Qualcomm Technologies, Inc.

Qualcomm® Secure Processing Unit SPU230 Core Security Target Lite (Public)

80-NU430-6 Rev. H

January 14, 2022

This technical data may be subject to U.S. and international export, re-export, or transfer ("export") laws. Diversion contrary to U.S. and international law is strictly prohibited.

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Revision history

Revision	Date	Description
A	April 2019	Initial release
B	May 2019	In Section 3.2.7, corrected revision for document number 80-PD867-16 from Rev. C to Rev. B.
C	August 2019	Updated for recertification (Phase II).
D	January 2021	Updated final release for recertification (Phase II).
E	February 2021	Renewed the diagram of TOE hardware component
F	July 2021	Updated final release for recertification
G	November 2021	Updated after scheme review
H	January 2022	Clarified SFRs

Contents

Figures

Tables

1 Introduction

1.1 ST reference

"Qualcomm Secure Processing Unit SPU230 Core Security Target Lite, Rev. H, Qualcomm Technologies, Inc., January 14, 2022"

1.2 TOE reference

The target of evaluation (TOE) described in this Security Target is named "Qualcomm Secure Processing Unit SPU230 in SDM855 SoC with MCP version spss.a1.1.2_00100".

1.3 Purpose

This Security Target is defined for the Qualcomm Secure Processing Unit (SPU230) hardware and firmware embedded in the SDM855 host system-on-chip (SoC) combined with a DDR in a package-on-package (PoP) configuration and its corresponding software and associated documentation.

The evaluation considers the SPU hardware and the package which covers the SoC and therefore SPU. The firmware and software comprise the operating system of the Secure Processing Unit and the software API providing cryptographic services to SPU applications. The documentation provided with the TOE describes the configuration requirements by the OEM (SoC integrator) and the usage of the software API by the SPU application developer.

This Security Target includes claims derived from the *Security IC Platform Protection Profile with Augmentation Packages* (BSI-CC-PP-0084-2014).

Because the target of evaluation (TOE) is not a *classical* smartcard IC, additional security functions of the hardware and the operating system have been added to this Security Target.

1.4 Conventions

In the Security Functional Requirements (SFRs) section:

- Assignments are underlined.
- Selections are marked with squared brackets [ ].
- Refinements are written in italic.
- Iterations are identified with an extension of the Security Functional Requirement name.

For example, the requirement in the Standard is written as follows:

> The TSF shall perform [assignment: list of cryptographic operations] in accordance with a specified cryptographic algorithm [assignment: cryptographic algorithm] and cryptographic key sizes [assignment: cryptographic key sizes] that meet the following: [assignment: list of standards].

In this document, the requirement is written as follows:

> The TSF shall perform message authentication code generation in accordance with a specified cryptographic algorithm HMAC using SHA-1 or SHA-256 and cryptographic key sizes 128-, 256-, 384-, 512-bit that meet The Keyed-Hash Message Authentication Code (HMAC) (FIPS PUB 198-1) and Secure Hash Standard (SHS) (FIPS PUB 180-4).

1.5 Technical assistance

For assistance or clarification on information in this document, submit a case to Qualcomm Technologies, Inc. (QTI) at https://createpoint.qti.qualcomm.com/.

If you do not have access to the CDMATech Support website, register for access or send email to support.cdmatech@qti.qualcomm.com.

2 Target of evaluation overview

2.1 Target of evaluation

The TOE is a subsystem called the Secure Processing Unit (SPU) serving as a secure element within a package system-on-chip (SoC). It is designed as a tamperproof device providing secure storage and a secure execution environment for processing of sensitive data and for performing cryptographic operations using protected keys stored in its secure storage. Secure elements can be used for multiple application areas that require a high level of security. Examples are as follows:

- User authentication and password storage
- Content protection
- Payment
- Subscriber identity module (SIM)
- Storage and management of digital identities
- Secure key storage
- Root of trust
- Storage of sensitive user data (e.g., health care records)

The TOE has dedicated interfaces to other components of the SoC, which allow those components to communicate with the TOE and request services from the TOE.

The TOE will allow for dedicated applications to execute on the operating system of the TOE to provide security services as listed above. Those applications are not part of the TOE, but the TOE operating system provides services to verify the integrity and authenticity of such applications using digital signatures.

Concentrating the critical functions of an SoC to such a Secure Element allows for an architecture where the high level of physical protection and protection against dedicated attacks or side channels can be limited to the Secure Element.

The TOE communicates with the other components of the SoC either using shared memory or using shared configuration and status registers (CSRs).

The TOE security functions (TSFs) of the TOE consists of the hardware of the SPU, the firmware of the SPU, and the operating system executing on the SPU.

The hardware of the TSF is internally structured into two main units:

- The Secure Processing Unit, which performs the general operations of the TSF
- The Crypto Management Unit, which performs the cryptographic operations and generates, manages, and protects keys

For a more detailed description of those units, their functions and how they are internally structured, see the Section 3.1 on TOE definition.

2.2 Internal security functions

The TOE implements the following internal security functions:

- Access control to the various memories (OTP, RAM, ROM) and peripherals
- Access control to keys managed in hardware through enforcement of key policy
- Secure boot and secure loading of TOE software stored outside the TOE using the TOE root of trust (ROM code)
- Protection of user data stored outside the TOE
- Secure loading of user applications stored outside the TOE
- Secure update of the TOE software or applications
- Domain separation between applications executed by the TOE
- Security level

2.3 Cryptographic services

The TOE provides cryptographic services using the support of the cryptographic management unit (CMU). Services provided through the API for user applications are as follows:

- Generation of random numbers (used for key generation)
- Secure key storage providing the possibility to have keys stored in the SP-CMU that are not readable by the SP-CPU. The SP-CPU can only request to perform cryptographic operations using those keys.
- Secure key zeroization
- Symmetric encryption and decryption using the following:
  - AES with 128- and 256-bit keys
  - TDES with 3 keying option (168-bit keys)
- Hash functions: SHA-1, SHA-256, SHA-384, SHA-512
- HMAC using keys up to 512-bit length and using SHA-1, SHA-256, SHA-384 or SHA-512
- CMAC with AES using 128- or 256-bit keys
- RSA encryption/decryption and signature/verification up to 2048 bits
- ECC signature/verification and key generation up to 521 bits

2.4 Physical protection

The TOE provides a number of functions and features that are designed to counter physical attacks. Those include the following:

- Memory scrambling/memory encryption
- Side-channel analysis countermeasures
- Fault attacks sensors and countermeasures
- Memory/registers integrity checking
- Design measures
- Package-on-package (PoP) form factor

3 Target of evaluation description

3.1 TOE boundary and interface

The TOE is an independent subsystem that will be integrated in a system-on-chip (SoC) in a manner that is agnostic to the hardware and software implementation details. The TOE serves as an independent root of trust within the SoC. It does not rely on any external entity for any security enforcement, allowing it to be evaluated as a separate entity. It has its own ROM code for secure boot operations.

The TOE and the hardware interfaces to the SoC into which it is integrated are shown in Figure 3-1.

The TOE hardware interfaces are:

- Battery monitor (indicates when power is lost)
- System access to allow SPU to read/write in the SPU shared memory in DDR (SP-sMEM) as well as in SPU protected memory partitions in DDR and NVM.
- Interface to read/write shared configuration and status registers (SP-sCSR) with the SoC. These CSRs are among other things used as doorbell for communication between the SPU and the rest of the SoC
- Interface to the RPM (power management) module to provide SPU power requirement
- Interface to the Clock Controller to provide clock requirement and receive external clock signal

As indicated earlier, the TOE provides a TSF to cryptographically protect user data before, storage in an SPU DDR partition or SPU NVM partition. This TSF also insured that user data read from these locations are trustworthy before processing them internally.

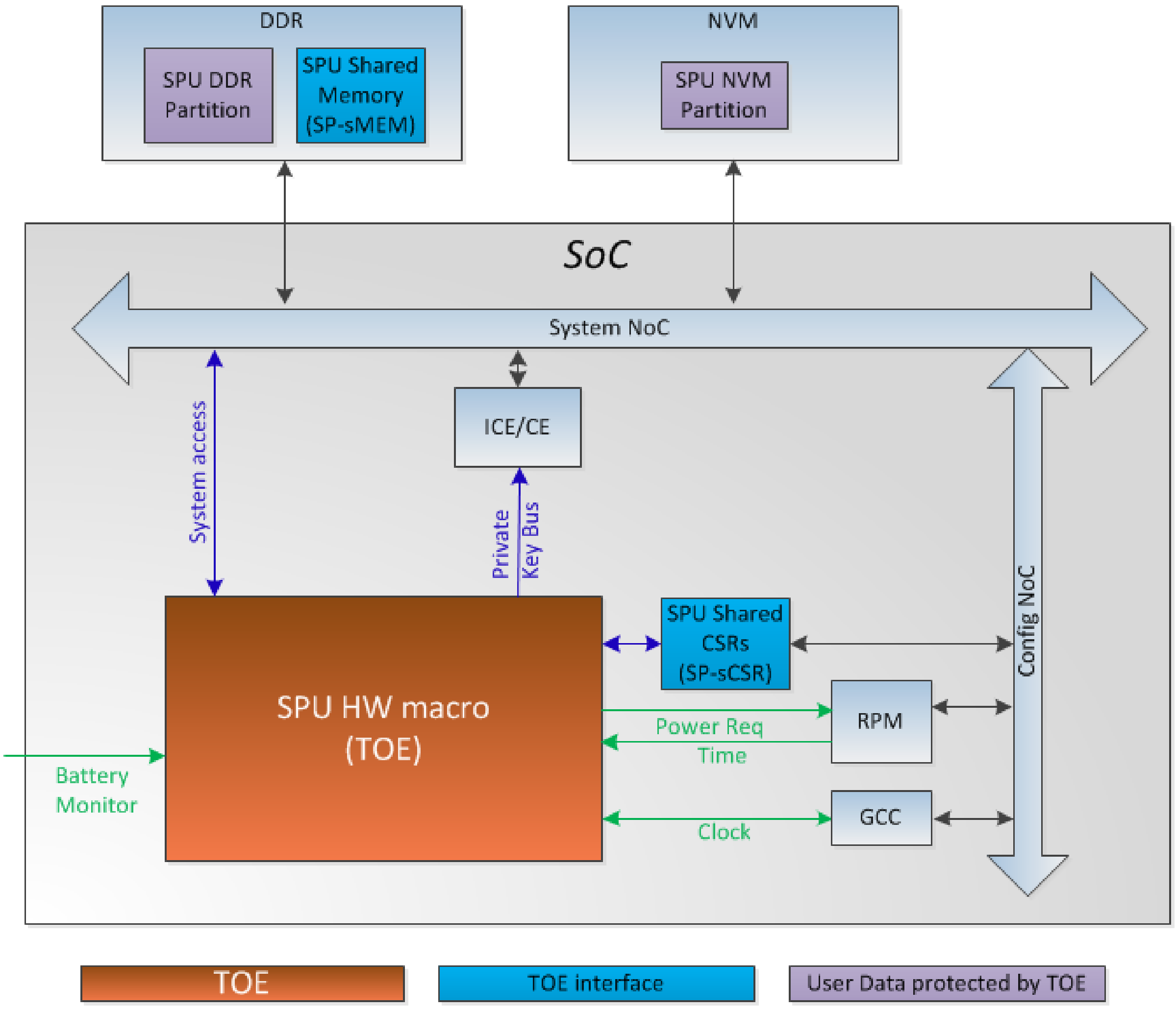

Figure 3-1 TOE components and their interfaces to the SoC

The TOE software interface consists in application programming interfaces providing:

- Communication services
- External memory storage (read/write) services
- Cryptographic services

The TOE communicates with the other components of the SoC via the SPU shared memory and the SPU shared configuration and status registers.

3.2 Scope of the TOE

3.2.1 Overview

The TOE is composed of:

- SPU hardware: a hardmacro synthetized independently of rest of SoC and integrated as a black box
- SPU firmware: ROM code that includes a primary boot loader used to load the software part of the TOE
- SPU software:
  - Main control program (MCP) which provide services for SPU user applications loaded on the TOE platform
  - SPU system applications: contain additional TOE functionalities that are not packaged in the SPU firmware or MCP
- Package: the final device in the field consist of a package-on-package (PoP). On package contains the SoC integrating the SPU hardware and the other package contains the DDR that is needed for SPU to be operational.

Figure 3-2 shows the software components of the TOE as well as the application programming interface of the TOE and used by user application running in the TOE.

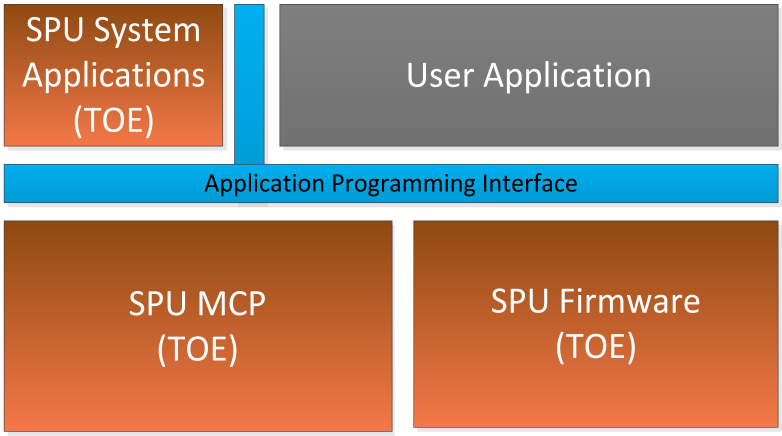
SPU System Applications (TOE)
User Application
Application Programming Interface
SPU MCP (TOE)
SPU Firmware (TOE)

Figure 3-2 TOE software components

3.2.2 Hardware

The physical boundary of the TOE includes the TOE hardware which consists of the SPU hardmacro and the firmware embedded in SP_ROM for the secure boot.

