



操作系统

Operating System

虚拟存储概念

- 虚拟存储的需求背景
- 覆盖技术
- 交换技术
- 局部性原理
- 虚拟存储概念
- 虚拟页式存储
- 缺页异常

增长迅速的存储需求

电脑游戏

一代	二代	三代	四代	五代	六代	七代	八代
437K	883K	1.9M	6M	6.3M	59M	100M	138M



程序规模的增长速度远远大于存储器容量的增长速度

存储层次结构

■ 理想中的存储器

容量更大、速度更快、价格更便宜的非易失性存储器

■ 实际中的存储器

存储器层次结构

访问时间

1 纳秒

2 纳秒

10 纳秒

10 毫秒

100 秒

寄存器

高速缓存

内存

磁盘

磁带

容量

< 1 KB

1 MB

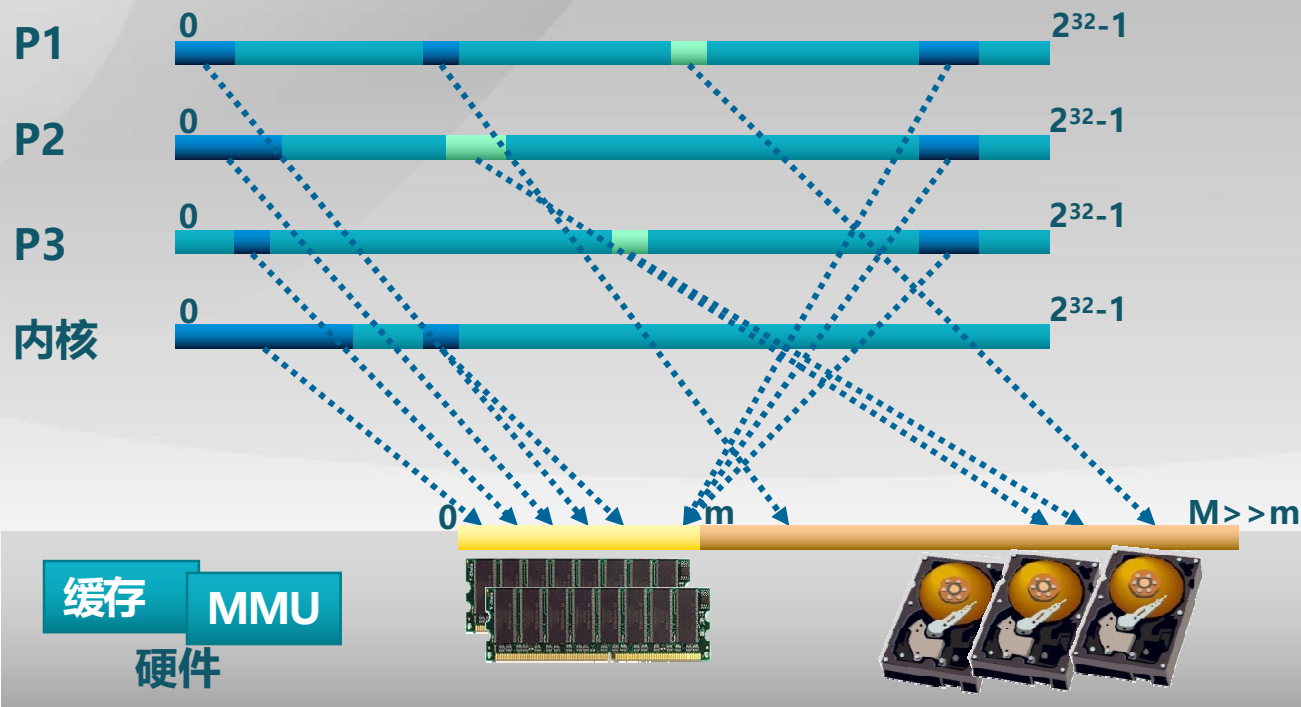
64-512 MB

5-50 GB

20-100 GB

操作系统的存储抽象

■ 操作系统对存储的抽象：地址空间



虚拟存储需求

■ 计算机系统时常出现**内存空间不够用**

▣ 覆盖 (overlay)

应用程序**手动**把需要的指令和数据保存在内存中

▣ 交换 (swapping)

