UiO : Institutt for informatikk
Det matematisk-naturvitenskapelige fakultet

Prototyping med Arduino del 2

Magnus Li
magl@ifi.uio.no

INF1060
29.01.2018

Arduinoundervisningen

Forelesninger
Mandag 29.01 & 05.02
Gjennomgang av grunnleggende temaer

Teknisk verksted **(rom C)**
Mandag **29.01**, 05.02, 12.02 & 19.02
Hjelp til ukesoppgaver

Gruppetimer
Uke 5 & 6
Hjelp til ukesoppgaver

Obligatoriske oppgaver

1) Frist 09.02
   Utvalgte ukesoppgaver skal leveres

2) Frist 23.02
   Miniprosjekt skal leveres

Nå

Grunnlag
- Litt om elektrisitet
- Litt om elektriske kretser
- Litt om signaler

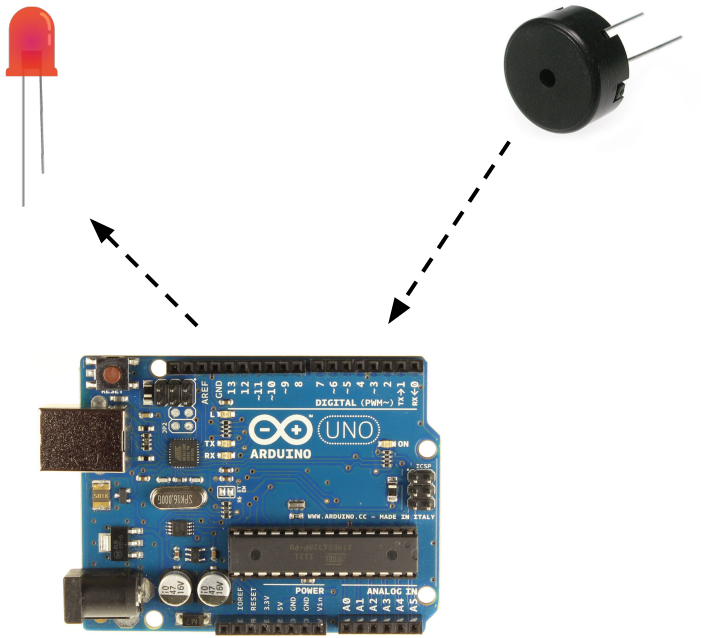
Arduinoprogrammering
- Komme i gang
- setup() og loop()
- Sende og lese digitale signaler
- Sende og lese analoge signaler

Hva er Arduino?

En liten datamaskin eller *mikrokontroller.*

Den lar oss:

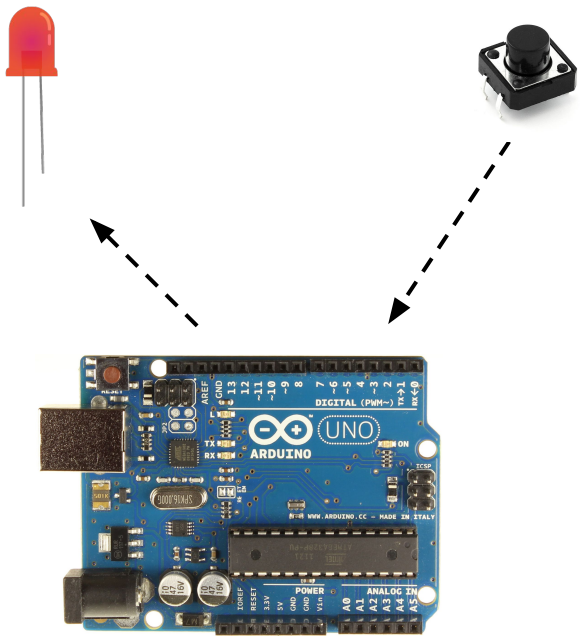
- Ta inn signaler
- Gjøre noe med signalene
- Sende signaler ut



Hva er Arduino?

Vi kan bestemme hva vi vil sende ut, og hva vi skal gjøre med signalene vi får inn, ved å programmere Arduinoen i et språk som likner på Java.

```
int led = 2;
int button = 8;

void setup() {
        pinMode(led, OUTPUT);
        pinMode(button, INPUT);
}

void loop() {
       if (digitalRead(button) == HIGH) {
              digitalWrite(led, HIGH);
       } else {
              digitalWrite(led, LOW);
       }
}
```

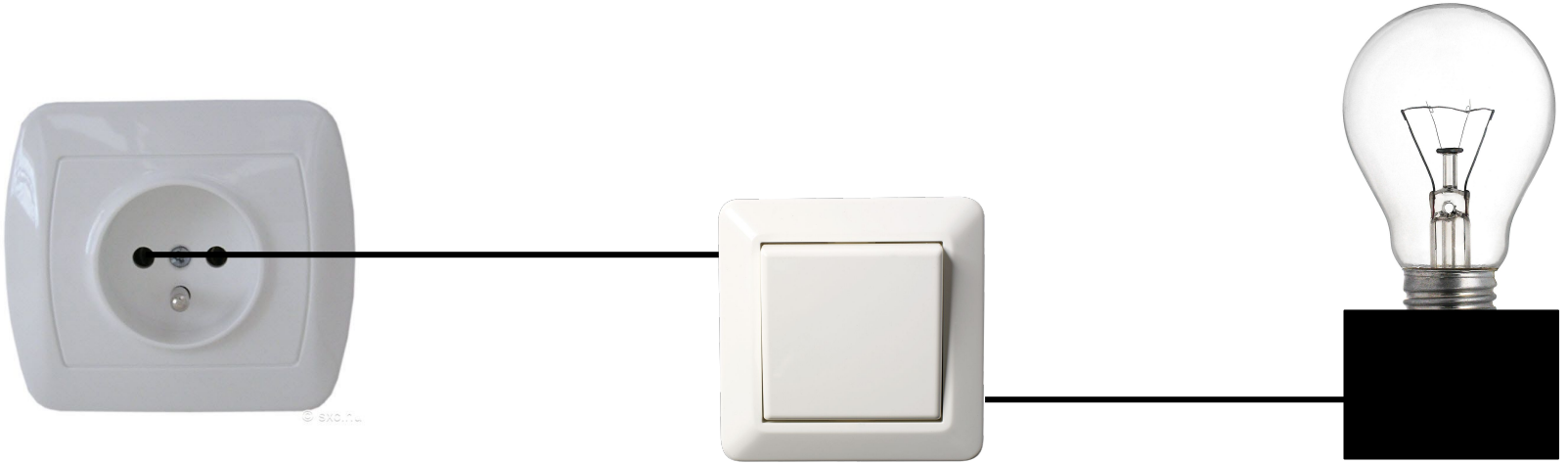


Grunnlag

Elektrisitet

Elektrisitet

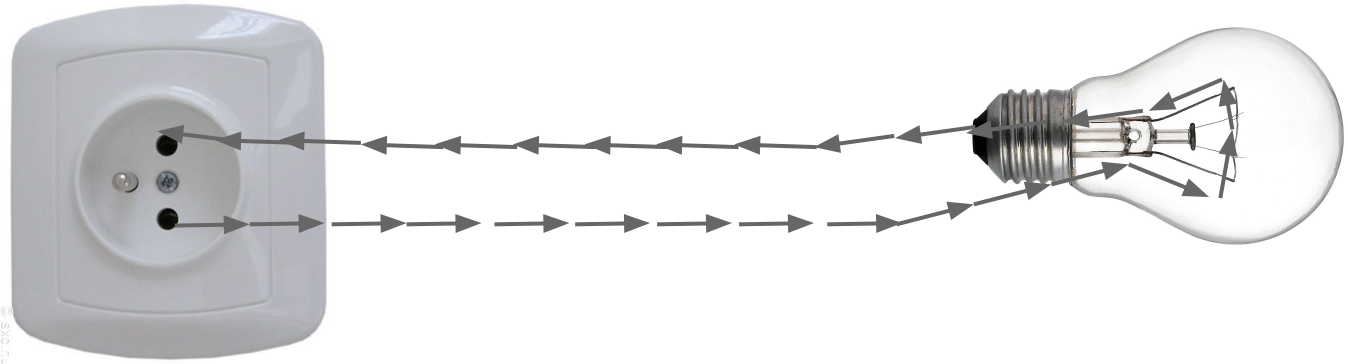
Målet her er å prøve å forklare elektrisitet så enkelt som mulig. Det vil si at mange detaljer blir abstrahert bort.



Elektrisitet

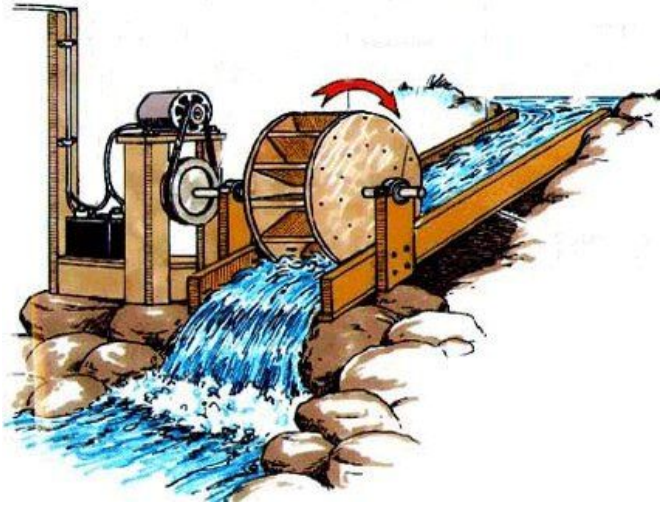
For enkelhetsskyld pleier mann å si at elektrisitet *flyter* fra pluss til minus.

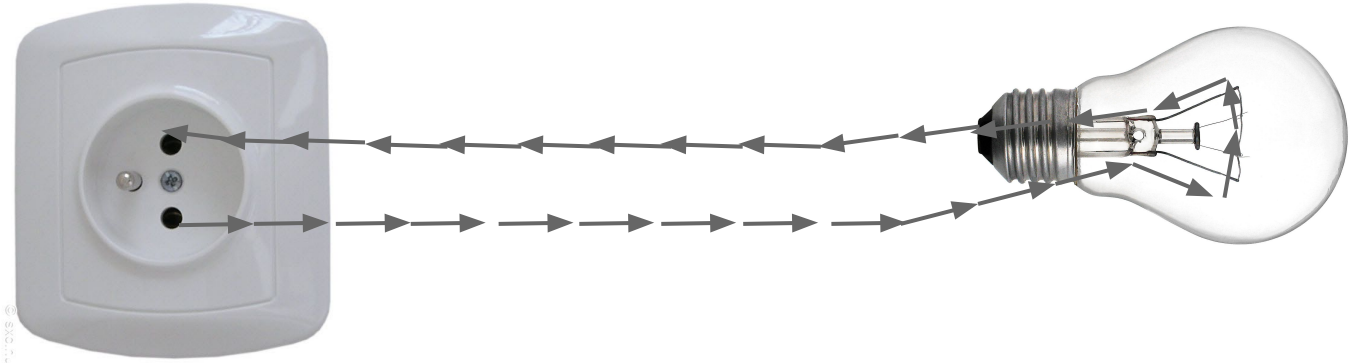
Viktige begreper
- Strøm (Ampere)
- Spenning (Volt)
- Motstand / resistans (Ohm)



Strøm

Strøm referer til strømmen av elektroner i en krets. Dette måles i enheten Ampere.



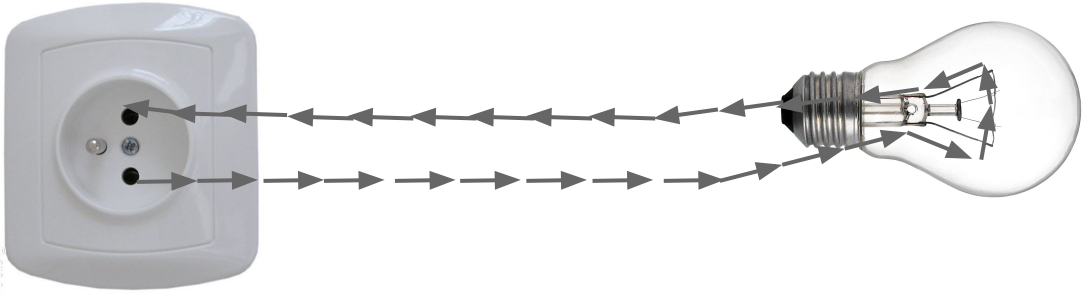


Spenning

Spenning er en forskjell i ladning. Dette driver strømmen i elektriske kretser.

Forskjell i ladning på hver side av demningen

Forskjell i ladning mellom + og - i kontakten



Spenning

Spenning måles i enheten Volt.

230V

5V

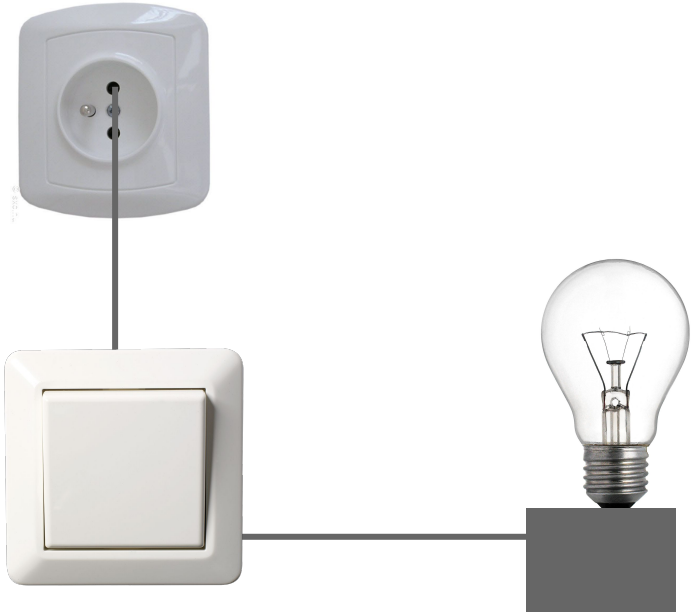
9V

1.5V

Måleenheten for spenningsforskjellen mellom to punkter er oppkalt etter Alessandro Volta. (Volt)

Strøm og spenning

Analogi: strøm og spenning er som vann og trykk



Motstand / resistanse

Motstand beskriver hvor mye et materiale bremser, eller reduserer mengden strøm som flyter gjennom.

