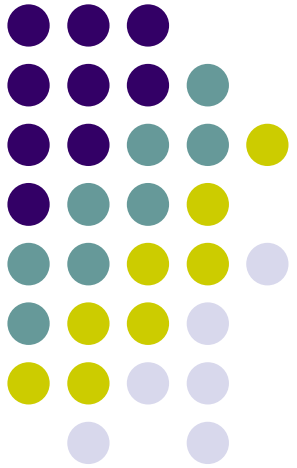
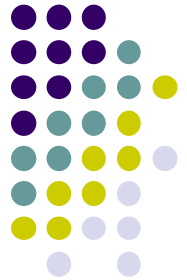
Computer Architecture

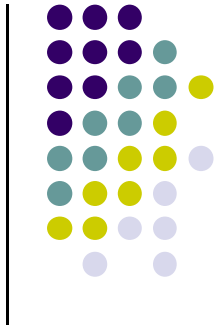
Assoc. Prof. Nguyễn Trí Thành, PhD
UNIVERSITY OF ENGINEERING AND TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS
ntthanh@vnu.edu.vn





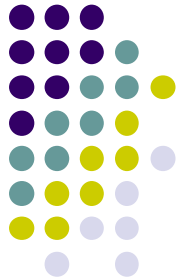
Introduction

Course objectives



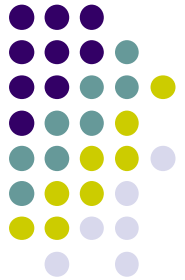
- Understand
  - what composes a computer
  - the main task of each component
  - how to make a CPU
  - how the CPU works
  - how to compile C program into machine language
  - how to optimize code
- Basic knowledge for embedded programming

The Computer Revolution



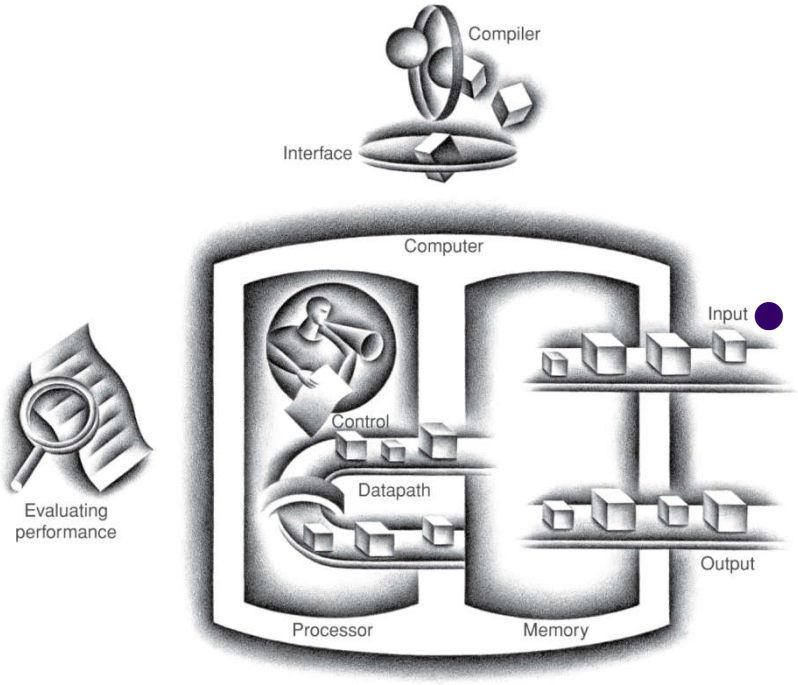
- Makes novel applications feasible
  - Computers in automobiles
  - Cell phones
  - World Wide Web
  - Search Engines
- Computers are pervasive

