Introduction

COS 450 - Fall 2018

1

Introduction

Define *Operating System*

Computer system organization

Operating system structure

Operating system components

2

What is an Operating System?

An operating system is a **program** that **manages** the computer hardware.

3

Page 1 of the textbook. Read it, learn it, love it. You will be tested on it.

hardware = CPU, memory, keyboard, video, audio, I/O devices

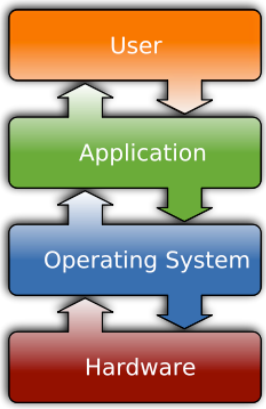
Operating System

Provides a basis for application programs and acts as an **intermediary** between the **user** and **hardware**.

4

Again from page 1 of the textbook. Read it, learn it, love it. You will be tested on it.

application programs = word processor, calculator, presentation software



5

It's clear here how the operating system lies between users, applications, and the hardware.

Applications should not get access to hardware without the operating system managing the interaction. Question: Why?

Operating System

Resource Allocator

Control Program

Programming Interfaces

6

Resources: CPU(time), memory, disks, keyboard, video display, serial ports, network.

Controls: access to all of the above.

API: Gives a way for applications (on behalf of users) to access the hardware, e.g. display characters.

Is the dog on the chair or not?

7

What is part of the operating system? What is not?

Operating System

An Operating System **is**...

- **running** all the time (the *kernel*)
- **between** the user and hardware
- **everything** the vendor ships?
- “like government, not useful by itself”

8

Internet Explorer is not part of the operating system (by law!?)

Operating System

An Operating System **is not**...

- *firmware*
- an *application* program
- easily defined

9

firmware is part of the hardware.

application programs are added on for direct user interactions. They are (one of) the reasons the operating system exists.

Introduction

Define *Operating System*

Computer system organization

Operating system structure

Operating system components

10

11

How does this thing work?

Hardware

Overall computer organization

busses, connections, etc.

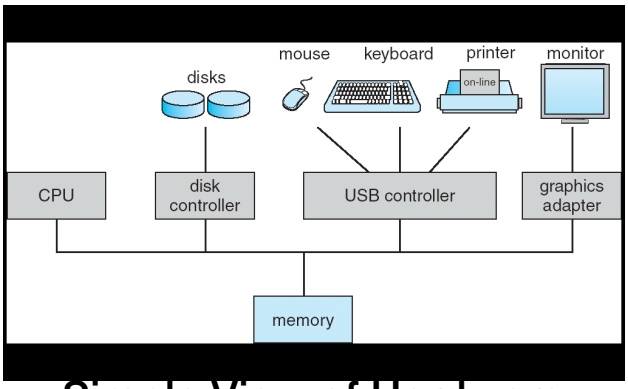
Interrupts and interrupt handling

Storage hierarchy

Single & multi-processor systems

12

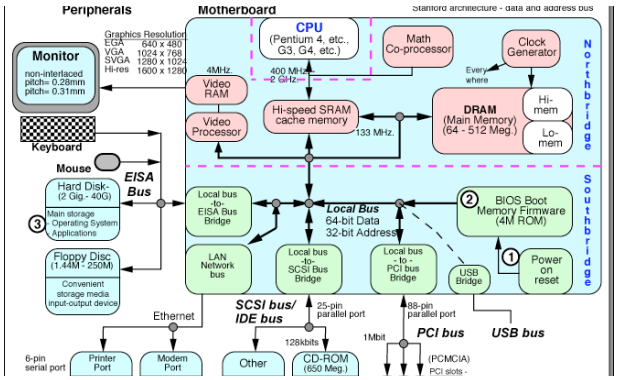
What does it look like, how does it work.



Simple View of Hardware

13

Note the shared memory bus. A likely spot for contention and a bottleneck.



