

GNU/Linux 嵌入式快速编程

[美] 鲁道夫·焦梅蒂 (Rodolfo Giometti) 著
俞辉 何旭莉 王晓红 李永 译

GNU/Linux
Rapid Embedded Programming

- 本书介绍如何使用和改造Linux内核中自有的驱动，如何找到相应的设备节点，如何对各种端口进行控制等。而且还提供了大量的案例程序，无论是入门者，还是进行二次开发的程序员，都可以从中汲取经验。
- 自由环境，自由学习。学习本书不用担心被局限在固定的某个平台之中进行少量的二次开发。因为丰富的硬件和Linux系统的开放性，会给阅读者带来自由的感受。



GNU/Linux 嵌入式快速编程

GNU/Linux
Rapid Embedded Programming

[美] 鲁道夫·焦梅蒂 (Rodolfo Giometti) 著
俞辉 何旭莉 王晓红 李永 译

常州大学图书馆
藏书章



机械工业出版社
China Machine Press

图书在版编目 (CIP) 数据

GNU/Linux 嵌入式快速编程 / (美) 鲁道夫·焦梅蒂 (Rodolfo Giometti) 著; 俞辉等译.

—北京: 机械工业出版社, 2018.5

(Linux/Unix 技术丛书)

书名原文: GNU/Linux Rapid Embedded Programming

ISBN 978-7-111-59829-9

I. G… II. ①鲁… ②俞… III. Linux 操作系统 - 程序设计 IV. TP316.85

中国版本图书馆 CIP 数据核字 (2018) 第 076115 号

本书版权登记号: 图字 01-2017-3410

Rodolfo Giometti: GNU/Linux Rapid Embedded Programming (ISBN: 978-1-78646-180-3).

Copyright © 2017 Packt Publishing. First published in the English language under the title “GNU/Linux Rapid Embedded Programming”.

All rights reserved.

Chinese simplified language edition published by China Machine Press.

Copyright © 2018 by China Machine Press.

本书中文简体字版由 Packt Publishing 授权机械工业出版社独家出版。未经出版者书面许可, 不得以任何方式复制或抄袭本书内容。

GNU/Linux 嵌入式快速编程

出版发行: 机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码: 100037)

责任编辑: 唐晓琳

责任校对: 殷虹

印刷: 北京市兆成印刷有限责任公司

版次: 2018 年 5 月第 1 版第 1 次印刷

开本: 186mm × 240mm 1/16

印张: 31.75

书号: ISBN 978-7-111-59829-9

定价: 109.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88379426 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzit@hzbook.com

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问: 北京大成律师事务所 韩光 / 邹晓东

The Translator's Words 译者序

本书是一本非常棒的嵌入式系统教程，硬件基于 BeagleBone Black 系统、SAMA5D3 Xplained 系统和 Wandboard 系统三种开发平台，软件则是非常流行的 Linux 系统。整本书分为两大部分，第一部分是基础部分，包括开发环境的建立、简单命令和编程工具的使用，涉及编译器、编程工具、脚本和守护进程等。在读者打好嵌入式系统的基础之后，本书将进入第二部分，也就是各种基于硬件设备的应用。本书使用的硬件设备几乎涵盖我们见过的所有连接各种开发板的方式，包括 I2C、SPI、USB、单总线、串行总线以及各类数字模拟设备等，每一章都会针对一种类型的总线或者设备类型，进行详细的介绍，哪怕是初学者都能够按照书中教程一步一步完成相应的开发。除了第一部分之外，后面各章之间关系不大，每章都对应一种设备总线，可以进一步帮助读者进行有针对性的学习和二次开发。

本书最主要的特点就是**知识点丰富，涵盖面大**。硬件平台包括目前最主流的 3 个嵌入式系统平台，操作系统也是目前最流行的 Linux。书中涵盖各种硬件端口、总线的驱动编写、编程控制等，说它是百科全书都不为过。没有涉及的比较深层的内容在书中也会提供相应的网址和资料，以帮助读者进一步学习。

本书的第二个特点就是**深入浅出，循循善诱**。哪怕读者是初学者也没关系，里面的命令和操作都是按照零基础的标准来安排的。书中会告诉你，如何使用和改造内核中原有的驱动，如何找到相应的设备节点，如何对各种端口进行控制等。而且还提供了大量的案例程序，无论是入门者，还是进行二次开发的程序员，都可以从中取经。

