山东大学 计算机科学与技术 学院

操作系统 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：201900130059 | 姓名： 孙奇 | | 班级： 2019级1班 |
| 实验题目：进程控制 | | | |
| 实验学时：2 | | 实验日期：2021/10/15 | |
| 实验目的：  加深对于进程并发执行概念的理解。实践并发进/线程的创建和控制方法。观  察和体验进程的动态特性。进一步理解进程生命期期间创建、变换、撤销状态变换  的过程。掌握进程控制的方法，了解父子进程间的控制和协作关系。练习 Linux 系  统中进/线程创建与控制有关的系统调用的编程和调试技术。 | | | |
| 硬件环境：  CPU: Intel i5-9300H  GPU: UHD630 | | | |
| 软件环境：  Ubuntu 18.04 | | | |
| 实验步骤与内容：   1. 进程实验说明：   进程可以通过系统调用 fork() 创建子进程并和其子进程并发执行。子进程初始的执行映像是父进程的一个副本：子进程会复制父进程的数据与堆栈空间，并继承父进程的用户代码、组代码、环境变量、已打开的文件描述符、工作目录和资源限制等。子进程可以通过系统调用族 exec() 装入一个新的执行程序。父进程可以使用wait() 或 waitpid() 系统调用等待子进程的结束并负责收集和清理子进程的退出状态。   1. 独立实验：   参考以上示例程序中建立并发进程的方法，编写一个多进程并发执行程序。父进程每隔 3 秒重复建立两个子进程，首先创建的子进程让其执行 ls 命令，之后创建的子进程让其执行 ps 命令，并控制 ps 命令总在 ls 命令之前执行。   1. 实现思路：   两层循环先后创建两个子进程’ls’和’ps’，子进程创建后pause()挂起，在父进程中依次调用kill先后唤醒子进程’ps’和’ls’，父进程waitpid()等待各子进程执行结束后，回收子进程资源，然后继续执行父进程，完成程序； | | | |
| 结论分析与体会：   1. 实验反映出的进程的特征和功能： 2. 并发性：父进程与子进程、子进程之间可以同时执行，分别完成自己的任务； 3. 动态性：进程是动态产生、动态消亡的，同时进程的运行状态处于经常性的动态变化中； 4. 独立性：进程是调度的基本单位，在执行中是独立的，能够参与并发执行； 5. 交互性：进程在运行过程中可与其他进程发生直接间接的相互作用； 6. 真实操作系统中反映的进程生命期、实体、状态控制： 7. 进一步理解进程概念和并发概念： 8. 子进程的创建和执行原理： 9. 信号的机理： 10. 信号实现的进程控制： | | | |

附录：程序源代码