目录

[计算机系统概述 2](#_Toc198283059)

[基本知识介绍 2](#_Toc198283060)

[1. 操作系统的功能是什么 2](#_Toc198283061)

[2. 操作系统的特征 2](#_Toc198283062)

[3. 操作系统运行环境 3](#_Toc198283063)

[4. 操作系统结构 4](#_Toc198283064)

[5. 操作系统引导 4](#_Toc198283065)

[6.虚拟机 4](#_Toc198283066)

[进程与线程 5](#_Toc198283067)

[2.1进程与线程 5](#_Toc198283068)

[2.2 CPU调度 7](#_Toc198283069)

[2.3 同步和互斥 8](#_Toc198283070)

[内存管理 10](#_Toc198283071)

[内存管理概念 10](#_Toc198283072)

[内存管理的基本原理 10](#_Toc198283073)

[连续分配管理方式 11](#_Toc198283074)

[基本分页存储管理 12](#_Toc198283075)

[基本分段存储管理 13](#_Toc198283076)

[段页式存储管理 13](#_Toc198283077)

[虚拟存储管理 13](#_Toc198283078)

[虚拟内存的概念 13](#_Toc198283079)

[请求分页管理方式 14](#_Toc198283080)

[页面置换算法 14](#_Toc198283081)

[文件管理 15](#_Toc198283082)

[文件系统基础 15](#_Toc198283083)

[文件的基本概念 15](#_Toc198283084)

[目录 17](#_Toc198283085)

[文件系统 17](#_Toc198283086)

[IO管理 18](#_Toc198283087)

# 计算机系统概述

## 基本知识介绍

这一部分我将介绍操作系统的基础知识

### 操作系统的功能是什么

**操作系统是计算机系统资源的管理者**：包括软件资源和硬件资源

1. 处理机管理：这个主要是和进程与线程调度相关，目的为了实现高并发
2. 存储器管理：负责内存分配回收，地址映射，目录管理及文件读写管理保护
3. 文件管理：文件系统进行管理，包括文件存储空间管理，文件读写保护等
4. 设备管理：完成用户IO请求，包括缓冲管理，设备处理，虚拟设备

对于上述功能你可能感觉到陌生，我们会在后续逐一介绍：对应以下四个章节

**操作系统是用户与计算机硬件之间的接口**

1. 命令接口：方式分为联机控制和脱机控制

联机控制：交互式命令接口，适用于分是或者实时系统的接口，类似于终端

脱机控制：批处理命令接口，脱机命令由一组作业控制命令组成，类似于程序段

1. 程序接口：

由一组系统调用组成，用户通过在程序中使用系统调用来请求操作系统服务

如今较为流行的为GUI（图形用户界面），即图形接口

操作系统是实现了对计算机资源的扩充

### 2. 操作系统的特征

1. 并发

并发是指两个或者多个事件在同一事件间隔发生，实现多道程序交替执行，使CPU处于忙碌状态

1. 共享

即资源共享，指的使系统中的资源可以共多个并发执行的进程共同使用

互斥共享资源：一段事件只允许一个进程访问该资源

同时访问资源：这类资源允许一段时间内多个进程同时访问

上述两个特征是操作系统最基本的特征，两者间互为存在关系

1. 虚拟

虚拟指的是将一个物理上的实体变为若干逻辑上对应物，对用户隐藏，使用户无法感知

1. 异步

多道程序环境允许多个程序并发执行，但是由于资源有限，进程执行不是一贯到底的，以不可预知的速度向前推进，这就是程序的异步性

上述特征十分重要，可以这么记忆，病了就享受（并发与共享不可分），走步都虚

### 3. 操作系统运行环境

#### 3.1处理器运行模式

在操作系统中，通常CPU会执行两种不同性质的模式，一种使操作系统内核程序，另一种是用户自编程序，而指令一般分为两种，一种为特权指令，另一种为非特权指令

特权指令：不允许用户直接使用的指令，如IO，关中断，内存清零等等

非特权指令：允许用户直接使用

那么上述指令的具体实现方式这里讲述一下

操作系统将CPU的运行模式分为用户态和内核态，内核态下就允许使用特权指令

设计一个标识，来标志当前CPU处于什么状态

现代操作系统，几乎都是分层式的结构

大部分操作系统的内核包括以下四方面内容

1. 时钟管理：计时，给用户提供标准的系统时间，还可用于进程切换
2. 中断机制：操作系统各项操作的基础，现代操作系统是靠中断驱动的软件
3. 原语：计算机底层可被调用的共用小程序，具有原子性，操作一气呵成，运行时间短，调用频繁
4. 系统控制的数据结构及处理：就是系统运行过程中的数据结构，如PCB，设备控制块，消息队列，缓冲区等等

时中语系统

#### 3.2中断和异常的概念

中断又称外中断：指的是CPU执行指令外部事件，通常用于IO和时钟中断

异常又称为内中断：指的是来自CPU内部：如程序非法操作，地址越界，运算溢出，虚存缺页等等

外中断分为可屏蔽中断和不可屏蔽中断：看名字也知道什么意思

异常分为故障，自陷，终止：

故障就是指令执行引起的

陷入是用于用户态下转换内核态使用的

终止是出现了CPU无法继续执行的硬件故障

**Note：故障和陷入为软件中断，而终止和外部中断属于硬件中断**

中断和异常处理过程大致如下

1. 执行到第i条指令检测到异常或中断请求
2. CPU打断当前用户程序，赚取处理异常或者中断
3. 在处理完成后，CPU通过执行中断或者异常返回指令，回到第i条指令或者第i+1条继续执行
4. 若中断或者异常处理程序发现本次中断或者异常不可恢复，则终止程序

