文章编号:1000-5013(2013)03-0286-06 **DOI**:10.11830/ISSN.1000-5013.2013.03.0286

i.MX51平台的Android系统移植

郭利全[1,2],谢维波[1,2]

(1. 华侨大学 计算机科学与技术学院,福建 厦门 361021;
2. 华侨大学 厦门软件园嵌入式技术开放实验室,福建 厦门 361008)

摘要: 分析Android平台的启动流程,选用交叉开发模式搭建软硬件平台,将Android手机操作系统移植于非手机硬件平台.阐述Android系统的移植经历源码的获取、源码裁剪、交叉编译源码、下载镜像,最后,进行Android系统的调试.结果表明:Android操作系统在i.MX51平台上能完成移植并成功调试.

关键词: Android系统;Linux系统;i.MX51平台;移植

中图分类号: TN 929.5;TP 316.5 **文献标志码**: A

近年来,随着经济的高速增长及科学技术的突飞猛进,人们的生活质量和生活水平有了显著地提高,智能手机逐渐走进了人们的生活.智能手机可以方便地获取互联网信息,带来高端的服务与享受.除苹果系列使用其专用的IOS操作系统外,其他智能手机大多使用Android操作系统.Android系统具有代码开源、兼容性好、系统可移植等特点,还具有界面美观、应用软件开发简单、音视频解码库齐全等优势.Android系统的优势使其在硬件平台的移植显得非常必要,将Android系统移植到非手机硬件平台具有巨大的商业价值.i.MX51平台是由美国Freescale公司自主研发的,中央处理器基于ARM Cortex A8核心的i.MX51处理器,主频可扩展到1 GHz.Android操作系统在i.MX51平台上能良好地运行,这为Android系统的移植奠定了硬件基础.本文对i.MX51平台上的Android系统移植进行研究.

1 Android系统启动流程

1.1 Android与嵌入式Linux

Android系统是基于Linux内核搭建的,Linux内核的优势在于大内存管理、进程管理、基于权限的安全模型、统一的驱动模型、对共享库的支持和代码开源等.Android系统在设计过程之中针对移动终端资源有限的特点,对Linux进行了一定程度地裁剪,去除了原生的窗口系统、对GUN libc的支持,并裁剪了一些标准的Linux工具.针对移动终端的特点,对内核在闹钟、内核调试、进程间通信、日志、电源管理等方面作了大量地优化.Android使用Linux完成其内存管理、进程管理、网络和其他服务工作,应用程序不会直接进行Linux的调用,但Linux在Android中的确存在[1].

1.2 Android系统启动流程

Android可以作为嵌入式系统的软件操作系统.Bootloader是嵌入式系统的引导加载程序,也是系统上电后运行的第一段程序,它除了完成基本的初始化系统和调用Linux内核的基本任务外,还对Linux的启动参数进行设置.Bootloader最后一项任务是调用Linux内核.

Linux内核一般存放于flash或SD卡中,但由于内核在flash中执行时代码会有限制,而且速度还没有在RAM中快,所以一般在嵌入式系统中都是将Linux内核解压到RAM中,然后跳转到RAM中去执行.Linux内核启动完成后,会创建init进程,而init进程首先进行一系列硬件初始化,然后通过命令行传递的参数挂载根文件系统.根文件系统包含系统引导和使其他文件系统得以挂载所必需的文件,

收稿日期: 2012-04-15

通信作者: 谢维波(1964-),男,教授,主要从事嵌入式技术、数字信号处理的研究.E-mail:xwblxf@hqu.edu.cn.

基金项目: 福建省自然科学基金资助项目(2010J01334);华侨大学高层次人才科研启动项目(11BS120)

也包括 Linux 启动时所必需的目录和关键性的文件.

