计算机组成原理

第 9 讲

左德承

哈尔滨工业大学计算学部

容错与移动计算研究中心

各检测位 C_i 所承担的检测小组为

C_1 检测的 g_1 小组包含第 1，3，5，7，9，11，…

C_2 检测的 g_2 小组包含第 2，3，6，7，10，11,…

C_4 检测的 g_3 小组包含第 4，5，6，7，12，13,…

C_8 检测的 g_4 小组包含第 8，9，10，11，12，13，14，15，24,…

g_i 小组独占第 2^{i-1} 位

g_i 和 g_j 小组共同占第 $2^{i-1}+2^{j-1}$ 位

g_i、g_j 和 g_l 小组共同占第 $2^{i-1}+2^{j-1}+2^{l-1}$ 位

例4.4 求 0101 按“偶校验”配置的汉明码

解：∵ $n=4$

根据 $2^k \geq n+k+1$

得 $k=3$

汉明码排序如下：

二进制序号	1	2	3	4	5	6	7
名称	C_1	C_2	0	C_4	1	0	1
	0	1		0			

∴ 0101 的汉明码为 **0100101**

练习1　按配偶原则配置 0011 的汉明码 4.2

解：　∵ $n=4$　　根据 $2^k \geq n+k+1$

取 $k=3$

二进制序号	1	2	3	4	5	6	7
名称	C_1	C_2	0	C_4	0	1	1
	1	0		0			

$C_1 = 3 \oplus 5 \oplus 7 = 1$

$C_2 = 3 \oplus 6 \oplus 7 = 0$

$C_4 = 5 \oplus 6 \oplus 7 = 0$

∴ 0011 的汉明码为 **1000011**

3. 汉明码的纠错过程

形成新的检测位 P_i，其位数与增添的检测位有关，如增添 3 位（$k=3$），新的检测位为 $P_4\,P_2\,P_1$。

以 $k=3$ 为例，P_i 的取值为

$$P_1 = \overset{C_1}{1} \oplus 3 \oplus 5 \oplus 7$$

$$P_2 = \overset{C_2}{2} \oplus 3 \oplus 6 \oplus 7$$

$$P_4 = \overset{C_4}{4} \oplus 5 \oplus 6 \oplus 7$$

对于按“偶校验”配置的汉明码不出错时 $P_1=0$，$P_2=0$，$P_4=0$

例4.5 已知接收到的汉明码为 0100111（按配偶原则配置）试问要求传送的信息是什么？

解：纠错过程如下

$P_1 = 1 \oplus 3 \oplus 5 \oplus 7 = 0$ 无错

$P_2 = 2 \oplus 3 \oplus 6 \oplus 7 = 1$ 有错

$P_4 = 4 \oplus 5 \oplus 6 \oplus 7 = 1$ 有错

$\therefore \quad P_4P_2P_1 = 110$

第 6 位出错，可纠正为 0100101，

故要求传送的信息为 0101。

练习2 写出按偶校验配置的汉明码

0101101 的纠错过程

$P_4 = 4 \oplus 5 \oplus 6 \oplus 7 = 1$

$P_2 = 2 \oplus 3 \oplus 6 \oplus 7 = 0$

$P_1 = 1 \oplus 3 \oplus 5 \oplus 7 = 0$

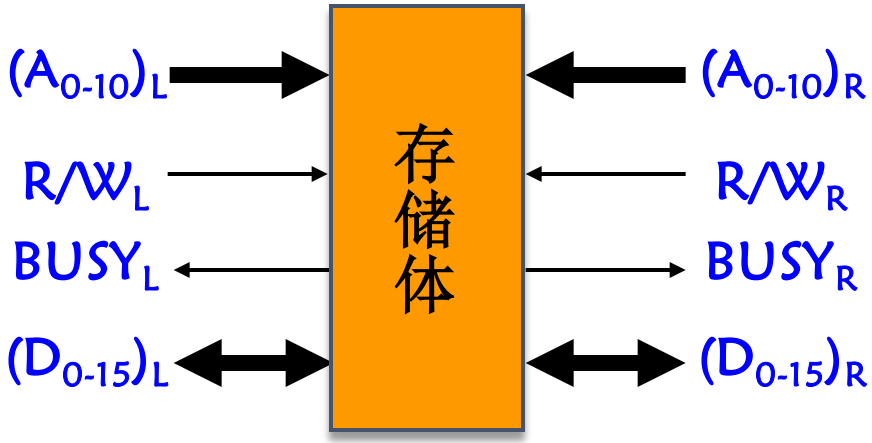
$\therefore\ P_4 P_2 P_1 = 100$　　第 4 位错，可不纠

练习3 按配奇原则配置 0011 的汉明码

配奇的汉明码为 0101011

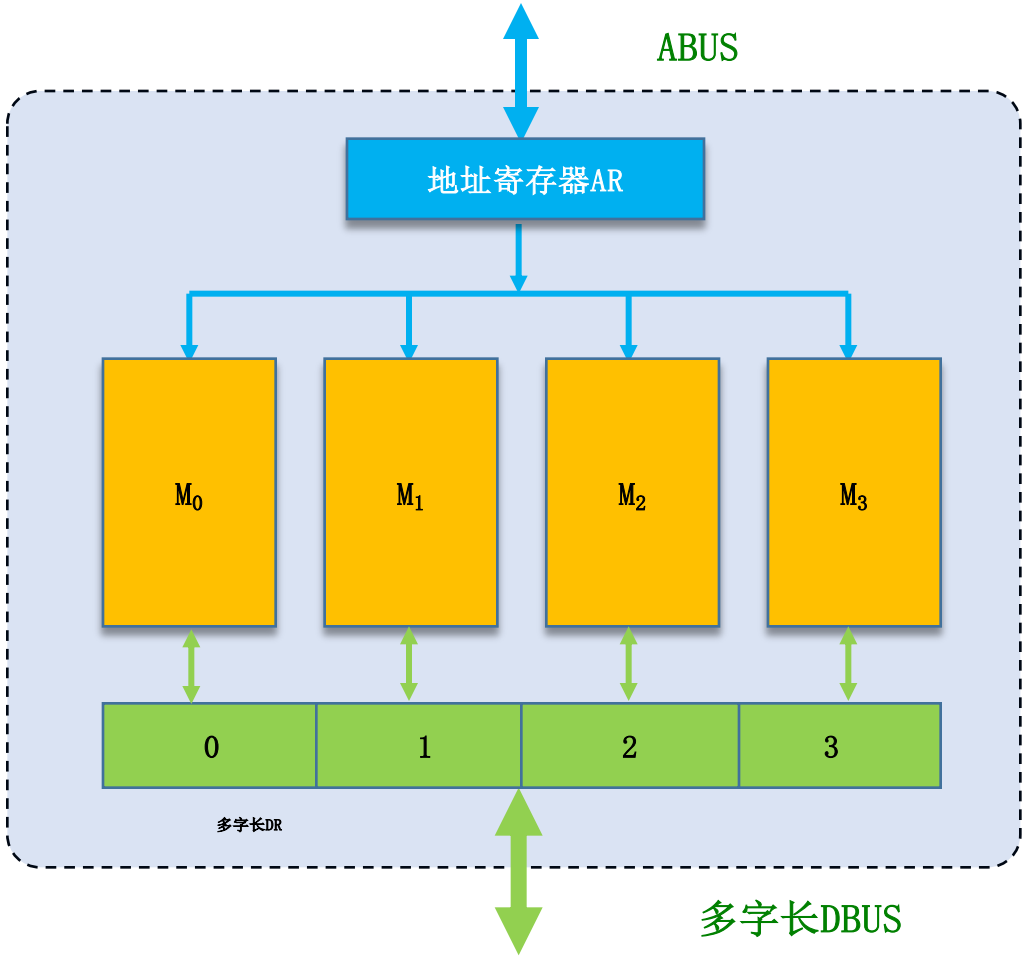
七、提高访存速度的措施

- 采用高速器件
- 采用层次结构 Cache –主存
- 调整主存结构
  - 单体多字：增加字长，在每个存储周期中存取多个字。
  - 多体并行：将主存划分为多个模块，多模块并行
- 采用双端口存储器



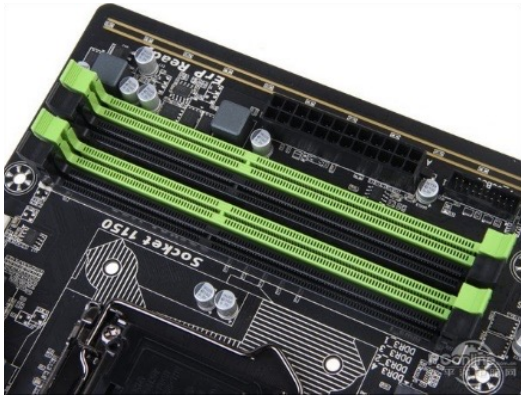
1. 单体多字系统

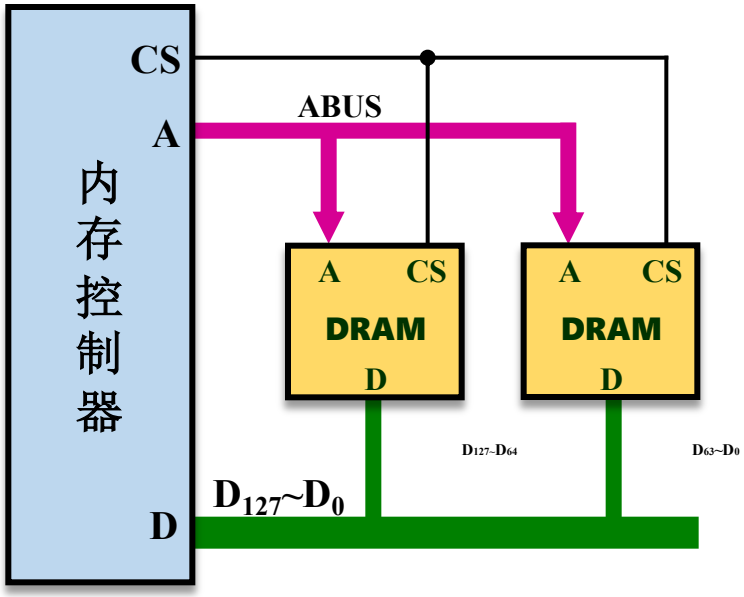
- 多个单字长存储模块
  同步并发
- 共用一个地址寄存器
- 单存储周期内访问多个存储字
- 性能线性增长，总线位宽变化



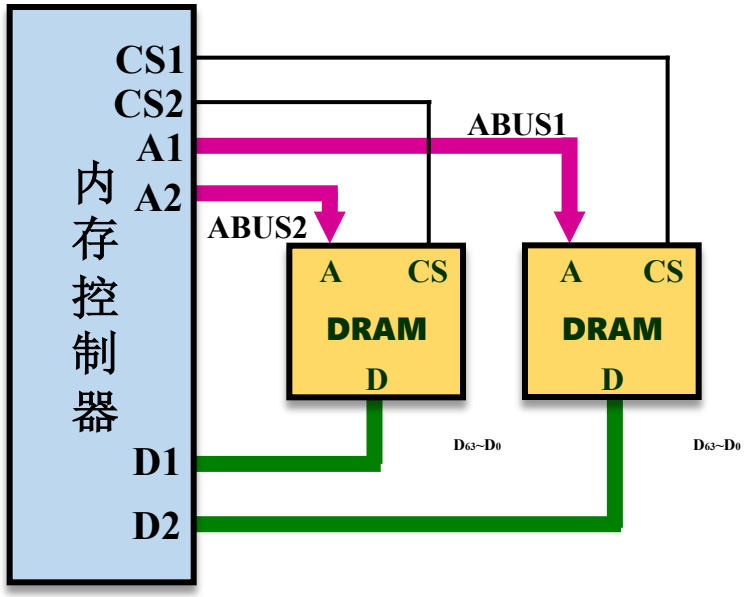
多通道内存

- 两条8G内存条
- 单条16G内存条　性能差异?





