计算机组成与系统结构

第三章 多层次的存储器 (4)

吕昕晨

lvxinchen@bupt.edu.cn

网络空间安全学院

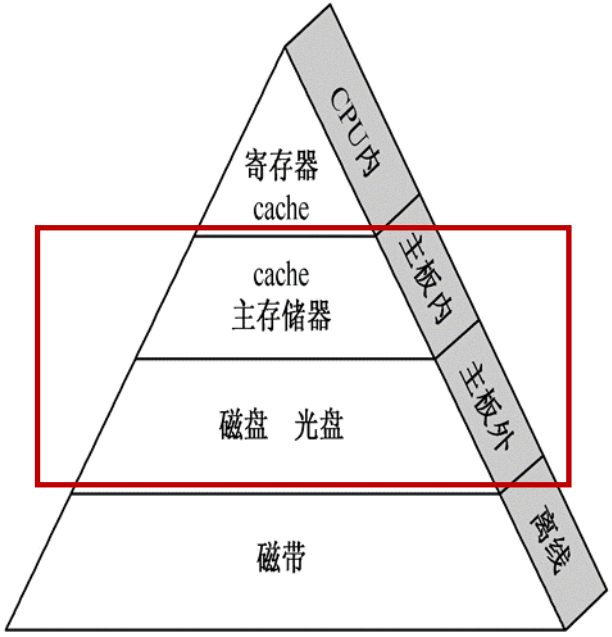
多层次的存储器

第三章 多层次的存储器

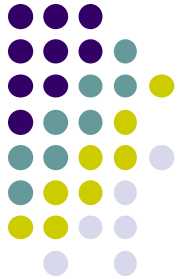
- 虚拟存储器
  - 虚存地址映射方式：页式/段式/段页式
  - 虚存替换算法
- 并行存储器
  - 双端口存储器（空间并行）
  - 多体交叉存储器（时间并行/流水线）

虚拟存储整体目标

- 问题：主存与辅存速度/容量不匹配
- 实现方法：由硬件+操作系统调度
- 出发点相同
  - 为了提高存储系统的性能价格比而构造的分层存储体系
- 原理相同
  - 均利用了程序运行时的局部性原理
  - 把最近常用的信息块从相对慢速而大容量的存储器调入相对高速而小容量的存储器



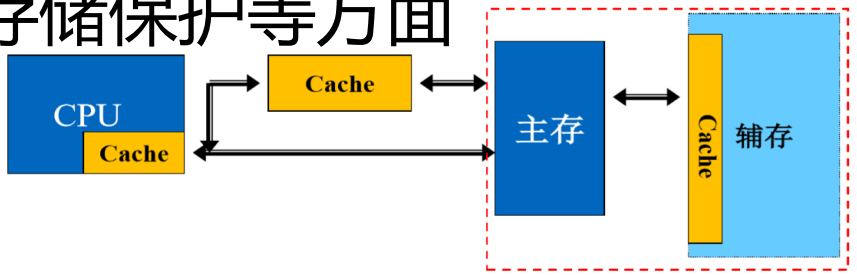
虚拟存储器与Cache对比—相似



- 主存辅存的访问机制与Cache—主存的访问机制类似
  - **功能：**由Cache存储器、主存和辅存构成的三级存储体系中的两个层次
  - **机制：**Cache和主存之间以及主存和辅存之间分别有**辅助硬件和辅助软硬件**负责地址变换与管理，以便各级存储器能够组成有机的三级存储体系
  - **效果：**Cache和主存构成了系统的内存，而主存和辅存依靠辅助软硬件的支持构成了虚拟存储器

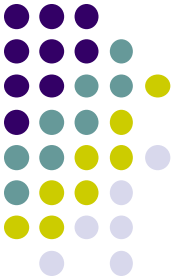
虚拟存储器与Cache区别（1）

- **侧重点**不同
  - Cache主要解决主存与CPU的**速度差异**问题
  - 虚存主要是解决**（主存）存储容量**问题，另外还包括存储管理、主存分配和存储保护等方面



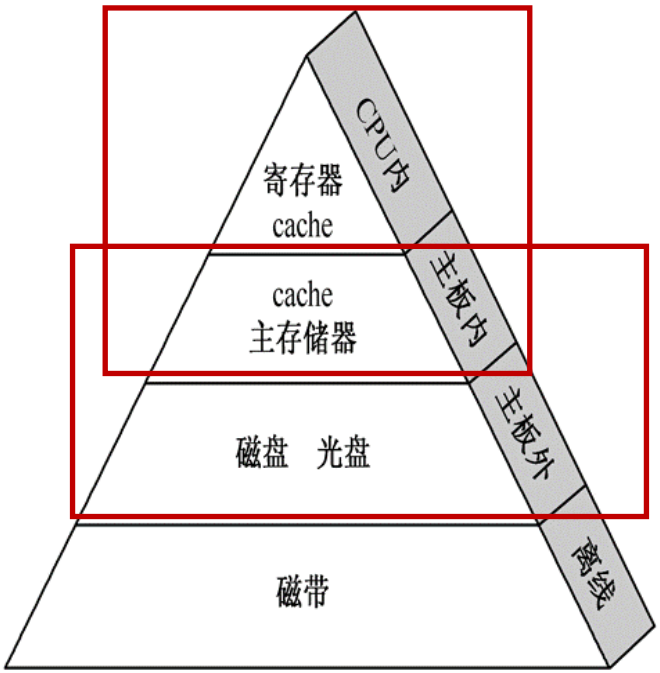
- **数据通路**不同
  - CPU与Cache和主存之间均有直接访问通路，Cache不命中时可直接访问主存
  - 虚存所依赖的辅存与CPU之间不存在直接的数据通路，**CPU最终还是要访问主存**

虚拟存储器与Cache区别（2）

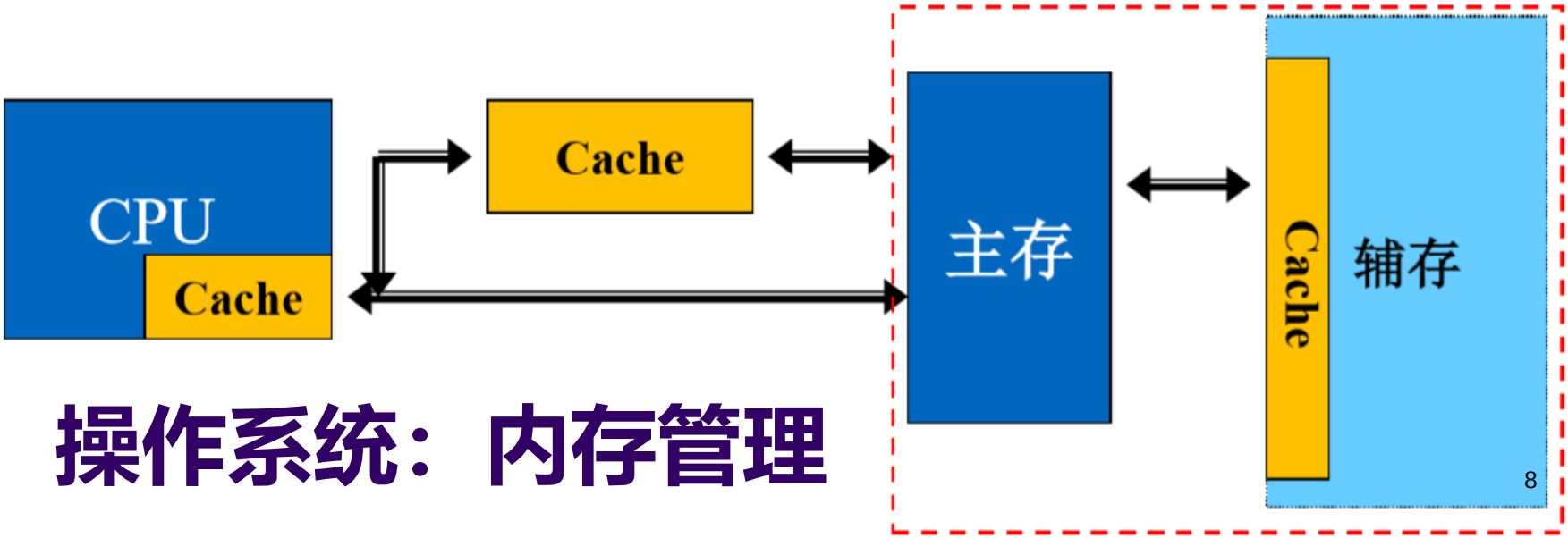


- **透明性**不同
  - Cache的管理完全由**硬件**完成，对系统程序员和应用程序员均透明
  - 虚存管理由**软件（操作系统）和硬件共同完成**，虚存对实现存储管理的系统程序员不透明，而只对应用程序员透明
- 未命中时的**损失不同**
  - 由于主存的存取时间是Cache的存取时间的10倍，而主存的存取速度通常比辅存的存取速度快上千倍（机械）
  - 主存未命中时系统的性能损失要远大于Cache未命中时的损失

