



# 南昌大学实验报告

学生姓名：\_\_\_\_丁俊\_\_\_\_ 学 号：\_\_\_\_8003119100\_\_\_\_ 专业班级：\_\_\_\_信息安全 193 班\_\_\_\_  
实验类型：☐ 验证 ☐ 综合 ☐ 设计 ☐ 创新 实验日期：\_\_\_\_2021.4.11\_\_\_\_ 实验成绩：\_\_\_\_

## 一、 实验项目名称

用 C 语言创建 Linux 进程

## 二、 实验目的

熟悉用 C 语言创建 Linux 进程

## 三、 实验基本原理

fork()函数的使用和执行，返回值为 0 表示从子进程返回的 id 值；-1 表示创建失败；大于 0 则表示从父进程返回的子进程。

pthread\_create()创建进程函数和 pthread\_join()等待进程结束函数。

fork()函数创建子进程，使得父进程和子进程运行同一段代码。

## 四、 主要仪器设备及耗材

PC

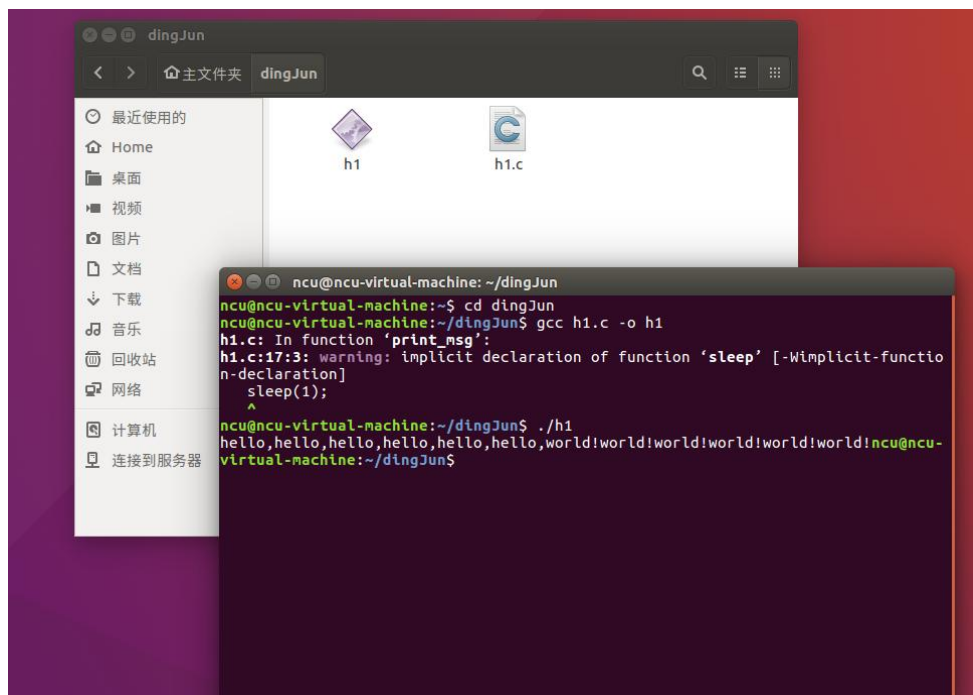
## 五、 实验步骤

用 C 语言创建进程

程序代码：

```
1 #include<stdio.h>
2 #define NUM 6
3 int main()
4 {
5     void print_msg(char*);
6     print_msg("hello,");
7     print_msg("world!");
8 }
9
10 void print_msg(char* m)
11 {
12     int i;
13     for(i=0;i<NUM;i++)
14     {
15         printf("%s",m);
16         fflush(stdout);
17         sleep(1);
18     }
19 }
```

编译并运行：



程序代码:

```
1 #include<stdio.h>
2 #include<pthread.h>
3 #define NUM 6
4 int main()
5 {
6     void print_msg(void*);
7
8     pthread_t t1,t2;
9     pthread_create(&t1,NULL,(void*)print_msg,(void *)"hello,");
10    pthread_create(&t2,NULL, (void*)print_msg,(void *)"world!\n");
11    pthread_join(t1,NULL);
12    pthread_join(t2,NULL);
13 }
14 void print_msg(void* m)
15 {
16     char *cp=(char*)m;
17     int i;
18     for(i=0;i<NUM;i++)
19     {
20         printf("%s",m);
21         fflush(stdout);
22         sleep(1);
23     }
24 }
```

运行结果:

```
ncu@ncu-virtual-machine:~/dingJun$ gcc h2.c -o thread -lpthread
h2.c: In function 'print_msg':
h2.c:20:10: warning: format '%s' expects argument of type 'char *', but arg
'void *' [-Wformat=]
    printf("%s",m);
    ^
h2.c:22:3: warning: implicit declaration of function 'sleep' [-Wimplicit-fu
n]
    sleep(1);
    ^
ncu@ncu-virtual-machine:~/dingJun$ ./thread
hello,world!
world!
hello,world!
hello,hello,world!
world!
hello,world!
hello,ncu@ncu-virtual-machine:~/dingJun$
```

