erasmus

HOGESCHOOL BRUSSEL

IT Essentials

Deel III: Operating Systems

3: Geheugenbeheer

INHOUD

- Introductie
- Address Spaces
- Swapping
- Virtual Memory

erasmus
HOGESCHOOL BRUSSEL

INTRODUCTIE

- Geheugen wordt beheerd door de memory manager
- Hardware component MMU
  - Vroeger deel van de northbridge
  - Tegenwoordig mee op de CPU
- Oorspronkelijk hadden programma's directe toegang tot het fysische geheugen = geen abstractie
  - Riskant
  - Moeilijk voor multiprogramming

erasmus
HOGESCHOOL BRUSSEL

ADDRESS SPACES

- Analoog met de abstractie van processen en threads
  - Virtuele cpu
- Memory
  - **Address space**
  - Virtueel
  - Geheel van geheugenadressen dat een process mag gebruiken
  - Elk process een eigen adress space
  - Threads binnen een process delen dezelfde address space

erasmus
HOGESCHOOL BRUSSEL

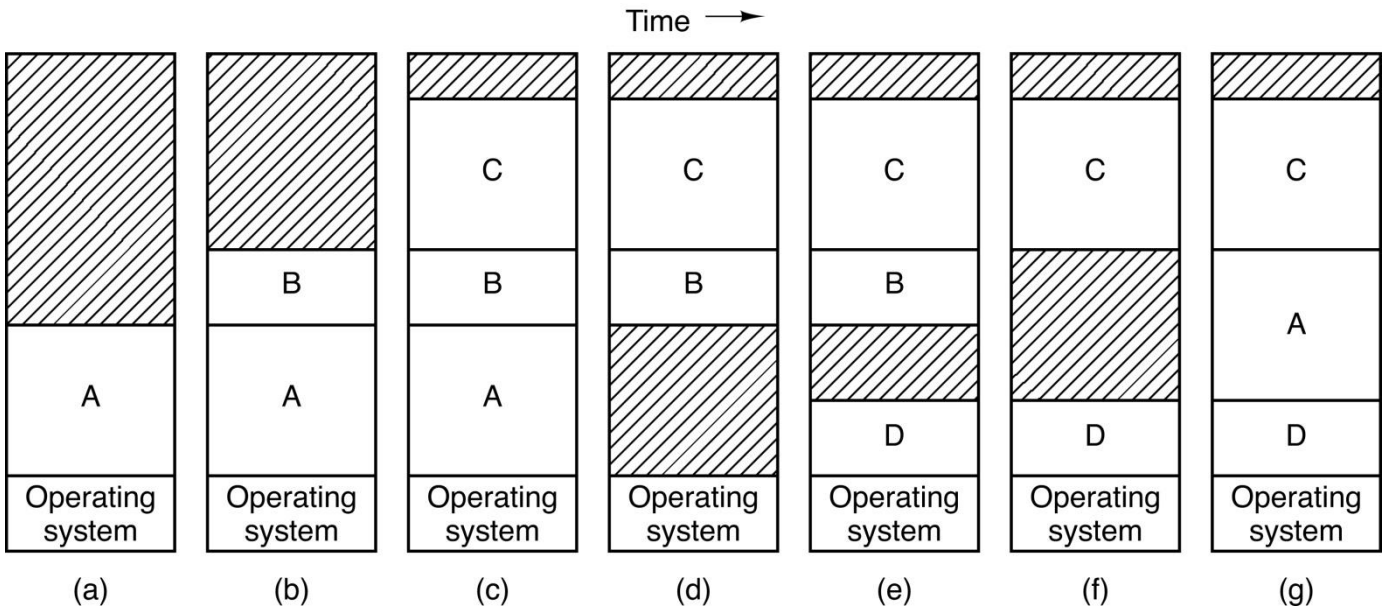
ADDRESS SPACES

- Huidige computers steeds te weinig RAM
  - Opstarten van gemiddelde PC 50-100 processen
  - Meerdere applicaties actief
- 2 oplossingen:
  - Swapping
  - Virtual Memory

