操作系统填空题 20221982 软工2班 潘铂曦

操作系统概念：

1.控制程序

2.资源管理器

3.拓展机器

操作系统中的资源包括**内存、处理器、I/O 设备**等

操作系统**内核**的组成包括：

1.CPU 管理器

2.内存管理器

3.文件系统

4.设备管理器

操作系统**类型**：

1.批处理系统

2.多道~~批处理~~程序系统

3.分时系统

4.桌面式操作系统

5.嵌入式操作系统

现代计算机和操作系统是由**中断**驱动的。

外围设备使用**中断**来向 CPU 发出发生了什么事情的信号。

异常也称为**软件生成的中断或同步中断（SGI）**。

CPU 通过**设备控制器**访问外围设备。

设备控制器包括**命令寄存器和数据寄存器**。

CPU 如何访问这些寄存器？

- **I/O 端口**
- **内存映射 I/O**（缺点是缓存）

硬件保护的方式：

1.二态模式

2.特权指令

3.CPU 保护

3.存储器保护（定时器、发生中断时操作系统通过 **ISR** 获得控制权）

程序应用只能通过**系统调用**请求操作系统提供的服务，界面中的系统调用因操作系统而异，也称为**主管呼叫**

如何**陷阱**进入操作系统：

1.异常
2.特殊指令

操作系统结构：
1.简单结构：MS-DOS
2.分层结构:THE、IBM OS/2
3.虚拟机:IBM VM/370
4.微内核：Mach、QNX
5.混合系统：Mac OS X、iOS、Android
6.模块:Solaris

虚拟机实现方式：
1.仿真
2.半虚拟化
3.完整虚拟化

一个进程不仅仅是一个程序，还包括了：
文本段、数据段、寄存器、堆栈、堆

程序是一个**被动**实体，进程是一个**活动**实体

进程的**状态**：
1.创建态 2.运行态 3.就绪态 4.等待态 5.结束态

每个进程在操作系统中都由**进程控制块（PCB）**表示。

和 **CPU 调度**有关的有：
1.从运行态到就绪态（抢占）
2.从等待态到就绪态（抢占）
3.从运行态到等待态
4.从运行态到结束态

进程协作有几个优点：
1.信息共享
2.计算加速
3.模块化

协作进程的并发执行需要允许进程相互**通信并同步**其操作的机制。

进程是
1.**资源**分配单位;（空间）
2.**调度**（scheduling）单位。（时间）

线程共享属于同一进程的资源，例如**其代码部分、数据部分、打开的文件**等。

但是，一个进程中的每个线程都有一个**私有**线程上下文（包括 **CPU 寄存器集和其他状态信息**）和一个私有**堆栈**。

线程优点：

1.可响应性

2.资源共享

3.经济

4.多处理器体系结构的利用

多线程：

1.在用户空间适用用户线程;

2.在内核中用于内核线程。

调度是操作系统的**基本功能**。

进程执行包括 **CPU 执行和 I/O 等待的周期**。

CPU 调度又称为**短期调度**

调度算法：

1.先来先服务

2.短作业优先（最短的平均等待时间）

3.优先级调度

4.轮转调度

5.多级反馈队列调度

静态优先级导致的问题：

1.饥饿 2.优先级反转

解决方法：

老化

临界区是**访问共享资源**的一段代码。

竞争条件解决方案必须满足：

1.互斥

2.前进性

3.有限等待

4.速度

硬件同步：

1.禁用中断（CLI、STL） 2.特殊指令（TSL 和 SWAP 以原子的方式执行）

信号量“值”的大小是可用资源数 （>0） 或等待该信号量的进程数 （<0）。

PV 操作必须以**原子**的方式执行，通过下面两种方式：
1.禁用单处理器系统中的中断
2.多处理器系统中的**自旋锁**

信号量经典问题:
1.生产者消费者问题（有限缓冲区问题）
2.读者写者问题
3.哲学家进餐问题

使用资源的操作：
1.请求 2.使用 3.释放

死锁特点：
1.互斥
2.保留并等待
3.不剥夺（无抢占）
4.循环等待

死锁恢复：
1.资源抢占
2.进程终止（回滚）

多任务环境下会带来许多**内存管理问题**：
1.重定位（relocation）问题；
2.内存保护（protection）问题；
3.内存分配（allocation）问题。

进行内存的分配和释放以减少外部碎片，提高内存使用率的算法：
1.首次适应（First fit）：分配第一个足够大的洞。
2.最佳适应（Best fit）：分配足够大的最小孔。
3.最坏适应（Worst fit）：分配最大的孔。

外部碎片和内部碎片

- **物理内存**被分解为固定大小的块，称为**帧**（frames）。
- **逻辑内存**也被分解成相同大小的块，称为**页**（pages）。
- 页表（Page tables）用于将页映射到帧。
- 页表的条目称为页表条目（PTE）（Page Table Entry）。

把逻辑地址分成两部分：
第一部分称为 page number（页码）
第二部分称为 page offset（偏移量）

地址转换：

在 page table 的帮助下，**MMU** 把 CPU 产生的逻辑地址转换成物理地址。

为每一个进程保存一个 page table

因此在分页中，每一个内存访问都需要**两次**内存操作

为了提高地址转换效率，MMU 中包含了一个**高速缓存**称为 translation look-aside buffers(TLBs)

有效访问时间

EAT = （1 + ε）α + （2 + ε）（1 − α） = 2 + ε − α

保护信息通常都保存在 **PTE** 中，

此外，不是所有的 PTE 都可以使用。因此，PTE 中的一位表示该 PTE 是否可以使用**（valid/invalid）**，

- 仅当该位有效时，MMU 才能用它进行地址转换，
- 否则，MMU 将通过异常向 OS 报告错误。

该位无效有两种情况：

1.不合法 2.合法但不在内存

现有**页表结构**：

1.层次型页表 （Hierarchical Page Tables）

2.哈希页表 （Hashed Page Tables）

3.倒排页表（Inverted Page Table）

分段内存管理：

逻辑地址由两部分组成：<段号，偏移>

段页式内存管理的地址转换包括两个步骤：**先分段，再分页**

虚拟内存可以通过以下方式实现：

1.按需分页（Demand paging）

2.需求细分（Demand segmentation）

按需分页优点：

- 所需的 I/O 更少;
- 需要更少的内存;
- 更快的响应;
- 更多流程;

辅助内存：此内存保存主内存中不存在的页面。它通常称为**交换空间**（swap space）;

我们可以通过将每个页面关联为**修改位（modify）（或脏位 dirty）**来**减少页面置换开销**

四种**页面置换算法：**
1.先进先出页面替换（FIFO）
2.最佳页面替换（Optimal page replacement）
3.LRU 页面替换
4.二次页面替换机会（second-chance page replacement）

Thrashing（抖动）导致：
• CPU 利用率低。
• 准入调度器认为它需要提高多进程并发的程度。
• 更多进程将添加到系统中。

抖动原理：局部模型（locality model）

工作集模型是**局部性模型的近似值**。

文件操作：
创建、打开、关闭、读取、写入、搜索、删除

访问控制列表 （ACL）（Access Control List） 指定用户名和每个用户允许的访问类型。

三大分配方法：
连续分配（contiguous allocation）
链接分配（linked allocation） ——FAT 文件系统
索引分配（indexed allocation）

空闲空间管理
1.位向量（位图（bit map））
2.链表