可以看到“hello”和“world”交替出现。从运行结果可以看出，各线程并发执行，不是顺序执行。

程序代码:

使用 vim 命令行操作编写代码并保存

```
ncu@ncu-virtual-machine:~/dingJun$ vim a21.c
ncu@ncu-virtual-machine:~/dingJun$ vim a21.c
```

```
#include<sys/types.h>
#include<unistd.h>
#include<stdio.h>
int main()
{
    int pid;
    pid = fork();
    switch(pid)
    {
        case -1:
            printf("fail to create process\n");
            return 1;

        case 0:
            printf("I'm son, my pid is %d. my father's pid is %d\n",getpid(),getppid());
            break;

        default:
            printf("I'm father,my pid is %d,my son's pid is %d\n",getpid(),pid());
    }
    return 0;
}
~
"a21.c" 22L, 376C 1,1
```

gcc 编译并运行:

```
ncu@ncu-virtual-machine:~/dingJun$ gcc a21.c -o a21
ncu@ncu-virtual-machine:~/dingJun$ ./a21
I'm father,my pid is 2922,my son's pid is 2923
I'm son, my pid is 2923. my father's pid is 1328
ncu@ncu-virtual-machine:~/dingJun$
```

程序代码:

```
ncu@ncu-virtual-machine: ~/dingJun
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>
int main()
{
    int p1,p2;
    while((p1=fork())!=-1);
    if(p1 == 0) printf("son is working \n");

    else{
        while((p2 == fork()) == -1);
        if(p2 == 0) printf("dauthter is working \n");
        else printf("parent \n");
    }
    printf("share \n");
}
a
~
~
~
t
~
~
"a22.c" 16L, 292C
```

编译运行:

```
ncu@ncu-virtual-machine:~/dingJun$ ./a22
parent
share
son is working
share
parent
share
ncu@ncu-virtual-machine:~/dingJun$ ./a22
parent
share
son is working
share
parent
share
```

程序代码:

```
ncu@ncu-virtual-machine: ~/dingJun
#include<stdio.h>
#include<unistd.h>
#include<stdio.h>

int main()
{
    int p1,p2,i;
    while((p1 = fork()) == -1);
    if(p1 == 0) for(i = 0;i<50;i++)        printf("son  %d\n",i);

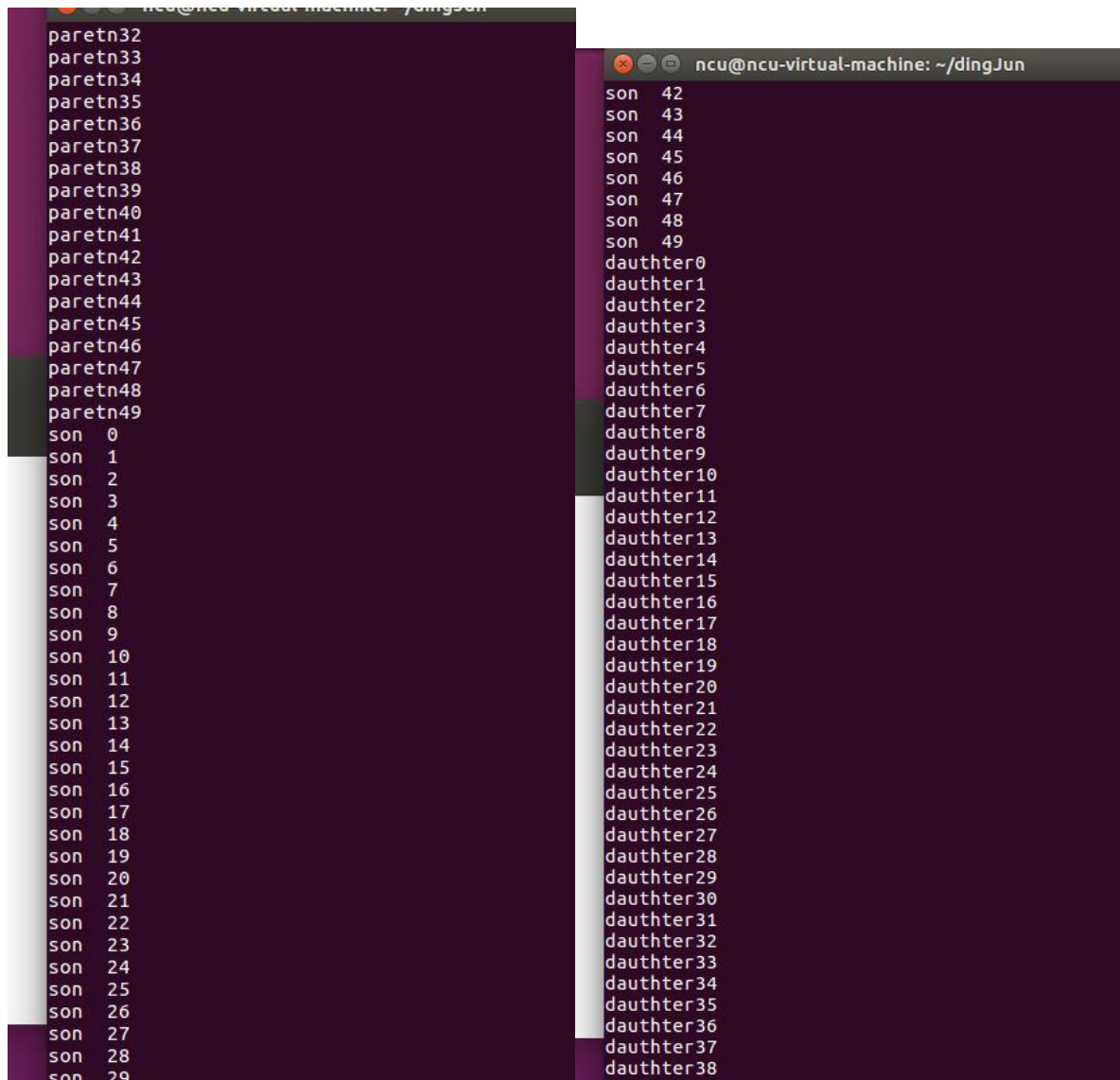
    else{
        while((p1 = fork()) == -1);
        if(p1 == 0) for(i = 0;i<50;i++)    printf("dauthter%d\n",i);
        else for(i=0;i<50;i++)            printf("paretn%d\n",i);
    }
}
~
~
~
~
~
"a33.c" 16L, 324C
```