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

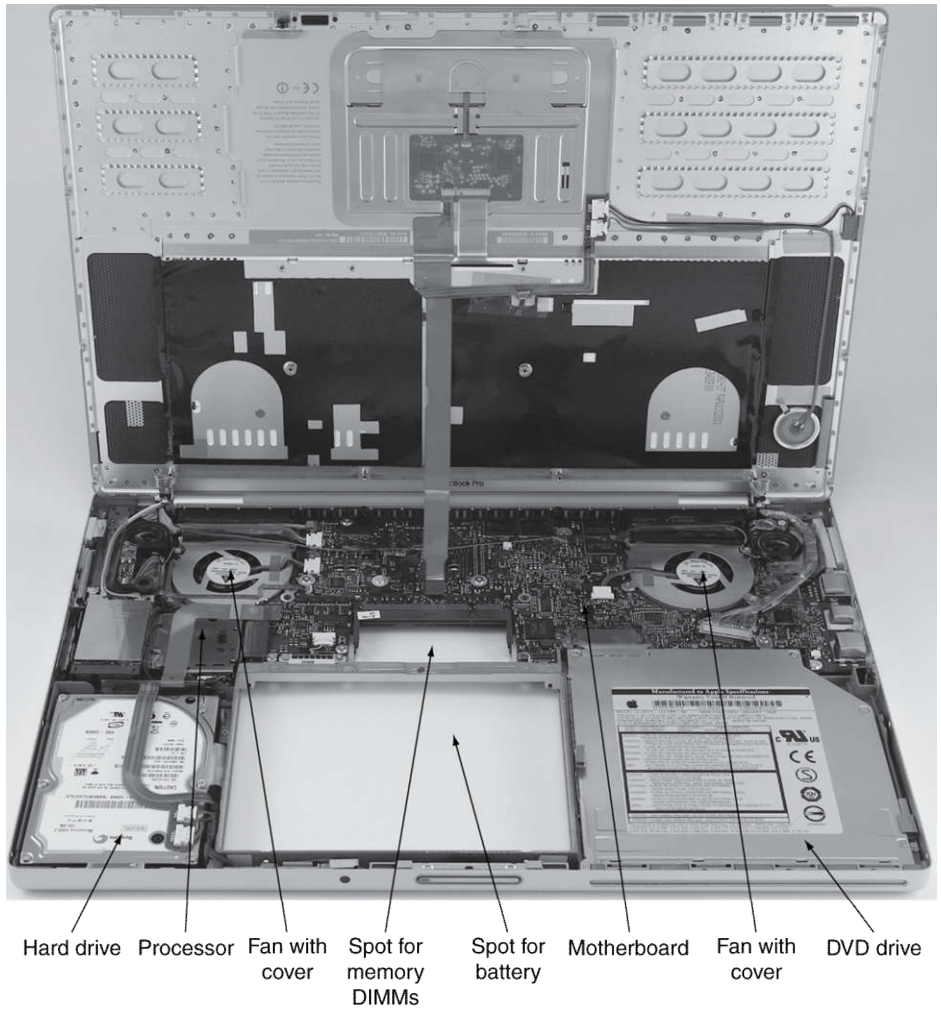
Components of a Computer

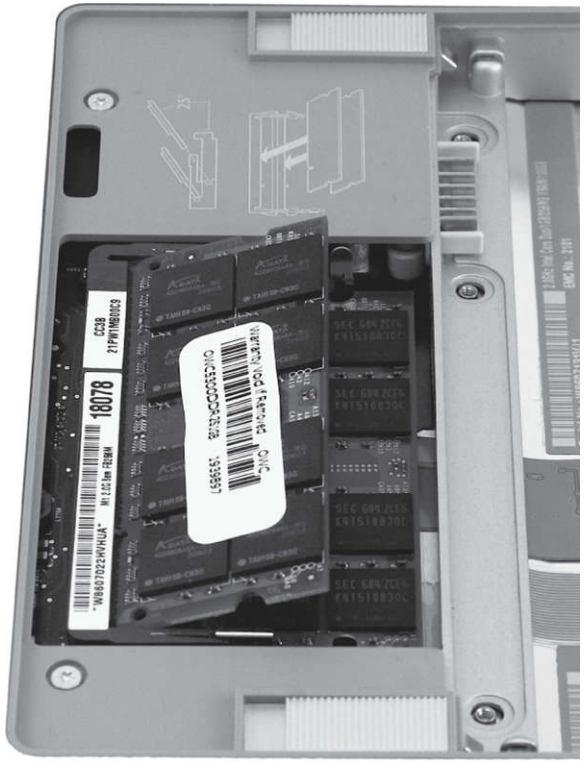


- Same components for all kinds of computer
  - Desktop, server, embedded
- Input/output includes
  - User-interface devices
    - Display, keyboard, mouse
  - Storage devices
    - Hard disk, CD/DVD, flash
  - Network adapters
    - For communicating with other computers

