COMP2611: Computer Organization

Introduction

Course Info.

Course's homepage http://course.cse.ust.hk/comp2611

Course Facebook: search HKUST CSE COMP2611 Spring 2016

Lecture 1

TuTh 10:30AM – 11:50AM, LTK

Instructor: Dr. Brahim Bensaou brahim@cse.ust.hk

Lecture 2

TuTh 3:00PM – 4:20PM, Room 5583 (Lift 29/30)

Instructor: Dr. Cindy LI lixin@cse.ust.hk

Lecture 3

MoWe 9:00AM-10:20AM, Room 2404 (Lift 17/18)

Instructor: Dr. Alex Lam lamngok@cse.ust.hk

You also need to attend Tutorials and Labs, which are necessary supplements to lectures

Reading the textbook is also a very important part in the workflow of this course.

- **Grading**
  - 2 Homework 15% (2 x 7.5%)
  - Programming Project 15%
  - 2 Midterm Exams 30% (2 x 15%)
    - Mar 10 (Thur) and Apr 14 (Thur) 7pm
  - Final Exam 40%
  - All students in COMP2611 use the same set of assessment materials and are graded together

- **Policies**
  - Course project should be individual work; both 'provider' and 'copier' will be penalized equally and harshly
  - Skipping the midterms or final examination without prior approval will automatically lead to an "F" grade for the course

Basic assumptions and terminology

- How do computers represent data? Electrical signals (two states)
  - Therefore computing relies on base 2 to represent numbers.
- What is base 2 anyway?
  - We actually use base 10 (decimal) in our daily calculations
    - 1452 is actually:

1	4	5	2
10^3	10^2	10^1	10^0

    - Base 10 has 10 digits 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9
  - Base 2 (binary) uses two digits or (Bits) 0 and 1
    - $8_{10} = 1000_2$; $17_{10} = 10001_2$
    - Conversion from base 10 to 2 is done via successive divisions by 2
- Many other bases have been used over the millennia
  - Base 60 (Sumerians civilization in Iraq, remnants are found in timekeeping)
  - Base 1  (herringbone) 
  - Base 16 (hexadecimal) very useful in Computer Science (seen later)
    - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Basic assumptions and terminology

- When dealing with a size (e.g., Memory or file)
  - Kilo – 2^{10} or 1024
  - Mega – 2^{20} or 1024 Kilo
  - Giga – 2^{30} or 1024 Mega
  - Tera – 2^{40} or 1024 Giga
  - Peta - 2^{50} or 1024 Tera
  - …

  Example:
  - The memory in my computer is 4 Gigabytes
  - The PPT file for this lecture is 2.5 Megabytes

- When dealing with a rate/frequency (e.g., # instructions per second, # clock ticks per second)
  - Kilo – 10^{3} or 1000
  - Mega – 10^{6} or 1000 Kilo
  - Giga – 10^{9} or 1000 Mega
  - Tera – 10^{12} or 1000 Giga
  - Peta - 10^{15} or 1000 Tera
  - …

  Example:
  - The speed of my network card is 1 Gigabit per second
  - The speed of my Intel processor is 2.89 Gigahertz

Classes of Computers

Computers have led to a **third revolution** for civilization:
agricultural -> industrial -> **information**

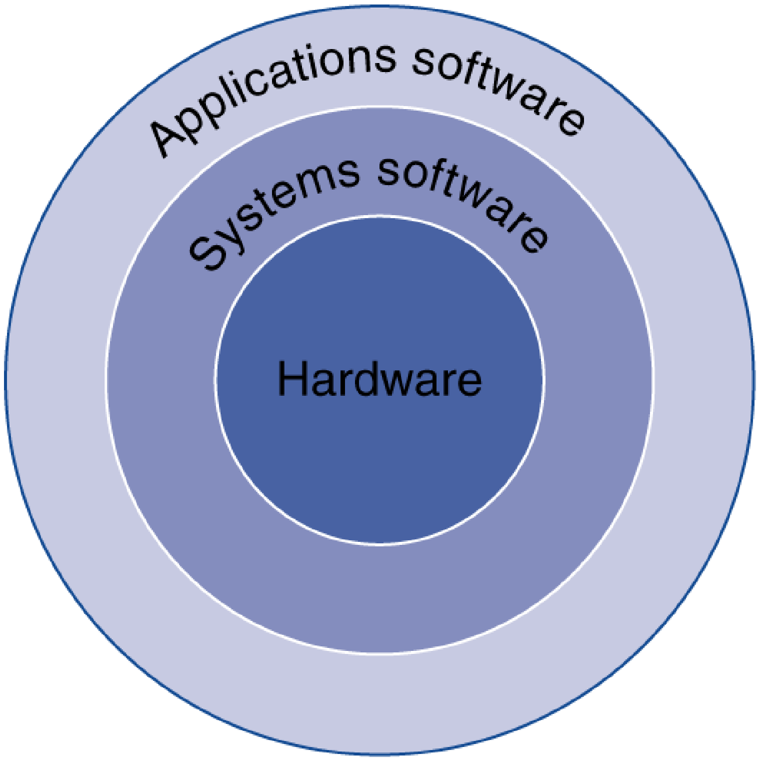
- **Desktop computers**:
  - Run a variety of general purpose software
  - Designed to achieve good performance at low cost
- **Embedded computers**:
  - Usually hidden as a component of a system (e.g., mobile phone, cars, airplanes, ATM machines, Smart card, …)
  - Run a predefined program
  - Subject to a stringent power/performance/cost constraint
- **Servers and Networked computers**:
  - High storage and computing capacity, performance and reliability
  - Used to run large programs for multiple users
  - Only accessible via a network
  - Range from small servers to building sized, to several thousand computers in a grid

