

1 简介

本应用笔记主要介绍了由 MCU SE 团队推出的安全 OTA 项目 Secure BootLoader (SBL)及 Secure FirmWare (SFW) 的设计框架以及 FOTA 功能的实现。

SBL 及 SFW 是由恩智浦边缘处理事业部的系统应用组推出的一个针对 MCU 的 OTA 项目，目前该项目支持 i.MXRT 系列的大多数芯片以及 LPC55S69 芯片。

本应用笔记基于 i.MXRT1010-EVK 及 i.MXRT1020-EVK 来介绍 FOTA 相关的设计与实现。

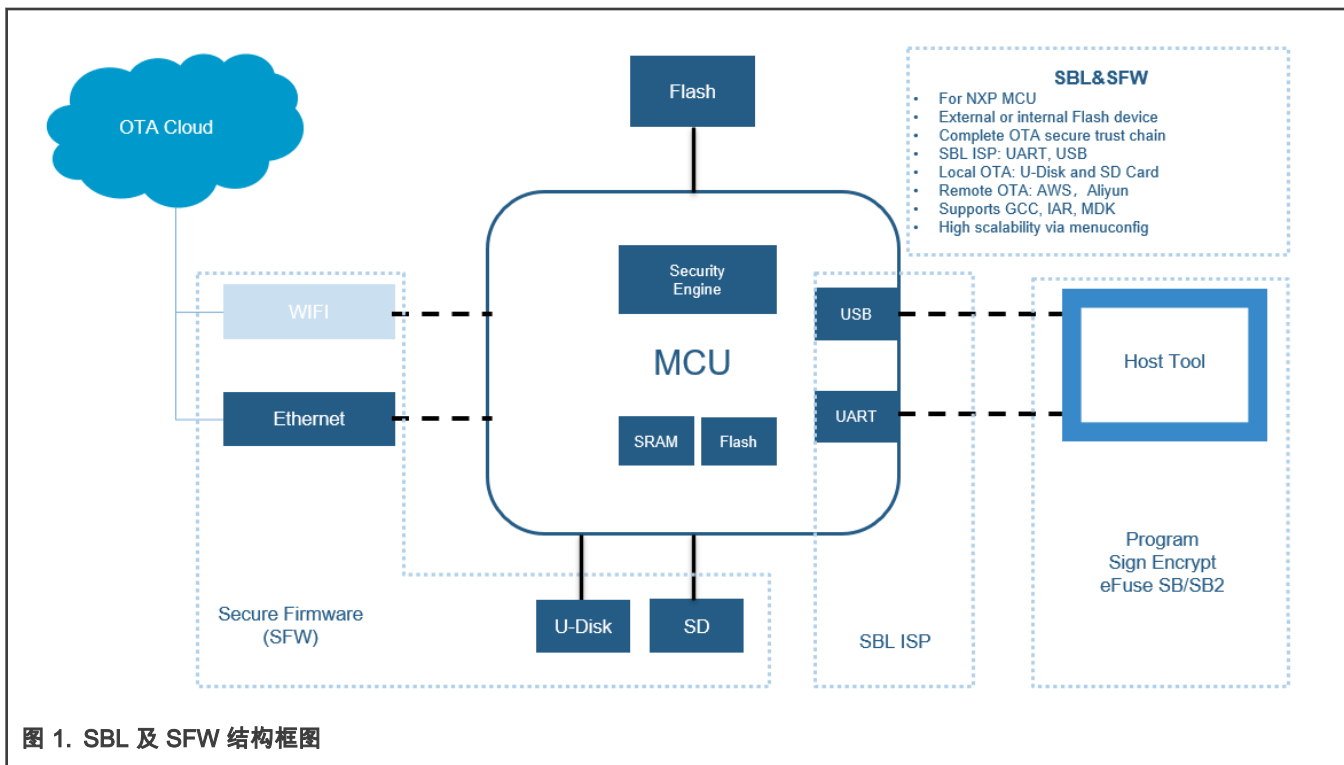
2 SBL 及 SFW 简介

SBL 及 SFW 是由恩智浦推出的一个面向 MCU 的安全固件升级项目。SBL 是一个配合具备 FOTA 能力的固件使用的二级 Bootloader，它通过验证和写入 OTA 镜像到内部/外部 flash 的指定区域来存储和管理 OTA 镜像升级。

SFW 是基于 FreeRTOS 创建的，被设计用于与 SBL 一起实现完整的 FOTA 流程。SFW 支持通过 U 盘、本地 SD 卡获取固件镜像，或者远程 AWS 云、阿里云获取 OTA 固件镜像，在获取新的固件镜像后，SFW 本身会将镜像写入 flash，并且置位相应标志位后重启设备。进入 SBL 后，SBL 会根据标志位状态，查找新的固件，并且完成升级。SBL 及 SFW 的结构框图如图 1 所示。

目录

1	简介.....	1
2	SBL 及 SFW 简介.....	1
3	FOTA 架构设计.....	2
3.1	Swap 模式.....	2
3.2	Remap 模式.....	6
4	参考资料.....	10
5	修订记录.....	10



3 FOTA 架构设计

本应用笔记中的 FOTA 泛指固件升级，下述 FOTA 都代表固件升级。

在本项目中，FOTA 事件的监测，新固件的接收与写入，标志位的设置由 SFW 负责。SFW 使用 FreeRTOS 进行开发，在用于实现应用功能的任务群的基础上，还会创建多个用于接收新固件的任务。在 SFW 工程中，分别设置了 SD 卡升级任务，U 盘升级任务，AWS 或阿里云升级任务。这些升级任务可在 SFW 的 menuconfig 中配置是否使能。

U 盘及 SD 卡的升级任务会检测插入的 U 盘或 SD 卡中是否存在新的固件镜像，如果存在新的固件镜像，就会读取镜像，并且将镜像存储到 Flash 的指定位置。AWS 或阿里云的升级任务，会接收云平台的 OTA 指令，当云平台发起 OTA 请求时，AWS 或阿里云的升级任务会从云平台接收新的固件镜像，并且存储到 Flash 的指定位置。基本的流程如 图 2 所示。

