按需调页 (Demand Paging)

按需调页 (Demand Paging)

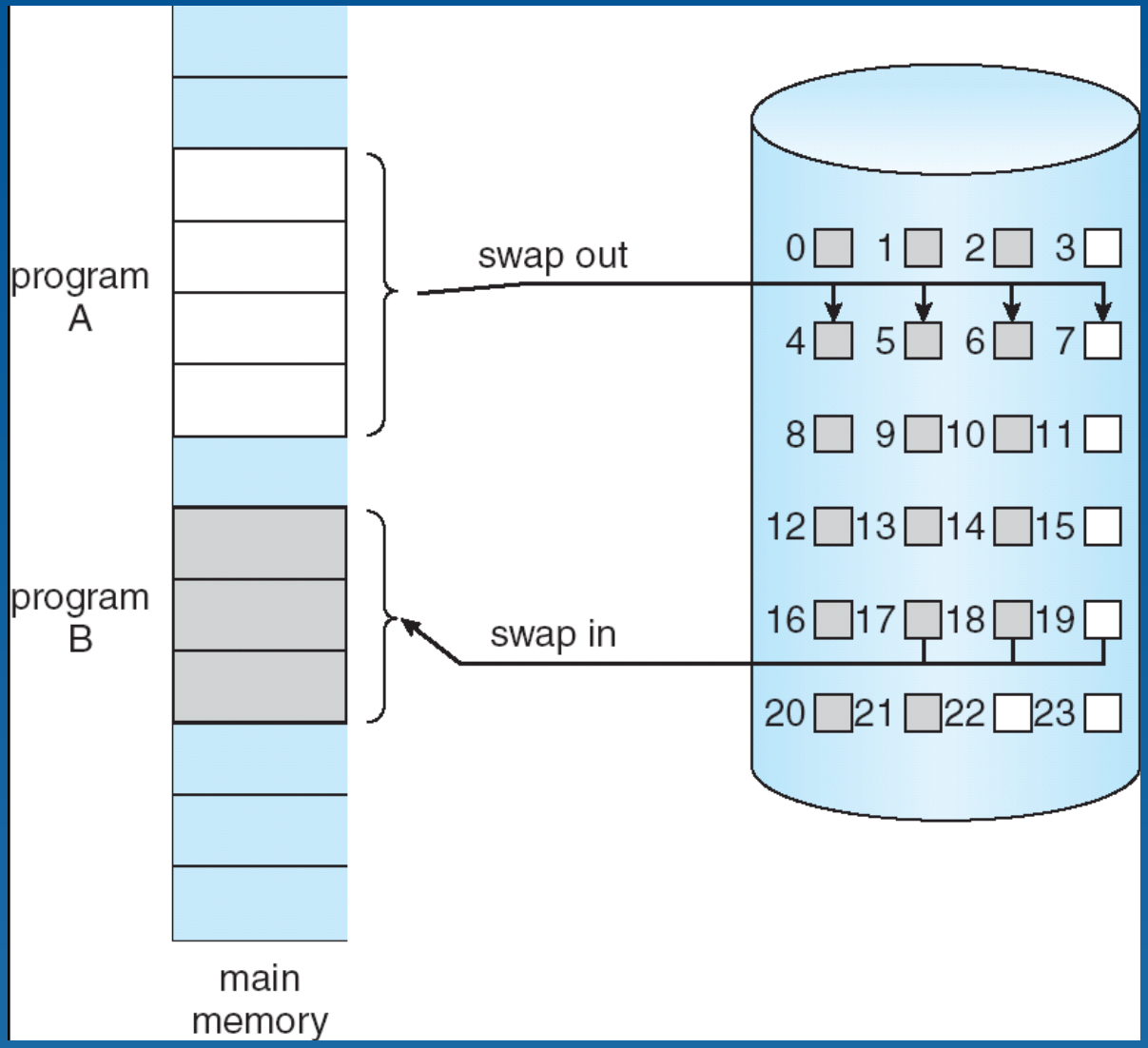
- CPU 指令含内存访问，即访问该内存地址所在页面。称作页面引用 (reference)
- 存在 3 种可能
  - 页面已经装入内存，有对应页帧

    ⇒ CPU 完成操作
  - 非法的页面引用

    ⇒ abort
  - 合法引用，但是页面不在内存

    ⇒ 把页面装入内存

示意图：页面换入、换出


program A
program B
swap out
swap in
0 1 2 3
4 5 6 7
8 9 10 11
12 13 14 15
16 17 18 19
20 21 22 23
main memory

