第一章

计算机系统概论

Computer Abstractions

合肥工业大学

系统结构研究所

陈 田

大纲要求

- 计算机发展历程（第二章）
- 计算机系统的层次结构
  1. 计算机硬件的基本组成
  2. 计算机软件的分类
  3. 计算机的工作过程
- 计算机性能指标

  吞吐量；响应时间；CPU时钟周期；主频；CPI；CPU执行时间；MIPS；MFLOPS

第一章 概 论

1.1 计算机系统概述

1.2 计算机的基本组成

1.3 计算机性能评价

1.1 计算机系统概述

一、 计算机软、硬件的概念

1. 计算机系统

计算机系统
- 硬件：计算机中电子线路和物理装置
如主机、外设等
- 软件：计算机中全部程序和文件

软　件

- **System software(系统软件) - 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
  - 语言处理系统：翻译程序+ Linker, Debug, Loader, etc …
    - 翻译程序(Translator)有三类：

      汇编程序(Assembler)：汇编语言源程序→机器语言目标程序

      编译程序(Complier)：高级语言源程序→汇编/机器语言目标程序

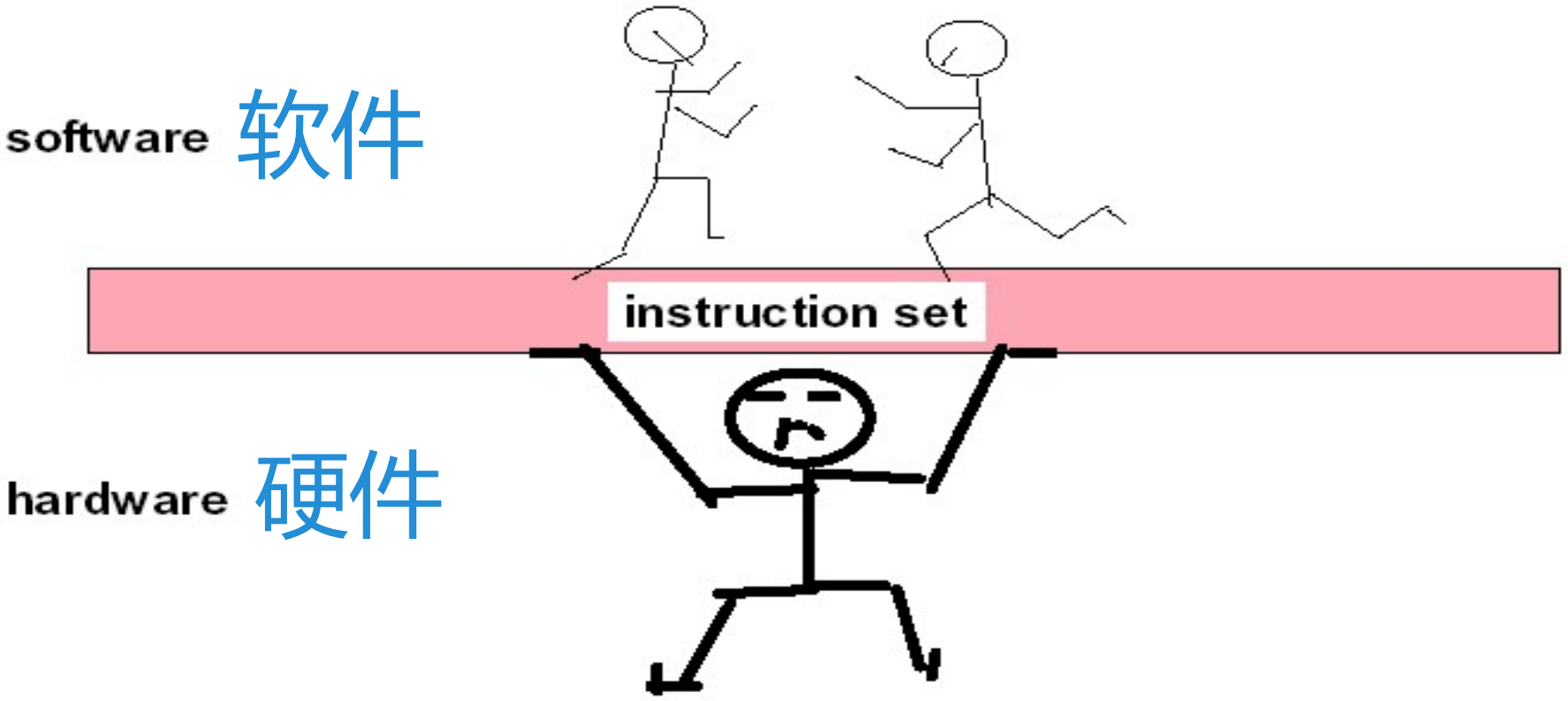
      解释程序(Interpreter)：将高级语言语句逐条翻译成机器指令并立即执行,不生成目标文件。
  - 其他实用程序: 如：磁盘碎片整理程序、备份程序等
- **Application software(应用软件) - 解决具体应用问题/完成具体应用任务**
  - 各类媒体处理程序：Word/ Image/ Graphics/…
  - 管理信息系统 (MIS)
  - Game，…

翻译方式

- 编译：使用编译程序把高级语言的源程序翻译成机器代码的目标程序，并以文件形式保留，然后执行。
  - C，PASCAL
  - 大多数编译程序直接产生机器语言的目标代码，形成可执行的目标文件，少数先产生汇编语言一级的符号代码文件，然后再调用汇编程序进行翻译加工处理，最后产生可执行的机器语言目标文件。

- 解释：使用解释程序，将源程序的一条语句翻译成对应机器语言的一条语句，并立即执行这条语句，接着再翻译源程序的下一条语句，并立即执行，并不形成目标程序，如此重复，直至完成源程序的全部翻译任务
  - JAVA
  - BASIC的翻译有解释和编译两种方式。

Hardware/Software Interface（界面）



软件和硬件的界面： ISA（Instruction Set Architecture）

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。

一、计算机软、硬件的概念

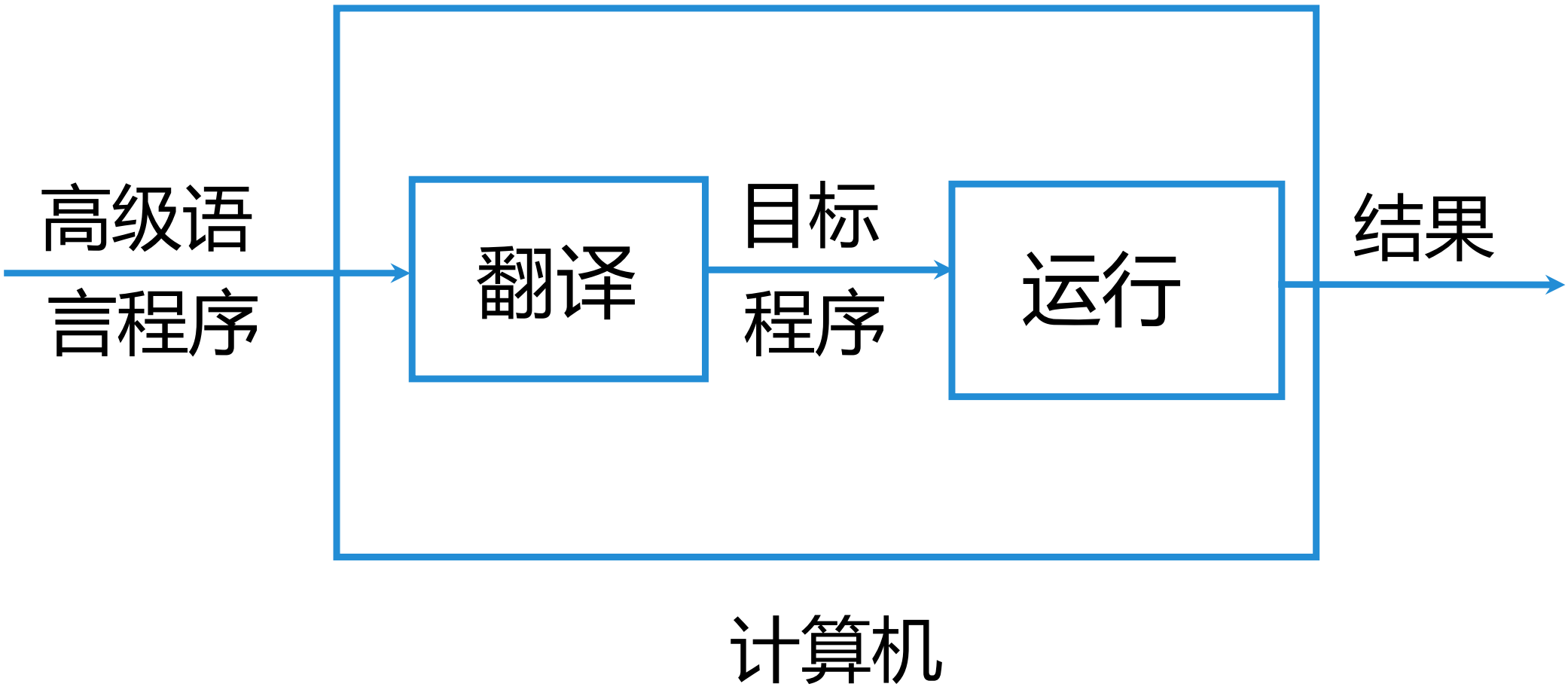
2. 计算机语言

1）机器语言：二进制代码表示的计算机语言，可以直接执行。

2）汇编语言：用助记符（如ADD、SUB、...）编写的语言，需要通过汇编程序翻译成目标程序后才执行。

3）高级语言：如PASCAL，C，JAVA等，便于编写、阅读，接近自然语言，提高了软件的产量，便于移植，独立于硬件环境。

计算机的解题过程

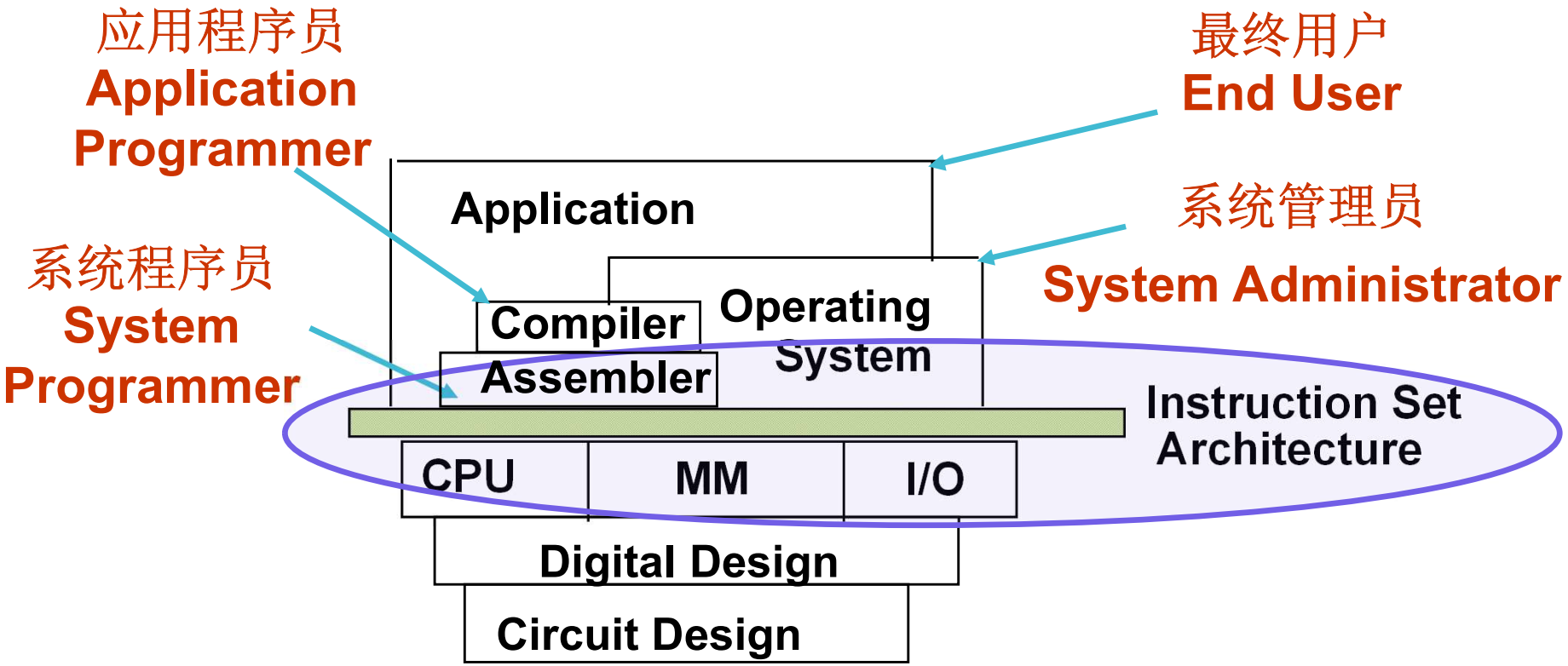


一、计算机软、硬件的概念

3. 硬件与软件的关系

- **相互关联：**
  - 硬件系统在最内层，是计算机系统的基础、核心
  - 系统软件为用户提供一个基本操作界面
  - 应用软件在最外层，为用户提供解决具体问题应用系统界面
- **相互转化 、互相渗透、互相融合**
  - 软硬件没有一条明确的分界线。
    - 硬件软化，可以增强系统的功能和适应性
    - 软件硬化，可以显著降低软件在时间上的开销
- **固件（Firmware）：具有软件特性的硬件**
    - 硬件的快速性
    - 软件的灵活性

What is Computer Architecture ?



