## 项目概述和开发环境配置

这是一篇阐述如何在基于Intel x86架构的IBM PC机及其兼容计算机上构建一个简单

的操作系统内核的本科毕业设计论文。我将带领大家一起来探索x86CPU的保护模式下操作系统内核的编写方法，一起感受一次完整的探索过程。虽然这个小内核和一个具有商业价值的操作系统内核相较而言依旧相差甚远。但是通过这样的探索，相信我们能充分的理解x86保护模式的运行方式和操作系统的基本原理，而这恰恰是传统的通过理论教学和阅读书籍的方式难以获得的深刻体验。

言归正传，开始我的介绍。工欲善其事，必先利其器，我先来阐述下开发环境和相关的工具配置。

### 工作环境

Windows 和Linux 之争由来已久，我不想在这篇论文里针对这个问题再费口舌，我

的工作环境选择Linux。使用Linux的原因很简单，这里有可以自由使用的一系列的开源软件能很好的协助我们的开发和调试工作，而在Windows下缺乏相应的免费工具。虽然我的构建环境使用的是Ubuntu 16.04 i386 Desktop，但是这不影响大家在对项目验证时对于Linux发行版的选择，因为使用的命令基本上都是相同的。经过四年本科的学习，我们对一些Linux基础的命令和基本的计算机概念有一定的理解和掌握，包括而不限于：

• 熟悉微机原理和基本的操作系统原理，了解基本的计算机原理概念。

• 了解和熟悉Intel x86保护模式下的一些名词和概念，至少需要熟悉Intel 8086.

• 熟悉和掌握Linux的常用命令，能在Linux下进行基本的系统程序编写。

• 掌握简单的x86汇编语言，能读懂和编写简单的汇编程序（至少能看懂）。

• 熟练掌握C语言程序的编写，对C语言中较为复杂的语法有所了解。

• 理解和掌握C语言程序编译的过程，了解链接的基本原理。

• ……

学习本来就是一个从无到有的过程，操作系统内核的编写本来就是一个及其复杂和麻烦的过程。我会尽量降低这个小内核的难度，给充满热情但相关基础较为薄弱的读者阐述尽可能多的背景资料和原理解析（至少也会给出参考资料的链接）。我相信哪怕你之前的基础再弱，至少也能"照猫画虎"的构建出一个可以在裸机上运行的小内核。尽管我做的东西甚至只是一个基本原理的演示，但那也是实打实的可以运行在裸机上的小内核。

### 开发语言

### 开发工具

接着是选择开发使用的工具了，这个我简单罗列出来。首先C语言的编译器肯定使用

gcc，链接器自然也就是ld了。同时大项目自然也少不了GNU make这个构建工具。至于

汇编编译器我们选择nasm这个开源免费的编译器，以便使用大多数读者习惯的Intel风格的汇编语法。不过考虑到需要在一些C语言代码中内联汇编指令，而gcc使用的是AT&T风格的汇编语法，所以我还是稍稍学习掌握了一部分的AT&T风格的语法。这些就是开发使用的基本工具了，其他的工具我会在使用的时候再介绍。

我们写用户级别程序自然可以直接运行，现在是要写一个操作系统内核。我们在哪里运行它？我们可以使用虚拟机。不过我们这次使用的不是大多数人熟悉的Vmware或者Virtual Box，而是一款叫做bochs的虚拟机。为什么呢？因为有调试的需要。我们需要一个能调试其上运行着的操作系统的虚拟机，而bochs是个不错的选择。选择另一款叫做qemu的虚拟机也支持调试，但本着简单易用的原则，这里不选择qemu。

bochs的安装方法很简单，以Ubuntu为例，只需执行以下命令即可。

sudo apt-get install bochs bochs-x

### 开发中用到的脚本文件

#### Makefile

#### kernel.ld

#### bochs 虚拟机的配置文件

## 总体设计

## BootLoader

## 内核雏形

## 进程管理

## 输入/输出系统

## 进程间通信

## FAT16文件系统

## 内存管理

## C运行时库与简易Shell