本书的第三个特点就是**自由**。自由的环境，自由的学习。学习本书不用担心被局限在固定的某个平台之中进行少量的二次开发。因为丰富的硬件和 Linux 系统的开放性会给阅读者带来自由的感受。Linux 本身就是自由的，其“一切都是文件”的特性在嵌入式系统的开发中

如鱼得水，在操作系统的支持下，嵌入式系统可以很方便地实现各种复杂功能。

当然，本书也有不足之处，例如书中有些内容介绍得过于细致了，其实在开始的章节第一次介绍时翔实一些是没问题的，不过到了后面还详细地说明就没有太大必要了。当然，这对那些跳跃着阅读的读者可能会比较有用。

Preface 前言

嵌入式计算机在最近几年变得非常复杂，而开发者专注的问题应该是如何解决当下的问题，而不用在寻找更好的外设（简称为外设）或者学习如何管理它们这些方面浪费时间。对于有经验的嵌入式程序员和工程师来说，最大的挑战是把一个想法变成现实到底需要多长时间，本书将展示如何做到这一点。

本书展示了行业中外部环境是如何通过特定的外设进行交互的。这里将使用最新的 Linux 内核版本 4.x，软件系统基于 Debian/Ubuntu 系统，而嵌入式系统则使用 OpenWrt 和 Yocto。

本书涉及的开发板平台基于目前元件市场和专业开发板制造商中最流行的 CPU。在简要介绍完每个平台之后，本书将展示如何设置这些平台，使之能运行一个完整的发行版 GNU/Linux 并且获得对其系统控制台的访问权。之后将介绍如何针对每个平台安装一个完整的开发系统，使开发者能够运行自己开发的程序。

第一步将在嵌入式套件下进行编程，使用 C、Bash 脚本或者 Python/PHP 等语言来编程以访问外设。之后将会介绍更多的设备驱动程序和访问外设的内容以奠定坚实的基础。读者将学习通过 C 程序或者脚本语言（Bash/Python）和外设进行交互，以及如何为特定的硬件配置相应的设备驱动程序。

本书使用的硬件设备几乎涵盖我们所能见过的所有连接各种开发板的方式，包括 I2C、SPI、USB、1-wire 总线、串行总线以及各类数字和模拟设备等。

本书中使用的编程语言依据的原则是，以最快速最简单的解决方案来解决当前的问题。特别是可以找到 Bash、C、PHP、Python 的示例代码。

在这些章节中将会需要使用一个守护进程、一个内核模块或者可能需要重新编译整个内核。这种情况下添加了一个简短的描述，包括读者应该做什么、可以从哪里得到更多的信息和使用工具。但要说明的是，管理一个 GNU/Linux 系统或者内核模块等一些基本的技能是必需的。

本书涵盖内容

本书可以分为两大部分：其中第一部分是第二部分的基础。第一部分介绍如何安装一个宿主机和开发系统以及如何通过串口控制台访问嵌入式套件。首先将看到一些引导加载程序的命令以及 C 的编译器和交叉编译器，之后会介绍内核模块、设备驱动程序以及一些基于网络文件系统的文件系统。读者也会在主机上使用模拟器来运行一个完整的目标机 Debian 系统，读者也会在 Bash、PHP 和 Python 下面看到系统的守护进程和脚本程序，读者还会看到闪存和 Linux 的 MTD（Memory Technology Device，内存技术设备）。其中会介绍 JFFS2 和 UBIFS 文件系统，以及两个非常著名的嵌入式系统 Yocto 和 OpenWrt。

下面是第一部分中每章的简短介绍：

第 1 章介绍目前工业应用中的最常用的三个开发环境：BeagleBone Black 系统、SAMA5 D3 Xplained 系统和 WandBoard 系统。在简要介绍了每个开发板之后，我们将看到如何设置它们以便运行一个完整的 GNU/Linux 系统。我们也将看到如何在每个平台（和虚拟机中）上安装一个完整的开发系统。

第 2 章首先将展示如何安装开发包，然后将进一步探索串口控制台和运行在其上的 shell。最后将介绍引导系统 bootLoader 的命令。

第 3 章将讲述 C 编译器以及交叉编译器，之后我们将看到内核模块、设备驱动程序以及一些基于 NFS 的文件系统，最后会看到一个开发者在主机上如何使用模拟器来运行一个完整的目标机 Debian 系统。

