INSTITUTT FOR ELEKTRONISKE SYSTEMER

NTNU Norges teknisk-naturvitenskapelige universitet

Motorlab

DC-motorer, motordrivere og servoer

IELET1002

DATATEK.

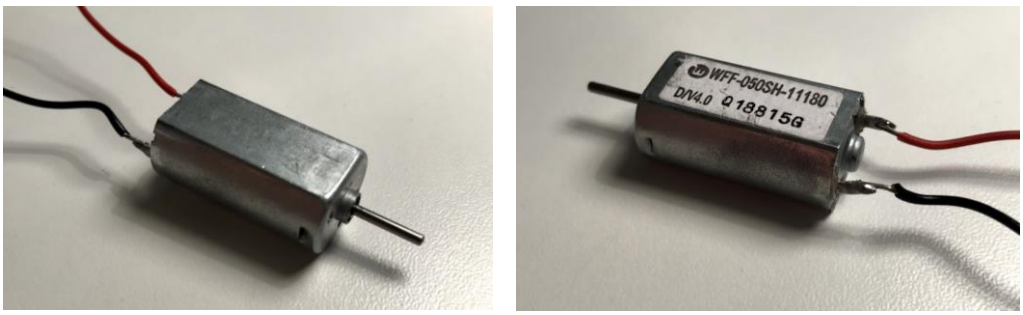
Innholdsfortegnelse

DC-motor og H-bro

Hva er en DC-motor?

En DC-motor er en motor som går på likestrøm, og kan rotere i begge retninger basert på strømretningen gjennom motoren. «DC» i «DC-motor» står for Directional Current, altså likestrøm.

DC-motorens form er som regel syllinderformet, og ofte med to flate sider langs motorene for å kunne feste motoren til rette overflater. I den ene enden av DC-motoren stikker det ut en aksling, stanga som roterer, og i den andre enden stikker det ut to tilkoblingspunkter for spenningsforskyning og jord.



Figur 1: 9V DC-motoren i Elektras Arduinokit

Å kalle et av tilkoblingspunktene for inngang og et for utgang er akseptabelt, men det er vilkårlig hvilket som er hvilket, da DC-motorer er symmetriske på innsiden. Å sende en strøm fra det ene punktet til det andre får akslingen til å rotere en vei, og om man bytter retning på strømmen roterer akslingen den andre veien.

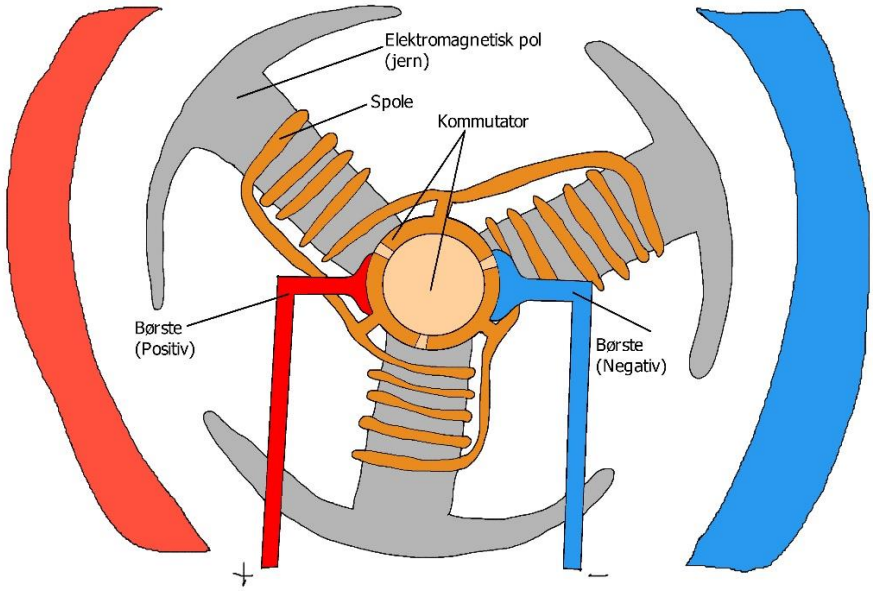
Innsiden av en DC-motor

På innsiden av en DC-motor er det permanente magneter som lager et tilnærmet uniformt magnetfelt, altså et magnetfelt som går i én retning. Alle de andre komponentene befinner seg i dette magnetfeltet.

Har du en magnet tilgjengelig, eksempelvis i et mobildeksel? Prøv å berøre DC-motoren din med den!

Rundt akslingen er det en rotor. Rotoren har to eller flere elektromagnetiske poler som magnetiseres i tur og orden. En kommutator er det som fører til at polene magnetiseres i riktig rekkefølge. Kommutatoren er altså en del av rotoren som er formet slik at ved rotasjon blir riktige poler på rotoren magnetisert.

