

CS241 System Programming Memory Management (III)

Klara Nahrstedt

Lecture 30

4/7/2006



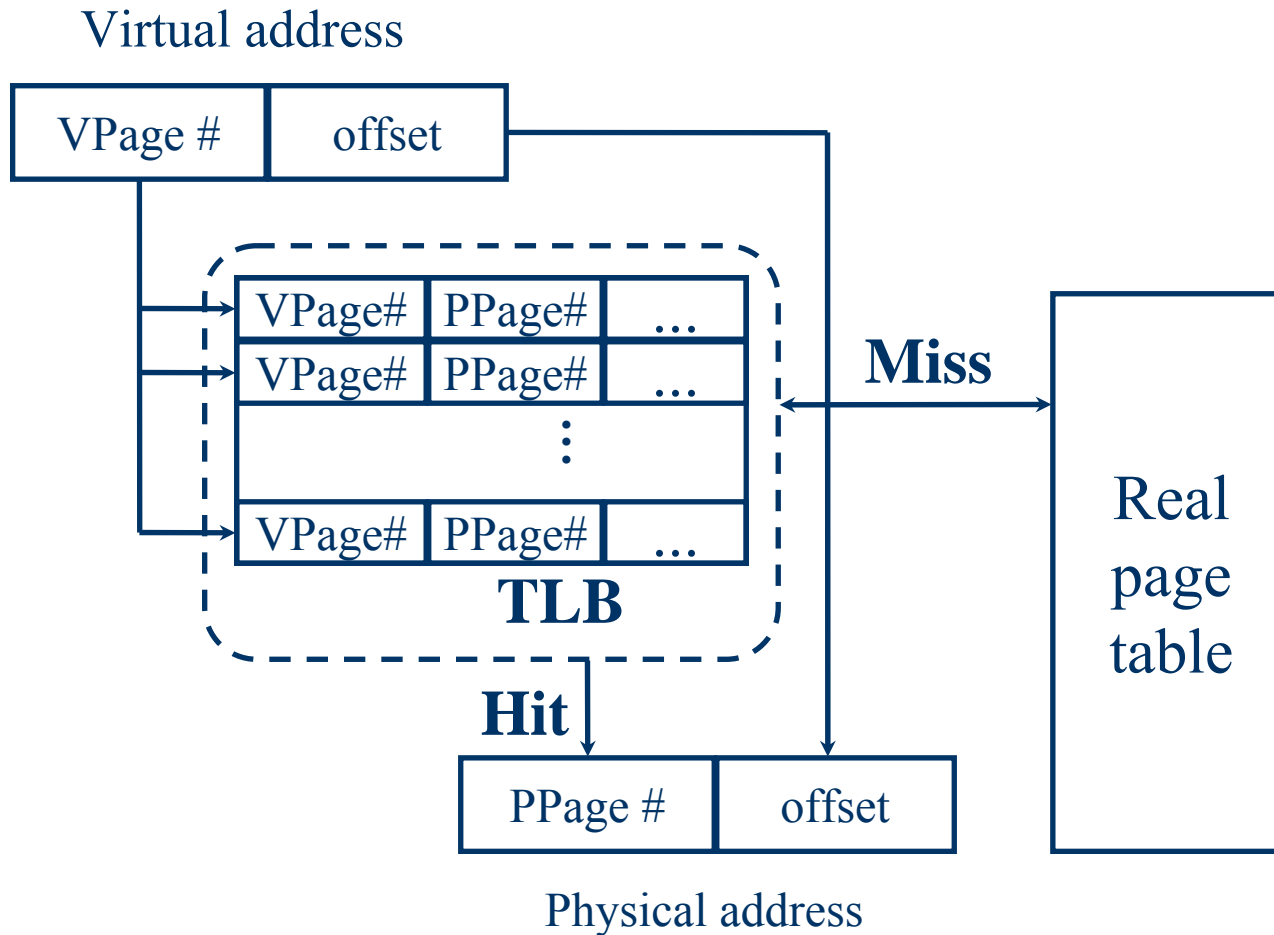
Content

- Paging Implementation
- TLB – Translation Look-aside Buffer
- Inverted Page Tables

Administrative

- MP4 is posted, due April 17, 2006
- Quiz 8 is April 7, 2006

Translation Lookaside Buffer (TLB)



TLB Function

- If a virtual address is presented to MMU, the hardware checks TLB by comparing all entries simultaneously (**in parallel**).
- If match is valid, the page is taken from TLB without going through page table.
- If match is not valid
 - MMU detects miss and does an ordinary page table lookup.
 - It then evicts one page out of TLB and replaces it with the new entry, so that next time that page is found in TLB.

