#uni/semester3/Betriebssysteme/chapter20/homework

1. With a linear page table, you need a single register to locate the page table, assuming that hardware does the lookup upon a TLB miss. How many registers do you need to locate a two-level page table? A three-level table?

   Operating Systems Homework 4

   ➡ by register I understand some base value to locate something. For two-level page table I would say you need only one register, namely the PDBR (page directory base register). For the three level that wouldn't change, because the next page directory is pointed to by the first one, and thus there is no need for an extra base register.

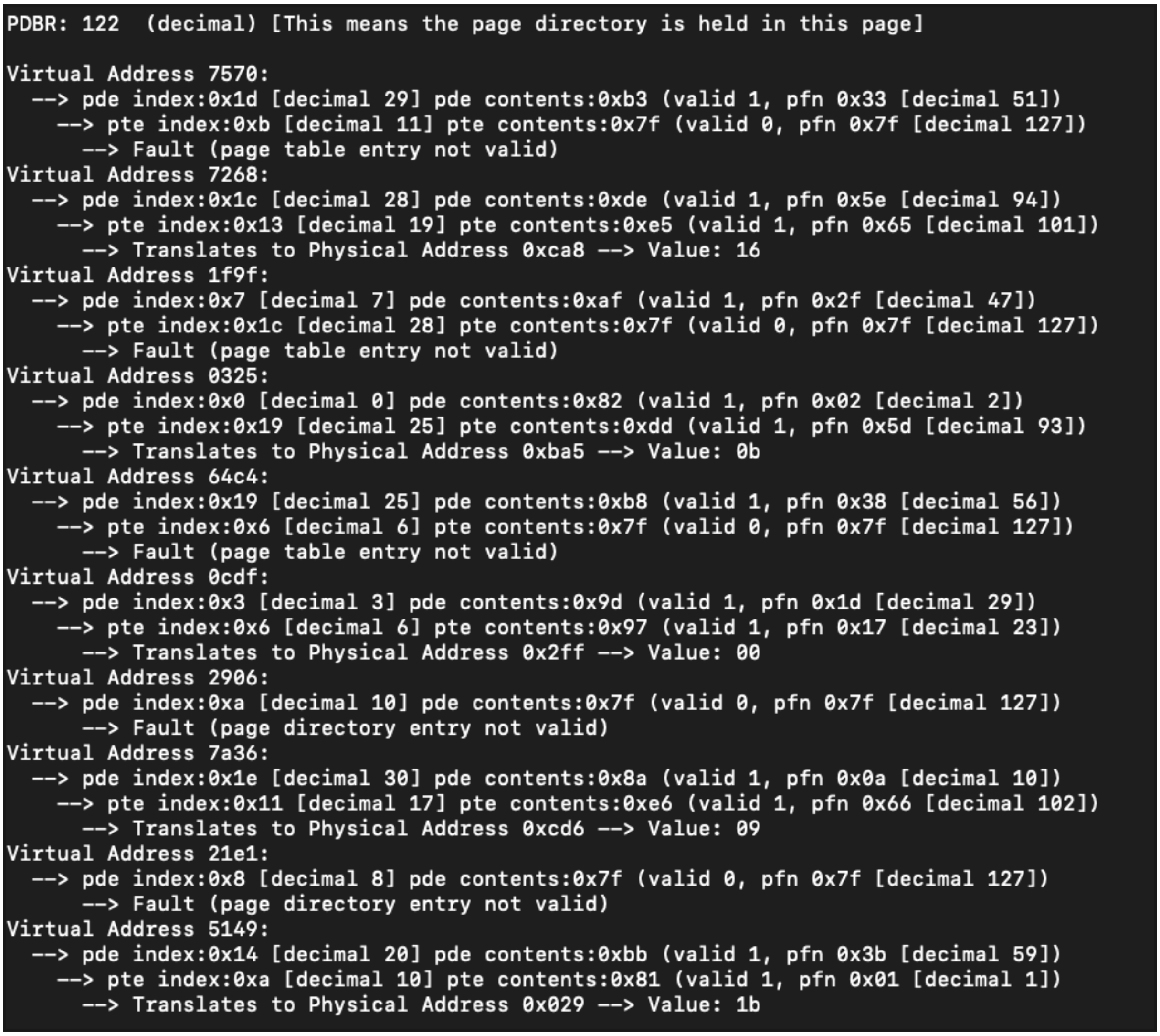
2. Use the simulator to perform translations given random seeds 0, 1, and 2, and check your answers using the -c flag. How many memory references are needed to perform each lookup?
   ➡ Assumptions:
   - ➡ page size is 32 byte
   - ➡ virtual address space is 1024 pages or 32KB
   - ➡ physical memory has 128 pages
   - ➡ virtual address needs 15 bits (5 offset and 10 for the VPN)
   - ➡ physical address needs 12 bits (5 offset and 7 for the PFN)
   - ➡ multi level page table, so the upper 5 bits of VA are used to index into a page directory. The rest of 5 bits are used to index the 32 entries which a page of the page table has.
   - ➡ Format of PDE is: VALID | PFN6 ... PFN0
   - ➡ Format of PTE is: VALID | PT6 ... PT0
   - ➡ You get PDBR (page where page directory is held) and how the pages look like

