Introduction to OS Concepts

What is the Job of an OS?

- Two views...
  - An interface between users/applications and hardware.
  - A hardware abstraction.

OR

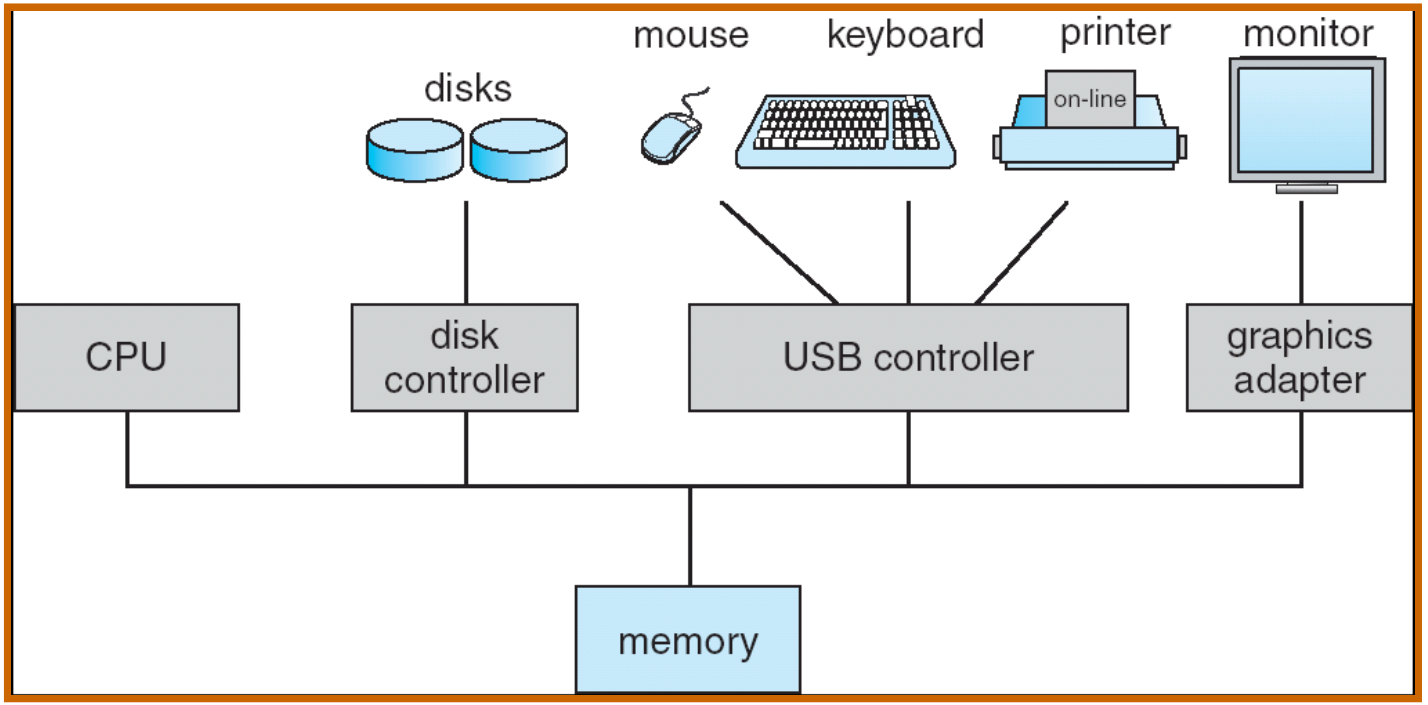
  - A resource allocator, striving for,
    - Efficiency
    - Fairness
    - Security

What Constitutes an OS?

- No agreed upon definition.
- Candidates:
  - All of the software that comes with a computer.
  - The one program that is always running.
  - The kernel.
  - The software responsible for interacting directly with hardware.
- Wrinkles:
  - Micro-kernels

Computer Hardware Review

- A simplified picture...



How Many Processors In That Picture?

