PA3-3 实验报告

1.Kernel的虚拟页和物理页的映射关系是什么？请画图说明

虚拟页到物理页采用全相联映射. 虚拟地址通过查询页表来转换为相应的物理地址.

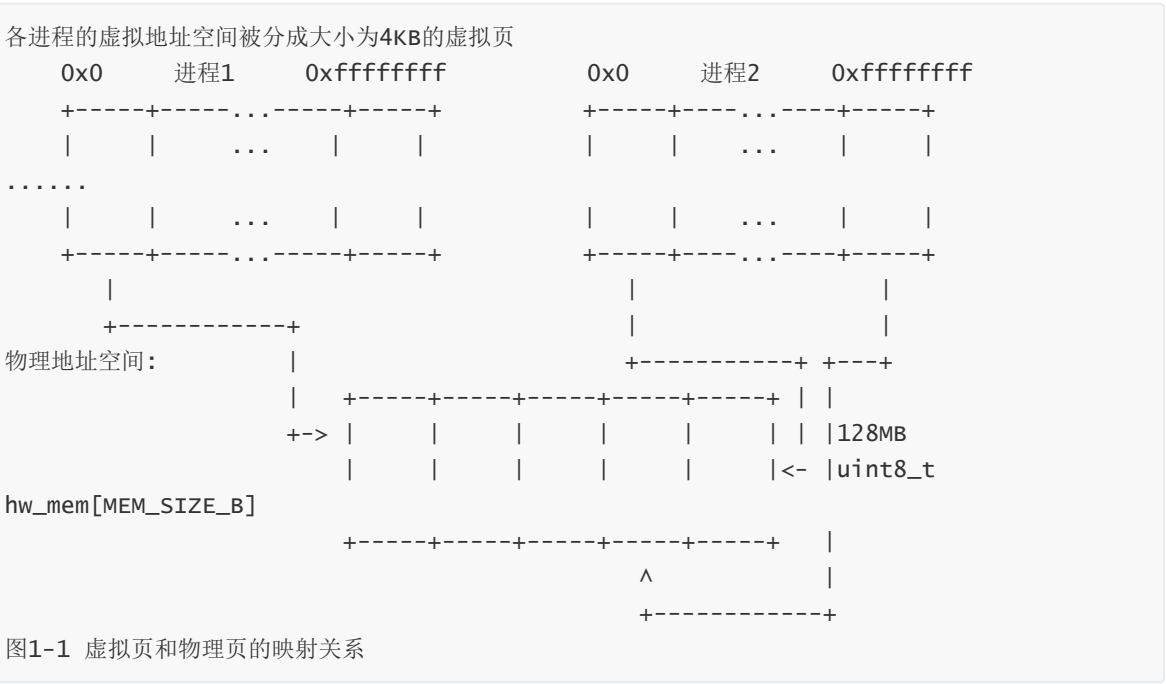

图1-1 虚拟页和物理页的映射关系

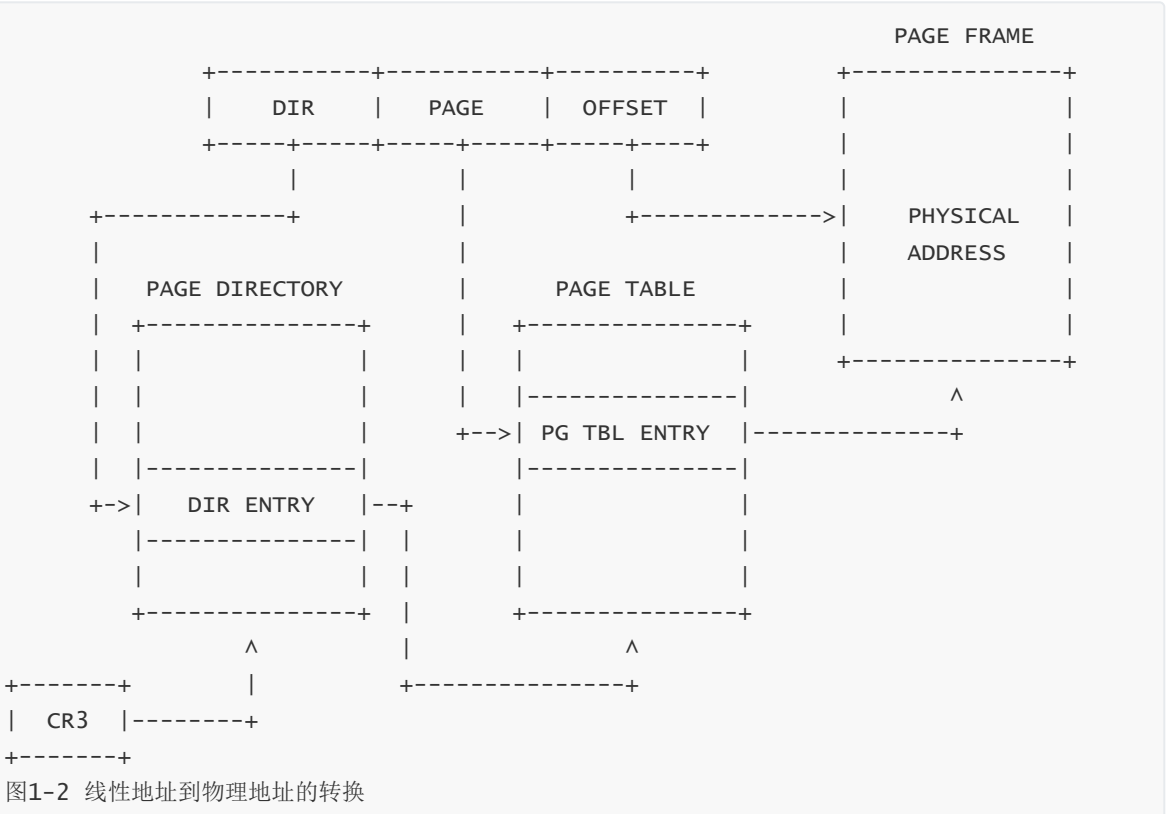

图1-2 线性地址到物理地址的转换

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。

以mov-c测试用例为例, 在`loader()`中通过`Log()`输出`mm_malloc`的结果为:

```
nemu trap output: [src/elf/elf.c,51,loader] {kernel} elf.c: mm_maloc.paddr =
0x1000000, ph->p_vaddr = 0x0
nemu trap output: [src/elf/elf.c,51,loader] {kernel} elf.c: mm_maloc.paddr =
0x1001000, ph->p_vaddr = 0x8049000
nemu trap output: [src/elf/elf.c,51,loader] {kernel} elf.c: mm_maloc.paddr =
0x1002000, ph->p_vaddr = 0x804a000
nemu trap output: [src/elf/elf.c,51,loader] {kernel} elf.c: mm_maloc.paddr =
