操作系统

第三章 进程与处理机管理

文艳军

计算机学院

回顾

一. 进程描述

二. 进程状态

三. 进程控制与调度

进程调度方式和时机、进程调度算法、进程切换

四. 线程的引入

目录

一. 进程切换过程

二. 演示

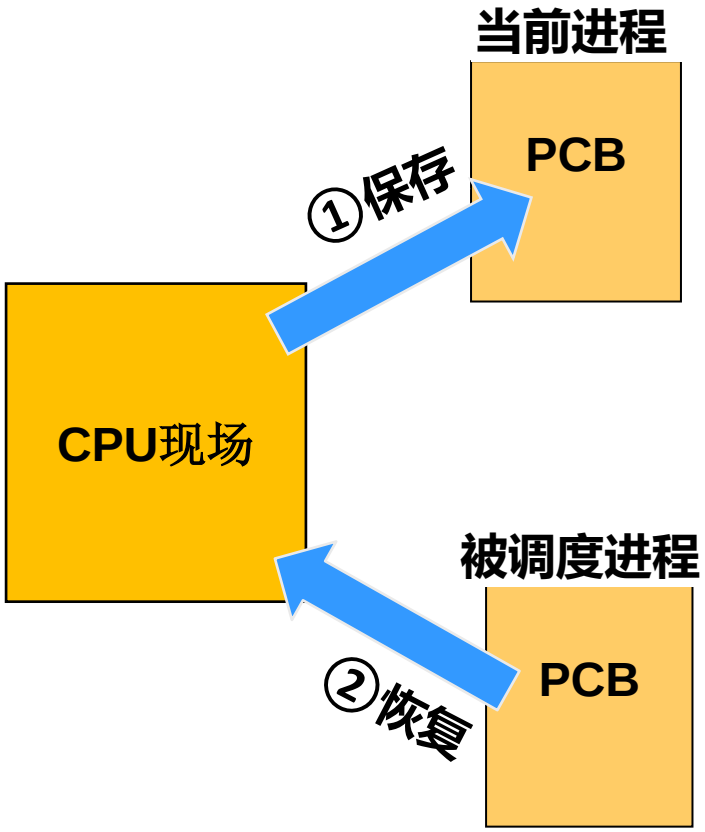
三. 独学&讨论

一. 进程切换过程

过程

① 保存当前进程的执行现场；

② 恢复被调度进程的执行现场（被调度进程开始运行，成为当前进程）。



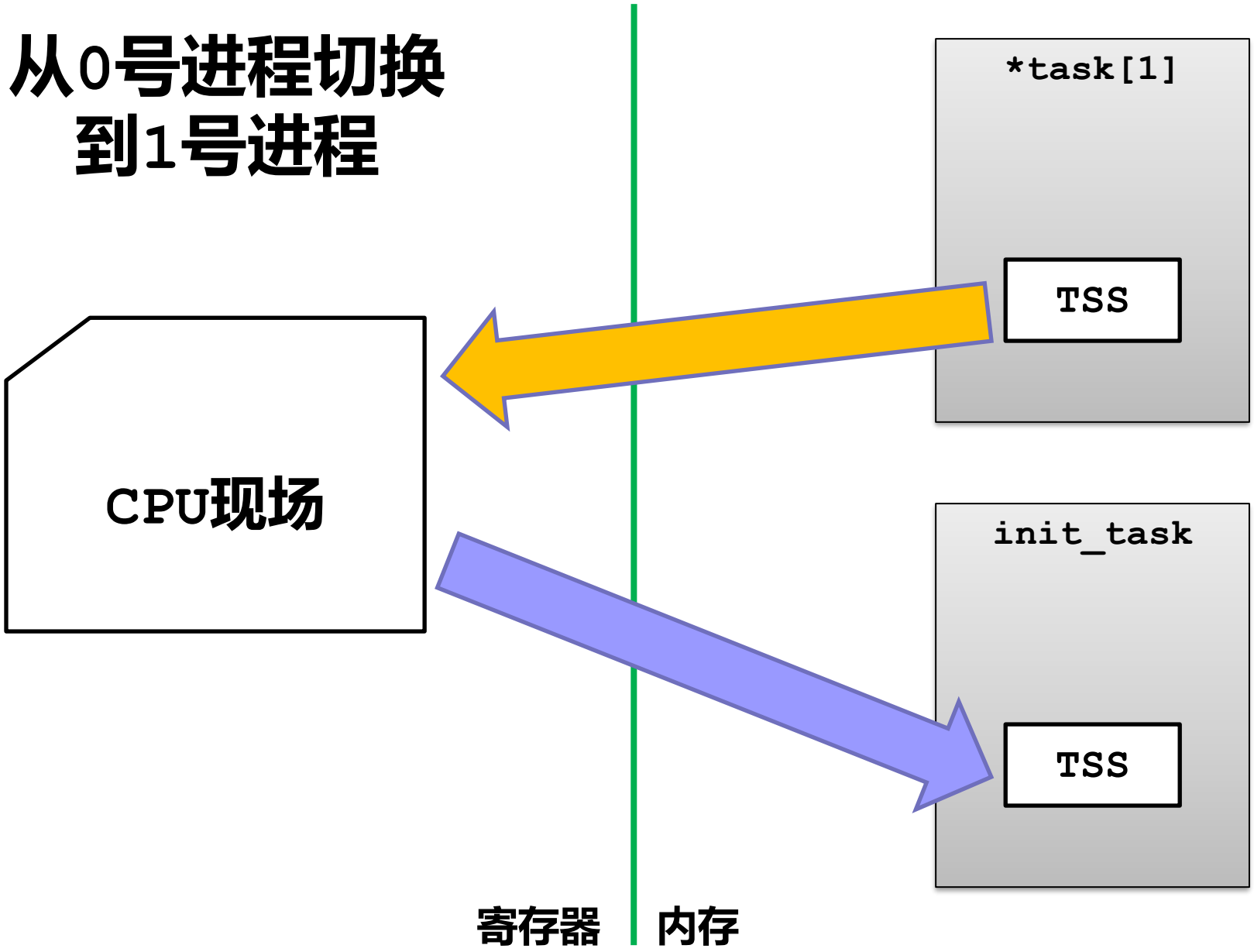
Linux 0.11的进程切换

- 第一次进程切换

```
(gdb)
131                             if (!*--p)
132                                     continue;
133                             if ((*p)->state == TASK_RUNNING && (*p)->counter > c)
134                                     c = (*p)->counter, next = i;
135                     }
136                     if (c) break;
137                     for(p = &LAST_TASK ; p > &FIRST_TASK ; --p)
138                             if (*p)
139                                     (*p)->counter = ((*p)->counter >> 1) +
140                                                     (*p)->priority;
(gdb)
141             }
142             switch_to(next);
143     }
```