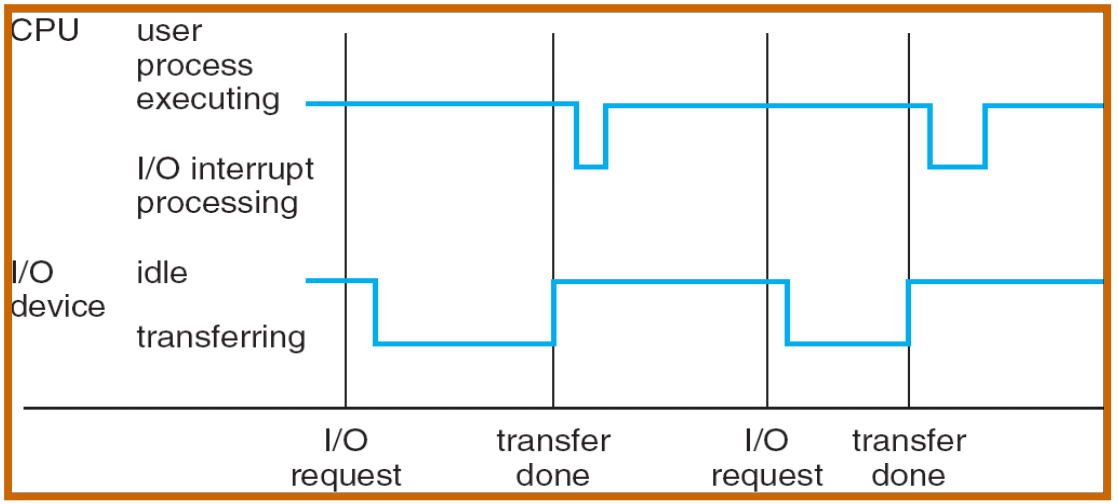
- The CPU executes arbitrary program instructions.
- Different devices have their own processors and buffers to manage device control.

CPU

- A very simple life – one instruction after another (OIAA)
- Life starts when the bootstrap program loads the OS (or an OS loader, or whatever it finds), and points the CPU to the first instruction.
- An exception to OIAA: interrupts.
  - Handling interrupts will be one of the OS's jobs.

