CPE 460 Operating System Design

Lecture 1: Course Overview

Ahmed Tamrawi

February 8, 2017

Important Equations for the Class

Erwin Schrödinger (1887-1961)

Schrödinger Equation:

$$i\hbar\frac{\partial}{\partial t}\Psi(x,t) = -\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2}\Psi(x,t) + V(x)\Psi(x,t)$$

Schrödinger equation does for a quantum-mechanical particle what Newton's Second Law does for a classical particle. The Solution to Schrödinger equation to determine how a particle evolves in time, just as we use Newton's Second Law to solve for future position and momentum of a classical particle.

Further Reading:

1. https://simple.wikipedia.org/wiki/Schr%C3%B6dinger_equation
2. https://www.quora.com/What-is-the-Schr%C3%B6dinger-wave-equation-and-what-are-its-applications

Werner Heisenberg (1901-1976)

Heisenberg Uncertainty Principal:

$$\Delta x \Delta p \geq \frac{h}{4\pi} = \frac{\hbar}{2}$$

In the world of very small particles, one cannot measure any property of a particle without interacting with it in some way. This introduces an unavoidable uncertainty into the result. Thus, One can never measure all the properties exactly.

The more accurately you know the position (the smaller Δx is), the less accurately you know the momentum (the larger Δp is); and vice versa

Further Reading:

https://en.wikipedia.org/wiki/Uncertainty_principle

B.Eng. Computer Engineering
(Class of 2007)

IOWA STATE UNIVERSITY

M.Sc. Computer Engineering
(Class of 2011)

IOWA STATE UNIVERSITY

Ph.D.. Computer Engineering
(Class of 2016)

Secure Programming

Static Program Analysis

Data & Pattern Mining

Software Analysis & Security

Bug finding and Malware detection

Build System Analysis

Abstractions and Symbolic Evaluations

Quantum Physics

Biology

Astronomy


i'm not on facebook®
WhatsApp

YOU

- *Name*
- *Year in undergraduate program.*
- *Something about you*
  - *Food you like.*
  - *Programming languages you used.*
  - *Open source projects you contributed to.*
- *What do you think of this course?*
- *What are your goals after graduation?*

Syllabus

Goal of the Class

Google

Cool Computing Stuff

Cool Magic

Improve our understanding of *how* computers work to **minimize** the magic

My Goals for Lectures?

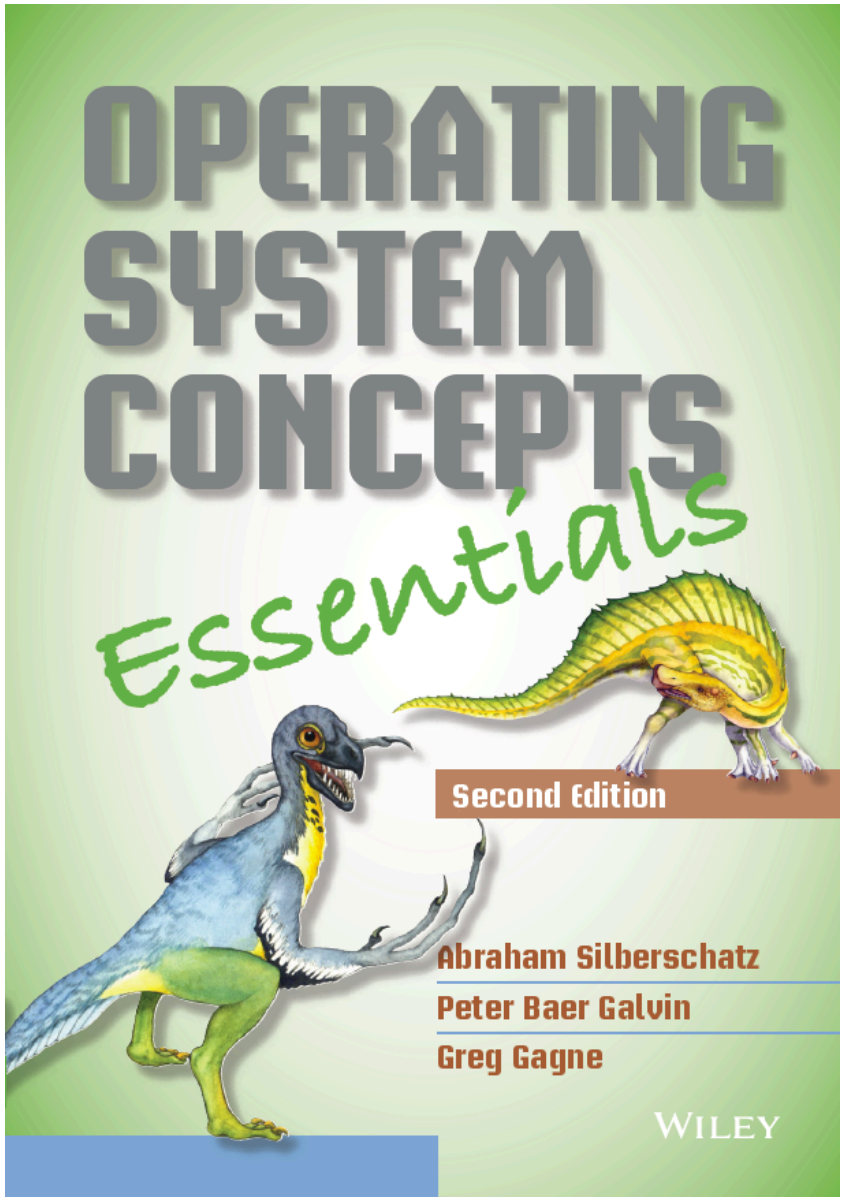
Convey some complex technical ideas	Lectures are *horrible* medium for learning complex ideas, many resource are available online.
Teach you what you need to know to do the labs, assignments, and the project	The point of labs, homework, and project is to teach you things I want you to learn in the class
Avoid being fired	Avoid being fired
Keep most of you awake for 90 minutes	You probably should be getting more sleep
Get you to laugh at dumb jokes	Gabriel Iglesias is more funnier (check him out)