- 上图给出的是计算机系统的层次结构
  指令系统（即ISA）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
- 中间阴影部分就是本课程主要内容，处于最核心的部分！

二、计算机系统的层次结构

- 实际机器：指由硬件或固件实现的机器。
- 虚拟机器：指以软件为主实现的机器。
  - 虚拟机器只对该级的观察者存在，某一级观察者只需通过该级的语言来了解和使用计算机，不必关心下级是如何工作和实现的。
    - 高级语言级的用户可以不了解机器的具体组成，不熟悉指令系统，直接用所指定的语言描述所要解决的问题。

二、计算机系统的层次结构

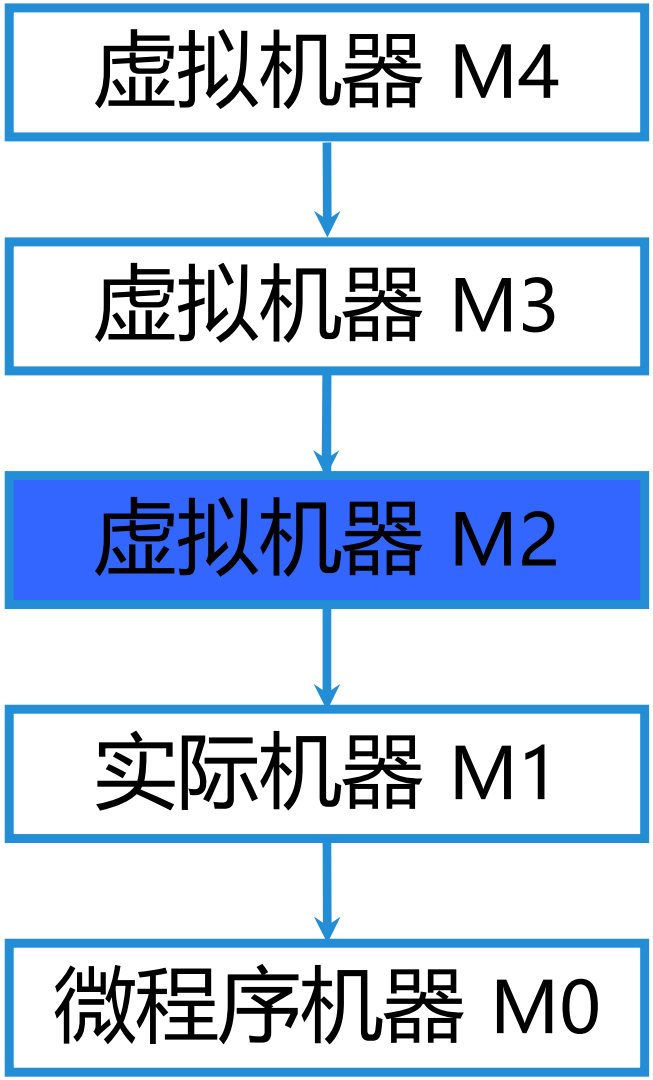
高级语言

汇编语言

操作系统

机器语言

微指令系统



二、计算机系统的层次结构

软件

虚拟机器 M4

用编译程序翻译成汇编语言程序

虚拟机器 M3

用汇编程序翻译成机器语言程序

虚拟机器 M2

用机器语言解释操作系统

硬件

实际机器 M1

用微指令解释机器指令

微程序机器 M0

由硬件直接执行微指令

High Level Language Program (e.g., C)

Compiler

Assembly Language Program (e.g., MIPS)

Assembler

Machine Language Program (MIPS)

Machine Interpretation

Hardware Architecture Description (e.g., block diagrams)

Architecture Implementation

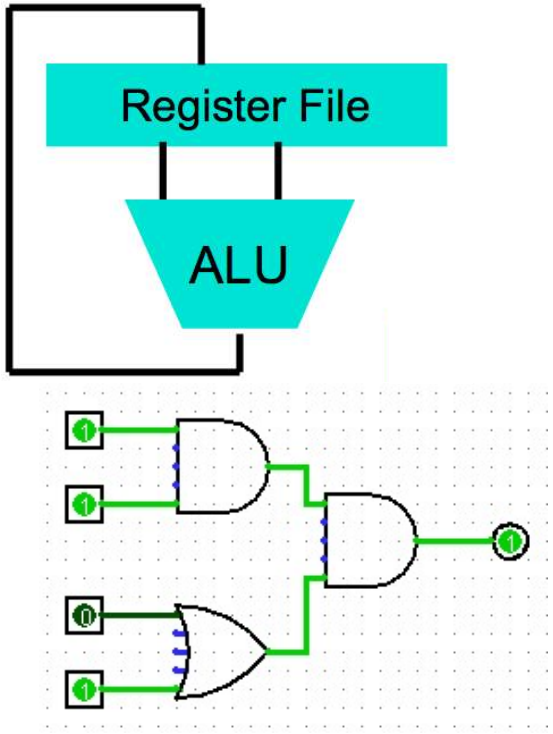
Logic Circuit Description (Circuit Schematic Diagrams)

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

```
lw   $t0, 0($2)
lw   $t1, 4($2)
sw   $t1, 0($2)
sw   $t0, 4($2)
```

Anything can be represented as a *number*, i.e., data or instructions

```
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
```



三、计算机体系结构和计算机组成

有无乘法指令

计算机体系结构
computer architecture

程序员所见到的计算机系统的属性
概念性的结构与功能特性

(指令系统、数据类型、寻址技术、I/O机理)

计算机组成
Computer organization

实现计算机体系结构所体现的属性

(具体指令的实现)

如何实现乘法指令

1.2 计算机的基本组成

一、冯·诺依曼计算机的特点

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. 采用存储程序方式：将指令序列描述的计算机程序和原始数据一起存储到计算机中，并可以按地址寻访。计算机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

