

手把手教你开发一款Android智能机

WIFI无线上网

QQ语音聊天

微信聊天

家庭影院

游戏

HD HDMI



Android嵌入式系统开发实战 ——玩转Linux内核驱动开发

疯壳团队 陈万里 黄世林 刘燃 编著



西安电子科技大学出版社
<http://www.xduph.com>

手把手教你开发一款 Android 智能机

Android 嵌入式系统开发实战

玩转 Linux 内核驱动开发

疯壳团队 陈万里 黄世林 刘 燃 编著

西安电子科技大学出版社

内 容 简 介

Android 是当下最火的智能操作系统。本书以实际开发为例,由浅入深,带领读者快速掌握 Android 驱动开发的所有技能。本书作者具有多年的项目实战开发经验,书中包含了 Android 驱动开发所需的各方面的技术知识,从开发工具获取、开发环境搭建、电路图的简单分析,到真机下载与调试,都有详细讲解。

本书分为 3 章,分别从开发前的准备、开发基础知识和开发实战三个方面介绍 Android 驱动的开发工作。本书注重理论和实践相结合,从电路图的 analysis 入手,用实际的实验环境和例子为 Android 驱动的开发提供完整的案例。

对于想要从事 Android 驱动程序研发工作的在校大学生、程序开发爱好者或转行从业者,这是一本很好的入门教材;而对于已经入行,正在从事 Android 驱动程序开发的程序员来说,本书也能给予一定的参考和指导。本书语言通俗易懂,即使是从没接触过 Android 驱动开发的读者也能顺利上手,并能根据书中的实例进行实践。

随书的源码、视频、套件都可以通过 https://www.fengke.club/GeekMart/su_f90xheDAs.jsp 社区论坛获取。

图书在版编目(CIP)数据

Android 嵌入式系统开发实战:玩转 Linux 内核驱动开发 / 疯壳团队等编著. —西安:

西安电子科技大学出版社, 2018.11

ISBN 978-7-5606-5137-8

I. ①A… II. ①疯… III. ①移动终端—应用程序—程序设计 ②Linux 操作系统—程序设计
IV. ①TN929.53 ②TP316.85

中国版本图书馆 CIP 数据核字(2018)第 249523 号

策划编辑 高 樱

责任编辑 祝婷婷 阎 彬

出版发行 西安电子科技大学出版社(西安市太白南路 2 号)

电 话 (029)88242885 88201467 邮 编 710071

网 址 www.xduph.com 电子邮箱 xdupfxb001@163.com

经 销 新华书店

印刷单位 陕西天意印务有限责任公司

版 次 2018 年 11 月第 1 版 2018 年 11 月第 1 次印刷

开 本 787 毫米×1092 毫米 1/16 印张 23.875

字 数 572 千字

印 数 1~3000 册

定 价 53.00 元

ISBN 978-7-5606-5137-8 / TN

XDUP 5439001-1

如有印装问题可调换





前言

PREFACE

Android 是一种基于 Linux 的自由及开放源代码的操作系统，主要用于移动设备，如智能手机和平板电脑，由 Google 公司和开放手机联盟主导其开发，目前它统一的中文名称是“安卓”。Android 操作系统最初由 Andy Rubin 开发，主要支持手机，2005 年 8 月由 Google 公司收购注资。2007 年 11 月，Google 公司与 84 家硬件制造商、软件开发商及电信营运商组建开放手机联盟，共同研发改良 Android 系统。随后 Google 公司以 Apache 开源许可证的授权方式，发布了 Android 的源代码。第一部 Android 智能手机发布于 2008 年 10 月，此后，Android 的应用逐渐扩展到平板电脑及其他领域，如电视、数码相机、游戏机等。2011 年第一季度，Android 在全球的市场份额首次超过塞班(Symbian)系统，跃居全球第一。2013 年第四季度，Android 平台手机的全球市场份额已经达到 78.1%。2013 年 9 月 24 日，谷歌开发的操作系统 Android 迎来了 5 岁生日，全世界采用这款系统的设备数量已经达到 10 亿台。

Android 系统应用越来越广泛，市面上介绍 Android 开发的相关书籍也不少，但多是介绍 Android 应用程序开发的，而这对于想要从事 Android 驱动开发的人员来说，连入门都有点困难，这是因为首先很难找到一个开放的 ARM 硬件平台，并且平台上还要引出相应的硬件引脚或已经连接了相应的硬件模块供开发调试。正是基于此种现状，编者决定撰写本书。本书根据编者多年的 Android 驱动研发经验，以平板电脑研发为例，讲解实际产品开发流程，总结实际项目开发中的常见问题及常用知识点，帮助读者快速入门并学会 Android 驱动开发技能。

我们的开发板采用 RK3128 芯片方案，并引出了多组接口供用户开发和测试，希望用户的创意和灵感可以赋予芯片更完美的表现。RK3128 采用 Cortex-A7 架构四核 1.3 GHz 处理器，集成 Mali-400MP2 GPU，拥有优秀的运算与图形处理能力；板载千兆以太网口、2.4 GHz Wi-Fi 和蓝牙 4.0，展现出不俗的网络扩展和传输性能；同时可支持 Android 4.4 系统，并拥有丰富的硬件资源与扩展接口，所以它是一台扩展性特别强的卡片电脑。

