CSE 2021
Computer Organization

Hugh Chesser, CSEB
1012U

W1-W

Agenda

- Introduction to course
- High level language versus Assembly language versus Machine Language
- Categorization of Software: Applications, Systems, Hardware
- Components of a Computer: Input, Output, Memory, Control, and Datapath
- Integrated Circuits (IC's)

Reading: Patterson, Sections 1.1 – 1.3.

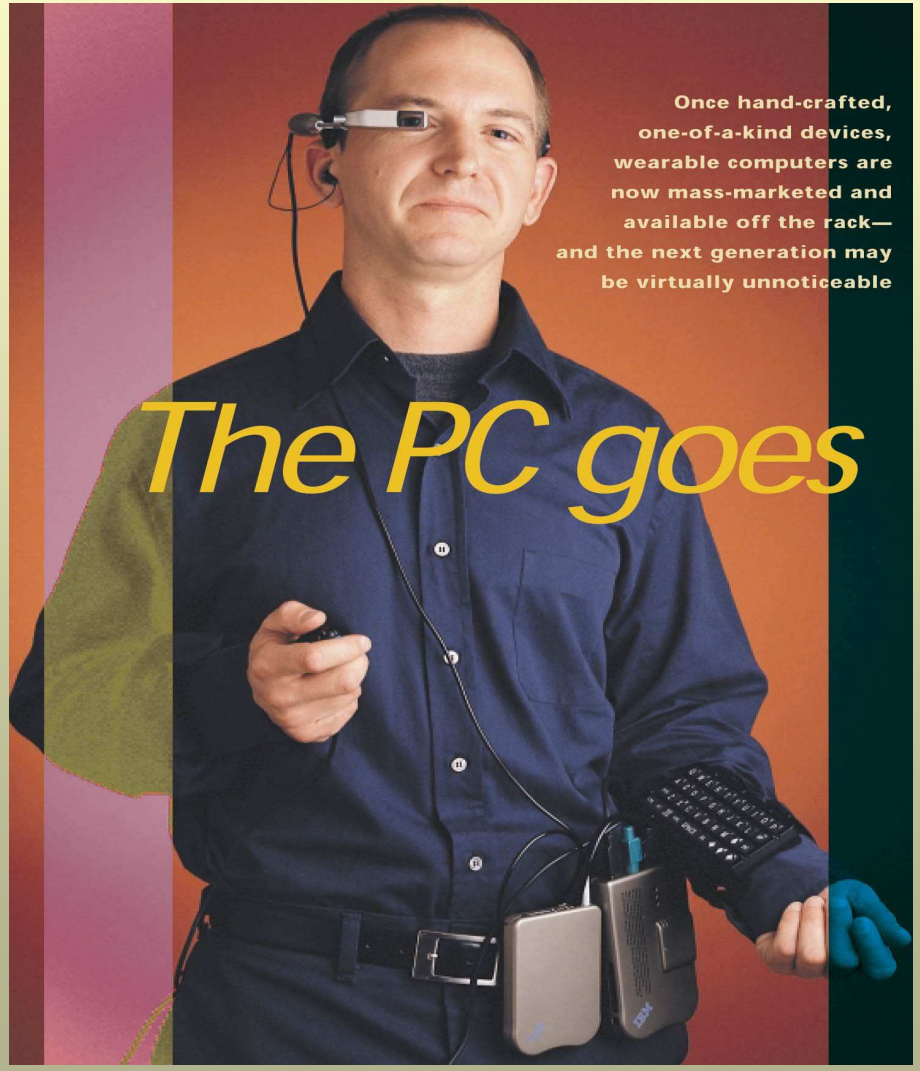
CSE 2021: Computer Organization
Section E

Course Instructor:	Hugh Chesser	
Teaching Assistants:	TBA	
Contact Information:	*Instructor* Office: CSB 1012U chesser@yorku.ca (416) 736-2100 X20760	*Teaching Assistants* TBA
Course URL:	http://www.cse.yorku.ca/course_archive/2009-10/F/2021/	
Text:	D. A. Patterson and J. L. Hassey, *Computer Organization and Design*, San Francisco, CA: Morgan Kaufmann Publishers, Inc., 4th edition (2008)	
Class Schedule:	MW 17:30 – 19:00, Room R S203	
Office Hours:	*Instructor*: CSEB 1012U, By appointment *Teaching Assistants*: TBA	
Laboratory:	CSE 2004, SPIM simulator is freeware, downloadable to PC's.	
Assessment:	Quizzes: 10% (Best 2 out of 3 counted) Lab Exercises: 35% (Your higher scoring 7 out of 8 labs at 5% each) Mid-term Exam: 20% Final Exam: 35%	

Future Applications of Computers (1)

Wearable Computers

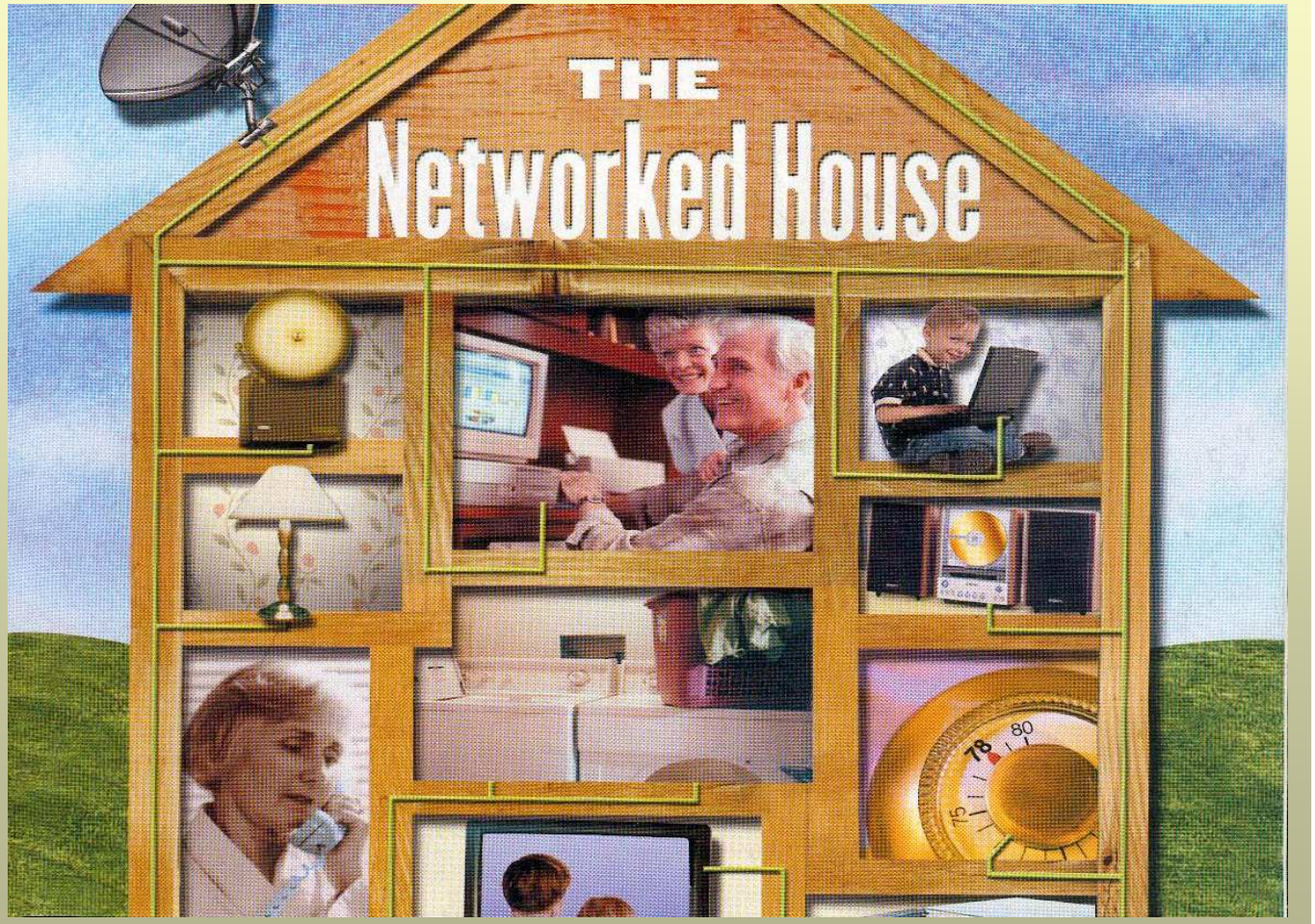
Next generation of computers that are cheaper, portable, and faster with added storage



