第六章 (实)存储管理

§1 地址和空间的概念

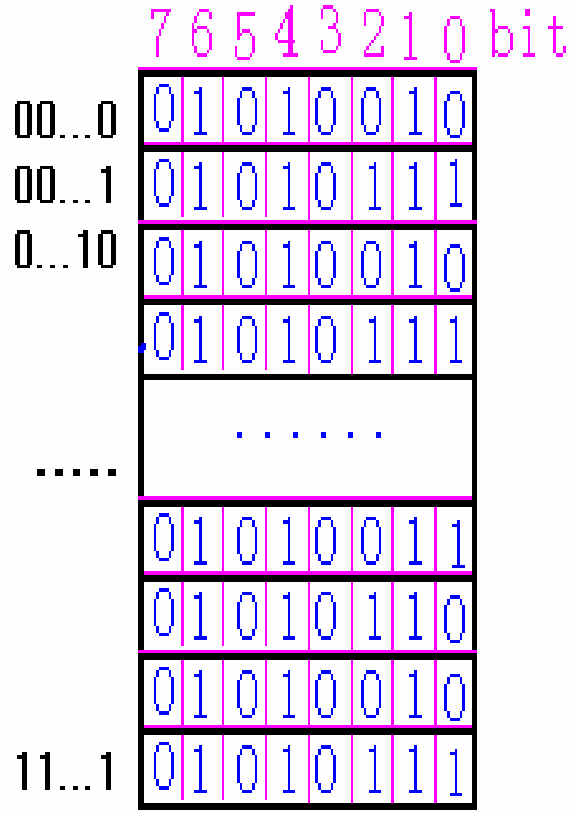
1.主存储器——存储空间

❖物理地址、绝对地址、实地址

主存中信息存储单元的地址编码。从统一的基地址顺序编址,用于芯片级内存单元寻址。

❖存储空间:主存全部存储单元的集合形成存储空间。总是从0开始一直到可用内存最高端,是一维的线性空间。

2.源程序



§1 地址和空间的概念

1.主存储器 ——存储空间

❖物理地址、绝对地址、实地址

主存中信息存储单元的地址编码。从统一的基地址顺序编址,用于芯片级内存单元寻址。

❖存储空间:主存全部存储单元的集合形成存储空间。总是从0开始一直到可用内存最高端,是一维的线性空间。

2.源程序 ——名空间

❖符号地址:源程序中语句(指令)或数据的地址。

❖名空间:源程序所限定的离散的符号地址空间。

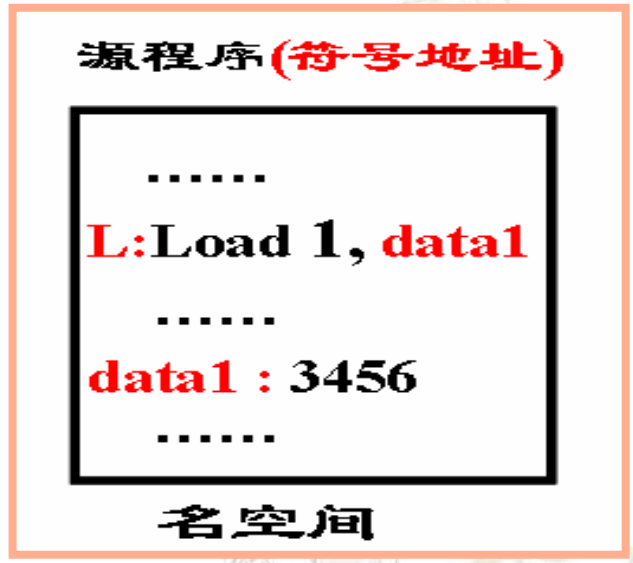
3.目标程序(.OBJ) ——地址空间

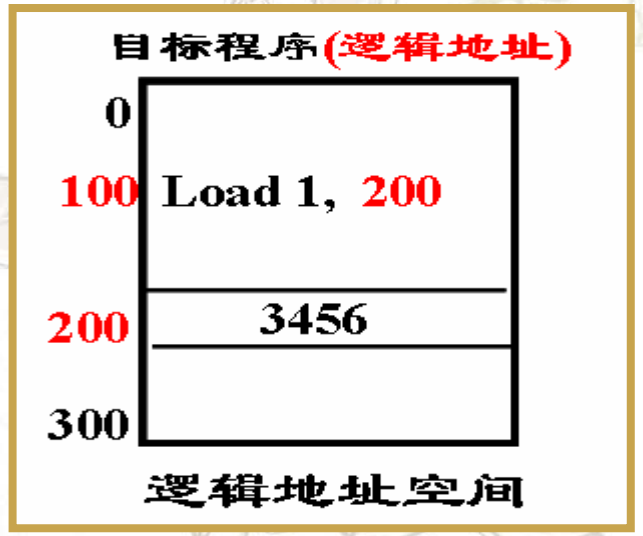
❖逻辑地址、相对地址、虚地址

目标代码常采用相对地址的形式,其首地址为0,其余指令或数据的地址都相对于首地址而编址。由此形成的地址即逻辑地址。

❖地址空间

目标程序所限定的从0开始线性编址的地址范围。





§1 地址和空间的概念

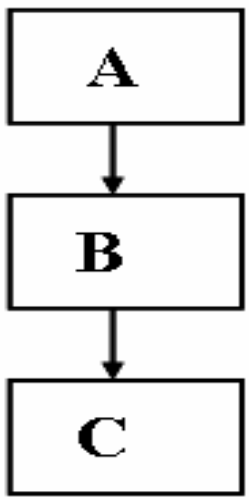
4.目标程序特点

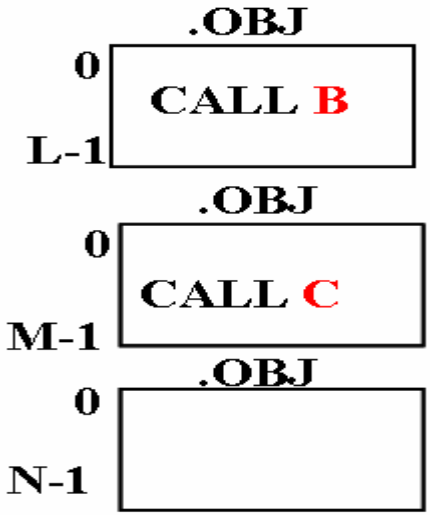
- 目标程序在功能上往往是不完备的，常包含库函数调用。
- 目标程序常包含对其它目标程序的引用，即包含外部调用符号。

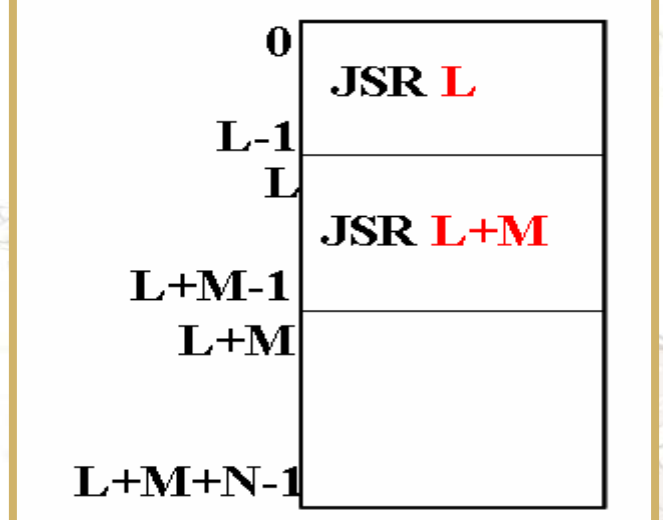
5.链接： 按照程序中各模块间的调用关系，把一组目标模块及库函数装配成完整的可执行程序的过程。**其实质是：** 把各模块的地址空间统一成一个从0开始的一维地址空间。

6.可执行程序.EXE(装入模块) ——采用逻辑地址编址

限定在一个由逻辑地址表示的从0开始的一维线性地址空间中。



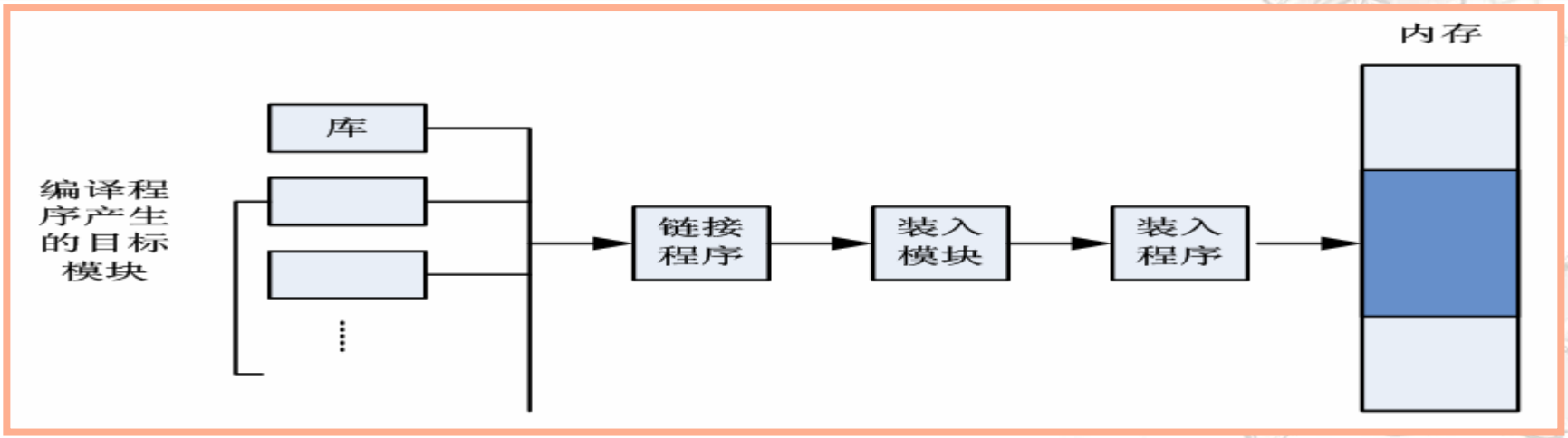




§2 程序链接技术

一、静态链接

定义：链接在程序装入主存前完成。 即由链接程序一次性地将目标模块和所需库函数加以链接，形成功能完备的装入模块，此后不再拆开重链的技术。



§2 程序链接技术

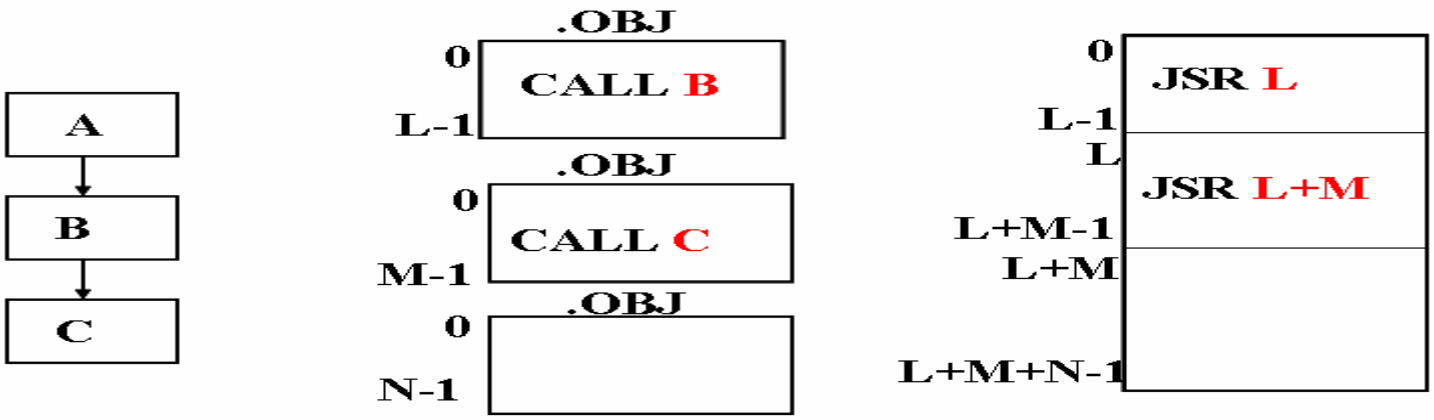
一、静态链接

定义：链接在程序装入主存前完成。 即由链接程序一次性地将目标模块和所需库函数加以链接，形成功能完备的装入模块，此后不再拆开重链的技术。

链接实现过程：

—修改逻辑地址

—变换外部调用符号：将外部调用符号转换为对定位后的逻辑地址的调用

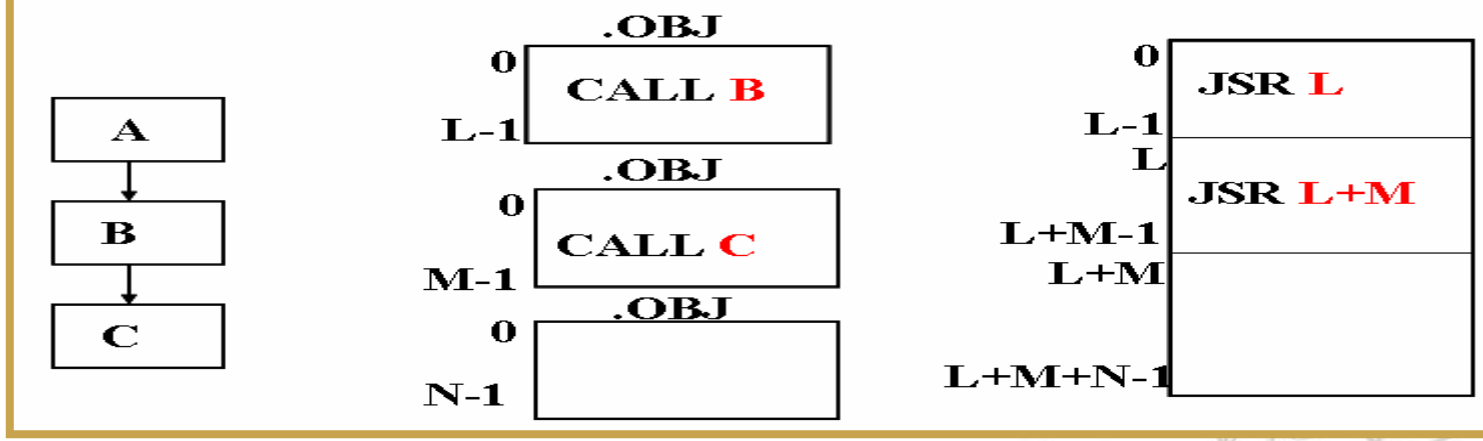


§2 程序链接技术

一、静态链接

定义：链接在程序装入主存前完成。 即由链接程序一次性地将目标模块和所需库函数加以链接，形成功能完备的装入模块，此后不再拆开重链的技术。

— 在程序一次运行期间，装入模块的结构静态不变
— 程序多次运行，装入模块的结构静态不变
— 不便于软件升级更新，除非重新链接
— 不支持软件共享



§2 程序链接技术

二、装入时动态链接

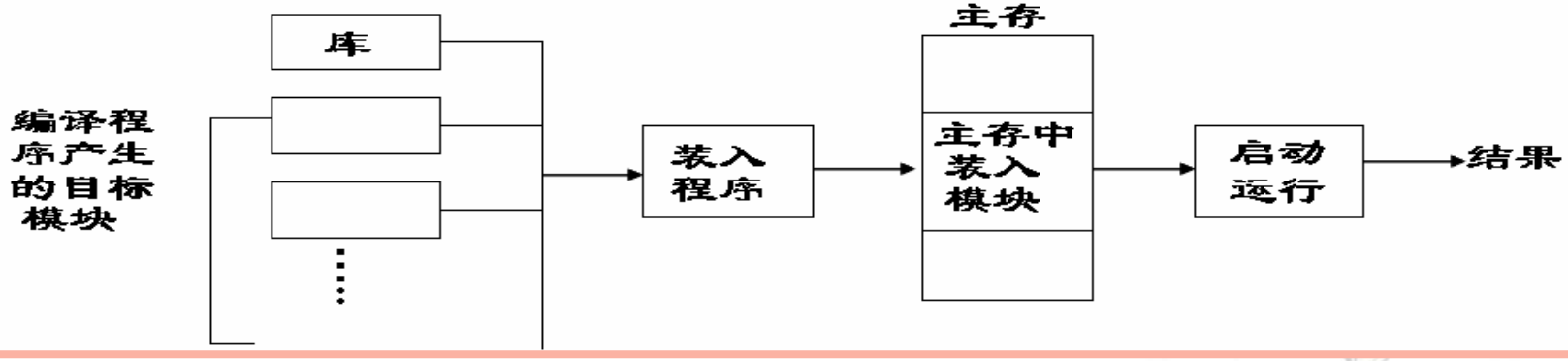
定义：链接在程序装入主存时同时进行。 即在装入一个目标模块时，如发生外部模块调用，就由装入程序检索并装入该模块，再将它链接到调用者模块上去。

—便于软件的修改和更新

—便于实现目标模块共享，即多个程序共享一个子模块的内存副本

— 在运行前完成链接，装入模块的结构是静态的，仍属静态链接技术

—可能链接在运行中无需使用模块，链接时间长，低效



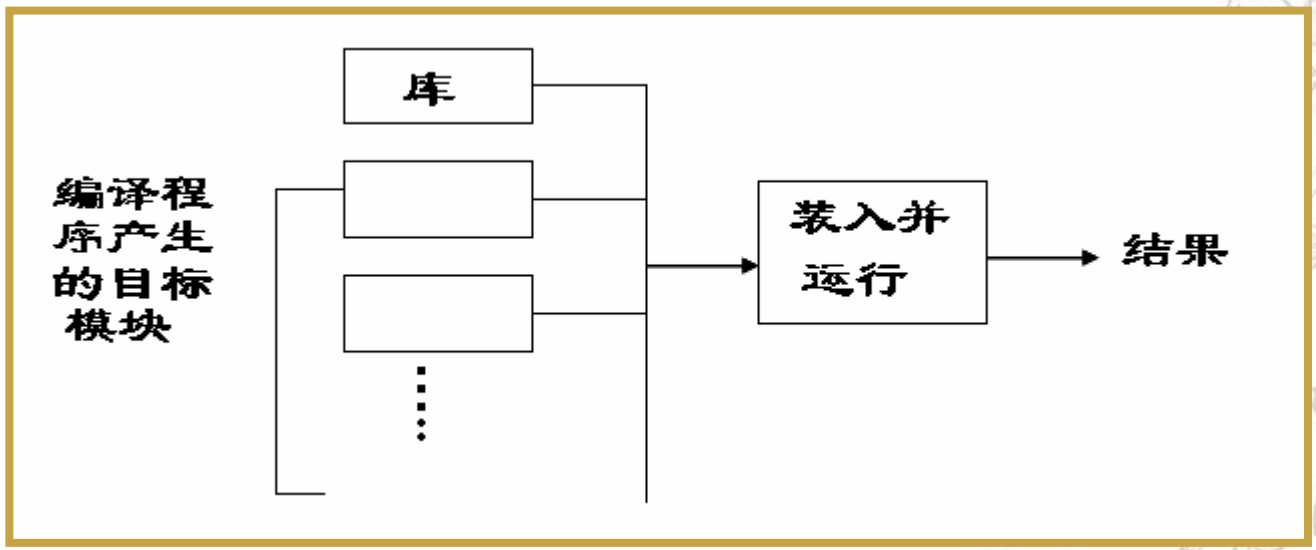
§2 程序链接技术

三、运行时动态链接

定义：链接在程序执行过程中动态进行。即在执行过程中，当真正需要一个被调用模块时，再由操作系统检索并装入主存，并将其链接到调用者模块上去。

—只链接本次运行所需目标模块，装入模块的结构是动态的

—无需链接所有目标模块，加快了链接过程和效率



§2 程序链接技术

❖关于地址与空间的总结

- 程序在成为进程前的准备工作(重点理解)：

— 编辑：形成源文件(符号地址——名空间)

— 编译：形成一或多个目标模块(模块内符号地址解析，形成逻辑地址——地址空间)

— 链接：装配多个目标模块及库函数生成可执行模块(模块间符号地址解析，逻辑地址——地址空间)

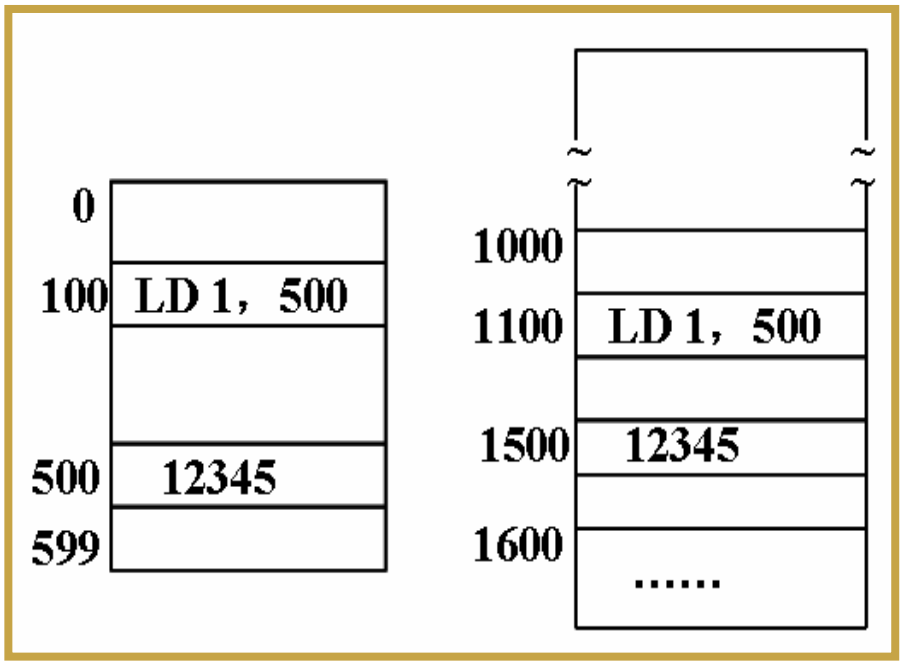
— 装入：构造PCB，形成进程

(执行时，使用物理地址——部分存储空间)

