

密级状态：绝密( )      秘密( )      内部( )      公开(√ )

# RK312X\_ANDROID7.1-TABLET- SDK\_V1.00\_20170519

## 发布说明

(技术部，第一系统产品部)

文件状态：  [ ] 正在修改  [√] 正式发布	当前版本：	V1.00
	作 者：	hy
	审 核：	CW, ZXZ
	完成日期：	2017-5-19

福州瑞芯微电子股份有限公司

(版本所有,翻版必究)

## 版本历史

版本号	作者	修改日期	修改说明	备注
Beta	Zj	2017.03.01	正式发布	
1.00	hy	2017.05.19	正式发布	

## 目 录

1	概述 .....	4
2	主要支持功能 .....	4
3	SDK 获取说明 .....	4
3.1	获取 SDK.....	4
4	SDK 编译开发 .....	5
4.1	软件开发指南 .....	5
4.2	JDK 安装 .....	5
4.3	jack-server 配置 .....	5
4.4	编译模式 .....	6
4.4.1	uboot 编译步骤 .....	6
4.4.2	kernel 编译步骤 .....	6
4.4.3	Android 编译步骤 .....	6
4.4.4	自动化编译 .....	7
5.	镜像刷机说明 .....	8
附录 A	编译开发环境搭建.....	10
附录 B	SSH 公钥操作说明.....	14
附录 B-1	SSH 公钥生成.....	14
附录 B-2	使用 key-chain 管理密钥 .....	14
附录 B-3	多台机器使用相同 ssh 公钥.....	15
附录 B-4	一台机器切换不同 ssh 公钥.....	16
附录 B-5	密钥权限管理.....	17
附录 B-6	Git 权限申请说明 .....	17

## 1 概述

本 SDK 是基于谷歌 Android7.1 32 bit 系统，适配瑞芯微 RK312x 芯片的软件包，适用于 RK3126-86V-RK818 样机、RK3128 SDK 板以及基于它们所开发的产品。

## 2 主要支持功能

参数	模块名
数据通信	Wi-Fi、USB、SDCARD
应用程序	浏览器、计算器、相机、资源管理器、设置、视频播放器、图库、音乐播放器、Email、录音等

## 3 SDK 获取说明

### 3.1 获取 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。

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。

下载方式：

```
cd ~
git clone ssh://git@www.rockchip.com.cn/repo/rk/tools/repo
~/repo/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-7.0 -m rk312x_tablet_nougat_release.xml
~/repo/repo sync -c --no-tags
```

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，该压缩包解压得到的源码，与通过 repo 下载的源码是一致的。

以 rk312x\_android7.1\_v1.00\_20170519.tgz 为例，拷贝到该初始化包后，通过如下命令可检出源码：

```
mkdir rk312x
tar zxvf rk312x_android7.1_v1.00_20170519.tgz -C rk3126
cd rk312x
.repo/repo/repo sync -l
.repo/repo/repo sync -c --no-tags
```

开发者可通过.repo/repo/repo sync -c --no-tags 命令获取 SDK 最新源码。

## 4 SDK 编译开发

### 4.1 软件开发指南

为帮助开发工程师快速熟悉开发调试工作，SDK 提供《RK312X\_Android7.1\_软件开发指南 V1.00-20170518》文档，同时提供相关具体模块的开发指导文档。

文档位于 RKDocs 目录，其内容不定期更新。

### 4.2 JDK 安装

Android7.1 系统编译依赖于 JAVA 8。Open-JDK8 安装可参考如下命令：

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

安装完成后，需配置 JAVA 环境变量。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。

```
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。开发人员需根据实际 JDK 实际安装路径对应修改脚本中环境变量。

脚本配置好后，可执行以下命令，完成配置 JDK：

