

Computer Organization and Design

Ch1: Computer Abstractions and Technology

计算机系统概念和技术

- **Basic Ideas and Definition**
- **Major Components of Software/Hardware**
- **Computer Revolution**

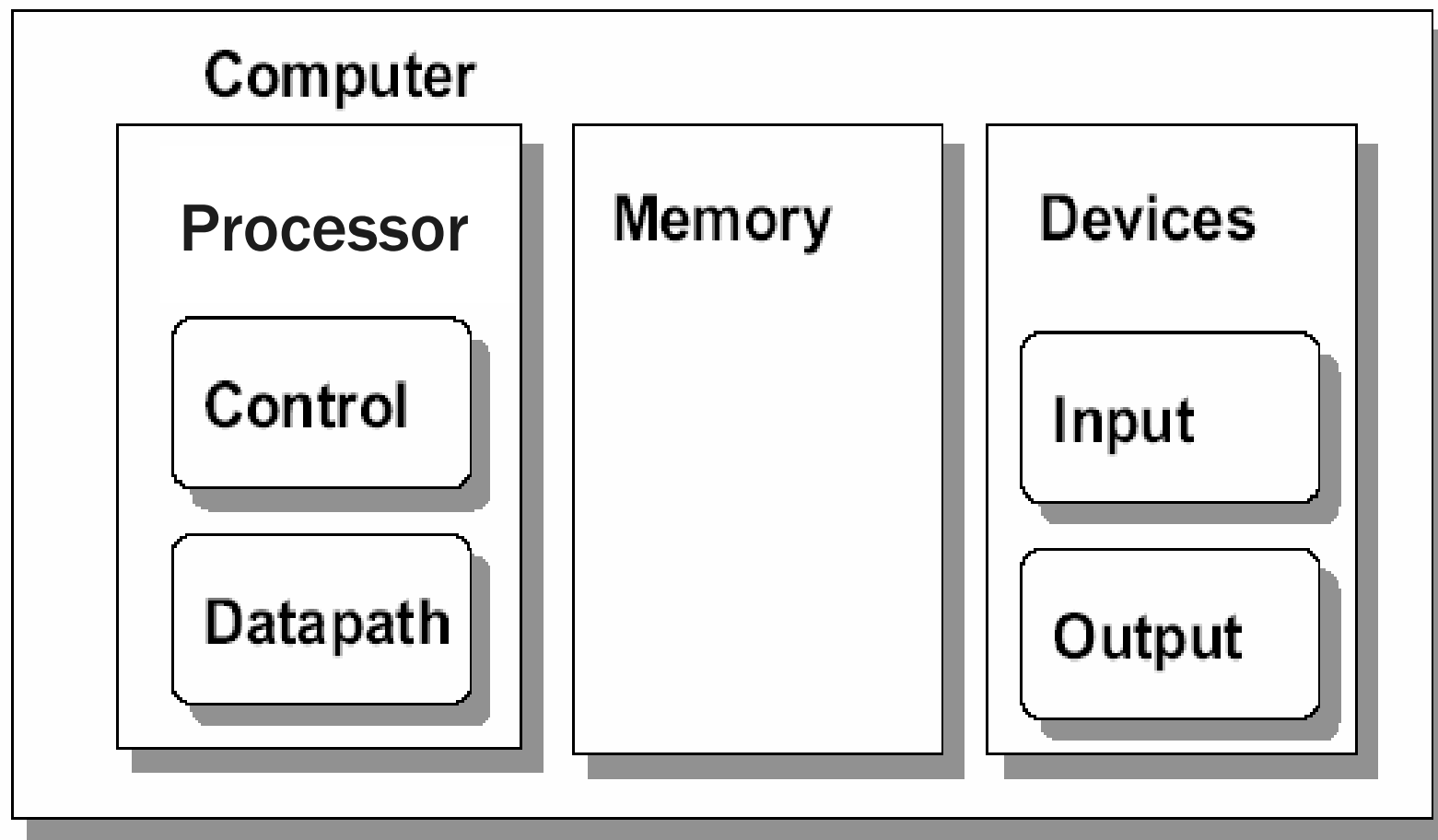
主要内容

- 回顾：计算机系统
 - 计算机硬件：CPU+MM+I/O
 - 计算机软件：系统软件+应用软件
- 回顾：计算机层次结构
- 回顾：计算机硬件和软件的接口：指令系统
- 回顾：计算机软件如何在硬件上执行
- 回顾：计算机发展简史
 - IAS通用计算机模型机：冯诺依曼结构
 - IBM360系列机：引入兼容性（系列机）概念
 - DEC PDP-8：引入总线结构
- 简介本课程内容
 - Computer Architecture = ISA + Organization

回顾: Computer System

- **Computer System = Hardware + Software**
- **Hardware** (tangible objects 具体的物理器件)
e.g integrated circuit, printed circuit board, cables, power supplies, memories, printer
- **Software** (**encoded symbol** 对抽象概念进行符号编码)
 - **Software=program+document**
 - **program = algorithms+data**
 - program 由若干条指令或语句组成

回顾: Hardware



回顾： functional component-1

- **Processor (CPU - Central Processing Unit)**
 - “Brain” of the computer
 - **Function:** *Execute programs*
 - **Form:** *Control Unit + Data path (ALU+Registers)*
 - *Control Unit: fetching instruction, decode instruction*
 - *ALU(Arithmetic Logic Unit): perform operations*
*+, -, *, / , And, Or, Move etc*
 - *Registers: store temporary results and control info*
 - *GRS(General Register Set): store operands and results*
 - *PC(Program Counter): point to the next instruction*
 - *IR(Instruction Register): hold the current instruction*

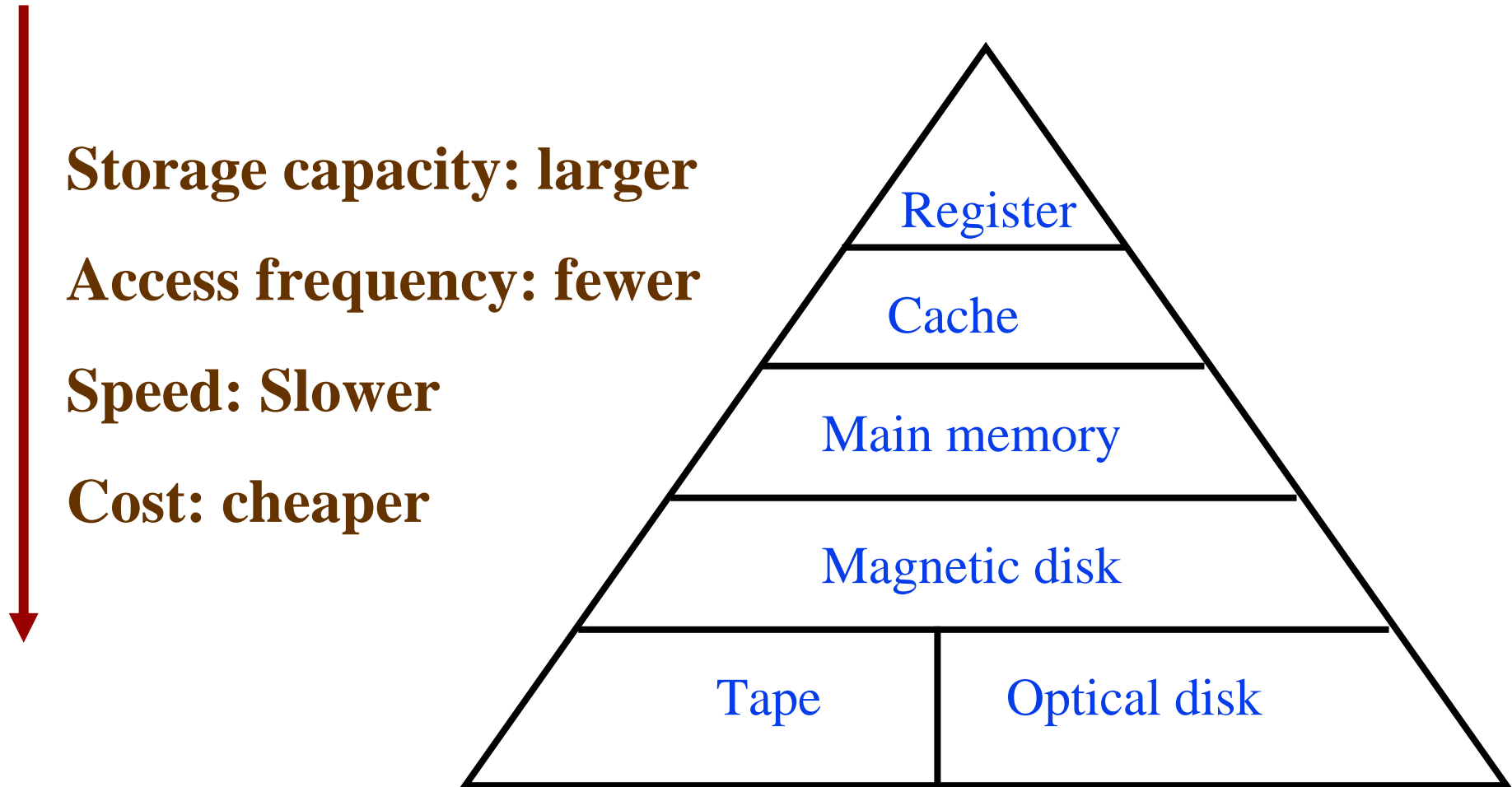
回顾: functional component-2

◦ Memory

“warehouse” of the computer

- **Function:** *store programs and data*
- **Form(Hierarchies):** *Primary + Secondary*
 - **Primary memory:** *Cache + Main memory (MM)*
 - **Cache:** *store the most heavily used word*
 - **MM:** *store the program to be started and its data*
 - **Secondary Storage:** *Magnetic disk, Tape, Optical disk*
 - **Magnetic disk:** *store all info inside a computer*
 - **Tape or Optical disk:** *for archival storage*

回顾：Memory Hierarchies

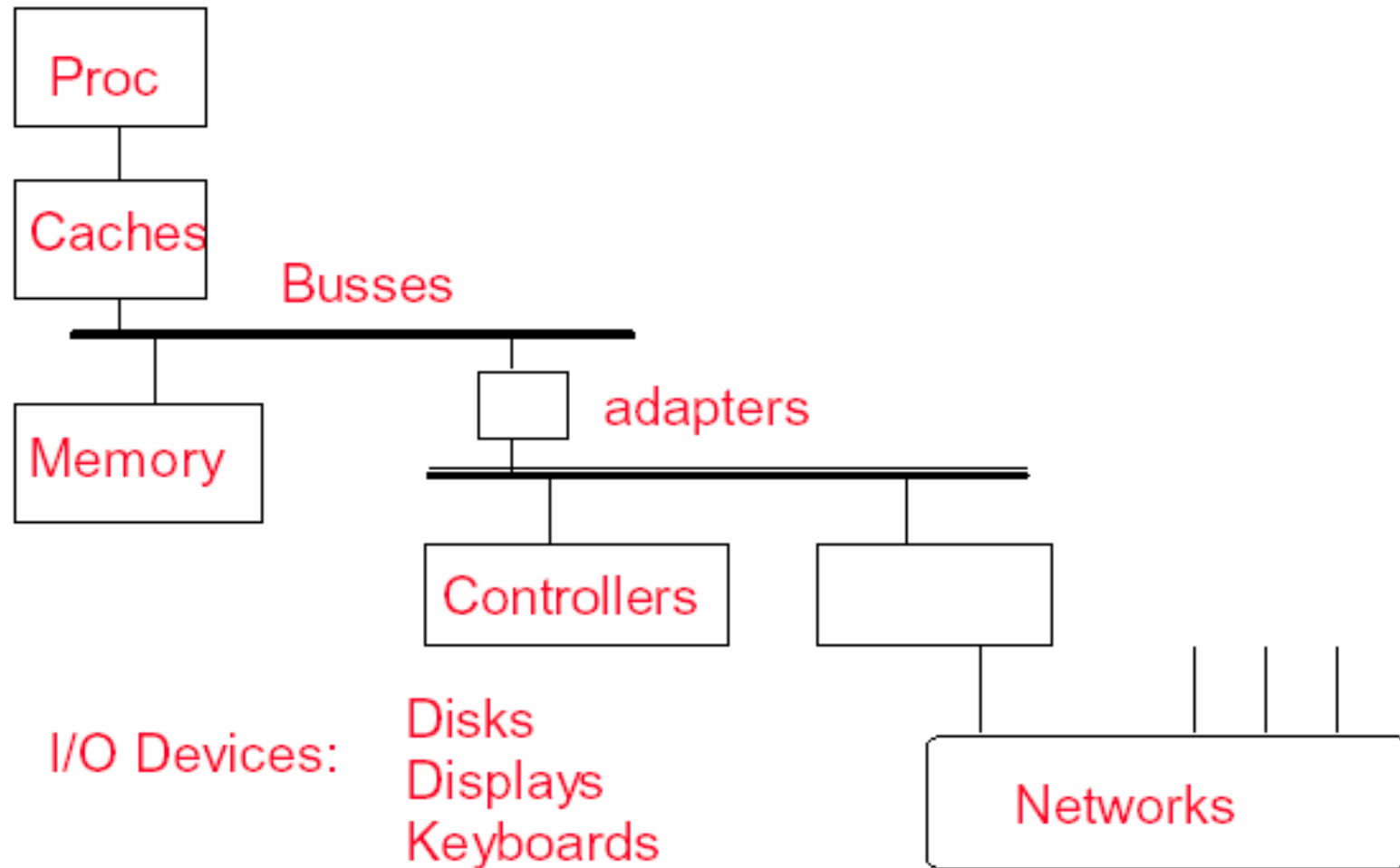


回顾: functional component-3

- **Input/Output (I/O)**

- **Function:** 各种信息的输入/输出
- **Form:** controller + I/O device itself
 - **Controller:** 控制外设, 完成与主机侧、外设侧的通信
 - **I/O device:** Input or Output (e.g Print or display etc)
 - *Keyboard*
 - *Printer*
 - *Display (CRT Monitor) etc*

