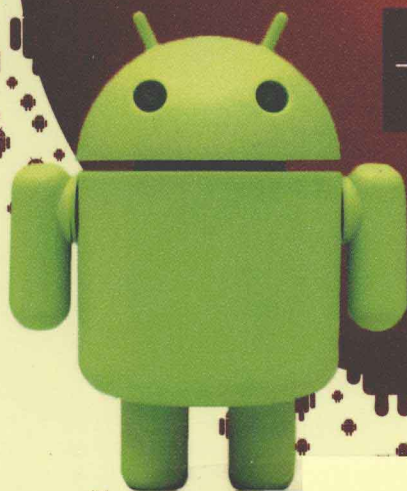


Android

系统级深入开发

——移植与调试

◎韩超 梁泉 著◎



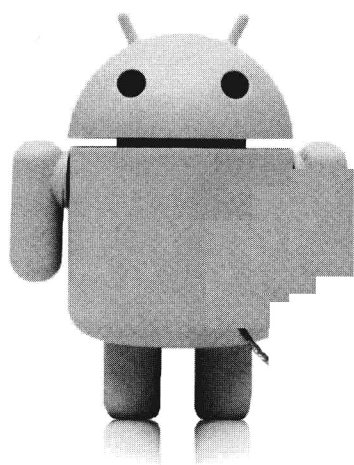
从Android开源工程到产品的开发宝典
全面介绍Android中与硬件相关的子系统
按照驱动程序和硬件抽象层两方面把握移植要点
高效粘合Linux系统经验和移动设备应用场景
以三种硬件平台为参考，参考开发环境宜于获得



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
<http://www.phei.com.cn>

Android

系统级深入开发



——移植与调试

◎韩超 梁泉 著◎

电子工业出版社

Publishing House of Electronics Industry

北京•BEIJING

内 容 简 介

本书是一本全面介绍 Android 系统级开发的作品,全书以移植和调试为重点。Android 具有一个庞大的软件系统,任何开发者都难以掌握系统的每一个细节。因此,如何高效地理解和开发系统,就成了 Android 系统级别工程师面对的主要问题。本书作者以实际的开发经验为基础,以软件工程思想为指导,完成了本书。本书介绍了从 Android 开源工程到一个基于实际硬件产品中的主要工作,一方面让读者清晰把握各个子系统的架构,另一方面让读者把握移植这个开发核心环节的要点。

本书适合 Linux 开发人员、移动设备开发人员、Android 系统框架层和底层开发人员、有意图深入学习 Android 的人员、以及从事手机研发的读者阅读

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有,侵权必究。

图书在版编目(CIP)数据

Android 系统级深入开发:移植与调试/韩超,梁泉著. —北京:电子工业出版社,2011.2
ISBN 978-7-121-12609-3

I. ①A… II. ①韩… ②梁… III. ①移动通信—携带电话机—应用程序—程序设计 IV. ①TN929.53

中国版本图书馆 CIP 数据核字(2010)第 249817 号

责任编辑:李 冰

印 刷:北京东光印刷厂

装 订:三河市皇庄路通装订厂

出版发行:电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本:787×1092 1/16 印张:24 字数:550 千字

印 次:2011 年 2 月第 1 次印刷

印 数:4000 册 定价:55.00 元

凡所购买电子工业出版社图书有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系,联系及邮购电话:(010) 88254888。

质量投诉请发邮件至 zlts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线:(010) 88258888。

前言

Android 系统是目前最为流行的手机系统之一，本书作者在推出了全面介绍 Android 系统的《Android 系统原理及开发要点详解》一书，同时出版了繁体版，并将版权成功输出到韩国，韩文版将在 2011 年 7 月出版。在本书出版之后，笔者切实感到越来越多的开发人员和企业在关注 Android 系统的各个方面。

Android 系统是开源的，它的一个比较大的优势就是可以移植到各个不同的硬件平台上。“移植”是 Android 系统的精华所在，也是 Android 开发中的难点兼重点。

本书《Android 系统级深入开发——移植与调试》出版的主要目的是分享开发经验，帮助众多开发者快速地掌握 Android 系统在不同硬件平台移植的方法。帮助工程师以最快的速度、最小的开销、最轻的工作量，让 Android 系统高效地运行在更多的硬件上。

本书特点

本书紧紧抓住了 Android 系统移植与调试的主题，从开发者的角度出发，其特点主要包括以下几个方面：

- 本书使用的代码以 Android 的开源工程为主，硬件也是比较常见的设备，保证读者可以很容易地获得开发环境。
- 根据 Android 系统自身的固有特点，从 Linux 内核中的驱动和硬件抽象层两个着手点作为切入点。
- 按照 Linux 的开发思路，从驱动程序的角度出发，让具有 Linux 开发经验的工程师，可以更快地熟悉主要用于移动设备开发的 Android 系统。
- 从模拟器、高通的 MSM 平台、德州仪器的 OMAP 平台这三种硬件平台出发，全面介绍，不以偏概全，既把握共同点，也突出差异。
- 对于 Android 中规模和难度各不相同的子系统，抓住它们和硬件相关的共同点，采取同样的格式和思路进行介绍，体现了“从特殊到一般”的理念。
- 简要介绍各个子系统的框架，并列代码路径，对移植部分的主要调用部分加强提示，让读者更全面地把握系统。

