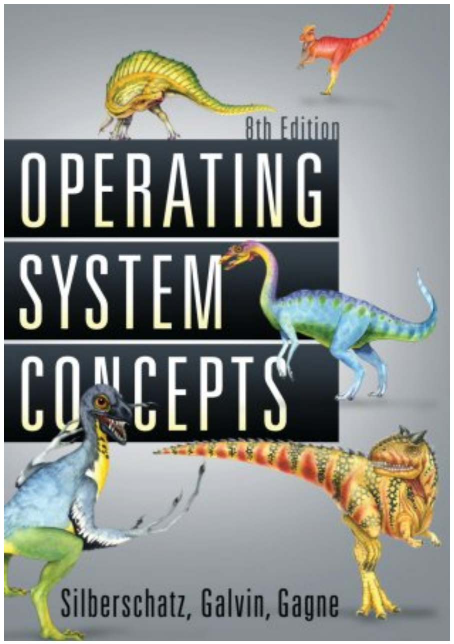
ECE 437 / CS 481 Operating Systems

By
Dr. Edward Nava

Course objectives



- To learn "what an operating system is and how it works"
- To understand OS design principles
- To understand "how OS concepts are implemented"
- To practice systems programming skills
- To learn about OS performance and its evaluation

A. Silberschatz, Peter Baer Galvin and Greg Gagne " Operating System Concepts", 9th edition, ISBN 978-1-118-06333-0, Wiley, Inc, 2013.

Course Details

- Lecture
  - DSH 120
  - Mon/Wed 11:00-12:15pm
- Prerequisites
  - ECE331 (Data structures & C Programming)
  - ECE344L or CS341 (Computer Organization background)
- Office Hours and Location
  - ECE 225C
  - MW 1:30-2:30pm or by emailing for appointment
  - ejnava@unm.edu

Course Outline

- OS overview, Linux & Windows
- Processes & Threads
- CPU scheduling
- Process synchronization
- Deadlocks
- Files and file systems
- I/O devices & management
- Memory management
- Protection & security

Course Grading

- Homework Assignment and Programming Projects: 40%
  - Homework and projects are intended to be individual assignments unless team work is specifically permitted. Evidence of copying will be penalized; all guilty parties will be given a zero.
  - Late programming assignments will be accepted within 3 calendar days but with a 50% PENALTY.
  - No late homework will be accepted; No makeup quizzes;
- Exams 60%
  - Exams will cover material presented in class, programming projects, and homework assignments.
  - Final Exam (optional, comprehensive): Replace min(Exam I, II, III)

Course logistics

- Course GA or PA
  - Mithun Mohan, mithunmohan@unm.edu
- Check course homepage for assignment/grades
  - http://learn.unm.edu
- Tentative schedule is posted at learn.unm.edu and will be updated through the semester
- Tentative exam dates
  - Exam I, Tuesday, Aug 26
  - Exam II, Tuesday, Oct 31
  - Exam III, Thursday, Nov 30
  - Final Exam week of Dec 5

Lecture 1: Introduction

Lecture 1: Introduction

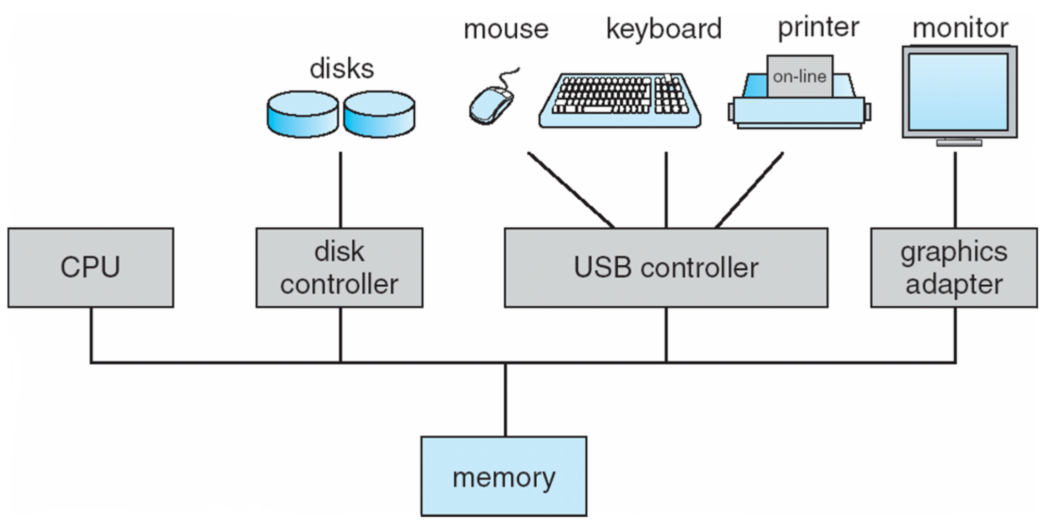
- What Operating Systems Do
- Computer-System Organization

