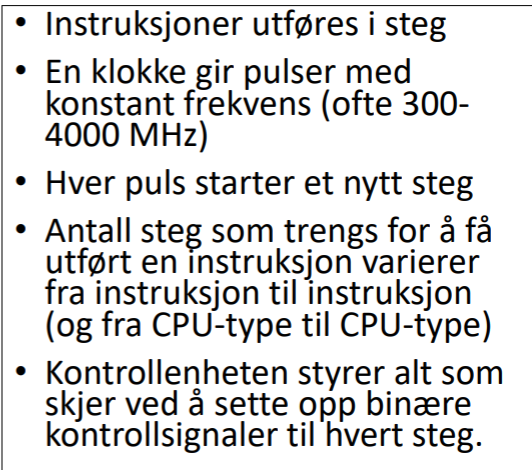
# CPU-ens komponenter



**Fire hovedkomponenter:**

1. Kontrollenhet
2. Registerblokk
3. Klokke
4. ALU (Arithmetic Logic Unit)

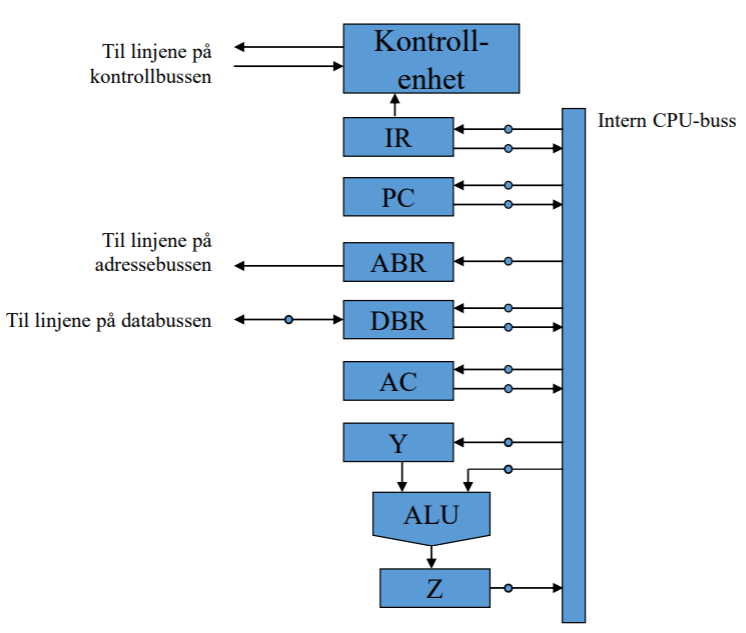
**Kontrollenheten** bestemmer hva som skal skje mellom hver klokkepuls. I den ene klokkepulsen kan en instruksjon kjøres, og i en annen puls en annen instruksjon. Dette bestemmer kontrollenheten.

**Klokken** er tidsenheten i prosessoren.

**ALU** står for selve regningen og komputasjonen av mange av instruksjonene.

**Registerblokken**

* **Synlige registre** (Viktigst for utviklere)
  + Registrene som er synlige for oss (programmerere).
  + Generelle registre, Adresseregistre (stakkpeker, segmentregistre, indeksregistre).
  + Flaggregister, sier noe om resultatet etter en instruksjon.
* **Kontroll- og statusregistre**
  + Programtellere, instruksjonsregister, registre som brukes av OS.
* **Interne registre**
  + Registre som brukes til bufring under utførelse av instruksjoner.

****

Bryter



**Busstopologi inni CPU**

**ABR:** Adresse Buffer Register er tilknyttet adressebussen slik at en adresse kan overføres til minne

**DBR:** Data Buffer Register er tilknyttet data-bussen slik at data kan flyttes mellom CPU og minne.

*ABR og DBR er koblet til «omverdenen». Hver bit er koblet til disse bufferene i CPU.*

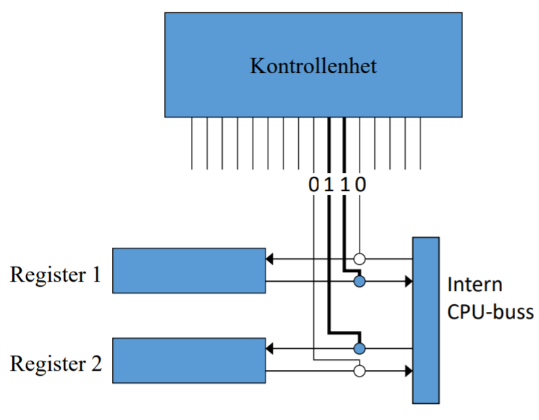
**Y og Z:** Dette er registre som trengs fordi ALU må ha registre som innganger og utgang samtidig.

