

密级状态： 绝密() 秘密() 内部资料() 公开(☒)

PX30 Linux SDK 发布说明

(技术部，第三系统产品部)

文件状态： <input type="checkbox"/> 草稿 <input type="checkbox"/> 正在修改 <input checked="" type="checkbox"/> 正式发布	文件标识：	RK-FB-CS-005
	当前版本：	1.3.0
	作 者：	Ziyuan Xu
	完成日期：	2020-02-24
	审 核：	Eddie Cai
	审核日期：	2020-03-02

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



文档修改记录

日期	修订版本	修订内容	修改人	核定人
2019-04-25	V1.1.0	初始版本	Ziyuan Xu	Eddie Cai
2019-09-17	V1.2.0	1. 更新 Linux_SDK_v1.2.0 说明 2. 更新应用编译说明 3. 新增 github 源码获取说明	Ziyuan Xu	Eddie Cai
2020-02-24	V1.3.0	1. 完善 robot 说明 2. Debian10 说明	Ziyuan Xu	Eddie Cai

目录

1 概述.....	4
2 主要支持功能.....	4
3 SDK 获取说明.....	4
4 软件开发指南.....	6
5 PX30 Linux 工程目录介绍.....	6
6 SDK 编译说明.....	6
6.1 U-Boot 编译.....	6
6.2 Kernel 编译步骤.....	7
6.3 Recovery 编译步骤.....	7
6.4 Rootfs 系统编译.....	7
6.5 重新编译 APP.....	7
6.6 全自动编译.....	7
6.7 Buildroot 切换 32 位配置.....	9
6.8 Robot 配置及编译.....	10
6.9 固件的打包.....	10
6.10 debian10 编译.....	10
7 刷机说明.....	10
7.1 Windows 刷机说明.....	11
7.2 Linux 刷机说明.....	12
7.3 系统分区说明.....	13
8 调试串口的参数设置.....	13
9 固件下载.....	14
10 SSH 公钥操作说明.....	14
10.1 SSH 公钥生成.....	14
10.2 使用 key-chain 管理密钥.....	15
10.3 多台机器使用相同 SSH 公钥.....	15
10.4 一台机器切换不同 SSH 公钥.....	16
10.5 密钥权限管理.....	17
10.6 参考文档.....	17

免责声明

本文档按“现状”提供，福州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。

版权所有 © 2019 福州瑞芯微电子股份有限公司

超越合理使用范畴，非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

福州瑞芯微电子股份有限公司

Fuzhou 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

1 概述

本 SDK 是基于 Buildroot 系统，内核基于 kernel 4.4，适用于 PX30 EVB 以及基于其上所有 Linux 产品开发。

本 SDK 支持 MIPI camera、Music、GPU 等功能。具体功能调试和接口说明，请阅读工程目录 docs/下文档。

2 主要支持功能

功能	模块名
数据通信	Wi-Fi, BT, Camera-MIPI, SDCARD, Ethernet
应用程序	音乐、系统设置，图库，相机，视频

3 SDK 获取说明

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 6 节 SDK 编译说明](#)。

获取 SDK 方法一：从瑞芯微代码服务器获取源码

获取 PX30 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[第 10 节 SSH 公钥操作说明](#)。

PX30 Linux SDK 下载命令如下：

```
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
px30_linux_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、管理项目的软件仓库，其下载地址如下：

```
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。

以 px30_linux_sdk_release_v1.3.0_20200224.tgz 为例，拷贝到该初始化包后，通过如下命令可检出源码：

```
mkdir px30
tar zxvf px30_linux_sdk_release_v1.3.0_20200224.tgz -C px30
cd px30
.repo/repo/repo sync -l
.repo/repo/repo sync
```

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”命令同步更新。

获取 SDK 方法二：从 Github 开源网站获取源码

下载 repo 工具：

```
git clone https://github.com/rockchip-linux/repo.git
```

建立 px30 linux 工作目录：

```
mkdir px30
```

进入 px30 linux 工作目录：

```
cd px30/
```

初始化 repo 仓库:

```
../repo/repo init --repo-url=https://github.com/rockchip-linux/repo -u  
https://github.com/rockchip-linux/manifests -b master -m px30_linux_releas  
e.xml
```

同步下载整个工程:

```
../repo/repo sync
```

4 软件开发指南

PX30 Linux SDK Kernel 版本是 Linux4.4, rootfs 分别是 buildroot(2018.02-rc3), 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_Linux_Software_CN.pdf》。

