东莞理工学院

操作系统课程设计报告

院　　系：计算机学院

班　　级：14 软卓

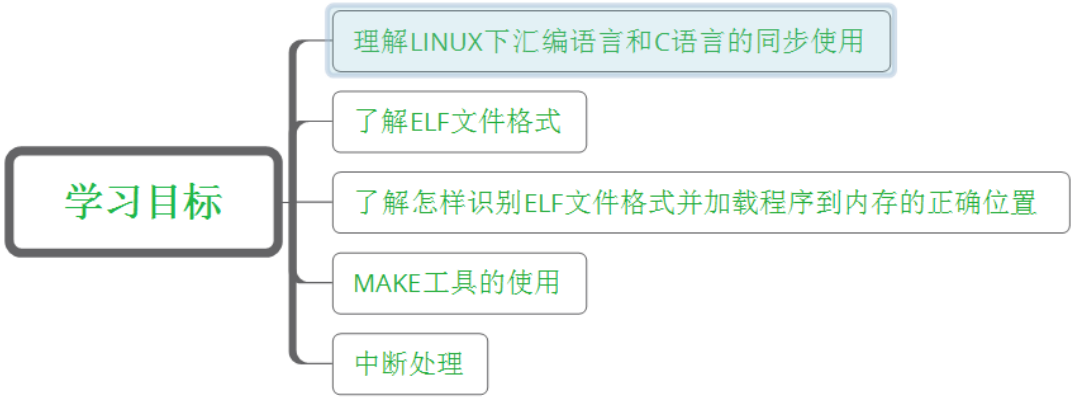
姓　　名：赖键锋

学　　号：201441402130

指导老师：李伟

日　　期：2016.6 - 2016.7

一、 相关说明



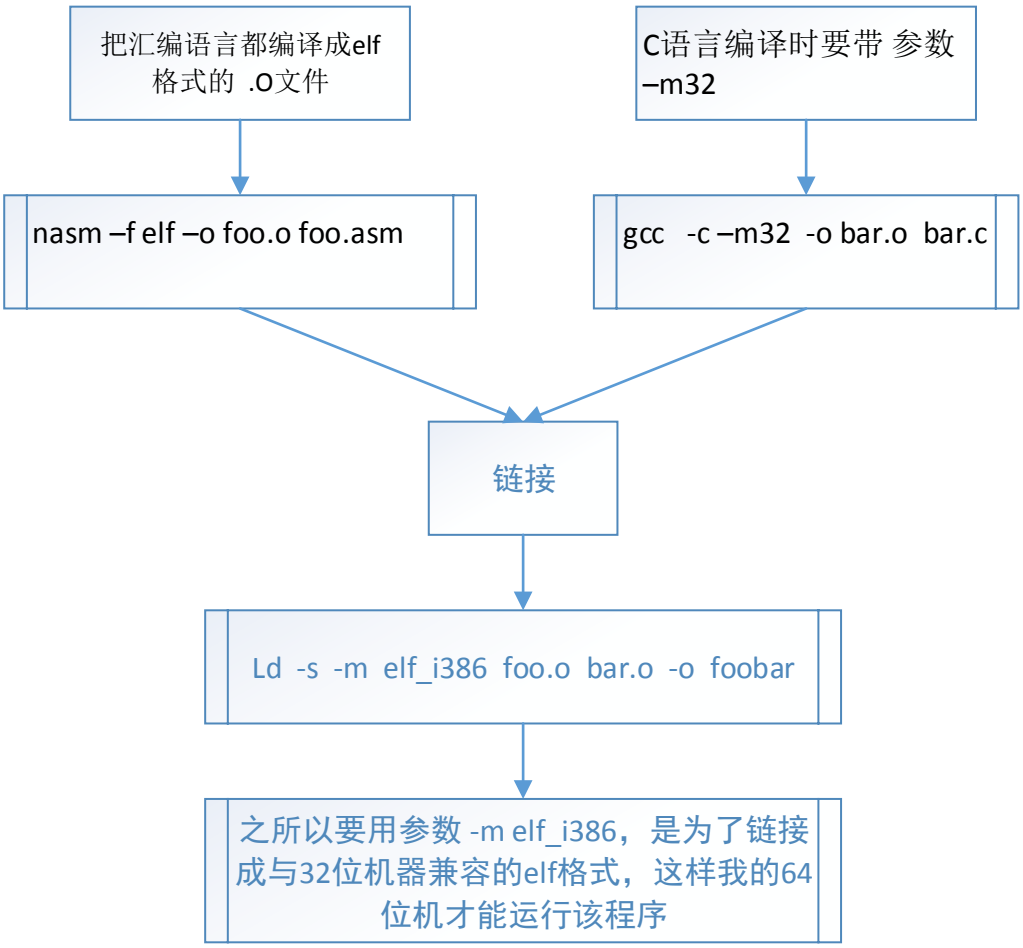
二、 相关知识的记录和说明

1、汇编文件中,到处文件内的函数给外面用要用关键字 global 导出;

