Gruppetime 10
30.10.2025



Sørg for at alle sitter ved siden av noen



Velkommen til gruppetimen! ☺

Plan

- Litt repetisjon om abstraksjonsnivåer
- Datamaskinarkitektur og hastighet:

Instruksjonsdekoding	Datapath	Pipeline	Minnehierarki
Primær- og sekundærminne	Transistorer	BUS	Hit or Miss i Cache og RAM
Trade-off	Koherens		

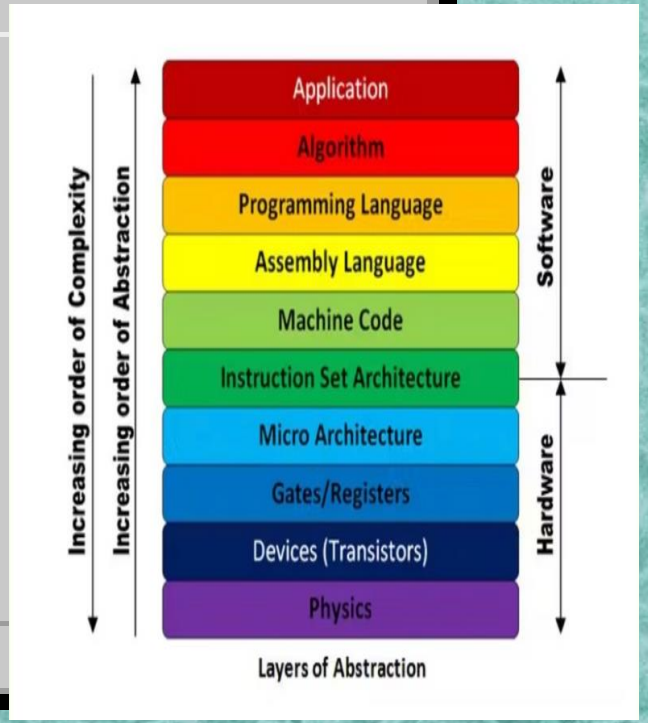
Repetisjon

Repetisjon

- Abstraksjonsnivåer: Håndtere kompleksitet

- Håndtere ett nivå om gangen
- Gjenbruke komponenter
- Endre på ett nivå uten å påvirke de andre

- Kompromiss: Høyere abstraksjonsnivå forenkler – men skjuler detaljer om ytelse og ressursbruk



Instruksjonsdekoding

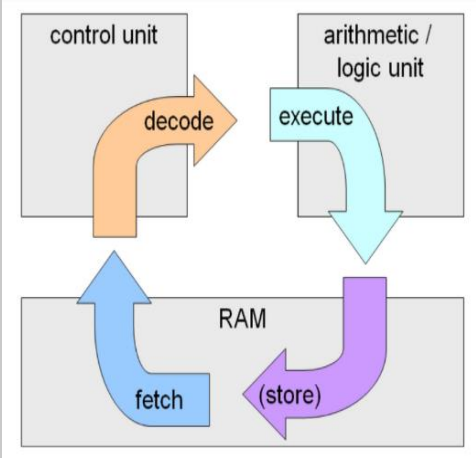
- Intruksjonskode: En binær representasjon av en kommando som forteller prosessoren hvilken operasjon den skal utføre
- Instruksjonsdekoding: Prosessen der maskinkodeinstruksjoner tolkes og utføres av prosessoren

1. Intruksjonskode (10110000 01100001)
2. Sendes til prosessoren
3. Dekoding: prosessoren («MOV AL, 97») «flytt verdien 97 inn i registeret AL»

Instruksjonssykel

Intruction Fetch → Decode → Execute → Write Back

- **Instruction Fetch (IF):** CPU henter neste instruksjon fra minnet.
- **Instruction Decude (DE):** Instruksjonen som ble hentet dekodes for å bestemme hvilken operasjon som skal utføres
- **Execute (EX):** Instruksjonen utføres
- **Write Back (WB):** Resultatet av operasjonen skrives tilbake til register eller minnet



