
河南大学
操作系统
计算机学院

第 4 章

存储器管理

大纲

1. 4.5　分页存储管理方式
2. 4.6　分段存储管理方式
3. 本章作业

离散分配方式

- 连续分配存储管理方式产生的问题：
  ①要求连续的存储区 ②碎片问题
- 变连续分配为离散分配，允许将作业离散放到多个不相邻接的分区中。

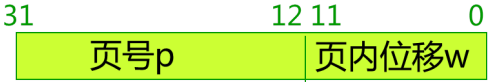
离散分配方式

1. 分页式存储管理：离散分配的基本单位是页
2. 分段式存储管理：离散分配的基本单位是段
3. 段页式存储管理：离散分配的基本单位是段、页

页面和物理块

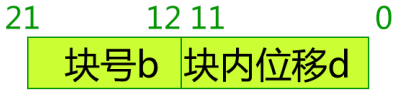
- 空间划分
  1. 将一个用户进程的地址空间（逻辑空间）划分成若干个大小相等的区域，称为页或页面，各页从 0 开始编号。
  2. 内存空间也分成若干个与页大小相等的区域，称为块（物理块）或页框（frame），同样从 0 开始编号。
- 内存分配
  - 在为进程分配内存时以块为单位，将进程中若干页装入到多个不相邻的块中，最后一页常装不满一块而出现页内碎片。

地址结构



逻辑地址

例：地址长为 32 位，其中 0-11 位为页内地址，即每页的大小为 2^{12}=4KB；
12-31 位为页号，地址空间最多允许有 2^{20} =1M 页。



物理地址

例：地址长为 22 位，其中 0-11 位为块内地址，即每块的大小为 2^{12}=4KB，与页相等；
12-21 位为块号，内存地址空间最多允许有 2^{10} =1K 块。

地址结构

- **已知逻辑地址求页号和页内地址**

  给定一个逻辑地址空间中的地址为 A，页面的大小为 L，则页号 P 和页内地址 d（从 0 开始编号）可按下式求得：

$$P = INT\left[\frac{A}{L}\right], d = [A] mod L$$

其中，INT 是整除函数，mod 是取余函数。

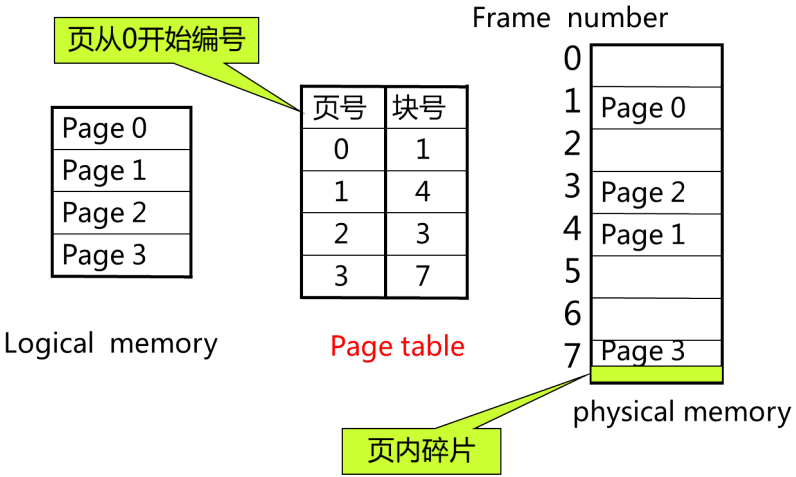
例：系统的页面大小为 1 KB，设 A = 2170 B，则由上式可以求得 P = 2，d = 122。

页表

- 为了便于在内存找到进程的每个页面所对应块，分页系统中为每个进程配置一张页表，进程逻辑地址空间中的每一页，在页表中都对应有一个页表项。
- 页表存放在内存中，属于进程的现场信息。
- 用途：①记录进程的内存分配情况 ②实现进程运行时的动态重定位。
- 访问一个数据需访问内存 2 次 (页表一次，内存一次)。
- 页表的基址及长度由页表寄存器给出。

页表始址	页表长度

页表


页从0开始编号
Page 0
Page 1
Page 2
Page 3
Logical memory
页号
块号
0
1
1
4
2
3
3
7
Page table
Frame number
0
1
2
3
4
5
6
7
Page 0
Page 2
Page 1
Page 3
physical memory
页内碎片

页面大小

- 若页面较小
  - 减少页内碎片和总的内存碎片，有利于提高内存利用率。
  - 每个进程页面数增多，使页表长度增加，占用内存较大。
  - 页面换进换出速度将降低。
- 若页面较大
  - 每个进程页面数减少，页表长度减少，占用内存较小。
  - 页面换进换出速度将提高。
  - 增加页内碎片，不利于提高内存利用率。

基本分页存储管理方式

- 在分页存储管理方式中，如果不具备页面对换功能，不支持虚拟存储器功能，这种存储管理方式称为纯分页或基本分页存储管理方式。
- 在调度作业运行时，必须将它的所有页面一次调入内存，但逻辑上连续的各个页所对应的内存块可以不连续。
- 特殊的固定分区 + 离散分配

地址变换机构

- 地址变换机构
  - 将用户地址空间中的逻辑地址变换为内存空间中的物理地址。
  - 实现逻辑地址向物理地址的转换（页号 ⇒ 块号）
  - 地址变换借助页表来完成。

地址变换机构种类

1. 基本的地址变换机构
2. 具有快表的地址变换机构

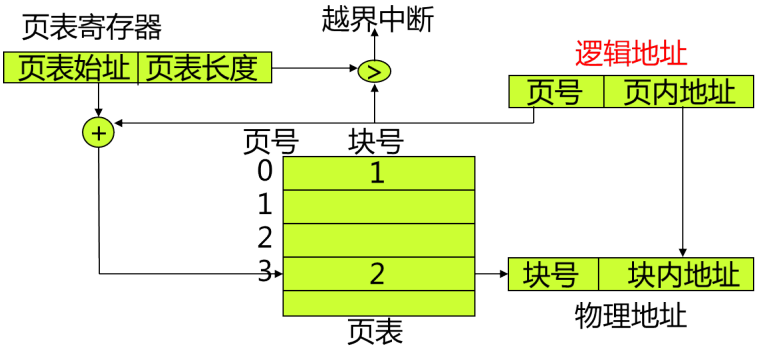
基本地址变换机构

- 地址变换借助页表来完成，页表驻留内存。
- 为了提高地址变换的速度，系统中设置一个页表寄存器 PTR (Page-Table Register)。
- 每个进程对应一页表，其信息（如长度、始址）放在 PCB 中，执行时将其装入页表寄存器。
- 在单处理机环境下，虽然系统中可以运行多个进程，但只需一个页表寄存器。