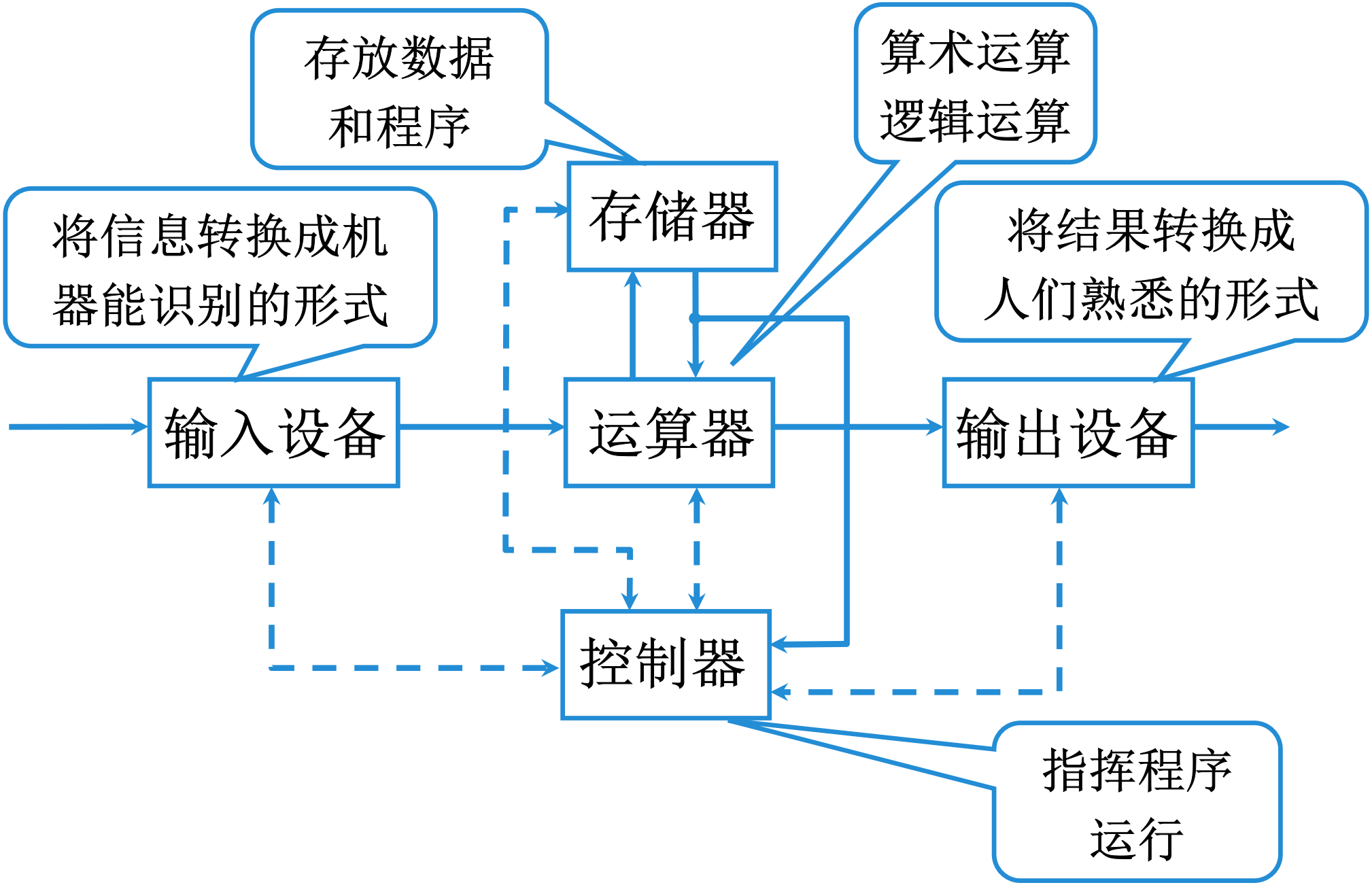
3. 指令和数据用二进制表示

4. 指令由操作码和地址码组成,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。

5. 指令在存储器中按顺序存放。一般按顺序执行，但可按运算结果或外界条件改变

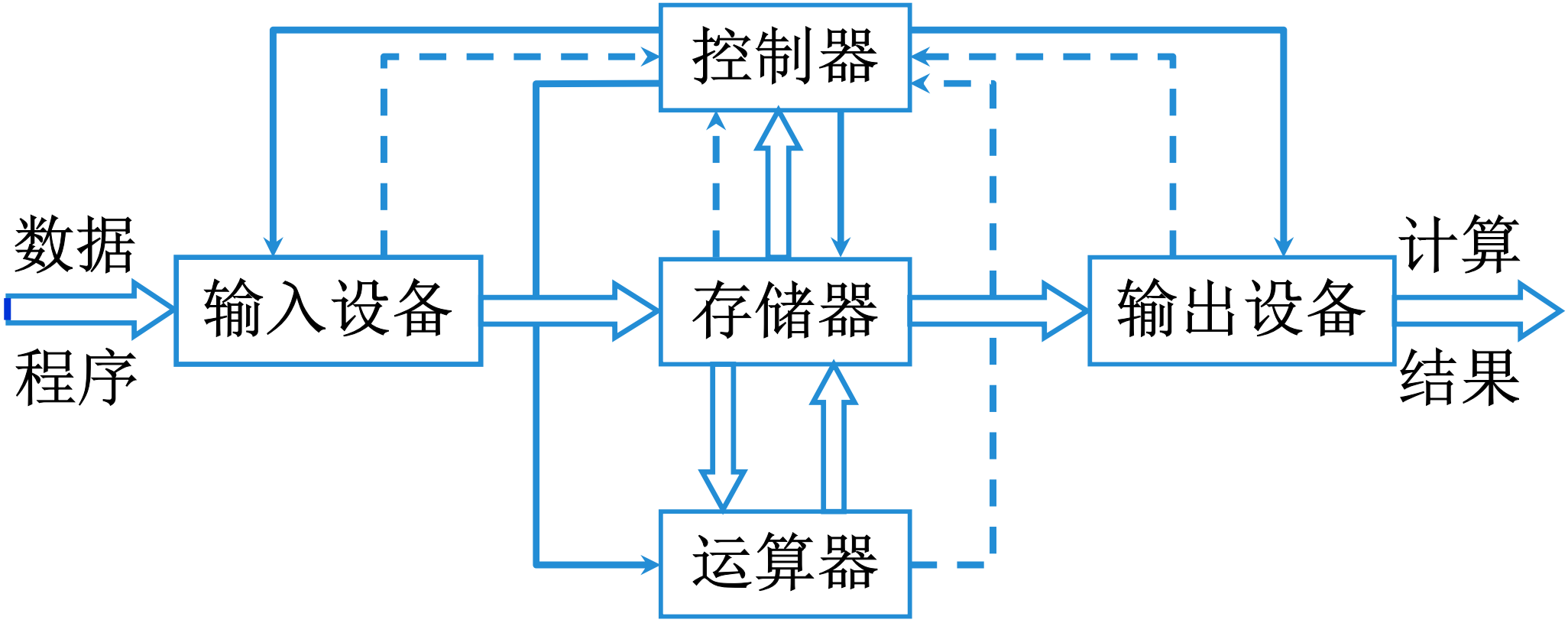
6. 以运算器为中心

冯·诺依曼计算机硬件框图



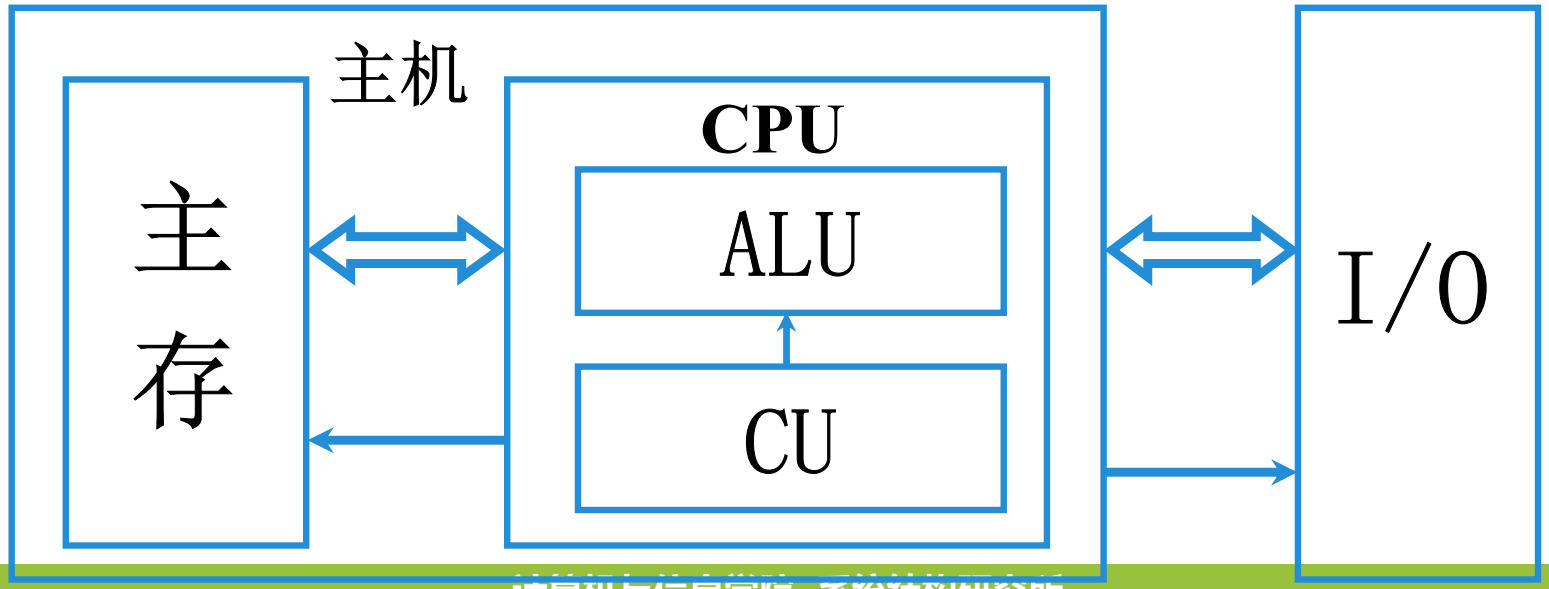
二、计算机硬件框图

1. 以存储器为中心的计算机硬件框图(现代)



