记住-考过-常考-必考-重点

笔记源文件：Markdown，长图，PDF，HTML

1. 操作系统基本概念

1.1. OS的特性

1. **最基本的是**：并发性，共享性(资源的互斥使用以及同时使用)，这两点基于多道程序技术实现
2. 虚拟性：一个物理实体映射为多个逻辑实体
3. **异步性(不确定性)**：多道程序下，每个程序推进的顺序不确定，执行结果也不确定

1.2. OS给用户的接口

1. **命令接口**：供用户控制作业的执行，管理计算机系统，有命令行，GUI，批处理
2. **程序接口**：供程序员使用OS提供的系统调用，来请求操作系统提供服务

1.3. OS的目标/作用/地位

1. 目标：方便性(方便用户使用计算机)，有效性(提高系统资源的利用率)，可扩充性，开放性
2. 作用(资源管理观点)：控制和管理计算机软硬件资源，包括CPU管理，存储器管理，文件管理，设备管理
3. 作用(用户观点)：用户与裸机之间的接口，裸机的扩充机器
4. 地位：是紧挨着硬件的第一层软件，是供其他软件运行的环境

1.4. 三大类操作系统

1. 分时系统：
   1. **分时技术**：通过把CPU时间分成很短的时间片，按照时间片轮流把CPU分给作业使用，使多个用户可以同时使用一台计算机
   2. **特点**：
      - 同时性：多个用户逻辑上共享一台计算机，而微观上是在轮流使用CPU等资源
      - 独立性：各个用户彼此独立，互不干扰地使用一台计算机
      - 及时性：系统对终端用户的请求能在足够快的时间之内得到响应
      - 交互性：采用人机对话方式
2. 实时系统
   1. **含义**：专为处理实时任务而设计的操作系统，能够快速响应请求
   2. **最基本特点**
      - 及时性：要求对外部请求在严格时间范围内做出响应
      - 可靠性
3. 批处理系统的特点
   - 多道：内存中同时存放多个作业，一个时刻只有一个作业运行

- 多道：内存中同时存放多个作业，一个时刻只有一个作业运行
- 成批：用户和作业之间没有交互性。用户不能干预作业的运行
- 系统吞吐量和资源的利用率有所提高

4️⃣ **分时vs实时**：

- 分时系统是为了给用户一个交互式开发运行环境，实时系统则是为特使用途提供专用系统
- 相比分时系统，实时系统的及时性和可靠性更高
- 相比实时系统，分时系统的交互性更高

1.5. 操作系统的两种接口&系统调用

1️⃣ 用户接口(作业级接口)：OS给终端用户的界面，用于与操作系统交互；有命令行，GUI，批处理

2️⃣ 系统调用接口(程序级接口)：OS提供给程序的接口，用于请求操作系统的服务

3️⃣ **系统调用**：

1. 含义：程序请求操作系统服务的接口
2. 分类：分为可中断的调用(如IO)和不可中断的调用(如原子操作)
3. 作用：**扩充机器功能、增强系统能力、方便用户使用**

1.6. 多道程序设计

1️⃣ 含义：在内存中同时存放多道用户作业，使他们都处于执行的开始与结束点之间

2️⃣ 特点：特点是多道，宏观并行，微观串行

3️⃣ 原理：利用了CPU和I/O设备的并行工作能力来提高系统效率的

或者说：让CPU和IO设备并行的技术是多道程序设计，分时技术

4️⃣ 多道程序的基础是：**存储保护与程序浮动；处理器的管理和分配；系统资源的管理和调度**

5️⃣ 好处：提高CPU和系统资源的利用率，增加吞吐量，减少程序响应时间，提高系统的并发性能

1.7. 其他

1️⃣ 系统吞吐量：指系统在单位时间内所完成的总工作量

2. 进程与线程

2.1. PCB

1️⃣ **概念**：是OS管理进程的专门数据结构，常驻内存

2️⃣ **功能**：记录进程的外部特征，描述进程的动态变化，OS用它控制和管理进程，感知进程的存在

4️⃣ 内容：进程标识符，CPU现场，进程调度信息(优先级/时间/时间)，进程控制信息(资源/地址)

3️⃣ **PCB的初始化工作**：初始化进程标识符，初始化处理机状态信息，初始化进程调度和控制信息

2.2. 进程

1 **进程概念**：程序在一个数据集合上的一次动态执行过程，是系统分配资源和调度的基本单位

2 为何引入进程

1. 多道程序下程序并发执行，破坏了程序的封闭性和可再现性
2. 由于资源共享，各个程序间相互制约，导致了动态性，独立性，异步性
3. 程序是静态的不能反映上述特征，需要引入动态的概念来描述系统和用户的活动，即进程