基本地址变换机构

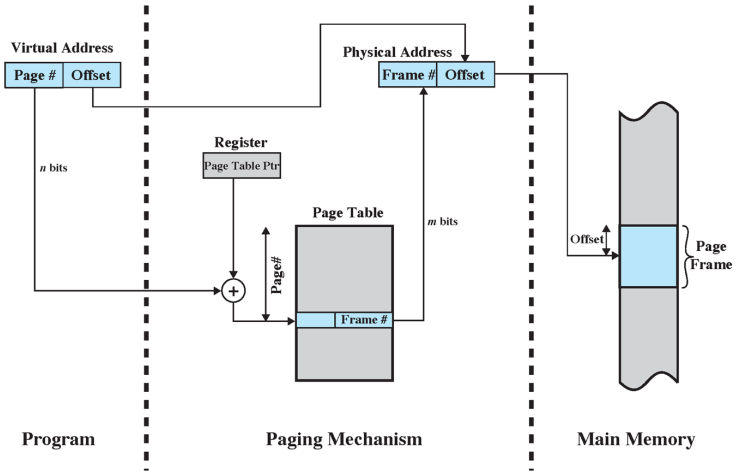
1. 当进程要访问某个逻辑地址中的数据时，分页地址变换机构会自动地将有效地址（相对地址）分为页号和页内地址两部分。
2. 将页号与页表长度进行比较，如果页号大于或等于页表长度，则表示本次所访问的地址已超越进程的地址空间，产生地址越界中断。
3. 将页表始址与页号和页表项长度的乘积相加，得到该表项在页表中的位置，于是可从中得到该页的物理块号，将之装入物理地址寄存器中。
4. 将有效地址寄存器中的页内地址送入物理地址寄存器的块内地址字段中。

基本地址变换机构



①逻辑地址: 把相对地址分为页号和页内地址两部分。②越界中断: 页号与页表长度做比较。
③页表定位：页表始址 + 页号 × 页表项长度。④查询页表：读出块号。
⑤物理地址：块号 + 块内地址。(块内地址 = 页内地址)

基本地址变换机构


Virtual Address
Page #
Offset
n bits
Register
Page Table Ptr
Page#
Page Table
m bits
Frame #
Physical Address
Frame #
Offset
Offset
Page
Frame
Program
Paging Mechanism
Main Memory

地址变换例题

例：若在一分页存储管理系统中，某作业的页表如下表所示，已知页面大小为 1024B，试将十进制逻辑地址 1011，2148，5012 转化为相应的物理地址。

页号	块号
0	2
1	3
2	1
3	6

地址变换例题

设页号为 P，页内位移为 W，逻辑地址为 A，内存地址为 M，页面大小为 L，则

P = int (A / L)

W = A mod L

对于逻辑地址 1011

P=int(1011/1024)=0

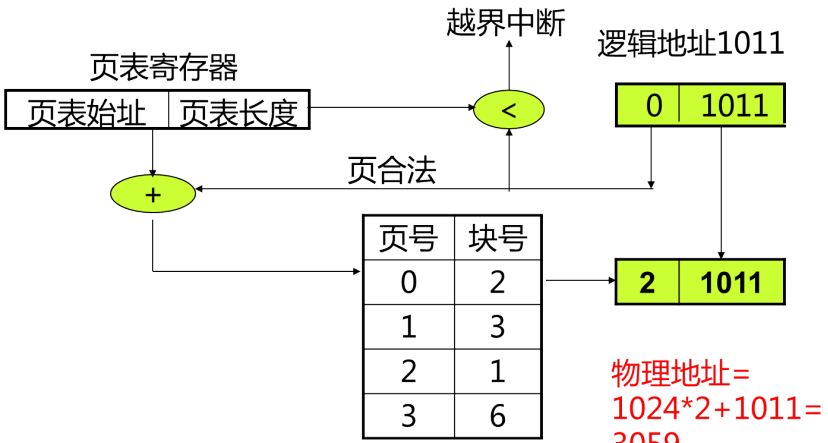
W=1011 mod 1024=1011

A=1011=(0,1011)

查页表第 0 页在第 2 块，所以物理地址为 M=1024*2+1011= 3059。

页号	块号
0	2
1	3
2	1
3	6

地址变换例题


越界中断
逻辑地址1011
页表寄存器
页表始址
页表长度
<
0
1011
+
页合法
页号
块号
0
2
1
3
2
1
3
6
2
1011
物理地址=
1024*2+1011=
3059

地址变换例题

对于逻辑地址为 2148

P= int (2148/1024)=2

W=2148 mod 1024=100

A=2148=(2,100)

查页表第 2 页在第 1 块，所以物理地址为

M=1024*1+100=1124。

对于逻辑地址 5012

P= int (5012/1024)=4

W=5012 mod 1024=916

页号超过页表长度，该逻辑地址非法。

页号	块号
0	2
1	3
2	1
3	6

地址变换例题

例：存储器的用户空间共有 32 个页面，每页 1KB，内存 16KB。假定某时刻系统为用户的第 0、1、2、3 页分别分配的物理块号为 5、10、4、7，试将逻辑地址 0A5C 和 093C 变换为物理地址。

地址变换例题

例：存储器的用户空间共有 32 个页面，每页 1KB，内存 16KB。假定某时刻系统为用户的第 0、1、2、3 页分别分配的物理块号为 5、10、4、7，试将逻辑地址 0A5C 和 093C 变换为物理地址。

解：逻辑地址为：页号 5 位（2^5=32），页内位移 10 位（2^{10}=1024）；物理地址为：物理块号 4 位（2^4=16），块内位移（2^{10}=1024）10 位。

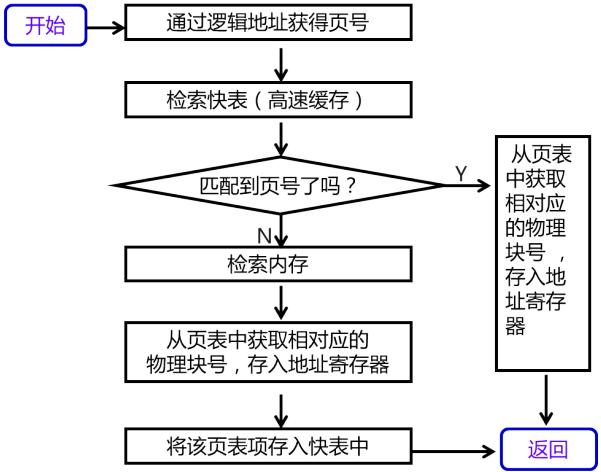
逻辑地址 0A5C 对应的二进制为：00010 1001011100，即逻辑地址 0A5C 的页号为 2，页内位移为 1001011100，由题意知对应的物理地址为：0100 1001011100 即 125C。

