[2][1][3]

西南林业大学

本科毕业（设计）论文

(二〇一八届)

题　　目： RongOS — 一个简单操作系统的实现

分院系部： 计算机与信息科学学院

专　　业： 计算机科学与技术专业

姓　　名： 蒲启元

导师姓名： 王晓林

导师职称： 讲师

二〇一八 年 六 月

RongOS — 一个简单操作系统的实现

蒲启元

（西南林业大学 计算机与信息科学学院，云南 昆明 650224）

摘　要： 操作系统管理着计算机的硬件和软件资源，它是向上层应用软件提供服务（接口）的核心系统软件，这些服务包括进程管理，内存管理，文件系统，网络通信，安全机制等。操作系统的设计与实现则是软件工业的基础与内核。为此，在国务院提出的《中国制造 2025》中专门强调了操作系统的开发。但长期以来，操作系统核心开发技术都掌握在外国人手中，技术受制，对于我们的软件工业来说很不利。本文拟从零开始设计开发一个简单的操作系统，包括 boot loader，中断，内存管理，图形接口，多任务，以及在这个系统上的几个小应用等。尽管这个系统很简单，但它为自主开发操作系统做了一个小小的尝试。

关键词： 操作系统，开发，自主

The implement of a simple OS — RongOS

Qiyuan Pu

School of Computer and Information Science
Southwest Forestry University
Kunming 650224, Yunnan, China

Abstract: Operating system manages the sources of hardware and software, it lie in the core of the system software and provide service(interface) to upper application. These service including process management, memory management, file system, network communication, security mechanism etc. The design and implement of operating system is the foundation and core of software industry. Therefore, «Made in China 2025» emphasize the development of operating system that put forward by The State Council. For a long time, however, the kernel development technology grasped in the hand of foreigner, it's bad for our software industry cause of limited technology. So this article will design and develop a simple operating system, including boot loader, interrupt, memory management, graphic interface, multitasking, and some little application depend on this system. In spite of the simple of this system, it's a small trying for autonomous development operating system.

Key words: operating system, development, autonomous

目　录

插图目录

表格目录

1 Introduction

1.1 Background

1.2 Preliminary Works

1.2.1 Development Environment

Operating System: Debian 4.11.0-1-amd64

Debug System: QEMU emulator version 2.8.1(Debian 1:2.8+dfsg-7)

Emacs version: GNU Emacs 25.2.2

1.2.2 Tools

Some tools used to develop RongOS, see tools.[1].

1.2.3 Install

Debian System: there is a small tutorial.[2]

QEMU, for my x86_64 architecture:

```
$ sudo apt-get install qemu-system-x86_64
```

Note that the tools is exe formate, so on Debian system, you need to install wine:

```
$ sudo apt-get update
$ sudo apt-get install wine
```

Maybe you also need to add i386 architecture cause of AMD64 on your machine to use these tools:

```
$ sudo dpkg --add-architecture i386
```

[1] https://github.com/Puqiyuan/RongOS/tree/master/Tools
[2] http://cs2.swfc.edu.cn/~wx672/lecture_notes/linux/install.html

```
$ sudo apt-get update
```

2 Design

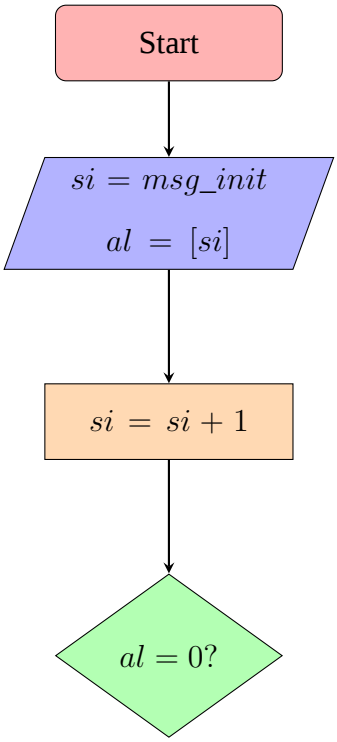
2.1 Top Level Design

2.2 Detailed Design

2.2.1 Boot Loader

Description

The Structure of Boot Loader Program



3 Implementation

3.1 Boot Loader

3.1.1 Chose Disk

There are many ways to boot a operating system, from hard disk, USB, floppy disk etc. I chose floppy disk, although it is out of date. For my purpose is that develop a simple operating system, pay my attention on how to development. The structure of floppy disk is simple and for my simple operating system it's enough.

