User's Manual

CONFIDENTIAL

Security Board Support Package

User's Manual: Software

R-Car Series, 3rd Generation

— Preliminary —
Specifications common to R-Car Series Products
R-Car H3
R-Car M3
R-Car M3N
R-Car E3
R-Car D3

Renesas Electronics
www.renesas.com

Rev.3.0.4 Dec. 2021

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Trademark

How to Use This Manual

- **[Readers]**
  This manual is intended for engineers who develop products which use the R-Car H3/M3/M3N/E3/D3 processor.

- **[Purpose]**
  This manual is intended to give users an understanding of the functions of the R-Car H3/M3/M3N/E3/D3 processor device driver and to serve as a reference for developing hardware and software for systems that use this driver.

- **[How to Read This Manual]**
  It is assumed that the readers of this manual have general knowledge in the fields of electrical
  — engineering, logic circuits, microcontrollers, and Linux.
  → Read this manual in the order of the CONTENTS.
  — To understand the functions of a multimedia processor for R-Car H3/M3/M3N/E3/D3
  → See the R-Car H3/M3/M3N/E3/D3 User's Manual.
  — To know the electrical specifications of the multimedia processor for R-Car H3/M3/M3N/E3/D3
  → See the R-Car H3/M3/M3N/E3/D3 Data Sheet.

- **[Conventions]**
  The following symbols are used in this manual.
  Data significance: Higher digits on the left and lower digits on the right
  Note: Footnote for item marked with Note in the text
  Caution: Information requiring particular attention
  Remark: Supplementary information
  Numeric representation: Binary ... ××××, 0b××××, or ××××B
  Decimal ... ××××
  Hexadecimal ... 0x×××× or ××××H
  Data type: Double word … 64 bits
  Word … 32 bits
  Half word ... 16 bits
  Byte ... 8 bits

1.Table of Contents

1. Overview

1.1 Overview

This manual explains how to use the R-Car H3/M3/M3N/E3/D3/H3e/M3e/M3Ne/E3e/D3e Security Board Support Package.

1.2 Function

After initializing R-Car H3/M3/M3N/E3/D3 Hardware, there are some modules executed, which is following Figure 1.1. The scope of Security Board Support Package is the Trusted Firmware-A (BL31) and the OP-TEE OS and the OP-TEE driver and the OP-TEE Client. This package provides environment that is customized source code distributed from the GitHub for the R-Car H3/M3/M3N/E3/D3 Hardware.

CPU cores on R-Car H3/M3/M3N/E3/D3 System Hardware that applied Secure Board Support Package are separated into the two worlds by Armv8-A TrustZone technology, one is the Secure World for trusted OS and the other is the Normal World for Non-secure OS. The Secure world is a trusted execution environment (TEE) enabling a strong security that protects secure resources (secret Device Keys, decrypted content data, Hardware Encryption etc.) from the Normal World called as Rich Execution Environment (REE). Programs running in the Secure World can access both secure and normal resources, but programs running the Normal World can only access normal resources.

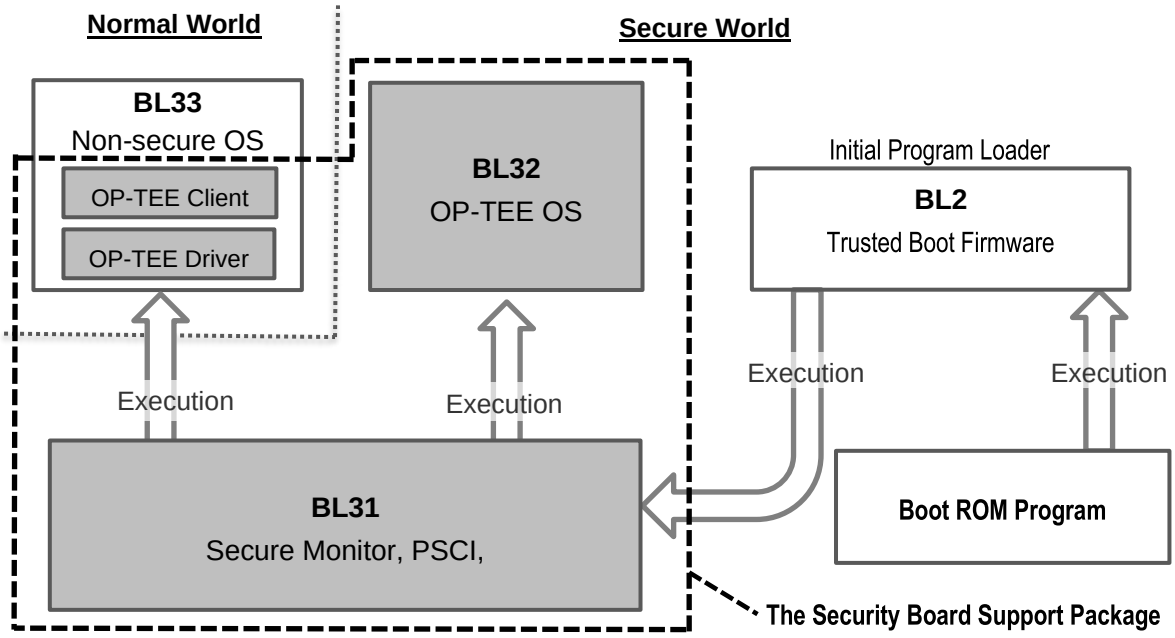


Figure 1.1 Scope of the Security Board Support Package

1.2.1 Trusted Firmware-A (BL31)

The Trusted Firmware-A (BL31) is the Secure World software including a Secure Monitor, various Arm interface standards such as the Power State Coordination Interface (PSCI), and also supports SMC's SiP Service.

•Secure Monitor

The Secure monitor manages the switches between the Secure World and the Normal World. When a SMC, FIQ and IRQ are generated, the Exception Handler decides to need to switch the world. If it needs to switch the world, the Secure Monitor saves register data and restore register for the next world.

See “1.3.1Related Document No.5” for details on the specification for SMC.

•PSCI

PSCI is the interface from the Normal World software to firmware implementing power management use-cases, Secondary CPU Boot, CPU Hotplug, CPU Idle and System Shutdown/Reset/Suspend.

See “1.3.1Related Document No.4” for details on the specification for PSCI.

System Shutdown/Reset/Suspend is implementation dependent on the PMIC mounted on the evaluation board Salvator-X(refer to "1.3.1 Related Document No.8 and No.9”)/Salvator-XS(refer to "1.3.1 Related Document No.10, No.11 and No.12”)/Ebisu(refer to "1.3.1 Related Document No.17”)/Ebisu-4D(refer to "1.3.1 Related Document No.18”).

For details of PMIC function, refer to "1.3.1 Related Document No.7".

SYSTEM_SUSPEND API implements Suspend to RAM. Suspend to RAM operating uses I2C for DVFS to control the power management IC. Program for operating I2C for DVFS is copied on System RAM and execute on there.

Suspend: When Non-secure OS calls SYSTEM_SUSPEND, BL31 executes the following processing.

- Backup Secure World register.(GIC ,CCI Generic Timer)
- Set DRAM to DDR self-refresh mode.
- Set ‘BKUP_CTRL_OUT‘ bit of BKUP Mode Cnt’ register on the power management IC to halt all power supplies except power supply for DRAM via program for operating I2C for DVFS.

Resume: At next booting, Initial Program Loader skips image load and cancels DDR self-refresh mode and sets parameter boot type for BL31. BL31 executes the following processing.

- Judge boot type parameter (register X1). If parameter is set 1, boot type is resumption from Suspend to RAM.
- Restore Secure World register. (GIC ,CCI Generic Timer)
- Go to the entry point of Non-secure OS.
- Generic timer restarts from the counter value backed up at entering Suspend state.

•Logging

The BL31 writes logs in the ring buffer. The ring buffer is implemented using an information header that contained character string (“TLOG”) and start of valid log for full buffer (Index) and amount of data currently in the buffer (Size).

Note) if the ring buffer is not full, start of logs is 0. (RCAR_BL31_LOG_BASE+Header size(0x16))

Layout of the logging buffer in the memory is Figure 1.2. LOG_BASE and LOG_SIZE is shown the following Definitions 3.3.1, a log_header is shown the following Structure 3.4.1.

If users want to get the logs of the ring buffer, users will dump on the system aborted or an appropriate timing for users (break point debugging etc.).

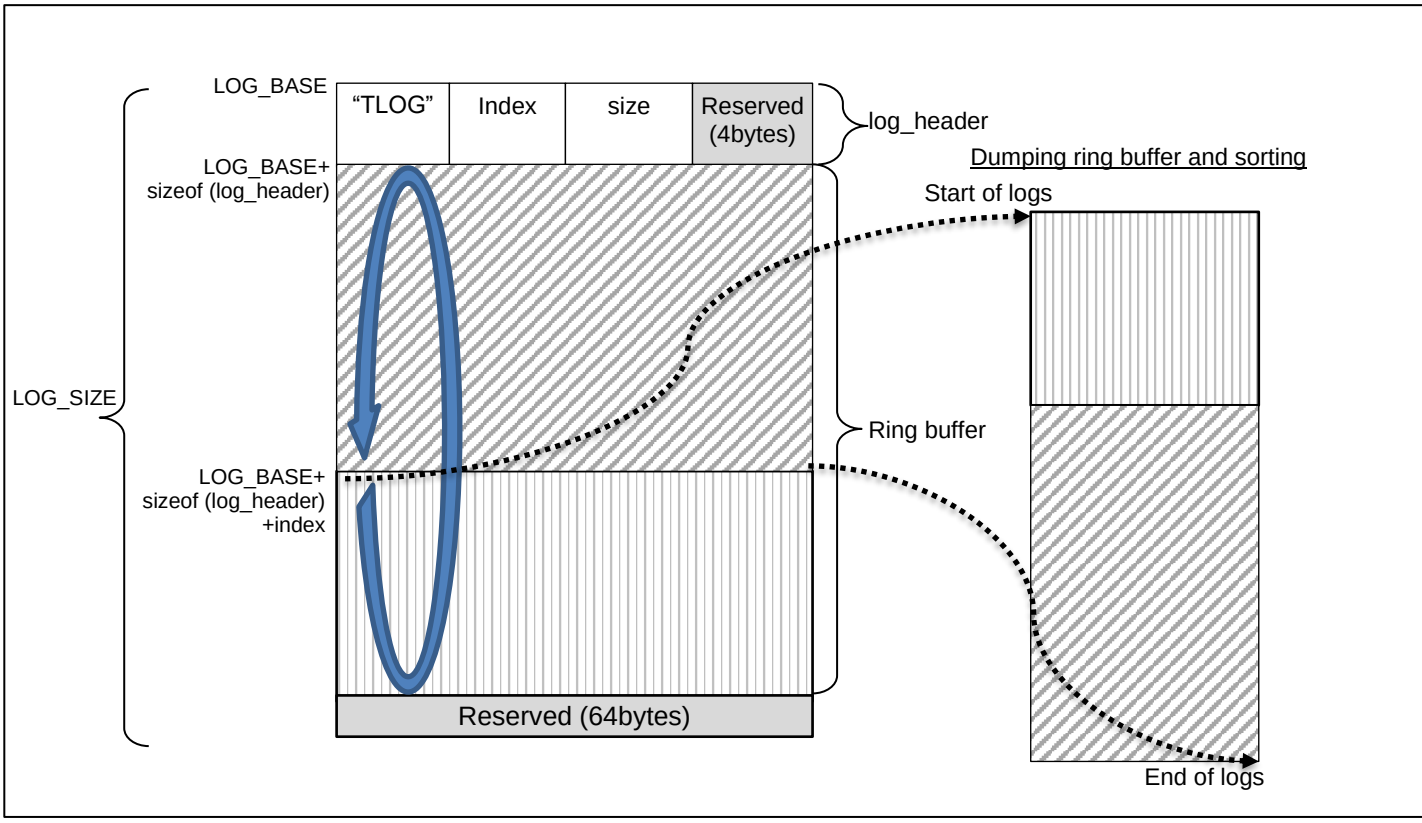


Figure 1.2 Layout of the buffer for logs

One of the logs is formed of two Message blocks, and a maximum number of characters is 256bytes. A format of a log and summaries of Message blocks are shown below.

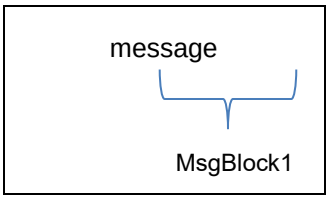


Figure 1.3 Format of a log

Table 1.1 Message block

Message block	summary
MsgBlock1	Messages are set in source code for the BL31. This package is prepared MACROs for users, which are ERROR(), NOTICE(),WARN(), INFO(). Each MACROs are controlled output by the log level. Relationship of the log level and output are described in the following 4.3.1

Note) The log output of BL31 is supported only the startup process, during runtime processing is not supported.

1.2.2 OP-TEE OS

The OP-TEE OS is a Trusted OS which is running in the Secure World. This package provides the TEE Internal API as defined by the GlobalPlatform TEE standard to the Trusted Applications that it executes and accesses secure resources.

•Trusted application (TA)

Applications running in the OP-TEE OS are called TA.
TA is a passive type of application.
TA receives the request command from Client Application (CA), to execute it. And, return the results to the CA.

CA running inside of the Normal World makes use of the TEE Client API to access facilities provided by TA inside the Secure World.

- **Provided functions for TA**

  TEE-Client API and TEE Internal API provide the set of functions as the following Table 1.2 Functions to TA developers.

Table 1.2 Functions

No.	Function	Description
1	Trusted Core Framework	Infrastructure and core framework.
2	Trusted Storage	To store the keys and general-purpose data.
3	Cryptographic Operations	Encryption functions
4	Time	This function provides access to three source of time. ・System Time ・TA Persistent Time ・REE Time
5	Arithmetical	Arithmetic processing to be used in the encryption process.
6	Socket	The generic C interface used by a TA to establish and utilize network communications to a remote server using a socket style approach.

- **TA Type**

The following TA types are supported.

Table 1.3 TA Types

Type	Description
Dynamic TA	・The Dynamic TA execute each processing through TEE Internal API. ・The execution image of the Dynamic TA is not included in OP-TEE OS image. It is necessary to build individually. ・Store the execute file of TA in filesystem of the Non-secure OS.
pseudo TA	・The pseudo TA must call the OP-TEE kernel function directly instead of calling the TEE Internal API, because it runs in kernel context. ・The execution image of the pseudo TA is included in OP-TEE OS image.
early TA	・The early TA executes each processing through TEE Internal API. ・The execution image of the early TA is included in OP-TEE OS image.

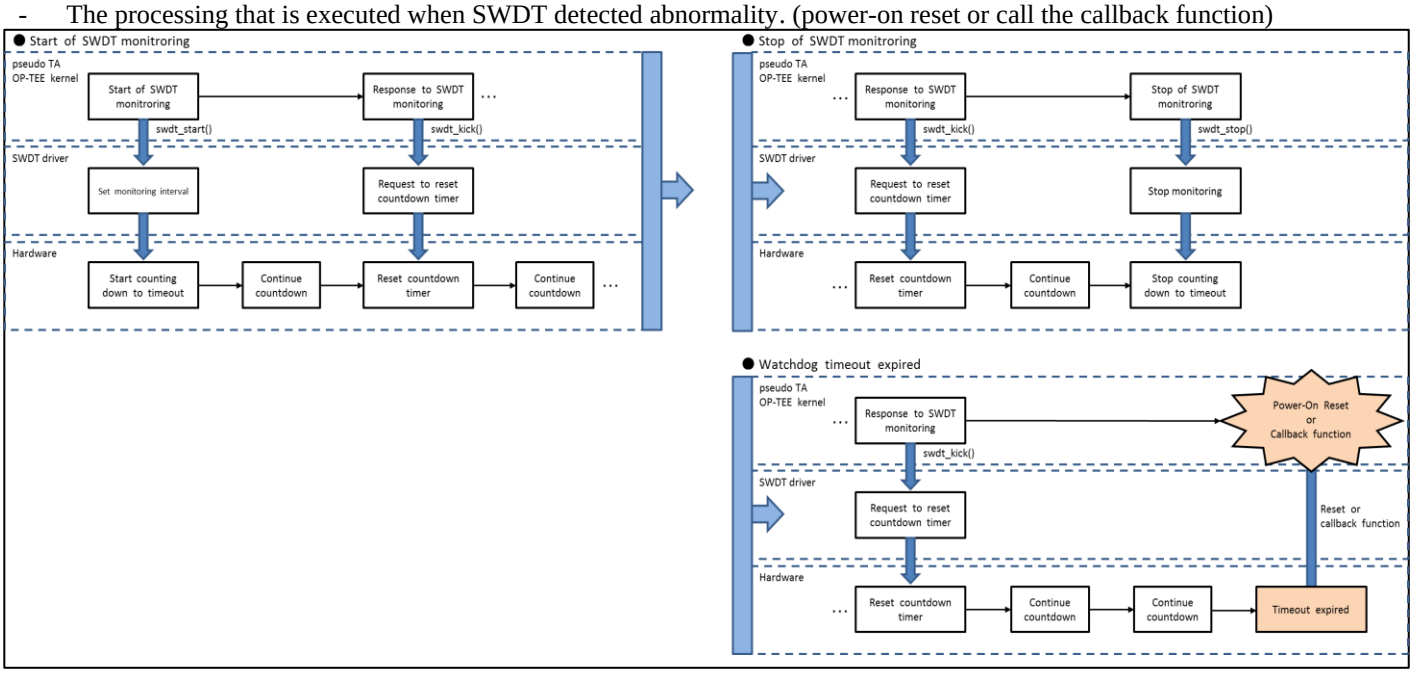
•Watchdog Timer

The OP-TEE OS provides driver for System Watchdog timer (SWDT). This driver can be controlled by a pseudo TA.

The SWDT is a single-channel timer that uses the OSCCLK as an input clock and can be used as a watchdog timer. OSCCLK is a clock which is generated by CPG.

SWDT API provides the function of the abnormal detection and enables the following setting.

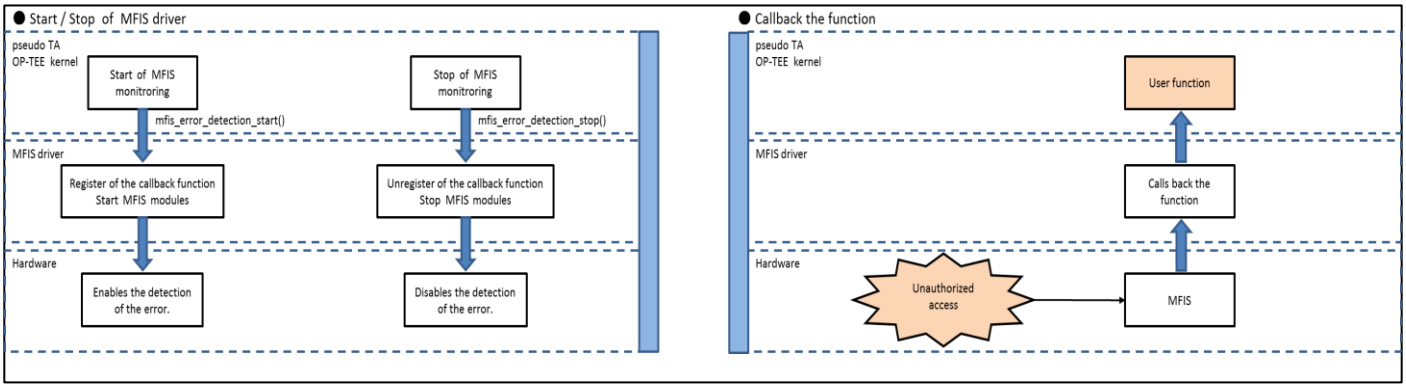
- The monitoring interval.
- The processing that is executed when SWDT detected abnormality. (power-on reset or call the callback function)



•MFIS Driver

The OP-TEE OS provides driver for MFIS Driver. This driver can be controlled by a pseudo TA.