同理可求 093C 的物理地址为 113C。

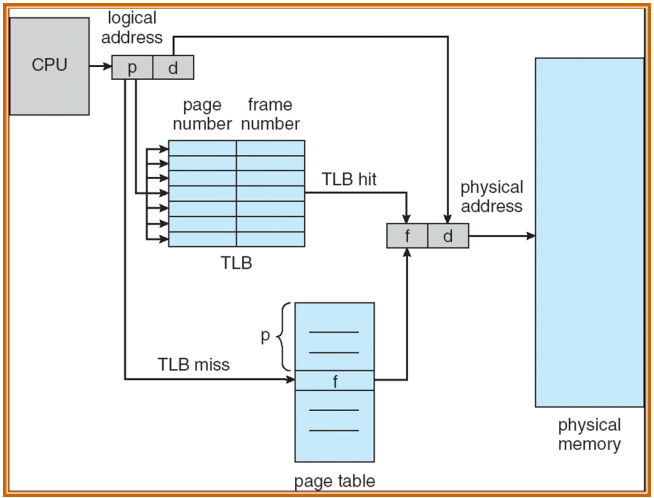
具有快表的地址变换机构

- 基本的地址变换机构存在的问题
  - 地址变换速度低（两次访问内存）
- 具有快表的地址变换机构
  - 目的：为提高地址变换速度。
  - 快表：又称为联想寄存器、联想存储器 (Associative Memory) 、IBM-TLB (Translation Lookaside Buffer)。
  - 快表是一种特殊的高速缓冲存储器（Cache），内容是页表中的一部分或全部内容。
  - CPU 产生逻辑地址的页号，首先在快表中寻找，若命中就找出其对应的物理块；若未命中，再到页表中找其对应的物理块，并将之复制到快表。若快表中内容满，则按某种算法淘汰某些页。

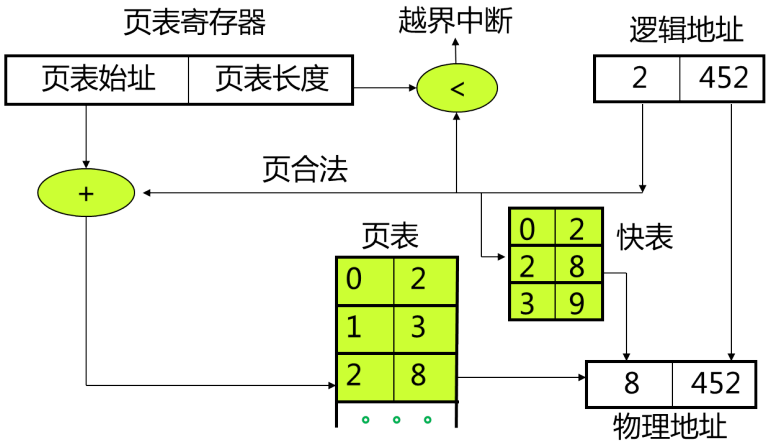
具有快表的地址变换机构



具有快表的地址变换机构



具有快表的地址变换机构



访问内存的有效时间

- 有效访问时间(Effective Access Time ,EAT) 是指从给定逻辑地址，经过地址变换，到在内存中找到对应物理地址单元并取出数据所用的总时间。
- 基本地址变换机构

  设 T_M 为内存的访问时间

$$EAT = 2T_M$$

- 具有快表的地址变换机构

  设 P_{TLB} 为快表的命中率， T_{TLB} 为快表的访问时间

$$EAT = P_{TLB} * (T_{TLB} + T_M) + (1 - P_{TLB}) * (T_{TLB} + 2T_M)$$

访问内存的有效时间

例：有一页式系统，其页表存放在内存中。（1）如果对内存的一次存取需要 100ns，试问实现一次页面访问的存取时间是多少？（2）如果有快表，对快表的一次存取需要 20ns, 平均命中率为 85%，试问此时的存取时间为多少？

访问内存的有效时间

例：有一页式系统，其页表存放在内存中。（1）如果对内存的一次存取需要 100ns，试问实现一次页面访问的存取时间是多少？（2）如果有快表，对快表的一次存取需要 20ns, 平均命中率为 85%，试问此时的存取时间为多少？

1. 页表放内存中，则实现一次页面访问需 2 次访问内存。所以实现一次页面访问的存取时间为：100ns*2=200ns

访问内存的有效时间

例：有一页式系统，其页表存放在内存中。（1）如果对内存的一次存取需要 100ns，试问实现一次页面访问的存取时间是多少？（2）如果有快表，对快表的一次存取需要 20ns, 平均命中率为 85%，试问此时的存取时间为多少？

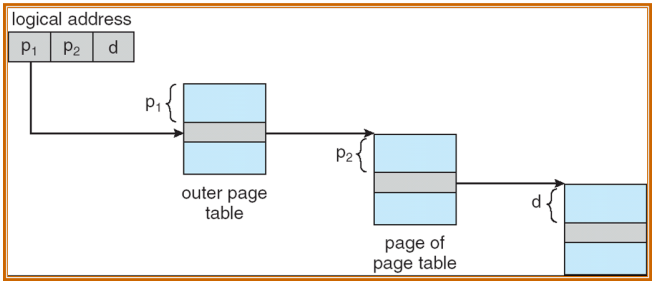
1. 页表放内存中，则实现一次页面访问需 2 次访问内存。所以实现一次页面访问的存取时间为：100ns*2=200ns
2. 系统有快表，则实现一次页面访问的存取时间为：0.85*(20ns+100ns)+(1-0.85)*(20ns+2*100ns)=135ns

多级页表

- 若逻辑地址空间很大 (2^{32}～2^{64})，则划分的页比较多，页表就很大，占用的存储空间大（要求连续），实现较困难。
- 例如，对于 32 位逻辑地址空间的分页系统，如果规定页面大小为 4 KB 即 2^{12} B，则在每个进程页表就由高达 2^{20} 页组成。设每个页表项占用一个字节，每个进程仅仅页表就要占用 1 MB 的内存空间。
- 解决问题的方法
  - 动态调入页表: 只将当前需用的部分页表项调入内存，其余的需用时再调入。
  - 多级页表

两级页表

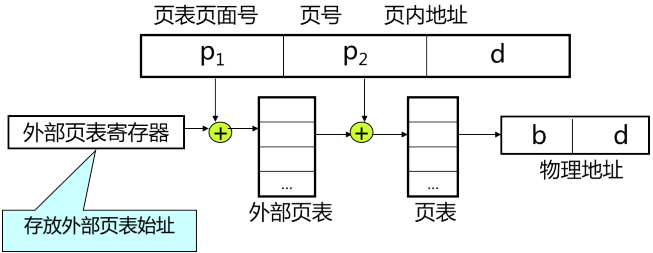
- 将页表再进行分页，离散地将各个页表页面存放在不同的物理块中，同时也再建立一张外部页表用以记录页表页面对应的物理块号。
- 正在运行的进程，必须把外部页表调入内存，而动态调入内部页表。只将当前所需的一些内层页表装入内存，其余部分根据需要再陆续调入。



