

引言

本用户手册介绍了如何开始使用X-CUBE-SBSFU STM32Cube扩展包。

安全启动（SB）和安全固件更新（SFU）解决方案允许使用新固件版本更新STM32微控制器内置程序、添加新功能并更正潜在问题。更新过程以安全方式执行，以防止未经授权的更新以及访问机密设备上的数据，如密码和固件加密密钥。

此外，安全启动（可信根服务）会检查并激活STM32安全机制，并在每次执行之前检查用户应用程序代码的真实性和完整性，确保无效或恶意代码无法运行。

安全固件更新应用程序会接收加密的固件映像，检查其真实性并对其进行解密，然后在安装之前检查代码的完整性。

安全固件更新应用程序支持：

- 两种操作模式：
 - 双映像模式，可实现安全映像编程，具有固件映像备份和回滚功能
 - 单映像模式，可最大限度地提高用户应用程序占用空间大小
- 三种加密方案，使用对称和非对称加密操作

X-CUBE-SBSFU补充了STM32Cube软件技术，使不同STM32微控制器之间更易移植。它随附了一个在NUCLEO-L476RG平台上运行的示例实现。

提供了X-CUBE-SBSFU作为参考代码，演示STM32安全保护机制的最新用法。这是OEM厂商根据其产品安全要求水平开发自己的SBSFU的起点。

X-CUBE-SBSFU归类为ECCN 5D002。



目录

1	一般信息	7
2	STM32Cube 概述	9
3	安全启动和安全固件升级（SBSFU）	10
3.1	产品安全介绍	10
3.2	安全启动	10
3.3	安全固件更新	11
3.4	加密操作	12
3.5	保护措施和安全策略	13
3.5.1	可抵御外部攻击的保护措施	14
3.5.2	可抵御内部攻击的保护措施	15
4	包说明	16
4.1	概述	16
4.2	架构	17
4.2.1	STM32CubeHAL	17
4.2.2	板级支持包（BSP）	17
4.2.3	加密库	18
4.2.4	安全引擎（SE）中间件	18
4.2.5	安全启动和安全固件升级（SBSFU）应用程序	19
4.2.6	用户应用程序	19
4.3	文件夹结构	20
4.4	API	21
4.5	利用IAR™工具链进行的应用程序编译过程	21
5	硬件和软件环境设置	23
5.1	硬件设置	23
5.2	软件设置	23
5.2.1	开发工具链和编译器	23
5.2.2	编程STM32微控制器的软件工具	23
5.2.3	终端仿真器	24
5.2.4	X-CUBE-SBSFU固件映像准备工具	24

6	逐步执行	25
6.1	STM32板子准备	25
6.2	应用程序编译	27
6.3	Tera Term连接	27
6.3.1	禁止 ST-LINK	27
6.3.2	Tera Term启动	28
6.3.3	Tera Term配置	28
6.3.4	欢迎屏幕显示	29
6.4	SBSFU应用程序执行	29
6.4.1	下载请求	29
6.4.2	发送固件	29
6.4.3	文件传输完成	31
6.4.4	系统重启	32
6.5	用户应用程序执行	32
6.5.1	下载新固件映像	32
6.5.2	测试保护	34
6.5.3	测试安全引擎用户代码	34
7	了解启动时的最后执行状态消息	35
附录A	安全引擎保护的环境.....	37
A.1	SE内核调用门机制	37
A.2	SE接口	39
附录B	双映像处理.....	41
B.1	元件和角色	41
B.2	映射定义	42
附录C	单映像处理.....	43
C.1	元件和角色	43
C.2	映射定义	43
附录D	加密方案处理.....	44
D.1	此软件包中包含的加密方案.....	44
D.2	非对称验证和对称加密方案.....	45
D.3	对称验证和加密方案	46

D.4	安全启动和安全固件更新流程	47
附录E	固件映像准备工具	49
E.1	工具位置	49
E.2	输入	49
E.3	输出	50
E.4	IDE集成	50
附录F	SBSFU应用程序状态机	51
F.1	双映像SBSFU	51
F.2	单映像SBSFU	51
F.3	SBSFU FSM状态	53
	版本历史	54

表格索引

表1. 缩略语列表 7

表2. 术语列表 8

表3. 密码方案比较 12

表4. 启动时的错误消息 35

表5. 双映像闪存组织 42

表6. 单映像闪存组织 43

表7. 加密方案列表 44

表8. 文档版本历史 54

表9. 中文文档版本历史 54

图片索引

图1.	安全启动可信根.....	11
图2.	典型的现场设备更新方案.....	11
图3.	保护功能概述.....	13
图4.	软件架构概述.....	17
图5.	项目文件结构.....	20
图6.	应用程序编译步骤.....	22
图7.	固件映像准备工具IDE集成.....	24
图8.	逐步执行.....	25
图9.	STM32CubeProgrammer连接菜单.....	26
图10.	STM32CubeProgrammer 选项字节.....	26
图11.	STM32CubeProgrammer擦除.....	27
图12.	Tera Term连接屏幕.....	28
图13.	Tera Term设置屏幕.....	28
图14.	SBSFU欢迎屏幕显示.....	29
图15.	SBSFU加密固件传输开始.....	30
图16.	正在进行SBSFU加密固件传输.....	30
图17.	加密固件传输后SBSFU重新启动.....	31
图18.	用户应用程序执行.....	32
图19.	通过用户应用程序下载加密的固件.....	33
图20.	用户应用程序测试保护菜单.....	34
图21.	防火墙调用门机制.....	38
图22.	安全引擎调用门机制.....	39
图23.	安全引擎接口.....	40
图24.	内部用户闪存交换.....	41
图25.	用户应用程序向量表.....	42
图26.	非对称验证和对称加密.....	45
图27.	对称验证和加密.....	46
图28.	SBSFU双映像启动流程.....	47
图29.	SBSFU单映像启动流程.....	48
图30.	双映像SBSFU应用程序状态图.....	51
图31.	单映像SBSFU应用程序状态图.....	52

1 一般信息

表 1 给出了相关的缩略语定义，帮助您更好地理解本文档。

表1. 缩略语列表

缩略语	说明
AAD	附加认证数据
AES	高级加密标准
CBC	AES 密码块链接
DMA	直接存储器访问
DSA	数字签名算法
ECC	椭圆曲线加密
ECCN	出口控制分类号
ECDSA	椭圆曲线数字签名算法
FSM	有限状态机
GCM	AES Galois/计数器模式
GUI	图形用户界面
HAL	硬件抽象层
IDE	集成开发环境
IV	初始化向量
IWDG	独立看门狗
FW	固件
FWALL	防火墙
MAC	消息认证码
MCU	微控制器单元
MPU	存储器保护单元
NONCE	仅使用一次的数值
PCROP	专有代码读保护
PEM	隐私增强邮件
RDP	读保护
SB	安全启动
SE	安全引擎
SFU	安全固件更新
SM	状态机
UART	通用异步收发器
UUID	通用唯一标识符
WRP	写保护

表 2给出了相关的术语定义，帮助您更好地理解本文档。

表2. 术语列表

缩略语	说明
固件映像	二进制映像（可执行文件），作为用户应用程序由设备来运行。
固件头文件	元数据包，描述要安装的固件映像。它包含固件信息和密码信息。
<i>sfb</i> 文件	包含固件头文件和固件映像的二进制文件。

X-CUBE-SBSFU安全启动和安全固件更新扩展包可在基于Arm^{®(a)} Cortex[®]-M处理器的STM3232位微控制器上运行。

arm

a. Arm是Arm Limited（或其子公司）在美国和/或其他地区的注册商标。



2 STM32Cube 概述

STM32Cube是什么？

STM32Cube™由意法半导体最初发起，通过减少开发工作量、时间和成本，让开发人员的生活更轻松。STM32Cube是STM32Cube™实现，涵盖了整个STM32产品组合。

STM32Cube包括：

- STM32CubeMX，一个图形化的软件配置工具，能够使用图形向导生成C初始化代码。
- 每个STM32微控制器系列都提供了全面的MCU封装（例如STM32F4系列的STM32CubeF4），包括：
 - STM32抽象层嵌入式软件STM32CubeHAL，确保在STM32各个产品之间实现最大限度的可移植性。
 - 底层API（LL）提供了一个专家级的快速轻量级层，它比HAL更靠近硬件。
 - 一套一致的中间件，比如RTOS、USB和图形
 - 提供了一套完整示例以及嵌入式软件工具。

此软件如何补充STM32Cube？

该软件基于STM32CubeHAL，即STM32微控制器的硬件抽象层。该软件包通过提供用于管理所有关键数据和操作（例如访问固件密钥等的加密操作）的中间件组件（安全引擎），扩展了STM32Cube。

该软件包包含有不同的示例应用程序，可提供完整的SBSFU解决方案：

- SE_CoreBin应用程序：提供包含所有“可信”代码的二进制文件。
- 安全启动和安全固件升级（SBSFU）应用程序：
 - 安全启动（可信根）
 - 通过UART虚拟COM进行本地下载
 - FW安装管理
- 用户应用程序：
 - 在双映像操作模式下下载新使用的固件
 - 提供了测试保护机制的示例

示例应用程序提供了双映像和单映像操作模式，能够配置为所支持的任意加密方案。

本用户手册介绍了软件包的典型用法：

- 基于NUCLEO-L476RG板
- 示例应用程序在双映像模式下运行，并配置了非对称身份验证和对称固件加密

有关配置选项和单映像操作模式的更多信息，请参阅本文档的附录。

3 安全启动和安全固件升级（SBSFU）

3.1 产品安全介绍

现场部署的设备在不受信任的环境中运行，因此会受到威胁和攻击。为了减轻受攻击风险，我们的目标是只在设备上运行可靠的固件。事实上，更新固件映像来修复错误，或引入新功能或对策，这些对于连接的设备来说是常见的事情，但如果不以安全方式来执行这些操作，它就会很容易受到攻击。

其后果可能是破坏性的，如固件克隆、恶意软件下载或设备损坏。必须设计安全解决方案来保护敏感数据（甚至可能是固件本身）和关键操作。

典型的对策基于加密技术（带有相关密钥）和内存保护：

- 加密可确保固件传输期间的完整性（确保数据未被破坏）、身份验证（确保某个实体确实符合其声明）以及机密性（确保只有经过授权的用户才能读取敏感数据）。
- 内存保护机制可以防止外部攻击（例如，通过JTAG物理访问设备）以及来自其他嵌入式进程的内部攻击。

以下章节介绍实现机密性、完整性和身份验证服务的解决方案，以解决IoT终端节点设备最常见的威胁。

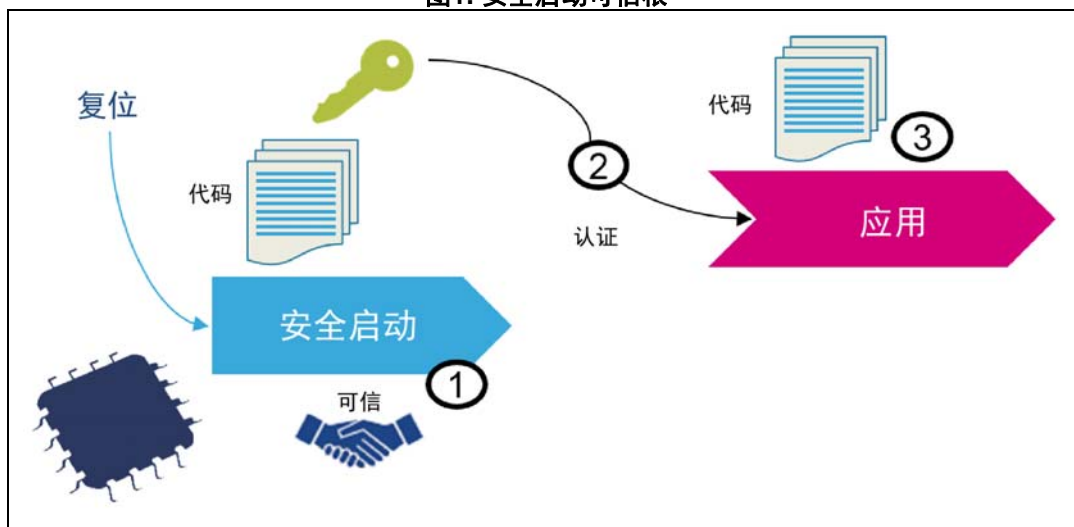
3.2 安全启动

安全启动（SB）确保所执行的用户应用程序映像的完整性和可靠性：使用加密检查，防止运行未经授权或恶意修改的软件。安全启动过程实现可信根（参见图 1）：从该可信组件（1）开始，其他每个组件在其执行之前（3）都要经过认证（2）。

对**完整性**进行验证，以确保即将执行的映像未被破坏或恶意修改。

可靠性检查旨在验证固件映像是来自可信且已知的源，以防止未经授权的实体安装及执行代码。

图1. 安全启动可信根



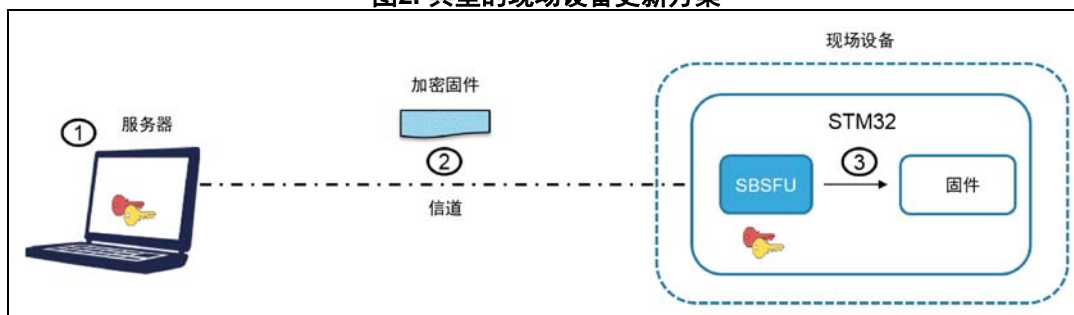
3.3 安全固件更新

安全固件更新（SFU）实现了安全的现场固件更新，可以安全地将新固件映像下载到设备。

如 图 2 中所示，通常有两个实体参与固件更新过程：

- 服务器
 - OEM制造商服务器 / Web服务
 - 存储设备固件的新版本
 - 与设备通信，如果可用，则以加密形式发送该新映像版本
- 设备
 - 现场部署
 - 嵌入了一个运行固件更新过程的代码。
 - 与服务器通信并接收新的固件映像。
 - 验证、解密并安装新固件映像，然后执行它。

图2. 典型的现场设备更新方案



固件更新通过以下步骤进行：

- 1. 如果需要更新固件，则创建一个新的加密固件映像并将其存储在服务器中。
- 2. 新的加密固件映像通过不受信任的通道发送到现场部署的设备。
- 3. 下载、检查并安装新映像。

固件更新容易受到[第 3.1 节：产品安全介绍](#)中所示风险的影响：加密技术用来确保机密性、完整性和身份验证。

实现**机密性**以保护固件映像，这可能是制造商的关键资产。通过不受信任的通道发送的固件映像被加密，因此只有具有密钥访问权的设备才能解密固件包。

验证**完整性**以确保接收的映像没有损坏。

可靠性检查旨在验证固件映像是来自可信且已知的源，以防止未经授权的实体安装及执行代码。

3.4 加密操作

X-CUBE-SBSFU STM32Cube扩展包提供了同时使用非对称和对称加密的三种加密方案。
默认的加密方案演示了用于固件验证的ECDSA非对称加密和用于固件解密的AES-CBC对称加密。利用非对称加密技术，固件验证可以通过公钥操作执行，因此设备中不需要任何保密信息。