可在 docs/目录下获取, 并会不断完善更新。

5 PX30 Linux 工程目录介绍

在工程目录下有 buildroot、app、kernel、u-boot、device、docs、external 等目录。每个目录或其子目录会对应一个 git 工程, 提交需要在各自的目录下进行。

- 1) buildroot: 定制根文件系统。
- 2) app: 存放上层应用 app, 主要是一些测试应用程序。
- 3) external: 相关库, 包括音频、视频等。
- 4) kernel: kernel 代码。
- 5) device/rockchip/px30: 存放一些编译和打包固件的脚本和预备文件。
- 6) docs: 存放工程帮助文件。
- 7) prebuilts: 存放交叉编译工具链。
- 8) rkbin: 存放固件和工具。
- 9) rockdev: 存放编译输出固件
- 10) tools: 存放一些常用工具。
- 11) u-boot: uboot 代码。

6 SDK 编译说明

Ubuntu 16.04 系统:

编译 **Buildroot** 环境搭建所依赖的软件包安装命令如下:

```
sudo apt-get install repo git-core gitk git-gui gcc-arm-linux-gnueabi u-boot-tools device-tree-compiler gcc-aarch64-linux-gnu mtools parted libudev-dev libusb-1.0-0-dev python-linaro-image-tools linaro-image-tools autoco
nf autotools-dev libsigsegv2 m4 intltool libdrm-dev curl sed make binutils
build-essential gcc g++ bash patch gzip bzip2 perl tar cpio python unzip r
sync file bc wget libncurses5 libqt4-dev libglib2.0-dev libgtk2.0-dev libglad
e2-dev cvs git mercurial rsync openssh-client subversion asciidoc w3m dbl
atex graphviz python-matplotlib libc6:i386 libssl-dev texinfo genext2fs cor
eutils
```

Ubuntu 17.04/18.04 系统:

除了上面外还需如下依赖包:

```
apt-get install liblz4-tool lib32gcc-7-dev g++-7 libstdc++-7-dev coreutils
```

6.1 U-Boot 编译

进入工程 u-boot 目录下执行编译命令 ./make.sh evb-px30 来获取

```
— u-boot
  |— px30_loader_v1.13.115.bin
  |— trust.img
```

└─ uboot.img

另外，也可以使用工程根目录的 `build.sh` 脚本进行编译（`./build.sh uboot`）。若通过 `make menuconfig` 修改配置，应保存配置，避免被覆盖。

注意：

关闭 bl32，即 `trust` 固件不包含 Secure OS（无 TEE 相关服务）

```
export TRUST_PACK_IGNORE_BL32=--ignore-bl32
```

若需要 Secure OS 功能，则不需要上述环境变量设置，直接执行编译命令即可。

6.2 Kernel 编译步骤

在工程根目录执行以下命令自动完成 `kernel` 的编译及打包：

PX30 EVB 开发板：

```
cd kernel
make ARCH=arm64 px30_linux_defconfig
make ARCH=arm64 px30-evb-ddr3-v10-linux.img -j12
```

另外，也可以使用工程根目录的 `build.sh` 脚本进行编译（`./build.sh kernel`）。

编译后在 `kernel` 目录生成 `boot.img`，这个 `boot.img` 包含内核镜像、DTB 以及 logo 资源。该版本 SDK 支持压缩 `kernel` 启动，对应 `kernel` 固件为 `zboot.img`。

6.3 Recovery 编译步骤

在工程根目录执行以下命令自动完成 `recovery` 的编译及打包：

PX30 EVB 开发板

```
./build.sh recovery
```

编译后在 Buildroot 目录 `/output/rockchip_px30_recovery/images` 生成 `recovery.img`，

6.4 Rootfs 系统编译

在工程根目录执行以下命令自动完成 `rootfs` 的编译及打包：

PX30 EVB 开发板：

```
./build.sh rootfs
```

编译后 `rockdev` 目录下生成 `rootfs.img`。

6.5 重新编译 APP

在工程根目录执行以下命令自动完成指定 APP 的编译，例如 `qcamera` 应用：

```
source envsetup.sh rockchip_px30_64
make qcamera-rebuild
```