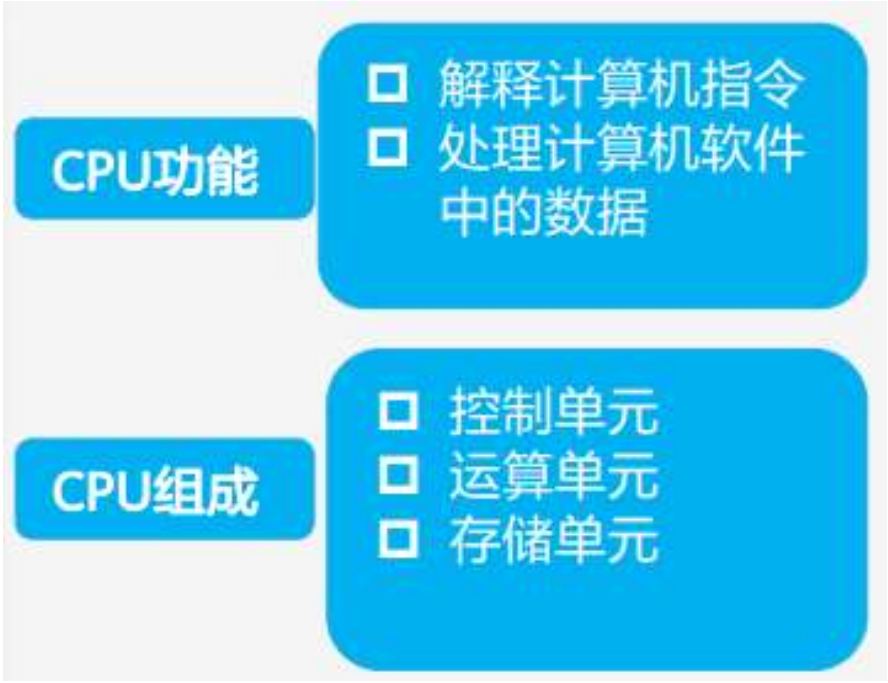
2. 现代计算机硬件框图

运算器 ALU
控制器 CU
} CPU
} 主机
存储器 { 主存 / 辅存
输入设备
输出设备
} I/O
} 硬件

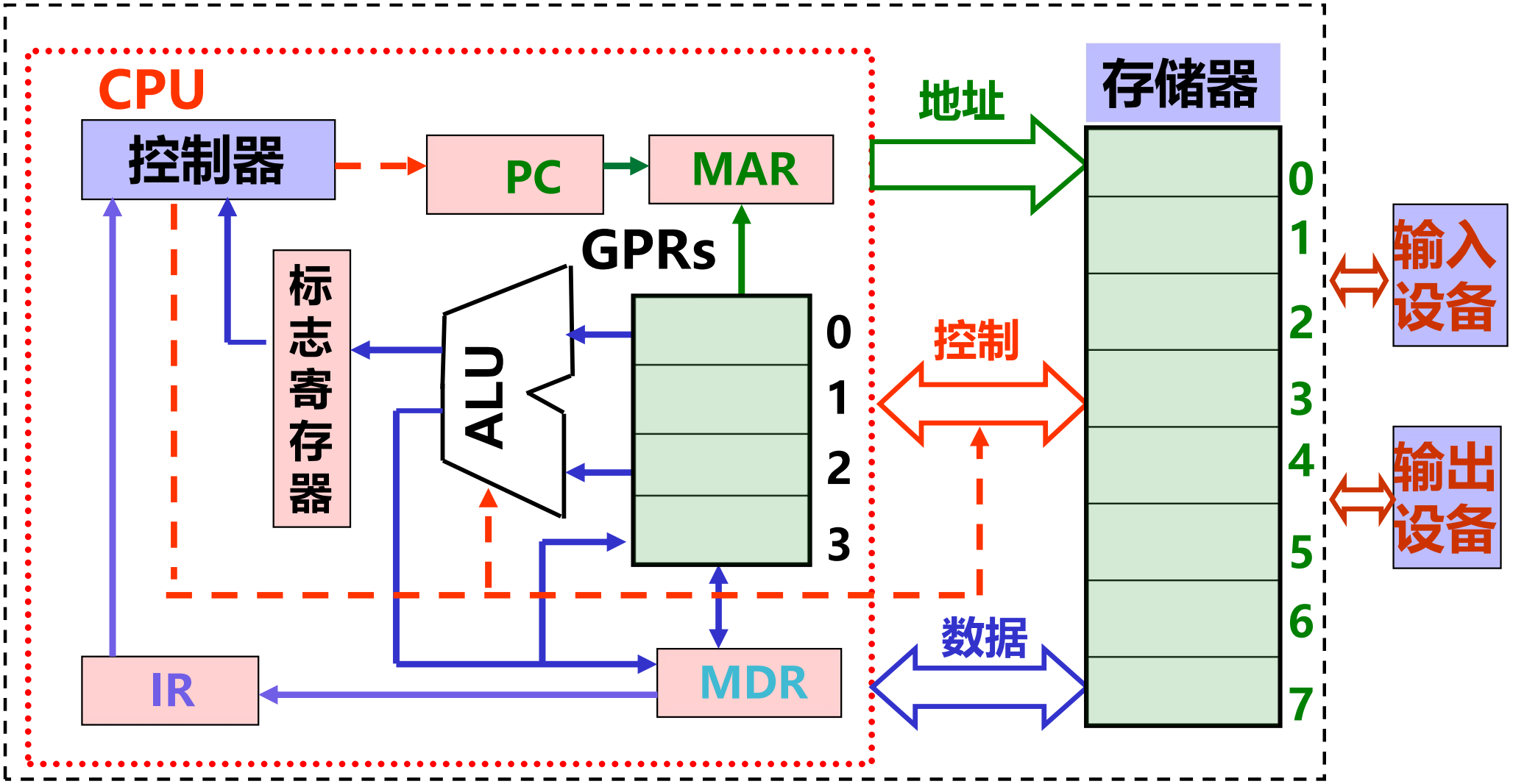


什么是中央处理器

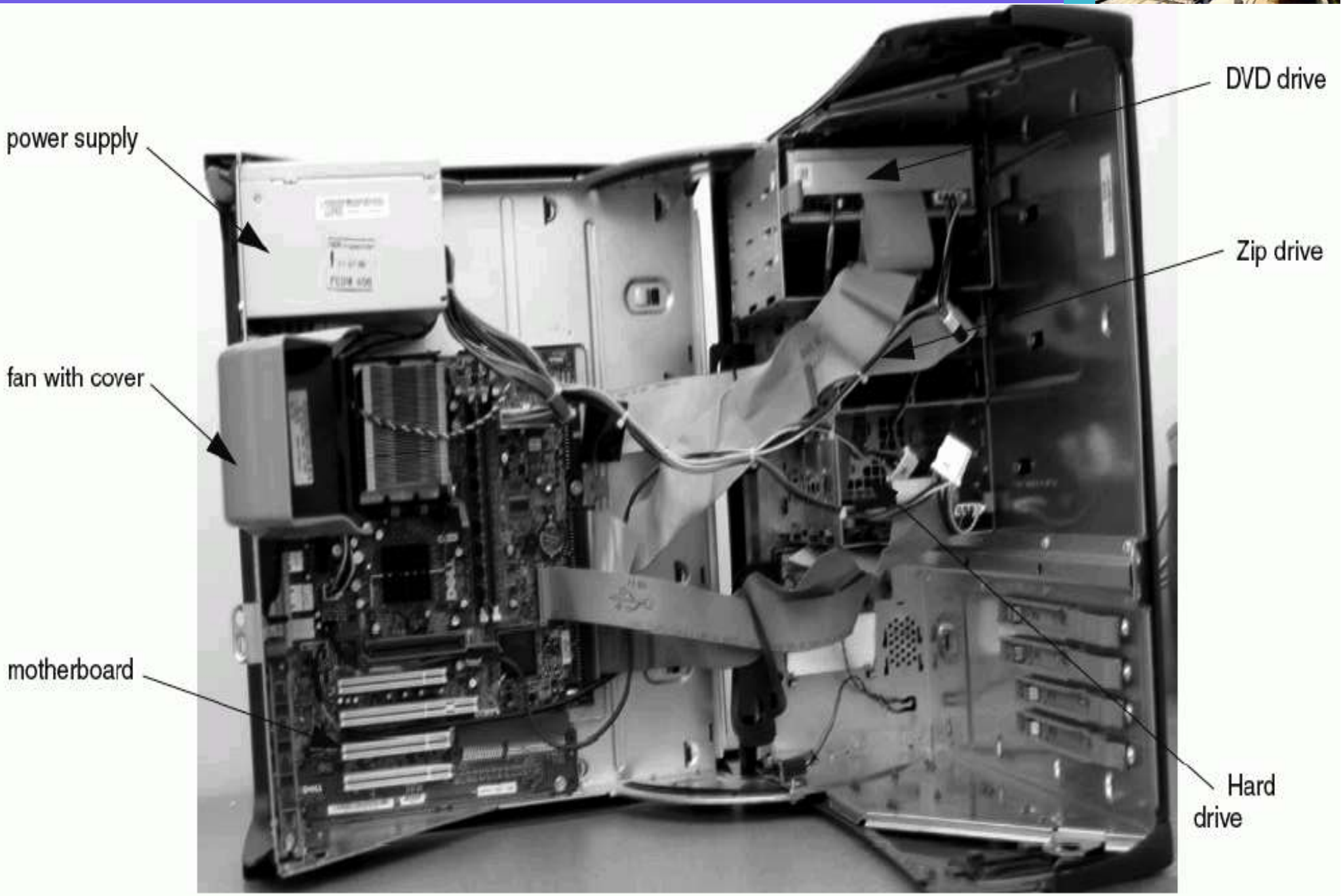
- 中央处理器 （Central Processing Unit，CPU）是计算机的主要设备之一，功能主要是解释计算机指令 以及处理计算机软件中的数据。
- 中央处理器主要包括控制器，运算器，高速缓冲存储器（Cache）及实现它们之间联系的数据、控制及 状态的总线。
- 它与主存（Memory）和输入/输出设备合称为电子计算机三大核心部件。



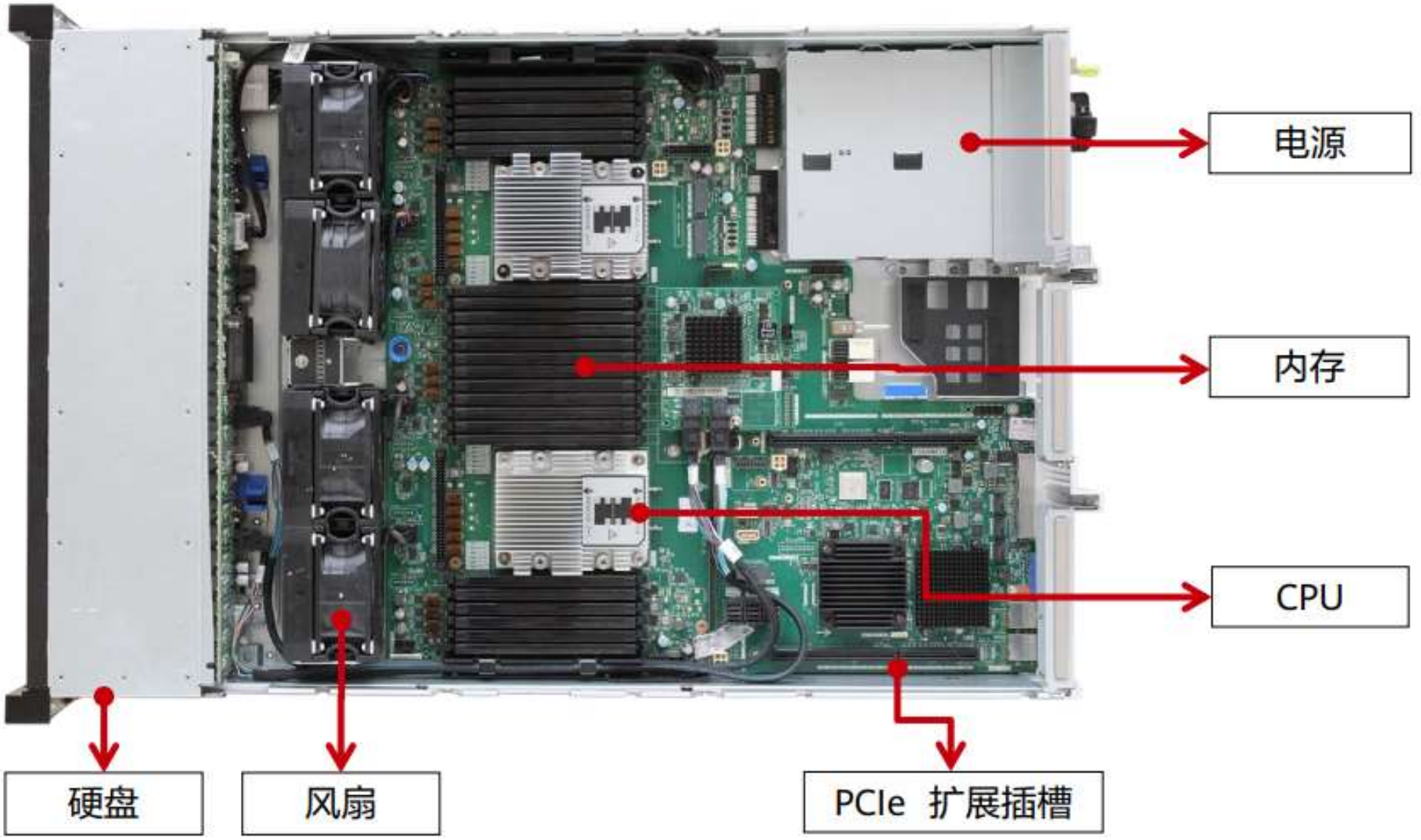
现代计算机结构模型


CPU
控制器
PC
MAR
GPRs
标志寄存器
ALU
0
1
2
3
IR
MDR
地址
存储器
控制
数据
0
1
2
3
4
5
6
7
输入设备
输出设备

