Operating Systems

Instructor: Dr. Liting Hu

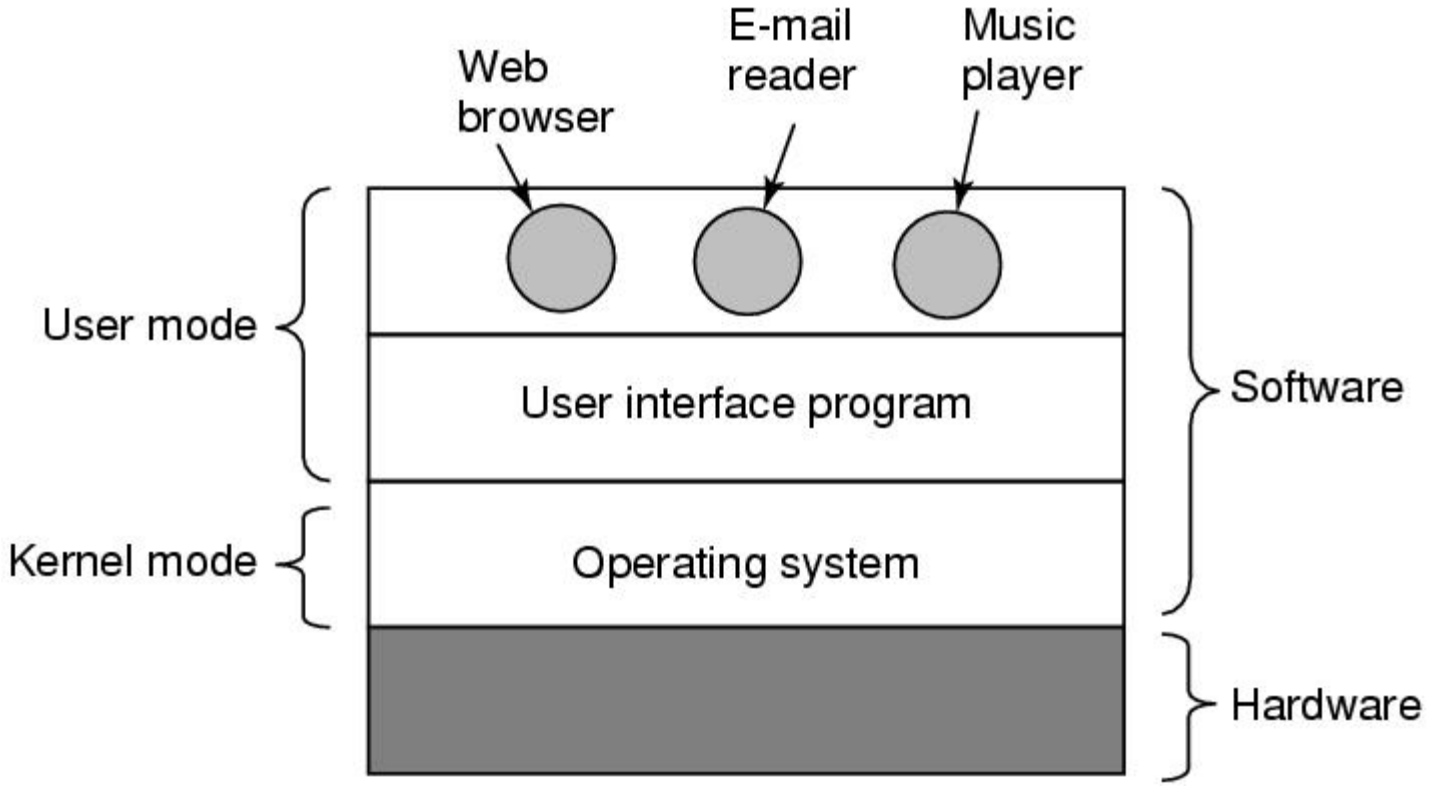
What is an operating system?

Lots of hardware

- One or more processors
- Main memory
- Disks
- Printers
- Various input/output devices

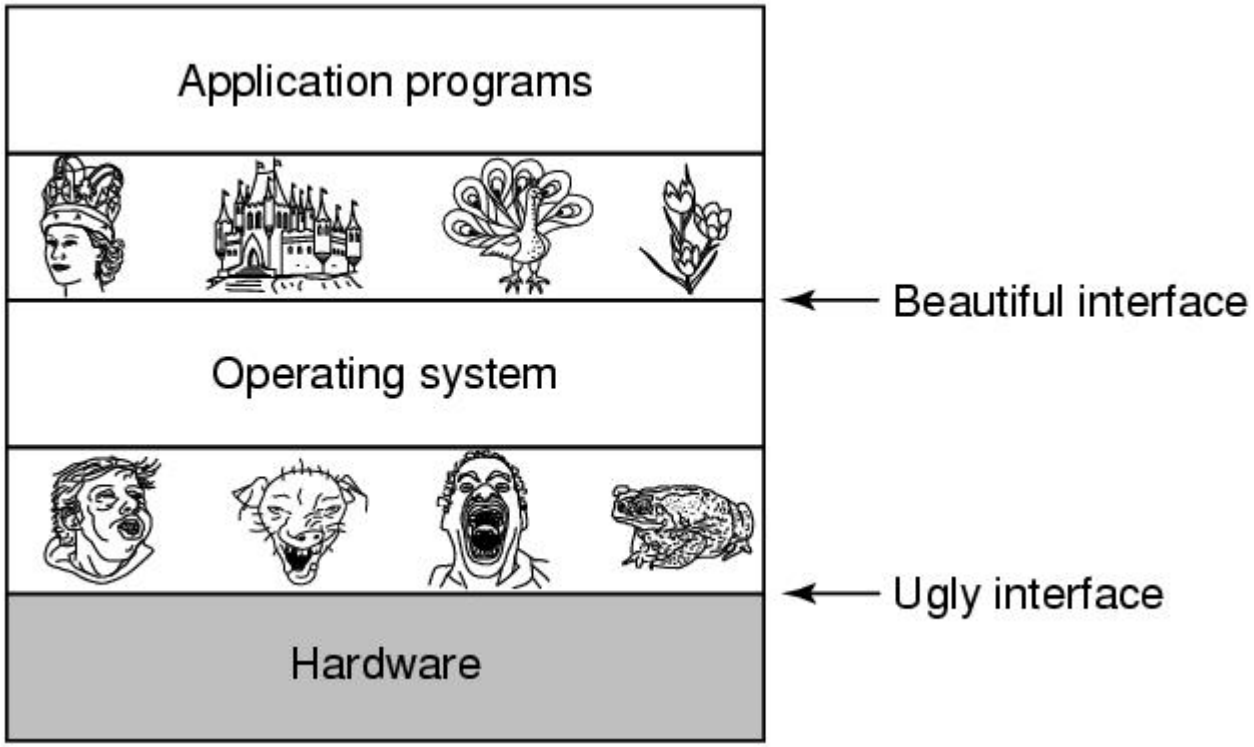
Managing all these components requires a layer of software – the **operating system**

Where is the software?



