



操作系统

Operating System

北京理工大学计算机学院

马 锐

Email: mary@bit.edu.cn



版权声明

- 本内容版权归北京理工大学计算机学院
操作系统课程组马锐所有
- 使用者可以将全部或部分本内容免费用
于非商业用途
- 使用者在使用全部或部分本内容时请注
明来源
 - 内容来自：北京理工大学计算机学院 +
马锐 + 材料名字
- 对于不遵守此声明或其他违法使用本内
容者，将依法保留追究权

第6章 设备管理

6.1 I/O硬件组成

6.2 I/O软件组成

6.3 磁盘管理

3

6.1 I/O硬件组成(1)

➤ I/O设备

● 数据传输速率

◆ 低速设备：键盘、鼠标

◆ 中速设备：打印机

◆ 高速设备：磁盘、光盘

● 设备共享属性

◆ 独占设备：慢速的字符设备

◆ 共享设备：快速的块设备，资源利用率高

◆ 虚拟设备：

4

6.1 I/O硬件组成(2)

- ◆ 将独占设备改造为共享设备(SPOOLing技术)
- ◆ 将低速设备改造为高速设备(虚拟磁盘)
- 数据传输单位
 - ◆ 块设备
 - 以块为单位传输信息
 - 传输速率较高、可寻址
 - 磁盘、磁带
 - ◆ 字符设备

5

6.1 I/O硬件组成(3)

- 以字符为单位传输信息
- 传输速率较低、不可寻址
- 鼠标、键盘、打印机
- 网络通信设备
- 设备控制器
 - I/O设备由机械和电子两部分组成
 - 机械部分是设备本身
 - 电子部分叫做设备控制器或适配器
 - 功能
 - 控制一个或多个I/O设备，以实现I/O设备和计算机之间的数据交换

6

6.1 I/O硬件组成(4)

- ◆ 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
 - 接收和识别命令
 - 数据交换
 - 标识和报告设备的状态
 - 地址识别
 - 数据缓冲
 - 差错控制

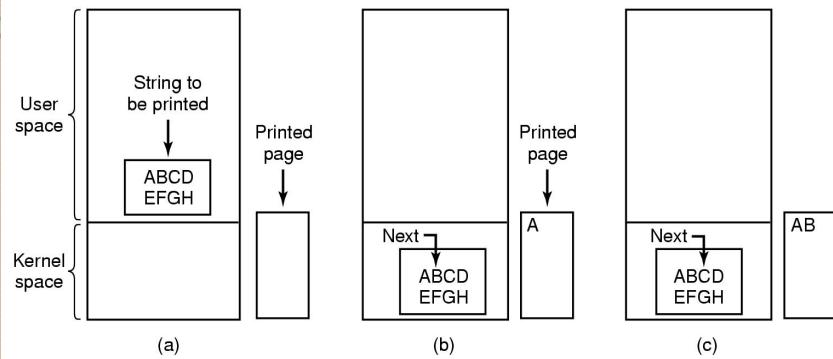
7

6.1 I/O硬件组成(5)

- I/O数据传输的控制方式
 - 程序查询方式
 - ◆ 循环查询
 - ◆ CPU与设备完全串行
 - 程序中断方式
 - ◆ 尽量减少主机对I/O控制的干预，将主机从烦杂的I/O控制事务中解脱
 - 直接存储器访问(DMA)
 - 通道方式

8

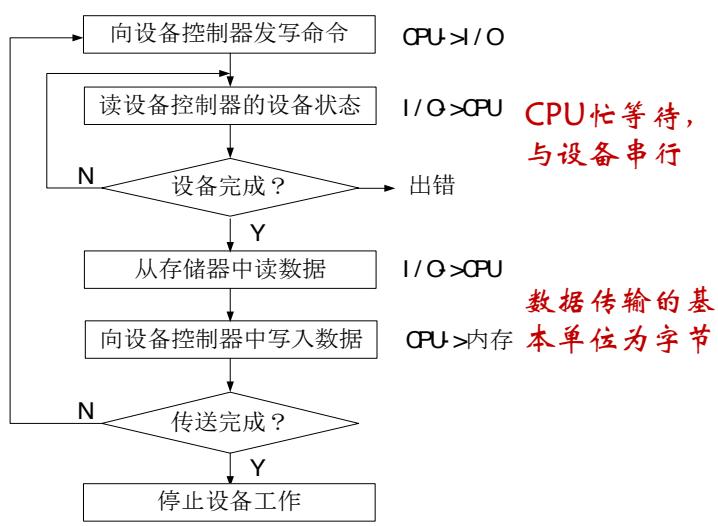
6.1 I/O硬件组成(6)



程序查询方式

9

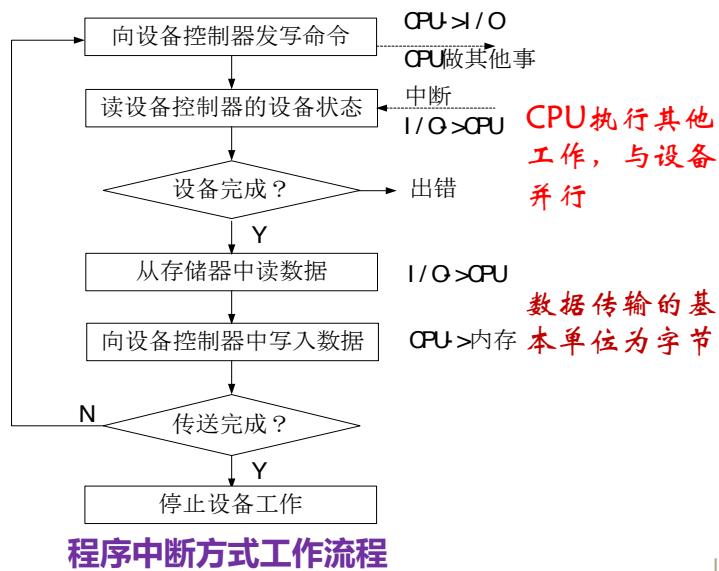
6.1 I/O硬件组成(7)



程序查询方式工作流程

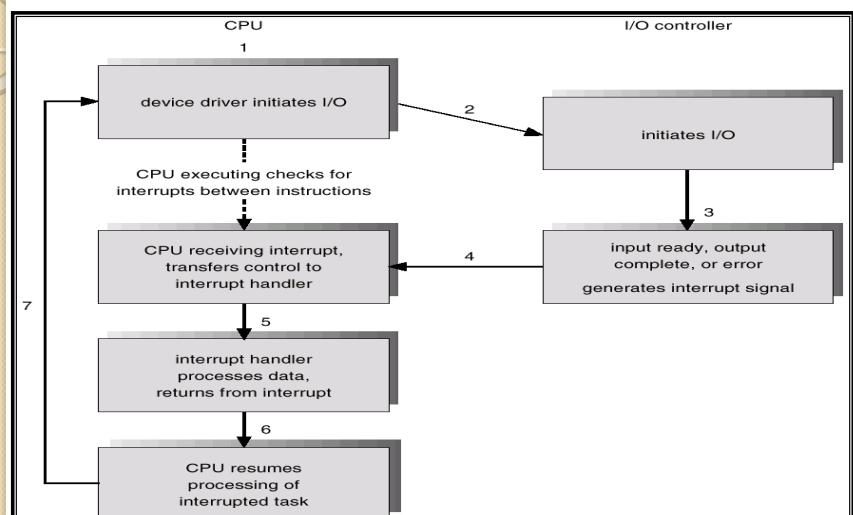
10

6.1 I/O硬件组成(8)



11

6.1 I/O硬件组成(9)



12

6.1 I/O硬件组成(10)

• 直接存储器访问(DMA)

