密级状态：绝密（ ） 秘密（ ） 内部（ ） 公开（ √ ）

Security Class: Top-Secret ( ) Secret ( ) Internal ( ) Public (√)

Rockchip Recovery 用户操作指南
Rockchip Recovery User Guide

（技术部/Technical Department）

文件状态： Status： [ ] 正在修改 [ ] Modifying [√] 正式发布 [√] Released	当前版本： Current Version：	V1.05
	作 者： Author：	杨汉兴、黄开辉、吴良清、纪大峣 Yang Hanxing, Huang Kaihui, Wu Liangqing, Ji Dayao
	完成日期： Finish Date：	2019-06-19
	审 核： Auditor：	
	完成日期： Finish Date:	

福州瑞芯微电子股份有限公司

Fuzhou Rockchip Electronics Co., Ltd.

版 本 历 史 Revision History

版本号 Version no.	作者 Author	修改日期 Revision Date	修改说明 Revision Description	备注 Remark
V1.00	杨汉兴、黄开辉 Yang Hanxing, Huang Kaihui	2016/9/2	构建初始版本 Initial version release	
V1.01	吴良清 Wu Liangqing	2017/6/7	增加 android 7.1 verifying boot ota 的修改 Add the modification of Android7.1 verifying boot ota	
V1.02	黄开辉 Huang Kaihui	2017/8/30	增加 android 7.1 parameter 升级说明 Add the upgrade instruction of Android7.1 parameter	
V1.03	黄开辉 Huang Kaihui	2017/11/23	增加 Backup 功能使用说明 Add the usage instruction of Backup function	
V1.04	纪大峣 Ji Dayao	2019/4/11	1. 更新第 4 节，Android 7.x 及以上平台不再支持系统直接升级 update.img Update section 4, system doesn't support directly upgrade update.img any more on Android7.x and above platforms 2. 更新 5.2 旋转配置 Update the rotation configuration in section 5.2	

V1.05	纪大峣 Ji Dayao	2019/6/19	1. 更新 2.3 Update 2.3 2. 删除不使用内容，包括原 2.4，3.2，3.3 和 3.4 等 Remove un-used contents, including previous 2.4, 3.2, 3.3, 3.4 etc.	

目 录 Contents

前 言 Preface

概述 Overview

文档主要介绍 Rockchip Recovery 用户操作指南，旨在帮助软件开发工程师更快上手 Recovery 升级的开发及调试。

This document mainly describes Rockchip Recovery user guide, aiming to help software engineers familiar with Recovery upgrading development and debugging quickly.

产品版本 Product version

芯片名称 Chipset name	内核版本 kernel version	Android 版本 Android version
		6.0.1

读者对象 Object

本文档（本指南）主要适用于以下工程师：

This document (guide) is mainly suitable for below engineers:

技术支持工程师

Field application engineers

软件开发工程师

Software development engineers

1 Recovery 简介 Recovery introduction

1.1 Recovery 模式简介 Recovery mode introduction

Recovery 模式指的是一种可以对安卓机内部的数据或系统进行修改的模式，（类似于 windows pe 或 DOS）。在这个模式下我们可以刷入新的安卓系统，或者对已有的系统进行备份或升级，也可以在此恢复出厂设置。

Recovery mode is a mode that can modify the internal data or system of the Android device (similar to windows pe or DOS). In this mode, we can flash new Android system, or back up or upgrade the existing system, also can reset to factory default setting.

1.2 Recovery 模式在框架层的位置 Recovery mode location in framework layer

Android 启动后，会先运行 bootloader。Bootloader 会根据某些判定条件决定是否进入 recovery 模式。Recovery 模式会装载 recovery 分区，该分区包含 recovery.img。Recovery.img 包含了标准内核(和 boot.img 中的内核相同)以及 recovery 根文件系统。

After Android boots up, it will firstly run bootloader. Bootloader will decide whether to enter recovery mode according to some determination conditions. Recovery mode will load recovery image, which includes the standard kernel (same as the kernel in boot.img) and recovery root file system.

Android recovery 三个部分两个接口，recovery 的工作需要整个软件平台的配合，从架构角度看，有三个部分：

Android recovery includes three parts and two interfaces. The work of recovery needs the support of the whole software platform. There are three parts from architecture point of view:

1. Main System：用 boot.img 启动的 Linux 系统，Android 的正常工作模式。

   Main System: Linux system booted up by boot.img, the normal working mode of

Android.

2. Recovery：用 recovery.img 启动的 Linux 系统，主要是运行 recovery 程序。

   Recovery: Linux system booted up by recovery.img, mainly to run recovery program.

3. Bootloader：除了加载、启动系统，还会通过读取 flash 的 MISC 分区获得来自 Main System 和 Recovery 的消息，并以此决定做何种操作。

   Bootloader: besides loading and starting the system, also can obtain the message from Main System and Recovery by reading MISC partition of flash and decide the operation based on it.

   两个通信接口：

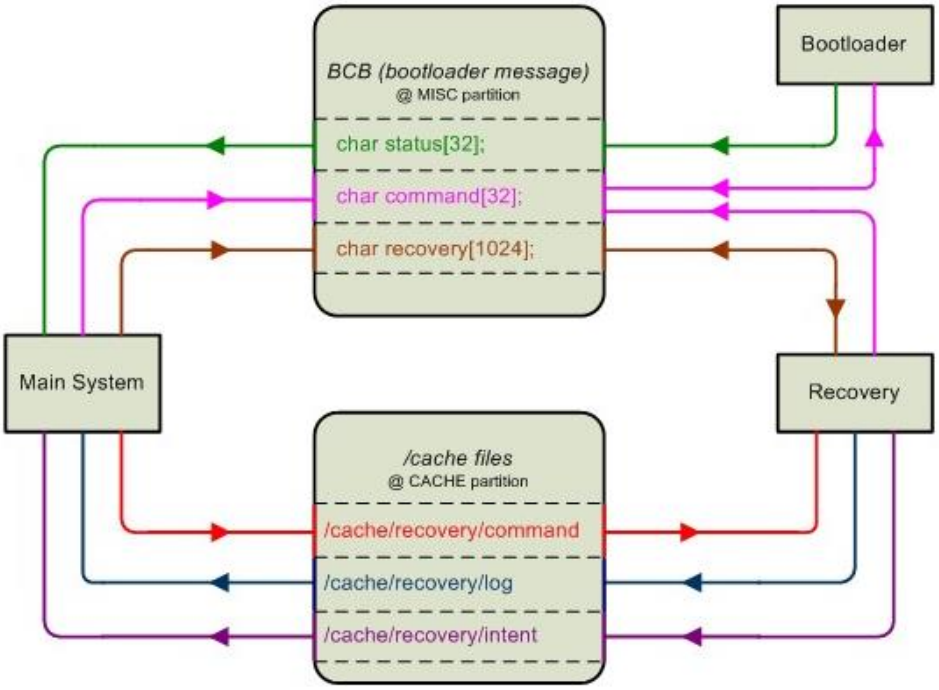
   Two communication interfaces:

1. /cache/recovery/：command、log、intent
2. BCB（Bootloader Control Block）：misc 分区

   BCB (Bootloader Control Block): misc partition

   我们先来看以上三部分是如何通信的，先看下图：

   Please refer to below picture to see how the above three parts communicate:



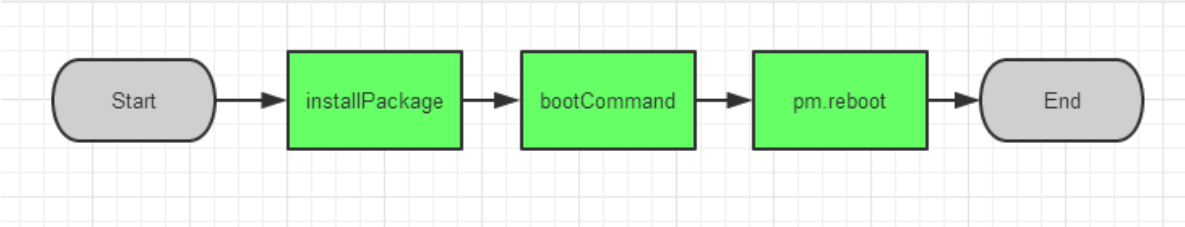
Main System 如何进入 Recovery 模式：当我们在 Main System 使用 update.zip 包进行升级时，系统会重启并进入 recovery 模式。在系统重启前，我们可以看到 Main System 定会向 command 域写入 boot-recovery（粉红色线），用来告知 bootloader 重启后进入 Recovery 模式。这一步是必须的，至于 Main System 是否会向 recovery 域写入值我们在源码中不能肯定这一点。即便如此，重启进入 Recovery 模式后，Bootloader 会从 /cache/recovery/command 中读取值并放入到 BCB 的 recovery 域。而 Main System 在重启之前肯定会向/cache/recovery/command 中写入 Recovery 将要进行的操作命令。

How Main System enters Recovery mode: when Main System uses update.zip package to upgrade, the system will reboot and enter recovery mode. Before system reboots, we can see Main System will write boot-recovery into command field (pink line), which is used to tell bootloader to enter Recovery mode after reboot. This step is necessary. We cannot confirm from the source code whether Main System will write the value into recovery field or not, however, after entering Recovery mode, Bootloader will read the value from /cache/recovery/command and put it into the recovery field of BCB. Main System will definitely write Recovery operation command

into /cache/recovery/command before reboot.

下图是 Main System 进入 Recovery 模式调用接口的流程图：

Below picture shows the process of Main System invoking interface after entering Recovery mode:



1. installPackage: RecoverySystem 的接口，完成升级包路径转换，并调用 bootCommand。

   installPackage: the interface of RecoverySystem, complete the path conversion of upgrade package, and invoke bootCommand.

2. bootCommand：RecoverySystem 的接口，将命令写入/cache/recovery/command，并调用 pm.reboot。

   bootCommand: the interface of RecoverySystem, write the command into /cache/recovery/command, and invoke pm.reboot.

3. Pm.reboot:PowerManager 的接口，重启并进入 Recovery 模式。

   Pm.reboot: the interface of PowerManager, reboot and enter Recovery mode.