多级存储结构—原理

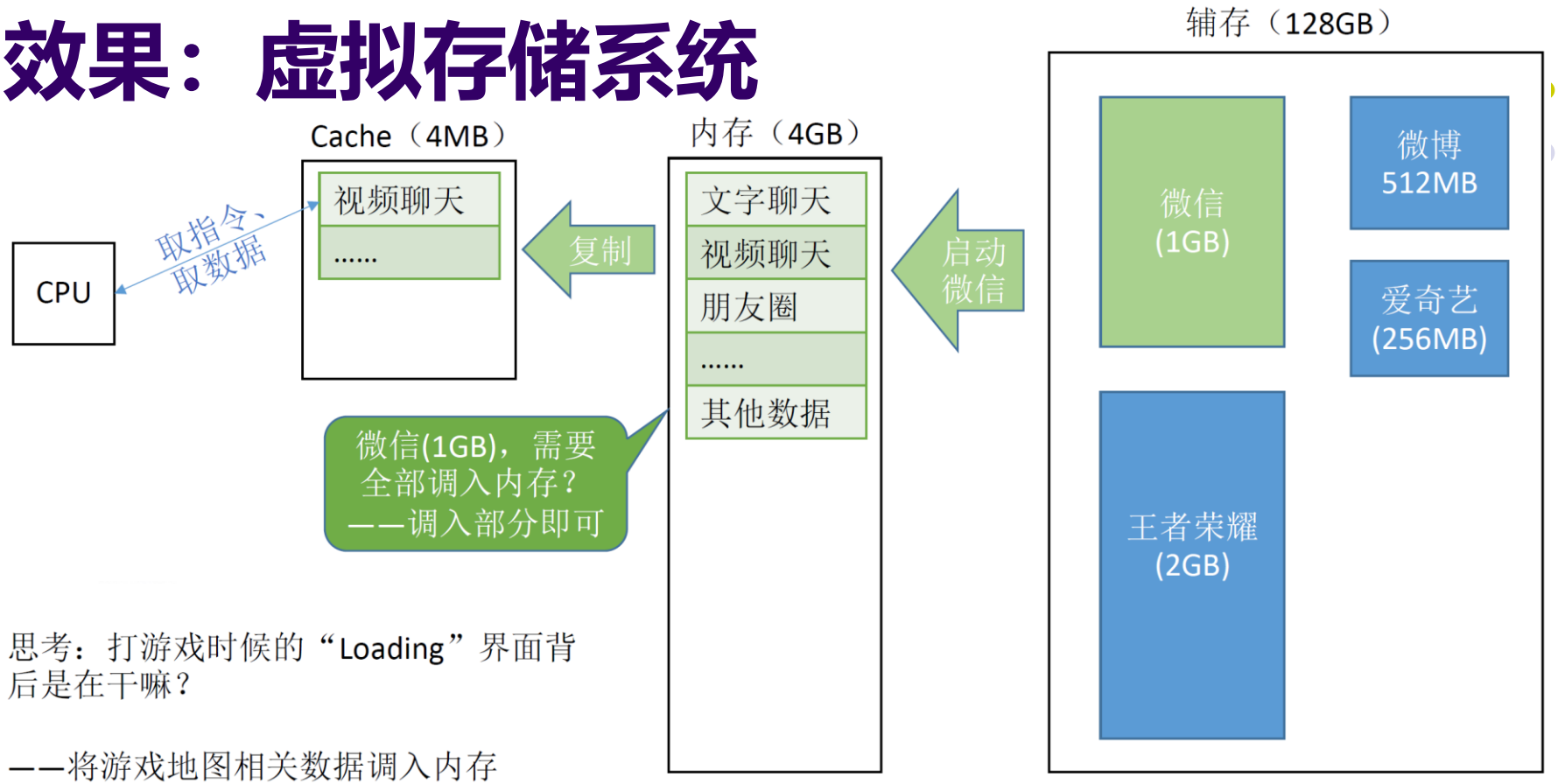


Cache-主存-辅存
多级存储结构



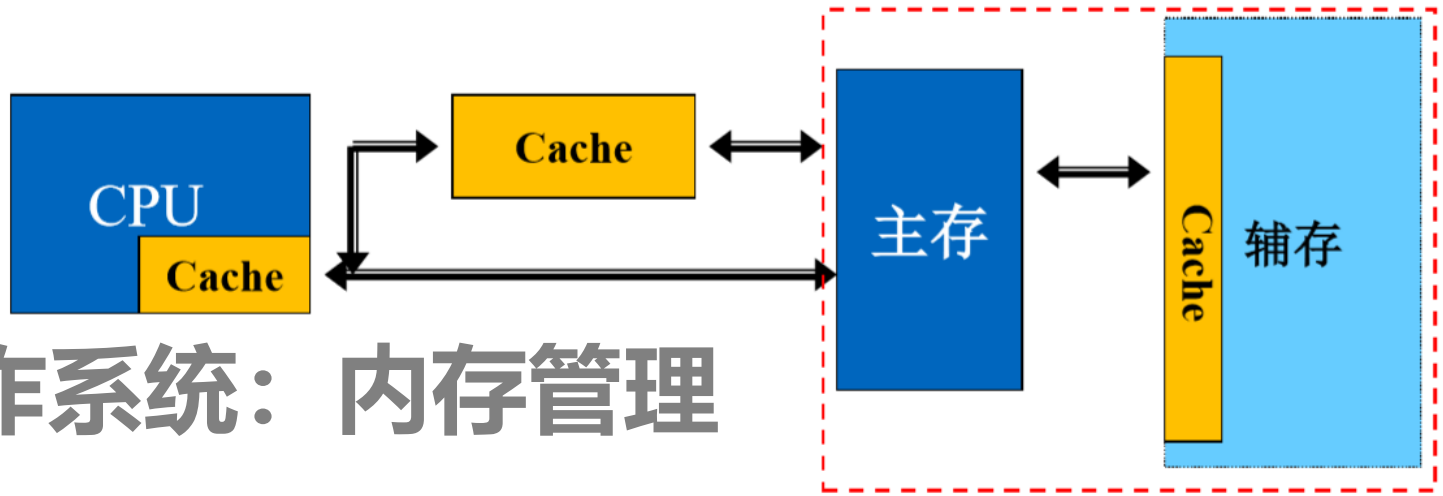
操作系统：内存管理

效果：虚拟存储系统



思考：打游戏时候的“Loading”界面背后是在干嘛？

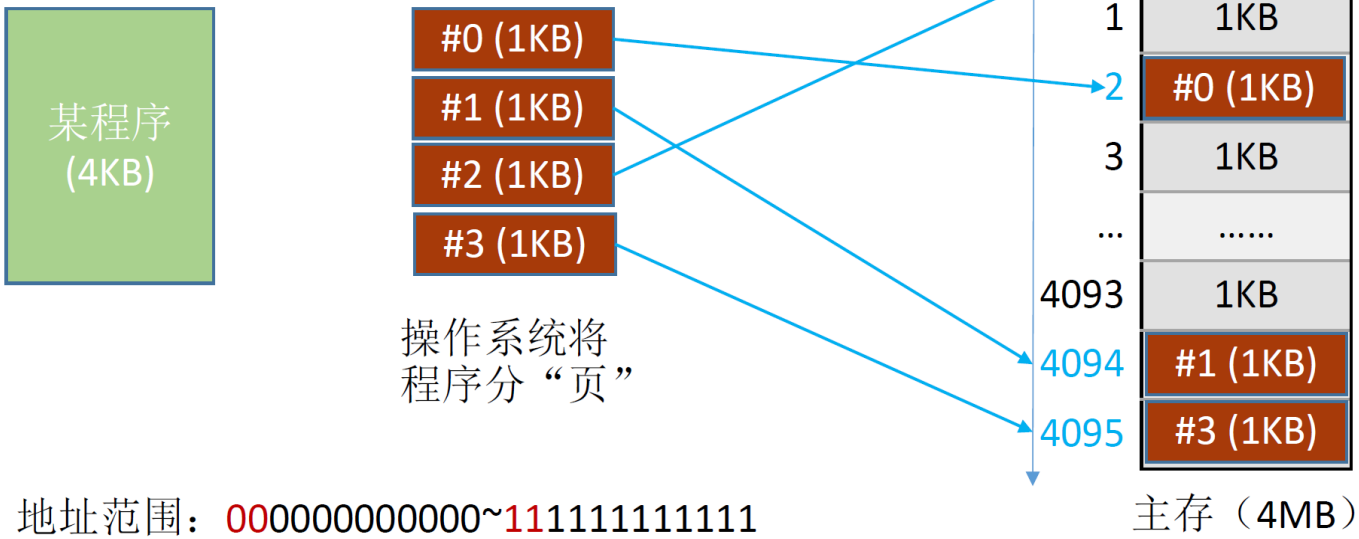
——将游戏地图相关数据调入内存



操作系统：内存管理

- 逻辑地址（虚地址）：程序员（CPU）眼中的地址
- 物理地址（实地址）：实际主存地址

实地址 v.s. 虚地址



程序员视角：
整个程序共 4KB=2^{12}B，地址范围：000000000000~111111111111

变量 x 的逻辑地址： 001000000011
变量 y 的逻辑地址： 110000001010

逻辑页号	页内地址
若干位	10位

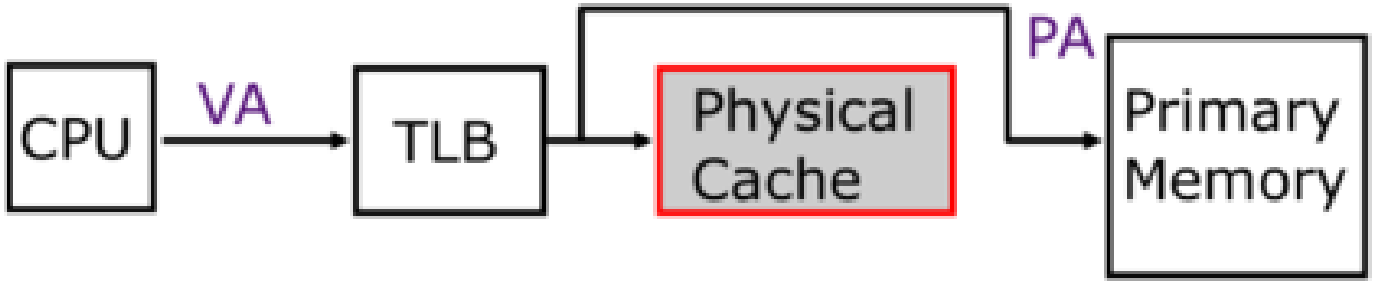
主存的物理地址共22位：

主存块号	块内地址
