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
Exceptions og Processer

Forelæsning 9 Intro
Brian Nielsen

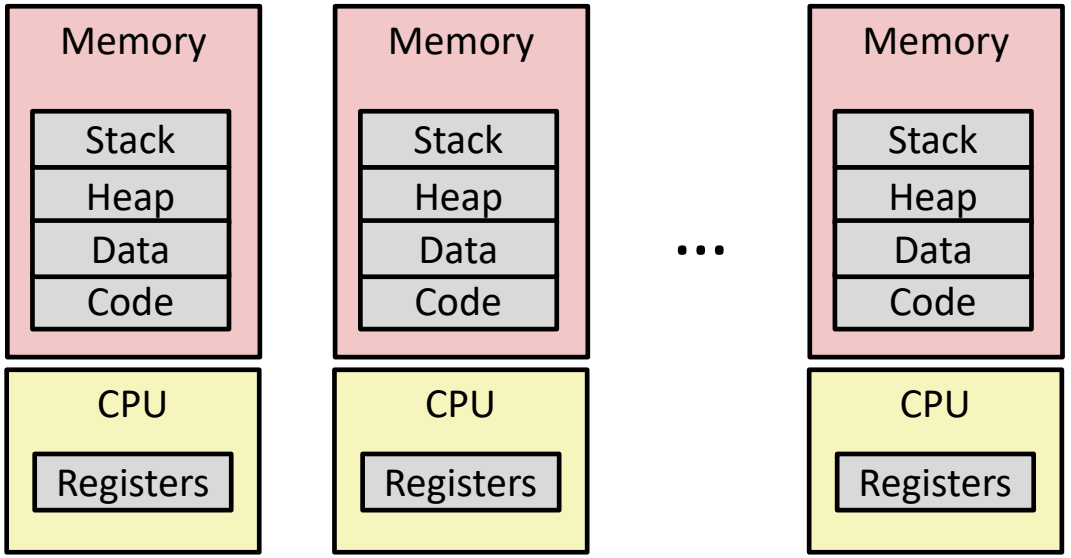
Credits to
Randy Bryant & Dave O'Hallaron (CMU)

Idag

- Hvad er formålene med et Operativ System?
- Hvordan bevarer det kontrol over hardware ressourcer?
- Hvad er en kerne?
- Hvad er en proces?
- Hvordan programmerer vi med processer?
- Hvad er en proces kontrol blok?
- Hvad er et kontekst-skift?
- Hvordan afvikles og schedulers processer?

Multi-programmering: En Illusion

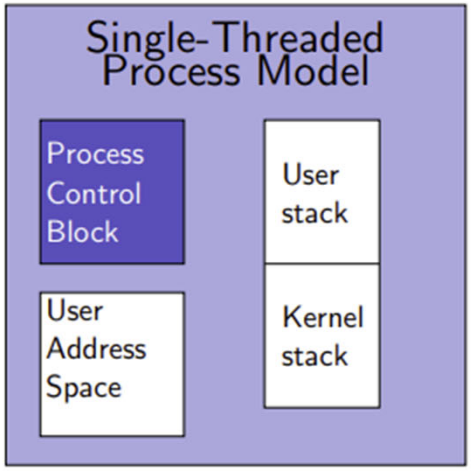
- CPU-virtualisering



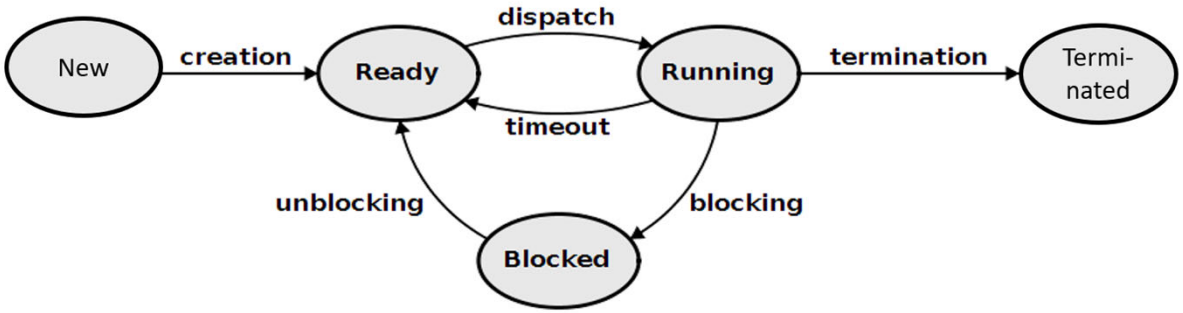
- I virkeligheden har vi kun én CPU men rigtigt mange processer!

