CSCI 3753: Operating Systems
Fall 2016
Problem Set One

Please write your answers in the space provided.
Due date: Tuesday, September 20 in class. No extensions will be given except at the instructor's discretion in documented cases of extreme hardship or emergencies. Please submit a hardcopy of your solutions.

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

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

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

1. Write the device driver, compile it to create an LKM (.ko file)
    This LKM contains an initialization routine init_module( ) function that registers various device functions contained in the LKM with the kernel using appropriate kernel functions such as register_chrdev, register_blkdev, etc.
    This registration fills the entry table in the kernel (dev_func_i[j]) with appropriate function pointers

2. Call insmod command that gets a unique device number for the new device and invokes the init_module systems call to load the LKM. This system call invokes the LKM's initialization routine.

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.

Interrupt driven I/O: CPU initiates the I/O and sets up an interrupt handler. After that it performs other useful work in parallel with I/O data transfer being performed by I/O device. When I/O data transfer is complete, the CPU is interrupted to complete the remaining work for completing the I/O.

DMA based I/O: Similar to interrupt-driven I/O with the addition that DMA controller manages the data transfer between memory and device registers.

Problem 5. [30 Points] Draw and label a figure to show the sequence of steps (step 1, step 2, etc.) 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.