按需调页（续）

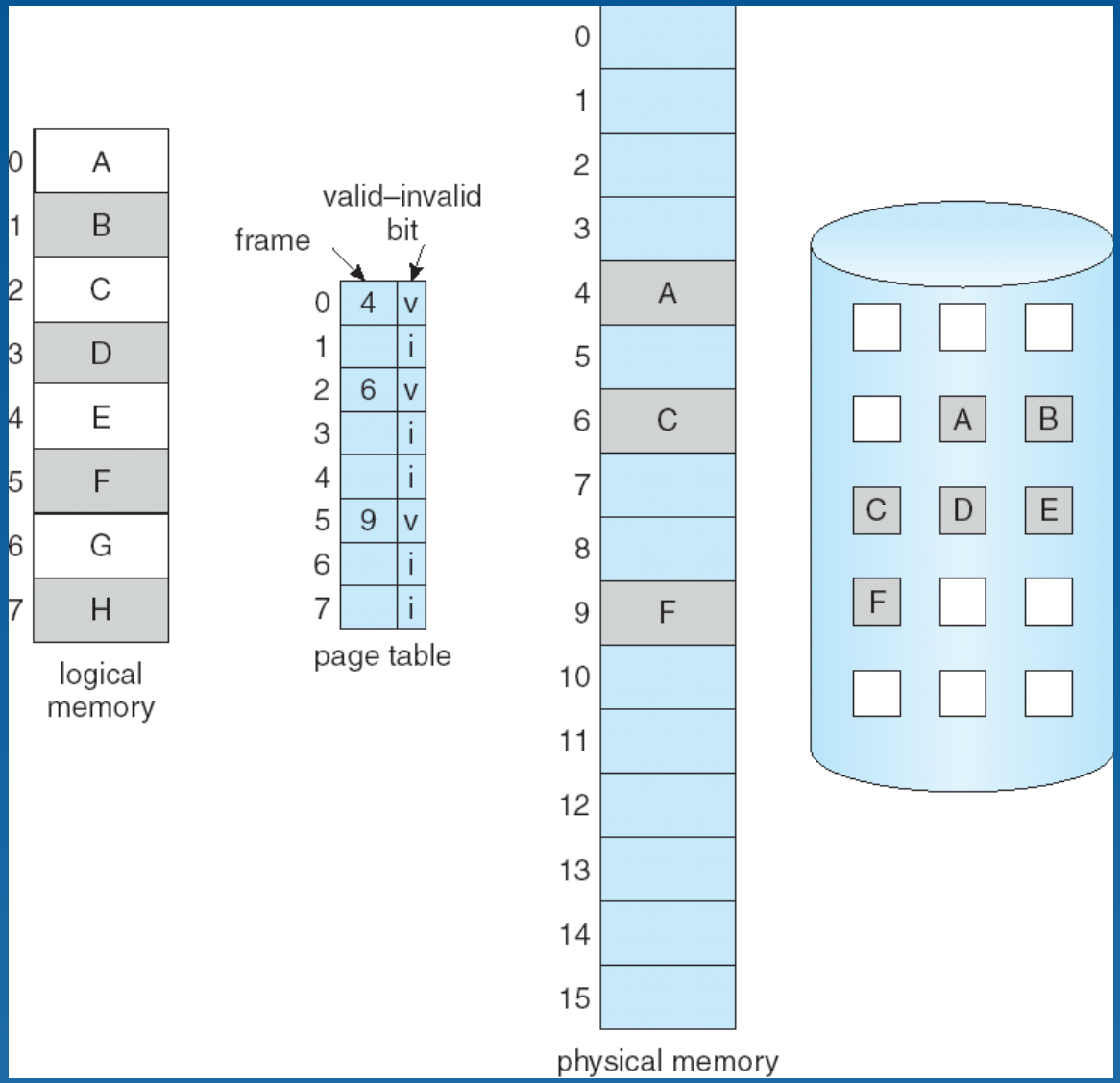
- 当且仅当需要该页面时，才把它调入内存
  - Less， I/O 操作
  - Less， 内存需求
  - Faster， 响应
  - More， 用户数
- **Lazy swapper** –
  - 从不换入这个页面，除非有进程真的访问这个页面
  - Swapper that deals with pages is a **pager**

有效位 (Valid-Invalid Bit)

- 为每个进程的每个页表项，设计一个“有效位”， a valid–invalid bit
  - **v** ⇒ 页面驻留内存
  - **i** ⇒ 进程非法访问该页面；
    或者，页面不在内存
- 初始时，所有“有效位”都设置成 **i**
- CPU 执行指令，访问内存单元时，一旦读到某个页表项，其“有效位”是 **i**