操作系统**自动**把暂时不能执行的程序保存到外存中

▣ 虚拟存储

在有限容量的内存中，以页为单位**自动**装入**更多更大**的程序



操作系统

Operating System



操作系统

Operating System

覆盖技术

■ 目标

- ▣ 在较小的可用内存中运行较大的程序

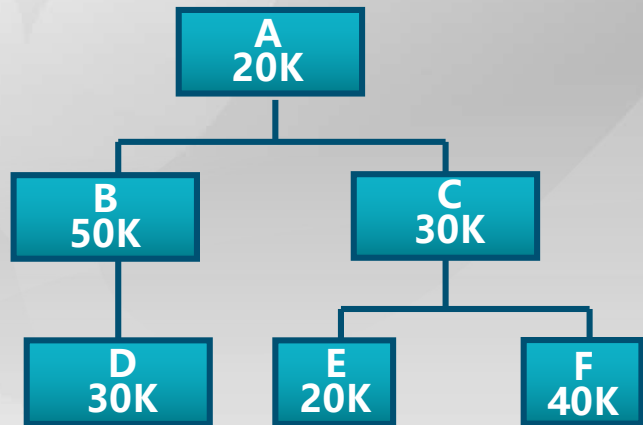
■ 方法

依据程序逻辑结构，将程序划分为若干**功能相对独立**的模块；将不会同时执行的模块**共享同一块内存区域**

- ▣ 必要部分（常用功能）的代码和数据常驻内存
- ▣ 可选部分（不常用功能）放在其他程序模块中,只在需要用到时装入内存
- ▣ 不存在调用关系的模块可相互覆盖，共用同一块内存区域

覆盖技术示例

程序调用结构：190K

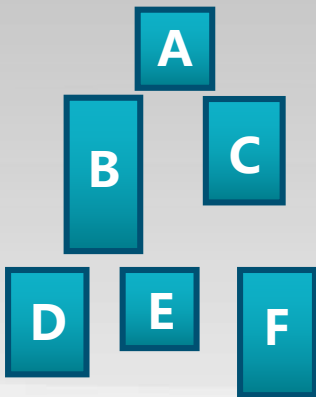


内存总共：110K

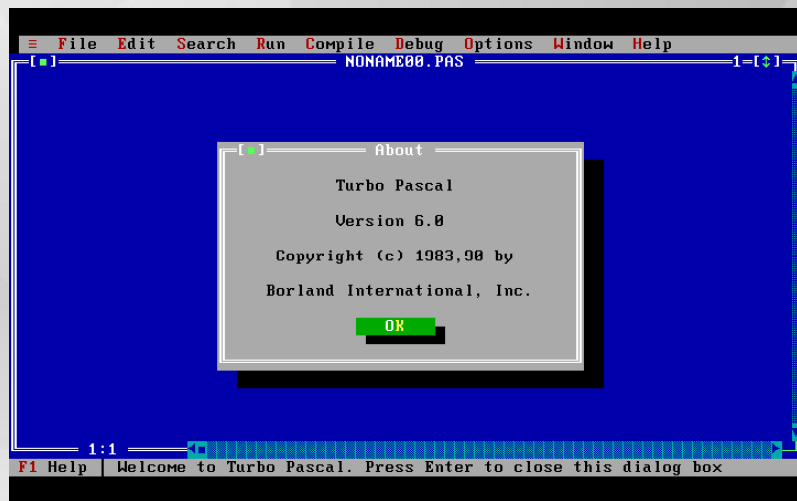


另一种调用方法：(100K)