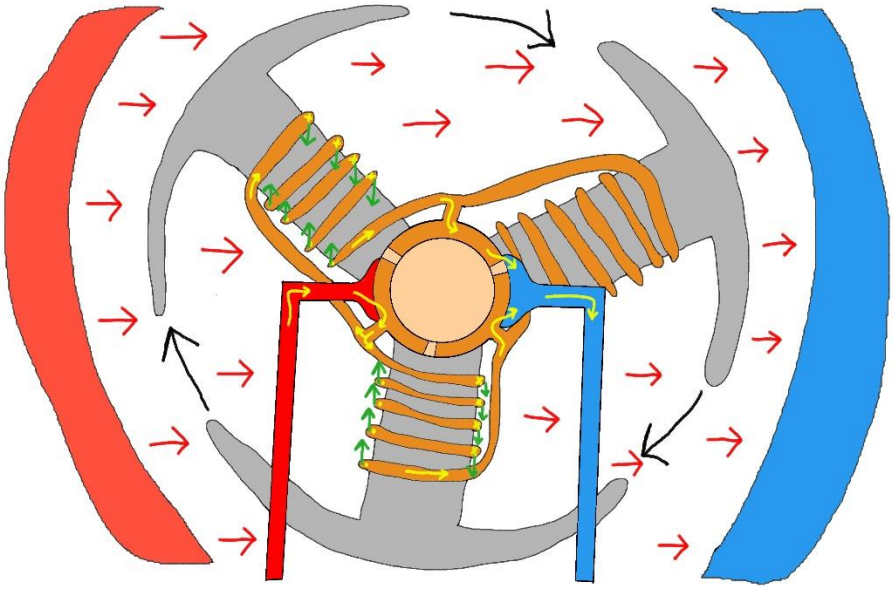
Polene blir magnetisert i en slik tur og orden at de alltid dytter motoren i riktig retning hvor i magnetfeltet de befinner seg. Strømmen blir gitt til kommutatoren gjennom to «børster». Ledninger som har et mykt, ledende materiale – gjerne karbon – i kontakt med kommutatoren.



Figur 2: Simplifert tegning av innsiden på en DC-motor

Dersom man ønsker å forstå DC-motoren er det greit å forestille seg hvordan strømmen går gjennom den. Det er illustrert i figur 3 hvordan strømmen – representert ved gule piler – beveger seg fra den positive børsten, gjennom spolene på polene, og til den negative børsten.

Ved å ha en strøm i den retningen et magnetfeltet – representert ved røde piler – vil spolene dyttes i retning de grønne pilene. De som har hatt fysikk 2 på videregående, og husker høyrehåndsregelnene fra magnetismen, kan teste dette om de vil. Merk på figur 4 at spolene er avlange. En kan også gjøre et liknende resonnement med Lenz' lov, men det tar litt lengre tid å forklare.

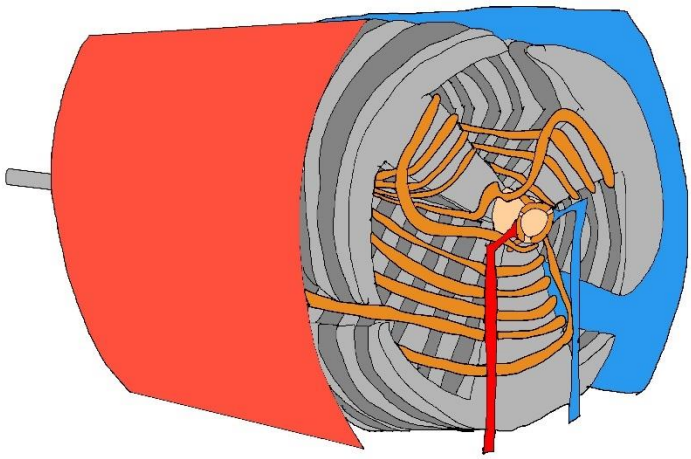


Figur 3: Simplifisert DC-motor med strøm (gul), magnetfelt (rød), kraftvektorer (grønn), og rotasjonsretning (sort)

Polene kan en tenke på som forsterkende til kraften fra spolene. I spolene er det tusenvis av vindinger rundt polene. Det er brukt få vindinger i figurene for å illustrere strømmen på enklere vis.

Dersom dette ble litt for mye fysikk og litt for lite elektronikk, frykt ikke; det eneste du trenger å huske er:

> *Det er en god del (2 eller flere) elektromagneter som skrur seg av og på så to permanente magneter alltid dytter akslingen i riktig retning. Dersom man bytter retning på strømmen byttes retningen på hvilken vei akslingen dyttes.*



Figur 4: Skisse av dybden til figur 2 og 3

Hva skal vi se etter på motorbladene til DC-motorer?