```
source javaenv.sh
```

### 4.3 jack-server 配置

Android7.1 系统使用 jack-server 作为 java 代码编译器。多个用户在同一个服务器上协同开发过程中，需各自配置独立的网络端口号，具体请参考《Rockchip RK312X Android 7.1 软件开发指南》章节 3.2.7 相关说明。

## 4.4 编译模式

SDK 默认以 userdebug 模式编译。

使用 adb 调试时，需先执行 adb root 使 shell 获取 root 权限，才可执行其它调试命令，如 adb remount、adb push 等。

WITH\_DEXPREOPT 宏为系统编译优化选项，开启此宏，系统会在编译阶段就执行 app 的 dex2oat 操作加快系统首次开机速度。

WITH\_DEXPREOPT 宏在 device/rockchip/rk312x/BoardConfig.mk 中配置，开发阶段不建议开启，以方便进行调试开发，详见《Rockchip RK312X Android 7.1 软件开发指南》章节 3.2.6.1 相关说明。

### 4.4.1 uboot 编译步骤

```
make rk3126_secure_defconfig  
make
```

编译后会生成 rk3126\_loader\_v2.0x.xxx.bin、uboot.img、trust.img 三个文件。

### 4.4.2 kernel 编译步骤

RK3126 816 开发板配置与编译如下：

```
make rockchip_defconfig  
make rk3126-86v-rk816.img
```

RK3126 816-rev1 样机配置与编译如下：

```
make rockchip_defconfig  
make rk3126-86v-rk816-rev1.img
```

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

### 4.4.3 Android 编译步骤

开发者参考附录 A 搭建开发环境，配置好 JDK 环境变量后，即可编译 Android 代码。

按照以下步骤配置完后，执行 make 即可。

```
$ source build/envsetup.sh  
$ lunch rk312x-userdebug  
$ make
```

完成编译后，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-rk312x 目录。

```
rockdev/ Image-rk312x
├── boot.img
├── kernel.img
├── misc.img
├── parameter.txt
├── pcba_small_misc.img
├── pcba_whole_misc.img
├── recovery.img
├── resource.img
├── system.img
├── trust.img
└── uboot.img
```

得到了所有镜像文件后，为了方便烧写及量产，通常可手动将这些单独的镜像通过脚本打包成为 update.img。

#### 4.4.4 自动化编译

SDK 中集成了全自动化编译脚本，方便固件编译、备份。

1) 该脚本位于：

```
device/rockchip/rk312x/build.sh
```

2) 在 repo sync 的时候，通过 manifest 自动拷贝到工程根目录。

3) 修改 build.sh 脚本中的特定变量以编出对应产品固件。

```
KERNEL_DTS= rk3126-86v-rk816-rev1
```

变量请按实际项目情况，对应修改。

Android 默认编译为 rk3126c-userdebug 模式，可根据实际项目在脚本中修改配置：

```
lunch rk3126c-userdebug
```

4) 指定 update.img 打包用的 loader:

如 RKTools\linux\Linux\_Pack\_Firmware\rockdev\mkupdate.sh 脚本所示：

```
17 ./afptool -pack ./ Image/update.img || pause
18 ./rkImageMaker -RK312A Image/MiniLoaderAll.bin Image/update.img update.img
19 echo "Making update.img OK."
20 #echo "Press any key to quit:"
21 #read -n1 -s key
22 exit $?
```

update.img 打包用的 loader 被命名位 MiniLoaderAll.bin，由于 SDK 更新兼容 Loader，所以在此通过 u-boot 目录编译生成 rk312x\_loader\_vxx.xx.bin（拷贝时会重命名为 MiniLoaderAll.bin），需要指定脚本中 loader 文件名。

#### 5) 执行自动编译脚本