"What" you will learn in COMP2611

- How programs are translated from high level programming language to machine language
- How the hardware executes programs written in machine language
- The interface between the hardware and the software or the Instruction Set Architecture (ISA)
- What determines the performance of a program and how it can be improved
- How hardware designers improve the performance

- How to measure and analyze computer performance
  - To tell why a design is good or bad – Chapter 1
- How computers work
  - Computer Arithmetic and implementation – Chapter 3
  - Issues affecting design of modern processors – Chapters 2, 4 (and 7)
  - Exploiting memory hierarchy – Chapter 5

Below Your Program



- **Application software**
  - Written in high-level language
  - Ex: Comp2011 assignment written in C++
- **System software**
  - Compilers: translates HLL code to machine code
  - Operating System: service code
    - Handle input/output
    - Manage memory and storage
    - Schedule tasks & share resources
- **Hardware**
  - Processor,
  - memory,
  - I/O controllers

Levels of Program Code

for human

for machine

High-level language program (in C)

```
swap(int v[], int k)
{int temp;
   temp = v[k];
   v[k] = v[k+1];
   v[k+1] = temp;
}
```

C compiler

Assembly (low-level) language program (for MIPS)

```
swap:
      muli $2, $5,4
      add  $2, $4,$2
      lw   $15, 0($2)
      lw   $16, 4($2)
      sw   $16, 0($2)
      sw   $15, 4($2)
      jr   $31
```

Assembler

Binary machine language program (for MIPS)

```
00000000101000010000000000011000
00000000100011100001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```

Levels of Abstraction in Program Code

- **High-level language**
  - Level of abstraction closer to the problem domain
  - Helps increase productivity, portability and simplify debugging
- **Assembly language**
  - Binary instructions represented in symbolic notation
  - One to one mapping with binary instructions
  - Assemblers translate from Assembly language to machine language
- **Hardware representation**
  - Computers only deal with binary digits (bits)
  - Instructions and data are encoded as bit strings

