微算機系統 第一章
Computer Abstractions and Technology

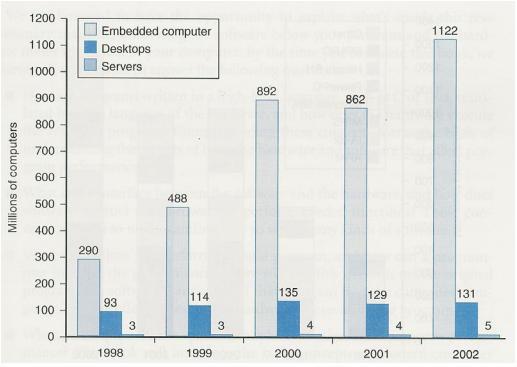
陳伯寧 教授
電信工程學系
國立交通大學

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!*
- *Both Hardware and Software affect performance:*
  - *Algorithm determines number of source-level statements*
  - *Language/Compiler/Architecture determine machine instructions (Chapter 2 and 3)*
  - *Processor/Memory determine how fast instructions are executed (Chapter 5, 6, and 7)*
  - *Assessing and Understanding Performance in Chapter 4*

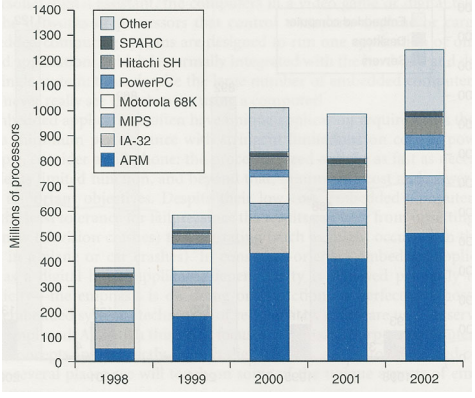
Introduction

- *The growth in the number of embedded computers has been much faster (40% annually) than that among desktops and servers (9% annually).*



Introduction

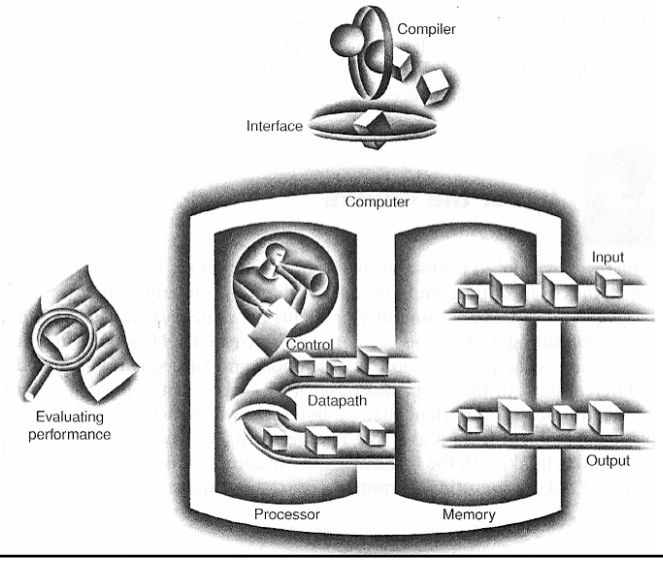
- *ARM core dominates the market of embedded computers, such as cell phone.*



Inside a computer

- *Five classic components inside a computer:*
  - *Input (mouse, keyboard, scanner, ...)*
    - *Writes data to memory*
  - *Output (CRT display, LCD display printer, plotter, ...)*
    - *Reads data from memory*
  - *Memory (either volatile or no non-volatile)*
    - *Primary or Main memory : DRAM, SRAM, ...*
    - *Secondary memory : disk drives, CD-ROM, ...*
  - *Data Path*
  - *Control*
- *Data Path and Control are sometimes combined and called the processor.*

Inside a computer



Processor

- *Our primary focus: The processor (datapath and control)*
  - *Implemented using millions of transistors*
  - *Impossible to understand by looking at each transistor*
  - *Designed based on a **computer architecture** paradigm*
  - *We therefore need **computer abstraction**.*
- *An abstraction omits unneeded detail, helps us cope with complexity.*