2、要用到文件外定义的函数，要用 extern 导入;

3、函数参数，后面的参数先入栈，并由调用者清理堆栈;

4、汇编语言和 C 语言一起使用的时候:



5、ELF 文件

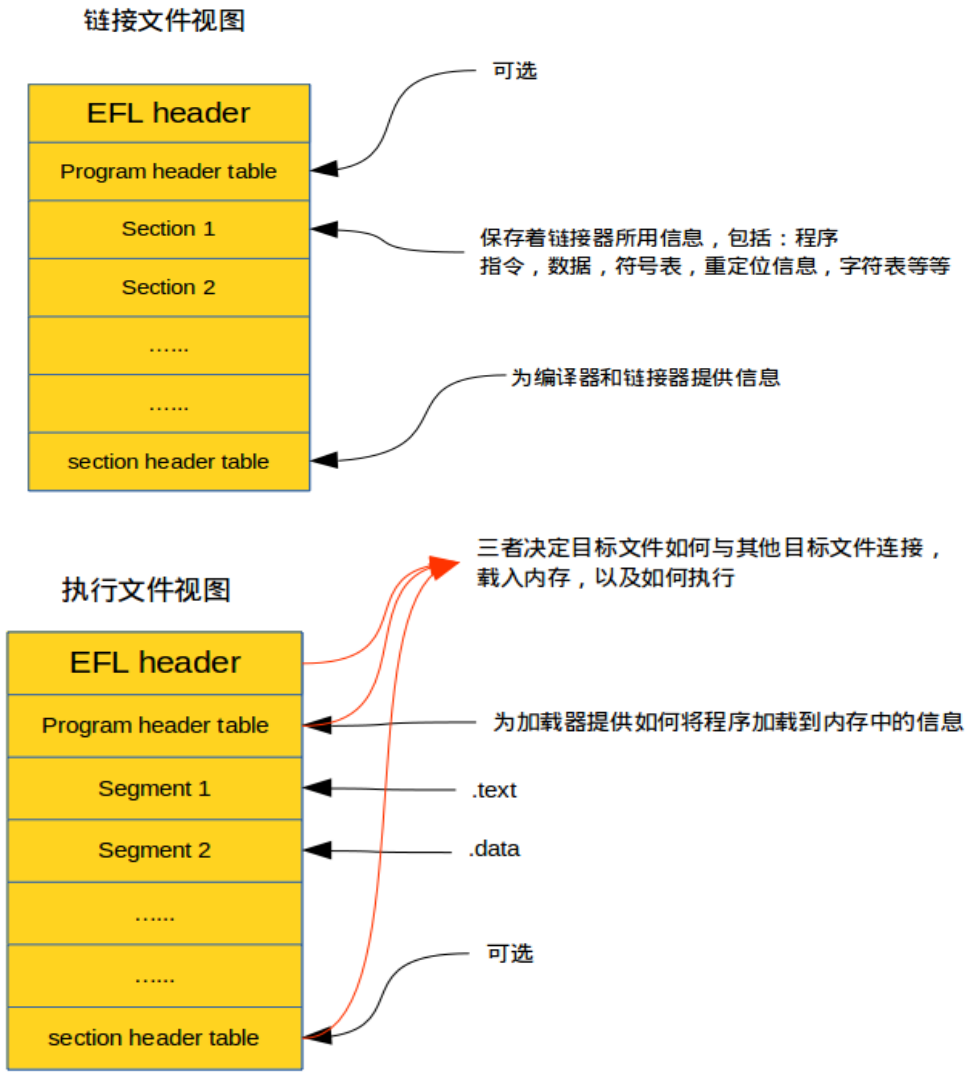
ELF 的三种可执行文件格式都很好的体现了设计思想中分层的概念，由一个总的头部刻画了文件的基本要素，再由若干子头部/条目刻画了文件的若干细节。

ELF 的作用：

1). 如果用于编译和链接(可重定位文件),则编译器和连接器将把 elf 文件看做是节头表描述的节的集合,程序头表可选.