The MFIS driver is a driver for detecting a security error at the interrupt handler.
When an interrupt is detected calls the callback function.

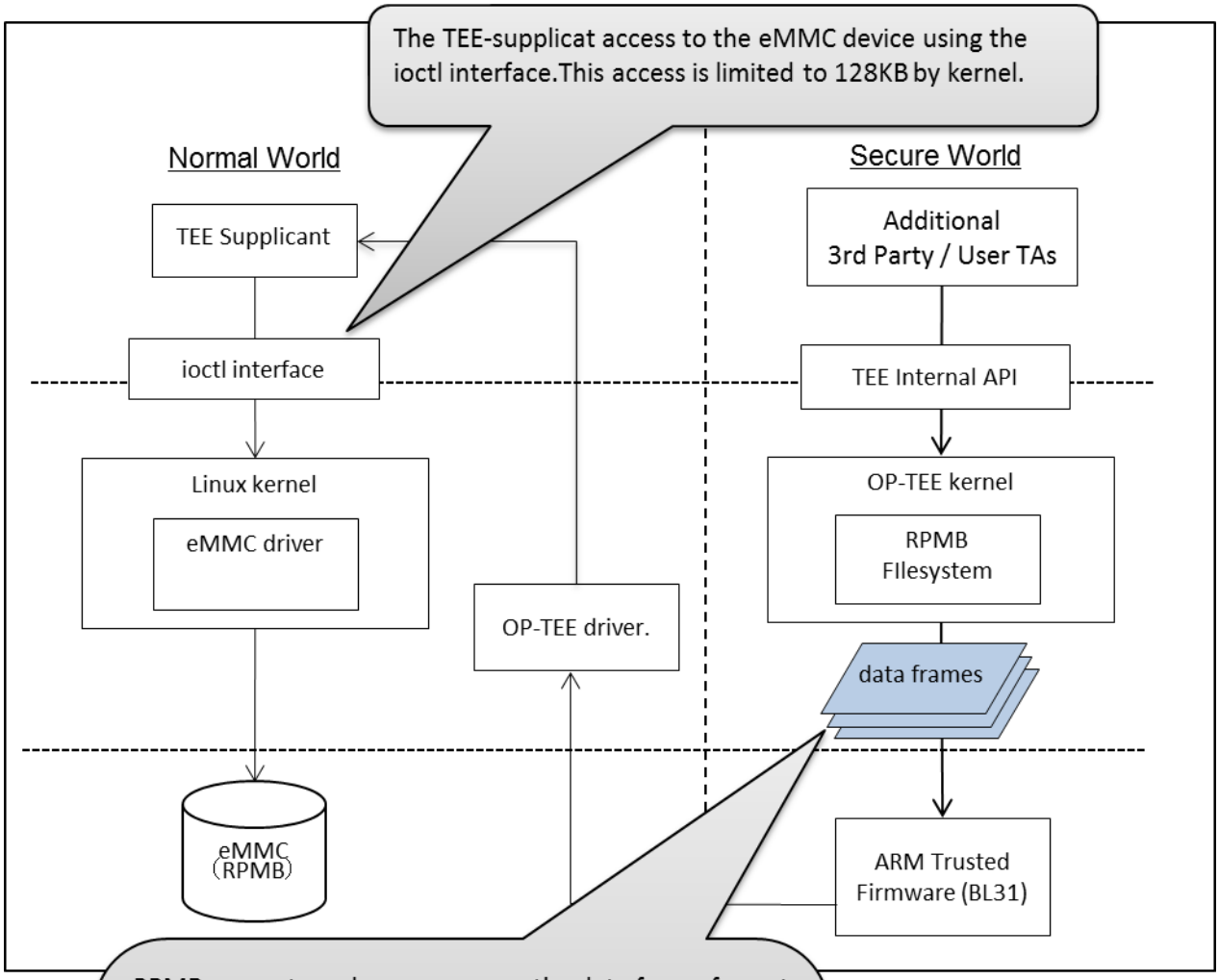


•Secure Storage

The OP-TEE OS provides storage according with GlobalPlatform's TEE Internal API specification. Secure Storage should be possible to store general-purpose data and key material that guarantees confidentiality.

There are three secure storage implementations in the OP-TEE OS:

(1) REE Filesystem
A storage relies on the Normal World (REE) file system.
(2) Stand-alone Filesystem
A storage stores a SPI / HyperFlash Memory in the Secure World.
(3) RPMB Filesystem
A storage makes use of the Replay Protected Memory Block (RPMB) partition of an eMMC device.
Note) Restrictions of the RPMB Filesystem, User TA can write at most 65,500 bytes of data.



RPMB requests and responses use the data frame format defined by the JEDEC eMMC specification.
Because of ioctl's 128 KB limit, OP-TEE can request 256 frames.
However, OP-TEE has 36 bytes of internal data and user TA can actually write 65,500 bytes of data.*1

*1: The frame data is 512 bytes and 256 bytes user data area is formed therein. Maximum data size = 256frames * 256bytes - 36byte = 65,500byte.

•SPI/HyperFlash Driver

The OP-TEE OS provides driver for SPI/HyperFlash Driver. This driver can be controlled by a secure storage (Stand-alone Filesystem).
The SPI/HyperFlash Driver can be control erase/read/write to flash memory.
Operating Frequency differs depending on the non-volatile memory as follows.

Table 1.4 Operating Frequency for non-volatile memory

non-volatile memory	model	control target	Maximum Operating Frequency(MHz)
HyperFlash	Cypress S26KS512S (*1)	erase	(*3)
		read	(*3)
		write	40 (*2)
QSPI Flash (on Board)	Cypress S25FS128S (*1)	erase	80
		read	80
		write	80

Note:
(*1) These flashes are mounted in the Evaluation Board described in chapter 2.1.
SPI/HyperFlash Driver are testing the operation on only these flashes.
(*2) HyperFlash writes using burst mode.
Burst mode must be used with HyperFlash HW limit of 50 MHz or less.
The maximum operating frequency that can be set with RPC register below this HW limit is 40 MHz.
(*3) Table 1.5 shows the operating frequencies of read and erase for each SoC.

Table 1.5 Operating Frequency of HyperFlash (erase and read)

SoC	Cut Version	Maximum Operating Frequency(MHz)
R-Car H3	Ver.3.0 or later	160
R-Car M3	Ver.1.2 or later	160
R-Car M3N	Ver.1.1 or later	160
R-Car E3	Ver.1.1 or later	150
R-Car D3	Ver.1.1 or later	150

•Logging

Logging of the OP-TEE OS is not only writing to the ring buffer, but also sending to the OP-TEE driver.

Logging to the ring buffer is the same as BL31. It uses an information header, too, however only the character string "SLOG" is different from the BL31. Layout of the logging buffer in the memory is shown below in the Figure 1.2 for BL31. LOG_BASE and LOG_SIZE (LOG_RAM_MAX_SIZE) are shown the following Definitions 3.3.1, a log_header is shown the following Structure 3.4.1.

If users want to get the logs of the ring buffer, users will dump at the system aborted or an appropriate timing for user (break-point debugging etc.).

There are three Message blocks for logging in the OP-TEE OS. One of the logs for ring buffer is formed of two Message blocks, and a maximum number of characters are 256bytes. The format of a log and summaries of Message blocks are shown below.

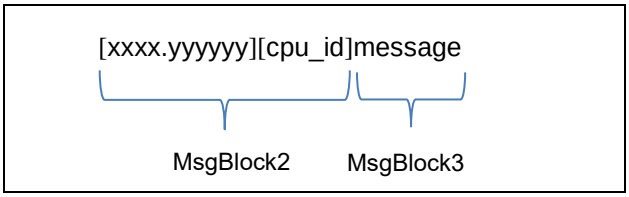


Figure 1.4 Format of a log for ring buffer

Table 1.6 Message block

Message block	summary
MsgBlock1	Prefix [OP-TEE]
MsgBlock2	Time [second.micro second] Length of second: 1-10 bytes Length of micro second: 6bytes
	core number [0-7]
MsgBlock3	Messages are set in source code for the OP-TEE OS and TAs. This package is prepared MACROs for users, which are MSG(), EMSG(), IMSG(), DMSG(), FMSG(). Each MACROs are controlled output by the log level. Relationship of the log level and output are described in the following 0

The OP-TEE OS built on debugging mode can send logs to the OP-TEE driver. The OP-TEE OS uses shared memory to send log messages to the Normal World. Shared memory area for log messages is separated for each CPU, which is shown the following Figure 1.5. When sending log messages, the OP-TEE OS writes log messages to the memory that was allocated in accordance with the CPU number invoking logging function, and send a command for logging (customized R-Car H3/M3/M3N/E3/D3 System, TEE_RPC_DEBUG_LOG is shown the following Definitions 3.3.2) to the OP-TEE driver by SMC. LOG_NS_BASE and LOG_NS_CPU_AREA_SIZE are shown the following Definitions 3.3.2.

When the OP-TEE driver receives command by SMC, it will get the log from the shared memory and output the log to its terminal immediately.

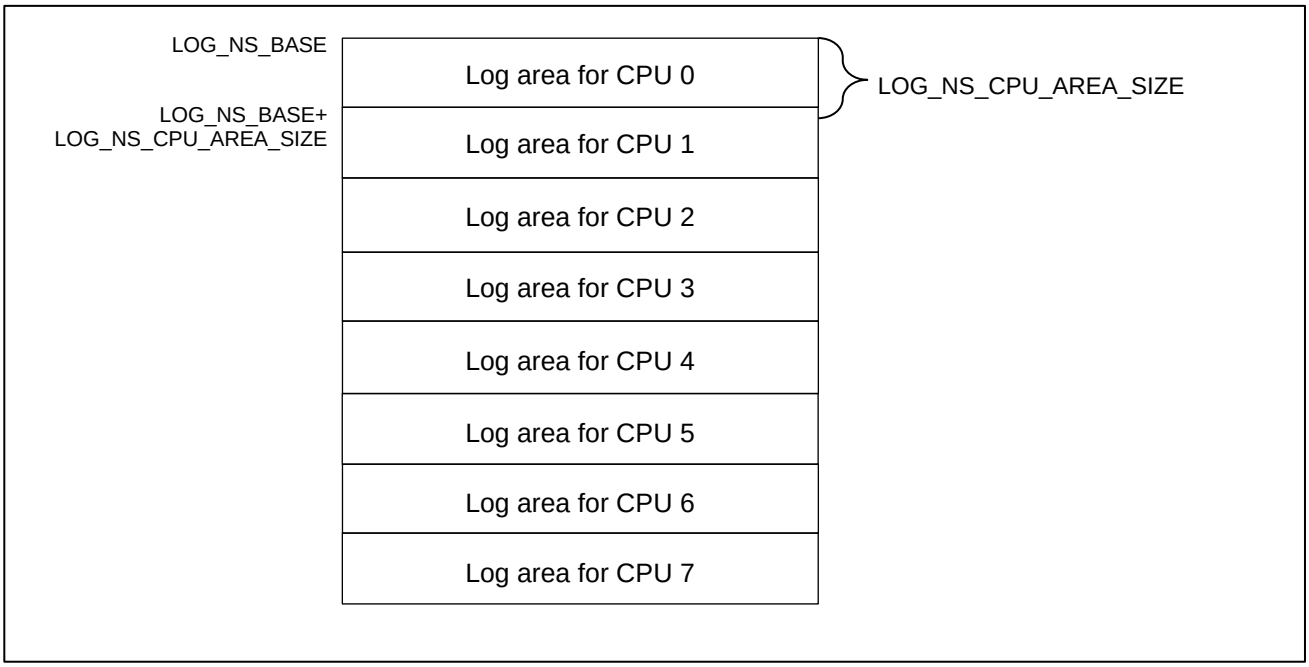


Figure 1.5 Layout of the shared memory for logs

A logs for OP-TEE driver (shared memory) is formed of three Message blocks, and a maximum number of characters is 256bytes. The format of a log and summaries of Message blocks are followed in Table 1.6 Message block

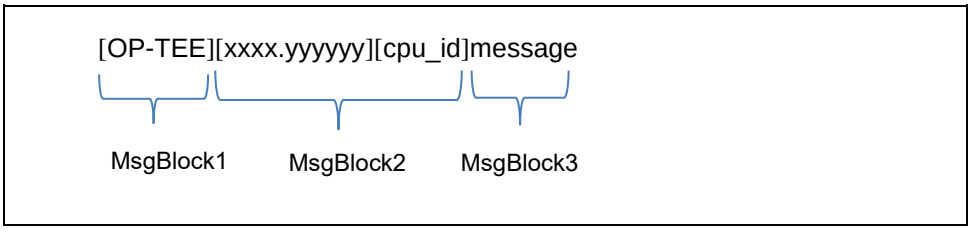


Figure 1.6 Format of a log for OP-TEE driver

•Unsupported features

Among the features that the OP-TEE OS of OSS supports, Table 1.7 shows the features which are not supported in the following version up when the platform is rcar.

Table 1.7 Unsupported features (platform: rcar)

No.	Features	Description	Version
1	Secure Data Path	Client and Trusted Applications can share references to secure memory.	tag 2.2.0 - tag 2.6.0
2	Dynamic shared memory	Non-contiguous, non-secure memory can be mapped into Trusted Applications VA space.	tag 2.2.0 - tag 2.6.0
3	Embedded secure device tree	Device tree file is not supported. The device tree source file is compiled into a device tree blob file which content is embedded in a read-only section of the core.	tag 3.1.0 - tag 3.8.0

•SuspendToRAM

The OS (Linux) that runs in Normal World has a power saving function called Suspend ToRAM that cuts off the power supply of some HWs except CPU and SDRAM.OP-TEE performs necessary processing when transitioning to or returning to the power saving state, but the user needs to be aware of the following.

1. Lnux is waiting for the transition of the power saving state by SuspnedToRAM while OP-TEE is executing the TA process. Therefore, the user needs to complete the TA process of OP-TEE before executing SuspendToRAM.
2. When OP-TEE move to the power saving state, FIQ interrupt mask and issuance of SMC is prohibited, and interrupts are disabled. When OP-TEE returns from a power-saving state, the FIQ interrupt mask and issuance of SMC is permited and interrupts are enabled.

1.2.2.1 Interrupt controller

SGI (Software-generated interrupt) and PPI (Private Peripheral Interrupt) and SPI (Shared Peripheral Interrupt) are defined as an interruption of the GIC (Generic Interrupt Controller).

Table 1.8 Kind of the interrupt

Kind	Interrupt ID	Description
SGI	0 to 15	Interrupt to be used between the CPU, which is issues by the software.
PPI	16 to 31	Unique interrupt in the CPU (such as the Generic Timer)
SPI	32 to 480	Interrupt from the built-in IP within the SoC.

The following is a classification of interrupts that GIC to be issued.
FIQ: The interrupt for the Secure World.
IRQ: The interrupt for the Normal World.

The following is a classification of the interrupt to be used as the FIQ.

Table 1.9 List of FIQ

Kind	Interrupt ID	Description
SGI	8 to 15	Reserved as software interrupt.
PPI	29	Interrupt for Secure physical timer.
SPI	70	Interrupt for RPC.
	97	Interrupt for Secure Engine sec.
	98	Interrupt for Secure Engine pub.
	102	Interrupt for Secure Engine sec.
	166	Interrupt for System Timer.
	173	Interrupt for System Watchdog Timer.
	446, 447	Interrupt for MFIS error.

1.2.3 OP-TEE Driver

The OP-TEE Driver for Non-secure OS allows communication between the Non-secure OS and the OP-TEE OS.

The OP-TEE Driver after updating to v1.0.8 shifts from a loadable Module to a static module bundled with the Linux Kernel. Confirming the version of the OP-TEE Driver refers to 3 Related Document No.16.

•Linux

The OP-TEE Driver can output logs from the OP-TEE OS that is built on debugging mode to the Linux terminal. When the OP-TEE Driver receives command for logging (TEE_RPC_DEBUG_LOG is shown 3.3.3) from the OP-TEE OS, it gets message from shared memory that address obtains its own CPU number and outputs log to the Linux terminal.

In the case of the CFG_TEE_CORE_LOG_LEVEL is over DEBUG (3), the behavior of Linux might be unstable when OP-TEE outputs a lot of terminal logs.

The OP-TEE Driver provides TEE Client API for the kernel mode and these APIs have the same function as TEE Client API provided to the user mode. Since TEE Client API for the kernel was deleted after updating to v1.0.8, it was additionally implemented for R-Car H3/M3/M3N/E3/D3.

1.2.4 OP-TEE Client

This package provides the TEE client library and the TEE supplicant. TEE client library is the library that contains APIs defined by the GlobalPlatform TEE standard. This library is used CA that are executed on Non-secure OS to communicate with the OP-TEE OS and TAs.

TEE Supplicant operates miscellaneous features of the OP-TEE OS in Secure World, such as file system access and loading a Dynamic TA.

•Linux

TEE Supplicant is a daemon serving the OP-TEE OS in Secure World.

The OP-TEE Client for Linux does not change source code from original software.

1.3 References

1.3.1 Related Document

The following table shows the document related to this function.

Table 1.10 Related Document

No.	Issue	Title	Edition
1	Renesas Electronics	Linux Interface Specification Yocto recipe Start-Up Guide	#0
2	GlobalPlatform	TEE Client API Specification	Ver.1.0
3	GlobalPlatform	TEE Internal Core API Specification	Ver.1.1
4	Arm	POWER STATE COORDINATION INTERFACE (PSCI)	ARM DEN 0022C
5	Arm	SMC CALLING CONVENTION	ARM DEN 0028B
6	Renesas Electronics	Initial Program Loader User's Manual	#0
7	Renesas Electronics	R-Car H3/M3/M3N/E3/D3 Series, Linux Interface Specification Power Management User's Manual: Software	#0
8	Renesas Electronics	R-CarH3-SiP System Evaluation Board Salvator-X Hardware Manual RTP0RC7795SIPB0011S	#0
9	Renesas Electronics	R-CarM3-SiP System Evaluation Board Salvator-X Hardware Manual RTP0RC7796SIPB0011S	#0
10	Renesas Electronics	R-CarH3-SiP System Evaluation Board Salvator-XS Hardware Manual RTP0RC7795SIPB0012S	#0
11	Renesas Electronics	R-CarM3-SiP System Evaluation Board Salvator-XS Hardware Manual RTP0RC7796SIPB0012S	#0
12	Renesas Electronics	R-Car M3N-SiP System Evaluation Board Salvator-XS Hardware Manual RTP0RC77965SIPB012S	#0
13	GlobalPlatform	TEE Sockets API Specification	Ver.1.0.1
14	GlobalPlatform	TEE Sockets API Annex A: TCP/IP Specification	Ver.1.0.1
15	GlobalPlatform	TEE Sockets API Annex B: UDP/IP Specification	Ver.1.0.1
16	Renesas Electronics	Security Board Support Package Release Note	#0
17	Renesas Electronics	R-Car E3-SiP System Evaluation Board Ebisu Hardware Manual RTP0RC77990SEB0010S	#0
18	Renesas Electronics	R-Car E3-SiP System Evaluation Board Ebisu-4D Hardware Manual RTP0RC77990SEB0020SA00	#0
19	Renesas Electronics	R-Car Series, 3rd Generation User's Manual: Hardware	#0
20	Renesas Electronics	R-Car D3-SiP System Evaluation Board Draak Hardware Manual RTP0RC77995SEB0010S	#0

#0 : This manual refers to the latest edition.