Android 系统自下而上分为 linux 内核及驱动层、系统运行库和 java 运行环境层、应用程序框架层、应用层[2]. 应用程序框架层包括有 java 层和 jni 层，jni 是 Android 程序访问硬件抽象层的一种手段. Android 系统底层采用 Linux2.6 内核，Android 系统的启动包括 Linux 启动的全过程. 在 Linux 启动 init 进程后，init 进程根据 init.rc 脚本文件建立起 servicemanamger 和 zygote 等最基本的服务，其中 servicemanamger 的功能是管理系统中各种服务，zygote 的功能是建立 java 程序运行时的环境并启动虚拟机，初始化建立的基本服务都是运行在 java 运行环境层的本地服务. zygote 启动虚拟机后，建立 systemserver 进程，systemserver 是 Android java 层的系统服务模块，主要功能是管理供 Android 应用开发的系统服务，最后由 system server 建立 Android 应用层要使用的服务，包括 Home Activity 启动所需的 Activity Manager.

Android 系统启动流程，如图 1 所示. 由图 1 可知：Android 系统的启动是以 Linux 启动为基础，Android 系统的移植包含有 Linux 系统的移植. 嵌入式 Linux 的移植包括 bootloader，Linux 内核、根文件系统 3 大部分移植[3]. Android 系统的移植除上述移植外，还必须进行 system.img，recover.img，userdata.img 等镜像的移植[4].

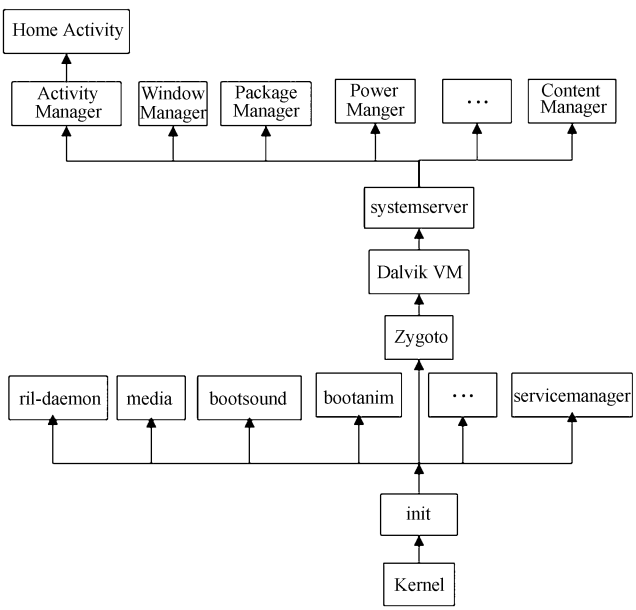

图 1　Android 系统启动流程图

Fig.1　Android system startup flowchart

2　软硬件平台的搭建

由于硬件平台的特殊性及局限性，在此选用交叉开发模式[5]. 交叉开发模式是指 Linux 交叉开发采用宿主机和目标机模式进行，宿主机是一台运行 Linux 的个人计算机，目标机为嵌入式硬件平台. 开发时使用宿主机上的交叉编译、汇编及连接工具，形成可以在目标机上执行的二进制代码，这种代码并不能在宿主机上执行，而只能在目标机上执行，然后把可执行文件下载到目标机上运行. 调试时可以通过串口、以太网口等进行.

本次移植过程中，宿主机为计算机，操作系统为 federa 13，在宿主机上下载 u-boot，Linux 2.6 内核、Android 源代码，并进行交叉编译. 为了支持 EABI，交叉编译工具采用 arm-none-Linux-gnueabi 工具链. 引导程序选用 u-boot，Linux 内核版本采用 linx-2.6.31.14，Android 系统版本采用 froyo. 编译好镜像后使用 ATK 工具通过串口进行镜像的烧写.

3　Android 系统的移植

Android 系统具有可移植性. 可移植性指的是在一定程度上，软部件从一种环境迁移到另一种环境

后还能正常工作的能力. Android 系统的移植需要经历源码的获取、源码裁剪、交叉编译源码、下载镜像这四个步骤[6].

