branch prediction
Meltdown
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)

branch prediction

- Ved en branch i programmet (if-test), vet man ikke hva neste instruksjon er
- Stort problem for pipelining, må vente på resultatet fra forrige instruksjon
- Gjetter, basert på erfaring, hvilken branch (gren) som følges i programmet og utfører den
- Speculative execution, må gjøres om hvis feil
- I et superskalær arkitektur kan begge grener delvis utføres på forhånd

branch prediction
Meltdown
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)

Meltdown

- Et hardware-sikkerhetshull funnet i 2018
- Rammet Intel, ARM og IBM-prosessorer
- Meltdown utnytter at både koden som sjekker om prosessen kan lese fra RAM og lesingen fra RAM delvis utføres
- Meltdown kan dermed lese data fra andre prosesser som er cache't men ennå ikke fjernet pga feil branch
- Spectre brukte lignende metoder til å lese passord og sensitive data
- Betegnet som sikkerhets-katastrofe
- Både CPU design og operativsystemer ble endret for å hindre Meltdown og Spectre i å virke

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)

Viktig å huske fra datamaskinarkitektur

- Alt CPU'en gjør er å slavisk utføre maskininstruksjoner en for en i en evigvarende løkke
- Ingen en til en forbindelse mellom instruksjoner i høynivkode og maskinkode
- En linje kode i høynivåspråk fører ofte til mange maskininstruksjoner i det kompilerte programmet

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)

CPU-løkke (hardware-nivå)

Pseudo-kode for den evige hardware-løkken som CPU'en kjører:

```
while(not HALT)
{
   IR = mem[PC];    # IR = Instruction Register
   PC++;            # PC = Program counter
   execute(IR);
   if(InterruptRequest)
   {
      savePC();
      loadPC(IRQ); # IRQ = Interrupt Request
                   # Hopper til Interrupt-rutine
   }
}
```

branch prediction

Viktig å huske fra datamaski-narkitektur

CPU-løkke (hardware-nivå)

Microsoft og Unix OS

Microsoft Desktop-OS

Microsoft Server-OS

Unix operativsystemer

Multitasking

CPU-løkke (hardware-nivå)

Microsoft og Unix OS

- Microsoft Desktop-OS
- Microsoft Server-OS
- Kommersiell Unix
- Linux og frie Unix kloner

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Desktop-OS
Windows 10 versions
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)

Microsoft Desktop-OS

MS-DOS	1981, 16-bit
Windows	1.0 i 1985, 3.0 i 1990, GUI på toppen av DOS
Windows 95	Noe 32-bit kode, mye 16-bit intel assembler,DOS-filsystem, bruker DOS til å boote
Windows 98	essensielt som 95, desktop/internet integrert
Windows Me	essensielt som 98, mer multimedia og nettverk support
Windows 2000	Første (ikke så vellykkede) forsøk med Desktop OS basert på NT (5.0).
Windows XP	Oktober 2001, Desktop-OS som kombinerer NT 5.1 kode med Win 9x. Home edition: 1 CPU, XP Professional: 2CPU'er, logge inn utenfra, flere muligheter.32 og 64 bit
Windows Vista	januar 2007. Kernel NT 6.0, Booter raskere, bedre filsøk, User Account Control (UAC). Ingen suksess.

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Desktop-OS
Windows 10 versions
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)

Microsoft Desktop-OS

Windows 7	oktober 2009, kjernen er Windows NT 6.1, PowerShell 2.0 default, 7 editions, Service Pack 1
Windows 8	oktober 2012, NT 6.2, Start Screen, touch screen, USB 3.0, Windows Store, Windows RT for ARM
Windows 8.1	oktober 2013, NT 6.3
Windows 10	July 2015, NT 10.0(!), Microsoft Edge, virtual desktops, native Ubuntu bash shell(samarbeid med Canonical) gjennom Windows Subsystem for Linux

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Desktop-OS
Windows 10 versions
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)

Windows 10 versions

Versions: Home,Pro ,Pro Education ,Education ,Pro for Workstations, Enterprise

Versjon	Release dato	Support til	Support LTSC
1507	July 29, 2015	May 9, 2017	Oct 14, 2025
1511	Nov 10, 2015	Oct 10, 2017	-
1607	August 2, 2016	April 10, 2018	Oct 13, 2026
1703	April 5, 2017	Oct 9, 2018	-
1709	Oct 17, 2017	April 9, 2019	-
1803	April 30, 2018	Nov 12, 2019	-
1809	Nov 13, 2018	May 12, 2020	Jan 9, 2029
1903	May 21, 2019	Dec 8, 2020	-
1909	Nov 12, 2019	May 11, 2021	-
2004	First half of 2020	Second half of 2021	-