单体多字存储器



多体多字存储器

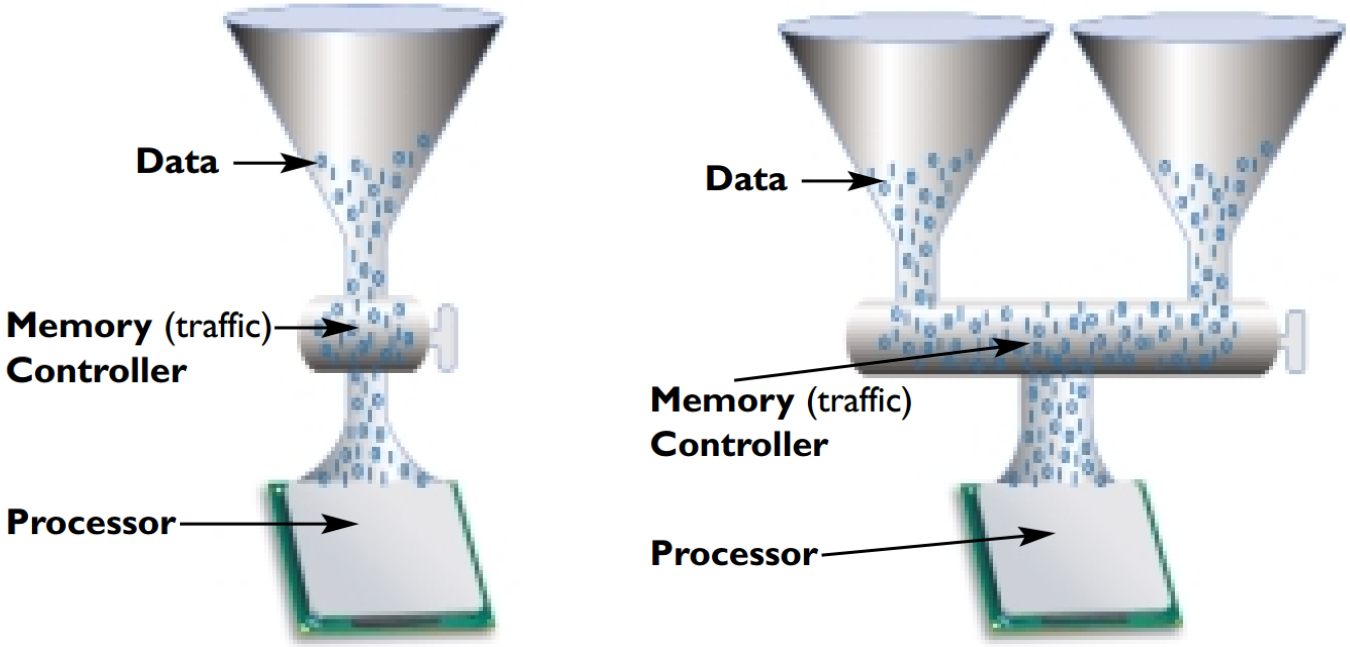
双通道内存性能评测

SiSoftware Sandra Pro Business 2011

HP DL120 G7　Intel SandyBridge

内存通道	双通道8GB	单通道8GB
总体内存性能	17.52GB/s	9GB/s
缓存/内存带宽	95.23GB/s	73.42GB/s
内部数据高速缓存	411.14GB/s	410.37GB/s
二级板载高速缓存	344GB/s	346GB/s
三级板载高速缓存	173.29GB/s	174.54GB/s