Hardware Organization

14

Diving into the details we can see it's actually a bit more complex. Note the different "local/front" and other busses.

Hardware Notes

15

CPU and IO devices execute *concurrently*.

Device controllers are *in charge* of the devices attached to them.

CPU moves data to/from memory and device buffers.

Devices use *interrupts* to alert CPU work is done.

Central Processing

Fetcher

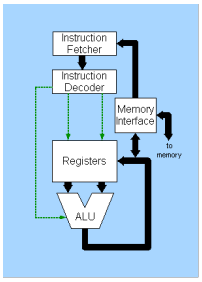
Decoder

Registers

Arithmetic Logic Unit

Memory Interface

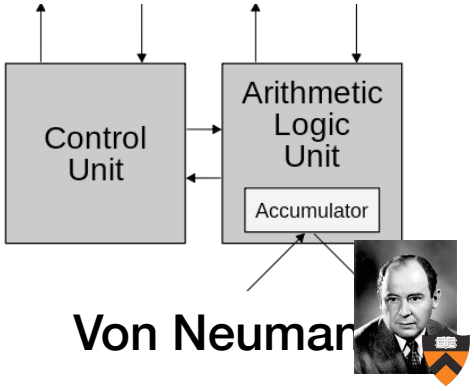
Interrupts



16

Registers: general purpose, stack pointer, instruction pointer, instruction register

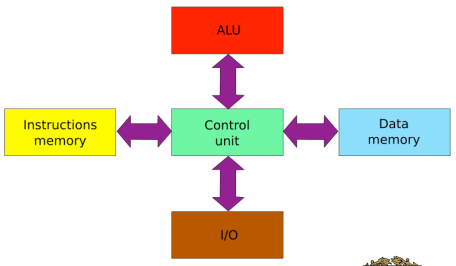
---



17

"a stored-program computer in which an instruction fetch and a data operation cannot occur at the same time because they share a common bus"

http://en.wikipedia.org/wiki/Von_Neumann_architecture



18

"The **Harvard architecture** is a computer architecture with physically separate storage and signal pathways for instructions and data."

http://en.wikipedia.org/wiki/Harvard_architecture

Interrupts

An **interrupt** transfers control to an *interrupt service routine* through an *interrupt vector table*.

Incoming interrupts are disabled when servicing another interrupt.

A **trap** is a software generated interrupt.

An **interrupt driven** operating system.

19

The word trap and interrupt are often used interchangeably.
What is the usefulness of “software generated interrupts?”

Interrupt Handling

When an interrupt happens...

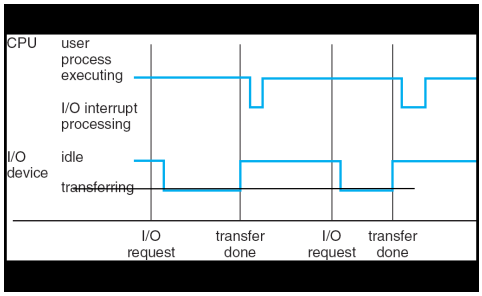
Preserve the state of the CPU

Determine type of interrupt (by *polling* or *vectored* methods)

Execute interrupt handler

Restore state of the CPU

20



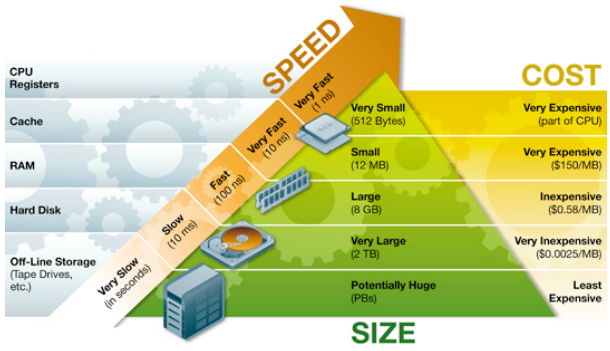
Interrupt Handler

21

Note how they can execute concurrently (asynchronously)
The vertical lines are when user/ kernel communication happens

- by system call
- by interrupt

22



Storage Structure

Also know as storage hierarchy

Main memory is usually too small to store all needed programs and data Main memory is a volatile *storage device that loses contents when powered off.*

flash or NVRAM (non-volatile) keeps data on power-off.

