CMSC 23000	**Operating Systems**	**Homework 3**
Winter 2006		**Due February 13**

1. Consider a system with a single-level page table that is kept in physical memory.

   (a) If a physical-memory access takes 20 nanoseconds, how long does it take to access a virtual-memory location?

   (b) If we add a TLB to the system with an access-time of 2 nanoseconds and we assume that 90% of memory references hit the TLB, what is the average memory reference time?

   (c) Suppose that we service TLB misses in software, instead of hardware, and that a TLB miss takes 100 nanoseconds to handle. What is the average memory reference time?

2. Give a user-space implementation of large arrays with the following operations:

   ```
   typedef ... *Array_t;
   Array_t *NewArray (int nPages);
   Array_t *Copy (Array_t *arr);
   void Update (Array_t *arr, int i, int v);
   int Subscript (Array_t *arr, int i);
   ```

   The size of an array is given as a multiple of the system's page size. The `Update` and `Subscript` operations should not involve any conditional branches (you do not have to check their bounds), and your implementation should use *copy-on-write* to implement the `Copy` operation.

   Hint: On a Linux or MacOS X system, read the man pages for the `mprotect` system call.

3. Assume you have a system with N processors running N threads in parallel. *Barrier synchronization* is a primitive that blocks a thread until all N threads have reached the barrier. Using mutex locks and condition variables, give an implementation for the barrier synchronization primitive

   ```
   void barrier_wait();
   ```

   You may assume that the number of threads involved in the barrier (N) is fixed.