Where the operating system fits in?

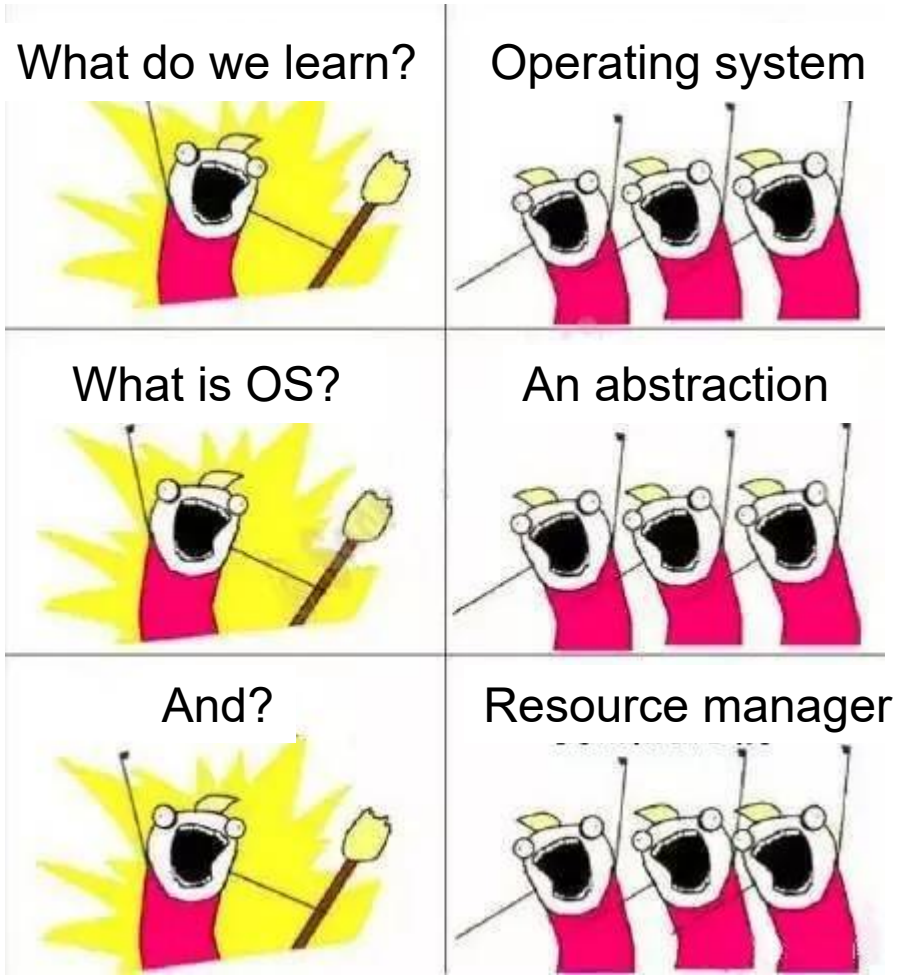
The Operating System as an Extended Machine



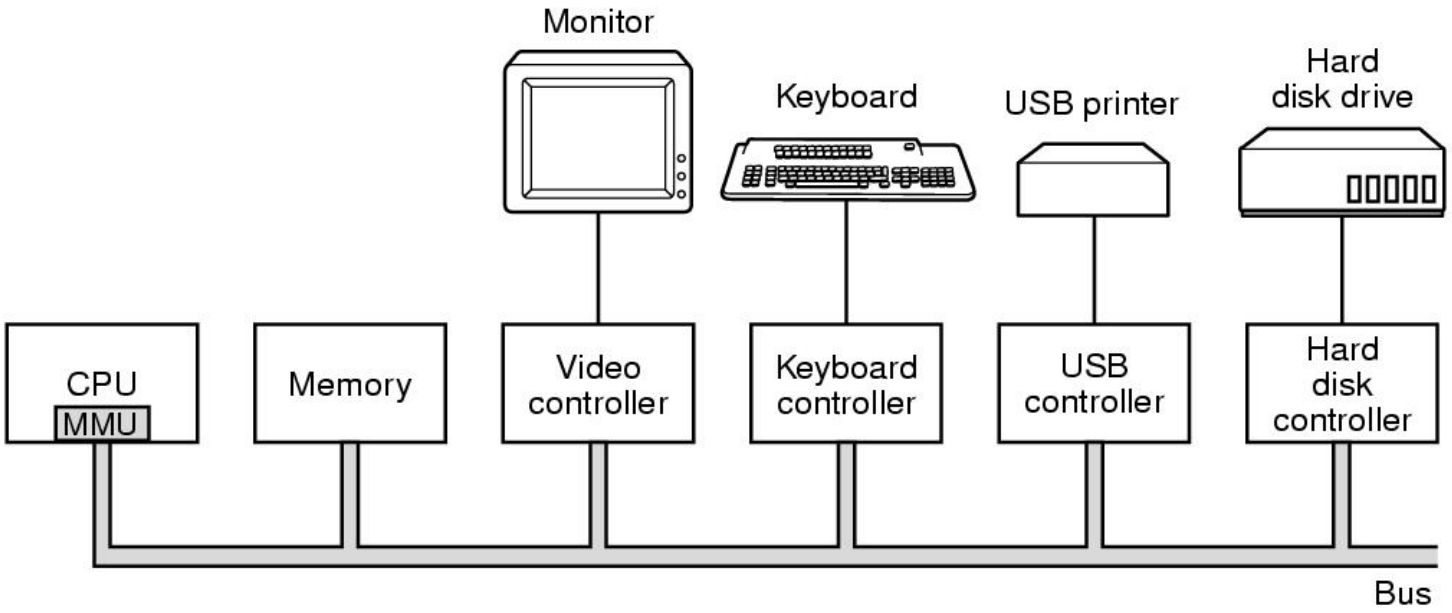
The Operating System as a Resource Manager

- Allow multiple programs to run at the same time
- Multiplexes (shares) resources in two different ways:
  - **In time**
  - **In space**
- Manage and protect memory, I/O devices, and other resources

What is an operating system?



