

Unisoc Confidential For hiar

# Android 9.0 OTA 升级至 Android 10.0 操作指南

文档版本

V1.0

发布日期

2020-10-18

**版权所有 © 紫光展锐（上海）科技有限公司。保留一切权利。**

本文件所含数据和信息都属于紫光展锐（上海）科技有限公司（以下简称紫光展锐）所有的机密信息，紫光展锐保留所有相关权利。本文件仅为信息参考之目的提供，不包含任何明示或默示的知识产权许可，也不表示有任何明示或默示的保证，包括但不限于满足任何特殊目的、不侵权或性能。当您接受这份文件时，即表示您同意本文件中内容和信息属于紫光展锐机密信息，且同意在未获得紫光展锐书面同意前，不使用或复制本文件的整体或部分，也不向任何其他方披露本文件内容。紫光展锐有权在未经事先通知的情况下，在任何时候对本文件做任何修改。紫光展锐对本文件所含数据和信息不做任何保证，在任何情况下，紫光展锐均不负任何与本文件相关的直接或间接的、任何伤害或损失。

请参照交付物中说明文档对紫光展锐交付物进行使用，任何人对紫光展锐交付物的修改、定制化或违反说明文档的指引对紫光展锐交付物进行使用造成的任何损失由其自行承担。紫光展锐交付物中的性能指标、测试结果和参数等，均为在紫光展锐内部研发和测试系统中获得的，仅供参考，若任何人需要对交付物进行商用或量产，需要结合自身的软硬件测试环境进行全面的测试和调试。

Unisoc Confidential For hiar

**紫光展锐（上海）科技有限公司**



# 前言

## 概述

本文档主要介绍 Android 9.0 升级至 Android 10.0 需求实现技术说明以及实际操作指导。升级要求无缝升级，关键指标是不能返厂重校、用户数据不能丢。

## 读者对象


本文档主要适用于进行 OTA 功能开发和测试的工程师，以及负责项目版本编制的工程师。

## 缩略语

缩略语	英文全名	中文解释
OTA	Over-the-Air	无线软件系统版本升级
Repart	Repartition	重新分区程序
DP	Dynamic Partition	动态分区调整

## 符号约定

在本文中可能出现下列标志，它所代表的含义如下。

符号	说明
 说明	用于突出重要/关键信息、补充信息和小窍门等。 “说明”不是安全警示信息，不涉及人身、设备及环境伤害。

## 变更信息

文档版本	发布日期	修改说明
V1.0	2020-10-18	第一次正式发布。

## 关键字

OTA、代码合入、掉电保护

Unisoc Confidential For hiar

# 目 录

1 OTA 升级介绍.....	1
2 OTA 升级实现.....	2
2.1 实现办法.....	2
2.2 代码合入.....	6
2.2.1 Android 9.0.....	6
2.2.2 Android 10.0.....	7
2.3 Android 10.0 升级包产生办法.....	8
2.4 掉电保护.....	8

Unisoc Confidential For hiar

# 1 OTA 升级介绍

---

## 1.1 注意事项

实现 Android 9.0 到 Android 10.0 的无缝升级，并完整保存升级前的用户数据及运行参数，需要注意的是：

- 若 Android 9.0 项目已实现量产：需要将 Android 10.0 上的 userdata 分区地址与 Android 9.0 对齐。此时需要调整 Android 10.0 空间最大的 super 分区，确保缩小后的 super 分区空间足够。
- 若 Android 9.0 项目未实现量产：在设计 Android 9.0 分区大小时要明确是否有升级到 Android 10.0 甚至 Android 11.0 的需求。如果需要继续升级，则提前在 Android 9.0 上扩大 system/vendor/product 分区，避免后续升级过程中出现由于系统较大，super 分区空间不足的问题。

## 1.2 实现要点

Android 10.0 相对于 Android 9.0 新增 DP、UDC、tee CFG、kernel ko 等独立功能，相应分区发生较大变化。其中 super 分区涵盖 Android 9.0 上的 system/vendor/product 分区，另新增 metadata、tee CFG、socko/odmko 等分区。因此，实现 userdata 不丢失的 OTA 升级，主要采取以下措施：

