

Luento 5

Tietokoneen rakenne

Muistin- hallinta

(Memory Management)

Stallings: Ch 8.3-8.6

- n Muistinhallintaongelma
- n Heittovaihto vs. virtuaalimuisti
- n Ohjelmisto- ja laitteistotuki
- n Esim: Pentium

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 1

Teemu's Cheesecake

Register, on-chip cache, memory, disk, and tape speeds relative to times locating cheese for the cheesecake you are baking...

hand

0.5 sec
(register)

table

1 sec
(cache)

refrigerator

10 sec
(memory)

moon

12 days
(disk)

Europa (Jupiter)

4 years
(tape)

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 2

Virtual Memory (virtuaalimuisti)

- n **Problem: How can I make my (main) memory as big as my disk drive?**
- n **Answer: Virtual memory**
 - u keep only most probably referenced data in memory, and rest of it in disk
 - § disk is much bigger and slower than memory
 - § address in machine instruction may be different than memory address
 - § need to have efficient address mapping
 - § most of references are for data in memory
 - u joint solution with HW & SW

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 3

Other Problems Often Solved with VM

- n If you must want to have many processes in memory at the same time, how do you keep track of memory usage?
- n How do you prevent one process from touching another process' memory areas?
- n What if a process needs more memory than we have?

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 4

Memory Management Problem

- n **How much memory for each process?**
 - u Is it fixed amount during the process run time or can it vary during the run time?
- n **Where should that memory be?**
 - u In a continuous or discontinuous area?
 - u Is the location the same during the run time or can it vary dynamically during the run time?
- n **How is that memory managed?**
- n **How is that memory referenced?**

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 5

Partitioning

- n **How much physical memory for each process?**
- n **Static (fixed) partitioning** (staattiset tai kiinteät partitiot)
 - u Amount of physical memory determined at process creation time
 - u Continuous memory allocation for partition
- n **Dynamic partitioning** (dynaamiset partitiot)
 - u Amount of physical memory given to a process varies in time
 - § Due to process requirements (of this process)
 - § Due to system (i.e., other processes) requirements

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 6

Static Partitioning

- Equal size - give everybody the same amount
 - Fixed size - big enough for everybody
 - too much for most
 - Need more? Can not run!
- Unequal size
 - sizes predetermined
 - Can not combine
- Variable size
 - Size determined at process creation time

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 7

Fragmentation

- Internal fragmentation (sisäinen pirstoutuminen)
 - unused memory inside allocated block
 - e.g., equal size fixed memory partitions
- External fragmentation (ulkoinen pirstoutuminen)
 - enough free memory, but it is splintered as many un-allocatable blocks
 - e.g., unequal size partitions or dynamic fixed size (variable size) memory partitions

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 8

Dynamic Partitioning

- Process must be able to run with varying amounts of main memory
 - all of memory space is not in physical memory
 - need some minimum amount of memory
- New process?
 - If necessary reduce amount of memory for some (lower priority) processes
- Not enough memory for some process?
 - reduce amount of memory for some (lower priority) processes
 - kick (swap) out some (lower priority) process

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 9

Address Mapping (1) (osoitteen muunnos)

Pascal, Java: `while (...)`
`X := Y+Z;`

Symbolic Assembler:
`loop: LOAD R1, Y`
`ADD R1, Z`
`STORE R1, X`

Textual machine language:
`1312: LOAD R1, 2510`
`ADD R1, 2514`
`STORE R1, 2600`
 (addresses relative to 0)

Execution time:
`101312: LOAD R1, 102510`
`ADD R1, 102514`
`ADD R1, 102600`
 (real, actual!)

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 10

Address Mapping (2)

Textual machine language: `1312: LOAD R1, 2510` (logical address)

Execution time:
`101312: LOAD R1, 102510` or
`101312: LOAD R1, 2510` (??)

physical address (constant?) logical addr

- Want: $R1 \leftarrow \text{Mem}[102510]$ or $\text{Mem}[2510]$?
 - Who makes the mapping? When?

