønsker vi å kunne svinge roboten til høyre og venstre, ved å dreie hånden mot henholdsvis høyre og venstre. Dette er mulig når vi vet hvordan akselerometeret i micro:bit'en fungerer.

Figuren til høyre viser akselerometerets akserettninger i forhold til micro:bit-kortet. 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):



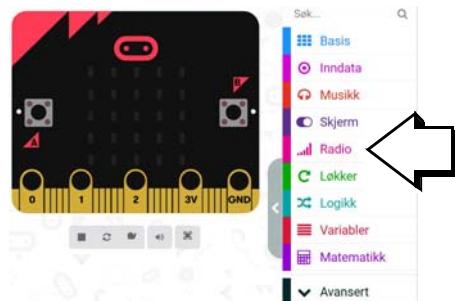
| Handling                                   | x-retning            | y-retning            | z-retning              |
|--|----------------------|----------------------|------------------------|
| Flatt på bordet, display opp               | 0 mg                 | 0 mg                 | - 1000 mg              |
| 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 |

Det er viktig å merke seg at vi får alle mellomliggende verdier i x- og y-retning når vi dreier mikro:bit'en fra horisontal til vertikal orientering.



## 6.1 Grunnleggende programmering av Bit:Bot

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



### 6.1.1 Programmering av sender-enheten

#### 1. Velg radiokanal

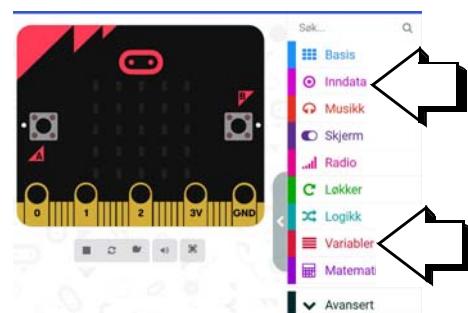
Startblokken hentes fra *Basis-menyen* (blå) og *radio sett gruppe* hentes fra *Radio-menyen* (rød). Velg en radiogruppe som skiller seg fra de andre om det er flere i rommet som gjør det samme.

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.

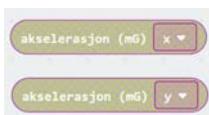


#### 2. Les av akselerometrene

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



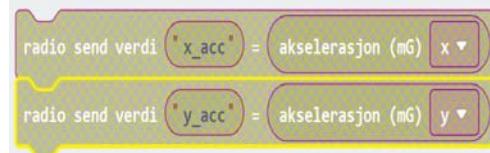
Siden vi skal lese av akselerometeret i både x- og y-retning, må vi gjøre to slike avlesninger, en for hver retning, x og y (se figuren under). Retning velges med den vesle pila til høyre i blokken.



Dernest må vi knytte avlesningene til *identifikatorer* 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 i *radio send verdi*-blokken samtidig som vi legger inn akselerometerverdiene til høyre i blokkene som vist på figuren under.

Da får vi knyttet sammen identifikatorer og avlest verdi.

Senderkommandoen *radio send verdi*-blokken finner vi i *Radio-menyen* (rød):





Blokkene vist i figuren over sender ut identifikatorene og akselerometerverdiene til de andre i gruppen, hos oss bare vår Bit:Bot.

### 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* (blå). 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*

```
gjenta for alltid
  radio send verdi "x_acc" = akselerasjon (mG) x
  pause (ms) 50
  radio send verdi "y_acc" = akselerasjon (mG) y
  pause (ms) 50
end
```

[Last ned](#)

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 kobles til PC'en med en USB-kabel.

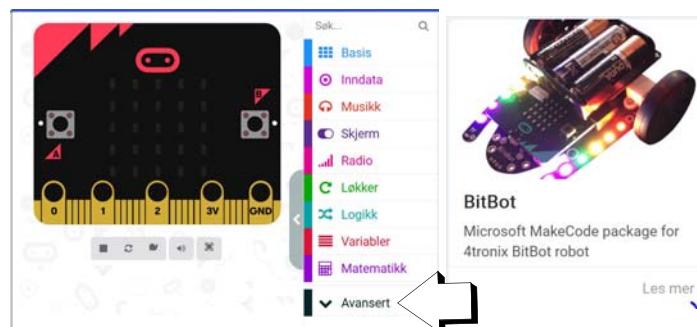
#### 6.1.2 Programmering av mottaker-enheten

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

Pass på at du fjerner programmet som har med sender-enheten og skriver inn navnet på programmet til mottakeren. F.eks.: *Bit-bot Mottaker 1*

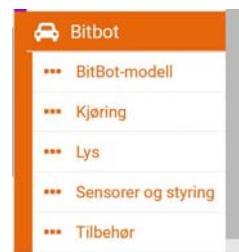
### 4. Installer bibliotek for styring av Bit:Bot

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. Biblioteket installeres 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 rubrikken hvor Bit:Bot er avbildet (figur over til høyre).
- Det finnes to versjoner av Bit:Bot: Classic og XL. Sjekk hva slag Bit:Bot du har og velg riktig robot (Skolelaboratoriet har Bit:Bot XL). Figuren til høyre viser en rekke tilleggsmenyer med kommandoer spesiallaget for Bit:Bot.



## 5. Velg radiokanal og modell

Hent fra funksjonen ved start fra Basis-menyen (blå). De blokkene som legges i funksjonsgapet til denne utføres bare når programmet startes opp.

Her velger vi samme radiokanal (gruppe) som senderen.

Dessuten velger vi *BitBot-modell* fra Bit:Bot-menyen (oransje), og deretter *BitBot modell* ved hjelp av den vesle pila til høyre i blokken, *classic* eller *XL*. Ved å velge *Auto* så vil



Bit:Bot'en selv velge riktig bibliotek, *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 hos micro:bit'en for å styre de to typene robot. Skolelaboratoriets Bit:Bot'er er som sagt 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* (rød) og skriver inn de to variablene i innboksen *Lag ny variabel*.

Ved oppstart velger vi å sette de variablene *X* og *Y* til 0 ved å bruke blokken *sett ... til ...* som vi finner i variabel-menyen. Blokken *ved start* kjører, som tidligere nevnt, bare når programmet starter opp.



## 7. Lytt etter meldinger på radio

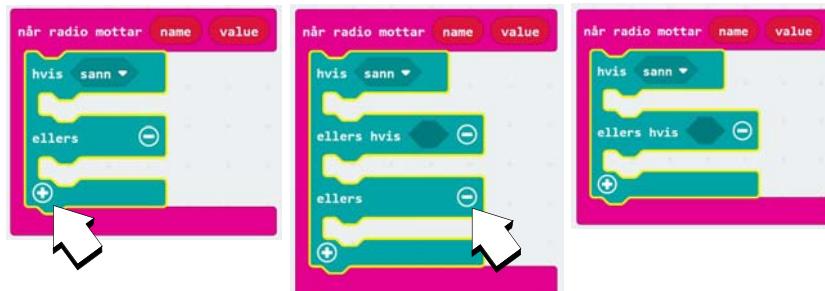
Dernest skal vi lytte etter radiomeldinger som sender på vår radiogruppe (kanal). Til dette bruker vi kommandoeblokken: *når radio mottar "name" "value"* fra Radio-menyen (fiolett). Her kan man enten bruke default navnene "name" og "value", eller man kan definere sine egne navn. Vi velger default verdier.



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 det mottas informasjon, vil *name* og *value* inneholde henholdsvis navnet på den overførte parameteren (*x\_acc* eller *y\_acc*) og verdien (0 – 1023).



Avhengig av om vi mottar  $x\_acc$  (dvs. name er lik “ $x\_acc$ ”) eller  $y\_acc$  (dvs. name er lik “ $y\_acc$ ”) skal vi legge verdien i henholdsvis  $X$  og  $Y$  variablene. For å få til det må vi bruke en *hvis-funksjon* (grønn blokk som vist under).



Hent *hvis-ellers-blokken* fra *Logikk-menyen* (grønn). Ved å trykke på + nederst i venstre hjørne av blokken, får vi 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*.

En *hvis-ellers-blokk* fungerer slik at ulike kommandoer kan utføres avhengig av hvilken betingelse som er oppfylt. Betingelsen setter vi inn i den sekskantede “åpningen”.

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



Så legger vi inn *name* på venstre side av betingelsene og skriver inn henholdsvis  $x\_acc$  og  $y\_acc$  mellom hermetegnene til høyre i betingelsen for å sjekke hvilken variabel som er over sendt. Variabelen *name* kan vi kopiere fra den ytre ramma ved kun å dra den over.



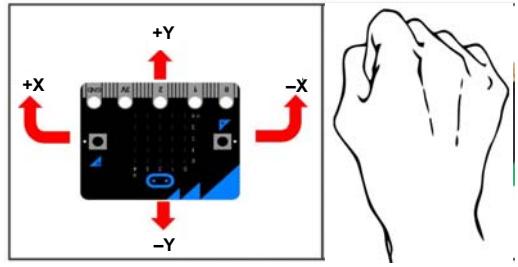
Dersom *name* er lik  $x\_acc$  så skal vi *sette X til value*, dvs. verdien i *value* legges inn i variablen  $X$ . Tilsvarende legges *value* inn i variablen  $Y$  dersom *name* er lik  $y\_acc$ .

Nå har vi verdiene for  $x\_acc$  og  $y\_acc$  liggende i henholdsvis variablene  $X$  og  $Y$ .



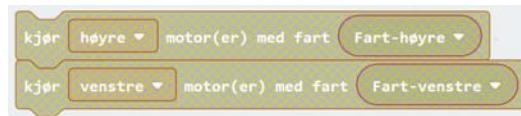
## 8. Styring av motorene

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.



Vi skal nå bruke kommandoer fra Bit:Bot-menyen (oransje) som vi installerte for litt siden.

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 kan vi definere to variabler *Fart-høyre* og *Fart-venstre* som vi gjør i variabel-menyen (rød).

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:

$$\text{Fart-høyre} = Y \quad (6.1)$$

$$\text{Fart-venstre} = Y \quad (6.2)$$

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:

$$\text{Fart-høyre} = Y - X \quad (6.3)$$

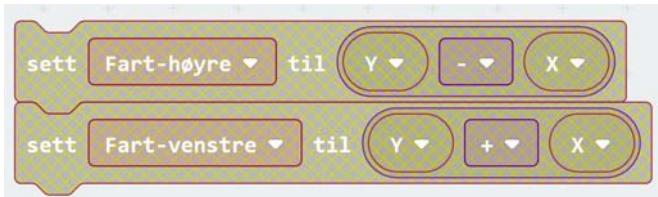
$$\text{Fart-venstre} = Y + X \quad (6.4)$$

Forsøk å resonner dere fram til hvorfor vi har valgt fortgnene slik som vist. Om vi har valgt rett får dere svar på når dere tester programmet i roboten.

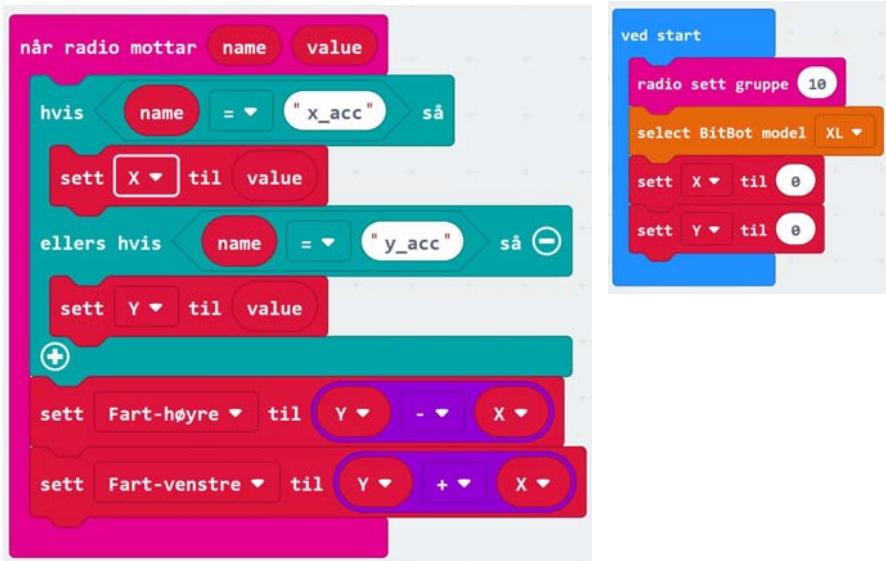
## 9. Beregn farten på hver av motorene



For å beregne farten bruker vi kommando-blokken *sett variabel til*.



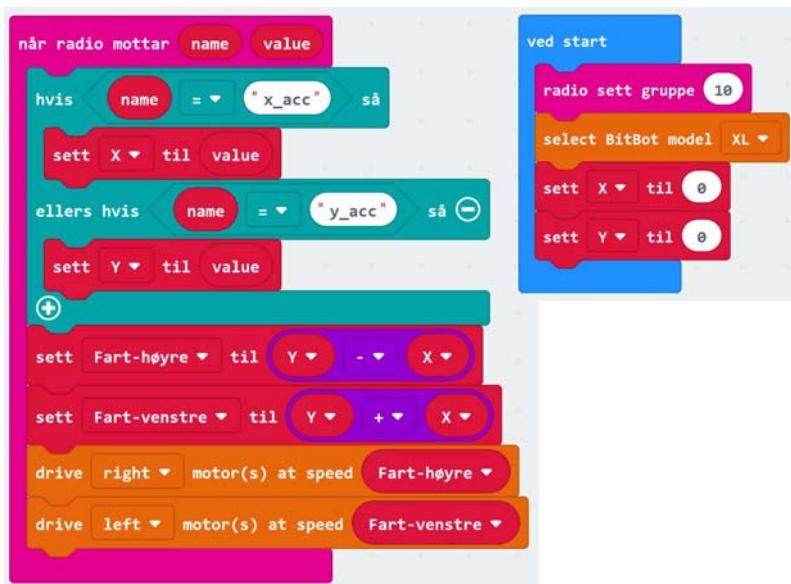
Om vi setter inn disse blokkene etter at verdiene er hentet fra radiomottakeren, vil programmet kunne se slik ut:



Nå har vi beregnet farten til de to motorene som er et tall og tallet er lagt i de to variablene *Fart-høyre* og *Fart-venstre*.

## 10. Sett fart på motorene

For å sette fart på motorene bruker vi to kommandoer fra Bit:Bot-menyen.



Legg merke til at *ved start*-blokken inkluderer variablene *X* og *Y* som ved oppstart er satt til 0, dvs. at vi alltid starter med å stå i ro før den første verdien kommer fra den håndholdte senderen.

## 11. Overføring av programmet til micro:bit

Dernest kan programmet flyttes over til micro:bit'en<sup>24</sup>.

## 12. Forenkling

Ser du en måte som gjør at programmet kan forenkles med færre blokker?

Inntil videre hopper vi over punkt 8 og 9. Uttestinger viser at denne ekstra finessen ikke er nødvendig. Dere kan ev. legge den inn senere.

## 6.2 Tilleggsoppgaver

Dette avsnittet viser programmering av noen ekstra egenskaper ved Bit:Bot'en

---

24.NB! Pass på at Bit:Bot løftes fra bordet når den slås på. Dersom den står på bordet vil lys-sensorene (reflektanssensorene) på undersiden av roboten vanligvis registrere mørke og gå inn i parringsmodus for bluetooth, hvilket vi ikke ønsker i denne omgangen. Det er mulig at dette kun er nødvendig for Classic versjonen av roboten, men greit å være klar over.



## 1. Endre på følsomheten på styrefunksjonen

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.

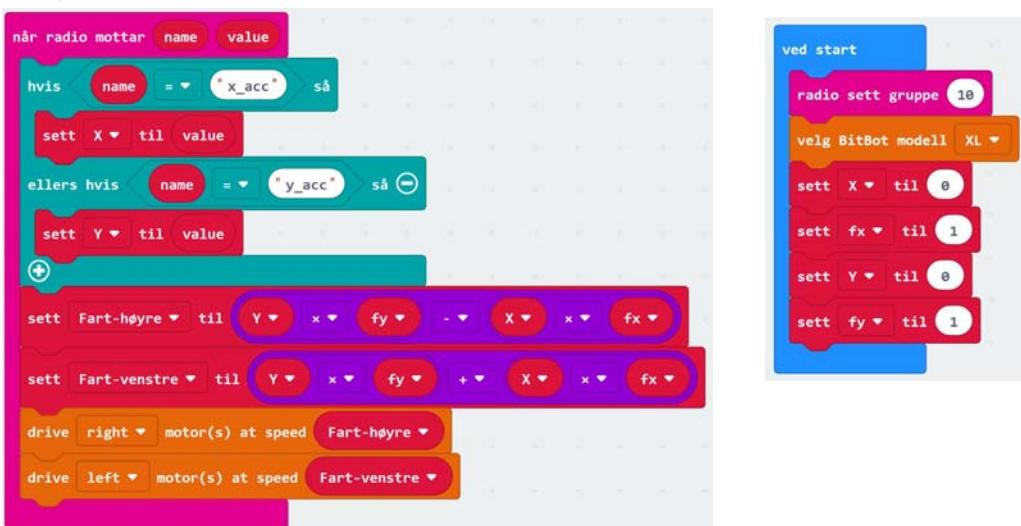
Økt følsomhet betyr at håndbevegelsen får større konsekvenser for farten, mens redusert følsomhet betyr at håndbevegelsen for mindre konsekvenser for farten.

$$\text{Fart-høyre} = Y*fy - X*fx \quad (6.5)$$

$$\text{Fart-venstre} = Y*fy + X*fx \quad (6.6)$$

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.



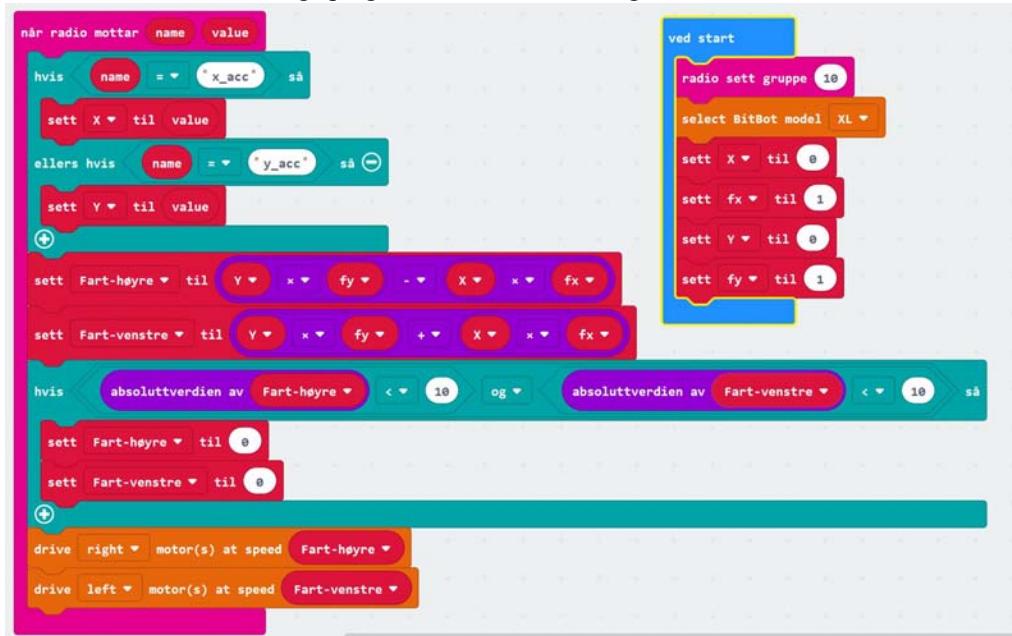
## 2. Sett opp et stoppvindu

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* (grønn).



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



### 3. Overføring av programmet til micro:bit på BitBot

Dernest kan programmet flyttes over til micro:bit'en<sup>25</sup>.

#### 6.2.1 Lys og lyd hos Bit:Bot

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.

25.NB! Pass på at Bit:Bot løftes fra bordet når den slås på. Dersom den står på bordet vil lys-sensorene (reflektanssensorene) på undersiden av roboten vanligvis registrere mørke og gå inn i parringsmodus for bluetooth, hvilket vi ikke ønsker i denne omgangen. Det er mulig at dette kun er nødvendig for Classic versjonen av roboten, men greit å være klar over.



## 7 Barometrisk høydemåler

Basert på en lufttrykksmåler (BME280) og et forbedret display skal vi lage en barometrisk høydemåler, dvs. en høydemåler som er basert på det faktum at lufttrykket faller med økende høyde over havet. Sammenhengen mellom lufttrykk og høyden over havet kan modelleres ganske godt. Skal man måle absolutt høyde må man dessuten passe på å kalibrere høydemåleren siden den er sterkt avhengig av Høy-trykk og Lav-trykk.

### 7.1 Modellering av sammenheng mellom lufttrykk og høyde over havet

I nedre del av atmosfæren er det ikke uvanlig å bruke ligning (7.1), som gir sammenhengen mellom lufttrykk og høyde gitt en referansehøyde ( $h_1$ ), et referansestrykk ( $p_1$ ) og en referansetemperatur ( $T_1$ ):

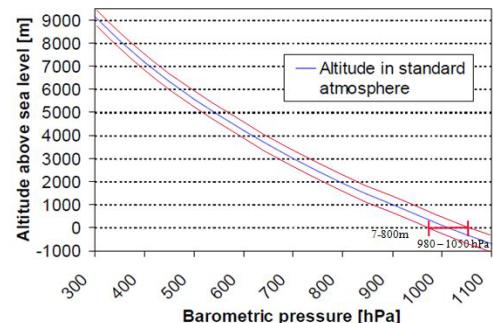
$$h = \frac{T_1}{a} \left( \left( \frac{p}{p_1} \right)^{-\frac{aR}{g_0}} - 1 \right) + h_1 \quad (7.1)$$

Hvor:

- $h$  Beregnet høyde i meter  
 $h_1$  Starthøyde i meter  
 $T$  Temperatur i Kelvin  
 $T_1$  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<sup>2</sup>  
 $R$  Den spesifikke gasskonstant 287,06 J/kg K

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 absolutt endring i høydeberegningen på 7 – 800 meter dersom man ikke kalibrerer målingene.

Vi skjønner derfor at det er svært viktig med hyppig kalibrering dersom man skal kunne stole på høydemålerens absolutte høydeverdier. Dette fikk Widerøs piloter føle på kroppen 10. februar 2020, da lufttrykket falt så lavt (ca. 920 mbar ved havnivå) at de barometriske høydemålerne i Widerøeflyene ikke kunne kalibreres og måtte settes på bakken til lufttrykket hadde steget.





## 7.2 Måling av lufttrykk

Det finnes mange gode sensorer som mäter lufttrykk. I denne sammenhengen velger vi BME280 siden denne i tillegg til trykkmålinger også mäter temperatur og luftfuktighet.

