第 1 章 操作系统概论

1. 早期操作系统设计的主要目标是什么?

   简化硬件操作：为用户和程序员屏蔽复杂的硬件操作。

   提高资源利用率：通过批处理和调度技术，提高计算机硬件的使用效率。

   提供基本服务：为程序执行提供必要的输入/输出、存储管理等服务。

2. 操作系统是资源管理程序，它管理系统中的什么资源?

   处理器资源：CPU 调度与多任务管理。

   内存资源：内存分配与回收。

   存储资源：磁盘、文件系统管理。

   输入/输出设备：管理外围设备的访问。

   网络资源：网络连接和通信。

3. 为什么要引入多道程序系统?它有什么特点?

   引入原因:

   - 提高 CPU 利用率，避免 CPU 空闲时间过多。
   - 加速任务处理，提高系统吞吐量。

   特点:

   - **并发性**：多个程序同时驻留内存，交替执行。
   - **资源共享**：多个程序共享 CPU、内存和 I/O 设备等资源。
   - **独立性**：各程序之间互不干扰

4. 叙述操作系统的基本功能。

   进程管理：创建、撤销、调度和同步进程。

   内存管理：分配、回收和保护内存。

   文件管理：提供文件存储、检索和保护功能。

   设备管理：管理外围设备的访问和操作。

   用户接口：提供命令行界面和图形用户界面。

   安全与保护：保护系统资源和数据的完整性。

5. 批处理系统、分时系统和实时系统各有什么特点？各适合应用于哪些方面?

类型	**特点**	**适用场景**
批处理系统	任务自动批量处理，无需人工干预，强调吞吐量。	数据处理、大型计算任务
分时系统	支持多个用户同时交互，响应时间短，强调交互性。	多用户系统、桌面操作系统
实时系统	实时响应外部事件，保证时间约束，强调及时性和可靠性。	工业控制、航空航天系统

6. 操作系统的主要特性有哪些?

   并发性：多个任务同时执行。

   共享性：多个任务共享系统资源。

虚拟性：通过抽象技术呈现虚拟资源。
异步性：任务执行时间不确定。

7. 衡量 OS 的性能指标有哪些？什么是吞吐量、响应时间和周转时间？

性能指标：吞吐量、响应时间、周转时间、CPU 利用率、系统可靠性。

- **吞吐量**：单位时间内完成的任务数量。
- **响应时间**：从提交任务到开始响应的时间。
- **周转时间**：从提交任务到完成任务所需的总时间。

8. 什么是嵌入式系统?

嵌入式系统是一种专用计算机系统，嵌入到设备中完成特定任务，通常具有低功耗、小体积和高可靠性的特点，例如家电、汽车控制系统。

9. 什么是对称多处理？它有什么好处？

对称多处理（SMP）：多个 CPU 共享内存和外部设备，平等地处理任务。
好处：

- 提高系统性能和并行处理能力。
- 负载均衡，避免资源浪费。

10. 为了实现系统保护，CPU 通常有哪两种工作状态？各种状态下分别执行什么程 序？什么时候发生状态转换？状态转换由谁实现？

两种状态：

- **用户态**：执行普通用户程序，访问受限。
- **内核态**：执行操作系统程序，权限高。

状态转换：

- **用户态 → 内核态**：系统调用、硬件中断。
- **内核态 → 用户态**：完成系统服务后返回用户程序。

实现者：通过硬件（如中断机制）和操作系统调度实现。

11. 什么是系统调用？什么是特权指令？特权指令执行时，CPU 处于哪种工作状态？

系统调用：用户程序向操作系统请求服务的接口。
特权指令：仅允许在内核态执行的指令（如 I/O 操作、内存管理）。
执行状态：特权指令只能在内核态执行

12. 操作系统通常向用户提供哪几种类型的接口？其主要作用是什么？

接口类型：

1. **命令行接口（CLI）**：提供文本命令输入，适合高级用户。
2. **图形用户界面（GUI）**：通过图形化方式操作，用户体验友好。
3. **系统调用接口**：为程序员提供访问 OS 功能的编程接口。

