HW3

赵洁

1. Operating System Concept Chapter 3 Exercise: 3.1

> *The output will be 5. Because the child updates its copy of value. When control returns to the parent, its value remains at 5.*

2. Operating System Concept Chapter 3 Exercise: 3.2

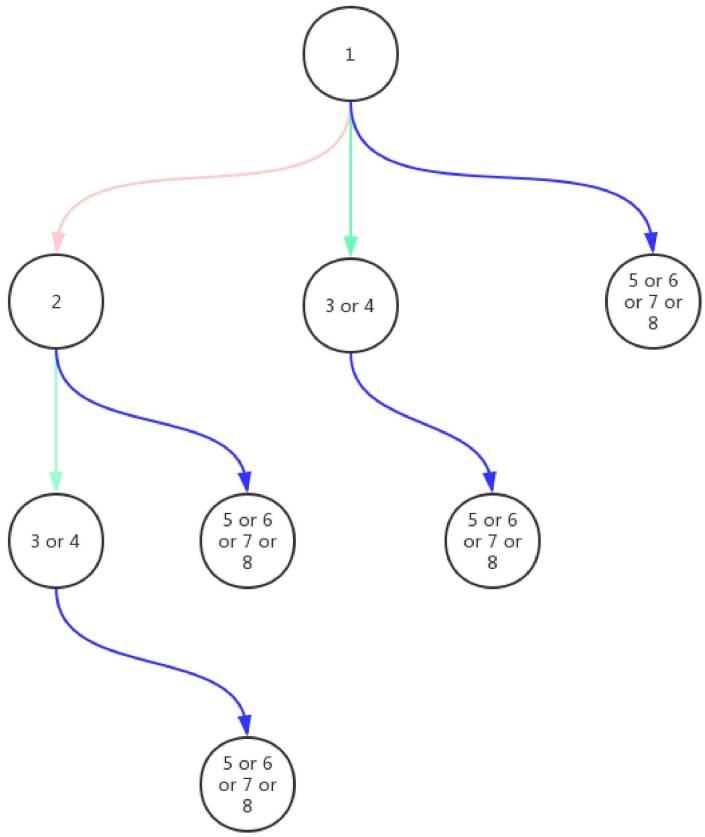
- 8 processes
- I add some debug information to the code. Use getpid everytime call fork() function.
- Result:

```
root@DESKTOP-5N08FI4:/home/h3# ./test
a:47
a:48
a:48
a:47
a:50
a:49
a:49
a:48
a:54
a:51
a:47
a:52
a:50
a:53
```

- Table:

NO.	pid	print1	print2	print3
1	47	x	x	x
2	48	x	x	x
3	49	o	x	x
4	50	o	x	x
5	51	o	o	x
6	52	o	o	x
7	53	o	o	x
8	54	o	o	x

- How to understand?



The pink curves stand for the first fork(), the green curves stand for the second fork(), and the blue curves stand for the third fork().

- During the 1st fork(), every existing process(actually 1 process) creates a process,so there will be 1+1=2 processes.
- During the 2nd fork(), every existing process(actually 2 process) creates a process, so there will be 2+2=4 processes.
- During the 3rd fork(), every existing process(actually 4 process) creates a process, so there will be 4+4=8 processes.

3. Write a kernel module.

- kernel version

```
jane@jane-VirtualBox:~/h~/h3$ cat /proc/version
Linux version 4.15.0-36-generic (buildd@lcy01-amd64-017) (gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.10))
:59:23 UTC 2018
```

- Makefile

```
obj-m :=processinfo.o
all:
  make -C /lib/modules/$(shell uname -r)/build SUBDIRS=$(PWD) modules
clean:
  make -C /lib/modules/$(shell uname -r)/build SUBDIRS=$(PWD) clean
```

- make:

```
jane@jane-VirtualBox:~/h~/h3$ make all
make -C /lib/modules/4.15.0-36-generic/build SUBDIRS=/home/jane/h~/h3 modules
make[1]: Entering directory '/usr/src/linux-headers-4.15.0-36-generic'
  CC [M]  /home/jane/h~/h3/processinfo.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/jane/h~/h3/processinfo.mod.o
  LD [M]  /home/jane/h~/h3/processinfo.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.15.0-36-generic'
```

- load the module

```
sudo insmod <module name>
```

```
  LD [M]  /home/jane/h~/h3/processinfo.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.15.0-36-generic'
jane@jane-VirtualBox:~/h~/h3$ sudo dmesg -C
jane@jane-VirtualBox:~/h~/h3$ sudo insmod processinfo.ko
```

`sudo dmesg -C `means clear the message, so we can see the result more easy.