本书的内容几乎涵盖了 Android 驱动开发中的所有知识点，虽然有些知识点讲得并不是很深入，但却告诉了读者如何获取相关资料。书中的章节内容都是根据实际项目开发步骤，按照从易到难的顺序安排的，建议读者按顺序学习。本书前面两章是 Android 平台相关的基础知识，读者首先需掌握开发环境的配置，然后掌握系统的编译方法。只有配置好了开发环境，并能使用相应的指令编译代码且上机通过，才能进行后面章节的学习。在学

习完所有的知识点后，编者以一个个的项目实战来提高读者的学习兴趣，让读者学会如何运用前面所学的知识。最后本书配套了一个商用的平板或机顶盒产品，作为读者实战开发的调试设备。

本书的特点如下：

(1) 实用性强。本书以真实的商用产品方案 RK3128 为例，全面讲解 Android 驱动开发流程和技能。虽然是以 RK3128 为例讲解的，但是相应的知识可以运用到任何使用 Android 的设备中。

(2) 专业权威。本书作者是 Android 驱动的一线开发者，拥有多年的 Android 项目开发经验，负责多款 Android 产品的开发及量产维护工作，书中内容全部来自真实项目的开发总结。

(3) 内容全面。本书基本涵盖了 Android 驱动开发的所有知识点。

(4) 实验可靠。书中所有源码都经过真实环境验证，有极高的含金量。

(5) 售后答疑。所有读者都可在 https://www.fengke.club/GeekMart/su_f90xheDAs.jsp 网站社区论坛提问，作者会不定期答疑。

本书的适用范围如下：

(1) 想从事 Android 驱动研发工作的在校大学生、程序开发爱好者或转行从业者。

(2) 已经入行，正在从事 Android 驱动开发的工程师。

(3) Android 驱动开发培训机构和单位。

(4) 高校教师或学生，本书可作为高校 Android 开发实验课程的教材。

本书由刘燃负责策划审校，第 1 章和第 3 章由刘燃、陈万里共同编写，第 2 章由陈万里、黄世林共同编写。特别感谢深圳疯壳团队的各位小伙伴们，他们为本书的编写提供了可靠的技术支撑与精神鼓励。此外，还要感谢西安电子科技大学出版社的工作人员，正是他们的支持才有本书的出版。

关于本书的源码，读者可以通过 https://www.fengke.club/GeekMart/su_f90xheDAs.jsp 社区论坛免费下载。由于时间仓促，尽管作者认真校验过本书的所有内容，但难免还有一些纰漏，读者可通过社区论坛与作者互动。

编者

2018 年 6 月



目 录

CONTENTS

第 1 章 开发前的准备	001
1.1 Android 系统移植准备及简介	001
1.1.1 Android 系统架构	001
1.1.2 Android 子系统	003
1.1.3 Android 应用程序开发过程	004
1.1.4 Android 源码开发过程	004
1.1.5 Android 系统移植	004
1.1.6 Linux 内核版本	005
1.1.7 Android 驱动程序开发过程	006
1.2 开发环境搭建	007
1.2.1 从零开始搭建 Ubuntu 开发环境	007
1.2.2 安装开发环境	012
1.2.3 安装编译工具和源代码	016
1.3 使用 Git 管理源代码	025
1.3.1 用 Git 命令初始化代码	025
1.3.2 用 smartgit 工具进行图形化管理	026
1.4 常用编译命令	030
1.5 搭建开发板的测试环境	033
第 2 章 开发基础知识	038
2.1 BootLoader 简介	038
2.1.1 BootLoader 的概念	038
2.1.2 U-Boot 简单分析	038
2.2 Kernel 简介	044
2.2.1 内核基本概念	044
2.2.2 内核理论基础——如何阅读内核源码	048
2.2.3 内核理论基础——设备树 DTS	065
2.2.4 内核理论基础——定时器	075
2.2.5 内核理论基础——内核异常中断	116

2.2.6 内核理论基础——SMP	134
2.2.7 内核理论基础——Linux 驱动编写中的并发控制	176
2.3 常用工具及命令	187
2.3.1 ADB	187
2.3.2 Logcat	189
2.3.3 grep 命令介绍	192
2.4 系统调试技巧	197
2.4.1 Android 的调试及 Log 机制详解	197
2.4.2 Linux 内核常用的调试方法	279
第 3 章 开发实战	283
3.1 PinCtrl(Pin Control) subsystem 子系统	283
3.1.1 PinCtrl 子系统介绍	284
3.1.2 PinCtrl 源代码简单介绍	304
3.1.3 PinCtrl 总结	319
3.2 内核 Kernel 移植	319
3.2.1 启动界面的更换	319
3.2.2 Linux 引导过程	320
3.2.3 GPIO 驱动	322
3.2.4 设备按键驱动	324
3.2.5 触摸屏驱动	328
3.2.6 Wi-Fi 驱动	331
3.2.7 G-sensor 驱动	333
3.3 Android 系统定制	338
3.3.1 Android HAL 硬件抽象层简介	338
3.3.2 GPS Android HAL 实现	345
3.3.3 Android 定制(LED 灯的 Android 程序)——硬件驱动程序及测试方法	348
3.3.4 Android 定制(LED 灯的 Android 程序)——增加 HAL 接口访问硬件驱动程序	350
3.3.5 Android 定制(LED 灯的 Android 程序)——编写服务的 JNI 方法	354
3.3.6 Android 定制(LED 灯的 Android 程序)——编写 Framework 层 Java 服务	357
3.3.7 Android 定制(LED 灯的 Android 程序)——编写通过 Manager 访问硬件服务	360
3.4 Android 应用开发	365
3.4.1 Android APP 基本结构	365
3.4.2 如何快速开发自己的 Android APP	373
参考文献	376