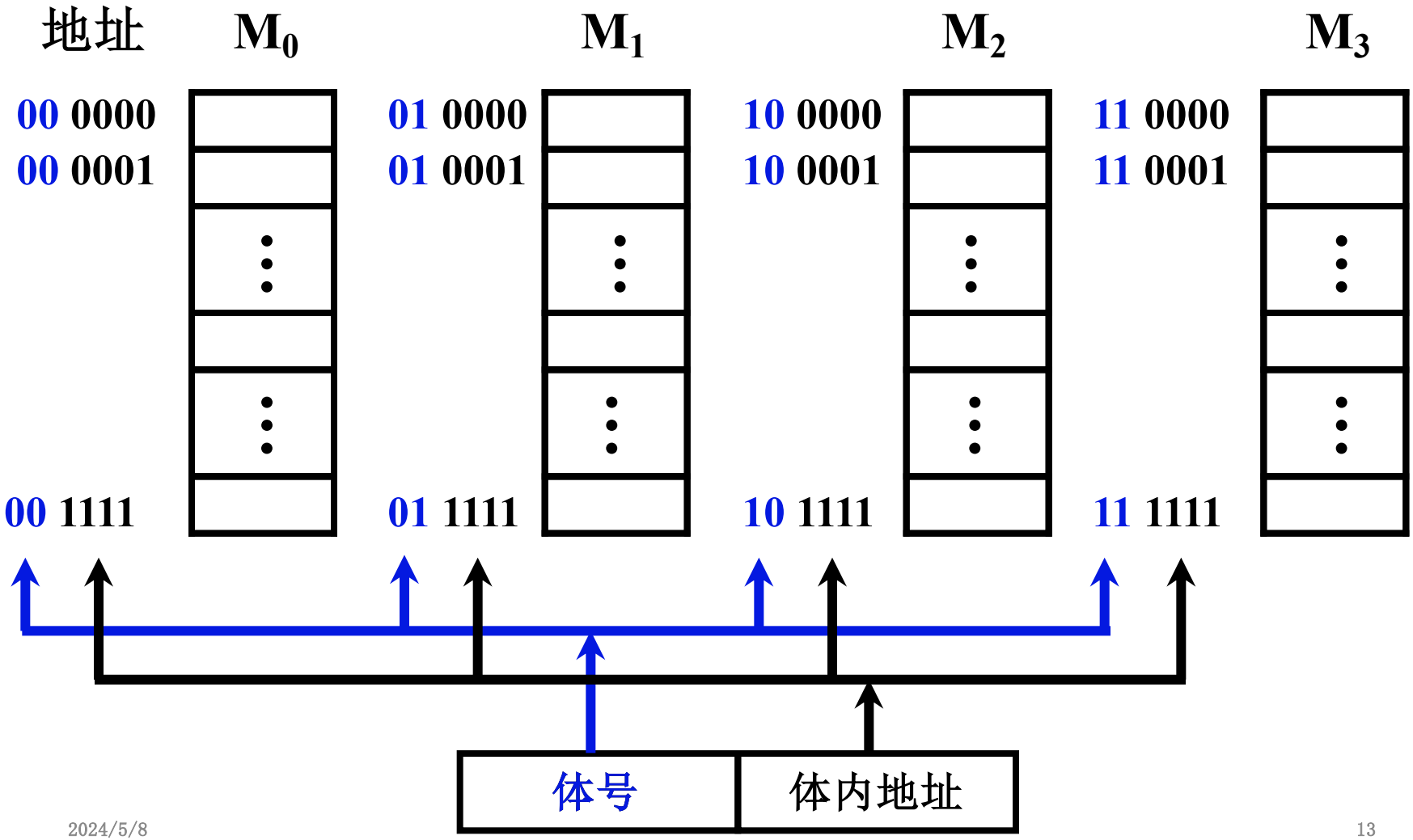
单通道内存与双通道内存



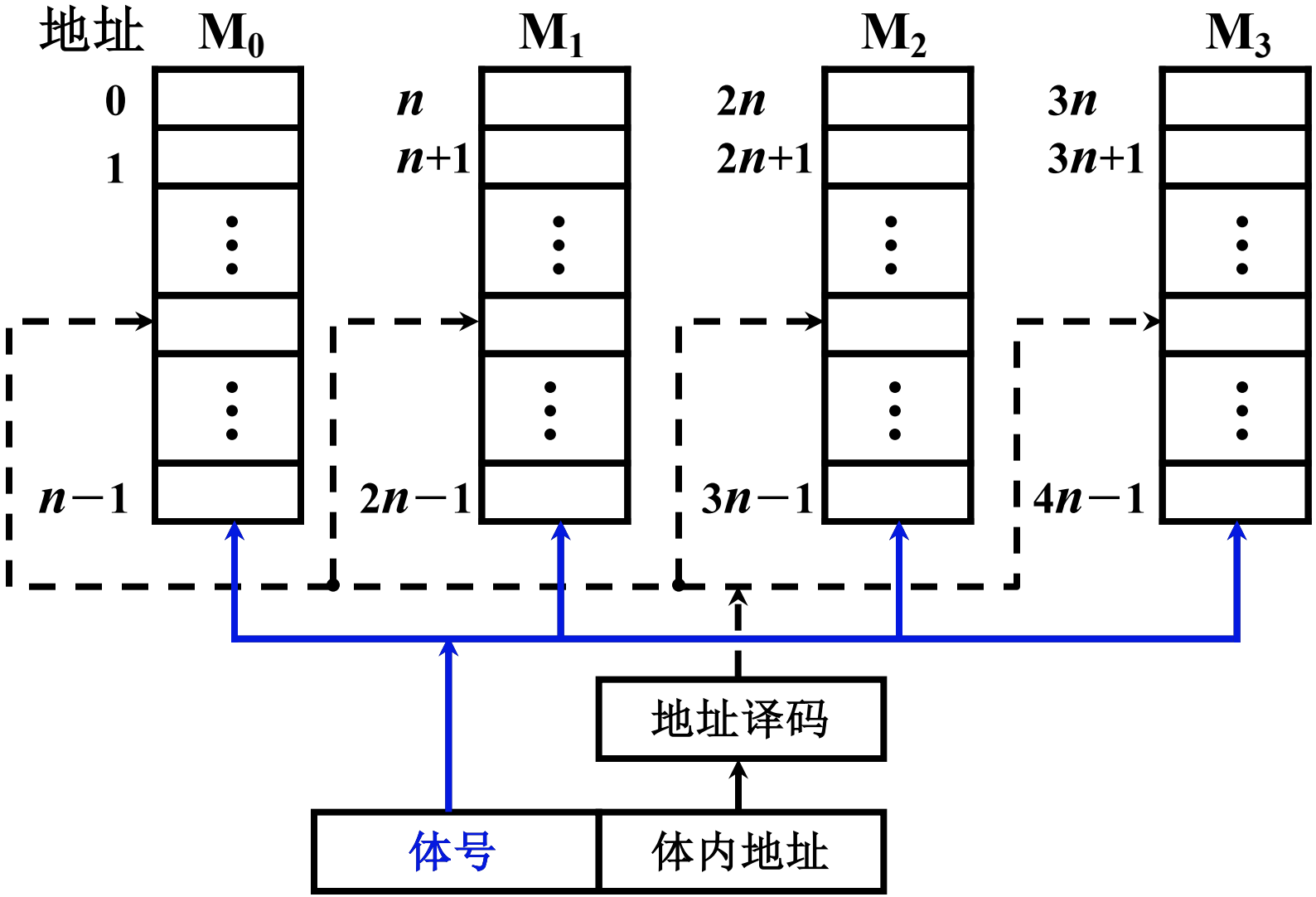
2. 多体并行系统

4.2

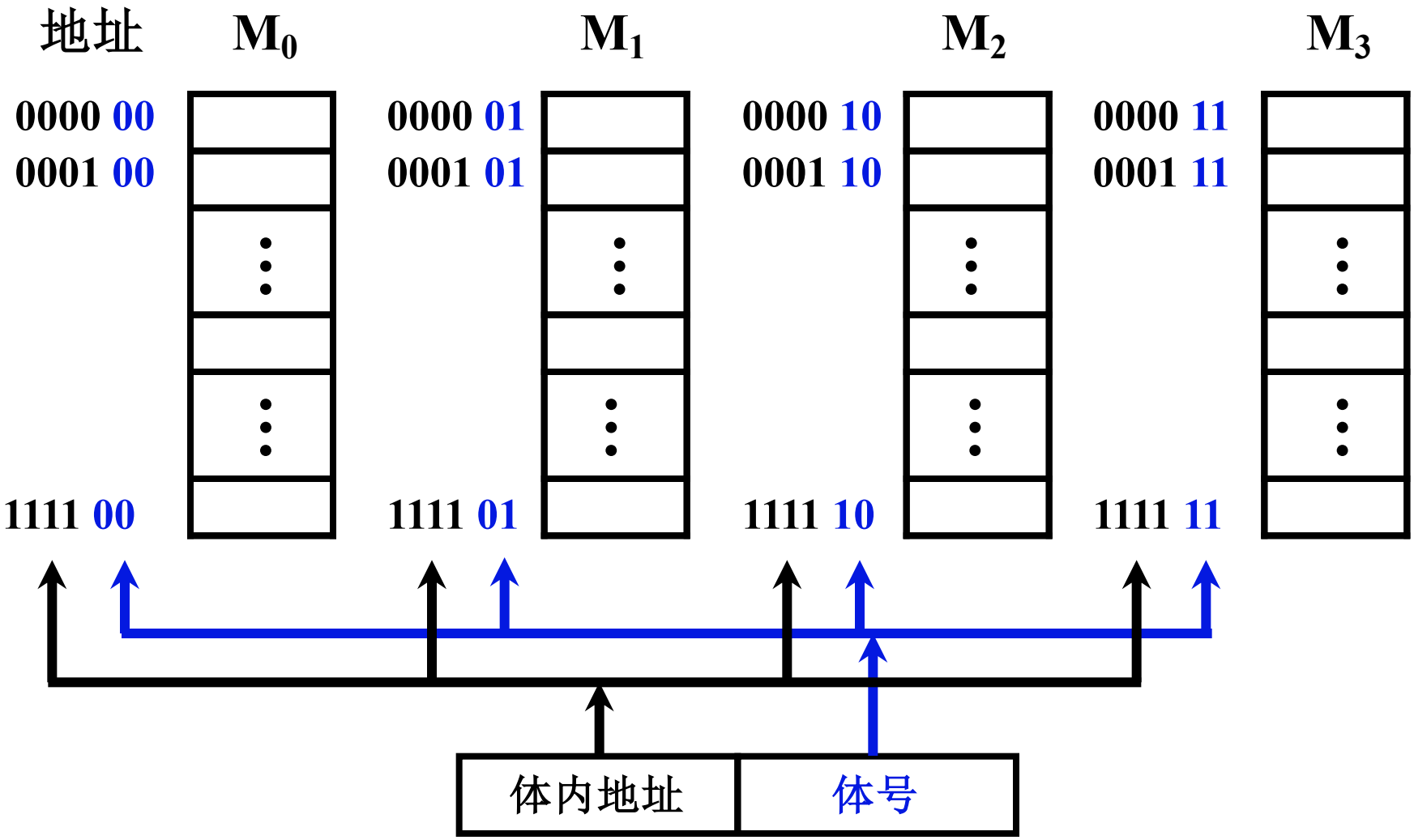
(1) 高位交叉　　各个体内顺序编址



(1) 高位交叉　各个体内顺序编址

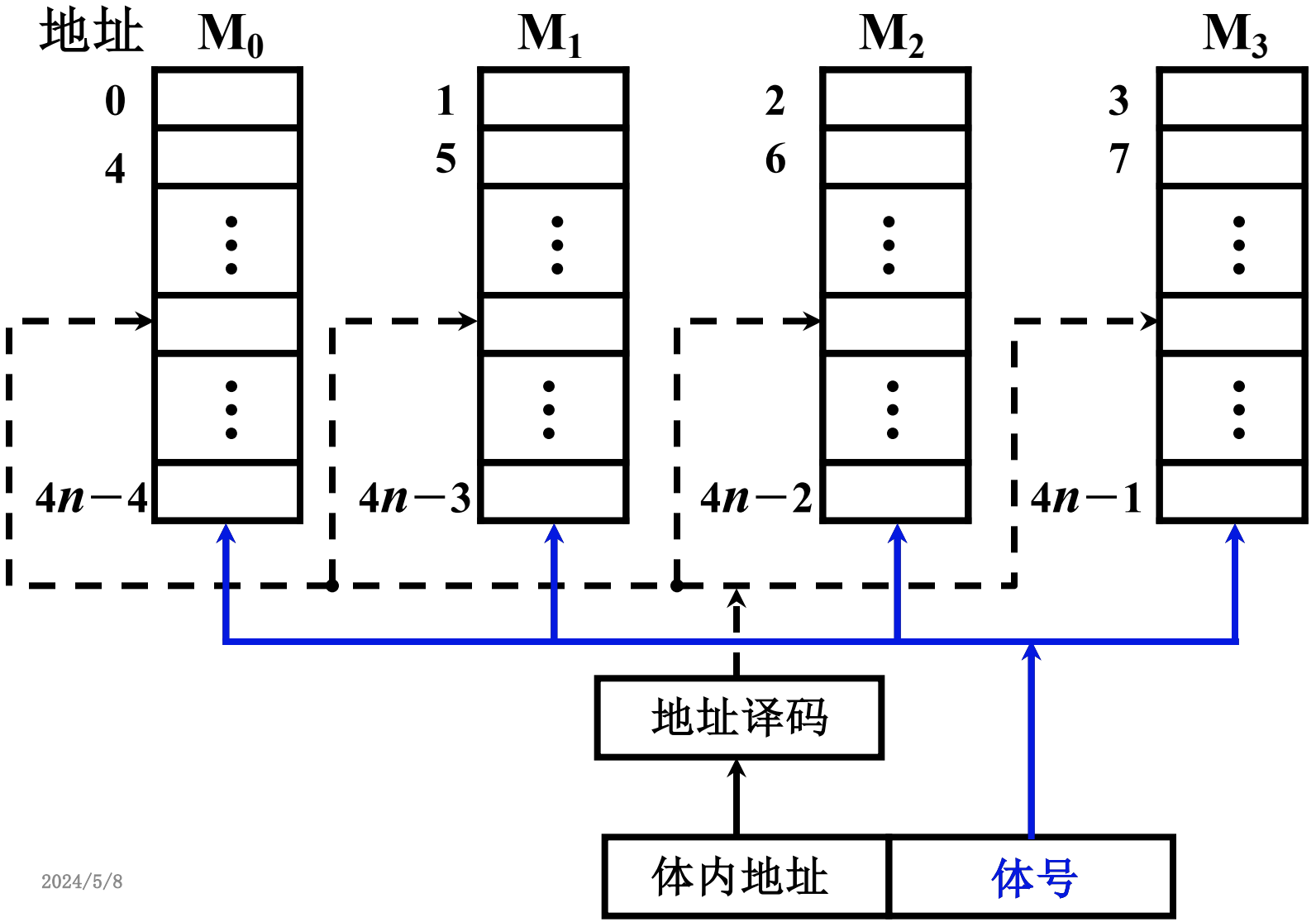


(2) 低位交叉　各个体轮流编址



(2) 低位交叉　各个体轮流编址

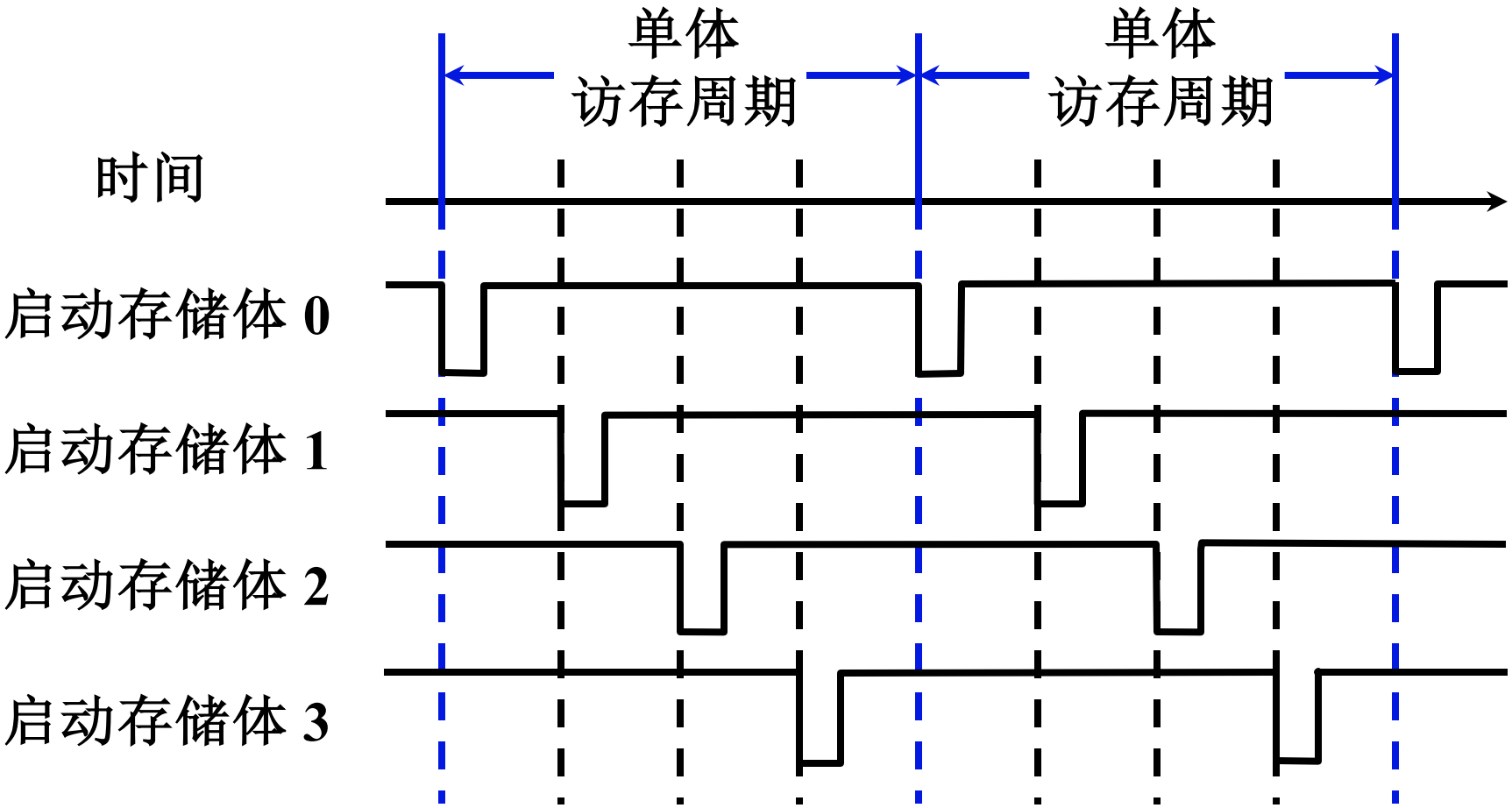
4.2



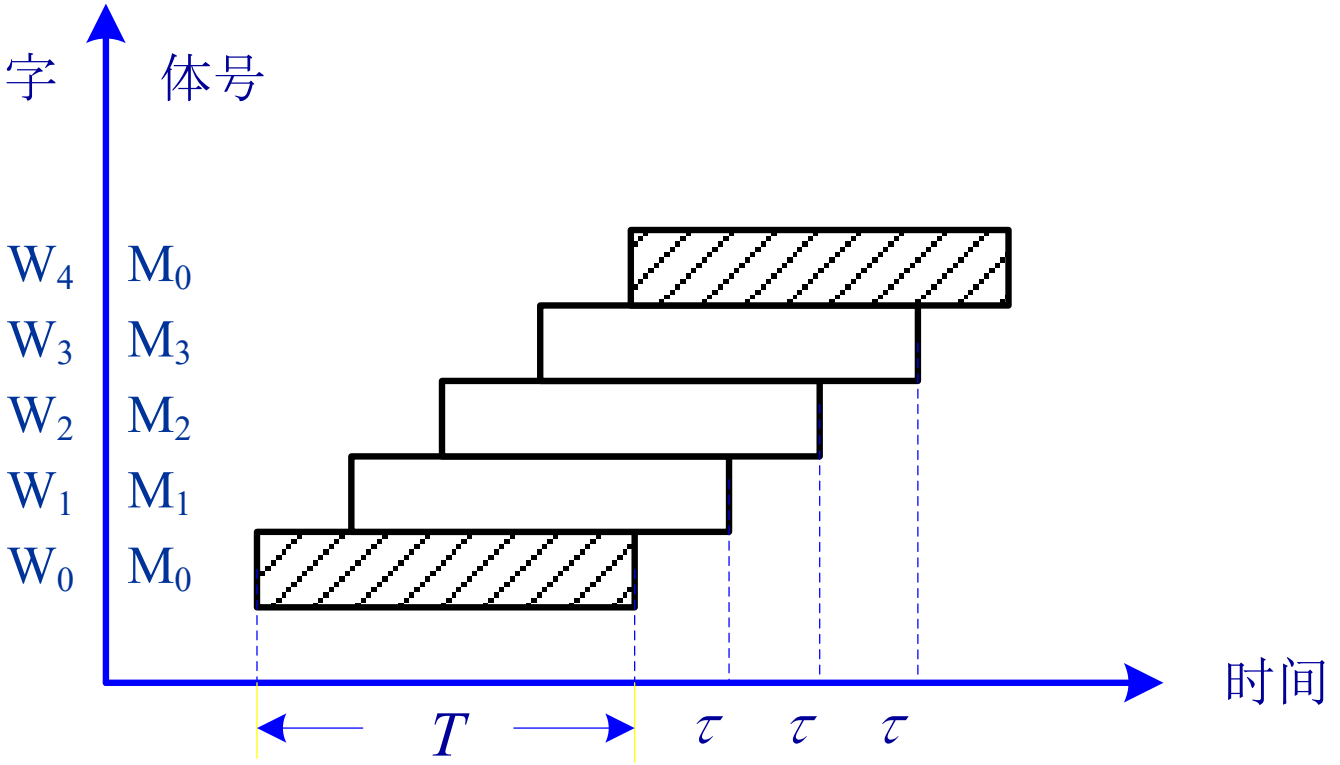
低位交叉的特点

4.2

在不改变存取周期的前提下，增加存储器的带宽



设四体低位交叉存储器，存取周期为T，总线传输周期为τ，为实现流水线方式存取，应满足 $T = 4\tau$。



连续读取 4 个字所需的时间为 $T+(4-1)\tau$

3.高性能存储芯片

(1) SDRAM (同步 DRAM)

在系统时钟的控制下进行读出和写入

CPU 无须等待

(2) DDR (Double Data Rate) SDRAM

