CSCI 4320 (Principles of Operating Systems), Fall 2002
Homework 5

Assigned: November 22, 2002.

Due: December 2, 2002, at noon. *Not accepted late.*

Credit: 30 points.

1 Reading

Be sure you have read (or at least skimmed) chapters 5 and 6.

2 Problems

Answer the following questions. You may write out your answers by hand or using a word processor or other program, but please submit hard copy, either in class or in my mailbox in the department office.

1. (5 points) Consider a computer system with the following characteristics: Reading or writing a memory word takes up to 10 nsec. It has 16 CPU registers, and when an interrupt occurs, all of them, plus the program counter and the PSW are pushed onto the stack (in memory). What is the maximum number of interrupts per second this machine can process? (*Hint:* Observe that after an interrupt is processed, the contents of CPU registers, program counter, and PSW must be restored to their pre-interrupt values by popping them back off the stack.)

2. (5 points) Consider a printer that prints at a maximum rate of 400 characters per second, connected to a computer system in which writing to the printer's output register takes essentially no time. If each character printed requires an interrupt that takes a total of 50 microseconds to process, would it make sense to use interrupt-driven I/O to write to this printer, or would it be better to use programmed I/O? Why?

   Now consider a system with a memory-mapped terminal, and suppose that interrupts take a minimum of 100 nsec to process and copying a byte into the terminal's video RAM takes 10 nsec. Would it make sense to use interrupt-driver I/O to write to the terminal, or would it be better to use programmed I/O? Why?

3. (5 points) The textbook divides the many routines that make up an operating system's I/O software into four layers, as shown in Figure 5-10. In which of these layers should each of the following be done?

   (a) Writing commands to a printer controller's device registers.

   (b) Detecting that an application program is attempting to write data from an invalid buffer address.

   (c) Converting floating-point numbers to ASCII for printing.

   (d) Computing the track, sector, and head for a disk read operation.

4. (5 points) Consider a simple operating system that provides only a single-level directory, but allows the directory to contain as many files as desired, with file names as long as desired. Would it be possible to use this system to simulate something resembling a hierarchical file system? How?

5. (5 points) Consider a digital camera that records photographs in some non-volatile storage medium (e.g., flash memory). Photographs are recorded in sequence until the medium is full; at that point, the photographs are transferred to a hard disk and the camera's storage is cleared. If you were implementing a file system for the camera's storage, what strategy would you use for file allocation (contiguous, linked-list, etc.) and why?

6. (5 points) The textbook describes two strategies for keeping track of free blocks in a file system, one using a list of free blocks and one using a bitmap. What would happen if this free list or bitmap was completely lost because of a system crash — is there a way to recover, or must you hope you have a backup of any critical data? Answer separately for file allocation using i-nodes and file allocation using a FAT.