X-CUBE-SBSFU扩展包中提供的其他加密方案是：

- 用于固件验证（无需固件加密）的ECDSA非对称加密技术。
- 或者用于固件验证和解密的AES-GCM对称加密技术。

[表 3](#)介绍了每种密码方案相关的各种安全特性。

表3. 密码方案比较

特性	非对称 使用AES加密	非对称 不加密	对称 (AES GCM)
机密性	AES CBC 加密 (FW二进制文件)	无：用户FW处于明文格式。	AES GCM 加密 (FW二进制文件)
完整性	SHA256 (FW头文件和FW二进制文件)		AES GCM 标记 (FW头文件和FW二进制文件)
真实性	FW头文件的SHA256是ECDSA签名的。 FW二进制文件的SHA256存储在FW头文件中。		
设备中的密钥	AES CBC私钥 (秘密) ECDSA公钥	ECDSA公钥	AES GCM私钥 (秘密)

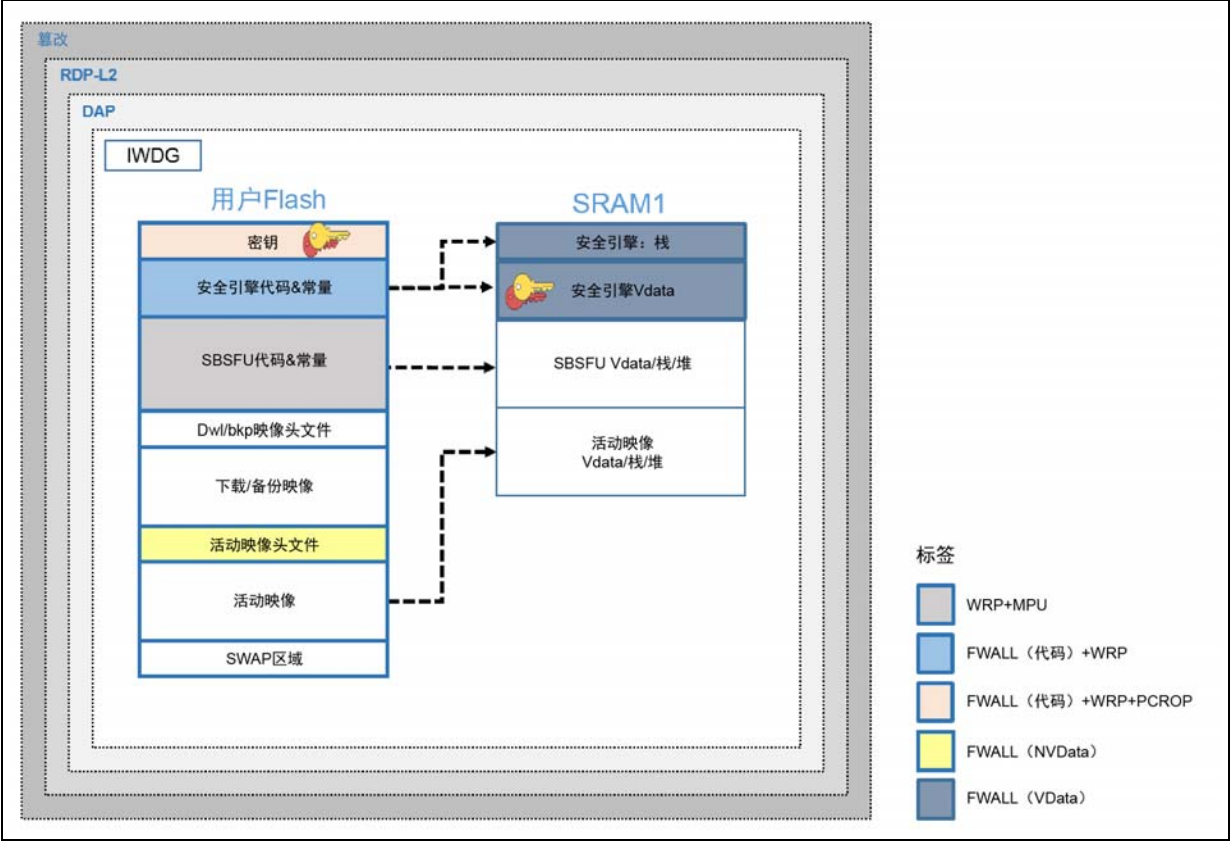


3.5 保护措施和安全策略

加密保证了完整性、真实性和机密性。但是，单独使用加密技术是不够的：需要一整套措施和系统级策略来保护关键操作和敏感数据（例如密钥），以及执行流程，以便抵御可能的攻击。

图 3说明了如何保护系统、代码和数据的X-CUBE-SBSFU应用程序示例。

图3. 保护功能概述



3.5.1 可抵御外部攻击的保护措施

外部攻击是指由外部工具触发的攻击，如调试器或探测器尝试访问设备时。在SBSFU应用示例中，RDP、tamper、DAP和IWDG保护用来保护产品免受外部攻击：

- **RDP（读保护）**：读保护级别2是强制性的，以实现最高级别的保护并实现可信根：
 - 禁止通过JTAG HW接口对SRAM1、SRAM2和Flash进行外部访问。这样可以防止改变SBSFU代码并因此来挖掘可信根的攻击。
 - 选项字节不能更改。这意味着其他保护措施如WRP和PCROP不能再进行更改。

注意 - 由于以下原因，不建议使用RDP级别1：

1. 无法确保安全启动/可信根（单入口点和不可变代码），因为在RDP L1情况下可以修改选项字节（WRP）。
2. 设备内部闪存可以利用新的FW进行完全重新编程（在通过RDP L0回归进行闪存整体擦除之后），无需任何安全措施。
3. 系统复位时，通过JTAG硬件接口连接调试器，可以访问受防火墙保护的RAM存储器中的秘密信息。

如果客户产品上不能进行JTAG硬件接口访问，并且客户使用信任可靠的用户应用程序代码，则上述风险无效。

- **Tamper**：防篡改保护用于检测设备上的物理篡改行为并采取相应的应对措施。在篡改检测的情况下，SBSFU应用示例强制重启。
- **DAP（调试访问端口）**：DAP保护在于可以停用DAP（调试访问端口）。一旦停用，JTAG引脚不再连接到STM32内部总线。使用RDP Level 2可自动禁用DAP。
- **IWDG（独立看门狗）**：IWDG是一个自由运行的减量计数器。一旦开始运行，它就不能再停止。在引起重置之前必须对其定期刷新。该机制能够控制SBSFU执行持续时间。

3.5.2 可抵御内部攻击的保护措施

内部攻击是指由STM32中运行的代码触发的攻击。攻击可能是由恶意固件利用错误或安全漏洞导致的，也可能由不需要的操作引起。在SBSFU应用示例中，WRP、防火墙、PCROP和MPU保护可防止产品受到内部攻击：

- **FWALL**（防火墙）：对防火墙进行配置以保护代码、易失性数据和非易失性数据。受保护的代码可以通过单入口点进行访问（调用门机制在[附录A](#)中有说明）。任何试图跳过并尝试不通过入口点而执行代码段中函数的操作都会导致系统重置。
- **PCROP**（专有代码读出保护）：通过对其施加PCROP保护，Flash的一个扇区可被定义为只执行区：在读取或写入时不能访问此扇区。作为一个只执行区域，只有当它“嵌入”一段代码时，密钥才受到PCROP保护：执行此代码会将该密钥移动到RAM中的特定指针。由于它在防火墙之后，因此不可能从外部执行。
- **WRP**（写保护）：写保护用于保护可信代码免受外部攻击或甚至内部修改，如对关键代码/数据进行不需要的写入/擦除操作。
- **MPU**（存储器保护单元）：通过将闪存和SRAM的存储器映射划分为具有各自访问权限的区域，MPU可使嵌入式系统更加稳健。在SBSFU应用实例中，配置MPU是为了确保在SBSFU代码执行期间，不从任何存储器执行其他代码。当SBSFU应用程序执行UserApp时，MPU被解除配置。

4 包说明

本节详细介绍了X-CUBE-SBSFU包的内容和使用方法。

4.1 概述

X-CUBE-SBSFU是用于STM32微控制器的软件包。

它提供了构建安全启动和安全固件更新应用的完整解决方案：

- 可支持对称和非对称加密方法，利用用于解密或验证的AES-GCM、AES-CBC和ECDSA算法，或用来同时解密和验证的X-CUBE-CRYPTOLIB。
- 两种操作模式：
 - 双映像模式，可实现安全映像编程，具有固件映像备份和回滚功能
 - 单映像模式，可最大限度地提高用户应用程序占用空间大小
- 将安全外设和机制整合集成，以实现SBSFU可信根。RDP、WRP、PCROP、防火墙、MPU、tamper和IWDG结合使用，实现最高的安全级别。
- 使用安全引擎（SE）模块作为中间件的一部分，以提供管理所有关键数据和操作（如安全密钥存储，加密操作等）的受保护环境。
- 可使用用户应用程序示例源代码。
- 可使用固件映像准备工具，它以可执行代码和源码的方式提供。

X-CUBE-SBSFU被移植到STM32L4系列上。

该软件包包含了应用示例，开发人员可以将其用于试验代码。

该软件包以zip文档的形式提供，其中包含源代码。

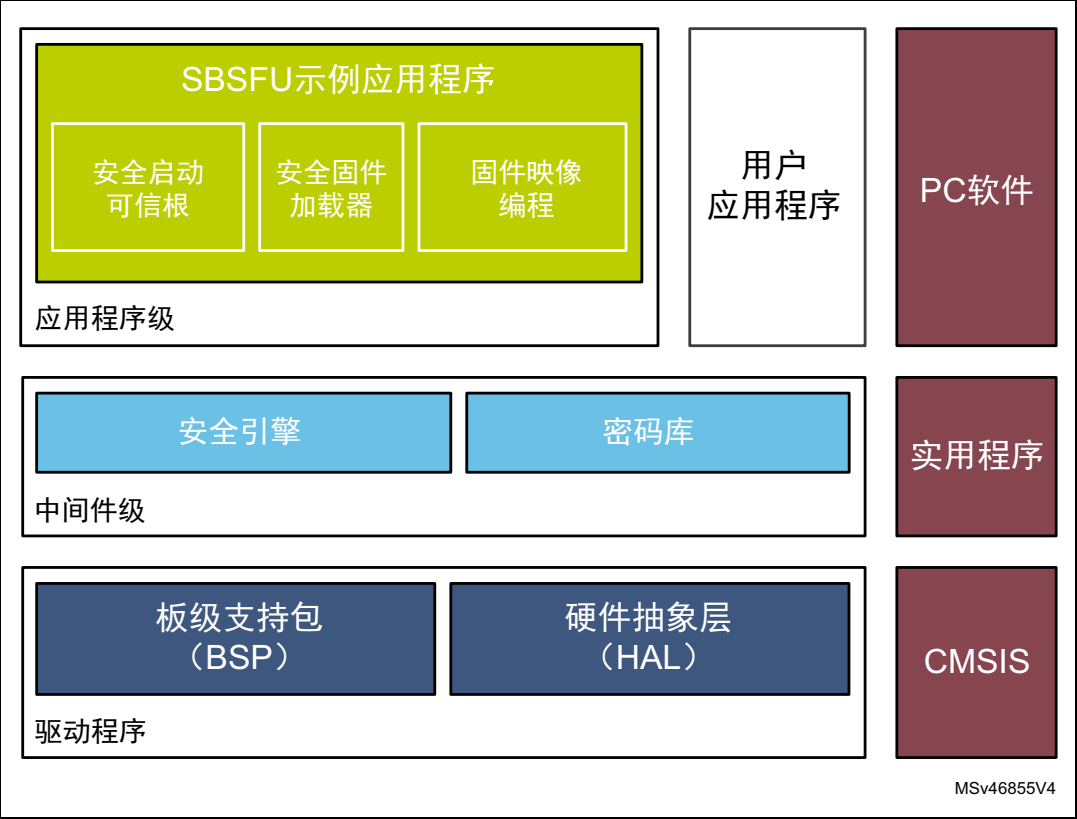
可支持以下集成开发环境：

- 用于Arm®（EWARM）的IAR Embedded Workbench®。
- Keil®微控制器开发套件（MDK-ARM）
- 用于STM32的系统工作台

4.2 架构

本节描述X-CUBE-SBSFU包的软件组成部分，如图 4 中所示。

图4. 软件架构概述



4.2.1 STM32CubeHAL

HAL驱动层提供通用的多实例简单API组（应用程序编程接口），以便与上层（应用、库和堆栈）交互。它由通用和扩展API构成。它直接围绕通用架构构建，允许在其基础上构建层，例如中间件层，实现了它的功能又无需依赖给定微控制器单元（MCU）的特定硬件配置。

此结构可提高库代码的可复用性，并确保可向其他设备轻松移植。

4.2.2 板级支持包（BSP）

除MCU之外，软件包需支持STM32板上的外设。板级支持包（BSP）中包含此软件。这是一个有限的API集，为板特有的某些外设（例如LED和用户按钮等）提供编程接口。

4.2.3 加密库

X-CUBE-CRYPTOLIB可支持对称和非对称密钥方法（AES-GCM, AES-CBC, ECDSA）以及用于解密和验证的哈希运算（SHA256）。使用软件加密函数可以避免将密钥存储在不受防火墙保护的HW Crypto IP寄存器中。

4.2.4 安全引擎（SE）中间件

安全引擎中间件提供了用于管理所有关键数据和操作（例如访问密钥的操作，等等）的受保护环境。受保护的代码和数据可通过一个单入口点（调用门机制）进行访问，因此不能在不经单入口点的情况下运行或访问任何SE代码或数据，否则会产生系统重置（请参阅[附录A](#)获取有关调用门机制的详细信息）。

注： *根据用户应用程序的需要，安全引擎关键操作可以扩展其他功能。只有可信代码才能被添加到安全引擎环境中，因为它有访问秘密信息的权限。*

4.2.5 安全启动和安全固件升级（SBSFU）应用程序

安全启动（可信根）

- 检查并应用STM32平台的安全机制以保护关键操作和机密免受攻击
- 如果在两次开机之间连续执行了3次*Boot On Error*启动序列，则检查从严重故障启动的次数并回滚到上一个有效映像（如果有）
- 每次执行前认证并核验用户应用程序