Computer Hardware Review



Some of the components of a simple personal computer.

CPU Pipelining

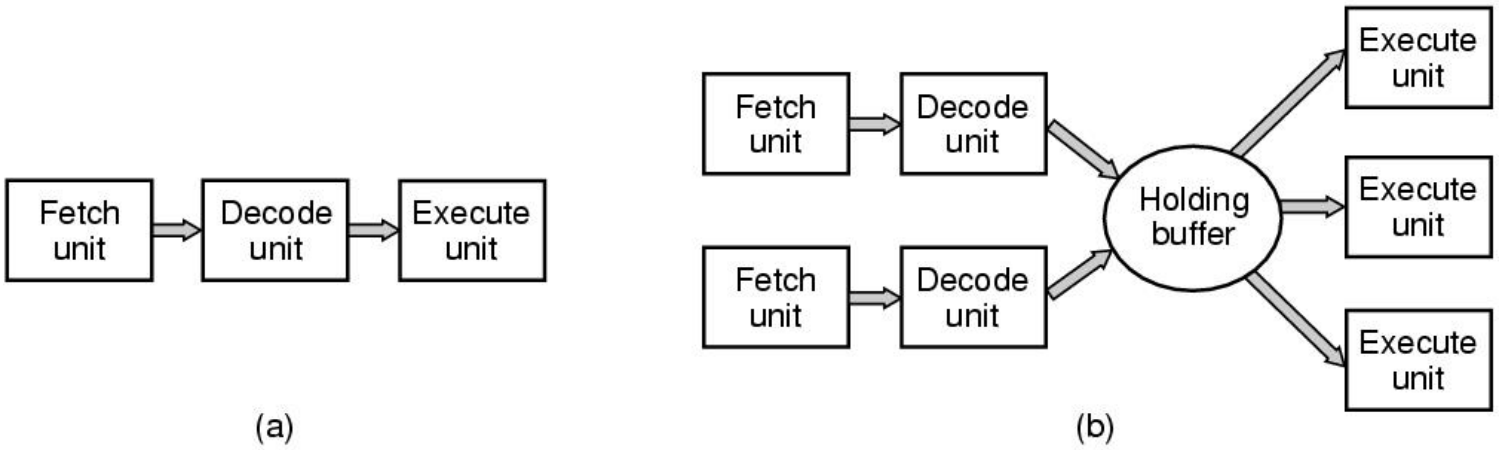
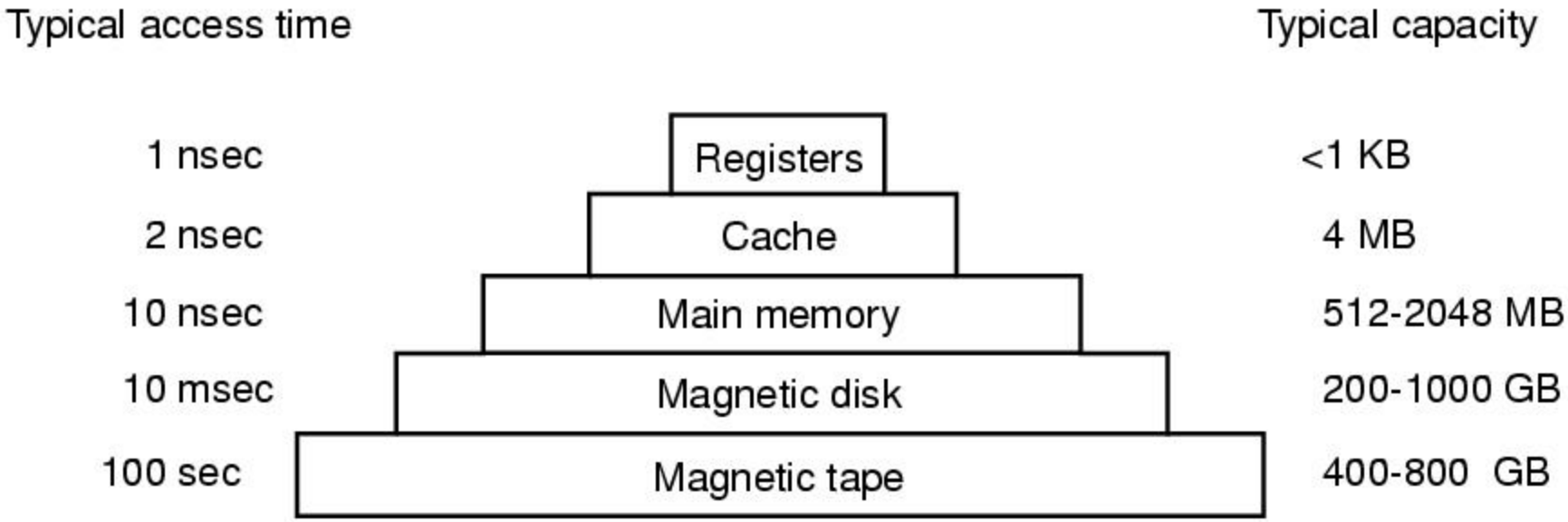


Figure 1-7. (a) A three-stage pipeline. (b) A superscalar CPU.

Memory Hierarchy



A typical memory hierarchy. The numbers are very rough approximations.