På et datablad til en DC-motor er det et par ting vi burde se etter.

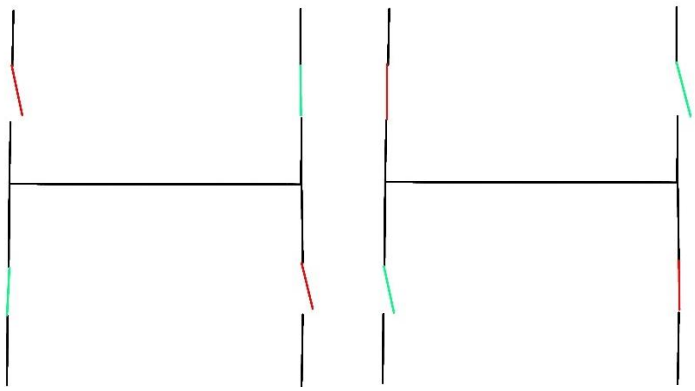
- RPM: Rotations Per Minute, er rotasjonshastigheten til motoren.
- (Rated) Voltage: Spenningen komponenten er laget for, hvor RPM og strømuttaket ble målt.
- Current: Hvor mye strøm komponenten trekker ved vanlig bruk.
- Størrelse og vekt.
- Torque: Hvor mange newtons motoren kan levere x antall cm fra sentrum.

For motoren vist i figur 1 gjelder:

- RPM: 8000RPM
- Voltage: 9V
- Current: 68mA (For et vanlig 9V-batteri på 400-600mAh er det 6-9 timer)
- 24 x 31 mm, 43g
- Torque: ukjent

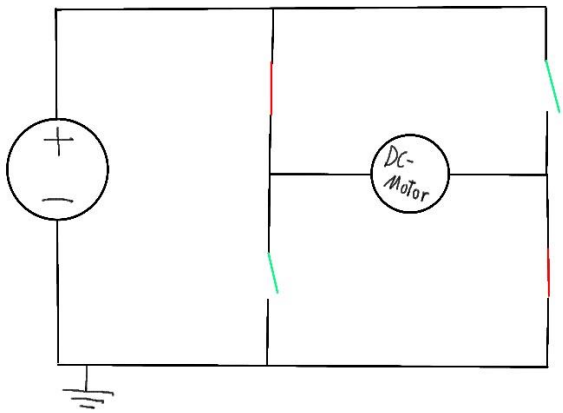
H-bro

En H-bro er en (under)krets vi bruker for å styre DC-motorer. Det er en sammenkobling som lar oss bytte retningen på strømmen gjennom en komponent. En motordriver, som det står mer om i neste kapittel, er bygget av flere H-broer satt sammen. En H-bro kan også tenkes på som en komponent som lar oss endre strømretningen gjennom to pinner, altså hvilken av pinnene som er lav, og hvilken som er høy.



Figur 5: To identiske H-broer i sine to forskjellige tilstander. Komponenten skal plasseres midt i. Øverst er det høy spenning, og nederst er jord.

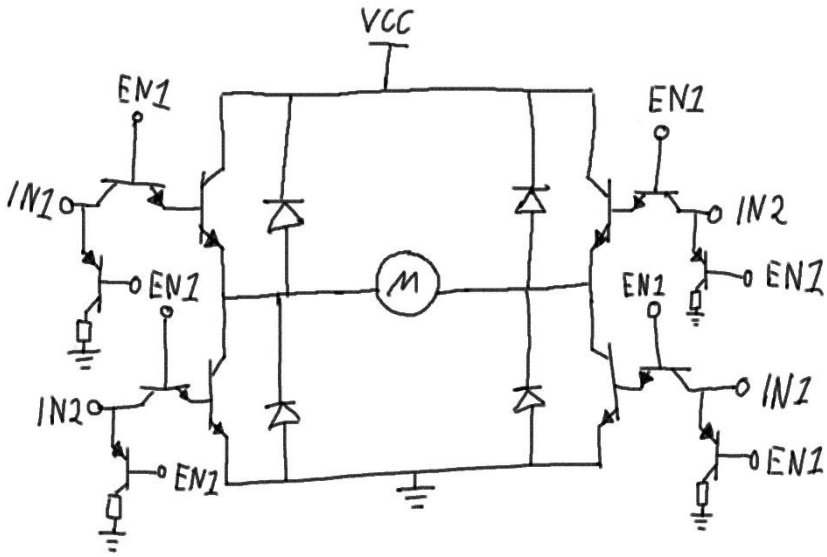
Slik du kan se i figur 5 er H-broer kalt H-broer av en ganske så enkel grunn. Funksjonaliteten til H-broen er slik at de to røde bryterene er enten begge på eller av, og samtidig er da begge de to grønne bryterene motstatte av de røde. Under, i figur 6, vises nytteverdien til en slik krets.