3 进程特性

1. 动态性：有生命周期
2. 并发性：并发执行
3. 独立性：独立获得资源，独立运行
4. 异步性：推进速度未知
5. 结构性：由程序段，数据段，PCB组成

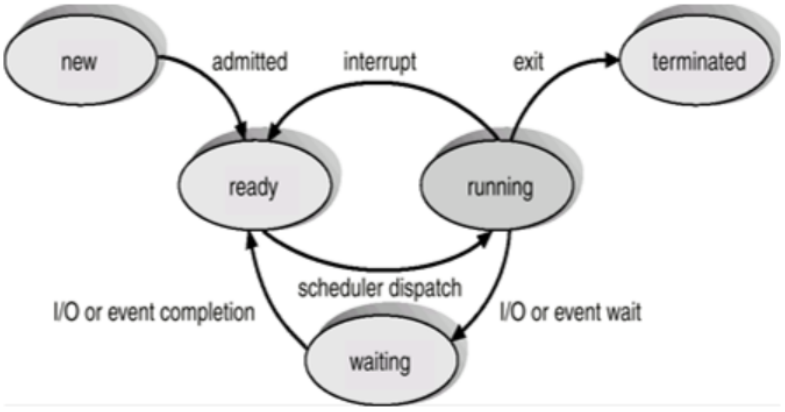
4 进程与程序

1. 进程是程序执行的动态过程，程序是进程运行的静态文本
2. 一个进程可以执行多个程序，同一程序也可能由多个进程同时执行
3. 程序可长期保存，而进程有生命周期
4. 进程是并发实体，程序不是

5 进程与线程

1. 进程是调度和资源分配的单位，线程是调度单位只拥有极少数必须的资源
2. 进程间可并发，同一进程的线程也可并发
3. 线程切换只涉及少量寄存器，开销小得多

2.3. 进程的状态与控制



1 **进入就绪态的进程来自**：创建，运行，阻塞

2 进程控制原理

1. 由OS内核完成，或者说是由OS内核调用原语完成
2. 控制原语：创建/终止、阻塞/唤醒、挂起/激活

2.4. 线程

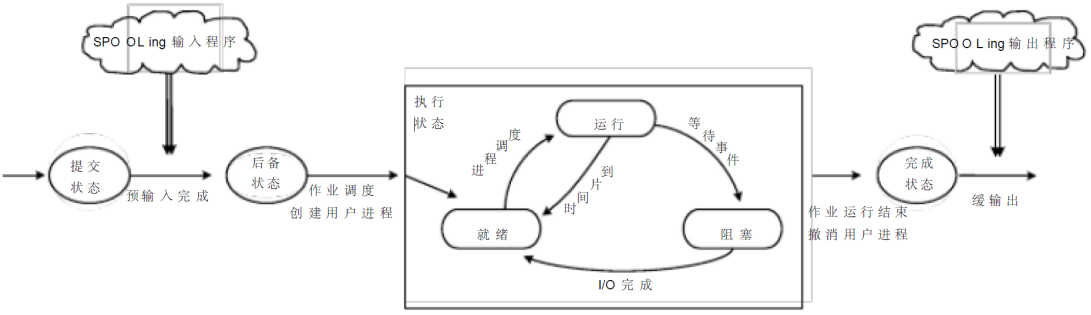
1. **线程概念**：是进程中可调度的一个实体，是处理机调度的基本单位
2. 线程特点：只拥有少量必要资源，同一进程的所有线程共享进程的资源
3. **引入线程的目的**：提高系统效率，提高资源利用率，减少进程并发的开销，使得OS并发性更好
4. 多线程模型：多个/一个/多个用户级线程，映射到一个/一个/多个内核级线程
5. 两种线程：
   1. 用户线程：存在于用户空间，其创建/撤销/切换，不需要OS支持
   2. 内核线程：依赖于内核，其创建/撤销/切换，由内核实现

3. CPU调度(=进程调度)

3.1. 三级调度

1. 高级调度/作业调度：选取外存中后备状态的作业→装入内存/IO后建立进程→进程就绪
2. **低级调度/进程调度**：按照一定的策略，从就绪队列中选择一个特定进程将CPU分配给它
3. 中级调度/交换调度：把内存中阻塞进程交换到外存对换区(挂起)，必要时再调入内存

+作业调度+进程调度



3.2. CPU调度算法

1. 调度算法：
   1. 短作业优先：平均等待最短
   2. **轮转法**：适合分时系统，时间片轮转调度不会让进程饥饿

   +时间片：分时操作系统分配给每个进程微观上的一段CPU时间
   3. 多队列反馈：不同队列不同算法
2. 算法评价的指标
   1. 等待时间：进程在就绪队列中等待CPU的时间
   2. 周转时间：等待时间+进程执行时间
   3. 响应时间：进程请求服务，到首次开始执行的时间差

3.3. 进程的抢占/非抢占调度

1. 概念

- 可抢占：允许系统中断正在运行的进程以启动更高优先级的进程
- 非抢占式：一旦进程开始执行，便会持续运行直到结束或主动放弃CPU

2️⃣ 二者所适用的环境：

- 可抢占：适用于实时系统，便于中紧急情况的处理
- 不可抢占：适用于分时系统，批处理系统

3️⃣ 系统开销：

可抢占方式开销更大，为确保优先级高的进程先执行，需要**频繁进行处理机调度，频繁上下文切换**

3.4. 其他

1️⃣ **所有进程都挂起时，系统不会陷入死锁**，进程挂起不代表其不能执行完

2️⃣ 进程调度的时机：进程完成或异常，进程阻塞，时间片用完，被更高优先级进程抢占

4️⃣ 当一个进程从等待状态变为就绪状态时，不一定会发生CPU调度，CPU发生在就绪到执行

4. 进程同步

4.1. 基本概念与名词

1️⃣ **进程同步/互斥**

1. 同步：异步环境下，互相合作的进程按各自独立的速度向前推进，但在某些确定点上协调工作
2. 互斥：确保多个进程，不会同时访问同一独占型资源

2️⃣ 原语：OS中不可分割的最小功能单位，原语的执行是不能被中断的

3️⃣ **信号量**：一个与队列有关的整型变量，其值表示当前可用资源数/等待该资源的进程数，保证某个代码段不被并发调用，其值只能由**P(申请一个单位资源)/V(释放一个单位资源)**操作改变

