为什么需要设备管理?

- 没有I/O (Input /Output)设备的计算机就像一个聋哑瞎子，就像一个没有轮子的汽车
- I/O性能经常成为系统性能的瓶颈
  - CPU性能不等于系统性能
  - I/O响应时间也是一个重要因素
  - CPU性能越高，与I/O差距越大
    - 弥补：更多的进程
    - 进程切换多，系统开销大

设备管理—重要性

- 计算机应用的多样性和使用的普及性
  - 设备管理决定操作系统的适用范围
- 设备管理本身的特性复杂
  - I/O的工作过程与结构是理解操作系统的工作过程与结构的关键

设备管理

- I/O硬件概念
  - 常见I/O设备分类
  - 设备控制器
  - I/O控制方式
  - I/O控制方式的发展过程
- I/O设备子系统
- 存储设备
- 小结

设备管理

- 目的与要求
  - 掌握I/O控制的原理
  - 理解设备管理子系统的层次，功能及技术
  - 了解磁盘设备
- 重点与难点
  - 三种不同的I/O控制方式
  - 三种不同的设备使用方法
  - 层次结构；设备驱动程序；
  - 磁盘调度的方法；
- 作业
  - 6.6， 6.11
  - 思维导图

课后阅读与思考

- 教材
  - 第6章
- Modern Operating System (2nd edition)
  (现代操作系统)
  - Chapter 5 Input/Output

I/O硬件概念

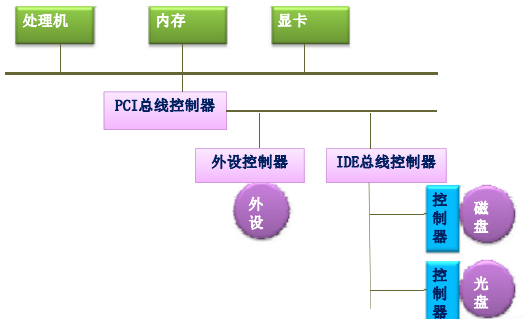
- 常见外部设备分类
  - 人机交互设备(字符设备,发送接收以字符方式)
  - 存储类型设备(块设备,读写以数据块方式)
  - 网络通信设备（速度介于两者之间）
- I/O设备的特点
  - 设备的使用目的不同
    - 操作系统实现策略不同
  - 控制的复杂性不同
    - 设备无关性：
      - 设备管理子系统屏蔽了对各类设备的控制细节，提供一个对各种设备尽可能一致的，简单易用的接口
  - 数据传输单位不同
    - 字节、字符流、数据块

设备控制器（I/O部件）

- I/O设备通常包含一个机械部件和一个电子部件
  - 电子部件被称作I/O部件或设备控制器
  - 机械部分是　设备本身
- 操作系统一般只与控制器打交道，而非设备本身

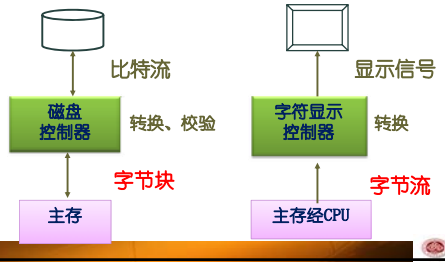
I/O设备模型

- 处理器、内存与外设之间 的连接



控制器

- 控制器的任务是在外部设备与CPU（或内存）之间完成比特流（外部信号）和字节流（块）之间的转换
- 每个控制器都有一些用来与CPU通讯的I/O寄存器
  - 操作系统通过向这些寄存器写命令字来执行I/O功能



控制器的寄存器编址

- 主存映像I/O（I/O统一编址）
  - 寄存器地址占用主存物理地址空间的一部分
  - 以访问内存的方式访问外设寄存器
  - 多数RISC计算机采用该方案
  - 优点：统一访问
  - 缺点：占用内存的地址空间
- I/O专用编址（I/O独立编址）
  - 寄存器有独立的I/O地址空间
  - 专用的I/O操作指令，例如IN，OUT
  - 优点：不占用内存空间，I/O地址空间大
  - 缺点：增加指令