Figur 6: En H-bro med en DC-motor

Hva skjer nå i figur 6 om tilstanden til H-broen endres? Jo, den får DC-motoren til å endre retning. DC-motoren kan da, uten å koble om noe som helst snu retning.

Figur 7 viser en mye mer virkelighetsnær utgave av en H-bro. EN1 står da for ENABLE 1, og IN1 og -2 står for INPUT 1 og 2. De er alle pinner du setter til lav eller høy for å styre strømmen videre. Diodene er til stede for ikke å skade transistorene, da en motor, M, som deaksellererer genererer strøm.



Figur 7: Et skissert design til en enkel H-bro

Motordriver

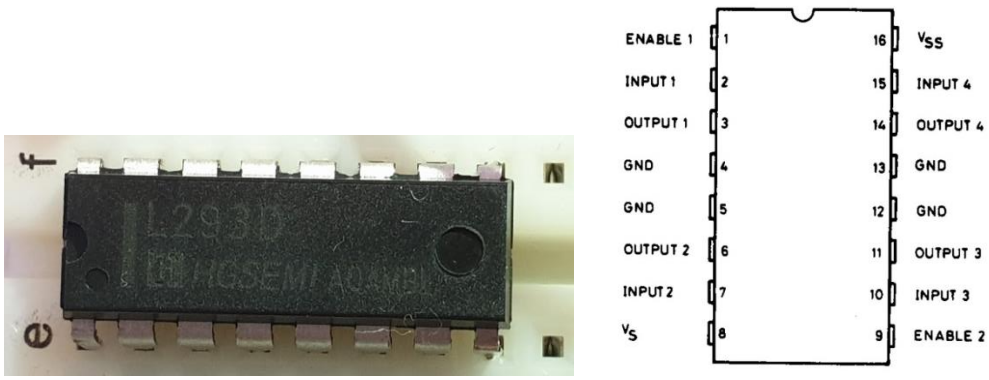
Hva er det?

En motordriver er en integrert krets laget for å styre DC-motorer. De bruker logiske inngangssignal, generelt fra mikrokontrollere, for å styre en sterkere spenningskilde. Vanligvis har motordriverne fire INPUT-pinner, to ENABLE-pinner, fire OUTPUT-pinner, en logisk spenningsforskyning, og en spenningsforskyning til motorene sine. Én motordriver kan som regel dermed styre to forskjellige *sett* av motorer.

En radiostyrt bil vil for eksempel kunne koble motorene på sine to venstre hjul til OUTPUT 1 og 2, og motorene på sine to høyre hjul til OUTPUT 3 og 4. Den kan da ikke ha flere hjul som skal styres separat, men den kan ha eksempelvis to ekstra bakhjul som skal gjøre som de andre motorene. Da kan bilen styre alle de venstre hjulene sine enten forover eller bakover, samtidig som den kan styre alle sine høyre hjul enten forover eller bakover.

L293D – Arduinokitens motordriver

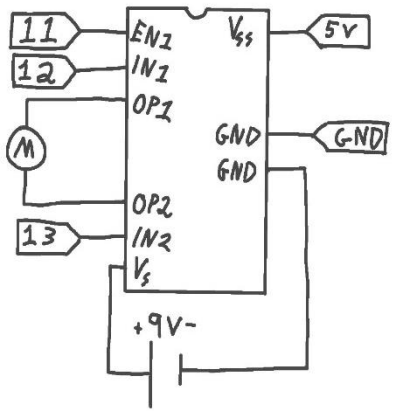
I arduinokitet fra Elektra finnes det en veldig vanlig motordriver, L293D. Den er blant de mest vanlige motordriverne for hobbyprosjekter i verden. L298N er også rimelig vanlig og så godt som lik L293D. Figur 8 viser DIP-versjon av motordriveren festet til et breadboard, samt pinne-oversikt fra datablad. Den bruker ikke transistorer slik som i figur 7, men bruker op-amper isteden.



Figur 8: L293D IC (integrert krets) DIP-versjon (Dual In-line Package) med pinne-oversikt fra databladet dens

Oppkobling

Under, i figur 10, er et eksempel på hvordan koble opp motordriveren til en arduino (pinner vist ved flagg) og en motor, M.