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 11

Address Mapping (2)

- At program load time
 - Loader (lataaja)
 - Static address binding (staattinen osoitteiden sidonta)
- At program execution time
 - CPU
 - With every instruction
 - Dynamic address binding (dynaaminen osoitteiden sidonta)
 - Swapping
 - Virtual memory

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 12

Heittovaihto (swapping)

- Prosessilla yhtenäinen muistialue
 - Prosessi joko muistissa tai levyllä
 - Prosessinkuvaaja (PCB) aina muistissa
- Ajonalkainen osoitemuunnos
 - Looginen osoite \neq fyysinen muistiosoite
- Laitteistotuki = MMU
 - Kanta- ja rajarekisteri
 - "Bounds exceeded"-keskeytys
- KJ
 - Kirjanpito vapaista muistialueista
 - Prosessien siirto levyllä muistiin / muistista levyllä
 - Prosessin vaihto: kanta- ja rajarekisterin asetus
 - Virheellinen muistiviite: tapa prosessi

Lisätietoja
KJ-kurssilla

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 13

Virtual Memory Implementation (Virtuaalimuistitoteutus)

- Methods
 - Base and limit registers (kanta- ja rajarekisterit)
 - Segmentation (segmentointi)
 - Paging (sivutus)
 - Segmented paging, multilevel paging
- Hardware support
 - MMU - Memory Management Unit
 - Part of processor
 - Varies with different methods
 - Sets limits on what types of virtual memory (methods) can be implemented using this HW

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 14

Base and Limit Registers

- Continuous memory partitions
 - One or more (4?) per process
 - May have separate base and limit registers
 - Code, data, shared data, etc
 - By default, or given explicitly in each mem. ref.
- BASE and LIMIT registers in MMU
 - All addresses logical in machine instructions
 - Exec. time address mapping for address (x):
 - Check: $0 \leq x < LIMIT$
 - Physical address: $BASE + x$

Tuttua Tiro-kurssilla

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 15

Address Mapping using Base and Limit Registers (Osoitteenmuunnos rajarekistereitä käyttäen)

(Sta05 Operating Systems Fig 7.8)

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 16

Virtuaalimuisti

- Vain tarvittavat prosessin palat muistissa, ei tarvitse sijaia peräkkäin muistissa
 - Tarvenotto (Demand paging)
- Palojen koko?
 - Vakiokokoiset palat = Sivutus
 - Palojen koko vaihtelee = Segmentointi
 - Yhdistettynä = Sivutettu segmentointi
- KJ:n kirjanpito (OS bookkeeping)
 - Sivutilataulu (page frame table)
 - Mitkä sivutilat vapaita, mitkä varattuja?
 - Jokaisella prosessilla oma sivutaulu (page table)
 - Onko sivu muistissa vai levyllä? Presence-bitti
 - Missä sivutilassa sivu majoilee?
 - Muuta kontrollitietoa? Viitebitit: Modified, Referenced

