

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

# RK3308 Linux SDK 发布说明

(技术部，第三系统产品部)

<div>文件状态：</div> <div><div><input type="checkbox"/> 草稿</div><div><input type="checkbox"/> 正在修改</div><div><input checked="" type="checkbox"/> 正式发布</div></div>	文件标识：	RK-FB-CS-012
	当前版本：	1.3.2
	作     者：	YHX、HKH
	完成日期：	2019-10-31
	审     核：	ZYY
	审核日期：	2019-10-31

福州瑞芯微电子股份有限公司

Fuzhou Rockchip Electronics Co.,Ltd.



## 版本历史

版本号	作者	修改日期	修改说明	备注
Pre-Alpha	YHX	2018-04-01	正式发布	
V1.0.0	YHX	2018-05-10	正式发布	
V1.3.2	HKH	2019-10-31	正式发布	

# 目 录

<b>1</b>	<b>基础信息.....</b>	<b>3</b>
1.1	Kernel 版本.....	3
1.2	驱动支持列表.....	3
1.3	支持的硬件板型.....	4
<b>2</b>	<b>软件开发指南.....</b>	<b>5</b>
<b>3</b>	<b>编译环境需求.....</b>	<b>5</b>
3.1	概述.....	5
3.2	Linux 服务器开发环境搭建 .....	6
3.2.1	发布包使用的 Linux 服务器系统版本 .....	6
3.3	网络环境搭建.....	6
3.3.1	软件包安装.....	6
<b>4</b>	<b>SDK 获取说明 .....</b>	<b>7</b>
4.1	安装 repo.....	7
4.2	配置 git .....	7
4.3	SDK 获取.....	7
4.3.1	SDK 下载命令 .....	7
4.3.2	SDK 代码压缩包.....	8
<b>5</b>	<b>SDK 编译说明 .....</b>	<b>8</b>
5.1	U-Boot 编译步骤.....	8
5.2	Kernel 编译步骤.....	8
5.3	Buildroot 编译步骤 .....	10
5.4	全自动编译脚本.....	11
5.5	Robot 配置和编译.....	12
<b>6</b>	<b>SDK 镜像烧写 .....</b>	<b>12</b>
附录 A	SSH 公钥操作说明 .....	13
附录 A-1	SSH 公钥操作说明 .....	13
附录 A-2	使用 key-chain 管理密钥 .....	13
附录 A-3	多台机器使用相同 SSH 公钥.....	13
附录 A-4	一台机器切换不同 SSH 公钥.....	14
附录 A-5	密钥权限管理.....	15
附录 A-6	git 权限申请说明.....	15

## 概述

本 SDK 是基于 Buildroot-2018.02 版本的软件开发包, 其包含 Linux 系统开发用到的系统源码, 驱动, 工具, 应用软件包。适配瑞芯微 RK3308 芯片平台, 适用于 RK3308 EVB 开发板及基于 RK3308 平台开发的所有产品。

# 1 基础信息

## 1.1 Kernel 版本

Kernel 版本为: Linux 4.4.120

```
commit 47356cfded444826565f2430bce8ba294372b861
Author: Greg Kroah-Hartman <gregkh@linuxfoundation.org>
Date:   Sat Mar 3 10:19:46 2018 +0100

Linux 4.4.120
```

## 1.2 驱动支持列表

```
1) Timer/Interrupts
2) Clocks
3) PinMux/GPIO/GPIO IRQ
4) UART
5) I2S/PCM/TDM
6) PDM
7) USB Host/OTG
8) CPU DVFS
9) TSADC
10) VAD
11) SDMMC
12) eMMC
13) OTP
14) SPI
15) SARADC/ADC KEY
16) I2C
17) PWM
18) SDIO WiFi
19) Bluetooth
```

- 20) LED
- 21) Ethernet
- 22) SecureOS
- 23) SecureBoot
- 24) Crypto
- 25) SPDIF

