Description of JEDEC Serial NOR Security HAL implementation in TF-M with ETSS partition

Introduction

JEDEC JESD261 standard, titled Serial NOR Security Hardware Abstraction Layer (HAL) defines a software layer that is intended to provide uniform cryptographic operations to compliant Secure Flash devices.

JESD261 HAL enables manufacturer-neutral communication with Secure Flash devices. This combines baseline security features, such as protection against replay, impersonation, and modification. This HAL intends to not only unify secure storage implementations but also facilitate second sourcing for customers.

This document mainly describes the implementation of JESD261 HAL in a trusted execution environment, ARM Trusted Firmware-M(TF-M) framework with extended External Trusted Secure Storage(ETSS) partition on MCU platforms with TrustZone technology.

Contents

1. General information

Throughout this application note, the terminology JEDEC security HAL refers to Serial NOR Security Hardware Abstraction Layer defined in JEDEC JESD261 standard. Table 1 presents the definition of acronyms that are relevant for a better understanding of this document.

Table 1. List of acronyms

Acronym	Definition
ETSS	External Trusted Secure Storage partition. A secure partition containing external trusted secure services allowing the access of assets stored in external secure Flash.
ITS	Internal trusted storage service. Internal trusted storage service provided by TF-M.
PS	Protected storage service. Protected storage service provided by TF-M.
SPE	Secure processing environment (PSA term). In TF-M this means the secure domain protected by TF-M.
TEE	Trusted execution environment.
TF-M	Trusted firmware for M-class Arm. TF-M provides a reference implementation of secure world software for Armv8-M.

2. References

The resources presented in Table 2 and Table 3 below are public and available either on Macronix web site at www.macronix.com or on third-parties websites.

Table 2. Document references

Reference	Document
ArmorFlash whitepaper	Macronix ArmorFlash whitepaper(1)
JESD261	

1. Available on www.mxic.com. Contact Macronix when more information is needed.

Table 3. Open-source software resources

Reference	Open-source software resources
[TF-M]	TF-M (Trusted firmware-M) Arm Limited driven open-source software framework: www.trustedfirmware.org/(1)

1. This URL belongs to a third party. It is active at document publication. However, Macronix shall not be liable for any change, move, or inactivation of the URL or the referenced material.

3. JEDEC security HAL introduction

The JEDEC security HAL is a software component that enables manufacturer-neutral communication with secure functionality with NOR flash memory devices. The HAL can be compiled for use within user space, kernel space, and bare metal applications. The protocols supported by the HAL assume nonvolatile key provisioning operations have previously been put in place so the HAL can also be compiled to run in a trusted execution environment. Shown below are possible locations for the HAL in relation to an operating system (or the absence thereof).

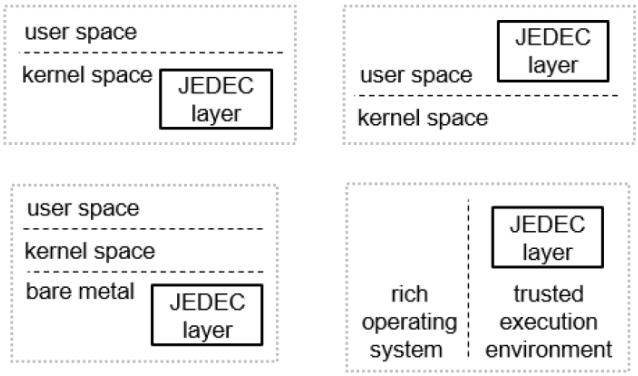

Figure 3-1 System Hierarchy

As shown in Figure 3-1, the JEDEC security HAL is designed to provide a set of device agnostic HAL APIs for upper Application API layer for operations such as secure program or secure erase. The JEDEC vendor driver is used to address differences among various security devices manufactured by different vendors.