Future Applications of Computers (2)

World Wide Web/ Computer Networks

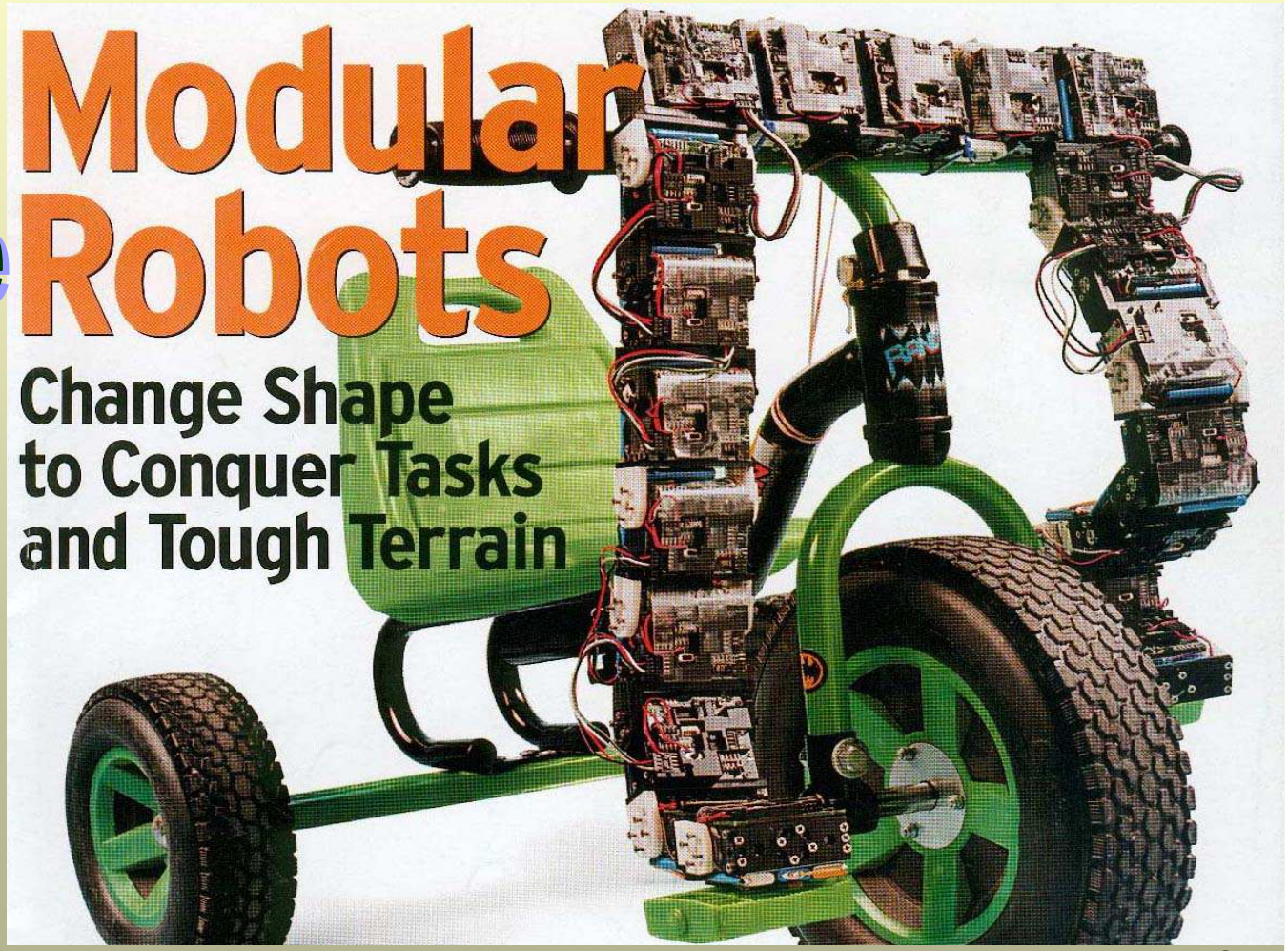
Networking and telecommunications to provide interconnectivity between electronic devices



Future Applications of Computers (3)

Robotics & Artificial Intelligence

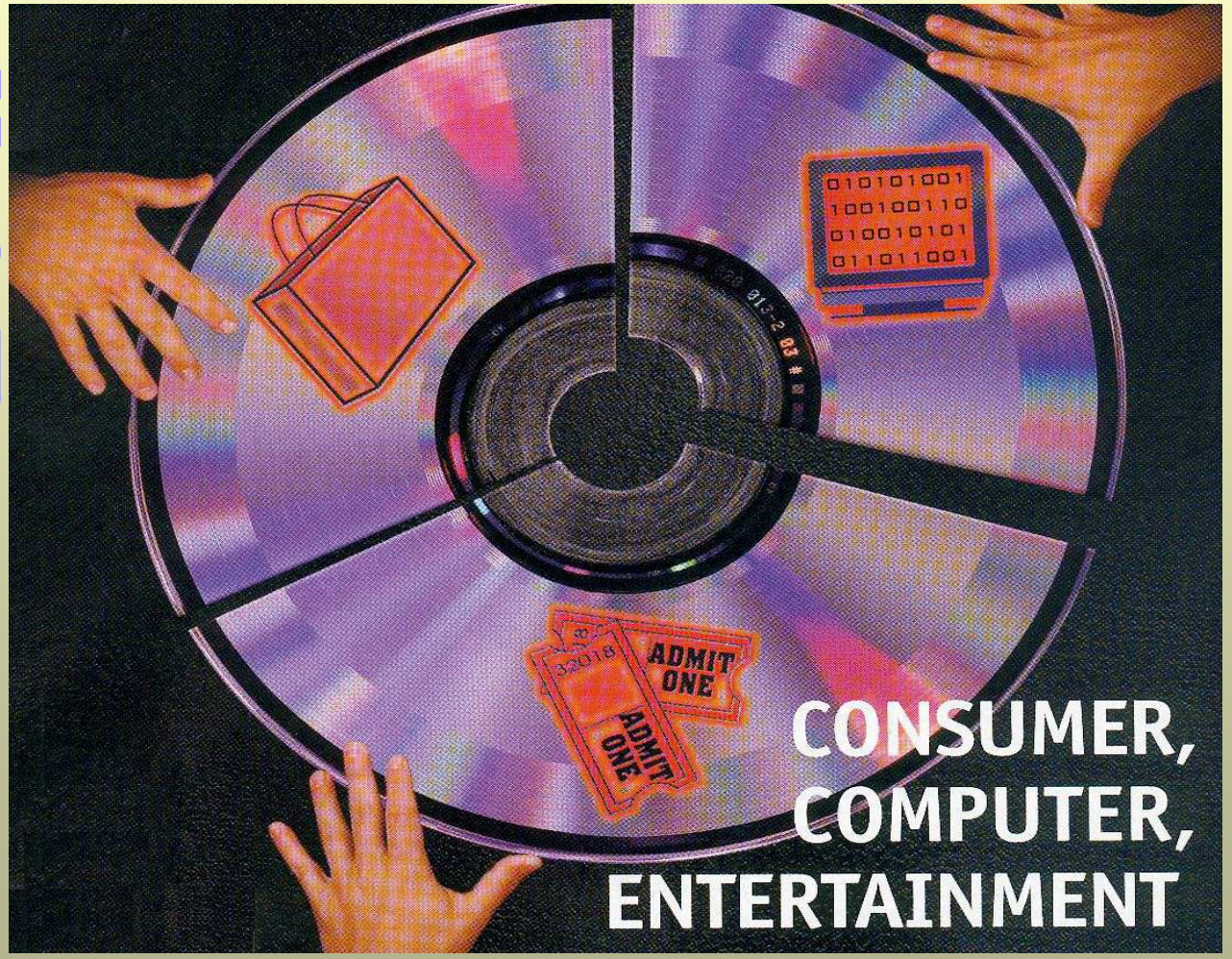
Intelligent and autonomous electromechanical systems for completing repetitive functions



Future Applications of Computers (4)

