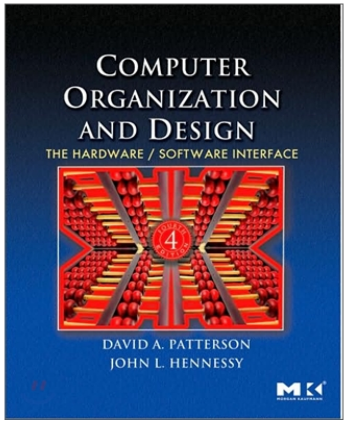
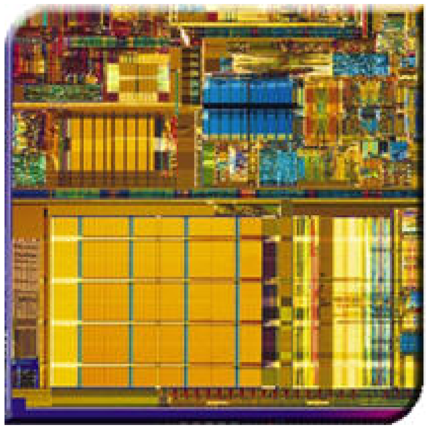
Computer Architecture

Lecture 1 Introduction





Prof. Jongmyon Kim

울산대학교
UNIVERSITY OF ULSAN

Course Information

- Instructor
  - Prof. Jong-Myon Kim (jongmyon.kim@gmail.com)

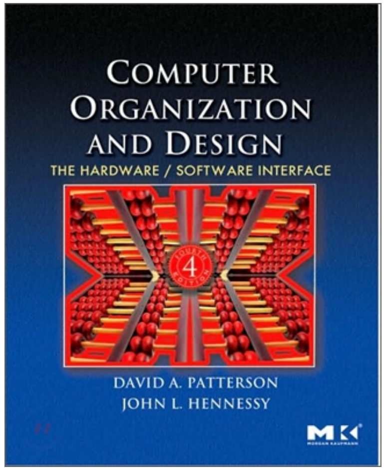
- Class page:
  - http://uclass.ulsan.ac.kr/
  - Constantly updated, check it out regularly