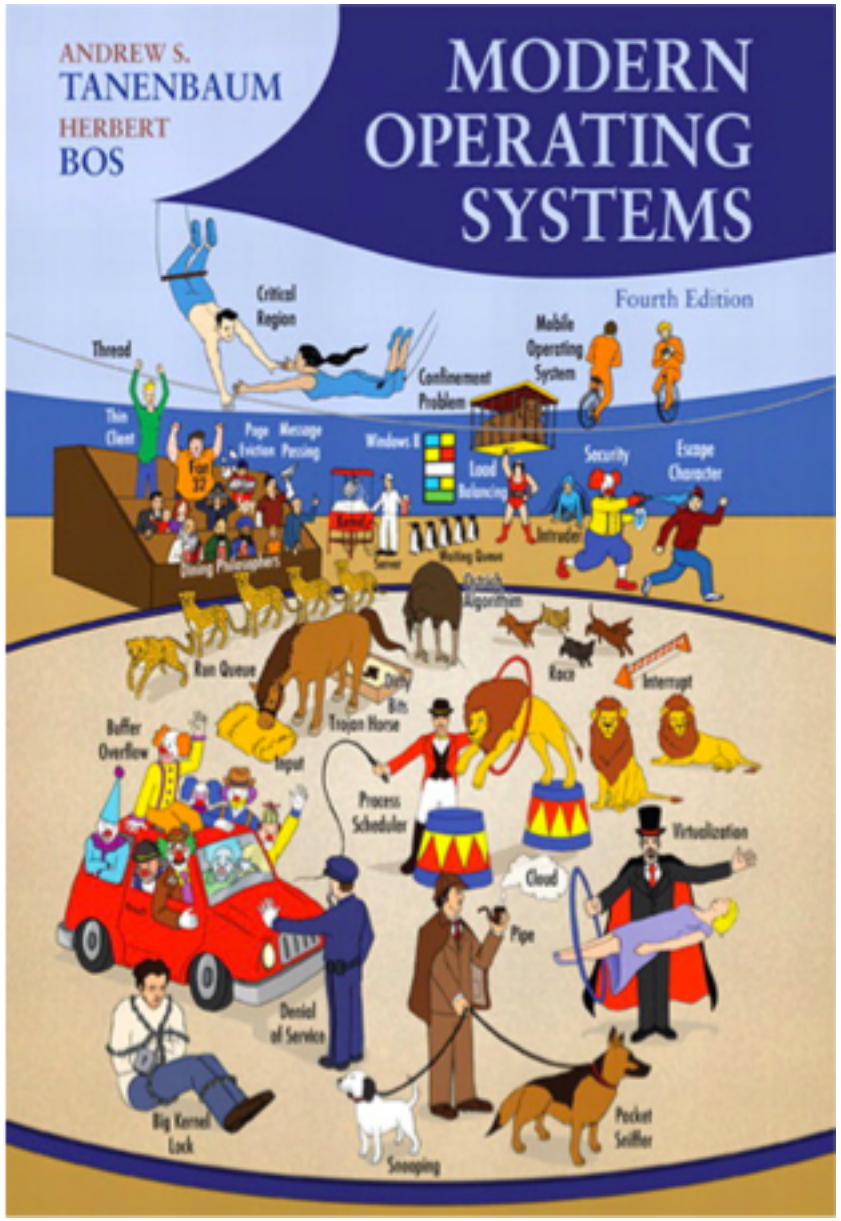
My **Real Goal** for Lectures

Provide **context** and **meaning** for the things you have or will later **learn on your own**

An **operating system** is a program that manages a computer's hardware. It also provides a basis for application programs and acts as an intermediary between the computer user and the computer hardware. An amazing aspect of operating systems is how they vary in accomplishing these tasks. Mainframe operating systems are designed primarily to optimize utilization of hardware. Personal computer (PC) operating systems support complex games, business applications, and everything in between. Operating systems for mobile computers provide an environment in which a user can easily interface with the computer to execute programs. Thus, some operating systems are designed to be *convenient*, others to be *efficient*, and others to be some combination of the two.

A more common definition, and the one that we usually follow, is that the operating system is the one program running at all times on the computer—usually called the **kernel**. (Along with the kernel, there are two other types of programs: **system programs**, which are associated with the operating system but are not necessarily part of the kernel, and application programs, which include all programs not associated with the operation of the system.)





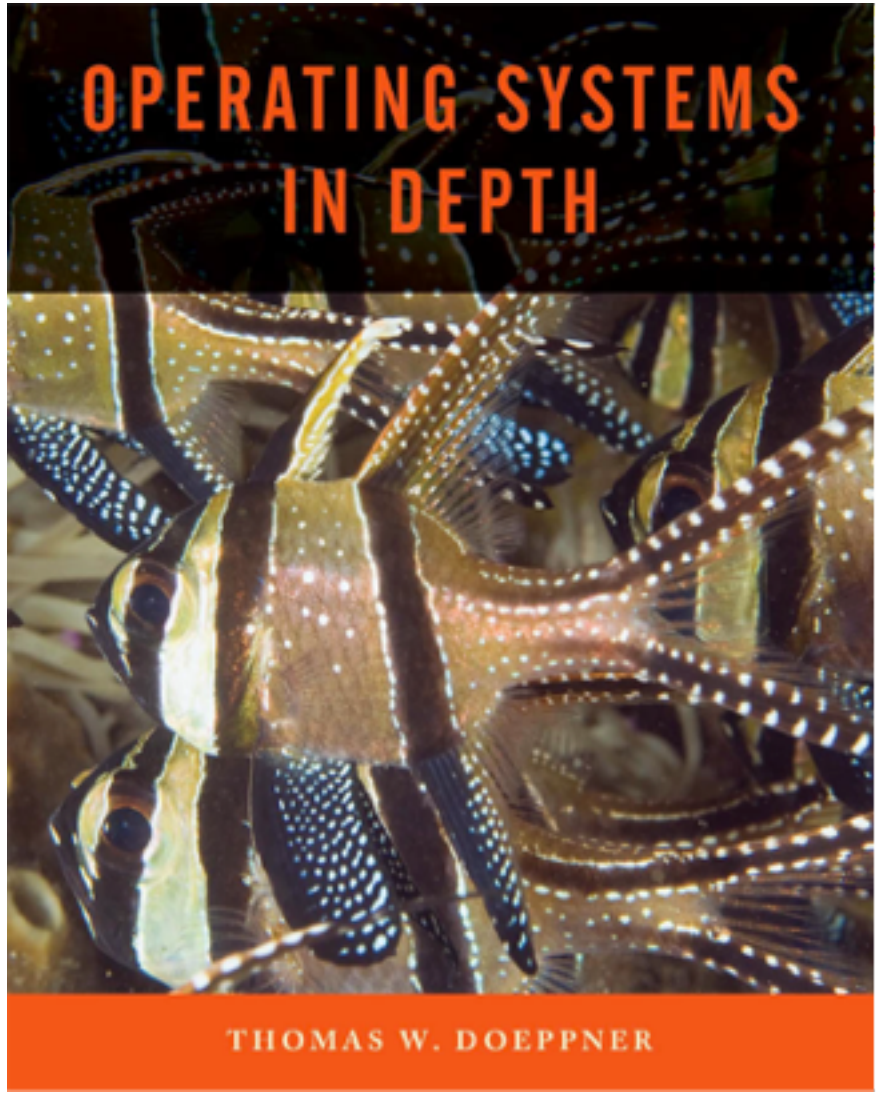
1.1 WHAT IS AN OPERATING SYSTEM?

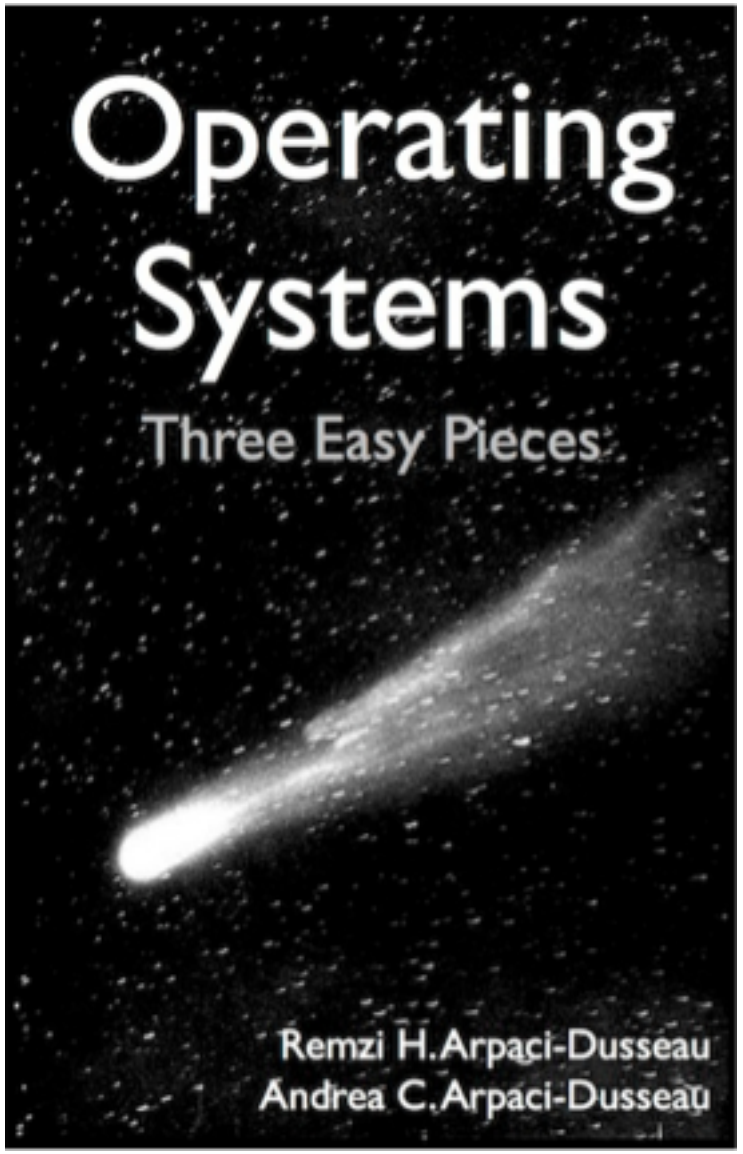
It is hard to pin down what an operating system is other than saying it is the software that runs in kernel mode—and even that is not always true. Part of the problem is that operating systems perform two essentially unrelated functions: providing application programmers (and application programs, naturally) a clean abstract set of resources instead of the messy hardware ones and managing these hardware resources. Depending on who is doing the talking, you might hear mostly about one function or the other. Let us now look at both.