next为1

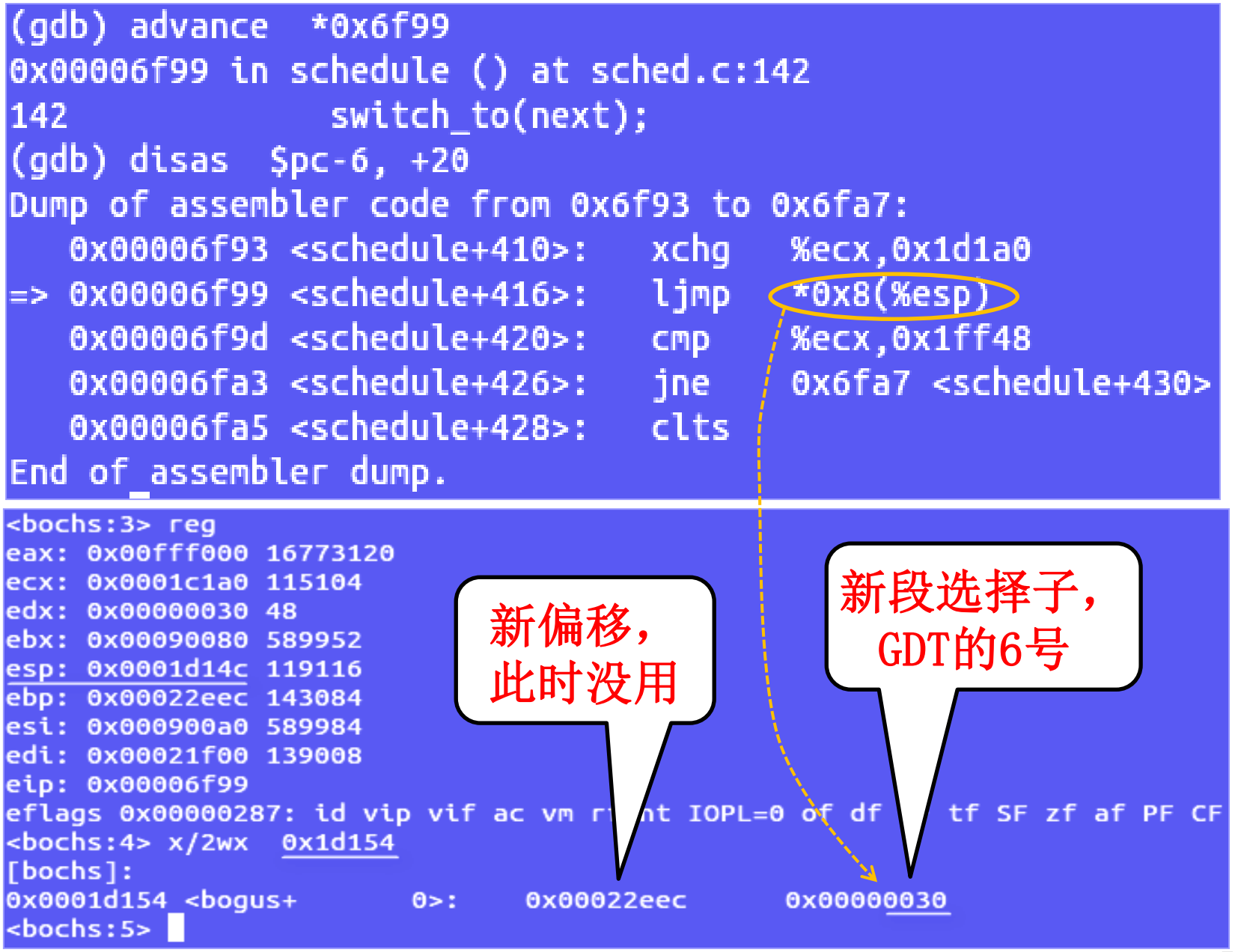
从0号进程切换到1号进程

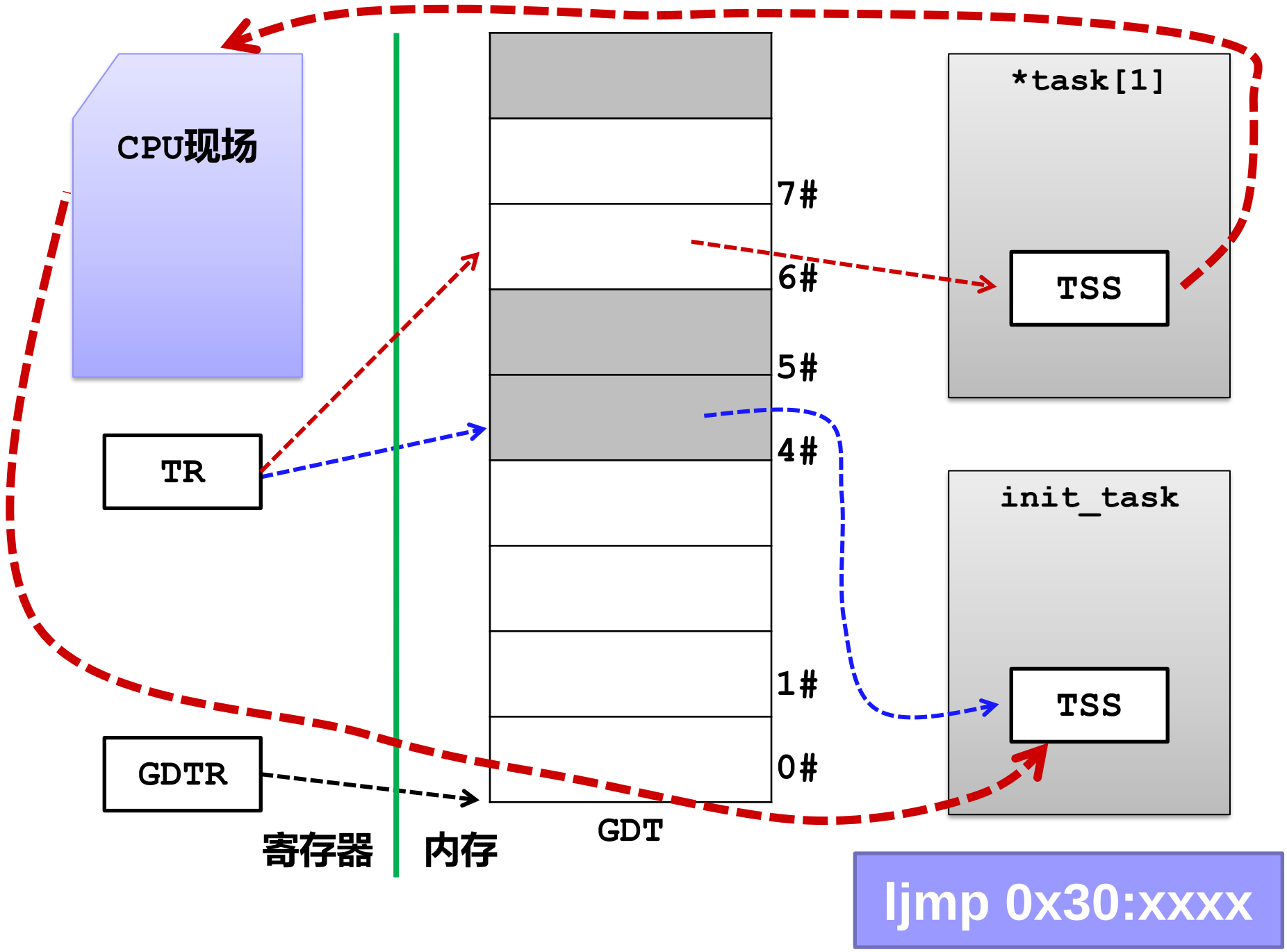


switch_to(1)的汇编码

```
(gdb) disas  $pc, +46
Dump of assembler code from 0x6f6f to 0x6f9d:
=> 0x00006f6f <schedule+374>:    mov     0x18(%esp),%eax
   0x00006f73 <schedule+378>:    shl     $0x4,%eax
   0x00006f76 <schedule+381>:    lea     0x20(%eax),%edx
   0x00006f79 <schedule+384>:    mov     0x18(%esp),%eax
   0x00006f7d <schedule+388>:    mov     0x1d1c0(,%eax,4),%eax
   0x00006f84 <schedule+395>:    mov     %eax,%ecx
   0x00006f86 <schedule+397>:    cmp     %ecx,0x1d1a0
   0x00006f8c <schedule+403>:    je      0x6fa7 <schedule+430>
   0x00006f8e <schedule+405>:    mov     %dx,0xc(%esp)
   0x00006f93 <schedule+410>:    xchg    %ecx,0x1d1a0
   0x00006f99 <schedule+416>:    ljmp    *0x8(%esp)
End of assembler dump.
(gdb)
```

间接长跳转指令


(gdb) advance *0x6f99
0x00006f99 in schedule () at sched.c:142
142 switch_to(next);
(gdb) disas $pc-6, +20
Dump of assembler code from 0x6f93 to 0x6fa7:
0x00006f93 <schedule+410>: xchg %ecx,0x1d1a0