第1章 开发前的准备

Android 是一个支持多种移动设备的开源软件堆栈以及对应的由 Google 公司领导的开源项目。Android 的开放源代码项目(AOSP)代码库可从 <https://source.android.com> 网站上获得,供读者创建定制的 Android 堆栈版本,将设备和相应的配件移植到 Android 平台,同时确保读者的设备符合兼容性要求。

此外,Android 平台不存在任何行业参与者一手限制或控制其他参与者创新的情况,Android 是一个开放的平台。所以,我们才可以针对消费类商品打造一个完整的高品质操作系统,并支持对源代码进行定制和移植,而且不需要向 Google 公司交一分钱。Android 系统是一个基于 Apache License 软件许可的开源手机操作系统,底层以 Linux 操作系统为内核,可以直接从 Android 的官方网站上下载最新的 Android 源码和相关开发工具包。

由于 Android 原生的代码不可能支持所有设备,因此想要在自己的设备(包括手机、智能电视、平板电脑、车载系统等)上运行某一个分支的 Android,就需要定制开发一些应用或者驱动程序,使得 Android 可以识别相应设备中的硬件设备(包括屏幕、蓝牙、Wi-Fi、音频、传感器等),这个为特定设备定制 Android 的过程可以叫做“移植”。在移植过程中做得最多的开发工作就是支持各种硬件设备。因为 Android 系统是基于 Linux 内核的,疯壳开发板的 Linux 内核是 3.10.49(版本号可以在内核的 Makefile 中看到),所以支持各种硬件的工作就变成编写 Linux 驱动程序了,因此讲移植就必须讲 Linux 驱动开发。

如果开发者想要有自己的启动界面,也可以叫 Launcher(如 MIUI、EMUI、Mifavor),或者定制的应用程序,则必须进行 Android 应用程序开发。这里推荐 Google 公司指定的应用开发工具 Android Studio。有时候为了开发方便,开发者也可以直接在 Android 源代码中直接开发应用程序,这样可以省去另外搭建环境的麻烦,而且可以随着 Android 源代码一起编译发布。

1.1 Android 系统移植准备及简介

1.1.1 Android 系统架构

Android 开发可以自由实现设备规格和驱动程序。硬件抽象层 (HAL) 提供了一种用于在 Android 平台堆叠和硬件之间创建软件的标准方法。Android 操作系统也是开放源代码系统,因此它可以贡献自己的接口。

为确保设备能够保持较高的质量水平并提供一致的用户体验,每部设备都必须通过兼



容性测试套件(CTS)中的测试。CTS 可验证设备是否符合一定的质量标准,此类标准可确保应用稳定运行并提供良好的用户体验。

在将 Android 移植到硬件之前,我们首先需要花点时间从更高层面上了解 Android 系统架构。由于驱动程序和 HAL 会与 Android 进行交互,因此了解 Android 的工作原理可帮助我们浏览 Android 开放源代码项目(AOSP)源代码树中的多个代码层。

1. 应用框架

应用框架最常被应用开发者使用。作为硬件开发者,应该非常了解开发者 API(应用程序编程接口),因为很多此类 API 都可直接映射到底层 HAL 接口,并可提供与实现驱动程序相关的实用信息。

2. Binder IPC

Binder 进程间通信(IPC)机制允许应用框架跨越进程边界并调用 Android 系统服务代码,从而使得高级框架 API 能与 Android 系统服务进行交互。在应用框架级别,开发者无法看到此类通信的过程,但一切似乎都在“按部就班地运行”。

3. 系统服务

应用框架 API 所提供的功能可与系统服务通信,以访问底层硬件。服务是集成的模块化组件,例如窗口管理器、搜索服务或通知管理器。Android 包含两组服务:“系统”(诸如窗口管理器和通知管理器之类的服务)和“媒体”(与播放和录制媒体相关的服务)。

4. 硬件抽象层

硬件抽象层(HAL)会定义一个标准接口以供硬件供应商实现,并允许 Android 忽略较低级别的驱动程序实现。借助 HAL 可以顺利实现相关功能,而不会影响或无需更改更高级别的系统。HAL 实现会被封装成模块(.so)文件,并会由 Android 系统适时地加载。必须为产品所提供的特定硬件实现相应的 HAL。Android 并不要求 HAL 实现与设备驱动程序之间进行标准交互,因此我们可以自由地根据具体情况执行适当的操作。不过,要使 Android 系统能够与硬件正确互动,必须遵守各个针对特定硬件的 HAL 接口中的定义。

Android 系统架构如图 1-1 所示。



图 1-1

1.1.2 Android 子系统

Android 是一个庞大的开放的嵌入式系统，它不仅可以实现基本的打电话、发信息的功能，还可以实现更复杂的多媒体处理、2D 和 3D 游戏处理、信息感知处理等。图 1-2 所示的硬件抽象层组件列出了一些 Android 所支持和实现的子系统。