#### 3.3 系统调用

看了上面那么多，你肯定在疑惑特权指令不允许用户使用，那么用户如何使用操作系统中的各类资源呢，那就是系统调用，一个操作系统提供给应用程序使用的接口，凡是和共享资源相关的，就需要通过系统调用方式向操作系统提出请求，由操作系统代为完成，结果返回用户

也就是说原本我的用户程序是不允许使用IO的，但是我可以通过（打电话）系统调用请老大哥（操作系统）来干这件事，干完后告诉我一下就行

但是CPU只有一个，我这边占着，老大哥就用不了，那么我只能先退下来（得告诉老大哥一声我下来了，它该上了），那么这个通知手段叫做陷入指令，也就是上面提到的软件中断，记住：老大哥很仗义，只有陷入指令才会让其抢夺我的控制权。

### 4. 操作系统结构

这边就简要说一下操作系统都有哪些结构，诸位只需要简单了解即可

1. 分层法

将操作系统分为若干层：底层为硬件，最上层为用户接口，相互之间通过接口互相调用

1. 模块化

将操作系统按照系统功能划分为若干独立的模块

1. 宏内核和微内核

按照内核架构划分为宏内核和微内核

宏内核：将操作系统的主要功能模块都作为一个紧密联系的整体运行在内核态

微内核：只在操作系统中保存最基本的功能，将移出操作系统内核的代码根据分层原理进行划分为若干服务程序，之间相互独立，通过微内核进行通信

那么微内核提供哪些功能

1. 进程管理
2. 低级存储器管理：只负责逻辑地址转为物理地址
3. 中断和陷入处理

### 5. 操作系统引导

1. 激活CPU：第一步肯定是这个，不然你在哪执行程序啊

2. 硬件自检：检测硬件是否故障

3. 加载带有操作系统的硬盘：加载引导扇区

4. 加载主引导记录：主引导记录的作用是告诉CPU去哪找操作系统

5. 扫描硬盘分区表：进行各盘区分

6. 加载分区引导记录：各盘还需要进行分区引导

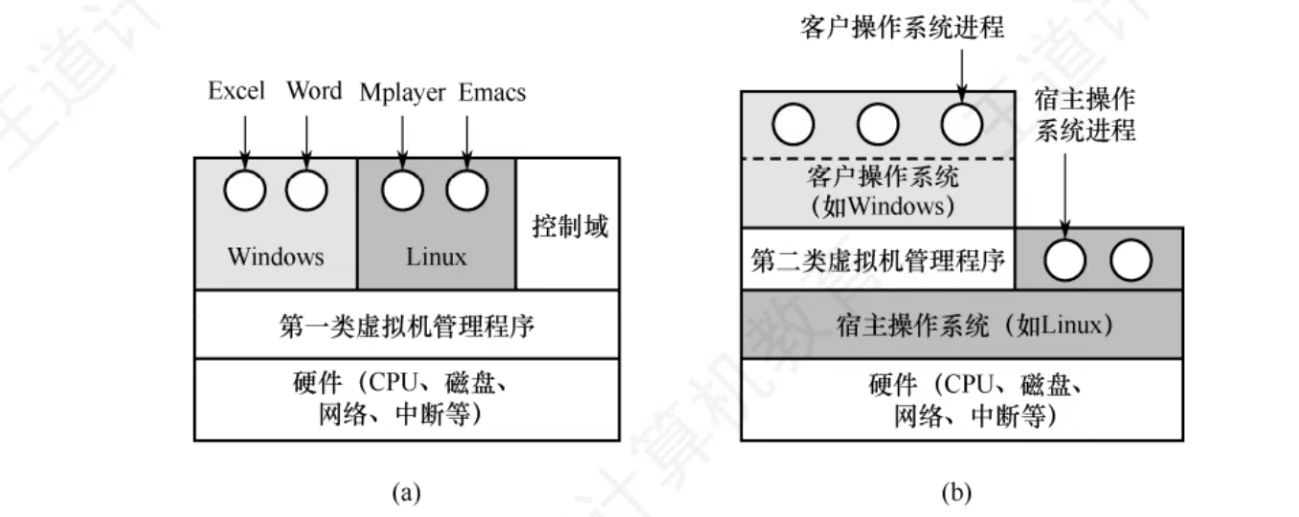
7. 加载启动管理器：启动

8. 加载操作系统：操作系统加载内存

### 6.虚拟机

通过虚拟化技术，将一台物理机器虚拟化为多台虚拟机

主要分为两种：



第一类虚拟机管理程序：

类似于操作系统，运行在特权级别的程序，向用户提供若干虚拟机

第二类虚拟机管理程序

就是目前市面上的一些软件,就是本机操作系统上运行的一个进程：如下



# 进程与线程

这个模块一直是面试的重点

### 2.1进程与线程

进程：为了使得参与并发执行的每个程序独立运行，引入进程控制块的概念，作为专门的数据结构来描述进程的基本情况和运行状态，创建进程转为创建PCB，撤销进程转为撤销PCB