1.1 OPERATING SYSTEMS

What's an operating system? You might say it's what's between you and the hardware, but that would cover pretty much all software. So let's say it's the software that sits between your software and the hardware. But does that mean that the library you picked up from some web site is part of the operating system? We probably want our operating-system definition to be a bit less inclusive. So, let's say that it's that software that almost everything else depends upon. This is still vague, but then the term is used in a rather nebulous manner throughout the industry.

Perhaps we can do better by describing what an operating system is actually supposed to do. From a programmer's point of view, operating systems provide useful abstractions of the underlying hardware facilities. Since many programs can use these facilities at once, the operating system is also responsible for managing how these facilities are shared.





There is a body of software, in fact, that is responsible for making it easy to run programs (even allowing you to seemingly run many at the same time), allowing programs to share memory, enabling programs to interact with devices, and other fun stuff like that. That body of software is called the **operating system** (**OS**)[3], as it is in charge of making sure the system operates correctly and efficiently in an easy-to-use manner.

Do we like any of these definitions?

No universally accepted definition

I know that I like Mansaf!


(Petra)

Petitions

House of Representative
(*Parliament*)

Resources & Commitments

Manage Resources

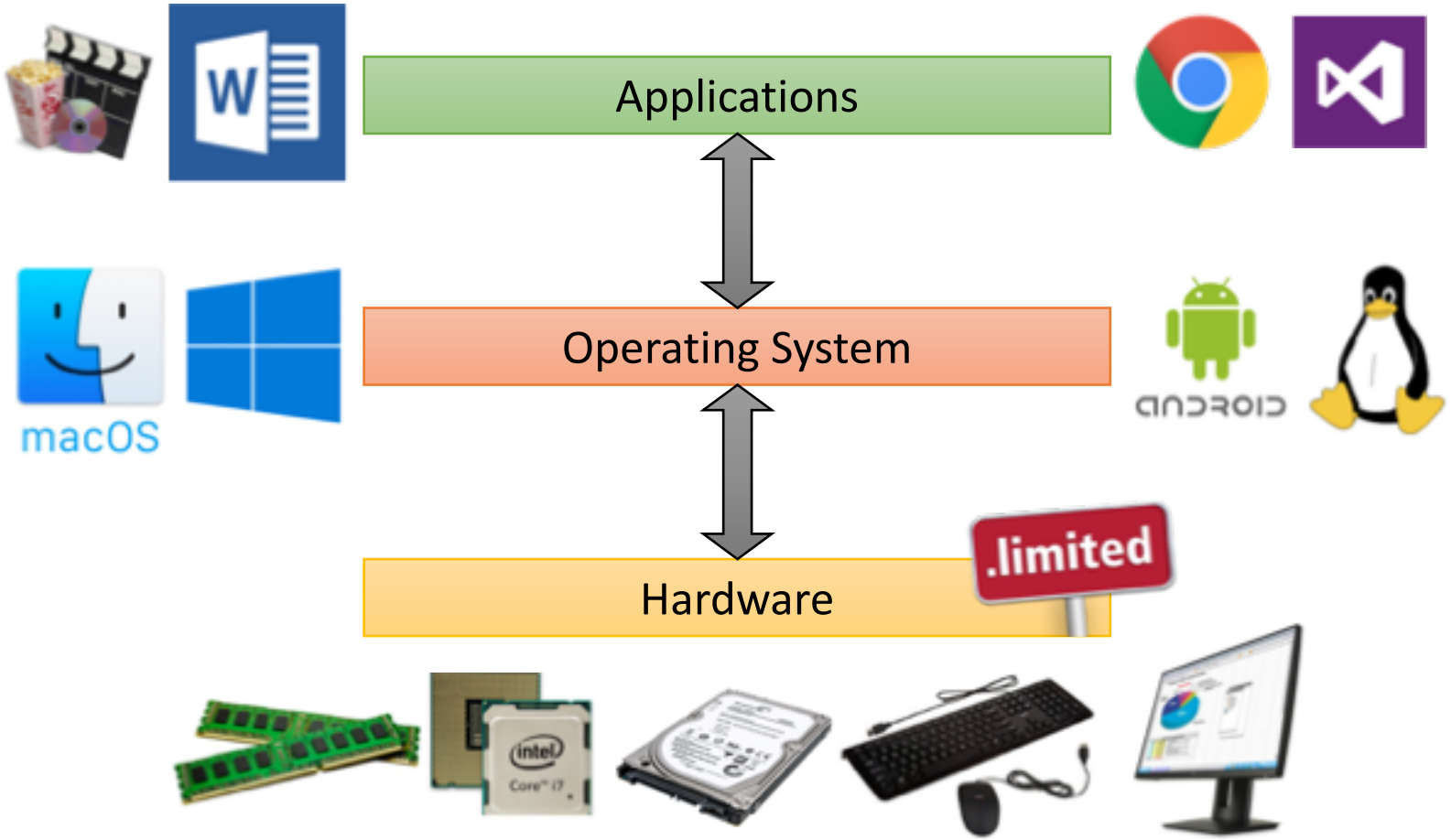
Provide Abstractions

Petitions

House of Representative
(*Parliament*)

Resources & Commitments

Realistic View of Operating System



CPE 460 OS Definition

An **operating system** is a program that
manages resources and ***provide abstractions***