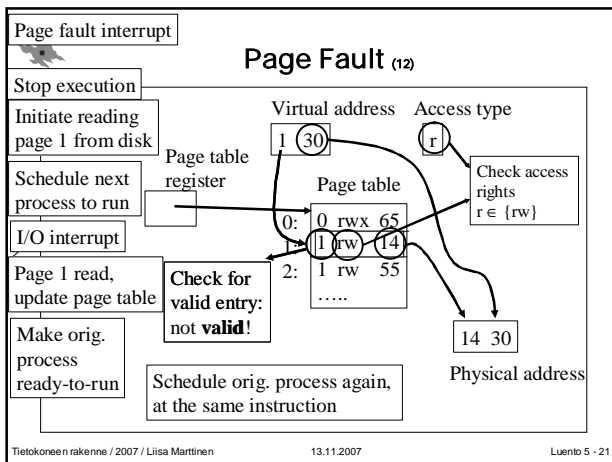
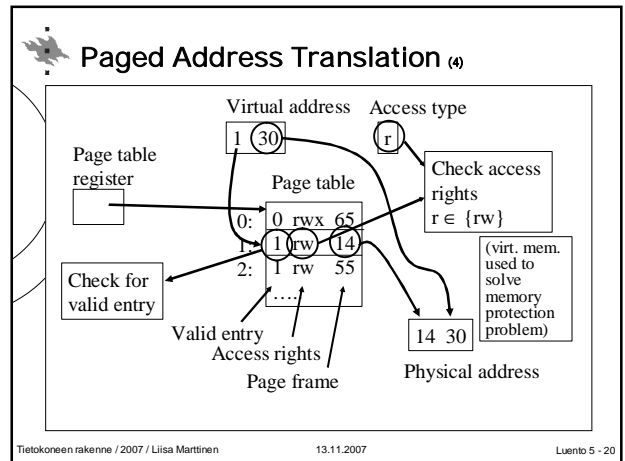
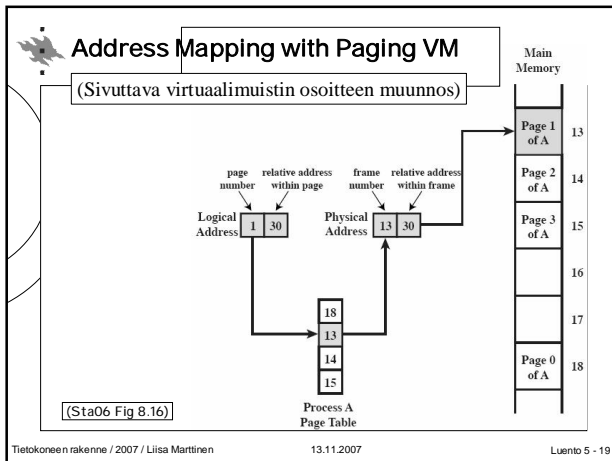
KJ kurssin pureskeltavaa
Sivutus "yleisintä" \emptyset nyt vain siitä

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 17

Virtual Memory: Paging (sivutus)

(Sta06 Fig 8.15)

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 18

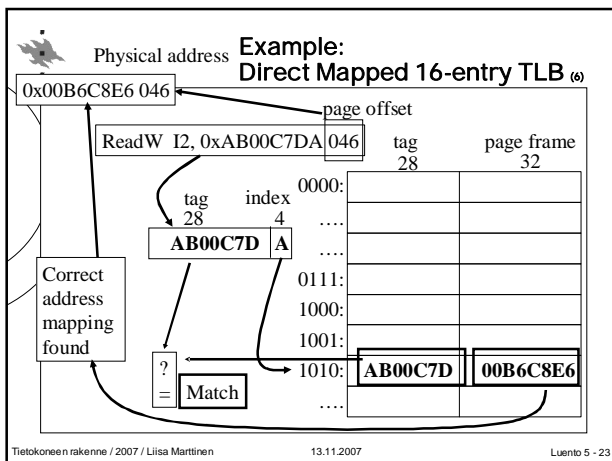


Virtuaalimuisti: TLB

Translation Lookaside Buffer

- Ositemuunnos jokaiselle muistivilltaukselle, vähintään kerran per käsky
- Sivutaulun alkio muistissa = ylimääräinen (useampi) muistinouto?
 - Liian hidasta
- Ratkaisu
 - Paikallisuus! Sitähän tarvitaan het'kohta uudestaan
 - Pida tallessa suorittimella edellisissa muunnoksissa käytetyt sivutaulun alkio
- TLB, ositemuunnospuskuri
 - Vrt. välimuisti
 - Nopea rekisterijoukko (esim. Pentium: 32 alkioita)
 - Assosiativinen haku
 - Osumatodennäköisyys 99.9% ? (eli lähes aina!)

