西安交通大学
软件学院

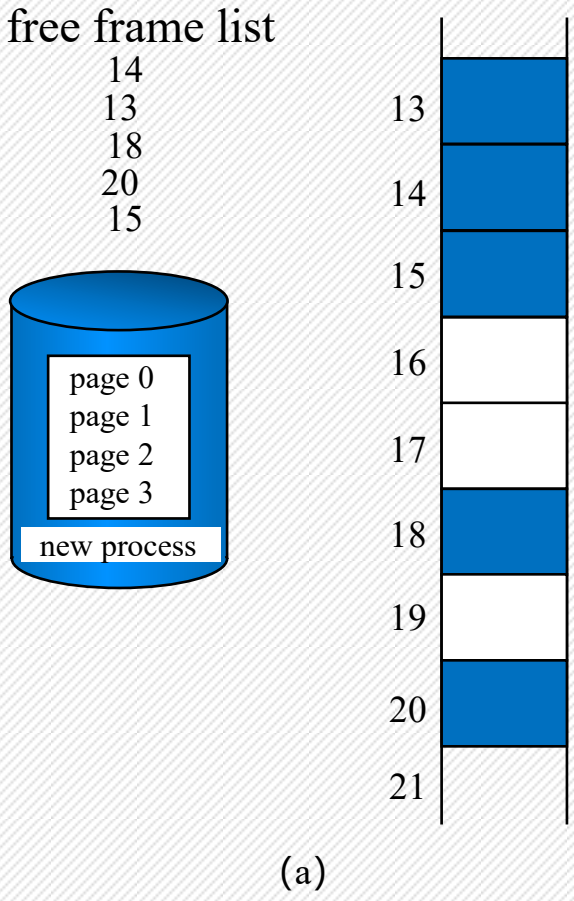
操作系统原理

Operating System Principle

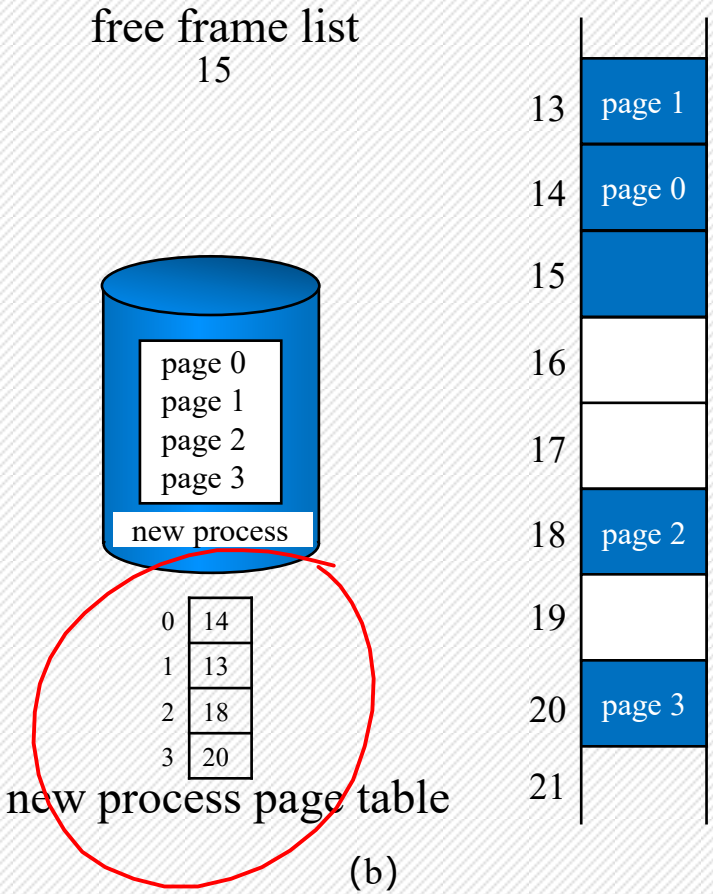
田丽华

8-5 分页硬件支持

Paging Example



(a)



(b)

Implementation of Page Table

- Page table is kept in main memory.（页表被保存在主存中）
- *Page-table base register* (PTBR) points to the page table.（页表基址寄存器指向页表）
- *Page-table length register* (PRLR) indicates size of the page table.（页表限长寄存器表明页表的长度）
- In this scheme every data/instruction access requires two memory accesses. One for the page table and one for the data/instruction.（在这个机制中，每一次的数据/指令存取需要两次内存存取，一次是存取页表，一次是存取数据）
- The two memory access problem can be solved by the use of a special fast-lookup hardware cache called *associative registers or translation look-aside buffers* (*TLBs*). (*通过一个联想寄存器，可以解决两次存取的问题*)