6.6 全自动编译

上面 `kernel/u-boot/recovery/rootfs` 各个部分的编译，进入工程根目录执行以下命令自动完成所有的编译：

./build.sh

具体参数使用情况，可 help 查询，比如下：

```
Usage: build.sh [OPTIONS]
Available options:
BoardConfig*.mk    -switch to specified board config
uboot               -build uboot
kernel              -build kernel
modules             -build kernel modules
rootfs              -build default rootfs, currently build buildroot as default
buildroot           -build buildroot rootfs
ramboot             -build ramboot image
multi-npu_boot      -build boot image for multi-npu board
yocto               -build yocto rootfs
debian              -build debian rootfs
pcba                -build pcba
recovery            -build recovery
all                 -build uboot, kernel, rootfs, recovery image
cleanall            -clean uboot, kernel, rootfs, recovery
firmware            -pack all the image we need to boot up system
updateimg           -pack update image
otapackage          -pack ab update otapackage image
save                -save images, patches, commands used to debug
allsave             -build all & firmware & updateimg & save

Default option is 'allsave'.
```

以 PX30 为例：

每个板子的板级配置需要在 `devic/rockchip/px30/BoardConfig.mk` 进行相关配置。

PX30 EVB 主要配置如下：

```

# Target arch
export RK_ARCH=arm64
# Uboot defconfig
export RK_UBOOT_DEFCONFIG=evb-px30
# Trust choose ignore bl32, including --ignore-bl32
export TRUST_PACK_IGNORE_BL32=
# Kernel defconfig
export RK_KERNEL_DEFCONFIG=px30_linux_defconfig
# Kernel dts
export RK_KERNEL_DTS=px30-evb-ddr3-v10-linux
# boot image type
export RK_BOOT_IMG=boot.img
# kernel image path
export RK_KERNEL_IMG=kernel/arch/arm64/boot/Image
export RK_KERNEL_ZIMG=kernel/arch/arm64/boot/Image.lz4
# parameter for GPT table
export RK_PARAMETER=parameter-buildroot.txt
# Buildroot config
export RK_CFG_BUILDR00T=rockchip_px30_64
# Recovery config
export RK_CFG_RECOVERY=rockchip_px30_recovery
# ramboot config
export RK_CFG_RAMBOOT=
# Pcba config
export RK_CFG_PCBA=rockchip_px30_pcba
# Build jobs
export RK_JOBS=12
# target chip
export RK_TARGET_PRODUCT=px30
# Set rootfs type, including ext2 ext4 squashfs
export RK_ROOTFS_TYPE=ext4
# rootfs image path
export RK_ROOTFS_IMG=rockdev/rootfs.${RK_ROOTFS_TYPE}
# Set oem partition type, including ext2 squashfs
export RK_OEM_FS_TYPE=ext2
# Set userdata partition type, including ext2, fat
export RK_USERDATA_FS_TYPE=ext2
# Set flash type. support <emmc, nand, spi_nand, spi_nor>
export RK_STORAGE_TYPE=emmc
#OEM config
export RK_OEM_DIR=oem_normal
#userdata config
export RK_USERDATA_DIR=userdata_normal
#misc image
export RK_MISC=wipe_all-misc.img
#choose enable distro module
export RK_DISTRO_MODULE=

```

6.7 Buildroot 切换 32 位配置

Linux SDK Buildroot 提供 64 位和 32 位的配置，默认 SDK 使用 64 位配置。可以通过以下命令切换到 32 位 Buildroot。

1. 修改为 32 位 Buildroot 配置：

```
./build.sh BoardConfig_32bit.mk
```

2. 再执行全固件编译:

```
./build.sh
```

6.8 Robot 配置及编译

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/px30/BoardConfig_robot64.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、小容量产品 Robot 开发者使用。

PX30 Linux Robot SDK 下载命令如下:

```
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
px30_robot_release.xml
```

如已下载 PX30 Linux SDK，可通过如下命令在 SDK 目录下切换到 Robot SDK:

```
.repo/repo/repo sync -m px30_robot_release.xml
```

此外，Robot SDK 支持 ROS（Robot Operating System），提供 indigo 和 kinetic 两种版本。如需使用 ROS 系统，需要在 buildroot 中使能配置。修改 buildroot 的 defconfig，添加 ros 配置，例如使用 indigo 版本:

```
--- a/configs/rockchip_px30_64_defconfig
+++ b/configs/rockchip_px30_64_defconfig
@@ -1,4 +1,5 @@
 #include "rk3326_arm64.config"
+#include "ros_indigo.config"
```

