1. &第二章：

操作系统的定义：操作系统是管理和控制计算机硬件与软件资源的计算机程序。

系统调用的定义：提供了操作系统提供的有效服务界面

操作系统的结构：

简单结构：利用最小的空间提供最多的功能，没有很好地区分接口和功能层次

层次结构：构造和调试的简单化，但是效率较差

微内核结构：便于扩充操作系统，但是系统性能低

模块结构：允许内核提供核心服务，也能动态的实现特定的功能。

1. ：

进程的概念：执行中的程序，是资源分配的基本单位

进程的组成：代码段、数据段、堆栈段、程序计数器，还可能包括堆

进程的状态模型

进程控制块PCB所包括的信息：进程状态、程序计数器、CPU寄存器、CPU调度信息、内存管理信息、记账信息、I/O状态信息

进程调度队列：作业队列、就绪队列、设备队列

长程调度（作业调度）：将创建好的进程放入就绪队列中等待

短程调度（CPU调度）：从准备执行的进程中选择进程

中程调度：平衡调度，根据主存状态决定主存中所能容纳的进程数目

上下文切换：将CPU切换到另一个进程需要保存当前进程的状态并恢复另一个进程的状态

消息传递和共享内存的区别：

消息传递比共享内存更易于实现

共享内存比消息传递快

消息传递通常需要系统调用来实现，共享内存仅在建立共享内存区时需要系统调用。

1. ：

线程的概念：是CPU调度的基本单位，由线程ID、程序计数器、寄存器集合和栈组成。

线程的优点：

1. 响应度高
2. 资源共享
3. 经济
4. 多处理器体系结构的利用

进程和线程的区别：

1. 进程有自己的独立地址空间，线程共享进程的地址空间。
2. 进程是资源分配的基本单位，线程是CPU调度的基本单位
3. 进程在执行过程中拥有独立的内存单元，而多个线程共享内存，从而极大地提高了程序的运行效率。
4. 线程的划分尺度小于进程，使得多线程程序的并发性高。

线程模型：

一对一、一对多、多对多、二级模型

线程库：

线程库为程序员提供创建和管理线程的API

POSIX Pthread，Win32，Java

1. ：

CPU调度的概念：即短程调度的概念

为什么需要CPU调度：提高CPU的利用率

分派程序：分配程序是一个模块，用来将CPU的控制交给由短期调度程序选择的进程。

多处理器调度：

处理器亲和性：努力使一个进程在同一个处理器上运行

负载平衡：设法将工作负载平均的分配到SMP系统的所有处理器上。

线程调度：局部调度（调用哪个用户线程占用一个内核线程）和全局调度（调用哪个内核线程占用内核）

内核级线程和用户级线程的区别：

1. 内核级线程是OS内核可感知的，用户级线程是OS内核不可感知的。
2. 在只有用户级线程的系统里，CPU调度还是以进程为基本单位，在有内核级线程的系统里，CPU调度以线程为基本单位。
3. 用户级线程运行在用户态，内核级线程能运行在任何状态
4. 用户级线程的创建、撤销和调度不需要OS支持，内核级需要。
5. 用户级系统调用导致进程中断，内核级系统调用导致线程中断。
6. ：
7. ：

