Computer Arkitektur og Operativ Systemer

Denne forelæsning optages og gøres efterfølgende tilgængelig på Moodle
MEDDEL VENLIGST UNDERVISEREN, HVIS DU IKKE ØNSKER, AT OPTAGELSE FINDER STED

This lecture will be recorded and afterwards be made available on Moodle
PLEASE INFORM THE LECTURER IF YOU DO NOT WANT RECORDING TO TAKE PLACE

Computer Arkitektur og Operativ Systemer

Processor Arkitektur: Den Sekventielle Y86-64

Lektion 6

Brian Nielsen

Credits to

Randy Bryant & Dave O'Hallaron (CMU)

Lektionsplan

1. Intro, bits og bytes
2. Heltals-repræsentation, lidt floats
3. Assembler 1: registre, dataflytning
4. Assembler 2: kontrol-strukturer & arrays
5. Assembler 3: procedurekald og buffer-overflow
6. Mikro-arkitektur: en sekventiel processor
7. Instruktions-niveau parallelitet og compiler-optimering
8. Hukommelses-hierarkiet, caching
9. Exceptions og processer
10. Virtuel hukommelse
11. Concurrency & multithreading 1
12. Concurrency & multithreading 2

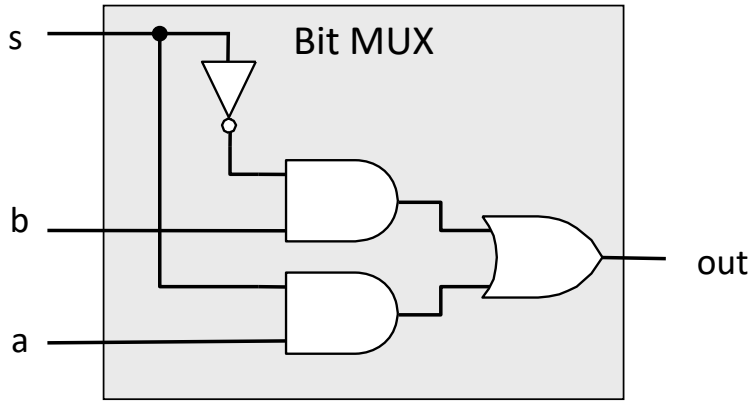
Lilla: mest comp. ark.
Grøn: mest OS

I dag: opbygning af SEQ processor fra bunden!

- Hvordan opbygges digitale kredsløb?
  - Kombinatoriske, opbygning med logiske ”gates”
  - Sekventielle (med register hukommelse)
- Y86-64 SEQ: forenklet instruktionssæt
- Hvordan repræsenteres instruktioner som bytes?
- Hvordan fortolker HW en instruktion?
  - 6 trins-behandlingsmodel
  - Timing
- Mindre vigtigt:
  - Transistorer
  - Bistabile elementer
  - Design af Y-86 kontrol logic (Sekventielt)

Digitale logiske kredsløb

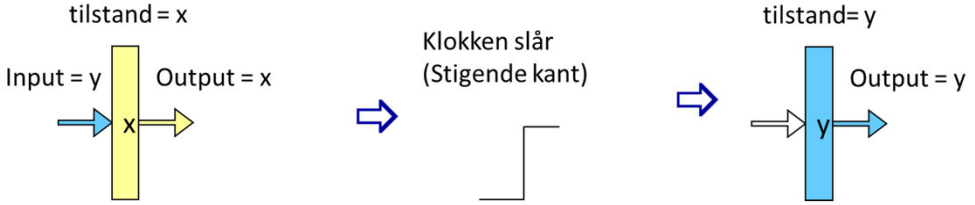
- Gates
  - AND, OR, NOT, NAND, NOR
- Boolsk Algebra
  - Beskrivelse af en ønsket beregning som formel
  - Reduktion til få gates,
  - primært bruge en bestemt gate-type, fx NAND
- Kombinatoriske: netværk uden hukommelses elementer
- Sekventielle: med hukommelses-elementer
  - kræver timing!