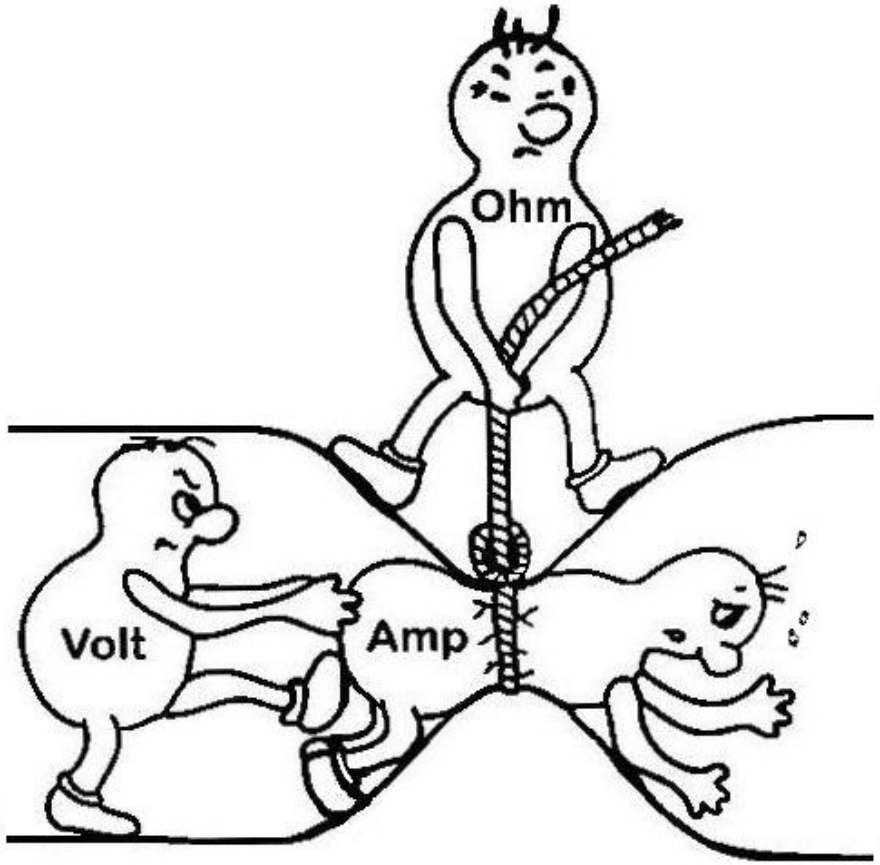
Alt har en bestemt motstand. Elektriske ledninger har forholdsvis liten motstand. Gummi har uendelig stor motstand, og leder dermed ikke strøm.

Vi ser motstand i aksjon i lyspærer og varmeovner.

Stor motstand i ledningen i lyspæren gjør at den begynner å gløde, og dermed lyser.

Strøm, spenning og motstand

- Strøm (amp)
- Spenning (volt)
- Motstand / resistan (ohm)

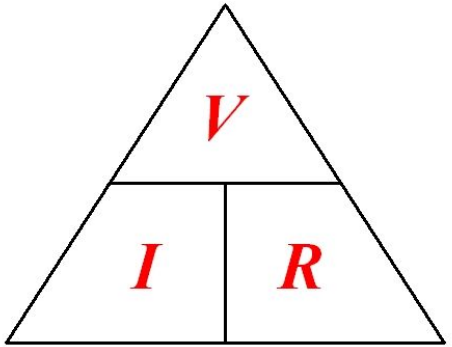


Strøm, spenning og motstand: Ohms lov

- Strøm (I)
- Spenning (V)
- Motstand / resistans (ohm)

Det er et forhold mellom disse, som kan beregnes med Ohm's lov. Ved interesse eller behov kan dere lese mer om dette i prosjektboken, eller på nettet.

$$V = I\,R$$



$$I = \frac{V}{R} \qquad R = \frac{V}{I}$$

Elektrisitet

Er det farlig?

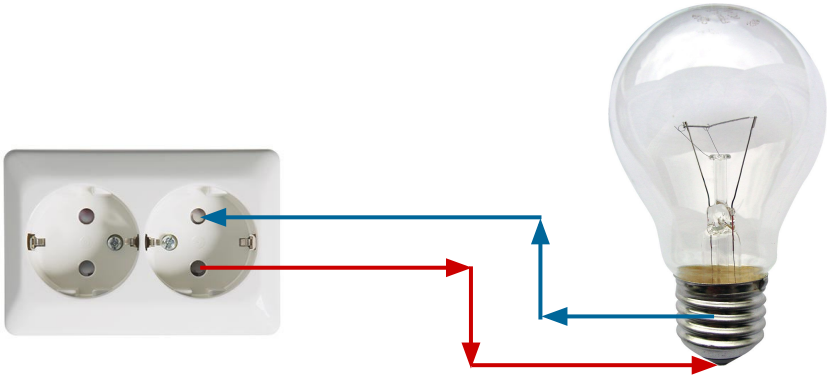
Elektrisitet ved høy spenning og ampere kan være svært skadelig, men på det spenningsnivået vi har med Arduino (9V - 3V) er det ingen fare.

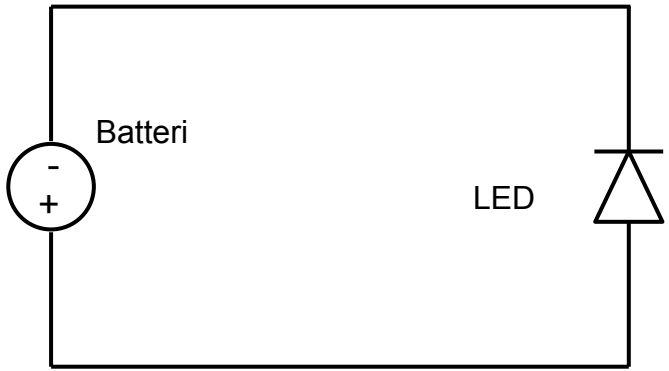
Grunnlag

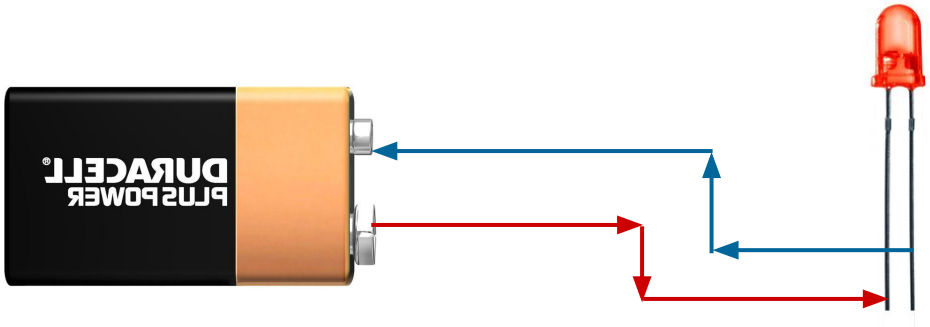
Elektrisitet i kretser

Elektrisitet i kretser

Elektrisitet i kontrollerte former beveger seg alltid i det vi kaller en sluttet krets.





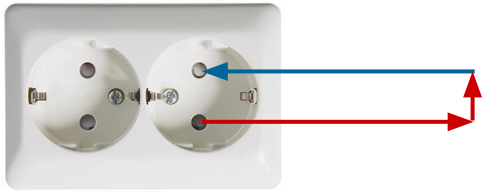


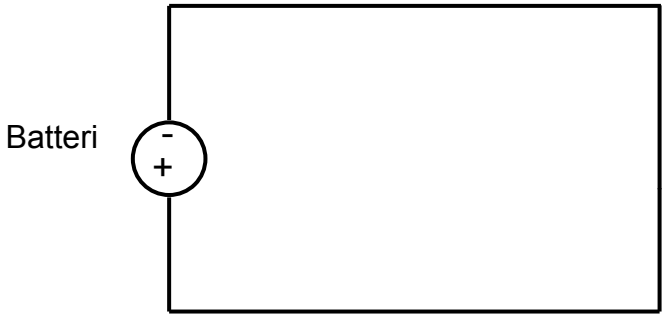
Elektrisitet i kretser

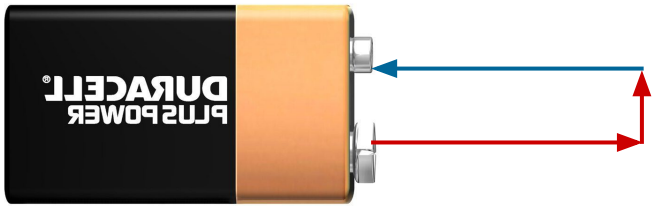
Kortslutning

Dersom vi ikke har noen komponent i koblingen mellom + og -, vil vi få en kortslutning.

Dette kan skade elektriske komponenter, og er noe vi må unngå.







Elektrisitet i kretser

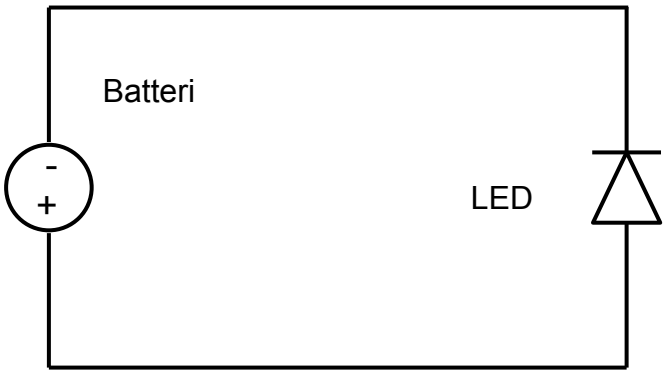
Kortslutning

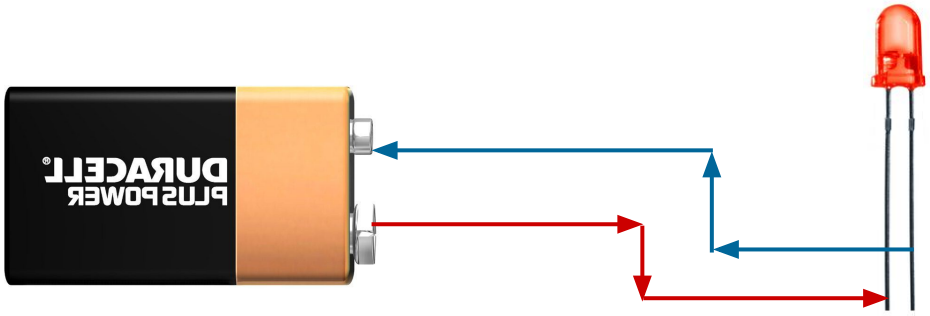
Dersom vi ikke har noen komponent i koblingen mellom + og -, vil vi få en kortslutning.

Dette kan skade elektriske komponenter, og er noe vi må unngå.

Elektrisitet i kretser

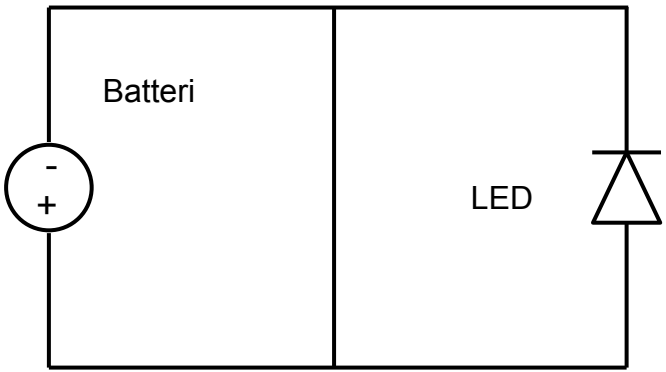
Elektrisitet i kontrollerte former beveger seg alltid i en sluttet krets, og **vil ta veien med minst motstand.**

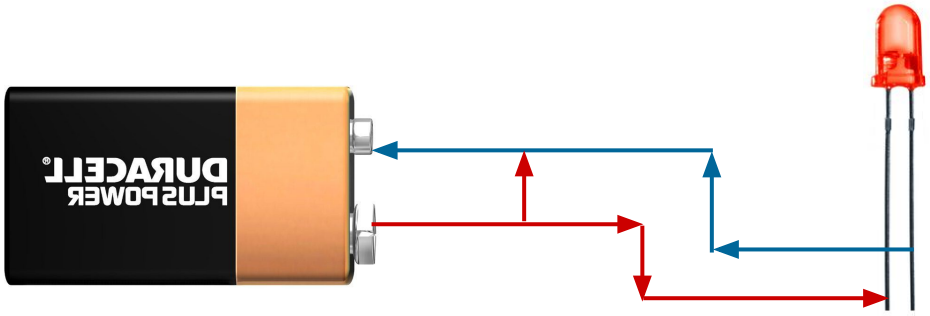




Elektrisitet i kretser

Elektrisitet i kontrollerte former beveger seg alltid i en sluttet krets, og **vil ta veien med minst motstand.**

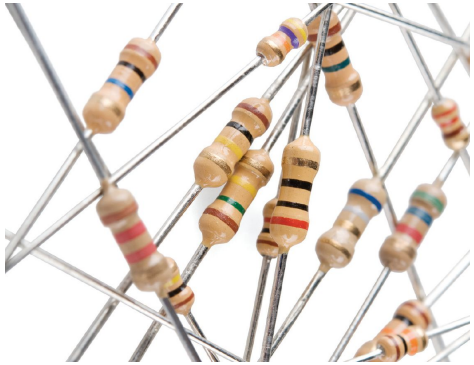


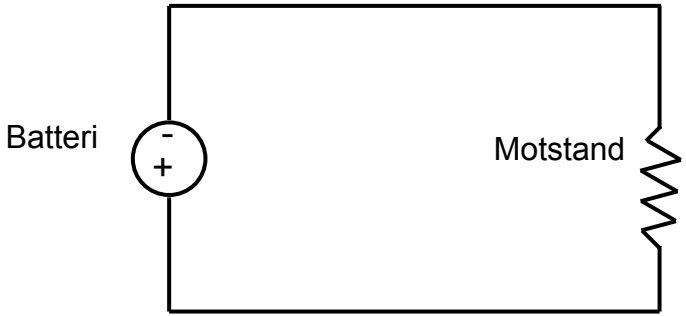


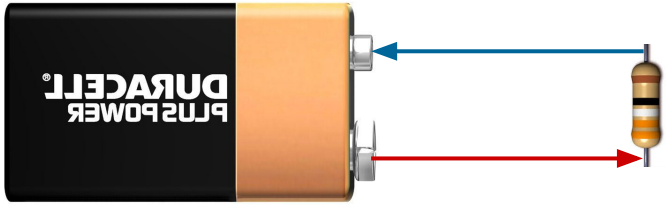
Motstander i kretser

Dersom vi ønsker en krets uten spesielle komponenter kan vi benytte resistor for å unngå kortslutning.

En resistor innehar en bestemt motstand. Dette reduserer mengden strøm i kretsen.