◆ 从以字节为单位扩展到以数据块为单位

◆ 特点

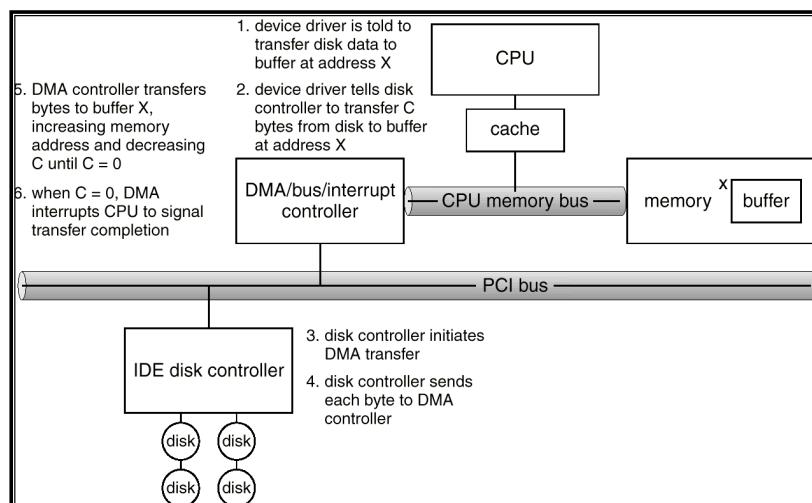
- 数据传输的基本单位是数据块
- 传送数据从设备直接送入内存或相反
- 仅在传送一个或多个数据块的开始和结束时，才需要CPU干预，整块数据的传送是在控制器的控制下完成的

◆ 实现

- 磁盘地址
- 主存的起始地址
- 传送的字节数

13

6.1 I/O硬件组成(11)



DMA访问步骤

14

6.1 I/O硬件组成(12)

➤ 通道方式

- 引入

- ◆ 进一步减少CPU的干预，将对一个数据块的干预减少为对一组数据块及其相关控制和管理的干预
- ◆ 实现CPU、通道和I/O设备三者的并行

- 实现方法

- ◆ 接受CPU的委托，独立地执行自己的通道程序，管理和控制输入输出设备，实现外围设备与主存储器之间的成批数据传送。当CPU委托的I/O任务完成后，通道发出中断信号，请求CPU处理

15

6.1 I/O硬件组成(13)

- 分类（信息交换方式）

- ◆ 字节多路通道

- 以字节为单位传输信息，可以分时执行多个通道程序。当一个通道程序控制某台设备传送一个字节后，通道硬件就转去执行另一个通道程序，控制另一台设备传送一个字节的信息

- 主要用来连接大量慢速的设备

- ◆ 选择通道

- 以成组方式工作，每次传送一批数据

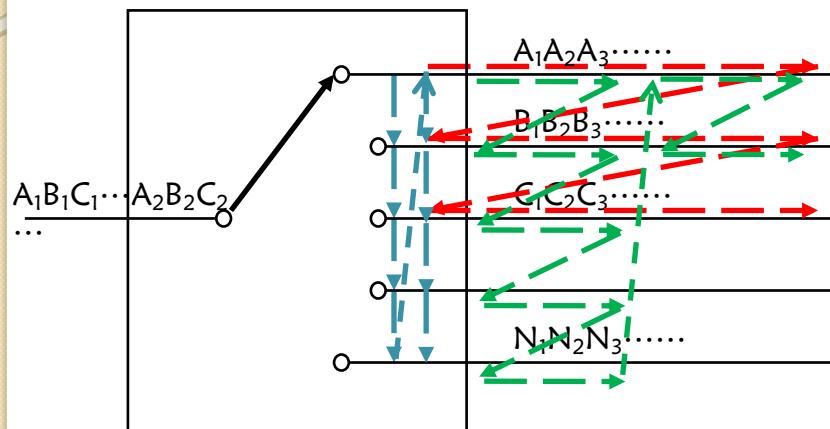
16

6.1 I/O硬件组成(14)

- 一段时间内只能执行一个通道程序，控制一台设备进行数据传输。当这台设备数据传输完成后，再选择与通道连接的另一台设备，执行它的相应的通道程序
- 常连接高速设备
- 数组多路通道
 - 结合了选择通道传送速度高和字节多路通道能进行分时并行操作的优点
 - 它先为一台设备执行一条通道指令，然后自动转接，为另一台设备执行一条通道指令

17

6.1 I/O硬件组成(15)



18

6.2 I/O软件组成

6.2.1 I/O软件的目标

6.2.2 I/O软件的组成

6.2.3 同步I/O和异步I/O

19

6.2.1 I/O软件的目标(1)

- 提供设备的独立性（设备无关性）
 - 应用程序独立于具体使用的物理设备
 - ◆ 独立于设备的类型
 - ◆ 独立于同类设备的具体台号
 - 设备的统一命名
 - ◆ 设备名不应依赖于设备
 - ◆ 在OS中，通常规定用户程序中不直接使用物理设备名（或设备的物理地址），而使用逻辑设备名

20

6.2.1 I/O软件的目标(2)

- ◆ 逻辑设备表示物理设备属性，它不指某个具体设备，而是对应一类设备
- ◆ 由操作系统根据系统设备情况完成相应的映射
- 逻辑设备表

| 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 |
| : | : | : |

(a)

| 逻辑设备名 | 系统设备表指针 |
|--------------|---------|
| /dev/tty | 3 |
| /dev/printer | 5 |
| : | |

(b) 21

6.2.1 I/O软件的目标(3)

- ◆ 优点
 - 设备分配灵活
 - 易于实现I/O重定向
- 设备独立软件
- 出错处理
 - 对于数据传输中的错误应尽可能地在接近硬件层上处理
 - 仅当低层软件无能为力时，才将错误上交高层处理

6.2.1 I/O软件的目标(4)

➤ 缓冲技术

- 使数据的到达率与离去率相匹配，提高系统的吞吐率
 - ◆ 改善I/O设备和CPU之间速度不匹配的情况
 - ◆ 减少中断CPU的次数，提高CPU利用率
 - ◆ 减少启动设备的次数，延长设备的寿命
 - ◆ 提高CPU和I/O设备之间的并行性

23

6.2.1 I/O软件的目标(5)

➤ 设备分配

- 设备类型
 - ◆ 独占设备：静态分配
 - ◆ 共享设备：动态分配
 - ◆ 虚拟设备：共享设备，FCFS
- 静态分配
 - ◆ 简单，但设备利用率低
- 动态分配
 - ◆ 设备利用率高，但容易引起死锁

24

6.2.2 I/O软件的组成(1)

➤ I/O软件的结构

- 基本思想

- ◆ 按分层思想构成
- ◆ 较低层的软件要使较高层的软件独立于硬件的特性
- ◆ 较高层软件要向用户提供一个友好的、清晰的、简单的、功能更强的接口

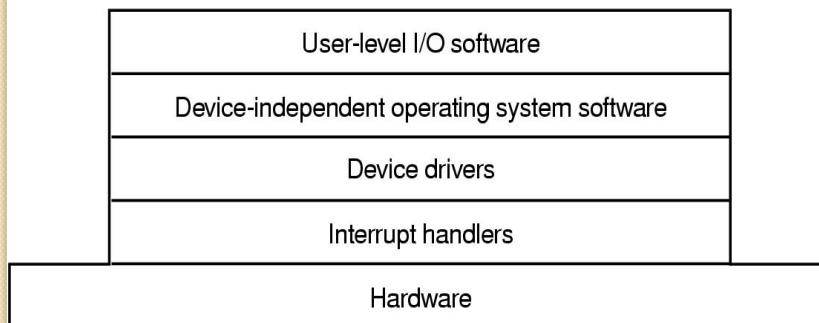
- 层次结构

- ◆ 中断处理程序
- ◆ 设备驱动程序

25

6.2.2 I/O软件的组成(2)

- ◆ 独立于设备的软件
- ◆ 用户空间的I/O软件



26

6.2.2 I/O软件的组成(3)

➤ 中断处理程序

- 中断通常隐藏在操作系统内
 - ◆ 每个进程在启动一个I/O操作后阻塞起来，直到I/O操作完成并产生一个中断，通过中断向CPU报告，CPU唤醒该进程，对设备进行中断处理
- 中断处理工作完全由OS完成，用户进程根本不知道中断的产生和处理过程
- 中断处理工作
 - ◆ 唤醒被阻塞的驱动程序进程

27

6.2.2 I/O软件的组成(4)

- ◆ 保护被中断进程的CPU环境
- ◆ 分析中断原因转入相应中断处理程序
- ◆ 中断处理
 - 检查设备状态寄存器的内容，看它是否正常完成
 - 正常完成则结束中断；若还有等待传输的I/O请求，则启动下一个请求
 - 若传输有错，判断是否允许重复传输，若允许则发启动传输命令，否则向上层报告“设备错误”的信息
- ◆ 恢复被中断进程的现场

