Rockchip_Developer_Guide_Android_AB_System_Upgrading_CN

文件标识：RK-SM-YF-208

发布版本：V1.1.0

日期：2023-02-06

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

本文档描述了Rockchip A/B系统的使用说明。

产品版本

芯片名称	内核版本
RK3326/RK3399/RK356x/RK3588	4.19, 5.10

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

修订记录

版本号	作者	修改日期	修改说明
V1.0.0	纪大峣	2022-02-25	初始版本
V1.1.0	纪大峣	2023-02-06	补充Android 13配置说明

目录

1. 概述

本文档描述了Rockchip A/B系统的使用说明，可以在Rockchip Android >= 12平台上使用。
使用该升级方法，所有待升级的分区（除super之外）都有a和b两份，相比传统的Recovery升级方式会大幅增加存储空间需求。其优点是A/B升级是无缝升级，可以在Android系统运行过程中，根据客户定制的客户端和服务器之间的交互策略与协议来完成升级。

2. 系统配置说明

在Rockchip Android 平台上，AB系统功能默认关闭，要使用AB系统，需要从Android系统、U-BOOT和kernel dts三个方面进行配置。以下对此进行详细说明。

2.1 Android系统配置说明

2.1.1 Android 12系统配置

Android 12系统的配置包括：

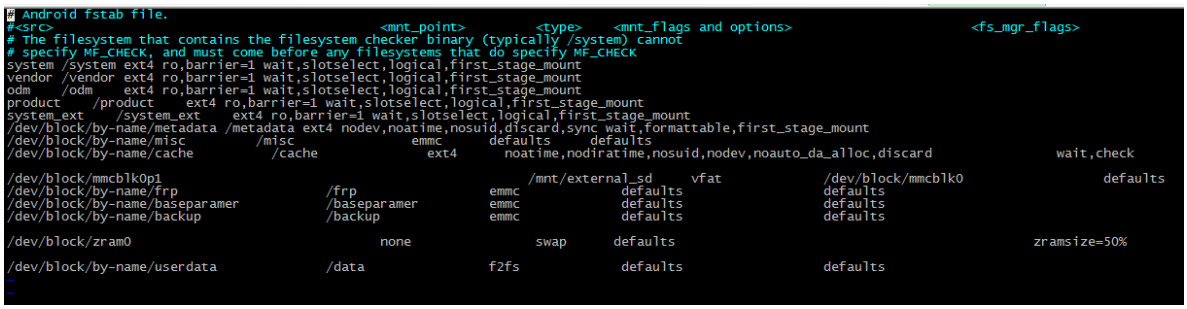
1.在device\rockchip\rkxxx\BoardConfig.mk中打开AB升级配置项。
将BOARD_USES_AB_IMAGE配置设置为true，默认为false.
以rk3326为例：

```
vim device/rockchip/rk3326_s/BoardConfig.mk
#AB image definition
-BOARD_USES_AB_IMAGE := false
+BOARD_USES_AB_IMAGE := true
```

注意：如果要开启虚拟AB功能（不建议），则需要同时配置BOARD_USES_AB_IMAGE 和BOARD_ROCKCHIP_VIRTUAL_AB_ENABLE为true，即：

```
-BOARD_USES_AB_IMAGE := false
-BOARD_ROCKCHIP_VIRTUAL_AB_ENABLE := false
+BOARD_USES_AB_IMAGE := true
+BOARD_ROCKCHIP_VIRTUAL_AB_ENABLE := true
```

2.在对应的device\rockchip\rkxxx目录下，确认是否已经有recovery.fstab_AB文件，如果已有，则直接跳过这一步；如果没有recovery.fstab_AB文件，则按如下步骤执行：
（1）新增针对AB的recovery fstab文件recovery.fstab_AB
与对应的fstab.rk30board的主要区别在于AB分区增加slotselect挂载参数（system/vendor/odm/product增加slotselect参数），同时将exernal_sd, frp, parameter, baseparameter, resource的分区节点添加进去，并更改data区的挂载方式选项。
一个参考文件如下：
device\rockchip\rk3399\rk3399_Android12\recovery.fstab_AB：



