



课程主要内容

👉 操作系统引论 (1章)

👉 进程管理 (2-3章)

👉 存储管理 (4章)

👉 设备管理 (5章)

👉 文件管理 (6章)

👉 操作系统接口 (7章)

人体与计算机类比

◆ 前四章内容学生掌握情况调查：在签名的后面写上你的选项

- a. 90%以上知识能听懂 b. 70%-90%能听懂
C. 60%-70% d. 60%以下 e. 几乎听不懂



第五章 设备管理



设备有哪些? **I/O**

设备正常而且空闲就能分配使用?

设备管理哪些内容?

I/O系统



6和8



第五章 设备管理

- ❖ I/O系统
- ❖ I/O控制方式
- ❖ 缓冲管理
- ❖ 设备分配
- ❖ 设备处理
- ❖ 磁盘存储器管理

本章作业

重点： I/O通道及控制方式

缓冲管理

设备无关性、Spooling技术

磁盘调度算法

必考

难点： 专用缓冲管理、公用缓冲管理输入输出
磁盘调度



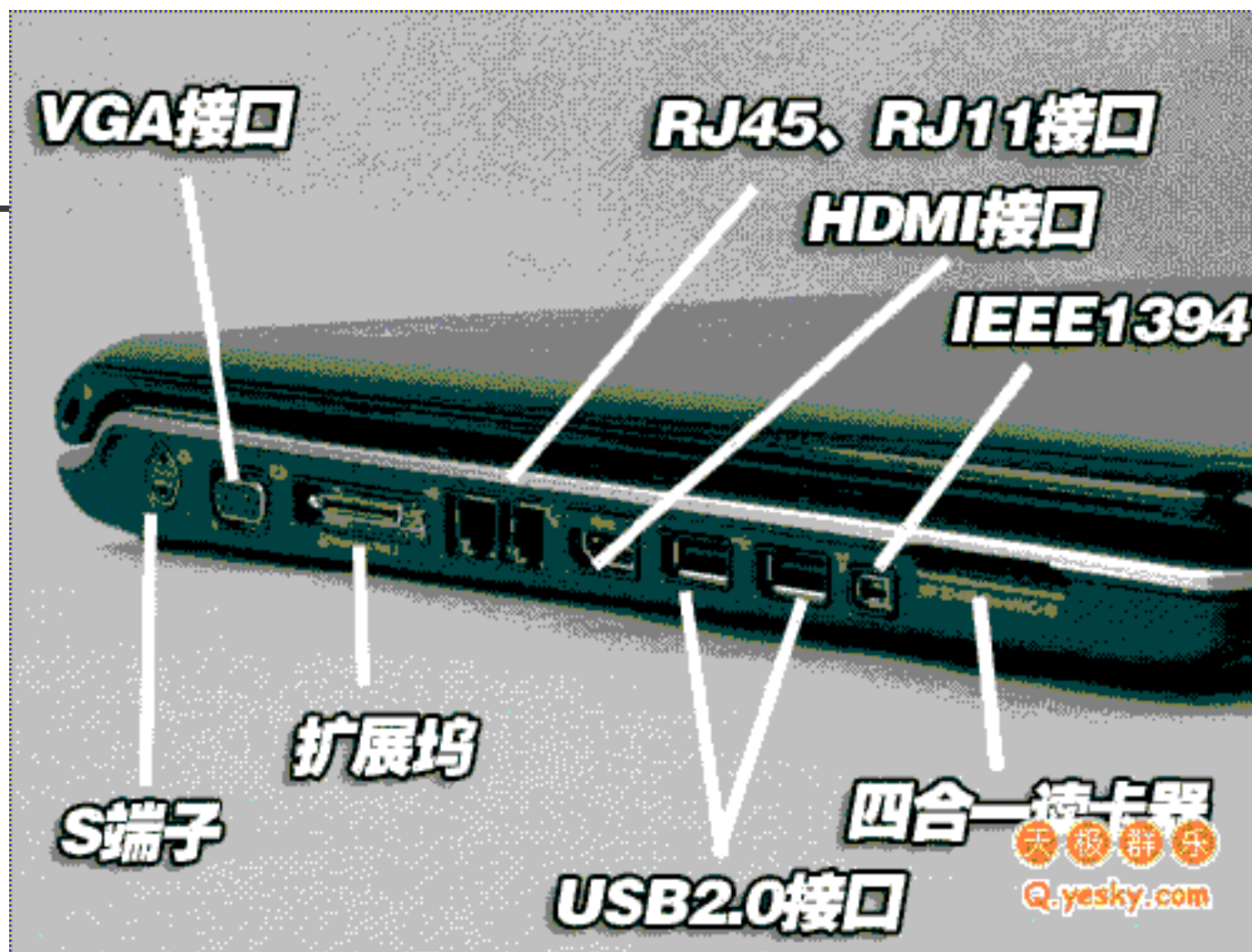


5.1 I/O 系统

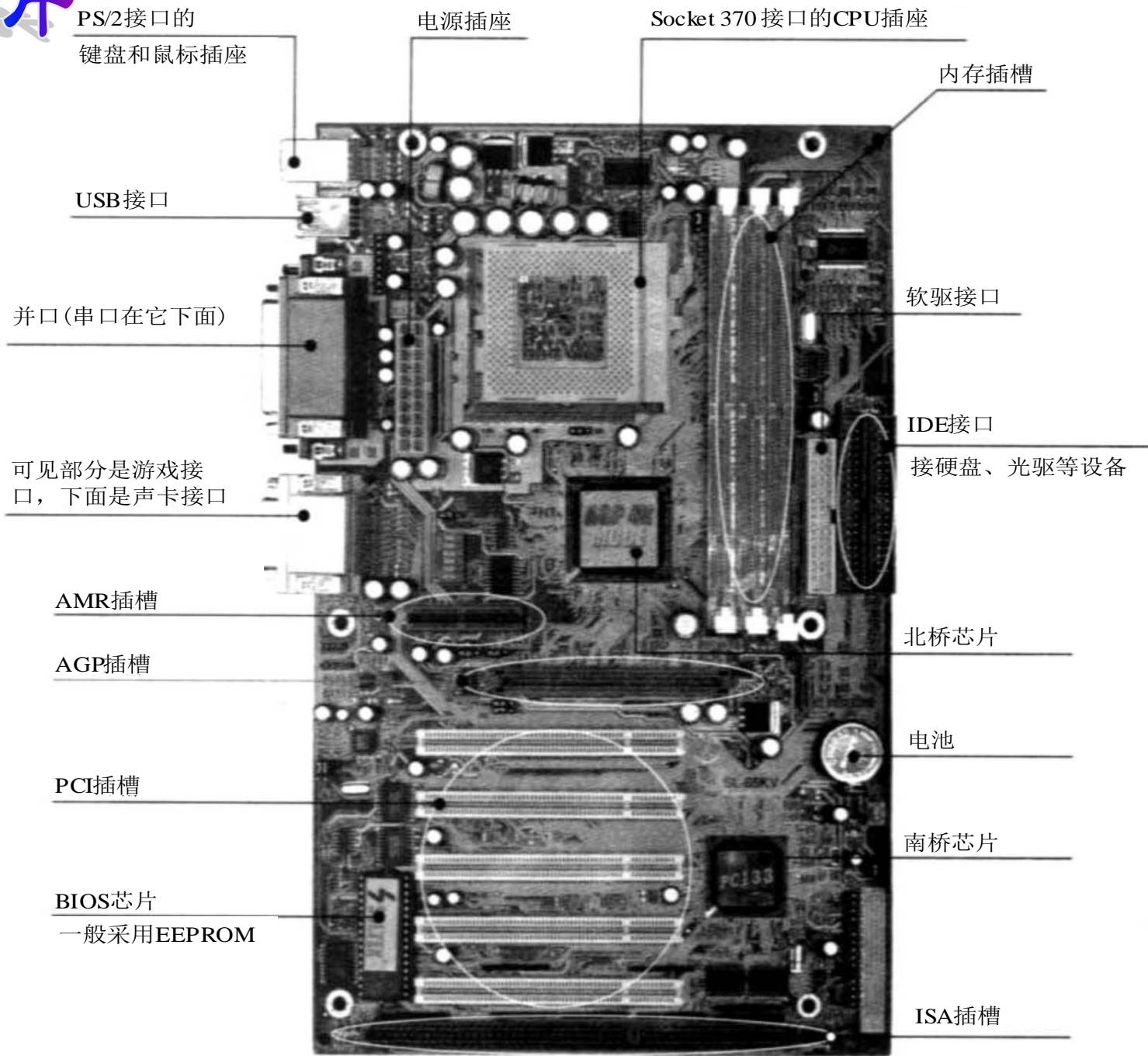
I/O 系统的组成: I/O设备、设备控制器、I/O通道*、总线及相应软件

- ❑ I/O设备
- ❑ 设备控制器
- ❑ I/O通道*
- ❑ I/O系统的总线系统
- ❑ I/O系统的结构

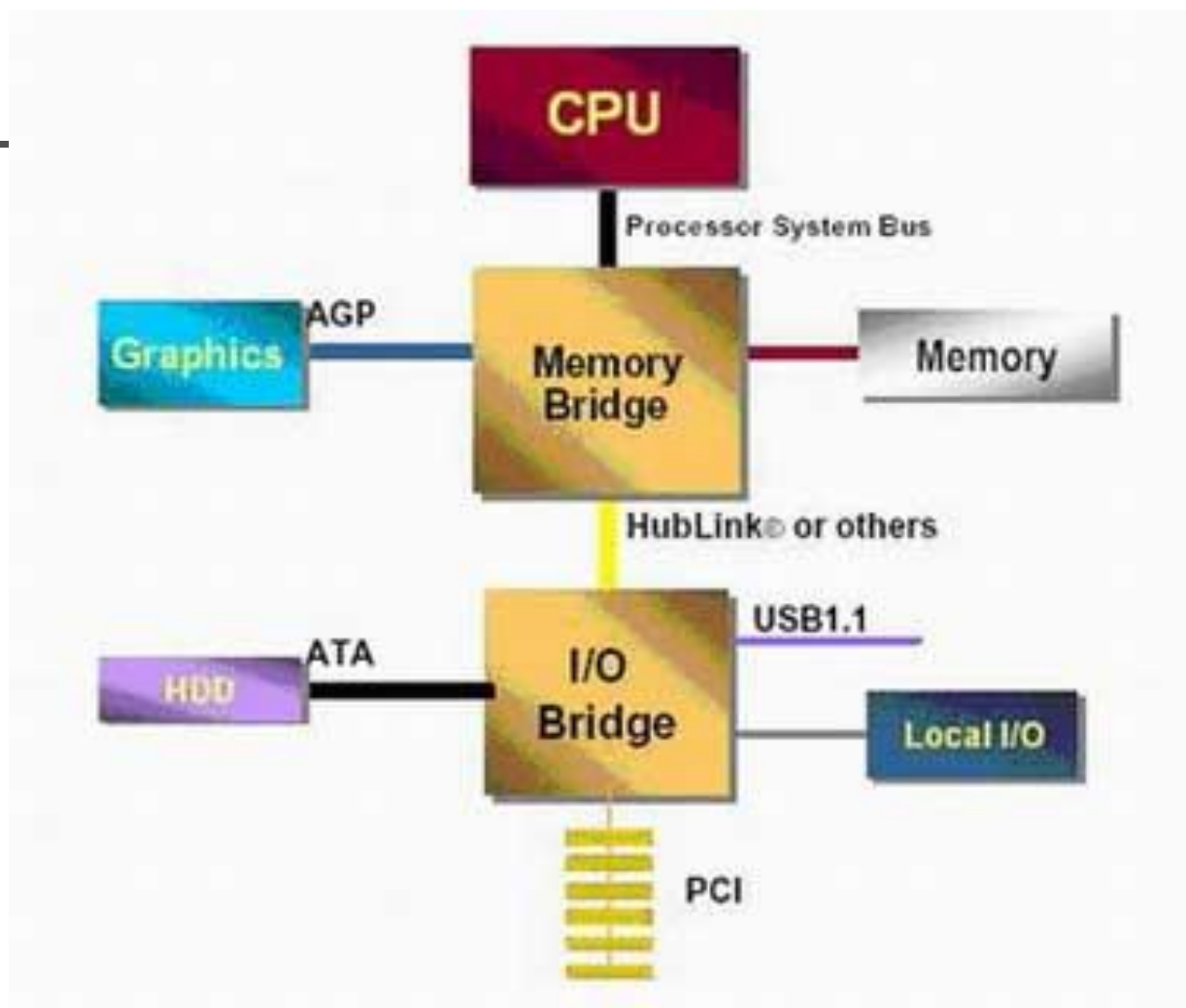
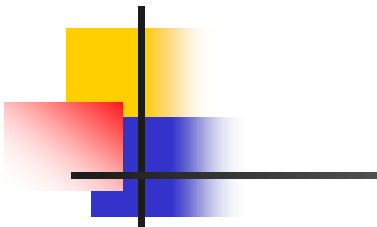




补充图片



补充图片





5.1.1 I/O 设备

1、 I/O设备的类型

系统设备

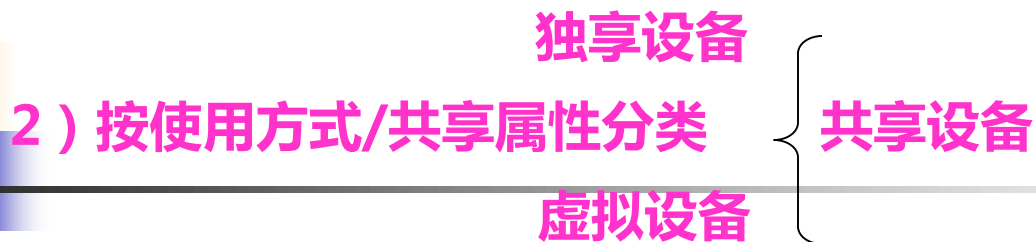
1) 按设备的从属关系分类



用户设备

系统设备：在OS生成时就已登记在系统中的标准设备，如键盘、显示器、打印机等。

用户设备：在OS生成时未登记在系统中的非标准设备，如鼠标、绘图仪、扫描仪等。



独享/独占设备：在一段时间**只允许一个**用户进程访问的设备。

多数低速设备属此类，**打印机**就典型的独享设备。

共享设备：在一段时间**允许多个**用户进程同时访问的设备。**磁盘**就典型的共享设备。

虚拟设备：指通过虚拟技术将一台独占设备变换为若干台逻辑设备，供若干个用户进程同时使用，通常把这种经过虚拟技术处理后的设备称为虚拟设备。

带问题 注意：虚拟设备与前面虚拟技术有什么不同？



3) 按传输速率分类

低速设备

中速设备

高速设备

低速设备：传输速率仅为每秒钟几个字节至数百个字节的设备。

典型的有：**键盘**、鼠标、语音的输入/输出等。

中速设备：传输速率仅为每秒钟数千个字节至数数万个字节的设备。典型的有：**打印机**等。

高速设备：传输速率仅为每秒钟数百千个字节至数十兆字节的设备。典型的有：**磁盘机**、磁带机、光盘机等。

块设备

4) 按信息交换的单位分类

字符设备

块设备：信息交换的基本单位为**字符块**，属于**有结构**设备，块大小一般为512B---4KB，**传输速率较高**，**可寻址**，**DMA方式**。典型的有：**磁盘**、**磁带**等。