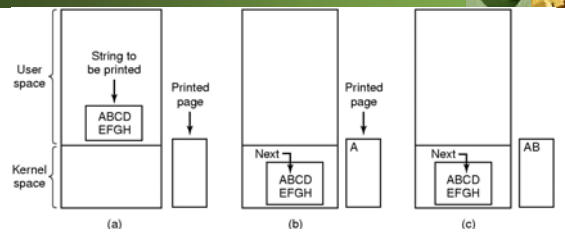
输入输出控制方式

- 程序直接控制方式
- 中断控制方式
- DMA方式
- 通道方式

程序直接控制方式

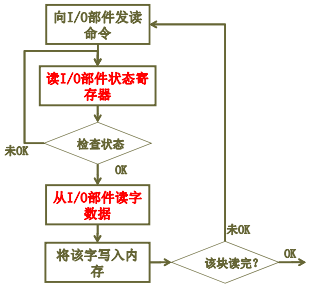
- 早期计算机系统对I/O设备的一种管理方式
  - CPU执行程序直接控制I/O操作的全过程，包括传输数据、发送读写命令、测试设备状态
  - I/O部件接收到相应的命令后，设置相应的I/O状态寄存器值
  - 由CPU执行相应指令读取I/O完成状态
  - I/O数据通过CPU寄存器转发

程序直接控制方式




```
copy_from_user(buffer, p, count);          /* p is the kernel bufer */
for (i = 0; i < count; i++) {              /* loop on every character */
    while (*printer_status_reg != READY) ; /* loop until ready */
    *printer_data_register = p[i];         /* output one character */
}
return_to_user();
```

程序直接控制方式



程序直接控制方式

- 优点
  - 轮询比I/O设备的速度要快很多，一般不会发生不能及时处理问题
- 缺点
  - 能处理的输入输出设备有限
  - 占据CPU处理时间
  - 效率低的方式，现代计算机很少使用

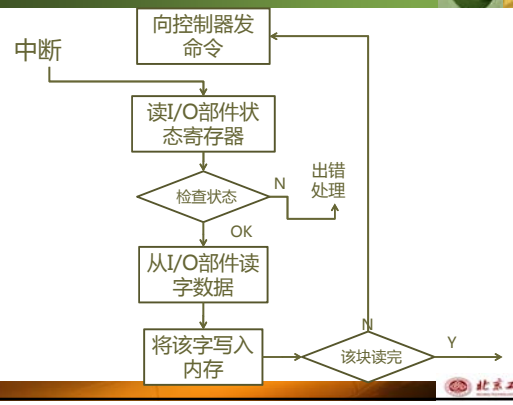
中断控制方式

- 方式
  - CPU向I/O部件发出命令后，转去做其他有用的工作
  - 当I/O部件准备好数据后，利用中断通知CPU
  - 再由CPU执行程序完成数据传输
- 优点
  - CPU不必反复测试寄存器状态，节约了时间
  - 提高处理器利用率
  - 能支持多道程序和I/O设备并行

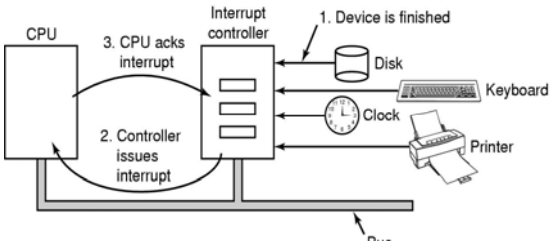
中断控制方式

- 缺点
  - 仍然消耗大量的CPU时间，因为每个字的数据传输都必须经过CPU寄存器转发
  - CPU每次处理的数据量少（通常不超过几个字节），只适于数据传输率较低的设备
  - 中断次数急剧增加会造成CPU无法响应中断和出现数据丢失现象