Interrupts

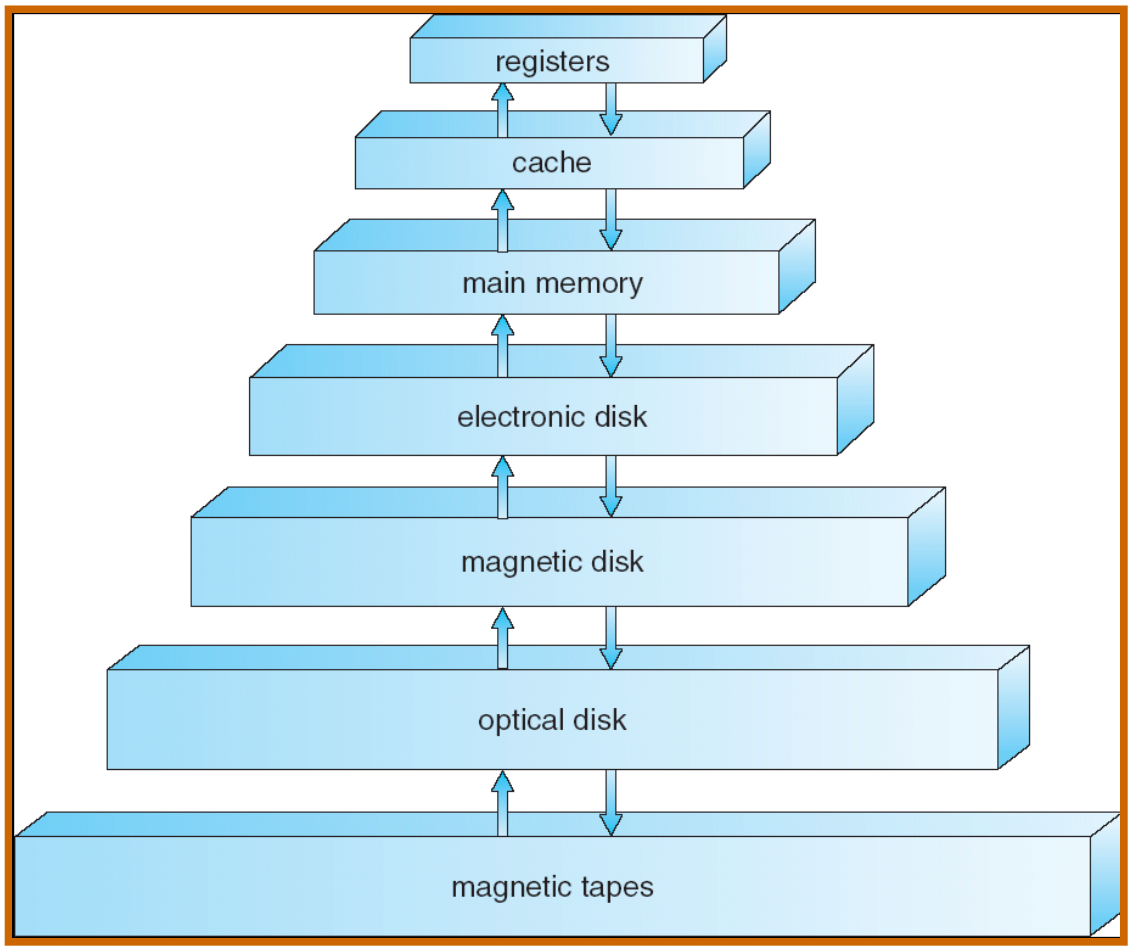
- Interrupt vector contains addresses of interrupt service routines.
- A trap is a software generated interrupt.
- An OS is interrupt driven.
- It doesn't do anything until an interrupt occurs.



Device I/O

- Two basic flavors:
  - Programmed I/O – CPU reads directly from device registers.
  - DMA – Direct memory access. Device and memory communicate directly.
    - More efficient for large transfers.

The Memory Hierarchy



Caching

- Fundamental issues:
  - We want CPU to be able to load and store data as quickly as possible.
  - We want storage to be cheap, but cheap storage is slow.
- Solution is caching:
  - Keep the data most likely to be accessed in small fast storage.
  - Not an easy problem, and one that will come up in operating system design over and over.

System Architectures

- So far we have been talking about single CPU systems.
- Multiple CPU systems are becoming more common, and raise a new set of issues:
  - Cache coherency.
  - Efficient utilization of multiple CPUs.

Operating System Structure

- Modern operating systems are designed to be:
  - Multiprogrammed – multiple programs running “at once”.
  - When one program takes a break, another is ready to step in.
  - Time shared – multiple interactive applications, possibly multiple users, running at once.
- Accomplishing this requires solving many problems...

Process Management

- A process is an executing program.
- The OS must:
  - Create and destroy
  - Pause and resume
  - Allow for synchronization
  - Allow for communication
  - Avoid deadlock

Memory Management

- Potentially more memory in use by all of the active processes than exists in the system.
- A solution is virtual memory.
- OS must allocate memory, and handle caching.

File System Management

- Disk controllers basically present secondary storage as an undifferentiated place to put 0's and 1's.
- It is the OS's job to organize that into a file system.

I/O Management

- A large chunk of the code in an OS is device drivers.
- The specific protocols for dealing with many different devices.

Networking

- Protocols for exchanging information with other computers.

Protection

- Users need to be protected from each other.
  - E.g. permissions.
- Processes need to be protected from each other.
  - E.g. virtual memory.
- The system needs to be protected from malicious or erroneous code.
  - E.g. kernel mode bit.

Security

- The system needs to be protected from malicious outsiders.

Stuff We Won't Talk About

- Real time OS's
- Embedded OS's
- Hand-held OS's