Anatomy of a Computer

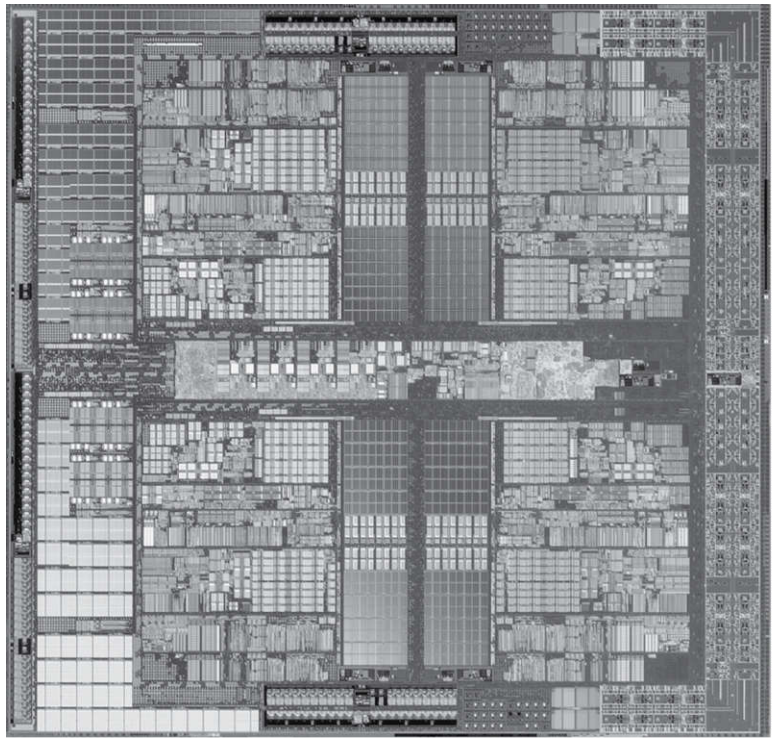
Opening the Box

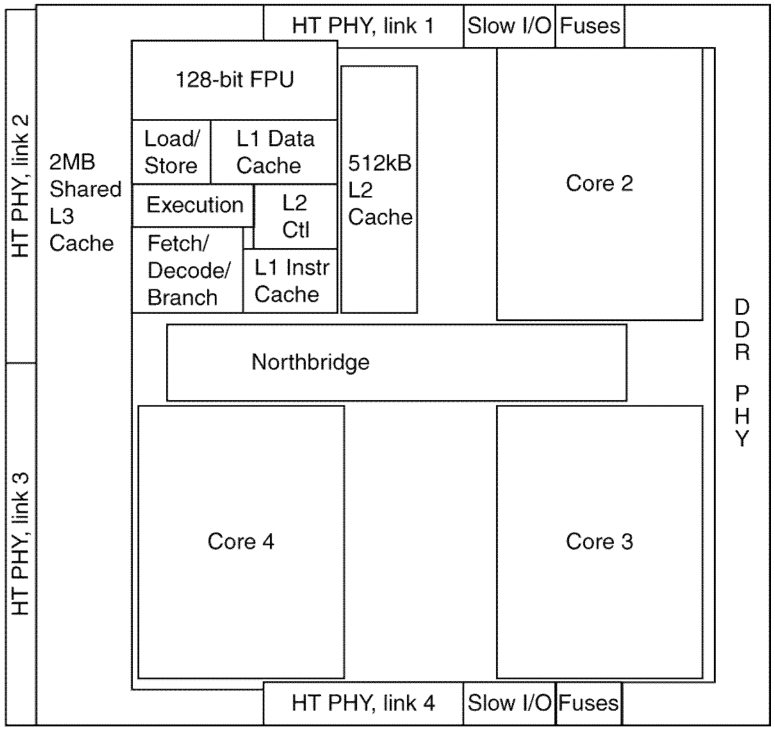




The Processor

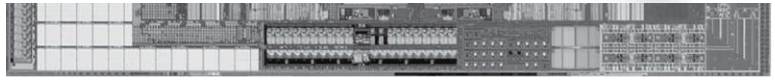
- AMD Barcelona: 4 processor cores

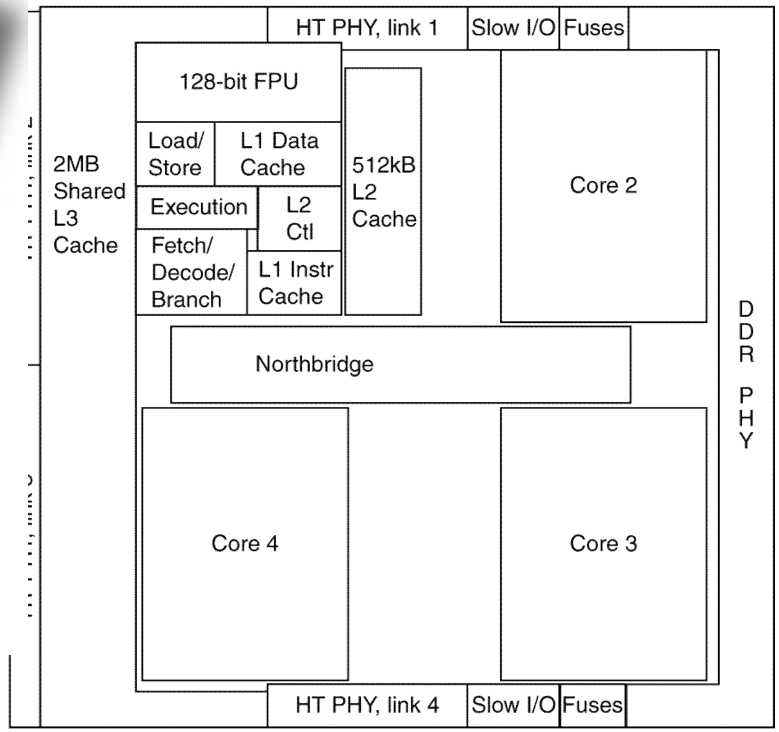




The Processor

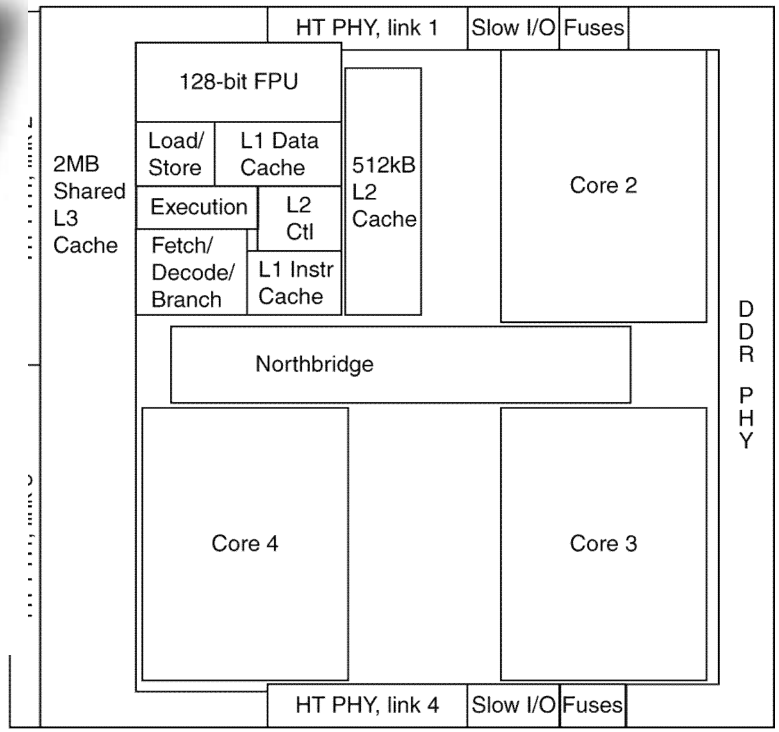
- ssor cores



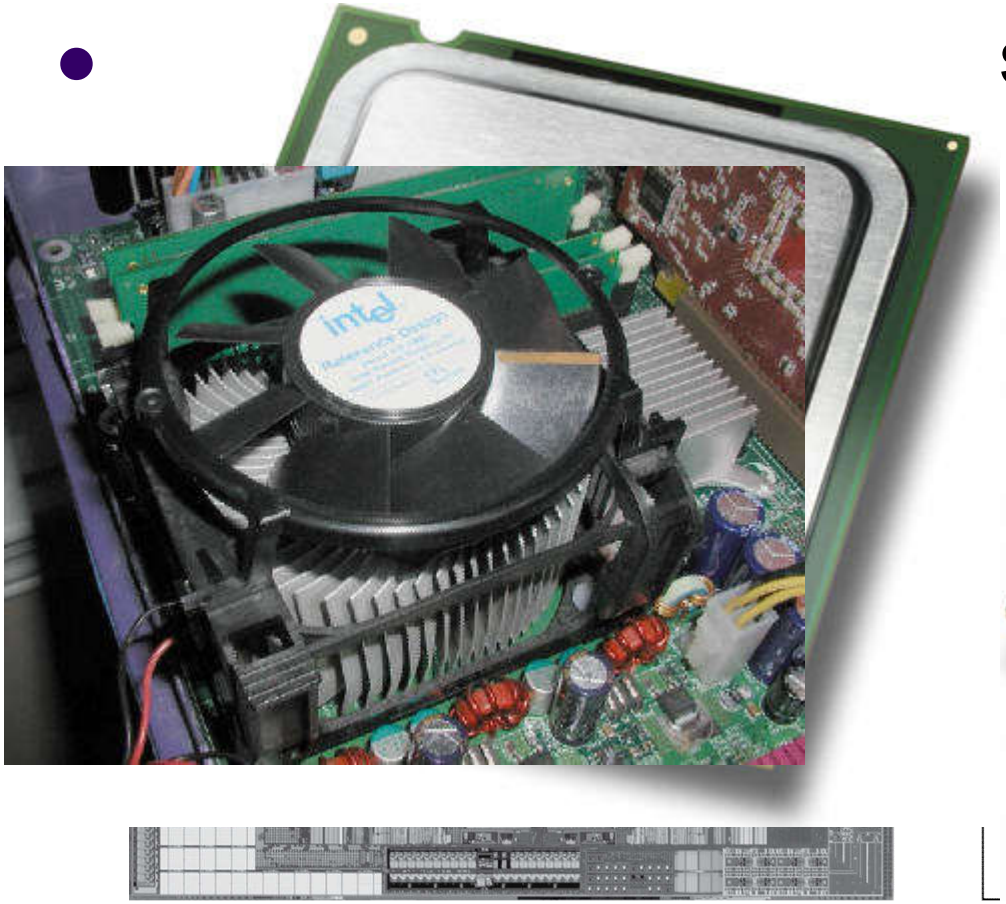