中断控制方式

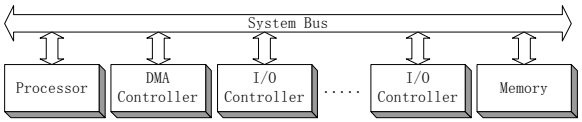


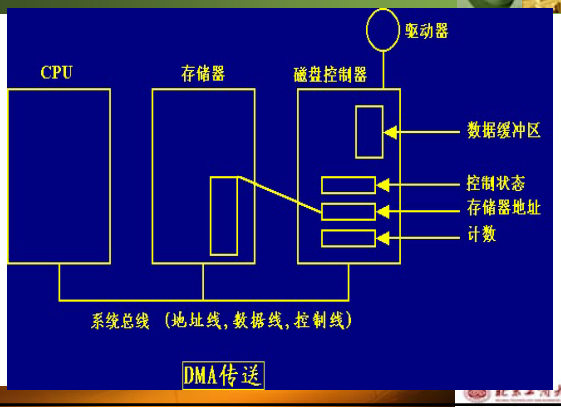
Interrupts



DMA方式

- 数据在内存与I/O设备间直接进行成块传输
  - 程序设置DMA控制器中的若干寄存器值（如内存始址，传送字节数）
  - DMA控制器完成内存与外设的成批数据交换
  - 操作完成时由DMA控制器向CPU发出中断





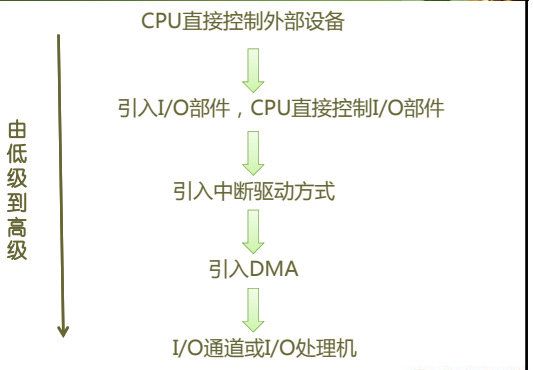
DMA方式

- 优点：
  - CPU只需干预I/O操作的开始和结束，而其中的一批数据读写无需CPU控制，适于高速设备
- 缺点：
  - DMA控制器功能的强弱是决定DMA效率的关键
  - 数据传送单位的增大意味着传送次数的减少

DMA/中断对比

	中断方式	DMA方式
发出中断的时机	在数据缓冲寄存器满之后	所要求传送的数据块全部传送结束时
控制	由CPU控制完成的	由DMA控制器控制
数据量	以字节为单位	以数据块为单位
并行设备多或速度不匹配时	数据丢失现象	无数据丢失

输入输出控制方式的发展过程



设备管理

- I/O硬件概念
- I/O设备子系统
  - 设备的使用方法
  - I/O层次结构
  - 设备驱动程序
  - 缓冲技术
- 存储设备
  - 常见存储设备
  - 磁盘调度
  - 磁盘阵列
- 小结

设备的使用方法

- 设备相关系统调用
  - 申请设备
    - 该系统调用中有参数说明了要申请的设备名称，操作系统处理该系统调用时，会按照设备特性（是独占还是分时共享式使用）及设备的占用情况来分配设备，返回申请是否成功标志
  - 将数据写入设备
  - 从设备读取数据
  - 释放设备
    - 这是申请设备的逆操作

设备相关系统调用

- 存储类外设，用户程序一般通过对文件的访问
  - 由文件管理模块读写存储外设间接使用它们
  - 系统也提供直接使用存储类外设的接口
- 用户级程序不直接使用网络通讯外设
  - 通过SOCKET通讯系统调用接口调用TCP/IP层程序
  - 由IP层程序选择调用网络通讯设备驱动程序

LINUX的系统调用

- 将数据直接写入软盘中
  - fd=open(“/dev/fd0”,O_RDRW);
    - 申请软盘，/dev/fd0代表软盘
  - lseek(fd,1024,0);
    - 将软盘当前I/O位置定位到1024字节位置
  - Write(fd,buffer,36);
    - 将用户缓冲区buffer中的36个字节写入软盘
  - …
  - close(fd) ;
    - 释放软盘
- 直接读/写软盘空间
  - 必须清楚软盘的什么位置存放了什么信息，才能做到正确的读写