- Prerequisite: Logic circuit or the equivalent

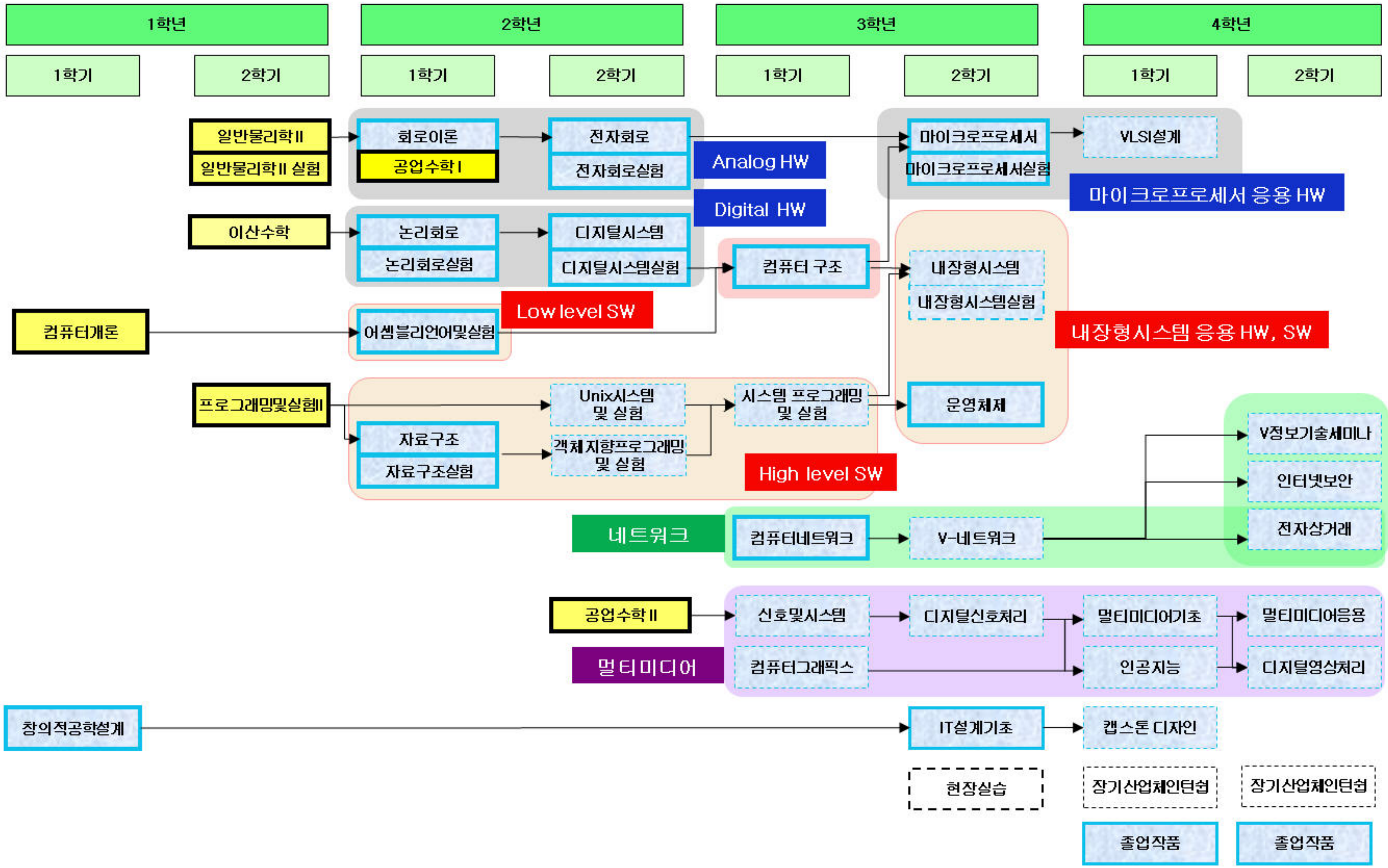
- Meeting time: Monday 2,3, Wednesday 3

- Meeting place : 7-613

- Textbook
  - Hennessy and Patterson, *Computer Organization and Design* (**4th edition**), Morgan Kaufmann, 2008.
- Other teaching materials
  - Slides & Lectures



Text Book


1학년
2학년
3학년
4학년
1학기
2학기
1학기
2학기
1학기
2학기
1학기
2학기
일반물리학II
일반물리학II 실험
회로이론
공업수학I
전자회로
전자회로실험
Analog HW
마이크로프로세서
마이크로프로세서실험
VLSI설계
마이크로프로세서 응용 HW
이산수학
논리회로
논리회로실험
디지털시스템
디지털시스템실험
Digital HW
컴퓨터 구조
내장형시스템
내장형시스템실험
컴퓨터개론
어셈블리언어및실험
Low level SW
내장형시스템 응용 HW, SW
프로그래밍및실험II
Unix시스템
및 실험
시스템 프로그래밍
및 실험
운영체제
자료구조
자료구조실험
객체지향프로그래밍
및 실험
High level SW
V정보기술세미나
인터넷보안
네트워크
컴퓨터네트워크
V-네트워크
전자상거래
공업수학II
신호및시스템
디지털신호처리
멀티미디어기초
멀티미디어응용
멀티미디어
컴퓨터그래픽스
인공지능
디지털영상처리
창의적공학설계
IT설계기초
캡스톤 디자인
현장실습
장기산업체인턴쉽
장기산업체인턴쉽
졸업작품
졸업작품

