操作系统复习纲（自定义版）

目录

Chapter 1

- DMA 技术（于 13 章详述）
- 双模式操作
  - 分类：用户模式、内核模式
  - 实现：模式位（内核为 0，用户为 1）、通过陷阱与返回利用系统调用实现互换更多参见图 1.8
  - 好处：提供了保护操作系统和用户程序不受错误用户程序影响的手段，实现方法是将指令分级
- 高速缓存
  - 将信息临时地复制到更快的存储系统中
  - 需要时，检查是否存在，并进行相关操作
  - 高速缓存一致性问题
- 分布式系统通用结构
  - 客户机-服务器模式

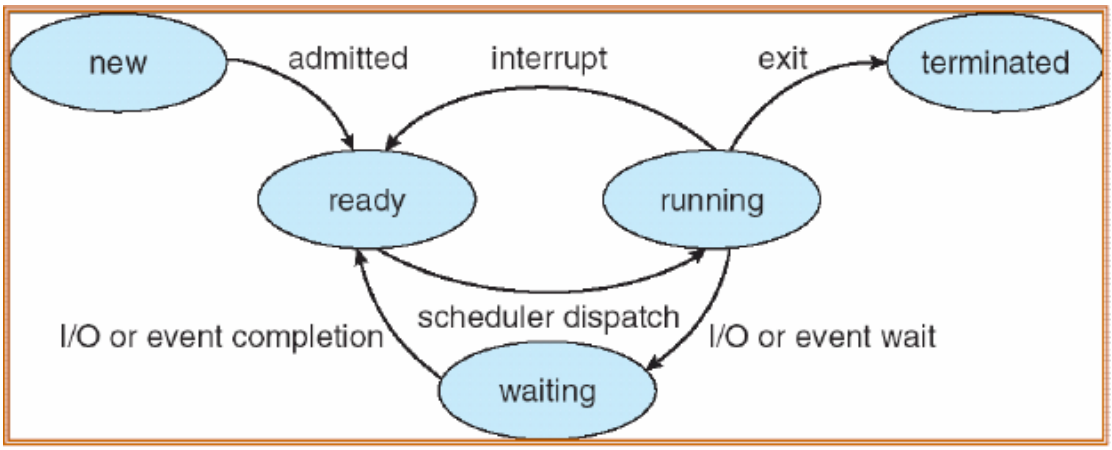
- 对等模式

Chapter 2

- 操作系统的用户界面
  - 命令解释程序 -> 外壳
  - 图形用户界面
- 系统调用
  - 是什么：运行在使用者空间的程序向操作系统内核请求需要更高权限运行的服务。系统调用提供了用户程序与操作之间的接口。大多数交互系统调用提供了用户程序与操作系统之间的接口。开发人员利用相关的应用程序接口来设计程序。
  - 做什么：进程控制、文件管理、设备管理、信息维护和通信
- 通信
  - 消息传递模型（进程名与邮箱）
  - 共享内存模型（读写公共区域）
- 操作系统结构
  - 简单结构：MS-DOS（未很好区分接口与功能层次）与 UNIX（由内核与系统程序两个独立部分组成）
  - 系统模块化：
    - 层次化的处理（分层法） 图 2.12 ->虚拟机
      - 优点：构造与调试的简单化，每层只利用较低层功能与服务，简化系统设计与实现。
    - 模块：例子 Solaris 系统
  - 实例：UNIX 与 Microkernel
- 虚拟机
  - 实现机制：虚拟用户模式与虚拟内核模式、虚拟机监控器
  - 好处：系统资源完全保护、用于研究和开发操作系统的好工具、正常系统操作无须进行中断来开发系统
  - 实例：VMWare 与 JAVA 虚拟机

Chapter 3

- 进程的概念
  - 分为作业与进程两种概念
  - 进程包含**文本段、程序计数器、堆栈段与数据段**，有时还包括堆
  - 进程与程序的区别（被动实体与活动实体）
- 进程状态
  - 分类：**New / Running / Waiting / Ready / Terminated** 即 **新的/运行/等待/就绪/终止** 部分操作系统还有挂起 (suspended)
  - 相互转化操作：**图 3.2**



