CS376 W16 Assignment

Directions: Edit this document to answer the following questions. If you need to draw a diagram or handwrite something, be sure to include it in this document also.

1. *(10 points)* Give two differences between logical and physical memory.

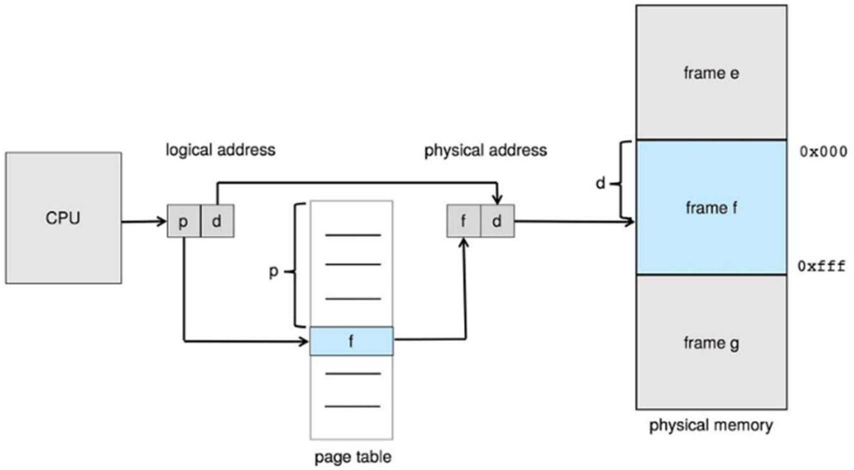
The Logical address is generated by the CPU whereas the physical address is the location in a memory unit such as a hard drive. The user can view the logical address of a program but the user can never view the physical address of a program.

2. *(10 points)* Why are page/frame sizes always a power of 2?

The power of 2 as a page size makes translation of a logical address into a page number and page offset easy. For instance, if the logical address is 2^m and the page size is 2^n bytes, then m-n designates the page number and n designates the page offset.

3. *(10 points)* Explain why there is a need for a logical and physical address? How does a logical address get translated to a physical address? Use the diagram below for reference:

The reason there needs to be logical and physical addresses is because the operating system needs to be able to manage memory securely. By having logical addresses the operating system is creating a mapping between the physical addresses allocated to a process and making sure that another process cannot access the memory of another process. Use the page number in the page table to find the frame of the physical address and then the page offset is the same for both the logical address and the physical address so you combined the frame and the offset to get the physical address.



4. *(15 points)* For this question, refer to section 9.3 specifically p 361 to help you with your answer: In a system with a logical address space of 64 pages of 1024 words each, mapped into a physical memory of 32 frames how many bits are there in the logical address? How many in the physical address? Explain your answers.

There are 18 bits in the logical address since 1024 = 2^10 and 64=2^6 so the logical address is 10 + 6 = 16 bits. For the physical address since there are 32 – 2^5 frames, the physical addresses are 5 + 10 = 15 bits.

5. *(10 points)* Read section 9.5 and explain the difference between swapping and paging. Why don't mobile devices use swapping?

Swapping is when a process is moved between main memory and a backing store. Paging involves breaking physical memory into fixed-sized blocks such as frames and pages. Mobile devices don't use swapping because they have flash memory which is limited in space and this space constraint makes paging more desirable. Also, flash memory has a limited number of writes before it becomes less reliable so swapping would use up a lot of these writes.

6. *(10 points)* Read section 10.1 and explain how virtual memory abstracts the actual implementation of memory to the programmer.

Virtual memory allows the programmer to no longer need to worry about the physical memory available and they can concentrate instead on programming.

For the following questions, Read section 10.2

7. *(10 points)* Why isn't it wise to load an entire program into memory at program execution time? What's the advantage of doing so?

It is not wise to load the entire program into memory at program execution time because we may not initially need the entire program in memory. For instance, if a user is presented with options and we load the whole program then we have loaded all the code for all options when we really only need the option the user selected. Paging can cause slowdown so while loading the entire program at once may not be wise it can sometimes be faster than paging.

8. *(13 points)* Explain the role of the valid-invalid bit with demand paging and its role in how demand paging is implemented.

The valid-invalid bit serves as a flag for memory in demand paging. When the bit is set to valid, the associated page is both legal and in memory. When the bit is set to invalid, the page is either not valid (i.e. not in the logical address space of the process) or it is valid but is currently on disk.

9. *(12 points)* Read 10.6 and give your interpretation of what causes thrashing and how it can be avoided.

Thrashing is when a process is spending more time paging than executing. Thrashing occurs when a process is under allocated the minimum number of pages it requires which causing it to page fault constantly. The system can detect thrashing by evaluating the level of CPU utilization and comparing it with the level of multiprogramming and reducing the level of multiprogramming when thrashing is detected.