字符块输出举例—汇编语言

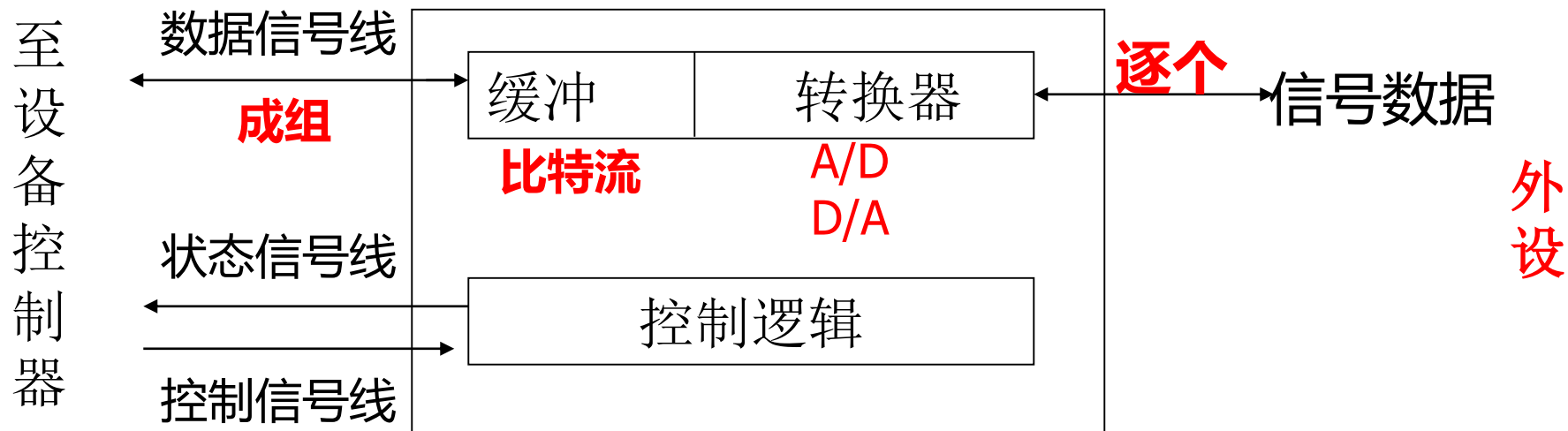
字符设备：信息交换的基本单位为**字符**，**无结构**，**传输速率低**，**不可寻址**，**中断方式**。典型的有：**键盘**、**打印机**和**显示器**等。

思维延伸

有结构、无结构



2、设备与控制器之间的接口 (P145 图5-1)



设备与控制器之间的接口



5.1.2 设备控制器--接口卡

❖ 处于CPU与I/O设备之间的**接口**，接收CPU发来的命令，并控制I/O设备

工作，是一个**可编址设备** 举例分析：主板、U头、串口；OUT AX，

❖ **功能**：接收和识别命令、实现数据交换、设备状态以及识别设备地址

❖ **组成**

- 设备控制器与处理机的接口

- 设备控制器与设备接口

- I/O逻辑

- 寄存器：控制寄存器(存放命令及参数)、数据寄存器(存放数据)、状态寄存器(记录设备状态).







5.1.3 I/O 通道*

存在问题：

- 所有外围设备的I/O工作全部都要由CPU来承担，CPU的I/O负担很重，不能专心于用户程序的计算。
- 大型计算机系统中的外围设备台数虽然很多，但是一般并不同时工作。

- 引入：减少CPU的负担，引入特殊的通道处理机---I/O通道
- 特点：指令类型单一；没有自己的内存，与CPU共享内存
- 根据信息交换方式的不同，通道可分成以下几种类型：
 - 字节多路通道 数组选择通道 数组多路通道

字节多路通道

□ 其工作原理:

数据传送是按**字节交叉**方式

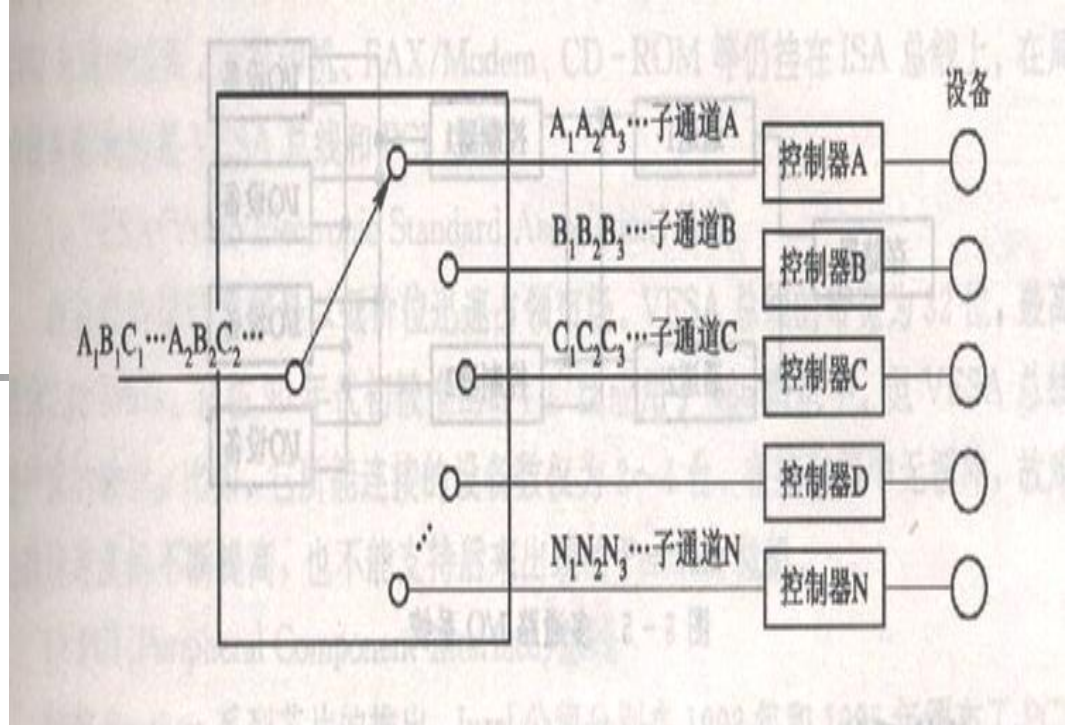
1) 有一个主通道。

2) 含有多个子通道A、B、C……，每子通道通过一控制器与一台中/低速的I/O设备相连，可同时并行向主通道传数据。

3) 各子通道以**时间片轮转方式**按字节交叉使用主通道。

□ **优点**：可连多台中/低速设备；能**分时并行**操作。

□ **缺点**：传输率较低。





数组选择通道

数据传送是按**成组方式**进行工作，**每次传输一批数据**。主要用于连接**高速I/O设备**。

1)有一个主通道 2)含有多个子通道A、B、C.....

3)每子通道通过一控制器与一台中/低速的I/O设备相连，**在一段时间内只能选择一个子通道程序执行。**

优点：可连多台高速设备；传输率较高。

缺点：某子通道不传数据，而使主通道闲置，其它子通道也不能传数据。所以**通道的利用率很低。**



数组多路通道

数据传送仍是按数组方式工作。

工作原理（结合两者：**并行+数组**）

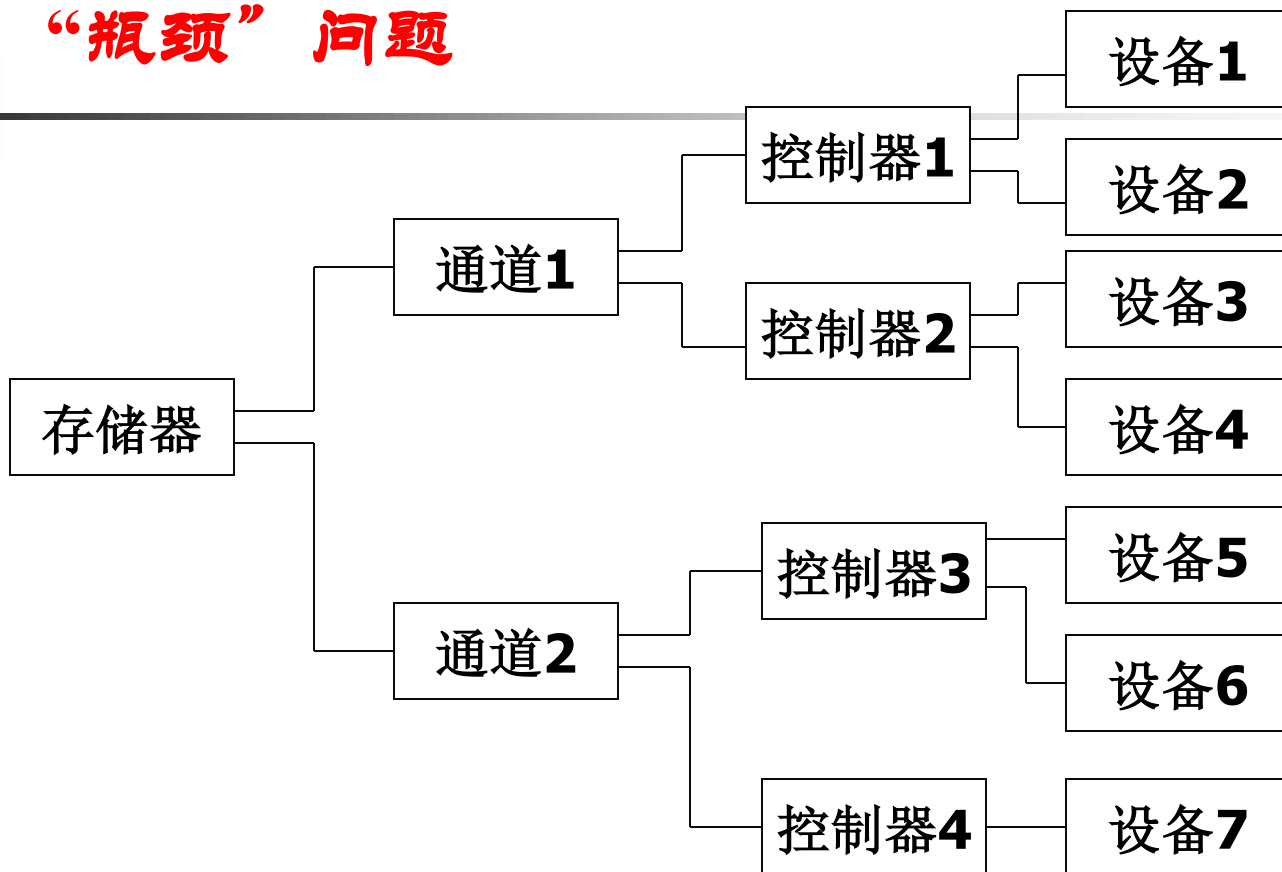
1)有一个主通道 2)含有多个子通道A、B、C.....