本书内容

本书分成 24 章，各章的内容如下所示：

第 1 章和第 2 章：概要部分	介绍 Android 的系统的大结构、移植的主要工作，开发环境的构建方法
第 3 章：全书的总纲	全面介绍 Linux 系统的驱动程序，突出介绍 Android 中硬件抽象层的不同实现方式，展示 Android 移植的各个方面

续表

第 4 章至第 6 章：Linux 内核方面	主要介绍用于模拟器的 Goldfish、MSM 的 mahimahi 平台和 OMAP 的 Zoom 平台的 3 种 Linux 内核，介绍了不同体系结构的移植，与硬件无关的 Android 专用驱动，并简单介绍了各个硬件设备的驱动程序
第 7 章和第 8 章：基本 GUI 的系统	包括显示输出和用户输入两个部分，是每个系统最优先移植的部分
第 9 章：传感器系统	体现 Android 系统最经典的移植方式
第 10 章至第 12 章：多媒体的输入输出环节	是 Android 移植重点关注的部分
第 13 章至第 15 章：连接系统	介绍连接方面的无线局域网、蓝牙和定位三个子系统，前两者使用 Linux 中比较标准的方式，定位主要通过 GPS 进行介绍
第 16 章：电话系统	介绍电话系统，这也是用于移动设备的核心部分
第 17 章：OpenGL	作为显示加速的 OpenGL 3D 的移植，体现 Android 和业界标准规范的接轨
第 18 章和第 19 章：多媒体	用于多媒体加速的编解码单元的移植，第 18 章介绍编解码较下层的标准模块 OpenMax 插件，第 19 章介绍 OpenMax 插件在 Android 系统中的使用
第 20 章：位块复制	介绍位块复制部分，主要用于原始图像数据处理的加速
第 21 章至第 24 章：几个小系统	介绍与移动电话相关的警报器、光、振动器、电池信息这 4 个 Android 中较小系统的移植

本书读者

本书适合不同类型的读者群，不同类型的读者在学习的时候需要使用不同的方法。作者的建议如下：

- 对于熟悉 Linux 内核但不熟悉 Android 的开发者，应该以驱动程序作为切入点，通过硬件抽象层的实现，将 Linux 的各个驱动程序应用到 Android 系统中。
- 对于熟悉 Android 系统但不熟悉 Linux 内核的开发者，向下了解 Linux 内核，这样既可以更深入地了解 Android 系统的运作方式，又可以拓展自己的技术领域。
- 对于经验较多，希望深入研究 Android 系统的开发者，应该更关注开发的细节，了解移植中调试的要点。

本书作者

本书的规划和统筹由中国大陆的韩超完成，韩超常年工作在 Linux 和 Android 的开发一线，兼具产品和理论经验。本书内容来源于工作在不同领域 Android 和 Linux 开发者多年的经验。韩超和梁泉完成了本书内容的主要部分，众多不同规模的企业开发成果也为本书的编写提供了重要的素材。参与本书编写的还有崔海斌、于仕林、张宇、张超、赵家维、黄亮、沈桢、徐威特、杨钰、马若劼、曹道刚等。广大社区开发者也对本书的成稿作出了贡献。

目 录

第 1 章	Android 系统移植开发概述	1
1.1	Android 系统架构和生态系统	1
1.1.1	Android 软件系统	1
1.1.2	Android 的生态系统	3
1.2	Android 移植的概念和方法	4
1.3	Android 移植的主要工作	5
第 2 章	Android 源代码和开发环境	8
2.1	开发环境和工具	8
2.2	Android 的代码库	9
2.3	Android 系统代码和编译	9
2.3.1	获取 Android 源代码	9
2.3.2	Android 源代码结构	12
2.3.3	编译 Android 系统源代码	13
2.3.4	编译 Android 结果	13
2.4	Android Kernel 代码和编译	14
2.4.1	Goldfish 内核源代码	15
2.4.2	MSM 内核源代码	15
2.4.3	Omap 内核源代码	16
2.5	仿真器的运行环境	16
2.5.1	仿真器的运行	16
2.5.2	使用附加工具	18
第 3 章	Android 系统移植的结构和方法	20
3.1	Android 的 Linux 操作系统	20
3.1.1	标准的 Linux 操作系统	20
3.1.2	Android 对 Linux 内核的使用	22
3.1.3	Linux 内核空间到用户空间的接口	24
3.1.4	平台数据和平台驱动	34
3.2	Android 的硬件抽象层	35
3.2.1	硬件抽象层的地位和功能	35
3.2.2	硬件抽象层接口方式	36
3.3	Android 中各个部件的移植方式	41
3.4	辅助性工作和基本调试方法	41
3.4.1	移植的辅助性工作	42
3.4.2	调试的方法	46
第 4 章	Android 的 GoldFish 内核和驱动	56
4.1	GoldFish 内核概述	56
4.2	GoldFish 体系结构移植	58
4.3	GoldFish 的 Android 专用驱动和组件	59
4.3.1	wakelock 和 earlysuspend	59
4.3.2	staging 中的驱动程序	61
4.3.3	Ashmem 驱动程序	66
4.3.4	Alarm 驱动程序	67
4.3.5	pmem 驱动程序	67
4.3.6	ADB Garget 驱动程序	68
4.3.7	Android Paranoid 网络	68
4.4	GoldFish 的相关设备驱动	70
4.4.1	Framebuffer 的驱动程序	70
4.4.2	键盘的驱动程序	70
4.4.3	实时时钟的驱动程序	71
4.4.4	TTY 终端的驱动程序	71
4.4.5	NandFlash 的驱动程序	72
4.4.6	MMC 的驱动程序	72
4.4.7	电池的驱动程序	73
4.4.8	EAC 音频的驱动程序	73