erasmus HOGESCHOOL BRUSSEL

SWAPPING

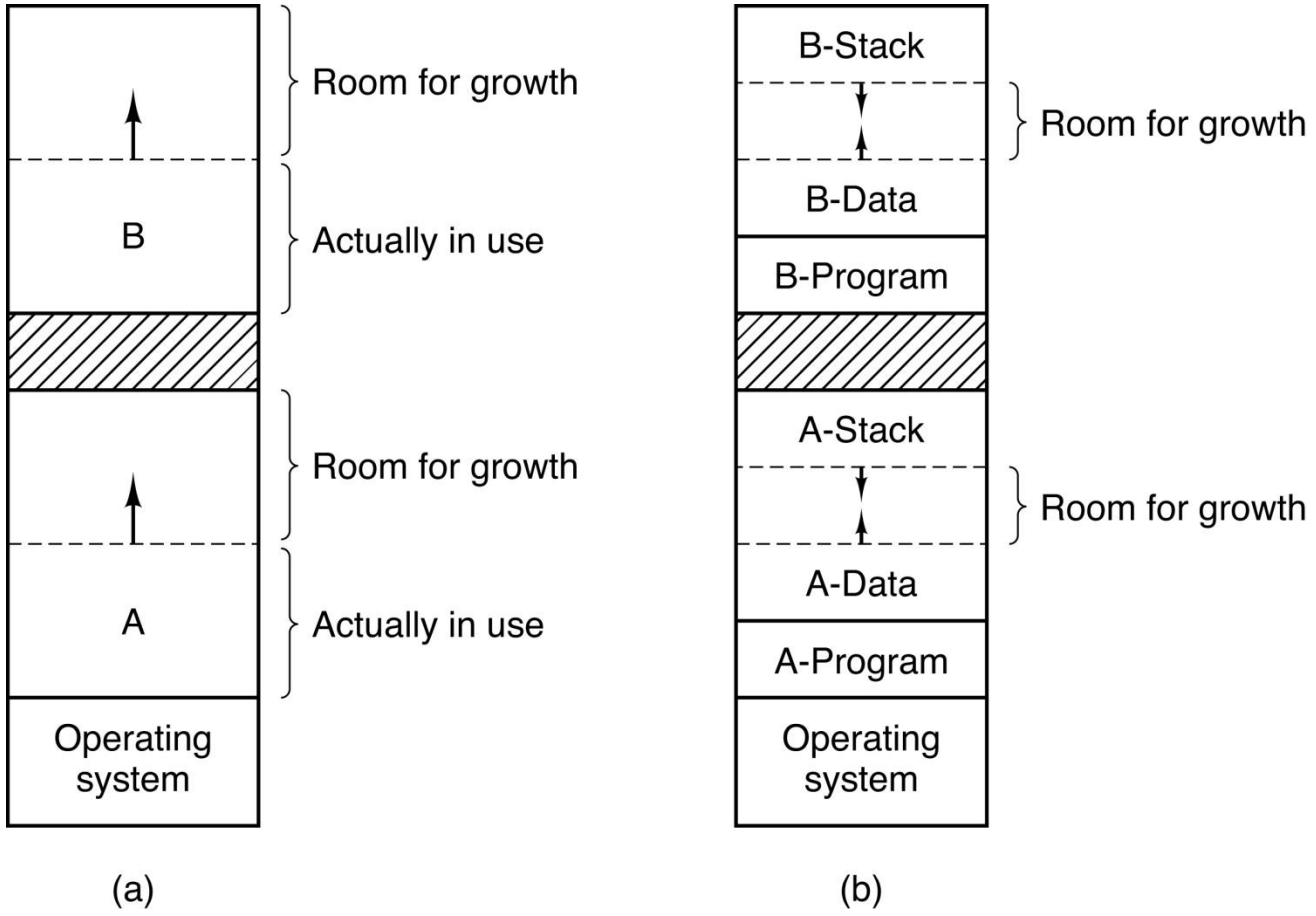
Niet actieve processen verplaatsen naar de HDD



SWAPPING

- Probleem bij swapping zijn de gaten die vallen in het geheugen
- Memory compaction
  - vraagt zeer veel CPU: weinig bruikbaar
  - Moeilijk bij groeiende processen: reserveruimte voorzien