Weekly Plan & Reading

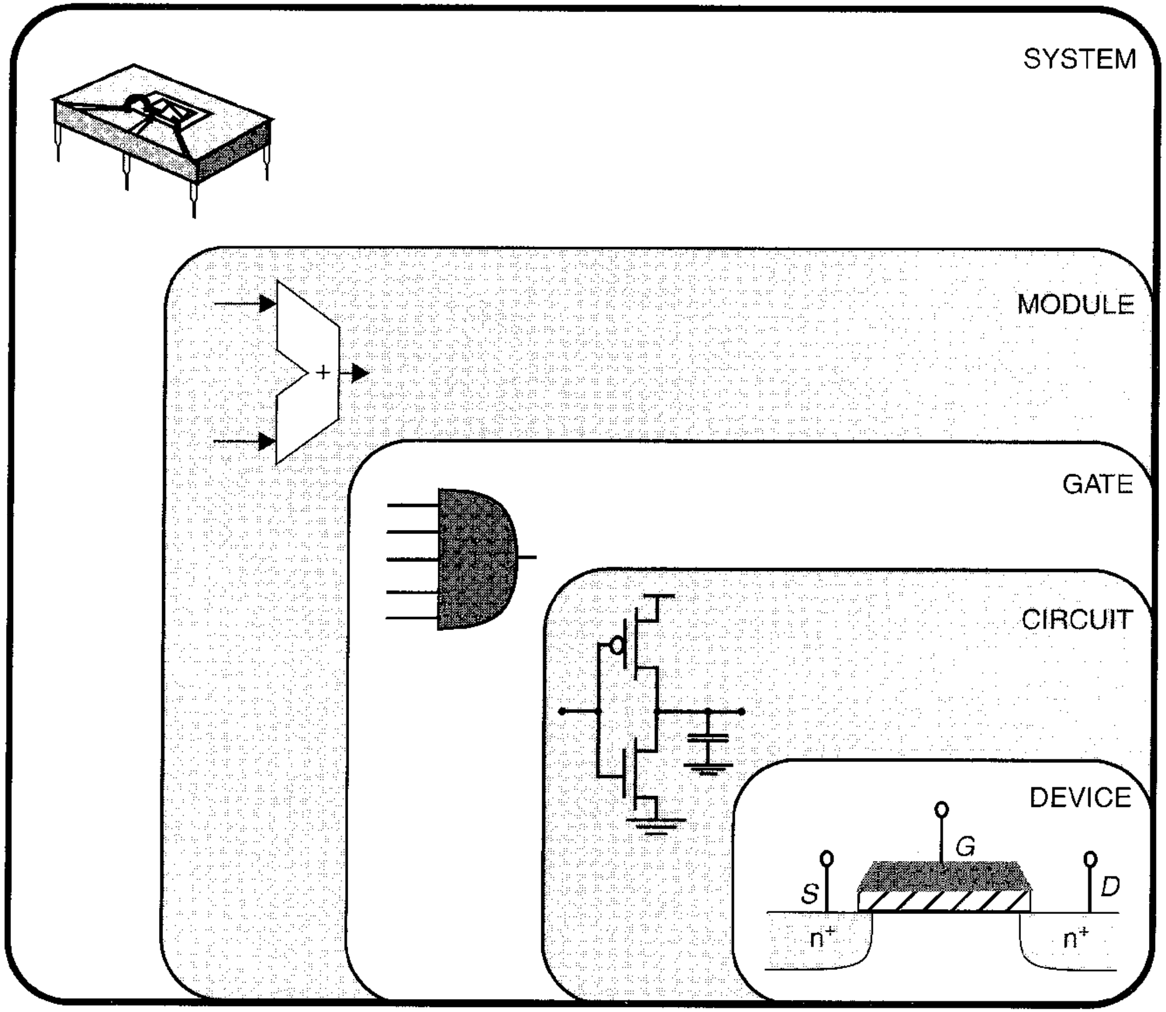
- Week 1: Introduction and Instruction Set Architecture
- Week 2: MIPS Instructions, Software Conversion, Stack
- Week 3: Arithmetic for Computers
- Week 4: Arithmetic for Computers
- Week 5: CPU Performance
- Week 6: Single-Cycle Datapath Processor Design
- Week 7: Single-Cycle Datapath Processor Design & Term project 1 (tentative)
- Week 8: *Mid-Term Exam*
- Week 9: Multi-Cycle Datapath Processor Design
- Week 10: Multi-Cycle Datapath Implementation – Exception & Microprogramming
- Week 11: Pipelining Processor : Overview & Datapath
- Week 12: Pipelining Processor : Control, Hazard, Exceptions
- Week 13: Memory Hierarchy : Caches
- Week 14: Memory Hierarchy : Cache performance
- Week 15: Storage, Networks, and Other Peripherals
- Week 16: *Final Exam*

울산대학교 UNIVERSITY OF ULSAN

Grading Policy

- Term Projects: 20%
  - Individual work, no collaboration unless otherwise announced
- Exams: 70%
  - Mid-Term Exam: 30%
  - Final Exam: 40%
- Class Participation: 10%
- Final Grade is relative to your peer in class

Design Steps



[Jan Rabaey's Digital Circuit Design 중에서]

Objectives — To Learn

- Core concepts of microprocessor architecture
  - ISA
  - Pipelining
  - Hazards
  - Cache/Memory hierarchy
  - Memory management

Introduction

- This course is all about how computers work

- But what do we mean by a computer?
  - Different types: desktop, servers, embedded devices
  - Different uses: automobiles, graphics, finance, genomics…
  - Different manufacturers: Intel, Apple, IBM, Microsoft, Sun…
  - Different underlying technologies and different costs!

- Best way to learn:
  - Focus on a specific instance and learn how it works
  - While learning general principles and historical perspectives

Why learn this stuff?

- You want to call yourself a “computer scientist”
- You want to build software people use (need performance)
- You need to make a purchasing decision or offer “expert” advice
- Both Hardware and Software affect performance:
  - Algorithm determines number of source-level statements
  - Language/Compiler/Architecture determine machine instructions
  - Processor/Memory determine how fast instructions are executed
- Assessing and Understanding Performance