回顾: BUS: Interconnection



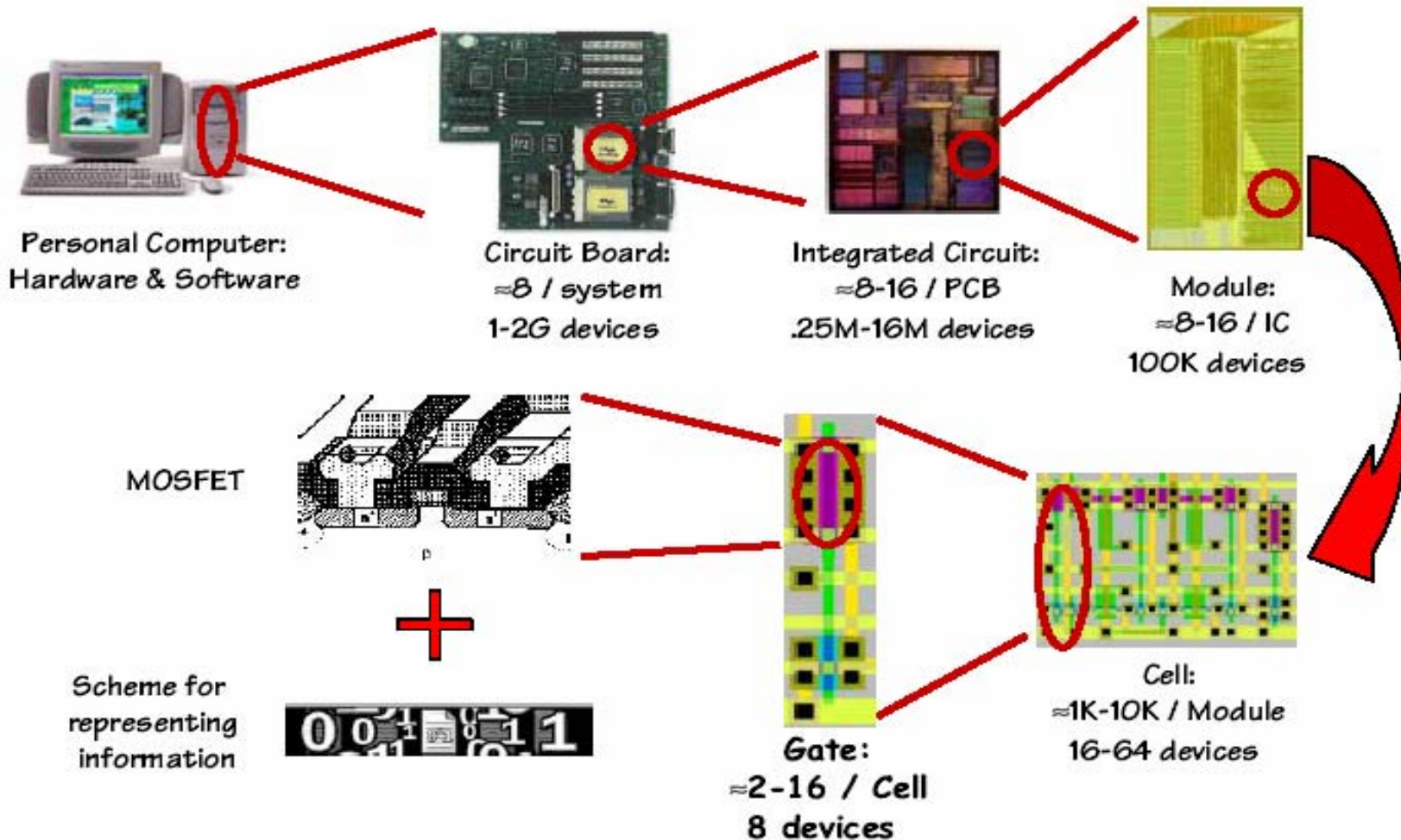
SKIP

回顾：打开计算机来看看

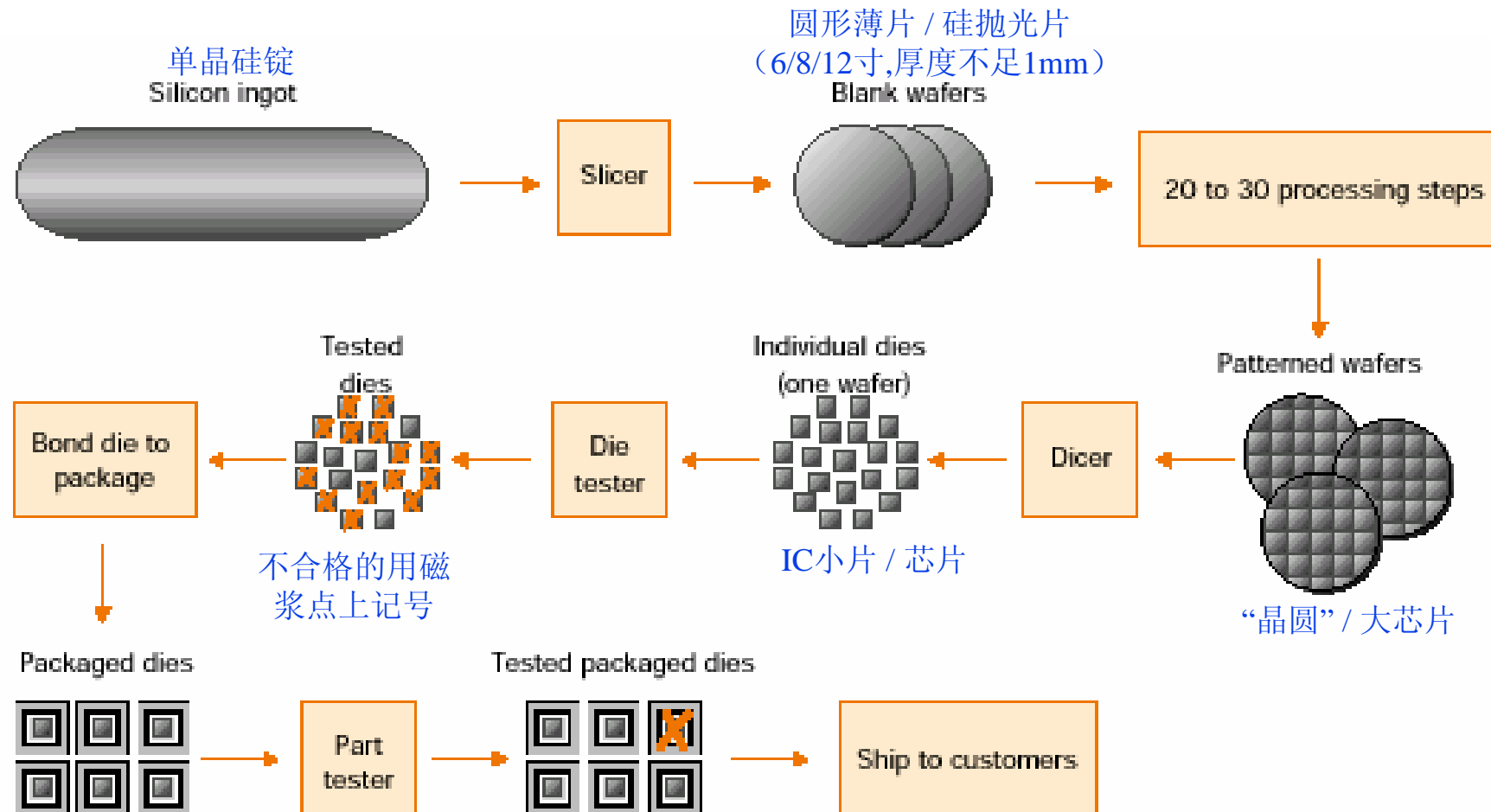


回顾：解剖计算机

How do you build systems with >1G components?



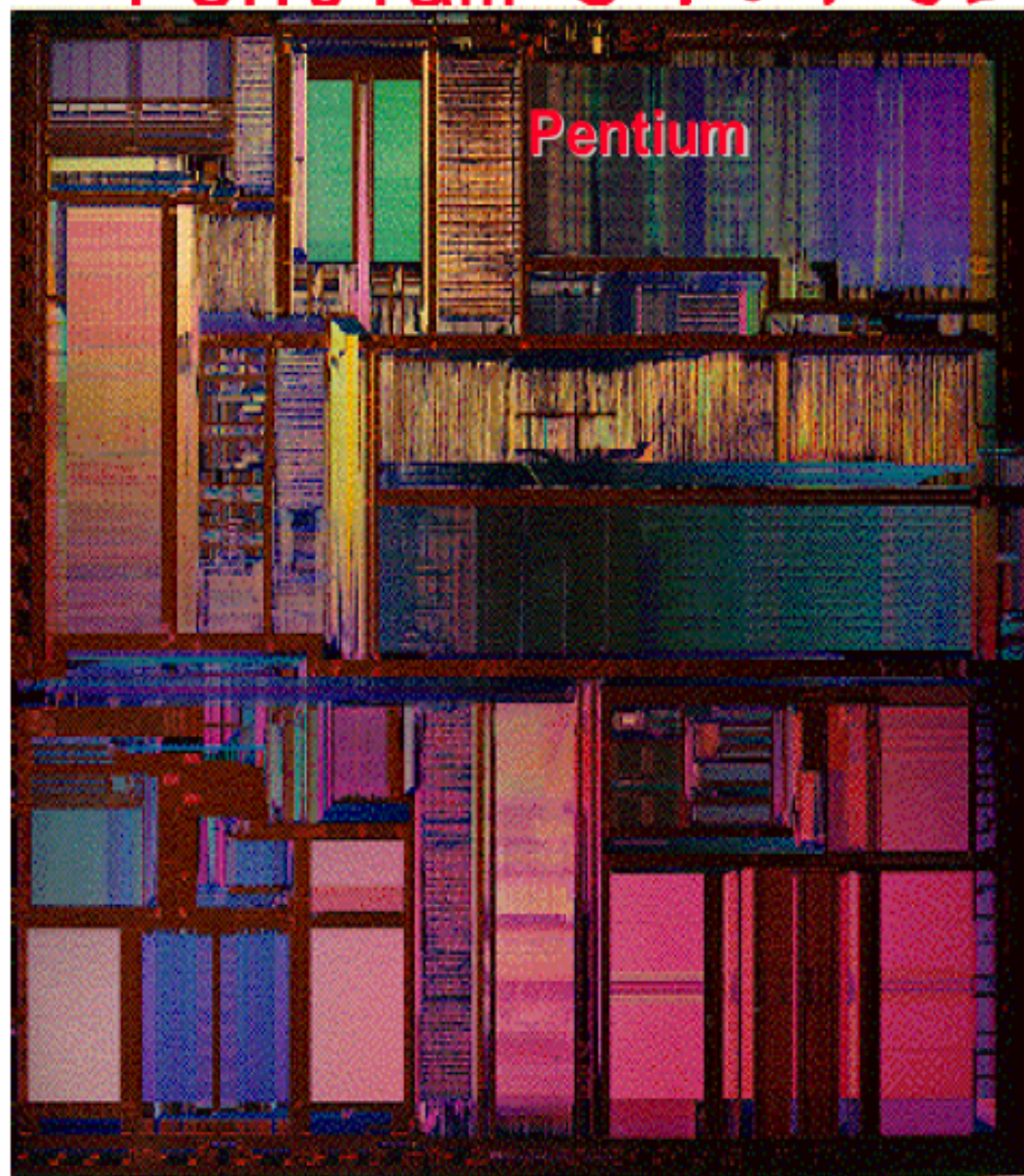
回顾：Integrated Circuits manufacturing process