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

Computer System Organization

- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles



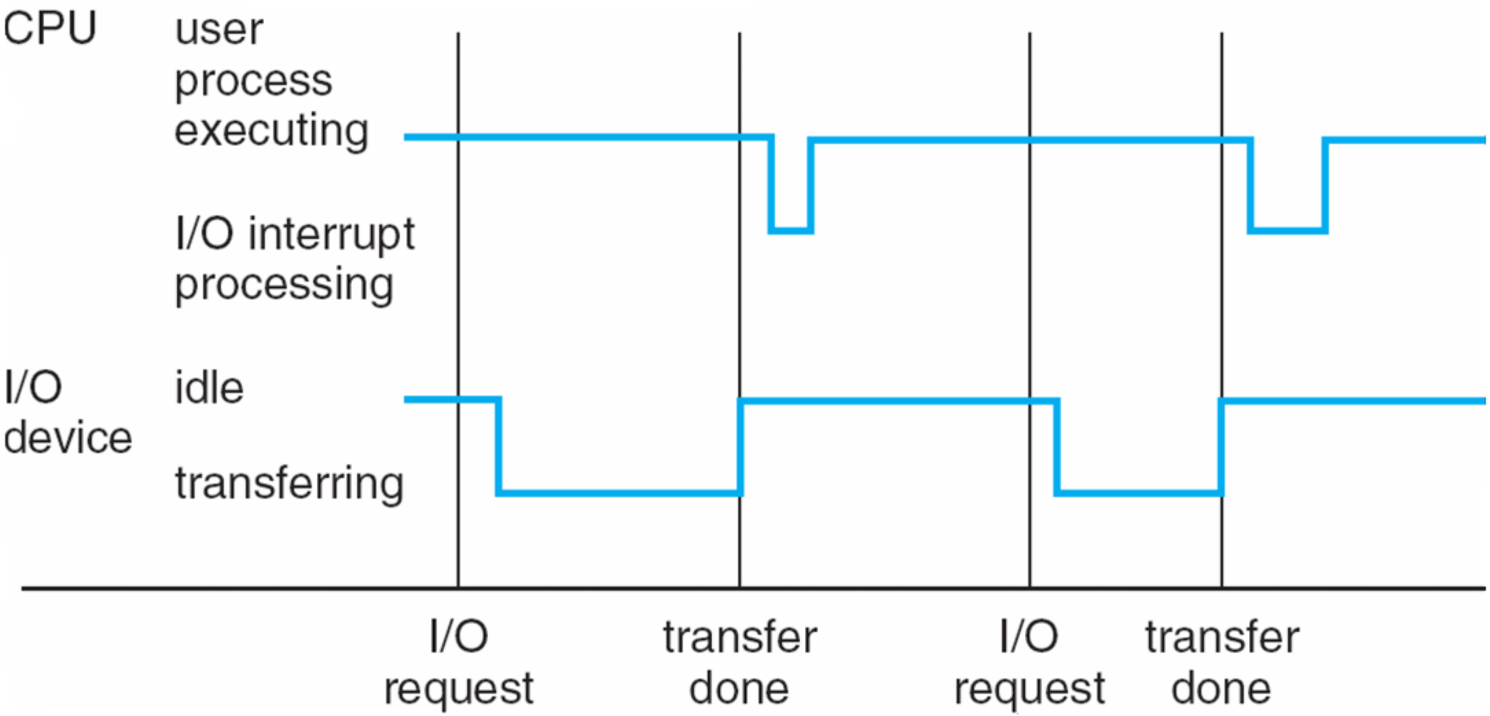
Computer Startup

- **bootstrap program** is loaded at power-up or reboot
  - Typically stored in ROM or EPROM, generally known as **firmware**
  - Initializes all aspects of system
  - Loads operating system kernel and starts execution