The Processor

- ssor cores



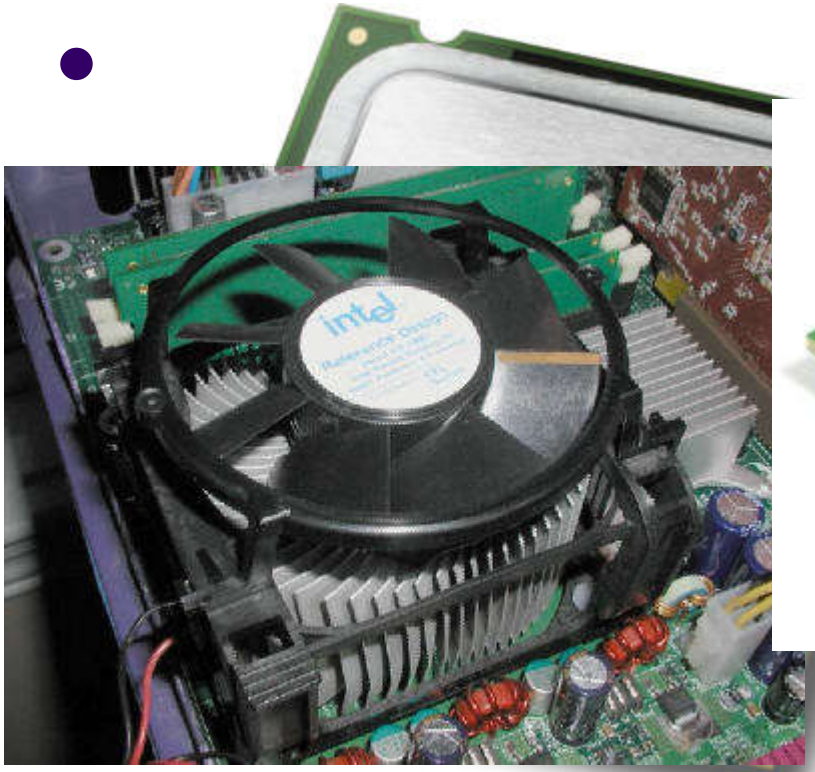
The Processor

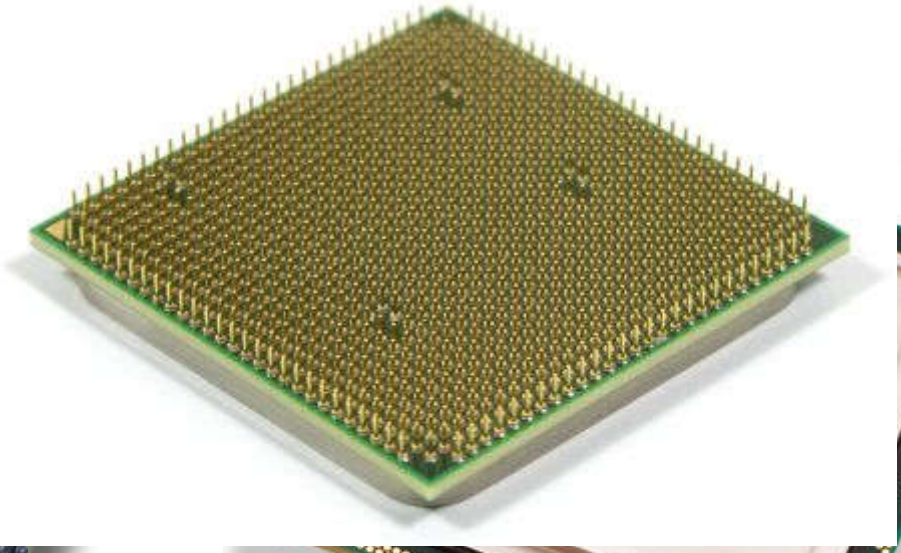
- 

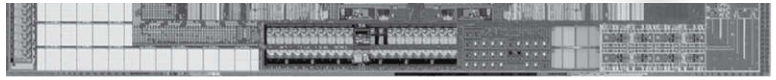
ssor cores

The Processor

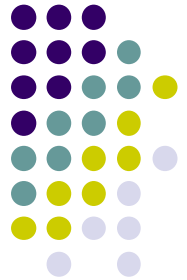
- ssor cores



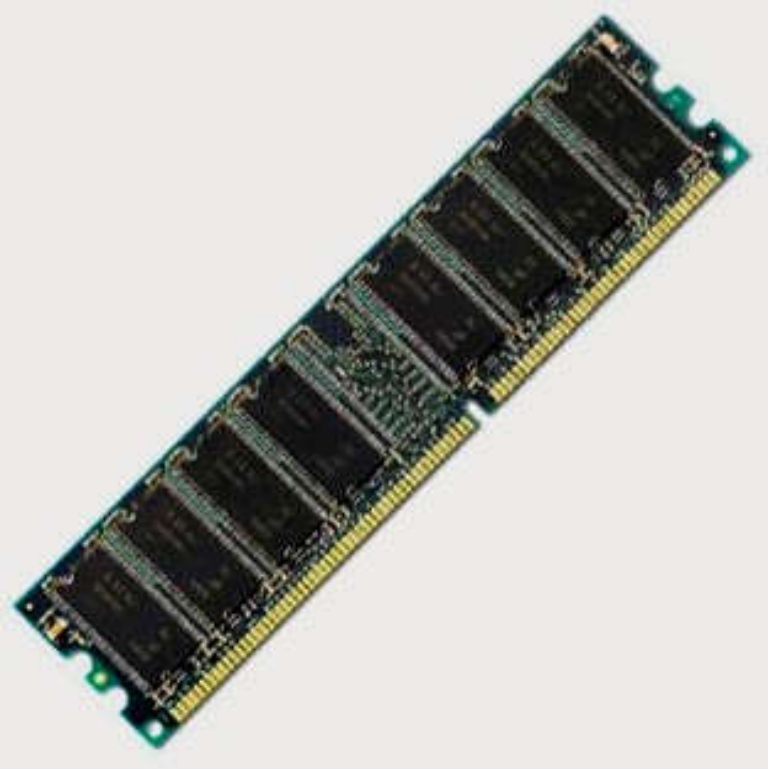




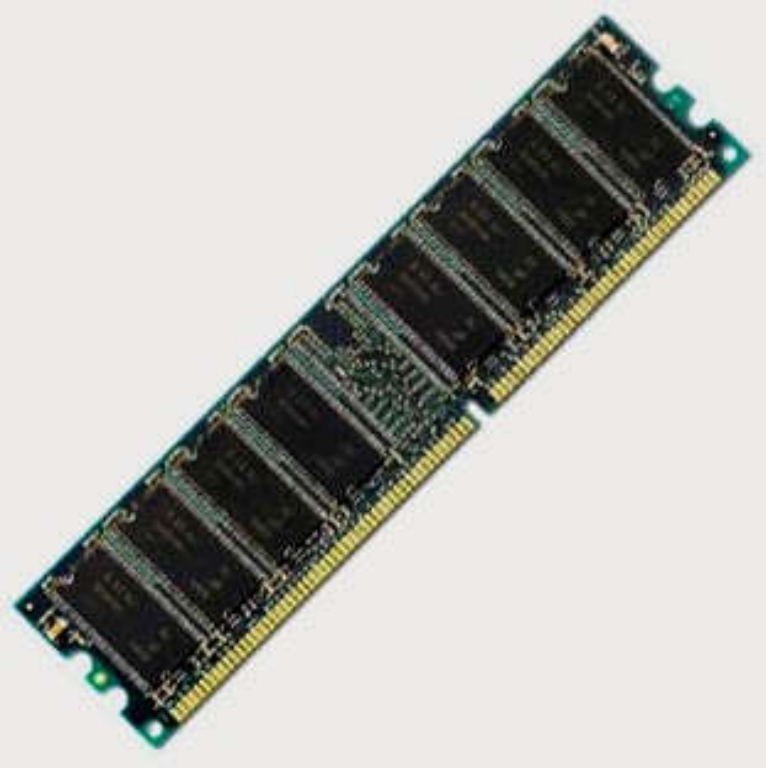
RAM (Random Access Memory)

