PA 3-3 实验报告

221220085 时昌军

一、实验目的

1. 了解保护模式下的分页机制。
2. 掌握线性地址向物理地址转换的方法。

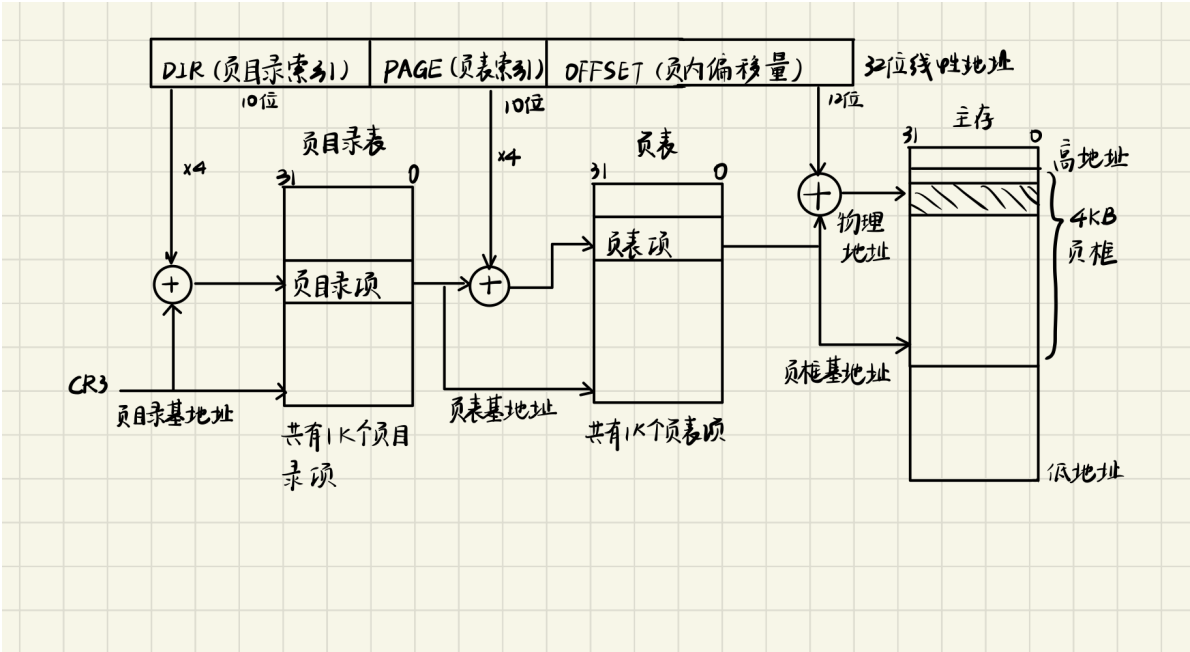
二、实验过程

1. 修改Kernel和testcase中`Makefile`的链接选项；
2. 在`include/config.h`头文件中定义宏`IA32_PAGE`并`make clean`；
3. 在`CPU_STATE`中添加`CR3`寄存器；
4. 修改`laddr_read()`和`laddr_write()`，适时调用`page_translate()`函数进行地址翻译；
5. 修改Kernel的`loader()`，使用`mm_malloc`来完成对用户进程空间的分配；
6. 通过`make test_pa-3-3`执行并通过各测试用例。

