Name: ___________________________

Operating Systems
V22.0202 Spring 2010

Midterm Exam

1. **True/False**. Circle the appropriate choice.

   (a) **T F** A process is a program in some state of execution.

   (b) **T F** A physical address indicates a block on the disk where data can be found.

   (c) **T F** The memory bus is a set of wires running between the CPU and main memory.

   (d) **T F** A controller is the portion of the OS responsible for communicating with an I/O device.

   (e) **T F** The Not Frequently Used (NFU) page replacement algorithm keeps a precise count of the number of times that a page has been referenced.

   (f) **T F** In a system where lottery process scheduling is used, efficient use of the CPU can be increased by allocating more "lottery tickets" to I/O-bound processes than to compute-bound processes.

   (g) **T F** If all the processes on a computer are blocked, the system must be rebooted because there is no way for any of the blocked processes to become ready.

   (h) **T F** Upon a TLB miss, a system that uses a 4-level page table will incur more memory references than a system that uses a single-level page table.

   (i) **T F** In round-robin scheduling, a longer quantum means that an interrupt from the disk controller will take longer to be handled by the OS.

   (j) **T F** The threads within a process have their own registers, but use the same code, global data, and stack.

2. **Fill in the blanks provided on this sheet.**

   (a) Suppose that on a machine with 16-bit addresses and 512 byte pages, the TLB contains the following:

1	0	1	0C	38
1	1	1	45	1D
1	0	0	19	22
0	1	1	37	66

   where the fields are, from left to right, the valid bit, the R bit, the M bit, the virtual page and the page frame. The numbers are given in hex. What is the physical address corresponding to the virtual address 194A? If that information cannot be determined from the TLB, indicate why. (NOTE: Consider the individual bits carefully)

   Answer: ____________________

   Please turn page over

(b) Suppose the TLB uses the NRU algorithm to evict an entry, when a page has been referenced for which there is no TLB entry. What is the first entry in the above TLB (from part (a)) that will be evicted? What is the second TLB entry to be evicted?

Answer: First:____________ Second:______________

3. Put your answer in the blue book.

This question is about how *you* would implement semaphores in *your* first programming project (the interrupt handler & scheduler assigment).

Suppose the first programming assignment had specified that you had to implement binary semaphores as follows:

- The OS supports 50 binary semaphores, which are numbered 0 through 49 and are referenced by number, not by name.
- There is a trap called `SEM_DOWN` that, when invoked by a running process, causes the number of the semaphore being used to be put in the R2 register.
- There is a trap called `SEM_UP` that, when invoked by a running process, causes the number of the semaphore being used to be put in the R2 register.

(a) How would you represent the 50 semaphores in your kernel.c code?

(b) Describe precisely, in C code or pseudo-code, how your trap handler procedure would handle the SEM_DOWN trap.

(c) Describe precisely, in C code or pseudo-code, how your trap handler procedure would handle the SEM_UP trap.

4. Put your answer in the blue book.

On a machine with 32-bit addresses, suppose an OS uses two-level page tables as follows:

- A first level page table has 2048 entries.
- Each second level page table has 2048 entries.

(a) What is the size of a page?

(b) Suppose a process has a text segment that occupies 64MB, a stack segment that occupies 128MB, and a data segment that occupies 32MB. How much space is occupied by the page tables for that process? Your answer can be given in powers of 2, if you want, to avoid arithmetic.

(c) What does each entry in the first-level page table contain?

(d) In a two-level page table mechanism, what does each TLB entry contain? How is it different than a TLB entry on a system with a single-level page table mechanism?

(e) When a process references an address, precisely how does the MMU determine if the page containing the referenced address is in RAM and, if so, in which page frame. Be precise regarding 1) how the bits of the address are used and 2) the steps that the MMU goes through.

(f) What happens if it turns out that the page containing the referenced address is not in memory?