1.3.2 Related Site for original software

The following table shows the original software related to this function.

Table 1.11 Related original software

No.	Software	Title and URL	Edition
1	Trusted Firmware-A (BL31)	Secure Monitor https://github.com/ARM-software/arm-trusted-firmware	#1
2	OP-TEE OS	Trusted side of the TEE https://github.com/OP-TEE/optee_os	#1
3	OP-TEE Driver	Normal World driver https://git.kernel.org/pub/scm/linux/kernel/git/geert/renesas-drivers.git Source code: drivers/tee	#1
4	OP-TEE Client	Normal World Client side of the TEE https://github.com/OP-TEE/optee_client	#1
5	optee_test	OP-TEE Test suite https://github.com/OP-TEE/optee_test	#1
6	OP-TEE OS	Virtualization https://optee.readthedocs.io/en/latest/architecture/virtualization.html	#1

#1 : Editions of the current release, see Security Board Support Package Release Note.

1.4 Restrictions

The following are restrictions of this Security Board Support Package.

No.	Descriptions	Impact on customers	Note
1	SoC:H3/M3/M3N/E3/D3 Target module:Trusted Firmware-A(BL31) Function of CPU_OFF cannot call from CPU0 in master boot processor.	CPU0 of master boot processor cannot be CPU_OFF.	-
2	SoC:H3/M3/M3N/E3/D3 Target module:OP-TEE OS Client API restrictions see “3.2.2.1 TEE Client API”	Some APIs are not supported.	-
3	SoC:H3/M3/M3N/E3/D3 Target module:OP-TEE OS TEE Internal API restrictions see “3.2.2.2 TEE Internal API”	Some APIs are not supported.	-
4	SoC:H3/M3/M3N/E3/D3 Target module:OP-TEE OS RPMB secure storage restrictions see “1.2.2 OP-TEE OS”	The maximum amount of data that can be written is 65,500 bytes.	-
5	SoC:H3/M3/M3N/E3/D3 Target module:OP-TEE OS When CFG_VIRTUALIZATION is enabled, please use REE Filesystem instead of Stand-alone Filesystem/RPMB Filesystem.	When CFG_VIRTUALIZATION is enabled, the available secure storage is the REE file system.	-
6	SoC:D3 Target module:Trusted Firmware-A(BL31)/OP-TEE OS SuspendToRAM is not available in D3 because Draak board is not equipped with the PMIC.	SuspendToRAM is impossible with D3.	-
7	SoC:H3/M3/M3N/E3/D3 Target module:OP-TEE OS If the close session process on Linux TEE Driver side fails due to a shared memory acquisition failure, the close session process is not performed on the OP-TEE side from Linux TEE Driver.	If the close session process on Linux TEE Driver side fails , the memory allocated at the time of open session in OP-TEE is not released, so memory exhaustion may occur in OP-TEE side. OP-TEE Client output following log "ERR [165089] TEEC:TEEC_CloseSession:674: Failed to close session 0x21"	When multiple processes are executed in parallel. (e.g. When more than 10 processes are executed in parallel.) Refer to section 5 for how to improve Linux TEE Driver to retry if the close session fails.
8	Target module:OP-TEE OS When virtulization is enabled (CFG_VIRTUALIZATION=y), TEE Client API might return TEEC_ERROR_OUT_OF_MEMORY. Frequency of this error can be reduced by making the heap size bigger with	When virtualization is enabled, the operation may become unstable.	Refer to “4.4.1 OP-TEE Example C” for OP-TEE OS Heap resizing procedure.

	CFG_CORE_HEAP_SIZE when build optee_os.		

Note) Restrictions of the current release, see Security Board Support Package Release Note.

1.5 Terminology and Abbreviation

The following table shows the terminology and Abbreviation related to this package.

Table 1.12 Terminology and Abbreviation

Terms	Explanation
Exception Levels (EL0/EL1/EL3)	The Armv8-A architecture defines a set of Exception levels, EL0 to EL3, where: • If ELn is the Exception level, increased values of *n* indicate increased software execution privilege. • Execution at EL0 is called unprivileged execution • EL2 provides support for virtualization of Non-secure operation. • EL3 provides support for switching between two Security states, Secure state and Non-secure state.
Secure World	It is one of the security states that defined Armv8-A architecture. When in this state, the CPU can access both the Secure and Non-secure space.
Normal World	It is one of the security states that defined Armv8-A architecture. When in this state, the CPU can access only Non-secure space.
PSCI	Power State Coordination Interface It defines a Standard interface for power management that can be used by OS vendors for supervisory software working at different levels of privilege on an Arm device.
SMC	Secure Monitor Call. An Arm assembler instruction that causes an exception that is taken synchronously into EL3.
Client Application (CA)	An application running outside of the Trusted Execution Environment making use of the TEE Client API to access facilities provided by Trusted Applications inside the Trusted Execution Environment.
Rich Execution Environment (REE)	An environment that is provided and governed by a Non-secure OS, potentially in conjunction with other supporting operating systems. It is outside of the TEE.
Trusted Application (TA)	An application running inside the Trusted Execution Environment that provides security related functionality to Client Applications outside of the TEE or to other Trusted Applications inside the Trusted Execution Environment.
Trusted Execution Environment (TEE)	An execution environment that runs alongside but isolated from an REE. A TEE has security capabilities and meets certain security-related requirements: It protects TEE assets from general software attacks, defines rigid safeguards as to data and functions that a program can access, and resists a set of defined threats.
RPMB	Replay Protected Memory Block
SWDT	System Watchdog Timer
MFIS	Multifunctional interface
SPI/HyperFlash	Serial Peripheral Interfase Flash Non-Volatile Memory / HyperFlash Non-Volatile Memory

Suspend to RAM	The function that halts all power supplies except power supply for DRAM. At next booting, boot program skips image load and initialization of the system.
Dynamic TA SDK	Files necessary to build Dynamic TA.
CA SDK	Files necessary to build CA.
SiP Service	Provides interfaces to SoC implementation-specific services on this platform.

2. Operating Environment

2.1 Hardware Environment

The following table lists the hardware needed to use this function.

Table 2.1 Hardware environment (R-Car H3/M3/M3N/E3/D3)

Name	Explanation
Evaluation Board	R-Car H3 SiP System Evaluation Board (Salvator-X/Salvator-XS) Renesas Electronics R-Car M3 SiP System Evaluation Board (Salvator-X/Salvator-XS) Renesas Electronics R-Car M3N SiP System Evaluation Board (Salvator-XS) Renesas Electronics R-Car E3 SiP System Evaluation Board (Ebisu/Ebisu-4D) Renesas Electronics R-Car D3 SiP System Evaluation Board (Draak) Renesas Electronics
Host PC 1	It is used as debugging environment. Terminal software is executed.
Host PC 2 (Linux)	TFTP server software It is used when HyperFlash is written by U-Boot or Image is downloaded.
	NFS server software It is used when File system is mounted by NFS.

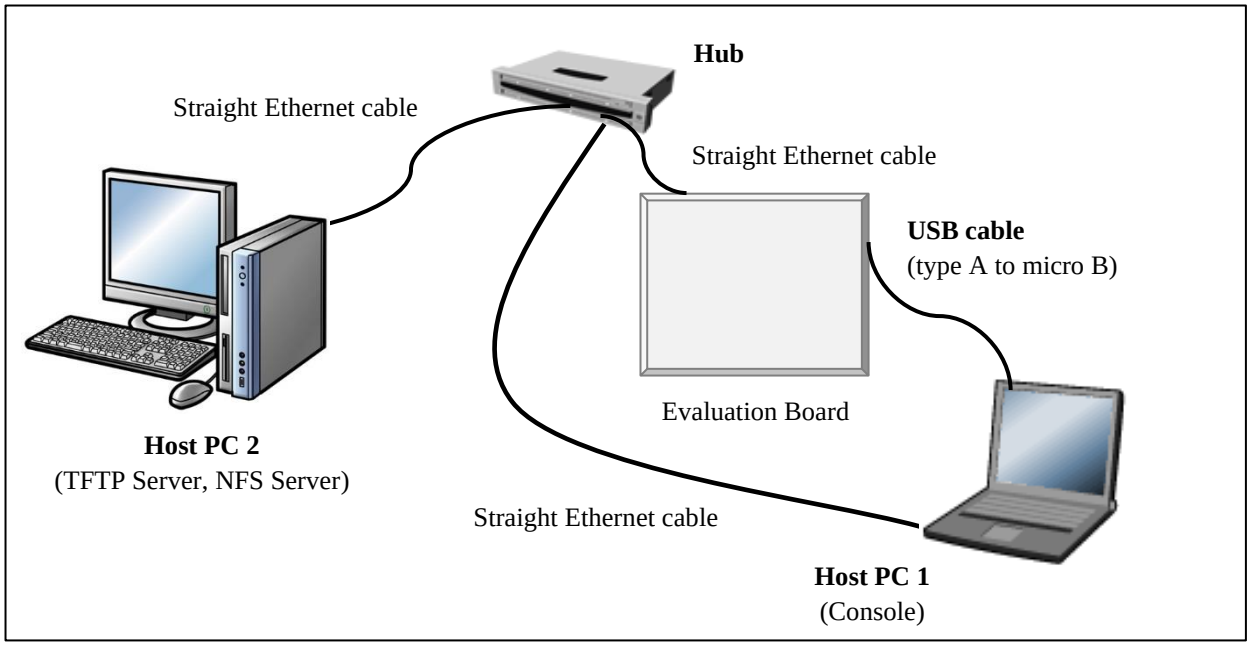


Figure 2.1 Recommended Environment

2.2 Module Configuration

This section explains software relationship and configurations.

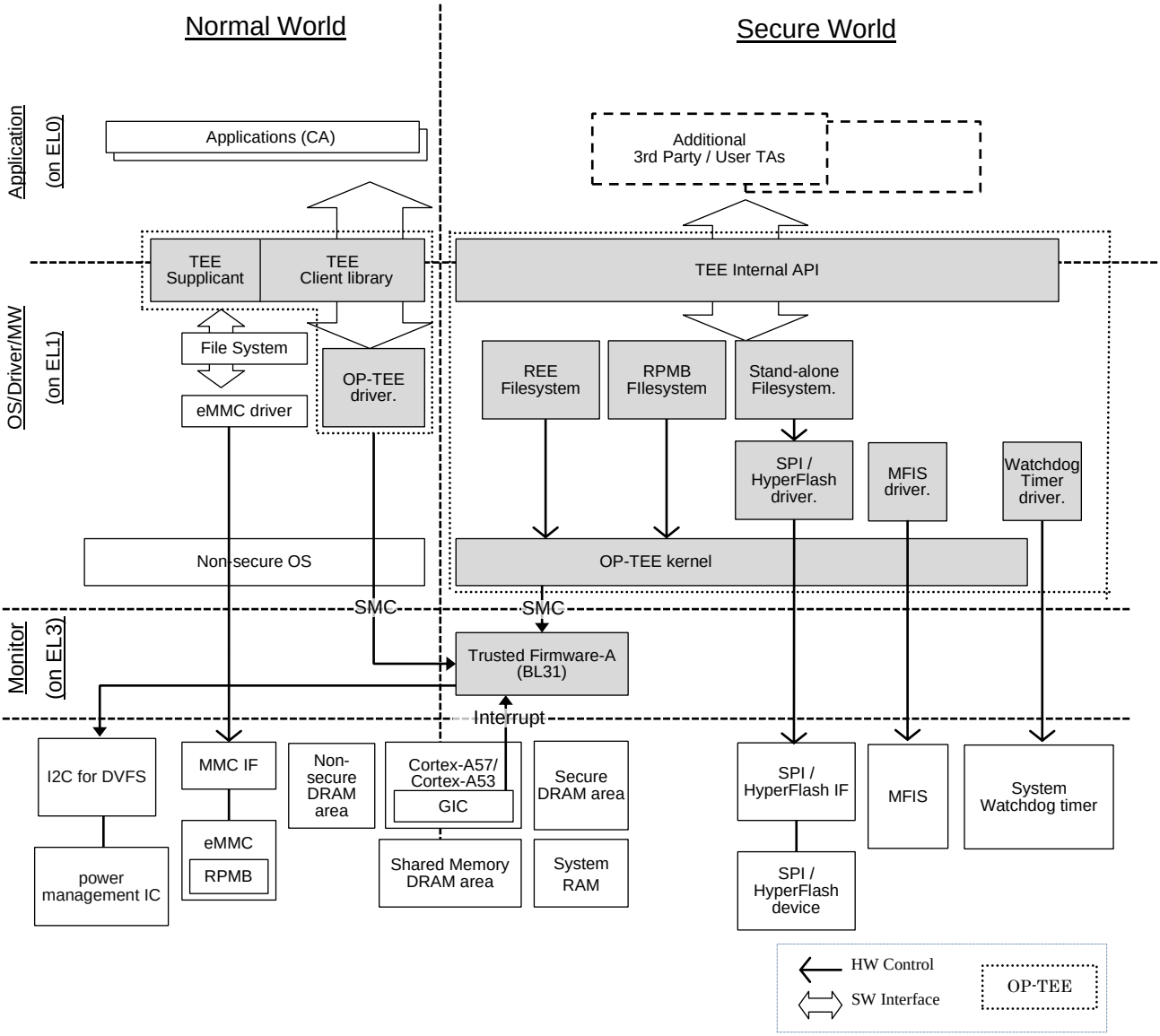


Figure 2.2 Software relationship

2.2.1 Trusted Firmware-A (BL31)

The BL31 is a software for Armv8-A platforms.

Hardware platform: rcar
Compiler: yocto 64bit cross-compiler toolchain

2.2.2 OP-TEE OS

The OP-TEE OS is a software for both Armv7-A and Armv8-A platforms. In the case of R-Car H3/M3/M3N/E3/D3, OP-TEE OS is built for 64bit Architecture of Armv8-A. Therefore, sets a build option “CFG_ARM64_core=y”, and 64bit toolchain is used.

Hardware platform: rcar
Cross-compiler selection: yocto 64bit cross-compiler toolchain

Note) TAs can execute 64bit architecture. Please refer to 4.4.3.5 How to build Dynamic TA.

2.2.3 OP-TEE Driver

OP-TEE Driver is built with Non-secure OS.

- **Linux**
  Cross-compiler selection: yocto 64bit cross-compiler toolchain

2.2.4 OP-TEE Client

OP-TEE Driver is built with Non-secure OS.

- **Linux**
  In R-Car H3/M3/M3N/E3/D3, TEE client library is made for 64bit CAs.

  Cross-compiler selection: yocto 64bit cross-compiler toolchain

  Note) An execution environment of CAs follow the environment of Non-secure OS. Please refer to 1.3.1 Related Document No.1.

2.3 State Transition Diagram

There is no state transition diagram for these modules.

2.4 DMA-resources

There is no DMA-resource for these modules.

3. External Interface

The following is components included in this package.

3.1 Device Node

The following table shows the device node of this module.

- **Linux**

Table 3.1 Device node

Device node	Major number	Minor number
/dev/tee0	Major number of /dev/teeX is dynamically assigned by alloc_chrdev_region function.	Minor number can be assigned from 0 to 15.

3.2 External Interface

3.2.1 Trusted Firmware-A (BL31)

3.2.1.1 PSCI

A Normal World software can access the BL31 via the SMC instruction. A Software executing in the Non-secure OS will request power management (for example, secondary CPU boot, hotplug and idle) by the SMC.

Details of SMC for power management are described in “1.3.1 Related Document No.4” Power State Coordination Interface (PSCI).

Table 3.2 PSCI

PSCI v1.1 API	R-Car H3/M3/M3N/E3 Supported	R-Car D3 Supported	Summary
PSCI_VERSION	Yes	Yes	To ascertain the current version of the interface.
CPU_SUSPEND	Yes	No	To move a topology node into a low-power state. The deepest low-power state is only supported via SYSTEM_SUSPEND.
CPU_OFF	Yes	No	For dynamic removal of the calling core from the system. CPU0 cannot be turned off.
CPU_ON	Yes	No	For dynamic addition of cores, for use in secondary boot, hotplug, or big.LITTLE migration.
AFFINITY_INFO	Yes	Yes	Enable the caller to request status of an affinity instance
MIGRATE (Optional)	No	No	The function should be used to move the OP-TEE OS to another core, and thus enable the original core to call CPU_OFF to power down.
MIGRATE_INFO_TYPE (Optional)	Yes	Yes	This function allows a caller to identify the level of multicore support present in the OP-TEE OS. Return always 1. This change is due to the following reasons. - CPU0 cannot be turned off
MIGRATE_INFO_UP_CPU (Optional)	Yes	Yes	For a uniprocessor OP-TEE OS, this function returns the current resident core. Return always the MPIDR of CPU0. This change is due to the following reasons. - CPU0 cannot be turned off
SYSTEM_OFF	Yes	Yes	Shutdown the system. When issuing the SYSTEM_OFF command, it is necessary to turn off the CPU other than CPU0.

PSCI v1.1 API	R-Car H3/M3/M3N/E3 Supported	R-Car D3 Supported	Summary
			This function shuts down all the power of CPUs and L2 in cluster and enters L2 shutdown mode.
SYSTEM_RESET	Yes	Yes	Reset the system. An evaluation board is restarted and master boot processor starts cold boot, however this control relies on a power management IC on the evaluation board. (supported only Table 2.1 Evaluation Board)
PSCI_FEATURES	Yes	Yes	Query API that allows discovering whether a specific PSCI function is implemented and its features.
CPU_FREEZE (Optional)	No	No	Places the core into an IMPLEMENTATION DEFINED low-power state. Unlike CPU_OFF it is still valid for interrupts to be targeted to the core. However, the core must remain in the low-power state until it a CPU_ON command is issued for it.
CPU_DEFAULT_SUSPEND (Optional)	No	No	Will place a core into an IMPLEMENTATION DEFINED low-power state. Unlike CPU_SUSPEND the caller need not specify a power state parameter.
NODE_HW_STATE (Optional)	No	No	This API is intended to return the true HW state of a node in the power domain topology of the system.
SYSTEM_SUSPEND (Optional)	Yes	No	Used to implement Suspend to RAM. The semantics are equivalent to a CPU_SUSPEND to the deepest low-power state. This control relies on DDR on the R-car H3/M3/M3N/E3. (supported only Table 2.1 Evaluation Board)
PSCI_SET_SUSPEND_MODE (Optional)	No	No	This API allows setting the mode used by CPU_SUSPEND to coordinate power states.
PSCI_STAT_RESIDENCY (Optional)	No	No	Returns the amount of time the platform has spent in the given power state since cold boot.
PSCI_STAT_COUNT (Optional)	No	No	Return the number of times the platform has used the given power state since cold boot.
SYSTEM_RESET2 (Optional)	No	No	This function extends SYSTEM_RESET. It provides: - Architectural reset definitions. - Support for vendor-specific resets.
MEM_PROTECT (Optional)	No	No	This function provides protection against cold reboot attacks, by ensuring that memory is overwritten before it is handed over to an operating system loader.

PSCI v1.1 API	R-Car H3/M3/M3N/E3 Supported	R-Car D3 Supported	Summary
MEM_PROTECT_CHECK_RANGE (Optional)	No	No	This function can be used to check whether a memory range is protected by MEM_PROTECT.

