

## <<30天自制操作系统>>

### 图书基本信息

书名：<<30天自制操作系统>>

13位ISBN编号：9787115287960

10位ISBN编号：7115287961

出版时间：2012-8

出版单位：人民邮电出版社

作者：[日]川合秀实

页数：710

字数：1063000

译者：周自恒,李黎明,曾祥江,张文旭

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## <<30天自制操作系统>>

### 前言

“好想编写一个操作系统呀！”

“笔者的朋友曾说这是所有程序员都曾经怀揣的一个梦想。

”所有的程序员”可能有点夸张了，不过作为程序员的梦想，它至少也应该能排进前十名吧。

也许很多人觉得编写操作系统是个天方夜谭，这一定是操作系统业界的一个阴谋（笑）。他们故意让大家相信编写操作系统是一件非常困难的事情，这样就可以高价兜售自己开发的操作系统，而且操作系统的作者还会被顶礼膜拜。

那么实际情况又怎么样呢？

和别的程序相比，其实编写操作系统并没有那么难，至少笔者的感觉是这样。

在各位读者之中，也许有人曾经挑战过操作系统的编写，但因为太难而放弃了。

拥有这样经历的人也许不会认同笔者的观点。

其实你错了，你的失败并不是因为编写操作系统太难，而是因为没有人告诉你那其实是一件很简单的事而已。

不仅是编写操作系统，任何事都是一样的。

如果讲解的人认为它很难，那就不可能把它讲述得通俗易懂，即便是同样的内容，也会讲得无比复杂。这样的讲解，肯定是很难懂的。

那么，你想不想和笔者一起再挑战一次呢？

如果你曾经梦想过编写自己的操作系统，一定会觉得乐在其中的。

可能有人会说，这本书足足有700多页，怎么会“有趣”和“简单”呢？

唔，这么一说笔者也觉得挺心虚的，不过其实也只是长了那么一点点啦。

平均下来的话，每天只有大约23页的内容，你看，也没有那么长吧？

这本书的文风非常轻松，也许你不知不觉中就会读得很快。

但是这样的话可能印象不会很深，最好还是能静下心来慢慢地读。

书中所展示的程序代码和文字的说明同样重要，因此也希望大家仔细阅读。

只要注意这些，理解本书的内容就应该没有问题了。

在本书中，我们使用C语言和汇编语言来编写操作系统，不过不必担心，你可以在阅读本书的同时来逐步学习关于这些编程语言的知识。

本书在这方面写得非常仔细，如果能有人通过本书终于把C语言中的指针给搞懂了，那笔者的目的也就达到了。

即便是从这样的水平开始，30天后你也能够编写出一个很棒的操作系统，请大家拭目以待吧！

## <<30天自制操作系统>>

### 内容概要

自己编写一个操作系统，是许多程序员的梦想。也许有人曾经挑战过，但因为太难而放弃了。其实你错了，你的失败并不是因为编写操作系统太难，而是因为没有人告诉你那其实是一件很简单的事。那么，你想不想再挑战一次呢？

这是一本兼具趣味性、实用性与学习性的书籍。作者从计算机的构造、汇编语言、C语言开始解说，让你在实践中掌握算法。在这本书的指导下，从零编写所有代码，30天后就可以制作出一个具有窗口系统的32位多任务操作系统。

本书以课题为主导，边做边玩，抛开晦涩难懂的语言，行文风格十分随性，还充满了各种欢乐的吐槽，适合操作系统爱好者和程序设计人员阅读。

## <<30天自制操作系统>>

### 作者简介

川合秀实 ( Hidemi Kawai )

生于1975年，是一位以“轻量化”编程思想见长的“非主流”开发者。

2000年因自行开发的OSASK项目而名声大噪。

OSASK是一个开源的32位微型操作系统，它并非以Linux等内核为基础，而是完全从零开始开发，在一张软盘的容量下实现了GUI、多任务、多语言等高级特性，启动时间只需1秒。

本书的内容可以看成是作者以OSASK为蓝本，教会读者从零开始开发一个操作系统，同时可以让初学者在编写操作系统的过程中，了解操作系统背后更多的知识。

## <<30天自制操作系统>>

### 书籍目录

#### 第0天 着手开发之前

- 1 前言
- 2 何谓操作系统
- 3 开发操作系统的各种方法
- 4 无知则无畏
- 5 如何开发操作系统
- 6 操作系统开发中的困难
- 7 学习本书时的注意事项（重要！）
- 8 各章内容摘要