### 1.3 支持的硬件板型

本节主要列出目前 RK3308 SDK 所支持的硬件板型，并列出对应硬件板型功能及外围设备。

#### **RK3308 EVB Board:**

● Rockchip RK3308 CPU
● 512MB DDR3 (K4B4G1646E-BCMA)
● eMMC Flash/Nand Flash/SPI Nor Flash 兼容设计
● V10 默认贴 8GB eMMC (KLM8G1GEAC-B041)
● V11 默认贴 256MB NandFlash (K4B2G1646F-BYK0)
● V12 默认贴 128MB NandFlash (GD9FU2G8F2AMG)
● V13 默认贴 128MB NandFlash (GD9FU2G8F2AMG)
● USB 2.0 OTG&USB 2.0 HOST
● SDIO WiFi/BT (AP6255)
● SPDIF IN/SPDIF OUT
● LINE OUT (外接 PA+喇叭)
● HPOUT (外接耳机)
● MIC-Analog Interface (RK3308 自带 8 路 ADC)
● MIC-Digital Interface (RK3308 I2S0 接口支持 8 输入 8 输出)
● 18bit RGB Panel/MCU Panel
● ADC Key *6
● 支持 10/100M Ethernet (RTL8201F)
● TF 卡
● UART Interface
● POWER (12V/2A)

#### **带屏方案硬件搭配:**

RK\_EVB\_RK3308\_DDR3P116SD4\_V13+RK\_EVB\_RK3308B\_MIPI\_DisplayV10\_20190708

## 2 软件开发指南

为帮助开发工程师方便的搭建开发环境，快速上手熟悉 SDK 的开发调试工作，随 SDK 发布《Rockchip\_Developer\_Guide\_Linux\_Software\_CN.pdf》。

SDK 下载完成后，可在 docs\目录下获取，并会不断完善更新。

## 3 编译环境需求

### 3.1 概述

本节主要介绍了如何在本地搭建编译环境来编译 RK3308 Linux SDK 源代码。当前 SDK 只支持在 Linux 环境下编译，并提供 Linux 下的交叉编译工具链。

一个典型的嵌入式开发环境通常包括 Linux 服务器、Windows PC 和目标硬件板，以 RK3308 为例，典型开发环境如图所示。

- Linux 服务器上建立交叉编译环境，为软件开发提供代码更新下载，代码交叉编译服务。
- Windows PC 和 Linux 服务器共享程序，并安装 SecureCRT 或 puTTY，通过网络远程登录到 Linux 服务器，进行交叉编译，及代码的开发调试。
- Windows PC 通过串口和 USB 与目标硬件板连接，可将编译后的镜像文件烧写到目标硬件板，并调试系统或应用程序。

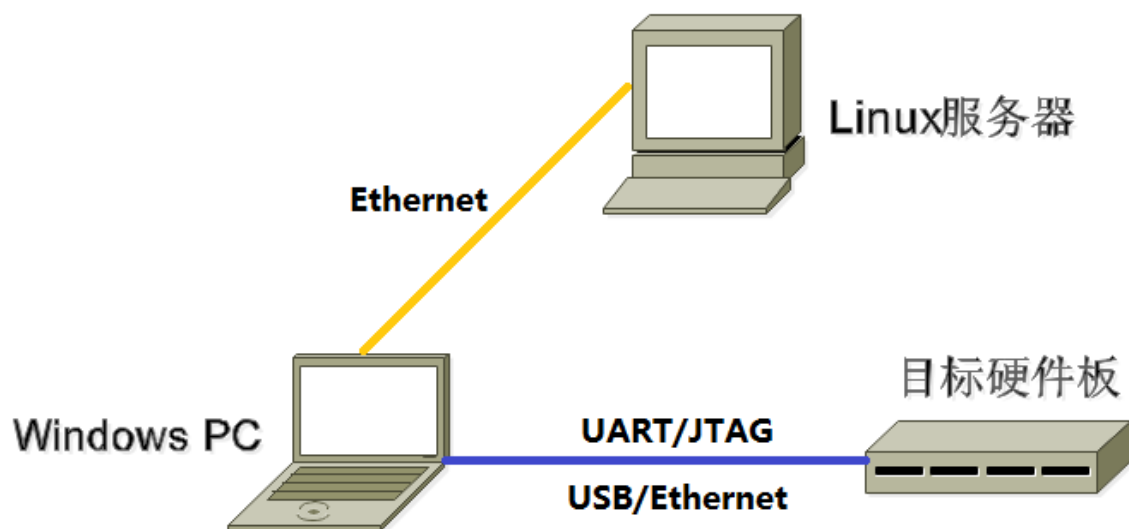


图 3-1 典型开发环境

注：开发环境中使用了 Windows PC，实际上很多工作也可以在 Linux PC 上完成，如使用 minicom 代替 SecureCRT 或 puTTY 等，用户可自行选择。