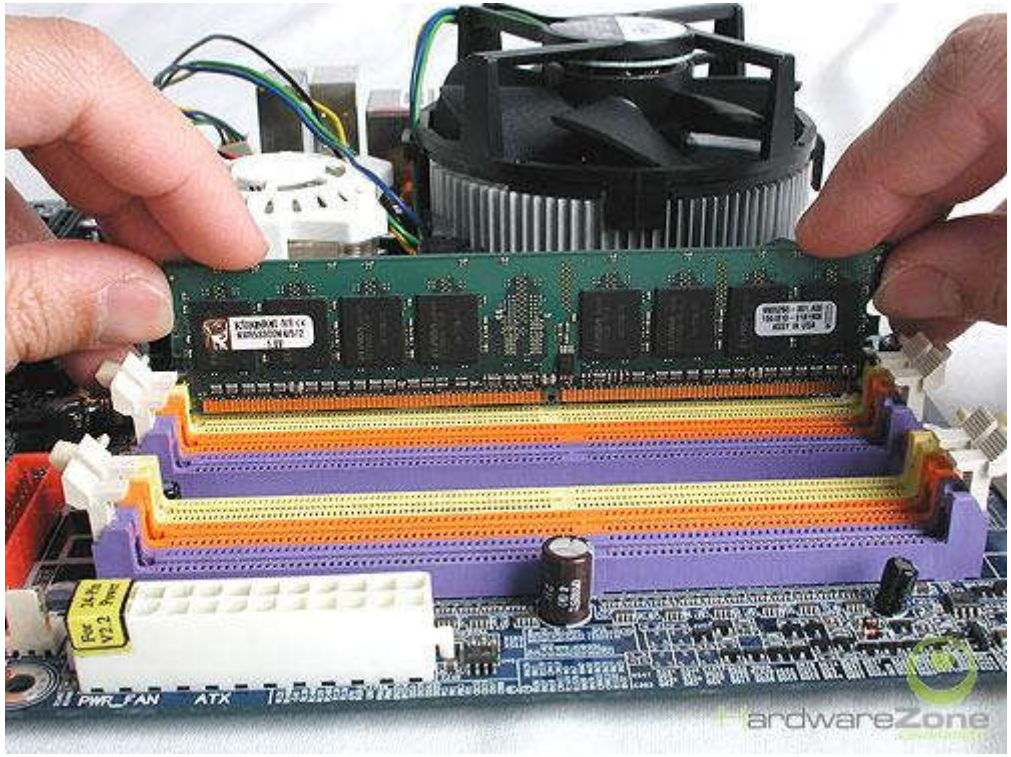


RAM (Random Access Memory)

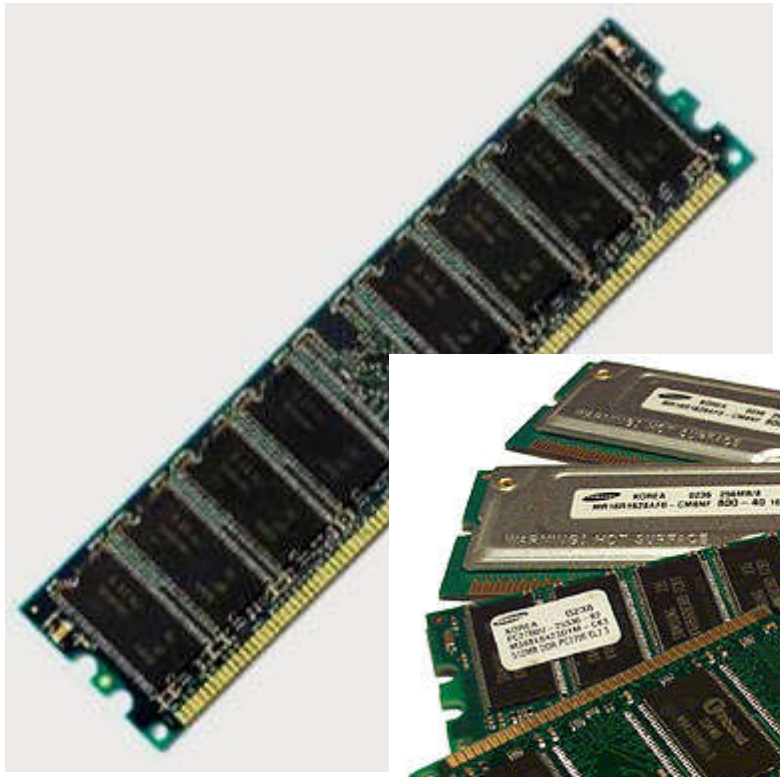


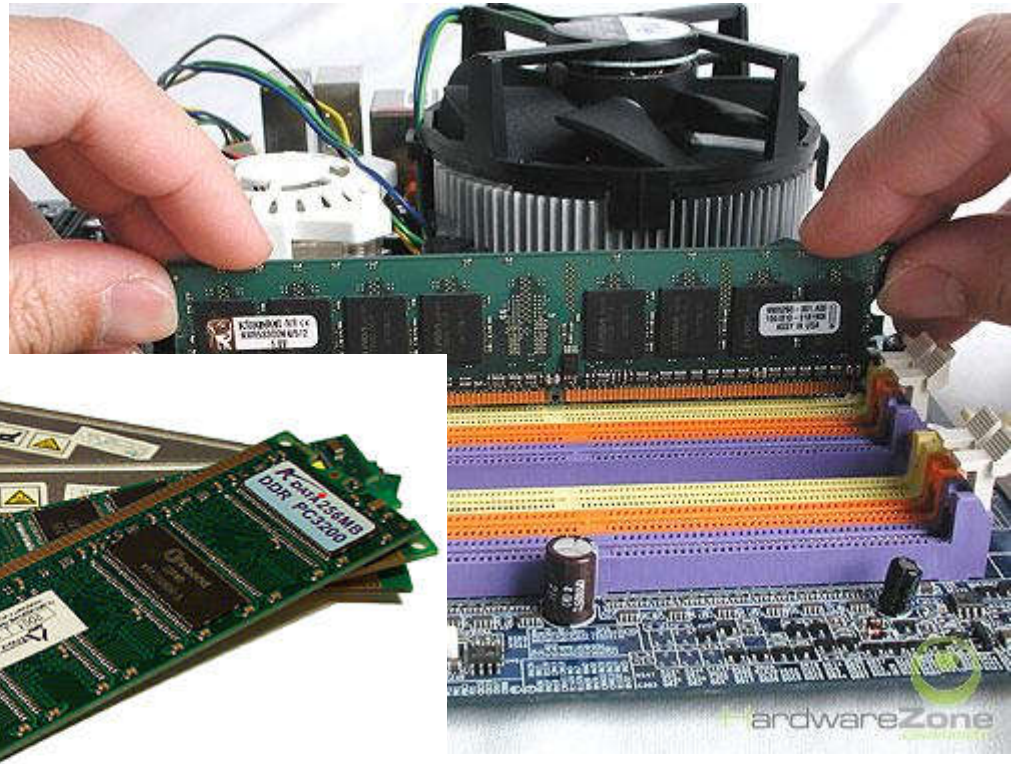
RAM (Random Access Memory)

