CSE 451
Optional Homework
Due: December 6, 2012, 12:30pm

Problem 1: (10 points)

Consider a demand paging system, where a dedicated disk is used for paging, and file system activity uses other disks. Measured utilizations (in terms of **time**, not space) are:

CPU utilization	20%
Paging disk	10%
Other I/O devices	95%

For each of the following changes, say what its likely impact will be on **CPU** utilization: will it will probably significantly increase, marginally increase, significantly decrease, marginally decrease, or have no effect on the CPU utilization, and why.

(a) Get a faster CPU

(b) Get a bigger paging disk

(c) Increase the degree of multiprogramming

(d) Decrease the degree of multiprogramming

(e) Get faster other I/O devices

Problem 2: (6 points)

Suppose an architecture with paged segmentation has a 32-bit virtual address that is divided into fields as follows:

4 bit segment number	12 bit page number	16 bit offset

The segment and page tables are as follows (all values are in hexadecimal):

Segment Table

0	Page Table A
1	Page Table B
x	(rest invalid)

Page Table A

0	CAFE
1	FEED
2	BEEF
3	BAAA
x	(rest invalid)

Page Table B

0	F000
1	B0B0
2	CACA
x	(rest invalid)

Find the physical address corresponding to each of the following (hexadecimal) virtual addresses (answer "invalid virtual address" if the virtual address is invalid):

a) 00010002

b) 10032002

c) 10020005

Problem 3: (34 points = 9 + 25)

You've just been hired by Mother Nature to help her out with the chemical reaction to form water, which she doesn't seem to be able to get right due to synchronization problems. The trick is to get two H atoms and one O atom all together at the same time. The atoms are threads. Each H atom invokes a procedure *hReady* when it's ready to react, and each O atom invokes a procedure *oReady* when it's ready. For this problem, you are to write the code for *hReady* and *oReady*. The procedures must delay until there are at least two H atoms and one O atom present, and then one of the procedures must call the procedure *makeWater* (which just prints out a debug message that water was made). After the *makeWater* call, two instances of *hReady* and one instance of *oReady* should return.

a) Give an example program using oReady() and hReady() (and presumably some other synchronization) that can deadlock.

b) Implement oReady() and hReady() using locks and Hansen/Mesa condition variables. Your solution must avoid starvation and busy-waiting, and it must work for arbitrary numbers of H and O atoms.