## 3.2 Linux 服务器开发环境搭建

Rockchip Buildroot Linux SDK 是在 Ubuntu 16.04 上开发测试的。因此，我们推荐使用 Ubuntu 16.04 的系统进行编译。其他版本没有具体测试，可能需要对软件包做相应调整。

除了系统要求外，还有其他软硬方面的要求。

- 硬件要求：64 位系统，硬盘空间大于 40G。如果您进行多个构建，将需要更大的硬盘空间。
- 软件包依赖：除了 python 2.7，make 3.8，git 1.7 之外，还需要安装一些额外的软件包，将在软件包安装章节中列出。

### 3.2.1 发布包使用的 Linux 服务器系统版本

本 SDK 开发环境安装如下版本 Linux 系统，SDK 默认均以此 Linux 系统进行编译：

```
Ubuntu 16.04.2 LTS
Linux version 4.4.0-62-generic (buildd@lcy01-30) (gcc version 5.4.0
20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.4) ) #83-Ubuntu SMP Wed Jan 18
14:10:15 UTC 2017
```

## 3.3 网络环境搭建

请用户自行配置网络，并安装 nfs，samba，ssh 等网络组件。

### 3.3.1 软件包安装

操作系统安装好后，且用户已自行配置好网络环境，则可继续如下步骤完成相关软件包的安装。

#### 1. apt-get update

```
sudo apt-get update
```

#### 2. 安装 Kernel 及 U-Boot 编译需要依赖的软件包

```
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 device-tree-compiler
```

#### 3. 安装 Buildroot 编译需要依赖的软件包

```
sudo apt-get install libfile-which-perl sed make binutils gcc g++ bash
patch gzip bzip2 perl tar cpio python unzip rsync file bc libmpc3 git repo
texinfo pkg-config cmake tree realpath
```

若编译遇到报错，可以视报错信息，安装对应的软件包。

## 4 SDK 获取说明

### 4.1 安装 repo

确保主目录下有一个 `bin/` 目录，并且该目录包含在路径中：

```
mkdir ~/bin
export PATH=~/bin:$PATH
```

如果可以访问 google 的地址，下载 Repo 工具，并确保它可执行：

```
curl https://storage.googleapis.com/git-repo-downloads/repo >
~/bin/repo
chmod a+x ~/bin/repo
```

中国国内环境如果执行上述命令后发现 `~/bin/repo` 为空，此时可以访问国内的站点来下载 repo 工具

```
curl https://mirrors.tuna.tsinghua.edu.cn/git/git-repo -o ~/bin/repo
chmod a+x ~/bin/repo
```

除以上两种方式外，也可以使用如下命令获取 repo

```
sudo apt-get install repo
```

### 4.2 配置 git

在使用 repo 之前请配置一下自己的 git 信息，否则后面的操作可能会遇到 hook 检查的障碍

```
git config --global user.name "your name"
git config --global user.email "your mail"
```

### 4.3 SDK 获取

SDK 通过瑞芯微代码服务器对外发布。其编译开发环境，参考[第 3 节 编译环境需求](#)。

获取 RK3308 Linux 软件包，需要有一个帐户访问 Rockchip 提供的源代码仓库。客户向瑞芯微技术窗口申请 SDK，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 A SSH 公钥操作说明](#)。

#### 4.3.1 SDK 下载命令

RK3308\_LINUX\_SDK 下载命令如下：

```
mkdir rk3308
cd rk3308
repo init --repo-url ssh://git@www.rockchip.com.cn/repo/rk/tools/repo
```



```
-u ssh://git@www.rockchip.com.cn/linux/rk/platform/manifests -b linux -m
rk3308_linux_release.xml
.repo/repo/repo sync -c
```

代码将开始自动下载，后面只需耐心等待。源代码文件将位于工作目录中对应的项目名称下。初始同步操作将需要 1 个小时或更长时间才能完成。

### 4.3.2 SDK 代码压缩包

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。

以 rk3308\_linux\_v1.00\_20180510.tgz 为例，拷贝到该初始化包后，通过如下命令可检出源码：

```
mkdir rk3308
tar xvf rk3308_linux_v1.00_20180510.tgz -C rk3308
cd rk3308
.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。

## 5 SDK 编译说明

### 5.1 U-Boot 编译步骤