4.2. 临界资源&临界区

1️⃣ **临界资源**：一次仅允许一个进程使用的资源，如打印机

2️⃣ **临界区**：在进程中**访问临界资源**的代码

4.3. 管程

1️⃣ 概念：**是一个封装了共享资源及其操作的对象，用于控制对共享资源的访问**

2️⃣ **Note**：管程的互斥是在进程调用其过程时，由OS来保证的

5. 死锁

5.1. 死锁概念

1. 系统的一组进程中，每个进程都占用了某些资源
2. 每个又都在无限等待该组中其它进程释放资源
3. 造成他们都无法向前推进

5.2. 死锁的产生

1️⃣根本原因：系统资源不足，进程推进顺序不合理

2️⃣**死锁的必要条件**

1. 互斥条件：资源仅为一个进程占有
2. 不可剥夺：资源在未使用完之前，不能被其他进程夺走
3. 保持和等待(请求)：进程已经持有了一些资源，同时还在等待其他进程所持有的资源
4. 环路等待

5.3. 动态/静态避免死锁的原理

1️⃣静态预防死锁的原理：

- 对进程申请资源**施加限制**
- 进程开始执行前便申请所需的所有资源，仅当系统满足进程申请要求时才分配资源，进程执行时不申请资源
- 破坏了死锁的占有和等待条件

2️⃣动态避免死锁的原理：

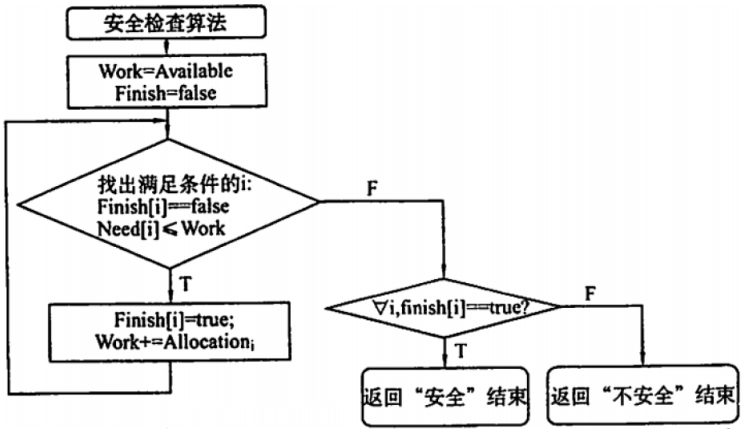
- 对进程发出的资源申请加以**动态检查**，根据检查结果决定是否分配资源
- 银行家算法的原理

5.4. 银行家算法

1️⃣银行家算法思想：

- OS(占有有限资源)当作银行家(占有有限资金)，资源当作周转资金，进程当作借款人
- 银行先借出有限资金满足部分借款人，还款后再借给另一批客户
- 原则是银行家的资金不能被借完

2️⃣安全检查测算法的思想



6. 内存管理

6.1. 内部与外部碎片

1️⃣内部碎片：给进程分配的内存略大于进程实际使用的内存， 从而造成其中一部分内存闲置

2️⃣外部碎片：由于内存空间太小而无法分配给作业的部分内存

6.2. 装入连续内存：分区存储管理

1️⃣单一连续分配：低地址给OS高地址给用户，再其余的浪费掉，有内部碎片

2️⃣静态多分区分配：OS分区+多个用户分区，用户分区大小在装入前预先确定，每个分区装一个程序

3️⃣**动态多分区分配**：作业进入主存时再简历分区，涉及三种分配算法

1. 首次适应：空闲分区链中从头顺序找到第一个大小合适的空闲区，倾向于找到低地址空闲分区
2. 最佳适应：空闲分区链中从小到大找到第一个大小合适的空闲区
3. 最差适应：最佳适应改成从大到小

找到合适空闲区后，劈成两半：和作业一样大的(占用)+剩余部分(空闲)

4️⃣**可重定位分区**：允许分区的物理地址在内存中移动，可以移动现有进程在内存的位置来放置新进程

1. 静态重定位在装入时完成，动态重定位依靠重定位寄存器(提供基址)完成
2. 重定位寄存器：用于存储基址，通过与虚拟地址相加得到逻辑地址
3. 紧凑(碎片拼接)技术：向一个方向移动已分配的作业，碎片就此紧缩在另一端