12位	10位

- 问题：虚地址与实地址映射与管理
- 操作系统（MMU）：常用方法-段页式管理方法

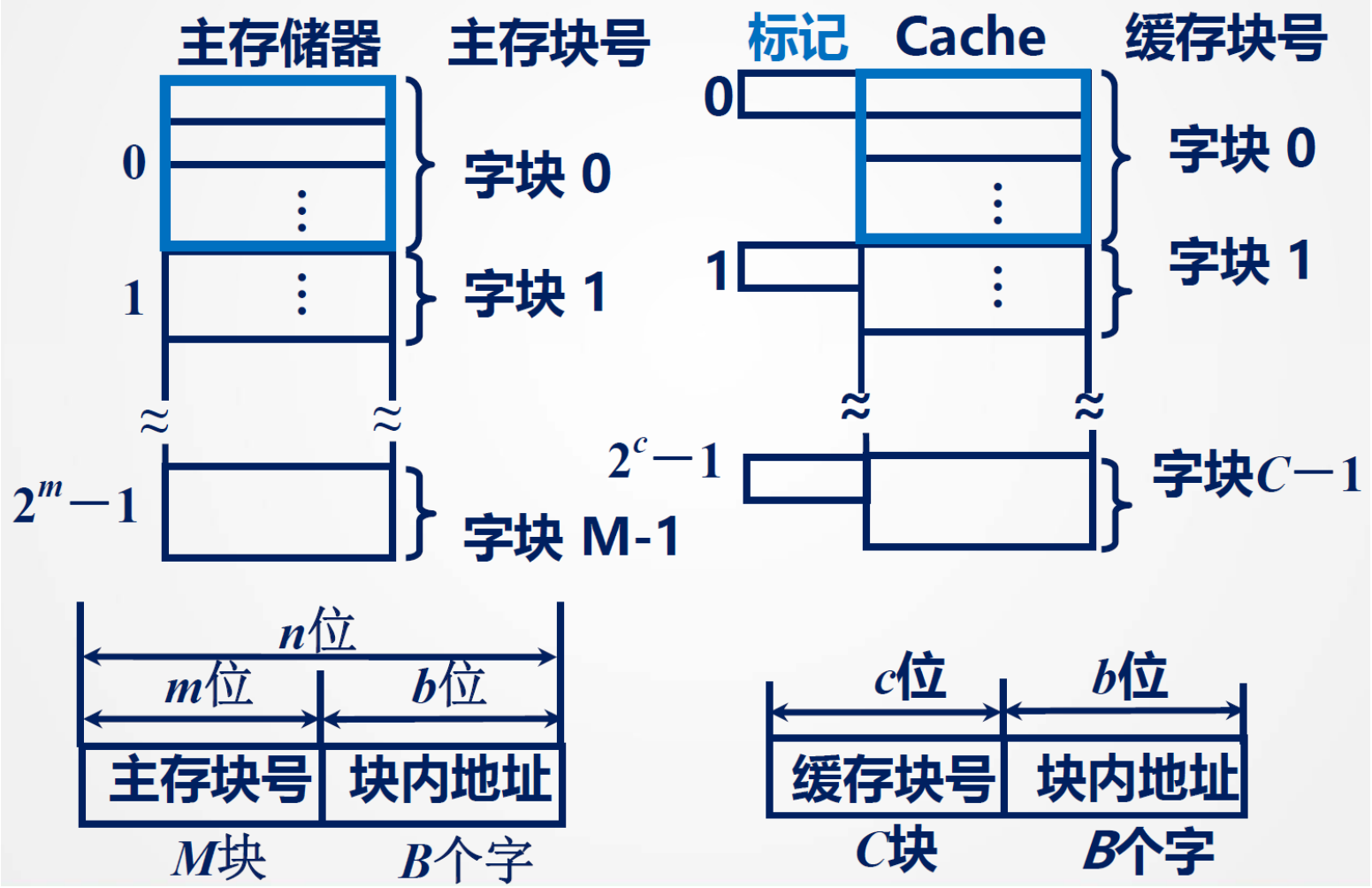
虚地址→实地址映射

- 访存流程——多级系统
  - ① CPU产生虚拟地址
  - ② TLB翻译成物理地址（如果命中）
  - ③ 将物理地址送入Cache（如果命中）
  - ④ Cache 返回数据



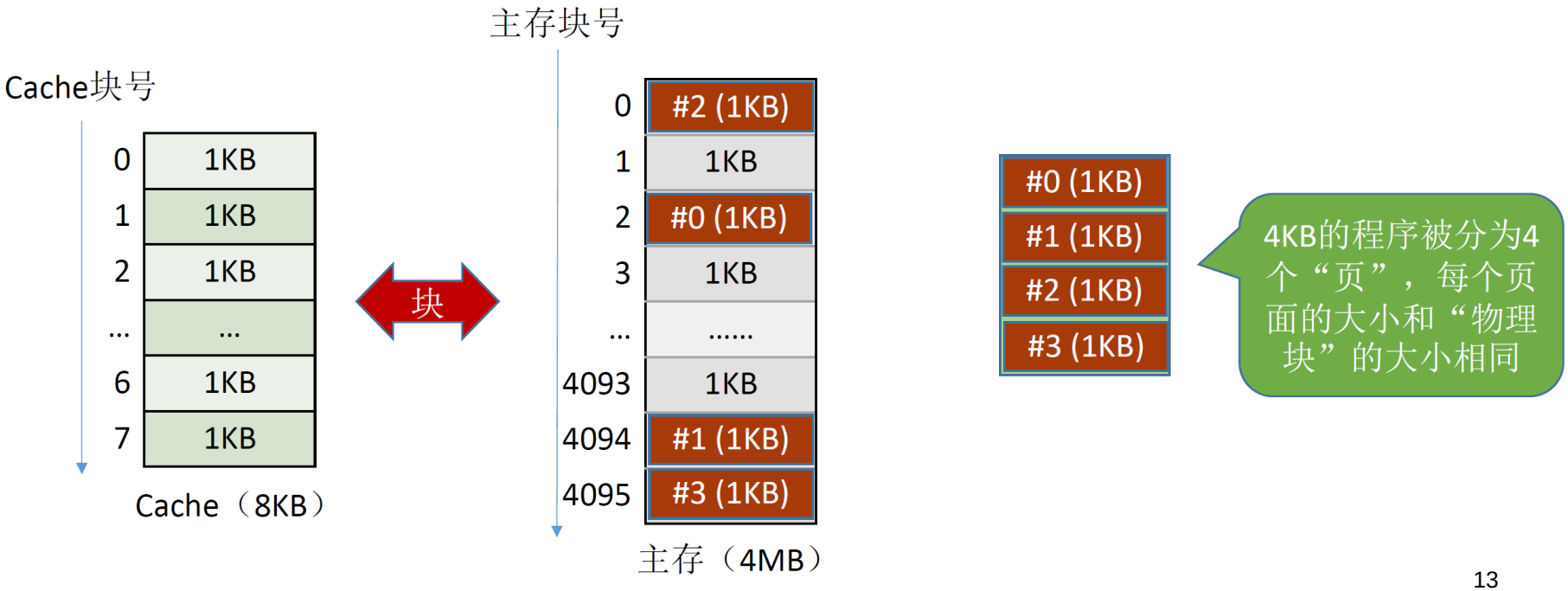
- **虚存组织方式**：页式、段式、段页式

回顾：Cache—主存地址管理



页式虚拟存储：特点

- 虚地址空间被分成等长的页（逻辑页）
- 主存空间也被分成同样大小的页（物理页）
- 优点：方便管理



页式虚拟存储：地址映射原理

- 页表：记录逻辑页与主存块的映射关系（查表）

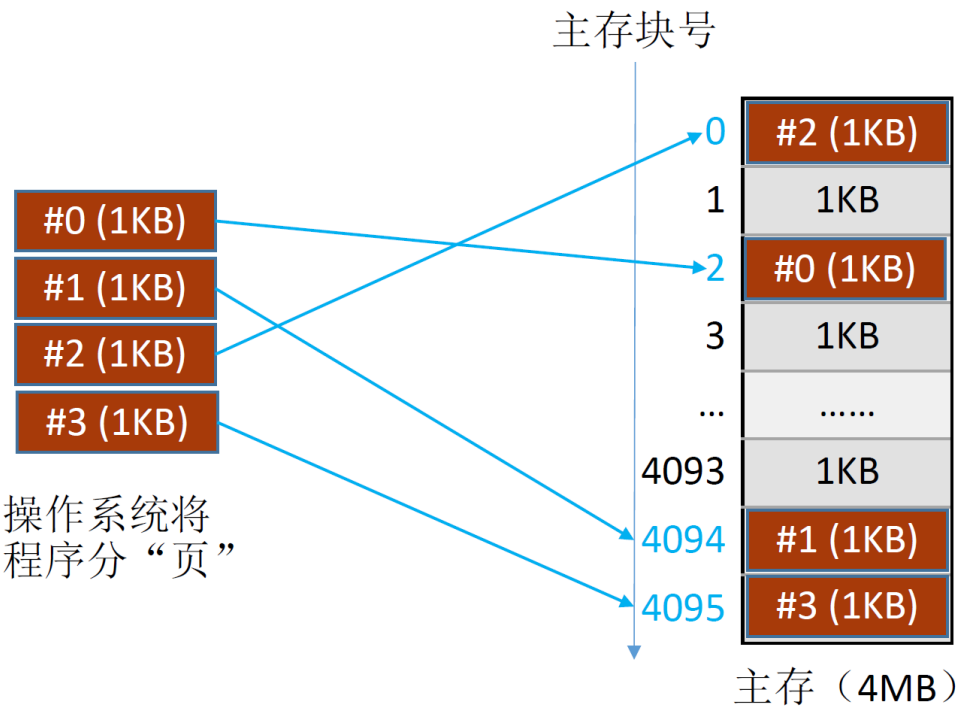
取变量 x 至ACC寄存器
机器指令： 000001 001000000011
操作码+地址码（使用逻辑地址）

