第三章 进程与处理机管理

第2讲：进程的创建

文艳军
计算机学院

回顾

一. 进程的描述

task_struct, TSS

二. 独学&讨论：TSS & TR

三. 进程的状态变化模型

五种基本状态，Linux 0.11的两种睡眠态

四. 独学&讨论：sys_alarm, sleep_on

目录

一. fork系统调用的功能

- fork()：创建子进程
  - 返回值：父进程：子进程的PID；子进程：0
  - 子进程从fork系统调用之后开始运行
  - 子进程会复制父进程的地址空间（代码和数据等）

```
...              /* 1.c */
main() {
    pid_t  pid;     int  stat;
    pid = fork();
    if (pid == 0) {
         printf("I am Child process\n");
         exit(0);
    }
    printf("I am Parent process\n");
    pid = wait(&stat);  /* 等待子进程结束 */
}
```

二. “进程创建”的处理过程

① 接收参数：如进程初始优先级、可执行程序及输入参数等；

② 申请PCB空间和PID，初始化PCB；

③ 设置进程地址空间相关数据结构，建立代码段、数据段、用户栈等。

④ 初始化进程现场；

⑤ 将进程置成就绪状态。

Linux 0.11中的fork系统调用

- 执行过程
  - 查找空进程
  - 申请并设置新进程控制块（包括现场）
  - 内存拷贝(设置地址空间)
  - 修改GDT
  - 返回

```
sys_fork:
    call find_empty_process
    testl %eax,%eax
    js 1f
    push %gs
    pushl %esi
    pushl %edi
    pushl %ebp
    pushl %eax
    call copy_process
    addl $20,%esp
1:  ret
```

Linux 0.11中的fork系统调用

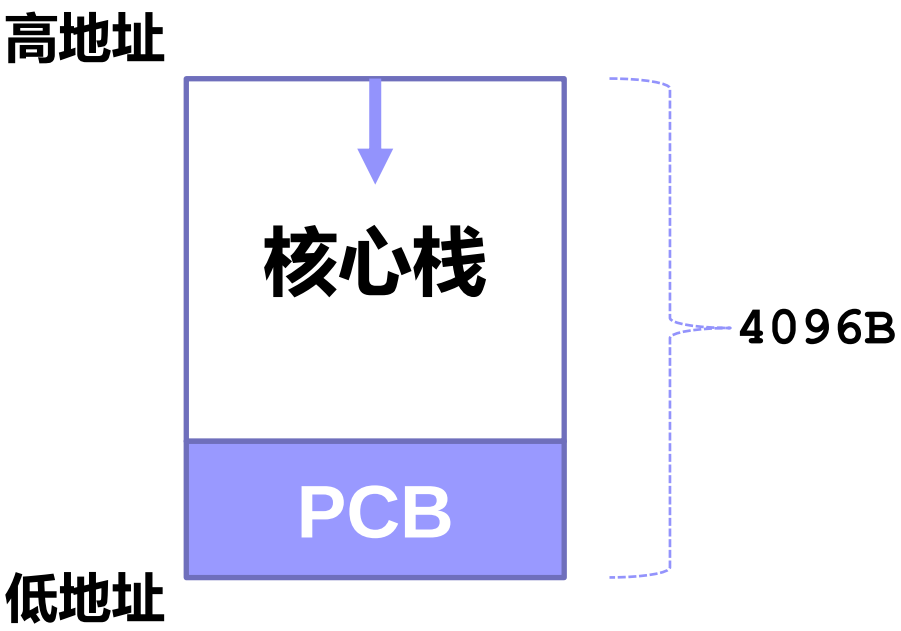
① 设置新进程控制块

```
p = (struct task_struct *) get_free_page();
if (!p)
        return -EAGAIN;
task[nr] = p;
__asm__ volatile ("cld");   /* by wyj */
*p = *current;  /* NOTE! this doesn't copy th
```

分配新进程控制块，拷贝父进程的控制块

Linux 0.11中的fork系统调用

① 设置新进程控制块



设置新进程控制块

```
p->state = TASK_UNINTERRUPTIBLE;
p->pid = last_pid;
p->father = current->pid;

p->counter = p->priority;
p->signal = 0;
p->alarm = 0;
p->leader = 0;                  /* process
p->utime = p->stime = 0;
p->cutime = p->cstime = 0;
p->start_time = jiffies;
```

修改新进程控制块：有些属性与父进程不同

设置新进程控制块

核心栈

```
p->tss.back_link = 0;
p->tss.esp0 = PAGE_SIZE + (long) p;
p->tss.ss0 = 0x10;

p->tss.eip = eip;
p->tss.eflags = eflags;
p->tss.eax = 0;
p->tss.ecx = ecx;
p->tss.edx = edx;
p->tss.ebx = ebx;
p->tss.esp = esp;
p->tss.ebp = ebp;
p->tss.esi = esi;
p->tss.edi = edi;
```