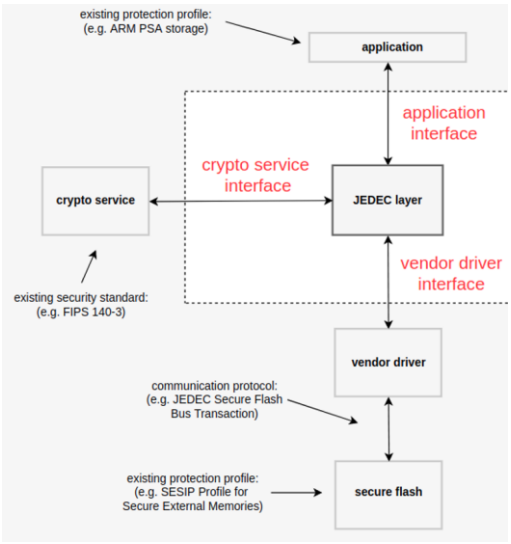

Figure 3-2 Block Diagram Overview of HAL

The JEDEC security HAL specific implementation includes reference source code accompanying the JEDEC common interface layer. Vendors can also have their own implementations fit into this HAL by complying to the function calls required by the JEDEC security HAL layer.

The APIs defined in JEDEC security HAL are shown in Table 4.

Table 4. Application interface

#	API	description
1	create_session	Establish a cryptographic session with a flash device
2	terminate_session	Terminate a cryptographic communication channel (session) with a flash device
3	secure_manage_region	Configure secure regions' security attributes
4	secure_get_regions_info	Obtain the current secure configuration parameters for the flash device
5	secure_init	Move the flash device's state from uninitialized to initialized state
6	secure_uninit	Move the flash device's state from initialized to uninitialized state
7	secure_program	Program data to a flash device
8	secure_erase	Erase one sector in a flash device specified by a base address
9	secure_copy	Copy data from one location in flash to another
10	secure_read	Read read-protected or ciphertext data from a flash device

4. ARM TF-M introduction

TF-M provides a Trusted Execution Environment (TEE) for Arm v7-M and v8-M devices. For Arm v8-M devices, TF-M leverages Arm TrustZone technology, and is the reference implementation of platform security architecture aligning with PSA Certified guidelines.
TF-M provides a highly configurable set of software components to create a Trusted Execution Environment. Secure boot in TF-M ensures integrity of runtime software and supports secure firmware upgrades. Besides, TF-M provides a set of secure runtime services like secure storage services, cryptography services, and attestation services.
These secure services manage critical assets isolated from the non-secure application. The non-secure application cannot directly access any of the critical assets, but can call secure services that use the critical assets. Figure 4-1 shows an example system of TF-M implementation:

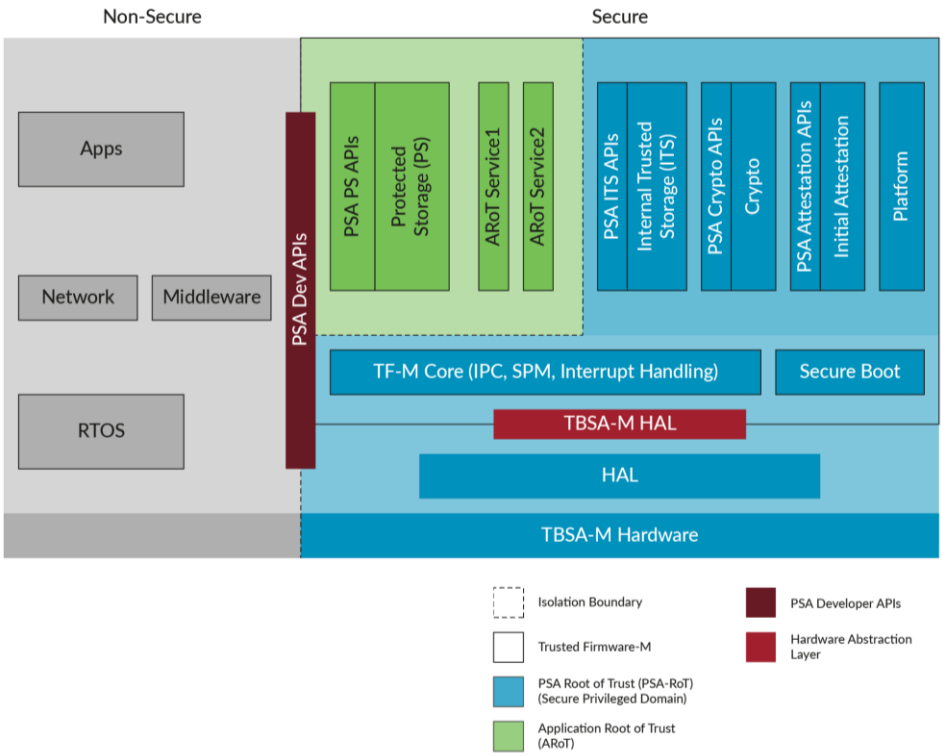

Figure 4-1 An example system of TF-M implementation

TF-M has implemented two types of storage interfaces for use with storage device. One is the PSA Internal Trusted Storage API implemented in TF-M ITS partition to allow the access of assets stored in a microcontroller built-in flash memory region that is isolated from non-secure or from unprivileged applications by hardware security protection mechanism.
The other is PSA Protected Storage API which is intended to protect larger data assets stored in storage media that are external to the MCU package, with a promise of data-at-rest protection, including device-bound encryption, integrity, and replay protection. The current implementation of TF-M Protected Storage (PS) partition mainly protects the normal external flash which is vulnerable to physical attacks.

5. ETSS partition introduction

External Trusted Secure Storage (ETSS) partition proposed by Macronix mainly provides external secure storage services for applications to access assets stored in external secure Flash, as shown in Figure 5-1, this partition is integrated in TF-M framework to ensure that all the operations related to secure Flash are performed in Trusted Execution Environment. The design document and implementation code of ETSS partition V1.0 has been merged in master branch of tf-m-extras git repository. Currently, this partition is maintained as a third-party secure partition attached to trusted-firmware-m git repository.

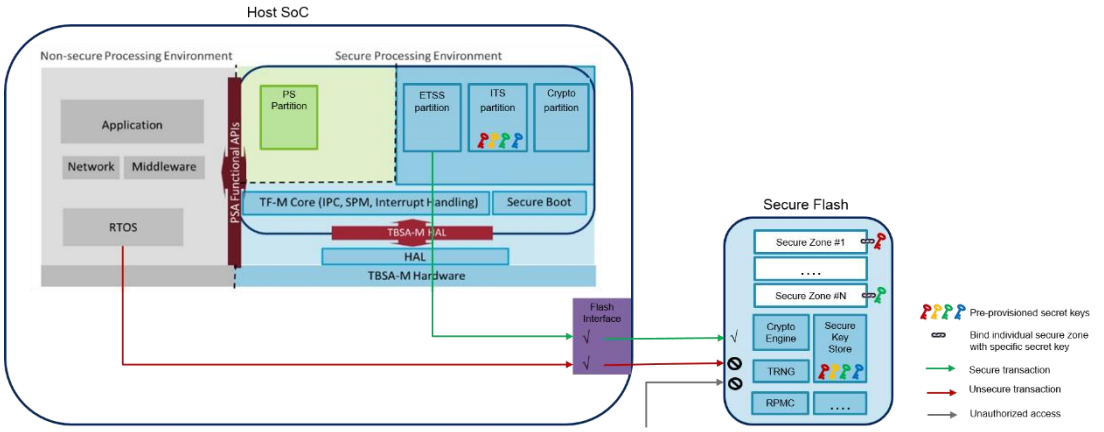

Figure 5-1 Integration of ETSS partition in TF-M framework

Figure 5-2 describes the detailed architecture of ETSS services provided in ETSS partition V1.0 based on secure Flash framework.