第 5 章	Android 的 MSM 内核和驱动	74
5.1	MSM 处理器概述	74
5.1.1	MSM 概述	74
5.1.2	MSM 适用于 Android 的 Linux 内核的结构	77
5.2	MSM 体系结构的移植	79
5.3	MSM 的 Android 专用驱动和 组件	80
5.4	MSM 的 mahimahi 平台的 主要设备驱动	81
5.4.1	显示的驱动程序	81
5.4.2	触摸屏的驱动程序	82
5.4.3	按键和轨迹球的驱动程序	82
5.4.4	实时时钟的驱动程序	83
5.4.5	摄像头的驱动程序	83
5.4.6	无线局域网的驱动程序	83
5.4.7	蓝牙的驱动程序	84
5.4.8	DSP 相关的驱动程序	84
5.4.9	高通特有的组件相关内容	85
第 6 章	Android 的 OMAP 内核和 驱动	87
6.1	OMAP 内核概述	87
6.1.1	OMAP 概述	87
6.1.2	OMAP 适用于 Android 的 Linux 内核的结构	91
6.2	OMAP 体系结构的移植	92
6.2.1	OMAP 平台部分的移植	92
6.2.2	OMAP 处理器部分的移植	94
6.3	OMAP 的 Android 专用 驱动和组件	96
6.4	OMAP 的主要设备驱动	97
6.4.1	显示的驱动程序	97
6.4.2	摄像头和视频输出的 驱动程序	98
6.4.3	i2c 总线驱动程序	98
6.4.4	键盘的驱动程序	99
6.4.5	触摸屏的驱动程序	99
6.4.6	实时时钟的驱动程序	99
6.4.7	音频的驱动程序	99
6.4.8	蓝牙的驱动程序	100
6.4.9	以太网的驱动程序	100
6.4.10	DSP 的驱动程序	100
第 7 章	显示系统	101
7.1	显示系统结构和移植内容	101
7.1.1	Donut 及其之前显示 系统的结构	102
7.1.2	Eclair 及其之后显示 系统的结构	102
7.1.3	移植的内容	103
7.2	移植和调试的要点	104
7.2.1	Framebuffer 驱动程序	104
7.2.2	Donut 及其之前的硬件 抽象层	106
7.2.3	Eclair 及其之后的硬件 抽象层	107
7.3	显示部分模拟器的实现方式	112
7.3.1	Goldfish 的 framebuffer 驱动程序	112
7.3.2	默认的 Gralloc 模块的 实现	113
7.4	MSM 中的实现	119
7.4.1	MSM 的 framebuffer 驱动程序	119
7.4.2	MSM 的 Gralloc 模块 的实现	120
7.5	OMAP 中的实现	126
7.5.1	OMAP 的 framebuffer 驱动程序	126
7.5.2	OMAP 的用户空间的实现	128
第 8 章	用户输入系统	129
8.1	用户输入系统结构和 移植内容	129

