操作系统Lab4 实验报告

2012011350 计24 周琳钧

练习1：分配并初始化一个进程控制块

1. 实验流程：实验代码在alloc\_proc函数中，本练习只需要进行初始化即可，并不需要对内部元素赋值，因此，凡是struct\*类型的都赋为NULL，uint32/int类型的都赋为0或-1

struct类型的都用memset赋为0，注意cr3的赋值为boot\_cr3即可

1. 问答题1：context保存的是该进程的8个寄存器值，这样当访问其他进程时（称作“上下文”），能保证再次回到此进程时寄存器值不会更改，tf是中断帧指针，记录了中断之前栈的状态，这两个变量共同保证了进程间切换时进程数据不会发生丢失。

练习2：为新创建的内核线程分配资源

1. 实验流程：第一步，分配进程，执行alloc\_proc函数初始化一个新进程，并将其parent赋为current，第二步，为进程分配栈，第三步，判定current->mm为NULL，否则为用户进程而非内核进程，以上三步出错时相应执行错误处理函数，第四步，设置中断帧，第五步，将该进程插入hash\_list和proc\_list，这里注意要禁止中断处理程序，否则会使进程链表异常，并获取一个新的pid，并把当前进程数加1，第六步，激活该进程，第七步，将返回值赋为该进程的pid号，结束该函数。这样完成了整个proc表的构建。
2. 问答题1：一定会分给一个唯一标识，我们考察函数get\_pid()，当0~4095某个数未匹配进程时，last\_pid会被唯一赋成该值，而如果全部装满，则会循环执行repeat直到某个进程被kill掉，这时last\_pid又会被唯一赋成该值，因此get\_pid()总能保证进程被分配为一个唯一标识。

练习3：理解进程切换

1. proc\_run函数理解：proc\_run函数完成了进程切换，具体做法是判断当前进程是否等于要更改后的进程，若不等，则可以完成切换，具体做法为：第一步，禁止中断，第二步，将当前进程替换为新进程，第三步，修改ESP0使之指向新进程，第四步，切换cr3寄存器数据使之指向新进程的页表，第五步，保存当前进程现场，将寄存器组数据切换为新进程的数据（通过访问context完成），第六步，恢复中断。
2. 问答题1：两个，一个是idleproc（第0个进程），另一个是initproc（第1个进程）
3. 问答题2：禁止中断和恢复中断。