

# Rockchip RK3288 Linux SDK 发布说明

---

文档标识: RK-FB-CS-006

发布版本: V2.2.0

日期: 2020-07-08

文件密级: ☐绝密 ☐秘密 ☐内部资料 ☒公开

## 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, 由其各自所有者所有。

版权所有 © 2020 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。

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](http://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](mailto:fae@rock-chips.com)

## 前言

## 概述

文档主要介绍 Rockchip RK3288 Linux SDK发布说明，旨在帮助工程师更快上手RK3288 Linux SDK开发及相关调试方法。

## 读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

## 各芯片系统支持状态

芯片名称	Buildroot	Debian 9	Debian 10	Yocto
RK3288	Y	Y	Y	N

## 修订记录

日期	版本	作者	修改说明
2018-04-16	V1.0.0	Nickey Yang	初始版本
2018-04-23	V1.1.0	Nickey Yang	重命名和修订格式
2018-04-26	V1.3.0	Nickey Yang	增加buildroot编译说明
2018-06-20	V2.0.0	Nickey Yang	更新 SDK buildroot到2018.02版本。
2019-09-26	V2.1.0	Nickey Yang	更新 SDK 编译和烧写方法具。
2020-07-08	V2.2.0	Caesar Wang	增加 Debian10支持，Markdown格式重写发布文档

# 目录

## Rockchip RK3288 Linux SDK 发布说明

1. 概述
2. 主要支持功能
3. SDK 获取说明
4. 软件开发指南
  - 4.1 开发指南
  - 4.2 软件更新记录
5. 硬件开发指南
6. SDK 工程目录介绍
7. SDK 编译说明
  - 7.1 U-Boot 编译
  - 7.2 Kernel 编译步骤
  - 7.3 Recovery 编译步骤
  - 7.4 Buildroot 编译
    - 7.4.1 Buildroot 的 Rootfs 编译
    - 7.4.2 Buildroot 中模块编译
  - 7.5 Debian 9 编译
  - 7.6 Debian 10 编译
  - 7.7 全自动编译
  - 7.8 固件的打包
8. 刷机说明
  - 8.1 Windows 刷机说明
  - 8.2 Linux 刷机说明
  - 8.3 系统分区说明
9. RK3288 SDK 固件
10. SSH 公钥操作说明
  - 10.1 多台机器使用相同 SSH 公钥
  - 10.2 一台机器切换不同 SSH 公钥
  - 10.3 密钥权限管理
  - 10.4 参考文档

## 1. 概述

本 SDK 支持三个系统分别基于 Buildroot 2018.02-rc3, Yocto Thud 2.6, Debian 9 和 Debian 10 上开发, 内核基于 Kernel 4.4, 引导基于 U-boot v2017.09, 适用于 RK3288 EVB 开发板及基于此开发板进行二次开发的所有 Linux 产品。

本 SDK 支持 VPU 硬解码、GPU 3D、Wayland/X11 显示、Qt 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、以太网卡、USB、SD 卡
应用程序	多媒体播放、设置、浏览器、文件管理

## 3. SDK 获取说明

SDK 通过瑞芯微代码服务器对外发布或者从 [Github](#) 开源网站上获取。其编译开发环境, 参考第 7 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 RK3288 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](#)。

RK3288\_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
rk3288_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, 进行同步后与通过 repo 下载的源码是一致的。

以 rk3288\_linux\_sdk\_release\_v2.2.0\_20200708.tgz 为例, 拷贝到该初始化包后, 通过如下命令可检出源码:

```
mkdir rk3288
tar xvf rk3288_linux_sdk_release_v2.2.0_20200708.tgz -C rk3288
cd rk3288
.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 rk3288 linux 工作目录

```
mkdir rk3288_linux
```

进入 rk3288 linux 工作目录

```
cd rk3288_linux/
```

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
rk3288_linux_release.xml
```

同步下载整个工程:

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

注意: 如果是已立项的项目请优先选择用方法一获取代码，不同于 Github 的是它会经过内部稳定测试和版本控制，方法二更多适用于爱好者和前期项目评估。

## 4. 软件开发指南

---

### 4.1 开发指南

RK3288 Linux SDK Kernel 版本是 Kernel 4.4，Rootfs 分别是 Buidlroot(2018.02-rc3) 和 Debian9/10，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

《RK3288\_Linux\_SDK\_Release\_xxx.pdf》。可在 docs/RK3288 目录下获取，并会不断完善更新。

### 4.2 软件更新记录

软件发布版本升级通过工程 xml 进行查看当前版本，具体方法如下:

```
.repo/manifests$ ls -l -h rk3288_linux_release.xml
```

软件发布版本升级更新内容通过工程文本可以查看，具体方法如下：

```
.repo/manifests$ cat rk3288_linux_v2.0/RK3288_Linux_Release_Note.txt
```

或者参考工程目录：

```
<SDK>/docs/Socs/RK3288/RK3288_Linux_SDK_Release_Note.txt
```

## 5. 硬件开发指南

---

硬件相关开发可以参考用户使用指南，在工程目录：

RK3288 EVB 硬件开发指南：

```
<SDK>/docs/RK3288/Rockchip_RK3288_User_Manual_EVB_V1.0_CN.pdf
```

## 6. SDK 工程目录介绍

---

SDK目录包含有 buildroot、debian、recovery、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。

- app: 存放上层应用 APP，主要是 qcamera/qfm/qplayer/qsetting 等一些应用程序。
- buildroot: 基于 Buildroot（2018.02-rc3）开发的根文件系统。
- debian: 基于 Debian 9 开发的根文件系统。
- device/rockchip: 存放各芯片板级配置以及一些编译和打包固件的脚本和预备文件。
- docs: 存放开发指导文件、平台支持列表、工具使用文档、Linux 开发指南等。
- distro: 基于 Debian 10 开发的根文件系统。
- IMAGE: 存放每次生成编译时间、XML、补丁和固件目录。
- external: 存放第三方相关仓库，包括音频、视频、网络、recovery 等。
- kernel: 存放 Kernel 4.4 开发的代码。
- prebuilts: 存放交叉编译工具链。
- rkbin: 存放 Rockchip 相关 Binary 和工具。
- rockdev: 存放编译输出固件。
- tools: 存放 Linux 和 Window 操作系统下常用工具。
- u-boot: 存放基于 v2017.09 版本进行开发的 U-Boot 代码。

## 7. SDK 编译说明

---

本 SDK 开发环境是在 Ubuntu 系统上开发测试。我们推荐使用 Ubuntu 18.04 的系统进行编译。其他的 Linux 版本可能需要对软件包做相应调整。除了系统要求外，还有其他软硬件方面的要求。

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。

软件要求：Ubuntu 18.04 系统：

编译 SDK 环境搭建所依赖的软件包安装命令如下：

```
repo git ssh make gcc libssl-dev liblz4-tool expect g++ patchelf chrpath gawk
texinfo chrpath diffstat binfmt-support qemu-user-static live-build
```

建议使用 Ubuntu18.04 系统或更高版本开发，若编译遇到报错，可以视报错信息，安装对应的软件包。

## 7.1 U-Boot 编译

进入工程 u-boot 目录下执行 make.sh 来获取 rk3288\_loader\_v1.08.258.bin trust.img uboot.img.

RK3288 EVB 开发板：

```
./make.sh rk3288
```

RK3288 Firefly 开发板：

```
./make.sh firefly-rk3288
```

编译后生成文件在 u-boot 目录下：

```
u-boot/
├─ rk3288_loader_v1.08.258.bin
├─ trust.img
└─ uboot.img
```

## 7.2 Kernel 编译步骤

进入工程目录根目录执行以下命令自动完成 kernel 的编译及打包。

RK3288 EVB RK808 开发板：

```
cd kernel
make ARCH=arm rockchip_linux_defconfig
make ARCH=arm rk3288-evb-rk808-linux.img -j12
```