28

6.2.2 I/O软件的组成(5)

➤ 设备驱动程序

- 与设备密切相关的代码放在设备驱动程序中，每个设备驱动程序处理一种设备类型
- 接收来自与设备无关的上层软件的抽象请求，并执行这个请求
- 在请求I/O的进程与设备控制器之间的一个通信和转换程序，它将进程的I/O请求经过转换后，传递给控制器，又把控制器中所记录的设备状态和I/O操作完成情况及时地反映给请求I/O的进程

29

6.2.2 I/O软件的组成(6)

● 设备驱动程序的处理过程

- ◆ 接收发出I/O请求的进程发来的命令和参数，并将命令中的抽象要求转换为具体要求
- ◆ 检查用户I/O请求的合法性
- ◆ 读出和检查设备的状态
 - 如果设备空闲则立即启动I/O设备完成指定的I/O操作
 - 如果设备处于忙碌状态，则将请求者的请求块挂在设备队列上等待
- ◆ 传送必要的参数并设置设备工作方式

30

6.2.2 I/O软件的组成(7)

- ◆ 启动I/O设备
- ◆ 阻塞自己，直至中断到来时被唤醒，并根据中断类型调用相应的中断处理程序进行处理
- 设备处理方式
 - ◆ 为每一类设备设置一个I/O进程
 - ◆ 在整个系统中设置一个I/O进程
 - ◆ 不设置专门的设备处理进程，只为各类设备设置相应的设备处理（驱动）程序，供用户进程或系统进程调用

31

6.2.2 I/O软件的组成(8)

- 设备独立的软件
- 基本任务
 - ◆ 实现所有设备都需要的功能，并向用户级软件提供一个统一的接口

| |
|--------------|
| 设备命名 |
| 设备保护 |
| 提供与设备无关的块尺寸 |
| 缓冲技术 |
| 块设备的存储分配 |
| 独占设备的分配与释放 |
| 报告错误信息 |
| 与设备驱动程序的统一接口 |

32

6.2.2 I/O软件的组成(9)

- 设备命名

- ◆ 将设备的逻辑名映射为物理设备名，也即把设备的符号名映射到正确的设备驱动程序上

- 设备保护

- ◆ 防止无权存取设备的用户存取设备，保证有权使用的用户正确使用设备
 - ◆ 为每一个设备设置正确的存取权
 - ◆ 禁止用户直接访问设备（通过系统调用访问）

33

6.2.2 I/O软件的组成(10)

- 提供与设备无关的块尺寸

- ◆ 向较高层软件掩盖不同磁盘采用不同扇区尺寸的事实并提供大小统一的块尺寸
 - ◆ 较高层的软件只与抽象设备打交道，使用等长的逻辑块，独立于物理扇区尺寸

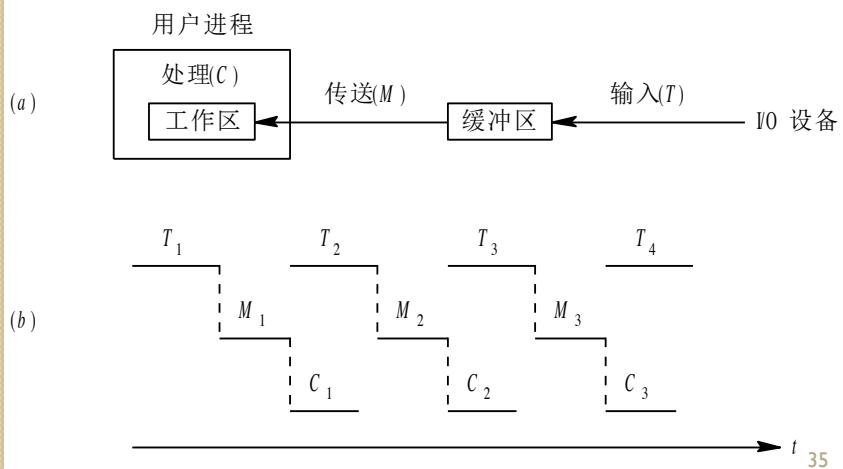
- 缓冲技术

- ◆ 对设备缓冲区进行有效管理，提高I/O效率

34

6.2.2 I/O软件的组成(11)

◆ 单缓冲



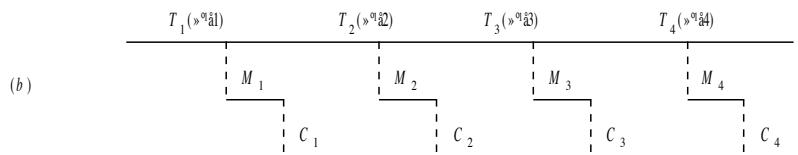
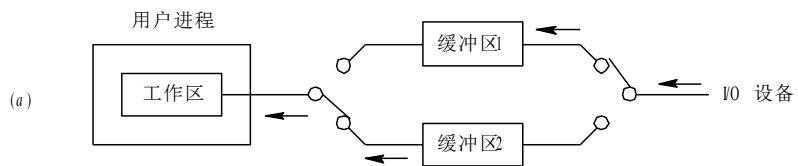
6.2.2 I/O软件的组成(12)

- 假定从磁盘将一块数据输入到缓冲区的时间为T，OS将缓冲区的数据传送到用户区的时间为M，CPU对一块数据处理的时间为C，则
- 系统对每块数据的处理时间为 $\text{Max}(C, T) + M$

◆ 双缓冲

- 系统对每块数据的处理时间为 $\text{Max}(C+M, T)$

6.2.2 I/O软件的组成(13)



37

6.2.2 I/O软件的组成(14)

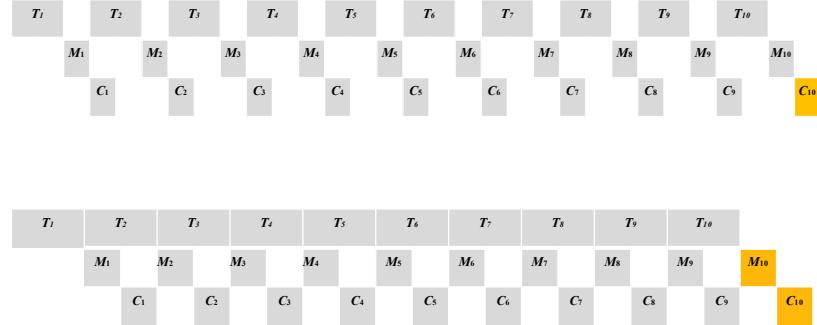
➤ **示例：**某文件占10个磁盘块，现要把该文件磁盘块逐个读入主存缓冲区，并送用户区进行分析。假设一个缓冲区与一个磁盘块大小相同，把一个磁盘块读入缓冲区的时间为 $100\ \mu s$ ，将缓冲区的数据传送到用户区的时间是 $50\ \mu s$ ，CPU对一块数据进行分析的时间为 $50\ \mu s$ 。在单缓冲区和双缓冲区结构下，读入并分析完该文件的时间分别是

- A. $1500\ \mu s, 1000\ \mu s$ B. $1550\ \mu s, 1100\ \mu s$
C. $1550\ \mu s, 1550\ \mu s$ D. $2000\ \mu s, 2000\ \mu s$

答案：B

38

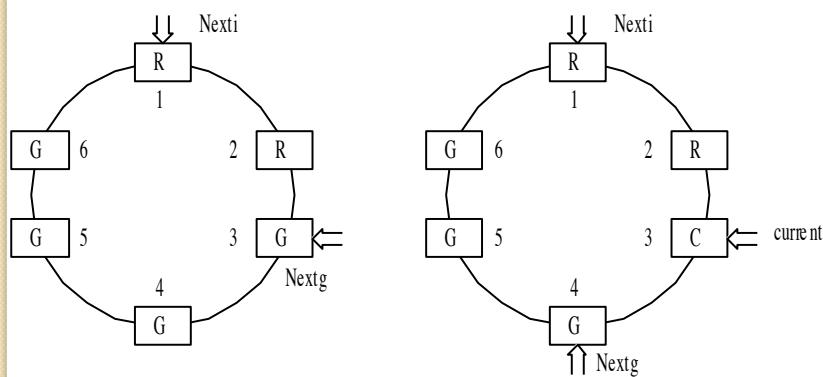
6.2.2 I/O软件的组成(14)



39

6.2.2 I/O软件的组成(15)

◆ 循环缓冲



40

6.2.2 I/O软件的组成(16)

- ◆ 缓冲池
 - 空缓冲区(队列)
 - 输入缓冲区(队列)
 - 输出缓冲区(队列)
 - 队列操作
 - 缓冲队列是临界资源
- 设备分配
 - ◆ 分配程序

