NAME: ____ (FIRST NAME FIRST) **SCORE**: _____

COSC 6360 FIRST MIDTERM OCTOBER 1, 2008

This exam is **closed book**. You can have **two pages** of notes. UH expels cheaters.

1. Consider a virtual memory system with 64-bit addresses and a clustered page table with a clustering factor of 2. What would be the length of a page table entry assuming that we are implementing:

a) *Complete subblocking* (5 points)? ___4×8=32__ bytes

b) *Partial subblocking* (5 points)? ___3×8=24__ bytes

2. The ***Midway*** distributed shared memory system uses a request/release scheme similar to that of Munin but requires users to specify which shared variables are associated with each request() call. How does this requirement affect (a) the performance of Midway compared to that of Munin and (b) the portability of user programs? (2×5 points)

a) Midway should perform better than Munin since a processor initiating a request will only get the updated values of the shared memory variables associated with that request.

b) Porting an existing program to Midway requires identifying which shared variables should be associated with each request, which is a fairly complex task for any non-trivial program.

3. What is the ***hidden cost*** of the BSD implementation of the Clock page replacement policy? (5 points) What did Babaoğlu and Joy do to limit this cost? (5 points)

The hidden cost of the BSD implementation of the Clock policy is the two context switches occasioned by the first reference to any page that had been marked invalid by the Clock hand. To limit this context switch overhead, Babaoğlu and Joy limited the *speed* at which the hand of the clock sweeps the circular list of active pages was limited to 300 pages/sec, which guaranteed that the software simulation of page referenced bits will never take more than 10% of the total CPU time)

4. Mach copy-on write policy is said to be a ***lazy policy***. Why? (5 points) What would be the corresponding ***eager policy***? (5 points) Has this eager policy been implemented? (5 points) Under which circumstances is it better than copy-on-write? (5 points)

Mach copy-on-write policy is said to be a lazy policy because it delays the creation of separate parent and child copies of a page until the page is modified by either the parent or the child. The corresponding eager policy would be to create separate address spaces for the parent and child process at fork(0 time without waiting for pages to be modified. This eager policy was used by the traditional implementation of the UNIX fork(). It works better than copy-on-write whenever parent and child modify a large number of their pages.

5. What is the main advantage of ***mapped files***? (5 points) Which ***memory object*** is associated with each mapped file? (5 points) What should be the ***inheritance*** and ***protection*** attributes of a mapped file? (2×5 points)

Mapped files reduce the number of system calls required to access a file as read(), write() and lseek() operations can be implemented as user-level procedures. The sole operations that require a kernel intervention are the actual transfers of data blocks between the main memory and the disk. The memory object associated with each mapped file is the file itself as it is stored on disk by the file system. The inheritance attribute of a mapped file will always be SHARED. It protection attributes are those specified by the process when it successfully opened the file. In UNIX these are READ-ONLY, WRITE-ONLY and READ-WRITE.

6. When does the ***ARC cache replacement policy***

a) ***Increase*** target_T1? (5 points)
Whenever a process accesses a page that is not in main memory but is in B1.________

b) ***Decrease*** target_T1? (5 points)
Whenever a process accesses a page that is not in main memory but is in B2.________

7. Consider an ***hypothetical*** file system organization for a ***64-bit version*** of UNIX. The i-node would still have 15 block addresses but all block addresses would be ***8-byte wide***. Assuming a block size of 4KB,

a) How many bytes could be accessed directly ***from the i-node***? (4 very easy points)

The contents of the 12 direct blocks: 12×4KB = 48KB ____________________________

b) How many bytes could be accessed directly ***using one level of indirection***? (4 points)

The contents of the 4K/8 blocks that can be accessed with one level of indirection __
4K/8×4KB = 512×4KB = 2MB ____

c) How many bytes could be accessed directly ***using two levels of indirection***? (4 points)

The contents of the $(4K/8)^2$ blocks that can be accessed with two levels of indirection
$(4K/8)^2 \times 4KB = 2^{18} \times 4KB == 2^{18} \times 2^{12}B = 2^{30}B$= one GB ________________________________

d) How many bytes could be accessed directly ***using three levels of indirection***? (4 points

The contents of the $(4K/8)^3$ blocks that can be accessed with two levels of indirection
$(4K/8)^3 \times 4KB = 2^{27} \times 4KB == 2^{27} \times 2^{12}B = 2^{39}B$= 512 GB________________________________

e) What would suggest to ***improve the performance*** of that organization? (4 points)

Increase the block size to 8KB (or more) to increase the number of bytes that can accessed for any given level of indirection. ______________________________________