封装：将芯片固定在塑胶或陶瓷基座上，把芯片上蚀刻出来的引线与基座底部伸出的引脚连接，盖上盖板并封焊成芯片

约需400多道工序！

Pentium芯片内的主要功能块



Die Area: 91 mm²

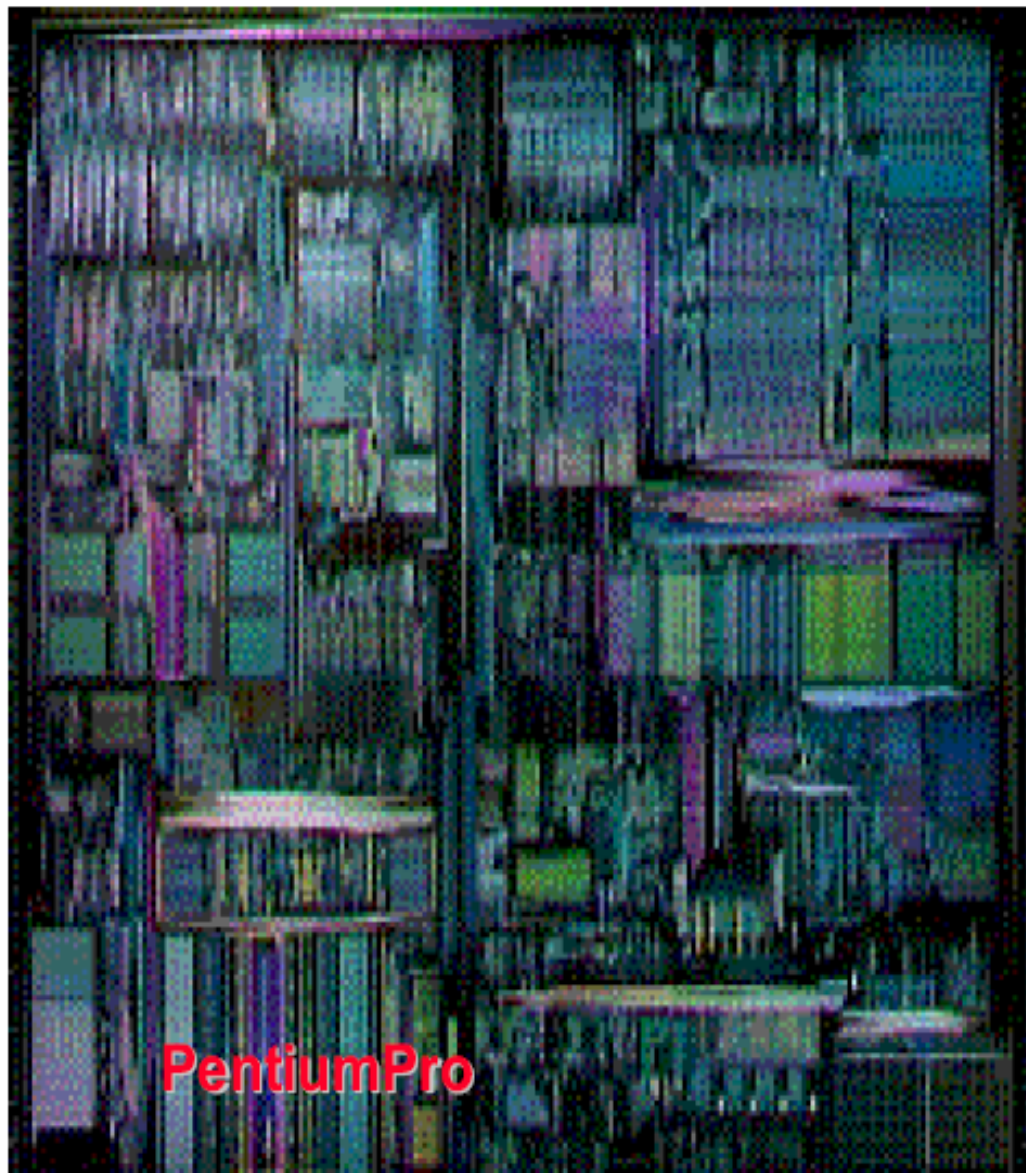
直径8 inch(200mm)的
Wafer最多可做 196个Die

≈ 3,300,000 Transistors

Cache: ≈1M Transistors

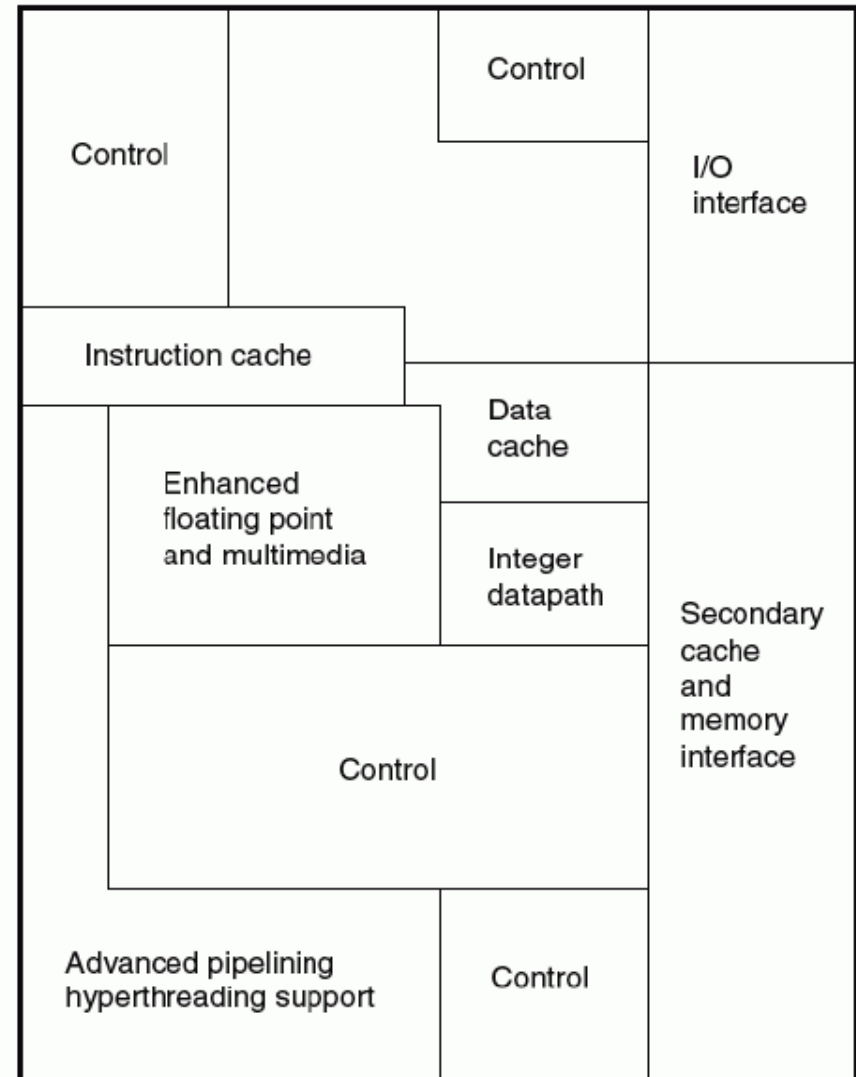
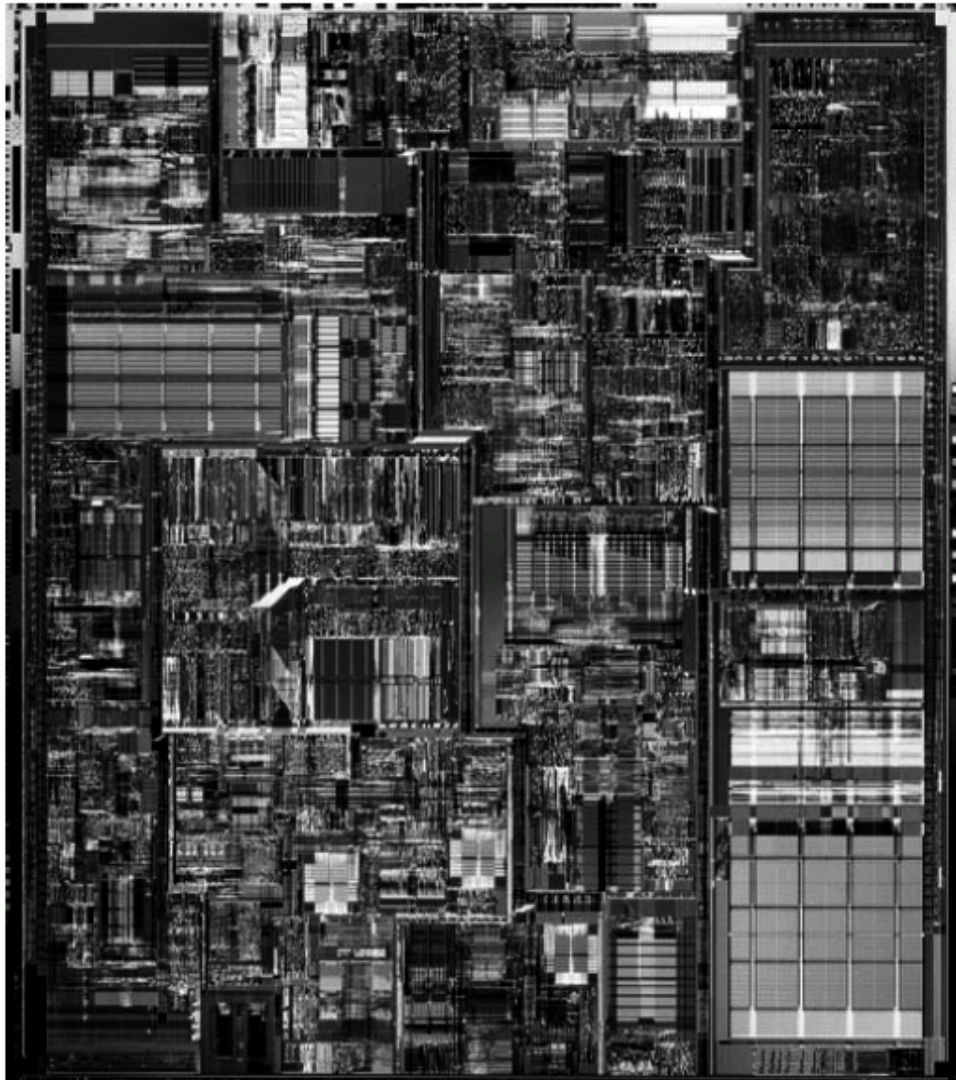
296 Pins

PentiumPro芯片内的主要功能块

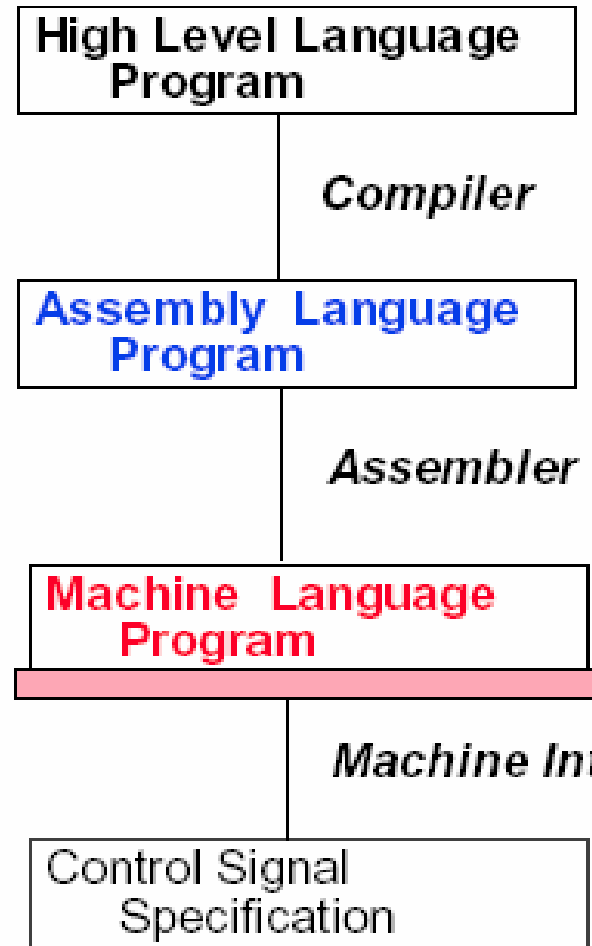


- Die Area: 306 mm²
 - 直径8 inch(200mm)的 Wafer最多可做 78个Die
 - $\approx 5,500,000$ Transistors
 - Cache: $\approx 1\text{M}$ Transistors
 - External Cache:
31M Transistors
- PentiumPro Package =
PentiumPro+ExternalCache
387 Pins

Pentium4处理器内部布局



Hardware/Software Interface



```
temp = v[k];  
v[k] = v[k+1];  
v[k+1] = temp;
```

```
lw $15, 0($2)  
lw $16, 4($2)  
sw    $16, 0($2)  
sw    $15, 4($2)
```

```
0000 1001 1100 0110 1010 1111 0101 1000  
1010 1111 0101 1000 0000 1001 1100 0110  
1100 0110 1010 1111 0101 1000 0000 1001  
0101 1000 0000 1001 1100 0110 1010 1111
```

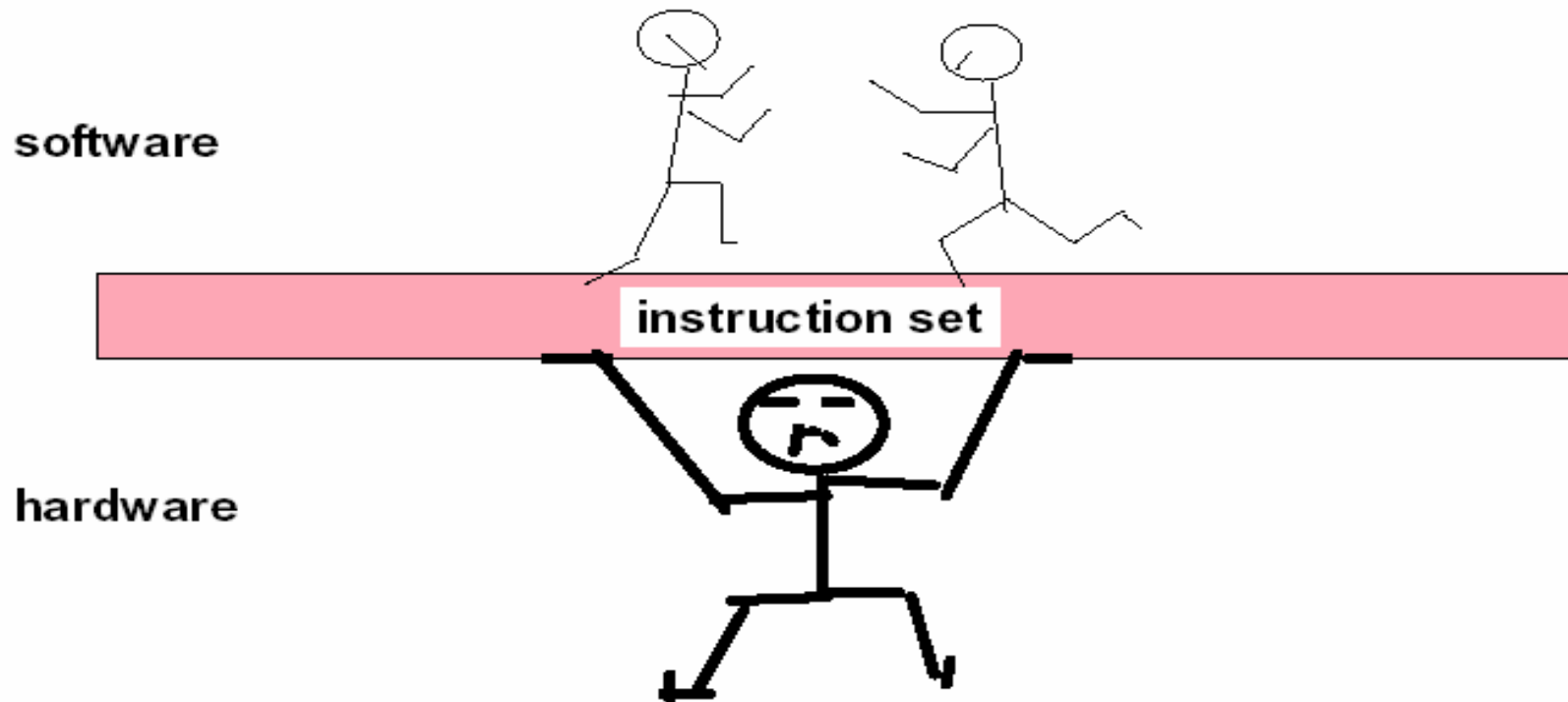
```
ALUOP[0:3] <= InstReg[9:11] & MASK
```

[Back](#)

回顾：Software

- **System software** - 简化编程过程，使硬件资源被有效利用
 - 操作系统（**Operating System**）：*硬件资源管理，用户接口*
 - 语言处理程序: **Translator & Interpreter**
 - *翻译程序（Translator , e.g Compiler, Assembler): 将高级语言源程序翻译为目标程序。如：C++, Fortran, MASM*
 - *解释程序（Interpreter）：直接对高级语言解释执行。如：BASIC, Java*
 - 其他实用程序: 如: *Linker, Debug, Loader, etc*
- **Application software** - 解决具体应用问题或完成具体应用任务
 - **Programs for processing Word/ Image/ Graphics/...**
 - **Management Information System (MIS)**
 - **Game, etc**

