

202X

操作系统寒假帮扶讲座2

2024.2.22

特别提醒：此PPT受众主要为补考同学，对于参加正常考试的同学仅供参考！

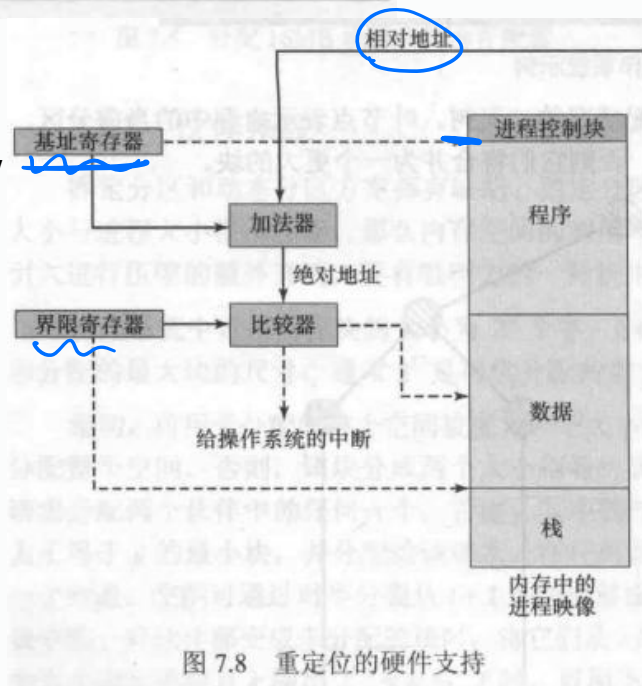
第7章 内存管理

- 术语：

表 7.1 内存管理术语

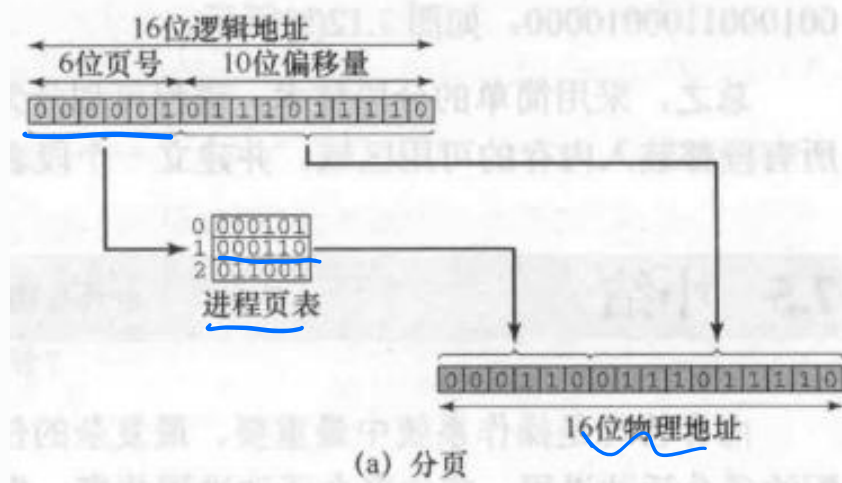
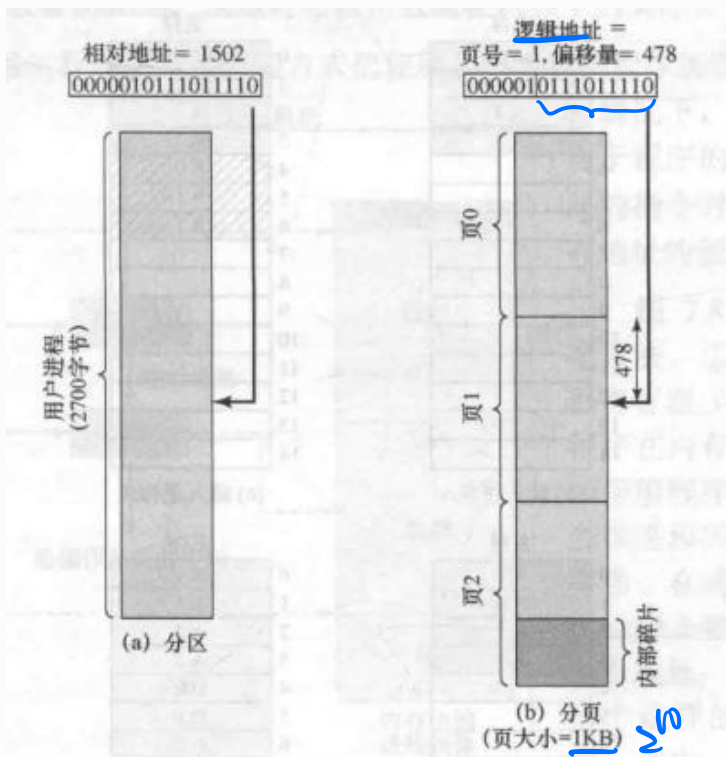
页框	内存中固定长度的块
页	固定长度的数据块，存储在二级存储器中（如磁盘）。数据页可以临时复制到内存的页框中
段	变长数据块，存储在二级存储器中。整个段可以临时复制到内存的一个可用区域中（分段），或可以将一个段分为许多页，然后将每页单独复制到内存中（分段与分页相结合）

- 内存分区：固定分区、动态分区、伙伴系统
- 逻辑地址：指与当前数据在内存中的物理分配地址无关的访问地址，在执行对内存的访问之前必须把它转换为物理地址
- 相对地址：逻辑地址的一个特例，相对于某些已知点（通常是程序的开始）的存储单元
- 物理地址：数据在内存中的实际位置
- 重定位：把使用相对地址的程序加载到内存，需要把相对地址转换为物理地址



第7章 内存管理

- 分页：内存被划分为大小固定、相等的块，且块相对比较小，每个进程也被分成同样大小的小块，则进程中称为页的块可以分配到内存中称为页框的可用块
- 页表：操作系统为每个进程维护一个页表，页表给出了该进程的每页所对应页框的位置
- 计算地址：



