

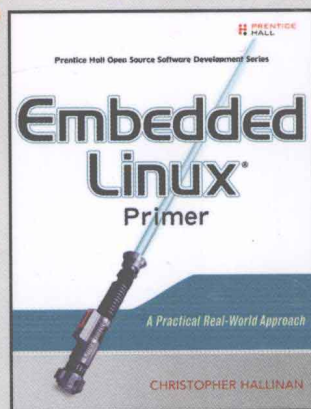
PRENTICE
HALL

典藏
原版书苑

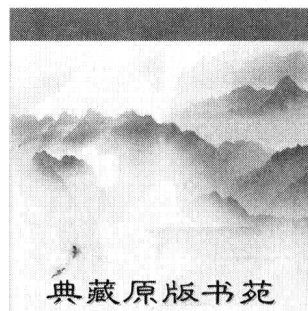
Embedded
Linux®
Primer

[美] Christopher Hallinan 著

嵌入式 Linux 开发 (英文版)



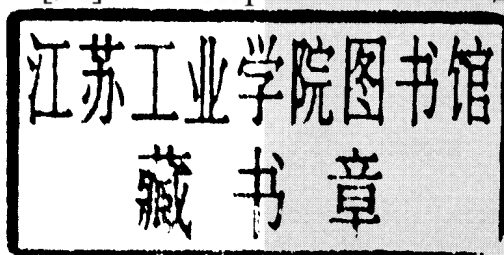
人民邮电出版社
POSTS & TELECOM PRESS



嵌入式 Linux 开发

(英文版)

[美] Christopher Hallinan 著



图书在版编目 (CIP) 数据

嵌入式 Linux 开发: 英文 / (美) 哈利南 (Hallinan, C.)
著. —北京: 人民邮电出版社, 2008.2
(典藏原版书苑)
ISBN 978-7-115-17225-9

I. 嵌… II. 哈… III. Linux 操作系统—程序设计—英文
IV. TP316.89

中国版本图书馆 CIP 数据核字 (2007) 第 179494 号

版 权 声 明

Original edition, entitled Embedded Linux Primer, 0131679848 by Christopher Hallinan, published by Pearson Education, Inc, publishing as Prentice Hall, Copyright © 2007 by Pearson Education, Inc.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc

China edition published by PEARSON EDUCATION ASIA LTD., and POSTS & TELECOMMUNICATIONS PRESS Copyright © 2007.

This edition is manufactured in the People's Republic of China, and is authorized for sale only in People's Republic of China excluding Hong Kong, Macau and Taiwan.

仅限于中华人民共和国境内 (不包括中国香港、澳门特别行政区和中国台湾地区) 销售。

本书封面贴有 Pearson Education (培生教育出版集团) 激光防伪标签。无标签者不得销售。

典藏原版书苑

嵌入式 Linux 开发 (英文版)

-
- ◆ 著 [美] Christopher Hallinan
责任编辑 付 飞
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
三河市海波印务有限公司印刷
新华书店总店北京发行所经销
 - ◆ 开本: 800×1000 1/16
印张: 35 25
字数: 748 千字 2008 年 2 月第 1 版
印数: 1—3 000 册 2008 年 2 月河北第 1 次印刷

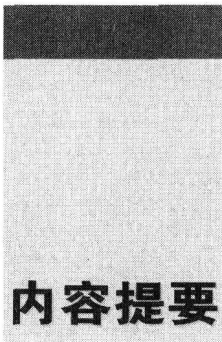
著作权合同登记号 图字: 01-2007-5296 号

ISBN 978-7-115-17225-9/TP

定价: 88.00 元

读者服务热线: (010) 67132705 印装质量热线: (010) 67129223

反盗版热线: (010) 67171154



内容提要

本书是一部全面易懂的嵌入式 Linux 开发实用指南。主要内容包括：处理器基础、Linux 内核、内核初始化、系统初始化、Bootloader、设备驱动基础、文件系统、MTD 子系统、BusyBox、嵌入式开发环境、开发工具、内核调试技术、调试嵌入式 Linux 应用程序、Linux 与实时等。

本书适合有一定的 C 语言编程基础，对本地局域网和互连网有基本了解的嵌入式开发人员和工程师阅读。

About the Author

Christopher Hallinan is currently field applications engineer for Monta Vista Software, living and working in Massachusetts. Chris has spent more than 25 years in the networking and communications marketplace mostly in various product development roles, where he developed a strong background in the space where hardware meets software. Prior to joining Monta Vista, Chris spent four years as an independent Linux consultant providing custom Linux board ports, device drivers, and bootloaders. Chris's introduction to the open source community was through contributions to the popular U-Boot bootloader. When not messing about with Linux, he is often found singing and playing a Taylor or Martin.