回顾: Hardware/Software Interface

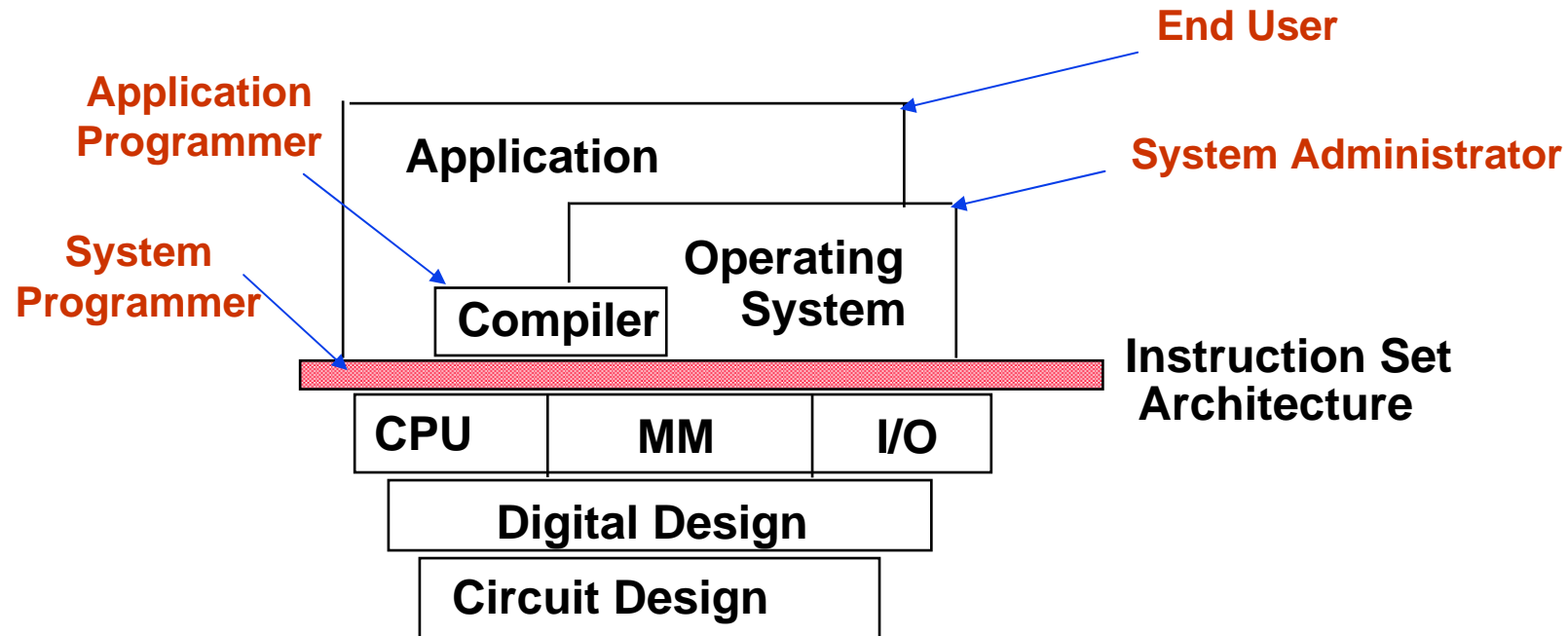


The Instruction Set Architecture (ISA): 指令集体系结构

机器语言由指令代码构成，能被硬件直接执行。

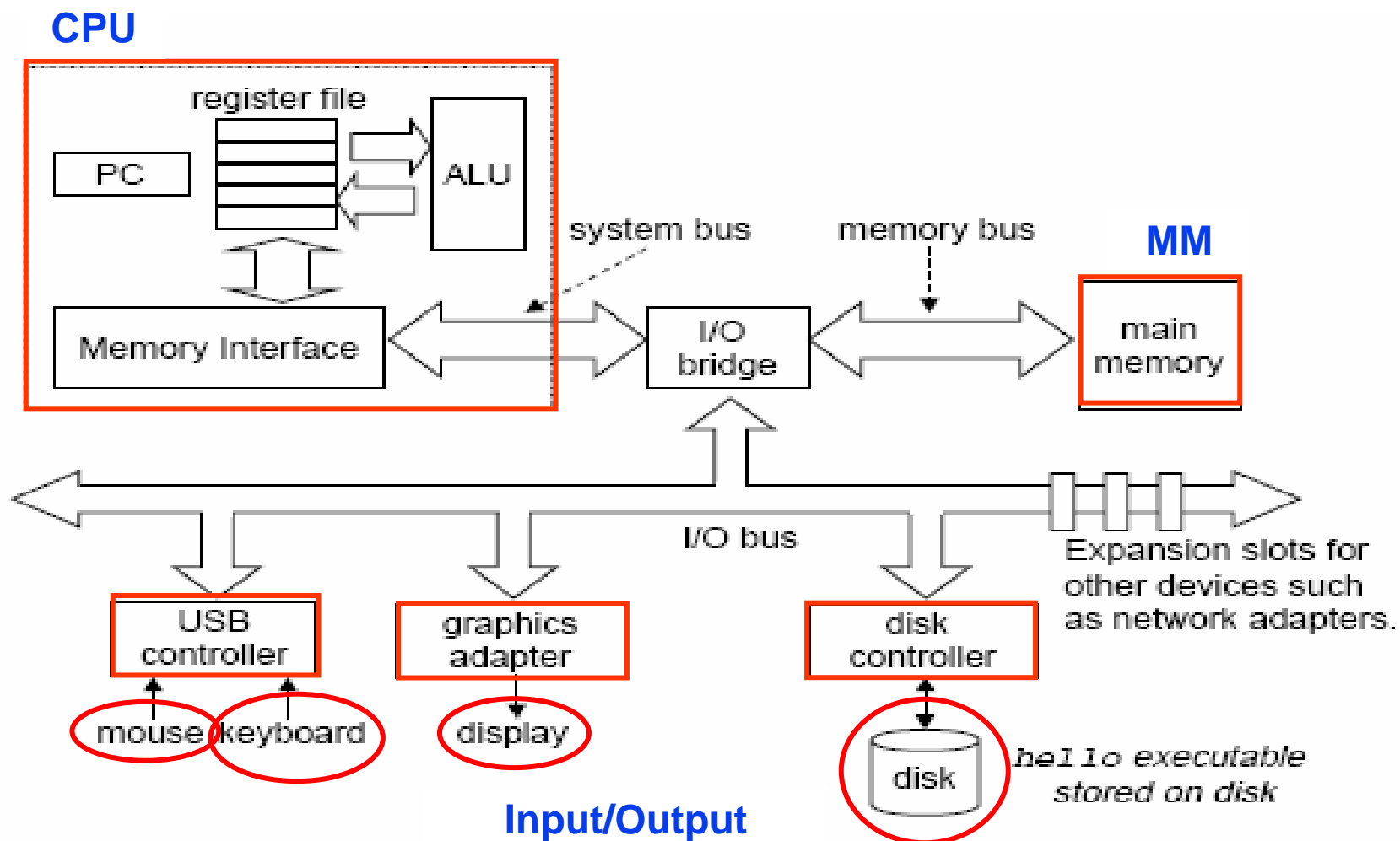
回顾: Computer Hierarchy

- Co-ordination of *levels of abstraction*



- 上图给出的是计算机系统的层次结构（指令系统是软/硬件的交界面）
- 不同用户工作在不同层次，所看到的计算机不一样

一个典型系统的硬件组成



PC: 程序计数器; **ALU:** 算术/逻辑单元; **USB:** 通用串行总线

一个典型程序的转换处理过程

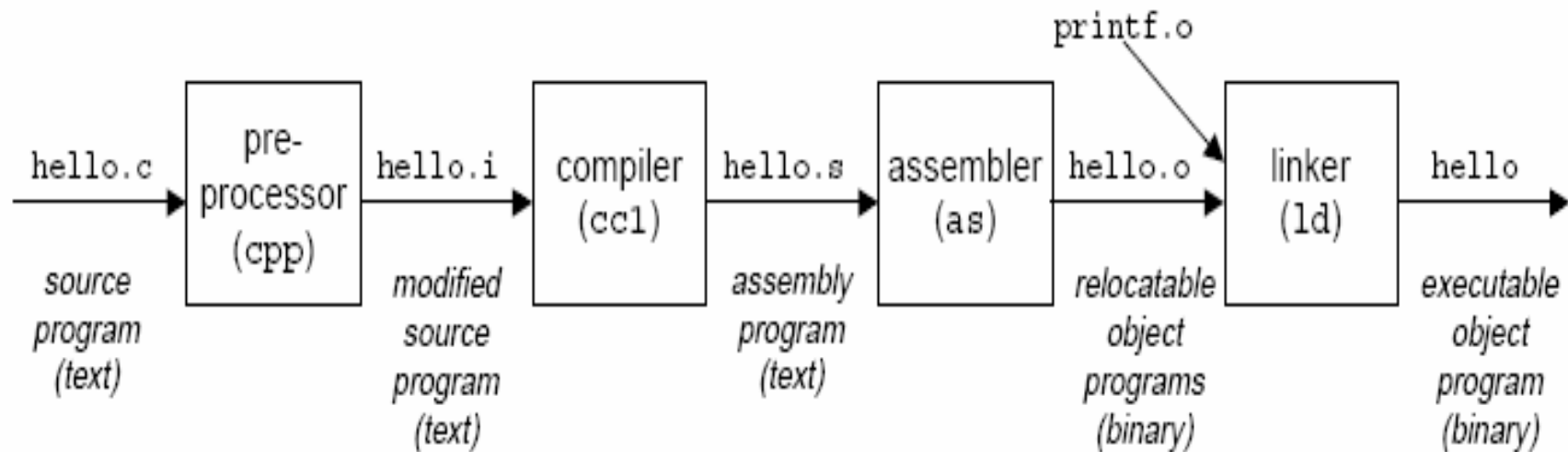
经典的“hello.c”C-源程序

```
1 #include <stdio.h>
2
3 int main()
4 {
5     printf("hello, world\n");
6 }
```

程序的功能是：
输出“hello,world”

hello.c的ASCII文本表示

```
# i n c l u d e < s p > < s t d i o .
35 105 110 99 108 117 100 101 32 60 115 116 100 105 111 46
h > \n \n i n t < s p > m a i n ( ) \n {
104 62 10 10 105 110 116 32 109 97 105 110 40 41 10 123
\n < s p > < s p > < s p > < s p > p r i n t f ( " h e l
10 32 32 32 32 112 114 105 110 116 102 40 34 104 101 108
l o , < s p > w o r l d \ n " ) ; \n }
108 111 44 32 119 111 114 108 100 92 110 34 41 59 10 125
```



Hello程序的执行过程

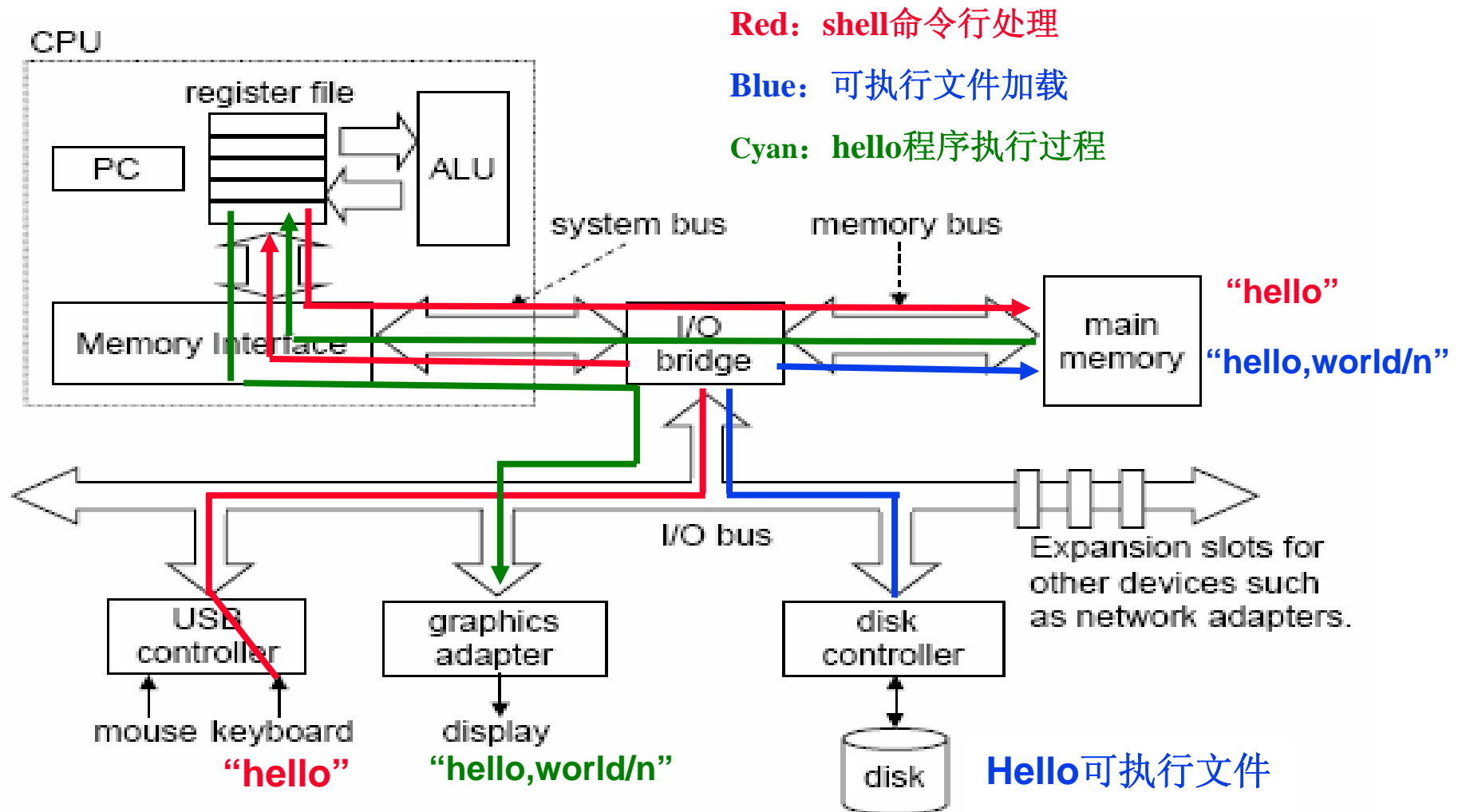
Unix系统启动可执行程序**hello**的**shell**命令行:

```
unix> ./hello [Enter]
hello, world
unix>
```

Hello程序被启动后，计算机的动作过程如下：

1. **Shell**程序读取字符串“**./hello**”中各字符到寄存器，然后存放到主存；
2. “**Enter**”键输入后，**shell**根据主存中的字符串“**hello**”到磁盘上找到特定的**hello**目标文件，将其包含的指令代码和数据（“**hello, world\n**”）从磁盘读到主存；
3. 处理器从**hello**主程序的指令代码开始执行；
4. **Hello**程序将“**hello, world\n**”串中的字节从主存读到寄存器，再从寄存器输出到显示器上

Hello程序的数据流动过程

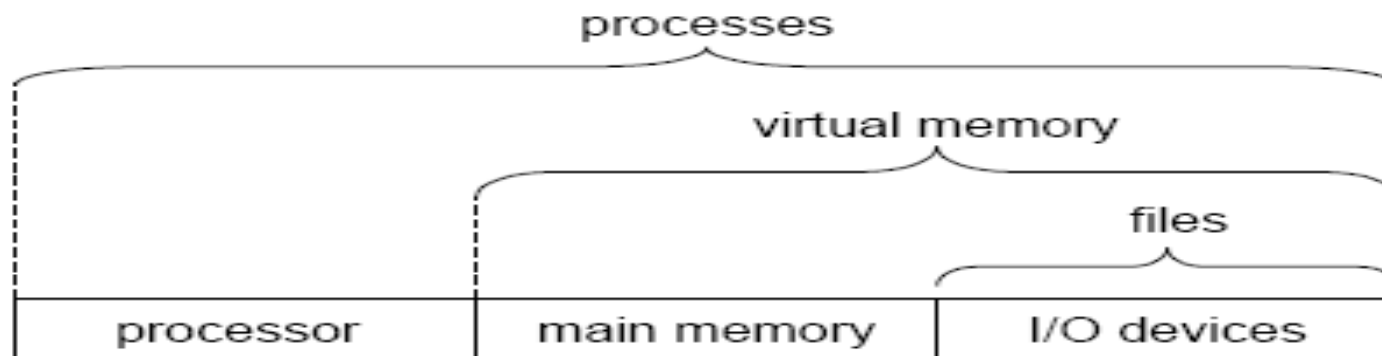


数据经常在各存储部件间传送。故现代计算机大多采用“缓存”技术！

所有过程都是在CPU执行指令所产生的控制信号的作用下进行的。

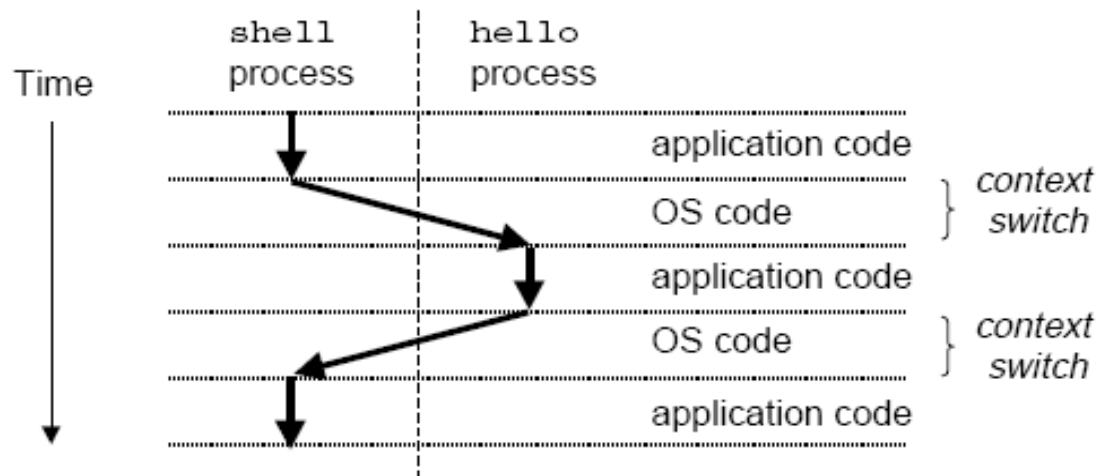
操作系统在程序执行过程中的作用

- 在**Hello**程序执行过程中，**Hello**程序本身没有直接访问键盘、显示器、磁盘和主存储器这些硬件资源，而是依靠操作系统提供的服务来间接访问。
- 操作系统是在应用程序和硬件之间插入的一个中间软件层。
- 操作系统的两个主要的作用：
 - 硬件资源管理，以达到以下两个目的：
 - 统筹安排和调度硬件资源，以防止硬件资源被用户程序滥用
 - 对于广泛使用的复杂低级设备，为用户程序提供一个简单一致的使用接口
 - 为用户使用系统提供一个操作接口
- 操作系统通过三个基本的抽象概念（进程、虚拟存储器、文件）实现硬件资源管理
 - 文件（**files**）是对**I/O**设备的抽象表示（第八章）
 - 虚拟存储器（**Virtual Memory**）是对主存和磁盘**I/O**的抽象表示（第七、八章）
 - 进程（**processes**）是对处理器、主存和**I/O**设备的抽象表