通过UART虚拟COM进行本地下载

- 检测固件下载请求
- 使用Ymodem协议和Tera Term工具，通过UART虚拟COM，在STM32闪存中下载新的加密固件映像（头文件+加密固件）

FW安装管理

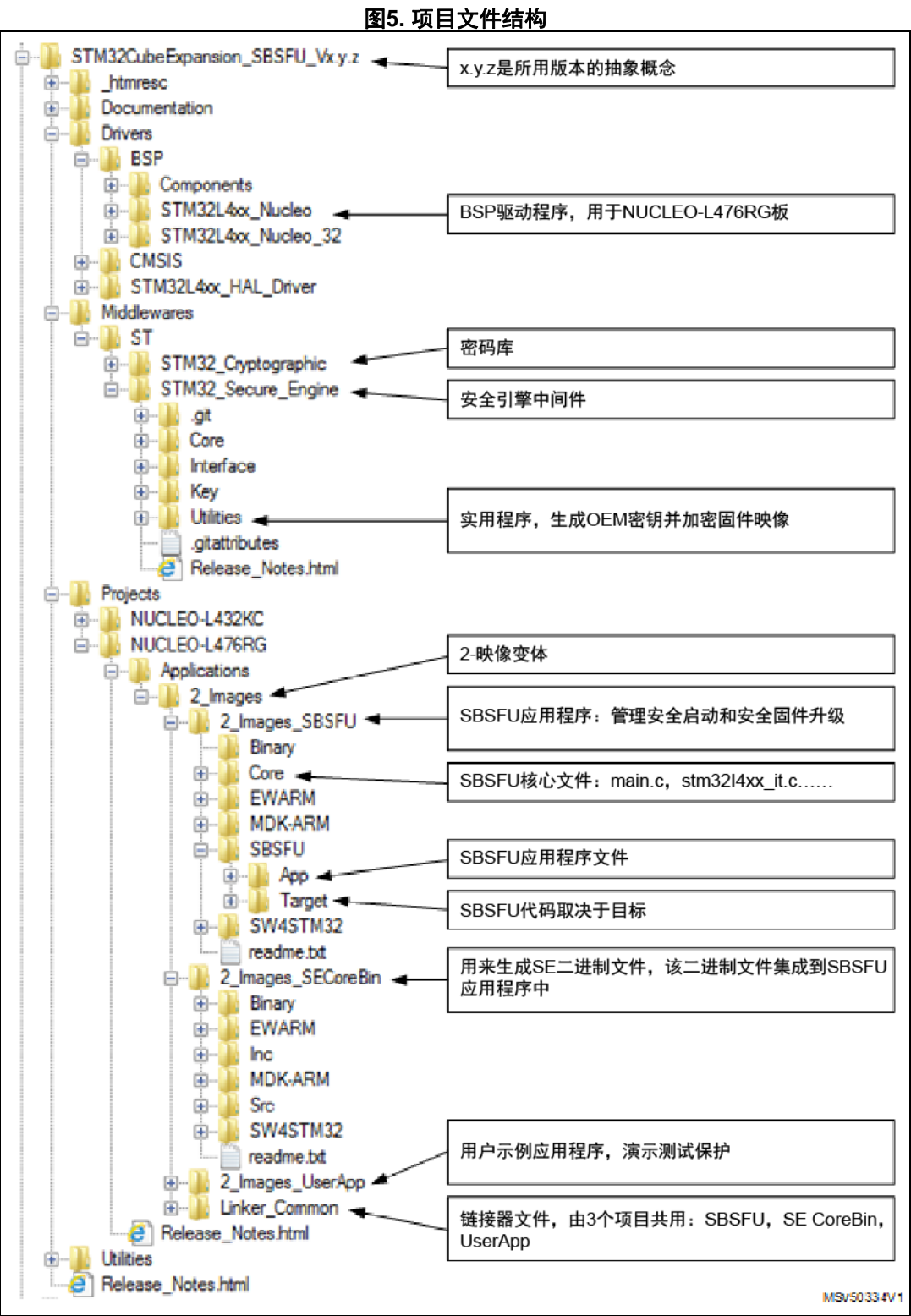
- 检测要安装的新固件版本
 - 通过UART接口从本地下载服务
 - 通过用户应用程序下载（仅适用于双映像变体）
- 安全FW升级：
 - 身份验证和完整性检查
 - FW 解密
 - FW安装
- 支持单映像，以最大化用户应用程序空间
- 支持双映像，以实现安全映像编程
 - 回滚管理：如果在固件安装期间发生错误，则重新安装先前的有效固件。
 - 多固件映像管理：处理存储在内部STM32闪存中的两个固件映像（UserApp1映像和UserApp2映像）。SWAP区域用来限制固件安装或回滚过程中所需的存储开销（请参阅[附录B](#)获取关于多映像管理的详细信息）。

4.2.6 用户应用程序

- 提供了一个利用Ymodem协议通过UART下载用户应用程序的示例（通过空中下载机制，例如BLE、Wi-Fi®或其他，可以在用户应用程序中实现，但未作为X-CUBE-SBSFU应用示例中的示例提供）。
- 提供了测试保护机制的示例。
- 提供了使用由SE导出的某些功能的示例，例如获取有关当前固件映像的信息。
- 用户应用程序可以使用SBSFU应用程序的本地下载功能或由用户应用程序本身管理的下载功能进行更新（加密固件由用户应用程序下载，安全安装由SBSFU应用程序管理）。

4.3 文件夹结构


图 5中显示了文件结构的顶层视图。



4.4 API

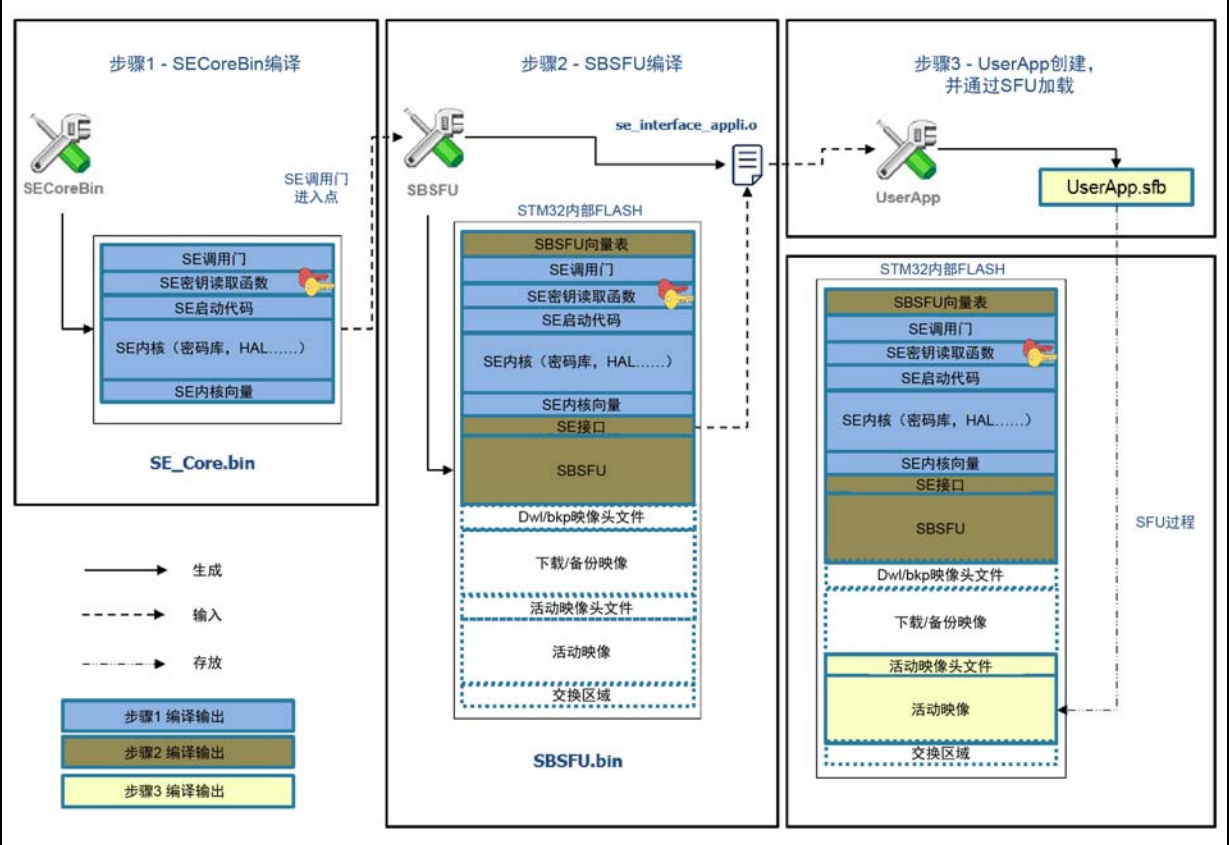
有关可供用户使用的API的详细技术信息，在软件包的*Documentation*文件夹中已编译HTML文件中提供，该软件包中介绍了所有函数和参数。

4.5 利用IAR™工具链进行的应用程序编译过程

 6概述了构建应用程序以及演示安全启动和安全固件更新所需的步骤：

- 步骤1：准备内核二进制文件
创建安全引擎内核二进制文件需要执行此步骤，该代码中包括了映射到防火墙代码段的所有“可信”代码和密钥。SE Callgate函数被指定为该二进制文件的入口点。步骤2中，该二进制文件链接到SBSFU代码。
- 步骤2：SBSFU
此步骤编译SBSFU源代码，实现状态机并配置保护。此外，它将该代码与步骤1中生成的SECore二进制文件链接起来，生成单独的包含SE可信代码的SBSFU二进制文件。它还生成一个包含符号的文件，以供用户应用程序调用SE接口方法，这是一组包装单个SE调用门API的用户友好API。
- 步骤3：用户应用程序示例
它会生成：
 - 用户应用程序二进制文件，使用安全固件更新程序（*UserApp.sfb*）可将其上传到设备。
 - 一个二进制文件，连接了SBSFU二进制文件、明文格式的用户应用程序二进制文件 and 对应FW头文件。当该二进制文件通过烧录工具烧录到设备中时，正确安置这三个元素，以便使SBSFU和用户应用程序运行。因此，SBSFU不需要FW安装程序来启动和引导用户应用程序。这是使用单个闪存级测试用户应用程序的便捷方式。

图6. 应用程序编译步骤



5 硬件和软件环境设置

本节说明了硬件和软件设置过程。

5.1 硬件设置

要设置硬件，必须利用USB线缆将某种支持的板（如 [第 4.1 节：概述第 16 页](#) 中所介绍）插入个人计算机。与PC的连接允许用户：

- 烧写板
- 通过UART控制台与板进行交互
- 在禁用保护时进行调试

5.2 软件设置

本节为开发人员列出了设置SDK、运行示例场景以及自定义应用的最低要求。

5.2.1 开发工具链和编译器

选择STM32Cube扩展包支持的其中一个集成开发环境。

考虑所选IDE供应商提供的系统要求和设置信息。

5.2.2 编程STM32微控制器的软件工具

ST-LINK Utility

STM32 ST-LINK Utility (STSW-LINK004) 是用于编程STM32微控制器的全功能软件接口。它为读取、写入和验证存储设备提供了一个易用高效的环境。

参考 www.st.com 上的STSW-LINK004 STM32 ST-LINK实用软件

注意： 确保使用了最新版本的ST-LINK（V2.J27或更高版本）。

STM32CubeProgrammer

STM32CubeProgrammer (STM32CubeProg) 是一款用于编程STM32微控制器的全功能多操作系统软件工具。它通过调试接口（JTAG和SWD）和bootloader接口（UART和USB）提供了一个易用高效的环境，用于读取、写入和验证设备内存。

STM32CubeProgrammer提供了广泛的功能，可编程STM32微控制器内部存储器（如Flash、RAM和OTP）以及外部存储器。STM32CubeProgrammer还允许选择编程和上传、编程内容验证以及通过脚本自动编程微控制器。

STM32CubeProgrammer提供了GUI（图形用户界面）和CLI（命令行界面）版本。

参考 www.st.com 上的STM32CubeProg STM32CubeProgrammer软件

5.2.3 终端仿真器

运行演示需要终端仿真器软件。

本文档中的示例基于Tera Term，这是一款开源免费软件终端仿真器，可从<https://osdn.net/projects/ttssh2/>网页下载。可以使用任何其他类似的工具（需要Ymodem协议支持）。

5.2.4 X-CUBE-SBSFU固件映像准备工具

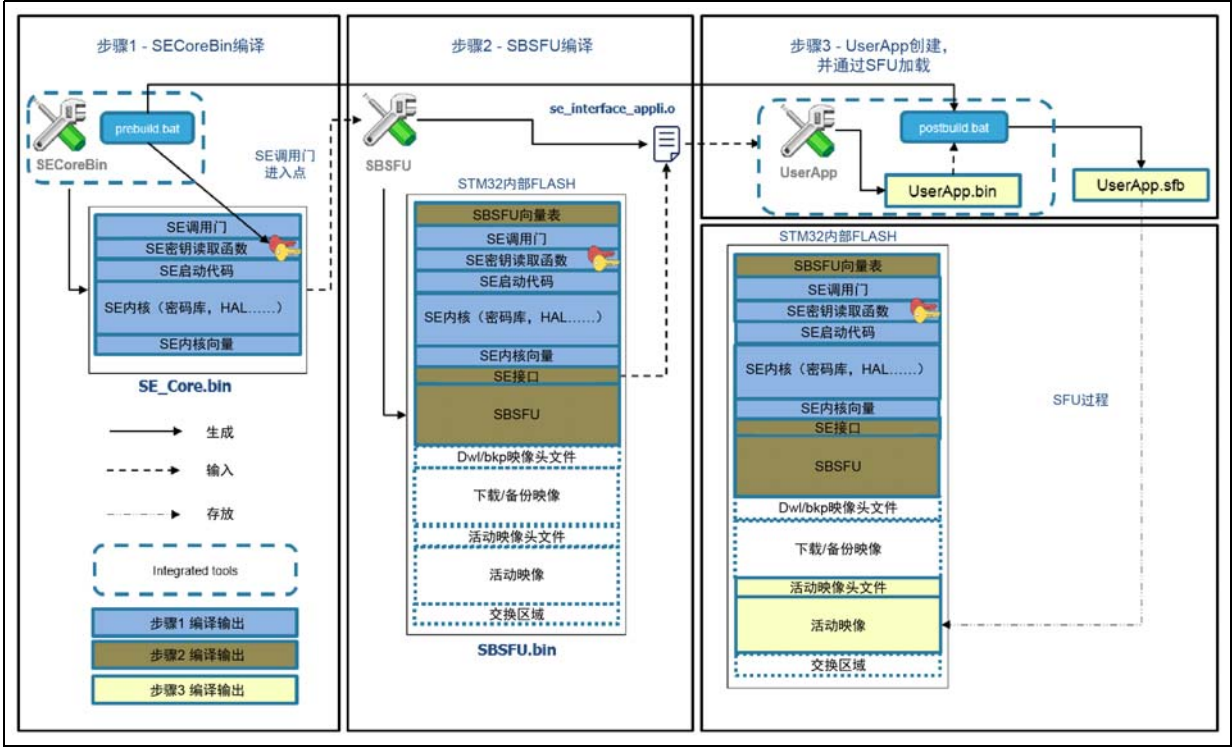
用于STM32Cube的X-CUBE-SBSFU扩展软件包随*prepareimage*工具一起提供，用于处理加密密钥和固件映像准备。

*prepareimage*工具以两种格式提供：

- Windows®可执行文件：需要标准Windows®命令解释器
- Python™脚本：Python™解释器以及
*Middlewares\ST\STM32_Secure_Engine\Utilities\KeysAndImages\readme.txt*中列出的组件是必需的

Windows®可执行文件完全集成在支持的IDE和编译过程中，如 图 7 中所示。

图7. 固件映像准备工具IDE集成



有关准备工具的更多信息，请参阅 附录E：固件映像准备工具 第 49 页。

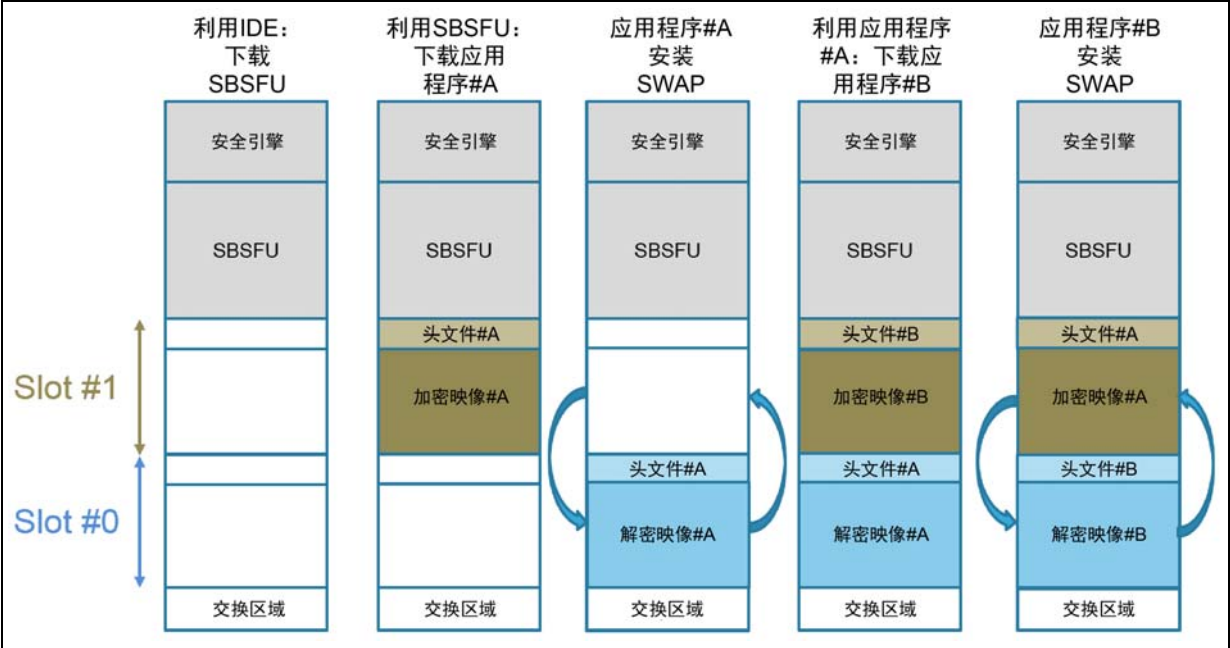
6 逐步执行

以下步骤描述了具有默认加密方案的双映像SBSFU方案，其进一步说明如 图 8 中所示：

1. 下载SBSFU应用程序
2. SBSFU正在运行：下载UserApp #A
3. UserApp #A已安装
4. UserApp#A正在运行，下载UserApp #B
5. UserApp #B安装并运行

