《计算机组成原理》总复习

Eslzzyl

2023 年 3 月 12 日

目录

一 计算机系统概述

1.1 计算机发展历程

本节已从新大纲中删除，故略。

1.2 计算机系统层次结构

1.2.1 计算机系统的组成

计算机系统 = 硬件系统 + 软件系统

1.2.2 计算机硬件

1. 冯 · 诺伊曼机基本思想

   冯 · 诺依曼机有如下特点：

   - 采用“存储程序”的工作方式，即：将事先编制好的程序和原始数据送入主存后才能执行，一旦程序被启动执行，就无需操作人员的干预，计算机会自动逐条执行指令，直至程序执行结束。
   - 计算机硬件系统由 5 大部件组成：
     - ◇ 运算器
     - ◇ 存储器
     - ◇ 控制器
     - ◇ 输入设备
     - ◇ 输出设备
   - 指令和数据以同等地位存储在存储器中，形式上没有区别，但计算机应当能够区分它们。
   - 指令和数据均用二进制代码表示。指令由操作码和地址码组成。

2. 计算机的功能部件

   (a) 输入设备。如键盘、鼠标、扫描仪、摄像机等。

   (b) 输出设备。如显示器、打印机等。

(c) 存储器

又分主存（内存）和辅存（外存）。CPU 可以直接访问主存，但不能直接访问辅存。

主存的工作方式是按存储单元的地址进行存取，即**按地址存取方式**。

MAR（存储器地址寄存器）、MDR（存储器数据寄存器）二者虽然属于存储器的一部分，但实际上是集成在 CPU 中的。

(d) 运算器

核心是 ALU。此外还包含若干通用寄存器，如：

- 累加器 ACC
- 乘商寄存器 MQ
- 操作数寄存器 X
- 变址寄存器 IX
- 基址寄存器 BR
  前三个是必须的。

运算器还包含程序状态寄存器（PSW），又称标志寄存器。

(e) 控制器

包含如下组件：

- 程序计数器 PC
- 指令寄存器 IR
- 控制单元 CU

1.2.3 计算机软件

1. 系统软件和应用软件

   系统软件：操作系统、数据库**管理系统**、语言处理程序、分布式软件系统、网络软件系统、标准库程序、服务性程序等。应用软件：各种科学计算类程序、工程设计类程序、数据统计与处理程序等。