独占式使用设备

- 独占式使用设备
  - 以设备完整使用过程（包含多次I/O操作）为单位使用设备
  - 在申请设备时，如果设备空闲，就将其独占，不再允许其它进程申请使用
    - 一直等到该设备被释放，才允许被其它进程申请使用
  - 设备利用率很低

分时式共享使用设备

- 分时式共享
  - 就是以一次I/O为单位分时使用设备
  - 不同进程的I/O操作请求以排队方式分时地占用设备进行I/O

SPOOLing方式使用外设

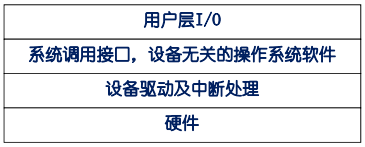
- 假脱机输入输出技术
  - 批处理操作系统时代引入
- 实质是对输入／输出数据成批处理
- 例如
  - 所有输出数据在进程运行时写到虚拟设备（文件/输出井）当中，文件排到打印输出队列，打印进程申请占用打印机后，成批读出文件中数据，并送打印机打印出去

设备管理

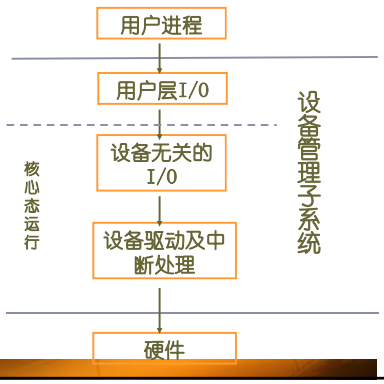
- I/O硬件概念
- I/O设备子系统
  - 设备的使用方法
  - I/O层次结构
  - 设备驱动程序
  - 缓冲技术
- 存储设备
  - 常见存储设备
  - 磁盘调度
  - 磁盘阵列
- 小结

输入输出层次结构

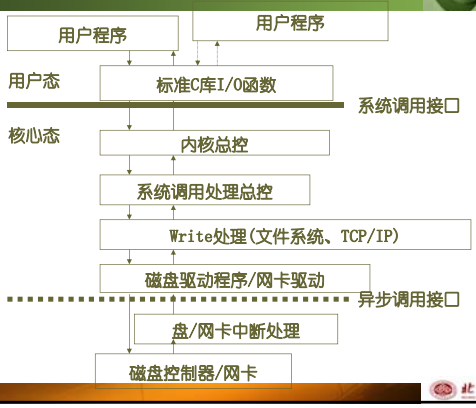
- 通常，操作系统将设备管理系统划分并组织成三个层次：
  - 用户层I/O
  - 设备无关的I/O
  - 设备驱动及中断处理
- 屏蔽实现细节，为上层服务



逻辑结构图



系统调用各模块关系图



用户层I/O

- 与设备的控制细节无关，不直接与设备打交道
- 将设备(或在设备上的逻辑对象）看作逻辑资源
- 为用户进程提供各类I/O函数
- 用户以设备标识符和一些简单的函数来使用设备，如打开、关闭、读、写等
  - 如C库中的函数fopen()；fread()；fwrite()；fclose()等

设备无关的I/O

- 基本功能
  - 执行适用于所有设备的公共I/O功能
  - 向其上层提供统一的系统调用接口
- 主要任务
  - 设备命名及与设备驱动程序的映射
    - 在UNIX中，如/dev/tty00唯一地确定了一个inode数据结构，其中包含了主设备号，通过主设备号可以找到相应的设备驱动程序
  - 设备访问保护（权限检查）
  - 设备I/O数据缓冲机制
  - 错误报告
  - 文件系统管理模块

设备驱动与中断处理

- 设备驱动程序
  - 所有与设备相关的代码
  - 从与设备无关的软件中接收I/O的请求，排入请求队列或执行之
- 中断处理
  - 当进程进行I/O操作时，将其阻塞至I/O操作结束并发生中断
  - 中断发生时，由中断处理程序启动请求排队的下一请求（如果有）并解除等I/O进程的阻塞状态，使其能够继续执行

设备管理

- I/O硬件概念
- I/O设备子系统
  - 设备的使用方法
  - I/O层次结构
  - 设备驱动程序
  - 缓冲技术
- 存储设备
  - 常见存储设备
  - 磁盘调度
  - 磁盘阵列
- 小结

