我这星期都学了以下内容：

# 1.系统调用

《操作系统概念》第二章有提到系统调用，我觉得讲得比较模糊，于是查看了xv6的中文文档，并阅读了相关代码，里面比较详细地描述了系统调用的执行过程。

在X86系统中，程序通过int指令产生一个中断来进行系统调用，具体就是调用int n指令。随后int从IDT中获得第n个描述符，检查cs中CPL是否小等于描述符中的DPL。如果小等于，则从任务段描述符中加载ss和esp，然后将ss、esp、eflags、cs和eip分别压栈，随后清楚eflags中的一些位，病设置cs、eip为描述符中的值，然后从用户态转换为内核态继续执行指令。最后是通过iret指令从内核态返回用户态，完成一次系统调用。

# 2.X86体系结构

我在学习系统调用的时候认识到，如果不先去学习X86的体系结构，是很难真正理解操作系统的实现过程的，于是我阅读了《linux内核源码剖析》里面的第四章节。

## 2.1 X86重要的寄存器

首先我学习了X86里面重要的寄存器。

X86有一个标志寄存器eflags和几个系统寄存器。eflags中有5个系统标志位，分别为TF、IOPL、NT、RF和VM。它们的功能在此就不细说了。

然后是非常重要的内存管理寄存器，CPU中有4个内存管理寄存器，分别是GDTR、LDTR、IDTR和TR。前三个寄存器用于寻址存放着描述符的段，TR寄存器用于寻址任务状态段TSS。它们和CPU的内存寻址关系紧密。

还有控制寄存器CR0到CR3。其中CR0与协处理器密切相关。

## 2.2 内存管理

这部分下星期再看

# 3. fork函数

老师上课的时候让我们回去看fork源码，于是我回去看了。

Linux源码中fork函数是sys\_fork()系统调用的辅助处理函数集，sys\_fork()主要使用了fork.c中的find\_empty\_process()和copy\_process()。因为我还没看汇编与C，所以先不看这个系统调用函数的实现了。

fork.c文件中主要函数是copy\_process()，这个函数用于创建并复制进程的代码段和数据段以及环境。进程复制工作中主要是在设置进程数据结构中的信息，首先是为进程结构分配内存，这里需要为新建进程在主内存区中申请一页内存。这里需要使用verify\_area()，用于验证内存的起始位置和范围。申请内存成功后，使用copy\_mem()函数复制当前进程任务数据结构中的所有内容作为新进程任务数据结构的模版。随后对新进程的一些信息进行修改，比如新任务代码段和数据的基址和段限长。

不得不承认我对这段代码看得不熟，以后对操作系统有更全面的认识，还要来阅读一次。

# 4. 进程控制

我阅读了《unix环境高级编程》的第八章进程控制。里面比较详细得介绍了创建进程、执行进程和终止进程。学习了fork函数和vfork函数。学习了进程8中终止方式。

本章还介绍了竞争条件。竞争条件的定义就是当多个进程都企图对共享数据进行某种处理，但是最后的结果又取决于进程运行的顺序，那么这就产生了竞争条件。文中讲述了可以使用poll或者信号机制来处理竞争条件。

# 5. 课文内容

我把《操作系统概念》前三章看完了。

# 6. 下星期预计的学习任务

看《linux内核剖析》的第三章汇编与C，并把第四章X86体系结构。

看《unix操作系统设计》的进程结构和进程间通信。

将《操作系统概念》看到老师上课讲的内容，并且根据情况进行课外补充阅读。