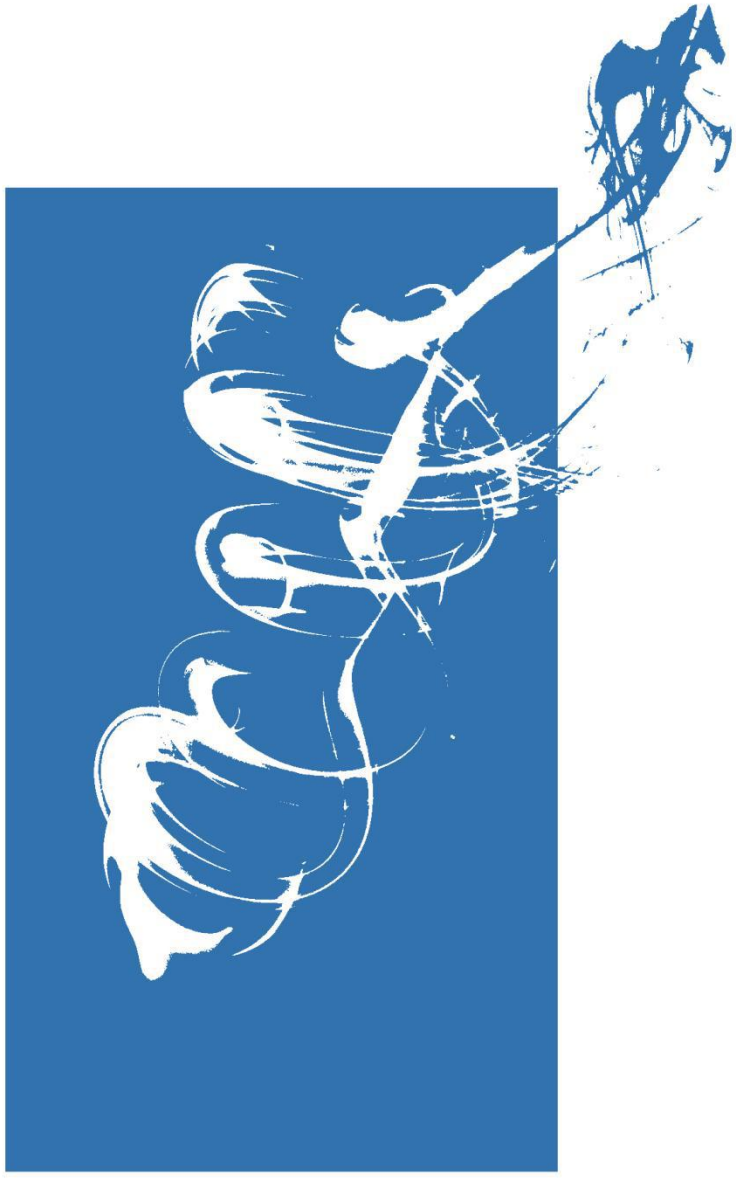
尚德机构™
SUNLANDS



# 密训资料

## 操作系统概论

SUNLANDS MI XUN ZI LIAO

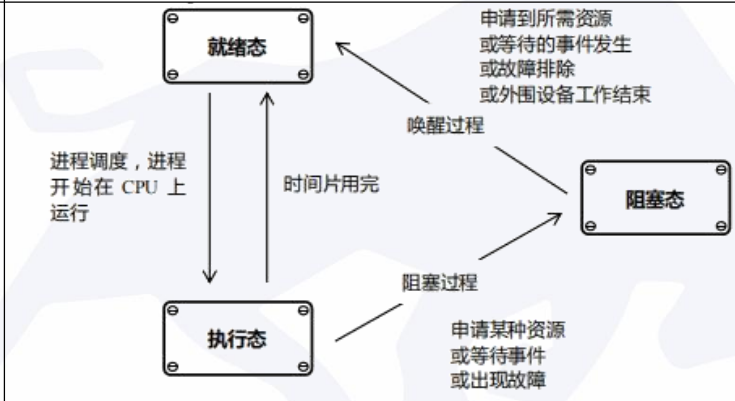
目录

## 第一章 操作系统简介

| 知识点名称 | 知识点内容 | | |
|---|---|---|---|
| 什么是操作系统 ★★★ | 操作系统（OS）是一种复杂的**系统软件**。<br>功能：（1）管理计算机硬件和软件资源（2）提供计算机用户与计算机硬件之间的接口（3）为应用程序的运行提供环境。 | | |
| 操作系统的发展 ★★★ | 无操作系统 | | 因等待人工操作暂停——运行，这样一种不能连续自动工作的状态。 |
| | 单道批处理系统 | | 特点：（1）自动性（2）顺序性（3）单道性<br>优点：减少了等待人工操作的时间。<br>缺点：CPU 资源不能得到充分利用。 |
| | 多道程序系统 | 多道批处理系统 | 特点：（1）多道性（2）无序性（3）调度性（4）复杂性<br>优点：能够提高 CPU、内存和 I/O 设备的利用率和系统吞吐量。<br>缺点：系统平均周转时间长，缺乏交互能力。 |
| | | 分时操作系统 | 特点：（1）多路性（2）独立性（3）及时性（4）交互性<br>优点：向用户提供了人机交互的方便性，使多个用户可以通过不同的终端共享主机。 |
| | 实时操作系统 | | 特点：（1）多路性（2）独立性（3）及时性（4）交互性（5）可靠性<br>用于实时控制和实时信息处理领域。优点：比分时系统要求有更高的可靠性。 |
| 嵌入式操作系统★ | 嵌入式操作系统的特征是小巧、实时性、可装卸、代码固化，弱交互性、强稳定性、接口统一、低能耗。 | | |
| 操作系统的特征 ★★★ | **并发** | | 两个或多个事件在同一时间间隔内发生，多道程序系统可以实现并发执行。 |
| | 共享 | | 指系统中的资源可供内存中多个并发执行的进程共同使用。资源共享有两种方式，即互斥共享和同时共享。 |
| | 虚拟 | | 指通过某种技术把一个物理实体变成若干逻辑上的对应物。 |
| | **异步性** | | 进程以不可预知的速度向前推进。 |
| 操作系统的主要功能 ★★ | （1）**内存管理**（目的：提高内存的利用率）：内存分配、内存保护、地址映射（将逻辑地址变换为物理地址）、内存扩充；<br>（2）**进程管理**：包括进程的描述与组织、进程控制、进程同步、进程通信及进程调度。<br>（3）**文件管理**：文件的读、写管理和存取控制。<br>（4）**设备管理**：主要完成用户的 I/O 请求，为用户分配 I/O 设备。应具有以下功能：1）缓冲管理 2）设备分配 3）设备处理 4）设备独立性和虚拟设备。 | | |
| 操作系统的体系结构 ★★ | 1、发展历程包括：（1）简单的监控程序模型（2）单体结构模型（3）层次结构模型（4）客户/服务器模型与**微内核结构**（5）动态可扩展结构模型<br>2、微内核结构的操作系统的代表有：（1）微软公司研制的 Windows NT；（2）我国自行研制的 COS-IX V2.3；（3）WindRiver 公司研制的 Vxworks；（4）卡内基梅隆大学研制的 Mach 等。 | | |
| 指令周期 ★★ | 一个单一指令需要的处理称为**指令周期**。<br>一个指令周期可以划分成两个步骤：**取指周期和执行周期**。 | | |