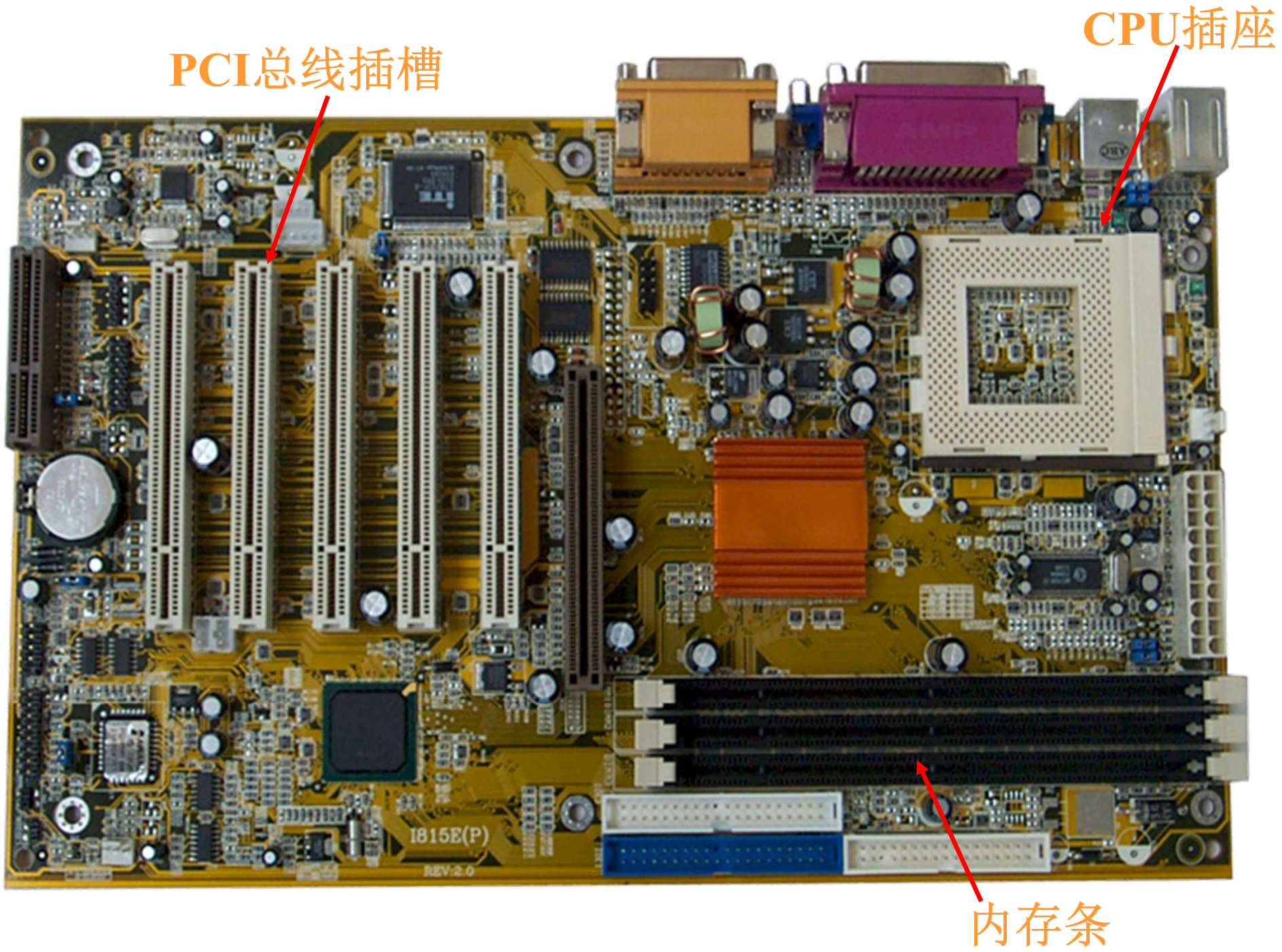
计算机硬件：打开计算机来看看



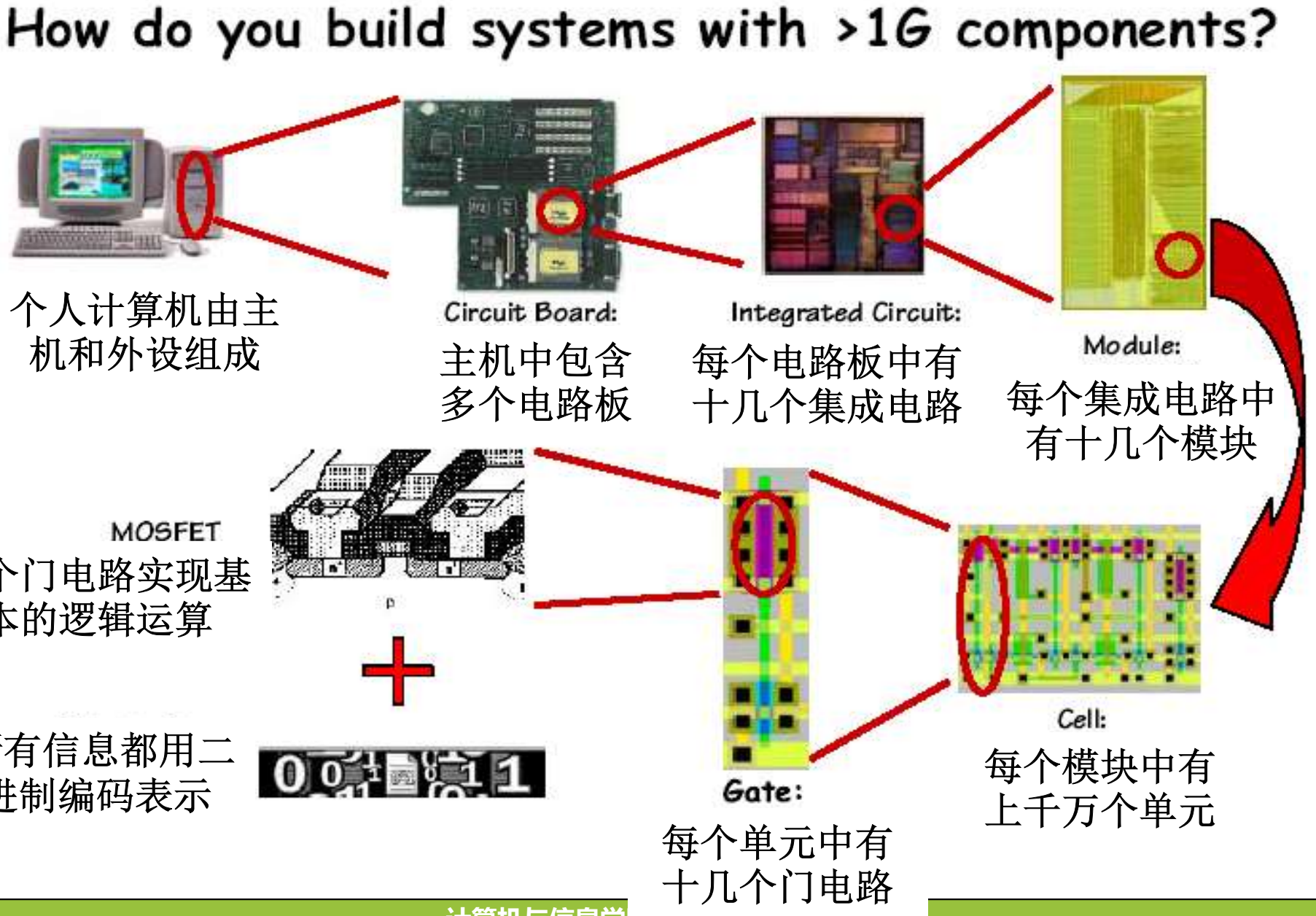
服务器内部视图



PC主板



解剖一台计算机


How do you build systems with >1G components?
个人计算机由主机和外设组成
Circuit Board:
主机中包含多个电路板
Integrated Circuit:
每个电路板中有十几个集成电路
Module:
每个集成电路中有十几个模块
MOSFET
每个门电路实现基本的逻辑运算
Cell:
每个模块中有上千万个单元
所有信息都用二进制编码表示
Gate:
每个单元中有十几个门电路

三、计算机的工作步骤

1. 上机前的准备

- 建立数学模型
- 确定计算方法

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \cdots$$

$$\sqrt{x} = \frac{1}{2}\left(y_n + \frac{x}{y_n}\right)\ (n = 0, 1, 2, \cdots)$$

- 编制解题程序

程序 — 运算的 全部步骤

指令 — 每 一个步骤

编程举例

计算 $ax^2+bx+c = (ax+b)x+c$

取x 至运算器中	取x 至运算器中
乘以x 在运算器中	乘以a 在运算器中
乘以a 在运算器中	加b 在运算器中
存ax^2 在存储器中	乘以x 在运算器中
取b 至运算器中	加c 在运算器中
乘以x 在运算器中	
加ax^2 在运算器中	
加c 在运算器中	

指令格式举例

操作码	地址码

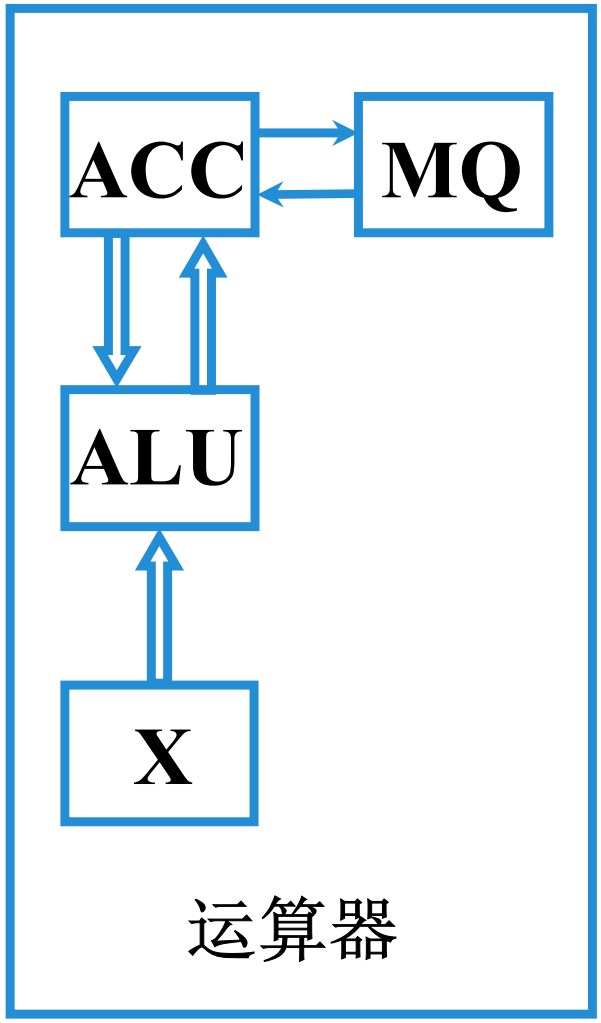
取数	α	[α] → ACC
000001	0000001000	
存数	β	[ACC] → β
加	γ	[ACC]+[γ] → ACC
乘	δ	[ACC]×[δ] → ACC
打印	σ	[σ] → 打印机
停机		

计算 $ax^2 + bx + c$ 程序清单

指令和数据存于主存单元的地址	指令		注释
	操作码	地址码	
0	000001	0000001000	取数x至ACC
1	000100	0000001001	乘a得ax存于ACC中
2	000011	0000001010	加b得ax+b,存于ACC中
3	000100	0000001000	乘x得（ax+b)x,存于ACC中
4	000011	0000001011	加c得$ax^2 + bx + c$,存于ACC
5	000010	0000001100	将$ax^2 + bx + c$存于主存单元
6	000101	0000001100	打印
7	000110		停机
8	x		原始数据x
9	a		原始数据a
10	b		原始数据b
11	c		原始数据c
12			存放结果

