
第五章
设备管理

设备管理基本任务及功能

I/O系统：实现数据输入、输出及数据存储的系统

设备管理的基本任务：

①完成用户提出的I/O请求，②提高I/O速度，③改善I/O设备的利用率。

设备管理的主要功能：

①缓冲区管理、②设备分配、③设备处理、④虚拟设备、⑤设备独立性。

设备管理

内 容 提 要

5.1.1 I/O系统—I/O设备

❖ I/O设备的重要性能指标：数据传输速率、数据传输单位、设备共享属性。

* 按速度分
  - 低（10^0—10^2B/S）：键盘、鼠标、语音
  - 中(10^3—10^4B/S）：打印机
  - 高(10^5—10^7B/S）：磁盘、磁带、光盘

* 按信息交换单位分
  - 块（磁盘）：有结构设备，高速，可定位或寻址
  - 字符（打印机）：无结构设备，低速，不可寻址

5.1.1 I/O系统—I/O设备

* 按设备的共享属性分

  - 独占：临界资源
  - 共享：磁盘(必须是可寻址、可随机访问设备)
  - 虚拟：如本身因有属性为独占，但将其虚拟为几个逻辑设备。

5.1.2 I/O系统—设备控制器

- 控制器：控制一个或多个I/O设备，以实现I/O设备与CPU之间的数据交换；是CPU与I/O设备之间的接口，接收CPU命令，控制I/O设备工作，使CPU从繁杂的设备控制事务中解放出来。
- 控制器可连接一个或多个设备，每个设备需要一个具体的地址，以便访问。
- 控制器分类
  - 字符设备控制器
  - 块设备控制器

5.1.2 I/O系统—设备控制器

❖ 功能： 接收CPU命令，控制I/O设备工作，解放CPU

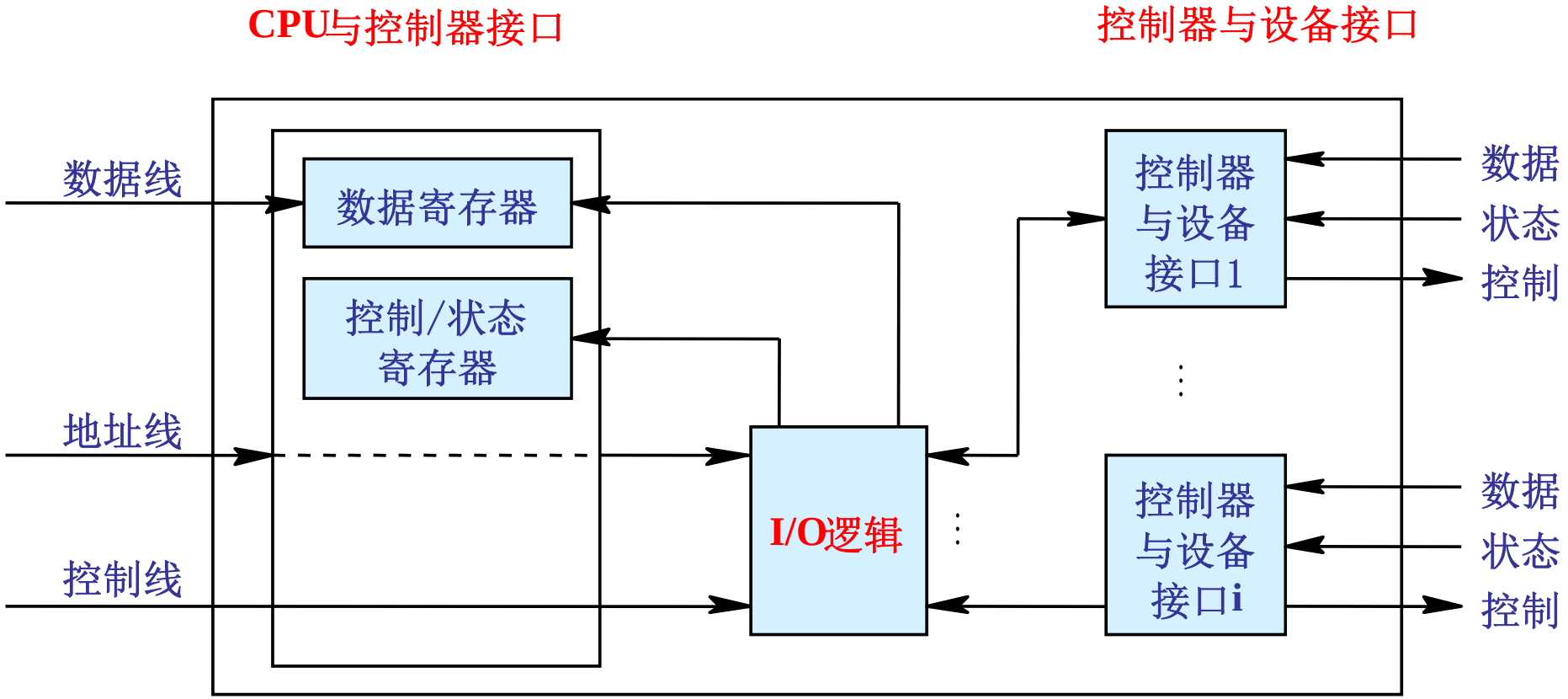
* 1.接收和识别命令
  - 接收CPU的各种指令，并识别
  - 应有相应的Register来存放命令（“命令寄存器”）
* 2.数据交换
  - CPU——控制器的数据寄存器——设备
* 3.了解和报告设备状态
  - 记录设备状态，供CPU了解，以便CPU决定如何操作设备
  - 控制器应具有“状态寄存器”

5.1.2 I/O系统—设备控制器

* 4. 地址识别
  - CPU通过“地址”与设备通信，设备控制器应能识别它所控制的设备地址以及其各寄存器的地址。
* 5. 数据缓冲
  - 采用“数据缓冲器”来解决I/O设备与CPU速率上的矛盾
* 6. 差错控制
  - 对I/O设备的输入数据进行差错检测
  - 出错重传

5.1.2 I/O系统—设备控制器

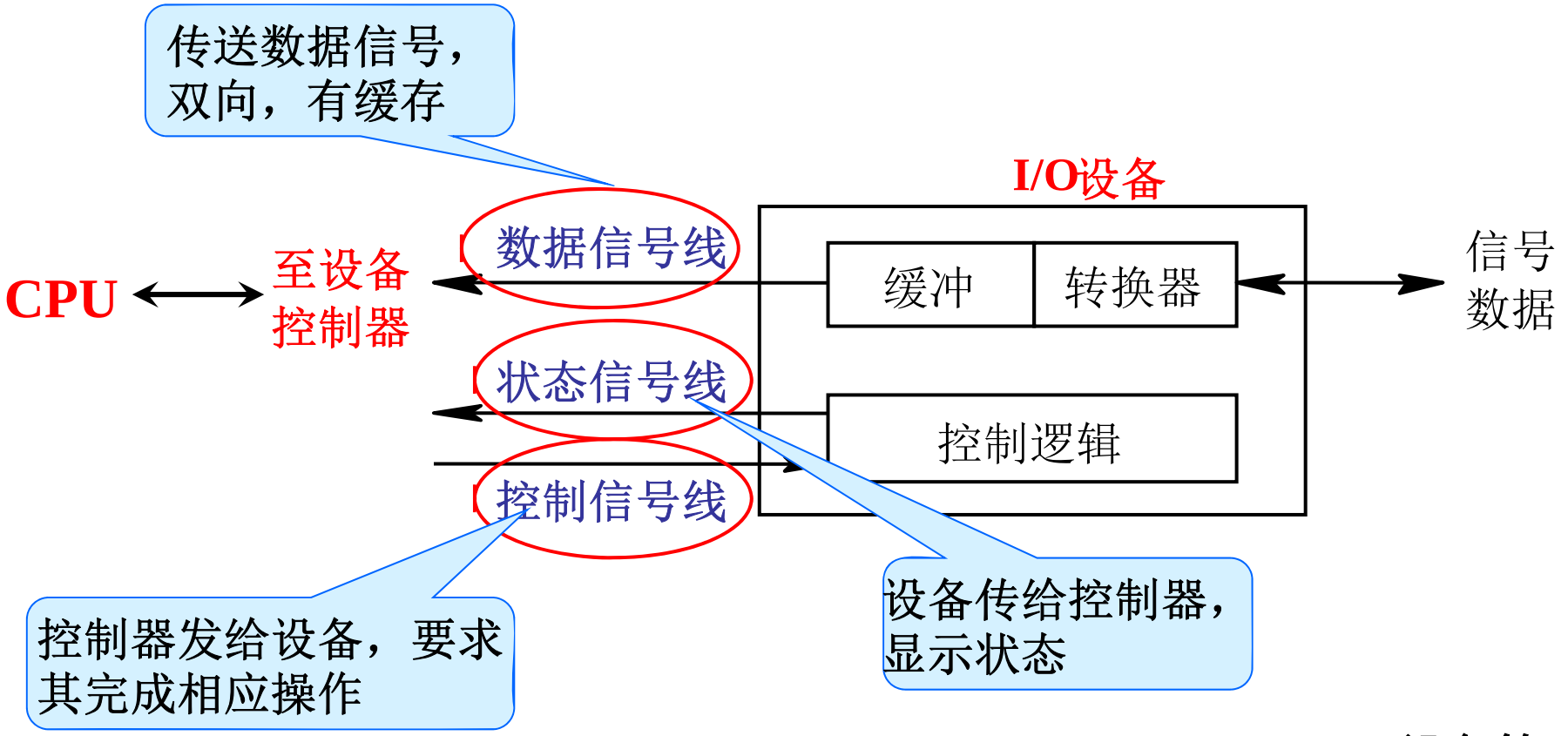
❖设备控制器的三部分组成



5.1.2 I/O系统—设备控制器

- 设备与控制器之间的接口

  通信形式：CPU——设备控制器——设备



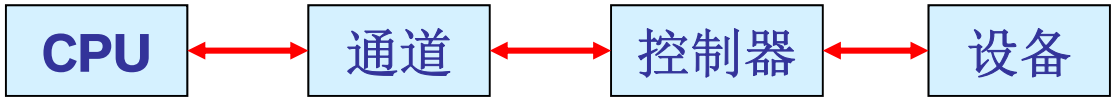
5.1.3 I/O系统—I/O通道

❖I/O通道（I/O Channel）的引入

* **I/O通道：** 一种特殊的能执行I/O指令的处理机，与CPU共享内存，可以有自己的总线。

* **引入目的：** 解脱CPU对I/O的组织、管理、数据传递。

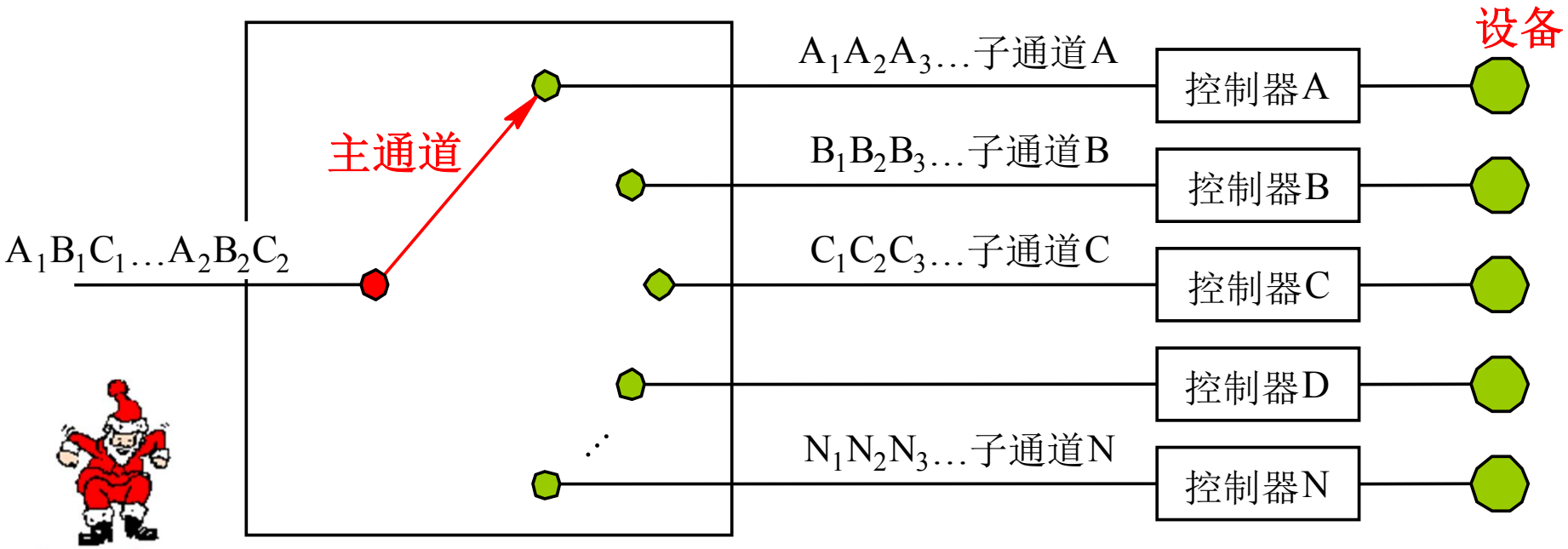
* **处理过程：** CPU只需发送I/O命令给通道，通道通过调用内存中的相应通道程序完成任务。