```
swap(int v[], int k)
{int temp;
  temp = v[k];
  v[k] = v[k+1];
  v[k+1] = temp;
}
```

C compiler

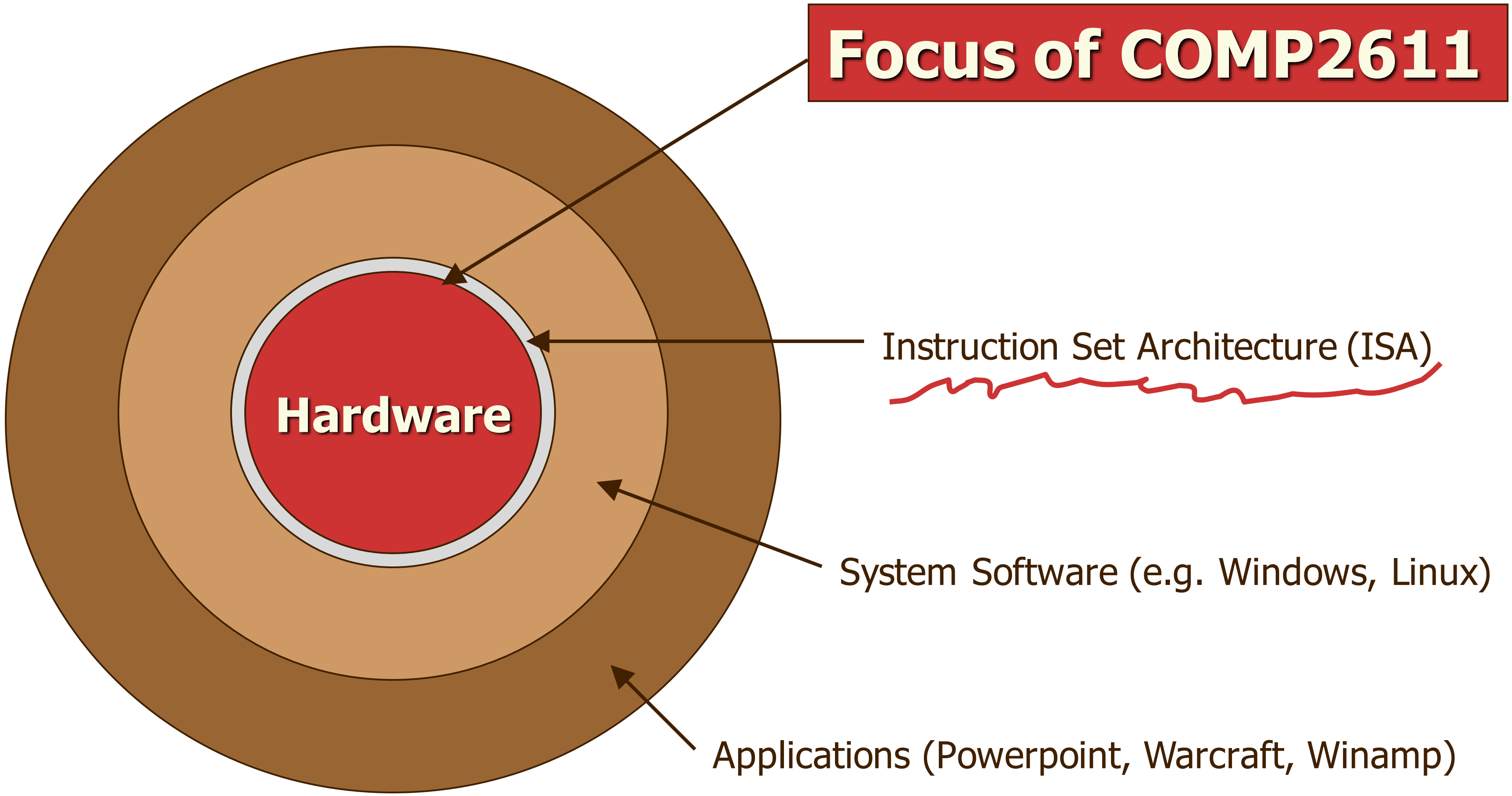
```
swap:
      muli $2, $5,4
      add  $2, $4,$2
      lw   $15, 0($2)
      lw   $16, 4($2)
      sw   $16, 0($2)
      sw   $15, 4($2)
      jr   $31
```

Assembler

```
00000000101000010000000000011000
00000000100011100001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```

Levels of Abstraction

Impossible to understand computer components by looking at every single transistor. Instead, **abstraction** is needed.



Levels of Abstraction (cont'd)

- **Key ideas**:
  - Both hardware and software are organized into **hierarchical layers**.
  - Hierarchical organization helps to cope with system **complexity**.
  - Lower-level details are **hidden** to offer a simpler view at the higher levels.
  - Interaction between levels occurs only through well-defined **interface**.

- **Example:**
  - Interface between hardware and software: Instruction set architecture (ISA)

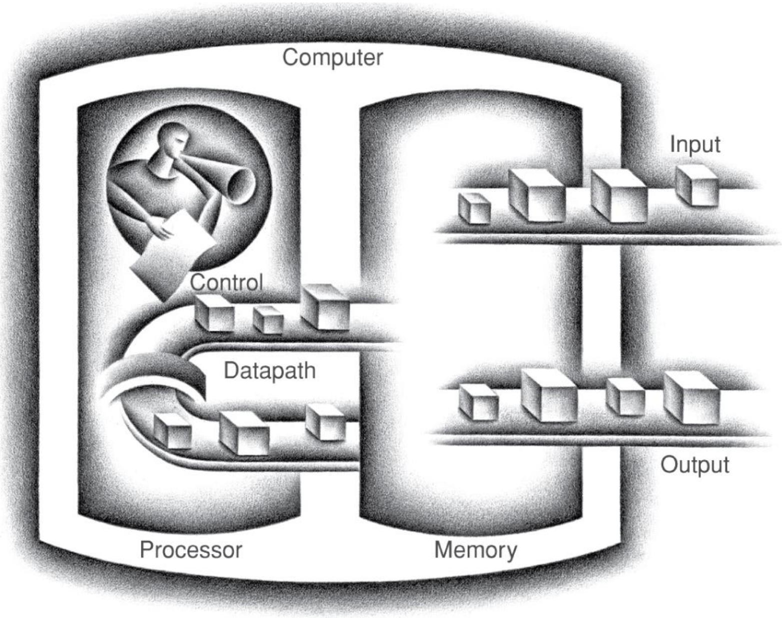
Instruction Set Architecture

An **instruction set architecture (ISA)** provides an **abstract interface** between hardware and low-level software.