设备驱动程序

- 设备驱动程序接口函数
  - 驱动程序初始化函数
    - 使驱动程序其它函数能被上层正常调用，而做一些针对驱动程序本身的初始化工作
    - 如向操作系统注册该驱动程序的接口函数
    - 在系统启动时或驱动程序安装入内核时执行
  - 驱动程序卸载函数
    - 驱动程序初始化函数的逆过程，在支持驱动程序可动态加载卸载的系统中才需要

设备驱动程序

- 设备驱动程序接口函数
  - 申请设备函数
    - 申请一个驱动程序所管理的设备
    - 按照设备特性进行独占式占用或者分时共享式占用
    - 如果是独占式申请成功还应该对设备做初始化工作
  - 释放设备函数
    - 是申请设备函数的逆过程

设备驱动程序

- 设备驱动程序接口函数
  - I/O操作函数
    - 实现对设备的I/O
    - 对独占型设备
      - 包含了启动I/O的指令
    - 对分时共享型设备
      - 该函数通常将I/O请求形成一个请求包，将其排到设备请求队列，如果请求队列空，则直接启动设备。
  - 中断处理函数
    - 在设备I/O完成时向CPU发中断后被调用
    - 对I/O完成作善后处理，一般是找到等待刚完成I/O请求的阻塞进程，将其就绪，使其能进一步作后续工作
    - 如果存在I/O请求队列，则启动下一个I/O请求

设备管理有关的数据结构

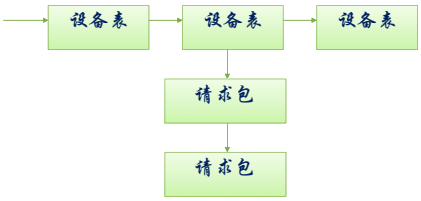
- 描述设备、控制器等部件的表格
  - 系统中常常为每一个部件、每一台设备分别设置一张表格，常称为部件控制块或设备表
  - 这类表格具体描述设备的类型、标识符、进行状态，以及当前使用者的进程标识符等
- 建立同类资源的队列
  - 系统为了方便对I/O设备的分配管理，通常在设备表的基础上通过指针将相同物理属性的设备联成队列（称设备队列）

设备管理有关数据结构关系

- 面向进程I/O请求的动态数据结构
  - 每当进程发出块I/O请求时，系统建立一张表格（称I/O请求包）
  - 将此次I/O请求的参数填入表中，同时也将该I/O有关的系统缓冲区地址等信息填入表中
  - I/O请求包随着I/O的完成而删除

设备管理有关数据结构关系

- 建立I/O队列
  - 如请求包队列



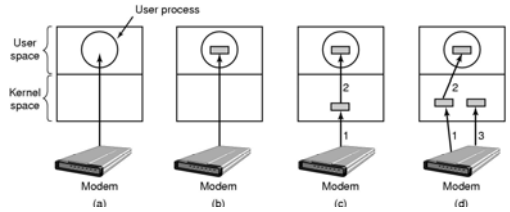
设备管理

- I/O硬件概念
- I/O设备子系统
  - 设备的使用方法
  - I/O层次结构
  - 设备驱动程序
  - 缓冲技术
- 存储设备
  - 常见存储设备
  - 磁盘调度
  - 磁盘阵列
- 小结

缓冲技术

- 缓冲的引入
  - 缓解CPU与外部设备之间速度不匹配的矛盾
    - 减少对CPU的中断次数，提高CPU和I/O设备之间以及各个I/O设备之间的处理并行性
    - 把I/O读写操作变成内存读写操作
  - 提高外设利用率，尽可能使外设处于忙状态
  - 支持进程换出内存
- 缓冲区设置
  - 硬缓冲：在设备中设置缓冲区，由硬件实现
  - 软缓冲：在内存中开辟一个空间，用作缓冲区
- 缓冲的种类
  - 单缓冲(single buffer)
  - 双缓冲(double buffer)
  - 循环缓冲 (buffer pool)

I/O Buffer



(a) Unbuffered input
(b) Buffering in user space
(c) Buffering in the kernel followed by copying to user space
(d) Double buffering in the kernel