5.1.3 I/O系统—I/O通道

❖通道类型（外围设备的复杂性导致了不同的通道）

* 字节多路通道：各子通道以时间片轮转方式共享主通道，适用于低、中速设备。

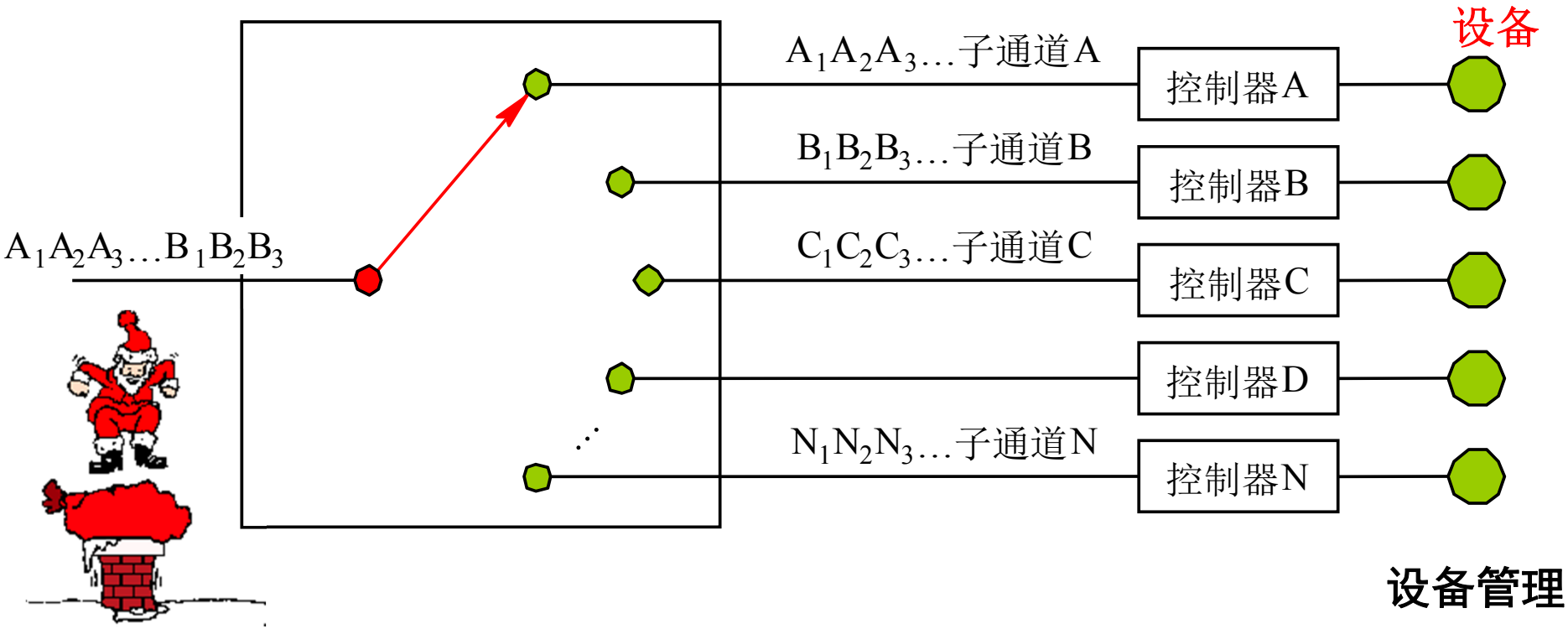


要求连接设备的速率不能太高

5.1.3 I/O系统—I/O通道

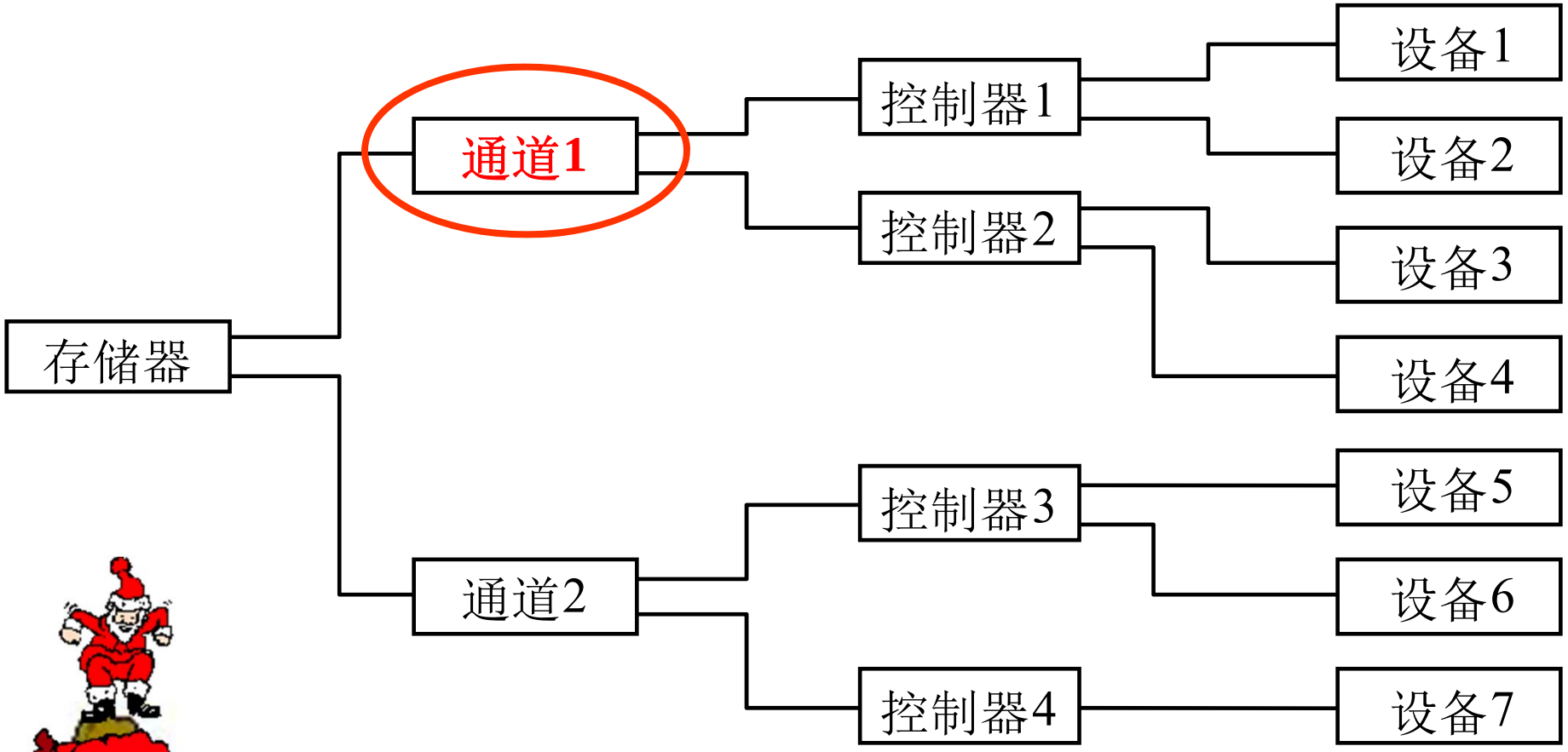
* **数组选择通道**：无子通道，仅一主通道，某时间由某设备独占，适于高速设备。但通道未共享，利用率低。

* **数组多路通道**：综合了前面2种通道类型的优点。多个子通道分时并行操作，也可实现“按需分配”。



5.1.3 I/O系统—I/O通道

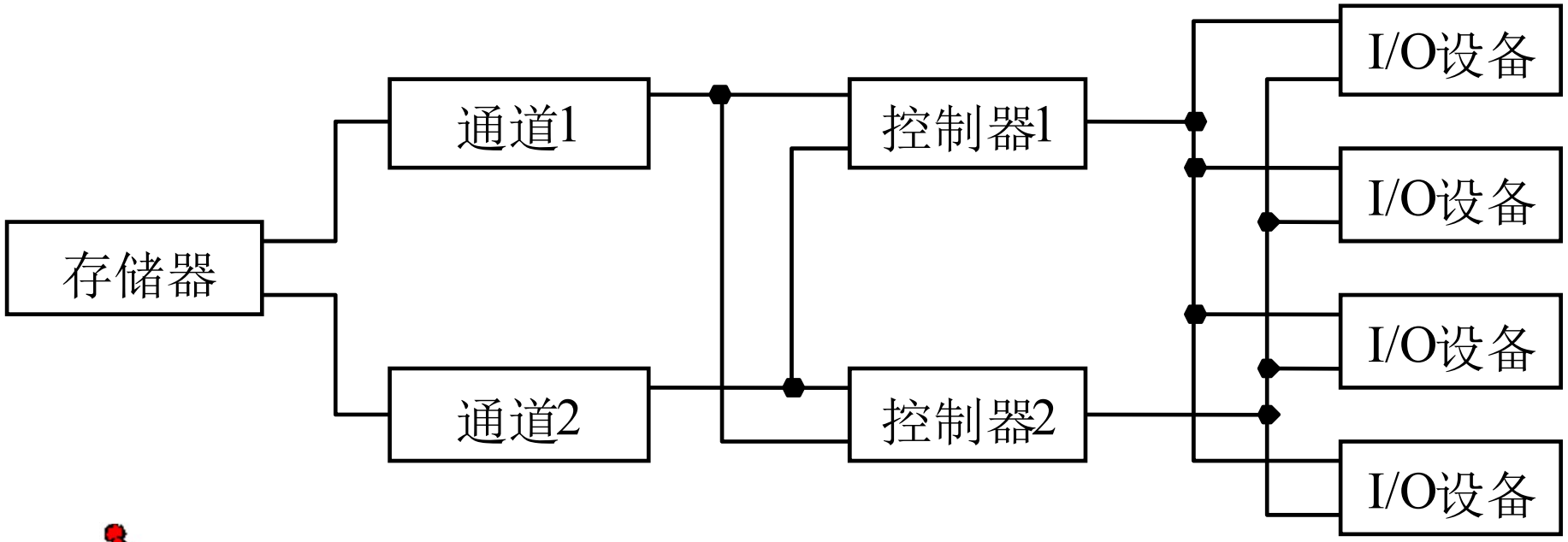
❖通道“瓶颈”问题



单通路I/O系统

5.1.3 I/O系统—I/O通道

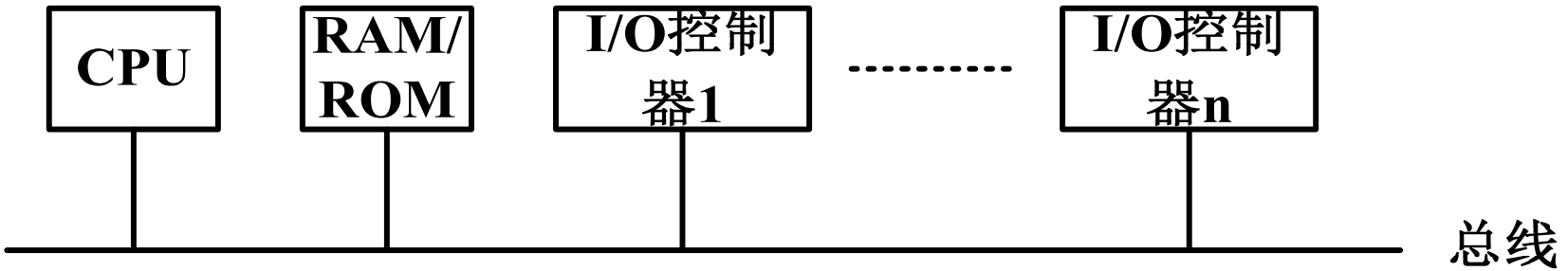
❖解决：采用复联方式(增加设备到主机间的通路)