3)每子通道通过一控制器与一台高/中速的I/O设备相连，可同时并行向主通道传数据。

4)各子通道以时间片轮转方式按数组方式使用主通道。

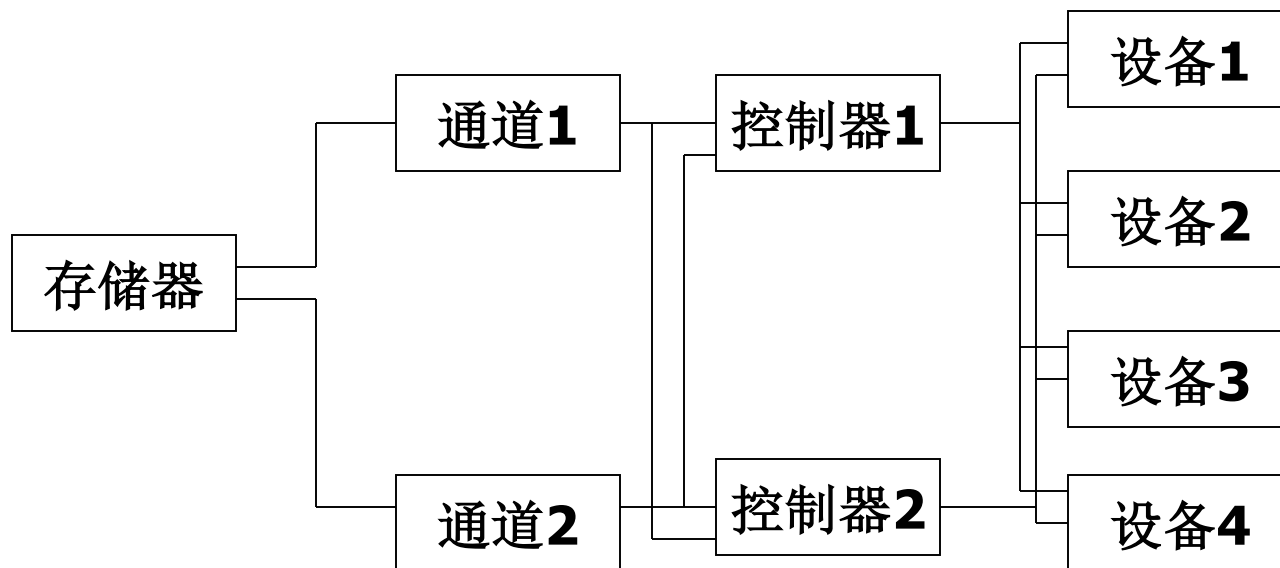
❖ **优点**：可连多台高/中速设备；能分时并行操作，传输率较高。

“瓶颈”问题



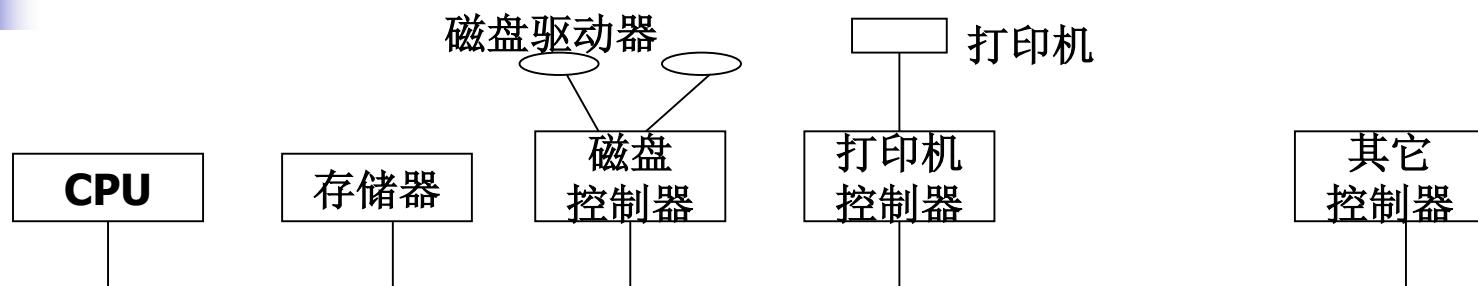
单通路 I/O系统

解决“瓶颈”问题的方法-多路方式



多通路I/O系统

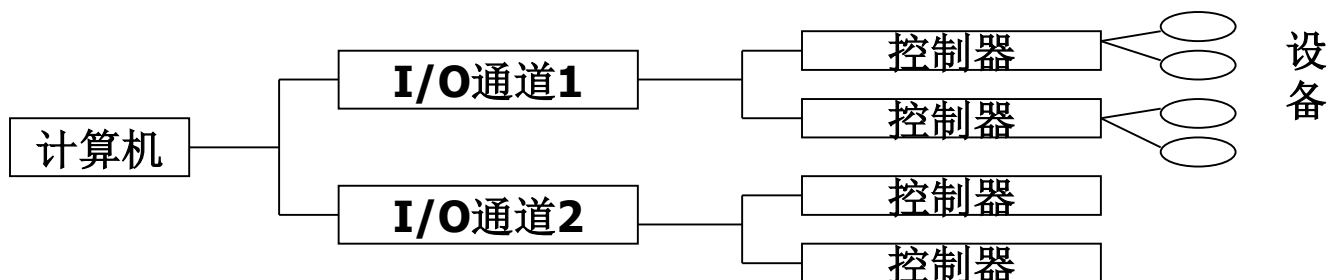
5.1.4 I/O 系统的结构



微型机I/O系统结构---总线型



总线



主机I/O系统—具有通道

通道与总线
光纤电缆、盒子



5.2 I/O 控制方式

常用的输入/输出控制方式：以减少**CPU**的干预为目标

- 1、程序控制方式
- 2、中断控制方式
- 3、直接存储器访问-**DMA**方式
- 4、通道控制方式 *

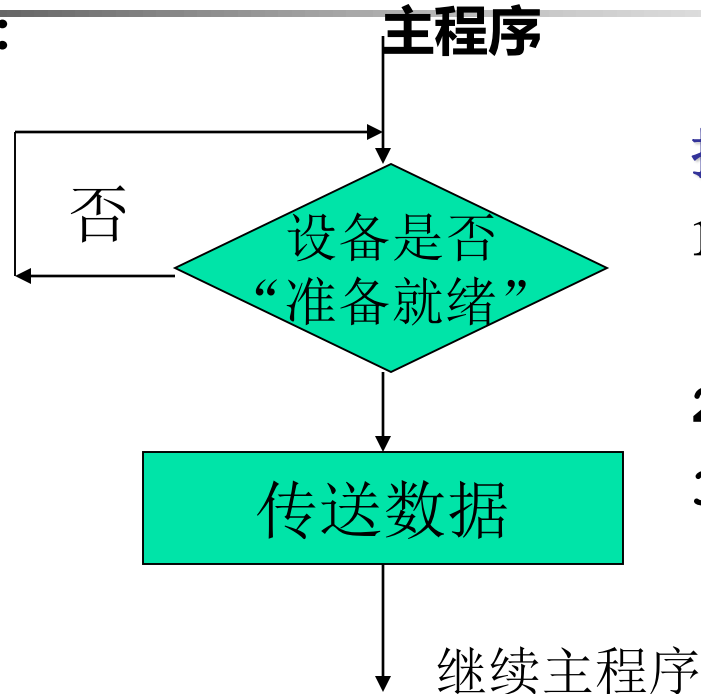


实例贯穿—学生进教室询问过程

1、程序直接控制方式

处理机对I/O的控制采用程序直接控制方式。

工作原理：



控制设备状态的控制位

- 1) **command-ready**
(等待命令)
- 2) **Busy** (忙)
- 3) **Error** (错误)

特点：控制简单，但CPU的利用率低（串行），出现忙等待（循环等待设备的I/O操作）--即**轮询（Polling）**

键盘操作 问题-----I/O中断

2、中断控制方式

- 1) 需数据的进程**向CPU发出指令启动I/O**设备输入数据。
- 2) 该进程**放弃处理机**，等待输入完成。
- 3) **输入完成后**，I/O控制器向CPU发出**中断**请求，CPU收到后，转向中断服务程序。中断服务程序将数据输入寄存器中的数据送指定内存单元，并将原进程唤醒，继续执行。
- 4) 在以后，该进程再被调度，从内存单元取出数据进行处理。

优点—CPU利用率大大提高（可以与I/O设备并行工作）。

缺点---若中断次数较多将耗去大量CPU处理时间。

键盘操作

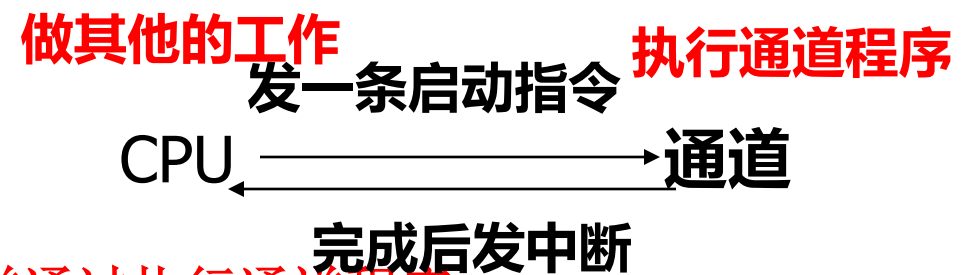


3、DMA方式

- 1) 需数据的进程向CPU发出指令,向**DMA控制器**写入数据存放的内存始址、传送的字节数,并置中断位和启动位,启动I/O设备输入数据并允许中断。
- 3) DMA控制器采用挪用CPU周期,将**一批数据**写入内存中。
- 4) **DMA控制器传送完数据后,向CPU发中断请求**,CPU响应后转向中断服务程序,唤醒进程,并返回被中断程序。

优点—CPU利用率进一步提高(并行度有所提高)。

缺点—数据传送方向、字节数、内存地址等需由CPU控制,且每一设备需一台DMA控制器,**设备增多**时,不经济。



4、通道控制方式*——通道通过执行通道程序

注意： 1) CPU发启动指令指明I/O操作、设备号和对应的通道。
2) 通道接收到CPU发来的启动指令后，取出内存中的通道程序执行，控制设备将数据传送到内存指定区域。

优点——一个通道可控制多设备，所需CPU干预更少。
CPU利用率较高（并行度较高）。

缺点：通道价格较高。

通道程序：由一系列通道指令组成。

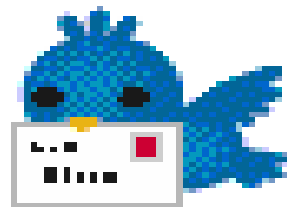
操作	P	R	计数	内存地址
WRITE	0	0	80	813
WRITE	0	0	140	1034
WRITE	0	1	60	5830
WRITE	0	1	300	2000
WRITE	0	0	250	1850
WRITE	1	1	250	720

P：通道指令结束标志
R：记录结束标志

? 与DMA比较



小贴士--I/O 通道



- 引入：减少CPU的负担
- 是一种特殊的处理机
- 特点：指令类型单一；没有自己的内存，与CPU共享内存
- CPU发启动指令给通道，通道通过**执行通道程序**来完成数据输入输出功能。

小贴士





5.3 缓冲管理

1、提高处理机与I/O设备的并行工作的技术：

- 1) 数据传送控制方式 2) 缓冲区技术

2、操作系统中，引入缓冲的主要原因

1) 缓冲CPU与I/O设备间速度不匹配的矛盾 **CPU速度：G级(3.8G)，打印机，M级(40M)。**

2) 减少中断CPU的次数 **举例**

3) 提高CPU与I/O设备的并行性

3、缓冲区：在I/O操作期间，专门用来临时存放输入/输出数据的一块存储区域。

4、分类

单缓冲；双缓冲；多缓冲：循环缓冲、缓冲池

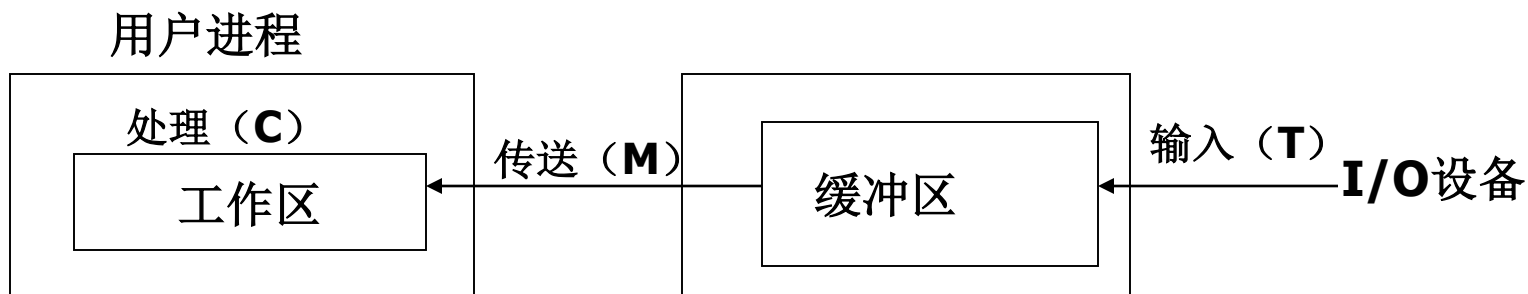


5.3.2 单缓冲和双缓冲

1. 单缓冲

在设备和处理机之间设置一个缓冲。

特点：缓冲区数只有一个；设备与处理机对缓冲区的操作是串行的。



并行？

一块数据的处理时间为多少？

T和C可以并行

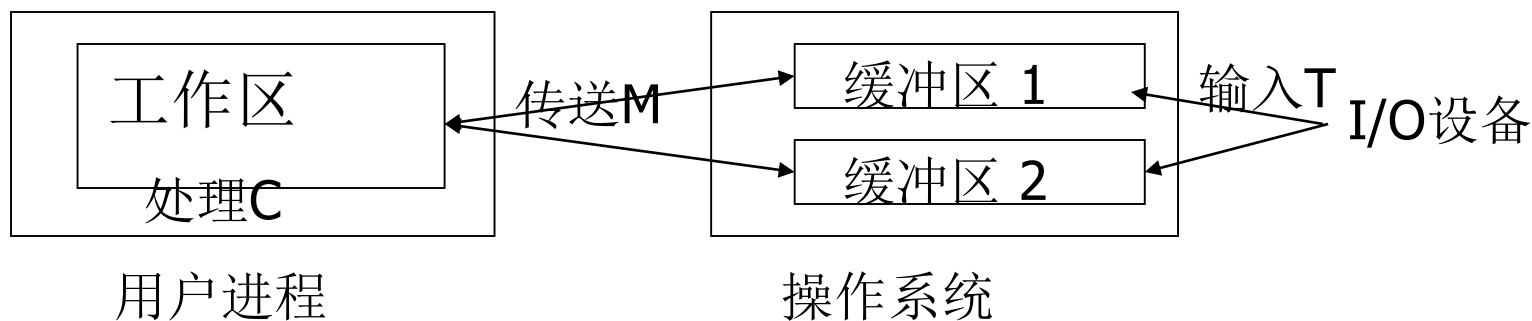
$\text{Max}(T, C) + M$



2. 双 缓 冲

在设备和处理机之间设置2个缓冲。

特点：缓冲区数有2个；设备与处理机对缓冲区的操作可并行，提高了设备与处理机并行操作的程度。



并行？

T和M可以并行，提高并行性

半双工、双工？

5.3.3 循环缓冲 — 专用

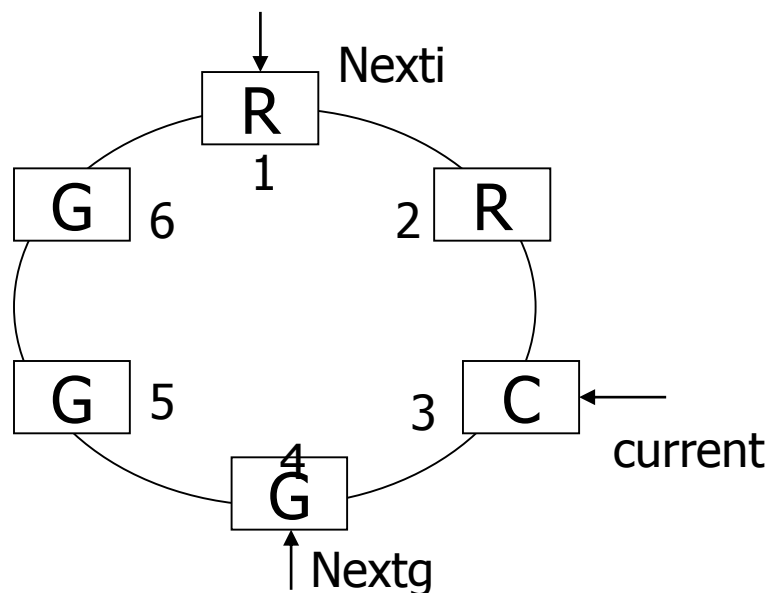
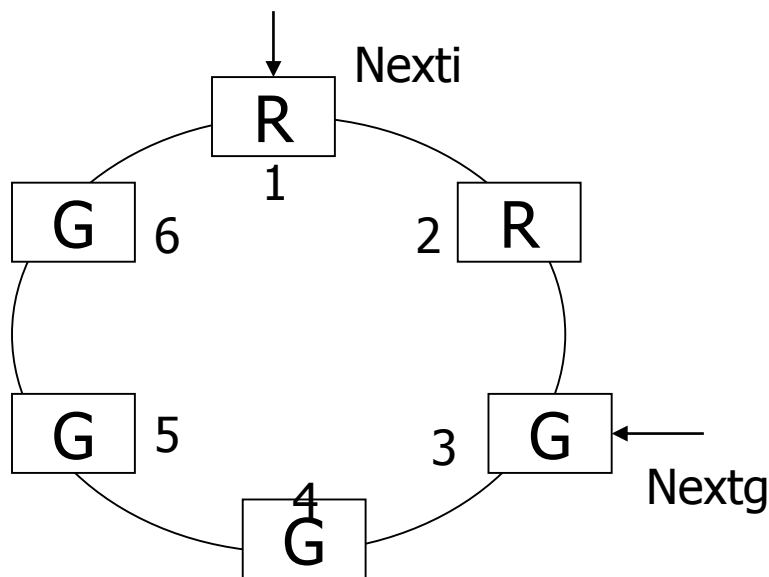