6.3. 分区的保护

1️⃣界地址寄存器：上界地址寄存器内容 <= 物理地址 <= 下界地址寄存器内容

2️⃣基址+限长寄存器法：物理地址-基址寄存器内容 <= 限长寄存器内容

6.4. 对换技术

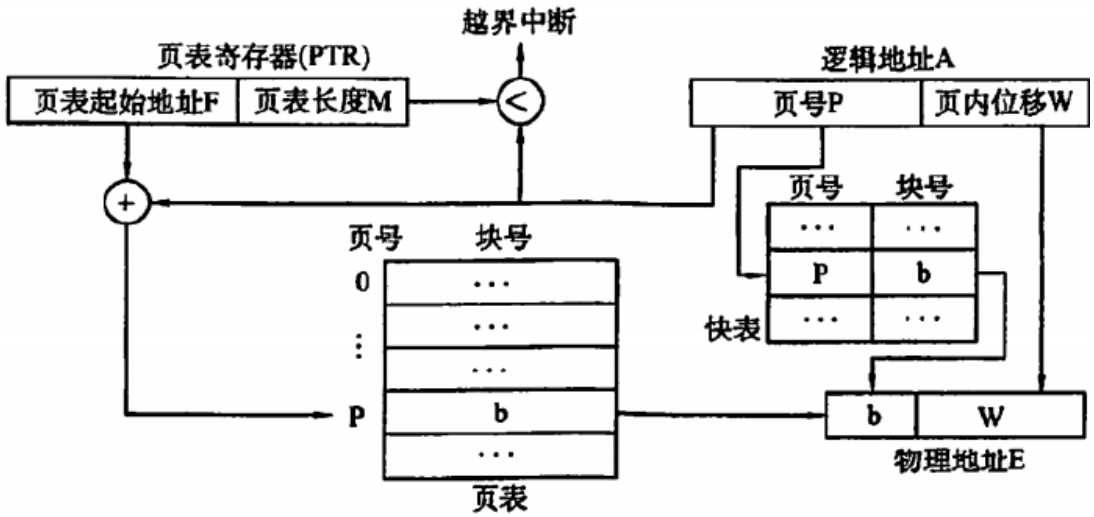
1️⃣含义：把主存中暂时不能运行的进程调出到外存，再将具备运行条件的进程调入内存

2️⃣目的：从逻辑上扩充内存空间 从而使整个系统资源利用率提高

3️⃣分类：整体兑换(进程对换)，部分对换(页面/分段对换)

5️⃣**对换技术的代价**：时间(交换操作需要时间)和空间(需要外存的空间)

6.5. 全部页装入离散内存：基本分页



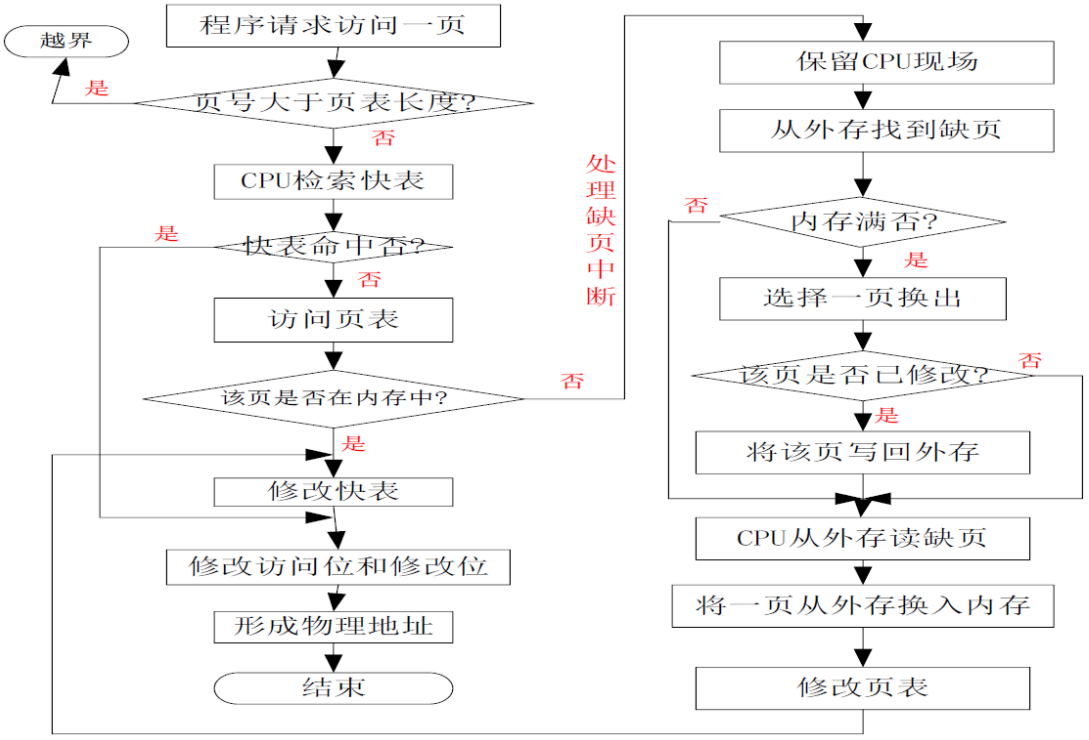
注意

1. 页表起始地址，页表长度这两个内容来自于PCB
2. **快表**：存放被频繁访问的页面的页表项，提高了内存访问速度
3. PS：多级页表，为页表分配大段连续内存→将页表分页，离散地将各个页表存放到内存块中