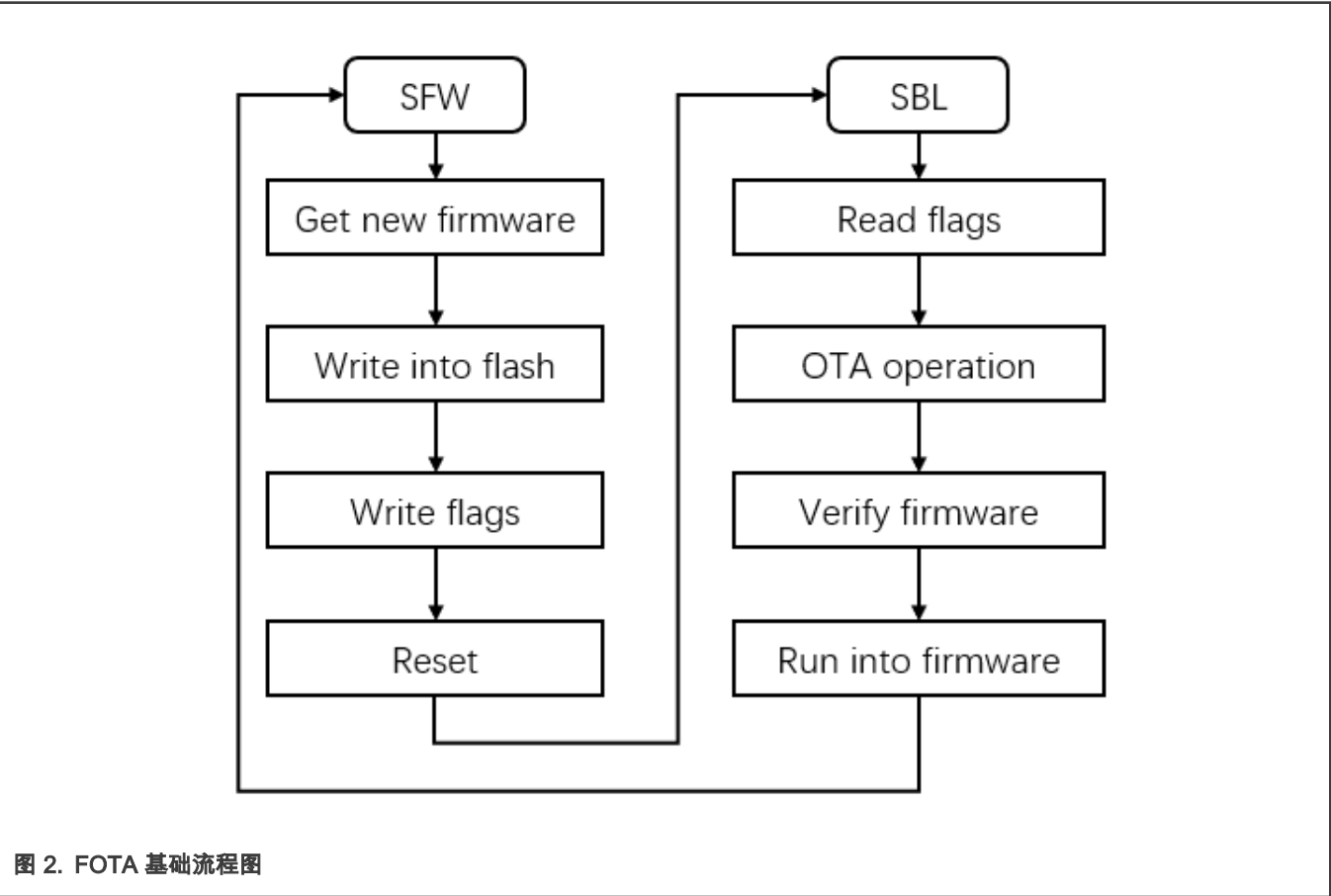


图 2. FOTA 基础流程图

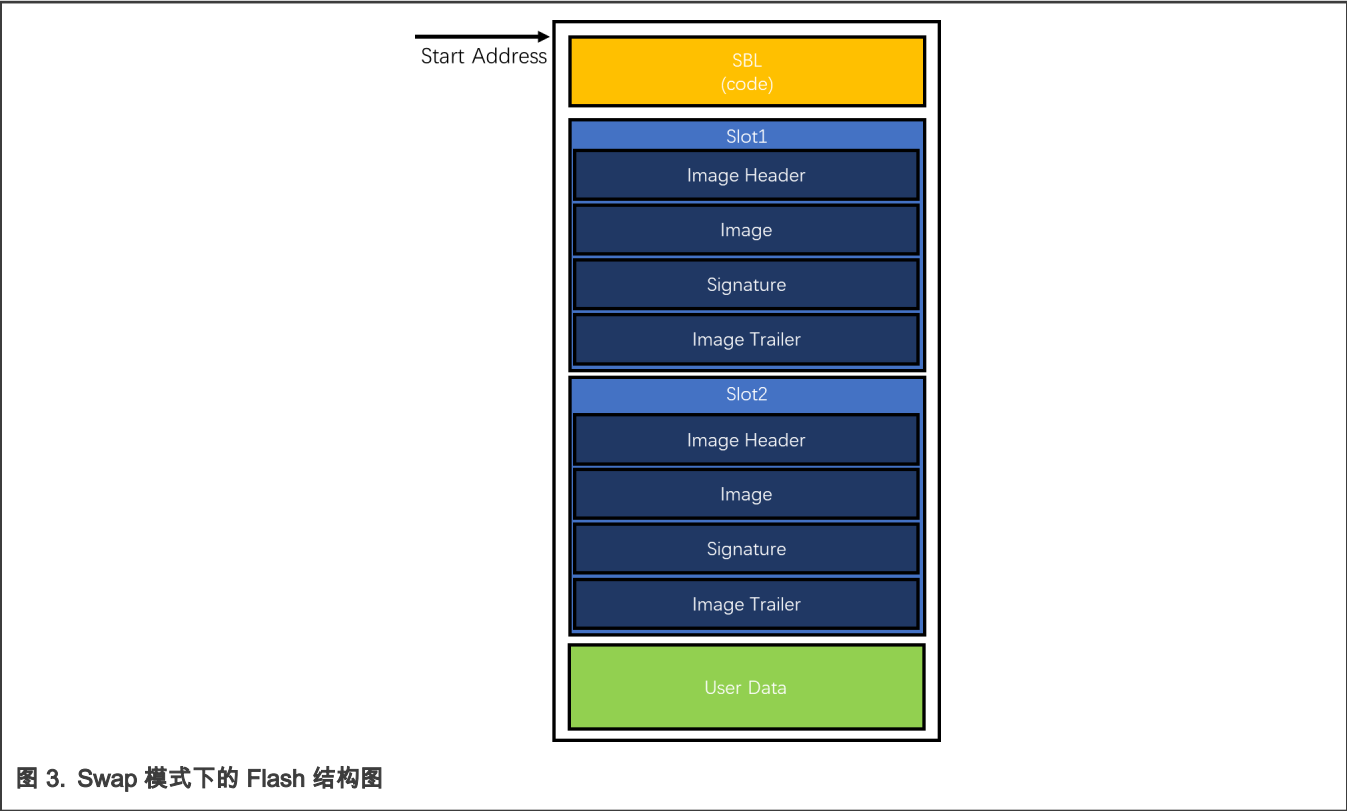
i.MXRT 系列的 MCU 中，有多款芯片具备 Remap 功能，所以在设计 FOTA 的框架时，定义了两种不同的模式分别是 Swap 模式和 Remap 模式。对应到流程图中的 write flags，read flags 以及 OTA operation 的操作都不一样。在 [Swap 模式](#)和 [Remap 模式](#)中会单独介绍这部分。