```
source build.sh
```

该脚本会自动配置 JDK 环境变量，编译 u-boot, kernel, Android，然后生成固件，并打包成 update.img。

#### 6) 脚本生成内容

脚本会将编译生成的固拷贝到：

IMAGE/RK312x-XXX\_7.1.2\_XXX\_RELEASE\_TEST/IMAGES 目录下，具体路径以实际生成为准。每次编译都会根据日期时间生成新的目录保存，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。

该目录下的 update.img 可直接用于 Android 开发工具及工厂烧写工具下载更新。

## 5. 镜像刷机说明

SDK 提供固件镜像刷机工具，位于 RKTools\windows\AndroidTool。工具的使用介绍文档，请参考 RKDocs\ RKTools manuals\Android 开发工具手册.pdf。

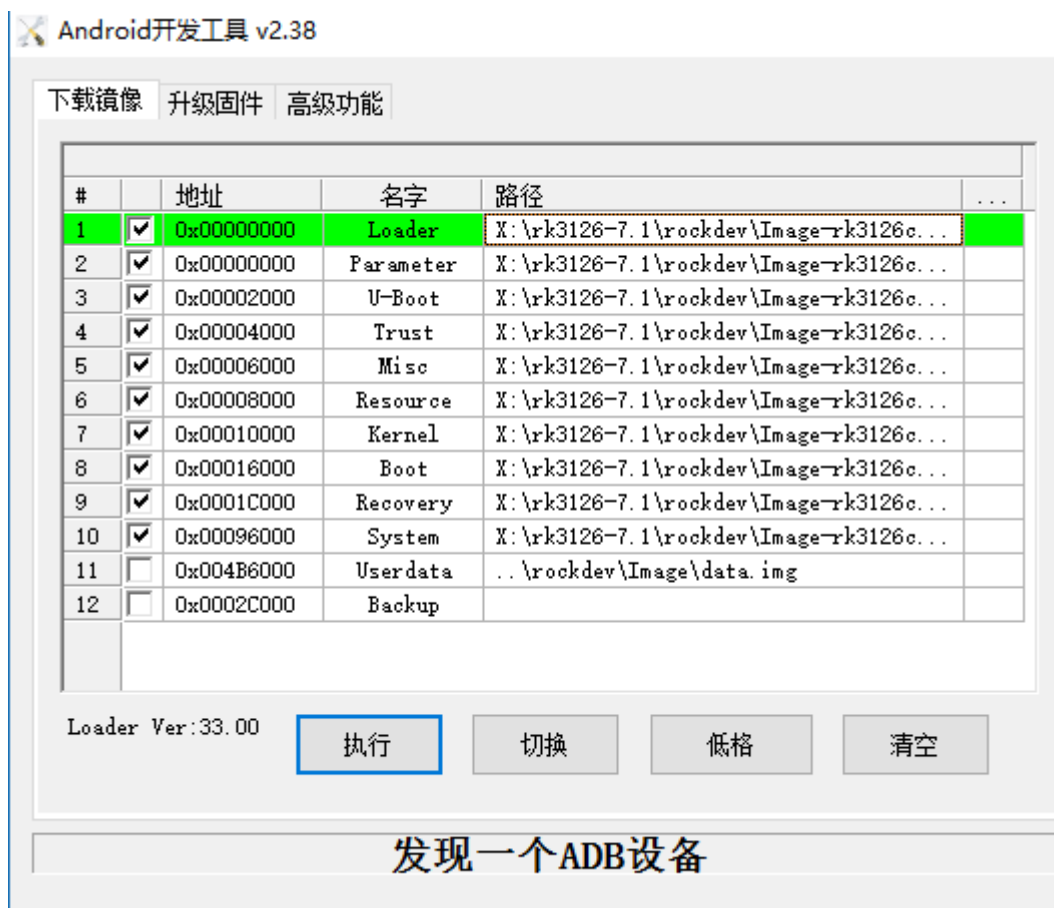
编译完生成相应的固件后，进入刷机模式，即可进行刷机，俗称“烧写”。对于刷过其它固件的机器，可以选择重新烧录固件，或选择低格设备，擦除 idb，然后进行刷机。

注：刷机之前，需安装最新的 USB 驱动程序，文件位于目录：