Ett år lenger support for Education og Enterprise

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)

Microsoft Server-OS

NT 3.1 1993 32-bit, skrevet fra scratch i C (lite assembler), David Cutler (VAX-VMS designer), mye bedre sikkerhet og stabilitet enn Windows. 3.1 millioner linjer kode.

NT 4.0 1996, Samme GUI som Win-95, 16 millioner linjer, portabelt -> alpha, PowerPC

Windows 2000 NT 5.0, opp til 32 CPU'er, features fra Win-98, Plug and Play, `\winnt\system32\ntoskrnl.exe`. 29 millioner linjer. *MS-DOS borte, men 32-bits kommando-interface med samme funksjonalitet.*

Windows Server 2003 bygger på 2000 server, NT 5.2, design med tanke på .NET: web, XML, C#, 32 og 64 bit, SP1, SP2, R2 i 2006

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)

Microsoft Server-OS

Windows Server 2008 februar 2008, felles basis med Vista, første OS med PowerShell, Kan installeres som **Server core** og styres fra CLI (Command Line Interface), Hyper-V virtualisering

Windows Server 2008 R2 oktober 2009, NT 6.1 (som Win 7), PowerShell 2.0 default, kun 64 bit

Windows Server 2012 september 2012, NT 6.2 (som Win 8), cloud computing, oppdatert Hyper-V, nytt filsystem: ReFS

Windows Server 2012 R2 oktober 2013, NT 6.3 (som Win 8.1)

Windows Server 2016 september 2016, NT 10.0, Windows Defender, nano server: uten gui, fjernstyres med PowerShell

Windows Server 2019 oktober 2018, NT 10.0, Windows Admin Center

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
64 bit Unix-OS
Frie Unix-kloner
Multitasking
CPU-løkke (hardware-nivå)

Unix operativsystemer

Dagens Unix-versjoner har utviklet seg fra to som dominerte rundt 1980:

- system V (AT&T Bell Labs)
- BSD (University of California at Berkely)

De fleste av dagens varianter er bygd på SVR4 som er en blanding.

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
64 bit Unix-OS
Frie Unix-kloner
Multitasking
CPU-løkke (hardware-nivå)

Kommersielle 64 bit Unix-OS for RISC-prosessorer

OS	Eier	hardware
AIX	IBM	RS6000, Power
Solaris	Sun	Sparc, intel-x86
HP-UX	Hewlett-Packard	PA-RISC, Itanium(IA-64)
Tru64 UNIX	HP(Compaq(DEC))	Alpha
IRIX	Silicon Graphics	SGI

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
64 bit Unix-OS
Frie Unix-kloner
Multitasking
CPU-løkke (hardware-nivå)

Frie Unix-kloner

OS	hardware
FreeBSD	x86, Alpha, Sparc
OpenBSD	(sikkerhet) x86, Alpha, Sparc, HP, PowerPC, mm
NetBSD	x86, Alpha, Sparc, HP, PowerPC (Mac), PlayStation, mm
Darwin	intel x86, ARM, PowerPC
Linux	x86, Alpha, Sparc, HP, PowerPC, PlayStation 3, Xbox, stormaskin

Darwin er basis for Mac OS X og iOS, kjernen til Darwin er XNU og bygger på FreeBSD og Mach 3 microkernel

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

CPU-løkke (hardware-nivå)

Pseudo-kode for den evige hardware-løkken som CPU'en kjører:

```
while(not HALT)
{
   IR = mem[PC];    # IR = Instruction Register
   PC++;            # PC = Program counter
   execute(IR);
   if(InterruptRequest)
   {
      savePC();
      loadPC(IRQ); # IRQ = Interrupt Request
                   # Hopper til Interrupt-rutine
   }
}
```