The TOE hardware is structured into the following components (as shown in Figure 3-3):

- Secure central processing unit (SP-CPU), which executes the main code of the TOE
- Cryptographic management unit (SP-CMU), which provides support for random number generation and key generation as well as symmetric and asymmetric cryptographic functions. It also holds a key table (SP-KT)

- Processor interconnect bus, which is used for data exchange between the SP-CPU and the SP-CMU
- Local resource manager (SP-LRM), which provides the interface to the clock, the reset line, and the interface to the sensors (voltage, temperature, logic fault, etc.)
- OTP and security controller (SP-SC) block, which holds the one-time fuse programmable ROM area, including keys that the SP-CPU can request to be used by the SP-CMU but cannot read directly.
- SP-Timer and SP-Watchdog, used to provide timer functionality for the TOE independent from other timers within the SoC
- Memory: SP_RAM and SP_ROM (for the primary boot loader image)
- The always-on island that contains the part of the anti-replay mechanism (SP-AR) and the always-on timer (SP-AOTimer)
- The external memory manager (SP-ExtMM) providing read/write capabilities to TOE external memory

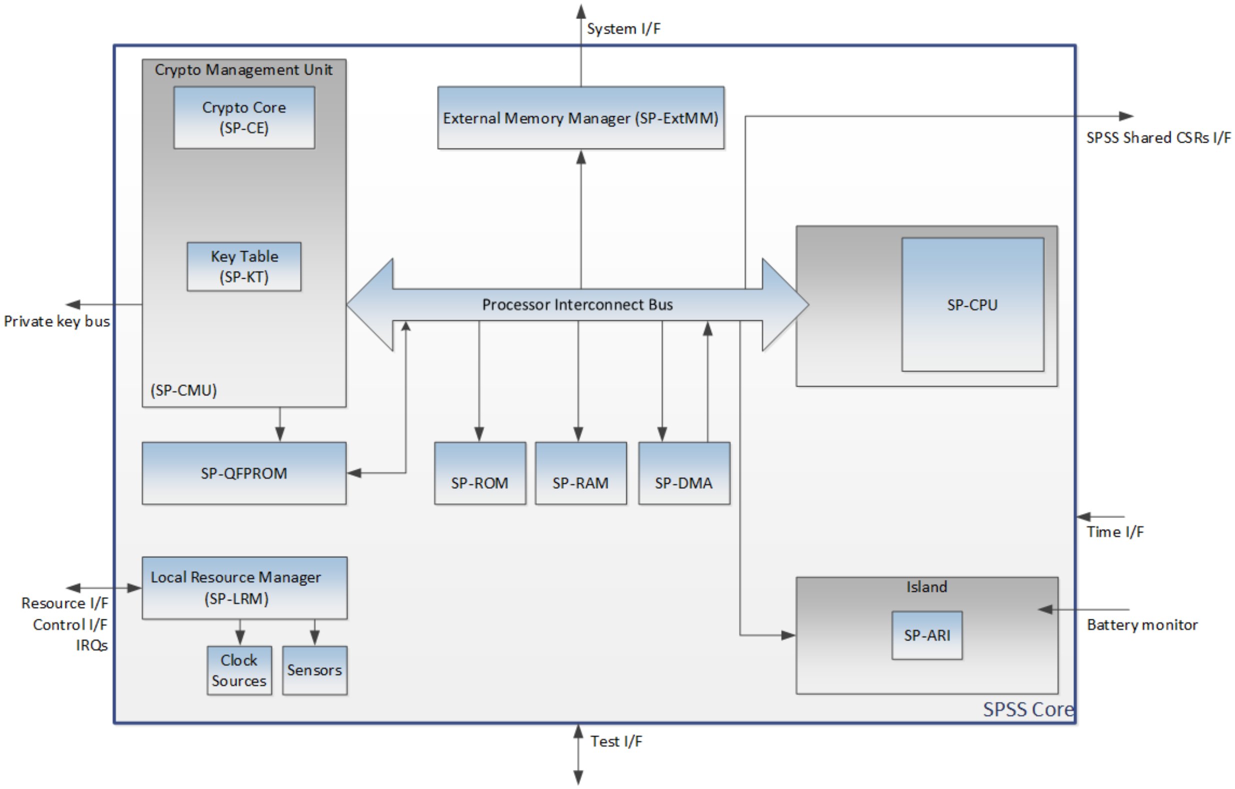

Figure 3-3 TOE hardware components

3.2.2.1 SP-CPU

The SP-CPU is the main processing unit of the TOE. It executes the general firmware and software of the TOE. This CPU provides different operational modes and the ability for memory protection that allows the implementation of a secure operating system that separates unprivileged applications from each other and allows protection of critical resources from direct access by applications without using the operating system services that control access to such critical resources.

3.2.2.2 SP-MPU

The memory management unit of the TOE (SP-MPU) is responsible for controlling access to memory (ROM and RAM) and the DMA controller in accordance to the access control attributes of the memory areas. It is programmed by the SP-CPU from its privileged mode, allowing the operating system running on the SP-CPU to protect memory areas from direct access by applications and protect memory areas assigned to one application from direct access by another application. The SP-MPU is supplemented by two permission checker embedded in the RAM and ROM. These combined hardware modules are referenced by the term SP-MMU in remaining of this document.

3.2.2.3 SP-CMU

The SP-CMU is a separate subsystem within the TOE that is responsible for the cryptographic operations performed by the TOE as well as the generation and protection of key material used for those operations. The SP-CMU subsystem actually acts like a separate hardware security module in a general purpose operating environment. It consists of the following:

- Crypto engine (SP-CE) as the central processing unit of the SP-CMU (which also includes the hardware implementation of the cryptographic coprocessors: AES, SHA-1 and SHA-256)
- Random number generation unit (SP-RNG), which consists of two physical noise sources and a DRBG
- Hardware support for accelerating asymmetric crypto operations (SP-PKA)
- Crypto management controller (SP-CMC), which manages the key storage including the transfer of keys to the key storage

The SP-CMU is programmed by the SP-CPU, which can request operations such as key generation, loading the key, or performing cryptographic operations; the SP-CPU does not have access to the keys themselves that are managed by the SP-CMC (unless the key attributes allow the key to be exported in clear outside the SP-CMU subsystem). The SP-CMC manages the keys stored in the SP-KT, which are the keys private to the TOE.

3.2.2.4 SP-SC

The security control component (SP-SC) contains the SP-QFPROM and is responsible for controlling access to OTP areas there. This includes protection of areas that are write-protected and control access of the SP-CPU to allow it to only access dedicated OTP items that are not indicated as read-protected. Some secret data stored in OTP are stored in encrypted form.

3.2.2.5 SP-LRM

The SP-LRM is responsible for interrupt handling and management of the TOE. The sensors that detect operational problems, faults, or potential attacks are connected to the SP-LRM and cause an interrupt when they detect a problem. The SP-LRM passes the interrupt to the SP-CPU for handling and requires the SP-CPU to clear the interrupt, indicating that it has received and processed the interrupt. Alternatively, the SP-LRM can be configured to perform a cold reset upon specific sensors detection.

Note that internal interrupts of the SP-CPU (such as system tick time expiration, SP-MMU permission error, and privilege exception) are handled directly by the SP-CPU and not by the SP-LRM, but an interrupt caused by the SP-Timer is handled by the SP-LRM.

3.2.3 Firmware, software, and application

The operating system with the software API are considered as logical boundary of the TOE. The operating system manages the access to the services provided by the TOE, implements software countermeasures and controls the applications

The TOE operating system consists of:

- Firmware (ROM)
- Software (stored in NVM)
  - MCP image
  - System applications
    - Cryptoapp
    - Asym_cryptoapp

In a nutshell, the firmware loads MCP which loads system applications.

The TOE contains firmware in SP_ROM that is used for the secure boot process (primary boot loader and its supporting cryptographic libraries). In addition, part of the firmware contains drivers that are used in Operational mode by the loaded software (after the secure boot).

The TOE also contains software (main control program (MCP)) which is stored in external non-volatile memory (NVM) and loaded into SP-RAM at runtime by the aforementioned primary boot loader (PBL). The MCP image is stored signed and encrypted in the external memory. The PBL verifies the signature and decrypts the software each time before the MCP is executed by the SP_CPU.

The operating system (OS) is formed by the MCP, the system applications and the associated drivers in the firmware. The OS is running on the SP-CPU and provides services to user applications loaded for the SP-CPU. The OS verifies the integrity and authenticity and enforce confidentiality of any applications loaded to the TOE including system applications.

The MCP image and the applications are stored in external memory and can be updated by downloading a newer version in the external memory. A set of rollback counters prevent the TOE from loading an older version of MCP or an application.

The operating system is also responsible to separate applications executing on it from each other and control that an application uses only those services and objects it is supposed to use (as defined in the downloaded and signed application package). This operating system is part of the TOE and implements some TSF.

The operating system and system applications provide the following services to applications:

- Cryptographic services (AES, TDES, Hashing, message authentication codes, RSA, ECDSA and ECDH) with keys either held as retained keys within the SP-CMU (in the SP-KT) or with keys provided by the application. If retained keys are used, the operating system verifies that the application is allowed to use those keys and if they are used in accordance with the key attributes. In addition, the TOE provides random number generation services.
- Non-volatile memory storage for user data. User data is stored (in external DDR/NVM) encrypted, authenticated and protected against replay. The TOE maintains a unique key for each application that is used for these cryptographic operations.
- Communication services with external entities (e.g., modem subsystem, HLOS or trusted execution environment).
- Application loading services

The system applications are stored in external non-volatile memory (NVM) and are loaded into SP-RAM at runtime by the aforementioned MCP. The TOE contains one system application:

- Cryptoapp: implements the RSA key generation service (not claimed as security functionality in this Security Target)

3.2.4 Package

The TOE package is shown in an abstracted way, not to scale, and without all connections in Figure 3-4. The package-on-package solution consists of:

- The Package A containing the SoC bare die integrating the SPU hardware
- The Package B containing the DDR bare die required for the system to work

The SoC is integrated into Package A during phase 4 (see 3.2.10 TOE lifecycle).

The package B is stacked on Package A during phase 6 (see 3.2.10 TOE lifecycle).

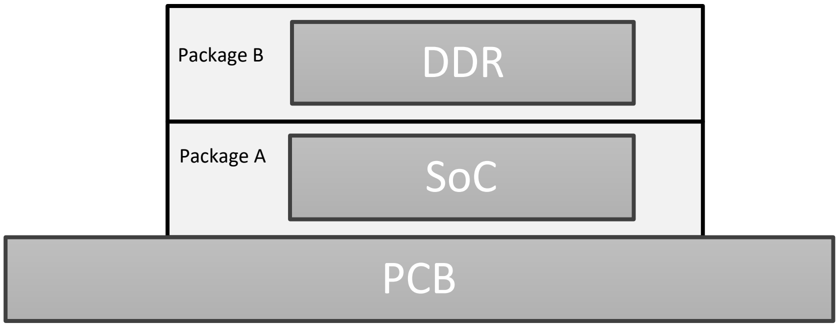

Figure 3-4 TOE package

3.2.5 Guidance documentation

The API for the use of the services of the secure processor as well as associated guidance are provided with the software development kit.

3.2.6 Forms of delivery

The TOE hardware and firmware (ROM) including the personalization data (in SP-QFPROM) will be delivered in the form of a partly packaged SoC to the device manufacturers to integrate the SoC into his devices.

The integration include a step of adding a DDR package on top of the partly packaged SoC. This is called a package-on-package (PoP).

In addition, the device manufacturers will receive the TOE software (MCP image and system applications) as part of an overall Qualcomm software package for the SoC. The TOE software will be loaded by the device manufacturer in the device non-volatile memory (e.g., flash memory).

Delivery protection for all TOE components is covered by ALC_DEL and ALC_DVS.

3.2.7 TOE configuration

Table 3-1 TOE configuration

Item type	Item	Version/label	Form of delivery
Hard macro	SPU230 hard macro (GDS) containing hardware design and ROM	3.1	GDS
Hardware	SoC embedding the SPU hard macro SDM855	V2.2	bare die
Hardware	Party packaged SoC	PX90-PC761-5	Packaged bare die without DDR package on top [1]
Firmware	ROM image - Secure boot loader (PBL) and platform API code (MissionROM)	MissionROM: ROM_V2_BINARIES_0 PBL: SDM855_SPSS_PBL_V2	Included in SPU hard macro ROM
Software	MCP image	spss.a1.1.2_00100	Software image encrypted and signed
Software	System application - cryptoapp image	spss.a1.1.2_00100	Software image encrypted and signed
Software	System application - asym_cryptoapp image	spss.a1.1.2_00100	Software image encrypted and signed
Document	80-PD867-16 Secure Processing Unit ARI	B	PDF
Document	80-PF777-965 SM8150 Qualcomm SPU Enablement UG	AC	PDF
Document	Qualcomm SPU Core – Hana API	V4.6	PDF

[1] The final TOE is the partly packaged SoC delivered to the OEM combined with a packaged DDR (according to the requirements of the User Guidance 80-PF777-965) resulting in a Package on Package (PoP) form factor.

3.2.8 TOE initialization

The TOE is provisioned with individual keys and transition to operational state during Final Test in OSAT premises

The TOE can only boot fully after it has been integrated in a device containing the software (MCP image) during the composite product integration phase.