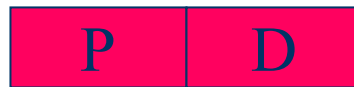
Page Mapping Hardware

Virtual Memory Address (P,D)

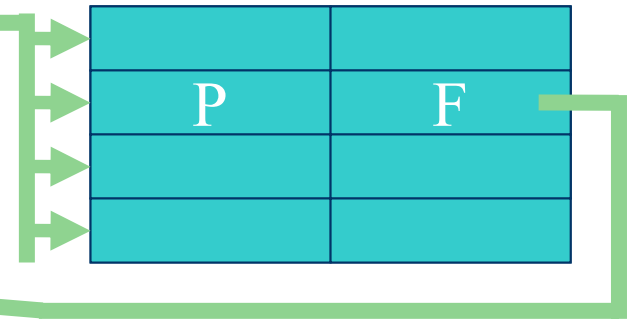
Page Table

P

0	
1	
0	
1	P → F
1	
0	
1	



Associative Lookup



First



Physical Address (F,D)



Page Mapping Example

Virtual Memory Address (P,D)

Page Table

4

0	
1	
0	
1	004 → 009
1	
0	
1	

004 006

First

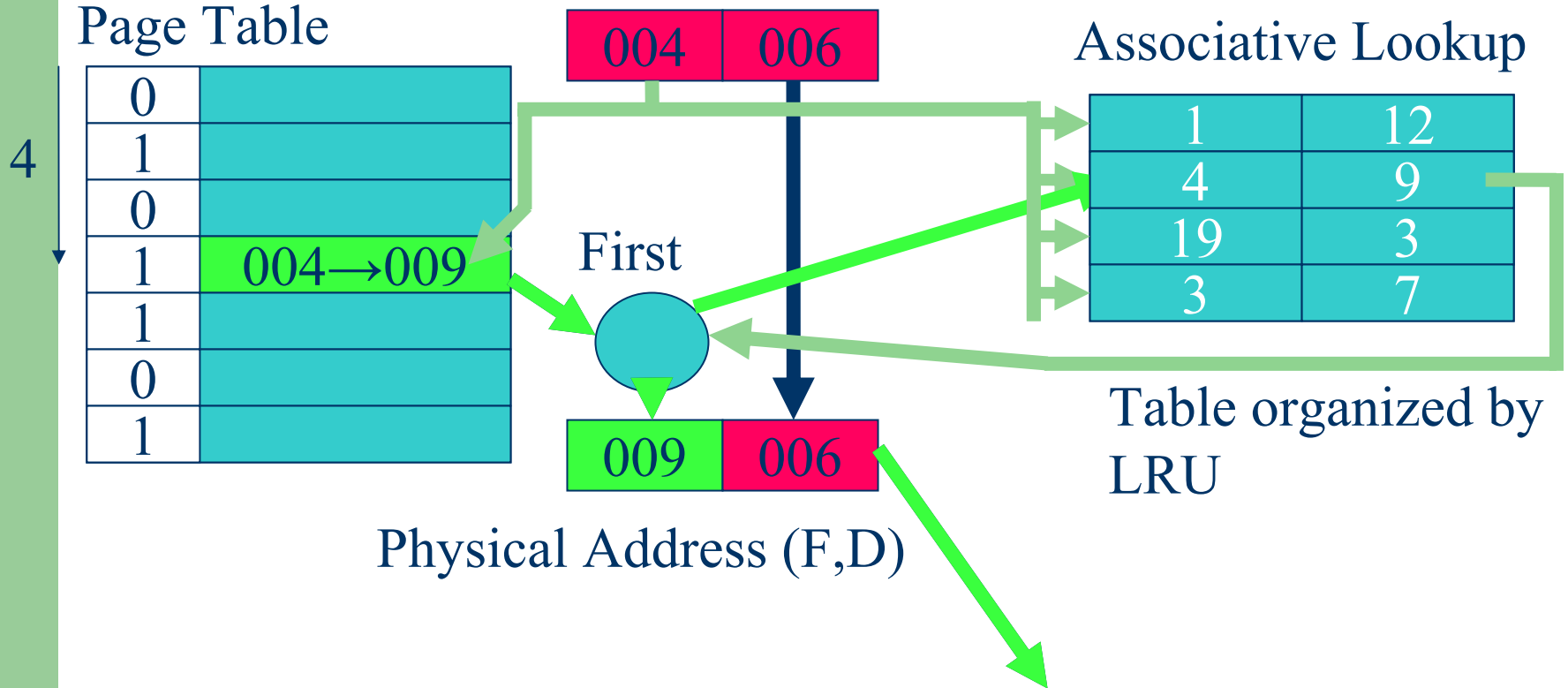
009 006

Physical Address (F,D)

