Maskinvare

av Elise

VIKTIGE BEGREPER

- Introduksjonsdekoding
  - Datapath
- Pipeline
- Minnehierarki
  - Primær- og sekundærminne
  - BUS
  - Hit or Miss I Cache & RAM
  - Kohernes

INSTRUKSJONSDEKODING

“Startpunkt for å forstå hvordan en datamaskin utfører oppgaver”

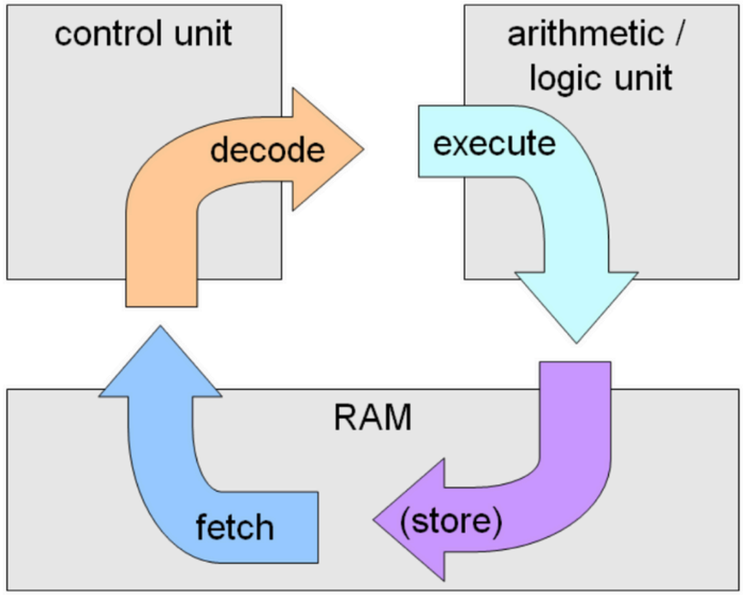
Instruksjonsdekoding: er prosessen der maskinkode tolkes og utføres av prosessoren.

Instruksjonssyklus består av fire hovedtrinn:

1. **Instruction Fetch (IF)** – hent neste instruksjon fra minnet
2. **Instruction Decode (DE)** – tolk instruksjonen og identifiser operasjon og operander
3. **Execute (EX) –** utfør beregning (aritmetisk, logisk eller adresseberegning)
4. **Write Back (WB) –** skriv resultatet tilbake til register eller minne

Prosessen gjentas for hver instruksjon i programmet

Dette ser vi når vi kjører LMC instruksjoner:)



DATAPATH

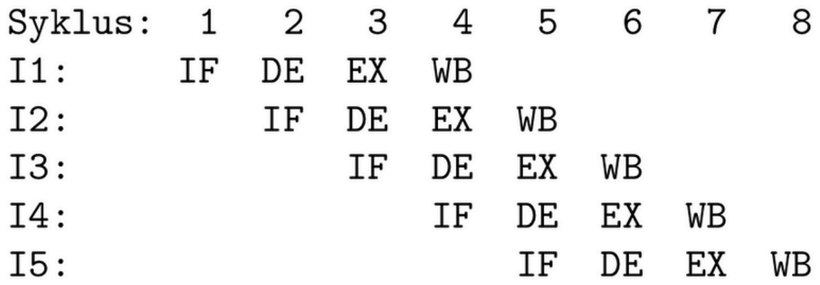
- **Datapath:** Serier av komponenter i prosessoren som data beveger seg gjennom under utførelsen av en instruksjon
- **Aritmetisk logisk enhet (ALU):** Utfører aritmetiske og logiske operasjoner
- **Instruksjonsregister:** Holder nåværende instruksjon som skal utføres
- **Programteller:** Viser til adressen til neste instruksjon som skal hentes
- **Minne-grensesnitt:** Koblig mellom prosessor og hovedminnet.
    - Håndterer lesing og skriving til og fra minnet

PIPELINE

- **Pipeline:** Parallell utførelse av flere instruksjoner. Deler opp prosesseringen av hver instruksjon i flere trinn som kan utføres samtidig for forskjellige intruskjoner
- 4-trinns pipeline
  - 1 Instruction: Fetch → 2.Decode → 3.Execture → 4.Write back

Fordeler:

- Økt gjennomstrømning, høyere 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
- Strukturell hazards → når to instruksjoner trenger samme maskinvareressurs samtidig.