- **Advantage**: allows different implementations of varying cost and performance to follow the same instruction set architecture (i.e., to run the same software).
  - Example: 80x86, Pentium, Pentium II, Pentium III, Pentium 4 all implement the same ISA

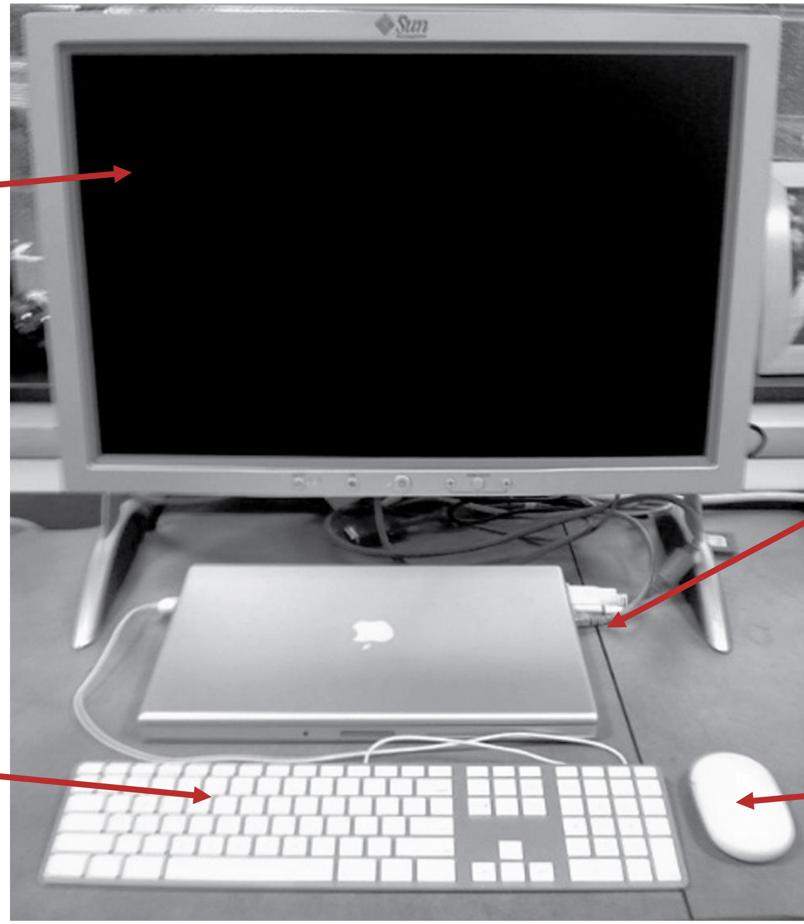
- Some instruction set architectures:
  - **80x86/Pentium/K6** (offers different implementations)
  - **MIPS**
  - **ARM**
  - **PowerPC**

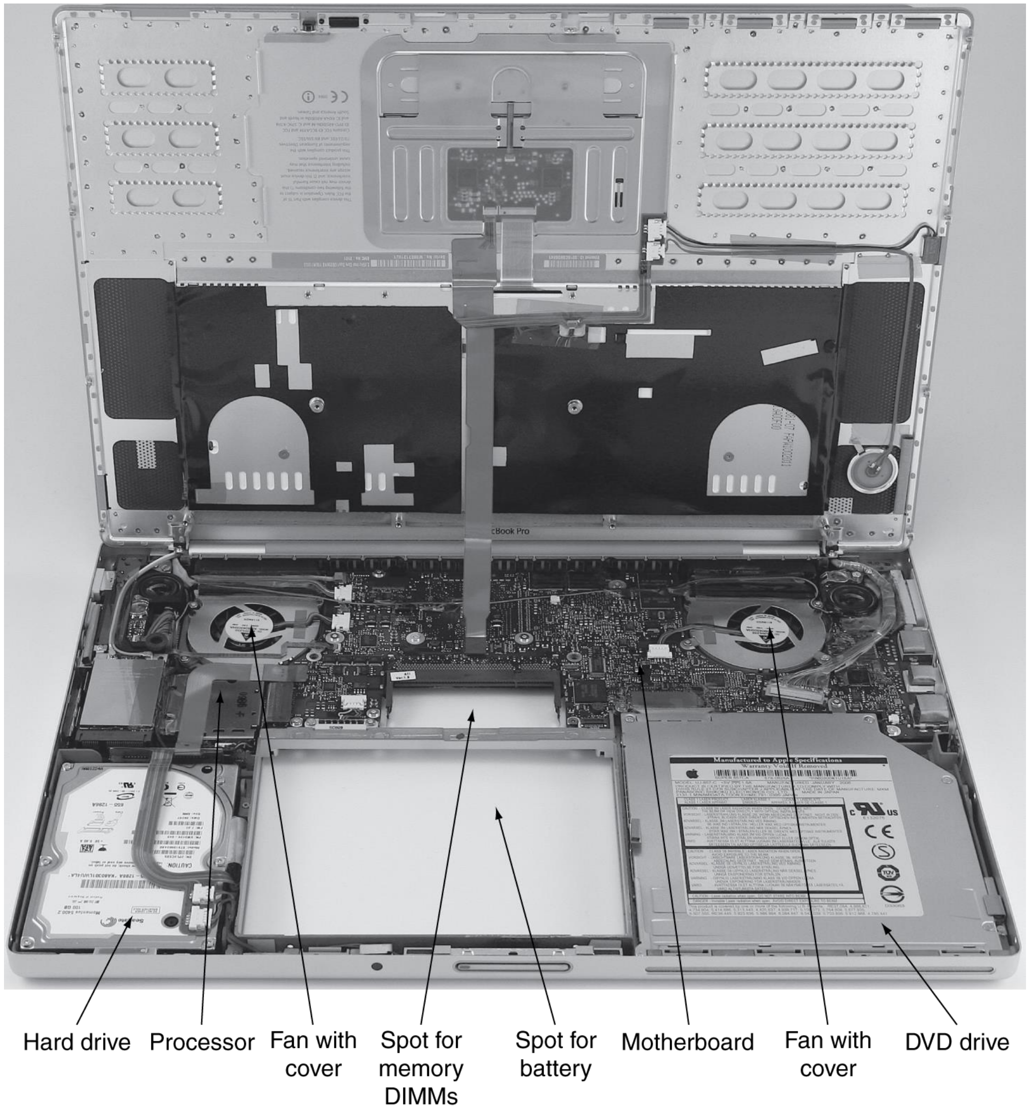
Five Basic Components (all kinds of computers)