*To my mother Edythe, whose courage, confidence, and grace
have been an inspiration to us all.*

Acknowledgments

I am constantly amazed by the graciousness of open source developers. I am humbled by the talent in our community that often far exceeds my own. During the course of this project, I reached out to many people in the Linux and open source community with questions. Most often my questions were quickly answered and with encouragement. In no particular order, I'd like to express my gratitude to the following members of the Linux and open source community who have contributed answers to my questions:

Dan Malek provided inspiration for some of the contents of Chapter 2, "Your First Embedded Experience."

Dan Kegel and Daniel Jacobowitz patiently answered my toolchain questions.

Scott Anderson provided the original ideas for the `gdb` macros presented in Chapter 14, "Kernel Debugging Techniques."

Brad Dixon continues to challenge and expand my technical vision through his own.

George Davis answered my ARM questions.

Jim Lewis provided comments and suggestions on the MTD coverage.

Cal Erickson answered my `gdb` use questions.

John Twomey advised on Chapter 3, "Processor Basics."

Lee Revell, Sven-Thorsten Dietrich, and Daniel Walker advised on real time Linux content. Klaas van Gend provided excellent feedback and ideas for my development tools and debugging content.

Many thanks to AMCC, Embedded Planet, Ultimate Solutions, and United Electronic Industries for providing hardware for the examples. Many thanks to my employer, Monta Vista Software, for tolerating the occasional distraction and for providing software for some of the examples. Many others contributed ideas, encouragement, and support over the course of the project. To them I am also grateful.

I wish to acknowledge my sincere appreciation to my primary review team, who promptly read each chapter and provided excellent feedback, comments, and ideas. Thank you Arnold Robbins, Sandy Terrace, Kurt Lloyd, and Rob Farber. Thanks also to David Brief, who reviewed the proposal and provided valuable input on the book's organization. Many thanks to Arnold for helping this newbie learn the ropes of writing a technical book. While every attempt has been made to eliminate mistakes, those that remain are solely my own.

I want to thank Mark L. Taub for bringing this project to fruition and for his encouragement and infinite patience! I wish to thank the production team including Kristy Hart, Jennifer Cramer, Krista Hansing, and Cheryl Lenser.

And finally, a very special and heartfelt thank you to Cary Dillman who read each chapter as it was written, and for her constant encouragement and her occasional sacrifice throughout the project.

—Chris Hallinan

Foreword

Computers are everywhere.

This fact, of course, is not a surprise to anyone who hasn't been living in a cave during the past 25 years or so. And you probably know that computers aren't just on our desktops, in our kitchens, and, increasingly, in our living rooms holding our music collections. They're also in our microwave ovens, our regular ovens, our cell-phones, and our portable digital music players.

And if you're holding this book, you probably know a lot, or are interested in learning more about, these embedded computer systems.

Until not too long ago, embedded systems were not very powerful, and they ran special-purpose, proprietary operating systems that were very different from industry-standard ones. (Plus, they were much harder to develop for.) Today, embedded computers are as powerful as, if not more than, a modern home computer. (Consider the high-end gaming consoles, for example.)

Along with this power comes the capability to run a full-fledged operating system such as Linux. Using a system such as Linux for an embedded product makes a lot of sense. A large community of developers are making it possible. The development environment and the deployment environment can be surprisingly similar, which makes your life as a developer much easier. And you have both the security of a protected address space that a virtual memory-based system gives you, and the power and flexibility of a multiuser, multiprocess system. That's a good deal all around.

For this reason, companies all over the world are using Linux on many devices such as PDAs, home entertainment systems, and even, believe it or not, cellphones!

I'm excited about this book. It provides an excellent “guide up the learning curve” for the developer who wants to use Linux for his or her embedded system. It's clear, well-written, and well-organized; Chris's knowledge and understanding show through at every turn. It's not only informative and helpful—it's also enjoyable to read.

I hope you both learn something and have fun at the same time. I know I did.

