1. 分页

1140310606 张茗帅

**基础知识：**

1.段部件： 段地址加上偏移地址输出线性地址

2.段部件计算的线性地址，高20位\*4得到映射表内偏移，表中双字指定一个物理页，低12位作为表内偏移。

3. 页表大小4kb，每个页表项4个字节，表示页的物理地址，一个页表有1024个页表项。

页目录表大小4Kb，存每个页表的物理地址，能存1024个页表的索引。

CR3存当前任务页目录表物理地址。

一个线性地址，高10位乘以4加上页目录表物理地址，从该地址得到页表地址，中间10位\*4加上页表地址，从该地址得到页地址。低12位加上页地址得到对应的物理地址。

1. 页目录表和页表的物理地址后12位被赋予了特殊的功能，CR3也一样。另外CR0的最高位控制着页功能的开闭，1为开启，只有保护模式才能开启，关闭时线性地址就是物理地址。

**①设置分页，准备并打开分页机制**

选定页目录的物理地址为0x00020000，利用循环将页目录表项清零，共1024项。页目录表设置自己的物理地址对应的目录项，P位和RW位均置1将0x00020003存入页目录表的最后一项，0xffc项



创建一个页表，添加到页目录表的第0x0项处，物理地址为0x00021000，P位和RW位均置1。



将存有内核的低端1MB的空间所包含的页的物理地址按顺序写到页表中，256项，同样P位和RW位均置1。256项之后全部清0。

添加第一项之后





打开分页机制，将页目录表物理地址存入CR3，将CR0最高位置1.



**②将内核映射到线性地址0x80000000以上1M地址空间**，便于用户程序将他作为全局空间进行访问。需要映射的有页目录自己的线性地址、GDT中的各种描述符、正在使用的CS、DS、SS等段寄存器。

映射页表的线性地址。因为已经开启了分页机制，那么线性地址就要通过页部件进行转化了。页表线性地址0xfffff200，根据高10位乘以4得到0xffc加上CR3中的页目录表物理基地址得到0x00020ffc，从中得到页表物理基地址为0x00020000.根据中间10位乘以4得到0xffc加上页表物理基地址0x00020000得到0x00020ffc，从中得到页物理基地址为0x00020000.最后12位乘以4得到0x800，加上页物理基地址，得到最终的物理地址为0x00020800。



GDT表中的描述符将最高位置1，即完成了映射。

用转移指令完成刷新代码段基地址，启用高端线性地址



刷新后，代码线性地址为



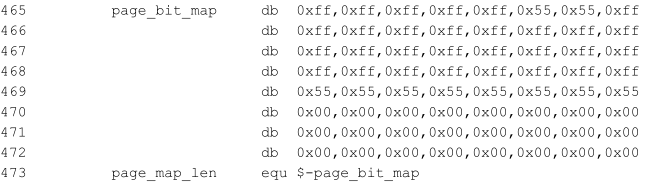
Ss、ds段寄存器的刷新通过重新加载完成。

**③内核虚拟内存分配**

创建TSS需要进行虚拟内存分配，内核初始化了一个双字的可分配线性地址，分配到新的内从空间后，会修正到下一个可分配线性地址处。



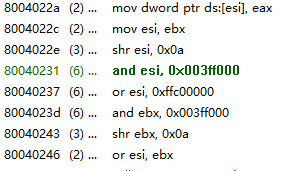
通过页部件的转化，我们可以得到可分配线性地址相对应的页目录项的线性地址，检查P位，若为1，页表存在，再继续计算对应的页的物理地址即可；若0，不存在，需要创建页表，并添加页目录项，调用allocate\_a\_4k\_page，分配空闲自然页。

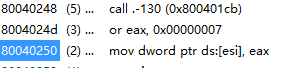


空闲页查找需要使用页面位映射串，即一个认为规定的串，每1个比特表示一个自然页的分配情况。0即空闲。程序中使用bts进行了空闲页的查找，找到空闲页，将对应的比特置1。找到的空闲页在比特串中的位置数乘以0x1000得到空闲页的物理地址，存到eax。

找到的空闲页物理地址需要登记在对应页表项中。为此，程序将低12位作为页表内偏移量，中间10位作为页目录内偏移量，用高10位访问页目录，找到对应的页表项，存入物理地址。

程序给的可分配地址空间对应的页表存在，下图进行空闲页查找和登记





得到页物理地址，低12位置0x007



对应的页表项



登记到页表项中



回到内核，可分配地址空间修正



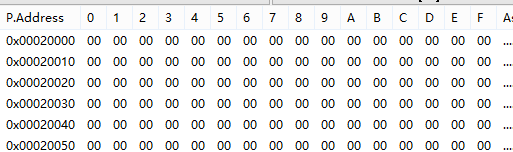
之后进行TSS创建。

**④用户任务虚拟地址空间分配**

在内核中创建TCB并添加到TCB链，分配内存时同上。这里的TCB比较简单，但是其中0x06项处是下一个可用的线性地址。

之后进行用户程序的加载，每一个用户程序有自己的一个分页系统，内核能够进行用户页目录表和页表的创建，但是问题是，内核创建过程中的操作建立在内核的分页机制下，依靠当前CR3进行线性地址映射，最终的解决办法是进行内核页目录表和页表与用户页目录表和页表的复制，毕竟我们进行了高端线性地址和低端线性地址的区分，用来对应全局空间和局部空间。

加载程序中，先清空内核页表前半部分，为用户任务预留空间。清空后的前半部分：



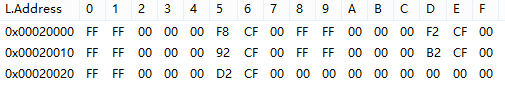
根据计算得到的用户扇区数0x1f88e，我们分配0x20个页给它，不是整32个页，凑整，也就是说有的页没有填满。



然后用两层循环进行加载，页的分配方式与之前相同。

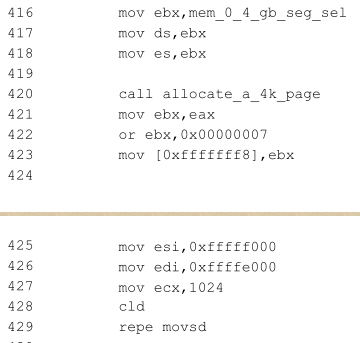
外循环负责页的分配，内循环负责扇区读取，每次读取8个扇区共4KB。

加载结束后为用户各个段创建描述符，用户程序是在平坦模式下的，段基地址都是0x00000000，存到LDT中。全部描述符添加结束后：



**⑤重定位和页目录表复制**

重定位与之前没什么不同。页目录表的复制调用了create\_copy\_cur\_pdir过程。



过程使用4GB数据段创建页目录表，并将它的物理地址添加在内核页目录表的倒数第二项，得到的页目录表物理地址0x00141000



存到内核页目录表中，倒数第一项是内核页目录表物理地址0x00020000



有了两个页目录表物理地址，由上图程序，进行复制即可，最后将用户程序页目录表物理地址存储到用户程序TSS中。



至此，核心内容基本完结。