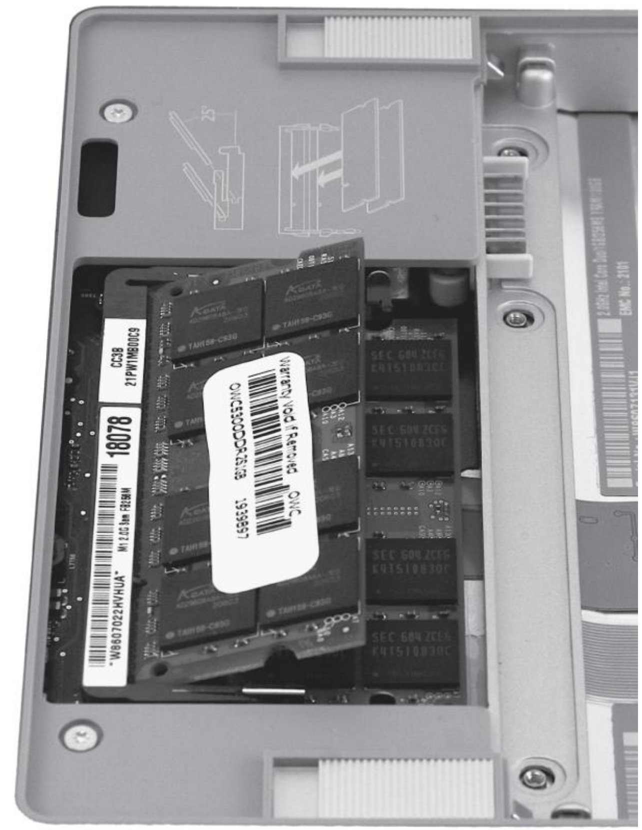
- **Input**:
  - To communicate with the computer
  - Data and instructions transferred to the memory
- **Output**:
  - To communicate with the user
  - Data is read from the memory
- **Memory**:
  - Large store to keep instructions and data
- **Processor**, which consists of:
  - **Datapath**: processes data according to instructions.
  - **Control**: commands the operations of input, output, memory, and datapath according to the instructions.

