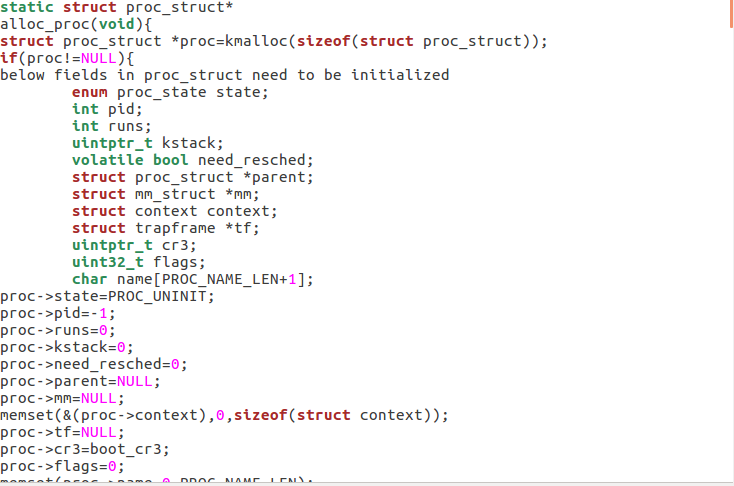
**（一）分配并初始化一个进程控制块**

alloc\_proc函数（位于kern/process/proc.c中）负责分配并返回一个新的struct proc\_struct结构，用于存储新建立的内核线程

的管理信息。ucore需要对这个结构进行最基本的初始化，你需要完成这个初始化过程。

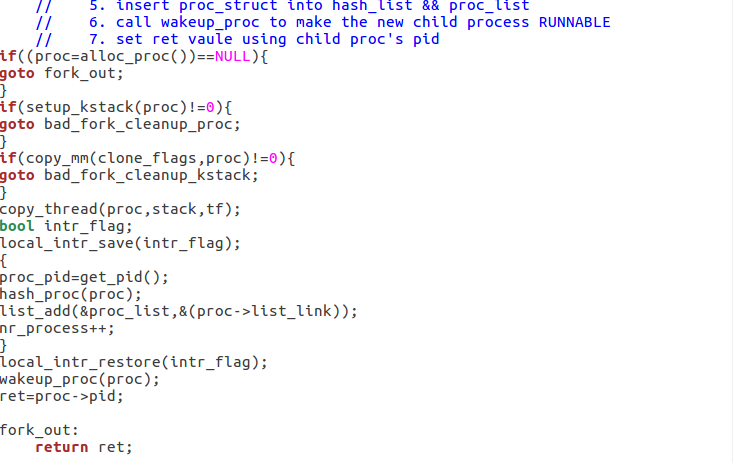
实验代码：



**请说明proc\_struct中 struct context context 和 struct trapframe \*tf 成员变量含义和在本实验中的作用是啥？**

①context：进程的上下文，用于进程切换。起到的作用就是保存了现场。在 ucore中，所有的进程在内核中也是相对独立的，因此context 保存寄存器的目的就在于在内核态中能够进行上下文之间的切换。实际利用context进行上下文切换的函数是在kern/process/switch.S中定义switch\_to。

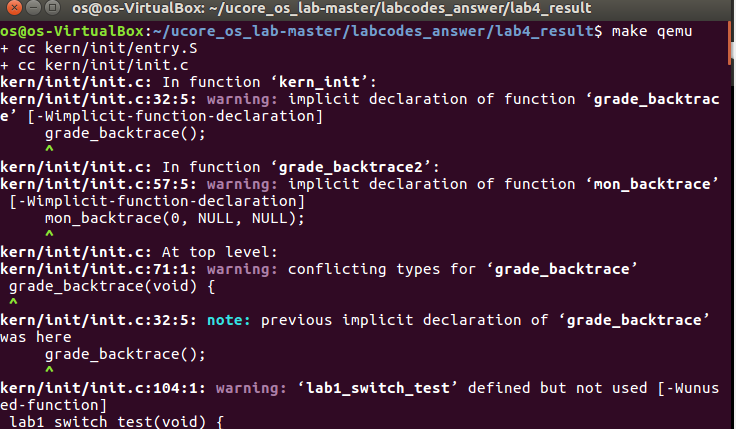
② tf：中断帧的指针，总是指向内核栈的某个位置：当进程从用户空间跳到内核空间时，中断帧记录了进程在被中断前的状态。当内核需要跳回用户空间时，需要调整中断帧以恢复让进程继续执行的各寄存器值。除此之外，ucore内核允许嵌套中断。因此为了保证嵌套中断发生时tf 总是能够指向当前的tf，ucore 在内核栈上维护了 tf 的链。

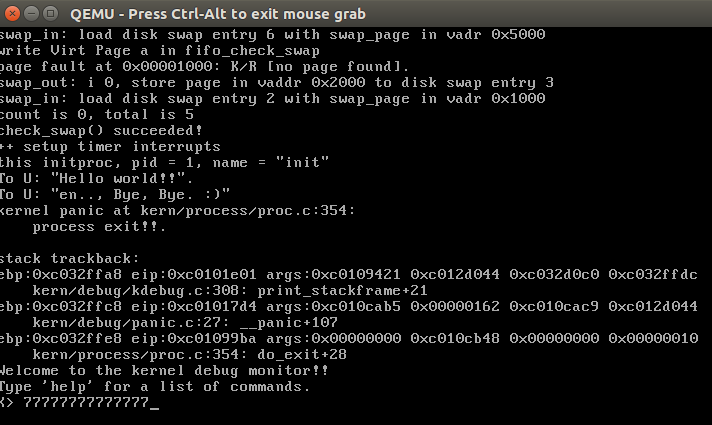
**（二）为新创建的内核线程分配资源**

**请说明ucore是否做到给每个新fork的线程一个唯一的id？请说明你的分析和理由**

首先，本实验不提供线程释放的功能，意味着pid只分配不回收。当fork的线程总数小于MAX\_PID时，每个线程的pid是唯一的。当fork的线程总数大于MAX\_PID时，后面fork的线程的pid可能与前面的线程重复（暂不确定）。

**（三）阅读代码，理解 proc\_run 函数和它调用的函数如何完成进程切换的。**

在指定文件下输入 make qemu 得到以下结果



**在本实验的执行过程中，创建且运行了几个内核线程？**

本实验创建且运行了两个内核线程，分别是idle和init线程。

语句 local\_intr\_save(intr\_flag);....local\_intr\_restore(intr\_flag); 在这里有何作用?请说明理由

避免在进程切换过程中处理中断。