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

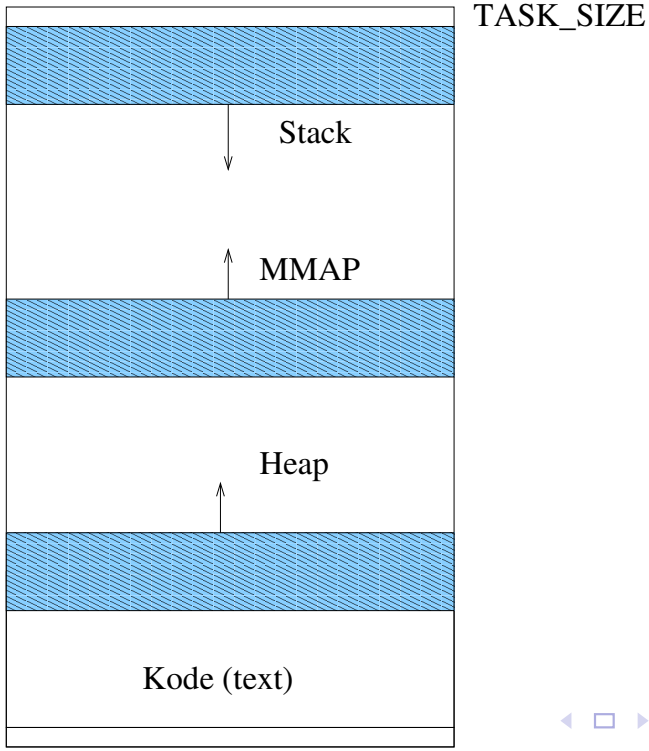
Interrupts (avbrytelser)

- Signal fra hardware
- CPU'en avbrytes for å håndtere signalet
- Lagrer adressen til neste instruksjon på stack og hopper til interrupt-rutinen
- Hvert interrupt-nr (IRQ) har sin rutine

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

En Linux-prosess sitt internminne

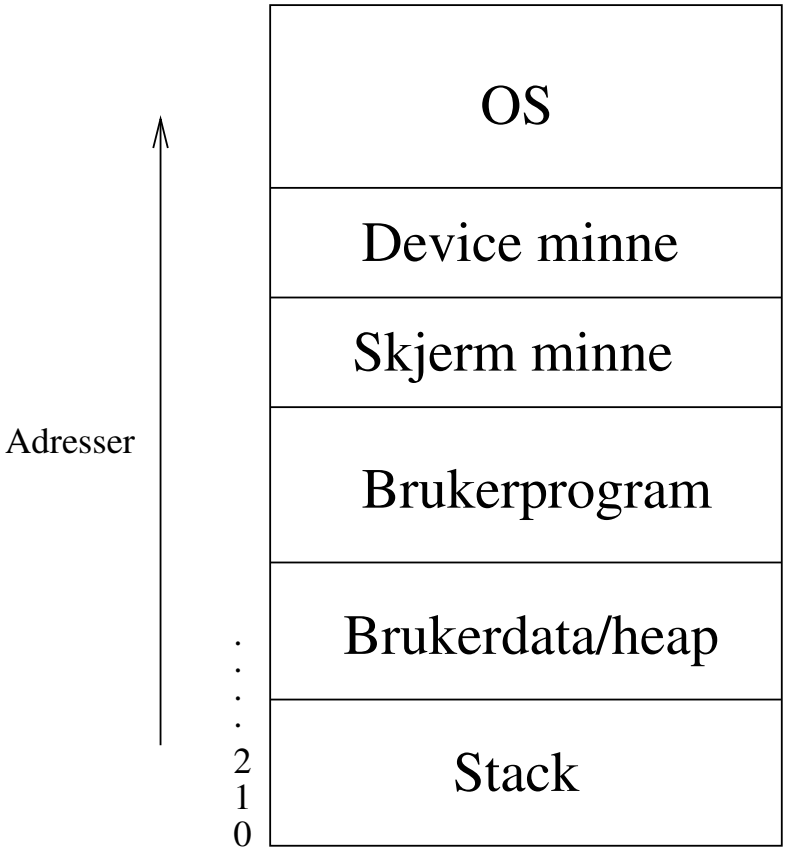
- Den statiske binære koden som prosessen kjører, text segmentet, ofte bare kalt text
- Heap'en hvor globale variabler og data som dynamisk generers lagres
- Stack'en hvor de lokale variablene lagres, brukes også til funksjonskall
- MMAP, minneavbildninger av filer(og devicer) på disk direkte i det virtuelle minnet



branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

Singletasking OS

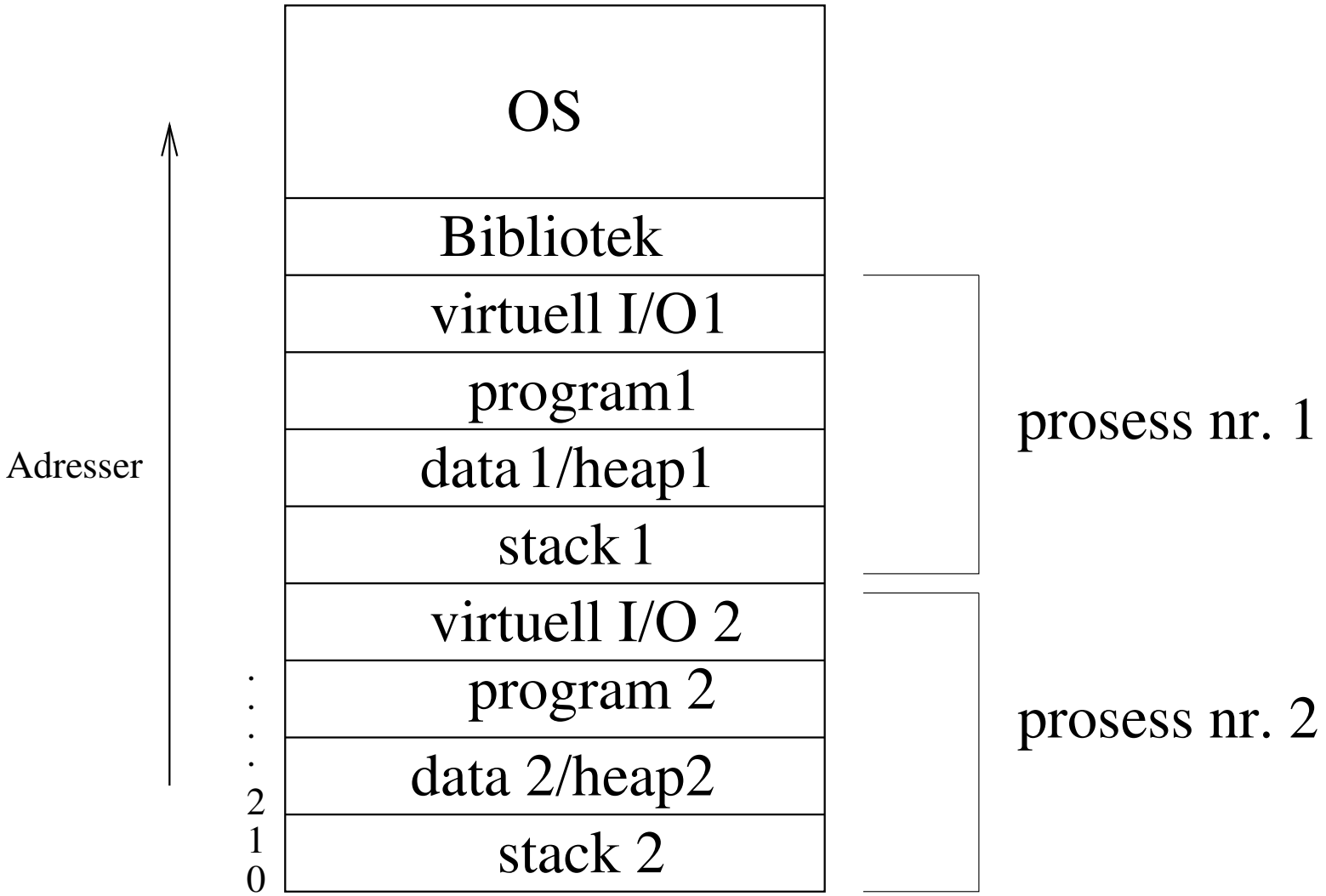
Basis for flerprosess-systemer.



branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

Multitasking-OS

For å lage et system som kan kjøre n programmer samtidig, må vi få en enprosess maskin til å se ut som n maskiner.



branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

Multitasking

- CPU gjør en og en maskinistruksjon av gangen
- Et multitasking operativsystem får det til å se ut som om mange programmer kan kjøre samtidig ved å dele opp tiden i små biter (timeslices)
- Hver prosess som kjører få en bit CPU-tid, 1/100 sek av gangen, i et køsystem (**Round Robin** kø)
- Preemptive multitasking: En hardware timer (klokke) sender hvert hundredels sekund et interrupt-signal til CPU
- Første OS-instruksjon legges inn i instruksjonsregisteret i CPU'en
- OS lar hver prosess etter tur bruke CPU'en i et 1/100 sekund
- Unngår at en brukerprosess henger eller tar over kontrollen
- Alle prosesser ser da ut til å kjøre samtidig

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

Context Switch

Når OS switcher fra prosess P1 til prosess P2 utføres en såkalt Contex Switch (kontekst svitsj).

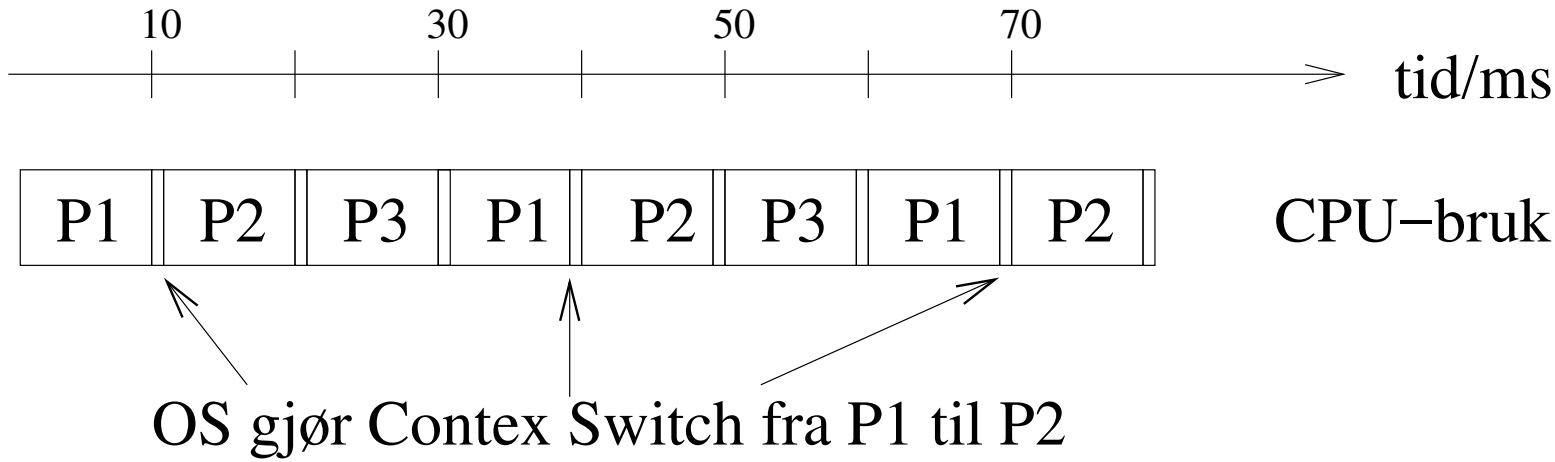


Figure: Prosessene P1, P2 og P3 kjører samtidig under et multitasking OS. En Context Switch utføres hver gang en prosess gis CPU-tid.

Typisk tid som brukes på en context-switch: 0.001 ms (ms = millisekund = tusendels sekund). Timeslice = 10 ms for Linux på Intel.

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

PCB -Process Control Block

Process Control Block (PCB) er Prosessens tilstandsbeskrivelse: prioritet, prosessormodus, minne, stack, åpne filer, I/O, etc. PCB inneholder bl. a. følgende:

- CPU registere
- pekere til stack
- prosesstilstand (sleep, run, ready, wait, new, stopped)
- navn (PID)
- eier (bruker)
- prioritet (styrer hvor mye CPU-tid den får)
- parent prosess
- ressurser (åpne filer, etc.)