2. 三个级别的语言

   (a) 机器语言。

   (b) 汇编语言。

(c) 高级语言。

将高级语言/汇编语言程序转换为机器语言程序的软件系统称为翻译程序，具体有以下三类：

(a) 汇编程序（汇编器）

(b) 解释程序（解释器）

(c) 编译程序（编译器）

3. 软件和硬件的逻辑功能等价性

对某一功能来说，既可以由硬件实现，又可以由软件实现，对于用户来说，它们在功能上是等价的。

硬件实现的性能往往优于软件实现。

1.2.4 计算机系统的层次结构

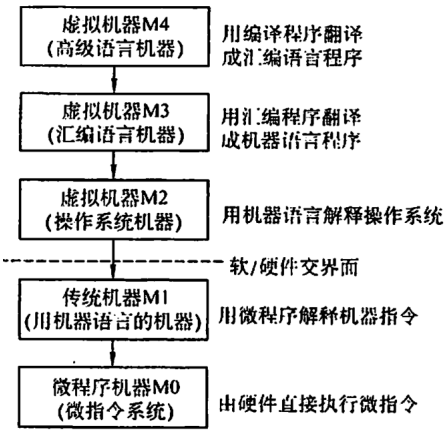

图 1: 计算机系统的多级层次结构

如图1所示：

- 第一级是微程序机器层，是实在的硬件层。
- 第二级是传统机器语言层，也是实在的硬件层。
- 第三级是操作系统层，这一层也成为混合层。

- 第四级是汇编语言层。
- 第五级是高级语言层。

软件和硬件的交界面就是 ISA。

1.2.5 计算机系统的工作原理

1. 从源程序到可执行文件

   gcc 编译流程：预处理-编译-汇编-链接

2. 指令执行过程的描述

   (a) 取指令：PC→MAR→M→MDR→IR

   (b) 分析指令：OP(IR)→CU

   (c) 执行指令：Ad(IR)→MAR→M→MDR→ACC

   表示数据通路时括号可以省略，但运算时括号不能省略。如果题目中的括号没有省略，答题时也最好不要省略。

例题 1. 冯 · 诺依曼机的基本工作方式是 控制流驱动方式。

例题 2. 下列（ ）不属于系统软件。

A. 数据库系统

B. 操作系统

C. 编译程序

D. 以上三种都属于系统程序

解答： A. 本题易误选 D。注意数据库系统 = 数据库 + 数据库管理系统 + 应用系统 + 数据库管理员，其中仅数据库管理系统是系统程序。

例题 3. 相联存储器 既可按地址寻址又可按内容寻址。

例题 4. 【2009 统考】冯 · 诺依曼计算机中指令和数据均以二进制形式存放在存储器中，CPU 区分它们的依据是（ ）

A. 指令操作码的译码结果

B. 指令和数据的寻址方式

C. 指令周期的不同阶段

D. 指令和数据所在的存储单元

解答： C。通常取指阶段取出的是指令，执行阶段取出的是数据。

1.3 计算机的性能指标

1.3.1 计算机的主要性能指标

1. 字长，指计算机进行一次整数运算所能处理的二进制数据的位数。
2. 数据通路带宽
3. 主存容量。MAR 的位数反映了存储单元的个数，MDR 的位数反映了存储单元的字长。
4. 运算速度
   (a) 吞吐量，主要取决于主存的存储周期。
   (b) 响应时间，通常包括 CPU 时间和等待时间。
   (c) CPU 时钟周期，是主频的倒数，是 CPU 中最小的时间单位。
   (d) 主频。
   (e) CPI：执行一条指令所需要的时钟周期数。一般是一个平均值。

(f) CPU 执行时间

(g) MIPS：百万条指令每秒

(h) 浮点操作次数每秒系列单位：MFLOPS、GFLOPS、TFLOPS、PFLOPS、EFLOPS、ZFLOPS、EFLOPS

注意：在描述存储容量、文件大小等时，K、M、G、T 通常用 2 的幂次表示，而描述速率、频率等时，通常用 10 的幂次表示。

例题 5. 从用户观点看，评价计算机系统性能的综合参数是（ ）

A. 指令系统

B. 吞吐率

C. 主存容量

D. 主频

解答： B

二　数据的表示和运算

2.1　数制与编码

2.1.1　进位计数制及其相互转换

计算机系统中所有信息都用二进制编码的原因：

- 使用有两个稳定状态的物理器件就可以表示二进制数的每一位，成本低廉。
- 二进制的“0”和“1”恰好与逻辑值“真”和“假”相对应。
- 二进制的编码和运算规则都比较简单。

1. 进位计数法

一个 r 进制数（$K_nK_{n-1}\cdots K_0K_{-1}\cdots K_{-m}$）的数值可表示为

$$K_nr^n+K_{n-1}r^{n-1}+\cdots+K_0r^0+K_{-1}r^{-1}+\cdots+K_{-m}r^{-m}=\sum_{i=n}^{-m}K_ir^i$$

式中，r 是基数；r^i 是第 i 位的位权（整数位最低规定为第 0 位）；K_i 的取值可以是 $0,1,\cdots,r-1$ 共 r 个数码中的任意一个。

2. 不同进制数之间的相互转换

   (a) 二进制转八进制和十六进制

   二进制混合数（整数 + 小数）的整数部分，高位补 0 至对齐（转八进制则对齐到 3 位的整数倍，转十六进制则对齐到 4 位的整数倍）；小数部分，低位补 0 至对齐，然后直接转换。

   八、十六进制互转时，可借助二进制作为中介。

   (b) 任意进制数转为十进制数：各位数码和权值相乘然后加起来

   (c) 十进制数转为任意进制数

   将十进制混合数拆成整数部分和小数部分，整数部分除基取余，小数部分乘基取整，最后再拼接起来。

   乘基取整：小数部分乘基取整，最先取得的整数为数的最高位，最后取得的整数为数的最低位，乘基为 1.0 或满足精度要求时结束。

   每个二进制小数都可以用十进制小数精确表示，但反之则不一定。

2.1.2 BCD 码

本节已从新大纲中删除，故略。

2.1.3 定点数的编码表示

1. 机器数的定点表示

2. 原码、反码、补码、移码

   (a) 原码表示法

   - 若字长为 $n+1$，则原码**小数**的表示范围为 $-(1-2^{-n})\le x\le 1-2^{-n}$（关于原点对称）

- 若字长为 $n+1$，则原码**整数**的表示范围为 $-(2^n-1) \le x \le 2^n-1$（关于原点对称）

真值 0 的原码表示有两种，即 +0 和-0。

(b) 补码表示法

小数补码比原码多表示一个-1，整数补码比原码多表示一个-2。

变形补码：又称模 4 补码，双符号位的补码小数，符号位 00 表示正，11 表示负。模 4 补码的符号位的存储仅仅需要一**位**而不是两位，因为任何一个正确的模 4 补码，符号位的两位都是相同的，如果不同表示发生了溢出。

