

**实 验（实训）报 告**

**项 目 名 称**  进程控制和通信

**所属课程名称**  操作系统

**项 目 类 型**  验证/设计型

**实验(实训)日期**  2024年11月13日

**班 级**  22软工3班

**学 号**  220110900601

**姓 名**  陈俊杰

**指导教师**  陈伟锋

浙江财经大学教务处制

|  |
| --- |
| **一、实验（实训）概述：** |
| **【目的及要求】**  安装gcc  完成hello.c  完成fork.c  完成proceses.c  完成communication.c  进程调度模拟  用C语言实现对N个进程采用动态优先权算法的调度  每个标示进程的进程控制看PCB用结构体来描述，包括  进程id  进程priority，值越大，优先级越高  已占用CPU时间cputime  还需占用CPU时间alltime，进程运行完成时，alltime=0  进程阻塞时间startblock，即进程每运行startblock时间片后，进入阻塞状态  进程被阻塞时间blocktime，即进程等待blocktime时间片后，进入就绪状态  基础状态state（枚举：READY/BLOCK/RUNNING，初始默认为READY)  优先级变化规则  就绪队列中经过一个时间片，+1  阻塞队列中经过一个时间片，0（不变）  运行一个时间片，-3  **【基本原理】**  GCC(GNU Compiler Collection)即GNU编译器套件，属于一种编程语言编译器，其原名为GCC（GNU C Compiler）即GNU c语言编译器，虽然缩写一样但是功能上区别很大。GCC的初衷是为GNU操作系统专门编写的一款编译器，原本的GNU是专用于编译C代码，现如今已扩展为可以编译C、C++、Java、Objective-C等多种编程语言的编译器集合了。  **【实施环境】**  **Ubuntu 22.04** |
| **二、实验（实训）内容：** |
| **【实验（实训）过程】（步骤、记录、数据、程序等）**  **（一）安装GCC**  1.安装GCC和G++：    命令：“gcc --version”  vim  **（二）完成hello.c**  Vim编写c语言文件    加权限    执行    **（三）完成fork.c**  **1.按照教程将代码打好：**    编译    执行和输出    运行结果    运行fork.c代码后，创建子进程成功，执行子进程中“execlp("/bin/ls", "ls", NULL);”命令，即列出当前目录下的所有文件，最后子进程执行完毕，父进程输出“Child Complete”。   1. **完成Processes.c**   编写文件      编译    执行和输出        进程的创建过程：  当程序执行到 pid = fork() 时，会创建一个新的子进程。  在父进程中，fork() 返回子进程的 PID（即一个正整数），而在子进程中，fork() 返回0。  父进程和子进程会从 fork() 返回的地方开始执行，但它们的执行路径是不同的。  在这里，变量 value 被初始化为0，并且在父进程和子进程中都被打印出来。由于两个进程共享同一份内存，虚拟地址，它们看到的 value 的值是一样的，都是0。然后，两个进程都对 value 进行了递增操作（value++），每次递增后都再次打印了 value 的值和地址。由于两个进程是并发执行的，所以它们交替打印输出。  **（五）完成communication.c**  编写代码      编译    执行并输出      总结：  在main函数中，首先打印父进程的PID，然后创建一个匿名管道。如果管道创建失败，则程序返回-1。  设置信号处理函数SignHandler1来处理SIGINT信号（通常由Ctrl+C发送）。对于子进程1，它被设置为忽略SIGINT信号，而对于子进程2，它被设置为接收到SIGINT信号时调用SignHandler2。  通过fork()函数创建两个子进程。子进程1在每次迭代中向管道写入一条消息，然后自增计数器。子进程2从管道中读取消息并打印它。这两个子进程都无限循环，除非它们接收到特定的信号。  进程调度模拟                    **优先级查找**    **阻塞加入等待队列**    阻塞队列检查    执行完成，移出队列          出现阻塞进程 |
| **三、指导教师评语及成绩：** |
| **评语：**  **成绩： 指导教师签名：**  **批阅日期：** |