逻辑页号	主存块号
#0	2
#1	4094
#2	0
#3	4095

页表项

页表数据在主存里

页表



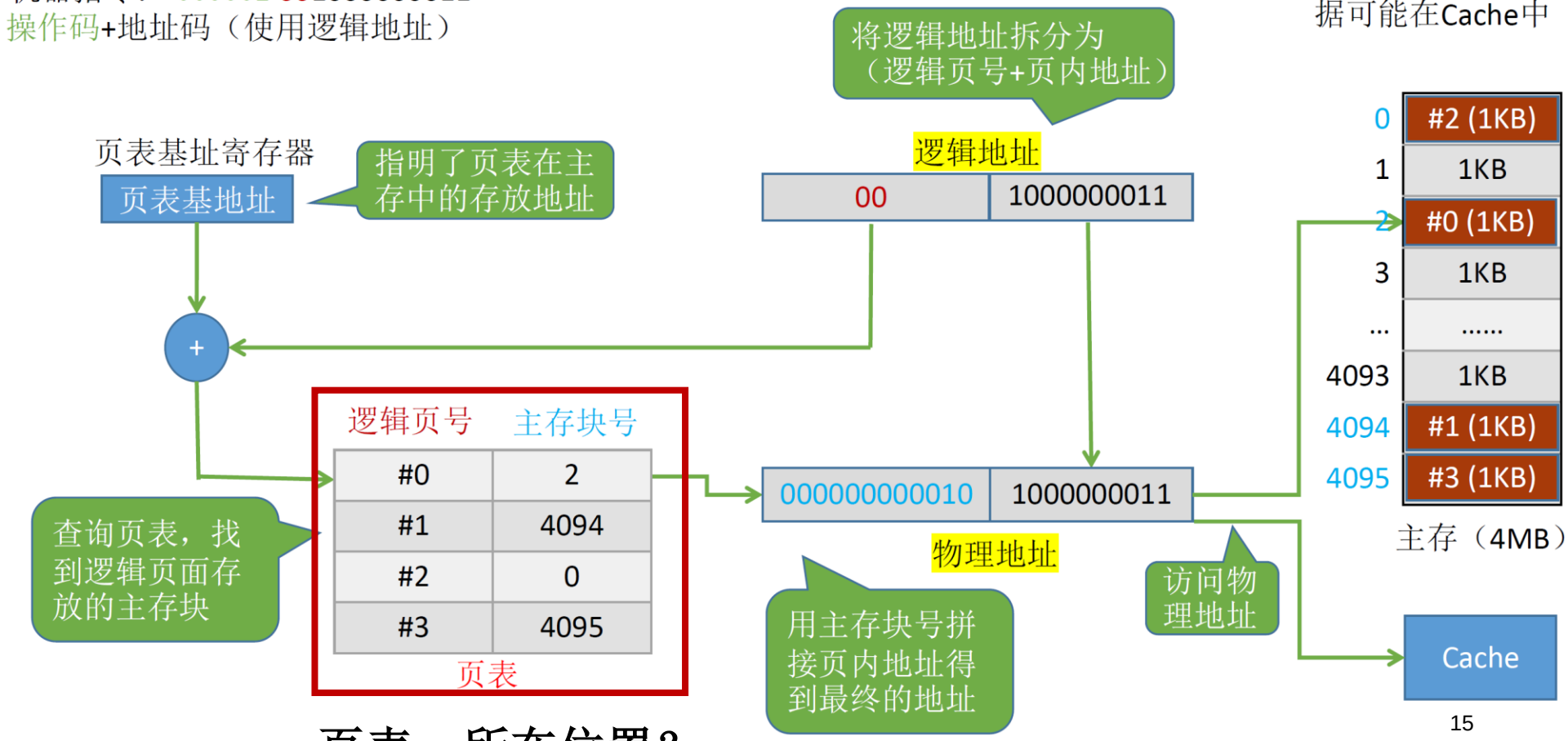
变量 x 的逻辑地址： 001000000011

变量 x 的物理地址： 000000000010000000011

页式虚拟存储：地址映射流程

取变量 x 至ACC寄存器
机器指令： 000001 001000000011
操作码+地址码（使用逻辑地址）

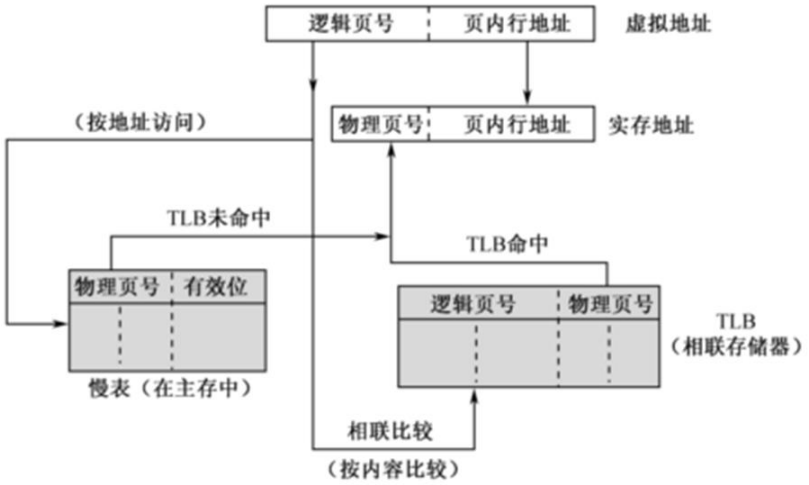
CPU访问某个主存地址：这个块的数据可能在Cache中



页表：所在位置？

转换后援缓冲器 (TLB)

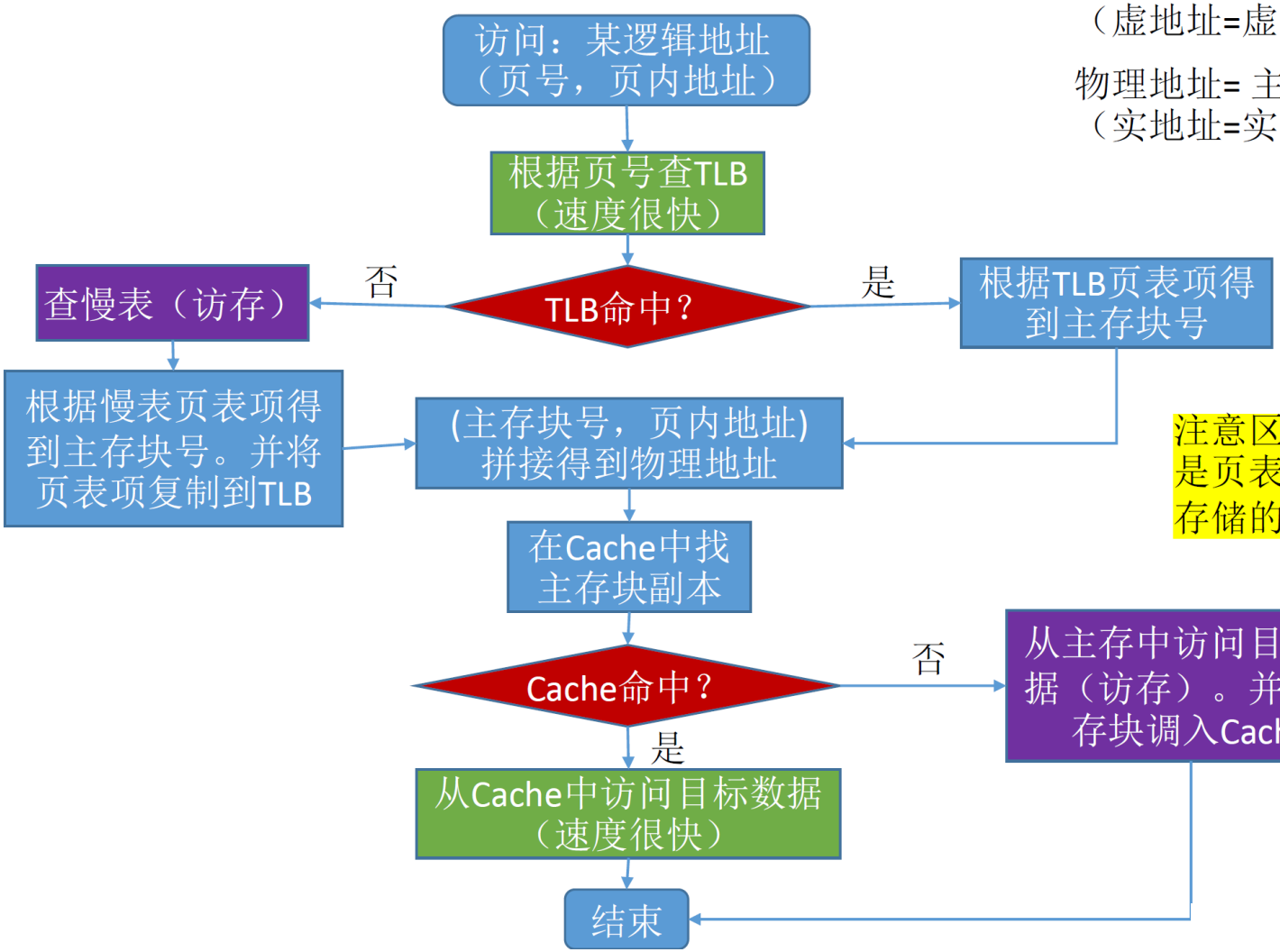
- 页表通常在主存中
  - 即使逻辑页已经在主存中，也至少要访问两次物理存储器才能实现一次访存（查找页表+物理地址）
  - 虚拟存储器的存取时间加倍
- 转换后援缓冲器（TLB）
  - 页表中的最活跃的部分存放在Cache（多级存储）中，组成快表
  - 保存在主存中的完整页表则称为慢表



页式地址转换流程总结

逻辑地址= 逻辑页号+页内地址
（虚地址=虚页号+页内地址）

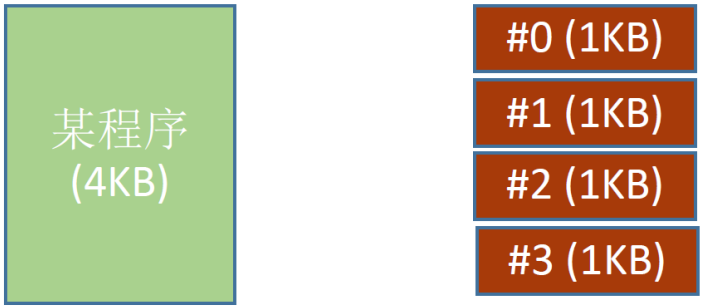
物理地址= 主存块号+页内地址
（实地址=实页号+页内地址）



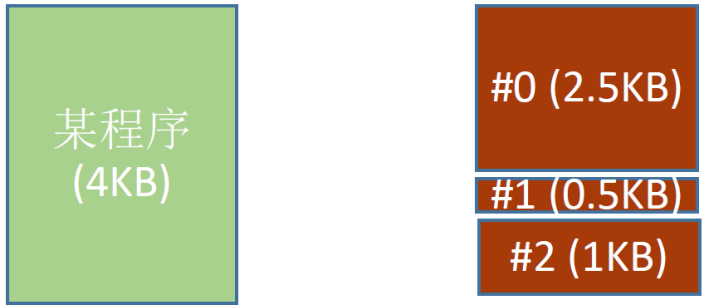
注意区别：快表中存储的是页表项的副本；Cache中存储的是主存块的副本

段式虚拟存储 v.s. 页式虚拟存储

- 段：划分的**长度可以动态改变**的区域
  - 子程序、操作数和常数等不同类型的数据划分到不同的段中
  - 按程序功能划分
- 段式虚拟存储映射方式
  - 段表：段号→实际物理地址