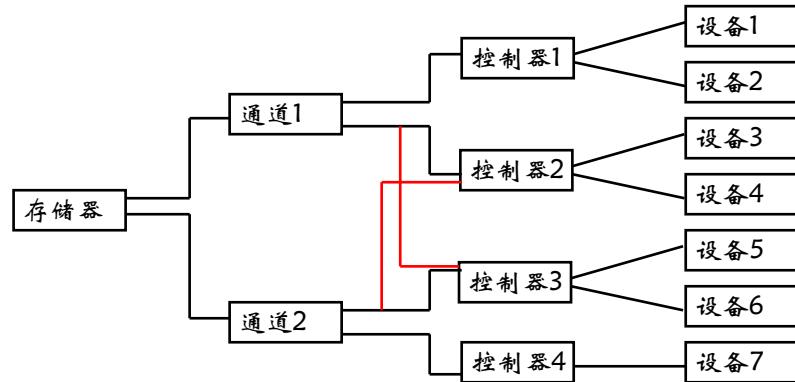
41

6.2.2 I/O软件的组成(17)

- 分配设备
- 分配控制器
- 分配通道
- ◆ 数据结构
 - 设备控制表
 - 系统设备表
 - 控制器控制表
 - 通道控制表

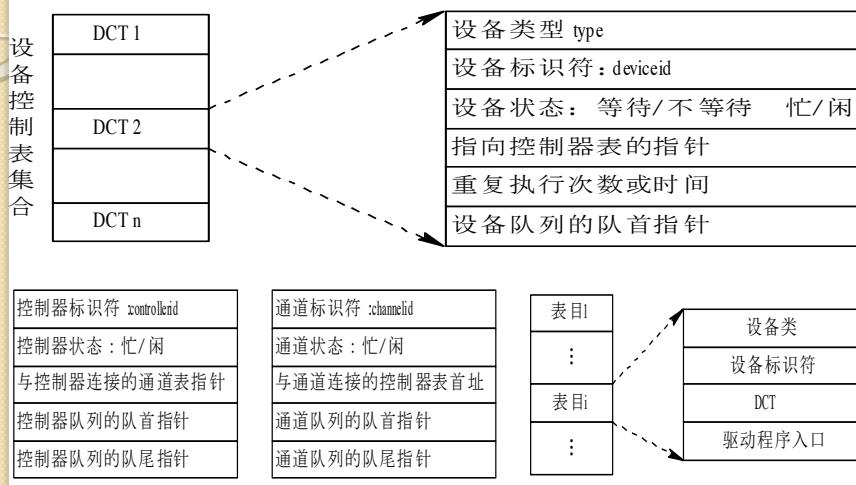
42

6.2.2 I/O软件的组成(18)



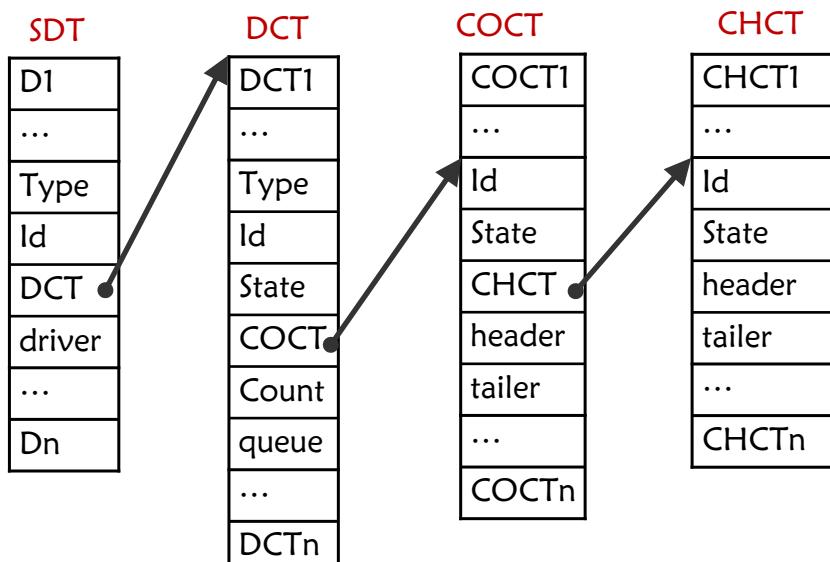
43

6.2.2 I/O软件的组成(19)



44

6.2.2 I/O软件的组成(20)



45

6.2.2 I/O软件的组成(21)

- ◆ 考虑因素
 - 设备固有属性：独占，共享，虚拟
 - 设备分配算法：FCFS, Priority
 - 设备分配的安全性
- 出错处理
 - ◆ 一般由设备驱动程序实现

46

6.2.2 I/O软件的组成(22)

➢ 用户空间的I/O软件

- 系统调用，包括I/O系统调用，通常由库过程实现
 - ◆ 这些过程所做的工作只是将系统调用时所用的参数放在适当的位置，实际由系统的I/O过程实现真正的操作
- SPOOLing(Simultaneous Peripheral Operating On-Line)系统，假脱机
 - ◆ 联机情况下实现的同时外围操作

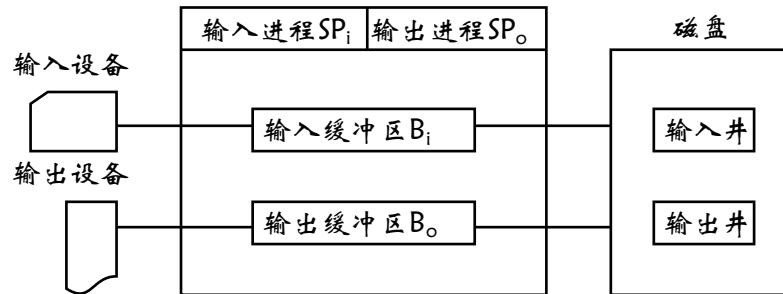
47

6.2.2 I/O软件的组成(23)

- ◆ 将独占设备改造成可共享设备
 - 提高了设备利用率
 - 加快了进程的执行速度
- 构成
 - 输入井和输出井
 - 输入缓冲区和输出缓冲区
 - 输入进程 SP_i 和输出进程 Sp_o

48

6.2.2 I/O软件的组成(24)



49

6.2.2 I/O软件的组成(25)

- ◆ 工作原理
- ◆ 井管理程序

- 负责管理输入和输出缓冲区，记录每个缓冲区的作用
- 当作业执行过程中要求启动某台设备输入或输出时，操作系统截获该请求并调出井管理程序，控制从相应的输入井读或向共享设备的输出井写。输入井和输出井上的信息是以文件的形式记录的

50

6.2.2 I/O软件的组成(26)

◆ 预输入程序

- 负责将输入设备上的信息预先输入到可共享设备的缓冲区域(输入井)并将存放信息的位置记录下来待用

◆ 缓输出程序

- 负责从磁盘缓冲区读信息向输出设备输出。当设备空闲或进程完成后，系统负责调用缓输出程序将各用户的信息依次从输出井送设备输出

51

6.2.2 I/O软件的组成(27)

➤ 小结

- 用户进程层执行输入输出系统调用，对I/O数据进行格式化，为假脱机输入/输出作准备
- 独立于设备的软件实现设备的命名、设备的保护、成块处理、缓冲技术和设备分配
- 设备驱动程序设置设备寄存器、检查设备的执行状态
- 中断处理程序负责I/O完成时进行中断处理，唤醒设备驱动程序进程
- 硬件层实现物理I/O的操作

52

6.2.2 I/O软件的组成(28)

➤ 示例

- 用户进程读取一个文件中的一块信息
- 处理过程
 - ◆ 用户进程发出一个读文件的系统调用
 - ◆ 设备独立软件
 - 检查参数的正确性，若正确，检查内存缓存中有无要读的信息块
 - 有，从缓冲区中直接将信息返回用户
 - 无，执行物理I/O
 - 独立于设备的I/O软件将设备的逻辑名转换成物理名，检查设备操作权限

53

6.2.2 I/O软件的组成(29)

- 将I/O请求排队，用户进程阻塞等待磁盘操作的完成
- 调用设备驱动程序，向I/O硬件泄放一个读请求
 - 分配缓冲区，准备接收数据，并向设备控制寄存器发启动读命令
 - 设备控制器控制设备，执行数据传输
 - 当磁盘将所需块读入缓冲区时，硬件产生一个中断
 - 系统响应中断后，转入中断处理程序