- A占一个分区：20K
- B、E和F共用一个分区：50K
- C和D共用一个分区：30K



覆盖技术的不足



Turbo Pascal的Overlay系统单元
支持程序员控制的覆盖技术

■ 增加编程困难

- ▶ 需程序员划分功能模块，并确定模块间的覆盖关系
- ▶ 增加了编程的复杂度；

■ 增加执行时间

- ▶ 从外存装入覆盖模块
- ▶ 时间换空间

交换技术

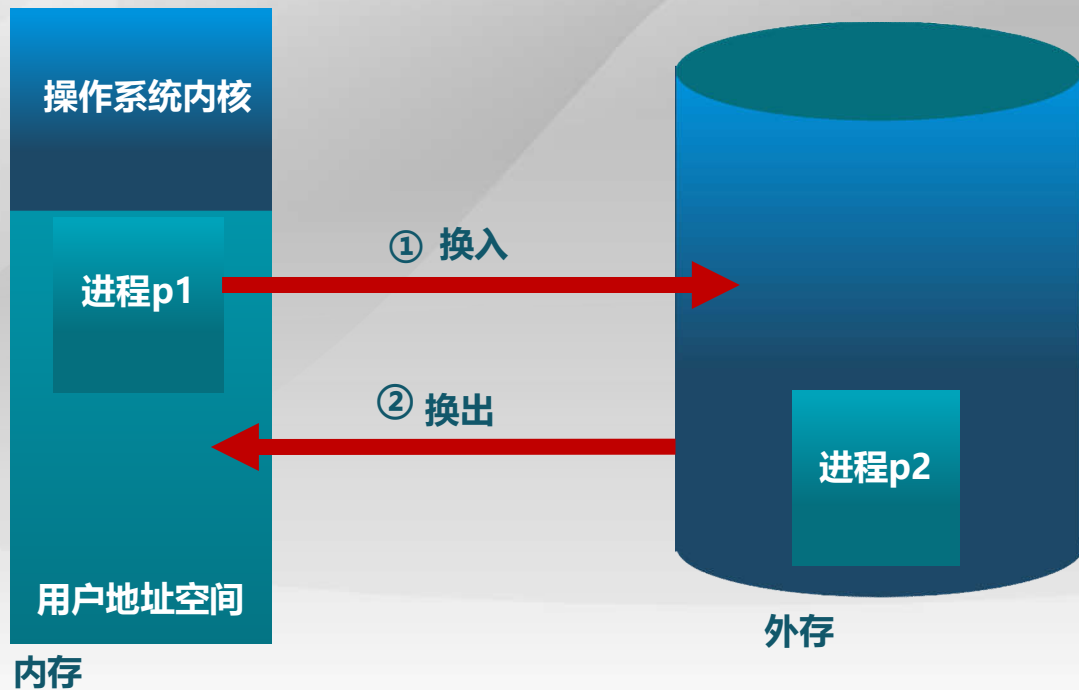
■ 目标

- ▣ 增加正在运行或需要运行的程序的内存

■ 实现方法

- ▣ 可将暂时不能运行的程序放到外存
- ▣ 换入换出的基本单位
 - ▣ 整个进程的地址空间
- ▣ 换出 (swap out)
 - ▣ 把一个进程的整个地址空间保存到外存
- ▣ 换入 (swap in)
 - ▣ 将外存中某进程的地址空间读入到内存

交换技术



(本图摘自Silberschatz, Galvin and Gagne: “Operating System Concepts”)

交换技术面临的问题

- 交换时机：何时需要发生交换？
 - ▣ 只当内存空间不够或有不够的可能时换出
- 交换区大小
 - ▣ 存放所有用户进程的所有内存映像的拷贝
- 程序换入时的重定位：换出后再换入时要放在原处吗？
 - ▣ 采用动态地址映射的方法

覆盖与交换的比较

■ 覆盖

- ▶ 只能发生在没有调用关系的模块间
- ▶ 程序员须给出模块间的逻辑覆盖结构
- ▶ 发生在运行程序的内部模块间

■ 交换

- ▶ 以进程为单位
- ▶ 不需要模块间的逻辑覆盖结构
- ▶ 发生在内存进程间



操作系统

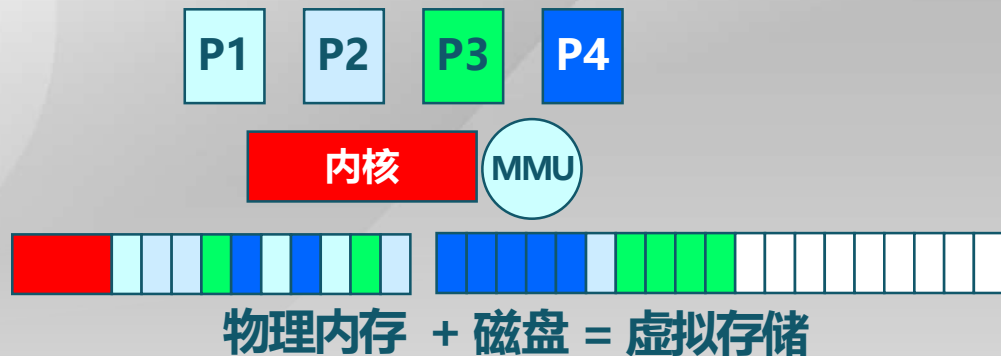
Operating System



操作系统

Operating System

虚拟存储技术的目标



- 只把部分程序放到内存中，从而运行比物理内存大的程序
 - ▶ 由操作系统自动完成，无需程序员的干涉
- 实现进程在内存与外存之间的交换，从而获得更多的空闲内存空间
 - ▶ 在内存和外存之间只交换进程的部分内容

局部性原理 (principle of locality)

- 程序在执行过程中的一个较短时期，所执行的指令地址和指令的操作数地址，分别局限于一定区域

- ▶ 时间局部性

- ▶ 一条指令的一次执行和下次执行，一个数据的一次访问和下次访问都集中在一个较短时期内

- ▶ 空间局部性

- ▶ 当前指令和邻近的几条指令，当前访问的数据和邻近的几个数据都集中在一个较小区域内

- ▶ 分支局部性

- ▶ 一条跳转指令的两次执行，很可能跳到相同的内存位置

- 局部性原理的意义

- ▶ 从理论上来说，虚拟存储技术是能够实现的，而且可取得满意的效果

不同程序编写方法的局部性特征

例子：页面大小为4K，分配给每个进程的物理页面数为1。在一个进程中，定义了如下的二维数组
`int A[1024][1024]`，该数组按行存放在内存，
每一行放在一个页面中