There's a lot of research and

23

I/O Structure

More than just storage.

Device ***drivers*** for each device ***controller***

Control and ***data*** registers

Direct Memory Access for large quantities of data

Switch rather than bus architecture (faster)

Control and Data registers may be in separate I/O address space or mapped into main-memory.

You can think of this as a parallel memory space, yet much smaller, and connected to the devices.

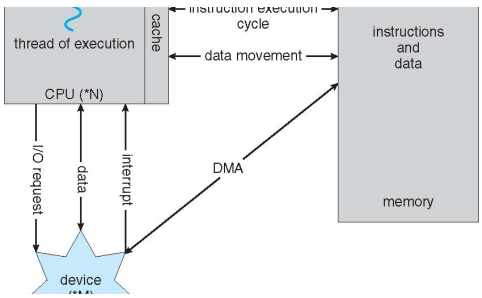
24

Processor Structures

Single Processor

Multiple Processor

Clustered

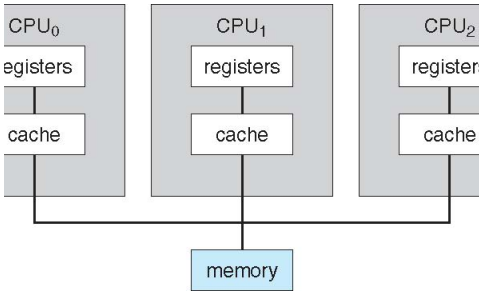


Single Processor

25

I/O processors generally have a limited instruction set and are not useful for general purpose execution.

Though more high end video cards are being used for special purposes (e.g. encryption cracking)

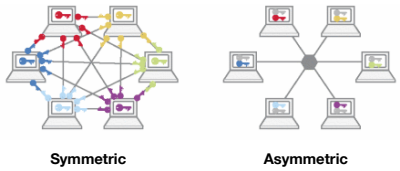


Multi-Processor

26

Why?
1. Increased throughput (more compute power)
2. Economy of scale (shared devices)
3. Increased reliability (redundancy)

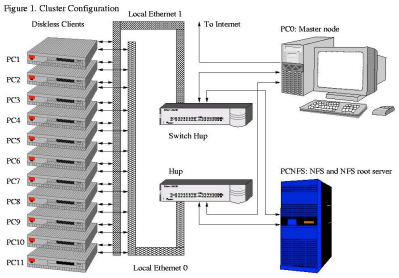
Multi-core -- effectively duplicate CPU “on chip”
Share caches, share memory, share bus



27

Asymmetric - each processor assigned a specific task
Symmetric - each processor performs all (any) tasks

28



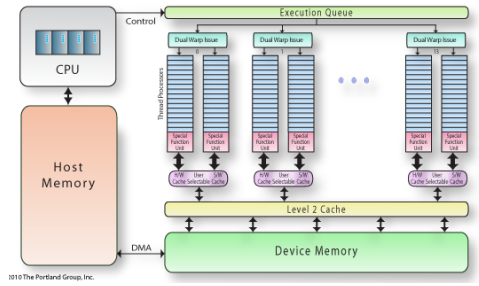
Cluster

Each system has its own devices and processing core(s), communicates over network.

Type of multiple CPU system using multiple “machines”
Connected by LAN.
Provides high-availability (failure tolerant)
Same symmetric and asymmetric models.
Can provide high-performance and parellelization with off the shelf equipment.

Need to deal with distributed locks.

29



GPU

Executes same operation across large number of data elements at the same time.