时钟周期的上沿和下沿分别进行两次数据传输，从而实现双倍数据传输速率

(3) 带 Cache 的 DRAM

在 DRAM 的芯片内 集成 了一个由 SRAM 组成的 Cache，有利于 猝发式读取

第 4 章 存 储 器

4.1 概述

4.2 主存储器

4.3 高速缓冲存储器

4.4 辅助存储器

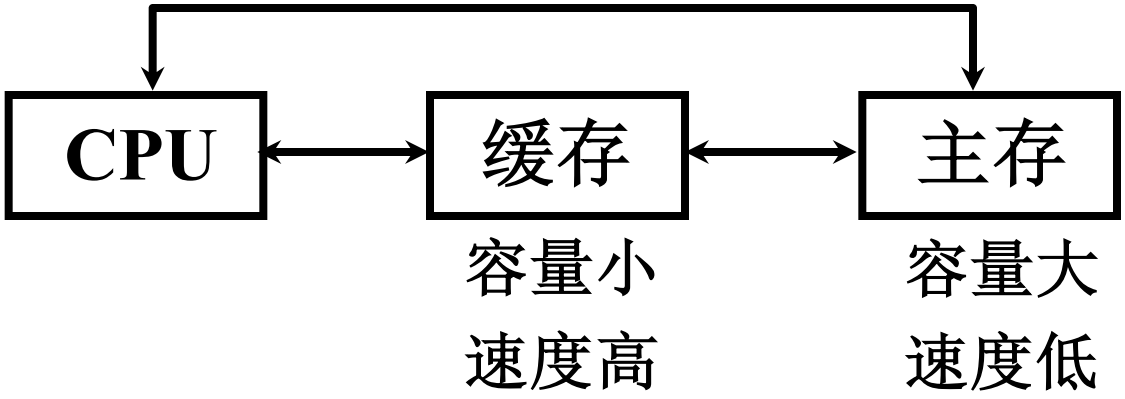
4.3 高速缓冲存储器

一、概述

1. 问题的提出

避免 CPU “空等” 现象

CPU 和主存（DRAM）的速度差异



程序访问的局部性原理

(1) 程序访问的局部性原理

对于绝大多数程序来说，程序所访问的指令和数据在地址上不是均匀分布的，而是相对簇聚的。

程序访问的局部性包含两个方面：

- 时间局部性：程序马上将要用到的信息很可能就是现在正在使用的信息。
- 空间局部性：程序马上将要用到的信息很可能与现在正在使用的信息在存储空间上是相邻的。

(2) 局部性举例

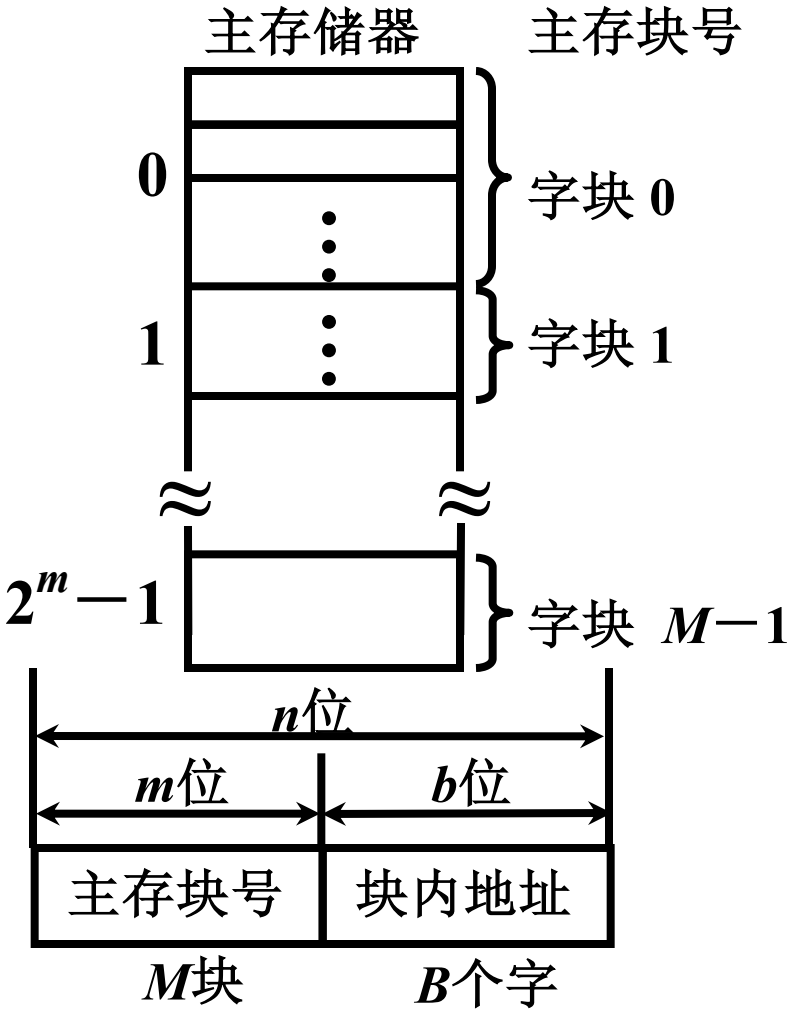

```
sum = 0;
for (i = 0; i < n; i++)
    sum += a[i];
return sum;
```