```
RKTools/windows/  
└── DriverAssitant_v4.5
```

镜像刷机工具 AndroidTool 的图示如下。





镜像刷机工具 AndroidTool 示意图

## 附录 A 编译开发环境搭建

### 1. Initializing a Build Environment

This section 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.

*Note: The source download is approximately 35GB in size. You will need over 45GB free to complete a single build, and up to 100GB (or more) for a full set of builds.*

For an overview of the entire code-review and code-update process, see [Life of a Patch](#).

### 2. Choosing a Branch

Some of the requirements for your build environment are determined by which version of the source code you plan to compile. See [Build Numbers](#) for a full listing of branches you may choose from. You may also choose to download and build the latest source code (called "master"), in which case you will simply omit the branch specification when you initialize the repository.

Once you have selected a branch, follow the appropriate instructions below to set up your build environment.

### 3. Setting up a Linux build environment

These instructions apply to all branches, including master.

The Android build is routinely tested in house on recent versions of Ubuntu LTS (10.04), but most distributions should have the required build tools available. Reports of successes or failures on other distributions are welcome.

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

*Note: It is also possible to build Android in a virtual machine. If you are running Linux in a virtual machine, you will need at least 16GB of RAM/swap and 30GB*

*or more of disk space in order to build the Android tree.*

Detailed instructions for Ubuntu and MacOS follow. In general you will need:

- A. Python 2.6 -- 2.7, which you can download from [python.org](http://python.org).
- B. GNU Make 3.81 -- 3.82, which you can download from [gnu.org](http://gnu.org),
- C. JDK 6 if you wish to build Gingerbread or newer; JDK 5 for Froyo or older. You can download both from [java.sun.com](http://java.sun.com).
- D. Git 1.7 or newer. You can find it at [git-scm.com](http://git-scm.com).

#### **4. Installing the JDK**

**The master branch of Android in the Android Open Source Project (AOSP) requires Java 7. On Ubuntu, use OpenJDK. Java 7: For the latest version of Android**

**\$ sudo apt-get update**

**\$ sudo apt-get install openjdk-7-jdk**

**Optionally, update the default Java version by running:**

**\$ sudo update-alternatives --config java**

**\$ sudo update-alternatives --config javac**

**If you encounter version errors for Java, set its path as described in the Wrong Java Version section.**

**To develop older versions of Android, download and install the corresponding version of the Java JDK:**

**Java 6: for Gingerbread through KitKat**

**Java 5: for Cupcake through Froyo**

## 5. Installing required packages (Ubuntu 12.04)

You will need a 64-bit version of Ubuntu. Ubuntu 12.04 is recommended. Building using an older version of Ubuntu is not 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:i386

$ sudo ln -s /usr/lib/i386-linux-gnu/mesa/libGL.so.1 /usr/lib/i386-linux-gn
u/libGL.so
```

## 6. Installing required packages (Ubuntu 10.04 -- 11.10)

Building on Ubuntu 10.04-11.10 is no longer supported, but may be useful for building older releases of AOSP.

```
$ sudo apt-get install git-core gnupg flex bison gperf build-essential \
zip curl zlib1g-dev libc6-dev lib32ncurses5-dev ia32-libs \
x11proto-core-dev libx11-dev lib32readline5-dev lib32z-dev \
libgl1-mesa-dev g++-multilib mingw32 tofrodos python-markdown \
libxml2-utils xsltproc
```

On Ubuntu 10.10:

```
$ sudo ln -s /usr/lib32/mesa/libGL.so.1 /usr/lib32/mesa/libGL.so
```

On Ubuntu 11.10:

```
$ sudo apt-get install libx11-dev:i386
```

## 7. Configuring USB Access

Under GNU/Linux systems (and specifically under Ubuntu systems), regular users can't directly access USB devices by default. The system needs to be configured to allow such access.

