Noemi Lemonnier
#40001085

COMP 346 Theory Assignment 1

Question 1

I. The operating system is a program that manages a computer's hardware or it can also be designed as one program running at all times on the computer —usually called the kernel. Its main goals are:

- Execute user programs and make solving user problems easier
- Make the computer system convenient to use
- Use the computer hardware in an efficient manner

II.

Batch: They are also referred as noninteractive systems because they handle collections of jobs, which are predefined sequence that are self-sufficient. It was the first system to use multiprogramming and it allows the operating system to do a couple of jobs concurrently.

Time Sharing: They support multiple interactive users. the user will establish an interactive session by giving commands, programs, data as they go during the session. This system will require the operating system to provide timely response for users and be careful about resource management and protection of mechanisms.

Dedicated: It is dedicated to the support of its host system. Some times it has only a single purpose. Embedded system techniques tend to trade off generality of operation for efficiency in order to ensure that real-time processing constraints are met.

Real time: It is based on rigid time requirements that have been placed on the operation of a processor. If the processing exceed such defined constraints, the system will fail.

Multiprogramming: It increases CPU utilization by organizing jobs (code and data) so that the CPU always has one to execute.

III.

There is 3 situations depending on what the user would prefer: when it is cheaper, faster, or easier. For example:

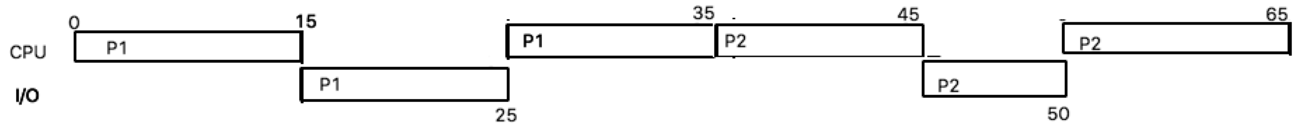
1. When the user is paying for management costs, and the costs are cheaper for a time-sharing system than for a single-user computer.

2. When running a simulation or calculation that takes too long to run on a single PC or workstation.

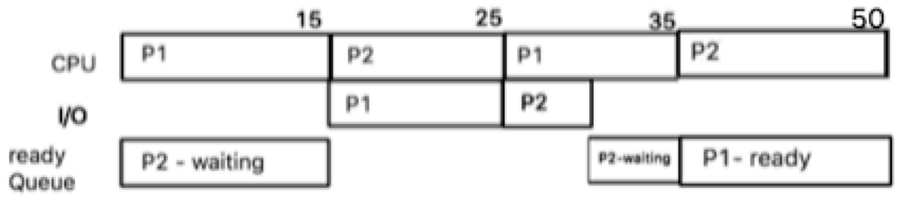
3. When a user is travelling and doesn't have laptop to carry around, they can connect remotely to a time-shared system and do their work.