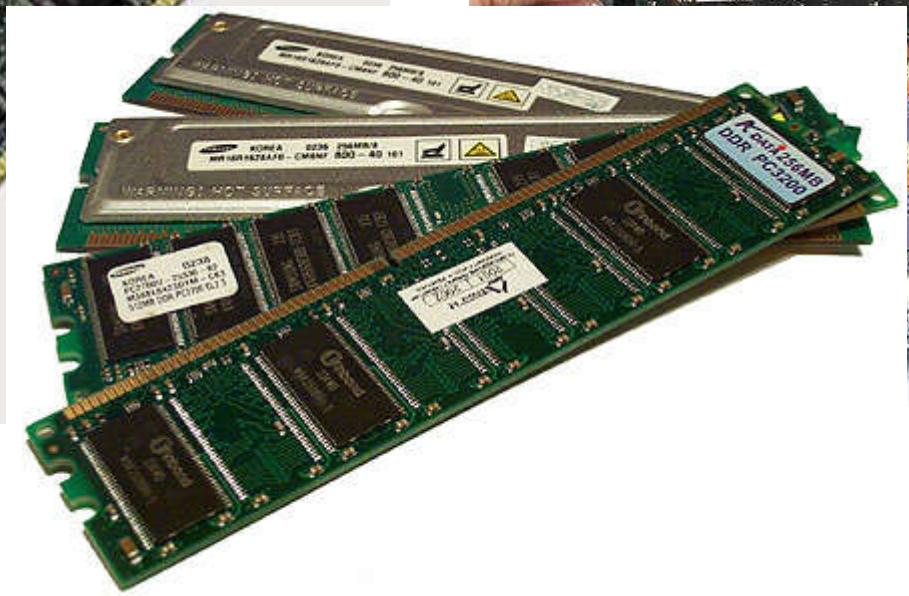




RAM (Random Access Memory)

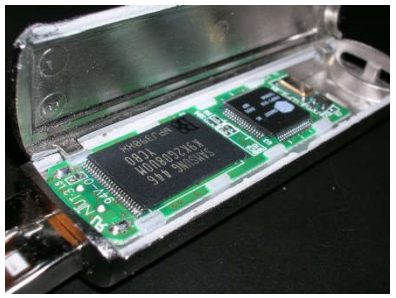




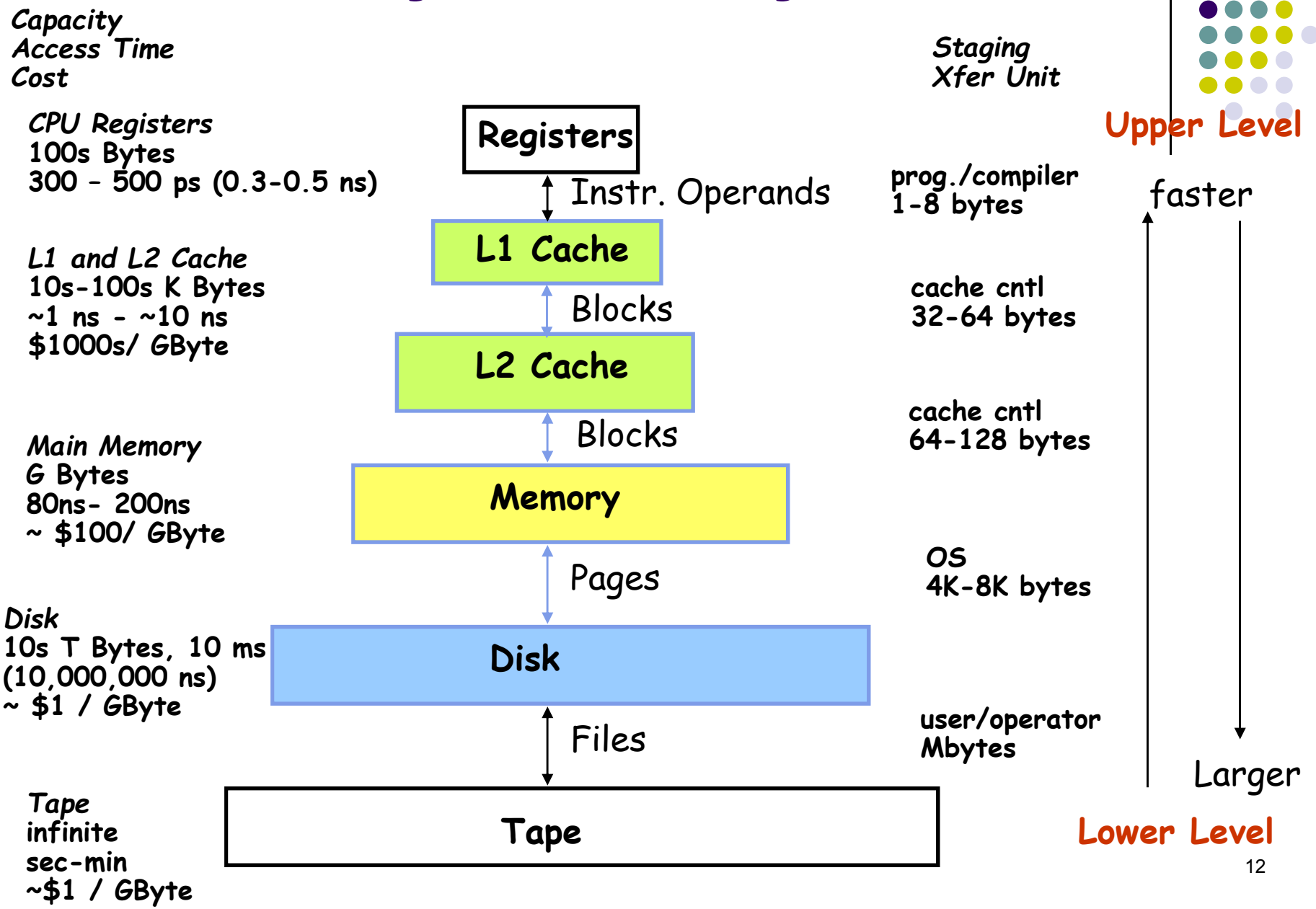


A Safe Place for Data

- Volatile main memory
  - Loses instructions and data when power off
- Non-volatile secondary memory
  - Magnetic disk
  - Flash memory