作用：简化用户与系统的交互，为程序开发提供抽象的硬件访问。

第 2-3 章 进程管理

1. 程序顺序执行的特点是什么？

顺序性：程序按编写的顺序一条一条指令执行。

封闭性：程序的执行与外部无关，不受其他程序影响。

确定性：在相同输入条件下，程序执行结果是确定的。

2. 何谓进程？进程由哪些部分组成？试述进程的四大特性（动态性、独立性、并 发性、结构性）及进程和程序的区别。

进程定义：进程是一个正在运行的程序，是系统资源分配和调度的基本单位。

组成部分：

- **程序代码**：描述进程要执行的操作。
- **数据**：程序执行中需要的数据。
- **进程控制块（PCB）**：存储进程状态和管理信息。

四大特性：

- **动态性**：进程是程序的动态执行过程。
- **独立性**：进程有独立的地址空间和资源。
- **并发性**：多个进程可以在时间上交替执行。
- **结构性**：进程包含代码、数据、PCB 三部分。

区别：

- **程序**是静态的指令集合，而**进程**是动态的执行实体。
- 一个程序可以对应多个进程实例。

3. 进程控制块的作用是什么？它主要包括哪几部分内容？

作用：PCB 是操作系统管理和调度进程的核心数据结构，记录进程的所有信息。

主要内容：

1. **标识信息**：进程 ID、父进程 ID。
2. **状态信息**：当前状态、优先级、寄存器值。
3. **资源信息**：打开的文件、设备分配。
4. **控制信息**：进程队列指针、调度信息。

4. 进程的基本状态有哪些？试举出使进程状态发生变化的事件并描绘它的状态转换图。

基本状态：

1. **就绪**：等待 CPU 分配。
2. **运行**：正在占用 CPU 执行。
3. **阻塞**：等待某事件完成（如 I/O）。

状态转换事件：

- **就绪 → 运行**：被 CPU 调度。
- **运行 → 阻塞**：等待 I/O 完成或资源不可用。
- **运行 → 就绪**：时间片用完，被抢占。
- **阻塞 → 就绪**：等待事件完成。

状态转换图：

就绪 → 运行 → 阻塞 → 就绪

运行 ← 就绪

5. 什么是原语？什么是进程控制？

原语：由操作系统提供的用于实现进程控制的基本操作，它不可分割且不被中断（例如创建进程、终止进程）。

进程控制：对进程的创建、撤销、挂起、唤醒和状态切换等操作。

6. 试述进程调度的功能、方式、时机、算法；作业调度，交换调度；作业的周转 时间和作业的带权周转时间。

进程调度：

- **功能**：分配 CPU 资源，提高系统效率。
- **方式**：抢占式、非抢占式。
- **时机**：进程创建、结束、阻塞、时间片到期时。
- **算法**：先来先服务（FCFS）、短作业优先（SJF）、优先级调度、时间片轮转（RR）。