- 重新划分变为 Android 10.0 的分区。
- 保证 userdata 起始不变。

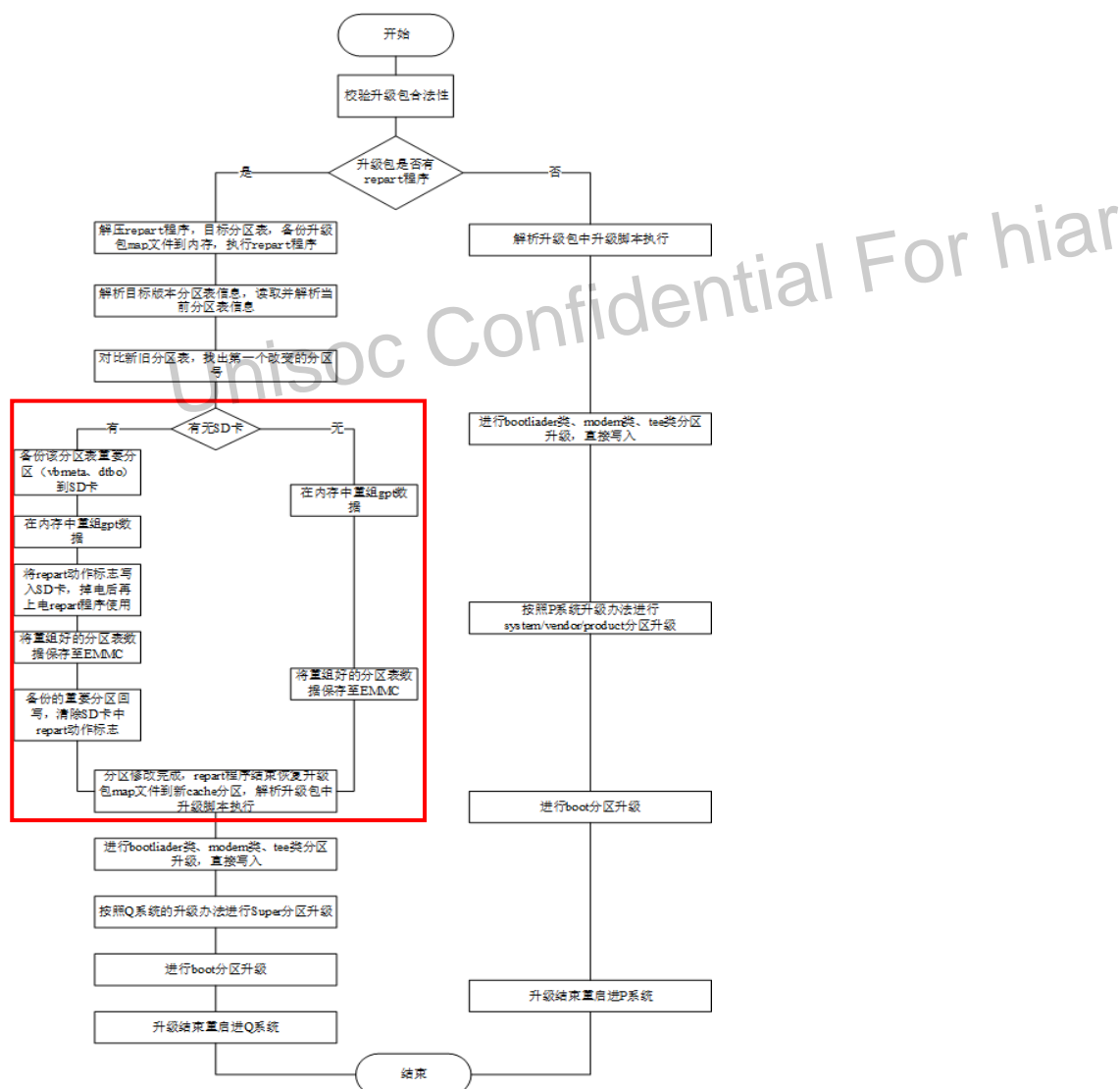
# 2 OTA 升级实现

## 2.1 实现办法

### 重新分区

将 Android 分 9.0 分区表改造为 Android 10.0 的分区表，并将 Android 10.0 各分区内容升级到分区表中，确保 userdata 不丢失。repart（下同）程序负责重新分区，该程序的整个升级流程如图 2-1 所示。