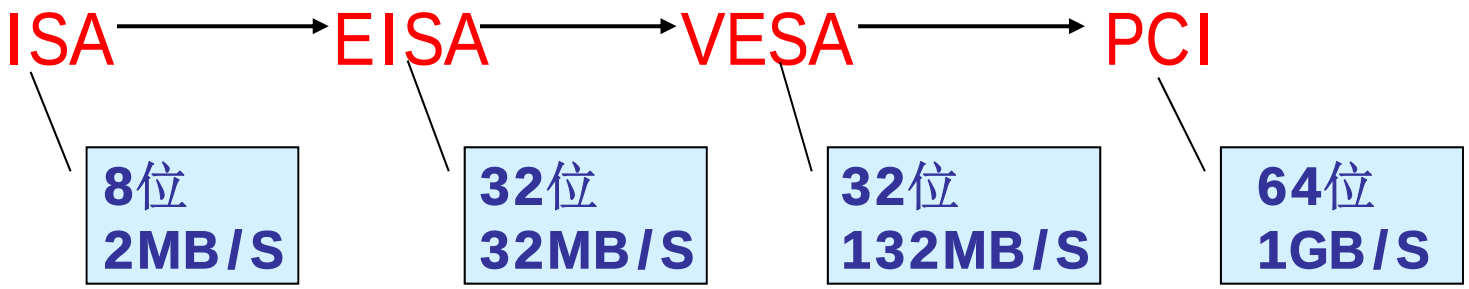


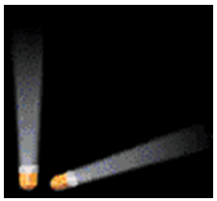
多通路I/O系统

5.1.4 I/O系统—总线系统

计算机系统各部件之间的通信，都是通过总线来实现的。总线性能指标一般有：时钟频率、带宽、传输速率。







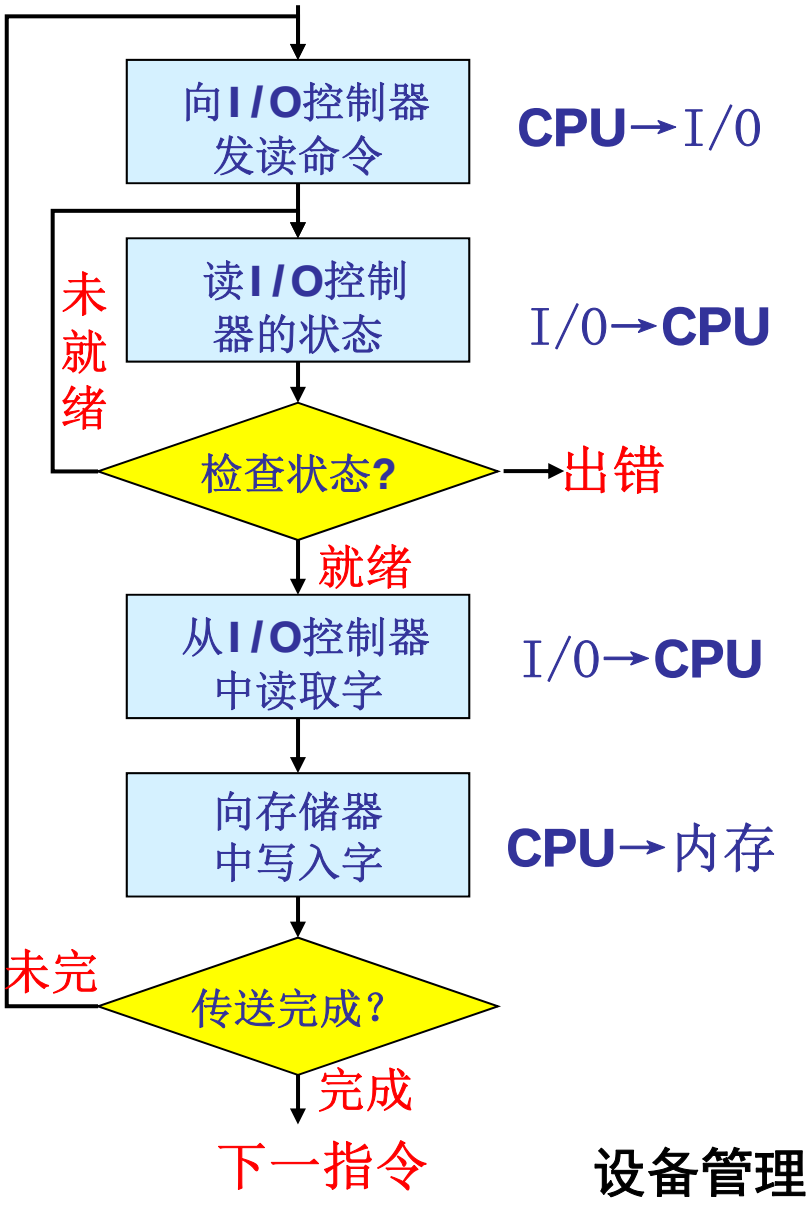
5.2 I/O控制方式

目标：尽量减少CPU对I/O控制的干预

* 速度越来越快
* 须CPU干预越来越少
* 并行度越来越高

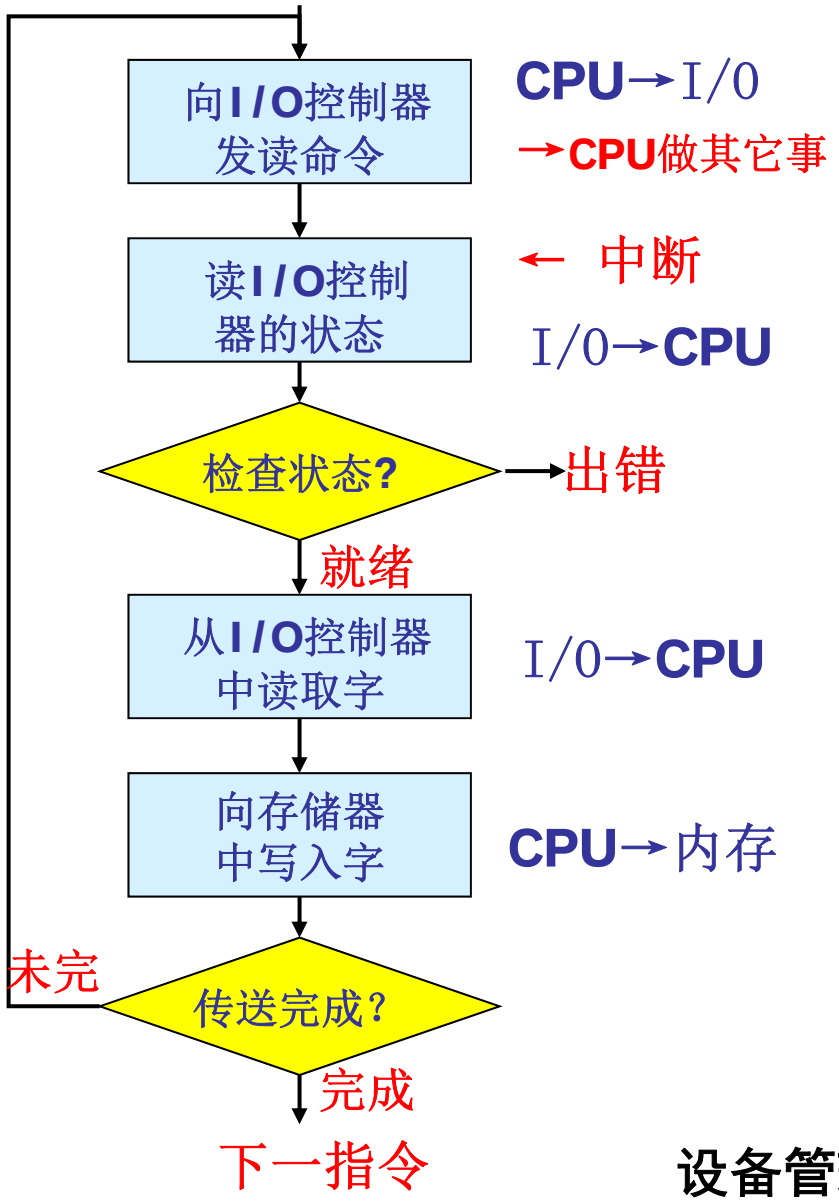
5.2.1 程序I/O（忙—等待方式）

- 传输单位：字符
- 查询方式：CPU需花代价不断查询I/O状态。
- CPU资源浪费极大。



5.2.2 中断I/O

- 传输单位：字符
- CPU向I/O发命令—返回—执行其它任务
- CUP与I/O并行执行
- I/O中断产生—CPU转到相应中断处理程序



程序I/O与中断I/O示例比较

例如，从终端输入一个字符的时间约为**100 ms**，而将字符送入终端缓冲区的时间小于0.1ms。

(1)程序I/O方式：CPU约有99.9ms的时间处于忙—等待状态中。

(2)中断I/O方式：CPU可利用99.9ms的时间去做其它事情，而仅用0.1ms的时间来处理由控制器发来的中断。

5.2.3 DMA方式——用于块设备中

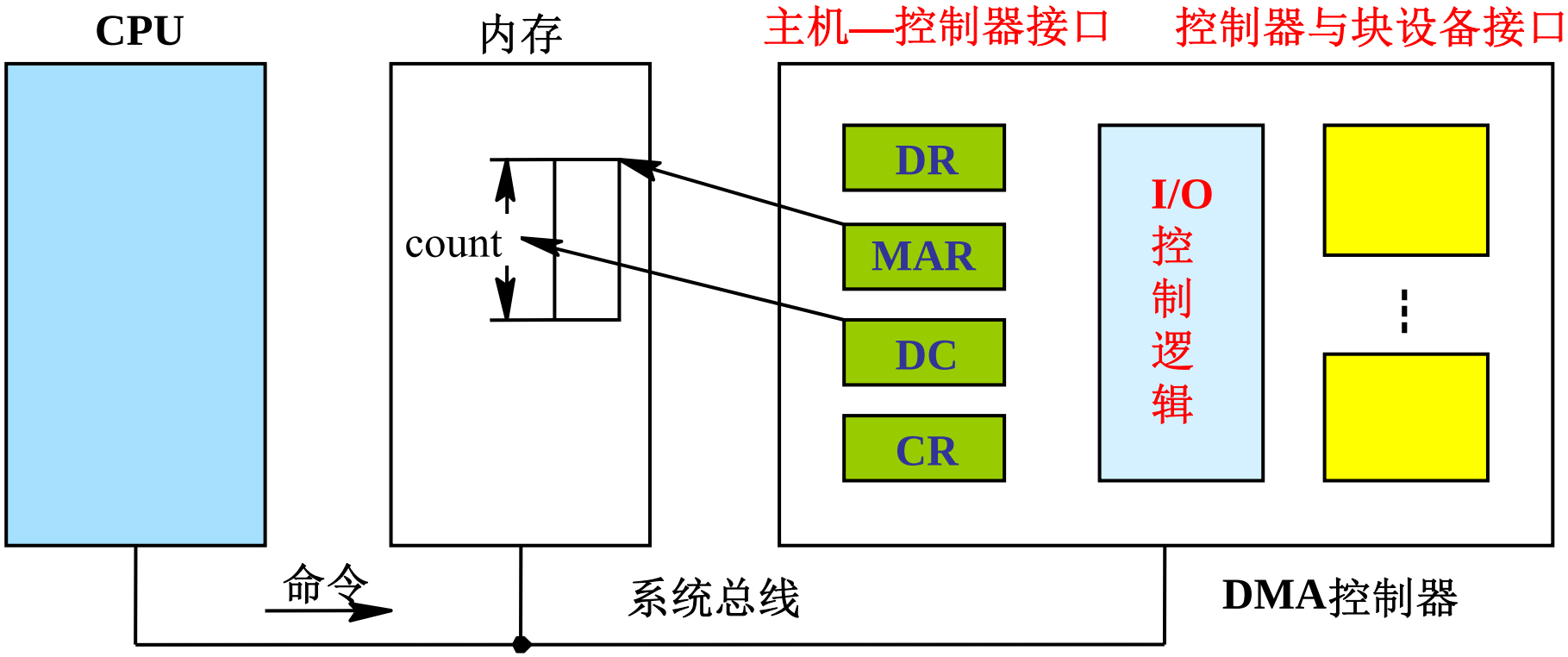
❖ DMA(Direct Memory Access) I/O的引入

* 解决在中断I/O中CPU需每“字节”干预一次的情况。

❖ DMA控制方式的特点

* 传输单位：数据块
* 数据传输方式：设备 ←→ 内存
* CPU干预点：仅在传送一个或多个数据块的开始和结束时，整块数据的传送是在控制器的控制下完成的。

DMA控制器的组成



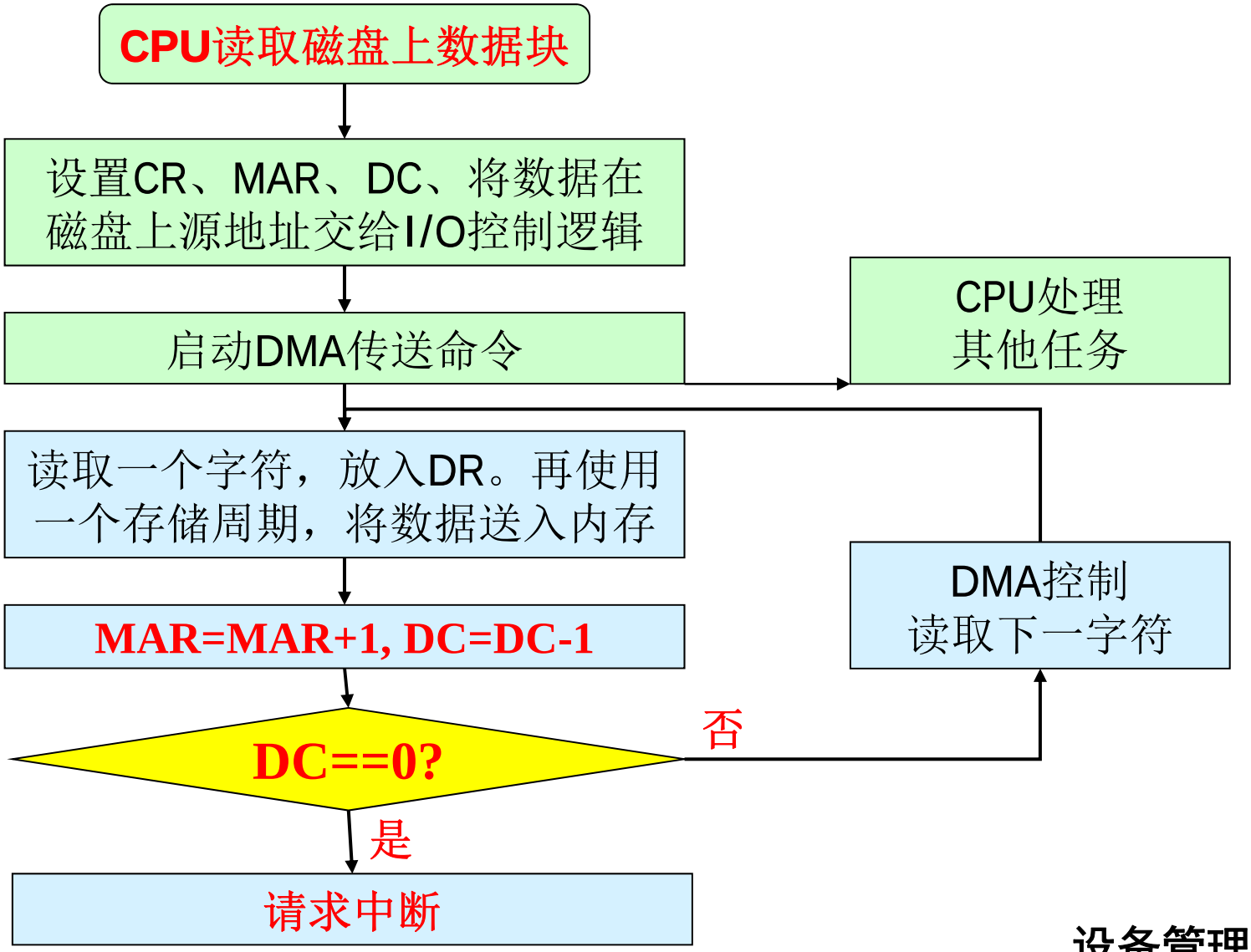
实现主机(CPU)与控制器间成块数据的直接交换，须以下机构：

CR(命令/状态寄存器)　　MAR(内存地址寄存器)

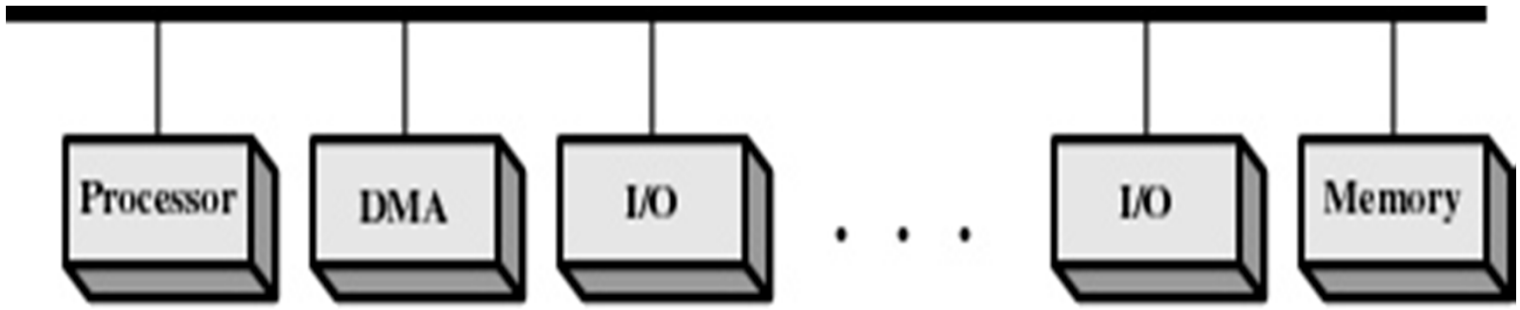
DR(数据寄存器)　　DC(数据计数器)

