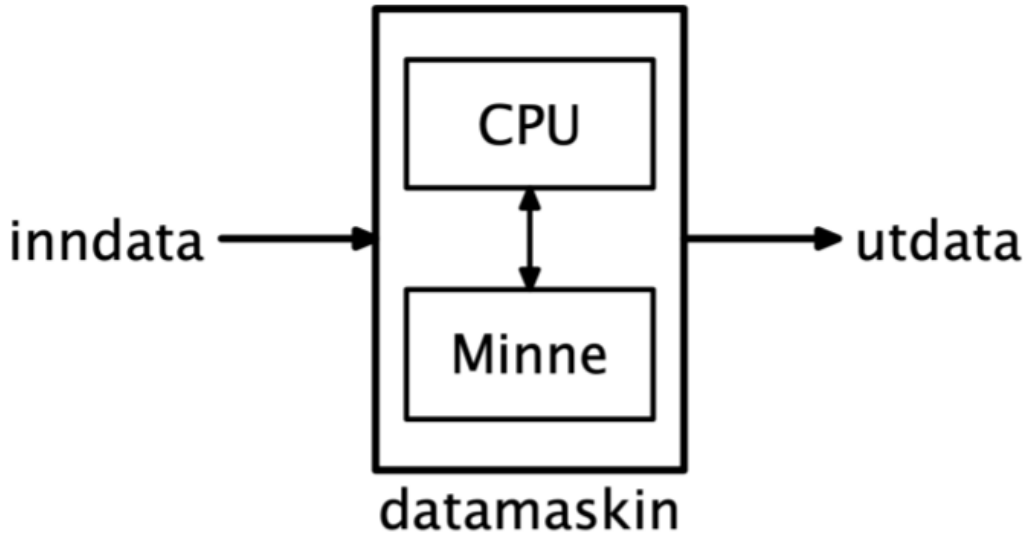
Seminar 4 (LMC 1)

Pensum

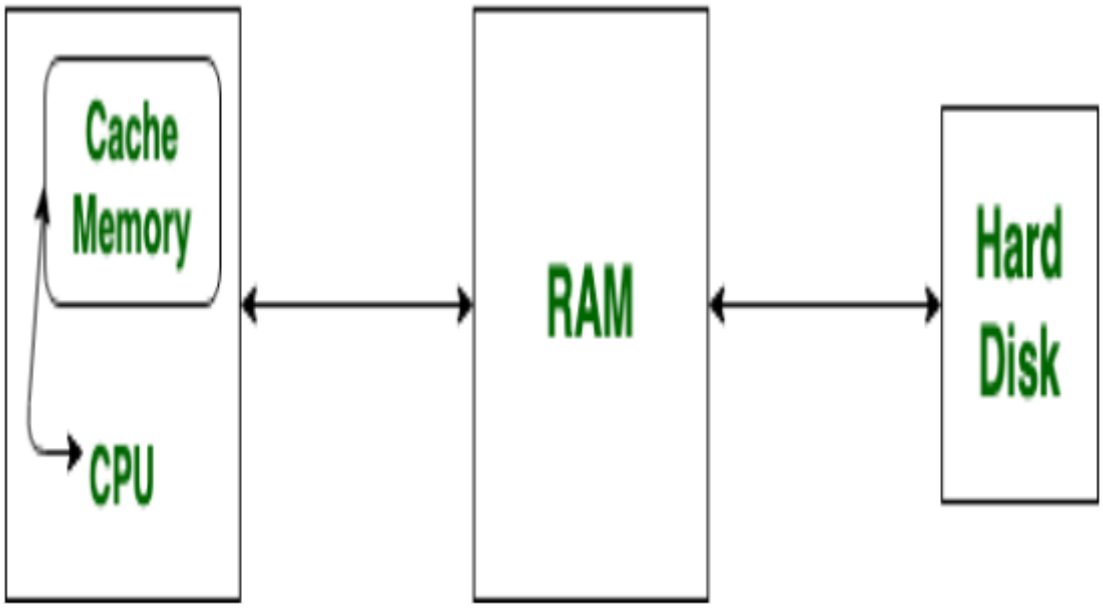
Von Neumann-arkitekturen



- Minnet holder på både *data* og *instruksjoner* til CPU-en

CPU (Central Processing Unit)