2 编译 OTA 包 OTA package compilation

2.1 OTA 介绍 OTA introduction

OTA （over the air）升级是 Android 系统提供的标准软件升级方式。它功能强大，提供了完全升级（完整包）、增量升级模式（差异包），可以通过本地升级，也可以通过网络升级。

OTA (over the air) upgrade is the standard software upgrade method provided by Android system. It provides complete upgrading (full package) and incremental upgrading mode (difference package). You can upgrade locally or over the network.

2.2 生成完整包 Generate complete package

完整包所包含内容：system、recovery、boot.img

The complete package includes: system, recovery, boot.img

发布一个固件正确的顺序：

The correct order to release an image:

1. make -j4
2. make otapackage -j4
3. ./mkimage.sh ota

发布固件必须使用./mkimage.sh ota,将 boot 与 kernel 打包，不需要单独烧 kernel，如果量产固件是分开的，将会影响后面差异包升级，除非你不需要用差异升级。

Must use "./mkimage.sh ota" to release image, pack boot and kernel, no need to flash kernel separately, if the MP image is separate, it will impact the incremental OTA in the future, unless you don't need to use the incremental OTA.

在 out/target/product/rkxxxx/目录下会生成 ota 完整包 rkxxxx-ota-eng.root.zip，改成 update.zip 即可拷贝到 T 卡或者内置的 flash 进行升级。

Ota complete package rkxxxx-ota-eng.root.zip will be generated in the directory of out/target/product/rkxxxx/, change it to be update.zip and then you can copy it to T card or built-in flash to do upgrading.

2.3 生成差异包 Generate the incremental OTA package

OTA 差异包只有差异内容，包大小比较小，主要用于 OTA 在线升级，也可 T 卡本地升级。OTA 差异包制作需要特殊的编译进行手动制作。

Incremental OTA package is relative small. It is mainly used for OTA on-line upgrade, and also can be used for T card local upgrade. It needs special compilation to manually make OTA difference package.

（1）首先发布 v1 版本的固件，生成 v1 版本的完整包

Firstly release v1 version image, generate v1 version complete package.

（2）保存 out/target/product/rkxxxx/obj/PACKAGING/target_files_intermediates/rk3188-target_files-eng.root.zip 为 rkxxxx-target_files-v1.zip，作为 v1 版本的基础素材包。

Save out/target/product/rkxxxx/obj/PACKAGING/target_files_intermediates/rk3188-target_files-eng.root.zip as rkxxxx-target_files-v1.zip, use it as the basic material package of v1 version.

（3）修改 kernel 代码或者 android 代码，发布 v2 版本固件，生成 v2 版本完整包

Modify kernel code or android code, release v2 version image, generate v2 version complete package.

（4）保存 out/target/product/rkxxxx/obj/PACKAGING/target_files_intermediates/rk3188-target_files-eng.root.zip 为 rkxxxx-target_files-v2.zip，作为 v2 版本的基础素材包。

Save out/target/product/rkxxxx/obj/PACKAGING/target_files_intermediates/rk3188-target_files-eng.root.zip as rkxxxx-target_files-v2.zip, use it as the basic material package of v2 version.

（5）生成 v1-v2 的差异升级包：

Generate the difference package of v1-v2:

Android 7.1 及以上版本，作差异包时必现加上 block 参数。

When Android Version >=7.1, must add "--block" option when making the difference package.

./build/tools/releasetools/ota_from_target_files -v -i rkxxxx-target_files-v1.zip -p out/host/linux-x86 -k build/target/product/security/testkey rkxxxx-target_files-v2.zip

out/target/product/rk3188/rkxxxx-v1-v2.zip

说明： 生成差异包命令格式:

Note: the command format to generate the difference package:

ota_from_target_files

-v -i 用于比较的前一个 target file

-v -i The former target file used for comparison

--block 使用 block 方式进行 OTA 升级，Android 版本>=7.1 时需加上这个参数

--block Use block to do OTA upgrade, please add this option when Android version >=7.1

-p host 主机编译环境

-p host The compiling environment of the host

-k 打包密钥

-k the private key for package

用于比较的后一个 target file

The latter target file used for comparison

生成的 ota 差异包

Incremental OTA package generated

2.4 注意事项 Notice

完整包和素材包是不一样的。

The OTA package and the material package are different.

从生成差异包的方法可以知道，差异包是两个版本的素材包做差异生成，所以每发布一个版本固件必须保存 obj 下的素材包以及升级的完整包和./mkimage.sh ota 生成的各个 image。

We can know from the way to generate incremental OTA package, the incremental OTA package is generated from two versions of material packages, so the material package in obj and the complete upgrade package and each image generated

by ./mkimage.sh ota must be saved for every version release.

如果有使用在线升级，必须给每个版本固件配置唯一的 ota 版本号：

If using on-line upgrade, must configure unique ota version number for every release version:

device/rockchip/rksdk/rksdk.mk 中的

ro.product.version = 1.0.0（也可以是其他格式的版本号字符串）

ro.product.version = 1.0.0 (also can be version number string in other format) in device/rockchip/rksdk/rksdk.mk

3 RK 平台分区升级说明 RK platform partition upgrade instruction

3.1 RkLoader 升级 RkLoader upgrade

一、 Loader 升级操作步骤

Loader upgrade step

a) 将需要升级的 loader，以 RK*Loader*.bin 格式放到 device/rockchip/rkxxxx/ota/loader 目录下，OTA 打包时就会加入到升级包中。

Put the loader which should be upgraded such as RK*Loader*.bin format to the directory of device/rockchip/rkxxxx/ota/loader, and it will be added to the upgrade package during OTA packing.

二、 注意事项：

Notices:

a) android5.1 放到目录 device/rockchip/common/loader，如果没有请创建。

Put android5.1 to the directory of device/rockchip/common/loader, please create if not existing.

b) android 4.4 放到目录 device/rockchip/rksdk/loader，如果没有请创建。

Put android4.4 to the directory of device/rockchip/rksdk/loader, please create if

not existing.

c) Loader 必须保证版本号比固件中的版本号更高。

Loader version number must be higher than the version number of image in the device.

3.2 releasetool 说明 releasetool instruction

releasetool 是一个描述 RK 平台分区升级的 Python 脚本。

Releasetool is a Python script describing RK platform partition upgrade.

1. 下面是一些必须要实现的 extension 接口，供外部调用：

Below are some extension interfaces which must be implemented for external invocation:

FullOTA_Assertions： 全量包升级脚本头部加入升级命令

FullOTA_Assertions: Add upgrade command in the head of full package upgrade script

IncrementalOTA_Assertions： 增量包升级脚本头部加入升级命令

IncrementalOTA_Assertions: Add upgrade command in the head of incremental package upgrade script

FullOTA_InstallEnd： 全量包尾部加入升级命令

FullOTA_InstallEnd: Add upgrade command in the end of full package

IncrementalOTA_InstallEnd： 增量包尾部加入升级命令

IncrementalOTA_InstallEnd: Add upgrade command in the end of incremental package

2. 下面是 RK 平台分区升级接口：

Below are RK platform partition upgrade interfaces:

InstallRKLoader： loader 升级 loader upgrade

InstallUboot ： uboot 升级 uboot upgrade

InstallTrust ： trust 升级 trust upgrade

3.3 Backup 分区使用说明 Backup partition usage instruction

当机器出现意外无法正常启动进入 Android，用户可以引导进入 Recovery，在 Recovery 菜单栏中选中 Recovery System 选项，进行操作系统恢复。

When the device cannot normally boot up to Android due to abnormality, users can boot into Recovery, select Recovery System option in Recovery menu bar, and restore the operation system.

使用该功能，必须将 update.zip 打包成 ext4 的 image，并烧写到/backup 中，操作可如下：

In order to use this function, must pack update.zip into ext4 image first, and then flash it to /backup. The steps are as below:

a) Android 5.1 6.0 可使用如下脚本将 update.zip 打包成 update_back.img 固件

For Android5.1, 6.0, you can use below script to pack update.zip into update_back.img.

```
#!/bin/bash
ota_size= ls -l ./update/update.zip | awk '{print $5;}'`
let increament_size=$ota_size/20
let ota_size=$ota_size+$increament_size

MAKE_EXT4FS_ARGS=" -L system -S $OUT/root/file_contexts -a backup ./update_back.img ./update/"
make_ext4fs -l $ota_size $MAKE_EXT4FS_ARGS
```

Android 7.1 可使用如下脚本将 update.zip 打包成 update_back.img 固件

For Android7.1, you can use below script to pack update.zip into update_back.img.

```
#!/bin/bash
croot
make img2simg_host
cd -
ota_size=`ls -l ./update/update.zip | awk '{print $5;}'`
let increament_size=$ota_size/20
let ota_size=$ota_size+$increament_size


MAKE_EXT4FS_ARGS=" -L system -S $OUT/root/file_contexts -a backup ./update_back.img ./update/"
make_ext4fs -l $ota_size $MAKE_EXT4FS_ARGS
img2simg ./update_back.img ./out.img
mv ./out.img ./update_back.img
```

脚本使用注意事项：

The notices of script usage:

1）脚本执行是基于编译环境，所以必须执行以下两条命令

The script is executed based on compiling environment, so you must execute below two commands first:

source build/envsetup.sh

lunch

2）在脚本的同级目录创建 update 目录，并将 OTA 完整包 update.zip 复制到该目录中

Create update directory at the level of the script directory, and copy OTA full package update.zip into this directory.

b） 修改 parameter 中 backup 分区的大小，使其足够放 update_back.img

Modify the backup partition size in parameter, to make it enough to save update_back.img.

c） 将 update_back.img 打包到 update.img 中，通过修改 packag-file，如下图

Pack update_back.img into update.img by modifying packag-file, as shown below:

```
# NAME          Relative path
#
#HWDEF         HWDEF
package-file    package-file
bootloader    Image/MiniLoaderAll.bin
parameter     Image/parameter.txt
#trust             Image/trust.img
uboot             Image/uboot.img
misc          Image/misc.img
resource      Image/resource.img
kernel        Image/kernel.img
boot              Image/boot.img
recovery      Image/recovery.img
system        Image/system.img
# 要写入backup分区的文件就是自身（update.img）
# SELF 是关键字，表示升级文件（update.img）自身
# 在生成升级文件时，不加入SELF文件的内容，但在头部信息中有记录
# 在解包升级文件时，不解包SELF文件的内容.
backup        Image/update_back.img
#update-script  update-script
#recover-script recover-script
~
```

代码说明：

Code instruction:

Android 5.1 6.0 补丁放于百度云：

The patch of Android5.1, 6.0 is saved in BaiduCloud:

链接：http://pan.baidu.com/s/1dEIGNzr 密码：sq0j

Link: http://pan.baidu.com/s/1dEIGNzr Password: sq0j

4 Update.img 升级方式 Update.img upgrade method

4.1 支持存储方式 Support memory upgrade

注：Android 7.x 及以上平台不再支持这种升级方式，请使用标准 update.zip 方式升级。

Note: Android 7.x and above platforms don't support this kind of upgrade any more, please use standard update.zip to upgrade.