RK3288 EVB ACT8846 开发板：

```
cd kernel
make ARCH=arm rockchip_linux_defconfig
make ARCH=arm rk3288-evb-act8846.img -j12
```

RK3288 Firefly 开发板：

```
cd kernel
make ARCH=arm rockchip_linux_defconfig
make ARCH=arm rk3288-firefly.img -j12
```

编译后在 kernel 目录生成 zboot.img，此 zboot.img 就是包含 Kernel 的 zimage 和 DTB。

## 7.3 Recovery 编译步骤

进入工程目录根目录执行以下命令自动完成 Recovery 的编译及打包：

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

编译后在 Buildroot 目录 output/rockchip\_rk3288\_recovery/images 生成 recovery.img。

需要特别注意 recovery.img 是包含 kernel.img，所以每次 Kernel 更改，Recovery 是需要重新打包生成。例如如下：

```
SDK$source envsetup.sh rockchip_rk3288
SDK$make recovery-rebuild
SDK$./build.sh recovery
```

## 7.4 Buildroot 编译

### 7.4.1 Buildroot 的 Rootfs 编译

进入工程目录根目录执行以下命令自动完成 Rootfs 的编译及打包：

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

编译后在 Buildroot 目录 output/rockchip\_rk3288/images 下生成 rootfs.ext4。

备注：

若需要编译单个模块或者第三方应用，需对交叉编译环境进行配置。交叉编译工具位于 buildroot/output/rockchip\_rk3288/host/usr 目录下，需要将工具的 bin/ 目录和 arm-buildroot-linux-gnu/bin/ 目录设为环境变量，在顶层目录执行自动配置环境变量的脚本（只对当前控制台有效）：

输入命令查看：

```
cd buildroot/output/rockchip_rk3288/host/usr
arm-linux-gcc --version
```

此时会打印如下信息：

```
gcc version 8.4.0 (Buildroot 2018.02-rc3-02301-ga3d3e23e2c)
```

### 7.4.2 Buildroot 中模块编译

比如 qplayer 模块，常用相关编译命令如下：



- 编译 qplayer

```
SDK$make qplayer
```

- 重编 qplayer

```
SDK$make qplayer-rebuild
```

- 删除 qplayer

```
SDK$make qplayer-dirclean  
或者  
SDK$rm -rf /buildroot/output/rockchip_rk3288/build/qplayer-1.0
```

## 7.5 Debian 9 编译

```
./build.sh debian
```

或进入 debian/ 目录:

```
cd debian/
```

后续的编译和 Debian 固件生成请参考当前目录 [readme.md](#)。

### (1) Building base Debian system

```
sudo apt-get install binfmt-support qemu-user-static live-build  
sudo dpkg -i ubuntu-build-service/packages/*  
sudo apt-get install -f
```

编译 32 位的 Debian:

```
RELEASE=stretch TARGET=desktop ARCH=armhf ./mk-base-debian.sh
```

编译完成会在 `debian/` 目录下生成: `linaro-stretch-alip-xxxxx-1.tar.gz` (xxxxx 表示生成时间戳)。

FAQ:

- 上述编译如果遇到如下问题情况:

```
noexec or nodev issue /usr/share/debootstrap/functions: line 1450:  
.../rootfs/ubuntu-build-service/stretch-desktop-armhf/chroot/test-dev-null:  
Permission denied E: Cannot install into target '/rootfs/ubuntu-build-  
service/stretch-desktop-armhf/chroot' mounted with noexec or nodev
```