Multimedia: Systems, Standards & Networking

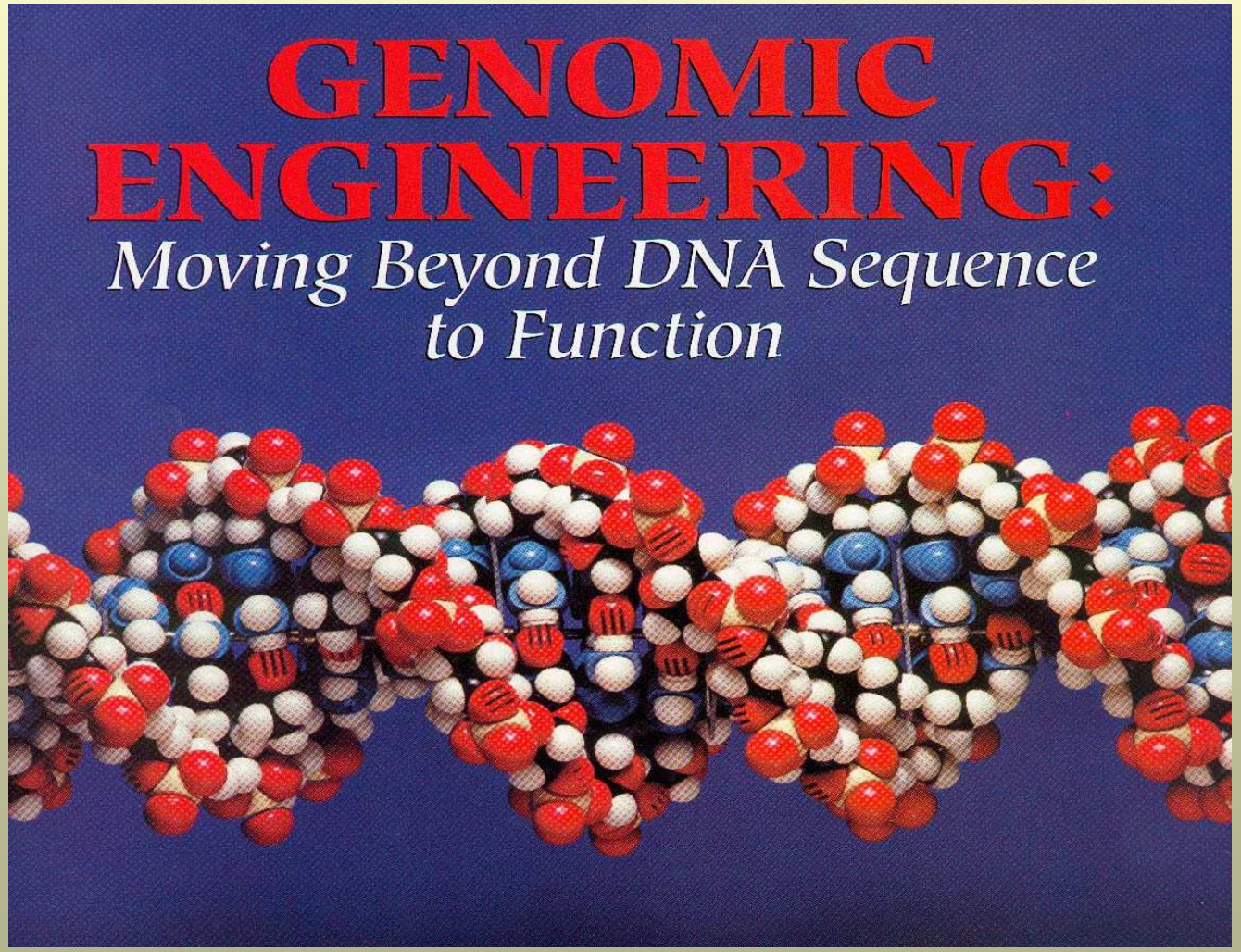
Novel multimedia applications, e.g., video-on-demand, multiple-view-TV, and VR based complex simulation systems.



Future Applications of Computers (5)

Bioinformatics / Genetics Engg.

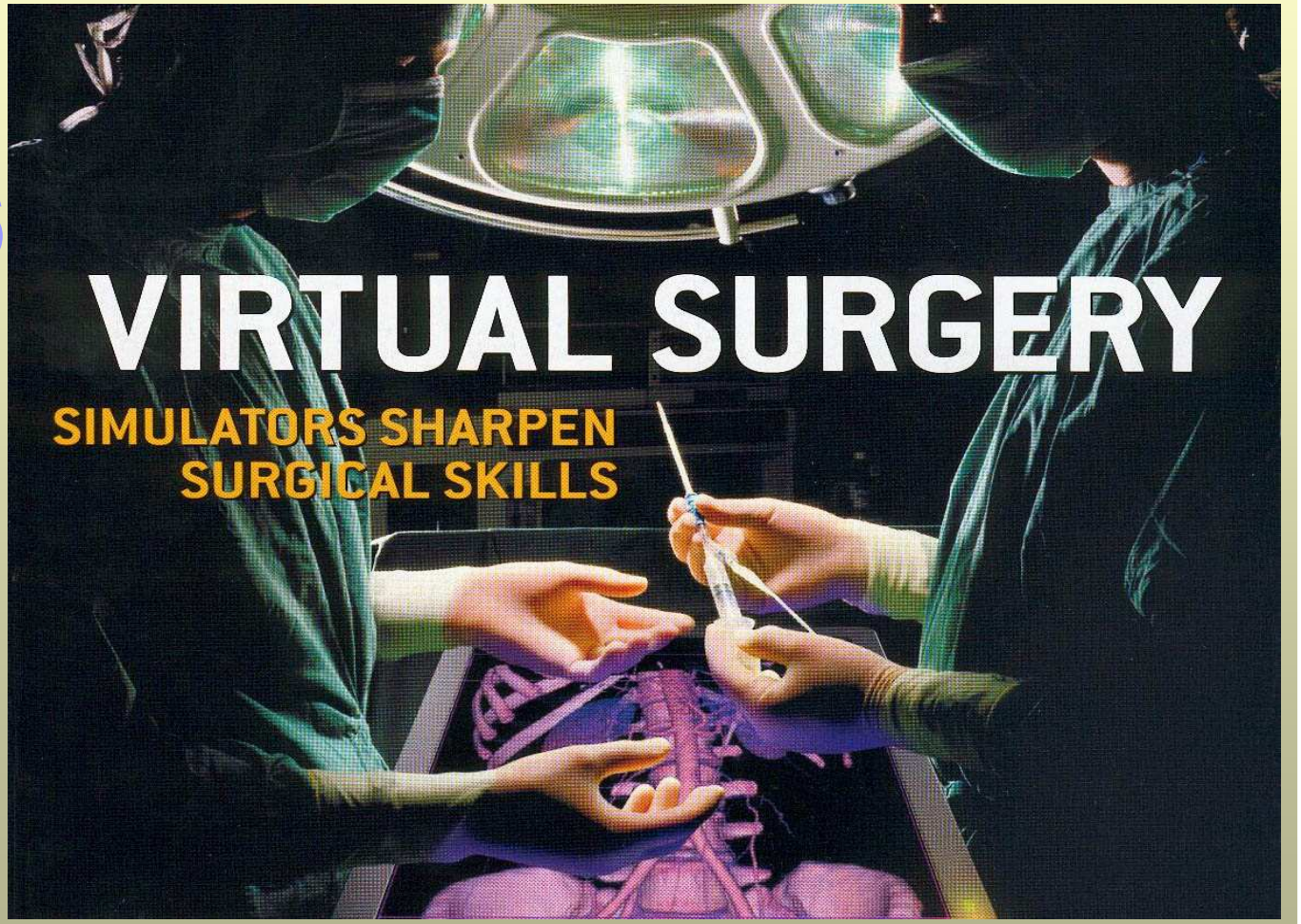
Analyze large biological (genetic data sets) to understand and control human evolution



Future Applications of Computers (6)

Virtual Reality / Computer Graphics

Design virtual immersive environments to simulate reality



Future Applications of Computers (7)

"Had the transport industry kept pace with the computer industry, today we would travel coast to coast in 5 seconds for about 50 cents !" (Patterson, 1998)

What is CSE 2021 about?

The course explains what is inside a computer, describing its hardware (HW), and introducing the assembly language representation of a program compiled from a high level language such as ANSI C.