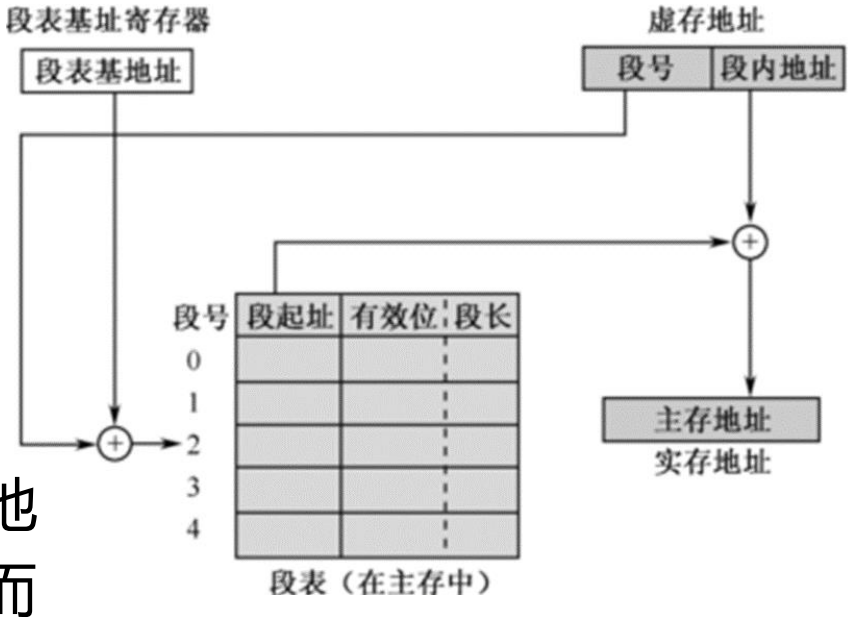
页式虚拟存储器——拆分成大小相等的页面



段式虚拟存储器——按照功能模块拆分
如：#0 段是自己的代码，#1 段是库函数代码，#2段是变量

段式虚拟存储器：段表

- 表项至少包含下面三个字段：
  - 有效位：指明该段是否已经调入实存
  - 段起址：指明在该段已经调入实存的情况下，该段在实存中的首地址
  - 段长：记录该段的实际长度
- 保证访问某段的地址空间时，段内地址不会超出该段长度导致地址越界而破坏其他段
- 段表本身也是一个段，一般是驻留在主存中



段式虚拟存储器的特点

- 优点
  - 段的逻辑独立性（安全性问题）使其易于编译、管理、修改和保护，也便于多道程序共享
  - 段长可以根据需要动态改变，允许自由调度，以便有效利用主存空间
- 缺点
  - 容易在段间留下许多外碎片，造成存储空间利用率降低
  - 由于段长不一定是2的整数次幂，因而不能简单地像分页方式那样用虚地址和实地址的最低若干二进制位作为段内偏移量，并与段号进行直接拼接
  - 必须用加法操作通过段起址与段内偏移量的求和得到物理地址，需要更多的硬件支持

段页式：段式+页式虚拟存储

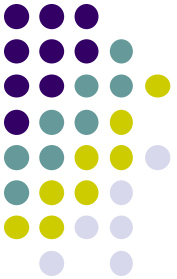
- 段页式虚拟存储器
  - 段式虚拟存储器和页式虚拟存储器的结合
  - 段号+页号+偏移量（基号—多任务操作系统）
- 实存被等分成页
  - 每个程序则先按逻辑结构分段，每段再按照实存的页大小分页
  - 程序按页进行调入和调出操作，但可按段进行编程、保护和共享

（基号 N）	段号 S	段内逻辑页号 P	页内地址偏移量 D

第三章 多层次的存储器

- 虚拟存储器
  - 虚存地址映射方式：页式/段式/段页式
  - 虚存替换算法
- 并行存储器
  - 双端口存储器（空间并行）
  - 多体交叉存储器（时间并行/流水线）

虚存的替换算法



- 当从辅存调页至主存而主存已满时，虚存替换
- 虚存替换与Cache替换算法类似
  - FIFO算法、LRU算法、LFU算法
- 不同点
  - Cache的替换全部靠硬件实现，而虚拟存储器的替换有操作系统的支持
  - 虚存缺页对系统性能的影响比Cache未命中要大得多，因为调页需要访问辅存，并且要进行任务切换
  - 虚存页面替换的选择余地很大，属于一个进程的页面都可替换

虚存的替换算法例题

假设主存只允许存放a、b、c三个页面，逻辑上构成a进c出的FIFO队列。某次操作中进程访存的序列是0,1,2,4,2,3,0,2,1,3,2（虚页号）。若分别采用FIFO算法、FIFO+LRU算法，请用列表法分别求两种替换策略情况下**主存的命中率**。

页表	0	1	2	4	2	3	0	2	1	3	2
a											
b											
c											
替换											
命中											

虚存的替换例题—FIFO

假设主存只允许存放a、b、c三个页面，逻辑上构成a进c出的FIFO队列。某次操作中进程访存的序列是0,1,2,4,2,3,0,2,1,3,2（虚页号）。若分别采用FIFO算法、FIFO+LRU算法，请用列表法分别求两种替换策略情况下主存的命中率。

页表	0	1	2	4	2	3	0	2	1	3	2
a	0	1	2	4	4	3	0	2	1	3	3
b		0	1	2	2	4	3	0	2	1	1
c			0	1	1	2	4	3	0	2	2
替换				替换		替换	替换	替换	替换	替换	
命中					命中						命中

命中率=2/11=18.2%

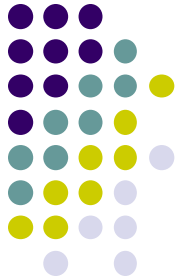
虚存的替换例题—FIFO+LRU

假设主存只允许存放a、b、c三个页面，逻辑上构成a进c出的FIFO队列。某次操作中进程访存的序列是0,1,2,4,2,3,0,2,1,3,2（虚页号）。若分别采用FIFO算法、FIFO+LRU算法，请用列表法分别求两种替换策略情况下主存的命中率。

页表	0	1	2	4	2	3	0	2	1	3	2
a	0	1	2	4	2	3	0	2	1	3	2
b		0	1	2	4	2	3	0	2	1	3
c			0	1	1	4	2	3	0	2	1
替换				替换		替换	替换		替换	替换	
命中					命中			命中			命中

命中率=3/11=27.3%

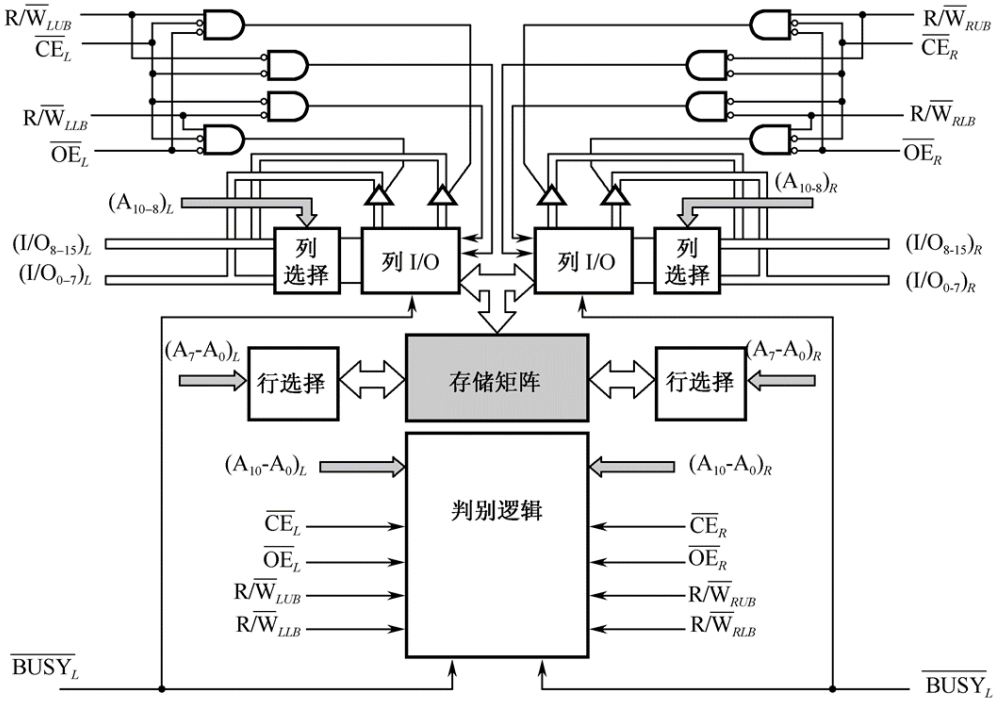
第三章 多层次的存储器



- 虚拟存储器
  - 虚存地址映射方式：页式/段式/段页式
  - 虚存替换算法
- 并行存储器
  - 双端口存储器（空间并行）
  - 多体交叉存储器（时间并行/流水线）

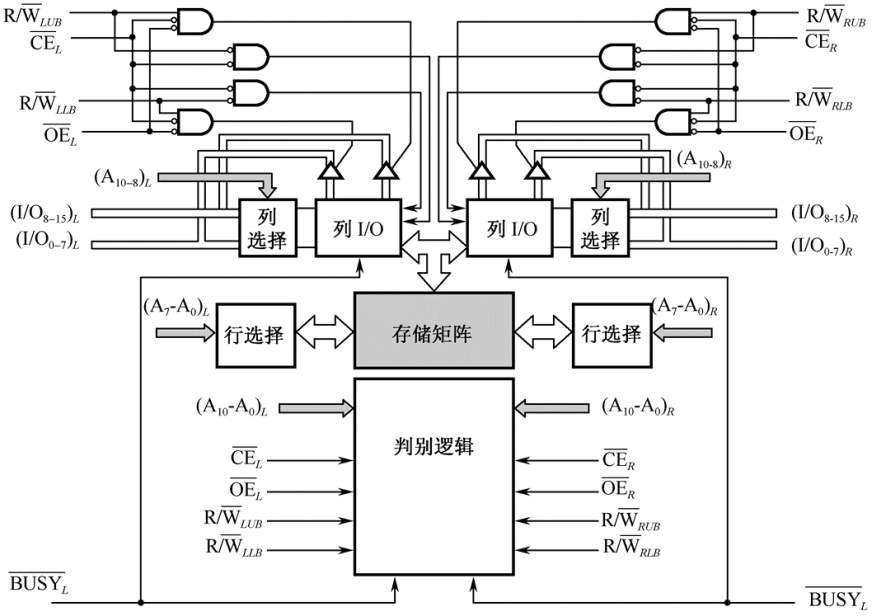
双端口存储器逻辑框图

- 双端口存储器
  - 同一个存储器具有两组相互独立的读写控制电路
  - 由于进行并行的独立操作，因而是一种高速工作的存储器
- 双端口存储器IDT7133
  - 并行读写控制电路
  - 判别逻辑（冲突）