图2-1 重新分区升级流程



## 说明

实线红框中是容易掉电的流程。

Q 系统即 Android 10.0, P 系统即 Android 9.0。

## 分区表差异

以 T7510 2h10 基线为例, Android 9.0 和 Android 10.0 分区表差异如表 2-1 所示:

表2-1 Android 9.0 和 Android 10.0 分区表差异

Android 9.0 Partition	Android 10.0 Partition
<Partition id="miscdata" size="5"/>	<Partition id="miscdata" size="5"/>
<Partition id="misc" size="1"/>	<Partition id="misc" size="1"/>
<Partition id="nr_fixnv1" size="8"/>	<Partition id="nr_fixnv1" size="8"/>
<Partition id="nr_fixnv2" size="8"/>	<Partition id="nr_fixnv2" size="8"/>
<Partition id="prodnv" size="10"/>	<Partition id="prodnv" size="10"/>
<Partition id="nr_runtimenv1" size="10"/>	<Partition id="nr_runtimenv1" size="10"/>
<Partition id="nr_runtimenv2" size="10"/>	<Partition id="nr_runtimenv2" size="10"/>
<Partition id="recovery" size="40"/>	<Partition id="recovery" size="40"/>
<Partition id="trustos" size="6"/>	<Partition id="trustos" size="6"/>
<Partition id="trustos_bak" size="6"/>	<Partition id="trustos_bak" size="6"/>
<Partition id="sml" size="1"/>	<Partition id="sml" size="1"/>
<Partition id="sml_bak" size="1"/>	<Partition id="sml_bak" size="1"/>
<Partition id="uboot" size="1"/>	<Partition id="uboot" size="1"/>
<Partition id="uboot_bak" size="1"/>	<Partition id="uboot_bak" size="1"/>
<Partition id="uboot_log" size="4"/>	<Partition id="uboot_log" size="4"/>
<Partition id="logo" size="7"/>	<Partition id="logo" size="8"/>
<Partition id="fbootlogo" size="7"/>	<Partition id="fbootlogo" size="8"/>
<Partition id="l_pmsys" size="1"/>	<Partition id="l_pmsys" size="1"/>
<Partition id="l_agdsp" size="6"/>	<Partition id="l_agdsp" size="6"/>
<Partition id="gnssmodem" size="1"/>	<Partition id="gnssmodem" size="1"/>
<Partition id="wcnmodem" size="10"/>	<Partition id="wcnmodem" size="10"/>
<Partition id="persist" size="2"/>	<Partition id="persist" size="2"/>
<Partition id="nr_spl" size="1"/>	<Partition id="nr_spl" size="1"/>
<Partition id="nr_sml" size="1"/>	<Partition id="nr_sml" size="1"/>
<Partition id="nr_uboot" size="1"/>	<Partition id="nr_uboot" size="1"/>
<Partition id="nr_boot" size="35"/>	<Partition id="nr_boot" size="35"/>
<Partition id="nr_pmsys" size="1"/>	<Partition id="nr_pmsys" size="1"/>
<Partition id="nr_agdsp" size="6"/>	<Partition id="nr_agdsp" size="6"/>
<Partition id="nr_modem" size="40"/>	<Partition id="nr_modem" size="40"/>
<Partition id="nr_v3phy" size="8"/>	<Partition id="nr_v3phy" size="8"/>
<Partition id="nr_nrphy" size="8"/>	<Partition id="nr_nrphy" size="8"/>