BME280 er en utvidelse av BMP280 hvor man har inkludert fuktighetsmåling. 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<sup>26</sup>:

- *Spennin og energiforbruk:*  
Supplyspenning: 1,7 – 3,6 V  
Strømforbruk standby: 0,1  $\mu$ A  
Strømforbruk med måling 1 x pr. sek.: 3,6  $\mu$ A (H, P, T)
- *Grensesnitt:*  
 $I^2C$  eller SPI
- *Luftfuktighetssensor:*  
Responstid: 1 sek.  
Nøyaktighet:  $\pm 3\%$  mellom 20 – 80% relativ fuktighet.
- *Lufttrykksensor:*  
Måleområde trykksensor: 300 – 1100 hPa  
Relativ nøyaktighet:  $\pm 0,12\text{hPa}$   
Absolutt nøyaktighet:  $\pm 1\text{hPa}$  (0 – 65°C)
- *Temperatursensor:*  
Måleområde: -40 – +65°C  
Nøyaktighet:  $\pm 1^\circ\text{C}$  (0 – 65°C)

### Oppkobling

Figuren under viser pinningen til to utgaver til BME280:

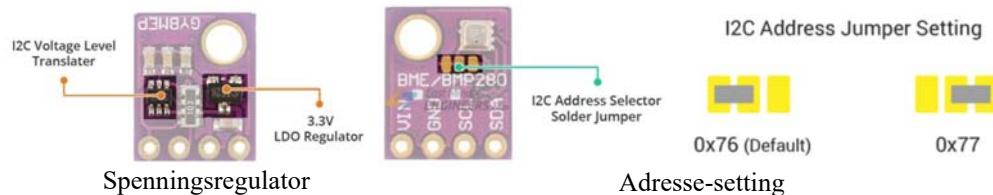


En viktig forskjell er at hos varianten til høyre kan man sette to ulike adresser i “adressefellet”. Default adresse er 0x76 HEX, men den kan endres til 0x77 HEX om man forbinder to pad’er som vist helt til høyre på figuren under. Dessuten har denne varianten en innebygget spenningsregula-

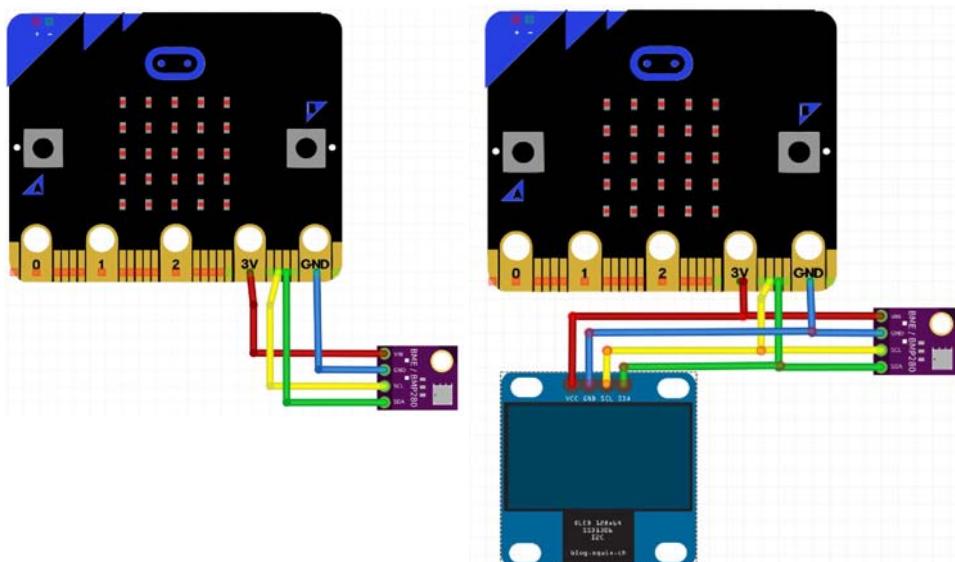
26. Informasjonen er hentet fra: <https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bme280-ds002.pdf>



tor slik at den kan arbeide på spenninger fra 3,3 – 5,0 V.



Siden BME280 har en arbeidsspenning på 3,3V og kommuniserer med micro:bit'en over I<sup>2</sup>C-bussen, så kan vi koble oss direkte til kantkontakten, riktig nok via en connector som ikke er vist på figuren under.



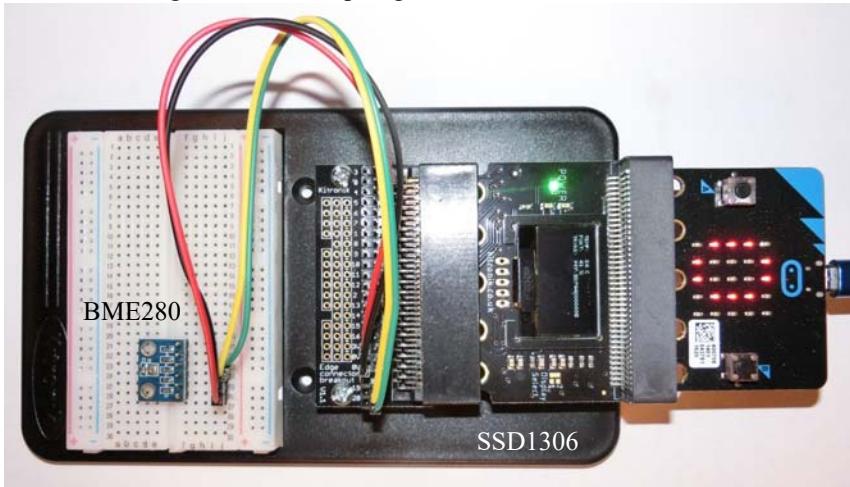
### 7.3 Display

Siden vi ønsker å vise temperatur, luftfuktighet, lufttrykk og høyde, er LED-displayet på mikrobit'en upraktisk og vi bruker et OLED-display med 128 x 64 pixler. Vi har valgt å bruk et nyutviklet tilleggskort hvor det nevnte displayet er montert, men det går også an å benytte et enkelt OLED-display som også har en arbeids-spenning på 3,3 – 5,0 V og kommuniserer vi en I<sup>2</sup>C-buss.





Display-kortet er utviklet av Kitronik og kan kobles direkte på kantkontakten til micro:bit'en. En ny kantkontakt finnes på displaykortet slik det kan kobles til andre applikasjoner, som f.eks. en kantkontakt med koblingsbrett som vist på figuren under.



Eneste ulempen er at teksten på displayet er særdeles liten, dette ser det ikke ut til at man kan gjøre noe med i denne versjonen. Til gjengjeld får man plassert 8 linjer under hverandre på displayet.

## 7.4 Biblioteker

Det er utviklet biblioteker, både for displayet og for BME280. Disse installeres ved å gå velge *Utvidelser* i menyen og søke etter *Display* og *BME280* i bibliotekskatalogen.

**kitronik-128x64Display**  
MakeCode extension for VIEW  
128x64 Display

[Les mer](#)

**BME280**  
humidity and pressure sensor pxt  
package

[Les mer](#)

**OLED**

**BME280**



Ved å velge og installere disse to har man alt man trenger for å bruke de to komponentene.

## 7.5 Beregning av eksponenter med micro:bit

Vi legger merke til at dersom vi skal regne om fra trykk til høyde så inngår en eksponentfunksjon. I utgangspunktet burde ikke dette være noe problem da vi har følgende kommando tilgjengelig i matematikkmenyen.



Man skulle derfor tro at denne kunne brukes i vårt tilfelle for å beregne omregningsformelen:

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

Imidlertid er det ikke så enkelt. Kommandoen klarer greit å regne med konstanter, som antydet på figuren under:



Men så snart en eller begge de to konstantene byttes ut med en variabel så avrundes verdien til nærmeste hele tall før beregningen gjøres. I vårt tilfelle betyr det at resultatet blir meningsløst. Så hva gjør man i dette tilfellet?

Det viser seg at matematikken har et trumfkort i ermet. I 1715 klare den engelske matematikeren **Brook Taylor** (1685 – 1731) å vise at nær et punkt på en funksjon så kan funksjonen uttrykkes som en sum av ledd bestående av stadig høyere deriverte av funksjonen. Anvender man denne metoden så kan man f.eks. finne at:

$$(1+x)^\alpha = 1 + \alpha x + \frac{1}{2}\alpha(\alpha-1)x^2 + \frac{1}{6}\alpha(\alpha-1)(\alpha-2)x^3 + \frac{1}{24}\alpha(\alpha-1)(\alpha-2)(\alpha-3)x^4 + \dots$$

En formel som ser temmelig “hårete” ut og kan skremme vannet av noen og enhver.

Vi ser at  $(1+x)^\alpha = \dots$  er en eksponentialfunksjon uttrykt som en lang rekke ledd bestående av  $x$  og  $\alpha$ , og summasjoner, subtraksjoner og multiplikasjoner ( $x^2, x^3 \dots$  kan betraktes som multiplikasjoner). Som alle er operasjoner som micro:bit kan håndtere uten avkorting, selv om de er uttrykt med variabler. Dermed burde det være mulig å erstatte eksponentialfunksjonen med rekken. Jo flere ledd vi tar med, jo mer nøyaktig blir svaret. Så spørsmålet er hvor mange ledd som trengs.

Dersom  $|x| < 1$  og  $\alpha x \ll 1$  så blir leddene raskt svært små og vi kan nøye oss med få ledd, kanskje bare de tre første. Siden grunntallet  $p/p_1$  alltid har en verdi nær 1 så vil dette være tilfelle.

$$(1+x)^\alpha = 1 + \alpha x + 1/2\alpha(\alpha-1)x^2 \quad (7.2)$$



Vår interesse i denne formelen er å få beregnet resultatet av et grunntall  $(1 + x)$  opphøyd i en eksponent. Grunntallet er uttrykt som  $(1 + x)$ . Dersom grunntallet  $g = (1 + x)$  så vil  $x = g - 1$ . Vi velger da å skrive formelen:

$$(g)^\alpha = 1 + \alpha \cdot x + 1/2\alpha(\alpha - 1) x \cdot x \quad (7.3)$$

$$\text{der } x = g - 1 \quad (7.4)$$

I vårt tilfelle er:

$$g = \frac{p}{p_1} \text{ og } \alpha = \text{alfa} = -\frac{aR}{g_0} \quad (7.5)$$

I vår blokkode har vi laget oss en funksjon, *apow(x,n)*, som utfører eksponentialberegningen. Denne funksjonen kalles med de to parametriene *x* (grunntallet) og *n* (eksponenten), hvor *x* og *n* er betegnelsene som brukes inne i funksjonen. Funksjonen gjør følgende beregning:

$$d = x - 1 \quad (7.6)$$

$$x^n = 1 + n \cdot d + (n \cdot (n - 1)) d \cdot d / 2 \text{ som returneres fra funksjonsberegningen.} \quad (7.7)$$

Vi lager en funksjon som utfører dette regnestykket.

## 7.6 Programmet skrevet i blokkode

### 7.6.1 Definisjon av variabler

Det første vi gjør er å deklarere å gi verdier til variablene vi trenger for beregningene. Her setter vi også referansehøyde (*H1*), referansetryk (*P1*) og referansetemperatur (*T1*) på den målestasjonen vi ønsker å bruke til å kalibrere instrumentet. Dette gjør vi i *ved start* blokka.

Legg også merke til at vi setter adressen til BME280, som er lik 0x76 (Hex).

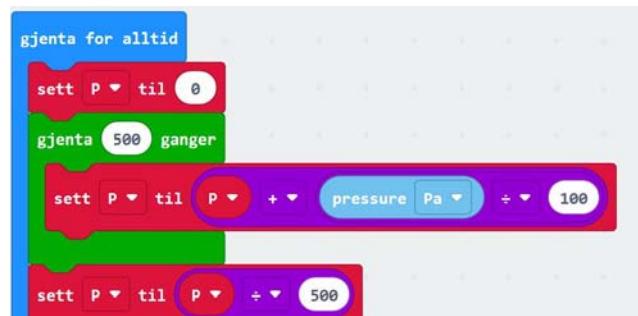




## 7.6.2 Innhenting av trykk fra BME280

Vi har lastet inn biblioteket for BME280 og får tilgang til lufttrykk, temperatur og luftfuktighet.

Programmet leser først inn målinger av lufttrykket ( $p$ ) og midler verdien over 500 målinger for å få stabile verdier.



## 7.6.3 Beregning av høyden

Når lufttrykket er målt og referansetrykk og høyde er satt, så kan vi beregne høyden over havet med den angitte formelen. Vi ser at vi som forventet, har behov for å beregne en eksponentifunksjon. Dette gjør vi med den spesiallagde funksjonen *apow* ( $g$ ,  $alfa$ ), hvor  $g$  er grunnallet og  $alfa$  er eksponenten..



## 7.6.4 Beregning av $x^n$

For å kunne skrive programmet i blokkode, deler vi opp operasjonene i blokker av enkle regneoperasjoner. For oversiktens skyld har vi laget variabler  $q_1$  til  $q_8$  som holder resultatene fra de enkelte regneoperasjonene. Dette er forsøkt vist i figuren til høyre.

$$x^n = 1 + n \cdot (x-1) + (n \cdot (n-1)) (x-1) \cdot (x-1) / 2$$

$$d = x - 1 \text{ og får: } \rightarrow$$

$$x^n = 1 + \underbrace{n \cdot d}_{q_1} + (n \cdot (n-1) d \cdot d) / 2 \rightarrow$$

$$q_1 = n \cdot d \rightarrow$$

$$q_2 = \overbrace{n - 1}^{} \rightarrow$$

$$q_3 = \overbrace{d \cdot d}^{} \rightarrow$$

$$q_4 = \overbrace{q_2 \cdot n}^{} \rightarrow$$

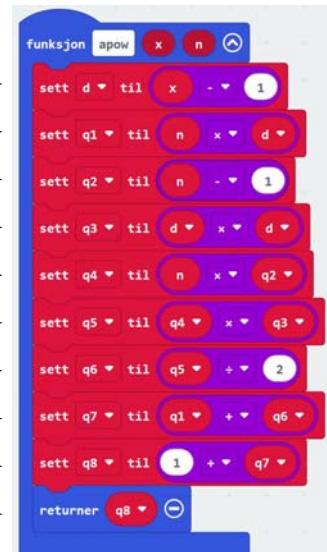
$$q_5 = \overbrace{q_4 \cdot q_3}^{} \rightarrow$$

$$q_6 = \overbrace{q_5 / 2}^{} \rightarrow$$

$$q_7 = \overbrace{q_1 + q_6}^{} \rightarrow$$

$$q_8 = \overbrace{1 + q_7}^{} \rightarrow$$

$$x^n = q_8 \rightarrow$$

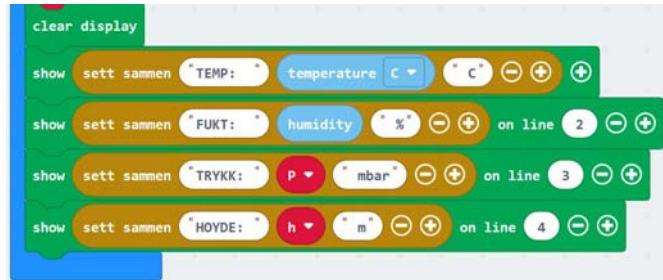




De ulike regneoperasjonene realiseres som enkle matematiske blokker som vist til høyde på figuren.  $q_8$  returneres fra funksjonen og er svaret på beregningen.

### 7.6.5 Utskrift

Deretter leser vi inn temperatur og luftfuktighet som vi skriver ut sammen med lufttrykk ( $p$ ) og den beregnede høyden ( $h$ ). Vi velger å sette sammen teksten, variabel-verdien og benevnningen til en tekststreng som vi sender til displayet.



Dermed er høydemåleren ferdig, som riktig nok må kalibreres jevnlig og helst flere ganger om dagen skal man beholde den absolutte nøyaktigheten.



## 8 Beskrivelse av en selvlaget robot

I dette kapittelet skal vi realisere en robot med Micro:bit. Dette er ikke ment som et endelig design, men grunnlag for videre utvikling. Vi har derfor tatt med et innledende avsnitt som diskuterer de ulike løsningene og begrunner de valg vi har gjort. I tillegg vil vi her peke på muligheter som ennå ikke er realisert i vår prototyp.

Roboten skal være enkel og det skal brukes billige deler som skoler også har råd til å anskaffe.

### 8.1 Innledende betrakninger om valg av løsning

Innledningsvis skal vi diskutere ulike løsninger og gi korte begrunnelser for de valg vi har gjort uten at dette skal være noen fasit.

#### 8.1.1 Chassie

Selve ramma til roboten bør være så enkel som mulig og kunne laserkuttet i MDF, finér eller akryl.

På oversiden bør den ha plass til:

- To små servomotorer med hjul, enten innkjøpte eller laserkuttede hjul
- Kantkontakt for Micro:bit
- Et lite koblingsbrett (for oppkobling av servomotorer, ev. ultralyd avstandsmåler, lysdioder)
- Plass til avstandssensor og lysdioder

På undersiden bør den ha plass til:

- To batteripakker for to AA batterier, min. 6V
- Et nesehjul eller støtte, ev. et kulehjul, må kunne monteres foran
- Mulighet for montering av to reflektanssensorer foran for å kunne følge en sort linje

Styring og kontroll:

- Roboten bør kunne fjernstyres med en Micro:bit

#### 8.1.2 Sensorer og aktuatorer er aktuelle for robot

Følgende aktuatorer er tilgjengelig ev. må være mulig å montere:

- To 360° servomotorer, hvor fart og retning kan styres
- Et display med 5 x 5 lysdioder (på Micro:bit)
- Ev. eksterne lysdioder
- En eller to 180° servoer påmontert armer

Følgende sensorer er tilgjengelig internt på Micro:bit:



- 
- Akselerometer, for styring fra håndholdt Micro:bit
  - Magnetometer (anvendelse ikke bestemt)
  - Radiokommunikasjon, kanal for fjernstyring

Følgende eksterne sensorer kan være aktuelle på roboten:

- Kollisjonssensor
- Reflektanssensorer
- Avstandssensor

## 8.2 Forslag til undervisningsopplegg

### 8.2.1 Hensikt:

Hensikten med undervisningsopplegget er å:

- ... tilby et litt større prosjekt knyttet til programmering av Micro:bit
- ... lage en robot med Micro:bit som kan fjernstyres gjennom en løype på kortest mulig tid.

De skal lære:

- ... hvordan en enkel robot fungerer
- ... hvordan de enkelte delene av en robot fungerer
- ... grunnleggende blokkprogrammering
- ... hvordan programmere servoer
- ... hvordan programmere fjernstyring med sving, forover og bakover med ulike hastighet
- ... å beskrive hvilke utfordringer byggingen, programmeringen og bruken av roboten ga
- ... å dokumentere hva de har gjort slik at andre kan bygge på det de har gjort

### 8.2.2 Enkelt eller avansert oppdrag

En kan tenke seg en enkel løsning eller en krevende løsning:

#### Enkelt oppdrag:

Sett sammen roboten med delene til bygesettet.

Installere programvare og sjekke at roboten fungerer som ønsket

#### Krevende oppdrag

Konstruer en robot med bestemte egenskaper

Skriv kode slik at roboten er i stand til å utføre et bestemt oppdrag

*Det krevende oppdraget* kan deles opp i mindre oppgaver med økende vanskelighetsgrad:

#### **Hovedoppdrag:**

Det skal konstrueres en robot som har følgende egenskaper:



- 
- Konstruer en robot som styres av ett Micro:bit kort. Roboten skal kunne gå framover og bakover og svinge til høyre og venstre
  - Robotens bevegelser skal kunne fjernstyres av et Micro:bit kortet
  - Roboten skal utstyres med en mekanisme som er i stand til å ødelegg en ballong montert på en annen robot. Mekanismen skal kunne fjernstyres.

Eller ev.

- Roboten skal kunne tegne en strek mens den går. Senking og heving av pennen skal kunne fjernstyres.

Hovedoppdraget (avansert) kan deles opp i mange påfølgende oppdrag som hjelper deltagerne til å nå det endelige målet:

1. Finn ut hvordan en kobler sammen en Micro:bit og en 360° servo og lag et blokkdiagram
2. Koble en Micro:bit til en 360° servo og batterier (2x2xAA) ved hjelp av en kantkontakt
3. Skriv en så enkel kode som mulig for å sjekke at servoen kan gå framover og bakover, ev. at det er mulig å styre hastigheten
4. Konstruer roboten (chassiet) med to servoer med drivhjul og nesehjul, Micro:bit kort i kantkontakt og nødvendig batterikasserter.
5. Skriv en enkel kode for Micro:bit og sjekk at roboten kan gå rett fram og rett bakover. Finn ut hva som må gjøres for å kalibrere den slik at den ikke går skjevt.
6. Skriv ny kode slik at du kan bruke en Micro:bit nr. to som fjernstyringsenhet for den som er montert på roboten. Med fjernstyringsenheten skal du kunne starte, stoppe og kjøre forover og bakover ved hjelp av de to knappene på fjernstyrings Micro:biten
7. Skriv ny kode slik at du ved å helle fjernstyrings Micro:biten til høyre eller til venstre klarer å svinge roboten mot høyre og venstre. Knappene skal fortsatt kunne brukes til å kjøre fram og tilbake og til å stoppe.
8. Skriv ny kode slik at du frigjør knappene og blir istrand å kjøre framover, bakover, svinge til høyre og venstre ved å bikkje fjernstyrings Micro:biten framover, bakover, bikkje til høyre og venstre.
9. Skriv ny kode slik at du blir istrand til å bruke knappene til å bevege armer med nåler framover og til sidene eller styre en penn.
10. Test ut og dokumenter det du har gjort

### 8.2.3 Anvendelse

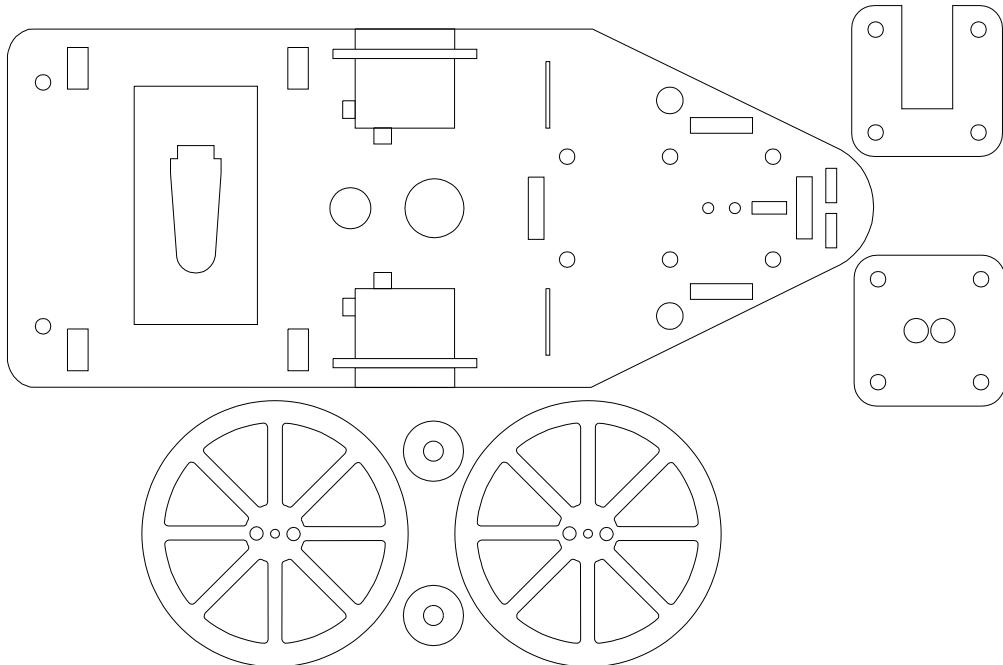
Deltagerne konkurrerer om å komme gjennom en løype med hinder på kortest mulig tid uten og berøre hindringene.

Eller de skal kjempe om å slå istykker ballonger på konkurrentenes roboter. En robot med ødelagt ballong er ute av konkurransen.

## 8.3 Betraktninger om realisering (“Tinkering”)

### 8.3.1 Chassie

Figuren under viser omrisset av selve chassiet som kan skjæres ut av en laserkutter.



De to kvadratlignende klossene til høyre er ment som holder for en servo foran over nesehjulet eller som holder for en servo som kan løfte og senke en penn. Det vil framgå hvordan dette er tenkt i neste bildene.

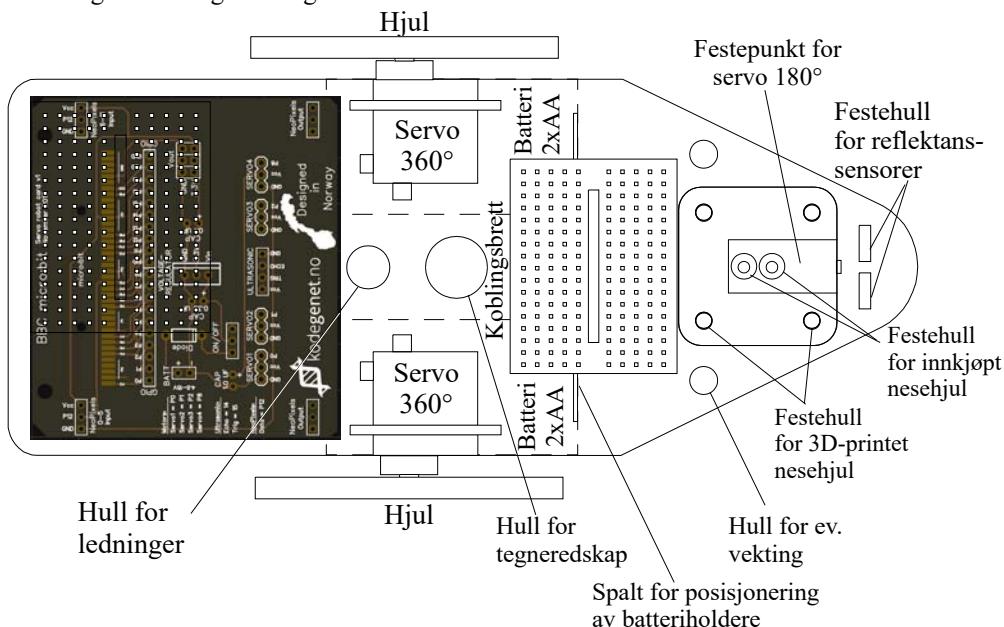
Chassiet kan skjæres ut i forskjellig typer materiale. Det kan være 3 eller 6 mm finér, MDF eller også akryl.

