进程
与处理机管理

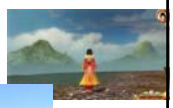
主讲教师：赵霞

小测验

- 1. 下列选项中，不可能在用户态发生的事件是
  - A. 系统调用 B. 外部中断 C. 进程切换 D. 缺页
- 2. 在支持多线程的系统中，进程P 创建的若干个线程不能共享的是
  - A. 进程P 的代码段　　B. 进程P中打开的文件
  - C. 进程P 的全局变量　　D. 进程P中某线程的栈指针
- 3. 下列选项中，导致创建新进程的操作是
  - I. 用户登录成功　　II. 设备分配
    III. 启动程序执行
  - A. 仅I和II　B. 仅II和III　　C. 仅I和III
    D. I、II和III

OS怎么管理处理器？

- 在OS构造的虚拟世界中，需要一个“活”的角色
  - 能够获得资源，执行一个应用程序
    - 占用处理器，执行程序
    - 占用内存，存放自己的代码、数据
    - 占用各种设备，有各种属性
  - 有生命周期
    - 创建、睡眠、终止
    - 活着的时候，占用资源
    - 终止的时候，释放资源
    - 能够繁衍和派生，继承资源
  - 能够和其他角色通信
    - 交换数据
    - 合作完成一些任务
    - 甚至能够在操作系统之间迁移



为了在计算机中同时运行多个程序，OS创造出了“进程”的概念

进程与处理机管理

- 进程描述
- 进程状态
- 进程控制与调度
- 作业与进程的关系
- 线程的引入

课程概要

- 目的与要求
  - 理解进程概念
  - 掌握进程在系统中的表示方法
  - 理解进程的创建及其状态变化
- 重点与难点
  - 进程表示与进程创建
  - 状态转换
- 作业
  - 思维导图，总结本章内容
  - 要求画出本人的特色

课后阅读与思考

- 教材
  - 第3章
- Operating System Concepts (6th edition)
  - Chapter 4 Process
  - Chapter 6 CPU Scheduling
  - Chapter 5 Thread
- Modern Operating System (2nd edition)
  - Section 2.1, 2.2, 2.5, 2.6

进程描述

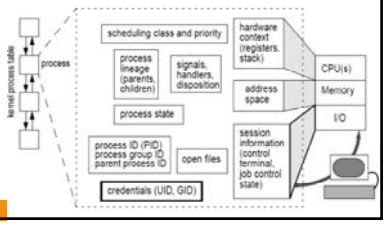
- 进程的概念
  - 是系统进行资源分配和调度的独立单位
- 进程的引入
  - 为了能在作业内某作业步等I/O时，另一逻辑上可并行作业步能使用CPU
  - 逻辑上可并行的作业步可在不同进程中运行
  - 处理机能在进程间切换
  - 原多道程序设计系统中的作业可看成是只有一个进程的作业

进程描述

- 进程
  - 是一个有独立功能的程序关于某个数据集合的一次运行活动
- 特点
  - 动态性
    - 可动态创建，结束
  - 并发性
    - 可以被独立调度占用处理机运行
  - 独立性
    - 尽量把并发事务安排到不同的进程
  - 制约性
    - 因访问共享数据或进程间同步而产生制约

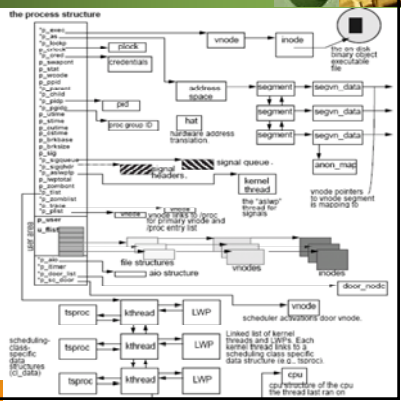
进程描述

- 进程 =进程控制块 +程序 + 数据 + 执行上下文
  - 操作系统 表示/管理进程的 PCB表
  - 执行程序及处理数据
  - 一片存放程序和数据的空间
  - 一个栈区（一个用户栈，一个核心栈）；
  - 进程使用的其他系统资源



进程控制块的内容

- 进程标识信息
  - PID
- 进程上下文（处理机状态信息）
  - 保存进程的运行现场信息：
    - 通用寄存器
    - 控制和状态寄存器
- 进程控制信息
  - 优先权，状态