程序编写方法1：

```
for (j = 0; j < 1024; j++)  
  for (i = 0; i < 1024; i++)  
    A[i][j] = 0;
```

程序编写方法2：

```
for (i=0; i<1024; i++)  
  for (j=0; j<1024; j++)  
    A[i][j] = 0;
```

不同程序编写方法的局部性特征

0	$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,1023}$
1	$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,1023}$
				
				
1023	$a_{1023,0}$	$a_{1023,1}$		$a_{1023,1023}$

访问页面的序列为：

解法1：

0, 1, 2,1023, 0, 1,, 共1024组

共发生了 1024×1024 次缺页中断

解法2：

0, 0, 1, 1,, 2, 2,, 3, 3,

共发生了1024次缺页中断



操作系统

Operating System



操作系统

Operating System

虚拟存储的基本概念

■ 思路

- ▣ 将不常用的部分内存块暂存到外存

■ 原理：

▣ 装载程序时

- ▣ 只将当前指令执行需要的部分页面或段装入内存

▣ 指令执行中需要的指令或数据不在内存（称为缺页或缺段）时

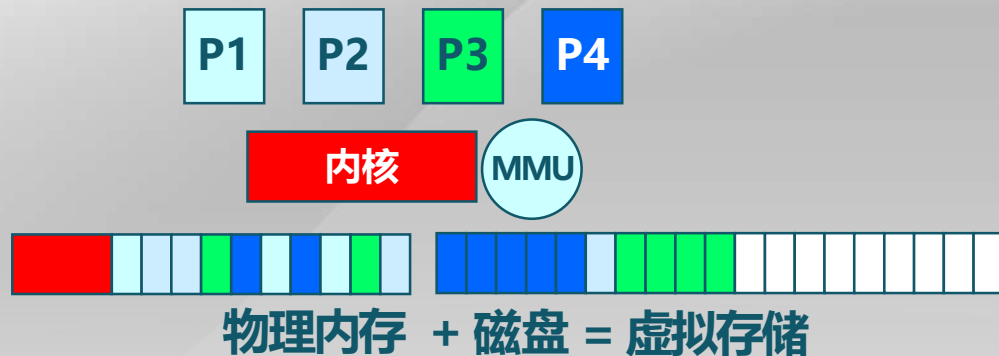
- ▣ 处理器通知操作系统将相应的页面或段调入内存

▣ 操作系统将内存中暂时不用的页面或段保存到外存

■ 实现方式

- ▣ 虚拟页式存储
- ▣ 虚拟段式存储

虚拟存储的基本特征



■ 不连续性

- ▣ 物理内存分配非连续
- ▣ 虚拟地址空间使用非连续

■ 大用户空间

- ▣ 提供给用户的虚拟内存可大于实际的物理内存

■ 部分交换

- ▣ 虚拟存储只对部分虚拟地址空间进行调入和调出

虚拟存储的支持技术

■ 硬件

- ▣ 页式或短时存储中的地址转换机制

■ 操作系统

- ▣ 管理内存和外存间页面或段的换入和换出



操作系统

Operating System



操作系统

Operating System

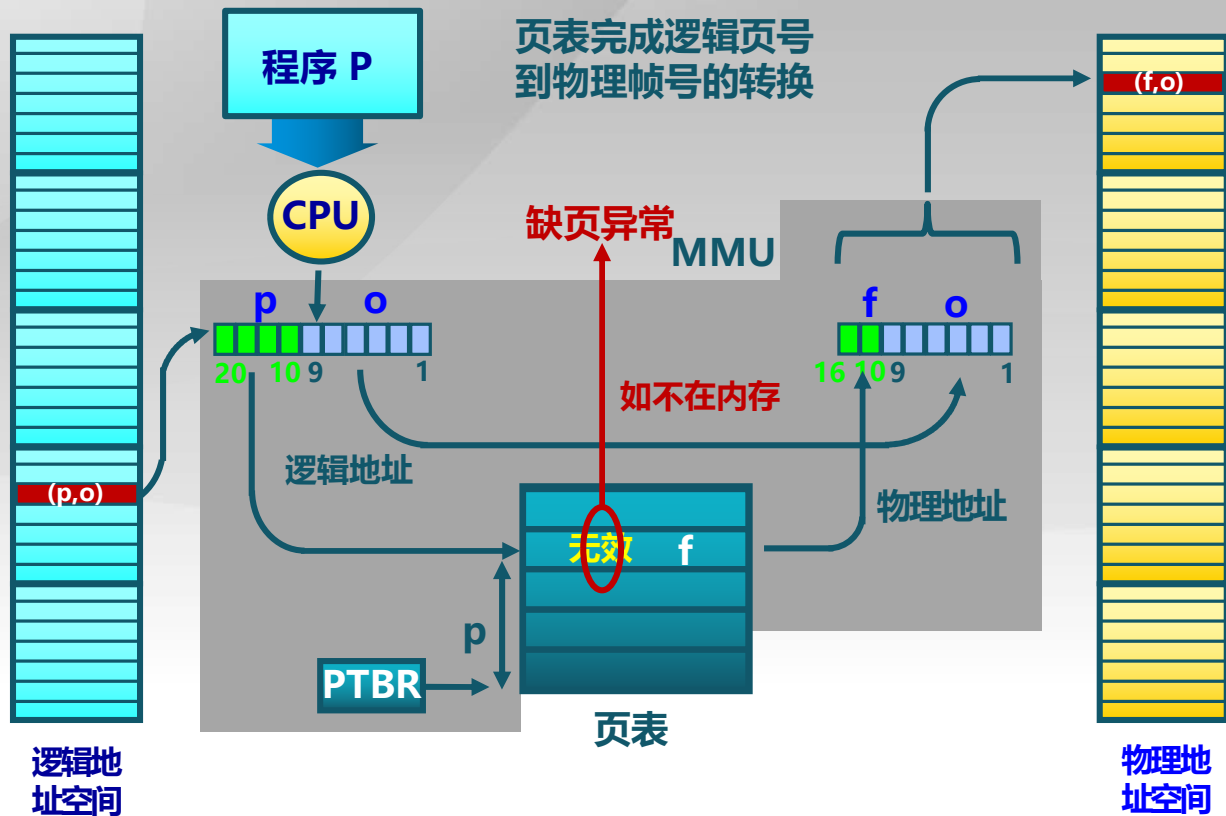
虚拟页式存储管理

- 在页式存储管理的基础上，增加请求调页和页面置换

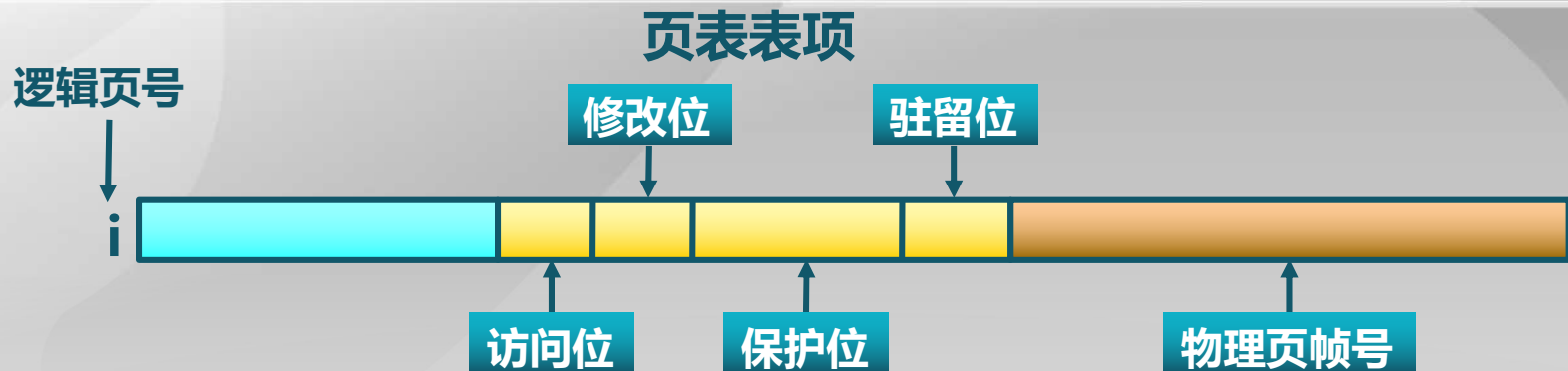
- 思路

- ▶ 当用户程序要装载到内存运行时，只装入部分页面，就启动程序运行
- ▶ 进程在运行中发现有需要的代码或数据不在内存时，则向系统发出缺页异常请求
- ▶ 操作系统在处理缺页异常时，将外存中相应的页面调入内存，使得进程能继续运行