Abstractions

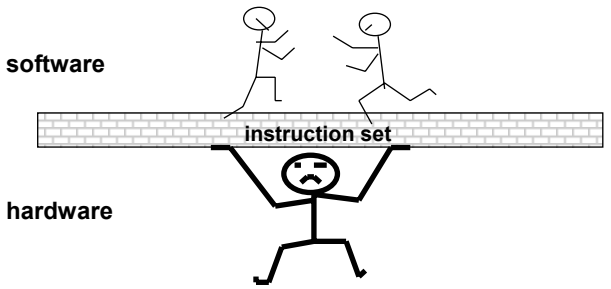
- *Examples of other **abstractions**:*
  - *Applications software*
  - *Systems software*
  - *Assembly Language*
  - *Machine Language*
  - *Architectural Issues: i.e., Caches, Virtual Memory, Pipelining*
  - *Sequential logic, finite state machines*
  - *Combinational logic, arithmetic circuits*
  - *Boolean logic, 1s and 0s*
  - *…*

What is computer architecture?

Computer Architecture =

Instruction Set Architecture +

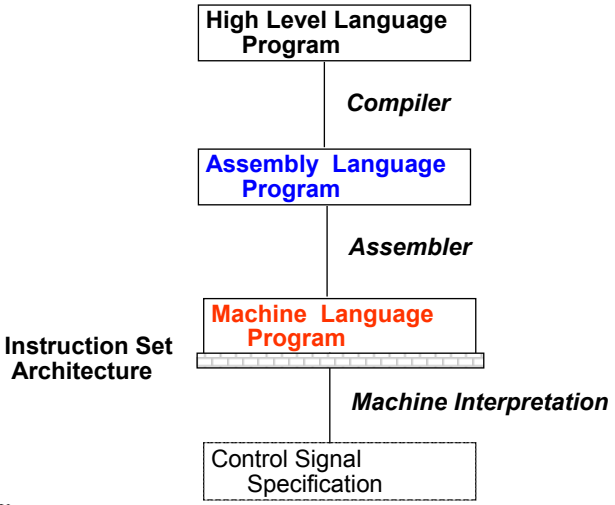
Machine Organization



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*

Demonstration of instruction set architecture



Demonstration of instruction set architecture

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)

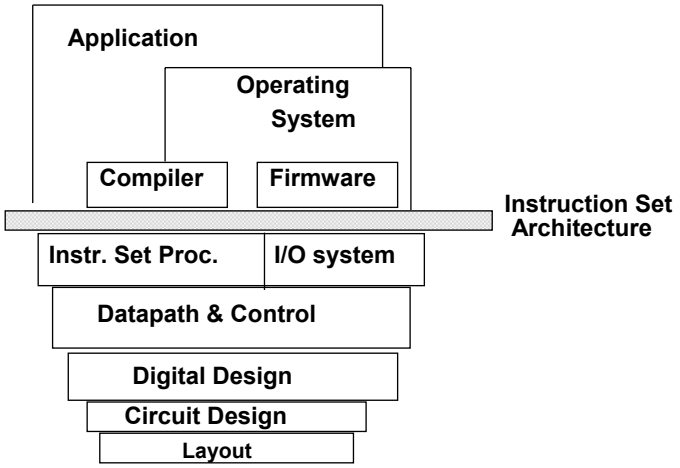
```
swap:   muli   $2, $5, 4
        add    $2, $4, $2
        lw     $15, 0($2)
        lw     $16, 4($2)
        sw     $16, 0($2)
        sw     $15, 4($2)
        jr     $31
```

Assembler

Binary machine language program (for MIPS)

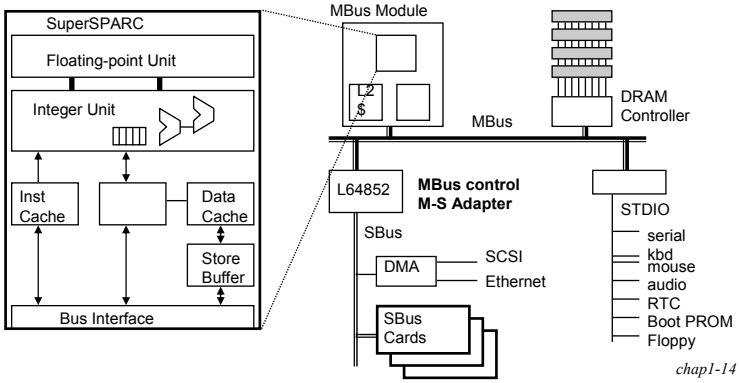
```
0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 1 1 1 0 0 0 0 1 1 0 0 0 0 0 1 0 0 0 0 1
1 0 0 0 1 1 0 0 0 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 1 1 0 0 1 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
1 0 1 0 1 1 0 0 1 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 1 1 0 0 0 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
```

