Rockchip RK3308 Linux5.10 SDK 快速入门

文档标识：RK-JC-YF-963

发布版本：V1.3.1

日期：2023-12-20

文件密级：□绝密 □秘密 □内部资料 ■公开

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

前言

概述

本文主要描述了RK3308 Linux5.10 SDK的基本使用方法，旨在帮助开发者快速了解并使用RK3308 Linux5.10 SDK开发包。

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	**Uboot版本**	**Kernel版本**	**Buildroot版本**
RK3308B/RK3308H/RK3308B-S/RK3308H-S	2017.9	5.10	2021.11

修订记录

日期	版本	作者	修改说明
2022-09-20	V1.0.0	LinJianHua	初始版本
2022-11-20	V1.0.1	Caesar Wang	更新刷机说明
2023-04-20	V1.1.0	LinJianHua	更新SDK到V1.1.0
2023-06-20	V1.2.0	LinJianHua	更新SDK到V1.2.0
2023-09-20	V1.3.0	LinJianHua	更新SDK到V1.3.0
2023-12-20	V1.3.1	zack.huang	增加智能家居产品的编译说明

目录

1. 开发环境搭建

1.1 准备开发环境

我们推荐使用 Ubuntu 22.04 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 22.04 系统或更高版本系统。

1.2 安装库和工具集

使用命令行进行设备开发时，可以通过以下步骤安装编译SDK需要的库和工具。

使用如下apt-get命令安装后续操作所需的库和工具：

```
sudo apt-get update && sudo apt-get install git ssh make gcc libssl-dev \
liblz4-tool expect expect-dev g++ patchelf chrpath gawk texinfo chrpath \
diffstat binfmt-support qemu-user-static live-build bison flex fakeroot \
cmake gcc-multilib g++-multilib unzip device-tree-compiler ncurses-dev \
libgucharmap-2-90-dev bzip2 expat gpgv2 cpp-aarch64-linux-gnu
```

> 说明： 安装命令适用于Ubuntu22.04，其他版本请根据安装包名称采用对应的安装命令，若编译遇到报错，可以视报错信息，安装对应的软件包。其中：
>
> - 如果PC在编译Buildroot时无法访问Google网站，需设置DNS来支持使用国内镜像kgithub.com下载dl包。
> - python要求安装python 3.6及以上版本，此处以python 3.6为例。
> - make要求安装 make 4.0及以上版本，此处以 make 4.2为例。
> - lz4要求安装 lz4 1.7.3及以上版本。

1.2.1 设置DNS支持kgithub.com

```
sudo sed -i '$a 43.154.68.204\tkgithub.com' /etc/hosts
sudo sed -i '$a 43.155.83.75\traw.kgithub.com
objects.githubusercontent.kgithub.com' /etc/hosts
```

1.2.2 检查和升级主机的`python`版本

检查和升级主机的`python`版本方法如下：

- 检查主机`python`版本

```
$ python3 --version
Python 3.10.6
```

如果不满足python>=3.6版本的要求， 可通过如下方式升级：

- 升级`python 3.6.15`新版本

```
PYTHON3_VER=3.6.15
echo "wget
https://www.python.org/ftp/python/${PYTHON3_VER}/Python-${PYTHON3_VER}.tgz"
echo "tar xf Python-${PYTHON3_VER}.tgz"
echo "cd Python-${PYTHON3_VER}"
echo "sudo apt-get install libsqlite3-dev"
echo "./configure --enable-optimizations"
echo "sudo make install -j8"
```

1.2.3 检查和升级主机的`make`版本

检查和升级主机的`make`版本方法如下：

- 检查主机`make`版本

```
$ make -v
GNU Make 4.2
Built for x86_64-pc-linux-gnu
```

- 升级`make 4.2`新版本

```
$ sudo apt update && sudo apt install -y autoconf autopoint
git clone https://gitee.com/mirrors/make.git
cd make
git checkout 4.2
git am $BUILDROOT_DIR/package/make/*.patch
autoreconf -f -i
./configure
make make -j8
sudo install -m 0755 make /usr/bin/make
```