补码真值互转：整数直接按原码转换，负数则各位取反加 1。

另一种方法：从后往前找到第一个 1，该 1 不变，前面所有位（含符号位）取反，符号去掉即可。

(c) 反码表示法

(d) 移码表示法

移码就是在真值上加一个偏置值：

$$[x]_{移} = 2^n + x \ (-2^n \le x < 2^n，其中机器字长为 n+1)$$

移码有如下特点：

- 0 的表示是唯一的。
- 同一个真值的补码和移码只差一个符号位。补码符号位取反，数值部分不变即得移码，反之亦然。
- 移码保持了数据的原有大小顺序，移码大真值就大，移码小真值就小。

2.1.4 整数的表示

1. 无符号整数的表示
2. 带符号整数的表示

例题 6. 对于相同位数（设为 N 位，不考虑符号位）的二进制补码小数和十进制小数，比值 $\frac{\text{二进制小数能表示的数的个数}}{\text{十进制小数能表示的数的个数}}$ 为 $\underline{(0.2)^N}$

解答： N 位二进制小数可以表示的数的个数为 $1+2^0+2^1+\cdots+2^{N-1}=2^N$，而十进制小数能表示的数的个数为 10^N，二者的商为 $(0.2)^N$。这也是计算机的运算中会出现误差的原因，它表明仅有 $(0.2)^N$ 的概率的十进制数可以精确地用二进制表示。

2.2　运算方法和运算电路

2.2.1　基本运算部件

408 很少涉及本节内容。
加法器是 ALU 的核心部件。

1. 一位全加器
2. 串行进位加法器

   串行进位加法器的最长运算时间主要是由进位信号的传递时间决定的，加法器本身的求和时间只是次要因素。
3. 并行进位加法器
4. 带标志加法器
5. 算数逻辑单元（ALU）

2.2.2　定点数的移位运算

1. 算术移位

   算术移位的对象是**有符号数**。

   算术移位不管怎么移，符号位永远是不变的，也即移位只作用于数值位。

   - 正数无论码制，左右移都一律补 0。
   - 负数稍复杂：
     - ◇ 原码：一律补 0
     - ◇ 反码：一律补 1
     - ◇ 补码：左移补 0，右移补 1

表 1: 不同机器数算术移位后的空位添补规则

	码制	添补代码
正数	原码、补码、反码	0
负数	原码	0
	补码	左移添 0
		右移添 1
	反码	1

2. 逻辑移位

   逻辑移位将移位对象视为**无符号数**。无论码制、左右移，一律补 0。

3. 循环移位

   又分带进位标志 CF 的循环移位和不带进位标志的循环移位。

   带 CF 的循环移位是将移出的位送入标志位 CF，并将 CF 的值送入腾出的位。

   即使是不带 CF 的循环移位，实际上 CF 的值也是会变的，但 CF 只是单纯输入，不再输出。输入的值是移出的位。

   循环移位操作常用于交换寄存器的高低字节。

2.2.3 定点数的加减运算

1. 补码的加减法运算

2. 补码加减运算电路

   这里有一个各标志位的含义，可能有用。待补。

3. 溢出判别方法

   仅当两个符号相同的数相加或者两个符号相异的数相减才有可能产生溢出。判断溢出的方法有三种：

   - 采用一位符号位：无论加减，只要参加操作的两个数符号相同，结果又与源操作数符号不同，就表示溢出。

- 采用双符号位：运算结果的两个符号位相同，表示未溢出；若不同则表示溢出。
  - ◇ 00：表示结果为正，无溢出。
  - ◇ 01：表示结果正溢出。
  - ◇ 10：表示结果负溢出。
- 采用一位符号位根据数据位的进位情况判断溢出
  当最高符号位的进位和数值位的进位不同时，说明发生溢出。