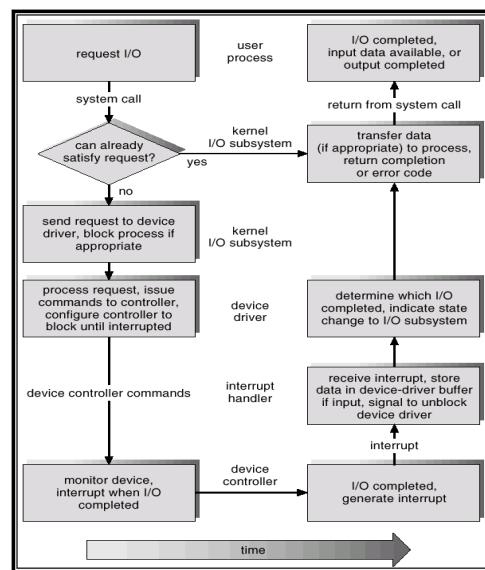
54

6.2.2 I/O软件的组成(30)

- ◆ 中断处理程序从设备获取所需的状态信息，检查中断的原因
- 若正常完成，将数据传输给指定的用户进程空间，唤醒等待该I/O完成的进程，将其放入就绪队列，等待调度
- 若出错，则向设备驱动程序发信号，若可重试，则再启动设备重传一次；否则，向上报告错误
- ◆ 用户进程继续运行

55

6.2.2 I/O软件的组成(31)



56

6.2.2 同步I/O和异步I/O

- 同步I/O：进程发出I/O请求后阻塞等待，直到数据传输完成后被唤醒，访问被传输的数据
- 异步I/O：允许进程发出I/O请求后继续运行
 - I/O完成后的通知方式：设置进程地址空间内的某个变量；通过触发信号或软件中断；进程执行流之外的某个回调函数
- 对于不必进行缓冲读写的快速I/O，使用同步I/O更有效；对于需要很长时间的I/O操作，可使用异步I/O

57

6.3 磁盘管理

- 6.3.1 磁盘结构
- 6.3.2 磁盘调度
- 6.3.3 磁盘的错误处理

58

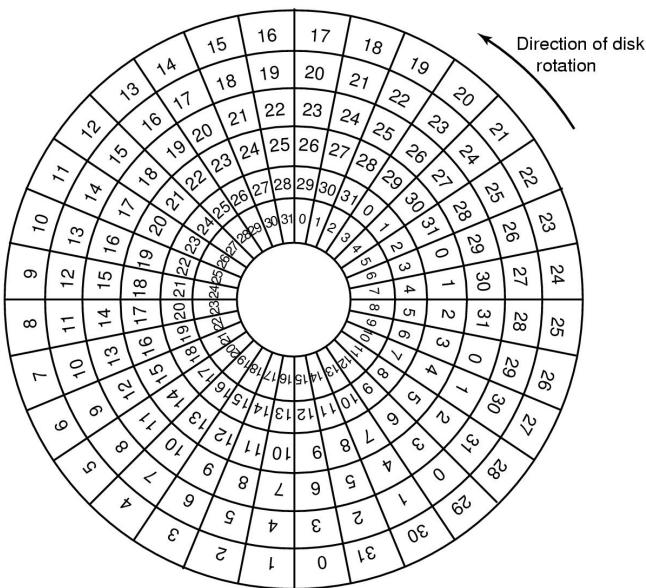
6.3.1 磁盘结构(1)

➤ HDD的物理结构

- 磁盘由若干盘片组成，每片分为两个表面，表面上覆盖着磁性材料，用于记录信息
- 磁盘表面又划分为磁道，磁道由一组同心圆构成（每条磁道存储相同数目的二进制位，内层磁道密度较外层磁道密度高）
- 每条磁道又分为若干个扇区
- 磁盘的每个表面都定位一个读／写磁头，负责将信息读出或写入磁道
- 磁盘系统的硬件分为两部分：磁盘驱动器和磁盘控制器

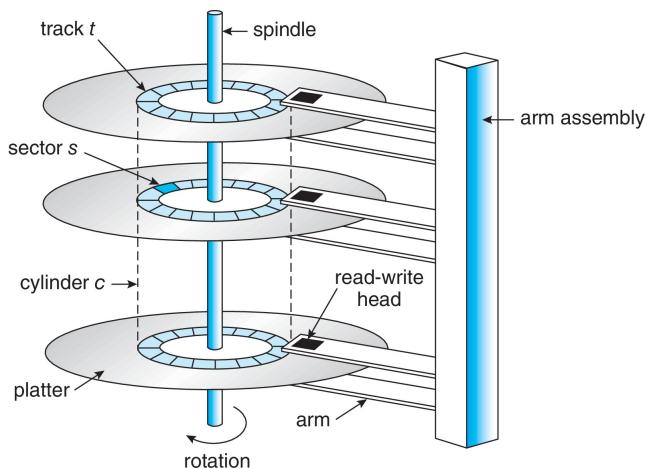
59

6.3.1 磁盘结构(2)



60

6.3.1 磁盘结构(3)



61

6.3.1 磁盘结构(4)

➤ HDD的类型

- 硬盘、固态硬盘
- 单片盘、多片盘
- 固定头磁盘
 - 每个磁道有一个磁头，磁道间的转换非常迅速，I/O速度快，但需要大量的磁头，设备成本很高，应用于大容量磁盘
- 浮动头磁盘
 - 在一个磁盘表面只安装一个磁头，通过移动磁头可方便地存取不同磁道上的信息，要求专门的硬件来移动磁头，设备成本降低，但I/O速度较慢，应用于中小型磁盘

62

6.3.1 磁盘结构(5)

➤ HDD的访问时间

- 磁盘上的信息通过多重编址定位
 - ◆ 定位信息
 - 硬盘：驱动器号、柱面号、磁头号及扇区号
 - 信息是按块存储的，每个块（称之为扇区）由硬件指定大小
 - 2010年前，扇区大小一般为512B
 - 目前，扇区大小一般为4KB

63

6.3.1 磁盘结构(6)

- 为了存取磁盘上的一个扇区，磁盘的速度由三部分时间组成
 - ◆ 寻道(seek)时间：系统移动磁头至相应的磁道或柱面上的时间
 - ◆ 旋转(Latency)时间：磁头到达指定磁道后，等待所需的扇区旋转到读/写头下的时间
 - ◆ 传输时间：数据在磁盘与主存之间实现数据传输所用时间
- 服务一个磁盘请求的总时间是上述三者（寻道时间、旋转时间、传输时间）之和

64

6.3.1 磁盘结构(7)

- NVD(Nonvolatile memory)的物理结构
 - NVD设备是电子设备
 - 主要由控制器和用于存储数据的闪存NAND芯片组成
 - 其他NVD技术包括DRAM等
 - 固态磁盘(solid-state disk, SSD)
 - ◆ 比HDD更可靠
 - ◆ 无需寻道，访问速度比较快
 - ◆ 相对于HDD，容量较小，每MB价格较贵

65

6.3.1 磁盘结构(8)

- NVD限制
 - ◆ 可以以“页”(类似于扇区)来读和写，但不能覆盖数据(overwrite)，只能先擦除NAND单元
 - ◆ 擦除以几页大小的“块”增量发生，比读取(最快的操作)或写入(比读取的速度慢，但比擦除的速度快)花费更多的时间
 - ◆ 其生命周期以“每日驱动器写数DWPD(Drive Writes Per Day)”为单位

66

6.3.2 磁盘调度(1)

➤ HDD调度

- 目标：使磁盘的平均寻道时间最少
- 调度算法
 - ◆ 先来先服务FCFS
 - ◆ 最短寻道时间优先SSTF
 - ◆ 扫描法SCAN及循环扫描法C-SCAN
 - ◆ 查询法LOOK及循环查询法C-LOOK
- 磁盘请求序列（磁头）：
98,183,37,122,14,124,65,67
- 初始磁头位置：53

