CSCI 3753 Spring 2014 **Vince Coghlan**

Homework #1

Problem 1. [10 Points] What is the role of instructions such as *trap ()* in operating system design?

When designing an operating system, you want to provide various operations to your processes, such as printing and reading to files, that interact with the various devices that the OS is controlling. When a trap instruction is executed, a number of things happen. Most importantly, the kernel takes over control, and decides whether or not to provide the requested servcie to the process. A trap is also used to catch errors. If something happens to a process to make an exception, a trap call is used so that the kernel can begin to handle it.

Problem 2. [10 Points] What role does the jump table play in executing a system call?

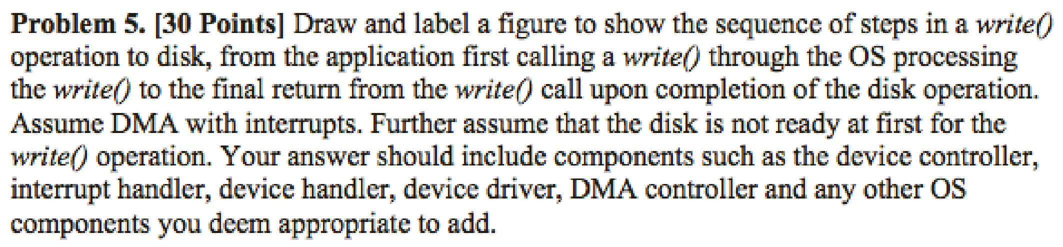
Since there are 300+ syscalls, it is most efficient and easiest to call on an array of function pointers with each syscall directly accesed using the ID number of the syscall. When the syscall is initiated, the kernel acceses this table to find the relavant call code.

Problem 3. [30 Points] Provide a step-by-step description for adding a new device in Linux operating system without requiring recompiling the kernel.

In order to add a new device, the kernel itself needs to be changed, without having to compile means that a loadable kernel module needs to be used. First, create you C code to access the device. Second, run insmod to add the module to the running kernel.

Problem 4. [20 Points] Explain two ways that I/O can be overlapped with CPU execution and how they are each an improvement over not overlapping I/O with the CPU.

One way is to have a seperate controller to access the I/O. This allows the second controller to read and buffer data from vaious devices while the CPU does other work. Then the CPU can read in all that data at once when the device manager decides to interrupt. This is faster since most CPU's are much much faster than most devices. If the CPU is waiting for the slow device, then it is not being fully utilized. The second method would be to use a polling system. Every now and then the CPU can stop execution of its main code to check all of the devices to see if they have done anything. This is good for the same reason, it allows the CPU to do work in between checking for I/O.



Problem 5. [30 Points] Draw and label a figure to show the sequence of steps in a *write()* operation to disk, from the application first calling a *write()* through the OS processing the *write()* to the final return from the *write()* call upon completion of the disk operation. Assume DMA with interrupts. Further assume that the disk is not ready at first for the *write()* operation. Your answer should include components such as the device controller, interrupt handler, device handler, device driver, DMA controller and any other OS components you deem appropriate to add.