Moore’s Law

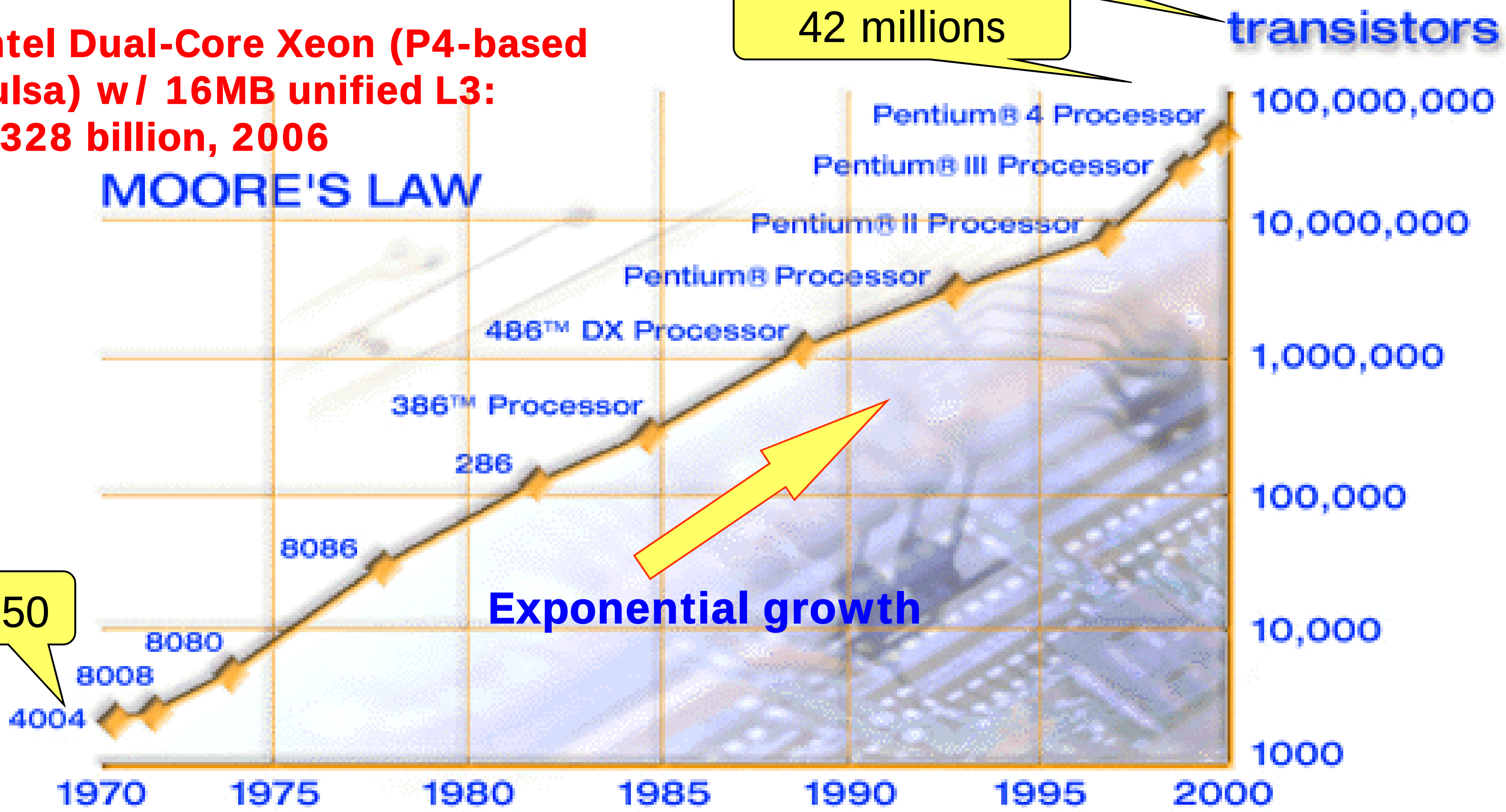
IBM latest POWER5 has 276 million transistors