Bildet til høyre viser det utskårne chassiet i 3 mm sort MDF.





Bildet under viser en tegning av hvordan roboten kan monteres. Denne varianten er forberedt for montering av koblingsbrett og servo i fronten.



De to batteripakkene er montert på undersiden av chassiset, på innsiden av drivhjulene på undersiden av servoene. Batteriledningene kommer opp gjennom et hull i plata ved kretskortet med kantkontakt og kobles til Micro:bit og servomotorene via kontakter på kretskortet. Alternativt kan ledningene fra batteriet loddet direkte til kretskortet på undersiden.

Figuren til høyre viser et bilde av den monterte prototypen.

Batterikontakten er loddet direkte til batterikontakten på undersiden. Batteriholderne er montert på undersiden under hver av servoene.



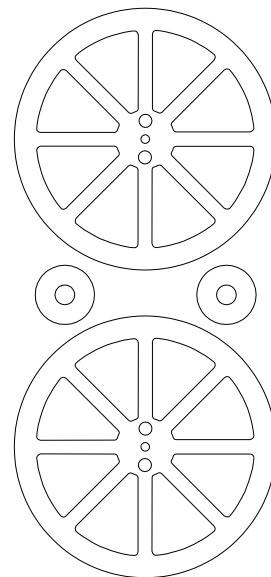
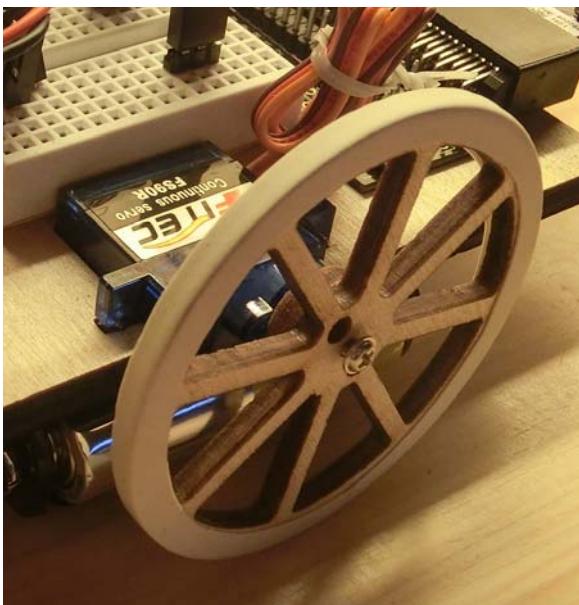
### 8.3.2 Hjul

Denne modellen har standard drivhjul som vist under.



Hjulene festes med en skrue i navet.

Vi har også prøvd å skrive ut hjul, hvilket er fult mulig. Vår erfaring er imidlertid at de lett blir noe vinglete. Bildet under viser et hjemmelaget hjul med ett hvitt bånd av ballonggummi tredt over felgen for å gi økt friksjon mot underlaget. Til høyre er vist en forminsket utgave av skjæremalen til hjulene. De to små hjulene monteres innerst ved navet og passer til akslingen på servomotoren.

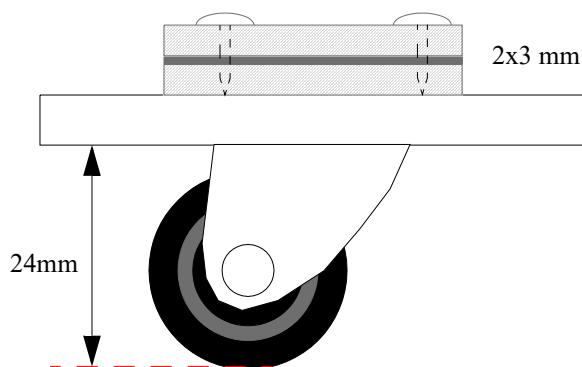




Vi har forsøkt å benytte et lite leddhjul fra Clas Ohlson, hvilket synes å fungere.



Leddhjulet (25/36 mm) er ført gjennom chassis-plata og festet mellom to plater med fire treskruer, som vist på figuren under. Dette hjulet vil kreve en noe annen utforming av chassiet enn det vi har beskrevet foran.



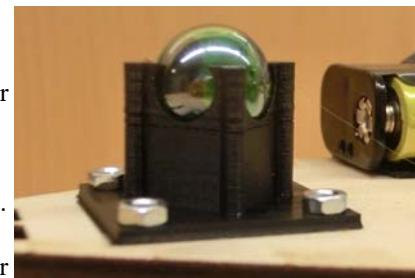
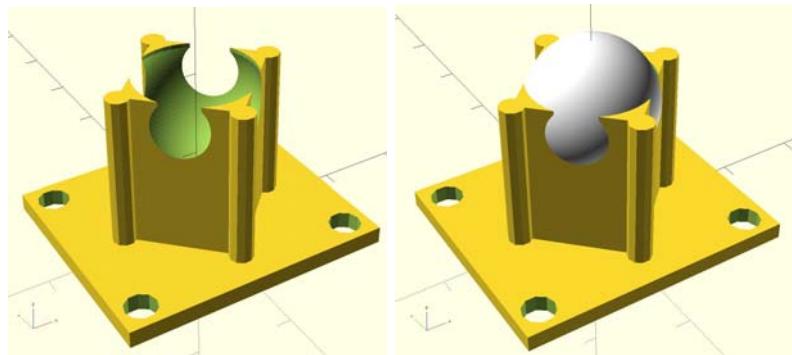
Dersom man ønsker å lage nesehjulet selv kan man f.eks. skrive det ut på en 3D-printer. Man tar da utgangspunkt i passende klinkekule og konstruerer en holder rundt denne. I vårt tilfelle valgte vi en klinkekule med en diameter på 15,5 – 16,0 mm.

Vi brukte så OpenSCAD for å lage selve holderen som vist på figuren under. Bildet over viser holderen med og uten kule slik den blir seende ut i OpenSCAD. En tomfingrerregel er at diametern til hullet bør være ca. 0,6 mm større enn kula.

OpenSCAD-koden er vedlagt i vedlegg C.1 på side 134.

Holderen til nesehjulet skrus fast med fire maskinskruer som er plassert i hjørnene i et kvadrat. Bildet til høyre viser hvordan det 3D-printede hjulet ser ut når det er montert.

Man kan også kjøpe nesehjul med kule fra f.eks. Pololu (<https://www.pololu.com/category/45/pololu-ball-casters>). Chassiet er også forberedt for en slik løsning. Som vist til høyre på figuren under så er det laget hull og gjort plass for montering av muttere i braketten for servoen.





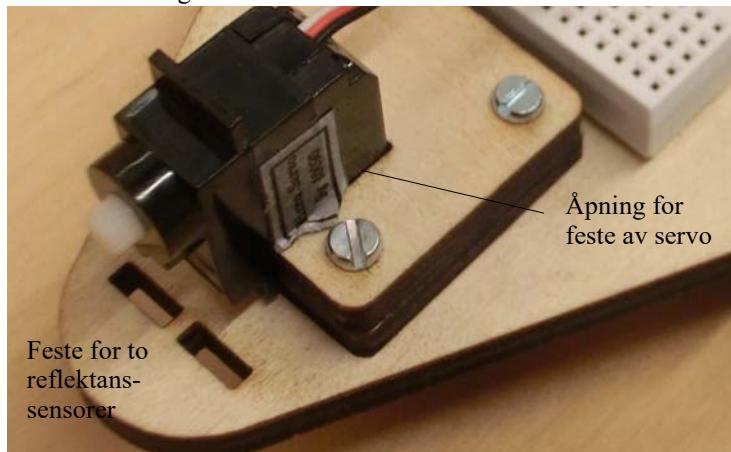
En liten spalt foran nesehjulet gjør det mulig å montere en *nesestøtte* som er en langt enklere løsning enn et hjul. Denne skjæres ut av chassiset og kan presses på plass i spalten som vist på bildet under.



En slik støtte vi kunne gli bort over et jevnt underlag, men egne seg dårlig på ujevnt underlag og tepper.

### 8.3.3 Servoer

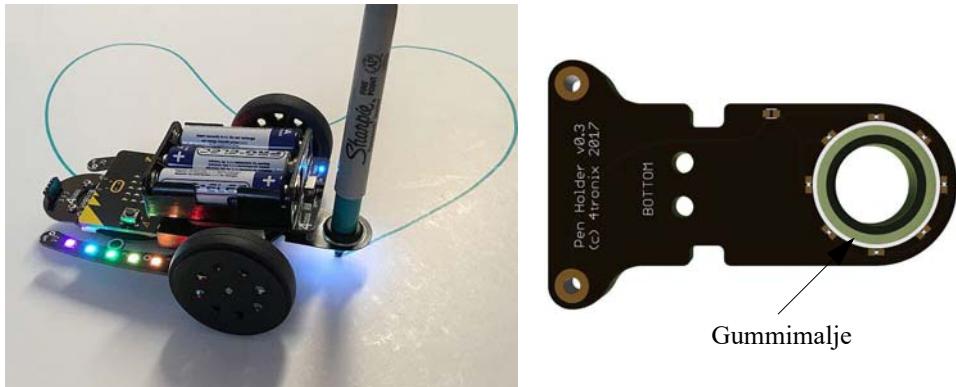
Feste for nesehjulet kombineres på oversiden med festeanordningen for en  $180^\circ$  servo. Siden festeanordningen er kvadratisk, kan den settes i en av fire posisjoner  $90^\circ$  på hverandre, slik at en ev. servo kan slå i ulike retninger.



Åpningen i kvadratet passer eksakt til en standard  $180^\circ$  servo som derfor bare kan presses ned i plata. Hensikten med en  $180^\circ$  servo i fronten kan f.eks. være å kunne slå med en nål for å spreng andre roboters ballonger, men kan selvfølgelig anvendes til langt fredeligere formål. De to  $360^\circ$  miniatyr servomotorene er presset ned i plata slik at man slipper annen festeanordning. Det er skåret ut to kvadratiske åpninger bak servoene. Dette er for at man skal komme til med trimmeverktøy for å justere "stoppområdet" for servoene dersom man velger å bruke et noe tykkere chassis som f.eks. 6 mm. For nærmere beskrivelse av justering av servoene se under avsnitt på side 41.

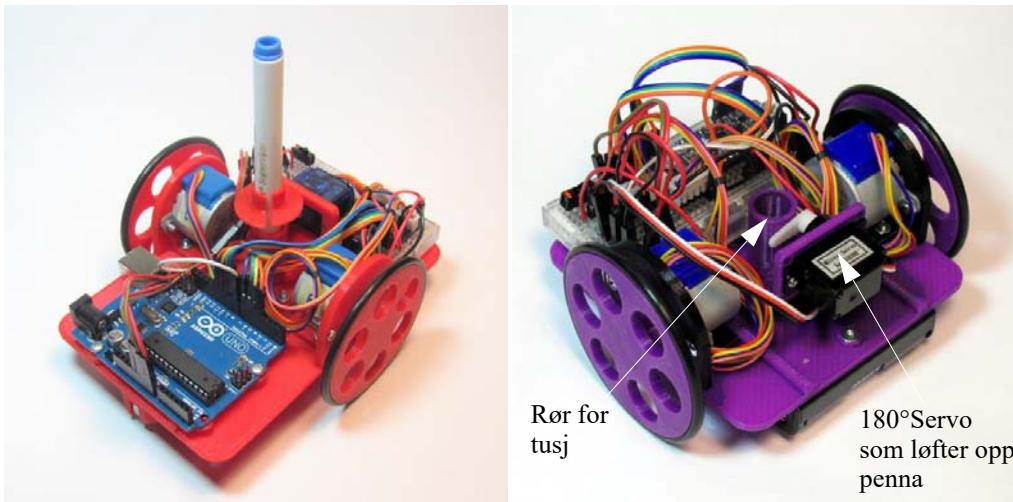
### 8.3.4 Tegnefunksjon

En artig funksjon hos noen roboter er at de kan tegne en strek der de beveger seg. Dette gjøres ved hjelp av en tusj som sklir langs papiret mens roboten beveger seg. Bit:bot har løst dette med en slags “tilhenger” bak på roboten<sup>27</sup>. Denne er utstyrt med en myk gummimalje som slutter omkring tusjen. Utfordringen blir lett at pennen slipper papiret med mindre holderen fjærer slik at tusjen presses ned mot papiret. En annen ulempe med en slik løsning er at pennen ikke er sentrert om aksen til roboten. Heller ikke er det mulig å heve pennan fra papiret mens den kjører.



Det finnes et alternativ til dette hvor tusjen sklir i et rør og på den måten kan følge papiret uavhengig av bevegelsene. En slik løsning vil neppe slippe papiret.

I denne Arduino-baserte roboten glir tusjen opp og ned i et 3D-printet rør. En flens omkring tusjen gjør det mulig å heve og senke den med en 180° servo.



27. Bildene er hentet fra: <https://shop.4tronix.co.uk/products/pen-holder-for-bit-bot-bitbot-or-crumbblebot>



Figuren under viser flensen som festes til penna og som servo-armen støter mot når tusjen skal løftes opp fra papiret.

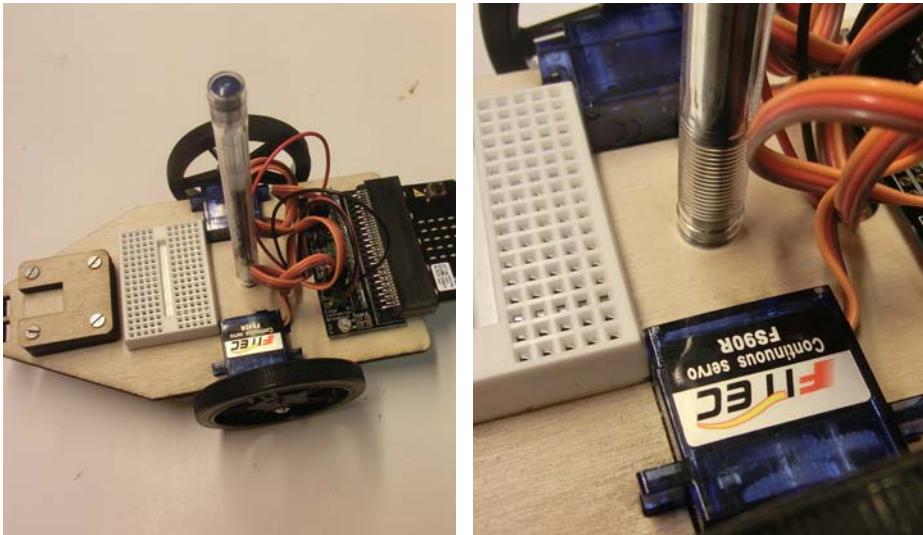


En tusj med filtpiss er nok det beste, alternativt kan en benytte gel-tusj som inneholder en refill som kan gli opp og ned i holderen. Vi har foreslått Pilot G-TEC-C4. Ulempen med denne er at den tegner en svært tynn strek (0,4 mm), fordelen er at selve tusj-refillen lett glir opp og ned i holderen som da også kan brukes som holder ombord på roboten. Dersom metallkjeglen foran skrus av, vil man se at selve plasthylsa ender i en innsnevring av røret med gjenger som ev. kan skrus fast i chassiet.



Metallkjegle

Bildet under viser hvordan pennen kan monteres. Dette er en temmelig primitiv måte å gjøre det på, men kan være utgangspunkt for et mer hensiktsmessig design, gjerne med heving av pennen ved hjelp av en servo.



I vår neste løsning har vi valgt å 3D-printe et rør med en flens som trykkes gjennom chassiset fra undersiden. Flensen er utformet slik at ved å vri den 90° så vil flensen kiles fast under batteriholdeiene og holdes på plass.

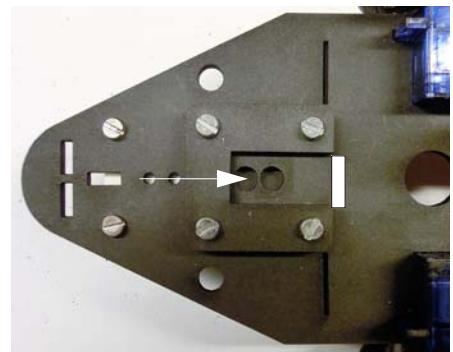


Røret vil stikke ca. 20 mm opp over chassiset og er tilpasset en filttusj av typen Edding 141 F (fine). Det er viktig at røret er så romt at penna kan gli lett opp og ned i røret. En tilsvarende moffe med flens er laget for å passe rundt penna (se figuren under). Denne må være så trang at den passer akkurat rundt penna, men uten å skli opp og ned med mindre at man bruker litt kraft.

Figuren under viser hvordan dette fungerer.

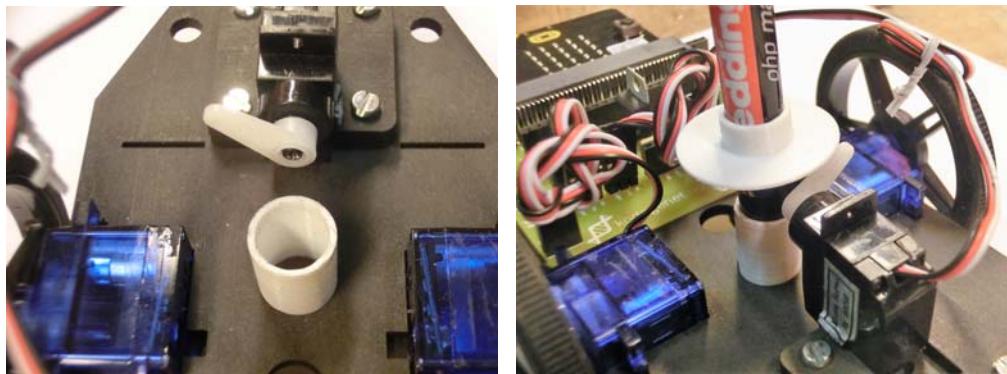


En 180° servo påmontert en liten arm stikkes inn under flensen slik at den kan løfte pennan som vist til høyre på figuren over. Som feste for servoen benyttes den samme anordningen som beskrevet for montering av servo over nesehjulet. Denne snus og flyttes til settet med hull som er nærmere sentrum av roboten, som vist på bildet til høyre. For å få til dette, må koblingsbrettet fjernes.





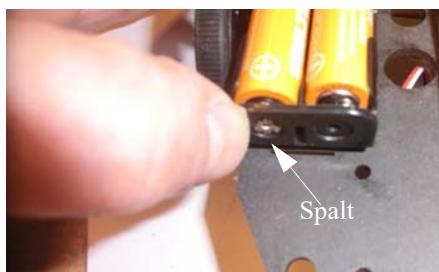
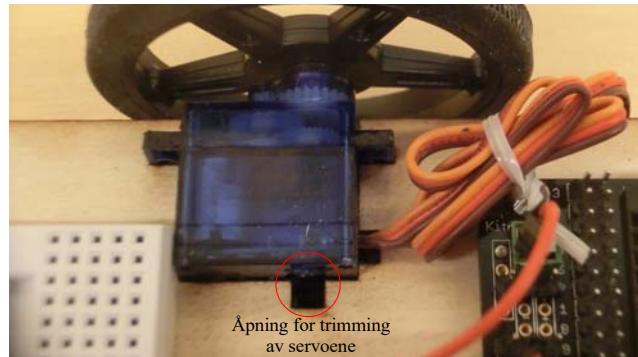
En 180 servo presses ned i åpningen i braketten som vist på bildet under.



En kan se for seg at pennen styres opp og ned ved hjelp av knapp A og B på Micro:biten (senderen).

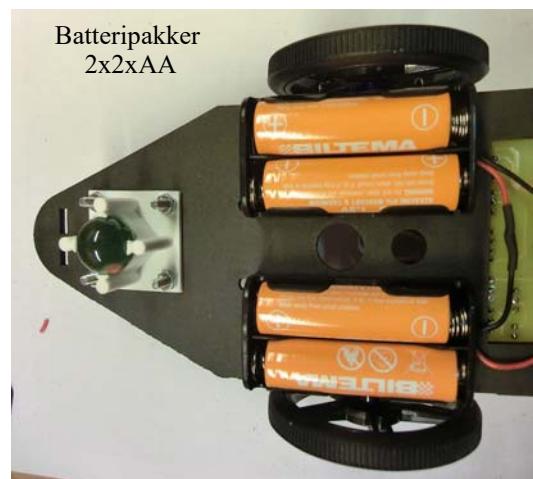
### 8.3.5 Batterier

Vi har valgt å montere to batteriholdere, hver for 2 stk. 1,5 V AA batterier. En smal spalte i chassiset indikerer kanten på batteriholderen. Det er viktig at batteriholderne monteres på rett sted både for å unngå konflikt med andre komponenter, men ikke minst for å sørge for at tyngdepunktet kommer tilstrekkelig langt foran slik at en unngår at roboten tipper bakover.



Figuren under viser hvordan de to batteriholderne er montert på undersiden av chassisplaten.

Vi foretrekker batteriholdere med innlegg fra oversiden da denne gjør det lettere å legge inn batteriene. Batteriholderne er festet til chassiset med dobbeltsidig tape. Vi har valgt å bruke 6V siden servoene får bedre ytelse med 6 enn med 3.3 V. Regulatoren på kortet senker spenningen til 3,3 V tilpasset Micro:biten i tillegg til at spenningen filtreres slik at støy fra servoene ikke når Micro:biten.

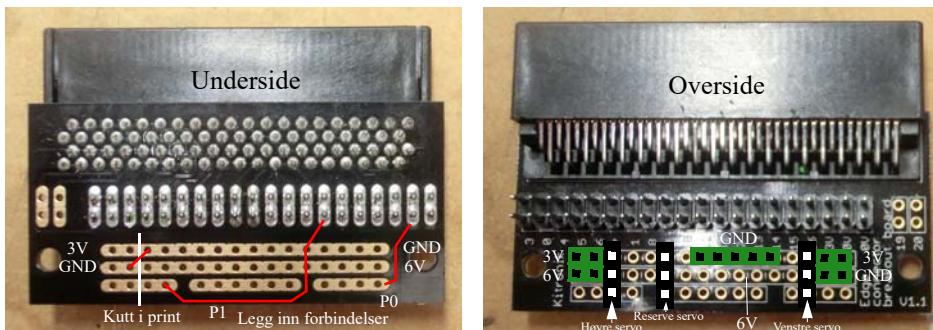


### 8.3.6 Kantkontakt- og Micro:bit kort

Her vil vi vise et par alternativer for tilkobling av Micro:bit-kortet til roboten.