The recommended approach is to create a file `/etc/udev/rules.d/51-android.rules` (as the root user) and to copy the following lines in it. `<username>` must be replaced by the actual username of the user who is authorized to access the phones over USB.

```
# adb protocol on passion (Rockchip products)
SUBSYSTEM=="usb", ATTR{idVendor}=="2207",
ATTR{idProduct}=="0010", MODE="0600", OWNER="<username>"
```

Those new rules take effect the next time a device is plugged in. It might therefore be necessary to unplug the device and plug it back into the computer.

This is known to work on both Ubuntu Hardy Heron (8.04.x LTS) and Lucid Lynx (10.04.x LTS). Other versions of Ubuntu or other variants of GNU/Linux might require different configurations.

References : <http://source.android.com/source/initializing.html>

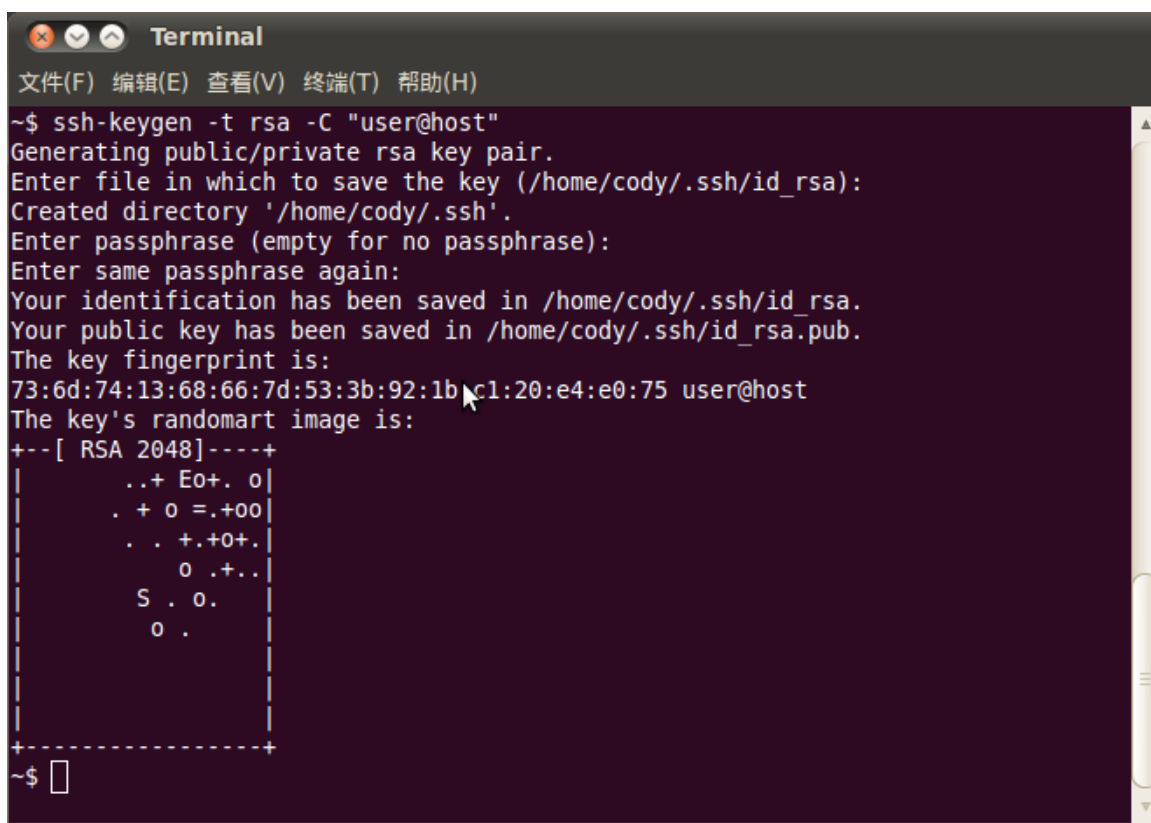
## 附录 B SSH 公钥操作说明

### 附录 B-1 SSH 公钥生成

使用如下命令生成：

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

请将 **user@host** 替换成您的邮箱地址。



```
Terminal
文件(F) 编辑(E) 查看(V) 终端(T) 帮助(H)
~$ ssh-keygen -t rsa -C "user@host"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/cody/.ssh/id_rsa):
Created directory '/home/cody/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/cody/.ssh/id_rsa.
