CSI 4500 Operating Systems

Lecture 1

What is an Operating System?

Bad News

- This is a TOUGH course
  - One of the “most difficult” courses rated by CSE alumni
- Unfamiliar low-level systems programming
  - C and assembly
  - No abstraction, close to hardware

Good News

- Heavy, but totally worth it
  - One of the “most useful courses after graduating”
- How to get a good grade (pass)?
  - Works hard!
  - Ask questions in class
  - Attend office hours
  - Study group
  - Etc.
- We will do our best to help you

Why Study Operating Systems

- The most complex software
  - ~8 million lines of code in Linux
- Arguably the most fundamental software
  - OSes are almost everywhere, e.g., Supercomputer, PC, phone…
  - Almost everything we do with computers through OS

What you should learn in this course

- You will learn how computers work
- You will gain a solid understanding of operating system concepts and hardware
  - A big picture of a computer system: how do hardware, programming language, compiler, algorithms, and OS work together?
- You will learn about system design and system research
  - develop intuition for which approaches work, and which don't
  - develop the ability to sense where bottlenecks lie in system design
  - remember where to look for more information when you are faced with a system problem
    - Simplicity, portability, performance, and trade-offs

Goals for Today

- To provide a grand tour of the major operating systems components

- To be able to answer the question of what an operating system is.

Interactive is important! Ask Questions!

Note: Some slides and/or pictures in the following are adapted from Operating System Concepts textbook website: http://codex.cs.yale.edu/avi/os-book/OS9/index.html

Basic Role of a multi-user multi-tasking Operating System

users





programs

email
chat

graphics
photos

spreadsheets
word processing

compiler
editor

System
Administration

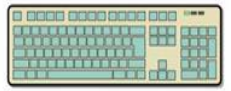
Operating System

hardware









Processor
Motherboard
Monitor
Memory
CD/DVD Drives
Network Adapters
Hard Drives
Keyboard

Basic Role of a multi-user multi-tasking Operating System

users

programs

email
chat

graphics
photos

spreadsheets
word processing

compiler
editor

System
Administration

Operating System

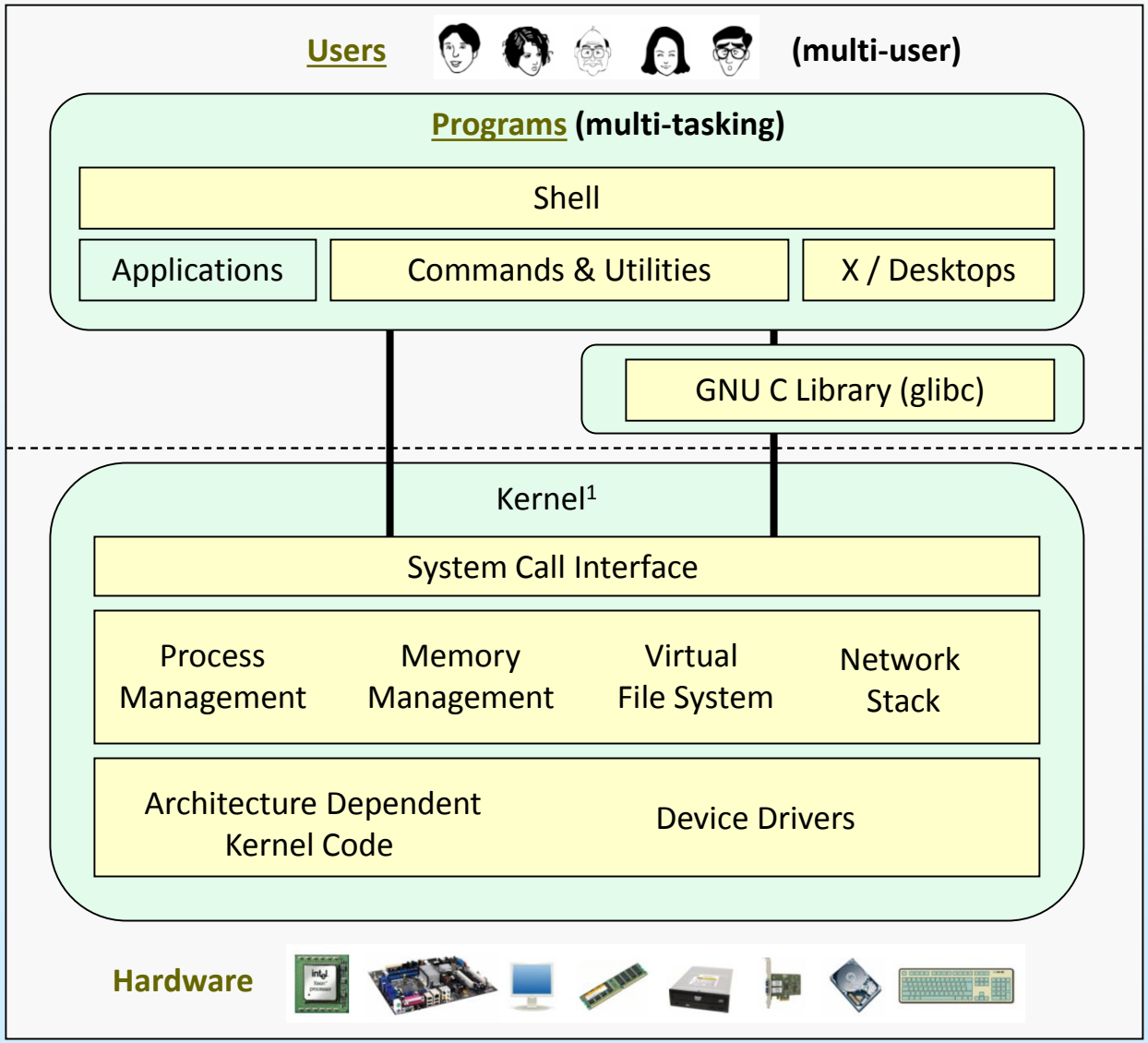
Juggles **users & programs** across limited **hardware** resources

