设备管理

概论

设备管理的对象是计算机的外围设备，可分为

- 存储型设备，以存储大量信息和快速检索为目标，如磁带机，磁盘
- 输入输出型设备，将外界信息输入计算机，将结果输出，如显示器，键盘

设备管理的目的是方便用户，同时提高设备的利用率，包括

- 外围设备中断管理
- 缓冲区管理
- 外围设备登记和使用情况跟踪，以及分配去配
- 外围设备驱动调度
- 虚拟设备及其实现

输入输出处理

输入输出系统

输入输出系统管理**主存**和**外围设备**之间的输入输出，分为硬件部分（IO 设备，控制部件，通道等）和软件部分（管理软件）

输入输出设备可分为字符型（如显示器）和块型（如硬盘），也可以按存取方式分为顺序存取和直接存取型

输入输出控制方式

按功能强弱和并行度可分为

- 询问方式：程序直接控制 IO，不断轮询（忙等待），CPU 与 IO 设备串行工作，效率低
- 中断方式：CPU 和 IO 设备部分并行，但数据传输仍要 CPU 参与，效率提高
- DMA 方式：IO 设备直接与主存进行数据交换，由 DMA 控制，数据以块为单位在主存和设备之间传输，但是块与块之间操作仍要 CPU 干预
- 通道方式：开始 IO 时执行相应指令，操作结束时通过中断通知 CPU，CPU 和 IO 设备完全并行，效率高，通道可分为

- 字节多路通道，以字节为单位交叉工作，连接大量慢速设备
- 选择（高速）通道，连接快速设备，一次为一个设备服务
- 数组多路通道，上述两种混合

IO 软件

IO 软件的设计目标是高效且通用，设计 IO 软件涉及的主要问题有

- 设备独立性：编写的程序与具体设备无关
- 出错处理（尽量使错误在接近硬件的层面处理）
- 实现同步和异步传输
- 缓冲区建立
- 设备的分配去配，要区别对待共享设备和独占设备，防止死锁

IO 软件的层次从上到下为：

- 用户级 IO 软件
- 设备无关 IO 软件
- 设备驱动程序
- 中断处理程序

中断处理程序主要负责保存现场，判断中断原因，然后唤醒对应设备驱动程序，并在之后完成中断返回

设备驱动程序直接控制 IO 设备，一般由用户进程实现，由内核进行操作。主要功能是设备初始化，以及将抽象的读写请求转换为具体的 IO 请求。设备驱动程序需要实现操作系统规定的 API，是中断驱动的（平时阻塞，中断唤醒）

设备无关 IO 软件是设备驱动程序统一的接口，功能包括设备的命名和保护，以及定义了设备无关的块的大小，以保证高层软件不用关心具体物理扇区大小，管理缓冲区，进行设备分配状态的跟踪和错误处理

用户级的 IO 软件体现为库函数，由用户调用，链接进用户的代码，功能是触发系统调用，启动实际的 IO 操作，也包含了输入输出格式化等功能（printf/scanf）

缓冲技术

缓冲技术产生的原因主要有

- 改善 CPU 和外围设备的速度不匹配
- 协调逻辑记录和物理记录的大小不一致
- 减少 IO 操作对 CPU 的中断
- 放宽 CPU 中断响应时间请求

基本思想为进程写操作时先申请输出缓冲区，然后向其中写入，之后继续计算工作时系统可以将缓冲区内容写入设备，读同理

- 单缓冲：一个缓冲区
- 双缓冲：两个缓冲区交替使用
- 多缓冲：一组缓冲区组成循环缓冲

驱动调度技术

驱动调度：多个访问存储器的 IO 操作，系统需要采用一种调度策略，使其按最佳次序执行

执行效率衡量指标为完成若干个 IO 操作的总时间

影响存取速度的因素有

- 信息在存储器上的排列方式
- 存储空间分配方法
- 调度算法

存储设备物理结构

存储设备可分为顺序存取（磁带）和随机存取（磁盘）

顺序存取的存取时间依赖于存储位置，而随机存取的存取时间与位置无关

磁盘数据在盘片上的存储可分为每磁道等扇区数和每磁道等密度。磁盘的基础单位为扇区，扇区包括前导码（标识扇区开始，柱面号，扇区号），数据块和校验码

磁盘的访问优化

循环排序

基本思想为根据磁道上数据分布顺序对 IO 请求排序，减少旋转时间

需要知道磁头的当前位置

优化分布

按照数据处理的规律，合理安排数据在磁盘上的分布

如扇区间隔排列，因为磁盘扇区读出后需要校验，花费一定时间，这段时间内磁盘会继续转动，可以将相邻的扇区放在校验结束后磁头的位置

交替地址

在磁盘中存放冗余的数据，提高访问速度（e.g.，柱面斜进）

会消耗较多的空间，较适合只读的场合（写的场合会产生一致性问题）

磁盘的搜索定位

即寻道延迟

先来先服务（FCFS）

按照请求的顺序执行，性能较差

最短时间优先（shortest seek time first）

执行查找时间最短的请求，寻道性能较好，但是会存在饥饿现象

扫描算法（scan）

移动臂每次向一个方向移动，扫过所有柱面，遇到 IO 请求则处理，直至到达最后一个柱面再向反向移动。可以克服最短时间优先产生的饥饿问题

电梯调度算法

类似扫描算法，每次选择距离移动方向最近的请求处理，但是并不扫描到最后一个柱面。当当前扫描方向前方已经没有请求时，则反向扫描。每次扫描记录移动方向和位置

分步扫描算法（N-steps scan）

一个柱面上多个请求会导致磁盘臂过久停留在该柱面，为了解决这个问题，将请求分成多个长度为 N 的子队列，每次对一个子队列执行扫描算法，按照 FCFS 处理完一个队列后再处理下一个队列，避免设备被垄断

循环扫描算法

每次从最小柱面扫描到最大柱面，然后返回最小柱面，返回途中不处理请求

独立磁盘冗余阵列（RAID）

使用一组容量较小的独立的可并行的磁盘组成阵列代替单一的大容量磁盘，并加入冗余技术

提高磁盘 IO 速度的方法

提前读

对于顺序存储的数据，在读取当前磁盘块时，提前把下一磁盘块的数据读入缓冲区

延迟写

将要写入磁盘的数据放在缓冲区队尾，当其最终到达队首时才将其写入磁盘，在此期间缓冲区的数据可以被进程多次访问，可以减少写盘次数

虚拟盘

使用内存模拟硬盘，是易失存储