Intel Dual-Core Xeon (P4-based Tulsa) w/ 16MB unified L3: 1.328 billion, 2006

P4 Extreme Ed.
178 millions w/ 2MB L3

Core 2 Duo (Conroe)
291 millions, July 2006

42 millions



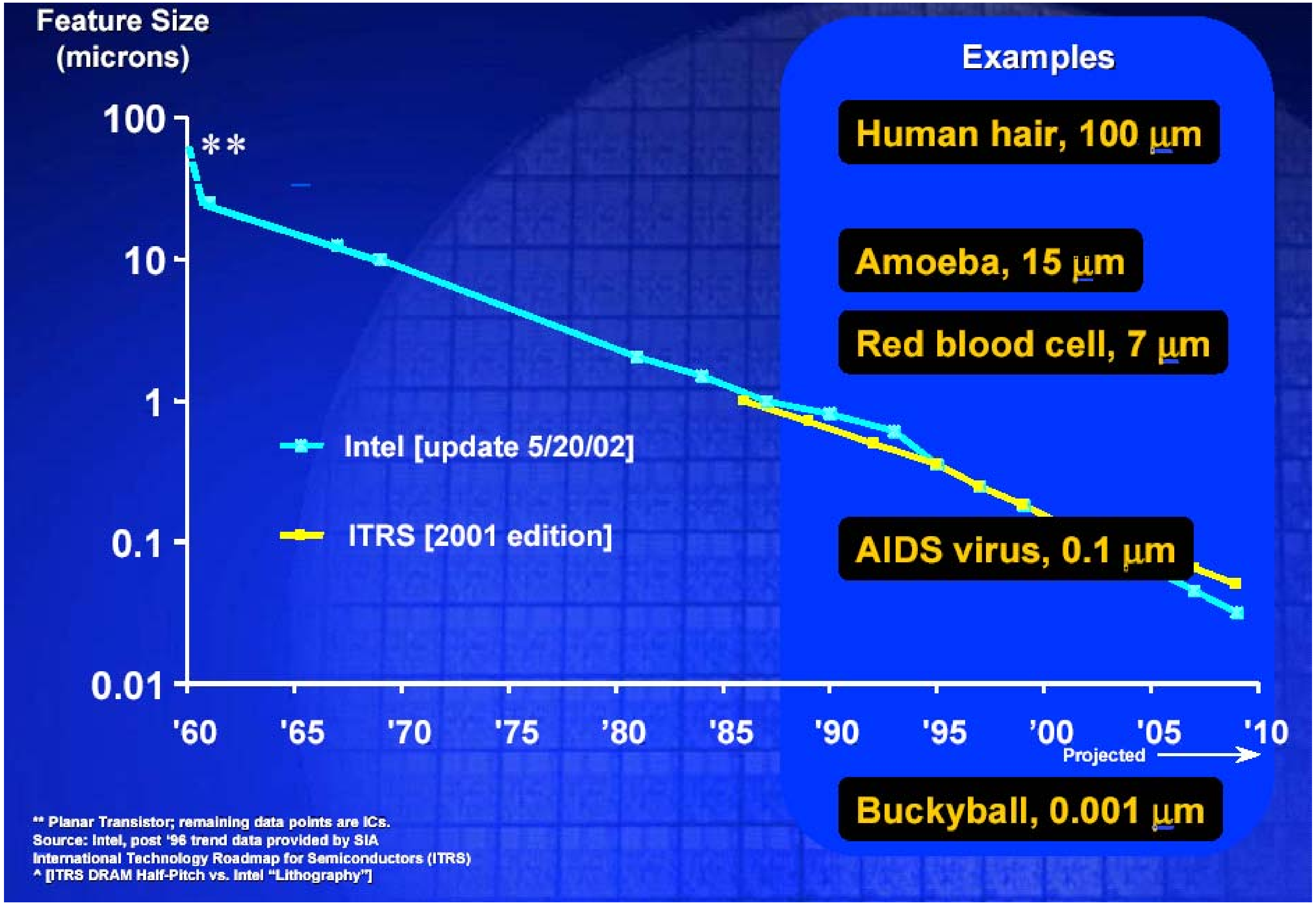
2,250

Exponential growth

Transistor count will be doubled every 18 months

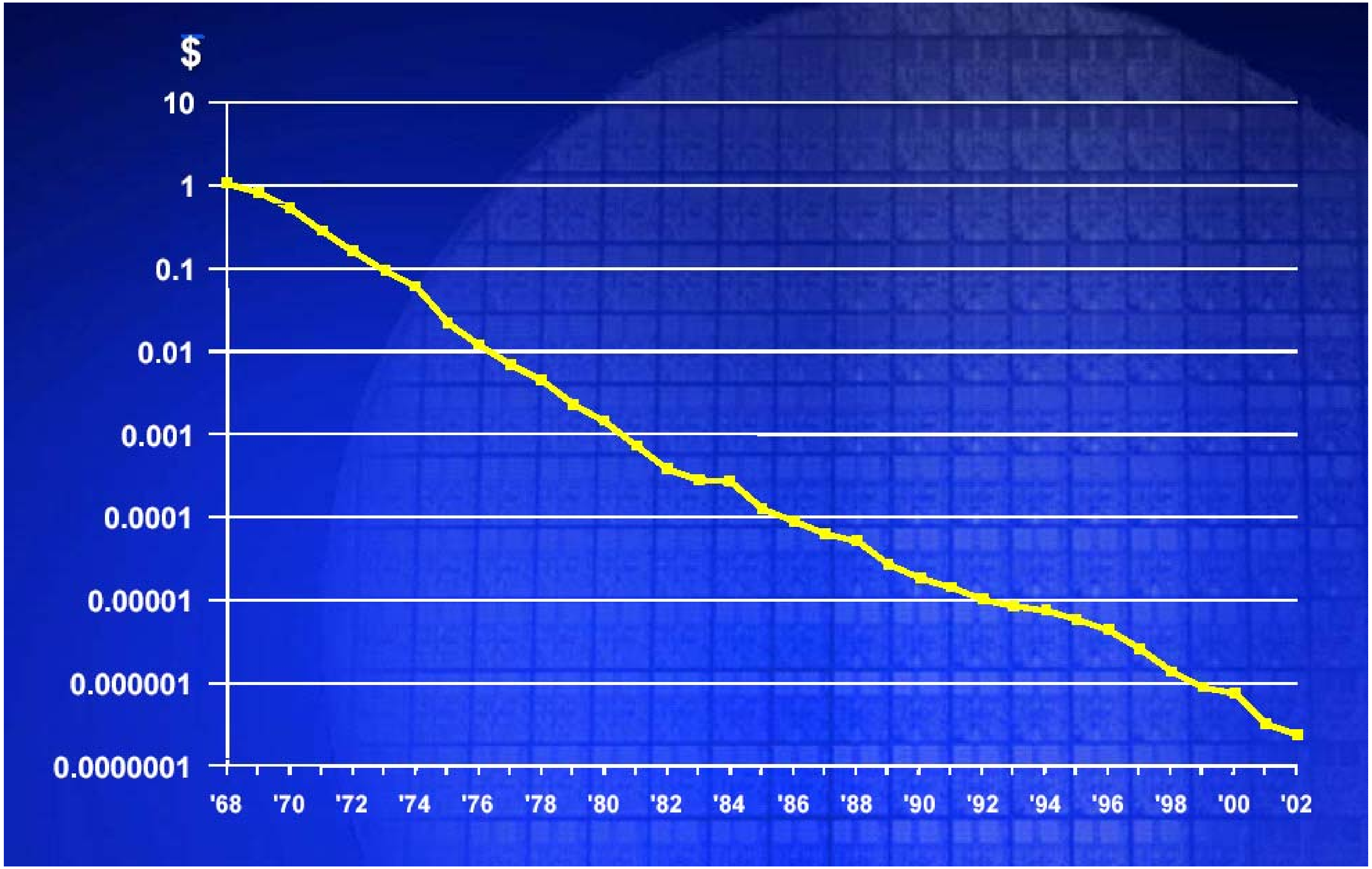
— Gordon Moore, Intel co-founder

Feature Size



We are currently at 0.09µm and moving towards 0.025µm

Average Transistor Cost Per Year



What is a computer?

- Components:
  - Processor(s)
  - Co-processors (graphics, security)
  - Memory (disk drives, DRAM, SRAM, CD/DVD)
  - input (mouse, keyboard, mic)
  - output (display, printer)
  - network
- Our primary focus: the processor (datapath and control)
  - implemented using millions of transistors
  - Impossible to understand by looking at each transistor
  - We need...

Abstraction

High Level Language

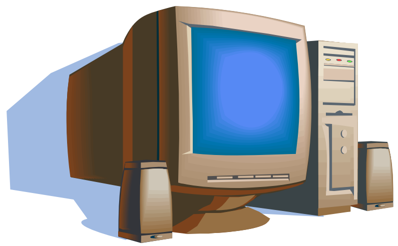
```
main() {
   int  i,b,c,a[10];
   for (i=0; i<10; i++)...
      a[2] = b + c*i;
}
```