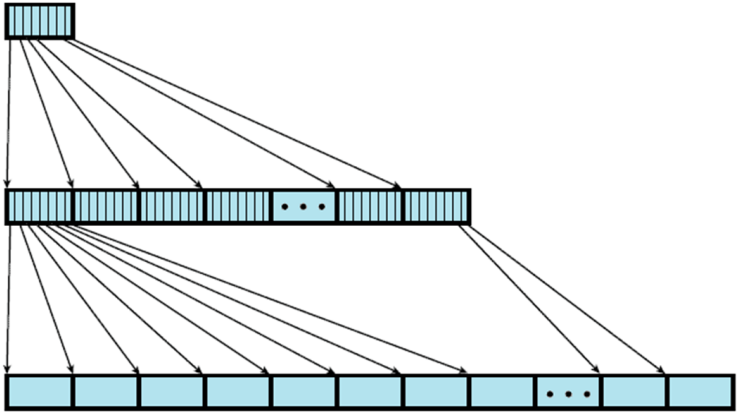
两级页表

- 逻辑地址

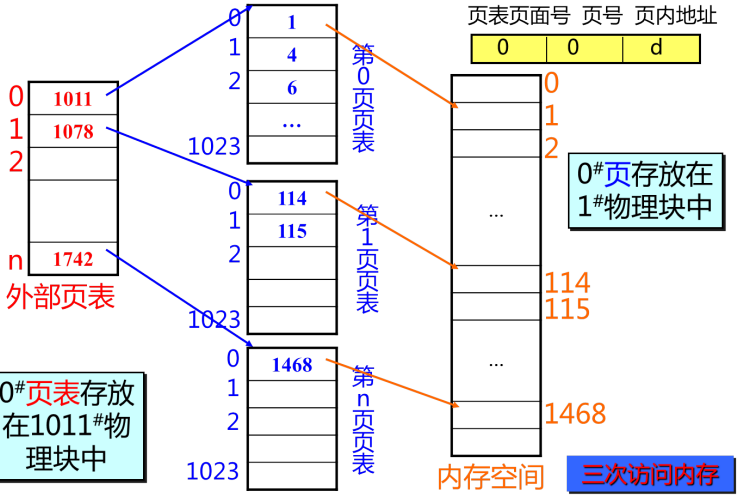
- 地址变换



两级页表



两级页表


页表页面号　页号　页内地址
0
0
d
0
1
2
n
1011
1078
1742
外部页表
0
1
2
1023
1
4
6
…
第0页页表
0
1
2
1023
114
115
第1页页表
0
1
2
1023
1468
第n页页表
0
1
2
…
114
115
…
1468
内存空间
0#页存放在1#物理块中
0#页表存放在1011#物理块中
三次访问内存

多级页表

- 将外层页表再进行分页，也将各外层页表页面离散地存放在不同的物理块中，再利用第 2 级的外层页表来记录它们之间的对应的关系。
- 逻辑地址

外层页表页面号	页表页面号	页号	页内偏移地址
p_1	p_2	p_3	d

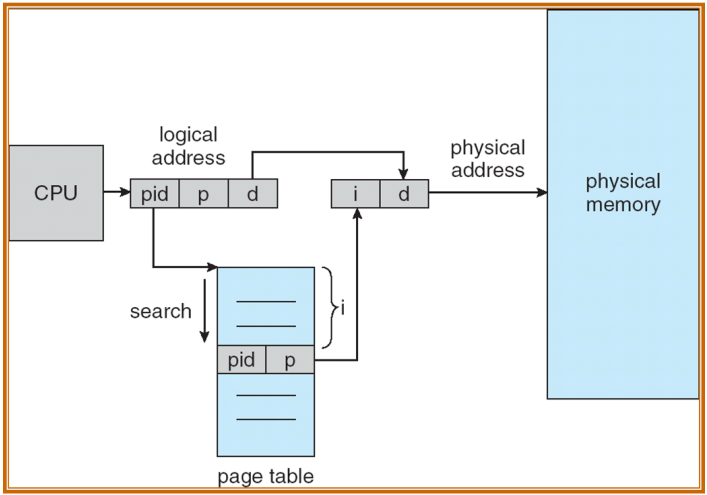
反置页表

- 对于 64 位逻辑地址空间的分页系统，如果规定页面大小为 4 KB 即 2^{12}B，则在每个进程页表就由高达 2^{52} 页组成。设表中每项为 8byte，则需 8 $*2^{52}$ =2^{55} =32768 TB 的内存空间。
- 解决问题的方法
  - 动态调入页表
  - 多级页表
  - 反置页表（ Inverted page tables ）

反置页表

- 一般页表的表项是按页号进行排序，页表项中的内容是物理块号。
- 反置页表是为每一个物理块设置一个页表项并按物理块号排序，其中的内容是页号 P 及隶属进程标志符 pid。
- 利用反置页表进行地址变换
  - 用进程标志符和页号去检索反置页表。
  - 如果检索完整个页表未找到与之匹配的页表项，表明此页此时尚未调入内存，对于具有请求调页功能的存储器系统产生请求调页中断，若无此功能则表示地址出错。
  - 如果检索到与之匹配的表项，则表项的序号 i 便是该页的物理块号，将该块号与页内地址一起构成物理地址。

反置页表

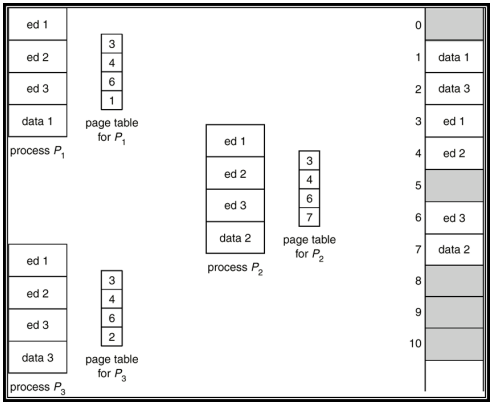

CPU
logical address
pid
p
d
i
d
physical address
physical memory
search
i
pid
p
page table

反置页表

- 反置页表可以有效地减少页表占用的内存，但反置页表中只包含已经调入内存的页面，未包含那些未调入内存的各个进程的页面，因此必须为每个进程建立一个外部页表 (External Page Table)。发现页面不在内存时才访问外部页表。
- 外部页表存放各页在外存中的物理位置。通过外部页表可将所需要的页面调入内存。
- 反置页表中是为每一个物理块设置一个页表项，通常页表项的数目也很大，通常又利用 Hash 表来检索。

页的共享

各进程把需要共享的数据/程序的相应页指向相同物理块。



页的共享与保护

- 页的保护
  - 页式存储管理系统提供了两种方式：
    ①地址越界保护
    ②在页表中设置保护位（定义操作权限：只读，读写，执行等）
- 共享带来的问题
  - 若共享数据与不共享数据划在同一块中，则：
    有些不共享的数据也被共享，不易保密。
    计算共享数据的页内位移较困难。
- 实现数据共享的最好方法：分段存储管理。