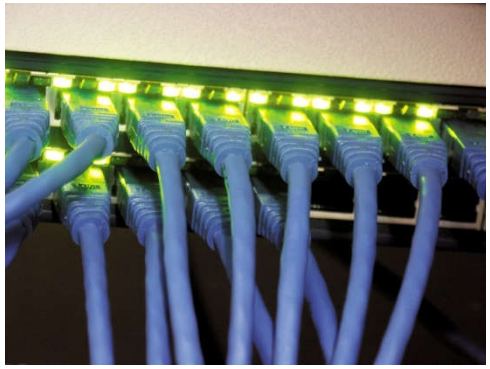


Memory Hierarchy Levels

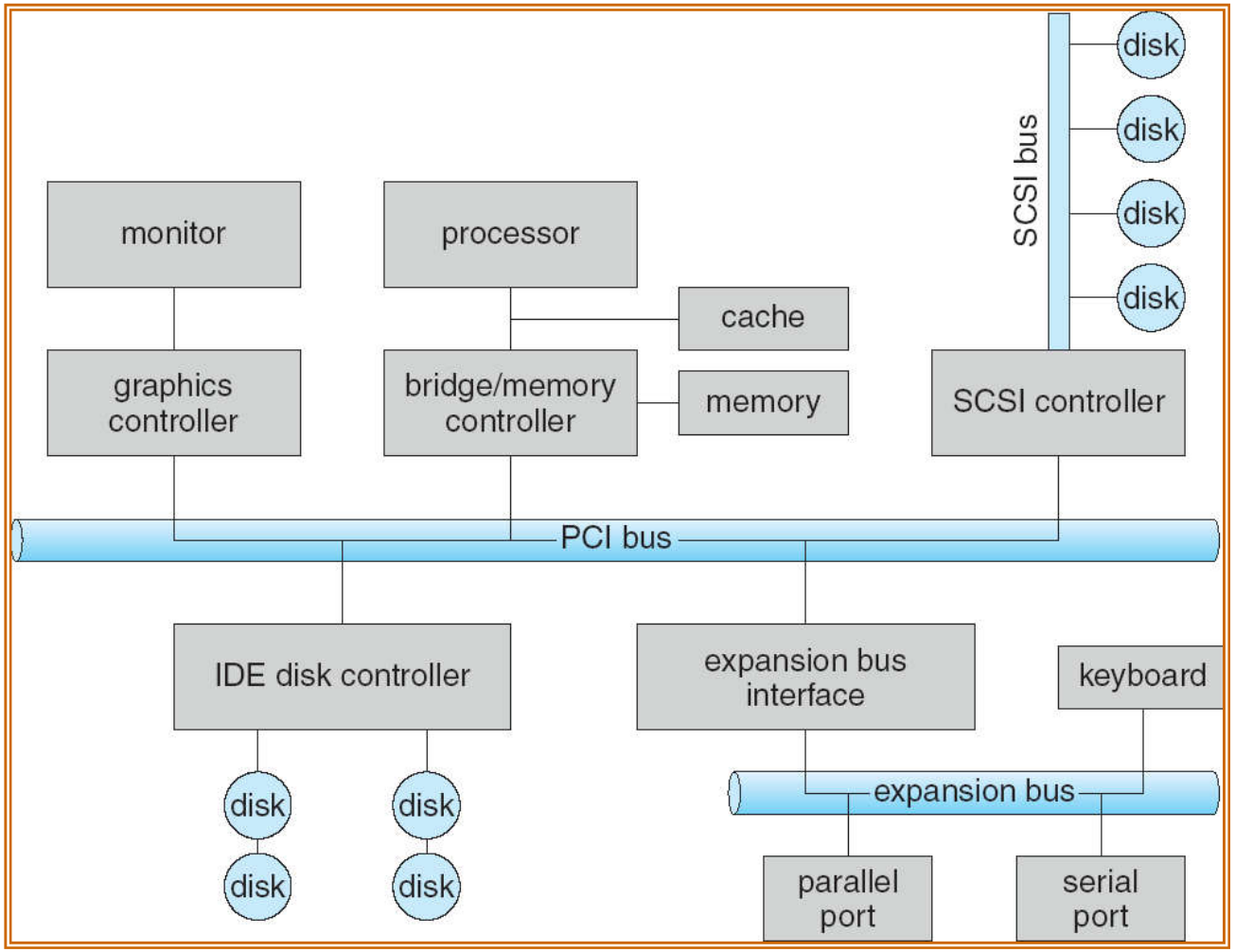


Networks

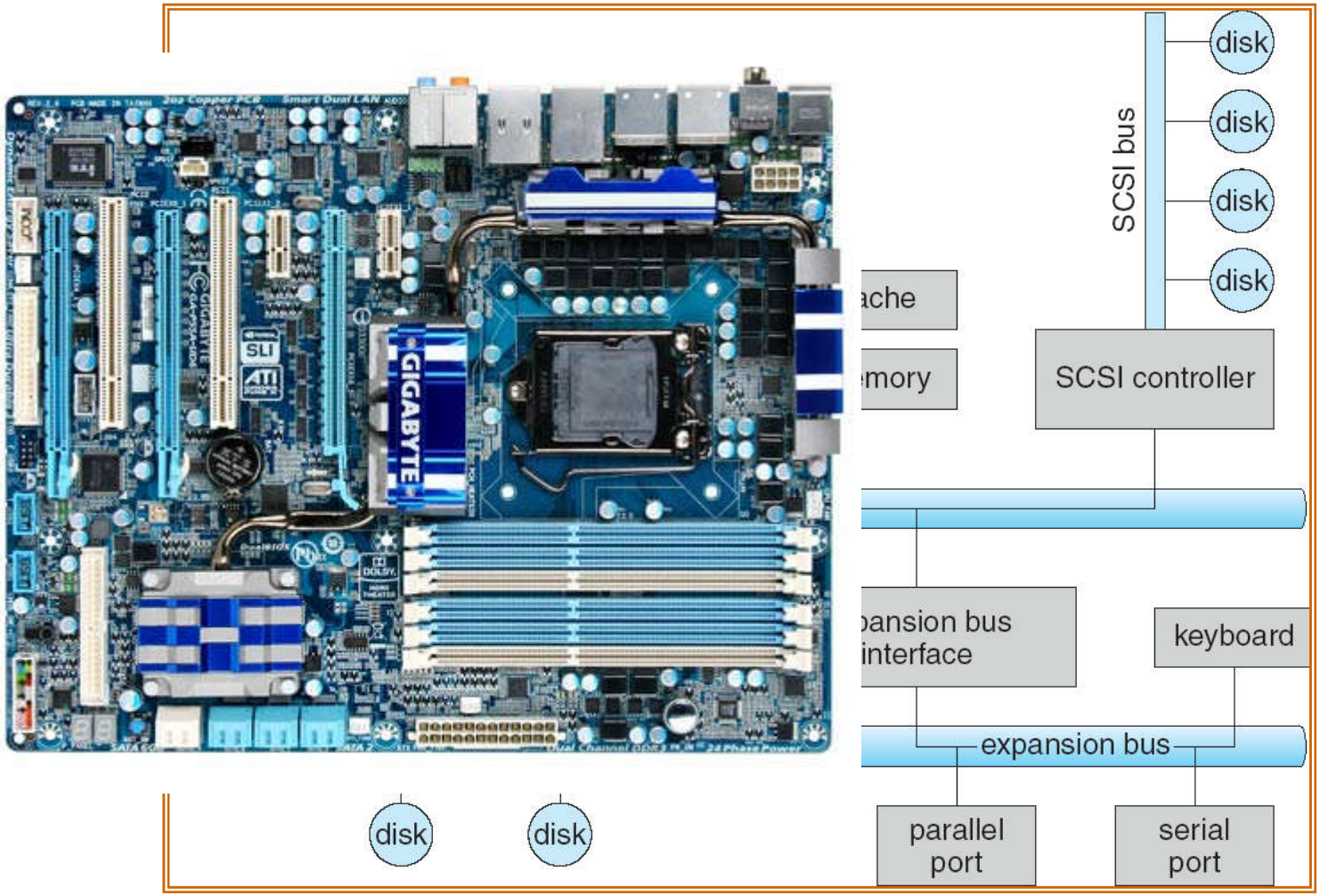
- Communication and resource sharing
- Local area network (LAN): Ethernet
  - Within a building
- Wide area network (WAN: the Internet
- Wireless network: WiFi, Bluetooth