2). 如果用于加载执行(可执行文件),则加载器将把 elf 文件看做是程序头表描述的 segment 的集合,一个 segment 段可能包含多个节,节头表可选.

3). 如果是共享文件(库文件),则两者都含有。



- Program Header 描述的是一个段在文件中的位置，大小以及他被放进内存后所在的位置和大小。如果我们想把一个文件加载进内存的话，需要的正是这些信息。
- 重定位是将符号引用与符号定义进行连接的过程。例如，当程序调用了一个函数时，相关的调用指令必须把控制传输到适当的目标执行地址。
- 一个可执行程序通常是由一个含有 main() 的主程序文件、若干目标文件、若干共享库（Shared Libraries）组成。一个 C 程序可能引

用共享库定义的变量或函数，换句话说就是程序运行时必须知道这些变量/函数的地址。在静态连接中，程序所有需要使用的外部定义都完全包含在可执行程序中，而动态连接则只在可执行文件中设置相关外部定义的一些引用信息，真正的重定位是在程序运行之时。

6. objdump 命令是 Linux 下的反汇编目标文件或者可执行文件的命令;

readelf <选项> elf-文件 ： 显示关于 ELF 格式文件内容的信息.

7. 用 Loader 加载 ELF, 处理内核时我们需要根据 Program header table 中的值把内核放到正确的位置。

加载文件最重要的是完成两件事情:

- 加载程序段和数据段到内存;
- 进行外部定义符号的重定位。

Makefile 的编写

- 最重要的语法:

  Target : prerequisites

  Command

- Target 依赖 prerequisites, 当 prerequisites 中至少有一个文件比 target 文件新时，command 才被执行。
- $@ 代表 target