（2）device\rockchip\rkxxx下的BoardConfig中导入AB配置，包括TARGET_RECOVERY_FSTAB，使其指向刚刚创建的recovery.fstab_AB文件。

```
device\rockchip\rk3399\rk3399_Android12\BoardConfig.mk:
diff --git a/ rk3399/rk3399_Android12/BoardConfig.mk b/
rk3399/rk3399_Android12/BoardConfig.mk
index 1e78940..159a3b6 100755
--- a/ rk3399/rk3399_Android12/BoardConfig.mk
+++ b/ rk3399/rk3399_Android12/BoardConfig.mk
+# AB image definition
+BOARD_USES_AB_IMAGE := true
+BOARD_ROCKCHIP_VIRTUAL_AB_ENABLE := false
+
+ifeq ($(strip $(BOARD_USES_AB_IMAGE)), true)
+    include device/rockchip/common/BoardConfig_AB.mk
+    TARGET_RECOVERY_FSTAB := device/rockchip/
rk3399/rk3399_Android11/recovery.fstab_AB
+endif
```

2.1.2 Android 13系统配置

Android 13系统的配置包括：

1.在device\rockchip\rkxxx\BoardConfig.mk中打开AB升级配置项。
将BOARD_USES_AB_IMAGE配置设置为true，默认为false.
以rk3326为例：

vim device/rockchip/rk3326_s/BoardConfig.mk
#AB image definition
-BOARD_USES_AB_IMAGE := false
+BOARD_USES_AB_IMAGE := true

注意：

（1）如果要开启虚拟AB功能（不建议，对升级性能有影响），则需要同时配置BOARD_USES_AB_IMAGE 和BOARD_ROCKCHIP_VIRTUAL_AB_ENABLE为true，即：
-BOARD_USES_AB_IMAGE := false
-BOARD_ROCKCHIP_VIRTUAL_AB_ENABLE := false
+BOARD_USES_AB_IMAGE := true
+BOARD_ROCKCHIP_VIRTUAL_AB_ENABLE := true

（2）从Android 13开始支持压缩的虚拟AB功能，如果要开启压缩的虚拟AB功能（不建议，对升级性能有影响），则需要同时配置BOARD_USES_AB_IMAGE 、BOARD_ROCKCHIP_VIRTUAL_AB_ENABLE和BOARD_ROCKCHIP_VIRTUAL_AB_COMPRESSION为true，即：

-BOARD_USES_AB_IMAGE := false
-BOARD_ROCKCHIP_VIRTUAL_AB_ENABLE := false
+BOARD_USES_AB_IMAGE := true
+BOARD_ROCKCHIP_VIRTUAL_AB_ENABLE := true
+BOARD_ROCKCHIP_VIRTUAL_AB_COMPRESSION := false

2.在对应的device\rockchip\rkxxx目录下，确认是否已经有recovery.fstab_AB文件，如果已有，则直接跳过这一步；如果没有recovery.fstab_AB文件，则按如下步骤执行：

（1）新增针对AB的recovery fstab文件recovery.fstab_AB

与对应的fstab.rk30board的主要区别在于AB分区增加slotselect挂载参数（system/vendor/odm/product增加slotselect参数），同时将exernal_sd, frp, parameter, baseparameter, resource的分区节点添加进去，并更改data区的挂载方式选项。