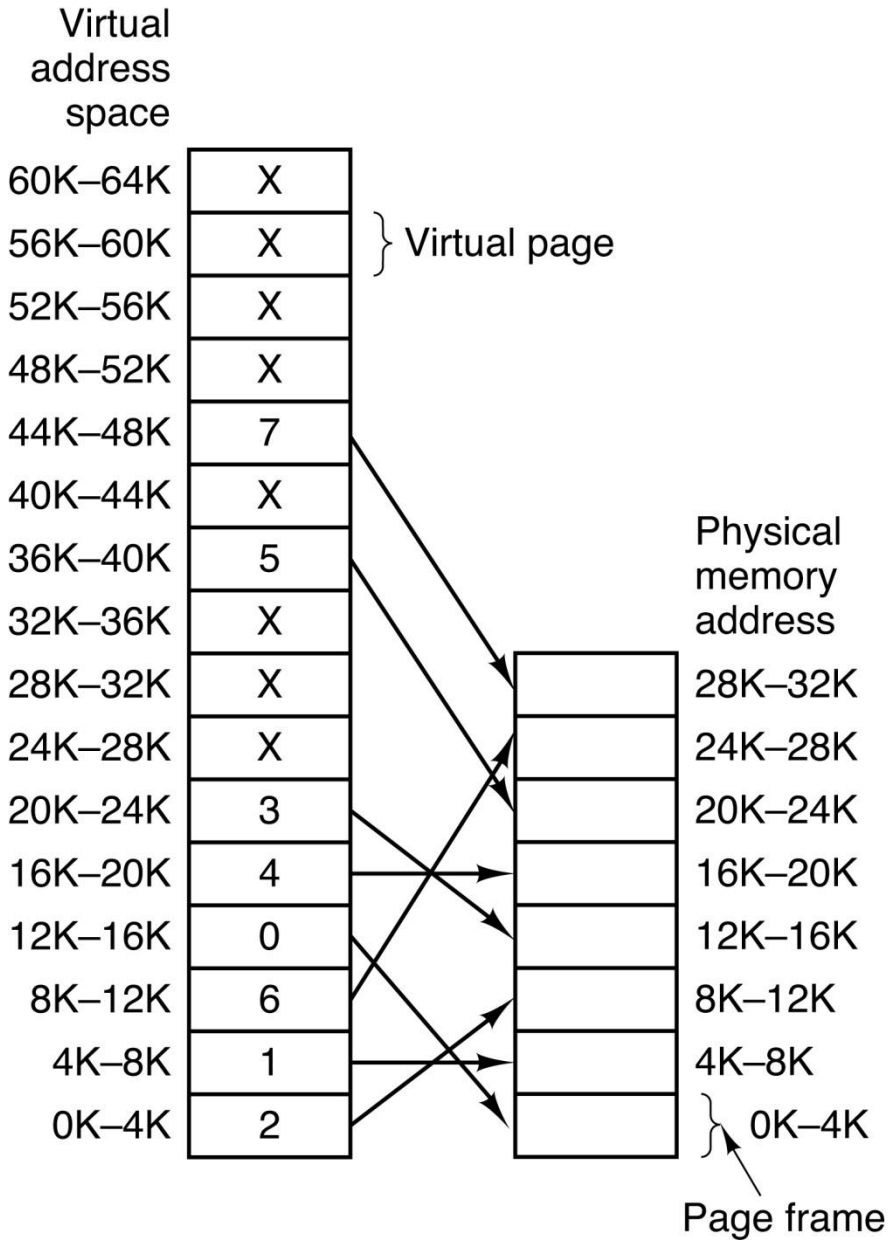
SWAPPING


Room for growth
B
Actually in use
Room for growth
A
Actually in use
Operating system
(a)
B-Stack
Room for growth
B-Data
B-Program
A-Stack
Room for growth
A-Data
A-Program
Operating system
(b)

VIRTUAL MEMORY

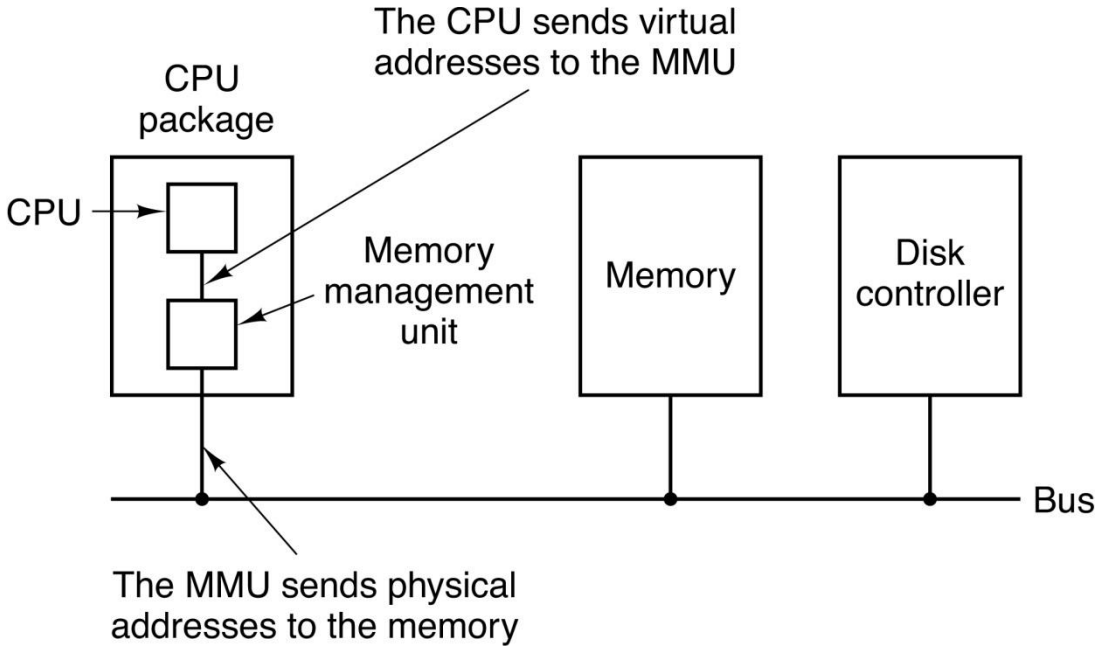
- Non-volatile storage snelheid te traag om via swapping geheugennoden op te vangen
- **Virtual memory**
  - Address spaces worden opgesplitst in **pages**
  - Elke page heeft een aaneensluitend adresblok
  - Worden op het fysieke geheugen geplaatst in corresponderende **page frames**
  - Niet alle pages moeten in het RAM zijn om het programma te kunnen laten draaien

erasmus
HOGESCHOOL BRUSSEL


Virtual address space
60K–64K
56K–60K
52K–56K
48K–52K
44K–48K
40K–44K
36K–40K
32K–36K
28K–32K
24K–28K
20K–24K
16K–20K
12K–16K
8K–12K
4K–8K
0K–4K
X
X
X
X
7
X
5
X
X
X
3
4
0
6
1
2
Virtual page
Physical memory address
28K–32K
24K–28K
20K–24K
16K–20K
12K–16K
8K–12K
4K–8K
0K–4K
Page frame