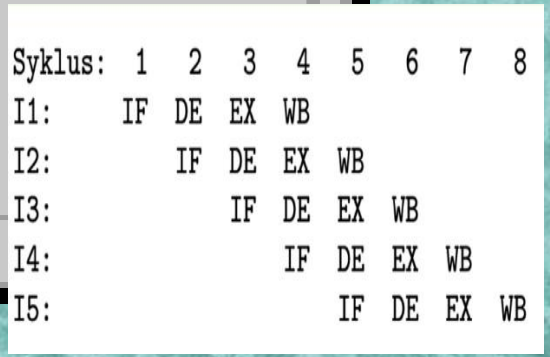
Pipeline

- Pipeline: Parallell utførelse av flere instruksjoner. Deler opp prosesseringen av hver instruskjon i flere trinn som kan utføres samtidig for forskjellige untriskjoner
- 4-trinns pipeline: 1. Instruction Fetch → 2. Decude → 3. Execute → 4. Write Back

Syklus:	1	2	3	4	5	6	7	8
I1:	IF	DE	EX	WB				
I2:		IF	DE	EX	WB			
I3:			IF	DE	EX	WB		
I4:				IF	DE	EX	WB	
I5:					IF	DE	EX	WB

4-trinns pipeline

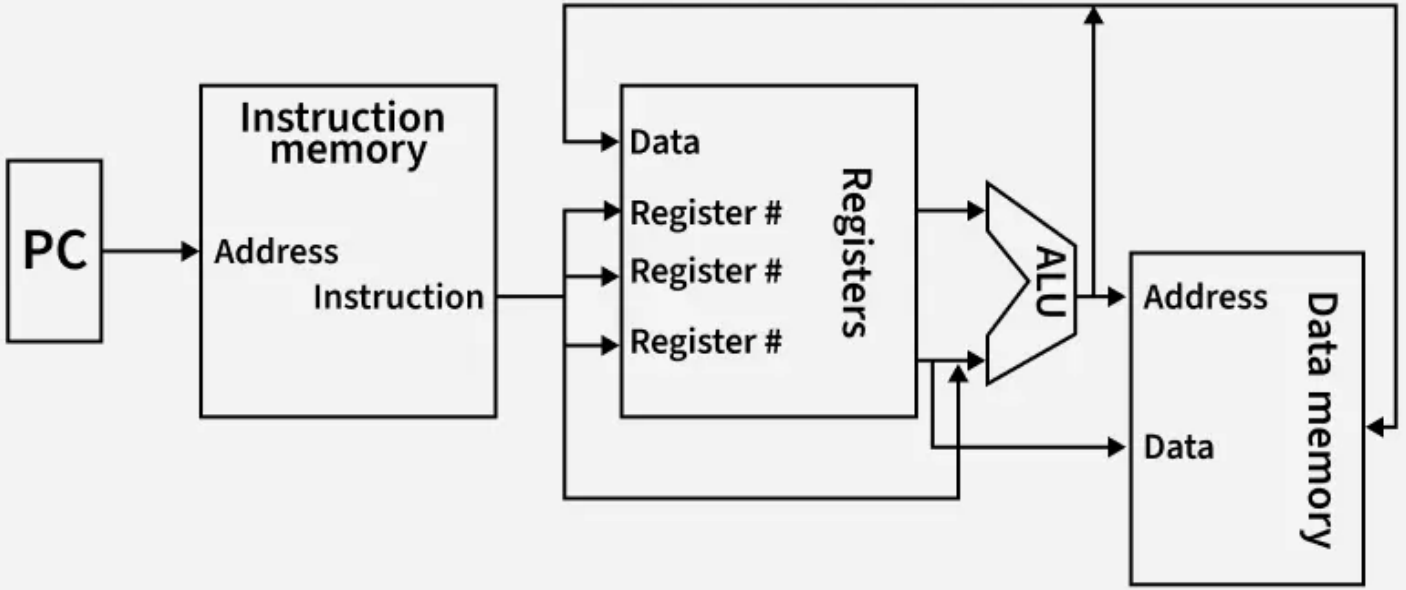
- Fordeler: Økt gjennomstrømning, høyre ytelse, bedre utnyttelse av maskinvareressurser, mulighet for høyere klokkehastighet
- Ulemper:
- Datahazards → Instruksjon er avhengig av resultatet fra en tidligere instruksjon som ennå ikke er fullført
- Kontrollhazards → branches pipeline tømmes og fylles på nytt
- Strukturelle hazards → når to instruksjoner trenger samme maskinvareressurs samtidig



Datapath

- Serie av komponenter i prosessoren som data beveger seg gjennom under utførelse av en instruksjon
- **Arimetisk logisk enhet (ALU):** Utfører aritmetiske og logiske operasjoner
- **Instruksjonsregister:** Holder nåværerende instruksjon som skal utføres
- **Programteller:** Viser til adressen til neste instruksjon som skal hentes
- **Minne-grensesnitt:** Kobling mellom prosessor og hovedminnet (håndterer lesing og skriving til og fra minnet)