第 4 章将介绍系统守护进程（描述它们是什么以及如何使用它们），之后将使用 Bash、PHP 和 Python 语言来进行脚本编程。

第 5 章首先介绍闪存和管理闪存的软件，之后将展示 Linux 的内存技术设备（MTD）和两个主要的文件系统 JFFS2 和 UBIFS，并运行它们。最后将展示如今两个最著名的嵌入式系统——Yocto 和 OpenWrt，学习如何在这些嵌入式系统上编写应用程序，并且把该程序添加到嵌入式系统中。

第二部分开始深入介绍所有外设。如果你是一位嵌入式开发人员，在你的职业生涯所遇到的各种外设都会涉及。在本书中，对于每一个外设，都会介绍其所支持的嵌入式套件以及如何访问和使用它。对于每种设备类型，都会结合实例展示如何管理它。

下面是第二部分中每章的简短介绍：

第 6 章首先简短描述 GPIO 线路，之后将会看到它们在嵌入式套件上是如何布局的，接着再详细了解如何以非常简单（但效率低）的方式访问这些线路，然后再以更智能（但更复

杂)的方式重新再次访问。最后会在内核中通过 IRQ 管理和 LED 设备支持再快速地介绍一下 GPIO 管理机制。

第 7 章将介绍串行端口(简称串口),串口是计算机中最重要的一个外设接口(至少可以让计算机用于控制自动化行业等)。简要描述串行端口和串行设备之后,我们将通过使用一个实际的串口设备来看如何在一个 GNU/Linux 系统中对其进行管理。最后我们将在内核中学会如何通过一条串行线来实现网线连接的两个嵌入式系统(二者通过一条以太网电缆连接)之间的通信。

第 8 章将介绍 USB 总线,这是一种通用的总线,广泛应用于现代计算机。人们可以将各种电子设备连接到计算机,例如硬盘、键盘或其他使用相同 USB 端口的串行设备。

第 9 章将介绍 I2C 总线,它通常用于连接板上设备,即主计算机会将所有设备都放在同一块板上。一些设备使用 I2C 总线与 CPU 进行通信,而这一章将尽量向读者提供尽可能广泛的场景:我们将看到几种不同配置的不同设备如何在该总线中尽量组合成各种情况。

第 10 章将介绍 SPI 总线,这是另一种总线类型,和 I2C 一样通常用于连接板上设备。然而,和 I2C 总线相反,该总线可以以更高的速度进行数据传输,因为它可以进行全双工操作,即可以同时双向传输数据。由于这些特性, SPI 总线通常用于为多媒体应用、数字信号处理、电信设备和 SD 卡实现有效的数据流传输。

第 11 章将介绍长距离下以太网设备之间的通信 1-Wire。GNU/Linux 系统对以太网设备及其相关网络协议提供相当不错的支持,这也是世界上大多数网络设备使用该技术的原因。

第 12 章介绍以太网网络设备。以太网设备很有趣,尽管速度较慢,但它只用一根电线就可以实现和远程设备的交互,这就简化了 CPU 及其外设之间的连接。如果要在嵌入式套件上使用识别、验证和交付校准数据或制造信息等功能,设计者可以用更经济简单的方式添加电子设备。

第 13 章介绍无线网络设备,可以在不使用电线的情况下实现多台计算机之间的通信。不过真正有趣的是,以太网接口上的大部分通信协议也可以与这些无线网络设备接口一起使用。

第 14 章将介绍 CAN 总线, CAN 总线允许系统中的微控制器、计算机和设备在没有主机的前提下使用基于消息的协议进行通信。CAN 总线不如以太网或者 WiFi 那么有名,但在嵌入式领域还是经常使用的,而且被 SoC 默认支持。

第 15 章介绍声音设备的一些用法,从而展示如何使用它们来产生音频,以及如何实现一个简单而低成本的示波器。

第 16 章介绍常见的视频采集设备和相关的一些使用方法,展示如何把嵌入式套件变成一个监控摄像头或远程图像记录仪。

第 17 章介绍模数转换器——ADC。模数转换器可用于从环境中得到模拟信号并转换成数字信号。本章将展示如何使用它们，以及如何使用特定的软件和硬件触发器，这样在特定时间或某些事件发生时可以启动转换。

第 18 章介绍脉宽调制——PWM。PWM 可以使用数字源并通过将消息编码到一个脉冲信号（通常是一个方波）的方式产生一个模拟信号，然后这些消息可以用来控制电动机或其他电子设备以及控制一个伺服电动机的位置。