- 进程控制块
  - 内容：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
  - 其中 CPU 寄存器使得进程间切换（图 3.4）成为可能
  - 类比：线程控制块
- 进程调度
  - 调度队列：作业队列、就绪队列与设备队列
  - 调度程序 图 3.8
    - 长期调度程序：适用于进程装入内存 （I/O-bound process / CPU-bound process）
    - 短期调度程序：适用于进程分配给 CPU
    - 中期调度程序：swapping，临时换出 CPU
  - 多道程序设计的度：内存中的进程数量
  - 上下文切换
    - 利用：进程控制块中的 CPU 寄存器的值、进程状态、内存管理信息等等
    - 操作：状态保存与状态恢复
- 进程创建
  - 利用系统调用 fork()实现整个操作
  - 父进程与子进程的关系
    - 资源分配：可能从操作系统那里直接获得资源，也可能只从其父进程那里获得资源
    - 执行状态：1 可能与父进程并发执行 2 可能让父进程等待，直到某个或全部子进程执行完
    - 地址空间：1 子进程与父进程的复制品 2 子进程装入另一个新程序
- 进程的终止
  - 调用 exit()，返回状态给父进程，系统释放
  - 父进程的终止造成的中止
- 进程间通信
  - 原语：send() 与 receive()
  - 分类：直接通信（面向进程）、间接通信（面向邮箱或端口）
  - 同步：阻塞式与非阻塞式
  - 缓冲：零容量、优先容量、无限容量

Chapter 4

- 线程的概念
  - CPU 使用的基本单元，由线程 ID、程序计数器、寄存器集合、栈组成
  - 四大优点：响应度高、资源共享、经济、多处理器体系结构的利用
- 多线程模型：多对一模型、一对一模型、多对多模型、二级模型
- 系统调用 fork()与 exec()
  - fork()：线程调用时，有两种调用模式，1 新进程会复制所有进程 2 只复制调用的线程
  - exec()：指定程序会替换整个进程，包括所有线程
- 线程池
  - 定义：创建一定数量的线程，放入池中等待工作，收到请求，唤醒线程，完成服务，返回池中等待
  - 优点：1 响应快 2 限制可用线程数量，对于不能支持大量并发线程的系统非常重要

Chapter 5

- CPU 调度
  - 作用：通过在进程之间切换 CPU，可以提高计算机的吞吐率，使 CPU 的使用率最大化。
  - 分类：
    - 非抢占式：仅 1）运行切换到等待、4）进程终止
    - 抢占式：除以上还包括 2）运行切换到就绪、3）等待切换到就绪
- 调度算法（思想与实现见课本——5.3）
  - 类型：先到先服务(FCFS)、最短作业优先(SJF)、优先级调度、轮转法调度(RR)、多级队列、多级反馈队列
  - 问题：是否抢占、饥饿问题、老化处理
- 多处理器调度：非对称多处理与对称多处理
- 算法评估：分析评估法、确定模型法、排队模型、模拟法

Chapter 6

- 进程同步：用于确保共享同一逻辑地址空间的协作进程可有序地执行，维护数据一致性。
- 临界区问题
  - 解释：设计一个以便进程协作的协议，每个进程必须请求允许进入临界区，特征是当一个进程进入临界区的时候，没有其他进程可以被允许在临界区内执行。
  - 三项要求：互斥、前进、有限等待
  - 处理方式：抢占内核（v）、非抢占内核
  - 解决方式：Peterson 算法

- 同步原子性
  - 特点：使用特殊硬件指令保证原子地执行
  - 分类：TestAndSet()（lock 初始 false） 与 Swap()（全局 lock 初始 false，另一局部 key 初始 true）
- 信号量【**】
  - 原子操作：P()或 wait() 与 V()或 signal() 【**伪代码表示】
  - 用法：mutex（互斥锁，初始化为 1）【**初始化的注意事项】
  - 实现：忙等待(busy waiting)、自旋锁 -> 改变：自我阻塞
  - 死锁与饥饿问题
- 管程
  - 同步机制：条件(condition)机制，wait 与 signal
  - 处理机制：唤醒并等待、唤醒并继续
- 原子事务
  - 事务的提交与撤销
  - 基于日志的恢复：undo（恢复）、redo（设置新值）
  - 检查点执行与维护