After composite product integration and during operational usage the TOE performs the following action upon boot:

- Lifecycle control
- SPU230 configuration
- Software (MCP image) loading in SPU internal memory (SP_RAM), signature verification and decryption
- Software (MCP image) execution
- System applications loading in SPU internal memory (SP_RAM), signature verification and decryption

3.2.9 TOE integration

The TOE is an integrated part of a larger SoC that itself is intended to be integrated into mobile or other devices.

3.2.10 TOE lifecycle

The lifecycle of the TOE has been modified (refined with additional phase) compared to PP0084 to match our TOE lifecycle.

The lifecycle control of the TOE ensures that a device in Test mode cannot run the TOE software and limits access to the TOE firmware (to the minimal set of code required to boot).

The lifecycle control of the TOE ensures that the device is in Perso mode before TOE initialization and pre-personalization (phase 5) is performed. Initialization includes static data provisioning while pre-personalization includes per chip data provisioning (such as keys)

The lifecycle control of the TOE ensures that TOE data (stored during phase 5) and user data (stored during phase 7) are protected in Operational mode.

The process of developing and manufacturing a composite product that contains the TOE is shown in Figure 3-5.

The firmware and software development is done as part of the TOE development and the firmware is internally delivered for the integration into the ROM.

The IC packaging phase embeds the SoC into PCBs on both sides that allow the addition of the DDR and the integration into the final product. DDR addition and integration into the final product is done in Phase 6.

TOE personalization includes the provisioning of keys that allow customers to perform further personalization of their SPU applications while the TOE is in Operational mode.

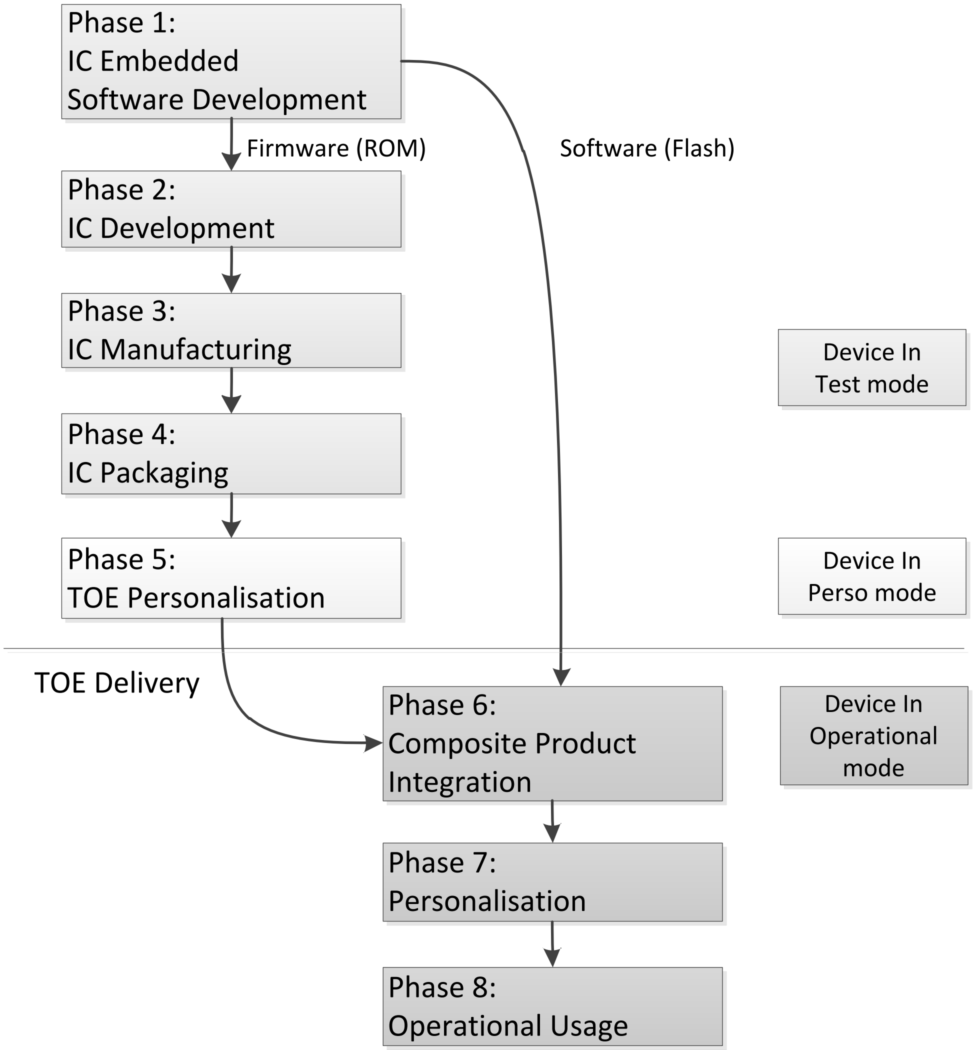

Figure 3-5 TOE lifecycle

4 Conformance claims

4.1 Common criteria (CC) claims

This Security Target and the TOE claim conformance to *Common Criteria for Information Technology Security Evaluation*, Parts 1, 2, and 3, Version 3.1, Revision 5, which is referred to in this document as [CC].

The Security Target conformance claimed is: Part 1 conformant, Part 2 extended, Part 3 conformant.

4.2 IC platform protection (ICPP) claims

This Security Target claims strict conformance to *Security IC Platform Protection Profile with Augmentation Packages*, Version 1.0 (BSI-CC-PP-0084-2014), which is referred to in this document as [ICPP].

4.3 Package claims

The packages for AES, TDES and Hash functions from [ICPP] have been included.

The Security Target claims conformance to EAL4 augmented by ALC_DVS.2 and AVA_VAN.5.

4.4 Conformance claim rationale

The TOE specified in this Security Target is a self-sufficient component of a packaged Qualcomm® Snapdragon™ system-on-chip (SoC). Apart from access to memory, power supply and the connectivity to consumers of the TOE functionality, the remainder of the SoC does not support any aspect of the operation of the TOE.

The TOE can therefore be regarded as a Security Integrated Circuit and its package which implements all functional aspects specified by the protection profile. The TOE provides different types of cryptographic services to external entities, stores and generates keys which can be used for different cryptographic operations. The keys stored and processed by the TOE are protected against logical or physical attacks.

In addition, the development and production lifecycle specified by the protection profile is consistent with the one applicable to the TOE.

This allows the conclusion that the protection profile with its intended use cases is applicable to the TOE.

5 Security problem definition

5.1 Definition of assets

The assets to be protected are as follows (as specified in the [ICPP]):

- User data stored in or processed by the TOE
- User data stored outside the TOE while under the control of the TOE (this covers data exported through the non-volatile memory driver but not the data exported on the communication driver)
- Security IC Embedded Software, stored and in operation
- Security services provided by the TOE to applications executed in the TOE

Components of the SoC that are not part of the TOE are considered external entities and they can communicate with the TOE in a similar way an external entity communicates with a smartcard. The communication between those components and the TOE is via the shared CSRs, interrupts, and the shared memory areas. The CSRs act as mailboxes where the other components send requests to the TOE, and the shared memory areas are used for bulk data transfer required to process those requests.

More precisely, the TOE assets that require protection are as follows:

- Root keys
- Keys derived from root keys
- Public keys used for code loading
- Revision
- Lifecycle state data
- Code execution control
- Code integrity, authenticity, confidentiality and rollback prevention (load and runtime)
- Data integrity, authenticity, confidentiality and replay protection (load and runtime)
- Debug/Test mode/interface

5.2 Threats

The following threats are defined in [ICPP].

Table 5-1 Threats

Threat	Description
T.Leak-Inherent	Inherent Information Leakage An attacker might exploit information, which is leaked from the TOE during usage of the Security IC, to disclose confidential user data as part of the assets.
T.Phys-Probing	Physical Probing An attacker might perform physical probing of the TOE (i) to disclose user data while stored in protected memory areas, (ii) to disclose/reconstruct user data while processed, or (iii) to disclose other critical information about the operation of the TOE to enable attacks disclosing or manipulating user data of the Composite TOE or the Security IC Embedded Software.
T.Malfunction	Malfunction due to Environmental Stress An attacker might cause a malfunction of TSF or the Security IC Embedded Software by applying environmental stress to (i) modify security services of the TOE, (ii) modify functions of the Security IC Embedded Software, or (iii) deactivate or affect security mechanisms of the TOE to enable attacks disclosing or manipulating user data of the Composite TOE or the Security IC Embedded Software. This might be achieved by operating the Security IC outside normal operating conditions.
T.Phys-Manipulation	Physical Manipulation An attacker might physically modify the Security IC to (i) modify user data of the Composite TOE, (ii) modify the Security IC Embedded Software, (iii) modify or deactivate security services of the TOE, or (iv) modify security mechanisms of the TOE to enable attacks disclosing or manipulating user data of the Composite TOE or the Security IC Embedded Software.
T.Leak-Forced	Information Leakage An attacker might exploit information, which is leaked from the TOE during usage of the Security IC, to disclose confidential user data of the Composite TOE as part of the assets even if the information leakage is not inherent but caused by the attacker.
T.Abuse-Func	Abuse of Functionality An attacker might use functions of the TOE that might not be used after TOE Delivery to (i) disclose or manipulate user data of the Composite TOE, (ii) manipulate (explore, bypass, deactivate, or change) security services of the TOE, or (iii) manipulate (explore, bypass, deactivate, or change) functions of the Security IC Embedded Software, or (iv) enable an attack disclosing or manipulating user data of the Composite TOE or the Security IC Embedded Software.
T.RND	Deficiency of Random Numbers An attacker might predict or obtain information about random numbers generated by the TOE security service, for instance, because of a lack of entropy of the random numbers provided.
The following threat is not part of [ICPP] and has been added:	
T.Boot-Compromise	Compromising the Boot Functionality An attacker might attempt to interfere with the boot process by attempting to boot TSF software not authorized by the TOE.
T.CONFID-TSF-CODE	The attacker executes an application without authorization to disclose the TSF software.

Threat	Description
T.CONFID-APPLI-DATA	The attacker executes an application without authorization to disclose data belonging to another application.
T.CONFID-TSF-DATA	The attacker executes an application without authorization to disclose data belonging to the TSF.
T.INTEG-APPLI-CODE	The attacker executes an application to alter (part of) its own or another application's code.
T.INTEG-TSF-CODE	The attacker executes an application to alter (part of) the TSF software.
T.INTEG-APPLI-DATA	The attacker executes an application to alter (part of) another application's data.
T.INTEG-TSF-DATA	The attacker executes an application to alter (part of) TSF data.
T.AUTH-TSF-DATA	The attacker replaces (part of) TSF data with (part of) TSF data from another device
T.AUTH-APPLI-DATA	The attacker replaces (part of) application data with (part of) application data from another device
T.RBP-TSF-DATA	The attacker performs a rollback operation on (part of) TSF data (replay an older version)
T.RBP-APPLI-DATA	The attacker performs a rollback operation on (part of) application data (replay an older version)

Threat agents that must be considered are:

- Software executing as unprivileged software on the SP-CPU, attempting to attack the functionality of the TOE or gain information by observing the behavior of the TOE. Only software analyzed vetted by Qualcomm can be loaded into the TOE. Thus such software is considered to still behave benign, i.e., it does not maliciously and deliberately try to subvert the security functionality of the TOE. To keep the evaluation to a reasonable effort, such software components are not included into the TOE.
- Hardware or software executing on other components of the SoC, attempting to attack the functionality of the TOE or gain information by observing the behavior of the TOE
- External entities accessing the SoC via its external interfaces
- External attackers that physically probe the TOE
- External attackers that attempt to gain access to critical information by observing the behavior of the TOE

5.3 Organizational security policies

The following organizational security policies are defined in [ICPP].

Table 5-2 Organizational security policies

Policy	Description
P.Process-TOE	Identification during TOE Development and Production An accurate identification must be established for the TOE. This requires that each instantiation of the TOE carries this unique identification.
P.Crypto-Service	Cryptographic services of the TOE The TOE provides secure hardware based cryptographic services for the IC Embedded Software. Application Note: the following crypto services are supported by hardware: AES, SHA-1, SHA-256, SHA-384, SHA-512, HMAC , CMAC AES and KDF and the asymmetric accelerator for ECDH/ECDSA, RSA_SIGN and RSA_ENC (for all modes, please refer to appendix B).
One organizational security policy has been added that is not included in [ICPP]:	
P.Least-Privilege	Least Privilege for TSF Components The TSF itself is structured into a number of components where some components have their own internal functions and data that is not directly accessible by other components of the TSF. This prohibits that a potential breach, e.g., by an application software executing on the SP-CPU, would give an attacker direct access to critical data such as keys.

5.4 Security assumptions

The following security assumptions are defined in [ICPP].

Table 5-3 Security assumptions

Assumption	Description
A.Process-Sec-IC	Protection during Packaging, Finishing, and Personalization It is assumed that security procedures are used after delivery of the TOE by the TOE Manufacturer up to delivery to the end-consumer to maintain confidentiality and integrity of the TOE and of its manufacturing and test data (to prevent any possible copy, modification, retention, theft, or unauthorized use). This means that the Phases after TOE Delivery (refer to Sections 1.2.2 and 7.1 in [ICPP]) are assumed to be protected appropriately. For a preliminary list of assets to be protected, refer to paragraph 96 (page 29 in [ICPP]).
A.Resp-Appl	Treatment of User Data of the Composite TOE All user data of the Composite TOE are owned by Security IC Embedded Software. Therefore, it must be assumed that security relevant user data of the Composite TOE (especially cryptographic keys) are treated by the Security IC Embedded Software as defined for its specific application context.