Your public key has been saved in /home/cody/.ssh/id_rsa.pub.
The key fingerprint is:
73:6d:74:13:68:66:7d:53:3b:92:1b:c1:20:e4:e0:75 user@host
The key's randomart image is:
+--[ RSA 2048 ]-----+
|      .+ Eo+. o |
|      . + o =. +oo |
|      . . +. +o+. |
|      o .+. . |
|      S . o. |
|      o . |
+-----+
~$
```

命令运行完成会在你的目录下生成 key 文件。

```
~$ 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。

### 附录 B-2 使用 key-chain 管理密钥

推荐您使用比较简易的工具 **keychain** 管理密钥。

具体使用方法如下：

1. 安装 keychain 软件包：

```
$sudo aptitude install keychain
```

2. 配置使用密钥：

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

增加下面这行：

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

其中，id\_rsa 是私钥文件名称。

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。

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。

## 附录 B-3 多台机器使用相同 ssh 公钥

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。

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。

```
~/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。

```
~$ 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。

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

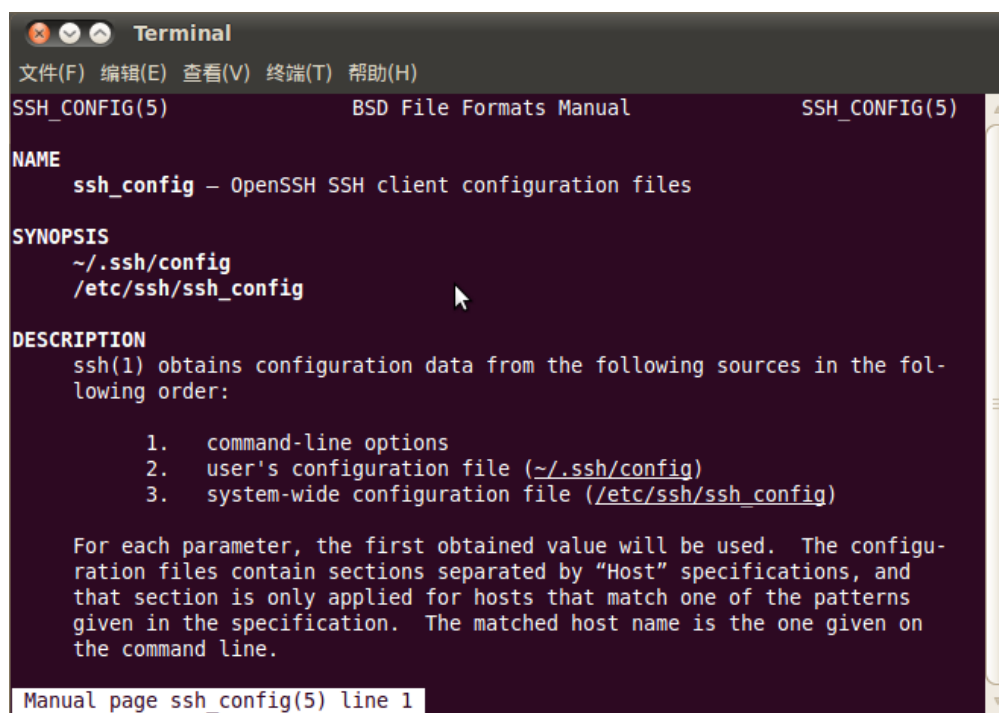
在 console 输入如下命令即可解决。

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

## 附录 B-4 一台机器切换不同 ssh 公钥

可以参考 ssh\_config 文档配置 ssh。