可以将 update.img 存到 SD 卡根目录，USB 根目录，内部存储/data/media/0/ 下，如果系统检测到有该包，会自动检测包是否合法，如果合法会重启进入 recovery 模式，并进行升级。

You can save update.img to SD card root directory, USB root directory, internal memory /data/media/0/, if the system detects the package, it will check if the package is valid, if it is valid, it will reboot and enter recovery mode to upgrade.

4.2 打包说明 Packing instruction

1. 进入到目录 RKTools/linux/Linux_Pack_Firmware/rockdev/

   Enter the directory of RKTools/linux/Linux_Pack_Firmware/rockdev/

2. 修改 package-file 文件，指向生成 img 的路径，或者将 img 拷贝进来。如图：

   Modify package-file file, point to the path img generated, or copy the img, as shown below:

```
# NAME      Relative path
#
#HWDEF      HWDEF
package-file    package-file
bootloader  Image-rk3399_box/RK3399MiniLoaderAll_V1.05.bin
parameter   Image-rk3399_box/parameter.txt
trust       Image-rk3399_box/trust.img
uboot       Image-rk3399_box/uboot.img
misc        Image-rk3399_box/misc.img
resource    Image-rk3399_box/resource.img
kernel      Image-rk3399_box/kernel.img
boot        Image-rk3399_box/boot.img
recovery    Image-rk3399_box/recovery.img
system      Image-rk3399_box/system.img
# 要写入backup分区的文件就是自身（update.img）
# SELF 是关键字，表示升级文件（update.img）自身
# 在生成升级文件时，不加入SELF文件的内容，但在头部信息中有记录
# 在解包升级文件时，不解包SELF文件的内容。
backup      RESERVED
#update-script  update-script
#recover-script recover-script
```

3. Parameter.txt：确认你的 parameter.txt 中 MACHINE_MODEL 的值是否正确，该值将用来校验 img 包是否合法，并且不能包含空格。如图所示，rk3399-box 前后都不能有空格。

Parameter.txt: confirm if the value of MACHINE_MODEL in your parameter.txt is correct or not. This value will be used to check if img package is valid and cannot include space. As shown in below picture, there must be no space before or after rk3399-box.

```
FIRMWARE_VER: 6.0.1
MACHINE_MODEL:rk3399-box
MACHINE_ID: 007
MANUFACTURER: RK3399
```

4. 运行 mkupdate.sh 脚本,会在该目录下生成 update.img,将拷贝到目标路径，即可进行升级。

Run mkupdate.sh script will generate update.img in the directory. Copy it to the target path can do upgrade.

5 配置说明 Configuration instruction

5.1 Log 重定向 Log re-direct

1. 功能说明：Log 可以输出到串口、SD 卡、/cache/recovery/、三个地方。

Function description: Log can be output to serial port, SD card, or /cache/recovery/.

2. 打开方式：修改 bootable/recovery/Android.mk 文件：

Enable method: modify bootable/recovery/Android.mk file:

REDIRECT_LOG_TO := UART 将日志输出到串口

REDIRECT_LOG_TO := UART output the log to serial port

REDIRECT_LOG_TO := CACHE 将日志输出到/cache/recovery/目录下

REDIRECT_LOG_TO := CACHE output the log to /cache/recovery/ directory

REDIRECT_LOG_TO := SDCARD 将日志输出到 SD 卡中

REDIRECT_LOG_TO := SDCARD output the log to SD card

3. 支持平台：android 6.0 及以上。

Support platform: Android6.0 and higher

5.2 屏幕旋转 Screen rotation

1. 功能说明：Recovery 屏幕可以旋转 0°，90°，180°，270°。

Function description: Recovery screen can be rotated 0°, 90°, 180°, 270°

2. 打开方式：修改 device/rockchip/common/BoardConfig.mk 文件：

Enable method: modify device/rockchip/common/BoardConfig.mk file

ROTATE_SCREEN := rotate_0 不旋转 No rotation

ROTATE_SCREEN := rotate_90 旋转 90° Rotate 90°

ROTATE_SCREEN := rotate_180 旋转 180° Rotate 180°

ROTATE_SCREEN := rotate_270 旋转 270° Rotate 270°

BOARD_HAS_FLIPPED_SCREEN := true 旋转 180°，针对 LCD 屏幕装反情况。

Rotate 180° when LCD panel is assembled reversely.

3.支持平台：android 6.0， 5.1 以补丁的形式存在。

Support platform: patch available for Android6.0, 5.1.

4.对于 Android 8.1 及以上平台，旋转配置方式如下：

For Android8.1 and above platforms, the rotation configurations are as below:

TARGET_RECOVERY_DEFAULT_ROTATION := ROTATION_NONE 不旋转

TARGET_RECOVERY_DEFAULT_ROTATION := ROTATION_NONE No rotate

TARGET_RECOVERY_DEFAULT_ROTATION := ROTATION_LEFT 旋转 270

TARGET_RECOVERY_DEFAULT_ROTATION := ROTATION_LEFT Rotate 270

TARGET_RECOVERY_DEFAULT_ROTATION := ROTATION_RIGHT 旋转 90

TARGET_RECOVERY_DEFAULT_ROTATION := ROTATION_RIGHT Rotate 90

TARGET_RECOVERY_DEFAULT_ROTATION := ROTATION_DOWN 旋转 180

TARGET_RECOVERY_DEFAULT_ROTATION := ROTATION_DOWN Rotate 180

5.3 屏幕分屏 Screen split

1. 功能说明：Recovery 屏幕分屏，将屏幕分为左右两部分，显示内容一样，用于 VR 功能。

Function description: Recovery screen split, divide the screen into left and right parts, displaying the same content, used for VR product.

2. 打开方式：修改 bootable/recovery/Android.m 文件：

Enable method: modify bootable/recovery/Android.m file:

DOUBLE_SCREEN := NO 不分屏 disable

DOUBLE_SCREEN := YES 分屏 enable

3. 支持平台：android 6.0，5.1 以补丁的形式存在。

Support platform: patch available for Android6.0, 5.1.

5.4 文件保存 File save

1. 功能说明：将文件保存到/cache/recovery/Recovery_*，格式化不会清除该文件。

Function description: save the file to /cache/recovery/Recovery_*, and formatting will not clear this file.

2. 支持平台：android 6.0，android 7.1

Support platform: Android6.0, Android7.1

5.5 RecoveryUI 说明 RecoveryUI instruction

recovery 作为一个简单的 rootfs，提供非常有限的几个功能，只包含了几个简单的库，UI 显示采用的是直接刷 framebuffer 的形式。首先浏览一下 recovery main 函数里面的流程

recovery as a simple rootfs provides very limited functions, only contains several simple libraries, and UI display uses the way of directly brushing framebuffer. Firstly look through the process in recovery main function:

```
Device* device = make_device();
ui = device->GetUI();
gCurrentUI = ui;

ui->SetLocale(locale);
ui->Init();
```

如上代码所示，make_device()函数返回一个 Device 设备，所以我们只需要继承 Device 类（一些虚函数接口），就可对 RecoveryUI 进行简单的修改。

As shown in above code, make_device() function returns a Device, so we just need to inherit Device class (some virtual function interfaces) to make simple modification to RecoveryUI.

1. 功能说明：自定义 recovery 菜单与按键实现

Function description: customize recovery menu and button implementation

2. 打开方式：修改 device/rockchip/common/BoardConfig.mk 文件：

Enable method: modify device/rockchip/common/BoardConfig.mk file:

TARGET_RECOVERY_UI_LIB ?= librecovery_ui_rk30sdk

3. 修改方式：device/rockchip/common/recovery，里面有例子，可修改或参照。

Modification: you can modify or refer to the example in device/rockchip/common/recovery.

5.6 支持 EXT4 文件系统 Support EXT4 file system

1. 打开方式：修改 device/rockchip/common/BoardConfig.mk 文件：

Enable method: modify device/rockchip/common/BoardConfig.mk file:

a) TARGET_USERIMAGES_USE_EXT4 ?= true

5.7 Update.img drm 签名校验 Update.img drm signature verification

1.功能说明：update.img 在升级时进行 drm 签名校验，防止第三方非法固件升级。Update.img 必须使用 secureboot 工具签名，并勾选 sign check 复选框。

Function description: update.img performs drm signature verification during upgrade to avoid illegal third party firmware upgrade. Update.img must be signed using secureboot tool, and select sign check checkbox.

2.打开方式：修改 device/rockchip/common/BoardConfig.mk 文件：

Enable method: modify device/rockchip/common/BoardConfig.mk file:

a) RECOVERY_UPDATEIMG_RSA_CHECK ?= true

3.支持平台：android4.4, android5.1

Support platform: Android4.4, Android5.1

5.8 Boardid 串号 Boardid serial number

1.功能说明：一种根据烧录 Boardid 串号实现一种固件支持不同硬件不同国家定制的 one image，该方案支持谷歌 OTA 完整升级和差异升级。

Function description: a firmware implementation based on the flashed Boardid serial number supports one image customized for different hardware and different countries. This solution supports Google OTA complete upgrade and difference upgrade.

2.打开方式：修改 device/rockchip/common/BoardConfig.mk 文件:

Enable method: modify device/rockchip/common/BoardConfig.mk file:

a) RECOVERY_BOARD_ID ?= true

3.支持平台：android 4.4,android5.1

Support platform: Android4.4, Android5.1

5.9 升级 U 盘升级包 Update the upgrade package of U disk

1.功能说明：当按键进入 recovery 时，自动升级 U 盘中的升级包(update.img 或 update.zip)。

Function description: when press the button to enter recovery, it will automatically update the upgrade package in U disk (update.img or update.zip).