HVORDAN KAN MAN HÅNDTERE HAZARDENE?
DISKUTER TO OG TO

TENK PÅ EN BILVASK SOM EN PIPLINE. HVILKE TRINN VILLE DU INKLUDERT, OG HVORDAN VILLE DETTE LIGNE PÅ EN PROSESSOR-PIPLINE? DISKUTER TO OG TO

MINNEHIERARKI

"Strukturert tilnærming til organisering av datamaskinsystemer"

MINNEHIERARKI

- Utnytter prinsippet om lokalitet for å optimalisere tilgangen til data og instruksjoner.
- Dette hierarkiet balanserer hastighet og kapasitet, og utnytter lokalitetsprinsippene enda mer effektivt.

Fire nivåer i minnehierakiet:

- **Registre:** Lagrer midlertidige data og mellomresultater under utførelsen av instruksjoner. Aksess-tid på under 1 nanosekund
- **Cache:** Cache er et raskt minne mellom prosessor og hovedminne. Den lagrer ofte brukte data for raskere tilgang. L1 er liten og rask, mens L2 og L3 er større, men tregere. 5-20 nanosekunder. Her spiller koherens en sentral rolle.
- **Hoved-minne (RAM):** Her aktive programmer og data lagres under utførelse. Aksess-tid på 50-100 nanosekunder.
- **Sekundærlagring:** brukes til langtidslagring av programmer og data når de ikke er i aktiv bruk. Aksesstid på millisekunder til sekunder.

MINNEELEMENTET I DATAMASKINARKITEKTUREN KAN VIDERE DELES INN I PRIMÆRE- OG SEKUNDÆR-MINNE

DISKUTER FORSKJELLEN TO OG TO

PRIMÆR OG SEKUNDÆRMINNE

- Primærminne: direkte tilgjengelig for prosessoren (registre, cache, RAM)
  - Rask aksess-tid
  - Begrenset kapasitet
  - Høyere kostnad per byte
- Sekundærminne: Lagringsenheter (SSD, harddisker og optiske disker)
  - Tregere aksesstider
  - Større kapasitet
  - Billigere per byte

HVA SKJER NÅR PROSESSOREN TRENGER Å LESE DATA ELLER INSTRUKSJONER

CPU (selve prosessoren ber om data)

L1-cache → sjekkes først

- Hvis hit → ferdig (data finnes her).
- Hvis miss → gå videre til neste nivå.

L2-cache → mellomnivå

- Hvis hit → data hentes herfra.
- Hvis miss → gå til L3.

L3-cache → delt mellom kjerner

- Hvis hit → data hentes herfra.
- Hvis miss → må hentes fra RAM.

RAM (hovedminne)

- Hvis hit → data hentes fra RAM.* Hvis miss → data må lastes inn fra sekundærlagring (SSD/HDD).

DATAMASKINARKITEKTUR <–> MINNEHIERARKI

BUS

BUSer knytter hovedkomponentene i en datamaskin sammen

“En BUS er en motorvei for data, hvor biler (data) beveger seg mellom ulike filer (komponenter)”

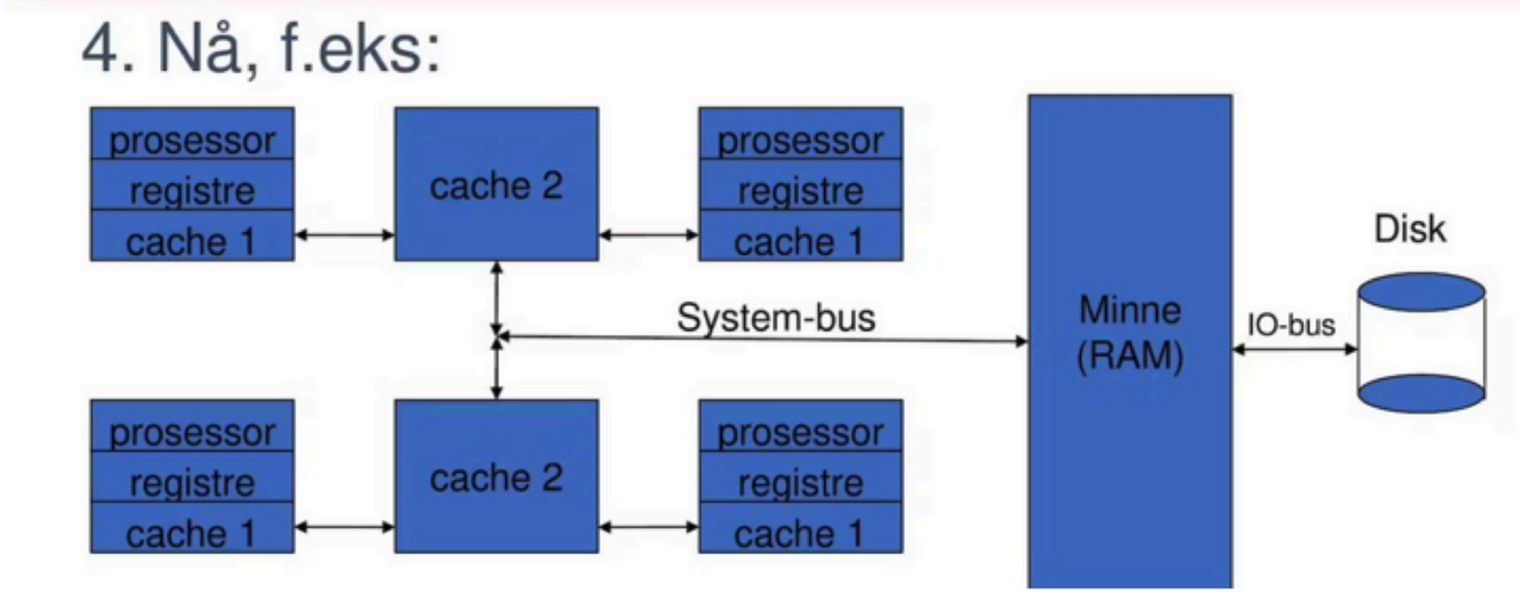
Eks:

1. Kan være kabler du kobler datamaskin til en skjerm, linjene inne i en CPU og registre

Flere typer:

- SystemBUS: er hovedveien som knytter sammen prosessor, minne og andre sentrale komponenter.
- I/O BUS: håndterer kommunikasjon med eksterne enheter.
- AdresseBUS: bærer adressen til minnet eller I/O-enheten som prosessoren vil kommunisere med.
- DataBUS flyter selve dataen mellom komponentene.
- KontrollBUS bærer signaler som koordinerer aktiitetene til ulike komponenter.

DATAMASKINARKITEKTUR



Eksamensoppgaver

Datapath 23

Velg de riktige alternativene for hvordan en instruksjon prosesseres.
HINT! husker du forelesningen "Pipeline" og "kretser som driver koden"?

Det er Velg alternativ (Cache, BUS, ALU, PC) som har adressen til neste instruksjon.

Nå værende instruksjon blir i dekode-fasen Velg alternativ (lagret, splittet opp, addert sammen, byttet).

Styresignalet til Velg alternativ (GPU, ALU, RAM, CPU, PC) kommer fra Velg alternativ (WB-steget, EX-steget, IF-steget, DE-steget).

I WB-steget sendes resultatet via en Velg alternativ (BUS, Inverter, Klokke, CPU).

Datapath 23

Velg de riktige alternativene for hvordan en instruksjon prosesseres.
HINT! husker du forelesningen "Pipeline" og "kretser som driver koden"?

Det er Velg alternativ (Cache, ALU, PC, BUS) som har adressen til neste instruksjon.

Nå værende instruksjon blir i dekode-fasen Velg alternativ (splittet opp, lagret, byttet, addert sammen).

Styresignalet til Velg alternativ (ALU, GPU, RAM, CPU, PC) kommer fra Velg alternativ (WB-steget, IF-steget, DE-steget, EX-steget).

I WB-steget sendes resultatet via en Velg alternativ (CPU, BUS, Inverter, Klokke).

Cache

Anta at det er igjen 3000 instruksjoner og at en instruksjon tar 1 klokkesykel, bortsett fra eventuelle cache-miss. Videre kan du anta at det vil være 50% cache-miss hvor det vil ta totalt 4 klokkesykel for hver instruksjon i cache-miss.

Hvor lang til vil det totalt ta?
Velg ett eller flere alternativer

- [ ] 10000
- [ ] 8250
- [ ] 9000
- [ ] 5500
- [ ] 3000
- [ ] 13250
- [ ] 5000
- [ ] 7500
- [ ] 6600
- [ ] 15000

Cache

Anta at det er igjen 3000 instruksjoner og at en instruksjon tar 1 klokkesykel, bortsett fra eventuelle cache-miss. Videre kan du anta at det vil være 50% cache-miss hvor det vil ta totalt 4 klokkesykel for hver instruksjon i cache-miss.

Hvor lang til vil det totalt ta?
Velg ett eller flere alternativer

- [ ] 7500 ✔
- [ ] 15000
- [ ] 8250
- [ ] 10000
- [ ] 9000
- [ ] 13250
- [ ] 5500
- [ ] 3000
- [ ] 6600
- [ ] 5000

Sikkerhetsmål

Sikkerhetsmål er et sentralt begrep innen informasjonssikkerhet. Hvilke av følgende defineres som sikkerhetsmål:

Velg ett eller flere alternativer

- [ ] Brannmur
- [ ] Orm
- [ ] Tilgangskontroll
- [ ] Tilgjengelighet
- [ ] Uavviselighet
- [ ] Tofaktor-autentisering
- [ ] Integritet
- [ ] Autorisering