#### 第1天 从计算机结构到汇编程序入门

- 1 先动手操作
- 2 究竟做了些什么
- 3 初次体验汇编程序
- 4 加工润色

#### 第2天 汇编语言学习与Makefile入门

- 1 介绍文本编辑器
- 2 继续开发
- 3 先制作启动区
- 4 Makefile入门

#### 第3天 进入32位模式并导入C语言

- 1 制作真正的IPL
- 2 试错
- 3 读到18扇区
- 4 读入10个柱面
- 5 着手开发操作系统
- 6 从启动区执行操作系统
- 7 确认操作系统的执行情况
- 8 32位模式前期准备
- 9 开始导入C语言
- 10 实现HLT (harib00j)

#### 第4天 C语言与画面显示的练习

- 1 用C语言实现内存写入 (harib01a)
- 2 条纹图案 (harib01b)
- 3 挑战指针 (harib01c)
- 4 指针的应用 (1) (harib01d)
- 5 指针的应用 (2) (harib01e)
- 6 色号设定 (harib01f)
- 7 绘制矩形 (harib01g)
- 8 今天的成果 (harib01h)

#### 第5天 结构体、文字显示与GDT/IDT初始化

- 1 接收启动信息 (harib02a)
- 2 试用结构体 (harib02b)
- 3 试用箭头记号 (harib02c)
- 4 显示字符 (harib02d)

## <<30天自制操作系统>>

- 5 增加字体 (harib02e)
- 6 显示字符串 (harib02f)
- 7 显示变量值 (harib02g)
- 8 显示鼠标指针 (harib02h)
- 9 GDT与IDT的初始化 (harib02i)
- 第6天 分割编译与中断处理
  - 1 分割源文件 (harib03a)
  - 2 整理Makefile (harib03b)
  - 3 整理头文件 (harib03c)
  - 4 意犹未尽
  - 5 初始化PIC (harib03d)
  - 6 中断处理程序的制作 (harib03e)
- 第7天 FIFO与鼠标控制
  - 1 获取按键编码 (harib04a)
  - 2 加快中断处理 (harib04b)
  - 3 制作FIFO缓冲区 (harib04c)
  - 4 改善FIFO缓冲区 (harib04d)
  - 5 整理FIFO缓冲区 (harib04e)
  - 6 总算讲到鼠标了 (harib04f)
  - 7 从鼠标接受数据 (harib04g)
- 第8天 鼠标控制与32位模式切换
  - 1 鼠标解读 (1) (harib05a)
  - 2 稍事整理 (harib05b)
  - 3 鼠标解读 (2) (harib05c)
  - 4 移动鼠标指针 (harib05d)
  - 5 通往32位模式之路
- 第9天 内存管理
  - 1 整理源文件 (harib06a)
  - 2 内存容量检查 (1) (harib06b)
  - 3 内存容量检查 (2) (harib06c)
  - 4 挑战内存管理 (harib06d)
- 第10天 叠加处理
  - 1 内存管理 (续) (harib07a)
  - 2 叠加处理 (harib07b)
  - 3 提高叠加处理速度 (1) (harib07c)
  - 4 提高叠加处理速度 (2) (harib07d)
- 第11天 制作窗口
  - 1 鼠标显示问题 (harib08a)
  - 2 实现画面外的支持 (harib08b)
  - 3 shtctl的指定省略 (harib08c)
  - 4 显示窗口 (harib08d)
  - 5 小实验 (harib08e)
  - 6 高速计数器 (harib08f)
  - 7 消除闪烁 (1) (harib08g)
  - 8 消除闪烁 (2) (harib08h)
- 第12天 定时器 (1)
  - 1 使用定时器 (harib09a)

## &lt;&lt;30天自制操作系统&gt;&gt;

- 2 计量时间 (harib09b)
- 3 超时功能 (harib09c)
- 4 设定多个定时器 (harib09d)
- 5 加快中断处理 (1) (harib09e)
- 6 加快中断处理 (2) (harib09f)
- 7 加快中断处理 (3) (harib09g)
- 第13天 定时器 (2)
  - 1 简化字符串显示 (harib10a)
  - 2 重新调整FIFO缓冲区 (1) (harib10b)
  - 3 测试性能 (harib10c ~ harib10f)
  - 4 重新调整FIFO缓冲区 (2) (harib10g)
  - 5 加快中断处理 (4) (harib10h)
  - 6 使用“哨兵”简化程序 (harib10i)
- 第14天 高分辨率及键盘输入
  - 1 继续测试性能 (harib11a ~ harib11c)
  - 2 提高分辨率 (1) (harib11d)
  - 3 提高分辨率 (2) (harib11e)
  - 4 键盘输入 (1) (harib11f)
  - 5 键盘输入 (2) (harib11g)
  - 6 追记内容 (1) (harib11h)
  - 7 追记内容 (2) (harib11i)
- 第15天 多任务 (1)
  - 1 挑战任务切换 (harib12a)
  - 2 任务切换进阶 (harib12b)
  - 3 做个简单的多任务 (1) (harib12c)
  - 4 做个简单的多任务 (2) (harib12d)
  - 5 提高运行速度 (harib12e)
  - 6 测试运行速度 (harib12f)
  - 7 多任务进阶 (harib12g)
- 第16天 多任务 (2)
  - 1 任务管理自动化 (harib13a)
  - 2 让任务休眠 (harib13b)
  - 3 增加窗口数量 (harib13c)
  - 4 设定任务优先级 (1) (harib13d)
  - 5 设定任务优先级 (2) (harib13e)
- 第17天 命令行窗口
  - 1 闲置任务 (harib14a)
  - 2 创建命令行窗口 (harib14b)
  - 3 切换输入窗口 (harib14c)
  - 4 实现字符输入 (harib14d)
  - 5 符号的输入 (harib14e)
  - 6 大写字母与小写字母 (harib14f)
  - 7 对各种锁定键的支持 (harib14g)
- 第18天 dir命令
  - 1 控制光标闪烁 (1) (harib15a)
  - 2 控制光标闪烁 (2) (harib15b)
  - 3 对回车键的支持 (harib15c)