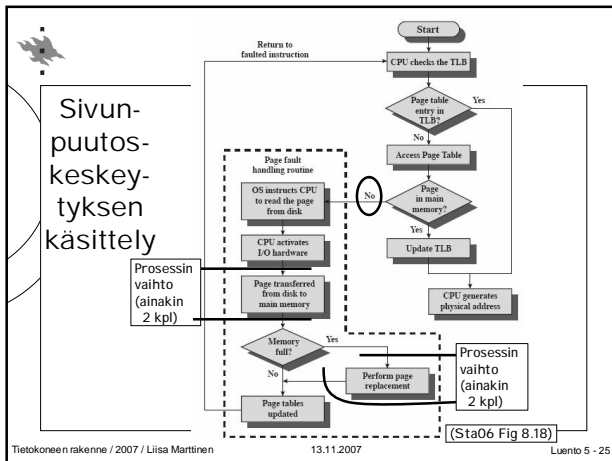
Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 22



Translation Lookaside Buffer (TLB)

- "Hit" on TLB?
 - address translation is in TLB - real fast
- "Miss" on TLB?
 - must read page table entry from memory
 - takes time - not much, just a memory reference
 - Entry might be in cache!
 - cpu waits idle until it is done
- Just like normal cache, but for address mapping
 - implemented just like cache
 - instead of cache line data have physical address
 - split TLB? 1 or 2 levels?

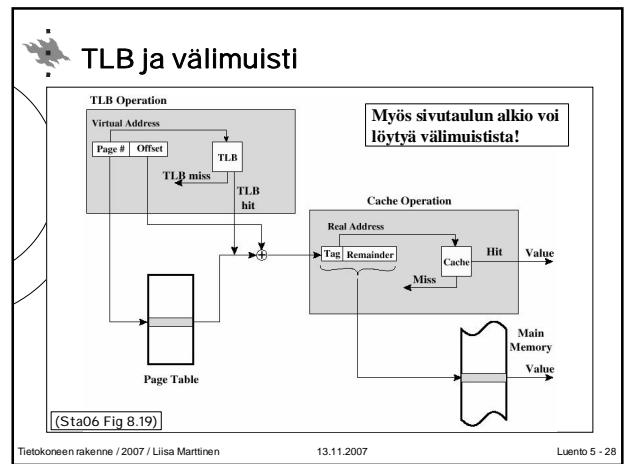
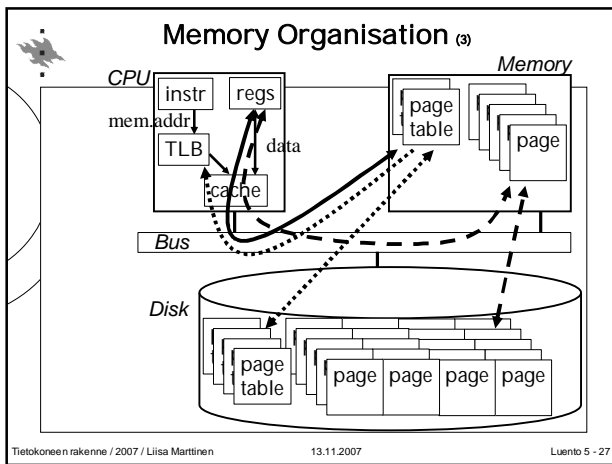
Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 24



Virtuaalimuisti

- Laitteistotuki = MMU ja sen erikoisrekisterit
 - PTR (page table register)
 - Sivutaulun fyysinen alkuosoite (tuotu PCB:stä)
 - TLB (translation lookaside buffer)
 - Aiemmissa osoitteenmuunnoksissa käytettyjä tietoja = sivutaulun alkioita
 - "Page fault" -keskeytys
 - Viitebittien ylläpito
- Prosessin vaihtuessa
 - PTR • sivutaulun fyysinen alkuosoite
 - TLB:n vanha sisältö invalidoitava
 - TLB:n alkoissa Validi-bitit
 - Muuttuneet alkiot takaisin muistiin ("cache block")

