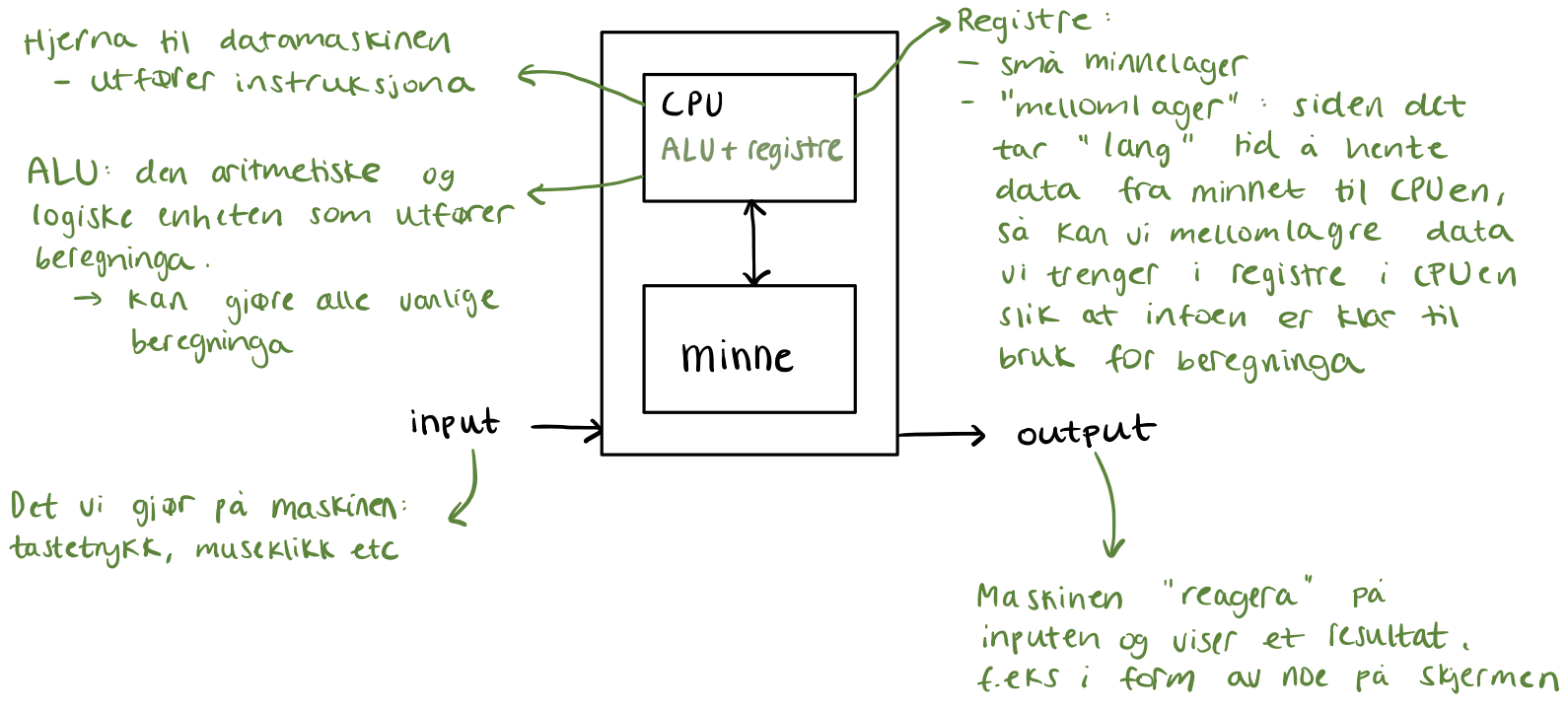
Repetisjon programvare

Om prosessoren:

Von Neumann arkitektur: En modell for hvordan en datamaskin er bygd opp.
- Mange moderne datamaskiner er basert på denne datamaskinarkitekturen.
- Inneholder mange av komponentene en datamaskin inneholder: input / output, CPU (ALU og registre) og minne
- Fordelen: data og instruksjoner er lagret i samme minne



Moores lov: antall transistora i en krets dobler seg annethvert år

Eksekveringsløkke: Hvordan prosessoren prosesserer en instruksjon. Det er på en måte alle stegene prosessoren går gjennom for å utføre en instruksjon. En eksekveringsløkke kan være slik:

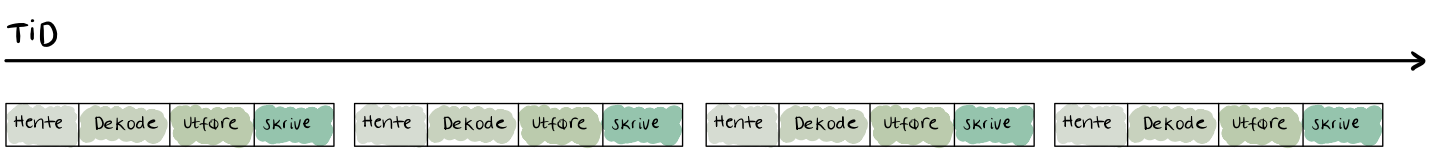


1) Hente instruksjonen i minnet
2) Dekod instruksjonen for å finne operasjonen som skal utføres og adressen i minnet
3) Utfør instruksjonen
4) Skriv svaret til minne