注意程序和进程的区别，进程是动态的，但是程序是静态的

也就是说进程是进程实体的运行过程，是系统资源分配的一个基本单位

（note：系统调度的基本单位是线程，后面介绍）

进程具有四个特征：

1. 动态性
2. 并发性
3. 独立性
4. 异步性：由于进程之间的相互制约，使得进程执行的结果不可预估：故引入信号量和同步机制（后续介绍）

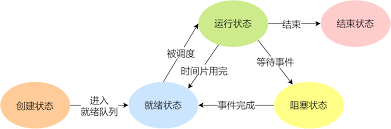
进程的组成：

1. 进程控制块：PCB
2. 程序段：被进程调度程序调度到CPU执行的代码
3. 数据段

进程有五种状态

1. 就绪态
2. 创建态
3. 阻塞态：等待某一事件发生而被阻塞
4. 运行态
5. 终止态

上述基本看名字就知道是什么意思



上述为状态转换图

进程创建

允许一个进程创建另一个进程，创建者称为父进程，被创建者称为子进程，子进程可以继承父进程所有资源，当子进程终止，需要将从父进程获得资源全部还给父进程

（这里会引发两个问题：僵尸进程和孤儿进程）

创建流程一般为

1. 分配进程标识并申请PCB
2. 为进程分配资源
3. 初始化PCB
4. 放入就绪队列

进程之间的通信：这个是面试考察的重点

1. 共享内存
2. 消息传递：也就是信箱（socket）加消息队列
3. 管道：这个分为匿名管道和命名管道
4. 信号：

谈完进程，那就得对线程有一些了解

引入线程是为了减少程序并发执行时候的时空开销

进程与线程的比较:

1. 调度：线程是操作系统CPU调度的基本单位，进程是资源分配的基本单位
2. 并发性：线程和进程都可以并发执行
3. 拥有资源：线程几乎不拥有系统资源（仅仅有一点必不可少，保证独立运行的资源）
4. 独立性：一个进程中的线程对其他进程不可见
5. 系统开销：线程的系统开销远远小于进程

Note：每一个线程都有一个唯一的标识符和线程控制块

### 2.2 CPU调度

在作业提交开始到完成，一般要经历三级别调度：后备队列，就绪队列，阻塞队列

高级调度：根据某种规则从外存上处于后备队列中挑选一个，给他们分配内存，IO设备等必要的资源，并建立相应的进程，使他们获得竞争CPU的权利

中级调度：将暂时不能运行的进程调入到外存等待，称为挂起态，当它们具备运行条件且内存中存在空闲的时候，再次调度回来

低级调度：按照某种算法从就绪队列选择一个进程，将CPU分配给他，进程调度是最基本的调度

调度一般有以下评价指标

1. CPU利用率：CPU有效工作时间/（CPU有效工作时间+CPU等待时间）
2. 系统吞吐率：单位时间CPU完成作业数目
3. 周转时间：为作业完成时间-作业提交时间
4. 带权周转时间：周转时间/作业实际运行时间 这个数大于等于1
5. 响应时间：用户提交请求到系统首次响应所用的时间

进程切换：

切换CPU到另一个进程需要保存当前进程运行状态并恢复另一个进程的状态，这个任务称为上下文切换

流程如下：

1. 挂起一个进程，将进程上下文保存到PCB中，包括程序计数器和其他寄存器
2. 将进程的PCB移入到相应的队列，如就绪，在某事件阻塞等队列
3. 选择另一个进程执行，并更新其PCB
4. 恢复新进程的上下文
5. 跳转到新进程的PCB程序计数器所指向的位置执行

CPU各种调度算法：非常重要：但是非常简单，看名字基本就知道什么意思

1. 先来先服务
2. 短作业优先
3. 优先级调度
4. 高响应比优先：响应比的计算：（等待时间+要求服务时间）/要求服务时间
5. 时间片轮转
6. 多级反馈队列调度算法：这个需要解释一下