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 26



TLB vs. Cache

TLB Miss	Cache Miss
<ul style="list-style-type: none"> CPU waits idling HW implementation Invisible to process Data is copied from memory to TLB <ul style="list-style-type: none"> from page table data from cache? Delay 4 (or 2 or 8?) clock cycles 	<ul style="list-style-type: none"> CPU waits idling HW implementation Invisible to process Data is copied from memory to cache <ul style="list-style-type: none"> from page data Delay 4 (or 2 or 8?) clock cycles

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 29

TLB Misses vs. Page Faults

TLB Miss	Page Fault
<ul style="list-style-type: none"> CPU waits idling HW implementation Data is copied from memory to TLB (or from cache) Delay 1-4 (?) clock cycles 	<ul style="list-style-type: none"> Process is suspended and cpu executes some other process SW implementation Data is copied from disk to memory Delay 30 ms (?)

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 30

Korvauspolitiikka

- Mikä sivu korvataan muistista, jos muistitilasta puutetta?
 - Lokaalit / globaalit algoritmit
 - Prosessin omien sivujen joukosta
 - Kaikkein prosessien sivujen joukosta
 - Algoritmeja
 - Clock, Second change, LRU, ...
 - MMU
 - asetta viitattaessa Referenced=1,
 - asetta Modified=1, jos sivun sisältö muuttuu
 - KJ
 - Nollaa bitit aika-ajoin
 - Korvaa tarvittaessa sellainen, jossa R=0, M=0
 - M=1 kirjoita muuttunut sivu levyille ennen uusiokäyttöä

KJ-kurssilla tarkemmin

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 31

Käänteinen sivutaulu

Nykyprosessin oikea sivu?

- Järjestelmässä vain yksi käänteinen ST
- MMU: PTR (page table reg), PidR (process id register), TLB

(Sta06 Fig 8.17 täydennettynä)

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 32

Monitasoinen sivutaulu (3)

- Monet järjestelmät sallivat suuren virtuaaliosoitteavaruuden
 - Myös ST jaetaan sivuihin, ja ST:n osia levyillä
 - Ylimmän tason ST mahtuu yhteen sivuun, aina muistissa

(Sta05 Fig 8.4)

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 33

Virtual Memory Policies (3)

- Fetch policy (noutopolitiikka)
 - demand paging: fetch page only when needed 1st time
 - working set: keep all needed pages in memory
 - prefetch: guess and start fetch early
- Placement policy (sijoituspolitiikka)
- Replacement policy (poistopolitiikka)
 - local, consider pages just for this process for replacement
 - global, consider also pages for all other processes
 - dirty pages must be written to disk (likaiset, muutetut)

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 34

Tietokoneen rakenne

Esimerkiksi
Pentium (IA-32)

Ks. myös Tan06

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 35

Pentiumin tuki muistinhallinnalle

- Unsegmented unpaged, max $2^{32} = 4 \text{ GB}$
 - Virtuaaliosoite = fyysinen osoite
 - Tehokas käyttö reaaliaikajärjestelmissä
- Unsegmented paged (Sivuttava), max 4 GB
 - Lineaarinen osoiteavaruus
 - Sivu 4KB tai 4MB
 - Käyttöoikeudet sivukohtaisesti
- Segmented unpaged (Segmentoiva), max $2^{48} = 64 \text{ TB}$
 - Useita segmenttejä useita lineaarisia osoiteavaruuksia
 - Käyttöoikeudet segmenttikohtaisesti
- Segmented paged (Sivuttava segmentointi), max 64 TB
 - Muistinhallinta sivutusta käyttäen
 - Käyttöoikeudet segmenttikohtaisesti

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 36

Pentium: Osoitemuunnos

• Jos *Paging=Enabled*, käytä sivutauluja, muuten lineaarinen osoite = fyysinen osoite (KJ, esim. laiteajuri?)
 • Kontrollirekisterit ks. s. 444 [Sta06]

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 37

Pentium: Osoitemuunnos

Segment selector

- Global / Local Table
- Segmentin numero
 - Global/Local Descriptor Table (GDT/LDT)
- CS/DS selectors for code/data segments

