

操作系统实验 4 进程的管道通信

计算机 1202 张艺瀚
学号:20123852

June 23, 2015

1 实验题目

编程实现进程的管道通信程序

2 实验目的

这是一个验证型实验。通过对给出的程序进行验证、修改，进一步加深理解进程的概念，了解同步和通信的过程，掌握进程通信和同步的机制，特别是利用缓冲区进行同步和通信的过程。通过补充新功能，加强对知识的灵活运用，培养创新能力。

3 实验要求

1. 加深对进程概念的理解，明确进程和程序的区别；
2. 学习进程创建的过程，进一步认识并发执行的实质；
3. 分析进程争用资源的现象，学习解决进程互斥的方法；
4. 学习解决进程同步的方法；
5. 掌握 Linux 系统进程间通过管道通信的具体实现方法。

4 程序流程图

见图 1

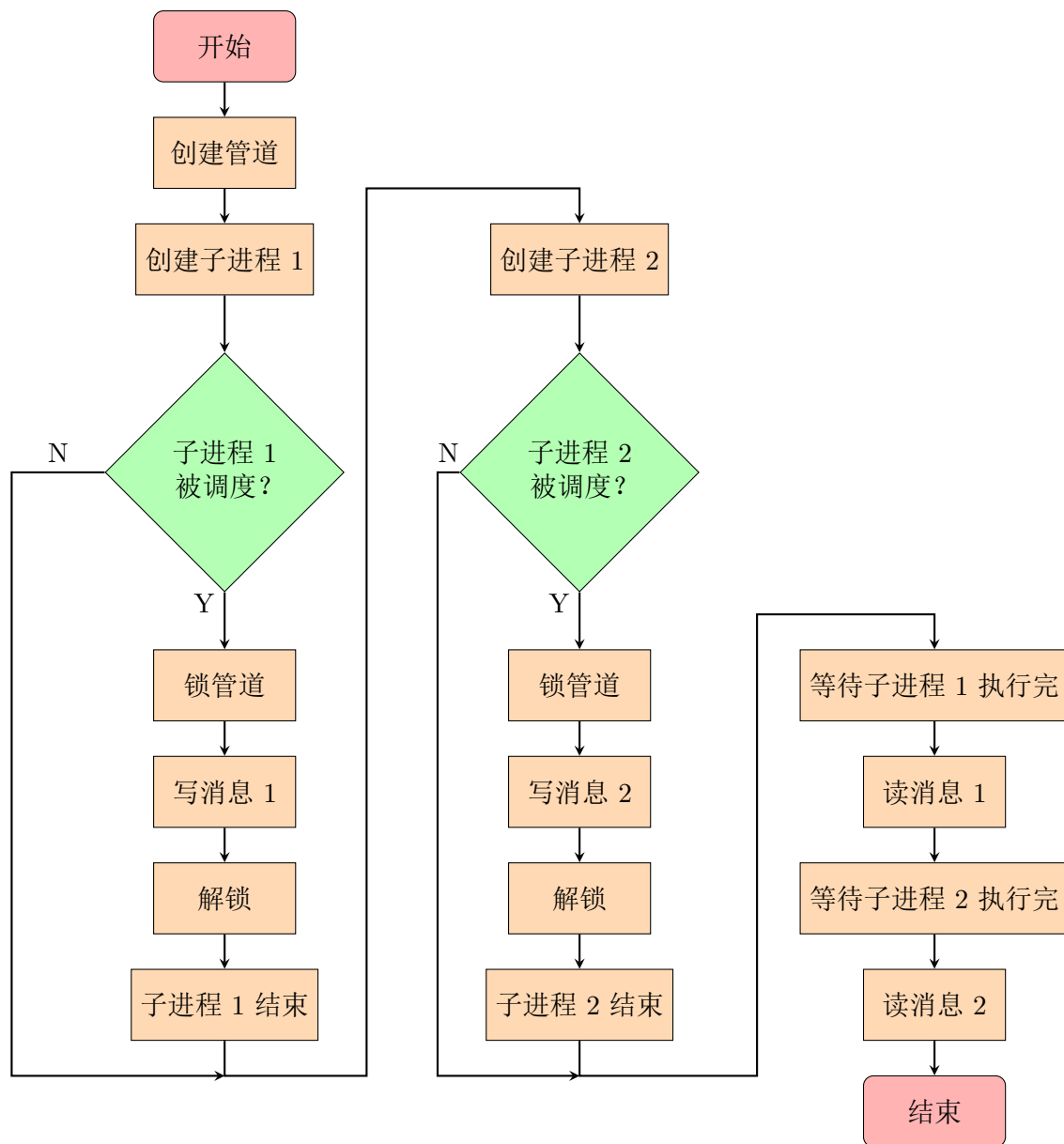


Figure 1: 主过程

5 源程序

```
1 #include <errno.h>
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include <sys/types.h>
5 #include <sys/wait.h>
6 #include <unistd.h>
7 int main()
8 {
9     pid_t pc1, pc2, pr1, pr2;
10    int fd[2];
11    char buf1[50], buf2[50], s[50];
12    pipe(fd);
13    while((pc1 = fork()) == -1);
14    if(pc1 < 0){
15        printf("create child process1 error: %s\n", strerror(errno));
16        exit(1);
17    }else if(pc1 == 0){
18        lockf(fd[1], 1, 0);
19        sprintf(buf1, "child process1 %d is sending a msg", getpid());
20        ;
21        write(fd[1], buf1, 50);
22        printf("in process1 %d, sending a msg\n", getpid());
23        lockf(fd[1], 0, 0);
24        exit(0);
25    }
26    while((pc2 = fork()) == -1);
27    if(pc2 < 0){
28        printf("create child process2 error: %s\n", strerror(errno));
29        exit(1);
30    }else if(pc2 == 0){
31        lockf(fd[1], 1, 0);
32        sprintf(buf2, "child process2 %d is sending a msg", getpid());