Anatomy of a Computer: I/O



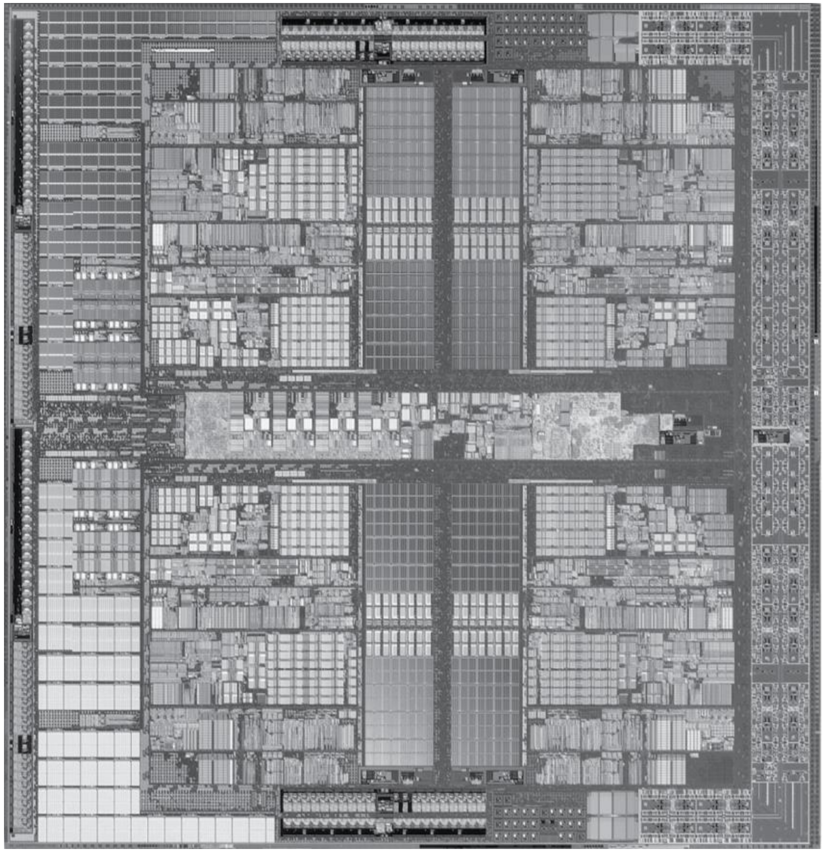


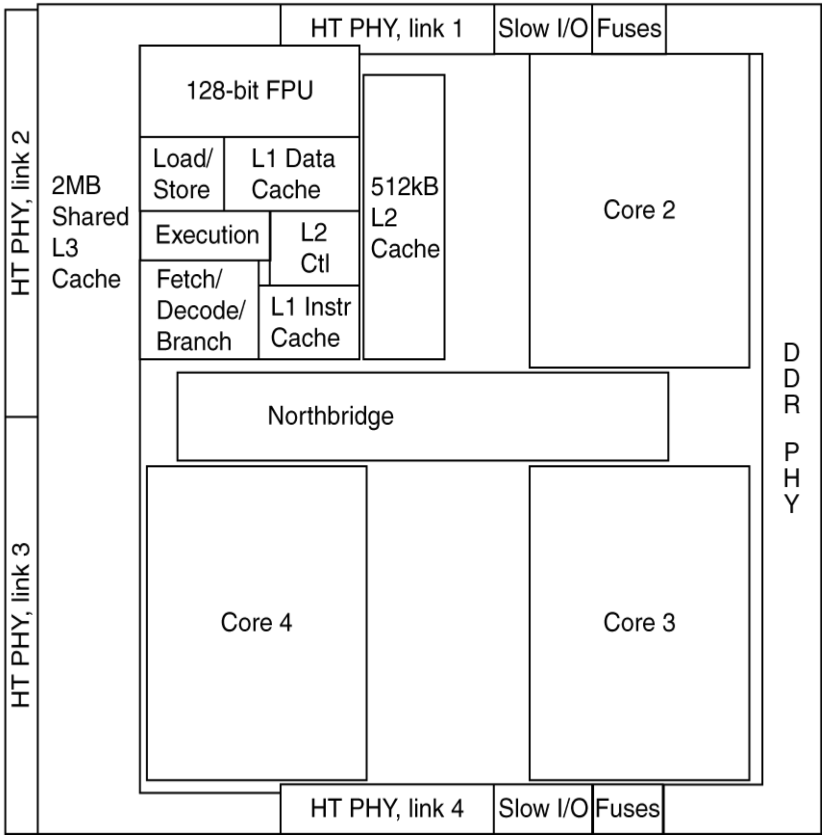
Hard drive
Processor
Fan with cover
Spot for memory DIMMs
Spot for battery
Motherboard
Fan with cover
DVD drive



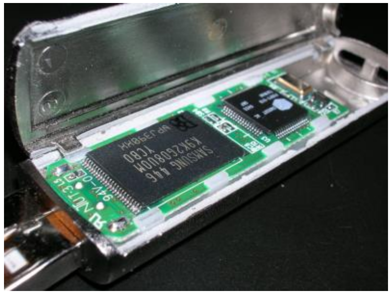
Anatomy of a Computer: Inside the Processor