解决方法:

```
mount -o remount,exec,dev xxx (xxx 是工程目录), 然后重新编译
```

另外如果还有遇到其他编译异常, 先排除使用的编译系统是 `ext2/ext4` 的系统类型。

- 由于编译 Base Debian 需要访问国外网站，而国内网络访问国外网站时，经常出现下载失败的情况：

Debian 9 使用 live build, 镜像源改为国内可以这样配置：

```
+++ b/ubuntu-build-service/stretch-desktop-armhf/configure
@@ -11,6 +11,11 @@ set -e
 echo "I: create configuration"
 export LB_BOOTSTRAP_INCLUDE="apt-transport-https gnupg"
 lb config \
+ --mirror-bootstrap "http://mirrors.163.com/debian" \
+ --mirror-chroot "http://mirrors.163.com/debian" \
+ --mirror-chroot-security "http://mirrors.163.com/debian-security" \
+ --mirror-binary "http://mirrors.163.com/debian" \
+ --mirror-binary-security "http://mirrors.163.com/debian-security" \
  --apt-indices false \
  --apt-recommends false \
  --apt-secure false \
```

如果其他网络原因不能下载包，有预编生成的包分享在[百度云网盘](#)，放在当前目录直接执行下一步操作。

## (2) Building rk-debian rootfs

编译 32位的 Debian：

```
VERSION=debug ARCH=armhf ./mk-rootfs-stretch.sh
```

## (3) Creating the ext4 image(linaro-rootfs.img)

```
./mk-image.sh
```

此时会生成 linaro-rootfs.img。

## 7.6 Debian 10 编译

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

或进入 distro/ 目录：

```
cd distro/ && make ARCH=arm rk3288_defconfig && ./make.sh
```

编译后在 distro/output/images/ 目录下生成 rootfs.ext4。

注意：目前Debian 10 QT的编译还依赖 Buildroot qmake的编译，所以编译 Debian 10 前，请先编译 Buildroot。

更多 Debian 10 的介绍参考文档：

```
<SDK>/docs/Linux/ApplicationNote/Rockchip_Developer_Guide_Debian10_CN.pdf
```

## 7.7 全自动编译

完成上述 Kernel/U-Boot/Recovery/Rootfs 各个部分的编译后，进入工程目录根目录执行以下命令自动完成所有的编译：

```
$. /build.sh all
```

默认是 Buildroot，可以通过设置环境变量 RK\_ROOTFS\_SYSTEM 指定 rootfs。  
比如需要 buildroot 可以通过以下命令进行生成：

```
$export RK_ROOTFS_SYSTEM=buildroot
$. /build.sh all
```

具体参数使用情况，可 help 查询，比如：

```
rk3288$ ./build.sh --help
Usage: build.sh [OPTIONS]
Available options:
BoardConfig*.mk  -switch to specified board config
uboot             -build uboot
spl              -build spl
kernel           -build kernel
modules          -build kernel modules
toolchain        -build toolchain
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 debian9 stretch rootfs
distro           -build debian10 buster 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'.
```

每个板子的板级配置需要在 /device/rockchip/rk3288/Boardconfig.mk 进行相关配置。  
RK3288 EVB 开发板主要配置如下：

```
# Target arch
export RK_ARCH=arm
# Uboot defconfig
export RK_UBOOT_DEFCONFIG=rk3288
# Kernel defconfig
export RK_KERNEL_DEFCONFIG=rockchip_linux_defconfig
# Kernel dts
export RK_KERNEL_DTS=rk3288-evb-rk808-linux
# boot image type
export RK_BOOT_IMG=zboot.img
# kernel image path
```

```
export RK_KERNEL_IMG=kernel/arch/arm/boot/zImage
# parameter for GPT table
export RK_PARAMETER=parameter.txt
# Buildroot config
export RK_CFG_BUILDROOT=rockchip_rk3288
# Debian 10 config
export RK_DISTRO_DEFCONFIG=rk3288_defconfig
# Recovery config
export RK_CFG_RECOVERY=rockchip_rk3288_recovery
```