1.2.4 检查和升级主机的`lz4`版本

检查和升级主机的`lz4`版本方法如下：

- 检查主机`lz4`版本

```
$ lz4 -v
*** LZ4 command line interface 64-bits v1.9.3, by Yann Collet ***
```

- 升级`lz4`新版本

```
git clone https://gitee.com/mirrors/LZ4_old1.git
cd LZ4_old1
make
sudo make install
sudo install -m 0755 lz4 /usr/bin/lz4
```

1.2.5 检查和升级主机的`git`版本

- 检查主机`git`版本

```
$ git -v
git version 2.38.0
```

- 升级`git`新版本

```
$ sudo apt update && sudo apt install -y libcurl4-gnutls-dev
git clone https://gitee.com/mirrors/git.git --depth 1 -b v2.38.0
cd git
make git -j8
make install
sudo install -m 0755 git /usr/bin/git
```

2. 软件开发指南

2.1 开发向导

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下获取，并会不断完善更新。

2.2 软件更新记录

软件发布版本升级通过工程 xml 进行查看，具体方法如下：

```
.repo/manifests$ realpath rk3308_linux5.10_release.xml
# 例如:打印的版本号为v1.3.0，更新时间为20230920
#<SDK>/.repo/manifests/rk3308_linux/rk3308_linux5.10_release_v1.3.0_20230920.xml
```

3. SDK 配置框架说明

3.1 SDK 工程目录介绍

SDK目录包含有 buildroot、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/qseting 等一些应用程序。
- buildroot：基于 Buildroot（2021.11）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。
- external：存放第三方相关仓库，包括音频、视频、网络、recovery 等。

- kernel：存放 Kernel 5.10 开发的代码。
- output：存放每次生成的固件信息、编译信息、XML、主机环境等。
- 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,实际软链接到`output/firmware`。
- 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。

4. SDK编译说明

SDK可通过`make`或`./build.sh`加目标参数进行相关功能的配置和编译。 具体参考`device/rockchip/common/README.md`编译说明。

4.1 SDK编译命令查看

`make help`, 例如：

```
$ make help
  menuconfig              - interactive curses-based configurator
  oldconfig               - resolve any unresolved symbols in .config
  syncconfig              - Same as oldconfig, but quietly, additionally update
deps
  olddefconfig            - Same as syncconfig but sets new symbols to their
default value
  savedefconfig           - Save current config to RK_DEFCONFIG (minimal config)
  ...
```