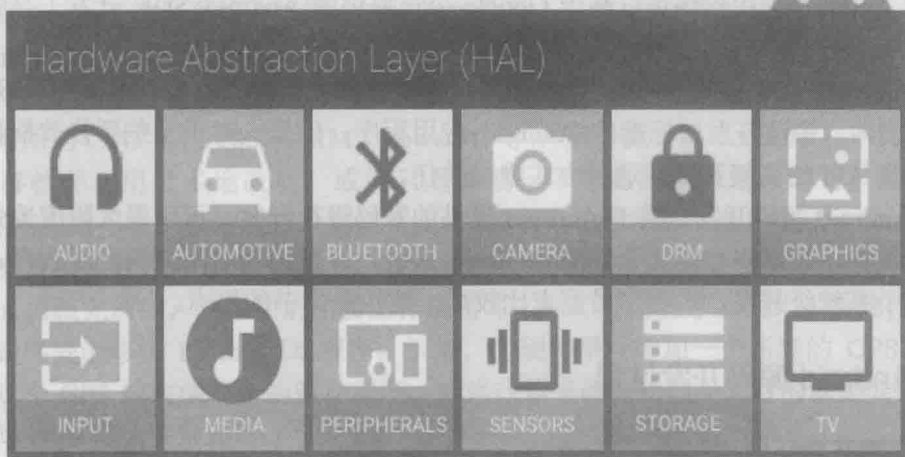


图 1-2

Android 的子系统主要包含以下几种：

(1) Android RIL。RIL(Radio Interface Layer)子系统即无线电接口系统，主要用于管理用户的电话、短信、数据通信等相关功能。

(2) Android Input。Input 子系统用来处理所有来自用户的输入数据，如触摸屏、声音控制物理按键和各种 sensor 设备。

(3) Android GUI。GUI 即图形用户接口，它用来负责显示系统图形化界面，主要是让用户和系统进行交互。Android 的 GUI 系统和其他各子系统密切相关，是 Android 中最重要子系统之一。例如：通过 OpenGL 库处理 3D 游戏，通过 SurfaceFlinger 来重叠几个图形界面。

(4) Android Audio。Audio 为 Android 的音频处理子系统，主要用于音频方面的数据流传输和控制功能，也负责音频设备的管理。Android 的 Audio 系统和多媒体处理紧密相连，如视频的音频处理和播放、电话通信及录音等。

(5) Android Media。Media 为 Android 的多媒体子系统，它是 Android 系统中最庞大的子系统，与硬件编解码、OpenCore 多媒体框架、Android 多媒体框架等相关，如视频播放器、Camera 摄像预览等。

(6) Android Connectivity。Connectivity 为 Android 的连接子系统，它是 Android 设备的重要组成部分，包括一般的网络连接，如以太网、Wi-Fi、蓝牙连接、GPS 定位、NFC 等。

(7) Android Sensor。Sensor 为 Android 的传感器子系统，它为当前设备提供了更有效的交互性，并且给一些应用程序和应用体验提供了技术支持。传感器子系统和手机的硬件



设备紧密相关，如 g-sensor 陀螺仪、proximity 距离感应器、magnetic 磁力传感器等。

1.1.3 Android 应用程序开发过程

Android 应用程序开发是基于 Android 架构提供的 API 和类库编写应用程序的，这些应用程序是完全的 Java 代码程序(也可以用 C/C++开发相应的 library 供 Java 调用)，它们构建在 Android 系统提供的 API 之上。

开发 Android 应用程序可以基于 Google 公司提供的 Android SDK 开发工具包，也可以直接在 Android 源码中进行编写。Android 应用程序开发方式如下：

(1) Android SDK 开发。它提供给程序员一种最快捷的开发方式，基于 IDE 开发环境和 SDK 套件，快速开发出标准的 Android 应用程序。但是，对于一些要修改框架代码或基于自定义 API 的高级开发，这种方式难以胜任。

(2) Android 源码开发。基于 Android 提供的源码进行开发，可以最大限度地体现出开源的强大优势，让用户自定义个性的 Android 系统，开发出与众不同的应用程序。这种方式更适合于系统级开发，对程序员要求比较高，这也是本书的重点。

1.1.4 Android 源码开发过程

1. 搭建开发环境

推荐使用单独一台电脑安装 Ubuntu 12.04 系统来下载和编译 Android 源代码。如果使用虚拟机编译，则推荐机器内存最少为 8 G。作者也在网站上为读者准备了一个可以编译整个 Android 系统(包括 Uboot、Kernel 和 Android)的虚拟机开发环境(最少配置 4G 内存)，读者可以下载下来直接使用，唯一缺点就是编译 Android 时有点慢。

2. 下载 Android 源码

受益于 Android 的开源特点，Android 源码中包含大量的技术知识，可以在阅读源码过程中更深入地了解 Android 系统的奥秘，为我们编写更高效、更有特点的应用程序打下基础。同时，Android 的源码中提供的应用程序示例、设计模式、软件架构为将来编写大型应用程序提供了经验。本书提供了相应的适合开发板的源码，大家可以在此基础上完成编译和调试。

3. 配置开发环境

为了编译 Android 源码，Ubuntu 系统可能需要安装很多工具包。为了代码的阅读，也需要特别配置 Samba 来和 Windows 系统共享代码，以便阅读。