Android 9.0 Partition	Android 10.0 Partition
<pre> &lt;Partition id="nr_nrdsp1" size="5"/&gt; &lt;Partition id="nr_nrdsp2" size="5"/&gt; &lt;Partition id="nr_deltanv" size="2"/&gt; &lt;Partition id="boot" size="35"/&gt; &lt;Partition id="dtb" size="8"/&gt; &lt;Partition id="dtbo" size="8"/&gt; &lt;Partition id="system" size="3000"/&gt; &lt;Partition id="cache" size="150"/&gt; &lt;Partition id="vendor" size="550"/&gt; &lt;Partition id="product" size="100"/&gt; &lt;Partition id="vbmeta" size="1"/&gt; &lt;Partition id="vbmeta_bak" size="1"/&gt; &lt;Partition id="sysdumpdb" size="10"/&gt; &lt;Partition id="userdata" size="0xFFFFFFFF"/&gt; </pre>	<pre> &lt;Partition id="nr_nrdsp1" size="5"/&gt; &lt;Partition id="nr_nrdsp2" size="5"/&gt; &lt;Partition id="nr_deltanv" size="2"/&gt; &lt;Partition id="teecfg" size="1"/&gt; &lt;Partition id="teecfg_bak" size="1"/&gt; &lt;Partition id="boot" size="35"/&gt; &lt;Partition id="dtbo" size="8"/&gt; &lt;Partition id="super" size="4100"/&gt; &lt;Partition id="cache" size="150"/&gt; &lt;Partition id="socko" size="75"/&gt; &lt;Partition id="odmko" size="25"/&gt; &lt;Partition id="vbmeta" size="1"/&gt; &lt;Partition id="vbmeta_bak" size="1"/&gt; &lt;Partition id="metadata" size="16"/&gt; &lt;Partition id="sysdumpdb" size="10"/&gt; &lt;Partition id="vbmeta_system" size="1"/&gt; &lt;Partition id="vbmeta_vendor" size="1"/&gt; &lt;Partition id="userdata" size="0xFFFFFFFF"/&gt; </pre>

## 说明

表 2-1 中的加粗字体为分区表差异内容。

## 对齐原则

分区对齐必须遵守如下原则：

- nr\_fixnv1 之前的分区大小必须保持一致。
- userdata 前所有分区大小之和相等。

## Android 10.0 分区确定办法

按照如下步骤进行确定：

- 步骤 1 将 logo/fbootlogo 的 size 保持跟 Android 9.0 中的一致（3MB），如果这两个分区在 Android 9.0 上扩大过，那么在 Android 10.0 上，两个分区大小保持不变。
- 步骤 2 调整 Android 10.0 上项目分区表（如表 2-2 所示），使 userdata 分区 size 值的和为 3538MB。

