CPS 110 Second Exam

Spring 2000

Answer all three questions. Please sign your name and staple your answers together. You have 75 minutes. Have a great summer.

Synchronization revisited. In class we discussed synchronization using semaphores and mutex/condition variable pairs. Are these facilities equivalent in power, i.e., is there any synchronization problem that can be solved with either facility that cannot be solved with the other? Prove your answer. (60 points)

Mode and Context. The following questions concern the handling of processes in a protected kernel-based operating system such as Unix or Nachos. (10 points each; total 40)

(a) Explain how the kernel initializes physical memory frames that it has allocated to back the virtual memory of a process.

(b) Explain how the kernel initializes the program counter register (PC) and stack pointer register (SP) before switching the CPU to user mode to start a fresh process for the first time. What about the other registers?

(c) Once the CPU is executing in user mode in the context of the fresh process, what could cause it to switch back into kernel mode? Give three distinct examples and outline how each affects the PC and SP registers on re-entry to the kernel. What about the other registers?

(d) Once the CPU is in kernel mode as a result of the examples in part (c), what could cause it to switch back into user mode? List as many distinct scenarios as you can think of, and outline for each case how the kernel determines values for the PC and SP registers before the switch to user mode. What about the other registers?

Nachos. Discuss the approach taken for synchronization in your Nachos kernel, answering the following questions. If you cannot recall the details of your Nachos work, then answer with reference to the kernel interfaces and implementation techniques for common kernel facilities for virtual memory, process management, file access, and inter-process communication. Your answer should be clear, complete, specific, and well illustrated with examples. (100 points total)

(i) What were the primary data structures or modules that required mutual exclusion? How would your kernel have failed to meet its design requirements if you did not enforce mutual exclusion for each of these structures?

(ii) What are the events that cause a process to block within your kernel? What events cause one process to wake another?

(iii) Outline ways in which your Nachos kernel might be different from a “real” kernel with respect to these questions.