4. 编译 Android 源码

通过编译 Android 源码，生成开发环境及目标系统，从而为系统底层开发、系统定制与优化做准备。通过分析编译过程，可以学习到大型工程的代码管理与编译方法。

1.1.5 Android 系统移植

Android 系统的移植主要分为两个部分：系统移植和应用程序移植。应用程序移植是

指将一些通用的应用程序,如串口程序、GPS 程序、浏览器等,移植到某一特定的硬件平台上。由于不同硬件平台有些许差异,故 Android SDK API 可能也有所不同(部分硬件厂商会更改部分 Android SDK API 来适应自身的硬件,当然也许这样兼容性会变差),或者将应用程序从低版本移植到高版本的 Android 上。为了保证应用程序可以在新的硬件平台上正常运行,需要对源码进行一些修改,但是如果没有或无法获取源码,则只能重新在新的平台上实现了。一般的 Android 应用程序移植并不涉及驱动和 HAL 程序库的移植,因为 Android 应用程序的移植并不在本书的讨论范围内,所以本书后面所说的 Android 移植通常都是指 Android 操作系统的移植(包括 Linux 驱动、HAL 程序库的移植)。

Android 系统的移植是指让 Android 操作系统在某一特定的硬件平台上运行。一个操作系统在特定硬件平台上运行首先必须要支持硬件平台的 CPU 架构。Linux 内核本身已经支持很多常用的 CPU 架构(x86、ARM、PowerPC 等),基于此,将 Android 在不同的 CPU 架构之间移植并不用过多地改动,或许只是调整一下相应的编译选项即可。如果要想 Android 在不同的平台上正常运行,则只是支持 CPU 架构还不够,必须要让 Android 可以识别平台支持的各种硬件外设(如声卡、显示设备、触摸设备、网络设备等),这些工作都是由 Linux 内核完成的,这里面的主要工作就是写 Linux 驱动程序。因此,系统移植的最主要工作最终变成移植 Linux 驱动程序。例如,为硬件平台增加一个新型的 GPS 模块,就需要为这个 GPS 模块编写新的驱动程序,或修改原来的驱动程序,目的就是要使 Linux 内核可以与 GPS 模块正常通信。

Android 系统增加了一个属于它自己特有的硬件抽象层(HAL, Hardware Abstraction Layer),为了方便,本书后面都用 HAL 表示硬件抽象层。HAL 不应该算是完全的驱动程序,只是一个普通的 Linux 程序库(.so 文件),它为 Android SDK 提供访问 Linux 驱动的媒介。也就是说,Android 系统并不像其他的 Linux 系统一样由应用程序可以直接访问驱动,而是中间通过 HAL 隔离。Google 这样设计的原因是:由于 Linux 内核是基于 GPL 开源协议的,而很多驱动厂商不愿意开放源代码,所以增加 HAL 将 Linux 驱动的业务逻辑放在 HAL 层,这样处理 Linux 驱动开源协议就会只开源非核心源代码并放在 Linux 驱动中,从而绕开 GPL 协议。

Android 移植在很大程度上是 Linux 内核的移植,Linux 内核移植主要就是移植驱动程序。不同 Linux 内核版本的驱动程序不能通用,需要重新修改源代码,并在新的 Linux 内核下重新编译才能运行在新的 Linux 内核版本中。Android 版本和 Linux 版本没有直接的联系,无论哪个 Android 版本,它的 Linux 内核版本都可以是 Linux 2.6 或 Linux 3.0,不同的可能是小版本号。

1.1.6 Linux 内核版本

Linux 内核版本有两种:稳定版和开发版。内核版本号由三组数字组成,其中第一组数字为目前发布的内核主版本;第二组数字若为偶数则表示稳定版本,若为奇数则表示开发中版本;第三组数字为错误修补的次数。

例如“2.6.18-128.ELsmp”中:第一组数字“2”为主版本号;第二组数字“6”为次版本号,表示稳定版本(因为有偶数);第三组数字“18”为修订版本号,表示修改的次



数。头两组数字合在一起可以描述内核系列，如稳定版的 2.6.0 表示它是 2.6 版内核系列。

“128”是这个当前版本的第 5 次微调 patch，而 ELsmp 指出了当前内核是为 ELsmp 特别调校的 EL(Enterprise Linux)；“smp”表示支持多处理器，即该内核版本支持多处理器。读者可以到网站 <http://www.kernel.org> 浏览详细的 Linux 内核版本信息。

1.1.7 Android 驱动程序开发过程

由于 Linux 的内核版本更新很快(稳定版本 1 至 3 月更新一次，升级版本 1 至 2 周更新一次)，每一次内核变化就意味着 Linux 驱动改变(至少驱动程序也要在新的 Linux 内核下保证编译通过)，所以 Linux 内核的不断变化对从事 Linux 驱动开发的程序员有较大的影响。但是不管学习哪个 Linux 版本的驱动，方法和步骤是基本相同的，所以对于学习驱动的版本影响有限，只要掌握了一个版本的驱动开发，就能够举一反三。

学习 Linux 驱动开发只有 Linux 内核还不够，还必须有一个可以开发 Linux 驱动的操作系统环境，并可以基于这个环境测试驱动程序。因为 Google 在测试 Android 源码时使用的是 Ubuntu Linux，因此，强烈建议使用 Ubuntu Linux 12.04 或以上版本来开发、编译、测试 Linux 驱动。本书所有的源码都是基于 Ubuntu Linux 12.04 编译和测试的。疯壳网站会提供相应的 Virtual Box 编译虚拟机映像文件(Ubuntu Linux 12.04，内存为 4 GB，登录名/密码为 fengke/fengke)，并且已经配置好了 Android 和 Linux 驱动编译环境。读者可以很方便地基于这个虚拟机开发环境来编译和运行本书所讲到的所有例子。

