# Operating Systems – Review Questions 3

Group: Ketchup

Author: Ali Sahibi

## 1)

* Logical addresses are addresses generated by the CPU. Physical addresses are addresses as seen by the memory unit(RAM).
* Logical addresses are “virtual” meaning that the logical address does not reference a physical slot in the memory array. Physical addresses are “real” in the sense that a physical address is a direct reference to a physical slot on the memory array. The MMU maps logical addresses to physical addresses.

## 2)

The logical address space is the set of all logical addresses. The physical address space is the set of all physical addresses.

1. Logical address space is of size: 64 \* 1024 words = 216 words. Each word is an address to a byte in a virtual memory array. To be able to generate 216 unique addresses the logical address must have 16 bits.
2. Since page size = frame size 🡪 Physical address space is of size: 32 \* 1024 words = 215 words. The physical address must have 15 bits.

## 3)

1 KB = 1024 bytes.

A memory array consists of addressable blocks of bytes. Page size of 1024 bytes 🡪 1024 addressable blocks of bytes in a page.

Page number can be calculated through page\_nr = floor(address/1024).

Offset can be calculated through offset = address mod 1024.

|  |  |  |
| --- | --- | --- |
| Address | Page number | Offset |
| a | 3 | 13 |
| b | 41 | 111 |
| c | 210 | 161 |
| d | 634 | 784 |
| e | 195 | 320 |

## 4)

1. 256 pages of size 4096 bytes 🡪 4096 addressable blocks of bytes in a page. 4096 addressable blocks in a page 🡪 offset requires 12 bits. Logical address space has 256 pages 🡪 page number requires 8 bits. The logical address requires 20 bits.
2. 64 frames of size 4096 bytes. You need 6 bits to address a specific frame and 12 bits required to address a specific byte in a frame. The physical address requires 18 bits.

## 5)

A page fault occurs when the extracted page number from a virtual address has the valid-invalid bit set to invalid in the page table. This means that the page is not in main memory and is instead on the disk.

Handling page faults:

First check an internal table for the process to check that the requested memory address is valid, i.e. check if the memory address is within the memory limits for the process. If the requested memory address is invalid the process is terminated.

If the requested memory address is valid but the valid-invalid bit for the page number is set to invalid in the page table we bring in the page into main memory from the disk. We find a free frame and read the desired page into the free frame, if there is no free frame we perform a page replacement according to some page replacement policy. We modify the page table to indicate that the page is in memory, i.e. setting the frame number to the base address of the allocated frame and setting the valid-invalid bit to valid. We then restart the instruction that caused the page fault.