为了支持固件镜像的回滚功能，需要将 Flash 划分出至少三片空间，分别用于存放 bootloader，firmware1 以及 firmware2，将这三片空间定义为 SBL 区，Slot1 和 Slot2。

3.1 Swap 模式

首先介绍 Swap 模式，对于 MCU 来说，程序的执行地址依赖于链接文件的指定，在代码进行链接时就决定了程序的起始地址。为了保持 Firmware 工程的简洁，只维护一个链接文件。这就需要 bootloader 来将新的固件搬运到链接文件指定的地址，然后再跳转运行。这个过程中需要交换 Flash 中存储的镜像的位置，所以把这一模式定义为 Swap 模式。

SBL 中的 Swap 模式基于开源的 mcuboot 开发，使用的是 1.7.0 版本的 mcuboot 源码。要实现 Swap 模式下的 FOTA 功能，所使用的固件镜像需要有一个固定的组织形式，以让 SBL 能够通过对指定位置的标志位读取来控制 FOTA 的流程。Swap 模式下的固件在 Flash 上的组织形式如 图 3 所示。



SBL 占据 Flash 的起始位置，这里也是 i.MXRT 系列芯片的 ROM 跳转的地方。Slot1 与 Slot2 中存储的都是固件镜像，它们分别由四个部分组成。在镜像的最前面是 Image Header，存储的是固件的长度，版本号等信息，地址上紧跟在 Image Header 后面的是固件本身，在固件本身的后面是固件的签名，最后在 Slot（前文中定义的存储器中的一块存储 firmware 的区域）的末尾是 Image Trailer，存储的是 FOTA 过程的 Flag 信息。

SFW 经过编译后，得到的是纯粹的不带 Image Header，签名以及 Image Trailer 的固件镜像。需要额外使用脚本给镜像添加 Header 及签名才可以用作更新用的新固件参与 FOTA。Image Trailer 部分则是在 SFW 及 SBL 的程序中编辑。Image Header 的结构如 表 1 所示。

表 1. Image header 结构

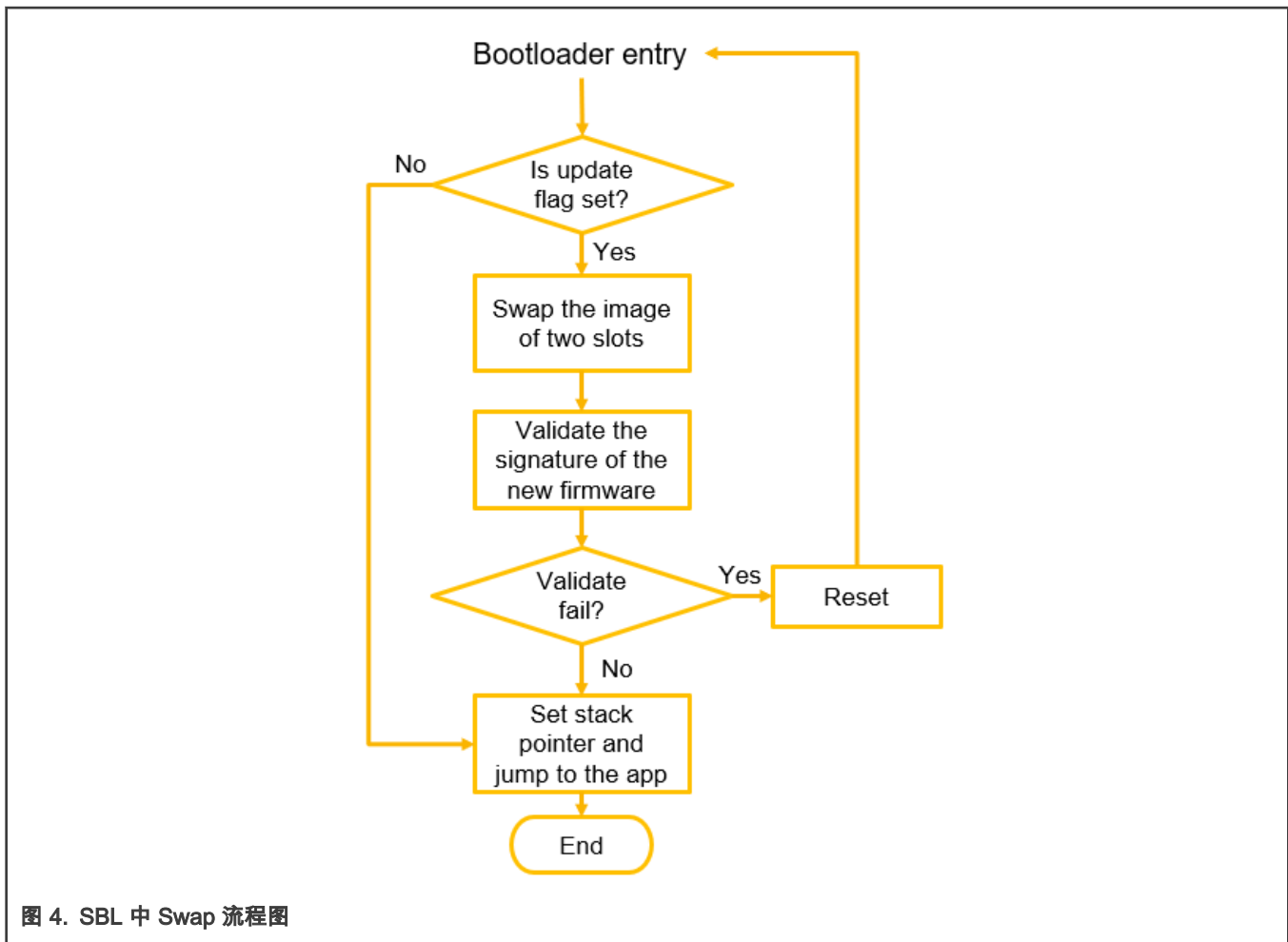
Offset	Width (bytes)	Field	Description
0x00	4	magic	Image header tag Fixed value
0x04	4	load_addr	Point to the load address of the application
0x08	2	header_size	Size of the image header
0x0a	2	reserved	Reserved for future use
0x0c	4	image_size	The size of the image (not including the Image Header Size)
0x10	4	flags	Not used now
0x14	8	image_version	Image version
0x1c	4	reserved	Reserved for future use