67

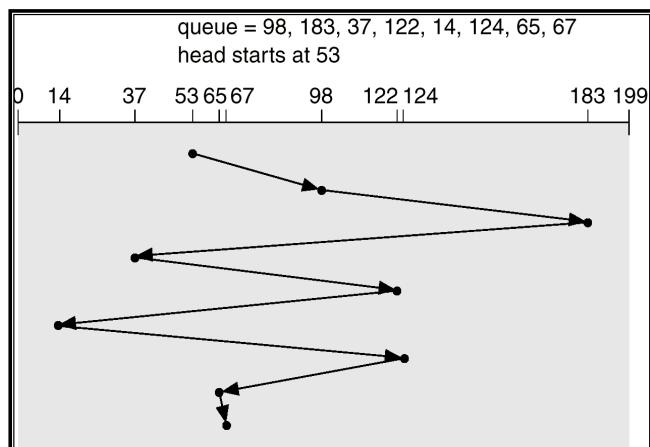
6.3.2 磁盘调度(2)

➤ 先来先服务FCFS

- 最简单的一种调度形式
- 磁头移动序列
 - ◆ 53,98,183,37,122,14,124,65,67
- 磁头移动经过的磁道
 - ◆ $45+85+146+85+108+110+59+2 = 640$
- 优点
 - ◆ 容易实现，公平合理
- 缺点
 - ◆ 完全不考虑队列中各个请求情况，致使磁头频繁地移动

68

6.3.2 磁盘调度 (3)



先来先服务FCFS

69

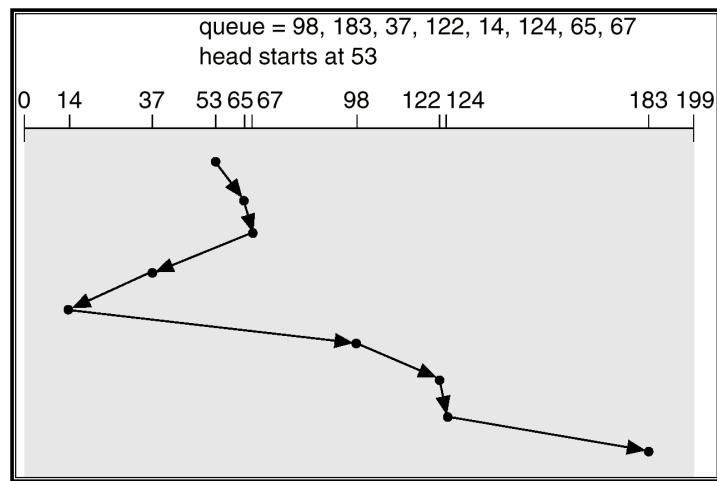
6.3.2 磁盘调度 (4)

➤ 最短寻道时间优先SSTF

- 在将磁头移向下一请求磁道时，总是选择移动距离最小的磁道
- 磁头移动序列
 - ◆ 53, 65, 67, 37, 14, 98, 122, 124, 183
- 磁头移动经过的磁道
 - ◆ $12+2+30+23+84+24+2+59 = 236$
- SSTF算法虽然比FCFS算法优越，但它不是最优的算法

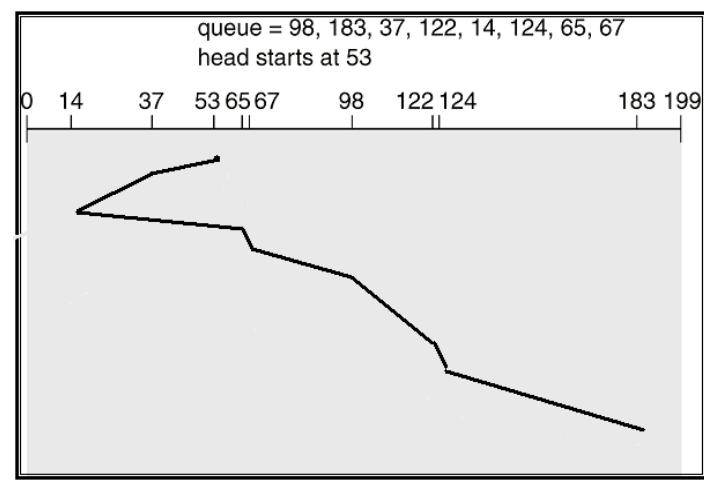
70

6.3.2 磁盘调度(5)



71

6.3.2 磁盘调度(6)



6.3.2 磁盘调度(7)

- ◆ 将磁头由53号先移至37号(即使它不是最近的), 再移到14号, 然后再移到65, 67, 98, 122, 124及183号
- ◆ 总的移动距离可减少到210个磁道
 - $18+23+51+2+31+24+2+59=210$
- ◆ 原因
 - 一方面使磁头改变方向最少
 - 另一方面大大加快了服务请求, 提高了系统处理效率
- SSTF可能会导致饥饿问题

73

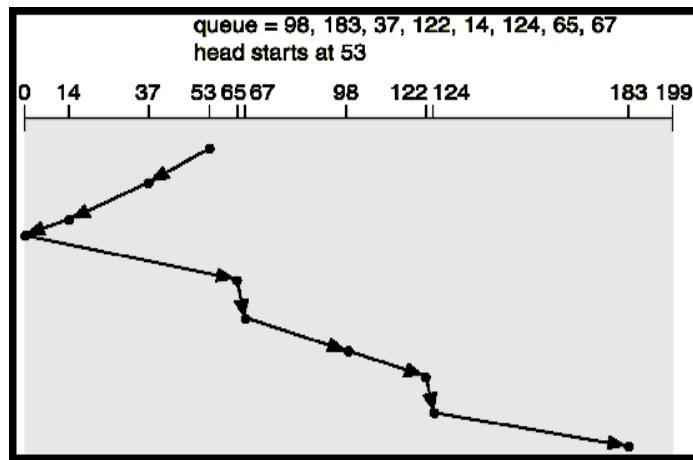
6.3.2 磁盘调度(8)

➤ 扫描法SCAN

- 读/写头在开始由磁盘的一端向另一端移动时, 随时处理所到达的任何磁道上的服务请求, 直到移到磁盘的另一端为止; 在磁盘另一端上, 磁头的方向反转, 继续完成各磁道上的服务请求
- 磁头总是连续不断地从磁盘的一端移动到另一端
- 磁头移动序列
 - ◆ 53,37,14,0,65,67,98,122,124,183
 - ◆ 53,65,67,98,122,124,183,199,37,14
- 磁头移动经过的磁道
 - ◆ $16+23+79+2+31+24+2+59=236$
 - ◆ $12+2+31+24+2+59+178+23=331$

74

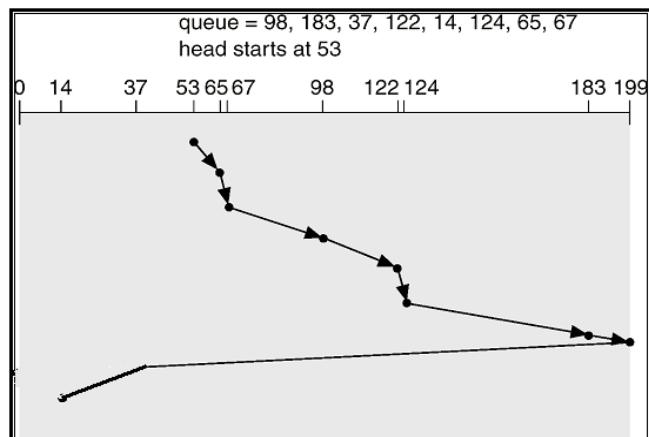
6.3.2 磁盘调度 (9)



扫描法SCAN-1

75

6.3.2 磁盘调度 (10)



扫描法SCAN-2

76

6.3.2 磁盘调度(11)

- 在使用SCAN之前，不仅要知道磁头移动的最后位置，而且还需要知道磁头移动的方向
- 扫描法也叫电梯算法(elevator)，该方式与电梯在各层间的往返移动非常类似
- 特点
 - ◆ 如果正好有一个请求在磁头前进方向上到达，那么，这个请求将会立即得到处理。然而，如果一个请求在磁头刚刚移动过后到达，那么它只能等到磁头反方向移到时才能得到处理

77

6.3.2 磁盘调度(12)

➤ 循环扫描法C-SCAN

- 假定：对磁道的请求是均匀分布的，考虑对磁道的请求密度，当磁头达到一端并反向时，落在磁头之后的请求相对较少，这是由于这些磁道刚刚被处理，而磁盘另一端的请求密度相当地高，且等待的时间较长
- 与SCAN算法类似，在将磁头从一端移向另一端时，随时处理到达的请求。但是，当它已到达另一端时，磁头立即返回到开始处。也即回程时，不处理任何请求