- unload modules

```
sudo rmmod <module name>
```

```
jane@jane-VirtualBox:~/h~/h3$ sudo rmmod processinfo.ko
```

- the message during the whole process:

```
jane@jane-VirtualBox:~/h~/h3$ dmesg kern
[11423.345902]  pid:[1]         TASK_INTERRUPRIBLE      name:systemd
[11423.345903]  pid:[2]         TASK_INTERRUPRIBLE      name:kthreadd
[11423.345904]  pid:[4]         TASK_IDLE               name:kworker/0:0H
[11423.345905]  pid:[6]         TASK_IDLE               name:mm_percpu_wq
[11423.345905]  pid:[7]         TASK_INTERRUPRIBLE      name:ksoftirqd/0
[11423.345906]  pid:[8]         TASK_IDLE               name:rcu_sched
[11423.345907]  pid:[9]         TASK_IDLE               name:rcu_bh
[11423.345908]  pid:[10]        TASK_INTERRUPRIBLE      name:migration/0
[11423.345908]  pid:[11]        TASK_INTERRUPRIBLE      name:watchdog/0
[11423.345909]  pid:[12]        TASK_INTERRUPRIBLE      name:cpuhp/0
[11423.345910]  pid:[13]        TASK_INTERRUPRIBLE      name:kdevtmpfs
[11423.345911]  pid:[14]        TASK_IDLE               name:netns
[11423.345911]  pid:[15]        TASK_INTERRUPRIBLE      name:rcu_tasks_kthre
[11423.345912]  pid:[16]        TASK_INTERRUPRIBLE      name:kauditd
[11423.345913]  pid:[17]        TASK_INTERRUPRIBLE      name:khungtaskd
[11423.345914]  pid:[18]        TASK_INTERRUPRIBLE      name:oom_reaper
[11423.345914]  pid:[19]        TASK_IDLE               name:writeback
[11423.345915]  pid:[20]        TASK_INTERRUPRIBLE      name:kcompactd0
[11423.345916]  pid:[21]        TASK_INTERRUPRIBLE      name:ksmd
[11423.345917]  pid:[22]        TASK_INTERRUPRIBLE      name:khugepaged
[11423.345917]  pid:[23]        TASK_IDLE               name:crypto
```

details omitted

```
[11423.346002]  pid:[1660]      TASK_INTERRUPRIBLE      name:zeitgeist-fts
[11423.346002]  pid:[1671]      TASK_INTERRUPRIBLE      name:gvfsd-metadata
[11423.346003]  pid:[1691]      TASK_INTERRUPRIBLE      name:update-notifier
[11423.346004]  pid:[1716]      TASK_INTERRUPRIBLE      name:deja-dup-monito
[11423.346005]  pid:[1728]      TASK_INTERRUPRIBLE      name:unity-scope-hom
[11423.346005]  pid:[1740]      TASK_INTERRUPRIBLE      name:unity-scope-loa
[11423.346006]  pid:[1741]      TASK_INTERRUPRIBLE      name:unity-files-dae
[11423.346006]  pid:[1794]      TASK_INTERRUPRIBLE      name:gnome-terminal-
[11423.346007]  pid:[1801]      TASK_INTERRUPRIBLE      name:bash
[11423.346008]  pid:[15029]     TASK_IDLE               name:kworker/0:0
[11423.346009]  pid:[15097]     TASK_INTERRUPRIBLE      name:VBoxService
[11423.346010]  pid:[22977]     TASK_IDLE               name:kworker/u2:1
[11423.346010]  pid:[23013]     TASK_IDLE               name:kworker/u2:2
[11423.346011]  pid:[23826]     TASK_IDLE               name:kworker/u2:0
[11423.346012]  pid:[24745]     TASK_INTERRUPRIBLE      name:sudo
[11423.346012]  pid:[24746]     TASK_RUNNING            name:insmod
[11461.511469] Goodbye, world!
```