5.2.3 DMA方式

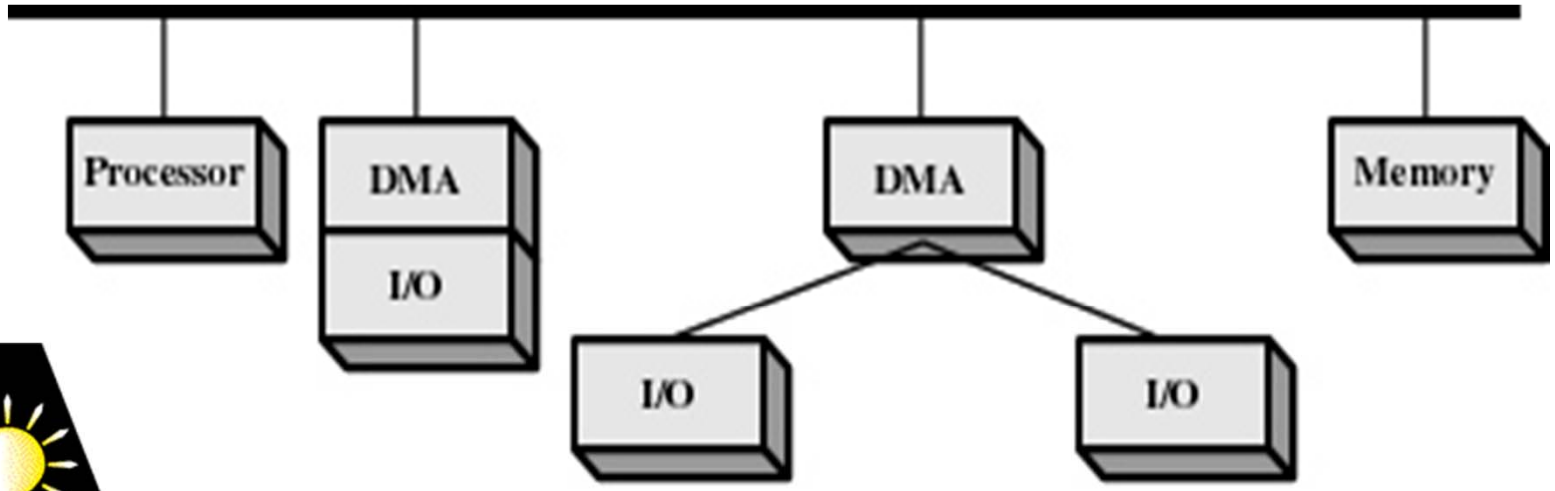
DMA工作过程



DMA的几种配置方式

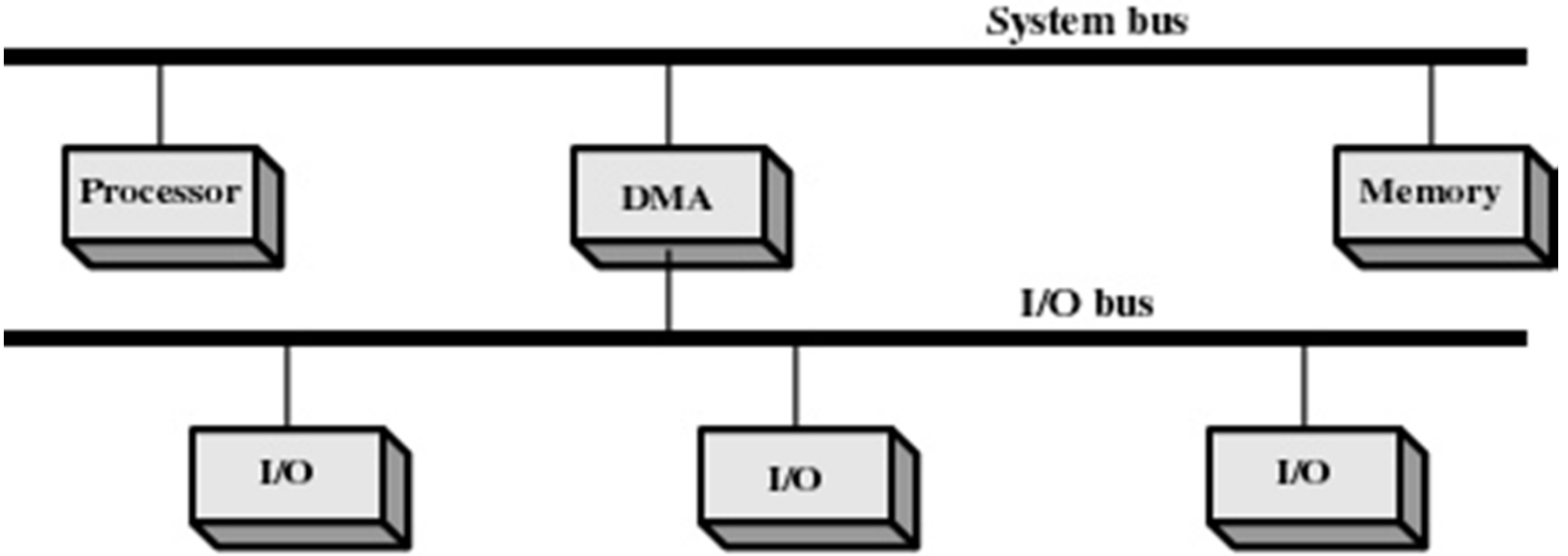


(a) Single-bus, detached DMA



(b) Single-bus, Integrated DMA-I/O

DMA的几种配置方式



(c) I/O bus

5.2.4 I/O通道控制方式

- DMA方式：对多个离散块的读取仍需要CPU多次中断。
- 通道方式：CPU只需给出
  - （1）通道程序首址。
  - （2）要访问I/O设备
- 通道程序可完成CPU指定的一组块操作
- 可实现CPU、通道、I/O设备三者并行运行

通道指令组成

操作	程序结束位P	记录结束位R	计数	内存地址
Write	0	0	80	813
Write	0	0	140	1034
Write	0	1	60	5830
Write	0	1	300	2000
Write	0	0	250	1850
Write	1	1	250	720

5.3 缓冲管理

❖缓冲的引入

* 缓和CPU和I/O设备间速度不匹配的矛盾。

  如：计算—打印buffer—打印

* 减少对CPU的中断频率

  如：buffer越大，“buffer满”信号发生频率越低。

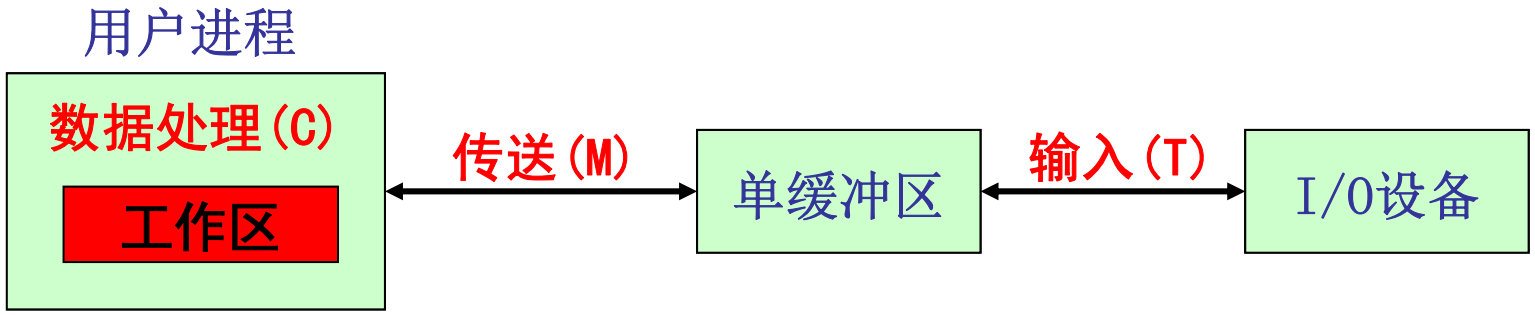
* 提高CPU和I/O并行性

❖缓冲管理目的

* 组织管理
* 分配buffer
* 释放buffer

5.3 缓冲管理—单缓冲区

❖单缓冲(Single Buffer)

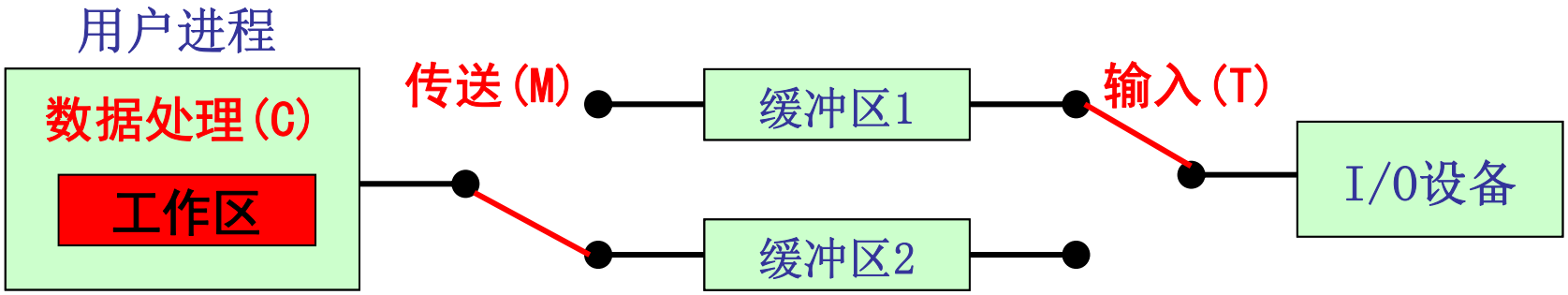


❖知识点

* 块设备：C和T可并行，M和C或M和T不能并行？一块数据的处理时间＝ Max(C,T)+M
* 字符设备：一次暂存一行数据
* 用户进程阻塞情况：输入，输出

5.3 缓冲管理—双缓冲区

❖双缓冲(Double Buffer)

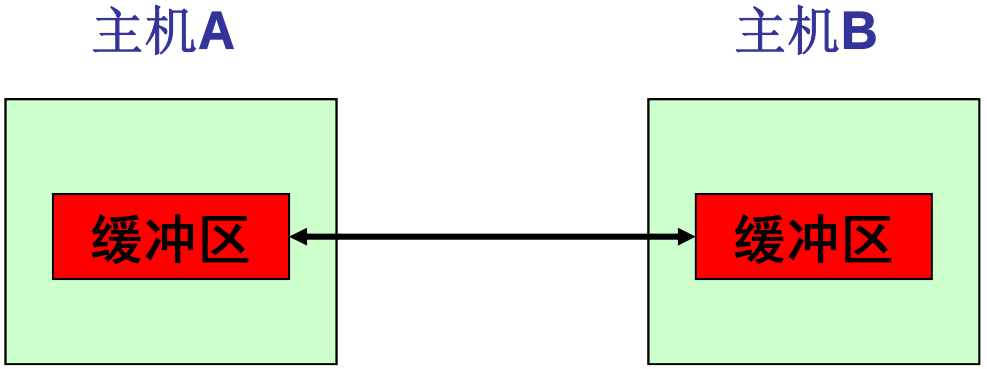


❖知识点

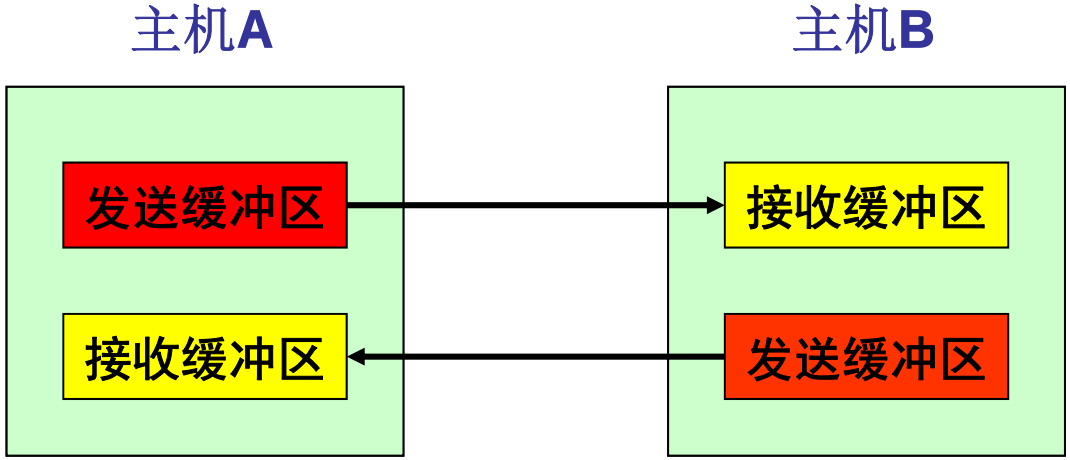
- 提高了设备利用率。
- 系统处理一块数据的时间约为：**MAX(C,T)**。**C**、**T**、**M**可并行，由于**M**非常小，所以进考虑**C**与**T**。
- 可实现输入、计算进程并行执行。
- 实现两台主机间的双向通信。

5.3 缓冲管理—双缓冲区

- 双缓冲支持的双向通信



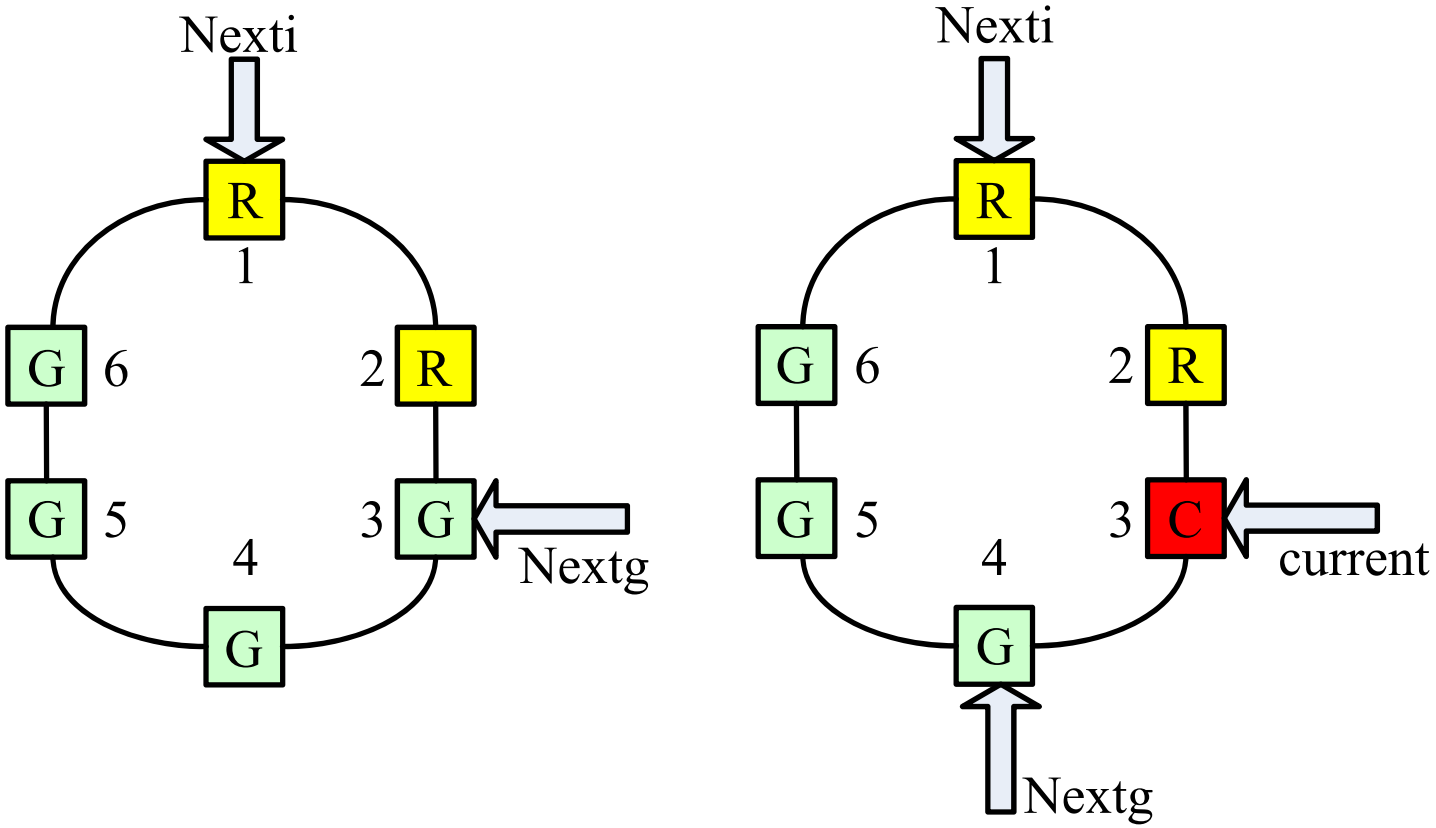
单缓冲区



双缓冲区

5.3 缓冲管理—循环缓冲

- 循环多缓冲（解决双缓存冲速度不匹配问题带来的缺陷）



- 缓冲类型：R：空缓冲；G：满缓冲；C：当前缓冲
- 指针类型：与上对应

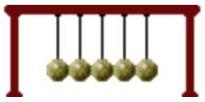
5.3 缓冲管理—循环缓冲

❖循环缓存区的使用（两过程）

* Getbuf
  - 取nextg对应缓冲区提供计算进程使用，将Nextg置为工作缓冲区，Nextg=（Nextg+1）Mod N
  - 将Nexti对应缓冲区提供输入进程使用，将Nexti置为工作缓冲区，Nexti=（Nexti+1）Mod N
* Releasebuf
  - 若C满，则改为G；
  - 若C空，则改为R；

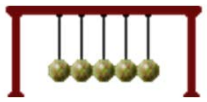
❖进程同步（输入和计算进程并行执行，指针顺时针移动）

* 系统受计算限制：Nextg追上Nexti，输入进程应阻塞
* 系统受I/O限制：Nexti追上Nextg，计算进程应阻塞



5.3 缓冲管理—缓冲池

- **缓冲池：** 系统提供多个进程共同使用的公用缓冲
- 组成与分类
  - 三个队列
    - Emq：空缓冲队列
    - Inq：输入队列
    - Out：q输出队列
  - 四个工作缓冲区
    - hin：收容输入数据
    - sin：提取输入数据
    - hout：收容输出数据
    - sout：提取输出数据



