任务四：Createimage 开发

1. 介绍

实现一个 Linux 工具，将启动程序和内核编译为一个 MIPS 架构支持的操作系统镜像。

编译程序编译出的可执行文件需要通过一个 Linux 工具（本实验中为 createimage）转换为内核镜像，内核镜像被写在磁盘中，供系统启动时加载使用。该工具将可执行文件转换为硬件支持的格式，可以被直接加载和运行。

1.1. 需要了解的部分

- ELF文件格式

2. 初始代码

2.1. 文件介绍

- bootblock.s: 程序运行最开始执行的程序，上一阶段完成
- kernel.c: 一个小的内核程序，最终输出一个字符串
- createimage.c：生成内核镜像的工具，初始提供一个框架，本次任务需要完成
- createimage: **本次实验不允许使用该文件**
- Makefile: 编译配置文件，**将createimage的编译选项去掉注释(第12~13行)**
- ld.script: 链接器脚本文件

2.2. 获取：

课程网站。

2.3. 运行

createimage 为提供的可执行文件，当 createimage.c 实现完成后，将 Makefile 中的 createimage 项去掉注释。

make 命令编译文件

make clean 对编译产生的文件进行清除

sudo dd if=image of=/dev/sdb 将产生的 image 写进 SD 卡中

在 minicom 中执行 *loadboot* 运行程序

3. 任务

3.1. 设计和评审

帮助学生发现设计的错误，及时完成任务。学生需要对这次的作业进行全面考虑，在实现代码之前有清晰的思路。学生讲解设计思路时可以用不同的形式，如伪代码、流程图等，每个组使用 PPT 的形式呈现（不要超过十分钟）。

设计介绍

- 创造内核镜像：可执行文件(ELF)的特点？怎样读取可执行文件？可执行文件bootblock和kernel在内核镜像的什么位置存放？

3.2. createimage 开发

3.2.1.要求

实现一个 Linux 工具，将 bootblock 和 kernel 结合为一个操作系统镜像，并

提供操作系统镜像的一些信息。其中 bootblock 存放在镜像的第一个扇区，kernel 存放在镜像的第二个扇区。一共需要实现以下函数：

- read_exec_file(); 读取ELF格式的一个文件。
- write_bootblock();将可执行文件bootblock写入内核镜像”image”文件中。
- write_kernel(); 将可执行文件kernel写入镜像文件”image”文件中。
- count_kernel_sectors();计算kernel有多少个扇区
- record_kernel_sectors();将kernel的扇区个数写入bootblock的os_size位置处。
- extend_opt();打印出—extend选项要打印出来的信息。

3.2.2.注意事项

为了实现一个 createimage，需要了解 ELF 文件格式（课程网站上提供 ELF 文件格式介绍）：

- ELF文件头，以及它的e_phnum和e_phoff域。
- 可执行程序文件头，以及p_offset和p_filesz域。

--extend 选项用来提供内核镜像的一些信息。包括内核的大小，可执行文件在磁盘上存放的扇区以及在磁盘上写的大小等信息。

想要获得更多信息，可以通过执行指令 *man –M man createimage* 获得更多信息，此处忽略”-vm”信息。

4.测试

createimage.c 可以正确的将 bootblock 写入硬盘的第一扇区，将 kernel 从第二扇区开始写入。当代码正确时，会打印出*“It’s a kernel!”*等字样。

5.参考资料

- ELF文件格式文档：课程网站