Main Ideas in CPE 460

Manage Resources

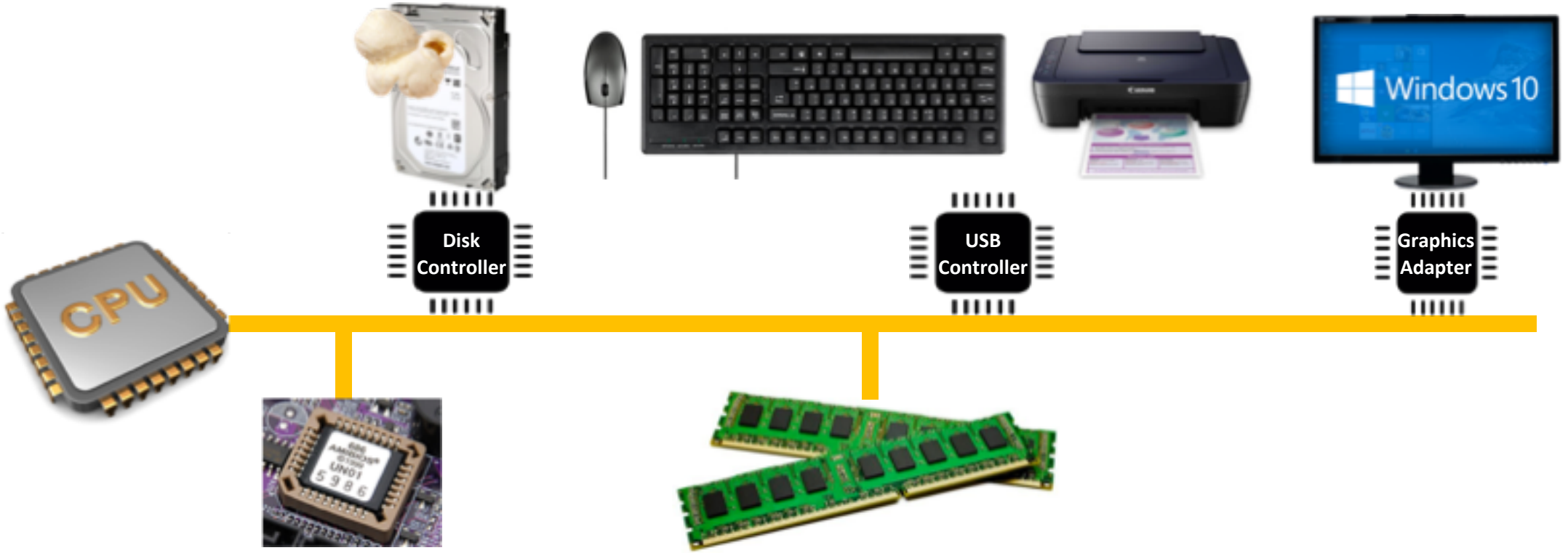
How do you *share* **processors**, **memory**, and **hardware devices** among programs?

Provide Abstractions

How do you provide programs with **clean** and **easy to use** interfaces to resources, without sacrificing (too much) **efficiency and flexibility?**

Does it have an Operating System?

CPE 460 Computer System



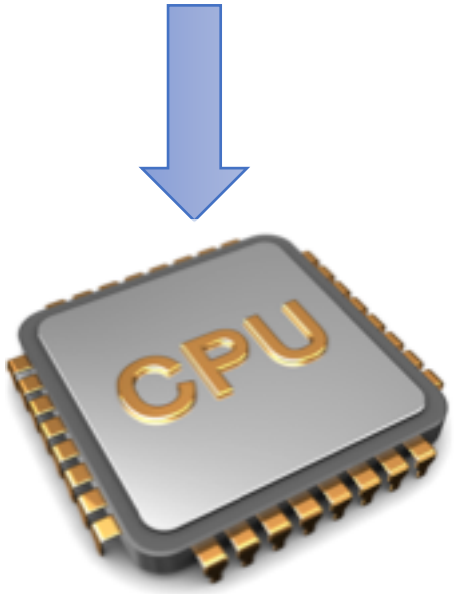
What happens at Computer Startup?