#### Kitroniks kantkontaktkort

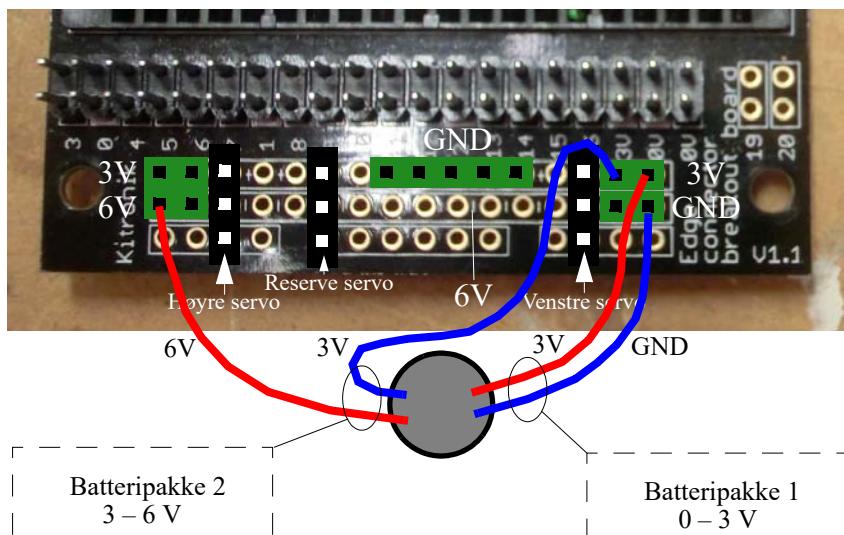
Et relativt rimelig alternativ er å bygge om kantkontakte som leveres fra Kitronik slik at de to 360° servoene og en ev. 180° servo for å styre en bevegelig arm, kan plugges rett inn i kretskortet til kontakten. Figuren under viser hvordan Kitroniks kantkontakt med kort kan modifiseres.



Fordelen med en slik løsning er at servoene kan kobles direkte til kantkontakte uten å gå om koblingsbrettet som kan brukes til andre ting. Dessuten kan seriekoblingen av de to batteripakkene kobles direkte inn i hylsekontaktene som vist under. Det er en fordel at uttaket fra batteripakkene legges nært opp til kantkontakte, det samme gjelder hullet for gjennomføring av ledningene fra

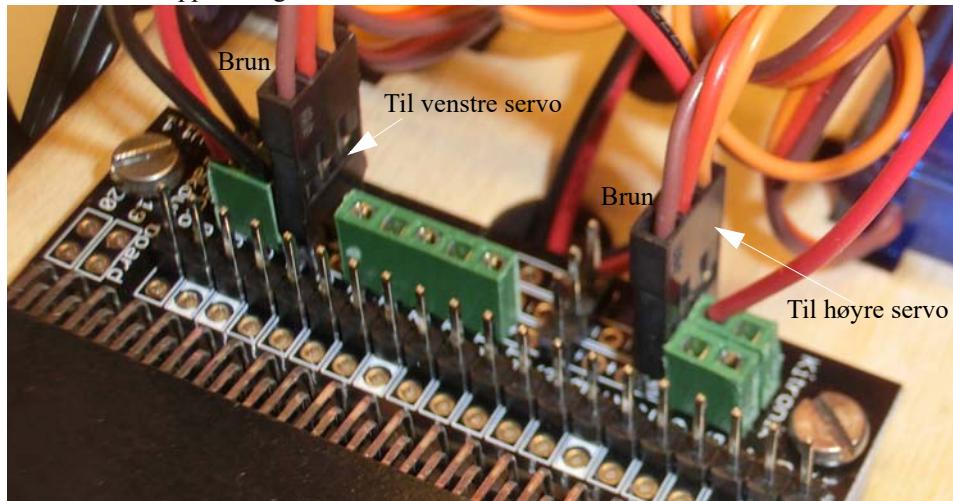


batteripakken.

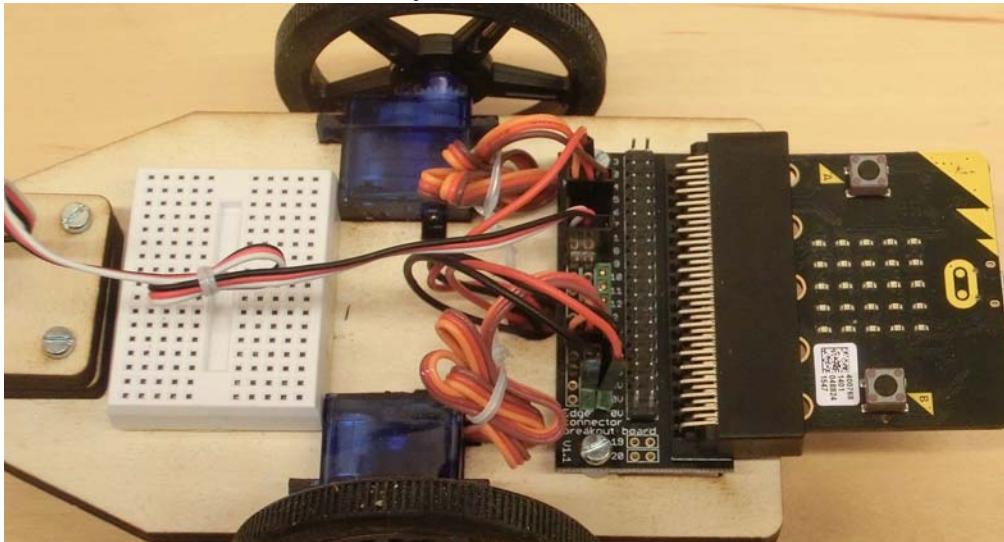


Som vi ser av figuren over så forsyner batteripakke 1 Micro:bit samtidig som den seriekobles med batteripakke 2 for å gi spenning til servoene. Det en bør være klar over er et en slik løsning ufiltrert kan gi støy fra servoene til Micro:bit-kortet. Vi har sett antydninger til det når vi bruker akselometeret på Micro:bit-kortet på robotten. Dette kan løses ved å filtrere spenningen til Micro:bit eller ved å skille spenningskilden for Micro:bit fra spenningskilden til servoene.

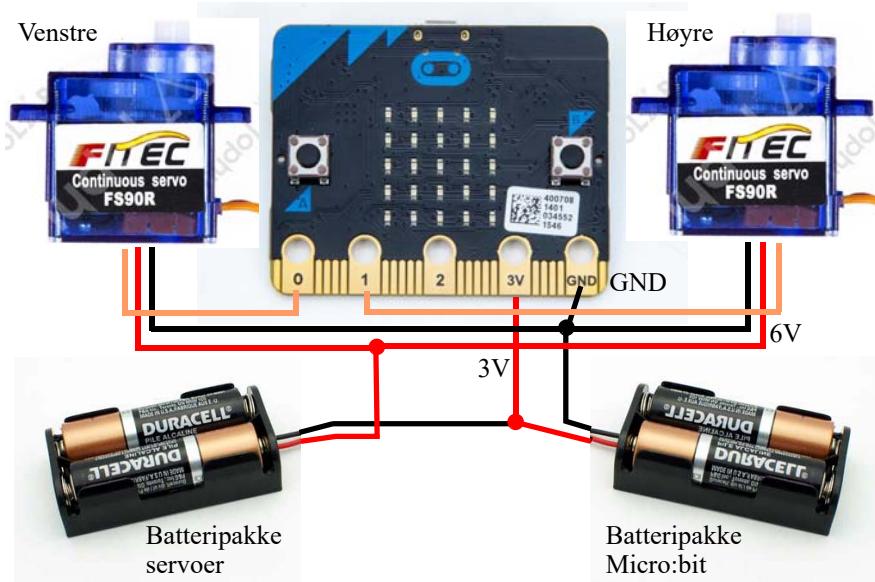
Bildet under viser oppkoblingen av de to  $360^\circ$  servoene.



Bildet under viser hvordan den tredje servoen tilkobles.



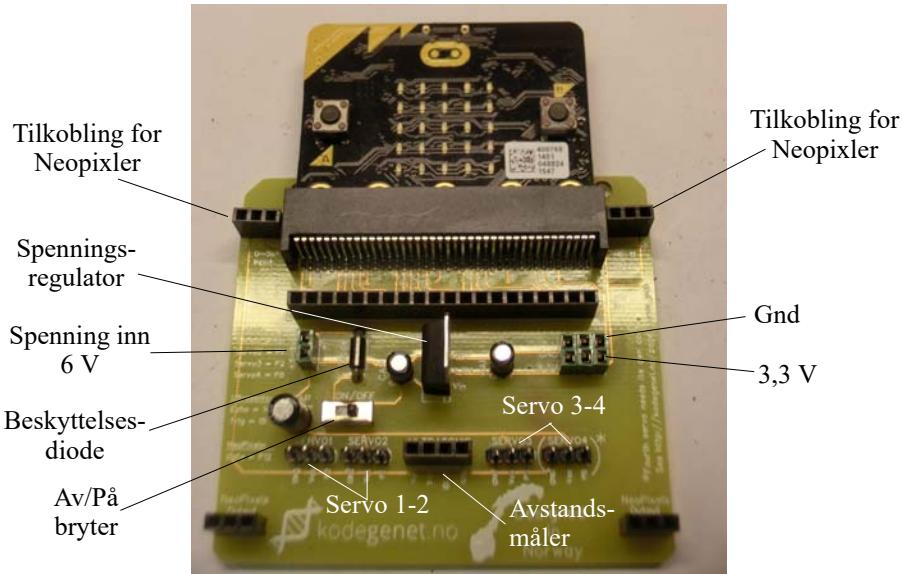
Figuren under viser et enkelt koblingsskjema for oppkobling av de to servomotorene og batteriene.





## Kodegenets Servo:bot-kort for Micro:bit

Senhøsten 2017 ble det utviklet et nytt kretskort for Micro:bit beregnet for bruk i roboter. Bildet under viser kortet.



### Servoer

Som vi ser så har kortet tilkobling for fire servoer, i praksis kun tre som kan styres med biblioteks-funksjoner. Den fjerde må ev. styres direkte med pulser. Følgende beskriver koblingen mellom servoutganger og porter:

- Servo 1 → P0 (port 0)
- Servo 2 → P1 (port 1)
- Servo 3 → P2 (port 2)
- Servo 4 → P8 (port 8)

### Ultrasonisk avstandsmåler

De fire hylsekontaktene er tilpasset en standard avstandsmåler som enten kan plugges rett i kontakten eller flyttes fram til koblingsbrettet lengre fram på roboten ved hjelp av jumpere.

I tillegg er alle portene (kontaktene) på kantkontakten tilgjengelig via den lange hylselisten nærmest kantkontakten. Følgende beskriver koblingen mellom trigger-puls og echo-puls og porter:

- Echo → P14 (port 14)
- Trig → P15 (port 15)



## Neopixler

Neopixler er produktnavnet på flerfargede lysdioder koblet i rekke. Disse kan styres fra Micro:bit ved å kobles til hylselistene øverst til venstre og høyre på kortet, som vist på bildet over. Følgende beskriver styringen av Neopxlene:

Data → P12 (port 12)

## Strømforsyning

Kortet kan tilføres batterispennin fra 4,5 – 15 V. Spenningen senkes til 3,3 V av spenningsregulatoren og som er spenningen Micro:bit skal ha. Siden servoene tilføres batterispenningen uregulert vil vi her begrense batterispenningen til 6 V som er den optimale spenningen for servoene. Spenningen kan brytes med en Av/På bryter. En diode beskytter mot feilkobling.



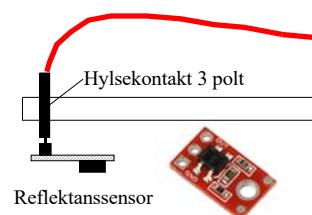
## Montering

Kantkontaktkortet monteres med to 3 mm maskinskruer. Utsparinger i chassiset under kretskortet gjør at det ligger støtt. Grunnen til dette er at loddepunktene under sokkelen stikker litt ut og vil uten utsparingene komme i veien under monteringen.

### 8.3.7 Alternative sensorer

#### Reflektanssensorer<sup>28</sup>

I tillegg er roboten klargjort for to reflektanssensorer i snuten på roboten. Disse kobles med tre ledninger til koblingsbrettet og videre til kantkontakten, ev. direkte til kantkontakten. Tegningen til høyre og bildene under viser hvordan reflektanssensorene (QTR-1A) er festet til chassiset. Disse sensorene kan brukes til å registrere forandringer i reflektert lys fra

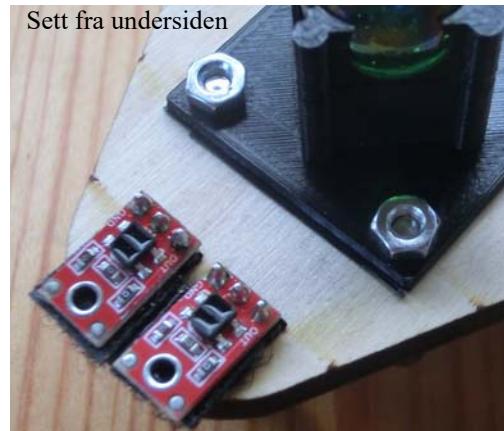
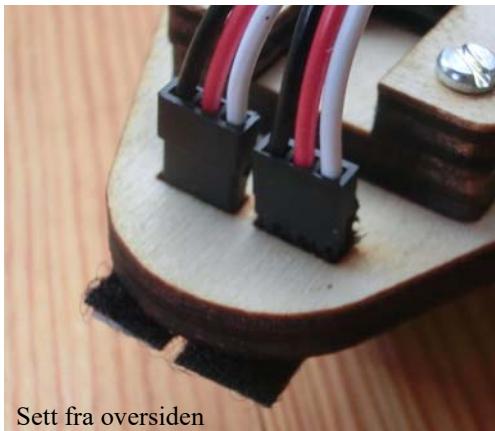


28. Kan kjøpes hos Polulu



---

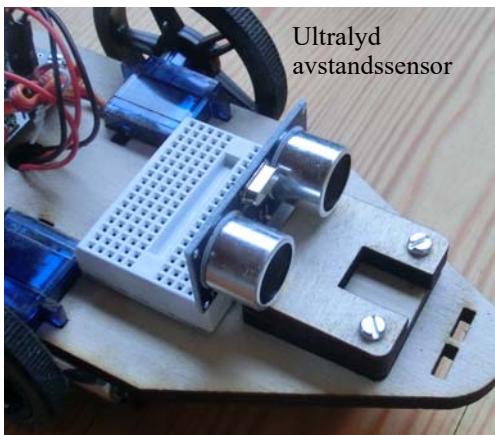
underlaget og kan for eksempel detektere en kant på et bord eller en sort stripe. En 3-polt hylsekontakt stikkes gjennom plata. Reflektanssensoren som har påmontert en trepolt stiftlist, kan da skyves rett opp i hylsekontaktene.



Denne delen av roboten er ennå ikke testet og vi er i skrivende stund ikke kjent med at det er skrevet programvare for disse sensorene for Micro:bit.

### Avstandssensor (Ultrasonic)<sup>29</sup>

Det er gjort plass på chassiset slik at koblingsbrettet kan monteres på tvers av kjøreretningen. På den måten er det mulig å montere sensorer som beker framover, f.eks. en ultralyd avstandssensor som vist på bildene under. Som det framgår av bildet er denne ennå ikke koblet til Micro:bit.



Denne sensoren registrerer hindringer og avstanden til hindringer. Den vil derfor være et godt hjelpemiddel til å unnvike hindringer som rager opp foran roboten.

---

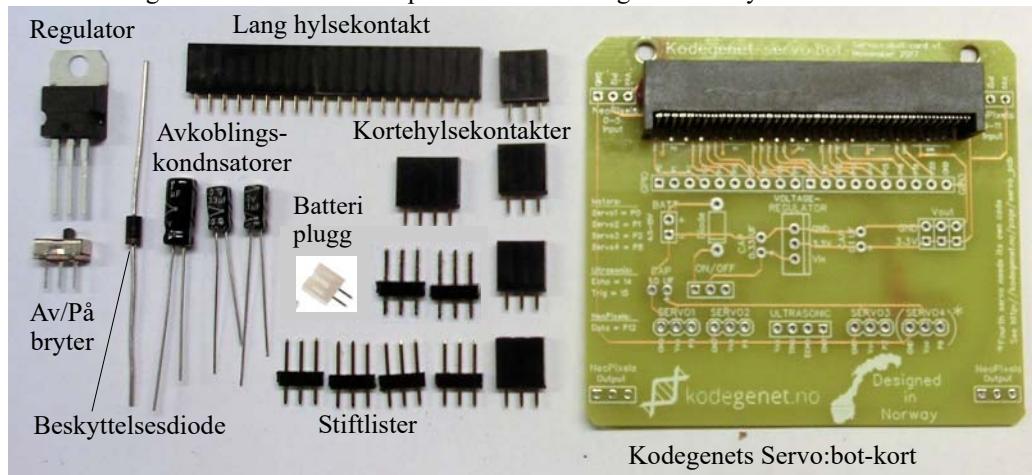
29. Kan kjøpes hos [www.kultogbillig.no](http://www.kultogbillig.no)

## 8.4 Detaljerte byggebeskrivelser

Her vil beskrive oppbygningen av vårt prototyp-robot slik den er pr. januar 2018.

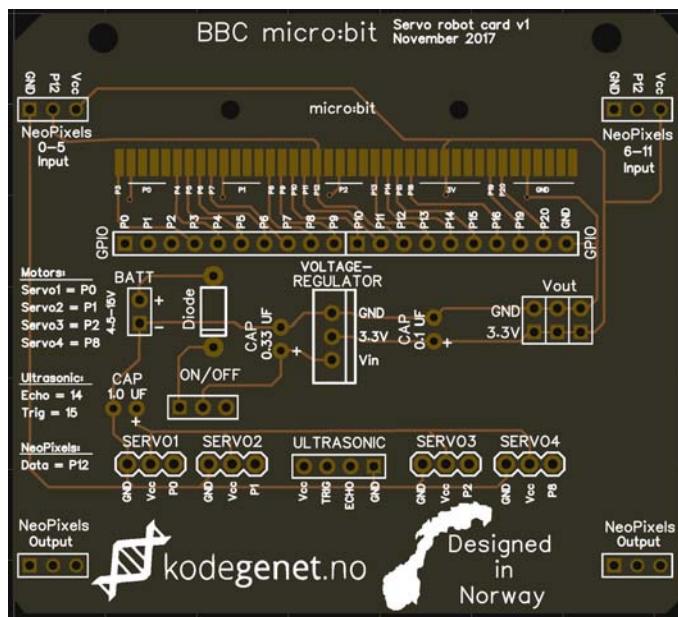
#### 8.4.1 Byggebeskrivelse for Kodegenets Servo:bot-kort for Micro:bit

Bildet under gir en oversikt over komponentene som trengs for å bestykke kortet.



Her har vi brukt fire topolte hylsekontakter. Disse vil i noen tilfeller bli byttet ut med en batterikontakt og to trepolte stiftlister som normalt følger med byggesettet.

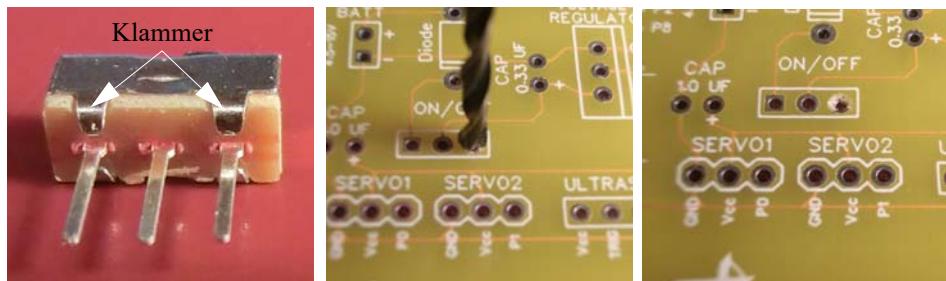
Figuren under viser komponentplasseringen på kortets *komponentside*.





## 1. Mulig feilkilde

Da vi først koblet opp kortet oppdaget vi at Av/På bryteren ikke virket. Uansett i hvilken stilling den sto var den på, unntatt i et lite område midt mellom av og på. Etter litt utforsking oppdaget vi at fire klammer (bildet under til venstre) som holder bryteren sammen, danned en kortslutning mellom de to ytterste loddelandene slik at når bryteren skulle ha vært avslått, så ledet metalltettsiden som omsluttet bryteren, strømmen forbi bryteren.



Problemet løses lett ved å bore bort loddelandet (paden) på komponentsiden av kortet som vist på bildene over. En trenger bare å vri en kvass bor på 2 – 3 mm rundt i hullet med hånda slik den fortinnde kobberringen forsvinner så skulle problemet være løst.

## 2. Monter lang hylsekontakt

Stikk kontakten inn fra komponentsiden. Lodd pinne til *loddelandene*<sup>30</sup> på *loddesiden* av kortet. Det kan være lurt å lodde fast ett bein først for deretter å se om kontakten står rett. Dersom den står litt skjevt så er det lett å varme opp loddeningen for så å rette opp kortet. Med alle beina fastloddet er dette nesten umulig.

## 3. Mini loddekurs

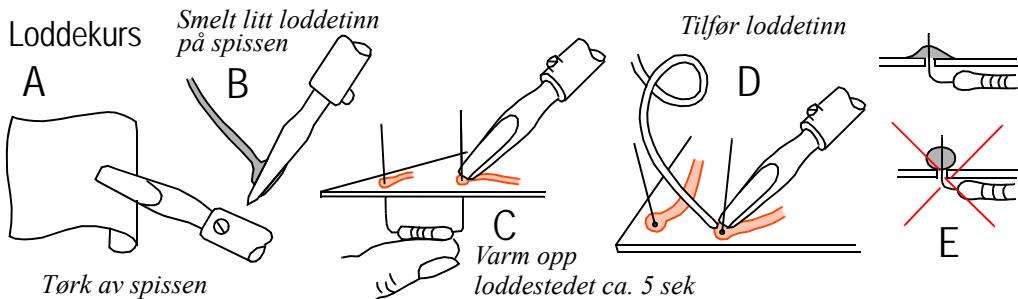
**VIKTIG: Alle komponenter skal loddes til banene på loddesiden selv om dette står i teksten.** Bein lengre enn 2 mm klippes av med avbiter etter at loddeningen er utført.



30. Loddeland er de blanke ringene på loddesiden (baksiden) av kortet. Komponentbeina skal loddes til disse ringene.



Det er viktig å lodde riktig, derfor ta en titt på disse fem tipsene



A) Tørk av den varme loddebolten på en våt klut eller svamp slik at spissen blir ren og blank

B) Smelt litt loddetinn på loddespissen slik at den blir blank å fin

C) Press loddespissen hardt ned mot loddestedet slik at både ringen på kortet og beinet til komponenten blir varmet opp. Hold den slik under hele loddningen.

D) Etter 4 – 5 sekunder er loddestedet varmt slik at loddetinn kan tilføres.

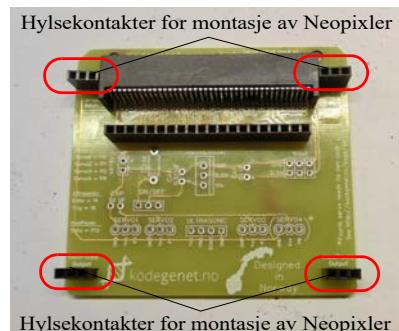
Tilfør loddetinn der loddespissen møter beinet og loddestedet.

E) Hos en god loddning skal loddetinnet flyte oppover beinet og ut over loddestedet.

Dannes det en kule rundt komponentbeinet har ikke loddestedet vært varmt nok og loddingen må smeltes på nytt

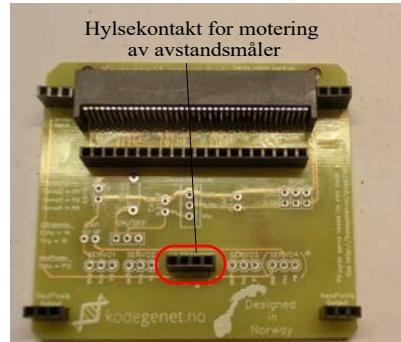
#### 4. Monter hylselister for Neopixler

Stikk fire trepolte hylsekontakter ned i hver sine hull som vist på figuren til høyre og lodd beina til loddlandene på loddesiden. Her kan det være spesielt viktig å lodd fast ett bein først for så ev. å gjøre justeringer.



#### 5. Montering av hylsekontakt for avstandsmåling

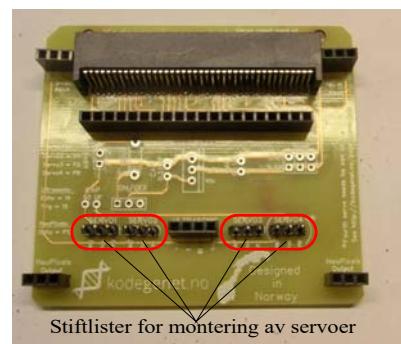
Stikk den firepolede hylsekontakten ned i hullene markert med "Ultrasonic" og lodd fast beina til loddelandene på loddesiden.