单缓冲

- 当用户进程发出I/O请求时，操作系统在主存的系统空间为该操作分配一个缓冲区，可以实现预读和延迟写
- 块设备的单缓冲
  - 输入——预读
    - 设备控制器来的数据先送入系统缓冲区
    - 用户进程将数据块移到用户进程空间
    - 立即请求下一个数据块
      - 假设下一数据块将被使用
  - 输出——延迟写
    - 数据块从用户空间复制到系统缓冲区
    - 继续执行用户进程，需要时可换出主存
    - 操作系统最终将系统缓冲区内容输出到设备上
  - 特点
    - 用户进程处理数据和读入数据操作并行
    - 用户进程可以被换出主存
    - 增加操作系统复杂度

单缓冲

- 字符设备的单缓冲
  - 一次一行的方式
    - 缓冲区保存一行
    - 输入时
      - 用户进程阻塞至整行内容全部输入后才被换醒
    - 输出时
      - 用户进程将要输出的一行信息送入缓冲区，继续运行
      - 若缓冲区不可用，用户进程被阻塞
  - 一次一个字节
    - 传感器、控制杆之类的外设

双缓冲

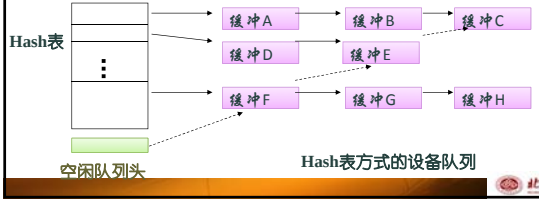
- 两个缓冲区，一个用来输入，一个用来输出
- CPU和外设都可以连续处理而无需等待对方
- 可以实现并行
  - 用户数据区-缓冲区之间交换数据
  - 缓冲区—外设之间交换数据

循环缓冲

- 多个缓冲区，一部分作输入，一部分作输出
- 利用有限缓冲区的生产者/消费者模型操作缓冲区
- 在多道环境中，系统中有多个I/O设备和多个进程时，缓冲区是有效工具
  - 可以提高系统性能
  - 改善进程运行时间

缓冲技术举例

- 磁盘I/O的高速缓冲，用于DMA方式
  - 空闲队列
    - 可用缓冲区队列
  - 设备队列
    - 存放设备数据块数据的缓冲区队列
  - 根据访问的设备号、块号映射到Hash表中的某个表项
  - 缓冲区采用LRU替换算法
    - 最新被用过的缓冲区总位于队尾



设备管理

- I/O硬件概念
  - 常见I/O设备分类
  - 设备控制器
  - I/O控制方式
  - I/O控制方式的发展过程
- I/O设备子系统
  - 设备的使用方法
  - I/O层次结构
  - 设备驱动程序
  - 缓冲技术
- 存储设备
  - 常见存储设备
  - 磁盘调度
  - 磁盘阵列
- 小结

存储设备

- 常见存储外部设备
  - 磁带存储设备
  - 磁盘存储设备
  - 光盘存储设备

磁带

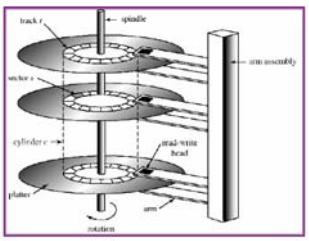
- 顺序存取
- 速度慢，容量小
  - 读写延迟时间5~10ms
- 适合用于增量式备份

光盘

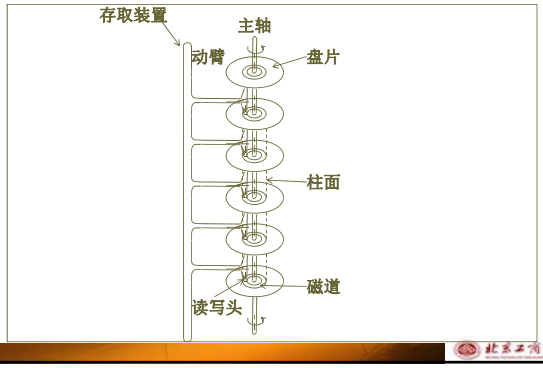
- 高密度
- 随机存取
- 成本低，容量大，存储更可高
- 不会因为磁和粉尘污染造成数据丢失