§3 地址重定位

一、地址重定位含义

将作业地址空间中的逻辑地址变换成主存物理地址的过程称地址重定位或地址映射。



§3 地址重定位

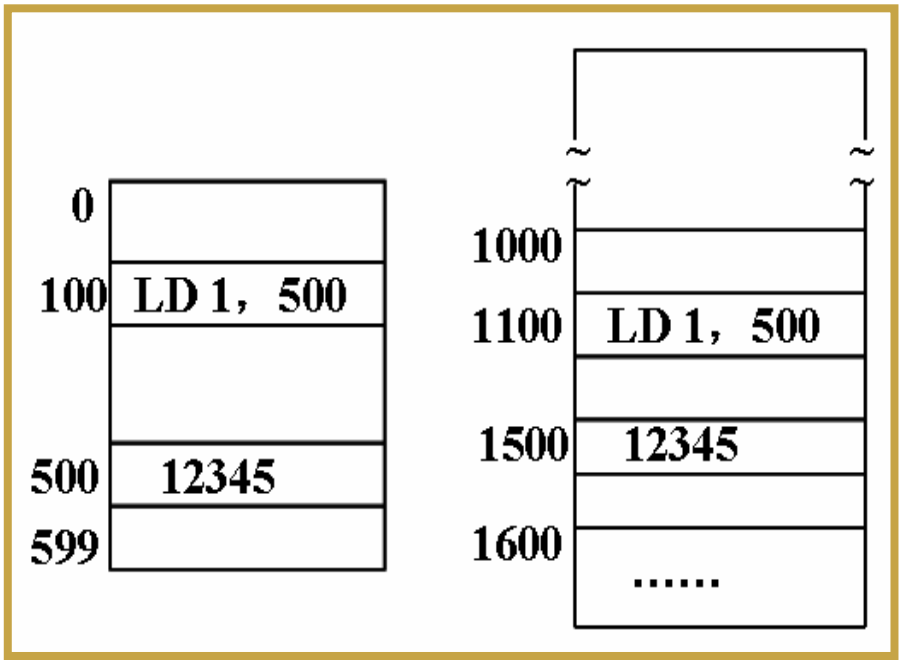
二、静态重定位

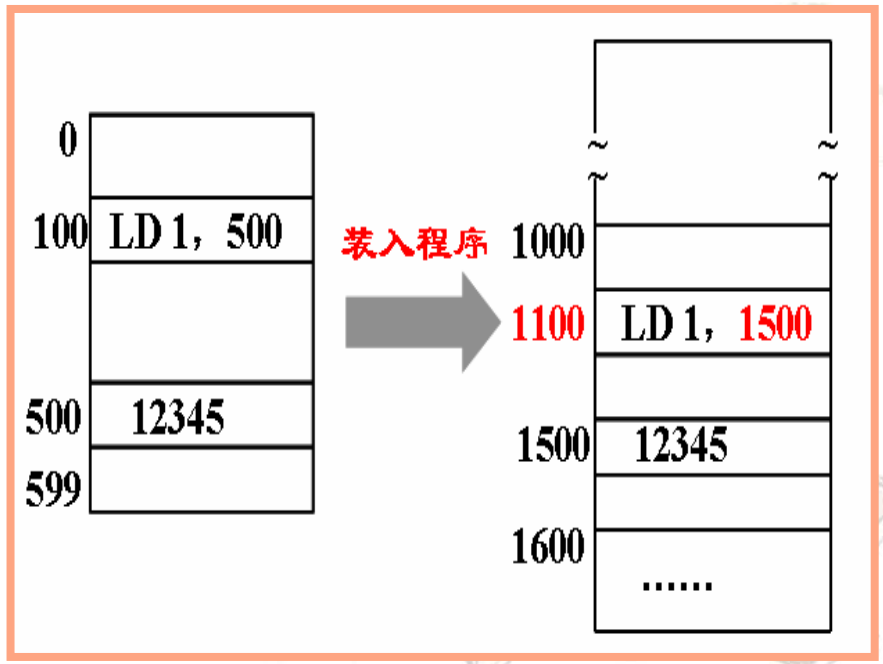
程序在装入主存时，由重定位装入程序一次性地将作业地址空间中的逻辑地址变换为主存物理地址。

—无需硬件地址变换机构支持，重定位开销小、易于实现

—程序一旦装入主存不允许移动，不允许动态追加主存

—必须给作业分配连续主存区域，不支持离散主存分配





§3 地址重定位

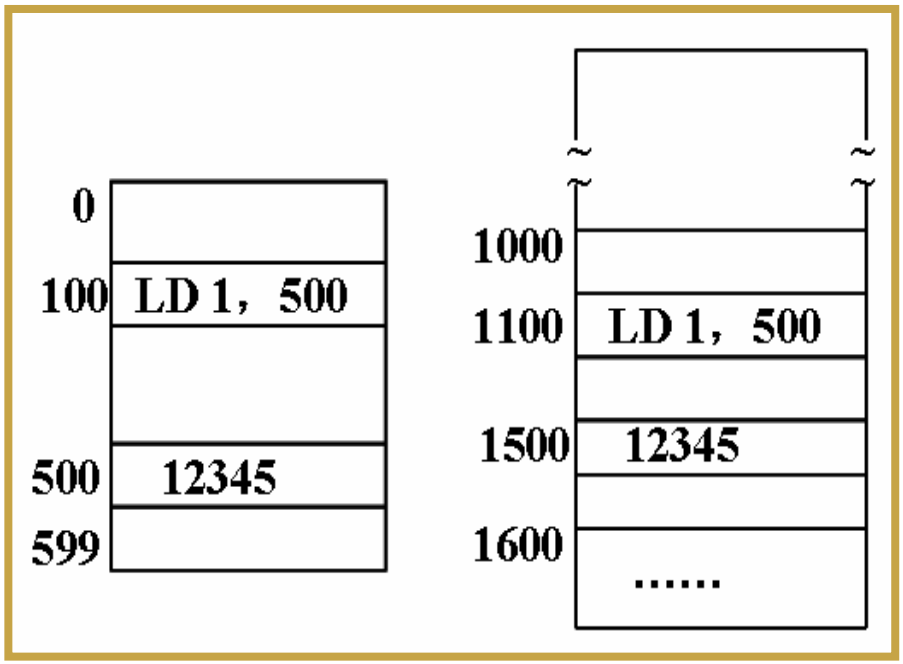
二、动态重定位

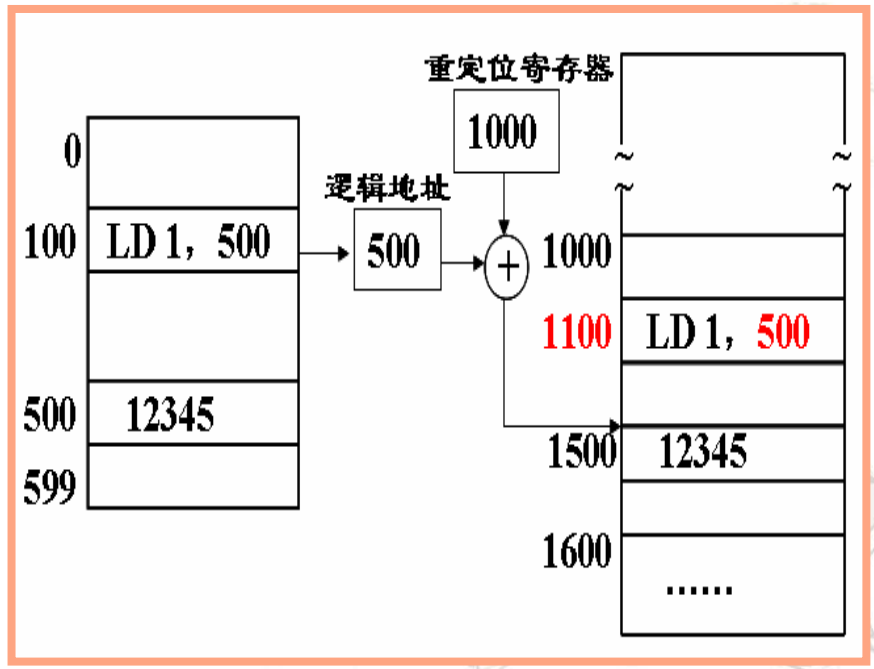
- 在执行指令或访问数据时,借助于重定位寄存器RR(Relocation Register),将指令或数据的逻辑地址映射成物理地址。

—允许程序在内存中移动，允许扩充主存及换进换出

—支持离散的主存分配，便于小主存块的利用

—需要硬件地址变换机构(设置RR等)的支持





§3 地址重定位

例1（北京理工）：程序经编译或汇编以后形成目标程序，其中的指令顺序是以0作为参考地址进行编址的，这些地址称为 逻辑地址 。

例2（西北工业大学）：把程序地址空间中使用的逻辑地址变成内存中物理地址称为 C 。

A. 加载　　B. 物理化　　C.重定位　　D.逻辑化

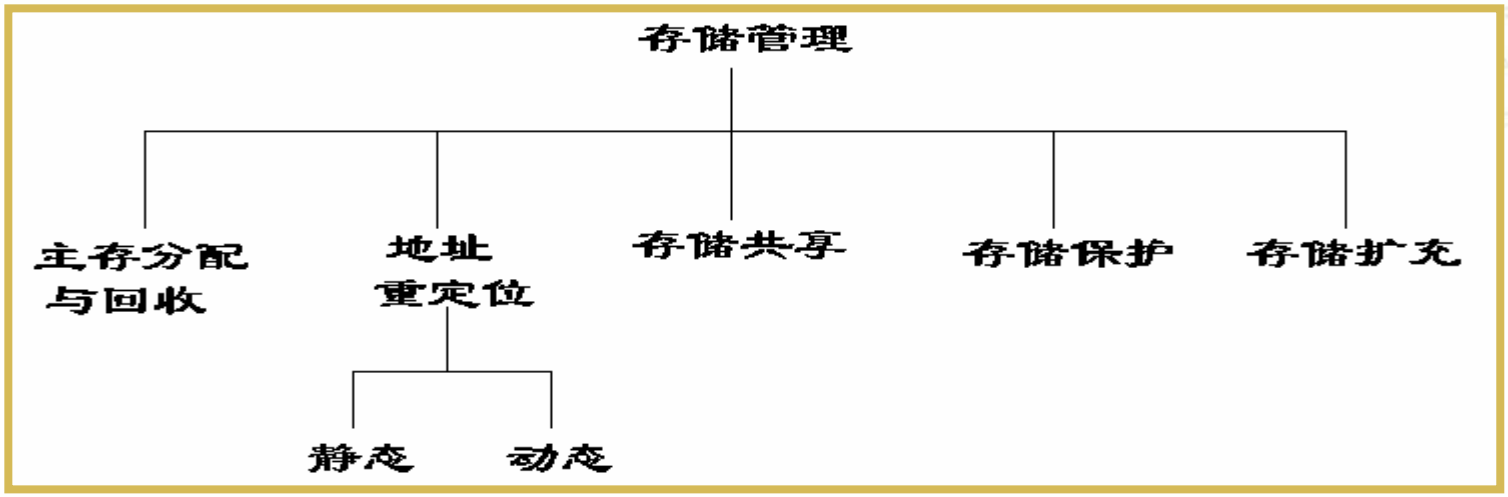
例3（西安电子科技大学）：采用静态重定位方式装入的作业，其地址变换工作是在 A 完成的；采用动态重定位方式装入的作业，其地址变换工作是在 B 完成的；

A.作业装入时　B. 执行指令时　C. 作业调度时　D.编译时

例4：动态重定位技术依赖于 B 。

A.装入程序　　B. 重定位寄存器　　C. 目标程序　　D.编译程序

§4 存储管理的功能



1.分配与回收：记录内存情况，并设计主存分配/回收算法

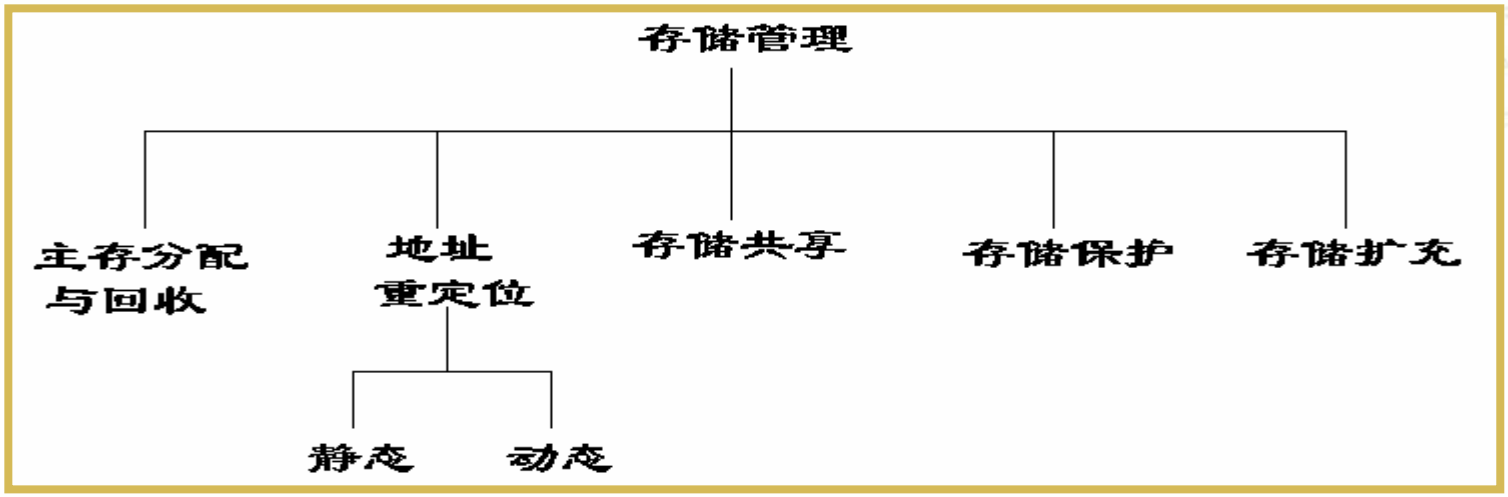
2.地址重定位：将地址空间中的逻辑地址映射成主存物理地址

3.共享：能使两或多个进程共用主存相同区域：程序、数据共享

4.存储保护：避免各道程序间的相互侵犯。特别是，当一道程序发生错误时，不能侵犯其它程序尤其是OS的区域

5.存储“扩充”：采用一定技术来“扩充”主存的容量，使用户感知比实际主存容量更大的主存空间：即实现“虚拟存储器”

§4 存储管理的功能



例1（西安电子科技大学）：存储管理应实现的功能是：主存空间分配和保护、地址重定位、主存空间共享与主存扩充。

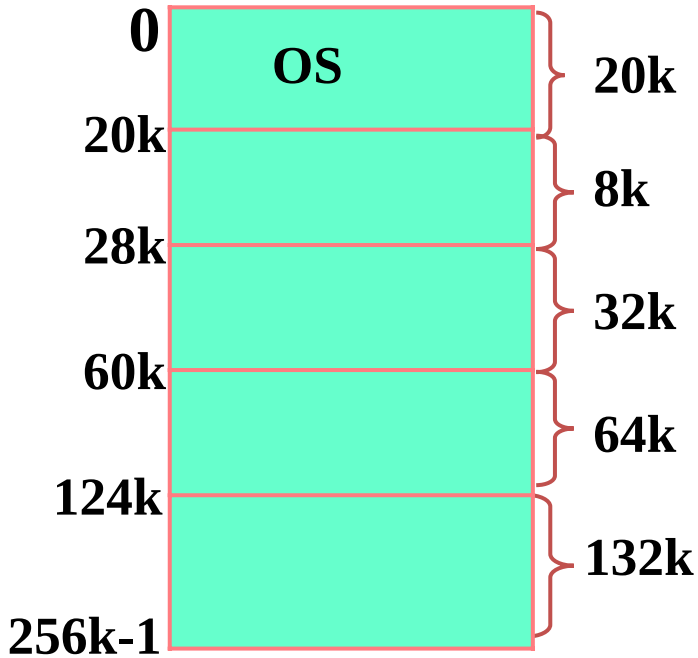
例2（华中科技大学）：在多用户环境中为了实现多用户之间的隔离，必须采用存储保护手段。

例3：如果一个程序为多个进程共享，那么该程序的代码在执行过程中不能被修改，即程序应该是可重入码(纯代码)。

§5 分区式分配

一、固定式分区

基本思想:预分主存为若干大小不变、位置固定的分区，规定一个分区最多分配给一道作业使用。



分区号	大小	始址	状态
1	8k	20k	未分配
2	32k	28k	未分配
3	64k	60k	未分配
4	132k	124k	未分配

表：分区说明表

§5 分区式分配

一、固定式分区

基本思想:预分主存为若干大小不变、位置固定的分区，规定一个分区最多分配给一道作业使用。

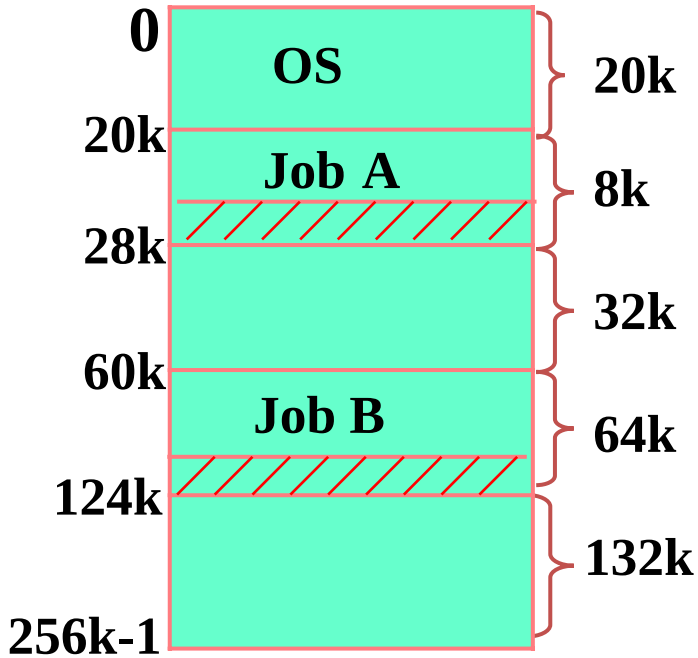
分区号	大小	始址	状态
1	8k	20k	未分配
2	32k	28k	未分配
3	64k	60k	未分配
4	132k	124k	未分配

表：分区说明表

§5 分区式分配

一、固定式分区

基本思想:预分主存为若干大小不变、位置固定的分区，规定一个分区最多分配给一道作业使用。



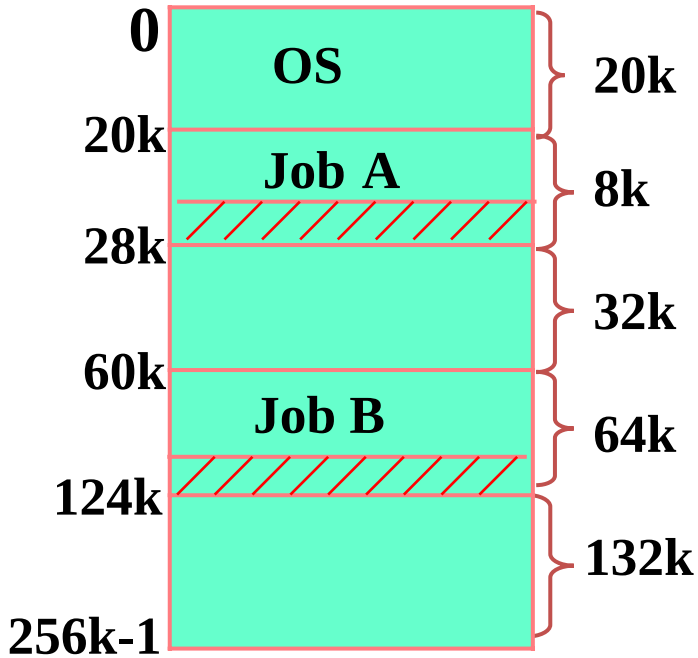
分区号	大小	始址	状态
1	8k	20k	已分配
2	32k	28k	未分配
3	64k	60k	未分配
4	132k	124k	未分配

表：分区说明表

§5 分区式分配

一、固定式分区

基本思想:预分主存为若干大小不变、位置固定的分区，规定一个分区最多分配给一道作业使用。



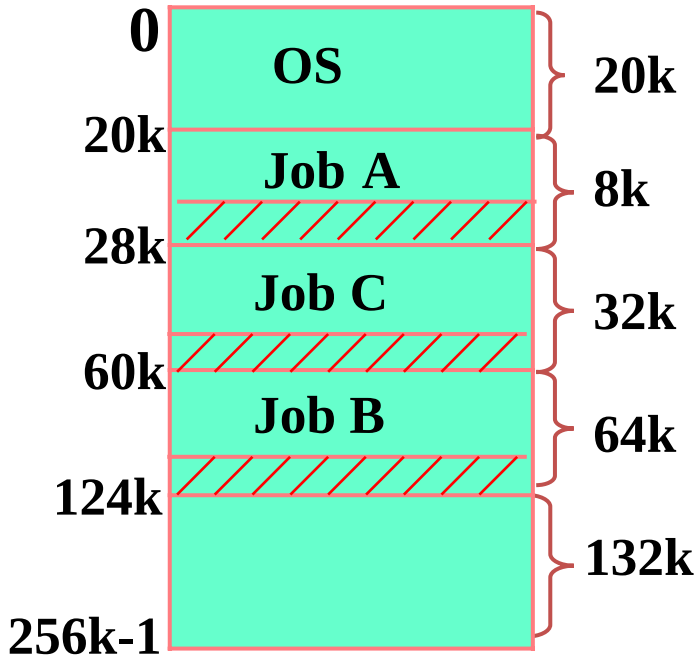
分区号	大小	始址	状态
1	8k	20k	已分配
2	32k	28k	未分配
3	64k	60k	已分配
4	132k	124k	未分配