3.2.2 OP-TEE OS

3.2.2.1 TEE Client API

See “1.3.1Related Document No.2” for details on the specification for TEE Client API. All APIs are declared in $WORK/build_<Building case>/tmp/work/salvator_x-poky-linux/optee-client/<*Properties of yocto environment*>/git/out/export/include/tee_client_api.h. In R-Car H3/M3/M3N/E3/D3, TEE Client API specification is according to original software implementations because there are no changes from the original source code of “1.3.2 Related Site for original software No.4”. See table as below for details.

Table 3.3 TEE Client API

Interface	Available	Changes from original	Tested by	Restrictions
TEEC_InitializeContext	Available	No	optee_test	
TEEC_FinalizeContext	Available	No	optee_test	
TEEC_RegisterSharedMemory	Available	No	optee_test	
TEEC_AllocateSharedMemory	Available	No	optee_test	
TEEC_ReleaseSharedMemory	Available	No	optee_test	
TEEC_OpenSession	Available	No	optee_test	
TEEC_CloseSession	Available	No	optee_test	
TEEC_InvokeCommand	Available	No	optee_test	
TEEC_RequestCancellation	N/A			

3.2.2.2 TEE Internal API

See "1.3.1 Related Document No.3" for details on the specification for TEE Internal API. All APIs are declared in $WORK/build_<Building case>/tmp/work/salvator_x-poky-linux/optee-os/<*Properties of yocto environment*>/git/lib/libutee/include/tee_api.h. In R-Car H3/M3/M3N/E3/D3, TEE Internal API specification is according to original software implementations because there are no changes from the original source code of "1.3.2 Related Site for original software No.2". See table as below for details.

Table 3.4 TEE Internal API(Trusted Core Framework API)

Interface	Available	Changes from original	Tested by	Restrictions
TEE_GetPropertyAsString	Available	No	optee_test	
TEE_GetPropertyAsBool	Available	No	optee_test	
TEE_GetPropertyAsU32	Available	No	optee_test	
TEE_GetPropertyAsBinaryBlock	Available	No	optee_test	
TEE_GetPropertyAsUUID	Available	No	optee_test	
TEE_GetPropertyAsIdentity	Available	No	optee_test	
TEE_AllocatePropertyEnumerator	Available	No	optee_test	
TEE_FreePropertyEnumerator	Available	No	optee_test	
TEE_StartPropertyEnumerator	Available	No	optee_test	
TEE_ResetPropertyEnumerator	Available	No	optee_test	
TEE_GetPropertyName	Available	No	optee_test	
TEE_GetNextProperty	Available	No	optee_test	
TEE_Panic	Available	No	optee_test	
TEE_OpenTASession	Available	Yes	optee_test	
TEE_CloseTASession	Available	No	optee_test	
TEE_InvokeTACommand	Available	Yes	optee_test	
TEE_GetCancellationFlag	N/A			
TEE_UnmaskCancellation	N/A			
TEE_MaskCancellation	N/A			
TEE_CheckMemoryAccessRights	Available	No	optee_test	
TEE_SetInstanceData	N/A			
TEE_GetInstanceData	N/A			
TEE_Malloc	Available	No	optee_test	
TEE_Realloc	N/A			
TEE_Free	Available	No	optee_test	
TEE_MemMove	Available	No	optee_test	
TEE_MemCompare	Available	No	optee_test	
TEE_MemFill	N/A			

Table 3.5 TEE Internal API(Trusted Storage API for Data and Keys)

Interface	Available	Changes from original	Tested by	Restrictions
TEE_GetObjectInfo1	Available	No	optee_test	
TEE_RestrictObjectUsage1	N/A			
TEE_GetObjectBufferAttribute	Available	No	optee_test	
TEE_GetObjectValueAttribute	Available	No	optee_test	
TEE_CloseObject	Available	No	optee_test	
TEE_AllocateTransientObject	Available	No	optee_test	
TEE_FreeTransientObject	Available	No	optee_test	
TEE_ResetTransientObject	Available	No	optee_test	
TEE_PopulateTransientObject	Available	No	optee_test	
TEE_InitRefAttribute	Available	No	renesas test	
TEE_InitValueAttribute	Available	No	renesas test	
TEE_CopyObjectAttributes1	Available	No	optee_test	
TEE_GenerateKey	Available	No	optee_test	
TEE_OpenPersistentObject	Available	Yes	optee_test	
TEE_CreatePersistentObject	Available	Yes	optee_test	
TEE_CloseAndDeletePersistentObject1	Available	Yes	optee_test	
TEE_RenamePersistentObject	Available	Yes	optee_test	
TEE_AllocatePersistentObjectEnumerator	Available	Yes	optee_test	
TEE_FreePersistentObjectEnumerator	Available	Yes	optee_test	
TEE_ResetPersistentObjectEnumerator	Available	Yes	optee_test	
TEE_StartPersistentObjectEnumerator	Available	Yes	optee_test	
TEE_GetNextPersistentObject	Available	Yes	optee_test	
TEE_ReadObjectData	Available	Yes	optee_test	
TEE_WriteObjectData	Available	Yes	optee_test	
TEE_TruncateObjectData	Available	Yes	optee_test	
TEE_SeekObjectData	Available	Yes	optee_test	

Table 3.6 TEE Internal API(Cryptographic Operations API)

Interface	Available	Changes from original	Tested by	Restrictions
TEE_AllocateOperation	Available	Yes	optee_test	
TEE_FreeOperation	Available	No	optee_test	
TEE_GetOperationInfo	Available	No	optee_test	
TEE_GetOperationInfoMultiple	N/A			
TEE_ResetOperation	Available	No	optee_test	
TEE_SetOperationKey	Available	No	optee_test	
TEE_SetOperationKey2	Available	No	optee_test	
TEE_CopyOperation	Available	No	optee_test	
TEE_DigestUpdate	Available	No	optee_test	
TEE_DigestDoFinal	Available	No	optee_test	
TEE_CipherInit	Available	No	optee_test	
TEE_CipherUpdate	Available	No	optee_test	
TEE_CipherDoFinal	Available	Yes	optee_test	
TEE_MACInit	Available	No	optee_test	
TEE_MACUpdate	Available	No	optee_test	
TEE_MACComputeFinal	Available	No	optee_test	
TEE_MACCompareFinal	Available	No	optee_test	
TEE_AEInit	Available	No	optee_test	
TEE_AEUpdateAAD	Available	No	optee_test	
TEE_AEUpdate	Available	No	optee_test	
TEE_AEEncryptFinal	Available	No	optee_test	
TEE_AEDecryptFinal	Available	No	optee_test	
TEE_AsymmetricEncrypt	Available	No	optee_test	
TEE_AsymmetricDecrypt	Available	No	optee_test	
TEE_AsymmetricSignDigest	Available	No	optee_test	
TEE_AsymmetricVerifyDigest	Available	No	optee_test	
TEE_DeriveKey	Available	No	optee_test	
TEE_GenerateRandom	Available	No	optee_test	

Table 3.7 TEE Internal API(Time API)

Interface	Available	Changes from original	Tested by	Restrictions
TEE_GetSystemTime	Available	No	optee_test	
TEE_Wait	Available	No	optee_test	
TEE_GetTAPersistentTime	Available	No	optee_test	
TEE_SetTAPersistentTime	Available	No	optee_test	
TEE_GetREETime	Available	No	optee_test	

Table 3.8 TEE Internal API(TEE Arithmetical API)

Interface	Available	Changes from original	Tested by	Restrictions
TEE_BigIntInit	Available	No	optee_test	
TEE_BigIntInitFMMContext	N/A			
TEE_BigIntInitFMM	N/A			
TEE_BigIntConvertFromOctetString	Available	No	optee_test	
TEE_BigIntConvertToOctetString	Available	No	optee_test	
TEE_BigIntConvertFromS32	Available	No	optee_test	
TEE_BigIntConvertToS32	Available	No	optee_test	
TEE_BigIntCmp	Available	No	optee_test	
TEE_BigIntCmpS32	Available	No	optee_test	
TEE_BigIntShiftRight	N/A			
TEE_BigIntGetBit	N/A			
TEE_BigIntGetBitCount	N/A			
TEE_BigIntAdd	Available	No	optee_test	
TEE_BigIntSub	Available	No	optee_test	
TEE_BigIntNeg	Available	No	optee_test	
TEE_BigIntMul	Available	No	optee_test	
TEE_BigIntSquare	N/A			
TEE_BigIntDiv	Available	No	optee_test	
TEE_BigIntMod	Available	No	optee_test	
TEE_BigIntAddMod	Available	No	optee_test	
TEE_BigIntSubMod	Available	No	optee_test	
TEE_BigIntMulMod	Available	No	optee_test	
TEE_BigIntSquareMod	N/A			
TEE_BigIntInvMod	Available	No	optee_test	
TEE_BigIntRelativePrime	N/A			

Interface	Available	Changes from original	Tested by	Restrictions
TEE_BigIntComputeExtendedGcd	N/A			
TEE_BigIntIsProbablePrime	Available	No	optee_test	
TEE_BigIntConvertToFMM	N/A			
TEE_BigIntConvertFromFMM	N/A			
TEE_BigIntComputeFMM	N/A			

See “1.3.1 Related Document No.13-15” for details on the specification for TEE iSocket API. TEE iSocket API are declared in $OPTEEINC/tee_isocket.h. In R-Car H3/M3/M3N/E3/D3, TEE iSocket API specification is according to original software implementations because there are no changes from the original source code of “1.3.2 Related Site for original software No.2”. See table as below for details.

Note: $OPTEEINC=$WORK/build_<Building case>/tmp/work/salvator_x-poky-linux/optee-os/<*Properties of yocto environment*>/git/lib/libutee/include

TEE_iSocket Instance Variable for TCP are declared in $OPTEEINC/tee_tcpsocket.h.

Table 3.9 TEE iSocket API (TEE_tcpSocket)

Interface	Available	Changes from original	Tested by	Restrictions
open	Available	No	optee_test	
close	Available	No	optee_test	
send	Available	No	optee_test	
recv	Available	No	optee_test	
error	Available	No	optee_test	
ioctl	Available	No	optee_test	

TEE_iSocket Instance Variable for UDP are declared in $OPTEEINC/tee_udpsocket.h.

Table 3.10 TEE iSocket API (TEE_udpSocket)

Interface	Available	Changes from original	Tested by	Restrictions
open	Available	No	optee_test	
close	Available	No	optee_test	
send	Available	No	optee_test	
recv	Available	No	optee_test	
error	Available	No	optee_test	
ioctl	Available	No	optee_test	

3.2.2.3 System Watchdog Timer

The System Watchdog timer (SWDT) is a single-channel timer that uses the OSCCLK as an input clock and can be used as a watchdog timer. OSCCLK is a clock which is generated by CPG.
The SWDT driver automatically stops if it is already started at the time of suspend and automatically start at the time of resuming, so no re-operation is necessary.
However, when resuming, the timer starts from 1.
Note) The SWDT cannot be execute in the debug mode(MD21bit is 1) because soft power-on reset cannot be issued in debug mode.

The following table shows the APIs list of SWDT. SWDT APIs are executable from pseudo TA and OP-TEE kernel, but there are not executable from Dynamic TA.
Note) The swdt_kick and swdt_stop can be execute from a different function than the function that performed the swdt_start.

Table 3.11 APIs list of SWDT

No	I/F	Description
1	swdt_start	Start of SWDT monitoring
2	swdt_kick	Response to SWDT monitoring
3	swdt_stop	Stop of SWDT monitoring

(1) swdt_start

Function				
swdt_start				
Parameters				
Type	Value	I/O	Range	Description
uint16_t	count	I	1 - 65535	Number of the counts.
uint8_t	clk	I	0 - 7	Clock for the monitoring. Select the following definitions. SWDT_FREQ_OSC_DIV_1_1 SWDT_FREQ_OSC_DIV_1_4 SWDT_FREQ_OSC_DIV_1_16 SWDT_FREQ_OSC_DIV_1_32 SWDT_FREQ_OSC_DIV_1_64 SWDT_FREQ_OSC_DIV_1_128 SWDT_FREQ_OSC_DIV_1_1024 SWDT_FREQ_EXPANDED
uint8_t	expanded_clk	I	0 - 255	The clock of the expansion mode. Lower 1-6bit (expanded_clk[5:0]) only use.
void	(*cb)(void)	I	NULL : power-on reset non-NULL : callback function	The processing executed when SWDT detected abnormality.
Return				
Type	Value	Description		
int32_t	SWDT_SUCCESS	Success		
	SWDT_ERR_PARAMETER	Parameter error		
	SWDT_ERR_SEQUENCE	Sequence error		
Description				
This function starts SWDT monitoring by the setting of the argument. When the setting of the argument includes a wrong value, this function returns SWDT_ERR_PARAMETER. When this function has already been called, this function returns SWDT_ERR_SEQUENCE.				

- About the setting of the SWDT monitoring interval

The monitoring interval is given by the following equation.

"monitoring interval"[s] = (1 / "Clock for the monitoring"[Hz]) × "Number of the counts"

"Clock for the monitoring" and "Number of the counts" can be set by the argument "clk" and "count".

< As an example >

If "clk" = SWDT_FREQ_OSC_DIV_1_4, "count" = 256, and OSCCLK = 130.20[kHz], the monitoring interval is given by the following equation.

"Clock for the monitoring"[Hz] = Rφ/ 4 = OSCCLK / 4 = 130.20 × 1000 / 4 = 32550[Hz]

"monitoring interval"[s] = (1 / "Clock for the monitoring"[Hz]) × "Number of the counts"

= (1 / 32550 [Hz]) × 256

= 0.00786[s] = 7.86[ms]

- About the monitoring using the clock of the expansion mode

When "count" is set to SWDT_FREQ_EXPANDED, it can be set for the longer monitoring interval than normal by using the clock of the expanded mode.

The monitoring interval of the expanded mode is decided by a combination of lower 1-4bit(expanded_clk[3:0]) and 5-6bit(expanded_clk[5:4]) of "expanded_clk". For details, refer to the description of the CKS1[5:0] bit of "78.2.3 System Watchdog Timer Control/Status Register B (SWTCSRB)" of R-Car Series, 3rd Generation Hardware User' s Manual.

- About the processing that is executed when SWDT detected abnormality

When swdt_kick() is not called within the monitoring interval, it is judged that SWDT detected abnormality. Then SWDT executes either one of following process.

(1) Execute a resetting operation

(2) Call the callback function that is registered

If "cb" is set to "NULL", SWDT executes prosess (1).

And if "cb" is set to a function pointer of the callback function, SWDT executes prosess (2). In this case, note that the kernel API of OP-TEE cannot be called from the callback function. This is because the kernel API of OP-TEE is not preemptive.

Note) Even in the case of (2), execute a power-on reset or software reset to initialize the System Watchdog.
When not initializing and using System Watchdog, behavior isn't guaranteed.

(2) swdt_kick

Function				
swdt_kick				
Parameters				
Type	Value	I/O	Range	Description
void	-	-	-	-
Return				
Type	Value		Description	
int32_t	SWDT_SUCCESS		Success	
	SWDT_ERR_SEQUENCE		Sequence error	
Description				
This function responds to SWDT monitoring. When this function is not called within the monitoring interval, it is judged that SWDT detected abnormality. When this function is called, the timer counter of the monitoring is reset and performs the monitoring in the monitoring interval again. When swdt_start() has not been called, this function returns SWDT_ERR_SEQUENCE.				

(3) swdt_stop

Function				
swdt_stop				
Parameters				
Type	Value	I/O	Range	Description
void	-	-	-	-
Return				
Type	Value		Description	
int32_t	SWDT_SUCCESS		Success	
	SWDT_ERR_SEQUENCE		Sequence error	
Description				
This function stops the SWDT monitoring. When swdt_start() has not been called, this function returns SWDT_ERR_SEQUENCE.				

3.2.2.4 MFIS Driver

The multifunctional interface (MFIS) driver is a driver for detecting a security error at the interrupt handler. MFIS driver has the interrupt handling in accordance with the 3rd generation R-Car series products.
The MFIS driver automatically stops if it is already started at the time of suspend and automatically start at the time of resuming, so no re-operation is necessary.

The following table shows the APIs list of MFIS Driver. MFIS Driver APIs are executable from pseudo TA and OP-TEE kernel, but there are not executable from Dynamic TA.
Note) The mfis_error_detection_stop can be execute from a different function than the function that performed the mfis_error_detection_start.

Table 3.12 APIs list of MFIS Driver

No	I/F	Description
1	mfis_error_detection_start	Start of a security error detection
2	mfis_error_detection_stop	Stop of a security error detection

(1) mfis_error_detection_start

Function				
mfis_error_detection_start				
Parameters				
Type	**Value**	**I/O**	**Range**	**Description**
MFIS_ERR_SETTING_T	*err	I	non-NULL	Define the following. control : Error detection enable. 1'b1 : Enable 1'b0 : Disable target : Error detection target. 1'b1 : To INTC (Interrupt Controller) 1'b0 : To GPIO (External Terminal)
void	(*cb)(MFIS_ERR_FACTOR_T*)	I	non-NULL	To callback the user function.

Return		
Type	**Value**	**Description**
int32_t	MFIS_SUICCESS	Success
	MFIS_ERR_PARAMETER	Parameter error
	MFIS_ERR_SEQUENCE	Sequence error

Description

This function enables the interrupt processing to the enabled register of error checking function or enabled register of INTC.
The security error registers that can be set to the structure are MFIERRxxxR5 and MFIERRxxxR6.
Also starts module start.

When the setting of the argument includes a wrong value, this function returns MFIS_ERR_PARAMETER.
Also, this function returns MFIS_ERR_PARAMETER if the setting of all controls of MFIS_ERR_SETTING_T is Disable (all zeros).

When this function has already been called, this function returns MFIS_ERR_SEQUENCE.

- About clearing of error cause

Clear of the register is implemented in the driver, but the error cause is cleared by lowering the error bit of the target register in the callback function.

(2) mfis_error_detection_stop

Function				
mfis_error_detection_stop				
Parameters				
Type	Value	I/O	Range	Description
void	-	-	-	-
Return				
Type	Value		Description	
int32_t	MFIS_SUCCESS		Success	
	MFIS_ERR_SEQUENCE		Sequence error	
Description				
This function disables the interrupt processing to the enabled register of error checking function or enabled register of INTC. Also starts module stop. When mfis_error_detection_start() has not been called, this function returns MFIS_ERR_SEQUENCE.				

3.2.3 OP-TEE Driver

3.2.3.1 TEE Client API for the kernel mode

All APIs are declared in $WORK/build_<Building case>/tmp/work-shared/salvator-x/kernel-source/include/linux/tee_drv.h. In R-Car H3/M3/M3N/E3/D3, Table 3.13 shows all TEE Client API for the kernel mode and the specifications are shown in (1) to (8) in this chapter.

LEGEND ()

Tested by :
optee_test : Confirmed the “OK” to test using the optee_test. (See “1.3.2 Related Site for original software No.5”).
renesas test : Confirmed the “OK” to test using the Renesas original Test program because test case is not included in the optee_test.

Table 3.13 TEE Client API for kernel mode

Interface	Available	Tested by	Note
tee_client_open_context	Available	renesas test	Provide the same function as TEEC_InitializeContext.
tee_client_close_context	Available	renesas test	Provide the same function as TEEC_FinalizeContext.
tee_shm_reg	Available	renesas test	Provide the same function as TEEC_RegisterSharedMemory.
tee_shm_alloc	Available	renesas test	Provide the same function as TEEC_AllocateSharedMemory.
tee_shm_free	Available	renesas test	Provide the same function as TEEC_ReleaseSharedMemory.
tee_client_open_session	Available	renesas test	Provide the same function as TEEC_OpenSession.
tee_client_close_session	Available	renesas test	Provide the same function as TEEC_CloseSession.
tee_client_invoke_func	Available	renesas test	Provide the same function as TEEC_InvokeCommand.