在设备和处理机之间设置多个大小相等的缓冲区，这些缓冲区构成环形，含有2个用于输入/输出的指针IN和OUT。

1. 组成——以输入缓冲区为例

(1) 缓冲区：用于装输入数据的空缓冲区R、已装满数据的缓冲区G、现行工作缓冲区C

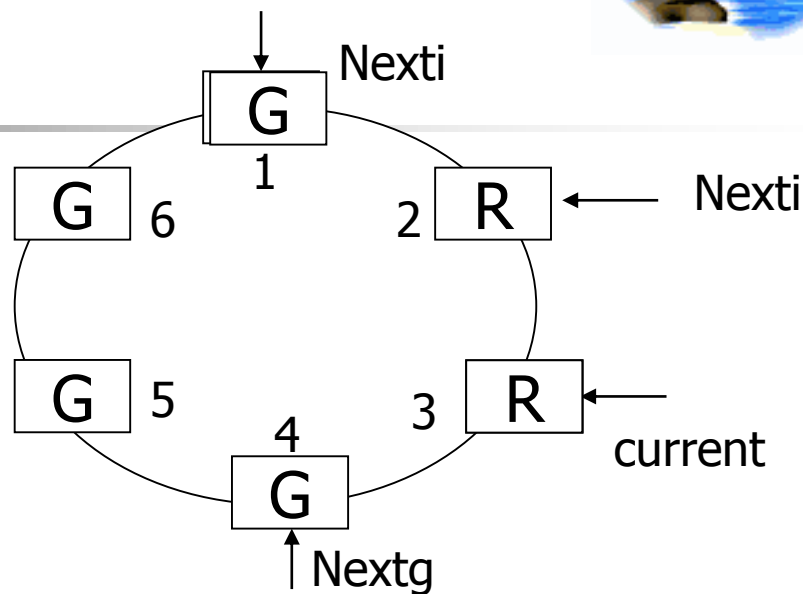
(2) 指针：Nextg、Nexti、Current





2. 缓冲区的使用

- Getbuf过程
- Releasebuf过程



Getbuf、Releasebuf过程

3. 进程同步

- Nexti指针追赶上Nextg指针, 满——输入进程阻塞
- Nextg指针追赶上Nexti指针, 空——计算进程阻塞



5.3.4 缓冲池 — 公用

1、**缓冲池**：能用于输入/输出的缓冲池。缓冲池通常由若干大小相同的缓冲区组成，是系统的公用资源，任何进程都可以申请使用缓冲池中的各个缓冲区。

2、**缓冲池的组成（数据结构）**

缓冲区：空缓冲区、装满输入数据的缓冲区、装满输出数据的缓冲区

队列：空缓冲队列emq、装满输入数据队列inq、装满输出数据队列outq

工作方式的缓冲区：收容输入缓冲区、提取输入缓冲区

收容输出缓冲区、提取输出缓冲区

3、getbuf过程和putbuf过程

Procedure Getbuf(type)

begin

wait(rs(type));

wait(ms(type));

B(number):=Takebuf(type);

signal(ms(type));

end

Procedure putbuf(type)

begin

wait(ms(type));

Addbuf(type,number);

signal(ms(type));

signal(rs(type));

end

Addbuf (type, number)

Takebuf (type)

ms(type):互斥信号量

rs(type):资源信号量





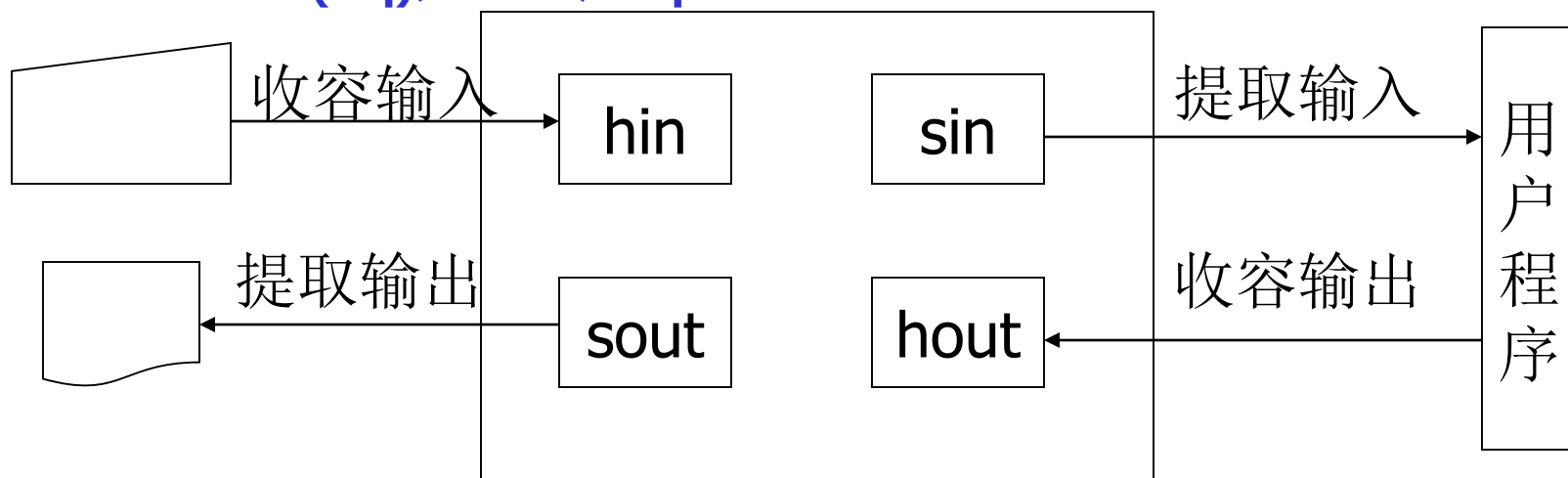
4、操作系统对缓冲池的管理—工作方式

Getbuf(emq), emq → hin

Putbuf(inq), hin → inq 缓冲池

Getbuf(inq) inq → sin

Putbuf(emq), sin → emq



缓冲区的工作方式

思考

输出数据的过程？



小结：操作系统对缓冲池的管理——工作方式

输入进程需要输入数据时：输入设备 → 收容输入缓冲区

- 1) 从空缓冲队列的队首取一空缓冲区用作收容输入缓冲区 **Getbuf(emq), emq → hin**
- 2) 输入设备将数据输入收容输入缓冲区并装满
- 3) 将此缓冲区挂到装满输入数据队列队尾。 **Putbuf(inq), hin → inq**

计算进程需要输入数据时：提取输入缓冲区 → CPU

- 1) 从装满输入数据队列队首取一满缓冲区用作提取输入缓冲区 **Getbuf(inq) inq → sin**
- 2) CPU从提取输入缓冲区中取出数据至用完
- 3) 将空缓冲区挂到空缓冲队列队尾。 **Putbuf(emq), sin → emq**

操作系统对缓冲池的管理--工作方式



计算进程需要输出数据时: **CPU--→**收容输出缓冲区

- 1)从空缓冲队列队首取一空缓冲区用作收容输出缓冲区 **Getbuf(emq), emq—→ hout**
- 2)CPU将数据输入其中并装满
- 3)将收容输出缓冲区挂到装满输出数据队列队尾。 **Putbuf(outq), hout—→ outq**

输出进程需要输出数据时: **提取输出缓冲区-→** 输出设备

- 1)从装满输出数据队列队首取一满缓冲区用作提取输出缓冲区 **Getbuf(outq) outq—→ sout**
- 2)输出设备从中取出数据至用完
- 3)将空缓冲区挂到空缓冲队列队尾 **Putbuf(emq), sout—→ emq**

跳转



上节回顾

- ❖ 设备管理的内容？
- ❖ I/O系统、I/O控制方式—I/O通道技术
- ❖ 问题反馈
- ❖ 缓冲管理

本节概括


- ❖ 设备分配
- ❖ 设备处理

重点：设备无关性、SPOOLING技术
难点：虚拟设备

跳转

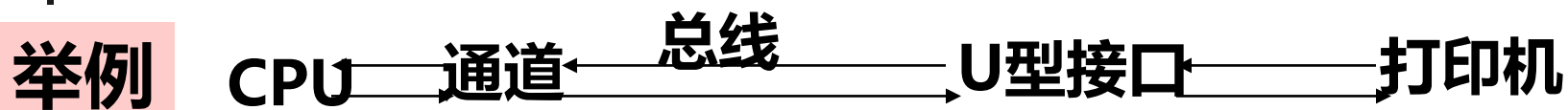


5.4 设备分配

- 设备分配中的数据结构 
- 设备分配的策略/应考虑的因素
- 设备独立性
- 设备分配程序
- **SPOOLING**技术



5.4.1 设备分配中的数据结构



设备控制表 DCT(device control table)

控制器控制表 COCT(controller control table)

通道控制表 CHCT (channel control table)

系统设备表 SDT(system device table)

SDT 记录了所有设备



表目1	
.....	
表目i	
.....	

设备类型	
设备标识符	
DCT指针	
驱动程序入口	
.....	

DCT

设备类型	
设备标识符	
设备状态：忙/闲	
COCT指针	
设备等待队列指针	
.....	

COCT

控制器标识符	
控制器状态：忙/闲	
CHCT指针	
控制器等待队列指针	
.....	

CHDT

通道标识符	
通道状态：忙/闲	
通道等待队列指针	
.....	

跳转



5.4.2 设备分配策略/应考虑的因素

- 1、设备的使用性质/固有属性：（独享分配、共享分配、虚拟分配）
- 2、设备分配算法：（先请求先服务、优先级高者优先）
- 3、设备分配的安全性（防止进程死锁、并发）
- 4、**设备独立性**：是指用户在编制程序时所用的设备(逻辑)与实际使用的设备无关

5.4.3 设备独立性--重点掌握

❖ 设备独立性概念 (**设备无关性**) : 应用程序独立于具体使用的物理设备

逻辑设备 实际执行时

❖ 逻辑设备名到物理设备名映射的实现--**逻辑设备表LUT** (Logical Unit Table)

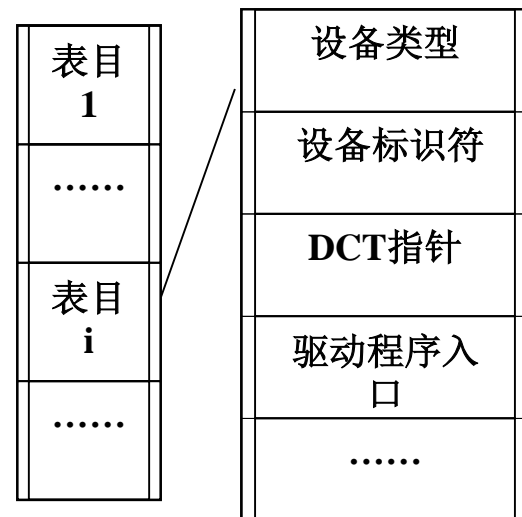
❖ LUT设置问题(图5-18):整个系统设置一张LUT ; 每个用户设一张LUT

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

逻辑设备名	系统设备表指针
/dev/tty	3
/dev/print	5
...	...

(a)单用户、多用户

(b)与图5-17的 (c)



SDT

(C)