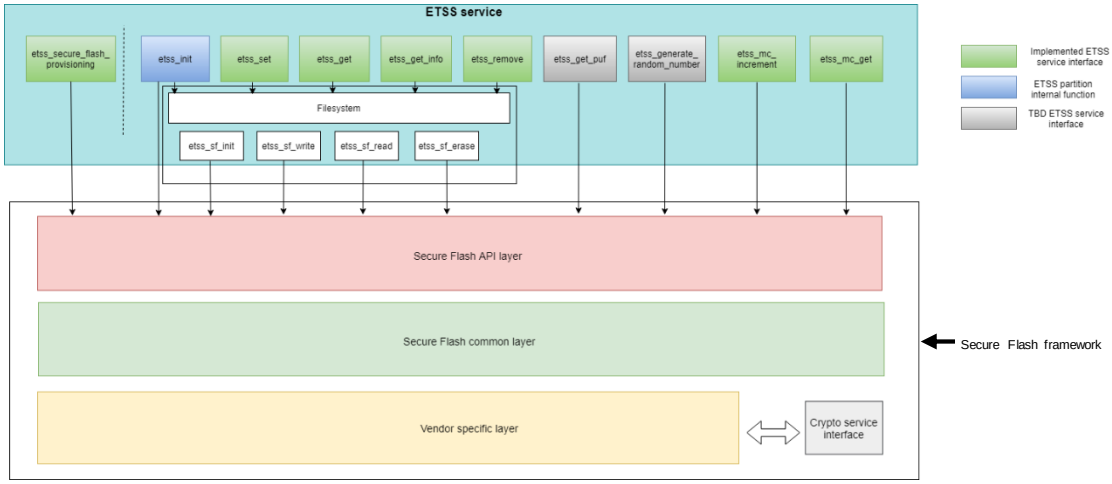

Figure 5-2 Layered architecture of ETSS partition V1.0

As the ETSS partition V1.0 was submitted prior to the publication of JEDEC security HAL standard, the implementation of JEDEC security HAL is not included in the first release of ETSS partition.

6. JEDEC security HAL implementation

6.1 Hardware

This JEDEC security HAL implementation has been tested with Macronix MX78 series ArmorFlash on STM32L562E-DK (Figure 6-1) and Renesas EK-RA6M5 (Figure 6-2) evaluation boards.