make实际运行是`./build.sh`

即也可运行`./build.sh <target>`来编译相关功能，具体可通过`./build.sh help`查看具体编译命令。

```
$ ./build.sh -h

############### Rockchip Linux SDK ###############

Manifest: rockchip_linux5.10.xml

Usage: build.sh [OPTIONS]
Available options:
chip[:<chip>[:<config>]]                choose chip
defconfig[:<config>]                    choose defconfig
 *_defconfig                            switch to specified defconfig
    available defconfigs:
        rockchip_32bit_defconfig
        rockchip_defconfig
        rockchip_rk3308_rtos_amp_32bit_defconfig
        rockchip_rk3308_rtos_linux_64bit_defconfig
        rockchip_rk3308_rtos_smp_32bit_defconfig
        rockchip_rk3308b_32bit_defconfig
```

```
        rockchip_rk3308b_64bit_defconfig
        rockchip_rk3308bs_32bit_defconfig
        rockchip_rk3308bs_32bit_display_defconfig
        rockchip_rk3308bs_64bit_defconfig
        rockchip_rk3308bs_evb_v20_ia_32bit_defconfig
        rockchip_rk3308h_32bit_defconfig
        rockchip_rk3308hs_32bit_defconfig
 olddefconfig                           resolve any unresolved symbols in .config
 savedefconfig                          save current config to defconfig
 menuconfig                             interactive curses-based configurator
config                                  modify SDK defconfig
shell                                   setup a shell for developing
print-parts                             print partitions
mod-parts                               interactive partition table modify
edit-parts                              edit raw partitions
new-parts:<offset>:<name>:<size>...     re-create partitions
insert-part:<idx>:<name>[:<size>]       insert partition
del-part:(<idx>|<name>)                 delete partition
move-part:(<idx>|<name>):<idx>          move partition
rename-part:(<idx>|<name>):<name>       rename partition
resize-part:(<idx>|<name>):<size>       resize partition
kernel-4.19[:cmds]                      build kernel 4.19
kernel-4.4[:cmds]                       build kernel 4.4
kernel-5.10[:cmds]                      build kernel 5.10
kernel-5.10-rt[:cmds]                   build kernel 5.10-rt
kernel-6.1[:cmds]                       build kernel 6.1
kernel-rt[:cmds]                build kernel rt
kernel[:cmds]                           build kernel
modules[:cmds]                          build kernel modules
linux-headers[:cmds]                    build linux-headers
kernel-config[:cmds]                    modify kernel defconfig
kernel-make[:<arg1>:<arg2>]             run kernel make (alias kmake)
wifibt[:<dst dir>[:<chip>]]             build Wifi/BT
rtos                                    build and pack RTOS
buildroot-config[:<config>]             modify buildroot defconfig
buildroot-make[:<arg1>:<arg2>]          run buildroot make (alias bmake)
rootfs[:<rootfs type>]                  build default rootfs
buildroot                               build buildroot rootfs
yocto                                   build yocto rootfs
debian                                  build debian rootfs
recovery                                build recovery
pcba                                    build PCBA
security_check                          check contidions for security boot
createkeys                              build security boot keys
security_ramboot                        build security ramboot
security_uboot                          build uboot with security
security_boot                           build boot with security
security_recovery                       build recovery with security
security_rootfs                         build rootfs with security
loader[:cmds]                           build loader (uboot)
uboot[:cmds]                            build u-boot
uefi[:cmds]                             build uefi
firmware                                pack and check firmwares
edit-package-file                       edit package-file
edit-ota-package-file                   edit A/B OTA package-file
updateimg                               build update image
otapackage                              build A/B OTA update image
all                                     build all images
```

```
save                                    save images and build info
allsave                                 build all images and save them
cleanall                                cleanup
clean[:module[:module]]...              cleanup modules
    available modules:
        all
        config
        firmware
        kernel
        loader
        pcba
        recovery
        rootfs
        updateimg
post-rootfs <rootfs dir>                trigger post-rootfs hook scripts
help                                    usage

Default option is 'allsave'.
```

4.2 SDK板级配置

进入工程`<SDK>/device/rockchip/rk3308`目录：

板级配置	说明
rockchip_defconfig	适用于 RK3308BS EVB V20 开发板带显示 运行64位系统
rockchip_32bit_defconfig	适用于 RK3308BS EVB V20 开发板带显示 运行32位系统
rockchip_rk3308bs_64bit_defconfig	适用于 RK3308BS EVB V11\V20 开发板 运行64位系统
rockchip_rk3308bs_32bit_defconfig	适用于 RK3308BS EVB V11\V20 开发板 运行32位系统
rockchip_rk3308hs_32bit_defconfig	适用于 RK3308HS MODULE V10 开发板 运行32位系统
rockchip_rk3308b_64bit_defconfig	适用于 RK3308B EVB V10 开发板 运行64位系统
rockchip_rk3308b_32bit_defconfig	适用于 RK3308B EVB V10 开发板 运行32位系统
rockchip_rk3308h_32bit_defconfig	适用于 RK3308H MODULE V10 开发板 运行32位系统
rockchip_rk3308bs_evb_v20_ia_32bit_defconfig	适用于RK3308BS EVB v11/v20开发板 运行32位系统 家居显控demo
rockchip_rk3308bs_32bit_display_defconfig	适用于RK3308BS EVB v11/v20开发板 运行32位系统 运行lvgl官方demo
rockchip-rk3308-evb-audio-v10-64bit-defconfig	适用于 RK3308 Audio EVB V10 开发板 运行64位系统

方法1 `./build.sh` 后面加上板级配置文件, 例如：

选择适用于**RK3308BS EVB V11\V20 开发板** 运行64位系统的板级配置：

```
rk3308$./build.sh device/rockchip/rk3308/rockchip_rk3308bs_64bit_defconfig
```