3.1 源码的获取

源码的获取使用 git 工具. git 是为了帮助管理 Linux 内核开发而开发的一个开放源码的分布式版本控制软件. 尽管开发之初是为了辅助 Linux 内核开发,但现在包括 Android 在内的很多其他自由开源软件项目也使用了 git.

Android 是由 kernel,Dalvik,Bionic,prebuilt 等多个 git 项目组成,Android 项目编写了一个名为 repo 的 Python 脚本统一管理这些项目使 git 的使用更加简单. Android 源码的获取分为以下 5 个步骤.

1) yum install git,安装 git 工具.

2) curl https://dl-ssl.google.com/dl/googlesource/git-repo/repo〉～/bin/repo 下载 repo 脚本.

3) chmod a+x ～/bin/repo 改变下载后的 repo 权限,使其可以直接被运行.

4) repo init-u https://android.googlesource.com/platform/manifest-b froyo. 由于 Android 现在的版本为 4.0,本次移植工作使用的版本为 2.2. 使用这条语句从服务器端的 master 主分支中切换参数"-b"后面指定版本分支.

5) repo sync-j4. 由于 repo sync 是从源码服务器上进行下载,Android 源码一般在 10 GB 左右. 由于执行时间比较长,可以使用多线程来并行下载,使用参数"-j4",4 是并行的线程数.

Android 源码目录及各目录包含的内容,如表 1 所示.

表 1 Android 源码目录及作用

Tab.1 Android source code directory role

目录名称	目录内容及作用	目录名称	目录内容及作用
bionic	代替 Linux 中的 glibc 库,专为 Android 服务的 C 库,支持多平台	build	存放系统编译规则及 generic 等基础开发包配置
bootable	启动引导相关代码	cts	Android 兼容性测试套件标准
dalvik	Java 虚拟机	development	应用程序开发相关目录
device	Android 设备文件目录	external	Android 使用的一些开源的模组
frameworks	核心框架,提供组件供应用程序调用	hardware	部分厂家开源的硬解适配层 HAL 代码
packages	应用程序包	ndk	NDK 工具,提供包括 ndk-build,ndk-gdb 工具
sdk	sdk 及模拟器	prebuilt	Arm 及 x86 架构下预编译的一些资源
		system	底层文件系统库、应用及组件

获取 Android 源码后,接着获取 u-boot 及 Linux 内核源码. 在 Linux shell 终端输入以下 git 命令.

git clone git://git.denx.de/u-boot.git uboot-imx 获取 u-boot 源码保存至 uboot-imx 目录.

git clone git://git.kernel.org/pub/scm/Linux/kernel/git/stable/Linux-2.6.31.y.git kernel_imx 获取 Linux 内核源码,保存至 kernel_imx 文件夹中.

3.2 Linux 内核裁剪

Linux 内核的裁剪是移植工作重要的环节,不同的硬件厂商,裁剪的方式各不相同. 每个硬件厂商都会提供硬件的板级支持包(即 BSP). BSP 可以屏蔽硬件,提供操作系统及硬件的驱动. 本文使用板级支持包加速对 Linux 内核裁剪工作.

i.MX51 的硬件支持包是以补丁文件的形式存在. BSP 包下载地址为 http://www.freescale.com. 将 Linux 内核拷贝到 Android 源码同一级目录,对源码打上补丁,即可完成对 Linux 内核的裁剪. 裁剪流程有如下 2 个步骤.

1) . imx-android-r9.2/code/r9.2/android_pach.sh. 运行支持包中的 bash 文件,设定环境变量.

2) c_patch imx-android-r9.2/code/r9.2 imx_r9.2. 使用支持包为源码打补丁.

裁剪后的 Linux 源码增加了对 i.mx51 处理器的支持,源码位于 kernel/arch/arm/mach-mx5 目录下;对共享内存处理方式上进行了改变,增加 kernel/mm/ashmem.c 文件,它为进程间提供大块内存,同时为内核提供管理和回收这些内存的机制;在驱动程序中增加了 mxc 目录,包括对 Android 相关的驱动的支持,如 IPC 系统、日志系统、电源、闹钟管理、内存控制台、时钟控制的 gpio,switch 驱动等,也

包括如图像处理单元 iPU，视频处理单元 VPU 等驱动.

