《计算机操作系统实验指导》

（Linux版）（附微课视频）

# 实验思考解答

第1章 操作系统实验体系介绍

第1章无实验思考。

第2章 实验环境的搭建与使用

1. 实验2.1 Linux常用命令的使用

（1）在Linux 中，图形界面与终端控制台以及各终端控制台之间在如何切换？

使用ALT+F1~ALT+F6进行各终端控制台切换

2. 实验2.2 Linux下C程序的编写

1. make 工具的编译原理是什么？

make 是一个命令工具，它解释 Makefile 中的指令（规则）。在 Makefile文件中描述了整个工程所有文件的编译顺序、编译规则。在执行 make  之前，需要一个命名为 Makefile  的特殊文件（本文的后续将使用Makefile 作为这个特殊文件的文件名）来告诉 make 需要做什么（完成什么任务），该怎么做。

（2）如何直接使用gcc 命令完成myapp.c、greeting.h、greeting.c 三个文档的编译？

输入以下两条命令：

1. gcc –c greeting.c //生成greeting.o文件

② gcc –o greeting myapp.c greeting.o //把greeting.o和myapp.c一起编译成可执行文件greeting。

第3章 进程控制与进程调度

1. 实验3.1 进程的创建

（1）总结调用fork()函数后的三种返回情况。

fork()系统调用有3种返回情况：返回值>0，表示当前进程是父进程，这个返回值为子进程的进程ID值；等于0，表示当前进程是子进程；小于0，表示进程创建失败，需要报错。

（2）总结fork()和wait()配合使用的情况，并尝试在父进程中取消wait()函数，观察进程的运行情况。

当父进程中同时使用fork()和wait()/waitpid()函数时，父进程会处于阻塞状态等待子进程的运行结束。如果父进程中没有调用wait()/waitpid()函数，则父进程和其创建的子进程属于并发进程，也就是父进程和子进程几乎是独立运行的。

2. 实验3.2 进程调度算法的模拟

（1）上述示例调度程序中的调度模式是抢占式还是非抢占式？

非抢占式

（2）若要将上述示例调度程序中的进程运行方式改为每运行一次就将优先数减2，同时将运行时间加1，其他条件不变，则该如何修改？

在running()函数中，将(p->nice)--; 改为 p->nice = p->nice - 2;

（3）如何将上述调度算法改为固定优先数调度算法？

需要保持示例代码中进程的p->nice值保持不变，即在running()函数中，删除(p->nice)--;

第4章 进程通信与进程同步

1. 实验4.1 两个进程相互通信

（1）示例代码中随机数的取值对于模拟“石头、剪刀、布”游戏很重要，如果取值不当，就可能出现大量平局的情况，故请思考Linux随机数的合理取值方法。

有几种方法可以实现随机数的合理取值：①当以时间作为种子，取随机数时，需要将时间岔开，以保证随机数的合理。②以其他数值作为种子，如使用getpid()来获得进程PID。③用其他方法产生随机数。

（2）比较Linux 操作系统中的几种IPC机制，并说明它们各自适用于哪些场合。

管道：无名管道简单方便．但局限于单向通信的工作方式．并且只能在创建它的进程及其子孙进程之间实现管道的共享：有名管道虽然可以提供给任意关系的进程使用．但是由于其长期存在于系统之中，使用不当容易出错。

消息队列：消息缓冲可以不再局限于父子进程．而允许任意进程通过共享消息队列来实现进程间通信．并由系统调用函数来实现消息发送和接收之间的同步．从而使得用户在使用消息缓冲进行通信时不再需要考虑同步问题．使用方便，但是信息的复制需要额外消耗CPU的时间．不适宜于信息量大或操作频繁的场合。

共享内存：共享内存针对消息缓冲的缺点改而利用内存缓冲区直接交换信息，无须复制，快捷、信息量大是其优点。但是共享内存的通信方式是通过将共享的内存缓冲区直接附加到进程的虚拟地址空间中来实现的．因此，这些进程之间的读写操作的同步问题操作系统无法实现。必须由各进程利用其他同步工具解决。另外，由于内存实体存在于计算机系统中．所以只能由处于同一个计算机系统中的诸进程共享。不方便网络通信。

2. 实验4.2 进程同步实验

（1）多线程并发与多进程并发有何不同与相同之处？

在单核环境下，多线程并发时，如果这些线程属于同一个进程，就属于同一进程的运行，整个进程的执行效率提高。如果这些线程分属于不同的进程，就意味着多进程之间的并发。在支持多线程的系统中，多进程并发就是多线程并发，而多线程并发不一定是多进程并发。在多核环境下，多线程还可能并行。