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

Scheduling

- CPU-scheduling = å fordele CPU-tid mellom prosessene = Time Sharing
- Ved hvert timer-interrupt til CPU, vurderer OS om scheduler-rutinen skal kalles
- Scheduleren avgjør hvilken prosess som skal velges til å bruke CPU'en
- Når OS switcher fra prosess P1 til prosess P2 utføres en Contex Switch
- All PCB-info må lagres i en Context Switch $\longrightarrow$ tar tid $\longrightarrow$ systemoverhead

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

PCB og Context Switch

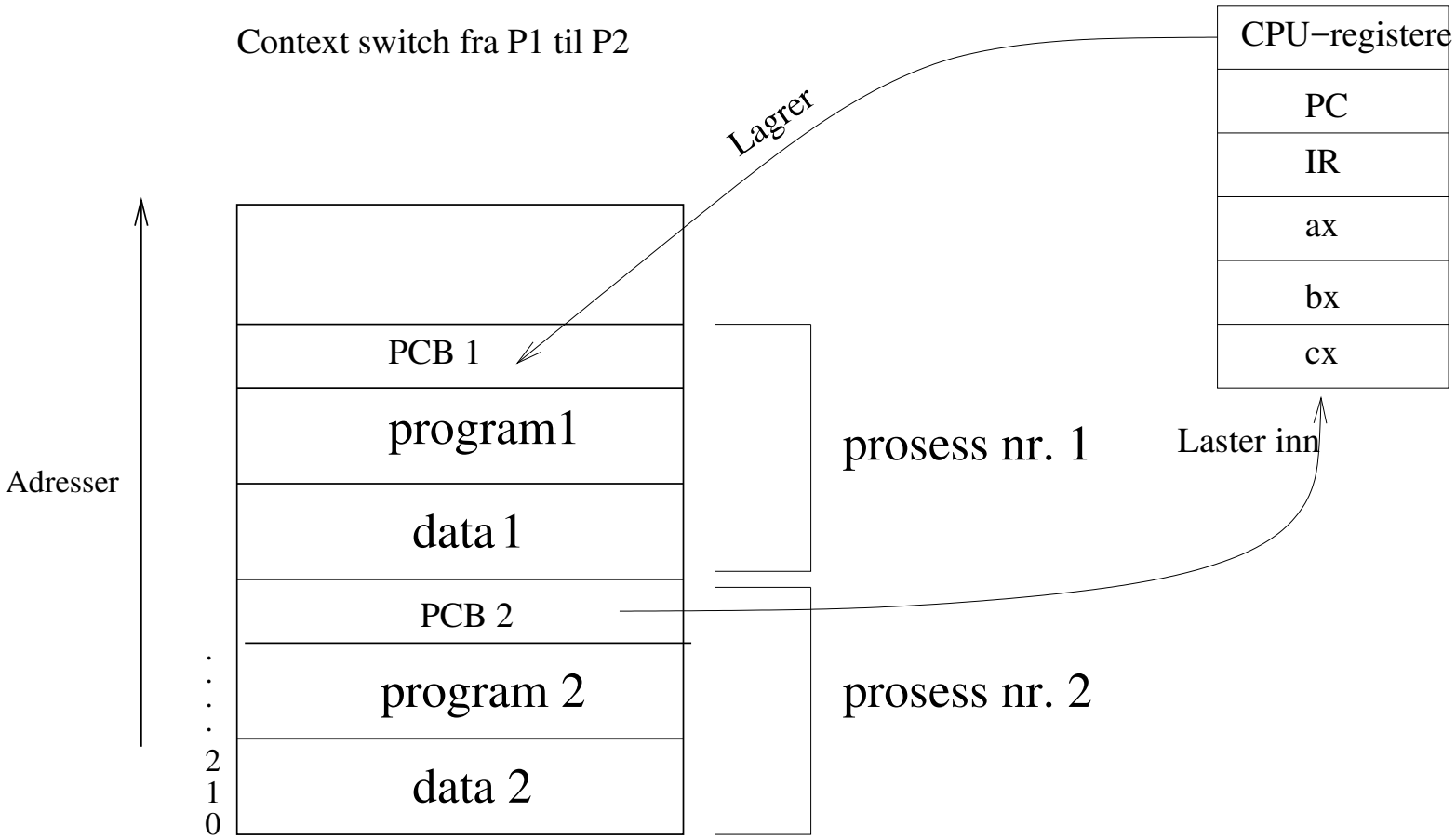


Figure: CPU info lagres i PCB ved en Context Switch

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

Multitasking i praksis, CPU-intensive programmer

- Et program som oversetter kildekode til maskinkode (kompilator) eller et program som hele tiden regner med tall, vil bruk så mye CPU-tid som det klarer å få tak i. Kalles CPU-intensive.
- De fleste vanlige programmer som browsere, tekstbehandlingsprogrammer, tengeprogram etc. bruker lite CPU
- Dette gjør at multitasking av hundretalls samtidige prosesser går helt greit uten at brukeren oppfatter datamaskinen som treg