分段存储管理方式的引入

分段存储管理方式的引入是为了满足用户的要求。

1. 方便编程：通常一个作业是由多个程序段和数据段组成的，一般情况下，用户希望按逻辑关系对作业分段，并能根据名字来访问程序段和数据段。
2. 信息共享：

①共享是以信息的逻辑单位为基础的。页是存储信息的物理单位，段却是信息的逻辑单位。

②页式管理中地址空间是一维的，主程序，子程序都顺序排列，共享公用子程序比较困难，一个共享过程可能需要几十个页面。

分段存储管理方式的引入

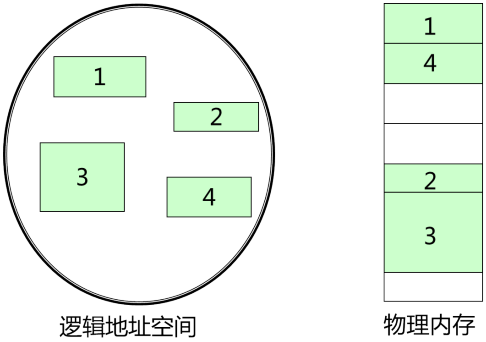
3 信息保护：页式管理中，一个页面中可能装有 2 个不同的子程序段的指令代码，不能通过页面共享实现共享一个逻辑上完整的子程序或数据块。段式管理中，可以以信息的逻辑单位进行保护。

分段存储管理方式的引入

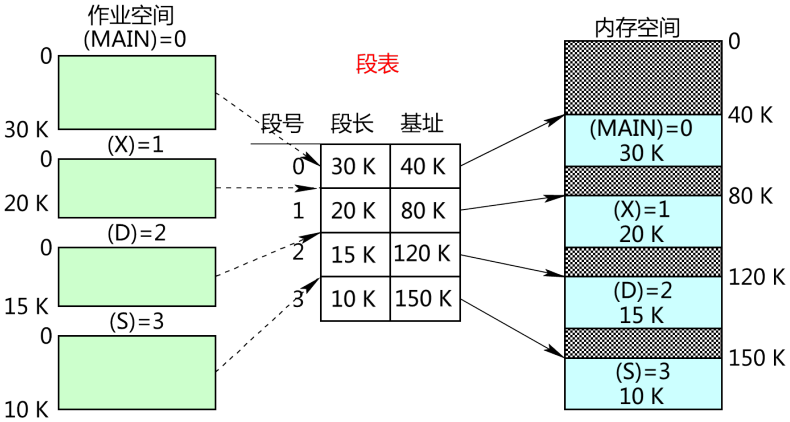
4. 动态增长：实际应用中，某些段（数据段）会不断增长，前面的存储管理方法均难以实现。
5. 动态链接：动态链接在程序运行时才把主程序和要用到的目标程序（程序段）链接起来。

分段系统的空间划分

- 将用户作业的逻辑地址空间划分成若干个大小不等的段（由用户根据逻辑信息的相对完整来划分）。各段有段名（常用段号代替），首地址为 0。



利用段表实现地址映射


作业空间
(MAIN)=0
0
30 K
(X)=1
0
20 K
(D)=2
0
15 K
(S)=3
0
10 K
段表
段号
段长
基址
0
30 K
40 K
1
20 K
80 K
2
15 K
120 K
3
10 K
150 K
内存空间
0
40 K
(MAIN)=0
30 K
80 K
(X)=1
20 K
120 K
(D)=2
15 K
150 K
(S)=3
10 K

段表

- 段表记录了段与内存位置的对应关系。
- 段表保存在内存中。
- 段表的基址及长度由段表寄存器给出。

段表始址	段表长度

- 访问一个字节的数据/指令需访问内存两次 (段表一次，内存一次)。
- 逻辑地址由段和段内地址组成。

段号	段内地址

例：采用段式存储管理的系统中，若地址用 24 位表示，其中 8 位表示段号，则允许段的最大长度是（ ）

A. 2^{24}　　B. 2^{16}　　C. 2^{8}　　D. 2^{32}

段表

- 段表记录了段与内存位置的对应关系。
- 段表保存在内存中。
- 段表的基址及长度由段表寄存器给出。

段表始址	段表长度

- 访问一个字节的数据/指令需访问内存两次 (段表一次，内存一次)。
- 逻辑地址由段和段内地址组成。

段号	段内地址

例：采用段式存储管理的系统中，若地址用 24 位表示，其中 8 位表示段号，则允许段的最大长度是（ ）

A.2^{24}　　B. 2^{16}　　C. 2^{8}　　D. 2^{32}

B

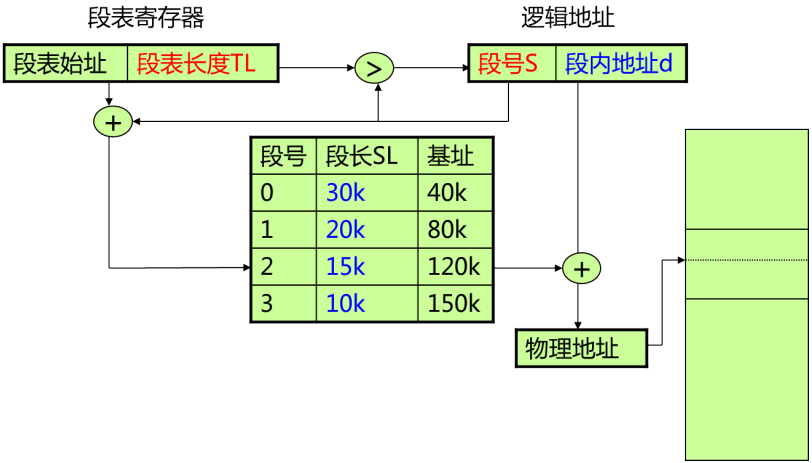
分段系统

- 在为作业分配内存时以段为单位，分配一段连续的物理地址空间，段间不必连续。
- 分页管理中，作业地址空间是一维的，逻辑地址是的线性地址。
- 分段管理中，整个作业的地址空间由于是分成多个段，因而是二维的，其逻辑地址由段号和段内地址所组成。

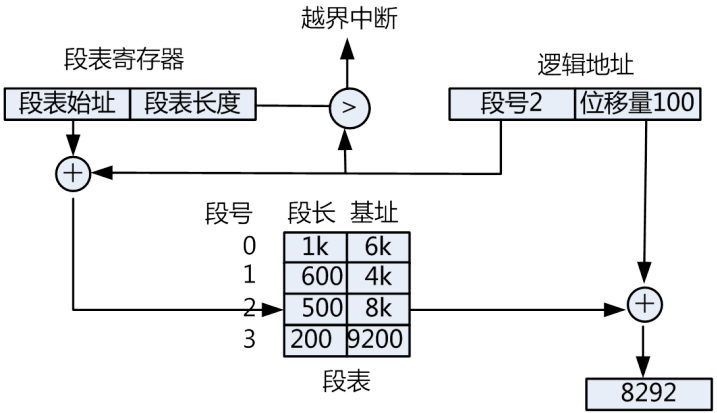
地址变换机构

- 系统将逻辑地址中的段号 S 与段表长度 TL 进行比较。
  - 若 S>TL，表示段号太大，是访问越界，于是产生越界中断信号。
  - 若未越界，则根据段表的始址和该段的段号，计算出该段对应段表项的位置，从中读出该段在内存的始址。
- 再检查段内地址 d，是否超过该段的段长 SL。
  - 若超过，即 d >SL，同样发出越界中断信号。
  - 若未越界，则将该段的基址与段内地址 d 相加，即可得到要访问的内存物理地址。