2.打开方式：修改 device/rockchip/common/BoardConfig.mk 文件:

Enable method: modify device/rockchip/common/BoardConfig.mk file:

a) RECOVERY_AUTO_USB_UPDATE ?= false

3.支持平台：android 5.1

Support platform: Android5.1

5.10 代码结构 Code structure

本节将对 recovery 涉及的相关代码结构进行简单的说明。

This chapter will simply describe recovery related code structure.

1. Bootable/recovery/:

a) Recovery 的主程序代码，其中 recovery.cpp 是入口，

Recovery main program code, recovery.cpp is the entry

b) rkimage.cpp 是处理 update.img 的升级流程，

rkimage.cpp handles the upgrade process of update.img

c) install.cpp 是处理 update.zip 的升级流程。

install.cpp handles the upgrade process of update.zip

2. Build/tools/releasetools/:

a) OTA 升级包的 Python 脚本，控制完整包与差异包的生成。

Python script of OTA upgrade package, control the generation of full package and incremental package.

3. Build/tools/drmsigntool/:

a) 如果开启 drm，生成 ota 包时对 boot.img 进行签名，使用 build/target/product/security/private.key,保证进行 OTA 升级后，drm 功能还能正常使用。

If drm is enabled, sign boot.img during ota package generation, use build/target/product/security/private.key, to ensure that drm function can still work normally after OTA upgrade.

4. Build/tools/remkloader/:

a) 编译 OTA 时打包可升级的 loader 的工具。

The tool used to pack the upgrade loader when compiling OTA package.

5. Build/target/product/security/:

a) 编译 OTA 包时签名使用的密钥。

The private key used for signature when compiling OTA package.

6.device/rockchip/common/recovery/:

a) Recovery 菜单及按键定制

Recovery menu and button customization

b) 在 android 4.4 上是 device/rockchip/rksdk/recovery 目录

It is device/rockchip/rksdk/recovery directory for Android4.4

8. Device/rockchip/common/loader:

a) 将需要升级的 loader 放在该目录，可以打包到 ota 升级包中。

Put the upgrade loader in this directory, can be packed into ota upgrade package.

b) 4.4 目录 device/rockchip/rksdk/loader

device/rockchip/rksdk/loader directory for Android4.4

c) 6.0 目录 device/rockchip/rkxxxx/ota/loader

device/rockchip/rkxxxx/ota/loader directory for Android6.0

9. out/target/product/rk3188/obj/PACKAGING/target_files_intermediates:

a) OTA 升级包编译生成的素材包，要做差异包必须保存该素材包。

The material package generated by compiling OTA upgrade package, if want to make the difference package, this material package must be saved.

10. Device/rockchip/common/releasetools.py

a) 定义 OTA 包中包含的内容。

Define the contents in the OTA package.

6 升级步骤 Upgrade steps

6.1 OTA 包本地升级操作步骤 OTA package local upgrade steps

1. 参照第二章，生成 OTA 包，并重新命名为 update.zip

Refer to chapter 2, generate OTA package, and rename it as update.zip.

2. 将 update.zip 拷贝到 USB、SD 卡根目录，或/data/media/0/ 目录下

Copy update.zip to USB, SD card root directory, or /data/media/0/ directory.

3. 会自动检测升级包，并弹出升级对话框。

It will automatically detect the upgrade package, and pop up the upgrade dialog.

4. 自动重启升级：重启进入 Recovery 系统中自动完成 OTA 包的升级，此时不能断电，等待升级完成会自动重启到 Android 主界面。

Automatic reboot and upgrade: reboot to Recovery system and automatically upgrade the OTA package, cannot power down during the period, it will automatically reboot to Android main interface after upgrade completed.

6.2 OTA 包网络升级操作步骤 OTA package network upgrade steps

1. 自动检测网络 OTA 升级包，提示下载升级。

   Automatically detect the network OTA upgrade package, prompt to download and upgrade.

2. 自动重启升级。

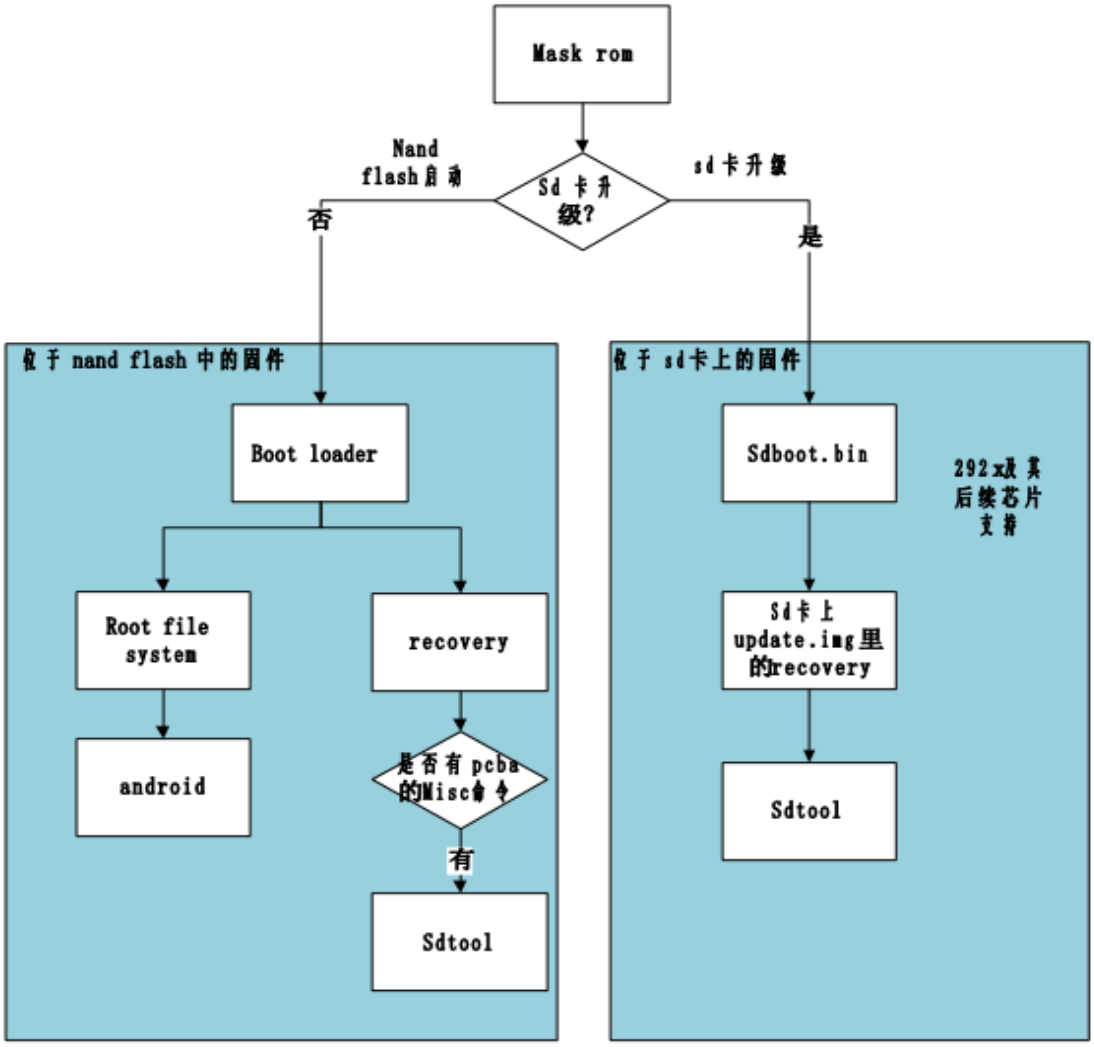
   Automatically reboot and upgrade.

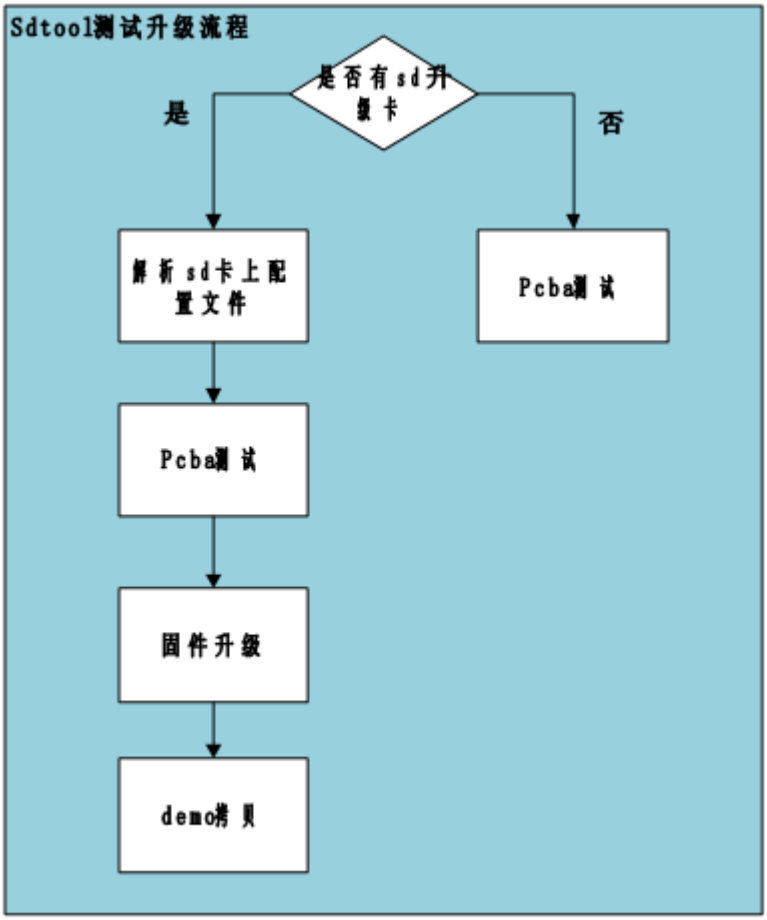
7 量产指导 MP Guidance

7.1 升级原理简介 Upgrade principle brief introduction

本节旨在指导开发商如何对 RK 平台 Android 终端进行量产升级，以及让开发工程师理解升级原理。在介绍升级流程之前，我们先大致了解一下 RK 平台的固件加载流程。

This chapter mainly describes for customers how to do MP upgrade for Android terminal based on RK platform, and helps development engineers understand the upgrade principle. Let us have a general understanding of image loading process based on RK platform before introducing the upgrade process.