第7章 内存管理

- 分段：把程序和与其相关的数据划分到几个段中，段的长度可以不等
- 段表：每个进程都有一个段表，每个段表项给出相应段在内存中的起始地址和段的长度
- 计算地址：

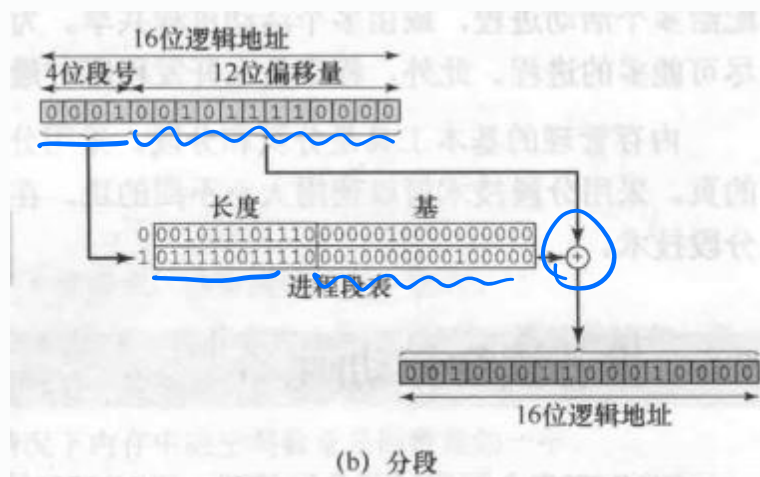
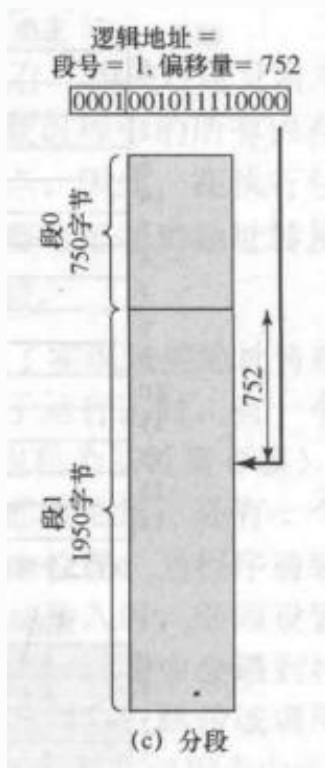


图 7.12 逻辑地址转换为物理地址示例

第8章 虚拟内存

- 术语：

表 8.1 虚拟内存术语

虚拟内存	在存储分配机制中，尽管备用内存是主存的一部分，但它也可被寻址。程序引用内存使用的地址与内存系统用于识别物理存储站点的地址是不同的，程序生成的地址会自动转换为机器地址。虚拟存储的大小受计算机系统寻址机制和可用的备用内存量的限制，而不受主存储位置实际数量的限制
虚拟地址	在虚拟内存中分配给某一位置的地址，它使得该位置可被访问，就好像是主内的一部分那样
虚拟地址空间	分配给进程的虚拟存储
地址空间	用于某进程的内存地址范围
实地址	内存中存储位置的地址

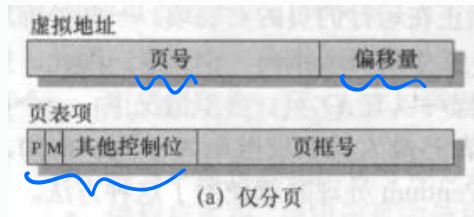
- 分页和分段特点：

表 8.2 分页和分段的特点

简单分页	虚存分页	简单分段	虚存分段
内存划分为大小固定的小块，称为页框	内存划分为大小固定的小块，称为页框	内存未划分	内存未划分
程序被编译器或内存管理系统划分为页	程序被编译器或内存管理系统划分为页	由程序员给编译器指定程序段（即由程序员决定）	由程序员给编译器指定程序段（即由程序员决定）
页框中有内部碎片 无外部碎片	页框中有内部碎片 没有外部碎片	无内部碎片 有外部碎片	无内部碎片 有外部碎片
操作系统须为每个进程维护一个页表，以说明每页对应的页框	操作系统须为每个进程维护一个页表，以说明每页对应的页框	操作系统须为每个进程维护一个段表，以说明每段中的加载地址和长度	操作系统须为每个进程维护一个段表，以说明每段中的加载地址和长度
操作系统须维护一个空闲页框列表	操作系统须维护一个空闲页框列表	操作系统须维护一个内存中的空闲空洞列表	操作系统须维护一个内存中的空闲空洞列表
处理器使用页号和偏移量来计算绝对地址	处理器使用页号和偏移量来计算绝对地址	处理器使用段号和偏移量来计算绝对地址	处理器使用段号和偏移量来计算绝对地址
进程运行时，它的所有页都必须都在内存中，除非使用了覆盖技术	进程运行时，并非所有页都须在内存页框中。仅在需要时才读入页 把一页读入内存可能需要把另一页写出到磁盘	进程运行时，其所有页都须在内存中，除非使用了覆盖技术	进程运行时，并非其所有段都须在内存中。仅在需要时才读入段 把一段读入内存可能需要把另外一段或几段写出到磁盘