8.1.1	用户输入系统的结构	129	10.3.1	用桩实现的 Audio 硬件 抽象层	172
8.1.2	移植的内容	131	10.3.2	提供 Dump 功能的 Audio 硬件抽象层	174
8.2	移植的要点	131	10.3.3	通用的 Audio 硬件 抽象层	177
8.2.1	input 驱动程序	131	10.4	MSM 系统的实现	178
8.2.2	用户空间的处理	134	10.4.1	Audio 驱动程序	178
8.2.3	移植需要注意的情况	139	10.4.2	Audio 硬件抽象层	180
8.3	模拟器中的实现	141	10.5	基于 OSS 和 ALSA 的 实现方式	183
8.3.1	驱动程序	141	10.5.1	OSS 驱动程序	183
8.3.2	用户空间的配置文件	141	10.5.2	基于 OSS 的硬件抽象层	184
8.4	MSM 中的实现	142	10.5.3	ALSA 驱动程序	185
8.4.1	触摸屏, 轨迹球和按键 驱动程序	142	10.5.4	基于 ALSA 的硬件 抽象层	186
8.4.2	用户空间的配置文件	144	第 11 章	视频输出系统	190
8.5	OMAP 中的实现	144	11.1	视频输出系统结构和移植 内容	190
8.5.1	触摸屏和键盘的驱动程序	144	11.1.1	视频输出系统的结构	191
8.5.2	用户空间的配置文件	146	11.1.2	移植的内容	192
8.6	虚拟按键的实现	146	11.2	移植和调试的要点	192
第 9 章	传感器系统	148	11.2.1	驱动程序	192
9.1	传感器系统结构和移植内容	148	11.2.2	硬件抽象层的内容	192
9.1.1	传感器系统的结构	148	11.2.3	上层的情况和注意实现	195
9.1.2	移植的内容	150	11.3	Overlay 硬件抽象层实现的 框架	199
9.2	移植和调试的要点	150	11.4	OMAP 系统的实现	200
9.2.1	驱动程序	150	11.4.1	OMAP 的视频输出 部分的驱动程序	200
9.2.2	硬件抽象层的内容	151	11.4.2	OMAP Overlay 硬件 抽象层	202
9.2.3	上层的情况和注意事项	153	第 12 章	照相机系统	205
9.3	模拟器中的实现	157	12.1	照相机系统结构和 移植内容	205
第 10 章	音频系统	162	12.1.1	照相机系统的结构	206
10.1	音频系统结构和移植内容	162			
10.1.1	音频系统的结构	162			
10.1.2	移植的内容	164			
10.2	移植和调试的要点	164			
10.2.1	Audio 驱动程序	164			
10.2.2	硬件抽象层的内容	164			
10.2.3	Audio 策略管理的内容	168			
10.2.4	上层的情况和注意事项	169			
10.3	通用的 Audio 系统实现	170			

12.1.2	移植的内容	207	14.2	移植和调试的要点	240
12.2	移植和调试的要点	207	14.2.1	驱动程序	240
12.2.1	Video for 4Linux 驱动程序	207	14.2.2	本地代码的配置部分	242
12.2.2	硬件抽象层的内容	210	14.2.3	上层的情况和调试方法	244
12.2.3	上层的情况和注意事项	215	14.3	MSM 系统的蓝牙实现	245
12.2.4	照相机系统的数据流 情况	219	14.3.1	驱动部分	245
12.3	Camera 硬件抽象层桩实现	222	14.3.2	用户空间的部分	247
12.4	MSM 平台的 Camera 实现	226	第 15 章	定位系统	248
12.4.1	MSM 平台的 Camera 驱动程序	226	15.1	定位系统的系统结构和 移植内容	248
12.4.2	MSM 平台的 Camera 硬件抽象层	227	15.1.1	定位系统的系统结构	248
12.5	OMAP 平台的 Camera 实现	228	15.1.2	移植的内容	250
12.5.1	OMAP 平台的 Camera 驱动程序	228	15.2	移植和调试的要点	250
12.5.2	OMAP 平台的 Camera 硬件抽象层	229	15.2.1	驱动程序	250
第 13 章	无线局域网系统	230	15.2.2	硬件抽象层	251
13.1	无线局域网系统结构和 移植内容	230	15.2.3	上层的情况和调试方法	253
13.1.1	无线局域网系统的结构	231	15.3	仿真器的 GPS 硬件适 配层实现	256
13.1.2	移植的内容	232	15.4	MSM 平台的 GPS 硬件 适配层实现	257
13.2	移植和调试的要点	232	第 16 章	电话系统	260
13.2.1	协议和驱动程序	232	16.1	电话系统结构和移植内容	260
13.2.2	用户空间的内容	233	16.1.1	电话系统的系统结构	260
13.2.3	上层的情况和调试方法	233	16.1.2	移植的内容	262
13.3	OMAP 系统的无线局域网 实现	235	16.2	移植和调试的要点	262
13.3.1	Linux 内核中的内容	235	16.2.1	驱动程序	262
13.3.2	用户空间的实现	236	16.2.2	RIL 实现库的接口	264
第 14 章	蓝牙系统	238	16.2.3	数据连接部分	266
14.1	蓝牙系统结构和移植内容	238	16.2.4	调试方法	267
14.1.1	蓝牙系统的结构	239	16.3	电话部分的 RIL 参考实现	268
14.1.2	移植的内容	240	16.3.1	端口初始化	268
			16.3.2	AT 命令处理流程	269
			16.3.3	Event 模块	270
			16.3.4	Modem AT 命令初始化	272
			16.3.5	请求和响应流程的处理	272
			16.3.6	特定命令类型的实现	274