Arnold Robbins

Series Editor

前言

虽然有很多好作品讲述 Linux，但是没有一本书能为嵌入式 Linux 开发人员提供如此广泛的信息和建议。当然，有一些非常优秀的书籍介绍了 Linux 内核和 Linux 系统管理方面的知识，本书也参考了其中的一些，而且我认为其中几本应该摆放在同类书中最醒目的位置。

书中大部分素材取自我在 Monta Vista 这些年收到的一些工程师的来信。当时我的职责是嵌入式 Linux 顾问、FAE（Field Application Engineer，现场应用工程师）。Monta Vista 是一个领先的嵌入式 Linux 发行版开发商。

即使对于很有经验的软件工程师来说，嵌入式 Linux 也带来了几个空前的挑战。首先，对那些具有多年实时操作系统（RTOS）开发经验的工程师，他们很难把思维转换到 Linux 上来；第二，应用程序开发人员通常很难理解交叉开发环境之间的复杂关系。

虽然本书是定位在刚接触嵌入式 Linux 开发人员的初级读物，但是我确信有经验的开发人员也一定能从书中找到有用的提示和技巧，因为这些是我多年积累的经验。

给嵌入式 Linux 开发者的建议

书中还包括我的一些观点。作为一名嵌入式工程师，要跟上嵌入式 Linux 环境的快速发展，你需要知道这些观点。书中没有重点讲解 Linux 内核，在谈论内核的章节中重点介绍了内核工程，你可以阅读专门介绍内核的著作来了解内核的知识。通过本书可以学习到内核源码树的组织和布局，可以了解内核镜像的二进制文件的组成部分，以及这些内容如何被载入，它们在嵌入式系统中的作用等。图 5-1 是我最欣赏的图之一，它以图解的方式形象地说明了创建内核镜像的完整过程。

在本书的一些章节中，你可以学习构建（build）系统的工作原理，以及怎样将定制的内核变化加入到内核，以满足项目的需要。你将了解用于驱动不同体系结构的配置和 Linux 内核源码树的特性。更重要的是，掌握如何修改系统使之满足自己的需求，我们将在内核命令行参数机制的内容中展开深入讨论。你还将学习到它如何工作、如何根据需求配置内核运行

时的行为、如何扩展系统功能等内容，学习如何导航内核源代码，为相关嵌入式系统的不同任务配置内核的能力。其它一些内容还包括一些嵌入式工程中非常有用的提示和技巧，内容涵盖了 bootloader、系统初始化、文件系统、Flash 存储和内核、应用程序调试技巧。

读者对象

阅读本书需要具有一定的 C 语言编程基础，对本地局域网和互连网有基本的了解，应该理解 IP 地址的概念和简单的本地局域网使用。你也需要理解十六进制和八进制编码方式，以及它们通常的使用方式。

本书也涉及一些 C 编译和链接中较为深入的概念，所以如果你能再粗略翻翻 C 语言编程中链接器的概念就更好了。同时，了解 GNU make 操作和语法对于阅读本书也很有帮助。

本书没有什么

本书不是一本详细介绍硬件的指南。嵌入式开发者面临的困难之一就是现在硬件设备之间的巨大不同。一款集成部分外围设备的 32 位处理器，其用户手册动辄 1000 页，这没有捷径可走。但从程序员的角度看，如果需要理解硬件设备，你必须花费大量时间研读硬件数据手册和参考指南，同时要花费更多的时间编写和测试这些硬件设备工作的代码。

这也不是一本讲述 Linux 内核和内部原理的书籍。书中无法学到实现 Linux 虚拟内存管理机制和方式的复杂内存管理单元。其实现在有一些关于这个主题的优秀书籍，建议翻阅每章后面的相关推荐参考读物。

排版约定

文件名和代码采用 Courier 字体，需要读者输入的命令使用加粗 **Courier** 字体。新术语或重要的概念使用斜体加以强调。

路径名前有三个小数点表示路径的顶层，这个约定是众所周知的。顶层目录根据上下文会有所不同，但大多数情况下是指 Linux 内核源码目录的顶层。例如，`.../arch/ppc/kernel/setup.c` 表示 `setup.c` 文件位于 Linux 内核源码树的体系结构分支。实际的路径可能是：

```
~/sandbox/linux.2.6.14/arch/ppc/kernel/setup.c.
```