**Mikrooperasjoner**

Instruksjoner som kan utføres i en klokkesyklus. En instruksjon er som regel satt sammen av flere ulike mikrooperasjoner. Hvilke mikrooperasjoner som skal utføres bestemmes av kontrollenheten.

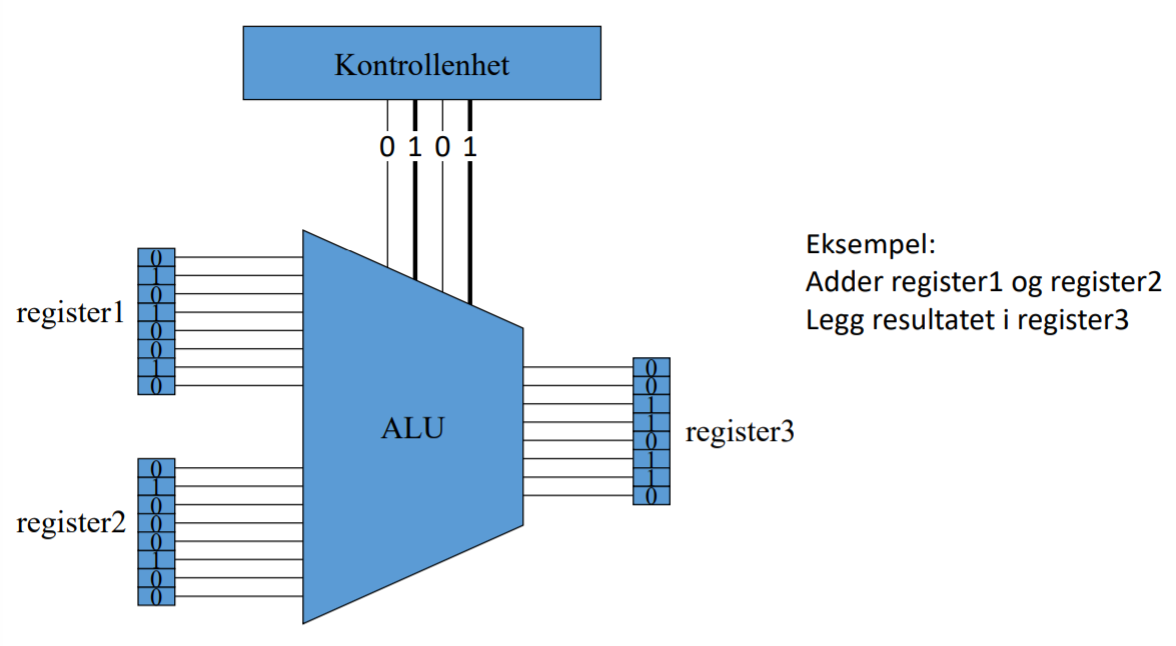
Det finnes tre ulike slike mikrooperasjoner:

1. **Flytte data mellom to registre (dataflyt)**

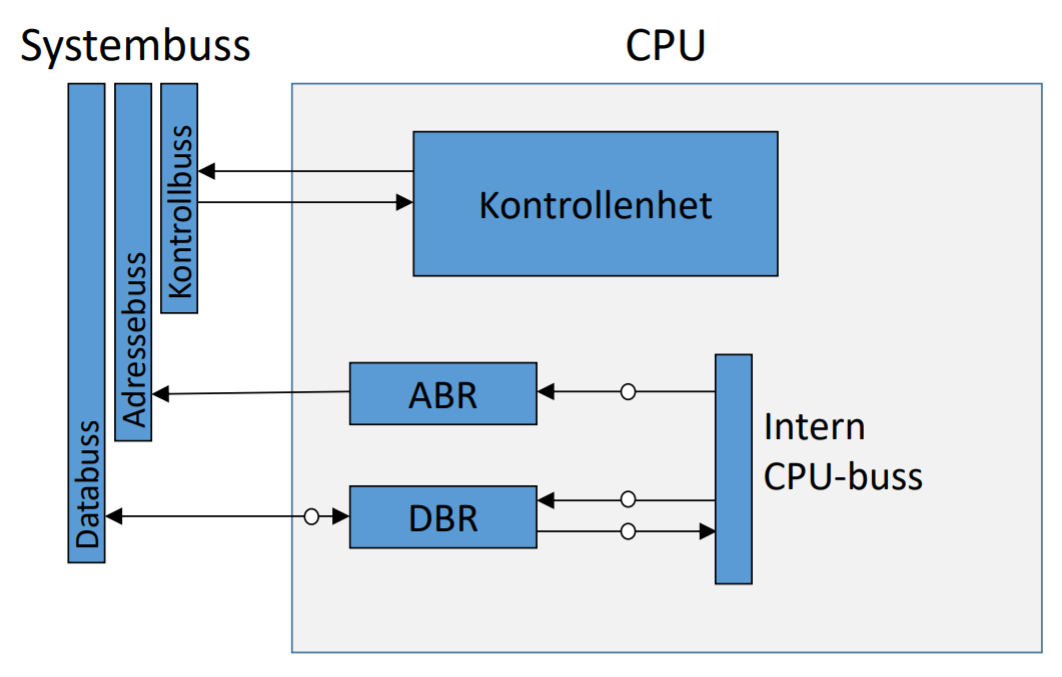


Kontrollenheten må kontrollere dataflyten ved å åpne og lukke de rette bryterne mellom registre.

1. Utføre aritmetiske operasjoner eller logiske operasjoner hvor input data ligger i registre og resultat legges i et register.