Noemi Lemonnier
#40001085

Question 2
a) First-come First-Serve scheduling algorithm
Minimal time is 65 ms



b)
Minimal time is 50 ms

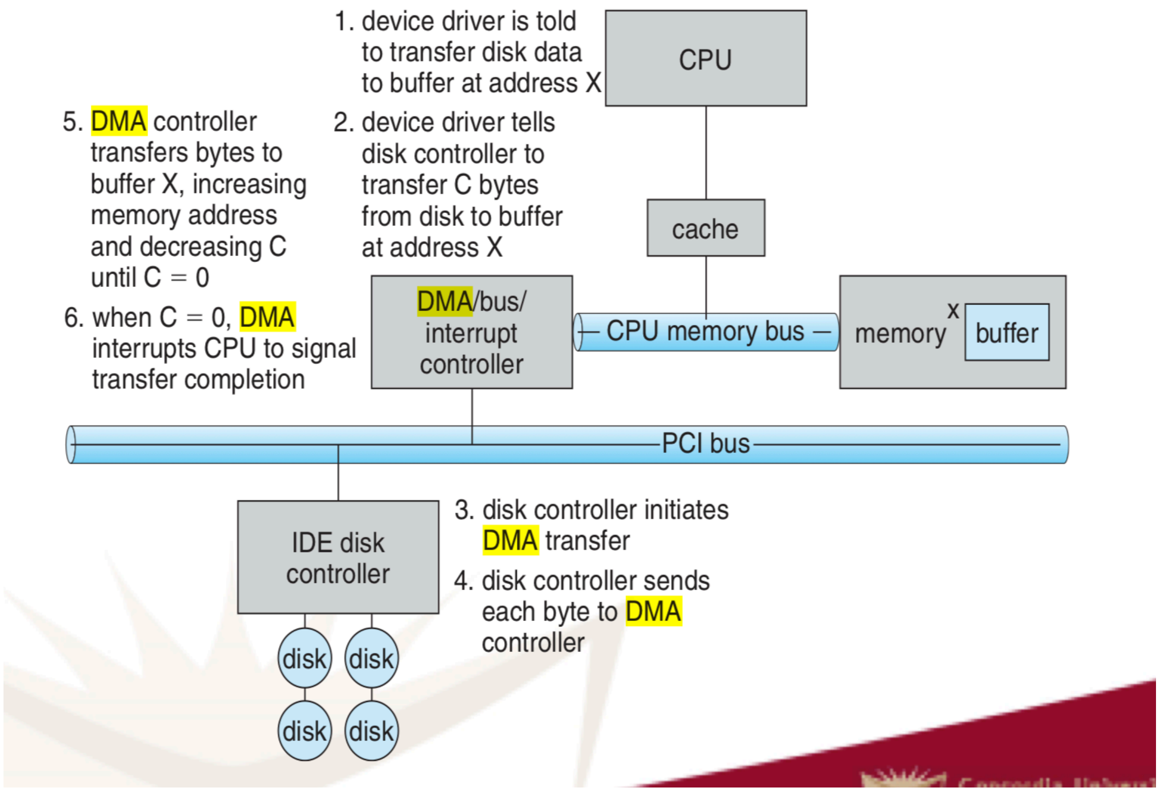


c) For a) 2/65 which is throughput = 0.03 and for b) throughput = 2/50 = 0.04. The multiprogramming affects the throughput because it manages tasks more efficiently so the throughput is bigger which means the scheduler is working properly.

Question 3
I) An interrupt has many advantages as the CPU can answer to different interrupts depending of their priority and react to the interrupt. Also the interrupt handler can decided if an interrupt is maskable –cant be ignored, or nonmaskable – can not be ignored.
Polling can be more advantageous in some situations as frequency of I/O, the mouse ps3 to mouse USB. Because it is more efficient when the I/O is frequent and of short duration.

II) In a DMA the memory can be accessed via the I/O. A DMA relies on interrupt handling because according to this image:

Noemi Lemonnier
#40001085


1. device driver is told to transfer disk data to buffer at address X
2. device driver tells disk controller to transfer C bytes from disk to buffer at address X
3. disk controller initiates DMA transfer
4. disk controller sends each byte to DMA controller
5. DMA controller transfers bytes to buffer X, increasing memory address and decreasing C until C = 0
6. when C = 0, DMA interrupts CPU to signal transfer completion
CPU
cache
CPU memory bus
DMA/bus/interrupt controller
memory
x
buffer
PCI bus
IDE disk controller
disk
disk
disk
disk

It will go to the CPU, then the CPU react to the interrupt by using the bus, then the disk controller will initiates the DMA transfer, to then send each byte to DMA controller, etc. Until the process is done and the interrupt will be resolved so the CPU can answer another interrupt.

If the system does not handle interrupt, it means you will need to use polling, which is not supported usually.

IV)
a) The Context Switch needs to be atomic as it needs to ensure consistency. Consistency will allow to do a save operation if a failure occurs: it will store the CPU registers' values, the process state, etc. For example, if an interrupt occurs, it will wakeup the scheduler OS. It will take everything on CPU and save it to memory. Then it will take the process state P2 and put it to the CPU memory. It is very costly

b) In practice, atomicity can be achieved by synchronizing the save and load methods, by using threads.

Question 4
I) We need a secure way to access critical resources that the user processes have to share. The overhead is not much compared to what can happen if I/O are performed in user mode.

Noemi Lemonnier
#40001085

II)
a) The user needs to have put his code in the kernel. As a result, he can have the ability to change the interrupt vector that is in the kernel or user space

b) To ensure this kind of issue does not happen, the system needs to have a security system that will not allow the user to go in administration mode.

Question 5
a) It would be possible if the system has multi-cores.
b) It would be possible, if we have a main thread that launches other threads because they are light weight and fast.

Question 6
Clear memory, Switch from user to monitor, Copy from one register to another, turn off interrupts cannot be done in user mode so it needs to be privileged.

Question 7
A network OS connects different independent computers that are connected on a server. For example, if you and a co-worker are working in an office and you are on a network OS. It means you each have your own computer but you can have a shared disk on it and share your work. A distributed OS is one OS where all computers that are connected can share a task. For example, if you work with a co-worker using a distributed OS, it means you can work on their computer.

They will have a common hardware but will be different in their software.