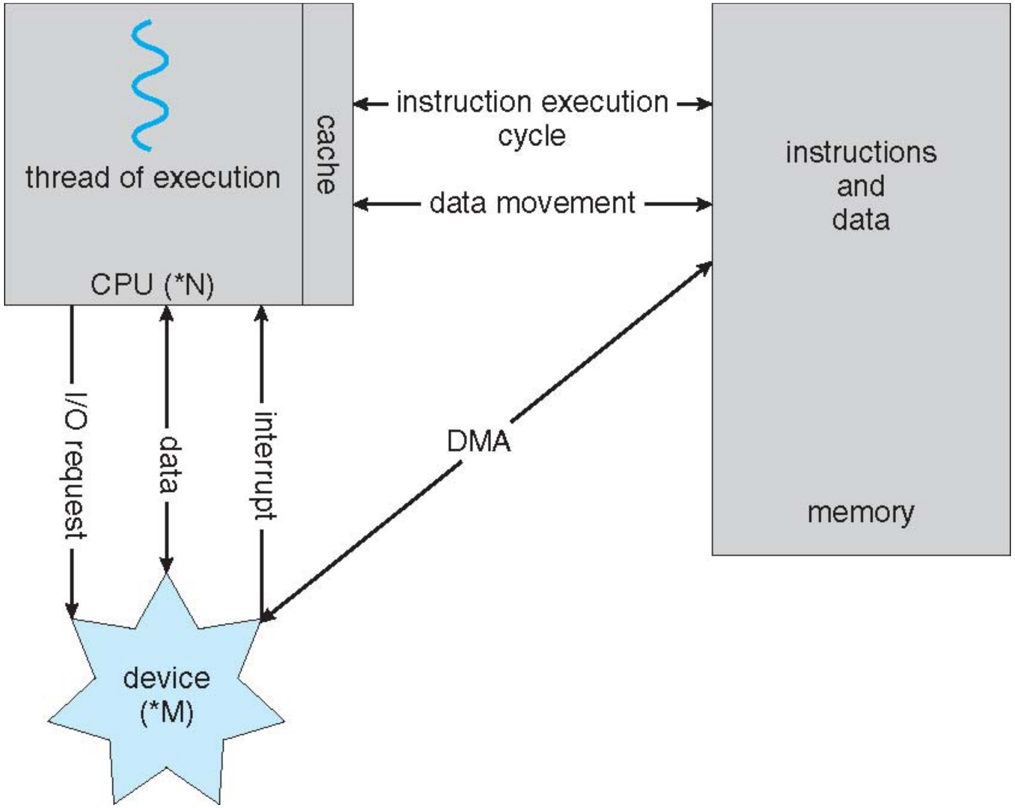
Common Functions of Interrupts

- 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 Timeline



How a Modern Computer Works

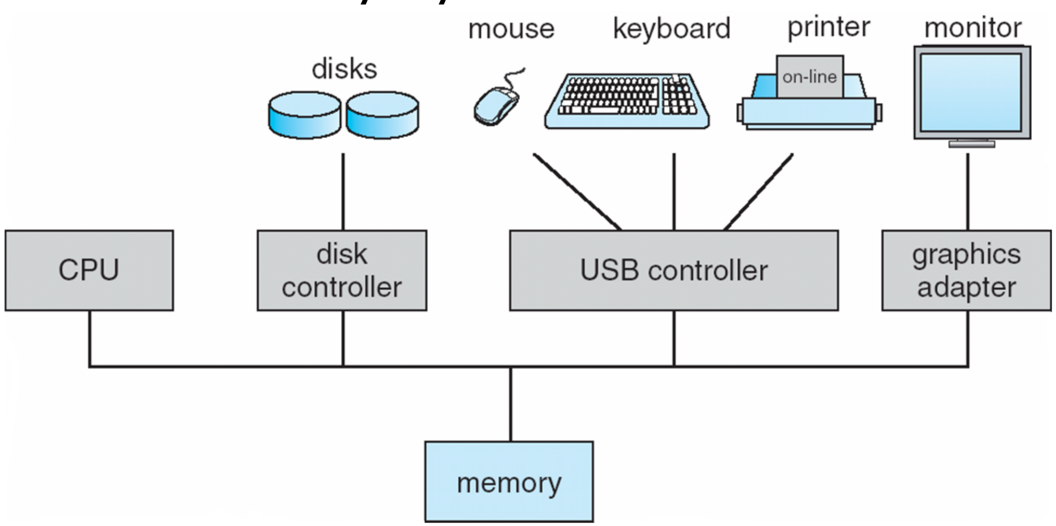


Computer-System Operation

- 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*

Computer System Organization

- Computer-system operation
  - One or more CPUs, device controllers connect through common bus providing access to shared memory
  - Concurrent execution of CPUs and devices competing for memory cycles