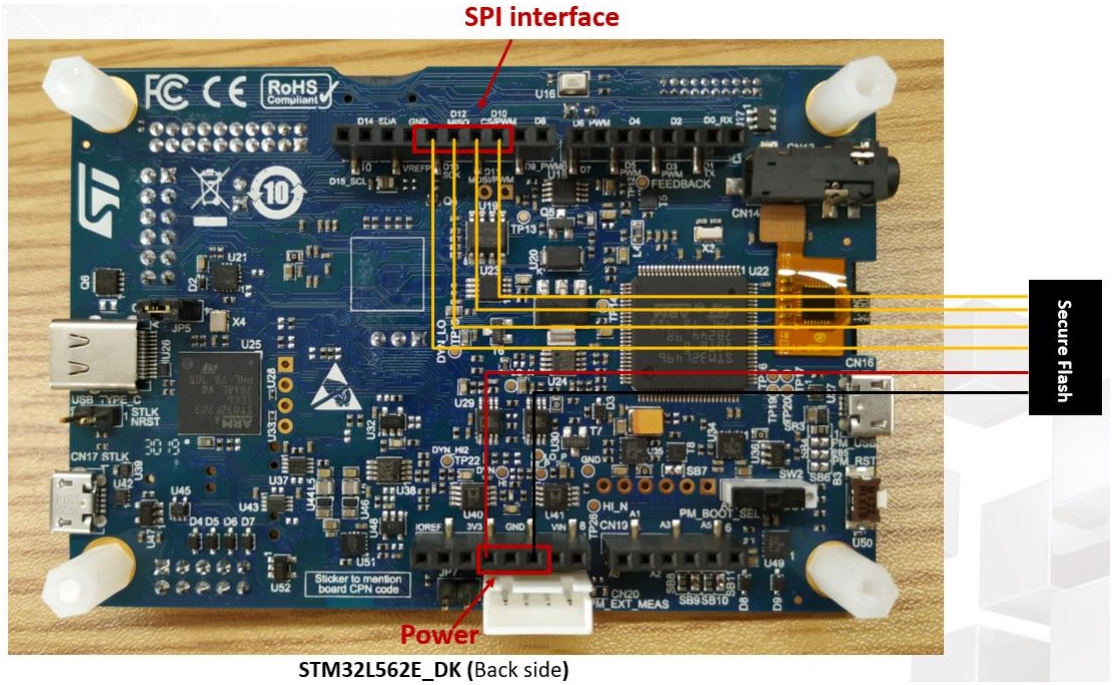

Figure 6-1 SPI interface of STM32L562E_DK evaluation board

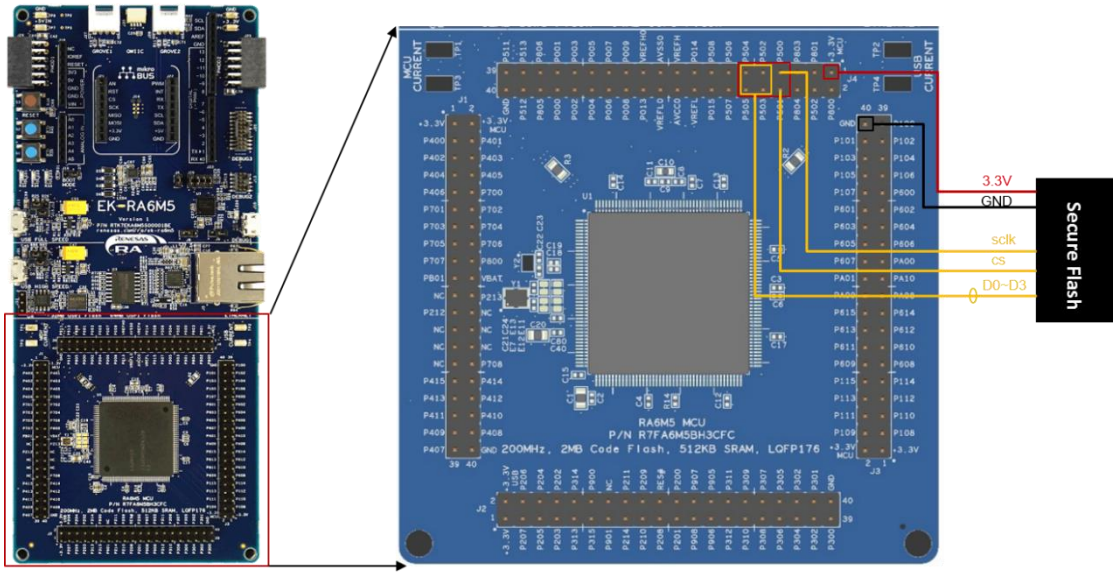

Figure 6-2 QSPI interface of EK-RA6M5 evaluation board

6.2 Software

6.2.1 JEDEC security HAL implementation overview

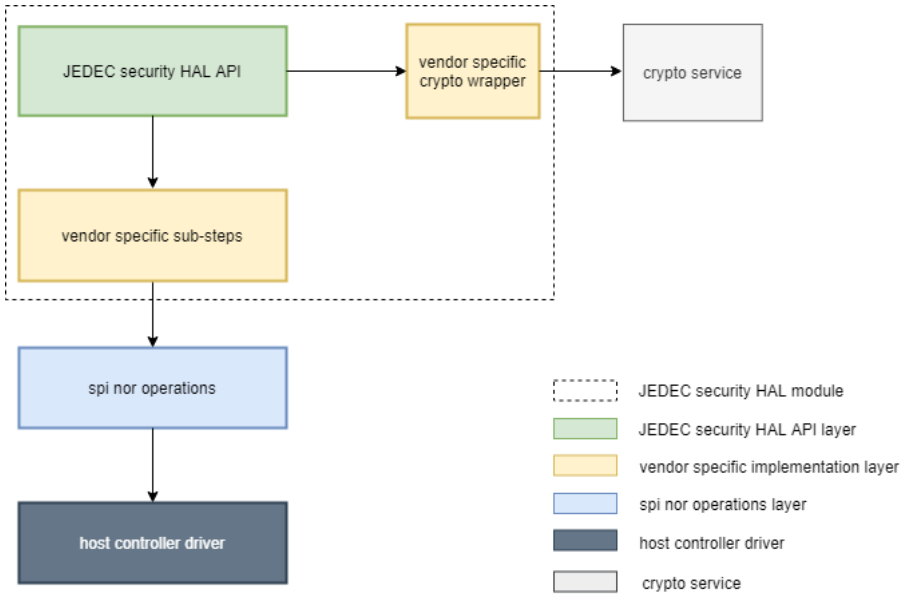

Figure 6-3 JEDEC security HAL implementation architecture

As shown in Figure 6-3, the JEDEC security HAL implementation consists of JEDEC security HAL API layer, and vendor specific implementation layer which consists of vendor specific implementation of JEDEC security HAL sub-steps and crypto wrapper functions. The spi nor operation layer implements normal spi nor flash operations and calls host controller's driver.