第 17 章 OpenGL 3D 引擎.....	276	19.1.2 移植的内容.....	312
17.1 OpenGL 系统结构和移植内容.....	276	19.2 OpenCore 引擎的结构和插件.....	313
17.1.1 OpenGL 系统的结构.....	277	19.2.1 OpenCore 的结构.....	313
17.1.2 移植的内容.....	283	19.2.2 OpenCore 的 Node 插件机制.....	314
17.2 移植和调试的要点.....	283	19.2.3 OpenMax 部分的结构、实现和插件结构.....	316
17.2.1 OpenGL 移植层的接口.....	283	19.2.4 关于媒体输入输出类 MediaIO.....	322
17.2.2 上层的情况和 OpenGL 的调试.....	285	19.2.5 OpenCore Player 的视频显示部分插件.....	325
17.2 Android 软件 OpenGL 的实现.....	288	19.3 Stagefright 引擎的结构和插件.....	326
17.3 不同系统中的实现.....	290	19.3.1 Stagefright 系统结构.....	326
第 18 章 OpenMax 多媒体引擎.....	292	19.3.2 Stagefright 对 Android 中 OpenMax 接口的实现.....	328
18.1 OpenMax 系统结构和移植内容.....	292	19.3.3 MediaSource 插件机制.....	328
18.1.1 OpenMax 系统的结构.....	293	19.3.4 OpenMax 和 VideoRenderer 插件机制.....	330
18.1.2 Android OpenMax 实现的内容.....	297	19.4 OMAP 平台实现的插件.....	332
18.2 OpenMax 的接口与实现.....	297	19.4.1 OpenCore 的 OpenMax 插件.....	333
18.2.1 OpenMax IL 层的接口.....	297	19.4.2 OpenCore 的视频输出插件.....	334
18.2.2 OpenMax IL 实现的内容.....	302	19.4.3 Stagefright 的 OpenMax 和视频输出插件.....	336
18.2.3 Android 中 OpenMax 的适配层.....	302	第 20 章 位块复制系统.....	339
18.3 OMAP 平台 OpenMax IL 的硬件实现.....	304	20.1 位块复制结构和移植内容.....	339
18.3.1 TI OpenMax IL 实现的结构和机制.....	304	20.1.1 位块复制系统的结构.....	339
18.3.2 TI OpenMax IL 的核心和公共内容.....	306	20.1.2 移植内容.....	340
18.3.3 一个 TI OpenMax IL 组件的实现.....	307	20.2 移植和调试的要点.....	340
第 19 章 多媒体系统的插件.....	310	20.2.1 驱动程序.....	340
19.1 Android 多媒体相关结构与移植内容.....	310	20.2.2 硬件抽象层的内容.....	341
19.1.1 多媒体处理过程.....	311	20.2.3 上层的情况和注意事项.....	342
		20.3 MSM 平台中的实现.....	343

第 21 章 警报器——实时时钟系统	346	22.3.1 驱动程序	359
21.1 警报器——实时时钟结构和移植内容	346	22.3.2 硬件抽象层	359
21.1.1 警报器——实时时钟系统的结构	346	第 23 章 振动器系统	361
21.1.2 移植内容	347	23.1 振动器系统结构和移植内容	361
21.2 移植与调试的要点	348	23.1.1 振动器部分的结构	361
21.2.1 RTC 驱动程序	348	23.1.2 移植内容	362
21.2.2 Alarm 驱动程序	349	23.2 移植与调试的要点	363
21.2.3 上层的情况和注意事项	349	23.2.1 驱动程序	363
21.3 模拟器环境中的实现	351	23.2.2 硬件抽象层的内容	363
21.4 MSM 平台的实现	351	23.2.3 上层的情况和注意事项	364
第 22 章 光系统	354	23.3 MSM 中的实现	365
22.1 光系统结构和移植内容	354	第 24 章 电池系统	367
22.1.1 光系统部分的结构	354	24.1 电池系统结构和移植内容	367
22.1.2 移植内容	355	24.1.1 电池系统部分的结构	367
22.2 移植与调试的要点	356	24.1.2 移植内容	368
22.2.1 驱动程序	356	24.2 移植和调试的要点	368
22.2.2 硬件抽象层的内容	356	24.2.1 驱动程序	368
22.2.3 上层的情况和注意事项	357	24.2.2 上层的情况和注意事项	369
22.3 MSM 中的实现	359	24.3 模拟器中的实现	371

第1章

Android 系统移植开发概述

1.1 Android 系统架构和生态系统

1.1.1 Android 软件系统

Android 是一个包括操作系统、中间件和关键应用的移动设备软件堆。Android 是目前最流行的手机开发平台，依靠 Google 的强大开发和媒体资源，Android 成为众多手机厂商竞相追逐的对象。

Android 系统在推出后，逐渐完善和增加功能。从最初的版本发布后，又陆续发布了 Cupcake, Donut, Eclair, Froyo 等版本，发布的时候使用 Android 1.5、Android 2.0 等版本号标示，后面版本对前面的版本兼容，如表 1-1 所示。每一个版本具有不同的 API 级别，目前 Android 的 API 基本从 2 到 8，这个 API 级别通常是指 Android 平台的 Java 层的 API 的接口。