```
cd u-boot
./make.sh evb-rk3308
```

编译完成后，u-boot 根目录，生成 trust.img、rk3308\_loader\_v1.17.101.bin、uboot.img 三个镜像文件。

### 5.2 Kernel 编译步骤

- RK3308 EVB V10 开发板硬件信息如下：

硬件版本	板上丝印	参考设计
V10	RK_EVB_RK3308_DDR3P116SD4_V10_20180301	RK3308_AI-VA_BETA_V01_20180307

RK3308 EVB V10 开发板搭配不同的麦克风阵列小板，需要选用不同的板级配置文件，区分如下：

麦克风阵列小板	对应板级配置文件	编译命令
I2S 数字麦克风	rk3308-evb-dmic-i2s-v10.dts	cd kernel make rk3308_linux_defconfig make rk3308-evb-dmic-i2s-v10.img
模拟麦克风	rk3308-evb-amic-v10.dts	cd kernel make rk3308_linux_defconfig make rk3308-evb-amic-v10.img
PDM 数字麦克风	rk3308-evb-dmic-pdm-v10.dts	cd kernel make rk3308_linux_defconfig make rk3308-evb-dmic-pdm-v10.img

- RK3308 EVB V11 开发板硬件信息如下：

硬件板本	板上丝印	参考设计
V11	RK_EVB_RK3308_DDR3P116SD4_V11_20180420	RK3308_AI-VA_REF_V10

RK3308 EVB V11 开发板搭配不同的麦克风阵列小板，需要选用不同的板级配置文件，区分如下：

麦克风阵列小板	对应板级配置文件	编译命令
I2S 数字麦克风	rk3308-evb-dmic-i2s-v11.dts	cd kernel make rk3308_linux_defconfig make rk3308-evb-dmic-i2s-v11.img
模拟麦克风	rk3308-evb-amic-v11.dts	cd kernel make rk3308_linux_defconfig make rk3308-evb-amic-v11.img
PDM 数字麦克风	rk3308-evb-dmic-pdm-v11.dts	cd kernel make rk3308_linux_defconfig make rk3308-evb-dmic-pdm-v11.img

编译完成后，kernel 根目录，生成 boot.img 镜像文件。

- RK3308 EVB V13 开发板硬件信息如下：

硬件板本	板上丝印	参考设计
V13	RK_EVB_RK3308_DDR3P116SD4_V13	RK3308_AI-VA_REF_V13_20190215

RK3308 EVB V11 开发板搭配不同的麦克风阵列小板，需要选用不同的板级配置文件，区分如下：

麦克风陈列小板	对应板级配置文件	编译命令
I2S 数字麦克风	rk3308-evb-dmic-i2s-v13.dts	cd kernel make rk3308_linux_defconfig make rk3308-evb-dmic-i2s-v13.img
模拟麦克风	rk3308-evb-amic-v13.dts	cd kernel make rk3308_linux_defconfig make rk3308-evb-amic-v13.img
PDM 数字麦克风	rk3308-evb-dmic-pdm-v13.dts	cd kernel make rk3308_linux_defconfig make rk3308-evb-dmic-pdm-v13.img
PDM 麦+MIPI 显示屏扩展板	rk3308-evb-mipi-display-v10.dts	cd kernel make rk3308_linux_rk618_display_defconfig make rk3308-evb-mipi-display-v10.img

### 5.3 Buildroot 编译步骤

客户配置好编译环境后，按照以下步骤配置完后，执行 **make** 即可。

```
$ source buildroot/build/envsetup.sh
```

```
You're building on Linux
Lunch menu...pick a combo:
1. rockchip_rk3308_release
2. rockchip_rk3308_debug
3. rockchip_rk3308_robot_release
4. rockchip_rk3308_robot_debug
5. rockchip_rk3308_mini_release
```

```
Which would you like? [1]
```

如选择 **rockchip\_rk3308\_release**，输入对应序号 1。

```
$ make
```

完成编译后，执行 **SDK** 根目录下的 **mkfirmware.sh** 脚本生成固件，所有烧写所需的镜像将都会拷贝于 **rockdev/Image-rk3308** 目录。

```
rockdev/Image-rk3308
├─ boot.img
├─ misc.img
├─ parameter.txt
├─ recovery.img
├─ MiniLoaderAll.bin (即 rk3308_loader_v1.29.119.bin)
└─ data.img
```