表2-2 调整后的 Android 10.0 分区表

Android 10.0 分区表	
650	<Partition id="miscdata" size="5"/>
651	<Partition id="misc" size="1"/>
652	<Partition id="nr_fixnv1" size="8"/>
653	<Partition id="nr_fixnv2" size="8"/>
654	<Partition id="prodnv" size="10"/>
655	<Partition id="nr_runtimenv1" size="10"/>
656	<Partition id="nr_runtimenv2" size="10"/>
657	<Partition id="recovery" size="40"/>
658	<Partition id="trustos" size="6"/>
659	<Partition id="trustos_bak" size="6"/>
660	<Partition id="sml" size="1"/>
661	<Partition id="sml_bak" size="1"/>
662	<Partition id="uboot" size="1"/>
663	<Partition id="uboot_bak" size="1"/>
664	<Partition id="uboot_log" size="4"/>
<b>665</b>	<b>&lt;Partition id="logo" size="7"/&gt;</b>
<b>666</b>	<b>&lt;Partition id="fbootlogo" size="7"/&gt;</b>
667	<Partition id="l_pmsys" size="1"/>
668	<Partition id="l_agdsp" size="6"/>
669	<Partition id="gnssmodem" size="1"/>
670	<Partition id="wcnmodem" size="10"/>
671	<Partition id="persist" size="2"/>
672	<Partition id="nr_spl" size="1"/>
673	<Partition id="nr_sml" size="1"/>
674	<Partition id="nr_uboot" size="1"/>
675	<Partition id="nr_boot" size="35"/>
676	<Partition id="nr_pmsys" size="1"/>
677	<Partition id="nr_agdsp" size="6"/>
678	<Partition id="nr_modem" size="40"/>
679	<Partition id="nr_v3phy" size="8"/>
680	<Partition id="nr_nrphy" size="8"/>
681	<Partition id="nr_nrdsp1" size="5"/>
682	<Partition id="nr_nrdsp2" size="5"/>
683	<Partition id="nr_deltanv" size="2"/>
684	<Partition id="teecfg" size="1"/>
685	<Partition id="teecfg_bak" size="1"/>
686	<Partition id="boot" size="35"/>
687	<Partition id="dtbo" size="8"/>
<b>688</b>	<b>&lt;Partition id="super" size="3538"/&gt;</b>

Android 10.0 分区表	
689	<Partition id="cache" size="150"/>
690	<Partition id="socko" size="75"/>
691	<Partition id="odmko" size="25"/>
692	<Partition id="vbmeta" size="1"/>
693	<Partition id="vbmeta_bak" size="1"/>
694	<Partition id="metadata" size="16"/>
695	<Partition id="sysdumpdb" size="10"/>
696	<Partition id="vbmeta_system" size="1"/>
697	<Partition id="vbmeta_vendor" size="1"/>
698	<Partition id="userdata" size="0xFFFFFFFF"/>

### 说明

表 2-2 中的加粗字体为调整后的内容。

----结束

## 2.2 代码合入

### 总体流程

所有代码合入步骤为：

1. 合入 Android 9.0 上的改动并编制该版本，该版本称为 Android 9.0 +。
2. 合入 Android 10.0 上所有改动的展锐主线，使用释放的代码，将分区对齐的修改根据实际项目合入即可。
3. 升级试验。

----结束

### 2.2.1 Android 9.0

#### 改动要点：

- 添加升级 super 分区时对 device mapper 的设备节点操作 se 权限。
- 改造 repart 程序，无 SD 卡亦能 repartition。
- 改造 recovery 程序，添加大版本升级流程，并与原始代码开关隔开（关闭宏开关即可恢复至最初状态）。
- 编译 recovery.img 时将 dtb 打包进去（关闭宏开关即可恢复至最初状态）。

## 合入办法：

差分/整包合入。

## 2.2.2 Android 10.0

### 改动要点：

- 将 Android 9.0 上的 repart 程序预置到 Android 10.0 对应的 board 中。
- 增加新划分区升级包打包流程。
- 增加 odmko 和 super metadata 打包及升级流程。
- 增加 tos 自适应获取 Android 9.0 或者 Android 10.0 对应版本的 keymaster 数据，使 userdata 的加密校验能通过并开机。

### 合入办法：

1. 判断是否需要替换 repart 程序文件。
  - 如果在 Android 9.0 上对 repart 程序做过修改，请遵循预置 repart 程序产生办法并提交。
  - 如果未做修改，则使用展锐 Android 10.0 版本 release 中的默认程序即可。
2. 分区跟 Android 9.0 对齐。
3. 其余使用 Android 10.0 的 release 原生代码。

### 预置 repart 程序

预置 repart 程序产生办法的步骤如下：

步骤 1 Android 9.0 上合入 samplecode。

步骤 2 在 Android 9.0 对应的 board 上，lunch 环境后运行 make repart 编出（user/userdebug 皆可，产生的目标程序路径为 out/target/product/s9863a1h10/system/bin/repart）。

步骤 3 将对应程序文件覆盖至 Android 10.0 对应的 device/sprd/sharkl3/s9863a1h10/recovery 目录下，并提交。

