


Chapter 0

INTRODUCTION



Contents

- References
- Grading policy
- The Content
- Course overview
- Why study computer architecture
- What is computer
- Organizations and Architecture
- What will you learn




References

- David A. Patterson and John L. Hennessy, **Computer Organization & Design–The Hardware/Software Interface**, 4th Edition, Morgan Kaufmann Publishers, 2008
- William Stallings, **Computer Organization and Architecture–Designing for Performance**, 7th Edition, Pearson International Edition, 2006.
- Linda Null, Julia Lobur, *The Essentials of Computer Organization and Architecture*, Jones and Bartlett Publishers, 2003
- <https://www.jdoodle.com/compile-assembler-nasm-online>




Grading policy

- Attend class
- Homework
- Midterm examination
- Labs
- Final examination



Contents

- Chapter 0 : Introduction to course
- Chapter 1 : Introduction
- Chapter 2 : Top-level view of Computer functions and interconnections
- Chapter 3 : Computer Memory
- Chapter 4 : The Central Processing Unit
- Chapter 5 : Data Presentation
- Chapter 6 : Instruction Set
- Chapter 7 : Addressing Mode and Stack Processing
- Chapter 8 : Macro and Function
- Chapter 9 : System services



Course overview

- Principle and organization of digital computers
- Performance issues in computer architecture
- Bus organization and memory design
- Principle of computer's instruction set and programming in assembly language (some popular processors are used such as Intel x86, ARM, ...)
-



Why study computer architecture

- To be a professional in any field of computing today, you should not regard the computer as just a black box that executes program by magic.
- You should understand a computer system's functional components, their characteristics, their performance, and their interactions.
- You need to understand computer architecture in order to build a program so that it runs efficiently on a machine.
- When selecting a system to use, you should be able to understand the tradeoff among various components, such as CPU clock speed vs. memory size.



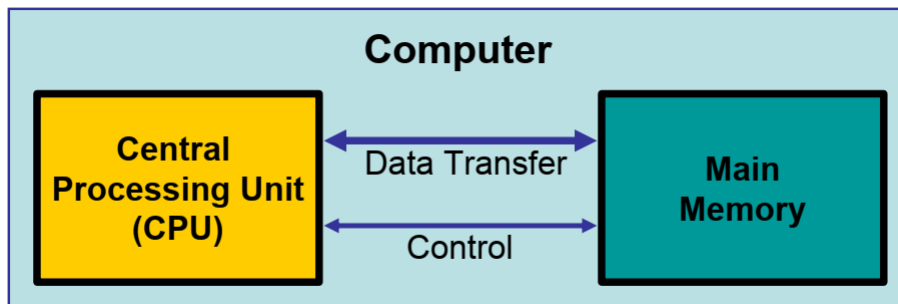
- Understand the complex trade-offs between CPU clock speed, cache size, bus organization, number of core processors,...
- What determines the performance of a program, and how can a programmer improve the performance? As we will see, this depends on the original program, the software translation of that program into the computer's language, and the effectiveness of the hardware in executing the program.
- Understand other areas of computing curriculum such as operating system, passing parameter/pointers, stack frame in high-level language,