```
~$ man ssh_config
```



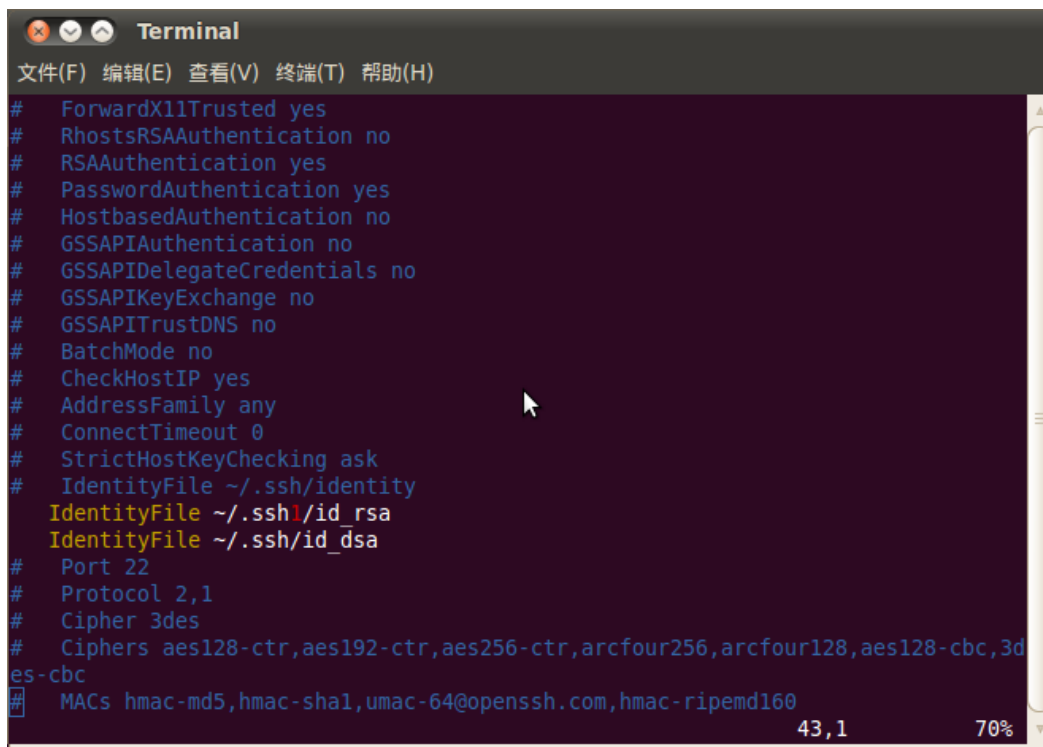
通过如下命令，配置当前用户的 ssh 配置。

```
~$ 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。



```
Terminal
文件(F) 编辑(E) 查看(V) 终端(T) 帮助(H)

# ForwardX11Trusted yes
# RhostsRSAAuthentication no
# RSAAuthentication yes
# PasswordAuthentication yes
# HostbasedAuthentication no
# GSSAPIAuthentication no
# GSSAPIDelegateCredentials no
# GSSAPIKeyExchange no
# GSSAPITrustDNS no
# BatchMode no
# CheckHostIP yes
# AddressFamily any
# ConnectTimeout 0
# StrictHostKeyChecking ask
# IdentityFile ~/.ssh/identity
IdentityFile ~/.ssh1/id_rsa
IdentityFile ~/.ssh/id_dsa
# Port 22
# Protocol 2,1
# Cipher 3des
# Ciphers aes128-ctr,aes192-ctr,aes256-ctr,arcfour256,arcfour128,aes128-cbc,3des-cbc
# MACs hmac-md5,hmac-sha1,umac-64@openssh.com,hmac-ripemd160

43,1 70%
```

## 附录 B-5 密钥权限管理

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。

请妥善保管私钥文件。并不要二次授权与第三方使用。

## 附录 B-6 Git 权限申请说明

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