CS 390
Chapter 1 Homework Solutions

1.1 `What are the three main ...`

The three main purposes of an operating system are

1. To simplify the use of the hardware by providing a convenient environment for the programmer to write programs, and the user to execute and interact with programs;
2. To enforce resource allocation policies;
3. To protect the hardware from accidental or malicious misuse.

1.5 `How does the distinction between ...`

Certain machine instructions can only be executed when the CPU is in kernel mode. These instructions typically are those that allow for direct access to hardware. Thus, the CPU has limited capabilities when executing in user mode, thereby enforcing the protection of critical resources.

1.6 `Which of the following instructions ...`

In this question, the term "instruction" does not refer to an actual machine language instruction / CPU instruction, but to an action performed in software.

a. Yes (Here, the term "timer" refers to the hardware component the generates periodic interrupts, and not the system clock that stores the time and date between boot cycles.)

b. No

c. Yes (This question means "clear the entire contents of the computer's memory".)

d. No

e. Yes

f. Yes

g. No (think carefully about why no is the correct answer)

h. Yes

1.7 `Some early computers protected the ...`

The data required by the operating system (passwords, filenames, accounting information) could not be stored in the same memory as the operating system, because that memory could not be modified. Thus, this information had to be stored in regular memory, where it was readable and writable by regular user processes. This allowed user processes to see information that they should not be able to see (like passwords) and modify data (like the contents of other processes' files) that they should not be able to modify.

In addition, the operating system would not be modifiable at run-time. If a new keyboard or mouse were added, the OS would have to be reconfigured to include the new code, and then recompiled and reinstalled.

1.10 `Give two reasons why caches ...`

1. Caches are useful when two components need to exchange data, and the components preform transfers at different speeds. Caches provide a buffer of intermediate speed between the two components. (Think of the cache memory between the CPU registers and the main memory. Many operating systems also provide a "file cache" where part of the contents of the hard disk is stored in main memory.)
2. Caches make the system more affordable, because they allow a small amount of fast storage to substitute for a much larger but slower storage.

Caches introduce difficulties in programming systems, because when a cache is used, it becomes possible for a piece of data to exist in two places in the system at the same time. Data in the cache must be kept consistent with the actual data in the component that is being cached.

If a cache could be made as large as the component it is caching, it could replace the component. Typically, however, faster components in the memory hierarchy are also more expensive. Also, some caches have different state-saving characteristics than the component they are caching. For example, if files are cached in primary memory, they must be written back to secondary storage before power is removed from the system.

Additional Exercises

A `Give three advantages that multiprocessor systems have over single processor systems. Do multiprocessor systems have any disadvantages of single processor systems?`

Advantages:

Increased throughput: A single multiprocessor machine can complete more work per unit of time.

Economy of scale: It is usually cheaper to construct a single machine with N CPU's than it is to build N machines each with a single CPU.

Increased reliability: Multiprocessor systems may be designed to that if a single processor fails, the system can continue to operate with the remaining good processors.

Disadvantages:

May use resources inefficiently: Unless they are carefully designed and programmed, multiprocessor systems are an inefficient use of hardware resources. An additional CPU gives no advantage unless the operating system schedules processes to run on it.

Not an advantage in batch systems: In addition, systems which run at most one job at a time (called batch systems) will not benefit from multiprocessing. An example of a batch system would be a computer at a bank that prints bank statements each month. (Technically, if the single job is multi-threaded, this statement is not strictly accurate. We will discuss threads later on.)

B `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?`

An interrupt is an electrical signal sent from one hardware component of the computer to another, signaling that an event has occurred which requires immediate action by the component receiving the

interrupt. A trap is a specific type of interrupt which is generated by software and signals either

1. some unusual condition, such as an array index out of bounds, or
2. a request for an operating system service (a system call.)

User programs can generate traps, although not all high-level programming languages make this facility directly available to the programmer. (C, C++, and Java do allow the programmer to generate traps. Traps are called "exceptions" in C++ and Java, and "signals" in C. In addition, C and C++ both allow programmers to bypass their languages' standard libraries and make system calls directly.)

Traps are used to notify the CPU that some unusual software event has taken place, or that some special service has been requested by the program. The CPU can then either terminate the program, or interrupt the normal execution sequence of the program and transfer control to a special segment of code.

C `Direct memory access is used is used for high-speed I/O devices in order to avoid increasing the CPU's execution load.`

a. `How does the CPU work with the device to coordinate the transfer?`

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

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.`

a. The CPU sets up a a buffer in memory where the I/O device will copy the data to/from. It then sends to the device the address of this buffer, along with the number of bytes to transfer.

b. The I/O device interrupts the CPU.

c. Yes, it can interfere with the execution of other user programs. If a user program wants to modify some memory that is being used for DMA I/O, the program must be blocked until the DMA completes. For example, if you are printing the contents of a string in C using printf, the DMA must complete before the next instruction in your program can execute, even if, technically, the CPU isn't being used to more the data to the output device.

In addition, since the I/O device is accessing memory, in can interfere directly with the CPU's memory access, since both typically use the same memory bus. This might cause the CPU to complete fewer instructions per unit of time.

D `Propose a method you might use in an operating system kernel to prevent one process from modifying the memory associated with another process.`

The term "memory protection" means preventing one process from reading or writing the memory of another process without explicit permission. Since the kernel is responsible for allocating memory to processes, the kernel knows, for each memory address, which process the address belongs to. Before a process reads or writes a memory location, the kernel could inspect the address and stop the process when it attempts to access memory that does not belong to it.

E `List some challenges you might encounter in designing an operating system for a mobile device compared to an OS for a traditional PC.`

- Mobile devices may have limited memory. (Apple is somewhat notorious for equiping its iPhones with only a small amount of RAM.) If so, the size of the kernel might be limited, and this might limit the features it can offer. In addition, the kernel might need to be more concerned with the allocation of memory to processes than it would in a general purpose system.
- Power consumption is an important issue with battery powered devices, so the kernel would need to control power usage by the processor and I/O devices.
- A mobile device will probably have a slower or less-sophisticated CPU than a desktop or server machine. That means a mobile

OS might need to be more concerned with the efficient allocation of CPU-cycles among processes than the OS of a larger system.