2.2.4　定点数的乘除运算

待补

2.2.5　C 语言中的整数类型及类型转换

本节为 408 常考内容。

当大字长变量向小字长变量强制类型转换时，多余的高位直接截掉，低位直接赋值，而不是进行类似循环溢出的操作。

短字长向长字长转换时，不仅要使相应的位值相等，还要对高位进行扩展：

- 如果原数字是无符号整数，则进行零扩展
- 如果原数字带符号，则进行符号扩展。

char 类型为 8 位无符号整数，转为 int 时高位补 0 即可。

2.2.6　数据的存储和排列

1. 数据的“大端方式”和“小端方式”存储
   - 大端：低字节放高地址
   - 小端：低字节放低地址
2. 数据按“边界对齐”方式存储

   注意：题目涉及**结构体**的，一定要留意对齐方式（一般都是边界对齐的）！

2.3 浮点数的表示与运算

2.3.1 浮点数的表示

1. 浮点数的表示格式

   通常，浮点数可表示为

   $$N = (-1)^S \times M \times R^E$$

   其中：

   - S 决定浮点数的符号，可取 0 或 1。
   - M 是尾数，是一个二进制定点小数，通常用原码表示。
   - E 是阶码，是一个二进制定点整数，通常用移码表示。
   - R 是隐含的基数，可取 2、4、16 等。

2. 浮点数的表示范围

   - 数据产生上溢（正上溢、负上溢）时，计算机必须停机处理溢出。
   - 数据产生下溢（正下溢、负下溢）时，计算机将其当作机器零处理。

3. 浮点数的规格化

   - 左规：尾数的最高位不是有效位，即尾数为 $0.0\cdots$ 时，进行左规，尾数每左移一次，阶码减一（仅基数为 2 时）。
   - 右规：尾数的有效位进到小数点前面时，进行右规，尾数右移一位，阶码加一（仅基数为 2 时）。

   基数不同，浮点数的规格化形式也不同。基数为 2 时，原码规格化数的尾数最高位一定是 1；基数为 4 时，原码规格化形式的尾数最高两位不全为 0。

4. IEEE 754 标准

   IEEE 754 规定：浮点数的基数隐含为 2，尾数采用隐藏位策略的原码表示，阶码用移码表示。对于短浮点数，偏置值为 127；对于长浮点数，偏置值为 1023。存储阶码之前，要先把真值加上偏置值；反之在解析真值时，需要将阶码减去偏置值。

表 2: IEEE 754 浮点数的格式

类型	数符	阶码	尾数数值	总位数	偏置值	
					十六进制	十进制
短浮点数	1	8	23	32	7FH	127
长浮点数	1	11	52	64	3FFH	1023
临时浮点数	1	15	64	80	3FFFH	16383

规格化的二进制浮点数，数值最高位总是 1，因此此位省略，尾数实际上能多表示一位有效位。

5. 定点、浮点表示的区别

2.3.2 浮点数的加减运算

分为以下几步：

1. 对阶

   原则是小阶向大阶对齐，即将阶码数小的浮点数的阶码逐位增加 1，同时尾数逐位右移，直到两个数阶码相等为止。

2. 尾数求和

3. 规格化

   尾数求和的结果不一定是规格化的，因此需要规格化。

4. 舍入

   为保证精度，一般将右移移出的位保留下来，在最后得出结果时舍入成 IEEE 754 格式。

   - 0 舍 1 入法：类似四舍五入。若保留下来的最高位为 1，则在尾数的末位加 1，否则就舍去。
   - 恒置 1 法：无脑将右移后的尾数末位置为 1。
   - 截断法：直接截断，最简单。

5. 溢出判断

三 存储系统

3.1 存储器概述

3.1.1 存储器的分类

1. 按在计算机中的作用（层次）分类
   - 主存储器
   - 辅助存储器
   - 高速缓冲存储器（Cache）
2. 按存储介质分类
   - 磁表面存储器（磁盘、磁带）
   - 磁芯存储器
   - 半导体存储器
     - ◇ MOS 型存储器
     - ◇ 双极型存储器
   - 光存储器（光盘）
3. 按存取方式分类
   - 随机存储器（RAM）：又分静态 RAM 和动态 RAM
   - 只读存储器（ROM）：可与 RAM 共同作为主存的一部分，统一构成主存的地址域。**同样支持随机访问**。
   - 串行访问存储器。包括：
     - ◇ 顺序存取存储器（如磁带）
     - ◇ 直接存取存储器（如磁盘、光盘）
4. 按信息的可保存性分类
   - 易失性存储器，如 RAM
   - 非易失性存储器，如 ROM、磁表面存储器和光存储器。