| 取指令和执行指令 ★ | （1）**取指令**：在每个指令周期开始时，处理器从存储器中取一条指令。在典型的固定长度指令的处理器中，程序计数器（PC）保存有下一次要取的指令的地址。<br>（2）**执行指令**：取到的指令被放置在处理器的指令寄存器（IR）中。指令中包含确定处理器将要采取动作的位，处理器解释指令并执行要求的动作。 | | |

## 第二章 进程管理

| 知识点名称 | 知识点内容 |
|---|---|

| | |
|---|---|
| 程序的顺序执行★ | 先进入内存的程序先执行，在一个程序执行完毕之前，不能执行其他程序。<br>特点：（1）顺序性；（2）封闭性；（3）可再现性。 |
| 程序的并发执行★★ | 指在同一时间间隔内运行多个程序。<br>特点：（1）间断性；（2）失去封闭性；（3）不可再现性。 |
| 进程定义★ | 定义 1：进程是允许并发执行程序在某个数据集合上的运行过程。<br>定义 2：进程是由正文段、用户数据段及进程控制块共同组成的执行环境。 |
| 进程特征★★★ | （1）并发性（2）动态性（3）独立性（4）异步性（5）结构特征。 |
| 进程和程序的比较★ | 区别：1）进程是动态的，程序是静态的；2）进程是暂时的，程序是永久的。<br>联系：1）进程总是对应至少一个特定的程序。2）一个程序可以对应多个进程。 |
| 进程控制块★★ | 应用程序对应的进程由**程序、用户数据**和操作系统管理进程所需要的**进程控制块**构成。是操作系统感知进程存在的唯一标志。 |
| 进程控制块中的信息★★ | 进程标识符信息：进程标识符用于唯一标识一个进程。<br>**处理机状态信息**：包括通用寄存器、指令计数器、程序状态字 PSW 和用户栈指针。<br>进程调度信息：进程状态信息、进程优先级和进程调度所需的其他信息。<br>进程控制信息：程序和数据的地址、进程同步和通信机制、资源清单，以及链接指针。 |
| 进程的 3 种基本状态★★ | **就绪态**：在多任务系统中，可以有多个处于就绪态。<br>**执行态**：单 CPU 系统中，任意时刻只能有一个进程处于执行态。有 N 个 CPU 的多 CPU 系统中，任意时刻系统中最多有 N 个进程处于执行态。<br>**阻塞态**：处于阻塞态的进程数量可以有很多。 |
| 进程状态的转换★★★ |  |
| 进程的组织★ | 链接方式：具有相同状态的进程的进程控制块用其中的链接字链接成一个队列。<br>索引方式：根据所有进程的状态，建立几张索引表，索引表的每一个表项指向一个 PCB 的物理块。<br>进程队列：把具有相同状态的进程放在同一个队列中，具有不同状态的进程就形成了不同的进程队列。 |
| 进程的创建★★ | 条件：（1）用户登录；（2）作业调度；（3）提供服务；（4）应用请求。<br>过程：（1）申请空白 PCB；（2）为新进程分配资源；（3）初始化进程控制块；（4）将新进程插入就绪队列。 |
| 进程的阻塞★ | 条件（1）请求系统服务；（2）启动某种操作；（3）新数据尚未到达；（4）无新工作可做。<br>过程：（1）将进程的状态改为阻塞态；（2）将进程插入相应的阻塞队列；（3）转进程调度程序，从就绪进程中选择进程为其分配 CPU。 |
| 进程的唤醒★ | 过程：（1）将进程从阻塞队列中移出；（2）将进程状态由阻塞态改为就绪态；（3）将进程插入就绪队列。 |
| 进程的终止★ | 条件：（1）当进程正常执行完毕，调用终止进程的系统调用，请求操作系统删除该进程；（2）一个进程调用适当的系统调用，终止另外一个进程。<br>过程：（1）从进程 PCB 中读进程状态；（2）若进程正在执行，则终止进程的执行；（3）若进程有子孙进程，在大多数情况下需要终止子孙进程。（4）释放资 |