进程（processes）

- 前面例子中的**Hello**程序运行时，**Hello**程序会以为（错觉）：
 - 所有系统资源都被自己独占使用
 - 处理器始终在执行本程序的一条条指令
- 进程是操作系统对运行程序的一种抽象
 - 一个系统上可以同时运行很多进程，但每个进程都好像自己是独占使用系统
 - 实际上，操作系统让处理器交替执行很多进程中的指令
 - 操作系统实现交替指令执行的机制称为“上下文切换（**context switching**）”
- 进程的上下文
 - 进程运行所需要的所有状态信息，例如：**PC**、寄存器堆的当前值、主存的内容、段/页表
 - 系统中有一套状态单元存放当前运行进程的上下文
- 上下文切换过程（任何**时刻**，系统中只有一个进程**正在运行**）
 - 上下文切换指把正在运行的进程换下，换一个新进程到处理器执行，上下文切换时，必须保存换下进程的上下文，恢复换上进程的上下文

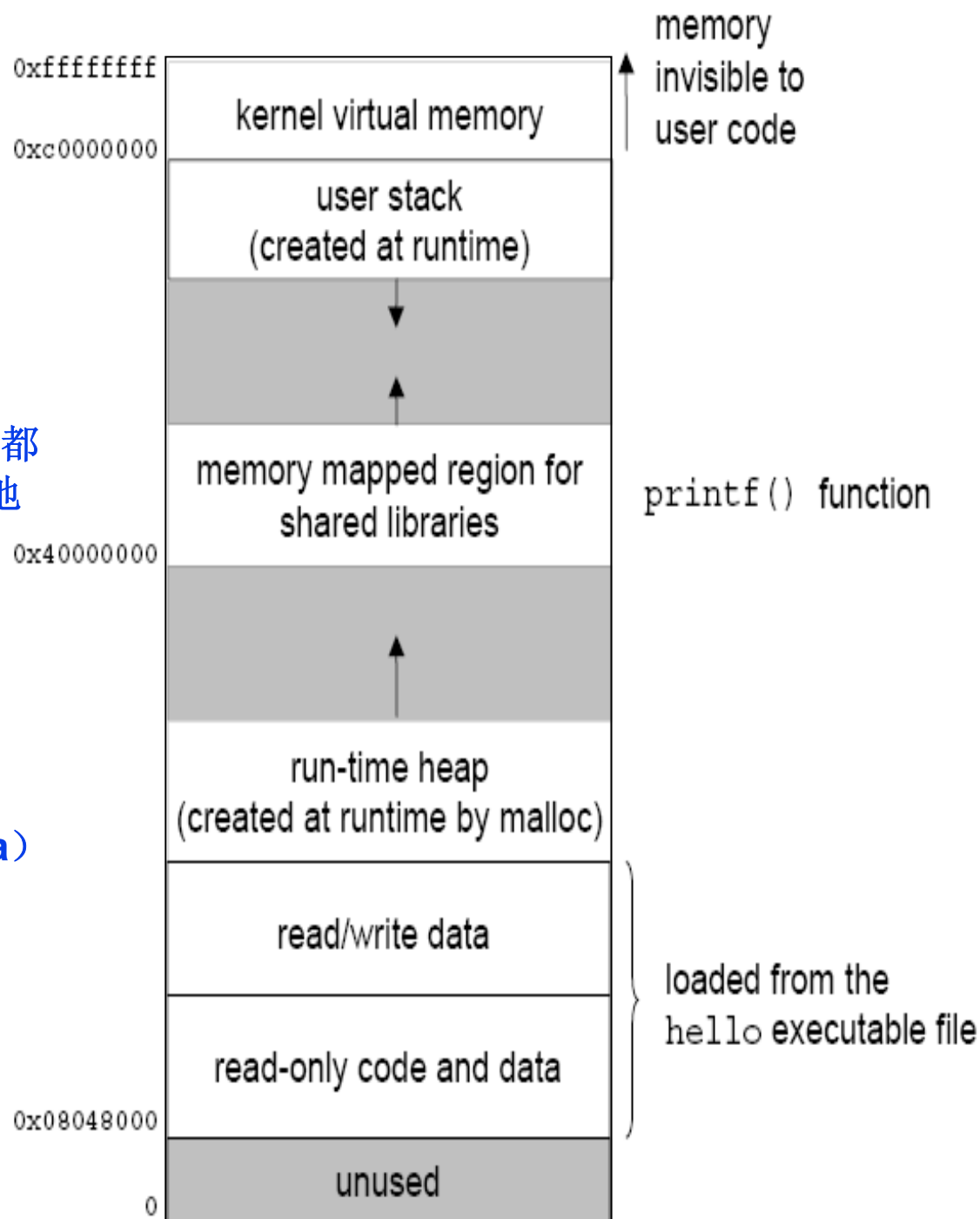


开始，**Shell**进程等待命令行输入
输入“**Hello**”后**Shell**进行系统调用
OS保存**shell**上下文，创建并换入**hello**进程
Hello进程中止时，进行系统调用
OS恢复**shell**进程上下文，并换入**shell**进程

由于在一个进程的整个生命周期中，有不同进程在处理器交替运行，所以运行时间很难准确、重复测量。

虚拟存储器

- ° 虚存为每个进程提供了一个假象
 - 好像每个进程都独占使用主存，并且主存空间极大
- ° 虚存是主存和磁盘I/O设备的抽象
 - **OS**使每个进程看到的存储空间都是一致的，称为虚拟（逻辑）地址空间
- ° **Linux**操作系统的虚拟地址空间（其他**Unix**系统的设计类此）
 - 代码（**Code**）
 - 只读数据（**Read-only Data**）
 - 可读写数据（**Read/Write Data**）
 - 堆（**heap**）
 - 共享库（**Shared Libraries**）
 - 用户栈（**User Stack**）
 - 内核（**Kernel**）

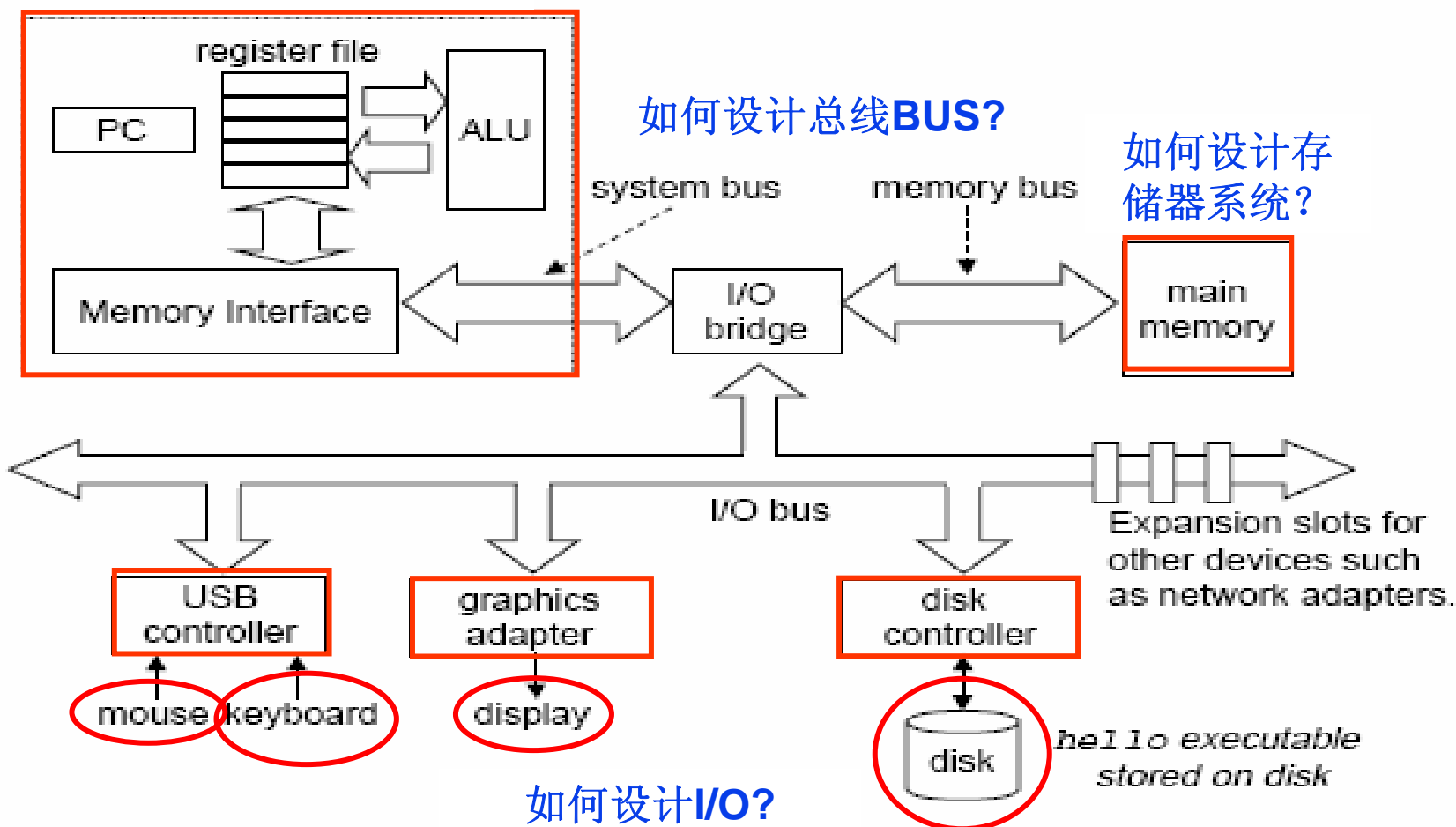


文件

- **I/O**设备控制复杂、种类繁多、千差万别
- 文件是对**I/O**设备的抽象表示
 - 每个**I/O**设备（包括键盘、磁盘、显示器、网络等）都被抽象成文件
 - 文件就是字节序列
 - 把所有**I/O**设备都看成是字节序列，它们就具有一个简单、一致的使用方式
 - **OS**为所有设备向用户程序提供了一组统一的文件读/写系统函数调用
 - 用户程序的所有输入/输出操作都通过使用文件读/写系统调用来实现
- 好处：
 - 使用户程序能够统一看待系统中的各种**I/O**设备
 - 对用户程序屏蔽各种**I/O**设备的硬件细节，使应用程序员无需了解硬件技术
 - 使同一个用户程序可以在采用不同**I/O**设备技术的系统上运行

该课程的主要学习内容

如何设计高性能CPU?



信息（指令和数据）在计算机中如何表示？

指令系统如何设计？

回顾：计算机发展简史

- 第一代：真空管（电子管Vacuum Tube）1946～57年
 - 46年诞生第1台电子计算机ENIAC
 - 体积庞大，重30吨，有18000多个真空管组成，5000次加法/秒。
 - 采用十进制表示/运算，其存储器由20个累加器组成，每个累加器可存10位十进制数，每一位数由10个真空管表示。
 - 采用手动编程，通过设置开关和插拔电缆来实现。
 - 冯·诺依曼机（Von Neumann Machine）
 - 45年冯·诺依曼（图灵也同时）提出“存储程序(Stored-program)”思想，并于46年开始设计“存储程序”计算机，被称为IAS计算机。
 - “存储程序”思想：

将事先编好的程序和原始数据送入主存中，然后启动计算机工作。计算机应能在不需操作人员干预下，自动完成逐条取出指令和执行指令的任务。

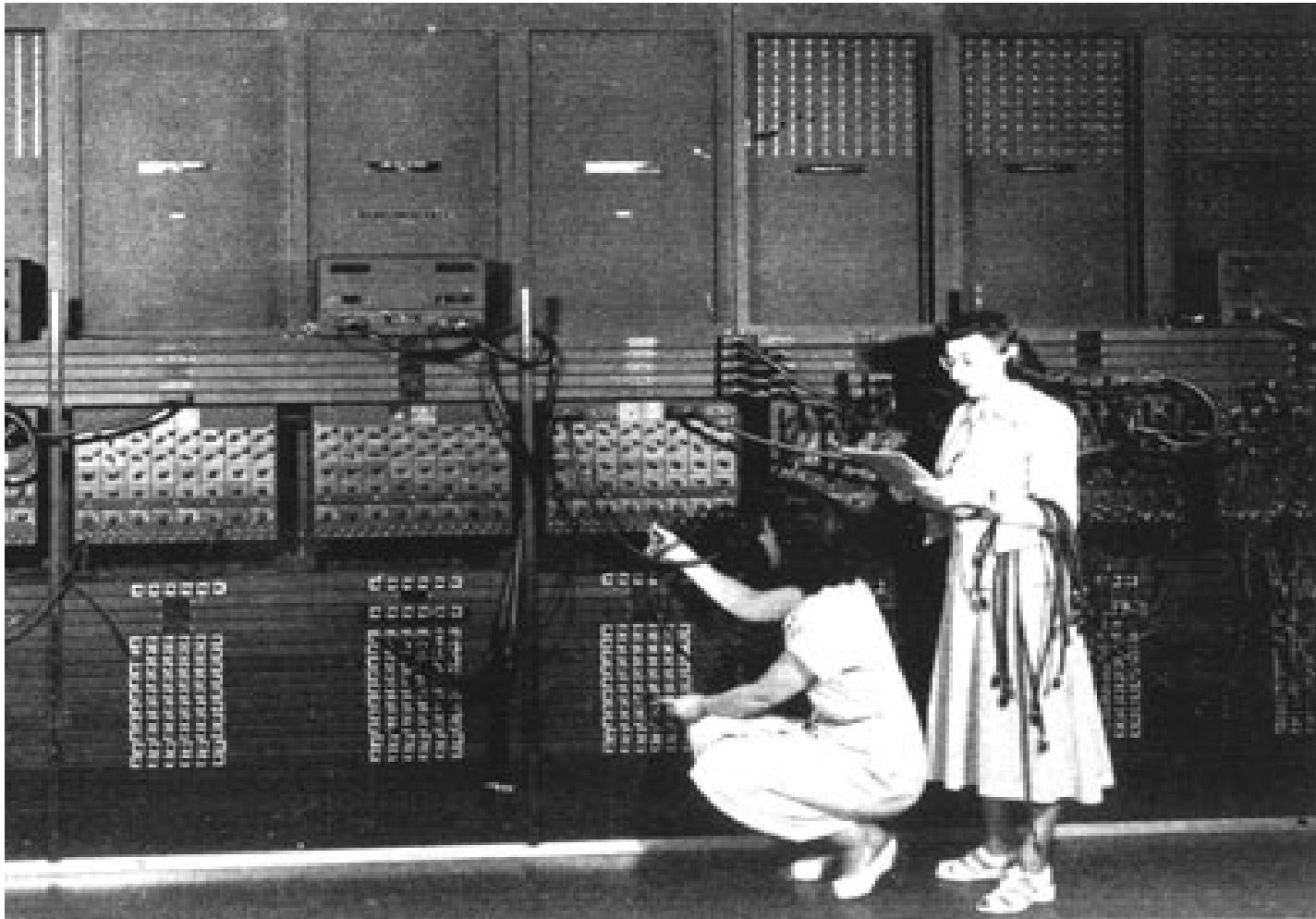
SKIP

The First Generation: Vacuum Tube Computers (1946 - 1957)



The first *general-purpose computer* - **ENIAC**

ENIAC---Non von Neumann Model



- Electronic Numerical Integrator and Computer (ENIAC)
- John Mauchly and J. Presper Eckert
- University of Pennsylvania, 1946