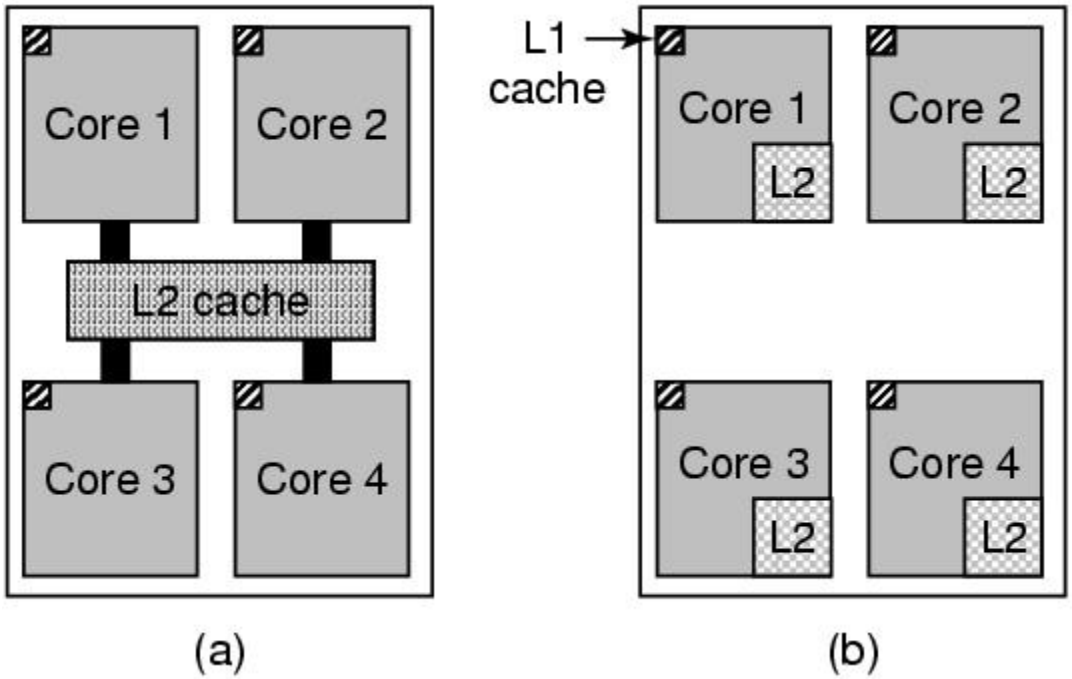
Main Memory

- **RAM**
- **ROM** – Can not be changed. Fast, cheap
- **EEPROM** (Electrically Erasable PROM) Can be re-written, but slowly
  - E.g.-serves as films in digital cameras, as disk in portable music players

Caches

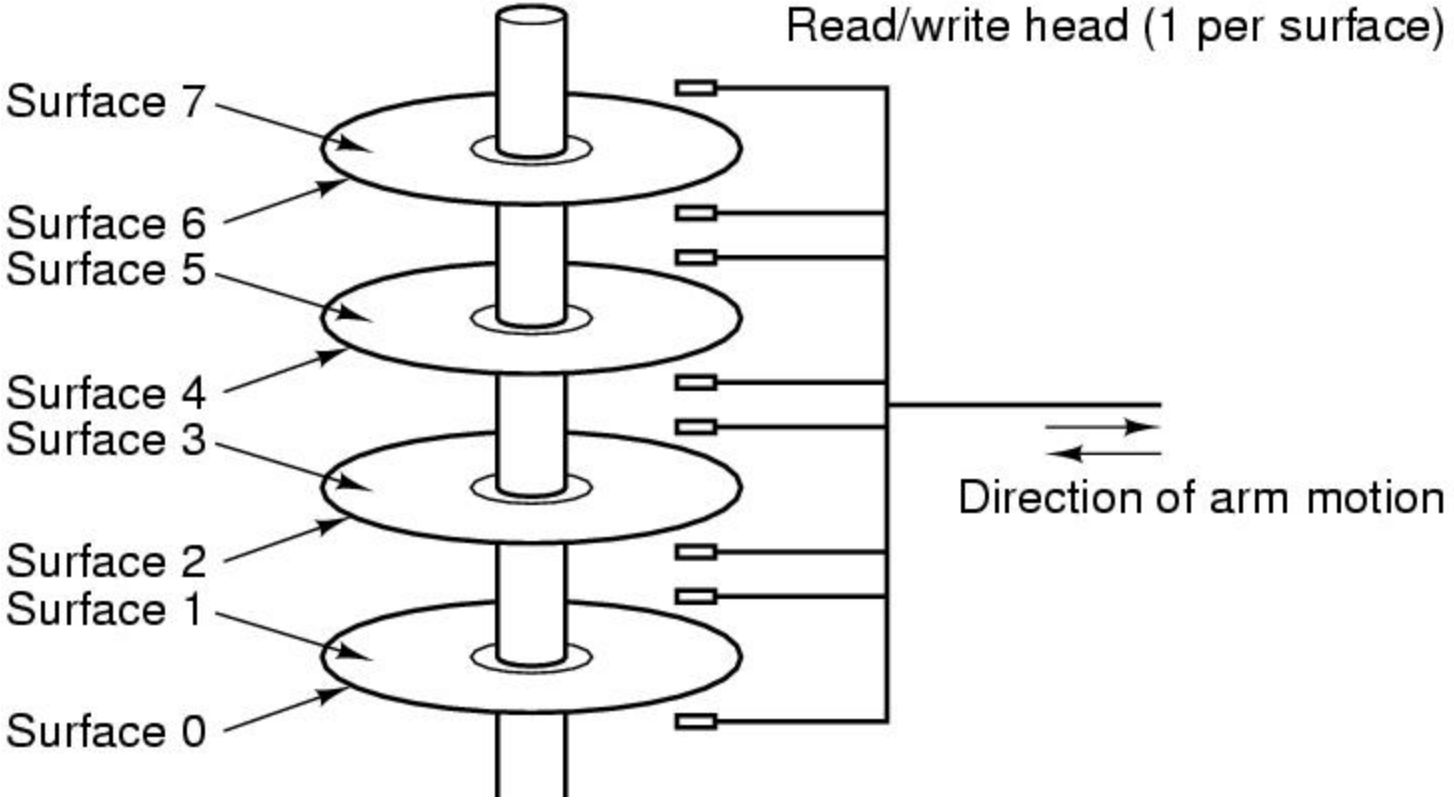
- Main memory is divided into **cache lines (64 bytes)**
- When program reads a word-cache hardware checks to see if in cache.
  - If so, then have a cache hit (2 cycles).
  - Otherwise, make request of main memory over the bus (expensive)
- Cache is expensive and is therefore limited in size
- Can have cache hierarchies
- Cache other things, like URL addresses

Multithreaded and Multicore Chips



(a) A quad-core chip with a shared L2 cache.
(b) A quad-core chip with separate L2 caches.