UserApp#A和UserApp#B二进制文件基于用户应用程序示例项目而生成。将应用程序定义为 #A或#B是通过更改应用程序main.c中声明的UserAppId变量的值来完成的。

图8. 逐步执行



6.1 STM32板子准备

目标选项字节设置如下：

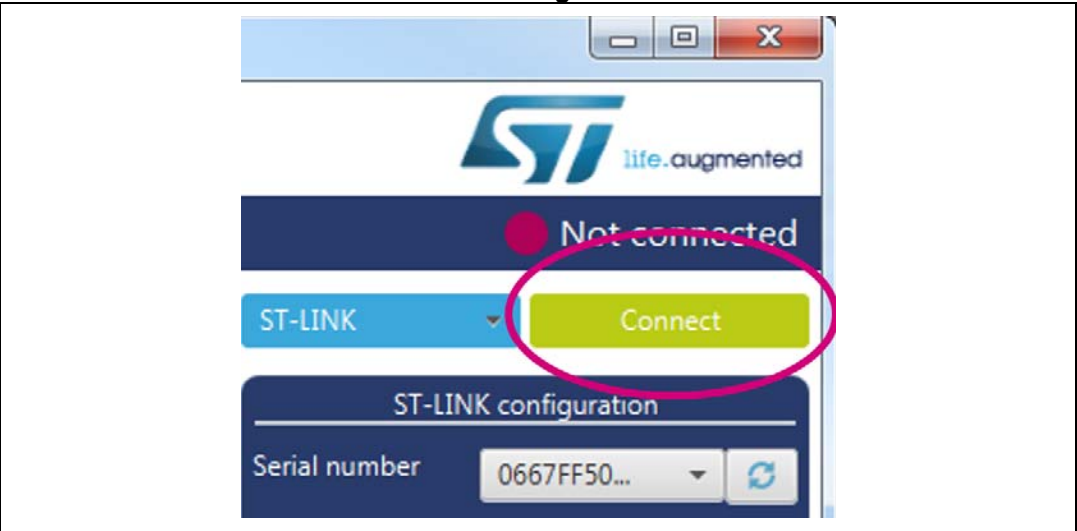
- 设置RDP Level 0
- 所有闪存页面上禁用写保护
- 禁用PCROP保护^(a)
- 芯片被擦除^(a)

选项字节设置由STM32CubeProgrammer通过以下四个步骤进行验证：

a. 这在从RDP级别1切换到RDP级别0时自动完成。

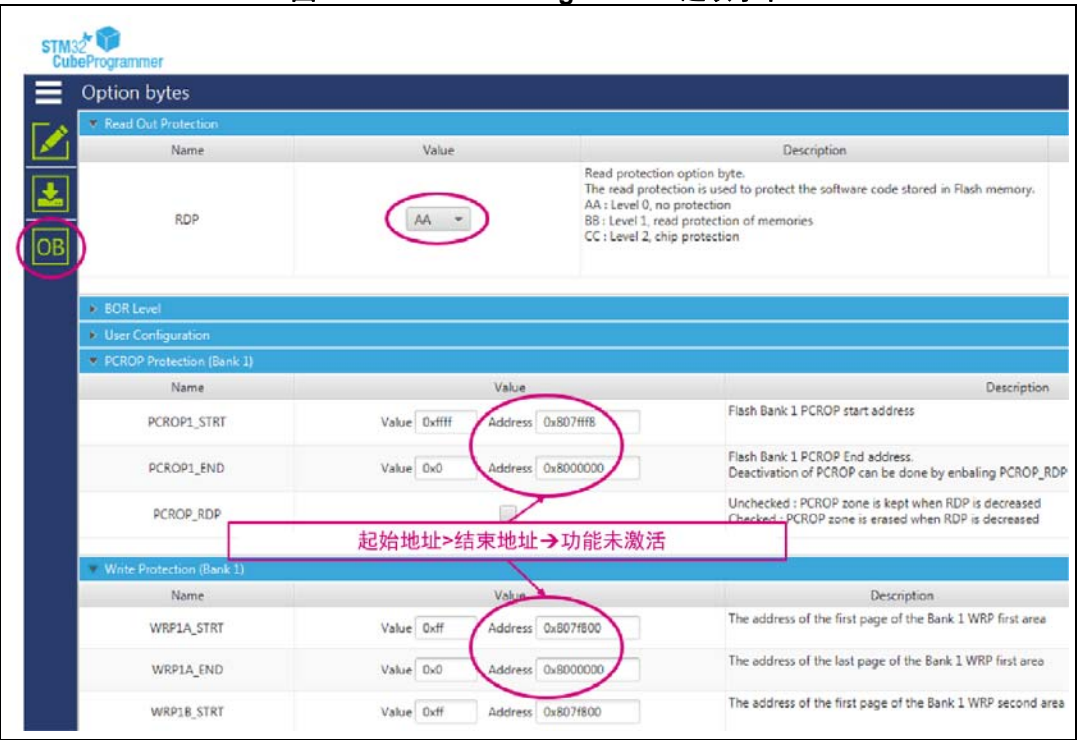
1. 连接：菜单目标 / 连接（参考图 9）

图9. STM32CubeProgrammer连接菜单



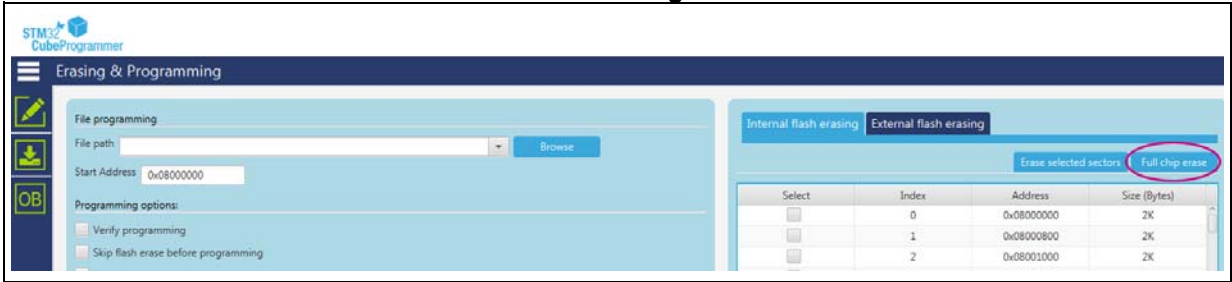
2. 选项字节设置：菜单目标 / 选项字节（参考图 10）

图10. STM32CubeProgrammer 选项字节



3. 擦除芯片：菜单目标 / 擦除芯片

图11. STM32CubeProgrammer擦除



4. 断开：菜单目标 / 断开

6.2 应用程序编译

使用选定的工具链（IAR、Keil或System Workbench）重新构建所有项目，如[第 4.5节：利用IAR™工具链进行的应用程序编译过程第 21页](#)中所述。

可将SB SFU项目软件下载到目标，无需启动调试会话（由SBSFU管理的安全保护会禁止JTAG连接，因为它被解释为外部攻击）。

6.3 Tera Term连接

Tera Term连接通过依次应用从[第 6.3.1章](#)到[第 6.3.4章](#)所述的步骤来实现。

6.3.1 禁止 ST-LINK

SBSFU管理的安全机制会禁止JTAG连接（解释为外部攻击）。必须禁用ST-LINK才能建立Tera Term连接。以下程序适用于从ST-LINK固件版本VJ26M15开始的更高版本^(a)：

- 在烧录SBSFU后重启板子（拔下/插入USB电缆）。
- SBSFU应用程序以开发模式启动并配置安全机制。在产品模式下，仅检查安全机制是否处于正确值。
- 再一次重启板子（拔下/插入USB电缆）：SBSFU应用程序启动，所配置安全措施打开，可以连接Tera Term。

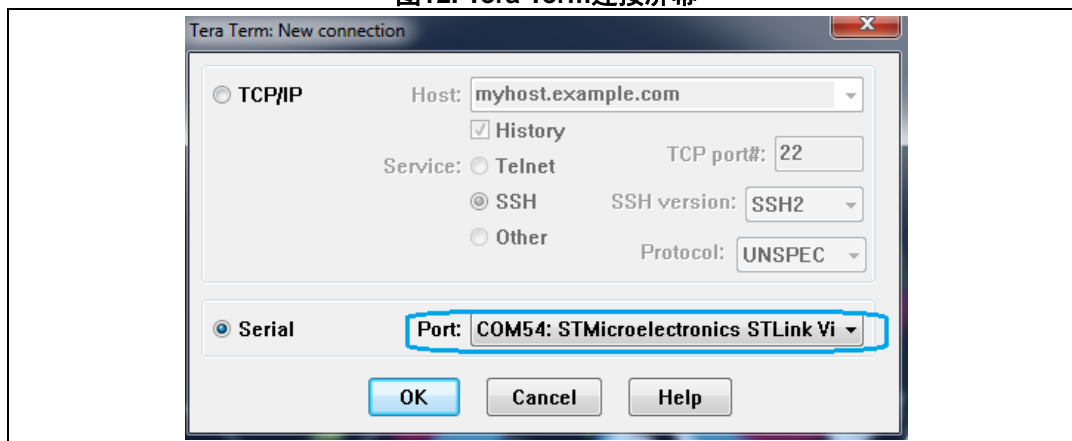
a. 确保嵌入在板上的ST-LINK调试器/编程器运行正确的固件版本（VJ26M15或更高版本）。如果版本情况不是这样，请首先升级此固件。

6.3.2 Tera Term启动

Tera Term启动要求端口选择为COMxx: *STMicroelectronics STLink*虚拟COM端口。

图 12说明了基于端口COM54选择的示例。

图12. Tera Term连接屏幕

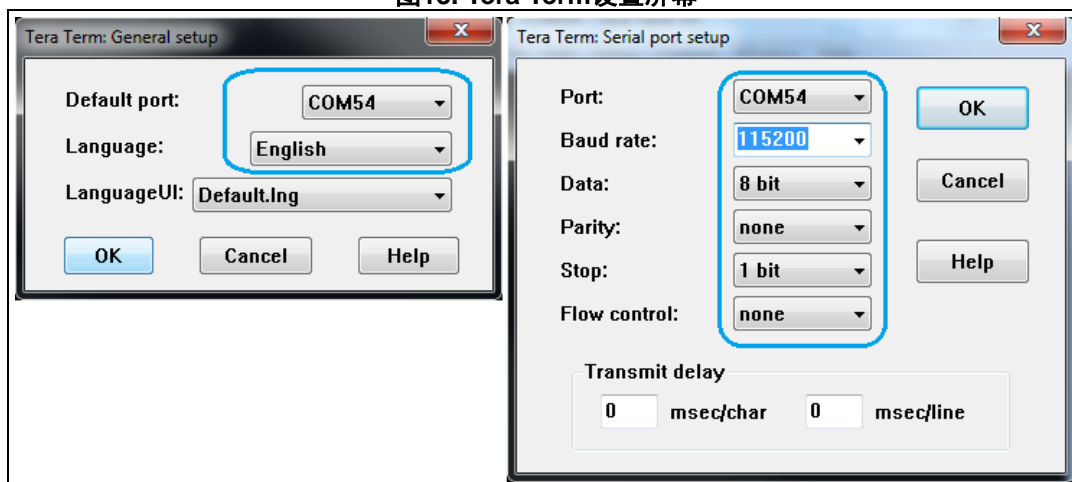


6.3.3 Tera Term配置

Tera Term配置通过*General*和*Serial port*设置菜单来实现。

图 12说明了*General setup*和*Serial port setup*菜单。

图13. Tera Term设置屏幕



利用*Menu Setup / Save Setup*来保存设置。

注意： 在每次插 / 拔USB电缆后，必须再次验证Tera Term *Serial port setup*菜单才能重新启动连接。按下Reset按钮来显示欢迎屏幕。

6.3.4 欢迎屏幕显示

欢迎屏幕显示在Tera Term中，如[图 14](#)所示

图14. SBSFU欢迎屏幕显示

```
=====
= (C) COPYRIGHT 2017 STMicroelectronics =
=                                         =
= Secure Boot and Secure Firmware Update =
=====

= [SBOOT] SECURE ENGINE INITIALIZATION SUCCESSFUL
= [SBOOT] STATE: CHECK STATUS ON RESET
= WARNING: A Reboot has been triggered by a Watchdog reset!
= Consecutive Boot on error counter = 1
= INFO: Last execution detected error was:Watchdog error.
= [EXCPT] WATCHDOG RESET FAULT!
= [SBOOT] STATE: CHECK NEW FIRMWARE TO DOWNLOAD
= [SBOOT] STATE: CHECK USER FW STATUS
= No valid FW found in the active slot nor new encrypted FW found in the UserApp download area
= Waiting for the local download to start...
= [SBOOT] STATE: DOWNLOAD NEW USER FIRMWARE
= File> Transfer> YMODEM> Send ....
```