1. Flytte data mellom CPU og eksternt utstyr (via data-bussen)



**Kontrollenheten har to oppgaver:**

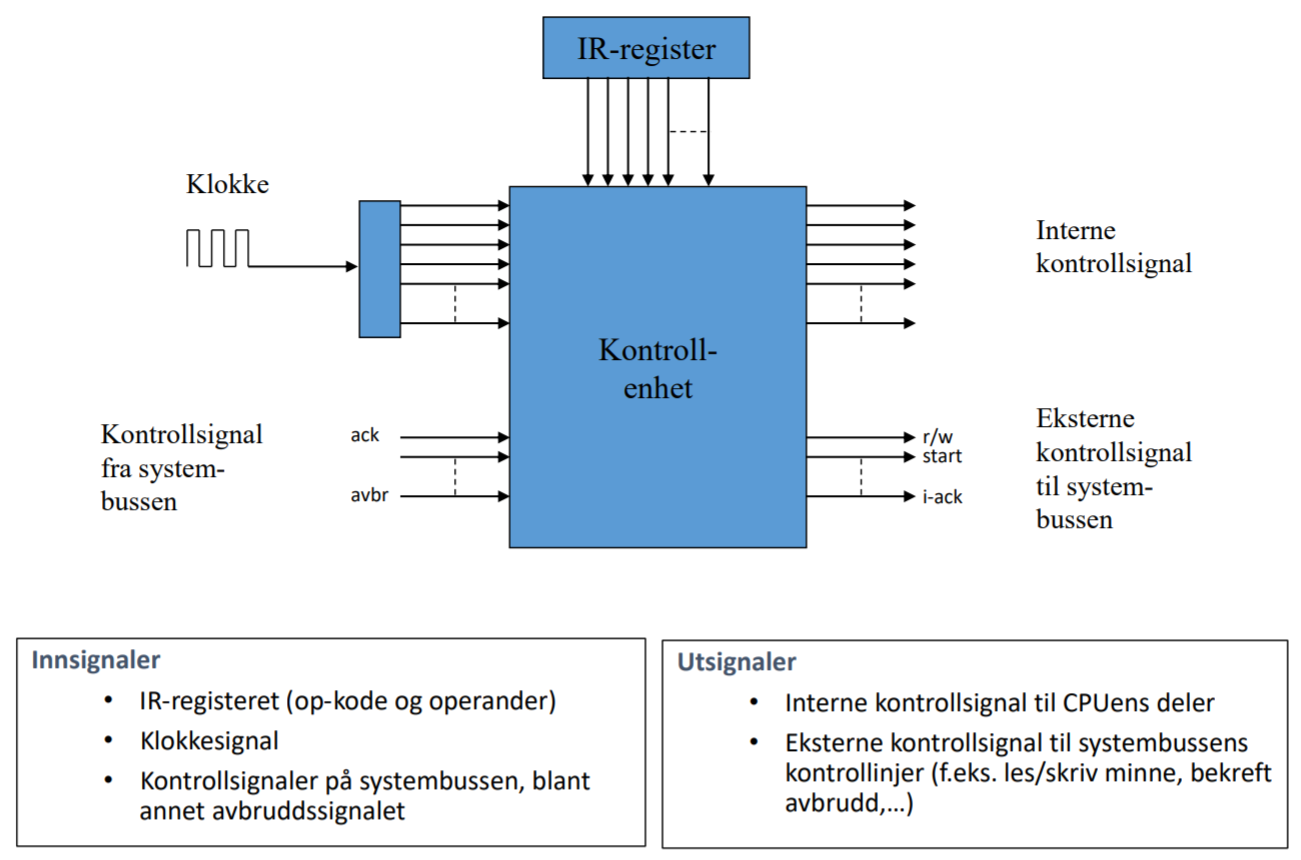
1. Sekvensering

* Sørge for å finne ut hvilke mikrooperasjoner som skal utføres

2. Eksekvering

* Vent til neste klokkepuls kommer, og sette de riktige kontrollsignalene for neste operasjon.

En mikrooperasjon utføres etter hver klokkepuls. En mikrooperasjon tar kun en klokkepuls. Kontrollenheten skal tolke instruksjonene den får tilsendt og deretter sette de riktige utgangene for at instruksjonen skal utføres. En instruksjon består av flere mikrooperasjoner.



*Kontrollenheten er en elektronisk enhet, denne er meget komplisert.*

**Oppbygging av en kontrollenhet**

1. Hardlogikk

* Kontrollenheten bygges opp av logiske porter, kalles da en avansert kombinatorisk krets.

2. Mikroprogrammert kontrollenhet

* Bygges opp av en programmbar enhet som kjører lavnivå program.

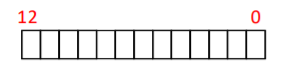
**Hva er best av de to byggemetodene?**

Hardlogikk er hurtigere, men vanskeligere å bygge. Programmert er tregere, men mer fleksibel.

* Etter 80-tallet: alle nyere design baserer seg på hardlogikk.

**Mikroprogrammert kontrollenhet**

1. Vi nummerer kontroll-signalene
2. Vi laget et bitmønster med en bit for hvert kontroll-signal.



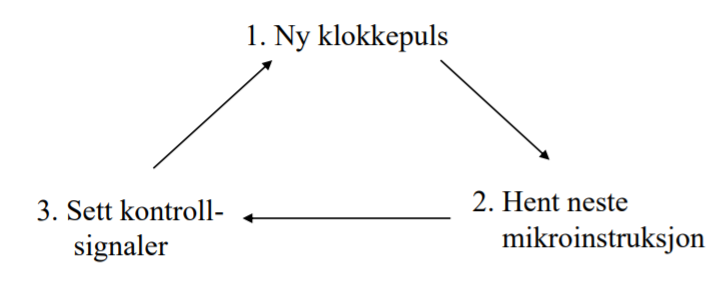
3. Vi skriver 1 for kontroll-signalene som skal settes, og 0 for de som skal nullstilles.

Dette kalles **mikrooperasjoner**! Altså en mikrooperasjon er et bitmønster som forteller om hvilke kontrollsignaler som skal brukes. Disse mikrooperasjonene ligger i et internt minne i kontrollenheten. Hver instruksjon (merk forskjellen mellom instruksjon og mikrooperasjon) lages av en sekvens av mikrooperasjoner som da lagres i det interne minnet.