表：分区说明表

§5 分区式分配

一、固定式分区

基本思想:预分主存为若干大小不变、位置固定的分区，规定一个分区最多分配给一道作业使用。



分区号	大小	始址	状态
1	8k	20k	已分配
2	32k	28k	未分配
3	64k	60k	已分配
4	132k	124k	未分配

表：分区说明表

§5 分区式分配

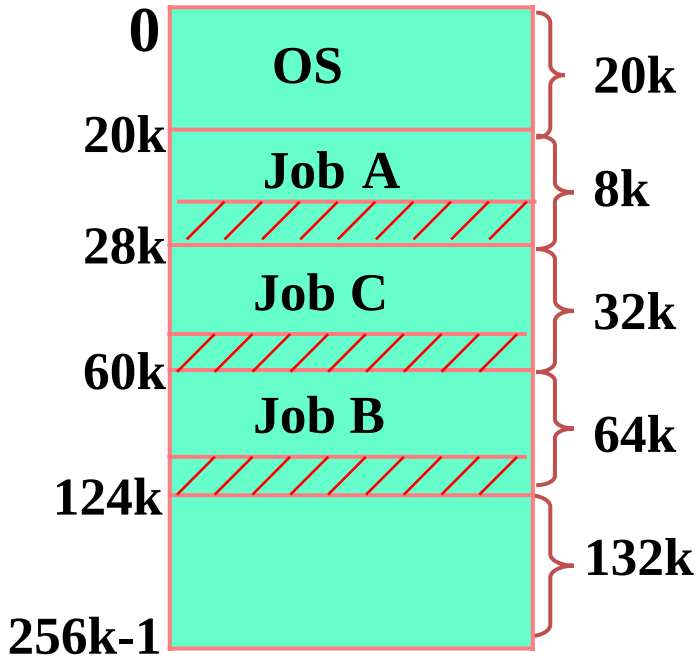
一、固定式分区

基本思想:预分主存为若干大小不变、位置固定的分区，规定一个分区最多分配给一道作业使用。

—是支持多道的最简单的存储管理方案： eg IBM 360 MFT OS

—分区“内零头”问题

—分区使用不均衡问题



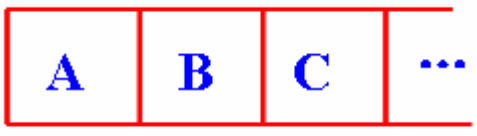
分区号	大小	始址	状态
1	8k	20k	已分配
2	32k	28k	已分配
3	64k	60k	已分配
4	132k	124k	未分配

表：分区说明表

§5 分区式分配

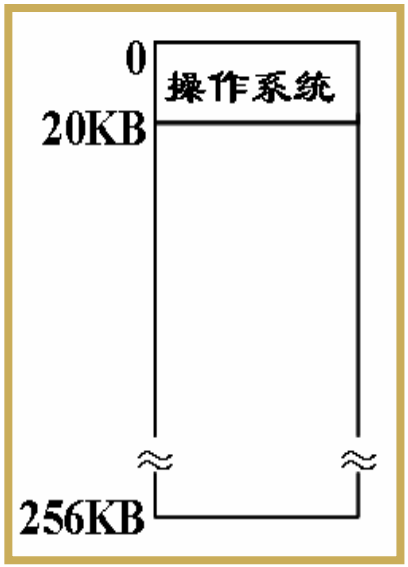
二、可变式分区

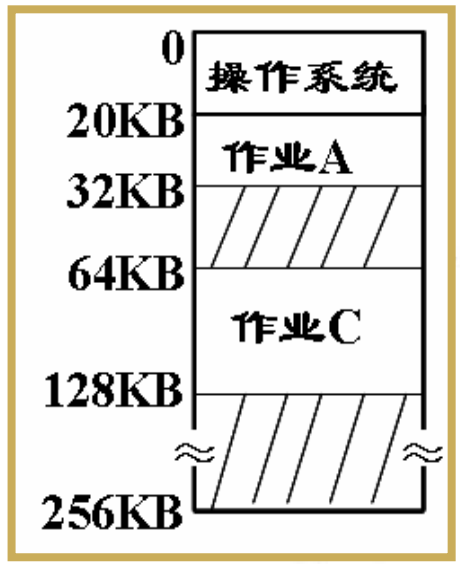
1.基本思想：伴随作业进入主存,动态划定**与作业等长**的占用分区

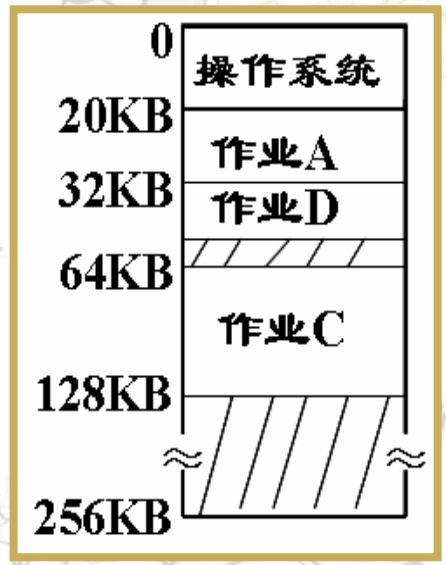


作业B结束，撤离

作业D进入，30K







§5 分区式分配

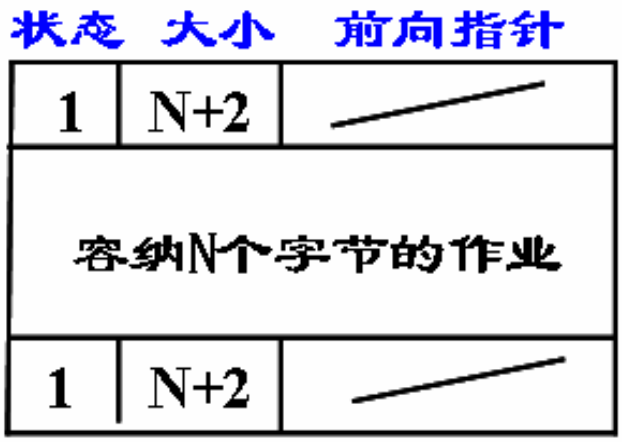
二、可变式分区

1.基本思想：伴随作业进入主存,动态划定与作业等长的占用分区

2.数据结构——空闲分区链

在占用分区首尾设立控制字，说明分区信息。

在空闲区首尾设立控制字，并将所有空闲分区拉成双向链表。



§5 分区式分配

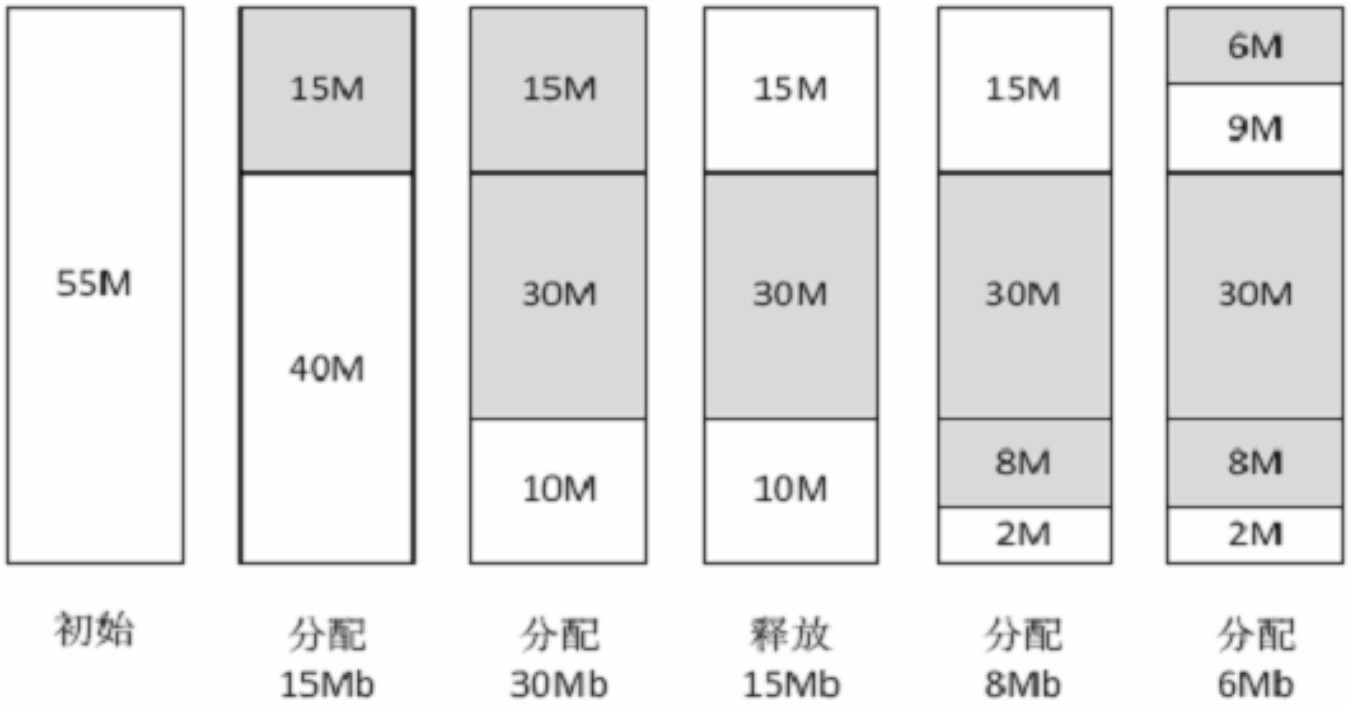
3.分区分配算法

- 首次适应(First Fit,FF)：将空白分区按地址增序链接，从链首找出第一个大小满足需求的空闲区予以分配。
  ——高地址部分的大空间易于满足大作业需求
  ——总从低址部分开始查找空闲区，分配效率逐步下降
- 循环首次适应(Next Fit,NF)：将空白区拉成双向循环链，例行从起始查寻位置找出第一个大小满足需求的空闲区。
  —— 存储空间使用均衡，合并几率大
  —— 对大作业适应性下降
- 最佳适应(Best Fit,BF)：空白区按大小递增次序链接，例行从链首查找第一个大小满足需求的空闲区。
  —命中满足作业要求的最小分区,但分配剩余部分常是”零头”
- 最坏适应(Worst Fit,WF)：空白区按大小递减次序链接，例行从链首最大分区开始分配。

§5 分区式分配

例(10年):某基于动态分区存储管理的计算机，其主存容量为55Mb(初始为空)，采用最佳适配算法(Best Fit)，分配和释放的顺序为：分配15Mb,分配30Mb,释放15Mb,分配8Mb,分配6Mb,此时主存中最大空闲分区的大小是 B 。

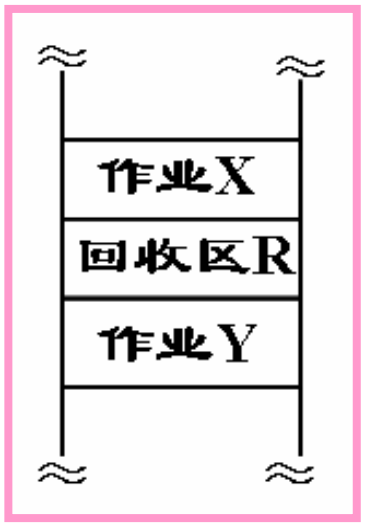
A.7Mb B.9Mb C.10Mb D.15Mb

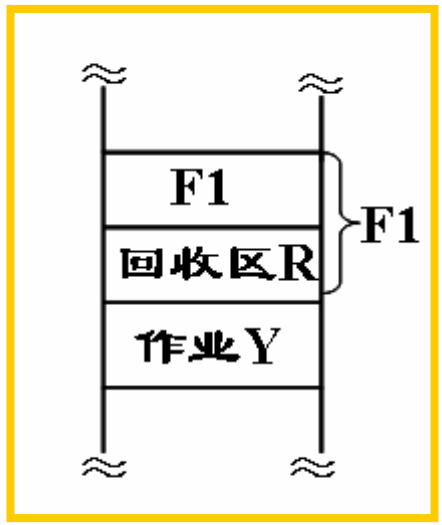


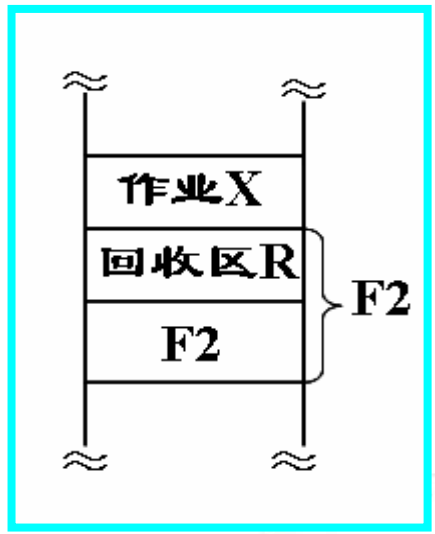
§5 分区式分配

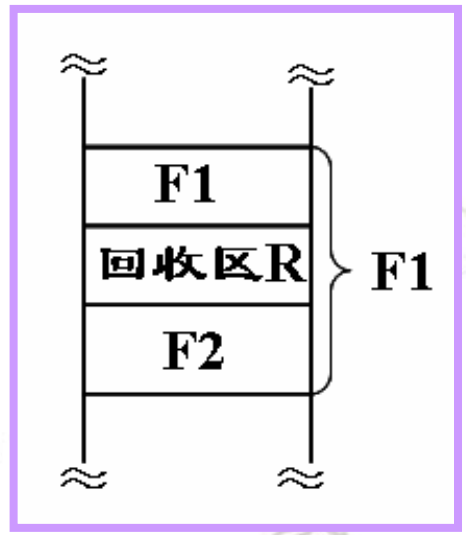
4.分区回收算法

当作业退出内存、释放分区时，回收算法应**判别**回收区是否有**相邻空闲区**，若**有则合并**成更大空闲区；没有则将回收分区链入空闲分区链适当位置。









例1(**西安理工大学**)：按最先适应算法分配的分区，一定与作业要求的容量大小最接近。（ F ）

§5 分区式分配

4.分区回收算法

当作业退出内存、释放分区时，回收算法应判别回收区是否有相邻空闲区，若有则合并成更大空闲区；没有则将回收分区链入空闲分区链适当位置。

例2(西安电子科技大学)：在可变分区存储管理中，一个作业释放出一个分区却导致内存空白分区数减1，是因为该作业的回收区 D 。

A.只有上邻而无下邻空白区

B.只有下邻而无上邻空白区

C.既无上邻也无下邻空白区

D.既有上邻也有下邻空白区

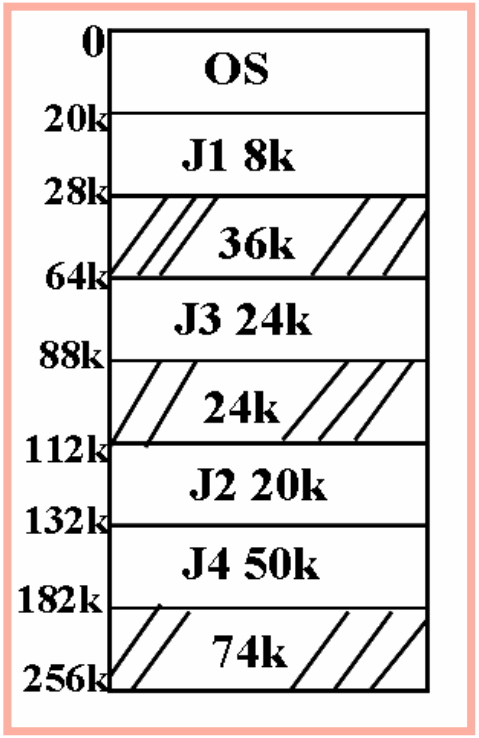
例1(西安理工大学)：按最先适应算法分配的分区，一定与作业要求的容量大小最接近。（ F ）