表 1-1 Android版本的升级

Android 的发布版本	名 称	API 级别
Android 1.1		2
Android 1.5	Cupcake	3
Android 1.6	Donut	4
Android 2.0	Eclair	5
Android 2.0.1	Eclair	6
Android 2.1	Eclair	7
Android 2.2	Froyo	8

作为一个开放式的移动设备的平台，Android 包含了众多的功能和庞大的代码。其代码基于 Linux 内核，在用户空间又分成本地代码（C 和 C++）和 Java 代码两种。从宏观的角度来看，Android 是一个开放的软件系统，它包含了众多的源代码。从下至上，Android 系

统分成 4 个层，如图 1-1 所示。

- 第 1 层：Linux 操作系统及驱动
- 第 2 层：本地代码框架和 Java 虚拟机
- 第 3 层：Java 框架
- 第 4 层：Java 应用程序

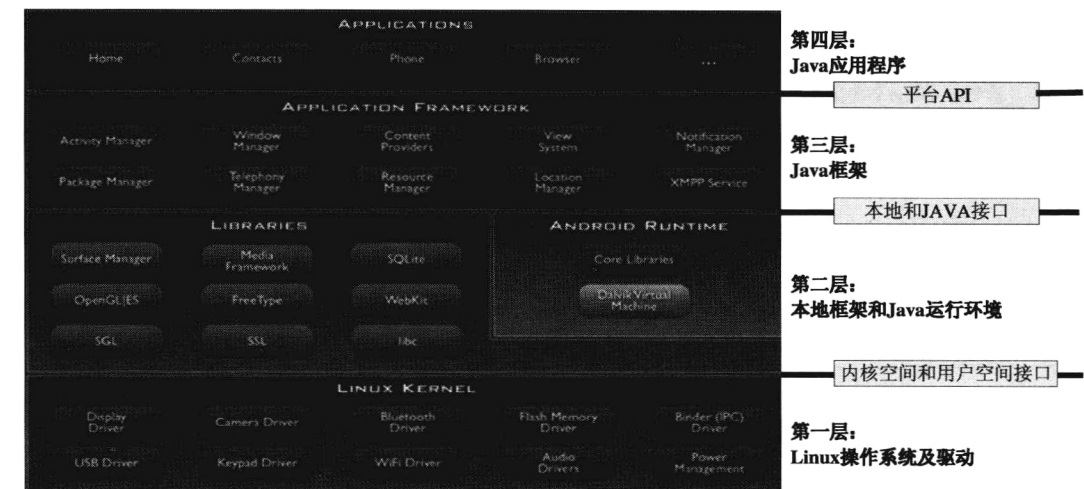


图 1-1 Android 软件系统架构

Android 的第 1 层由 C 语言实现，第 2 层由 C 和 C++ 实现，第 3、4 层主要由 Java 代码实现。

第 1 层和第 2 层之间，从 Linux 操作系统的角度来看，是内核空间与用户空间的分界线。

第 2 层和第 3 层之间，是本地代码层和 Java 代码层的接口。

第 3 层和第 4 层之间，是 Android 的系统 API 的接口。

由于 Android 系统需要支持 Java 代码的运行，这部分内容是 Android 的运行环境 (Runtime)，由虚拟机和 Java 基本类组成。这部分内容通常也可以认为是第 2 层的一个部分。

第 1 层次运行于内核空间，第 2、3、4 层运行于用户空间。

提示：通常情况下，可以将第 1 层视为 Android 的 Linux 内核，将第 2、3、4 层视为 Android 系统。

Android 几个层次的功能如下所示：

- Android 的操作系统

使用 Linux 2.6 内核，包括安全性、内存管理、进程管理、网络协议栈和驱动模型。Linux 内核也同时作为硬件和软件系统之间的抽象层。

- Android 本地库

包含一些 C/C++ 库, 这些库可以被 Android 系统中不同的组件使用, 它们通过 Android 应用程序框架为开发者提供服务。

- Android 运行环境

为 Java 的运行环境, 提供了 Java 编程语言核心库的大多数功能, 由 Dalvik Java 虚拟机和基础的 Java 类库组成。Dalvik 是 Android 中使用的 Java 虚拟机, Dalvik 被设计成一个可以同时高效地运行多个虚拟机实例的虚拟系统。

- Android 应用程序框架

包含了 4 种基本的组件 (活动、服务、广播接收器、内容提供者), 丰富的控件 (在 Android 中称为视图), 内容提供器, 资源管理器, 通知管理器, 活动管理等。

- Android 应用程序

一般由 Java 语言编写, 核心应用程序和系统一起发布, 包括: 桌面、电话、短信息、E-mail、日历、浏览器、联系人管理程序等。

1.1.2 Android 的生态系统

基于 Android 软件系统, 各个厂商可以实现自己的产品, 由于 Android 具有成熟、完整的软件系统, 各个厂商可以缩短自己的产品开发周期。

同时, Android 的应用程序开发者, 可以基于 Android 的平台 API 来开发软件, Android 的平台 API 是 Java 接口, 与具体的硬件无关, 甚至可以在没有硬件的情况下在仿真器环境中完成。