6.9 固件的打包

执行全固件编译脚本 build.sh 后，脚本将自动完成固件打包，会在 IMAGE 目录下生成以当前时间命名的目录，例如 IMAGE/PX30-EVB-LP3-V10-LINUX_20190917.0608_RELEASE_TEST。

6.10 debian10 编译

在工程根目录执行以下命令自动完成 debian10 的编译.

```
./build.sh distro
```

更多详细说明，可参考文档 docs/Linux/ApplicationNote/Rockchip_Debian10_Developer_Guide_CN.pdf。

注意：目前 Debian10 QT 的编译还依赖 Buildroot qmake 的编译，所以编译 Debian10 前，请先编译 Buildroot。

7 刷机说明



图 7-1 PX30 EVB

7.1 Windows 刷机说明

SDK 提供 Windows 烧写工具(工具版本需要 **V2.55 或以上**)，工具位于工程根目录：
tools/

└─ windows/AndroidTool

如下图，编译生成相应的固件后，设备烧写需要进入 MASKROM 烧写模式，连接好 USB 下载线后，按住按键“MSROM”不放并按下复位键“RST”后松手，就能进入 MASKROM 模式，加载编译生成固件的相应路径后，点击“执行”进行烧写，也可以按“recovery”按键不放并按下复位键“RST”后松手进入 loader 模式进行烧写，下面是 MASKROM 模式的分区偏移及烧写文件。(Note: Window PC 需要在管理员权限运行工具才可执行)

图 7-2 烧写工具 **AndroidTool.exe**

注：烧写前，需安装最新 USB 驱动，驱动详见：
tools/USB 驱动/DriverAssitant_v4.8

7.2 Linux 刷机说明

Linux 下的烧写工具位于 tools/linux 目录下(**Linux_Upgrade_Tool** 工具版本需要 **V1.33 或以上**)，请确认你的板子连接到 maskrom/loader rockusb。比如编译生成的固件在 rockdev 目录下，升级命令如下：

```
./upgrade_tool ul rockdev/MiniLoaderAll.bin
./upgrade_tool di -p rockdev/parameter.txt
./upgrade_tool di -uboot rockdev/uboot.img
./upgrade_tool di -trust rockdev/trust.img
./upgrade_tool di -misc rockdev/misc.img
./upgrade_tool di -boot rockdev/boot.img
./upgrade_tool di -recovery rockdev/recovery.img
./upgrade_tool di -oem rockdev/oem.img
./upgrade_tool di -rootfs rockdev/rootfs.img
./upgrade_tool di -userdata rockdev/userdata.img
./upgrade_tool rd
```

或在根目录，机器在 **maskrom** 状态运行如下升级：

```
./rkflash.sh
```

7.3 系统分区说明

默认分区说明 (下面是 PX30 EVB 分区参考):

Number	Start (sector)	End (sector)	Size	Code	Name
1	16384	24575	4096K	700	uboot
2	24576	32767	4096K	700	trust
3	32768	40959	4096K	700	misc
4	40960	106495	32.0M	700	boot
5	106496	172031	32.0M	700	recovery
6	172032	237567	32.0M	700	backup
7	237568	368639	64.0M	700	oem
8	368640	3514367	1536M	700	rootfs
9	3514368	30535646	12.8G	700	userdata

uboot 分区: 烧写 uboot 编译出来的 uboot.img。

trust 分区: 烧写 uboot 编译出来的 trust.img。

misc 分区: 烧写 misc.img。给 recovery 使用。

boot 分区: 烧写 kernel 编译出来的 boot.img。

recovery 分区: 烧写 recovery.img。

backup 分区: 预留。

oem 分区: 给厂家使用, 存放厂家的 app 或数据。只读, 挂载在/oem 目录。

rootfs 分区: 存放 buildroot 或者 debian 编出来的 rootfs.img,只读。

userdata 分区:存放 app 临时生成的文件或者是给最终用户使用。可读写, 挂载在 /userdata 目录下。

8 调试串口的参数设置

默认 SDK 使用 UART2 作为调试串口, 参数设置为波特率 1500000,8 位数据位, 1 位停止位, 无校验位, 关流控。

9 固件下载

PX30_Linux_SDK_V1.3.0_20200224 固件下载链接:

PX30_Minievb_V11:

Buildroot:

[百度云盘](#)

[Google Drive](#)

Debian10:

[百度云盘](#)

[Google Drive](#)

Robot:

[百度云盘](#)

[Google Drive](#)

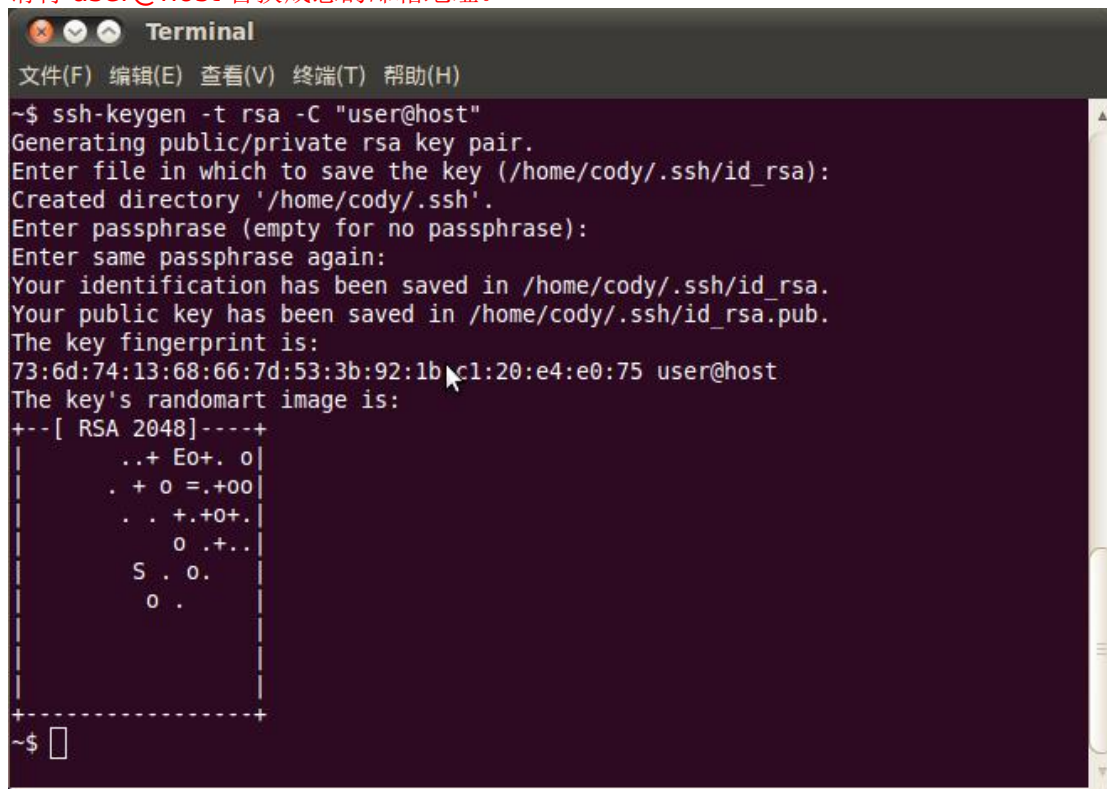
10 SSH 公钥操作说明

10.1 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。



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
```

请妥善保管生成的私钥文件 `id_rsa` 和密码，并将 `id_rsa.pub` 发邮件至 `fae@rock-chips.com`，抄送对应业务申请开通 SDK 下载权限。

10.2 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。

10.3 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
```