Mask rom
Nand flash启动
Sd卡升级?
sd卡升级
否
是
位于nand flash中的固件
Boot loader
Root file system
recovery
android
是否有pcba的Misc命令
有
Sdtool
位于sd卡上的固件
Sdboot.bin
292x及其后续芯片支持
Sd卡上update.img里的recovery
Sdtool



上电第一步，首先启动的是固化在芯片内部的 MaskRom：MaskRom 完成芯片基础的初始化后，将判断是否插入特殊的 SD 升级卡。这里分两种情况：

After power up, firstly it will start MaskRom solidified inside the chipset: after MaskRom finishes the basic initialization of the chipset, it will judge whether the special SD upgrade card is inserted. There are two cases:

第一种从 flash 启动，去加载位于 flash 中的 Bootloader，Bootloader 再根据 misc 分区中指令决定加载 recovery 还是 boot 的根文件系统，如果是加载根文件系统，那文件系统会把 android 的 system 挂载起来，启动各项服务。

The first one is to start from flash, load Bootloader in the flash, and then Bootloader determines to load recovery or boot root file system according to the instruction in the misc partition. If it is to load into boot, then root file system included in boot image will be loaded, and start the services.

第二种从 SD 卡启动，maskrom 引导卡上的 sdboot.bin，sdboot 会做一些文件系统初始化等操作，然后将 update.img 从 recovery 中解压出来在内存中运行，此时 recovery 会识别到命令启动工厂模式将控制权交给 sdtool，sdtool 里可以做一些工厂量产相关的处理流程，比如 pcba 测试，大固件升级，demo 文件拷贝等等。

The second one is to start from SD card, maskrom loads sdboot.bin of the card, sdboot will do file system initialization and some other operations, and then unzip update.img from recovery and run in the memory, now recovery will recognize the command to start the factory mode and give the control to sdtool, sdtool can do some factory MP related processes, such as pcba test, big image upgrade, demo file copy and so on.

这样我们可以得到几种升级模式：

In this way we can get several upgrade modes:

1. MaskRom 模式，系统在 Flash 中找不到 BootLoader，系统就会停留在 MaskRom 状态，等待 USB 连接 PC 升级。一般在贴片完成后，第一次开机，机器就停留在 MaskRom 状态；也可以通过开机时短接 Flash 的数据线，让系统扫描 Flash 出错而停留在 MaskRom 状态。 该状态下可以通过 pc 端工具直接烧写固件。

MaskRom mode, if system cannot find BootLoader in Flash, it will stay in MaskRom state, waiting for USB connecting with PC to upgrade. Generally the first time power up after SMT, the device will stay in MaskRom state. Also you can make system scanning Flash error and stay in MaskRom by short connecting the data line of Flash during power up. In this state, you can directly flash the image using PC tool.

2. BootLoader 模式，BootLoader 模式生效的前提是在机器中已经有烧入可工作的 Loader.bin。进入 BootLoader 模式的方法一般是通过组合键开机，在其他分区校验出错时， 系统也会停留在 BootLoader 状态。 这种模式下也可以通过 pc 端工具进行固件烧写。

BootLoader mode, the pre-condition to make BootLoader mode valid is that there is workable Loader.bin in the device. The general way to enter BootLoader mode is to

power up through the combination of buttons, when other partition verification error occurs, the system also will stay in BootLoader state. In this mode, you can also flash image through PC tool.

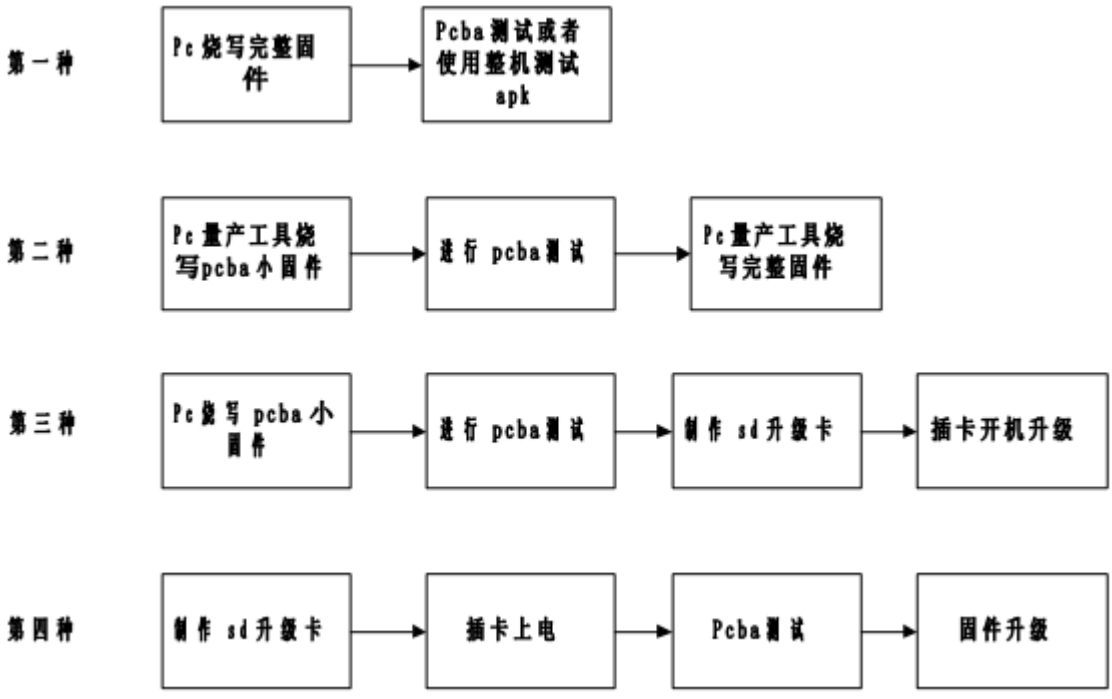
3. Recovery 模式，Recovery 包含两种模式，通过不同的启动命令区分，一种是最终用户模式，一种是工厂模式（也就是上图中的 sd 卡升级的情况）。最终用户模式包含本地升级功能，也是 OTA 升级的基础。工厂模式包括 pcba 测试、固件升级、音视频 demo 文件拷贝， 主要为了支持工厂生产的相关需求。

Recovery mode, Recovery includes two modes, using different boot commands to identify, one is the end user mode, and the other is factory mode (that is the sd card upgrade as above picture). End user mode includes local upgrade function, which is also the base of OTA upgrade. Factory mode includes pcba test, image upgrade, audio and video demo file copy, mainly to support factory production related requirements.

7.2 量产流程分析 MP process analysis

目前的量产流程主要分为四种：

Currently there are mainly four kinds of MP process:



前三种升级流程全系列芯片支持，第四种流程目前只有 292x 及以后的芯片支持。使用第四种流程可以完全脱离 pc，因此生产效率高。 对于 maskrom 非支持 sd 卡引导的芯片，建议使用第三种流程，贴片厂用 pc 烧一个 pcba 小固件进行测试，整机厂商使用 sd 卡升级完整固件。

The previous three upgrade processes are supported on all chipsets, and the forth process currently is only supported by 292x and later chipsets. The forth process doesn't need pc at all, so the production efficiency is high. For the chipset not supporting to use sd card to load maskrom, recommend to use the third process, SMT factory uses pc to flash a small image to do pcba test, while the device manufacturer uses sd card to upgrade the complete image.

7.3 编译固件注意事项 Notices of image compilation

这里再次强调，为了与 google 的 ota 升级统一，sd 卡升级时不会单独烧写 kernel 分区，kernel 需要跟 boot 一起打包。

Again, in order to be consistent with Google ota upgrade, sd card upgrade will not flash kernel partition separately. Kernel needs to be packed together with boot.

编译完，生成固件时使用./mkimage.sh ota 命令，这样生成出来的固件即是 boot 里包含 kernel 的固件。

After compilation, use "./mkimage.sh ota" to generate image. In this way the image generated is just the image with kernel included in boot.

7.4 制作 pcba 小固件 Make pcba small image

1. pcba_small_misc.img 用来定义一个 pcba 小固件，也就是说打包了该 misc 的 update.img 就具备了 pcba 测试小固件的功能。

   pcba_small_misc.img is used to define a pcba small image, that means, the update.img packed with this misc has the function of pcba testing small image.

2. 将 loader、parameter、pcba_small_misc.img、recovery 打包到 update.img 里：

Pack loader, parameter, pcba_small_misc.img and recovery into update.img:

请看打包工具的 package-file 文件配置方法：

```
# NAME    Relative path
#
#HWDEF    HWDEF
package-file  package-file
bootloader  RK292xLoader(L)_V1.18.bin
parameter parameter
misc    Image/pcba_small_misc.img
#kernel   Image/kernel.img
#boot        Image/boot.img
recovery  Image/recovery.img
#system   Image/system.img
# 要写入backup分区的文件就是自身（update.img）
# SELF 是关键字，表示升级文件（update.img）自身
# 在生成升级文件时，不加入SELF文件的内容，但在头部信息中有记录
# 在解包升级文件时，不解包SELF文件的内容。
# RESERVED不打包backup
#backup   backupimage/backup.img
backup RESERVED
#update-script  update-script
#recover-script recover-script
```

3. 打包好 pcba 小固件后通过量产工具烧写到机器后，每次启动机器都能自动进入 pcba 测试。

After packing pcba small image and flashing it to the device through MP tool, it will automatically enter pcba test when the device starts up every time.

7.5 制作带 pcba 测试功能的完整固件 Make the whole image with pcba testing function

1. pcba_whole_misc.img 用来定义带 pcba 功能的完整固件，也就是说打包了该 misc 的大固件具有烧写完第一次启动进行 pcba 测试的功能。

pcba_whole_misc.img is used to define the whole image with pcba function, that means, the big image packed with this misc has the function to do pcba testing for the first boot up after flashing.