6.2.2 JEDEC security HAL implementation file structure

Figure 6-4 shows the file structure of JEDEC security HAL module. It consists of JEDEC security HAL API layer under the folder "JEDEC_security_HAL" and vendor specific implementation layer under the folder "vendor_impl".

The JEDEC security HAL APIs are defined and implemented in jedec_security_hal.h and jedec_security_hal.c files. The queue operations implemented in *queue.c* and *queue.h* files are used by JEDEC security HAL API layer to receive responses from vendor specific implementation layer, which are verified in JEDEC security HAL API layer by calling crypto services. Header file *crypto_wrapper.h* contains crypto wrapper functions declaration, header file *vendor_security_impl.h* contains the prototype of vendor specific operations' function pointer, while corresponding flash vendors' specific implementation are placed under *vendor_impl* folder.

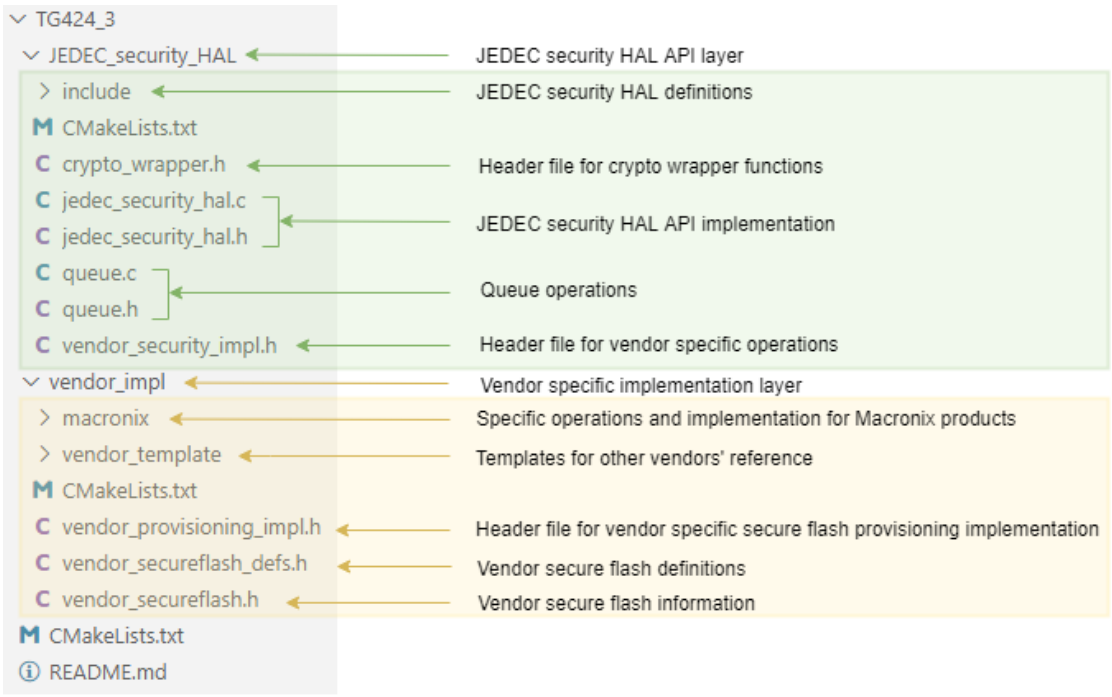

Figure 6-4 File structure of JEDEC security HAL module

7. Integrate JEDEC security HAL in ETSS partition V1.1

The integration of JEDEC security HAL in ETSS partition V1.1 is shown in Figure 7-1, JEDEC security HAL interfaces are called by secure Flash API layer. The intention of secure Flash API layer is to check access requests from applications based on application id and pre-provisioned application information. Each application could only access its pre-binding root key and secure region. Then parameters are passed to JEDEC security HAL layer with calling API interface. The corresponding file structure is shown as Figure 7-2.