6.6. 虚拟存储器

1. 理论基础：程序局部性
2. 概念：具有请求调入功能、置换功能，能从逻辑上对内存容量加以扩充的存储器系统
3. 软件支持：建立在离散分配基础上，如请求分页，全球分段
4. 硬件支持：一定容量的内存，较大的外存(且有对换区，这是关键)，缺页中断机构，地址变换机构
5. **虚存的好处(引入原因)**
    1. 从逻辑上扩展内存的空间，是的用户层面能使用到更大的内存空间
    2. 使得作业部分装入内存便可开启运行，使得内存中可以装入更大的作业，也提高了多道程序的性能
6. 虚存的寻址空间由CPU字长觉醒，虚存的实际大小=min{外存大小，寻址空间}

6.7. 请求分页



1. 原理：基本分页基础上，加上请求调页+页面置换，内容如上图
2. 页表新增内容：存在位(是否存在)+访问字段(是否被访问)+修改位(是否被修改)
3. 缺页中断处理程序：完成页面的调入

6.8. 请求分页的页面置换算法

1. 最佳置换算法：已知未来页面访问的顺序，淘汰以后不再使用/最迟被使用的页

2️⃣先进先出

3️⃣最近最久未使用

4️⃣CLOCK算法：基于局部性原理，淘汰不被访问的

1. 每页设置访问位=1代表访问过，所有页构成循环链表
2. 指针遍历循环链表，一路上将所有=1的访问位置零，淘汰所有访问位=0的(不被访问就滚)

5️⃣改进CLOCK：考虑到淘汰未修改的页开销小，所以优先淘汰不被修改的页

1. 增设修改位=1表示被修改过
2. 先试图找(**访问位=0**, **修改位=0**)页替换，若没找到⬇️
3. 再试图找(**访问位=0**, **修改位=1**)页替换，所扫描过之处皆置**访问位=0**
4. 一直找下去一定能找到

6.9. 抖动

1️⃣抖动概念：页面频繁地调入或换出，CPU利用率低下

2️⃣解决方案：

1. 采用局部置换策略：抖动进程不会去抢别的进程的页，不会导致别的进程也抖动
2. 给进程足够的物理块
3. 控制缺页率

6.10. 其他

1️⃣**一条指令有可能引发多次缺页中断，比如一条指令中两个变量都在外存，而普通中断和指令则一一对应**

2️⃣**清理内存指令应该是特权指令，内存清理是指重新分配前释放内存，特权指令是指只能由OS内核执行的指令**

7. 文件管理

7.1. 文件与文件管理

1️⃣**文件**：具有文件名的一组相关信息的集合

2️⃣文件系统：OS中文件管理有关软件，被管理的文件，文件属性的集合

3️⃣文件系统的组成：文件集合(储存有关数据)，目录结构(组织并提供关于系统中的所有文件)

4️⃣**文件管理的任务**：将逻辑文件映射到磁带或磁盘等物理设备上

5️⃣文件的使用：用户通过文件系统提供的系统调用来实施对文件的操作

7.2. 文件逻辑结构

1️⃣无结构文件：流式文件

2️⃣记录式文件

1. 顺序文件：记录定长，顺序存取
2. 索引文件：记录边长，直接存取

3. 索引顺序文件：将顺序文件的记录分组，索引找到组，在组内顺序找到记录

7.3. 文件操作：OPEN和CLOSE

1 引入OPEN/CLOSE的意义：管理文件访问，分配资源，保证数据一致性

2 OPEN操作

1. 检查路径和文件名的有效性，检车用户圈子按
2. 在文件系统中定位文件
3. 分配必要资源，如内存，文件描述符
4. 更新文件系统的状态信息，如打开文件表

3 CLOSE操作

1. 确保所有打开文件的修改都已经写回外存
2. 释放打开文件时分配的资源
3. 更新文件系统的状态信息，如修改文件最后访问的时间，从打开文件表中删除改目录项

7.4. 文件物理结构

1 连续分配：顺序访问，可随机存取，但是有外碎片

2 **链接分配**：

1. 隐式链接：提高了文件存储空间的利用率，只能顺序存取
2. 显式链接：FAT占据空间，但也只能低效随机存取

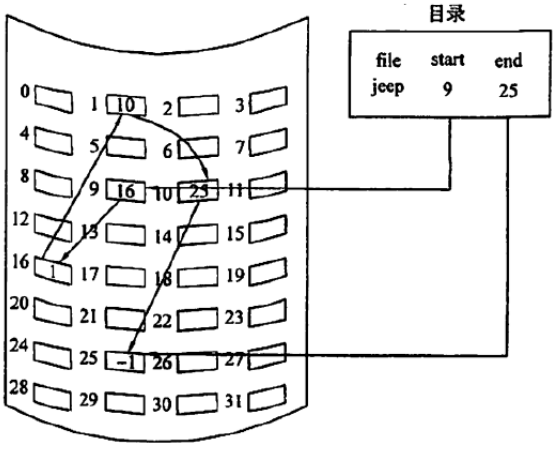


图 4-9 隐式链接

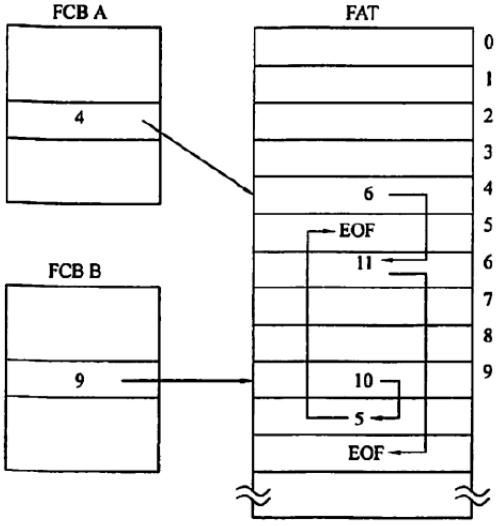


图 4-10 显式链接

3 索引分配：每个文件一张缩引表，指出给文件的所有物理块号和顺序；可高效随机存取，无外部碎片，文件可动态增长

4 混合缩引(UNIX)