Figur 9: Eksempeloppkobling av L293D med Arduino

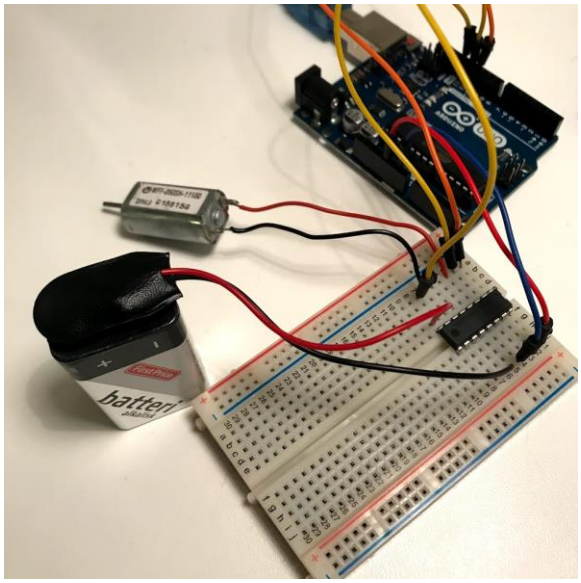
Standarden for hvordan styre slike motordrivere er:

- Sett en av IN1 og IN2 til LOW, og den andre til HIGH med digitalWrite() for å bestemme motorens retning.
- Bruk analogWrite til hurtig å skru av og på EN1. Høyere analogWrite gir raskere rotasjon.

La oss se på et eksempel. Med oppkobling lik vist i figur 9 og i figur 10 kjører vi denne koden:

```
const int8_t EN1 = 11;
const int8_t IN1 = 13;
const int8_t IN2 = 12;

void setup() {
  pinMode(EN1, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);

}

void loop() {
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);

  for(int8_t i = 30; i < 55; i++){
    analogWrite(EN1, i);
    delay(100);
  }
}
```

Kodesnutt 1: Få én DC-motor til å snurre i forskjellige hastigheter



Figur 10: Oppkobling til kodesnutt 1. Se figur 9 dersom uklart

Hvorfor skriver vi PWM til ENABLE istedenfor INPUT?

«Når koden kjører vil det at ENABLE1 skrus av og på (PWM) være det samme som om vi hadde skrudd INPUT1 av og på», kan en fort tenke på å si. Om du derimot prøver å koble et PWM-signal til INPUT1 vil du merke at motoren ikke fungerer like godt. ENABLE-pinnene er nemlig optimalisert for å ta imot PWM-signalene, og mer effekt blir overført til OUTPUT om de brukes.

I tillegg til dette er det å tilby PWM på en mikrokontroller plasskrevende. Det ønskes derfor å begrense mengden PWM-pinner. Ved å bruke kun to PWM-pinner til ENABLE1 og ENABLE2, i stedet for å bruke en på hver eneste INPUT-pinne, lar det oss bruke PWM-pinnene våre til andre ting imens.

Lite tips:

På alle Dual In-line Packages

(slik som L293D)

vær forsiktig med pinnene når du tar de ut av breadboardet så de ikke bøyes.

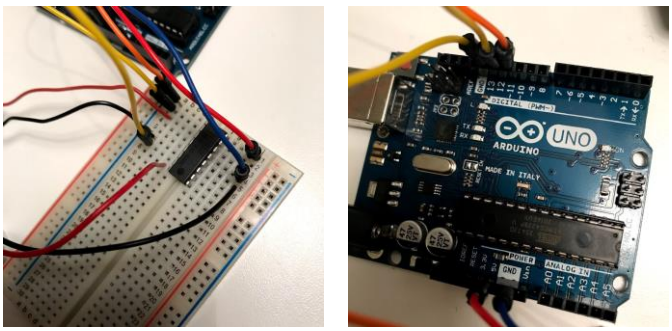
Det er også strukturelt pent og pyntelig at vi har egne pinner for *retning* (INPUT), og egne pinner for *styrke* (ENABLE). Det gjør gjerne programmene våre enklere å skrive, og enklere å tolke.

Oppgaver – del 1 av 2 – DC-motorer

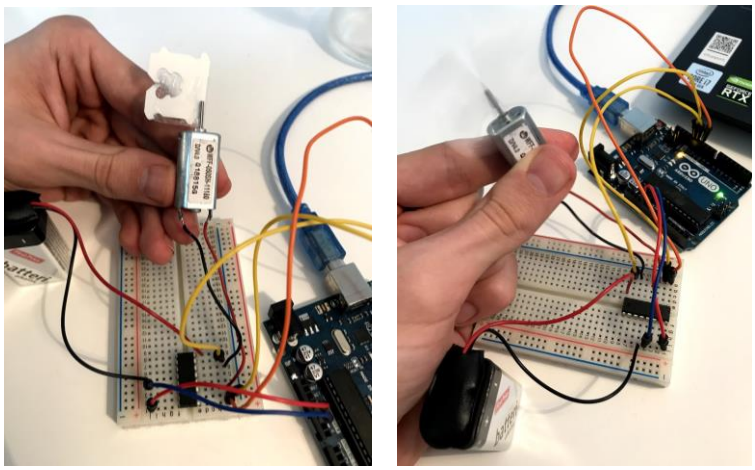
Oppgave 1 – oppkobling

NB: Før du går videre, gjør deg selv veldig obs på at V_{SS} og V_S er to helt forskjellige spenninger. V_{SS} skal være 5V. V_S skal være 9V.