为了测试 Linux 驱动在 Android 中的运行效果，最好准备一块开发板。当开发完驱动程序后，需要在支持 Android 的开发板上测试驱动程序以便正常运行。本书建议采用疯壳的 Android 开发板，这样可以完整无误地支持本书所有的例子程序。

嵌入式设备的硬件由 CPU、存储器和各种外设组成。随着技术的飞速发展，芯片的集成度也越来越高，往往 CPU 内部就集成了存储器和外设适配器，这样的芯片也称为 SOC。ARM、PowerPC、MIPS 等处理器都集成了 UART、USB 控制器、SDRAM 控制器等，有的处理器还集成了片内 RAM 和 Flash。

驱动所指的对象是存储器和外设，不是针对 CPU 核的。Linux 将存储器和外设分为三大类。

(1) 字符设备：以字节为最小访问单位的设备，一般通过字符设备文件来访问字符设备驱动程序。字符驱动程序负责驱动字符设备，这样的驱动通常支持 open、close、read、write 系统调用，应用程序可以通过设备文件(比如/dev/ttySAC0 等)来访问字符设备(串口)。

(2) 块设备：以块(一般 512 字节)为最小传输单位的设备，常见的块设备包括硬盘、Flash、sd 卡。大多数 Unix 系统中，块设备不能按字节处理数据；而在 Linux 系统中则允许块设备传送任意数目的字节。

块设备的特别之处有：

① 操作硬件接口的实现方式不一样。块设备驱动程序是先将用户发来的数据组织成块，再写入设备的；或从设备中读出若干块数据，再从中挑出用户需要的数据的。

② 数据块上的数据可以有一定的格式。通常在块设备中按一定的格式存放数据，不同的文件系统类型就是定义这些格式的。内核中，文件系统的层次位于块设备驱动程序上

面的,这意味着块设备驱动程序除了向用户层提供与字符设备一样的接口外,还要向内核其他部件提供一些接口,这些接口用户看不到,但是可以使用这些接口在块设备上存放文件系统,挂载块设备。

块设备与字符设备的区别仅仅在于驱动向内核提供的接口不一样,而向用户层提供的接口是一样的。

(3) 网络接口:可以是一个硬件设备,如网卡;也可以是纯软件的设备,比如回环接口(lo)。一个网络接口负责发送和接收数据报文。网络驱动程序不同于字符设备和块设备。内核提供了一套和数据包传输相关的函数,而不是普通的系统调用(open/write)。

Linux 驱动开发的难点是什么,有什么好的方法来克服?

Linux 内核对各种设备的驱动开发提供了完善的框架支持,对应某个驱动,把对外的接口弄清楚即可。打个比方,一个设备可能不同的 OS 上需要支持,比如 FreeBSD/Windows 等,每个 OS 都有自己定义的接口,设备的驱动定义好与这些 OS 接口的连接,剩下的就是设备本身的特性管理以及驱动接口中对设备管理函数的调用,比如寄存器访问、配置管理、缓冲区管理、数据收发等,比较重要的是中断和同步的控制,要避免数据处理时的死锁。

比如网卡驱动,基本的要求是提供内核需要的接口,这样网卡驱动才能挂接到系统中,剩下的就是接口需要调用网卡驱动的内部函数,来对网卡进行控制、数据收发和管理等。

Linux 支持的设备种类繁多,不可能所有的都能掌握,即使是某一子系统也只能熟悉,因为同类设备还有许多自由的特性。写驱动的步骤可以概括如下:

(1) 阅读设备规范,对设备的运行机理有所了解。为了减少干扰,不考虑要支持的 OS,独立于 OS 考虑基本的功能如何实现。

(2) 参考同类设备在 Linux 内核中的驱动架构。

(3) 提供基本的 Linux 设备驱动接口和实现设备的基本功能,比如网卡收发小数据量。

(4) 在性能上逐步提示,比如网卡传输的数据量加大、中断及时处理、避免死锁等。

(5) 对边界条件进行完善,网卡驱动可保证特殊大小的数据包完整传输。

(6) 对设备进行更高级控制的支持,比如网卡支持 ethtool 等工具。

(7) 反复调试、改进和优化。

1.2 开发环境搭建

1.2.1 从零开始搭建 Ubuntu 开发环境

我们为大家准备了一个安装好的不带有任何编译工具包并基于免费的 VirtualBox 的 Ubuntu12.04 原始虚拟机,名字叫 ubuntu-fengke-raw.vdi,初学者可以在此基础上完成编译环境的搭建工作。下面开始讲解如何设置虚拟机,复制 ubuntu-fengke-raw.vdi 到当前目录并改名为 ubuntu-fengke-rk3128.vdi(为什么复制一个新的 vdi 文件?这样做的目的是在接下来对虚拟机的操作中如果出现了错误可以推翻重来)。