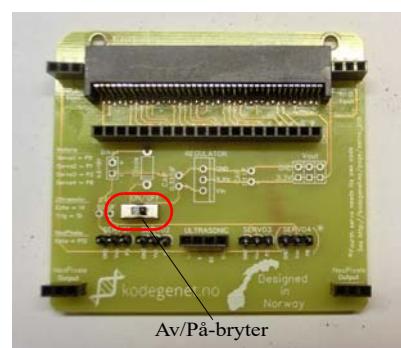
## 6. Montering av fire trepolte stiftlister

Stikk den korte enden av de fire trepolete stiftlistene ned i hullene der det står SERVO1, 2, 3 og 4. Lodd fast en av pinnene på hver av stiftlistene. Unngå trykk på enkelt pinner under oppvarmingen da de lett forskyver seg i den sorte plastholderen. Sjekk at stiftene står rett, ev. rett opp. Lodd fast samtlige stifter når de står rett.



## 7. Montering av Av/På-bryter

Stikk de tre beina til glidebryteren ned i de tre hullene som vist på bildet til høyre. Det er det samme hvilken vei bryteren monteres.

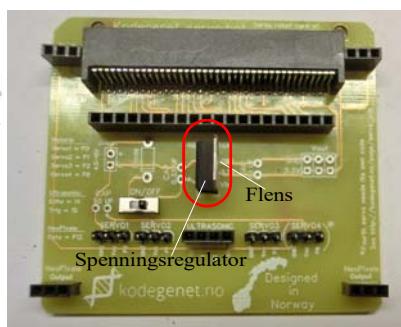


## 8. Montering av spenningsregulatoren

Spenningsregulatoren er en trebent komponent med en metallflens på den ene siden.

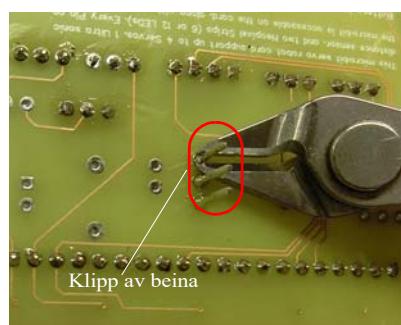


LD33V



## 9. Klipp av beina

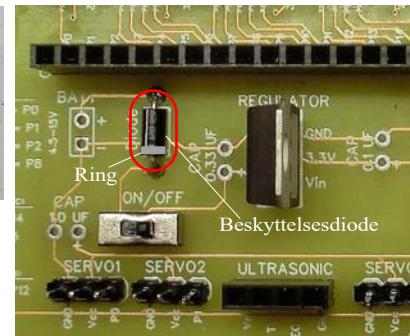
Etter loddning vil beina til regulatoren stikke ut på baksiden. Bruk en *sideavbiter* og klipp av beina tett inntil loddgene.



## 10. Montering av beskyttelsesdiode

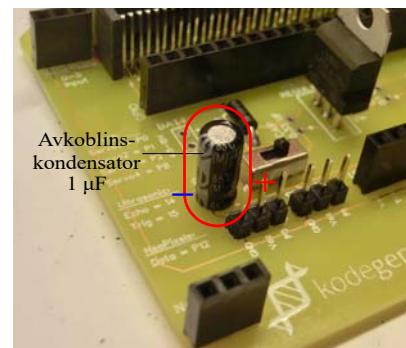
Bøy beina på dioden i rett vinkel tett inn til diodehuset som vist til venstre på bildet til høyre. Stikk beina til dioden gjennom hullene der det står *Diode* på kretskortet. Pass på at dioden plasseres rett vei med den grå ringen i nederkant som vist på bildet.

Lodd beina på baksiden og klipp av tett inntil loddingen.



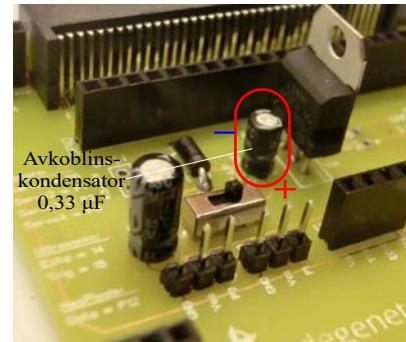
## 11. Montering av avkoblingskondensator $1\mu\text{F}$

Monter avkoblingskondensatoren som vist på bildet til høyre. Merk at kondensatoren må plasseres rett vei – til venstre som vist på bildet til høyre.



## 12. Montering av avkoblingskondensator $0,33\mu\text{F}$

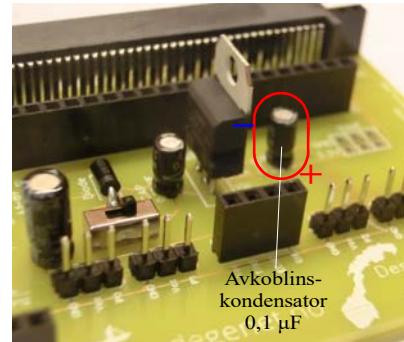
Monter avkoblingskondensatoren som vist på bildet til høyre. Merk at kondensatoren må plasseres rett vei – bak som vist på bildet til høyre.





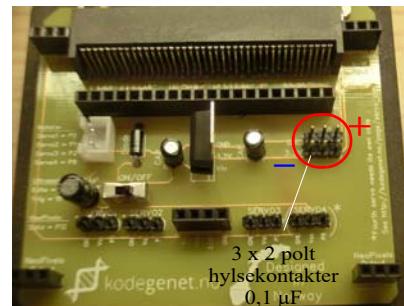
### 13. Montering av avkoblingskondensator 0,1 $\mu$ F

Monter avkoblingskondensatoren som vist på bildet til høyre. Merk at kondensatoren må plasseres rett vei – bak som vist på bildet til høyre.



### 14. Montering av 2 x 3 stiftlister

Monter 2 x 3 stiftlister. Merk at stiftlistene skal stå tett sammen. Disse er ment å være en tilkoblingsressurs for spenning (3.3 V) og jord (GND). Merk at man må bruke jumpere med hunn-kontakter.



### 15. Montering av batterikontakt

Monter en JST<sup>31</sup>-plugg (batterikontakt) for tilkobling av batterispenninng (6 V eller 9 V). MERK: Legg spesielt merke til hvilken vei pluggen står, det er avgjørende om polariteten blir riktig.

Kortet er nå klart for å tilkobles batterispenninng og servoer.



## 8.5 Test av kortet og servoene

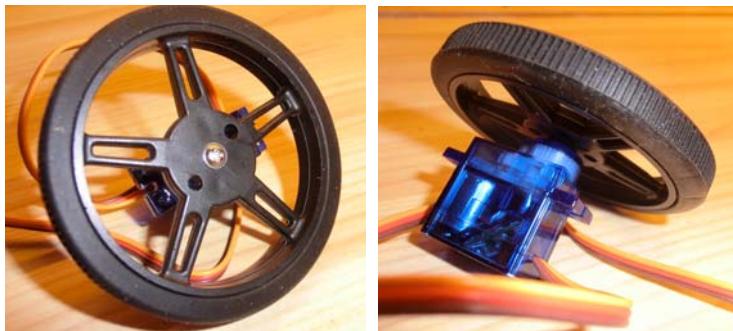
Vi skal nå teste at kortet fungerer som det skal for at vi skal få til dette må vi koble på servomotorene. For at vi skal være istrand til å se at de beveger seg kan det være lurt å sette på hjulene.

31.JST - Japan Solderless Terminal



## 1. Sett på hjulene

Før servomotorene monteres på chassiset kan vi sette på hjulene. Disse presses inn på akslingen og holdes på plass av en skrue som skrus inn i navet. Skroen ligger i posen som følger med servoene.



## 2. Kontroller Servo:bot kortet – NB!

Tiden er nå inne for å kontrollere at kortet er koblet opp riktig og at det ikke er kortslutninger på kortet. Ta gjerne kontakt med en av kurslederne slik at dere sammen kan kontrollere kortet.

## 3. Koble servoene til Servo:bot kortet

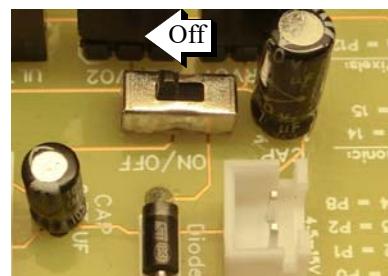
Plugg ledningene fra servoene ned i de to stiftkontaktene "Servo 1 og 2".

PASS PÅ at den sorte ledningen er på rett side av kontakten (se bildet til høyre).



## 4. Sett på/av-bryter i "Off"

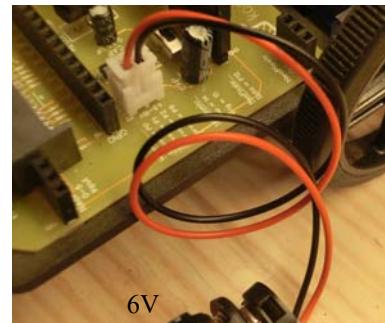
Før batteriet kobles til, sett på/av-bryteren i "Off" posisjon. Dvs. skyv den mot venstre sett fra kantkontakten som vist på bildet til høyre.





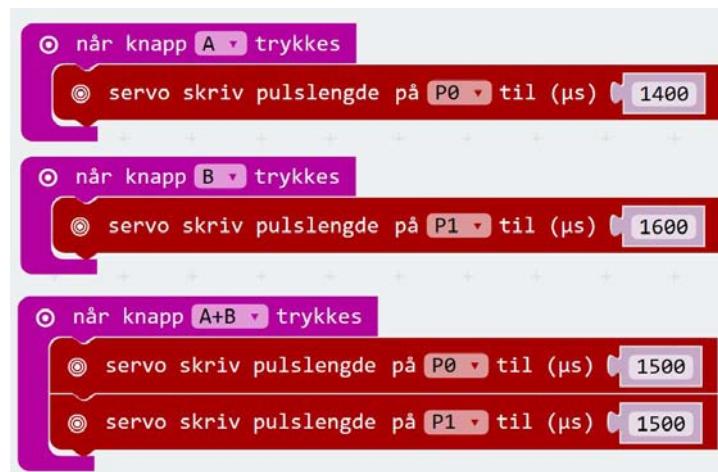
## 5. Koble til batteripakken og slå på strømmen

Siden vi ikke har montert roboten, velger vi å bruke en 6V batteripakke med JST-plugg (hvit). Sett JST-pluggen inn i den hvite sokkelen. Pass på at denne står riktig vei. Sett på/av-bryteren i “on” posisjon.



## 6. Legg inn testprogrammet i Micro:biten

Vi skal nå legge inn et enkelt testprogram i Micro:biten som skal styre roboten.



Når knapp A trykkes starter høyre motor, når knapp B trykkes starter venstre motor. Når knapp A og B trykkes skal begge motorene stoppe. Om de ikke står helt stille må de trimmes slik at de står i ro.

## 7. Trimming av motorene

Trykk A+B slik at motorene står stille. Om de likevel går bruk et lite stjerne-skrutrekker og juster skruen bak på motoren som ikke står stille. Juster skruen til motoren står i ro. Det skal svært lite til.





## 8. Modifiser programmet

Skriv i følgende program og forklar for hverandre hvordan programkoden virker. Undersøk om beskrivelsen ble riktig.

```
ved start
  sett flagg 1 til 0
  sett flagg 2 til 0

når knapp A trykkes
  hvis [flagg 1 = 0]
    servo skriv pulslengde på P0 til (μs) 1600
    sett flagg 1 til 1
  ellers hvis [flagg 1 = 1]
    servo skriv pulslengde på P0 til (μs) 1400
    sett flagg 1 til 0

når knapp B trykkes
  hvis [flagg 2 = 0]
    servo skriv pulslengde på P1 til (μs) 1600
    sett flagg 2 til 1
  ellers hvis [flagg 2 = 1]
    servo skriv pulslengde på P1 til (μs) 1400
    sett flagg 2 til 0

når knapp A+B trykkes
  servo skriv pulslengde på P0 til (μs) 1500
  servo skriv pulslengde på P1 til (μs) 1500
```

## 8.6 Byggebeskrivelse robot

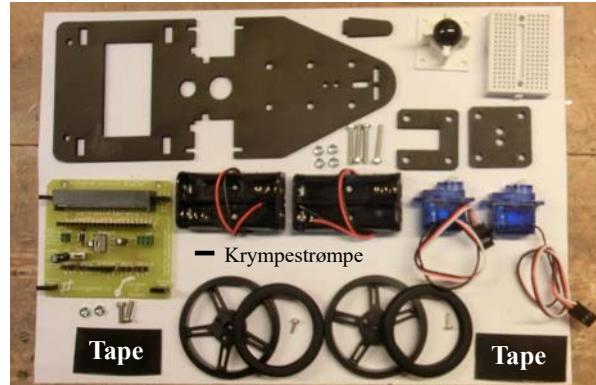
Bygge beskrivelsen av roboten er delt i to:

1. Oppbygging av grunnenhet med chassis
2. Oppbygging av tegneenhet



### 8.6.1 Byggebeskrivelse – Grunnenhet med chassis

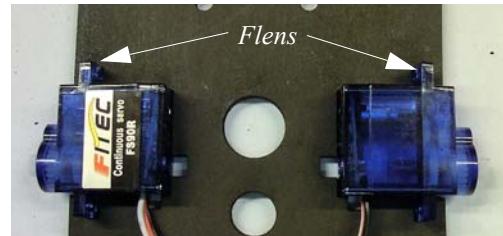
Dette vedlegget gir en detaljert byggebeskrivelse for en enkel standardutgave av roboten. Med utgangspunkt i denne kan man lage sine egen versjoner.



Bildet viser en oversikt over komponentene som inngår i byggessettet (se også komponentliste vedlegg A.1 på side 122). I enkelte sammenhenger vil det være aktuelt at deltagerne selv skriver ut chassis og nesehjul ev. deler til pennholder.

#### 1. Montering av servoer

Servoene presses ned i de profilerte åpningsene på hver side av chassiset. Det kan være nødvendig å file litt i spaltene der flensene skal inn. Servoene presses ned i åpningen til de ligger kant i kant med plata på undersiden.



#### 2. Montering av kantkontakt-kort

Vi forutsetter at kantkontakt-kortet er ferdig oppkoblet (se byggebeskrivelse avsnitt 8.4.1 på side 98). Kantkontakt-kortet monteres med to skruer (M3, 12 mm) med mutter (M3). Utsparingene i chassiset gjør det unødvendig å løfte plata fra kortet.



### 3. Klargjøring av batteriholdere

Sett en dobbeltsidig tape på undersiden av hver av batteriholderne. Dersom batteriholderne krever batterikontakter, klips disse på kontaktene.



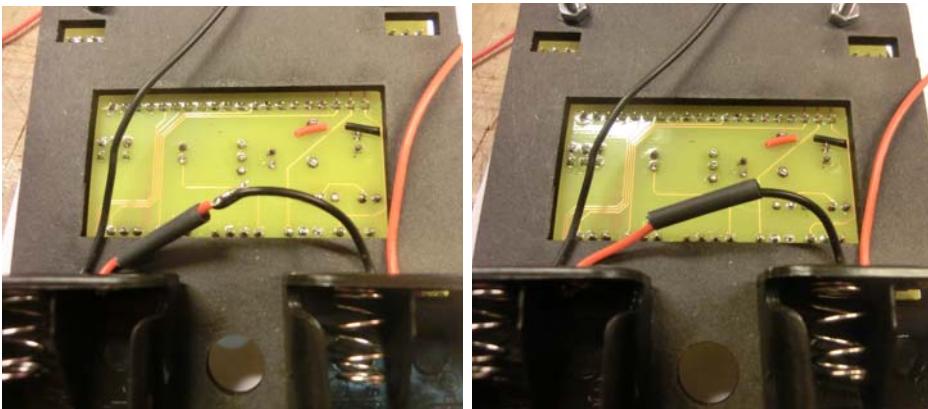
### 4. Monter batterikassettene

De to batterikassettene monteres under hver av de to servomotorene med dobbeltsidig tape. Framenden av batteriholderne skal gå kan ti i kant med to spalter i chassiset. Ledningene skal vende mot kantkontakten. Dersom ledningene skal tilkobles batterikontakten på kortet, føres ledningene gjennom hullet og opp på oversiden.



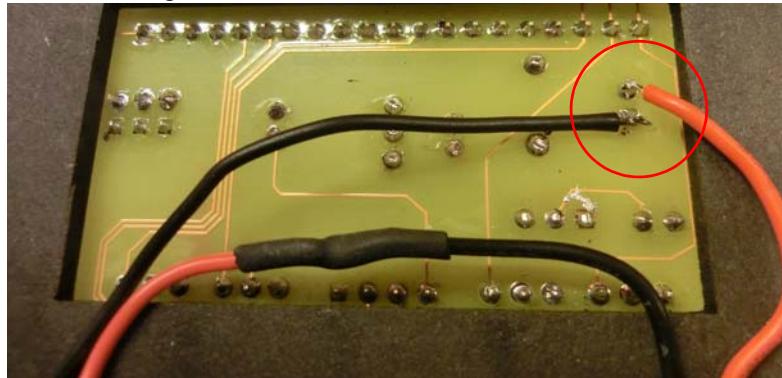
### 5. Tilkobling av batteriledninger

De to batterikassettene skal seriekobles slik at vi får totalt 6 V. Dette gjøres ved å lodde den røde ledningen (+) fra den ene til den sorte (-) fra den andre, som vist på bildene under. Husk å tre på krympestrømpa før loddning. Dra deretter strømpa over loddestedet og bruk loddebolten til å varme opp strømpa slik at den krymper omkring skjøten.

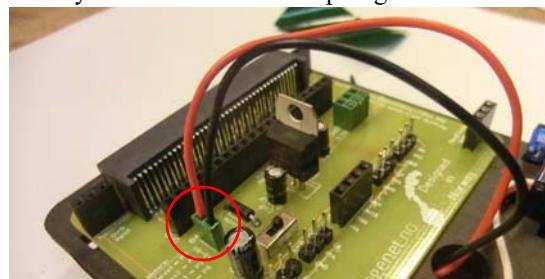




Den mest elegante løsningen er å lodde de to gjenværende ledningene til undersiden av batterikontakten som vist på bildet under. En må da tilpasse lengden til ledningene og sørge for at polariteten blir riktig som vist.

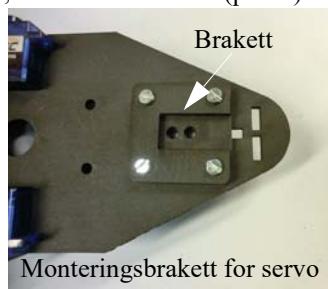
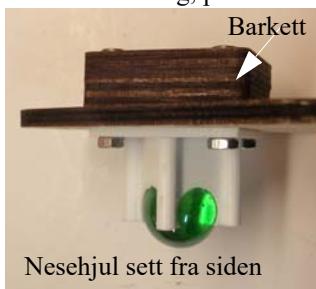


Alternativt kan bruke ledningene i full lengde, føre dem gjennom hullet i plata og stikke dem ned i JST-kontakt eller hylsekontakten som vist på figuren under.



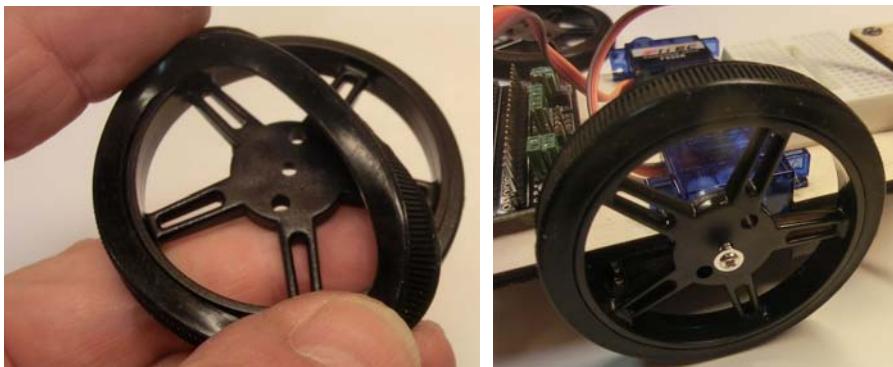
#### 6. Montering av nesehjul

Nesehjulet monteres med fire skruer med mutter (M3, 20 mm). Sammen med nesehjulet kan man samtidig, på oversiden, montere en brakett (plater) for montasje av en 180° servo.



## 7. Alternativ 1. Montering av gummihjul

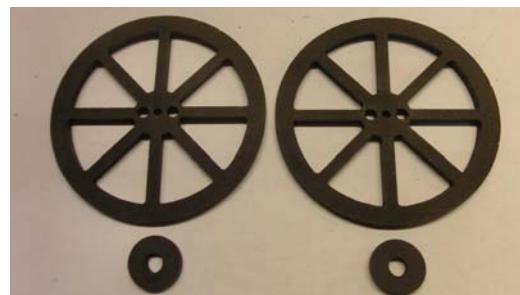
Gummiringen legges på felgen slik at forhøyningen langs felgen fyller slissen til dekkene. Hjulene skrus så fast til akslingen med en liten skrue som følger med servoene.



## 8. Alternativ 2 - Bruk av laserkuttede hjul

I chassis-fila medfølger to hjul. Disse kan alternativt brukes i stedet for standard plast- og gummihjul som kan kjøpes.

Sammen med hjulene følger to navringar som limes (ev. skrues) fast i navet til hvert av hjulene. Tanken er at navringene skal skyves inn på akslingen til servomotoren. Deretter skrus en skrue gjennom hjulet og inn i akslingen til servoene.



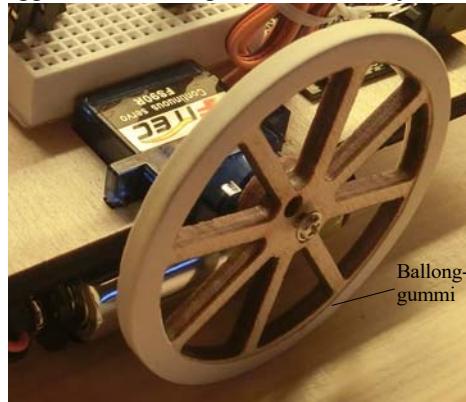
Bildene under viser montering av navringene på hjulene.





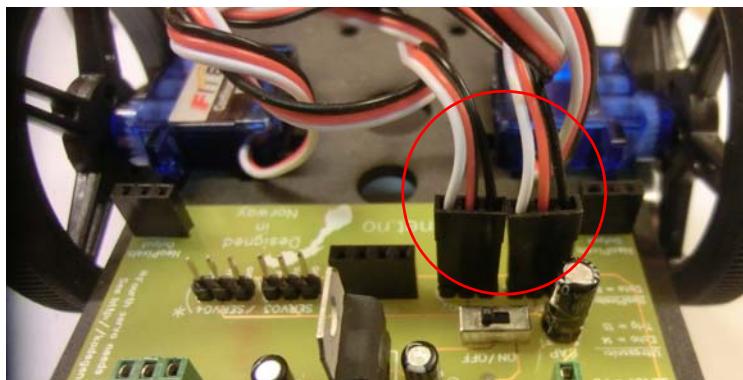
---

En kan legge en ballonggummi rundt felgen for å øke friksjonen.