设置多个就绪队列，并且为每个队列赋予不同的优先级，第一级最高

赋予每个队列进程的时间片大小各不相同，优先级越高，时间片越小

每个队列采用FCFS算法，如果未在时间片内完成，则放到下面队列

仅当第一级为空的时候，才轮到老二，如果老大又来了，还可以抢老二的

### 2.3 同步和互斥

临界资源：只能同时为一个进程所使用的资源

临界区：访问临界资源的代码

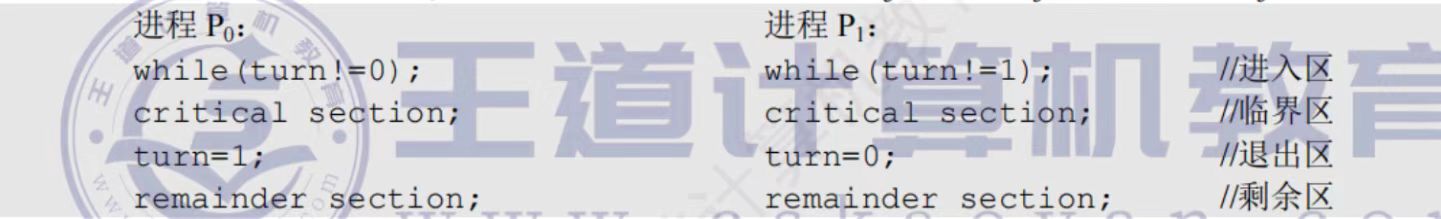
同步指的是直接制约关系，源于进程之间的相互合作

互斥指的是间接制约关系，来源于进程之间的资源竞争

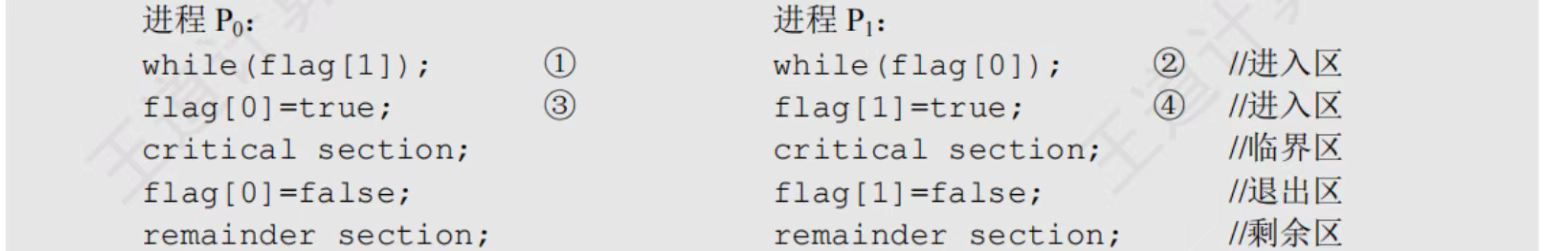
满足以下原则：空闲让进，忙则等待，有限等待，让权等待

软件实现方法：

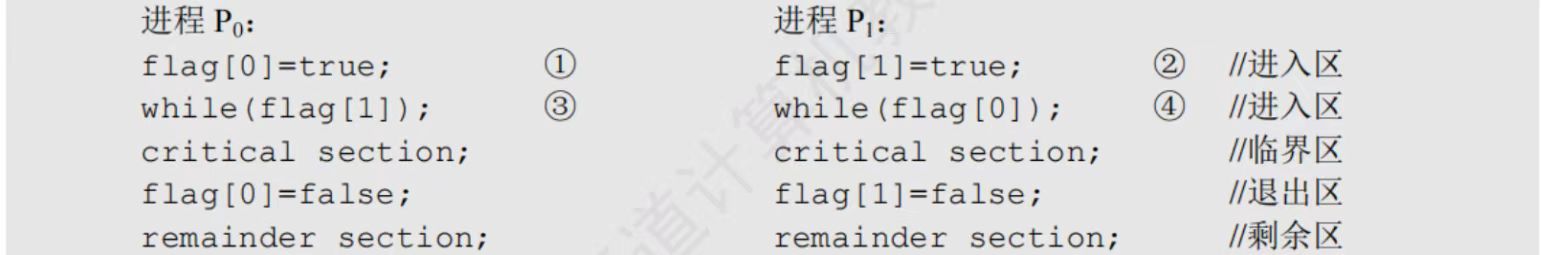
单标志法



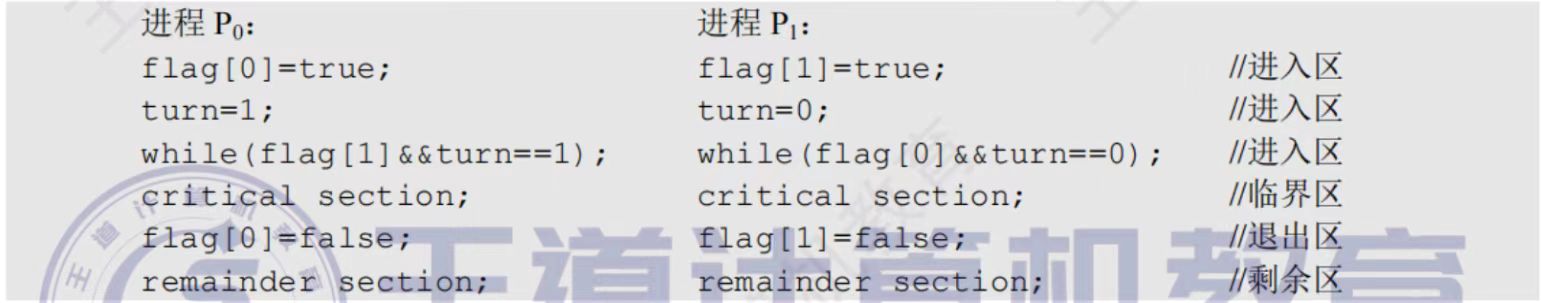
双标志先检查法



双标志后检查法



Peterson算法：皮特森算法：结合了算法1和算法3的思想，在进入临界区的时候，先观察对方是否进入临界区，如果对方请求进入，那就让给他，但是如果对方又让了回来，那么自己就不需要客气了，直接进入临界区即可



信号量：  
也就是传说中的P，V操作，P申请资源，没有就阻塞，V释放资源

Semaphore定义信号量 如：Semaphore wjm = 1；系统中存在wjm资源1个

这里比较重要的是几个常见模型，这个我后面有时间会着重进行补充，单开一篇文章，这里简单进行介绍

1. 生产者消费者问题

生产者和消费者互斥访问缓冲区，生产者生产产品增加资源数量，执行V操作

消费者消耗产品减少资源数量，执行P操作

1. 读者写者问题

读者写者互斥访问，但是多个读者可同时访问，存在一个count计数来标识当前读者数量，读者都结束后才释放对缓冲区的锁，而为了避免写进程饿死，设置一个信号量保证写优先，读者进程进入缓冲区的时候，需要判断是否存在写者进程

1. 哲学家进餐问题

每两个哲学家之间存在一根筷子，哲学家围成一圈就餐

管程：类似于类的概念，资源的分配和释放只能由管程中的函数进行

条件变量：管程中设置多个条件变量，每个条件变量保存了一个等待队列，用于记录因为该条件变量阻塞的队列

死锁：指的是多个进程因为竞争资源而造成的一种僵局

产生原因：进程推进顺序非法，系统资源竞争

条件：

互斥

不可剥夺

请求并保持

循环等待

死锁预防：破坏死锁的四个条件

1. 互斥使用的资源改为共享使用
2. 如果请求新的资源得不到满足，必须释放已有资源
3. 采用预先静态分配方法，一次性分配 全部资源
4. 顺序资源分配法：给资源编号，只有得到小号资源后才可以请求序号更高的资源