6 Security objectives

The Security objectives are taken from [ICPP] with some additional security objectives added.

6.1 TOE security objectives

Table 6-1 TOE security objectives

Objective	Description
O.Leak-Inherent	Protection against Inherent Information Leakage The TOE must provide protection against disclosure of confidential data stored and/or processed in the Security IC by measurement and analysis of the shape and amplitude of signals (e.g., on the power, clock, or I/O lines) and by measurement and analysis of the time between events found by measuring signals (for instance, on the power, clock, or I/O lines).
O.Phys-Probing	Protection against Physical Probing The TOE must provide protection against disclosure/reconstruction of user data while stored in protected memory areas and processed or against the disclosure of other critical information about the operation of the TOE. This includes protection against measuring through galvanic contacts, which is direct physical probing on the chips surface except on pads being bonded (using standard tools for measuring voltage and current) or measuring not using galvanic contacts but other types of physical interaction between charges (using tools used in solid-state physics research and IC failure analysis) with a prior reverse-engineering to understand the design and its properties and functions. The TOE must be designed and fabricated so that it requires a high combination of complex equipment, knowledge, skill, and time to derive detailed design information or other information that could be used to compromise security through such a physical attack.
O.Malfunction	Protection against Malfunctions The TOE must ensure its correct operation. The TOE must indicate or prevent its operation outside the normal operating conditions where reliability and secure operation have not been proven or tested. This is to prevent malfunctions. Examples of environmental conditions are voltage, clock frequency, temperature, or external energy fields.
O.Phys-Manipulation	Protection against Physical Manipulation The TOE must provide protection against manipulation of the TOE (including its software and TSF data), the Security IC Embedded Software, and user data of the Composite TOE. This includes protection against reverse-engineering (understanding the design and its properties and functions), manipulation of the hardware and any data, as well as undetected manipulation of memory contents.

Objective	Description
O.Leak-Forced	Protection against Forced Information Leakage The Security IC must be protected against disclosure of confidential data processed in the Security IC (using methods as described under O.Leak-Inherent) even if the information leakage is not inherent but caused by the attacker by forcing a malfunction (see O.Malfunction: Protection against Malfunctions) and/or by a physical manipulation (see O.Phys-Manipulation: Protection against Physical Manipulation). If this is not the case, signals that normally do not contain significant information about secrets could become an information channel for a leakage attack.
O.Abuse-Func	Protection against Abuse of Functionality The TOE must prevent that functions of the TOE, some of which might not be used after TOE Delivery, can be abused to (i) disclose critical user data of the Composite TOE, (ii) manipulate critical user data of the Composite TOE, (iii) manipulate Security IC Embedded Software, or (iv) bypass, deactivate, change, or explore security features or security services of the TOE. Details depend, for instance, on the capabilities of the Test Features provided by the IC Dedicated Test Software, which are not specified here.
O.Identification	TOE Identification The TOE must provide means to store Initialization Data and Pre-personalization Data in its non-volatile memory. The Initialization Data (or parts of it) are used for TOE identification.
O.RND	Random Numbers The TOE will ensure the cryptographic quality of random number generation. For instance, random numbers shall not be predictable and shall have a sufficient entropy. The TOE will ensure that no information about the produced random numbers is available to an attacker because they might be used, for instance, to generate cryptographic keys.
O.AES	Cryptographic service AES The TOE provides secure hardware based cryptographic services for the AES for encryption and decryption.
O.TDES	Cryptographic service Triple-DES The TOE provides secure software based cryptographic services implementing the Triple-DES for encryption and decryption
O.SHA	Cryptographic Service Hash function The TOE provides secure hardware-based cryptographic services for secure hash calculation.
Security Objectives in addition to the ones defined in [ICPP]:	
O.Defense-in-Depth	Defense-In-depth The TOE shall ensure that critical functions and TSF data cannot be accessed by a simple breach of security of the software executing on the SP-CPU.
O.Secure-Boot	Secure Boot Process The TOE shall ensure that only authorized software is loaded during the boot process after the integrity and authenticity of that software has been verified.
O.RSA_SIGN	The TOE provides secure cryptographic services implementing the RSA algorithm for signature generation and verification. This implementation uses dedicated hardware support provided by the TOE.
O.RSA_ENC	The TOE provides secure cryptographic services implementing the RSA algorithm for encryption/decryption This implementation uses dedicated hardware support provided by the TOE.

Objective	Description
O.ECDSA	The TOE provides secure cryptographic services implementing the ECDSA algorithm based on the NIST curves P-192, P-224, P-256, P-384, P-521 for signature generation and verification. This implementation uses dedicated hardware support provided by the TOE.
O.ECDH	The TOE provides secure cryptographic services implementing the ECDH algorithm based on the NIST curves P-192, P-224, P-256, P-384, P-521 for key generation and shared secret generation. This implementation uses dedicated hardware support provided by the TOE.
O.HMAC	Cryptographic Service HMAC Code Function The TOE provides secure hardware-based cryptographic services for HMAC calculation.
O.CMAC	Cryptographic Service CMAC Function The TOE provides secure hardware-based cryptographic services for CMAC calculation.
O.KDF	The TOE provides secure cryptographic services implementing the Key Derivation Function algorithm based on NIST SP 800-108. This implementation uses dedicated hardware support provided by the TOE.
O.OSData.Access	Restriction of access to data maintained by operating system The TSF must restrict access of software applications exclusively to its own data stored in volatile and non-volatile memory.
O.OSData.Protect	Protection of data maintained by operating system The TSF must protect TSF code, TSF data, application code and application data using cryptographic functions with a cryptographic strength commensurate with the value of the data against loss of confidentiality, integrity, authenticity and against rollback.
O.Reallocation	Reallocation of resources The TOE shall ensure that the re-allocation of an internal memory block for the runtime areas maintained by the TOE operating system does not disclose any information that was previously stored.

6.2 Development and operational environment security objectives

Table 6-2 Development and operational environment security objectives

Objective	Description
OE.Process-Sec-IC	Protection during Composite Product Manufacturing Security procedures shall be used after TOE Delivery up to delivery to the end-consumer to maintain confidentiality and integrity of the TOE and of its manufacturing and test data (to prevent any possible copy, modification, retention, theft, or unauthorized use). This means that Phases after TOE Delivery up to the end of Phase 6 (refer to Section 1.2.3 of [ICPP]) must be protected appropriately. For a preliminary list of assets to be protected, see Section 5.1.
OE.Resp-Appl	Treatment of user data of the Composite TOE Security relevant user data of the Composite TOE (especially cryptographic keys) are treated by the Security IC Embedded software as required by the security needs of the specific application context.

6.3 Security objectives rationale

For the threats, assumptions, organizational security policies, and security objectives that are listed in [ICPP], the rationale described there applies. This includes the policy P.Crypto-Service and the included objectives O.TDES , O.AES and O.SHA.

There have been the following threats, assumptions, organizational security policies, and objectives added in this Security Target:

- Threats:
  - T.Boot-Compromise
  - T.CONFID-TSF-CODE
  - T.CONFID-APPLI-DATA
  - T.CONFID-TSF-DATA
  - T.INTEG-APPLI-CODE
  - T.INTEG-TSF-CODE
  - T.INTEG-APPLI-DATA
  - T.INTEG-TSF-DATA
  - T.RBP-TSF-DATA
  - T.RBP-APPLI-DATA
  - T.AUTH-TSF-DATA
  - T.AUTH-APPLI-DATA
- Assumptions:
  - None
- Organizational security policies:
  - P.Least-Privilege
- Security objectives:
  - O.Defense-in-Depth
  - O.Secure-Boot
  - O.RSA_SIGN
  - O.RSA_ENC
  - O.ECDSA
  - O.ECDH
  - O.KDF
  - O.HMAC
  - O.CMAC
  - O.OSData.Access
  - O.OSData.Protect
  - O.Reallocation

The threat T.Boot-Compromise is addressed by the security objective O.Secure-Boot, which requires that the software loaded during the boot process is verified for its authenticity and integrity before it is executed.

The threats T.CONFID-TSF-CODE, T.CONFID-TSF-DATA, T.INTEG-TSF-CODE, T.INTEG-TSF-DATA, T.RBP-TSF-DATA and T.AUTH-TSF-DATA are addressed by the security objective of O.OSData.Protect that requires that TSF data maintained by the operating system is protected with cryptographic mechanisms including encryption, MAC and replay counter to prevent unauthorized disclosure or modification or exchange. The threats are derived from [JCSPP].

The threats T.CONFID-APPLI-DATA and T.INTEG-APPLI-DATA are addressed by the security objective O.OSData.Access limiting applications to access only their own data maintained by the operating system. The threats are derived from [JCSPP].

The threats T.CONFID-APPLI-DATA, T.INTEG-APPLI-CODE, T.INTEG-APPLI-DATA, T.RBP-APPLI-DATA and T.AUTH-APPLI-DATA are addressed by the security objective O.OSData.Protect requires that user data maintained by the operating system is protected with cryptographic mechanisms to prevent unauthorized access. The threats are derived from [JCSPP].

The threat of T.CONFID-APPLI-DATA, T.INTEG-APPLI-DATA, T.CONFID-TSF-DATA, and T.INTEG-TSF-DATA are addressed by the security objective O.Reallocation requiring the operating system to clear internal memory resources upon reallocation. The threats and objective are derived from [JCSPP].

The organizational security policy P.Least-Privilege is addressed by the additional security objective O.Defense-in-Depth, which requires that critical resources are protected by more than just one ring of protection.

The organizational security policy P.Crypto-Service is used as intended in [ICPP]. P.Crypto-Service implements a number of cryptographic functions supported by the hardware platform.

7 Extended component definition

This Security Target uses the extended components defined in [ICPP]:

- FCS_RNG.1
- FMT_LIM.1
- FMT_LIM.2
- FAU_SAS.1
- FDP_SDC.1

For the complete specification and justification of the extended SFRs, refer to [ICPP].

The following additional extended components are defined for this Security Target.

7.1 FMT_CMT control over management by TSF components

7.1.1 Family behavior

This family allows the specification of defined TSF components that take control over the management of TSF data.

The main difference between the existing components of Part 2 of [CC] and the one defined here is that the management function is not restricted to a specified role, but is restricted to defined TSF components.

This SFR does not have any dependencies on other management SFRs as it does not require administrative intervention since the management mechanism is hard-coded into the TSF.

7.1.2 Component leveling

FMT_CMT.1 Management of TSF data allows dedicated TSF components to manage TSF data.

7.1.3 Management: FMT_CMT.1

There are no management activities foreseen.

7.1.4 Audit: FMT_CMT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

- Basic: All modifications to the values of TSF data.

7.1.5 FMT_CMT.1 management of TSF data by TSF components

Hierarchical to: No other components.

Dependencies: No other components

FMT_CMT.1.1 The TSF shall restrict the ability to [selection: change_default, query, modify, delete, clear, [assignment: other operations]] the [assignment: list of TSF data] to [assignment: the defined TSF components].

7.2 FDP_SDA stored data authenticity

7.2.1 Family behavior

This family provides requirements that address protection of user data and dedicated TSF data authenticity while these data are stored within memory areas protected by the TSF.

The TSF provides access to the data in the memory through the specified TOE interfaces only and detect modification of memory information bypassing these TOE interfaces.

The TSF complement the family Stored data integrity (FDP_SDI) which protects the user data and dedicated TSF data from integrity errors while being stored in the memory.

7.2.2 Component leveling

FDP_SDA management of user data and dedicated TSF data allows dedicated TSF components to manage |user data and dedicated TSF data.

7.2.3 Management: FDP_SDA.1

There are no management activities.

7.2.4 Audit: FDP_SDA.1

There are no actions defined to be auditable

7.2.5 FDP_SDA.1 management of TSF data by TSF components

Hierarchical to: No other components.

Dependencies: No other components

FDP_SDA.1.1 The TSF shall ensure the authenticity of the information of user data and dedicated TSF data while it is stored in the [assignment: memory area]

7.3 FDP_SDR stored data replay protection

7.3.1 Family behavior

This family provides requirements that address protection of user data and dedicated TSF data against replay attack while these data are stored within memory areas protected by the TSF.

The TSF provides access to the data in the memory through the specified TOE interfaces only and detect replay of otherwise valid data through bypassing these TOE interfaces.

It complement the family Stored data integrity (FDP_SDI) which protects the user data and dedicated TSF data from integrity errors while being stored in the memory.

7.3.2 Component leveling

FDP_SDR management of user data and dedicated TSF data allows dedicated TSF components to manage user data and dedicated TSF data.

7.3.3 Management: FDP_SDR.1

There are no management activities.

7.3.4 Audit: FDP_SDR.1

There are no actions defined to be auditable

7.3.5 FDP_SDR.1 management of TSF data by TSF components

Hierarchical to: No other components.

Dependencies: No other components

FDP_SDR.1.1 The TSF shall detect replay of the information of user data and dedicated TSF data while it is stored in the [assignment: memory area]

8 Security requirements

8.1 Security functional requirements

8.1.1 Security functional requirements from the body of the protection profile

FRU_FLT.2 – Limited fault tolerance

FRU_FLT.2.1 – The TSF shall ensure the operation of all the TOE's capabilities when the following failures occur: exposure to operating conditions that are not detected according to the requirement Failure with preservation of secure state (FPT_FLS.1).

Refinement: The term *failure* above means *circumstances*. The TOE prevents failures for the circumstances defined above.

FPT_FLS.1 – Failure with preservation of secure state

FPT_FLS.1.1 – The TSF shall preserve a secure state when the following types of failures occur: exposure to operating conditions that might not be tolerated according to the requirement "Limited fault tolerance (FRU_FLT.2)" and where, therefore, a malfunction could occur.