- Runs programs for multiple users
- Provides common services for programs and users
- Shares hardware resources between competing programs and users

hardware

Processor | Motherboard | Monitor | Memory | CD/DVD Drives | Network Adapters | Hard Drives | Keyboard

GNU/Linux Operating System Architecture



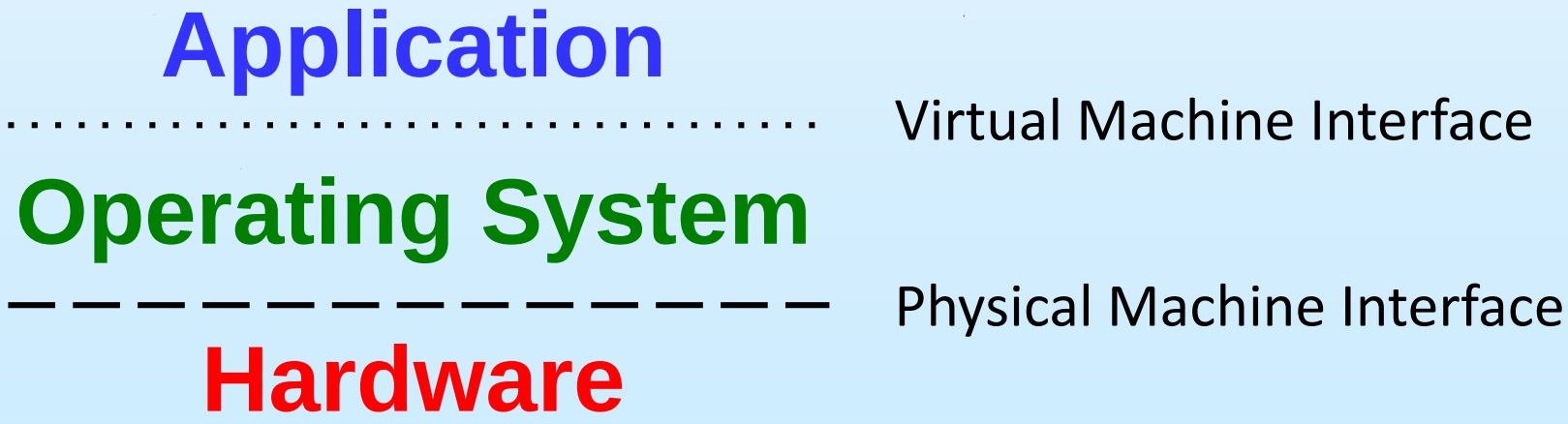
Richard Stallman started the GNU project in 1983 to create a free UNIX-like OS. He Founded the Free Software Foundation in 1985. In 1989 he wrote the first version of the GNU General Public License

Linus Torvalds, as a student, initially conceived and assembled the Linux kernel in 1991. The kernel was later re-licensed under the GNU General Public License in 1992.

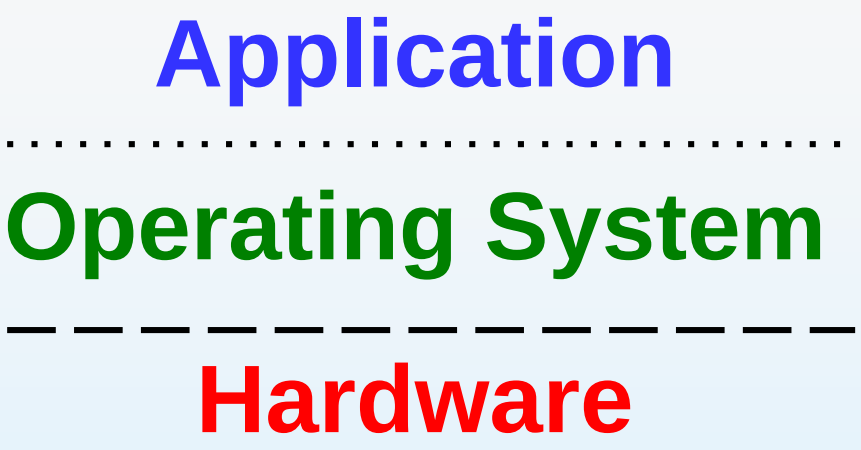
[1]See "Anatomy of the Linux kernel" by M. Tim Jones at http://www-128.ibm.com/developerworks/linux/library/l-linux-kernel/

What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware.
- Operating system goals:
  - Execute user programs and make solving user problems easier.
  - Make the computer system convenient to use.
- Use the computer hardware in an efficient manner.



Operating System as an Extended Machine




```
fprintf(fd, "%d", data);

write(fd, buffer, count);

file->f_op->write(file, buffer, count, pos);

        .
        .
        .
        .

load(block, length, device);

seek(device, track);

out(device, sector);
```

What does an Operating System do?