6.4 SBSFU应用程序执行

SBSFU状态机详见[附录F](#)。

在每次重启时，应用程序都会检查用户是否通过按住用户按钮请求了新的固件下载。

如果没有下载请求，则应用程序将检查用户固件的状态

- 由于板子被擦除，所以没有固件可用。
- 应用程序不能跳转到固件，并返回来检查是否有下载请求。

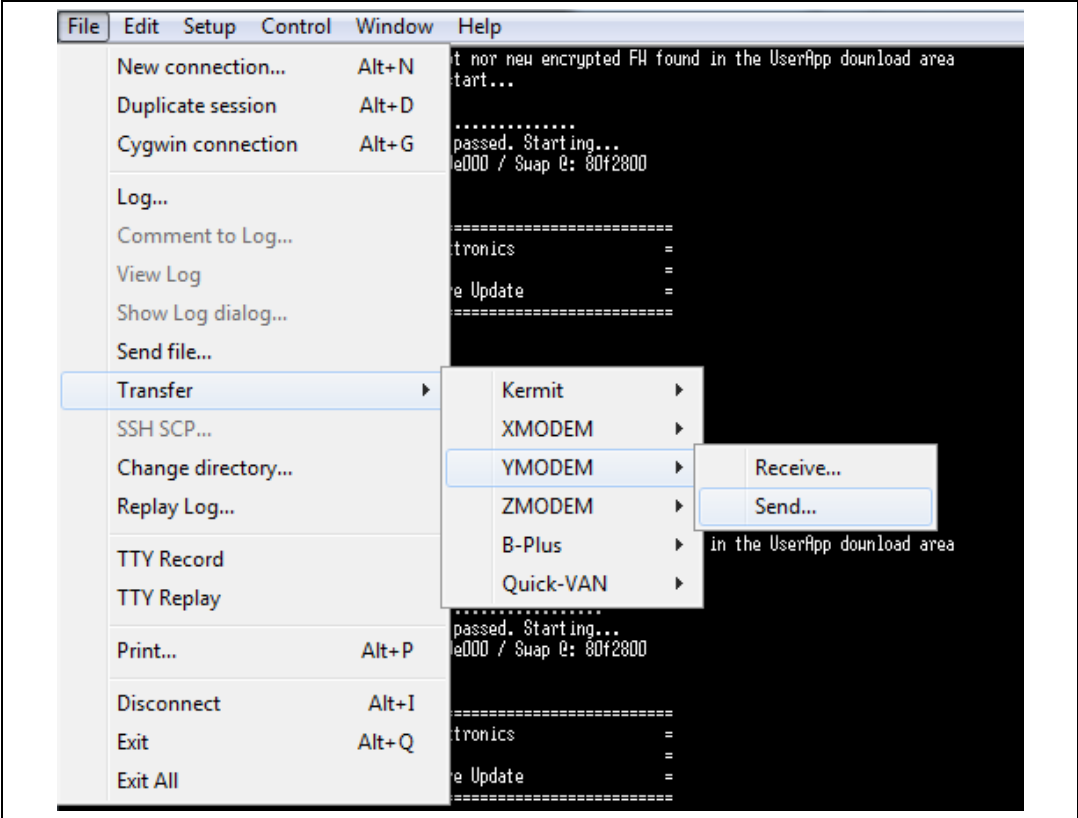
6.4.1 下载请求

当没有用户固件存在时，SBSFU自动等待下载程序来启动。否则，通过在STM32Nucleo板上按住用户按钮来获得下载请求。

6.4.2 发送固件

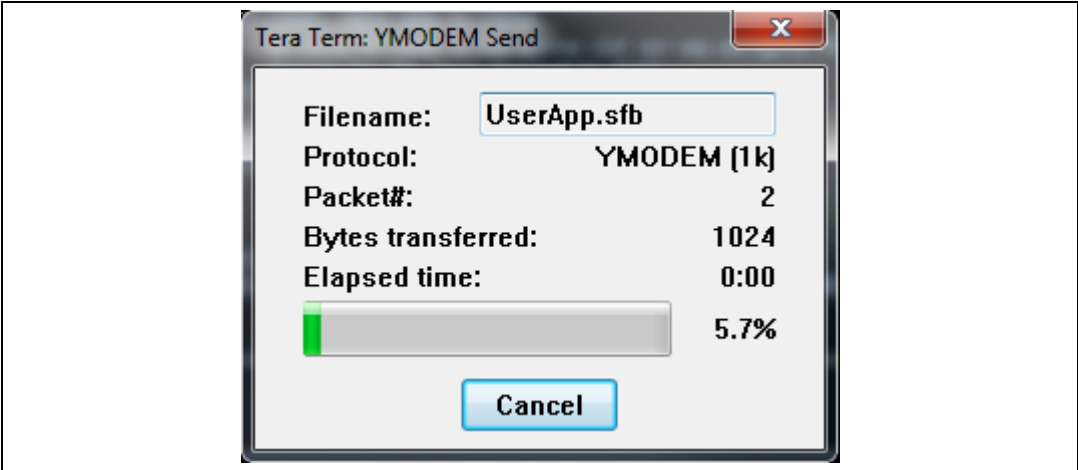
为了发送固件（*.sfb），需要使用Tera Term中的File > Transfer > YMODEM > Send菜单，如[图 15](#)所示。

图15. SBSFU加密固件传输开始



当选择了 *UserApp.sfb* 文件时，Ymodem传输开始。报告传输进度如 图 16 中所示。

图16. 正在进行SBSFU加密固件传输



在程序开始时，进度表会暂停一会儿，此时SBSFU将验证固件头文件的有效性并擦除下载固件映像的Flash插槽。

6.4.3 文件传输完成

文件传输完成后，系统会强制重新启动，如 [图 17](#) 所示。

图17. 加密固件传输后SBSFU重新启动

```

= [SBOOT] STATE: DOWNLOAD NEW USER FIRMWARE
  File> Transfer> \MODEM> Send ..
= [SBOOT] STATE: REBOOT STATE MACHINE
===== End of Execution =====

= [SBOOT] System Security Check successfully passed. Starting...
= [FWIMG] Slot #0 @: 8080800 / Slot #1 @: 800e000 / Swap @: 80f2800

=====
= (C) COPYRIGHT 2017 STMicroelectronics =
= Secure Boot and Secure Firmware Update =
=====

= [SBOOT] SECURE ENGINE INITIALIZATION SUCCESSFUL
= [SBOOT] STATE: CHECK STATUS ON RESET
  INFO: A Reboot has been triggered by a Software reset!
  Consecutive Boot on error counter = 0
  INFO: Last execution detected error was: No error. Success.
= [SBOOT] STATE: CHECK NEW FIRMWARE TO DOWNLOAD
= [SBOOT] STATE: CHECK USER FW STATUS
  New Fw Encrypted, to be decrypted
= [SBOOT] STATE: INSTALL NEW USER FIRMWARE .....
= [SBOOT] STATE: VERIFY USER FW SIGNATURE
= [SBOOT] STATE: EXECUTE USER FIRMWARE

=====
= (C) COPYRIGHT 2017 STMicroelectronics =
= User App #A =
=====

===== Main Menu =====
Download a new Fw Image ----- 1
Test Protections ----- 2
Test SE User Code ----- 3

```

随后 [图 17](#) 中显示的系统状态将提供以下信息：

- 没有固件可供下载。
- 固件被检测为已加密。用户固件被解密。
- 如果解密完成，则安装用户固件。
- 如果安装完成，则验证用户固件签名。
- 如果验证完成，则执行用户固件。

6.4.4 系统重启

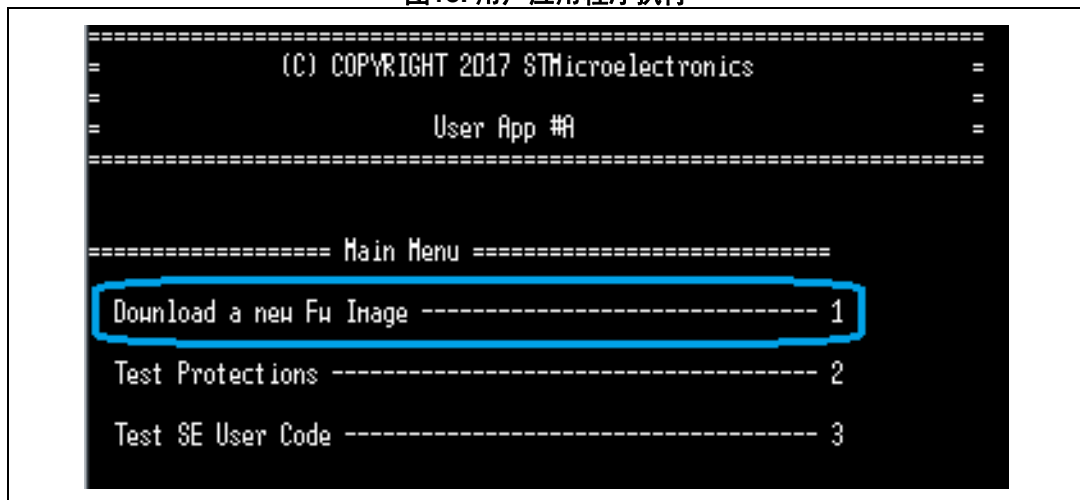
按下复位按钮强制系统重启：用户应用程序由SBSFU启动。

注：在复位期间按住用户按钮，会触发强制下载状态，而不是执行用户应用程序。

6.5 用户应用程序执行

用户应用程序根据图 18中所示选择来执行，并进一步按照从第 6.5.1章到第 6.5.3章所述来执行。

图18. 用户应用程序执行



6.5.1 下载新固件映像

下载新的固件映像的过程与第 6.4 节第29页中SBSFU的步骤相同：

1. 发送固件
 - 在Tera Term中，点击File>Transfer>YMODEM>Send
 - 选择UserApp.sfb（编译为UserApp#B）
2. 系统重启
3. 安全启动状态机处理新映像
 - 验证固件头文件
 - 解密固件
 - 安装固件
 - 验证固件签名
 - 执行固件

图19. 通过用户应用程序下载加密的固件

```

===== New Fu Download =====

-- Send Firmware

-- -- Erasing download area ...

-- -- Programming Completed Successfully!

-- -- Bytes: 17872

-- Image correctly downloaded - reboot

= [SB00T] System Security Check successfully passed. Starting...
= [FWIMG] Slot #0 @: 8080800 / Slot #1 @: 800e000 / Swap @: 80f2800

=====
= (C) COPYRIGHT 2017 STMicroelectronics =
= Secure Boot and Secure Firmware Update =
=====

= [SB00T] SECURE ENGINE INITIALIZATION SUCCESSFUL
= [SB00T] STATE: CHECK STATUS ON RESET
INFO: A Reboot has been triggered by a Software reset!
Consecutive Boot on error counter = 0
INFO: Last execution detected error was:No error. Success.
= [SB00T] STATE: CHECK NEW FIRMWARE TO DOWNLOAD
= [SB00T] STATE: CHECK USER FW STATUS
New Fu Encrypted, to be decrypted
= [SB00T] STATE: INSTALL NEW USER FIRMWARE .....
= [SB00T] STATE: VERIFY USER FW SIGNATURE
= [SB00T] STATE: EXECUTE USER FIRMWARE

=====
= (C) COPYRIGHT 2017 STMicroelectronics =
= User App #8 =
=====

===== Main Menu =====

Download a new Fu Image ----- 1

Test Protections ----- 2

Test SE User Code ----- 3

```

6.5.2 测试保护

测试保护菜单如图 20所示。

图20. 用户应用程序测试保护菜单



作为测试运行功能，在每次禁止操作或错误注入的测试尝试时都会打印测试保护菜单：

- 固件测试（#1, #2）
 - 尝试访问受保护的代码或数据（在RAM或Flash中）会导致重置
- PCROP测试（#3）
 - 尝试访问保护密钥的PCROP区域时会导致错误
- WRP测试（#4）
 - 尝试擦除写保护的代码时会导致错误
- IWDG测试（#5）
 - 不刷新看门狗会引起一个模拟死锁的重置
- TAMPER测试（#6）
 - 检测到篡改事件会导致重置
 - 为了产生篡改事件，用户必须将PA0（CN7.28）连接到GND（将手指靠近CN7.28可能就足够了）。
- CORRUPT IMAGE测试（#7）
 - 在下次启动时会导致签名验证失败。

按下x键可以返回上一级菜单。

6.5.3 测试安全引擎用户代码

当前用户固件的版本和大小利用安全引擎服务进行检索，并打印在控制台中。

7 了解启动时的最后执行状态消息

表 4列出了主要错误信息及其解释。

表4. 启动时的错误消息

错误消息	意义
无错误。成功。	没有遇到问题。
固件错误。	发生防火墙异常：由防火墙保护的某些代码或数据已被解除出安全引擎上下文。
看门狗错误。	看门狗到期：处理时间过长，看门狗没有在适当的时间被重新加载。
内存故障。	内存故障由MPU故障处理程序报告。
硬性故障。	Arm® Cortex®-M硬性故障异常。
篡改故障。	TAMPER检测报告。
检查保护错误。	示例代码中未使用。这可用在对所应用保护机制进行定期验证时记录错误。
检查复位错误的状态。	在检查启动时的状态时遇到错误（一般错误）。
检查新用户FW以下载错误。	检查是否存在本地下载请求时遇到错误（一般错误）。
下载新用户FW错误。	执行本地下载时遇到错误（一般错误）。
验证用户FW状态错误。	验证用户固件状态时遇到错误。如果Flash状态不允许确定固件状态，则会出现此错误（一般错误）。
解密用户FW错误。	当前示例代码中未使用。这可以用来记录与解密固件相关的一般错误。在该示例中，使用了更特殊的错误：“解密失败”。
安装用户FW错误。	安装新固件时遇到错误（一般错误）。
验证用户FW签名。	验证活动固件的签名时遇到错误。在示例代码中，在固件状态检查过程中已经检查过签名，所以不应报告这个错误。
回滚上一个用户FW错误。	恢复过程中遇到错误（一般错误）。
执行用户FW错误。	尝试启动活动固件时遇到错误（一般错误）。
最大值 连续出现错误。	系统已达到连续错误的最大数量（在固件或SBSFU上下文中）。本例中，这意味着连续发生3次IWDG或防火墙异常。
不能设置SE锁。	在启动活动固件之前尝试在“固件执行”模式（非特权模式）下配置安全引擎时遇到错误。
FW大小不一致。	此错误意味着在本地下载过程中，头文件中指定的大小与下载文件的大小不匹配。
FW过大。	此错误意味着在本地下载过程中，头文件中指定的固件大小比插槽#1的容量要大。
Ymodem com故障。	在本地下载过程中，下载操作未成功完成（Ymodem协议问题）。
文件未正确接收。	在本地下载过程中，二进制文件未能正确接收。目前这种检测是极简检测。
头文件认证失败。	在本地下载过程中，头文件无法成功认证。只有当存储在RAM中的头文件被更改时才会发生此错误（否则，下载会被忽略而不会触发严重故障）。
解密失败。	解密插槽#1的内容时遇到错误。此错误报告解密或认证问题，因为解密的最后阶段是对签名的检查。

表4. 启动时的错误消息（续）

错误消息	意义
签名检查错误。	在安装过程中验证解密固件的签名时遇到错误。在示例代码中，不会发生此错误，因为在解密阶段就会捕获签名问题（报告“解密失败”）。
二进制格式错误（未加密）。	在安装过程中遇到错误：插槽#1中的二进制文件未加密。也许下载的是固件的明文版本而不是加密版本？
闪存错误。	安装过程中遇到闪存错误。
FWIMG模式问题。	在安装过程中遇到错误：写入某些SBSFU模式时出现内部问题。
在交换插槽#0和插槽#1中的映像时出错。	在安装过程中遇到错误：交换映像时出现故障（以前的固件和解密的固件）。
固件版本被反回滚检查所拒绝。	在安装过程中遇到错误：该固件版本不能被接受（新版固件已经安装或者版本低于允许的最低版本）。
未知错误。	未记录的错误（意外的异常或意外的状态机问题）。



附录A 安全引擎保护的环境

安全引擎（SE）概念定义了一个受保护的区域，用来输出可信环境中执行的一组安全功能。

SE向SBSFU应用示例提供以下功能：

- 安全引擎初始化功能
- 安全加密功能
 - AES-GCM和AES-CBC 解密
 - SHA256哈希和ECDSA验证
 - 敏感数据（密钥，AES上下文）不能离开受保护的环境，并且不能被未受保护的代码访问。
- 对固件映像信息的安全读/写访问
 - 对与用户应用程序共享的受保护Flash区域进行读写操作。
 - 只有受保护的代码才能访问此区域。
- 锁定安全引擎中一些功能的安全服务
 - 单向锁定机制：一旦锁定，除非通过系统重置，否则无法解锁
 - 一旦锁定，就不能再通过调用门机制来执行功能了
 - 在安全引擎示例中通过锁定机制锁定的功能为：
 - 安全引擎初始化功能
 - 带OEM密钥的安全加密功能
 - 对固件映像信息的安全读/写访问
 - 锁定安全引擎中一些功能的安全服务