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

Multitasking i praksis

- Kjører et lite shell-script som står i en løkke og regner og regner
- Vil hele tiden bruke så mye CPU det kan få
- Har aldri behov for å vente på data fra disk, tastatur eller andre prosesser, kan regne uten stans

```
#! /bin/bash
# regn (bruker CPU hele tiden)
(( max = 100000 ))
(( i = 0  ))
(( sum = 0  ))

echo $0 : regner....
while (($i < $max))
do
        (( i += 1 ))
        (( sum += i  ))
done
echo $0, resultat: $sum
```

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

I/O prosess

- vanlige programmer som webbrowsere, teksteditorer, regneark etc bruker stort sett lite CPU
- venter på I/O(Input/Output), input fra brukere, nettverkskort eller disk.

branch prediction
Viktig å huske fra datamaski-narkitektur
CPU-løkke (hardware-nivå)
Microsoft og Unix OS
Microsoft Desktop-OS
Microsoft Server-OS
Unix operativsystemer
Multitasking
CPU-løkke (hardware-nivå)
Interrupts (avbrytelser)
Linux-prosess og RAM
Singletasking OS
Multitasking-OS
Multitasking
Context Switch
PCB -Process Control Block
Scheduling
PCB og Context Switch
Multitasking i praksis, CPU-intensive programmer

Multitasking eksempel

Bare rene regneprosesser bruker CPU hele tiden. Vanlige prosesser venter mye på I/O (Input/Output fra disk, nettverk etc.) og multitasking gir da mer effektiv utnyttelse av CPU.

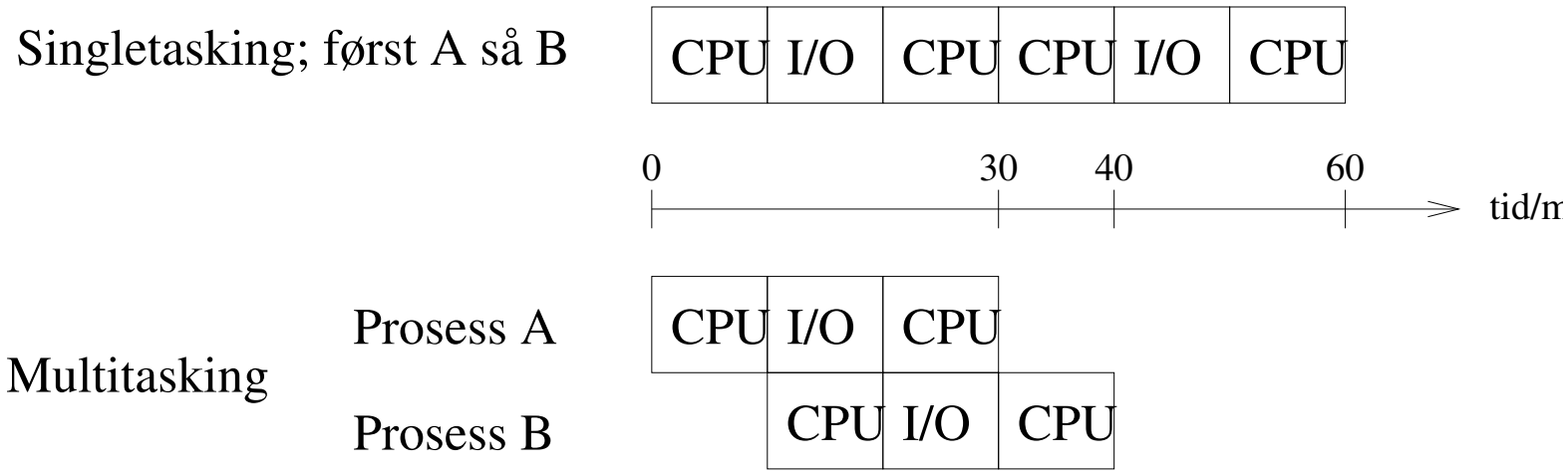


Figure: Prosessene A og B kjørt med single og multitasking