You will learn:

1. How computers work?
2. How to analyze their performance?
3. How to code directly in MIPS?
4. What are the issues affecting modern processors (e.g. caches, pipelines)?

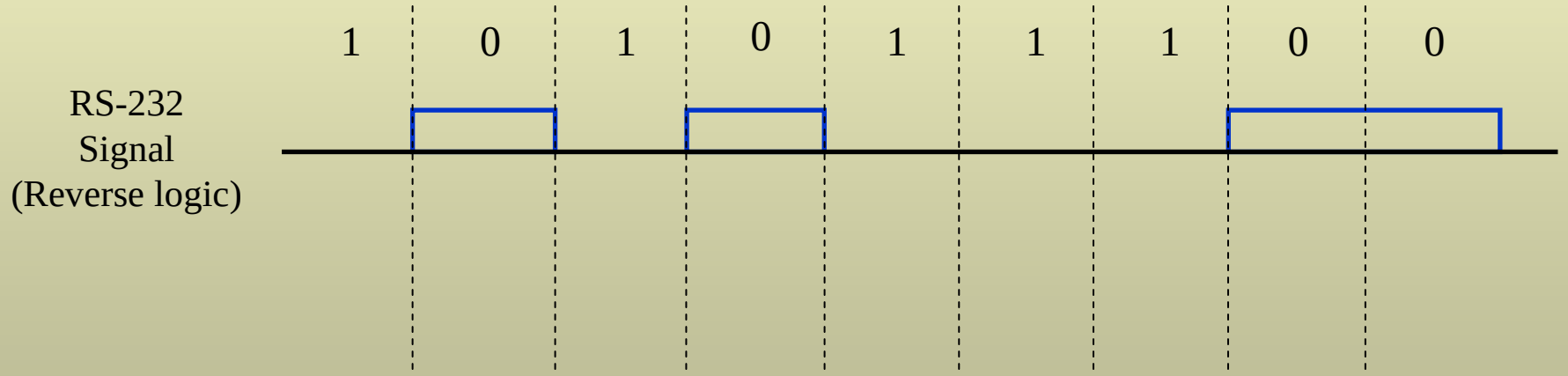
Why do I learn this stuff?

1. To build *better* software people use (improved performance)
2. To offer *expert* advice in applications, purchasing, etc.

Binary Digits (Bits)

— Communication between different components of a computer takes place in terms of *on* and *off* electrical signals

— Symbols used to represent these electrical states are the numbers 1 and 0; binary digit 1 corresponding to high voltage and binary digit 0 corresponding to low voltage



— All operations and data inside a computer are expressed in terms of the binary digits or bits

Instructions

- Instructions: are commands given to a computer to perform a particular task.

  Example: *Addition of variables A and B*

  High Level Language: `(A + B)`

  Binary notation for the add operation: `100011001010000`

- Binary machine language program: is a one-to-one binary representation of a program written in a high level language.
- Clearly, binary machine language programs are tedious to write and debug.
- Instead a symbolic notation is used as an intermediate step between the high level language and its binary representation. This symbolic notation is referred to as the assembly language.

  Example: *Addition of variables A and B*

  High Level Language: `(A + B)`

  Assembly Language: `add A,B`

  Binary notation for the add operation: `100011001010000`

Levels of Programming

Why use High-level Language?

1. Ease in writing & debugging
2. Improved productivity
3. HW independence

Compiler: converts a program written in high-level language
into its equivalent symbolic assembly language representation.

Assembler: translates
assembly language into the binary machine language.