| | |
|---|---|
| | 源。（5）将终止进程的 PCB 移出。 |
| 操作系统内核<br>★★★ | 定义：是计算机硬件的第一次扩充，内核执行操作系统与硬件关系密切，执行频率高的模块，常驻内存。<br>功能：<br>（1）支撑功能：中断处理、时钟管理和原语操作。<br>（2）资源管理功能：进程管理、存储器管理和设备管理。 |
| 中断<br>★★ | 1、定义：是改变处理器执行指令顺序的一种事件。<br>2、过程：（1）系统关闭中断，保护断点（2）转中断处理程序（3）执行中断处理子例程（4）恢复现场，开中断。<br>3、中断子程序的入口地址相关信息在内存中的地址=idtr 中的地址+8×中断向量的值。 |
| 时钟的重要性★ | 时钟是计算机系统的脉搏。 |
| 计算机系统中的时钟★ | 实时时钟 RTC→BIOS→OS 时钟 →应用程序 |
| 操作系统的时钟机制<br>★★ | OS 时钟管理硬件（可编程间隔定时器 PIT）：主要由 3 部分构成：晶振、计数器和保持寄存器。 |
| | 时钟软件——时钟驱动程序，也称为时钟中断处理程序。 |
| 什么是系统调用★ | 是一群预先定义好的模块，它们提供一条管道让应用程序或一般用户能由此得到核心程序的服务。系统调用是系统程序与用户程序之间的接口 |
| 系统调用与一般函数的区别<br>★★★ | 用户态执行：用户空间是指用户进程所处的地址空间，一个用户进程不能访问其他进程的用户空间，只有系统程序才能访问其他用户空间。当 cpu 执行用户空间的代码时，称该进程在用户态执行。<br>系统态执行：系统空间是指含有一切系统核心代码的地址空间，当 CPU 执行系统核心代码时，称进程处于系统态执行。 |
| 进程同步的基本概念★<br>★★ | 进程同步有两个任务：（1）对具有资源共享关系的进程，保证诸进程以互斥的方式访问临界资源。（2）对具有相互合作关系的进程，保证相互合作的诸进程协调执行。<br>临界资源是必须以互斥方式访问的共享资源。<br>临界区是进程中访问临界资源的那段代码。 |
| 同步机制应遵循的准则<br>★★★ | （1）空闲让进（2）忙则等待（3）有限等待（4）让权等待 |
| 记录型信号量机制<br>★★★ | 设某一临界区对应的记录型信号量 mutex<br>mutex.value>=0 时，值表示资源数量。<br>mutex.value<0 时，表示资源分配完毕。其绝对值表示阻塞队列的进程个数。 |
| 管程的基本概念★ | 是描述共享资源的数据结构和在数据结构上的共享资源管理程序的集合。其中包括：变量的定义、变量的初始化代码，以及管理共享资源的过程 |
| 管程的应用<br>★★ | 需建立的管程名为 PC，其中包括了两个过程：en- ter(item)过程，生产者进程调用该过程向缓冲池中投放消息；另一个是 remove(item)过程， 消费者进程调用该过程从公共缓冲池中取消息。 |
| 进程通信<br>★★ | 进程间通信的基本方式：共享存储器系统、消息传递系统、管道通信、消息缓冲队列。<br>进程之间的**高级通信机制**分为：（1）共享存储器系统；（2）消息传递系统；（3）管道通信系统。 |
| 共享存储器系统★ | 相互通信的进程共享某些数据结构或共享存储区。分为基于共享数据结构的通信方式；基于共享存储区的通信方式。 |
| 消息传递系统★ | 进程间通过操作系统提供的一组通信程序传递格式化的消息。分为直接通信方式和间接通信方式。 |
| 管道通信★ | 发送进程通过管道（连接读写进程的一个特殊文件）把数据以字符流的形式发送 |