虚拟页式存储中的地址转换

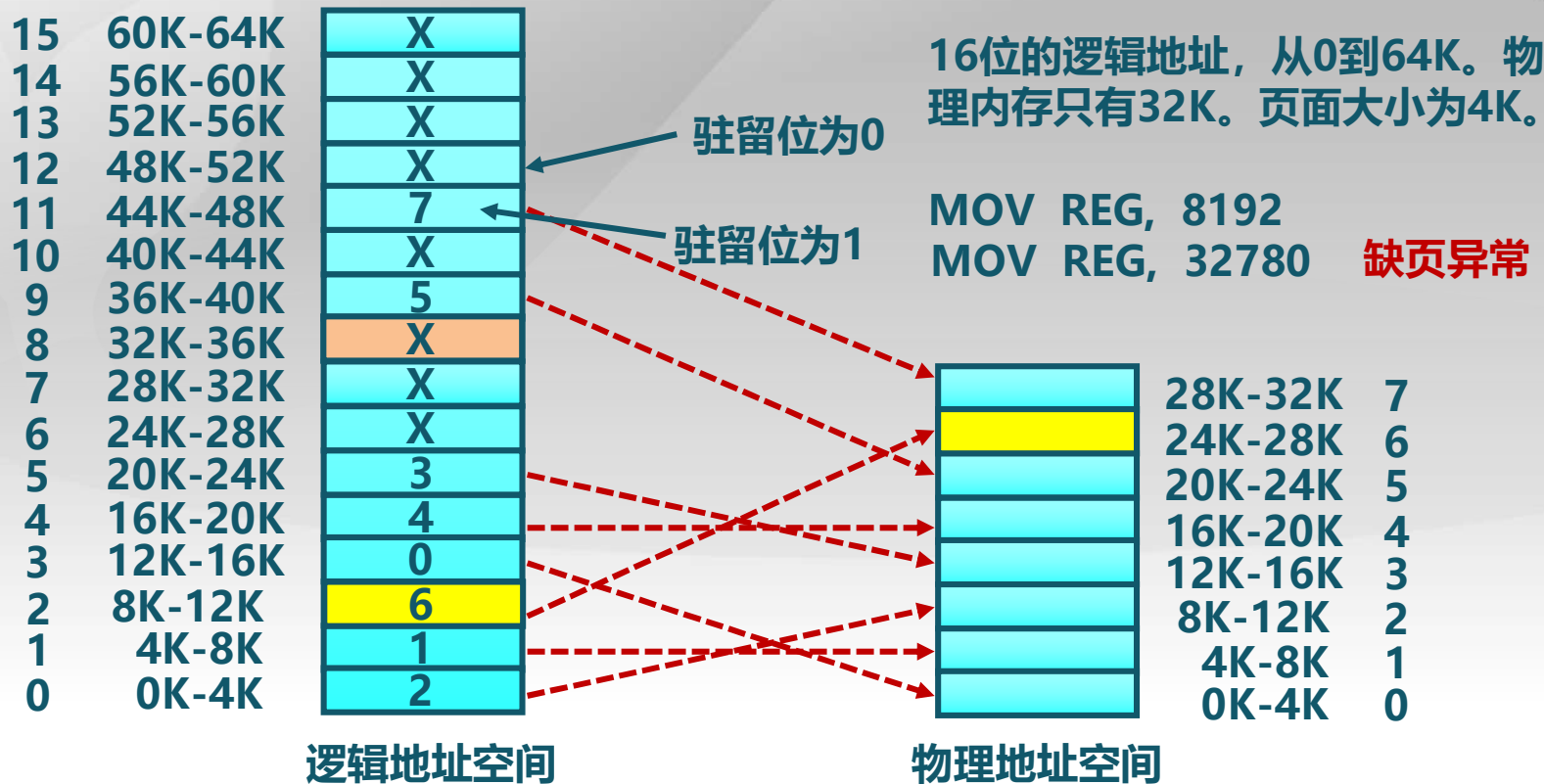


虚拟页式存储中的页表项结构

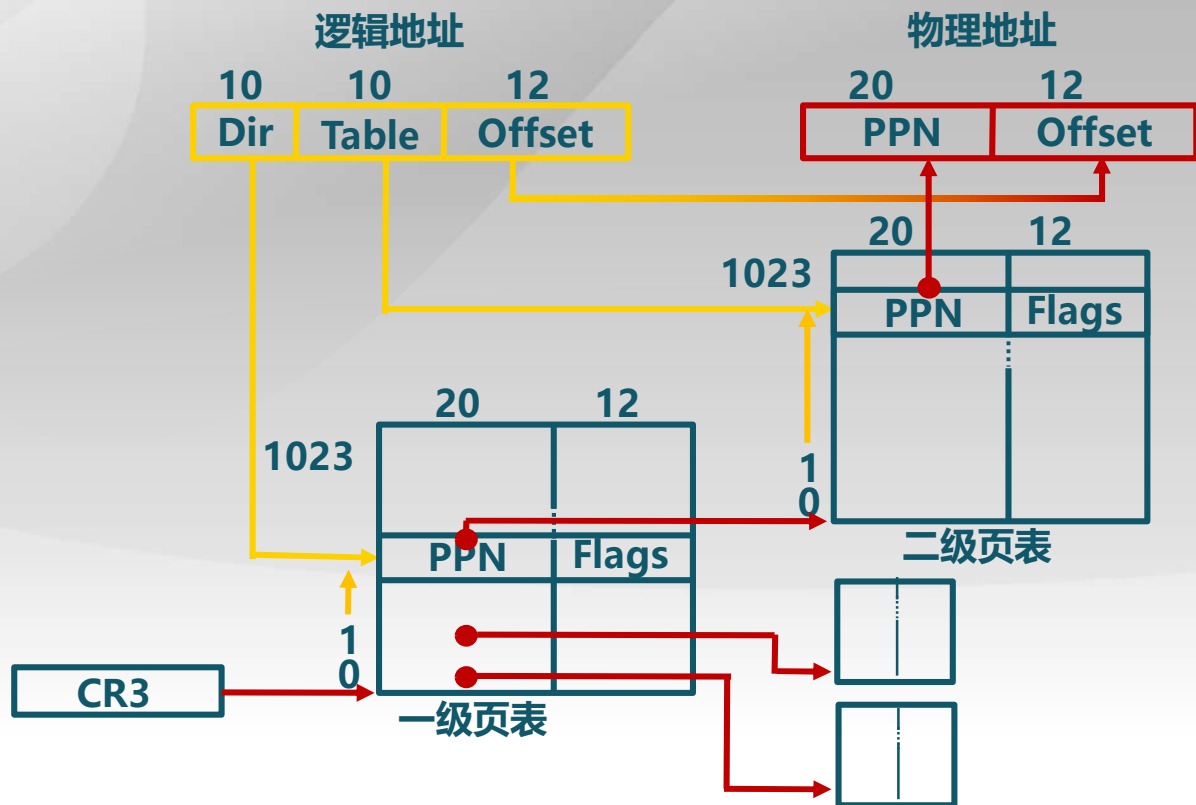


- 驻留位：表示该页是否在内存
- 修改位：表示在内存中的该页是否被修改过
- 访问位：表示该页面是否被访问过（读或写）
- 保护位：表示该页的允许访问方式
 - 只读、可读写、可执行等