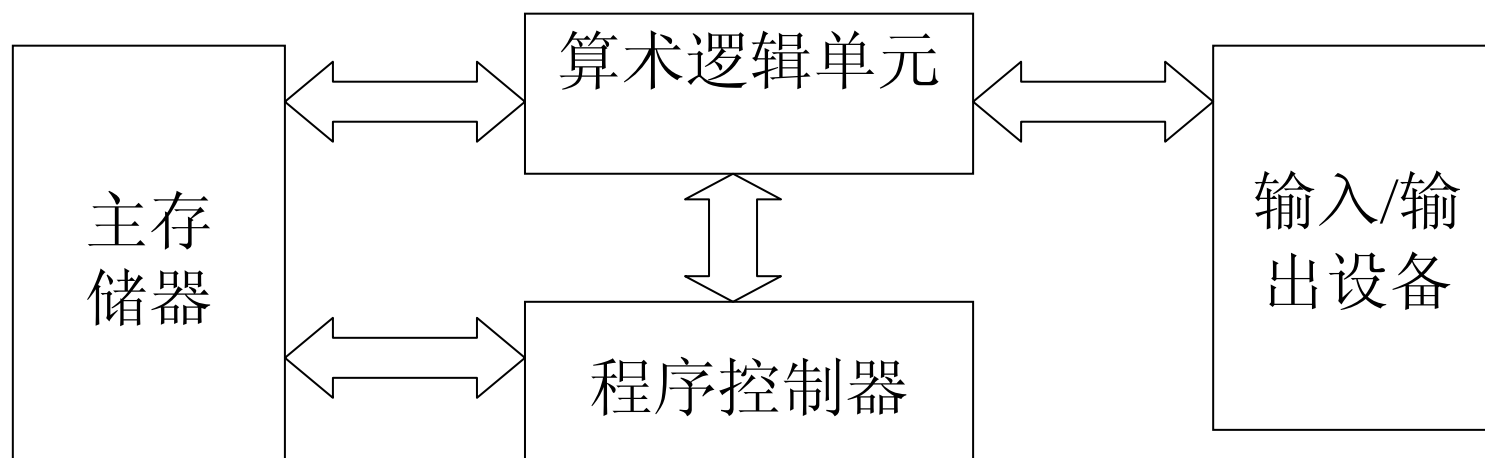
[BACK](#)

2008年3月28日星期五

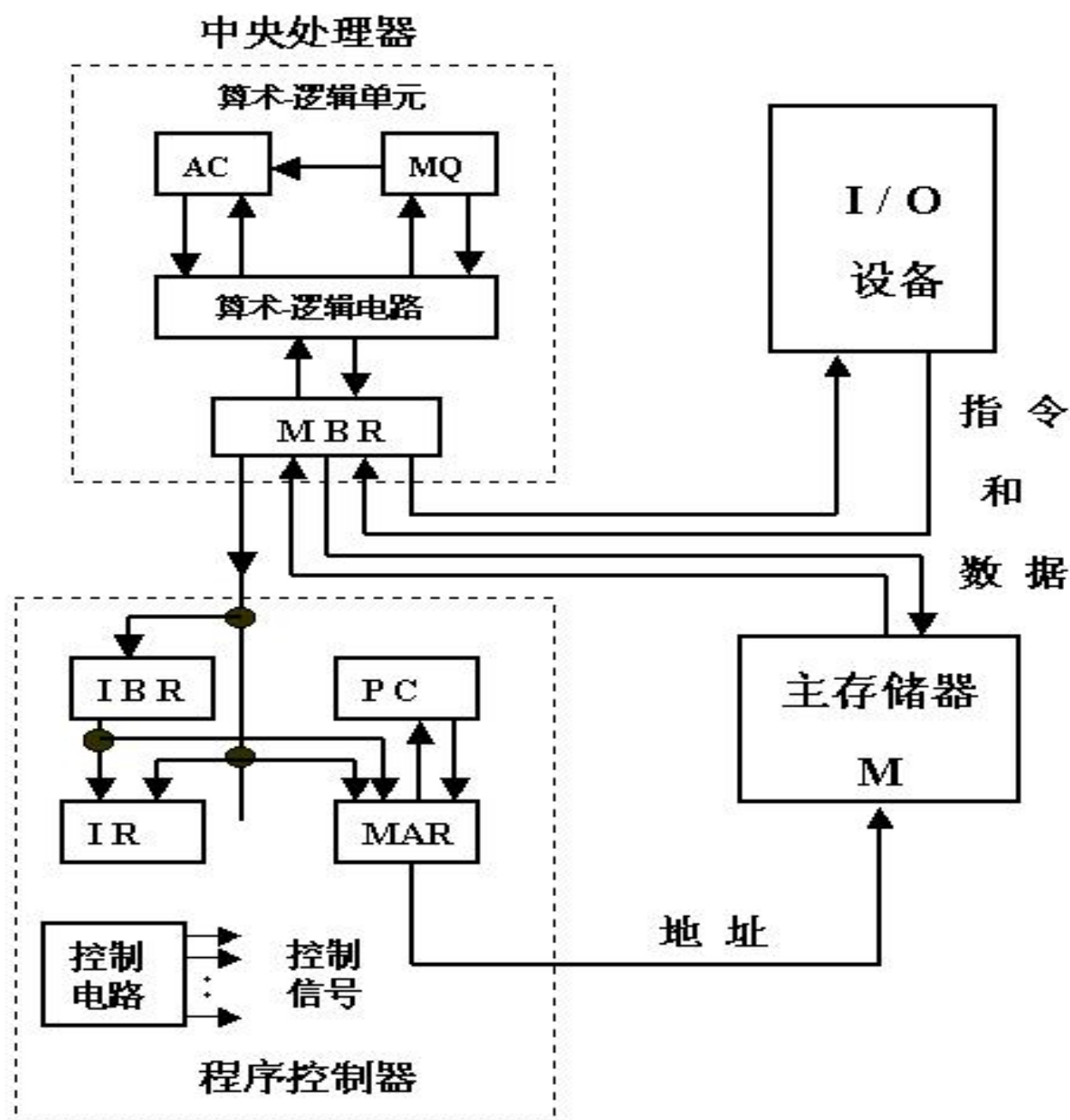
IAS计算机-通用计算机原型

1946年由冯·诺依曼及其同事在普林斯顿高级研究院IAS设计以“存储器”为中心，其后通用机原型

IAS计算机的一般结构，它包含5个部件：



IAS (the Institute for Advance Study at Princeton)



IAS 计算机结构扩展图

[BACK](#)

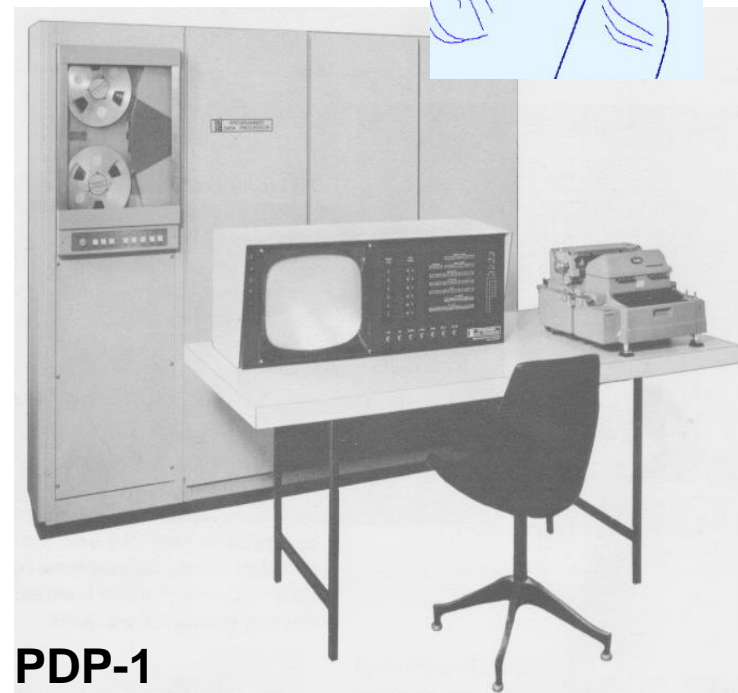
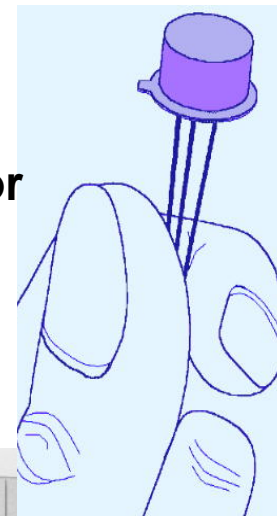
回顾：冯·诺依曼结构的主要思想

1. 计算机应由运算器、控制器、存储器、输入设备和输出设备五个基本部件组成。
2. 各基本部件的功能是：
 - ❖ **存储器**不仅能存放数据，而且也能存放指令，形式上两者没有区别，但计算机应能区分数据还是指令；
 - ❖ **控制器**应能自动执行指令；
 - ❖ **运算器**应能进行加/减/乘/除四种基本算术运算，并且也能进行一些逻辑运算和附加运算；
 - ❖ 操作人员可以通过**输入设备**、**输出设备**和主机进行通信。
3. 内部以**二进制表示**指令和数据。每条指令由操作码和地址码两部分组成。操作码指出操作类型，地址码指出操作数的地址。由一串指令组成程序。
4. 采用“**存储程序**”工作方式。

回顾：计算机发展简史

- 第二代：晶体管 1958~64年
 - 元器件：逻辑元件采用晶体管，内存由磁芯构成，外存为磁鼓与磁带。
 - 特点：变址，浮点运算，多路存储器，I/O 处理机，中央交换结构(非总线结构)。
 - 软件：使用高级语言，提供了系统软件。
 - 代表机种：IBM 7094 (scientific)、1401 (business)和 DEC PDP-1

晶体管：
Transistor



DEC PDP-1

回顾：计算机发展简史

- 第三代：SSI/MSI 1965~71年
 - 元器件：逻辑元件与主存储器均由集成电路（**Integrated Circuit**）实现。
 - 特点：微程序控制，**Cache**，虚拟存储器，流水线等。
 - 代表机种：**IBM 360**和**DEC PDP-8**（大/巨型机与小型机同时发展）
 - 巨型机(**Supercomputer**): **Cray-1**
 - 大型机(**Mainframe**): **IBM360**系列
 - 小型机(**Minicomputer**): **DEC PDP-8**



回顾： IBM System/360系列计算机

- IBM公司于1964年研制成功
- 引入了“兼容机”（“系列机”）概念
 - 兼容机的特征：
 - 相同的或相似的指令集
 - 相同或相似的操作系统
 - 更高的速度
 - 更多的I/O端口数
 - 更大的内存容量
 - 更高的价格

低端机指令集是高端机的一个子集，称为“向后兼容”。功能相同，而性能不同。



问题1：引入“兼容机”有什么好处？

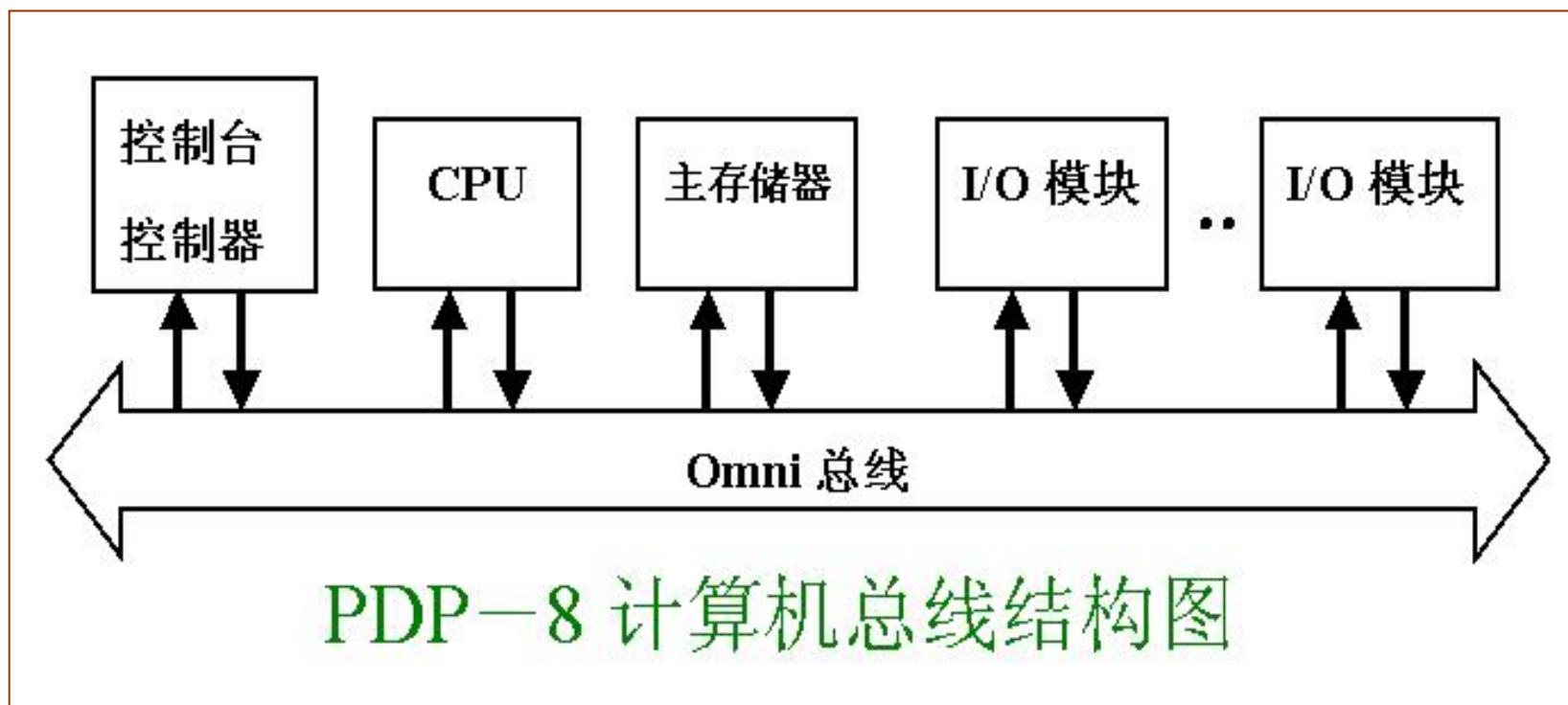
问题2：实现“系列机”的关键是什么？

回顾：DEC公司的PDP-8机

- 同在**64**年出现。与**IBM 360**相比，价格更低、更小巧，因而被称为小型机（**Minicomputer**）
- **PDP-8**“创造了小型机的概念，并使之成为数十亿美元的工业”，使**DEC**成为了最大的小型机制造商。
- 主要特点：首次采用总线结构。

Omnibus总线包含了**96**个独立的信号通道，用以传送控制、地址和数据信号。这种结构具有高度的灵活性，允许将模块插入总线以形成各种配置。

回顾：PDP-8/E计算机系统框图



问题：“总线结构”有什么好处？

回顾：计算机发展简史

- 以后几代（标准、意见不一）
 - （注：有称第四代是**VLSI**，从**80**年代开始；也有称第四代是**LSI**，从**72**年开始；有的又分成**LSI**时代和**VLSI**时代）
- （第四代：**LSI/VLSI/ULSI 1972～至今**）
 - 微处理器和半导体存储器的技术发展迅猛，微型计算机出现。
使计算机以办公设备和个人电脑的方式走向普通用户。
 - 半导体存储器
 - **70**年**Fairchild**公司生产出第一个相对大容量半导体存储器
 - **74**年位价格低于磁芯的半导体存储器出现，并快速下跌
 - 从**70**年起，存储密度呈**4**倍提高（几乎是每**3**年）。
 - 微处理器
 - 微处理器芯片密度不断增加，使**CPU**中所有元件放在一块芯片上成为可能。**71**年开发出第一个微处理器芯片**4004**。
 - 特点：共享存储器，分布式存储器及大规模并行处理系统

回顾：计算机的五个发展阶段

代号	大致年份	技术标志	代表速度(每秒执行指令数)
1	1946—1957	真空管	40,000
2	1958—1964	晶体管	200,000
3	1965—1971	中小规模集成电路	1,000,000
4	1972—1977	大规模集成电路	10,000,000
5	1978—	超大规模集成电路	100,000,000

摘自《**COMPUTER ORGANIZATION AND ARCHITECTURE Design for Performance**》William Stallings

除了元器件外，系统结构的特点是换代的一个重要标志

1965-75期间，元器件延迟降为**1/10**，但指令时间却降为**1/100**。这说明了计算机结构改进的重要性。

Moore's Law

Moore's Law



- The number of transistors on a microchip doubles about every 18-24 months,
- The speed of a microprocessor doubles about every 18-24 months,
- The price of a microchip drops about 48% every 18-24 months,
 - ➔ assuming the performance metric (processor speed or memory capacity) of the chip stays the same.
- **Official Definition of Moore's Law:**

<http://www.intel.com/intel/museum/25anniv/hof/moore.htm>

Computer Arch. = Instruction Set Arch. + Organization

Computer Design

```
graph TD; A[Computer Design] --> B[Instruction Set Design]; A --> C[Computer Hardware Design];
```

Instruction Set Design

- Machine Language
- **Compiler View**
- **"Computer Architecture"**
- **"Instruction Set Processor"**

"Building Architect"

“建筑设计师”

功能定义与设计

Computer Hardware Design

- Machine Implementation
- **Logic Designer's View**
- **"Processor Architecture"**
- **"Computer Organization"**

"Construction Engineer"

“建造师”

考虑用什么材料，如何布线等

Instruction Set Architecture

... the attributes of a [computing] system as seen by the programmer, i.e. the conceptual structure and functional behavior, as distinct from the organization of the data flows and controls the logic design, and the physical implementation.

