***JOS Lab4 文档***

杨健邦 515030910223

*本文档描述了本次LAB各个Exercise实现的方法或者我自己的理解 (只陈序一部分相对有难度比较复杂的设计)。至于对Question的解答以及以及Challenge的实现方法，请阅读answer-lab4.txt。*

**Part A: Multiprocessor Support and Cooperative Multitasking**

**Exercise 1-3.**

* 现在page\_init为物理地址MPENTRY\_PADDR预留一个物理页，然后将AP的启动代码映射到此处。在trap\_init\_percpu使用thiscpu->ts\_cpu0设置好内核栈以及段。由于之前的lab使用的是sysenter来进行系统调用，在trap\_init\_percpu的时候同时也要给每个CPU注册好它相应的MSR，为每个CPU开启systenter系统调用的方式。
* 除了BSP，其它AP都没有开启大页，因此修改pmac.c中的mem\_init，全部采用小页映射的方式。

**Exercise 4-4.1.**

* 在相应的地方拿锁放锁。要注意的是，在systenter的时候也要拿锁，sysexit之前要放锁。在实现ticket spin lock的时候，为了防止乱序或者相应的变量被存放到寄存器中读（但是在内存里的值被其它CPU所更改），有两种解决方法，一种是在定义spinlock结构体中，own变量要加上volatile修饰词，强制每次读的时候都要从内存中读取；另一种方法是读操作也使用原子指令atomic\_return\_and\_add进行读取。
* 由于用sysenter的方法进行系统调用，不会自动保存更新env的trapframe，调用sys\_yield之后，当前env的环境没有正常保存，因此对于sys\_yield的情况需要更新当前env的trapframe中的eip和esp。

**Exercise 5.**

* Round-robin调度方法。分为两种情况，第一种是CPU之前没有运行的CPU（thisenv）为空，那么从0到NENV遍历，找到第一个可运行的env进行调度；第二种是CPU之前有运行过的env，那么从当前env的后一个开始环形遍历，找到第一个可运行的env进行调度。

**Exercise 6.**

* Lab3的sysenter的方法只能传递4个参数，而sys\_page\_map需要使用5个参数，因此需要对lib/syscall进行修改，用%esi来传递第五个参数。在systenter之前，先将返回%eip入栈，因为用户的%esp保存在%ebp中，因此在systenter之后，可以通过%ebp来找到用户的%eip。

## Part B: Copy-on-Write Fork

**Exercise 7-10.**

* 为了使得处理完page fault之后，用户env的trapframe能够恢复正常，需要将trameframe的eip压进用户env运行的栈，然后将esp设置成trapframe中的esp – 4的值，此时使用ret指令便可以同时使得eip和esp恢复正常。
* 要理解vpt和vdt为什么分别对应着所有的page table和directory table。虽然在映射的时候，只做了一个将UVPT映射到pgdir的物理地址的映射。但是，根据地址翻译的方法，相当于当前环境的所有页表就在UVPT-UVPT+PTSIZE的这一段虚拟地址上（一共1024个页表，正好等于PTSIZE），页表按顺序排列，从UVPT排到UVPT+PTSIZE。Page directory table也被看成是一个page table，在(UVPT+(UVPT>>12)\*4)的虚拟地址处，它相当于第UVPT>>22个页表。相关链接：<https://pdos.csail.mit.edu/6.828/2014/lec/l-josmem.html>

**Exercise 11.**

* 要将parent和child的UTOP以下的所有页，除了user exception stack，在parent和child的页表中映射成COW的方式。这样他们对页数据的读取和修改就不会互相影响。
* 由于user exception stack就是用来处理这些COW页的，因此必须给parent和child分配不用的页，不能是COW的形式。

## Part C: Preemptive Multitasking and Inter-Process communication (IPC)

**Exercise 12-14.**

* JOS采用了一个CPU一次只会处理一个中断的方式，这就意味着，在进入内核态处理中断的时候要关中断。在设置trapgate的时候有一个参数是istrap，当istrap参数是0的时候，CPU处理该中断时会自动关闭中断，在JOS中，应该要将所有的interrupt/trap gate设置成interrupt gate，也就是istrap应总是设置为0。
* 我们的系统调用采用的是systenter的方法，执行systenter 指令会自动关中断，但是sysexit并不会打开中断，因此，在sysexit之前还需要使用sti指令来打开中断。