=> 0x00006f99 <schedule+416>: ljmp *0x8(%esp)
0x00006f9d <schedule+420>: cmp %ecx,0x1ff48
0x00006fa3 <schedule+426>: jne 0x6fa7 <schedule+430>
0x00006fa5 <schedule+428>: clts
End of assembler dump.
<bochs:3> reg
eax: 0x00fff000 16773120
ecx: 0x0001c1a0 115104
edx: 0x00000030 48
ebx: 0x00090080 589952
esp: 0x0001d14c 119116
ebp: 0x00022eec 143084
esi: 0x000900a0 589984
edi: 0x00021f00 139008
eip: 0x00006f99
eflags 0x00000287: id vip vif ac vm rf nt IOPL=0 of df tf SF zf af PF CF
<bochs:4> x/2wx 0x1d154
[bochs]:
0x0001d154 <bogus+ 0>: 0x00022eec 0x00000030
<bochs:5>
新偏移，
此时没用
新段选择子，
GDT的6号


CPU现场
TR
GDTR
寄存器
内存
7#
6#
5#
4#
1#
0#
GDT
*task[1]
TSS
init_task
TSS
ljmp 0x30:xxxx

目录

一. 进程切换过程

二. 演示

三. 独学&讨论

二. 演示

- 主题：Linux 0.11的进程切换

第1次进程切换

  - 位置：schedule，sched.c:136
    switch_to

三. 独学&讨论

- 学习《Linux内核完全剖析》：
  - §4.7.4：任务切换
  - 源码：schedule()，switch_to()
- 问题：
  - 切换前后GDTR、LDTR的值会变吗？

小测试

1. 进程的TSS什么时候更新？
2. 从1号进程切换回0号进程时，从哪里开始执行？
3. 执行前述ljmp指令时，在内存与CPU之间传递了多少字节？

小结

一. 进程描述

二. 进程状态

三. 进程控制与调度

进程调度方式和时机、进程调度算法、进程切换

四. 线程的引入