Image Trailer 的结构如 表 2 所示。

表 2. Image trailer 结构

Offset	Width (bytes)	Field	Description
0x00	1	copy_done	Flag that the Swap done
0x01	7	Pad	Reserved
0x08	1	image_ok	Flag that control the OTA state
0x09	7	pad	Reserved
0x10	16	magic	Image trailer tag Fixed value

Swap 模式下，SBL 的跳转地址被固定为 Slot1 的固件起始地址，所以从 SFW 中所得到的新的固件镜像存储的位置一直都会是 Slot2。在 SFW 接收到新的固件镜像后，都会将它写入 Slot2 的 Flash 空间内。写入完成后将会置位 Trailer 中的 magic 值以向 SBL 传递有新固件需要更新的信息。



在 Reset 回 SBL 后，SBL 的代码逻辑遵循 图 4 的流程，这里与 mcuboot 中的流程稍有不同，在 mcuboot 项目中会先对 Slot2 中存储的新的镜像进行验签，验签完成后再进行 Slot1 和 Slot2 中镜像的交换。但是 SBL 的项目为了支持 i.MXRT 系列芯

片中 ROM 的验签功能，ROM 验签的功能是与链接地址相关的，验证签名时镜像所在的地址需要与链接地址一致，所以 SBL 中将交换镜像的操作放在了验签之前，在新的镜像存入 Slot1 后再进行验签。

再回到交换镜像的上一步，SBL 需要判断当前 FOTA 的状态，再决定是否需要交换两个 Slot 的镜像。在 SBL 内对 Swap 模式下的 FOTA 定义了四种状态，分别是 Test，Perm，Revert 和 None。

- **Test** 表示临时交换。
- **Perm** 表示永久交换。
- **Revert** 表示回滚交换。
- **None** 表示无交换。

通过读取两个 Slot 的 Image Trailer 的值决定了当前 FOTA 的状态，前三种状态下，Image Trailer 的配置如 图 5 所示。

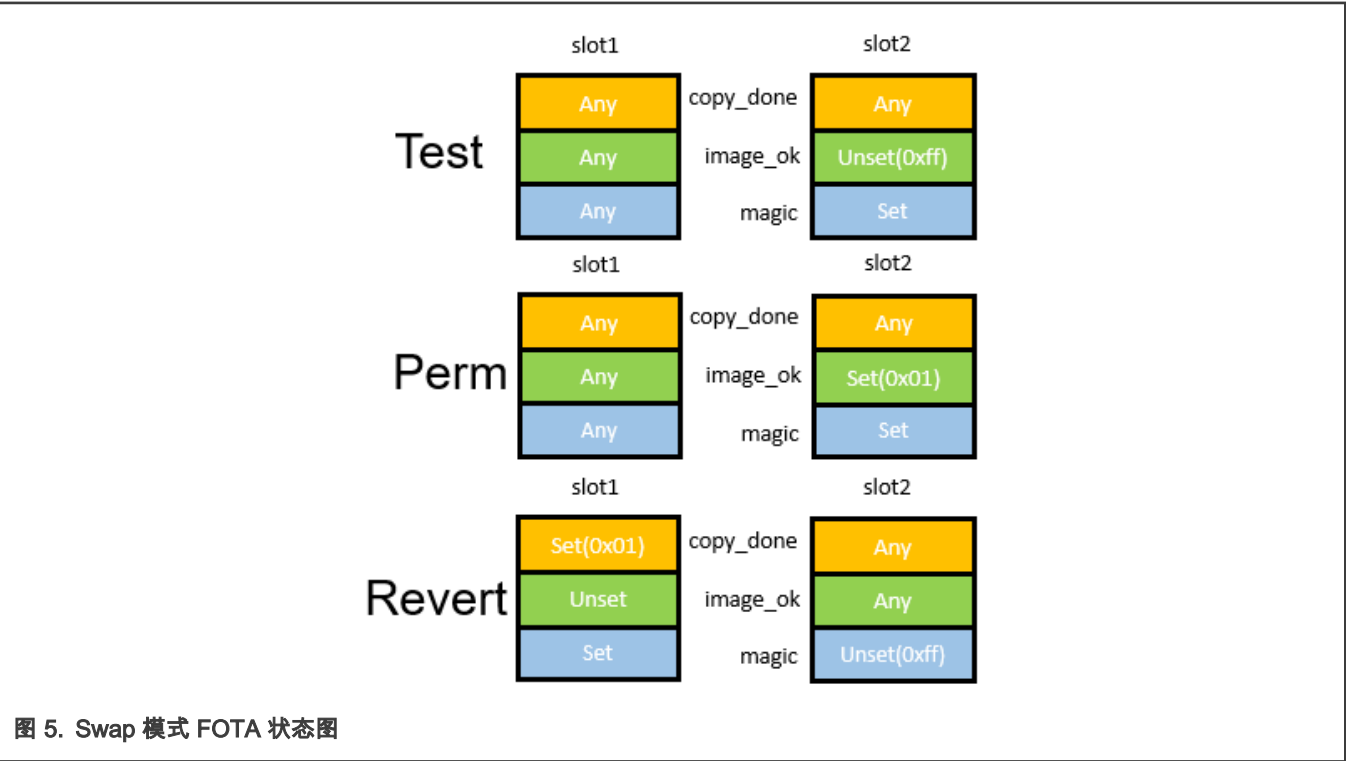
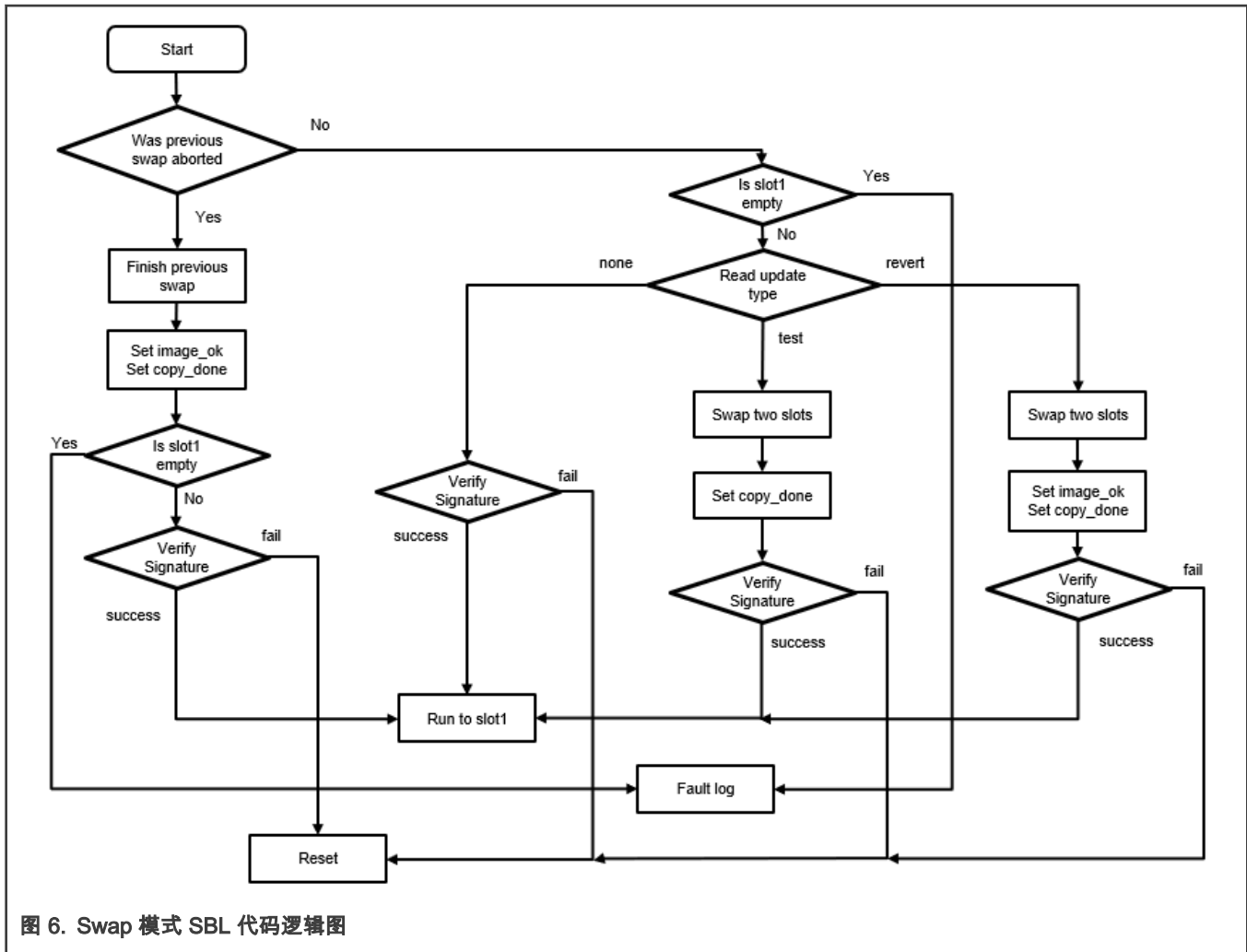