Figure 7-1 Integration of JEDEC security HAL in ETSS partition

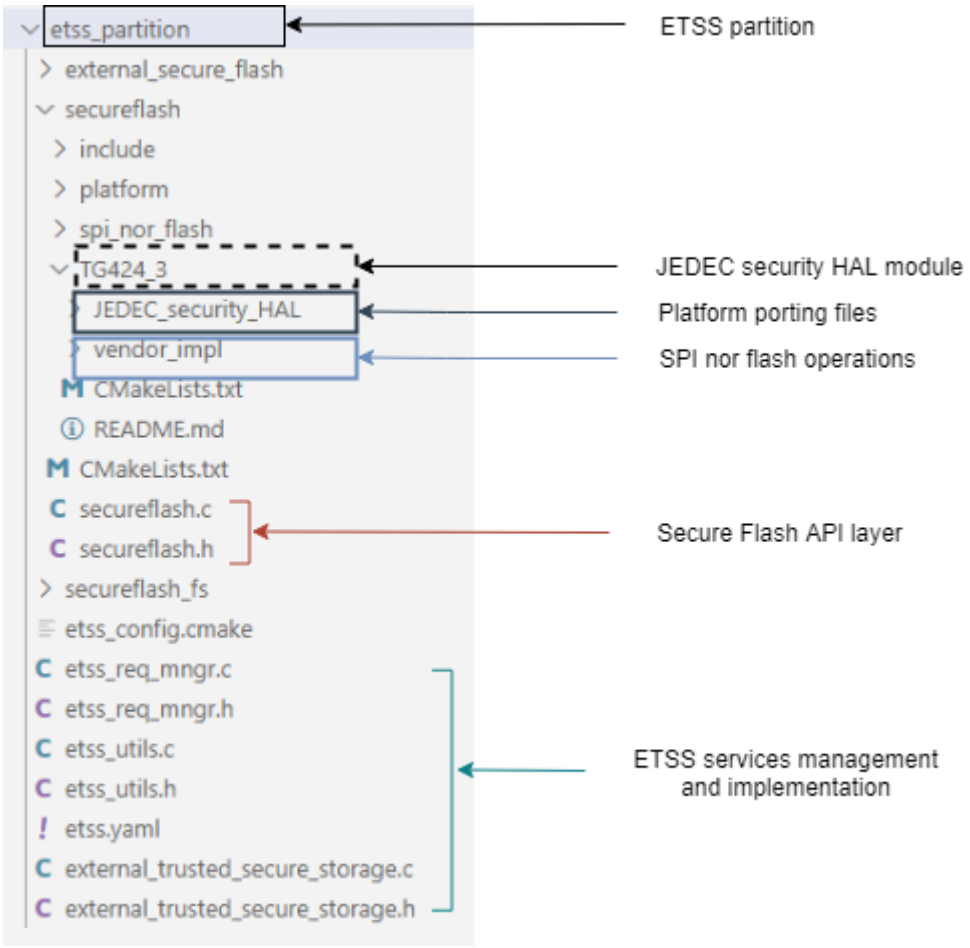

Figure 7-2 ETSS partition file structure

A secure Flash provisioning process should be performed to configure each secure Flash region's security attributes, share binding keys between host MCU/SoC and secure Flash, and grant applications' access rights prior to being put into use.

The ``etss_secure_flash_provisioning`` service is implemented to facilitate secure Flash provisioning in the manufacture process. Specific secure Flash provisioning information can be assigned while calling ``etss_secure_flash_provisioning`` service. As the provisioning of secure Flash is not addressed in JESD261 standard, and the specific provisioning implementation process may vary with security memory vendors and platforms. In this design the specific secure Flash provisioning is implemented by each vendor in vendor specific implementation layer as shown below.

Figure 7-3 Block diagram of secure Flash provisioning

Below figure shows a simplified flow diagram of etss_set service. Application's service request is sent to ETSS partition along with application's unique id (app_id). Secure Flash API layer checks application's access permission based on pre-provisioned information, if application was entitled to access requested secure region, then call JEDEC security HAL APIs to complement subsequent operations.

For more information about the integration of JEDEC security HAL in Arm TF-M with ETSS partition, please refer to the design document of ETSS partition V1.1.