Inside structure

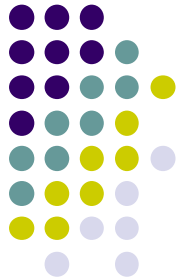


Inside structure



Inside structure

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

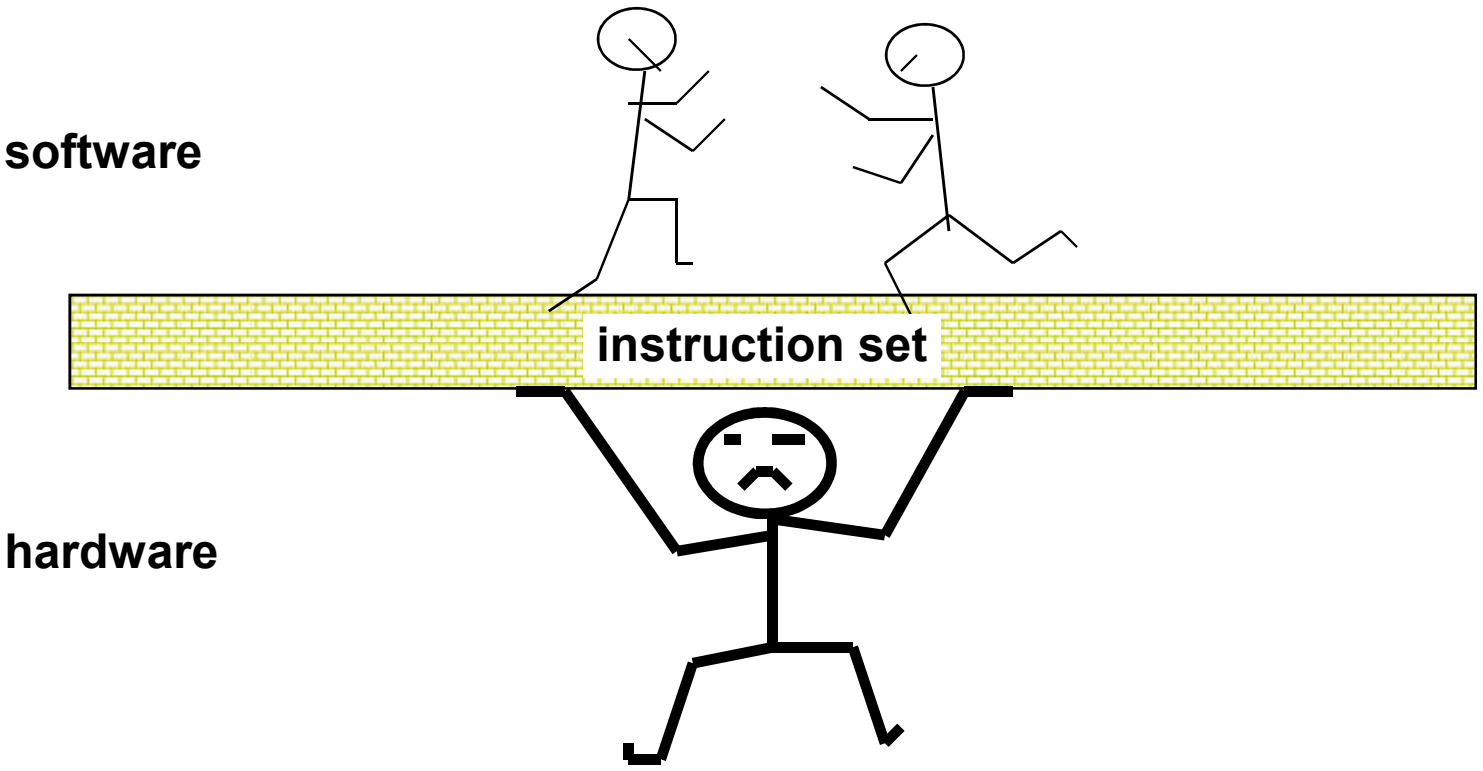
What is Computer Architecture? Easy Answer

Computer Architecture =

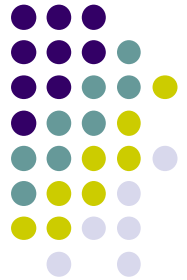
Instruction Set Architecture +

Machine Organization

The Instruction Set: a Critical Interface



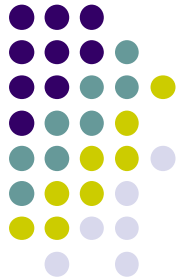
Instruction Set Architecture



- A very important abstraction:
  - *interface* between hardware and low-level software
  - *standardizes* instructions, machine language bit patterns, etc.
  - advantage: *allows different implementations of the same architecture*
  - disadvantage: *sometimes prevents adding new innovations*

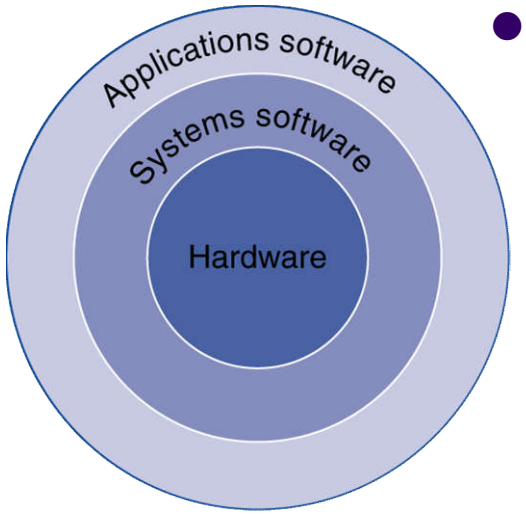
- Modern instruction set architectures:
  - 80x86/Pentium/K6, PowerPC, DEC Alpha, MIPS, SPARC, HP

What You Will 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

Below Your Program



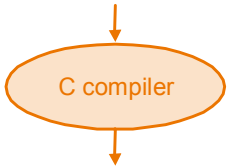
- Application software
  - Written in high-level language
- System software
  - Compiler: translates HLL code to machine code
  - Operating System: service code
    - Handling input/output
    - Managing memory and storage
    - Scheduling tasks & sharing resources
- Hardware
  - Processor, memory, I/O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

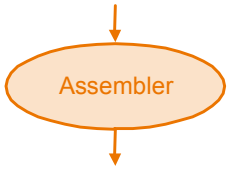
High-level language program (in C)

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



Assembly language program (for MIPS)

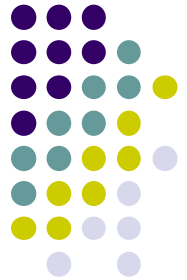
```
swap:
     muli $2, $5,4
     add  $2, $4,$2
     lw   $15, 0($2)
     lw   $16, 4($2)
     sw   $16, 0($2)
     sw   $15, 4($2)
     jr   $31
```



Binary machine language program (for MIPS)

```
00000000101000010000000000011000
00000000100011110000110000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```

Quiz?



- How can computers play audio files?
- How can they understand characters?