Sikkerhetsmål

Sikkerhetsmål er et sentralt begrep innen informasjonssikkerhet. Hvilke av følgende defineres som sikkerhetsmål:

Velg ett eller flere alternativer

- [ ] Integritet ✔
- [ ] Autorisering
- [ ] Uavviselighet ✔
- [ ] Brannmur
- [ ] Tilgjengelighet ✔
- [ ] Orm
- [ ] Tilgangskontroll
- [ ] Tofaktor-autentisering

Abstraksjonsnivå H2022

Plasser disse elementene i riktig rekkefølge med hensyn på abstraksjonsnivå

Høyeste nivå: Velg alternativ (STA 04, Register, Transistor, Pipeline)

Velg alternativ (STA 04, Register, Pipeline, Transistor)

Velg alternativ (Register, Transistor, STA04, Pipeline)

Laveste nivå: Velg alternativ (Pipeline, Register, STA 04, Transistor)

Abstraksjonsnivå H2022

Plasser disse elementene i riktig rekkefølge med hensyn på abstraksjonsnivå

Høyeste nivå: Velg alternativ (Pipeline, STA 04, Transistor, Register)

Velg alternativ (Transistor, STA 04, Register, Pipeline)

Velg alternativ (Transistor, Pipeline, Register, STA04)

Laveste nivå: Velg alternativ (STA 04, Pipeline, Transistor, Register)

Minne 23

I denne oppgaven skal du arbeide med en spesifikk datamaskin som har et hierarki av minneelementer bestående av registere, en fire-nivås cache, RAM og en flash-drive (minnepenn). Maskinens minnehierarki er utformet for å håndtere forskjellige dataaksess- og skriveoperasjoner med varierende hastigheter.

I henhold til informasjon levert av produsenten, har cache-hierarkiet følgende karakteristikker:

- Leseoperasjoner tar 5 klokkesykler per nivå i cache.
- Skriveoperasjoner tar 10 klokkesykler per nivå i cache.

Videre har du informasjon om de andre minneneelementene:

- Leseoperasjoner fra RAM har en tilgangstid på 50 klokkesykler.
- Skriveoperasjoner til RAM har en tilgangstid på 100 klokkesykler.

- Leseoperasjoner fra Flash-drive (minnepenn) tar 760 klokkesykler.
- Skriveoperasjoner fra Flash-drive (minnepenn) tar 1120 klokkesykler.

HINT! For hver minneaksess operasjon, vær oppmerksom på at de oppgitte verdiene er en 'økt' tilgangstid.

Fyll ut riktig verdi i feltene (kun heltall):

(a) Hva er total antall klokkesykler i forsinkelse når du får cache-hit på nivå 3? ☐

klokkesykler

(b) Hva er total antall klokkesykler det tar å skrive en verdi som er endret i pipeline tilbake til RAM? ☐ klokkesykler

c) Anta at en optimalisering tillater at cache-nivå 1 og 2 kan oppdateres i parallell, mens de øvrige nivåene fortsatt krever sekvensiell oppdatering. Hva vil være den nye totale klokkesykkel-forsinkelsen for å skrive en verdi til RAM? _____ klokkesykler

(d) Du skal beregne hvor mange klokkesykler det totalt vil ta å gjennomføre 100 instruksjon hvor det tar 1 klokkesykel per instruksjon uten minneaksessering. Ettersom du er ekspert på å lage gode programmer og kan masse om minnehierarki har du klart å lage rekkefølgen for instruksjonene dine slik at de som aksesserer minne alltid vil ha cache-hit på nivå 1. Hvor mange klokkesykler vil det hele ta når du forventer 20% minneaksessering? _____ klokkesykler

(a) Hva er total antall klokkesykler i forsinkelse når du får cache-hit på nivå 3? (15)

klokkesykler

(b) Hva er total antall klokkesykler det tar å skrive en verdi som er endret i pipeline tilbake til RAM? (140) klokkesykler

c) Anta at en optimalisering tillater at cache-nivå 1 og 2 kan oppdateres i parallell, mens de øvrige nivåene fortsatt krever sekvensiell oppdatering. Hva vil være den nye totale klokkesykkel-forsinkelsen for å skrive en verdi til RAM? (130) klokkesykler

(d) Du skal beregne hvor mange klokkesykler det totalt vil ta å gjennomføre 100 instruksjon hvor det tar 1 klokkesykel per instruksjon uten minneaksessering. Ettersom du er ekspert på å lage gode programmer og kan masse om minnehierarki har du klart å lage rekkefølgen for instruksjonene dine slik at de som aksesserer minne alltid vil ha cache-hit på nivå 1. Hvor mange klokkesykler vil det hele ta når du forventer 20% minneaksessering? (200) klokkesykler

Ukesoppgaver

Takk for idag!