方法2

```
~/3308/5.10_sdk$ ./build.sh lunch

############### Rockchip Linux SDK ###############

Manifest: rockchip_linux5.10.xml

Log saved at /home/ljh/3308/5.10_sdk/output/sessions/2023-09-15_17-56-01

Pick a defconfig:

1. rockchip_defconfig
2. rockchip_32bit_defconfig
3. rockchip_rk3308_rtos_amp_32bit_defconfig
4. rockchip_rk3308_rtos_linux_64bit_defconfig
5. rockchip_rk3308_rtos_smp_32bit_defconfig
6. rockchip_rk3308b_32bit_defconfig
7. rockchip_rk3308b_64bit_defconfig
8. rockchip_rk3308bs_32bit_defconfig
9. rockchip_rk3308bs_32bit_display_defconfig
10. rockchip_rk3308bs_64bit_defconfig
11. rockchip_rk3308bs_evb_v20_ia_32bit_defconfig
12. rockchip_rk3308h_32bit_defconfig
13. rockchip_rk3308hs_32bit_defconfig
Which would you like? [1]: 10
```

4.3 SDK配置不同启动/内核/系统等组件

SDK可通过`make menuconfig`进行相关配置，目前可配组件主要如下：

```
(rk3308) SoC
   Rootfs  --->
   Loader (u-boot)  --->
   Kernel  --->
   Boot  --->
   Recovery (buildroot)  --->
   PCBA test (buildroot)  --->
   Security  --->
   Extra partitions  --->
   Firmware  --->
   Update (Update image, OTA and A/B)  --->
   Others configurations  --->
```

通过以上config，可选择不同rootfs/loader/kernel等配置，进行各种定制化编译。另外还带有强大命令行切换功能。

> 注意：menuconfig配置之后，需要`make savedefconfig`保存配置

4.4 自动编译

进入工程根目录执行以下命令自动完成所有的编译：

```
./build.sh all # 只编译模块代码（u-Boot，kernel，Rootfs，Recovery）
               # 需要再执行`./build.sh ./mkfirmware.sh 进行固件打包

./build.sh     # 编译模块代码（u-Boot，kernel，Rootfs，Recovery）
               # 打包成update.img完整升级包
               # 所有编译信息复制和生成到out目录下
```

4.5 各模块编译及打包

4.5.1 U-Boot编译

```
./build.sh uboot
```

4.5.2 Kernel编译

- 方法一

```
./build.sh kernel
```

- 方法二

```
cd kernel
export CROSS_COMPILE=../prebuilts/gcc/linux-x86/aarch64/gcc-arm-10.3-2021.07-
x86_64-aarch64-none-linux-gnu/bin/aarch64-none-linux-gnu-
make ARCH=arm64 rk3308_linux_defconfig
make ARCH=arm64 rk3308bs-evb-amic-v11.img -j8
或
make ARCH=arm64 rk3308_linux_defconfig rk3308bs_mipi_display.config
make ARCH=arm64 rk3308bs-evb-mipi-display-v11.img -j8
```

4.5.3 Recovery编译

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

注：Recovery是非必需的功能，有些板级配置不会设置

4.5.4 Buildroot 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_rk3308_bs_release/images下生成rootfs.squashfs。

4.5.5 交叉编译

4.5.5.1 SDK目录内置交叉编译

SDK prebuilts目录预置交叉编译，如下：

目录	说明
prebuilts/gcc/linux-x86/aarch64/gcc-arm-10.3-2021.07-x86_64-aarch64-none-linux-gnu	gcc arm 10.3.1 64位工具链
prebuilts/gcc/linux-x86/arm/gcc-arm-10.3-2021.07-x86_64-arm-none-linux-gnueabihf	gcc arm 10.3.1 32位工具链

4.5.5.2 Buildroot内置交叉编译

可通过 `source buildroot/envsetup.sh` 来设置不同目标功能的配置

```
~/3308/5.10_sdk$ source buildroot/envsetup.sh
Top of tree: /home/ljh/3308/5.10_sdk

Pick a board:

...
11. rockchip_rk3308_32_release
12. rockchip_rk3308_b_32_release
13. rockchip_rk3308_b_release
14. rockchip_rk3308_bs_32_ia_release
15. rockchip_rk3308_bs_32_lvgl_release
16. rockchip_rk3308_bs_32_release
17. rockchip_rk3308_bs_lvgl_release
18. rockchip_rk3308_bs_release
19. rockchip_rk3308_h_32_release
20. rockchip_rk3308_recovery
21. rockchip_rk3308_release
...
Which would you like? [1]:
```