类比、举例

❖ 优点

- 设备分配时的灵活性 空闲
- 易于实现I/O重定向 举例

CPU

设备独立性软件

设备驱动程序

设备

逻辑设备表

显示终端：显印器

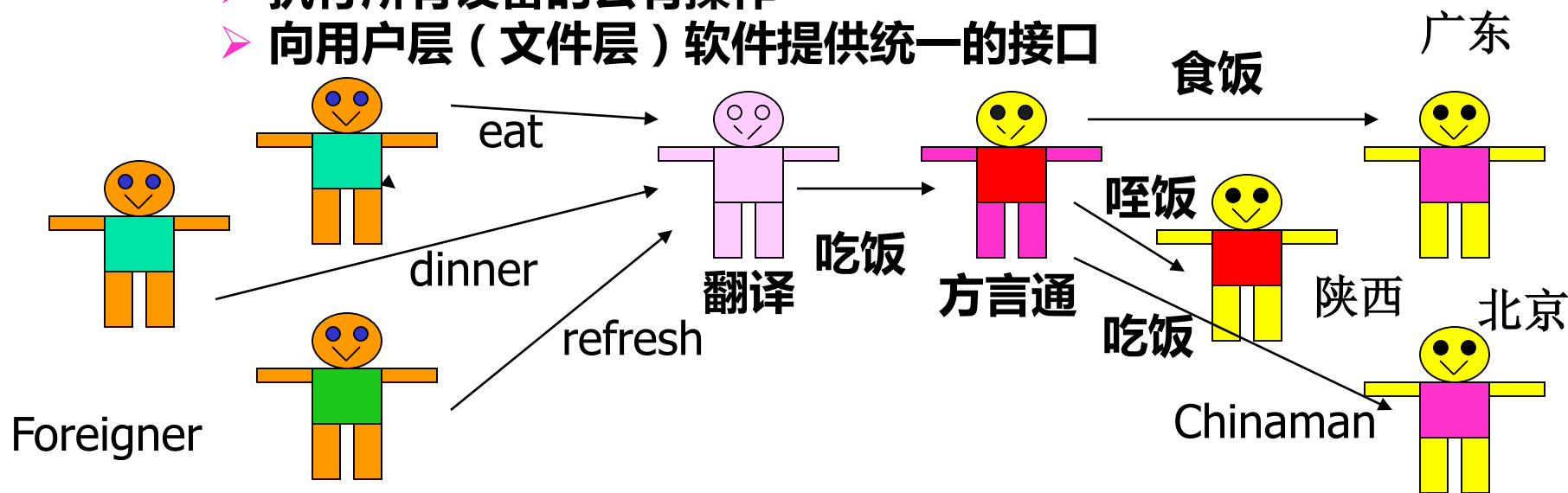
调试程序时

打印运行结果时，
不用修改应用程序

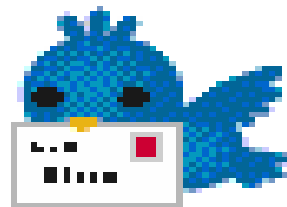
I/O重定向

❖ 设备独立性软件

- 执行所有设备的公有操作
- 向用户层（文件层）软件提供统一的接口



小贴士——设备独立性



❖ 设备独立性概念（设备无关性）

- 为提高OS的可适应性和可扩展性，而**将应用程序独立于具体使用的物理设备。**

❖ I/O重定向

- 用于I/O操作的**设备可以更换**，即重定向，**不必改变应用程序。**

❖ 所有设备的公有操作

- 独立设备的分配与回收；将逻辑设备名映射为物理设备名；对设备进行保护（禁直访）；缓冲管理；差错控制。



小贴士



5.4.4 独占设备的分配程序一略

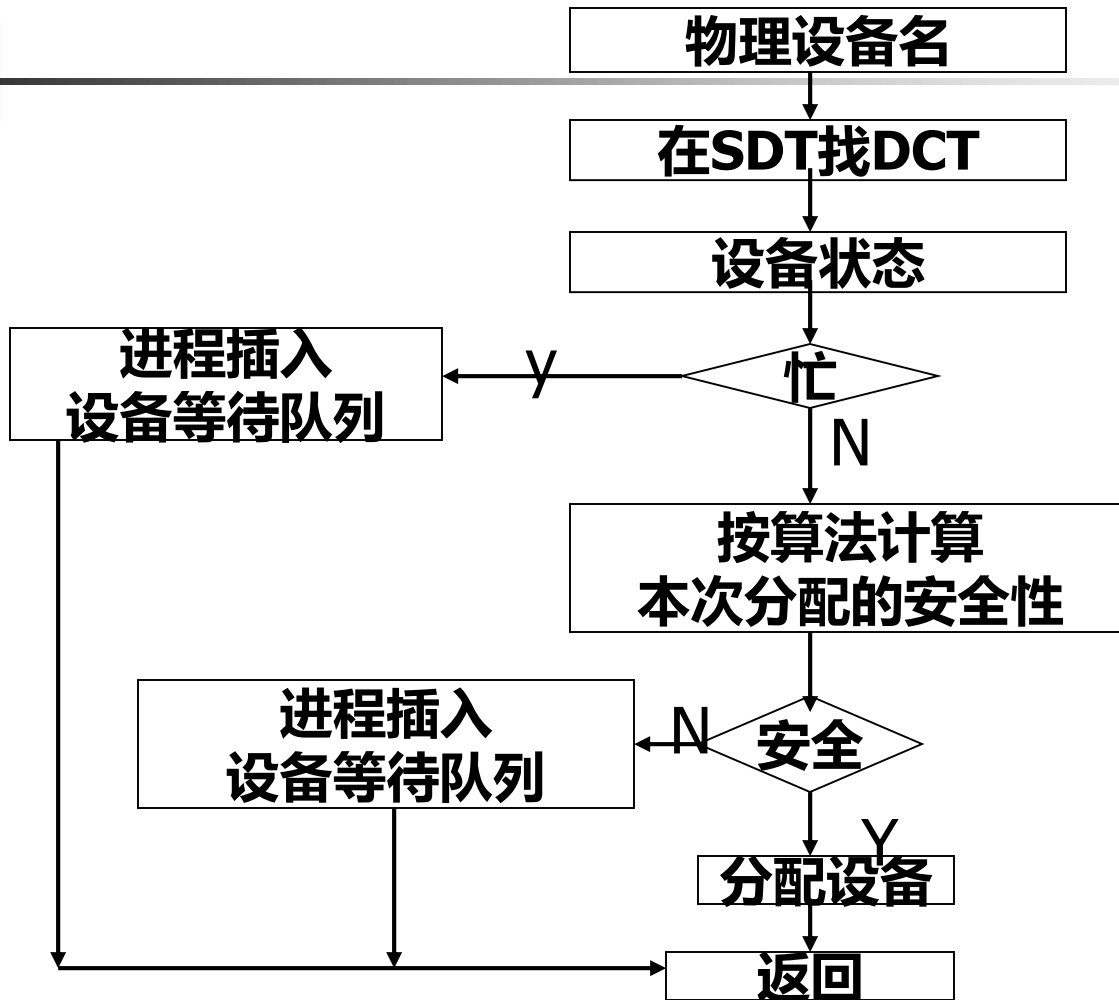
❖ 基本的设备分配程序

- 分配设备
- 分配控制器
- 分配通道
- 问题（“瓶颈”）
 - 进程以物理设备名来提出I/O请求
 - 采用的是单通路的I/O系统结构

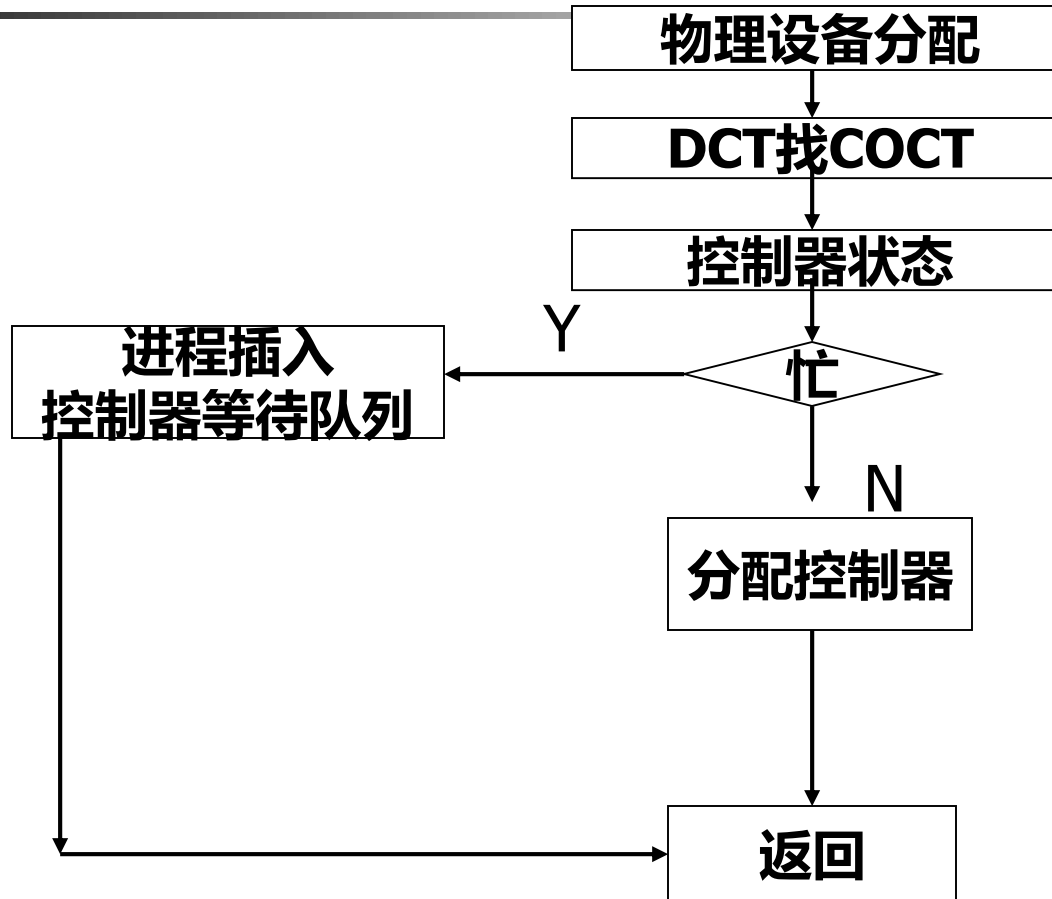
❖ 设备分配程序的改进

- 增加设备的独立性(进程以逻辑设备名来提出I/O请求)
- 考虑多通路情况

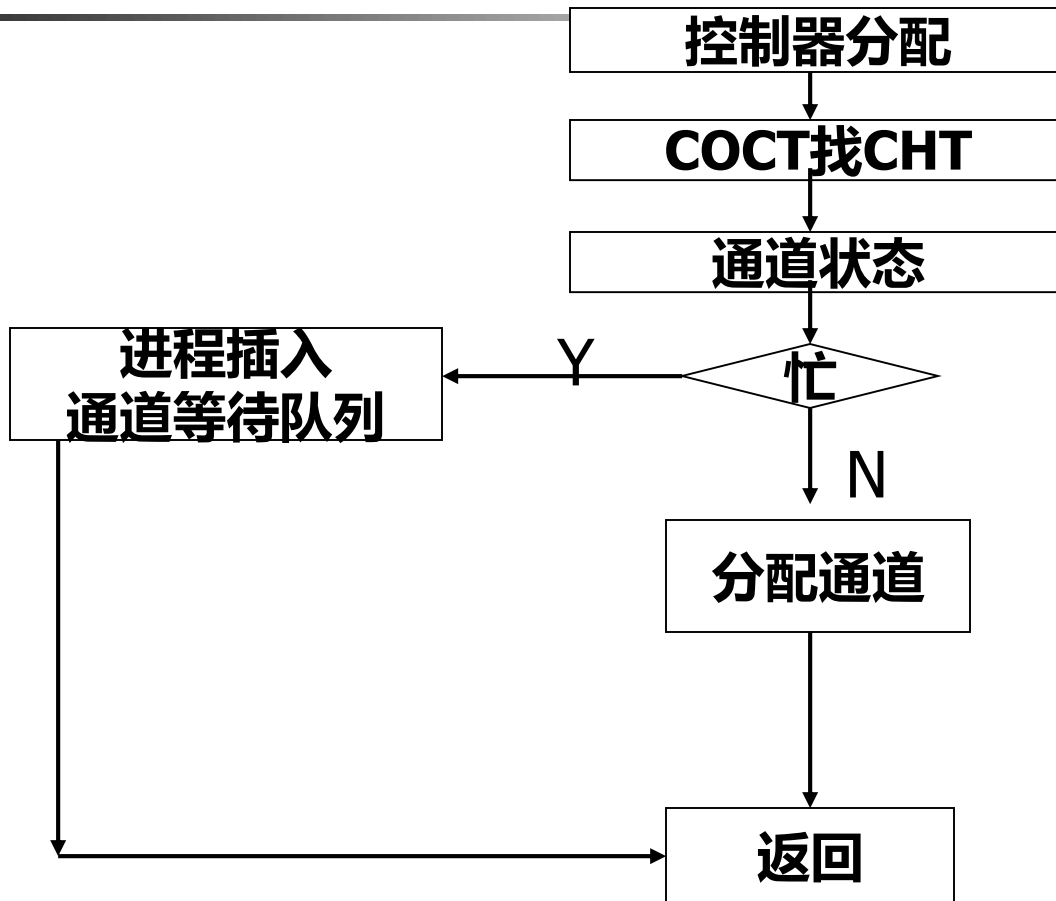
分配设备



分配控制器



分配通道



引入：I / O控制方式和缓冲区；继续匹配速度，提高并行

5.4.5 SPOOLING技术(Simultaneous Peripheral Operations On-Line) --重点掌握

❖ 脱机输入、输出技术



为了缓和CPU的高速性与I/O设备的低速性间矛盾而引入，
在**外围控制机**的控制下实现低速的I/O设备与高速的磁盘之间进行
数据传送。

❖ SPOOLING技术

❖ SPOOLING系统的组成

❖ SPOOLING系统的特点



SPOOLING技术

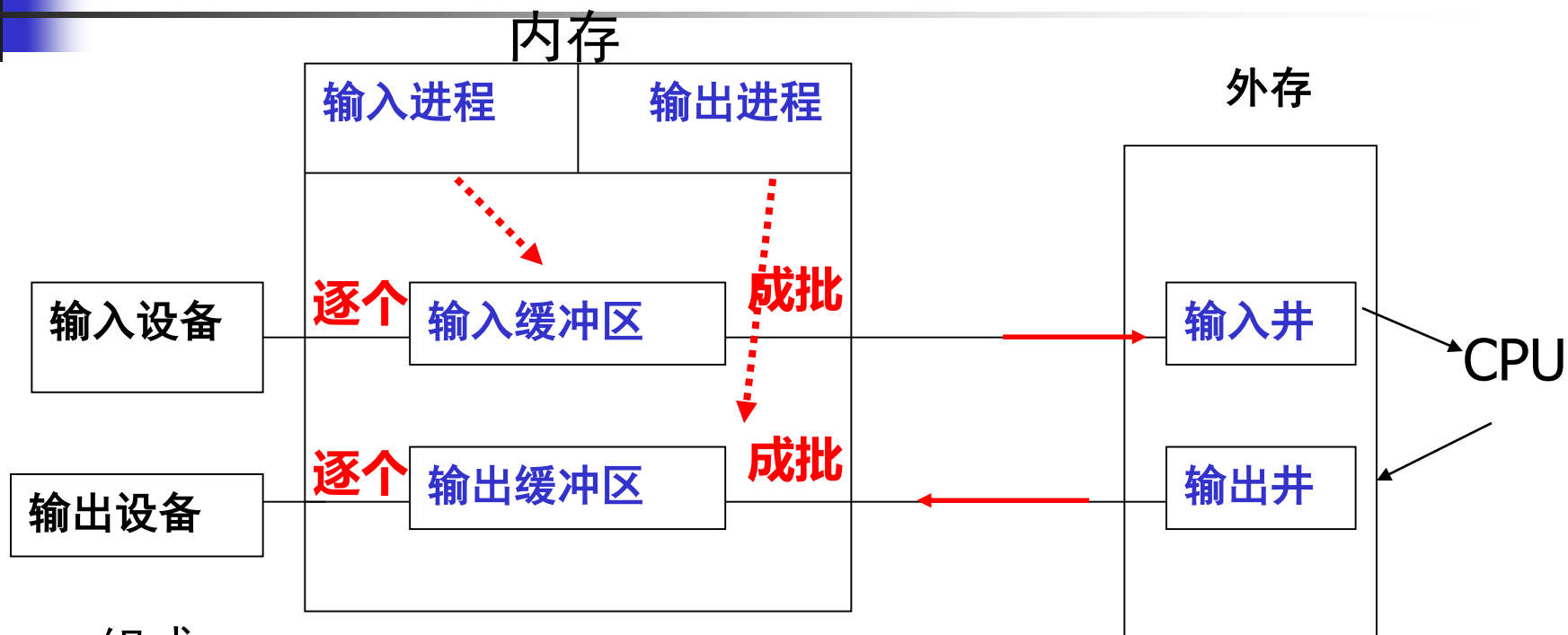
(Simultaneous Peripheral Operations On-Line)

❖ SPOOLING技术

在多道程序下，用一程序来模拟外围控制机，实现将数据从磁盘传送到低速的输出设备上，从而可在主机的直接控制下，实现脱机输入、输出功能，进而实现外围操作与CPU对数据处理的并行操作，这种在联机情况下实现的~~同时~~外围操作称为SPOOLING技术，是对脱机输入、输出工作的模拟，是操作系统中采用的一项将独占设备改造成为共享设备的技术。



❖ SPoolING系统的组成





❖ SPOOLING系统的特点

- 提高了I/O速度
- 将独占设备改造为共享设备
- 实现了**虚拟设备**功能

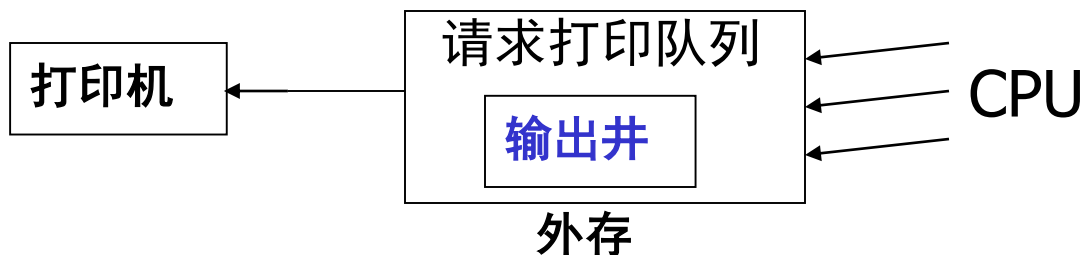
❖ 在操作系统中，引入虚拟设备的原因

引入虚拟设备是为了克服独占设备速度较慢、降低设备资源利用率的缺点，从而提高设备的利用率。

❖ 虚拟设备

是指通过虚拟技术将**一台独占设备**变换为若干台逻辑设备，供若干个用户进程使用，通常把这种经过虚拟技术处理后的设备称为虚拟设备。

举例：打印机





? 讨论

讨论

思考：

虚拟设备与前面的虚拟技术一样吗？



5.5 设备处理—了解、自行

❖ 设备驱动程序的功能和特点

- 设备驱动程序的功能
- 设备处理方式
- 设备驱动程序的特点

❖ 设备驱动程序的处理过程

❖ 中断处理程序的处理过程



设备驱动程序的功能

- ❖ **将接收到的抽象要求转换为具体要求。**
- ❖ 检查用户I/O请求的合法性， I/O设备状态，传参数，设设备的工作方式。
- ❖ 按处理机的I/O请求去启动指定的设备进行I/O操作
- ❖ 及时响应由控制器或通道发来的中断请求，并进行相应处理
- ❖ 按I/O请求构成相应通道程序。