3.3　源码的编译

下载 arm-none-linux-gnueabi 交叉编译工具链，将其 bin 目录加入 PATH 环境变量. Android 源码的编译过程有如下 3 个步骤.

1）. build/envsetup. sh 使用 envsetup. sh 脚本初始化环境变量.

2）lunch imx51_bbg-eng 完整编译在 imx51 平台运行，使能所有的调试方法.

3）make-j4 使用 make 命令进行编译. 采用 4 个线程的多线程编译方式. 编译源码的配置情况，如图 2 所示.

由图 2 可知：Android 的版本为 2.2.1，宿主机为 Linux 操作系统 x86CPU，目标机为 arm 型 CPU. 对源码的编译配置通过指定参数进行，本次配置参数为 imx51_bbg-eng，参数含义如表 2～3 所示.

```
[root@localhost Android2.2]# lunch imx51 bbg-eng
============================================
PLATFORM VERSION CODENAME=REL
PLATFORM VERSION=2.2
TARGET PRODUCT=imx51 bbg
TARGET BUILD VARIANT=eng
TARGET SIMULATOR=false
TARGET BUILD TYPE=release
TARGET BUILD APPS=
TARGET ARCH=arm
HOST ARCH=x86
HOST OS=linux
HOST BUILD TYPE=release
BUILD ID=FRF85B
============================================
```

图 2　Android 源码编译配置

Fig. 2　Android source code compiled configuration

表 2　平台参数说明

Tab. 2　Parameter description of platform

参数名称	目标设备平台	说明
full	emulator	完全配置所有语言，程序，输入法
full_maguro	maguro	在 Galaxy Nexus 平台上运行
full_panda	panda	在 PandaBoard 开发板上运行
imx51_bbg	imx51	在 imx51 开发板上运行

表 3　BUILDTYPE 参数说明

Tab. 3　Parameter description of BUILDTYPE

参数名称	用途
user	受限访问，适合于生产产品的情况
userdebug	像“user”参数一样，但 root 用户及调试除外
eng	配置其他调试开发工具

3.4　镜像的烧写

Android 系统源码编译完成后，在 out 文件夹下会生成对应的镜像文件，包括 system. img，userdata. img，ramdisk. img 等镜像. system. img 包括了主要的包、库等文件；userdata. img 包括了一些用户数据；ramdisk. img 是 emulator 的文件系统，emulator 加载这 3 个镜像文件后，会把 system. img 和 userdata. img 分别加载到 ramdisk 文件系统中的 system 和 userdata 目录下，ramdisk. img 在文件系统中解压后为 root 目录，其中包括了 init，init. rc 等文件. ATK 是 Freescale 公司开发出的一款针对 MX 系列 CPU 为核心的 flash、SD 卡烧录软件，可以用来烧录 bootloader 和 kernel 到 SD 卡上. 本次移植过程中使用 ATK 工具通过串口进行烧写镜像.

使用 ATK 工具将 u-boot，kernel，ramdisk. img 镜像烧写到 SD 卡上，烧写的起始地址分别为 1 K，1 M，4 M 的地址处. 使用 Linux 中的 fdisk 命令对 SD 卡进行分区. 分区时不应将 u-boot 和 kenel 划入任何 1 个分区，第 1 个分区从 10 MB 的位置开始划分. 分区后，各分区的文件系统信息及分区大小设计，如表 4 所示.