### 📖 说明

同项目的非 go board 和 go board，在不同平台下必须单独产生。因为两者存在 64 位和 32 位的差异，不同平台下，程序无法兼容运行。

---结束

## 2.3 Android 10.0 升级包产生办法

### 升级路线

由于 Android 9.0 上有 recovery 等改动才能进行升级，假定 Android 9.0 上合入 patch 后版本为 9.0+，最终升级路线图是“Android 9.0 → Android 9.0+ → Android 10.0”。由于是修改分区后再进行升级，该升级路线只支持整包升级。

### 产生条件

Android 10.0 代码改动就绪。

### 产生步骤

步骤 1 执行 lunch 工程并 make 全编。

步骤 2 执行 make otapackage，编译产生的 target 包命名为 q-target.zip。

步骤 3 使用如下命令产生升级整包。

- User 版本：

```
/build/make/tools/releasetools/ota_from_target_files --repart -k build/target/product/security/release/releasekey q-target.zip ptoq_ota_updatet.zip
```

- Userdebug 版本：

```
/build/make/tools/releasetools/ota_from_target_files --repart -k build/target/product/security/testkey q-target.zip ptoq_ota_updatet.zip
```

----结束

## 2.4 掉电保护

### 各阶段升级时间

以 SC9863A User 版本基线为例，recovery 模式中升级各阶段用时统计（单位：秒），如图 2-2 所示。

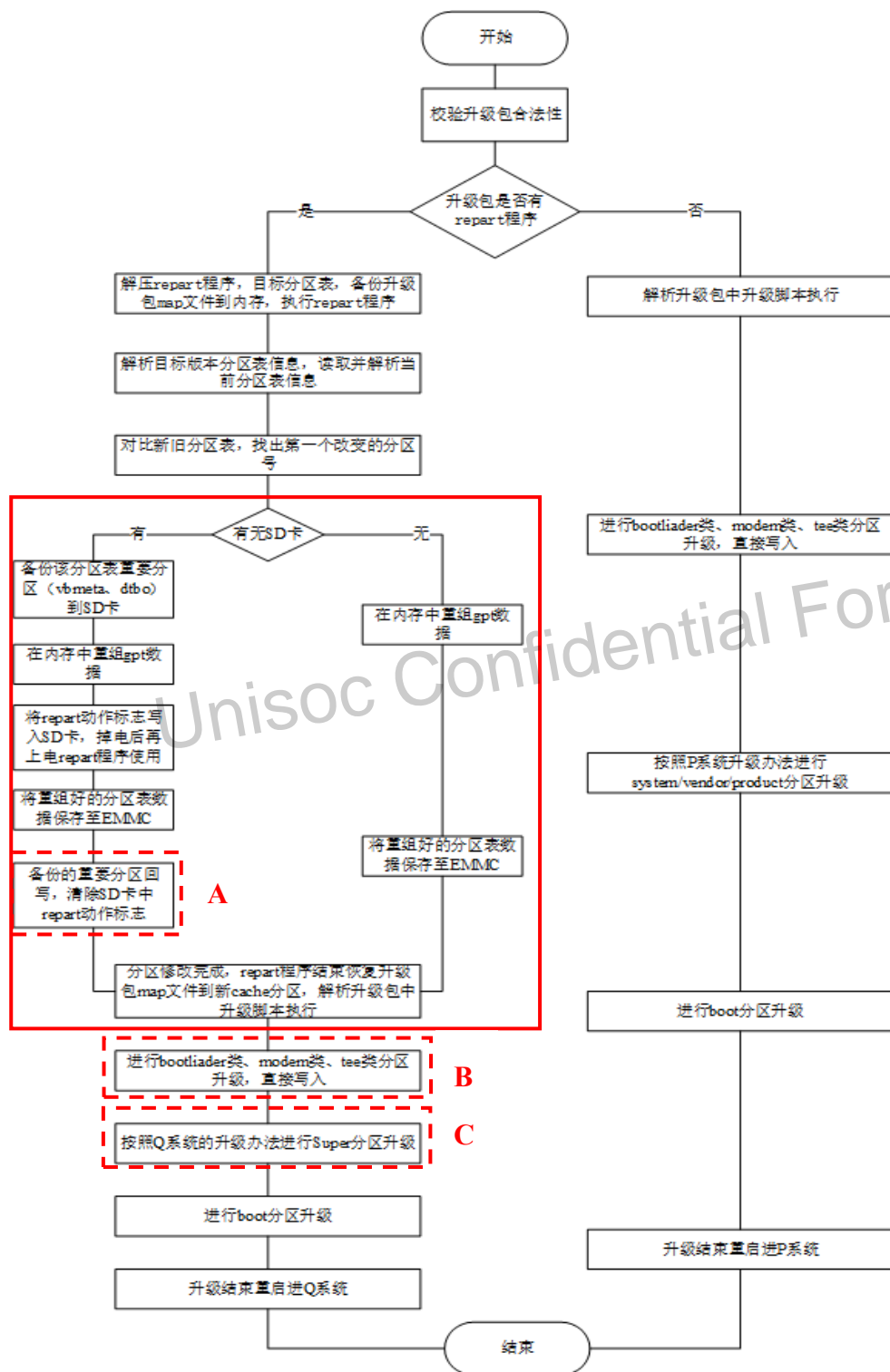
图2-2 recovery 中升级各阶段用时统计



## 各阶段掉电保护

升级时可能掉电节点如图 2-3 所示。

图2-3 掉电节点



## 说明

实线红框中是容易掉电的部分。虚线红框是掉电点。

Q 系统即 Android 10.0, P 系统即 Android 9.0。