运行结果:

可能结果不一定, 例如

```
paretn19
paretn20
paretn21
paretn22
paretn23
son  0
paretn24
paretn25
son  1
paretn26
son  2
paretn27
son  3
paretn28
son  4
paretn29
son  5
paretn30
son  6
paretn31
son  7
paretn32
son  8
paretn33
son  9
paretn34
son 10
paretn35
son 11
paretn36
son 12
paretn37
```

两个进程交替运行，但是大概率还是会每个进程的 for 循环连续运行。

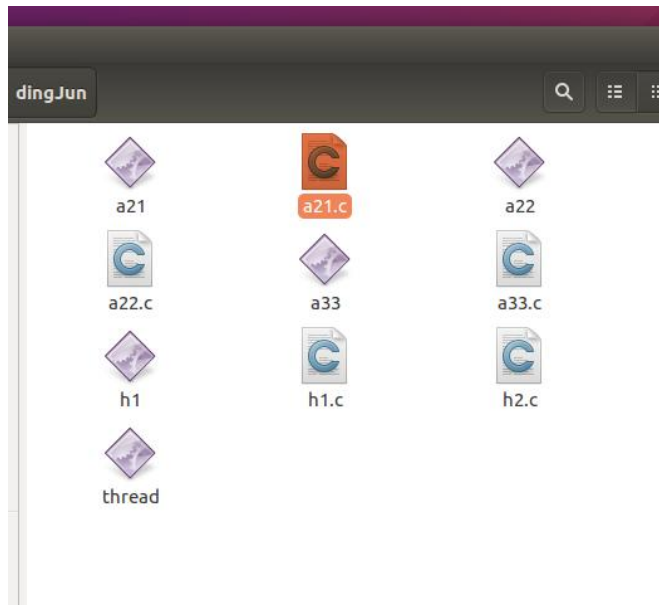


```
ncu@ncu-virtual-machine: ~/dingJun
paretn32
paretn33
paretn34
paretn35
paretn36
paretn37
paretn38
paretn39
paretn40
paretn41
paretn42
paretn43
paretn44
paretn45
paretn46
paretn47
paretn48
paretn49
son 0
son 1
son 2
son 3
son 4
son 5
son 6
son 7
son 8
son 9
son 10
son 11
son 12
son 13
son 14
son 15
son 16
son 17
son 18
son 19
son 20
son 21
son 22
son 23
son 24
son 25
son 26
son 27
son 28
son 29

ncu@ncu-virtual-machine: ~/dingJun
son 42
son 43
son 44
son 45
son 46
son 47
son 48
son 49
dauthter0
dauthter1
dauthter2
dauthter3
dauthter4
dauthter5
dauthter6
dauthter7
dauthter8
dauthter9
dauthter10
dauthter11
dauthter12
dauthter13
dauthter14
dauthter15
dauthter16
dauthter17
dauthter18
dauthter19
dauthter20
dauthter21
dauthter22
dauthter23
dauthter24
dauthter25
dauthter26
dauthter27
dauthter28
dauthter29
dauthter30
dauthter31
dauthter32
dauthter33
dauthter34
dauthter35
dauthter36
dauthter37
dauthter38
```



最后的程序代码和编译文件：



## 六、 思考讨论题或体会或对改进实验的建议

这次实验让我体会和掌握了如何用 C 语言在 linux 系统中创建子进程并且观察实验结果体会各种进程函数的功能和作用。

## 七、 参考资料