Segmentation

(1) How are relocation and protection implemented in Pentium architecture? Consider the roles of both segmentation and paging.

(2) How is segmentation different from paging? Why was each technique invented?

(3) Using either Multics or Pentium architecture as an example, explain how segmentation is used in enforcing protection?

(4) How is a virtual address converted to a physical address, considering both segmentation and paging in (a) Multics and (b) Pentium architectures?

(5) In paged-segmentation implementations: Multics has one page table per segment, whereas Pentium has one page table for multiple segments. Why the difference? Which one is better? Why?

(6) OR

(7) How many page tables are there per segment in Multics and Pentium. (b) Which one (Multics/Pentium) is better? Why?

(8) What problem does segmentation solve that paging doesn't solve? What problem does paging solve that segmentation doesn't solve?

(9) Which of the following memory designs can cause internal fragmentation only, external fragmentation only, both, or neither? Briefly explain why.

A. Pure paging
B. Pure segmentation
C. Paging with Segmentation
D. Using superpages of the same size (no base pages or any other superpage size)
E. Using a mix of superpages of different sizes

(10) Considering memory protection, explain how the operating system ensures that user-level processes don't access kernel-level memory?

(11) Consider segment-level protection and CPU privilege levels in x86. Explain how the MMU, together with OS, determines whether a code executing on the CPU, has the permissions to access a given memory address?