Koble opp kretsen fra figur 11, og skriv et kort program som får motoren til å snurre. Figur 12 er et eksempel på hvordan slike bilder kan se ut sammen. Begrens gjerne styrken til analogWrite på ENABLE til maksimalt 55, motoren kan gå **veldig** fort.



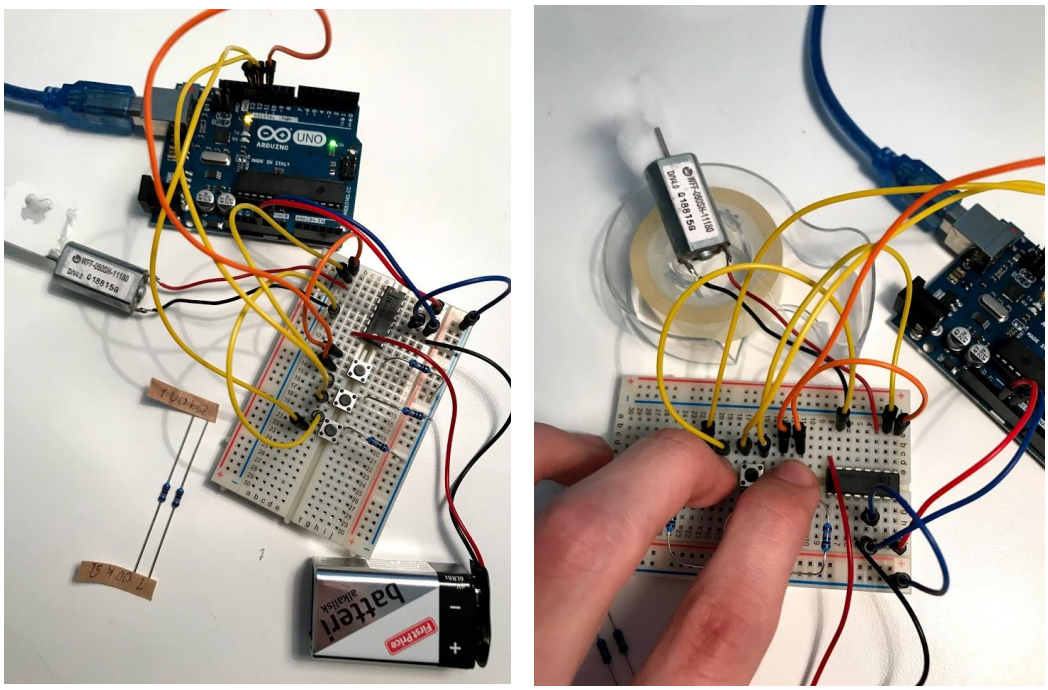
Figur 11: Oppkobling til oppgave 1.



Figur 12: Eksempel på oppkobling. Det er et pappklipps fra en brødpose festet til motoren med teip for å vise rotasjonen (t.h). Finn kreative løsninger, men pass på at motoren ikke går for fort. En isoporbit/bit med svamp kan gjerne brukes.

Oppgave 2 – knapper

Med utgangspunkt i forrige oppgave. Skriv analogWrite() mellom 30 og 55 til ENABLE, og digitalWrite(HIGH) til INPUT-pinnene. Fest så pull-down-knapper til ENABLE- og INPUT-pinnene, slik at dersom du ikke trykker inn knappene er alle inngangene lav.



Figur 13: Oppkoblings- og innleveringseksempel for oppgave 2

Oppgave 3 – fade

Ta innspirasjon fra kodesnutt 1. Oppgaven er å få motoren til å gå bakover i høy hastighet, for så å gradvis sakke farten, til så å gradvis aksellerere til høy hastighet forover. Den skal så gradvis sakkes tilbake til høy hastighet bakover igjen, og så videre.

Oppgave 4 – manuell kontroll

Bruk et potensiometer som en spenningsdeler, les av spenningen fra tappepunktet på potensiometeret, og bruk verdien til å bestemme hastigheten på motoren. Om potensiometeret er vridd fullstendig mot én retning vil det si at motoren skal ha høy hastighet – analogWrite(55) – i den retningen.