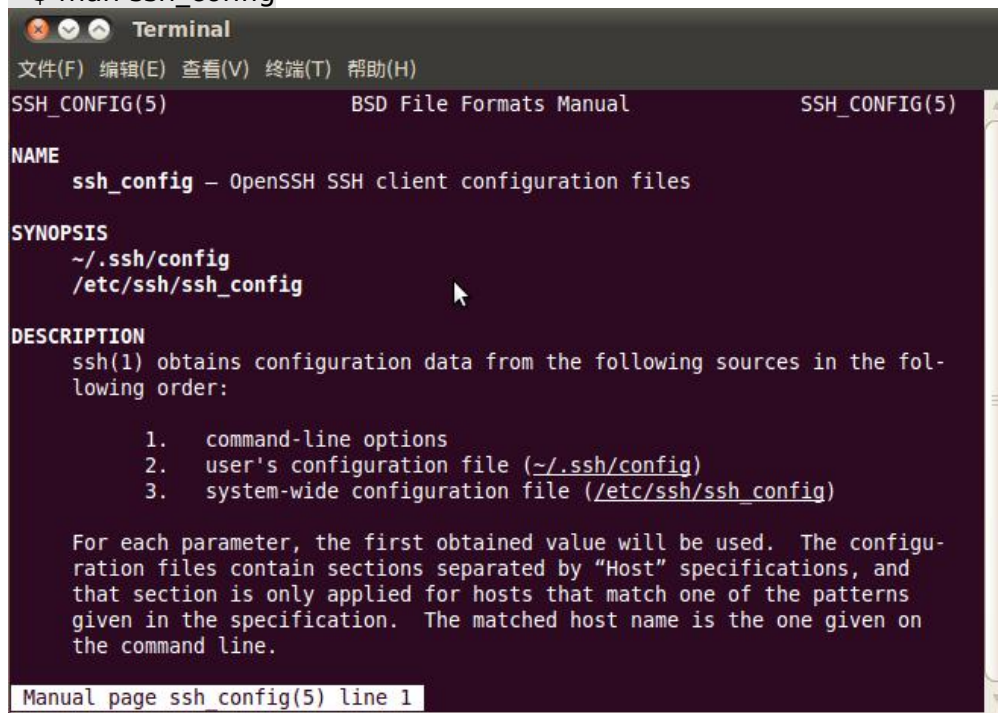
在 `console` 输入如下命令即可解决。

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


10.4 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`

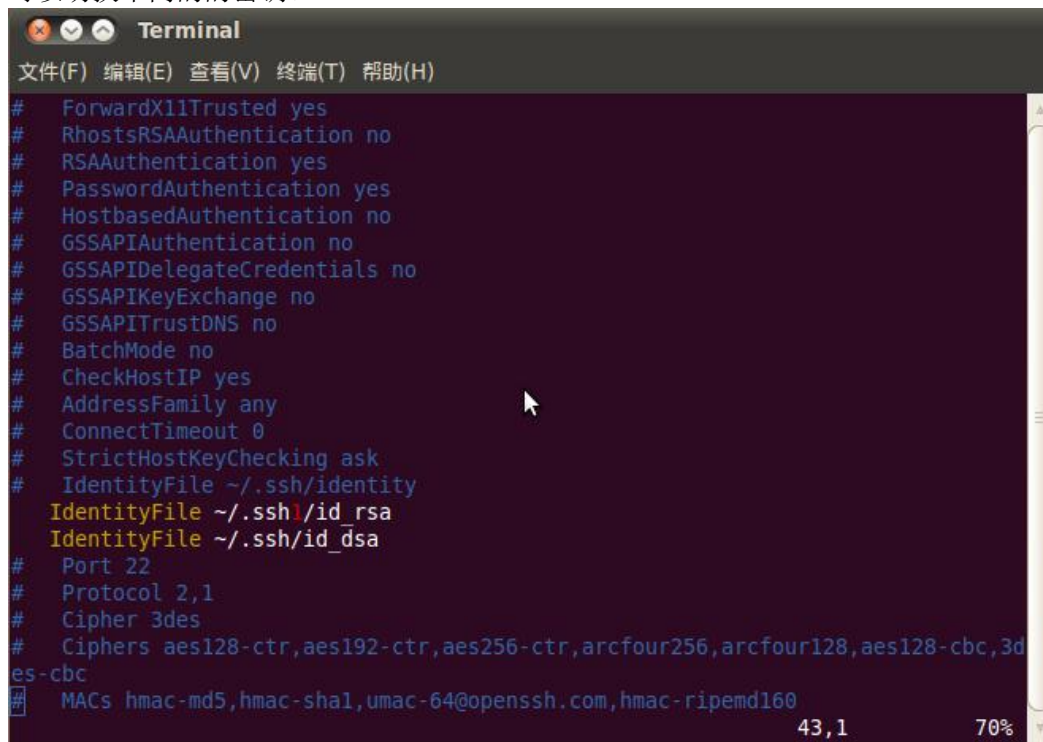


通过如下命令，配置当前用户的 SSH 配置。

~\$ `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。



10.5 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。

10.6 参考文档

更多详细说明，可参考文档 `sdk/docs/RKTools manuals/Rockchip SDK Kit 申请指南 V1.6-201905.pdf`。