第 19 章将介绍上述章节之外的其他外设。最后一章将展示我们可能会在某个监测或控制系统里遇到的一个额外的外设列表，如 RFID 和智能卡读卡器、数字或模拟传感器、GSM/GPRS 调制解调器、Z-Wave 等。

需要的准备工作

本书中你需要准备的东西包括以下内容。

软件准备

本书默认读者有一些非图形化的文本编辑器的相关知识，编辑器可以是 vi、emacs 或 nano 等。即使你能够通过嵌入式套件与液晶显示器、键盘或者鼠标直接相连并使用图形界面，但在本书中我们还是假设读者只能通过文本编辑器来对文本文件进行修改。

宿主机——将用于交叉编译代码和管理嵌入式系统的计算机——要求运行的是 GNU/Linux 系统。我的宿主机系统是 Ubuntu 15.10，但只要稍稍修改，你就可以使用一个更新版本的 Ubuntu 系统或一个 Debian 系统，或者可以用另一个 GNU/Linux 系统，但需要安装交叉编译工具、支持库和进行软件包管理。本书不会涉及如 Windows、Mac OS 或者其他类似系统，因为不应该使用这些低技术层次的系统来为高技术层次的系统开发代码。

读者还需要了解一个 C 编译器是如何工作的，以及 Makefile 是如何管理的。

本书将介绍一些内核编程技术，但这些不能作为核心编程课程的内容。这方面你需要一本适合这类主题的书。然而，每个示例都有很好的文档说明，而且你会看到一些相关的建议资源。本书使用的内核版本是 4.4.x。

最后一点，假设读者已经知道如何在 Internet 上连接一个 GNU/Linux 系统，这样你可以下载某个包或者通用文件。

硬件准备

在本书中所有开发的代码是基于 BeagleBone Black 平台、SAMA5D3 平台和 WandBoard

平台的。但是你可以使用一个旧版本的嵌入式套件，这不会有任何问题。事实上，代码是可移植的，它可以在其他系统上运行（不过 DTS 文件是需要分开的）。

关于本书中所使用的计算机外设，每一章中都说明了可以从哪里买到，当然读者也可以从网上找到更好更便宜的报价。本书还说明了能找到相应的数据手册的网址。

这样你在使用本书所涉及的嵌入式套件和工具包时不会遇到任何困难，因为这些连接非常简单，而且文档齐全。这些内容不需要任何特定的硬件技能，但如果你知道一点相关的电子知识可能会有所帮助。

本书读者对象

如果你想学习如何使用嵌入式系统，包括通过访问 GNU/Linux 设备驱动程序的方式从外设收集数据或控制某个设备，那这本书是为你编写的。

如果你感兴趣于如何方便快捷地访问不同的计算机外设，以便为行业应用实现基于 GNU/Linux 系统的功能控制或监控系统，这本书也是为你编写的。

如果你有一些硬件或电气工程的经验，并且熟悉 UNIX 环境中的 C、Bash、Python 和 PHP 编程的基本知识，并且希望能够将它们应用到嵌入式系统中，这本书也是为你编写的。

约定

在本书中你可以发现很多不同信息的文本样式。下面将对这些类型进行举例并进行含义解释。

内核和日志消息

通常情况下 GNU/Linux 系统的内核消息如下所示：

```
Oct 27 10:41:56 hulk kernel: [46692.664196] usb 2-1.1: new high-speed USB
device number 12 using ehci-pci
```

对于本书来说，这一行命令太长了，这也是为什么从第一个字符开始直到真正的信息开始之前的这些字符会被删除。因此在刚才的示例中，行输出将会显示如下：

```
usb 2-1.1: new high-speed USB device number 12 using ehci-pci
```

然而，正如上面所说的，如果一行命令太长将会换行。

在终端中过长的输出、重复或者不太重要的内容会以三个点“...”代替，如下所示：

```
output begin
output line 1
output line 2
...
output line 10
output end
```

这三个点在结束时表示输出仍在继续，但为了省空间而省略了。

文件的修改

当修改某个文本文件时，我将使用统一的上下文差异格式进行说明。在表示文本修改方面，这是一种非常高效和紧凑的方式。这种格式可以通过使用带有 `-u` 选项参数的 `diff` 命令获得。

举个简单例子，下面的文本内容为 `file1.old`：

```
This is first line
This is the second line
This is the third line
...
...
This is the last line
```

