密级状态： 绝密(　) 秘密(　) 内部资料(　) 公开(√)
Security Class：Most Confidential() Confidential() Internal() Public(√)

Rockchip_RK356X_Android11.0_Box_SDK_Release_V1.0.1_20210408_CN&EN

(Technical Department，R & D Dept. I)

文件状态： **File Status:** [　] 草稿 Draft [ ] 正在修改 Modifying [√] 正式发布 Released	**文件标识：** **File Identification**	RK-FB-YF-536
	当前版本： **Current Version**	1.0.1
	作　者： **Author**	WGH
	完成日期： **Completion Date**	2021-04-08
	审　核： **Checked By**	CW,ZXZ
	审核日期： **Audit Date**	2021-04-09

瑞芯微电子股份有限公司
Rockchip Electronics Co.,Ltd.

免责声明

本文档按“现状”提供，瑞芯微电子股份有限公司（“本公司”，下同）不对本文档的任何陈述、信息和内容的准确性、可靠性、完整性、适销性、特定目的性和非侵权性提供任何明示或暗示的声明或保证。本文档仅作为使用指导的参考。

由于产品版本升级或其他原因，本文档将可能在未经任何通知的情况下，不定期进行更新或修改。

商标声明

“Rockchip”、“瑞芯微”、“瑞芯”均为本公司的注册商标，归本公司所有。

本文档可能提及的其他所有注册商标或商标，由其各自拥有者所有。

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址： 福建省福州市铜盘路软件园 A 区 18 号

网址： www.rock-chips.com

客户服务电话： +86-4007-700-590

客户服务传真： +86-591-83951833

客户服务邮箱： fae@rock-chips.com

Warranty Disclaimer

This document is provided according to "current situation" and Rockchip Electronics Co., Ltd. (“the company”, the same below) is not responsible for providing any express or implied statement or warranty of accuracy, reliability, completeness, marketability, specific purpose or non-infringement of any statement, information and content of this document. This document is intended as a guide only.

Due to product version upgrades or other reasons, this document may be updated or modified from time to time without notice.

Brand Statement

Rockchip, “瑞芯微”, “瑞芯”, and other Rockchip trademarks are trademarks of Rockchip electronics Co., Ltd., and are owned by Rockchip electronics Co., Ltd.

All other trademarks or registered trademarks mentioned in this document are owned by their respective owners.

Rockchip Electronics Co., Ltd.

Address: No.18 Building, A District, No.89 Software Boulevard, FuZhou, FuJian, PRC

Website: www.rock-chips.com

Customer service Tel.： +86-4007-700-590

Customer service Fax： +86-591-83951833

Customer service e-Mail: fae@rock-chips.com

Rockchip_RK356X_Android11.0_Box_SDK_Release_V1.0.1_20210408_CN&EN

版 本 历 史 Version History

版本号 Version No.	作者 Author	修改日期 Modification Date	修改说明 Modification Description	备注 Remark
V1.0.0	WGH	2021-01-07	发布初始版本； Initial Release Version	
V1.0.1	WGH	2021-04-09	附录 C，风险提示 Risk reminder in appendix C	

Rockchip_RK356X_Android11.0_Box_SDK_Release_V1.0.1_20210408_CN&EN

目录 Contents

1 概述 Overview

本 SDK 是基于谷歌 Android11.0 64bit 系统，适配瑞芯微 RK356x 芯片的软件包，适用于 RK356x Box 产品及基于其上开发的所有产品。

This SDK software package is based on Google Android11.0 64bit system, which is adaptive to Rockchip RK356x chipset. It is suitable for RK356x box product and all development products based on it.

2 主要支持功能 Main Functions

参数 Parameter	模块名 Module Names
数据通信 Data Communication	WiFi、以太网卡、BT、USB、SDCard WiFi, Ethernet Card, BT, USB, SDCard
应用程序 Application	RkTvLauncher、媒体中心、TvSetting、WiFi Display、DLNA、浏览器、计算器 RkTvLauncher, Media Center, TvSetting, WiFi Display, DLNA, Browser, Caculator

3 SDK 获取说明 SDK Acquisition

3.1 获取 SDK Acquire SDK

SDK 通过瑞芯微代码服务器对外发布。其编译开发环境，参考附录 A 编译开发环境搭建。

SDK is released through Rockchip code server. Please refer to Appendix A to setup the compiling and development environment.

客户向瑞芯微技术窗口申请 SDK，需同步提供 SSH 公钥进行服务器认证授权，获得授权后即可同步代码。关于瑞芯微代码服务器 SSH 公钥授权，请参考附录 B SSH 公钥操作说明。

Customers apply for SDK from Rockchip FAE contacter, and will be able to sync code after obtaining the server certificate authorization with SSH public key provided. For more details about Rockchip code server SSH public key authorization, please refer to Appendix B SSH public key operation instruction.

RK356x_ANDROID11.0_BOX_SDK 下载命令如下：

RK356x_ANDROID11.0_BOX_SDK downloading command is as below:

```
repo init --repo-url
ssh://git@www.rockchip.com.cn/repo/rk/tools/repo -u
ssh://git@www.rockchip.com.cn/gerrit/rk/platform/manifest -b
android-11.0 -m rk356x_box_android11_release.xml
```

Repo 是 Google 用 Python 脚本写的调用 Git 的一个脚本，主要是用来下载、管理 Android 项目的软件仓库，其下载地址如下：

Repo is a script invoking git developed by Google using Python script, and mainly used to download, manage Android project software repository. The download address is as below:

```
git clone ssh://git@www.rockchip.com.cn/repo/rk/tools/repo
```

为方便客户快速获取 SDK 源码，瑞芯微技术窗口通常会提供对应版本的 SDK 初始压缩包，开发者可以通过这种方式，获得 SDK 代码的初始压缩包，该压缩包解压得到的源码，与通过 Repo 下载的源码是一致的。

Rockchip FAE contacter usually will provide the initial compressed package of the corresponding version SDK in order to help customers acquire SDK source code quickly.

以 Rockchip_Android_RK356x_11.0_BOX_SDK_V1.0.0_Release_20210107.tar.gz 为例，拷贝到该初始化包后，通过如下命令可检出源码：

Take Rockchip_Android_RK356x_11.0_BOX_SDK_V1.0.0_Release_20210107.tar.gz as an example, you can sync the source code through below command after copying the initial package:

```
mkdir rk356x
tar zxvf
Rockchip_Android_RK356x_11.0_BOX_SDK_V1.0.0_Release_20210107.tar.gz
-C rk356x
cd rk356x
.repo/repo/repo sync -l
.repo/repo/repo sync -c
```

后续开发者可根据 FAE 窗口定期发布的更新说明，通过”.repo/repo/repo sync -c”命令同步更新。

Later, developers can synchronize and update code using command ”.repo/repo/repo sync -c”, according to update information released by FAE contacter.