5.3 缓冲管理—缓冲池

由于缓冲池中的缓冲区是多进程公共的，多进程可独立的访问；所以，缓冲区队列成为临界资源，应保证同步。

```
Getbuf(type)
Begin
  wait(RS(type));
  wait(MS(type));
  B(number):=takebuf(type);
  signal(MS(type));
end
```

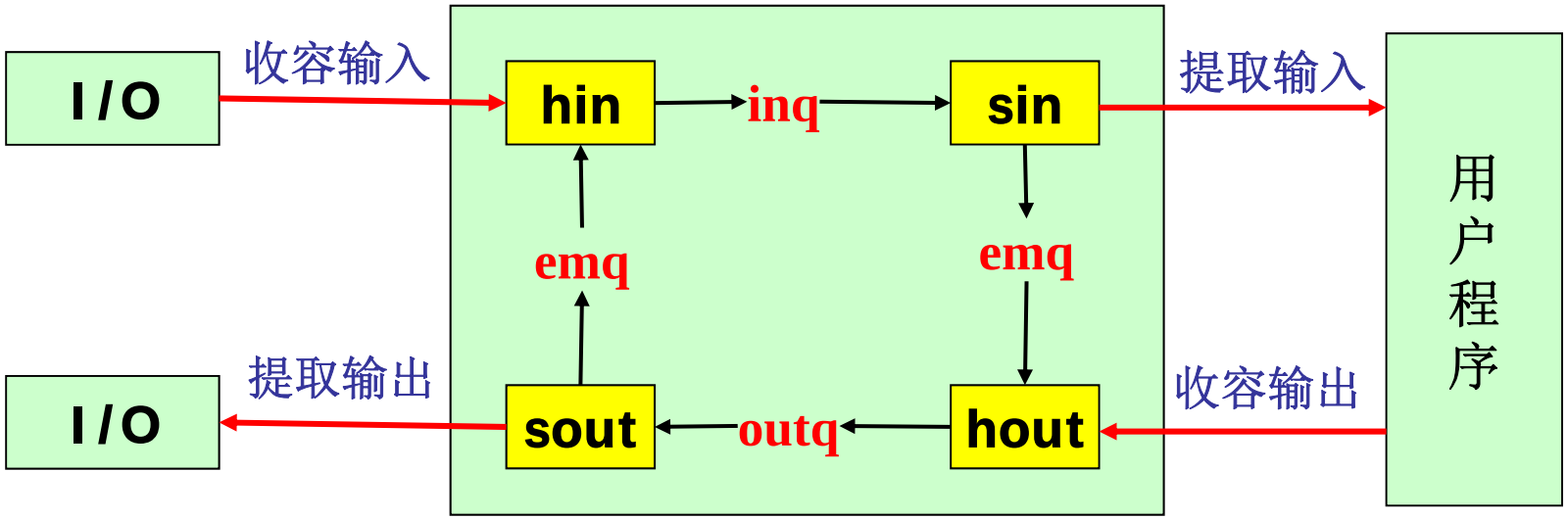
```
Putbuf(type,number)
Begin
  wait(MS(type));
  addbuf(type,number);
  signal(MS(type));
  signal(RS(type));
end
```

参数说明：type缓存区类型、number指示一个缓冲区、RS(type)资源信号量、 MS(type)互斥信号量

5.3 缓冲管理—缓冲池

❖缓冲区的四种工作方式

* 收容输入：输入进程进行数据输入
* 提取输入：计算进程提取输入数据
* 收容输出：计算进程进行数据输出
* 提取输出：输出进程输出数据



5.4 设备分配

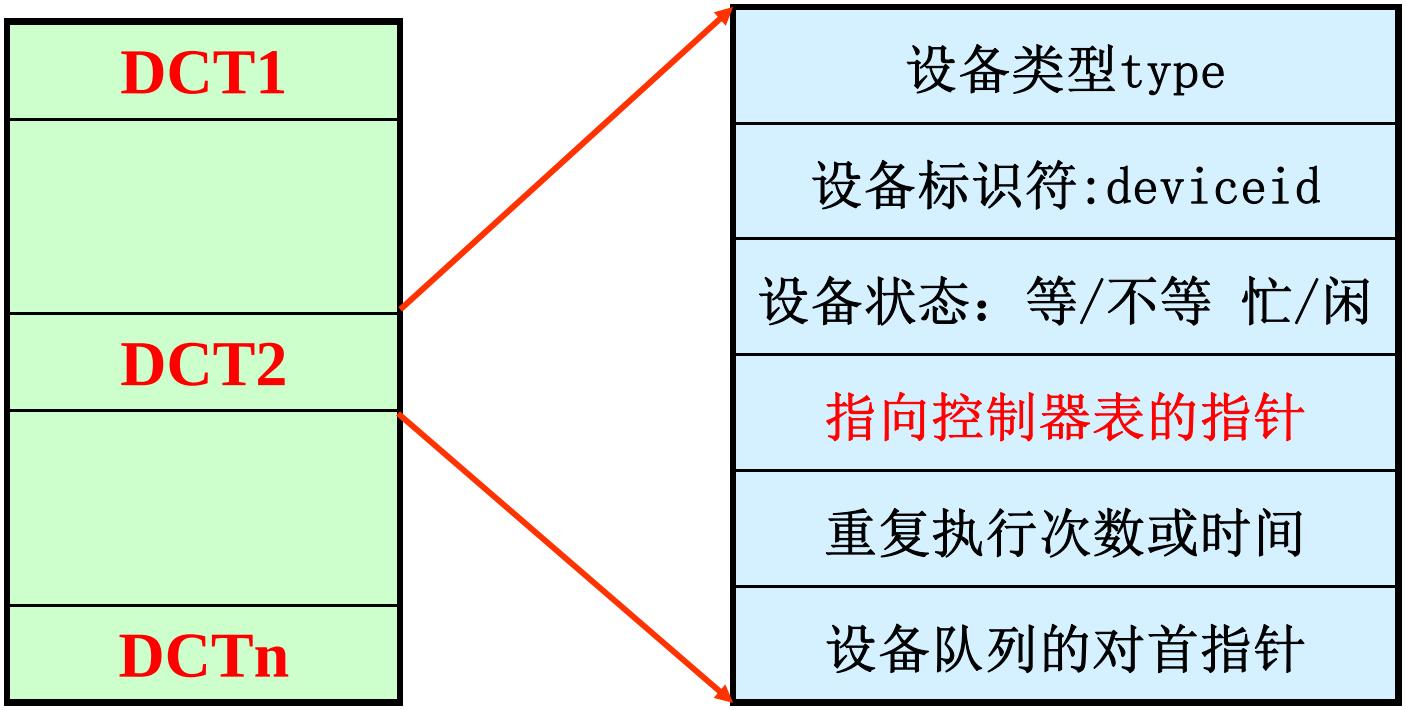
多道程序环境下，系统中的设备供所有进程共享。为防止进程无序的访问资源，要求资源必须在系统下统一分配。包括①设备分配、②控制器分配、③通道分配

❖基本数据结构

* 设备控制表DCT：每个设备一张，记录该设备情况
* 控制器控制表COCT：每控制器一张，记录该控制器情况
* 通道表CHCT：每通道一张，记录通道信息
* 系统设备表SDT ：记录系统中全部设备及其驱动程序地址

5.4.1 设备分配中的数据结构

❖ 设备控制表DCT



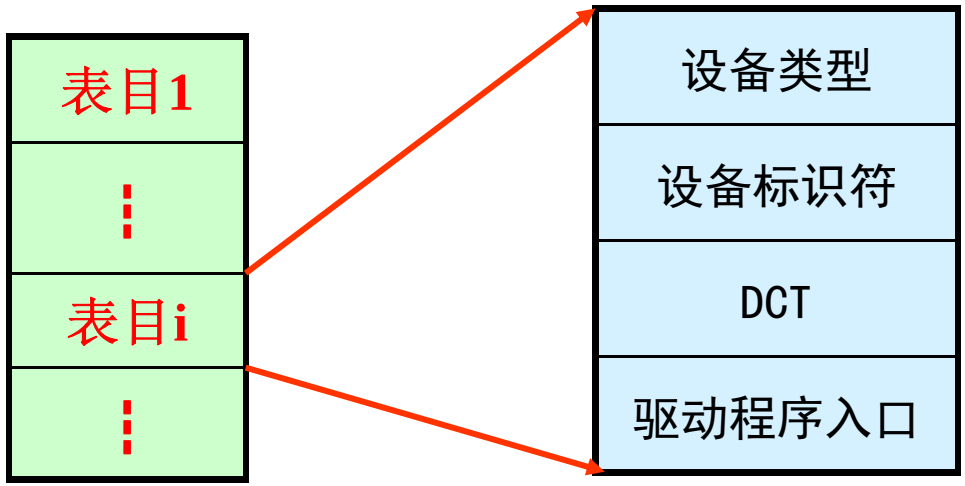
5.4.1 设备分配中的数据结构

控制器标识符：controllerid
控制器状态：忙/闲
与控制器连接的通道表指针
控制器队列队首指针
控制器队列队尾指针

控制器表**COCT**

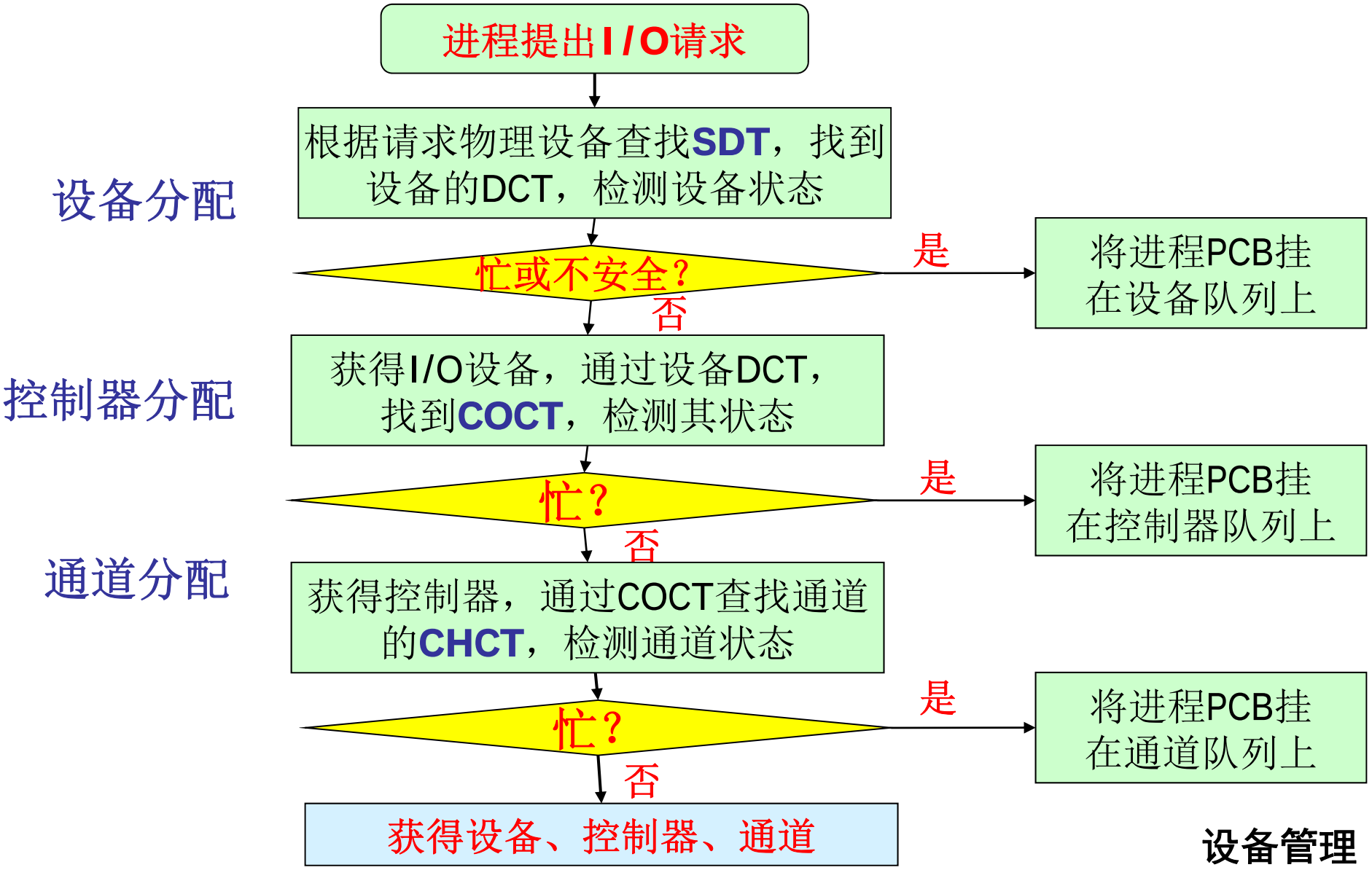
通道标识符：controllerid
通道状态：忙/闲
与通道连接的控制器表指针
通道队列队首指针
通道队列队尾指针

通道表**CHCT**



系统设备**SDT**

四种数据结构之间的关系



5.4.2 设备分配应考虑的因素

- 设备的固有属性
  - 共享+虚拟：注意调度的合理性
  - 独享：排它性分配，控制不好可能死锁
- 分配算法
  - FIFO
  - 优先权
- 设备分配的安全性
  - 安全分配方式：摒弃“请求和保留”条件
  - 不安全分配方式：需进行安全性检查再分配

5.4.3 设备独立性

- 基本概念：又称设备无关性，指应用程序独立于具体使用的物理设备。
- 引入的目的：提高OS的可适应性和可扩展性。
- 实现方法
  - 逻辑设备：在应用程序中使用
  - 物理设备：在实际操作设备时使用
  - 逻辑设备表(LUT)：实现 逻辑地址←→物理地址

逻辑设备	物理设备	Driver入口

LUT表项示例

5.4.3 设备独立性

❖ 名字映射(逻辑设备名→物理设备名)

* LUT的生成：在用户进程第一次请求设备时完成映射并在LUT中生成相应项
* LUT的配置
  - 整个系统一张LUT表：逻辑名不重复,一般用于单用户系统
  - 每个用户一张LUT表:可重名/可限制用户对某些设备的使用。

逻辑设备名	物理设备名	驱动程序入口地址
/dev/tty	3	1024
/dev/printer	5	2046
---	---	---

逻辑设备名	物理设备名
/dev/tty	3
/dev/printer	5
---	---

5.4.3 设备独立性

❖设备独立性的优点

* 设备分配更灵活：逻辑设备和物理设备间可以是多—多的映射关系。提高了物理设备的共享性，以及使用的灵活性。例如：
  - 某逻辑名可对应这一类设备，提高均衡性与容错性。
  - 几个逻辑名可对应某一个设备，提高共享性。
* 易于实现I/O重定向：不变程序，只需改变LUT表的映射关系。

5.4.3 设备独立性

❖设备独立性软件

* 配置在设备驱动程序之上
* 执行所有设备的公有操作
  - 分配回收
  - 名字映射
  - 保护
  - 缓冲
  - 差错控制
* 向用户层软件提供统一接口
  - Read
  - write