图 5. Swap 模式 FOTA 状态图

除了这三种状态以外的 Image Trailer 组合都划分为 None 状态，不会发生镜像的交换。在实际的 SBL 工程中，默认支持镜像的回滚功能，所以不会使用到上述的 Perm 状态。

回滚功能的实现同样依赖于 Image Trailer 状态的变化，当 SFW 接收到新的镜像并且写入到 Slot2 后，会将 Slot2 的 Image Trailer 的 magic 置上。重启回 SBL 后，SBL 会读取到 Test 状态，此时会完成两个 Slot 的交换，此处 Slot1 的 Image Trailer 不会被复制到 Slot2，交换后原本 Slot2 的 Image Trailer 现在变为 Slot1 的 Image Trailer，现在的 slot2 的 Image Trailer 为全 0xFF。当交换完成后，SBL 会将 Slot1 的 Image Trailer 的 copy_done 标志位写为 0x01。交换完成后会进行签名的验证，验签成功后即会进行跳转。跳转到新的镜像后，在基础的任务运行一遍没有出现问题后，SFW 会将 Slot1 的 image_ok 标志位写为 0x01，如果在运行过程中出现了问题，没有写入这一位，程序复位后 SBL 就会读取到 Revert 状态，会重新交换 Slot1 与 Slot2 的镜像，最终完成镜像的回滚。

整个 Swap 模式下的 SBL 代码逻辑如 图 6 所示。



Swap 模式中两个 Slot 的交换没有使用到第三块 Flash 空间，所以固件镜像的大小不能超过 Slot 的大小减去两个 Sector 大小（一个是 Image Trailer 部分），采用的交换方法是，Slot1 的镜像向低地址平移一个 Sector，Slot2 的第一个 Sector 复制到 Slot1 的第一个 Sector，Slot1 的第二个 Sector 复制到 Slot2 的第一个 Sector，依次类推，完成两边的交换。

3.2 Remap 模式

Remap 模式的加入是基于 i.MXRT 部分芯片具有的 Remap 功能。使能该功能需要设置三个寄存器，分别是起始地址，终止地址以及偏移地址。使能该功能后，MCU 通过 AHB 总线对 Flash 的访问如果落在了之前设置的起始地址与终止地址之间，那么实际访问的物理地址会变为访问地址加上偏移地址的值。

这项功能天然适合用在 FOTA 的应用场景下。使能此功能后 Bootloader 不需要再进行镜像的交换，而是可以直接在不同的物理地址上使用相同的链接文件运行。相比于 Swap 模式，此功能大幅减少了 Flash 的擦写次数，延长了使用寿命，并且进行 FOTA 的时间也会大幅缩短。