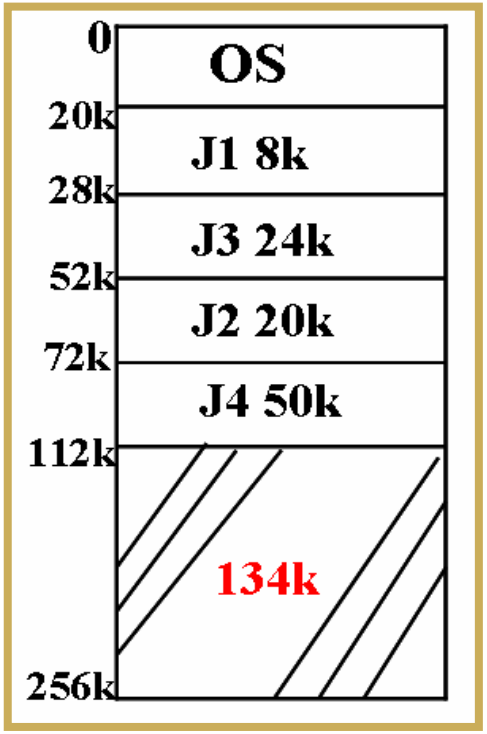
§5 分区式分配

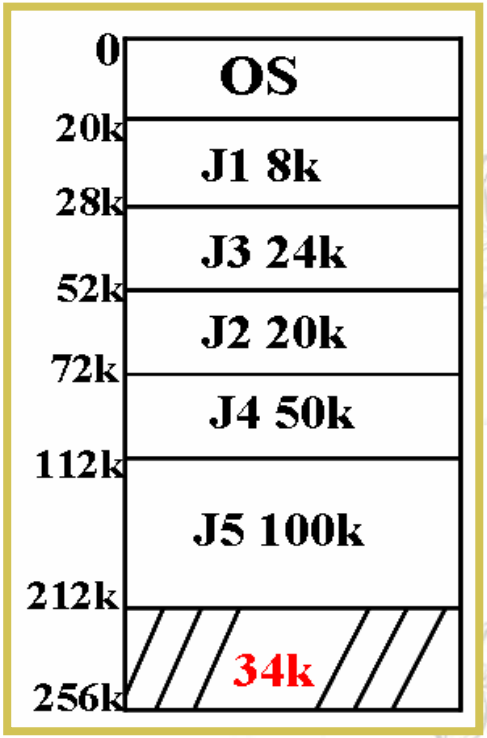
三、可重定位式分区

拼接(紧凑)：通过移动占用区，把多个分散的空闲小分区(碎片)合并成一片大的连续区域的过程。

——允许作业**移动**，**必须**使用**动态重定位**技术







$\sum$// =134k

J5

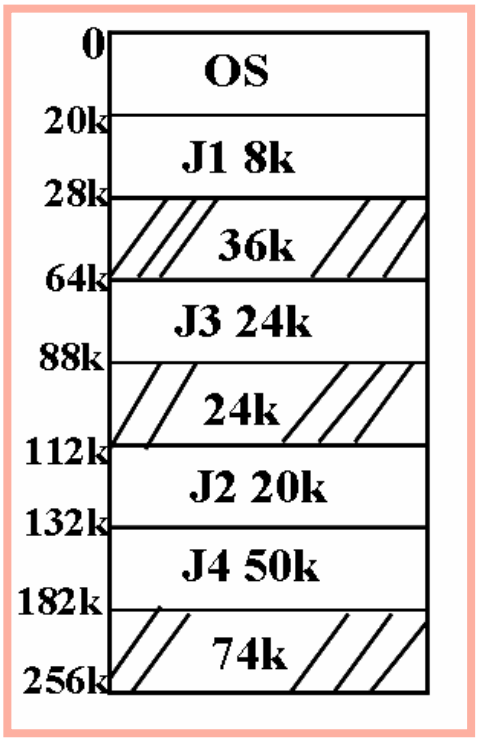
100k

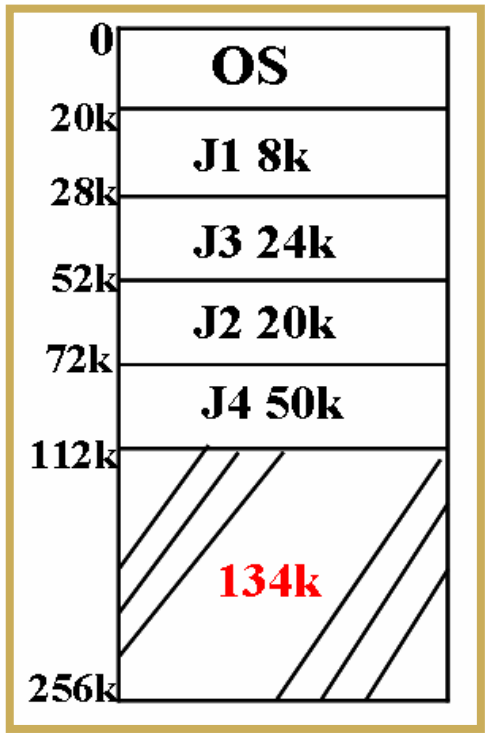
§5 分区式分配

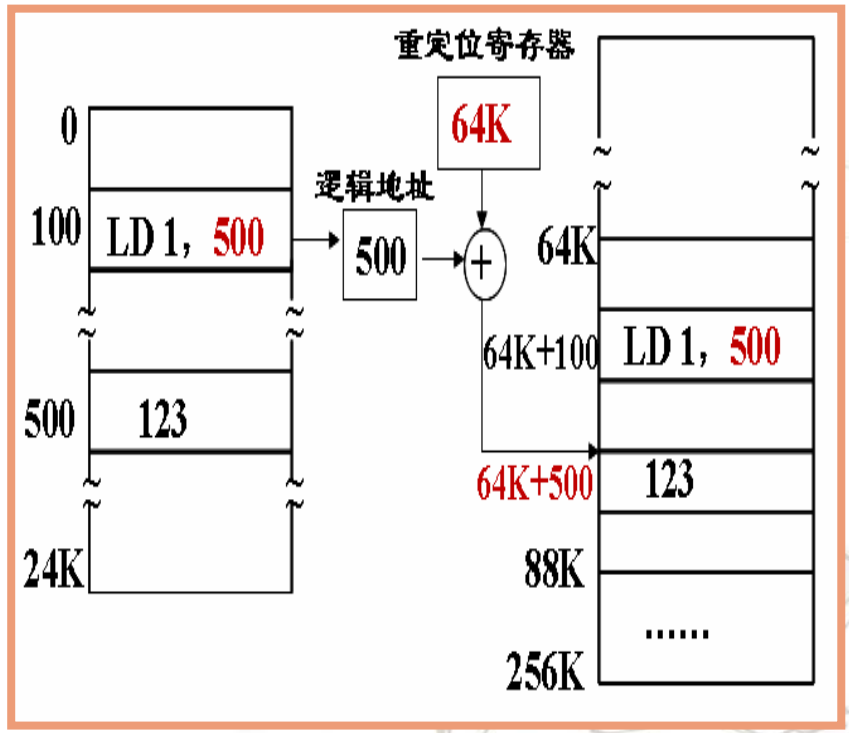
三、可重定位式分区

拼接(紧凑)：通过移动占用区，把多个分散的空闲小分区(碎片)合并成一片大的连续区域的过程。

——允许作业**移动**，**必须**使用**动态重定位**技术







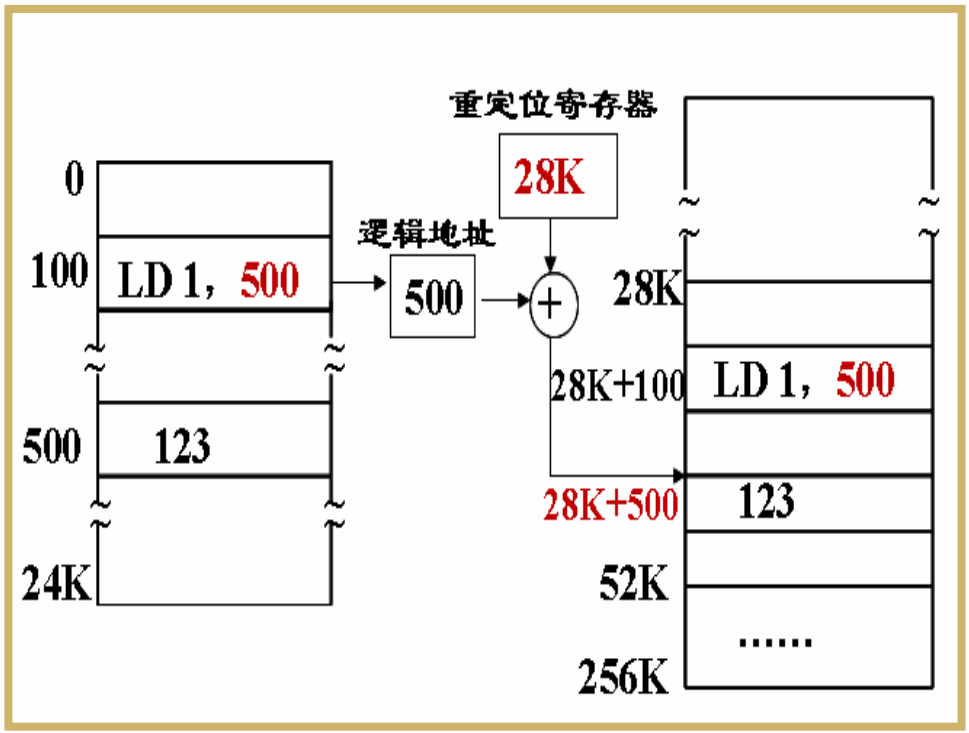
§5 分区式分配

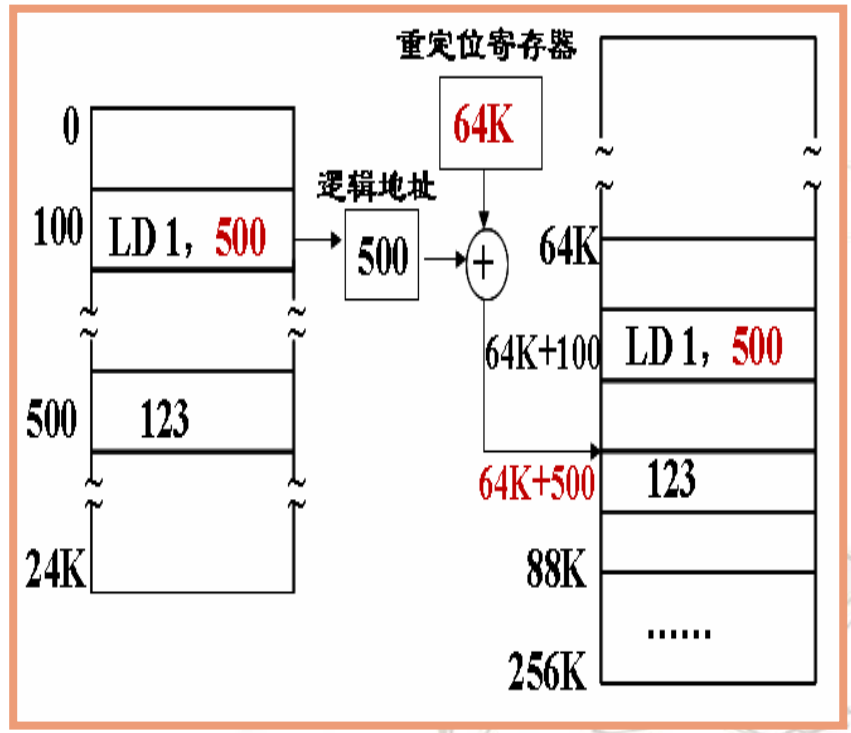
三、可重定位式分区

拼接(紧凑)：通过移动占用区，把多个分散的空闲小分区(碎片)合并成一片大的连续区域的过程。

——允许作业**移动**，**必须**使用**动态重定位**技术

——以时间换空间





§5 分区式分配

例1(北京理工)：当内存碎片容量大于某一作业所申请的内存空间时，__C__。

A.可以直接为这一作业分配内存

B.不可以为这一作业分配内存

C.拼接后，可为这一作业分配内存

D.可把作业分成几个部分，分别放入几个碎片中

例2(南大)：在可变分区存储管理中，可以采用移动技术提高主存利用率，但不能被移动的作业是__C__。

A.正在计算一个表达式的值的作业

B.正在主存中取数据准备计算的作业

C.正在等待外围设备传输信息的作业

D.正在把计算结果写入主存的作业

§5 分区式分配

例3(东大):在一个多道系统中，用户主存空间为100k，磁带机2台，打印机1台。系统采用可变分区存储管理策略，并采用首次适应算法进行主存分配。对外设采用静态分配方式，作业调度采用FCFS算法。现有作业序列如下表所示：

作业号	到达时间	要求计算时间	内存需求	申请磁带机数	申请打印机数
1	8:00	25分钟	15k	1台	1台
2	8:20	10分钟	30k	0台	1台
3	8:20	20分钟	60k	1台	0台
4	8:30	20分钟	20k	1台	0台
5	8:35	15分钟	10k	1台	1台

假设I/O操作及调度时间忽略不计，且内存中的作业平分CPU时间。试问：

(1)作业调度选中作业的次序是什么？ J1→J3→J4→J2→J5

(2)最大及最小作业周转时间分别是多少？

(3)作业全部执行结束时间是多少？

§5 分区式分配

例3(东大):在一个多道系统中，用户主存空间为100k，磁带机2台，打印机1台。系统采用可变分区存储管理策略，并采用首次适应算法进行主存分配。对外设采用静态分配方式，作业调度采用FCFS算法。现有作业序列如下表所示：

JOB	T_{si}	T_{Bi}	T_{ci}	T_i
1	8:00	8:00	8:30	30min
2	8:20	9:00	9:15	55min
3	8:20	8:20	9:00	40min
4	8:30	8:30	9:10	40min
5	8:35	9:15	9:30	55min

假设I/O操作及调度时间忽略不计，且内存中的作业平分CPU时间。试问：

(1)作业调度选中作业的次序是什么？ J1→J3→J4→J2→J5

(2)最大及最小作业周转时间分别是多少？55min，30min

(3)作业全部执行结束时间是多少？ 9:30

§5 分区式分配

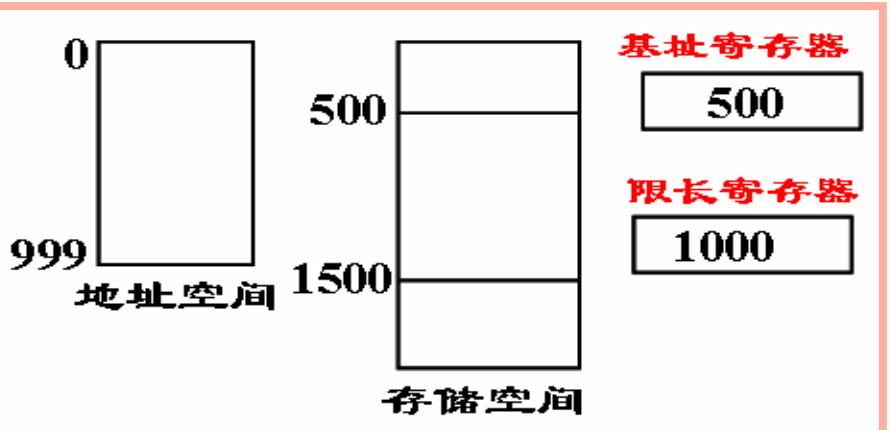
四、存储保护

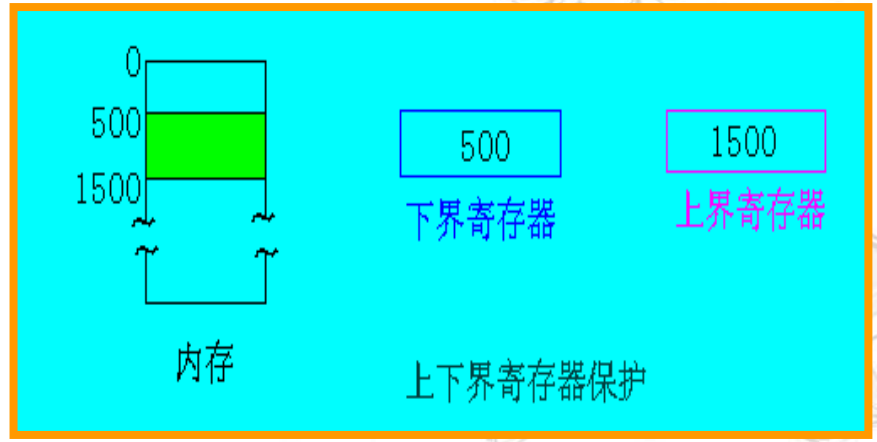
1.基址、限长寄存器

- 基址寄存器存放程序被装入内存的起始地址，
  限长寄存器存放程序地址空间总长度。
- **判别式：** 0≤被访问的逻辑地址<限长寄存器内容

2.上下界寄存器

- 下界寄存器存放程序被装入内存的起始地址，
  上界寄存器存放程序装入内存后的末地址的下一地址。
- **判别式：** 下界寄存器之值≤物理地址<上界寄存器之值





§5 分区式分配

例1(华中理工)：在某系统中采用基址、限长寄存器的方法来保护存储信息，判断是否越界的判别式为__A__。

A. 0 ≤被访问的逻辑地址<限长寄存器的内容

B. 0 ≤被访问的逻辑地址≤限长寄存器的内容

C. 0 ≤被访问的物理地址<限长寄存器的内容

D. 0 ≤被访问的物理地址≤限长寄存器的内容

例2(09年):分区分配内存管理方式的主要保护措施是__A__。

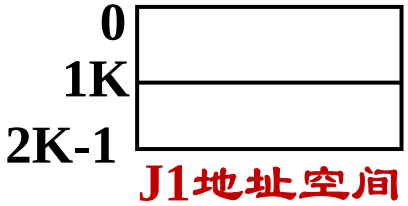
A. 界地址保护　　B. 程序代码保护

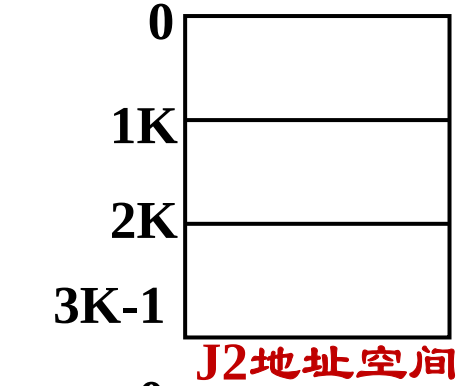
C. 数据保护　　D. 栈保护

§6 分页存储管理

一、基本原理

- 等分主存为“块”(页框Page Frame)
- 等分作业地址空间为“页”，且页长=块长
- 作业运行前，装入地址空间全部页面，一页装入一块





§6 分页存储管理

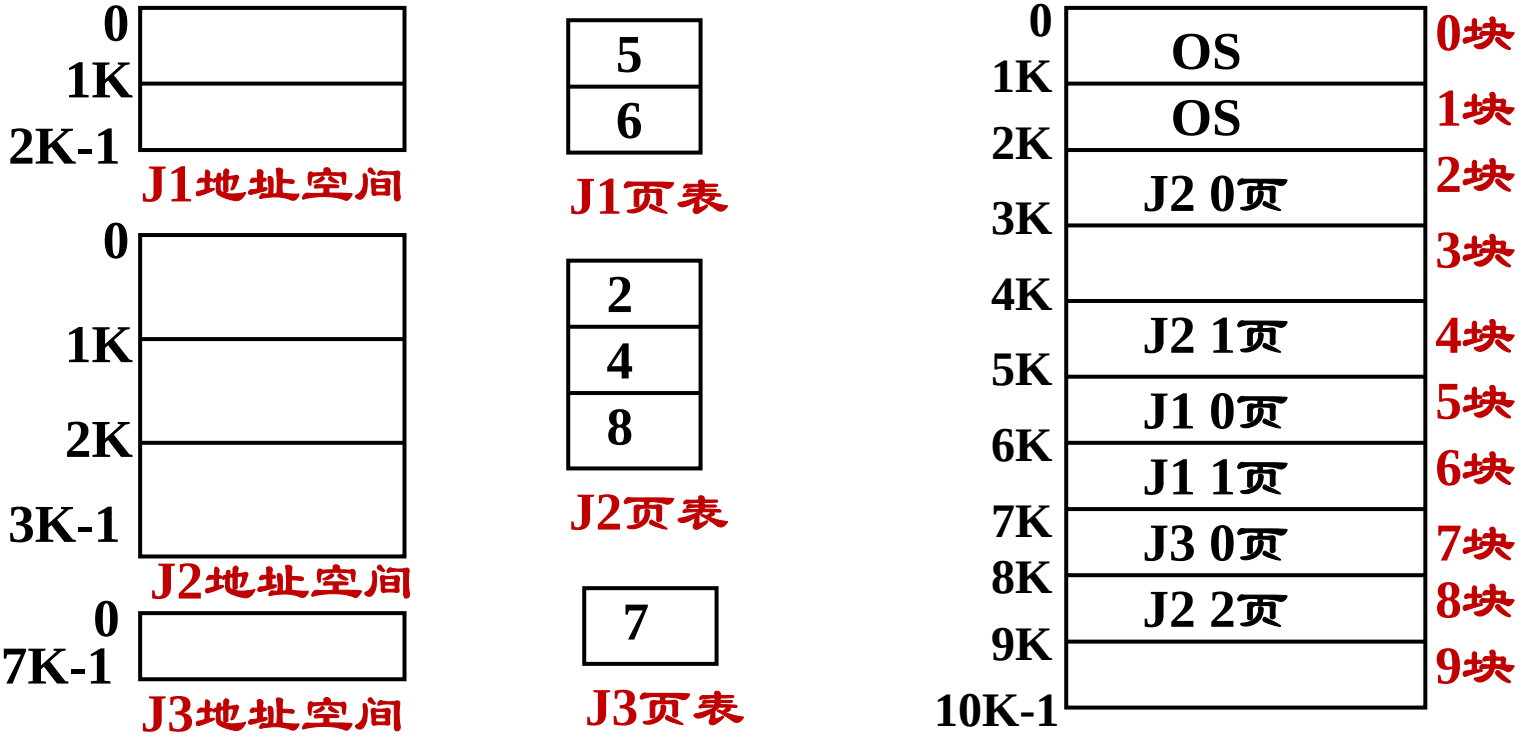
二、地址重定位

1.页表PT(Page Table)

页面号和主存块号的映像表，以页号为序记录各页所在块号。

2.页表控制寄存器PTR(Page Table Register)

设置现行进程页表长度和页表起始地址。



§6 分页存储管理

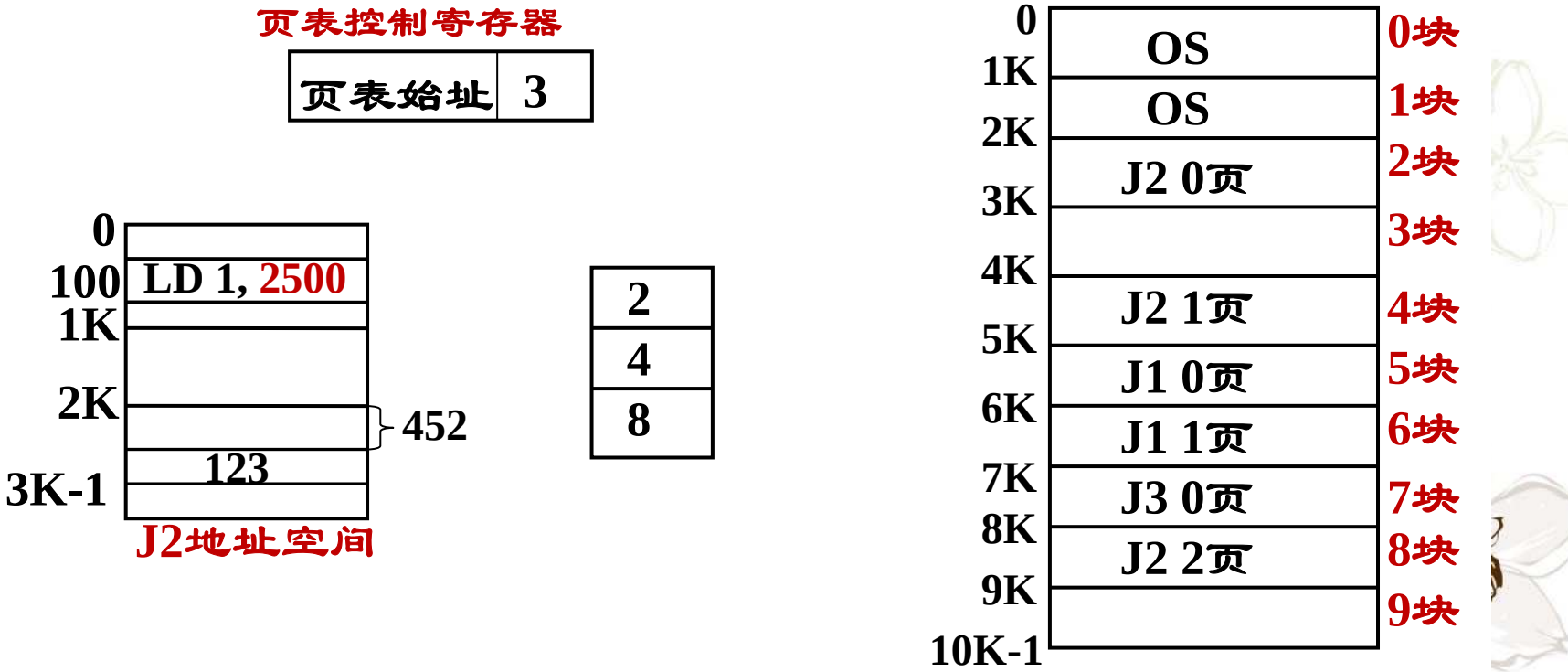
二、地址重定位

1.页表PT(Page Table)

页面号和主存块号的映像表，以页号为序记录各页所在块号。

2.页表控制寄存器PTR(Page Table Register)

