CMSC 412 Midterm #2 (Spring 2010) - Solution

1.) (20 points) Define and explain the following terms:

a) Inverted Page Table

Maps physical addresses to virtual. Allow for smaller page pages with the amount of physical memory is much smaller than the virtual address range.

b) Sticky-bit

A bit in UNIX file systems that is used to indicate if a file should be kept it memory after the program terminates. When applied to directories, it restricts directory updates to files owned by the user.

c) ACL

Access control list. A list of groups/users and their permissions that can be associated with individual files and directories in a file system.

d) Working Set

A set of pages actively used by a process. The pages of a working set must be kept in memory for a process to run well (i.e. to prevent thrashing).

2) (20 points) Memory Systems

e) (7 points) The x86 processor uses a two-level page table (page directory and table) when operating in 32-bit mode. Why does AMD-64 (x86-64) in 64-bit mode use a 4-level page table when using 4KB pages?

To kept the page tables smaller. With only two levels, page tables would be 2^{26} = 8MB each

f) (5 points) In 64-bit mode on an AMD-64 (with 4KB pages) there are 512 entries per level in the page table, but in 32-bit mode there are 1024. Why?

Because in 64-bit mode the entries need to be 64-bites (8byes) long rather than 32-bits (4 bytes) so there is only room for half the number of entries with the page size fixed at 4KB

g) (8 points) Explain what TLB reach is and how do super pages improve the TLB reach?

How many bytes of memory can be accessed without a TLB miss (fault). Super pages are bigger so we can reach more bytes per TLB entry, so the total reach of the table gets bigger (without needing a larger table)

3.) (20 Points) Synchronization: You need to synchronize the process of getting and making coffee in the CS department lounge. In the department, there are two types of coffee urns (regular and decaf) and one coffee maker. If someone goes to get a cup of coffee of one type, and that pot is empty they make a new pot of that type of coffee. Provide a solution using semaphores (include variable declarations and initial semaphore values) to the coffee problem that ensures:

- Only one person at a time is taking coffee out of the decaf urn
- Only one person at a time is taking coffee out of the regular urn

- Only one urn is on the coffee maker at a time
- If one type of coffee is being made, people requesting coffee of the other type can get it if it is available.

You may assume there exists a function emptyUrn(bool checkDecaf) that returns true if the coffee urn indicated by the parameter is empty.

Variables

```
Semaphore decaf(1), regular(1), maker(1)
```

getCoffee(bool usesDecaf) {

```
If (usesDecaf) {
    P(decaf)
    If (emptyUrn(usesDecaf)) {
       P(maker);
       // make decaf coffee
       V(maker);
    }
     // take decaf coffee
    V(decaf);
} else {
    P(regular)
    If (emptyUrn(usesDecaf)) {
        P(maker);
        // make regular coffee
        V(maker);
    }
    // take regular coffee
    V(regular);
}
```

4.) (20 points) File Systems

a) (6 points) List three types of meta-data that are typically associated with a file

```
File size
Permissions
Medication/creation time
File name
File type (directory/regular file)
```

b) (14 points) Compare and contrast how windows and UNIX define which program should be invoked when a user requests to “execute” a file that contains a script (i.e. not a native binary).

```
Unix:
      Check for the execution bit being on
Examine the magic number. If the first two bytes of the file are #!,
invoked the program name that follow the magic number in the file.
Windows:
      Check the file name suffix (after the last dot)
Invoke the command associate with the suffix based on the information
stored in a per-user table that maps suffix to program to invoke.
```

4.) (20 points) Project

a) (6 points) For a fully functional program #4, the kernel and its data are non-pageable. However, you can still have a page fault (for a correct program) when running in kernel mode. How is this possible?

When the kernel access a user page (as part of a system call for example), that page might be paged out.

b) (7 points) Why did you need to copy the identity mapped physical region from the kernel page directory into each user process's page directory given that user-mode process can't touch kernel memory?

When user processes run in kernel mode (i.e. for system calls or traps), they run using the PDBR (cr3) of the user process, so every user process needs the kernel's identity mapped pages in it's address space.

c) (7 points) What changes would you need to make to your project to make page tables pageable?

Allocate the page table from the pageable memory

On page fault, would need to check for the page table being present when figuring out why an address faulted. This might result in several faults per instruction.

When page out the page, need to make the entry in the page table as paged out.