- OS is a **resource allocator**
  - Manages all resources
    - Time (?), Space (?)
  - Decides between conflicting requests for efficient and fair resource use
    - Who is the user?

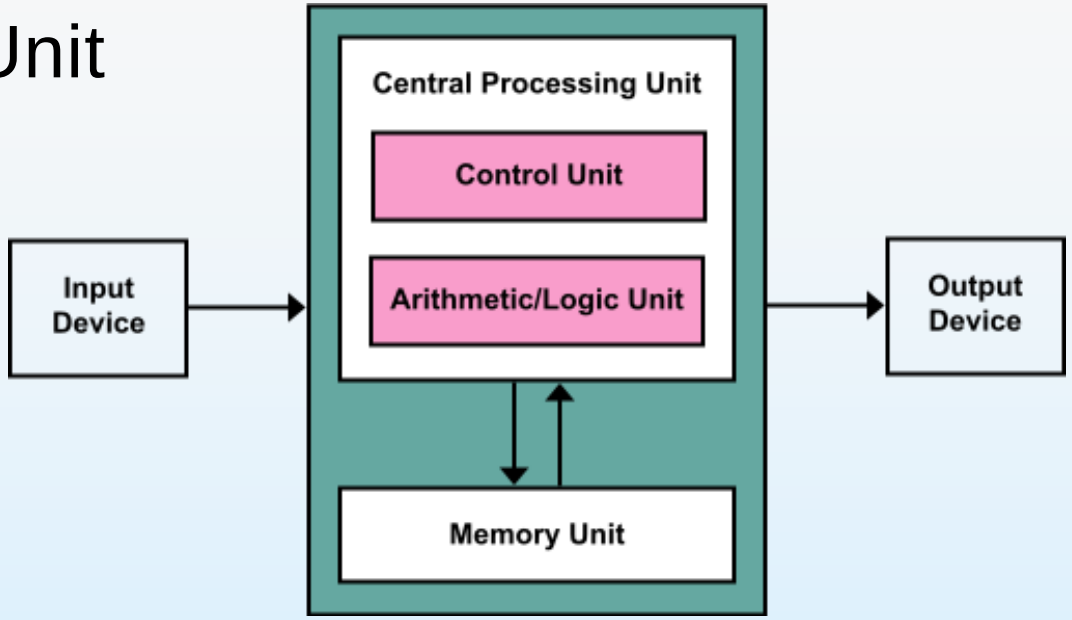
- OS is a **control program**
  - Controls execution of programs to prevent errors and improper use of the computer

Operating System definition

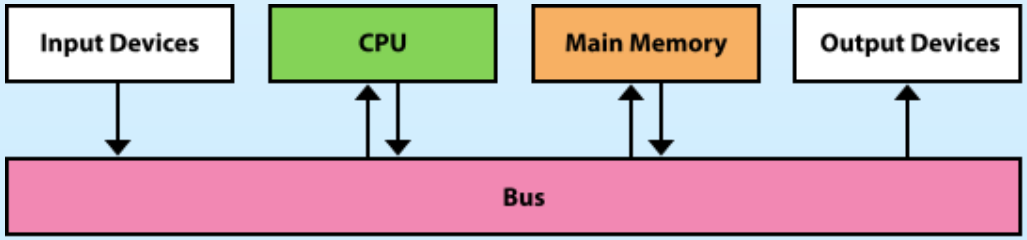
- An operating system (OS) is a collection of software that manages computer hardware resources and provides common services for computer programs. –wikipedia
- "Everything a vendor ships when you order an operating system" is good approximation
  - But varies wildly
- "The one program running at all times on the computer" is the **kernel**. Everything else is either a system program (ships with the operating system) or an application program