2. 计算机的解题过程

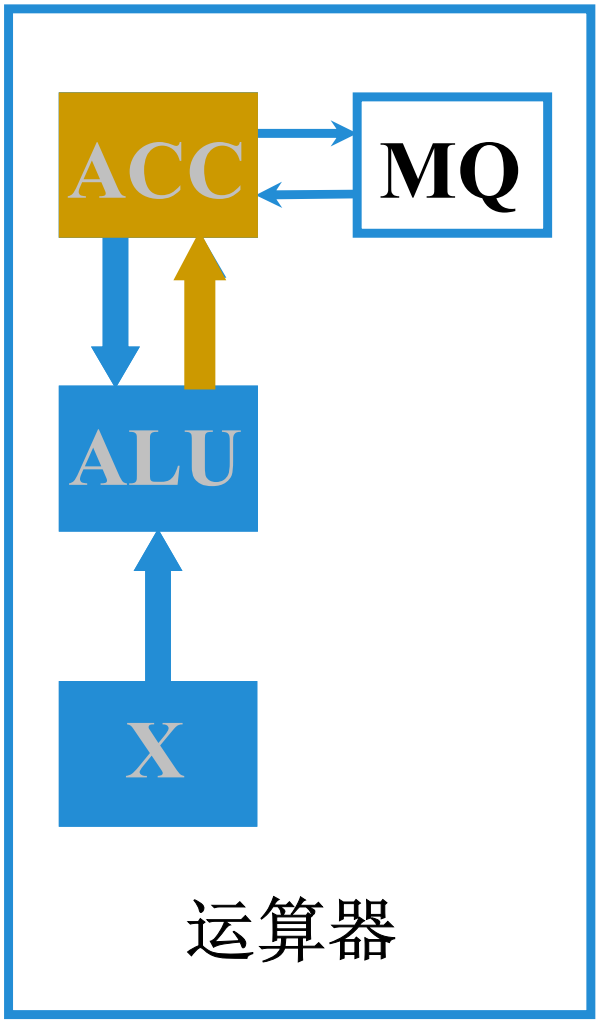
(1)运算器的基本组成及操作过程



	ACC	MQ	X
加法	被加数 和		加数
减法	被减数 差		减数
乘法	乘积高位	乘数 乘积低位	被乘数
除法	被除数 余数	商	除数

(1)运算器的基本组成及操作过程

① 加法操作过程



指令 | 加 | M

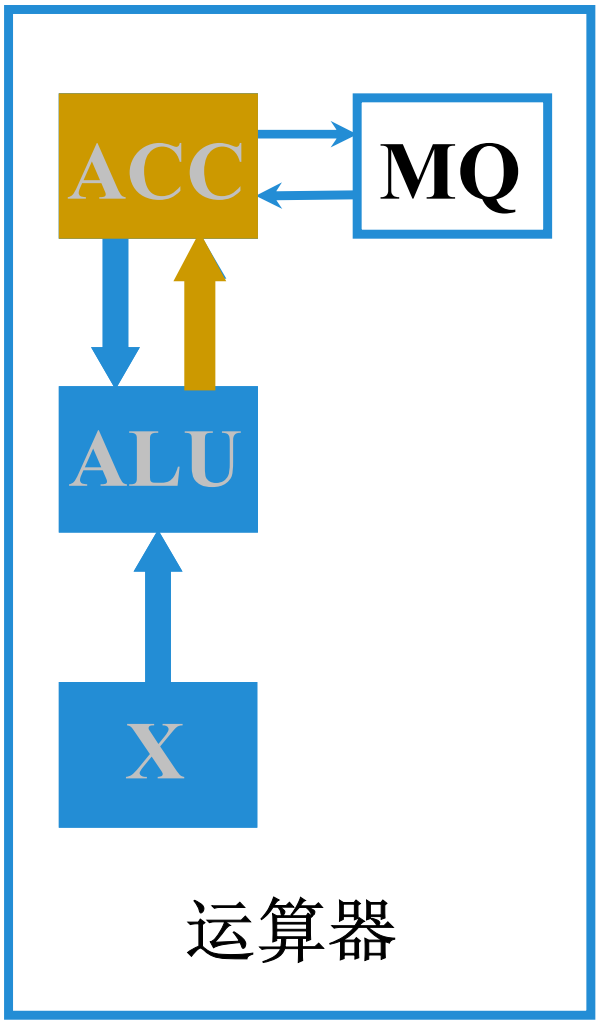
初态 ACC 被加数

[M] ⟶ X

[ACC]+[X] ⟶ ACC

(1)运算器的基本组成及操作过程

② 减法操作过程



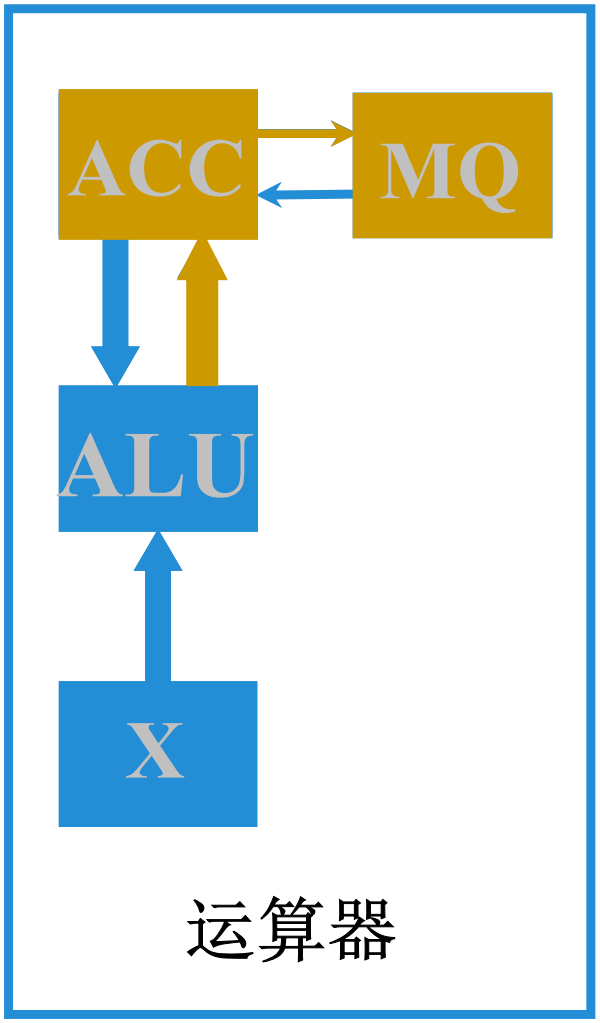
指令 | 减 | M

初态 ACC 被减数

[M] ⟶ X

[ACC]-[X] ⟶ ACC

③ 乘法操作过程



指令 | 乘 | M

初态 ACC 被乘数

[M] ⟶ MQ

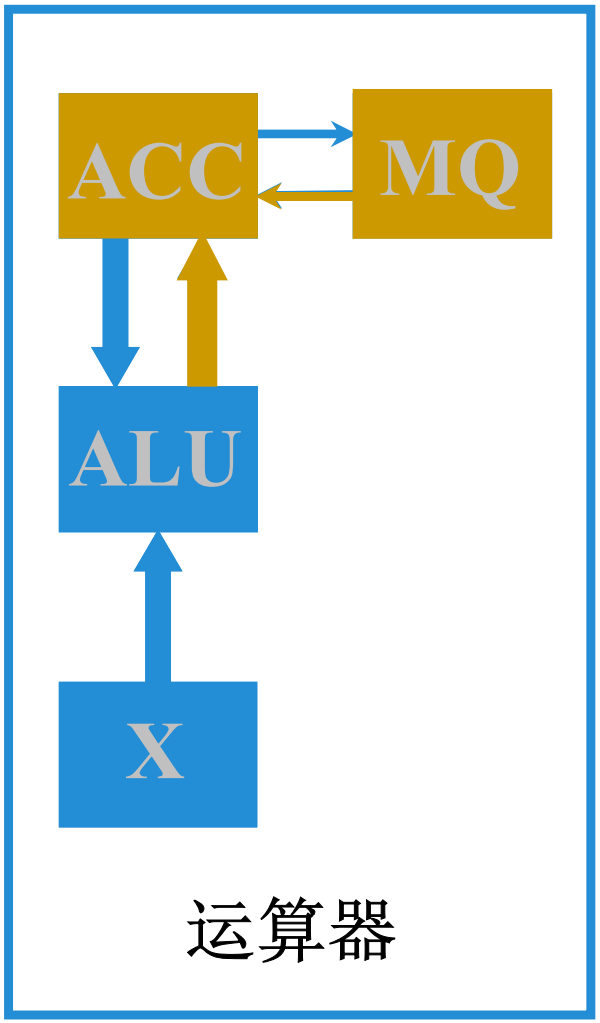
[ACC] ⟶ X

0 ⟶ ACC

[X]×[MQ] ⟶ ACC//MQ

(1)运算器的基本组成及操作过程

④ 除法操作过程



指令 | 除 | M

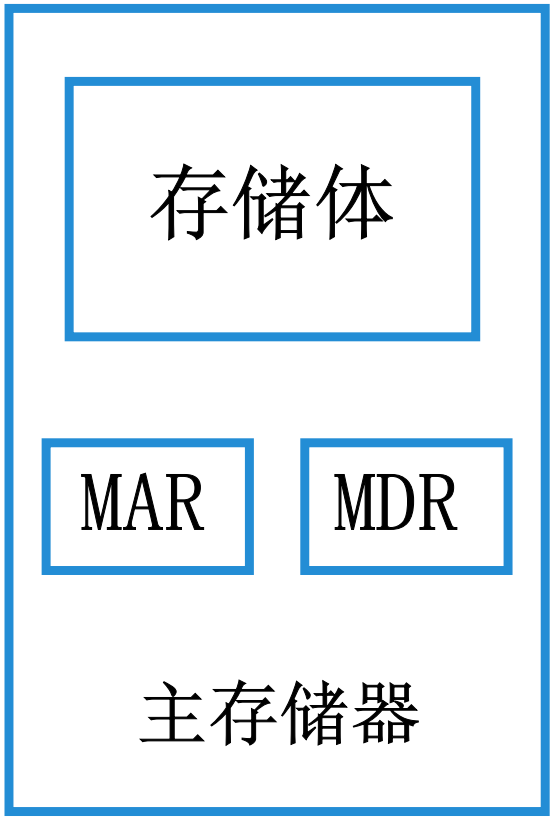
初态　ACC　被除数

[M] ⟶ X

[ACC] ÷ [X] ⟶ MQ

余数在ACC中

(2) 存储器的基本组成



存储体 – 存储单元 – 存储元件 （0/1）

[教学楼 – 教室 – 座位（无人/有人）]