4GIFs.com

Finds itself in Real Mode

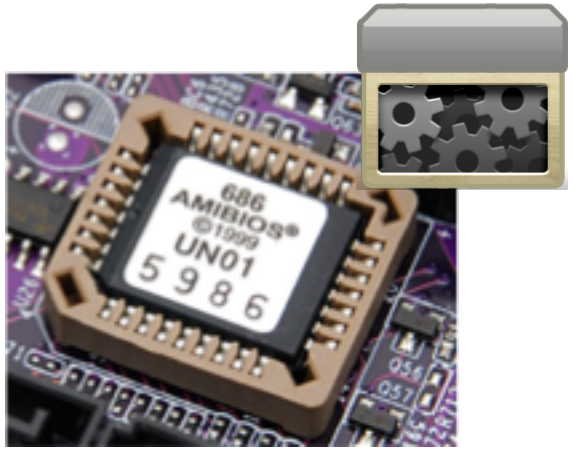
Power On Self Test

Executes the code at address 0xFFFF0 which corresponds to BIOS

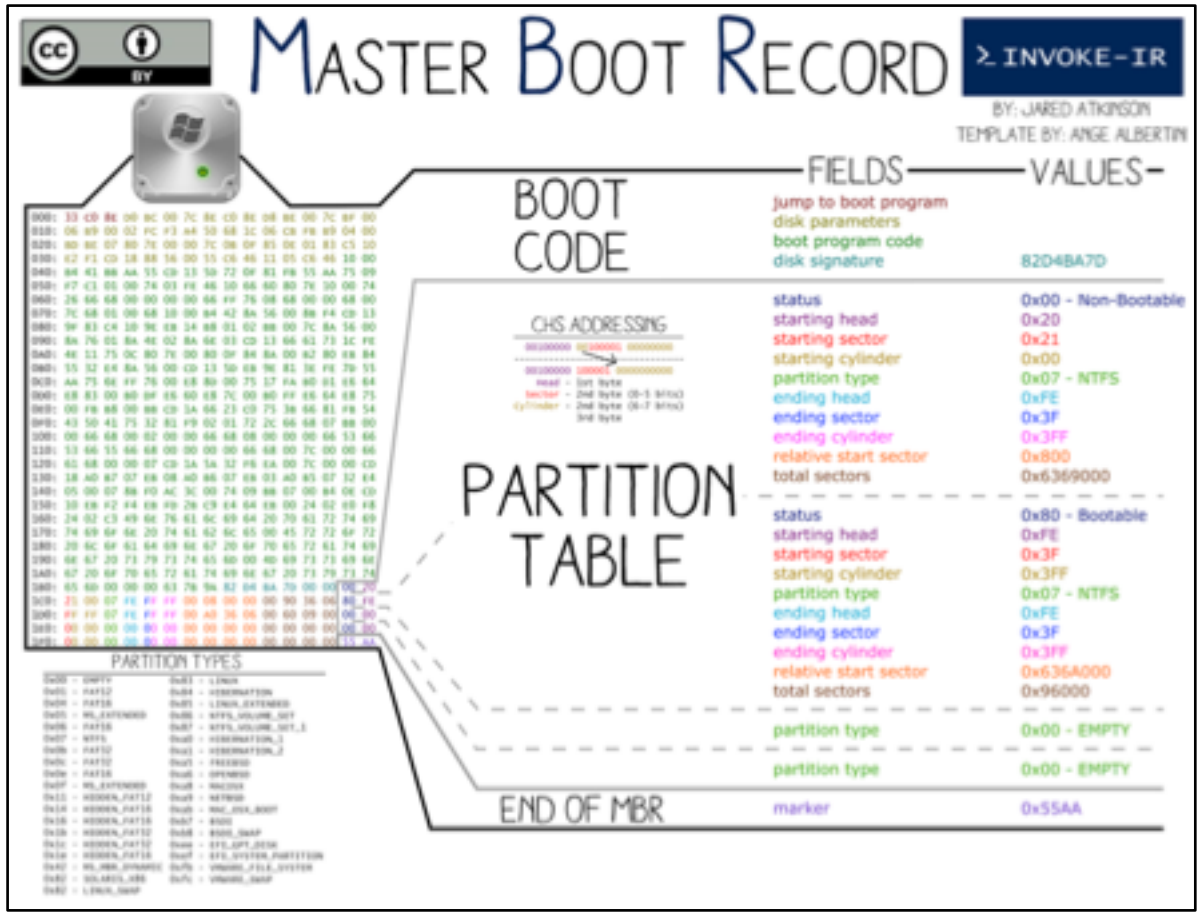


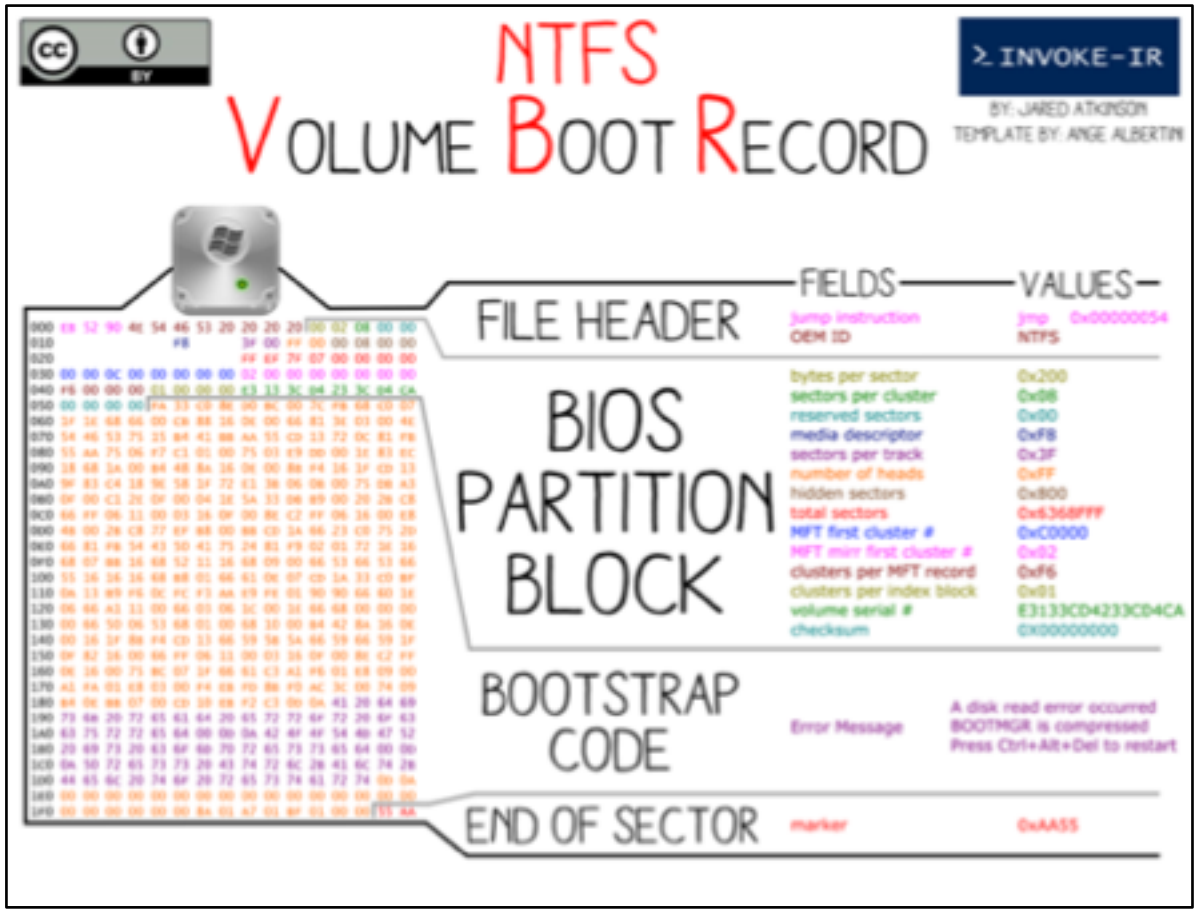
- Finds itself in Real Mode
- Power On Self Test
- Executes the code at address 0xFFFF0 which corresponds to BIOS

Bootstrap Program



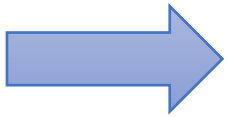
- Autoprobing I/O ports
- Looks for bootloader in Boot Device
- It loads the first sector of a bootable device at 0x7C00 and jumps to it. Then it executes the MBR bootloader located in the first sector of a bootable disk (/dev/hda or /dev/sda)

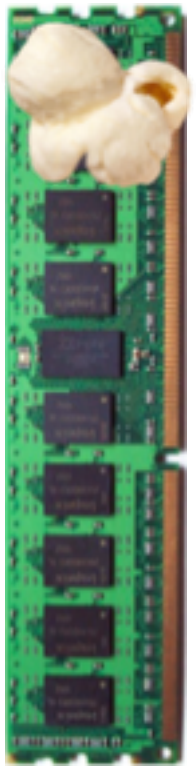




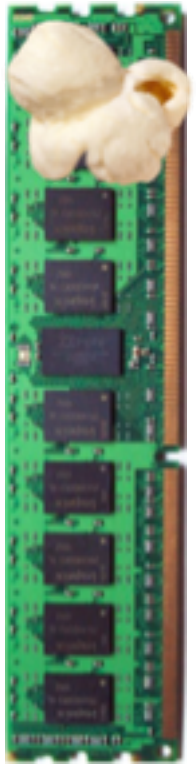
http://www.invoke-ir.com/2015/05/ontheforensictrail-part2.html

Any program to run **must** be loaded in memory

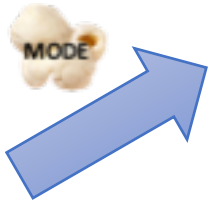




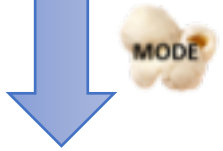
The kernel is decompressed from its image and its loaded into memory



The kernel is decompressed from its image and its loaded into memory



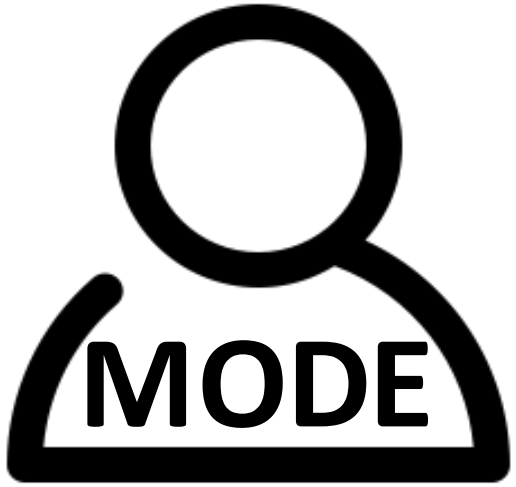
Autoprobing I/O ports

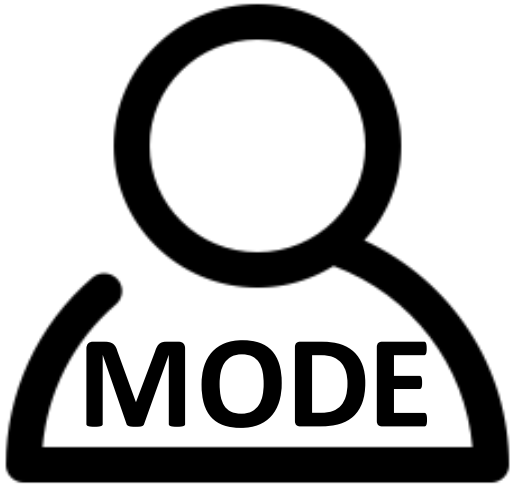


init process

System Processes

System Daemons


MODE



Wait for Event to Occur


انا مش فاهم حاجة خالص

What happens when you move the cursor?