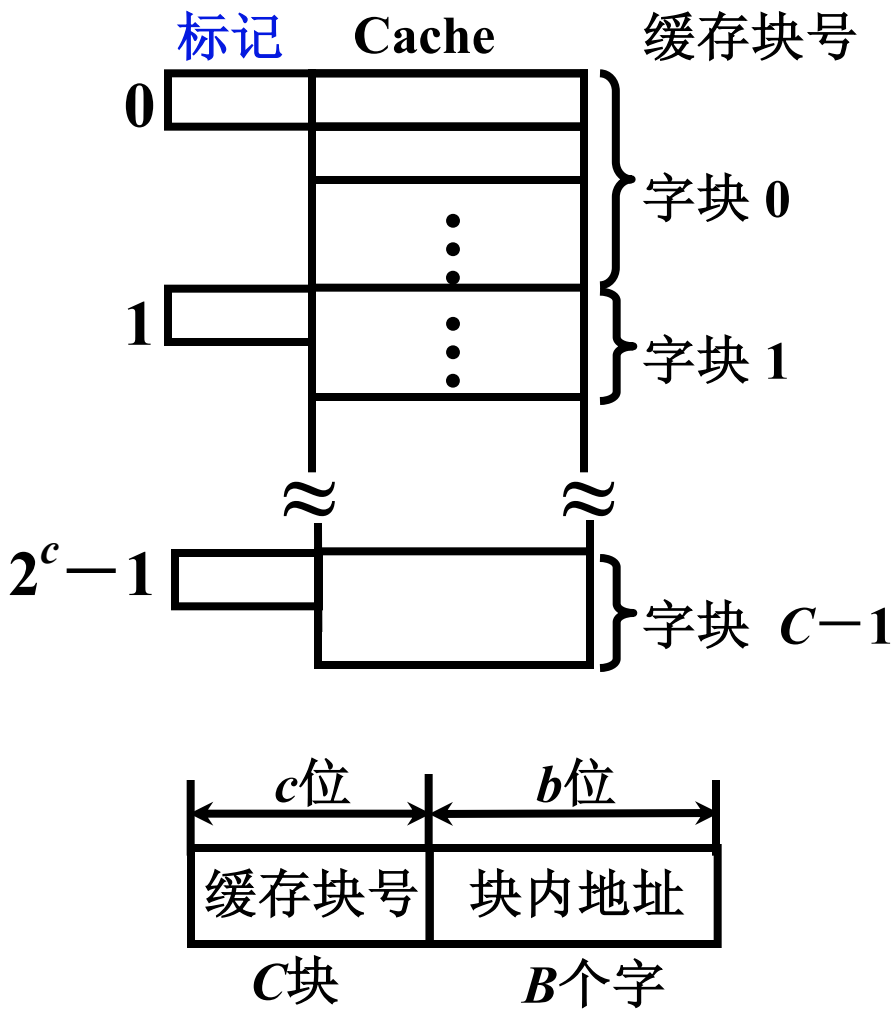
- 对数据的引用
  - 顺序访问数组元素
    (步长为1的引用模式)　　空间局部性
  - 变量sum在每次循环迭代中被引用一次　　时间局部性

- 对指令的引用
  - 顺序读取指令　　空间局部性
  - 重复循环执行for循环体　　时间局部性

2. Cache 的工作原理

(1) 主存和缓存的编址





主存和缓存按块存储　　块的大小相同　*B* 为块长

(2) 命中与未命中

缓存共有 C 块

主存共有 M 块　$M >> C$

命中　　主存块 调入 缓存

主存块与缓存块 建立 了对应关系

用 标记记录 与某缓存块建立了对应关系的 主存块号

未命中　　主存块 未调入 缓存

主存块与缓存块 未建立 对应关系

(3) Cache 的命中率

CPU 欲访问的信息在 Cache 中的 比率

命中率 与 Cache 的 容量 与 块长 有关

一般每块可取 4～8 个字

块长取一个存取周期内从主存调出的信息长度

CRAY_1	16体交叉	块长取 16 个存储字
IBM 370/168	4体交叉	块长取 4 个存储字

（64位×4＝256位）

(3) Cache 的命中率和平均访存时间 4.3

命中率： $H=\dfrac{N_1}{N_1+N_2}$

N1 —— 访问Cache的次数

N2 —— 访问主存的次数

不命中率： $F=1-H$

设 Cache 命中率 为 h，访问 Cache 的时间为 t_c，

访问 主存 的时间为 t_m

平均访存时间 $t_a=h\times t_c+(1-h)\times t_m$

(4) Cache –主存系统的效率

效率 e 与 命中率 有关

$$e = \frac{\text{访问 Cache 的时间}}{\text{平均访问时间}} \times 100\%$$

则 $$e = \frac{t_c}{h \times t_c + (1-h) \times t_m} \times 100\%$$

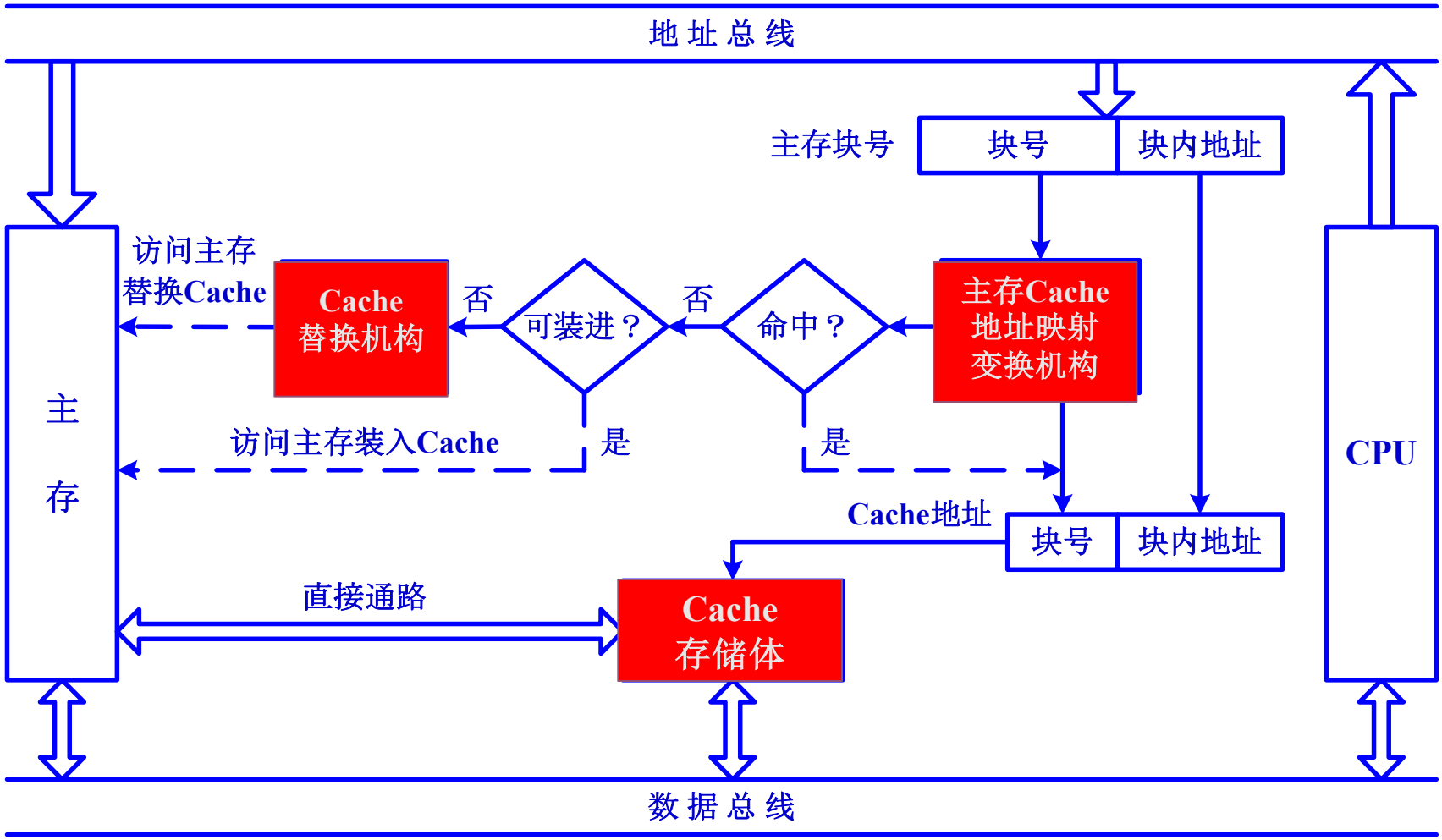
存储层次的四个问题

4.3

1. 当把一个块调入高一层(靠近CPU)存储器时，可以放在哪些位置上?
   (映象规则　调入块可以放在哪些位置)
2. 当所要访问的块在高一层存储器中时，如何找到该块?
   (查找算法 如何在映象规则 规定的候选位置查找)
3. 当发生失效时，应替换哪一块?
   (替换算法　规定的候选位置均被别的块占用)
4. 当进行写访问时，应进行哪些操作?
   (写策略　如何处理写操作)