0x1003000, ph->p_vaddr = 0x804c000
```

由`init_mm()`中的`ucr3.val = (uint32_t)va_to_pa((uint32_t)updir) & ~0x3ff`语句，知Kernel系统的虚拟页地址为`updir[]`数组的首地址， 物理页地址映射在ucr3.val中. 通过`Log()`输出得到:

```
nemu trap output: [src/memory/mm.c,44,init_mm] {kernel} mm.c: updir[] =
0xc0093000
nemu trap output: [src/memory/mm.c,43,init_mm] {kernel} mm.c: ucr3.val = 0x93000
```

Kernel映射至0x93000物理页.

内核映射关系如下图所示:

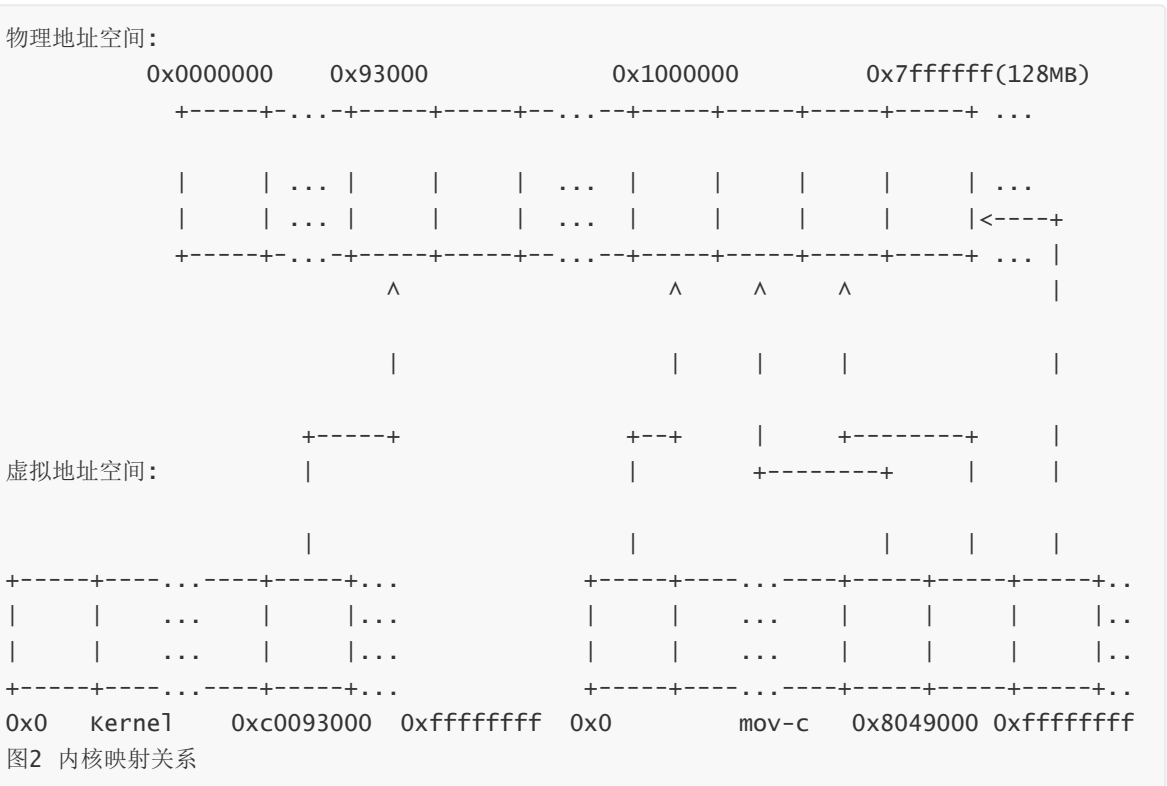

图2 内核映射关系

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?

1)

错误. 当PE,PG位都为0, 即段页机制尚未开启时, 全局变量的逻辑地址就是物理地址, kernel可以直接根据该地址访问到该全局变量.

2)

`init_page()`用于初始化两级页表，将页目录首地址装入cr3寄存器和将PG位置1开启分页. 相当于将全局变量线性地址所在虚拟页映射至内存中的物理页.