磁盘

- 随机存取，容量大，存取速度快
- 扇区为基本传输单位
- 组成
  - 一个磁盘有多个盘面组成
  - 盘面由同心圆磁道组成
  - 盘组中直径相同的磁道形成一个柱面
  - 磁道由扇区组成
  - 物理特性
  - 单磁头
  - 多磁头



多磁头活动头盘示意图

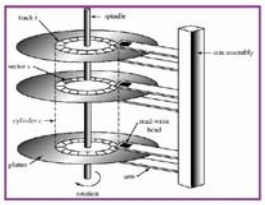


设备管理

- I/O硬件概念
  - 常见I/O设备分类
  - 设备控制器
  - I/O控制方式
  - I/O控制方式的发展过程
- I/O设备子系统
  - 设备的使用方法
  - I/O层次结构
  - 设备驱动程序
  - 缓冲技术
- 存储设备
  - 常见存储设备
  - 磁盘调度
- 小结

磁盘调度

- 磁盘地址编号
  - 台号 - 柱面号 - 盘面号 - 扇区号
- 磁盘上的位置计算
  - t：每个柱面上的磁道数
  - s：每个磁道上的扇区数
  - 那么：第i 柱面，j 磁头，k 扇区对应的块的编号为：
  - b = k + (s * (j + i * t))

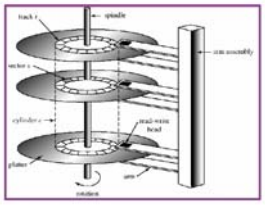


磁盘调度

- 读写磁盘的时间
  - 寻找时间
    - 磁头移到指定柱面的时间
  - 延迟时间
    - 磁头定位于指定磁道的指定扇区的时间
  - 传输时间
    - 数据写入磁盘或从磁盘读出的时间
- 一次访盘时间 = 寻找时间
- + 延迟时间
- + 传输时间

磁盘调度

- 磁盘调度考虑的问题：
  - 寻找时间和延迟时间和信息存储的位置有关
- 磁盘调度的目标
  - 减少寻找时间
  - 减少延迟时间



磁盘调度

- 原则
  - 寻道和延迟都尽可能小的访问者先得到服务
  - 降低访问者的总访问时间
  - 增加磁盘单位时间内的操作次数

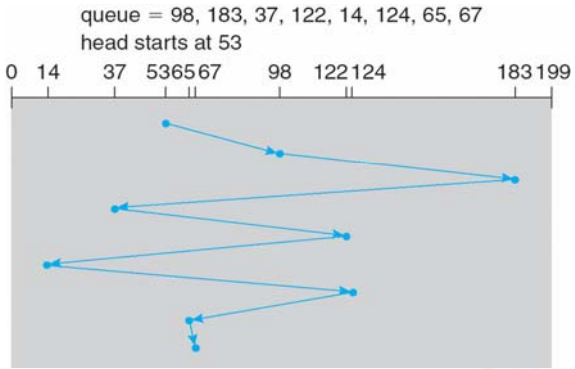
磁盘调度

- 减少寻找时间的方法
  - 先来先服务（FCFS）
  - 最短寻找时间优先（SSTF）
  - 电梯调度（SCAN）
  - 单向扫描（C-SCAN）
- 减少延迟时间的方法
  - 按扇区交替编号连续存放

先来先服务

- FIFO, First In First Out
  - 磁盘I/O执行顺序为磁盘I/O请求的先后顺序
  - 公平性
    - 在磁盘I/O负载较轻且每次读写多个连续扇区时，性能较好
- 优点
  - 简单，公平
- 缺点
  - 效率不高
    - 相邻两次请求可能会造成最内到最外的柱面寻道，使磁头反复移动，增加了服务时间，对机械也不利