一个参考文件如下：

device\rockchip\rk3399\rk3399_Android12\recovery.fstab_AB：

```
# Android fstab file.
#<src>                                  <mnt_point>         <type>    <mnt_flags and options>                       <fs_mgr_flags>
# 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
system /system ext4 ro,barrier=1 wait,slotselect,logical,first_stage_mount
vendor /vendor ext4 ro,barrier=1 wait,slotselect,logical,first_stage_mount
odm    /odm    ext4 ro,barrier=1 wait,slotselect,logical,first_stage_mount
product    /product    ext4 ro,barrier=1 wait,slotselect,logical,first_stage_mount
system_ext    /system_ext    ext4 ro,barrier=1 wait,slotselect,logical,first_stage_mount
/dev/block/by-name/metadata /metadata ext4 nodev,noatime,nosuid,discard,sync wait,formattable,first_stage_mount
/dev/block/by-name/misc         /misc             emmc     defaults       defaults
/dev/block/by-name/cache        /cache            ext4     noatime,nodiratime,nosuid,nodev,noauto_da_alloc,discard               wait,check

/dev/block/mmcblk0p1                                        /mnt/external_sd     vfat            /dev/block/mmcblk0                 defaults
/dev/block/by-name/frp                  /frp                emmc            defaults                defaults
/dev/block/by-name/baseparamer          /baseparamer        emmc            defaults                defaults
/dev/block/by-name/backup               /backup             emmc            defaults                defaults

/dev/block/zram0                              none              swap      defaults                                                zramsize=50%

/dev/block/by-name/userdata             /data               f2fs            defaults                defaults
~
~
```

（2）device\rockchip\rkxxx下的BoardConfig中导入AB配置，包括TARGET_RECOVERY_FSTAB，使其指向刚刚创建的recovery.fstab_AB文件。

```
device\rockchip\rk3399\rk3399_Android12\BoardConfig.mk:
diff --git a/ rk3399/rk3399_Android12/BoardConfig.mk b/
rk3399/rk3399_Android12/BoardConfig.mk
index 1e78940..159a3b6 100755
--- a/ rk3399/rk3399_Android12/BoardConfig.mk
+++ b/ rk3399/rk3399_Android12/BoardConfig.mk
+# AB image definition
+BOARD_USES_AB_IMAGE := true
+BOARD_ROCKCHIP_VIRTUAL_AB_ENABLE := false
+
+ifeq ($(strip $(BOARD_USES_AB_IMAGE)), true)
+    include device/rockchip/common/BoardConfig_AB.mk
+    TARGET_RECOVERY_FSTAB := device/rockchip/
rk3399/rk3399_Android11/recovery.fstab_AB
+endif
```