3.2 SDK 镜像 SDK Mirror

客户可以自己搭建 SDK 的 REPO 镜像服务器，来方便团队协同开发，镜像 SDK 的命令如下：

Customers can build SDK Repo mirror server, which is convenient for cooperative team development. Mirror SDK’s command is as below:

```
repo init --mirror --repo-url
ssh://git@www.rockchip.com.cn/repo/rk/tools/repo -u
ssh://git@www.rockchip.com.cn/gerrit/rk/platform/manifest -b
android-11.0 -m
rk356x_box_android11_release.xml
```

更多关于 REPO 镜像服务器搭建可以参考 RKDocs\common\RKTools manuals 目录下的《REPO 镜像服务器搭建和管理_V2.2_20131231.pdf》

For more information about Repo mirror server’s building, you can refer to document

“Rockchip_Introduction_REPO_Mirror_Server_Build_and_Management_CN.pdf” under directory “RKDocs\common\RKTools manuals”.

3.3 更改 remote 更快下载代码 Modify Remote to Download the Code Faster

AOSP 部分的代码是公开的，下载 RK 的 release 代码时速度较慢，您可以将 AOSP 的 remote 修改为您下载速度快的镜像源，如国外的客户可以修改为 google 的镜像源。这样可以提高下载速度。具体操作方法如下：

AOSP part code is public, the speed of downloading RK release code is relatively slow, then you can change the remote of AOSP to your mirror source with faster download speed, for example, foreign customers can change it to Google mirror source. In this way, you can increase the download speed. The detailed method is described as below:

执行 repo init（或者解压 base 包）后，修改.repo/manifests/remote.xml，把其中的 aosp 这个 remote 的 fetch 从

After executing repo init (or unpacking base package), modifying .repo/manifests/remote.xml, Change the remote fetch of aosp from

<remote name="aosp" fetch="./" review="https://10.10.10.29" />

改为

to

中国客户：（国内以清华大学镜像源为例，可以根据需要修改为其他国内镜像源)

Chinese customers: (here we take Tsinghua university mirror source as an example. You can change to other domestic mirror source as required)

<remote name="aosp" fetch="https://aosp.tuna.tsinghua.edu.cn" review="https://10.10.10.29" />;

国际客户：（google 镜像源）

International customers: (Google mirror source)

<remote name="aosp" fetch="https://android.googlesource.com" review="https://10.10.10.29" />

4 软件开发指南 Software Developer Guide

4.1 开发指南 Developer Guide

RK356x Box SDK Kernel 版本：Linux4.19，Android 版本：11.0，为帮助开发工程师更快上手熟悉 SDK 的开发调试工作，随 SDK 发布《Rockchip_Developer_Guide_Android10.0_V1.2.6_CN》。

RK356x Box SDK Kernel Version：Linux4.19, Android Version：11.0, to help developers know SDK’s developing and debug work quickly, document

“Rockchip_Developer_Guide_Android10.0_V1.2.6_CN.pdf” is also released with SDK together.

文档可在源码 RKDocs\目录下获取，并会不断完善更新。

You can get this document under directory “RKDocs\”. It will be updated more complete continuously.

5 SDK 编译说明 SDK Compilation Instruction

5.1 JDK 安装 JDK Installation

Android11.0 系统编译依赖于 Java 8。编译之前需安装 OpenJDK。

Android11.0 system compiling is dependent on JAVA 8. You need to install OpenJDK before compiling.

安装命令如下:

Installing command is as below:

```
sudo apt-get install openjdk-8-jdk
```

配置 Java 环境变量，例如，安装路径为/usr/lib/jvm/java-8-openjdk-amd64，可在终端执行如下命令配置环境变量。

Configure JAVA environment variable, for example, installation path is “/usr/lib/jvm/java-8-openjdk-amd64”. You can execute below command to configure environment variable on the terminal:

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export PATH=$JAVA_HOME/bin:$PATH
export CLASSPATH=.:$JAVA_HOME/lib:$JAVA_HOME/lib/tools.jar
```

SDK 带有 Open JDK8 的配置脚本，在工程根目录下，命名为 javaenv.sh。

There is configuration script for Open JDK8 along with SDK. It is under project root directory, named javaenv.sh

可直接执行以下命令，配置 JDK：

You can run below command to configure JDK:

```
source javaenv.sh
```

5.2 编译模式 Compilation Mode

SDK 默认以 userdebug 模式编译。

SDK default compilation mode is userdebug.

使用 ADB 时，需要先执行 adb root 使 shell 获取 root 权限，进而执行其它像 adb remount、adb push 等操作。

When using ADB, first you need to execute adb root to make shell acquire root authority, and then execute other operations such as adb remount, adb push, and so on.

5.3 代码编译 Code Compiling

5.3.1 uboot 编译步骤 Uboot Compiling Steps

```
./make.sh rk3566
```

编译完，会生成 trust.img、rk356x_loader_vx.xx.xxx.bin、uboot.img 三个文件。

After compilation, it will generate trust.img, rk356x_loader_vx.xx.xxx.bin and uboot.img three files.

5.3.2 kernel 编译步骤 Kernel Compiling Steps

RK356x BOX 样机板配置与编译命令如下：

The configuration and compilation of RK356x BOX is as below:

```
make ARCH=arm64 rockchip_defconfig android-11.config
make ARCH=arm64 rk3566-box-demo-v10.img -j8
```

编译完成后，在 kernel 根目录生成 kernel.img，resource.img 两个镜像文件。

After compilation, it will generate kernel.img and resource.img two mirror files under kernel root directory.

5.3.3 Android 编译步骤 Android Compiling Steps

客户按实际编译环境配置好 JDK 环境变量后，按照以下步骤配置完后，执行 make 即可。

Firstly, customers configure JDK environment variable according to the actual compiling environment, secondly, configure following below steps, at last execute make.

```
$ source build/envsetup.sh
$ lunch
编译 BOX，则选择 rk356x_box 相关选项
Compile BOX, then choose rk356x_box relevant option
```

如选择 rk356x_box-userdebug

Choose rk356x_box-userdebug

```
$ make -j16
-编译结束
-compiling end
```

5.3.4 固件打包 Packing Images

完成以上编译后，执行 SDK 根目录下的 mkimage.sh 脚本生成固件，所有烧写所需的镜像将都会拷贝于 rockdev/Image-rk356x_box 目录。

After the above compilation, execute mkimage.sh script under SDK root directory to generate images. All the mirror files required for flashing will be copied to rockdev/Image-rk356x_box directory.