作业调度：决定作业何时进入内存。

交换调度：将不活跃的进程换出到外存。

周转时间：作业完成时间 - 提交时间。

7. 试述线程的定义，线程与进程的比较，系统对线程的支持（用户级线程、核心 级线程、两级组合）。

定义：线程是进程中的一个执行流，是 CPU 调度的基本单位。

比较：

方面	线程	进程
调度单位	线程	进程
资源分配	共享父进程资源	独立分配资源
开销	低（线程间切换）	高（进程间切换）

支持方式：

- **用户级线程**：线程管理在用户空间完成。
- **核心级线程**：线程管理由操作系统内核完成。
- **两级组合**：用户级和核心级线程结合。

8. 并发执行的进程在系统中通常表现为几种关系？各是在什么情况下发生的？

竞争关系：共享临界资源时（如文件）。

协作关系：进程间需要通信或同步（如管道通信）。

9. 什么叫临界资源？什么叫临界区？对临界区的使用应符合的四个准则（互斥使 用、让权等待、有空让进、有限等待）。

临界资源：在同一时间只允许一个进程访问的资源。

临界区：访问临界资源的代码片段。

四个准则：

1. **互斥使用**：一个时刻只有一个进程进入。
2. **让权等待**：无资源时主动放弃 CPU。
3. **有空让进**：空闲时允许进入。
4. **有限等待**：进入等待的进程不会无限期等待。

10. 试述解决进程之间互斥的办法（开、关中断，加锁、开锁（又叫测试与设置， 通常由一条机器指令完成），软件方法，信号量与 P、V 操作）。

- **硬件方法**：开/关中断，测试与设置指令（TSL）。
- **软件方法**：Dekker 算法、Peterson 算法。
- **信号量方法**：通过 P（wait）、V（signal）操作管理资源访问。

11. 若信号量 S 表示某一类资源，则对 S 执行 P、V 操作的直观含意是什么？当进 程对信号量 S 执行 P、V 操作时，S 的值发生变化，当 S>0、S=0、和 S<0 时， 其物理意义是什么?

- **P 操作**：请求资源（S--）。
- **V 操作**：释放资源（S++）。

物理意义：

- **S > 0**：资源可用数为 S。
- **S = 0**：无可用资源。
- **S < 0**：等待队列长度为|S|。

12. 在用 P/V 操作实现进程通信时，应根据什么原则对信号量赋初值？

资源数：信号量表示资源数量时，初值为资源个数。

同步需求：信号量表示同步需求时，初值根据通信顺序设置。

13. 掌握经典的 IPC 问题。

生产者-消费者问题。

读者-写者问题。

哲学家进餐问题。

14. 进程高级通信有哪些实现机制？

共享内存。

消息队列。

管道通信。

套接字通信。

15. 死锁产生的必要条件及解决死锁的方法。

必要条件：

1. **互斥**：资源不可共享。
2. **占有并等待**：持有资源时申请新资源。
3. **不可剥夺**：资源不可强制剥夺。
4. **循环等待**：形成资源依赖环。

解决方法：

- **预防**：破坏死锁条件。
- **避免**：银行家算法。
- **检测与恢复**：周期性检测，释放资源恢复系统。

16. 理解银行家算法的实质。能够利用银行家算法避免死锁。
实质：通过模拟资源分配，判断是否会进入不安全状态，避免死锁。
步骤：

1. 判断请求是否满足安全条件。
2. 分配资源并更新状态。
3. 若资源不足，则等待。

第 4 章 存储器管理

1. 存储器管理的功能与基本概念

功能：

1. **内存分配与回收**：为用户进程分配主存，释放已完成进程的内存空间。
2. **地址映射**：将逻辑地址转换为物理地址。
3. **存储保护**：防止进程非法访问内存。
4. **虚拟存储管理**：实现主存和辅存的联合使用。

基本概念：

- **名字空间**：系统内标识符的集合，用于描述程序中的变量和数据。
- **地址空间**：程序可用的逻辑地址范围。
- **存储空间**：物理内存的实际存储容量。
- **逻辑地址**：程序产生的地址，独立于物理内存。
- **物理地址**：内存单元的实际地址。

---

2. 地址重定位及其分类

定义：地址重定位是将逻辑地址转换为物理地址的过程。
分类：

1. **静态重定位**：在程序加载时完成，根据装载地址调整逻辑地址。
    - **优点**：无需运行时支持，开销低。
    - **缺点**：内存利用率低，不能适应动态需求。
2. **动态重定位**：在程序运行时完成，通过硬件地址转换实现。
    - **优点**：灵活，可支持动态内存分配。
    - **缺点**：需硬件支持，转换开销较高。