Demonstration of instruction set architecture



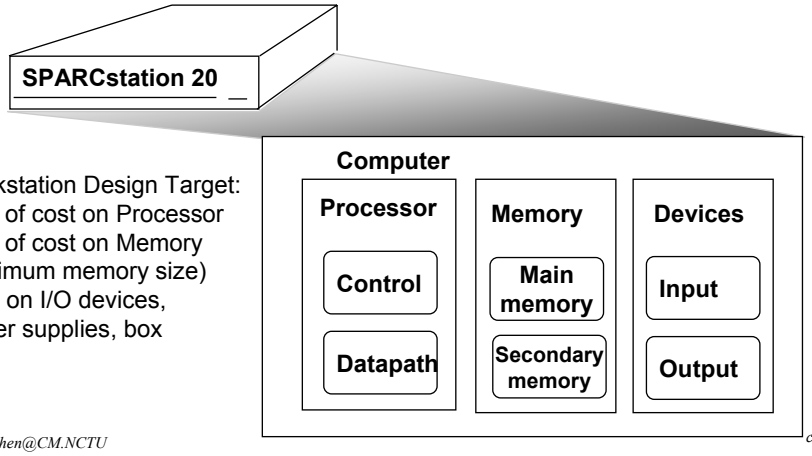
Demonstration of machine organization

- *Machine Organization*
  - *Ways in which the principle function units, such as ALUs and registers, inside a computer are interconnected and interoperated.*



Demonstration of machine organization

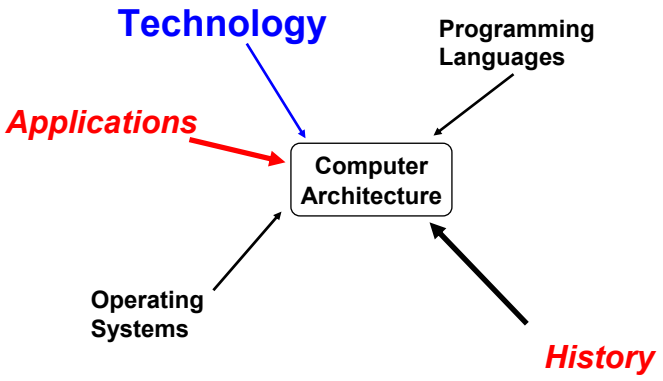
- *Machine Organization*



Workstation Design Target:
25% of cost on Processor
25% of cost on Memory
(minimum memory size)
Rest on I/O devices,
power supplies, box

Computer architecture revisited

- *Forces that drive the computer architecture revolution*



Technology – dramatic change

- *Processor*
  - *Logic capacity: Improves about 30% per year*
  - *Clock rate: Improves about 20% per year*
- *Memory*
  - *DRAM capacity: Improves about 60% per year (4x every 3 years)*
  - *Memory speed: Improves about 10% per year*
  - *Cost per bit: Improves about 25% per year*
- *Disk*
  - *Capacity: Improves about 60% per year*

Where we are headed?

- *A specific instruction set architecture (Chapter 2)*
- *Arithmetic and how to build an ALU (Chapter 3)*
- *Performance evaluations (Chapter 4)*
- *Constructing a processor to execute our instructions (Chapter 5 & Appendix C)*
- *Pipelining to improve performance (Chapter 6)*
- *Memory: caches and virtual memory (Chapter 7)*
- *I/O and peripherals (Chapter 8)*
- *<u>We will not cover</u> Multiprocessors and clusters (Chapter 9)*

Where we are headed?

- *Data Path: Chapters 3, 5, and 6*
- *Control: Chapters 5 and 6*
- *Memory: Chapter 7*
- *Input: Chapter 8*
- *Output: Chapter 8*
- *Performance: Chapter 4*

Suggestive exercises

- *1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 1.01, 1.11, 1.12, 1.14, 1.18, 1.20, 1.21, 1.22, 1.24, 1.25, 1.26*
- *1.29, 1.30, 1.31, 1.32, 1.35, 1.36, 1.37, 1.38, 1.40, 1.41, 1.42, 1.43, 1.45*
- *1.52, 1.53, 1.54*

- *Check Yourself*
  - *Section 1.3: page 27*

- *Should you have questions on exercises, you can ask the teaching assistants (助教) at their office hours.*