进程与处理机管理

- 进程描述
- 进程状态
- 进程控制与调度
- 作业与进程的关系
- 线程的引入

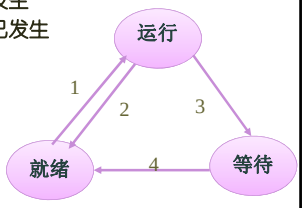
进程状态

- 运行态（Running）：
  - 进程占有CPU，并在CPU上运行
- 就绪态（Ready）：
  - 一个进程已经具备运行条件，但由于无CPU暂时不能运行的状态（当调度给其CPU时，立即可以运行）
- 等待态（Blocked）：
  - 阻塞态、封锁态
  - 指进程因等待某种事件的发生而暂时不能运行的状态
  - 即使CPU空闲，该进程也不可运行

进程的状态及其转换

进程状态转换：对用户是透明的

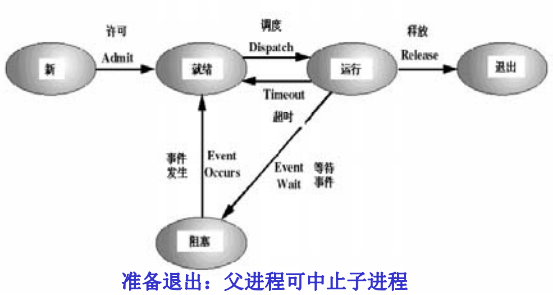
1 就绪—运行：进程被调度程序选中
2 运行—就绪：时间片用完
3 运行—等待：等待某事件发生
4 等待—就绪：等待的事件已发生



进程的状态及其转换

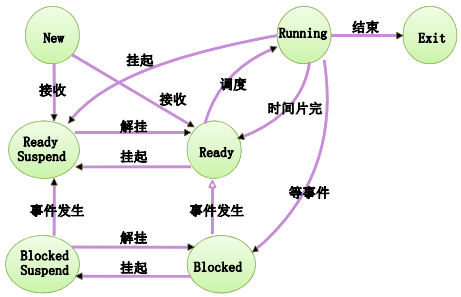
- 其他状态：
  - 创建状态
    - OS 已完成为创建一进程所必要的工作
      - 已构造了进程标识符
      - 已创建了管理进程所需的表格
    - 但还没有允许执行该进程
      - 因为资源有限
  - 终止（退出exit)状态
    - 中止后进程移入该状态
    - 它不再有执行资格
    - 当数据不再需要后，进程(和它的表格)被删除

五状态进程模型



准备退出：父进程可中止子进程

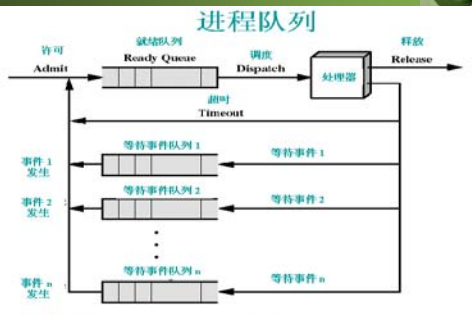
有挂起的进程状态变化图



进程创建

- 进程创建过程
  - 接收进程运行初始值，初始优先级，初始执行程序名字，其它资源等参数
  - 请求分配进程描述块PCB空间，得到一个内部数字进程标识
  - 用执行“进程创建”进程传来的参数初始化PCB表
  - 产生描述进程空间的数据结构，用初始执行文件初始化
  - 用进程运行初始值设置处理机现场保护区（如设置用户程序入口执行的运行现场）
  - 造一个进程运行栈帧
  - 置好父进程等关系域
  - 将PCB表挂入就绪队列，等待时机被调度运行程空间，建立程序段，数据段、栈段等

进程创建后的队列结构



- 就绪队列无优先级（例：FIFO）
- 当事件n发生，对应队列移进就绪队列

进程的结束

- 进程结束过程
  - 将进程状态改到结束状态
  - 关闭所有打开数据文件、设备
  - 释放对进程程序文件的使用
  - 进行相关信息统计
  - 清理其相关进程的链接关系，如在UNIX中，将该结束进程的所有子进程链到1号进程，作为1号进程的子进程，并通知父进程自己已结束
  - 释放进程映像空间（对于虚存来说如：交换区，所占物理页，将页表所占空间返还系统）
  - 释放进程控制块(PCB)。
  - 调用进程调度与切换程序