- $<代表 prerequisites 的第一个名字。
- .PHONY 关键字后面仅仅表示一种行为的标号

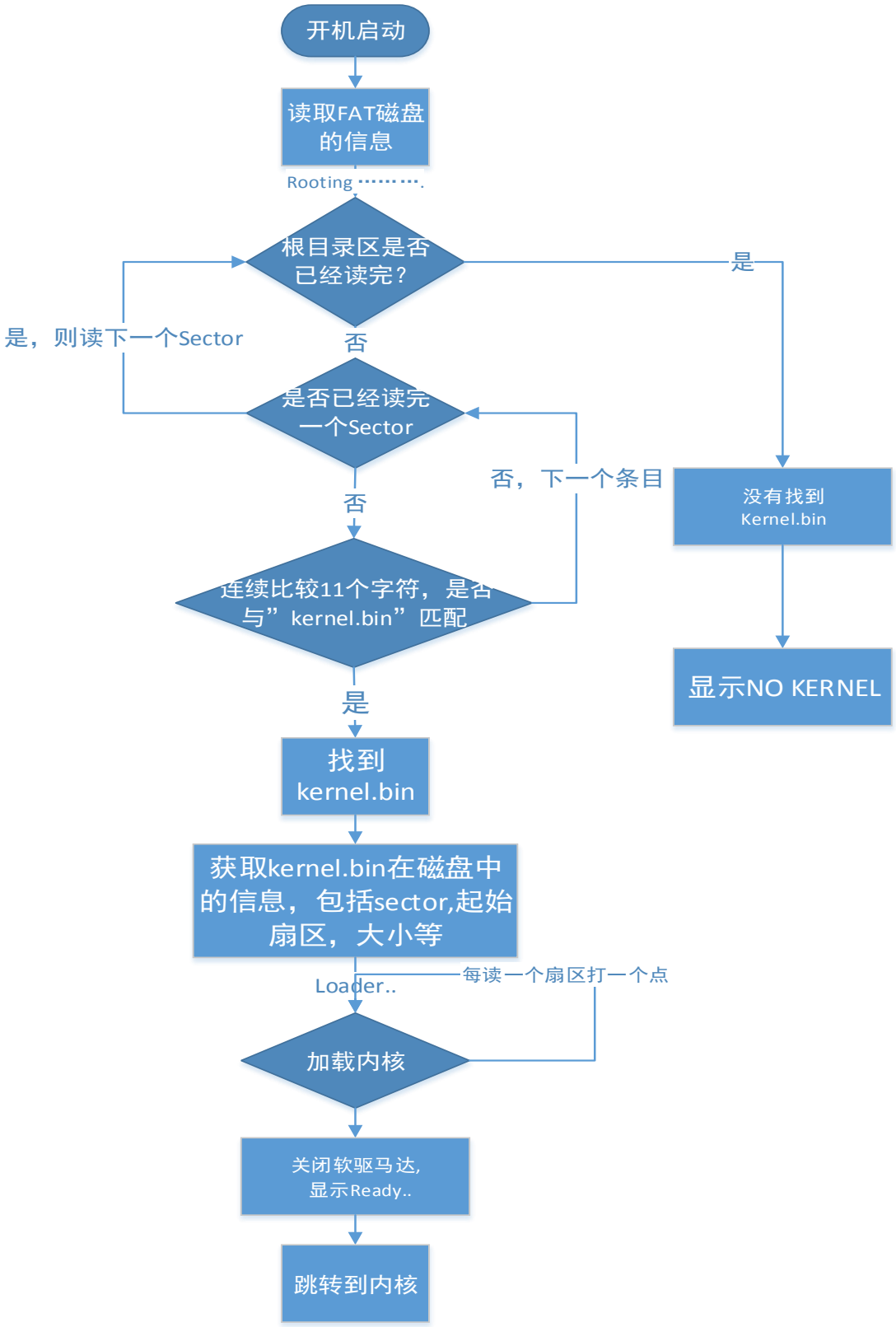
Make 程序的原则是由果寻因，先看要生成什么，再找生成它需要的条件。

中断处理

1) 设置 8259A：
   - 初始化 8259A 的步骤是固定的；
   - 想要打开或屏蔽某中断，只要往 8259A 写入 OCW1 就可以了，如果要屏蔽某中断，则通过 OCW1 把对应的位设为 1
2) 建立 IDT
   - 定义 IDT 描述符 GATE 的结构：h/include/protect.h
   - 声明全局变量，用 EXTERN
   - 中断和异常表
   - 对异常的处理思想：如果有错误码（如果没有，则把 0xFFFFFFFF 压栈，表示没有错误码），则直接把向量号压栈，然后直接执行异常处理的控制函数 exception_handler；
   - exception_handler 的执行过程：打印空格，把屏幕前五行清空，根据向量号打印数组中的异常信息，然后打印堆栈中的信息，包括寄存器的值；

