# Android应用和驱动开发介绍

一 驱动

* 1. 环境配置
     1. 开发环境

由于驱动开发可能会涉及到Android系统的编译，推荐使用64位linux系统作为开发环境（官方要求编译Android在64位系统下进行）。本例中使用的系统是Ubuntu 12.04.3 LTS，通过制作U盘启动方式安装，详情参见<http://www.ubuntu.com/download/desktop/install-desktop-latest>。安装完系统后，驱动编译所需的kernel源码可在官网下载（下载方式详情参见<http://source.android.com/source/downloading.html>）。之后可通过命令：make -C “kernel dir” M=“pwd” modules 将源码独立编译成模块文件。

* + 1. 调试环境

主机与Android调试远程通信工具是ADB。ADB是一个客户端-服务器端程序，其中客户端是操作的电脑，服务器是Android设备。在电脑上安装的客户端包含在sdk中（<http://developer.android.com/sdk/index.html>）。设备上不需要安装，只需要在Android设备上打开选项settings->applications->development->USBdebbugging。之后，可以将sdk中包含的adb命令添加到PATH环境变量中，具体：打开~/.bashrc文件，在里面加入一行内容:export PATH=${PATH}:<sdk dir>/platform-tools。然后就可以使用adb命令了。adb命令使用方法下一节详解。

* 1. 工具集

以下讲解了android驱动开发调试的常用工具。其中insmod、rmmod、dmesg为系统工具，不需要手动安装。

**vim**

vim是linux常用强大的文本编辑工具。安装：sudo apt-get install vim。其相关教程文档网上很多，它只是个文本编辑器，可以用其他任意的文本编辑器代替。

**insmod**

insmod是加载单个模块到linux内核的命令，需要root权限。格式为：insmod [filename]。加载过程中的详细错误信息可通过dmesg命令查看。

**rmmod**

rmmod是从linux内核中卸载单个模块，需要root权限。格式为：rmmod [modulename]。其错误详细信息可通过dmesg命令查看。

**minicom**

minicom是linux下常用的串口通信程序，需要root权限。格式为：minicom [OPTINS]。安装：sudo apt-get install minicom。第一次运行minicom时会提示没有默认的配置文件，但不影响使用。进入minicom程序后会自动连接串口设备，如/dev/ttyUSB0。在现实用此工具的时候需要进行配置（运行minicom -s启动程序配置），步骤如下：

1、串口设置

运行minicom -s 命令启动后，有如下选项：

+-----[configuration]------+

| Filenames and paths |

| File transfer protocols |

| Serial port setup |

| Modem and dialing |

| Screen and keyboard |

| Save setup as dfl |

| Save setup as.. |

| Exit |

| Exit from Minicom |

+-----------------------------+

选择Serial port setup，出现的配置选项如下：

A — Serial Device : /dev/ttyS0

B — lockfile Location : /var/lock

C — Callin Program :

D — Callout Program:

E — Bps/par/Bits : 9600 8N1

F — Hardware Flow Control : YES

G — Software Flow Control : NO

Change with setting?

将选项A 的值设为/dev/ttyUSB0表示USB口1（由于主机没有多余串口，所以用了USB口转串口线，具体按照所遇情况而定）。

将选项E的值设为115200 8N1。

2、保存配置信息

在初始界面中选择Save as dfl，保存配置信息（默认保存到/etc/minicom/minirc.dfl）。

3、退出工具

选择Exit from minicom退出命令。

之后使用minicom命令时，就可以连接到/dev/ttyUSB0端口与远程开发板进行连接并同步显示调试信息。

minicom命令的其他操作选项，请查看man手册。

**dmesg**

dmesg是显示内核启动信息的命令。在驱动程序中用printk函数输出的信息，除了minicom串口调试工具同步显示外，还可以用此命令查看。

**adb**

adb作为google官方提供的Android设备远程控制工具，其提供了健全的控制Android设备的功能。常用的功能如下：

adb devices：查看当前连接的设备，连接到计算机的Android设备或模拟器将列出。

adb install <文件名>：将指定的apk程序安装到设备上。

adb uninstall <软件名>：卸载设备上指定的程序。

adb shell：此命令将登录到设备shell，相当于远程命令行。

adb push <文件名1> <文件名2>：将本地的文件1作为文件名2复制到设备。

adb pull <文件路径> <本地路径>：将设备上的文件复制到本地。

adb help：查看常用的adb选项。

* 1. bontouchusb-st驱动问题总结

1、ETIME、EPIPE错误

错误位置：驱动程序中的中断回调函数 usbtouch\_irq(struct urb \*urb)，457～478行。

错误类型：ETIME为超时错误，EPIPE为端口连接错误。

错误原因：未知。

修复状态：已修复完成。

修复方法：原程序中遇到这两种错误会重新上报传输请求（函数usb\_submit\_urb）。而修改后的程序直接返回函数，不做处理。

2、设备电源引用失败

错误位置：驱动程序中的设备打开函数usbtouch\_open(struct input\_dev \*input)中的语句：usb\_autopm\_get\_interface(usbtouch->interface)，500行左右。