死锁避免：银行家算法：只有分配资源后不会使得系统处于不安全状态才会分配资源

死锁检测：资源分配图

死锁接触

撤销进程法

资源剥夺法

进程回退法

# 内存管理

## 内存管理概念

### 内存管理的基本原理

内存管理主要功能包括：

地址转换，内存分配和回收，内存空间的扩充（虚拟存储技术），内存共享，存储保护

逻辑地址和物理地址

编译后，每个模块都从0开始进行编址，这称为目标模块的逻辑地址

Note：进程在运行过程中，看到的和使用的都是逻辑地址

逻辑地址经过地址转换变为物理地址

物理地址指的是内存中的实际地址，内存地址空间是内存中物理单元的合集

地址转换是由MMU（内存管理部件）实现的

程序的链接和装入

将用户程序编程可执行程序，一般需要以下几步

**编译，链接，装入**

装入分为以下几种

1. 绝对装入：

逻辑地址与实际地址完全相同，只适合单道程序环境，在编译时就知道程序存放在内存哪个位置

1. 可重定位装入（静态重定位）

地址转换在进程装入时就完成，缺点是一个作业装入内存，必须给它分配足够的存储空间，且运行期间无法移动

1. 动态运行时装入

看到这个名字，相比你就知道其与上面的区别了，虽然装入模块被装入到内存，但是只有运行的时候才会发生地址转换，存在一个重定位寄存器来实现动态运行时装入‘

然后就是链接：分为静态链接·和动态链接

1. 静态链接

将各目标模块与其所需库函数链接成一个完整的装入模块，以后不在拆分

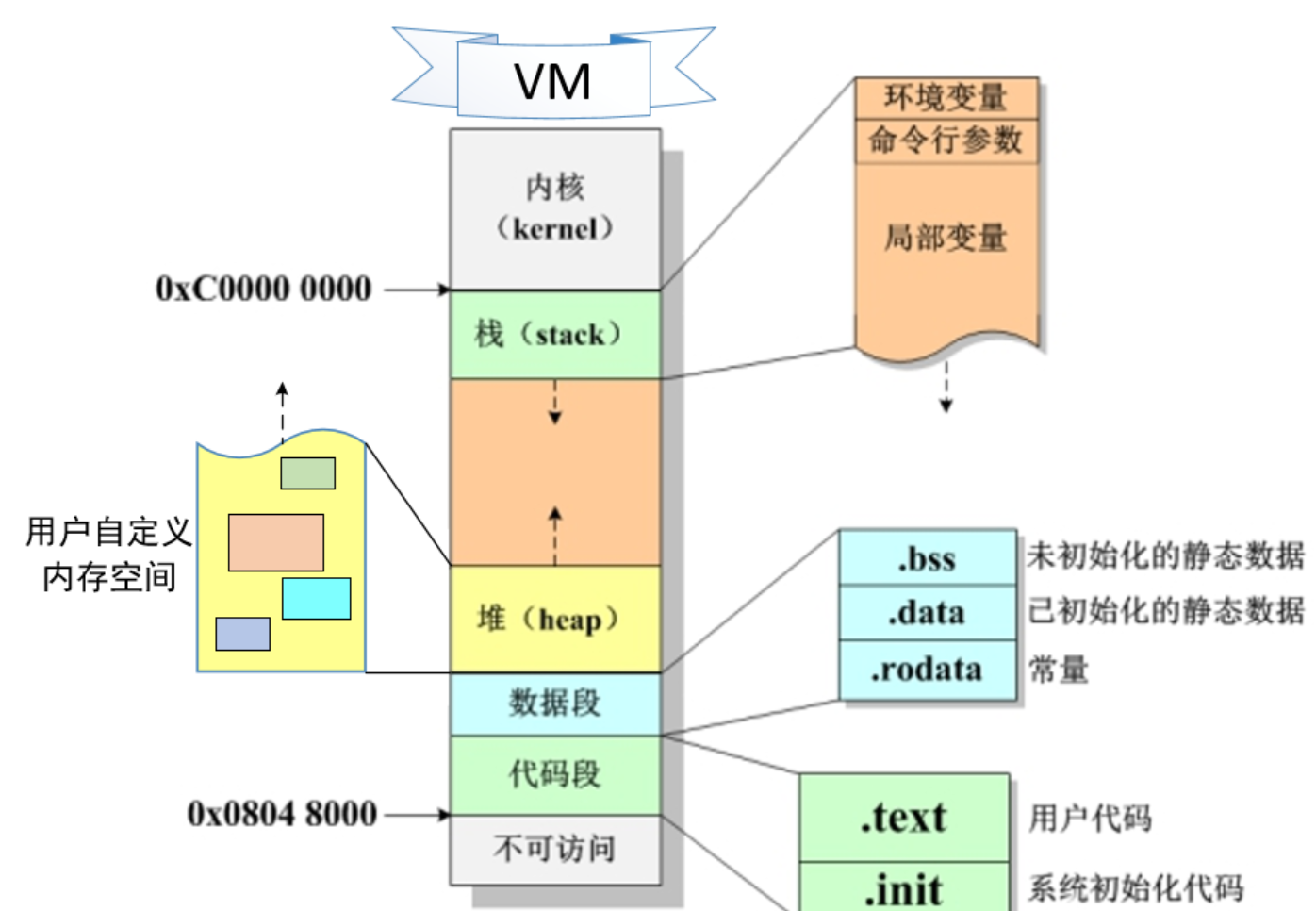
1. 动态链接

采用边装入边链接的方式

1. 运行时动态链接

在程序运行需要某目标模块的时候，才对其进行链接

内存映像



内存映像一般包含：代码段，数据段，堆，栈等基本要素

内存保护

1. 在CPU设置一对上下限寄存器，存放用户进程在主存中的下限和上限地址
2. 采用重定位寄存器和界地址寄存器进行越界检查，重定位寄存器内存放的为初始物理地址，界地址寄存器存放的为进程最大逻辑地址（由于逻辑地址从0开始，所以这就是长度）