Mouse sends out pulses, one pulse for every 1000th of an inch or so

What happens when you move the cursor?

Mouse sends out pulses, one pulse for every 1000th of an inch or so

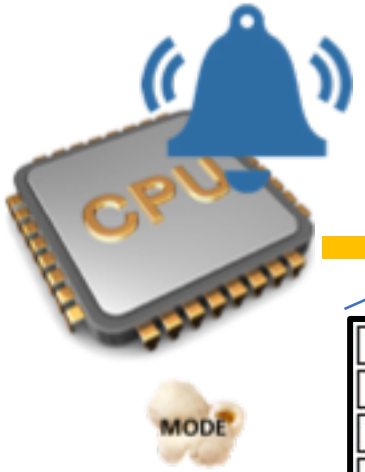
The pulses are received through a USB packet or through an old serial line

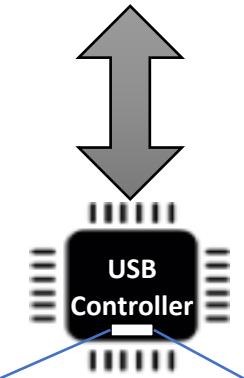
What happens when you move the cursor?

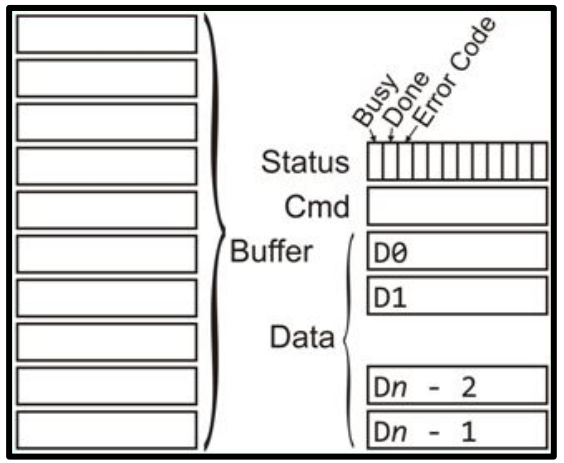
Hardware Interrupt

Mouse sends out pulses, one pulse for every 1000th of an inch or so

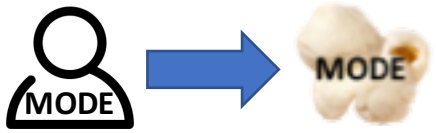
The pulses are received through a USB packet or through an old serial line







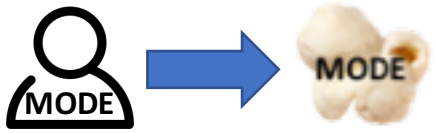
What happens when CPU is interrupted?

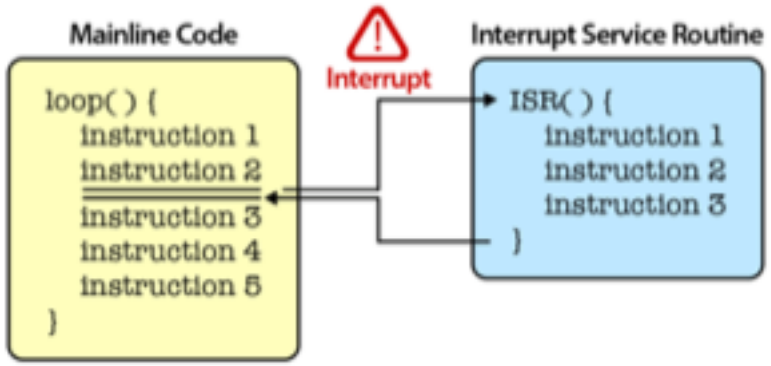


CPU preserves the current state of the CPU by storing registers and the program counter

Interrupt transfers control to the interrupt service routine generally, through the **interrupt vector**

What happens when CPU is interrupted?





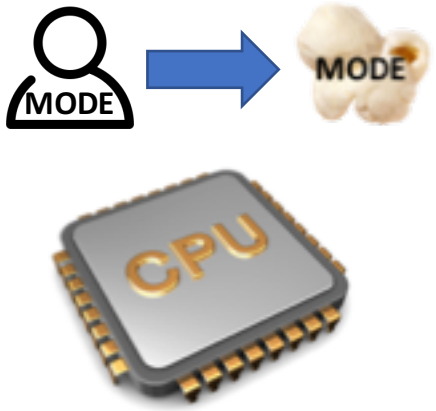
CPU preserves the current state of the CPU by storing registers and the program counter

Separate segments of code determine what action should be taken for each type of interrupt

Interrupt transfers control to the interrupt service routine generally, through the **interrupt vector**

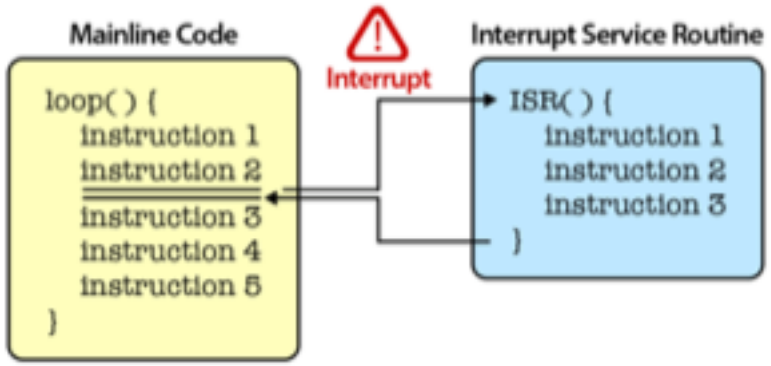
Reads the interrupt and realizes it's from the mouse, and calls the proper ISR which calls the mouse driver.

What happens when CPU is interrupted?



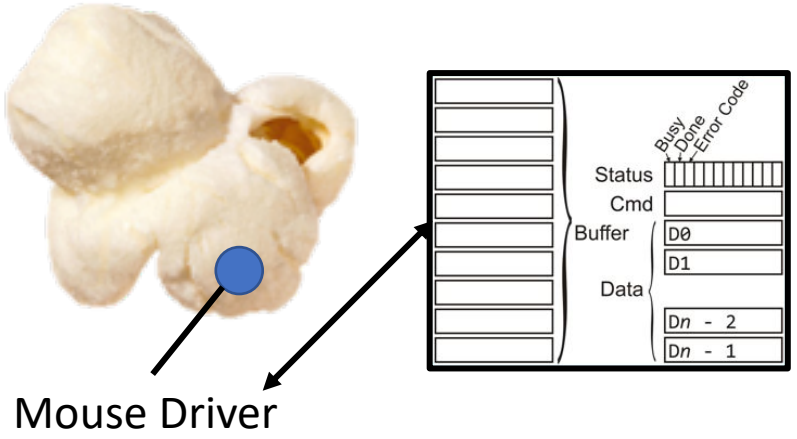
CPU preserves the current state of the CPU by storing registers and the program counter

Interrupt transfers control to the interrupt service routine generally, through the **interrupt vector**

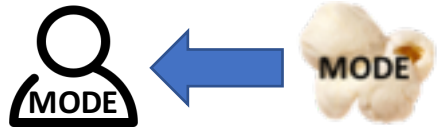


Separate segments of code determine what action should be taken for each type of interrupt

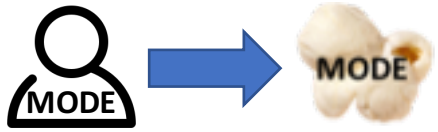
Reads the interrupt and realizes it's from the mouse, and calls the proper ISR which calls the mouse driver.



