# 每个进程一个内核页表

1. 结构体proc添加kernelPageTable属性
2. 依照kvminit改造，定义kvmmake方法，可以生成一个新kernel page table
3. 之前内核栈的定义放在procinit中，放在全局kernelPageTable中，现在定义在每个进程的kernelPageTable中，在allocproc中利用uvmmake创建一个kernel page table，赋值给p->kernelPageTable，再为这个页表分配一个kernel stack
4. 在scheduler中发生进程切换，切换前将satp更新为进程的kernel\_pagetable，使该进程期间内核页表用的是此进程自己的页表，返回到scheduler中后，再把satp更新为全局的kernel\_pagetable
5. kvmpa中kernel\_pagetable改为调用myproc()->kernelPageTable
6. 进程结束后，回收进程资源。取消页表与物理页的映射，回收页表

# 简化 copyin/copyinstr

1. 内核更改进程用户映射时，相应地更改内核页表的映射，proc中的fork()、exec中的exec()、sysproc中的sys\_sbrk()
2. 在proc的userinit中，更改用户页表映射的同时，也更改进程内核页表
3. 修改pte中的PTE\_U标志位为0，让内核可以访问，用户不可以访问
4. 设置PLIC限制，让用户空间虚拟地址不超过这个值

# 问题

1. kernel/proc.c:scheduler()方法中进程切换前，修改satp中的值为此进程kernelPageTable，修改kernel/vm.c:kvmpa()中walk方法第一个参数为myproc()->kernelPageTable，进程切换回来后，再设置satp为全局内核页表