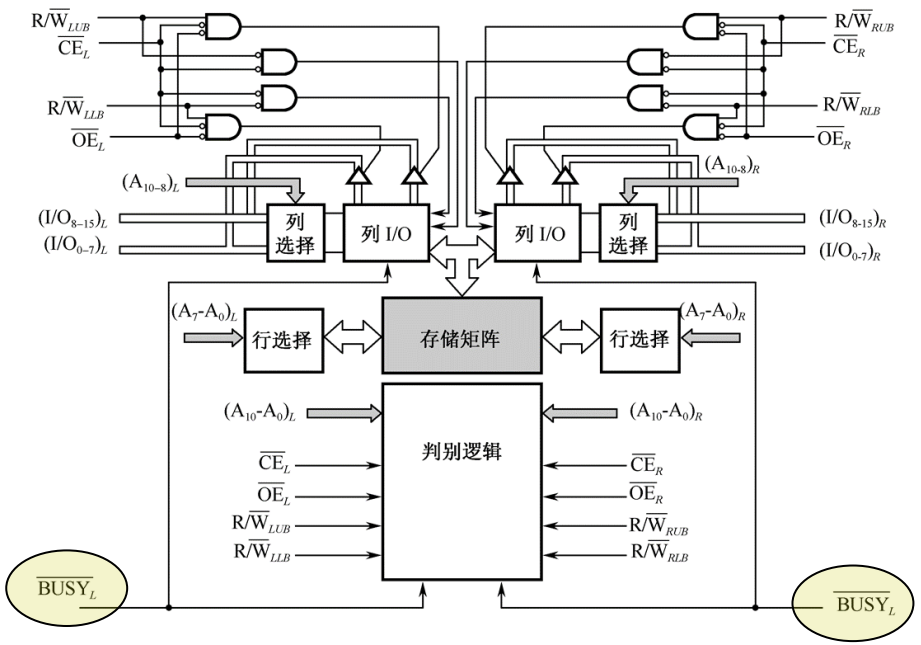
双端口存储器读写控制（1）

- 无冲突读写控制
  - 当两个端口的地址不相同时，在两个端口上进行读写操作，不会发生冲突
  - 每一个端口都有自己的片选控制(CE)和输出驱动控制(OE)
  - 读操作时，端口的OE(低电平有效)打开输出驱动器，由存储矩阵读出的数据就出现在I/O线上



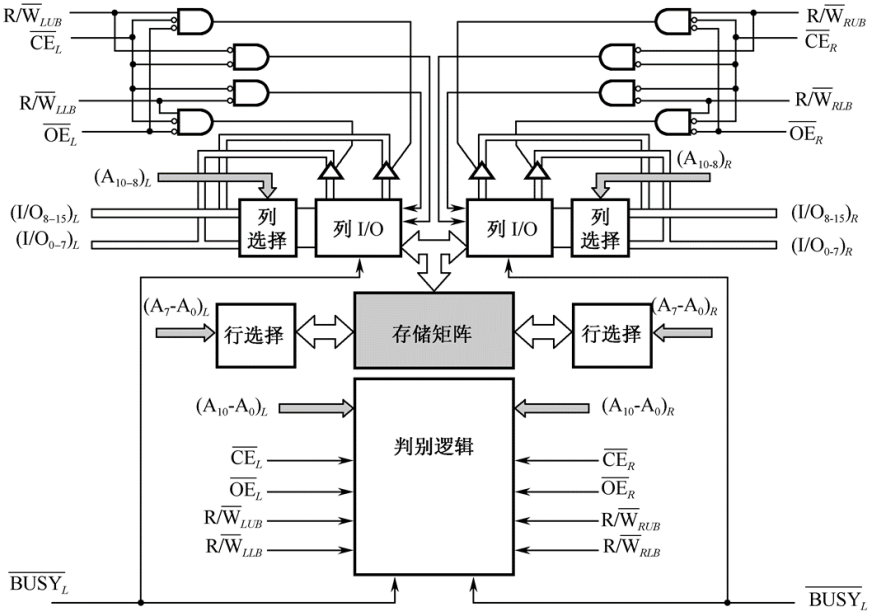
双端口存储器读写控制（2）

- 有冲突读写控制
  - 当两个端口同时**存取**存储器**同一存储单元**时，便发生读写冲突
  - 设置了BUSY标志
  - 片上的判断逻辑可以决定对哪个端口优先读写操作
  - 另一个被延迟的端口置BUSY标志(BUSY变为低电平)，即暂时关闭此端口

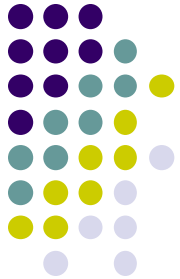


双端口存储器读写控制（3）

- 有冲突读写控制判断方法
- CE判断（片选有效判断）
  - 如果地址匹配且在CE之前有效，片上的控制逻辑在CEL和CER之间进行判断来选择端口
- 地址有效判断
  - 如果CE在地址匹配之前变低，片上的控制逻辑在左、右地址间进行判断来选择端口
- 判断标准：有效时间



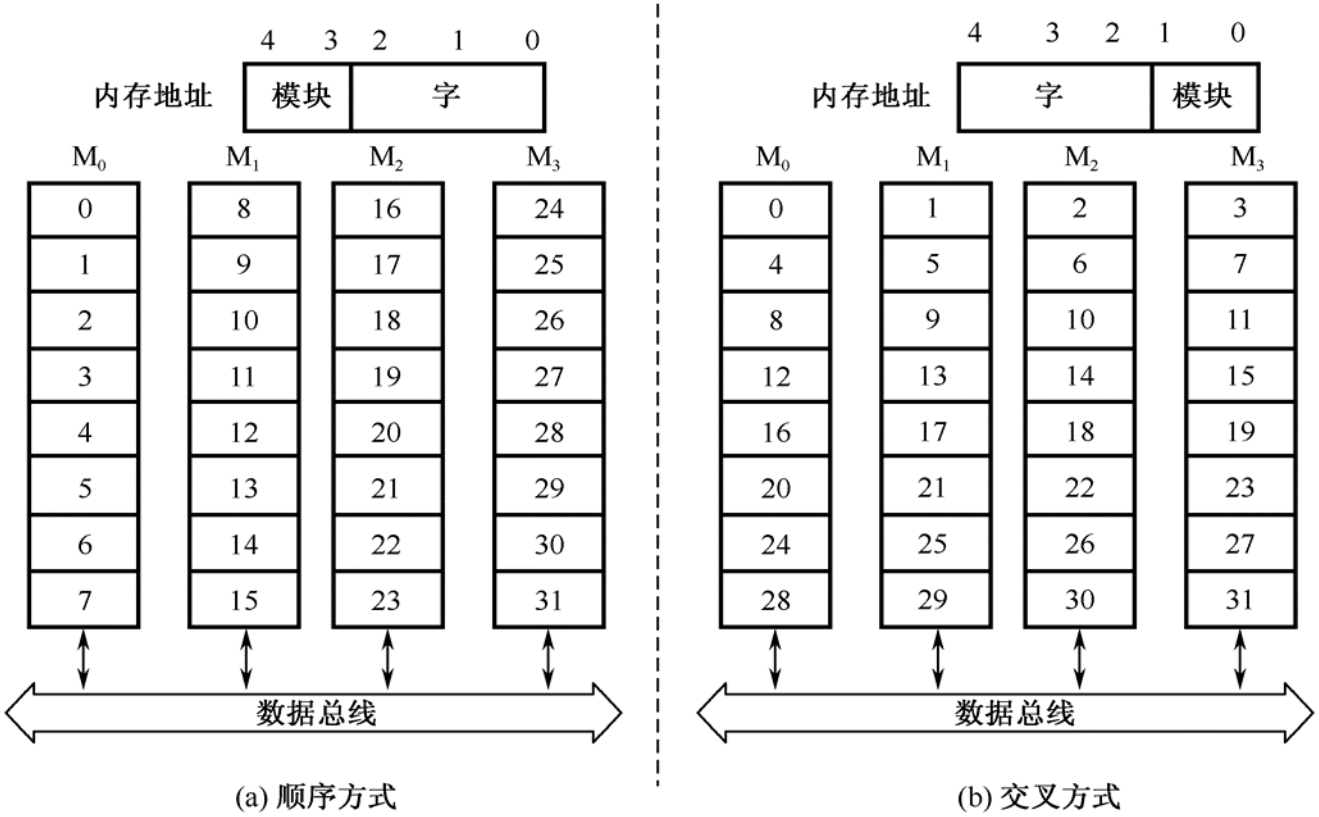
第三章 多层次的存储器



- 虚拟存储器
  - 虚存地址映射方式：页式/段式/段页式
  - 虚存替换算法
- 并行存储器
  - 双端口存储器（空间并行）
  - 多体交叉存储器（时间并行/流水线）

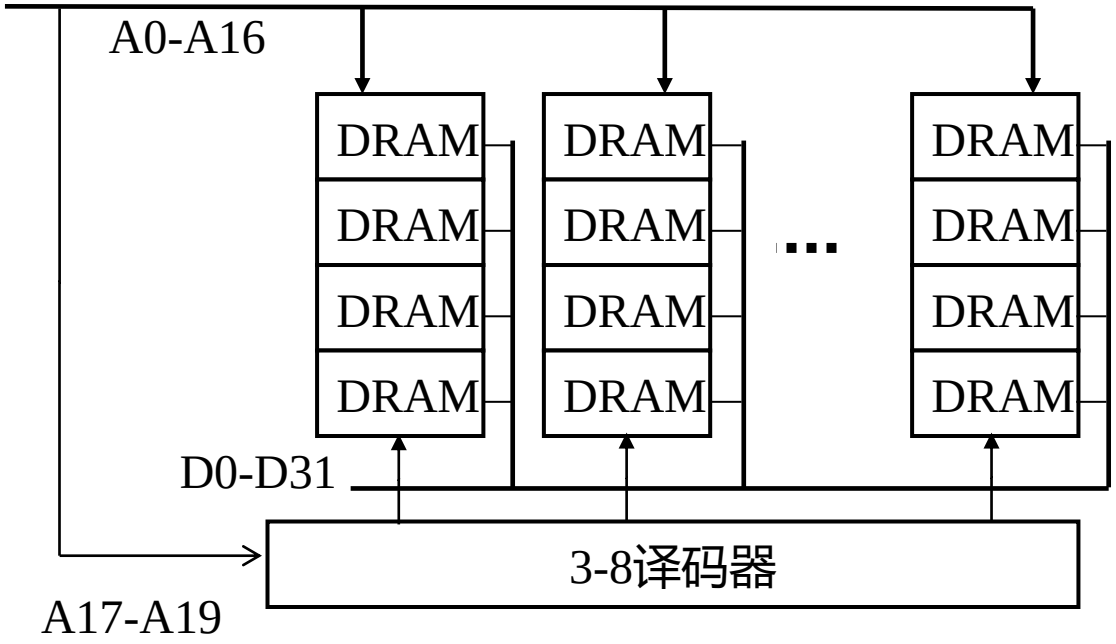
存储器的模块化组织

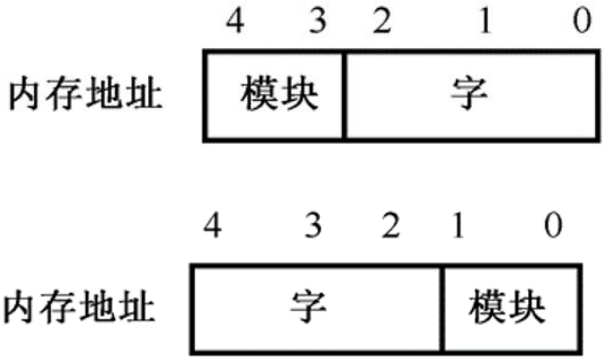
- 由若干个模块组成的主存储器：线性编址
- 编址方式
  - 顺序方式/交叉方式