注： SE导出的功能可以根据最终用户应用程序的需要进行扩展

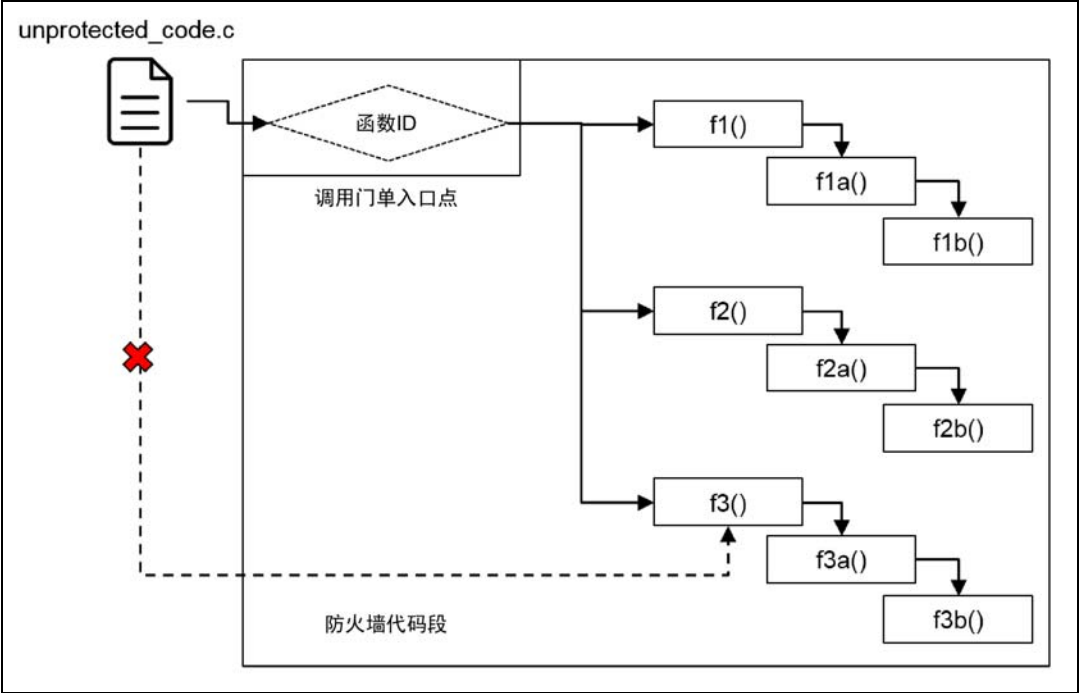
为了处理防火墙调用门机制并为用户提供一组安全的API，SE设计了由SE内核和SE接口组成的两级体系结构。

A.1 SE内核调用门机制

防火墙使用特定的“调用门”机制打开或关闭：必须使用单入口点（位于代码段基址的第二个字）来打开门并执行由防火墙保护的代码。如果访问受保护的代码而不通过调用门机制，则会产生系统重置。

因为对应调用门序列的唯一方法是通过单个调用门入口点，因此，假如外部应用程序需要调用受防火墙保护的多个功能（例如，加密和解密功能），则需要选择执行哪个内部函数。一种解决方案是，使用一个参数来指定执行哪个函数，例如CallGate(F1_ID)、CallGate(F2_ID)等等。根据参数，来内部调用正确的函数。

图21. 防火墙调用门机制



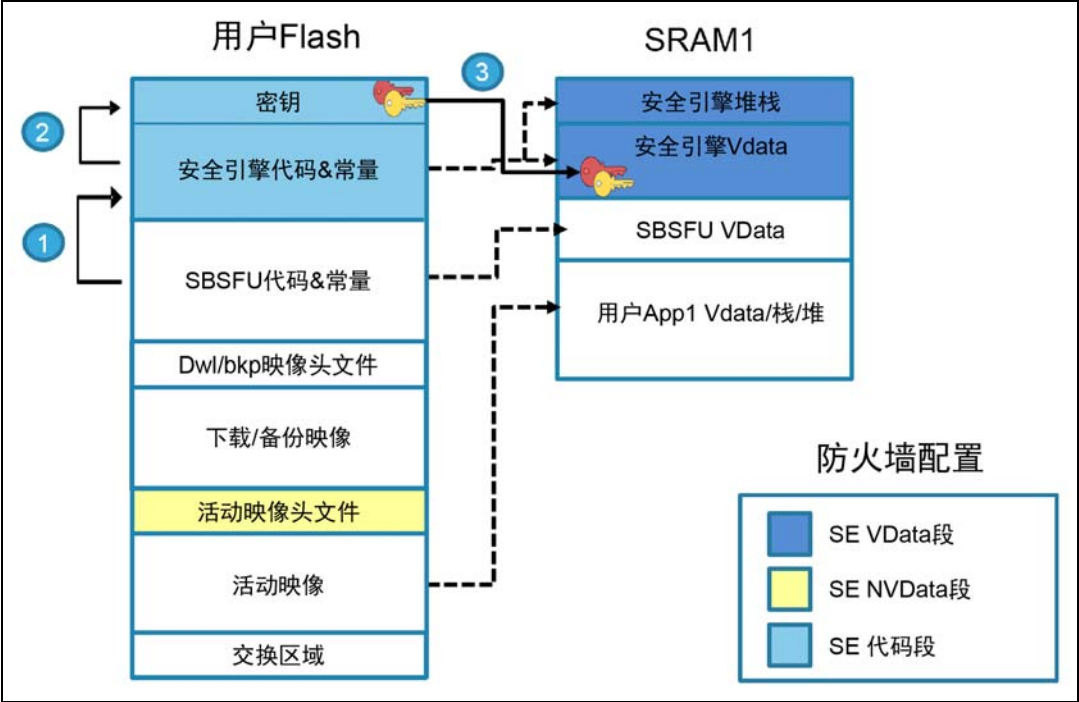
注意： 代码段必须包含防火墙打开时执行的所有代码。例如，如果调用序列是callgate->f1()->f1a()->f1b()，则全部三个函数f1()、f1a()和f1b()必须包含在代码段中。

[图 22](#)显示了执行加密操作（需要访问密钥）的步骤，以遵守调用门机制。

对于加密函数：

1. SBSFU代码调用调用门函数，打开防火墙并执行受保护的代码
2. 调用门函数检查参数和安全性，然后调用请求的Crypto函数
3. SE Crypto函数调用内部ReadKey函数，此函数可将密钥移入SRAM1的受保护部分，然后在加密操作中使用它们。

图22. 安全引擎调用门机制

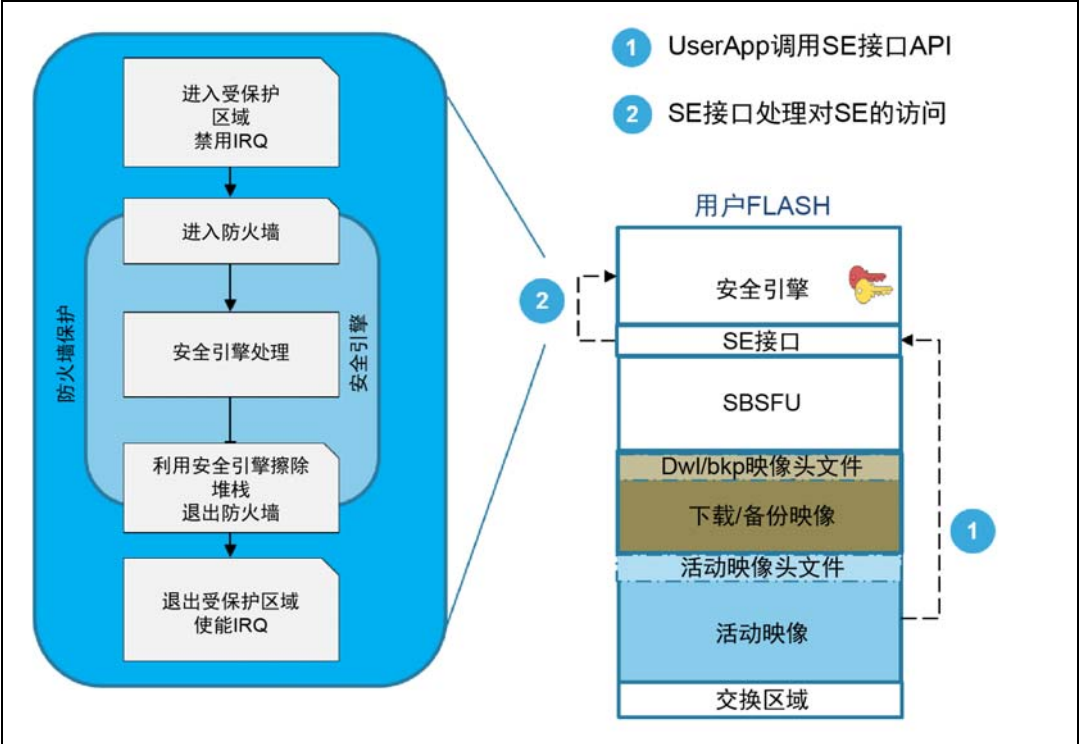


A.2 SE接口

由防火墙保护的代码必须是不能中断的，在打开防火墙之前，是否禁用中断取决于用户代码。

SE接口提供了一个用户友好的封装，用来处理受保护区域的入口和出口，并在该区域中执行实际的SE调用门函数，如 [图 23](#) 中所示。

图23. 安全引擎接口



SE接口机制简化了对调用门的访问控制，可独立于用户实现。SE接口API与用户应用程序共享，因此可使用SE提供的服务以安全方式执行敏感操作（如果未被安全引擎锁定服务锁定）。

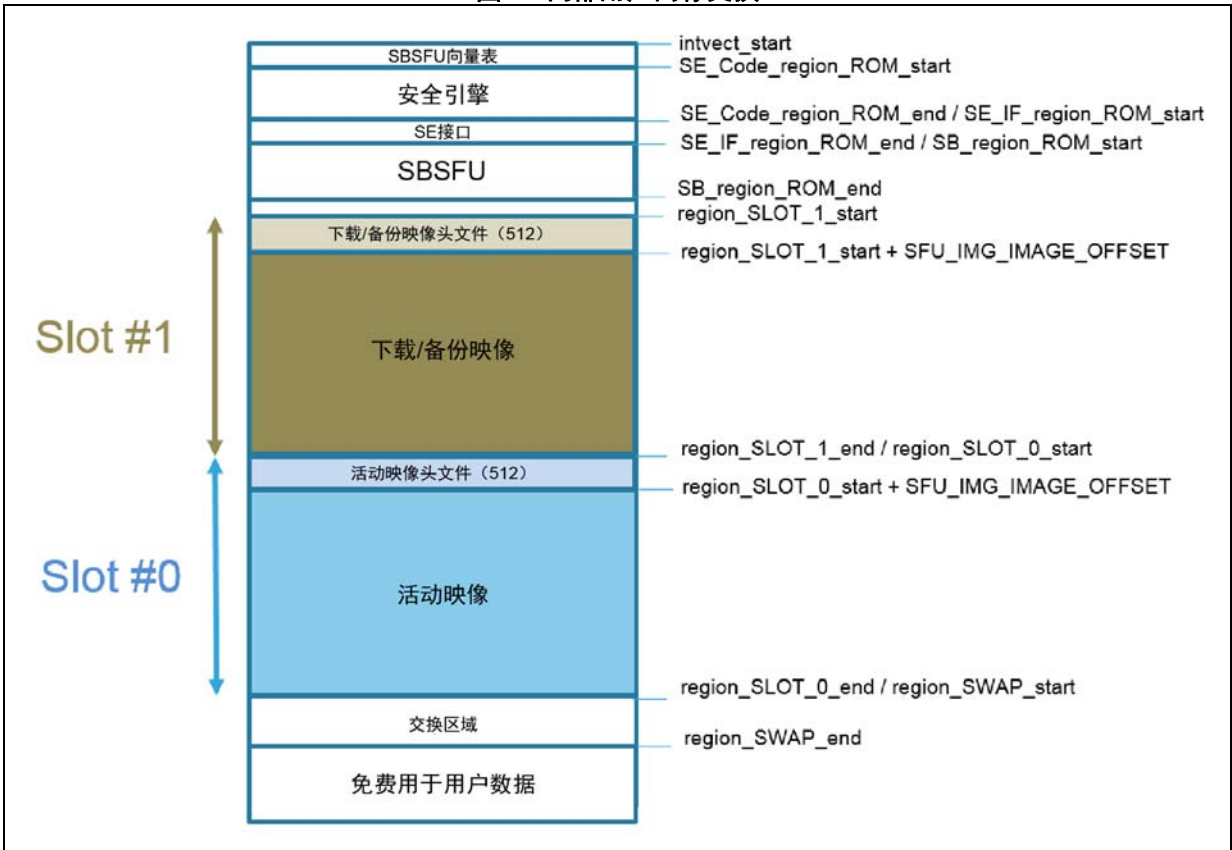
附录B 双映像处理

一些SBSFU应用程序示例可处理存储在内部闪存中的两个固件映像。

B.1 元件和角色

- Slot #0:
 - 该插槽包含了有效固件（固件头文件+固件）。这是SBSFU在启动时启动的用户应用程序（验证其有效性后）。
- Slot #1:
 - 下载过程完成后，此插槽用于存储下载的固件（固件头文件+加密固件），以便在下次重启时安装。
 - 在安装过程之后，此插槽中含有备份固件（明文形式），直到发生回滚过程或触发新的下载过程。
- 交换区域:
 - 这是用来交换插槽#0和插槽#1的内容的Flash区域。
 - 不过，此区域并不是每次及每个Flash扇区交换都要使用的缓冲区。它用于移动第一个扇区，因此会在闪存中产生移位，从而可以逐扇区交换两个插槽。

图24. 内部用户闪存交换



B.2 映射定义

闪存的组织方式如表 5所示，其中显示了来自某些STM32L4系列产品IAR示例项目的图。

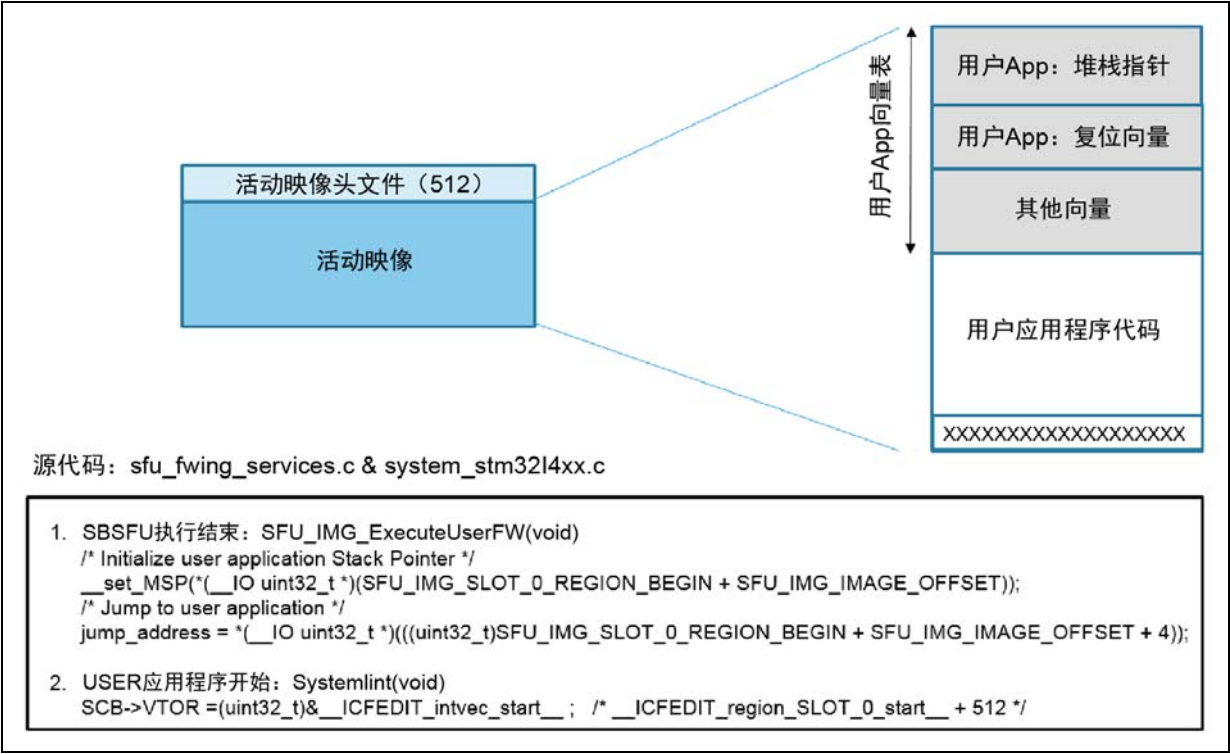
表5. 双映像闪存组织

	NUCLEO-L476RG	NUCLEO-L432KC	B-L475E-IOT01A	32L496GDISCOVERY
代码起始地址	0x0800 0000	0x0800 0000	0x0800 0000	0x0800 0000
代码量	54 KB	54 KB	54 KB	58 + 4 KB ⁽¹⁾
插槽#0大小	456 KB	96 KB	456 KB	448 KB
插槽#1大小	456 KB	96 KB	456 KB	448 KB
交换区域大小	8 KB	4 KB	8 KB	16 KB