3. Cache 的基本结构

4.3



二、Cache－主存的地址映射 4.3

1. 直接映射

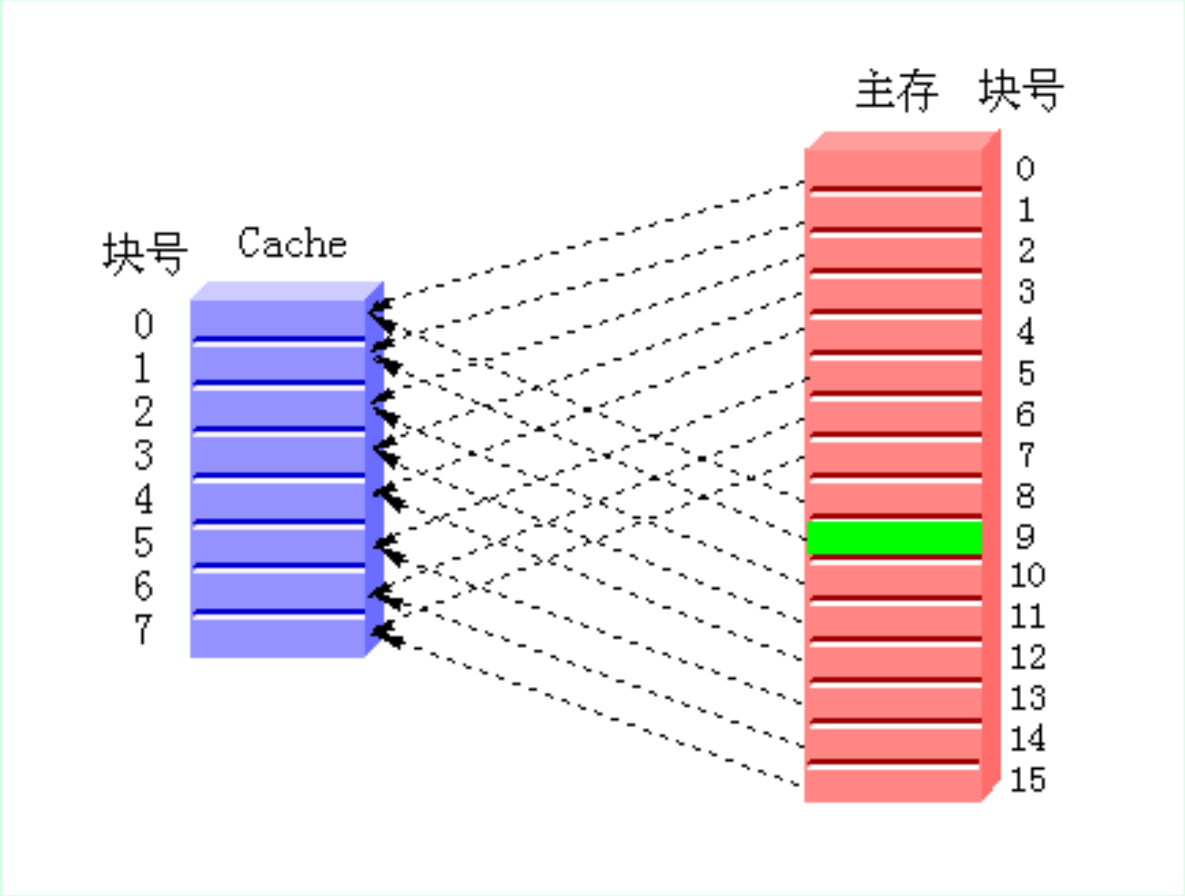
直接映射：主存中的每一块只能被放置到Cache中唯一的一个位置。（循环分配）

对比：阅览室位置 --只有一个位置可以坐

特点：空间利用率最低，冲突概率最高，实现最简单。

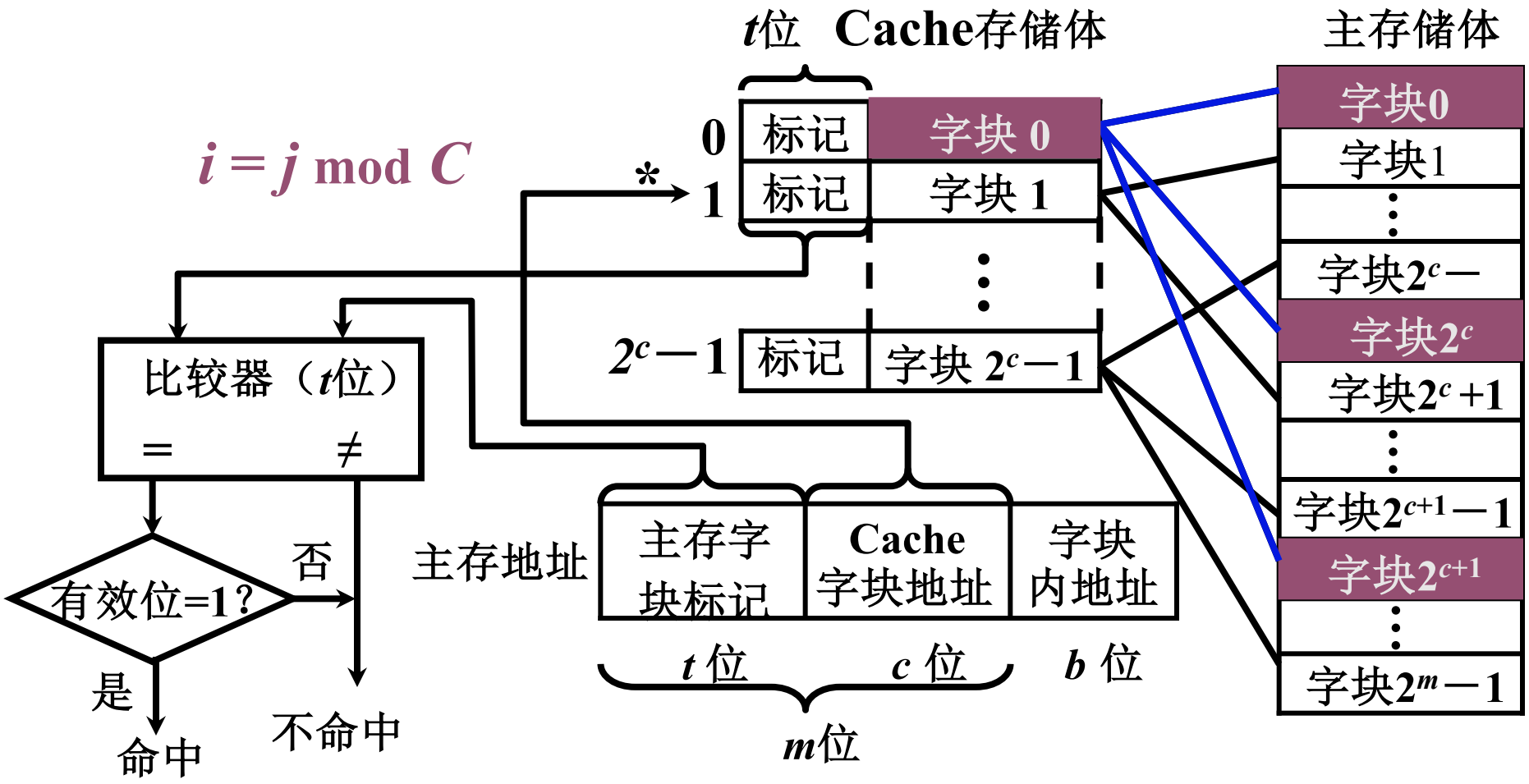
直 接 映 射

（举 例）



1. 直接映射

4.3



每个缓存块 i 可以和 若干 个 主存块 对应

每个主存块 j 只能和 一 个 缓存块 对应

2. 全相联映射

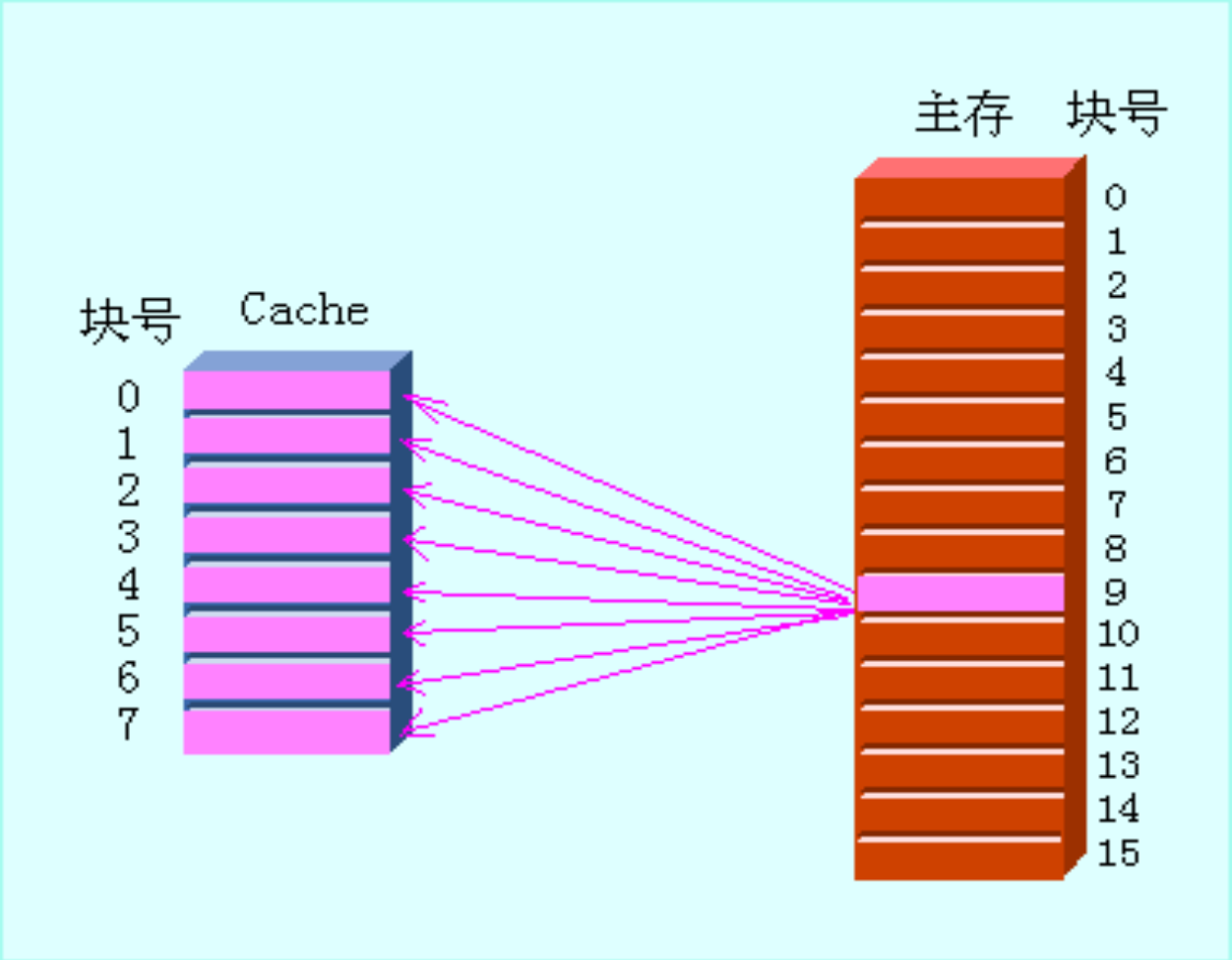
全相联：主存中的任一块可以被放置到Cache中的任意一个位置。

对比：　阅览室位置--随便坐

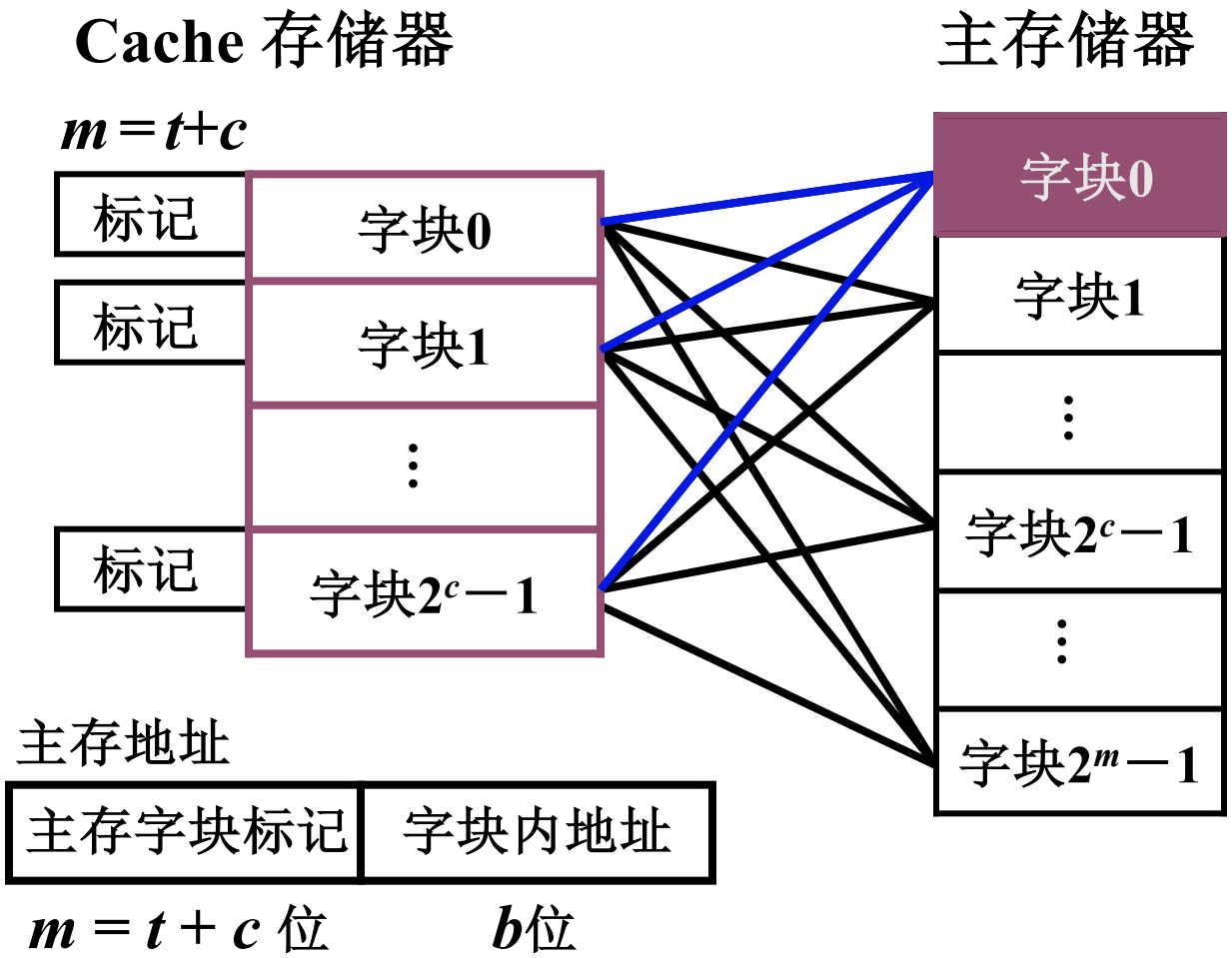
特点：　空间利用率最高，冲突概率最低，实现最复杂。

全相联映射

（举例）



2. 全相联映射

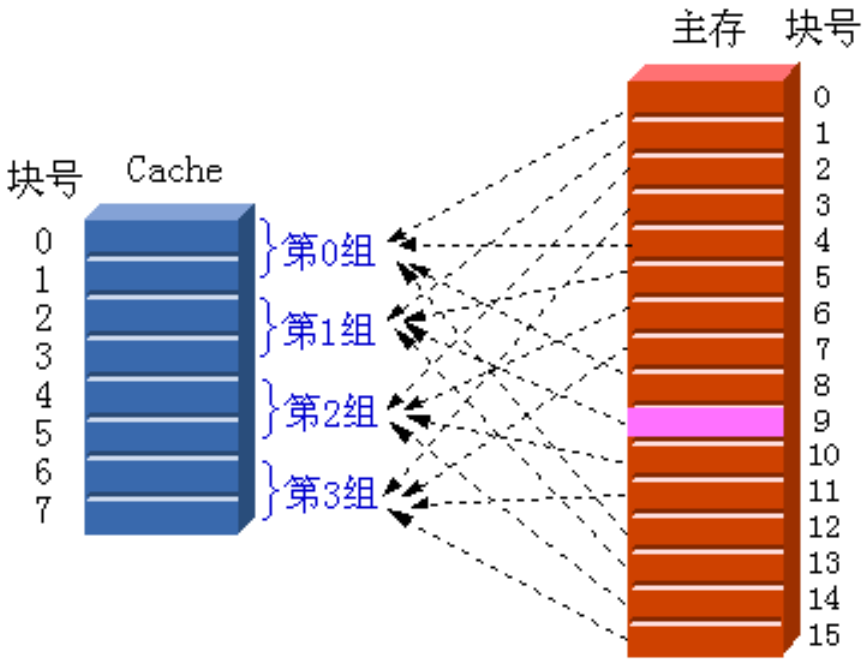


主存 中的 任一块 可以映射到 缓存 中的 任一块

3. 组相联映射

组相联：主存中的每一块可以被放置到Cache中唯一　的一个组中的任何一个位置。

组相联是直接映象和全相联的一种折中



3. 组相联映射

主存储器

组　Cache　共 Q 组，每组内两块（$r=1$）

组	标记	字块 0	标记	字块 1
0				
1	标记	字块 2	标记	字块 3
	⋮	⋮	⋮	⋮