地址变换机构


段表寄存器
逻辑地址
段表始址
段表长度TL
>
段号S
段内地址d
+
段号
段长SL
基址
0
30k
40k
1
20k
80k
2
15k
120k
3
10k
150k
+
物理地址

地址变换机构



分段系统地址变换机构

分段地址变换例题

例：在一个段式存储管理系统中，其段表为：

段号	内存起始地址	段长
0	210	500
1	2350	20
2	100	90
3	1350	590
4	1938	95

0	430
2	120

试求右表中逻辑地址对应的物理地址是什么？

分段地址变换例题

例：在一个段式存储管理系统中，其段表为：

段号	内存起始地址	段长
0	210	500
1	2350	20
2	100	90
3	1350	590
4	1938	95

0	430
2	120

试求右表中逻辑地址对应的物理地址是什么？

解：逻辑地址 | 0 | 430 | 对应的物理地址为：210+430=640。

逻辑地址 | 2 | 120 | 段内地址 120> 段长 90，所以该段为非法段。

分段地址变换例题

例：某段式存储管理系统中，有一作业的段表（SMT）如下表所示，求逻辑地址 [0，65]，[1，55]，[2，90]，[3，20] 对应的主存地址（按十进制）。（其中方括号中的第一个元素为段号，第二个元素为段内地址）

段号	段长	起始地址	状态
0	200	600	1
1	50	850	1
2	100	1000	1
3	150	—	0

状态位：是否已调入内存

分段地址变换例题

例：某段式存储管理系统中，有一作业的段表（SMT）如下表所示，求逻辑地址 [0，65]，[1，55]，[2，90]，[3，20] 对应的主存地址（按十进制）。（其中方括号中的第一个元素为段号，第二个元素为段内地址）

段号	段长	起始地址	状态
0	200	600	1
1	50	850	1
2	100	1000	1
3	150	—	0

状态位：是否已调入内存

逻辑地址 [0，65] 对应的主存地址为 600 + 65 = 665。

逻辑地址 [1，55] 段内地址超过段长，产生段地址越界中断。

逻辑地址 [2，90] 对应的主存地址为 1000 + 90 = 1090。

逻辑地址 [3，20] 因为状态位为 0，该段在辅存中，缺段中断。

信息共享

分段系统的一个突出优点，是易于实现段的共享，对段的保护也十分简单。

例：一个多用户系统，可同时接纳 40 个用户，都执行一个文本编辑程序(Text Editor)。如果文本编辑程序有 160 KB 的代码和另外 40 KB 的数据区，如果不共享，则总共需有 8 MB 的内存空间来支持 40 个用户。

- 可重入代码(Reentrant Code) 又称为“纯代码”(Pure Code)，是一种允许多个进程同时访问的代码。
- 为使各个进程所执行的代码完全相同，绝对不允许可重入代码在执行中有任何改变。因此，可重入代码是一种不允许任何进程对它进行修改的代码。

信息共享

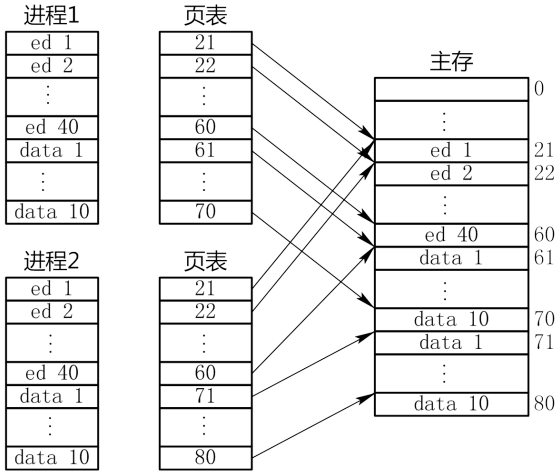
- 如果 160 KB 的代码是可重入的，则无论是在分页系统还是在分段系统中，该代码都能被共享。
- 在内存中只需保留一份文本编辑程序的副本，此时所需的内存空间仅为 1760 KB(40×40+160)，而不是 (160+40)×40= 8000 KB 。

分页系统的共享

假定每个页面的大小为 4 KB，160 KB 的代码将占用 40 个页面，数据区占 10 个页面。

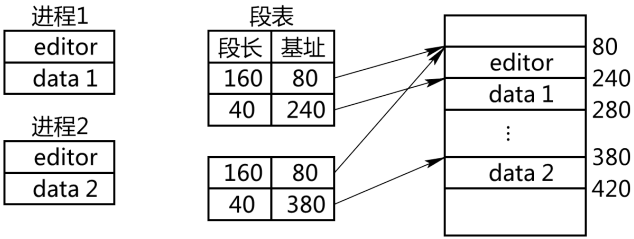
为实现代码的共享，应在每个进程的页表中都建立 40 个页表项，它们的物理块号是 21 ~ 60。

在每个进程的页表中，数据区页表项的物理块号分别是 61 ~ 70、71 ~ 80 等。



分段系统的共享

- 在分段系统中，实现共享容易得多，只需在每个进程的段表中为文本编辑程序设置一个段表项。



分页与分段的主要区别

- 页是信息的物理单位，分页仅仅是由于系统管理的需要，对用户透明的。段是信息的逻辑单位，分段的目的是为了能更好的满足用户的需要。
- 页的大小固定且由系统确定，把逻辑地址划分为页号和页内地址两部分。段的长度却不固定，决定于用户所编写的程序。
- 分页的作业地址空间是一维的，分段的作业地址空间是二维的。
- 页和段都有存储保护机制。但存取权限不同：段有读、写和执行三种权限；而页只有读和写两种权限。

分页与分段的主要区别

	页式存储管理	段式存储管理
目的	实现非连续分配， 解决碎片问题	更好地满足用户需要
信息单位	页（物理单位）	段（逻辑单位）
大小	固定（由系统定）	不定（由用户程序定）
内存分配单位	页	段
作业地址空间	一维	二维
优点	有效解决了碎片问题（没有外碎片，每个内碎片不超过页大小）；有效提高内存的利用率；程序不必连续存放。	更好地实现数据共享与保护；段长可动态增长；便于动态链接