78

6.3.2 磁盘调度 (13)

- 磁头移动序列

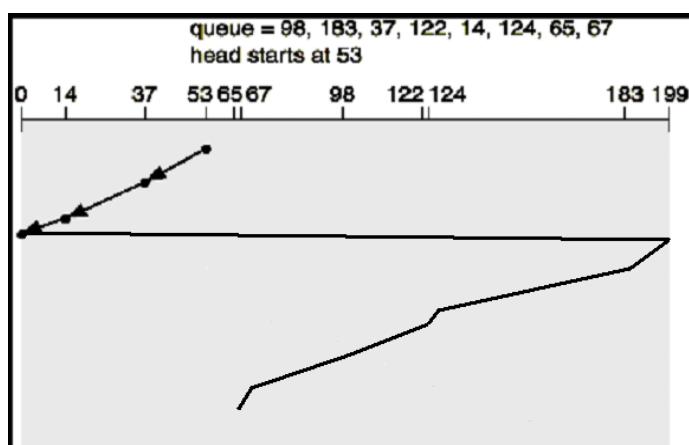
- ◆ 53, 37, 14, 0, 199, 183, 124, 122, 98, 67, 65
- ◆ 53, 65, 67, 98, 122, 124, 183, 199, 0, 14, 37

- 磁头移动经过的磁道

- ◆ $16+23+(14+199+16)+59+2+24+31+2=386$
- $16+23+169+59+2+24+31+2=326$
- ◆ $12+2+31+24+2+59+(16+199+14)+23=382$
- $12+2+31+24+2+59+169+23=322$

79

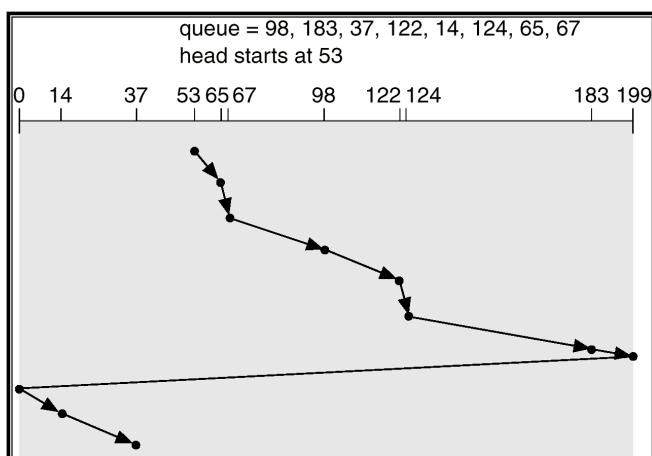
6.3.2 磁盘调度 (14)



循环扫描法C-SCAN-1

80

6.3.2 磁盘调度 (15)



循环扫描法C-SCAN-2

81

6.3.2 磁盘调度 (16)

➤ 查询LOOK及循环查询C-LOOK

- 扫描法和循环扫描法都是将磁头由磁盘的一端移向另一端，但实际上没有一个算法是这样实现的。通常，磁头在向任何方向移动时都是只移到最远的一个请求的磁道上。一旦在前进的方向上没有请求到达，磁头就反向移动
- 即在将磁头向前移动之前，先查询有无请求，若有，才移动，否则，立即反向

82

6.3.2 磁盘调度 (17)

- 磁头移动序列

- ◆ 53, 37, 14, 65, 67, 98, 122, 124, 183

- ◆ 53, 65, 67, 98, 122, 124, 183, 37, 14

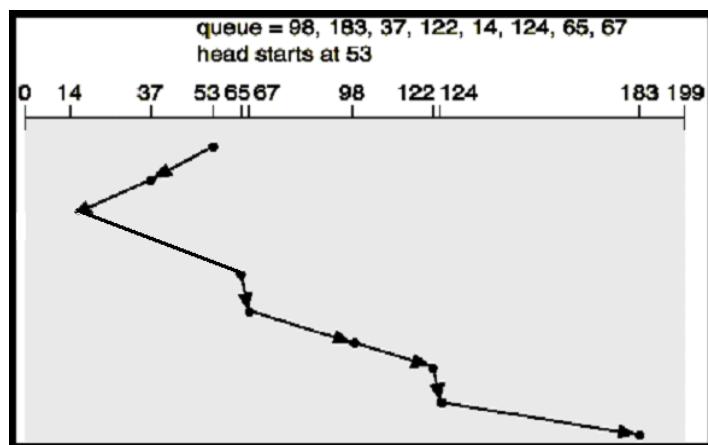
- 磁头移动经过的磁道

- ◆ $16 + 23 + 51 + 2 + 31 + 24 + 2 + 59 = 208$

- ◆ $12 + 2 + 31 + 24 + 2 + 59 + 146 + 23 = 299$

83

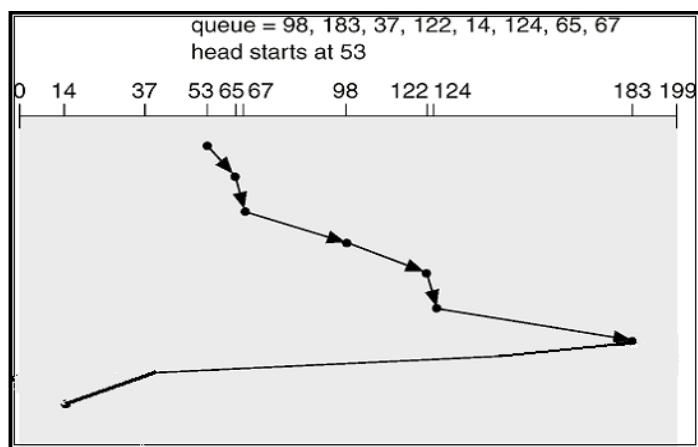
6.3.2 磁盘调度 (18)



查询LOOK-1

84

6.3.2 磁盘调度 (19)



查询LOOK-2

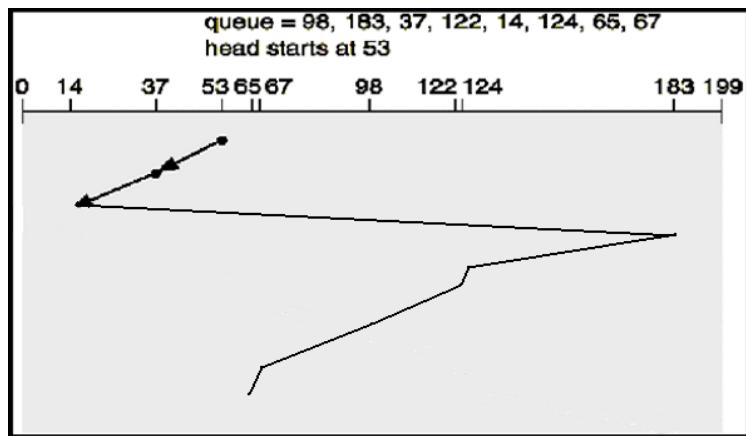
85

6.3.2 磁盘调度 (20)

- 磁头移动序列
 - ◆ 53, 37, 14, 183, 124, 122, 98, 67, 65
 - ◆ 53, 65, 67, 98, 122, 124, 183, 14, 37
- 磁头移动经过的磁道
 - ◆ 16+23+169+59+2+24+31+2=326
 - ◆ 12+2+31+24+2+59+169+23=322

86

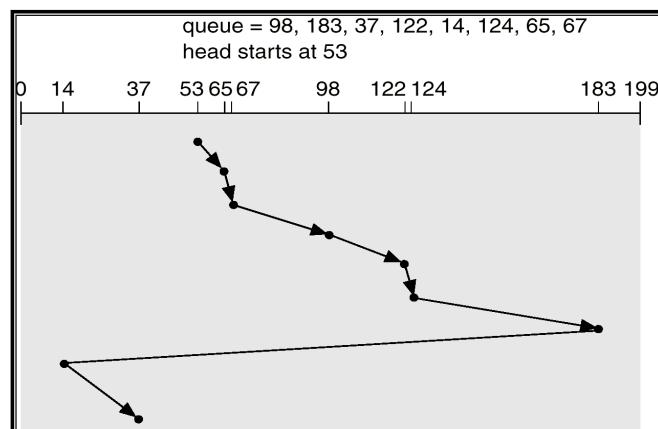
6.3.2 磁盘调度 (18)



循环查询C-LOOK-1

87

6.3.2 磁盘调度 (19)