思考：存储器组织—扩展关系

- 由若干个模块组成的主存储器：线性编址
- 编址方式
  - 顺序方式/交叉方式

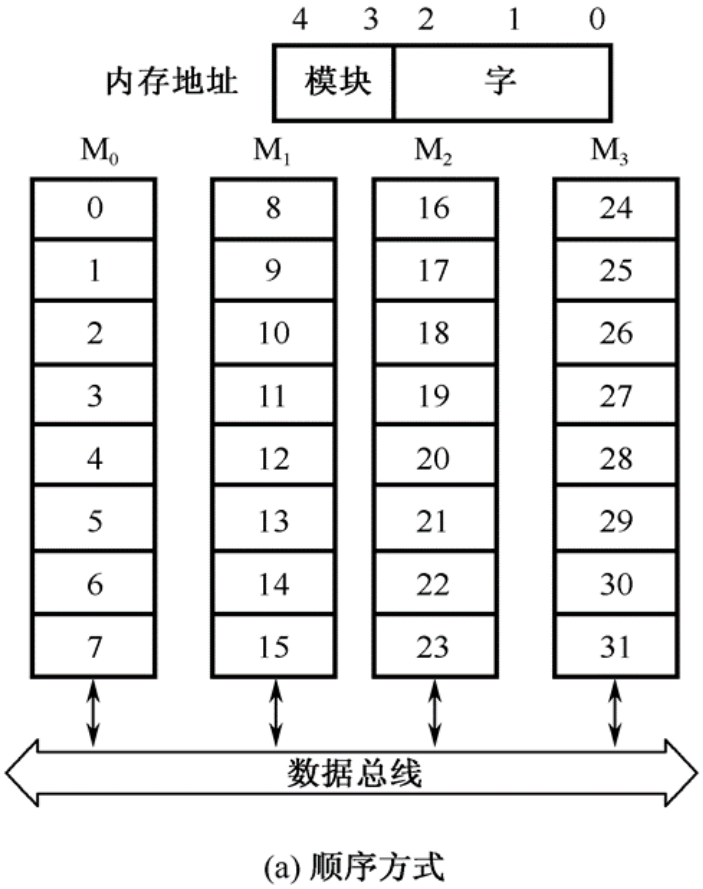




由128K*8位的DRAM芯片构成1024K*32位存储器

顺序方式

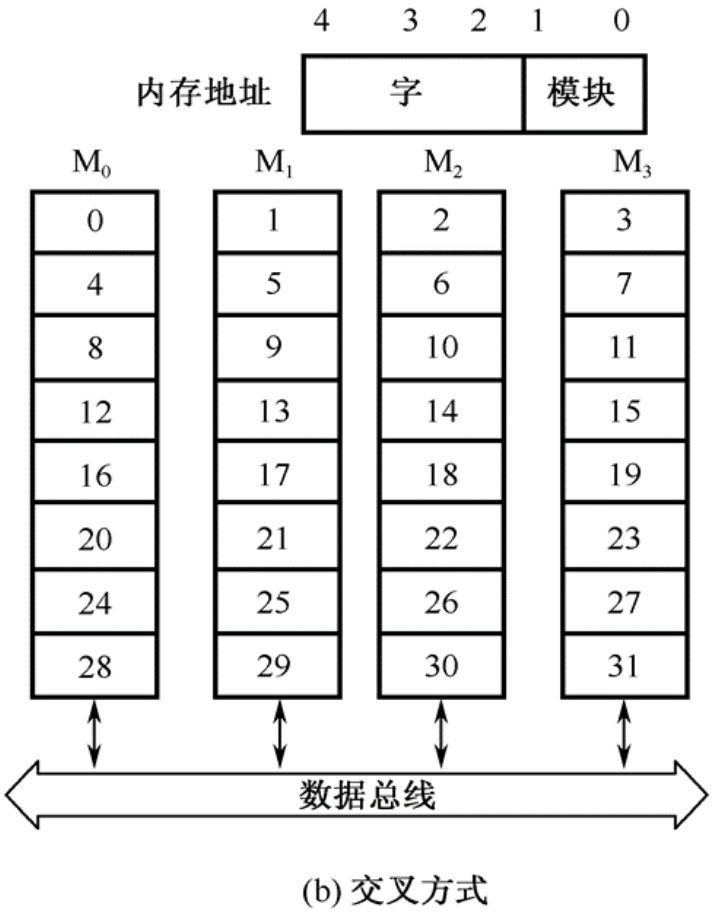
- 顺序方式
  - M0：0—7、M1：8 - 15
  - M2：16 - 23、M3：24 - 31
- 5位地址组织
  - 高位选模块，低位选块内地址
- 特点
  - 某个模块存取时，其他模块不工作
  - 优点：某一模块出现故障时，其他模块可以照常工作，通过增添模块来扩充存储器容量比较方便
  - 缺点：各模块串行工作，限制带宽



(a) 顺序方式

交叉方式

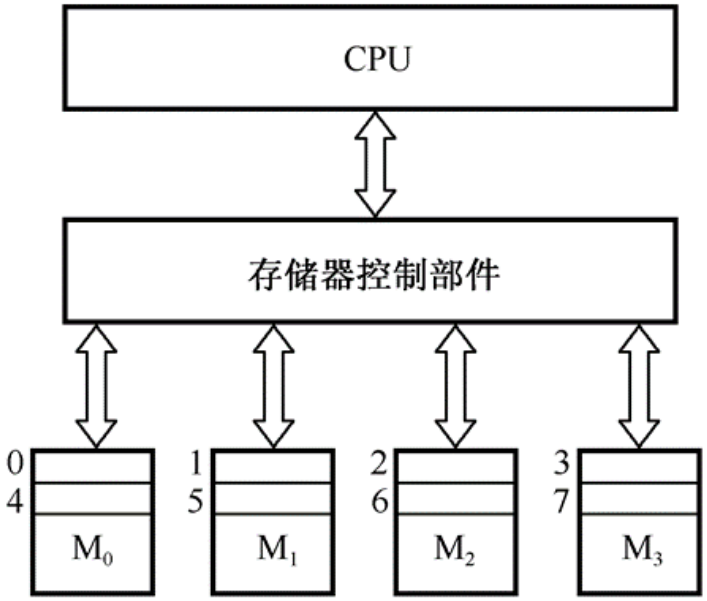
- 交叉方式：
  - M0：0, 4, … （除以4余数为0）
  - M1：1, 5, … （除以4余数为1）
  - M2：2, 6, … （除以4余数为2）
  - M3：3, 7, … （除以4余数为3）
- 5位地址组织
  - 高位选块内地址，低位选模块
- 特点
  - 连续地址分布在相邻的不同模块内
  - 优点：连续字的成块传送可实现多模块流水式并行存取，提高存储器带宽
  - 使用场合为成批数据读取



(b) 交叉方式

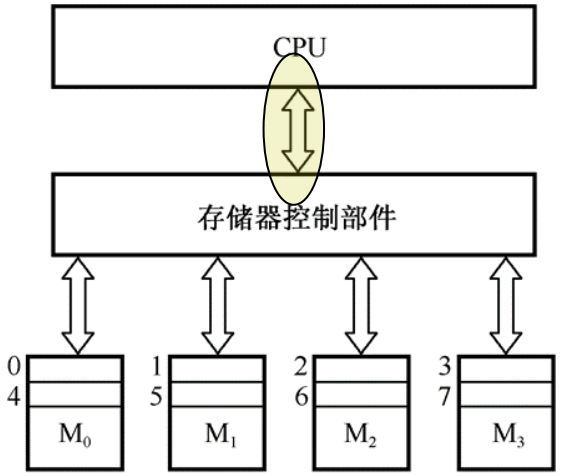
多模块交叉存储器基本结构

- 四模块交叉存储器结构框图
  - 主存被分成4个相互独立、容量相同的模块M0，M1，M2，M3
  - 每个模块都有自己的读写控制电路、地址寄存器和数据寄存器，各自以等同方式与CPU传送信息
  - 在理想情况下
    - 如果程序段或数据块都是连续地在主存中存取
    - 主存的访问速度提升接近4倍



多模块交叉存储器效率

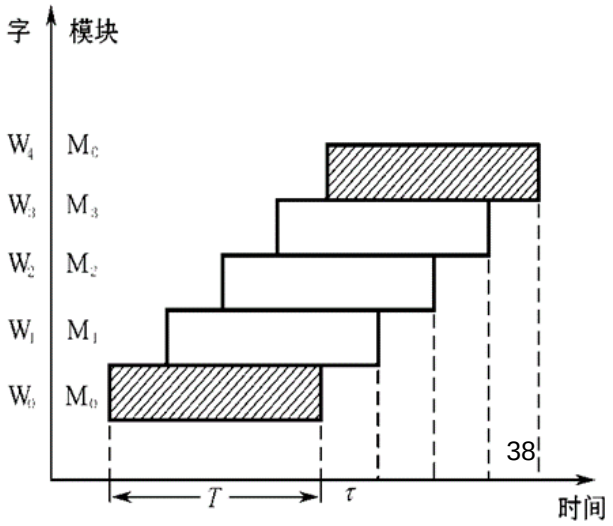
- 通常在一个存储器周期内
  - n个存储体必须分时启动
  - 存储体的启动间隔为 $\tau = T / n$
  - n为交叉存取度（顺序方式n=1，交叉方式为扩展芯片数）