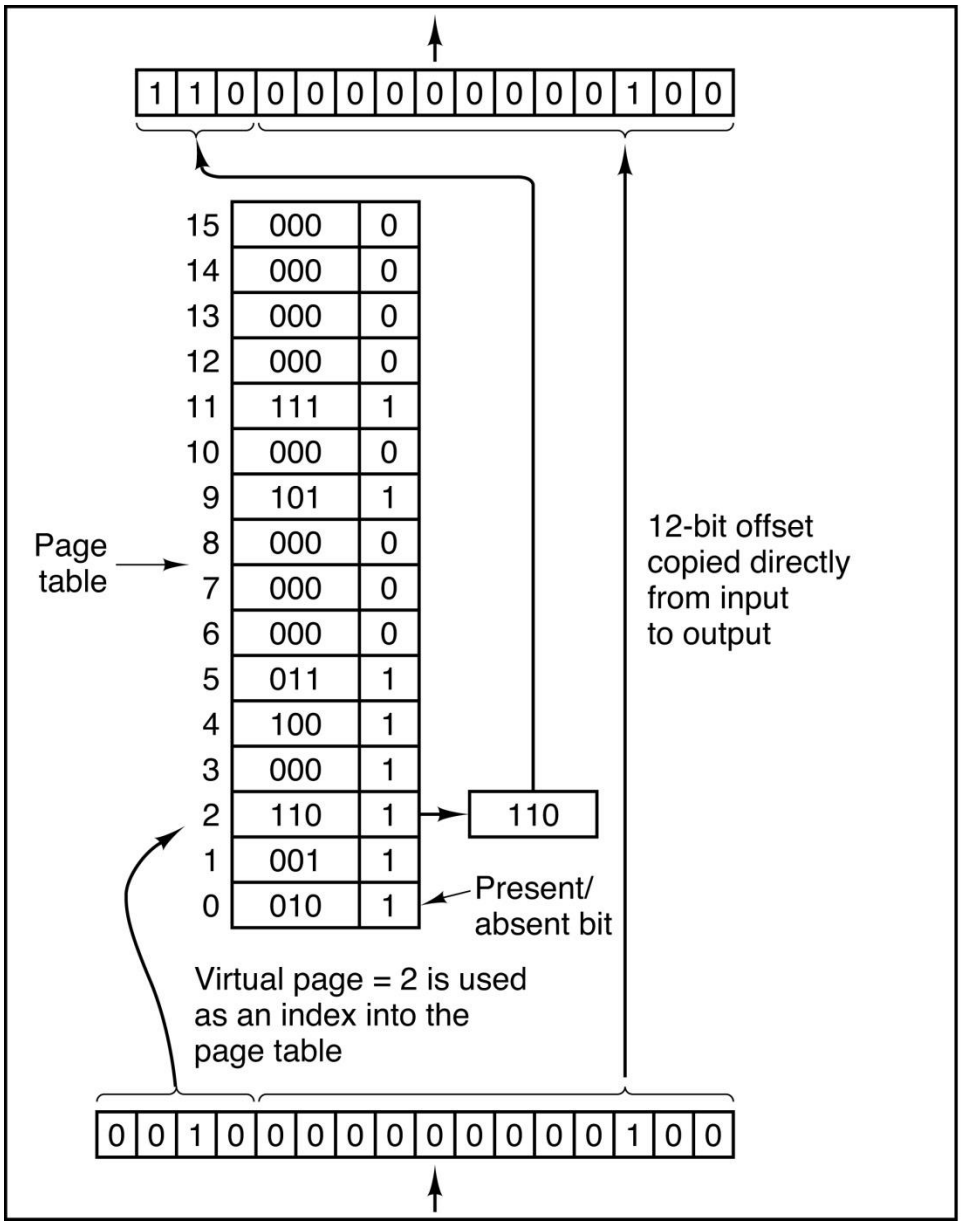
VIRTUAL MEMORY

- **Paging**: virtuele addressen worden aangemaakt voor het process om zo een virtuele address space te maken
- De omschakeling tussen virtuele en fysieke adressen gebeurt door de MMU