2.2 kernel dts配置

对于Android >=12来说，kernel不需要任何配置。

2.3 uboot配置

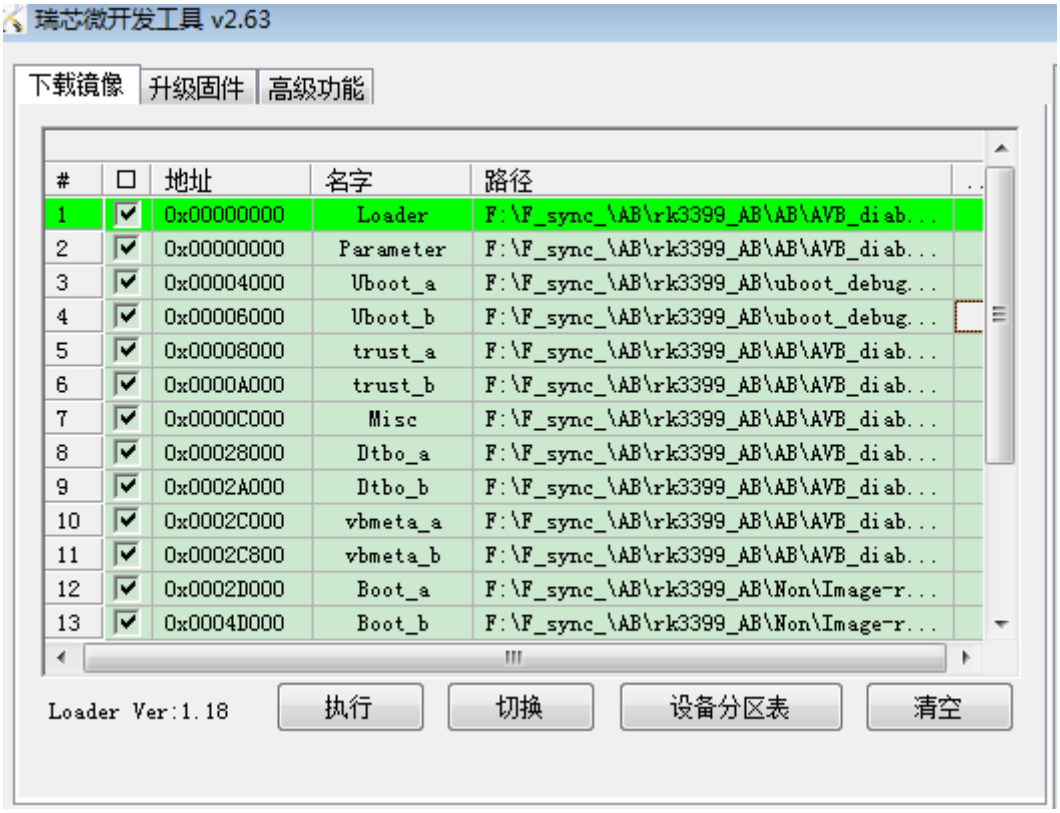
在uboot中，针对具体芯片的配置文件，添加CONFIG_ANDROID_AB=y配置项，参考配置如下截图所示：

```
diff --git a/configs/rk3326_defconfig b/configs/rk3326_defconfig
old mode 100644
new mode 100755
index 0465f23..d64648c
--- a/configs/rk3326_defconfig
+++ b/configs/rk3326_defconfig
@@ -114,3 +114,4 @@ CONFIG_OPTEE_CLIENT=y
 CONFIG_OPTEE_V2=y
 CONFIG_OPTEE_ALWAYS_USE_SECURITY_PARTITION=y
 CONFIG_TEST_ROCKCHIP=y
+CONFIG_ANDROID_AB=y
```

2.4 开发烧写工具增加分区

下载工具增加B分区的下载项，同时所有AB分区都烧写相同的固件。
对于>=Android 12来说，根据“3.AB系统与OTA包编译”指令完成编译后，在rockdev/xxxx/下自动产生config.cfg文件，将该配置文件替换RKDevTool工具下的同名文件，然后再重新打开RKDevTool即可，不需要手动配置。

一个烧写工具截图如下：



注意，对于rk3566/rk3568/rk3588等芯片来说，没有trust分区，则不需要trust_a和trust_b分区。

3. AB系统与OTA包编译

编译AB系统固件步骤（务必按如下步骤执行）：
lunch xxx
make installclean –j16 OR make clean -j16
make –j16
make dist –j16
mkimage_ab.sh ota