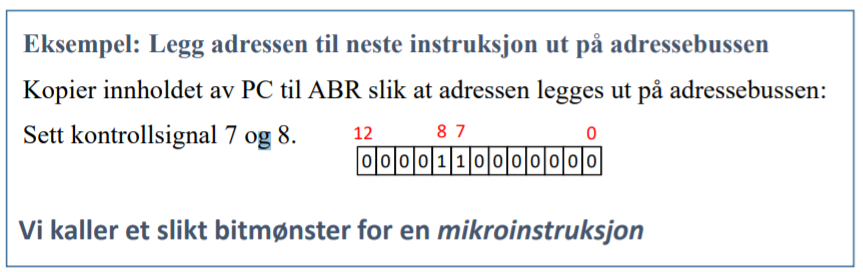
Man kan si at man har laget en «mikro-datamaskin» internt inni kontrollenheten. Vi har et *kontroll*- eller *mikrominne*, mikro-programteller (hvor neste mikroinstruksjon ligger), *mikro-instruksjons-register.* Denne mikro-maskinen har en instruksjonssyklus på lik måte som vanlige maskiner.

Klokkefrekvensen til prosessoren (CPU) blir mindre med mikroprogrammert kontrollenhet. Som er grunnen til at hardlogikk er foretrukket.

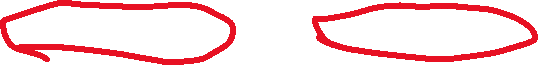
*Hvordan kontrollenheten opererer.*

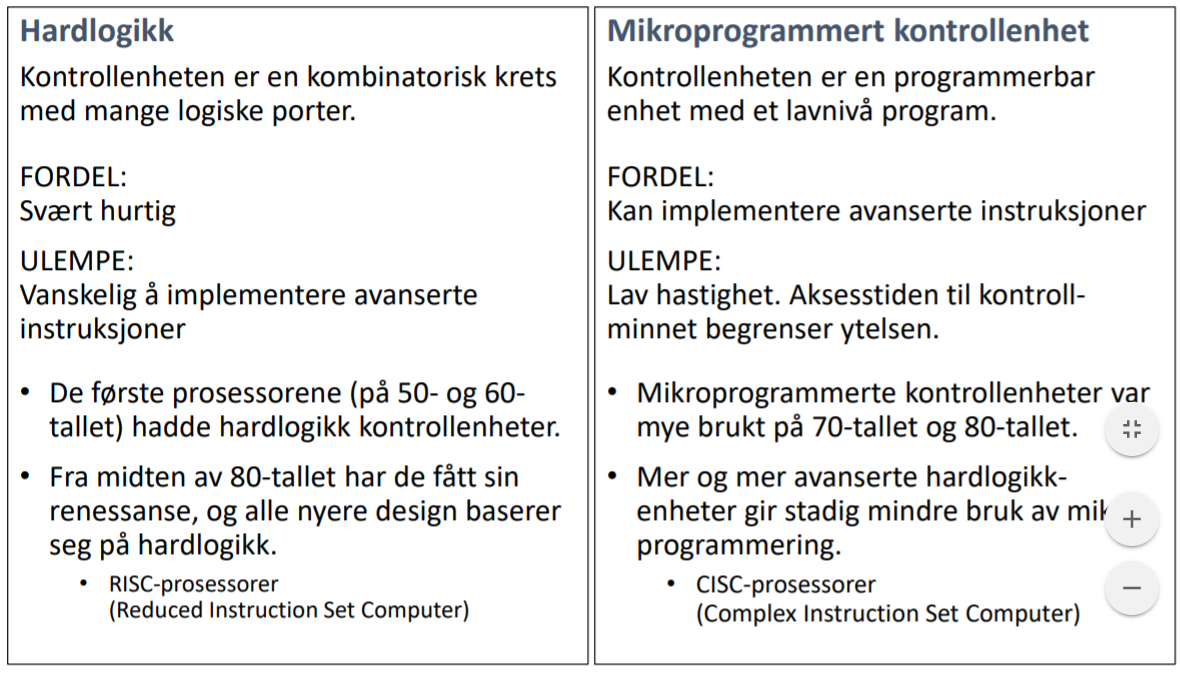


*Eksempel*:



**Kontrollenheten metoder oppsummert**





Pentium er et eksempel på en CISC-prosessor.