存储单元　存放一串二进制代码

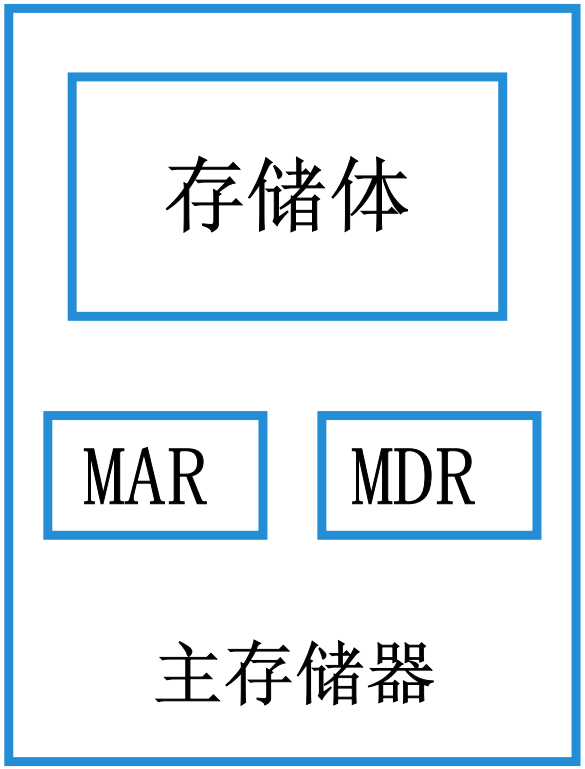
存储字　存储单元中二进制代码的组合

存储字长　存储单元中二进制代码的位数

每个存储单元赋予一个地址号

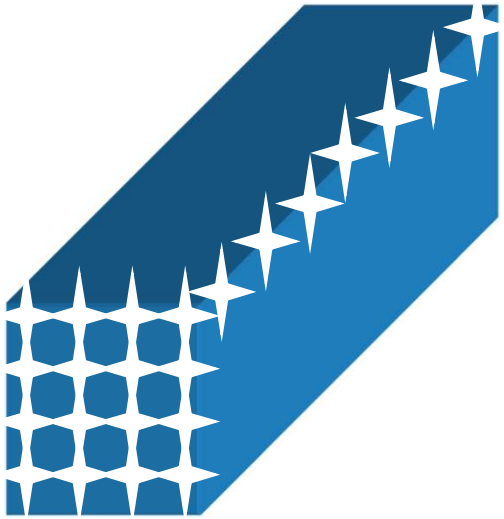
按地址寻访

(2) 存储器的基本组成



MAR　存储器地址寄存器
反映存储单元的个数

MDR　存储器数据寄存器
反映存储字长



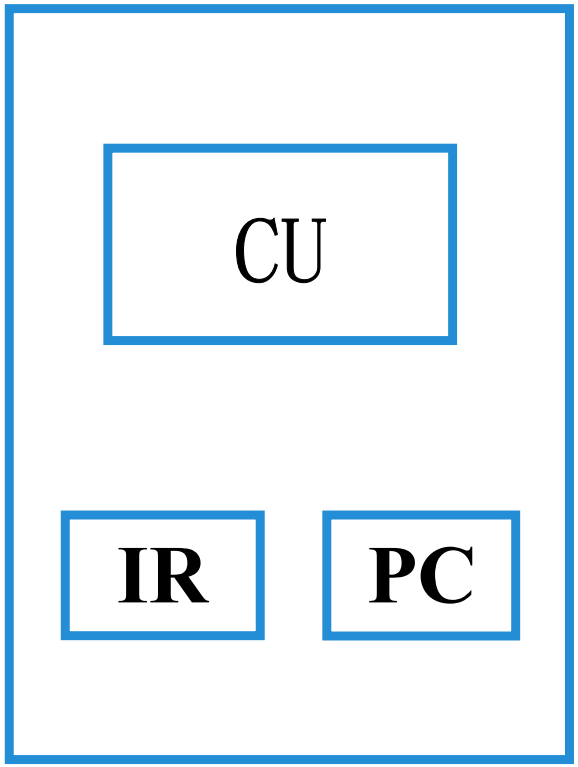
设　MAR = **4** 位

MDR = **8** 位

存储单元个数 **16**

存储字长 **8**

(3)控制器的基本组成



完成一条指令

- 取指令　PC
- 分析指令　IR

} 取指　访存

- 执行指令　CU　执行　访存

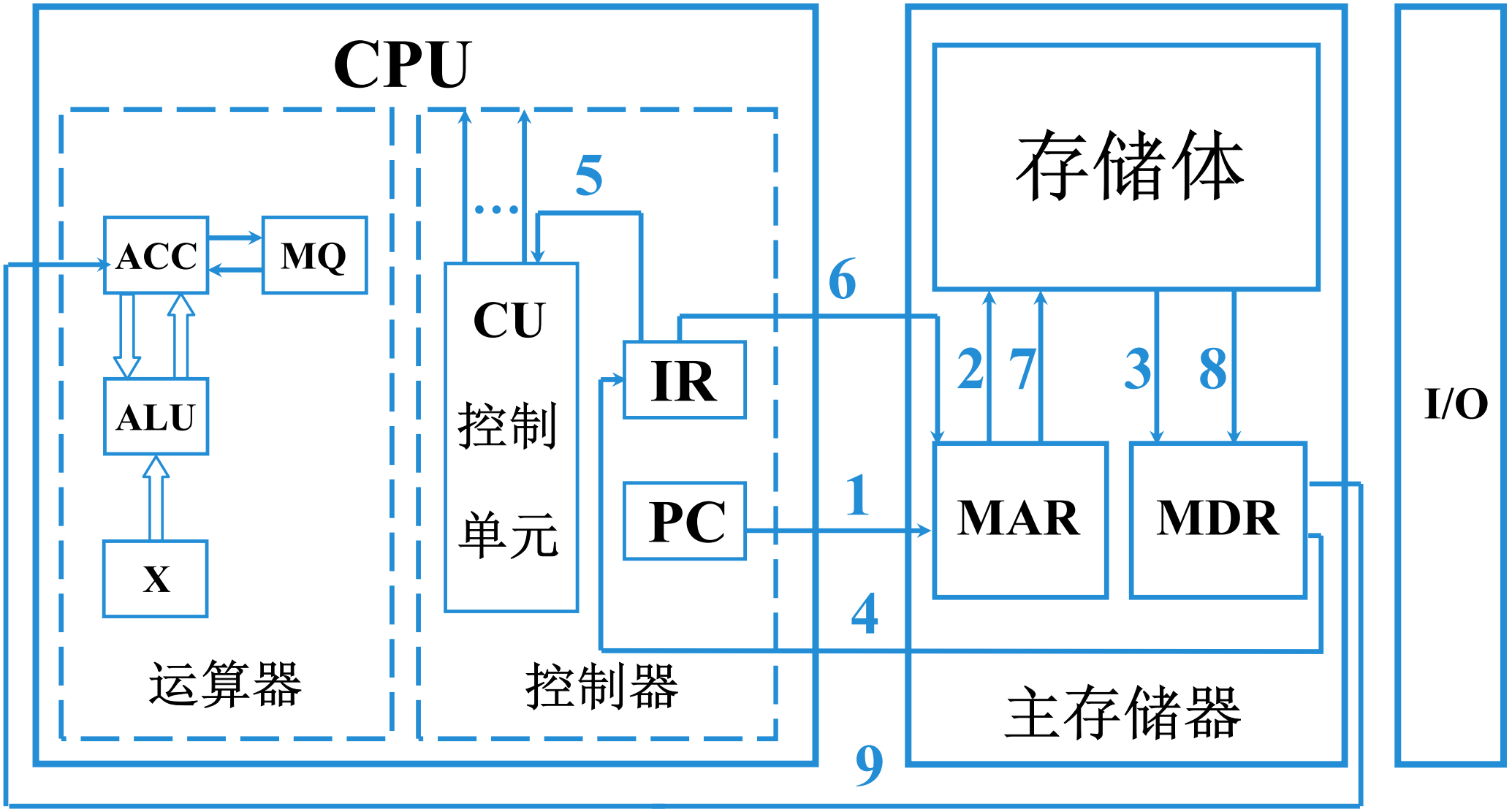
PC　存放当前欲执行指令的地址

具有计数功能（PC）+1 → PC

IR　存放当前欲执行的指令

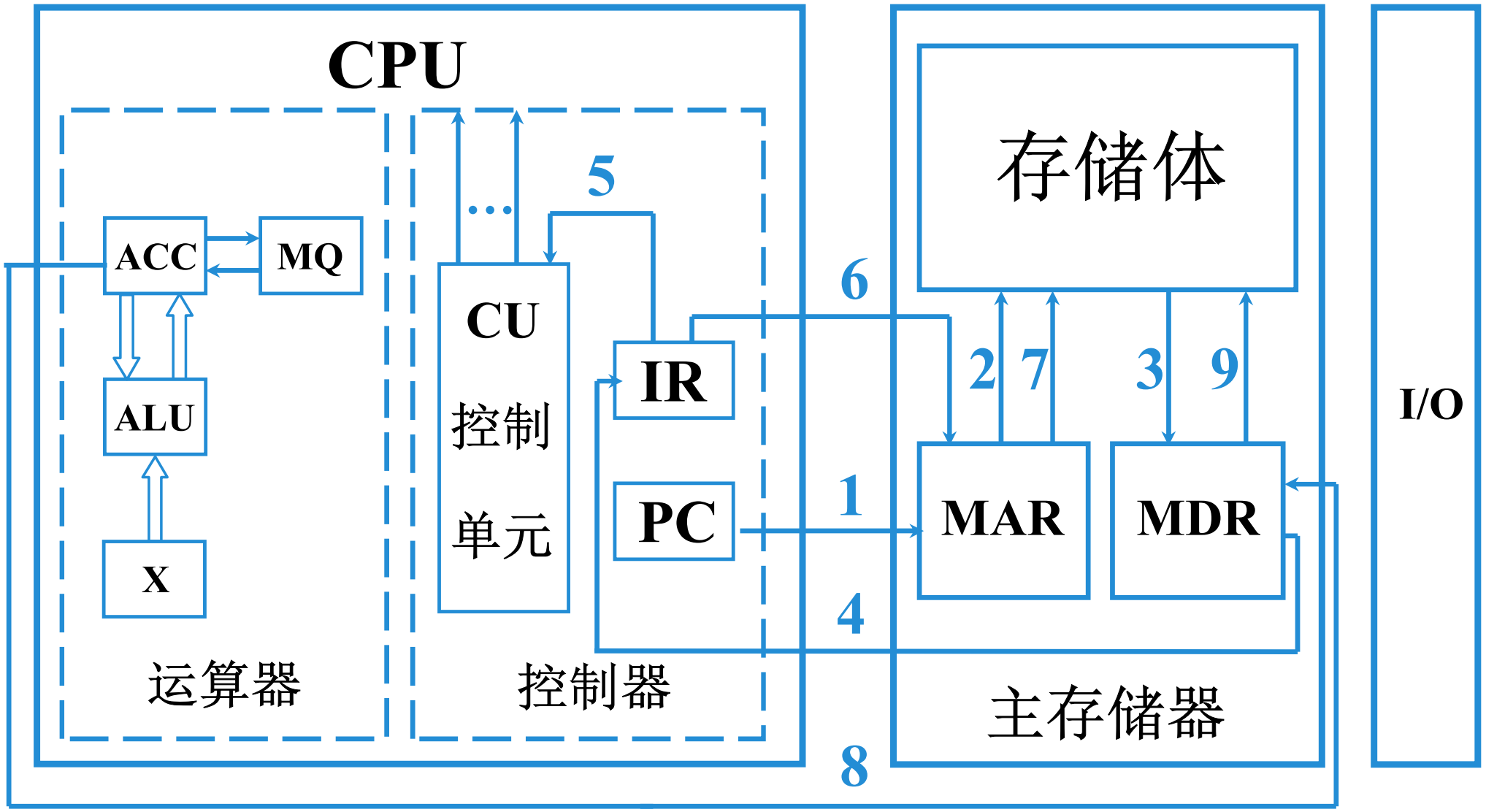
(4) 主机完成一条指令的过程

以取数指令为例



(4) 主机完成一条指令的过程

以存数指令为例



(5) $ax^2 + bx + c$ 程序的运行过程

- 将程序通过输入设备送至计算机
- 程序首地址 ⟶ PC
- 启动程序运行
- 取指令 PC → MAR → M → MDR → IR ，(PC)+1 → PC
- 分析指令 OP(IR) → CU
- 执行指令 Ad(IR) → MAR → M → MDR → ACC

  ⋮
- 打印结果
- 停机

1.3 计算机系统性能评价

- 制造成本（manufacturing cost）
- 衡量计算机性能的基本指标
  - 响应时间（response time）：从作业提交开始到作业完成所花的时间
    - 执行时间（execution Time）、等待时间（latency）
    - 一个程序的响应时间除了程序包含的指令在CPU上执行所花的时间，还包括磁盘访问时间、存储器访问时间、输入输出操作所需时间以及操作系统运行这个程序所花的额外开销
  - 吞吐量（throughput）：单位时间内所完成的工作量
    - 带宽（bandwidth）
- 计算机性能测量
- 指令执行速度（MIPS、MFLOPS）
- 基准程序（ Benchmark）

响应时间与吞吐量

- 假定用户不间断地输入请求，则在系统资源充裕的情况下，单个用户的吞吐量与响应时间成反比，即响应时间越短，吞吐量越大。
- 缩短响应时间通常意味着提高吞吐率，因此当处理器速度更快时，响应时间和吞吐率都得到了改善。
- 为了缩短某一用户或服务的响应时间，可以分配给它更多的资源。

响应时间与吞吐量

- 如果计算机中使用了多个处理器，不同处理器处理的任务不同，那么单个任务处理速度没有加快，只有吞吐率提高。
- 如果作业请求数量超过系统吞吐能力，则任务请求需要排队处理。此时由于采用多个处理器，排队时间减少了，响应时间也得到改善。

计算机性能的基本评价指标

- 计算机有两种不同的性能
  - Time to do the task
    - 响应时间（response time）
    - 执行时间（execution time）
    - 等待时间或时延（latency）
  - Tasks per day, hour, sec, ns. ..
    - 吞吐率（throughput）
    - 带宽（bandwidth）

不同应用场合用户关心的性能不同：

要求吞吐率高的场合，例如：

多媒体应用（音/视频播放要流畅）

要求响应时间短的场合：例如：

事务处理系统（存/取款速度要快）

要求吞吐率高且响应时间短的场合：

ATM、文件服务器、Web服务器等

- 基本的性能评价标准是：CPU的执行时间

" X is n times faster than Y" means

$$\frac{\text{ExTime(Y)}}{\text{ExTime(X)}} = \frac{\text{Performance(X)}}{\text{Performance(Y)}}$$

相对性能用执行时间的倒数来表示！

计算机性能的测量

- 比较计算机的性能时，用执行时间来衡量
  - 完成同样工作量所需时间最短的那台计算机就是性能最好的
  - 处理器时间往往被多个程序共享使用，因此，用户感觉到的程序执行时间并不是程序真正的执行时间
  - 通常把用户感觉到的响应时间分成两个时间：
    - CPU时间：指CPU真正花在程序执行上的时间。又包括两部分：
      - **用户CPU时间：用来运行用户代码的时间**
      - **系统CPU时间：为了执行用户程序而需要运行操作系统程序的时间**
    - 其他时间：指等待I/O操作完成或CPU花在其他用户程序的时间
  - 系统性能和CPU性能不等价，有一定的区别
    - 系统性能(System performance)：系统响应时间，与CPU外的其他部分也都有关系
    - CPU性能(CPU performance)：用户CPU时间
  - 本章主要讨论CPU性能，即：CPU真正用在用户程序执行上的时间

几个重要概念及技术指标

- 机器字长: CPU 一次能处理数据的位数与 CPU 中的 寄存器位数 有关
- 时钟周期(clock cycle,tick,clock tick,clock):又称为节拍或T周期，计算机执行指令的过程被分成若干步骤和相应的动作来完成，每一步动作都要有相应的控制信号进行控制，这些控制信号何时发出、作用时间多长，都要有相应的定时信号进行同步。CPU产生的同步时钟定时信号，也就是CPU主脉冲信号，宽度称为时钟周期
- 时钟频率（clock rate,主频)：CPU中的主脉冲信号的时钟频率，是CPU时钟周期的倒数。