## &lt;&lt;30天自制操作系统&gt;&gt;

4 对窗口滚动的支持 (harib15d)

5 mem命令 (harib15e)

6 cls命令 (harib15f)

7 dir命令 (harib15g)

第19天 应用程序

1 type命令 (harib16a)

2 type命令改良 (harib16b)

3 对FAT的支持 (harib16c)

4 代码整理 (harib16d)

5 第一个应用程序 (harib16e)

第20天 API

1 程序整理 (harib17a)

2 显示单个字符的API (1) (harib17b)

3 显示单个字符的API (2) (harib17c)

4 结束应用程序 (harib17d)

5 不随操作系统版本而改变的API (harib17e)

6 为应用程序自由命名 (harib17f)

7 当心寄存器 (harib17g)

8 用API显示字符串 (harib17h)

第21天 保护操作系统

1 攻克难题——字符串显示API (harib18a)

2 用C语言编写应用程序 (harib18b)

3 保护操作系统 (1) (harib18c)

4 保护操作系统 (2) (harib18d)

5 对异常的支持 (harib18e)

6 保护操作系统 (3) (harib18f)

7 保护操作系统 (4) (harib18g)

第22天 用C语言编写应用程序

1 保护操作系统 (5) (harib19a)

2 帮助发现bug (harib19b)

3 强制结束应用程序 (harib19c)

4 用C语言显示字符串 (1) (harib19d)

5 用C语言显示字符串 (2) (harib19e)

6 显示窗口 (harib19f)

7 在窗口中描绘字符和方块 (harib19g)

第23天 图形处理相关

1 编写malloc (harib20a)

2 画点 (harib20b)

3 刷新窗口 (harib20c)

4 画直线 (harib20d)

5 关闭窗口 (harib20e)

6 键盘输入API (harib20f)

7 用键盘输入来消遣一下 (harib20g)

8 强制结束并关闭窗口 (harib20h)

第24天 窗口操作

1 窗口切换 (1) (harib21a)

2 窗口切换 (2) (harib21b)



## &lt;&lt;30天自制操作系统&gt;&gt;

- 3 移动窗口 ( harib21c )
  - 4 用鼠标关闭窗口 ( harib21d )
  - 5 将输入切换到应用程序窗口 ( harib21e )
  - 6 用鼠标切换输入窗口 ( harib21f )
  - 7 定时器API ( harib21g )
  - 8 取消定时器 ( harib21h )
- 第25天 增加命令行窗口
- 1 蜂鸣器发声 ( harib22a )
  - 2 增加更多的颜色 ( 1 ) ( harib22b )
  - 3 增加更多的颜色 ( 2 ) ( harib22c )
  - 4 窗口初始位置 ( harib22d )
  - 5 增加命令行窗口 ( 1 ) ( harib22e )
  - 6 增加命令行窗口 ( 2 ) ( harib22f )
  - 7 增加命令行窗口 ( 3 ) ( harib22g )
  - 8 增加命令行窗口 ( 4 ) ( harib22h )
  - 9 变得更像真正的操作系统 ( 1 ) ( harib22i )
  - 10 变得更像真正的操作系统 ( 2 ) ( harib22j )
- 第26天 为窗口移动提速
- 1 提高窗口移动速度 ( 1 ) ( harib23a )
  - 2 提高窗口移动速度 ( 2 ) ( harib23b )
  - 3 提高窗口移动速度 ( 3 ) ( harib23c )
  - 4 提高窗口移动速度 ( 4 ) ( harib23d )
  - 5 启动时只打开一个命令行窗口 ( harib23e )
  - 6 增加更多的命令行窗口 ( harib23f )
  - 7 关闭命令行窗口 ( 1 ) ( harib23g )
  - 8 关闭命令行窗口 ( 2 ) ( harib23h )
  - 9 start命令 ( harib23i )
  - 10 ncst命令 ( harib23j )
- 第27天 LDT与库
- 1 先来修复bug ( harib24a )
  - 2 应用程序运行时关闭命令行窗口 ( harib24b )
  - 3 保护应用程序 ( 1 ) ( harib24c )
  - 4 保护应用程序 ( 2 ) ( harib24d )
  - 5 优化应用程序的大小 ( harib24e )
  - 6 库 ( harib24f )
  - 7 整理make环境 ( harib24g )
- 第28天 文件操作与文字显示
- 1 alloca ( 1 ) ( harib25a )
  - 2 alloca ( 2 ) ( harib25b )
  - 3 文件操作API ( harib25c )
  - 4 命令行API ( harib25d )
  - 5 日文文字显示 ( 1 ) ( harib25e )
  - 6 日文文字显示 ( 2 ) ( harib25f )
  - 7 日文文字显示 ( 3 ) ( harib25g )
- 第29天 压缩与简单的应用程序
- 1 修复bug ( harib26a )
  - 2 文件压缩 ( harib26b )