## 7.8 固件的打包

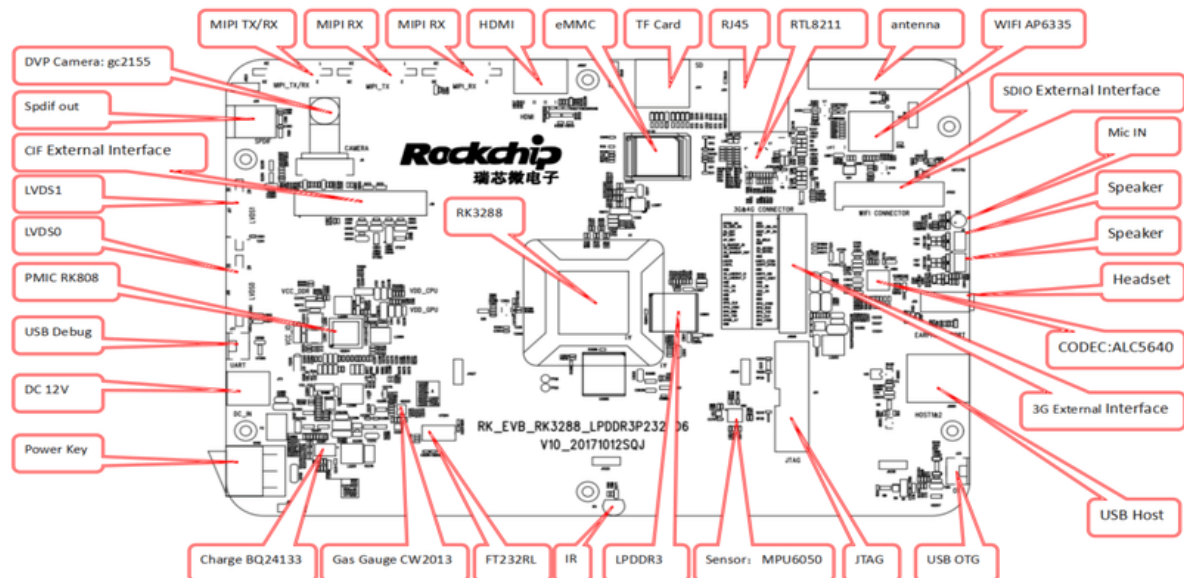
上面 Kernel/U-Boot/Recovery/Rootfs 各个部分的编译后，进入工程目录根目录执行以下命令自动完成所有固件打包到 rockdev 目录下：