9. *Tilkobling av servoer*

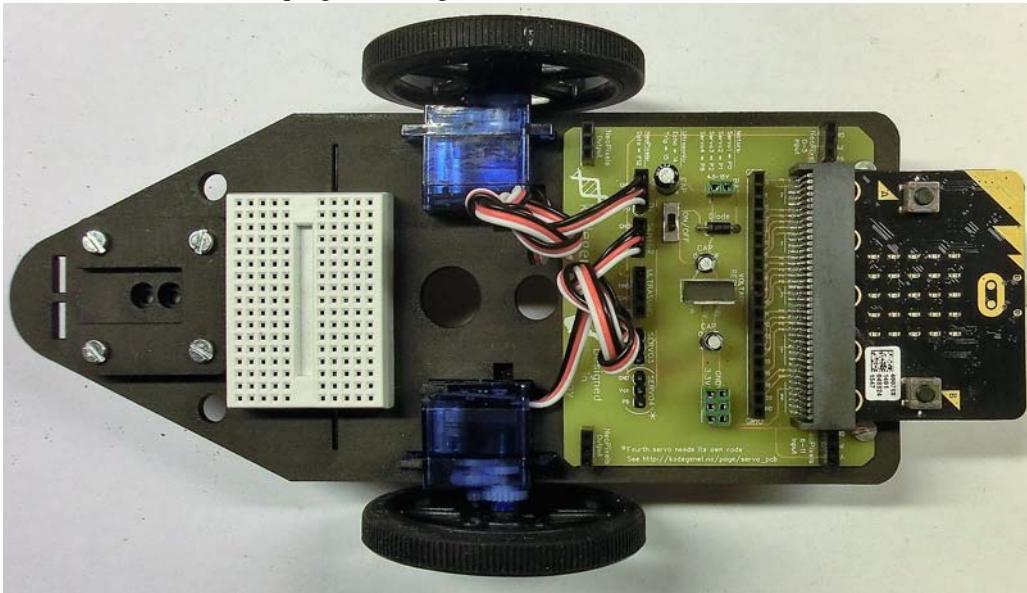
De to servoene kobles til de to stiftlistene som vist på bildet under (innringet). Pass på at hylsekontakten blir stående rett vei, med den hvite ledningen til venstre sett bakfra som vist på bildet.



10. *Den ferdige roboten*

Merk at Micro:bit skal monteres slik at lysdiodedisplayet er vendt opp. Dersom man har behov for å utstyre roboten med ekstra elektronikk, så kan det gjøres på et koblingsbrett som monteres foran servoene som vist på bildet under. Dersom man ønsker å montere en penn, bør man kanskje vente med å montere koblingsbrettet.

Så er roboten klar for programmering.



Vi har erfart at roboten kan bli litt baktung til tross for at plasseringen av servoyer og batterier skal ta hensyn til dette. I tilfelle at roboten skulle bikk bakover er det laget to hull, et på hver side av servobraketten, hvor det er mulig å montere skruer med muttere som kan fungere som vekter.



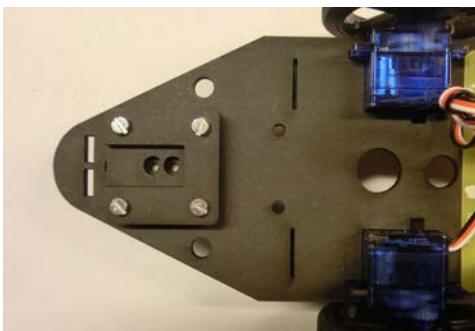
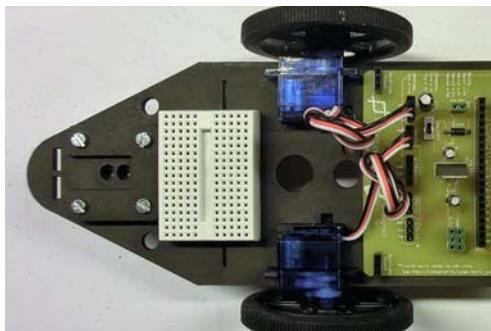
#### 8.6.2 Byggebeskrivelse – Tegneenhet

I dette avsnittet skal vi beskrive hvordan vi kan utstyre roboten med en tusjpenn slik at roboten kan tegne streker mens den kjører. Vi ønsker også å kunne fjernstyre pennen ved hjelp av knappene på Micro:bit.



---

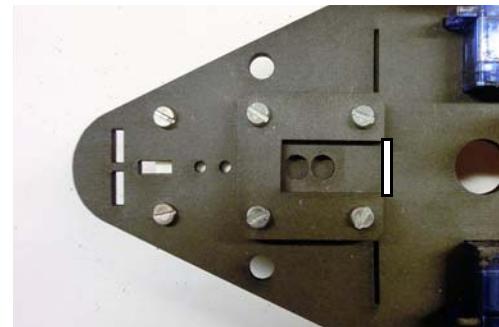
1. *Fjern koblingbrettet*



Dersom koblingbrettet er montert med den medfølgende dobbeltsidige tapen, kan det være vanskelig å ta av dette uten ødelegge brettet. Det som kan skje er at tapen tar med seg hele baksiden av koblingsbrettet slik at koblingskinnene blir med ut.

2. *Snu servobraketten*

Chassiet har flere sett med hull som gjør det mulig å flytte og endre retning på braketten slik at 180° servoen kan snus mot hullet som er ment for pennen. En spalt i plata gir plass til flensen til servoen når den skyves inn gapet til brakken.



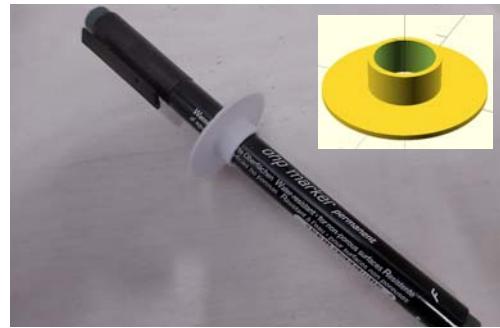
3. *Monter røret som styrer pennen*

Et 3D-printet rør med flens stikkes opp fra undersiden av chassiet og låses ved å dreie røret 90° slik at flensen smetter under batteriholderne.



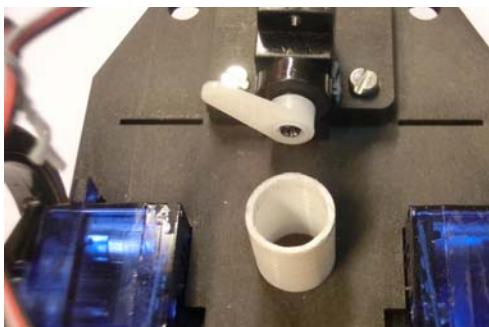
#### 4. Monter muffe med flens på pennen

En 3D-printet muffe tilpasset den valgte pennen skyves inn på pennen. Denne er så trang at muffa blir sittende i den ønskede posisjonen. Muffa skal plasseres slik at når servoens arm heves, så skal pennen forlate papiret.



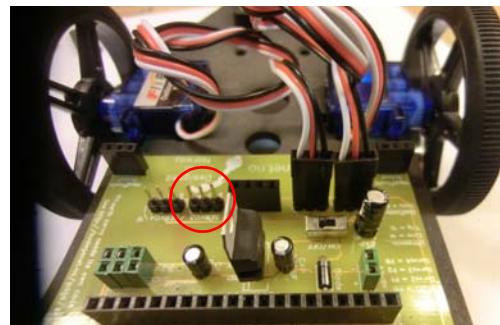
#### 5. Montering av servo og penn

En 180° servo presses inn i braketten og pennen slippes ned i røret. Armen på servoen monteres slik at den er i stand til å løfte pennen opp fra papiret.



#### 6. Tilkobling av servo

Servoen kobles til servo 3 på kretskortet (innringet på bildet til høyre).



### 8.7 Programmering av Micro:bit roboten

I dette avsnittet skal vi se hvordan vi kan bygge opp programvaren som skal styre roboten etter intensjonen. I innledningen har vi valgt meget enkle programmer som har stort potensial for forbedring. I tillegg ønsker vi å løfte og senke pennen fra papiret, til denne funksjonen ønsker vi å bruke de to bryterne A og B.

For å være i stand til å programmere de tre servoene må vi vite hvilke porter som styrer de ulike servoene. Disse opplysningene står også på kretskortet.



---

Servo 1 → P0 (port 0)

Servo 2 → P1 (port 1)

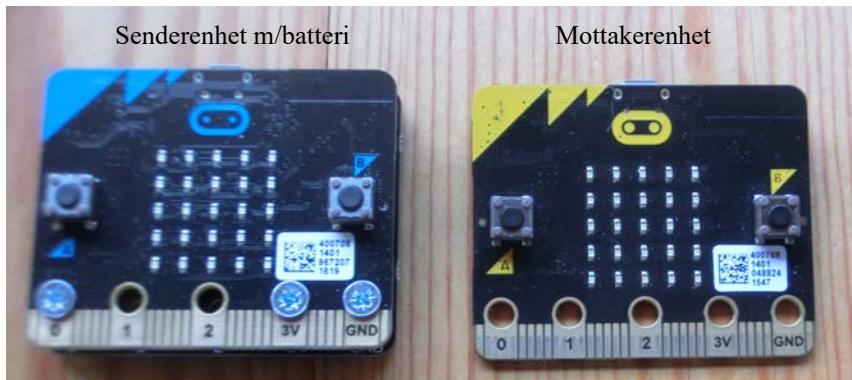
Servo 3 → P2 (port 2)

Servo 4 → P8 (port 8)

I tillegg er kretskortet forberedt for en avstandssensor (HC-SR04)

### 8.7.1 Styring av roboten

Siden målet er at roboten skal kunne styres framover, bakover og svinge til høyre og venstre fra en håndholdt Micro:bit, så må vi lage program både for senderen (Micro:bit'en i hånda) og mottakeren (Micro:bit'en på roboten). Både sender- og mottakerenheten er Micro:bits. Senderenheten må ha eget batteri, mens mottakerenheten kan forsynes fra robotens strømkilde.

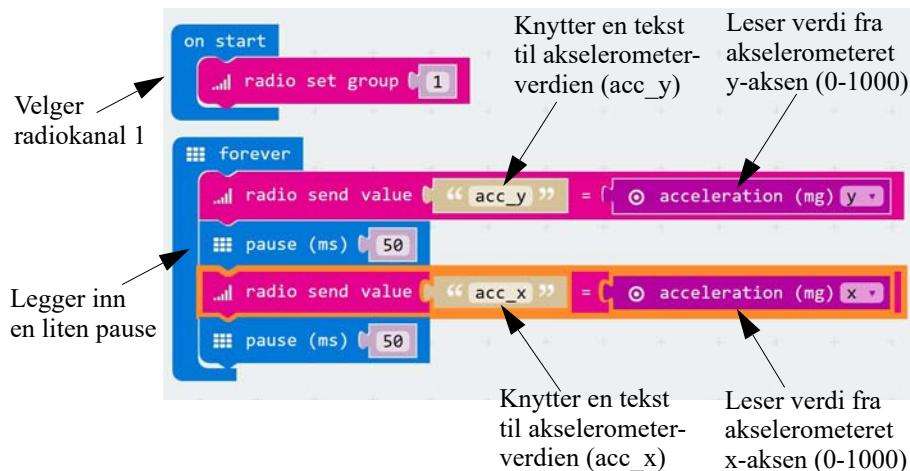


**Oppdraget:** Siden en Micro:bit kun har to knapper velger vi å bruke akselerometeret for styring. Vi ønsker at roboten skal kjøre forover når vi heller senderen framover og bakover når vi heller den bakover. Tilsvarende vil vi at den skal svinge til venstre når vi heller den mot venstre og tilsvarende mot høyre. I tillegg vil vi at farten og hvor brått den svinger skal være avhengig av helningsvinkelen.

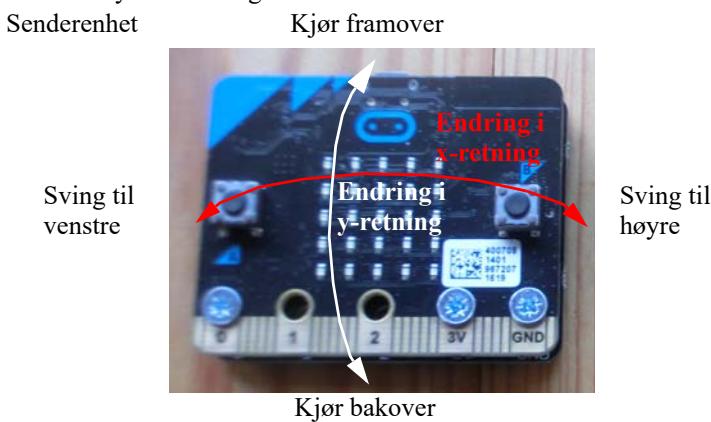


### 8.7.2 Programmering av senderenheten

Figuren under gir et forslag til programmet i senderen.



Figuren over er nesten selvforklarende. Variasjoner i y-aksen er knyttet til tilting forover og bakover, mens x-aksen er knyttet til tilting sideveis.

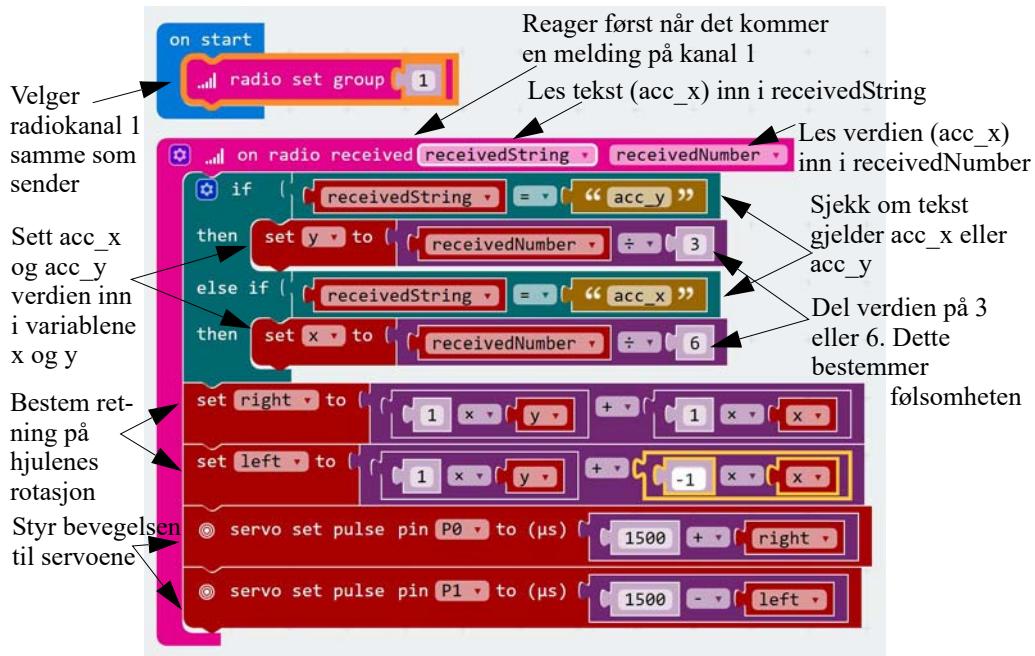


Å holde orden på retningene er mottakerens oppgave. En kan imidlertid med fordel legge inn piler på displayet slik at fjernstyreren ser at senderen fungerer korrekt.



### 8.7.3 Programmering av mottakerenheten

Programmet for mottakeren er noe mer komplisert da dette programmet må tolke signalene fra senderen. Figuren under viser programmet for mottakeren som er så kompakt det er mulig å lage det.

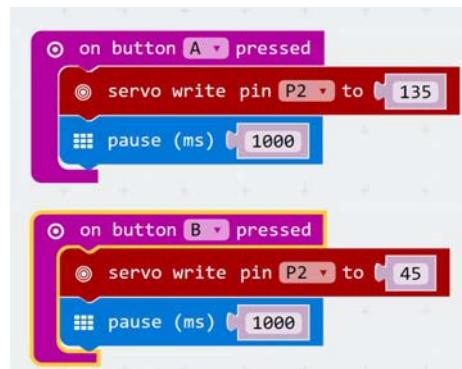


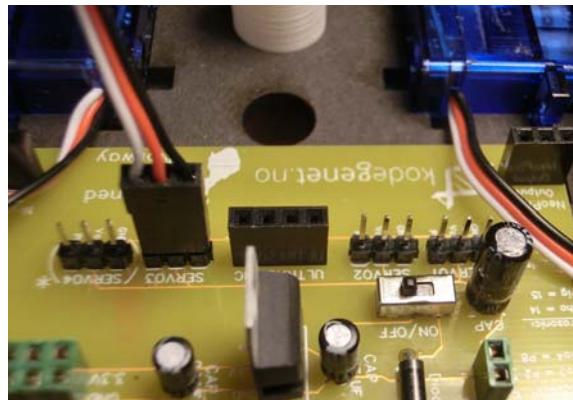
Fordelen med dette programmet er at det er enkelt og det fungerer faktisk ganske godt når man legger senderen i hånda. Ev. avvik vil automatisk korrigeres i bevegelsen av hånda på bakgrunn av observasjoner av roboten. Likevel er det stort potensial for å forbedre koden.

### 8.7.4 Program for styring av penna

Vi ønsker å kunne løfte og senke penna ved å trykke knappene A og B på Micro:biten. Vi ønsker at også denne funksjonen skal være fjernstyrt og på sikt integrert med styring av robotens framdrift.

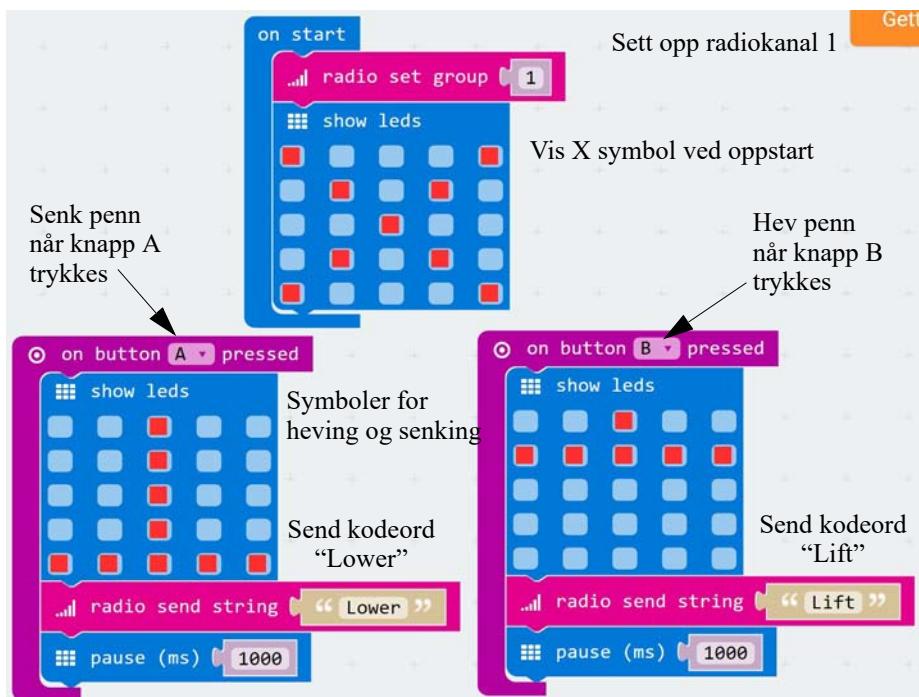
Figuren til høyre et enkelt testprogram som installeres på Micro:biten som er montert på roboten for å teste servoene som kobles til Servo 3 som igjen styres av port P2.





## Penn kontroll – sender

Vi bruker samme teknikk her som for styre roboten. Siden vi ikke trenger å overføre noen parameter velger vi å bruke en enklere kommunikasjonskommando som kun overfører ordene “Lower” for å senke pennen, og “Lift” for å heve pennen. I tillegg vises et symbol på skjermen som indikerer om pennen heves eller senkes.

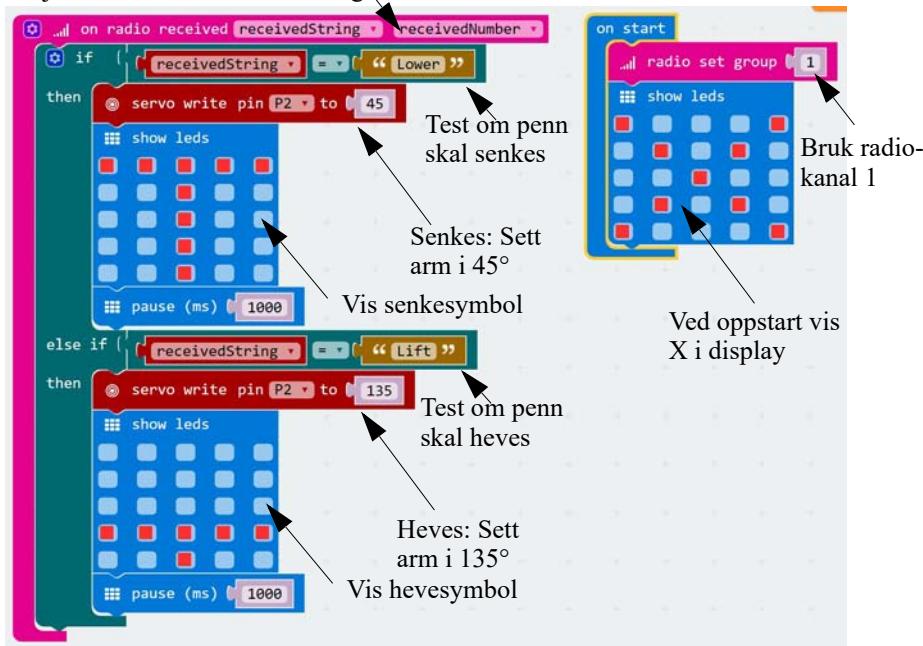




## Penn kontroll – mottaker

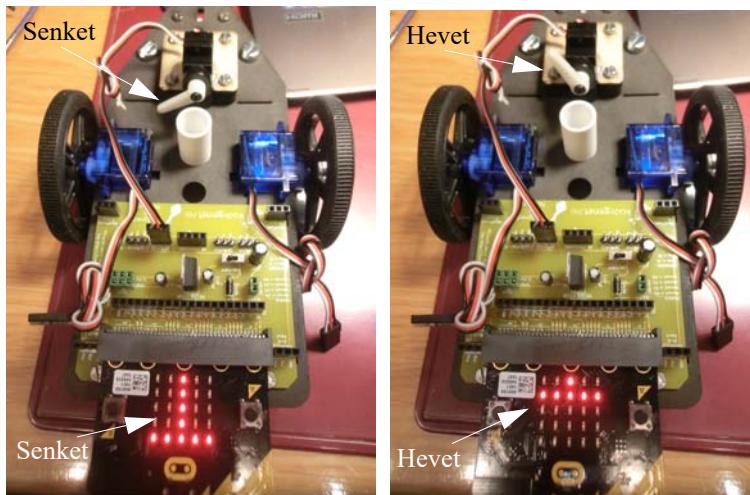
Også i mottakeren benyttes samme metode som vi brukte for å styre roboten.

Kjør rutine ved mottatt melding



Avhengig av hvilket kommandoord som overføres, løftes eller senkes armen til servoen. Legg merke til at symbolene som viser stillingen til pennen er tegnet opp-ned slik at det skal se fornuftig ut når roboten betraktes bakfra.

Bildene under viser hvordan servoen oppfører seg og hvordan displayet illustrerer det som skjer.

