实验日志 2.10

【50%】比较 trace06 执行不同结果，编程实现 sigint_handler 捕获 INT 响应，waitfg()等待，sigchld_handler 回收僵死。

（1）比较 trace06 执行不同结果

```
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make test06
./sdriver.pl -t trace06.txt -s ./tsh -a "-p"
#
# trace06.txt - Forward SIGINT to foreground job.
#
tsh> ./myspin 4
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make rtest06
./sdriver.pl -t trace06.txt -s ./tshref -a "-p"
#
# trace06.txt - Forward SIGINT to foreground job.
#
tsh> ./myspin 4
Job [1] (3041) terminated by signal 2
```

原因：没有编写对应的 sigint_handler 函数故看不到前台的进程，与参考运行结果不一致

（2）编程实现 sigint_handler 捕获 INT 响应

```
void sigint_handler(int sig)
{
        pid_t pid=fgpid(jobs);
        int jid=pid2jid(pid);
        if(pid!=0)
        {
        printf("Job [%d] terminated by SIGINT .\n",jid);
        deletejob(jobs,pid);
        kill(-pid,sig);
        }
    return;
}
```

（3）编程实现 waitfg()等待

```
void waitfg(pid_t pid)
{
        while(pid==fgpid(jobs))
        pause();
       return;
}
```

（4）编程实现 sigchld_handler 回收僵死。

```
void sigchld_handler(int sig)
{
        pid_t pid;
        int status;
        struct job_t *job;
        while((pid=waitpid(-1,&status,WNOHANG | WUNTRACED))>0){
        if(WIFEXITED(status)){
        if(verbose)
        printf("Job [%d] (%d)terminated normally with exit status %d\n",pid2jid(pid),pid,WEXITSTATUS(status));
        deletejob(jobs,pid);
        }
        else if(WIFSTOPPED(status)){
        printf("Job [%d] (%d)stopped by signal %d\n",pid2jid(pid),pid,WSTOPSIG(status));
        job=getjobpid(jobs,pid);
        if(job!=NULL)job->state=ST;
        }
        else if(WIFSIGNALED(status)){
        printf("Job [%d] (%d)terminated by signal %d\n",pid2jid(pid),pid,WTERMSIG(status));
        deletejob(jobs,pid);
        }
      }
    return;
}
```

【10%】验证 trace06~07,了解接收信号，信号处理，信号阻塞概念。

（1） 验证 trace06

```
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make test06
./sdriver.pl -t trace06.txt -s ./tsh -a "-p"
#
# trace06.txt - Forward SIGINT to foreground job.
#
tsh> ./myspin 4
Job [1] (3170) terminated by signal 2
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make rtest06
./sdriver.pl -t trace06.txt -s ./tshref -a "-p"
#
# trace06.txt - Forward SIGINT to foreground job.
#
tsh> ./myspin 4
Job [1] (3176) terminated by signal 2
```

（2） 验证 trace07

```
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make test07
./sdriver.pl -t trace07.txt -s ./tsh -a "-p"
#
# trace07.txt - Forward SIGINT only to foreground job.
#
tsh> ./myspin 4 &
[1] (3185) ./myspin 4 &
tsh> ./myspin 5
Job [2] (3187) terminated by signal 2
tsh> jobs
[1] (3185) Running ./myspin 4 &
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make rtest07
./sdriver.pl -t trace07.txt -s ./tshref -a "-p"
#
# trace07.txt - Forward SIGINT only to foreground job.
#
tsh> ./myspin 4 &
[1] (3194) ./myspin 4 &
tsh> ./myspin 5
Job [2] (3196) terminated by signal 2
tsh> jobs
[1] (3194) Running ./myspin 4 &
```

（3） 接受信号

当内核从一个异常处理程序返回，准备将控制传递给进程 p 时，它会检查进程 p 的未 被阻塞的待处理信号的集合。如果这个集合为空（通常情况下），那么内核将控制传递到 p 的逻辑控制流中的下一条指令（Inext）。然而，如果集合是非空的，那么内核选择集合中的某个信号 k（通常是最小的 k）， 并且强制 p 接受信号 k。收到这个信号会触发进程的某种行为。一旦进程完成了这个行为，那么控制就传递回 p 的逻辑控制流中的下一条指令（Inext） 。每个信号类型都有一个预定义的默认行为，是下面中的一种：

①进程终止。②进程终止并转储存储器。③进程停止直到被 SIGCONT 信号重启。④进程忽略该信号。

（4） 信号处理

当处理程序执行 return 语句时，为信号 k 设置的处理程序被调用，一个整数参数被设置为 k。这个参数允许同一个处理函数捕获不同类型的信号。当一个程序要捕获多个信号时，会产生以下问题：

·待处理信号被阻塞。Unix 信号处理程序通常会阻塞当前处理程序正在处理的类型的待处理程序。

·待处理信号不会排队等待，任意类型至多只有一个待处理信号。

·系统调用可以被终端。像 read、wait 和 accept 这样的系统调用潜在地会阻塞进程一段较长的时间，成为慢速系统调用。

（5） 信号阻塞

如果一个 sigint 信号传递到一个正在运行上一个 sigint 信号的进程时，那么新的 sigint 信号会变成待处理，但是不会被接受，直到处理程序返回。

【30%】比较 trace08 执行不同结果，编程实现 sigtstp_handler 捕获 TSTP 响应。

（1） 比较 trace08 执行不同的结果

```
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make test08
./sdriver.pl -t trace08.txt -s ./tsh -a "-p"
#
# trace08.txt - Forward SIGTSTP only to foreground job.
#
tsh> ./myspin 4 &
[1] (3224) ./myspin 4 &
tsh> ./myspin 5
tsh> jobs
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make rtest08
./sdriver.pl -t trace08.txt -s ./tshref -a "-p"
#
# trace08.txt - Forward SIGTSTP only to foreground job.
#
tsh> ./myspin 4 &
[1] (3233) ./myspin 4 &
tsh> ./myspin 5
Job [2] (3235) stopped by signal 20
tsh> jobs
[1] (3233) Running ./myspin 4 &
[2] (3235) Stopped ./myspin 5
```

原因：没有 sigstp 函数实现打印停止信息，发送 sigstsp 信号给前台进程的功能

（2） 编程实现 sigtstp_handler 捕获 TSTP 响应。

```
void sigtstp_handler(int sig)
{
        pid_t pid=fgpid(jobs);
        int jid=pid2jid(pid);
        if(pid!=0)
        {
        printf("Job [%d] stopped by SIGTSTP .\n",jid);
        (*getjobpid(jobs,pid)).state=ST;
        kill(-pid,sig);
        }
    return;
}
```

【10%】验证 trace08

```
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make test08
./sdriver.pl -t trace08.txt -s ./tsh -a "-p"
#
# trace08.txt - Forward SIGTSTP only to foreground job.
#
tsh> ./myspin 4 &
[1] (3266) ./myspin 4 &
tsh> ./myspin 5
Job [2] (3268) stopped by signal 20
tsh> jobs
[1] (3266) Running ./myspin 4 &
[2] (3268) Stopped ./myspin 5
zhangjiwei@zhangjiwei-virtual-machine:~/shlab-handout$ make rtest08
./sdriver.pl -t trace08.txt -s ./tshref -a "-p"
#
# trace08.txt - Forward SIGTSTP only to foreground job.
#
tsh> ./myspin 4 &
[1] (3275) ./myspin 4 &
tsh> ./myspin 5
Job [2] (3277) stopped by signal 20
tsh> jobs
[1] (3275) Running ./myspin 4 &
[2] (3277) Stopped ./myspin 5
```