固件生成：

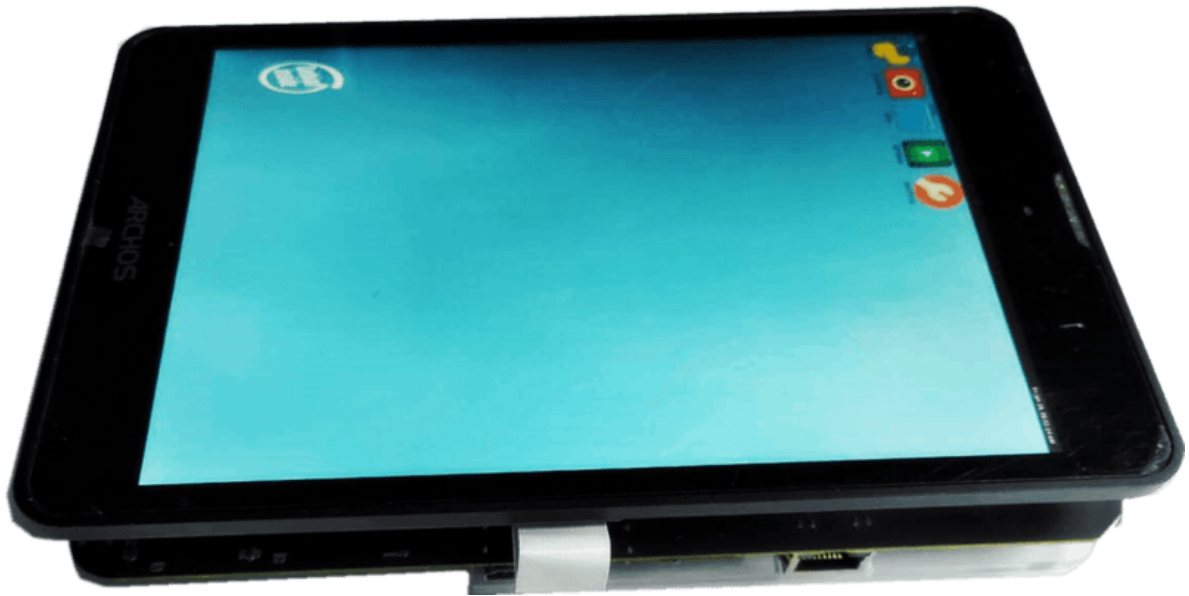
```
./mkfirmware.sh
```

## 8. 刷机说明

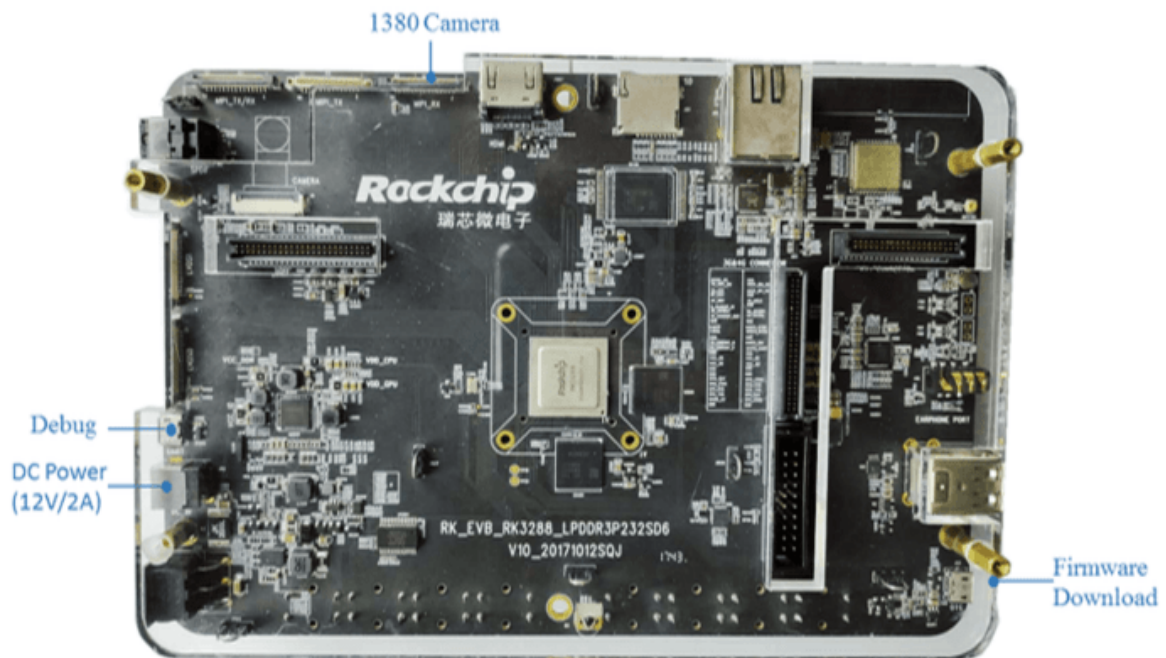
- RK3288 EVB PCB 接口分布图如下：



- RK3288 EVB 正面图：



- RK3288 EVB 背面图:



## 8.1 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/RKDevTool
```

如下图, 编译生成相应的固件后, 设备烧写需要进入 MASKROM 或 BootROM 烧写模式, 连接好 USB 下载线后, 按住按键“MASKROM”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。(注意: Windows PC 需要在管理员权限运行工具才可执行)



注：烧写前，需安装最新 USB 驱动，驱动详见：

```
<SDK>/tools/windows/DriverAssitant_v4.8.zip
```

## 8.2 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：

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

或升级打包后的完整固件：