| | |
|---|---|
| | 给接收进程。 |
| 消息缓冲队列★ | 通过消息缓冲区（数据结构）、发送原语和接收原语通信。消息缓冲队列机制广泛用于本地进程之间的通信。 |
| 线程 ★★★ | 在支持线程的操作系统中，线程是进程的一个实体，**线程是被系统独立调度和分派的基本单位；进程是资源分配的基本单位。** |
| 线程的概念和分类★ | （1）用户级线程（2）内核级线程 |

## 第三章 进程调度与死锁

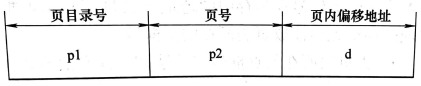
| 知识点名称 | 知识点内容 |
|---|---|
| 进程调度的功能★ | 按照某种策略和算法从就绪态进程（在 Linux 中是可执行进程）中为当前空闲的 CPU 选择在其上运行的新进程。 |
| 进程调度的时机★ | 当一个进程运行结束（包括正常结束和异常结束）、进程阻塞、中断返回、在支持抢占式调度的系统中有比当前运行进程优先级更高的进程到来、当前运行进程的时间片用完等。 |
| 选择调度方式和算法的若干准则★ | （1） 周转时间短 ；（2） 响应时间快 ；（3） 截止时间的保证 ；（4） 系统吞吐量高 ；（5） 处理机利用率好 。 |
| 周转时间短 ★★★ | **周转时间**：指从作业被提交给系统开始，到作业完成为止的这段时间间隔。它包括 4 部 分时间：作业在外存后备队列上等待调度的时间，进程在就绪队列上等待进程调度的时间，进程在 CPU 上执行的时间（服务时间 Ts），以及进程等待 I/O 操作完成的时间。<br>**平均周转时间**：如果系统中有 n 个作业，系统的平均周转时间 T 等于 n 个作业的周转时间之和除以 n，其公式为：$T=\frac{1}{n}\left[\sum_{i=1}^{n}T_i\right]$<br>**带权周转时间**：W 是作业的周转时间 T 与系统为它提供的服务时间 Ts 之比，即 W=T/Ts。n 个作业的平均带权周转时间表达式为：$W=\frac{1}{n}\left[\sum_{i=1}^{n}\frac{T_i}{T_s}\right]$ |
| **先来先服务调度算法（FCFS）** ★★ | 在进程调度中，FCFS 就是从就绪队列的队首选择最先到达就绪队列的进程，为该进程分配 CPU。<br>优缺点：FCFS 适合长进程，不利于短进程。有利于 CPU 繁忙型进程，不利于 I/O 繁忙型进程。 |
| **短进程优先调度算法（SPF）** ★★★ | 从就绪队列中选择估计运行时间最短的进程，将处理机分配给它，使它立即执行并一直执行完成，或发生某事件而被阻塞放弃处理机时，再重新调度。<br>优点：能有效降低进程的平均等待时间，提高系统的吞吐量。<br>缺点：对长进程不利；不能保证紧迫进程的及时处理；不一定能真正做到短进程优先。 |
| **优先权调度算法** ★★★ | 系统将 CPU 分配给就绪队列中优先权值最高的进程。<br>非抢占式优先权调度算法：有高优先权进程到来，系统也不能剥夺当前进程的 CPU 使用权，高优先权进程只能先进入就绪队列。<br>抢占式优先权调度算法：如果新到达进程的优先权高于当前正在运行进程的优先权，那么系统会抢占 CPU，把它分配给新到达的高优先权进程，而正在执行的低优先权进程暂停执行。<br>问题：低优先权进程无穷等待问题，可通过老化技术解决。 |
| **时间片轮转调度算法（RR）** ★★ | 在现代分时系统中广泛使用。<br>系统将所有的就绪进程按先来先服务的原则，排成一个队列，每次调度时把 CPU 分配给队首进程，并令其执行一个时间片。当时间片用完时，调度程序终止当前进程的执行，并将它送到就绪队列的队尾。<br>时间片大小的确定： |