```
├─ cfg.img
├─ rootfs.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一个 `update.img`，若使用全自动编译脚本会自动打包 `update.img` 出来。

## 5.4 全自动编译脚本

为了提高编译的效率，降低人工编译可能出现的误操作，该 SDK 中集成了全自动化编译脚本，方便固件编译、备份。

1) 该全自动化编译脚本原始文件存放于：

```
device/rockchip/rk3308/build.sh
```

2) 在 `repo sync` 的时候，通过 `manifest` 中的 `copy` 选项拷贝至工程根目录下：

3) 修改 `build.sh` 脚本中的特定变量以编出对应产品固件。

```
#buildroot defconfig
LUNCH=rockchip_rk3308_release
#uboot defconfig
UBOOT_DEFCONFIG=rk3308
#kernel defconfig
KERNEL_DEFCONFIG=rk3308_linux_defconfig
#kernel dts
KERNEL_DTS=rk3308-evb-dmic-pdm-v13
```

以下变量请按实际项目情况，对应修改：

`LUNCH` 变量指定 Buildroot 编译 `defconfig`。

`KERNEL_DTS` 变量指定编译 Kernel 的产品板极配置。

4) 执行自动编译脚本：

```
./build.sh
```

该脚本会自动配置环境变量，编译 U-Boot，编译 Kernel，编译 Buildroot，继而生成固件。

5) 脚本生成内容：

脚本会将编译生成的固件拷贝至：

`IMAGE/RK3308-EVB-DMIC-PDM-V13_****_RELEASE_TEST/IMAGES` 目录下，具体路径以实际生成为准。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。

## 5.5 Robot 配置和编译

对于 Robot 开发者，我们提供了针对 Robot 裁减的 BoardConfig 板级配置，在 /device/rockchip/rk3308/BoardConfig\_robot32.mk，配置中 buildroot 删去 QT, App 等 UI 显示相关配置，大大降低了固件大小，适用于无屏幕、小容量 Robot 产品开发者使用。

RK3308 Linux Robot SDK 下载命令如下：

```
mkdir rk3308
cd rk3308
repo init --repo-url ssh://git@www.rockchip.com.cn/repo/rk/tools/repo
-u ssh://git@www.rockchip.com.cn/linux/rk/platform/manifests -b linux -m
rk3308_robot_release.xml
.repo/repo/repo sync -c
```

如果已经下载了 RK3308 Linux SDK，可以通过如下命令切换到 RK3308 Linux Robot SDK：

```
.repo/repo/repo init -m rk3308_robot_release.xml
.repo/repo/repo sync -c
```

## 6 SDK 镜像烧写

SDK 镜像烧写说明详见 docs\目录下《Rockchip\_Developer\_Guide\_Linux\_Software\_CN.pdf》第 7 章 SDK 镜像烧写。

## 附录 A SSH 公钥操作说明

### 附录 A-1 SSH 公钥操作说明

请根据《Rockchip SDK 申请及同步指南》文档说明操作，生成 SSH 公钥，发邮件至 fae@rock-chips.com，申请开通 SDK 代码。

该文档会在申请开通权限流程中，释放给客户使用。

### 附录 A-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 A-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
```

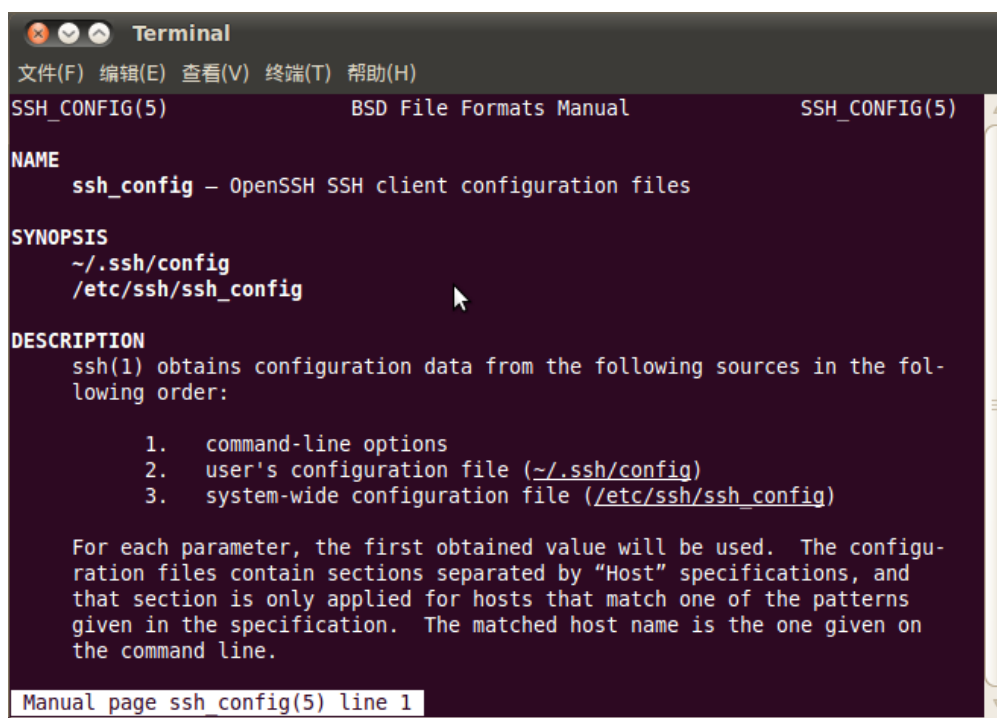
在 console 输入如下命令即可解决。

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

## 附录 A-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
```