2. 将 loader、parameter、pcba_whole_misc.img、boot、recovery、system 等都打包到 update.img 里，即是带 pcba 测试功能的完整固件。用量产工具烧入机器第一次启动会进入 pcba 测试，测试通过后下次启动会进入 android，测试不通过下次启动还是进

pcba 测试。

Pack loader, parameter, pcba_whole_misc.img, boot, recovery, system, etc. into update.img, and that is the whole image with pcba testing function. Flash it into the device using MP tool, and it will enter pcba test for the first boot up. If test passed, it will enter android for next boot up, while it will still enter pcba test for next boot up if test failed.

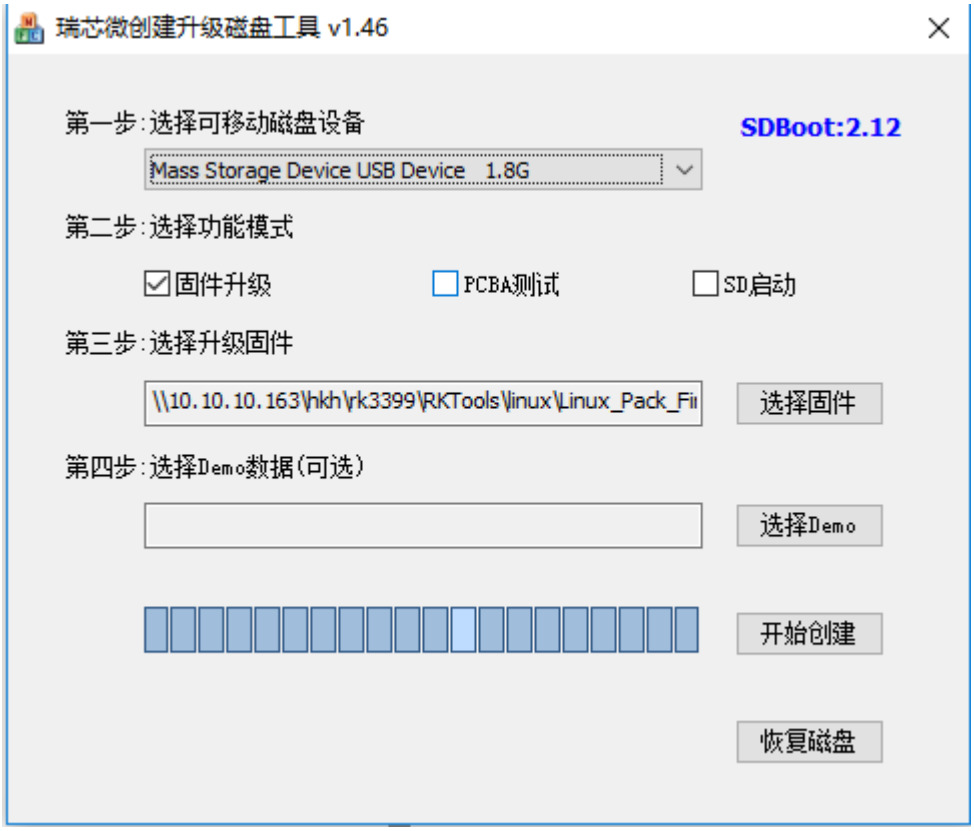
3. 如果不需要第一次启动进入 pcba 测试，可以不打包 pcba_whole_misc.img，而使用 普通的 misc，这样量产工具升级完第一次启动只做格式化操作然后正常启动 android。

If no need to enter pcba test for the first boot up, don't pack pcba_whole_misc.img, just use normal misc, in this way it will only do formatting and then normally boot up android for the first boot up after upgrade by MP tool.

7.6 制作 SD 升级卡 Make SD upgrade card

使用 SD_Firmware_Tool 创建一张 sd 升级卡，具体使用方法参考工具目录下的 《SD Card BootUserGuide》文档。

Use SD_Firmware_Tool to create a sd upgrade card. For the specific method, please refer to the document of 《SD Card BootUserGuide》 under tool directory.



1) 如果贴片厂不需要升级完整固件，只需要做 PCBA 测试，有以下两种方式：

If no need to upgrade the whole image in SMT, only need to do PCBA test, there are two methods as below:

a) 只勾选 PCBA 测试

Only select PCBA test

b) 只勾选 SD 启动

Only select SD boot up

注意：选择此 SD 启动时，需要将 kernel 配置中的（ Device Drivers -> MMC/SD/SDIO card support -> RK29 SDMMC0 controller support 关掉，不然会造成无法从 SD 卡启动）

Note: when select SD boot up, need to (disable Device Drivers -> MMC/SD/SDIO card support -> RK29 SDMMC0 controller support, otherwise it will cause failure to boot up from SD card) in kernel configuration.

使用以上两种方法都可以实现：机器直接插卡进行 PCBA 测试，由于方法 a)需要先将固件

升级到 flash 中才进行测试，所需要的时间比方法 b)大概要长 10 秒左右，所以建议使用方法 b)进行 PCBA 测试，方法 b)无法在 PCBA 中进行 SD 卡测试，但是由于是 SD 卡启动，可以证明 SD 卡功能是正常的，所以就不需要测试 SD 卡。

Use both methods above can realize: directly insert the card in the device to do PCBA test, because method a) needs to upgrade the image to flash before testing, it will take 10s more than method b), it is recommended to use method b) to do PCBA test. It is not able to do SD card test in PCBA for method b), but it is booted up by SD card, which indicating SD card function is normal, so there is no need to test SD card.

2) 整机厂商需要升级完整固件并拷贝 demo 数据文件，制作的时候应该选择完整固件，并只勾选固件升级，不勾选 pcba 测试。

The device manufacturer needs to upgrade the whole image and copy demo data file, need to select the whole image and only select image upgrade, don’t select pcba test.

7.7 SD 升级卡的使用 SD upgrade card usage

1. 创建好 sd 卡后，对于支持 sd 卡引导的芯片，只需要将卡插入机器，重新开机即可。

After sd card is created, for the chipset supporting to load through sd card, only need to insert the card into the device and then power up.

2. 对于不支持 sd 卡引导的芯片，如 3066，需要在机器烧过 pcba 小固件的情况下，插卡开机才能使用，也就是上文说的第三种量产升级流程。

For the chipset not supporting to load through sd card, such as RK3066, need to flash pcba small image to the device first, and then insert the cart and power up, that is the third process of MP upgrade as described in previous chapter.

7.8 注意事项 Notices

sd 升级卡不是普通卡，是带有引导信息的卡，支持 sd 卡引导的机器插着卡开机就会从卡启

动，不要当成普通卡拷入 update.img 在 android 状态下去做本地升级。如果需要用卡去本地升级需要先用 SD_Firmware_Tool 工具恢复这张卡。

Sd upgrade card is not normal card, it is the card with load information, the device supporting to load through sd card will boot up from card if it is powered up with card inserted. Do not use it as normal card to copy update.img to do local upgrade in android state. If need to use the card to do local upgrade, need to use SD_Firmware_Tool to restore the card first.

8 常见问题处理 FAQ

8.1 差分升级错误处理 Incremental OTA error handling

差分升级容易出错，原因是生成的烧到机器里的 image 固件与 make otapackage 编译出来的完整包不一致，包括人为的对 image 的修改或是对完整包的修改，以下总结了几种容易造成差异升级失败的不当操作：

Incremental OTA is easy to fail, because the image flashed to the device is different from the whole package compiled by make otapackage, including the manual modification to image or the whole package. Below summarize several incorrect operations which will cause the differential upgrade failure easily:

1. 发布固件未使用./mkimage.sh ota 生成，量产时单独烧写 kernel，原因是 ota 包中的 boot.img 是有带 kernel 的，所以生成的差分补丁都是基于带 kernel 的 boot，在需要差分升级到下一个版本时，boot.img 的补丁会打不上。

   Not use "./mkimage.sh ota" to generate the release image, separately flash kernel during MP, because boot.img in ota package already includes kernel, the generated differential patch is based on boot with kernel, when need differential upgrade to next version, boot.img patch cannot be applied.

2. 未通过编译系统直接修改 out 目录下的文件后再生成 ota package。

   Not directly modify the file in out directory through compiling system and then

generate ota package.

3. 使用固件工厂工具直接修改 update.img。

   Use image factory tool to directly modify update.img.

4. build/tools/release/security/目录有 publicKey.bin 和 privateKey.bin 这两个文件，update.img 却未使用 secureboot 工具签名。原因是源码中包含 privatekey.bin 时 make otapackage 生成完整包时会对 boot.img 进行 drm 签名，但是量产 update.img 却没有对 boot.img 进行签名，造成两个固件不一致。

   build/tools/release/security/ directory has publicKey.bin and privateKey.bin files, update.img is not signed by secureboot tool. The reason is that when the source code includes privatekey.bin, make otapackage to generate the whole package will perform drm signature to boot.img, but during MP update.img doesn't sign to boot.img, which makes the two images inconsistent.

5. 发布每个版本固件未保存对应的 target 素材包，差异包是用 out/target/product/rk3188/obj/PACKAGING/target_files_intermediates/目录下的素材包制作的，如果发布的每一版固件没有保存对应的素材包将无法生成可用的差异包。

   The corresponding target material package is not saved for every release image, the incremental package is made from the material package in the directory of out/target/product/rk3188/obj/PACKAGING/target_files_intermediates/. The incremental package cannot be generated if the corresponding material package of each release image is not saved.

8.2 升级包签名校验错误处理 Upgrade package signature verification error handling

保证升级前后使用同一套签名密钥，即工程源码下 build/target/product/security/保持一致，比如 4.2 要升级到 4.4，要将 4.2 的该目录下的 key 覆盖到 4.4 下，再生成 ota 升级包。

Ensure to use the same set of signature private key before and after upgrade, that

is, build/target/product/security/ of the project source code should keep consistent, such as upgrade from 4.2 to 4.4, need to overwrite the key in this directory from 4.2 to 4.4, and then generate ota upgrade package.