Computer System Overview

- Central Processing Unit
  - —Control Unit
  - —ALU
- Memory
- IO
- Bus

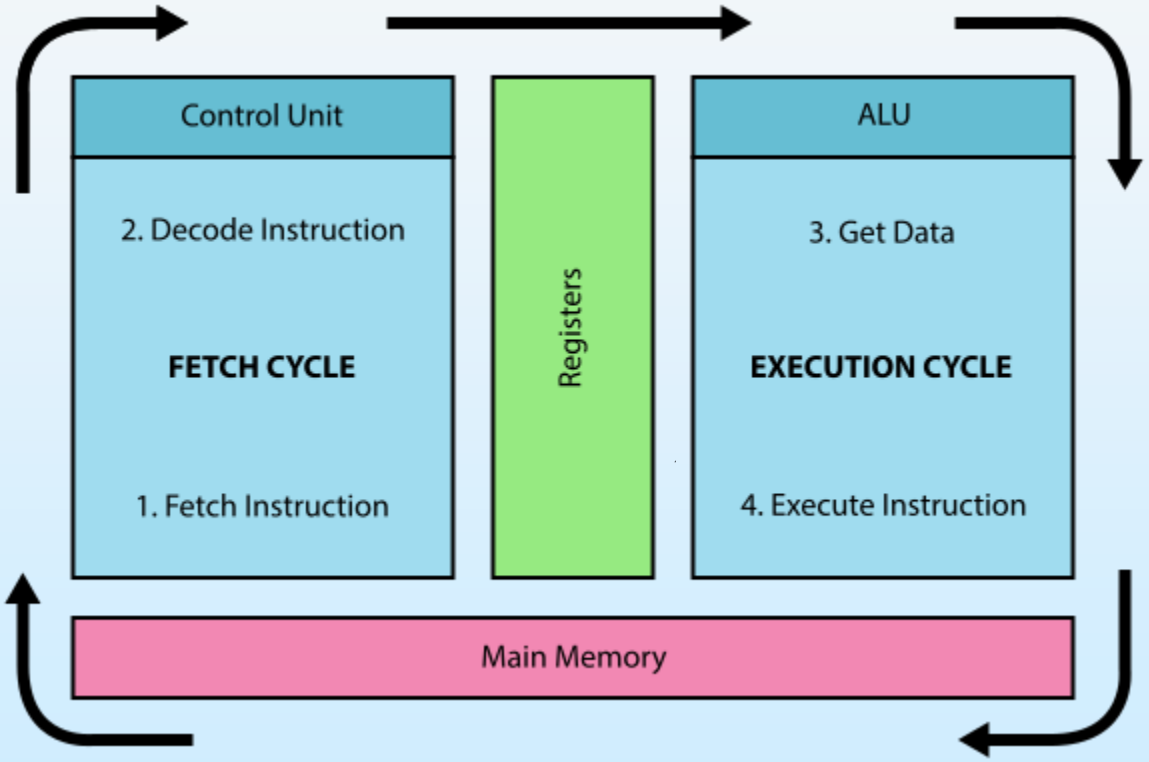


(a) Von Neumann architecture



(b) information flow

Fetch-Execute Cycle



Computer-System Operation

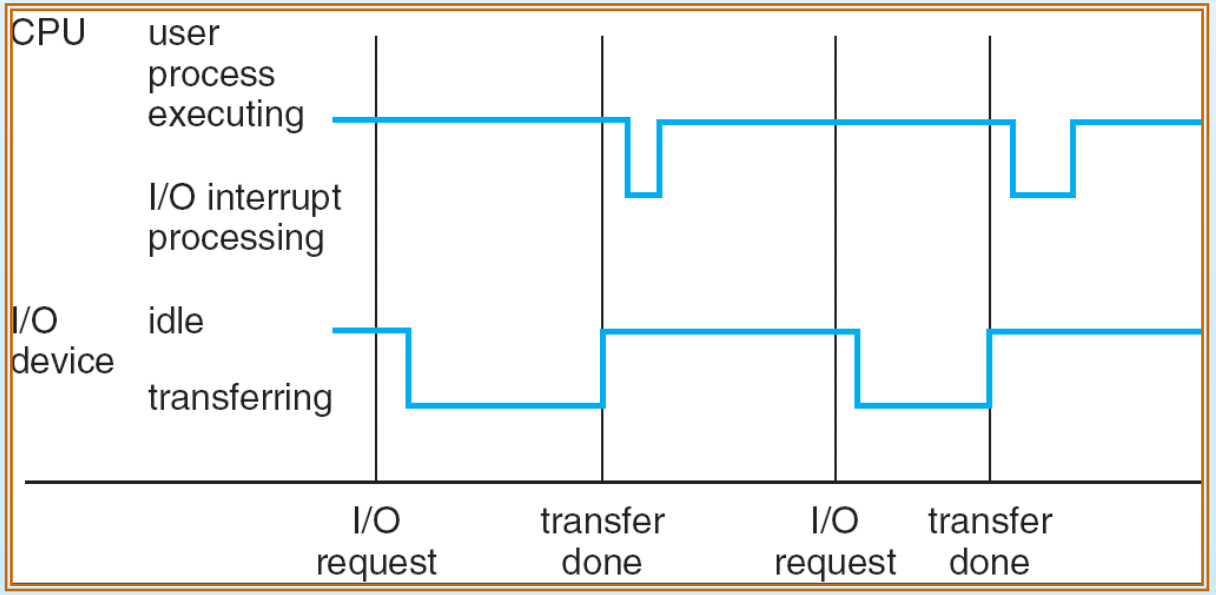
- I/O devices and the CPU can execute concurrently.
- Each device controller is in charge of a particular device type.
- Each device controller has a local buffer.
- CPU moves data from/to main memory to/from local buffers
- I/O is from the device to local buffer of controller.
- Device controller informs CPU that it has finished its operation by causing an *interrupt*.

Common Functions of Interrupts

- Interrupt transfers control to the interrupt service routine generally, through the ***interrupt vector***, which contains the addresses of all the service routines.
- Interrupt architecture must save the address of the interrupted instruction.
- Incoming interrupts are ***disabled*** while another interrupt is being processed to prevent a ***lost interrupt***.
- A ***trap*** is a software-generated interrupt caused either by an error or a user request.
- An operating system is ***interrupt*** driven.