三、思考题

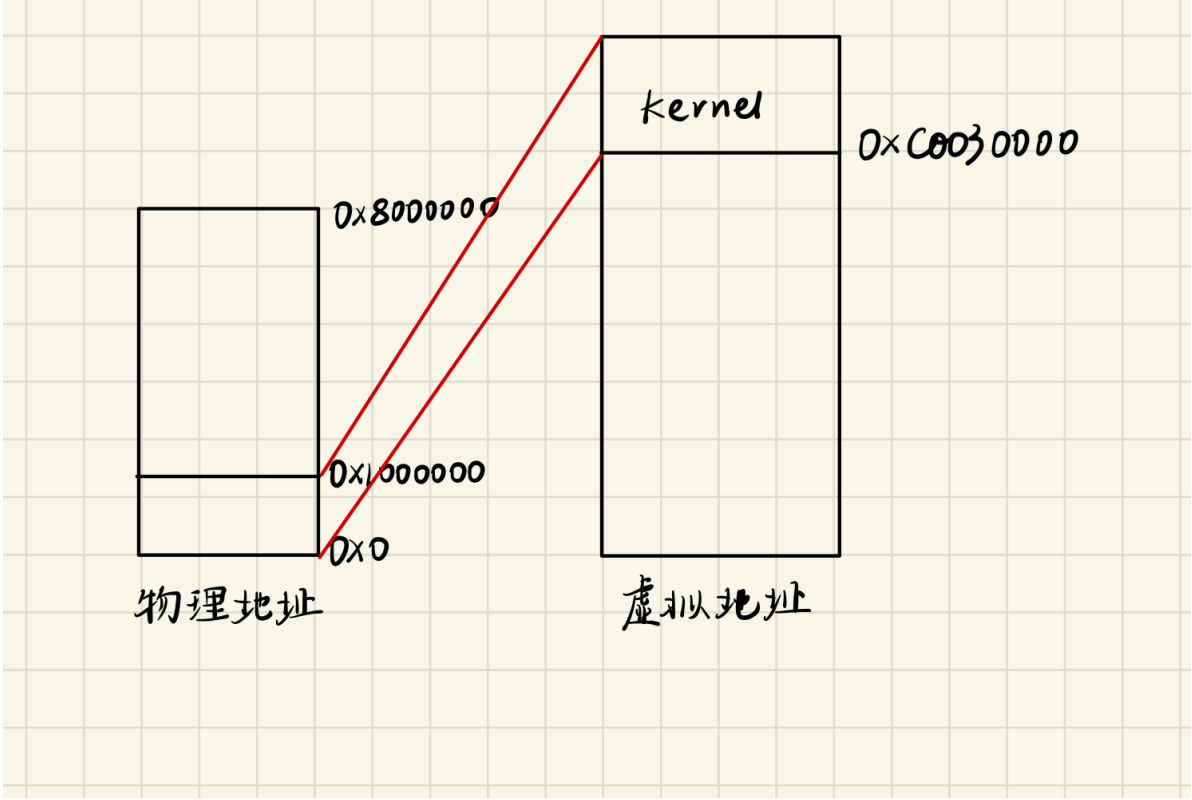
1. **Kernel的虚拟页和物理页的映射关系是什么？请画图说明；**

答：每个进程都有自己独立的页表来描述该进程的虚拟地址空间和物理地址空间之间的映射关系。从较为直观的角度来说，一个32位的线性地址可以分为两个部分：高20位指出该线性地址属于哪一个虚拟页，而低12位则指出该地址具体指向的字节在该虚拟页内的偏移量是多少。若参照段表的组织方式，将所有2^{20}个页表项都按顺序存放在一个数组中。那么地址转换的过程就可以简单地表达为：使用线性地址中的高20位作为索引，查找页表；提取对应页表项中的物理页号，加上线性地址中低12指出的页内偏移量；便能够获取对应的物理地址了。



2. **以某一个测试用例为例，画图说明用户进程的虚拟页和物理页间映射关系又是怎样的？Kernel映射为哪一段？你可以在`loader()`中通过`Log()`输出`mm_malloc`的结果来查看映射关系，并结合`init_mm()`中的代码绘出内核映射关系。**

答:




```
./nemu/nemu --autorun --testcase mov-c --kernel
NEMU load and execute img: ./kernel/kernel.img  elf: ./testcase/bin/mov-c
nemu trap output: [src/main.c,82,init_cond] {kernel} Hello, NEMU world!
nemu trap output: [src/elf/elf.c,29,loader] {kernel} ELF loading from ram disk.
nemu trap output: [src/elf/elf.c,54,loader] {kernel} paddr:1000000,vaddr:0
nemu trap output: [src/elf/elf.c,54,loader] {kernel} paddr:1001000,vaddr:8049000
nemu trap output: [src/elf/elf.c,54,loader] {kernel} paddr:1002000,vaddr:804a000
nemu trap output: [src/elf/elf.c,54,loader] {kernel} paddr:1003000,vaddr:804c000
nemu: HIT GOOD TRAP at eip = 0x080490c2
```

第一个可分配给用户进程的物理页首地址是 0x1000000。
0x0 映射到物理地址0x1000000。
0x8049000 映射到物理地址0x1001000。
0x804a000 映射到物理地址0x1002000。
0x804c000 映射到物理地址0x1003000。

内核映射关系:kernel 从虚拟地址 0xc0030000 开始，映射到物理地址 0x30000开始的地方。

3. **“在Kernel完成页表初始化前，程序无法访问全局变量”这一表述是否正确？在`init_page()`里面我们对全局变量进行了怎样的处理?**

答：正确。访问全局变量只知道虚拟地址，需要进行地址转换后才能找到全局变量的物理地址。而 Kernel 页表初始化前，地址转换过程不能正常进行，无法进行虚拟页到物理页的映射，无法访问全局变量。 在`init_page()`中，对全局变量的访问通过调用`va_to_pa（）`函数来实现。该函数将虚拟地址减去`KOFFSET`，即得到物理地址。