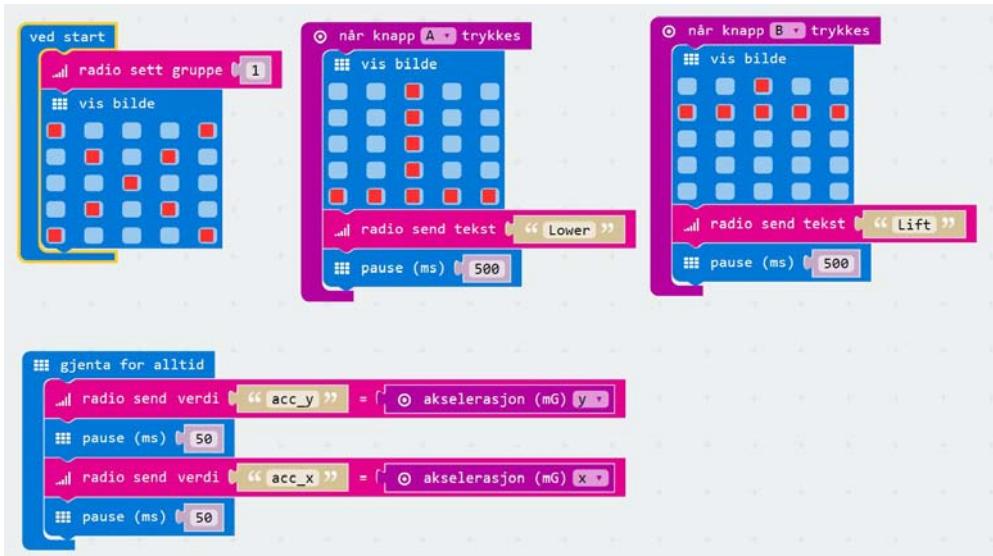




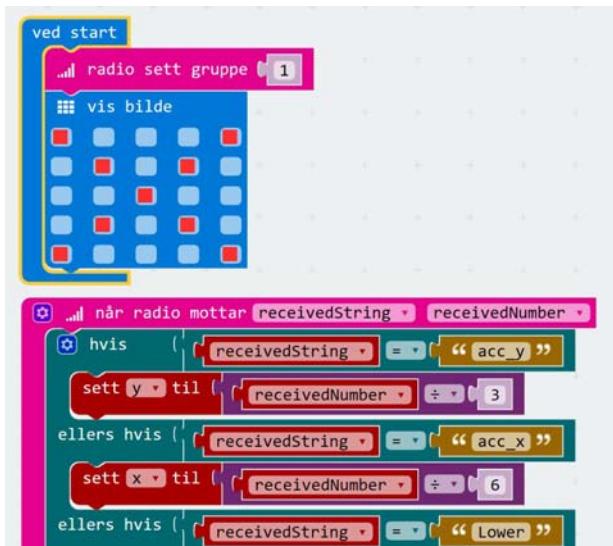
## 8.7.5 Kombinert program med kjørekontroll og penn

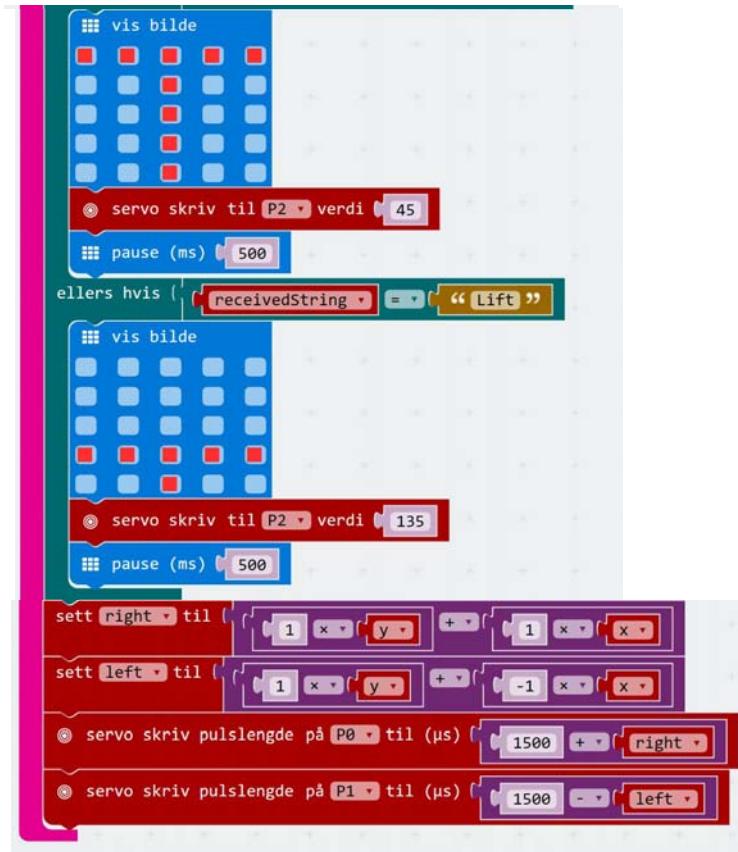
Vi skal nå kombinere styring av robotens bevegelser og løfting og senking av pennen. Dette er ikke vanskeligere enn at vi kombinerer de to programmene i ett.

### Kombinert robotprogram med kjørekontroll og styring av penn – sender



### Kombinert robotprogram med kjørekontroll og styring av penn – mottaker







# Vedlegg A Komponentliste

## A.1 Komponent og innkjøpsliste

Komponentene kan kjøpes fra mange forskjellige forhandlere. Liste under gir en mulig løsning:

|  |                                   |
|--|-----------------------------------|
| 2 stk. Hjul for FS90R 60 x 8 mm                        | Kitronik (pris 2.30 £ pr. par)    |
| 1 stk. Servo:bot-kretskort for Micro:bit (må monteres) | Kodegenet (pris kr. 75,- pr. stk) |
| 1 stk. Stiftlist header                                | Kultogbillig.no (pris ca. 1 kr.)  |
| 1 stk. Hylselist socket                                | Kultogbillig.no (pris ca. 1 kr.)  |
| 2. stk. 360° servo FS90R                               | Kitronik (pris 7.70 £ pr. par)    |
| 1. stk. Mini prototyp bread board                      | Kitronik (pris 1.89 £ pr. stk.)   |
| 2. stk. PP3 batteri klips                              | Kitronik (pris 0.52 £ pr. par)    |
| 2. stk. 2xAA Batteriholder                             | Kitronik (pris 0,66 £ pr. par)    |
| 6 stk. Sporskruer M3x20/16                             | Clas Ohlson 11-1124-320           |
| 6 stk. muttere M3                                      | Clas Ohlson 11-503-3              |
| 4 stk. AA batterier                                    | Biltema (pris 14,00 kr. pr. bil)  |
| 1 stk. Glidelås poser for pakking                      | Biltema (pris 26.90 kr. 10 stk)   |
| 1 stk. Klinkekule                                      | BR-Leker                          |
| 1 stk. chassis 3 - 6 mm MDF/bjørkefiner                | ViT i Trondheim                   |
| 1 stk. Nesehjulholder - PLA                            | ViT i Trondheim                   |

Alternativer:

|   |                                    |
|---|------------------------------------|
| 1 stk. Kretskort m/kantkontakt for Micro:bit (må monteres)  | Kitronik (pris 2.10 £ pr. stk)     |
| 1 stk. Kretskort m/kantkontakt for Micro:bit (montert)<br><a href="https://www.kitronik.co.uk/5601b-edge-connector-breakout-board-for-bbc-microbit-pre-built.html">https://www.kitronik.co.uk/5601b-edge-connector-breakout-board-for-bbc-microbit-pre-built.html</a> | Kitronik (pris 4.16 £ pr. stk)     |
| 1 stk. Servo:bot, kretskort for Micro:bit (ferdig loddet)   | Kodegenet (pris kr. 139,- pr. stk) |

Leverandører:

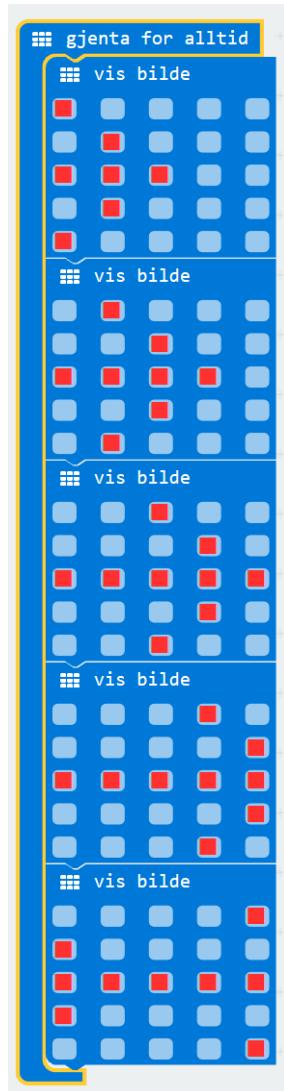
|  |   |
|--|---|
| Kodegenet (Norsk)                            | <a href="https://kodegenet.no/shop/category/microbit">https://kodegenet.no/shop/category/microbit</a> |
| Kult og Billig (Norsk)                       | <a href="http://kultogbillig.no/Elektronikk">http://kultogbillig.no/Elektronikk</a>                   |
| Kitronik (Engelsk)                           | <a href="https://www.kitronik.co.uk/microbit.html">https://www.kitronik.co.uk/microbit.html</a>       |
| Polulu Robotics and Electronics (Amerikansk) | <a href="https://www.pololu.com/">https://www.pololu.com/</a>   |



## Vedlegg B Løsningsforslag øvingsoppgaver

### B.1 Løsningsforslag øving 1 – Lag en animasjon

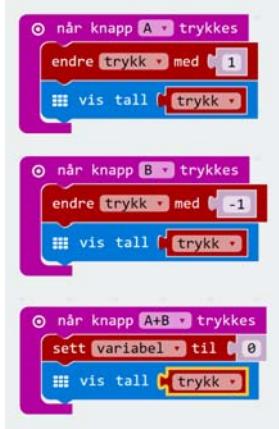
Hensikten med øvingen er å erfare hvor lav terskelen for å få til et enkelt program.





## B.2 Løsningsforslag øving 2 – Bruk av variable

Hensikten med øvingen er å bli kjent med bruken av variable



## B.3 Løsningsforslag øving 3 – Bruk av akselerometer

### B.3.1 Løsningsforslag til øving 3A

Hensikten med øvingen er å vise hvor enkelt det er å bruke knappene som inngangsparametere for aksjoner.



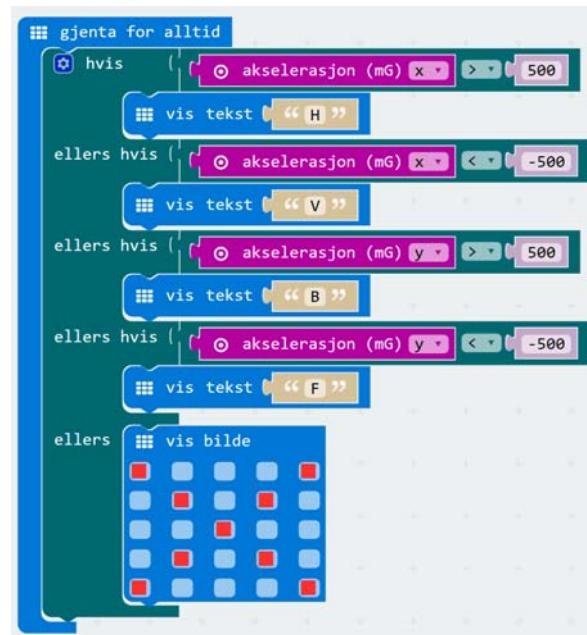
### B.3.2 Løsningsforslag til øving 3B

Her finnes det sannsynligvis flere løsninger, men bruk av akselerometeret er en.



### B.3.3 Løsningsforslag til øving 3C

Hensikten med øvingen er å lære å bruke if-setning, operatorene  $>$  og  $<$ , og å hente in data fra akselerometeret. Dette krever også at en forstår hvordan akselerometeret opererer.



### B.4 Øving 4 Overføring av informasjon via radio

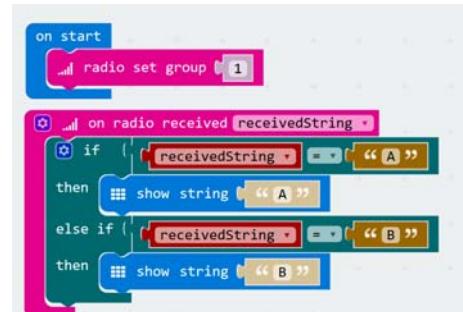
#### B.4.1 Løsningsforslag øving 4 – Overføring av en aksjon

Hensikten med øvingen er å vise hvordan informasjon kan overføres via en trådløs radioforbindelse.

Senderdel



Mottakerdel





## B.5 Løsningsforslag skritteller

Hensikten med denne øvelsen er å kunne forstå hvordan akselerometeret fungerer og kan brukes til å bestemme hellingsvinkelen til Micro:biten, videre å lære å bruke tellevariabler.

Det er ikke vedlagt noe løsningsforslag.

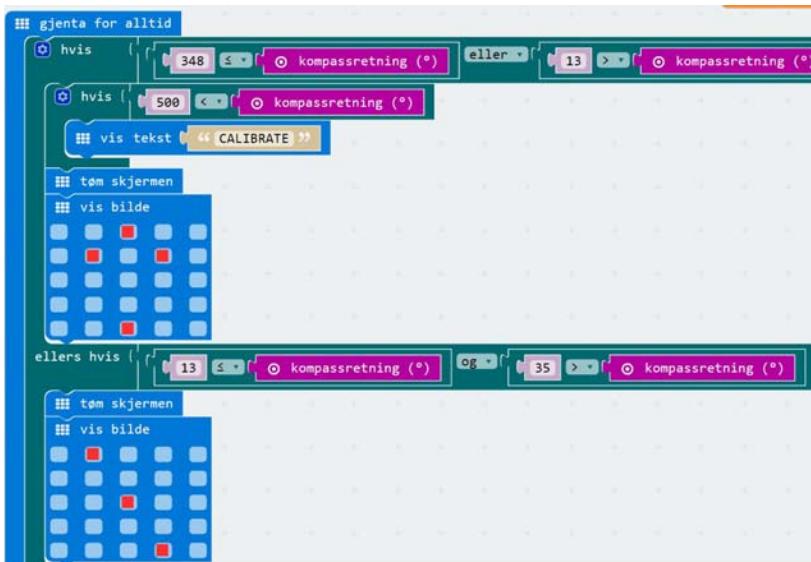
## B.6 Løsningsforslag lysteremin

Hensikten med øvelsen er å lære hvordan man kan lese av lysstyrken og konvertere denne verdien til en tone.



## B.7 Løsningsforslag til oppgave 7 – Lag kompass som peker mot nord

Hensikten er å lære å benytte magnetometeret som kompass, og å få erfaring med hvor pålitelig kompassfunksjonen er.



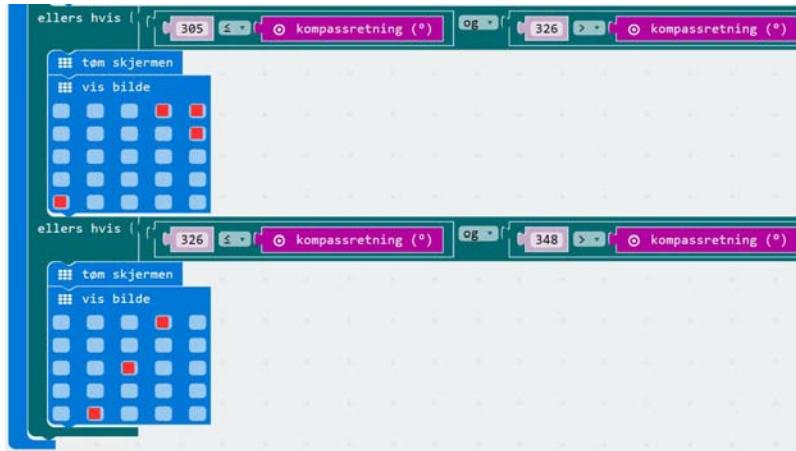




The image shows a Scratch script with four identical 'ellers hvis' (else if) blocks stacked vertically. Each block has a condition 'kompassretning (<>)' followed by a value (168, 193, 215, or 236). The 'ellers hvis' blocks are connected by an 'og' (and) connector. Each block contains the following script:

```
ellers hvis [kompassretning (<) 168 og kompassretning (<) 193]
    [tøm skjermen viss bildet ikke er sett til øverst]
    [vis bildet]
    [grid1 viss]
ellers hvis [kompassretning (<) 193 og kompassretning (<) 215]
    [tøm skjermen viss bildet ikke er sett til øverst]
    [vis bildet]
    [grid1 viss]
ellers hvis [kompassretning (<) 215 og kompassretning (<) 236]
    [tøm skjermen viss bildet ikke er sett til øverst]
    [vis bildet]
    [grid1 viss]
ellers hvis [kompassretning (<) 236 og kompassretning (<) 258]
    [tøm skjermen viss bildet ikke er sett til øverst]
    [vis bildet]
    [grid1 viss]
ellers hvis [kompassretning (<) 258 og kompassretning (<) 283]
    [tøm skjermen viss bildet ikke er sett til øverst]
    [vis bildet]
    [grid1 viss]
ellers hvis [kompassretning (<) 283 og kompassretning (<) 305]
    [tøm skjermen viss bildet ikke er sett til øverst]
    [vis bildet]
    [grid1 viss]
```

The 'grid1 viss' block contains a 4x4 grid of squares, with red squares at positions (1,1), (1,3), (2,2), (2,4), (3,1), (3,3), (4,2), and (4,4).



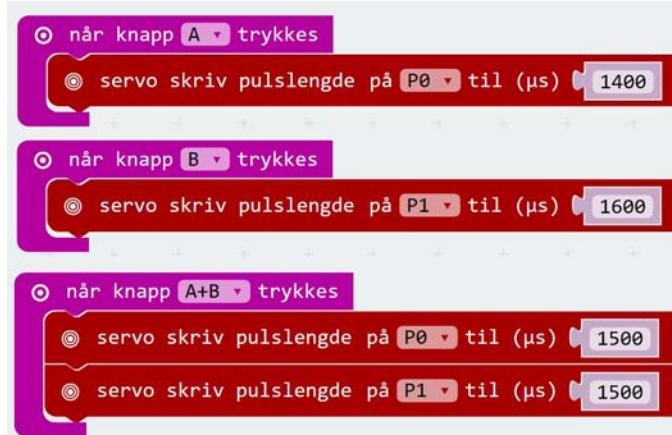
## B.8 Løsningsforslag til oppgave 8 – Når Micro:bit ”forstår” hva som blir sagt

Hensikten med denne oppgaven er å tenke kreativt for å kunne løse oppgaven. Det finnes sannsynligvis flere løsninger på oppgaven. Foreløpig er det derfor ikke foreslått noen løsning.

## B.9 Styring av 360° servo

### B.9.1 Løsningsforslag oppgave 9 A

Hensikten med oppgaven er dels å få erfaring med styring av 360° servoer ved hjelp av knapp A og B.





## B.9.2 Løsningsforslag oppgave 9 B 360° Servo – Styring med knapper

Hensikten med oppgaven er dels å få erfaring med styring av 360° servoer og dels med å bruke samme bryter til flere funksjoner eller i dette tilfellet “toggling”.

The Scratch script consists of four main sections:

- ved start:** Sets flag 1 = 0 and flag 2 = 0.
- når knapp A trykkes:** Checks if flag 1 = 0. If true, it sets servo P0 to 1600 µs and flag 1 = 1. If false, it sets servo P0 to 1400 µs and flag 1 = 0.
- når knapp B trykkes:** Checks if flag 2 = 0. If true, it sets servo P1 to 1600 µs and flag 2 = 1. If false, it sets servo P1 to 1400 µs and flag 2 = 0.
- når knapp A+B trykkes:** Sets both servo P0 and P1 to 1500 µs.

## B.10 Løsningsforslag til Turbidimeter

### B.10.1 Løsningsforslag enkelt turbidimeter uten midling

Enkelt turbidimeter uten midling

The Scratch script consists of a loop:

- Initializes digital pin P0 to 1.
- Pauses for 500 ms.
- Reads analog value from pin P1 and stores it in variable **Avlest\_lys**.
- Displays the value stored in **Avlest\_lys**.
- Initializes digital pin P0 to 0.
- Pauses for 500 ms.



## B.10.2 Løsningsforslag enkelt turbidimeter med midling

Turbidimeter med midling

The Scratch script consists of the following blocks:

- A blue **ved start** (when green flag is shown) hat block.
- A blue **gjenta for alltid** (repeat forever) control block.
- Inside the loop:
  - A red **skriv digital til P0 verdi 1** (write digital to P0 value 1) control block.
  - A blue **pause (ms) 500** (wait (ms) 500) control block.
  - A green **gjenta 100 ganger** (repeat (100) [ ]) control block.
  - Inside the green loop:
    - A red **endre Avlest\_lys med les analogverdi fra P1** (change Avlest\_lys by read analog value from P1) control block.
    - A red **sett Avlest\_lys til Avlest\_lys ÷ 100** (set Avlest\_lys to Avlest\_lys ÷ 100) control block.
    - A blue **vis tall Avlest\_lys** (show number Avlest\_lys) control block.
  - A red **skriv digital til P0 verdi 0** (write digital to P0 value 0) control block.
  - A blue **pause (ms) 500** (wait (ms) 500) control block.
  - A red **sett Avlest\_lys til 0** (set Avlest\_lys to 0) control block.



### B.10.3 Løsningsforslag for turbidimeter med midling og bruk av variabel

Turbidimeter med midling og bruk av variabel

```
ved start
  sett Antall til 100
gjenta for alltid
  skriv digital til P0 verdi 1
  pause (ms) 500
  gjenta Antall ganger
    endre Avlest_lys med les analogverdi fra P1
    sett Avlest_lys til Avlest_lys ÷ Antall
    vis tall Avlest_lys
  skriv digital til P0 verdi 0
  pause (ms) 500
  sett Avlest_lys til 0
```



## B.10.4 Løsningsforslag for turbidimeter som skriver ut antall dråper

Turbidimeter som skriver ut antall dråper

```
ved start
  sett Antall til 100
gjenta for alltid
  skriv digital til P0 verdi 1
  pause (ms) 500
  gjenta Antall ganger
    endre Avlest_lys med les analogverdi fra P1
    sett Avlest_lys til Avlest_lys + Antall
    hvis Avlest_lys < 100 så
      vis tall 1
    ellers hvis Antall ≥ 100 og Antall < 200 så
      vis tall 2
    ellers hvis Antall ≥ 200 og Antall < 300 så
      vis tall 3
    ellers hvis Antall ≥ 300 og Antall < 400 så
      vis tall 4
    ellers hvis Antall ≥ 400 og Antall < 500 så
      vis tall 5
    ellers hvis Antall ≥ 500 og Antall < 600 så
      vis tall 6
    ellers hvis Antall ≥ 600 og Antall < 700 så
      vis tall 7
    ellers hvis Antall ≥ 700 og Antall < 800 så
      vis tall 8
    ellers hvis Antall ≥ 800 og Antall < 900 så
      vis tall 9
    ellers
      vis tall 0
  skriv digital til P0 verdi 0
  pause (ms) 500
  sett Avlest_lys til 0
```



---

## Vedlegg C OpenSCAD kode

### C.1 Holder for nesehjul

Koden under viser koden som skriver ut holderen for nesehjulet.

```
difference ()  
{  
cylinder($fn = 4, $fa = 3, $fs = 2, h = 20, r1 = 10, r2 = 10, center =  
false);  
translate ([0,0,17]) sphere ($fn = 100, 7.9);  
translate ([0,0,18]) cylinder($fn = 40, $fs = 2, h = 4, r1 = 7.5, r2 =  
7.5, center = false);  
}  
difference ()  
{  
translate ([-17.0,-14.5,0]) cube(size = [34, 29, 2], center = false);  
translate ([-14,-11.5,-1]) cylinder($fn = 10, h = 7, r1 = 2.0, r2 = 2.0,  
center = false);  
translate ([-14,11.5,-1]) cylinder($fn = 10, h = 7, r1 = 2.0, r2 = 2.0,  
center = false);  
translate ([14,-11.5,-1]) cylinder($fn = 10, h = 7, r1 = 2.0, r2 = 2.0,  
center = false);  
translate ([14,11.5,-1]) cylinder($fn = 10, h = 7, r1 = 2.0, r2 = 2.0,  
center = false);  
}  
  
translate ([0,10,0]) cylinder($fn = 10, h = 20, r1 = 1.5, r2 = 1.5, center  
= false);  
translate ([10,0,0]) cylinder($fn = 10, h = 20, r1 = 1.5, r2 = 1.5, center  
= false);  
translate ([0,-10,0]) cylinder($fn = 10, h = 20, r1 = 1.5, r2 = 1.5, cen-  
ter = false);  
translate ([-10,0,0]) cylinder($fn = 10, h = 20, r1 = 1.5, r2 = 1.5, cen-  
ter = false);  
// Ønsker man å inkludere kula legger man til følgende setning  
color("white") translate ([0,0,17]) sphere ($fn = 200, 7.8);
```

### C.2 Deler for løfting av penn

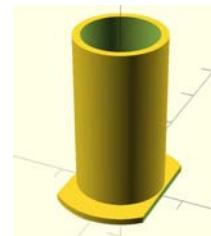
I dette vedlegget presenteres deler til pennholderen.