注意：
1.开启AB后，第一次需要make clean后再编译。
2.差异包的编译方法与非AB一致。

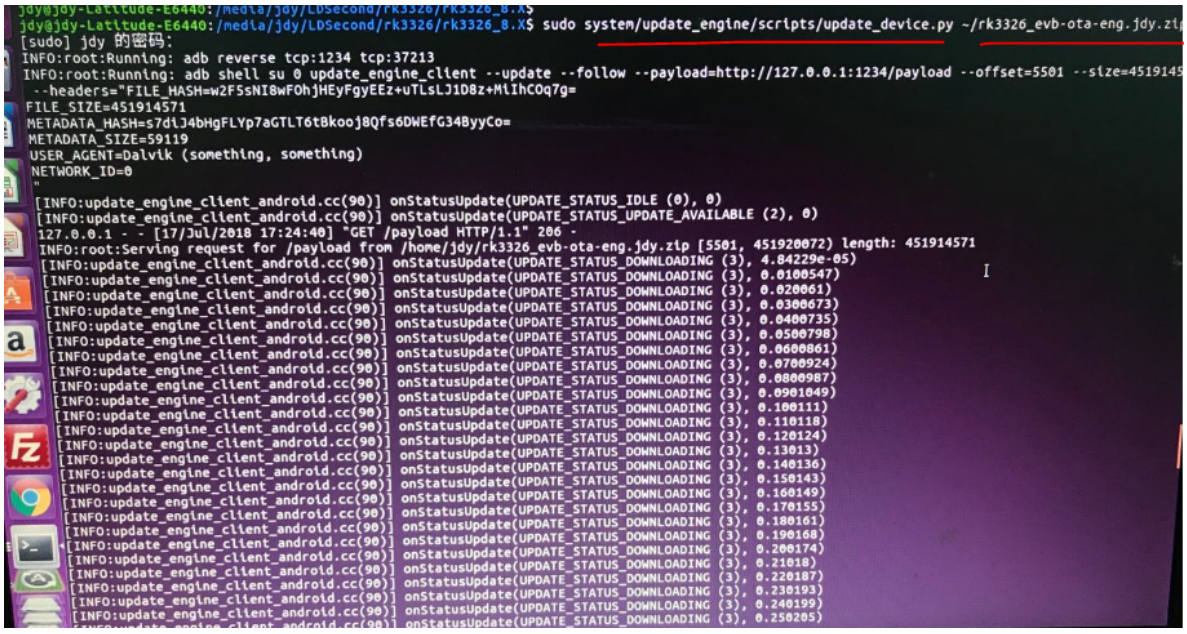
4. 验证方法（客户端与服务器）

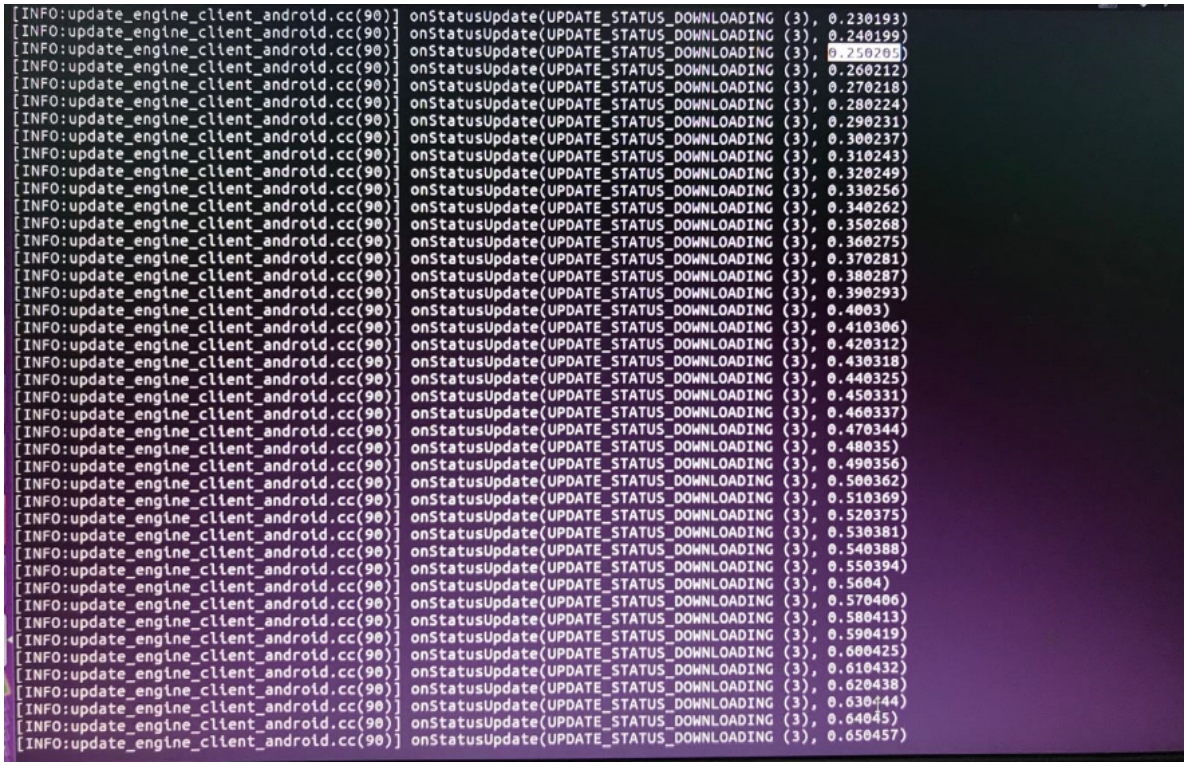
在无缝升级过程中，升级包可以一边下载，一边升级。这时候需要有一个HTTP服务器和一个升级客户端。
请使用以下update_device.py验证。至于产品化的升级客户端和升级服务器需要客户自行搭建，升级客户端可以参考Android默认提供的update_engine_client。关于update_engine_client的使用方法请参考如下验证方法update_device.py，该脚本就是通过adb最终调用update_engine_client来实现升级的。
验证方法：update_device.py
update_device.py脚本通过adb方式，将主机变成HTTP服务器，然后调用update_engine_client来实现无缝升级。
使用方法如下：