Appendix

```c
#include<linux/init.h>
#include<linux/module.h>
#include<linux/kernel.h>
#include<linux/printk.h>
#include<linux/sched.h>
#include<linux/sched/signal.h>

MODULE_LICENSE("GPL");
MODULE_AUTHOR("Jane");
MODULE_DESCRIPTION("A Linux module to output pid, pname, pstate.");
MODULE_VERSION("0.01");

static int __init os_lkm_example_init(void){
    struct task_struct *task;

    for_each_process(task){
    switch(task->state){
        case 0:printk(KERN_INFO "\tpid:[%d] \tTASK_RUNNING        \tname:%s\n",task->pid,task->comm);break;
        case 1:printk(KERN_INFO "\tpid:[%d] \tTASK_INTERRUPRIBLE  \tname:%s\n",task->pid,task->comm);break;
        case 2:printk(KERN_INFO "\tpid:[%d] \tTASK_UNINTERRUPTIBLE\tname:%s\n",task->pid,task->comm);break;
        case 3:printk(KERN_INFO "\tpid:[%d] \tTASK_NORMAL         \tname:%s\n",task->pid,task->comm);break;
        case 4:printk(KERN_INFO "\tpid:[%d] \t__TASK_STOPPED \tname:%s\n",task->pid,task->comm);break;
        case 8:printk(KERN_INFO "\tpid:[%d] \t__TASK_TRACED       \tname:%s\n",task->pid,task->comm);break;
        case 16:printk(KERN_INFO "\tpid:[%d] \tEXIT_DEAD          \tname:%s\n",task->pid,task->comm);break;
        case 32:printk(KERN_INFO "\tpid:[%d] \tEXIT_ZOMBIE         \tname:%s\n",task->pid,task->comm);break;
        case 48:printk(KERN_INFO "\tpid:[%d] \tEXIT_TRACE          \tname:%s\n",task->pid,task->comm);break;
        case 64:printk(KERN_INFO "\tpid:[%d] \tTASK_DEAD           \tname:%s\n",task->pid,task->comm);break;
        case 128:printk(KERN_INFO "\tpid:[%d] \tTASK_WAKEKILL     \tname:%s\n",task->pid,task->comm);break;
        case 130:printk(KERN_INFO "\tpid:[%d] \tTASK_KILLABLE     \tname:%s\n",task->pid,task->comm);break;
        case 132:printk(KERN_INFO "\tpid:[%d] \tTASK_STOPPED      \tname:%s\n",task->pid,task->comm);break;
        case 136:printk(KERN_INFO "\tpid:[%d] \tTASK_TRACED       \tname:%s\n",task->pid,task->comm);break;
        case 143:printk(KERN_INFO "\tpid:[%d] \tTASK_ALL          \tname:%s\n",task->pid,task->comm);break;
        case 256:printk(KERN_INFO "\tpid:[%d] \tTASK_WAKING       \tname:%s\n",task->pid,task->comm);break;
        case 512:printk(KERN_INFO "\tpid:[%d] \tTASK_PARKED       \tname:%s\n",task->pid,task->comm);break;
        case 1024:printk(KERN_INFO "\tpid:[%d] \tTASK_NOLOAD       \tname:%s\n",task->pid,task->comm);break;
        case 1026:printk(KERN_INFO "\tpid:[%d] \tTASK_IDLE         \tname:%s\n",task->pid,task->comm);break;
        case 2048:printk(KERN_INFO "\tpid:[%d] \tTASK_STATE_MAX    \tname:%s\n",task->pid,task->comm);break;
        default:printk(KERN_INFO "\tpid:[%d] \tstate:%ld name:%s\n",task->pid,task->state,task->comm);break;
    }
    }
    return 0;
}
static void __exit os_lkm_example_exit(void){
    printk(KERN_INFO "Goodbye, world!\n");
}

module_init(os_lkm_example_init);
module_exit(os_lkm_example_exit);
```