Lab6 文档

网络驱动写的比较简单应该是简单到不能再简单了，因此发包效率也是很低的，但是因为以太网对于网络环境的假设很少，数据的准确性，数据的发送速度，以及组帧等都是上层程序来做的。因此网络驱动效率低的话，程序会自动降低发包速度。发错的话，上层的程序也能从发送的内容中获知，然后重复发包，总有成功的。

因为有了前面 dma 的经验，对于硬件相关的编程已经有所了解，因此这次只要对照着 intel 文档写就基本可以了。还稍微看了看 linux2.6 对于 e100 的实现。数据传输只要知道数据的地址和大小就可以了，我写的接口就传入这么两个参数。我在管理 CBL 的时候考虑到整个 LINK 是一个环，于是用了两个指针，一个指向头一个空的 Block。一个指向最后一个 Block。每次分配一个 BLOCK，指向头的 BLOCK 往后一个 BLOCK。每次释放一个 BLOCK，指向尾的指针往后一个 BLOCK。每次要传输之前查看成功的 BLOCk，并且释放。RBL 也差不多，Lab 的要求的意思是可以用中断也可以不用。我就没用中断，反正丢包不怕。

看了数据传输的接口

```
struct jif_pkt {
        int jp_len;
        char jp_data[0];
};
```

在 ipc 的时候 map 的 page 其实里面就是这个结构。

Output 是接到请求就调用网络传输的 syscall。Input 一开始以为也是接到请求才干事，后来看看没反应，看了相关代码，于是改成主动和 serv 进行 ipc。一开始我在 Input 里面每次读数据都写进同一块页里面，但是发现会出现下一次的数据将上一次的覆盖的情况。于是我每次读取数据，Input 都申请一块页面，用完释放。

在 httpd 里面是基本很简单，就是因为对于发的包有大小限制，因此在发送数据的时候将大的数据拆成几个小的包来发。

硬盘读写用的是 DMA。

404Err 的通不过，不过看看代码完全没有问题，wget 的 log 对的。想上去应该是 wget 版本问题。中文版显示的是错误 404，而测试代码要求 Err404.

8 小时阅读 intel 文档。一天 code&debug。