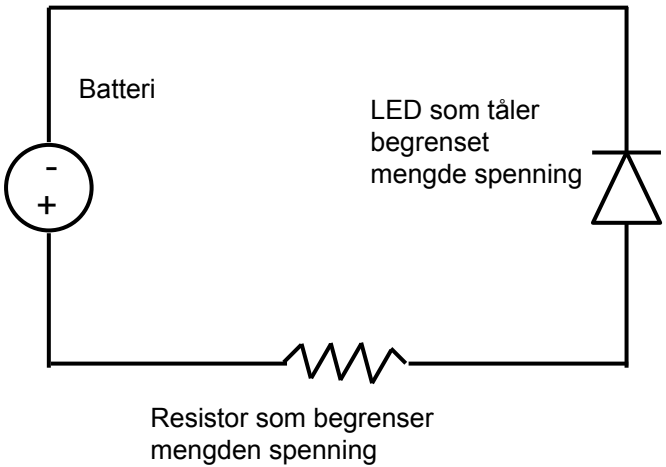


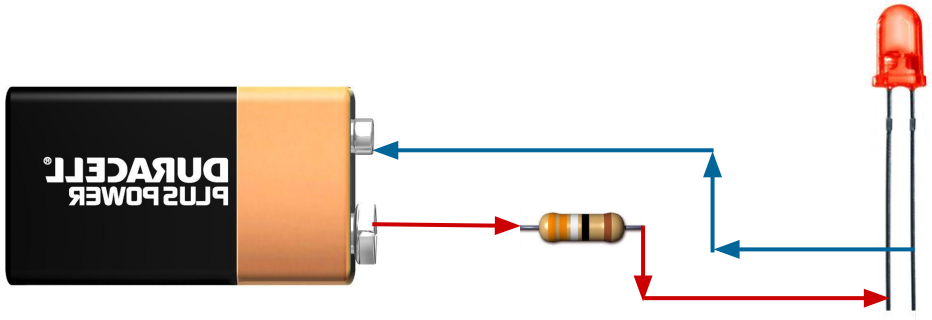




Motstander i kretser

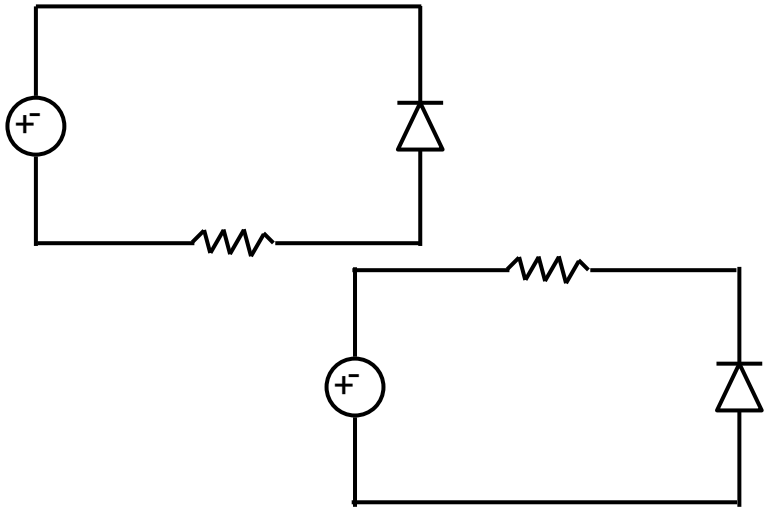
Vi bruker også resistorer for å aktivt redusere mengden strøm i en krets. Dette fordi noen komponenter tåler mindre strøm enn vi i utgangspunktet får fra strømkilden.

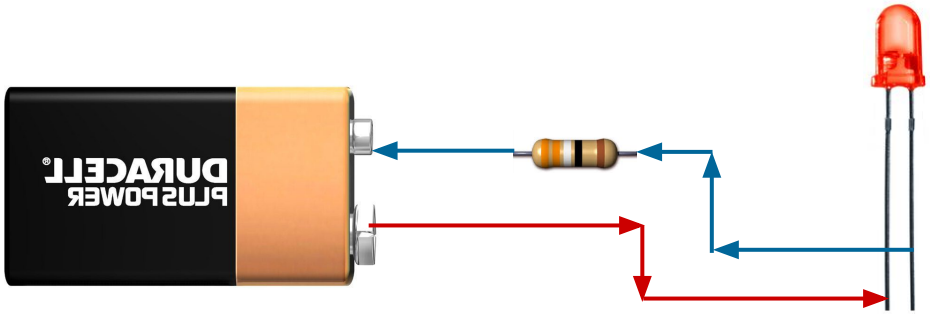


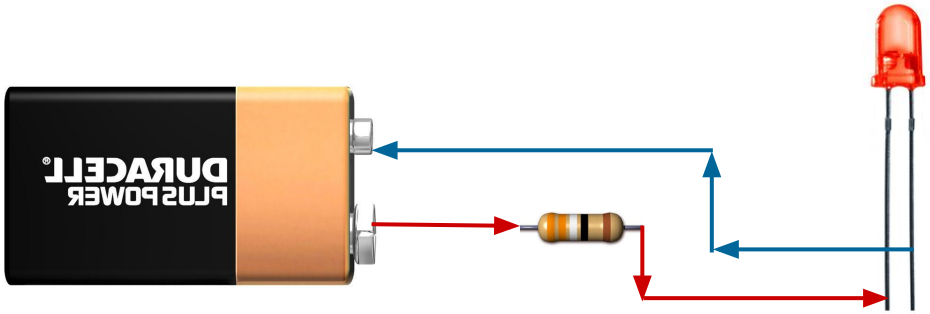


Motstander i kretser

Det spiller ingen rolle hvor i kretsen vi putter resistor. Strømmen i hele kretsen blir redusert uansett.



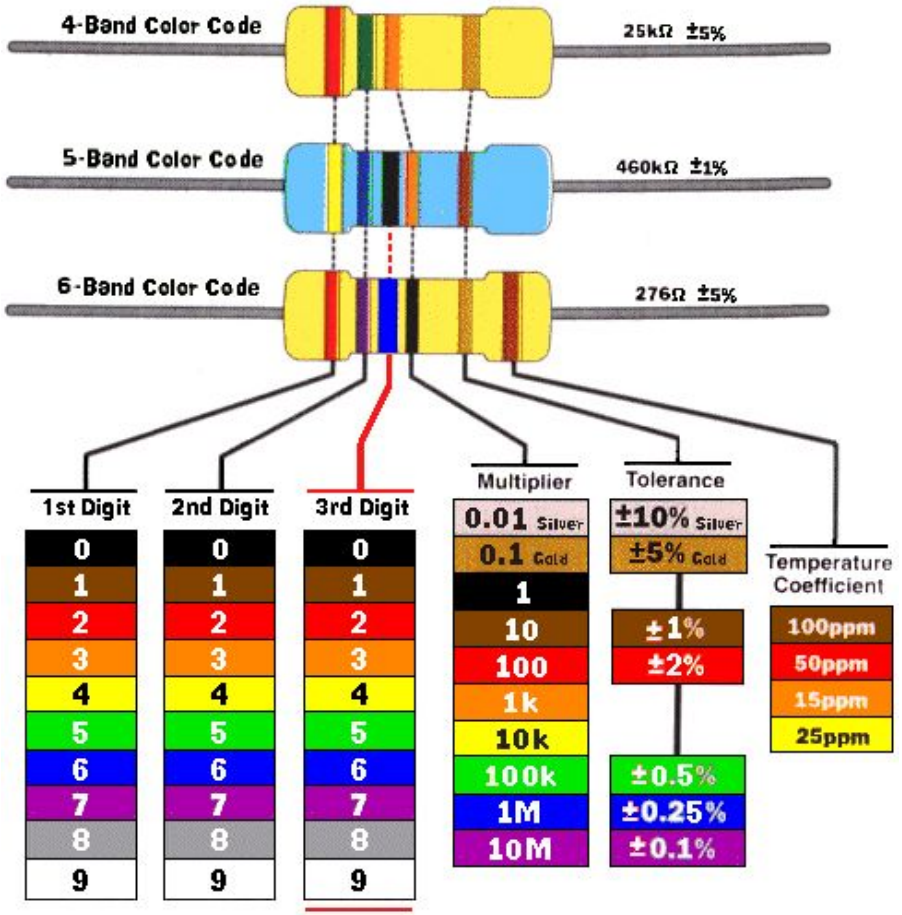




Motstander i kretser

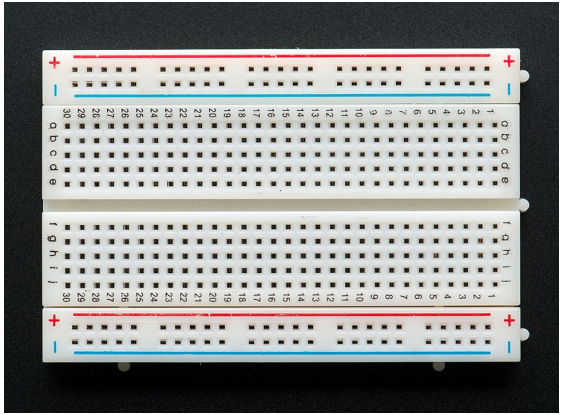
Ulike komponenter og kretser krever ulike resistorer. Fargen på resistoren sier noe om hvor stor motstand de har i måleenheten Ohm.

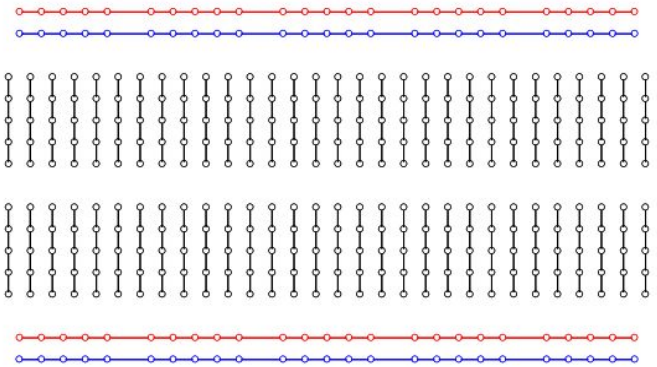
I starter-kit boken på side 41 er det en fin oversikt over dette.



Koble kretser: breadboards

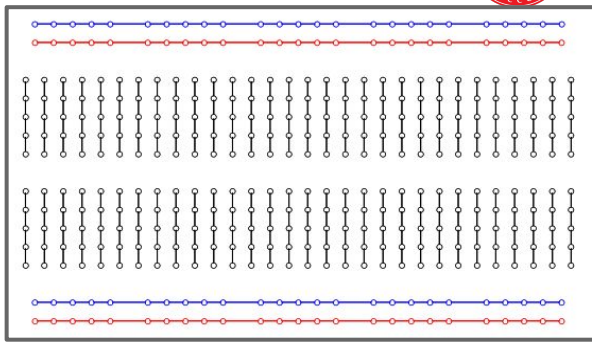
Til Arduino følger det med et såkalt breadboard. Dette bruker vi for at det skal være lettere og mer oversiktelig å sette opp kretser med komponenter.

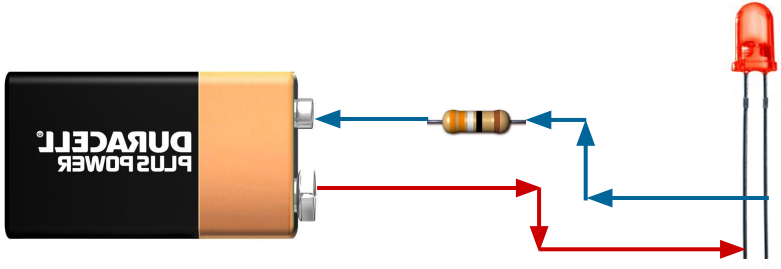




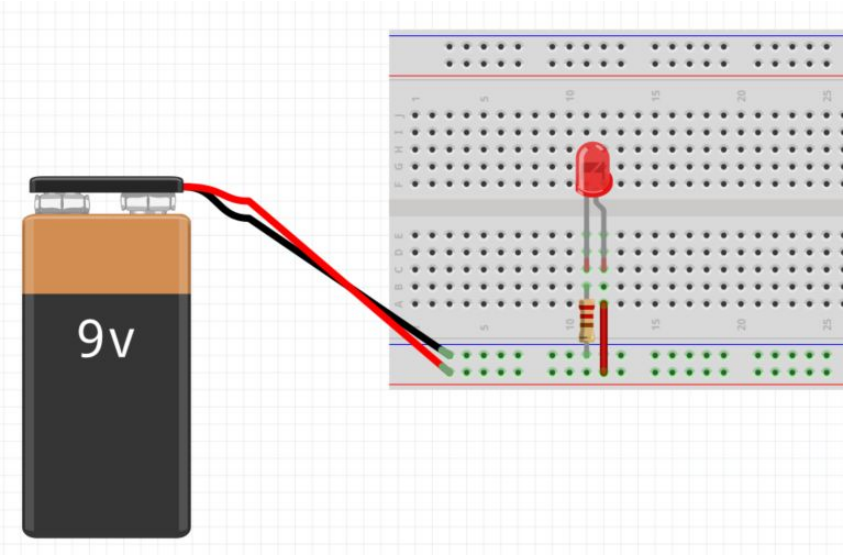
Koble kretser: breadboards

De to koblingene vist under er de samme, men det er mye enklere å arbeide på et breadboard.



