什么是进程

狭义定义: 进程就是一段程序的执行过程(或者我们可以简单的理解为进行中的程序)

广义定义: 进程是一个具有一定独立性功能的程序关于某个数据集合的一次运行活动. 它是操作系统动态执行的基本单元, 在传统的操作系统中, 进程是基本的分配单元, 也是基本的执行单元 (后续的内容会帮助理解)

进程结构

进程的结构由 3 部分组成:

代码段(程序)

数据段(数据)

堆栈段(进程控制块 PCB)

—PCB: PCB 是进程存在的唯一标识, 因为系统是通过 PCB 的存在来感知进程的存在 (后续会对 PCB 进行一个详细的总结)—

进程特征

动态性: 进程的实质是程序在多道程序系统中的一次执行过程, 动态产生, 动态消亡

独立性: 每个进程都有独立的 PCB 和 进程地址空间, 因此进程之间相互独立, 多个进程之间互不影响

结构性: 进程由 3 个部分组成(以上)

竞争性: 进程多, 但是资源有限, 因此每个进程之间相互竞争, 相互制约

并行: cpu 资源足够, 多个进程同时运行

并发: cpu 资源不足, 多个进程切换运行, 同时推进

进程状态

就绪状态(Reading): 进程已经获得了除了处理器以外的所有资源, 等待分配处理器资源, 只要分配好了处理器资源, 进程就进入了运行状态. 就绪进程可以按照不同的优先级来划分队列, 比如时间片用完排入低优先队列, 由 I/O 操作完成排入高优先级队列.

–时间片:分时操作系统为进程分配的一段运行时间–

运行状态(Running): 进程占用处理器资源进行运行, 在没有其他进程运行时(或进程都处于阻塞状态), 通常会自动执行系统的空闲进程.

阻塞状态(Blocked): 进程等待某种条件, 比如说 I/O 操作和进程同步, 在条件满足之前, 进程无法执行, 即时为其分配处理器资源

引入进程的意义

理论角度: 对正在运行的程序的过程进行抽象

现实角度: 作为一种数据结构, 能清晰的刻画动态系统的内在规律, 有效的调度和管理进入计算机系统主存运行的程序

进程与程序

区别

1.状态: 进程是程序在处理机上的一次执行过程, 是动态的概念; 程序是指令和数据的有序集合, 是静态的概念.

2.生命周期: 进程是动态的, 因此有一定的生命周期; 程序是静态的, 因此它没有生命周期

3.功能: 进程可以创建其他进程(子进程); 程序没有类似的功能.

进程是资源分配和独立运行的基本单元; 程序则不能独立运行.

联系

程序是进程的一部分, 并且同一个运行的程序可以属于多个进程, 即同一程序可以对应多个进程.

当操作系统发现了引起进程创建的事件之后, 便调用进程创建原语 Creat() 进行一系列步骤创建进程:

1 申请空白 PCB: PCB 作为进程的唯一标识符最先被创建, 从 PCB 集合(RAM内存)中索取一块空白 PCB.

2.为进程分配资源: 分配进程地址空间

3.初始化 PCB:

4.将进程插入就绪队列

进程终止

当操作系统发现引起进程终止的事件之后, 便调用进程终止的原语, 进行一系列步骤终止进程:

1.根据进程终止的标识符, 从 PCB 中读取进程状态

2.将进程以及进程的子孙进程终止

3.将终止进程的所有资源归还给父进程和操作系统

4.将 PCB 从进程队列中移除

当操作系统发现引起进程阻塞的事件后, 便调用进程阻塞原语block() 把自己阻塞

1.将 PCB 从进程队列移出

2.将 PCB 中进程状态改为阻塞

3.将 PCB 插入阻塞队列

当操作系统发现引起进程唤醒的事件后, 便调用进程唤醒原语wakeup() 将等待该事件的唤醒

1.将 PCB 从阻塞队列移出

2.将 PCB 中进程状态改为就绪

3.将 PCB 插入就绪队列