(1) tee_client_open_context

Function				
tee_client_open_context				
Parameters				
Type	Value	I/O	Range	Description
struct tee_context	*start	in/out	address	If this parameter is NULL, tee_client_open_context allocate and initialize a context for TEE Client API, otherwise initialize a context.
int (*match)(struct tee_ioctl_version_data *, const void *)	-	in	non-NULL	This parameter sets a function to check TEE device. *1 If the confirmation result is satisfactory, this function returns non-zero, otherwise return zero. The confirmation method is user dependent.
const void	*data	in	-	This data is used as the second argument of “int (*match)(struct tee_ioctl_version_data *, const void *)”.
struct tee_ioctl_version_data	*vers	in	-	This data is used as the first argument of “int (*match)(struct tee_ioctl_version_data *, const void *)”.

Return		
Type	Value	Description
struct tee_context *	Available addresses	Success
	ERR_PTR	Failed

Description
This function opens a context for TEE Client API of kernel mode. *1 TEE device ID defined in $WORK/build_<Building case>/tmp/work-shared/salvator-x/kernel-source/include/uapi/linux/tee.h as TEE_IMPL_ID_OPTEE, TEE_OPTEE_CAP_TZ and TEE_GEN_CAP_GP. Theses value also is obtained by calling optee_get_version.

(2) tee_client_close_context

Function				
tee_client_close_context				
Parameters				
Type	Value	I/O	Range	Description
struct tee_context	*ctx	in	non-NULL	A context of TEE Client API to close.
Return				
Type	Value	Description		
void	-	-		
Description				
This function closes a context for TEE Client API of kernel mode.				

(3) tee_shm_reg

Function				
tee_shm_reg				
Parameters				
Type	Value	I/O	Range	Description
struct tee_context	*ctx	in	non-NULL	A context that allocates the shared memory.
size_t	size	in	0~2^64	The requested shared memory size in bytes.
u32	flags	in	TEE_SHM_MAPPED TEE_SHM_DMA_BUF	Flags setting properties for the requested shared memory. *1

Return		
Type	Value	Description
struct tee_context *	Available addresses	Success
	ERR_PTR	Failed to allocate shared memory.

Description
This function redefines tee_shm_alloc as a function on kernel space corresponding to TEEC_SharedMemory of TEE Client API. Therefore, this function provides the same function as tee_shm_alloc. *1 TEE_SHM_MAPPED must currently always be set for allocating memory. If TEE_SHM_DMA_BUF is also set, the allocated memory is acquired from the shared memory, else the driver uses private memory pool.

(4) tee_shm_alloc

Function				
tee_shm_alloc				
Parameters				
Type	Value	I/O	Range	Description
struct tee_context	*ctx	in	non-NULL	A context that allocates the shared memory.
size_t	size	in	0~2^64	The requested shared memory size in bytes.
u32	flags	in	TEE_SHM_MAPPED, TEE_SHM_DMA_BUF	Flags setting properties for the requested shared memory. *1

Return		
Type	Value	Description
struct tee_context *	Available addresses	Success
	ERR_PTR	Failed to allocate shared memory.

Description
This function allocates the shared memory used in a context for TEE Client API of kernel mode. *1 TEE_SHM_MAPPED must currently always be set for allocating memory. If TEE_SHM_DMA_BUF is also set, the allocated memory is acquired from the shared memory, else the driver uses private memory pool.

(5) tee_shm_free

Function				
tee_shm_free				
Parameters				
Type	Value	I/O	Range	Description
struct tee_shm	*shm	in	non-NULL	A shared memory context that free the shared memory.
Return				
Type	Value			Description
void	-			-
Description				
This function frees a shared memory used in a context for TEE Client API of kernel mode.				

(6) tee_client_open_session

Function				
tee_client_open_session				
Parameters				
Type	Value	I/O	Range	Description
struct tee_context	*ctx	in	non-NULL	A context that opens a session to TA.
struct tee_ioctl_open_session_arg	*arg	in	non-NULL	Open session arguments. *1 Before setting parameters, initialize this structure with zeros.
struct tee_param	*param	in	non-NULL	Set the start address of the array storing data to be passed from the user, and the number of array of this data depends on (*arg)->num_params. Before setting parameters, initialize this structure with zeros.

Return		
Type	Value	Description
int	0	Success
	others	Failed to open a session.

Description
This function opens a session to a TA. *1 The setting of the following members must be done before calling this function. (*arg)->uuid: Setting the UUID of TA to communicate. (*arg)->num_params: Setting number of input data using tee_param structure from users.

(7) tee_client_close_session

Function				
tee_client_close_session				
Parameters				
Type	Value	I/O	Range	Description
struct tee_context	*ctx	in	non-NULL	A context of TEE Client API to close.
u32	session	in	non-NULL	Set the start address of the array storing data to be passed from the user, and the number of array of this data depends on (*arg)->num_params. Before setting parameters, initialize this structure with zeros.

Return		
Type	Value	Description
int	0	Success
	others	Failed to close a session.

Description
This function closes a session for TEE Client API of kernel mode.

(8) tee_client_invoke_func

Function				
tee_client_invoke_func				
Parameters				
Type	Value	I/O	Range	Description
struct tee_context	*ctx	in	non-NULL	A context that opens a session to TA.
struct tee_ioctl_version_data	*arg	in	non-NULL	Invoke arguments. *1 Before setting parameters, initialize this structure with zeros.
struct tee_param	*param	in	non-NULL	Parameters passed to the Trusted Application *2

Return		
Type	Value	Description
int	0	Success
	others	Failed to invoke a function in a TA.

Description

This function invokes a function in a TA.

*1 The setting of the following members must be done before calling this function.
(*arg)->func: Setting an identifier that is used to indicate which of the exposed TA should be invoked.
(*arg)->session: Setting the session which is a member of struct tee_ioctl_open_session_arg after inputting to tee_client_open_session.
(*arg)->num_params: Setting number of input data using tee_param structure from users.

*2 The setting of the following members must be done before calling this function.
(*param)->attr: The type of each parameter can take one of the values, which are defined as TEE_IOCTL_PARAM_ATTR_TYPE_* in $WORK/build_<Building case>/tmp/work-shared/salvator-x/kernel-source/include/uapi/linux/tee.h.
(*param)->u: If you pass a shared memory sets the address and the size of the buffer to memref, and if you want to pass a single value to set its value to value.

3.3 Definitions

This section shows the definitions that these packages are defined.

3.3.1 Trusted Firmware-A (BL31)

Definitions of the logging buffer's address and size are below.

```
#define   RCAR_BL31_LOG_BASE            0x44040000
#define   RCAR_BL31_LOG_SIZE            0x00014000
```

Definitions of the system ram address and size using System Shutdown/Reset/Suspend are bellow.

```
#define   DEVICE_SRAM_BASE              0xE6310000
#define   DEVICE_SRAM_SIZE              0x00002000
#define   DEVICE_SRAM_SHADOW_BASE  (DEVICE_SRAM_BASE+DEVICE_SRAM_SIZE)
#define   DEVICE_SRAM_STACK_BASE     \
                          (DEVICE_SRAM_SHADOW_BASE+DEVICE_SRAM_SIZE)
```

Definition of the stack area size for each core is as follows.

```
  :
#elif IMAGE_BL31
#define PLATFORM_STACK_SIZE           U(0x800)
  :
```

3.3.2 OP-TEE OS

Definitions of the logging buffer's address and size are below.

```
#define OPTEE_LOG_BASE                0x46400000
#define LOG_RAM_MAX_SIZE              0x00014000

#define OPTEE_LOG_NS_BASE             0x47FEC000
#define LOG_NS_CPU_AREA_SIZE          1024

#define TEE_RPC_DEBUG_LOG             0x3F000000
```

3.3.2.1 TEE Client API

See section 4.4 in "1.3.1 Related Document No.2" for details on the specifications for TEE Client API. All definitions are declared in $WORK/build_<Building case>/tmp/work/salvator_x-poky-linux/optee-client/<*Properties of yocto environment*>/git/out/export/include/tee_client_api.h.

3.3.2.1 TEE Internal API

See section 3 in “1.3.1 Related Document No.3” for details on the specifications for TEE Internal API. All definitions are declared in $WORK/build_<Building case>/tmp/work/salvator_x-poky-linux/optee-os/<*Properties of yocto environment*>/git/lib/libutee/include/tee_api_defines.h.

3.3.2.2 System Watchdog Timer

The following table shows the definitions used in SWDT.

Table 3.14 Definitions of SWDT

Model	Value	Description
SWDT_SUCCESS	(0)	Success
SWDT_ERR_PARAMETER	(-1)	Parameter error
SWDT_ERR_SEQUENCE	(-2)	Sequence error
SWDT_FREQ_OSC_DIV_1_1	(0U)	Rφ(OSCCLK)
SWDT_FREQ_OSC_DIV_1_4	(1U)	Rφ(OSCCLK) / 4
SWDT_FREQ_OSC_DIV_1_16	(2U)	Rφ(OSCCLK) / 16
SWDT_FREQ_OSC_DIV_1_32	(3U)	Rφ(OSCCLK) / 32
SWDT_FREQ_OSC_DIV_1_64	(4U)	Rφ(OSCCLK) / 64
SWDT_FREQ_OSC_DIV_1_128	(5U)	Rφ(OSCCLK) / 128
SWDT_FREQ_OSC_DIV_1_1024	(6U)	Rφ(OSCCLK) / 1024
SWDT_FREQ_EXPANDED	(7U)	The clock of the expansion mode

3.3.2.3 MFIS Driver

The following table shows the definitions used in MFIS Driver.

Table 3.15 Definitions of MFIS Driver

Model	Value	Description
MFIS_SUCCESS	(0)	Success
MFIS_ERR_PARAMETER	(-1)	Parameter error
MFIS_ERR_SEQUENCE	(-2)	Sequence error
MFIS_ERR_DET_MAX	(2)	The number of MFIS register

Note)
MFIS_ERR_DET_MAX is the number of registers supported by MFIS Driver.
The security error registers are MFIERRxxxR5 and MFIERRxxxR6.
Therefore, the maximum number of MFIS_ERR_DET_MAX is 2.

3.3.3 OP-TEE Driver

•Linux
Definitions of the logging are shown below.

```
#define TEE_LOG_NS_SIZE          0x00014000
#define TEE_LOG_NS_BASE          0x47FEC000
#define LOG_NS_CPU_AREA_SIZE     1024

#define TEE_RPC_DEBUG_LOG        0x3F000000
```

3.3.3.1 TEE Client API for the kernel mode

All definitions are declared in $WORK/build_<Building case>/tmp/work-shared/salvator-x/kernel-source/include/linux/tee_drv.h and $WORK/build_<Building case>/tmp/work-shared/salvator-x/kernel-source/include/uapi/linux/tee.h

3.4 Structure

This section shows the structure that these packages are defined.

3.4.1 Trusted Firmware-A (BL31)

This structure defines information for Logging.

```
typedef struct log_head
{
        uint8_t   head[4];
        uint32_t  index;
        uint32_t  size;
        uint8_t   res[4];
} loghead_t;
```

This structure defines information for DRAM. This structure is used for RCAR_SIP_SVC_GET_DRAMCONF.

```
typedef struct {
        uint64_t  DramType;
        uint64_t  LegacyAddress;
        uint64_t  LegacySize;
        uint64_t  Address[4];
        uint64_t  Size[4];
} rcar_dram_conf_t;
```

3.4.2 OP-TEE OS

This structure defines information for Logging.

```
typedef struct {
        uint8_t   prefix[4];
        uint32_t  index;
        uint32_t  size;
        uint32_t  reserve;
} log_buf_header_t;
```

3.4.2.1 TEE Client API

See section 4.3 in “1.3.1 Related Document No.2” for details on the specifications for TEE Client API. And also refer to $WORK/build_<Building case>/tmp/work/salvator_x-poky-linux/optee-client/<*Properties of yocto environment*>/git/out/export/include/tee_client_api.h for some Implementation-Defined Behavior that described in “1.3.1 Related Document No.2”. There are no changes in the source code from original software.

3.4.2.2 TEE Internal API

See section 3 in "1.3.1 Related Document No.3" for details on the specifications for TEE Internal API. All structures are declared in $WORK/build_<Building case>/tmp/work/salvator_x-poky-linux/optee-os/<*Properties of yocto environment*>/git/lib/libutee/include/ tee_api_types.h.

3.4.3 OP-TEE Driver

3.4.3.1 TEE Client API for the kernel mode

All structures are declared in $WORK/build_<Building case>/tmp/work-shared/salvator-x/kernel-source/include/linux/tee_drv.h and $WORK/build_<Building case>/tmp/work-shared/salvator-x/kernel-source/include/uapi/linux/tee.h

3.4.3.2 MFIS Driver

The following table shows the structures used in MFIS Driver.

Table 3.16 Structures of MFIS monitoring settings

Structure Name		
MFIS_ERR_SETTING_T		
Description		
Structure for MFIS monitoring settings		
Model	**Value**	**Description**
uint32_t	control[MFIS_ERR_DET_MAX]	MFIS Error Control Register
uint32_t	target[MFIS_ERR_DET_MAX]	MFIS Error Target Register

Note)
Registers to be set to MFIS_ERR_SETTING_T are as follows.
control[0]/target[0]: MFIERRxxxR5
control[1]/target[1]: MFIERRxxxR6
In order to set the bit information as a parameter, please refer to Section 14.2.11 and Section 14.2.13 of the H/W of the manual.

Table 3.17 Structures of MFIS error register

Structure Name		
MFIS_ERR_FACTOR_T		
Description		
Structure for MFIS error status register		
Model	**Value**	**Description**
uint32_t	error[MFIS_ERR_DET_MAX]	MFIS Error Status Register

Note)
Status of the register which detects the error is stored in the elements of the subject.

4. Integration

4.1 Directory Configuration

The directory configuration is shown below.

4.1.1 Trusted Firmware-A (BL31)

The BL31 software environment of this package added directories to the original software directories configuration. At Yocto environment, added directories and the BL31 source code are located on the following path.

[$WORK/build_<Building case>/tmp/work/salvator_x-poky-linux/arm-trusted-firmware/<Properties of yocto environment>/git/]

```
$WORK
|--build_<Building case>
|    |--tmp
|    |    |--work
|    |    |    |--<Board name>-poky-linux
|    |    |    |    |--arm-trusted-firmware
|    |    |    |    |    |--<Properties of yocto environment>
|    |    |    |    |    |    |--git
|    |    |    |    |    |    |    |--drivers
|    |    |    |    |    |    |    |    |--renesas
|    |    |    |    |    |    |    |    |    |--common
|    |    |    |    |    |    |    |    |    |    |--auth
|    |    |    |    |    |    |    |    |    |    |--avs
|    |    |    |    |    |    |    |    |    |    |--console
|    |    |    |    |    |    |    |    |    |    |--ddr
|    |    |    |    |    |    |    |    |    |    |--delay
|    |    |    |    |    |    |    |    |    |    |--dma
|    |    |    |    |    |    |    |    |    |    |--emmc
|    |    |    |    |    |    |    |    |    |    |--iic_dvfs
|    |    |    |    |    |    |    |    |    |    |--io
|    |    |    |    |    |    |    |    |    |    |--pwrc
|    |    |    |    |    |    |    |    |    |    |--rom
|    |    |    |    |    |    |    |    |    |    |--rpc
|    |    |    |    |    |    |    |    |    |    |--scif
|    |    |    |    |    |    |    |    |    |    |--watchdog
|    |    |    |    |    |    |    |    |    |--rcar
|    |    |    |    |    |    |    |    |    |    |--board
|    |    |    |    |    |    |    |    |    |    |--cpld
|    |    |    |    |    |    |    |    |    |    |--pfc
|    |    |    |    |    |    |    |    |    |    |--qos
```

4.1.2 OP-TEE OS

The OP-TEE OS software environment of this package added directories to the original software directories configuration. At Yocto environment, added directories and the OP-TEE OS source code are located on the following path.

[$WORK/build_<Building case>/tmp/work/salvator_x-poky-linux/optee-os/<Properties of yocto environment>/git/]

```
$WORK
|--build_<Building case>
|    |--tmp
|    |    |--work
|    |    |    |--salvator_x-poky-linux
|    |    |    |    |-- optee-os
|    |    |    |    |    |--<Properties of yocto environment>
|    |    |    |    |    |    |--git
|    |    |    |    |    |    |    |--core
|    |    |    |    |    |    |    |    |--arch
|    |    |    |    |    |    |    |    |    |--arm
|    |    |    |    |    |    |    |    |    |    |--plat-rcar
```

4.1.3 OP-TEE Driver

•Linux

At Yocto environment, the OP-TEE Driver source code is located on the following path.

[$WORK/build_<Building case>/tmp/work-shared/salvator-x/kernel-source/]

```
$WORK
|--build_<Building case>
|    |--tmp
|    |    |--work-shared
|    |    |    |--salvator-x
|    |    |    |    |--kernel-source
|    |    |    |    |    |--drivers
|    |    |    |    |    |    |--tee
|    |    |    |    |    |    |    |--optee
|    |    |    |    |    |    |    |    |--Makefile          OP-TEE driver makefile
|    |    |    |    |    |    |    |    |--core.c            OP-TEE driver core source file
|    |    |    |    |    |    |    |    |--optee_rcar.h      OP-TEE driver R-Car header file
|    |    |    |    |    |    |    |    |--rcar.c            OP-TEE driver R-Car source file
|    |    |    |    |    |    |    |    |--rcar_version.h    OP-TEE driver R-Car version header file
|    |    |    |    |    |    |    |    |--rpc.c             OP-TEE driver rpc source file
|    |    |    |    |    |    |    |--tee_shm.c             TEE subsystem shared memory source file
```

4.1.4 OP-TEE Client

•Linux

The OP-TEE Client software environment of this package does not modify the original source code of “Related Site for original software No.4”. At Yocto environment, the OP-TEE Client source code is located on the following path.

[$WORK/build_<Building case>/tmp/work/salvator_x-poky-linux/optee-client/<Properties of yocto environment>/git/]

```
$WORK
|--build_<Building case>
|    |--tmp
|    |    |--work
|    |    |    |--salvator_x-poky-linux
|    |    |    |    |-- optee-client
|    |    |    |    |    |--<Properties of yocto environment>
|    |    |    |    |    |    |--git
```

4.2 Integration Procedure

The BL31, the OP-TEE OS, the OP-TEE Driver and the OP-TEE Client environment are included in Yocto Environment. To build these software images, refer to “1.3.1 Related Document No.1” Linux Interface Specification Yocto recipe Start-Up Guide documentation for the following function: 3. Building Instructions.

4.3 Option Setting

These packages support the following build options.

Options that are not described in this section have not been set to build command, so default value has been assigned to options in building.

4.3.1 Trusted Firmware-A (BL31)

• PLAT

Fixed string set is rcar.
Any string other than above, please do not set.

• RESET_TO_BL31

This value is set 0 internally.

• ENABLE_PLAT_COMPAT

Any value other than 1, please do not set.
If this option is not set, the value is set 1 internally.

• LOG_LEVEL

BL31 provides logging functions ERROR(), NOTICE(), WARN(), INFO() and VERBOSE().
Logging functions value in the following table is valid.
Set value is 0 to 50, 0 is no functions for output logs, 50 is all functions for output logs.