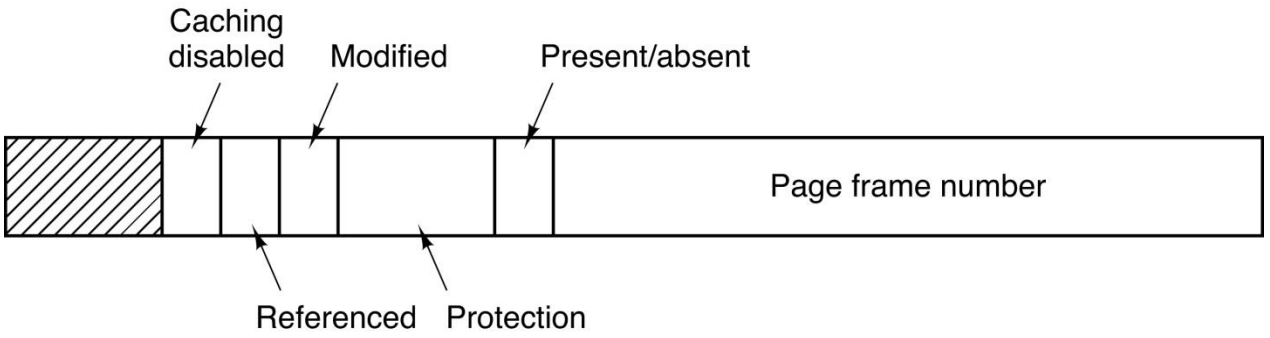
VIRTUAL MEMORY

- Een present/absent bit houdt bij of een page al dan niet gemapt staat op het fysiek geheugen
- Indien een absent page gevraagd wordt, wordt een trap naar het OS gestuurd: **page fault**
- een page frame wordt vrijgemaakt, de niet gemapte page van storage gehaald en in de vrije frame geplaatst
- Pages worden bijgehouden in de **page table**
- Page groottes zijn steeds machten van 2


Outgoing physical address (24580)
1 1 0 0 0 0 0 0 0 0 0 0 1 0 0
15 000 0
14 000 0
13 000 0
12 000 0
11 111 1
10 000 0
9 101 1
8 000 0
7 000 0
6 000 0
5 011 1
4 100 1
3 000 1
2 110 1
1 001 1
0 010 1
Page table
110
Present/ absent bit
12-bit offset copied directly from input to output
Virtual page = 2 is used as an index into the page table
0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0
Incoming virtual address (8196)

VIRTUAL MEMORY

- 1 page table verwijzing bevat o.a. volgende elementen
  - Referenced bit: is de page (recentelijk) nog aangesproken?
  - Modified: is de page gewijzigd? 'Dirty' frame
  - Hardware address van niet gemapte pages niet in de page table: aparte tabel in OS