各个厂商生产产品和 Android 的软件开发者是相互促进的过程: 众多的 Android 软件可以让 Android 的各个设备具有更多的功能; 不同厂商制造的各种设备可以让 Android 软件具有更多的运行载体。

在众多的 Android 产品和 Android 软件的开发过程中, Google 可以提供自己的服务, 例如 Gtalk, Gmail, Google Search 等, 既促进 Android 平台的发展, 也让自己的业务拓展到各个基于 Android 的移动设备和其他产品。

OHA (Open Handset Alliance, 开放手机联盟) 是 Google 与 33 家公司联手为 Android 移动平台系统的发展而组建的一个组织。

Google 的 Android 系统是一个完全开放的系统, 也是一个完整的生态系统, 它分成了 3 个有机的组成部分:

- Android 源代码工程 (Android Open Source Project)

网址: <http://source.android.com/>

- Android 开发者 (Android Developer)

网址: <http://developer.android.com/>

- Android 市场 (Android Market)

网址: <http://www.android.com/market/>

Android 的生态系统如图 1-2 所示。

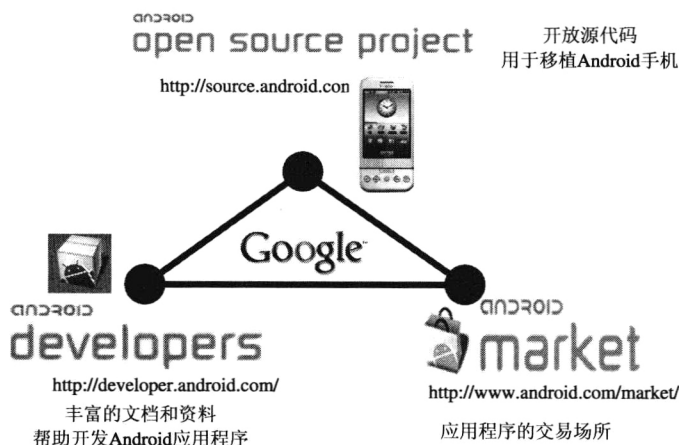


图 1-2 Android 的生态系统

1.2 Android 移植的概念和方法

在各个厂商开发基于 Android 系统的产品的时候，虽然有的时候也需要修改 Android 的框架，但是移植是其中的主要工作。

Android 系统本身是一个庞大的系统，移植并不需要精通 Android 的每一个部分，需要考虑的是 Android 系统的硬件抽象层（HAL）和 Linux 中的相关设备驱动程序。如图 1-3 所示。

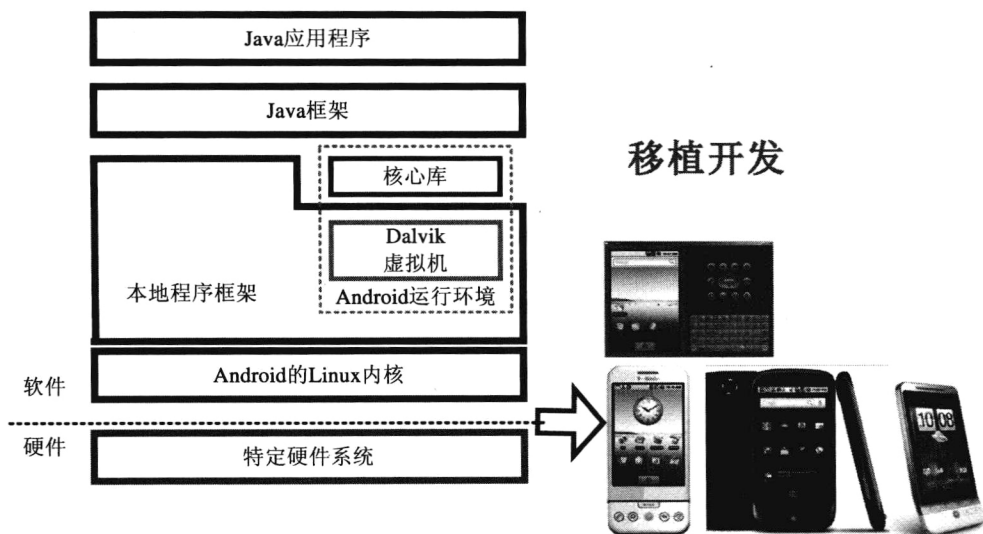


图 1-3 Android 移植的概念

基于 Android 系统的手机，有官方的 G1、Hero 和 Nexus One 等，其他的手机厂商也推出了几款 Android 手机。这些手机所使用的处理器和各种外围硬件各不相同，但是其使用的大