```
rockdev/Image-rk356x_box/
├── baseparameter.img
```

```
├── boot-debug.img
├── boot.img
├── config.cfg
├── dtbo.img
├── MiniLoaderAll.bin
├── misc.img
├── parameter.txt
├── pcba_small_misc.img
├── pcba_whole_misc.img
├── recovery.img
├── resource.img
├── super.img
├── uboot.img
├── update.img
└── vbmeta.img
```

5.3.5 自动编译脚本 Compiling Script Automatically

为了提高编译的效率，降低人工编译可能出现的误操作，SDK 中集成了全自动化编译脚本，方便固件编译、备份。脚本的执行命令如下（在 SDK 工程根目录下）：

In order to improve compilation efficiency, and reduce miss operations by artificial compiling. In the SDK, there is automatic compiling script, it is convenient for firmware compiling and backup. The script's executing command is below(under SDK project root directory):

如果编译的是 box-userdebug 版本，请执行如下命令，其它版本请更改 lunch 选项即可：

If you are compiling the box-userdebug version, please execute the following command. For other versions, please change lunch option:

```
$ source build/envsetup.sh
$ lunch rk356x_box-userdebug
$ ./build.sh -UCKApu

(  备注：Remarks：

    U：means compile u-boot

    K：means compile kernel

    A：means compile Android

    p：代表将 img 拷贝到 根目录/IMAGE 目录 means copying img to the
root/IMAGE directory
```

```
    u：代表打包为 update.img  （要跟命令参数 p 搭配使用） means packaging
to update.img(need to use together with parameter p)
 )
```

该脚本会自动配置 JDK 环境，编译 U-Boot、kernel 和 Android，然后生成固件并打包生成 update.img。

This script will configure JDK enviroment, compile U-Boot, Kernel and Android automatically. And then images are generated and packed to be a update.img automatically too.

另外脚本会将编译生成固件拷贝至 IMAGE 目录下。每次编译都会新建目录保存，自动备份调试开发过程的固件版本，并存放固件版本的各类信息。

In addition, this automatic script will copy firmware generated by compiling to IMAGE directory. It will create new directory to save firmware everytime compiling, backup these firmwares and coresponding information.

5.3.6 单独编译 kernel 验证方法 Separate compilation of kernel validation methods

默认 kernel 编译完会将其打包到 boot.img 中，当我们不想完整编译 Android 又要快速验证 kernel 的相关问题的时候，可以使用如下命令：

By default, it will be packaged into boot.img after compiling the kernel. When we do not want to complete the compilation of Android but want to quickly verify the kernel related issues, we can use the following command:

```
make ARCH=arm64 rockchip_defconfig android-11.config && make
rk3566-box-demo-v10.img ARCH=arm64 BOOT_IMG=./boot_sample.img -j64
```

说明：执行上述命令前需要拿一个之前已经编译且可以正常使用的 boot.img 放到 kernel 目录并重命名为 boot_sample.img，编译命令执行结束后，在 kernel 目录会生成一个新的 boot.img，单独烧录这个 boot.img 即可验证当前 kernel 修改；

Note: Before executing the above command, you need to take a previously compiled and usable boot.img into the kernel directory and rename it to boot_sample.img. After executing the compilation command, a new boot.img will be generated in the kernel directory, which can be used to verify the current kernel modification.

6 刷机说明 Flashing Instruction

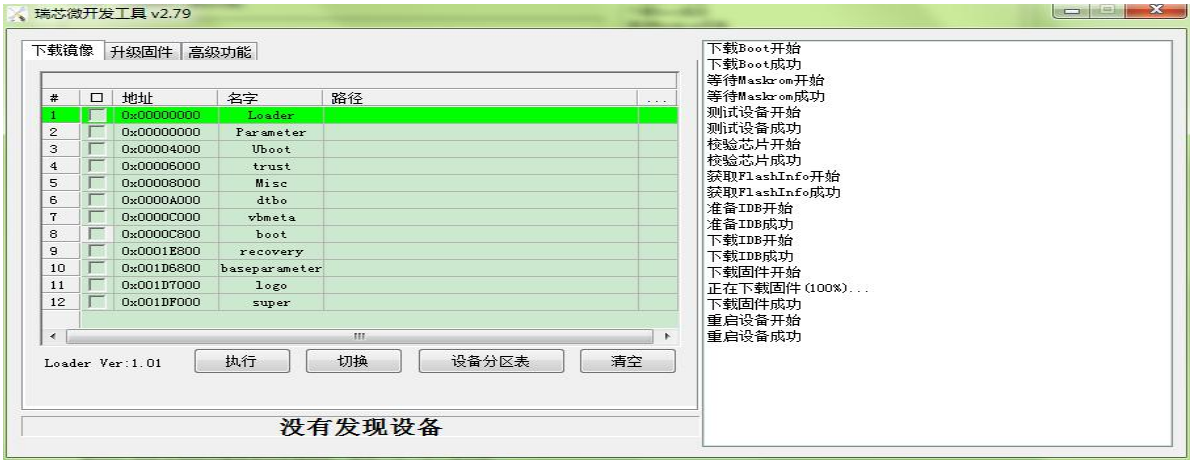
刷机说明详见 RKDocs\common\RKTools manuals 目录下《Android 开发工具手册.pdf》。

For detailed flashing instruction, please refer to “RKDocs\common\RKTools manuals\Rockchip_User_Manual_Android_Development_Tool_CN.pdf”

SDK 提供烧写工具，如下图所示。编译生成相应的固件后，进入烧写模式，即可进行刷机。

对于已烧过其它固件的机器，可以选择重新烧录固件，或是选择低格设备，擦除 idb，然后进行刷机。

SDK provides the flashing tool shown as below picture. After compiling to generate images, enter loader mode, and then you can flash the images. For those devices with existing images, please select to format the device, erase idb, and then flash the images.



注：烧写前，需安装最新的的 USB 驱动，驱动详见

Note:Before flashing, need to install newest USB driver, see details:

```
RKTools/windows/
├── DriverAssitant_v5.1.1.zip
```

注意：

Note：

1. RK356x ANDROID 11.0 最新版本使用了新的 spare 脚本来减少 system.img 和 super.img 的大小，需要使用 v2.79 版本或更高版本的工具进行烧写，否则会出现无法启动 Android 的情况；

RK356x ANDROID11.0 newest version uses new spare script to reduce the size of system.img and super.img. Need to use v2.79 or above version tool to flash, otherwise the phenomenon of device unable to boot Android occurs.

2. RK356x ANDROID 11.0 使用自动生成 parameter 的工具，基本分区已经定义完善，用户无需修改，具体位置为：

device/rockchip/rk356x_box/rk356x_box/rk356x_box.mk : BOARD_WITH_SPECIAL_PARTITIONS

其中已经定义了 box 需要的 baseparameter 和 logo 分区，用户可参照定义其它分区即可。

Rk356x Android 11.0 uses the tool of automatically generating parameters. The basic partition has been well defined and the user does not need to modify it. The specific location is：

device/rockchip/rk356x_box/rk356x_box/rk356x_box.mk : BOARD_WITH_SPECIAL_PARTITIONS

The baseparameter and logo partition required by the box have been defined. Users can refer to define other partitions.

附录 A 编译开发环境搭建 Compiling and Development Environment Setup

本章节介绍了如何设置本地工作环境来编译 Android 源文件。您需要使用 Linux 或 Mac OS。目前不支持在 Windows 环境下进行编译。

This chapter describes how to set up your local work environment to build the Android source files. You will need to use Linux or Mac OS. Building under Windows is not currently supported.

Android 11.0 建议软硬件配置：

Android 11.0 suggested software and hardware configuration：

- 操作系统：64 位 Ubuntu 14.04 及以上
  Operating System：64 bit Ubuntu 14.04 or above
- 硬盘空间：最小 200GB
  Hardware Size：At least 200GB
- Python 版本：2.7.6 及以上
  Python Version：2.7.6 and above
- JDK 版本：OpenJDK 8
  JDK Version：OpenJDK 8

注意：

Note:

从 Android (2.3.x) Gingerbread 版本开始，编译环境都要求为 64 位操作系统。

Starting from Android(2.3.x) Gingerbread Version， all compiling enviroment requires 64 bit operating system.

References :https://source.android.com/setup/initializing

附录 A.1 硬件要求 Hardware Requirements

您的开发工作站必须达到或超出以下软硬件要求：

Your development workstation must meet or exceed these requirements:

- 如果是 Gingerbread (2.3.x) 及更高版本（包括 master 分支），需要使用 64 位环境。如果是较低的版本，则可以在 32 位系统中进行编译。
  A 64-bit environment is required for Android 2.3.x(Gingerbread) and higher versions(including the master branch). You can compile older versions on 32-bit systems.
- 如果是校验代码，至少需要 100GB 可用磁盘空间；如果要进行编译，则还需要 150GB。如果要进行多次编译或使用 ccache，则需要更多空间。
  At least 100GB of free disk space to check out the code and an extra 150GB to build it. If you conduct multiple builds or use ccachee, you need additional space.
- 如果您在虚拟机中运行 Linux，则至少需要 16GB 的 RAM/交换空间。

If you're running Linux in a virtual machine, you need at least 16GB RAM/swap.

附录 A.2 软件要求 Software Requirements

Android 开源项目 (AOSP) master 分支历来都是在 Ubuntu Long Term Support (LTS) 版本中进行开发和测试，但您也可以使用其他 Ubuntu 分发版本。要查看建议使用的版本，请参阅下面的列表。

The Android Open Source Project(AOSP) master branch is routinely developed and tested on Ubuntu Long Term Support (LTS) versions, but other Ubuntu distributions may be used. See versions suggested to use， please refer to below lists.

附录 A.2.1 操作系统和 JDK Operating System and JDK

如果您要针对 AOSP master 分支进行开发，请使用下列操作系统之一：Ubuntu 14.04 (Trusty)/Mac OS v10.10 (Yosemite) 或更高版本（具有 Xcode 4.5.2 和命令行工具）。

If you're developing for the AOSP master branch, use Ubuntu 14.04 (Trusty)/Mac OS v10.10(Yosemite) or higher versions(with Xcode 4.5.2 and command line tools installed).

附录 A.2.2 主要软件包 Main Software Packages

- python.org 中提供的 Python 2.6 - 2.7
  Python 2.6-2.7 from python.org
- gnu.org 中提供的 GNU Make 3.81 - 3.82
  GNU Make 3.81-3.82 from gnu.org
- git-scm.com 中提供的 Git 1.7 或更高版本
  Git1.7 or higher version from git-scm.com

附录 A.2.3 操作系统 Operating System

Android 通常是在 GNU/Linux 或 Mac OS 操作系统中进行编译。您也可以使用虚拟机在不支持的系统（例如 Windows）中编译 Android。

Android is usually compiled under GNU/Linux or Mac OS operating systems. You can also use virtual machines to compile Android on unsupported systems, such as Windows.

- **GNU/Linux**
  - Android 6.0 (Marshmallow) - AOSP master：Ubuntu 14.04 (Trusty)
  - Android 2.3.x (Gingerbread) - Android 5.x (Lollipop)：Ubuntu 12.04 (Precise)
  - Android 1.5 (Cupcake) - Android 2.2.x (Froyo)：Ubuntu 10.04 (Lucid)
- **Mac OS (Intel/x86)**
  - Android 6.0 (Marshmallow) - AOSP master：Mac OS v10.10 (Yosemite) 或更高版本，具有

Xcode 4.5.2 和命令行工具

Android 6.0 (Marshmallow) - AOSP master： Mac OS v10.10 (Yosemite) or higher version，with Xcode 4.5.2 and command line tools installed

- Android 5.x (Lollipop)：Mac OS v10.8 (Mountain Lion)，具有 Xcode 4.5.2 和命令行工具

  Android 5.x (Lollipop)： Mac OS v10.8 (Mountain Lion)， with Xcode 4.5.2 and command line tools installed
- Android 4.1.x-4.3.x (Jelly Bean) - Android 4.4.x (KitKat)：Mac OS v10.6 (Snow Leopard) 或 Mac OS X v10.7 (Lion)，以及 Xcode 4.2（Apple 的开发者工具）

  Android 4.1.x-4.3.x (Jelly Bean) - Android 4.4.x (KitKat)：Mac OS v10.6 (Snow Leopard) or Mac OS X v10.7 (Lion)，and Xcode 4.2（Apple's Developer Tool）
- Android 1.5 (Cupcake) - Android 4.0.x (Ice Cream Sandwich)：Mac OS v10.5 (Leopard) 或 Mac OS X v10.6 (Snow Leopard)，以及 Mac OS X v10.5 SDK

  Android 1.5 (Cupcake) - Android 4.0.x (Ice Cream Sandwich)： Mac OS v10.5 (Leopard) or Mac OS X v10.6 (Snow Leopard)，and Mac OS X v10.5 SDK

注意：请考虑在 GNU/Linux（而不是其他操作系统）上进行编译。Android 编译系统通常使用编译设备上运行的 ART 来预编译系统 dex 文件。由于 ART 只能在 Linux 上运行，因此编译系统会在非 Linux 操作系统上跳过这个预编译步骤，从而导致 Android 编译的性能下降。

Note: Consider compiling under GNU/Linux systems rather than any other operating system. Android build system usually uses ART, running on the build machine, to pre-compile system dex files. Since ART is able to run only on Linux, the build system skips this pre-compilation step under non-Linux operating systems, resulting in an Android build with reduced performance.

附录 A.2.4 JDK

- Android 9.0 (Pie) - Android 11.0 (R)：Ubuntu - OpenJDK 8；Mac OS - jdk 8u45 或更高版本

  Android 9.0 (Pie) - Android 11.0 (R)： Ubuntu - OpenJDK 8； Mac OS - jdk 8u45 or a higher version
- Android 7.0 (Nougat) - Android 8.0 (O)：Ubuntu - OpenJDK 8；Mac OS - jdk 8u45 或更高版本

  Android 7.0 (Nougat) - Android 8.0 (O)： Ubuntu - OpenJDK 8； Mac OS - jdk 8u45 or a higher version
- Android 5.x (Lollipop) - Android 6.0 (Marshmallow)： Ubuntu - OpenJDK 7； Mac OS - jdk-7u71-macosx-x64.dmg
- Android 2.3.x (Gingerbread) - Android 4.4.x (KitKat)： Ubuntu - Java JDK 6； Mac OS - Java JDK 6
- Android 1.5 (Cupcake) - Android 2.2.x (Froyo)： Ubuntu - Java JDK 5

附录 A.3 设置 Linux 编译环境 Setting up a Linux Compiling Enviroment

以下说明适用于所有分支（包括 master ）。

Below instructions apply to all branches(including master).

我们会定期在最近推出的一些 Ubuntu LTS (14.04) 版本中对 Android 编译过程进行内部测试，但大多数 Ubuntu 分发版本都应该有所需的编译工具。欢迎向我们报告在其他分发版本中的测试结果（无论结果是成功还是失败）。

The Android build is routinely tested in house on recent versions of Ubuntu LTS (14.04). But most Ubuntu distributions should have the required build tools available. Reports of successes or failures on other distributions are welcome.

如果是 Gingerbread (2.3.x) 及更高版本（包括 master 分支），需要使用 64 位环境。如果是较低的版本，则可以在 32 位系统中进行编译。

For Gingerbread (2.3.x) and higher versions, including the master branch, a 64-bit environment is required. Older versions can be compiled on 32-bit systems.

附录 A.3.1 安装 JDK Installing JDK

在 Ubuntu 上，请使用 OpenJDK。要了解确切的版本，请参阅 JDK 要求；要了解相关说明，请参阅以下各个部分。

On Ubuntu, please use OpenJDK. See JDK Requirements for precise versions and the sections below for instructions.

- **如果 Ubuntu >= 15.04**

  For Ubuntu >= 15.04

  请运行以下命令：

  Run the following:

```
sudo apt-get updatesudo apt-get install openjdk-8-jdk
```

- **如果是 Ubuntu LTS 14.04**

  For Ubuntu LTS 14.04

目前没有适用于 Ubuntu 14.04 的受支持 OpenJDK 8 程序包。Ubuntu 15.04 OpenJDK 8 软件包能够在 Ubuntu 14.04 中顺利使用。我们发现，按照以下说明操作时，更高的程序包版本（例如适合 15.10、16.04 的版本）在 Ubuntu 14.04 中无法正常工作。

Currently, there is no supported OpenJDK 8 package applying to Ubuntu 14.04. The Ubuntu 15.04 OpenJDK 8 packages can be used successfully on Ubuntu 14.04. Higher package versions (for example, those for 15.10, 16.04) can't work normally on Ubuntu14.04 using the instructions below.

1. 从 old-releases.ubuntu.com 下载适用于 64 位架构的 .deb 软件包：

   Download the .deb packages for 64-bit architecture from old-releases.ubuntu.com:

   - openjdk-8-jre-headless_8u45-b14-1_amd64.deb

   （SHA256：

0f5aba8db39088283b51e00054813063173a4d8809f70033976f83e214ab56c0）

- openjdk-8-jre_8u45-b14-1_amd64.deb

（SHA256：9ef76c4562d39432b69baf6c18f199707c5c56a5b4566847df908b7d74e15849）

- openjdk-8-jdk_8u45-b14-1_amd64.deb

（SHA256：6e47215cf6205aa829e6a0a64985075bd29d1f428a4006a80c9db371c2fc3c4c）

2. （可选）对照随以上每个程序包列出的 SHA256 字符串，确认已下载文件的校验和。例如，使用 sha256sum 工具：

(Optionally) Confirm the checksums of the downloaded files comparing to the SHA256 string for each package listed above. For example, use the sha256sum tool:

```
sha256sum {downloaded.deb file}
```

3. 安装程序包：

Install the packages:

```
sudo apt-get update
```

为下载的每个 .deb 文件运行 dpkg。运行过程中可能会因缺少依赖项而出现错误：

Run dpkg for each of the .deb files you have downloaded. It may produce errors due to missing dependencies:

```
sudo dpkg -i {downloaded.deb file}
```

解决缺少依赖项的问题：

To fix missing dependencies:

```
sudo apt-get -f install
```

➢ **更新默认的 Java 版本 - 可选 Update the default Java version - optional**

（可选）对于以上 Ubuntu 版本，您可以通过运行以下命令来更新默认的 Java 版本：

(Optionally) For the above Ubuntu versions, update the default Java version by running below command:

```
sudo update-alternatives --config javasudo update-alternatives --
config javac
```

在编译过程中，如果您遇到 Java 版本错误，请按照错误的 Java 版本部分中的说明设置其路径。

If you encounter a Java version error during compilation, see Wrong Java version instruction to set the path.

附录 A.3.2 安装所需的程序包 (Ubuntu 14.04) Installing Required Packages (Ubuntu 14.04)

您将需要 64 位版本的 Ubuntu。建议您使用 Ubuntu 14.04。

You need a 64-bit version of Ubuntu (14.04 is recommended).

```
sudo apt-get install git-core gnupg flex bison gperf build-
essential zip curl zlib1g-dev gcc-multilib g++-multilib libc6-dev-
i386 lib32ncurses5-dev x11proto-core-dev libx11-dev lib32z-dev
ccache libgl1-mesa-dev libxml2-utils xsltproc unzip
```

注意：要使用 SELinux 工具进行政策分析，您还需要安装 **python-networkx** 软件包。
Note: To use SELinux tools for policy analysis, you also need to install the python-networkx package.
如果您使用 LDAP 并且希望运行 ART 主机测试，则还需要安装 **libnss-sss:i386** 软件包。
Note: If you're using LDAP and want to run ART host tests, you might also need to install the libnss-sss:i386 package.

附录 A.3.3 安装所需的程序包 (Ubuntu 12.04) Installing Required Packages (Ubuntu 12.04)

您可以使用 Ubuntu 12.04 来编译较低版本的 Android。master 或最近推出的一些版本不支持 Ubuntu 12.04。

You can use Ubuntu 12.04 to build older versions of Android. Ubuntu 12.04 isn't supported on master or recent releases.