Mouse driver adds the x and y increments to its current cursor position and return the result to OS



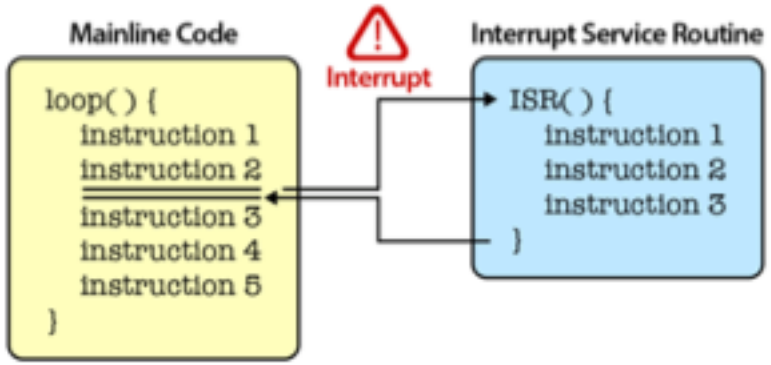
How to notify Monitor of cursor movement?



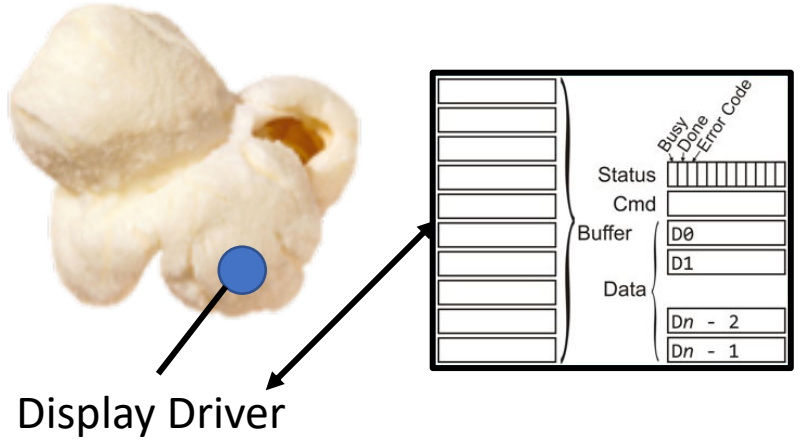
Software Interrupt (Trap)

OS gets interrupted through a **system call** to update the screen

OS preserves the current state of the CPU by storing registers and the program counter



Reads the interrupt and realizes it's from OS to monitor. It calls the display driver with the updated screen