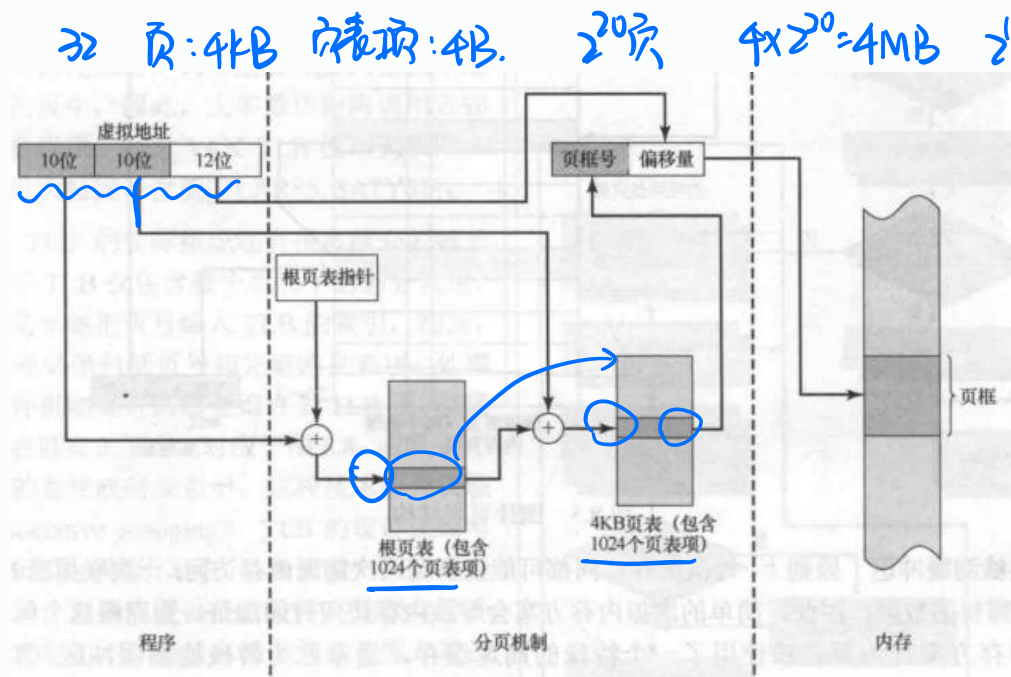
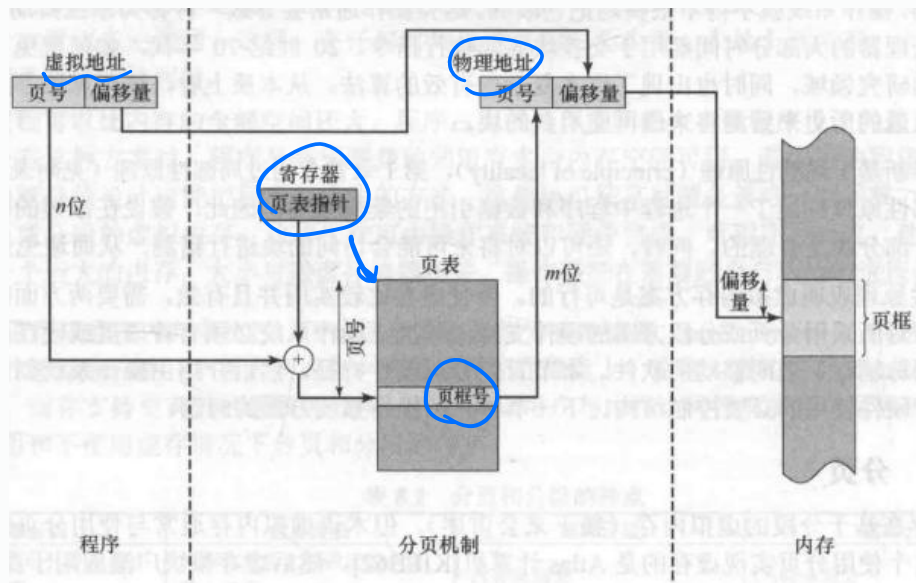
第8章 虚拟内存