1. i结点存储文件的管理信息，和文件名一一对应实现了按名存取
2. i结点有13个地址项：直接地址0-9、1次间址10、2 次间址11、3次间址12

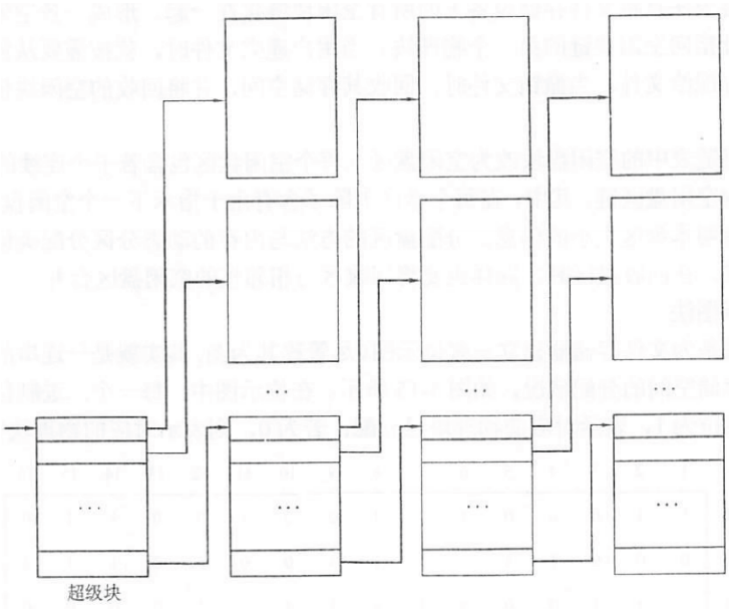
7.5. 空闲存储空间管理

1️⃣空闲表：空闲表的每个表项对应一个空闲区，记录了起始块号+块数，按照首次适应/最佳适应分配

2️⃣空闲链：所用空闲块组织成一个链表

3️⃣**位视图**：图位于外存，图中每一位对应文件存储器中的一个物理块，取值0/1分别代表空闲/占用

4️⃣成组链接法：



1. 结构：100空闲块分一组，每组在前一组第一块记录块数+所有块编号，第一组的这两数据记录在超级块中，每组第一块链成链表，组内多块构成堆栈
2. 分配：检查第一组空闲块数，若有空闲块则超级块中空闲块数-1&第一组栈顶块分给文件，以此类推一块块一组组，但注意如果第一组块用完了超级块会指向第二组
3. 回收：
    - 若第一组不足100块，则直接把回收的块塞进去，然后超级块中放入该块号&空闲块数+1
    - 若第一组已经100块，则**将该块独立建组，然后插入超级块和原来第一组中**，更新超级块和新建组中的空闲块&块号信息

7.6. 目录管理

1️⃣**目录管理目标**：实现按名存取，提供快速的目录查询方法来加快文件检索速度

2️⃣**FCB**：是文件存在的标志，OS通过FCB来管理文件，记录了如下信息

- 文件结构信息：物理/逻辑结构信息
- 文件管理信息：文件名称，大小，长度，属性，建立日期，上次存取日期
- 文件存取控制信息：问价主任的权限，同组用户权限，其他用户权限

3️⃣**文件目录**：FCB的有序集合，每个FCB叫做一个目录项

4️⃣如何提高目录检索效率：将文件名，描述信息分开(如UNIX的iNode)；或者用哈希表

5️⃣目录结构：

1. 单级目录
2. 两级目录：主文件目录，用户文件目录，用户不可建立目录
3. 多级目录：**当前目录**(OS中用户正在操作的文件夹位置)，绝对路径，相对路径