3.1.2 存储器的性能指标

三大性能指标：

1. 存储容量。= 存储字数 × 字长
2. 单位成本。
3. 存储速度。
    - 存取时间：启动一次存储器操作到完成该操作所经历的时间。
    - 存取周期：又称读写周期或访问周期，是指存储器进行一次完整的读写操作所需的全部时间，即连续两次独立访问存储器操作之间所需的最小时间间隔。
    - 主存带宽：表示每秒从主存进出信息的最大数量。

存取周期通常大于存取时间。因为存储器在读写操作之后往往需要一段恢复内部状态的复原时间。

3.1.3 多级层次的存储系统

存储系统层次结构主要体现在 Cache-主存层和主存-辅存层。

- 主存和 Cache 之间的数据调动由硬件自动完成，对所有程序员都是透明的。
- 主存和辅存之间的数据调动由硬件和操作系统共同完成，对应用程序员是透明的。

3.2 主存储器

3.2.1 SRAM 芯片和 DRAM 芯片

DRAM 的常见刷新方式：

- 集中刷新：在一个规定的**刷新周期**内，对所有行进行刷新。这段时间 RAM 无法读写，也即存在死区。
- 分散刷新：将刷新分散到每个存取周期中进行。无死区，但存取周期变长。
- 异步刷新：二者的结合。顺序刷新各行。也有死区，但是死区时间变短。

表 3: SRAM 和 DRAM 各自的特点

类型 / 特点	SRAM	DRAM
存储信息	触发器	电容
破坏性读出	非	是
需要刷新	不要	需要
送行列地址	同时送	分两次送
运行速度	快	慢
集成度	低	高
存储成本	高	低
主要用途	高速缓存	主机内存

DRAM 的刷新单位是行，刷新时不需要选片，因为所有芯片同时被刷新。SRAM 和 DRAM 的比较见表3。

3.2.2 只读存储器 ROM

ROM 包括 MROM、PROM、EPROM、EEPROM、Flash 等。

3.2.3 主存储器的基本组成

本节易考“芯片最少需要多少引脚”的问题。

对于 SRAM，地址线的条数是存储单元数量的对数（行列信号同时传输），而数据线的条数就是一个存储单元的长度，此外还要附加若干条片选线（根据芯片数量确定，一片芯片时片选线为一位宽）和读写控制线。读写控制线一般为 2 条（WE、RD）。

对于 DRAM，一般采用地址复用技术。地址线的数量应当减半，同时片选线数量应该加倍，因为分出了行通选线和列通选线（先行后列）。其他和 SRAM 一致。

例题 7. 某一 SRAM 芯片，其容量为 1024×8 位，除电源和接地端外，该芯片的引脚的最小数目为 21。

解答： 地址线 10 条，数据线 8 条，片选线 1 条，读写控制线 2 条，共 21 条。

例题 8. 某一 DRAM 芯片，采用地址复用技术，容量为 1024×8 位，除电源线和接地端外，该芯片的引脚数最少是 17。

解答： 地址线 5（$10 \div 2$）条，数据线 8 条，片选线 2 条（行通选 + 列通选），控制线 2 条，共 17 条。

3.2.4 多模块存储器

1. 单体多字存储器

   一次并行读出 m 个字，要求指令和数据必须连续存放。

2. 多体并行存储器

   - 高位交叉编址（顺序方式）

     不能提高存储器的吞吐率。

   - 低位交叉编址（408 常用的名字是“交叉方式”）

     假设模块存取一个字的存取周期为 T，总线传送周期为 r，为实现流水线方式存取（某个模块被再次访问时已经准备好），存储器交叉模块数应大于等于

     $$m = \frac{T}{r}$$

     其中 m 称为交叉存取度。每隔 r 时间延迟后启动下一个模块。这样，连续存取 m 个字所需的时间为

     $$t_1 = T + (m-1)r$$

     而顺序方式连续读取 m 个字所需的时间为 $t_2 = mT$。

3.3 主存储器与 CPU 的连接

3.3.1 连接原理

略

3.3.2 主存容量的扩展

1. 位扩展法

   各芯片连接地址线的方式相同，但连接数据线的方式不同，在某一时刻选中所有的芯片，所以片选信号要连接到所有的芯片。

2. 字扩展法

   各芯片连接地址线的方式相同，连接数据线的方式也相同，在某一时刻只需选中部分芯片，所以将片选信号直接连接到相应的芯片，或者通过译码器连接。

3. 字位同时扩展法

注意：MAR 的位数应该和 CPU 支持的内存大小保持一致，而不是和系统安装的内存大小保持一致。

3.3.3 存储芯片的地址分配和片选

1. 线选法

2. 译码片选法

例题 9. 【2018 统考】假定 DRAM 芯片中存储阵列的行数为 r、列数为 c，对于一个 2K×1 位的 DRAM 芯片，为保证其地址引脚数最少，并尽量减小刷新开销，则 r、c 的取值分别是（ ）