设置现行进程页表长度和页表起始地址。



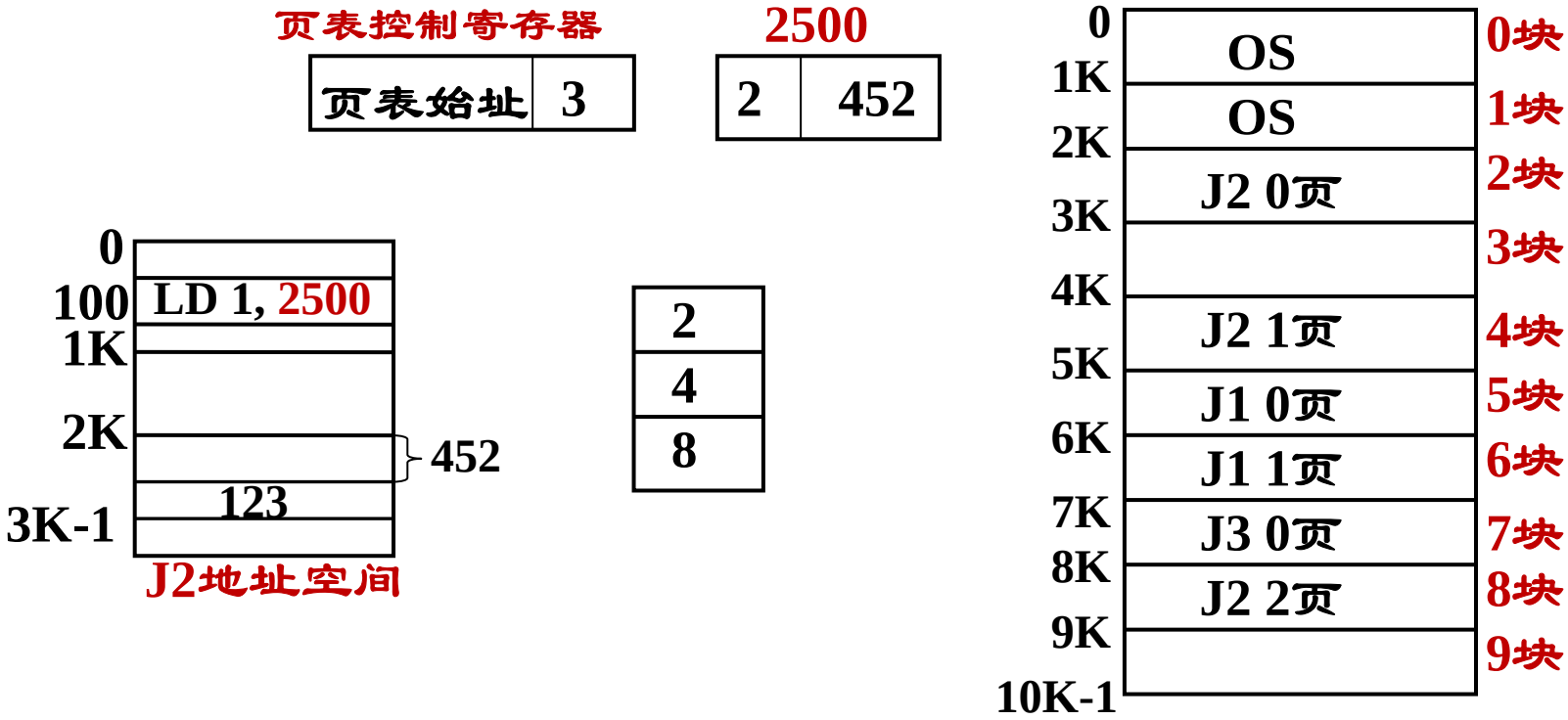
§6 分页存储管理

二、地址重定位

3.逻辑地址拆分

- 设CPU有效地址长15位，块长1K：

【结论】逻辑地址高5位表示页号P，低10位为页内偏移W

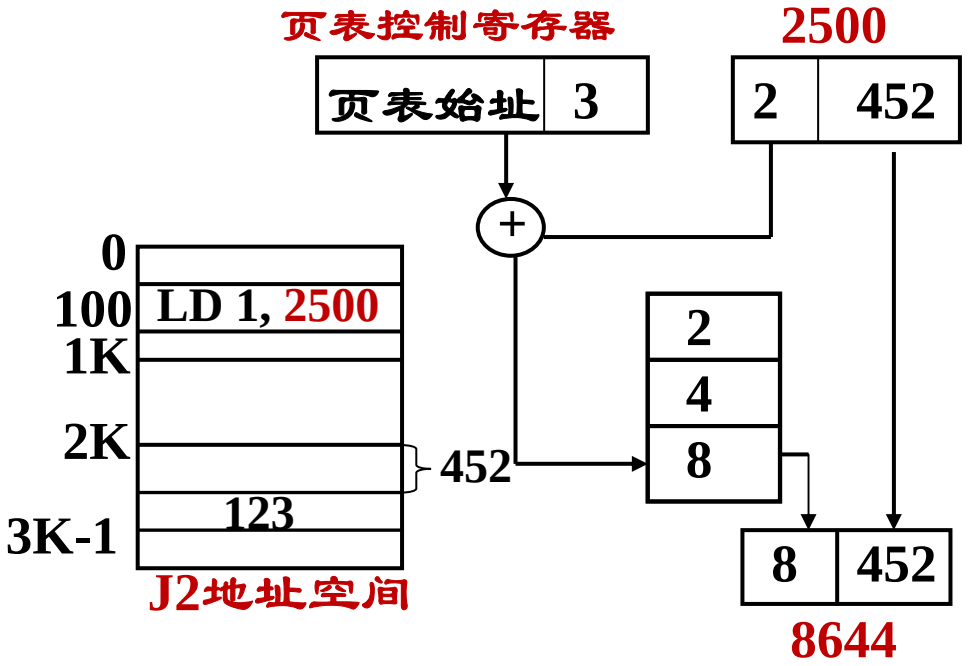


§6 分页存储管理

二、地址重定位

4.地址变换过程

- 将逻辑地址拆分成高位页号P和低位页内偏移量W
- 由页表控制寄存器找到页表始址,以P为索引找到 P页的块号B
- 拼接B和W， 形成物理地址

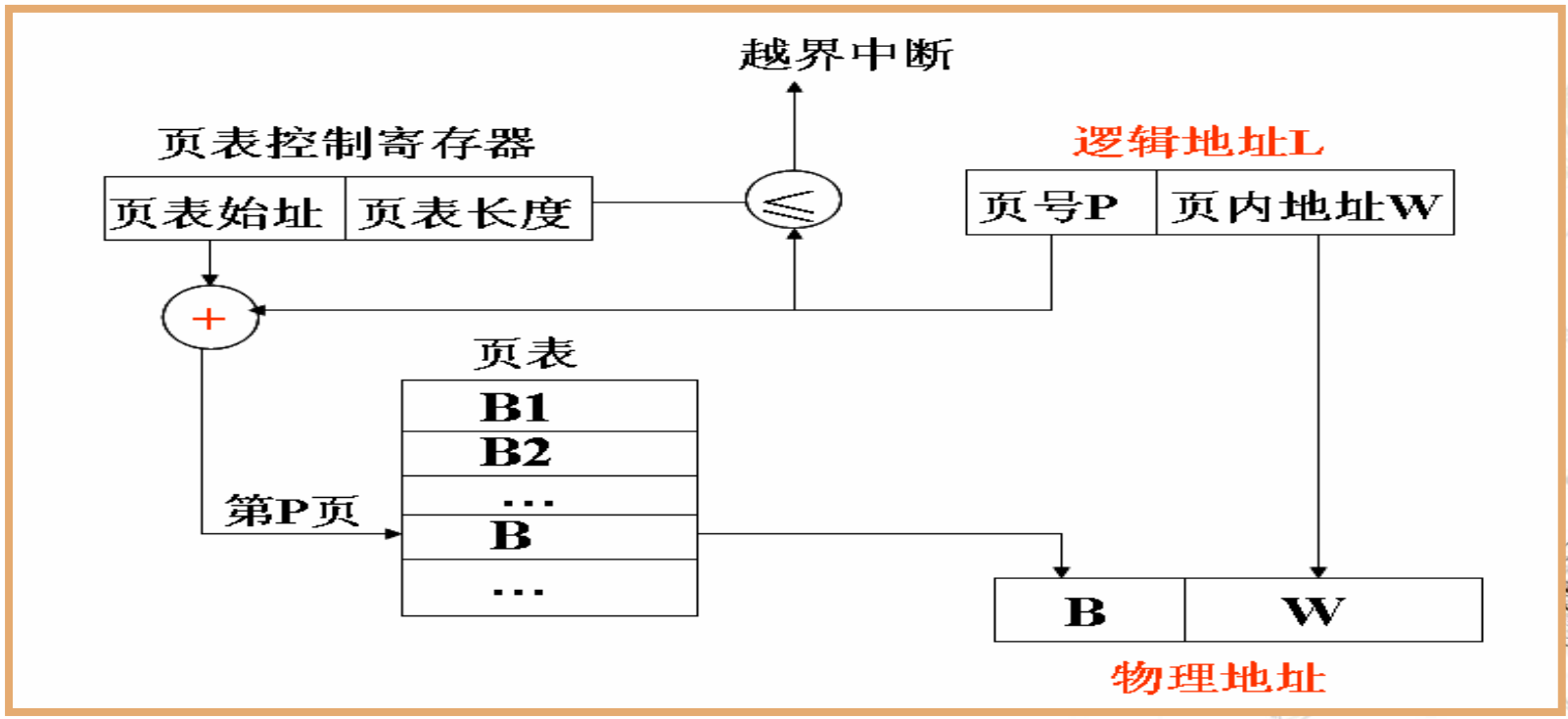


§6 分页存储管理

二、地址重定位

4.地址变换过程

- 将逻辑地址拆分成高位页号P和低位页内偏移量W
- 由页表控制寄存器找到页表始址,以P为索引找到 P页的块号B
- 拼接B和W， 形成物理地址



§6 分页存储管理

例1(南理工)：操作系统采用分页存储管理方法，要求 A 。

A. 每个进程拥有一张页表，且进程的页表驻留在内存中

B. 每个进程拥有一张页表，但只要执行进程的页表驻留在内存中，其它进程的页表不必驻留内存中

C. 所有进程共享一张页表，以节省有限的内存空间，但页表必须驻留在内存中

D. 所有进程共享一张页表，只有页表中当前使用的页面必须驻留内存中，以节省有限的内存空间

例2(华中理工)：分页系统中的页面是为 B 。

A. 用户所感知的　　B. 操作系统所感知的

C. 编译系统所感知的　　D. 链接装配程序所感知的

例3(南理工)：在页式管理中，每个页表中的每个页表项实际上都是用于实现 C 。

A. 内存单元　B. 静态重定位　C. 动态重定位　D. 加载程序

§6 分页存储管理

例4：一分页系统，用户地址空间共有32个页面，每页1KB，主存16KB。某作业页表如图所示。问：

(1) 逻辑地址需用多少位表示，物理地址需用多少位表示？

(2)逻辑地址05ACH对应的物理地址是多少？

(3)逻辑地址3000对应的物理地址是多少？

2
3
1
6

(1) 逻辑地址需15bit，物理地址需14bit。

(2) 【拆分/拼接法】 05ACH=000 ,01 01,1010,1100_2

05ACH地址位于第1页。查PT可知在第3块，故

05ACH的物理地址为：00 ,11 01,1010,1100_2=0DACH

(3) 【计算法】 P=int(逻辑地址/页长)

W=逻辑地址% 页长=逻辑地址-P*页长

P=int(3000/1024)=2，w=3000mod1024=952

由计算知，3000逻辑地址位于第2页第952单元，

查PT知，第2页的主存块号为第1块，故3000对应的

物理地址是：1×1024+952=1976

§6 分页存储管理

例5：一分页系统，逻辑地址长度为16位，页面大小为4096字节，页表如图所示。问：

（1）地址空间有多大？

（2）逻辑地址2F6AH的物理地址是多少？

5
10
11

（1）逻辑地址空间=2^{16}B=64K

（2）2F6AH=0010 ,1111,0110,1010$_2$

⟶ 1011,1111,0110,1010$_2$

=BF6AH

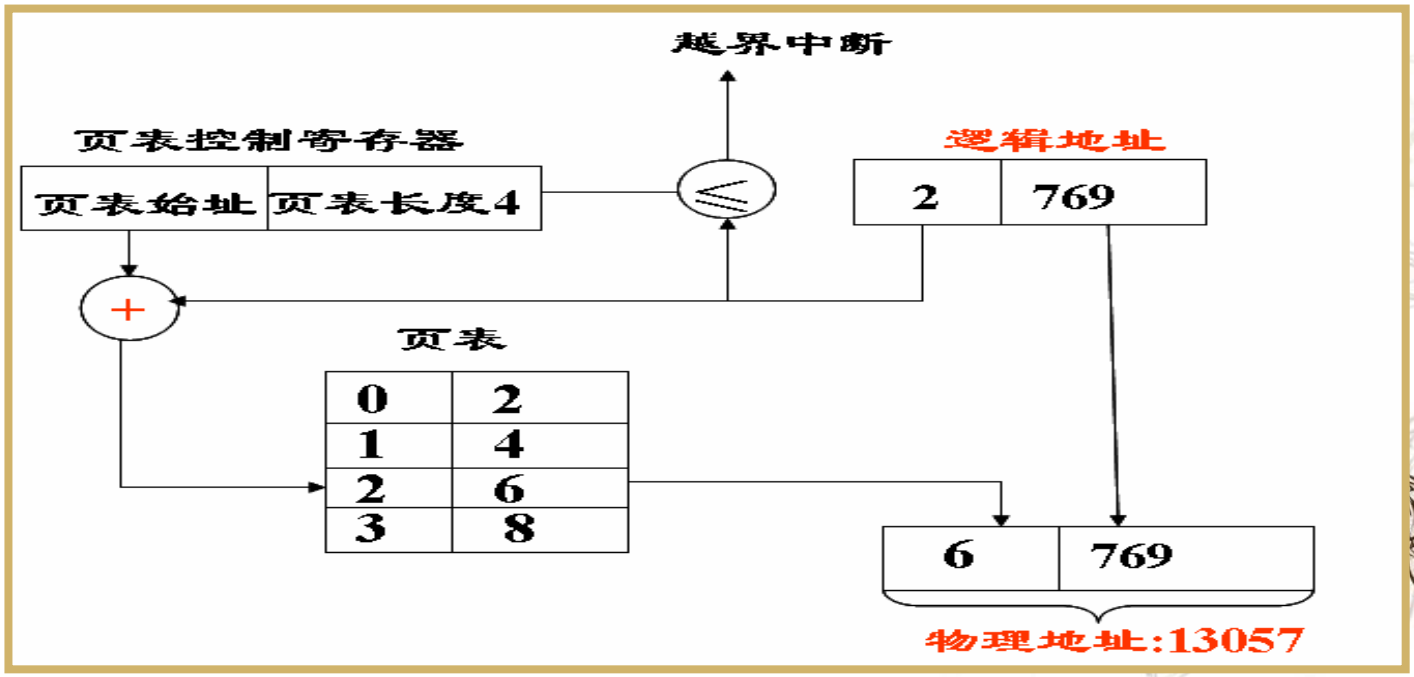
§6 分页存储管理

例6（南开大学）：在采用分页存储的系统中，某作业J的逻辑地址空间为4页，每页2048字节，且已知该作业页面映象表如下表所示，试借助地址变换图（即要求画出地址变换图）求出逻辑地址4865所对应的物理地址。

2
4
6
8

【分析】页号=4865/2048=2, 页内地址=4865-2*2048=769

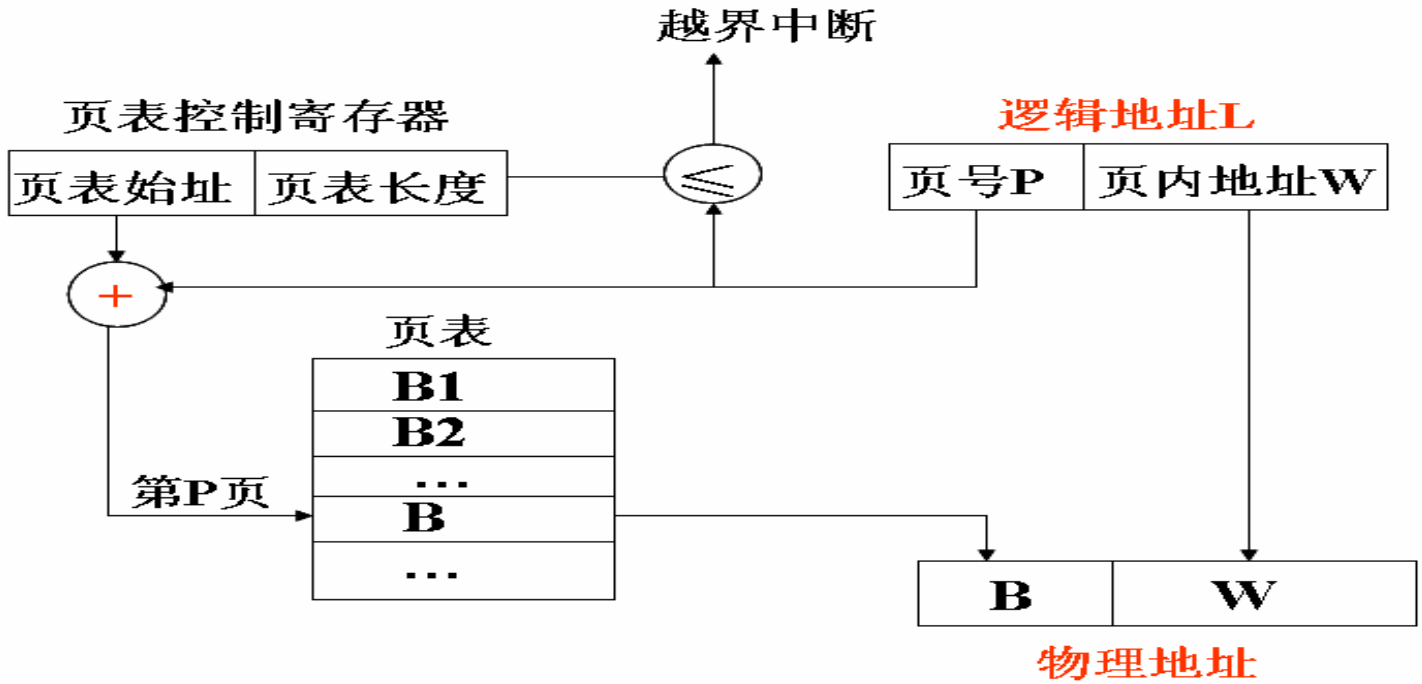
物理地址=6*2048+769=13057



§6 分页存储管理

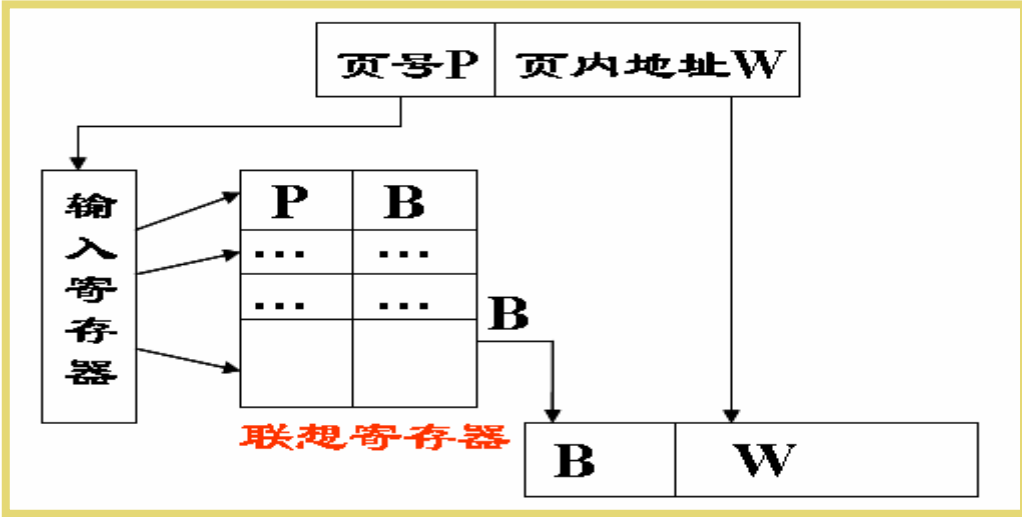
5.基于页表的地址映射 ——访问内存2次，慢表！

- 访问内存页表,获得P页所在块号B,以便拼接形成物理地址
- 根据物理地址,去内存相应单元存取数据或指令



§6 分页存储管理

三、基于快表的地址映射

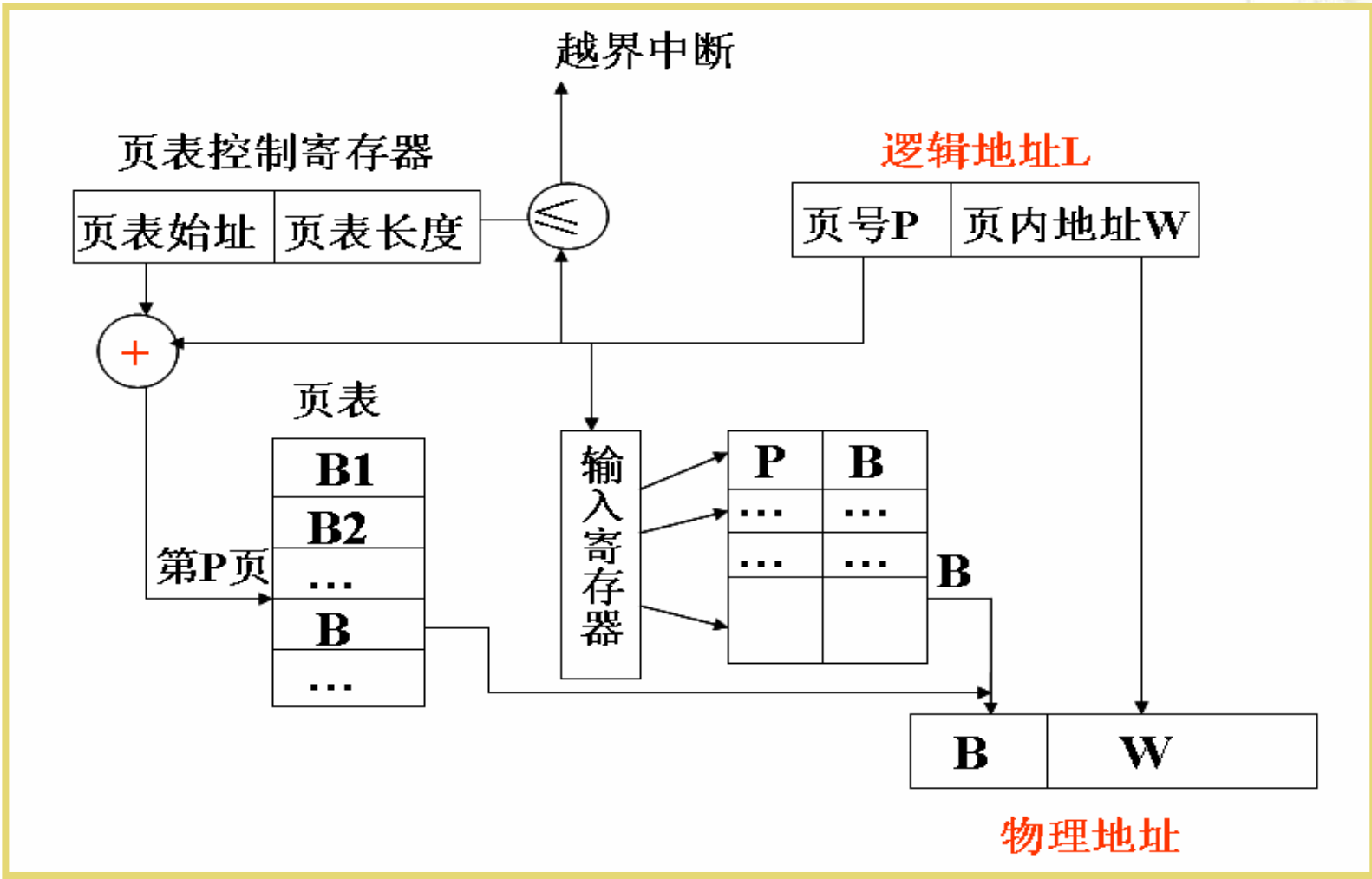


1.联想寄存器(Associative Memory) —— 快表

- 是介于内存与CPU之间的，具有并行查找功能的小容量、高速缓冲存储器。用以存放现行作业的部分页表项。
  - —— 具并行查找功能，故称“联想寄存器”
  - ——访问速度比内存访问速度高一个以上数量级
  - ——快表只存放全部页表的部分活跃页表项，且动态变化