6 目录查询：用户给定路径名→OS查询目录→找到对应FCB或索引结点→找到具体文件

7.7. 其他

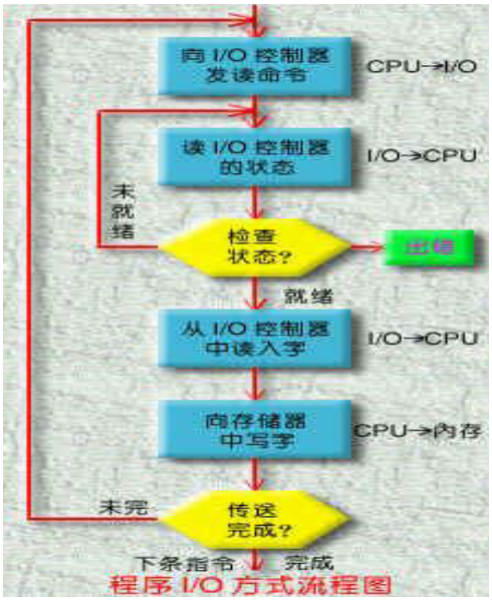
1 **访问文件的方式**：顺序/连续访问，直接/随机访问

2 不同存储介质文件物理结构**必定不相同**

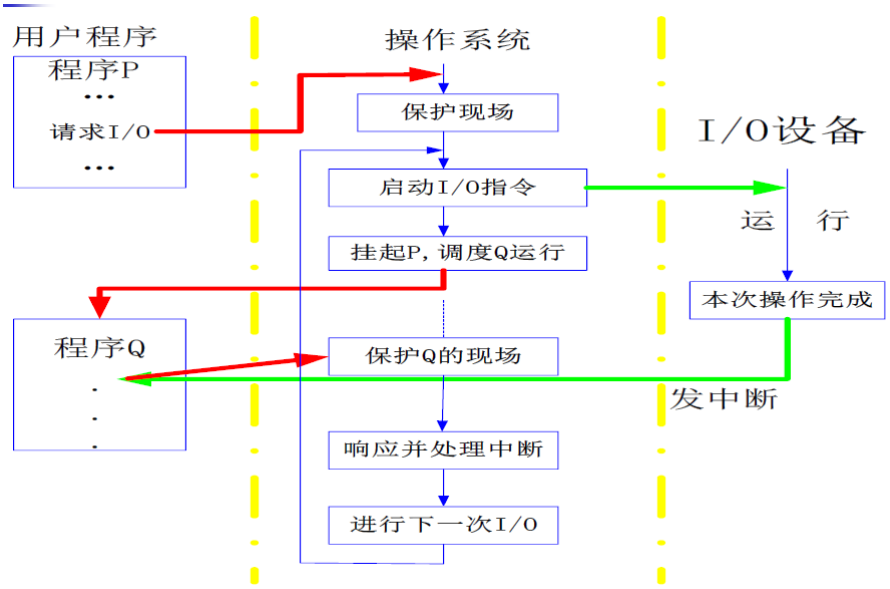
8. 设备管理

8.1. 四种IO控制方式

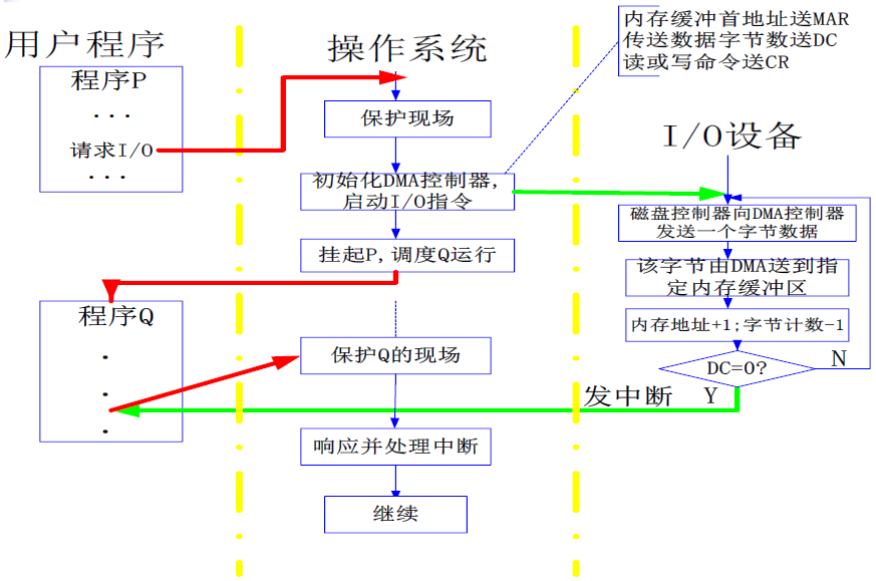
1 轮询/程序IO



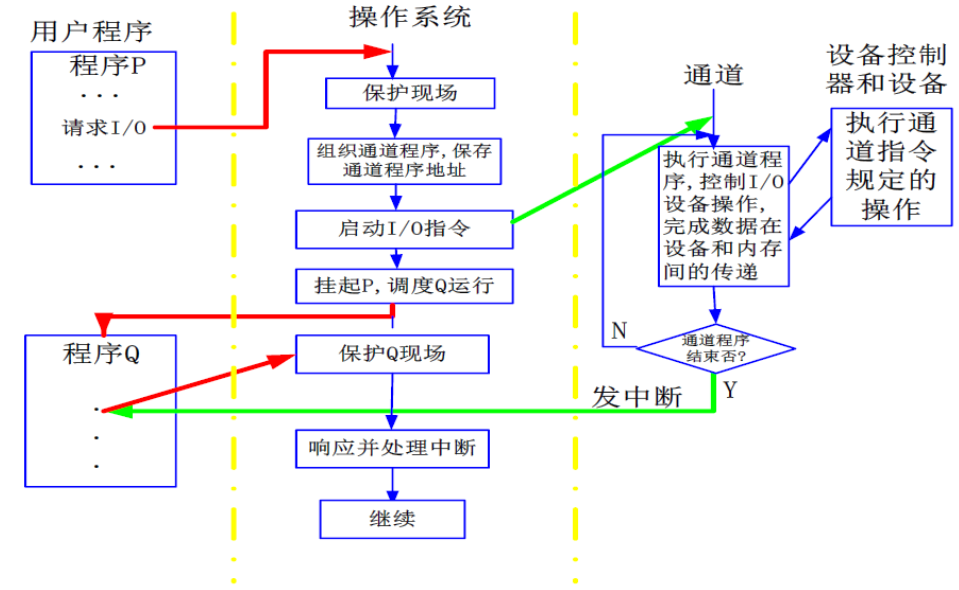
2 中断驱动



3 DMA控制方式



4 通道控制



5 总结

知识点回顾与重要考点

	完成一次读/写的过程	CPU干预频率	每次I/O的数据传输单位	数据流向	优缺点
程序直接控制方式	CPU发出I/O命令后需要不断轮询	极高	字	设备→CPU→内存 内存→CPU→设备	每一个阶段的优点都是解决了上一阶段的最大缺点。 总体来说，整个发展过程就是要尽量减少CPU对I/O过程的干预，把CPU从繁杂的I/O控制事务中解脱出来，以便更多地去完成数据处理任务。
中断驱动方式	CPU发出I/O命令后可以做其他事，本次I/O完成后设备控制器发出中断信号	高	字	设备→CPU→内存 内存→CPU→设备	
DMA方式	CPU发出I/O命令后可以做其他事，本次I/O完成后DMA控制器发出中断信号	中	块	设备→内存 内存→设备	
通道控制方式	CPU发出I/O命令后可以做其他事。通道会执行通道程序以完成I/O，完成后通道向CPU发出中断信号	低	一组块	设备→内存 内存→设备	

8.2. 缓冲管理

1️⃣目的：

1. 缓和 CPU 和 I/O 设备速度不匹配的矛盾
2. 降低CPU中断频率
3. 提高 CPU 和 I/O 设备之间的并行性，从而提高系统的吞吐量和设备的利用率

2️⃣**缓冲池**：由多个缓冲区组成，临时存储IO设备读出或写入的数据，调和CPU和I/O设备间速度不匹配问题