假设我们必须在下面的代码段中突出显示修改的第三行：

```
This is first line
This is the second line
This is the new third line modified by me
...
...
This is the last line
```

这样你可以很容易地理解。每次对整个文件进行简单修改时，这是相当模糊且占用篇幅的，但是通过使用统一的上下文差异格式，前面的修改可以写成如下形式：

```
$ diff -u file1.old file1.new
--- file1.old 2015-03-23 14:49:04.354377460 +0100
+++ file1.new 2015-03-23 14:51:57.450373836 +0100
@@ -1,6 +1,6 @@
  This is first line
  This is the second line
-This is the third line
+This is the new third line modified by me
  ...
  ...
  This is the last line
```

现在修改哪些地方就非常明确了，而且篇幅紧凑。开始两行里面，`---` 表示修改之前的原始文件，而 `+++` 表示修改之后的新文件，然后是文件所包含的修改内容的差异。原始文件中不修改的内容在最前面增加一个空格字符，而要添加的行前面加一个 `+` 字符，被删除的行前面加一个 `-` 字符。

由于篇幅原因，本书中的大多数代码都减少了缩进以适合纸质版的页面宽度，但它们仍然保持可读的正确形式。那些真正的代码补丁可以从 GitHub 或 Packt 网站上获取。

串行和网络连接

在这本书中我主要使用两种不同类型的连接方式来实现宿主机与嵌入式套件的交互：使用串口控制台实现的串行连接方式，以及使用 SSH 终端实现的以太网连接方式。

串口控制台是在和嵌入式套件相连接的同一个 USB 接口上实现的，主要用于通过命令行管理系统。它主要用于监控系统，特别是对内核消息进行控制。

SSH 终端和串口控制台非常相似，即使不完全相同（例如内核消息不会自动在终端上显示），但它可以像串口控制台一样使用，通过命令行输入命令或者编辑文件。

在本书后面的章节中，我将通过串口控制台或 SSH 连接的一个终端来实现大多数的命令和配置操作。

要从你的主机开始访问串口控制台，可以使用如下 minicom 命令：

```
$ minicom -o -D /dev/ttyACM0
```

或者根据开发板的 USB 转串行适配器使用如下命令：

```
$ minicom -o -D /dev/ttyUSB0
```

然而，在第 1 章中对这方面已经做了解释，所以并不用担心。有些系统可能需要 root 权限才能够访问 /dev/ttyACM0 设备。这种情况下你可以通过使用 sudo 命令或更好的处理方式，例如添加新的系统用户到正确的用户组中来解决这个问题，命令如下：

```
$ sudo adduser $LOGNAME dialout
```

然后注销并再次登录，你将可以畅通无阻地访问串行设备。

要访问 SSH 终端，你可以使用与串口控制台相同的 USB 接口电缆来模拟以太网连接。事实上，如果你的主机配置正确，当插入 USB 连接线一段时间后，你会看到一个新的附带 IP 地址的连接（如果连接 BeagleBone Black 开发平台，你应该得到地址 192.168.7.1 的连接；如果连接 SAMA5D3 Xplained 平台，则是 192.168.8.1 地址；如果连接 WandBoard 平台，则是 192.168.9.1 地址。具体请见第 1 章）。然后通过这个新连接，就可以使用下面的命令来访问 BeagleBone Black 嵌入式套件。

```
$ ssh root@192.168.7.2
```

最后一个可以使用的通道是以太网连接。它主要用于从宿主机或因特网上下载文件，通过将以太网水晶头连接到每个嵌入式套件的以太网端口，然后将端口配置为读者的局域网设置，从而可以建立连接。

但很重要的一点是，你也可以通过使用上面提到的 USB 仿真以太网连接来连接因特网。事实上，通过使用宿主机上的命令（宿主机是 GNU/Linux 系统），你可以把它作为一个路由器，让你的嵌入式套件与以太网端口连接时能够实现上网：

```
# iptables --table nat --append POSTROUTING --out-interface eth1
    -j MASQUERADE
# iptables --append FORWARD --in-interface eth4 -j ACCEPT
# echo 1 >> /proc/sys/net/ipv4/ip_forward
```

然后在 BeagleBone Black 平台上通过 USB 连接线，可以使用如下命令设置网关：

```
root@bbb:~# route add default gw 192.168.7.1
```