- ALU (Arithmetic Logic Unit) - holder styr på aritmetiske og logiske operasjoner. LMC sin ALU kun adderer og subtraherer tall.
- Registre - små minneceller med hurtig tilgang med hver sin unik oppgave
- Cache vs Register



LMC

LMC

ASCII tabell (både base 10 og 16)

- Alle instruksjonene til LMC:

Kode	Navn	Beskrivelse
0xx	HLT	Stopper eksekveringen
1xx	ADD	Adderer verdien i angitt minnelokasjon med akkumulatoren
2xx	SUB	Subtraherer verdien i angitt minnelokasjon med akkumulatoren
3xx	STO	Lagrer akkumulatoren i angitt minnelokasjon
4xx	-	Ikke i bruk
5xx	LDA	Henter verdi fra minnet til akkumulatoren
6xx	BRA	Hopper til angitt adresse
7xx	BRZ	Hopper til angitt adresse hvis akkumulatoren er 0
8xx	BRP	Hopper til angitt adresse hvis akkumulatoren er ≥ 0
901	INP	Leser verdi fra input, og legger svaret i akkumulatoren
902	OUT	Skriver ut verdien i akkumulatoren
922	OTC	Skriver ut ASCII-tegn (ikke i boka)

Ukeoppgaver

Oppgave 1

Tallkode representasjon:

901
399
901
398
901
198
199
902
000

Assemblerkode representasjon:

```
INP // Les tall 1.
STO 99 // Lagre tall 1 i celle 99.
INP // Les tall 2
STO 98 // Lagre tall 2 i celle 98.
INP // Les tall 3.
ADD 98 // Legg tall 2 til tall 3.
ADD 99 // Legg tall 1 til delsummen av tall 2 og 3.
OUT // Skriv ut svaret.
HLT // Avslutt.
```

Oppgave 2

Tallkode representasjon:

901
399
199
902
000

Assemblerkode representasjon:

```
INP // Les et tall.
STO 99 // Legg det i celle 99.
ADD 99 // Legg de to (like) tallene sammen.
```

OUT // Skriv ut summen.
HLT // Avslutt.

Oppgave 3

Tallkode representasjon:

901
805
392
292
292
902
000

Assemblerkode representasjon:

INP // Les inn et tall.
BRP *posi* // Om tallet er positiv, hopp til *posi.* Om den er negativ, gjør ingenting.
STO 92 // Lagre tallet i minnecelle 92.
SUB 92 // Subtraher det tallet fra seg selv, resulterer i 0.
SUB 92 // Subtraher igjen til å få det tilsvarende positive tallet.
posi OUT // print(a)
HLT //

Oppgave 4

Variant 1 (fra fasit, men litt omgjort)

Pseudo-kode:

1. Skrive inn et tall, addere den med det som står i en gitt minnecelle, og sette summen inni den samme minnecellen.
2. Gjør så mange ganger du vil, og avslutt med 0, sånn at programmet skjønner at vi er ferdig med input.
3. Hente den eventuelle summen fra minnecellen.
4. Gi summen som output.

Tallkode representasjon:

901
705

199
399
600
599
902
000

Assemblerkode representasjon:

```
        // Gå i løkke:
INP // Les et tall.
BRZ skriv // Avslutt løkken om input er 0.
ADD 99 // Adder tallet til det som er på plass 99.
STO 99 // Lag den nye nåværende summen der.
BRA 0 // Gå tilbake til starten av løkke.
skriv LDA 99 // Hent summen fra plass 99.
OUT // Skriv ut summen.
HLT //
```

Variant 2 (det som vi prøvde å lage i gruppetimen)

Denne løsningen er utrolig knotete og mye mindre effektiv enn den som er presentert i fasit (fordi det er svært mange steg vi må ta og fordi det er kun så mange tall vi kan lagre pga minnebegrensninger), så vi bare dropper det.

Oppgave 5

Samme som i løsningsforslaget, nesten ingenting å kode her.

Oppgave 6

Samme som i fasit.