Disks


Read/write head (1 per surface)
Surface 7
Surface 6
Surface 5
Surface 4
Surface 3
Surface 2
Surface 1
Surface 0
Direction of arm motion

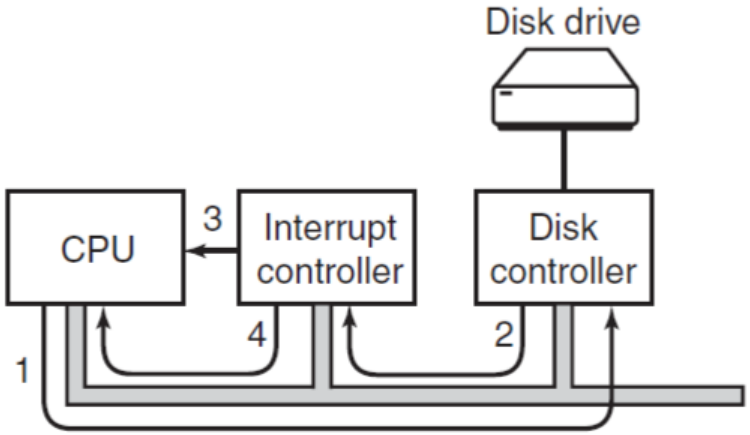
Device Driver

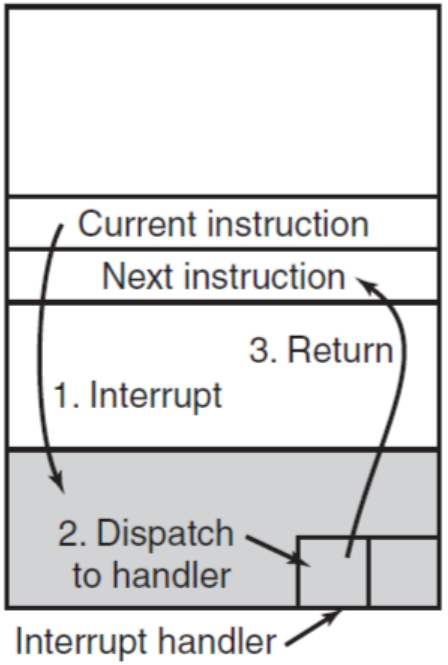
- Complicated business
  - Eg. Gets command to read sector x on disk y. Must convert to (cylinder, sector, head) address, move arm to correct cylinder, wait for sector to rotate under the head, read and store bits coming off the drive, compute checksum, store bits as words in memory
- Each device controller manufacturer supplies a driver for each OS
- Three modes of communication
  - Polling
  - Interrupt
  - DMA

I/O by polling device

- Driver issues command to controller
- Driver polls device until it is ready
  - Eg Send character to printer controller and poll until it is ready to accept the next character
- Big use of CPU
- Called programmed I/O-not really used any more

I/O by Interrupts



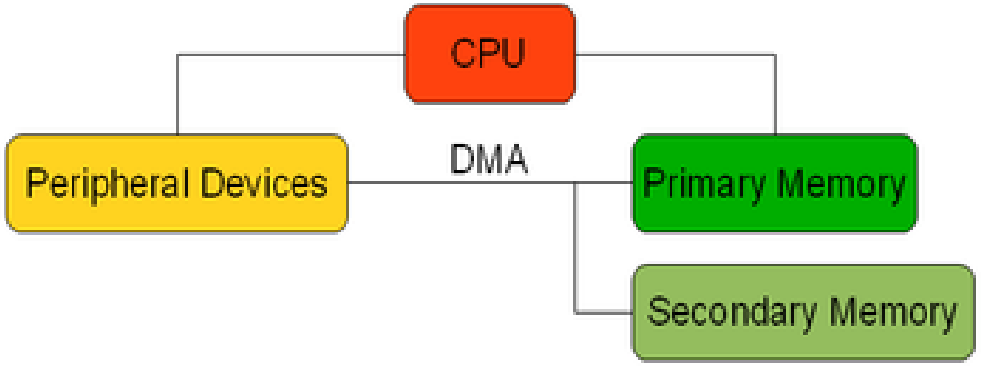


Generate an interrupt when I/O is finished.

Eg When character is finished being printed, interrupt CPU. Allows CPU to do something else while character is being printed

I/O by DMA

- Special (controller) chip
- Avoids using the CPU as part of the transfer to/from memory
- Avoids using the CPU as part of the transfer to/from memory
- Chip does as it told and interrupts CPU when it is finished



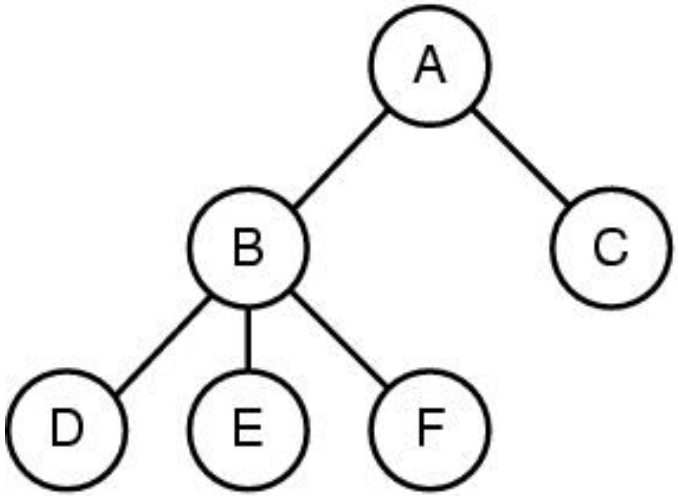
Operating System Concepts

- Processes
- Address spaces
- Files
- I/O and Protection
- System calls

Processes

- Program in execution
- **Lives in address space**
- **Process table**
  - Keeps info about process
  - Used to re-start process
- Can communicate with one another
- Have UID's and group ID's (GID)
- Can communicate with one another (IPC)