3.1.2 The Structure of Floppy Disk

This picture show the inside of floppy disk:

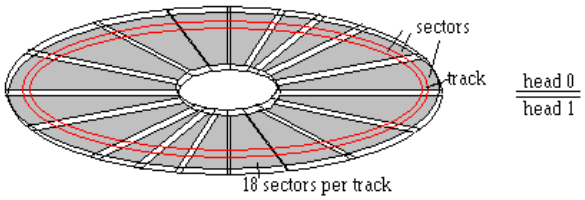


图 3-1 Floppy Disk Structure

The floppy store information in two sides. There are 80 cylinders from the outermost to the core in each side, numbering 0, 1, ..., 79. The head can assign be 0 or 1, representing two sides of floppy. When specify head number and cylinder number, forming a ring, named track in jargon. The track is large so we divide it to 18 small parts, named sector. A sector can store 512 byte. So the capacity of a floppy is:

$$18 * 80 * 2 * 512 = 1474560Byte = 1440KB.$$

The IPL(Initial Program Loader) in C0-H0-S1(cylinder 0, head 0, sector 2), and the next sector is C0-H0-S2.

3.1.3 The Implementation of Boot Loader

```
1         CYLS equ 10
2
3  org 0x7c00 ; load the program to address 0x7c00.
4         jmp entry
5         ; The next codes specify the format of standard FAT12 floppy disk.
6  db 0x90 ;db is the abbreation of "define byte", it literally places that byte
7         ; right there in the executable.
8  db "RONGBOOT" ;The name of boot sector, must be 8 byte.
9  dw 512 ; the size of every sector, must be 512 byte.
10 db 1 ; the size of cluster, must be 1.
11 dw 1 ; the start point of FAT, 1 general case.
12 db 2 ; the number of FAT, must be 2.
13 dw 224 ; the size of root directory, 224 in general.
14 dw 2880 ; the size of this floppy disk, must be 2880.
15 db 0xf0 ; the kind of disk.
16 dw 9 ; the length of FAT.
17 dw 18 ; how many sectors in one track, must be 18.
18 dw 2 ; the number of head, must be 2.
19 dd 0 ; no partion, must be 0.
20 dd 2880 ; the size if re-writer one time.
21 db 0,0,0x29 ; just fixed, no meaning.
22 dd 0xffffffff
23 db "RONGBOOTOS " ; the name of disk.
24 db "FAT12   " ; the name of disk formate.
25 resb 18 ; reserved 18 byte.
26        ; end FAT12 formate.
27
28 entry:
29        mov ax, 0 ; init the registers.
30        mov ss, ax
```

```
31          mov sp, 0x7c00
32          mov ds, ax
33
34          mov si, msg_init
35          jmp init
36
37
38  init:
39          mov al, [si]
40          add si, 1 ; increment by 1.
41          cmp al, 0
42          je load ; if al == 0, jmp to load, the msg_init info displayed.
43  ; the lastest character is null character, coding in 0.
44
45          mov ah, 0x0e ; write a character in TTY mode.
46          mov bx, 15   ; specify the color of the character.
47          int 0x10 ; call BIOS function, video card is number 10.
48          jmp init
49
50
51  msg_init:
52  db 0x0a ; new line
53  db 0x0d
54  db "Copyright: GPL"
55  db 0x0a
56  db 0x0d
57  db "Author: Qiyuan Pu"
58  db 0x0a
59  db 0x0d
60  db "https://github.com/Puqiyuan/RongOS"
61  db 0x0a
```

```
62  db 0x0d
63  db "IPL is loading, please waiting..."
64  db 0x0a
65  db 0x0d
66  db "......"
67
68
69  load:
70
71          mov ax, 0
72