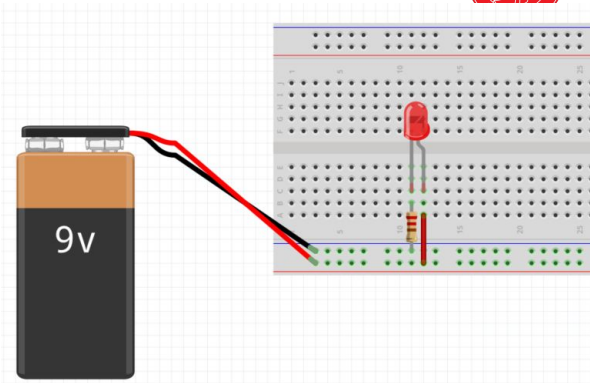


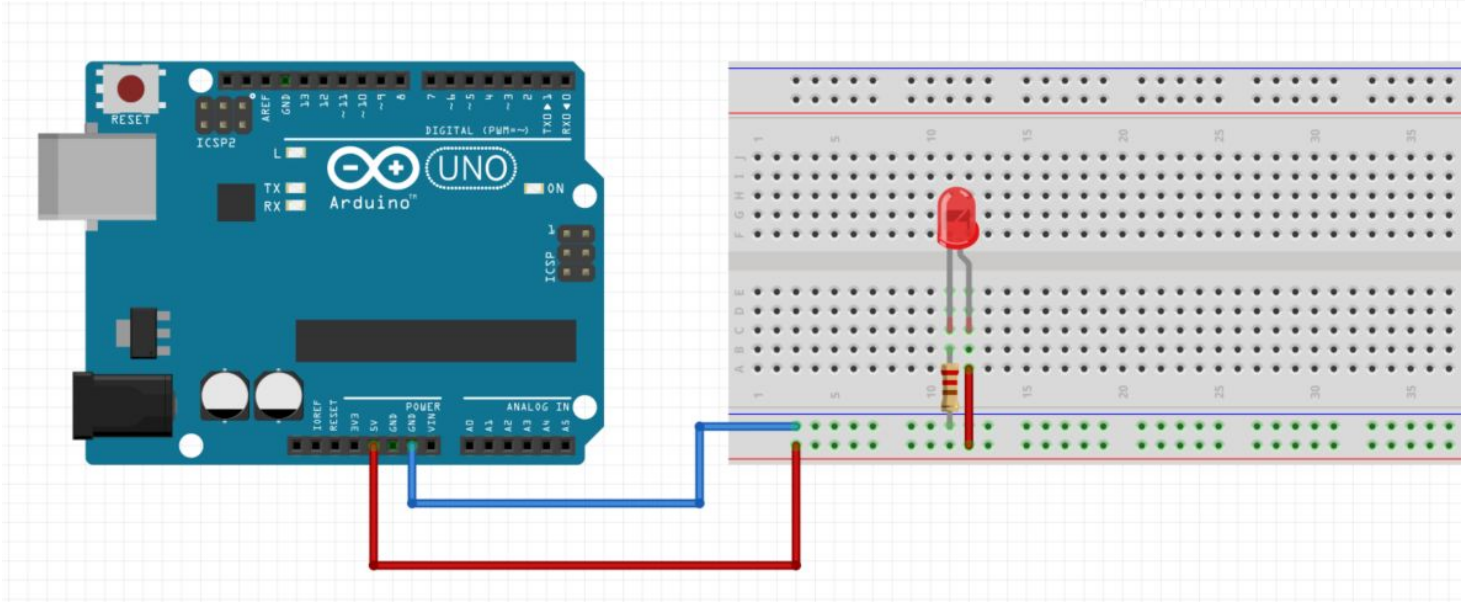
=



Fargekoder

Rød symboliserer +
Blå, sort eller annet symboliserer - (jord)

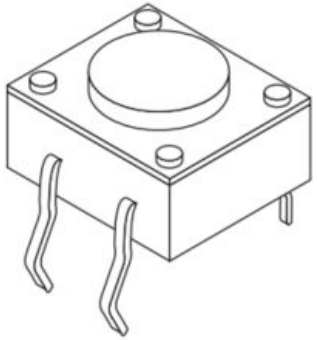


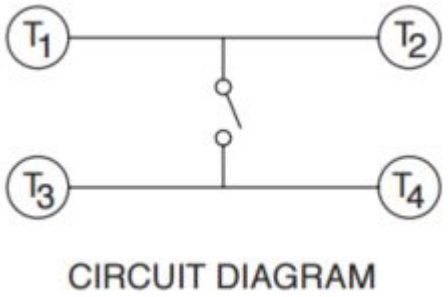


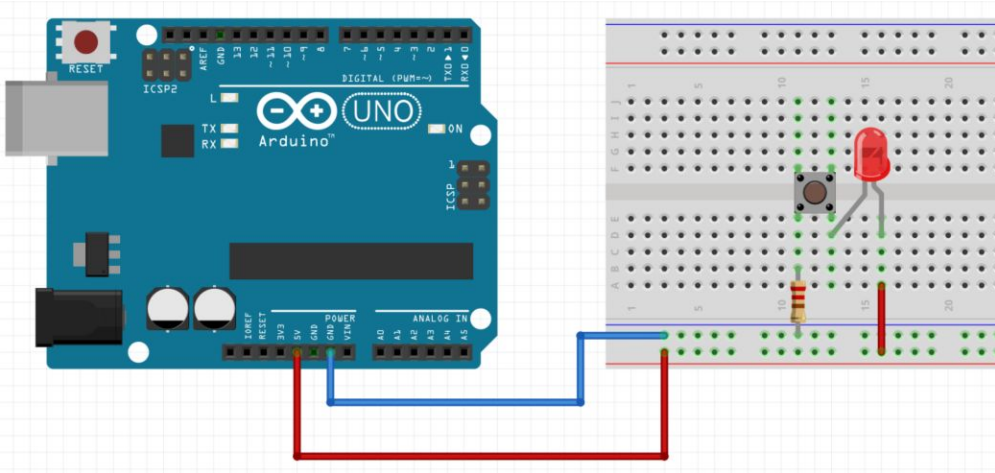
Kretser og brytere

En vanlig komponent i en krets er brytere. I starterkitet følger det med en såkalt pushbutton. Dette er en bryter som kobler sammen kretsen når man klikker på knappen.

I eksemplet til høyre er kretsen brutt, helt til vi klikker inn knappen. Da vil kretsen være komplett, og strøm vil flyte gjennom LED-pæren.







Grunnlag

Signaler

Digitale og analoge signaler

Med Arduino må vi forholde oss til to typer signaler.

Digitale, som kun består i AV og PÅ.

Analoge, som kan ha mange verdier innenfor et bestemt spekter.

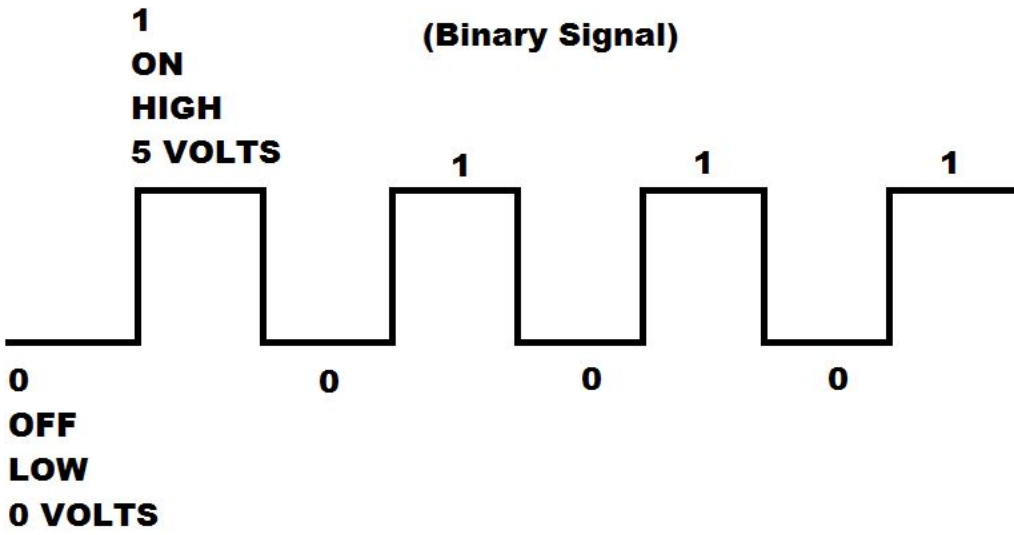
Digitale signaler

Digitale signaler kommer kun i form av binære verdier, altså AV eller PÅ.

På Arduino betyr dette enten ingen spenning (0 volt), eller full spenning (5 volt).

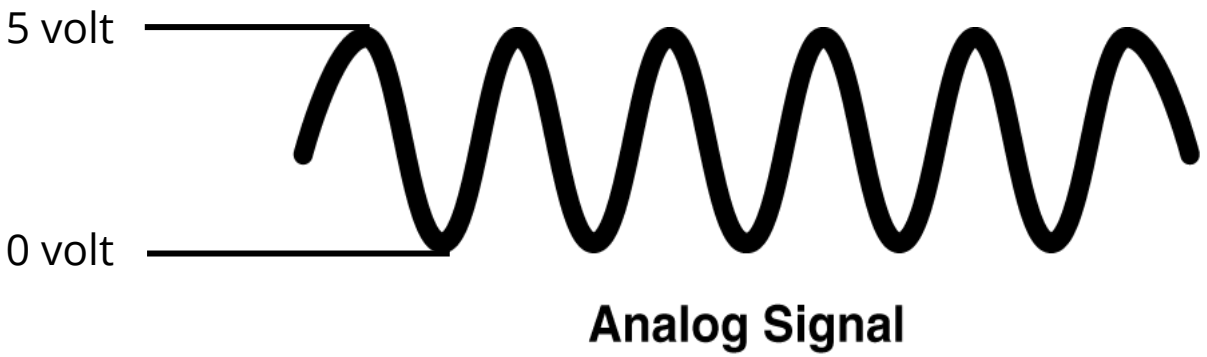
0 volt = LOW / FALSE / 0

5 volt = HIGH / TRUE / 1

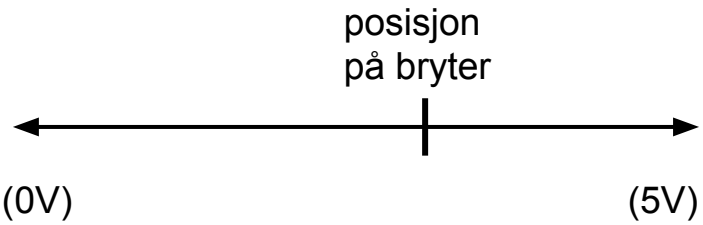


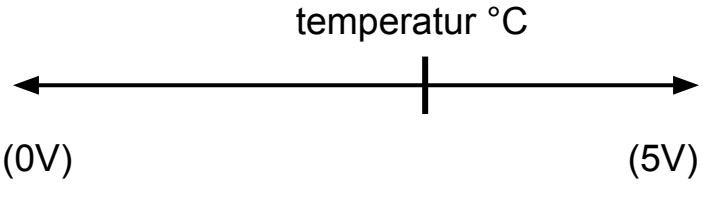
Analoge signaler

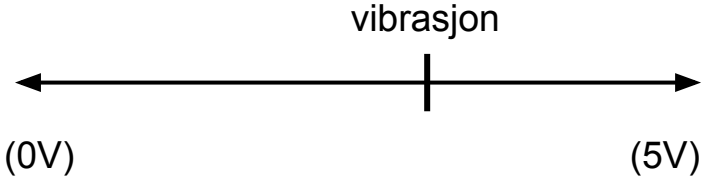
Analoge signaler kommer som spenning i varierende mengde. For eksempel alle verdier mellom 0 og 5 volt.



Eksempler på analoge komponenter i starter-kit



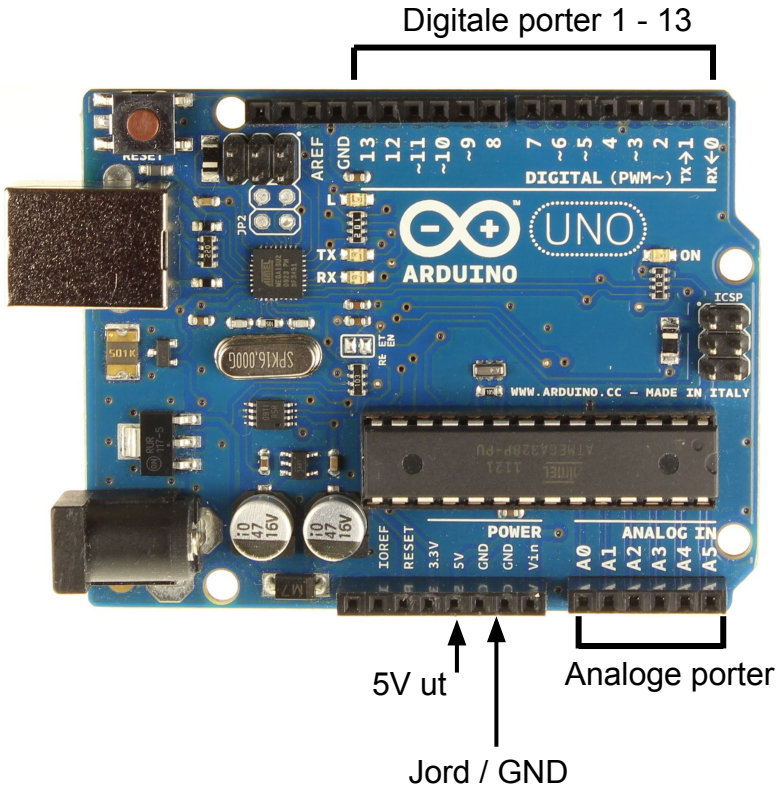




Signaler inn og ut på Arduino

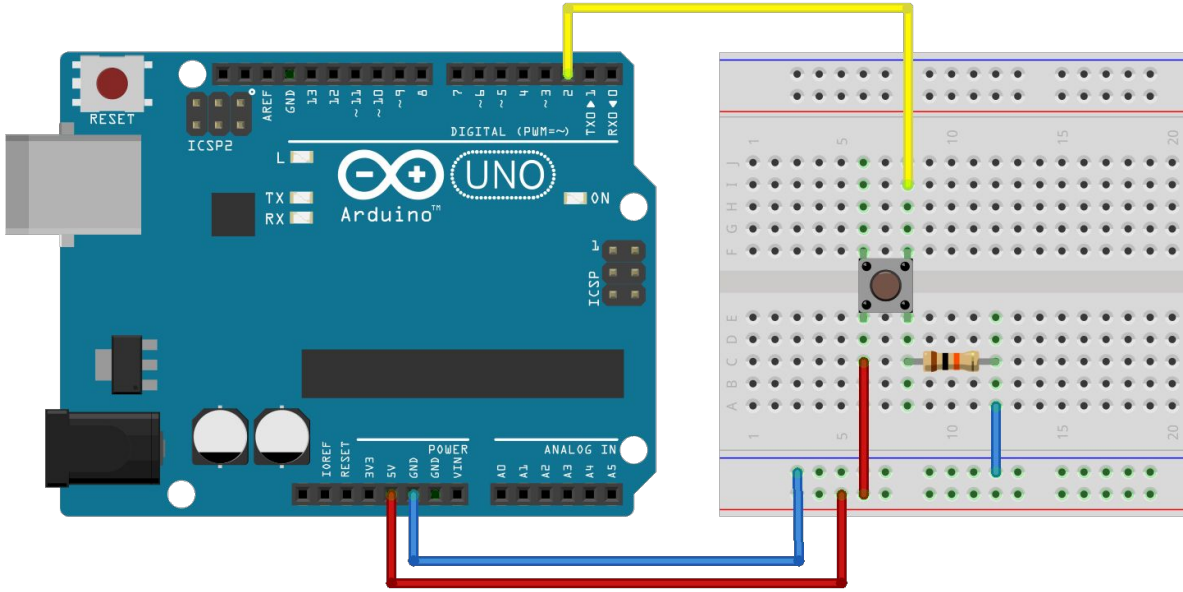
Arduino er utstyrt med mange tilkoblingsporter som både kan sende strøm ut og inn.

De øverste portene nummerert 1 - 13 kaller vi digitale, da de kan sende eller motta full spenning eller ingen spenning (av og på).