A	B	S	A MUX B
0	0	0	0
0	1	0	1
1	0	0	0
1	1	0	1
0	0	1	0
0	1	1	0
1	0	1	1
1	1	1	1

HCL Udtryk

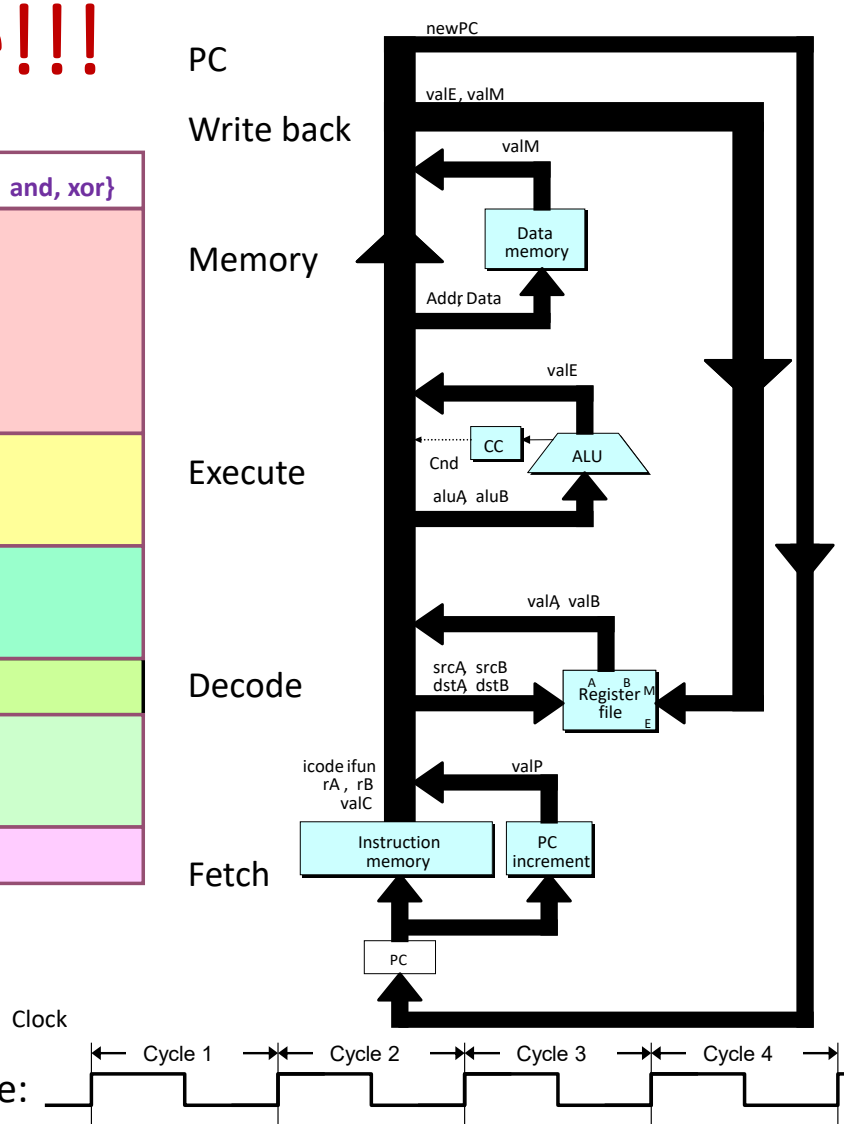
```
bool out = (s&&a)||(!s&&b)
```



Instruktionsbehandlings-løkke!!!

1. Fetch
   - Læs instruktion fra instruktionshukommelse
   - Beregn værdi for næste PC
2. Decode
   - Udlæs Operander fra program registre
3. Execute
   - Beregn værdi eller adresse
4. Memory
   - Læs eller skriv data i hukommelsen
5. Write Back
   - Skriv resultat til program registre
6. PC
   - Opdatér program tæller
7. Gentag fra 1.

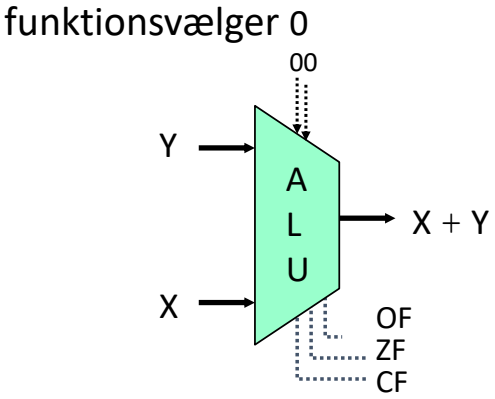
	OPq rA, rB //OP∈{add, sub, and, xor}
Fetch	icode:ifun ← M_1[PC] rA:rB ← M_1[PC+1] valP ← PC+2
Decode	valA ← R[rA] valB ← R[rB]
Execute	valE ← valB OP valA Set CC
Memory	
Write back	R[rB] ← valE
PC update	PC ← valP



I en SEQ processor: 1 instruktion pr. cyclus, afsluttes med skrivning af registre:

Øvelserne

- Y86-64 instruktionssættet og instruktionssæt indkodning
- Hvorfor det er vigtigt præcist at specificere hvad instruktionerne gør!
  -Semantik er også relevant på assembler niveau ;-)
- Beskrivelse og håndtracing af en instruktionen
- Challenge 6: lav et kredsløb til addition



Brug Hjælpelærer assistancen!