Remap 模式下固件在 Flash 上的组织形式如 图 7 所示。

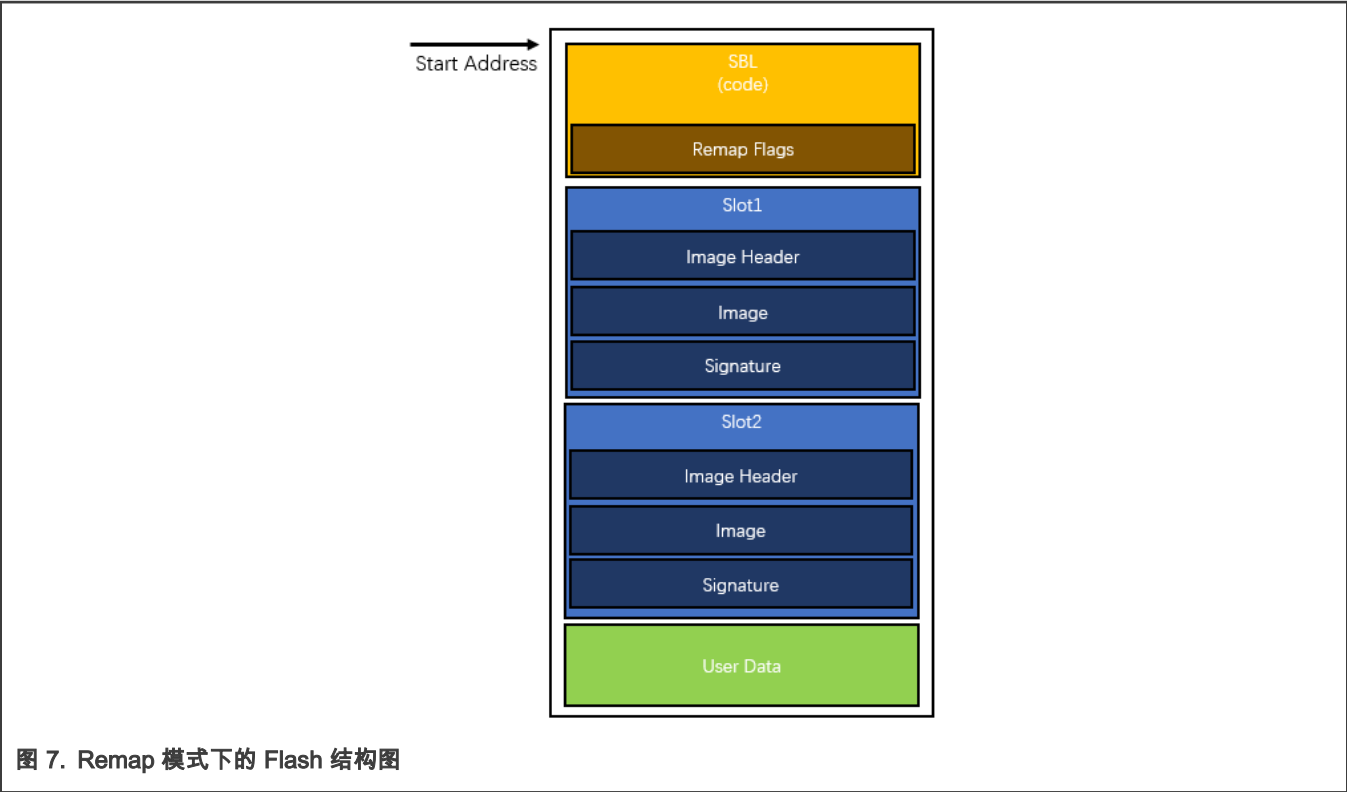


图 7. Remap 模式下的 Flash 结构图

与 Swap 模式的划分一样，分为 SBL，Slot1，Slot2 和 User Data 块。不同的是 Slot1 与 Slot2 中存储 FOTA 过程的 Flag 的 Image Trailer 部分被取消了，取代它的是在 SBL 区域最后的一块 Remap Flags 用于控制 FOTA 的过程。

Image Header 部分的作用和结构与 Swap 模式相同，可以参考 表 1。

Remap Flags 的结构如 表 3 所示。

表 3. Remapping flag 结构

Offset	Width (bytes)	Field	Description
0x00	1	Image_position	The current firmware position 0x01 0x02
0x01	7	Pad	Reserved
0x08	1	image_ok	Flag that control the OTA state
0x09	7	Pad	Reserved
0x10	16	magic	Image trailer magic Fixed value

Remap 式下，SBL 的逻辑跳转地址仍然被固定为 Slot1 的固件起始地址，与 Swap 不同的是，SBL 通过将 Remap 区域设置为 Slot1 的大小，offset 的值也设置为 Slot1 的大小后，从 AHB 总线上来看，对于落在 Slot1 区域内的访问实际所访问的物理地址会落在 Slot2 内相同的位置。也就是说，通过使能和失能 Remap 的功能，SBL 可以以相同的跳转位置，跳转到不同的物理地址执行固件镜像。

因此从 SFW 中所得到的新的固件镜像不会具有固定的存储位置，根据之前的固件所运行的 Slot，决定接收到的新的固件镜像存放在另一个 Slot 中。所以在上表的 Remap Flags 中有一个字节用于存放当前固件运行的位置信息。当 SFW 接收到新的固件镜像，并将它写入另一个 Slot 后，会置位 Remap Flags 中的 magic 字段以向 SBL 传递有新固件需要更新的信息。

在 Reset 回 SBL 后，SBL 的代码逻辑遵循 图 8 的流程。同样为了支持 i.MXRT 系列芯片中 ROM 的验签功能，SBL 先行读取 Remap Flags 确定当前的固件运行位置，并且根据 magic 字段的状态决定是否是否有新的固件需要更新。

