OS假期预习微型指导书

前言

首先，恭喜各位通过了计组的挑战来到了大二的下半段。在这个学期，您们将会学习操作系统这门课程，与计组类似，该课程也分为理论课和实验课两部分。本文旨在让同学们了解操作系统课程的**主要内容、考核机制与学习经验。**

声明：本文所依据的课程情况为2017级6系学生的操作系统课程情况，不排除与18级实际情况有所出入的可能性。

操作系统实验课程

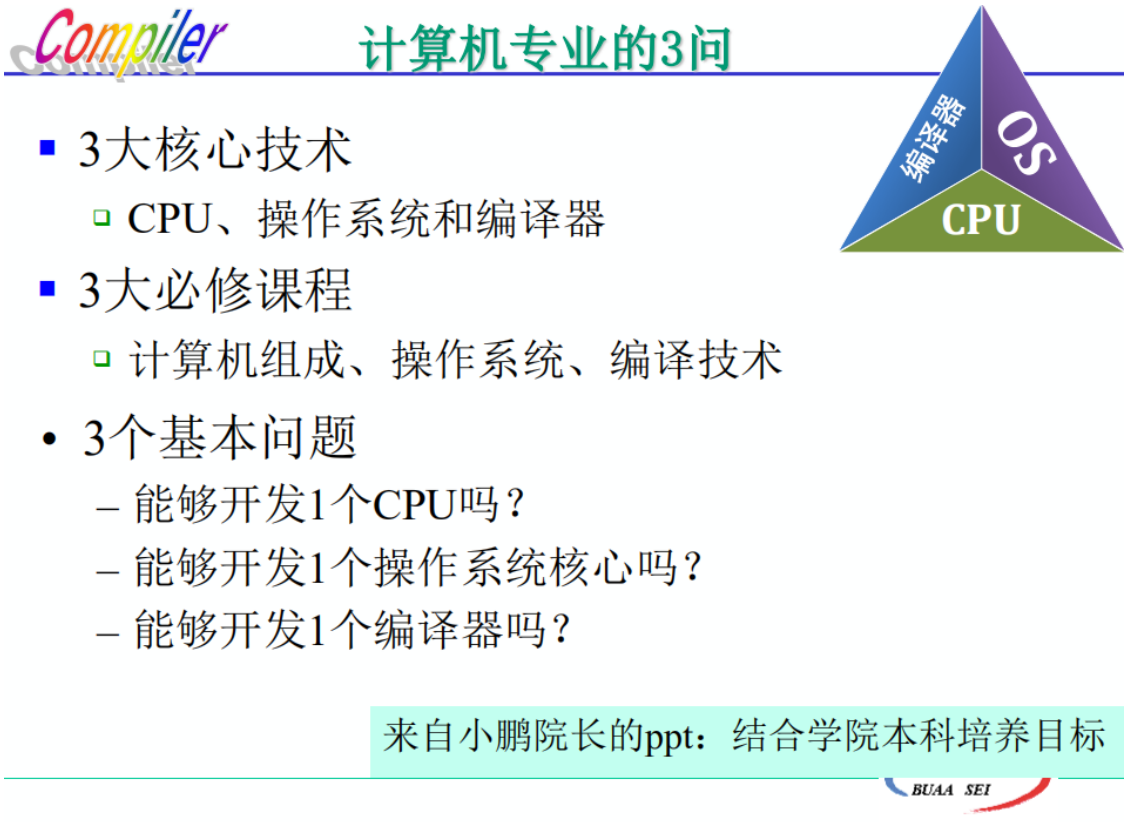
课程目的

> “掌握操作系统原理的最好途径就是自己编写一个操作系统，我希望大家 都能写出自己的操作系统。”
>
> ——
>
> 王雷

与计组实现$MIPS$微系统类似，操作系统实验课的目的是实现一个运行在$MIPS$架构CPU上的简易操作系统。

为什么需要开发一个操作系统

- 首先，这是计算机专业三大核心之一(摘自编译技术课件)



- 在实践中将理论课所学习的内容消化并运用
- 极大地锻炼工程能力与系统能力

实验环境相关的提醒

实验环境仅支持CLI

近期看到大家都在尝试安装并使用Linux系统，这里就简单地提醒一下大家。

首先，我们课程组会为每个同学提供**完整的实验环境**，可使用ssh进行连接。实验环境至少包括以下内容。

- gxemul (运行操作系统内核的仿真软件)
- git
- vim
- nano
- gcc

同学尝试自己本地配置实验环境值得鼓励，**但不是必须。**

课程内容

实验分为$lab0 \quad - \quad lab6$与挑战性任务共8个模块。

大家可能还看不懂下面内容的具体含义，不过不必担心，都会学到的。

- $lab0$ 主要是熟悉基础的命令行操作，以及基本工具(比如git、 vim)的使用。是同学们适应操作系统实验环境的环节
- $lab1$同学们将初步了解什么是内核，并接触一些与编译、链接相关的基本内容与工具(比如$Makefile$、ELF)此外同学们将自己实现C语言的printf的部分功能
- $lab2$是关于内存管理的部分，主要涉及MMU、TLB、地址映射、页表与TLB缺失等内容
- $lab3$是关于进程与中断异常的章节。需要实现进程的创建、运行、调度
- $lab4$首先将学习系统调用($System\ Call$)的实现机制，并实现一些系统调用函数。然后实现进程间的通信机制(IPC)，最终实现FORK机制
- $lab5$是关于文件系统的，$lab6$是关于管道与$Shell$的，涉及内容较多，不再一一列举。
- 挑战性任务是基于各个lab的内容的拓展延伸