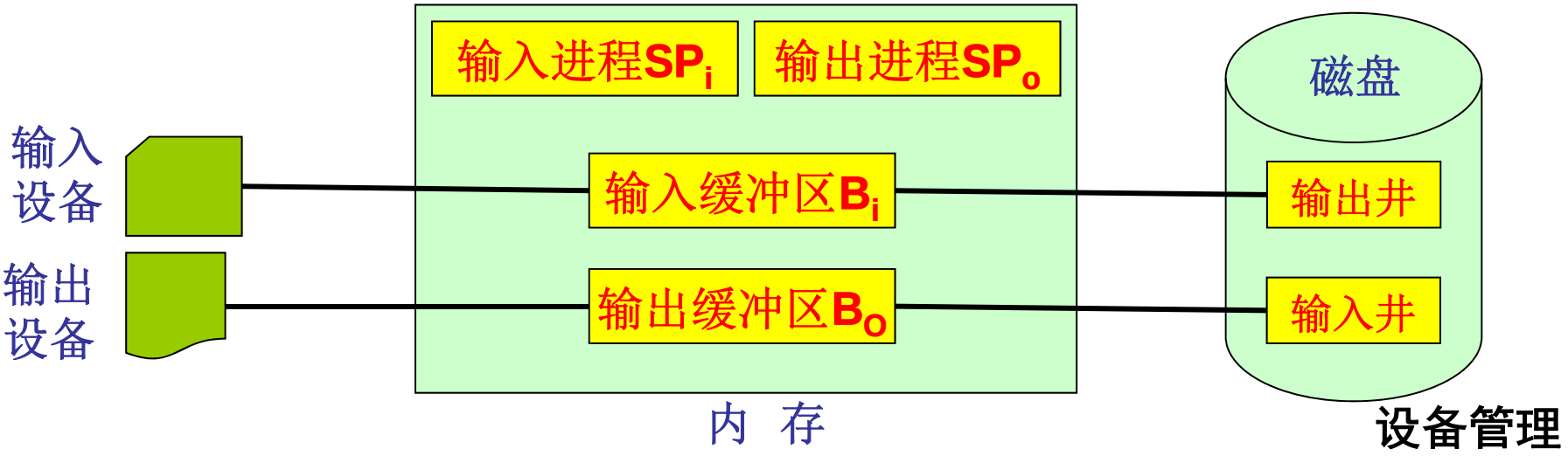
5.4.5 SPOOLING技术

❖基本知识

* 概念：假脱机技术，通过对脱机输入、输出系统的模拟（利用进程），在联机情况下同时实现外围操作。

* 作用：通过缓冲方式，将独占设备改造为共享设备。

❖SPOOLING系统的组成



5.4.5 SPOOLING技术—工作过程

❖(1)输入过程

* 进程n请求→SPi为n在输入井中分配空间→设备数据由输入buf送输入井→生成输入请求表挂输入请求队列。
* CPU空→取请求表中的任务,送进程缓冲区。

❖（2）输出过程：（打印）

* 进程n请求→SPo为n在输出井中分配空间→将数据由进程buf转到输出井→生成一打印请求表挂打印请求队列。
* 打印机空→查打印请求表中的任务→取输出＃中对于数据→输出buf→打印

5.4.5 SPOOLING技术—特点

- 1. 提高I/O速度
  - 对低速设备操作→变为对输入/出井操作。
- 2. 将独占设备改造为共享设备
  - 分配设备的实质是分配输入/出井，并建立I/O请求表
- 3. 实现了虚拟设备功能

5.5 设备处理

设备处理程序即是设备驱动程序，其主要任务是实现I/O进程于设备控制器之间的通信。

❖设备驱动程序的功能

- 接收进程的I/O命令和参数
- 检查命令合法性
- 检查设备状态
- 设置设备工作方式
- 驱动I/O操作
- 响应设备中断
- 构成通道程序

❖设备驱动程序的特点

- 低级系统例程，用于在请求I/O进程和设备控制器之间进行通信和转换。
- 和硬件紧密相关，由于 I/O 设备的复杂性，各个(类)设备有自己的设备驱动程序。
- 程序的一部分应用汇编语言编写

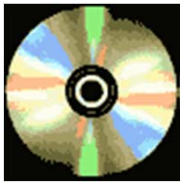
5.5 设备处理—驱动程序处理过程

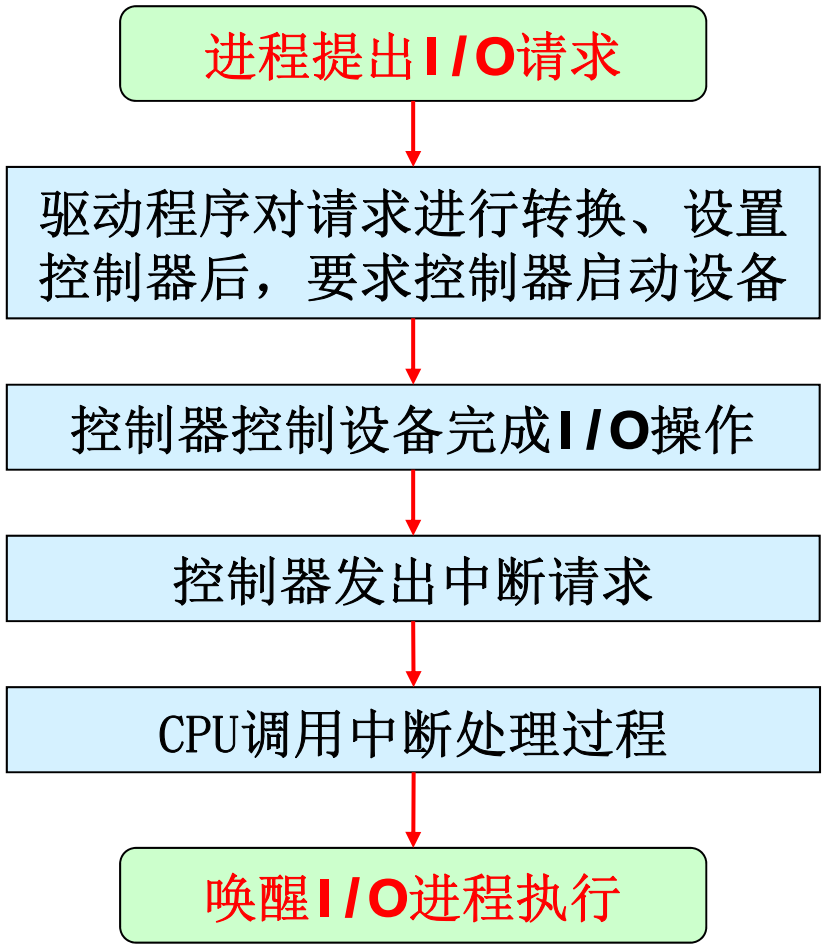
❖ 处理两过程

- 启动过程
- 中断处理过程

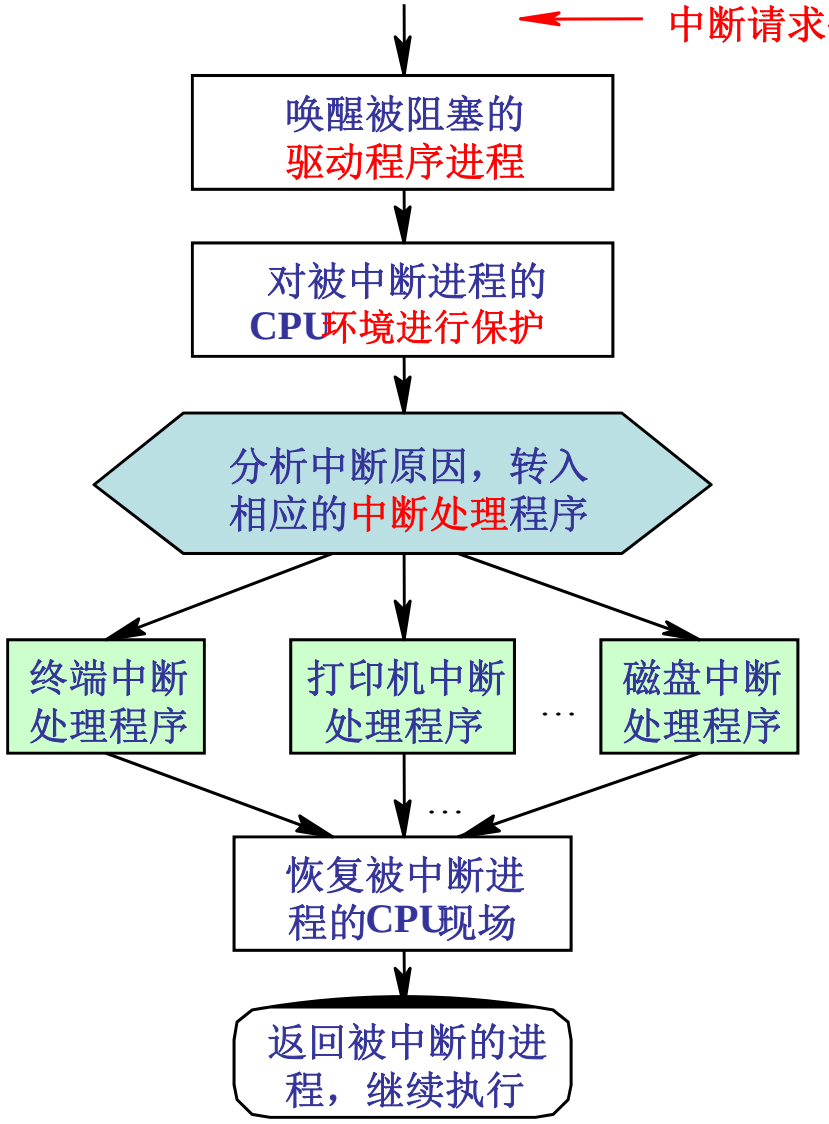
❖ 启动过程(启动控制器)

- 将抽象要求→具体要求
- 检查I/O请求合法性
- 读出和检查设备状态
- 传送必要的参数
- 设置工作方式
- 启动I/O设备





5.5 设备处理—中断处理过程



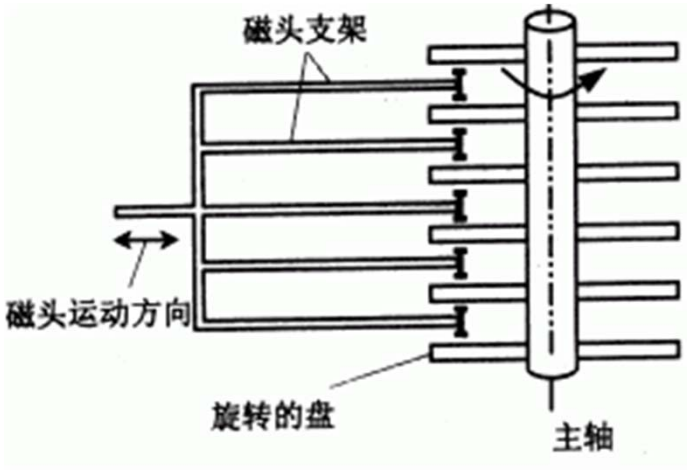
5.6 磁盘存储器管理

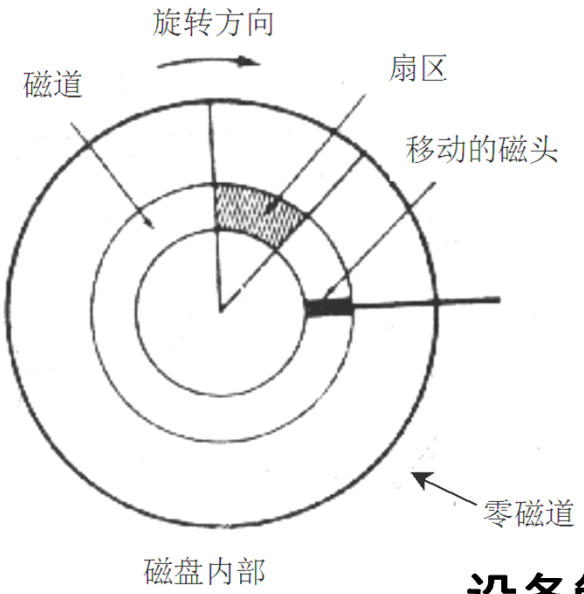
磁盘存储器由于容量大、速度快、可随机存取等特点，成为现代计算机数据存取的主要设备。磁盘的性能直接影响到系统的性能，所以，改善磁盘系统性能成为现代操作系统的重要任务之一。

5.6.1 磁盘性能简述

❖磁盘基本结构

* 盘片→磁道→扇区→字节
* 磁盘密度：位数/英寸
* 每个磁道存储相同的字节数
  内磁道密度>外磁道密度

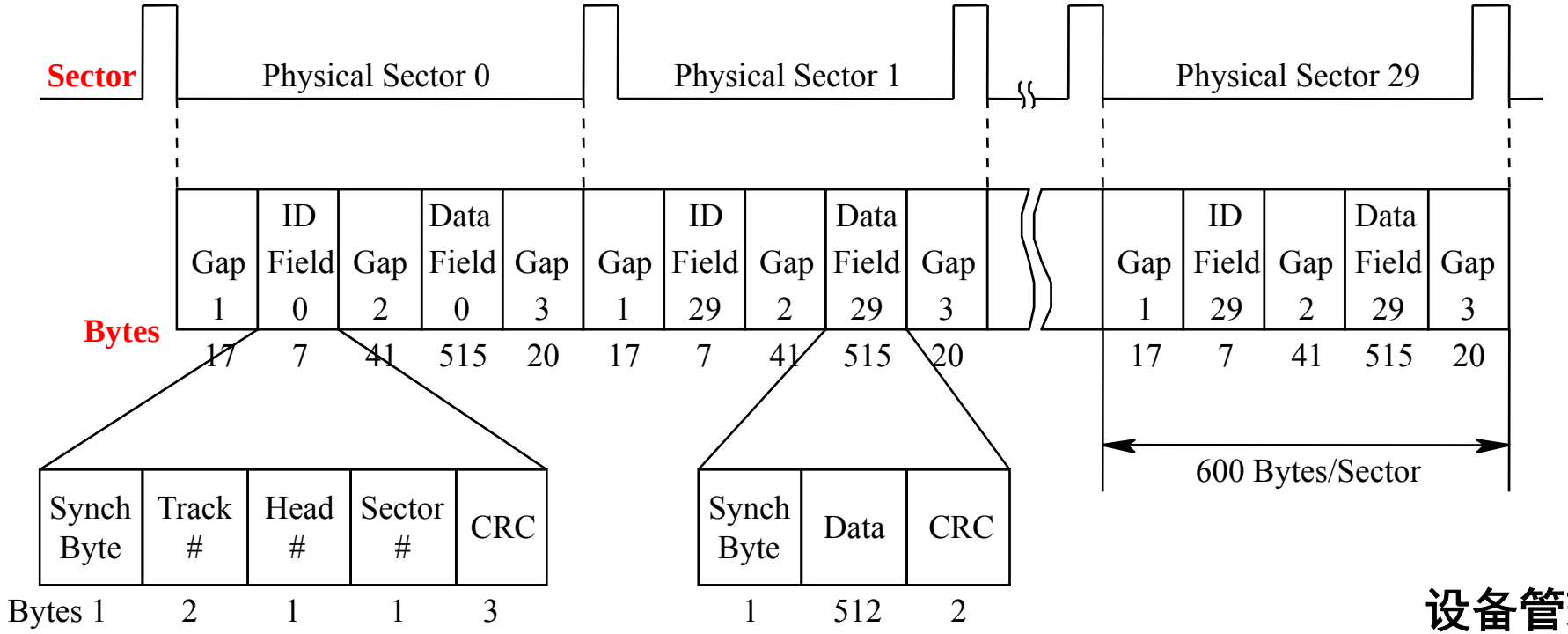




5.6.1 磁盘性能简述

❖磁盘数据格式

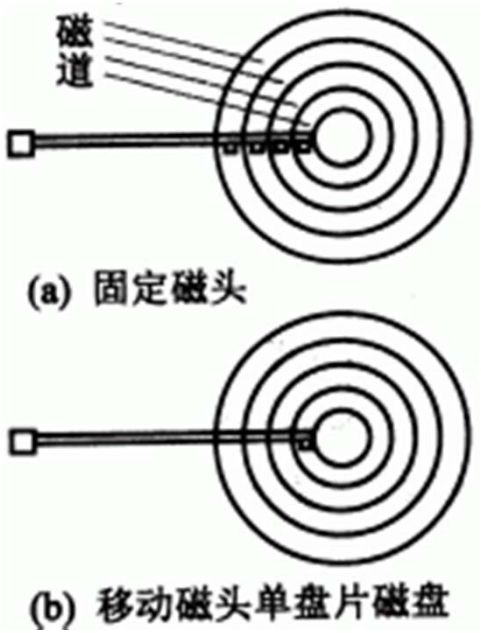
* 欲存储数据必须格式化
* 温(温切斯特)盘结构：30扇区/磁道、600字节/扇区、扇区＝512数据段＋88标识符段