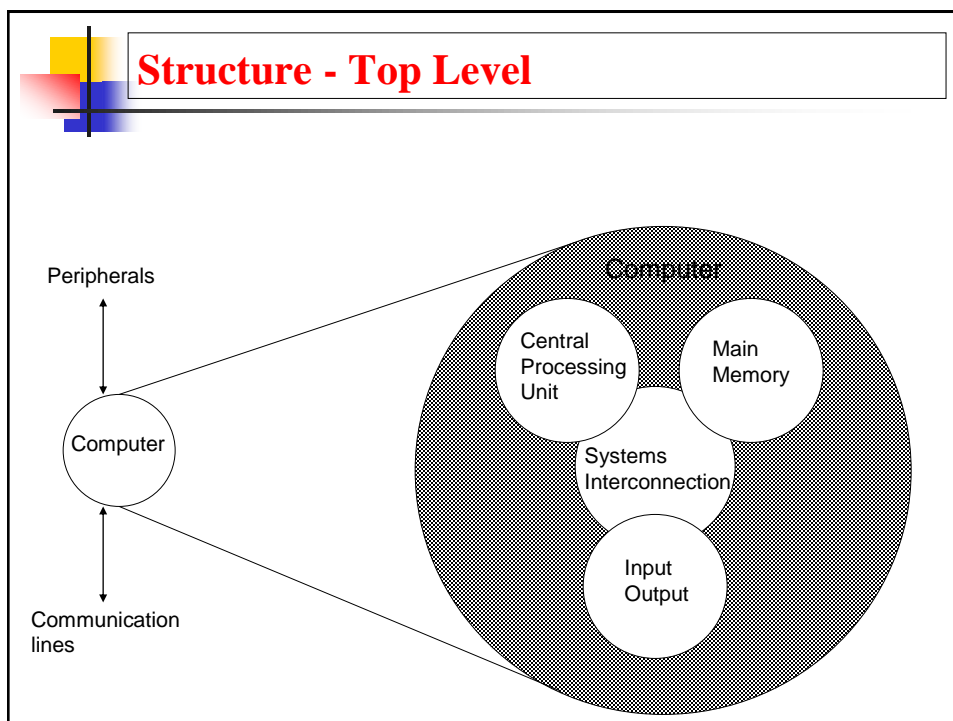
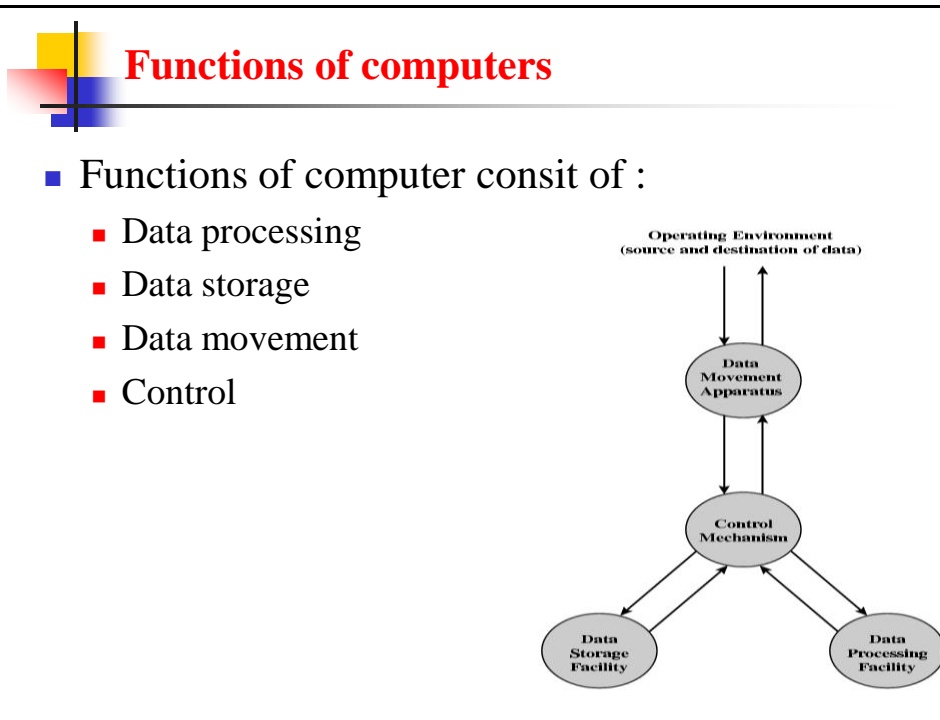
What is Digital computer