  ⇒ page fault， 缺页

示例：一个页表的快照。某些页面不在内存

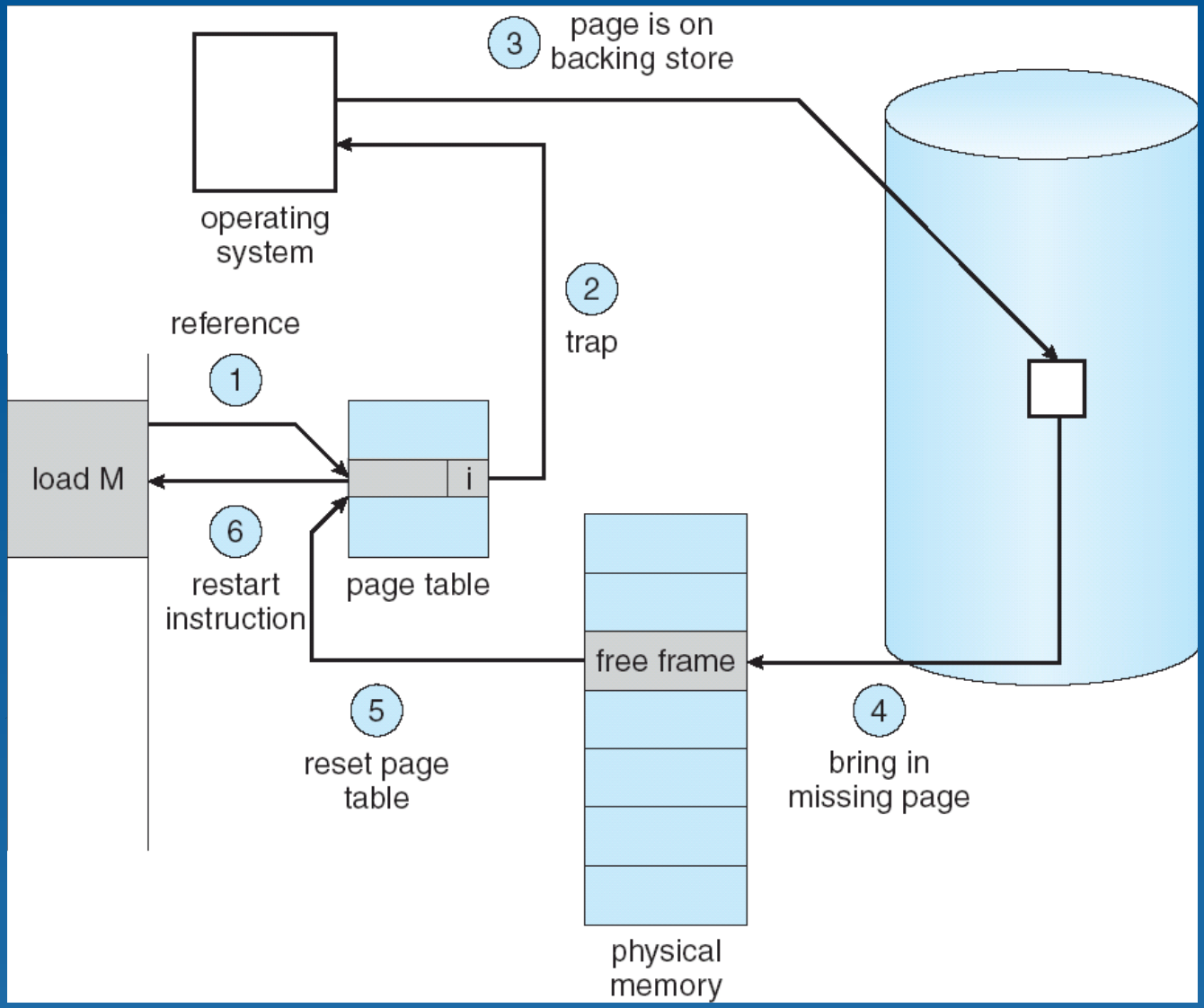


缺页 (page fault)

- 第一次引用某页面，对应页表项的“有效位”为 i ，引起缺页中断。操作系统响应这个中断：

1. 操作系统查找内核的数据结构，判断：
   - 它是非法引用 ⇒ abort
   - 它是合法引用，但是页面不在内存
2. 操作系统查找内核的数据结构，找出一个空闲页帧
3. 把页面从外存换入至该空闲页帧
4. 更新内核的数据结构，更新进程的页表
5. 把进程页表中（该页面）页表项的“有效位”重置为 v
6. 缺页中断程序返回
7. 重新执行引起缺页中断的那条指令

缺页中断，及其响应


page is on backing store
3
operating system
2
trap
reference
1
load M
i
6
restart instruction
page table
free frame
5
reset page table
4
bring in missing page
physical memory


End