设备处理方式

- ❖ 为每一类设备设置一进程，专门执行其I/O操作。
- ❖ 在整个系统中设置一个进程，执行所有的I/O操作。
- ❖ 不设置专门的设备处理进程，而为各类设备设置相应的设备驱动程序。



设备驱动程序的特点

- ❖ 是请求I/O的进程与设备控制器之间的一个通信程序。
- ❖ 与I/O设备的特性紧密相关
- ❖ 与I/O控制方式紧密相关
- ❖ 与硬件紧密相关，因而**其中一部分程序必须用汇编语言编写。**



设备驱动程序的处理过程

- ❖ 将接收到的抽象要求转换为具体要求。
- ❖ 检查用户I/O请求的合法性
- ❖ 读出和检查 I/O设备状态
- ❖ 传送必要参数
- ❖ 设置设备的工作方式。
- ❖ 按处理机的I/O请求去启动指定的设备进行I/O操作



中断处理程序的处理过程

- ❖ 唤醒被阻塞的驱动程序进程
- ❖ 保护被中断进程的CPU环境
- ❖ 分析中断原因、转入相应的设备中断处理程序
- ❖ 进行中断处理
- ❖ 恢复被中断进程的现场



上节回顾

❖ 缓冲管理 **重点**

❖ 设备分配 **重点**—设备无关性、SPOOLING技术、两层软件的作用、关系？

❖ 设备处理—略

本节概括

❖ 磁盘存储器管理—性能、调度算法、磁盘高速缓存

❖ 作业习题分析

重点、难点：磁盘性能、磁盘调度算法—寻道时间



5.6 磁盘存储器

(1) 磁盘性能简述

提高磁盘I/O速度的主要途径:

(2) 磁盘调度算法—重点

(3) 设置磁盘高速缓存 (Disk Cache)

(4) 其它方法

(5) 采用高度可靠、快速的容量磁盘系统__廉价磁盘冗余阵列

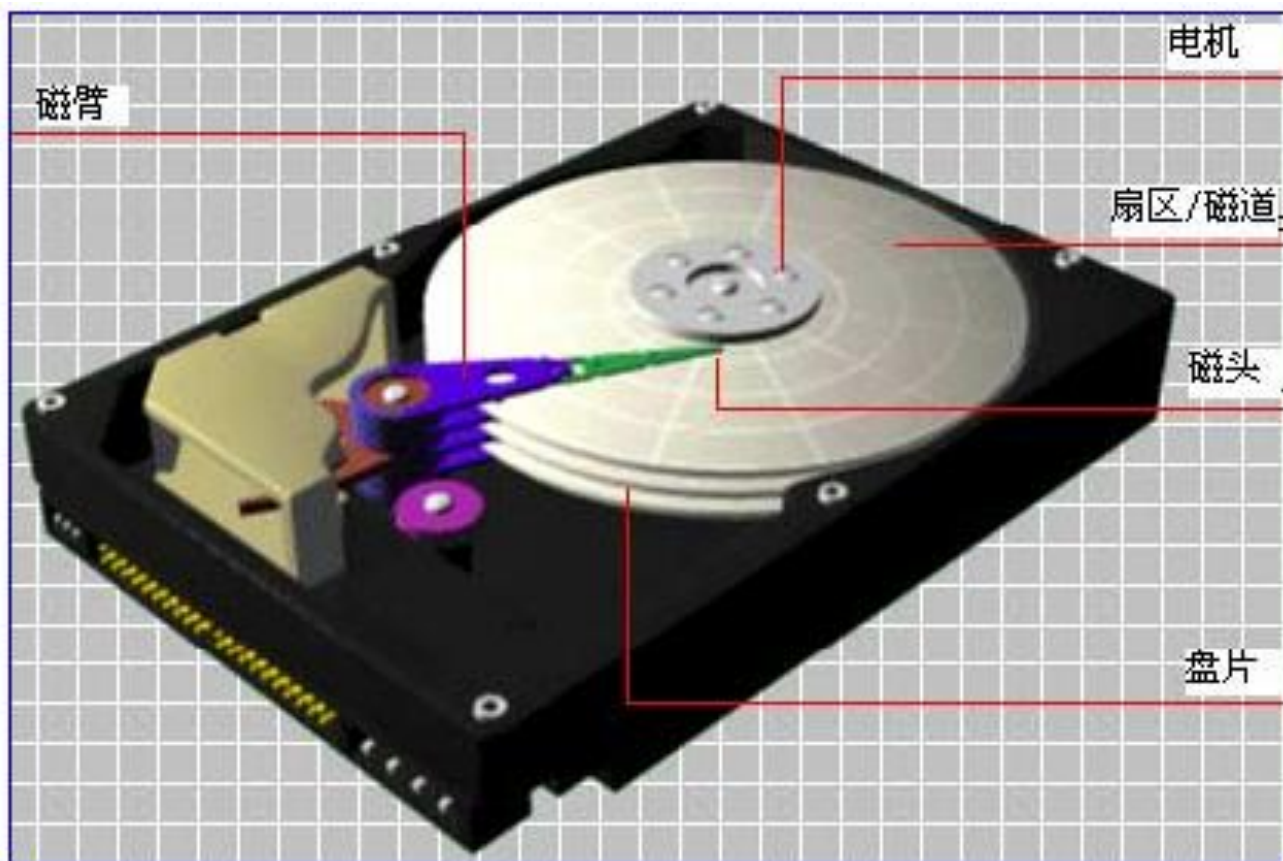
硬盘



硬盘结构



- ❖ 硬盘的盘片组在2~14片不等，通常有2~3个盘片，故盘面号（磁头号）为0~3或0~5
- ❖ 硬盘的每一个盘片都有两个盘面（Side），即上、下盘面，按顺序从上至下从“0”开始依次编号。
- ❖ 磁道从外向内从0开始顺序编号。硬盘的每一个盘面有300~1024个磁道，新式大容量硬盘每面的磁道数更多。
- ❖ 扇区从“1”开始编号，一个标准的3.5in硬盘盘面通常有几百到几千条磁道。磁道是“看不见”的，只是盘面上以特殊形式磁化了的一些磁化区，在磁盘格式化时就已规划完毕。

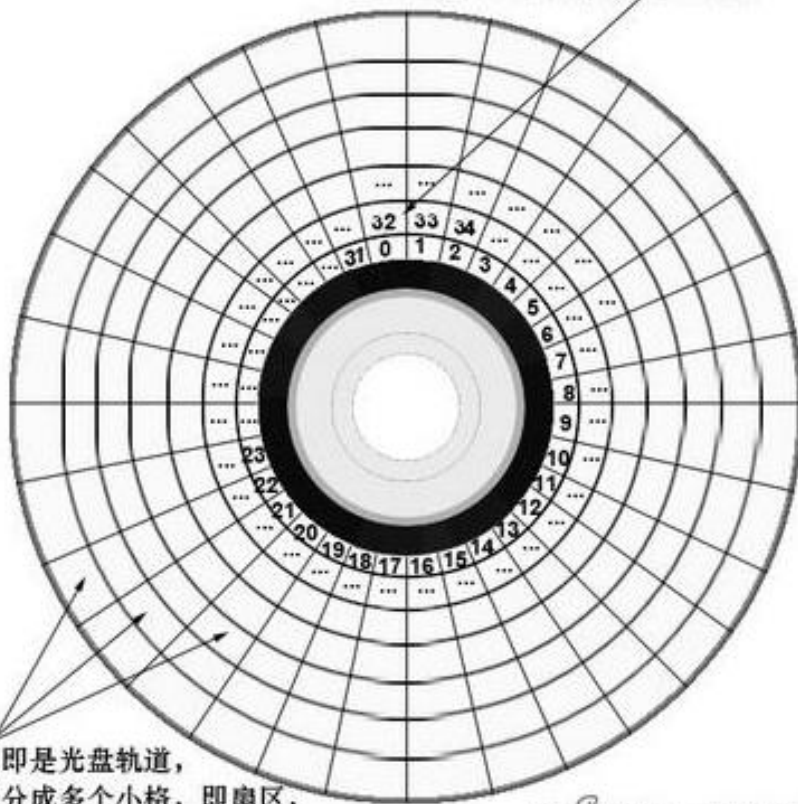


硬盘 结构

返回

光盘结构

从0扇区开始计数，依次为0扇区、1扇区、2扇区等。0扇区的第1个字节的地址，用16进制表示即0000h，接下来依次为0001h、0002h、0003h等，这即所谓的偏移地址。



一个个圆圈即是光盘轨道，
每条轨道又分成多个小格，即扇区，
每个扇区能够存储2048个字节的数据。

可移动硬盘

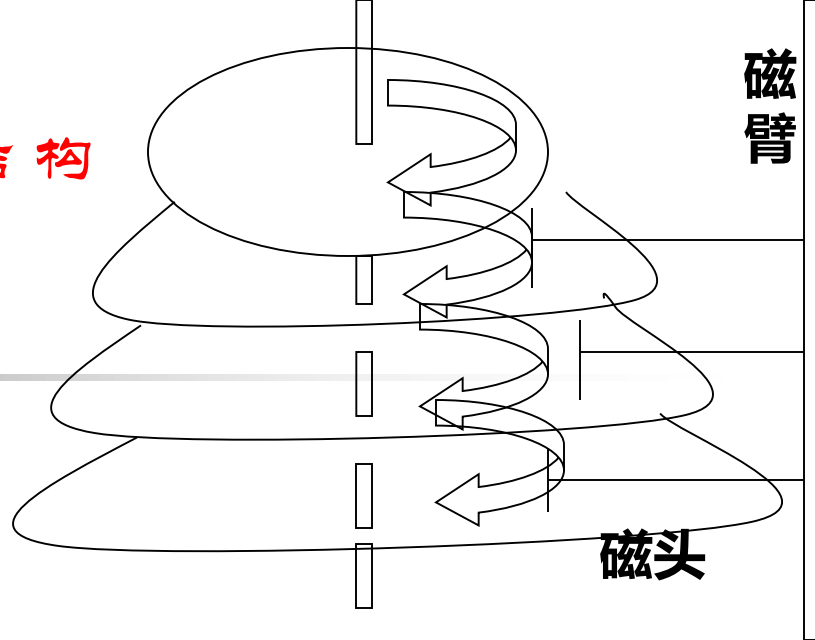


U盘



磁盘结构

磁臂



5.6.1 磁盘性能简述

❖ 数据的组织

- 磁盘结构、柱面、磁道、扇区
- **磁盘格式化**
- 磁盘物理块的地址：柱面号 磁头号 扇区号

❖ 磁盘类型：

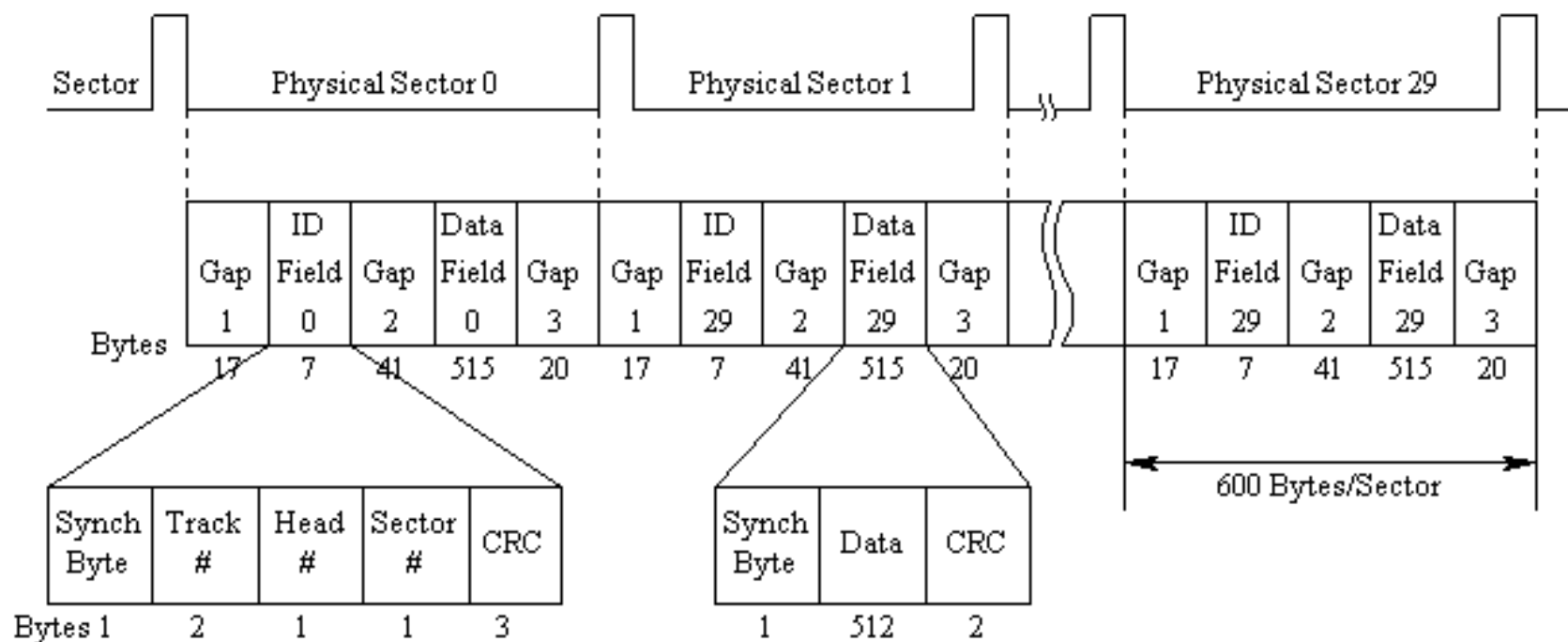
固定头磁盘（每条磁道上一个磁头）、**移动头磁盘**（每盘面上一个磁头）

❖ 磁盘访问时间 寻道时间占比例大

- **寻道时间**：将磁头从当前位置移到指定磁道所经历时间
- **旋转延迟时间**：指定扇区移动到磁头下面所经历时间
- **传输时间**：将扇区上的数据从磁盘读出/向磁盘写入数据所经历的时间。