Signaler inn og ut på Arduino

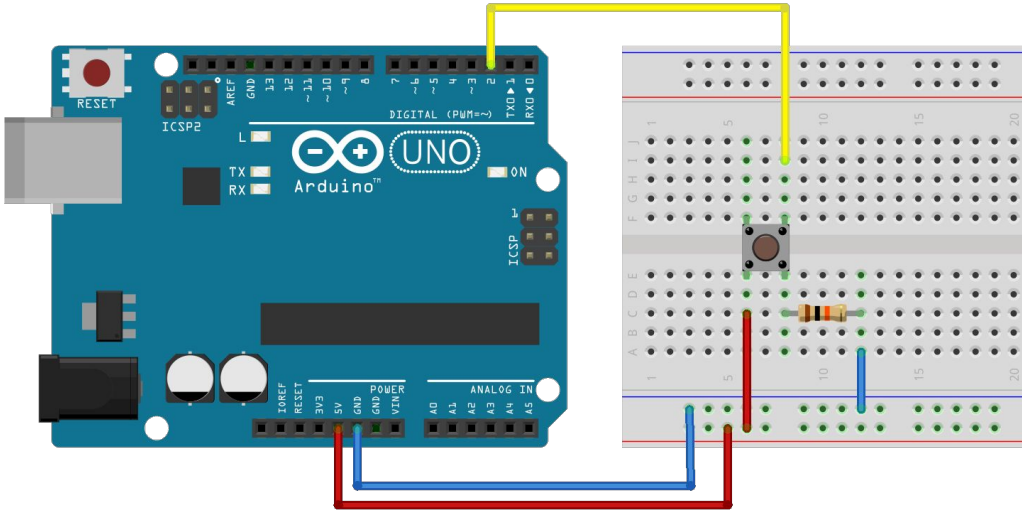
I dette eksemplet bruker vi en digital port på Arduinobrettet til å “lytte” på kretsen for å se om knappen er klikket.



Signaler inn og ut på Arduino

Ved hjelp av innebygde metoder i Arduino kan vi definere at porten skal “lytte”, og dermed merke om knappen blir klikket.

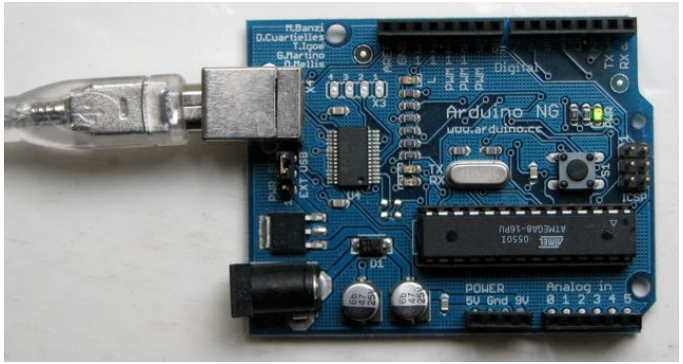
```
if (digitalRead(2) == HIGH) {

    //GJØRE NOE

}
```

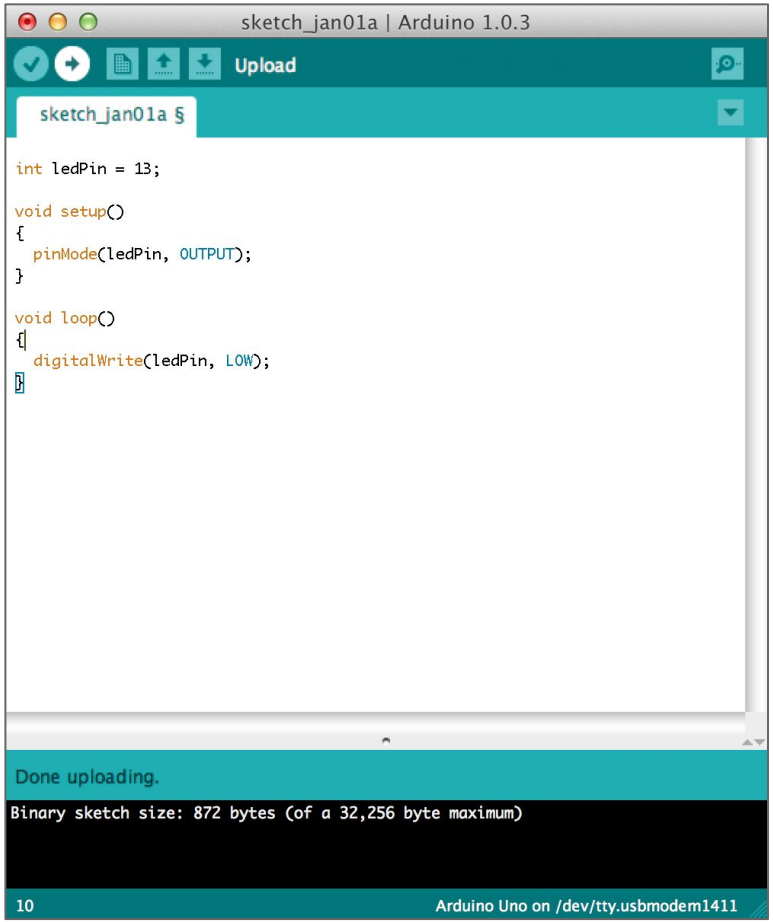


Programmering i Arduino

Komme i gang

Arduino kommer med en egen editor. Her kan vi skrive kode, og laste den opp til Arduinobrettet for testing.



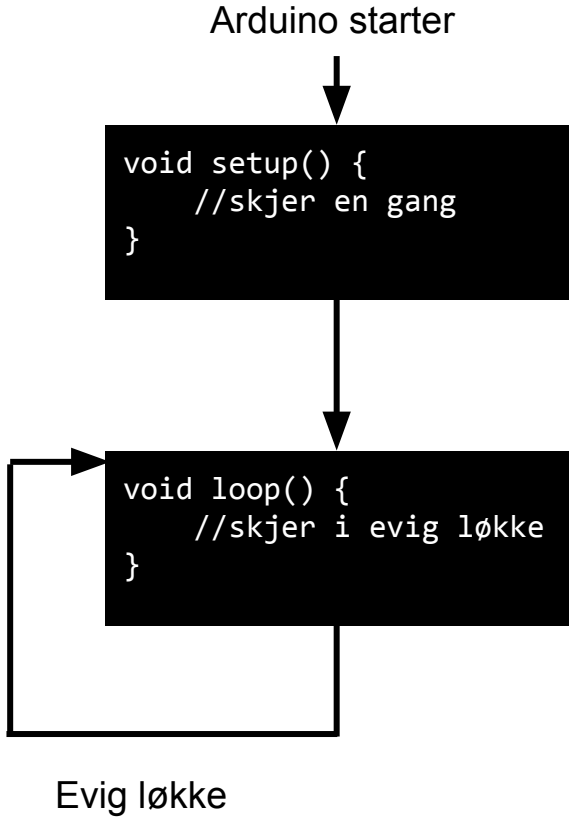


SETUP OG LOOP

Når vi programmerer Arduino tar vi utgangspunkt i to forhåndsdefinerte metoder.

setup()
Kjøres med en gang Arduino starter opp, og aldri igjen. Dette er dermed et naturlig sted å definere funksjonen til ulike porter etc.

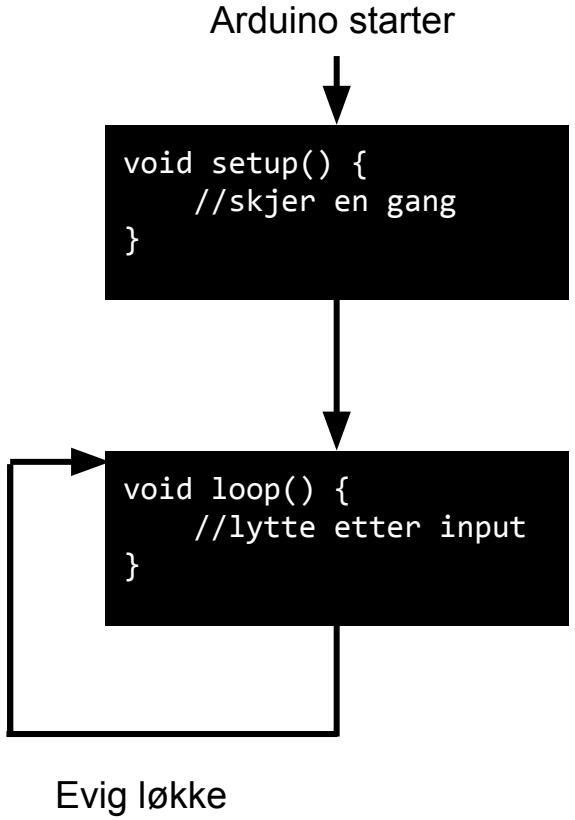
loop()
En evig løkke som kjøres igjen og igjen så lenge Arduino er på.



SETUP OG LOOP

Da loop kjører i en evig løkke vil all kode her skje igjen og igjen, noe som former måten vi programmerer på.

Løkken gjør at vi kan lytte etter signaler fra portene hele tiden.



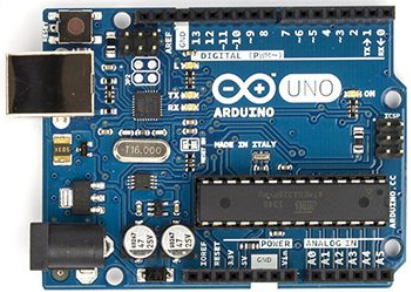
Programmering i Arduino

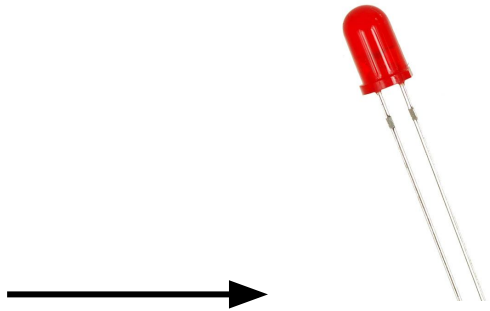
Sende digitale signaler

DIGITAL UT

Mål

Målet er å sende strøm til LED slik at den blinker. Det betyr at vi må sende strøm, og ikke sende strøm med en bestemt intervall.

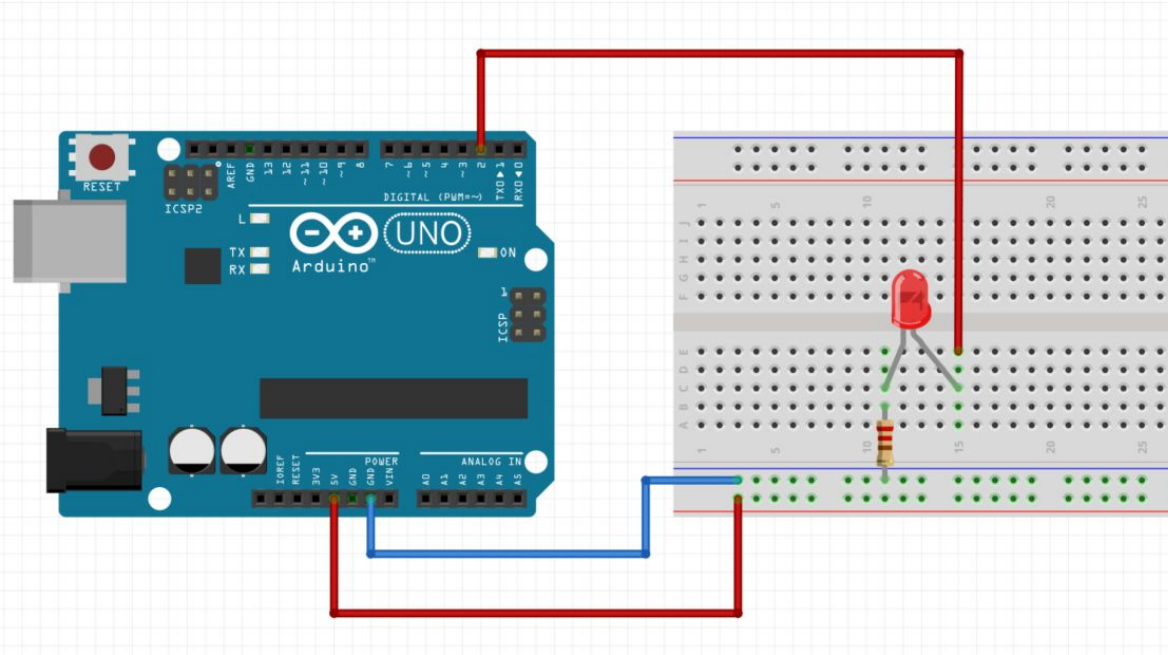




DIGITAL UT

Det er her naturlig å starte med å koble opp en LED-pære.

For å kunne kontrollere strømmen som sendes til LED, kobler vi + fra en av de digitale portene på brettet, i stedet for fra 5V som vi har brukt til nå.

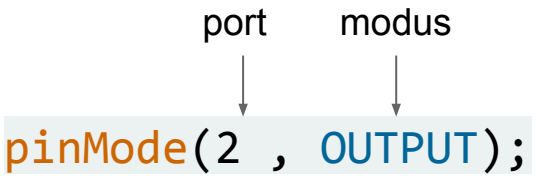


DIGITAL UT

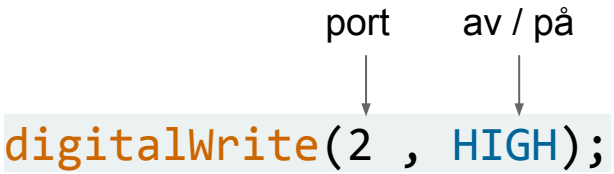
I koden må vi først fortelle Arduino at port 2 skal benyttes til å sende strøm ut.

Deretter må vi si at porten skal settes til HIGH for at strøm skal sendes.

`pinMode()` tar inn portnummer og modus.



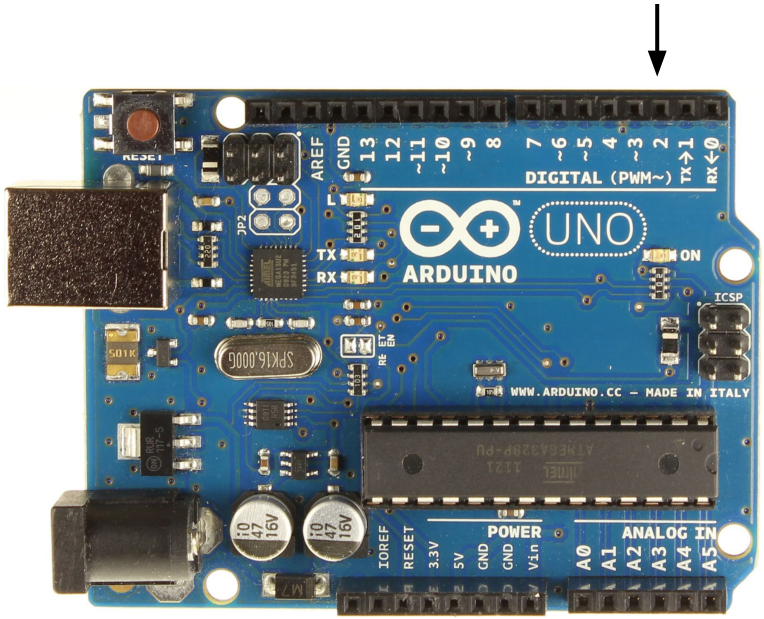
`digitalWrite()` tar inn portnummer, og nivå HIGH eller LOW.