本书的组织

第 1 章简要介绍了快速采用嵌入式 Linux 环境的因素，介绍了和嵌入式 Linux 相关的几个重要的标准和组织。

第 2 章向读者介绍许多和嵌入式 Linux 相关的概念，这些概念在后续章节中将提到。

第 3 章我们将站在更高的层面了解在嵌入式 Linux 系统中广为流行的处理器和平台，介绍从主要处理器厂商选择的几款产品，几乎所有主流的体系结构都会谈到。

第 4 章从另一个不同的角度说明 Linux 内核。我没有重点讲解内核的原理或深入进去，只是介绍了内核的结构、布局和编译过程，目的是使读者能够开始进行这门庞大的软件工程项目的学习。更重要的是，要知道哪些内容是必须重点关注的，包括对创建内核系统的详细讲解。

第 5 章详细说明了 Linux 内核的初始化过程。读者可以学习到与体系结构和引导程序相关的代码如何链接到一起，并下载到 Flash 适当的位置，最终通过嵌入式系统的 bootloader 启动的。在本章学到的知识将帮助你定制 Linux 内核，使之满足嵌入式应用的需要。

第 6 章继续讲述初始化过程。当 Linux 内核完成自身初始化后，应用程序将根据预先确定的方式继续初始化过程。读完本章以后，读者将具备定制用户空间应用程序启动顺序的能力。

第 7 章主要介绍 bootloader 及其在嵌入式 Linux 系统中的作用。本章以现在流行的开源项目 U-Boot 为例，说明了移植的概念。本章还简要介绍了其他几种现在使用的 bootloader，这样当我们有特殊需求时，可以有多种选择。

第 8 章介绍了 Linux 设备驱动模型，提供了很多进行设备驱动开发的背景资料，这些资料都在推荐参考读物中列出。

第 9 章列出了目前嵌入式系统中使用的一些流行的文件系统，包括在 Flash 存储设备上最常用的 JFFS2 文件系统。本章还简要介绍了如何创建自己的文件系统镜像，这也是嵌入式 Linux 开发者所面临的一个艰巨的任务。

第 10 章介绍了内存技术设备（Memory Technology Devices, MTD）子系统。MTD 是 Linux 文件系统和硬件内存设备之间一种非常有用的抽象层，特别是 Flash 存储中。

第 11 章介绍了 BusyBox，它是我们创建小型嵌入式系统中最常用的工具。本章讲述如何根据特殊需要进行配置并编译，随后介绍了仅使用 BusyBox 环境完成系统初始化全过程的方法。附录 C 列出了最新版本 BusyBox 提供的命令。

第 12 章详细介绍了典型的交叉开发环境的特殊需求。在本章介绍的一些技术能有效地提高嵌入式开发人员的工作效率，例如非常有用的 NFS 的配置。

第 13 章介绍了一些有用的开发工具，包括使用 gdb 进行调试、核心转储分析。以实例的形式介绍了 strace、ltrace、top、ps、内存剖析工具 mtrace 和 dmalloc 工具。本章最后介绍了更重要的一些工具，如 readelf 等。

第 14 章深入介绍了一些用于 Linux 内核的调试技术，介绍了内核调试器 KGDB 的用法，提出了 gdb 和 KGDB 组合使用的一些调试技巧。本章涉及的内容还包括当内核无法启动时使用硬件 JTAG 调试器的方法和技巧。

第 15 章把调试的内容从内核转移至应用程序上。本章继续使用前面两章用到的 gdb 例子，这里讲述了多线程和多进程的调试技巧。

第 16 章介绍了将 Linux 移植到定制开发板的相关问题。本章通过一个简单的实例，逐步说明移植 Linux 内核到 PowerPC 板的详细过程。初次接触 Linux 内核移植的开发人员也将学到几个重要的概念。和第 13 章、第 14 章提出的技术一样，读完本章后，要求读者能够对自己的开发板进行移植工作。

第 17 章介绍了嵌入式 Linux 中一个令人激动的发展：通过配置 CONFIG_RT 选项实现实时。这里介绍的特性通过 RT 选项得以实现，同时还介绍了如何在设计中使用这些特性。本章还介绍了在应用程序中测试延时性的技巧。

附录内容包括 GNU 公共许可、U-Boot 命令、BusyBox 命令、SDRAM 接口注意事项、开源开发者的资源和 BDI-2000 调试器的配置文件范例。BDI-2000 是目前很流行的硬件 JTAG 调试器。

其他

如果你能够边看书，边在 Linux 工作站上动手实验的话，将会从书中得到最大的收获。找一个旧一点的 x86 计算机完成嵌入式系统实验。如果你有条件能连接其他架构的平台进行实验就更好了。