§6 分页存储管理

三、基于快表的地址映射



§6 分页存储管理

2.内存有效访问时间EAT(Effective Access Time)

EAT是指从进程发出指定逻辑地址的访问请求，经过地址变换，到访问到内存中对应物理地址中的信息，所需花费的总时间。

设访问一次内存的时间为t，h为快表命中率，t_1为由快表命中的访问时间，t_2为由页表命中的访问时间。

无快表：EAT=2*t

设置快表：$EAT=h*t_1+(1-h)*t_2$

§6 分页存储管理

2.内存有效访问时间EAT(Effective Access Time)

例：页式系统页表放置于内存，内存访问时间为750ns，问：

（1）系统无快表，则实现一次页面访问的存取时间是多少？

（2）若设置快表，其检索速度为50ns，命中率为80%，则与非页式系统相比，访问速度降低多少？

（3）若命中率提高至85%，且快表检索速度忽略为0，则系统有效访问时间为多少？

【解答】

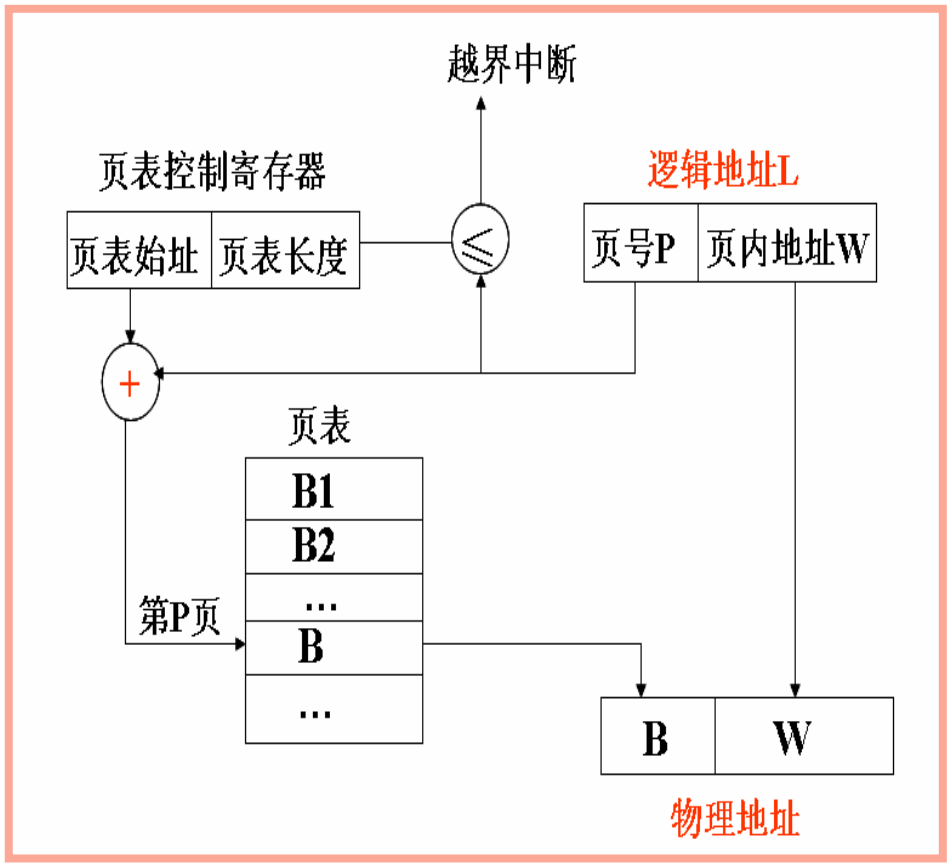
(1)一次页面存取时间T=2*750=1500ns。

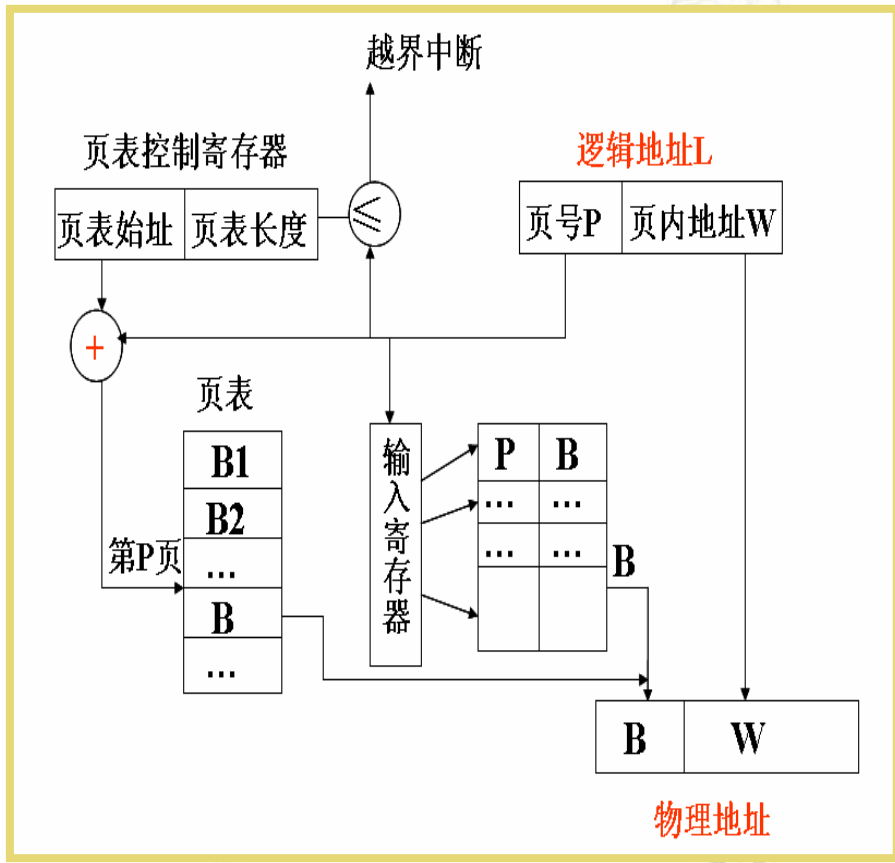
(2)T=80%*(50+750)+20%*(50+750+750)=950ns

(950-750)/750=26.6%

(3)T=85%*750+15%*1500=862.5ns

§6 分页存储管理





【结论】作业的页表需全部保存在内存系统区，且必须为每个页表分配连续的主存空间！

§6 分页存储管理

四、二级页表

【背景】系统逻辑地址长32位，页面尺寸为4KB，每个页表项为4Byte。试分析一道作业所需页表空间。

【分析】

逻辑地址空间=2^{32}B=$2^{20}\times 2^{12}=2^{20}\times 4$K

地址空间共：

1M个页面，即1024×1024个页面。

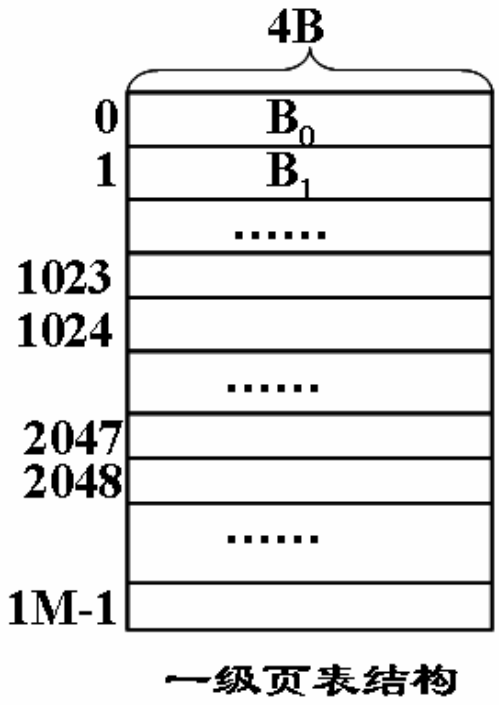
页表空间=页表长×页表项长=1M*4B=4M

【分析】

在一级页表中：

每1024个页表项形成1个外部页面，

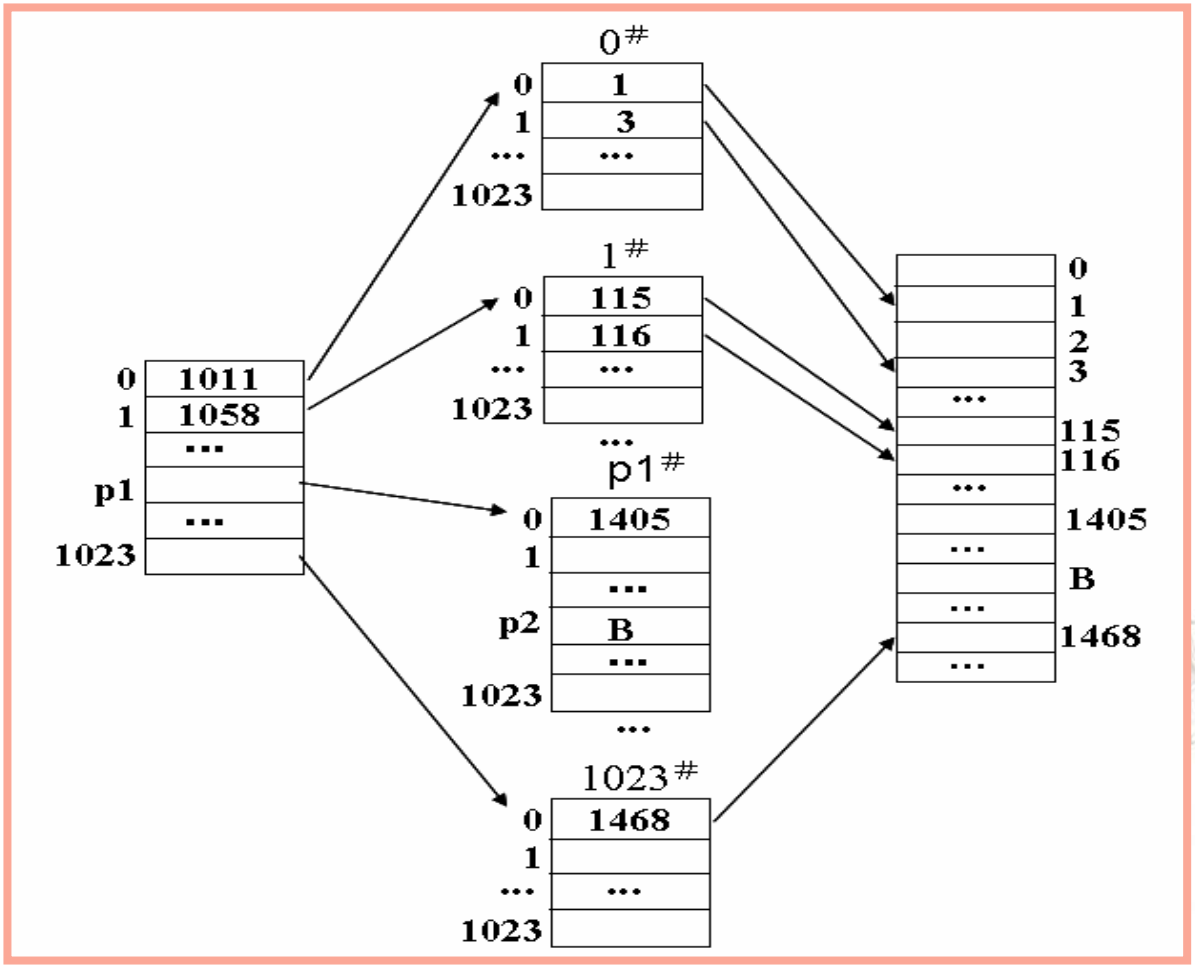
一级页表共含有1024个外部页面。

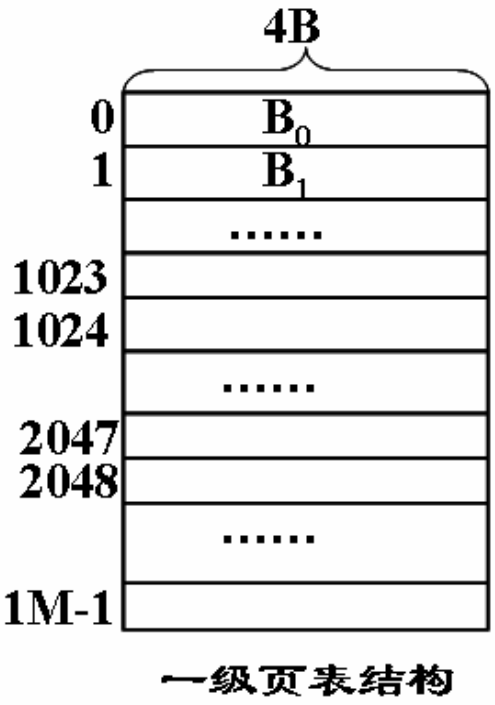


一级页表结构

§6 分页存储管理

四、二级页表

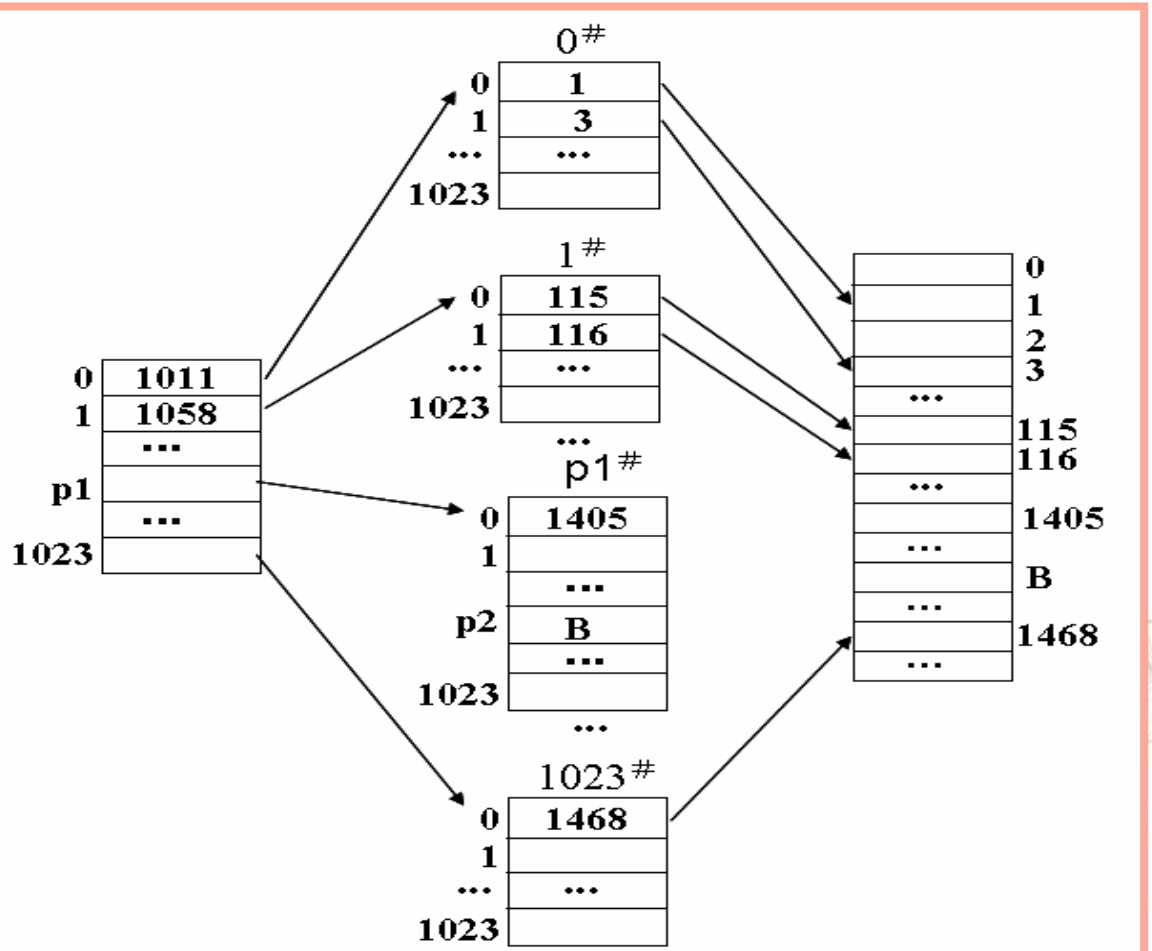




一级页表结构

§6 分页存储管理

四、二级页表



1.等分页表为$0^{\#}$,$1^{\#}$....$n^{\#}$页,使外部页面大小等于块长

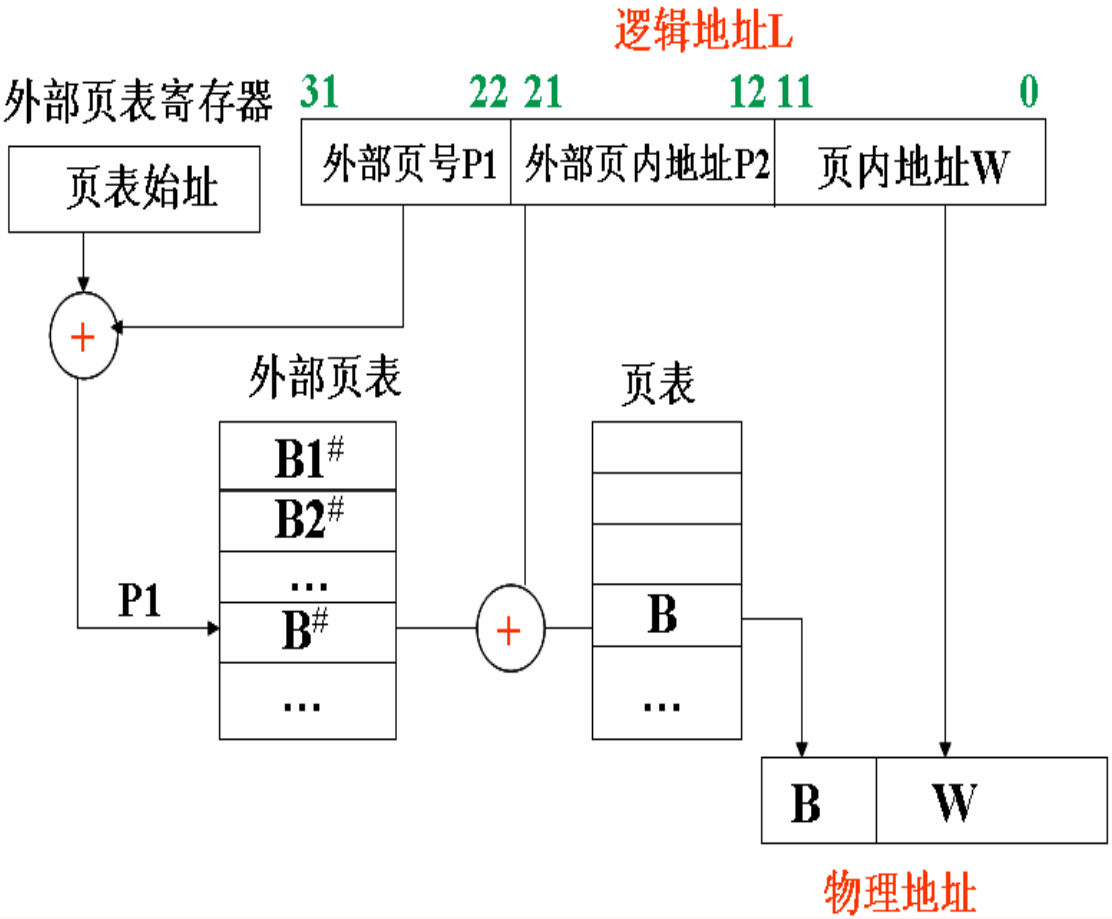
2.离散地将外部页面放置到主存系统区的不同块中

3.为离散分配的页表建立外部页表,说明外部页面号及块号

4.除一级页表外，只将部分二级页表放置在主存

§6 分页存储管理

四、二级页表



1.等分页表为$0^{\#}$,$1^{\#}$....$n^{\#}$页,使外部页面大小等于块长

2.离散地将外部页面放置到主存系统区的不同块中

3.为离散分配的页表建立外部页表,说明外部页面号及块号

4.除一级页表外，只将部分二级页表放置在主存

§6 分页存储管理

例1(南航):在具有两级页表的分页存储管理系统中，CPU每次要存取一个数据时，须访问 3 次内存。

例2(南航):一个使用32位虚地址的计算机系统使用两级页表，虚地址被分成10位的顶级页表域、10位二级页表域、12位偏移。则页面长度是 4K ，在虚空间中共有 1M 页。