Interrupt Handling

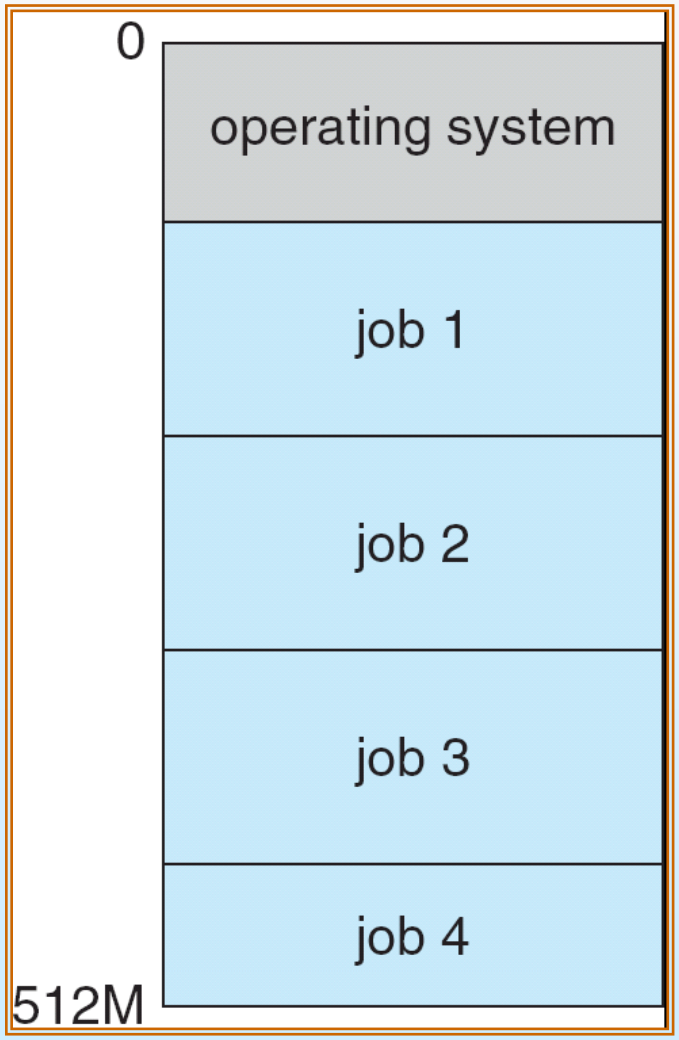
- The operating system preserves the state of the CPU by storing registers and the program counter.
- Determines which type of interrupt has occurred:
- Separate segments of code determine what action should be taken for each type of interrupt



Operating System Structure

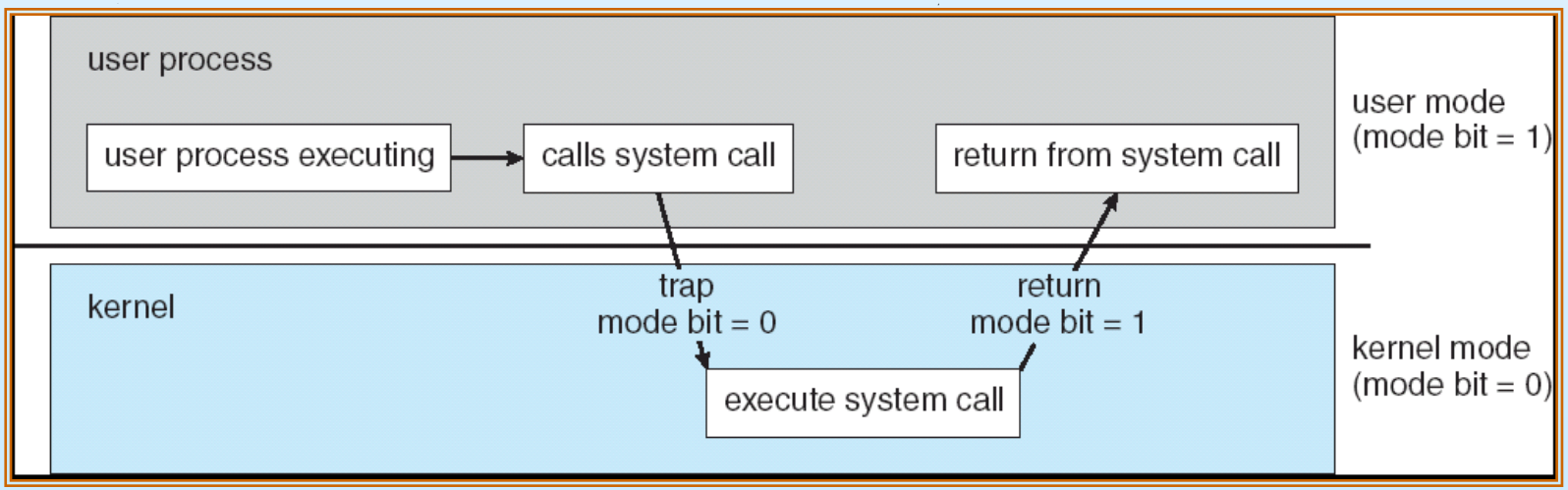
- **Multiprogramming** needed for efficiency
  - Single user cannot keep CPU and I/O devices busy at all times
  - Multiprogramming organizes jobs (code and data) so CPU always has one to execute
  - A subset of total jobs in system is kept in memory
  - One job selected and run via **job scheduling**
  - When it has to wait (for I/O for example), OS switches to another job
- **Timesharing (multitasking)** is logical extension in which CPU switches jobs so frequently that users can interact with each job while it is running, creating **interactive** computing
  - **Response time** should be < 1 second
  - Each user has at least one program executing in memory ⇨**process**
  - If several jobs ready to run at the same time ⇨ **CPU scheduling**
  - If processes don't fit in memory, **swapping** moves them in and out to run
  - **Virtual memory** allows execution of processes not completely in memory