| | | |
|---|---|---|
| | 系统响应时间为 T，进程数目为 N，时间片为 q，有 T = Nq<br>（1）时间片的大小与响应时间成正比；（2）时间片的大小就与系统允许的最大进程数成反比。（3）系统的处理能力。 | |
| **多级队列调度**★ | 将就绪队列分成多个独立队列，根据进程的某些属性，如需要占用的内存大小等，进程会被永久地分配到一个队列。 | |
| **多级反馈队列调度**<br>★★ | 在采用多级反馈队列调度的系统中建立多个优先权不同的就绪队列，为每个队列赋予大小不同的时间片。 | |
| 提供必要的调度信息<br>★★★ | 就绪时间、开始截止时间和完成截止时间、处理时间、资源要求、优先级。 | |
| 采用抢占式调度机制★ | 抢占式调度算法根据抢占 CPU 的时机不同，可以分为基于时钟中断的抢占和立即抢占。 | |
| 具有快速切换机制<br>★★ | 应具有的能力（1）对外部中断的快速响应能力：要求系统具有快速的硬件中断机构，还应使禁止中断的时间间隔尽可能短。（2）快速的进程切换能力：应使系统中的每个运行功能单位适当地小，以减少进程切换的时间开销。 | |
| 最低松弛度优先（LLF）算法<br>★★★ | 松弛度用来表示一个实时进程的紧迫程度。松弛度越小，进程的优先级越高，越优先获得处理机。<br>如果一个进程的完成截止时间为 T，当前时间为 Tc，处理完该任务还需要的时间为 Ts，则松弛度 L 的计算式表示为：L = T - Tc - Ts | |
| 多处理器系统（MPS）的类型、进程分配方式<br>★★★ | 根据处理器的耦合程度 | 紧密耦合的多处理器系统：共享主存储器系统和 I/O 设备。 |
| | | 松弛耦合的多处理器系统：每台计算机都有自己的存储器和 I/O 设备，每一台计算机都能独立工作。 |
| | 根据处理器结构是否相同 | 对称多处理器系统：属于同构的多处理器系统，各处理单元在功能和结构上都是相同的。<br>**进程分配方式：静态分配、动态分配** |
| | | 非对称多处理器系统：有多种类型的处理单元，它们的功能和结构各不相同。其中只有一个主处理器，有多个从处理器。<br>**进程分配方式：主—从式** |
| **自调度**<br>★ | 最常用的调度方式之一，也是最简单的一种调度方式。<br>定义：在系统中设置有一个公共的就绪队列，任何一个空闲的处理器都可以自行从该就绪队列中选取一个进程或者一个线程运行。<br>优点：易移植和有利于提高 CPU 的利用率。<br>缺点：瓶颈问题、低效性和线程切换频繁。 | |
| **成组调度**<br>★ | 由系统将一组相互合作的进程或线程同时分配到一组处理器上运行，进程或线程与处理器一一对应。<br>优点是减少线程切换和减少调度开销。时间分配有两种方式：面向所有的应用程序平均分配处理器时间和面向所有的线程平均分配处理器时间。 | |
| **专用处理器分配**<br>★ | 在一个应用程序执行期间，专门为该应用程序分配一组处理器，每个线程一个，这组处理器供该应用程序专用，直至应用程序完成。<br>优点：加速了应用程序的运行速度和避免了进程切换。缺点：会造成处理器资源的严重浪费。 | |
| 产生死锁的原因和必要条件<br>★★ | 死锁定义：由于多个进程竞争共享资源而引起的进程不能向前推进的僵死状态称为死锁<br>原因：竞争共享资源且分配资源的顺序不当。<br>产生死锁的必要条件：（1）互斥条件（2）请求和保持条件（3）不剥夺条件（4）环路等待条件 | |
| 死锁的预防<br>★★★ | 摒弃请求和保持条件 | ①所有进程执行前要一次性地申请在整个运行过程中所需要的全部资源。<br>②对某些进程在申请其他资源前要求该进程必须释放已经分 |

| | | |
|---|---|---|
| | | 配给它的所有其他资源。 |
| | 摒弃不剥夺条件 | 一个已保持了某些资源的进程，当它再提出新的资源要求而不能立即得到满足时，必须释放它已经保持的所有资源。 |
| | 摒弃环路等待条件 | 进程必须按规定的顺序申请资源。 |
| 死锁的避免 ★★★ | 当系统能找到一个进程执行序列，使系统只要按此序列为每个进程分配资源，就可以保证进程的资源分配和执行顺利完成，不会发生死锁时，称系统处于安全状态。若系统不存在 这样的安全序列，则称系统处于不安全状态。<br>系统处于安全状态，死锁就不会发生。系统进入不安全状态，便可能进入死锁状态。<br>公式：设系统有一类数量为 M 的独占性资源，系统中 N 个进程竞争该类资源，每个进程对资源的最大需求为 W。当 M、N、W 满足 N（W-1）+1 <= M 时，系统处于安全状态。 | |
| 银行家算法 ★★★ | 银行家算法是一种能够避免死锁的资源分配算法。能保证至少有一个进程可得到需要的全部资源而执行到结束，然后释放资源。其实质是避免系统处于不安全的状态。<br>过程：①进行资源试分配；②对试分配后系统的状态做安全性检测。<br>还需要的数量 need[]=最大需求 max[]-已分配 allocation[]。 | |
| 死锁的检测 ★★ | 死锁定理为：S 为死锁状态的充分条件是当且仅当 S 状态的资源分配图是不可能完全简化的。 | |
| 死锁的解除 ★★ | 进程终止：终止所有进程或一次只终止一个处于死锁的进程，直到死锁解除。<br>资源抢占：逐步从进程中抢占资源给其他进程使用，直到死锁环被打破为止。 | |