设置新进程的CPU现场

一号进程执行的第一条指令

```
p->tss.back_link = 0;
p->tss.esp0 = PAGE_SIZE + (long) p;
p->tss.ss0 = 0x10;

p->tss.eip = eip;
p->tss.eflags = eflags;
p->tss.eax = 0;
p->tss.ecx = ecx;
p->tss.edx = edx;
p->tss.ebx = ebx;
p->tss.esp = esp;
p->tss.ebp = ebp;
p->tss.esi = esi;
p->tss.edi = edi;
```

```
(gdb) n
147     if (!fork()) {

(gdb) x/3i $eip
=> 0x692b <main+288>:   mov     $0x2,%eax
   0x6930 <main+293>:   int     $0x80
   0x6932 <main+295>:   mov     %eax,0x2c(%esp)
```

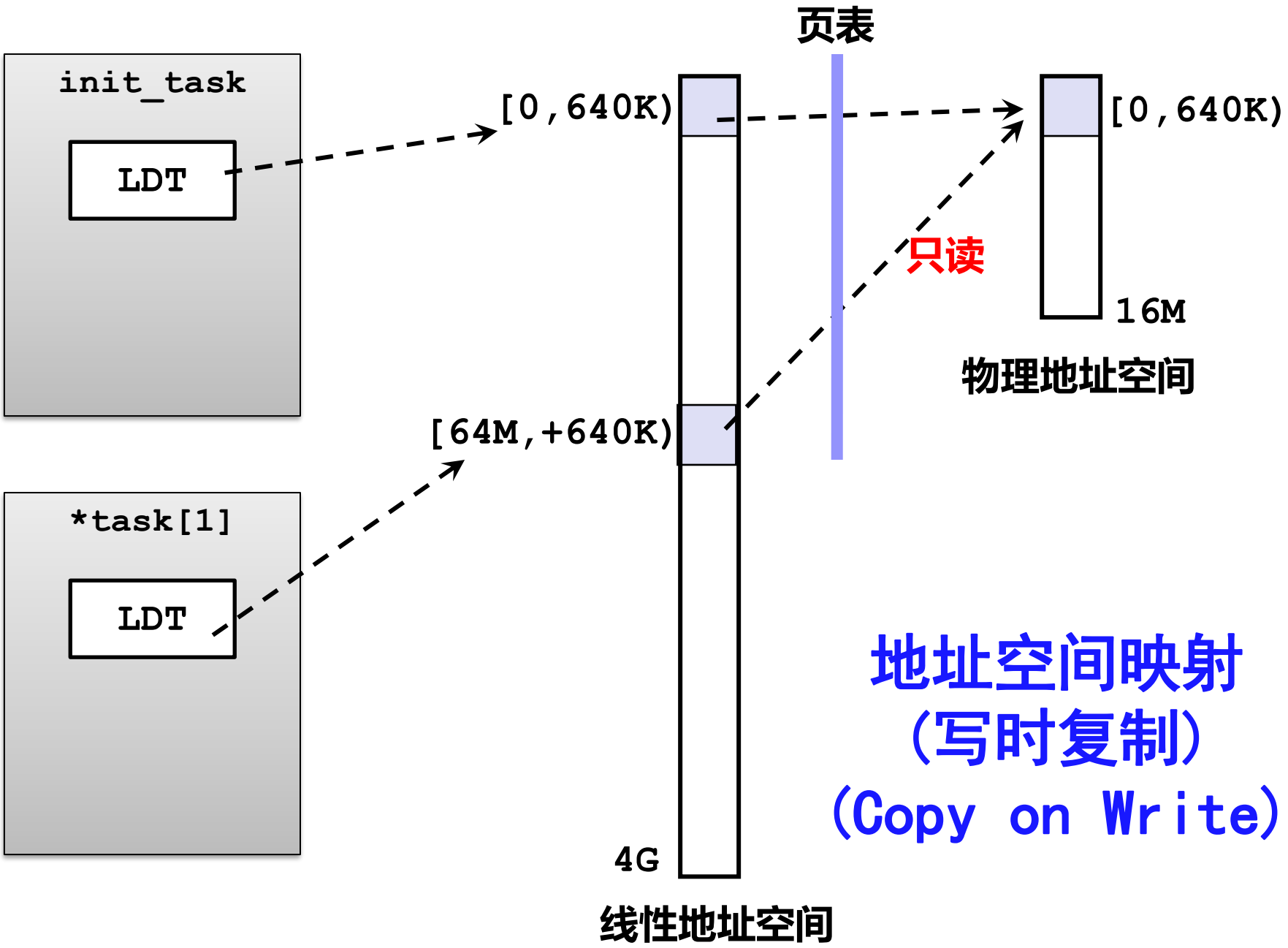
Linux 0.11中的fork系统调用

- 执行过程
  - 查找空进程
