### 辽师

### 《操作系统实验二课程报告》

——Linux下多进程编程

### 

**学 院：计算机与信息技术学院**

**专 业：计算机科学与技术（师范）**

**班级序号：01班**

**学 号：201621012299**

**学生姓名：周惠馨**

**完成时间：2018.11.05**

1. **实验目的**
   1. 要求运用fork(),wait(),pipe(),read(),write()及execve()等系统调用编写多进程的程序。
   2. 完成一程序两进程、两进程两程序，一程序三进程的程序。
   3. 通过不同情况下运行结果，理解Linux系统下进程的调度，进程的并发执行以及父子进程之间的关系（代码段、数据段之间的关系）。
2. **实验环境**

Linux，Vi,CC编译器。

1. **实验内容**
   1. 编写一程序两进程的程序，父进程循环显示parent,子进程循环显示child.

#include <stdio.h>

main()

{ int pid,i;

while((pid=fork())== -1);

if(pid!=0)

{

while(1)

{

for(i=0;i<10000000;i++);

printf("parent");

}

}

else

{

while(1){

for(i=0;i<10000000;i++);

printf("child");

}

}

}

* 1. 运行下列程序，说明父子进程之间代码段与数据段之间的关系。

#include <stdio.h>

main()

{

int pid,n=0,i;

while((pid=fork())==-1);

if(pid!=0)

{

while(1)

{ printf("%8d\n",n++);

for(i=0;i<100000000;i++);

printf("parent");

}

}

else

{

while(1){ printf("%8d\n",n++);

for(i=0;i<100000000;i++);

printf("child");

}

}

}

* 1. 将程序1改写成两程序两进程，并运行。

参考如下程序：

//fork3.c

#include <stdio.h>

main()

{

int pid,n=0,i;

while((pid=fork())==-1);

if(pid!=0)

{

while(1)

{ printf("%8d\n",n++);

for(i=0;i<100000000;i++);

printf("parent");

}

}

else

execve("./sp",0,0);

}

//sp.c

#include <stdio.h>

main()

{int n=0,i;

while(1){ printf("%8d\n",n++);

for(i=0;i<100000000;i++);

printf("child");

}

}

* 1. 创建一程序三进程的程序，第一个子进程显示a,第二个子进程显示b,父进程显示c.

多次执行，并分析执行结果产生的原因。

#include <stdio.h>

main()

{ int pid1,pid2,i;

while((pid1=fork())==-1||(pid2=fork())==-1)

{ if(pid1!=0 && pid2!=0)

Printf(“c”);

else if(pid1==0)

Printf(“a”);

else if(pid2==0)

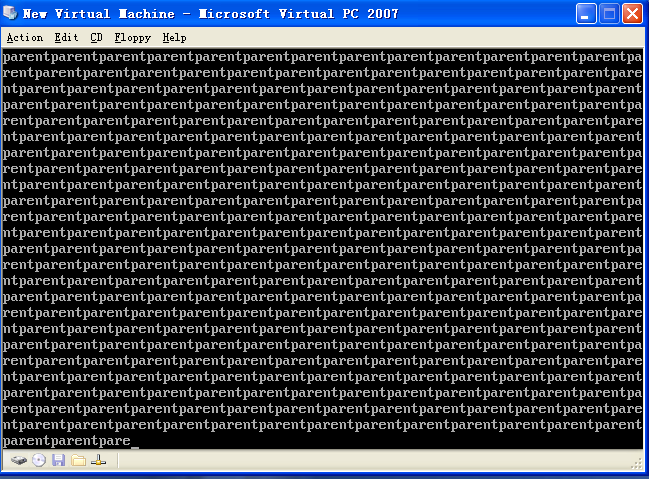
Printf(“b”);