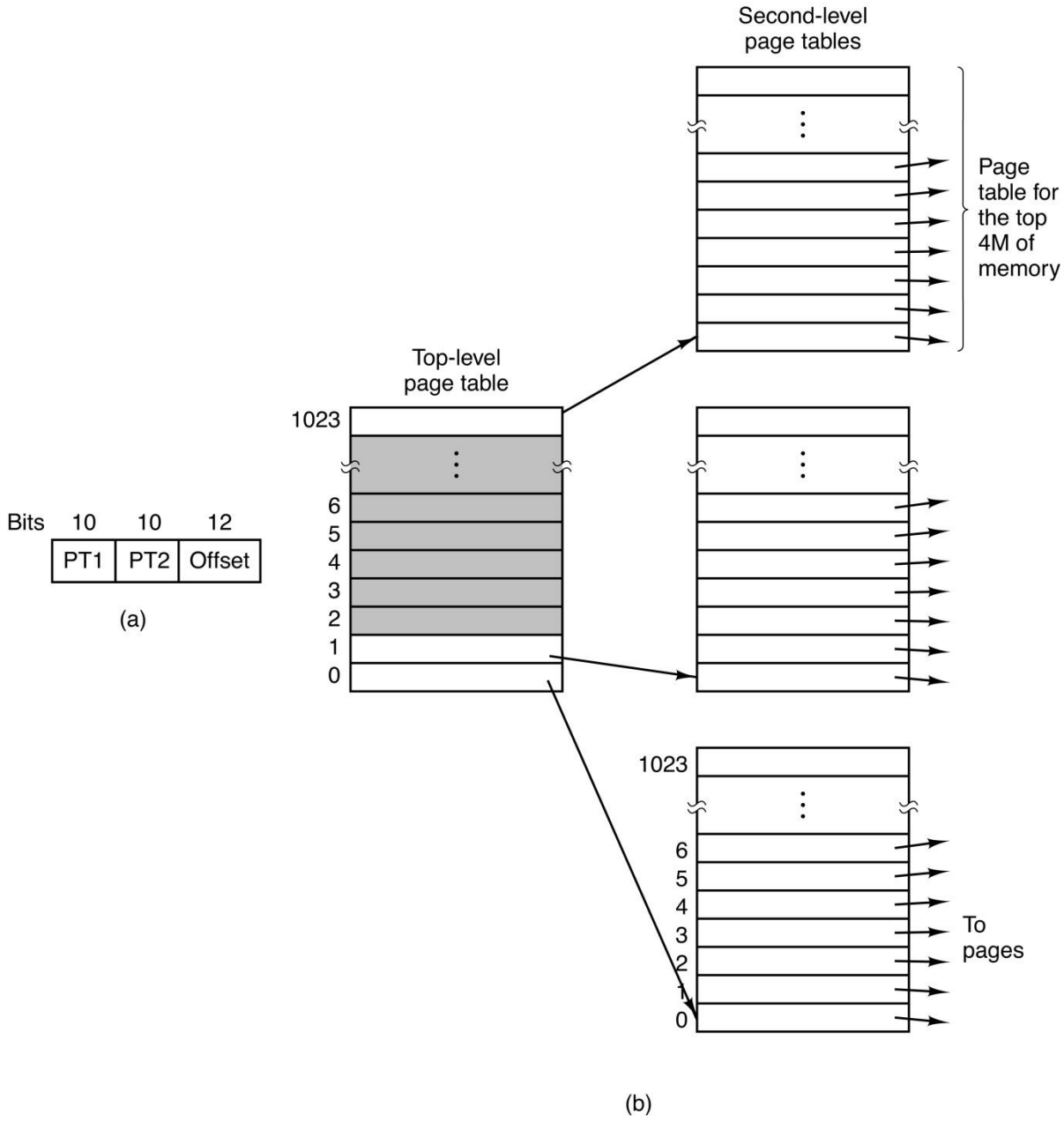
VIRTUAL MEMORY

- TLB
  - Bij elke memory reference: omzetting virtueel naar fysisch adres
  - Vaak meerdere per instructie
  - moet zeer snel kunnen verlopen
  - Meeste programma's gebruiken zeer vaak steeds dezelfde pages
  - Gebruik van cache: **Translation Lookaside Buffer (TLB)**
  - Parallelle (gelijktijdige) lookup in TLB
  - Tot 256 entries in TLB

VIRTUAL MEMORY

- Multilevel page tables
  - Sinds x64 zeer grote hoeveelheid RAM mogelijk => zeer grote page tables
  - page tables opsplitsen in partities of levels
  - Pages die niet nodig zijn worden niet in het geheugen bewaard
  - Kunnen al 4 levels diep gaan met 9 bits per level
    $2^9 x 2^9 x 2^9 x 2^9 x 2^{12} = 2^{48} = 256$ TB
  - Bijvoorbeeld page tables voor vrije reserve ruimte tussen data en stack

erasmus HOGESCHOOL BRUSSEL


Second-level
page tables
Page
table for
the top
4M of
memory
Top-level
page table
1023
6
5
4
3
2
1
0
Bits 10 10 12
PT1
PT2
Offset
(a)
1023
6
5
4
3
2
1
0
To
pages
(b)

VIRTUAL MEMORY

- Page replacement algoritmes
  - Welke frame mag leeggemaakt worden bij een page fault?
  - Meerdere mogelijkheden in de keuze van page frame via algoritmes
- Optimaal
  - Page die in de toekomst het laatst zal aangeroepen worden
  - Niet realiseerbaar: onmogelijk te weten welke page dit zal zijn
  - Kan gemeten worden bij een 2nd run en gebruikt worden als benchmark voor andere algoritmes

erasmus
HOGESCHOOL BRUSSEL

VIRTUAL MEMORY

- Not recently used (NRU)
  - Maakt gebruik van de Reference en Modified bit
  - R bit wordt op vaste tijdstippen terug op 0 gezet voor alle pages
  - 4 combinaties mogelijk
    - Class 0: R=0, M=0
    - Class 1: R=0, M=1
    - Class 2: R=1, M=0
    - Class 3: R=1, M=1
  - Willekeurige page uit laagst mogelijke klasse wordt verwijderd
  - Redelijke performantie, eenvoudig

erasmus HOGESCHOOL BRUSSEL

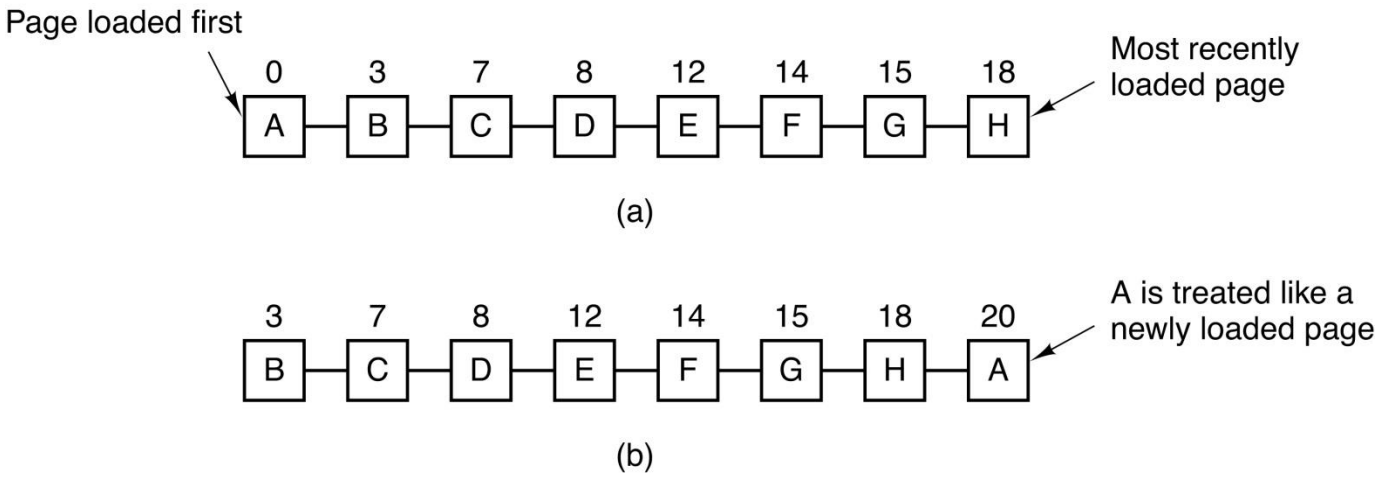
VIRTUAL MEMORY

- First-in, First-out (FIFO)
  - Oudste page wordt verwijderd
  - Kan nog steeds een veel gebruikte page zijn
  - Analogie: artikel in supermarkt dat al het langste wordt aangeboden wordt verwijderd