8.3. 设备分配

1️⃣含义：按一定的策略分配设备、控制器和通道

2️⃣**设备分配的数据结构**：**系统**设备表SDT，**设备**控制表DCT，**控制器**控制表COCT，**通道**控制表CHCT

3️⃣**设备独立性**：

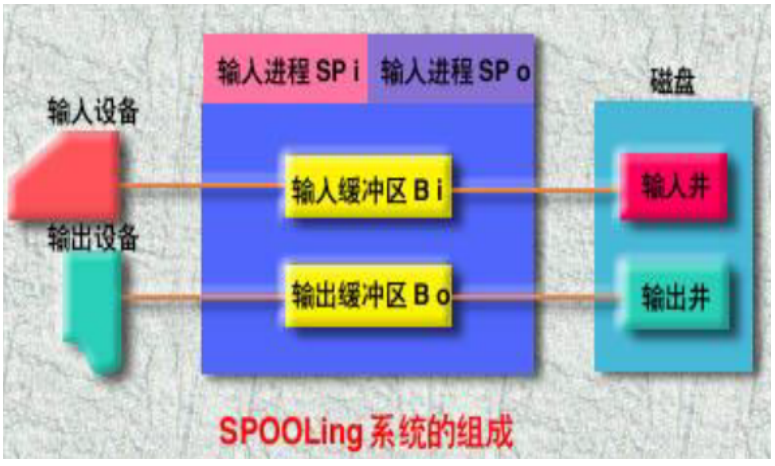
1. 含义：应用程序独立于具体使用的物理设备，它可提高设备分配的**灵活性**和设备的**利用率**
2. 原理：编程时只使用设备的逻辑名，OS执行时将逻辑设备名转化为具体物理设备名，再实施IO

8.4. SPOOLing：假脱机技术

1️⃣**含义**：

1. 多道程序下，利用一道或两道程序来模拟脱机 I/O 中的外围控制机的功能，以达到脱机I/O目的
2. 由此将一台独占物理设备虚拟为多台逻辑设备，从而使该物理设备可被多个进程共享

2️⃣SPOOLing组成：磁盘上的输入/输出井，内存的输入/输出缓冲区，输入进程和输出进程



3️⃣ **虚拟设备**：通过虚拟技术，将一台独占设备虚拟成多台逻辑设备，供多个用户进程同时使用

4️⃣**虚拟打印机实现的原理**

- 进程要求打印输出时，OS分给进程一块磁盘输出井区域
- 进程的输出数据快速存入输出井某区域，输出井该区域此时相当于一台虚拟打印机
- 各进程的打印输出数据在输出井形成一个输出队列
- 由SPOOLing的缓输出程序，依次将输出队列中的数据实际地打印输出

9. 磁盘管理

1 **磁盘调度**：

1. 含义：OS管理硬盘驱动器读写请求的一种技术
2. 目的：优化磁盘访问效率，减少读写头移动时间，平衡请求的响应时间
3. 算法：FCFS，SSTF(最短寻找时间优先)，SCAN，C-SCAN，LOOK，C-LOOK

2 **磁盘调度优化的目标**：使磁盘的平均寻道时间最短

3 **磁盘访问时间**：寻道时间，旋转等待时间，传输时间

4 提高磁盘IO的方法：磁盘高速缓存，提前读，延迟写，优化物理块布局，虚拟盘