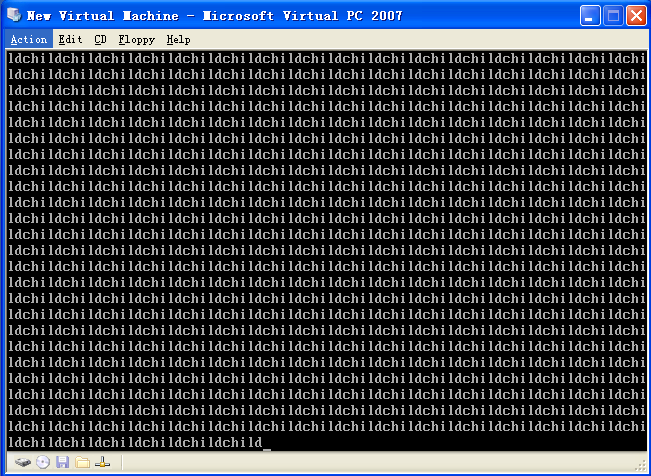
}

}

1. **实验结果**

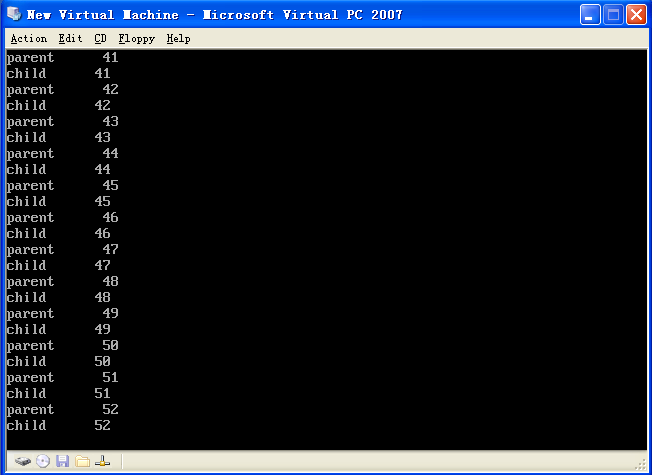
1、

****

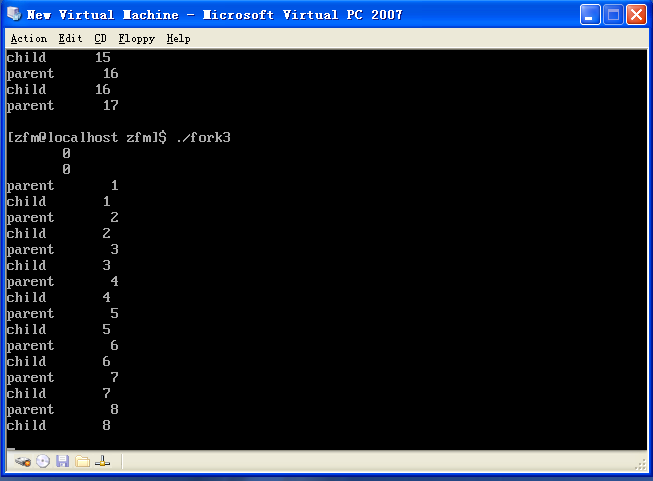
****

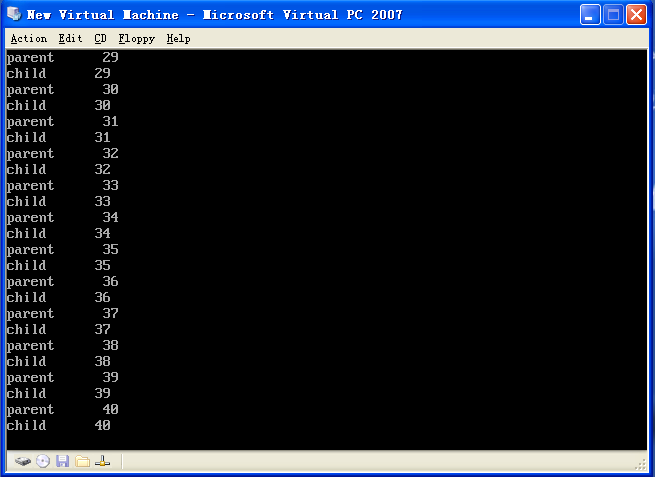
2、

****

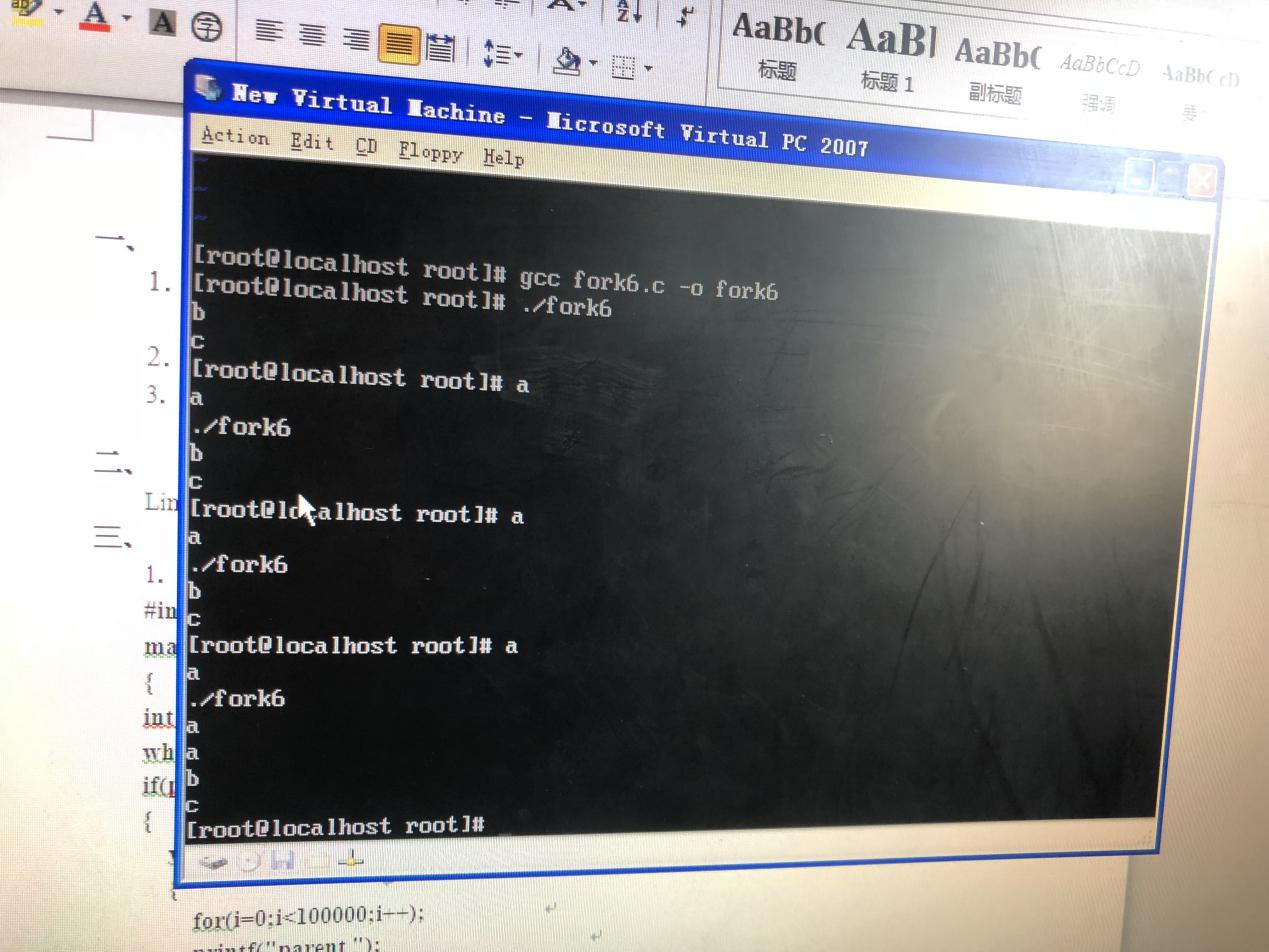
****

3、

****

****

4、

****

1. **思考分析**

通过以上程序的运行，说明一下多进程的特点？至少包括创建后特点，及执行的特点？

并得出多进程编程的结论。

多进程的特点：

可以并行的执行多个任务，提高运行效率，空间独立，数据安全，创建方便。缺点是进程的创建和销毁过程需要消耗较多的计算机资源，在需要频繁创建和删除较多进程的情况下，资源消耗过多，不适宜使用多进程完成任务。

Linux下**不管是多线程编程还是多进程编程，最终都是用do\_fork实现**的多进程编程，只是进程创建时的参数不同，从而导致有不同的共享环境。

子进程和父进程使用同一个内存空间的变量，但当子进程或父进程要改变变量x的值时就会复制该变量，从而导致父子进程里的变量值不同。**父子进程变量是互不影响的，由于父子进程地址空间是完全隔开的，变量的地址可以是完全相同的**。