分页与分段的主要区别

	页式存储管理	段式存储管理
目的	实现非连续分配， 解决碎片问题	更好地满足用户需要
信息单位	页（物理单位）	段（逻辑单位）
大小	固定（由系统定）	不定（由用户程序定）
内存分配单位	页	段
作业地址空间	一维	二维
优点	有效解决了碎片问题（没有外碎片，每个内碎片不超过页大小）；有效提高内存的利用率；程序不必连续存放。	更好地实现数据共享与保护；段长可动态增长；便于动态链接

二者优点的结合：段页式存储管理

段页式存储管理的基本原理

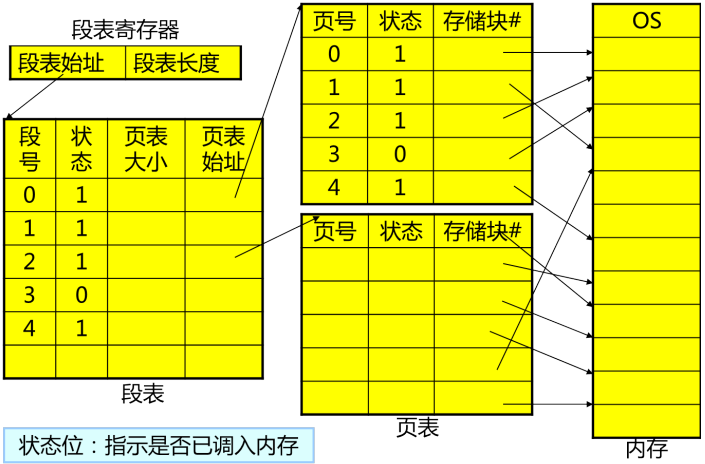
- 段页式存储管理是分段和分页原理的结合，即先将用户程序分成若干个段（段式），并为每一个段赋一个段名，再把每个段分成若干个页（页式）。
- 其地址结构由段号、段内页号、及页内位移三部分所组成。

段号（S）	段内页号（P）	页内地址（W）

段页式存储管理的基本原理

- 系统中设段表和页表，均存放于内存中。读一字节的指令或数据须访问内存三次。为提高执行速度可增设高速缓冲寄存器。
- 每个进程一张段表，每个段一张页表。
- 段表含段号、页表始址和页表长度。页表含页号和块号。

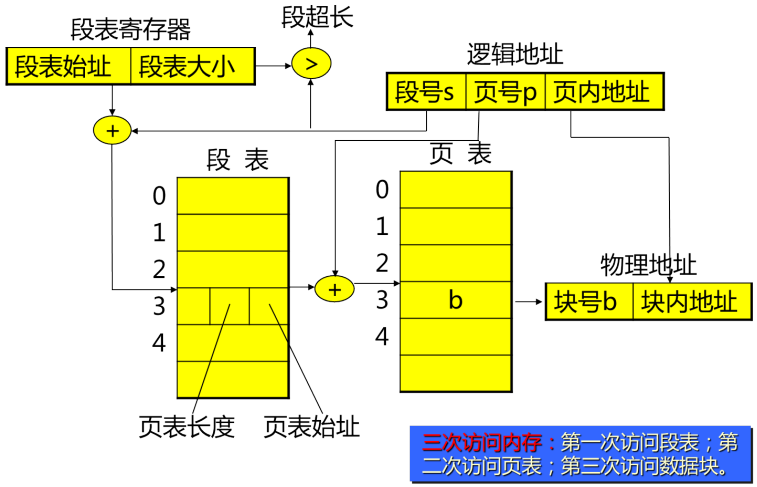
利用段表和页表实现地址映射



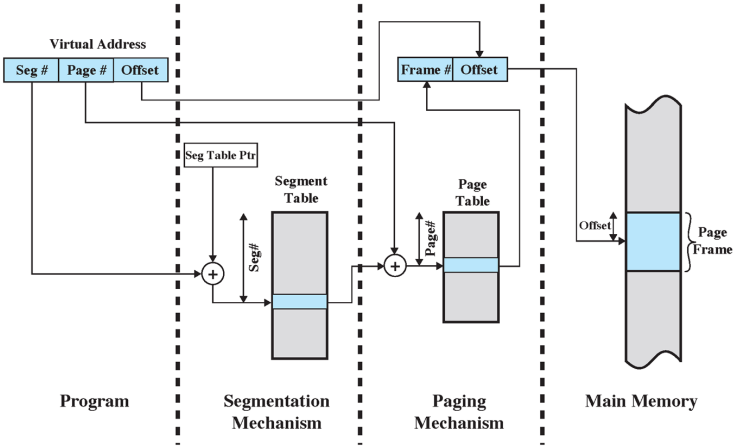
段页式存储管理的地址变换

- 从 PCB 中取出段表始址和段表长度，装入段表寄存器。
- 将段号与段表长度进行比较，若段号大于或等于段表长度，产生越界中断。
- 利用段表始址与段号得到该段表项在段表中的位置。取出该段的页表始址和页表长度。
- 将页号与页表长度进行比较，若页号大于或等于页表长度，产生越界中断。
- 利用页表始址与页号得到该页表项在页表中的位置。
- 取出该页的物理块号，与页内地址拼接得到实际的物理地址。

段页式存储管理的地址变换



段页式存储管理的地址变换



课堂练习

1. 分区管理要求对每一个作业都分配（　　）的内存单元。
   - A、地址连续　　B、地址不连续
   - C、连续块　　D、不连续块
2. （　　）存储管理方式提供二维地址结构。
   - A、固定分区　　B、分页
   - C、分段　　D、可变分区
3. 下面除哪一个外都是不连续的内存分配方法（　　）。
   - A、页式　　B、段式
   - C、可变分区　　D、段页式

课堂练习

1. 分区管理要求对每一个作业都分配（　　）的内存单元。
   - A、地址连续　　B、地址不连续
   - C、连续块　　D、不连续块
2. （　　）存储管理方式提供二维地址结构。
   - A、固定分区　　B、分页
   - C、分段　　D、可变分区
3. 下面除哪一个外都是不连续的内存分配方法（　　）。
   - A、页式　　B、段式
   - C、可变分区　　D、段页式

1A　2C　3C

课堂练习

4 段页式存储管理汲取了页式管理和段式管理的长处，其实现原理结合了页式和段式管理的基本思想，即（　　）。

- A、用分段方法来分配和管理物理存储空间，用分页方法来管理用户地址空间。
- B、用分段方法来分配和管理用户地址空间，用分页方法来管理物理存储空间。
- C、用分段方法来分配和管理主存空间，用分页方法来管理辅存空间。
- D、用分段方法来分配和管理辅存空间，用分页方法来管理主存空间。

课堂练习

4 段页式存储管理汲取了页式管理和段式管理的长处，其实现原理结合了页式和段式管理的基本思想，即（　　）。

- A、用分段方法来分配和管理物理存储空间，用分页方法来管理用户地址空间。
- B、用分段方法来分配和管理用户地址空间，用分页方法来管理物理存储空间。
- C、用分段方法来分配和管理主存空间，用分页方法来管理辅存空间。
- D、用分段方法来分配和管理辅存空间，用分页方法来管理主存空间。