Refinement: The term *failure* above also covers *circumstances*. The TOE prevents failures for the circumstances defined above.

FMT_LIM.1 – Limited capabilities

FMT_LIM.1.1 – The TSF shall be designed and implemented in a manner that limits their capabilities so that in conjunction with "Limited availability (FMT_LIM.2)" the following policy is enforced: Deploying Test Features after TOE Delivery does not allow user data of the Composite TOE to be disclosed or manipulated, TSF data to be disclosed or manipulated, software to be reconstructed, and no substantial information about construction of TSF to be gathered which might enable other attacks.

Application Note: *Composite TOE* refers to the Secure Process hardware component part of the SoC (excluding the remainder of the SoC) with the firmware and software executing on the SP-CPU. Software executing on other parts of the SoC (including software executing in the ARM TrustZone of the SoC master processor) is not considered here. Such software can be viewed similar to the software in entities external to the IC such as the software in a smartcard reader. All such software in other parts of the SoC is considered as non-interfering.

FMT_LIM.2 – Limited availability

FMT_LIM.2.1 – The TSF shall be designed and implemented in a manner that limits their availability so that in conjunction with "Limited capabilities (FMT_LIM.1)" the following policy is enforced: Deploying Test Features after TOE Delivery does not allow user data of the Composite TOE to be disclosed or manipulated, TSF data to be disclosed or manipulated, software to be reconstructed, and no substantial information about construction of TSF to be gathered which might enable other attacks.

FAU_SAS.1 – Audit storage

FAU_SAS.1.1 – The TSF shall provide the test process before TOE Delivery with the capability to store [the Initialization Data, Pre-personalization data] in the SP-QFPROM.

FDP_SDC.1(1) – Stored data confidentiality

FDP_SDC.1.1(1) – The TSF shall ensure the confidentiality of the information of user data *and dedicated TSF data* while it is stored in the memory area within TOE hardware boundary.

Application Note: Different memory areas have different access protection mechanisms. Especially, key material stored in the SP-CMU key tables is confidentiality-protected even from any software executing on the SP-CPU. These keys can be either TSF data or user data and, therefore, the SFR has been refined to also include TSF data. The TSF data to which this applies are keys used by the TSF internally (hardware keys), while managed keys can be user data when they are defined for and used by an application executing on the operating system of the SP-CPU.

FDP_SDI.2(1) – Stored data integrity monitoring and action

FDP_SDI.2.1(1) – The TSF shall monitor user data stored in containers controlled by the TSF for parity errors using parity check/Error Correcting Code and errors after partial power collapse using a checksum function on all objects, based on the following attributes: data stored in SP-RAM, data stored in the SP-CMU key tables.

FDP_SDI.2.2(1) – Upon detection of a data integrity error, the TSF shall perform a cold reset.

Application Note: Another SFR defined further down in this document handles the special case of the OS image stored in RAM.

FPT_PHP.3 – Resistance to physical attack

FPT_PHP.3.1 – The TSF shall resist physical manipulation and physical probing to the TSF by responding automatically such that the SFRs are always enforced.

Refinement from [ICPP]: The TSF will implement appropriate mechanisms to continuously counter physical manipulation and physical probing. Due to the nature of these attacks (especially manipulation), the TSF can by no means detect attacks on all of its elements. Therefore, permanent protection against these attacks is required ensuring that security functional requirements are enforced. Hence, "automatic response" means here (i) assuming that there might be an attack at any time and (ii) countermeasures are provided at any time.

FDP_ITT.1 – Basic internal transfer protection

FDP_ITT.1.1 – The TSF shall enforce the Data Processing Policy to prevent the [disclosure] of user data when it is transmitted between physically separated parts of the TOE.

Refinement: The different memories, the SP-CPU, and other functional units of the TOE (the SP-CMU, SP-ROM, and SP-RAM) are seen as physically separated parts of the TOE.

FPT_ITT.1 – Basic internal TSF data transfer protection

FPT_ITT.1.1 – The TSF shall protect TSF data from [disclosure] when it is transmitted between separate parts of the TOE.

Refinement: The different memories, the SP-CPU, and other functional units of the TOE (the SP-CMU, SP-ROM, and SP-RAM) are seen as physically separated parts of the TOE.

FDP_IFC.1 – Subset information flow control

FDP_IFC.1.1 – The TSF shall enforce the Data Processing Policy on all confidential data when they are processed or transferred by the TOE or by the Security IC Embedded Software.

The following Security Function Policy (SFP) Data Processing Policy is defined for the requirement "Subset information flow control (FDP_IFC.1)":

"User data of the Composite TOE and TSF data shall not be accessible from the TOE except when the Security IC Embedded Software decides to communicate the user data of the Composite TOE via an external interface. The protection shall be applied to confidential data only but without the distinction of attributes controlled by the Security IC Embedded Software."

Application Note: Security IC Embedded Software in the case of the TOE is any software running on the SP-CPU.

Application Note: The component FDP_IFC.1 in the CC has a dependency on FDP_IFF.1, which is not resolved in [ICPP]. The discussion of this omission in [ICPP] is not very convincing. Basically, it requires that the TOE decides which data it considers to be confidential such that it shall not be exported over any of the TOE external interfaces.

FCS_RNG.1 – Random number generation (Class PTG.3)

FCS_RNG.1.1 The TSF shall provide a [hybrid physical] random number generator that implements:

(PTG.3.1) A total failure test detects a total failure of entropy source immediately when the RNG has started. When a total failure is detected, no random numbers will be output.

(PTG.3.2) If a total failure of the entropy source occurs while the RNG is being operated, the RNG [prevents the output of any internal random number that depends on some raw random numbers that have been generated after the total failure of the entropy source].

(PTG.3.3) The online test shall detect non-tolerable statistical defects of the raw random number sequence (i) immediately when the RNG has started, and (ii) while the RNG is being operated. The TSF must not output any random numbers before the power-up online test and seeding of the

DRG.3 post-processing algorithm have finished successfully or when a defect has been detected.

(PTG.3.4) The online test procedure shall be effective to detect non-tolerable weaknesses of the random numbers soon.

(PTG.3.5) The online test procedure checks the quality of the raw random number sequence. It is triggered [continuously]. The online test is suitable for detecting non-tolerable statistical defects of the statistical properties of the raw random numbers within an acceptable period of time.

(PTG.3.6) The algorithmic post-processing algorithm belongs to Class DRG.3 with cryptographic state transition function and cryptographic output function, and the output data rate of the post-processing algorithm shall not exceed its input data rate.

FCS_RNG.1.2 The TSF shall provide [numbers in 32-bit blocks] that meet:

(PTG.3.7) Statistical test suites cannot practically distinguish the internal random numbers from output sequences of an ideal RNG. The internal random numbers must pass test procedure A [and no other test suite].

(PTG.3.8) The internal random numbers shall [use PTRNG of class PTG.2 as random source for the post-processing].

8.1.2 Security functional requirements from augmentation packages

The following section contains security functional requirements from packages defined in [ICPP]:

- Cryptographic services package “TDES”
- Cryptographic services package “AES”
- Cryptographic services package “Hash functions”

8.1.2.1 Cryptographic services package – TDES

This package is included to address the provision of Triple-DES encryption/decryption.

The package’s organizational security policy and security objectives were added to the respective lists in chapters 5 and 6. This package adds the following SFRs.

8.1.2.1.1 Security functional requirements

FCS_COP.1/TDES – Cryptographic operation – TDES

FCS_COP.1.1/TDES – The TSF shall perform encryption and decryption in accordance with a specified cryptographic algorithm TDES in ECB mode, CBC mode and cryptographic key size 168-bit that meet the following Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher (NIST SP 800-67) and Recommendation for Block Cipher Modes of Operation, Methods and Techniques (NIST SP 800-38A).

FCS_CKM.4/TDES Cryptographic key destruction – TDES

FCS_CKM.4.1/TDES – The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method overwriting with random values that meets the following: none.

8.1.2.2 Cryptographic services package – AES

This package is included to address the provision of Advanced Encryption Standard encryption/decryption.

The package's organizational security policy and security objectives were added to the respective lists in chapters 5 and 6. This package adds the following SFRs.

8.1.2.2.1 Security functional requirements

FCS_COP.1/AES– Cryptographic operation – AES

FCS_COP.1.1/AES – The TSF shall perform encryption and decryption *and authentication when using CCM mode* in accordance with a specified cryptographic algorithm AES in ECB mode, CBC mode, CTR mode, CCM mode and cryptographic key sizes 128-, 256-bit that meet the following: Specification for the ADVANCED ENCRYPTION STANDARD (AES) (FIPS PUB 197), Recommendation for Block Cipher Modes of Operation, Methods and Techniques (NIST SP 800-38A), Recommendation for Block Cipher Modes of Operation: The CCM Mode For Authentication and Confidentiality (NIST SP 800-38C).

FCS_CKM.4/AES – Cryptographic key destruction – AES

FCS_CKM.4.1/AES – The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method overwriting with zeroes or overwriting the protecting key encryption key in the key hierarchy with zeroes that meets the following: none.

8.1.2.3 Cryptographic services package – Hash functions

This package is included to address the provision of secure hash functions.

8.1.2.3.1 Security functional requirements

FCS_COP.1/SHA– Cryptographic operation – SHA

FCS_COP.1.1/SHA – The TSF shall perform hashing in accordance with a specified cryptographic algorithm [SHA-1, SHA-256, SHA-384, SHA-512] and cryptographic key sizes none that meet the following: Secure Hash Standard (SHS) (FIPS PUB 180-4).

8.1.3 Security functional requirements beyond those in [ICPP]

The TOE implements a set of additional security functions that are beyond what is defined in [ICPP]. This includes functions provided by the hardware as well as functions provided by the operating system of the SP-CPU, which is part of the TOE and its TSF.

The SFRs for those functions are grouped for dedicated additional functions, which are described in general at the beginning of each group before stating the SFRs for those functions in CC terminology.

8.1.3.1 Group 1: Cryptographic functions

This group describes the cryptographic functions that are beyond those defined in [ICPP]. This includes the asymmetric algorithms.

FCS_CKM.1/SYM – Cryptographic key generation – Symmetric Keys

FCS_CKM.1.1/SYM – The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm capable of generating a random bit sequence and specified cryptographic key sizes :

- AES: 128 bits, 256 bits
- CMAC AES: 128 bits, 256 bits
- HMAC SHA-1: 256 bits
- HMAC SHA-256: 256 bits
- HMAC S HA-384: 256 bits
- HMAC SHA-512: 256 bits

that meet the following: NIST SP800-133 chapter 6 with V being a string of zeroes.

FCS_CKM.1/KDF – Cryptographic key generation – Key Derivation Function

FCS_CKM.1.1/KDF – The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm Key Derivation Function in Counter Mode using HMAC SHA-256 and specified cryptographic key sizes:

- AES: 128 bits, 256 bits
- CMAC AES: 128 bits, 256 bits
- HMAC SHA-1: 256 bits
- HMAC SHA-256: 256 bits
- HMAC SHA-384: 512 bits
- HMAC SHA-512: 512 bits

that meet the following: NIST SP 800-108 Section 5.1, FIPS 198-1, FIPS 180-4.

FCS_CKM.4/HMAC/CMAC – Cryptographic key destruction

FCS_CKM.4.1/HMAC/CMAC – The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method overwriting with random values for keys of size above 32 bytes or overwriting the protecting key encryption key in the key hierarchy with zeroes for all other keys that meets the following: none.

FCS_COP.1/CMAC – Cryptographic operation

FCS_COP.1.1/CMAC – The TSF shall perform message authentication code generation in accordance with a specified cryptographic algorithm CMAC using AES and cryptographic key sizes 128 bits, 256 bits that meet the following: Specification for the ADVANCED ENCRYPTION STANDARD (AES) (FIPS PUB 197), and Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication (NIST SP 800-38B).

FCS_COP.1/HMAC – Cryptographic operation

FCS_COP.1.1/HMAC – The TSF shall perform message authentication code generation in accordance with a specified cryptographic algorithm HMAC using SHA-1 or SHA-256, SHA-384, SHA-512 and cryptographic key sizes 256 bits for SHA-1 and SHA-256, 512 bits for SHA-384 and SHA-512 that meet the following: The Keyed-Hash Message Authentication Code (HMAC) (FIPS PUB 198-1) and Secure Hash Standard (SHS) (FIPS PUB 180-4).

FCS_CKM.1/ECC – Cryptographic key generation – ECC

FCS_CKM.1.1/ECC – The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm ECDH/ECDSA key generation with ECDH/ECDSA key lengths corresponding to the ECDH/ECDSA parameters for NIST P-192, P-224, P-256, P-384 and P-521 and specified cryptographic key sizes implied by the curve that meet the following: NIST SP 800-56A revision 2 section 5.6.1.2 supported by FIPS PUB 186-4 Appendix D.1.2 and RFC5639.

FCS_CKM.4/RSA/ECC – Cryptographic key destruction

FCS_CKM.4.1/RSA/ECC – The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method overwriting with zeroes that meets the following: none.

FCS_COP.1/RSA_SIGN – Cryptographic operation

FCS_COP.1.1/RSA_SIGN – The TSF shall perform signature generation and verification in accordance with a specified cryptographic algorithm RSA and cryptographic key sizes 1024 bits, 2048 bits (RSA) that meet the following: FIPS PUB 186-4 Section 5, PKCS#1 v2.1 PSS, PKCS#1 v1.5.