内存共享：只有只读区域才可以共享

内存分配和回收

### 连续分配管理方式

指的是为一个用户空间分配一篇连续的内存空间

1. 单一连续分配：用户独占用户区，内存中只有一道程序
2. 固定分区分配：将用户空间划分为多个固定大小的分区
   1. 分区大小相等
   2. 分区大小不等

使用分区使用表来标识各跟去的起始地址，大小和状态

1. 动态分区分配：根据进程需要，动态分配内存

动态分区分配涉及多种分配算法：

1. 首次适应算法：按地址递增次序排列，每次分配，顺序查找
2. 临近适应算法（循环首次适应算法）
3. 最佳适应算法：空闲分区按照容量递增排序，每次顺序查找第一个满足大小的分区
4. 最坏适应算法：空闲分区按照容量递减排序
5. 索引分配算法：
   1. 伙伴系统：所有分区大小伟2的k次幂
   2. 哈希算法：
   3. 快速适应算法：用常用空间大小进行划分

这三种算法的共同特点就是在主存中连续存放用户程序

### 基本分页存储管理

思想：将内存分为若干固定大小的分区，称为页框，页帧，物理块

进程的逻辑地址空间同样分为若干与块大小相等的区域，称为页或者页面

这样就不会产生外部碎片，且只会在进程最后一个不完整的块申请主存块空间的时候，才会产生内部碎片

分页存储的几个基本概念

1. 页面和页面大小  
   进程的逻辑地址空间每个页面有一个编号，称为页号，从0开始，同样的，内存页框也有自己的编号，称为页框号，从0开始，为方便地址转换，页面大小为2的整数次幂。
2. 地址结构
   1. 逻辑地址，每个分页存储管理的逻辑地址由页号和页内偏移量组成
3. 页表  
   为了便于找到每个页面在内存中存放位置，系统为每个进程建立了一张内存映射表，简称页表，进程每个页面对应一个页表项，记录页号和块号
4. 基本地址变换机构  
   主要任务是将逻辑地址转换为物理地址，为了提高地址变换速度，系统中存在一个页表寄存器，存放页表在内存中的起始地址和页表长度，一般单CPU系统只有一个页表寄存器，当进程被调度执行的时候，将页表的起始地址和页表长度放到页表寄存器中。
5. 物理地址的计算  
   页号P = A/L ，就是逻辑地址/页面大小  
   页内偏移两 W = A%L 就是逻辑地址%页面大小  
   先判断是否越界，如果没有越界的话，那么就区找对应页表项，即查询物理块号  
   note：这里的P对应页表项的地址为页表起始地址加上P乘以页表项长度  
   计算物理地址 E = bL \* W，b为对应物理块号  
   note：地址转换是由硬件来完成的

快表（TLB）

具有并行查询能力的高速缓冲寄存器  
由于局部性原理，这里面会存储一些访问过的若干页表项，对快表进行遍历查询的时间几乎可以忽略不计，通过快表命中访存次数就会降低到1次

多级页表

如果页表项较多，但是页表还要求连续存储，这显然不切实际

故为了使得页表离散存储，引入多级页表，对页表在进行分页，然后上一级页表项存储对应页表对应的块号

### 基本分段存储管理

分段管理的方式考虑到了程序员和用户，将用户进程逻辑地址空间划分为若干大小不等的段，并分配一段连续的地址空间

段表：包含段号，段长，本段在主存中的起始地址

分段和分页的区别

1. 页是物理信息的基本单位，段是信息的逻辑单位
2. 页的大小固定，段大小不固定
3. 分页管理的地址空间是一维的，分段管理的地址空间是二维的

段的共享

内存中配置张共享段表，所有共享的段在共享段表中占据一个表项

### 段页式存储管理

其实就是将每段再分为若干页

逻辑地址结构如下

段号 页号 业内偏移量

## 虚拟存储管理

### 虚拟内存的概念

局部性原理：

时间局部性：程序的指令或数据被执行或者访问，不久后可能再次执行或者访问

空间局部性：一旦程序访问了某个存储单元，不久后其附近的存储单元也将被访问

虚拟存储器的定义和特征

在程序执行过程中，仅将程序当前运行要用到的少数页面装入内存，而其余部分暂时留到外存，有需要再调入内存，当内存空间不够时，再将用不到的信息换出内存

上述的方法就叫做页面置换，而系统好像提供了一个比内存大的多的存储器，称为虚拟存储器

特征包括：

多次性：无需一次将作业全部装入内存

对换性：

虚拟性：从逻辑上扩充了内存的容量

Note：虚拟内存的实现需要建立在离散分配的内存管理方式的基础上

### 请求分页管理方式

在该模式下，只需要将当前需要的部分页面调入内存即可，当内存不够的时候，通过页面置换功能将暂时用不到的页面换出外存

页表机制

存储调入页面内存的状态

包括：状态位（是否调入内存），访问字段（一段时间被访问的次数）

修改位（是否被修改过，未被修改的会直接写回外存），外存地址

缺页中断机制

每当要访问的页面不存在于内存的时候，就产生一个缺页中断，请求操作系统的缺页中断处理程序处理，缺页的进程阻塞，放到阻塞队列，如果内存存在空闲页框，就为内存分配一个页框，将所缺页面装入页框中，并修改页面对应表项