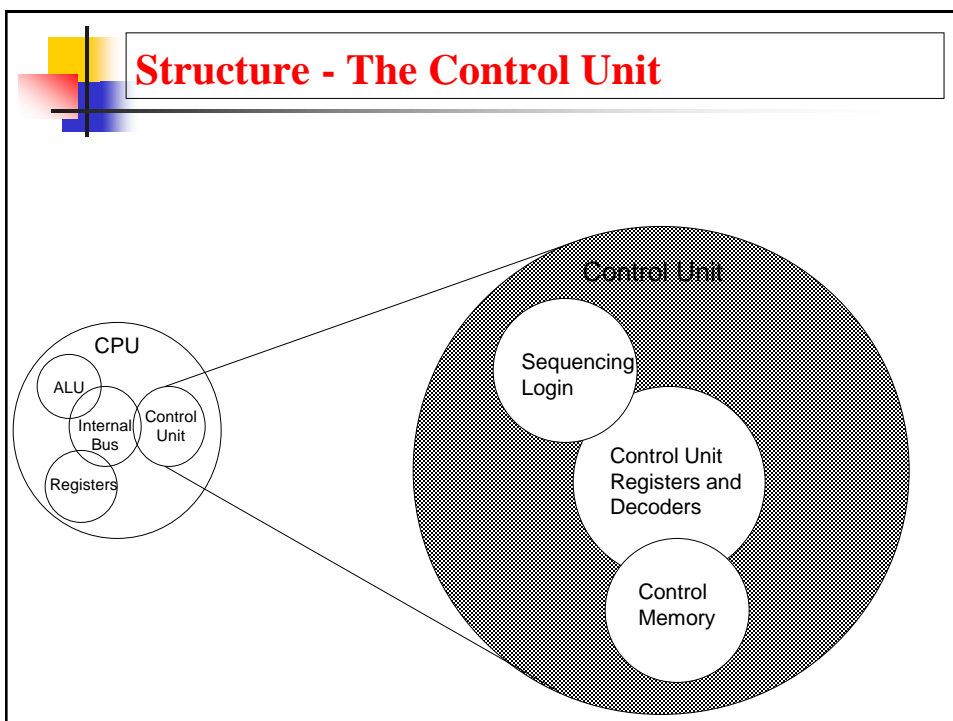
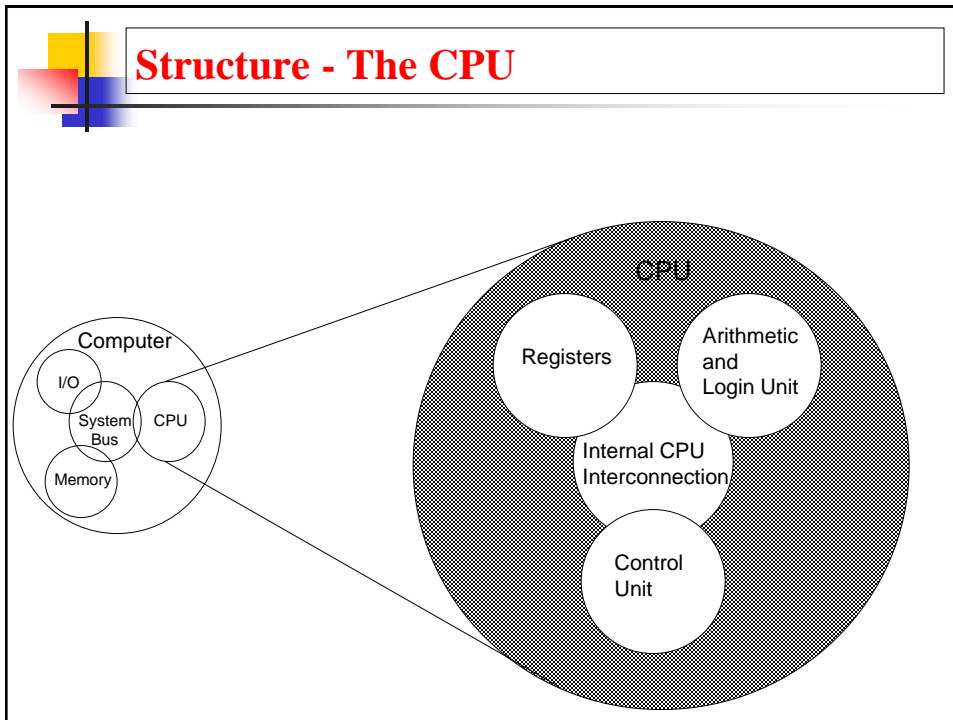