8.3 NTFS 格式 U 盘升级支持 NTFS format U disk upgrade support

内核默认不支持 NTFS 格式 U 盘的挂载，所以 recovery 中无法支持。如需支持 NTFS，需要打开 kernel 中的相应配置，recovery 默认已支持：

Kernel doesn't support NTFS format U disk load by default, so it is not able to support in recovery. If need to support NTFS, need to enable the corresponding configuration in kernel, which recovery already support by default:

```
Symbol: NTFS_FS [=n]
Type  : tristate
Prompt: NTFS file system support
  Location:
    -> File systems
(3)   -> DOS/FAT/NT Filesystems
  Defined at fs/ntfs/Kconfig:1
  Depends on: BLOCK [=y]
  Selects: NLS [=y]
```

8.4 Logo 未更新成功 Logo update failure

1. 位置：Logo 包含 uboot logo 和 kernel logo，这两个 Logo 都存储于 resource.img 里面。

Location: Logo includes uboot logo and kernel logo, both Logo are saved in resource.img.

2. 显示策略：分为两种情况

Display strategy: there are two cases

boot.img 包含/不包含 resource.img

boot.img with/without resource.img

策略一：如果 parameter 里面有声明 resource 分区

Strategy one: if there is resource partition statement in parameter

优先从 resource 分区读取 Logo

Prefer to read Logo from resource partition

策略二：如果 parameter 里面没有声明 resource 分区

Strategy two: if there is no resource partition statement in parameter

优先从 boot 里面读取 Logo

Prefer to read Logo from boot

由于 boot.img 较大，读取需要时间，为了较快的显示 Logo，我们采用策略一，所以如果出现 Logo 升级失败，应该优先检查 resource 是有升级成功。

Because boot.img is relatively large, reading takes time, in order to display Logo quickly, we use strategy one, if fail to upgrade Logo, first need to check whether resource is upgraded successfully.

3. 总结：所以 Logo 如果未更新成功，是 resource.img 升级失败，可以参考 recourse.img 升级。

Summary: if fail to upgrade Logo, it is caused by resource.img upgrade failure, and you can refer to recourse.img upgrade.

9 在线升级服务器 On-line upgrade server

9.1 服务器运行环境 Server running environment

Ota 服务器建议运行在 ubuntu 系统，文档以 ubuntu 系统下搭建方法为例进行说明。

It is recommended to run OTA server in ubuntun system. The document describes the setup method in Ubuntu system as an example.

9.2 Ubuntu 连接数优化设置 Ubuntu connection number optimization setting

由于 Ubuntu 默认的进程最大开启文件句柄数为 1024（ulimit–n 命令查看），限制了服务器所能打开的 socket 连接最大值，无法发挥服务器的性能，可以通过以下步骤设置：

Because the maximum number of open file handle of Ubuntu default process is 1024 (use ulimit–n command to check), which limits the maximum socket connection opened by the server, the performance of the server is limited. You can set as the following steps:

1）.打开/etc/security/limits.conf，里面有很详细的注释，找到如下设置(如果没有就插入)

Open /etc/security/limits.conf, there are detailed notes, find the following settings (add if not existing):

* soft nofile 65535

* hard nofile 65535

2）.编辑/etc/pam.d/common-session，加入一行

Edit /etc/pam.d/common-session, add a line

session required pam_limits.so

3）.修改后重启 ubuntu 即可。

Reboot Ubuntu after modification.

9.3 JDK 安装 JDK installation

Ota 服务器要求 JDK1.6 以上版本，具体如何安装配置可以参考网上，这里不做详细说明。

OTA server requires JDK1.6 above version. For the specific install configuration, you can refer to web information. We don’t elaborate here.

9.4 服务器配置 Server configuration

1. 解压 apache-tomcat-7.0.29.zip 到任意目录下

Unzip apache-tomcat-7.0.29.zip to any directory

如：mmk@hp-PC:~/webdevelop/apache-tomcat-7.0.29

Such as: mmk@hp-PC:~/webdevelop/apache-tomcat-7.0.29

2. 修改整个目录的权限

Modify the authority of the whole directory

如：mmk@hp-PC:~/webdevelop$chmod 775 -R apache-tomcat-7.0.29

Such as: mmk@hp-PC:~/webdevelop$chmod 775 -R apache-tomcat-7.0.29

3. 服务器应用部署在 webapps/OtaUpdater

Deploy the server application in webapps/OtaUpdater

mmk@hp-PC:~/webdevelop/apache-tomcat-7.0.29/webapps/OtaUpdater/WEB

- INF$ls classes lib log4j.properties manifast.xml packages web.xml

manifest.xml 和 packages 目录需要根据产品型号和版本号，手动进行配置

manifest.xml and packages directories need to be manually configured according to the product type and version number.

4. manifast.xml 配置文件写法说明

manifast.xml configuration file instruction

```
这里以新加入一个产品 TD8801 为例进行说明
<product name="TD8801" full_package_path="null" rkimage_path="null">
    <version name="1.0.0" package_path="packages/ TD8801/1.0.0/1.0.2.zip" />
    <version name="1.0.1" package_path="packages/ TD8801/1.0.1/1.0.2.zip" />
    <version name="1.0.2" package_path="packages/ TD8801/1.0.2/1.0.3.zip" />
</product>
```

product 标签定义了产品属性，产品名称为 TD8801， full_package_path 指的是 ota 最新完整包路径，如果没有就设为 null 一个产品下有多个版本通过 version 标签来定义，name 为版本号（与固件编译时设置的版本号一一对应），package_path 为该版本对应的升级包路径（相对主工作目录）。

Product label defines the product attribute, product name is TD8801, full_package_path means the latest OTA whole package path, set it as null if none.

One product with multiple versions is defined through version label, name is the version number (corresponding to the version number set during image compilation), package_path is the upgrade package path corresponding to this version (relative to the main working directory)

5. 每个产品各个版本对应的升级包可以放在 packages 下任意位置，只要 manifast.xml 指定好相对路径即可。

Each upgrade package corresponding to the versions of the product can be put in any location of packages, only need to specify the relative path for manifast.xml.

9.5 服务器监听端口修改 Server monitor port modification

默认监听端口 2300，如果用户需要修改，按如下步骤：

The default monitor port is 2300, which can be modified by users according to the following steps:

mmk@hp-PC:~/webdevelop/apache-tomcat-7.0.29$vimconf/server.xml

找到如下内容：

Find below content:

```
<Connector port="2300" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />
```

将 2300 改为您需要的端口号就可以了。

Just change 2300 to the port number you need.

9.6 服务器运行和停止 Server running and stop

1. 服务器包部署在具有外网能访问 ip 地址的主机上，需要向网络运营商申请域名和 IP。

Server package is deployed on the host which can be accessed through IP address by the external network. Need to apply for domain name and IP from network carrier.

2. 配置完 manifast.xml，并将升级包都按规则放好后，就可以运行服务器了。

After manifast.xml is configured and the upgrade package is placed according to the rule, it is able to run the server.

3. startup.sh 运行服务器。

Execute startup.sh to run the server.

4. shutdown.sh 停止服务器。

Execute shutdown.sh to stop the server.

9.7 服务器运行日志 Server running log

运行日志保存在 apache-tomcat-7.0.29/mylogs 目录下，每天会产生一个日志文件，方便 Debug。

The running log is saved in the directory of apache-tomcat-7.0.29/mylogs, and every day it will generate a log file for Debug.

9.8 编译注意事项 Notices of compilation

Device/rockchip/rk*****/rk****.mk

```
PRODUCT_NAME := rk29sdk
PRODUCT_DEVICE := rk29sdk
PRODUCT_BRAND := Android
PRODUCT_MODEL := TD8801
PRODUCT_CHARACTERISTICS := tablet

PRODUCT_PROPERTY_OVERRIDES += \
        ro.product.version = 1.0.0 \
        ro.product.ota.host = www.rockchip.com:2300
```

1. 定义好产品型号：修改 PRODUCT_MODEL 值为相应的产品型号，如 TD8801，A22，TSB 等。注意产品型号不能带空格。

Define the product model: modify PRODUCT_MODEL value as the corresponding

product model, such as TD8801, A22, TSB and so on. It is noted that product model cannot include space.

2. 定义当前编译的固件版本号（重要） ro.product.version 属性值 1.0.0 即为当前系统版本号，这个版本号与服务器的 manifast.xml 文件中定义的必须一致，以后每发布一个新版本固件都切记更新版本号，如果不更新或是错误的版本号，将使服务器无法工作或是推送了错误的升级包， 将造成不可预计的后果。

Define currently compiled image version number (important) ro.product.version attribute value 1.0.0 which is current system version number, this version number must be consistent with the one defined in manifast.xml file of the server, remember to update the version number when each new version image is released, if the version number is not updated or wrong, it will make the server fail to work or push wrong upgrade package, which will cause unpredictable result.

需要注意的是修改过这个属性要将 out/target/product/rk29sdk/system/build.prop 删除再 make 改变才能生效，或者 makeclean 后再 make。

Need to notice that, to make this attribute modification valid, you need to delete out/target/product/rk29sdk/system/build.prop and then make, or makeclean and then make.

3. 定义 ota 服务器地址：ro.product.ota.host 属性定义了 ota 服务器主机地址，这个域名需要向网络运营商申请。2300 指的是端口号，对应服务器监听的端口。

Define OTA server address: ro.product.ota.host attribute defines the host address of OTA server, need to apply for this domain name from network carrier. 2300 means the port number, corresponding to the monitor port of server.