注：以上内容更详细信息可以参考实验指导书。

看到这么多内容大家可能会开始害怕，内容太多，工程量太大。不必担心！

原因如下。

- 我们小操作系统只是一个功能较为简单的"玩具级"工程(当然，dalao可以在此基础上玩出花来)
- 各位的工作并不是像计组那样从0开始实现一个CPU，而是课程组给出大部分的工程代码，大家只需要实现部分核心功能即可

考核形式

由于与计组课设在考核形式上有所相近，故做比较进行介绍。

相似之处

- 分为课上测试与课下测试
- 每次测试的题目为当前部分内容，即不会出现$lab2$出$lab0$的题目的情况
- 课上测试不通过可能是由于**课下BUG**所致，且曾经出现$lab5$因为$lab1$的$printf$的BUG而翻车的惨案

不同之处

- $lab0$仅一次课上实验，$lab1-5$各两次，$lab6$仅有课下测试(不代表没有现场考核，后文详述)

习惯上，我们称某次实验的名字采用以下方式，$lab5-2$代表$lab5$的第二次课上测试。

值得注意的是，分为两次的lab，其考核内容也分为两次，比如，$lab4-1$考核IPC的内容，$lab4-2$考核FORK的内容。

由于操作系统本身复杂性，每个lab如果仅仅完成了第一部分内容，本地(实验环境)运行可能无法得到任何实验结果，**只有正确完成**了该lab的全部内容才可以在本地(实验环境)运行得出正确现象。不过，**这并不影响课下自动化测试。**即，如果你第一部分正确完成了，但没有完成第二部分，本地虽然无法跑出现象，但是不影响我们对你第一部分的自动化测试。

- **不是闯关制！**即，如果本次实验$fail$了，下次实验如果通过，则记本次实验通过。

  举个例子，我$lab3-2$挂了，但我$lab4-1$过了，则旧账一笔勾销。

- 课上测试的有两道题目，一道必做的基础题，一道选做的附加题。

  通过本次测试的条件为通过基础题的测试。

  如果通过附加题却没有通过基础题，则附加题无效。

  通过任何一道题目的条件是该题目得分不低于60分(满分100分)(我们会设置若干个测试点，每个测试点赋予不同分值)

- 课下测试未通过不影响参加课上测试的资格(最好还是把课下做到100分叭，不然去课上多难受：))

关于评分制度

今年的评分制度可能做出一些调整，往年的经验具有一定参考价值但不代表今年情况。

需要的基本知识

- 熟练掌握C语言，尤其是指针、结构体
- 对MIPS有基本了解(包括但不限于)
  - 会编写汇编程序
  - 了解软硬件交互方式
  - 注意: 计组时涉及不多的tlb相关内容将会用到 :-)
- 了解异常与中断

课程难点

- 理解代码

  这是最大的难点。虽然需要各位编写的代码量不大，但是，需要各位熟悉课程组提供的代码。即，不属于您们亲自编写的部分。OS最大的难点就在于你需要把整个操作系统代码的逻辑大致捋清楚而不是在于实现几个简单的功能。

- 开发环境恶劣

  大家大多数人已经适应了IDE的温床，现在你失去了图形化界面，也失去了完善的断点调试功能。有的只是黑漆漆的屏幕、汇编代码为单位的断点调试与二进制码。这对大家形成了不小的挑战，这个时候一切都可以利用起来，printf大法(似乎还得自己实现：))、反汇编技术、目力debug等等。

- 基础要求高

  为了理解一些内容，大家可能需要重新捡起在计组课程、数据结构课程甚至程序设计课程的一些细碎的小知识。

  不过，不必担心，一点点而已，不会让您们重头再来的，遇到不会的随手查一下就好：)

学习建议

我们建议

- 抱团取暖
  - 积极参与讨论，交流对课程组代码的理解的经验甚至提出对课程组代码的质疑都是很好的学习方式
  - 互相交流测试方法
  - 一同de神秘的玄学bug
- 理解至上

  课上测试的题目很多都涉及到了并非课下题目直接相关的内容，而是课下题目连带着的机制。我们**要求**您们阅读有关代码，理解系统的工作机制。
- 尝试"骚"路数
  - 大胆创新不仅仅体现在我们做出来的成品上，也可以体现在我们实现这一成品的方法中。

    比如，17级的何岱岚同学 ：- ）利用系统调用机制实现了一个高效的断点机制。
  - 挑战性任务就是一个尝试各种骚路数的平台，欢迎大家展示自己的真知灼见
- 理论指导实践
  - 操作系统的代码每一部分都有其对应的功能模块，了解这个模块的功能与其他基本信息可以帮助我们更好地理解代码
  - 这些基本信息在理论课中基本上会覆盖

推荐参考资料

- **《See MIPS Run Linux》**
- 《深入理解Linux内核》(比较硬核，学有余力的同学可以尝试)
- JOS 相关网课或博客
- 前辈的有关博客
- **github**

操作系统理论课程

注意：3学分

考核形式

与传统的理论课程大体一致。有期中、期末考试，有平时分。

试题有难度、有区分度。需要做到深入理解与记忆。

与实验课联系

这是笔者认为操作系统课程最为出彩的几点之一。OS理论和实验结合较为紧密。OS理论课所讲的内容可以较好地辅助大家理解实验课程的代码，实验课的内容也可以加深对理论课程的理解。

结语

大二下会是非常辛苦的一个学期，希望大家合理安排时间，从各个课程中真正学到知识，而不仅仅是为了混分数。祝各位在操作系统理论课程和实验课程中取得满意成绩。