掉电对升级的影响分有 SD 卡插入和无 SD 卡插入两种情况。

## 有 SD 卡插入

有 SD 卡即能进行 vbmeta 和 dtbo 分区备份恢复, 图 2-3 中 A、B、C 掉电结果如下。

- A 点掉电: 在 repart 程序运行期间, 备份在 SD 卡的两分区回写时掉电, 由于新划出 vbmeta 和 dtbo 为乱数, 再上电时无法进入 recovery 模式继续升级。
- B 点掉电: 由于新增的 vbmeta\_system 和 vbmeta\_vendor 未升级到分区中, 升级至 Android 10.0 的 bootloader 会校验这两项, 导致升级过程卡在 uboot 阶段, 无法进入 recovery 模式继续升级。
- C 点掉电: 在 Android 10.0 中, 红框中各个 bin 已写入新划出的分区中, 再上电重启能进入 recovery 模式 (将 dtb 打进 Android 9.0 的 recovery.img 中可解决) 继续升级, 直至结束。

## 无 SD 卡插入

无 SD 卡设备无备份恢复 vbmeta 和 dtbo 分区数据动作, 图 2-3 中 A、B、C 掉电结果如下。

- A 点掉电: repart 程序运行期间, 新分区表写入后, vbmeta/dtbo 已经破坏, 由于新划出 vbmeta 和 dtbo 为乱数, 再上电时无法进入 recovery 模式继续升级。
- B 点掉电: 在 Android 10.0 中, 红框中各个部分 (尤其 avb/dtb/dtbo) 再上电时无法进入 recovery 模式继续升级。
- C 点掉电: 在 Android 10.0 中, 红框中各个 bin 已写入新划出的分区中, 再上电重启能进入 recovery 模式 (将 dtb 打进 Android 9.0 的 recovery.img 中可解决) 继续升级, 直至结束。

## 掉电宕机概率

从各动作所占时间看, 掉电落在 A 和 B 点的概率为  $3.5/195.5 = 0.0179$ , 即一旦有掉电, 宕机概率为 1.79%。从掉电时间点看, 有无 SD 卡的宕机概率一致。因此, 更改分区的方案无法 100% 规避掉电宕机问题。

## 解决办法

整个 FOTA 升级过程中, 请保持通电状态, 不可掉电!