几个重要概念及技术指标

- CPI（Cycle Per Instruction）：执行一条指令所需时钟周期数

- 吉普森法(Gibson)：根据不同类型指令在计算过程中出现的频繁程度，乘以不同系数，求得统计平均值，即平均运算速度。

$$T_M = \sum_{i=1}^{n} f_i t_i$$

T_M：机器运行速度；

f_i：第i种指令占全部操作的百分比；

t_i：第i种指令的执行时间

几个重要概念及技术指标

- **CPU执行时间的计算：**

CPU 执行时间 = CPU时钟周期数 / 程序 X 时钟周期

= CPU时钟周期数 / 程序 ÷ 时钟频率

= 指令条数 / 程序 X CPI X 时钟周期

CPU时钟周期数 / 程序 = 指令条数 / 程序 X CPI

CPI = CPU时钟周期数 / 程序 ÷指令条数 / 程序

CPI 用来衡量以下各方面的综合结果

- Instruction Set Architecture（ISA）
- Implementation of that architecture
- Program（Compiler、Algorithm）

如何计算CPI？

对于某一条特定的指令而言，其CPI是一个确定的值。但是，对于某一类指令、或一个程序、或一台机器而言，其CPI是一个平均值，表示该类指令或该程序或该机器的指令集中每条指令执行时平均需要多少时钟周期。

假定CPI $_i$ 和 C $_i$分别为第 i类指令的CPI和指令条数，则程序的总时钟数为：

$$\text{总时钟数} = \sum_{i=1}^{n} CPI_i \times C_i$$

所以，

$$\text{CPU时间} = \text{时钟周期} \times \sum_{i=1}^{n} CPI_i \times C_i$$

假定CPI$_i$、 F$_i$是各指令CPI和在程序中的出现频率，则程序综合CPI为:

$$\text{CPI} = \sum_{i=1}^{n} CPI_i \times F_i \qquad \text{where } F_i = \frac{C_i}{Instruction\_Count}$$

已知CPU时间、时钟频率、总时钟数、指令条数，则程序综合CPI为:

$$\text{CPI} = (\text{CPU 时间} \times \text{时钟频率}) / \text{指令条数} = \text{总时钟周期数} / \text{指令条数}$$

问题：指令的CPI、机器的CPI、程序的CPI各能反映哪方面的性能?

单靠CPI不能反映CPU的性能！为什么？如：单周期处理器CPI=1，但性能差！

Example1

程序P在机器A上运行需10 s， 机器A的时钟频率为400MHz。现在要设计一台机器B，希望该程序在B上运行只需6 s.

机器B时钟频率的提高导致了其CPI的增加，使得程序P在机器B上时钟周期数是在机器A上的1.2倍。机器B的时钟频率达到A的多少倍才能使程序P在B上执行速度是A上的10/6=1.67倍?

Answer:

CPU时间A = 时钟周期数A / 时钟频率A

时钟周期数A = 10 sec x 400MHz = 4000M个

时钟频率B = 时钟周期数B / CPU时间B

= 1.2 x 4000M / 6 sec = 800 MHz

机器B的频率是A的两倍，但机器B的速度并不是A的两倍!

Marketing Metrics （产品宣称指标）

MIPS = Instruction Count / Time x10^6

= Clock Rate / CPI x 10^6

Million **I**nstructions **P**er **S**econd

平均每秒执行百万条指令数

因为每条指令执行时间不同，所以**MIPS**总是一个平均值。

- 不同机器的指令集不同
- 程序由不同的指令混合而成
- 指令使用的频度动态变化
- **Peak MIPS:** （不实用）

用**MIPS**数表示性能有没有局限?

所以**MIPS**数不能说明性能的好坏（用下页中的例子来说明）

MFLOPS = FP Operations / Time x10^6

Million **Flo**ating-point Operations **P**er **S**econd

- 与机器相关性大
- 并不是程序中花时间的部分

用**MFLOPS**数表示性能也有局限！

Example: MIPS数不可靠！

Assume we build an optimizing compiler for the load/store machine. The compiler discards 50% of the ALU instructions.

1) What is the CPI ?

仅仅在软件上进行优化，没有涉及到任何硬件措施。

2) Assuming a 20 ns clock cycle time (50 MHz clock rate). What is the MIPS rating for optimized code versus unoptimized code? Does the MIPS rating agree with the rating of execution time?

Op	Freq	Cycle
ALU	43%	1
Load	21%	2
Store	12%	2
Branch	24%	2

Optimizing compiler

21.5/ (21.5+21+12+24)=27%

21 / (21.5+21+12+24)=27%

12 / (21.5+21+12+24)=15%

24 / (21.5+21+12+24)= 31%

New	Freq
	27%
	27%
	15%
	31%

CPI	1.57	50M/1.57=31.8MIPS	1.73
MIPS	31.8	50M/1.73=28.9MIPS	28.9

结果：因为优化后减少了ALU指令（其他指令数没变），所以程序执行时间一定减少了，但优化后的MIPS数反而降低了。

浮点运算速度

- FLOPS:
  - 1 Megaflop/s指每秒执行100万次浮点运算
  - 1 Gigaflop/s指每秒执行10亿次浮点运算
  - 1 Teraflop/s指每秒执行1万亿次浮点运算
  - 1 Petaflop/s指每秒执行1000万亿次浮点运算
  - 1 Exaflop/s即每秒执行100万万亿次浮点运算

K、M、G、T、P的定义

单 位	通常意义	实际表示
K(Kilo)	10^3	$2^{10}=1\ 024$
M(Mega)	10^6	$2^{20}=1\ 048\ 576$
G(Giga)	10^9	$2^{30}=1\ 073\ 741\ 824$
T(Tera)	10^{12}	$2^{40}=1\ 099\ 511\ 627\ 776$
P(Peta)	10^{15}	$2^{50}=1\ 125\ 899\ 906\ 842\ 624$

时间、存储容量（或带宽）的单位

Notations and Conventions for Numbers

Prefix	Abbreviation	Meaning	Numeric Value
mill	m	One thousandth	10^{-3}
micro	μ	One millionth	10^{-6}
nano	n	One billionth	10^{-9}
pico	p	One trillionth	10^{-12}
femto	f	One quadrillionth	10^{-15}
atta	a	One quintillionth	10^{-18}
kilo	K (or k)	Thousand	10^{3} *or* 2^{10}
mega	M	Million	10^{6} *or* 2^{20}
giga	G	Billion	10^{9} *or* 2^{30}
tera	T	Trillion	10^{12} *or* 2^{40}
peta	P	Quadrillion	10^{15} *or* 2^{50}
exa	E	Quintillion	10^{18} *or* 2^{60}

概念及技术指标

- 存储容量：　存放二进制信息的总数量

主存容量

存储单元个数 × 存储字长

如：MAR　MDR　容量

10　8　1K×8位

16　32　64K×32位

字节数

如：2^{13} = 1KB

2^{21} = 256KB

辅存容量　字节数　80GB

$1K = 2^{10}$

$1Byte = 2^{3}$

$1G = 2^{30}$

概念及技术指标

- 系列机：具有基本相同的体系结构，使用相同的基本指令系统的多个不同型号的计算机组成的一个产品系列。
  - 系列机的出现是计算机发展过程中的重要事件，对计算机推广应用起到重要作用。

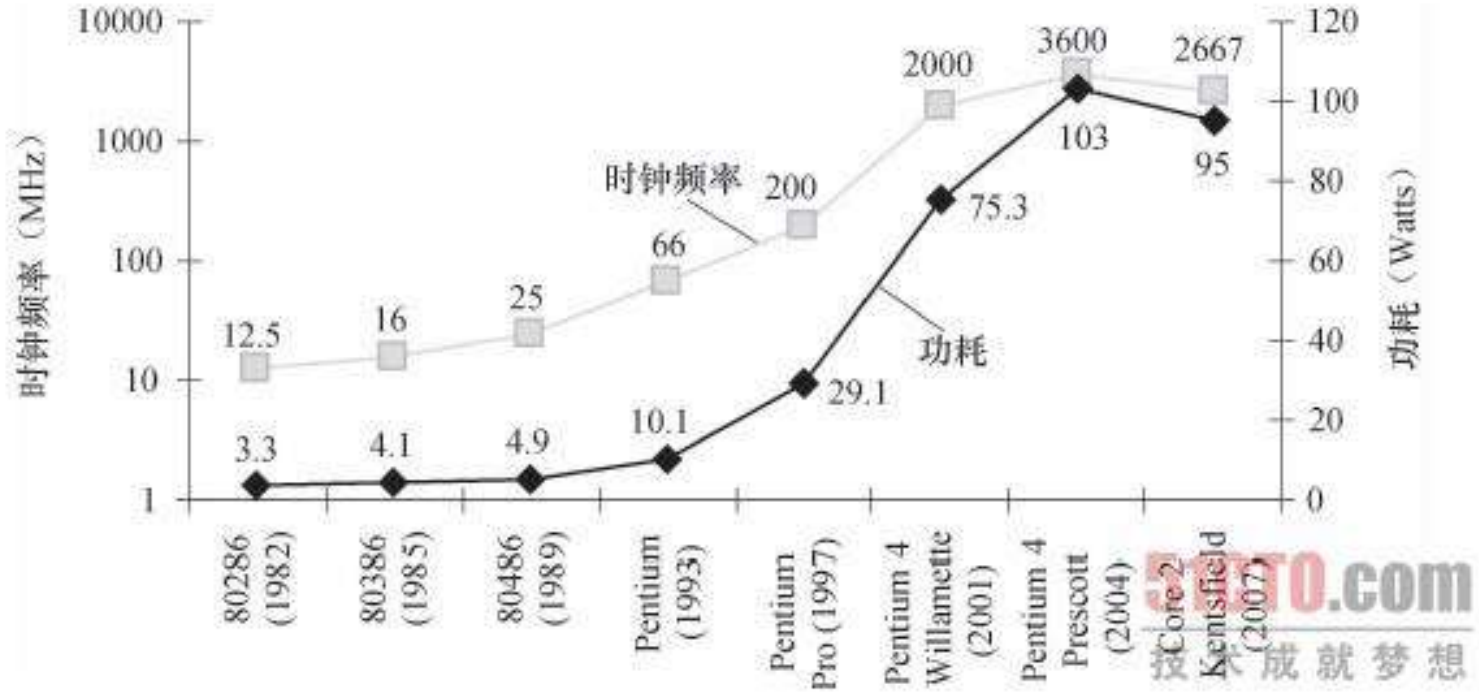
概念及技术指标

- 兼容性：指计算机软件或硬件的通用性，即使用或运行在某个型号的计算机系统中的硬件、软件也能应用于另外一个型号的计算机系统时，则称这两台计算机在硬件或软件上存在兼容性。
  - 通常，在同一个系列的不同型号的计算机之间存在兼容性
  - 当某个厂家的成功计算机系统或某些部件被其它厂家在保持相同的功能特性的前提下，通过合法渠道重新设计、组织生产并投入市场营销，这些产品称为兼容计算机或兼容部件产品。

概念及技术指标

- **软件可移植性：指把使用在某个系列计算机中的软件直接或很少修改就能运行在另外一个系列计算机的可能性。**
  - 希望同一系列的计算机中要有软件兼容性，至少期望后来出现的性能更高的计算机要能运行早期计算机系统中已有的软件。
  - 也多指不同系列的计算机中的程序是否可以直接移植或者实现移植的难易程度，通常只有高级语言的源程序才有移植的可能性，汇编语言及机器语言基本没有移植可能性。

功耗墙



25年间Intel x86八代微处理器的时钟频率和功耗增长趋势

- 两者的增长几乎保持了将近20年，但近几年来突然缓和下来
- 两者是密切相关的，而且功耗已经到达了极限，无法再将处理器冷却下来
- 奔腾4处理器时钟频率和功耗提高很大，但是性能提升不大。
- Prescott发热问题导致奔腾4处理器（生产线）的放弃。
- Core 2（生产线）恢复使用低时钟频率的简单流水线和片上多处理器。

主要性能指标

- 可靠性
- 能耗
- 可用性
- 可维护性
- 安全性

Computer Organization

Thank You !

Institute of Computer Architecture