1. 虚拟机创建

虚拟机创建的步骤如下：

(1) 在图 1-3 中点击“New”按钮新建一个 Virtual Machine，然后点击“Next”按钮。

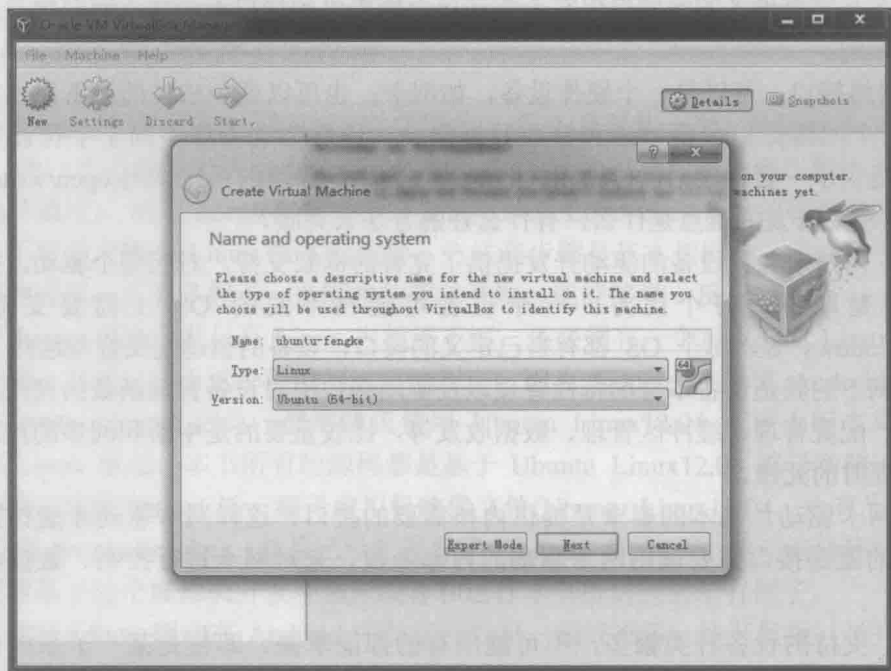


图 1-3

(2) 在图 1-4 所示的界面中设置内存大小为 4096 MB(4 GB)，然后点击“Next”按钮。

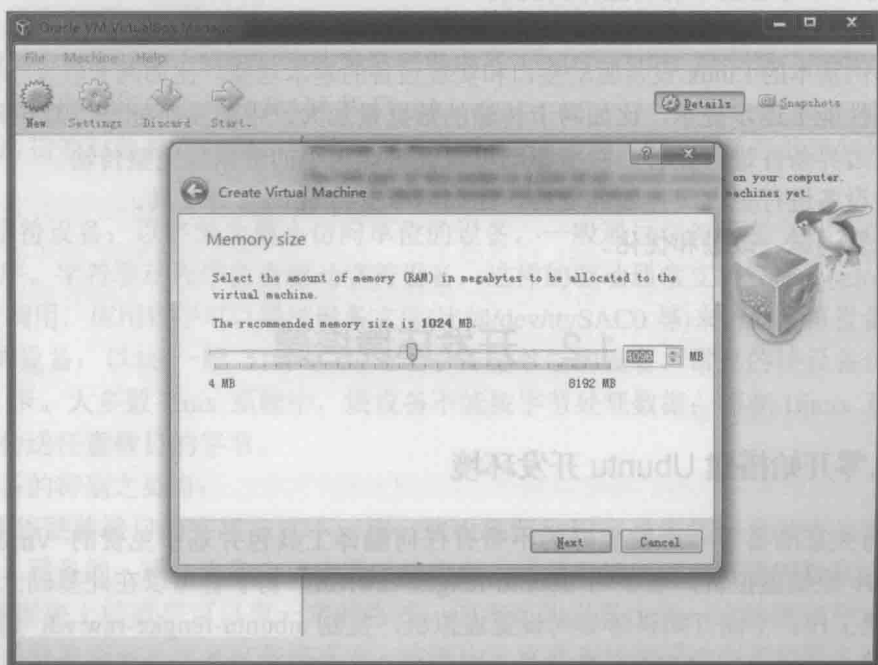


图 1-4

(3) 如图 1-5 所示, 选择名字是 ubuntu-fengke-rk3128.vdi 已经存在的虚拟硬盘, 并点击“Create”按钮。



图 1-5