Chapter 7

- 死锁问题
  - 定义
  - 四个必要条件：1）互斥 2）占有并等待 3）非抢占 4）循环等待
  - 描述：资源分配图
  - 破坏：将必要条件之一变为不满足
- 死锁的预防
  - 互斥：无法实现
  - 占有并等待：资源利用率低、发生饥饿
  - 非抢占：变为部分抢占，用于状态可以保存或恢复的资源
  - 循环等待：排序并按序申请资源
- 死锁的避免
  - 状态：安全状态、不安全状态（不是死锁状态） –> 判定标准：是否为安全序列
  - 算法：银行家算法
    - 数据结构：**Available、Max、Allocation、Need**
    - 内容：安全性算法、资源请求算法
- 死锁的检测
  - 资源单个实例：等待图（图 7.8）
  - 资源多个实例：类银行家算法 –> 使用 **Available、Allocation、Request**

Chapter 8

- 主存基础知识

- 空间保护：**基地址寄存器、界限地址寄存器** （图 8.1 8.2）
- 地址绑定：
  - 编译时绑定 –> 绝对代码
  - 加载时绑定 –> 可重定位代码
  - 执行时绑定 （需要硬件支持）

- 物理地址空间与逻辑地址空间
  - 区别：逻辑地址空间（CPU 生成的地址）、物理地址（内存单元加载到内存地址寄存器的地址）
  - 映射实现：内存管理单元（MMU） 重定位寄存器
  - 意义：动态加载（调用时加载），摆脱进程的大小受物理内存大小的限制，获得更好的内存空间使用率
- 动态库与交换
  - 动态库：链接延迟到运行时
  - 交换：滚入滚出，并通过就绪队列管理
- 内存分配方式
  - 连续内存分配
    - 实现分类：首次适应、最佳适应、最差适应
    - 问题：内部碎片、外部碎片存在
  - 分页（更多见后文）
    - 问题：内部碎片
  - 分段
    - 问题：外部碎片
    - 实现：分段表 【逻辑分段】
  - 内部碎片与外部碎片
    - 内部碎片：进程分配内存比所需要的大，两者数值之差即为内部碎片。
    - 外部碎片：随着进程装入和移出内存，空闲内存空间被分为小片段。当所有总的可用内存之和可以满足请求，但并不连续时，即有外部碎片。
    - 内部碎片的 50%原则
- 分页技术
  - 机制：将物理内存中的帧对应到逻辑内存中的页
  - 实现：页表构建，CPU 地址（页号与页偏移）（图 8.8）
  - 映射问题：m 位 其中 m-n 位页号 n 位页偏移 这是一种动态重定位
  - TLB 的构建与映射
    - 寄存器：页表基寄存器
    - 缓冲区：转换表缓冲区（TLB）
    - TLB 命中与失效：直接寻址与返回页表
    - 硬件支持：并行快速查找
  - 分页机制保护与共享
    - 保护位：有效无效位
    - 共享处理：页表前置共享代码段，即可重入代码，其执行期间不会改变
  - 层次页表
    - 二级分页（向前映射页表），即多级映射
    - 哈希页表（链式哈希）
    - 反向页表（通过 pid 唯一对应）

- 分段技术与其区别
  - 分段技术：用户视角的内存管理
  - 与分页的区别与联系
    - 相同：均可在物理内存基础上进行地址空间扩展
    - 差异
      - 程序员仅能意识到分段
      - 过程与数据仅在分段中得以保护
      - 分段允许表扩张
      - 分段可促进进程间的过程分享
      - 纯分段有外部碎片的困扰

Chapter 9

- 虚拟内存的背景与实现
  - 背景：其允许执行进程不必完全在内存中，使程序可以比物理内存大，并可以很容易地共享文件与地址空间。
  - 好处：不受现有内存限制，获得巨大的虚拟地址空间；使用更少物理内存，提高 CPU 使用率；减少内存 I/O 交互，用户程序运行更快。
  - 实现：内存管理单元的逻辑页映射到内存的物理页帧。
- 按需分页与页错误
  - 基本概念：懒惰交换、页错误陷阱、页错误率
  - 实现：共有 6 步（图 9.6）
  - 原理：程序局部引用
- 写时复制
  - 按需清零（具体方式见图 9.7 9.8）
- 页面替换
  - 定义：如果没有空闲帧，那么就查找当前没有使用的帧，并将其释放。可采用这样的方式释放一个帧：将其内容写到交换空间，并改变页表，以表示该页不在内存中。
  - 相关定义：牺牲帧、修改位
  - 算法（工作方式具体参阅课本）
    - FIFO 置换：性能（图 9.13）、问题（Belady 异常）
    - 最优置换：性能最优、无异常、置换最长使用页、问题：无法实现
    - LRU 置换：计数器实现与栈实现、问题：需要硬件支持
    - 近似 LRU 置换：附加引用位算法、（增强型）二次机会算法
    - 基于计数的页置换：LFU 与 MFU
- 帧分配
  - 平均分配算法
  - 全局分配与局部分配
- 系统颠簸
  - 原因：页错误过多，换页时间多于执行时间（程序的度与 CPU 利用率之间的恶性循环）
  - 解决方案（利用局部模型）
    - 工作集合模型：工作集合Δ 的取值