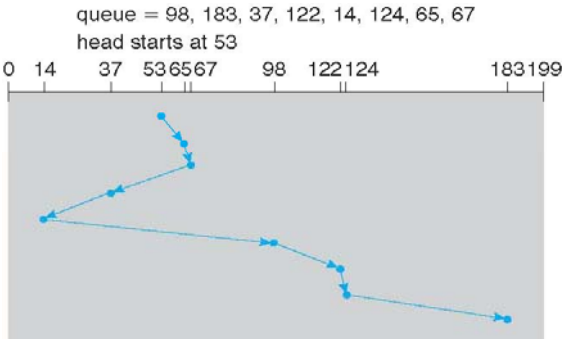
实例



最短查找时间优先

- SSTF, Shortest Service Time First
  - 考虑磁盘I/O请求队列中各请求的磁头定位位置
  - 选择从当前磁头位置出发，移动最少的磁盘I/O请求
  - 目标
    - 使每次磁头移动时间最少
    - 比FIFO算法有更好的性能
- 优点
  - 改善了磁盘平均服务时间
- 缺点
  - 对中间的磁道有利，造成某些访问请求长期等待得不到服务

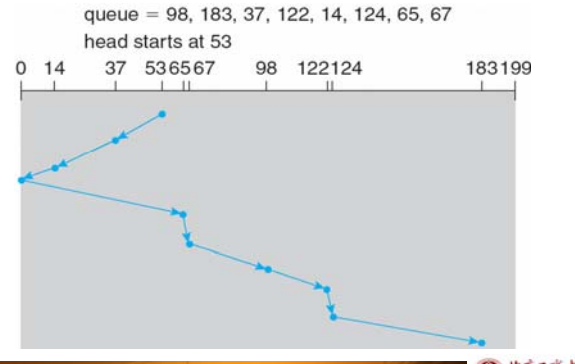
实例



电梯调度

- SCAN调度
  - 从移动臂当前位置开始沿着臂的移动方向去选择离当前移动臂最近的那个柱访问者
  - 如果沿臂的移动方向无请求访问时，就改变臂的移动方向再选择
- 优点
  - 简单、实用而且高效的调度算法
- 缺点
  - 需要记住读写磁头的当前位置和移动臂的当前移动方向

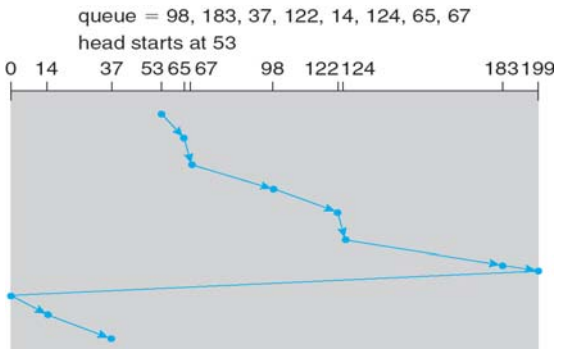
实例



单向扫描调度

- C-SCAN
  - 不考虑访问者等待的先后次序，总是从0号柱面开始向里道扫描，按照要访问的柱面位置次序选择访问者
  - 到达最后一个柱面后，立即返回到0柱面，返回时不为任何访问者服务

实例

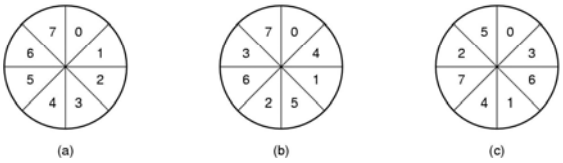


磁盘调度

- 小结
  - 除先来先服务算法以外
    - 根据欲访问的柱面位置来进行调度
    - 新来的访问请求，一旦读写头超过了请求的访问位置，则只能在以后的调度中被选择
  - 每个柱面的各个访问请求都完成后，再改变移动臂的位置

磁盘调度

- 减少延迟时间的方法
  - 信息在磁道上的排列方式影响磁盘的I/O操作时间



小结

- 常见I/O设备分类
  - 字符、块、网络设备
  - 设备无关性
- 设备控制器
  - 基本概念
- I/O控制方式
  - 三种方式的原理、特点
- I/O设备子系统
  - 层次结构，每个层的主要功能，设备驱动程序
- 存储设备
  - 磁盘调度算法

Q&A