Associative Register

Associative registers – parallel search （联想寄存器—并行查找）

Page #	Frame #

Address translation (p, d)（地址转换）

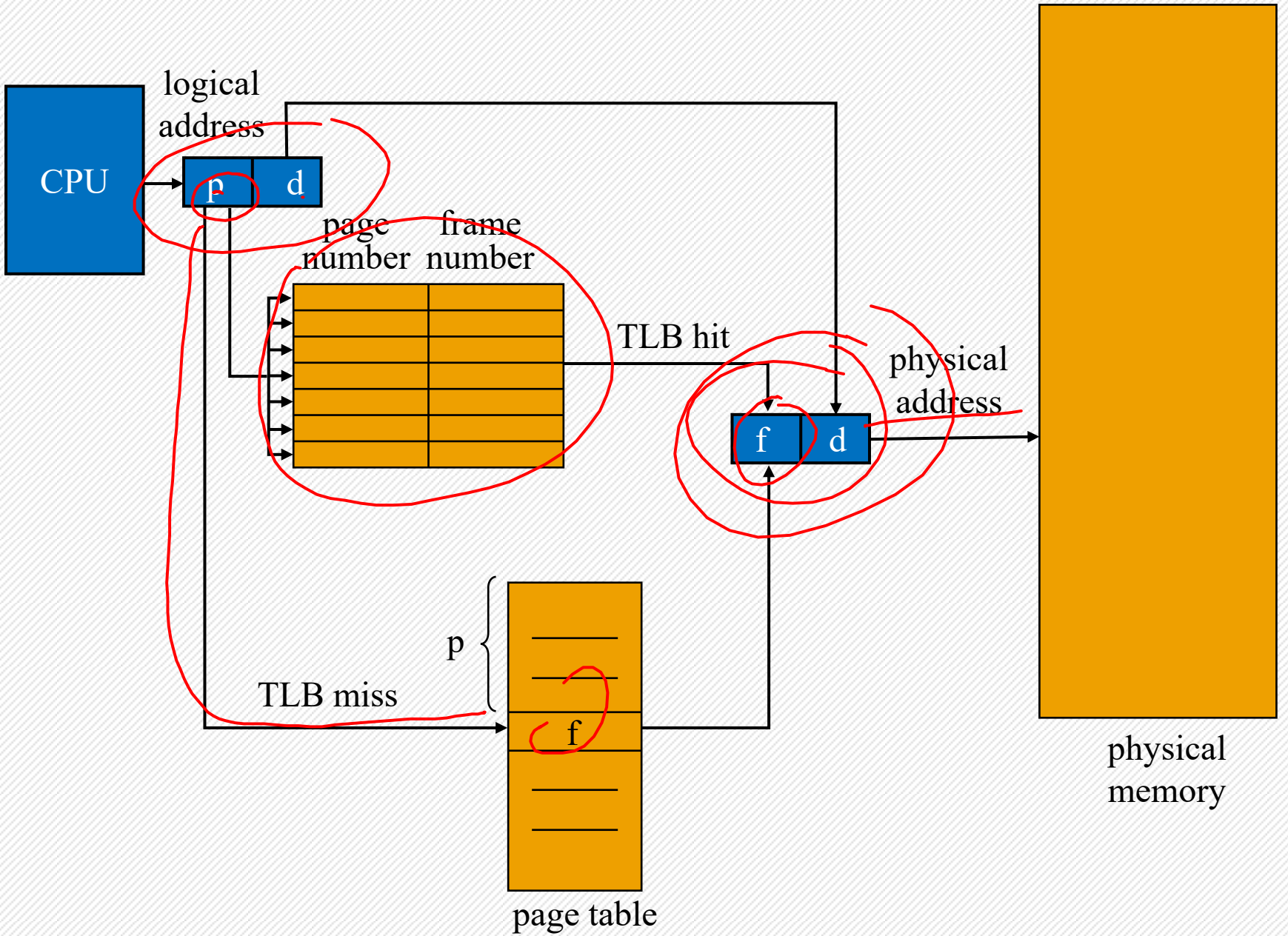
If p is in associative register, get frame # out.
（如果p在联想寄存器中，把页框#取出来。）