Midt på potensiometeret skal ikke motoren bevege seg. Vries potensiometeret til den andre retningen skal motoren ha høy hastighet i den retningen. Det anbefales at du bruker map()-funksjonen til denne oppgaven for å gjøre den enklere og mer effektiv.

Forklaring – map()

Syntaks:

map(value, fromLow, fromHigh, toLow, toHigh)

Forklaring:

Returnerer den brøkvise ekvivalenten av value, som var i intervallet [fromLow, fromHigh], men nå i intervallet [toLow, toHigh]. Eksempelvis vil map(2, 1, 3, 1, 7) = 4 fordi 2 (value) er midt mellom 1 (fromLow) og 3 (fromHigh), og 4 (returverdi) er midt mellom 1 (toLow) og 7 (toHigh).

Definisjon:

```
long map(long x, long in_min, long in_max, long out_min, long out_max)
{
      return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min;
}
```

For de mer interesserte, merk dere: Arduino C er ganske snilt, og lar oss ta imot long som andre integer-variabeltyper, men desimalregning fungerer ikke med map.

Oppgave 5 – nødstopp

Dersom denne motoren hadde styrt noe litt viktig, slik som kanskje en rulletrapp for kaniner eller et løpehjul for hamstre, så ville vi gjerne hatt en nødstopp tilgjengelig. Uten nødstopp kan det hende at en uheldig gnager blir skadet, tross alt. I arduinokiten deres befinner det seg en tilt-sensor.

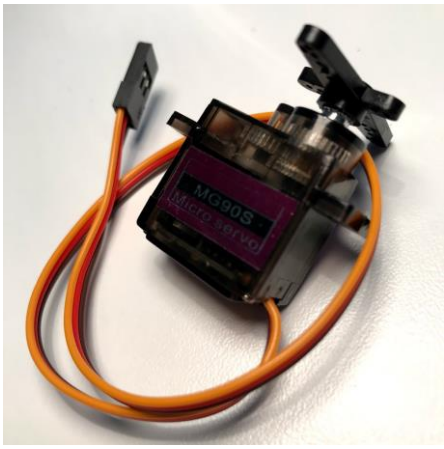
Tilt-sensoren er ekvivalent med en kortslutning dersom den står vertikalt, men om den vippes 45-90 grader eller mer fungerer den som en åpen krets. Det er altså en knapp, «but with extra steps». Dersom denne sensoren vippes, skal det implementeres en interrupt på pinne 1 eller 2 for å stoppe motoren frem til sensoren vippes tilbake.

Servomotor

Hva er en servomotor?

En servomotor er som en DC-motor som er begrenset til å rotere 180 grader og som har en kontroller som vet hvordan den er orientert. Slik kan motorens *posisjon* kontrolleres med et PWM-singal, i stedet for hastigheten dens. Servomotorer er egentlig mye mer enn det, men slik er de mest vanlige servomotorene.

Servomotorer kommer også i AC-versjon, synkron eller asynkron, fra 90 til 270 grader, eller i kontinuerlig roterbar form med fartskontroll (som en DC-motor, men med kontrollerbar fart i steden for kontrollerbar effekt). Videre vil «servomotor» ikke referere til alle disse versjonene, men kun 180-graders DC-servomotorer; servomotoren i arduinokitet, MG90S, er slik. Du kan se et bilde av MG90S i figur 14.



Figur 14: MG90S fra arduinokitet

Servomotorer tar imot et inngangssignal i form av et PWM-signal. Ved hjelp av en sensor, ofte et potensiometer, vet servomotoren hvordan den er rotert, og justerer motorens posisjon til å passe til det inngangssignalet betyr. Servomotorer er egentlig bare DC-motorer med masse ekstra rundt seg. Hva inngangssignalet betyr for servoens posisjon kommer i neste underkapittel.

Det er også en liten ikke-justerbar «girboks» i servomotorene, som begrenser fart for å gi sterkere rotasjonskraft. Med tanke på at de fleste servomotorer kun roterer 180 grader er ikke

ekstreme hastigheter et stort krav uansett, og kraft kan prioriteres. Tannhjulene bidrar også til større nøyaktighet i bevegelsene til servomotoren, da større rotasjoner med DC-motoren fører til mindre rotasjoner i akslingen.

Hvordan kontrollerer vi en servomotor?

Servomotoren tar imot et PWM-signal på 50Hz. Lengden av den høye pulsen i PWM-signalet avgjør motorens posisjon.

- Kort puls (1ms høy etterfulgt av 19ms lav) gir én retning (0 grader).
- Mellomlang puls (1,5ms høy etterfulgt av 18,5ms lav) gir at servomotoren står midt på (90 grader).
- Lang puls gir (2ms høy etterfulgt av 18ms lav) den andre retningen (180 grader).
- Ingen signal (20ms lav) gir ingen posisjon, servoen står bare stille.