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 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
```

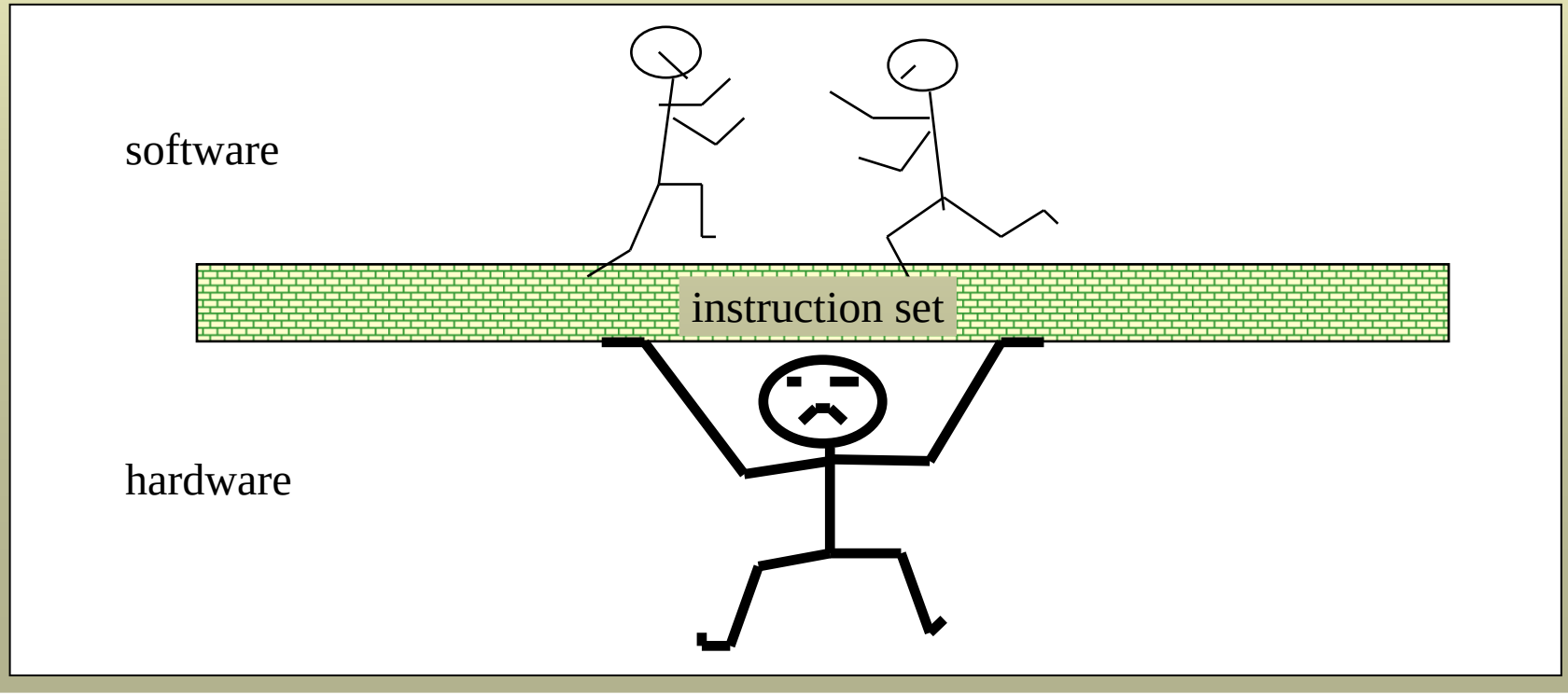
Assembler

Binary machine language program (for MIPS)

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

Instruction Set (1)

- Computer Architecture = Instruction Set Architecture + Machine Organization
- Machine Organization: Ways in which different computer components (Registers, ALU, Shifters, Logic Units, ...) are interconnected.
- Recall instructions are commands given to a computer to perform a particular task.
- Instruction Set: is a collection / library of instructions that a computer can execute.



Instruction Set (2)

- Programs written for a computer can only use the instructions provided in its instruction set.
- Examples of modern instruction set architectures (ISA's):
  1. 80x86/Pentium/K6/MMX (Intel, 1978-96)
  2. DEC Alpha (Digital,1992-97) v1, v3
  3. MIPS (SGI, 1986-96) I, II, III, IV, V
  4. SPARC (Sun, 1987-95) v8, v9
  5. RISC (HP/IBM, 1986-96) v1.1, v2.0
- Instructions in the MIPS instruction set can be divided in five categories:
  1. Arithmetic operations: `add,sub(subtract),mult(multiply),div (division),etc.`
  2. Logical operations: `and, or, sll` **`(shift left logical), etc.`**
  3. Data Transfer: `lw (load), sw (save), etc.`
  4. Conditional branch: `beq(branch if equal), slt set if less than), etc.`
  5. Unconditional branch: `j (jump), etc.`

Question: Will the ISA developed on one machine be compatible to another machine of a different manufacturer?