请注意，我的主机系统中 eth1 设备是首选的网络连接，而在我的嵌入式系统中 eth4 设备是 BeagleBone Black 的设备。

其他约定



警告或注意标志



提示或技巧标志

下载示例代码及彩图

本书的示例代码文件可以从作者的 GitHub 库 https://github.com/giometti/gnu_linux_rapid_embedded_programming 获取，只要使用如下命令：

```
$ git clone
  https://github.com/giometti/gnu_linux_rapid_embedded_programming.git
```

读者也可以访问华章图书官网 www.hzbook.com，通过注册并登录账号，下载本书的源代码和彩图。

作者简介

Rodolfo Glometti 是一位工程师、IT 行家、GUN/Linux 专家和自由软件的传道者。

他也是书籍《BeagleBone Essentials》和《BeagleBone Home Automation Blueprints》(Packt 出版社初版)的作者, LinuxPPS 工程(Linux 的 Pulse Per Second 子系统)的维护者。目前仍乐于贡献行业应用设备的相关补丁和新设备驱动的 Linux 源代码。

在他二十多年的工作经历中, 使用过 x86、ARM、MIPS、PPC 等平台。

现在, Rodolfo 是 HCE 工程 S.r.l 的联合首席, 同时也是 Cosino 工程的联合创始人。这些工程能够为工业环境、自动控制和远程监测领域的快速原型机提供新的软硬件系统。

我要感谢我的妻子 Valentina 以及我的孩子 Romina 和 Raffaele, 感谢他们在我写本书期间付出的耐心。我也非常感谢 Packt 的工作人员: Vivek Anantiaraman (特别感谢 Juliana Nair, 她为了完成本书, 好多个晚上一直在忙碌), 他们给了我写这本书的机会。非常感谢 Luca Zulberti 和 Chris Simmonds 对本书的支持和帮助, 尤其是 Luca Zulberti 非常认真地校对书中的文字、代码以及所有的电路。还要感谢 Mohd Riyan Khan, 他最后对本书审稿时非常耐心地抄写我的修改意见。

