# CSCI 4320 (Principles of Operating Systems), Fall 2004

# Homework 4

**Assigned:** November 18, 2004.

**Due:** November 30, 2004, at 5pm. *Not accepted late.*

**Credit:** 50 points.

## 1 Reading

Be sure you have read 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 (i.e., $10 \times 10^{-9}$ seconds). It has 32 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 (i.e., $50 \times 10^{-6}$ seconds) 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? (*Hint:* How much time is required for interrupt processing if the printer is printing at its maximum rate?)

   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) Converting floating-point numbers to ASCII for printing.

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

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

(d) Detecting that an application program is attempting to write data from an invalid buffer address. (Assume that detecting an invalid buffer address can only be done in supervisor mode.)

4. (5 points) Consider a system that uses its local area network as follows. An application program makes a system call to write data packets (each 1024 bytes, ignoring headers) to the network. The operating system first copies the data to be sent to a kernel buffer. Working on one packet at a time, it then copies the data to the network controller. When all 1024 bytes have been copied to the network controller, it sends them over the network at a rate of 10 megabits ($10 \times 10^6$ bits) per second. The receiving controller receives each bit a microsecond after it is sent. When the last bit in the packet is received, the destination CPU is interrupted, and its operating system copies the packet into a kernel buffer, inspects it, and copies it into a buffer owned by the application program that should receive it. It then sends back an acknowledgment (assume one bit) to the sending computer, which interrupts the sending CPU, and work can begin on the next packet. How long does it take to send each packet, if it takes one millisecond to process an interrupt (on either CPU) and one microsecond to copy a byte? Assume that the time taken for the receiving CPU to inspect the packet is negligible. What is the effective transfer rate (in bits per second) over this connection?

   (*Hints:* Notice that some times are per bit and some are per byte. If you think you need to make additional assumptions, do so and explain them. If you show your calculations and briefly explain what you are doing, your odds of getting partial credit are better.)

5. (5 points) Suppose at a given point in time a disk driver has in its queue requests to read cylinders 10, 22, 20, 2, 40, 6, and 38, received in that order. If a seek takes 5 milliseconds (i.e., $5 \times 10^{-3}$ seconds) per cylinder moved, and the arm is initially at cylinder 20, how much seek time is needed to process these requests using each of the three scheduling algorithms discussed (FCFS, SSF, and elevator)? Assume no other requests arrive while these are being processed.

6. (5 points) Student H. Hacker installs a new disk driver that its author claims improves performance by using the elevator algorithm and also processing requests for multiple sectors within a cylinder in sector order. Hacker, very impressed with this claim, writes a program to test the new driver's performance by reading 10,000 blocks spread randomly across the disk. The observed performance, however, is no better than what would be expected if the driver used a first-come first-served algorithm. Why? What would be a better test of whether the new driver is faster? (*Hint:* The test program reads the blocks one at a time. Think about how many requests will be on the disk driver's queue at any one time.)

7. (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?

8. (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?

9. (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 UNIX V7 and MS-DOS FAT-16 filesystems. (*Hint:* Read the last paragraph of section 6.4.3 carefully.)

10. (5 points) Consider a UNIX filesystem (as described in section 6.4.5) in which each i-node contains 10 direct entries, one single-direct entry, one double-indirect entry, and one triple-indirect entry. If a block is 1KB (1024 bytes) and a disk addresses is 4 bytes, what is the maximum file size, in KB? (*Hint:* Use the blocksize and size of disk addresses to determine how many entries each indirect block contain.)

## 3 Optional Programming Problems

Optionally, do the following programming problems. You will end up with at least one code file per problem. Turn in your code by sending mail to `csci4320-homework@cs.trinity.edu`, with each of your code files as an attachment. If there's any question of which file(s) correspond to which problems, explain in the body of the mail message. Please use a subject line such as "homework 4" or "hw4". You can develop your programs on any system that provides the needed functionality, but I will test them on one of the department's Fedora Core 2 Linux machines, so you should probably make sure they work in that environment before turning them in.

1. (Up to 5 extra-credit points) Write a program that given a directory $D$, blocksize $B$, and maximum number of blocks $M$ as command-line arguments prints out how many files in $D$ and its subdirectories are of size $B$ or less, how many are of size between $B$ and $2B$, etc., up to size $MB$. Include directories and symbolic links (but count the size of the link and not the file/directory it links to). Here is a sample execution.

```
[bmassing@Xena02]$ ./filesizes /var/www 512 20
Unable to open /var/www/HTML-Documents/apache-documentation/manual/search:  Permission denied
Unable to open /var/www/HTML-Documents/About/The_Courses/cs3394.hci/dcernose/javapres/turnin/COM:  Permission denied
Unable to open /var/www/HTML-Documents/About/The_Courses/cs3291.java/dcernose/javapres/turnin/COM:  Permission denied
Unable to open /var/www/accesswatch-1.33:  Permission denied
Results for directory /var/www:
               2339 files of size          1 blocks
                478 files of size          2 blocks
                967 files of size          3 blocks
               1192 files of size          4 blocks
               7202 files of size          5 blocks
               9816 files of size          6 blocks
               5804 files of size          7 blocks
               3621 files of size          8 blocks
               1769 files of size          9 blocks
               1013 files of size         10 blocks
               1606 files of size         11 blocks
                706 files of size         12 blocks
                430 files of size         13 blocks
                645 files of size         14 blocks
                514 files of size         15 blocks
                525 files of size         16 blocks
                444 files of size         17 blocks
                303 files of size         18 blocks
                303 files of size         19 blocks
                264 files of size         20 blocks
               6296 files of size         21 blocks or more
```

(Of course, you won't be able to examine files in directories you don't have access to. That's okay; just print error messages as above.)

*Hints:* Read the man pages for `opendir`, `readdir`, and `lstat`. You might also be interested in the man pages for `chdir` and `strerror`.