Single-cycle datapath

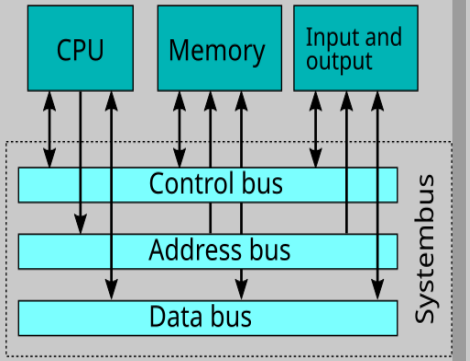


BUS

- BUSer knytter hovedkomponentene i en datamaskin sammen.
- En BUS kan sammenlignes med en motorvei for data, der biler (data) beveger seg mellom ulike filer (komponenter).
- Kan være kabelen du kobler datamaskinen til en skjerm, linjene inne i en CPU og registre, osv
- Det finnes flere typer BUSer med ulike funksjoner og hastigheter.

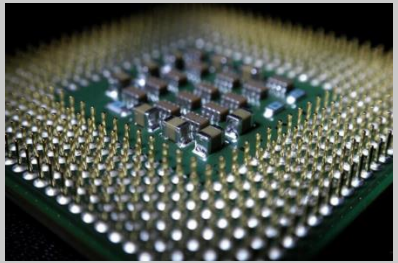
BUS

- **SystemBUS:** Hovedmotorveien som kobler prosessor, minne og andre sentrale komponenter.
- **DataBUS:** Frakter selve dataen mellom komponentene.
- **AdresseBUS:** Bærer adressen til minnet eller I/O-enheten som prosessoren vil kommunisere med.
- **KontrollBUS:** Sender kontrollsignaler som koordinerer aktivitetene mellom komponentene.
- **I/O BUS:** Håndterer kommunikasjonen mellom datamaskinen og eksterne enheter som tastatur, skjerm og mus.



Transistorer

- Transistorer er bittesmå elektroniske brytere som kan være enten av eller på
- Representerer den binære tilstanden av 0ere og 1ere. True eller False
- Det finnes milliarder av transistorer i en prosessor, som gir prosessoren ENORM regnekraft





Minnehierarki

- Utnytter prinsipper om lokalitet for å optimalisere tilgangen til data og instruksjoner
- Dette hierarkiet balanserer hastighet og kapasitet, og utnytter lokalitetsprinsippene enda mer effektivt
- **Registre**: lagrer midlertidige data og mellomresultater under utførelse av instruksjoner
- **Cache**: Fungerer som en buffer mellom prosessoren og hovedminnet
- **Hovedminne (RAM):** her aktive programmer og data lagres under utførelse
- **Sekundærlagring**: brukes til langtidslagring av programmer og data når de ikke er i aktiv bruk

Primær og sekundærminne

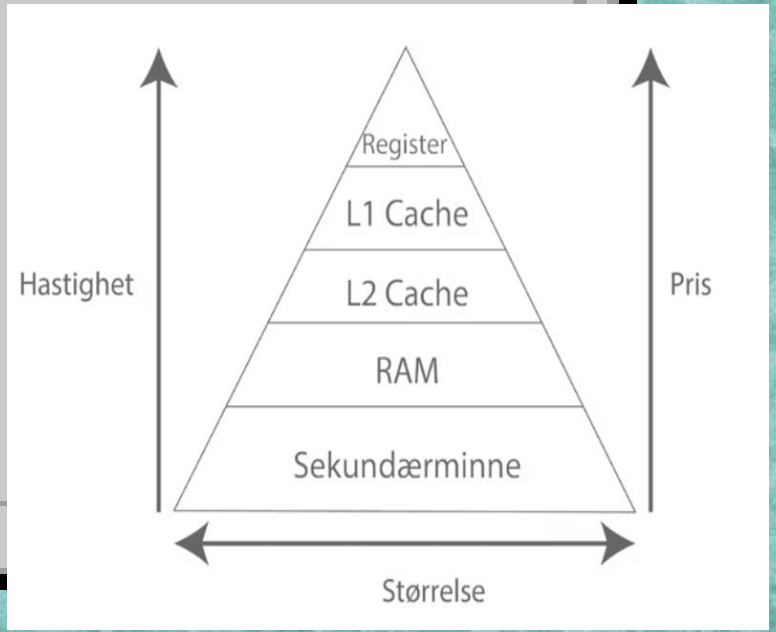
- **Primærminne**: direkte tilgjengelig for prosessoren (registere, cache, RAM)
- Rask aksess-tid
- Begrenset kapasitet
- Høyere kostnad per byte
- Mer volatile