进程与处理机管理

- 进程描述
- 进程状态
- 进程控制与调度
- 作业与进程的关系
- 线程的引入

进程控制

- 进程执行
  - 系统参考模型：内核嵌入进程运行模型
- 执行模式
  - 进程可在用户态和核心态下运行
- 进程模式切换
  - 用户进程运行用户态程序
  - 在异常、系统调用和中断时转换到核心态时运行操作系统核心程序
- 进程切换
  - 指进程进入操作系统核心后，因为自身等事件或有更迫切需要运行的进程就绪而让出处理机
  - 处理机转去运行其他进程

进程切换过程

- 保存处理机的上下文，包括程序计数器PC、处理机状态字PSW、其它寄存器等处理机现场
- 修改当前运行进程的进程控制块内容，包括将进程状态从运行态改成其它状态
- 选择另一个进程执行，这是进程调度的内容
- 修改被调度进程的进程控制块，包括把其状态改变到运行态
- 修改存储管理数据结构，如修改进程内存起始地址，或将系统当前运行进程页表指针改为指向选定的进程页表
- 恢复被选进程上次切换出处理机时的处理机现场，按原保护的程序计数器值重置程序计数器，运行新选进程

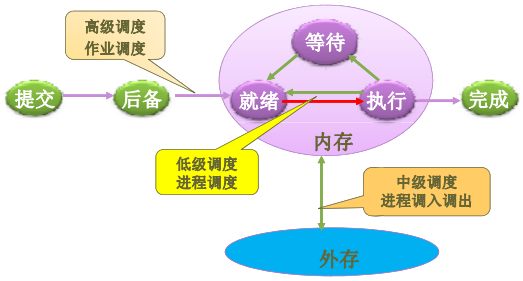
进程调度

- 选择进程占用处理机
- 调度概念
  - 操作系统管理系统的有限资源，当有多个进程（即多个进程发出的请求）要使用这些资源时，必须按照一定的原则选择进程（请求）来占用资源
- 调度目的
  - 控制资源使用者的数量，选取资源使用者占用资源

多级调度

- 高级调度
  - 选取输入井中的作业（仅限于批作业调度），生成根进程
  - 目的是控制使用系统资源的进程数
- 中级调度
  - 选取进程占用内存或有资格占用内存
  - 又称进程调入调出
- 低级调度
  - 选取进程占用处理机
  - 又称进程调度

多级调度



进程调度方式

- 剥夺调度（preemptive）
  - 当进程运行时可以被系统以某种原则剥夺其处理机
- 非剥夺调度（nonpreemptive）
  - 只有当处理机上的进程主动放弃处理机时才重新调度
- 进程调度在核心态进行

引起进程调度的因素

- 进程主动放弃处理机时
  - 正在执行的进程执行完毕
    - 操作系统在处理进程结束系统调用后应请求重新调度
  - 正在执行的进程发出I/O请求
    - 当操作系统启动外设I/O后，在I/O请求没有完成前要将进程变成阻塞状态，应该请求重新调度
  - 正在执行的进程要等待其它进程或系统发出的事件时
    - 如等待另一个进程通讯数据，这时操作系统应将现运行进程挂到等待队列，并且请求重新调度
  - 正在执行的进程得不到所要的系统资源
    - 如要求进入临界区，但没有得到锁时，这时等锁的进程应自动放弃处理机或者阻塞到等锁队列上，并且请求重新调度

引起进程调度的因素

- 为了支持可剥夺进程调度方式，在以下情况发生时，也应该申请进行进程调度：
  - 当中断处理后
    - 如I/O中断、通讯中断，引起某个阻塞进程变成就绪状态时，应该请求重新调度
  - 当进程释放资源，出临界区，引起其他等待该资源进程从阻塞状态进入就绪状态时，应该请求重新调度
  - 当进程发系统调用后，引起某个事件发生，导致等待事件的进程就绪时
  - 其它任何原因引起有进程从其它状态变成就绪状态，如进程被中级调度选中时