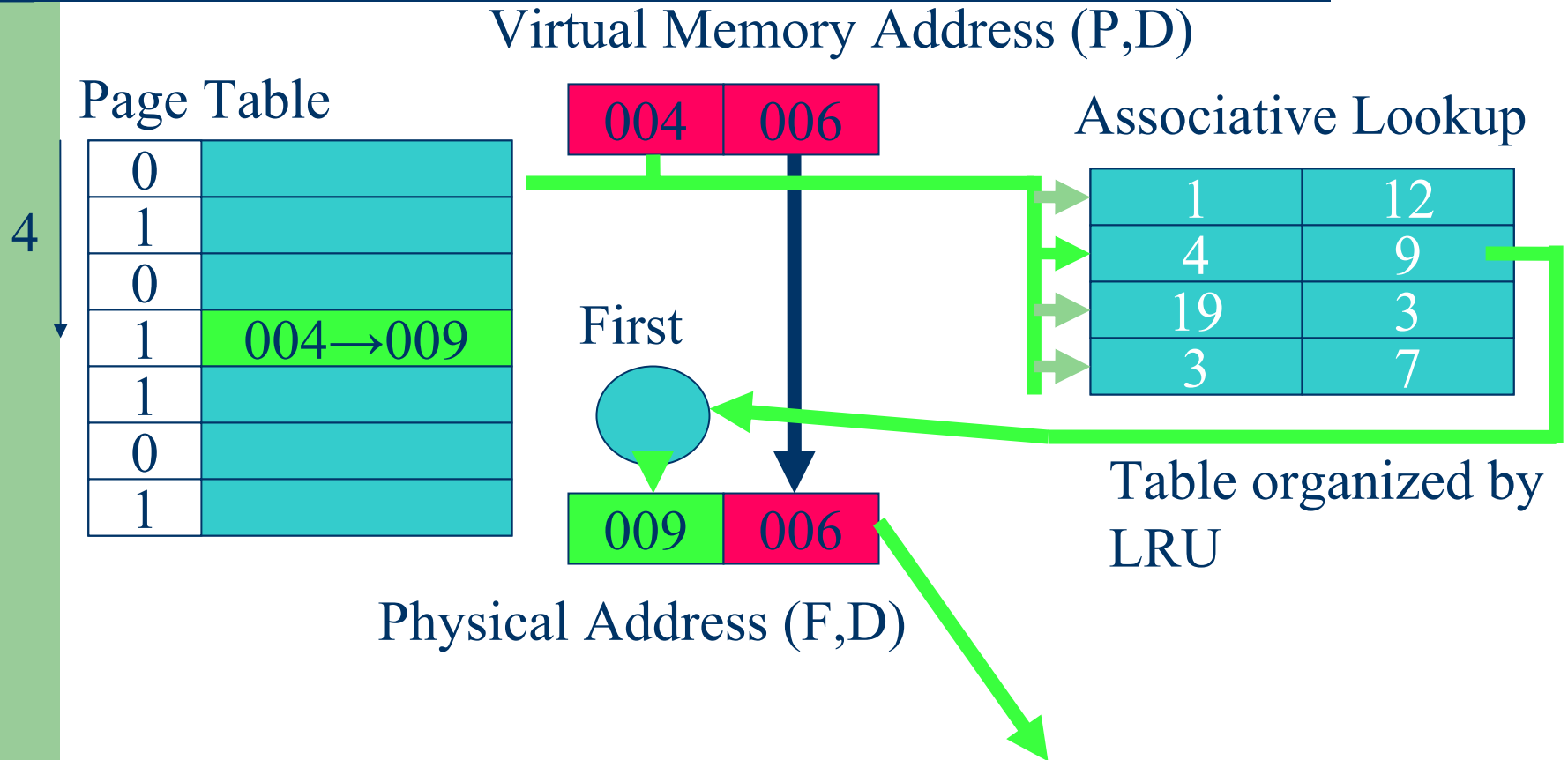
Associative Lookup

1	12
4	9
19	3
3	7

Table organized by LRU

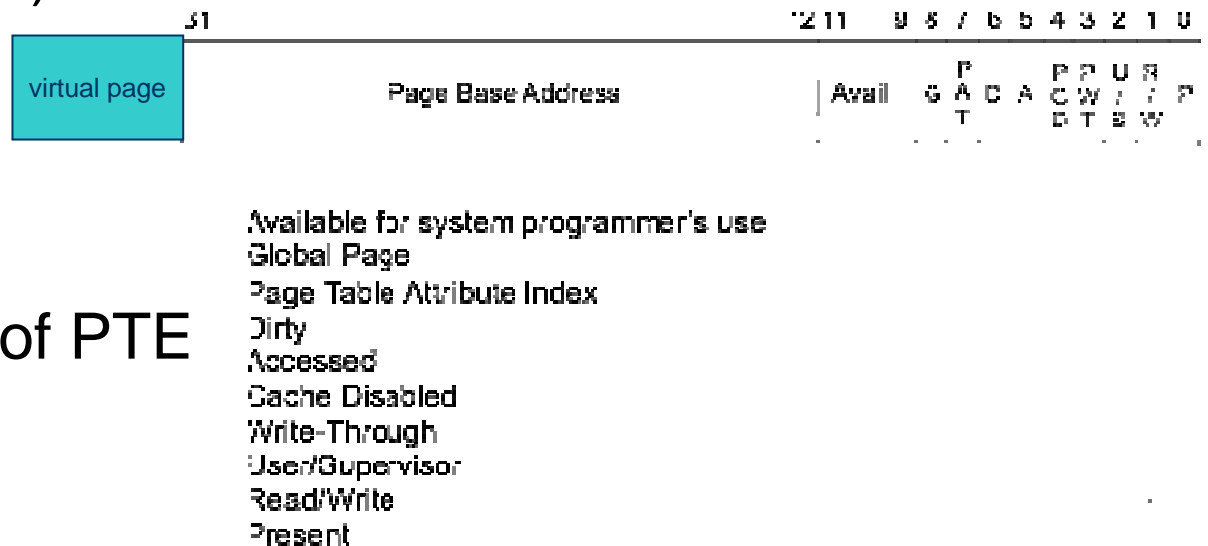


Page Mapping Example: next reference



Bits in a TLB Entry

- Common (necessary) bits
 - Virtual page number: match with the virtual address
 - Physical page number: translated address
 - Valid
 - Access bits: kernel and user (nil, read, write)
- Optional (useful) bits
 - Process tag
 - Reference
 - Modify
 - Cacheable
- Includes parts of PTE
 - Example: x86



Paging Implementation Issues

- TLB can be implemented using
 - Associative registers
 - Look-aside memory
 - Content-addressable memory
- TLB hit ratio (Page address cache hit ratio)
 - Percentage of time page translation found in associative memory

Software-Controlled TLB

- On a miss in TLB, **VM software**
 - Write back if there is no free entry
 - Check if the page containing the PTE is in memory
 - If no, perform page fault handling
 - Load the PTE into the TLB
 - Restart the faulting instruction
- On a hit in TLB, the hardware checks valid bit
 - If valid, pointer to page frame in memory
 - If invalid, the hardware generates a page fault
 - Perform page fault handling
 - Restart the faulting instruction
- Example: SPARC, MIPS, Alpha, HP-PA, IA-64, PowerPC

Issues

- What TLB entry to be replaced?
 - Random
 - Pseudo Least Recently Used (LRU)
- What happens on a context switch?
 - Process tag: change TLB registers and process register
 - No process tag: Invalidate the entire TLB contents
- What happens when changing a page table entry?
 - Change the entry in memory
 - Invalidate the TLB entry

Effective Access Time

- TLB lookup time = ε time unit
- Memory cycle = $m \mu\text{s}$
- TLB Hit ratio = α
- Effective access time (Eat)
 - $Eat = (m + \varepsilon)\alpha + (2m + \varepsilon)(1 - \alpha)$
 - $Eat = 2m + \varepsilon - m\alpha$

Summary

- TLB allows for rapid mapping of virtual addresses to physical addresses without going through the page table
- TLB management possible in software