30


Microsoft
Windows 95

31


Microsoft
Windows 98

32


Microsoft
Microsoft
Windows 2000
Professional
Built on NT Technology
Starting up...
Copyright © 1985-1999 Microsoft Corporation

33

34


Microsoft®
Windows®xp
Copyright © Microsoft Corporation
Microsoft®

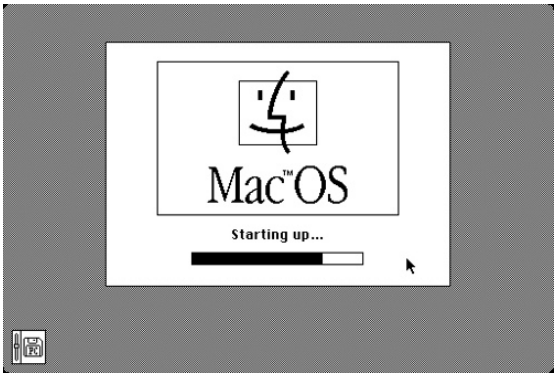
35


Microsoft®
Copyright © 1985-1996
Microsoft Corporation
Microsoft®
Windows NT®
Workstation 4.0
with Microsoft Internet Explorer
This product is protected by US and international copyright laws as described in the About Box.

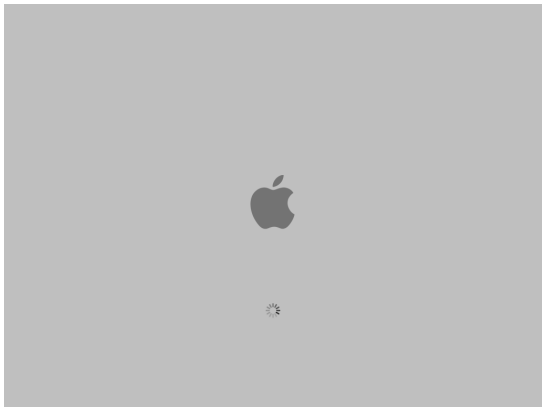
36


Starting Windows
© Microsoft Corporation

37



38



39

Bootstrap

When a computer is **powered on**...

processor starts executing at fixed location

typically in ROM (flash) *firmware*

initialize all aspects of system (hardware)

load and execute the *kernel*...

Where does execution start on an 80x86 processor?

… at what address? (0xFFFF:0000, 16-bytes before the end of memory)

… what's typically there?

Introduction

Define *Operating System*

Computer system organization

Operating system structure

Operating system components

40

Multitasking

41

The most important aspect of an operating system is managing the multiple things that should happen at the same time.

Multitasking Structure

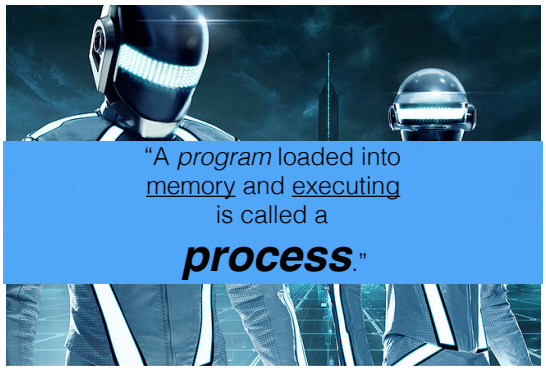
Multiprogramming; organizing jobs so there is always code to execute.

Time shared (**multitasking**) executes multiple jobs by switching quickly among them.

job scheduling and **memory management** are key to multi-success.

42

Working with the “scheduler” is what Pintos Project 1 - Threads is all about.



43

The process is the fundamental "thing" we manage in an operating system.
Processes are the object of a scheduler's affection.

Often "thread" and "process" are used interchangeably. That's o.k. for now, we will work on a more precise definition later.

In Tron the actors played processes

Keys to Process Management

Dual-mode Operation

privileged instructions

- Interrupts (and traps)

The timer

44

Or, better named, how do we accomplish this process management. What tools do we have?