- 分页：
 - 内存管理格式：
 - 地址转换：



P: 存在位、M: 修改位

$2^{12} = 4KB$



第8章 虚拟内存

- 分页：
 - TLB：保存最近用过的页表项

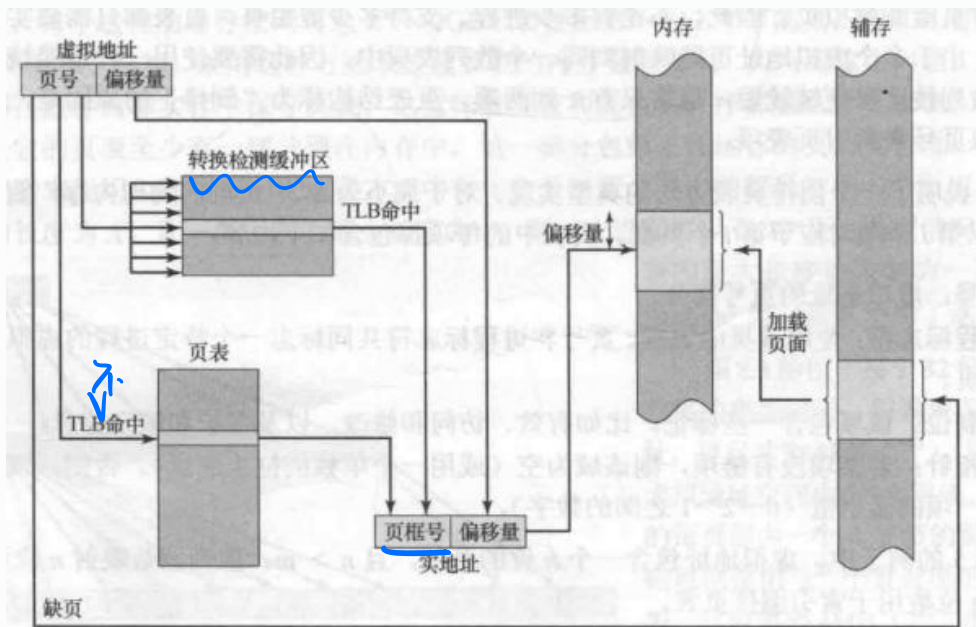


图 8.6 转换检测缓冲区的用法

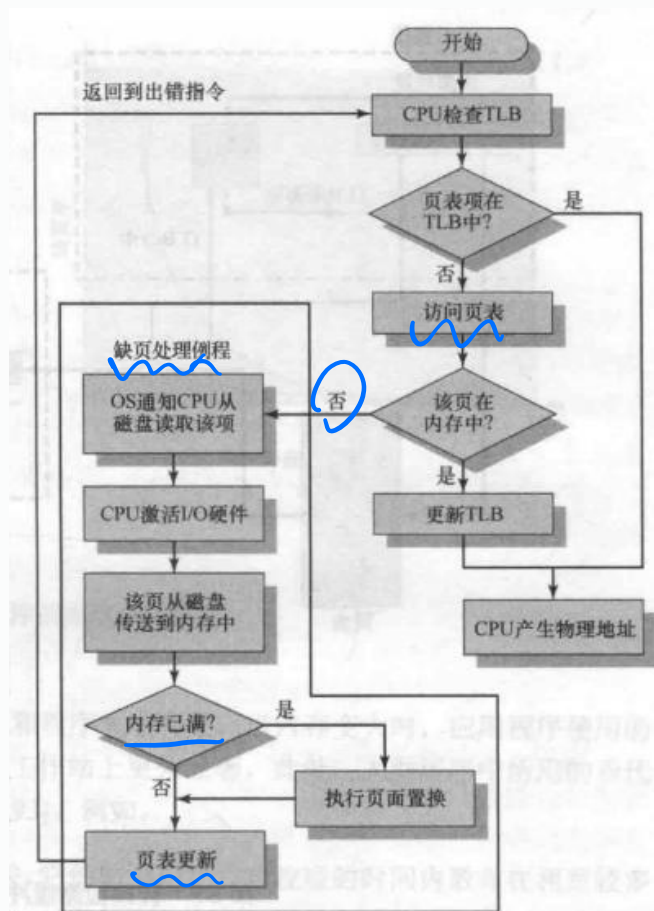


图 8.7 分页和转移检测缓冲区的操作

第8章 虚拟内存

- 分段：
 - 内存管理格式：
 - 地址转换：

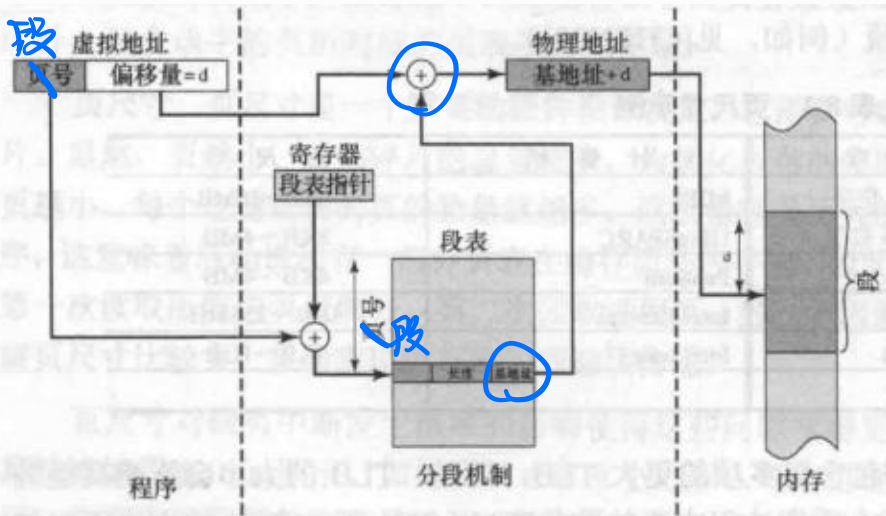
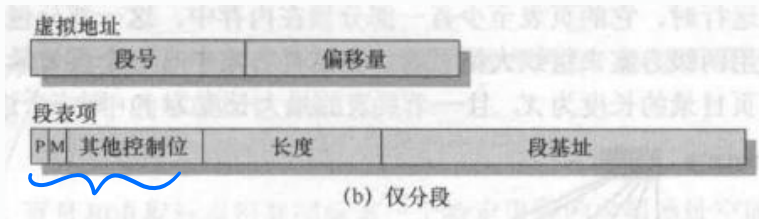
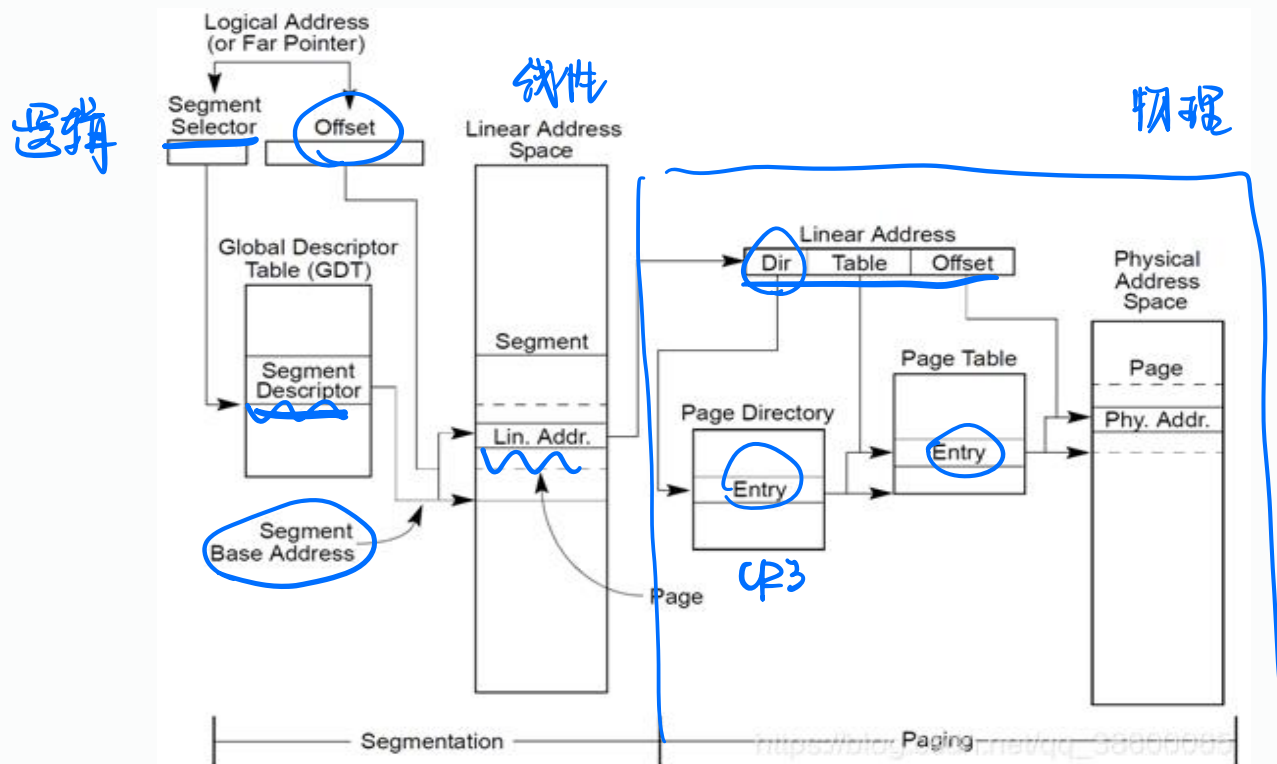