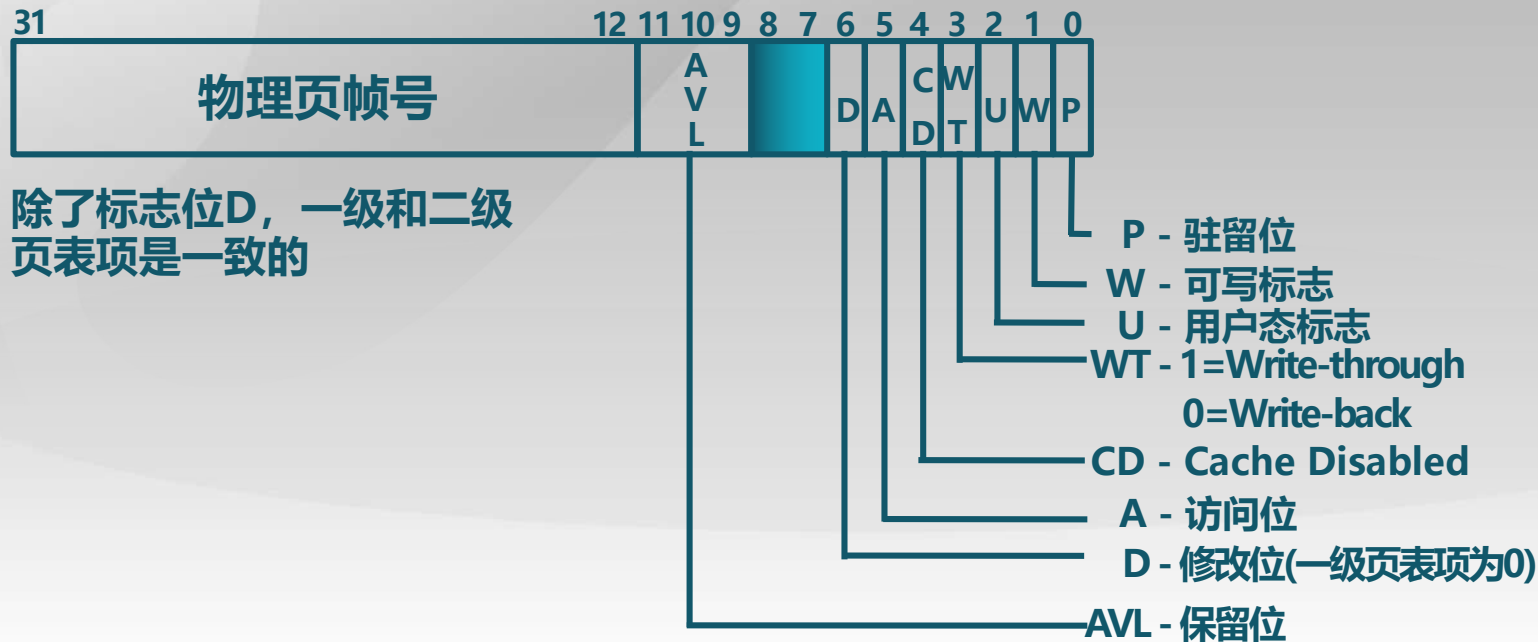
虚拟页式存储示例



X86页表结构



X86页表项结构





操作系统

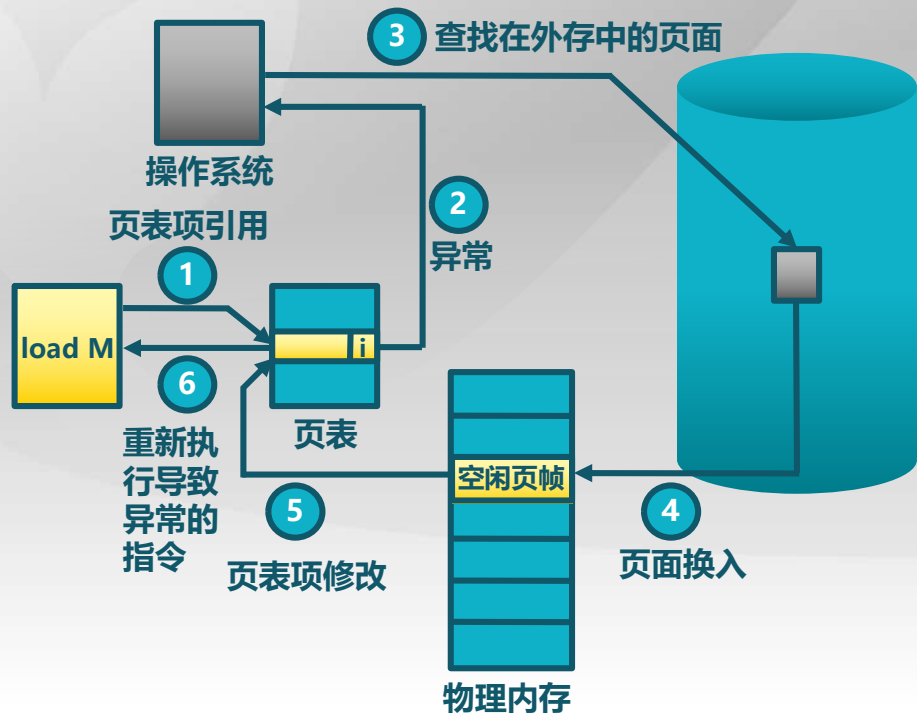
Operating System



操作系统

Operating System

缺页异常（缺页中断）的处理流程



- A. 在内存中有空闲物理页面时，分配一物理页帧 f ，转第E步；
- B. 依据页面置换算法选择将被替换的物理页帧 f ，对应逻辑页 q
- C. 如 q 被修改过，则把它写回外存；
- D. 修改 q 的页表项中驻留位置为0；
- E. 将需要访问的页 p 装入到物理页面 f
- F. 修改 p 的页表项驻留位为1，物理页帧号为 f ；
- G. 重新执行产生缺页的指令

虚拟页式存储中的外存管理

- 在何处保存未被映射的页？
 - ▣ 应能方便地找到在外存中的页面内容
 - ▣ 交换空间（磁盘或者文件）
 - ▣ 采用特殊格式存储未被映射的页面
- 虚拟页式存储中的外存选择
 - ▣ 代码段：可执行二进制文件
 - ▣ 动态加载的共享库程序段：动态调用的库文件
 - ▣ 其它段：交换空间

虚拟页式存储管理的性能

■ 有效存储访问时间 (effective memory access time EAT)

▶ $EAT = \text{访存时间} * (1-p)$
+ 缺页异常处理时间 * 缺页率 p

▶ 例子

▶ 访存时间: 10 ns

▶ 磁盘访问时间: 5 ms

▶ 缺页率 p

▶ 页修改概率 q

▶ $EAT = 10(1-p) + 5,000,000p(1+q)$



操作系统

Operating System