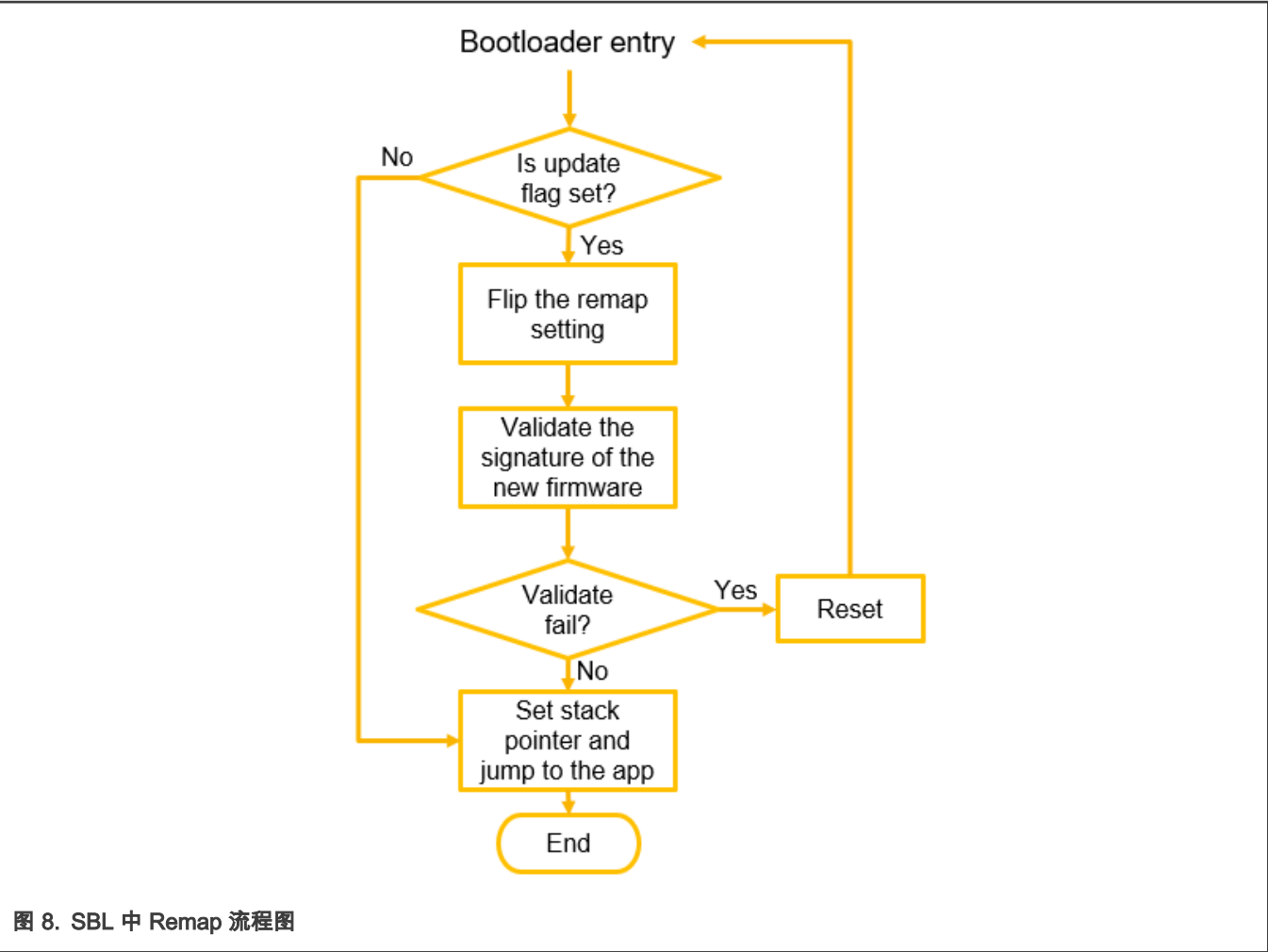


图 8. SBL 中 Remap 流程图

由于整个 Slot 进行了 Remap，并且 Flag 的内容有所变化，所以 SBL 与 SFW 项目将 Remap Flags 放在了 Flash 中 SBL 区的最末尾 32 字节。Remap 模式下同样支持固件镜像的回滚，也是由 Remap Flags 的状态来控制 FOTA 的过程，在 SBL 内对 Remap 模式下的 FOTA 也定义了四种状态，同样定义为 Test，Perm，Revert 和 None。

- Test 表示临时更新。
- Perm 表示永久更新。
- Revert 表示回滚。
- None 表示无更新。

SBL 通过读取 Remap Flags 的值决定了当前 FOTA 的状态，前三种状态下，Remap Flags 的设置如图 7 所示。

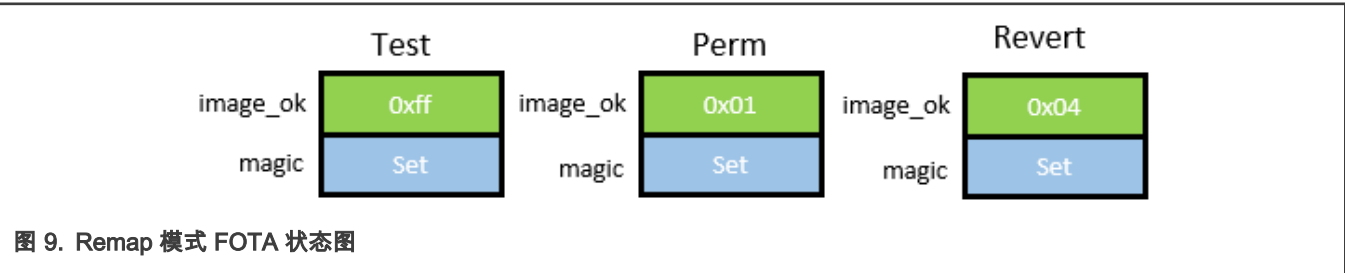


图 9. Remap 模式 FOTA 状态图

Remap 模式的回滚功能的实现依赖于 image_ok 字段的值，当 SFW 接收到新的镜像后，读取 Remap Flags 中的 image_position 字段，确定当前固件所在的 Slot，将新的固件镜像存入另一个 Slot，最终写入 magic 字段到 Remap Flags。重启回 SBL 后，SBL 会读取现固件镜像的位置，并且读取到 Test 状态。如果当前的固件镜像处在 Slot1，那么 SBL 会使能 Remap 功能，将 Slot1 的

地址映射到 Slot2 的物理地址；如果当前的固件镜像处在 Slot2，那么 SBL 会失能 Remap 功能，Slot1 的逻辑地址与物理地址保持一致。在验证新的固件镜像的签名无误后，SBL 会将 image_position 字段写为另一个 Slot，image_ok 字段写为 0x04，即将状态转换为 Revert，然后跳转到 Slot1 的逻辑地址。跳转到新的镜像后，在基础的任务运行一遍没有出现问题后，SFW 会将 Remap Flags 的 image_ok 字段写为 0x01，如果在运行过程中出现了问题，没有写入这一字段，程序复位后 SBL 就会读取到 Revert 状态，会重新使能或失能 Remap 功能，最终完成镜像的回滚。