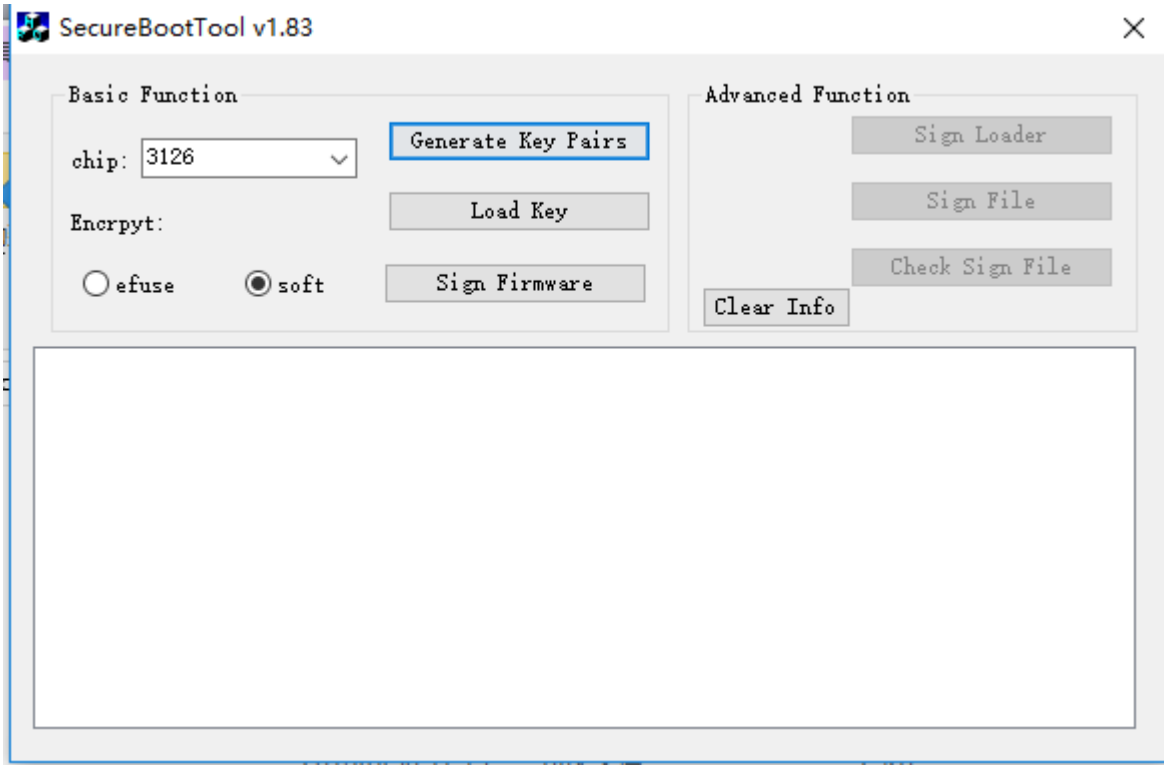
10 SecureBoot 签名工具 SecureBoot signature tool

本节只对工具的使用进行简单的介绍，更为详细的请参考工具目录下的文档说明。

This chapter only simply introduces the tool usage. For more details, please refer to the document in the tool directory.

《RockchipSecureBootApplicationNote》

10.1 win 平台签名 win platform signature

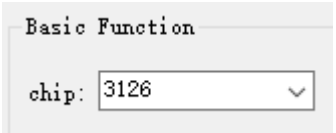


如上图是 win 平台签名工具的界面，以下对签名工具做详细的介绍。

Above is the interface of the signature tool on win platform. The detailed instruction of the signature tool is shown as below.

配置说明：

Configuration instruction:



：选择芯片型号，3188 之前的芯片选择 others。

Select the chipset. Select others for the chipset before 3188.

：配置加密类型是写 efuse 加密还是软件加密（注：3128，3288,3228,3368 支持 efuse 级加密，其他的旧芯片不支持）。

Configure the encryption type as effuse or software encryption (note: 3128, 3288,

3228, 3368 support effuse encryption, while other old chipsets don’t support).

Generate Key Pairs：生成 RSAKEY，每款机器只能生成一次，请一定备份好 KEY，如果丢失，那么机器将不能再更新固件。

Generate RSAKEY, each device can only generate once, please back up KEY, if it is lost, the device cannot update image any more.

Load Key：加载之前备份的 RSAKEY（工具支持 openssl 生成的.pem 的 2048key）。

Load previously backup RSAKEY (the tool supports 2048key with .pem generated by openssl).

Sign Firmware：进行签名，签名的固件必须使用./mkimage ota 生成。

Sign the firmware, which must be generated using ./mkimage ota.

以上步骤完成后，签名成功会有如下弹框：

After the above steps finish, it will prompt below dialog if sign successfully:

10.2 Linux 平台签名 Linux platform signature

```
projects@bogon:~/hkh_test/rk3228-4.4/SignOtaPackageRelease$ ./SecureBootConsole
SecureBootConsole v1.83
**********Usage**********
SecureBootConsole -k|-kk SaveKeyDir //generate key -k(1024) -kk(2048)
SecureBootConsole -si privatekey image//Sign Image
SecureBootConsole -sl publickey loader //Sign Loader
SecureBootConsole -slx privatekey publickey loader [LE|BE]//SignEx Loader
SecureBootConsole -sh privateKey firmware //Sign Firmware's digest
SecureBootConsole -sf privateKey file //Sign file
SecureBootConsole -gh file [160|256|256LE]//Generate hash
SecureBootConsole -sz privateKey otazipfile //Sign ota zip file
********** End **********
```

Linux 平台签名工具，如上图，运行不带参数的 SecureBootConsole 会有 Usage，这里不再做重复说明。

Linux platform signature tool is shown as above, run SecureBootConsole without parameter will show Usage, so we don't repeat here.

11 Block 升级方式 Block upgrade

11.1 配置文件 Configuration file

1. 编辑 build/core/Makefile，增加如下一行

Edit build/core/Makefile, add the line as below:

```
diff --git a/core/Makefile b/core/Makefile
index cce2647..db61c25 100644
--- a/core/Makefile
+++ b/core/Makefile
@@ -1667,6 +1667,7 @@ $(INTERNAL_OTA_PACKAGE_TARGET): $(BUILT_TARGET_FILES_PACKAGE) $(DISTTOOLS)
        @echo "Package OTA: $@"
        $(hide) MKBOOTIMG=$(MKBOOTIMG) \
           ./build/tools/releasetools/ota_from_target_files -v \
+          --block \
           -p $(HOST_OUT) \
           -k $(KEY_CERT_PAIR) \
           $(if $(OEM_OTA_CONFIG), -o $(OEM_OTA_CONFIG)) \
```

2. 编辑 device/rockchip/common/device.mk

Edit device/rockchip/common/device.mk

```
hkh@RD-DEP1-SERVER-163:~/rk3368/5.1/device/rockchip/common$ git diff
diff --git a/device.mk b/device.mk
index 03eeb4c..291be88 100755
--- a/device.mk
+++ b/device.mk
@@ -505,8 +505,8 @@ endif

 # setup dm-verity configs.
 # uncomment the two lines if use verity
-#PRODUCT_SYSTEM_VERITY_PARTITION := /dev/block/platform/ff0f0000.rksdmmc/by-name/system
-#$(call inherit-product, build/target/product/verity.mk)
+PRODUCT_SYSTEM_VERITY_PARTITION := /dev/block/platform/ff0f0000.rksdmmc/by-name/system
+$(call inherit-product, build/target/product/verity.mk)

 ifeq ($(strip $(BUILD_WITH_GOOGLE_MARKET)), true)
 ifeq ($(strip $(BUILD_WITH_GOOGLE_MARKET_ALL)), true)
```

注意：打开的同时请确保 system 分区的路径正确，不同芯片这个路径是不一样的。

Note: Please ensure the system partition path is correct. This path is different for different chipset.

3. 编辑 device/rockchip/rkxxxx/fstab.rk30board.bootmode.emmc，切换 system 的配置到 verify

Edit device/rockchip/rkxxxx/fstab.rk30board.bootmode.emmc, switch system configuration to verify.

```
diff --git a/fstab.rk30board.bootmode.emmc b/fstab.rk30board.bootmode.emmc
index 2e49d90..a38b119 100644
--- a/fstab.rk30board.bootmode.emmc
+++ b/fstab.rk30board.bootmode.emmc
@@ -3,9 +3,9 @@
 # The filesystem that contains the filesystem checker binary (typically /system) cannot
 # specify MF_CHECK, and must come before any filesystems that do specify MF_CHECK

-/dev/block/platform/ff0f0000.rksdmmc/by-name/system        /system             ext4      ro,noatime,nodiratime,noauto_da_alloc
+#/dev/block/platform/ff0f0000.rksdmmc/by-name/system        /system             ext4      ro,noatime,nodiratime,noauto_da_alloc
 # use this line below instead to enable verity
-#/dev/block/platform/ff0f0000.rksdmmc/by-name/system        /system             ext4      ro,noatime,nodiratime,noauto_da_alloc
+/dev/block/platform/ff0f0000.rksdmmc/by-name/system        /system             ext4      ro,noatime,nodiratime,noauto_da_alloc
 /dev/block/platform/ff0f0000.rksdmmc/by-name/cache         /cache              ext4      noatime,nodiratime,nosuid,nodev,noauto_da_alloc,discard
 /dev/block/platform/ff0f0000.rksdmmc/by-name/metadata      /metadata           ext4      noatime,nodiratime,nosuid,nodev,noauto_da_alloc,discard
 /dev/block/platform/ff0f0000.rksdmmc/by-name/userdata      /data               f2fs      noatime,nodiratime,nosuid,nodev    wait,check,encryptable=/met
```

4. 编辑 device/rockchip/rkxxxx/BoardConfig.mk，按你的需求修改 system 分区大小（请考虑后续 OTA 的需求，且与 parameter 中的值一致）

Edit device/rockchip/rkxxxx/BoardConfig.mk, modify the system partition size according to your requirement (please consider future OTA requirement, and should be consistent with the value in parameter).

```
//MAX-SIZE=512M, for generate out/.../system.img
BOARD_SYSTEMIMAGE_PARTITION_SIZE ?= 1073741824
BOARD_FLASH_BLOCK_SIZE ?= 131072
```

或者路径 device/rockchip/common/BoardConfig.mk

or the path device/rockchip/common/BoardConfig.mk

5. 拷贝打包脚本到源码根目录 cp device/rockchip/common/mkimage_verity.sh ./

Copy the package script to the source code root directory cp device/rockchip/common/mkimage_verity.sh ./

6. 按正常编译流程编译，最后一步打包的时候把 mkimage.sh ota 替换成 mkimage_verity.sh ota

Compile according to the normal compilation process, replace mkimage.sh ota with mkimage_verity.sh ota at the last step of packing.

7. 差异包生成也需要增加--block 选项

Also need to add --block option for differential package generation.

```
6 ./build/tools/releasetools/ota_from_target_files --block -v -i $source_zip -p out/host/linux-x86 -k build/target/product/security/testkey
  $target_zip $target_name
```