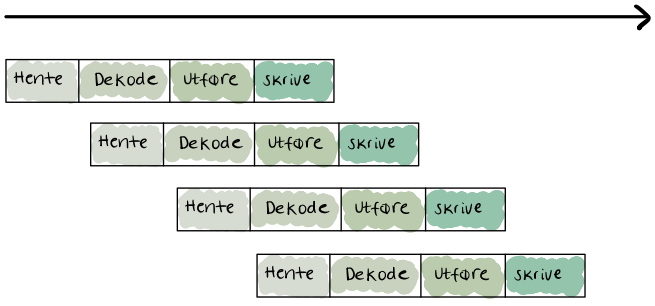
Pipelining:

- Hver del av prosessoren har en hovedoppgave: en tar seg av henting av instruksjoner fra minnet, en annen utfører instruksjonen etc.
- Hvis prosessoren skulle utført instruksjonene en og en i rekkefølge ville ikke det vært veldig effektivt og mange av komponentene i CPUen ville ventet lenge uten å gjøre noe (siden en og en komponent utfører sin del av eksekveringsløkka).
- Vi kan utnytte hele potensialet til prosessoren ved å bruke pipelining. Da vil hver del av prosessoren jobbe, men hver del jobber med en annen instruksjon.

Uten pipeline:



Med pipeline:



→ når første instruksjon er hentet, så er "hent instruksjon"-delen av prosessoren ledig, så den kan hente neste instruksjon osv....

CPU-arkitekturer: Det finnes 2 vanlige tilnærminger til prosessorarkitektur: CISC og RISC:

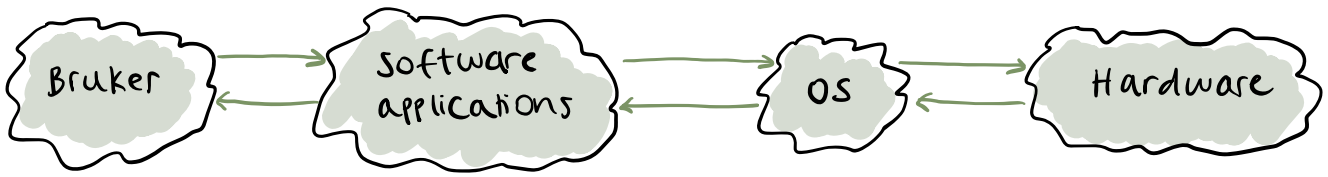
- CISC (complex instruction set computer): har kompliserte instruksjona i tillegg til enkle instruksjona
- RISC (reduced instruction set computer): har kun et begrenset antall med enkle instruksjona, men disse utføres raskt og effektivt

Operativsystemer (OS):

→ OS: et "program" som er mellomlaget mellom en bruker og datamaskinens hardware.

Hva gjør et OS?

- Håndterer all hardware og software på datamaskinen:
- Gjør grunnleggende oppgaver som håndtering av filer, minne og prosesser
- Håndterer input og output
- Sørger for at flere programmer kan kjøre samtidig
  - Hvert program trenger tilgang til CPUen og minnet, så OSet må fordele ressursene mellom programmene
- Muliggjør at en datamaskin kan ha ulike brukere
- Skjuler detaljer om datamaskinens hardware fra brukeren
- Eksempler på OS: Windows, Linux, MacOS, Ubuntu....



Hvilke hovedkomponenter inneholder et operativsystem?

Kommunikasjonstjenester:
- muliggjør kommunikasjon mellom prosesser (de kan være på samme maskin eller ulike)

Brukergrensesnitt:
- definerer hvordan brukeren interagerer med maskinen
- f. eks gjennom en terminal eller GUI

Prosessorkontroll:
- håndtere prosesser som kjører samtidig og fordeler ressurser mellom dem
- overvåker alle prosessene
- lager og sletter prosesser
- synkronisering

OS

Enhetskontroll:
- håndterer periferienheter slik at disse skal brukes av maskinen
- f. eks mus, tastatur, minnepenn eller skjerm

Minnekontroll:
- muliggjør deling av minne
- allokerer minne til programmer som trenger det - bestemmer hvilken prosess skal ha tilgang til hvilken del av minnet og hvor mye minne den får tilgang til
- de-allokerer minne når et program er ferdig

Filkontroll:
- håndterer filer
- muliggjør for brukeren å slette, lage, flytte eller manipulere filer

Brukernivå vs. kjernenivå: Det er vanlig at et operativsystem skiller mellom brukernivå og kjernenivå.

- Vanlige programmer kjører i brukernivå, og har dermed ikke så høye privilegier i systemet. De har begrenset tilgang til f.eks minnet og prosessoren og kan ikke direkte kontakte maskinvaren.

- OSet kjører på kjernenivå og dermed er mest priviligert. Den har tilgang til hele minnet, alle instruksjoner kan bli utført og går forbi alt av sikkerhet.