$2^{c-r}-1$	标记	字块 2^c-2	标记	字块 2^c-1

主存地址

主存字块标记	组地址	字块内地址
$s=t+r$ 位	$q=c-r$ 位	b 位
m 位		

主存储器

- 字块0
- 字块1
- ⋮
- 字块$2^{c-r}-1$
- 字块2^{c-r}
- 字块$2^{c-r}+1$
- ⋮
- 字块2^{c-r+1}
- ⋮
- 字块2^m-1

$i=j \bmod Q$

直接映射　全相联映射

某一主存块 j 按模 Q 映射到 缓存 的第 i 组中的 任一块

- *n* 路组相联：每组中有*n* 个块(*n*＝*M*/*G*)，*n* 称为相联度

相联度越高，Cache空间的利用率就越高，块冲突概率就越低，失效率也就越低。

	n (路数)	*G* (组数)
全相联	*M*	1
直接映象	1	*M*
组相联	*1＜n＜M*	*1＜G＜M*

- 大多数计算机的Cache: *n* ≤4

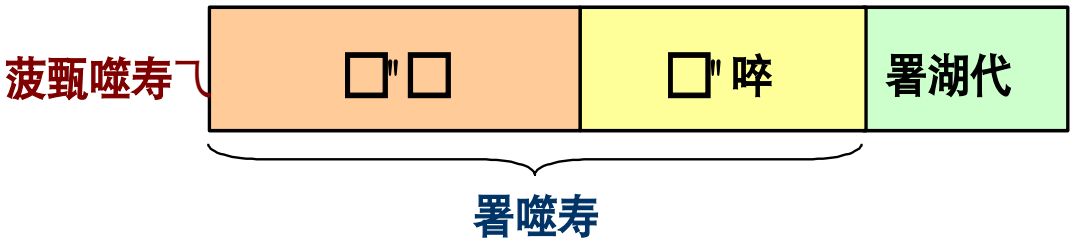
想一想：相联度是否越大越好？

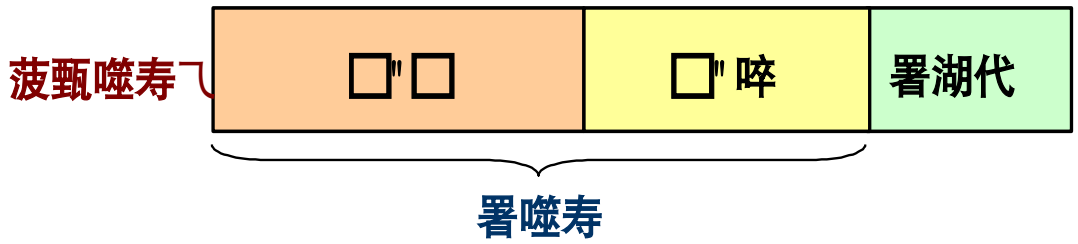
四、查找方法

4.3

- 当CPU访问Cache时，如何确定Cache中是否有所要访问的块？
- 若有的话，如何确定其位置？

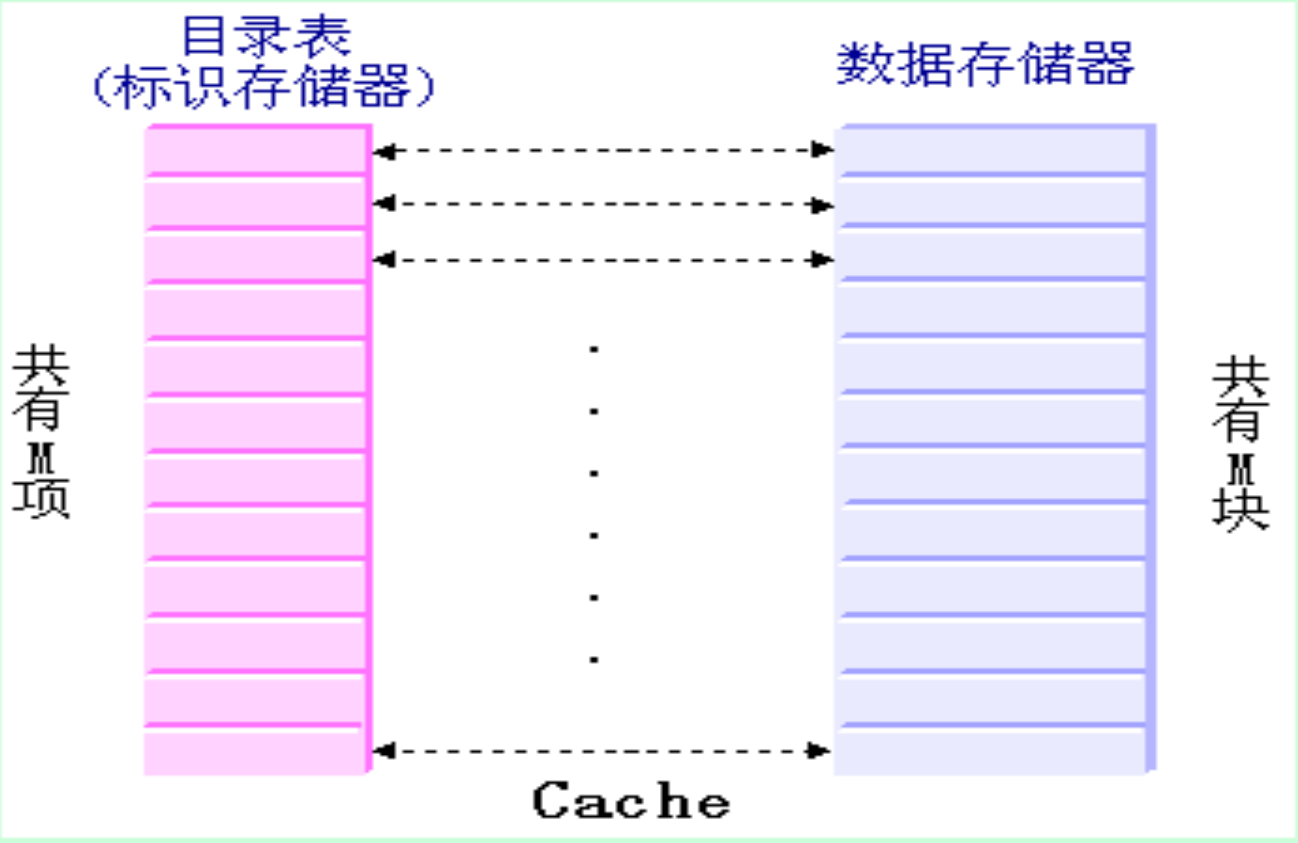
- 通过查找目录表来实现
  - 目录表的结构
    - 主存块的块地址的高位部分，称为标识。
    - 每个主存块能唯一地由其标识来确定
    - 每一项有一个有效位，指出Cache中的块是否有效
  - 只需查找候选位置所对应的目录表项