## 第四章 内存管理

| 知识点名称 | 知识点内容 | |
|---|---|---|
| 内存管理★ | 目标：（1）实现内存分配、内存回收等基本内存管理的功能；<br>（2）提高内存空间的利用率和内存的访问速度。 | |
| 局部性原理 ★ | （1）时间局部性：如果程序中的某条指令一旦执行，则不久后该指令可能再次执行。<br>（2）空间局部性：一旦程序访向了某个单元，在不久之后，其附近的存储单元也将被访问。 | |
| 程序的链接 ★ | 静态链接 | 静态链接是在程序运行前，用链接程序将目标模块链接成一个完整的装入模块。静态链接程序的任务一是对逻辑地址进行修改，二是变换外部调用符号。<br>优点：运行速度较快；缺点：内存开销大，程序开发不灵活。 |
| | 动态链接 | 可将某些目标模块的链接推迟到这些模块中的函数被调用执行时才进行。优点：节省内存和外存空间，方便了程序开发。 |
| 程序的装入 ★★★ | 绝对装入方式 | 按照装入模块的物理地址将程序和数据装入内存。 |
| | 可重定位装入方式（静态重定位） | 在程序装入时对目标程序中的指令和数据地址的修改过程称为重定位。 |
| | 动态运行时装入（动态重定位） | 一个进程在被换出之前所在的内存位置与后来被从外存重新调入内存时所在的内存位置不同，在这种情况下，地址映射必须延迟到进程执行时再进行，把这种装人方式称为动态运行装入。 |
| 动态分区分配 ★★ | 空闲分区表：每个表项中包括 分区编号、分区大小和分区起始地址。<br>空闲分区链：每个结点包括分区大小、 分区起始地址、指向前一个空闲分区结点的指针，以及指向后一个空闲分区结点的指针。<br>算法：（1）首次适应算法 FF（2）循环首次适应算法 NF（3）最佳适应算法 BF（内存利用率高，但易留下难以利用的小空闲区） | |
| 分页存储管理的基本原理 | **页**：将一个进程的逻辑地址空间分成若干个大小相等的片。<br>**页框**：将物理内存空间分成与页大小相同的若干个存储块。进程的最后一页一般装不满一个页框，形成了不可利用的碎片，称为“页内碎片”，是一种**内部碎片**。 | |

| | |
|---|---|
| ★★★ | **页表**：系统为进程建立的数据结构，页表的作用是实现**从页号到页框号的映射**。 |
| 基本分页存储管理方式中的地址结构★★★ | 1、基本分页的逻辑地址结构包含两部分：页号 P 和页内偏移量 W。用 m 位表示逻辑地址，页大小为 $2^n$字节，则用低 n 位表示页内偏移量 W，用高 m-n 位表示页号 P。<br>2、物理地址=页框大小 x 页框号+页内偏移量。<br>若 A 为逻辑地址，L 为页大小，P 为页号，W 为页内偏移量，则有以下计算关系。<br>P = INT（A/L）<br>W= MOD （A/L） |
| 分页地址变换★★ | 基本任务：实现逻辑地址到物理地址的变换。 |
| 引入 TLB 的性能分析★★★ | 在 TLB 中找到某一个页号对应的页表项的百分比称为 TLB 命中率。<br>有 TLB：有效访存时间=一次访问 TLB 的时间加上一次访问内存的时间。<br>没有 TLB：访存时间=一次访问 TLB 的时间加上两次访问内存(一次访问内存页表，一次访问内存读写数据或指令）的时间。 |
| 两级页表★★ | 在二级分页系统中，为了能在地址映射时得到离散存放的页表在物理内存中的地址，需要为页表再建立一个连续存放的外层页表，本书也称之为**页目录表**。页目录表的表项中存放了每一个页表在物理内存中所在的**页框号**。<br>逻辑地址结构：  |
| 基于分页的虚拟存储系统★ | 虚拟存储器：指具有请求调入功能和置换功能，能从逻辑上对内存容量进行扩充的一种存储器系统。 |
| 页表★★ | 是支持请求分页系统最重要的数据结构。<br>页表结构：  |
| 页分配和置换策略★ | 页分配策略：固定分配 / 可变分配<br>页置换策略：局部置换 / 全局置换<br>组合成以下 3 种页分配和置换策略：<br>定分配局部置换<br>可变分配全局置换<br>可变分配局部置换 |
| 页置换算法★★★ | 最佳置换算法和先进先出置换算法（FIFO）：**最佳置换算法**：选择以后永远不会被 访问的页或者在未来最长时间内不再被访问的页作为换出页。（理论）<br>**先进先出页置换算法（FIFO）**:最简单的页置换算法。实现这种算法的一种方式是为每个页记录该页调入内存的时间，当选择换出页时，选择进入内存时间最早的页。<br>**最近最久未使用 LRU 置换算法**：择最近最久未使用的页换出（最近最久未使用的页在最近的将来被访问的可能性也比较小）。（常用） |
| 抖动产生的原因和预防方法★ | **原因**：系统中的进程数量太多，每个进程能分配到的页框太少，以至于进程运行过程中频繁请求调页。<br>预防：采取局部置换策略；在 CPU 调度程序中引人工作集算法；挂起若干进程。 |
| 分段机制的引入★ | 在使用分段存储管理的系统中，程序员使用二维的逻辑地址，一个数用来表示段，另一个数用来表示段内偏移。<br>优点：方便编程、分段共享、分段保护、动态链接，以及存储空间的动态增长。 |
| 分段系统的基本原理★ | 原理：进程的地址空间被划分成若干个段。段的大小不一样，每个段的逻辑地址从 0 开始，采用一段连续的地址空间。<br>逻辑地址结构：二维的，由段号和段内地址组成。 |