表 4　SD 卡分区说明

Tab. 4　Partition description of SD card

分区类型	设备结点	文件系统	分区内容
主分区 1	/dev/sdb1	VFAT	媒体文件存储分区(>1 GB)
主分区 2	/dev/sdb2	EXT4	System 镜像分区(>300 MB)
扩展分区 3	/dev/sdb3	N/A	N/A
逻辑分区	/dev/sdb4	EXT4	data 镜像分区(>200 MB)
逻辑分区	/dev/sdb5	EXT4	Cache 分区(>100 MB)
主分区 4	/dev/sdb6	EXT4	Recovery 镜像分区(>100 MB)

在宿主机上，使用 dd 命令将 system. img 及 recovery. img 拷贝到 sd2，sd6 分区，拷贝的步骤如下：

dd if=system. img of=/dev/sd2

sync

dd if=recovery.img of=dev/sd6

sync

经过以上的过程,Android 系统平台运行所需的全部文件都已经烧写到 SD 卡上. SD 卡上存储区域分布,如图 3 所示. 图 3 中:MBR 主要是存储 SD 卡的分区信息,起始地址为 0 kB.

由图 3 可知:第一个分区(Media 分区)从大于 10 MB 的位置开始划分,可避免 MBR、引导程序、内核、根文件系统在分区中因误操作而被破坏的情况发生,保证系统正常启动.

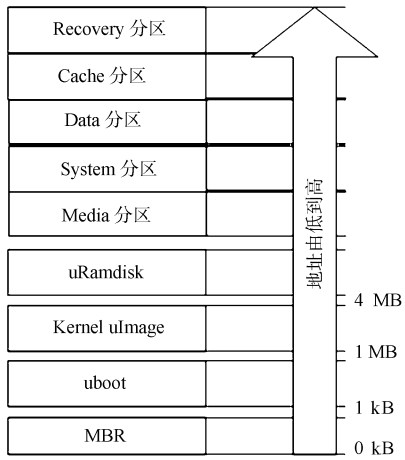

图 3 SD 卡存储区域分布图

Fig. 3 Storage area maps of SD card

4 Android 系统的调试

Linux 内核开发者为保证内核代码正确性,不愿在 Linux 内核源代码中加入调试器. 内核调试可采用监视内核代码和错误跟踪的方法. 对 Android 的调试方法包括 Linux 方法和 Android 特殊方法. 标准 Linux 调试有如下 3 个方法.

1) 启动 Android 仿真器环境,使用 adb shell 进行连接,在终端使用 ps 查看系统各个进程. 也可以使用 linux 的 proc 文件系统跟踪进程相关信息.

2) 使用 Linux 的 vmstat 和 top 命令统计系统中性能的信息.

3) 使用 dmesg 查看内核打印出来的信息,可以方便地跟踪内核状态信息.

Android 的 toolbox 中包含了一些非标准化的辅助命令,这些命令在 Linux 的 Shell 中没有,专为 Andorid 系统所使用. Android 特殊调试命令有如下 4 个方法.

1) 使用 netcfg 命令进行网络配置调试,可用于检查 Android 系统的网络状况.

2) 使用 service 命令获取 Android 系统已经启动的服务,显示某一个服务调用进程的情况.

3) Android 的 am 命令可以在控制台启动和管理活动、服务和发送广播等.

4) 对基于 Android 系统的程序开发,使用 logcat 命令可以方便得到程序的 log 信息.

5 实现效果

Android 系统在 i.MX51 平台移植完成并调试成功后,将开发板上的启动模式设置成从 SD 卡引导. 串口调试信息,如图 4 所示.

```
MMC read: dev # 0, block # 2048,count 6144 ...
6144 blocks read: OK

MMC read: dev # 0, block # 8192,count 600 ...
600 blocks read: OK
## Booting kernel from Legacy Image at 90800000 ...
   Image Name:   Linux-2.6.31-01020-g90ecff8
   Image Type:   ARM Linux Kernel Image (uncompressed)
   Data Size:    2663780 Bytes =  2.5 MB
   Load Address: 90008000
   Entry Point:  90008000
   Verifying Checksum ... OK
## Loading init Ramdisk from Legacy Image at 90b00000 ...
   Image Name:   Android Root Filesystem
   Image Type:   ARM Linux RAMDisk Image (uncompressed)
   Data Size:    212936 Bytes = 207.9 kB
   Load Address: 90308000
   Entry Point:  90308000
   Verifying Checksum ... OK
   Loading Kernel Image ... OK
OK

Starting kernel ...

Linux version 2.6.31-01020-g90ecff8 (b18293@madcmand) (gcc version 4.1.2) #1 PR0
CPU: ARMv7 Processor [412fc085] revision 5 (ARMv7), cr=10c53c7f
CPU: VIPT nonaliasing data cache, VIPT nonaliasing instruction cache
Machine: Freescale MX51 Babbage Board
```