1. 由于32L496GDISCOVERY板的BSP（I/O扩展器）带来的额外4 KB。

要启动应用程序，SBSFU使用用户应用程序堆栈指针值初始化SP寄存器，然后跳转到用户应用程序重置向量（请参考图 25）。

图25. 用户应用程序向量表



附录C 单映像处理

一些SBSFU应用程序示例处理存储在内部闪存中的单固件映像。

这种操作模式允许通过以下方式最大化用户固件大小：

- 减少闪存中的SBSFU占用空间
- 为用户应用程序分配更多的Flash空间

这些好处是以一些功能为代价的：

- 不能确保安全固件映像编程：
 - 触发更新时，不会备份活动固件映像
 - 如果活动固件无效，则不会发生回滚
- 用户应用程序无法下载新的固件映像：本地下载过程是更新活动用户代码的唯一方法。

C.1 元件和角色

插槽#0：

- 该插槽包含了有效固件（固件头文件+固件）。这是SBSFU在启动时启动的用户应用程序（验证其有效性后）。
- 当下载并安装新的固件映像时（固件头文件验证之后），该插槽会直接更新。

C.2 映射定义

闪存的组织方式如 表 6所示，其中显示了来自某些STM32L4系列产品IAR示例项目的图。

表6. 单映像闪存组织				
	NUCLEO-L476RG	NUCLEO-L432KC	B-L475E-IOT01A	32L496GDISCOVERY
代码起始地址	不支持 ⁽¹⁾	0x0800 0000	不支持 ⁽¹⁾	不支持 ⁽¹⁾
代码量		47 KB		
插槽#0大小		208 KB		

1. 不支持单映像变体，因为双存储区闪存设备上：
- 防火墙代码段必须位于存储区1中，防火墙数据段位于存储区2中。
- 防火墙代码和数据段必须位于与每个存储区的基址具有相同偏移的位置（确保即使存储区交换也能保护机密）

要启动应用程序，SBSFU使用用户应用程序堆栈指针值初始化SP寄存器，然后跳转到用户应用程序重置向量（请参考 图 25：用户应用程序向量表）。

附录D 加密方案处理

提供了三种密码方案作为例子来说明加密操作。默认的加密方案同时使用对称（AES-CBC）和非对称（ECDSA）加密。因此，它可以处理私钥（AES128私钥）以及公钥（ECC密钥）。
提供了两种备选方案，可以通过SECoreBin编译器开关（名为“SECBOOT_CRYPT0_SCHEME”）进行选择。

D.1 此软件包中包含的加密方案

表 7显示了使用SECBOOT_CRYPT0_SCHEME编译器开关选择的加密方案。

表7. 加密方案列表

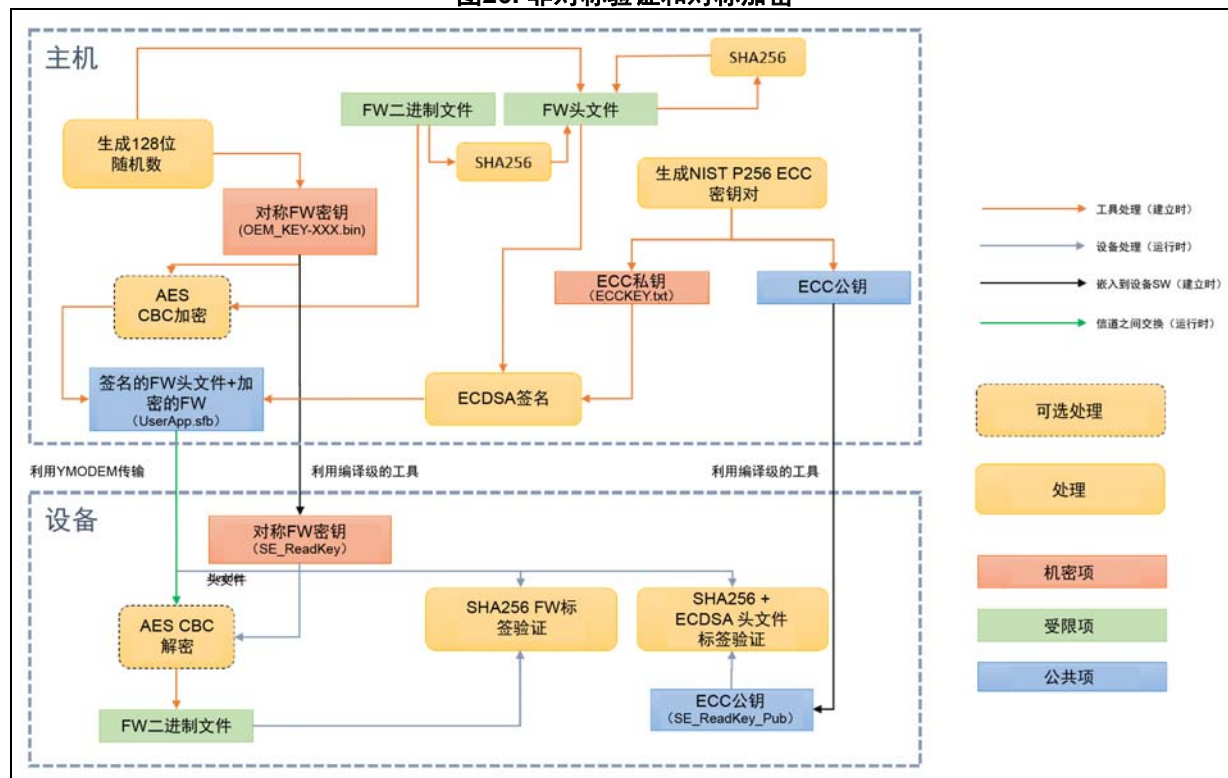
SECBOOT_CRYPT0_SCHEME值	认证	机密性	完整性
SECBOOT_ECDSA_WITH_AES128_CBC_SHA256（默认）	ECDSA	AES128-CBC	SHA256
SECBOOT_ECDSA_WITHOUT_ENCRYPT_SHA256	ECDSA	无 ⁽¹⁾	SHA256
SECBOOT_AES128_GCM_AES128_GCM_AES128_GCM	AES GCM		

1. 还必须通过将SFU_IMAGE_PROGRAMMING_TYPE编译器开关设置为值SFU_CLEAR_IMAGE，对SBSFU项目进行配置，来处理明文固件映像。

D.2 非对称验证和对称加密方案

这些方案 (SECBOOT_ECCDSA_WITH_AES128_CBC_SHA256, SECBOOT ECCDSA WITHOUT ENCRYPT SHA256) 用于固件解密和验证, 如 [图 26](#) 中所示。

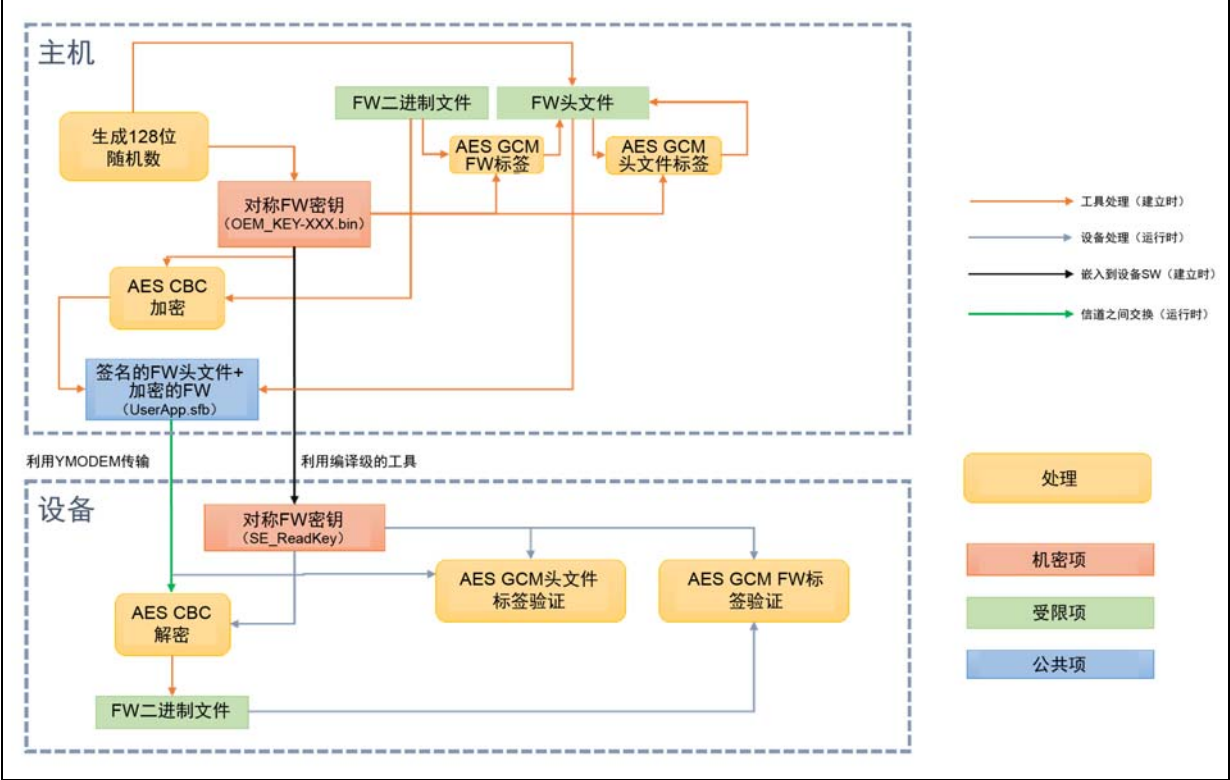
图26. 非对称验证和对称加密



D.3 对称验证和加密方案

此方案（SECBOOT_AES128_GCM_AES128_GCM_AES128_GCM）用于固件解密和验证，如图 27 中所示

图27. 对称验证和加密



D.4 安全启动和安全固件更新流程

图 28和图 29表明了加密操作（采用FW加密的非对称加密方案）如何集成到SBSFU执行启动流程中。

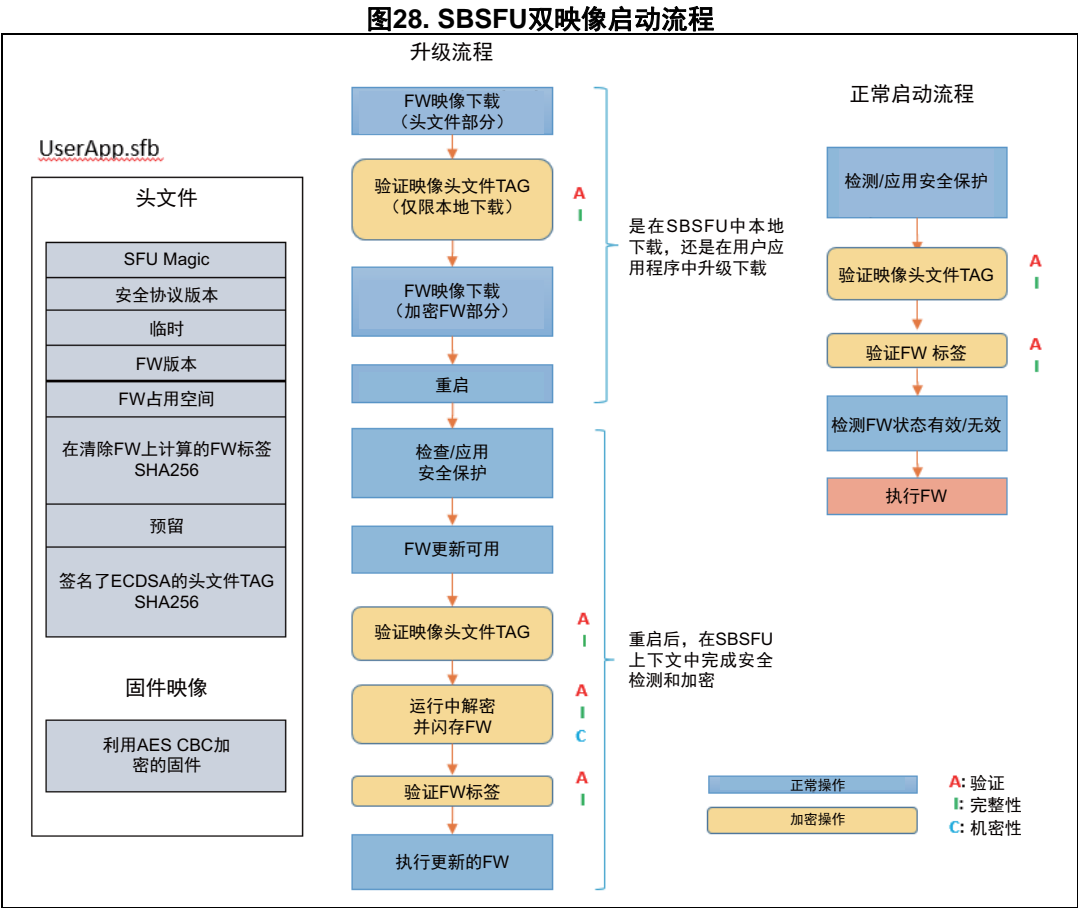
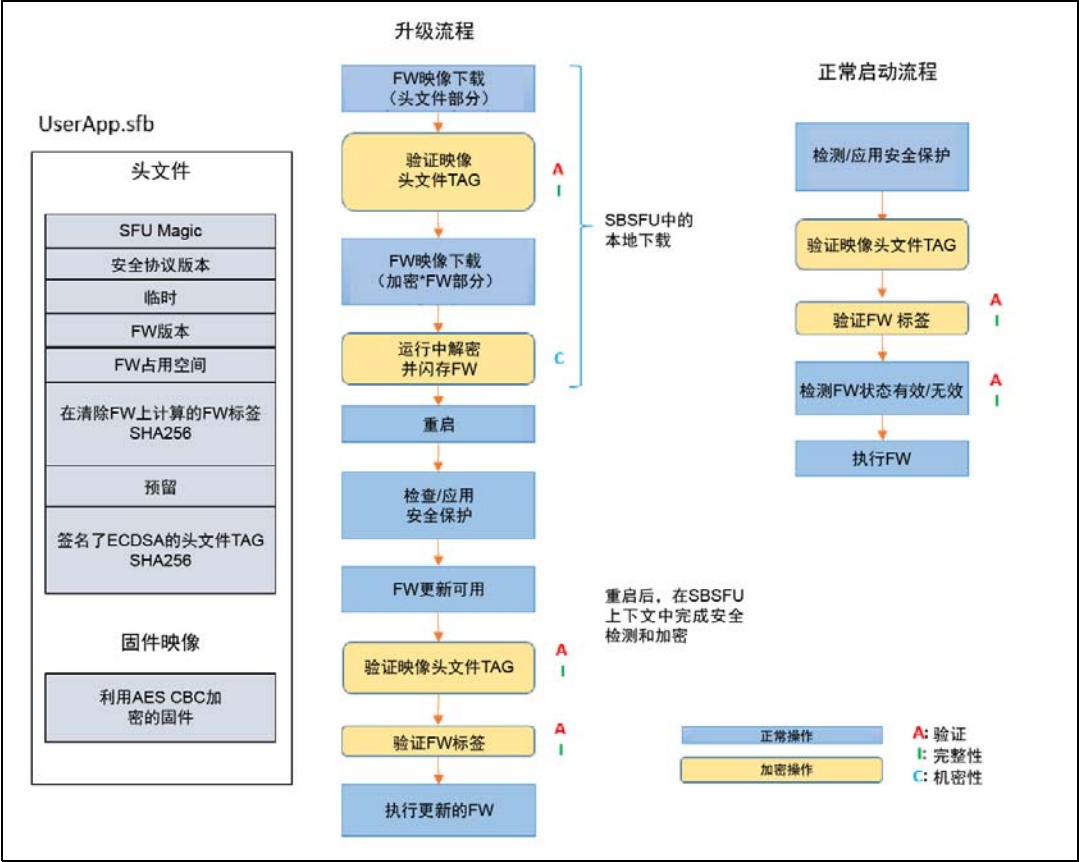