Table 4.1 Association table for the LOG_LEVEL value and valid logging functions.

Log level	Valid logging function
0	No functions output logs
10	ERROR()
20	ERROR(),NOTICE()
30	ERROR(),NOTICE(), WARN()
40	ERROR(),NOTICE(), WARN(), INFO()
50	ERROR(),NOTICE(), WARN(), INFO(), VERBOSE()

• DEBUG

Set value is 0 or 1. 0 is a release build, and 1 is a debug build.
If LOG_LEVEL value hasn't been set, LOG_LEVEL is set as the DEBUG value in the following table.

Table 4.2 Association table the DEBUG value and valid logging functions.

DEBUG	build	LOG_LEVEL	Valid logging functions
0	release	20	ERROR(), NOTICE()
1	debug	40	ERROR(), NOTICE(),WARN(), INFO()

If this option is not set, the value is set 0 internally.

• SPD

Set string is opteed or none.
String opteed means that OP-TEE OS will start.
String none means that OP-TEE OS will not start.
If this option is not set, the string is set "opteed" internally.

• ARM_CCI_PRODUCT_ID

This value is set 500 internally.

• LSI

This option has no effect on the build of Trusted Firmware-A (BL31)
Please refer to 1.3.1 Related Document No.6.

• PMIC_ROHM_BD9571

Set 1, when users use the evaluation board described in the 2.1 Hardware environment.
If this option is not set, the value is set 1 internally.
Note) This option must be set to disabled in the case of R-Car D3.

• RCAR_SYSTEM_SUSPEND

Set 1, when users use the Suspend To RAM.
If this option is not set, the value is set 1 internally.

Note)
When setting this option to 1, users need to set PMIC_ROHM_BD9571 to 1.
In the case of the following options, the board power will not turn off because the PMIC is invalid.
Therefore, build errors will occur in the following options.
PMIC_ROHM_BD9571 : set 0
RCAR_SYSTEM_SUSPEND : set 1
Note) This option must be set to disabled in the case of R-Car D3.

When use Suspend To RAM other than Salvator-X/Salvator-XS/Ebisu/Ebisu-4D, processing equivalent to that implemented in PMIC_ROHM_BD9571 is necessary.

• PMIC_LEVEL_MODE

Set 1, when users use the evaluation board described in the 2.1 Hardware environment.
If this option is not set, the value is set 1 internally.

4.3.2 OP-TEE OS

• PLATFORM

Fixed string set is rcar.
Any string other than above, please do not set.

• CFG_TEE_CORE_LOG_LEVEL

OP-TEE OS provides logging functions MSG() , EMSG(), IMSG(), DMSG() and FMSG().
Logging functions beside log level in the following table are valid.
Set one value in the ranges from 1 to 4.

Table 4.3 OP-TEE OS Log level

Log level	Function
1	MSG(), EMSG ()
2	MSG(), EMSG (), IMSG()
3	MSG(), EMSG (), IMSG(), DMSG()
4	MSG(), EMSG (), IMSG(), DMSG(), FMSG()

If this option is not set, the value is set 1 internally.

• RCAR_DEBUG_LOG

This option is for outputting log via Linux terminal.
Set value is 0 or 1. 0 is invalid, and 1 is valid.
If this option is not set, the value is set 0 internally.
The log of OP-TEE OS before booting Linux doesn't output to the console.

• DEBUG

Set value is 0 or 1.
0 specifies optimization of code size. 1 specifies minimal optimization.
If this option is not set, the value is set 0 internally.

• CFG_ARM64_core

Set value is "y" or "n".
The y specifies to build a 64bit OP-TEE OS and the "n" specifies to build a 32bit one.
Since the 32bit does not support, be sure to specify a 64bit.

• CFG_EARLY_TA

Set value is "y" or "n".
The y specifies to enable early TA.
If this option is not set, the value is set "n" internally.

• EARLY_TA_PATHS

If CFG_EARLY_TA is set in "y", this option can appoint binary file of the TA.
Set value is a file path of ELF of the TA.
If this option is not set, the early TA is not incorporated in OP-TEE OS.
The size of tee.bin is limited to 1MB. If tee.bin exceeds 1MB, a build error occurs.
Please refer to chapter 4.4.1 for the setting example.

• CFG_CRYPTO_WITH_CE

Set value is "y" or "n".
The y specifies to enable Armv8 acceleration of AES, SHA1 and SHA256 algorithms.
If this option is not set, the value is set "y" internally.

• CFG_RCAR_UNSUPPORT_TA_VER_DB

If CFG_RPMB_FS or CFG_STANDALONE_FS is set in "y", this option is set in "y" internally.
The y specifies to disable the function of optee_os/core/tee/tadb.c.

• CFG_MMAP_REGIONS

If PLATFORM is set in "rcar", this option is set in "21" internally.
This option indicates the number of memory areas supported by the OP-TEE OS.
If you do not change the number of memory areas, do not change the value of this option. It may fail to boot.

• CFG_TEE_CORE_DEBUG

Set value is "y" or "n".
The y specifies that assertions and lock checks are enabled.
If this option is not set, the value is set "n" internally.

4.3.2.1 Secure Storage

Secure storage does not support data stored in previous to Yocto 3.4.0 environment.
To use Secure storage, please delete the storage data.
Please refer to 4.4.5 How to clear Secure Storage data.

- **CFG_REE_FS**

Set value is "y" or "n".
The y specifies to enable REE Filesystem.
If this option is not set, the value is set "n" internally.

- **CFG_RPMB_FS**

Set value is "y" or "n".
The y specifies to enable RPMB Filesystem.
If this option is not set, the value is set "n" internally.
Note)To build with RPMB Filesystem option "y", Secure engine require Secure Core.

- **CFG_RPMB_WRITE_KEY**

Set value is "y" or "n".
The y specifies to enable RPMB security key programming.
If this option is not set, the value is set "n" internally.
Note)
If this option is set "y", the key will be written at the first boot when RPMB security key is not written on eMMC.
The key is unique on each SoC, so different key is written on each SoC.
The key written on eMMC can’t be rewritten, therefore only the SoC with the key written can be used if CFG_RPMB_FS is set in "y".

- **CFG_RPMB_RESET_FAT**

Set value is "y" or "n".
The y specifies to initialize RPMB Filesystem.
If this option is set to "y", OP-TEE clearing RPMB File Allocation Table (FAT), when the first accesses to the RPMB. The FAT has elements of the start address, size, and file name of the file data.
If this option is not set, the value is set "n" internally.

- **CFG_STANDALONE_FS**

Set value is "y" or "n".
The y specifies to enable Stand-alone Filesystem.
If this option is not set, the value is set "y" internally.

• STANDALONE_FS_SECTOR_ADDR

If CFG_STANDALONE_FS is set in "y", this option must be set.
This option sets the top address of the sector. You must assign the address from a SPI / HyperFlash.
If this option is not set, the value is set 0x300000 internally.
Hexadecimal must add a 0x as a prefix.

• STANDALONE_FS_SECTOR_NUM

If CFG_STANDALONE_FS is set in "y", this option must be set.
This option sets the number of the total sectors of storage data.
If this option is not set, the value is set 4 internally.
The minimum value is 2. Value must be a multiple of 2.

When saving data, if one or more filesystem is enabling and filesystem is not specified, filesystem of choice see below.

Table 4-4 Priority of the file system.

Priority of the file system			Filesystem
High	Middle	Low	
Stand-alone Filesystem	REE Filesystem	RPMB Filesystem	
√	√	√	Stand-alone Filesystem
√	√	-	Stand-alone Filesystem
√	-	√	Stand-alone Filesystem
√	-	-	Stand-alone Filesystem
-	√	√	REE Filesystem
-	√	-	REE Filesystem
-	-	√	RPMB Filesystem
-	-	-	can not use. Compile error occurs.

√:enable -:disable

4.3.3 OP-TEE Driver

- **CONFIG_OPTEE**

  Set value is "y" or "n".
  User must set y to enable the build of OP-TEE Driver.
  A Linux Kernel configuration file defines this option as "y".
  The configuration file is placed in "$WORK/meta-renesas/meta-rcar-gen3/recipes-kernel/linux/linux-renesas/defconfig".

4.3.4 OP-TEE Client

- **RPMB_EMU**

  OP-TEE Client also has an emulation mode which implements a virtual RPMB device for test purposes.
  Set value is 0 or 1.
  0 specifies to access to a real RPMB device.
  1 specifies to access to a virtual RPMB device.

4.4 How to

4.4.1 How to set an option for build

The following shows the path of recipe file, and the sample for adding or changing build option.
When users change building option, please modify a recipe file to add option.

• Trusted Firmware-A (BL31)

Example for R-Car H3.

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/arm-trusted-firmware/arm-trusted-firmware_git.bb

```
COMPATIBLE_MACHINE = "salvator-x"
PLATFORM = "rcar"
ATFW_OPT_r8a7795 = "LSI=H3 RCAR_DRAM_SPLIT=1 XXXXX=YY"
ATFW_OPT_r8a7796 = "LSI=M3 RCAR_DRAM_SPLIT=2"
ATFW_OPT_r8a77965 = "LSI=M3N"
ATFW_OPT_r8a77990 = "LSI=E3"
ATFW_OPT_r8a77995 = "LSI=D3"
```

Example for R-Car M3.

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/arm-trusted-firmware/arm-trusted-firmware_git.bb

```
COMPATIBLE_MACHINE = "salvator-x"
PLATFORM = "rcar"
ATFW_OPT_r8a7795 = "LSI=H3 RCAR_DRAM_SPLIT=1"
ATFW_OPT_r8a7796 = "LSI=M3 RCAR_DRAM_SPLIT=2 XXXXX=YY"
ATFW_OPT_r8a77965 = "LSI=M3N"
ATFW_OPT_r8a77990 = "LSI=E3"
ATFW_OPT_r8a77995 = "LSI=D3"
```

Example for R-Car M3N.

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/arm-trusted-firmware/arm-trusted-firmware_git.bb

```
COMPATIBLE_MACHINE = "salvator-x"
PLATFORM = "rcar"
ATFW_OPT_r8a7795 = "LSI=H3 RCAR_DRAM_SPLIT=1"
ATFW_OPT_r8a7796 = "LSI=M3 RCAR_DRAM_SPLIT=2"
ATFW_OPT_r8a77965 = "LSI=M3N XXXXX=YY"
ATFW_OPT_r8a77990 = "LSI=E3"
ATFW_OPT_r8a77995 = "LSI=D3"
```

Example for R-Car E3.

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/arm-trusted-firmware/arm-trusted-firmware_git.bb

```
COMPATIBLE_MACHINE = "salvator-x"
PLATFORM = "rcar"
ATFW_OPT_r8a7795 = "LSI=H3 RCAR_DRAM_SPLIT=1"
ATFW_OPT_r8a7796 = "LSI=M3 RCAR_DRAM_SPLIT=2"
ATFW_OPT_r8a77965 = "LSI=M3N"
ATFW_OPT_r8a77990 = "LSI=E3 XXXXX=YY"
ATFW_OPT_r8a77995 = "LSI=D3"
```

Example for R-Car D3.

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/arm-trusted-firmware/arm-trusted-firmware_git.bb

```
COMPATIBLE_MACHINE = "salvator-x"
PLATFORM = "rcar"
ATFW_OPT_r8a7795 = "LSI=H3 RCAR_DRAM_SPLIT=1"
ATFW_OPT_r8a7796 = "LSI=M3 RCAR_DRAM_SPLIT=2"
ATFW_OPT_r8a77965 = "LSI=M3N"
ATFW_OPT_r8a77990 = "LSI=E3 "
ATFW_OPT_r8a77995 = "LSI=D3 XXXXX=YY"
```

DEBUG option

When users add option DEBUG and set it to 1, additionally need to modify like the following steps.

Step1: modify the recipe file

Add the patch file name in the recipe file and change output directory as shown below.

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/arm-trusted-firmware/arm-trusted-firmware_git.bb

```
SRC_URI = \
    "git://github.com/renesas-rcar/arm-trusted-firmware.git;branch=${BRANCH}"
SRCREV = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"


SRC_URI += " \
    file://sample.patch \
"

PV = "v1.5+renesas+git${SRCPV}"

do_deploy() {
    install -d ${DEPLOYDIR}
    install -m 0644 ${S}/build/${PLATFORM}/debug/bl2/bl2.elf
 ${DEPLOYDIR}/bl2-${MACHINE}.elf
    install -m 0644 ${S}/build/${PLATFORM}/debug/bl2.bin ${DEPLOYDIR}/bl2-
 ${MACHINE}.bin
    install -m 0644 ${S}/build/${PLATFORM}/debug/bl2.srec ${DEPLOYDIR}/bl2-
 ${MACHINE}.srec
    install -m 0644 ${S}/build/${PLATFORM}/debug/bl31/bl31.elf
 ${DEPLOYDIR}/bl31-${MACHINE}.elf
    install -m 0644 ${S}/build/${PLATFORM}/debug/bl31.bin
 ${DEPLOYDIR}/bl31-${MACHINE}.bin
    install -m 0644 ${S}/build/${PLATFORM}/debug/bl31.srec
```

• **OP-TEE OS**

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/optee/optee-os_git.bb

```
do_compile() {
    oe_runmake PLATFORM=${PLATFORM} CFG_ARM64_core=y XXXXX=YY
}
```

Example A: How to disable the SPI/HyperFlash Driver.

To disable the SPI/HyperFlash Driver, please set as follows.

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/optee/optee-os_git.bb

```
do_compile() {
    oe_runmake PLATFORM=${PLATFORM} CFG_ARM64_core=y \
                CFG_REE_FS=y CFG_STANDALONE_FS=n
}
```

Example B: How to incorporate TA of optee_test in the OP-TEE OS as early TA.

To incorporate TA of optee_test in the OP-TEE OS as early TA, please set as follows.

Note: Compile the source code of optee_test beforehand and store the ELF file in $TA_WORK.

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/optee/optee-os_git.bb

```
do_compile() {
    oe_runmake PLATFORM=${PLATFORM} CFG_ARM64_core=y \
                CFG_EARLY_TA=y EARLY_TA_PATHS=" \
     $TA_WORK/5b9e0e40-2636-11e1-ad9e-0002a5d5c51b.stripped.elf \
     $TA_WORK/5ce0c432-0ab0-40e5-a056-782ca0e6aba2.stripped.elf \
     $TA_WORK/614789f2-39c0-4ebf-b235-92b32ac107ed.stripped.elf \
     $TA_WORK/731e279e-aafb-4575-a771-38caa6f0cca6.stripped.elf \
     $TA_WORK/873bcd08-c2c3-11e6-a937-d0bf9c45c61c.stripped.elf \
     $TA_WORK/b689f2a7-8adf-477a-9f99-32e90c0ad0a2.stripped.elf \
     $TA_WORK/c3f6e2c0-3548-11e1-b86c-0800200c9a66.stripped.elf \
     $TA_WORK/cb3e5ba0-adf1-11e0-998b-0002a5d5c51b.stripped.elf \
     $TA_WORK/d17f73a0-36ef-11e1-984a-0002a5d5c51b.stripped.elf \
     $TA_WORK/e13010e0-2ae1-11e5-896a-0002a5d5c51b.stripped.elf \
     $TA_WORK/e626662e-c0e2-485c-b8c8-09fbce6edf3d.stripped.elf \
     $TA_WORK/e6a33ed4-562b-463a-bb7e-ff5e15a493c8.stripped.elf \
     $TA_WORK/f157cda0-550c-11e5-a6fa-0002a5d5c51b.stripped.elf \
     "
}
```

Example C: How to increase the Heap size of OP-TEE OS.

To increase the Heap size of OP-TEE OS, please set it as follows.

Note: The initial value of heap size is 196608.

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/optee/optee-os_git.bb

```
do_compile() {
    oe_runmake PLATFORM=${PLATFORM} CFG_ARM64_core=y \
                CFG_CORE_HEAP_SIZE=393216" \
}
```

- **OP-TEE Client**

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/optee/optee-client_git.bb

```
do_compile() {
    make XXXXX=YY
}
```

4.4.2 How to import change code

When users want to import edited code into Yocto Environment, need to create a patch file.
Please create a patch file in accordance with the following steps.

Step1: copy & edit source file

Once building Yocto Environment, SecurityBSP package codes appear under the git directory(see 4.1).
Copy the data under the git directory to the user's two directories (e.g. copy to both org and mod directories).
Then edit the source code only in one directory(e.g. mod directory only).

Step2: create patch file (*.patch)

Use the diff or git diff command between two directories to create a patch.
The followings are examples of command.

```
$diff -uprN unedited_directory_path edited_directory_path > sample.patch
```

```
$git diff unedited_directory_path edited_directory_path > sample.patch
```

Patch file is output under the directory in which user enters the command.

Step3: put created patch file in appropriate directory

Put a patch file in the directory of the same level as the recipe file.
(In most cases, the folder's name is the same name as removing the " _git " from the recipe file name.)

Step4: edit recipe file

Edit the recipe file to import a created patch file into Yocto Environment.

The following is example that user applies a created patch file(sample.patch) to build arm-trusted-firmware(BL31).

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/arm-trusted-firmware/arm-trusted-firmware_git.bb

```
SRC_URI = \
    "git://github.com/renesas-rcar/arm-trusted-firmware.git;branch=${BRANCH}"
SRCREV = "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"

SRC_URI += " \
    file://sample.patch \
"
```

The following is example that user applies a created patch file(sample.patch) to build OP-TEE OS.

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/optee/optee-os_git.bb

```
SRC_URI += " \
    file://0001-add-optee_os-R-Car-support.patch \
    file://sample.patch \
"
```

If do not start up OP-TEE OS, delete the following from the recipe so that TEE supplicant is not started from Linux.

On $WORK/meta-renesas/meta-rcar-gen3/recipes-graphics/images/core-image-renesas-base.inc

```
# Support secure environment
IMAGE_INSTALL_append = " \
    optee-client \
"
```

Note) In this case, need to set the build option SPD to none(see 4.3.1).

Step5: building with bitbake

BL31,OP-TEE OS,OP-TEE Client and OP-TEE Driver can be built individually, respectively.
The following bitbake command is an example of a case to build individually by specifying the receipe file(e.g.XXXX_git.bb).

```
cd $WORK/build_<Building case>
$ bitbake XXXX
```

If the build is successful, the image will be created in $WORK/build_<Building case>/tmp/deploy/images/salvator-x/directory.
If user wants to clean environment, user can run command “bitbake XXXX -c clean”.
For various settings before user run the bitbake command, refer to “1.3.1 Related Document No.1”.

4.4.3 How to implement a Dynamic TA

The following shows a method of implementation of the Dynamic TA.

Please create working directory on the host PC.
Hereafter, describing the "$WORK" as the working directory.

4.4.3.1 Basic instruction of the TA implementation

The basic instruction of the TA implementation, follow slide in the following URL.
http://www.slideshare.net/linaroorg/lcu14103-how-to-create-and-run-trusted-applications-on-optee

The following shows a URL with the sample TA.
https://github.com/jenswi-linaro/lcu14_optee_hello_world
Commit ID: 80346727750aa1009e0e2bc97af29ccaa5b7df77

4.4.3.2 File structure

The following shows file and directory structure of The Dynamic TA.

The files of Dynamic TA are stored at “ta” directory.
The files of CA are stored at “host” directory.

