***JOS Lab6 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。至于对Question的解答以及以及Challenge的实现方法，请阅读answers-lab6.txt。*

**Part A: Initialization and transmitting packets**

**Exercise 1.**

在trap.c中，在处理时钟中断的时候，调用time\_tick()来计时，但是要注意，每一次时钟中断都会向所有CPU发送，为了防止重复计时，当且仅当CPU0处理时钟中断的时候调用time\_tick() 。

**Exercise 3.**

查阅E1000的手册，得到相关网卡设备的DEVICE ID和VENDOR ID，加到数组里面，在e1000的attach函数中，调用一下pci\_func\_enable即可 。

**Exercise 4.**

在e1000的attach函数中，将bar 0(物理地址)映射到KSTACKTOP的地方，权限位应该置为PTE\_P | PTE\_W | PTE\_PCD | PTE\_PWT，因为这里KSTACKTOP的地方之前没有映射，因此它不会出现在TLB之中，也不需要invalidate TLB。用一个带有volatile关键字的全局变量bar0来记录其起始的虚拟地址，也就是KSTACKTOP，当映射完成之后，assert(bar0[E1000\_STATUS] == 0x80080783)来确保映射成功。

**Exercise 5.**

通过设置全局变量的方式，使得descriptor ring buffer所在的物理页在链接和加载程序的时候被自动分配在连续的物理地址空间，在设置全局变量的时候，加上\_\_attribute\_\_((aligned(16)))保留字来使得链接器会将这个buffer放到16字节对齐的地址。

同时为了防止出现cache与memory之间存在不一致问题，在attach的时候，将descriptor ring buffer以及packet data buffer所在的物理页全部映射成PTE\_PCD和PTE\_PWT的形式(禁用缓存以及写穿)。

其它直接根据手册一步步初始化即可。

**Exercise 6.**

发送网络包的函数的函数名是e1000\_transmit，由于传入的指针参数和长度的有效性在syscall里面已经进行检验了，因此这个函数里面不再需要检验。通过检查DD是是否被置上来判断ring buffer是否已满(所有desc里面的DD位在attach初始化的时候就已经被清掉了，由于不支持长包，所以所有的desc在初始化的时候通过也把EOP位给置上了)。

**Exercise 7.**

与sys\_ipc\_try\_send类似，我定义的新的系统调用的名称是sys\_net\_try\_transmit。这个系统调用首先使用user\_mem\_assert来确保用户传进来的指针是有效的，以及指针所指向的空间足够大道容纳一个网络数据包的内容，之后直接调用e1000\_transmit函数即可 。

**Exercise 8.**

output函数是一个死循环，首先使用之前的ipc\_recv接口，不断尝试，接收一个ipc消息，当接收到IPC且IPC是来自ns这个env 且IPC的值是NSREQ\_OUTPUT的时候，才会进入另一个循环，不断调用sys\_net\_try\_transmit来尝试发送网络包。

# Part B: Receiving packets and the web server

**Exercise 10.**

对于receive所需要的物理空间分配，跟transmit的物理空间分配类似，这里就不多叙述。初始化工作同样是在attach函数中，根据手册一步步进行即可。

**Exercise 11.**

和e1000\_transmit相似，receive函数被命名为e1000\_receive，用户传入的指针的有效性检查已经在 syscall层进行检验了。如果没有要接收的包，则返回一个-E\_NO\_RECV\_PACKET错误码让用户自行处理。

**Exercise 12.**

写一个while循环，首先通过syscall来接收一个网络包，如果没有收到网络包的话，则调用sys\_yield()来放掉CPU资源，等待下次被唤醒继续查看是否有收到网络包。如果有收到网络包，则新分配一个页，把网络包的数据拷到这个页之中，再通过IPC将数据发给ns env，这里不能使用之前的页，因为当接收到一个新的网络包时，不知道ns env有没有处理上次通过IPC发给其的数据，重复使用同一个页会导致数据因覆盖而被丢失。

**Exercise 13.**

通过open来判断文件或目录是否存在，通过fstat得到一个struct Stat，然后判断路径是否是一个目录，如果不存在或者是一个目录则调用send\_error来返回错误。在send\_data函数中，有个循环，每次从文件中读入至多1518个Byte，然后写到sock对应的fd中，直到不能从文件中读入更多的Byte为止。