图 4 Android OS 串口打印信息图

Fig. 4 Serial print of Android OS

从启动信息中可以看出,u-boot 先从 MBR 中读取分区信息,将 Linux 镜像解压到 RAM 地址为 0x9000800 处,将 Android 镜像解压到地址为 0x90308000 处. 启动信息中也包括了 CPU 信息、内核版

本、大小、Android 文件系统版本等信息. Android 平台在 i. MX51 平台启动后，其程序测试效果图如图 5 所示.

图 5　Android 程序测试效果图

Fig. 5　Effect diagram of Android application test

6　结论

随着嵌入式技术的发展，嵌入式系统的软件从只有前后台程序，发展到使用 μC/OS-Ⅱ等小型操作系统，再发展到现在应用如 Linux，WinCE 等大型操作系统. Android 系统是目前最前沿的嵌入式操作系统，虽然设计之初只是针对手机等移动平台，但在其他非手机硬件平台的应用已经成为一种趋势. 只有将最前沿的嵌入式操作系统与硬件平台相结合，嵌入式系统才会更具生命力和竞争力.

所介绍的 Android 系统在 i. MX51 平台的移植过程及调试方法，对 Android 系统在其他嵌入式平台的移植具有指导意义. 然而，由于对 Linux 内核裁剪部分使用的是 i. MX51 的板级支持包，在其他硬件平台的移植应使用相应的板级支持包，有一定的局限性，也是移植工作的不足，有待今后进一步提高.

参考文献：

[1] 王茜. Android 嵌入式系统架构及内核浅析[J]. 电脑开发与应用，2011，24(4)：59-61.

[2] 李凯. Android 操作系统分析与移植[D]. 广州：华南理工大学，2011：59-61.

[3] 曹木莲，姚放吾. 基于 i. MX21 的嵌入式 Linux 研究与移植[J]. 计算机技术与发展，2009，19(9)：97-98.

[4] ABHYUDAI S，SOMYA L. Android porting concepts[C]// Proceedings of 3rd International Conference on Electronics Computer Technology. Kanyakumari：[s. n.]，2011：129-133.

[5] 孙纪坤，张小全. 嵌入式 Linux 系统开发技术详解[M]. 北京：人民邮电出版社，2007：129-132.

[6] 韩超，梁泉. Android 系统级深入开发-移植与调试[M]. 北京：电子工业出版社，2011：70-72.

Porting Android System to i. MX51 Platform

GUO Li-quan[1,2]，XIE Wei-bo[1,2]

(1. College of Computer Science and Technology，Huaqiao University，Xiamen 361021，China；
2. Laboratory of Embedded Technology of Xiamen Software Park，Huaqiao University，Xiamen 361008，China)

Abstract：This paper analyzes the Android platform startup process，the choice of cross-development model to build the hardware and software platform，porting the Android mobile operating system to non-phone hardware platform. Description Android system transplanted concrete steps，the source code getting and cutting，cross-compile the source code，download mirror image，and debugging Android system. The results show that：the Android operating system can complete porting to i. MX51 platforms and successful commissioning.

Keywords：Android operating system；Linux operating system；i. MX51 platform；porting

（**责任编辑**：钱筠　　**英文审校**：吴逢铁）