## 第五章 文件系统

| 知识点名称 | 知识点内容 |
|---|---|
| 文件★★★ | 文件系统管理是操作系统的重要功能之一，它为用户提供了在计算机系统中对数据信息进行长期、大量存储和访问的功能。 |

| | | |
|---|---|---|
| | 文件系统的用户接口，包括文件的命名、类型、属性和对文件的操作。 | |
| 文件命名 | 多数操作系统都支持文件名用圆点隔开分为两部分 | |
| 文件结构<br>★★ | 无结构字节序列：也称为流式文件，操作系统所见到的就是字节。<br>固定长度记录序列：具有固定长度的记录。<br>树形结构：文件由一棵记录树构成，记录长度不定，在记录的固定位置包含一个关键字域，记录树 按关键字域排序。 | |
| 文件类型<br>★★★ | （1）正规文件，一般分为 ASCII 文件和二进制文件。ASCII 文件由多行正文组成，在某些系统中每行用回车符结束，某些则用换行符结束，而有些系统还同时采用回车符和换行符，各行的长度不必相同。二进制文件：二进制文件具有一定的内部结构，如可执行的.exe 文件。<br>（2）目录文件（3）字符设备文件（4）块设备文件 | |
| 文件存取<br>★★ | 常用的文件存取方式有两种：顺序存取和随机存取。 | |
| 文件属性<br>★★ | 为方便管理，除了文件名和文件数据外，文件系统还会保存其他与文件相关的信息，如文件的创建日期、文件大小和修改时间等，这些附加信息称为文件属性。 | |
| 文件操作<br>★★ | CREATE：创建文件。OPEN：打开文件。CLOSE：关闭文件。READ：从文件中读取数据。WRITE：往文件中写数据。APPEND：该操作是 WRITE 调用的限制形式，它只能在文件末尾添加数据。SEEK：对于随机存取文件，要指定从何处开始取数据。GETATTRffiUTES：获取文件属性。SETATTRIBUTES：修改属性。RENAME：修改已有文件的名件名。 | |
| 目录★★ | 目录：文件系统中实现按名访问文件的重要数据结构。 | |
| 层次目录系统★★ | 目录文件的结构：属性放在目录项中和放在 i 结点中。<br>目录类型：（1）单层目录（2）两级目录（3）树形目录 | |
| 路径名 | 绝对路径名：由从根目录到文件的路径组成。<br>相对路径名：“.”指当前目录，“..”指当前目录的父目录。 | |
| 目录操作★ | CREATE：根据给定的目录文件名，创建目录。DELETE：删除目录。OPENDIR：目录内容可以被读取。CLOSEDIR：关闭目录。READDIR：以标准格式返回打开目录的下一级目录项。RENAME：更换目录名。 | |
| 连续分配<br>★ | 把每个文件作为一连串连续数据块存储在磁盘上。<br>优点：实现简单、读操作性能好。<br>缺点：随着时间的推移，磁盘会变得零碎。 当删除文件时，文件所占的簇被释放，这些空闲的连续簇形成“空洞”。 | |
| 使用磁盘链接表的分配<br>★★ | 优点：可以充分利用每个簇，不会因为磁盘碎片（除了最后一块中的内部碎片）而浪费存储空间，管理也比较简单。<br>缺点：随机存取相当缓慢。 | |
| 使用内存的链接表分配<br>★ | 将文件所在的磁盘的簇号存放在内存的表(文件分配表)中。访问文件时，只需从内存文件分配表中顺着某种链接关系查找簇的簇号。不管文件有多大，在目录项中只需记录文件的第一块数据所在簇的簇号，根据它查找到文件的所有块。<br>缺点：必须把整个表都存放在内存中。不适合大容量的磁盘。<br>MS-DOS 就使用这种方法进行磁盘分配。 | |
| i-结点★ | 为每个文件赋予一个被称为 i 结点数据结构，其中列出了文件属性和文件块的磁盘地址。 | |
| 簇大小 | 一般簇大小是 2 的整数次幂个连续的扇区。 | |
| 记录空闲块 | 空闲簇链接表 | 用一些空闲簇存放空闲簇的簇号。对于 1 KB 大小的簇，可以存放 256 个 32 位 的簇号(有一个存放指向下一个块的指针) |
| | 位图 | 位图方法所需空间少，因为每个簇只用一个二进制位标识，而在链接表方法中，每一个簇号都要用 32 位。 |