73          mov ax, 0x0820 ; load these sectors to memory begin with 0x0820.
74          mov es, ax
75          mov ch, 0 ; cylinder 0.
76          mov dh, 0 ; magnetict head 0.
77          mov cl, 2 ; sector 2.
78
79  readloop:
80          mov si, 0 ; si register is a counter, try read a sector
81          ; five times.
82
83  retry:
84          mov ah, 0x02 ; parameter 0x02 to ah, read disk.
85          mov al, 1 ; parameter 1 to al, read disk.
86          mov bx, 0
87          mov dl, 0x00 ; the number of driver number.
88          int 0x13 ; after prepared parameters, call 0x13 interrupted.
89
90          jnc next ; if no carry read next sector.
91          add si, 1 ; tring again read sector, counter add 1.
92          cmp si, 5 ; until five times
```

```
93          jae error ; if tring times large than five, failed.
94
95          ; reset the status of floppy and read again.
96          mov ah, 0x00
97          mov dl, 0x00
98          int 0x13
99          jmp retry
100
101 next:
102         mov ax, es
103         add ax, 0x0020
104         mov es, ax
105         add cl, 1
106         cmp cl, 18
107         jbe readloop
108         mov cl, 1
109         add dh, 1
110         cmp dh, 2
111         jb readloop
112         mov dh, 0
113         add ch, 1
114         cmp ch, CYLS
115         jb readloop
116         jmp correct
117
118
119 fin:
120         hlt
121         jmp fin
122
123
```

```
124 error:
125         mov si, msg
126
127
128 correct:
129         mov si, msg_corr
130
131
132 putloop:
133         mov al, [si]
134         add si, 1
135         cmp al, 0
136         mov [0x0ff0], ch
137         je 0xc200
138         mov ah, 0x0e
139         mov bx, 15
140         int 0x10
141         jmp putloop
142
143
144 msg_corr:
145 db 0x0a
146 db 0x0d
147 db 0x0a
148 db 0x0d
149 db "OK: IPL loaded"
150 db 0x0a
151 db 0x0d
152 db 0
153
154
```

```
155 msg:
156 db 0x0a
157 db "IPL load error"
158 db 0x0a
159 db 0
160 resb 0x7dfe-$
161
162
163 db 0x55, 0xaa ; the sector end with 0x55 0xaa, the sector is
164         ;boot sector.
```

3.2 32-bit Mode and Import C Codes

3.3 Screen Display and Text

3.4 Control Mouse

3.5 Memory Management

3.6 Making Window

3.7 Timer

3.8 Multitasking

3.9 Command Line Window

3.10 API

3.11 OS Protection

3.12 Graphics Processing

3.13 Window Operation

3.14 Application Protection

3.15 File Operation

3.16 Some Applications

3.17 Prospects and Shortages

参考文献

[1] 国务院。*中国制造 2025*，2015-05。。

[2] WiKipedia. *Operating System*, 2017-08..

[3] 川合秀实. *30 天自制操作系统*. 人民邮电出版社, 2012-08.

指导教师简介

王晓林，男，49 岁，硕士，讲师，毕业于英国格林尼治大学，分布式系统专业，现任西南林业大学计信学院教师，执教 Linux、操作系统、网络技术等方面的课程，有丰富的 Linux 教学和系统管理经验。

致　谢

首先我想感谢我的老师，王晓林。大学期间，他给了我很多指导，包括专业方面和上大学的意义等。很多时候，他对学生的要求看起来都是不近情理的，但正是通过这个“痛苦”的过程，我锻炼了坚强的意志，和战胜困难的信心。谢谢你，王老师。我最想感谢的是我的女友，她容忍我在完成这个设计时的很多个夜晚不陪她，给我支持，鼓励我，不抱怨。所以我愿意把这个简单操作系统命名为 RongOS, 蓉便是她名字的最后一个字。谢谢你，我最亲爱的。