图 8.11 分段系统中的地址转换

第8章 虚拟内存

- 段页式：用户的地址空间被程序员划分为许多段，每段依次划分为许多固定大小的页
 - 地址转换：每个进程都使用一个段表和一些页表（多级）



第9章 单处理器调度

- 调度：决定了哪个进程须等待、哪个进程能继续运行。本质上属于队列管理问题，用于在排队环境中减少延迟并优化性能
- 调度算法：

① 先来先服务FCFS：也称先进先出FIFO（非抢占）

② 轮转RR：周期性产生时钟中断，出现中断时，当前正运行的进程放置到就绪队列中，基于FCFS选择下一个就绪队列运行（抢占）

③ 最短进程优先SPN：选择预计处理时间最短的进程（非抢占）

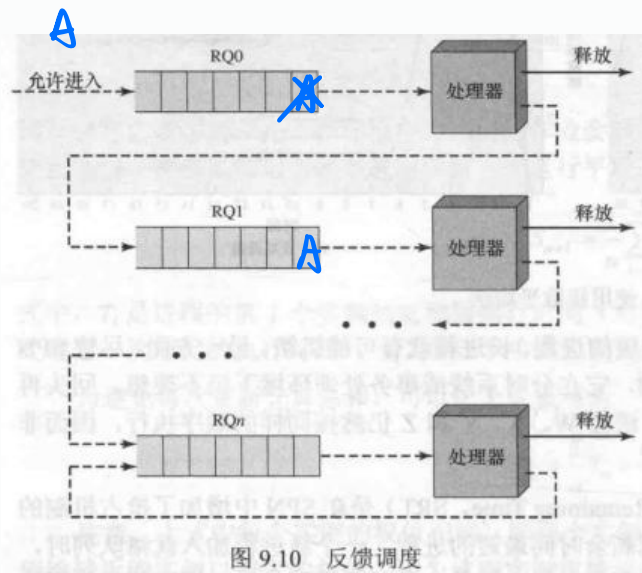
④ 最短剩余时间SRT：选择预期剩余时间最短的进程（抢占）

⑤ 最高响应比优先HRRN：选择响应比R最大的就绪进程（非抢占）

w 为等待处理器的时间， s 为预计服务的时间

⑥ 反馈法：调度基于抢占原则（按时间片）并使用动态优先机制

$$R = \frac{w + s}{s}$$



第9章 单处理器调度

q=2 AABBC EE ABBEE BEE.

A:8 C:5
B:13 E:15

(4/4)

- 例1: 考虑下面的进程, 在时刻0以1-4的顺序到达 (到达间隔可以忽略), 采用基于时间片轮转 (RR) 算法, 计算时间片 $q=1$ 和 $q=2$ 时平均周转时间

q=1 ABC E ABE AB E BE E.

A:8 E:15

B:13

C:3

(3/4)

进程名	到达顺序	执行时间
A	1	3
B	2	5
C	3	1
E	4	6

- 例2: 假设下列进程在所指定的时刻到达就绪队列等待执行, 每个进程将运行相应的区间时间。

①使用FCFS算法, 计算进程的平均周转时间; ②使用SPN算法, 计算进程的平均周转时间;

① 到 结

P1 0.0 8

P2 0.4 12

P3 1.0 14

P4 2.0 15

② 到 结

P1 0.0 8

P2 0.4 15

P3 1.0 11

P4 2.0 9

进程	到达时间	区间时间
P1	0.0	8
P2	0.4	4
P3	1.0	2
P4	2.0	1

$$\frac{1}{4}(8+11.6+13+13)=11.4$$

$$\frac{1}{4}(8+14.6+10+7)=9.9$$

第11章 I/O管理和磁盘调度

- 磁盘性能参数：

- ①寻道时间：将磁头臂移到指定磁道所需的时间
- ②旋转延迟：将磁盘的待访问地址区域旋转到读/写磁头可访问的位置所需的时间
- ③传输时间：向磁盘传送或从磁盘传送的时间，取决于磁盘的旋转速度。

总平均存取时间： $T_a = T_s + \frac{1}{2r} + \frac{b}{rN}$ ， b 表示要传送的字节数； N 表示一个磁道中的字节数； r 表示旋转速度

- 磁盘调度策略：磁盘请求队列

- ①先进先出FIFO：按顺序处理队列中的项目
- ②最短服务时间优先SSTF：选择使磁头臂从当前位置开始移动最少的磁盘I/O请求
- ③SCAN（~~X~~电梯算法）：要求磁头臂仅沿一个方向移动，并在途中满足所有未完成的请求，直到（这个方向上的最后一个磁道，或者在）这个方向上没有其他请求为止。接着反转服务方向，沿相反方向扫描
- ④C-SCAN（循环SCAN）：扫描限定在一个方向上。当访问到沿某个方向的最后一个磁道时，磁头臂返回到磁盘相反方向末端的磁道，并再次开始扫描。