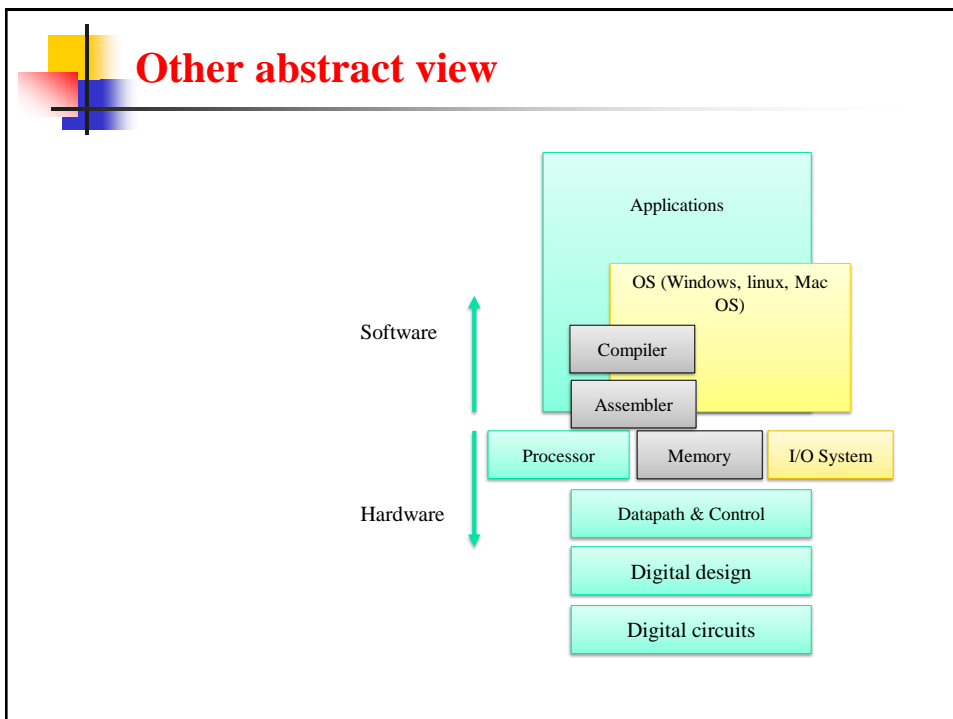
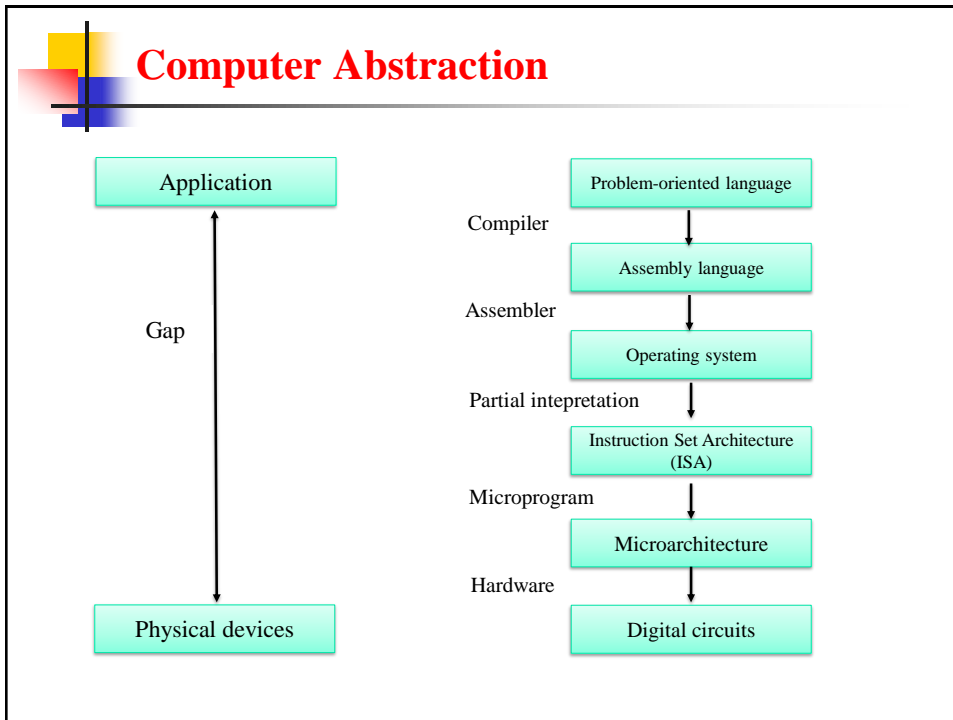
Define


- Computer is a data processing machine, operation automatically under the control of one list of commands (called programs) is stored in memory