你将受益于学习到海量代码（如 Linux 内核）工程的布局和组织，在浏览内核并动手实验时，能学到一些更重要的知识和经验。

看一下本书使用的实例代码并试着理解它们，使用不同的设置、配置选项设置和不同的硬件设备。除此之外，阅读本书还充满了乐趣！

GPL 版权事项

本书使用的开源代码的版权由很多的个人或公司提供。所复制的代码遵循 GNU 公共许可或 GPL 条款。

附录 A 列出了 GNU 通用公共许可证的正文。

Contents

Chapter 1	Introduction.....	1
1.1	Why Linux?	2
1.2	Embedded Linux Today	3
1.3	Open Source and the GPL	3
1.3.1	Free Versus Freedom	4
1.4	Standards and Relevant Bodies	5
1.4.1	Linux Standard Base	5
1.4.2	Open Source Development Labs	5
1.5	Chapter Summary	7
1.5.1	Suggestions for Additional Reading	7
Chapter 2	Your First Embedded Experience	9
2.1	Embedded or Not?	10
2.1.1	BIOS Versus Bootloader	11
2.2	Anatomy of an Embedded System	12
2.2.1	Typical Embedded Linux Setup	14
2.2.2	Starting the Target Board	15
2.2.3	Booting the Kernel	16
2.2.4	Kernel Initialization: Overview	17
2.2.5	First User Space Process: init	19

2.3 Storage Considerations	19
2.3.1 Flash Memory	20
2.3.2 NAND Flash	22
2.3.3 Flash Usage	23
2.3.4 Flash File Systems	24
2.3.5 Memory Space	25
2.3.6 Execution Contexts	26
2.3.7 Process Virtual Memory	28
2.3.8 Cross-Development Environment	30
2.4 Embedded Linux Distributions	32
2.4.1 Commercial Linux Distributions	33
2.4.2 Do-It-Yourself Linux Distributions	34
2.5 Chapter Summary	34
2.5.1 Suggestions for Additional Reading	35
Chapter 3 Processor Basics	37
3.1 Stand-alone Processors	38
3.1.1 IBM 970FX	39
3.1.2 Intel Pentium M	39
3.1.3 Freescale MPC7448	40
3.1.4 Companion Chipsets	41
3.2 Integrated Processors: Systems on Chip	43
3.2.1 PowerPC	44
3.2.2 AMCC PowerPC	44
3.2.3 Freescale PowerPC	47
3.2.4 MIPS	52
3.2.5 Broadcom MIPS	53
3.2.6 AMD MIPS	55
3.2.7 Other MIPS	56
3.2.8 ARM	56
3.2.9 TI ARM	56
3.2.10 Freescale ARM	58
3.2.11 Intel ARM XScale	59
3.2.12 Other ARM	60
3.2.13 Other Architectures	60
3.3 Hardware Platforms	61
3.3.1 CompactPCI	61
3.3.2 ATCA	61

3.4 Chapter Summary	62
3.4.1 Suggestions For Additional Reading	63
Chapter 4 The Linux Kernel—A Different Perspective.....	65
4.1 Background	66
4.1.1 Kernel Versions	67
4.1.2 Kernel Source Repositories	69
4.2 Linux Kernel Construction	70
4.2.1 Top-Level Source Directory	70
4.2.2 Compiling the Kernel	71
4.2.3 The Kernel Proper: vmlinux	73
4.2.4 Kernel Image Components	75
4.2.5 Subdirectory Layout	79
4.3 Kernel Build System	79
4.3.1 The Dot-Config	80
4.3.2 Configuration Editor(s)	82
4.3.3 Makefile Targets	85
4.3.4 Kernel Configuration	88
4.3.5 Custom Configuration Options	90
4.3.6 Kernel Makefiles	94
4.3.7 Kernel Documentation	95
4.4 Obtaining a Linux Kernel	96
4.4.1 What Else Do I Need?	96
4.5 Chapter Summary	97
4.5.1 Suggestions for Additional Reading	98
Chapter 5 Kernel Initialization.....	99
5.1 Composite Kernel Image: Piggy and Friends	100
5.1.1 The Image Object	103
5.1.2 Architecture Objects	104
5.1.3 Bootstrap Loader	105
5.1.4 Boot Messages	106
5.2 Initialization Flow of Control	109
5.2.1 Kernel Entry Point: head.o	111
5.2.2 Kernel Startup: main.c	113
5.2.3 Architecture Setup	114