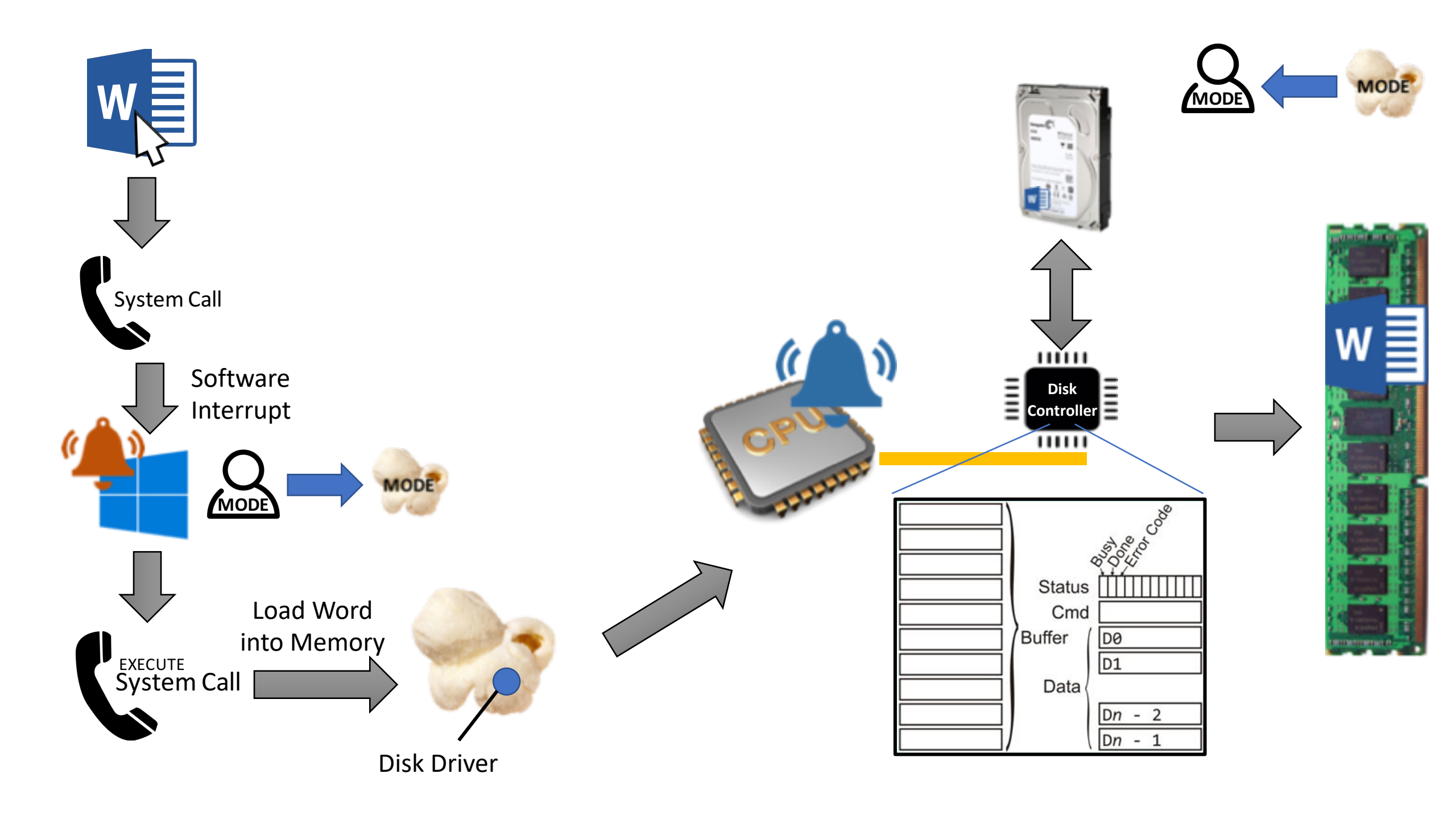


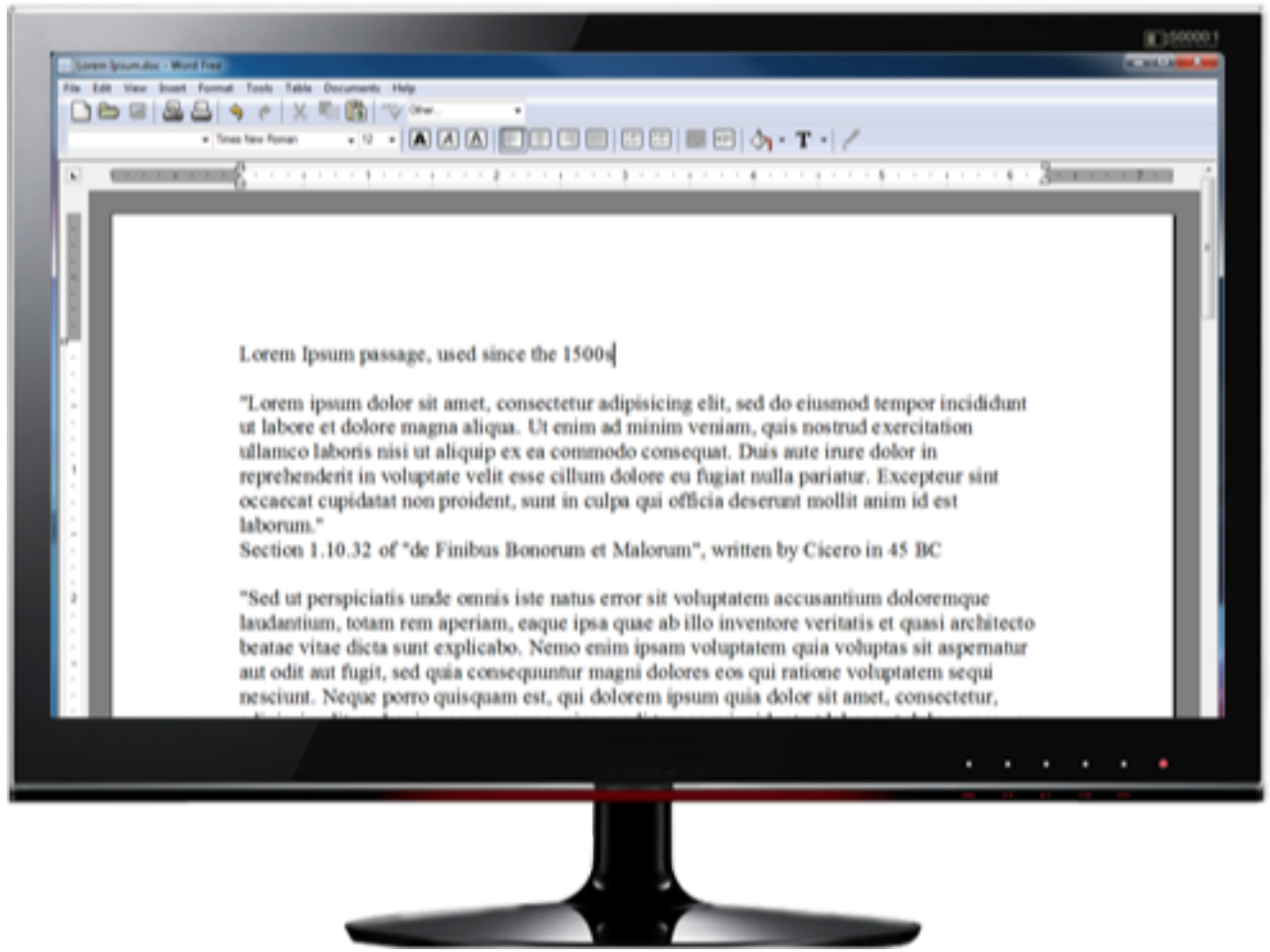
Monitor device drivers sets the proper registers and buffer data in the graphics adapter

Software Interrupt (Trap)

MODE

Any program to run **must** be loaded in memory


System Call
Software Interrupt
MODE
MODE
EXECUTE System Call
Load Word into Memory
Disk Driver
CPU
Disk Controller
Busy
Done
Error Code
Status
Cmd
Buffer
D0
D1
Data
Dn - 2
Dn - 1
MODE
MODE


Lorem Ipsum passage, used since the 1500s
"Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."
Section 1.10.32 of "de Finibus Bonorum et Malorum", written by Cicero in 45 BC
"Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur,

An operating system is interrupt driven


انا مش فاهم حاجة خالص

As long as their processes fit in memory, we do not have a memory problem

Each process needs resources to accomplish its task: CPU, memory, I/O, files, etc.

Process termination requires reclaim of any reusable resources

Typically system has many processes running concurrently, how this is achieved?

Many Processes

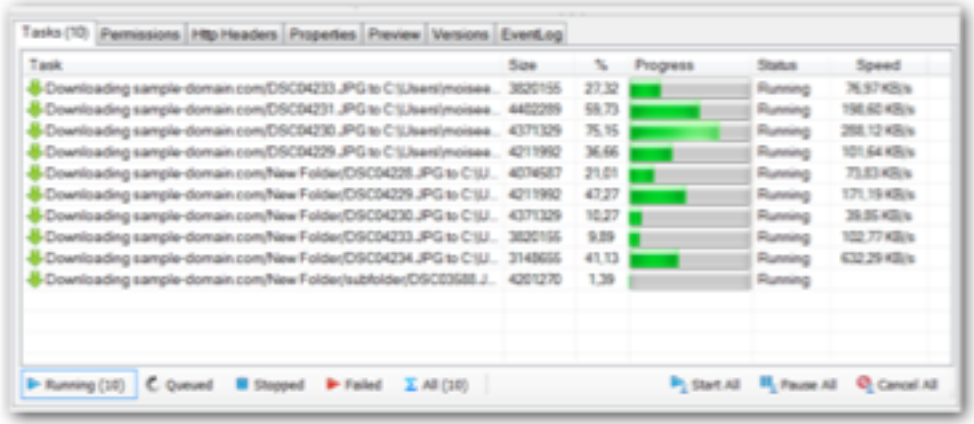
Creating/deleting user and system processes

Suspending/resuming processes

Process Synchronization & Communication

Process Management

(Chapters 3,4 & 5)



The memory is not enough memory for all my processes!

Memory is not Enough

Keeping track of which parts of memory are currently being used and by whom

Deciding which processes and data to move into and out of memory

Allocating and deallocating memory space as needed

Memory Management

(Chapters 7 & 8)

Level	1	2	3	4	5
Name	registers	cache	main memory	solid state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25 - 0.5	0.5 - 25	80 - 250	25,000 - 50,000	5,000,000
Bandwidth (MB/sec)	20,000 - 100,000	5,000 - 10,000	1,000 - 5,000	500	20 - 150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

Different Kinds of Storage Devices

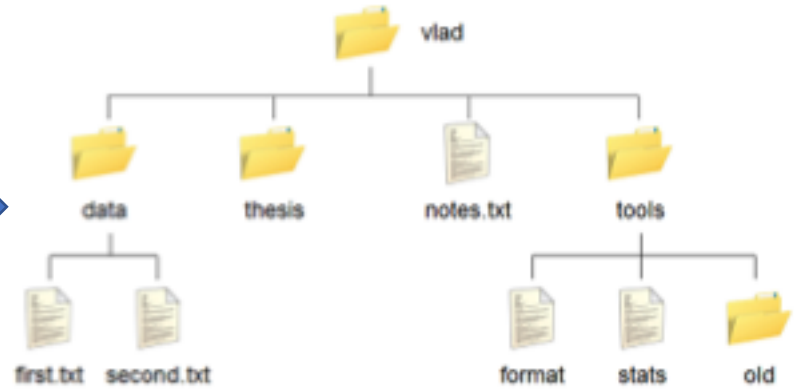
Usually disks is used to store data that does not fit in main memory or data that must be kept for a “long” period of time

Entire speed of computer operation hinges on disk subsystem and its algorithms

Free-space management, Storage Allocation, and Disk Scheduling

Mass-Storage Management

(Chapter 9)



OS provides uniform, logical view of information storage

Abstracts physical properties to logical storage unit : files, directories

Bits, Bytes, and Files

Access control to determine who can access what

Creating and deleting files and directories

Mapping and Backing files onto secondary storage

File-System Management

(Chapters 10 & 11)

Many I/O Devices

Hides peculiarities of hardware devices from the user

Memory management of I/O including buffering, caching, spooling

General device-driver interface

I/O Management

(Chapter 12)

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 including: denial-of-service, worms, viruses, identity theft, theft of service

Protection & Security

(Chapters 13 & 14)

つづく