附录：介绍常用的系统调用

1. fork()

功能：创建子进程，其中子进程的代码与父进程的相同。

返回值：-1 失败，创建不成功。

1. 子进程的返回值。

非0 父进程的返回值。

1. wait(0) ,父进程调用此调用后，自动阻塞，直到子进程执行完撤消时才将父进程唤醒。
2. waitpid(pid) 父进程调用些调用后，只有进程号等于pid的子进程执行完毕，才能唤醒父进程。
3. Sleep(n)，其中n为整数，为秒级。父子进程都可以调用，调用的进程调用后将自动阻塞n秒，n秒后自动唤醒，进入就绪，等待调度。
4. pipe(x), 创建管道的系统调用，其中x定义为 int x[2];

创建成功后，x[0]中保存管道读端的文件代号，x[1]保存管道写端的文件代号。

1. read(文件代号，数组名，长度)；

读文件的系统调用，从文件中读出长度个字节存放到数组中。

1. write(文件代号，数组名，长度)；

写文件的系统调用，将长度个字节的数组写入到文件中。

1. execve(“可执行的程序名”，传给程序的参数通常为0,环境参数通常为0);
2. getpid(),可以获得当前的进程的id号的系统调用。
3. getppid(),可以获得当前的进程的父进程的id号的系统调用。