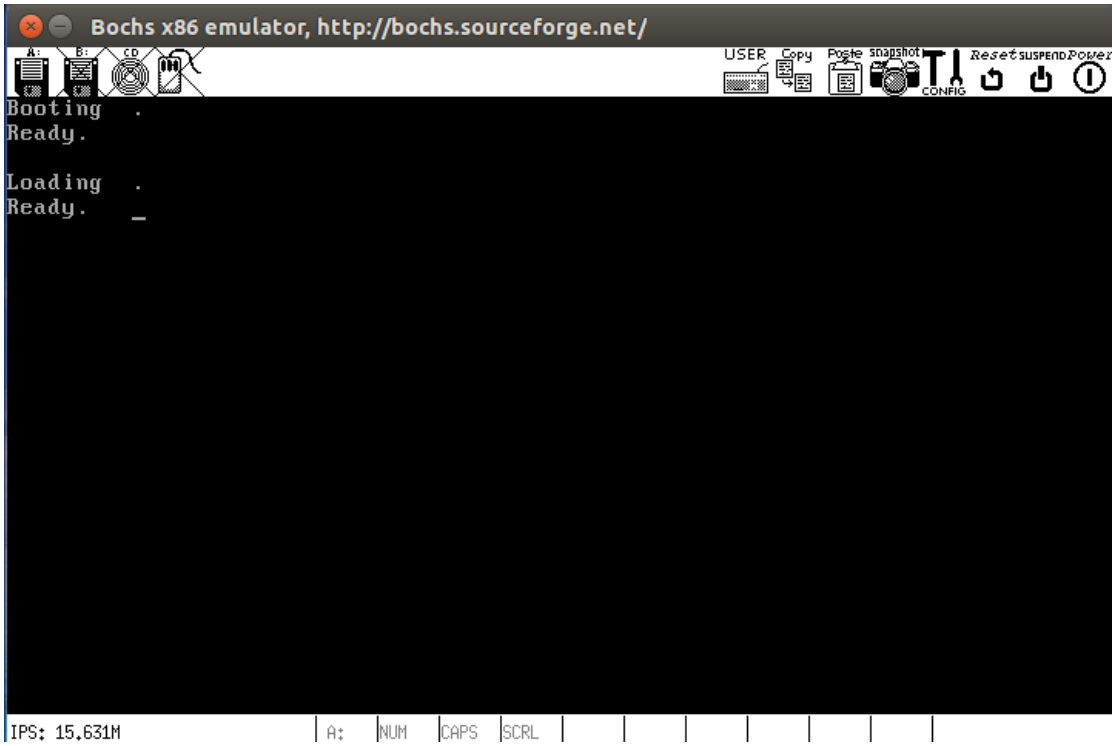
三、 程序关系图或流程图

8. 寻找内核，加载内核

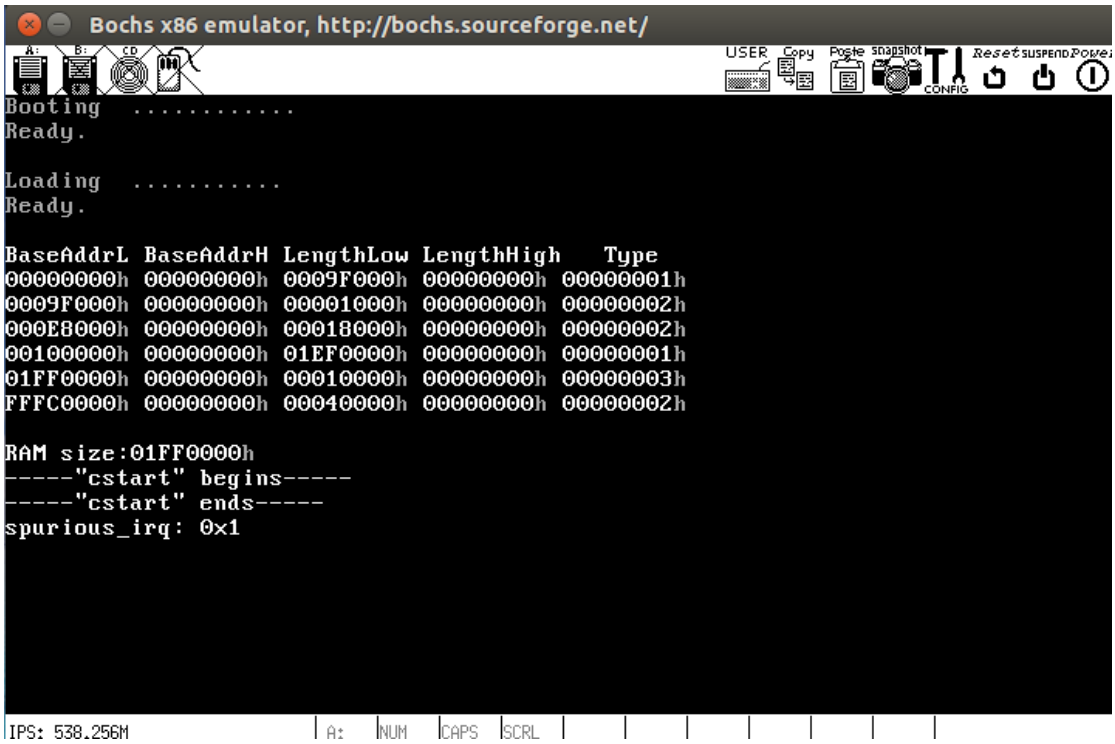


四、 运行过程及理解

5.c



5.i



五、 重点知识总结