4B

课堂练习

5 在可变式分区分配方案中，某一作业完成后，系统收回其主存空间，并与相邻空闲区合并，为此需修改空闲区表，造成空闲区数减 1 的情况是（　　）。

- A、无上邻空闲区，也无下邻空闲区。
- B、有上邻空闲区，但无下邻空闲区。
- C、有下邻空闲区，但无上邻空闲区。
- D、有上邻空闲区，也有下邻空闲区。

6 地址重定位的结果是得到（　　）。

- A、源程序　　B、编译程序
- C、目标程序　　D、执行程序

课堂练习

5 在可变式分区分配方案中，某一作业完成后，系统收回其主存空间，并与相邻空闲区合并，为此需修改空闲区表，造成空闲区数减 1 的情况是（　　）。

- A、无上邻空闲区，也无下邻空闲区。
- B、有上邻空闲区，但无下邻空闲区。
- C、有下邻空闲区，但无上邻空闲区。
- D、有上邻空闲区，也有下邻空闲区。

6 地址重定位的结果是得到（　　）。

- A、源程序　　B、编译程序
- C、目标程序　　D、执行程序

5D　6D

课堂练习（判断正误）

1. 静态重定位后不可能使用紧缩技术解决碎片问题。

课堂练习（判断正误）

1. 静态重定位后不可能使用紧缩技术解决碎片问题。 ✓
2. 在逻辑地址空间中通常采用不连续编址方式。

课堂练习（判断正误）

1. 静态重定位后不可能使用紧缩技术解决碎片问题。 ✓
2. 在逻辑地址空间中通常采用不连续编址方式。 ×
3. 虚拟存储空间实际上就是辅存空间。

课堂练习（判断正误）

1. 静态重定位后不可能使用紧缩技术解决碎片问题。 ✓
2. 在逻辑地址空间中通常采用不连续编址方式。 ×
3. 虚拟存储空间实际上就是辅存空间。 ×
4. 段页式存储管理中段是作业地址空间的最小单位。

课堂练习（判断正误）

1. 静态重定位后不可能使用紧缩技术解决碎片问题。 ✓
2. 在逻辑地址空间中通常采用不连续编址方式。 ×
3. 虚拟存储空间实际上就是辅存空间。 ×
4. 段页式存储管理中段是作业地址空间的最小单位。 ×
5. 取消了存储分配连续性要求的存取管理技术是可变式分区技术。

课堂练习（判断正误）

1. 静态重定位后不可能使用紧缩技术解决碎片问题。 ✓
2. 在逻辑地址空间中通常采用不连续编址方式。 ×
3. 虚拟存储空间实际上就是辅存空间。 ×
4. 段页式存储管理中段是作业地址空间的最小单位。 ×
5. 取消了存储分配连续性要求的存取管理技术是可变式分区技术。 ×
6. 对于段页式系统，当要访问内存中的一个数据时，若联想存储器匹配失败则需要 3 次访问内存。

课堂练习（判断正误）

1. 静态重定位后不可能使用紧缩技术解决碎片问题。 ✓
2. 在逻辑地址空间中通常采用不连续编址方式。 ×
3. 虚拟存储空间实际上就是辅存空间。 ×
4. 段页式存储管理中段是作业地址空间的最小单位。 ×
5. 取消了存储分配连续性要求的存取管理技术是可变式分区技术。 ×
6. 对于段页式系统，当要访问内存中的一个数据时，若联想存储器匹配失败则需要 3 次访问内存。 ✓
7. 页式存储管理中，一个作业可以占用不连续的内存空间，而段式存储管理，一个作业则是占用连续的内存空间。

课堂练习（判断正误）

1. 静态重定位后不可能使用紧缩技术解决碎片问题。 ✓
2. 在逻辑地址空间中通常采用不连续编址方式。 ×
3. 虚拟存储空间实际上就是辅存空间。 ×
4. 段页式存储管理中段是作业地址空间的最小单位。 ×
5. 取消了存储分配连续性要求的存取管理技术是可变式分区技术。 ×
6. 对于段页式系统，当要访问内存中的一个数据时，若联想存储器匹配失败则需要 3 次访问内存。 ✓
7. 页式存储管理中，一个作业可以占用不连续的内存空间，而段式存储管理，一个作业则是占用连续的内存空间。 ×

课堂练习

- 在基本分页存储管理系统中，假定页面大小为 1024 字节，每个页表项占用 4 个字节。（题目中所有数据都是十进制）
  1. 若采用一级页表，系统能支持的逻辑空间最多为多少？

课堂练习

- 在基本分页存储管理系统中，假定页面大小为 1024 字节，每个页表项占用 4 个字节。（题目中所有数据都是十进制）
  1. 若采用一级页表，系统能支持的逻辑空间最多为多少？256K
  2. 若采用两级页表，系统能支持的逻辑空间最多为多少？

课堂练习

- 在基本分页存储管理系统中，假定页面大小为 1024 字节，每个页表项占用 4 个字节。（题目中所有数据都是十进制）
  1. 若采用一级页表，系统能支持的逻辑空间最多为多少？256K
  2. 若采用两级页表，系统能支持的逻辑空间最多为多少？64M
  3. 在两级页表中，现有逻辑地址为 542783，求该地址对应的外层页号（页表页面号）、页号和页内地址。

课堂练习

- 在基本分页存储管理系统中，假定页面大小为 1024 字节，每个页表项占用 4 个字节。（题目中所有数据都是十进制）
  1. 若采用一级页表，系统能支持的逻辑空间最多为多少？256K
  2. 若采用两级页表，系统能支持的逻辑空间最多为多少？64M
  3. 在两级页表中，现有逻辑地址为 542783，求该地址对应的外层页号（页表页面号）、页号和页内地址。

2-18-63

课堂练习

- 在一个段页式存储管理系统中，页面大小为 1024 字节。在一个进程中，它的段表、页表如下所示。求逻辑地址为（1，2099）对应的物理地址。（题目中所有数据都是十进制）

段表	
段号	页表始址
0	76
1	33
2	52

段内页表（第76块）	
页号	块号
0	18
1	22
2	36

段内页表（第33块）	
页号	块号
0	65
1	13
2	40

段内页表（第52块）	
页号	块号
0	6
1	98
2	82

课堂练习

- 在一个段页式存储管理系统中，页面大小为 1024 字节。在一个进程中，它的段表、页表如下所示。求逻辑地址为（1，2099）对应的物理地址。（题目中所有数据都是十进制）

段表	
段号	页表始址
0	76
1	33
2	52

段内页表（第76块）	
页号	块号
0	18
1	22
2	36

段内页表（第33块）	
页号	块号
0	65
1	13
2	40

段内页表（第52块）	
页号	块号
0	6
1	98
2	82

41011

本章作业

谢 谢

School of Computer & Information Engineering
Henan University
Kaifeng, Henan Province
475001
China