Segment descriptor

Hietään GDT:sta tai LDT:sta ja tallennetaan MMU:n rekistereihin (Tan06 Fig 6-12, 6-13)

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 38

Pentium: Segmenttikuvaaja

Segment Descriptor (Segment Table Entry)

- Base**: Defines the starting address of the segment within the 4-GByte linear address space.
- D/B bit**: In a code segment, this is the D bit and indicates whether operands and addressing modes are 16 or 32 bits.
- Descriptor Privilege Level (DPL)**: Specifies the privilege level of the segment referred to by this segment descriptor.
- Granularity bit (G)**: Indicates whether the Limit field is to be interpreted in units by one byte or 4 KBytes.
- Limit**: Defines the size of the segment. The processor interprets the limit field in one of two ways, depending on the granularity bit: in units of one byte, up to a segment size limit of 1 MByte, or in units of 4 KBytes, up to a segment size limit of 4 GBytes.
- S bit**: Determines whether a given segment is a system segment or a code or data segment.
- Segment Present bit (P)**: Used for nonpaged systems. It indicates whether the segment is present in main memory. For paged systems, this bit is always set to 1.
- Type**: Distinguishes between various kinds of segments and indicates the access attributes.

(Sta06 Table 8.5)

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 39

Pentium: Sivutaulu

Page Directory Entry and Page Table Entry

- Accessed bit (A)**: This bit is set to 1 by the processor in both levels of page tables when a read or write operation to the corresponding page occurs.
- Dirty bit (D)**: This bit is set to 1 by the processor when a write operation to the corresponding page occurs.
- Page Frame Address**: Provides the physical address of the page in memory if the present bit is set. Since page frames are aligned on 4K boundaries, the bottom 12 bits are 0, and only the top 20 bits are included in the entry. In a page directory, the address is that of a page table.
- Page Cache Disable bit (PCD)**: Indicates whether data from page may be cached.
- Page Size bit (PS)**: Indicates whether page size is 4 KByte or 4 MByte.
- Page Write Through bit (PWT)**: Indicates whether write-through or write-back caching policy will be used for data in the corresponding page.
- Present bit (P)**: Indicates whether the page table or page is in main memory.
- Read/Write bit (RW)**: For user-level pages, indicates whether the page is read-only access or read/write access for user-level programs.
- User/Supervisor bit (US)**: Indicates whether the page is available only to the operating system (supervisor level) or is available to both operating system and applications (user level).

(Sta06 Table 8.5)

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 40

Pentium: Käyttöoikeudet

Privilege level CPU:n tilarekisterissä PSW:ssä

- 00=korkein, 11 = matalin
- Korkeampi voi vitata alemmille
- Etuoikeudet käskyt vain, kun level=00

Käyttöoikeus rajattavissa

- Segmentin valinta
 - RPL, requested privilege level
- Segmenttikuvaaja
 - DPL, descriptor privilege level
 - Type: koodi/data? R/W
- Sivutaulu
 - R/W-bitti

Linux ja Windows: vain kaksi käytössä


(Tan06 Fig 6-16)

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 41

Hennessy-Patterson: Computer Architecture, Fig 5.47 Alpha AXP

- Instr. TLB**: fully assoc, 12 entries
- Data TLB**: fully assoc, 32 entries
- Instr. CACHE**: direct mapped, 8 KB, 256 lines (a'32B)
- Data CACHE**: direct mapped, 8 KB, 256 lines
- Unified Level 2 CACHE**: 2 MB, 64K lines (a'32B) direct mapped, write-back
- Main Memory**
- Disk**

Tietokoneen rakenne / 2006 / Teemu Kerola

 **Kertauskysymyksiä**

- n Mitä laitteistotason tukea tarvitaan VM:n toteuttamiseksi?
- n Miten sivutus ja segmentointi eroavat toisistaan?
- n Miksi ne joskus yhdistetään?
- n Miten TLB ja välimuisti suhtautuvat toisiinsa?

Tietokoneen rakenne / 2007 / Liisa Marttinen 13.11.2007 Luento 5 - 43