erasmus
HOGESCHOOL BRUSSEL

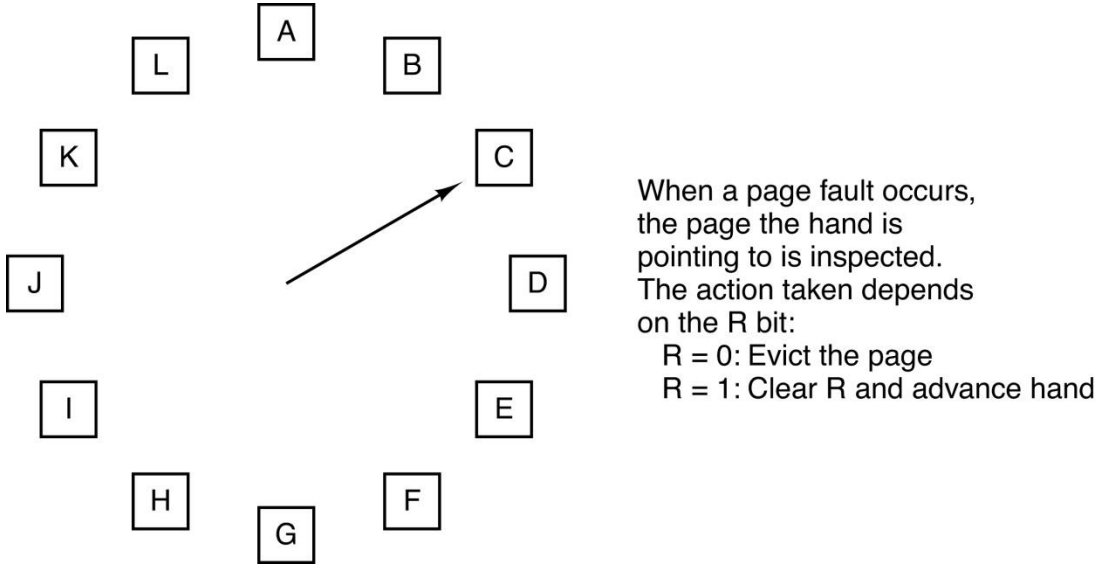
VIRTUAL MEMORY

- Second-chance
  - FIFO + R bit
  - Als R=0 : dadelijk vervangen
  - Als R=1 : 2de kans: page wordt de jongste met R op 0



VIRTUAL MEMORY

- Clock
  - = Second chance maar i.p.v. steeds pagina's rond te switchen (inefficiënt) worden de pagina's in in een cirkelvorm bijgehouden



VIRTUAL MEMORY

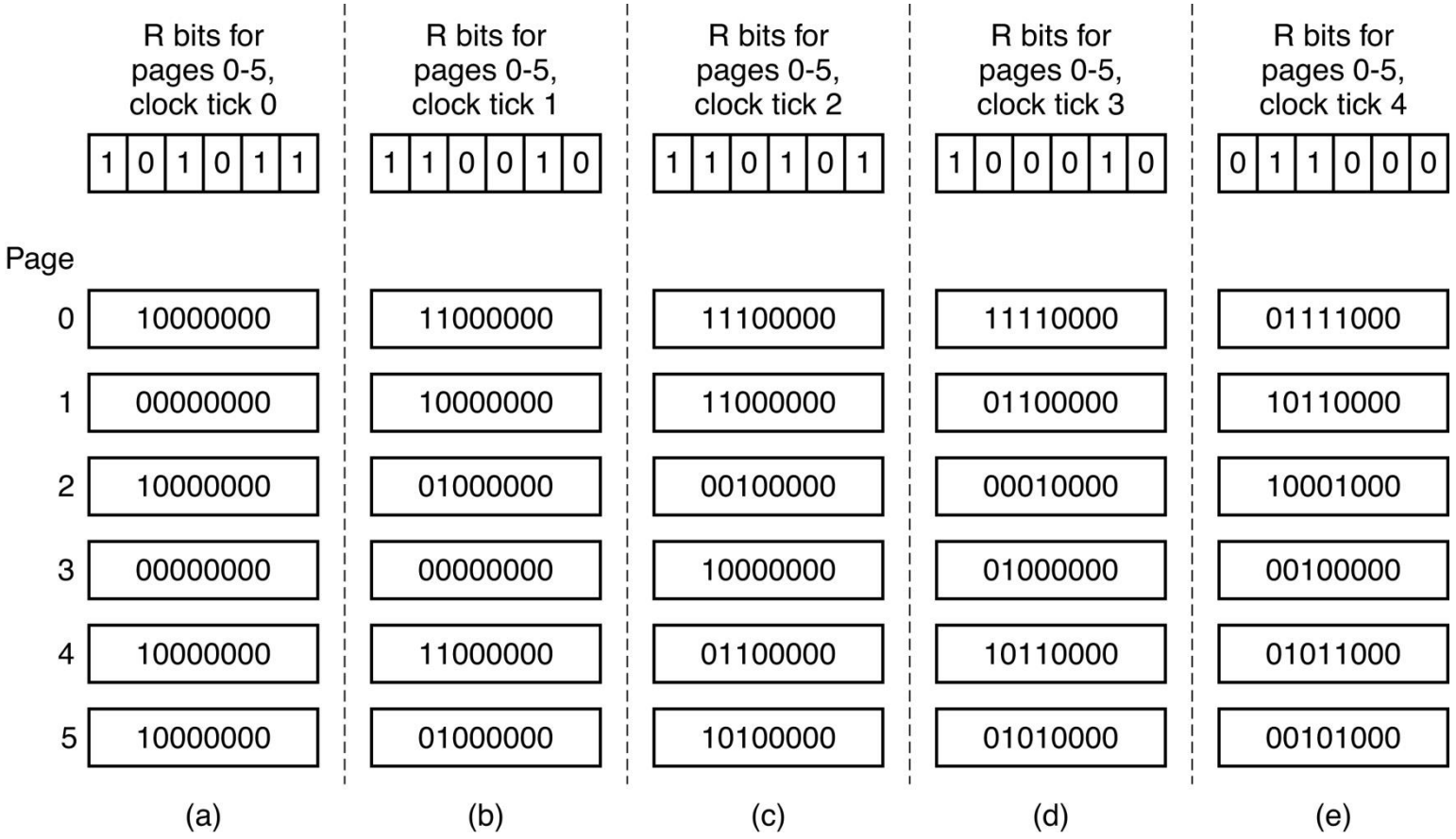
- Least recently used (LRU)
  - Pages die lange tijd niet gebruikt werden zullen in de toekomst waarschijnlijk ook niet snel nodig zijn
  - Pagina wisselen die lange tijd niet meer gebruikt werd
  - Vereist aparte tabel: gesorteerde tabel van alle processen in volgorde van gebruik
    - Dient bij elke memory reference aangepast te worden
    - Hardware optie: hardware counter bijhouden die bij elke instructie omhoog gaat en bijgehouden wordt voor elke table entry
    - Werd niet gerealiseerd