Storage Definitions and Notation Review

The basic unit of computer storage is the **bit**. A bit can contain one of two values, 0 and 1. All other storage in a computer is based on collections of bits. Given enough bits, it is amazing how many things a computer can represent: numbers, letters, images, movies, sounds, documents, and programs, to name a few. A **byte** is 8 bits, and on most computers it is the smallest convenient chunk of storage. For example, most computers don't have an instruction to move a bit but do have one to move a byte. A less common term is **word**, which is a given computer architecture's native unit of data. A word is made up of one or more bytes. For example, a computer that has 64-bit registers and 64-bit memory addressing typically has 64-bit (8-byte) words. A computer executes many operations in its native word size rather than a byte at a time.

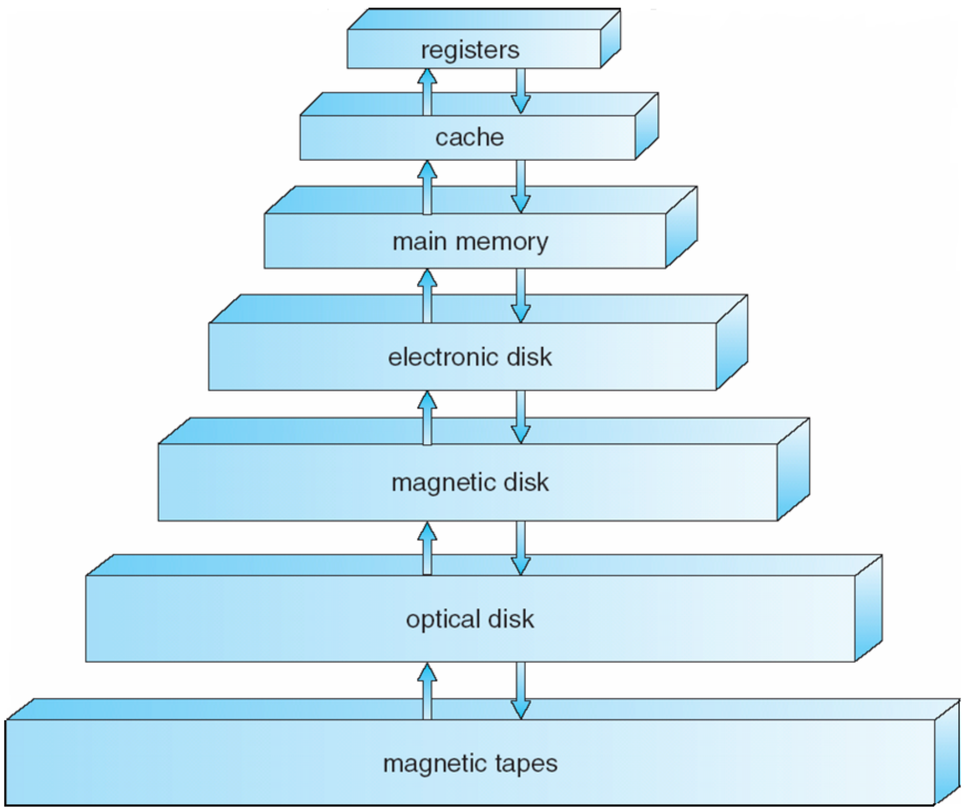
Computer storage, along with most computer throughput, is generally measured and manipulated in bytes and collections of bytes.
A **kilobyte**, or **KB**, is 1,024 bytes
a **megabyte**, or **MB**, is $1{,}024^2$ bytes
a **gigabyte**, or **GB**, is $1{,}024^3$ bytes
a **terabyte**, or **TB**, is $1{,}024^4$ bytes
a **petabyte**, or **PB**, is $1{,}024^5$ bytes

Computer manufacturers often round off these numbers and say that a megabyte is 1 million bytes and a gigabyte is 1 billion bytes. Networking measurements are an exception to this general rule; they are given in bits (because networks move data a bit at a time).

Storage Structure

- Main memory – only large storage media that the CPU can access directly
  - **Random access**
  - Typically **volatile**
- Secondary storage – extension of main memory that provides large **nonvolatile** storage capacity
- Hard 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
- **Solid-state disks** – faster than hard disks, nonvolatile
  - Various technologies
  - Becoming more popular

Storage-Device Hierarchy



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

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