Memory Layout for Multiprogrammed System



Operating-System Dual-Mode Operations

- Problem: processes modifying each other or the operating system
- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
  - **Mode bit** provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as **privileged**, only executable in kernel mode



Timer

- Problem: infinite loop, not calling system services, never returning control to the OS
- Timer to prevent infinite loop / process hogging resources
  - Set interrupt after specific period
  - Operating system decrements counter
  - When counter zero generate an interrupt
  - Set up before scheduling process to regain control or terminate program that exceeds allotted time

Process Management

- A process is a program in execution. It is a unit of work within the system. Program is a ***passive entity***, process is an ***active entity***.
- Process needs resources to accomplish its task
  - CPU, memory, I/O, files
  - Initialization data
- Process termination requires reclaim of any reusable resources
- Single-threaded process has one ***program counter*** specifying location of next instruction to execute
  - Process executes instructions sequentially, one at a time, until completion
- Multi-threaded process has one program counter per thread
- Typically system has many processes, some user, some operating system running concurrently on one or more CPUs
  - Concurrency by multiplexing the CPUs among the processes / threads

Process Management Activities

The operating system is responsible for the following activities in connection with process management:

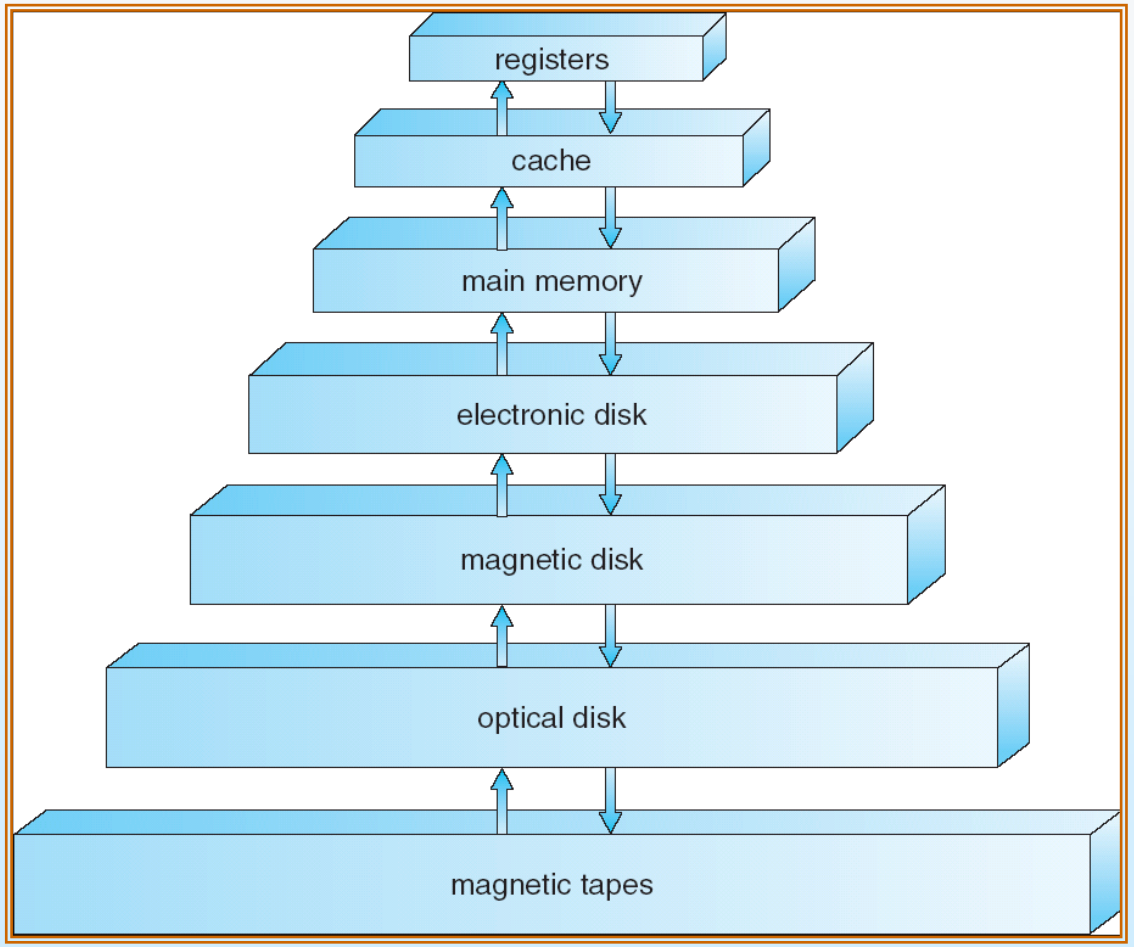
- Creating and deleting both user and system processes
- Suspending and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanisms for deadlock handling

Storage Structure

- Main memory – only large storage media that the CPU can access directly.
- Secondary storage – extension of main memory that provides large nonvolatile storage capacity.
- Magnetic disks – rigid metal or glass platters covered with magnetic recording material
  - Disk surface is logically divided into *tracks*, which are subdivided into *sectors*.
  - The *disk controller* determines the logical interaction between the device and the computer.