资源的使用模型

死锁的特征：互斥、占有并等待、非抢占、循环等待

处理死锁的三种方法：

死锁预防：确保至少一个必要条件不成立

死锁避免：资源分配图算法（每个资源一个实例）、银行家算法（重点，每个资源多个实例）

死锁恢复：死锁检测（单个实例-等待图，多个实例-类似于银行家算法），死锁恢复（进程终止、资源抢占）

1. ：

内存的结构：很大一组字或字节组成，每个字或字节都有它们自己的地址

地址类型：

逻辑地址：CPU生成的地址

物理地址：内存单元看到的地址

动态加载（一个子程序只有在调用时才被加载）：为了获得更好的内存空间使用率

动态链接：将链接延迟到运行时，常常需要操作系统的帮助

交换的概念

连续内存分配管理：（动态分区只可能导致外部碎片、静态分区可能导致内部碎片）

静态分区：分配为固定大小的分区

动态分区：孔的概念，选择一个空闲孔的方法：首次适应（分配第一个足够大的）、最佳适应（分配最小的足够大的）、最差适应（分配最大的）

碎片的概念：

内部碎片：进程所分配的内存与所要的之差

外部碎片：空闲内存之和可以满足请求但是并不连续，产生外部碎片。

分页技术：物理地址的计算

TLB：有效访问时间的计算

页表的三种结构：

层次页表：向前映射页表

哈希页表（通常用来处理超过32位地址空间）

反向页表（64位的UltraAPARC和PowerPC）

分段技术：用户视角的内存和实际物理内存的分离

分段和分页的区别：

1. 分页是以系统性能为导向的，分段是以服务用户为导向的。
2. 页的大小是固定的，段的大小是不固定的。
3. 页是一维的，段是二维的。
4. ：

虚拟内存的过程：

按需调页的过程，要能够结合内存，外存进行描述每个步骤

计算虚拟内存的性能：按需调页有效访问时间的计算

写时复制的使用环境：允许父进程和子进程开始时共享同一个页面。如果任何一个进程需要对页进行写操作，就创建一个共享页的副本。

页面置换算法

最优置换和LRU置换都没有Belady异常，都属于栈算法。

帧分配：

平均分配、比例分配

全局置换（允许一个进程从所有帧集合中选择一个置换帧，而不管该帧是否已分配给其他进程）和局部置换（每个进程只能从自己的分配帧中进行选择）的概念。

系统颠簸

定义：频繁的页调度行为

特征：一个进程在换页上的时间多于执行时间

原因：空闲物理帧不能满足要求，导致频繁的页调度行为

抖动检测：工作集合模型和页错误频率。

抖动消除：降低系统多道程序的程度

工作集合模型：基于局部性假设。最近x个引用的页合称为工作集合。工作集合是程序局部的近似。防止了颠簸，并尽可能地提高了多道程序的程度，优化了CPU使用率。

通过固定定时中断和引用位，能近似模拟工作集合模型。

页错误频率：为所期望的页错误率设置上限和下限。

工作集合和页错误率：当一个新的局部按需调页时，页错误率进入波峰，一旦新局部的工作集合在内存内，页错误率开始下降。当进程进入一个新的工作集合，页错误率又一次升到波峰，然后再次降到波谷。从一个波峰的开始到下一个波峰的开始，显示了工作集合的迁移。

1. ：

文件的概念：文件是记录在外存的相关信息的具有名称的集合

文件的属性：名称、标识符、类型、位置、大小、保护、时间、日期和用户标识

文件的操作

文件的类型

文件的结构

文件的内部组织

访问方法：

顺序访问：基于文件的磁带模型

直接访问：基于文件的磁盘模型，可立即访问大量信息。

索引访问

目录的操作：搜索文件、创建文件、删除文件、遍历目录、重命名文件、跟踪文件系统

目录结构：

单层目录结构：便于理解和支持

双层目录结构：解决名称冲突问题，但是对用户加以隔离

树状结构目录：最常用的目录结构，禁止共享文件和目录

绝对路径（从根开始）和相对路径（从当前目录开始）

1. ：

文件系统的层次结构：应用程序-逻辑文件系统-文件组织结构-基本文件结构-I/O控制-设备，前三个为用户视图，后三个计算机视图

文件系统实现过程中的内存数据结构：安装表、目录结构缓存、

文件系统实现过程中的外存数据结构：引导控制块、卷控制块、目录结构、FCB（文件控制块）

目录的实现方式：线性列表、哈希列表

磁盘的数据分配方式：

连续分配：每个文件在磁盘上占有一组连续的块，支持顺序访问和直接访问。访问容易，困难的是为新文件找到空间，可能导致外部碎片。

链接分配：支持顺序访问。创建新文件很容易，没有外部碎片，不能有效支持直接访问，同时，指针也需要空间。变种为文件分配表，可支持直接访问。

索引分配：把链接分配的所有指针放到一个索引块中。支持直接访问，而且没有外部碎片问题。

空闲空间管理：

位图（位向量）:掌握计算

链表

成组

计数

虚拟文件系统VFS：

通过定义一个清晰的VFS接口，以将文件系统的通用操作和具体实现分开。提供了在网络上唯一标识一个文件的机制。

1. ：

磁盘的结构

磁盘的访问性能：寻道、旋转和读取

磁盘的调度算法：FCFS、SSTF、SCAN、C-SCAN、C-LOOK

磁盘的管理：初始化、引导、坏块管理

交换空间的组织与管理：为虚拟内存提供最佳吞吐量

交换空间可有两个位置：普通文件系统上和一个独立的磁盘分区

1. ：

I/O硬件的结构：端口、总线、控制器

I/O端口通常有四种寄存器：数据输入/输出寄存器、状态寄存器、控制寄存器

CPU对I/O的编址方式：I/O映射、内存映射

I/O通信方式：轮询、中断、DMA

DMA的优势和缺点：改善系统总体性能，但是也破坏了系统的安全和稳定。

应用程序接口（隐藏设备控制器之间的差异）：

块设备、字符设备、同步异步（阻塞非阻塞）、时钟

内核子系统：调度、缓冲、高速缓存、假脱机、设备预留、错误处理、I/O保护

缓冲：

缓冲区是用来保存两个设备之间或在设备和应用程序之间所传输数据的内存区域。

采用缓冲的三个理由：

处理数据流的生产者和消费者之间的速度差异。

协调传输数据大小不一致的设备

支持应用程序I/O的复制语义（用户内存的缓冲区被分到内核内存中）

假脱机：用来保存设备输出的缓冲区，这些设备不能接收交叉的数据流。

为了防止用户执行非法I/O，定义所有I/O指令为特权指令，I/O调用均为系统调用。