- AMD Barcelona: 4 processor cores

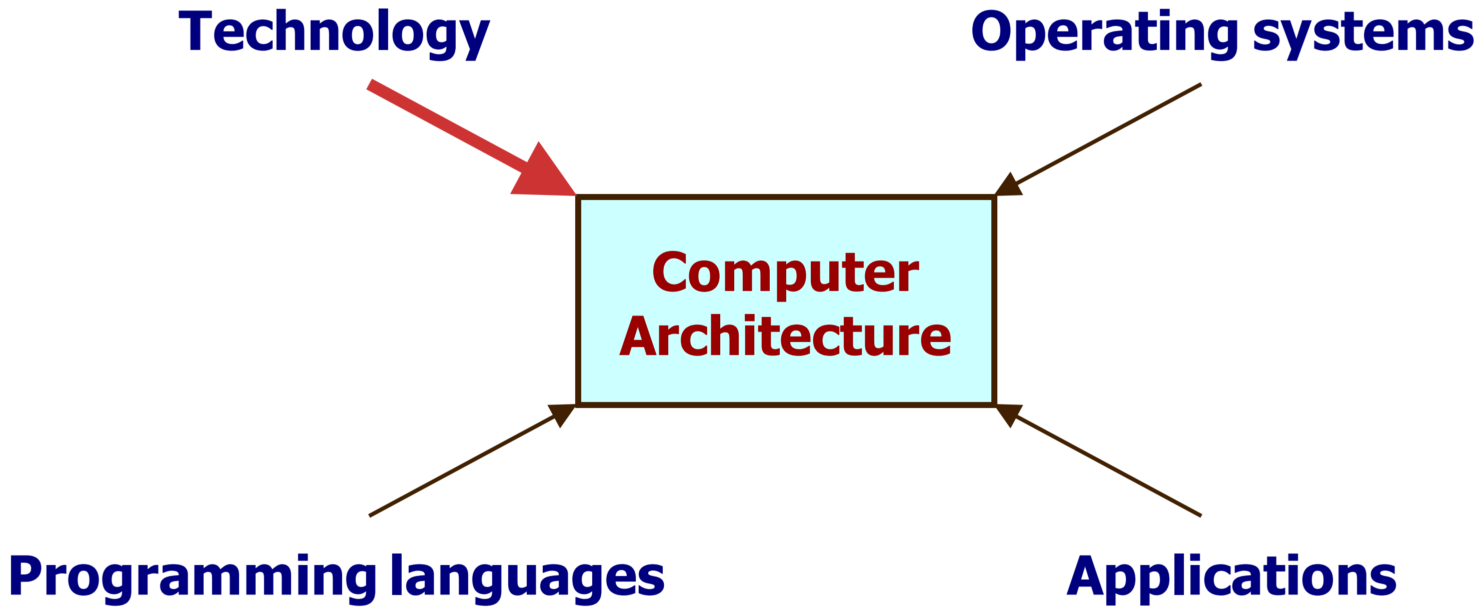




- Volatile main memory (RAM)
  - Used by the processor to store programs and data
  - Loses instructions and data when powered off
- Non-volatile secondary memory
  - Magnetic disk
  - Flash memory
  - Optical disk (CDROM, DVD)



Rapidly Changing Forces on Computer Architecture

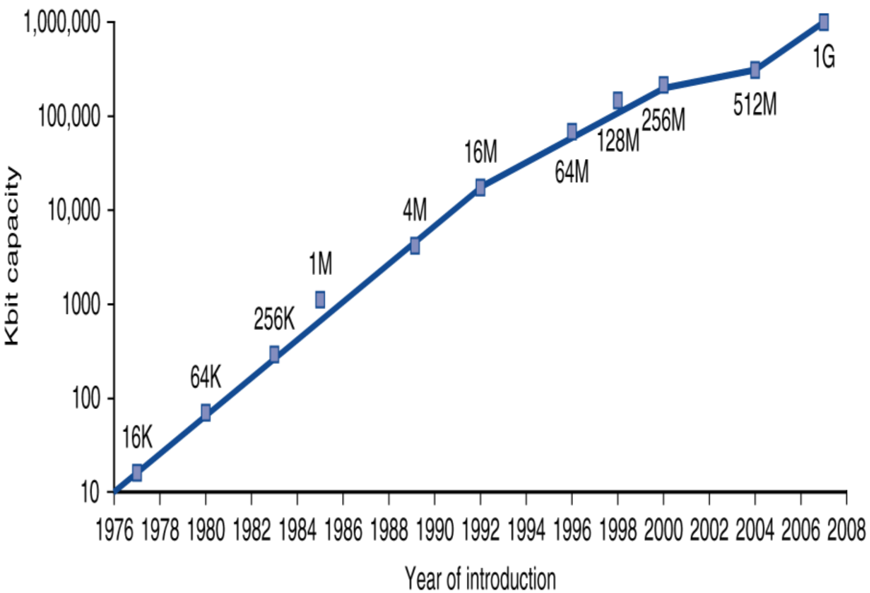


- Increased capacity and performance
- Reduced cost

- **Processor**:
  - **Logic capacity**: ~30% per year
  - **Clock rate**: ~20% per year

- **Memory**:
  - **DRAM capacity**: ~60% per year (or ~4X every 3 years)
  - **Memory speed**: ~10% per year
  - **Cost per bit**: decreases ~25% per year