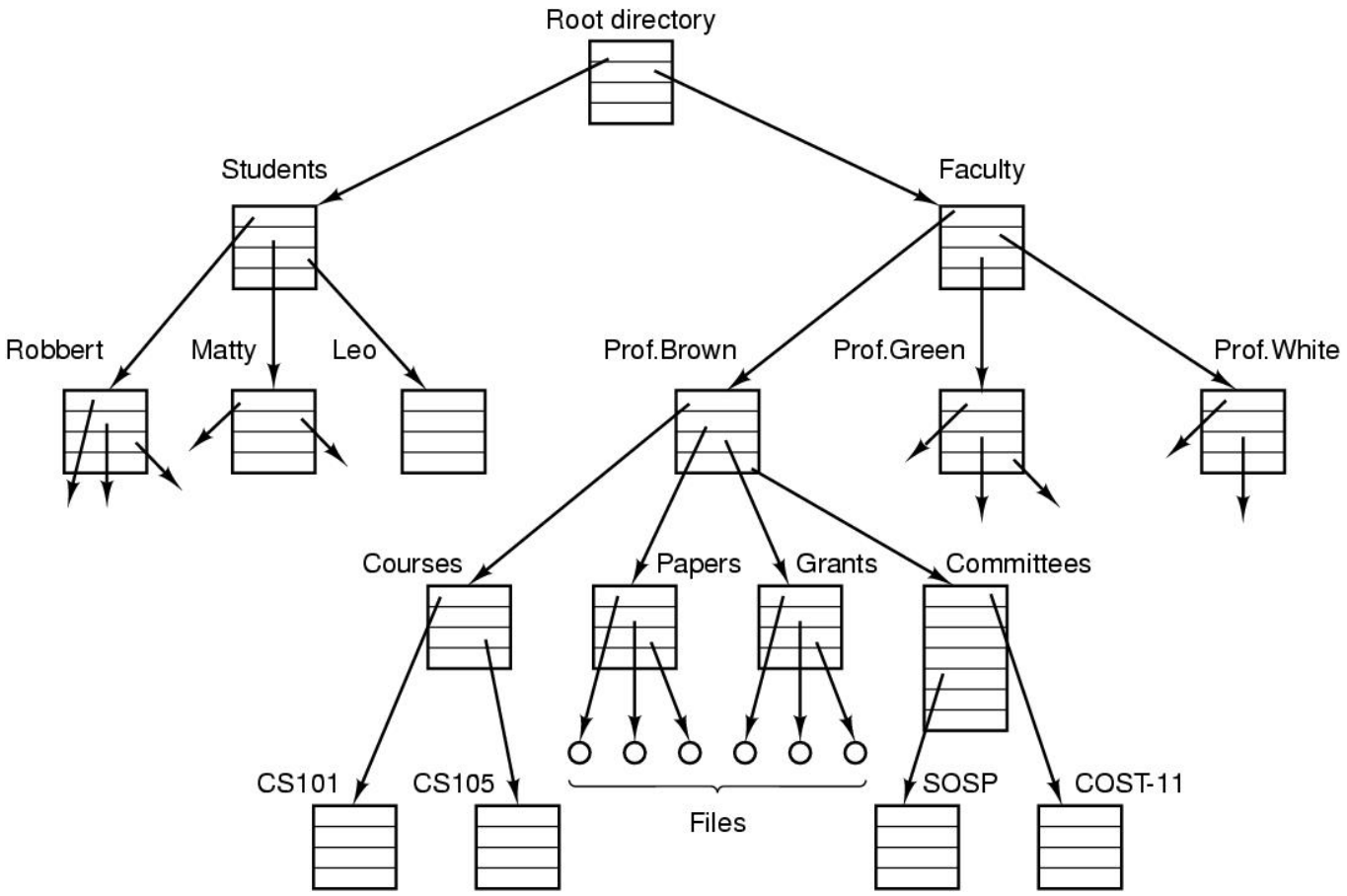
A Process Tree



Process creates child processes

Tree structure

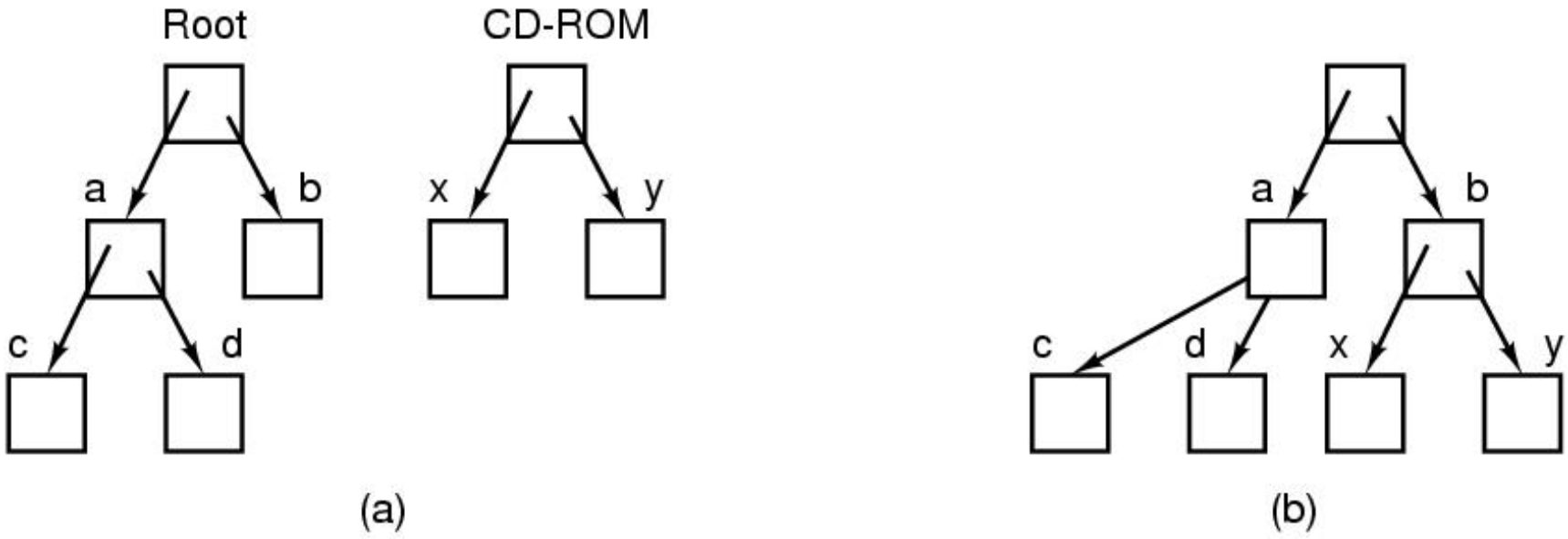
File directory



The directory is organized as a tree.