Categorization of Software (1)

— Software can be categorized in different categories:

1. Systems SW: provides commonly useful services to the hardware, e.g., operating systems, compilers, and assemblers.
2. Applications SW: are programs / packages used by the computer users, e.g., Excel sheet, Emacs / vi text editor.

— Such a simplified view of software has certain problems, e.g., how to categorize compilers that produce assembly language programs for both applications and systems software.

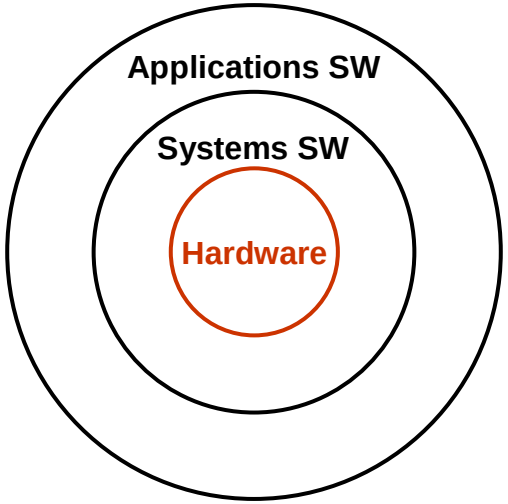
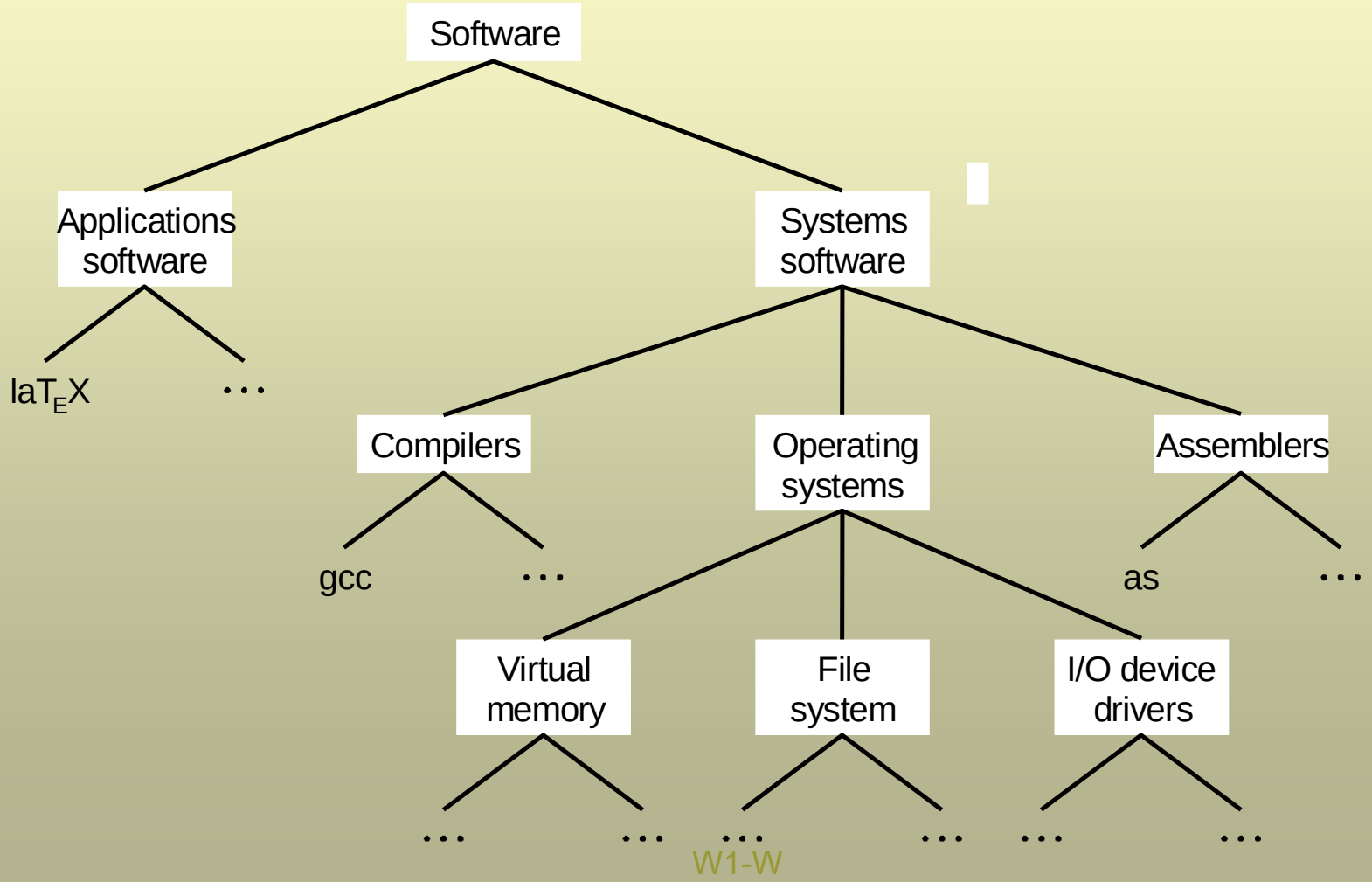


Fig. 1.2

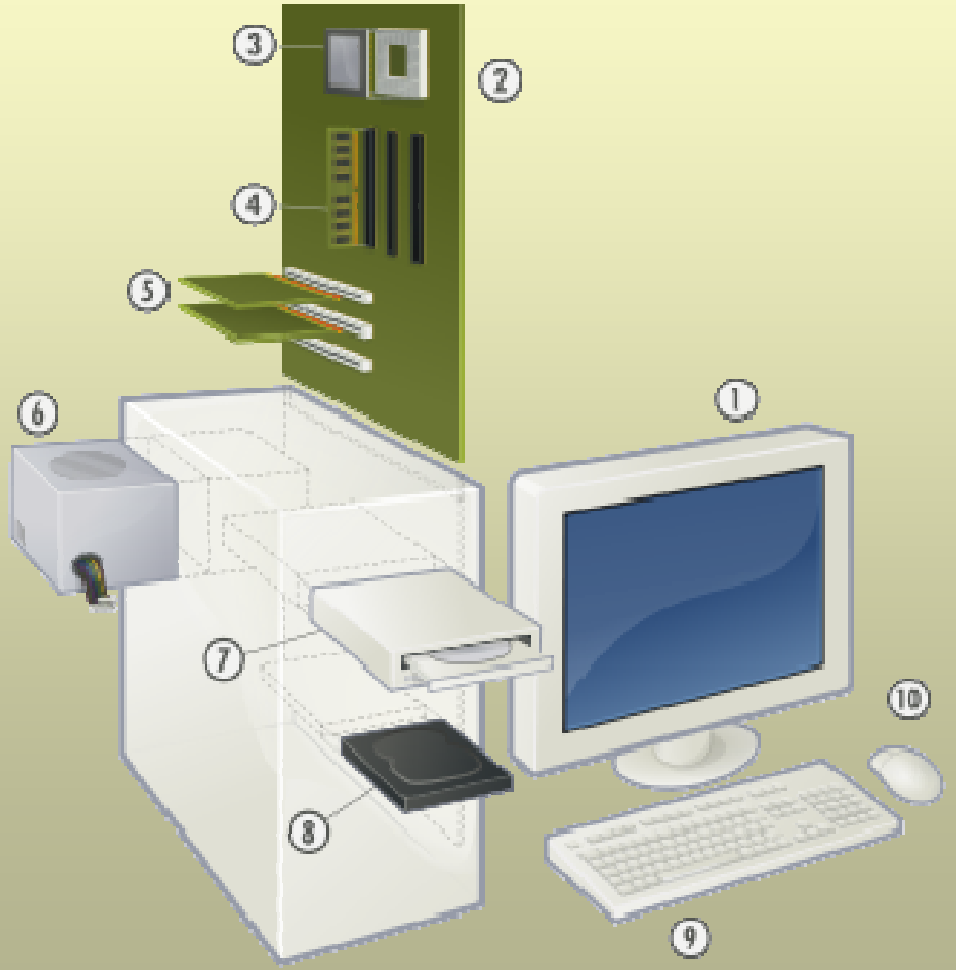
Categorization of Software (2)

— More realistic categories for software are shown below:



Computer Architecture

Hardware Elements: Computer, Monitor, Keyboard, Mouse, Network, …



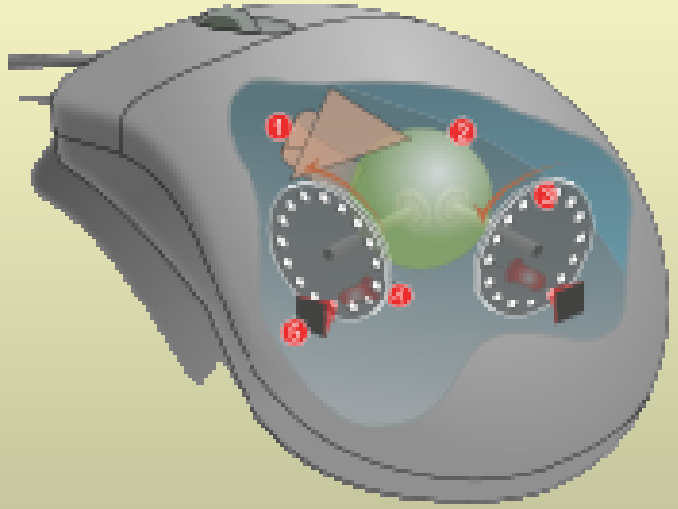
Input Devices

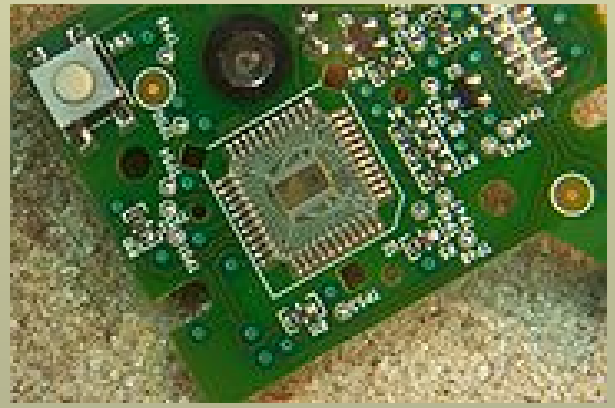
Keyboard, Mouse, Joystick, Touch Screen, Stylus, Graphic Tablet, …

Operation of a mechanical Mouse:

1. Consists of a small ball
2. Ball has contacts with 2 wheels on the *x* and *y* axis
3. Each wheel increments or decrements a counter
4. Counters records how far the mouse has moved
5. Cursor on the screen is moved accordingly

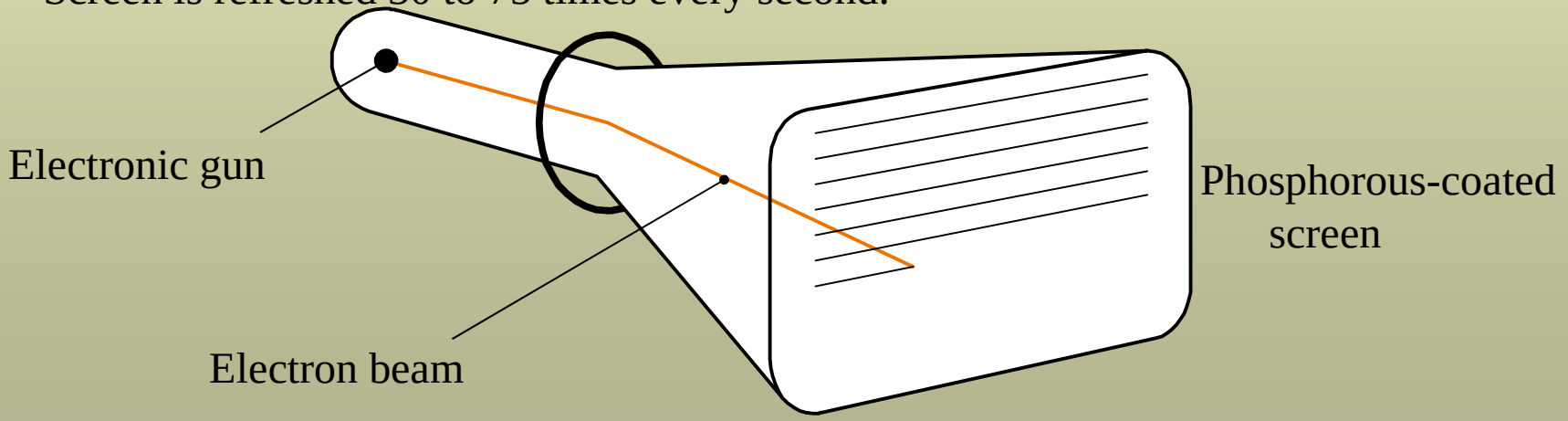
Optical mice uses an optical sensor to detect motion





Output Device: Cathode Ray Tube

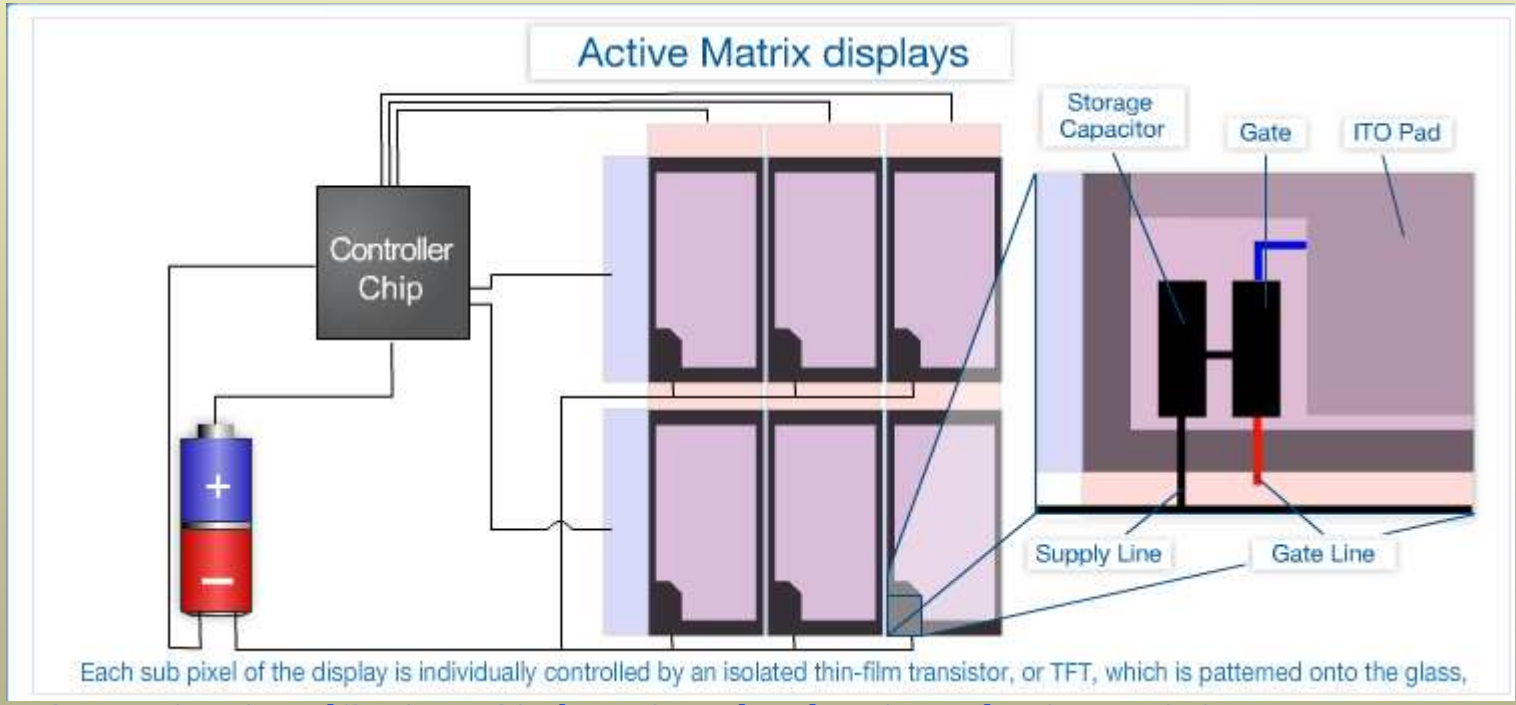