### C.2.1 Pennholder ytre rør med flens

Program for 3D-printing av ytre rør med flens:

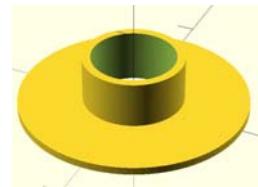
```
difference()
{
    union()
    {
        cylinder($fn = 100, h = 25, r1 = 6.6, r2 = 6.6, center = false);
        cylinder($fn = 100, h = 1, r1 = 10, r2 = 10, center = false);
    }
    translate([0, 0, -1]) cylinder($fn = 100, h = 27, r1 = 5.5, r2 = 5.5,
center = false);
    translate([21, 0, -1]) rotate ([0, 0, 45]) cylinder($fn = 4, h = 3, r1
= 20, r2 = 20, center = false);
    translate([-21, 0, -1]) rotate ([0, 0, 45]) cylinder($fn = 4, h = 3, r1
= 20, r2 = 20, center = false);
}
```



### C.2.2 Pennholder muffle med flens

Program for 3D-printing av muffle med flens som tres ned over pennen:

```
difference()
{
    union()
    {
        cylinder($fn = 100, h = 7, r1 = 6.6, r2 = 6.6, center = false);
        cylinder($fn = 100, h = 1, r1 = 17, r2 = 17, center = false);
    }
    translate([0, 0, -1]) cylinder($fn = 100, h = 9, r1 = 5.5, r2 = 5.5,
center = false);
}
```





---

## Vedlegg D Læreplaner

### D.1 Teknologi og Forskningslære 1

Fargekodene forsøker å antyde hva som *lett kan* knyttes til bygging av en robot med Micro:bit og programmering (rødt/kursiv), hva som er litt på siden (blått) og det som vanskelig lar seg oppfylle innen prosjektet (sort).

#### Den unge ingenieren - Beskrivelse av hovedområde

Mål for opplæringen er at eleven skal kunne:

- *planlegge og bygge en konstruksjon som er fast eller bevegelig, og som har en definert funksjon*
- bruke tredimensjonale tegninger eller skisser i utvikling av konstruksjoner
- bruke forskjellige materialer og former for sammenføyninger og begrunne valg av materialer og byggemåte ut fra materialenes egenskaper og konstruksjonens funksjon
- *bruke sensorer og styringssystemer i forbindelse med forsøk og konstruksjoner*
- *dokumentere og vurdere konstruksjoners fysiske egenskaper og funksjonalitet ved hjelp av målinger og enkle beregninger*

#### Den unge forskeren - Beskrivelse av hovedområde

Mål for opplæringen er at eleven skal kunne:

- *gjøre rede for hvordan et naturvitenskapelig prosjekt planlegges, gjennomføres og etterarbeides før det blir publisert*
- *planlegge, gjennomføre, analysere og dokumentere systematiske målinger*
- om støy, luftforurensning, inneklima og vannkvalitet, og drøfte virkninger på helse og miljø

#### Teknologi, naturvitenskap og samfunn - Beskrivelse av hovedområde

Mål for opplæringen er at eleven skal kunne

- *drøfte etiske, miljømessige, kulturelle og politiske sider ved teknologisk utvikling (bruk av robotteknologi)*
- *beskrive den historiske utviklingen av en teknologisk innretning, forklare virkemåten og drøfte anvendelser i samfunnet*
- gjøre rede for utvikling og produksjon av et teknologisk produkt og vurdere produktets brukervennlighet, utviklingsmuligheter og miljøpåvirkning
- *beskrive prinsipper og virkemåte for noen moderne instrumenter i industri, helsevesen eller forskning, og gjøre rede for nytten og eventuelle skadevirkninger*
- kartlegge og presentere praktisk bruk av realfag i en lokal bedrift eller institusjon



---

## Design og produktutvikling - Beskrivelse av hovedområde

Mål for opplæringen er at eleven skal kunne

- *gjøre rede for funksjonen til vanlige komponenter i elektroniske kretser, og gjenkjenne komponentene i en krets*
- *lage elektroniske kretser ved å lodde komponenter og simulere og teste kretsene*
- *forme og utvikle produkter som har en definert funksjon og inneholder elektronikk*
- *dokumentere og presentere designprosesser fra idé til ferdig produkt*
- *begrunne valg av materialer i produkter og vurdere produktenes form og funksjon, miljømessige konsekvenser, estetikk og forbedringsmuligheter*
- *utføre målinger med eller teste et eget produkt, og vurdere kvaliteten på produktet med tanke på funksjonalitet*

## D.2 Teknologi og forskningslære 2

### Den unge forskeren - Beskrivelse av hovedområde

Mål for opplæringen er at eleven skal kunne

- gjøre rede for et forskningsprosjekt i en bedrift eller institusjon, og beskrive problemstillinger, organisering, måleutstyr, resultater og finansiering
- *planlegge og gjennomføre naturvitenskapelige undersøkelser basert på egne ideer, og presentere arbeidet i en vitenskapelig form*
- *drøfte resultater fra egne undersøkelser i forhold til relevant kunnskap på området, og vurdere hvordan kontroll av variabler og reproducertbarhet er ivaretatt*

### Naturvitenskapelige arbeidsmetoder - Beskrivelse av hovedområde

Mål for opplæringen er at eleven skal kunne

- *forklare hva som menes med modell, teori og hypotese, og gjøre rede for hvordan de brukes og utvikles i forskning*
- drøfte ved å bruke eksempler hvordan empiriske data kan styrke eller forkaste en hypotese
- *gjøre rede for hvordan forskning utvikles og kvalitetssikres gjennom samarbeid, kritisk vurdering og argumentasjon*
- *gjøre rede for strukturen i en vitenskapelig publikasjon eller presentasjon*

### Forskning, teknologi og samfunn - Beskrivelse av hovedområde

Mål for opplæringen er at eleven skal kunne

- beskrive kjennetegn ved grunnforskning, anvendt forskning og utviklingsarbeid og gjøre rede for hovedtrekk ved finansiering og styring



- 
- gjøre rede for betydningen av naturvitenskapelig forskning og teknologiutvikling for næringsliv og samfunn
  - drøfte økonomiske, miljømessige og etiske spørsmål i forbindelse med naturvitenskapelig forskning og teknologiutvikling
  - *drøfte og gi eksempler på hvordan forskningsresultater og ny teknologi formidles og brukes av forskningsinstitusjoner, medier, bedrifter, interessegrupper og myndigheter*

## **Vitenskapsfilosofi og vitenskapsteori - Beskrivelse av hovedområde**

Mål for opplæringen er at eleven skal kunne

- beskrive hovedtrekk i den historiske utviklingen av vitenskapelige tenkemåter og drøfte teknologiens rolle i denne utviklingen
- gjøre rede for hovedideene til noen sentrale vitenskapsteoretikere og vitenskapsfilosofer vurdere hvordan argumentasjon i aktuelle naturvitenskapelige debatter bygger på empiriske resultater, teoretisk kunnskap og ideologisk ståsted

## **D.3 Teknologi i Praksis (TiP) – ungdomsskolen**

### **Formål**

Valfaga skal medverke til at elevane, kvar for seg og i fellesskap, styrker lysta til å lære og opplever meistring gjennom praktisk og variert arbeid. Valfaga er tverrfaglege og skal medverke til heilskap og samanheng i opplæringa.

Teknologi handlar om den menneskeskapte verda og om innretningar og system som kan gjere kvardagen betre. Opp gjennom tidene har menneska brukt kreativitet og skaparevner til å utvikle reiskapar, maskinar og andre teknologiske produkt og løysingar. Teknologien grip inn på mange område, og har gjeve og kan gje både moglegheiter og utfordringar, både for den enkilde og for samfunnet. Innanfor teknologien finn vi dei enklaste verktøy og produkt og den mest avanserte elektronikken. Erfaring med og innsikt i teknologi kan fremje personleg utvikling, demokratisk deltaking og medverke til eit aktivt forhold til ein teknologisk kvardag.

Valfaget teknologi i praksis skal motivere elevane til å utvikle teknologiske produkt med utgangspunkt i lokale behov og problemstillingar. Prosessen frå idé til eit ferdig produkt kan medverke til skaparglede og meistringsoppleving. Gjennom eige arbeid og i samarbeid med andre kan elevane utvikle ferdigheiter og innsikt. Det inneber å prøve ut eigne talent og moglegheiter på ulike steg i prosessen, vurdere prosessar og produkt og få tilbakemeldingar frå andre.

Valfaget handlar om å planleggje, konstruere og framstille gjenstandar og produkt med varierte materiale og teknologiske løysingar. Kunnskap om teknologiske produkt som blir brukte i dagleidivet, gjev eit godt grunnlag for å forbetre produkt og utvikle nye produkt.

Valfaget hentar hovudelement frå matematikk, naturfag og kunst og handverk/duodji. Element frå norsk/samisk, RLE og samfunnsfag kan også inngå.



---

## Hovedområder

### *Undersøkingar*

Hovedområdet handlar om korleis teknologiske produkt er konstruerte og verkar, kva for prosesser som inngår i utvikling og bruk, og kva for behov produkta dekkjer. Utvikling, konstruksjon og produksjon av teknologi inngår i hovudområdet, i tillegg til helse, miljø og sikkerheit (HMS). Kunnskap om korleis teknologien byggjer på nokre grunnleggjande prinsipp, og korleis ny teknologi byggjer på tidlegare erfaringar, høyrer også med til hovudområdet.

### *Idéutvikling og produksjon*

*Hovudområdet omfattar planlegging, framstilling og utprøving av eigne produkt og konstruksjonar. Planar for framstilling og utprøving av eigne produkt og konstruksjonar byggjer på kravspesifikasjon.*

*I utviklingsfasen er kjennskap til design og verkemåte til andre produkt viktig. Diskusjon omkring ulike sider ved produkta er viktig i alle fasar av produktutviklinga og kan også medverke til å forbete prosessar og produkt.*

## Kompetansemål

### *Undersøkingar*

- undersøke teknologiske produkt og dei vala som er gjorde med omsyn til bruk, tekniske løysingar, funksjonalitet og design
- demonstrere riktig bruk av utvalde verktøy*
- vurdere teknologiske produkt ut frå brukartilpassing, HMS-krev og miljøtilpassing

### *Idéutvikling og produksjon*

- utvikle ein realistisk kravspesifikasjon for eit teknologisk produkt og beskrive kva behov produktet skal dekke*
- framstille produktet med eigna materiale, komponentar, og funksjonelle teknologiske løysingar*
- bruke kunnskap om andre produkt i arbeidet med eige produkt*
- teste eigne produkt og foreslå moglege forbetringar*

## D.4        Programmering – ungdomsskolen

### **Formål**

Valgfagene skal bidra til at elevene, hver for seg og i fellesskap, styrker lysten til å lære og opplever mestring gjennom praktisk og variert arbeid. Valgfagene er tverrfaglige og skal bidra til helhet og sammenheng i opplæringen.

Vi lever i en teknologirik verden hvor de fleste forholder seg til en rekke digitale enheter daglig. Bruken av digital teknologi øker i alle samfunnsmråder, innan alt fra samferdsel til helse, og stiller nye krav til digitale ferdigheter. Utviklingen av det teknologiske samfunnet vi har i dag og i



fremtiden, utfordrer både måten vi lærer på og hvilke kompetanser som blir viktige. Uavhengig av yrkesvalg vil en grunnleggende forståelse av teknologiens oppbygning og virkemåter være en nødvendig del av fremtidens kompetanse.

Programmering er en viktig ferdighet i dagens samfunn, som inngår i de fleste fagområder, fra digital musikk til naturvitenskap og matematikk. Programmering åpner for å utforske komplekse og realistiske modeller av virkeligheten. Det gir også utvidede muligheter til å behandle store datamengder. Å gi elever grunnleggende ferdigheter i programmering er med på å forberede dem for fremtidige realfaglige jobber, i tillegg til å øke forståelsen for naturvitenskapelige og matematiske problemer.

**Valgfaget programmering handler om å lage programkode, det vil si et sett med regler og uttrykk for å styre digitale enheter. I dette inngår prosessen fra å identifisere problemer og utforme mulige løsninger, til å lage kode som kan forstås av en datamaskin, systematisk feilsøke og forbedre denne koden, og dokumentere løsningen på en forståelig måte. Det omfatter alle nivåer fra å forutse og analysere hva et program skal gjøre, til å kjenne igjen mønstre, eksperimentere og evaluere mulige løsninger, og samarbeide med andre. Summen av disse ferdighetene kalles algoritmisk tankegang.**

Opplæringen skal legge til rette for at elevene lærer å løse problemer på nye måter. Elevene skal få muligheten til å utvikle sin kreativitet og skape produkter ved hjelp av programmering. Gjennom å lage programmer oppører elevene også ferdigheter i å vurdere eget og andres arbeid, gi konstruktive tilbakemeldinger og samarbeide med andre. Å skape og produsere digitalt, krever forståelse og kompetanse i programmering.

## Hovedområder

Valgfaget er strukturert i to hovedområder. Hovedområdene utfyller hverandre og må ses i sammenheng.

Oversikt over hovedområder:

### *Modellering*

Hovedområdet tar for seg stegene som kreves for å løse problemer ved hjelp av programmering, også kjent som algoritmisk tankegang. Til hovedområdet hører kunnskap om hva slags problemer som egner seg for å løses av en datamaskin, hvordan disse kan brytes ned i delproblemer, og hvordan løsninger kan utformes. Modellering av matematiske og naturvitenskapelige fenomener er en sentral del av dette. I hovedområdet inngår hvordan datamaskiner og programmer er konstruert og virker, ulike programmeringsspråk, og styrker og svakheter ved de ulike språkene. Prinsipper som ligger til grunn for god programmeringspraksis inngår også i hovedområdet, deriblant forklaring og dokumentasjon av løsninger og programkode.

### *Koding*

Hovedområdet handler om å utvikle egne programmer ved hjelp av ulike programmeringsspråk. I dette inngår å bruke og forstå grunnleggende prinsipper i programmering, slik som løkker, tester, variabler, funksjoner og enkel brukerinteraksjon. Hovedområdet omfatter også kontrolleringer eller simulering av fysiske objekter, så som roboter, sensorer, baller som spretter og molekylers beve-



---

gelse. Videre dreier det seg om simuleringer og beregninger basert på matematiske og naturfaglige problemstillinger. Hovedområdet omfatter også feilsøking, generalisering og gjenbruk av løsninger, inkludert vurdering og analyse av egen og andres programkode.

## Kompetanse mål

### *Modellering*

Mål for opplæringen er at eleven skal kunne

- gjøre rede for hvordan datamaskiner og programmer fungerer, inkludert et utvalg utbredte programmeringsspråk og deres bruksområder
- omgjøre problemer til konkrete delproblemer, vurdere hvilke delproblemer som lar seg løse digitalt, og utforme løsninger for disse
- dokumentere og forklare programkode gjennom å skrive hensiktsmessige kommentarer og ved å presentere egen og andres kode

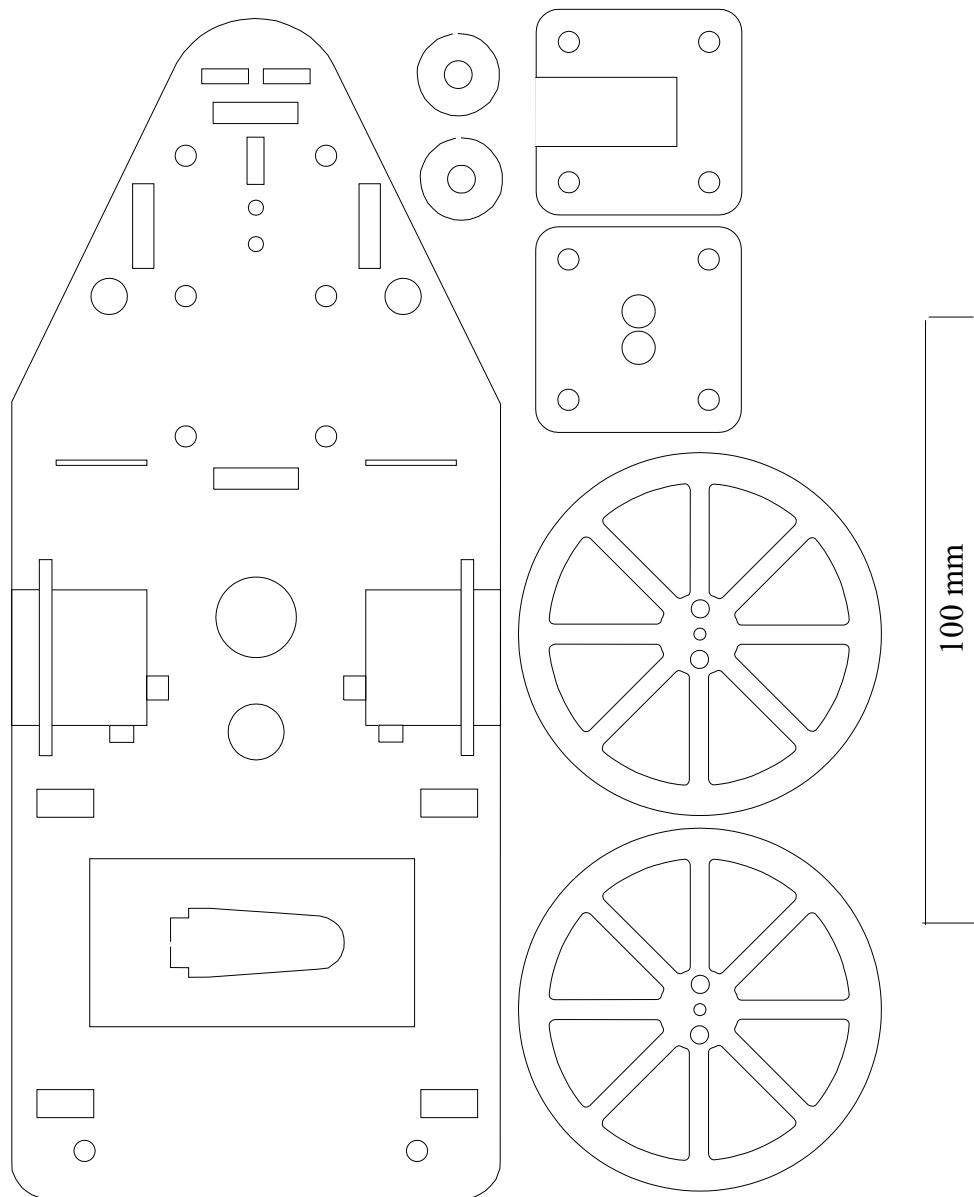
### *Koding*

Mål for opplæringen er at eleven skal kunne

- bruke flere programmeringsspråk der minst ett er tekstbasert
- bruke grunnleggende prinsipper i programmering, slik som løkker, tester, variabler, funksjoner og enkel brukerinteraksjon
- utvikle og feilsøke programmer som løser definerte problemer, inkludert realfaglige problemstillinger og kontrollering eller simulering av fysiske objekter
- overføre løsninger til nye problemer ved å generalisere og tilpasse eksisterende programkode og algoritmer.

## Vedlegg E Filer for laserkutter

### E.1 Chassie for robot

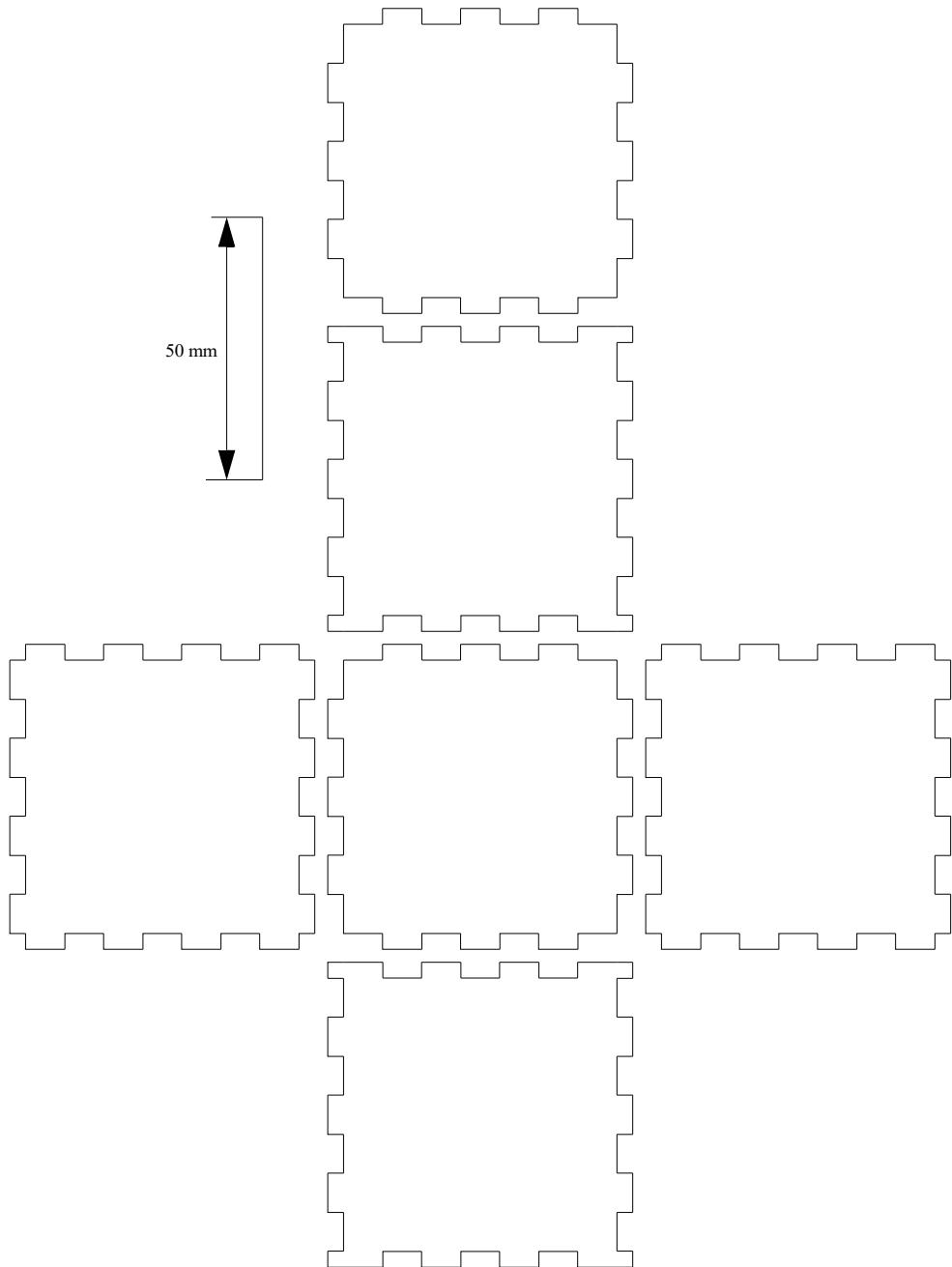


**Viktig!!!** Juster størrelsen slik at målet til venstre blir akkurat 100 mm



## E.2 Magisk terning

Denne tagningen er forminsket pass på at størrelsen blir juster til riktig størrelse.



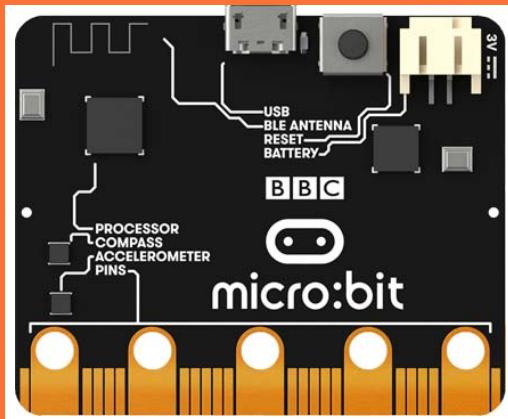












Heftet er ment som et forslag til prosjekter hvor BBCs Micro:bit er benyttet som programmerbar styringsenhet. Det beskrives flere prosjekter, Den magiske terningen og fjernstyrte roboter, deriblant en bit:bot XL og en "hjemmebygget" robot. Det legges spesielt vekt på beskrivelse av og bruk av radiokommunikasjon mellom micro:bits. Heftet er ikke en lærebok i blokkprogrammering av micro:bits, men heller et idehefte til et par prosjekter som kan brukes til å lære seg programmering av Micro:bit. Tanken er at heftet ikke skal være en fasit, men skape ideer for videre arbeid.

**Nils Kr. Rossing**

Prosjektleder ved Vitensenteret i Trondheim

Dosent ved Skolelaboratoriet ved NTNU

e-post: nkr@vitensenteret.com