

DEPARTMENT OF COMPUTER SCIENCE

TDT4186 - OPERATING SYSTEMS

Exercise 4

 $Paging,\ Page\ Tables\ \ \ \ Translation\ Lookaside\ Buffers$

Professor:
Di Liu
Teaching Assistant:
Roman K. Brunner
Student Assistants:
Eik Hvattum Røgeberg
Halvor Linder Henriksen
Ole Ludvig Hozman

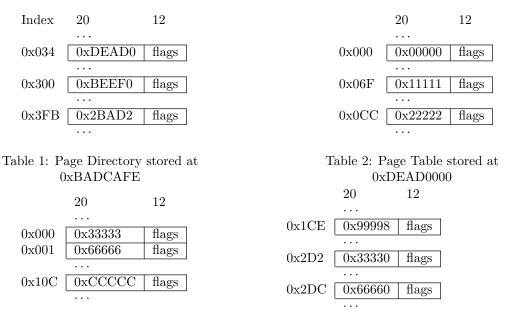
Handout: 28.02.2023

Introduction

In exercise four, we focus on connecting virtual memory with its physical counterpart through page tables, their "caches" the translation lookaside buffer (TLB), and the MMU.

1 Paging

For all the exercises below, assume the page tables and page directory described in Tables 1-4



0xBEEF0000

Table 3: Page Table stored at

- Table 4: Page Table stored at 0x2BAD2000
- 1. Can you motivate why we use nested structures instead of storing the mapping in a single table? What are the advantages and disadvantages of a multi-level page table solution? How can we address the issues that arise from the multilevel approach?
- 2. In the Page Directory we only use 20 bits for storing the address of the respective Page Tables, the rest is used to store flags. How can we still resolve, where the page tables are stored?
- 3. Assuming that both, Page Directory and Page Tables contain 1K entries and that the physical page is of size 4KB. Which physical addresses do the following virtual addresses map to?

- 4. Assuming that both, Page Directory and Page Tables contain 1K entries and that the physical page is of size 4KB. Which virtual address does the physical address 0x99998765 translate to?
- 5. Given the following split up of the virtual address, how many Page Directories and Page Tables are there? How big are the respective tables? How big are the physical pages? What is the max amount of memory we can handle like that? Would another split change the amount of memory that we can manage? Assume we have a 48-bit address, that is divided up as follows:
 - Part 1 (most significant bits): 14 bits

- Part 2: 13 bits
- Part 3: 12 bits
- Part 4 (least significant bits): 9 bits
- 6. How does the use of page table enable us to share memory between multiple processes? Do the processes know that a page is shared between multiple processes? Do they have to be at the same virtual address?

2 Swapping/Page Replacement

1. Let's assume we are working with a small machine that has 6 physical frames. Given the access pattern below, how would LRU and the Belady's MIN replacement strategy perform? How many replacements are necessary and when do they occur? How many page faults will occur?

```
Access pattern: 1, 3, 6, 8, 4, 2, 4, 7, 5, 5, 6, 8, 9, 4, 1, 3, 8, 8, 2, 5, 1, 3, 4, 8, 1, 3, 6
```

2. Assuming that a memory access takes 20 ns and an access to the disk (we are assuming an SSD here) takes $5*10^5$ ns, how much faster is Belady than LRU? What if we don't count the warmup phase (the pagefaults that can't be avoided) and only look at the steady execution?