5.6.1 磁盘性能简述

❖磁盘类型

* 硬盘、软盘；单片盘、多片盘；固定磁头盘、移动磁头盘
* 固定头磁盘
  - 每条磁道上有一个磁头，可并行读/写，速度快
* 移动头磁盘
  - 每个盘面仅有一个磁头，读/写必须移动磁头，读/写采用串行方式，速度慢



5.6.1 磁盘性能简述

磁盘工作时以恒定速率旋转。为了读写，磁头必须先移动到所要求的磁道上，并等待所要求的扇区的开始位置旋转到磁头下方，然后才开始读写操作。磁盘访问时间一般分以下三部分：

❖寻道时间：$T_S=m\times n+S$

把磁头移动到指定磁道上所经历的时间。

m：常量，n：磁道数，s：磁盘启动时间，一般约2ms。

❖旋转延时间：$T_r=1/2r$（均值）

指定扇区旋转到磁头下所需时间。r为每秒转数。

5.6.1 磁盘性能简述

❖数据传输时间：$T_t = b/(rN)$

把数据从磁盘读出或写入数据所经历的时间。

b：读写字节数　　N：每磁道上的字节数

❖访问时间：$T_a = T_s + T_r + T_t$

由上式可知，寻道时间和旋转延迟时间基本上与读/写数据的多少无关，但占了访问时间的大部分。只有集中放置数据，集中读写（b大）才能更好提高传输效率。

5.6.2 磁盘调度

- **调度思想：** 使访问磁盘的各进程的平均访问时间最小。
- **调度目标：** 减少寻道时间（移动磁头的距离）。
- **FCFS（Fisrt Come First Second）**
  - 根据进程请求访问磁盘的先后顺序调度
  - 特点：简单，寻道时间长，相当于随机访问模式。
- SSTF（最短寻道优先）
  - 要求所选进程的访问磁道，与当前磁头所在磁道的距离最近，使每次寻道时间最短。
  - 存在进程“饥饿现象”：某进程的访问一直得不到满足

FCFS调度算法

100道开始	
被访问的下一个磁道	移动距离
55	45
58	3
39	19
18	21
90	72
160	70
150	10
38	112
184	146
平均寻道长度：55.3	

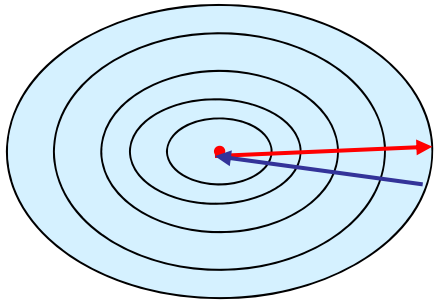
SSTF调度算法

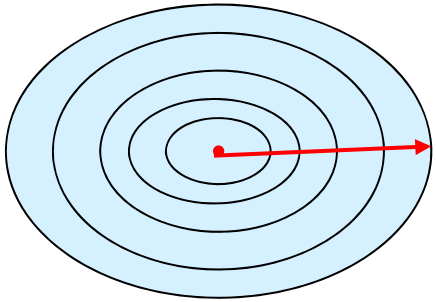
100道开始	
被访问的下一个磁道	移动距离
90	10
58	32
55	3
39	16
38	1
18	20
150	132
160	10
184	24
平均寻道长度：27.5	

5.6.2 磁盘调度

❖SCAN算法

* 在移动方向固定的情况下采用了SSTF，以避免饥饿现象





❖CSCAN(循环扫描)

* SCAN对于进程访问磁头刚扫描过的磁道，需要2T的时间（T为由内到外或相反一次扫描的时间）
* CSCAN规定扫描方向只有一个，从而将上述情况降为 $T+S_{max}$

SCAN调度算法

100道开始，增加方向	
被访问的下一个磁道	移动距离
150	50
160	10
184	24
90	94
58	32
55	3
39	16
38	1
18	20
平均寻道长度：27.8	

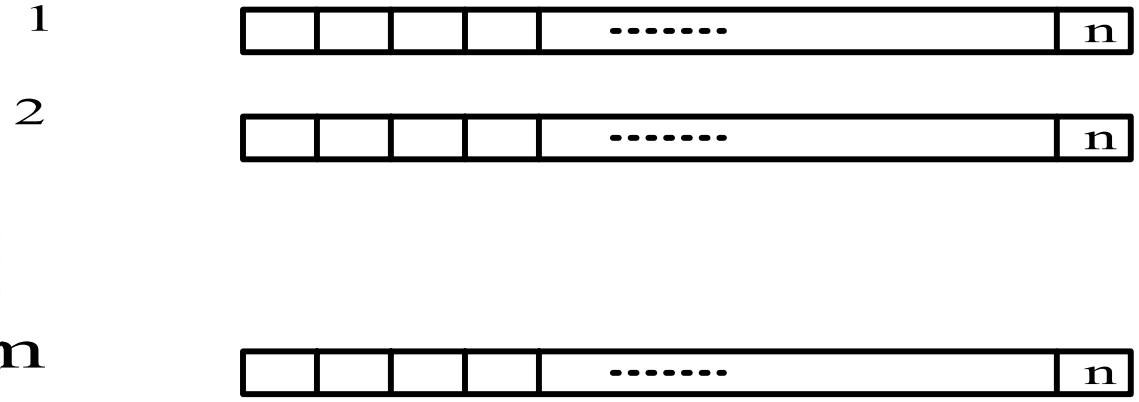
CSCAN调度算法

100道开始，增加方向	
被访问的下一个磁道	移动距离
150	50
160	10
184	24
18	166
38	20
39	1
55	16
58	3
90	32
平均寻道长度：27.5	

5.6.2 磁盘调度

❖ N—Step—SCAN

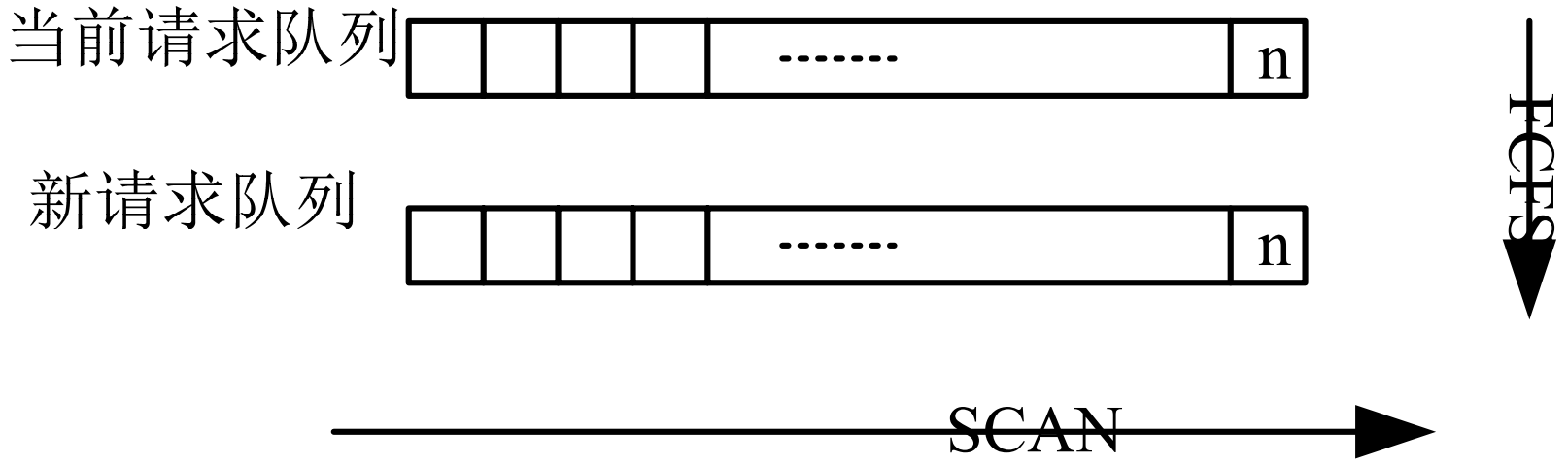
* 粘臂： 由于连续对某磁道访问引起的垄断访问。
* 解决方案： 将磁盘请求队列按FCFS策略，分为长为N的子队列m个。
  - 每个队列采用SCAN算法调度
  - 对一个队列处理完毕，方再处理下一个队列
* 当N=1时，为FCFS。当N很大时，为SCAN.



5.6.2 磁盘调度

❖ FSCAN

对N—Step—SCAN 的简化，仅将访问磁盘的进程划分为两个子队列。



5.6.3 磁盘高速缓存

磁盘访问速度比内存访问速度低4～6个数量级。磁盘I/O已成为计算机系统的瓶颈。提高磁盘I/O采用磁盘高速缓存。

❖磁盘高速缓存的形式形式

* 逻辑上是磁盘、物理上是驻留在内存中的盘块
* 固定大小和可变大小

❖数据交付方式

* 数据交付指将磁盘高速缓存中的数据传送给请求者进程
* 基本步骤：先查缓存→后查磁盘并更新缓存
* 方式：数据交付、指针交付

5.6.3 磁盘高速缓存

❖置换算法（如何管理高速缓存？）

* 最近最久
* 特征：访问频率、可预见性、数据一致性
* 数据一致性策略：将需要一致性的块放在替换队列的头部，优先回写。

❖周期性回写磁盘

* 采用一段时间间隔就将磁盘高速缓存中的数据写回磁盘的策略。（例如**WORD**中定时保存策略）

5.6.4 提高磁盘I/O速度的其它方法

- **提前读：** 对顺序存放的数据，提前访问指令事先读入后续数据。
- **延迟写：** 访问频率高的磁盘块放在替换队列的尾部，减少回写次数，延迟到要被覆盖时再写回。
- 优化物理块的分布
  - 目的是减小磁头移动距离
  - 簇分配方式：将多个连续的块组成一个簇
- 虚拟盘（RAM盘）
  - 与磁盘高速缓存区别：虚拟盘由用户控制；磁盘高速缓存由系统控制。

5.6.5 廉价磁盘冗余阵列

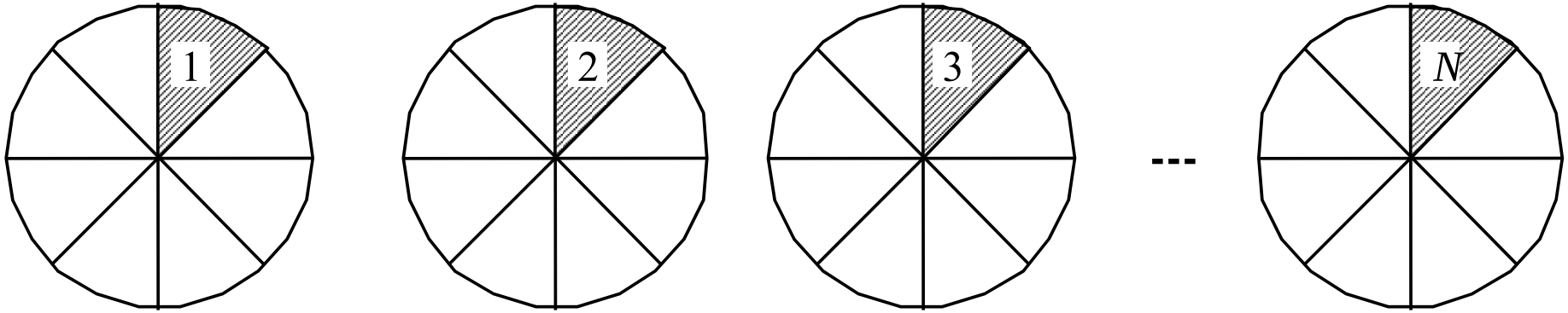
廉价磁盘冗余阵列**RAID(Redundant Array of Inexpensive Disk)** 于1987由美国加利福尼亚大学伯克莱分校提出。利用一台磁盘阵列控制器，统一管理和控制一组磁盘驱动器，组成一个高度可靠的、快速的大容量磁盘系统。