整个 Remap 模式下的 SBL 代码逻辑如图 10 所示。

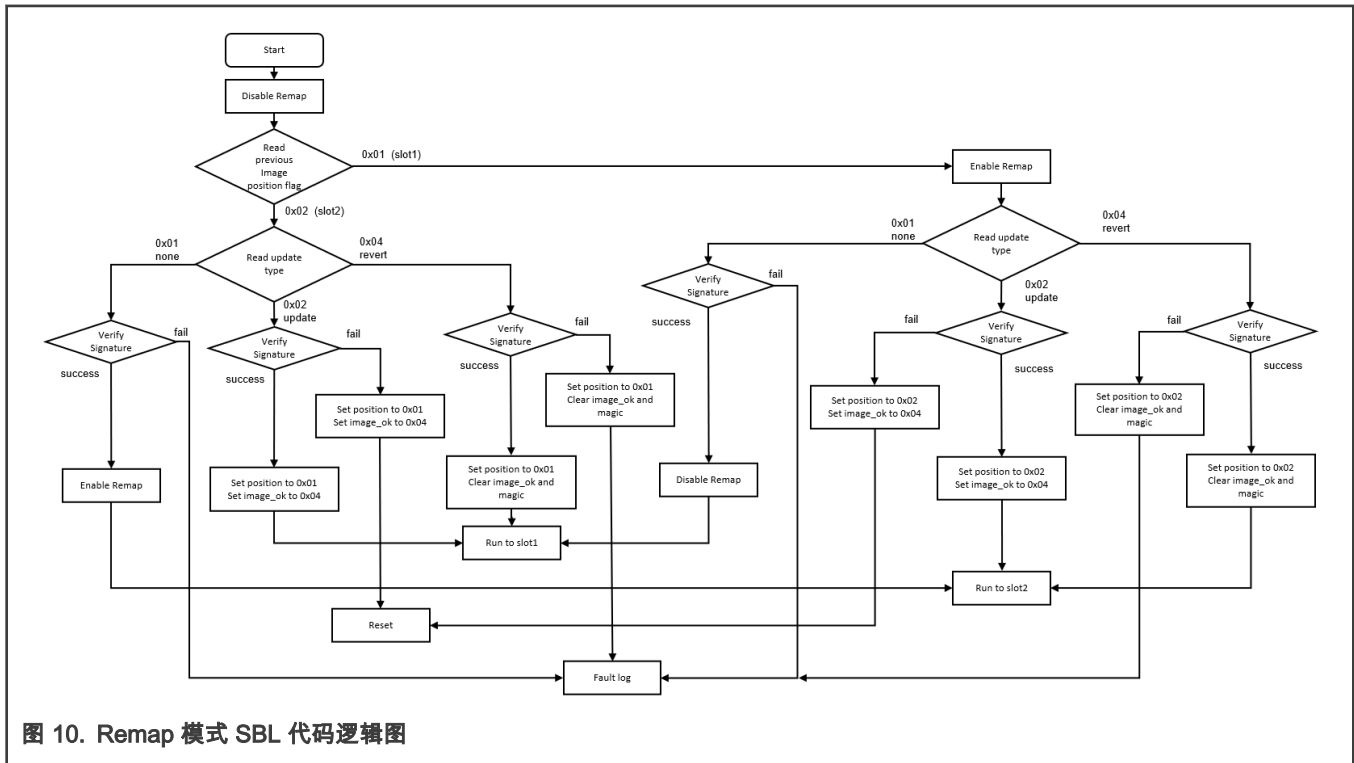


图 10. Remap 模式 SBL 代码逻辑图

整个判断过程当中，当升级的固件的签名没能通过验证时，SBL 会调用 NVIC_SystemReset 重启板卡，重新进入判断流程，得到 revert 状态，完成程序的回滚。

图 11 描述了一次不成功的升级后回滚，然后又进行一次成功升级的全部流程。

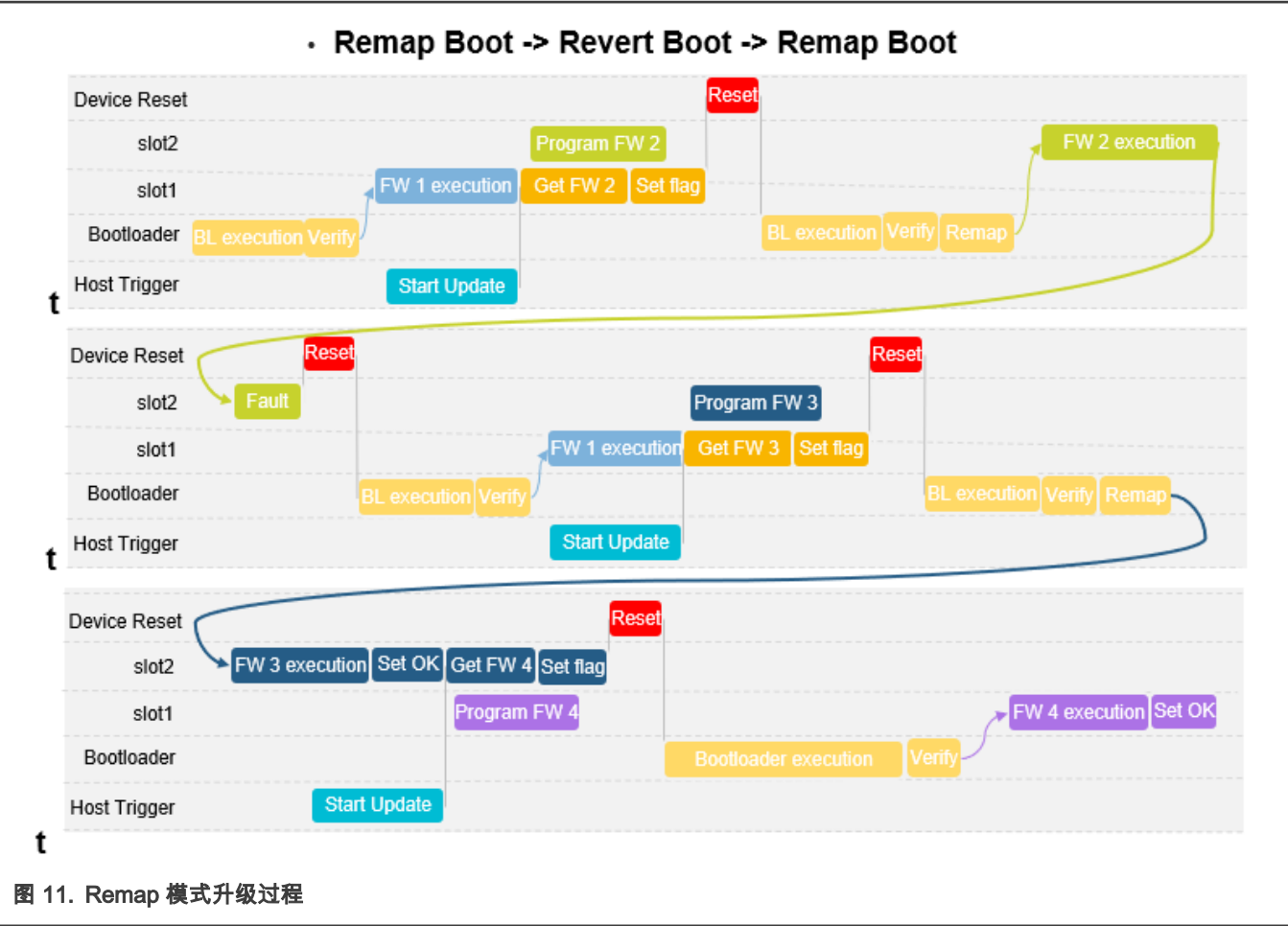


图 11. Remap 模式升级过程

4 参考资料

1. SBL Repository <https://github.com/NXPmicro/sbl>
2. SFW Repository <https://github.com/NXPmicro/sfw>
3. MCU-OTA SBL and SFW User Guide (document [MCUOTASBLSFWUG](#))

5 修订记录

版本	日期	说明
0	2021 年 11 月 20 日	初次发布

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Limited warranty and liability — Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

Right to make changes - NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Security — Customer understands that all NXP products may be subject to unidentified or documented vulnerabilities. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately. Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, ICODE, JCOP, LIFE, VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, AltiVec, CodeWarrior, ColdFire, ColdFire+, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, Tower, TurboLink, EdgeScale, EdgeLock, eIQ, and Immersive3D are trademarks of NXP B.V. All other product or service names are the property of their respective owners. AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org. M, M Mobileye and other Mobileye trademarks or logos appearing herein are trademarks of Mobileye Vision Technologies Ltd. in the United States, the EU and/or other jurisdictions.

© NXP B.V. 2021.

All rights reserved.

For more information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 2021 年 11 月 20 日

Document identifier: AN13460