   How many memory references are needed to perform each lookup?
   - ➡ For a TLB hit, only one memory reference
   - ➡ If PDE entry valid and PTE entry valid
     - ➡ 1 memory reference to get PDE content
     - ➡ 1 memory reference to get PTE content
     - ➡ 1 memory reference to get final value

- → If PDE entry valid and PTE entry not valid
    - → 1 memory reference to get PDE content
    - → 1 memory reference to get PTE content
- → If PDE entry is not valid
    - → 1 memory reference to get PDE content

→ For the exercises see ordner.

Seed 0:



```
PDBR: 122  (decimal) [This means the page directory is held in this page]

Virtual Address 7570:
  --> pde index:0x1d [decimal 29] pde contents:0xb3 (valid 1, pfn 0x33 [decimal 51])
    --> pte index:0xb [decimal 11] pte contents:0x7f (valid 0, pfn 0x7f [decimal 127])
      --> Fault (page table entry not valid)
Virtual Address 7268:
  --> pde index:0x1c [decimal 28] pde contents:0xde (valid 1, pfn 0x5e [decimal 94])
    --> pte index:0x13 [decimal 19] pte contents:0xe5 (valid 1, pfn 0x65 [decimal 101])
      --> Translates to Physical Address 0xca8 --> Value: 16
Virtual Address 1f9f:
  --> pde index:0x7 [decimal 7] pde contents:0xaf (valid 1, pfn 0x2f [decimal 47])
    --> pte index:0x1c [decimal 28] pte contents:0x7f (valid 0, pfn 0x7f [decimal 127])
      --> Fault (page table entry not valid)
Virtual Address 0325:
  --> pde index:0x0 [decimal 0] pde contents:0x82 (valid 1, pfn 0x02 [decimal 2])
    --> pte index:0x19 [decimal 25] pte contents:0xdd (valid 1, pfn 0x5d [decimal 93])
      --> Translates to Physical Address 0xba5 --> Value: 0b
Virtual Address 64c4:
  --> pde index:0x19 [decimal 25] pde contents:0xb8 (valid 1, pfn 0x38 [decimal 56])
    --> pte index:0x6 [decimal 6] pte contents:0x7f (valid 0, pfn 0x7f [decimal 127])
      --> Fault (page table entry not valid)
Virtual Address 0cdf:
  --> pde index:0x3 [decimal 3] pde contents:0x9d (valid 1, pfn 0x1d [decimal 29])
    --> pte index:0x6 [decimal 6] pte contents:0x97 (valid 1, pfn 0x17 [decimal 23])
      --> Translates to Physical Address 0x2ff --> Value: 00
Virtual Address 2906:
  --> pde index:0xa [decimal 10] pde contents:0x7f (valid 0, pfn 0x7f [decimal 127])
      --> Fault (page directory entry not valid)
Virtual Address 7a36:
  --> pde index:0x1e [decimal 30] pde contents:0x8a (valid 1, pfn 0x0a [decimal 10])
    --> pte index:0x11 [decimal 17] pte contents:0xe6 (valid 1, pfn 0x66 [decimal 102])
      --> Translates to Physical Address 0xcd6 --> Value: 09
Virtual Address 21e1:
  --> pde index:0x8 [decimal 8] pde contents:0x7f (valid 0, pfn 0x7f [decimal 127])
      --> Fault (page directory entry not valid)
Virtual Address 5149:
  --> pde index:0x14 [decimal 20] pde contents:0xbb (valid 1, pfn 0x3b [decimal 59])
    --> pte index:0xa [decimal 10] pte contents:0x81 (valid 1, pfn 0x01 [decimal 1])
      --> Translates to Physical Address 0x029 --> Value: 1b
```

3. Given your understanding of how cache memory works, how do you think memory references to the page table will behave in the cache? Will they lead to lots of cache hits (and thus fast accesses?) Or lots of misses (and thus slow accesses)?

→ During the exercises the VAs were pretty random, and I was jumping from physical page to physical page. The same page in a row didn't really happen. So I guess because of this, it would lead to many TLB misses, because the physical page wasn't really accesses before.

➡ Because of temporal and spatial locality I would expect a very high hit rate, because else the TLB wouldn't be too very useful if too many TLB misses happen.