(4) 最终生成一个如图 1-6 所示的新的虚拟机 ubuntu-fengke。

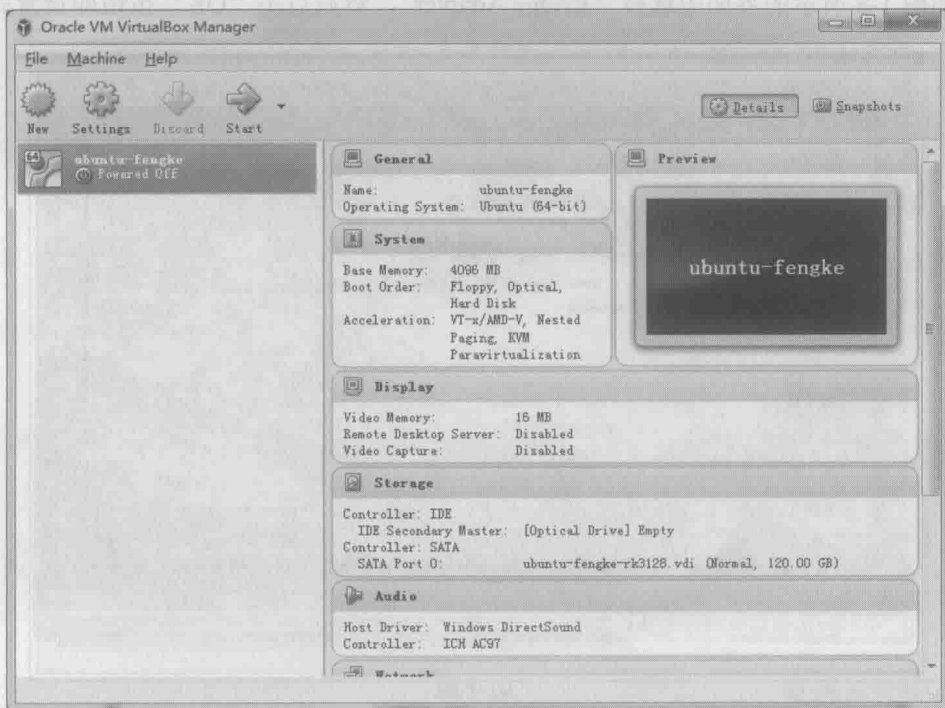


图 1-6



2. 网络设置

为了能够和主机共享网络，这里将网络设置成 bridge 模式。具体步骤为在图 1-7 所示的界面中点击“Network”。

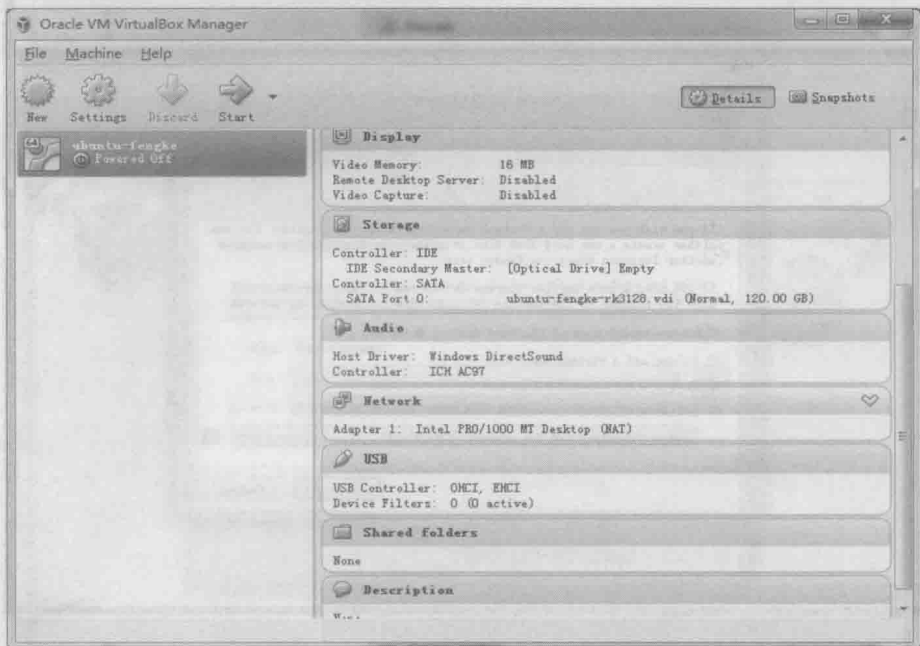


图 1-7

在如图 1-8 所示的界面中选择“Bridge Adapter”，然后点击“OK”按钮即设置完成。

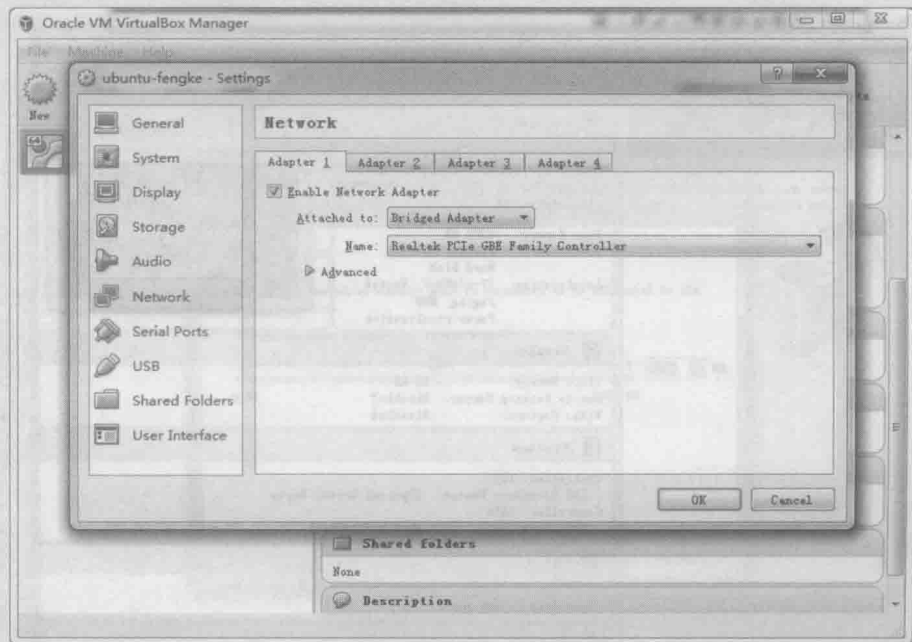


图 1-8