1

2

Otherwise get frame # from page table in memory.（否则从内存中的页表中取出页框#。）

Implementation of Page Table



Effective Access Time

- Associative Lookup = ε time unit（联想寄存器的查找需要时间ε）
- Assume memory cycle time is 1 microsecond（假设内存一次存取要1微秒）
- Hit ration – percentage of times that a page number is found in the associative registers; ration related to number of associative registers.（命中率—在联想寄存器中找到页号的比率，比率与联想寄存器的大小有关。）
- Hit ratio = α
- Effective Access Time (EAT)（有效存取时间）

$$EAT = (1 + \varepsilon)\alpha + (2 + \varepsilon)(1 - \alpha)$$

$$= 2 + \varepsilon - \alpha$$

Effective Access Time

例如，假设检索联想存储器的时间为20ns，访问内存的时间为100ns，访问联想存储器的命中率为85%，则CPU存取一个数据的平均时间:

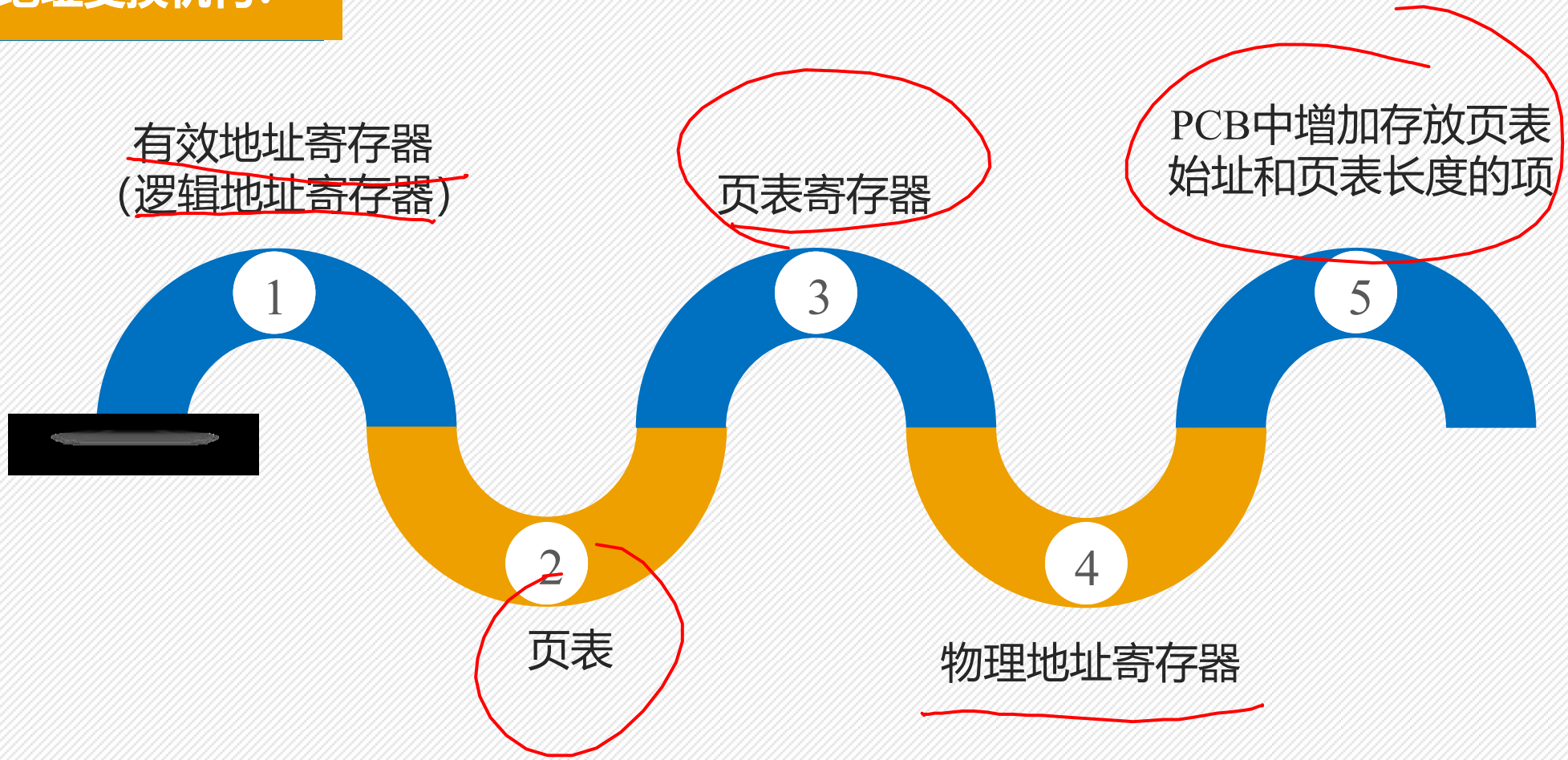
- T=0.85*120+0.15*220=135ns，

访问时间只增加35%。如果不引入联想存储器，其访问将延长一倍（达200ns）。

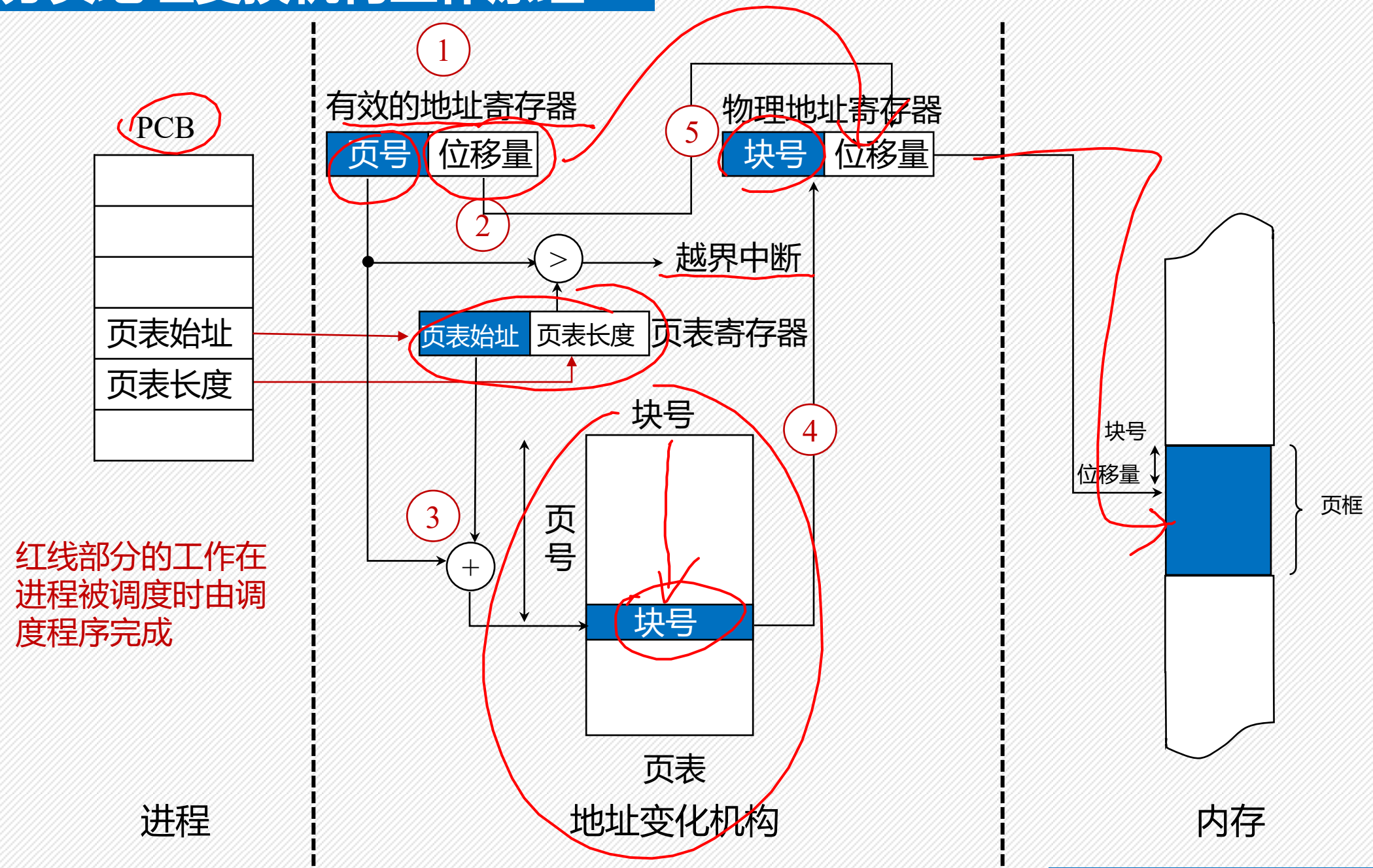
基本的地址变换机构

实现从逻辑地址到物理地址的转换:将用户程序中的页号变换成内存中的物理块号

地址变换机构：



分页地址变换机构工作原理



红线部分的工作在进程被调度时由调度程序完成

地址变换过程

01 按页的大小分离出页号和位移量，放入有效地址寄存器中

02 将页号与页表长度进行比较，如果页号大于页表长度，越界中断；

03 以页号为索引查找页表:将页表始址与页号和页表项长度的乘积相加，便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；

04 将该物理块号装入物理地址寄存器的高址部分；

05 将有效地址寄存器中的位移量直接复制到物理地址寄存器的低位部分，从而形成内存地址。

页式地址变换举例