- 存储器的存取速度有望提高n倍

$$t_{顺序} = xT$$

$$t_{交叉} = T + (x-1)\tau = T(\frac{x+n-1}{n})$$

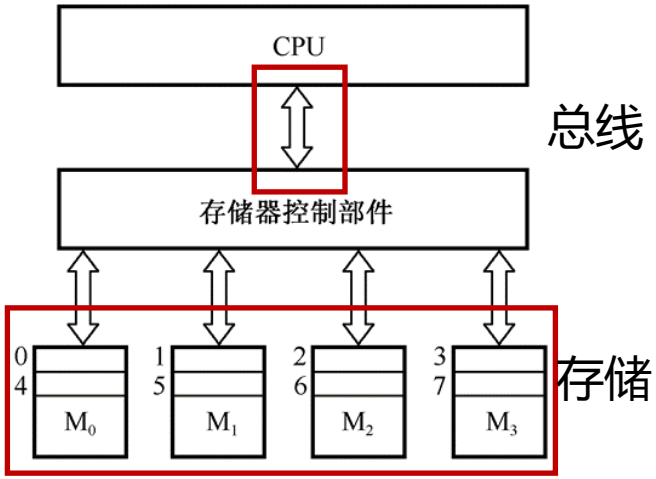


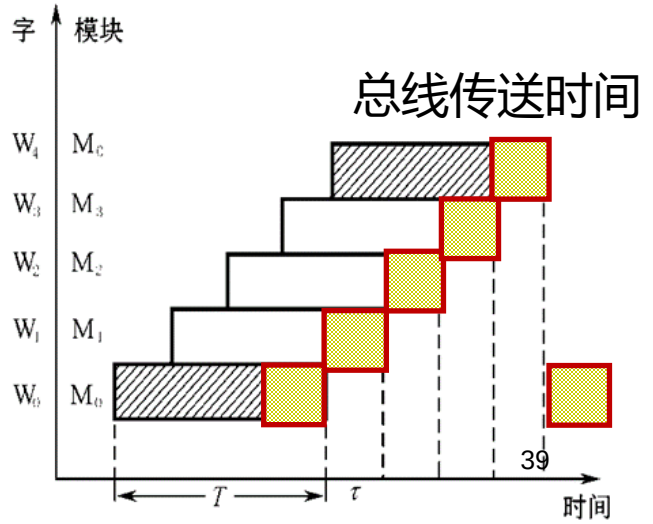
存储周期与总线周期

- 存储周期
  - 存储器读出一个字所需时间
  - 记作：T
- 总线传送周期
  - 总线传送一个字所需时间
  - 记作：τ（最高带宽=字长/总线周期）
- 关系
  - 各存储体共用总线
  - 存储体的启动间隔不低于总线周期
  - 一般设计上使得

$$\tau = T / n$$

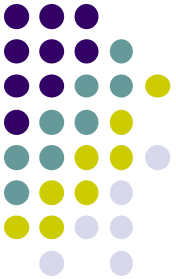
  - 调整模块数n，流水平衡状态





例题：设存储器容量为32字，字长64位，模块数m=4，分别用顺序方式和交叉方式进行组织。存储周期T=200ns，数据总线宽度为64位，总线传送周期=50ns。

若连续读出4个字，问顺序存储和交叉存储的带宽各是多少?



例题：设存储器容量为32字，字长64位，模块数m=4，分别用顺序方式和交叉方式进行组织。存储周期T=200ns，数据总线宽度为64位，总线传送周期=50ns。
若连续读出4个字，问顺序存储和交叉存储的带宽各是多少?

解：

顺序存储器和交叉存储器连续读出m=4个字的信息总量都是：

$$q=64\text{（字长）}\times 4=256\ \text{bit}$$

顺序存储器和交叉存储器连续读出4个字所需的时间分别是：

$$t_{顺序}=mT=4\times 200ns=800ns=8\times 10^{-7}s$$

$$t_{交叉}=T+(m-1)\tau=200+3*200/4=350ns=35\times 10^{-7}s$$

顺序存储器和交叉存储器的带宽分别是：

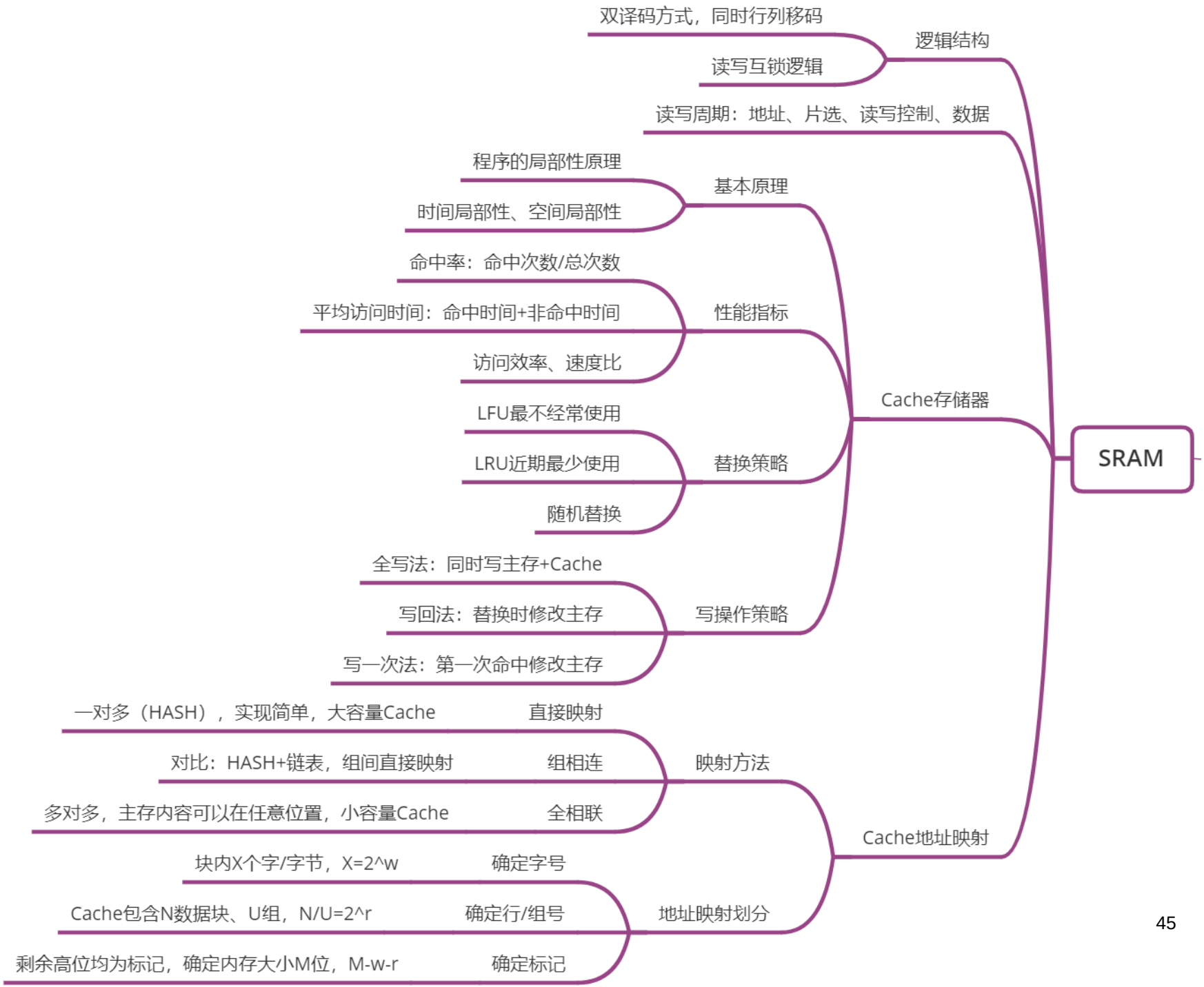
$$W_{顺序}=q/t_{顺序}=256b\div(8\times 10^{-7})s=320Mb/s$$

$$W_{交叉}=q/t_{交叉}=256b\div(35\times 10^{-7})s=730Mb/s$$

多层次的存储器——总结

- 基本概念
  - 存储器分类
    - 存储特性
      - 可读可写
        - 一般只读：BIOS、光盘
        - 可读可写：内存、硬盘等
      - 访问方式
        - 随机访问：RAM、ROM
        - 串行访问：磁带（顺序）
      - 内容是否易失
        - 非易失：硬盘、BIOS
        - 易失：内存、Cache
    - 存储单元
      - 随机存储器（RAM）
        - DRAM：电容存电荷、内存
        - SRAM：双稳态触发器、Cache
      - 只读存储器（ROM）
        - 掩膜/EPROM/EEPROM
        - FLASH：SSD、U盘
      - 其他存储器
        - 磁性介质（硬盘）、光盘（外部存储讲）
  - 技术指标
    - 存储容量：16K*8bit
    - 存储周期
      - 连续启动两次读操作所需最小时间间隔
      - 存储周期略大于存取时间
    - 存储器带宽
      - 单位：MB/s、Mbps
  - 进制
    - 1000进制：Mbps、Gbps、MHz、GHz
    - 1024进制：KB、MB、GB
  - 多层次的存储结构
    - 寄存器—Cache（高速缓存）—主存储器—外存
    - 目标：成本 v.s. 性能


DRAM
逻辑结构
分时送地址码，扩大寻址范围
刷新计数器
输入输出缓冲放大器
读写时序
行地址、列地址分别使能
刷新
结构特点：按行刷新（一次一行）
刷新方法
集中刷新（连续对所有行进行刷新）
分散/异步刷新（刷新信号穿插）
刷新周期/行数、向下取整（按存储周期）
行数计算（单芯片）：2^17=2^9（行数）*2^8（列数）
存储容量扩展
位扩展
地址线、控制线公用，数据线分别连接至各芯片
多个存储芯片作为整体，当作一个新的存储体
字扩展
低位地址线、数据线公用；高位地址线用作译码（控制线端口）
多个芯片分时工作，每次仅一个芯片选中（控制）
混合扩展
高级DRAM方式
异步存储：FPM DRAM、CDRAM
同步存储：SDRAM、DDR


SRAM
Cache存储器
逻辑结构
双译码方式，同时行列移码
读写互锁逻辑
读写周期：地址、片选、读写控制、数据
基本原理
程序的局部性原理
时间局部性、空间局部性
性能指标
命中率：命中次数/总次数
平均访问时间：命中时间+非命中时间
访问效率、速度比
替换策略
LFU最不经常使用
LRU近期最少使用
随机替换
写操作策略
全写法：同时写主存+Cache
写回法：替换时修改主存
写一次法：第一次命中修改主存
Cache地址映射
映射方法
直接映射
一对多（HASH），实现简单，大容量Cache
组相连
对比：HASH+链表，组间直接映射
全相联
多对多，主存内容可以在任意位置，小容量Cache
地址映射划分
确定字号
块内X个字/字节，X=2^w
确定行/组号
Cache包含N数据块、U组，N/U=2^r
确定标记
剩余高位均为标记，确定内存大小M位，M-w-r


虚拟存储器
目的
解决主存、辅存速度/容量不匹配
（虚拟）扩大内存空间
地址映射方式
页式、段式、段页式
虚存替换算法
FIFO、LRU、LFU
并行存储器
双端口存储器
片选有效判断、地址有效判断
端口冲突（BUSY标志）
多体交叉存储器
顺序方式/交叉方式
多模块交叉存储器效率
t顺序=mT
t交叉=T+（m-1）tau