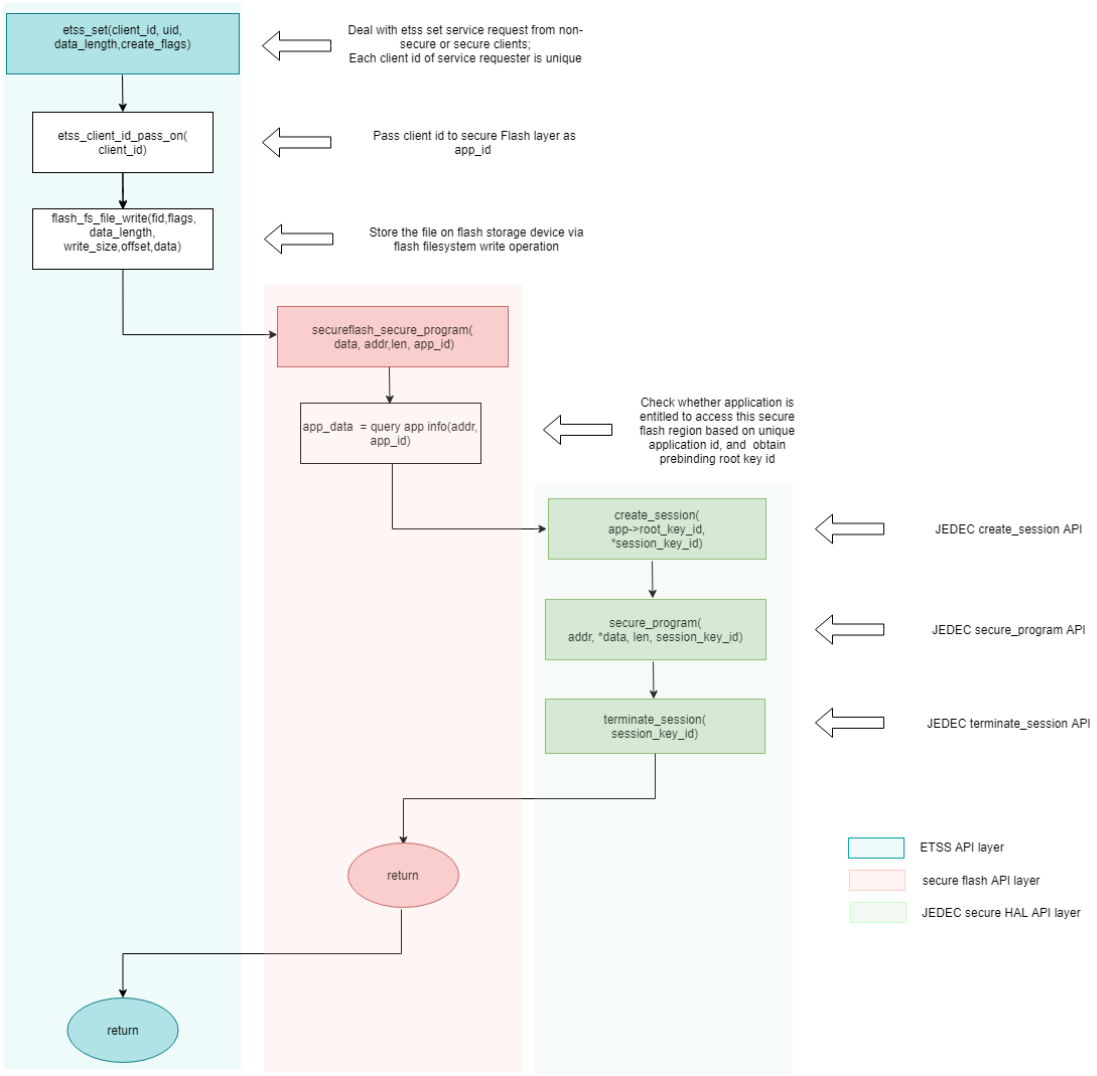

Figure 7-4 Simplified flow diagram of ETSS set implementation

Revision History

Revision No.	Date	Description
1.0	6-Feb-2023	Initial release.

Except for customized products which have been expressly identified in the applicable agreement, Macronix's products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only, and not for use in any applications which may, directly or indirectly, cause death, personal injury, or severe property damages. In the event Macronix products are used in contradicted to their target usage above, the buyer shall take any and all actions to ensure said Macronix's product qualified for its actual use in accordance with the applicable laws and regulations; and Macronix as well as its suppliers and/or distributors shall be released from any and all liability arisen therefrom.

For the contact and order information, please visit Macronix's Web site at: http://www.macronix.com.