Compiler

ISA

```
...
lw    r2, mem[r7]
add  r3, r4, r2
st     r3, mem[r8]
```

Assembler

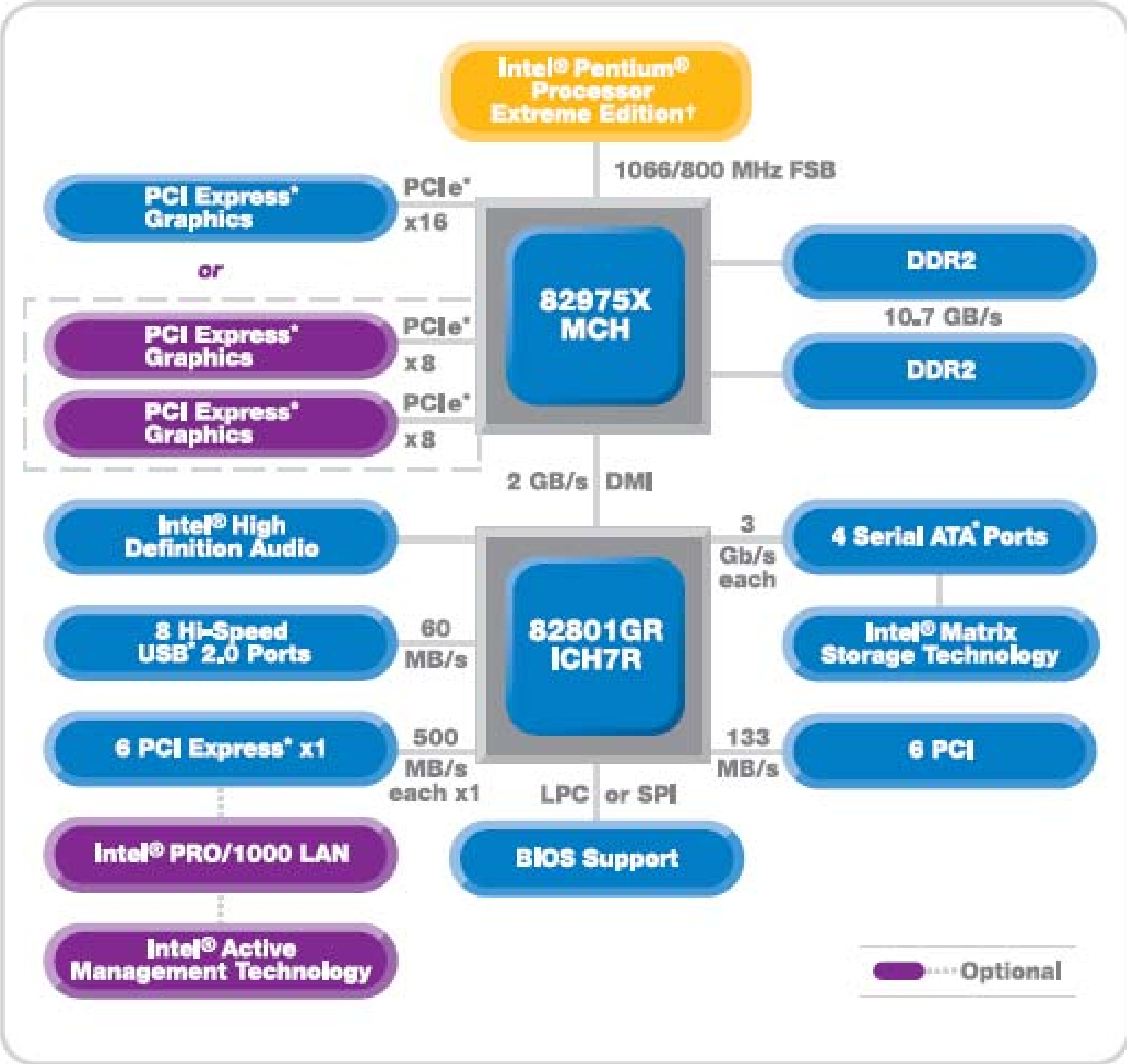


- Delving into the depths reveals more information
- An abstraction omits unneeded detail, helps us cope with complexity
- *What are some of the details that appear in these familiar abstractions?*