Application note: It is mandatory to provide both the private and public keys for the RSA signature generation and use the certified APIs in order to protect against the DFA and the side channel attacks. Please see Qualcomm SPU Core Hana API Rev 4.6 for the further detail.

FCS_COP.1/RSA_ENC – Cryptographic operation

FCS_COP.1.1/RSA_ENC – The TSF shall perform encryption and decryption in accordance with a specified cryptographic algorithm RSA and cryptographic key sizes 1024 bits, 2048 bits (RSA) that meet the following: FIPS PUB 186-4 Section 5, PKCS#1 v2.1 OAEP, PKCS#1 v1.5.

Application note: It is mandatory to provide both the private and public keys for the RSA decryption and use the certified APIs in order to protect against the DFA and the side channel attacks. Please see Qualcomm SPU Core Hana API Rev 4.6 for the further detail.

FCS_COP.1/ECDSA – Cryptographic operation

FCS_COP.1.1/ECDSA – The TSF shall perform signature generation and verification in accordance with a specified cryptographic algorithm ECDSA (NIST P-192, P-224, P-256, P-384 and P-521 curves) and cryptographic key sizes compliant to NIST curves: 192 bits, 224 bits, 256 bits, 384 bits, 521 bits that meet the following: FIPS PUB 186-4 Section 6 and Appendix D.1.2, and RFC5639.

FCS_COP.1/ECDH – Cryptographic operation

FCS_COP.1.1/ECDH – The TSF shall perform shared secret generation in accordance with a specified cryptographic algorithm ECDH (NIST P-192, P-224, P-256, P-384 and P-521 curves) and cryptographic key sizes 192 bits, 224 bits, 256 bits, 384 bits, 521 bits that meet the following: NIST SP800-56A revision 2 section 5.7.1.2 supported by FIPS 186-4 Appendix D.1.2, and RFC5639.

Application note: The static Diffie-Hellman, where a fixed start secret is used, is prohibited – only the ephemeral scheme is allowed. Please see Qualcomm SPU Core Hana API Rev 4.6 for the further detail.

8.1.3.2 Group 2: Secure boot

This group describes the functions for the secure boot and startup process of the TOE. This includes the verification of the authenticity and integrity of the boot code and application executables, and the decryption of that data for a cold boot and the integrity verification of that data in SP-RAM in case of a warm boot. The case for the cold boot is defined using SFR FDP_ITC.1 and the case of the warm boot is defined using SFR FDP_SDI.2(2).

FDP_ITC.1 – Import of user data without security attributes

FDP_ITC.1.1 – The TSF shall enforce the no access control policy when importing *the boot image and application executables as* user data, controlled under the SFP, from outside of the TOE.

FDP_ITC.1.2 – The TSF shall ignore any security attributes associated with user data when imported from outside of the TOE.

FDP_ITC.1.3 – The TSF shall enforce the following rules when importing *the boot image and application executables as* user data controlled under the SFP from outside the TOE

- The TSF shall verify the digital signature of the boot image and application executables.
- The TSF shall verify the version number of the boot image and application executables to be equal or larger than the version number known to the TOE to prevent a rollback.
- The TSF shall stop execution when any of the aforementioned validation steps fail.
- When the aforementioned validation steps are successful, the TSF shall decrypt the boot image and application executables.

FDP_SDI.2(2) – Stored data integrity monitoring and action (Refinement)

FDP_SDI.2.1(2) – The TSF shall monitor *OS image and application executables and data stored in SP-RAM* for any modification during partial power collapse on all objects, based on the following attributes: 64-bits SUM complemented value of the SP-RAM content.

FDP_SDI.2.2(2) – Upon detection of a data integrity error, the TSF shall start a cold boot process, which erase the SP-RAM.

Refinement: The term “user data stored in containers controlled by the TSF” is refined by the specific user data “OS image and application executables and data stored in SP-RAM”.

8.1.3.3 Group 3: Access control policies for hardware components

The TOE implements a set of policies that control access to critical hardware components such as OTP items, access control of applications to memory areas, and access control to keys. Access control to OTP items includes access by the TSF itself. Access control to keys is also a TSF internal access control mechanism where access to keys from SP-CPU is controlled, in general, by the SP-CMU.

Those access control policies are related to TSF data rather than user data and are, therefore, expressed by extended SFRs for the management of TSF data.

FMT_CMT.1(1) – Management of TSF data by TSF components

FMT_CMT.1.1(1) The TSF shall be implemented to restrict the ability to [modify and define] the memory areas shared with other components of the SoC to SP-ExtMM controlled by the SP-CPU.

FMT_CMT.1(2) – Management of TSF data by TSF components

FMT_CMT.1.1(2) The TSF shall be implemented to restrict the ability to [modify and access] the key table to SP-CMU.

FMT_CMT.1(3) – Management of TSF data by TSF components

FMT_CMT.1.1(3) The TSF shall be implemented to restrict the ability to [use] the hardware support for cryptographic operations and the random number generator to SP-CMU.

FMT_CMT.1(4) – Management of TSF data by TSF components

FMT_CMT.1.1(4) The TSF shall be implemented to restrict the ability to [program] the SP-ExtMM and SP-MMU to SP-CPU when in privileged mode.

FMT_CMT.1(5) – Management of TSF data by TSF components

FMT_CMT.1.1(5) The TSF shall be implemented to restrict the ability to [access] the areas in the SP-ROM to SP-CPU based on the restrictions implemented by the SP-ROM permission checker.

FMT_CMT.1(6) – Management of TSF data by TSF components

FMT_CMT.1.1(6) The TSF shall be implemented to restrict the ability to [patch] the SP-ROM memory to the OTP patch logic and the CSR patch mechanism.

8.1.3.4 Group 4: Access control policies for software-managed data

The TOE implements a set of policies that control access to user and TSF data managed by the operating system. The operating system access control SFP relates to all processes managed by the TOE and relies on data encryption on a per process basis. Each process uses its own key and therefore prevent access to its data by other processes. A process can be the TOE operating system or an application running in the TOE.

FDP_ACC.2 Complete access control

FDP_ACC.2.1 – The TSF shall enforce the operating system access control SFP on

- Subjects: operating system, applications, and
- Objects: all non-volatile data storage objects holding application data, application code, TSF data and TSF code

and all operations among subjects and objects covered by the SFP.

FDP_ACC.2.2 – The TSF shall ensure that all operations between any subject controlled by the TSF and any object controlled by the TSF are covered by an access control SFP.

FDP_ACF.1 Security attribute based access control

FDP_ACF.1.1 – The TSF shall enforce the operating system access control SFP to objects based on the following:

- Subjects: all subjects are associated with dedicated encryption keys,
- Objects: non-volatile data storage objects encrypted with one of these keys and associated anti-rollback information.

FDP_ACF.1.2 – The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: non-volatile storage data will be encrypted with the subject-specific key during write operations and will be decrypted with that key during read operations, and will be marked with the anti-rollback information to prevent data rollback.

FDP_ACF.1.3 – The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: none.

FDP_ACF.1.4 – The TSF shall explicitly deny access of subjects to objects based on the following additional rules: none.

FMT_MSA.3 - Static attribute initialization

FMT_MSA.3.1– The TSF shall enforce the operating system access control SFP to provide [restrictive] default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2– The TSF shall allow ~~the~~ nobody to specify alternative initial values to override the default values when an object or information is created.

FDP_RIP.1/Keys - Subset residual information protection

FDP_RIP.1.1/Keys – The TSF shall ensure that any previous information content of a resource is made unavailable upon the [deallocation of the resource from] the following objects: the interface buffers to the cryptographic services.

FDP_RIP.1/Transient - Subset residual information protection

FDP_RIP.1.1/Transient – The TSF shall ensure that any previous information content of a resource is made unavailable upon the [deallocation of the resource from] the following objects: any transient object.

8.1.3.5 Group 5: Protection of data stored outside the TOE

The TOE implements a set of security mechanisms to protect user data and TSF dedicated data stored outside the TOE in memories shared with the host SoC.

The provided protection include integrity, confidentiality, authenticity and anti-rollback to protect against eavesdropping, modification and replay attacks.

FDP_SDI.2(3) – Stored Data Integrity

FDP_SDI.2.1(3) – The TSF shall monitor user data stored in containers controlled by the TSF for any integrity errors on all objects, based on the following attributes: SHA-256 hash tree and CMAC-AES-256 for persistent data store and AES-CCM-256 for volatile data store *while data is stored in the memory area outside the TOE hardware boundary*.

FDP_SDI.2.2(3) – Upon detection of a data integrity error, the TSF shall prevent the data from being used by the TOE.

FDP_SDC.1(2) – Stored Data Confidentiality

FDP_SDC.1.1(2) – The TSF shall ensure the confidentiality of the information of user data *and dedicated TSF data* while it is stored in the memory area outside the TOE hardware boundary.

Refinement: “user data” shall also comprise “dedicated TSF data”.

Application Note: The TOE encrypt (AES-CBC-256) persistent user data before being exported outside the TOE (NVM storage use case). Confidentiality is ensured by the fact that the TOE generates a cryptographic key using a random number to encrypt the user data. This key is not known outside of the TOE nor is it exportable.

Application Note: The TOE encrypt (AES-CCM-256) transient user data before being exported outside the TOE (swap use case). Confidentiality is ensured by the fact that the TOE generates a cryptographic key using a random number to encrypt the user data. This key is not known outside of the TOE nor is it exportable.

FDP_SDA.1 – Stored Data Authenticity

FDP_SDA.1.1 The TSF shall ensure the authenticity of the information of *its own* user data and dedicated TSF data while it is stored in the memory area outside the TOE hardware boundary.

Application Note: The TOE is capable of identifying whether user data and dedicated TSF data originate from the TOE or not. Only such data is loaded back into the TOE for processing. This is ensured by the fact that the TOE generates a cryptographic key using a random number to authenticate the user data. This key is not known outside of the TOE nor is it exportable.

Application Note: For persistent data, the key is persistent across reset. For transient data, the key is volatile (stored in SP-RAM).

FDP_SDR.1 – Stored Data Replay Protection

FDP_SDR.1.1 The TSF shall detect replay of the information of user data and dedicated TSF data while it is stored in the <u>memory area outside the TOE hardware boundary</u>.

Application Note: The TOE is capable of identifying whether user data is replay or not (current version of user data is replaced by an older version with valid encryption / authentication code). This is ensured by the fact that the TOE manage and store internally a monotonic counter that is used in the computation of the authentication code as defined in FDP_SDI.2(3).

Application Note: For persistent data, the counter is managed by the Anti-Replay Island (ARI) and is persistent across reset. For transient data, the counter is volatile (stored in SP-RAM).

8.2 Security assurance requirements

The Security Assurance Requirements are as defined in [ICPP], including the refinements defined there.

8.3 Security requirements rationale

Section 6.3 of [ICPP] defines the security requirements rationale for the objectives and SFRs defined in [ICPP], and Section 7 of [ICPP] defines the additional security requirements rationale for the optional packages. This rationale applies for all security objectives and SFRs taken from [ICPP]. This section, therefore, addresses only the rationale for the additional security objectives and SFRs defined in this Security Target.

This Security Target adds the following additional security objectives:

- O.Defense-in-Depth
- O.Secure-Boot
- O.RSA_ENC
- O.RSA_SIGN
- O.HMAC
- O.CMAC
- O.ECDSA
- O.ECDH
- O.KDF
- O.OSData.Access
- O.OSData.Protect
- O.Reallocation

Those security objectives are addressed by the following additional SFRs. In addition, the following table includes the objectives rationale for the optional packages defined in [ICPP].

Table 8-1 Security requirement vs objectives mapping

Objective	Description
O.AES	The security objective of the TOE to provide secure hardware based cryptographic services for the AES for encryption and decryption is addressed by the SFRs of FCS_COP.1/AES and FCS_CKM.4/AES defining the cryptographic operation of the cipher and the associated functionality of key destruction.
O.TDES	The security objective of the TOE to provide cryptographic services for the TDES for encryption and decryption is addressed by the SFRs of FCS_COP.1/TDES and FCS_CKM.4/TDES defining the cryptographic operation of the cipher and the associated functionality of key destruction.
O.HMAC	The security objective of the TOE to provide secure hardware based cryptographic services for the HMAC is addressed by the SFRs of FCS_COP.1/HMAC and FCS_CKM.4/HMAC/CMAC and FCS_CKM.1/SYM defining the cryptographic operation of the cipher used for Message Authentication Code generation and the associated functionality of key destruction.[2]
O.CMAC	The security objective of the TOE to provide secure hardware based cryptographic services for the CMAC-AES is addressed by the SFRs of FCS_COP.1/CMAC and FCS_CKM.4/HMAC/CMAC and FCS_CKM.1/SYM defining the cryptographic operation of the cipher used for Message Authentication Code generation and the associated functionality of key destruction.
O.SHA	The security objective of the TOE to provide secure hardware-based cryptographic services for secure hash calculation is addressed by the SFR of FCS_COP.1/SHA defining the cryptographic operation of the cipher.
O.Defense-in-Depth	The security objective of the TOE to ensure that critical functions and TSF data cannot be accessed by a simple breach of security of the software executing on the SP-CPU is addressed by the SFRs FMT_CMT.1(1) to FMT_CMT.1(6) which define various security functions offered by the SP-CPU that can and must be utilized by the TOE.
O.Secure-Boot	The security objective of the TOE to ensure that only authorized software is loaded during the boot process after the integrity and authenticity of that software has been verified is addressed by the SFRs FDP_ITC.1 and FDP_SDI.2(2). Both define the integrity protection mechanism of the software imported from outside of the TOE.
O.RSA_SIGN	The security objective of the TOE to provide secure cryptographic services implementing the RSA algorithm for signature generation and verification is addressed by the SFRs FCS_CKM.4/RSA/ECC and FCS_COP.1/RSA_SIGN which collectively define the key destruction and cryptographic operation of RSA.
O.RSA_ENC	The security objective of the TOE to provide secure cryptographic services implementing the RSA algorithm for encryption//decryption is addressed by the SFRs FCS_CKM.4/ RSA/ECDSA/ECDH, and FCS_COP.1/RSA_ENC which collectively define the key destruction and cryptographic operation of RSA.