举例：假定寻道时间和旋转时间为20ms，磁道传输率10MB/s，传输10KB数据，总访问时间为21ms。

磁盘格式化



返回

5.6.2 磁盘调度算法

注意：寻道时间



❖ 磁盘调度算法

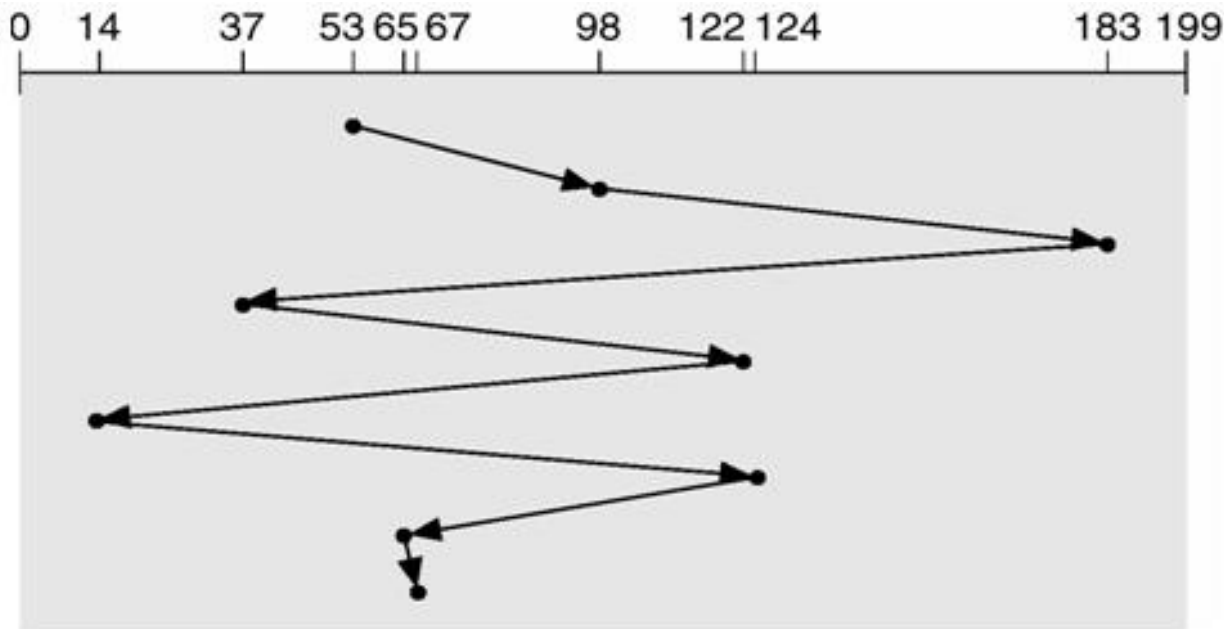
- 早期的磁盘调度算法
 - 先来先服务FCFS
 - 最短寻道时间优先SSTF
- 扫描算法
 - 扫描(SCAN)算法
 - 循环扫描(CSCAN)算法
 - ****N-STEP-SCAN调度算法**
 - ****FSCAN调度算法**

例：假设一个请求序列：98，183，37，122，14，124，65，67
磁头当前的位置在53。

FCFS 先来先服务

按进程请求访问磁盘的先后次序进行调度。

请求序列：98, 183, 37, 122, 14, 124, 65, 67
磁头当前的位置在53



下磁道	移道数
98	$98-53=45$
183	$183-98=85$
37	$183-37=146$
122	85
14	108
124	110
65	59
67	2
总道数	640
平均	80

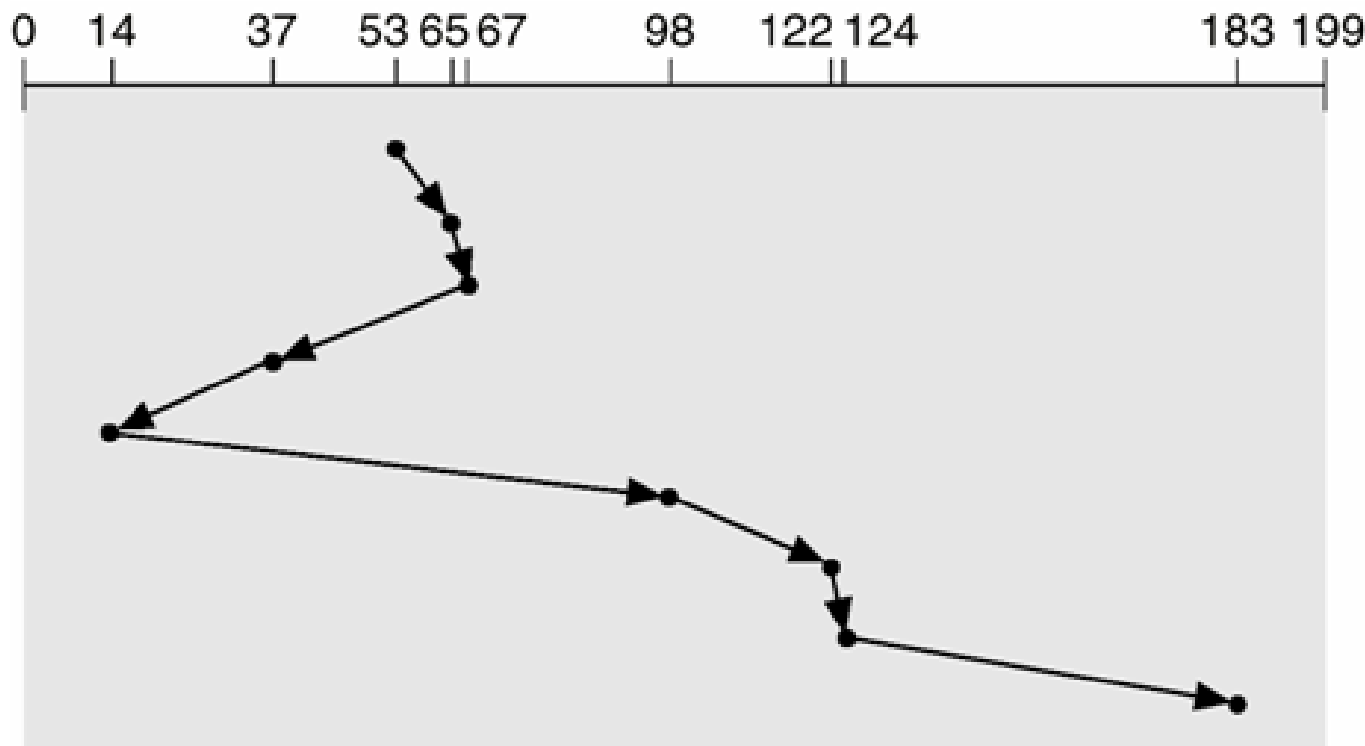
特点：简单、较合理，但未对寻道进行优化。

最短寻道时间优先 (SSTF)

选择从当前磁头位置所需寻道时间最短的请求。

请求序列：98, 183, 37, 122, 14, 124, 65, 67

磁头当前的位置在53



下磁道	移道数
65	12
67	2
37	30
14	23
98	84
122	24
124	2
183	59
总道数	236
平均	29.5



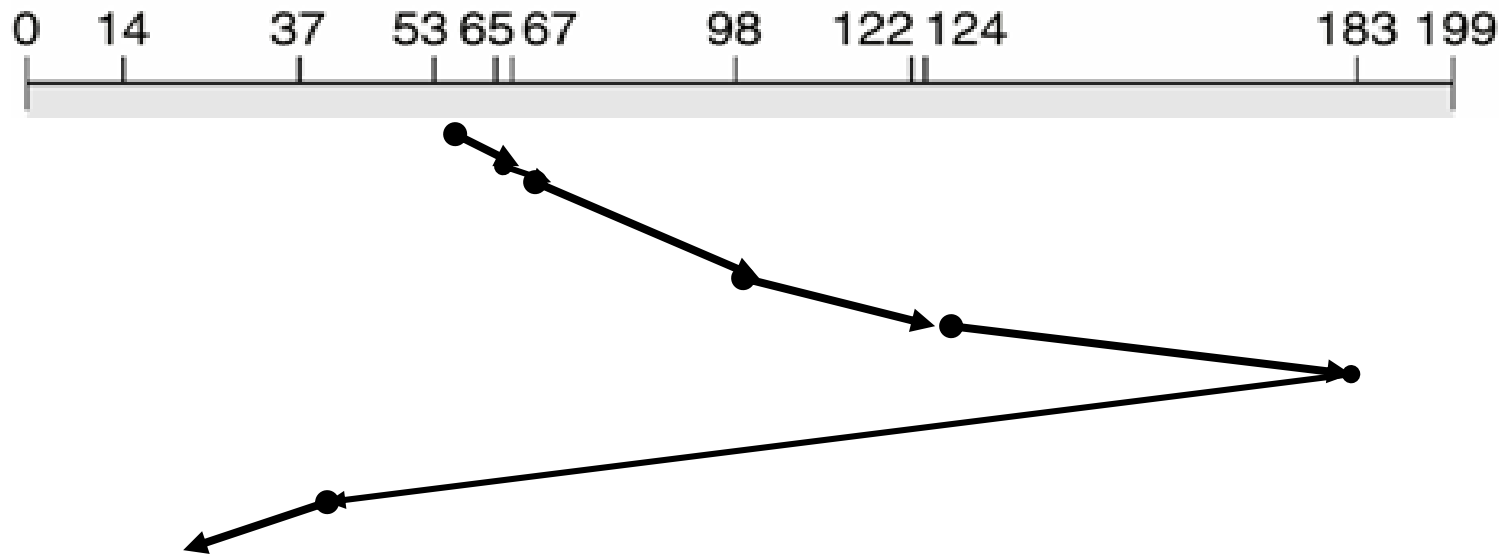
特点：寻道性能比FCFS好，有可能引起某些请求的饥饿。

扫描算法 (SCAN)

磁头从磁盘的一端开始向另一端移动，再反向移动，也称为电梯算法。

请求序列：98, 183, 37, 122, 14, 124, 65, 67

磁头当前的位置在53

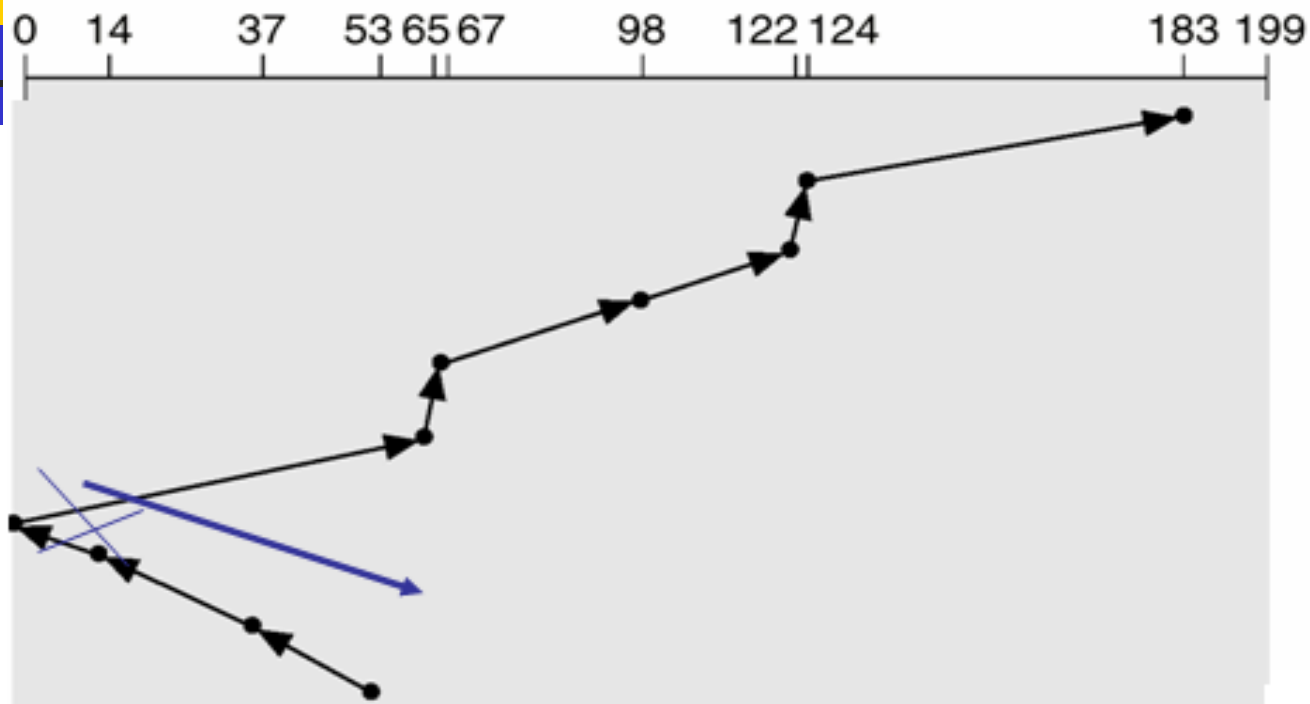


调度顺序：65, 67, 98, 122, 124, 183, 37, 14

磁头开始向磁道号增加的方向移动。

下磁道	移道数
65	12
67	2
98	31
122	24
124	2
183	59
37	146
14	23
总道数	299
平均	37.4

请求序列：98, 183, 37, 122, 14, 124, 65, 67
磁头当前的位置在53



调度顺序：37, 14, 65, 67, 98, 122, 124, 183

磁头开始向磁道号减少的方向移动

下磁道	移道数
37	16
14	23
65	51
67	2
98	31
122	24
124	2
183	59
总道数	208
平均	26

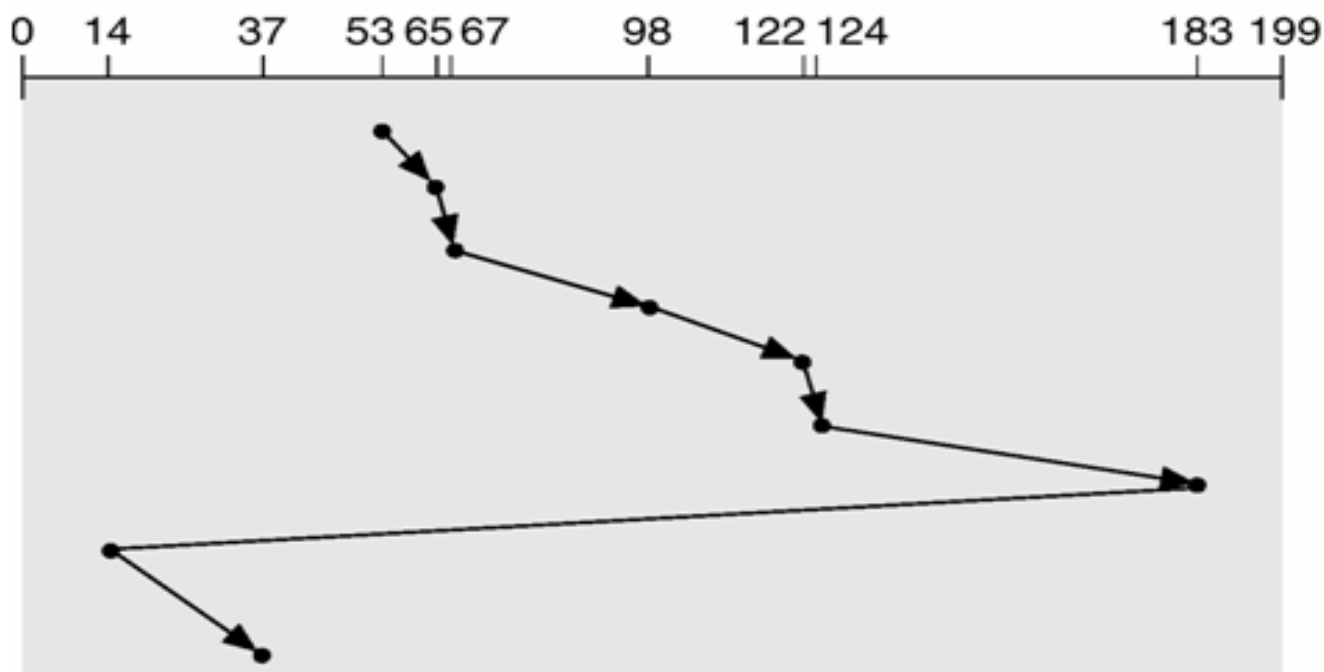
特点：寻道性能较好，避免了饥饿，但不利于远离磁头一端的访问请求。

循环扫描算法 (CSCAN)