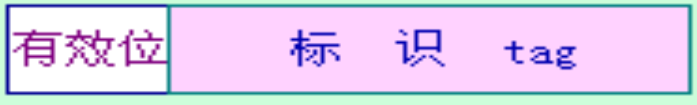


- 块内地址偏移用来从块中选出需要的数据
- 索引域用来选择组
- 通过比较标识域来判断是否发生命中
- 块内偏移是没有必要比较的，这是因为Cache命中与否的单位是整个块
- 由于索引域是用来选择被检查的组， 所以对索引域的比较是多余的

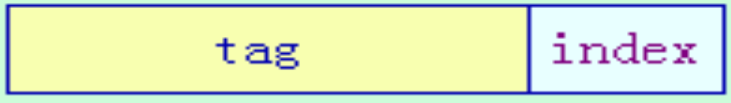
Cache目录表的结构



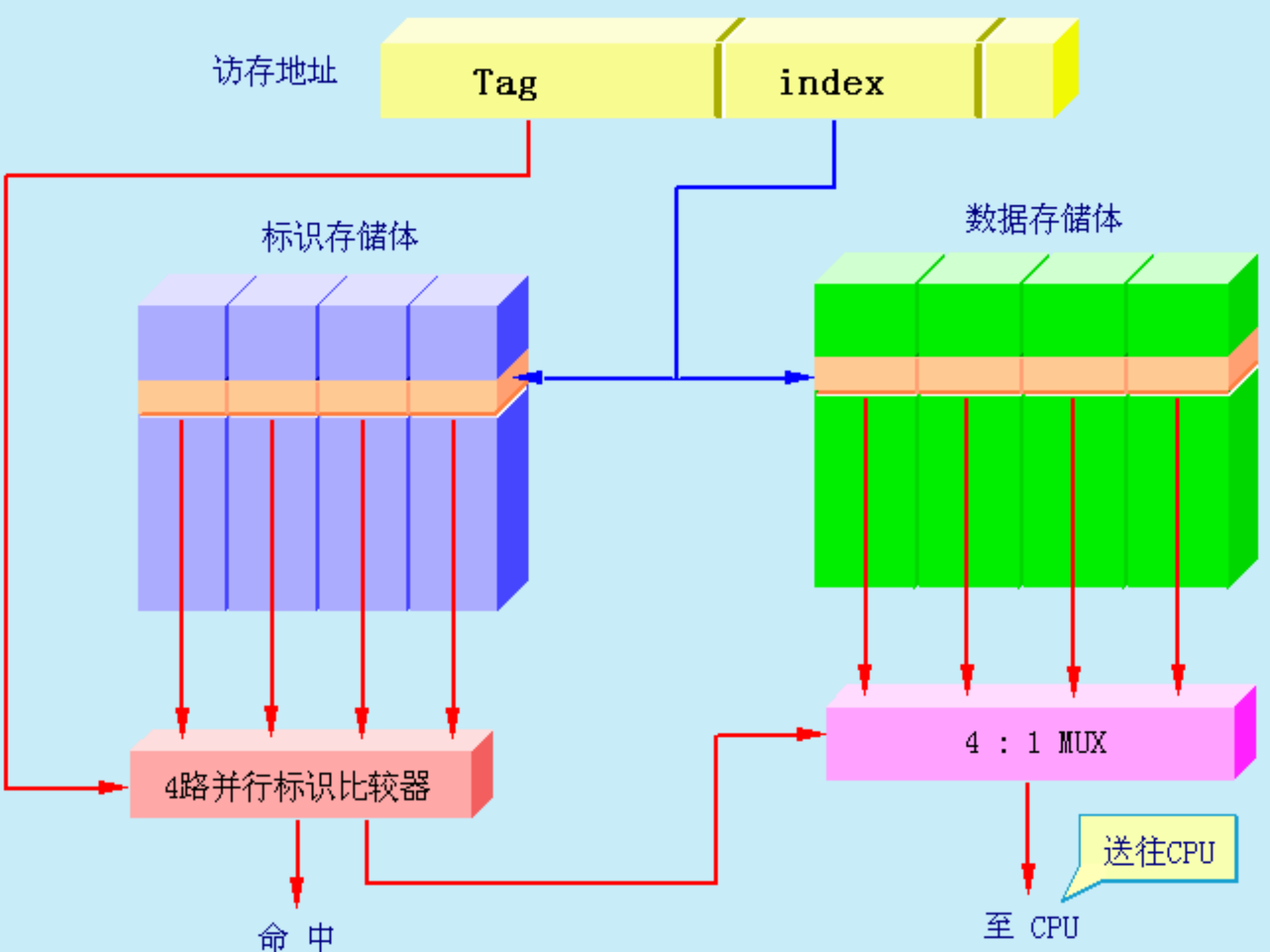
目录表项：

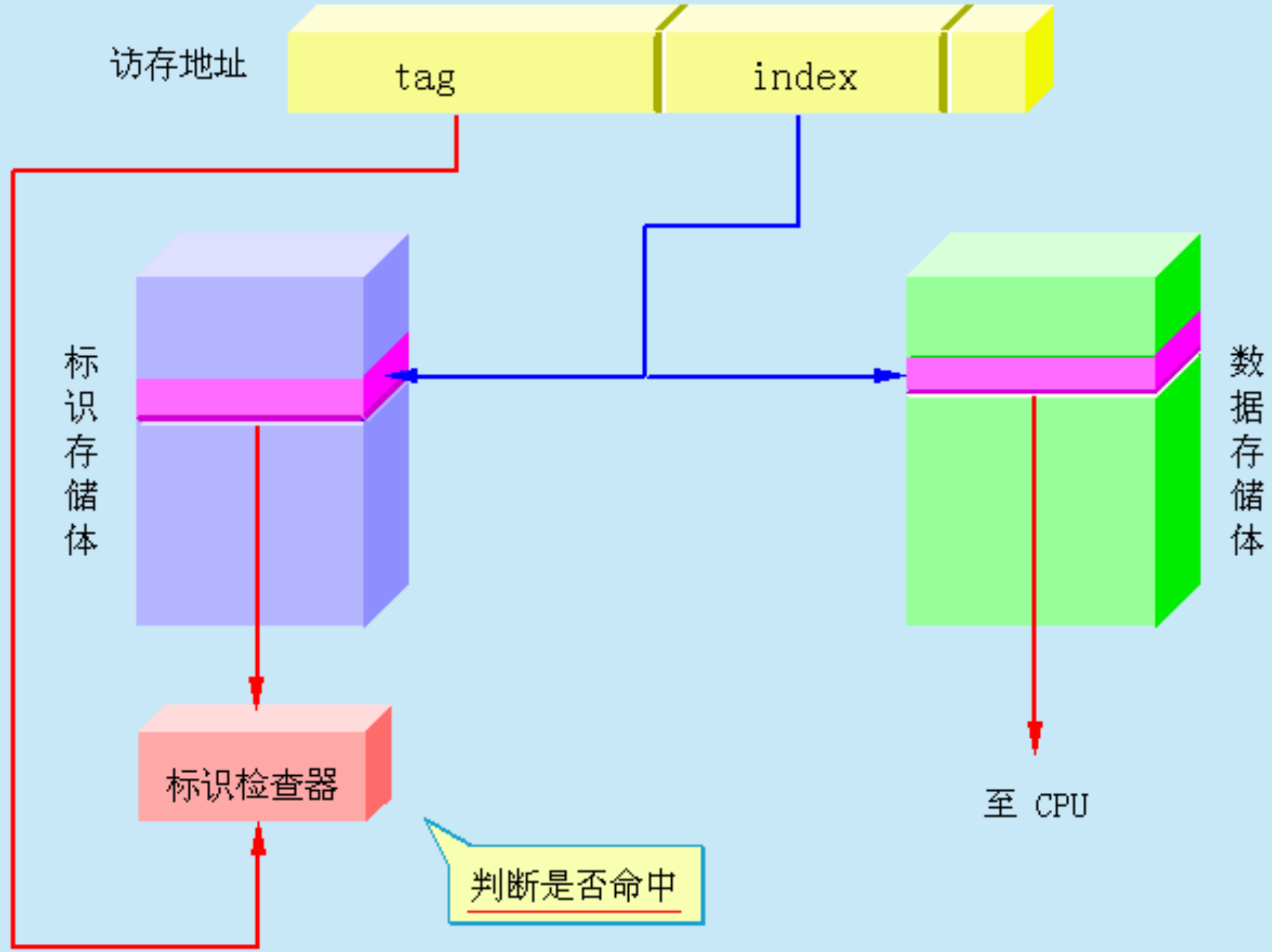


访存地址：



- 4 路组相联Cache的查找过程
  - 优缺点
    - 不必采用相联存储器，而是用按地址访问的存储器来实现。
    - 当相联度n增加时，不仅比较器的个数会增加，而且比较器的位数也会增加。
- 直接映象Cache的查找过程


访存地址
Tag
index
标识存储体
数据存储体
4路并行标识比较器
4 : 1 MUX
送往CPU
命 中
至 CPU


访存地址
tag
index
标识存储体
数据存储体
标识检查器
至 CPU
判断是否命中

Intel Itanium2（版图布局）