Amdahl, Blaw, and Brooks, 1964

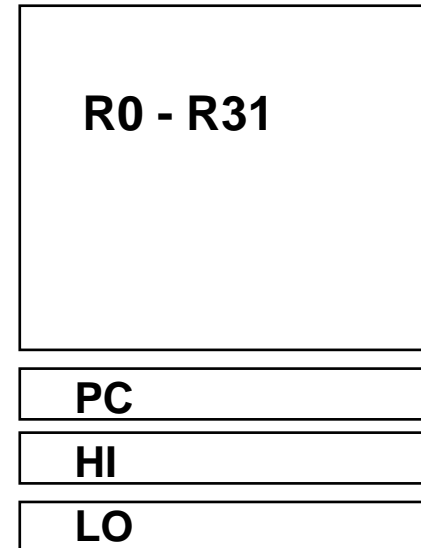
主要包括：

- Organization of Programmable Storage (程序员可见存储的组织)
(如：寄存器的个数、名称、长度；内存单元的长度、主存地址长度等等)
- Data Types & Data Structures (数据类型和结构) :
 Encodings & Representations (编码和表示)
- Instruction Formats (指令格式)
- Instruction (or Operation Code) Set (操作码集合，即指令功能)
- Modes of Addressing and Accessing Data Items and Instructions
(寻址方式和数据 / 指令存取方式)
- Exceptional Conditions (异常条件)

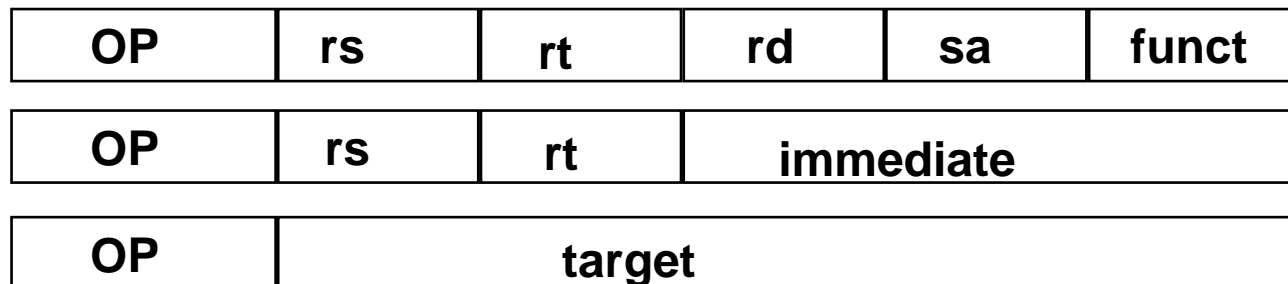
MIPS R3000 Instruction Set Architecture

◦ Instruction Categories

- Load/Store
- Computational
- Jump and Branch
- Floating Point
 - coprocessor
- Memory Management
- Special



Instruction Format



Organization

ISA Level

Logic Designer's View

FUs & Interconnect

Choreography of FUs to realize the ISA （为了实现ISA，如何安排功能部件）

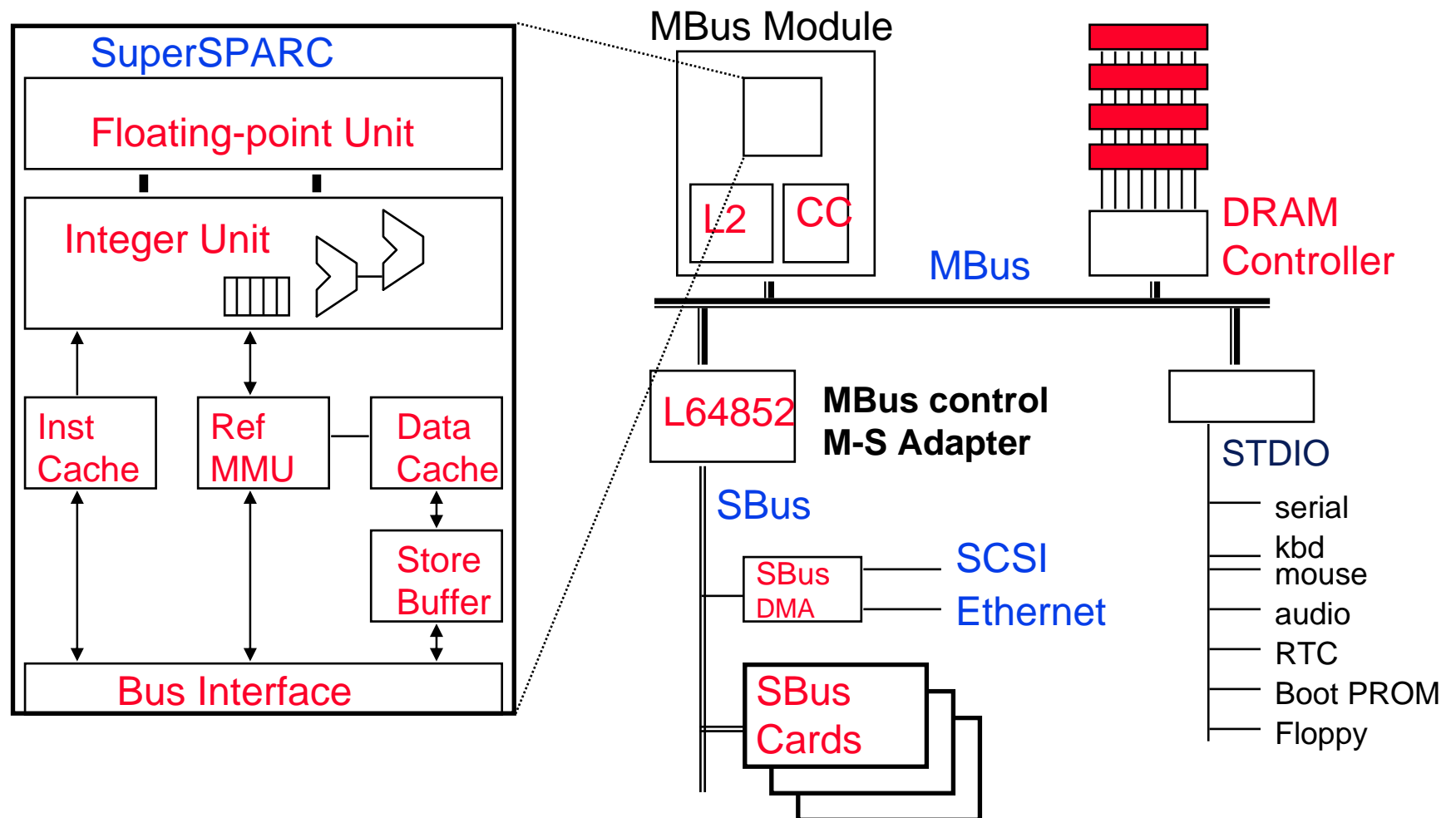
Register Transfer Level Description （寄存器传送级的描述）

主要包括：

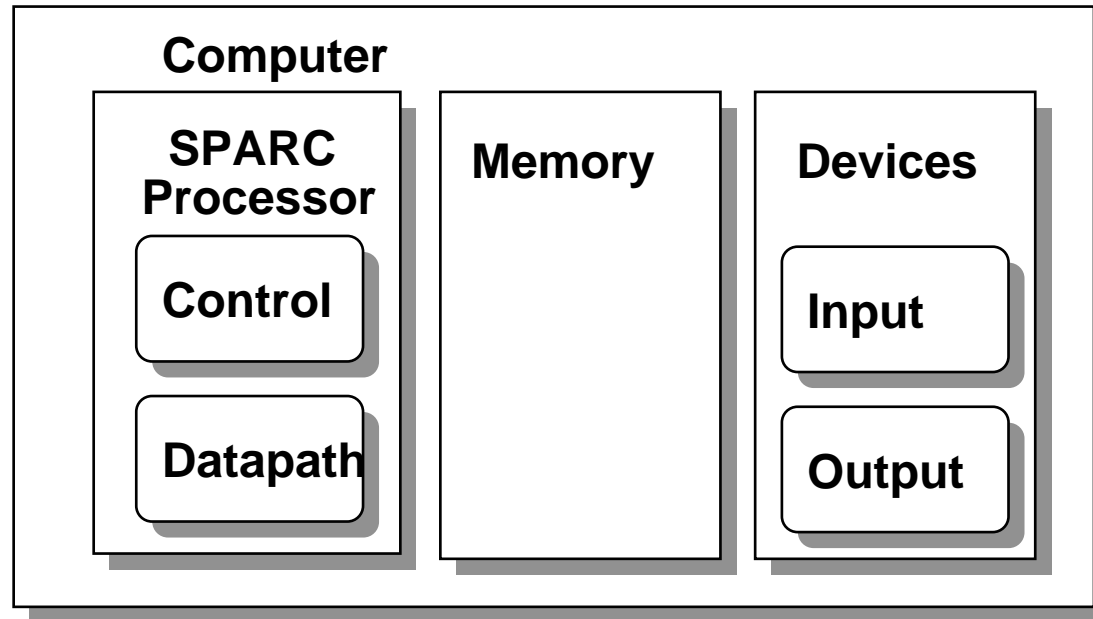
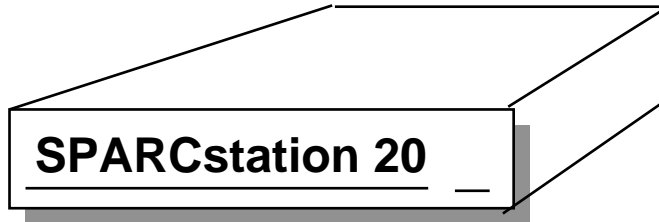
- **Capabilities & Performance Characteristics of Principal Functional Units**
（主要功能部件的能力和工作特性）
（e.g., Registers, ALU, Shifters, Memory, Cache, etc.）
- **Ways in which these components are interconnected**
（互连方式）
- **nature of information flows between components**
（部件间的信息流动方式）
- **logic and means by which such information flow is controlled.**
（部件间信息流动的控制逻辑和控制方法）

Example Organization: Sun SPARCstation20

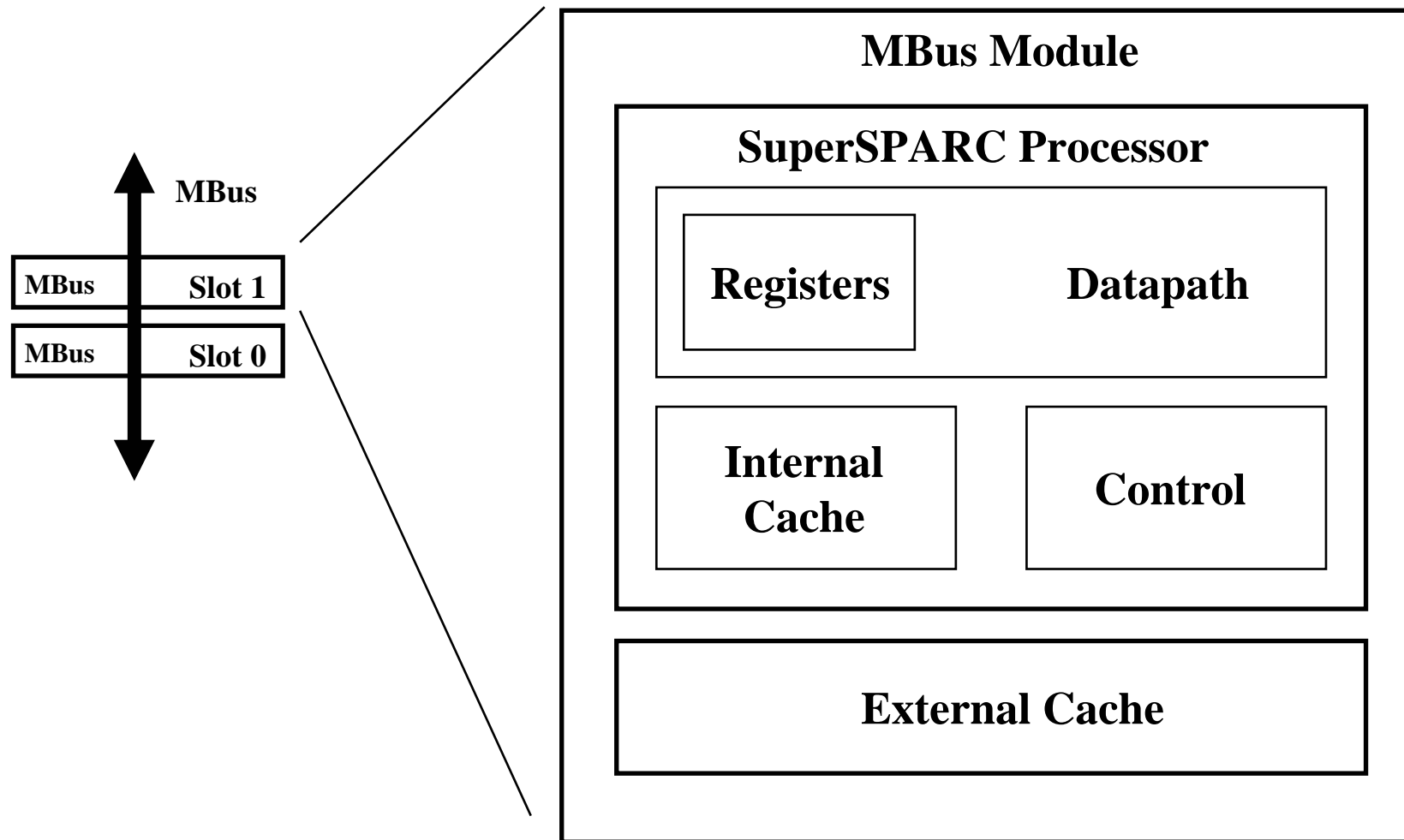
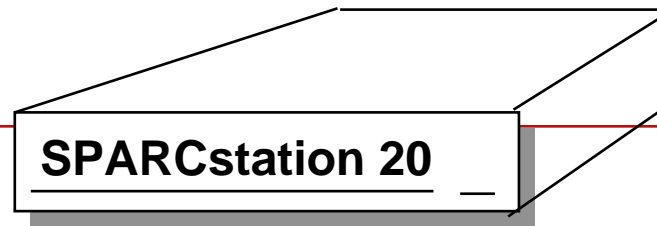
- TI SuperSPARC™ TMS390Z50 in Sun SPARCstation20