图29. SBSFU单映像启动流程



附录E 固件映像准备工具

X-CUBE-SBSFU STM32Cube扩展包随附prepareimage固件映像准备工具，可以：

- 考虑到选定的加密方案和密钥
- 有需要时加密固件映像
- 生成固件头文件，其具有验证和完整性检查所需的所有数据

prepareimage工具以两种格式提供：

- Windows®可执行文件：需要标准Windows®命令解释器
- Python™脚本：Python™解释器以及
Middlewares\ST\STM32_Secure_Engine\Utilities\KeysAndImages\readme.txt中列出的组件是必需的

Windows®可执行文件可以快速轻松地使用该软件包，其中有全部三种预定义的加密方案。作为源代码提供的Python™脚本提供了以灵活方式定义其他加密方案的可能性。

E.1 工具位置

Python™脚本以及Windows®可执行文件位于安全引擎组件的文件夹
Middlewares\ST\STM32_Secure_Engine\Utilities\KeysAndImages。

E.2 输入

软件包随附了一些默认密钥和加密设置，在文件夹\Projects\NUCLEO-L476RG
Applications\2_Images\2_Images SECoreBin\Binary中。

以下每个文件都可以这样使用，或者考虑用户设置而修改：

- ECCKEY.txt: PEM格式的ECC私钥。它用来签名固件头文件。此密钥未嵌入到SECoreBin中，文件se_key.s中的工具只生成相应公钥
- nonce.bin: 这是一个随机数（使用AES-GCM时）或IV（使用AES-CBC时）。该值由工具自动添加到固件头文件中。
- OEM_KEY_COMPANY1_key_AES_CBC.bin: 对称AES-CBC密钥。此密钥用于AES-CBC加密和解密操作，并嵌入在文件se_key.s中。此文件由
OEM_KEY_COMPANY1_key_AES_GCM.bin独占
- OEM_KEY_COMPANY1_key_AES_GCM.bin: 对称AES-GCM密钥。此密钥可用于所有AES-GCM操作，并嵌入在文件se_key.s中。此文件由
OEM_KEY_COMPANY1_key_AES_CBC.bin独占

该工具根据文件Projects\NUCLEO-L476RG Applications\2_Images\2_Images
SECoreBin\vnclse_crypto_config.h中的SECBOOT_CRYPTOScheme选择的加密方案，使用适当的一组文件。

E.3 输出

该工具会生成：

- *SECoreBin*项目中编译的*se_key.s*文件：该文件包含嵌入在设备中的密钥（适时使用的对称私钥和ECC公钥）以及用于访问它们的代码。从IDE运行该工具时，该文件位于 *Projects\NUCLEO-L476RG\Applications\2_Images\2_Images\SECoreBin\Src*中。
- *.sfb*文件，其中包含用户固件头文件和加密的用户固件映像（当所选加密方案对用户固件进行加密时）。当从IDE运行该工具时，该文件将在 *Projects\NUCLEO-L476RG\Applications\2_Images\2_Images\UserApp\Binary*中生成。
- *.bin*文件，连接了SBSFU二进制文件、UserApp二进制文件和活动的FW映像头文件。使用flasher工具将此文件闪存到设备中会使UserApp安装过程变得简单，因为FW头文件和FW映像已经正确安装。安装UserApp不需要使用SBSFU应用程序。

E.4 IDE集成

*prepareimage*工具与IDE集成，作为Windows®批处理文件，用于：

- *SECoreBin*应用程序的预生成操作：在此阶段，管理加密密钥
- *UserApp*应用程序的后生成操作：在此阶段，构建固件映像

使用IDE进行编译时，会处理密钥和固件映像。用户不需要额外操作。编译步骤结束时：

- 所需密钥嵌入到*SECoreBin*二进制文件中
- 要安装的固件映像以适当格式生成，并具有适当的固件头文件*.sfb*文件。此*.sfb*文件可以通过Ymodem协议来传输，以供SBSFU安装。
- *.bin*文件可以用于UserApp测试（*SBFU_UserApp.bin*）。

批处理文件，在IDE中集成该工具，位于文件夹

*Projects\NUCLEO-L476RG\Applications\2_Images\2_Images\SECoreBin\EWARM*中：

- *prebuild.bat*：在编译*SECoreBin*项目时调用该工具来执行预生成操作
- *prebuild.bat*：在编译*UserApp*项目时调用该工具来执行后生成操作

这些批处理文件允许从Windows®可执行变体无缝切换到 *prepareimage*工具的Python™脚本变体。该过程在文件本身中进行了描述。

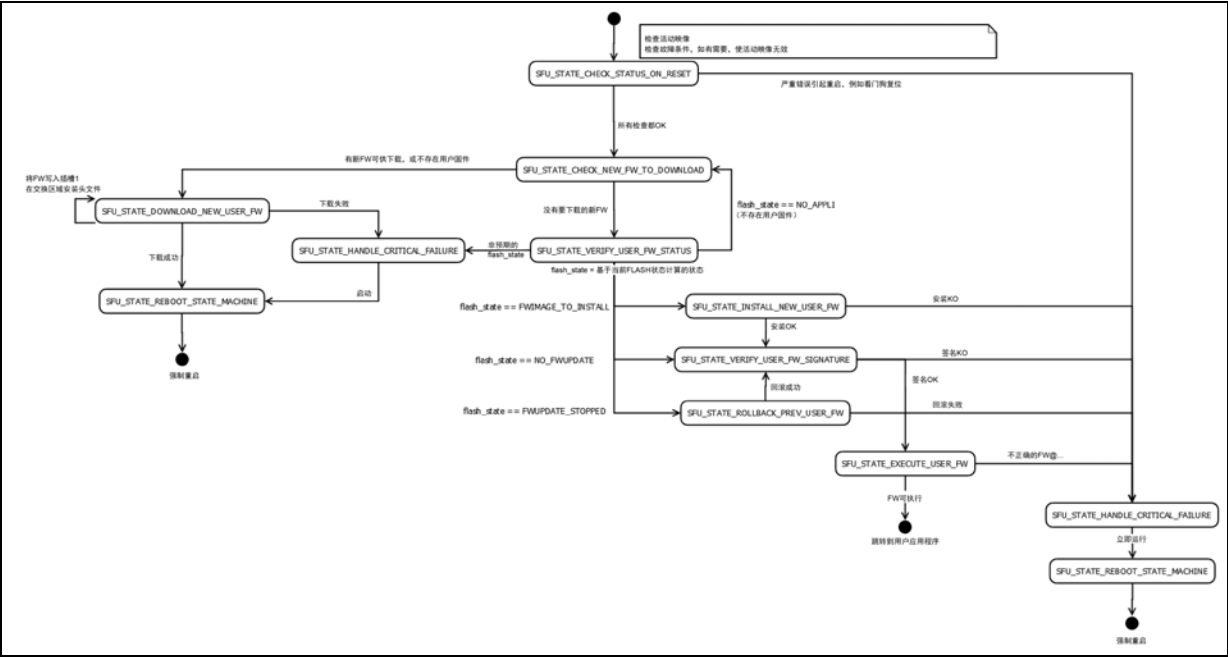
附录F SBSFU应用程序状态机

本节描述了SBSFU应用程序状态机的状态，该状态机在X-CUBE-SBSFU软件包中提供。

F.1 双映像SBSFU

对应状态图如图 30中所示。

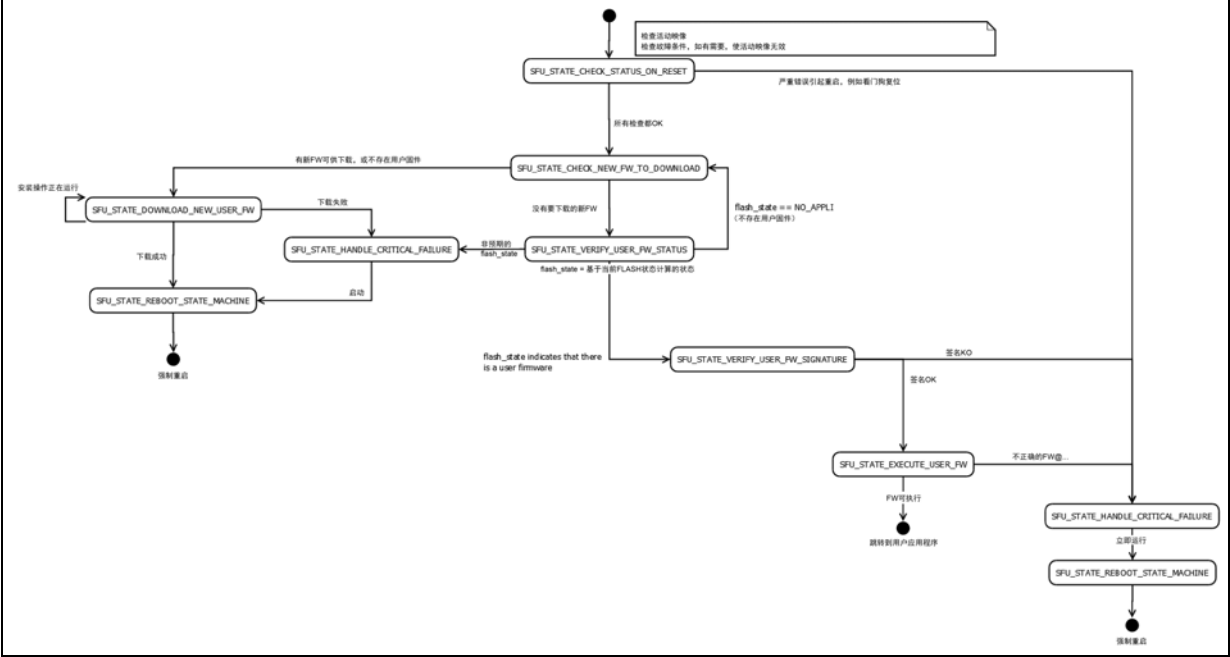
图30. 双映像SBSFU应用程序状态图



F.2 单映像SBSFU

对应状态图如图 31中所示。

图31. 单映像SBSFU应用程序状态图



F.3 SBSFU FSM状态

FSM状态为：

- SFU_STATE_CHECK_STATUS_ON_RESET: 在这个阶段，SBSFU检查复位原因并决定如何处理
 - 检查本地固件下载可用性
 - 或作为关键故障情况来处理复位原因
- SFU_STATE_CHECK_NEW_FW_TO_DOWNLOAD: 在这个阶段，SBSFU检查是否必须处理本地下载
 - SBSFU检查用户按钮是否被按下
 - 如果这样，则下载并验证固件头文件
 - 如果此头文件正常，那么就会触发下载
 - 如果没有按下该按钮或者如果头文件遇到问题，则SBSFU切换到SFU_STATE_VERIFY_USER_FW_STATUS状态
- SFU_STATE_DOWNLOAD_NEW_USER_FW: 在此阶段，SBSFU下载加密的固件并通过UART进行安装，并将其存储在内部闪存（插槽#1）中
 - 如果此下载过程正常，则会触发重启
 - 如果遇到问题，则处理该严重故障
- SFU_STATE_VERIFY_USER_FW_STATUS: 在此阶段，SBSFU检查内部闪存状态，以导出一个状态
 - 如果固件安装被中断：将触发恢复过程
 - 如果安装了固件并且没有新固件可供安装，则在执行之前验证当前安装的固件（安装在插槽#0中）
 - 如果新固件准备好安装，则会触发安装过程
- SFU_STATE_INSTALL_NEW_USER_FW: 在此阶段，SBSFU解密并安装存储在插槽#1中的固件
 - 如果一切正常，插槽#0和插槽#1的内容将被交换，并进入固件验证阶段
 - 如果遇到问题，则处理该严重故障
- SFU_STATE_VERIFY_USER_FW_SIGNATURE: 这是运行用户应用程序之前的最后阶段。SBSFU检查活动固件的有效性（安装在插槽#0中）
 - 如果固件有效，则启动它
 - 如果固件无效，则处理该严重故障
- SFU_STATE_EXECUTE_USER_FW: 在此阶段，SBSFU准备上下文切换以启动用户应用程序
 - 如果一切正常，则用户应用程序启动
 - 如果遇到问题，则处理该严重故障
- SFU_STATE_ROLLBACK_PREV_USER_FW: 在此阶段，SBSFU会重新安装备份的固件（如果有的话）
 - 如果恢复过程成功，则固件在启动前会被验证
 - 如果遇到问题，则处理该严重故障
- SFU_STATE_HANDLE_CRITICAL_FAILURE: 这是一个占位符，用于处理遇到的所有严重故障
- SFU_STATE_REBOOT_STATE_MACHINE: 强制重启

版本历史

表8. 文档版本历史

日期	版本	变更
2017年12月7日	1	初始版本。
2017年12月20日	2	删除了对 第 7 节：了解启动时的最后执行状态消息 和 B.2 映射定义 中的集成指南的引用。更新了 表 4：启动时的错误消息 和 第 5.2.2 节：编程STM32微控制器的软件工具 。
2018年4月20日	3	文档范围扩展到非对称和对称加密方案。添加了单映像模式。扩展了STM32L4系列支持的功能： <ul style="list-style-type: none">– 更新了所有章节。– 更新了 附录A 安全引擎保护的环境 和 附录B 双映像处理– 增加了 附录C 单映像处理，附录D 加密方案处理，以及 附录E 固件映像准备工具。– 删除了MSC附录。

表9. 中文文档版本历史

日期	版本	变更
2018年8月27日	1	中文初始版本。

重要通知 - 请仔细阅读

意法半导体公司及其子公司（“ST”）保留随时对 ST 产品和 / 或本文档进行变更、更正、增强、修改和改进的权利，恕不另行通知。买方在订货之前应获取关于 ST 产品的最新信息。ST 产品的销售依照订单确认时的相关 ST 销售条款。

买方自行负责对 ST 产品的选择和使用，ST 概不承担与应用协助或买方产品设计相关的任何责任。

ST 不对任何知识产权进行任何明示或默示的授权或许可。

转售的 ST 产品如有不同于此处提供的信息的规定，将导致 ST 针对该产品授予的任何保证失效。

ST 和 ST 徽标是 ST 的商标。所有其他产品或服务名称均为其各自所有者的财产。

本文档中的信息取代本文档所有早期版本中提供的信息。本文档的中文版本为英文版本的翻译件，仅供参考之用；若中文版本与英文版本有任何冲突或不一致，则以英文版本为准。

© 2018 STMicroelectronics - 保留所有权利