如果不存在，就使用页面置换算法进行淘汰

页框分配：

驻留集大小：给进程分配的页框的集合

内存分配策略：

固定分配局部置换：固定物理块，缺页只能从分配给该进程的页面换出

可变分配全局置换：动态物理块，程序运行期间物理块的数量动态增多或者减少

（最好）可变分配局部置换：动态分配物理块，但是只能从分配给该进程的页面换入换出

### 页面置换算法

1. 最佳置换算法  
   选择淘汰的页面是近期不会被使用到的页面，由于无法预测，该算法无法实现
2. 先进先出算法  
   先进来的先出去，没有使用局部性原理，会产生Belady异常
3. 最近最久未使用算法  
   淘汰的页面是最近最长时间没有被使用的算法
4. 时钟置换算法  
   LRU的性能接近OPT（最佳置换）算法，但是实现开销大
   1. 简单CLOCK算法  
      当某页被调入到内存中的时候，其访问位置为1，算法将内存中的页面链接为一个循环队列，并且存在一个替换指针，当某一页被替换后，其会执行被替换页面的下一页，在选择淘汰一个页面的时候，若为0，淘汰，若为1，改为0。
   2. 改进CLOCK算法  
      考虑了修改位和访问位  
      1. 从指针当前位置开始，扫描循环队列，将遇到的第一个访问位为0修改位0的页面换出，没有这样的页面进行第二次扫面，如2  
      2. 寻找访问位0，修改位为1的页面换出，且该过程将访问位置为0，没有执行3  
      3. 重复第1步，第一步再失败重复第二步

实际上，我们考虑b算法，一共有四种状态，（0，0）（0，1）（1，0）（1，1）  
按照b算法，我们按顺序淘汰上面四种状态

工作集：再一段时间内进程访问的页面集合，驻留集一般小于工作集

# 文件管理

## 文件系统基础

### 文件的基本概念

文件是以硬盘为载体的存储在计算机上的信息集合

用户的输入输出是以文件为基本单位的

文件的定义：我们通过自底向上的方式来定义

数据项：文件系统中最低级的数据组织形式，分为基本数据项和组合数据项

记录：是一组相关数据项的集合，用来描述一个对象在某方面的属性

文件：由创建者定义的具有文件名的一组相关元素的集合

文件属性一般包括

名称，类型，创建者，所有者，位置，大小，保护，创建时间和最后一次访问/修改时间

与进程管理一样：文件管理引入FCB 文件控制块来进行文件管理

FCB一般包含以下信息：（与文件相关的信息大多存储在FCB中）

基本信息：文件名称，文件物理位置，文件逻辑结构，文件物理结构

存取控制信息：实现文件保护的

使用信息：文件建立时间，上次修改时间等

索引节点：当文件很多的时候，文件目录会占用大量的盘块，但是在检索过程中，只用到了文件名称，所以在其他描述信息也就不需要调入内存，于是，UNIX系统便采用了文件名和信息分离的方法，使得文件描述信息单独形成一个索引节点的数据结构，inode结点，从而在文件目录中每隔目录项仅有文件名称和对应的索引结点号构成

Note：使用索引结点可以减少开销，降低磁盘IO次数

文件的操作：

创建文件：在外存分配空间，创建一个目录项

删除文件：回收空间，删除目录项

读文件

写文件

文件的打开与关闭

为了避免多次检索目录，大多数操作系统都会设置一个包含所有打开文件信息的打开文件表，每当用户**首次**对某个文件发出请求的时候，调用open系统调用。

所谓打开，其实就是当系统检索到某个目录项的时候，将该目录项从外存复制到内存打开文件表的一个表目中，并将该文件的索引号返回给用户，当用户再次访问该文件的时候，直接通过该索引号（也叫做文件描述符）来访问文件，当文件关闭的时候，调用close系统调用，系统将从打开文件表中删除这一表目

如果是多进程打开和关闭文件，那么在文件系统中每隔打开文件表中的表项就维护一个打开计数器来记录多少进程打开了该文件，如果某个进程关闭文件，那么该计数器减一，如果为0，那么操作系统删除该表项

每个打开文件都具有如下关联信息：

1. 文件指针：系统跟踪上次读写位置作为当前文件位置的指针，这种指针对于打开文件的某个进程是唯一的
2. 文件打开计数
3. 文件磁盘位置：如果修改了文件，那么就需要进行写回操作
4. 访问权限，每个进程对于文件都有各自的访问权限，不可越权

文件的逻辑结构：从用户角度出发看到的文件组织形式

无结构文件：最简单的文件组织形式，由字符流构成的文件，所以也称为流式文件，无结构，访问只能通过穷举搜索

有结构文件：也成为记录式文件，就是由记录构成的文件（记录：是一组相关数据项的集合，用来描述一个对象在某方面的属性），根据各个记录的长度是否相等，分为定长记录和变长记录