---

3. 内存划分与管理

- **用户空间**：为用户程序提供运行环境。
- **操作系统空间**：存储操作系统核心和关键数据。
- **管理目标**：提高用户空间的利用率。

---

4. 存储保护的目的及实现

目的：防止非法访问，保证系统和用户进程的稳定运行。
实现方式：

- **硬件支持**：基址寄存器和界限寄存器、分页和分段保护。
- **软件支持**：操作系统提供进程访问权限管理。

---

5. 可变式分区管理与存储保护

空闲区管理方法：

1. **首次适配**：从头扫描，找到第一个足够大的空闲区。
2. **最佳适配**：找到最接近需求大小的空闲区。
3. **最坏适配**：选择最大的空闲区。

保护方式：使用基址和界限寄存器。

---

6. 页式存储器的内零头与分区管理的碎片

- **内零头**：分配的内存单元未被完全利用的部分，与页大小成正比。
- **分区碎片**：可变分区管理中的内存碎片包括**内部碎片**和**外部碎片**。

---

7. 覆盖与交换的特点

- **覆盖**：将不常用的程序部分移出内存，根据需求加载。
- **交换**：将整个进程移入或移出内存，用于释放主存资源。

---

8. 页表的作用及地址转换过程

作用：记录每一页的物理地址映射关系。

地址转换过程：

1. 逻辑地址分为页号和页内偏移。
2. 查页表获取物理页号。
3. 结合页内偏移形成物理地址。

常用数据结构：页表、帧表、自由帧链表。

---

9. 段式与页式存储器管理的主要区别

方面	段式管理	页式管理
单位	段	页
大小	不固定	固定大小
目的	符合逻辑需求	提高内存利用率
碎片类型	外部碎片	内部碎片

---

10. 虚拟存储器及其容量问题

定义：虚拟存储器通过辅存与主存联合提供大于主存的存储空间。

容量：不能大于主存容量加辅存容量之和，但可通过页面置换提高效率。

---

11. 请求页式管理中的状态位、修改位、访问位

- **状态位**：表示页面是否在内存中。
- **修改位**：页面是否被修改过，决定是否需要回写到辅存。
- **访问位**：页面是否被访问，用于页面置换算法。

---

12. 缺页中断的处理流程

1. 判断地址是否非法。
2. 分配空闲帧或置换页面。
3. 将所需页面调入内存。

4. 更新页表和硬件状态。
5. 重新执行中断指令。

---

13. 页面置换算法及 Belady 异常
常见算法：
1. **FIFO**：先进先出，易产生 Belady 异常。
2. **LRU**：最近最少使用，性能较优。
3. **OPT**：最优置换，不现实但为理论参考。
4. **Clock**：时钟算法，利用访问位优化。

Belady 异常：页框数增加导致缺页率反而上升的现象。

---

14. 程序局部性原理与系统抖动
局部性原理：程序访问局部区域内的内存。
- **时间局部性**：重复访问最近使用的数据。
- **空间局部性**：访问邻近的内存地址。

系统抖动：频繁缺页导致 CPU 等待，性能下降。
工作集模型：通过动态调整工作集大小避免抖动。

---

15. 多级页表与写时复制技术
- **多级页表**：分层存储页表，减少内存开销，页表在需要时动态创建。
- **写时复制**：进程间共享页面，只有在写操作时才创建副本。

---

16. 页的共享与管理
共享机制：
- 共享代码或数据页。
- 专门数据结构记录共享关系（如页表和引用计数）。

第 5 章 文件系统

1. 文件与文件系统
文件：文件是操作系统用于存储数据的逻辑单位，具有命名、存储和访问功能。
文件系统：文件系统是操作系统中用于管理和存储文件的数据结构和相关功能模块。
主要功能：
1. 提供文件的**存储**、**检索**和**管理**功能。
2. 提供文件的**命名机制**和**目录结构**。
3. 实现文件的**存取控制**和**共享管理**。