循环查询C-LOOK-2

88

6.3.2 磁盘调度 (20)

➤ 磁盘调度算法的比较

- SSTF算法是公认的、最具吸引力的算法
- SCAN和C-SCAN对于磁盘负载较重的系统更为合适
- 任何调度算法性能优劣都与进程对磁盘的请求数量和方式紧密相关；当磁盘等待队列中的请求数量很少超过一个时，所有的算法都是等效的。在这种情况下，最好采用FCFS算法

89

6.3.2 磁盘调度 (21)

- 文件的分配方法将大大地影响对磁盘的服务请求
 - ◆ 当一个程序读磁盘上的一个大的连续文件时，尽管请求读盘的要求很多，但由于各信息块连在一起，磁头的移动距离却很小
 - ◆ 若程序读的是一个链接文件或索引文件，尽管这种文件的磁盘空间利用充分，但可能使信息块分布在整個盘上，导致磁头的可观的移动代价

90

6.3.2 磁盘调度 (22)

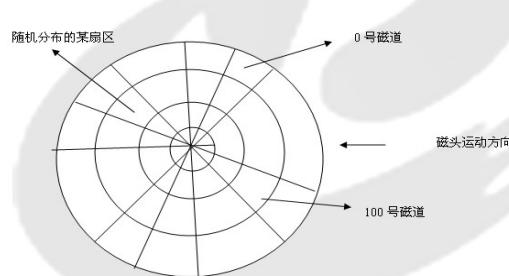
- 索引块的位置也非常重要
 - ◆ 目录文件需要频繁进行存取，应将目录文件放在磁盘的中间，而不是两端，从而可以有效地减少磁头的移动
 - ◆ 应将磁盘调度算法写成一个独立的模块，以便必要时用不同的算法来替换或干脆去掉不用
- 初始算法可选用FCFS算法或SSTF算法

91

6.3.2 磁盘调度 (23)

示例：假设计算机系统采用CSCAN（循环扫描）磁盘调度策略，使用2KB的内存空间记录16384个磁盘块的空闲状态。

(1) 请说明在上述条件下如何进行磁盘块空闲状态管理。



92

6.3.2 磁盘调度 (24)

(2) 设某单面磁盘旋转速度为每分钟6000转，每个磁道有100个扇区，相邻磁道间的平均移动时间为1ms。若在某时刻，磁头位于100号磁道处，并沿着磁道号增大的方向移动（如下图所示），磁道号请求队列为50, 90, 30, 120，对请求队列中的每个磁道需读取1个随机分布的扇区，则读完这些扇区共需要多少时间？要求给出计算过程。

93

6.3.2 磁盘调度 (25)

(1) 因为 $2KB = 2 * 1024 * 8bit = 16384bit$ ，因此可以使用位图法对磁盘块空闲状态进行管理，每1个bit表示一个磁盘块。

(2) A. 因为磁盘旋转速度为每分钟6000转，因此磁盘旋转一圈的时间为10ms，平均旋转时间为 $10 * 0.5 = 5ms$ ，因此，读取4个扇区的旋转时间为 $5 * 4 = 20ms$ 。

B. 由于每个磁道有100个扇区，所以通过一个扇区的时间为0.1ms，所以读取4个扇区需 $0.1 * 4 = 0.4ms$ 。

94

6.3.2 磁盘调度 (26)

- C. 磁道号请求队列为 50, 90, 30, 120, 磁头位于 100 号磁道处沿着磁道号增大的方向移动, 根据 CSCAN 算法, 被访问的磁道号顺序则为 100, 120, 30, 50, 90, 因此寻道时间为 $[(120-100)+(120-30)+(50-30)+(90-50)] * 1\text{ms} = (20+90+20+40) * 1\text{ms} = 170\text{ms}$ 。
- D. 全部时间为 $20 + 0.4 + 170 = 190.4\text{ms}$ 。

95

6.3.3 磁盘的错误处理 (1)

- 常见错误
 - 程序性错误(例如, 申请不存在的扇区)
 - 瞬时检查和错误(例如, 磁头上有灰尘)
 - 永久性的检查和错误(磁盘块物理损坏)
 - 寻道错误(例如, 寻找柱面6, 磁臂却定位到柱面7)
 - 控制器错误(控制器拒绝接收命令)
- 所有错误都由磁盘驱动程序一一进行处理
- 程序性错误

96

6.3.3 磁盘的错误处理(2)

- 驱动程序命令控制器去查找一个不存在的柱面、读一个不存在的扇区、使用不存在的磁头、与一个不存在的存储器地址交换数据时，都产生程序性错误。大多数控制器对发给它的参数进行检查，并告知是否合法
 - 瞬时检查和错误是由于磁盘表面与磁头之间的灰尘引起
 - 通常是重复执行这个操作，就可消除错误

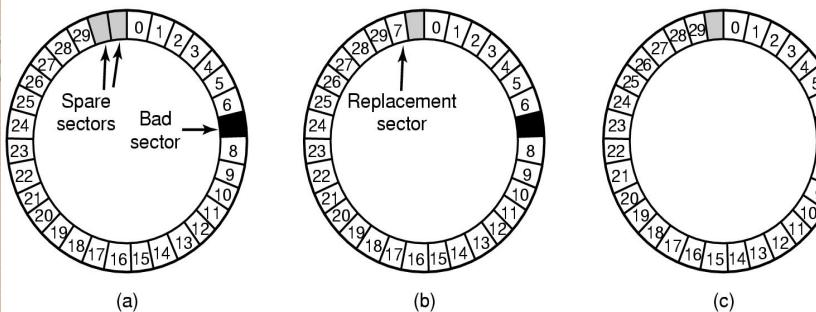
97

6.3.3 磁盘的错误处理(3)

- 倘若错误继续存在，则将该块标记为坏块
- 一些“智能”磁盘控制器保留了几个备用磁道，这些磁道对用户程序不开放。当磁盘进行格式化时，控制器确定哪些块是坏的，自动由备份磁道替换它。将坏磁道映射到备用磁道的表格保留在控制器内部存储器和磁盘上，对驱动程序透明
- 若控制器秘密地改用其它备份柱面，为磁盘设计的优化调度算法可能变得很坏

98

6.3.3 磁盘的错误处理(4)



- (a)磁道中有一个坏扇区
(b)使用一个空白扇区替换该坏扇区
(c)跳过该坏扇区，并顺序移动后续扇区

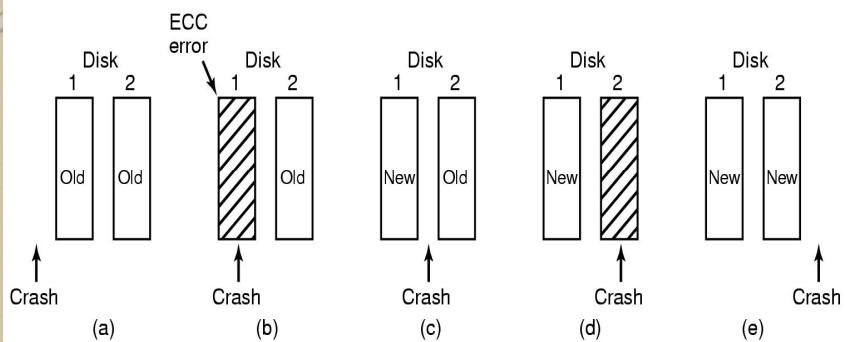
99

6.3.3 磁盘的错误处理(5)

- 寻道错误是由于磁臂的机械故障引起
 - 控制器内部记录磁臂位置，为了执行寻道，它泄放一系列脉冲给磁臂马达，每个柱面一个，这样将臂移到新的柱面。当臂移到目标位置时，控制器读实际的柱面号(驱动器格式化时写的)，如果位置不对，则出现寻道错误
 - 关于寻道错误，有些控制器可以自动修正，而有些控制器(包括IBM PC在内)只设置一个错误标志位，其它工作留给驱动程序

100

6.3.3 磁盘的错误处理(6)



101

本章要点

- I/O管理的目标与功能
- I/O设备及其分类
- I/O控制方式
- 中断处理程序
- 设备驱动程序
- 设备独立性
- 缓冲区和缓冲区管理
- SPOOLing技术
- 先来先服务(FCFS)磁盘调度算法
- 最短寻道时间优先(SSTF) 磁盘调度算法
- 扫描(SCAN)及循环扫描(CSCAN) 磁盘调度算法

102