默认选择18，`rockchip_rk3308_bs_release` 。然后进入RK3308的Buildroot目录，开始相关模块的编译。

其中`rockchip_rk3308_bs_32_release`是32位Buildroot系统编译，`rockchip_rk3308_recovery`是用来编译Recovery模块。

比如编译 rockchip_test 模块，常用相关编译命令如下：

进入 buildroot目录

```
SDK#cd buildroot
```

- 编译 rockchip-test

```
buildroot#make rockchip-test
```

- 重编 rockchip-test

```
buildroot#make rockchip-testt-rebuild
```

- 删除 rockchip-test

```
buildroot#make rockchip-test-dirclean
或者
buildroot#rm -rf output/rockchip_rk3562/build/rockchip-test-master/
```

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。比如RK3308,其交叉编译工具位于`buildroot/output/rockchip_rk3308_bs_release/host/usr`目录下，需要将工具的bin/目录和`aarch64-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：

```
source buildroot/envsetup.sh rockchip_rk3308_bs_release
```

输入命令查看：

```
cd buildroot/output/rockchip_rk3308_bs_release/host/usr/bin
./aarch64-linux-gcc --version
```

此时会打印如下信息：

```
aarch64-linux-gcc.br_real (Buildroot -g900f5662) 11.3.0
```

4.5.6 固件的打包

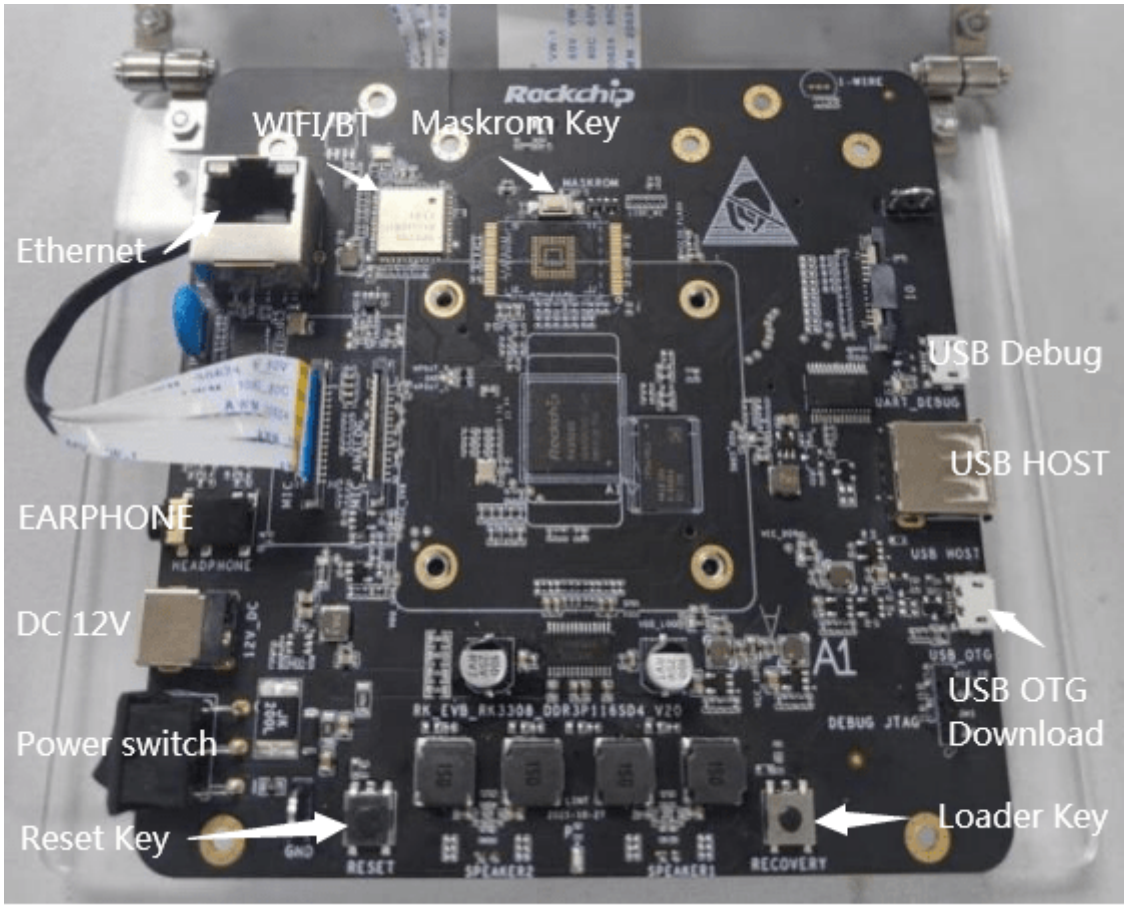
上面 Kernel/U-Boot/Recovery/Rootfs 各个部分的编译后，进入工程目录根目录执行以下命令自 动完成所有固件打包到`output/firmware`目录下：