- Cathode ray tube (CRT) consists of an electronic gun that shoots electrons onto a phosphorous-coated screen (display).
- The screen is divided into picture elements (or pixels) with a resolution varying from (512 x 340) to (1560 x 1280) pixels.
- The electronic gun illuminates each pixel depending upon the given intensity.
- Typically the number of intensity levels depend upon the number of bits allocated per pixel. For 8 bits/pixel gray-scale displays, the number of intensity levels are 256.
- Color displays use 24 bits/pixel, 8 bits for each of the three primary colors (R,G,B).
- Screen is refreshed 30 to 75 times every second.



A CRT Display

Output Device: Cathode Ray Tube

- LCD displays consist of “light valves” which turn on and off the light to the 3 sub-pixels (RGB)
- “Light valves” – formed by liquid crystals which turn on and off in response to applied voltage
- Each light valve is varied using a sophisticated method known as an active matrix



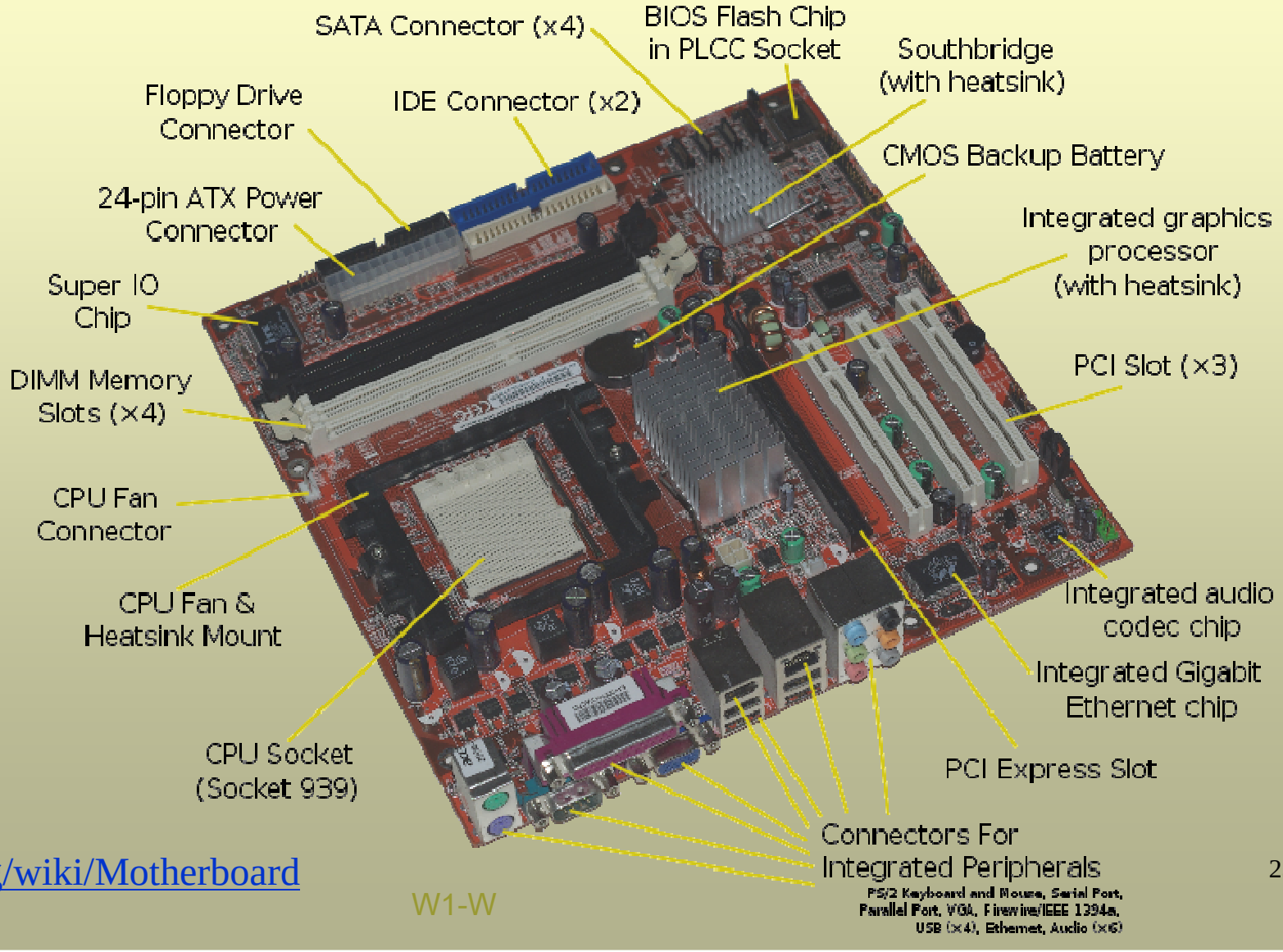
http://solutions.3m.com/wps/portal/3M/en_US/Vikuiti1/BrandProducts/secondary/optics101/

Inside a PC (1)

PC consists of a motherboard (CPU, onboard memory, i/o devices), hard disk, floppy drives, power supply, and connectors.

Inside a PC: Motherboard (2)

PC Motherboard consists of a central processing unit (CPU), typically PCI card slots, Dynamic random access memory (DRAM), and connectors for I/O devices

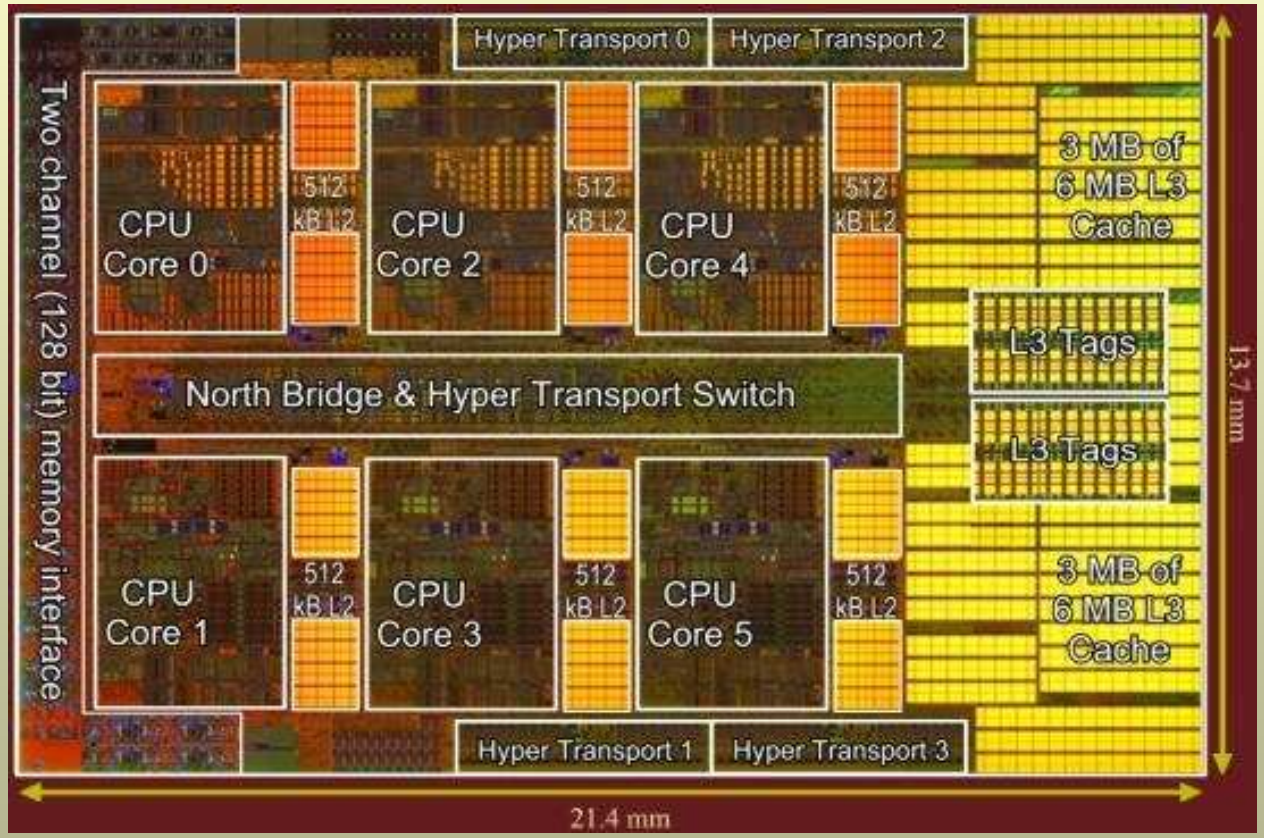


http://en.wikipedia.org/wiki/Motherboard

Inside a PC: CPU (3)

