Frank, Sunday February 9, 2020

COMPSCI-3SH3 Operating Systems Assignment 1

1. **Ethics (10 Marks)** : **Consider the following situation: Suppose that you are a development engineer with responsibility for an embedded system employed in one of your company's major products. You seek to improve the efficiency of your software system and, following some research, you discover an algorithm posted on the Web that would provide a vast improvement for your system. The algorithm is written in the same language as that used by your system.**

   A) Would it ever be ethical to copy the code that implements the algorithm and incorporate it in your embedded system?

   This practice would be ethical, but must be treated with certain guidelines. With the advent of websites such as StackOverflow, where computer science experts help those who are struggling with certain implementations, we often see full solutions with code included to various problems that people have posted before. Often times, these experts have come up with clever, efficient solutions to one you may have at hand. By including a reference to the author of the code, I believe that copying someone elses implementation can save your company a lot of valuable time, and thus, I believe it would be ethical. Of course, it is also important to not blindly copy the code, and make sure you understand it and add documentation/testing if needed so that maintenance can be done in the future.

   B) Would it ever be good engineering practice to incorporate the code that implements the algorithm in your system?

   Again, as stated above, as long as you properly include citations to the original author of the code, I believe it would be ethical. In terms of good practice, we need to ensure a few more additional things. Firstly, as memory usage is crucial, especially in embedded systems where RAM is on the order of kilobytes instead of Gb like in our operating systems, one must carefully look over the code and understand the implementation fully in terms of memory usage. The same thing can be said for running time; sometimes, it is important to consider tradeoffs between the most optimal running time and memory usage. In addition, one should fully test the functionality of someone else's implementation, just as you would your own, instead of assuming the correctness/robustness of the code just because it came from an "expert" in the field. Again, since a huge part of a software's life-cycle and cost is associated with debugging and maintenance, this is crucial.

2. **Introduction**
   Q1 (4 marks): What is the purpose of interrupts? How does an interrupt differ from a trap? Can traps be generated intentionally by a user program? If so, for what purpose?

   Sometimes, it is needed to change the flow of control within a system or a current program running in a system. Thus, an interrupt is a signal gtenerated by the hardware in our operating systems that changes this flow of control. These interrupts are handled by interrupt handlers; if more than one interrupt is initiated, the handler prioritizes and allows them to be executed sequentially. After the execution of an interrupt is finished, the control is returned to the context/program in question right before the interrupt was executed.

Traps, on the other hand, are interrupts generated by software instead of hardware. Traps are generally used to invoke a system call. Compared to interrupts, there are a few key differences. Firstly, unlike interrupts, traps cannot disable hardware components. Also, Trap handlers operate Synchronously while interrupt handlers operate asynchronously. Finally, interrupts are required to preserve the state of CPU while traps do not have this requirement.

Finally, user programs can generate traps intentionally. One common usecase would be to throw exceptions to handle error conditions, such as as division by 0 or a file not found case.

Q2(6 marks): Direct memory access is used for high-speed I/O devices in order to avoid increasing the CPU's execution load.

a) How does the CPU interface with the device to coordinate the transfer?

First, to start the transfer, the CPU sets the Direct Memory Access (DMA) registers. This involves a pointer to the source of transfer, the pointer to the destination of the transfer, and the counter of the number of bytes to be transferred. Next, the DMA controller places the addresses on the bus to perform the transfers, while the CPU is available for other processes.

b) How does the CPU know when the memory operations are complete?

The DMA controller interrupts the CPU once the transfer is finished with all the bytes transferred.

c) The CPU is allowed to execute other programs while the DMA controller is transferring data. Does this process interfere with the execution of the user programs? If so, describe what forms of interference are caused.

Both the CPU and the DMA controllers are bus masters. A problem would be created if both the CPU and the DMA controller want to access memory at the same time. Thus, there should be mechanisms in place to prevent the scenario of the CPU getting access to main memory when the DMA controller seizes the memory bus. However, the CPU is still allowed to access data in its primary and secondary caches. Thus, an issue could arise if both the CPU and the DMA controller update the same memory locations in cache.

3. OS Structures(10 marks)

Q1(6 marks): How are iOS and Android similar? How are they different?

In terms of similarities, both Android and IOS are operating systems for mobile devices. In addition, both are based off of UNIX and are programmed in c/c++. In terms of functionality, both have calling, messaging, voice commands, navigation, etc. The UI are also very similar, with both integrating smooth touch screen operations such as swiping, tapping, zooming, drag-drop.

In terms of differences, android is an open-source OS, while iOS is a closed source OS. This means that android is much more customizable. In addition, this also means that Android is adapted and manufactured by many different companies such as Samsung, Oneplus, Nokia, while iOS is strictly managed and manufactured by Apple. Finally, iOS and Android differe in their kernel type; the iOS kernel is Hybrid, while the Android kernel is Monolithic.

Q2(4) marks:

a) **What is the purpose of system calls?**

System calls provide certain OS services to the user such as main memory management, I/O handling, networking, etc through an API

b) **What is the purpose of system programs?**

System programs coordinates the activities and functions of hardware and software of a OS, and controls the operations of the hardware.

c) **What is the purpose of the command interpreter? Why is it usually separated from the kernel?**

The command interpreter is also known as the command shell. It facilitates direct communication between the OS and the user to allow for ease of use of the computer, as well as efficiency. In UNIX, shells have unique control structures and syntax, thus allowing shell scripts to be used for automating certain tasks. The command shell is usually separated from the kernel because the kernel is the center of the operating system that manages everything. The shell, on the other hand, is a program that provides an interface and translates commands into low-level calls to the kernel. Another reason is because the command shell is usually subject to changes.

4. **Processes(10 marks)**

Question 3.13: Using the program in Listing 1, identify the values of pid at lines A, B, C, and D (Assume that the actual pids of the parent ahd child are 2600 and 2603, respectively)

The Pids are as follows:
A = 0, as PID is 0 in a child process
B = 2603
C = 2603
D = 2600

Question 3.3: Original versions of Apple's mobile iOS operating system provided no means of concurrent processing. Discuss three major complications that concurrent processing adds to an operating system.

First of all, one needs to be careful with implementing programs using concurrent processing. Deadlocking, where a busy-wait cycle occurs between multiple threads that are waiting for resources that other threads already own while they themselves are waiting for resources, is a consequence that can arise, and freezes the OS as there are no actions that can be performed in a deadlocked state.

Second, processes and system resources must have protections from each other. Any process must be limited in the amount of memory it can use and the operations it can perform on devices such as disks.

Finally, a time sharing method must be implemented to allow each process to have access ot the system. This involes actively preempting processes that do not voluntarily give up the CPU, and the kernel being reentrant (so more than one process may be executing kernel code concurrently).