固件生成:

```
./build.sh firmware
```

5. 刷机说明

RK3308B EVB V20 开发板正面接口分布图如下：

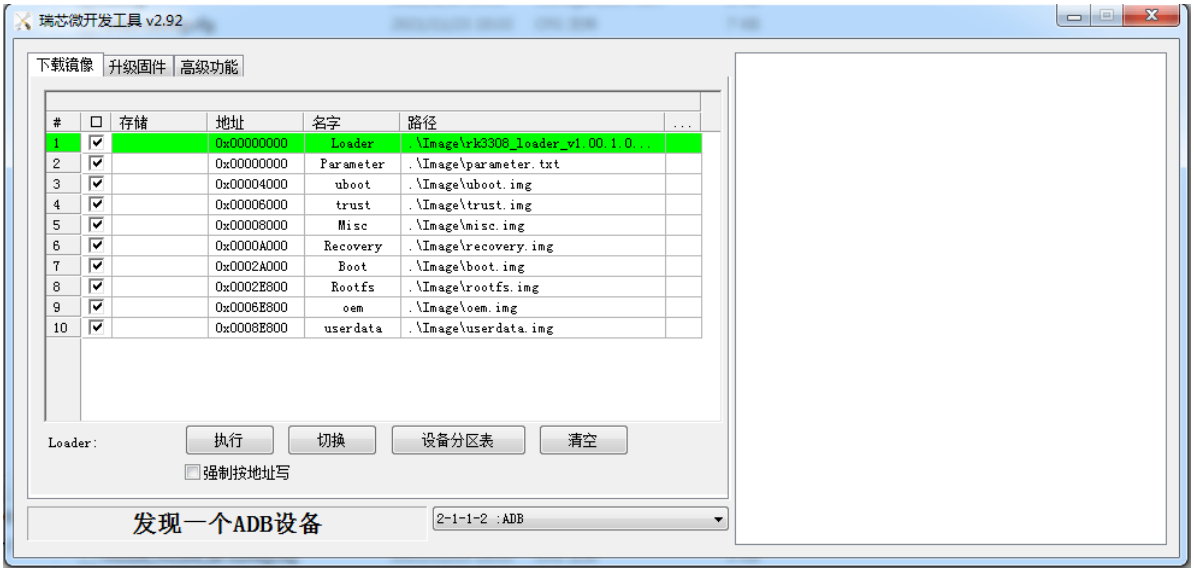


5.1 Windows 刷机说明

SDK 提供 Windows 烧写工具(工具版本需要 V2.91 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_v5.11.zip
```

5.2 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 V2.1 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 -noreset
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 rocdev/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
```

5.3 系统分区说明

默认分区说明 (下面是 RK3308 EVB 分区参考）

- 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。
- rootfs 分区：供 buildroot、或 debian 编出来的 rootfs.img。
- oem 分区：给厂家使用，存放厂家的 APP 或数据。挂载在 /oem 目录。
- 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。

6. RK3308 SDK 固件

链接：https://console.zbox.filez.com/l/yJ42Ab