引起进程调度的因素

- 为了支持可剥夺调度，即使没有新就绪进程，为了让所有就绪进程轮流占用处理机，可在下述情况下申请进行进程调度：
  - 当时钟中断发生
    - 时钟中断处理程序调用有关时间片的处理程序，发现正运行进程时间片到
  - 在按进程优先级调度的操作系统中，任何原因引起进程的优先级发生变化时，应请求重新调度
    - 如进程通过系统调用自愿改变优先级
    - 或者，系统处理时钟中断时，根据各进程等待处理机的时间长短而调整进程的优先级

调度与切换时机

- 当发生引起调度条件，且当前进程无法继续运行下去时
  - 如发生各种进程放弃处理机的条件，马上进行调度与切换
- 当中断处理结束或自陷处理结束返回被中断进程的用户态程序执行前，若请求调度标志置上，即可马上进行进程调度与切换
  - 如果操作系统支持这种情况下运行调度程序，即实现了剥夺方式的调度
- 实时系统还有其他调度与切换时机

进程调度算法

- 进程调度
  - 控制协调进程对CPU的竞争，按一定的调度算法从就绪队列中选中一个进程，把CPU的使用权交给被选中的进程
- FCFS
  - 按照进程进入就绪队列的先后次序选择，按照先进先出的原则
    - 优点:实现简单
    - 缺点:没考虑进程的优先级
- 短进程优先
  - 取一个下次所需运行时间最短的进程
  - 该算法能使平均等待时间最短

进程调度算法

- 时间片轮转法（RR—Round Robin）
  - 把CPU划分成若干时间片,并且按顺序赋给就绪队列中的每一个进程，进程轮流占有CPU
  - 当时间片用完时，未执行完毕的进程排在就绪队列末尾
  - 选择另一个进程运行

进程调度算法

- 与时间片大小有关的因素：
  - 时间片 = 响应时间 / 进程数目
  - 系统响应时间：
    - 进程数目一定，时间片与响应时间成正比
  - 就绪进程个数：
    - 响应时间一定，时间片与进程数目成反比
  - CPU能力：处理速度高，则时间片可以设置小些
- 时间片选择问题：
  - 固定时间片：多数微机采用，简单易行
  - 可变时间片：根据进程的优先数来设定时间片的大小

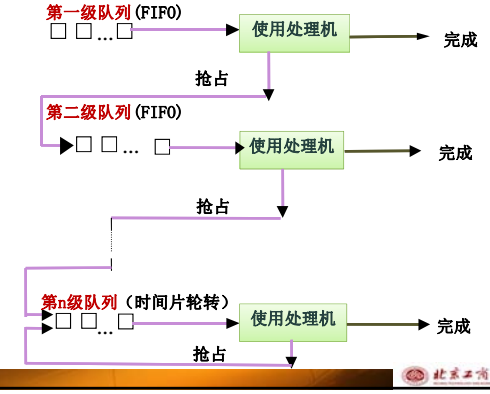
进程调度算法

- 优先级调度：(HPF—Highest Priority First)
  - 优先选择就绪队列中优先级最高的进程投入运行
  - 确定优先级的方法
    - 静态优先级
      - 在进程创建时指定优先级，在进程运行时优先级不变
    - 动态优先级
      - 在进程创建时创立一个优先级，但在其生命周期内优先级可以动态变化
      - 最高响应比优先级调度
        - » 优先数 = （等待时间 + 要求的服务时间）/ 要求的服务时间
  - 两类
    - 可剥夺的优先级调度
    - 非剥夺的优先级调度

进程调度算法

- 多队列调度法
  - 按属性将就绪进程分类
  - 不同类进程可有不同的调度算法
- 多级反馈队列调度法
  - 设置多条就绪队列
  - 进程被调度执行后被剥夺或放弃处理机后
  - 在就绪时改变其就绪队列

多级反馈队列调度法



调度算法的选取原则

- 系统的设计目标
  - 效率，公平
- 处理器利用率
  - 提高效率
- 吞吐量
  - 单位时间让更多的进程完成工作
- 等待时间
  - 减少进程在就绪队列中的等待时间
- 响应时间
  - 在交互式系统中尽快响应用户请求