- 存储特点
  - 并行交叉存取（条化存取）
  - 冗余存取
  - 校验存取

5.6.5 廉价磁盘冗余阵列

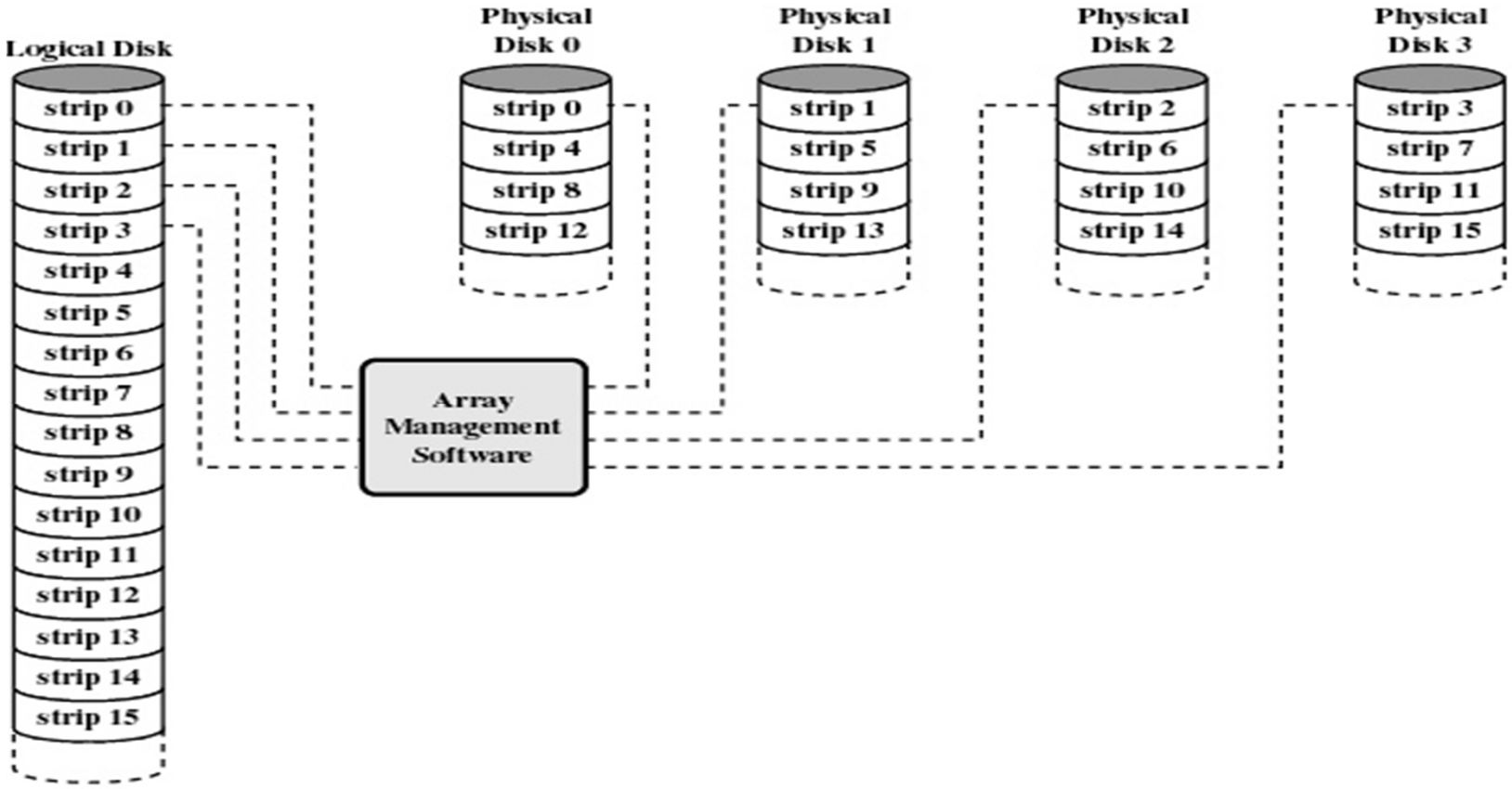
❖并行交叉存取（条化存取）

系统将一个盘块中的数据分为若干个子盘块数据，再把每一个子盘块的数据分别存放到各个不同磁盘中的相同位置上。在读写时，采用并行方式对多个子盘进行读写。



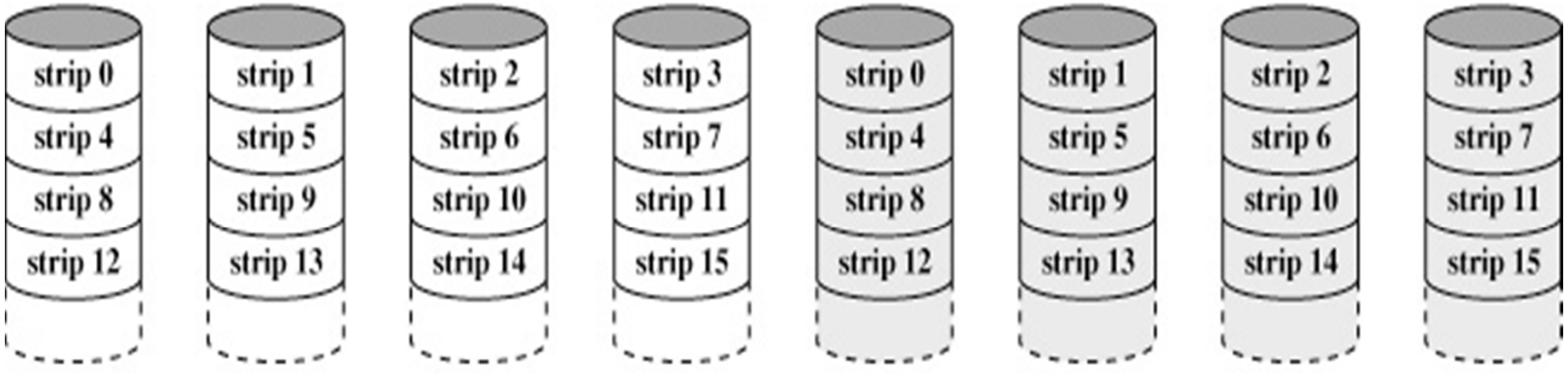
RAID 0

仅提供交叉存取，无冗余，不校验、可靠性差



RAID 1（镜像）

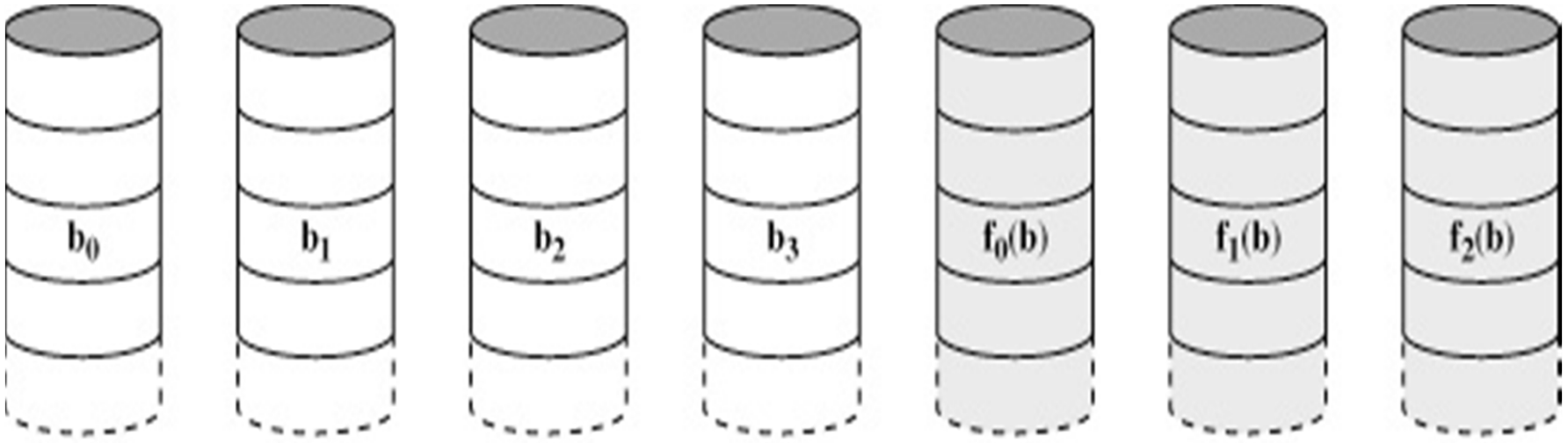
- 分布存放、镜像冗余、不校验
- 读性能比 RAID 0好（选择寻道时间小的磁盘访问）、写性能比 RAID 0差、存储开销大、可靠性高



(b) RAID 1 (mirrored)

RAID 2（汉明码校验冗余）

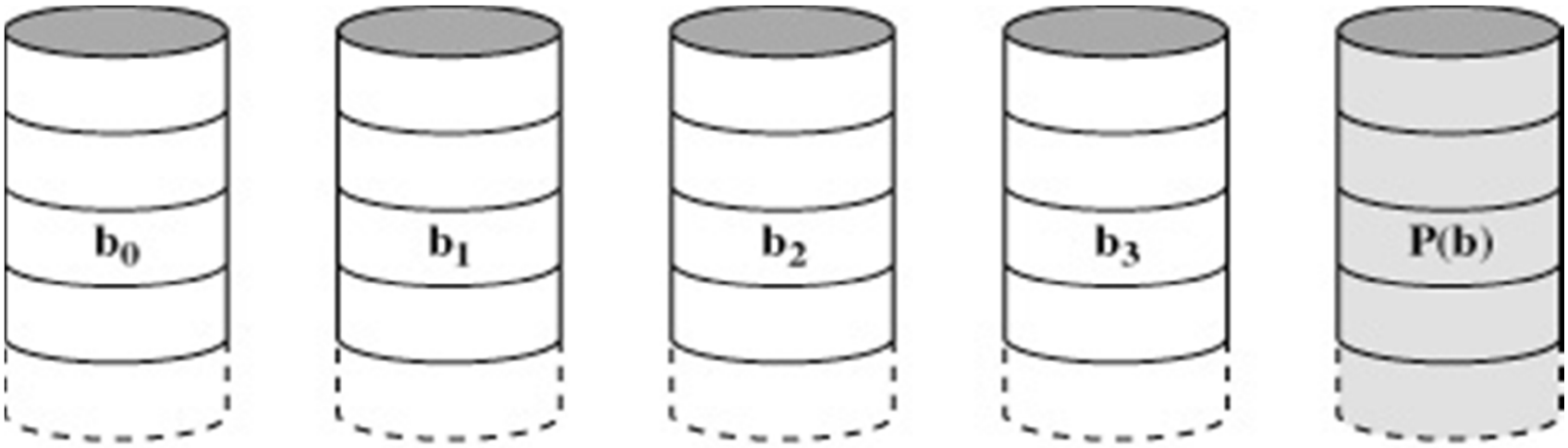
又称位交叉，它采用海明码作盘错校验，采用按位交叉存取，运用于大数据的读写，但冗余信息开销太大（校验盘为多个），已被淘汰。



(c) RAID 2 (redundancy through Hamming code)

RAID 3

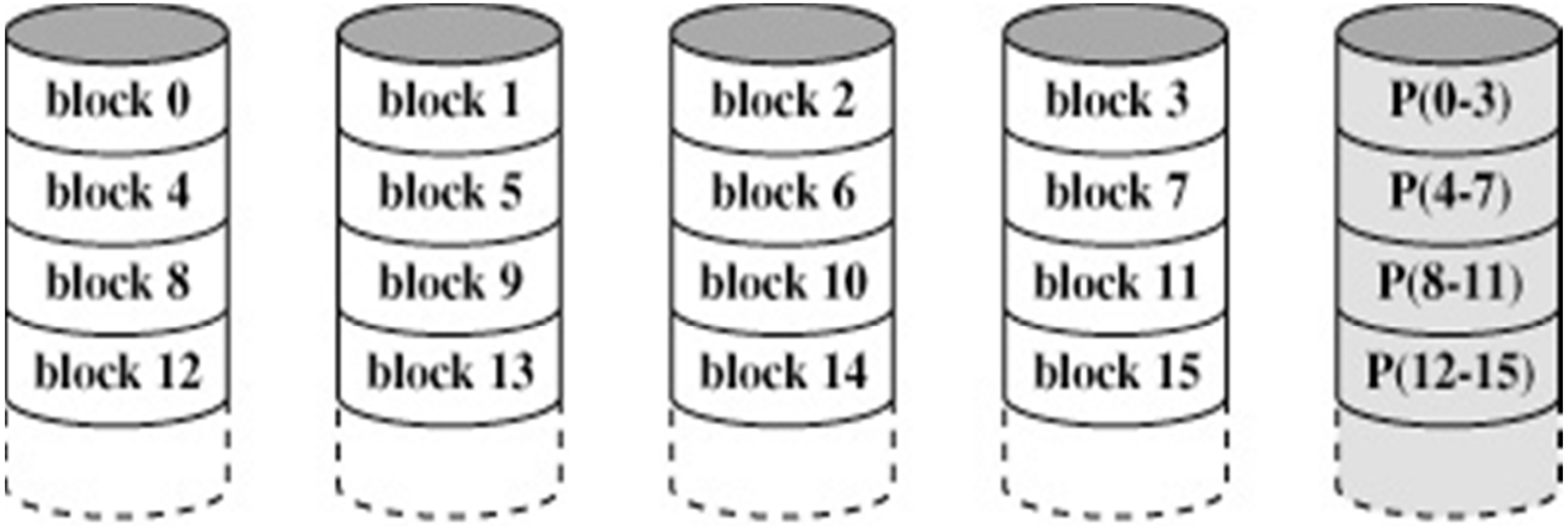
用一个校验盘，减少了保证可靠性所需要的开销



(d) RAID 3 (bit-interleaved parity)

RAID 4 (Block-Level Parity)

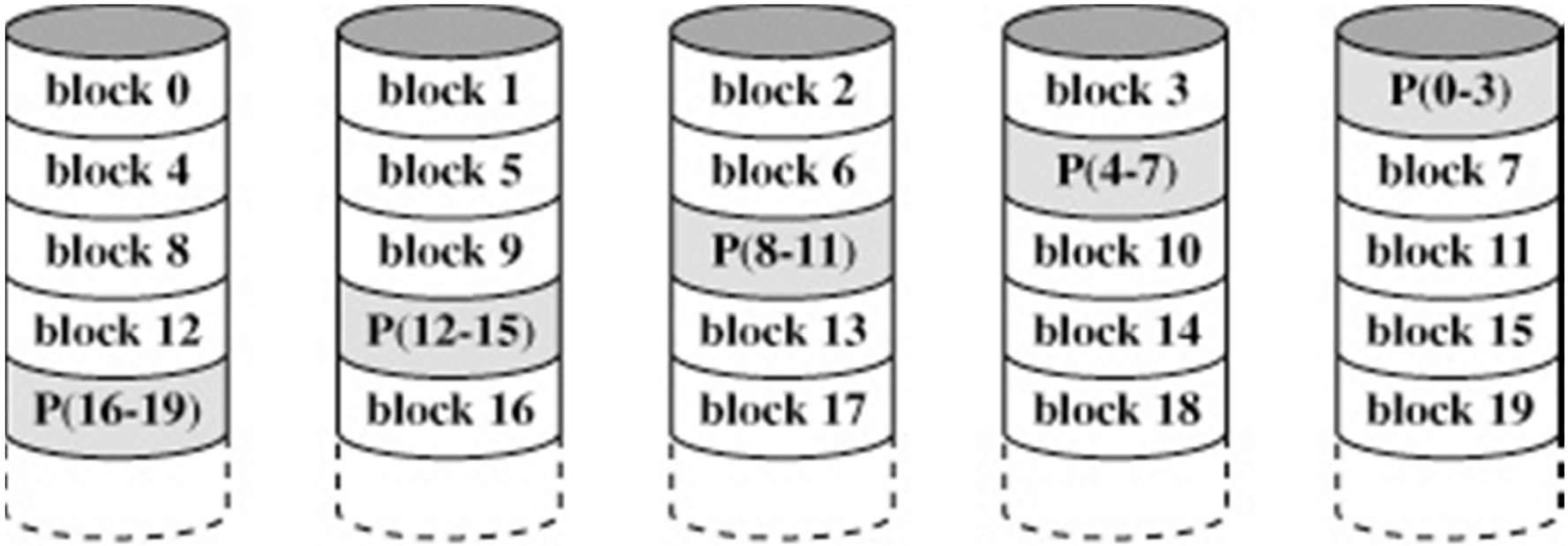
RADI3的改进型，RAID4基于大的块校验



(e) RAID 4 (block-level parity)

RAID 5

每个磁盘驱动器都有自己的数据通路，独立进行读/写，校验信息以螺旋方式分散在所有数据盘上。比RAID 4更可靠。



(f) RAID 5 (block-level distributed parity)

5.6.5 廉价磁盘冗余阵列

❖RAID的优点

* 可靠性高
* 磁盘I/O速度高
* 性价比高

谢谢大家

本章作业

课后习题（P142-P143）

1，2，5，6，9，14，15，16，18，21，23，27共九题