规定磁头单向移动

请求序列：98, 183, 37, 122, 14, 124, 65, 67

磁头当前的位置在53



调度顺序：

特点：消除了对两端磁道请求的不公平。

下磁道	移道数
65	12
67	2
98	31
122	24
124	2
183	59
14	169
37	23
总道数	322
平均	40.3

*N-STEP-SCAN*调度算法

❖ SSTF、SCAN及CSCAN存在的问题---磁臂粘着



如何解决

可能出现磁臂停留在某处的情况，即反复请求某一磁道，从而垄断了整个磁盘设备，这种现象称为磁臂粘着。

❖ N-STEP-SCAN调度算法

将磁盘请求队列分成若干个长度为N的子队列，磁盘调度将按FCFS算法依次处理这些子队列，而每一子队列按SCAN算法处理。

N=1

FCFS算法

N很大

SCAN算法

N取半长度

FSCAN算法



5.6.3 设置磁盘高速缓存 (Disk Cache)

目前，由于磁盘的I/O速度远低内存的访问速度，而致使磁盘的I/O成为计算机系统的瓶颈。为提高磁盘的I/O速度，便采用磁盘高速缓存——内存的一块区域。

- 磁盘高速缓存的形式
- 数据交付方式
- 置换算法
- 周期性地写回磁盘



思考一下使用什么思想？



(1)磁盘高速缓存的形式

❖ 磁盘高速缓存

是指**内存中的一部分存储空间**，用来暂存从磁盘读出的一系列盘块中的信息。所以它是一组在**逻辑上属于磁盘**，而物理上是**驻留在内存中的盘块**。

❖ 磁盘高速缓存的形式

- 内存中单独的存储空间（大小固定）
- 未利用的存储空间_缓冲池（大小不固定）

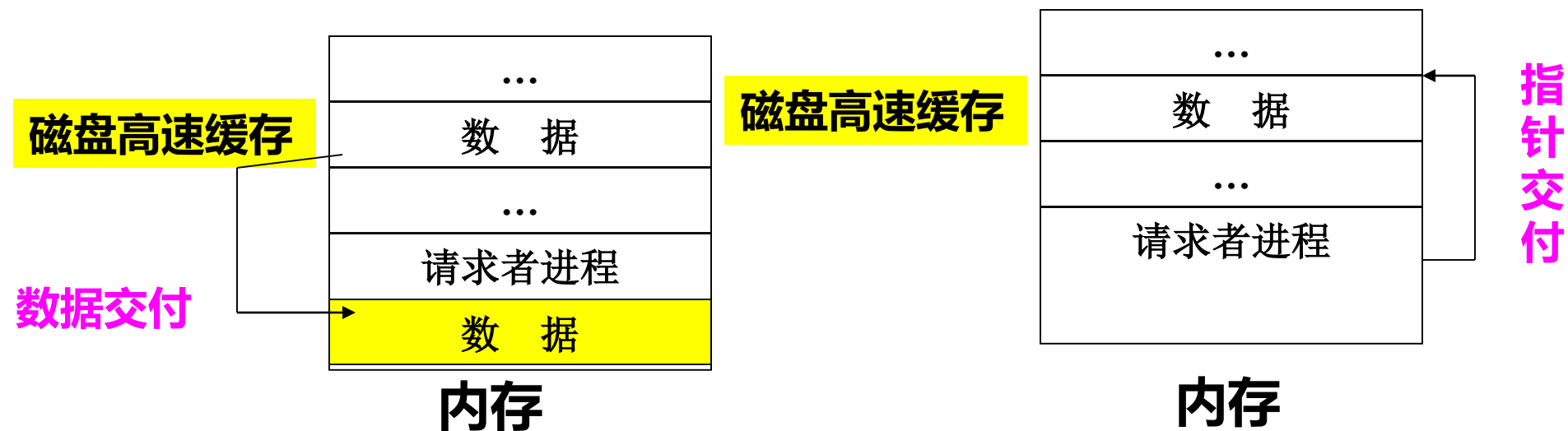


再思考一下这是什么思想？

❖ **数据交付方式**-指**磁盘高速缓存**中的数据**传送给请求者**进

数据交付 系统直接将磁盘高速缓存中的数据**传送到**请求者进程的**内存工作区**。

指针交付 只将指向磁盘高速缓存中该数据的**指针**，交付给请求者进程。



(3) 置换算法

在将磁盘中的盘块读入到磁盘高速缓存中时，若因磁盘高速缓存已满，则采用常用的算法进行置换：

- 最近最久未使用算法LRU
- 最近未使用算法NRU
- 最少使用算法LFU
- 置换时除算法外还应考虑的问题
 - 访问频率
 - 可预见性
 - **数据的一致性** __解决方法将系统中所有盘块数据，拉成一条**LRU链**，对将会严重影响到数据一致性的数据和很久都可能不再使用的盘块数据，放在LRU头部，到时优先写回磁盘。



(4) 周期性地写回磁盘

若经常访问的数据将一直保留磁盘高速缓存中，长期不会被写回磁盘，若**系统出故障**，则存在磁盘高速缓存中的**数据将丢失**。

❖ 问题解决

- **周期性**地将磁盘高速缓存中的数据**写回**磁盘。
- 磁盘高速缓存中的数据若修改，则**立即写回**磁盘。

5.6.4 提高磁盘I/O速度的其它方法

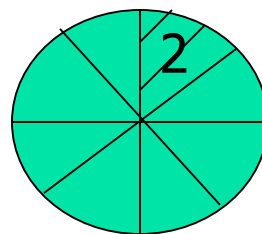
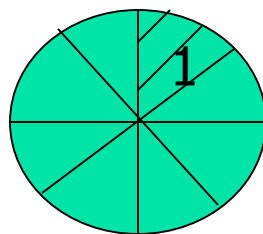
- ❖ **提前读 (Read_Ahead)** : 常用**顺序方式** , 在读当前块时 , 可采用**预先读方式**
- ❖ **延迟写** : 在缓存中的数据,本应立即写回磁盘,考虑不久之后可能会再用,故**不立即写回磁盘**。
- ❖ **优化物理块的分布** : 使磁头移动的距离最小 (优化物理块的分布、优化索引结点的分布) 。 **板书举例**
- ❖ **虚拟盘** : 利用**内存**去仿真磁盘 , 又称为**RAM盘**。(与磁盘高速缓存的区别 : RAM盘中的内容由用户控制 , 而缓存中的内容则由OS控制)



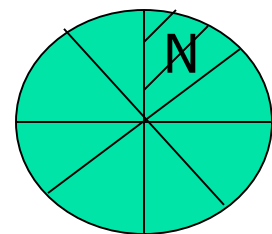
5.6.5 廉价磁盘冗余阵列 (RAID)

1987年由美国加利福尼亚大学伯莱分校提出的，1988年问世
即利用一台磁盘阵列控制器，来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。图

- 并行交叉存取
- RAID的分级
- RAID的优点



...



(1) 并行交叉存取

在一个配置多台磁盘驱动器的系统中，如P179页图5-27所示。

- 系统将每一盘块的数据分成若干个子盘块数据
- 再把每一个子盘块的数据分别存储到各个不同磁盘中的相同位置上，
- 当要将一个盘块的数据传送到内存时，采取并行传输方式，将该盘块中的各个子盘块数据同时向内存传输，从而使传输时间大大减少。

跳转



（2）廉价磁盘冗余阵列（RAID）的分级—略

- ❖ RAID 0级 本级仅提供了并行存取技术，无冗余校验功能，至使磁盘系统的可靠性不好，故较少使用。
- ❖ RAID 1级 本级具有磁盘锁像功能，即每次访问磁盘时，采用并行技术将数据同时主盘（数据盘）和磁盘镜像盘中。磁盘系统的可靠性好，但磁盘利用率不高。
- ❖ RAID 3级 本级采用并行存取技术，增加了冗余校验功能，即用一个盘作校验盘，常用科学计算和图像处理。
- ❖ RAID 5级 本级具有独立传送功能，各磁盘驱动器可独立读、写，校验信息在所有盘上，常用于I/O较频繁的事务处理中。
- ❖ RAID 6级 本级设置了一个专用的、可快速访问的异步校验盘，该盘具有独立的数据访问通路。
- ❖ RAID 7级 本级是对RAID 6级的改进，所有磁盘均具有较高的传输速率和优异的性能，是目前最高档次的磁盘阵列。



(3) 廉价磁盘冗余阵列 (RAID) 的优点

- ❖ 可靠性高

是RAID 的最大优点 (除RAID 0 级外)

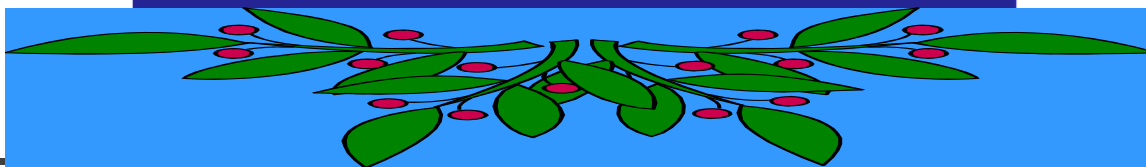
- ❖ 磁盘I/O速度高

由于RAID 可采用并行交叉存取方式, 故提高了磁盘I/O速度。

- ❖ 性能/价格比高

用RAID技术实现大容量存储器时, 与大型磁盘系统相比, 其体积和价格只是它的1/3, 且可靠性高。

本章小结



- ❖ 1. I/O系统、I/O控制方式（理解）--I/O通道
- ❖ 2. 缓冲管理（重点掌握）--多缓冲的数据结构和使用过程
- ❖ 3. 设备分配（理解）--设备无关性
- ❖ 4. SPOOLING技术（重点掌握）--概念、原理
- ❖ 5. 磁盘调度（重点）--性能、算法、磁盘高速缓冲



本章作业

1、设某磁盘有200个柱面，编号为0, 1, 2, ..., 199，磁头刚从140道移到143道完成了读写。若某时刻有9个磁盘请求分别对如下各道进行读写：

86, 147, 91, 177, 94, 150, 102, 175, 130

试分别求FCFS、SSTF及SCAN磁盘调度算法响应请求的次序及磁头移动的总距离。——板书

习题

一、 选择题

- 1、使用户所编写的程序与实际使用的物理设备无关，这是由设备管理的（ ）功能实现的。
A . 设备独立性 B . 设备分配 C . 缓冲管理 D . 虚拟设备
- 2、数据传送方式的作用是在外围设备和内存之间开辟直接的数据交换通道。（ ）
A . 程序直接控制 B . DMA C . 通道控制 D . 中断
- 3、通道是一种（ ）
A. 保存I/O信息的部件 B. 传输信息的电子线路 C. 通用处理器 D. 专用处理器
- 4、CPU对通道的请求形式是（ ）
B. 中断 C. 通道命令 D. 转移指令
- 5、通道对 CPU的请求形式是（ ）
B. 中断 C. 通道命令 D. 跳转指令
- 6、环形缓冲区是一种（ ）
A. 单缓冲区 B. 双缓冲区 C. 多缓冲区 D. 缓冲池
- 7、系统利用 SPooling技术实现（ ）
A. 对换手段 B. 虚拟设备 C. 系统调用 D. 虚拟存储

8、在配有操作系统的计算机中，用户程序通过（ ）向操作系统提出使用外部设备的要求。

A. 作业申请 B. 原语 C. 系统调用 D. I/O指令

9、CPU与通道可以并行执行，并通过（ ）实现彼此间的通讯和同步。

A. I/O指令 B. I/O中断 C. I/O指令和I/O中断 D. 操作员

10、（ ）是直接存储设备。

A. 磁盘 B. 磁带 C. 打印机 D. 键盘显示终端

11、下列叙述，正确的一条是（ ）

A. 在设备I/O中引入缓冲技术的目的是为了节省内存

B. 指令中的地址结构和外存容量是决定虚存作业地址空间的两个因素

C. 处于阻塞状态的进程被唤醒后，可直接进入运行状态

D. 在请求页式管理中，FIFO置换算法的内存利用率是较高的

12、系统中，象键盘、终端、打印机等以字符为单位组织和处理信息的设备称为（ ）

A. 字符设备 B. 块设备 C. 虚拟设备 D. 独享设备

答案：ABDCB CBCCA BA



二、填空题

1、缓冲区的设置可分为_____、_____、
_____和_____。

2、利用缓冲区能有效地缓和_____和_____之间速度不匹配地矛盾，
虚拟设备功能是为使_____变成能被多个进程同时使用的_____。

3、从资源分配的角度看，可以把设备分为独占设备和共享设备。
打印机属于_____设备，而磁盘属于
_____设备。

4、虚拟设备是通过_____技术把_____设备变成能为若干
用户_____的设备。

5、通道是一个独立于_____的专管的处理机，它控制
_____与内存之间的信息交换。

答案：1、单缓冲、双缓冲、多缓冲、缓冲池2、外围设备、
处理机；一个物理设备；逻辑设备3、独占；共享4、
SPOOLing；独占；共享5、CPU；外围设备



三、问答题

- 1、为什么要设置I/O缓冲区？，通常有哪几类缓冲区？
- 2、如何将独占型输入设备改造成可共享使用的虚拟设备？
- 3、在设备管理中，何谓设备独立性？如何实现设备独立性？
- 4、为什么要引入SPOOLING技术？SPOOLING技术可带来哪些好处？
- 5、SPOOLing技术的原理和组成？SPOOLing技术如何使一台打印机虚拟成多台打印机？
- 6、数据传送方式有哪几种？
- 7、什么是通道？试画出通道控制方式时的CPU、通道和设备的工作流程图。
- 8、UNIX系统中将设备分为块设备和字符设备，它们各有什么特点？
- 9、什么叫通道技术？通道的作用是什么？



The End