例3(2010年):某计算机采用二级页表的分页存储管理方式，按字节编制，页大小为2^{10}字节，页表项大小为2字节，逻辑地址结构如图示。逻辑地址空间大小为2^{16}页，则表示整个逻辑地址空间的页目录表中包含表项的个数至少是 B 。

A.64 B.128 C.256 D.512

页目录号	页号	页内偏移量

§6 分页存储管理

五、反置页表IPT

Inverted Page Table以主存块为序、为每个物理块设置一个页表项，表明该块所放置的页面号及其隶属进程标识符。

	虚页号	进程标识符
0块	P_1	pid1
1块	P_2	pid2
	…	…
n-1块		

图：反置页表构成形式

例：一个64位的计算机，其内存64M，块长4K，页表项4B。试计算PT和IPT所需内存空间。

(1)PT空间

地址空间=2^{64}B=$2^{52}\times 2^{12}=2^{52}\times$4K

PT空间=PT表长×页表项=$2^{52}\times 2^2$

$\approx 1.8\times 10^{16}>10^{15}$B(一百万GB)

(2)IPT空间

内存=64M=2^{26}B=$2^{14}\times 2^{12}$

IPT空间=IPT表长×页表项=$2^{14}\times 2^2$=64K

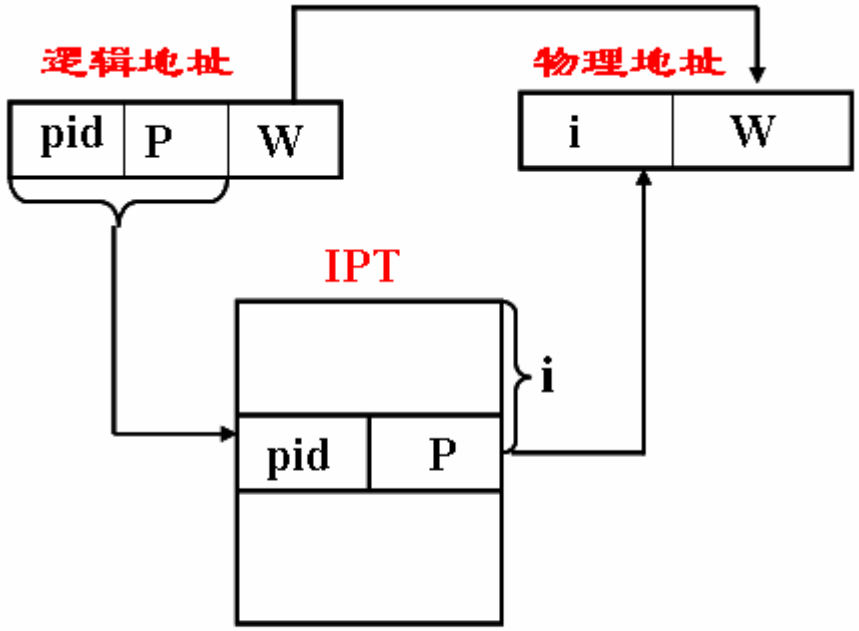
§6 分页存储管理

五、反置页表IPT

Inverted Page Table以主存块为序、为每个物理块设置一个页表项，表明该块所放置的页面号及其隶属进程标识符。

	虚页号	进程标识符
0块	P_1	pid1
1块	P_2	pid2
	…	…
n-1块		

图：反置页表构成形式



图：基于IPT的地址映射图

§6 分页存储管理

六、页式系统的特点

1.离散的主存分配技术，有效解决了主存“碎片”，改善了主存利用率。

2.需要设置硬件地址变换机构，以实现地址映射。

3.为提高速度，需设置“快表”。

4.页式系统的“内零头”：平均1/2个页长。

5.页式系统的“外零头”：页表、二级及多级页表、反置页表的开销。

§7 分段存储管理

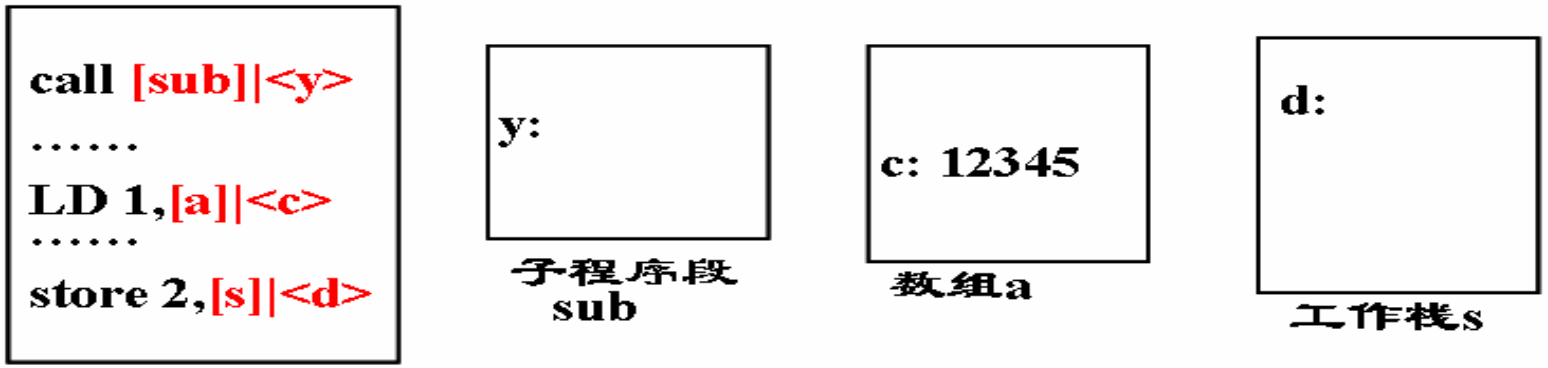
一、一维线性地址空间的不足之处

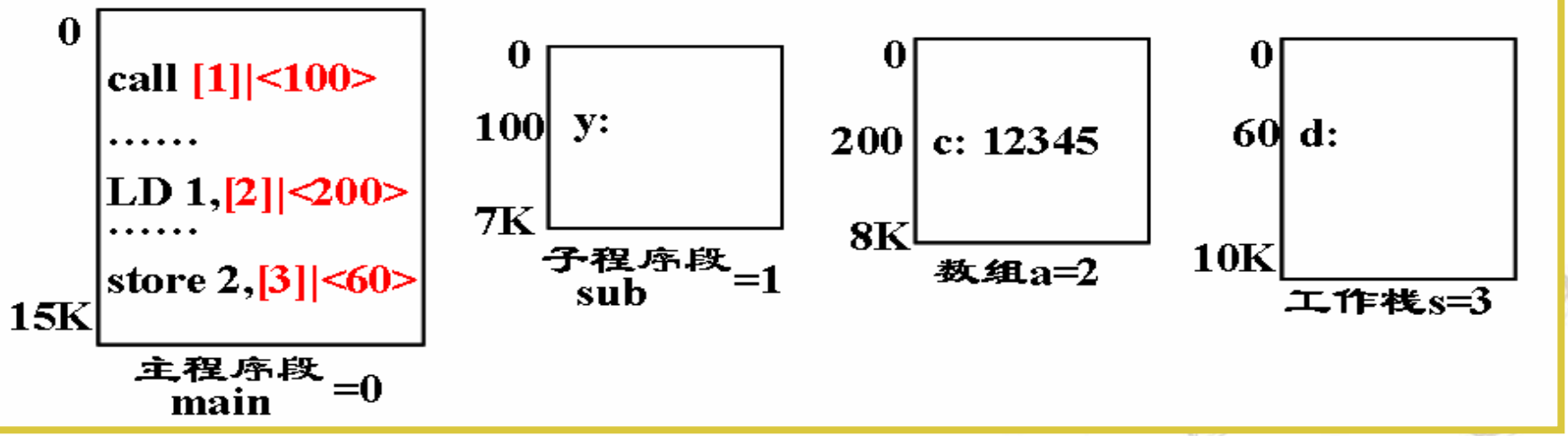
- **难以清晰表达程序的逻辑结构关系。** 应该为程序中的每个模块和数据提供独立地址空间，地址空间应由多个独立段组成。
- **难以方便地实现信息共享。** 因为信息共享是以逻辑单位为基础的。段才是信息的逻辑单位，页面不是。
- **难以方便地实现信息保护。** 因为信息保护也是以逻辑单位为基础的，即以段作为信息保护单位，页面不是。
- **难以实现数据的动态增长。** 数据的动态增长在多段式的二维空间中易于实现。
- **难以实现动态链接。** 因为程序的动态链接是以目标程序(即段)为基本链接单位的。

结论： 为满足用户编程多方面需求，作业的地址空间应是一个由多个独立的段所组成的**二维多段式**的地址空间。

§7 分段存储管理

二、分段系统的空间





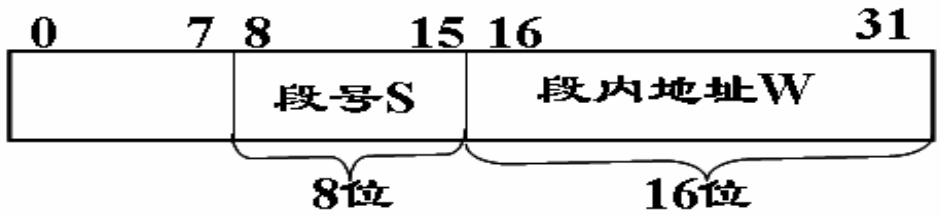
§7 分段存储管理

三、分段基本原理

1.作业的逻辑地址:是由段号S和段内地址W构成的二维地址(S,W)

2 .作业运行前,装入全部段。每段装入主存连续区域，段与段之间不要求连续。

例:在IBM 370系统中,字长为32位,指令地址24位,位于第8~31位处,其中低8位为段号,高16位为段内地址。问：作业最多可有几段？每段最大长度是多少？作业地址空间有多大？



段数=2^8=256 段

最大段长=2^{16}B=64K

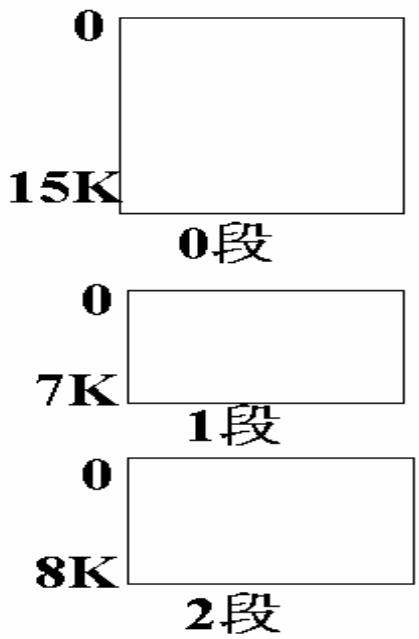
地址空间=2^8*2^{16}B=2^{24}B=16M

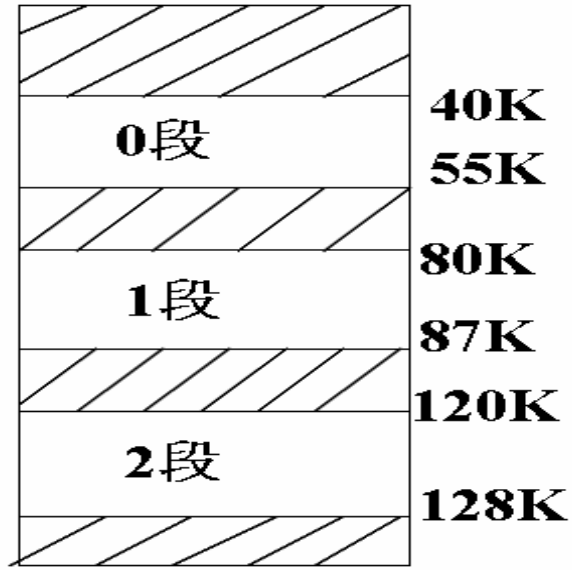
§7 分段存储管理

三、分段基本原理

1.作业的逻辑地址:是由段号S和段内地址W构成的二维地址(S,W)

2 .作业运行前,装入全部段。每段装入主存连续区域，段与段之间不要求连续。

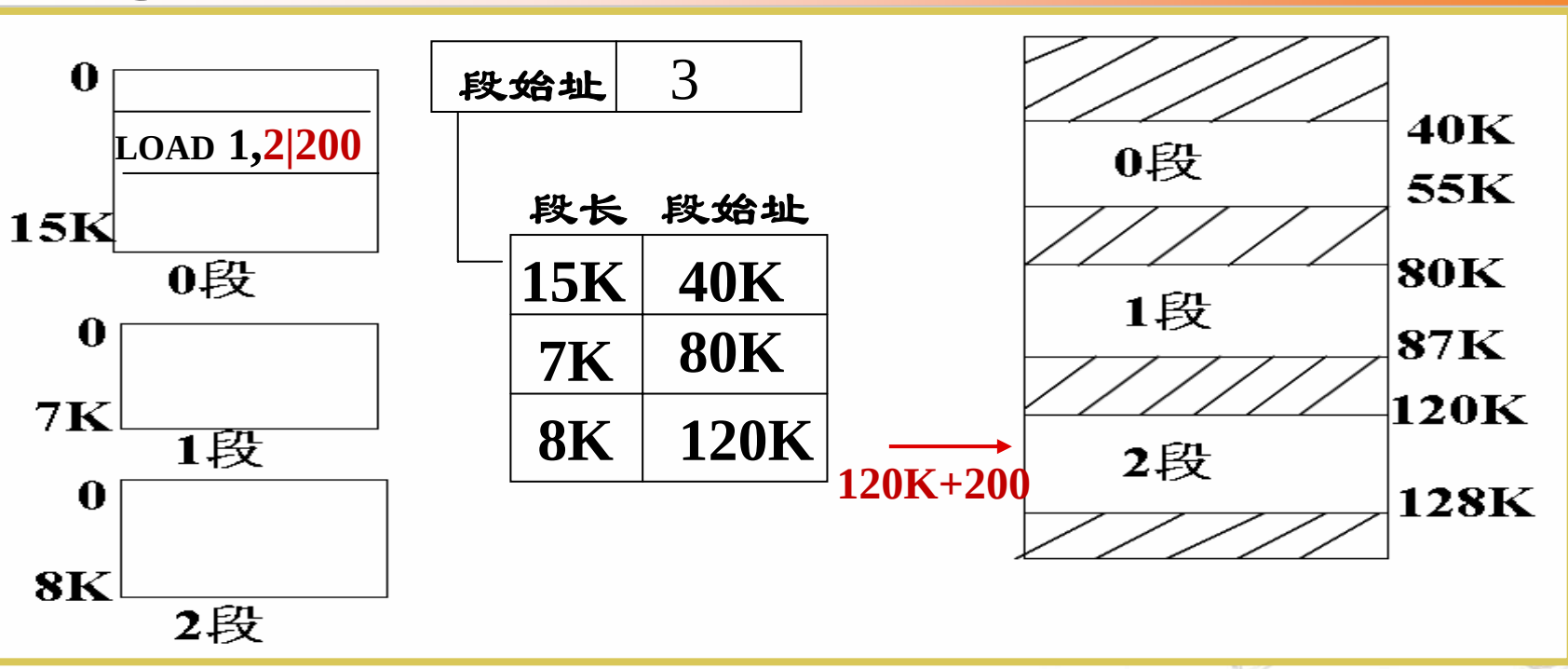




§7 分段存储管理

四、地址重定位

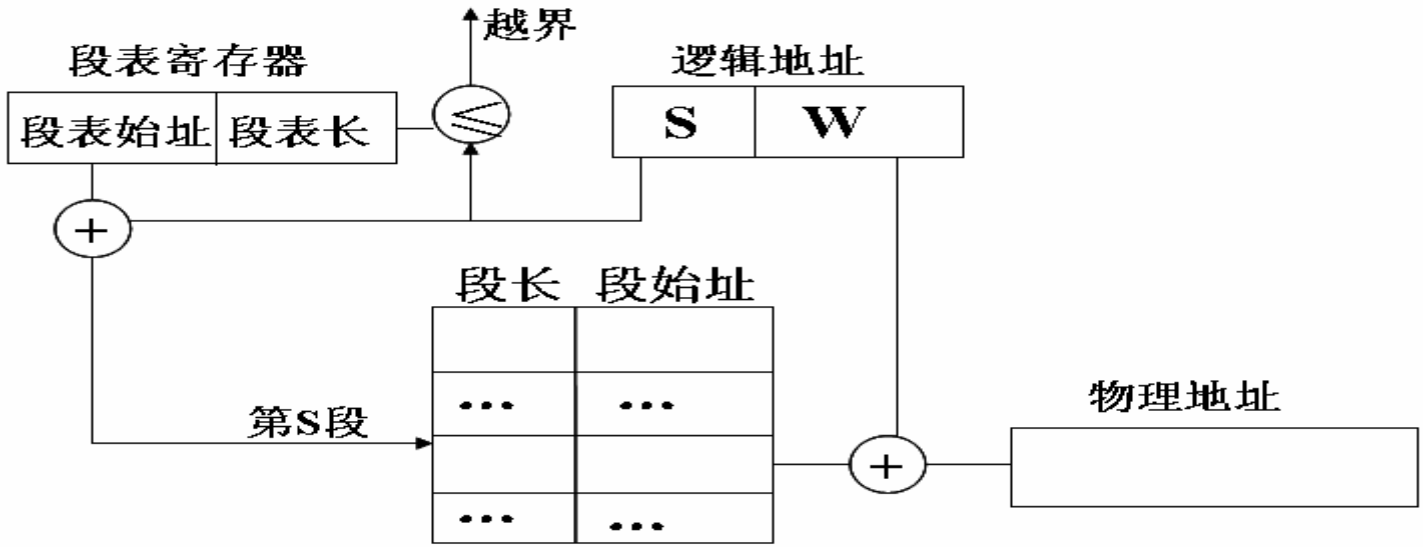
1. 段表：以段号为序记录各段长度及主存始址
2. 段表控制寄存器：记录现行进程段表长和段表始址
3. 对任一逻辑地址(S,W)，地址变换分二步完成：①由段表控制寄存器找到段表始址，以段号S为索引，在段表中找到S段内存始址。②将S段内存始址和W相加即获得物理地址。



§7 分段存储管理

四、地址重定位

1. 段表：以段号为序记录各段长度及主存始址
2. 段表控制寄存器：记录现行进程段表长和段表始址
3. 对任一逻辑地址(S,W)，地址变换分二步完成：①由段表控制寄存器找到段表始址，以段号S为索引，在段表中找到S段内存始址。②将S段内存始址和W相加即获得物理地址。



同分页系统，获取内存指令或数据需访问内存2次

§7 分段存储管理

例（浙大）：某系统采用段式存储管理，一作业段表如下。试计算逻辑地址[0,124]、[3,240]、[2,532]、[5,32]所对应的物理地址，并给出[2,532]的地址映射过程。

