Jeremy Holloway
CPSC-3220-001
6/5/2019

Pre-class Assignment #13

1. Define the following terms:
    - virtual address: An address that must be translated to produce an address in physical memory.
    - physical address: An address in physical memory.
    - core map: A data structure used by the memory management system to keep track of the state of physical page frames, such as which processes reference the page frame.
    - page frame: An aligned, fixed-size chunk of physical memory that can hold a virtual page.

2. What are two problems that the textbook identifies for a simple base-bounds approach to address translation and memory protection?

    It isn't possible to prevent a program from overwriting its own code, and it also has difficulty sharing regions of memory between two processes.

3. Explain how two processes can share the same code segment in a system with segmented virtual memory.

    The hardware can support an array of pairs of base and bounds registers, for each process. Setting up the segment table to point to the same region of physical memory.

4. Do different threads have different segment tables? Explain.

    If they were created by different processes then they would have a different segment table, otherwise they would have the same table as their process.

5. Identify two reasons why fixed-size pages are typically preferred over variable-length segments.

    Fixed sized chunks are easier to allocate, result in more efficient disk transfers, and no external fragmentation.

6. In a single-level paged virtual memory system, such as the one depicted in Figure 8.6, is the size of the page table proportional to the size of the virtual memory or to the size of the physical memory?

    The size of the page table is proportional to the size of the virtual address space.

7. In a demand paging system, how can program execution start (and continue) if all the pages of the process are not yet loaded into the physical memory?

As the program starts up , if it happens to jump to a location that has not not been loaded yet, the hardware will cause an exception (page fault), and the kernel can stall the program until that page is available.

8. Is it possible to combine variable-length segments and fixed-size pages in the same system? Explain. In a system such as shown in Figure 8.7, what is the granularity of segment size (i.e., byte, word, page)?

Yes, figure 8.7 in the book shows an example of such a program but either option is feasible. To keep the allocation simple the maximum segment size is chosen to allow the page table to be some small multiple of the page size, thus it's granularity is measured in page size.

9. How does a multilevel translation scheme handle a sparse virtual address space?

To simplify the address translations the lower levels of the pages table are only filled in if that portion of the page table is in use, so if that space is not in use then the lowers levels would not be filled in.

10. Assume that you can allocate large blocks of contiguous physical memory. Identify and explain the conditions under which a multilevel translation scheme can omit building lower-level page tables for these large blocks.

If you have a large continuous block, you can map it with a single entry in the page table, or the contiguous block does not have the same access(read/write/etc...).