For an overview of each file, see Table 4.5.

```
$WORK
    └─lcu14_optee_hello_world
        ├─doc (unused)
        ├─host
        │   ├─hello_world.c
        │   └─Makefile
        ├─ta
        │   ├─include
        │   │   └─ta_hello_world.h
        │   ├─hello_world_ta.c
        │   ├─Makefile
        │   ├─sub.mk
        │   └─user_ta_header_defines.h
        ├─Android.mk (unused)
        └─Makefile
```

Table 4.5 File list of the sample TA

No.	Filename	Description
1	host\ **hello_world.c**	Source file of CA.
2	host\ **Makefile**	Maikefile of the CA. Specify an executable file name.
3	ta\include\ **ta_hello_world.h**	Header file of CA and TA. Both CA and TA use this header. UUID is defined in this file.
4	ta\ **hello_world_ta.c**	Source file of TA.
5	ta\ **Makefile**	Maikefile of the TA. Specify an executable file name.
6	ta\ **sub.mk**	Sub Makefile of the TA.
7	ta\ **user_ta_header_defines.h**	Configurations file of the TA. Specify configurations.(e.g. TA name, heap size, stack size)
8	**Makefile**	CA of Makefile (host directory) and the TA of the Makefile (ta directory) to build.

The following definition shows the interface and configurations required for dynamic TA.

- TA interface
- TA configuration
  - UUID
  - TA control flag
  - Heap size
  - Stack size

For more information on each definition, described in the following chapters.

4.4.3.3 TA interface

TA must provide the implementation with a number of functions, collectively called the "TA interface".
TA interface is defined as TEE Internal API provided by the OP-TEE.

The following Table 4.6 shows the entry point function implements of TA.
TA interface specification Refer: 1.3.1 Related Document No.3 chapter 4.3 TA Interface.

Table 4.6 Entry point function of TA

No.	Entry point function of TA	Description
1	TA_CreateEntryPoint	This function is invoked at the time of the first session open request.
2	TA_DestroyEntryPoint	This function is invoked at the end of the session close request.
3	TA_OpenSessionEntryPoint	This function is invoked when the session open request.
4	TA_CloseSessionEntryPoint	This function is invoked when the session close request.
5	TA_InvokeCommandEntryPoint	This function is invoked when the client has specified command.

4.4.3.4 TA configuration

The header file determines constitution necessary for TA.

The following shows the header(user_ta_header_defines.h) of Dynamic TA.

```
/*
 * The name of this file must not be modified
 */
#ifndef USER_TA_HEADER_DEFINES_H
#define USER_TA_HEADER_DEFINES_H

#include <ta_hello_world.h> /* To get the TA_HELLO_WORLD_UUID define */

#define TA_UUID TA_HELLO_WORLD_UUID

#define TA_FLAGS                    (TA_FLAG_MULTI_SESSION | TA_FLAG_EXEC_DDR)
#define TA_STACK_SIZE                (2 * 1024)
#define TA_DATA_SIZE                 (32 * 1024)

#define TA_CURRENT_TA_EXT_PROPERTIES \
    { "gp.ta.description", USER_TA_PROP_TYPE_STRING, \
        "Hello World TA" }, \
    { "gp.ta.version", USER_TA_PROP_TYPE_U32, &(const uint32_t){ 0x0010 } }

#endif /*USER_TA_HEADER_DEFINES_H*/
```

The following shows the header(ta_hello_world.h) of CA.

```
#ifndef TA_HELLO_WORLD_H
#define TA_HELLO_WORLD_H

/* This UUID is generated with uuidgen
   the ITU-T UUID generator at http://www.itu.int/ITU-T/asn1/uuid.html */
#define TA_HELLO_WORLD_UUID { 0x8aaaf200, 0x2450, 0x11e4, \
                    { 0xab, 0xe2, 0x00, 0x02, 0xa5, 0xd5, 0xc5, 0x1b} }

/* The TAFs ID implemented in this TA */
#define TA_HELLO_WORLD_CMD_INC_VALUE    0

#endif /*TA_HELLO_WORLD_H*/
```

(1) UUID

UUID is the Universally Unique Resource Identifier type. This type is used to identify TAs and CAs.
See section 3.2.4 in “1.3.1 Related Document No.3” for details on the specifications for UUID.

CA specifies the UUID of the TA on the parameters of the session open request. OP-TEE associate the session and TA. Therefore, it is possible to control the connection of the TA.

TA_UUID of "user_ta_header_defines.h" will be the setting of the UUID.

TA_HELLO_WORLD_UUID of "ta_hello_world.h" will be the setting of the UUID.

(2) TA control flags

This flag means controlling the operation of the TA.
TA_FLAGS of "user_ta_header_defines.h" will be the setting of the TA control flags.

Table 4.7 TA control flags

bit	Use	Description
0	TA_FLAG_USER_MODE	User TA. The flag needed as Dynamic TA.
1	TA_FLAG_EXEC_DDR	All TA must execute from DDR. The flag needed as Dynamic TA.
2	TA_FLAG_SINGLE_INSTANCE	Instance of the TA. The flag needed to instance of TA. Note) See the details Refer: 1.3.1 Related Document No.3 of Table 4-11 / gpd.ta.singleInstance.
3	TA_FLAG_MULTI_SESSION	Multi-session. The flag needed to multi-session. Note) See the details Refer: 1.3.1 Related Document No.3 of Table 4-11 / gpd.ta.multiSession.
4	TA_FLAG_INSTANCE_KEEP_ALIVE	If the flag is enabled, TA instance is keep until the TEE is reset or shutdown. Note) See the details Refer: 1.3.1 Related Document No.3 of Table 4-11 / gpd.ta.instanceKeepAlive.
5	TA_FLAG_UNSAFE_NW_PARAMS	This flag must be disabled.
6	TA_FLAG_REMAP_SUPPORT	Unused.
7	TA_FLAG_CACHE_MAINTENANCE	Following API can be used. TEE_CacheClean TEE_CacheFlush TEE_CacheInvalidate The following URL reference for the API details.

		https://github.com/OP-TEE/optee_os/blob/master/documentation/extensions/extensions.md

Example of setting TA control flag.

Flag setting a single-instance and multi-session will be the following.

#define TA_FLAGS (TA_FLAG_MULTI_SESSION | TA_FLAG_SINGLE_INSTANCE | TA_FLAG_EXEC_DDR)

Note) definition more information Refer: 1.3.1 Related Document No.3 of Table 4-11 / gpd.ta.singleInstance, gpd.ta.multiSession, gpd.ta.instanceKeepAlive reference.

(3) Heap size and stack size

Heap size and stack size are specified in byte.

In case of the Dynamic TA, OPTEE kernel allocates the area of the heap and stack in user space based on the configuration when it loads the TA.

TA_STACK_SIZE of "user_ta_header_defines.h" will be the setting of the stack size.
TA_DATA_SIZE of "user_ta_header_defines.h" will be the setting of the heap size.

Note) See the details Refer: 1.3.1 Related Document No.3 of Table 4-11 / gpd.ta.dataSize, gpd.ta.stackSize reference.

(4) Extensions properties (optional)

The following property shows an example of setting options.

・gp.ta.description：TA Name “Hello World TA”

・gp.ta.version：TA Version

TA_CURRENT_TA_EXT_PROPERTIES of "user_ta_header_defines.h" will be the setting of extensions properties.

(5) Log level of the Dynamic TA

Definition of output to the log is defined in the following files.
$WORK\lcu14_optee_hello_world\ta\Makefile

```
CFG_TEE_TA_LOG_LEVEL ?= 2
CPPFLAGS += -DCFG_TEE_TA_LOG_LEVEL=$(CFG_TEE_TA_LOG_LEVEL)
BINARY=8aaaf200-2450-11e4-abe20002a5d5c51b

include $(TA_DEV_KIT_DIR)/mk/ta_dev_kit.mk

all: $(BINARY).ta

clean: clean_ta_file
.PHONY: clean_ta_file
clean_ta_file:
        rm -f $(BINARY).ta
```

Note) CFG_TEE_TA_LOG_LEVEL defined in ta_dev_kit.mk, but if you want to change the level, please change prior to include.

• CFG_TEE_TA_LOG_LEVEL

OP-TEE OS provides logging functions MSG() , EMSG(), IMSG(), DMSG() and FMSG().
Logging functions beside log level in the following table are valid.
Set one value in the ranges from 1 to 4.

Table 4.8 Dynamic TA Log level

Log level	Function
1	MSG(), EMSG()
2	MSG(), EMSG(), IMSG()
3	MSG(), EMSG(), IMSG(), DMSG()
4	MSG(), EMSG(), IMSG(), DMSG(), FMSG()

Note) When the log level is selected outside the range from 1 to 4, Log level 4 will be selected.

• Log format

The following is the log format that is output from the Dynamic TA.

Example)
CFG_TEE_TA_LOG_LEVEL is built with level 2, the following is an example of an implementation of the IMSG() and DMSG().The log of IMSG() is output, but the log of DMSG() is not output.

Source code

```
static TEE_Result inc_value(uint32_t param_types, TEE_Param params[4])
{
        uint32_t exp_param_types = TEE_PARAM_TYPES(TEE_PARAM_TYPE_VALUE_INOUT,
                                                   TEE_PARAM_TYPE_NONE,
                                                   TEE_PARAM_TYPE_NONE,
                                                   TEE_PARAM_TYPE_NONE);

        if (param_types != exp_param_types)
                return TEE_ERROR_BAD_PARAMETERS;

        params[0].value.a++;
        IMSG("Test IMSG\n");
        DMSG("Test DMSG\n");
        return TEE_SUCCESS;
}
```

Log

```
[     35.463765] [OP-TEE][62.560000][0]INFO:       USER-TA: Test IMSG
```

4.4.3.5 How to build Dynamic TA

The following is the method to build the Dynamic TA.

(1) Precondition

Please build Yocto environment from the beginning, or prepare the necessary SDK from the already built Yocto environment.
When building Yocto environment, follow the procedure of the following reference to build a Yocto environment.
Refer to the "3. Building Instructions" described in the 1.3.1 Related Document No.1.

(2) How to build Dynamic TA

The following explains the make method of Dynamic TA, as an example sample TA.

Step1 : Prepare Dynamic TA group of files the (ta).
Configuration of the file is described in the implementation of the 4.4.3.2.

Step2: Specify the directory path of OP-TEE OS was individually build.

```
$ export TA_DEV_KIT_DIR=$WORK/build_<Building case>/tmp/work/salvator_x-poky-linux/optee-os/<Properties of yocto environment>/git/out/arm-plat-rcar/export-ta_arm64
```

Note) The path of Step2 is the Yocto environment. The Dynamic TA SDK can also be prepared from the path of Step2 without building the Yocto environment.

Step3: Setting the path for Toolchain

```
$ PATH=/opt/poky/<SDK version>/sysroots/x86_64-pokysdk-linux/usr/bin/aarch64-poky-linux:$PATH
```

Step4: Build of Dynamic TA.

```
$ cd $WORK/lcu14_optee_hello_world

$ make -C ta CROSS_COMPILE=aarch64-poky-linux- O=./out
```

The following directory is the output after the successful build.
$WORK/lcu14_optee_hello_world/ta/out/

(3) Output file of Dynamic TA

Build the Dynamic TA is output to the $WORK/lcu14_optee_hello_world/ta/out.
Set the output by Dynamic TA to Linux. Placement is See 4.4.3.6 Method to execute Dynamic TA/Step2(Copy TA)

Output file：**8aaaf200-2450-11e4-abe20002a5d5c51b.ta (128KB)**

Note) String in front of the extension ".ta" is a UUID, different for each was built TA.

Note) Output file name is specified by the Makefile of TA.(4.4.3.2 File structure/Table 4.5/No.5).

Note) Output file is must be less than 1MB.
Shared memory is used for communication between the Normal World and Secure World.
Even when you load the Dynamic TA, it is temporarily place the image in the same shared memory.
The size of the shared memory is 1MB, will be limited loadable Dynamic TA image size, by using state of the shared memory by the world inter-communication.

(4) How to build CA

Following the details of the CA build procedure.

Step1: Prepare CA group of files the (host).
Configuration of the file is described in the implementation of the 4.4.3.2.

Step2: Setting the path of the external reference directory of the generated optee-client in bitbake.

```
$ export TEEC_EXPORT=/opt/poky/<SDK version>/sysroots/aarch64-poky-linux/usr
```

Note) The path of Step2 is the Yocto environment. The CA SDK can also be prepared from the path of Step2 without building the Yocto environment.

Step3: Setting the path for Toolchain (64bit) and environment variable.

```
$ source /opt/poky/<SDK version>/environment-setup-aarch64-poky-linux
```

Step4: Disable the LDFLAGS.

```
$ export LDFLAGS=""
```

Step5: Setting of GCC

$WORK/lcu14_optee_hello_world/host/Makefile
Line: 2
[Before] CC = $(CROSS_COMPILE)gcc
[After] CC ?= $(CROSS_COMPILE)gcc

Step6: Build of CA

```
$ cd $WORK/lcu14_optee_hello_world
$ make -C host CROSS_COMPILE=aarch64-poky-linux-
```

The following directory is the output after the successful build.
$WORK/lcu14_optee_hello_world/host

(5) Output file of CA

Build the CA is output to the $WORK/lcu14_optee_hello_world/host.
Output filename：**hello_world**

Note) Output file name is specified in the Makefile of CA. (4.4.3.2 File structure/Table 4.5/No.2).

4.4.3.6 Method to execute Dynamic TA

The following is the method to execute Dynamic TA.

Step1: Turn on the board, make sure that linux is started.

Note) See the details Refer: 1.3.1 Related Document No.1 chapter 4 and chapter 5.

```
$ salvator-x login: root
```

Step2: Store the executable files (CA and TA).

Copy CA

Location on the environment : $WORK/lcu14_optee_hello_world/host/

Executable file name : hello_world

Location on the board : /usr/bin/

Copy the executable file from the build environment to the board.

Copy TA

Location on the build environment : $WORK/lcu14_optee_hello_world/ta/out/

Executable file name : 8aaaf200-2450-11e4-abe20002a5d5c51b.ta

Location on the board : /lib/optee_armtz/

Copy the executable file from the build environment to the board.

Note) If optee_armtz directory does not exist, create it.

Step3: Execute CA.

```
$ hello_world
```

The following is output if successful.

In case of Log OFF at OP-TEE OS output setting.
Note) See the details Refer: 4.3 Option Setting/0

```
root@salvator-x:/# hello_world
Invoking TA to increment 42
TA incremented value to 43
root@salvator-x:/#
```

In case of Log ON at OP-TEE OS output setting.

```
root@salvator-x:/# hello_world
[    53.597468] [OP-TEE][75.741000][1]DEBUG:    TEE-CORE:tee_ta_init_pseudo_ta_session:294: Lookup pseudo TA 8aaaf200-
2450-11e4-abe2-0002a5d5c51b
[    53.610147] [OP-TEE][75.741000][1]DEBUG:    TEE-CORE:tee_ta_init_user_ta_session:637: Lookup user TA 8aaaf200-2450-
11e4-abe2-0002a5d5c51b (REE)
[OP-TEE][75.741000][1]FLOW:     TEE-CORE:plat_prng_add_jitter_entropy:97: plat_prng_add_jitter_entropy: 0x53
Invoking TA to i[    53.618218] [OP-TEE][75.762000][0]DEBUG:    TEE-CORE:ta_load:317: ELF load address 0x80001000
ncrement 42
[    53.618401] [OP-TEE][75.762000][0]FLOW:     TEE-CORE:trace_syscall:158: syscall #1 (syscall_log)
[OP-TEE][75.762000][0]FLOW:     TEE-CORE:plat_prng_add_jitter_entropy:97: plat_prng_add_jitter_entropy: 0x60
[    53.618433] [OP-TEE][75.762000][0]INFO:     USER-TA: Hello World!
TA incremented v[    53.634273] [OP-TEE][75.778000][0]FLOW:     TEE-CORE:trace_syscall:158: syscall #1 (syscall_log)
alue to 43
[    53.634324] [OP-TEE][75.778000][0]INFO:     USER-TA: Sizeof Pointer=8
[OP-TEE][75.778000][0]FLOW:     TEE-CORE:plat_prng_add_jitter_entropy:97: plat_prng_add_jitter_entropy: 0x11
[    53.643888] [OP-TEE][75.788000][0]DEBUG:    TEE-CORE:tee_ta_close_session:378: tee_ta_close_session(0x441abd50)
[    53.643980] [OP-TEE][75.788000][0]DEBUG:    TEE-CORE:tee_ta_close_session:397: Destroy session
[    53.644030] [OP-TEE][75.788000][0]FLOW:     TEE-CORE:trace_syscall:158: syscall #1 (syscall_log)
[OP-TEE][75.788000][0]FLOW:     TEE-CORE:plat_prng_add_jitter_entropy:97: plat_prng_add_jitter_entropy: 0x9D
[    53.644069] [OP-TEE][75.788000][0]INFO:     USER-TA: Goodbye!
[    53.644354] [OP-TEE][75.788000][0]DEBUG:    TEE-CORE:tee_ta_close_session:423: Destroy TA ctx
root@salvator-x:/#
```

4.4.3.7 How to use TEE Client API from Linux kernel

In the explanation from chapter 4.4.3.1 to chapter 4.4.3.6, CA called TEE Client API as a user mode application, but TEE Client API can also be called from the kernel module. When communicating with the kernel module and TA, TA can use the same binary that you built as described in chapter 4.4.3.1 to chapter 4.4.3.6. CA that calls TEE Client API as a kernel mode application can use the API shown in chapter 3.2.3.1. And user also needs to change the header file to include as below.

In user mode:

$WORK/build_<Building case>/tmp/work/salvator_x-poky-linux/optee-client/<Properties of yocto environment>/git/out/export/include/tee_client_api.h.

In kernel mode:

$WORK/build_<Building case>/tmp/work-shared/salvator-x/kernel-source/include/linux/tee_drv.h

4.4.4 How to operate the Suspend to RAM and resumption

The following is the method to control Suspend to RAM and resumption by Non-secure OS.

(1) Using IPs

The function of Suspend to RAM will use I2C for DVFS, and control the power management IC.

Before Non-secure OS use this function, there is a need to finish controlling I2C for DVFS.

(2) Specification

PSCI API SYSTEM_SUSPEND can only be requested from CPU0 in the master boot processer, and it can also be restored from the requested CPU.

(3) Steps

Step1:

(Non-secure OS) Turn off all of the secondary CPU by CPU_OFF API.

(Non-secure OS) Buck up Normal World registers.

Step2: Suspend

(Non-secure OS) Set ‘KEEPON_DDRx‘ bits of ‘BKUP Mode Cnt’ register on the power management IC to enter DDR backup mode.

(Non-secure OS) Call SYSTEM_SUSPEND API with parameter for entry point of Non-secure OS at CPU0 in the master boot processer.

(BL31) Backup Secure World registers.

(BL31) Set DRAM to DDR self-refresh mode.

(BL31) Halt all power supplies except power supply for DRAM by power management IC.

Step3: Resume(Boot)

(Initial Program Loader) If DRAM status is DDR self-refresh mode, Initial Program Loader skips image load and cancels DDR self-refresh mode.

(BL31) Restore Secure World registers and jump to entry point of Non-secure OS.

(Non-secure OS) Restore Normal World registers.

4.4.5 How to clear Secure Storage data

4.4.5.1 Stand-alone Filesystem

Step1: Please make binary file

Execute the following command in the build environment.
Set "count" to the same value as the build option **STANDALONE_FS_SECTOR_NUM**.
For details of option, refer to 0

```
dd if=/dev/zero ibs=256k count=4 | tr "\000" "\377" > sstdata_clear.bin
```

Step2: Please make S-record file

Execute the following command in the build environment.

```
objcopy -I binary -O srec --adjust-vma=0x44200000 --srec-forceS3
sstdata_clear.bin sstdata_clear.srec
```

Step3: Writing of S-record file.