33        ;
34        write(fd[1], buf2, 50);
35        printf("in process2 %d, sending a msg\n", getpid());
36        lockf(fd[1], 0, 0);
37        exit(0);
38    }
39 }
```

```
37 printf("in parent process %d\n", getpid());
38 pr1 = wait(0);
39 if(pr1 > 0){
40     read(fd[0], s, 50);
41     printf("parent process %d received a msg: %s\n", getpid(), s)
42     ;;
43 }else{
44     printf("error: %s\n", strerror(errno));
45 }
46 pr2 = wait(0);
47 if(pr2 > 0){
48     read(fd[0], s, 50);
49     printf("parent process %d received a msg: %s\n", getpid(), s)
50     ;
51 }else{
52     printf("error: %s\n", strerror(errno));
53 }
54 return 0;
55 }
```

Listing 1: 代码清单

6 运行结果及其说明

运行结果见图 2。两个子进程互斥使用管道，父子进程之间同步传递消息。子进程的调度顺序随机，不受用户控制，有时会出现子进程 2 先于子进程 1 调度的情况。

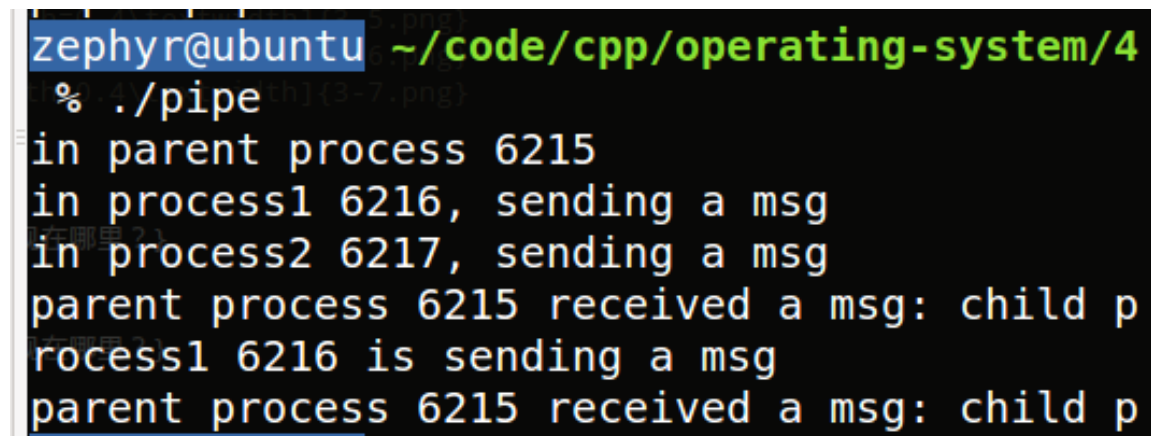
7 回答问题

7.1 进程间的互斥表现在哪里？

两个子进程都要对管道写消息，管道作为两个子进程共享的资源，应当互斥使用，每个子进程使用时要做互斥保护（使用前加锁，使用后解锁）。

7.2 进程间的同步表现在哪里？

父子进程之间传递消息，只有子进程写入后，父进程才能读出，父进程读子进程 i 的消息前要先等待子进程 i 执行完毕。



```
zephyr@ubuntu ~/code/cpp/operating-system/4
% ./pipe
in parent process 6215
in process1 6216, sending a msg
in process2 6217, sending a msg
parent process 6215 received a msg: child p
rocess1 6216 is sending a msg
parent process 6215 received a msg: child p
```

Figure 2: 运行结果

7.3 你的程序采用什么方法实现上述关系？

互斥：子进程写消息前使用 `lockf` 对管道加锁，写后用 `lockf` 解锁；同步：子进程结束后 `exit` 表执行完毕，父进程读消息前 `wait` 对应子进程。使用 `fork` 的返回值区分父进程和两个子进程的代码模块。