在主机中执行如下命令（请确保该主机adb功能和python环境正常）：

```
system/update_engine/scripts/update_device.py {升级包名字}
```

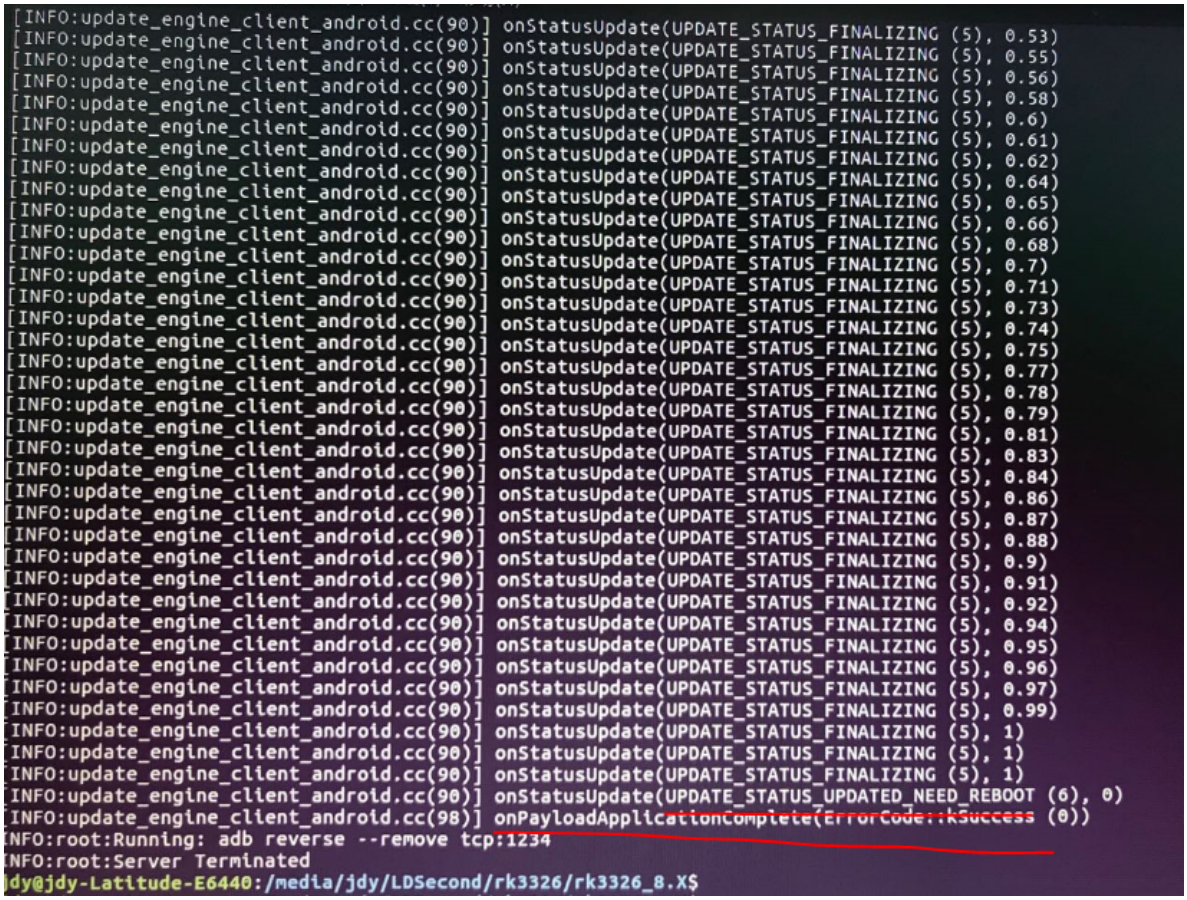
示例如下：





```
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.230193)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.240199)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.250205)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.260212)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.270218)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.280224)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.290231)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.300237)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.310243)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.320249)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.330256)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.340262)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.350268)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.360275)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.370281)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.380287)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.390293)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.4003)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.410306)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.420312)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.430318)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.440325)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.450331)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.460337)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.470344)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.48035)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.490356)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.500362)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.510369)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.520375)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.530381)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.540388)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.550394)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.5604)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.570406)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.580413)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.590419)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.600425)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.610432)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.620438)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.630444)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.64045)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.650457)
```

```
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.780538)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.790545)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.800551)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.810557)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.820563)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.83057)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.840576)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.850582)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.860589)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.870595)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.880601)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.890607)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.900614)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.91062)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.920626)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.930633)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.940639)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.950645)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.960651)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.970658)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.980664)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_DOWNLOADING (3), 0.99067)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.01)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.03)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.04)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.05)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.08)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.1)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.12)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.13)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.14)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.16)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.17)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.19)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.21)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.22)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.23)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.25)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.26)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.27)
[INFO:update_engine_client_android.cc(90)] onStatusUpdate(UPDATE_STATUS_FINALIZING (5), 0.29)
```