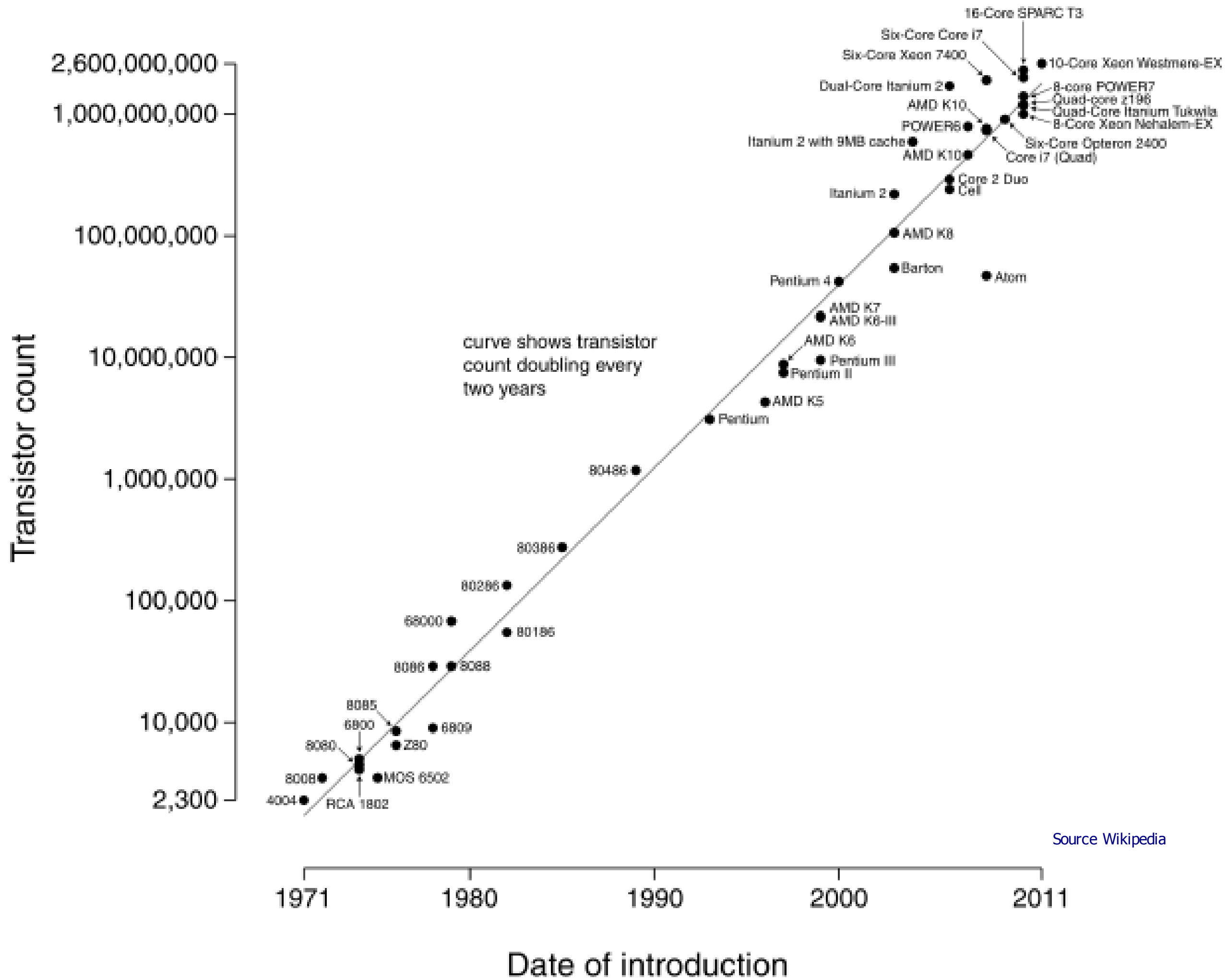
- **Disk**:
  - **Capacity**: ~60% per year



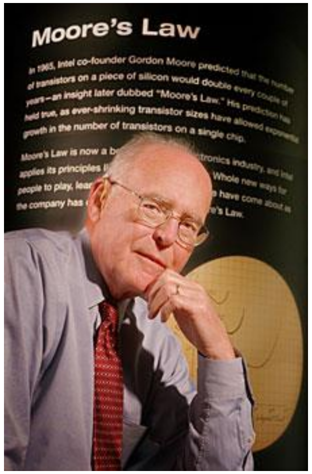
Year	Technology used in computers	Relative performance per unit cost
1951	Vacuum tube	1
1965	Transistor	35
1975	Integrated circuit (IC)	900
1995	**Very large scale integrated (VLSI) circuit**	2,400,000
2005	**Ultra large scale integrated circuit**	6,200,000,000

Moore's Law

Microprocessor Transistor Counts 1971-2011 & Moore's Law



Source Wikipedia



Key Concepts to Remember

- Five basic components of a computer
  - **input**, **output**, **memory**, **processor** (**datapath** + **control**)

- **Principle of abstraction**

  Help cope with design complexity by hiding low level details

- **Instruction set architecture**
  - Important abstraction interfaces hardware with low-level software