Root directory at the top. Path proceeds from the root

Mounting Files in UNIX

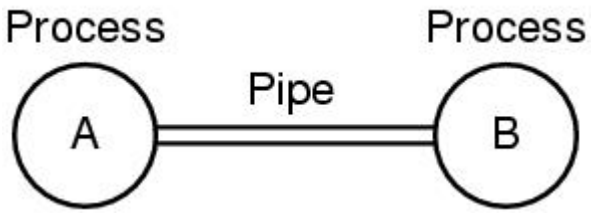


A CD-ROM is mounted on directory b.

Special files

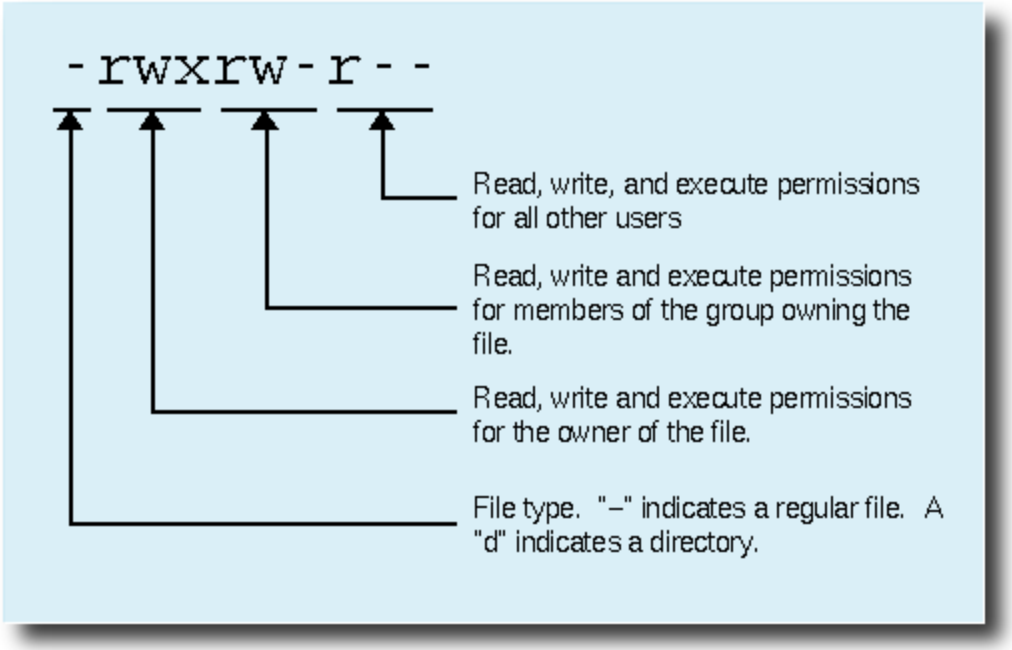
- Special files-can use same calls for I/O as for files. OS treats them as files.
  - Block special files (disks)
  - Character special files (line printers, modems)
  - Kept in /dev directory, e.g. /dev/lp is line printer

Processes communicate by writing into/reading from a file in Unix



I/O, Protection/Shell

- I/O-big part of OS
- Protection-UNIX uses rwx bits for each file
  - 3 bits for owner, 3 for group, 3 for everyone else