UNIX 文件分类：
1. 普通文件
2. 目录文件
3. 设备文件（字符设备和块设备）
   好处：分类清晰，便于管理和访问，支持统一的文件接口。

---

2. 文件目录与文件控制块（FCB）
文件目录的作用：
1. 提供文件的**命名与索引**功能。

2. 支持快速定位文件的存储位置。

目录项内容:

1. 文件名
2. 文件属性（如类型、大小、创建时间等）
3. 文件的存储位置指针

文件控制块（FCB）:

文件系统中保存文件信息的数据结构，通常包括:

- 文件标识符
- 文件属性
- 存储位置指针
- 文件状态信息

---

3. 文件的逻辑结构与存取方法

逻辑结构:

1. **顺序文件**：数据按顺序组织，适合顺序访问。
2. **索引文件**：提供索引，支持快速查找。
3. **链接文件**：使用指针串联，节省存储空间。

存取方法:

1. **顺序存取**：按顺序访问文件内容。
2. **直接存取**：通过逻辑地址直接访问。
3. **索引存取**：通过索引表定位文件数据。

---

4. 文件的物理结构与管理

物理结构:

1. **连续结构**：文件存储在连续的物理块中，易管理但会产生外部碎片。
2. **链接结构**：通过指针连接文件块，节省空间但效率低。
3. **索引结构**：使用索引表记录块号，适合大文件存储。

管理方法:

- **连续结构**：需要分区或空闲表管理。
- **链接结构**：通过链表维护文件块。
- **索引结构**：维护索引表并进行快速查找。

---

5. DOS 文件卷与物理结构

文件卷结构:

- 引导扇区
- 文件分配表（FAT）
- 根目录区
- 文件数据区

文件物理结构:

- 使用 FAT 链表记录文件块的分配情况。

---

6. 记录的组块与分解

组块：将记录分组存储以提高磁盘访问效率。

分解：将大的存储块分解为记录，便于数据读取和处理。

---

7. 文件存储空间的管理方法

1. **连续分配**：文件占用连续的磁盘块。
    - **优点**：高访问效率。
    - **缺点**：容易产生碎片。
2. **链接分配**：通过指针连接文件块。
    - **优点**：节约空间。
    - **缺点**：访问速度较慢。
3. **索引分配**：使用索引表存储块号。
    - **优点**：适合随机存取。
    - **缺点**：需要额外存储索引表。

---

8. 多级目录与文件共享

多级目录的好处：

1. 提高文件组织性和查找效率。
2. 支持文件分层管理，减少冲突。

解决重名和共享：

1. 使用目录路径区分文件名。
2. 共享文件使用硬链接或符号链接。

---

9. 文件操作命令

常见命令及功能：

- **创建文件**：建立文件。
- **删除文件**：释放文件空间。
- **读写文件**：操作文件内容。
- **重命名文件**：更改文件名称。
- **打开/关闭文件**：打开文件加载 FCB，关闭释放资源。

---

10. 存取控制表（ACL）

概念：

ACL 是一种存取控制机制，为每个文件定义哪些用户或进程有特定权限（如读、写、执行）。

---

11. 内存映射文件（Memory Mapped File）

过程：

1. 将文件内容映射到内存地址空间。
2. 应用程序通过内存访问操作文件内容，无需 I/O 系统调用。
3. 优点：访问速度快，便于文件共享。

第 6 章 设备管理

1. I/O 设备分类

两大类:

1. **字符设备**：按字节或字符进行数据传输，如键盘、鼠标、打印机。
    - 特点：传输单位为字符，速度较慢，通常需要即时响应。
2. **块设备**：以数据块为单位传输，如磁盘、光盘。
    - 特点：传输单位为块，支持随机访问，速度较快。

---

2. 数据传输方式

四种常用传输方式:

1. **程序直接控制方式**：由 CPU 直接控制数据传输，适用于简单设备。
    - 优点：实现简单。
    - 缺点：占用 CPU 时间，效率低。
2. **中断驱动方式**：设备完成操作后发出中断信号，通知 CPU 处理。
    - 优点：减少 CPU 等待时间。
    - 缺点：适合较低速设备，复杂性增加。
3. **DMA 方式（直接存储器访问）**：设备直接与内存传输数据，无需 CPU 干预。
    - 优点：效率高，适合高速设备。
    - 缺点：需要额外的 DMA 控制器支持。
4. **通道控制方式**：利用专用 I/O 通道处理设备与内存的数据传输。
    - 优点：并行处理能力强，适用于高性能系统。
    - 缺点：硬件成本较高。

---

3. 设备的使用方式与虚拟设备

设备类型:

1. **独占设备**：一次只能供一个用户使用，如打印机。
2. **共享设备**：多个用户可同时使用，如磁盘。
3. **虚拟设备**：通过软件技术将独占设备转换为可供多个用户同时使用的设备。

虚拟设备的实现:

通过 **SPOOLING 技术**（同时外围设备操作与联机操作）。

---

4. I/O 软件的层次结构

I/O 软件分层:

1. **设备驱动程序**：直接与硬件交互，完成具体操作。
2. **设备独立性软件**：提供设备无关的接口和功能。
3. **用户级 I/O 库**：为用户程序提供系统调用接口，简化 I/O 操作。
4. **缓冲管理模块**：对数据进行缓冲，提高传输效率。

---

5. 设备独立性

定义:

设备独立性指用户程序不需要关心具体设备的类型或编号，只需操作逻辑设备，具体映射由操作系统完成。

实现:

通过设备驱动和设备独立性软件隔离设备的物理特性与用户逻辑操作。

---

6. SPOOLING 技术

概念:

SPOOLING（Simultaneous Peripheral Operation On-Line）是一种以空间换时间的技术，用于解决独占设备共享的问题。

实现原理（以输出为例）:

1. 将用户输出请求存入磁盘的**缓冲区**中。
2. 一个专用输出进程负责将缓冲区数据逐个输出到打印机。
3. 用户请求与设备操作分离，实现虚拟设备。

---

7. 磁盘信息编址方式

磁盘上的信息通过**盘面号**、**磁道号**、**扇区号**进行唯一标识:

- **盘面号**：指定磁盘的哪一面（双面磁盘）。
- **磁道号**：指定盘面上的哪一圈磁道。
- **扇区号**：指定磁道上的具体扇区。
  （等价于柱面号、磁头号和扇区号）

---

8. 磁盘数据传输时间

将磁盘上的一个数据块传输到主存涉及以下时间:

1. **寻道时间（Seek Time）**：磁头移动到目标磁道所需时间。
2. **旋转延迟时间（Rotational Latency）**：目标扇区旋转到磁头下所需时间。
3. **传输时间（Transfer Time）**：数据从磁盘传输到主存的时间。

---

9. 常用磁盘调度算法

1. **先来先服务（FCFS）**：按照请求到达的顺序服务。
   - 优点：公平，简单。
   - 缺点：可能引起长时间寻道。
2. **最短寻道时间优先（SSTF）**：优先处理与当前磁头位置最近的请求。
   - 优点：减少平均寻道时间。
   - 缺点：可能导致饥饿现象。
3. **扫描法（SCAN）**：磁头在磁盘上来回移动，按方向处理请求。
   - **C_SCAN**：只在一个方向处理请求，到边界后直接返回起点。
   - 优点：减少饥饿现象，提高响应一致性。
4. **LOOK 法**：只扫描有请求的磁道，不到达磁盘边界。
   - **C_LOOK**：只在一个方向服务请求后返回起点。

Belady 异常：在某些情况下增加磁盘块反而导致更多缺页。