- **Sekundærminne**: lagringsenheter (SSD, harddisker og optiske disker)
- Tregere aksesstider
- Større kapasitet
- Billigere per byte
- Non-volatile

Trade-off

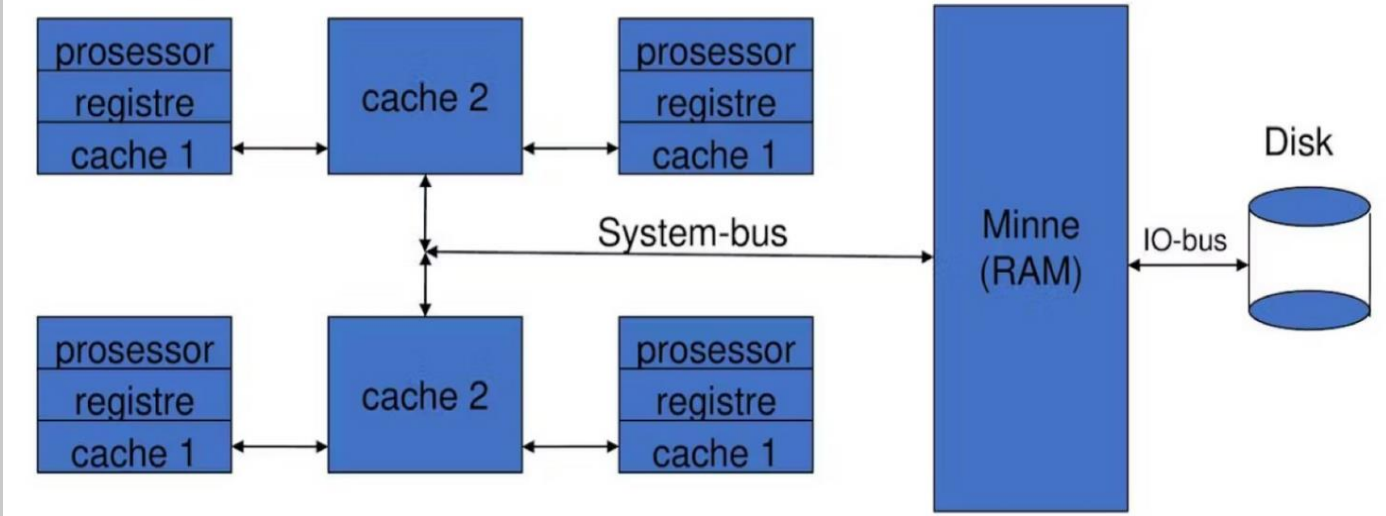
- Handler om kompromisser.
- Man kan få noe veldig bra, men må ofte noe for å få det..

- F.eks er registre kjemperaske, men har veldig lite kapasitet. De er veldig små, men inneholder milliarder av transistorer som gjør den sensitiv og kostbar å lage



Datamaskinarkitektur

Flerkjernearkitektur



Koherens

- Minnekoherens: sørger for at alle prosessorkjerner i et system ser den samme, oppdaterte versjonen av data, selv om de har hver sin lokale cache.
- Moderne arkitekturerer implementerer sofistikerte koherens-protokoller som MESI for å hondtere dette:
- **Modified**: dataen endret i en cache og ikke i minnet
- **Exclusive**: datateten finnes kun i en cache og er lik minnet
- **Shared**: dataen finnes i flere cacher og er lik minnet
- **Invalid**: dataen er ugyldig og må hentes på nytt

Cache og RAM: Hit og Miss

- CACHE + RAM: Balansegangen mellom hastighet og kapasitet
- De er forskjellige typer minne, med hver sin unike egenskaper.
- Når prosessoren trenger data, søker den først i cache
- **Cache hit:** Hvis dataen finnes der, har vi en «cache hit»
- - Ideelt, cache er EKSTREMT raskt
- **Cache miss:** Hvis dataen ikke finnes i cache, oppstår en «cache miss»
- - Systemet må da lete i neste nivå av minnet, vanligvis RAM

Spørsmål?

Jobbe med oblig? ☺