Abstractions

- Abstraction helps us deal with complexity
 - Hide lower-level detail
- Instruction set architecture (ISA)
 - The hardware/software interface
- Application binary interface
 - The ISA plus system software interface
- Implementation
 - The details underlying and interface




Organizations and Architecture

- What Is Computer Architecture?
 - Computer Architecture refers to those attributes of a system that have a direct impact on the logical execution of a program. Examples:
 - The instruction set
 - The number of bits used to represent various data types
 - I/O mechanisms
 - Memory addressing techniques
- ISA, Organization, Implementation
- Instruction Set Architecture: SW/HW interface

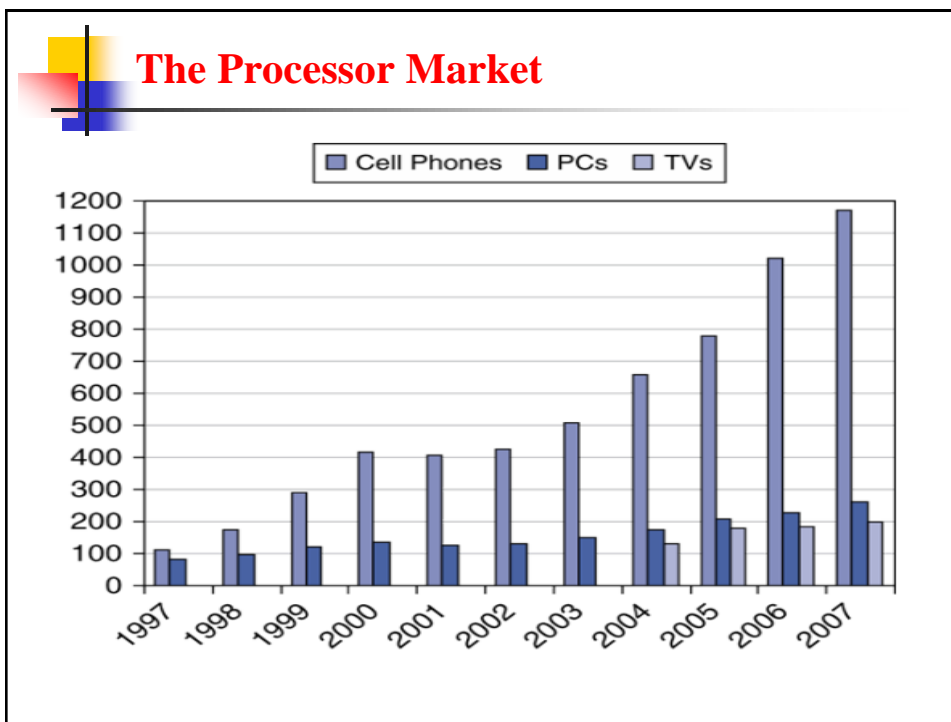


- Instructions
- Memory management and protection
- Interrupts and traps
- Floating-point standard (IEEE)
- Organization: also called microarchitecture
 - Computer Organization refers to the operational units and their interconnections that realize the architectural specifications. Examples are things that are transparent to the programmer:
 - Control signals
 - Interfaces between computer and peripherals
 - The memory technology being used




Classes of Computers

- Desktop computers
 - General purpose, variety of software
 - Subject to cost/performance tradeoff
- Server computers
 - Network based
 - High capacity, performance, reliability
 - Range from small servers to building sized
- Embedded computers
 - Hidden as components of systems
 - Stringent power/performance/cost constraints




What will you learn

- How programs are translated into the machine language
 - And how the hardware executes them
- The hardware/software interface
- What determines program performance
 - And how it can be improved
- How hardware designers improve performance
- What is parallel processing



Below Your Program

- Application software
 - Written in high-level language
- System software
 - Compiler: translates HLL code to machine code
 - Operating System: service code
 - Handling input/output
 - Managing memory and storage
 - Scheduling tasks & sharing resources
- Hardware
 - Processor, memory, I/O controllers



Levels of Program Code

- High-level language
 - Level of abstraction closer to problem domain
 - Provides for productivity and portability
- Assembly language
 - Textual representation of instructions
- Hardware representation
 - Binary digits (bits)
 - Encoded instructions and data