  - 申请并设置新进程控制块（包括现场）
  - 内存拷贝(设置地址空间)
  - 修改GDT
  - 返回

Linux 0.11中的fork系统调用

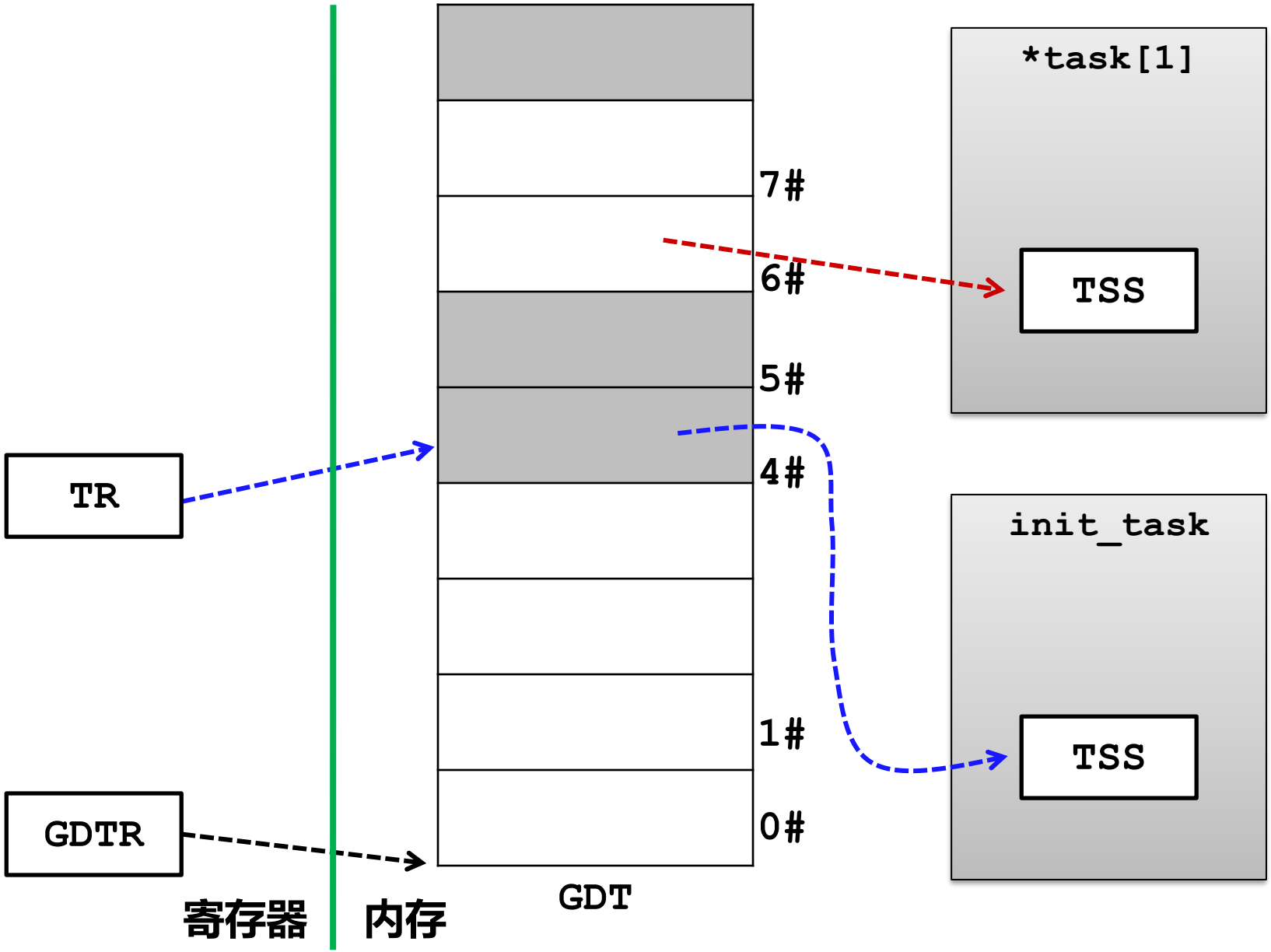
② 内存拷贝

目的：建立起新进程的地址空间。


页表
init_task
LDT
[0,640K)
[0,640K)
只读
16M
物理地址空间
[64M,+640K)
*task[1]
LDT
4G
线性地址空间
地址空间映射
(写时复制)
(Copy on Write)

Linux 0.11中的fork系统调用

- 执行过程
  - 查找空进程
  - 申请并设置新进程控制块（包括现场）
  - 内存拷贝(设置地址空间)
  - 修改GDT
  - 返回


*task[1]
TSS
7#
6#
5#
4#
1#
0#
TR
GDTR
GDT
寄存器
内存
init_task
TSS

Linux 0.11中的fork系统调用

- 演示：

#1进程的创建

- 位置：fork之前，0x6932
- 数据：task数组、GDT、current

小结

一. 进程描述

二. 进程状态

进程状态变化模型、“进程创建”处理过程

三. 进程控制与调度

四. 线程的引入