最后, 我无法忘记我的父母。当我还是孩子时他们给了我第一台电脑, 并且允许我一直做我感兴趣的事情。

审校者简介

Luca Zulberti 在 2015 年毕业于意大利里窝那市 ITIS G.Galilei 的电子系, 之后在比萨大学的电子工程系继续他的学业。

他在空闲时间中, 会进行多种编程技术和语言的研究, 对嵌入式系统、嵌入式操作系统开发板和 GNU/Linux 有浓厚的兴趣。

他是 Cosino 项目的成员, 发表过一些技术文章, 也完成过一些工程。

目 录 Contents

译者序

前言

第一部分 基础知识

第1章 开发系统的安装 2

1.1 嵌入式术语 2

1.2 系统概述 4

1.2.1 BeagleBone Black 开发平台 4

1.2.2 SAMA5D3 Xplained 开发
平台 6

1.2.3 Wandboard 开发平台 8

1.3 安装开发系统 11

1.3.1 设置宿主机 12

1.3.2 设置 BeagleBone Black 开发
平台 14

1.3.3 设置 SAMA5D3 Xplained 开发
平台 24

1.3.4 设置 Wandboard 开发平台 30

1.4 设置开发系统 35

1.4.1 通过 USB、网络、overlay

系统设置 BeagleBone Black 36

1.4.2 通过 USB、网络设置

SAMA5D3 Xplained 39

1.4.3 通过 USB、有线网络和无线

网络设置 Wandboard 42

1.4.4 通用设置 45

1.5 总结 46

第2章 管理系统控制台 47

2.1 基本操作系统管理 47

2.1.1 文件操作和命令 47

2.1.2 软件包管理 55

2.1.3 管理内核消息 60

2.2 快速浏览 bootloader 62

2.2.1 环境 63

2.2.2 管理存储设备 65

2.2.3 GPIO 管理 69

2.2.4 访问一个 I2C 设备 70

2.2.5 从网络加载文件 71

2.2.6 内核命令行 75

2.3 总结 76

第3章 C编译器、设备驱动及其

开发技术..... 77

3.1 C 编译器..... 77

3.1.1 宿主机和目标板不同的体系 结构..... 78

3.1.2 编译一个 C 程序..... 80

3.1.3 编译一个内核模块..... 85

3.2 内核和 DTS 文件..... 87

3.2.1 重新编译内核..... 87

3.2.2 设备树..... 91

3.3 什么是设备驱动? 93

3.3.1 字符设备、块设备和网络 设备..... 94

3.3.2 模块与内置设备..... 95

3.3.3 modutils (管理内核模块的 一个软件包)..... 95

3.3.4 编写一个设备驱动..... 97

3.4 根文件系统..... 106

3.4.1 /dev 目录..... 107

3.4.2 tmpfs 文件系统..... 109

3.4.3 procfs 文件系统..... 110

3.4.4 sysfs 文件系统..... 113

3.5 网络文件系统..... 115

3.5.1 本机输出一个 NFS..... 116

3.5.2 设置内核以挂载 NFS..... 117

3.5.3 通过 U-Boot 和内核命令行 使用 NFS..... 119

3.5.4 NFS 开发..... 122

3.6 模拟器的使用..... 122

3.6.1 执行一个程序..... 123

3.6.2 进入一个 ARM 的根文件 系统树..... 124

3.7 总结..... 128

第4章 使用脚本和系统守护进程的

快速编程..... 129

4.1 设置系统..... 129

4.2 系统守护进程..... 130

4.2.1 有用的和即用的守护进程... 131

4.2.2 系统守护进程管理..... 132

4.3 脚本编程语言..... 149

4.3.1 基于 PHP 管理 LED..... 150

4.3.2 基于 Python 管理 LED..... 153

4.3.3 基于 Bash 管理 LED..... 155

4.4 编写自定义守护进程..... 159

4.4.1 基于 C 的守护进程..... 160

4.4.2 基于 PHP 的守护进程..... 161

4.4.3 基于 Python 的守护进程..... 162

4.4.4 基于 Bash 的守护进程..... 162

4.5 总结..... 163

第5章 配置一个嵌入式操作系统..... 164

5.1 MTD 与块设备..... 164

5.1.1 什么是 MTD 设备..... 165

5.1.2 管理 MTD 设备..... 167

5.1.3 Flash 存储系统..... 170

5.2 OpenWrt 系统..... 179

5.2.1 使用缺省配置..... 180

5.2.2 添加(准) LAMP 系统..... 188

5.2.3 添加自定义配置..... 197

5.3 Yocto 系统..... 200

5.3.1 使用缺省配置..... 201

5.3.2 添加图形支持..... 210

5.3.3 添加自定义程序	212
5.4 总结	215

第二部分 外围设备

第6章 通用输入输出信号——

GPIO	218
6.1 GPIO	218
6.1.1 BeagleBone Black 系统下的 GPIO	221
6.1.2 SAMA5D3 Xplained 系统下的 GPIO	221
6.1.3 Wandboard 系统下的 GPIO	222
6.2 Linux 下的 GPIO	223
6.3 访问 GPIO	225
6.3.1 基于 Bash 的使用	225
6.3.2 基于 C 的使用	227
6.4 基于脚本语言使用 GPIO	234
6.4.1 基于 PHP 的使用	235
6.4.2 基于 Python 的使用	236
6.5 内核管理 GPIO	237
6.5.1 使用 GPIO 的输入设备	238
6.5.2 LED 和触发器	244
6.6 总结	248

第7章 串行端口和TTY设备——

TTY	249
7.1 TTY、串行和串口线	249
7.1.1 电气线路	250
7.1.2 BeagleBone Black 系统下的 TTY	251

7.1.3 SAMA5D3 Xplained 系统下的 TTY	253
--	-----

7.1.4 Wandboard 系统下的 TTY	254
7.2 串行端口的实现	254
7.3 Linux 系统下的串行端口	255
7.4 通信参数	256
7.5 访问 TTY	256
7.5.1 距离传感器	257
7.5.2 LF 低频 RFID 读写器	261
7.6 使用 SLIP 管理内核中的 TTY	263
7.7 总结	268

第8章 通用串行总线——USB

8.1 通用串行总线	269
8.1.1 电气线路	270
8.1.2 BeagleBone Black 系统下的 USB 端口	270
8.1.3 SAMA5D3 Xplained 系统下 的 USB 端口	270
8.1.4 Wandboard 系统下的 USB 端口	271
8.2 Linux 下的 USB 总线	271
8.2.1 USB 主机介绍	272
8.2.2 USB 设备介绍	275
8.3 USB 工具	284
8.4 raw USB 总线	286
8.5 总结	290

第9章 内部集成电路——I2C

9.1 内部集成电路总线	291
9.1.1 电子线路	292