可以使用索引文件为变长文件分配索引，使得变长文件可以进行随机查找，但是增加了存储开销

索引顺序文件：相当于分块存储，将变长文件记录文件中的所有记录分为若干组，然后为每个组建立一个索引表项

文件的物理结构：

连续分配：要求每个文件在磁盘上占有一组连续的块

链接分配：

隐式链接：目录项中含义文件第一块的指针和最后一块的指针，每隔文件对应一个链表

显示链接：将用于链接文件各物理块的指针，显示的存放在一张链接表中，该表只设置一张，成为文件分配表，相当于静态链表

索引分配：

单级索引分配

为每隔文件分配一个索引块，将该文件所有的盘块号记录在索引块中，而调入内存的时候，只需要调入索引块即可，而不需要整个FAT（文件分配表）

多级索引分配：类似于多级页表

混合索引（超级块）：分为直接块，一级间址，多级间址

## 目录

FCB的有序集合被称为文件目录，一个FCB就是一个文件目录项，文件目录也是一种数据结构

单级目录结构：整个文件系统只建立一张目录表，每个文件占据一个目录项，当建立一个新文件的时候，必须检索整个目录表，确保没有重名目录

两级目录结构：将文件目录分为主文件目录和用户文件目录，主文件目录记录用户名和对应文件目录所在位置，用户文件目录项记录该用户所有文件的FCB

树形目录结构，将两级目录结构加以推广，就形成了树形目录结构，也就是用路径来访问文件，Linux操作系统就是该目录结构

无环图目录结构：便于共享，维护共享计数器

目录的实现

1. 线性列表
2. 哈希表：需要一些特殊方式避免冲突

文件共享：

基于索引节点的共享方式（硬链接）

在索引节点维持一个引用计数，表示链接到本索引节点的用户目录项数量

利用符号链实现文件共享（软链接）

创建一个LINK类型的新文件L，并将文件写到用户B的目录，来实现B的目录和文件F的链接，文件L中包含被链接文件F的路径，当用户B访问L的时候，就可以根据路径去查询文件F

## 文件系统

文件系统采用层次化架构

从上到下依次是

应用程序

逻辑文件系统：管理文件系统的元数据信息，元数据指的是文件系统的索引结构，管理目录结构

文件组织模块：组织文件及其物理块和逻辑块，将文件的逻辑块地址转化为物理块地址

基本文件系统：向设备驱动程序发送命令，管理内存缓冲区

IO控制：包括设备驱动程序和中断处理程序

设备

文件系统布局

主引导记录（MBR）：位于磁盘0号扇区，用来引导计算机，后面为分区表，分区表给出每个分区的起始和结束地址，MBR做的第一件事就是确定活动分区，读入第一块（引导块）

引导块：MBR执行引导块的程序，启动该分区的操作系统

超级块：包含文件系统的所有关键信息，在计算机启动的时候，超级块被读入内存，：一般包含分区块的数量，块的大小，空闲块的数量：位于引导块后面

然后就是空闲空间管理了

空闲表法：为所有空闲区建立一个空闲表，信息包括第一个空闲盘块号以及空闲盘块数

空闲链表法:包括空闲盘块链和空闲盘区链

位视图法：值为1表示被分配，值为0表示未被分配，每个块用一个二进制数来表示使用情况

成组链接法：第一个盘块记录下一个组的空闲盘块数和本组的空闲盘块号，由各组的第一个盘块链接成一条链，利用栈来实现

# IO管理

IO接口：又称为设备控制器：是CPU和设备之间的桥梁，用来实现设备和计算机之间的数据交换，设备控制器主要由三部分组成

1. IO逻辑：实现对设备的控制，包含以下功能：接收和识别命令，数据交换，标识和报告设备状态，地址识别，数据缓冲，差错控制
2. 设备控制器与CPU接口：数据线，地址线，控制线
3. 控制器和设备的接口：每个接口都可以传输数据，控制和状态

IO端口：IO接口中可以被CPU直接访问的寄存器

数据寄存器：用于缓存从设备送来的输入数据或者从CPU送来的输出数据

状态寄存器，保存设备执行结构或者状态信息

控制寄存器：由CPU写入，以便于启动命令或者更改设备模式

设备的编址

独立编址：IO地址空间与主存地址空间为两个独立地址空间

统一编址：将主存地址空间的一部分分给IO端口进行编址

IO控制方式：

1. 程序直接控制：长轮询，完全由CPU操作
2. 中断驱动：允许设备主动打断CPU的允许并请求服务
3. DMA方式：IO设别与主存之间开辟之间的数据交换通道
4. 通道控制方式：弱鸡的CPU，只需要CPU发送IO指令就可以了

IO软件层次结构

从上到下依次是

用户层软件：产生IO请求，格式化IO，SPOOLing

设备独立型软件：映射，保护，分块，缓冲，分配（设备无关性）

设备驱动程序：与硬件之间相关的命令都由其负责

中断处理程序：进行中断处理的

硬件

阻塞IO和非阻塞IO

阻塞IO：阻塞IO是指用户进程调用IO操作的时候，进程就被阻塞，并且移动到阻塞队列，IO操作完成后，才被唤醒

非阻塞IO：不阻塞进程，但是进程需要不断询问IO操作是否完成

文件系统的组织信息放在磁盘上，这些信息和代码一起组成文件系统

磁盘为共享设备，但是每个时刻只为一个作业服务

磁盘调度的目的是为了缩短寻道时间：还与磁臂移动速度相关

延迟时间和传送时间和磁盘转速有关

启动时间和电气特性有关

处理时间：读取一个扇区后，磁头需要休息一段时间（其存放于磁盘控制器内的缓冲区）

硬盘的操作系统引导扇区，也就是引导扇区，（存放于固定位置）

一般ROM中只存放很小的自举装入程序，通过其可以找到自举程序，随后将其读入内存，完成初始化

而磁盘格式化分为

硬盘出厂-> 低级格式化（磁盘分扇区）->磁盘分区（区分C盘D盘）->高级格式化（又叫逻辑格式化）（初始化位示图，创建文件系统根目录等等）

磁盘调度算法：

最短寻道时间优先算法：找最近的磁道请求进行处理

电梯调度：SCAN

先来先服务：

循环扫描：C-SCAN 这个是到达一端后迅速返回起点

note:SCAN于LOOK如果没有特殊说明：默认移动到最大最小处即可

磁盘地址结构

柱面号，盘面号，扇区号

这里有个计算

单缓冲区：用户为进程的IO请求分配一块缓冲区平均处理时间为Max(C,T)+M

双缓冲区：MAX（C+M，T）