[2] HMAC keys larger than 32-byte are not protected by the CMU (key table mechanism).

Objective	Description
O.ECDSA	The security objective of the TOE to provide secure cryptographic services implementing the ECDSA algorithm based on the NIST curves P-192, P-224, P-256, P-384, P-521 for signature generation and verification is addressed by the SFRs FCS_CKM.1/ECC, FCS_CKM.4/RSA/ECC, and FCS_COP.1/ECDSA which collectively define the key generation, destruction and cryptographic operation of ECDSA.
O.ECDH	The security objective of the TOE to provide secure cryptographic services implementing the ECDH algorithm based on the NIST curves P-192, P-224, P-256, P-384, P-521 for key generation and shared secret generation is addressed by the SFRs FCS_CKM.1/ECC, FCS_CKM.4/RSA/ECC, and FCS_COP.1/ECDH which collectively define the key generation, destruction and cryptographic operation of ECDH.
O.KDF	The security objective of the TOE to provide secure cryptographic services implementing the Key Derivation Function algorithm based on NIST SP 800-108 is addressed by the SFR FCS_CKM.1/KDF specifying such key derivation function.
O.OSData.Access	The security objective of the TSF to restrict access of software applications exclusively to its own data stored in volatile and non-volatile memory is addressed by the SFRs of FDP_ACC.2/OS, FDP_ACF.1/OS, FMT_MSA.3/OS which collectively define such access control mechanism.
O.OSData.Protect	The security objective of the TSF to protect TSF code, TSF data, application code and application data using cryptographic functions with a cryptographic strength commensurate with the value of the data against loss of confidentiality, integrity, authenticity and against rollback is addressed by the SFRs of FDP_ACC.2/OS, FDP_ACF.1/OS, FMT_MSA.3/OS, FDP_SDI.2(3), FDP_SDI.2(2), FDP_SDC.1(2), FDP_SDA.1, FDP_SDR.1 which collectively define such protection control mechanism. The hardware managed symmetric keys maintained by the TOE are protected as specified in FMT_CMT.1(2).
O.Reallocation	The security objective of the TOE to ensure that the re-allocation of a memory block for the runtime areas maintained by the TOE operating system does not disclose any information that was previously stored is addressed by the SFRs of FDP_RIP.1/Keys, FDP_RIP.1/Transient which define a clearing of residual information from storage locations before reassignment.

The following table provides the SFR dependency analysis for the SFRs covering the aforementioned objectives.

Table 8-2 Security requirement dependencies

SFR	Dependencies	Fulfillment
FCS_CKM.1/SYM	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4	FCS_COP.1/AES, FCS_COP.1/CMAC, , FCS_COP.1/HMAC FCS_CKM.4/AES, FCS_CKM.4/ HMAC/CMAC
FCS_CKM.1/KDF	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4	FCS_COP.1/AES, FCS_COP.1/CMAC, , FCS_COP.1/HMAC FCS_CKM.4/AES, FCS_CKM.4/ HMAC/CMAC
FCS_CKM.1/ECC	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4	FCS_COP.1/ECDH FCS_COP.1/ECDSA FCS_CKM.4/RSA/ECC

SFR	Dependencies	Fulfillment
FCS_CKM.4/AES	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1]	FCS_CKM.1/SYM
FCS_CKM.4/TDES	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1]	FCS_CKM.1/TDES is not fulfilled. Indeed TDES implementation does not leverage the CMU HW to protect the TDES key, so TDES key management is let to the caller.
FCS_CKM.4/HMAC/CMAC	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1]	FCS_CKM.1/SYM, FCS_CKM.1/KDF
FCS_CKM.4/RSA/ECC	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1]	FCS_CKM.1/ECC
FCS_COP.1/AES	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/SYM, FCS_CKM.1/KDF FCS_CKM.4/AES
FCS_COP.1/TDES	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/TDES is not fulfilled. Indeed TDES implementation does not leverage the CMU HW to protect the TDES key, so TDES key management is let to the caller. FCS_CKM.4/TDES
FCS_COP.1/SHA	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	The hash generation does not require any key which implies that the dependency on key generation or key import is not applicable and thus unmet. The hash generation does not use cryptographically sensitive parameters which implies that no such parameters must be destroyed. Thus, the dependency on key destruction is unmet.
FCS_COP.1/RSA_SIGN	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/RSA is not fulfilled. Instead, the caller must provide the RSA key for the RSA operation FCS_CKM.4/RSA/ECC
FCS_COP.1/RSA_ENC	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/RSA is not fulfilled. Instead, the caller must provide the RSA key for the RSA operation FCS_CKM.4/RSA/ECC
FCS_COP.1/ECDSA	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/ECC is not fulfilled. Instead, the caller must provide the ECC key for the ECDSA operation FCS_CKM.4/RSA/ECC
FCS_COP.1/ECDH	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/ECC is not fulfilled. Instead, the caller must provide the ECC key for the ECDH operation FCS_CKM.4/RSA/ECC
FCS_COP.1/HMAC	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/SYM, FCS_CKM.1/KDF FCS_CKM.4/HMAC/CMAC

SFR	Dependencies	Fulfillment
FCS_COP.1/CMAC	[FCS_ITC.1 or FCS_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1/SYM, FCS_CKM.1/KDF FCS_CKM.4/HMAC/CMAC
FDP_ITC.1	[FDP_ACC.1 or FDP_IFC.1] FMT_MSA.3	The data forming the TOE software and that is imported by the TOE is encrypted and signed. No access control policy or information flow control is required because the associated keys are stored inside the TOE. By relying on the cryptographic protection the TOE ensures the application of the rules defined FDP_ITC.1.
FDP_SDI.2(2)	No dependencies	
FMT_CMT.1(1)	No dependencies	
FMT_CMT.1(2)	No dependencies	
FMT_CMT.1(3)	No dependencies	
FMT_CMT.1(4)	No dependencies	
FMT_CMT.1(5)	No dependencies	
FMT_CMT.1(6)	No dependencies	
FDP_ACC.2/OS	FDP_ACF.1	FDP_ACF.1/OS
FDP_ACF.1/OS	FDP_ACC.1 FMT_MSA.3	FDP_ACC.2/OS FMT_MSA.3/OS
FMT_MSA.3/OS	FMT_MSA.1 FMT_SMR.1	The dependency of FMT_MSA.3/OS to FMT_MSA.1 and FMT_SMR.1 is unmet, because the default value cannot be altered or managed.
FDP_RIP.1/Keys	No dependencies	
FDP_RIP.1/Transient	No dependencies	
FDP_SDI.2(3)	No dependencies	
FDP_SDC.1(2)	No dependencies	
FDP_SDA.1	No dependencies	
FDP_SDR.1	No dependencies	

NOTE: The security objectives of O.Leak-Inherent, O.Phys-Probing, O.Malfunction, O.Phys-Manipulation, and O.Leak-Forced defined in [ICPP] support the secure implementation of all crypto services introduced by P.Crypto-Service. The TOE will ensure the confidentiality of the user data and TSF data for these crypto services.

9 TOE summary specification

9.1 TOE summary specification rationale

Table 9-1 maps the SFRs to the sections of the TSS containing the descriptions of how those SFRs are implemented.

Table 9-1 TOE summary specification rationale

SFR	9.1.1.1 Random number generation	9.1.1.2 AES coprocessor	9.1.1.3 Hashing	9.1.1.4 Authentication	9.1.1.5 Key Derivation Function	9.1.1.6 Key protection	9.1.1.7 TDES	9.1.1.8 Asymmetric Crypto	9.1.2.1 Secure boot	9.1.2.2 Secure software update	9.1.3 Application manager	9.1.4 Domain separation between applications executed by the TOE	9.1.5 Physical protection	9.1.6.1 Control memory areas shared with other components of the SoC	9.1.6.2 Control access to keys and the key table	9.1.6.3 Control use of hardware support for cryptographic operations and random number generation	9.1.6.4 Control access to the SP-RAM by software on the SP-CPU	9.1.7.1 Cryptographic protection of persistent data stored outside the TOE	9.1.7.2 Cryptographic protection of transient data and code stored outside the TOE	9.1.7.3 Reallocation of shared resources	9.1.8.1 Logical protection, SP-ROM	9.1.8.2 Logical protection, SP-RAM	9.1.9 Production data and OTP handling	9.1.10 Life Cycle Control
FRU_FLT.2													X											
FPT_FLS.1													X											
FMT_LIM.1																								X
FMT_LIM.2																								X
FAU_SAS.1																							X	
FDP_SDC.1(1)																					X			
FDP_SDI.2 (1)																					X			

SFR	9.1.1.1 Random number generation	9.1.1.2 AES coprocessor	9.1.1.3 Hashing	9.1.1.4 Authentication	9.1.1.5 Key Derivation Function	9.1.1.6 Key protection	9.1.1.7 TDES	9.1.1.8 Asymmetric Crypto	9.1.2.1 Secure boot	9.1.2.2 Secure software update	9.1.3 Application manager	9.1.4 Domain separation between applications executed by the TOE	9.1.5 Physical protection	9.1.6.1 Control memory areas shared with other components of the SoC	9.1.6.2 Control access to keys and the key table	9.1.6.3 Control use of hardware support for cryptographic operations and random number generation	9.1.6.4 Control access to the SP-RAM by software on the SP-CPU	9.1.7.1 Cryptographic protection of persistent data stored outside the TOE	9.1.7.2 Cryptographic protection of transient data and code stored outside the TOE	9.1.7.3 Reallocation of shared resources	9.1.8.1 Logical protection, SP-ROM	9.1.8.2 Logical protection, SP-RAM	9.1.9 Production data and OTP handling	9.1.10 Life Cycle Control
FPT_PHP.3													X											
FDP_ITT.1													X											
FPT_ITT.1													X											
FDP_IFC.1													X											
FCS_RNG.1	X																							
FCS_COP.1/AES		X																						
FCS_CKM.4/AES						X																		
FCS_COP.1/TDES							X																	
FCS_CKM.4/TDES						X																		
FCS_COP.1/SHA			X																					
FCS_CKM.1/SYM	X																							
FCS_CKM.1/KDF					X																			
FCS_COP.1/HMAC/CMAC				X																				
FCS_CKM.4/HMAC/CMAC						X																		
FCS_COP.1/RSA_SIGN								X																
FCS_COP.1/RSA_ENC								X																
FCS_COP.1/ECDSA								X																
FCS_COP.1/ECDH								X																
FCS_CKM.1/ECC								X																

SFR	9.1.1.1 Random number generation	9.1.1.2 AES coprocessor	9.1.1.3 Hashing	9.1.1.4 Authentication	9.1.1.5 Key Derivation Function	9.1.1.6 Key protection	9.1.1.7 TDES	9.1.1.8 Asymmetric Crypto	9.1.2.1 Secure boot	9.1.2.2 Secure software update	9.1.3 Application manager	9.1.4 Domain separation between applications executed by the TOE	9.1.5 Physical protection	9.1.6.1 Control memory areas shared with other components of the SoC	9.1.6.2 Control access to keys and the key table	9.1.6.3 Control use of hardware support for cryptographic operations and random number generation	9.1.6.4 Control access to the SP-RAM by software on the SP-CPU	9.1.7.1 Cryptographic protection of persistent data stored outside the TOE	9.1.7.2 Cryptographic protection of transient data and code stored outside the TOE	9.1.7.3 Reallocation of shared resources	9.1.8.1 Logical protection, SP-ROM	9.1.8.2 Logical protection, SP-RAM	9.1.9 Production data and OTP handling	9.1.10 Life Cycle Control
FCS_CKM.4/RSA/ECC						X																		
FDP_ITC.1									X	X	X													
FDP_SDI.2(2)																								
FMT_CMT.1(1)														X										
FMT_CMT.1(2)															X									
FMT_CMT.1(3)																X								
FMT_CMT.1(4)												X					X							
FMT_CMT.1(5)																					X			
FMT_CMT.1(6)																					X			
FDP_ACC.2											X													
FDP_ACF.1											X													
FMT_MSA.3											X													
FDP_RIP.1/Keys						X																		
FDP_RIP.1/Transient																				X				
FDP_SDI.2(3)																		X	X					
FDP_SDC.1(2)																		X	X					
FDP_SDA.1																		X	X					
FDP_SDR.1																		X	X					

9.1.1 Cryptographic services and random number generation

9.1.1.1 Random number generation

The implemented physical random number generator together with the associated post processing provide the functionality defined by FCS.RNG.1 and FCS_CKM.1/SYM.

9.1.1.2 AES coprocessor

A first AES coprocessor implemented as part of the SP-CMU provides AES encryption and decryption with the various crypto modes as required by FCS_COP.1/AES.

9.1.1.3 Hashing

A coprocessor implemented as part of the SP-CMU supports the different hashing algorithms as required by FCS_COP.1/SHA. The coprocessor needs to be configured with the required hashing algorithm before the operation is started.

9.1.1.4 Authentication

A second AES coprocessor implemented as part of the SP-CMU provides AES authentication functionality as required by FCS_COP.1/CMAC.