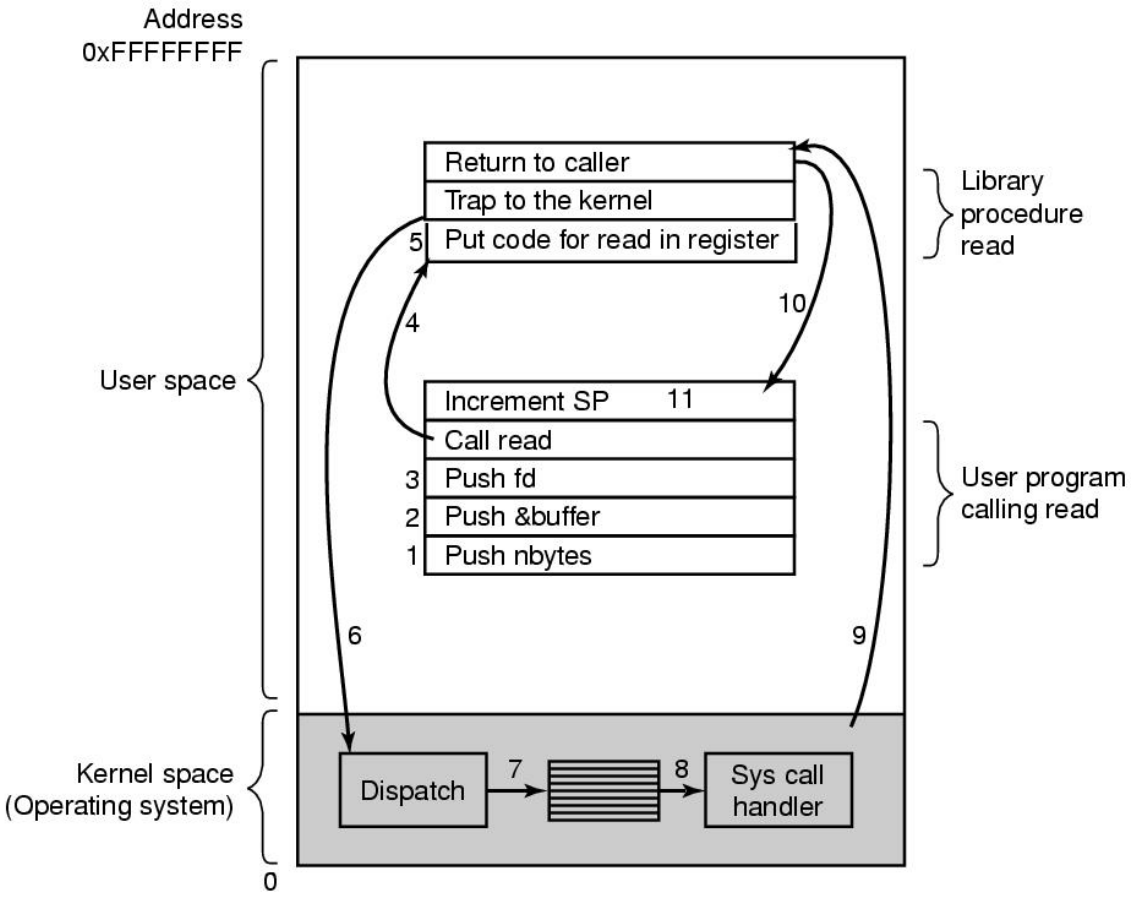
System calls

- Interface between user programs and OS
  - Varies from OS to OS
- System call issued by user program
  - Call uses a library call of the same name
- Library routine puts machine into kernel modes (by issuing a special instruction)
- Finds actual routine for system call in a table
- Does the work involved in the call
- Returns to user program

Unix Read System Call

- **Count = read(fd, buffer, nbytes)**
  - fd is a file descriptor. When a file is opened, permissions are checked. If access is allowed, a number (fd) is returned. Then file can be read/written
  - nbytes is number of bytes in file
  - buffer is where read deposits the bytes

Count = read(fd, buffer, nbytes)



System call examples

Process management

Call	Description
pid = fork( )	Create a child process identical to the parent
pid = waitpid(pid, &statloc, options)	Wait for a child to terminate
s = execve(name, argv, environp)	Replace a process' core image
exit(status)	Terminate process execution and return status

File management

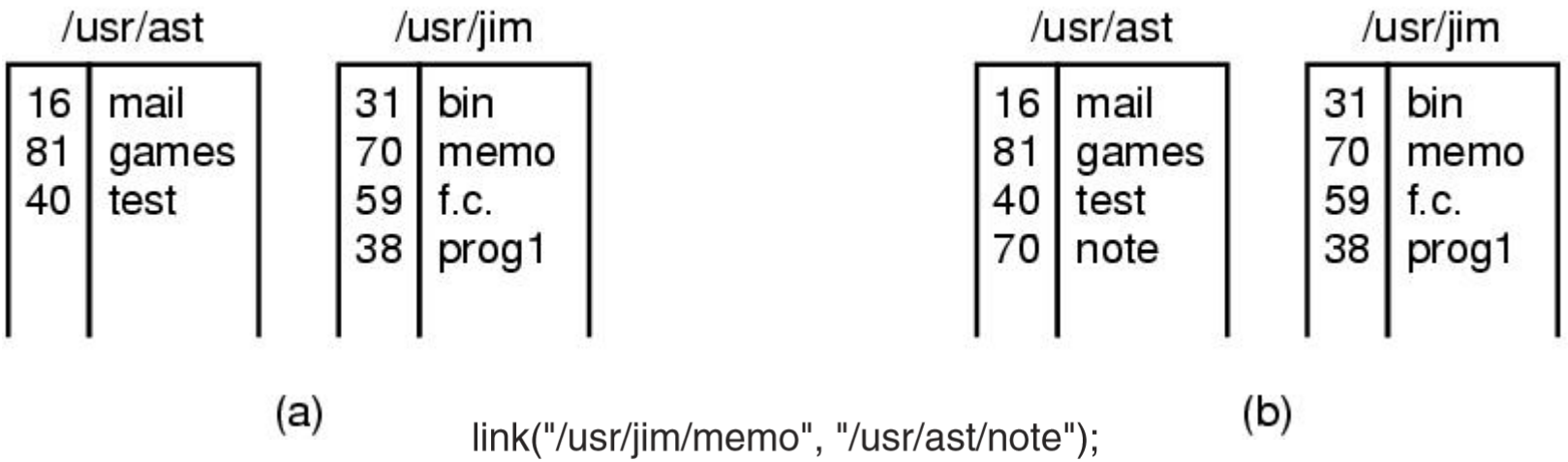
Call	Description
fd = open(file, how, ...)	Open a file for reading, writing, or both
s = close(fd)	Close an open file
n = read(fd, buffer, nbytes)	Read data from a file into a buffer
n = write(fd, buffer, nbytes)	Write data from a buffer into a file
position = lseek(fd, offset, whence)	Move the file pointer
s = stat(name, &buf)	Get a file's status information

System call examples

Call	Description
s = mkdir(name, mode)	Create a new directory
s = rmdir(name)	Remove an empty directory
s = link(name1, name2)	Create a new entry, name2, pointing to name1
s = unlink(name)	Remove a directory entry
s = mount(special, name, flag)	Mount a file system
s = umount(special)	Unmount a file system

Call	Description
s = chdir(dirname)	Change the working directory
s = chmod(name, mode)	Change a file's protection bits
s = kill(pid, signal)	Send a signal to a process
seconds = time(&seconds)	Get the elapsed time since Jan. 1, 1970

Linking



In Unix, each file is identified by an i-number

i-number indexes into i-node table

Link creates a new directory entry with same i-number

Has a new name (note instead of memo)

Operating Systems Structure

- Microkernels
  - Small number of processes are allowed in the kernel
  - Put mechanism in kernel and policy outside the kernel
  - Minimizes effects of bugs

- Monolithic systems
  - A main program that invokes the requested service procedure.
  - A set of service procedures that carry out the system calls.
  - A set of utility procedures that help the service procedures.

Takeaways

- Explain the main purpose of an operating system?
- What is a trap instruction? Explain its use in operating systems.
- What is i-Node? Explain its use in operating systems.
- What is virtual memory and virtual memory address?
- What are time sharing systems? What are space sharing systems?
- What is CPU pipeline?
- What are interrupts?