第12章 文件管理

- 文件管理的要素：

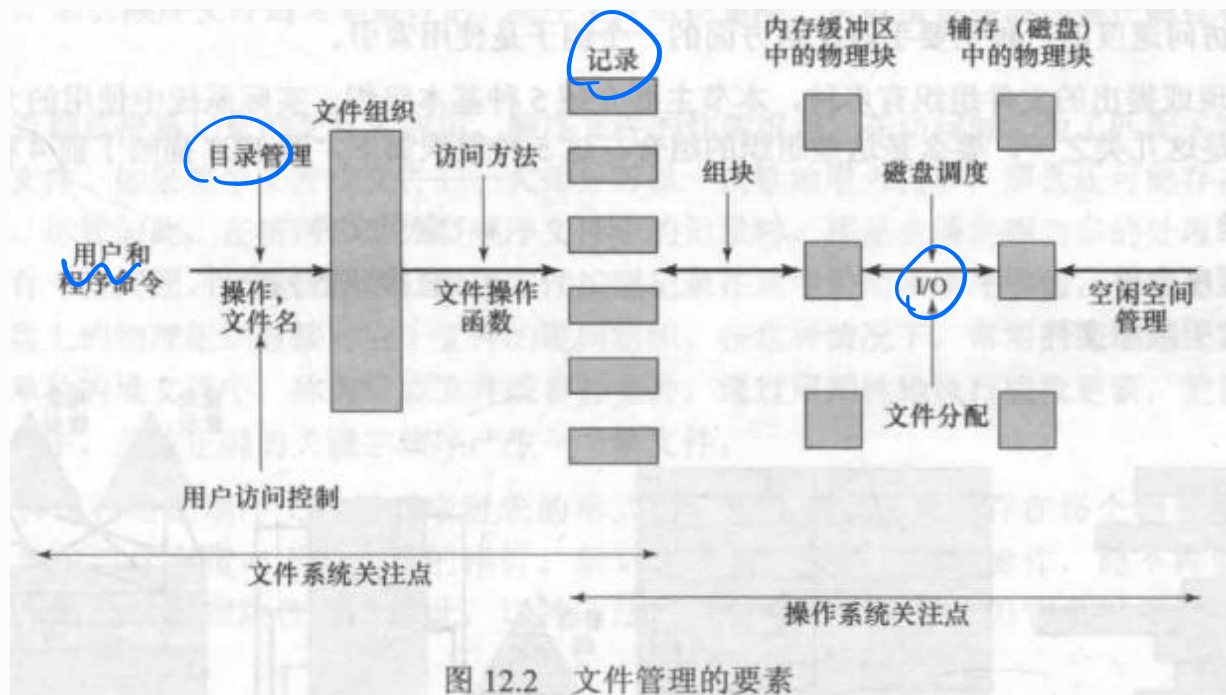


图 12.2 文件管理的要素

第12章 文件管理

- 文件目录：

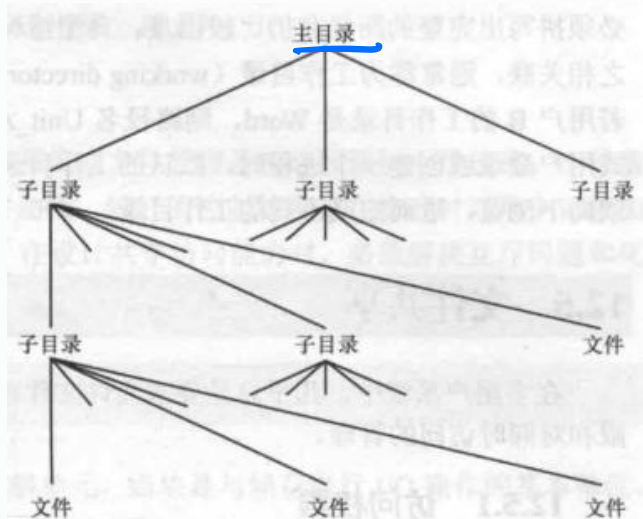


图 12.6 树状结构目录

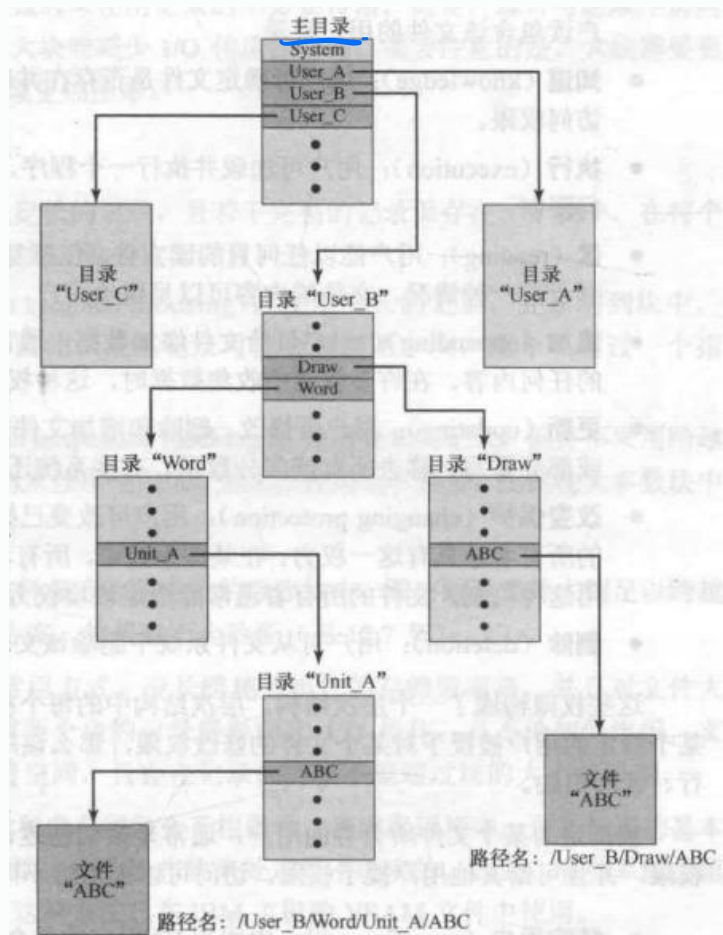


图 12.7 树状结构目录示例

第12章 文件管理

• 文件分配方法:

- ①连续分配: 创建文件时, 给文件分配一组连续的块
- ②链式分配: 基于单个块, 链中的每块都包含指向下一块的指针
- ③索引分配: 每个文件在文件分配表中都有一个一级索引, 分配给该文件的每个分区在索引中都有一个表项, 文件的索引保存在一个单独的块中



图 12.9 连续文件分配



图 12.11 链式分配

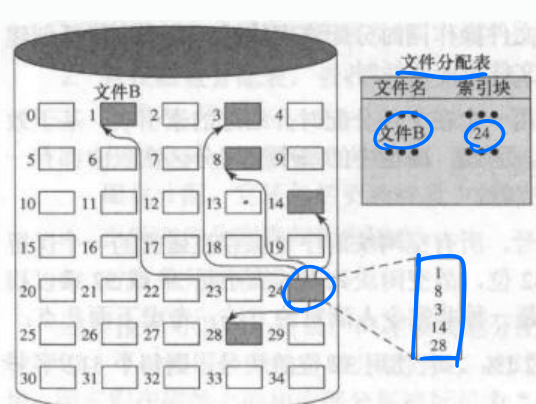
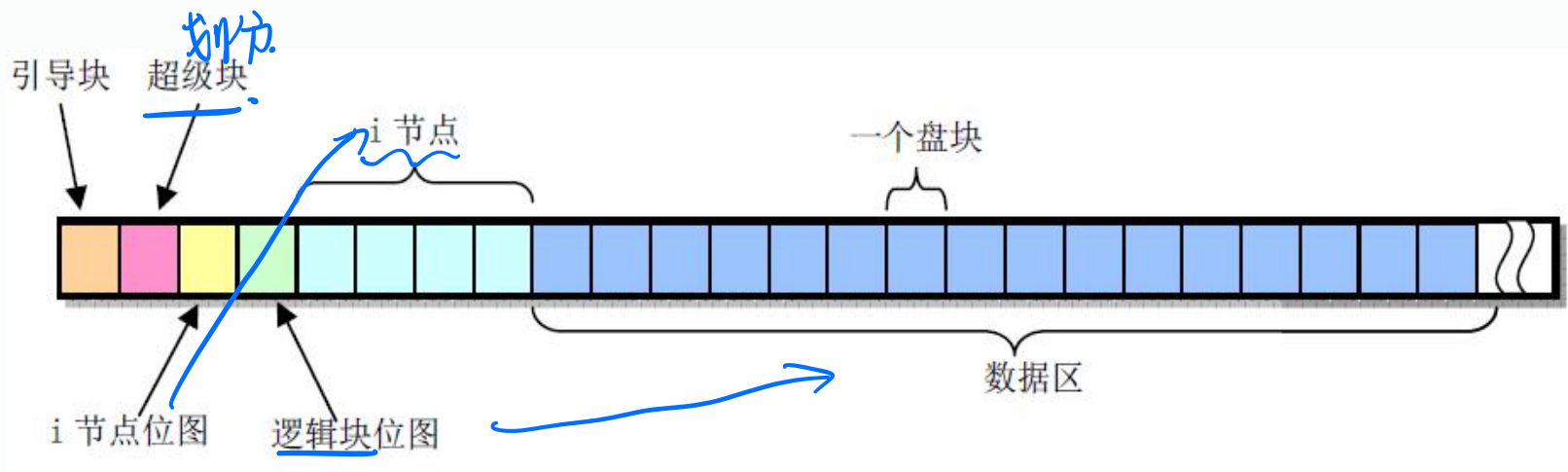


图 12.13 基于块的索引分配

第12章 文件管理

- Minix 文件系统：

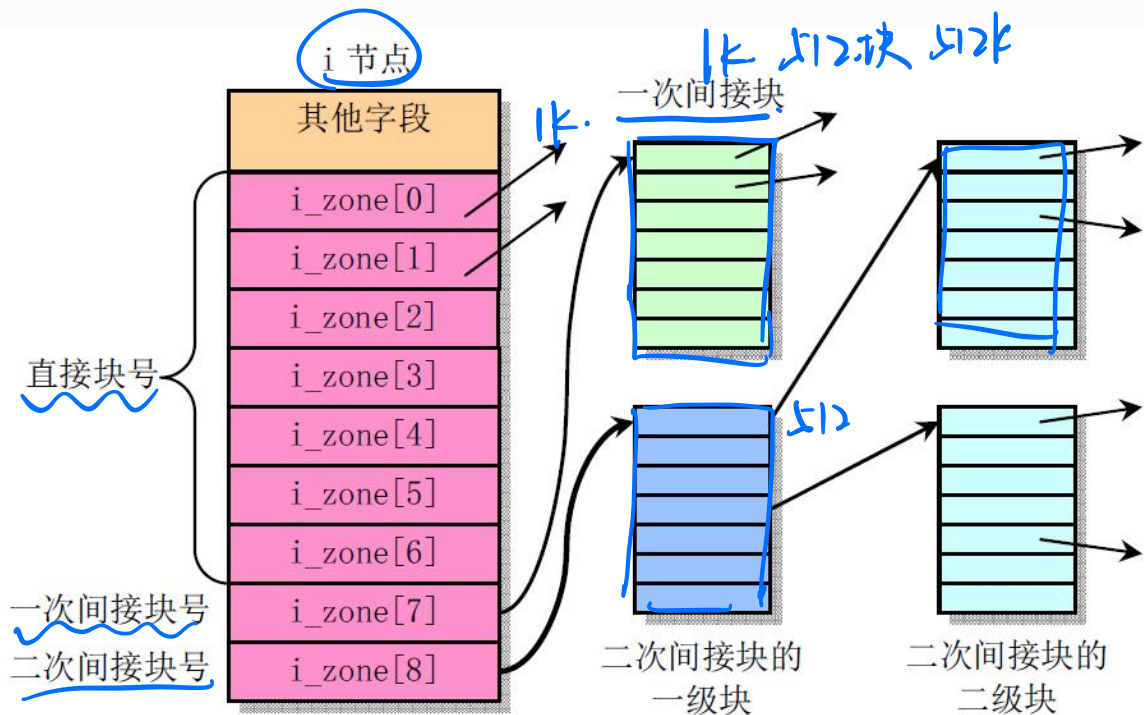


第12章 文件管理

7k + 512k + 512 * 512k.