Heldigvis har Arduino gjort det litt enklere for oss å styre servoer, og vi trenger ikke huske disse reglene utenatt. Noen har nemlig laget et bibliotek for oss, Servo.h. Biblioteket lar oss opprette servo-objekter, feste de til pinner, og skrive hvor mange grader vi ønsker servoene skal stå på. Da slipper vi altså å sende inn egendefinerte PWM-signaler, så deilig. Å bruke biblioteket hindrer oss forresten i å bruke pinne 9 og 10 til analogWrite(), selv om servoene ikke står på de pinnene. Det er derfor greit å bruke pinne 9 og 10 til servoer.

```
#include <Servo.h>

const int servoPin = 9;
const int t = 1000;

Servo servo;

void setup() {
  servo.attach(servoPin);
}

void loop() {
  servo.write(0);
  delay(t);
  servo.write(90);
  delay(t);
  servo.write(180);
  delay(t);
}
```

Kodesnutt 2: Servokode

Vi kobler opp den brune ledningen på servoen til jord, den røde til 5V, og den oransje til pinne 9 eller 10. Med servobiblioteket oppretter vi et nytt servo-objekt, og bruker «.attach()» til å feste det til pinnen vi bruker. Deretter kan vi skrive «.write()» til objektet, og spesifiserer antall grader vi ønsker servoen skal gå til. Dette kan vi se et eksempel av i kodesnutt 2. Enklere blir det ikke.

Hva skal vi se etter på databladet til en servo?

- Torque: Dette er hvor mye kraft servoen kan tilby x antall cm ut fra sentrum av akslingen. Eksempelvis vil $20\text{N} * \text{cm}$ si at den kan dytte med 40N en halv cm fra sentrum, 20N én cm fra sentrum, og 10N to cm fra sentrum. Dette oppgis som regel i antall kilo den kan løfte mot tyngdekraften ($\text{kgf} * \text{cm}$) i stedet for newtons. $20\text{N} * \text{cm}$ er da $20/\text{g} = 20/9.81 = 2.038$ [$\text{kgf} * \text{cm}$].
- Speed: Dette er hvor fort servomotoren kan rotere, og er oftest oppgitt i antall ms det tar å rotere 60 grader. Eksempelvis 0.1s/60deg.
- Hvor mange grader den kan snurre.
- Tannhjulmaterial: Servoer har som regel tannhjul av enten metall eller plast. Plast er billigere, lettere, men fordi materialet ikke er like solid som metall må hjulene ha større tenner. Det fører til at servoen kan oppleves som litt rykkende. Metall er tyngre og dyrere, men gir mer nøyaktige bevegelser, og varer lengre.

  Merk at mange servoer gjerne har ett tannhjul av plast, selv om de sier de er «metal geared». Dette er gjort med vilje for å gi mekanikken et bristepunkt. Uten bristepunktet kan motoren, dersom hindret bevegelse, ta all effekten, varmes opp, og smelte eller ta fyr. Det er derfor en positiv ting om du ser at servoen din har ett plast-tannhjul.

For MG90S gjelder (ved 4.8V):

- Torque: $1.8\text{kgf} * \text{cm}$
- Speed: 0.1s/60deg
- 0 til 180 grader
- Metall-tannhjul, med ett plast-tannhjul (i figur 14 kan du se det rett over det lilla merket)

Oppgaver – del 2 av 2 – servo

Oppgave 6 – test

Få servoen til å gå sakte fram og tilbake fra 0 til 180 til 0 til 180 grader, og så videre. Gjør slik at rotasjonshastigheten er nøyaktig 600ms/60deg.

Oppgave 7 – potensiometer

Koble opp et potensiometer slik som i oppgave 4, og få servoen til å bevege seg fra 0 til 180 grader slik at servoens rotasjon passer til vridningen av potensiometeret.

Oppgave 8 – fartsmåler

Med samme oppkobling og kode som i oppgave 5, legg nå på servoen i kretsen også. Servoen skal utfylle nesten samme funksjon som i oppgave 7, altså å være basert på potensiometerets innstilling. I kombinasjon med motoren blir servoen dermed en slags fartsmåler.

I motsetning til oppgave 7 skal servoen vise motorens absolutte fart, altså være helt til venstre når motoren ikke beveger seg, og helt til høyre når motoren har høy hastighet.

Oppgave 9 – fotoresistor-basert fart

Bytt ut potmeteret til motoren med to fotoresistorer i serie som spenningsdelere. Setter du en kondensator fra målepunktet til jord får du jevnere kontroll.