```
sudo ./upgrade_tool uf rockdev/update.img
```

或在根目录，机器在 MASKROM 状态运行如下升级：

```
./rkflash.sh
```

## 8.3 系统分区说明

默认分区说明 (下面是 RK3288 EVB 分区参考)

Number	Start (sector)	End (sector)	Size	Name
1	16384	24575	4096K	uboot
2	24576	32767	4096K	trust
3	32768	40959	4096K	misc
4	40960	106495	32M	boot
5	106496	303104	32M	recovery
6	172032	237567	32M	bakcup
7	237568	368639	64M	oem
8	368640	12951551	6G	rootfs
9	12951552	15269854	1.1G	userdata

- uboot 分区: 供 uboot 编译出来的 uboot.img。
- trust 分区: 供 uboot 编译出来的 trust.img。
- misc 分区: 供 misc.img, 给 recovery 使用。
- boot 分区: 供 kernel 编译出来的 boot.img。
- recovery 分区: 供 recovery 编译出的 recovery.img。
- backup 分区: 预留, 暂时没有用, 后续跟 Android 一样作为 recovery 的 backup 使用。
- oem 分区: 给厂家使用, 存放厂家的 APP 或数据。挂载在 /oem 目录。
- rootfs 分区: 供 buildroot、debian 编出来的 rootfs.img。
- userdata 分区: 供 APP 临时生成文件或给最终用户使用, 挂载在 /userdata 目录下。

## 9. RK3288 SDK 固件

RK3288\_Linux\_SDK\_V2.2.0\_20200708 固件下载链接如下

(包含 Buildroot/Debian 9/Debian 10 的固件)

- RK3288 EVB 开发板

[Buildroot](#)

[Debian9](#)

[Debian10](#)

## 10. 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](mailto:fae@rock-chips.com), 申请开通 SDK 代码下载权限。

该文档会在申请开通权限流程中, 释放给客户使用。



## 10.1 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
```