段号	基地址	长度
0	340	300
1	1300	500
2	2650	750
3	3870	200

[0,124]对应的物理地址是：340+124=464

[3,240]：地址非法，产生段内地址越界中断

[2,532]对应的物理地址是:2650+532=3182

[5,32]：地址不合法，产生地址越段中断

§7 分段存储管理

五、分段和分页的区别

1.段是信息的逻辑单位，分段是出于用户应用的需要,故段的划分是用户行为，对用户是不透明(可见)的。

页是信息的物理单位，分页是出于系统管理的需要，故页的划分是系统行为，对用户是透明(不可见)的。

2. 各段长度通常不等，由其完成的功能决定(物理地址要做+)。

页的长度固定不变，各页等长，由系统决定(物理地址拼接)。

3. 分段系统提供的是二维多段式地址空间,编程采用二维地址。

分页系统向用户提供的是一维线性地址空间，其页号和页内地址拆分是机器硬件的功能。

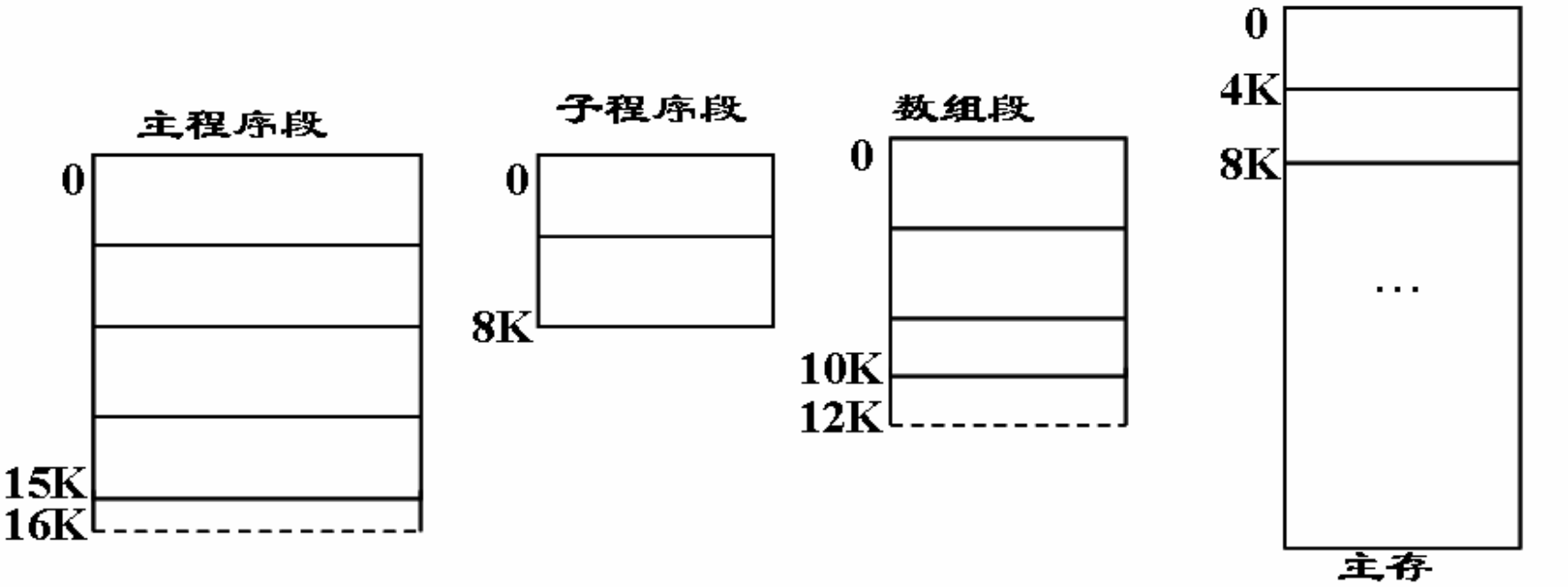
4.由于段是信息的逻辑单位，故分段便于信息保护和共享。

页是信息的物理单位，故页的保护和共享收到限制。

§8 段页式存储管理

一、段页式原理

- 作业的地址空间按段划分，是一个二维虚空间
- 主存等分为实页面,相应地,作业的各段等分为虚页面
- 作业运行前装入作业所有段的所有页，一页装入一块

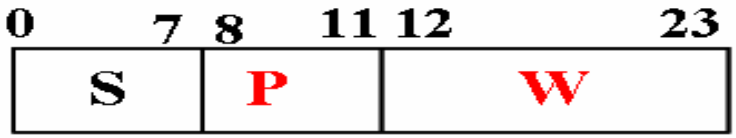


§8 段页式存储管理

二、地址重定位

1.逻辑地址拆分 ——由地址变换机构将(S,W')拆分成(S,P,W)

例：假设CPU有效地址长24位，其中段号S占8位，段内位移量W'占16位，块长为4K。



【分析】段长=2^{16}=64K，每页4K，共16页。
故16位的段内地址前4位表示页号P，
后12位表示页内地址W。

§8 段页式存储管理

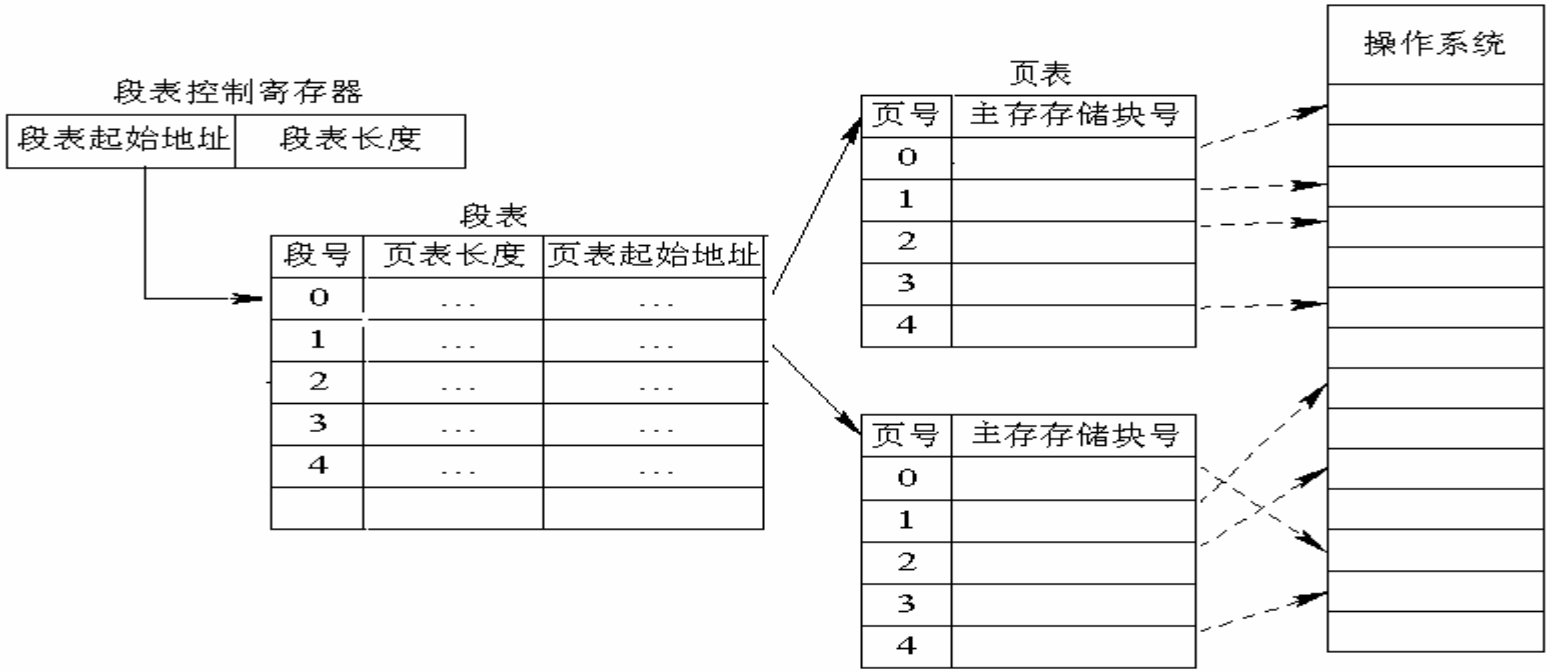
二、地址重定位

1.逻辑地址拆分 ——由地址变换机构将(S,W')拆分成(S,P,W)

2.为各段建立一张页表：记录该段中各页所在的主存块号

3.为作业建立一张段表：记录各段的页表长及页表始址

4.段表控制寄存器:设置当前作业的段表长及段表始址。

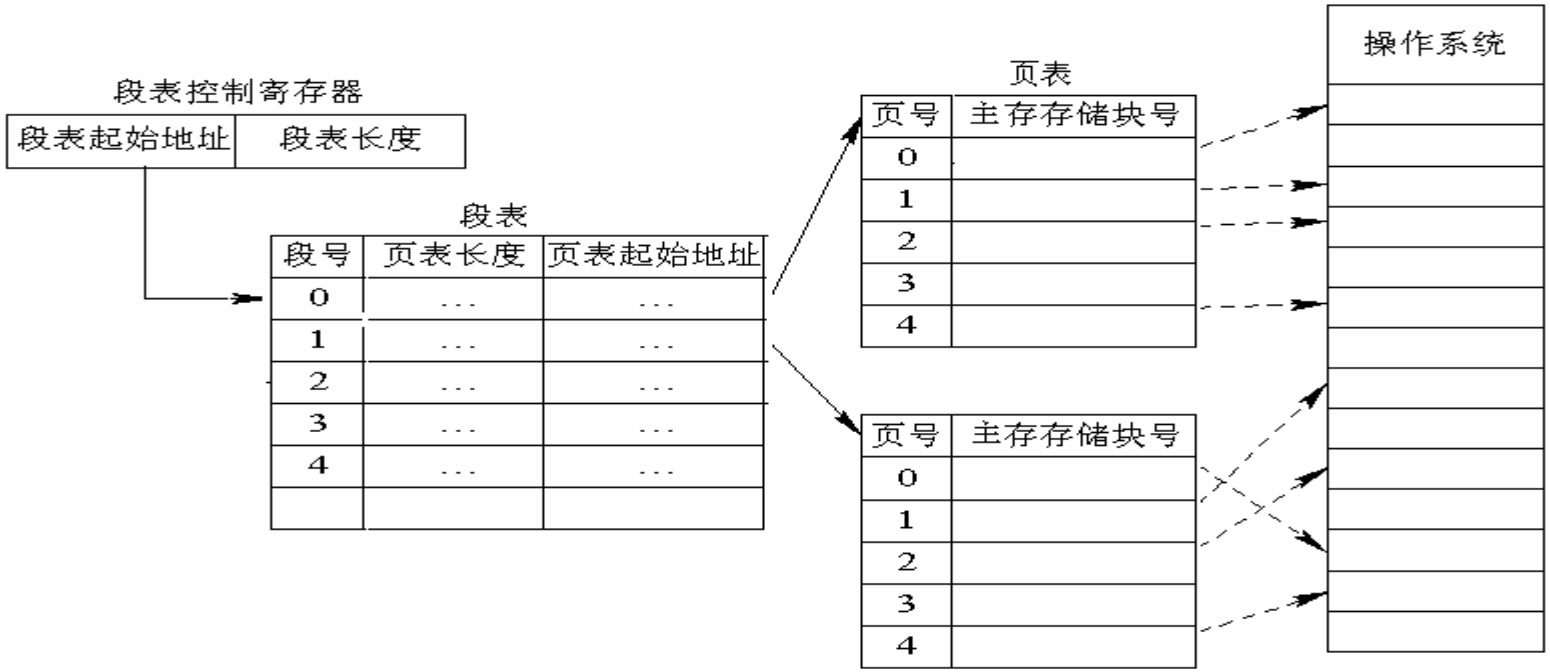


§8 段页式存储管理

二、地址重定位

5.对任一逻辑地址(S,P,W)，其地址变换分三步完成：

①由段表控制寄存器找到段表始址，以段号S为索引，在段表中找到页表始址。②以页号P为索引，在页表中获得主存块号B。③拼接B和W即形成主存物理地址。

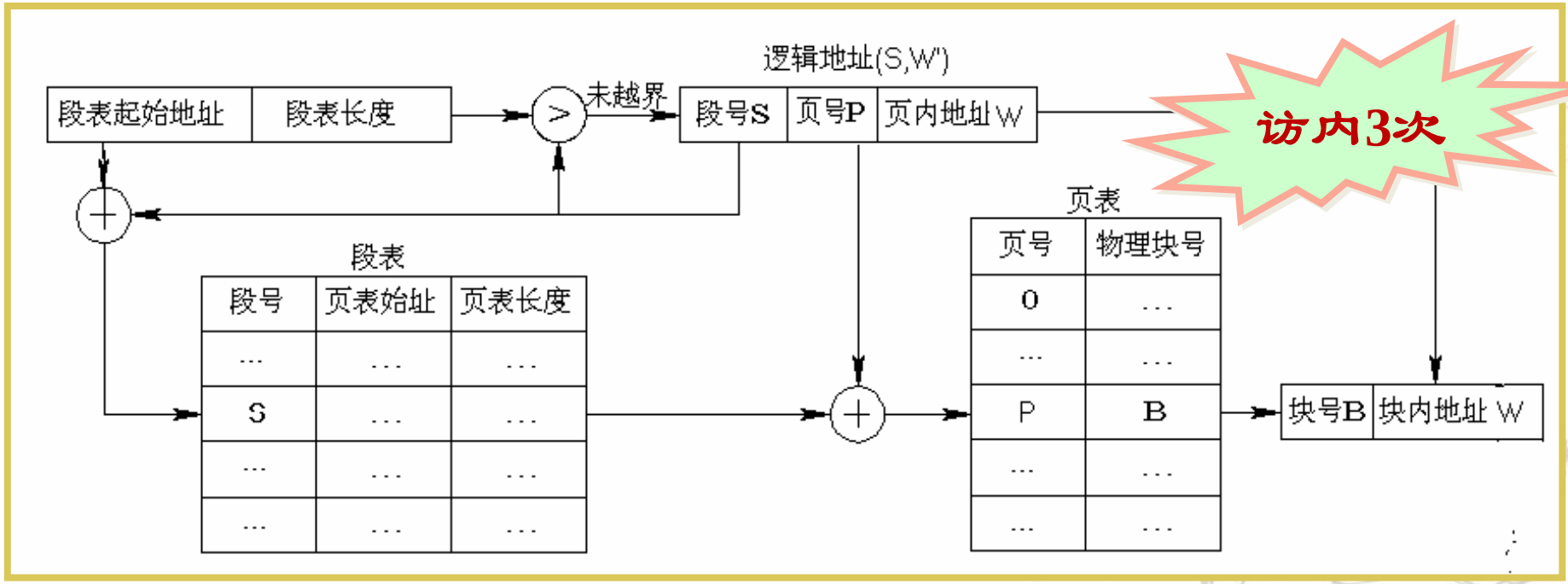


§8 段页式存储管理

二、地址重定位

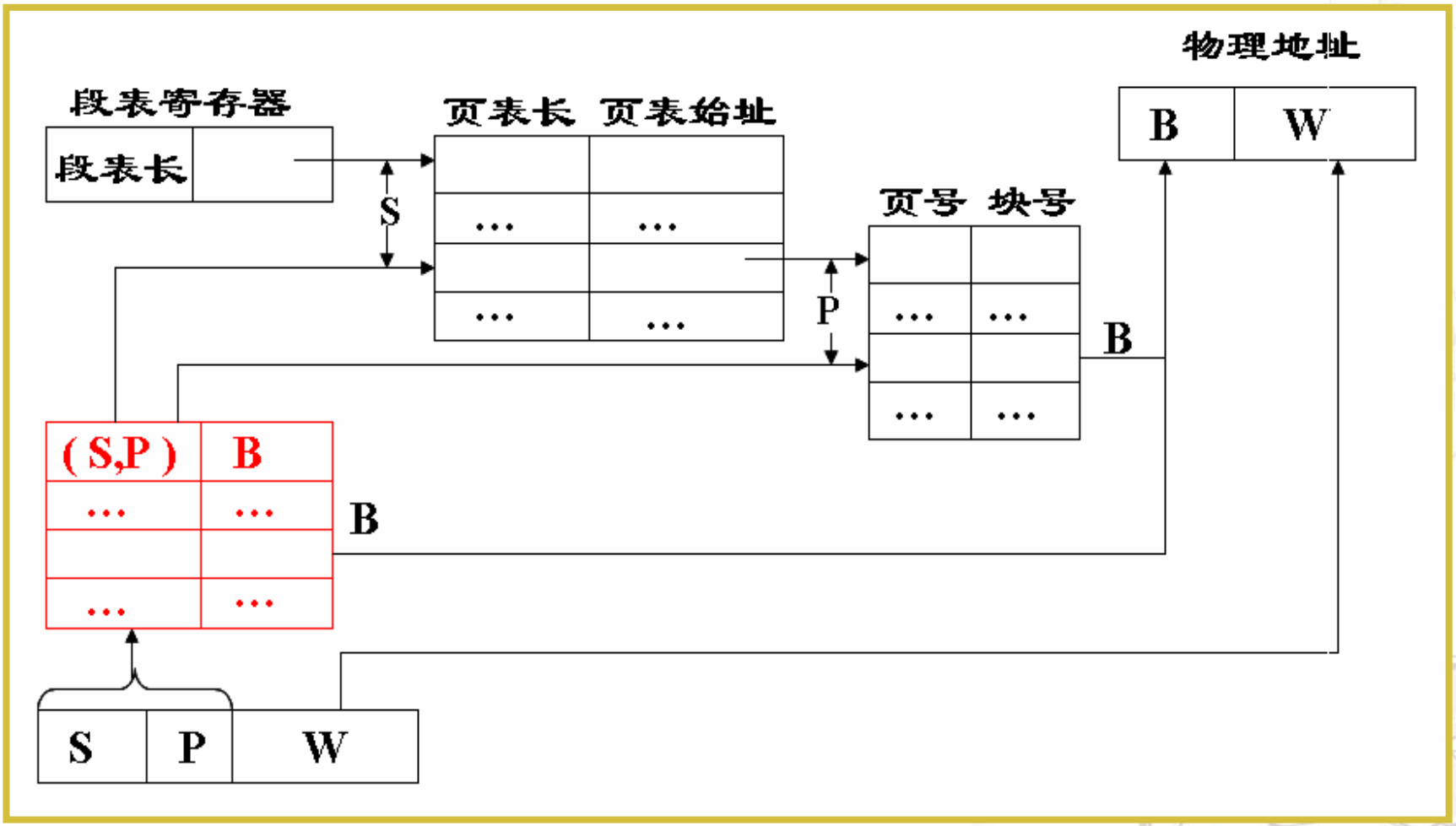
5.对任一逻辑地址(S,P,W)，其地址变换分三步完成：

①由段表控制寄存器找到段表始址，以段号S为索引，在段表中找到页表始址。②以页号P为索引，在页表中获得主存块号B。③**拼接**B和W即形成主存物理地址。



§8 段页式存储管理

三、基于快表的地址映射



§8 段页式存储管理

例1(东大)：段页式存储管理中，其逻辑地址是 B 。

A. 一维 B.二维 C.三维 D.不确定

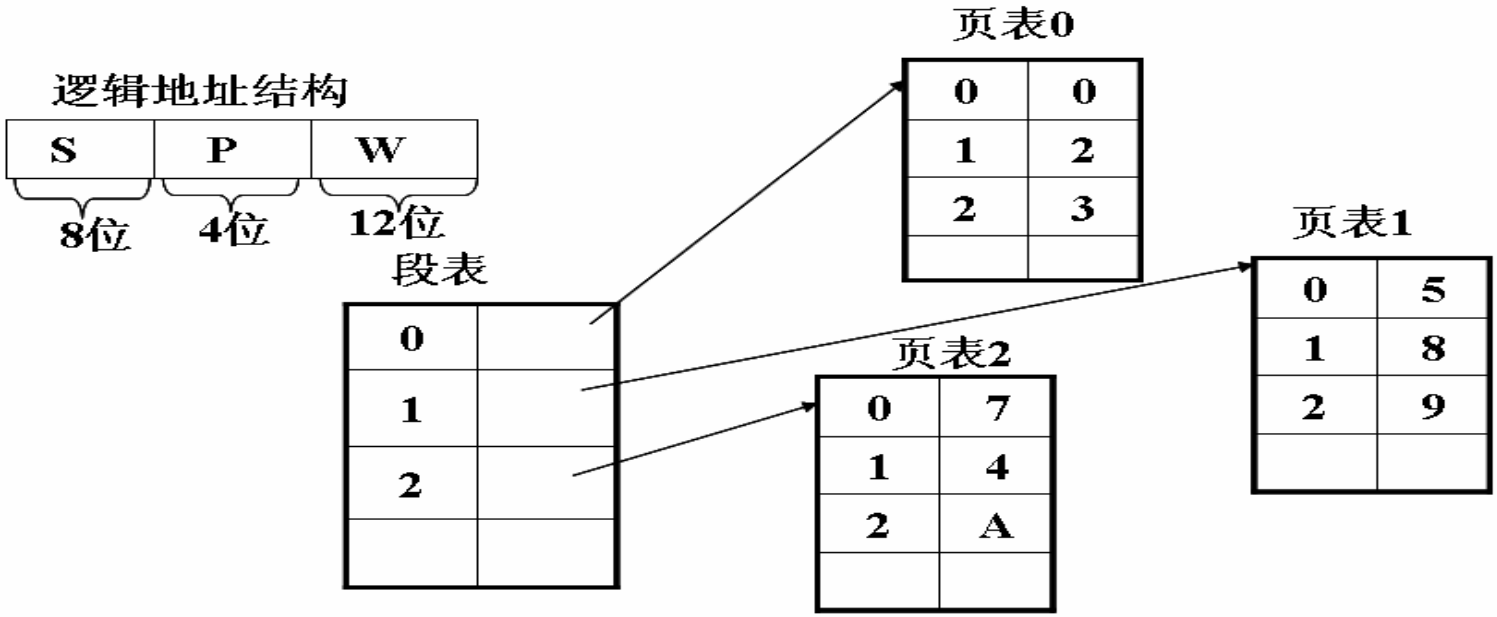
例2 (东大): 在段页式系统中（无快表），为获得一条指令或数据都需三次访问内存。第一次从内存取得 页表始址，第二次从内存取得 物理块号，第三次从内存取得指令或数据。

例3(苏大): 段式存储系统的内存保护要进行两次判断，其一是判断逻辑地址的 段号 是否超过 段表长，其二是判断 段内地址 是否超过 段长 。

段页式存储管理的内存保护也要进行两次判断，其一是判断逻辑地址的 段号 是否超过段表长度，其二是判断 页号 是否大于该段页表长度。

§8 段页式存储管理

例4 (南航) :假设段页式系统有关数据结构如图所示。求虚地址69732的物理地址（用十进制表示）。



解：$69732_{10}=10001000001100100_2$，虚地址位于1段1页，查段页表知在主存第8块。实地址=$1000\ 000001100100_2=32868_{10}$