DIGITAL UT

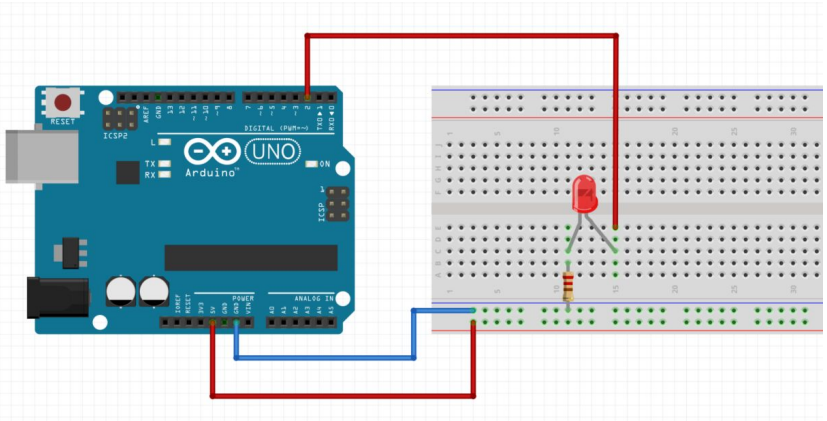
Nummeret porten er merket med på brettet korresponderer med nummeret vi gir til pinMode.

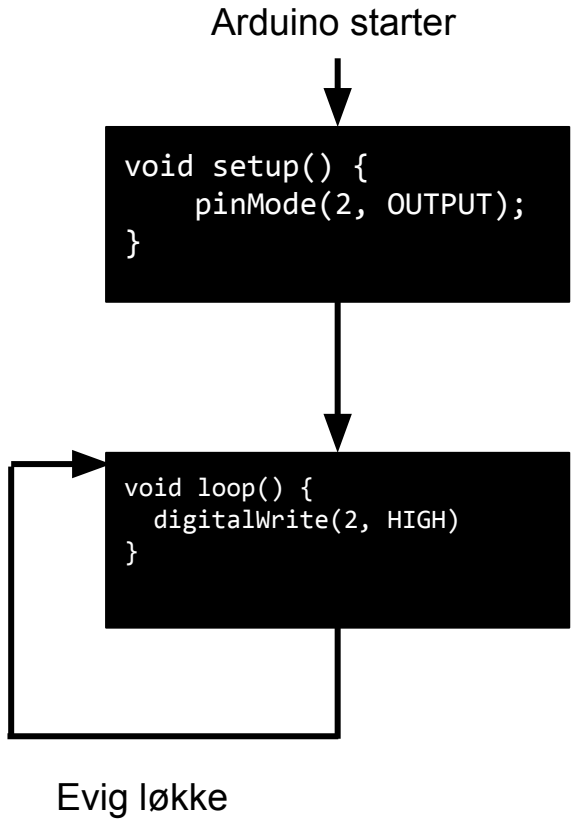
```
void setup() {
      pinMode(2, OUTPUT);
}
```



DIGITAL UT

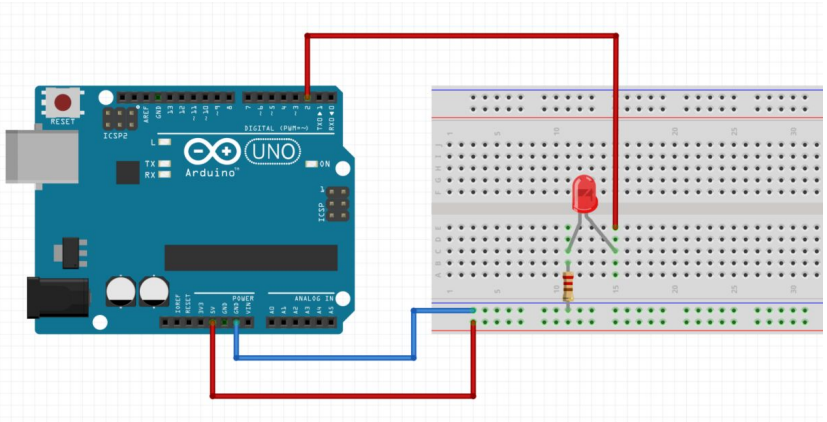
```
void setup() {
    pinMode(2, OUTPUT);
}

void loop() {
    digitalWrite(2, HIGH);
}
```

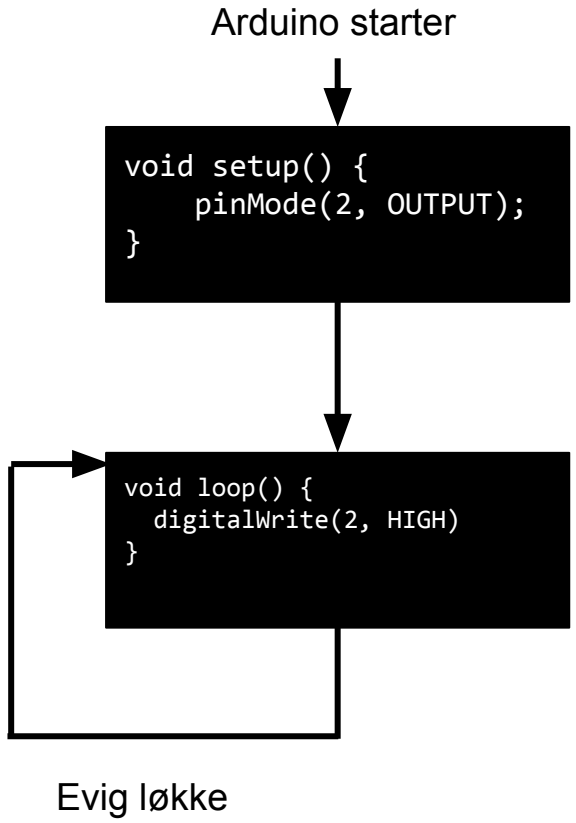




DIGITAL UT

```
int led = 2;

void setup() {
     pinMode(led, OUTPUT);
}

void loop() {
    digitalWrite(led, HIGH);
}
```



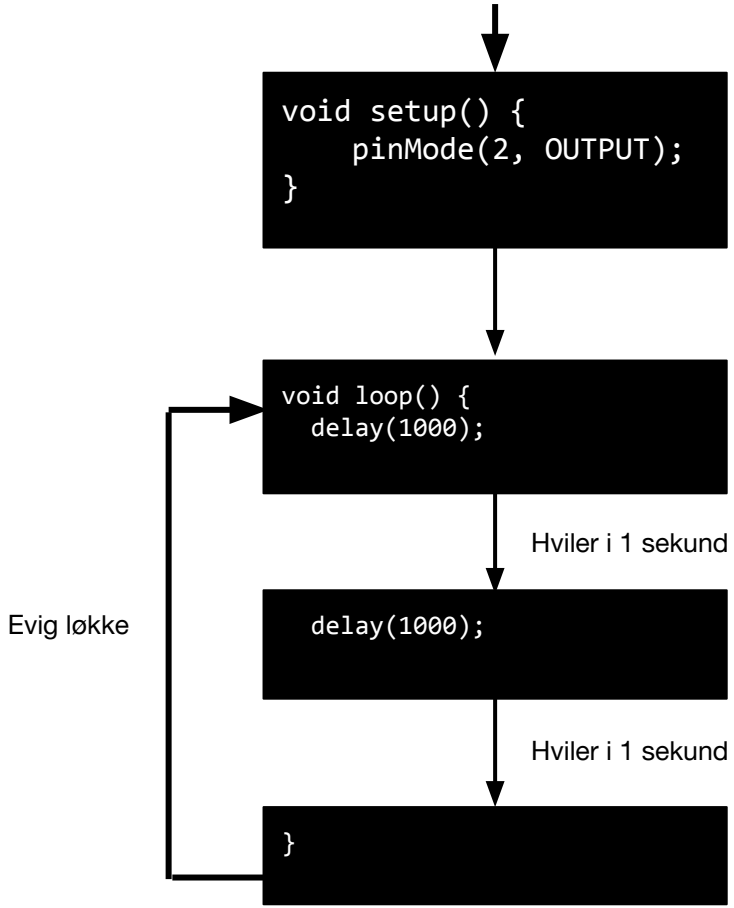


DELAY()

loop() kjører som nevnt i en evig løkke. Av og til ønsker vi et tidsbestemt opphold i denne. Da kan vi bruke metoden delay().

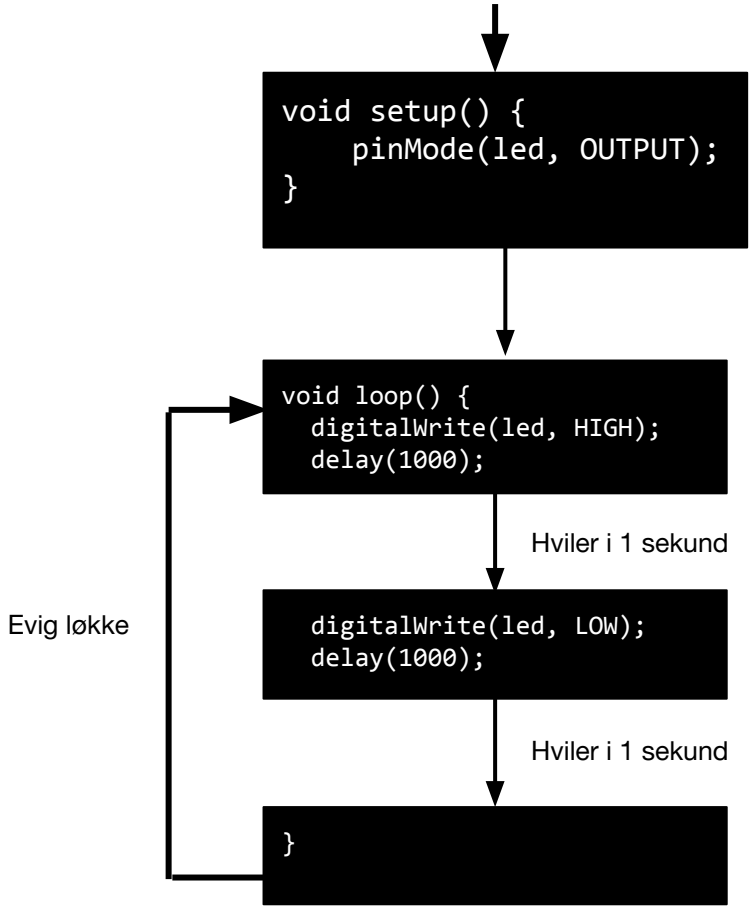
tid i millisekunder

```
delay(1000);
```



DIGITAL UT + DELAY = BLINK

```
int led = 2;
void setup() {
    pinMode(led, OUTPUT);
}

void loop() {
    digitalWrite(led, HIGH);
    delay(1000);
    digitalWrite(led, LOW);
    delay(1000);
}
```



Programmering i Arduino

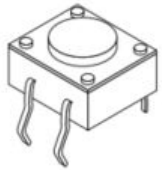
Motta digitale signaler

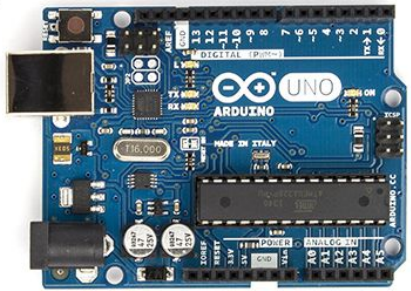
DIGITAL INN

Mål

Målet er å ta imot signaler fra en komponent. Her benytter vi en knapp.

Når signalet registreres kan vi for eksempel få en LED-pære til å lyse, eller blinke.



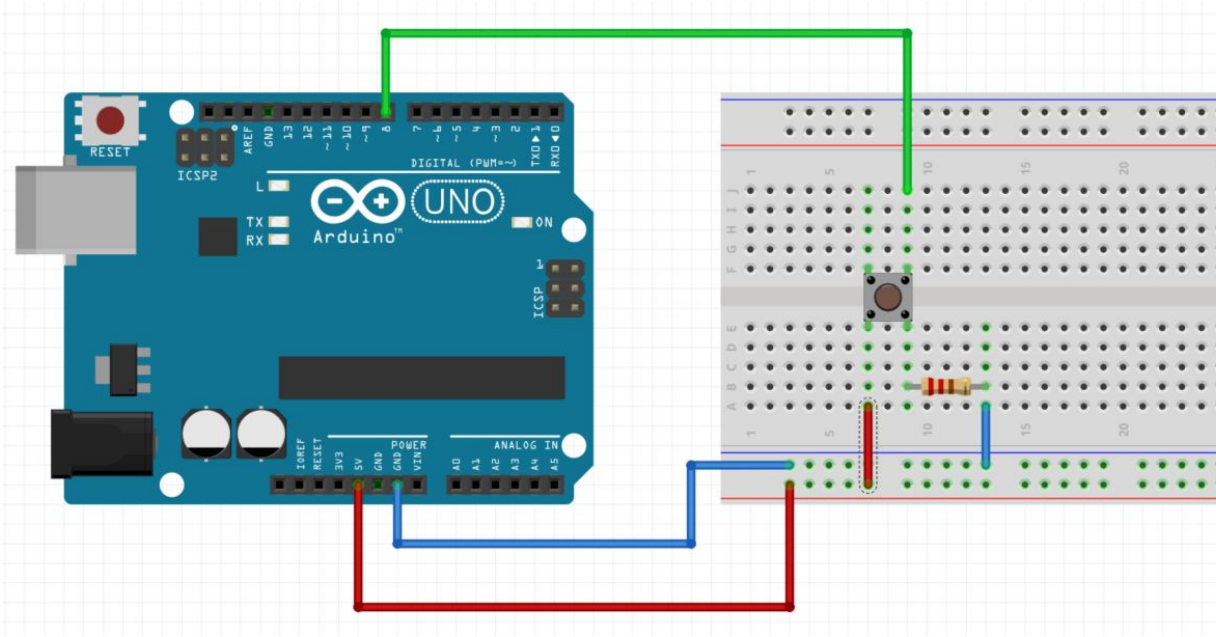


DIGITAL INN

Vi starter med å koble opp en krets med en knapp.

Når knappen klikkes vil strøm passere gjennom kretsen.