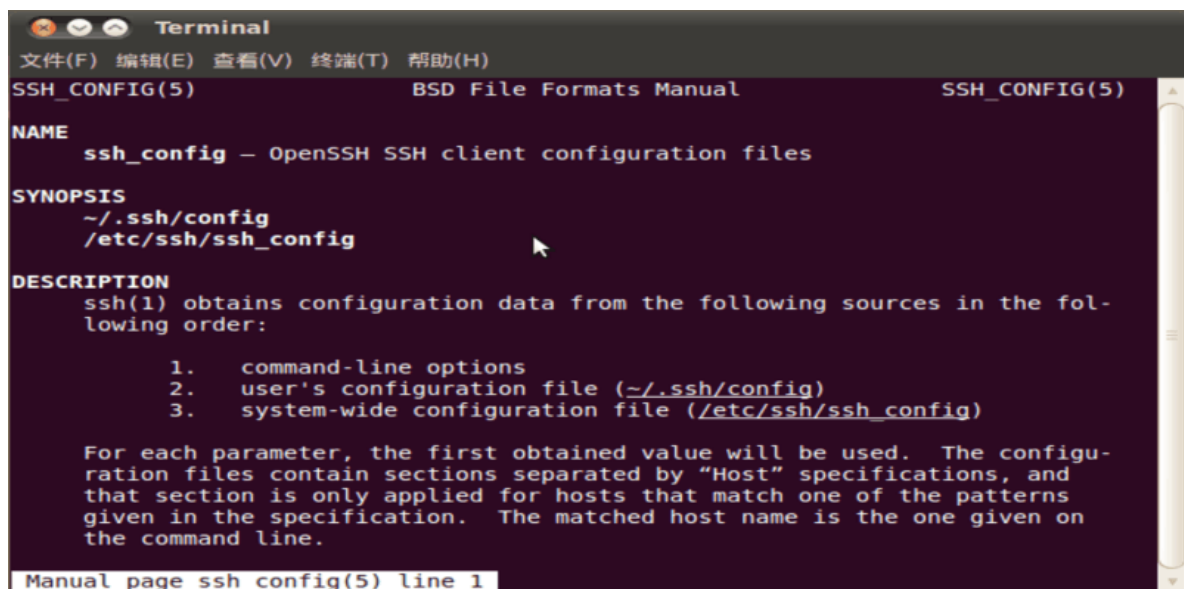
在 console 输入如下命令即可解决。

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

## 10.2 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
```

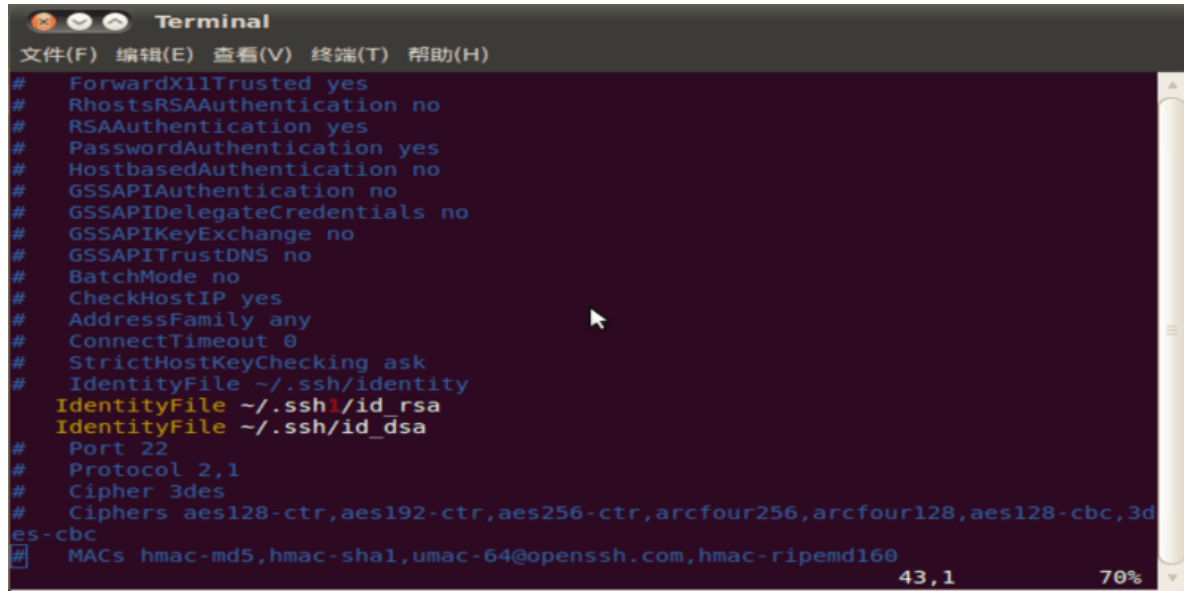




通过如下命令，配置当前用户的 SSH 配置。

```
~$ 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，可以切换不同的密钥。

A terminal window titled "Terminal" with a menu bar (File(F), Edit(E), View(V), Terminal(T), Help(H)). The terminal displays the contents of the ~/.ssh/config file. The configuration includes various SSH options like ForwardX11Trusted, RhostsRSAAuthentication, RSAAuthentication, PasswordAuthentication, HostbasedAuthentication, GSSAPIAuthentication, GSSAPIDelegatedCredentials, GSSAPIKeyExchange, GSSAPITrustDNS, BatchMode, CheckHostIP, AddressFamily, ConnectTimeout, StrictHostKeyChecking, IdentityFile, Port, Protocol, Cipher, Ciphers, and MACs. The IdentityFile is set to ~/.ssh1/id\_rsa. The terminal shows the file is 43,1 lines and 70% zoomed.

```
# ForwardX11Trusted yes
# RhostsRSAAuthentication no
# RSAAuthentication yes
# PasswordAuthentication yes
# HostbasedAuthentication no
# GSSAPIAuthentication no
# GSSAPIDelegatedCredentials 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
```

## 10.3 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。

## 10.4 参考文档

更多详细说明，可参考文

档/docs/Others/Rockchip\_User\_Guide\_SDK\_Application\_And\_Synchronization\_CN.pdf。