部分 Android 系统的软件都是相同的（包括本地框架、虚拟机、Java 框架和 Java 应用等部分）。

移植的目的就是为了改动较小的内容，支撑较为庞大上层的系统。同时由于硬件抽象层具有标准的接口，在各个不同的平台的实现中可以互相参考，虽然具体实现的内容不同，但是思路类似，可以相互参考。

1.3 Android 移植的主要工作

Android 系统的移植工作的目的是为了在特定的硬件上运行 Android 系统。在移植的过程中，把握关键要点，减少工作量是一个重要的方面。从工作的角度，通常的方法为，首先要熟悉硬件抽象层的接口，其次要集成和复用已有的驱动程序，主要的工作量在硬件抽象层的实现中。为了更好地理解和调试系统，也应该适当地了解上层对硬件抽象层的调用情况。

移植方面主要的工作有两个部分：

- Linux 驱动
- Android 系统硬件抽象层

Linux 中的驱动工作在内核空间，Android 系统硬件抽象层工作在用户空间，有了这两个部分的结合，就可以让庞大的 Android 系统运行在特定的硬件平台上。

Android 移植的主要工作如图 1-4 所示。

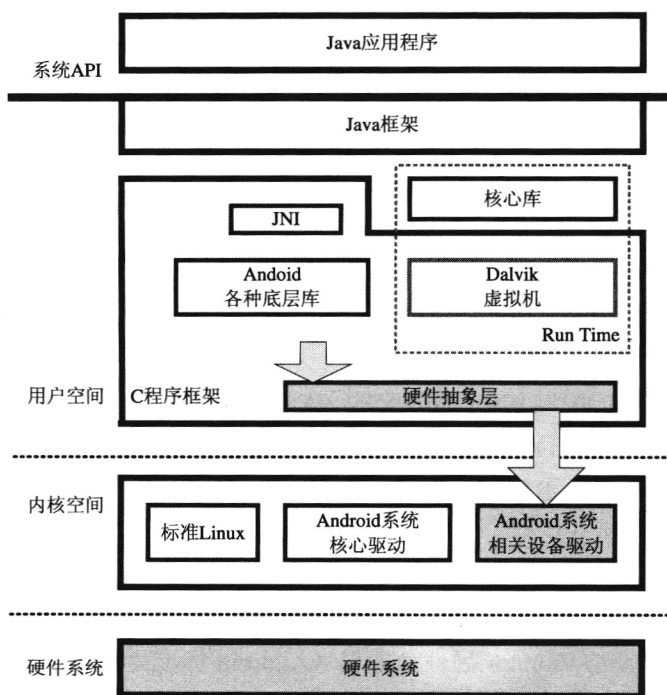


图 1-4 Android 移植的主要工作

在具有了特定的硬件系统之后，通常在 Linux 中需要实现其驱动程序，这些驱动程序通常是 Linux 的标准驱动程序，在 Android 平台和其他 Linux 平台基本上是相同的。主要的

实现方面是 Android 系统中的硬件抽象层 (Hardware Abstract Layer)，硬件抽象层对下调用 Linux 中的驱动程序，对上提供接口，以供 Android 系统的其他部分 (通常为 Android 本地框架层) 调用。

 **提示：**Android 硬件抽象层的接口是本地移植层的接口，不属于标准 API，不具有向前或者向后兼容性。

在 Android 系统需要移植的内容，主要包含了以下的各个部分：

- 显示部分 (Display)
包括 framebuffer 驱动+Gralloc 模块 (可选择是否实现)
- 用户输入部分 (Input)
包括 Event 驱动+EventHub (Android 标准内容)
- 多媒体编解码 (Codec)
包括硬件 Codec 驱动+Codec 插件 (如 OpenMax)
- 3D 加速器部分 (3D Accelerator)
包括硬件 OpenGL 驱动+OpenGL 插件
- 音频部分 (Audio)
包括 Audio 驱动+Audio 硬件抽象层
- 视频输出部分 (Video Out)
包括视频显示驱动+Overlay 硬件抽象层
- 摄像头部分 (Camera)
包括 Camera 驱动 (通常是 v4l2) +Camera 硬件抽象层
- 电话部分 (Phone)
Modem 驱动程序+RIL 库
- 全球定位系统部分 (GPS)
包括 GPS 驱动 (通常为串口) +GPS 硬件抽象层
- 无线局域网部分 (WIFI)
包括 Wlan 驱动和协议+WIFI 的适配层 (Android 标准内容)
- 蓝牙部分 (Blue Tooth)
包括 BT 驱动和协议+BT 的适配层 (Android 标准内容)
- 传感器部分 (Sensor)
包括 Sensor 驱动+Sensor 硬件抽象层
- 震动器部分 (Vibrator)
包括 Vibrator 驱动+Vibrator 硬件抽象层 (Android 标准内容)
- 背光和指示灯部分 (Light)
包括 Light 驱动+Light 硬件抽象层
- 警告器—实时时钟部分 (Alarm&RTC)
包括 Alarm 驱动和 RTC 系统+用户空间调用 (Android 标准内容)