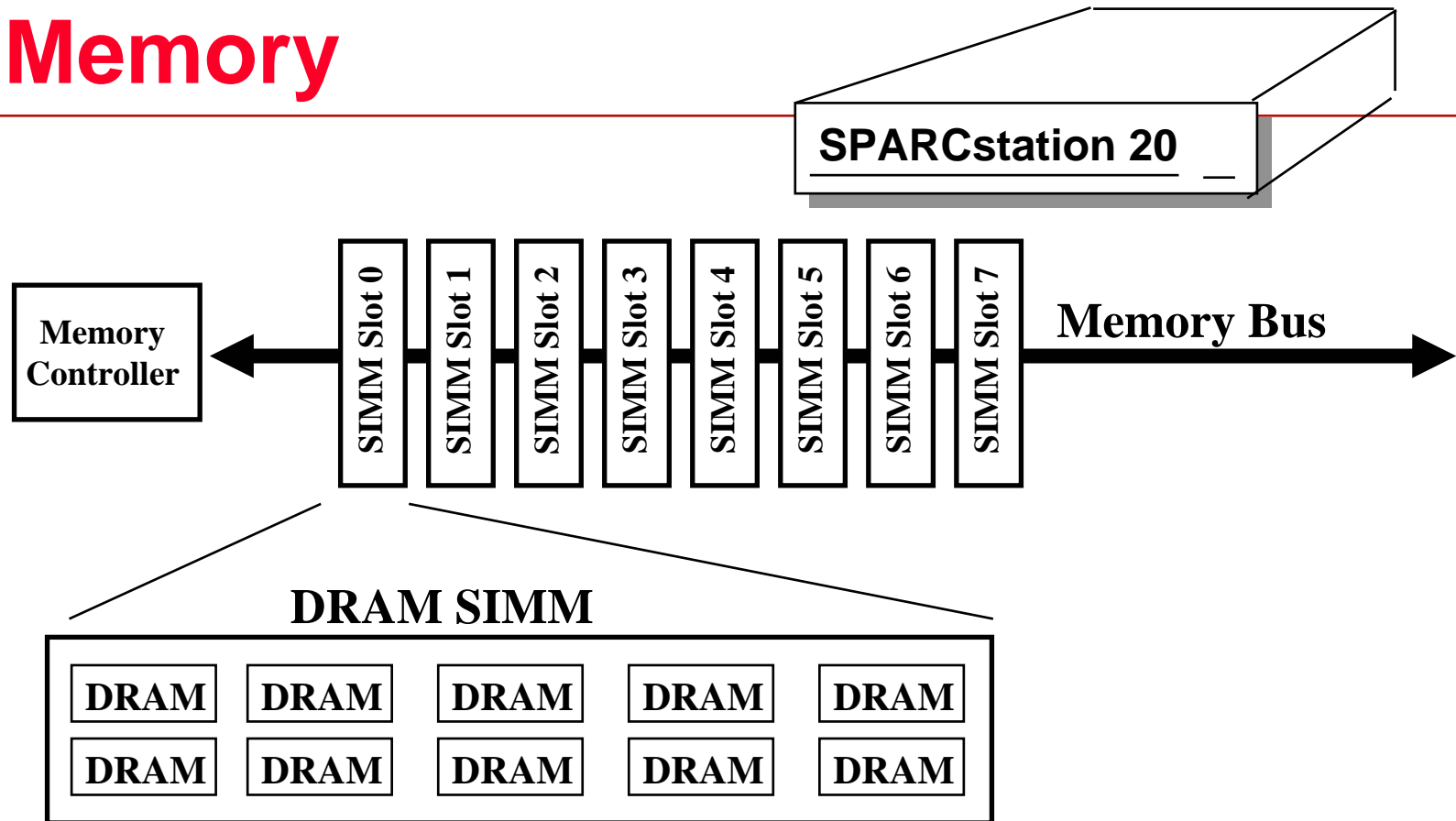
Levels of Organization



Processor and Caches

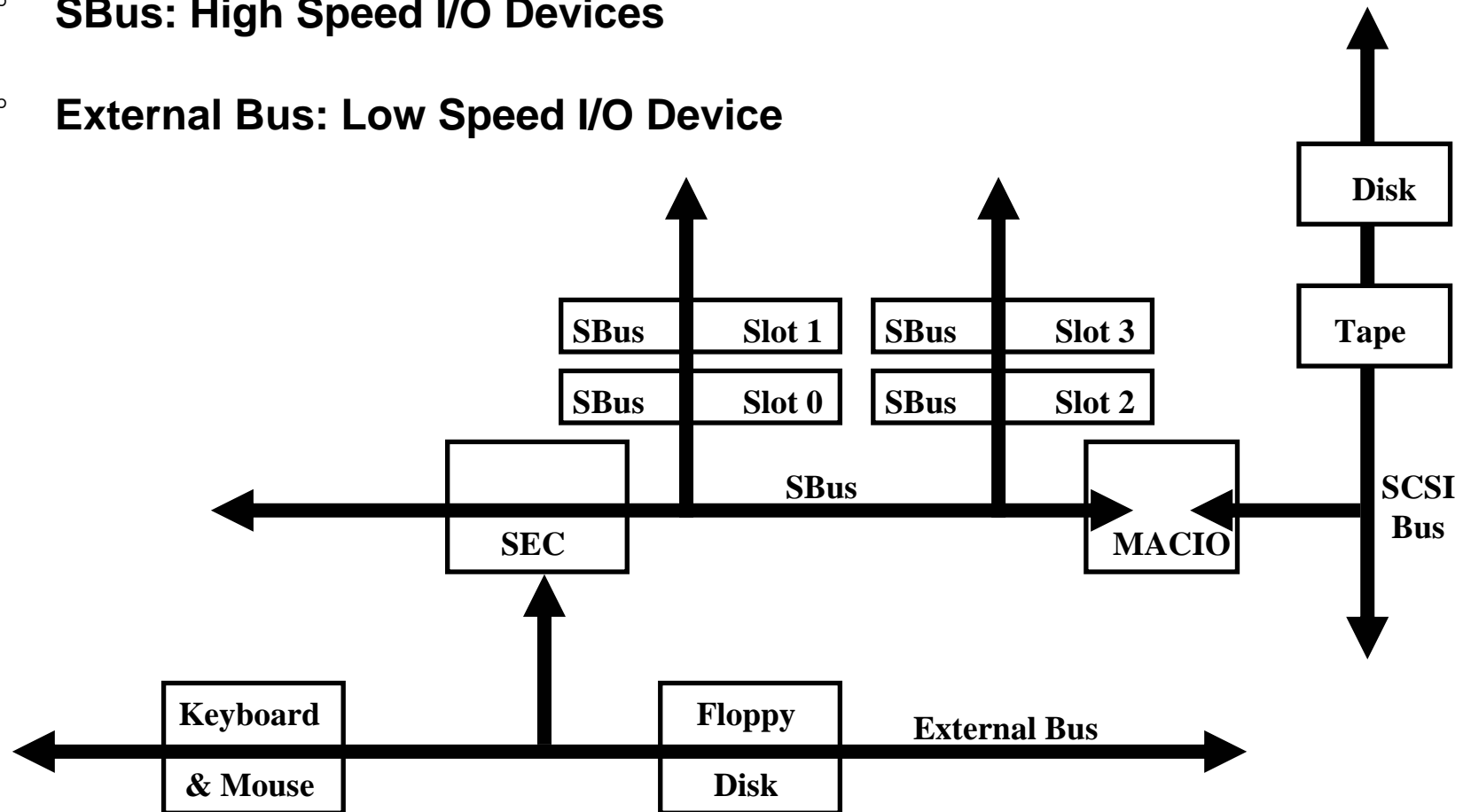
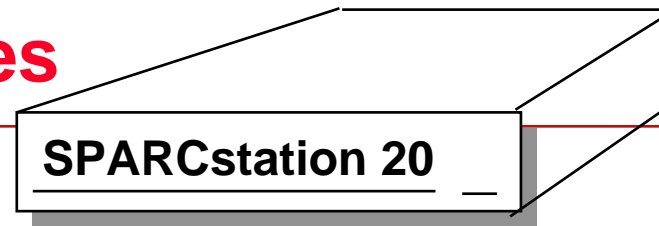


Memory

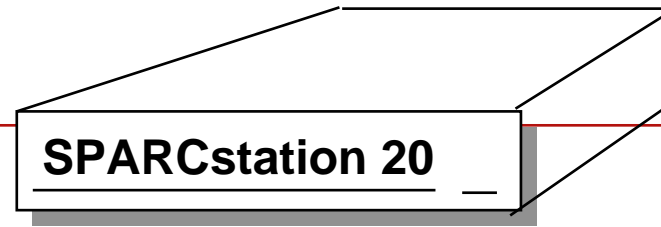


Input and Output (I/O) Devices

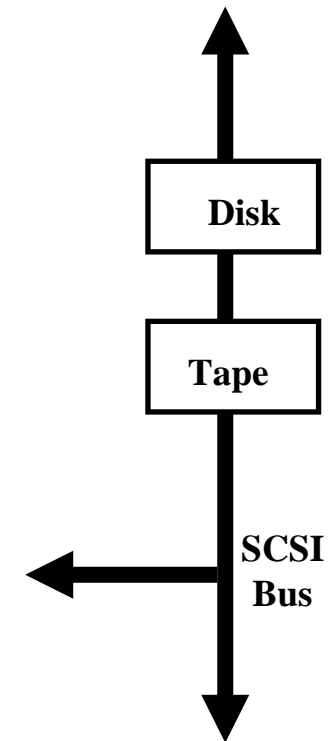
- **SCSI Bus: Standard I/O Devices**
- **SBus: High Speed I/O Devices**
- **External Bus: Low Speed I/O Device**



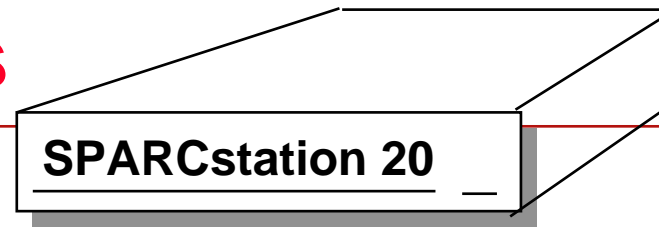
Standard I/O Devices



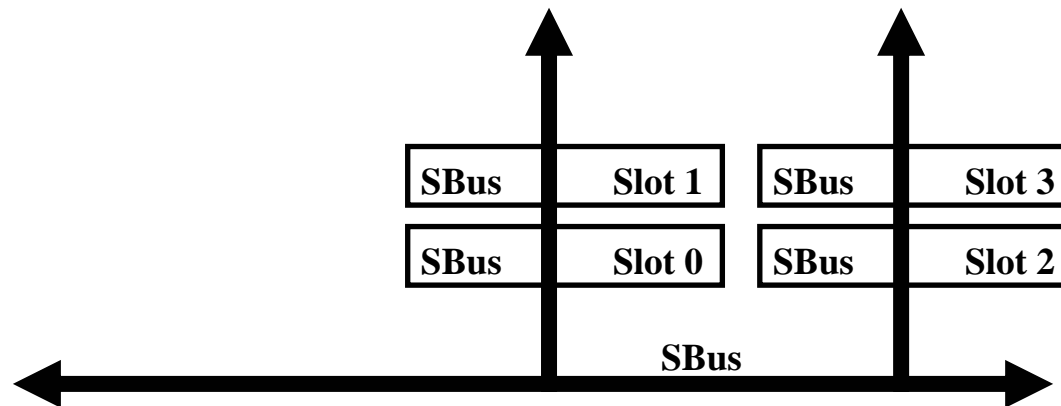
- **SCSI = Small Computer Systems Interface**
- **A standard interface (IBM, Apple, HP, Sun ... etc.)**
- **Computers and I/O devices communicate with each other**
- **The hard disk is one I/O device resides on the SCSI Bus**



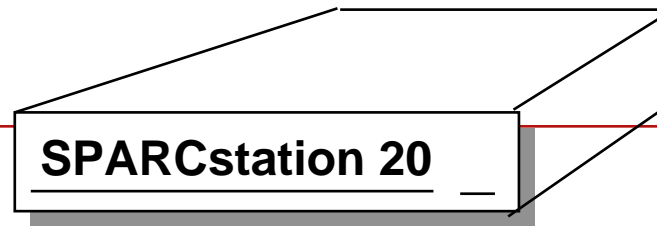
High Speed I/O Devices



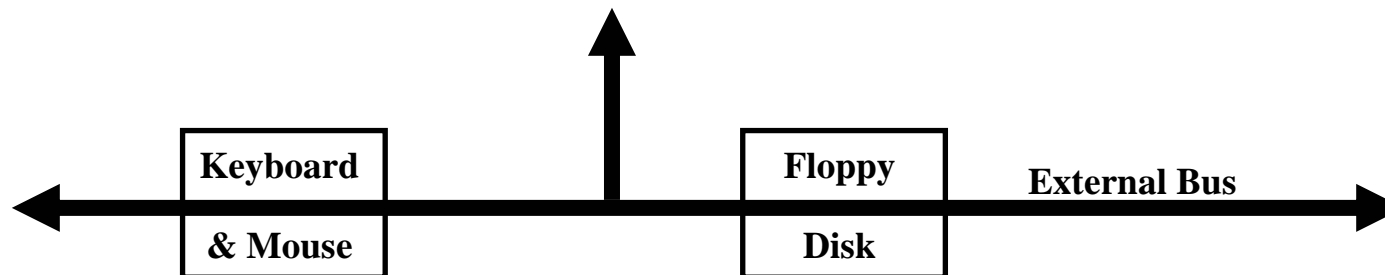
- **SBus is SUN's own high speed I/O bus**
- **SS20 has four SBus slots where we can plug in I/O devices**
- **Example: graphics accelerator, video adaptor, ... etc.**
- **High speed and low speed are relative terms**



Slow Speed I/O Devices



- There are only four SBus slots in SS20--"seats" are expensive
- The speed of some I/O devices is limited by human reaction time--very very slow by computer standard
- Examples: Keyboard and mouse
- No reason to use up one of the expensive SBus slot



本章总结

- 软件、硬件之间的接口：指令系统
 - **ISA**: 屏蔽硬件细节，简化程序员的使用
- 层次化的好处
 - 隐藏低层层次的实现细节，简化各层次上用户的使用
- 由五部分组成（有些书称为三部分），总线互连
 - **Processor: (1) datapath and (2) control**
 - **(3) Memory**
 - **I/O: (4) Input devices and (5) Output device**
- 存储系统按照容量、速度、价格分层
 - 寄存器：最贵，最快，容量最小，离**ALU**最近
 - **Cache**: 贵，快，容量小，离**CPU**近
 - **Main memory**: 便宜，慢，容量大，离**CPU**远
 - 磁盘：最便宜，最慢，容量最大，离**CPU**最远
- **I/O**的组织呈多样性
 - 速度范围大: **graphics vs. keyboard**
 - 考虑因素多: **speed, standard, cost ... etc.**
 - 实现方式多样: 机械、光电、磁性材料, **etc.**

第一章作业

- 作业题目
 - 第三版 1.1- 1.45, 1.56, 1.57
- 作业写在作业本上
 - 可以用英文答题，也可以用中文答题
- 下星期二上课时交
- 要求：
 - 作业书写一定要工整、清楚，答题要完整