erasmus HOGESCHOOL BRUSSEL

VIRTUAL MEMORY

- Not frequently used (NFU)
  - Software oplossing voor LRU
  - Software counter bij elke table entry
  - Bij elke clock interrupt wordt R (0 of 1) opgeteld bij de counter
  - Probleem: pages die in het begin veel gebruikt werden, maar daarna niet meer, kunnen nog steeds hoge counts hebben
  - Oplossing **Aging**: r bit schuift steeds 1 positie op naar rechts en nieuwe r bits worden links toegevoegd
  - Meest recente gebruik genereert hoogste waarde: pagina met de laagste waarde wordt verwijderd

erasmus HOGESCHOOL BRUSSEL


R bits for pages 0-5, clock tick 0
R bits for pages 0-5, clock tick 1
R bits for pages 0-5, clock tick 2
R bits for pages 0-5, clock tick 3
R bits for pages 0-5, clock tick 4
1 0 1 0 1 1
1 1 0 0 1 0
1 1 0 1 0 1
1 0 0 0 1 0
0 1 1 0 0 0
Page
0 10000000 11000000 11100000 11110000 01111000
1 00000000 10000000 11000000 01100000 10110000
2 10000000 01000000 00100000 00010000 10001000
3 00000000 00000000 10000000 01000000 00100000
4 10000000 11000000 01100000 10110000 01011000
5 10000000 01000000 10100000 01010000 00101000
(a)
(b)
(c)
(d)
(e)

VIRTUAL MEMORY

- Working set
  - Programma's gebruiken doorgaans slechts een kleine fractie van alle pagina's in een bepaald stadium van executie
    = **Working Set**
  - Eens de working set is ingeladen: relatief weinig page faults tot volgende executie fase
  - Indien working set niet in de ram geraakt: massa's page faults: **thrashing**
  - bij het switchen van processen bij multiprogramming, eerst de working set inladen en dan het proces pas laten lopen =**Prepaging**
  - Vermijdt veel page faults

erasmus HOGESCHOOL BRUSSEL

VIRTUAL MEMORY

- Working set
  - Working set = set van pagina's gebruikt bij een vooraf vastgelegd aantal memory references of een vastgelegde tijd
  - Pagina verwijderen die niet in de working set zit
  - Nadeel: hele page table moet gescand worden om te weten wanneer een page met R=0 laatst gebruikt werd

VIRTUAL MEMORY

- WSClock
  - =working set + clock
  - Er wordt gezocht naar R=0 met M=0 om het swappen van dirty pages te vermijden

erasmus HOGESCHOOL BRUSSEL

Samenvatting page replacement algoritmes

Algorithm	Comment
Optimal	Not implementable, but useful as a benchmark
NRU (Not Recently Used)	Very crude approximation of LRU
FIFO (First-In, First-Out)	Might throw out important pages
Second chance	Big improvement over FIFO
Clock	Realistic
LRU (Least Recently Used)	Excellent, but difficult to implement exactly
NFU (Not Frequently Used)	Fairly crude approximation to LRU
Aging	Efficient algorithm that approximates LRU well
Working set	Somewhat expensive to implement
WSClock	Good efficient algorithm