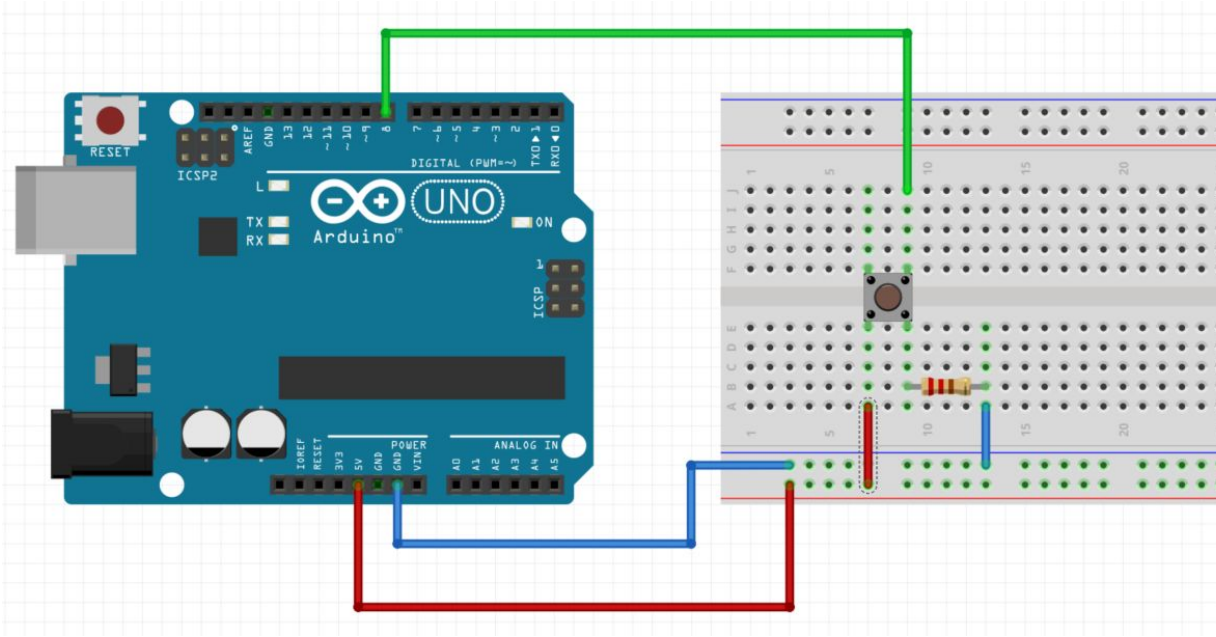
Dette kan vi måle med den grønne ledningen vi har koblet til port 8 på brettet.



DIGITAL INN

Legg merke til at hvilket punkt vi lytter på i kretsen her er viktig.

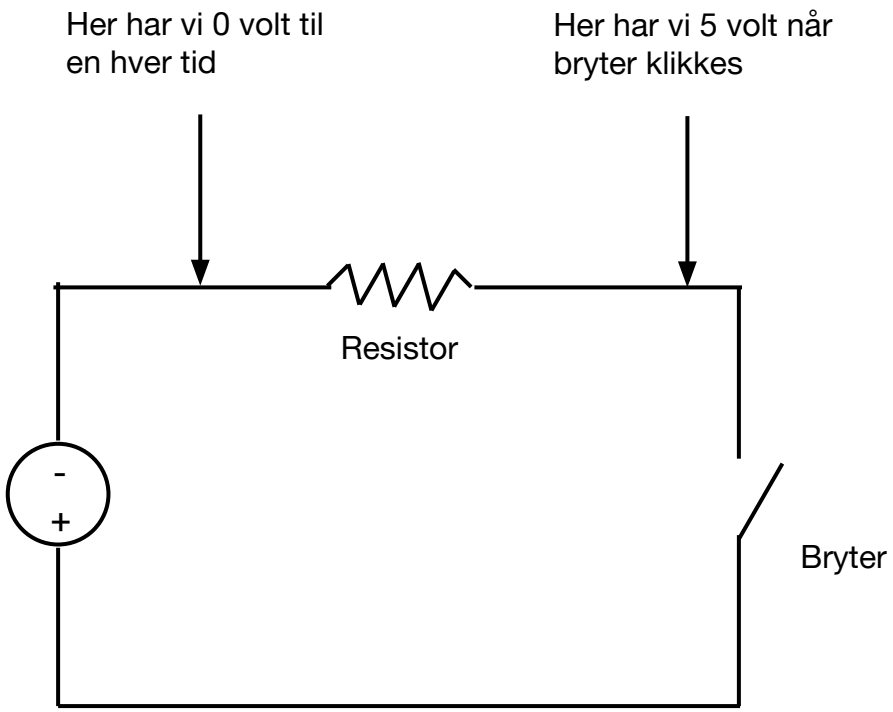
Vi ønsker å lytte etter bryteren, men før motstanden.



DIGITAL INN

På tegningen ser vi hvor man må måle.

Etter siste komponent i en krets vil det være 0 volt til en hver tid. Dermed er det viktig at vi her måler før resistoren.



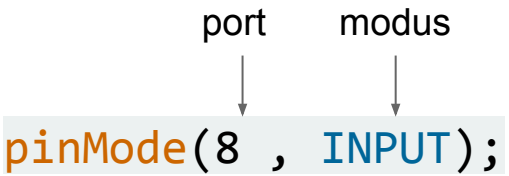
DIGITAL INN

I koden må vi først fortelle Arduino at port 8 skal benyttes til å lytte etter strøm/spenning inn.

Deretter kan vi sjekke om vi får strøm eller ikke med digitalRead().

Hvis vi får strøm (/spenningen er 5 volt) vil vi få returnert HIGH. Hvis 0 volt, får vi LOW.

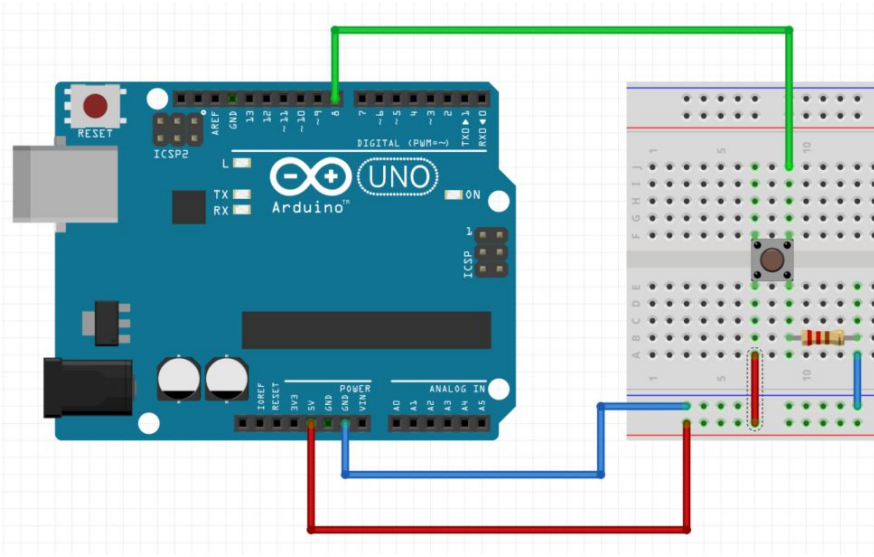
`pinMode()` tar inn portnummer og modus.



`digitalRead()` tar inn portnummer, og returnerer HIGH, eller LOW (1 og 0).

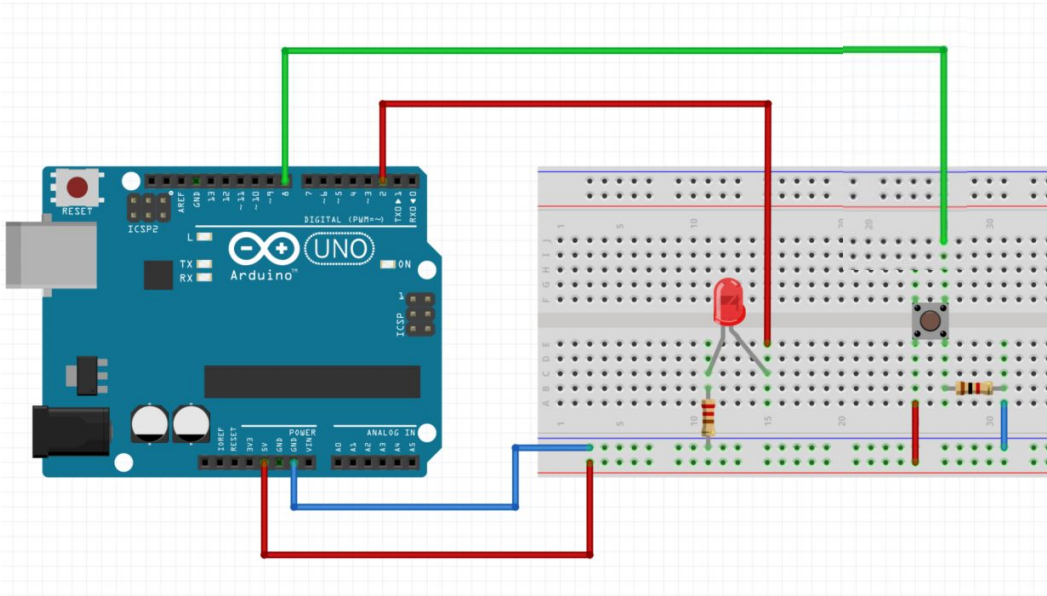
DIGITAL INN

```
int button = 8;

void setup() {
      pinMode(button, INPUT);
}

void loop() {
     if (digitalRead(button) == HIGH) {
          //GJØR NOE HVIS KNAPP HOLDES INNE
     } else {
          //GJØR NOE ANNET HVIS IKKE
     }
}
```



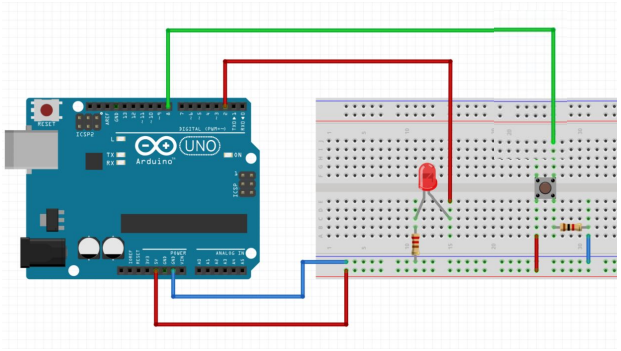
DIGITAL INN + UT

```
int led = 2;
int button = 8;

void setup() {
       pinMode(led, OUTPUT);
       pinMode(button, INPUT);
}

void loop() {

      if (digitalRead(button) == HIGH) {

            digitalWrite(led, HIGH);

      } else {

            digitalWrite(led, LOW);

      }
}
```



DIGITAL INN + UT

```
int led = 2;
int button = 8;

void setup() {
        pinMode(led, OUTPUT);
        pinMode(button, INPUT);
}

void loop() {
       if (digitalRead(button) == HIGH) {
              digitalWrite(led, HIGH);
       } else {
              digitalWrite(led, LOW);
       }
}
```



```
void setup() {
    pinMode(led, OUTPUT);
    pinMode(button, INPUT);
}
```

```
void loop() {
   if (digitalRead(button) == HIGH) {
      digitalWrite(led, HIGH);
   } else {
      digitalWrite(led, LOW);
   }

}
```

Evig løkke

Programmering i Arduino

Serial print

Serial.print()

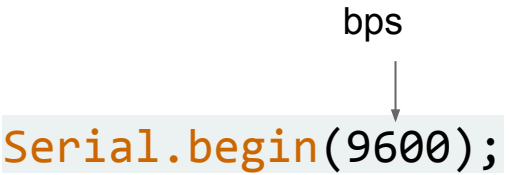
Man kan sende informasjon fra Arduino til tilkoblet PC, og vise dette som leselig tekst (liknende terminalen i java).

For å oppnå dette må man først starte en seriell sesjon med Serial.begin();

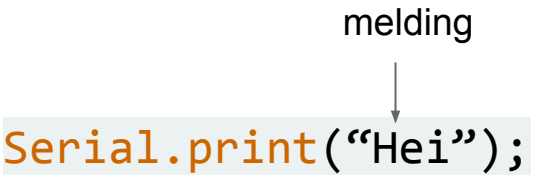
Deretter kan man benytte metoden Serial.print() for å skrive til monitor.

println() kan også benyttes for å legge inn linjeskift.

`Serial.begin()` tar inn hastighet i bps

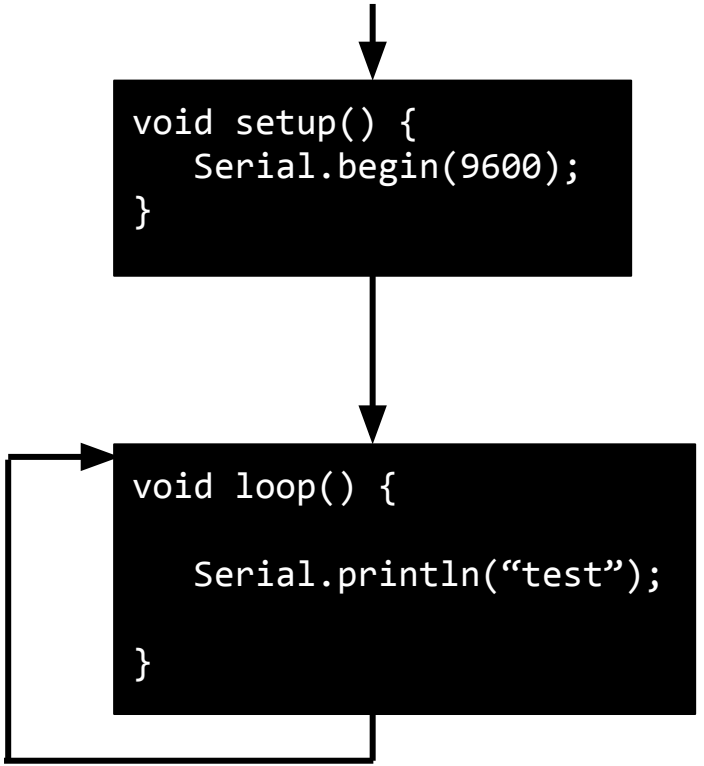


`Serial.print()` tar inn melding i de fleste datatyper



Serial.print()