Storage Hierarchy

- Storage systems organized in hierarchy.
  - Speed
  - Cost
  - Volatility



Caching

- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storage temporarily
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly from the cache (fast)
  - If not, data copied to cache and used there
- Cache smaller than storage being cached
  - Cache management important design problem
  - Cache size and replacement policy

Memory Management

- All data in memory before and after processing
- All instructions in memory in order to execute
- Memory management determines what is in memory when
  - Optimizing CPU utilization and computer response to users
- Memory management activities
  - Keeping track of which parts of memory are currently being used and by whom
  - Deciding which processes (or parts thereof) and data to move into and out of memory
  - Allocating and deallocating memory space as needed

Storage Management

- OS provides uniform, logical view of information storage
  - Abstracts physical properties to logical storage unit - **file**
  - Each medium is controlled by device (i.e., disk drive, tape drive)
    - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)

- File-System management
  - Files usually organized into directories
  - Access control on most systems to determine who can access what
  - OS activities include
    - Creating and deleting files and directories
    - Primitives to manipulate files and dirs
    - Mapping files onto secondary storage
    - Backup files onto stable (non-volatile) storage media

Mass-Storage Management

- Usually disks used to store data that does not fit in main memory or data that must be kept for a “long” period of time.
- Proper management is of central importance
- Entire speed of computer operation hinges on disk subsystem and its algorithms
- OS activities
  - Free-space management
  - Storage allocation
  - Disk scheduling
- Some storage need not be fast
  - Tertiary storage includes optical storage, magnetic tape
  - Still must be managed
  - Varies between WORM (write-once, read-many-times) and RW (read-write)

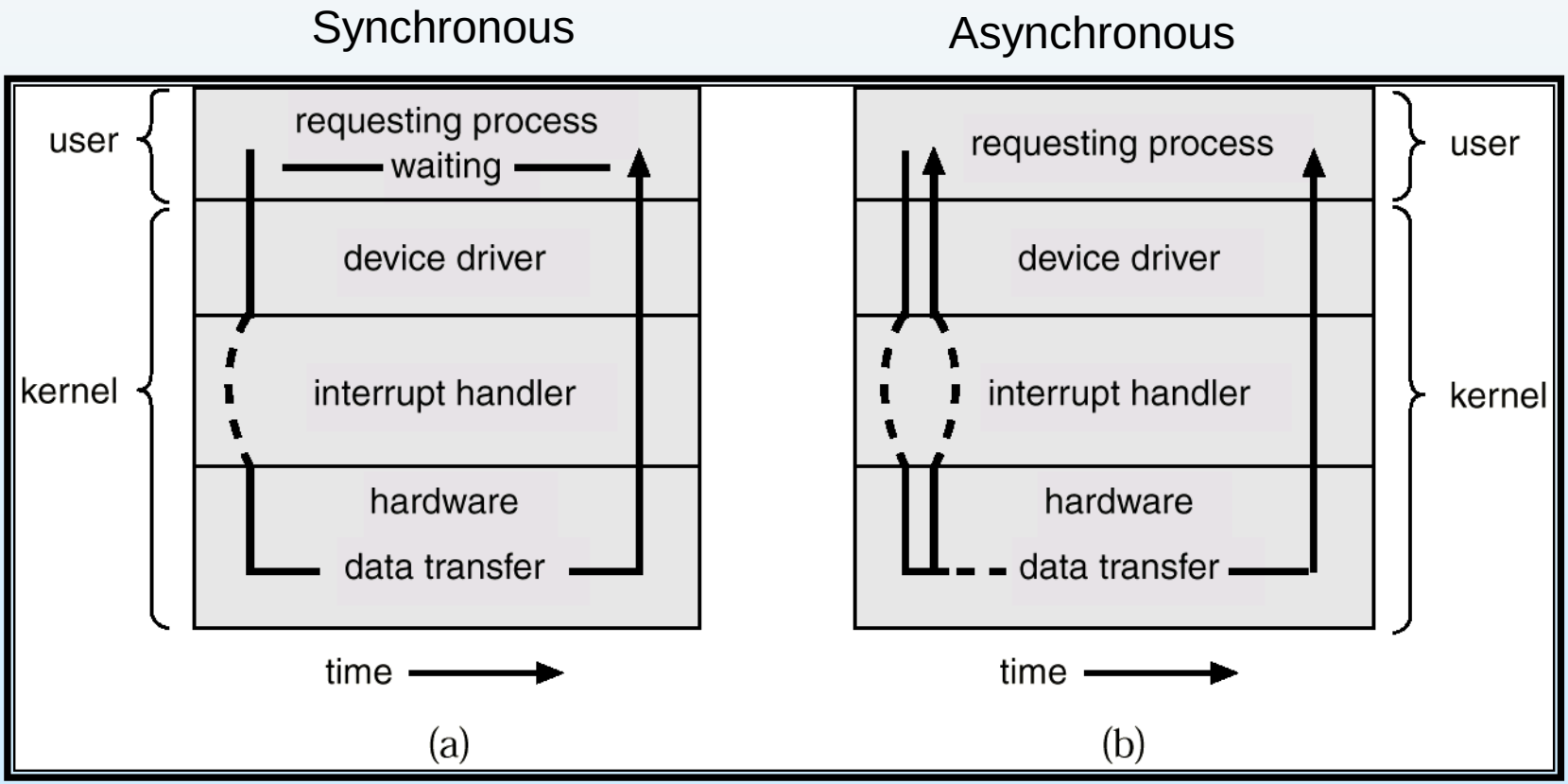
I/O Subsystem

- One purpose of OS is to hide peculiarities of hardware devices from the user
- I/O subsystem responsible for
  - Memory management of I/O including buffering (storing data temporarily while it is being transferred), caching (storing parts of data in faster storage for performance), spooling (the overlapping of output of one job with input of other jobs)
  - General device-driver interface
  - Drivers for specific hardware devices