        - ◆ 页错误频率：上下限的设置
- 内核内存分配
    - 伙伴系统：二分法
- 其他因素与影响
    - 预调页：降低页错误
    - 页大小：考虑碎片、页表大小、I/O 开销、程序局部性
    - TLB 范围：考虑成本

Chapter 10

- 访问方法
    - 顺序访问：磁带模型
    - 直接访问：磁盘模型、相对块号
    - 其他访问：索引访问
- 目录结构
    - 单层结构：便于理解支持，但是文件名称唯一
    - 双层结构：出现用户文件目录与主文件目录，解决名称冲突，但是无分组能力
    - 树状结构：可完成分组、搜索更快，用户间互相访问实现，但是不能共享与误删除操作
    - 无环图结构：实现树状的共享目录，但是没解决共享文件与复用与删除问题
    - 通用图结构：垃圾收集
    - 【**】悬空指针：无环图结构的共享文件删除时，可能留下悬空指针指向不存在的文件，所以采用链接实现共享系统

Chapter 11

- 设备驱动程序的概念：实现内存和磁盘之间的信息传输，作为翻译器存在。
- 打开文件表：FCB（图 11.2 11.3）
- 文件系统分配
    - 连续分配：连续块访问，但是外部碎片问题
    - 链接分配：采用链表访问，但是增加寻道时间
    - 索引分配：通过索引块解决问题可采用链接方案、多层索引与组合方案
- 空闲空间管理
    - 位向量：需要数组向量
    - 链表：将空块使用链表相连

Chapter 12

- 寻道时间：移动磁臂到所要的柱面所需时间

- 旋转延时：等待所要的扇区旋转到磁臂下所需时间
- 磁盘调度方法
  - 分类：FCFS、SSTF、SCAN、C-SCAN、LOOK
  - 策略：最近或最方便，是否从头开始等
- RAID
  - 作用：通过多块磁盘，提高性能与可靠性
  - 分类：RAID 0-6 （图 12.11）
  - 目标：通过冗余实现可靠性，通过分散实现性能提升，加快吞吐量

Chapter 13

- I/O 硬件概念：总线、端口、控制器、驱动
- 控制方式：
  - 轮询：快速及时处理，但是出现忙等
  - 中断：使用中断处理程序通过中断解决忙等
  - 直接内存访问：使内存与 I/O 设备直接访问，而不需要一直经过 CPU
- I/O 的阻塞与非阻塞
  - 系统调用使用阻塞式
  - 一般的输入输出为非阻塞式
- I/O 内核子系统
  - 缓冲：协调数据传输不一致
  - 高速缓存：高效访问数据
  - 假脱机与设备预留：设备保存设备输出的缓冲区

Chapter 14

- 保护
  - 定义指一种控制程序、进程或用户对计算机系统资源进行访问的机制。
  - 与安全的区别：保护是对内部操作的访问资源的保护；而安全是防止外部的非法非授权操作等。
- 保护域：进程、对象、权限的三元组（图 14.1）
- 访问矩阵
  - 结构：行代表域，列代表对象，中间定义权限（图 14.4.-14.7）
  - 实现与优缺点
    - 全局表：实现简单，但是表很大，不能充分分组，查找费时
    - 对象访问列表：访问对象域方便，访问域集合困难
    - 域权限列表:：访问进程局部信息有用，但是撤回权限操作效率低

Chapter 15

- 安全的定义（参见名词解释）
- 木马、后门、逻辑炸弹定义（参见名词解释）
- 病毒蠕虫的定义区别（参见名词解释）

卷面分配

- **名词解释 15’（5x3’）**
- **问答题 40’（8x5‘）**
- **综合分析计算设计题 45’（x4）**