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

void loop() {
   Serial.println(“test”);
}
```

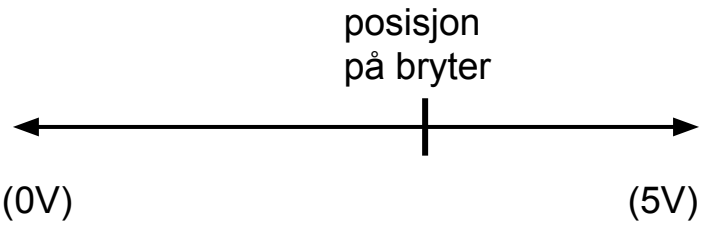


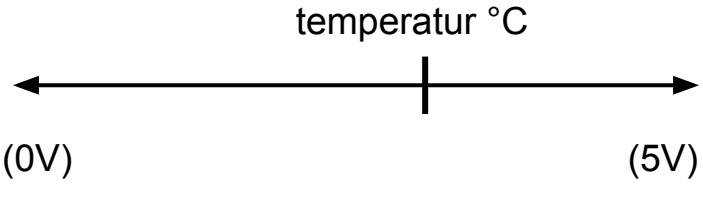
Evig løkke

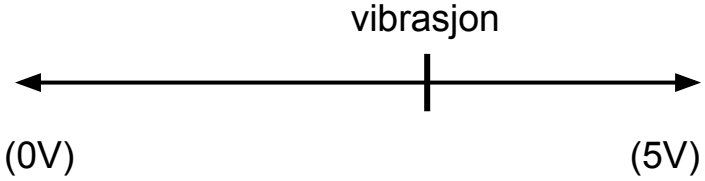
Programmering i Arduino

Motta / lese analoge signaler

Eksempler på analoge komponenter i starter-kit



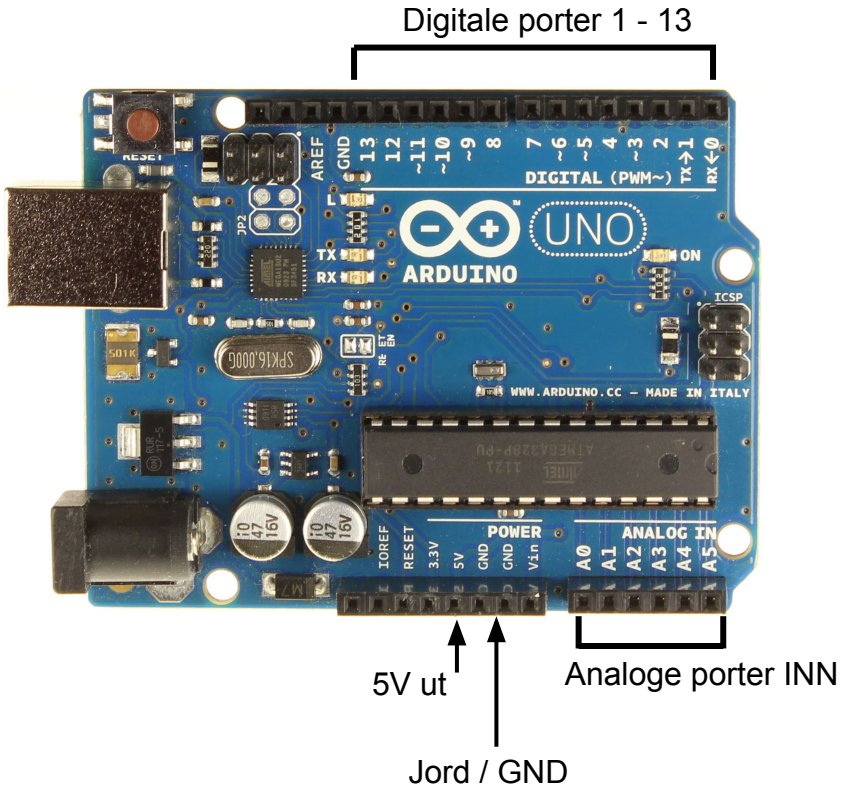




ANALOG INN

De seks portene merket A0 - A5 nederst til høyre på brettet kan ta inn / lytte til analoge signaler.

Signalene man kan få inn varierer fra 0 - 5 volt, men er mappet om til 0 - 1023, hvor 1023 er 5V.



ANALOG INN

For å lytte etter analoge signaler benyttes metoden analogRead(). Vi får da returnert en verdi mellom 0 - 1023 avhengig av spenningen den analoge porten leser.

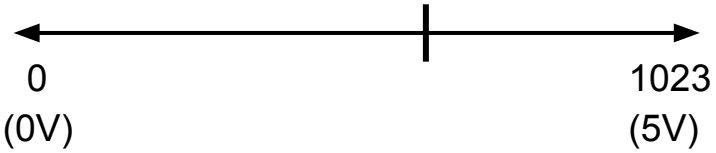
`analogRead()` tar inn portnummer og returnerer int mellom 0 og 1023

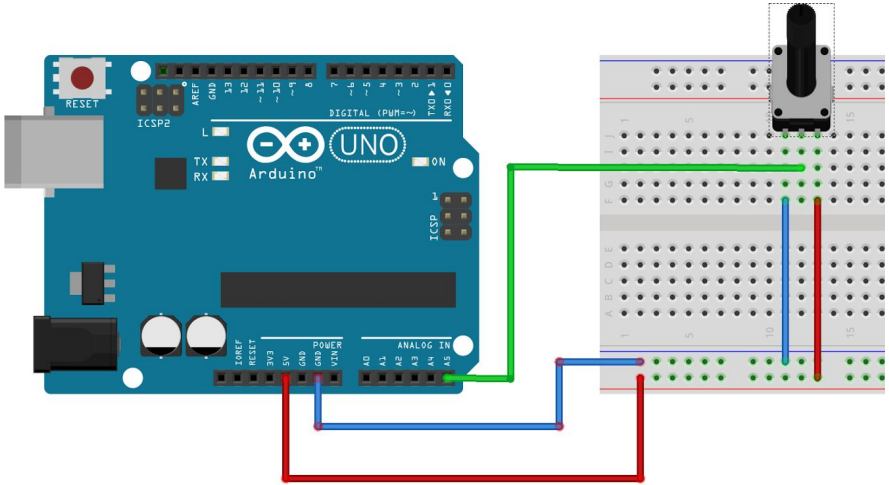
analog-port

```
analogRead(5);
```

ANALOG INN

Skala

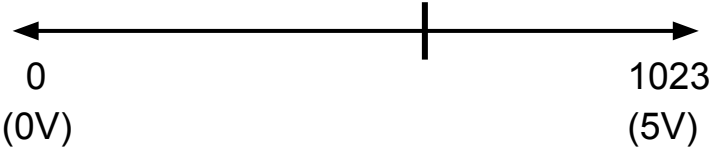


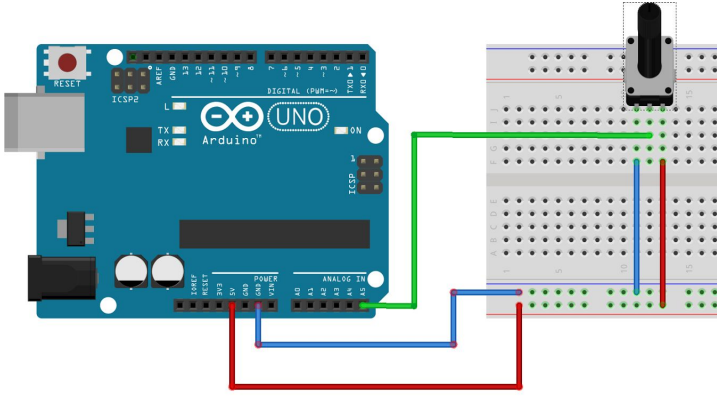



```
int potentiometer = 5;

void setup() {

}

void loop() {
   analogRead(potentiometer);
}
```

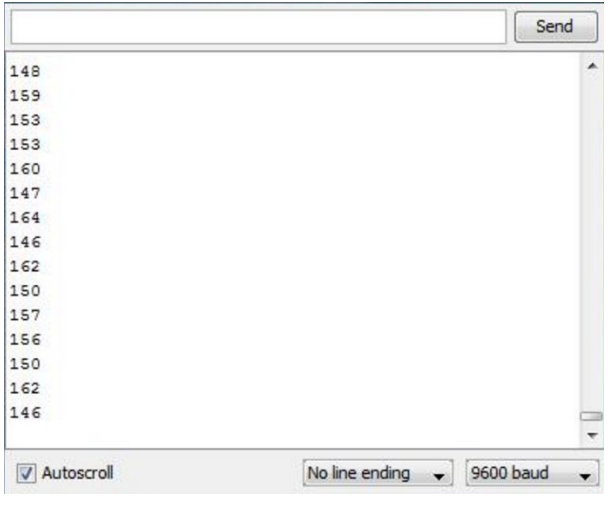
ANALOG INN

Skala






```
int potentiometer = 5;

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

void loop() {
   Serial.println(analogRead(potentiometer));
}
```



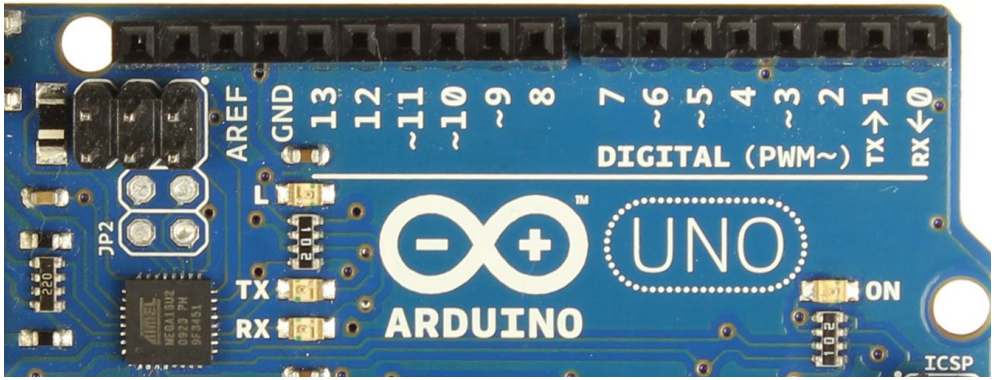
Programmering i Arduino

Sende analoge signaler

ANALOG UT

Noen av de digitale portene øverst på brettet er merket med tilde ~. Dette betyr at de kan sende analoge signaler ut.

På Arduino UNO gjelder dette port 11, 10, 9, 6, 5 og 3.



For å simulere analoge signaler bruker Arduino PWM. En teknikk hvor spenningen slås av og på fort, og gir dermed lavere spenning.

ANALOG UT

For å styre analoge signaler ut, benyttes metoden analogWrite().

Her må vi angi aktuell PWM-port, og styrke fra 0 (av) til 255 (full styrke).

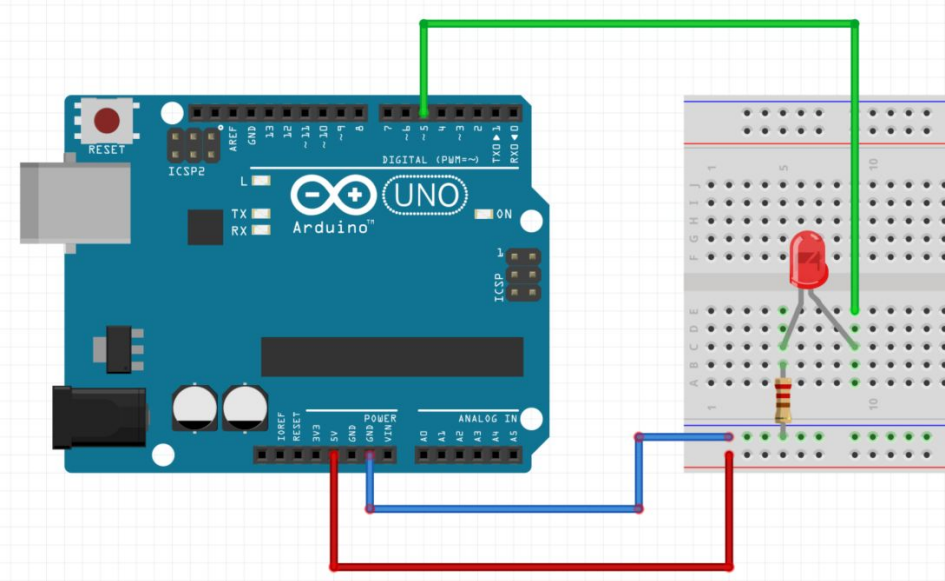
`analogWrite()` tar inn portnummer og styrke.

pwm-port styrke

```
analogWrite(5 , 150);
```

ANALOG UT

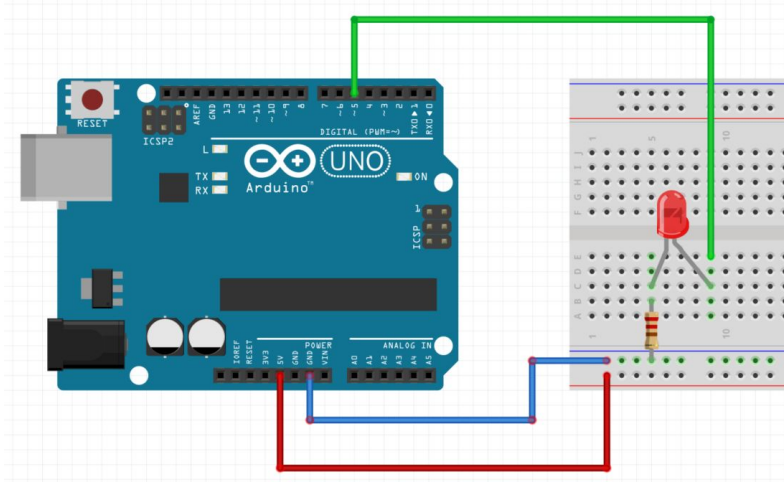
```
int led = 5;
int lightStrenght = 0;

void setup() {
    pinMode(led, OUTPUT);
}

void loop() {
   if (lightStrenght >= 255) {
      lightStrenght = 0;
   }

   analogWrite(led, lightStrenght++);
   delay(4);
}
```

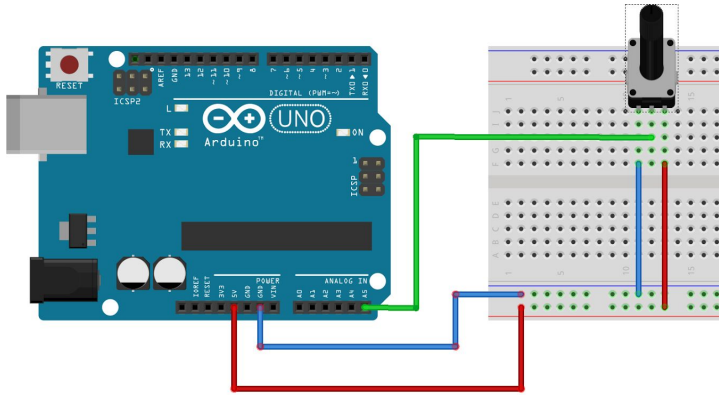


ANALOG UT OG INN

I dette eksemplet leser vi verdi vi får fra potensiometeret, og bruker denne til å bestemme hvor mye som skal sendes ut til led. Vi må dele på 4 for å gjøre om fra skala på 0 - 1023 til 0 - 255.

```
int led = 5;
int lightStrenght;
int potentiometer = 5;

void setup() {
   pinMode(led, OUTPUT);
}

void loop() {
   lightStrenght  = analogRead(potentiometer) / 4;
   analogWrite(led, lightStrenght);
}
```





Videre

Forelesning neste uke

Mandag 05.02

Arduinoprogrammering

- Tidsutsettelse
- Problemer med knappelytting
- Parallell og serie-kobling
- Kodeskikk / modularisering

Veien videre

Arduinoundervisningen

Forelesninger
Mandag 05.02
Mer Arduino

Teknisk verksted **(rom C)**
Mandag **idag**, 05.02, 12.02 & 19.02
Hjelp til ukesoppgaver

Gruppetimer
Uke 5 & 6
Hjelp til ukesoppgaver

Obligatoriske oppgaver

1) Frist 09.02
   Utvalgte ukesoppgaver skal leveres

2) Frist 23.02
   Miniprosjekt skal leveres