错误类型： usb\_autopm\_get\_interface函数在内核中为设备的电源引用值加1，成功返回0失败返回错误码。

错误原因：客户Android系统中设备电源管理功能未打开，usb\_autopm\_get\_interface函数调用失败。每个设备都有对应的驱动程序，当驱动程序没有运行时会被内核自动挂载。为了防止内核将正在工作的设备对应的驱动挂载，linux内核为每个设备提供了设备供电计数变量，当设备工作时变量加1，结束工作后变量减1，直到此变量为0就挂载驱动程序。

修复状态：已修复完成。

修复方法：手动打开内核中的设备电源管理功能。具体：在驱动中声明usb\_driver驱动结构体时，为supports\_autosuspend赋值1，1077行左右。

* 1. 资料推荐

对于Android的资料首推官方文档：

<http://developer.android.com/index.html>

<http://source.android.com/>

论坛：

<http://www.oschina.net/android/>

<http://www.apkbus.com/>

<http://android.eoe.cn/>

书籍：

《Android-系统级深入开发》

《Android内核剖析》

《Android技术内幕-系统卷》

《Android NDK》

《Linux Device Driver》

《Linux In a Nutshell》

二 应用

* 1. 环境搭配

**安装eclipse**

Android应用开发所用的IDE是eclipse。<http://www.eclipse.org/downloads/>，提供了linux/window/os x三个操作系统下适合Android开发的工具。

**安装ADT插件**

* + - * 1. 运行eclipse，打开Help->Install New Software。
        2. 点击右上方的Add。
        3. 在弹出框中填入以下URL：

https://dl-ssl.google.com/android/eclipse/

* + - * 1. 点击OK。
        2. 在接下来的窗口中选择需要的开发工具点击Next。
        3. 此时会列出需要下载的工具列表，点击Next。
        4. 点击Finish完成选择，开始下载安装。
        5. 安装完成后重启Eclipse。

**配置ADT**

重启Eclipse后必须指定Android SDK的路径（Android SDK可从官网下载）：

* + - * 1. 在”Welcome to Android Development”窗口中选择Use existing SDKs。
        2. 浏览目录，找到SDK所在目录。
        3. 点击Next。

**添加平台库**

平台库可以用SDK管理器下载安装。安装步骤如下：

* + - * 1. 打开Window->Android SDK Manager。如图2-1

图2-1 Android SDK Manager界面

选择所需平台库，点击又下角的Install安装。

**创建模拟器**

* + - * 1. 打开Window->Android Virtual Device Manager。
        2. 点击弹出窗口又上方的New按钮。
        3. 按照需要填入要创建的模拟器的配置信息。点击OK完成。
  1. “CalibrationTools”校准程序简介
     1. 通信原理

Android基于linux内核，应用层与驱动进行通信支持POSIX标准中的ioctl函数方式。也就是说本程序是用ioctl系统接口与驱动程序进行通信的。通信通过以下步骤：

* + - * 1. 打开设备设备文件，调用系统函数open打开/dev/下的相应设备文件。
        2. 调用ioctl系统函数，向驱动发送命令字实现相应的操作。在此程序中ioctl函数有3个参数，第一个参数是所打开的设备文件的描述符，第二个参数是所发送的命令字（驱动定义的所有命令字详情请查看驱动源码），第三个参数是存储驱动中ioctl函数返回值的变量指针。
        3. 调用close函数关闭设备文件。
    1. NDK开发

Android下的NDK开发支持编写C/C++代码，来实现一些java语言不能完成的工作。其中C/C++代码会被编译成.so的库文件。而校准程序与驱动通信所用方法用到系统接口，需要用到C语言开发，所以NDK开发是必须的。想做NDK开发，首先需要下载Android NDK开发工具包（<http://developer.android.com/tools/sdk/ndk/index.html>）。在官网下载解压之后，配置~/.bashrc文件，在其中添加一行：export PATH=$PATH:<NDK dir>/。

在写C代码前需要做以下准备工作：

1. 在Android项目中创建新的类取名为MyNDKTools。

2. 加载本地库（库名随定，本程序中库名为“calibtools”），代码如下：

**static** {

System.*loadLibrary*("calibtools");

}

3. 声明库中的函数接口，例如：

public static native int calibrate();

在Android项目目录下创建jni目录，新建Android.mk文件，输入以下内容：

LOCAL\_PATH := $(call my-dir)

include $(CLEAR\_VARS)

LOCAL\_MODULE := calibtools

LOCAL\_SRC\_FILES := calibtools.c

include $(BUILD\_SHARED\_LIBRARY)

进入Android项目中的bin/classes目录下，运行命令：

javah -d ../../jni/ com.bontouch.calibration.CalibTools

之后会在jni/目录下生成com\_bontouch\_calibration\_CalibTools.h头文件。

在jni目录中新建calibtools.c文件实现头文件中定义的函数。

进入Android项目目录运行命令：

ndk-build

会在libs/armeabi/目录中生成.so库文件。

在eclipse中运行程序并调试程序。

* + 1. 校准算法

* 1. 资料推荐