A SHA co-processor supports an HMAC implementation as required by FCS_COP.1/HMAC

9.1.1.5 Key derivation function

The key table implemented in the SP-CMU provides the key derivation functionality as required by FCS_CKM.1/KDF.

9.1.1.6 Key protection

The key table evaluates the key properties and ensures that keys are only used for the intended usage. In addition, the implementation of the key table limits access to the keys by SP-CPU. Keys released by the software cannot be further used. Thereby, the key table implements the protection of keys as required by FCS_CKM.4/AES, FCS_CKM.4/HMAC/CMAC and FDP_RIP.1/Keys

For RSA/ECDH/ECDSA, once the key is released by the asymmetric crypto function, it is destructed as required by FCS_CKM.4/RSA/ECC.

For TDES, once the key is released by the TDES crypto function, it is destructed as required by FCS_CKM.4/TDES.

9.1.1.7 TDES

Software implementation providing TDES encryption and decryption with the various crypto modes as required by FCS_COP.1/TDES.

9.1.1.8 Asymmetric crypto

RSA and ECC implemented with support of the SP-CMU(SP-PKA) which provide signing/verification and encryption/decryption functionality as required by

FCS_COP.1/RSA_SIGN[3], FCS_COP.1/RSA_ENC[4], FCS_COP.1/ECDSA, FCS_COP.1/ECDH[5], FCS_CKM.1/ECC.

9.1.2 Secure boot and secure update

9.1.2.1 Secure boot

The boot image is stored outside the TOE and loaded during startup by the boot loader stored in the SP-ROM as part of the TOE. The boot loader performs the following cold boot sequence:

1. SP-CPU initializes all memory areas (programming of the SP-MMU of the SP-CPU).
2. SP-CPU copies the MCP image from the shared SoC RAM to the SP-RAM (not accessible to the rest of the SoC).
3. SP-CPU verifies the digital signature (RSA2048-PSS) of the (encrypted) MCP image stored in the SP-RAM. If this verification fails, the execution stops.
4. SP-CPU decrypts the MCP image in place.
5. SP-CPU passes control to the MCP image (jump to Main Control Program).

The boot image does not maintain specific properties or access control during the storage outside the TOE. The protection relies on the digital signature, the encryption and the protected version number as required by FDP_ITC.1.

9.1.2.2 Secure software update

In case the boot loader detects an updated boot image during the verification process, the new software version is verified in the same way. In addition, the version number maintained in the SP-QFPROM is verified and updated accordingly in case a greater version number is identified. Thereby the update functionality implements the security mechanisms required by FDP_ITC.1.

9.1.3 Application manager

The application manager uses the same security mechanisms required by FDP_ICT.1. This comprises the integrity and authenticity verification based on a valid signature, the decryption of the application image and the rollback protection for version control. User and TSF data belonging to a dedicated application is encrypted on process basis with different keys to enforce the access control policy required by FDP_ACC.2, FDP_ACF.1 and FMT_MSA.3. Thereby the user data and TSF data of different applications is only accessible by the associated application.

[3] It is mandatory to provide both the private and public keys for the RSA signature generation and use the certified APIs in order to protect against the DFA and the side channel attacks. Please see Qualcomm SPU Core Hana API Rev 4.6 for the further detail.

[4] It is mandatory to provide both the private and public keys for the RSA decryption and use the certified APIs in order to protect against the DFA and the side channel attacks. Please see Qualcomm SPU Core Hana API Rev 4.6 for the further detail.

[5] The static Diffie-Hellman, where a fixed start secret is used, is prohibited – only the ephemeral scheme is allowed. Please see Qualcomm SPU Core Hana API Rev 4.6 for the further detail.

9.1.4 Domain separation between applications executed by the TOE

The TOE implements a dedicated control for the access to external memory as well as for the access to internal memories and resources that can be configured in privileged mode. This separation is required by FMT_CMT.1(4).

9.1.5 Physical protection

The TOE provides protection mechanisms against non-invasive, semi-invasive, and invasive physical attacks. These countermeasures protect against side-channel attacks, fault injection attacks, and against the various physical attacks.

The TOE implements countermeasures against side-channel attacks including constant execution time, masking (e.g., by the AES coprocessors) and random polarity switching for the SP-CPU bus as required by FDP_ITT.1, FPT_ITT.1 and FDP_IFC.1.1.

The TOE implements countermeasures against fault injection attacks comprising redundant logic and error detection/correction code for the following components SP-CPU, SP-QFPROM, SP-ROM, SP-RAM, SP-KT and TOE internal buses as required by FPT_FLS.1.

Further on, the TOE includes sensors to detect invalid operational conditions. Sensors are implemented to control the TOE internal voltages, the TOE internal frequency and the TOE internal temperature. In addition, the TOE implements digital fault sensors and fault detection logic as required by FPT_FLS.1.

Internal clock generation and filtering of the power supply provide the limited fault tolerance as required by FRU_FLT.2.

General countermeasures like the generation of additional temporal noise by various means (software and hardware), the memory protection mechanism implemented for SP-ROM and SP-RAM and the SP-QFPROM as well as the RTL obfuscation together with specific options and constrains for the physical layout provide the protection as required by FPT_PHP.3.

9.1.6 Access control and management (hardware)

9.1.6.1 Control memory areas shared with other components of the SoC

Access to the external RAM used to share data with other components of the SoC is controlled by a dedicated memory manager as required by FMT_CMT.1(1). The memory manager is able to support different windows in parallel.

9.1.6.2 Control access to keys and the key table

Access to keys stored in the key table is limited to the SP-CMU and does not allow software executed on the SP-CPU to compromise keys via software. This protection is required by FMT_CMT.1(2).

9.1.6.3 Control use of hardware support for cryptographic operations and random number generation

The access to coprocessors and the random number generator providing the cryptographic support to the software is limited to SP-CMU. This protection is required by FMT_CMT.1(3).

9.1.6.4 Control access to the SP-RAM by software on the SP-CPU

Access to the SP-RAM of the TOE is controlled by a Memory Management Unit (MMU that can only be configured in privileged SP-CPU mode as required by FMT_CMT.1(4).

9.1.7 Access control and management (operating system)

9.1.7.1 Cryptographic protection of persistent data stored outside the TOE

The confidentiality, integrity, authenticity and replay-protection of data stored in the non-volatile memory outside the TOE boundary is ensured by cryptographic mechanisms. The encryption of the data is required by FDP_SDC.1(2), the integrity protection is required by FDP_SDI.2(3), the authenticity is required by FDP_SDA.1 and the replay protection is required by FDP_SDR.1.

9.1.7.2 Cryptographic protection of transient data and code stored outside the TOE

The confidentiality, integrity, authenticity and replay-protection of data stored in the DDR memory outside the TOE boundary is ensured by cryptographic mechanisms. The encryption of the data is required by FDP_SDC.1(2), the integrity protection is required by FDP_SDI.2(3), the authenticity is required by FDP_SDA.1 and the replay protection is required by FDP_SDR.1.

9.1.7.3 Reallocation of shared resources

The operation system implements the protection of memory areas that may get accessible by other applications or processes based on re-allocation of resources. Buffers with sensitive parameters and memory areas that are de-allocated are cleared as required by FDP_RIP.1/Transient.

9.1.8 Logical protection

9.1.8.1 SP-ROM

The SP-ROM is only accessible to the SP-CPU based on the control of the SP-MMU and the associated permission checker of SP-ROM as required by FMT_CMT.1(5). The ROM is partitioned and ROM content can only be patched by OTP patch logic or CSR patch mechanism as required by FMT_CMT.1(6). The SP-ROM implements memory encryption and parity control as required by FDP_SDC.1(1) and FDP_SDI.2(1) .

9.1.8.2 SP-RAM

The SP-RAM access is restricted to SP-CPU based on the control of the SP_MMU and the SP-DMA controller as well as the associated permission checker of SP-RAM. The SP-RAM implements memory encryption and parity control as required by FDP_SDC.1(1) and FDP_SDI.2(1) . Further on a specific integrity check during warm boot is implemented as required by FDP_SDI.2(2).

9.1.9 Production data and OTP handling

Data for the identification of the TOE and the associated initialization and pre-personalization data is stored in the SP-QFPROM as required by FAU_SAS.1.

9.1.10 Lifecycle control

The TOE implements lifecycle control based on a combination of access control to TOE functionality and the coding of the lifecycle phase in the SP-QFPROM. This implements the requirements FMT_LIM.1 and FMT_LIM.2.

A References

A.1 Related documents

Title	Number
Standards	
Common Criteria for Information Technology Security Evaluation, Parts 1, 2, and 3, Version 3.1, Revision 4; [CC]	Part 1: CCMB-2017-04-001 Part 2; CCMB-2017-04-002 Part 3: CCMB-2017-04-003
Secure Hash Standard (SHS)	FIPS PUB 180-4
Digital Signature Standard (DSS)	FIPS PUB 186-4
Specification for the Advanced Encryption Standard (AES)	FIPS PUB 197
Keyed-Hash Message Authentication Code (HMAC)	FIPS PUB 198-1
Recommendation for Block Cipher Modes of Operation, Methods and Techniques	NIST SP 800-38A
Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication	NIST SP 800-38B
Recommendation for Random Number Generation Using Deterministic Random Bit Generators	NIST SP 800-90A
Resources	
Security IC Platform Protection Profile with Augmentation Packages, Version 1.0; [ICPP]	BSI-CC-PP-0084-2014
Java Card Protection Profile – Open Configuration, Version 3.0, [JCSPP]	May 2012

A.2 Acronyms and terms

Acronym or term	Definition
CSR	Configuration and status register
OTP	One-time programmable
PoP	Package-on-package
SFP	Security function policy
SFR	Security functional requirement
SoC	System-on-chip
SP-CMC	Crypto management controller
SP-CMU	Cryptographic management unit
SP-CPU	Central processing unit
SP-KT	Key table

Acronym or term	Definition
SP-LRM	Local resource manager
SP-SC	Security controller
SP-sCSR	Shared configuration and status registers
SPU	Secure Processing Unit
TIC	Test interface controller
TOE	Target of evaluation
TSF	TOE security function
TSS	TOE summary specification
xPU	External protection unit; (x is memory/register/address)

B Cryptographic mechanisms

Purpose	Cryptographic mechanism	Standard of implementation	Key size in bits/ Key types	Key origin
Image Authenticity and Integrity	RSA-signature verification with SHA-256 and PKCS#1 1.5 padding	FIPS 186-4 FIPS 180-4 RFC3447	2048	Qualcomm managed private key
Image Confidentiality	AES-CBC	FIPS 197 NIST SP800-38A	128	Qualcomm managed symmetric key
User and TOE data while in NVM Confidentiality	AES-CBC	FIPS 197 NIST SP800-38A	256	Key generated and managed by TOE operating system using TOE RNG
User and TOE data while in NVM Integrity and Authenticity	SHA-256 tree and AES in CMAC mode	FIPS-180-4 FIPS 197 NIST SP800-38B Hash tree specifics provided in developer document	256	Key generated and managed by TOE operating system using TOE RNG
User and TOE data while in external DDR Confidentiality	AES-CBC	FIPS 197 NIST SP800-38A	256	Key generated and managed by TOE operating system using TOE RNG
User and TOE data while in external DDR Integrity and Authenticity	SHA-256	FIPS 180-4	256	N/A Reference hash stays in TOE
API	TDES (ECB, CBC)	FIPS46-3 SP800-38A	168	
API	AES (ECB, CBC, CTR, CMAC, CCM)	FIPS 197 NIST SP800-38A NIST SP800-38B NIST SP800-38C NIST SP800-38E	128, 256	User-provided key managed by the TOE or key generated and managed by TOE operating system or CMU using TOE RNG
API	SHA1 SHA-256 SHA-384 SHA-512	FIPS 180-4	none	N/A
API	RNG	PTG.3 According to AIS-31 NIST SP800-90A	none	N/A
API	HMAC-SHA1 HMAC-SHA256 HMAC-SHA-384 HMAC-SHA-512	FIPS198-1	256 (SHA-1, SHA-256) 512 (SHA-384, SHA-512)	User-provided key managed by the TOE or key generated and managed by TOE operating system or CMU using TOE RNG

Purpose	Cryptographic mechanism	Standard of implementation	Key size in bits/ Key types	Key origin
API	Generate random symmetric key	NIST SP800-133	128, 192, 256	
API	Key Derivation Function in Counter Mode based on CMAC AES-256	NIST SP800-108 FIPS 198-1 FIPS 180-4	256	
API	RSA signature/verify with SHA-1, SHA-256, SHA-384, SHA-512 and RSASSA-PSS and PKCS#1 v1.5 padding schemes	FIPS 186-4 FIPS 180-4 RFC3447	1024, 2048	User-provided key or key generated by TOE operating system using TOE RNG
API	RSA encryption / decryption with RSAES-OAEP and PKCS#1 v1.5 padding schemes	FIPS186-4 RFC3447	1024, 2048	User-provided key or key generated by TOE operating system using TOE RNG
API	ECDSA Cryptographic key generation Signature Generation / Verification	FIPS186-4 Appendix B.4.2	ECC key lengths corresponding to following ECC parameters: NIST P-192, NIST P-224, NIST P-256, NIST P-384, NIST P-521	User-provided key or key generated by TOE operating system using TOE RNG
API	ECDH Cryptographic key generation Shared secret generation	SP800-56A revision 2 section 5.6.1.2 SP800-56A revision 2 section 5.7.1.2 FIPS 186-4 Appendix D.1.2	ECC key lengths corresponding to following ECC parameters: NIST P-192, NIST P-224, NIST P-256, NIST P-384, NIST P-521	User-provided key or key generated by TOE operating system using TOE RNG