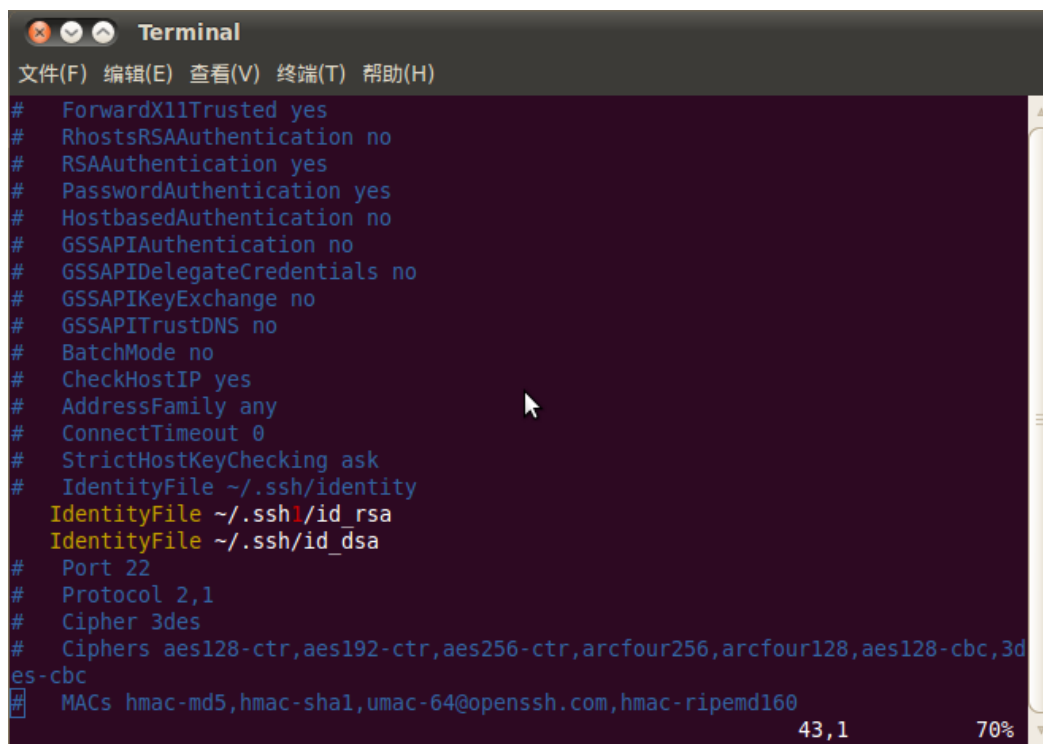
通过如下命令，配置当前用户的 ssh 配置。

```

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

```

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

A terminal window titled "Terminal" with a menu bar containing "文件(F)", "编辑(E)", "查看(V)", "终端(T)", and "帮助(H)". The terminal displays a list of SSH configuration options, each preceded by a hash symbol (#). The options are: 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 ~/.ssh/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, and MACs hmac-md5,hmac-sha1,umac-64@openssh.com,hmac-ripemd160. The status bar at the bottom right shows "43,1" and "70%".

```
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 ~/.ssh/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 A-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 A-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。