Karlsruhe Institute of Technology

Operating Systems 2016/17
Solutions for Assignment 11

Prof. Dr. Frank Bellosa
Dipl.-Inform. Marc Rittinghaus

T-Question 11.1: Paging

Consider a system that translates virtual addresses to physical addresses using two-level page tables in hardware. Every page table comprises 1024 entries, with each entry having a size of 4 bytes and providing read/write page protection as presented in the lecture. The page size is 4096 bytes. The system neither possesses a cache nor a TLB.

a. How many memory accesses are necessary to read a contiguous buffer of 8 MiB, starting at offset 0, reading 4 bytes at a time. **1 T-pt**

Solution:
The number of memory accesses is the sum of the accesses necessary to (a) read the actual buffer and (b) to read the page tables for address translation.

(a) 8 MiB / 4 bytes = 2 Mi

*(b) For every memory access of the buffer, we need to read the first- and second-level page tables, giving us two extra accesses per buffer access: 2 Mi * 2*

*Total: 2 Mi * 3 = 6 Mi*

b. How many memory accesses are required if a TLB is added to the system? **1 T-pt**

Solution:
Compared to the previous question, we only need to traverse the page table hierarchy (2 accesses) once for every page. The buffer occupies 8 MiB / 4 KiB = 2 Ki pages.

*Giving us a total of 2 Mi + 2 Ki * 2 accesses.*

c. How can shared memory between two processes A and B be realized an the page table level? **1 T-pt**

Solution:
The PTEs for the two pages in process A and B are configured to map to the same physical frame, thus allowing both processes to work on the same memory.

d. How does the page protection in the PTE for a copy-on-write page need to be configured? Explain your answer! **1 T-pt**

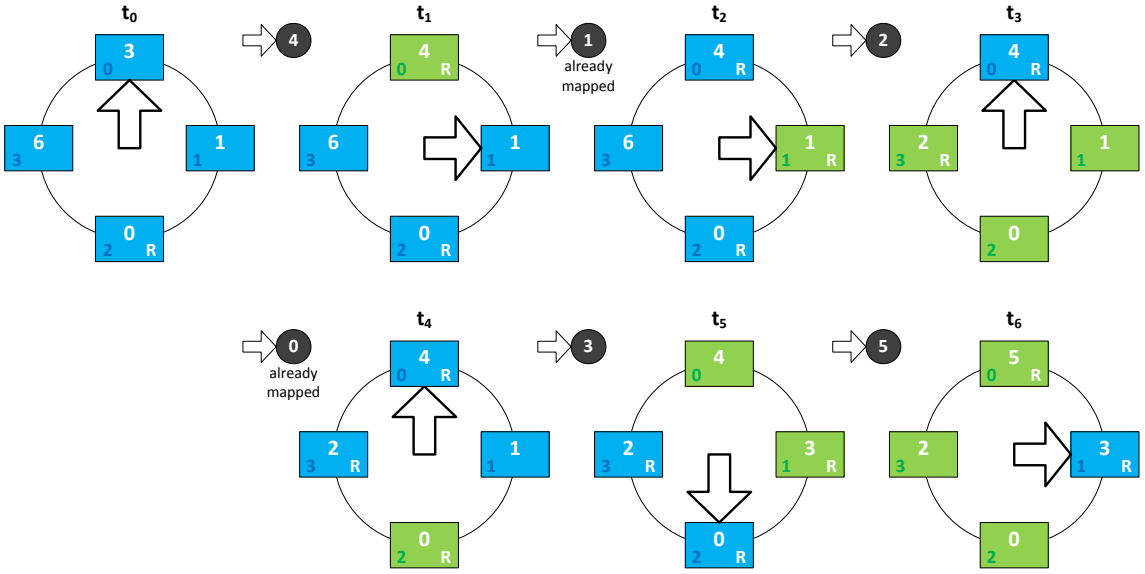
Solution:
The page need to be configured as read-only, so the CPU generates a page fault when a process performs a write access. The OS can then copy the page, map the private version and restart the failed write instruction.

T-Question 11.2: Page Replacement

a. Consider a system with 4 page frames. Complete the mapping table for a process that accesses pages in the given order if clock page replacement is used. Assume the circular buffer of the clock to be in ascending order (i.e., frame 0, 1, 2, 3), the clock hand to be positioned at frame 0 and the reference bit for page 0 to be set. Reference string for virtual page numbers (VPN): **4 1 2 0 3 5** **3 T-pt**

Solution:

frame	$VPN(t_0)$	$VPN(t_1)$	$VPN(t_2)$	$VPN(t_3)$	$VPN(t_4)$	$VPN(t_5)$	$VPN(t_6)$
0	*3*	*4*	*4*	*4*	*4*	*4*	*5*
1	*1*	*1*	*1*	*1*	*1*	*3*	*3*
2	*0*	*0*	*0*	*0*	*0*	*0*	*0*
3	*6*	*6*	*6*	*2*	*2*	*2*	*2*



b. What difficulty do you see when implementing the clock algorithm for systems that allow shared memory? **1 T-pt**

Solution:
The clock algorithm uses the reference bit of a page to determine if the page should be replaced. However, while the reference bit is present per virtual page and not per frame (i.e., every PTE has its own reference bit), the MMU only sets the reference bit in the PTE that it used for translation.

Therefore, to determine if the data of a frame is still used, it is not sufficient to look at a single PTE, but the OS has to check all PTEs that map the same frame. Without having an extra data structure, which maintains the list of relevant PTEs, the OS needs to scan all page tables.

Total: 8 T-pt