第5章 内存管理

1. 实验5.2 页面置换算法的模拟

（1）分析比较各种页面置换算法之间的差异。

FIFO实现方便，缺页率可以较高；OPT性能最佳，但在现实中无法实现；LRU实现时较复杂，且需要硬件支持，现实中常用近似算法如LFU等，但性能较靠近OPT算法。

第6章 简单文件系统设计

1. 实验6.1：文件备份实验

（1）使用系统调用函数open()、read()、write()和close()实现简单文件备份的原理是什么？

首先分别打开源文件（只读方式）和目标文件（只写方式）。对于打开的源文件，每次读1个字节，然后写入目标文件，如此循环，直到文件结束。最后关闭两个文件。

（2）使用C 语言库函数fopen()、fread()、fwrite()和fclose()实现简单文件备份的原理是什么？

首先分别打开源文件（只读方式）和目标文件（只写方式）。对于打开的源文件，每次读1024个字节（示例代码中buf的长度），然后写入目标文件，如此循环，直到文件结束。最后关闭两个文件。

（3）上述两种方式的区别是什么？

这两种方法基本是类似的，主要区别是一个是Linux操作系统提供的系统调用，一个C语言提供的库函数；另外，从示例代码看使用C语言库函数进行文件备份的速度较快。

2.实验6.2 简单文件系统的模拟

（1）示例代码中没有给出明确的文件打开（open）操作和读（read）操作，请修改程序，实现文件打开操作和读操作，并说明在文件系统提供与不提供打开操作的情况下，读写文件时的不同。

当系统没有显式提供打开文件的操作时，需要每次读文件或写文件时，都需要判断该文件是否已经打开，如果未打开，则打开该文件，否则就直接执行读写操作。

第7章 编译内核

（1）总结内核编译过程中遇到的问题及相应的解决方案。

略

第8章 系统调用

（1）总结添加系统调用时出现的问题及其原因与解决方案。

略

第9章 虚拟内存管理

1. 实验9.1 统计系统缺页次数

（1）说明本实验中统计缺页次数的原理，并阐述其合理性。

本实验是通过修改内核源代码来实现的，基本原理是增加一个长整型变量pfcount（初值为0），用来统计缺页次数，在每次缺页时，对该变量的值增加1，输出该变量的值，即为缺页次数。这是从内核层面统计缺页次数，结果是合理的。

（2）总结实验过程中出现的问题及对应的解决方法。

略

2. 实验9.2 统计一段时间内的缺页次数

（1）如何验证实验结果的准确性？

验证该实验的结果可以借助于实验9.1，即可通过编写程序的方式，在某个固定时间输出系统缺页次数和/proc/vmstat中的pgfault字段的值，然后再过一段时间再输出系统缺页次数和/proc/vmstat中的pgfault字段的值，取两次的差值，相互验证实验结果是否一致。

（2）尝试使用更方便的方法读取/proc/vmstat 中的字段，如使用Python 编程或Shell 编程。

略

（3）总结实验过程中出现的问题及对应的解决方案。

略

第10章 内核模块编写

1.实验10.1 编写一个简单的内核模块

（1）总结并分析实验中出现的问题及对应的解决方法。

略

（2）如何实现将多个源文件合并到一个内核模块中？

2. 实验10.2 利用内核模块创建一个设备文件节点

（1）说明为什么内核源代码中的输出函数选用了printk()而不是常用的printf()。

printf()在终端显示,printk()函数为内核空间里边的信息打印函数，就像c编程时用的printf()函数一样，专供内核中的信息展示用。在内核源代码中没有使用printf()的原因是在编译内核时还没有C的库函数可以供调用。

（2）思考bar、foo、jiffies 和jiffies\_too文件分别是什么类型，它们是否可以进行读写。

bar和foo为可读文件，jiffies为只读文件，jiffies\_too为jiffies的链接文件（可读可写）。

（3）总结并分析实验中出现的问题。

略

第11章 文件系统设计

实验11 设计一个简单的文件系统

结合前面章节讲述的内核编译知识，将myext2 文件系统直接编译进内核，然后重启内核并挂载文件系统进行测试。

略。

第12章 设备管理

1. 实验12.1 编写字符设备驱动程序

（1）修改测试文件，实现向字符设备写数据。

int main()

{

char buf[4096] = {"I have already told you 1 time hello world"};

int fd =open("/dev/hello",O\_RDWR );

int ret = write(fd ,buf ,sizeof(buf));

buf[ret] = '\0';

printf("%s\n",buf);

}

（2）总结并分析实验中出现的问题及对应的解决方法。

略。

2. 实验12.2 编写块设备驱动程序

（1）总结并分析实验中出现的问题及对应的解决方法。

略。

（2）分析字符设备与块设备的驱动程序，指出它们在实现过程中的异同点。

相同点：设备驱动程序编写实现的大体框架相同，都是先要编写设备驱动源代码，然后使用make命令编译成内核驱动模块，再使用insmod命令安装模块，最后测试模块。

不同点：字符设备驱动的源代码与块设备驱动程序的源代码不同，Makefile文件内容不同。安装完模块后，这两个设备的使用方式也不同。

祝您学习愉快