A. 2048, 1

B. 64, 32

C. 32, 64

D. 1, 2048

解答： C。为什么不选 D？注意到 DRAM 采用行列地址线复用技术，这要求地址线需要和行数及列数中较大的那个保持一致。也就是说，行数和列数不能差得太多。

3.4 外部存储器

磁盘的平均存储时间 = 寻道时间 + 旋转延迟时间 + 传输时间。其中寻道时间和旋转延迟时间通常取平均值。

磁盘地址 = 驱动器号 + 柱面号 + 盘面号 + 扇区号。

磁盘的读写操作是串行的，不可能在同一时刻既读又写。

3.5 高速缓冲存储器

3.5.1 程序访问的局部性原理

略

3.5.2 Cache 的基本工作原理

- CPU 和 Cache 之间的数据交换以**字**为单位
- Cache 与主存之间的数据交换则以 **Cache 块**为单位

设某个程序执行期间，Cache 的总命中次数为 N_c，访问主存的总次数为 N_m，则命中率为

$$H = \frac{N_c}{N_c + N_m}$$

设 t_c 为命中时的 Cache 访问时间，t_m 为未命中时的访问时间，$1 - H$ 表示未命中率，则 Cache-主存系统的平均访问时间 T_a 为

$$T_a = Ht_c + (1 - H)t_m$$

3.5.3 Cache 和主存的映射方式

- 为了说明当前 Cache 块对应主存中的哪一块，需要为每个 Cache 块添加一个**标记（tag）**。

- 为了说明 Cache 行中的信息是否有效，需要为每个 Cache 行添加一个**有效位（valid bit）**。

地址映射有如下三种方法：

1. 直接映射

$$\text{Cache 行号} = \text{主存块号} \ \ mod \ \ \text{Cache 总行数}$$

假设 Cache 有 2^c 行，主存有 2^m 块。上式表明，主存块号的低 c 位正好是它要装入的 Cache 行号。给每个 Cache 行设置一个长为 $t = m - c$ 的标记（tag），当主存某块调入 Cache 后，就将其块号的高 t 位设置在对应 Cache 行的标记中。

2. 全相联映射

   - 优点：灵活，Cache 块的冲突概率低，空间利用率高，命中率也高；
   - 缺点：标记的比较速度慢，实现成本高，通常需要采用昂贵的**按内容寻址**的相联存储器进行地址映射。

3. 组相联映射

   将 Cache 分成若干个大小相等的组，组间采用直接映射，组内采用全相联映射。这是对上面两种方式的折中。

   组相联的成本接近直接映射，性能接近全相联映射。

3.5.4 Cache 中主存块的替换算法

可结合《操作系统》复习。
常见算法：

- 随机算法
- 先进先出算法（FIFO）
- 近期最少使用算法（LRU）
- 最不经常使用算法

3.5.5 Cache 写策略

- 写命中
  - ◇ 写直达法（write-through）：写命中时，数据同时写入 Cache 和主存。
    - ∗ 优点：简单，能随时保持主存数据的正确性。
    - ∗ 缺点：增加了访存次数，降低了 Cache 的效率。
  - ◇ 写回法（write-back）：写命中时，仅仅把数据写入 Cache，而不立即写入主存。仅当此块被换出时才写回主存。
    - ∗ 优点：减少了访存次数。
    - ∗ 缺点：存在不一致的隐患。需要为每个 Cache 行设置一个脏位。
- 写不命中
  - ◇ 写分配法（write-allocate）：加载主存中的块到 Cache 中，然后更新这个 Cache 块。
  - ◇ 写不分配法（not-write-through）：只写入主存，不动 Cache。

写不分配经常和写直达法搭配，写分配经常和写回法搭配。

3.6 虚拟存储器

3.6.1 虚拟存储器的基本概念

四 指令系统

五 中央处理器

六 总线

七 输入/输出系统