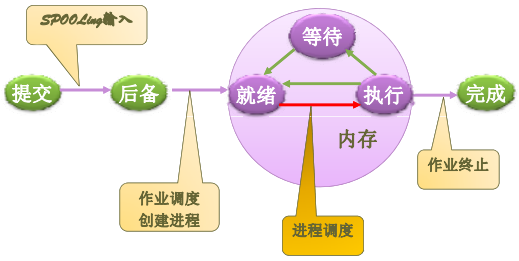
进程与处理机管理

- 进程描述
- 进程状态
- 进程控制与调度
- 作业与进程的关系
- 线程的引入

进程与作业

- 作业
  - 是用户对计算机的一次独立的使用过程
- 进程
  - 是分配计算机资源的单位
  - 是用户任务运行的实体
  - 一个作业可包含多个进程(至少一个)
- 批处理系统作业与进程关系
  - 作业调度程序每选择一道作业运行时，首先为该作业创建一个根进程
  - 该进程执行作业控制语言解释器程序，并可根据需要创建多个子进程

作业和进程状态转换图



分时系统作业与进程之关系

- 把用户的一次上机过程看成是一个交互作业
  - 无论从内部表示及外部特征，它都有别于批作业
  - 系统为每个终端设备生成一个进程
  - 该进程运行终端命令解释器
  - 该进程根据需要还可以创建多个子进程
- 支持分时与批处理的系统作业提交方法：交互式命令
  - 如:at -f /root/bin/ss now
    - 表示提交一个作业控制说明书文件名为ss的作业到作业输入队列
  - 或 直接输入“shell ss”
    - 表示马上生成一个进程执行命令解释器，解释执行ss中的命令

进程与处理机管理

- 进程描述
- 进程状态
- 进程控制与调度
- 作业与进程的关系
- 线程的引入

线程的引入

- 进程的缺点
  - 进程独立占用资源，不易共享
  - 进程切换时间空间开销大，限制并发度的提高
- 线程
  - 轻量级进程(Light Weight Process)
  - 进程中的一个实体
  - 是CPU调度的单位
  - 同进程的线程共享进程拥有的所有资源

线程的引入

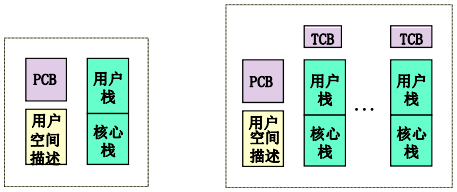
- 引入线程的好处：
  - 管理开销小
    - 创建新线程花费时间少（结束亦如此）
    - 两个线程的切换花费时间少
  - 同一用户进程内的用户线程间相互通信无须调用内核
  - 线程可以独立执行，能充分利用和发挥处理器与外部设备并行工作能力
    - 发挥多核处理器的优势

线程的概念

- 线程
  - 有线程上下文（寄存器信息）
  - 有一个执行栈
  - 有生命周期的状态
    - 就绪、等待和运行
  - 有一些局部变量的静态存储
  - 可存取所在进程的内存和其他资源
  - 可以创建、撤消另一个线程

线程的概念

- 属性
  - 唯一标识符和一张线程描述表
    - 记录线程执行的寄存器及栈等现场状态
  - 同一进程中的各个线程共享该进程的内存地址空间
  - 线程是处理器的独立调度单位，多个线程可以并发执行



a. 传统进程模型　　b. 多线程进程模型

线程与进程的比较

- 地址空间和其他资源（如打开文件）
  - 进程间相互独立
  - 进程的各线程间共享
- 通信
  - 进程间通信IPC
  - 线程间可以直接读写进程数据段（如全局变量）来进行通信
    - 需要进程同步和互斥手段的辅助，以保证数据的一致性

线程与进程的比较

- 调度
  - 线程上下文切换比进程上下文切换要快得多
  - 同进程中的线程切换不会引起进程的切换，不同进程中的线程切换会引起进程的切换
- 并发性
  - 一个进程中可以有多个线程并发执行，提高系统资源的利用率
- 系统开销
  - 系统要为进程分配或回收资源，其开销将大于为线程创建和撤销的开销

小结

- 进程的描述
  - 进程的概念
- 进程状态
  - 状态转换
- 进程控制与调度
  - 进程创建和结束，进程调度算法，进程切换
- 进程与作业的关系
- 线程
  - 引入原因，与进程比较