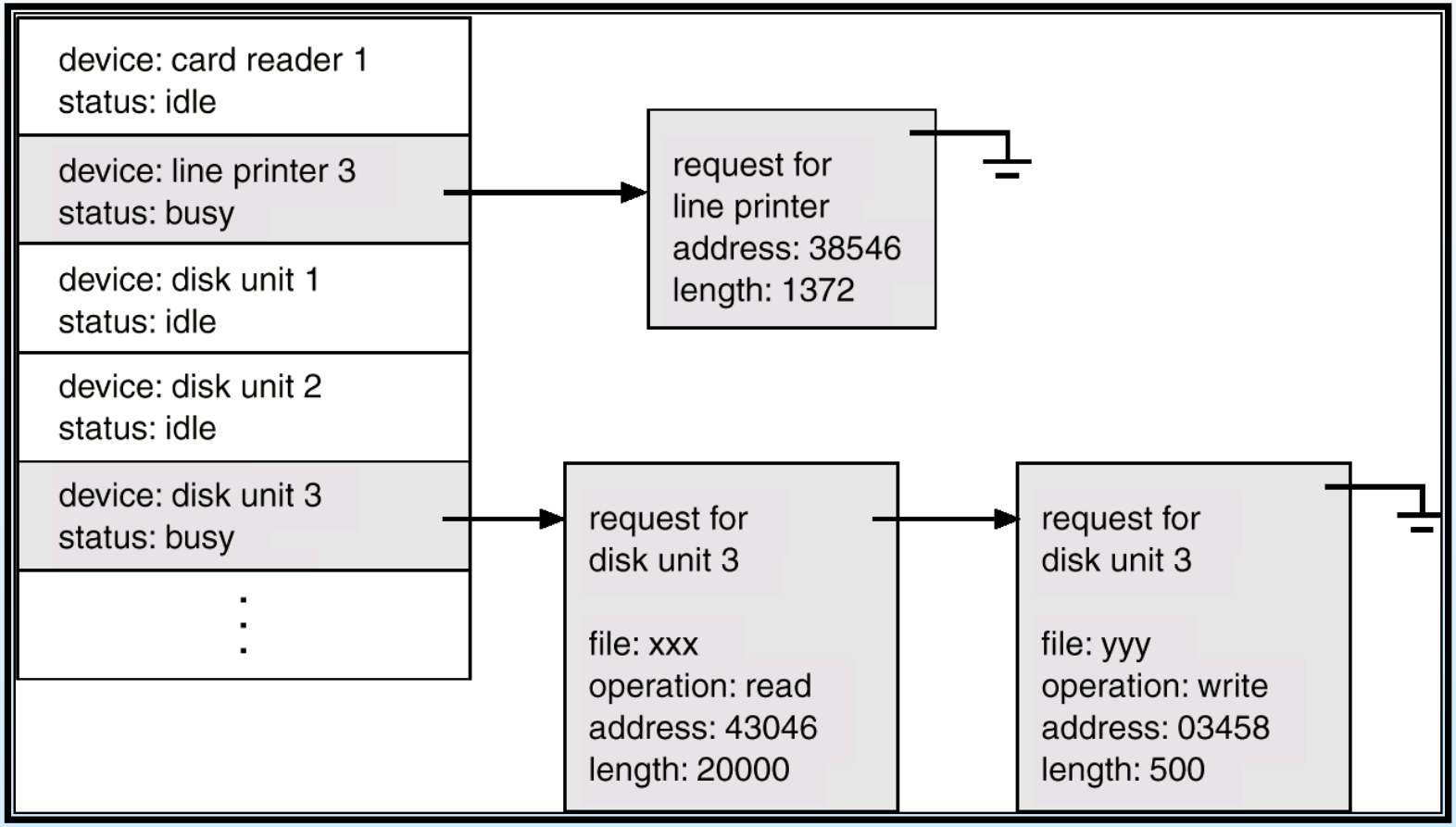
I/O Structure

- After I/O starts, control returns to user program only upon I/O completion.
  - Wait instruction idles the CPU until the next interrupt
  - Wait loop (contention for memory access).
  - At most one I/O request is outstanding at a time, no simultaneous I/O processing.
- After I/O starts, control returns to user program without waiting for I/O completion.
  - *System call* – request to the operating system to allow user to wait for I/O completion.
  - *Device-status table* contains entry for each I/O device indicating its type, address, and state.
  - Operating system indexes into I/O device table to determine device status and to modify table entry to include interrupt.

Two I/O Methods



Device-Status Table



Direct Memory Access Structure

- Used for high-speed I/O devices able to transmit information at close to memory speeds.
- Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention.
- Only one interrupt is generated per block, rather than the one interrupt per byte.

Protection and Security

- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
  - User identities include name and associated number, User ID then associated with all files, processes of that user to determine access control
  - Group identifier allows set of users to be defined and controls managed, then also associated with each process, file
  - **Privilege escalation** allows user to change to effective ID with more rights