Dual-Mode Operation

To enforce protection need at lea
operation

user mode

kernel mode
(or supervisor mode,
protected mode, privileged mode

45

That's AC DC in the photo.

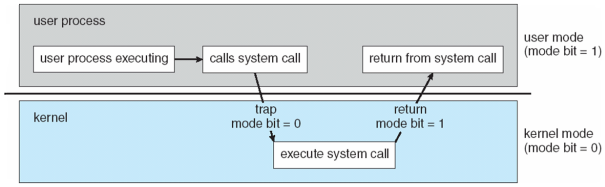
Privileged Instructions

46

Some instructions are privileged and **cannot** be executed in **user mode**.

User mode execution of a privileged instruction **causes** an **interrupt**.

This is how an operating system can communicate between user and kernel modes.



Kernel mode transition

(by trap or interrupt)

47

The kernel executes protected instructions and operations on behalf of the user (which executes in user-mode).

This is what Pintos Project 2 - User Programs is all about.

Timer

48

The **timer** regularly interrupts the CPU

- Causes CPU to handle interrupt in kernel mode.
- Check on health of system.
- Opportunity to fix errant programs.
- Used to multitask or schedule jobs

You will see the timer, and it's interrupt handler, in Pintos Project 1 - Threads

Introduction

49

Define *Operating System*

Computer system organization

Operating system structure

Operating system components

Major Subsystems

50

Process Management

Memory Management

Storage Management

Protection & Security

Process Management

51

A **program** in execution is a *process*.

- Scheduling processes and threads
- Creating and removing processes
- Providing for intraprocess synchronization
- Providing for intraprocess communication

Project 1 and Project 2

Memory Management

52

Processes and their data reside in memory

Keep track of what memory is used and by which process.

Deciding which processes (or parts) and data move into and out of memory

Allocating and deallocating space as needed.

virtual memory, swapping, etc.
Project 3

Storage Management

53

Physical, long-term, program and data storage

File-system management

Mass-storage management

Caching (to speed up access)

I/O Subsystems (buffers, etc.)

Project 4 - Graduate Students Only

Protection and Security

54

Multiple users, concurrent execution, multiple processes, all need protection from each other

Protection = controlling access to resources.

Security = defend from attacks

Introduction

Define *Operating System*

Computer system organization

Operating system structure

Operating system components

55

Types of Operating Systems

General purpose

Distributed systems

Real-time and embedded systems

Multimedia systems

Mobile systems

56

We will primarily be looking at general purpose OSs

Distributed; using a network of connected computers over LAN, MAN, small-area network

Real-time; for special purpose controllers; fixed time constraints. These are the most prevalent, you just don't see them -- cars, microwaves, etc..

Types of Computing

Traditional

Client-Server

Peer-to-Peer

Web-based

57

Traditional; you have your computer
Client-Server; Internet browser, connect to servers to compute
Peer-to-Peer; Like client-server but no single server, file sharing.
Web-based: very much akin to Client-server; Google Docs.

Popular Operating Systems

58

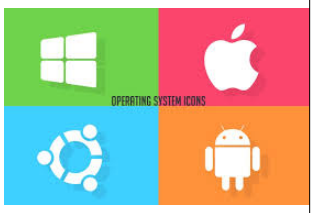
Microsoft Windows

UNIX, GNU/Linux, Solaris

Apple Mac OS X

iOS

Android



Open source vs closed source.

Summary

59

An operating system...

- is a **program** that **manages** the computer **hardware**.
- acts as an **intermediary** between **users** and the **hardware**
- is a **resource allocator**
- is a **control program**

More Resources

60

Security Now! Podcast (older episodes)

- grc.com/securitynow.htm

CS 162 - UC Berkeley OS Course

- inst.eecs.berkeley.edu/~cs162

Wikipedia

- wikipedia.org/wiki/Operating_system

End

Introduction