```
sudo apt-get install git gnupg flex bison gperf build-essential zip
curl libc6-dev libncurses5-dev:i386 x11proto-core-dev libx11-
dev:i386 libreadline6-dev:i386 libgl1-mesa-glx:i386 libgl1-mesa-dev
g++-multilib mingw32 tofrodos python-markdown libxml2-utils
xsltproc zlib1g-dev:i386sudo ln -s /usr/lib/i386-linux-
gnu/mesa/libGL.so.1 /usr/lib/i386-linux-gnu/libGL.so
```

附录 A.4 设置 Mac OS 编译环境 Setting up MacOS Build Environment

在默认安装过程中，Mac OS 会在一个保留大小写但不区分大小写的文件系统中运行。Git 并不支持此类文件系统，而且此类文件系统会导致某些 Git 命令（例如 git status）的行为出现异常。因此，我们建议您始终在区分大小写的文件系统中对 AOSP 源文件进行操作。使用下文中介绍的磁盘映像可以非常轻松地做到这一点。

During the default installation, Mac OS runs on a case-preserving but case-insensitive file system. This type of file system isn't supported by Git and causes the behavior of some Git commands (such as git status) to be abnormal. Therefore, we recommend you always work with the AOSP source files on a case-sensitive file system. This can be done very easily using the disk image, discussed below.

有了适当的文件系统，在新型 Mac OS 环境中编译 master 分支就会变得非常简单。要编译较低版本的分支，则需要一些额外的工具和 SDK。

With a proper file system, building the master branch in a new macOS environment is straightforward. Earlier branches require some additional tools and SDKs.

附录 A.4.1 创建区分大小写的磁盘映像 Creating a Case-sensitive Disk Image

您可以使用磁盘映像在现有的 Mac OS 环境中创建区分大小写的文件系统。要创建磁盘映像，请启动磁盘工具，然后选择“新建映像”。完成编译至少需要 25GB 空间；更大的空间能够更好地

满足未来的需求。使用稀疏映像有助于节省空间，而且以后可以随着需求的增加进行扩展。请务必选择“Case sensitive, Journaled”存储卷格式。

You can create a case-sensitive file system within your existing Mac OS environment using a disk image. To create the image, please launch Disk Utility and select New Image. At least size of 25 GB to complete the build; more space is better for future growth. Using sparse images saves space while allowing growth as needed. Select case sensitive, journaled as the volume format.

您也可以通过 shell 使用以下命令创建磁盘映像：

You can also create the disk image from shell using the following command:

```
hdiutil create -type SPARSE -fs 'Case-sensitive Journaled HFS+' -
size 40g ~/android.dmg
```

这将创建一个 .dmg（也可能是 .dmg.sparseimage）文件，该文件在装载后可用作具有 Android 开发所需格式的存储卷。

This creates a .dmg (or possibly a .dmg.sparseimage) file. Which when mounted, acts as a volume with the required format for Android development.

如果您以后需要更大的存储卷，还可以使用以下命令来调整稀疏映像的大小：

If you need a larger volume later, you can resize the sparse image using the following command:

```
hdiutil resize -size <new-size-you-want>g ~/android.dmg.sparseimage
```

对于存储在主目录下的名为 android.dmg 的磁盘映像，您可以向 ~/.bash_profile 中添加辅助函数：

For a disk image named android.dmg stored in your home directory, you can add helper functions to ~/.bash_profile:

- 要在执行 mountAndroid 时装载磁盘映像，请运行以下命令：

  To mount the image when you execute mountAndroid, please run below command:

```
# mount the android file image
mountAndroid() { hdiutil attach ~/android.dmg -mountpoint
/Volumes/android; }
```

注意：如果系统创建的是 **.dmg.sparseimage** 文件，请将 **~/android.dmg** 替换为 **~/android.dmg.sparseimage**。

Note: If your system created a **.dmg.sparseimage** file, replace **~/android.dmg** with **~/android.dmg.sparseimage**.

- 要在执行 umountAndroid 时卸载磁盘映像，请运行以下命令：

  To unmount disk image when you execute umountAndroid, please run below command:

```
# unmount the android file image
umountAndroid() { hdiutil detach /Volumes/android; }
```

装载 android 存储卷后，您将在其中开展所有工作。您可以像对待外接式存储盘一样将其弹出（卸载）。

After you've mounted the android volume, you do all your work there. You can eject it (unmount it) just as you would for an external drive.

附录 A.4.2 安装 JDK Installing JDK

要查看要在开发各种 Android 版本时使用的 Java 版本，请参阅上述软件要求。

See available Java versions when developing diferent Android versions, refer to the above software requirements

➢ **安装所需的程序包 Install required packages**

1. 使用以下命令安装 Xcode 命令行工具：

   Install the Xcode command line tools:

   ```
   xcode-select --install
   ```

   对于较低版本的 Mac OS（10.8 或更低版本），您需要通过 Apple 开发者网站安装 Xcode。如果您尚未注册成为 Apple 开发者，则需要创建一个 Apple ID 才能下载。

   For older versions of macOS (10.8 or lower), you need to install Xcode from the Apple developer site. If you haven't registered as an Apple developer, you need to create an Apple ID to download.

2. 通过 macports.org 安装 MacPorts。

   Install MacPorts from macports.org.

   注意：请确保在路径中 /opt/local/bin 显示在 /usr/bin 之前。否则，请将以下内容添加到 ~/.bash_profile 文件中：

   Note: Ensure that /opt/local/bin appear before /usr/bin, otherwise, add below content to ~/.bash_profile file:

   ```
   export PATH=/opt/local/bin:$PATH
   ```

   注意：如果主目录中没有 .bash_profile 文件，请创建一个。

   Note: If there is no .bash_profile file existing in your home directory, create it.

3. 通过 MacPorts 获取 Make、Git 和 GPG 程序包：

   Get Make, Git and GPG packages by MacPorts:

   ```
   POSIXLY_CORRECT=1 sudo port install gmake libsdl git gnupg
   ```

   如果您使用 Mac OS X v10.4，还需要安装 bison：

   If you use Mac OS X v10.4, also need to install bison:

   ```
   POSIXLY_CORRECT=1 sudo port install bison
   ```

➢ **设置文件描述符数量上限 Setting a file descriptor limit**

在 Mac OS 中，可同时打开的文件描述符的默认数量上限太低，在高度并行的编译流程中，可能会超出此上限。

On MacOS, the number of file descriptors can be opened at the same time is too low defaultly. In a highly parallel build process, the number may exceed this limit.

要提高此上限，请将下列行添加到 ~/.bash_profile 中：

To increase the cap, add the following lines to your ~/.bash_profile:

```
ulimit -S -n 1024
```

附录 B　SSH 公钥操作说明 SSH Public Key Operation Instruction

附录 B.1　SSH 公钥生成 SSH Public Key Generation

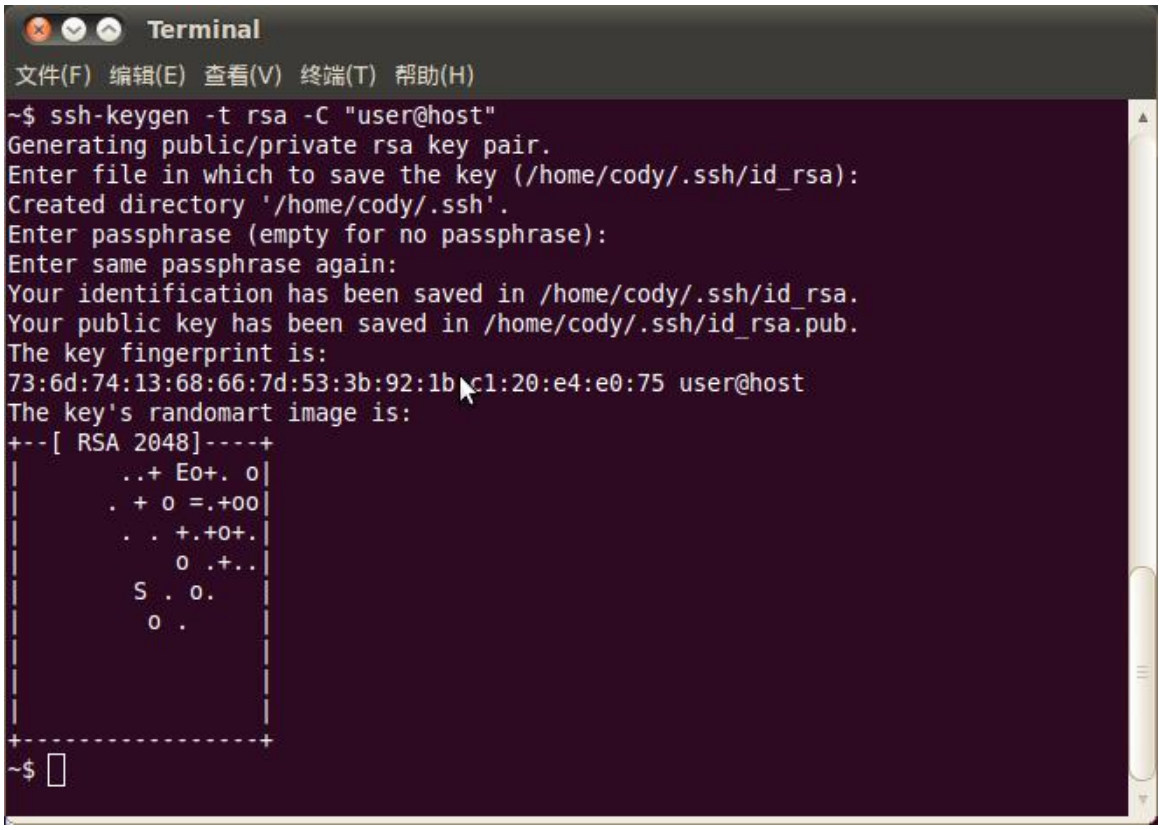
使用如下命令生成：

Use below command to generate:

```
ssh-keygen -t rsa -C "user@host"
```

请将 user@host 替换成您的邮箱地址。

Please replace user@host with your email address



命令运行完成会在你的目录下生成 key 文件。

It will generate key file under your directory after the command executes successfully.

```
~$ ls -l .ssh/
总用量 8
-rw------- 1 cody cody 1675 2012-10-15 11:38 id_rsa
-rw-r--r-- 1 cody cody  391 2012-10-15 11:38 id_rsa.pub
```

请妥善保存生成的私钥文件 id_rsa 和密码，并将 id_rsa.pub 发邮件给 SDK 发布服务器的管理员。

Please keep carefully the generated private key file id_rsa and password, and send id_rsa.pub to SDK release server's administrator using email.

附录 B.2 使用 key-chain 管理密钥 Use Key-chain to Manage the Private Key

推荐您使用比较简易的工具 keychain 管理密钥。

Recommend you use the simple tool keychain to manage the private key.

具体使用方法如下：

The detailed usage is as below:

1. 安装 keychain 软件包： Install keychain software package:

```
$sudo aptitude install keychain
```

2. 配置使用密钥： Configure to use the private key:

```
$vim ~/.bashrc
```

增加下面这行：

Add below line

```
eval `keychain --eval ~/.ssh/id_rsa`
```

其中，id_rsa 是私钥文件名称。

Among which, id_rsa is the file name of the private key.

以上配置以后，重新登录控制台，会提示输入密码，只需输入生成密钥时使用的密码即可，若无密码可不输入。

Log in the console again after configuring as above, and it will prompt to input the password. Only need to input the password used for generating the private key if there is one.

另外，请尽量不要使用 sudo 或 root 用户，除非您知道如何处理，否则将导致权限以及密钥管理混乱。

Besides, please avoid using sudo or root user unless you know how to deal with, otherwise it will cause the authority and private key management problems.

附录 B.3 多台机器使用相同 ssh 公钥 Multiple Devices Use the Same SSH Public Key

在不同机器使用，可以将你的 ssh 私钥文件 id_rsa 拷贝到要使用的机器的“~/.ssh/id_rsa”即可。

In order to use on different devices, you can copy ssh private key file id_rsa to the target device “~/.ssh/id_rsa”.

在使用错误的私钥会出现如下提示，请注意替换成正确的私钥。

Below hint will appear if use wrong private key. Please replace with correct private key.

```
~/tmp$ git clone git@172.16.10.211:rk292x/mid/4.1.1_r1
Initialized empty Git repository in /home/cody/tmp/4.1.1_r1/.git/
The authenticity of host '172.16.10.211 (172.16.10.211)' can't be established.
RSA key fingerprint is fe:36:dd:30:bb:83:73:e1:0b:df:90:e2:73:e4:61:46.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.16.10.211' (RSA) to the list of known hosts.
git@172.16.10.211's password:
```

添加正确的私钥后，就可以使用 git 克隆代码，如下图。

After adding the correct private key, you can use git to clone code, shown as below picture:

```
~$ cd tmp/
~/tmp$ git clone git@172.16.10.211:rk292x/mid/4.1.1_r1
Initialized empty Git repository in /home/cody/tmp/4.1.1_r1/.git/
The authenticity of host '172.16.10.211 (172.16.10.211)' can't be established.
RSA key fingerprint is fe:36:dd:30:bb:83:73:e1:0b:df:90:e2:73:e4:61:46.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.16.10.211' (RSA) to the list of known hosts.
remote: Counting objects: 237923, done.
remote: Compressing objects: 100% (168382/168382), done.
Receiving objects:   9% (21570/237923), 61.52 MiB | 11.14 MiB/s
```

添加 ssh 私钥可能出现如下提示错误。

Below error may appear while adding ssh private key:

```
Agent admitted failture to sign using the key
```

在 console 输入如下命令即可解决。

Input below command on console can fix it.

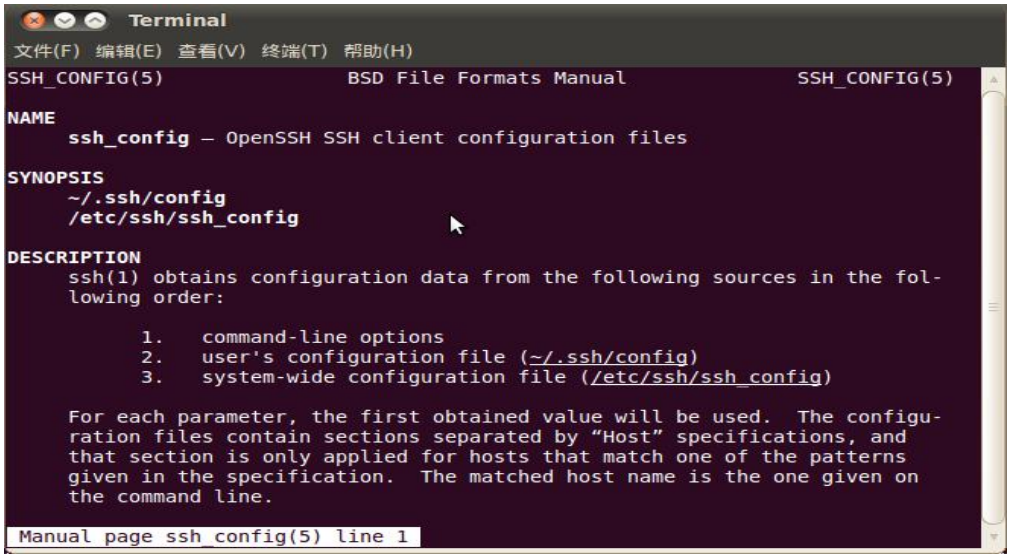
```
ssh-add ~/.ssh/id_rsa
```

附录 B.4 一台机器切换不同 ssh 公钥 Switch Different SSH Public Keys on One Device

可以参考 ssh_config 文档配置 ssh。

You can refer to ssh_config manual to configure ssh.

```
~$ man ssh_config
```



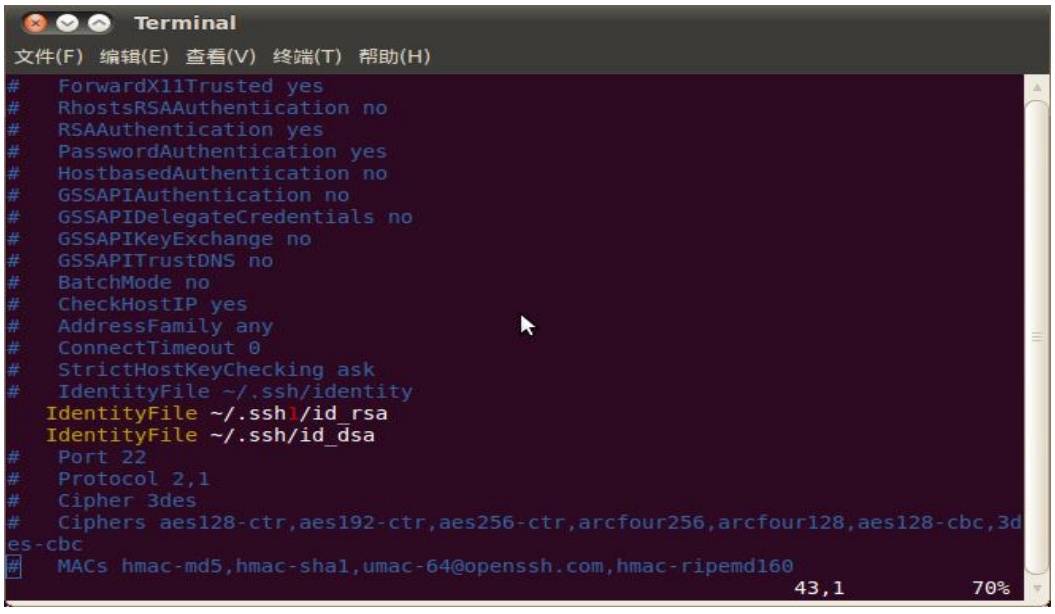
通过如下命令，配置当前用户的 ssh 配置。

Use below commands to configure ssh for current user.

```
~$ cp /etc/ssh/ssh_config ~/.ssh/config
~$ vi .ssh/config
```

如图，将 ssh 使用另一个目录的文件“~/.ssh1/id_rsa”作为认证私钥。通过这种方法，可以切换不同的的密钥。

As below picture, use another directory ssh file ”~/.ssh1/id_rsa” as certificate private key. In this way, you can switch different private keys.



附录 B.5 密钥权限管理 Private Key Authority Management

服务器可以实时监控某个 key 的下载次数、IP 等信息，如果发现异常将禁用相应的 key 的下载权限。

The server can real-time monitor a specific key's information, such as download times, IP etc. If there is abnormal case, it will prohibit the download authority of the corresponding key.

请妥善保管私钥文件。并不要二次授权与第三方使用。

Please keep carefully the private key file. DO not re-authorize it to a third party.

附录 B.6 Git 权限申请说明 Git Authority Application Instruction

参考上述章节，生成公钥文件，发邮件至 fae@rock-chips.com，申请开通 SDK 代码下载权限。

Refer to the above sections, generate a public key file, then send email to fae@rock-chips.com and apply for SDK code download authority.

附录 C 注意事项 Matters needing attention

附录 C.1 RK356X GPIO 电压域配置

•IO 电源管脚名，如 VCCIO1 表示， VCCIO1 电源域的电平， 支持 3.3V/1.8V。其他的 IO 电源管脚名分别为： VCCIO2， VCCIO3， VCCIO4， VCCIO5，VCCIO6， VCCIO7。

•VCCIO1/2/3/4/5/6/7 可以支持 3.3V/1.8V。

•PMUIOO 电源域的 IO 电平（管脚名 PMUIOO）：只支持 1.8V。

•PMUIO1 电源域的 IO 电平（管脚名 PMUIO1）：只支持 3.3V。

•PMUIO2 电源域的 IO 电平（管脚名 PMUIO2）：支持 3.3V/1.8V。

•VCCIO2 软件不要配置电压域，由硬件选择；PMUIOO 软件不要配置电压域，PMUIO1 只能设置 3.3v。其余各路均需与硬件确认之后，正确配置电压域值。

•主控电源域的 IO 电平要与对接外设芯片的 IO 电平保持一致，还要注意软件的电压配置要跟硬件的实际电压一致， 否则可能会导致 GPIO 的损坏。比如硬件 IO 电平接 1.8V， 软件的电压配

置也要相应的配成 1.8V； 硬件 IO 电平接 3.3V， 软件的电压配置也要用 3.3V， 否则可能会导致 GPIO 永久损坏。

- 具体的 io-domain 的配置参考文档《Rockchip IO-Domain 开发指南 》。