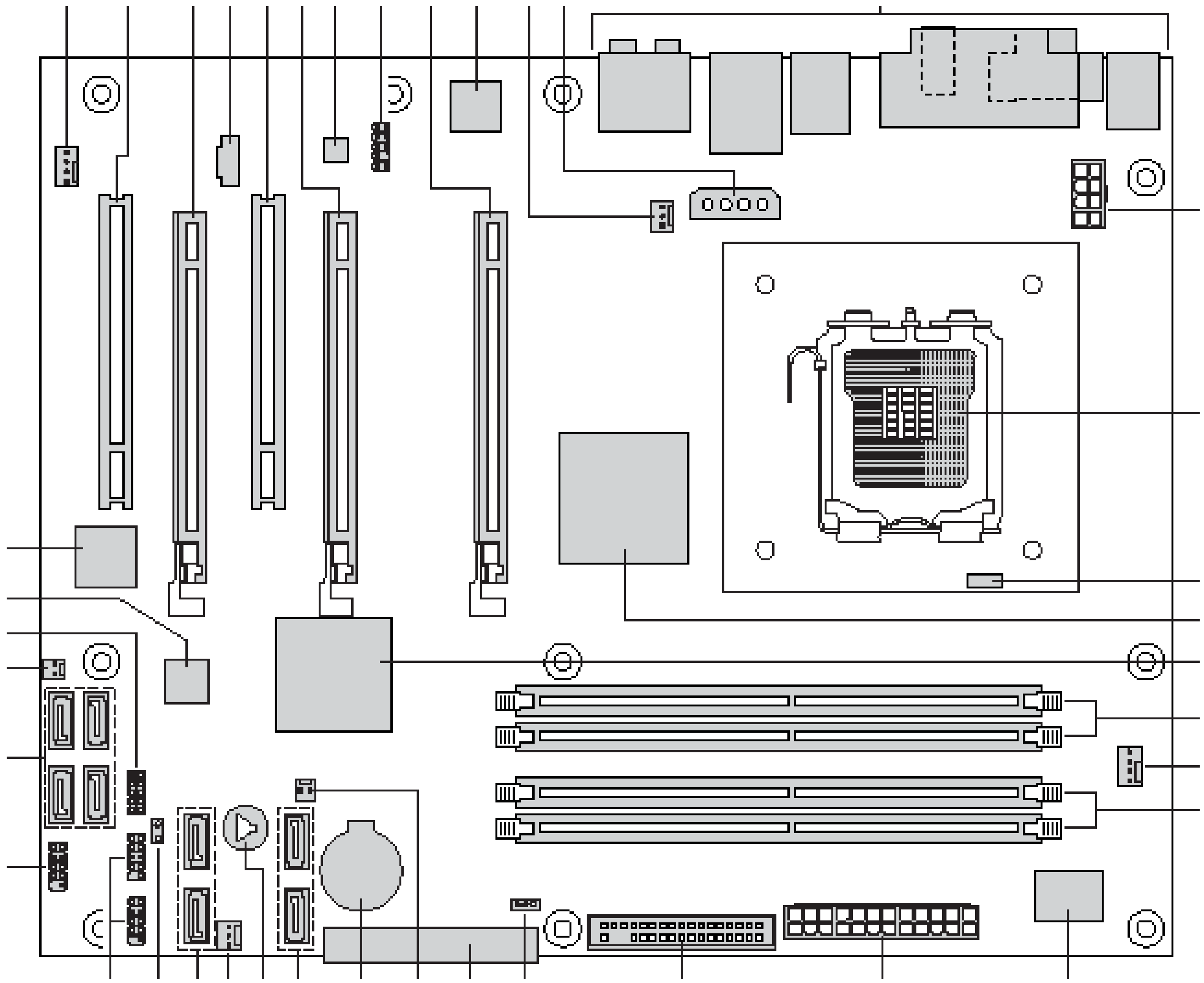
How do computers work?

- Need to understand abstractions such as:
  - Applications software
  - Systems software
  - Assembly Language
  - Machine Language
  - Architectural Issues: i.e., Caches, Virtual Memory, Pipelining
  - Sequential logic, finite state machines
  - Combinational logic, arithmetic circuits
  - Boolean logic, 1s and 0s
  - Transistors used to build logic gates (CMOS)
  - Semiconductors/Silicon used to build transistors
  - Properties of atoms, electrons, and quantum dynamics
- So much to learn!

A Typical PC System Architecture



A Typical PC Motherboard (D975XBX)



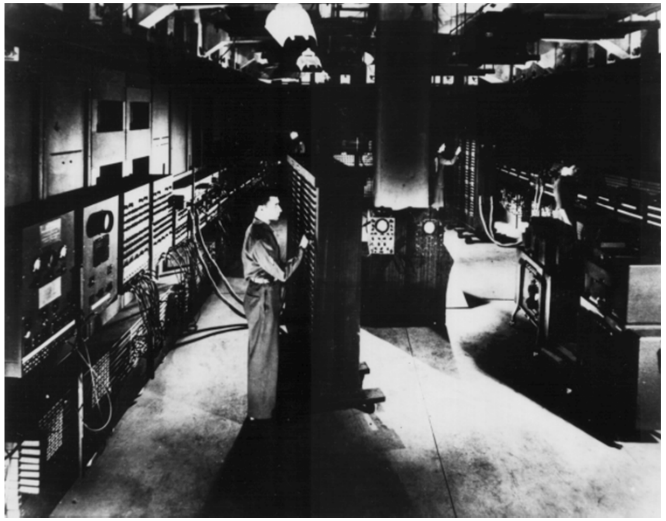
A Typical PC Motherboard (D975XBX)

Instruction Set Architecture

- A very important abstraction
  - interface between hardware and low-level software
  - standardizes instructions, machine language bit patterns, etc.
  - advantage: *different implementations of the same architecture*
  - disadvantage: *sometimes prevents using new innovations*
- Modern instruction set architectures:
  - IA-32, PowerPC, MIPS, SPARC, ARM, and others

Historical Perspective

- ENIAC built in World War II was the first general purpose computer
  - Used for computing artillery firing tables
  - 80 feet long by 8.5 feet high and several feet wide
  - Each of the twenty 10 digit registers was 2 feet long
  - Used 18,000 vacuum tubes
  - Performed 1900 additions per second



–Since then:

Moore's Law:

transistor capacity doubles every 18-24 months

Where we are headed

- A specific instruction set architecture
- Performance issues
- Arithmetic and how to build an ALU
- Constructing a processor to execute our instructions
- Pipelining to improve performance
- Memory: caches and virtual memory
- I/O