Refer to the "4.3 How to write" described in the 1.3.1 Related Document No.1.
Set "Flash Save Address" to the same value as the build option **STANDALONE_FS_SECTOR_ADDR**.
For details of option, refer to 0

Filename	Program Top Address	Flash Save Address
sstdata_clear.srec	0x44200000	0x300000

4.4.5.2 REE Filesystem

Step1: Turn on the board, make sure that linux is started.

Refer to the "5. Confirm starting of U-Boot and Linux" described in the 1.3.1 Related Document No.1.

Step2: Clear REE file system data

The followings are examples of command.

```
$ rm -fr /data/tee/
```

4.4.5.3 RPMB Filesystem

Step1: Build OP-TEE OS to set RPMB reset option.

Please change the OP-TEE build option. The following shows sample recipe file.

On $WORK/meta-renesas/meta-rcar-gen3/recipes-bsp/optee/optee-os_git.bb

```
do_compile() {
    oe_runmake PLATFORM=${PLATFORM} CFG_ARM64_core=y
 RCAR_DEBUG_LOG=1 CFG_REE_FS=n CFG_STANDALONE_FS=n
 CFG_RPMB_FS=y CFG_RPMB_RESET_FAT=y
 CFG_CRYPT_HW_CRYPTOENGINE=y
}
```

For details of recipe file, refer to 4.4.1 How to set an option for build.
For details of option, refer to 0

Step2: Writing of OP-TEE image.

Refer to the "4.3 How to write" described in the 1.3.1 Related Document No.1.

Step3: Turn on the board, make sure that linux is started.

Refer to the "5. Confirm starting of U-Boot and Linux" described in the 1.3.1 Related Document No.1.

Step4: Access to RPMB file system and clear storage.

To start clearing the storage, TA needs to access RPMB file system.
Note) In the following example, accessing RPMB using optee test.

When log output is valid, the following log is output.

```
root@salvator-x:~# ./xtest 6001
Test ID: 6001
Run test suite with level=0

TEE test application started with device [(null)]
########################################################
#
# regression
#
########################################################

* regression_6001 Test TEE_CreatePersistentObject
o regression_6001.1 Storage id: 00000001
[    40.740218] [OP-TEE][63.149000][1]ERROR:    TEE-CORE:rpmb_fs_setup:1671: **** Clearing Storage ****
  regression_6001.1 OK
```

Step5: After clearing RPMB file system.

Please set the OP-TEE build option **CFG_RPMB_RESET_FAT** to the value of "n".
For details of option, refer to 0

5. Appendix

5.1 Linux TEE driver fails close session

If the close session fails OP-TEE OS memory resources are not released. To solve this problem, the following is the sample code with the addition of a process to retry a closed session.

$WORK/build_<Building case>/tmp/work-shared/salvator-x/kernel-source/drivers/tee/optee/call.c

```
   #inclede <linux/uaccess.h>
   #include <linux/types.h>
   #include <linux/uaccess.h>
  +#include <linux/delay.h>
   #include "optee_private.h"
   #include "optee_smc.h"


                        return -EINVAL;
                kfree(sess);

  -             shm = get_msg_arg(ctx, 0, &msg_arg, &msg_parg);
  -             if (IS_ERR(shm))
  -                     return PTR_ERR(shm);
  +             while(1) {
  +                     shm = get_msg_arg(ctx, 0, &msg_arg, &msg_parg);
  +                     if (IS_ERR(shm)) {
  +                             pr_err("%s: pid=%d Failed to allocate SHM.\n",
  +                                     __func__, current->pid);
  +                             msleep(10);
  +                     } else {
  +                             break;
  +                     }
  +             }
```

$WORK/build_<Building case>/tmp/work-shared/salvator-x/kernel-source/drivers/tee/optee/core.c

```
   #include <linux/types.h>
   #include <linux/uaccess.h>
   #include <linux/workqueue.h>
  +#include <linux/delay.h>
   #include "optee_private.h"
   #include "optee_smc.h"


                if (!ctxdata)
                        return;

  -             shm = tee_shm_alloc(ctx, sizeof(struct optee_msg_arg), TEE_SHM_MAPPED);
  +             while(1) {
  +                     shm = tee_shm_alloc(ctx, sizeof(struct optee_msg_arg), TEE_SHM_MAPPED);
  +                     if (IS_ERR(shm)) {
  +                             pr_err("%s: pid=%d Failed to allocate SHM.\n",
  +                                     __func__, current->pid);
  +                             msleep(10);
  +                     } else {
  +                             break;
  +                     }
  +             }
```

6. Appendix Virtulization

OP-TEE have experimental virtualization support. This is when one OP-TEE instance can run TAs from multiple virtual machines. This chapter explains for virtulized environment.

Refer to the "1.3.2 Related Site for original software No.20” for the detail about virtulization.

6.1 Hypervisor

Hypervisor is necessary for virtulized environment which has multiple guests OS on Normal World. To use the function of OP-TEE, the function called "TEE mediator" is necessary.

In a virtualized environment that Hypervisor which is implemented "TEE mediator" runs, OP-TEE OS must operate with the virtulization option enabled (CFG_VIRTUALIZATION=y).

6.2 APIs list for virtulization

The following table shows the APIs list for virtulization.

Table 6.23 APIs list of Virtulization

No	I/F	Description
1	rcar_nex_mutex_lock	Get lock of Nexus mutex object
2	rcar_nex_mutex_unlock	Release lock of Nexus mutex object

(1) rcar_nex_mutex_lock

Function				
rcar_nex_mutex_lock				
Parameters				
Type	Value	I/O	Range	Description
struct mutex *	m	in	-	struct of nexus mutex information
Return				
Type	Value		Description	
void	-		-	
Description				
This function gets lock of nexus mutex object. Sleep time when waiting for get ting mutex lock can be defined with CFG_RCAR_MUTEX_DELAY. Default sleep time is set as 1ms. This function is prepared to controll object exclusively over multiple guest OS when CFG_VIRTUALIZATION=y. Note) This function is banned to be called from interrupt context.				

(2) rcar_nex_mutex_unlock

Function				
rcar_nex_mutex_unlock				
Parameters				
Type	Value	I/O	Range	Description
struct mutex *	m	in	-	struct of nexus mutex information
Return				
Type	Value		Description	
void	-		-	
Description				
This function releases lock of nexus mutex object. This function is prepared to controll object exclusively over multiple guest OS when CFG_VIRTUALIZATION=y. Note) This function is banned to be called from interrupt context. The operation of calling this function before calling rcar_nex_mutex_lock() is not guaranteed.				

6.3 Option Setting for virtualization

• CFG_VIRTUALIZATION

The "y" specifies to enable virtualization.

• CFG_VIRT_GUEST_COUNT

If CFG_VIRTUALIZATION is set in "y", this option must be set.
This option sets max number of virtual guests.
If this option is not set, the value is set 3 internally.
Note)
TA_RAM_SIZE must be divisible by the value. And the divided number must be aligned on 4KB boundary.

• CFG_RCAR_MUTEX_DELAY

This option sets the number of sleep time(ms) to wait mutex lock.
If this option is not set, the value is set 1 internally.

• CFG_CORE_RESERVED_SHM

If CFG_VIRTUALIZATION is set in "n", this option is set in "y" internally.
If CFG_VIRTUALIZATION is set in "y", this option is set in "n" internally and this internal setting must no be changed.
The "y" specifies to enable reserved shared memory.
The "n" specifies to enable dynamic shared memory.

• PLATFORM_FLAVOR

If CFG_VIRTUALIZATION is set in "y", set value.
Please refer to Table 5.1 for the setting example.

Table 5.1 Option setting of PLATFORM_FLAVOR

No	Board	SoC environment	PLATFORM_FLAVOR	Default value
1	Salvator-XS	H3 Ver.3.0 (8GB)	salvator_h3_4x2g	This is used as default.
2	Salvator-XS / Salvator-X	H3 Ver.3.0 (4GB)	salvator_h3	
3	Salvator-XS	M3 Ver.3.0 (8GB)	salvator_m3_2x4g	
4	Salvator-XS / Salvator-X	M3 Ver.1.x (4GB)	salvator_m3	
5	Salvator-XS / Salvator-X	M3N	salvator_m3n	
6	Ebisu	E3	ebisu_e3	
7	Ebisu-4D	E3	ebisu_4d_e3	
8	Draak	D3	draak_d3	

REVISION HISTORY	Security Board Support Package User's Manual: Software

Rev.	Date	Description	
		Page	Summary
1.0.0	Mar. 25, 2016	—	New creation.
1.0.1	Apr. 27, 2016	3	Changed the description in the order of ERROR(), NOTICE() .
		8	Changed the edition of 'Linux Interface Specification Yocto recipe Start-Up Guide' 0.90.
		12	Add Hardware environment M3.
		13	Add Shared memory area on Figure 2.2 Software relationship.
		29	4.3.1 Modified valid logging function of log level 10, NOTICE() to ERROR() . Changed the description in the order of ERROR(), NOTICE() .
		30	Changed the description in the order of ERROR(), NOTICE() .
1.0.2	May 27, 2016	9	Changed the edition of 'Linux Interface Specification Yocto recipe Start-Up Guide'
		32	Changed the description to set build option of ARM Trusted Firmware (BL31).
		40	Changed the character "\" in the TA configuration.
1.0.3	Aug. 29, 2016	2	Added a mention of SYSTEM_SUSPEND.
		6	Added a mention of SPI/HyperFlash Driver.
		9	Added to the chapter of the interrupt controller.
		12	Removed restrictions (1) II,III Changed the word primary core to CPU0 in master boot processor. Added restriction (1) II about SYSTEM_OFF and SYSTEM_RESET command.
		13	Changed the title 'Terminology and Abbreviation'. Added in the table SWDT, MFIS, SPI/HyperFlash, Suspend to RAM.
		16	Add Figure 2.2 Secure Storage Filesystem, MFIS, I2C for DVFS, power management IC, System RAM
		19	Changed the word primary core to master boot processor.
		23	Changed the 'Changes from original' of the column on Table 3.5 TEE Internal API.
		5 27 35	Added to the chapter of the System Watchdog Timer.
		6 31 35 38	Added to the chapter of the MFIS Driver.
		34	Added the definitions of System RAM
		44 45	Add the Compile Option of Secure Storage.
		49 52	Changed the Sample of recipe file.
		59 60	Added the description of the Dynamic TA log.
		61 62	Delete the 32-bit from how to build Dynamic TA and CA.
		66	Added the 4.4.4 How to operate the system suspend

1.0.4	Dec. 22, 2016	2	Added about timer count in suspending.
		6	Added the restrictions of the RPMB Filesystem.
		15	Added terminology.
		20	Changed summary of SYSTEM_OFF
		28	Added suspend and resume operation of SWDT driver.
		30	Added the notes for the SWDT.
		32	Added suspend and resume operation of MFIS driver.
		44	Changed the Compile Option SPD Added the Compile Option LSI,PMIC_ON_BOARD,PMIC_LEVEL_MODE.
		45	Deleted the Compile Option RCAR_INTCTX_LOG.
		47	Added the Compile Option RPMB_EMU.
		49	Deleted compile flag addition explanation from DEBUG option.
		51	Added to modify a recipe file of OP-TEE Client
		53	Added recipe correction procedure at building.
		63 64	Changed the build procedure for the Dynamic TA and CA.
1.0.5	Mar. 15, 2017	2	Added a mention of PMIC.
		12	Added the Related Document of PMIC.
		17	Changed the Explanation of Evaluation Board.
		21	Changed the summary of CPU_SUSPEND,CPU_OFF,MIGRATE_INFO_TYPE and MIGRATE_INFO_UP_CPU. Changed the R-Car H3/M3 Supported of MIGRATE_INFO_TYPE and MIGRATE_INFO_UP_CPU.
		45	Changed the Compile Option PMIC_ON_BOARD.
1.0.6	Jul. 12, 2017	—	Fixed the format of the document (trademark, etc.)
		6	Changed the description of MFIS Driver.
		10	Changed the Interrupt ID.
		33	Changed the description of MFIS Driver.
		34	Changed the description of mfis_error_detection_start function.
		37	Changed the description of MFIS Driver definitions.
		40	Changed the description of MFIS Driver structure.
		46	Changed the Option Setting. Divide the option in PMIC and SYSTEM_SUSPEND.
		69	Add the description of how to use the TEE client API from the Linux kernel.
1.0.7	Nov. 14, 2017	—	R-Car M3N support.
		8	Add the Operating Frequency for non-volatile memory.

1.0.8	Jan. 12, 2018	—	Fixed the format of the document (trademark, etc.)
		—	Changed the name optee_linuxdriver to OP-TEE Driver.
		4	Added Socket to a provided function for TA.
		5	Changed the Static TA to the pseudo TA by a name of TA Type of Table1.3. Added the early TA to Table1.3. Changed the Static TA to the pseudo TA by chapter of Watchdog Timer.
		6	Changed the Static TA to the pseudo TA by chapter of MFIS Driver.
		11	Added the Unsupported features of OP-TEE OS. - Secure Data Path - Dynamic shared memory
		13	Added the description of the function of OP-TEE Driver.
		14	Added the TEE Sockets API Specification. Added the Release Note of Security BSP.
		15	Changed the URL uploading the OP-TEE Driver source code.
		22	Changed the device node from OP-TEE Linux Driver to OP-TEE Driver.
		27	Changed the TEE_ResetPersistentObjectEnumerator from 'N/A' to 'Available'.
		28	Changed the TEE_AllocateOperation from original.
		30	Added the TEE_tcpSocket and TEE_udpSocket of TEE iSocket API.
		31	Changed the Static TA to the pseudo TA.
		35	Changed the Static TA to the pseudo TA.
		38-46	Added specifications of TEE Client API for the kernel mode.
		47	Changed the value of DEVICE_SRAM_SIZE.
		49	Add a description of definitions of OP-TEE Driver.
		50	Add a description of structures of OP-TEE Driver.
		53	Fixed the file tree of OP-TEE Driver.
		57	Remove the Compile Option PSCI_DISABLE_BIGLITTLE_IN_CA57BOOT.
		58-60	Removed the Compile Option CFG_ENC_FS. Added the Compile Option CFG_RPMB_WRITE_KEY. Added the Compile Option RPMB_RESET_FAT. Added the Compile Option CFG_EARLY_TA. Added the Compile Option EARLY_TA_PATHS. Added the Compile Option CONFIG_OPTEE.
		62	Changed the PV of arm-trusted-firmware_git.bb.
		63-64	Added the build optional example of optee-os_git.bb.
		66	Deleted the description about the recipe of optee_linuxdriver.
		75	Changed the log format of the OP-TEE OS.
		79	Changed the path of the directory storing TA.
		80	Changed the log output image.
		81	Changed the path to header files to build CA using TEE Client API for kernel mode.
		83	Added the How to clear Secure Storage.
1.0.9	Mar. 14, 2018	—	R-Car E3 support.
		8	Changed maximum operating frequency of HyperFlash to 160MHz.
1.0.10	Apr. 11, 2018	60	Fixed the description about the Compile Option RPMB_EMU.

1.0.11	Jun. 11, 2018	8	Added M3 Ver.1.2 to Table 1.5. Added M3N Ver.1.1 to Table 1.5.
		59	Added the Compile Option CFG_CRYPTO_WITH_CE.
1.0.12	Oct. 12, 2018	—	Ebisu-4D support.
		1	Remove BL1 from Figure 1.1.
		24-25	Added the SYSTEM_RESET2, MEM_PROTECT, and MEM_PROTECT_CHECK_RANGE to Table 3.2.
2.0.0	Nov. 5, 2018	—	Fixed the format of the document (Address List.)
		2 4	Added mention of SiP Service.
		13	Add a description of CFG_TEE_CORE_LOG_LEVEL of OPTEE Driver.
		14	Added Hardware Manual to related documents.
		18	Added terminology.
		24	Changed the summary of SYSTEM_OFF.
		25-30	Added specifications of SiP Service API.
		56	Added a description of the structures of the SiP Service.
		58	Changed directory configuration of BL31.
		64	Added the Compile Option RCAR_SMC_GET_DRAMCONF.
2.0.1(Internal)/ 2.0.2	Mar. 11, 2019	25 31-32	Added RCAR_SIP_SVC_GET_BOARD_TYPE to SiP Service API.
		61	Added the file of the source code of OP-TEE Driver.
		66	Changed the description of RCAR_DEBUG_LOG.
2.0.3	Mar. 22, 2019	—	No update.
2.0.4	Jul. 12, 2019	—	No update.
2.0.6	Feb. 9, 2020	—	No update.
3.0.0	Jun. 9, 2020	12	Added embedded secure device tree in Unsupported features.
		39	Added the note for the SWDT.
3.0.1	Dec. 11, 2020	-	Changed the software name from ARM Trusted Firmware to Trusted Firmware-A.
		-	Removed the description of older versions H3 Ver.1.0/1.1 and M3 Ver.1.0/1.1.
		4	Changed a log format.
		15	Add Reference of virtualization.
		46-47	Added the description of virtualization.
		68	Delete the RCAR_SMC_GET_DRAMCONF option.
		70	Added the Compile Option CFG_RCAR_UNSUPPORT_TA_VER_DB.
		71	Added the note for the RPMB Compile Option CFG_RPMB_WRITE_KEY.
		96-98	Add chapter 5. Appendix Virtulization
3.0.2	Apr. 6, 2021	—	R-Car D3 support.
		—	Changed ARM notation to Arm.
		17	Added the restrictions of the Draak board. Added the restrictions of virtualization environment.
3.0.3	Aug. 16, 2021	—	Fixed the format of the document (Notice, Address List.)

		-	Add information of Gen3e.
		-	Removed the description about SiP Service.
		-	Change the description of restrictions to tabular format.
		60	Added the Compile Option.
3.0.4	Dec. 1, 2021	4	Added note that BL31 run-time log is not supported.
		12	Added specifications of SuspendToRAM.
		17-18	Added the restriction No.7 and No.8.(Moved from Security Board Support Package Release Note: Software)
		50	Added the description of definitions for stack area size for each core.
		70	Added how to increase the Heap size.

CONFIDENTIAL

Security Board Support Package
User's Manual: Software

Publication Date: Rev.1.0.0 Mar. 25, 2016
Rev.3.0.4 Dec. 1, 2021

Published by: Renesas Electronics Corporation

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics Corporation
TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

Renesas Electronics America Inc. Milpitas Campus
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics America Inc. San Jose Campus
6024 Silver Creek Valley Road, San Jose, CA 95138, USA
Tel: +1-408-284-8200, Fax: +1-408-284-2775

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 101-T01, Floor 1, Building 7, Yard No. 7, 8th Street, Shangdi, Haidian District, Beijing 100085, China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai 200333, China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, #06-02 Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit No 3A-1 Level 3A Tower 8 UOA Business Park, No 1 Jalan Pengaturcara U1/51A, Seksyen U1, 40150 Shah Alam, Selangor, Malaysia
Tel: +60-3-5022-1288, Fax: +60-3-5022-1290

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 7.0

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口

http://www.renesas.com

※営業お問合せ窓口の住所は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス株式会社 〒135-0061 東京都江東区豊洲3-2-24（豊洲フォレシア）

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：https://www.renesas.com/contact/

Colophon 3.1

Security Board Support Package User's Manual

Renesas Electronics Corporation