- Minix 文件系统:

字段名称	数据类型	说明
<u>i_mode</u>	short	<u>文件的类型和属性 (rwx 位)</u>

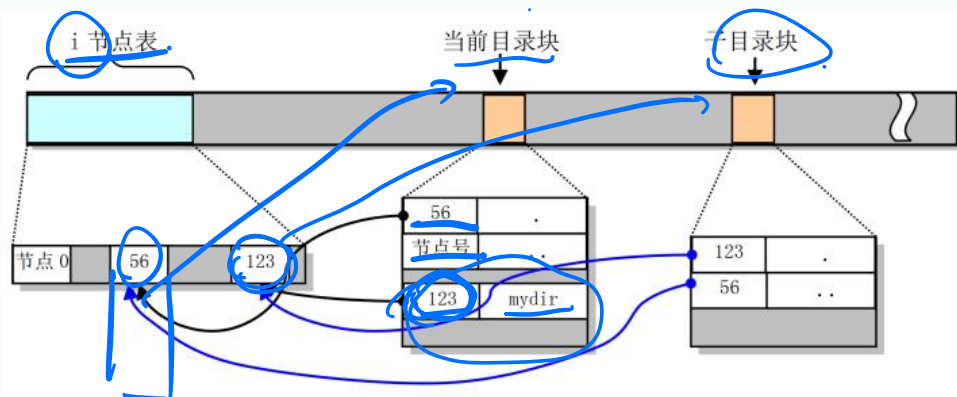


第12章 文件管理

• Minix 文件系统:

例1: 如何访问文件 /usr/bin/文件名3?

1. 根目录文件的i节点位置是固定的, 即第1号i节点
2. 1号i节点的数据块内容为根目录下的目录项列表
3. 通过该目录项列表匹配目录名 "usr"
4. 若找到, 则可得到文件 "/usr"的i节点号i1
5. 根据i1号i节点的数据块, 可以取得目录文件 "/usr"的内容, 即子目录usr的文件目录项列表
6. 通过该目录项列表匹配目录名 "bin"
7. 若找到, 则可得到文件 "/usr/bin"的i节点号i2
8. 根据i2号i节点的数据块, 可以取得目录文件 "/usr/bin"的内容, 即子目录bin的文件目录项列表
9. 通过该目录项列表匹配文件名 "文件名3"
10. 若找到, 则可得到文件 "/usr/bin/文件名3"的i节点号i3
11. 根据i3号i节点的数据块, 可以取得文件 "文件名3"的内容



第12章 文件管理

- Minix 文件系统:

例2: 如何删除文件/usr/bin/vi?

1. 根目录文件的i节点位置是固定的, 即第1号i节点
2. 1号i节点的数据块内容为根目录下的目录项列表
3. 通过该目录项列表匹配目录名 "usr"
4. 若找到, 则可得到文件 "/usr"的i节点号i1
5. 根据i1号i节点的数据块, 可以取得目录文件 "/usr"的内容, 即子目录usr的文件目录项列表
6. 通过该目录项列表匹配目录名 "bin"
7. 若找到, 则可得到文件 "/usr/bin"的i节点号i2
8. 根据i2号i节点的数据块, 可以取得目录文件 "/usr/bin"的内容, 即子目录bin的文件目录项列表
9. 通过该目录项列表匹配文件名 "vi"
10. 若找到, 则可得到文件 "/usr/bin/vi"的i节点号i3
11. 找到i3节点, 根据文件长度从i3节点中找到每个盘块的索引号, 将i3节点位图和逻辑块位图中相应位置0 (释放)

总结

- 假设当前操作系统采用**段页式**内存管理，并使用**二级页表**，页大小为**4K**，物理内存中已有多个进程，且实际物理内存的未分配空间为**14K**。现有一个编译好的程序test放在一个指定的路径下，其大小为**18K**，此程序的运行**至少需要两个时间片**。请结合你所学过的**进程管理**、**内存管理**、**文件管理**等操作系统知识，详细描述该程序从创建进程到运行结束，操作系统所做的主要操作有哪些。

(1) 创建进程控制块 (PCB)：为新进程分配一个PCB，用于**存储进程的状态**、程序计数器、寄存器集合、内存管理信息（如**页目录表指针**（CR3存储页目录表的物理地址））、优先级、I/O状态信息等。

(2) 分配内存空间：由于**程序大小为18K**，它将需要一个**4K的页目录表**和一个**4K的页表**

(3) 定位程序文件：查询文件分配表，得到根目录所在盘块，依次查询下一级目录所在盘块，最后查询到test程序文件所在的盘块。

(4) 加载程序到内存：操作系统将test程序**代码和数据**从磁盘加载到分配给内存中（一页）。

(5) 初始化进程上下文：设置程序计数器指向程序的起始地址，初始化所需的寄存器。

(6) 执行程序：操作系统将进程状态设置为**就绪态**，等待调度运行

(7) 执行程序时发生**缺页中断**：当test数据或代码对应页不在物理内存中时，触发**缺页中断**，从**用户态转到内核态**，操作系统执行相应的中断处理程序，若内存不足，操作系统使用**页面置换算法**决定哪个页面被换出，由于需要I/O，**该进程被阻塞**，直到完成I/O转为就绪态

(8) 执行程序时发生**时钟中断**：由于该运行该程序至少需要2个时间片，第一个时间片耗尽时，执行时钟中断，**该程序转为就绪态**，CPU转而执行其他进程

(9) 运行结束：更新进程控制块，标记进程为退出态。与作业相关的表和其他信息会临时被操作系统保留，辅助或支持程序提取所需的信息，然后操作系统就不再保留任何与该进程相关的数据，从系统中删除该进程。