## 第六章 I/O 设备管理

| 知识点名称 | 知识点内容 | |
|---|---|---|
| I/O 系统的结构 | 微机 I/O 系统 | 总线型 I/O 系统：CPU 与内存之间可以直接进行信息交换，但是不能与设备直接进行信息交换，必须经过设备控制器。 |
| | 主机 I/O 系统 | 四级结构，包括主机、通道、控制器和设备。一个通道可以控制多个设备控制器，一个设备控制器也可以控制多个设备。 |
| I/O 设备的分类 ★★★ | 按传输速率分类 | 低速设备、中速设备、高速设备。 |
| | 按信息交换的单位分类 | 块设备和字符设备 |
| | 按设备的共享属性分类 | 独占设备、共享设备和虚拟设备 |
| 设备控制器的功能★ | （1）接收和识别命令（2）数据交换（3）设备状态的了解和报告（4）地址识别）（5）数据缓冲（6）差错控制 | |
| I/O 通道★ | 是大型主机系统中专门用于 I/O 的专用计算机。 | |
| DMA 控制方式★★★ | 1、I/O 控制方式有：轮询、中断、DMA。<br>2、DMA 控制器的逻辑组成包括 3 部分：主机与 DMA 的接口、DMA 与设备的接口，以及 I/O 控制逻辑。<br>3、DMA 控制器中 4 类寄存器：命令/状态寄存器 CR、内存地址寄存器 MAR、数据寄存器 DR 和数据计数器 DC。 | |
| 单缓冲★ | 对于面向块的设备：输入数据被传送进入系统缓冲区。<br>对于面向流的 I/O：如键盘、打印机和传感器等。一般用于面向流的设备 | |
| 循环缓冲 | 申请：Getbuf 过程；释放：Releasebuf 过程。 | |
| 设备独立性 ★★ | 含义：为了提高操作系统的可适应性和可扩展性，在现代操作系统中都毫无例外地实现了设备独立性，也称为设备无关性。<br>主要功能：(1)执行所有设备的公有操作(2)向用户层软件提供统一的接口。 | |
| 独占设备的分配程序 ★★ | 如果系统有 I/O 通道分配步骤：（1）分配设备（2）分配控制器（3）分配通道。<br>如果系统，没有 I/O 通道分配步骤：（1）分配设备（2）分配控制器 | |
| I/O 软件原理★★ | 设备管理软件的层次结构：1）用户层软件：进行 I/O 调用，格式化 I/O。2）与设备无关的软件层：命名、保护、阻塞、缓冲、分配。3）设备驱动程序：包括中断处理程序、设备驱动程序。4）中断处理程序（底层）：执行 I/O 操作。 | |
| 设备管理软件的功能★ | （1）实现 I/O 设备的独立性（2）错误处理（3）异步传输（4）缓冲管理（5）设备的分配和释放（6）实现 I/O 控制方式 | |
| 最短寻道时间优先 | 简称：SSTF。其要求访问的磁道与当前磁头所在的磁道距离最近，以使每次的寻道时间最短。 | |
| 扫描算法 | 简称：SCAN。解决“饥饿”现象。<br>不仅考虑到要访问的磁道与当前磁道的距离，更优先考虑磁头当前的移动方向。<br>又称电梯调度算法。 | |
| 循环扫描算法 | 简称：CSCAN<br>规定磁头是单向移动 | |
| NStepSCAN 算法 | 解决“磁臂粘着”现象。<br>将磁盘请求队列分成若干个长度为 N 的子队列，磁盘调度将按 FCFS 算法依次处理这些子队列。每处理一个队列时又是按 SCAN 算法，对一个队列处理完后，再处理其他队列。 | |
| 提高磁盘 I/O 速度的方法 | （1）提前读（2）延迟写（3）优化物理块的分布（4）虚拟盘（5）磁盘高速缓存。 | |