Process Realisering

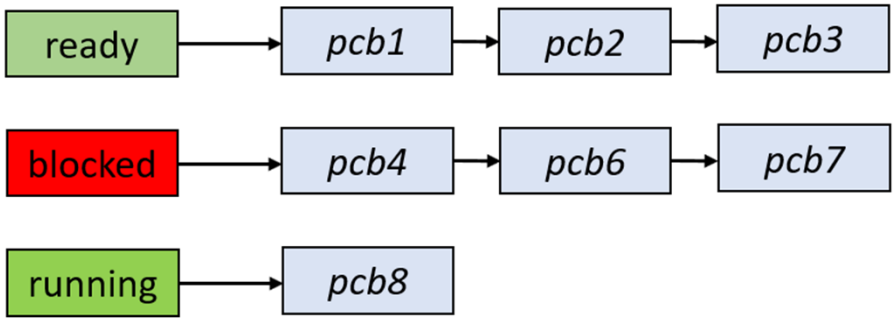
- Bestand-dele



- Fx 5 tilstandsmodel

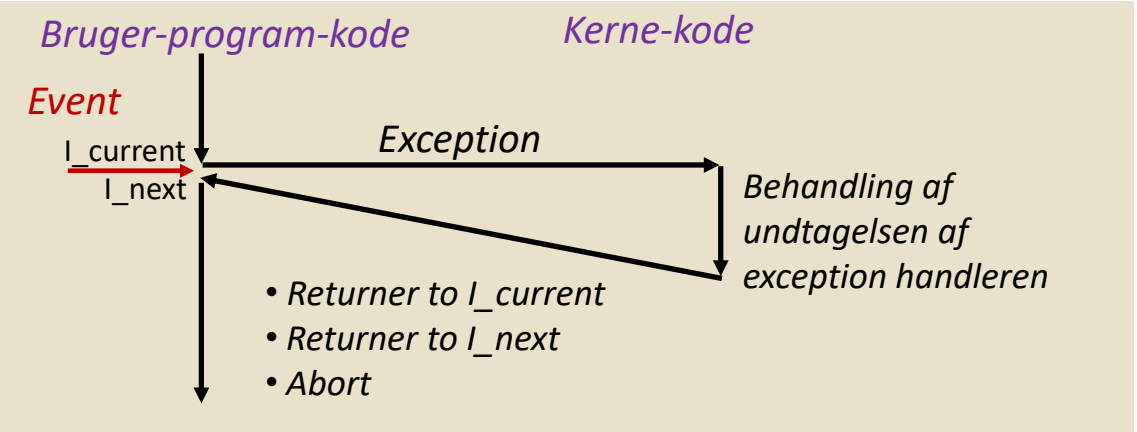


- Mulig kø-strukturer i OS kerne



Exception mekanismen er central for realisering af OS og processer

- Hardware undtagelser: forgrening forårsaget af system-tilstand
  - System-kald
  - Fejl-situationer
  - Interrupts



Om XV6

- I OSTEP henvises ofte til XV6 studie-OS!
  - XV6, et simpelt Unix-lign. undervisnings-OS
  - Gen-implementation af den oprindelige Unix-V6!
  - Kilde-tekst tilgængeligt, lærerrigt at grave i
- I CAOS kun illustrativt
  - Vedligeholdes ikke længere for X86 (findes kun som X86-32)
  - RISC-V ISA lag (et akademisk open-source ISA)

```
1  # t0 = 0
2  li      t0, 0
3  li      t2, 10
4  loop_head:
5  bge     t0, t2, loop_end
6  # Repeated code goes here
7  addi    t0, t0, 1
8  j       loop_head
9  loop_end:
```

Øvelserne

- Review af vigtige begreber.
- Side-ordnet afvikling (Concurrency)
  - Multi-programmering
  - Proces-graf: fork
  - ***Uforudsigelig indbyrdes afviklingsrækkefølge af processerne!***
    - ***Kun bestemte synkroniseringshændelser skaber en ordning.***
- Implementer Fibonacci(***n***) vha `fork()`, `wait()`, `exit()` for SMÅ ***n*** ;-)
- Challenge 11: Lav din egen shell / CLI
- Kan laves i enhver unix/linux, incl. CAOS-VM, Linux, WSL, WSL

Brug Hjælpelærer assistancen!