上面的截图展示了完整的升级过程，升级成功后，会有UPDATE_STATUS_UPDATED_NEED_REBOOT地打印信息，如上截图所示。此时手动重启设备，就可以切换到新的升级后的系统。

5. 注意事项

1. 打包生成update.img时要注意修改package-file增加对应的分区。
   比如trust和uboot配置如下，其他分区类似。
   trust_a Image/trust.img
   trust_b Image/trust.img
   uboot_a Image/uboot.img
   uboot_b Image/uboot.img
   注意，对于rk3566/rk3568/rk3588等芯片来说，没有trust分区，则不需要trust_a和trust_b分区。

6. A/B miniloader OTA

Miniloader是Rockchip平台的一级引导程序，正常情况下该固件不需要进行OTA升级，并且也不建议这么做，在特殊情况下，如果需要对miniloader进行OTA升级，针对Rockchip A/B系统平台，可采用本节介绍的方案来实现。

需要注意的是该miniloader OTA升级包与正常的A/B升级包没有任何关系，相互独立。也就是说当有需要升级A/B系统中的miniloader时需按照本节6.1介绍的方法生成对应的miniloader升级包，并且客户端需按照6.2的说明进行开发（在升级miniloader升级包时执行6.2的参考指令后会进入recovery模式升级A/B系统的miniloader，升级完成后自动重启）。

6.1 rkloader升级包构建

对于>=Android 12来说，以OTA方式完成固件编译后（即make installcelan && make -j16 && make dist -j16 && ./mkimage_ab.sh ota），在rockdev下自动生成的update_loader.zip文件就是rk loader OTA升级包。

6.2 升级命令参考

升级客户端apk可参考如下命令方式实现：

```
adb root
adb push update.zip /cache/
adb shell "echo \"--fw_rkloader=/cache/update.zip\" > /cache/recovery/command"
adb reboot recovery
```