## <<30天自制操作系统>>

- 3 标准函数
- 4 非矩形窗口 ( harib26c )
- 5 bball ( harib26d )
- 6 外星人游戏 ( harib26e )
- 第30天 高级的应用程序
  - 1 命令行计算器 ( harib27a )
  - 2 文本阅读器 ( harib27b )
  - 3 MML播放器 ( harib27c )
  - 4 图片阅读器 ( harib27d )
  - 5 IPL的改良 ( harib27e )
  - 6 光盘启动 ( harib27f )
- 第31天 写在开发完成之后
  - 1 继续开发要靠大家的努力
  - 2 关于操作系统的大小
  - 3 操作系统开发的诀窍
  - 4 分享给他人使用
  - 5 关于光盘中的软件
  - 6 关于开源的建议
  - 7 后记
  - 8 毕业典礼
  - 9 附录

## &lt;&lt;30天自制操作系统&gt;&gt;

## 章节摘录

版权页：插图：现在我们要进行内存管理了。

首先必须要做的事情，是搞清楚内存究竟有多大，范围是到哪里。

如果连这一点都搞不清楚的话，内存管理就无从谈起。

在最初启动时，BIOS肯定要检查内存容量，所以只要我们问一问BIOS，就能知道内存容量有多大。

但问题是，如果那样做的话，一方面asmhead.nas会变长，另一方面，BIOS版本不同，BIOS函数的调用方法也不相同，麻烦事太多了。

所以，笔者想与其如此，不如自己去检查内存。

下面介绍一下做法。

首先，暂时让486以后的CPU的高速缓存（cache）功能无效。

回忆一下最初讲的CPU与内存的关系吧。

我们说过，内存与CPU的距离地与CPU内部元件要远得多，因此在寄存器内部MOV，要比从寄存器MOV到内存快得多。

但另一方面，有一个问题，CPU的记忆力太差了，即使知道内存的速度不行，还不得不频繁使用内存

。考虑到这个问题，英特尔的大叔们在CPU里也加进了一点存储器，它被称为高速缓冲存储器（cache memory）。

cache这个词原是指储存粮食弹药等物资的仓库。

但是能够跟得上CPU速度的高速存储器价格特别高，一个芯片就有一个CPU那么贵。

如果128MB内存全部都用这种高价存储器，预算上肯定受不了。

高速缓存，容量只有这个数值的千分之一，也就是128KB左右。

高级CPU，也许能有1MB高速缓存，但即便这样，也不过就是128MB的分之一。

为了有效使用如此稀有的高速缓存，英特尔的大叔们决定，每次访问内存，都要将所访问的地址和内容存入到高速缓存里。

也就是存放成这样：18号地址的值是54。

如果下次再要用18号地址的内容，CPU就不再读内存了，而是使用高速缓存的信息，马上就能回答出18号地址的内容是54。

往内存里写入数据时也一样，首先更新高速缓存的信息，然后再写入内存。

如果先写入内存的话，在等待写入完成的期间，CPU处于空闲状态，这样就会影响速度。

所以，先更新缓存，缓存控制电路配合内存的速度，然后再慢慢发送内存写入命令。

## <<30天自制操作系统>>

### 编辑推荐

39.1K迷你系统，实现多任务、汉字显示、文件压缩，还能听歌看图玩游戏。

拥有《30天自制操作系统》一书，只需30天，从零开始编写一个五脏俱全的图形操作系统将不再只是个梦想。

日本编程天才川合秀实，将通过本书，揭开CPU、内存、磁盘以及操作系统底层工作模式的神秘面纱。

## <<30天自制操作系统>>

### 版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>