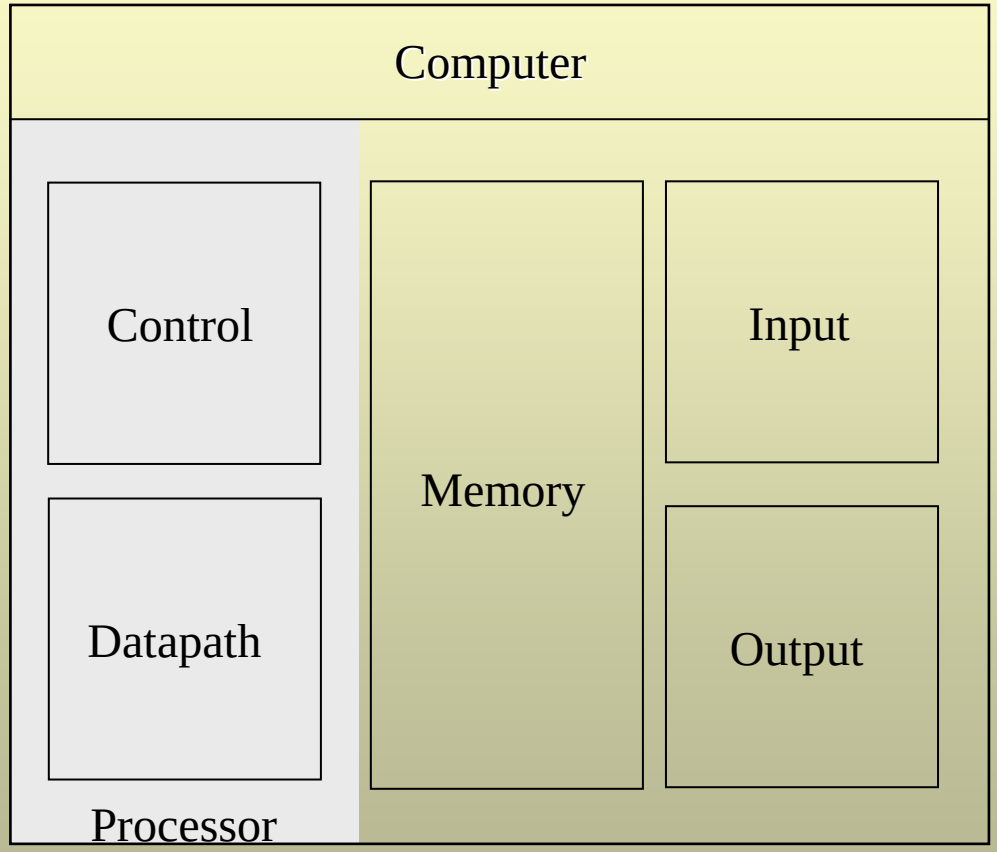
CPU comprises of two main components:

1. Datapath: consists of Data and instruction cache, Bus, and integer and floating point data path. The latter performs integer and floating point arithmetic operations
2. Control: tells the datapath memory and I/O devices what to do based on the program



http://www.hardware.info/en-US/news/ymickpqWwp2acJY/AMD_hexacore_CPU_in_2009_Montreal_deleted/

Inside a PC: The Big Picture (4)



Innovation (1)

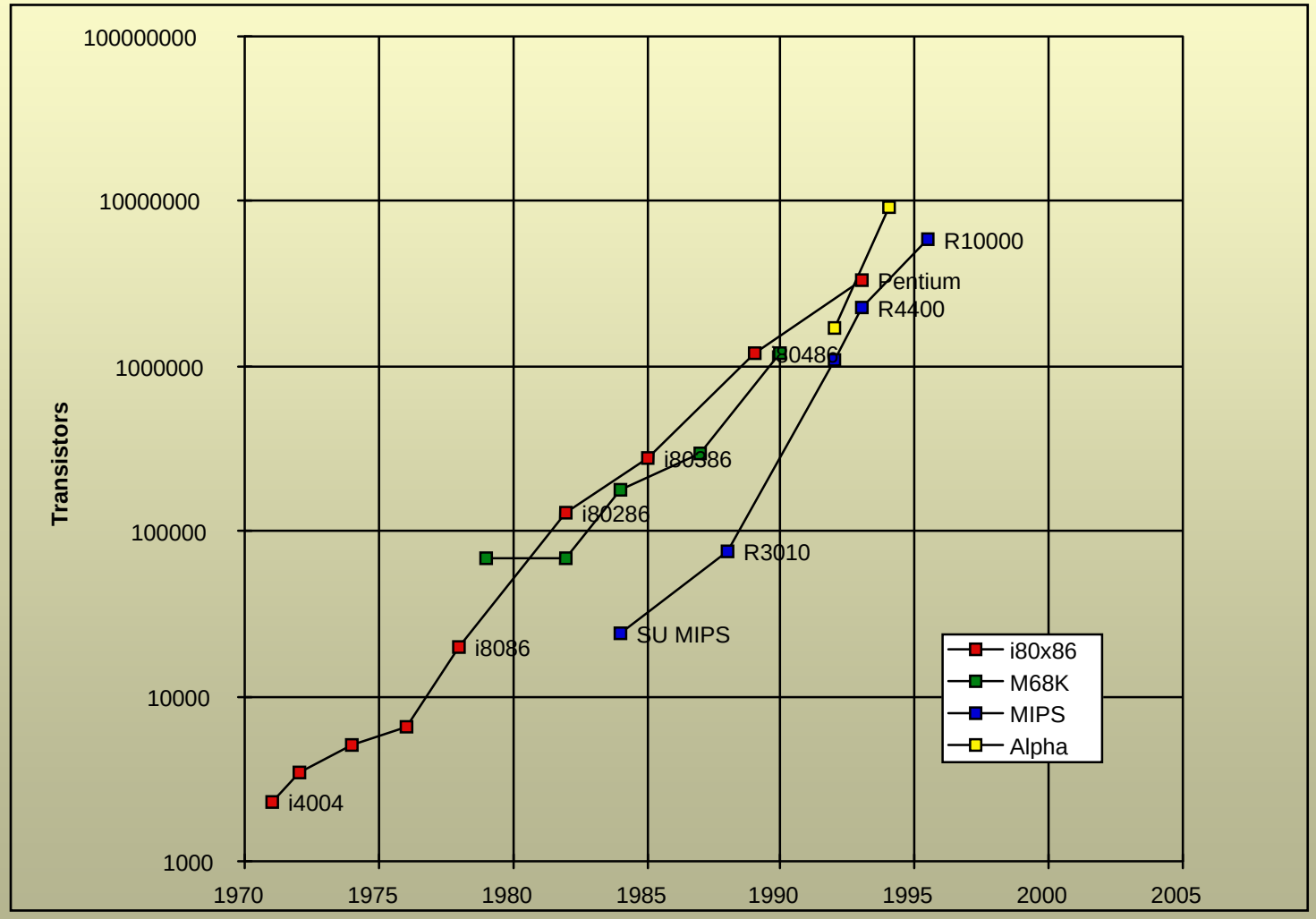
- Technology used in computers have advanced from vacuum tubes (1951), transistors (1965), Integrated circuits (IC's) (1975), to Very large scale integrated circuits (VLSI) (1995).
- Capacity of a DRAM has quadrupled every 3 years

Year	Size (bit)
1980	64 Kb
1983	256 Kb
1986	1 Mb
1989	4 Mb
1992	16 Mb
1996	64 Mb
1999	256 Mb
2002	**1 Gb**

Fig. 1.14: Capacity of DRAM over 1980 - 2002

Innovation (2)

Moore's Law states that the transistor density on integrated circuits doubles every couple of years. This exponential growth and ever-shrinking transistor size has resulted in increased performance with decreased cost.



Where are we headed?

- Performance issues (Chapter 1.4 – 1.8) *vocabulary and motivation*
- MIPS instruction set architecture (Chapter 2)
- Arithmetic and how to build an ALU (Chapter 3)
- Constructing a processor to execute our instructions (Chapter 4)
- Pipelining to improve performance (Chapter 4)
- Memory: caches and virtual memory (Chapter 5)
- I/O (Chapter 6)