

Linux Interface Specification Device Driver RSCI

User's Manual: Software

RZ/G3E Group

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Electronics Corp. website (<http://www.renesas.com>).

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 depend 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.1.00 Jul 2025)

Corporate Headquarters

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

www.renesas.com

Trademarks

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

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/.

General Precautions in the Handling of Micro processing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Micro processing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.

2. Processing at power-on

The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.

3. Input of signal during power-off state

Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.

4. Handling of unused pins

Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.

5. Clock signals

After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.

6. Voltage application waveform at input pin

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).

7. Prohibition of access to reserved addresses

Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.

8. Differences between products

Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a micro processing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

How to Use This Manual

1. Purpose and Target Readers

This document is designed to provide the user with an understanding of the software development environment for RZ/G3E Group processors. It is intended for users developing software incorporating the processors. A basic knowledge of software development and Linux systems is necessary to use this document.

Particular attention should be paid to the precautionary notes when using the manual. These notes occur within the body of the text, at the end of each section, and in the Usage Notes section.

The revision history summarizes the locations of revisions and additions. It does not list all revisions. Refer to the text of the manual for details.

The following documents apply to the RZ/G3E Group. Make sure to refer to the latest versions of these documents. The newest versions of the documents listed may be obtained from the Renesas Electronics Web site.

Document Type	Description	Document Title	Document No.
User's manual for Hardware	Hardware specifications (pin assignments, memory maps, peripheral function specifications, electrical characteristics, timing charts) and operation description Note: Refer to the application notes for details on using peripheral functions.	RZ/G3E Group User's Manual: Hardware	---
User's manual for Software	Software specifications (basic function of Linux kernel, memory maps, interrupt, clock, pins mux, device tree)	Linux interface Specification - Device Driver RSCI	This User's manual
Application Note	Information on using peripheral functions and application examples Sample programs Information on writing programs in assembly language and C	Available from Renesas Electronics Web site.	
Renesas Technical Update	Product specifications, updates on documents, etc.		

2. Notation of Numbers and Symbols

3. Register Notation

4. List of Abbreviations and Acronyms

Abbreviation	Full Form
bps	bits per second
CRC	Cyclic Redundancy Check
DMA	Direct Memory Access
DMAC	Direct Memory Access Controller
Hi-Z	High Impedance
I/O	Input/Output
LSB	Least Significant Bit
MSB	Most Significant Bit
PLL	Phase Locked Loop
UART	Universal Asynchronous Receiver-Transmitter
I2C	Inter-Integrated Circuit
SPI	Serial Peripheral Interface

Table of Contents

1. Overview.....	1
1.1 Overview	1
1.2 Function	1
1.2.1 Asynchronous Communications Interface (UART)	1
1.2.2 Simple I2C (master-only)	1
1.2.3 Simple SPI.....	1
1.3 Connected Port.....	3
1.3.1 Asynchronous Communications Interface (UART)	3
1.3.2 Simple I2C (master-only)	4
1.3.3 Simple SPI.....	5
1.4 Reference.....	6
1.4.1 Standard.....	6
1.4.2 Related Documents.....	6
1.5 Restrictions	6
1.6 Notice	6
1.6.1 Asynchronous Communications Interface (UART)	6
1.6.2 Simple I2C (master-only)	6
1.6.3 Simple SPI.....	6
2. Terminology.....	7
3. Operating Environment.....	8
3.1 Hardware Environment	8
3.2 Module Configuration.....	9
3.2.1 Asynchronous Communications Interface (UART)	9
3.2.2 Simple I2C (master-only)	10
3.2.3 Simple SPI.....	11
3.3 State Transition Diagram	11
4. External Interface.....	12
4.1 Device Node	12
4.2 External Interface	12
5. Integration.....	13
5.1 Directory Configuration	13
5.1.1 Asynchronous Communications Interface (UART)	13
5.1.2 Simple I2C (master-only)	13
5.1.3 Simple SPI.....	13
5.2 Integration Procedure	14
5.2.1 Enable channel of RSCI module	14
5.2.2 Enable function.....	16
5.3 Device Tree Setting	20
5.3.1 SoC Device Tree	20
5.3.2 Board Device tree	23
5.4 Option Setting	26
5.4.1 Module Parameters	26
5.4.2 Kernel Parameters	26

1. Overview

1.1 Overview

This manual explains the driver module that controls the RSCI (Renesas Serial Communications Interface) module on RZ/G3E Group.

Note: Currently, the devices that support this function are RZ/G3E Group.

1.2 Function

This module control is configurable to 3 interfaces:

- Asynchronous Communications Interface (UART).
- Simple I2C (master-only).
- Simple SPI.

1.2.1 Asynchronous Communications Interface (UART)

- Number of channels: 10
- Data transmission and reception for RS232C.
- Control of communication settings
 - Baud rate 9600, 19200, 38400, 57600, 115200[bps].
 - Parity bit (none/ODD/EVEN).
 - Stop bit (1bit or 2bit).
 - Data transfer bit length (7bit or 8bit).
- Each setting can be changed from the standard ioctl for tty.

1.2.2 Simple I2C (master-only)

- Number of channels: 10.
- Driver functions:
 - Master-only mode.
 - Support transfer with 7-bit address.
 - Transfer rate: Up to 400[kbps].

1.2.3 Simple SPI

- Number of channels: 10.
- Driver functions:
 - SPI transfer functions.
 - Data format.

- Bit rate.
- Buffer configuration.
- Support both master and slave modes.
- Interrupt sources.
- Loop back mode.
- PIO mode when data transmitted is less than 32 bytes.
- DMA mode when data transmitted is greater than 32 bytes.

1.3 Connected Port

1.3.1 Asynchronous Communications Interface (UART)

- Table 1-1 describes port connected to UART function on the RZ/G3E SMARC Evaluation Board Kit.
- The sci2 channel is enabled when SW_SER0_PMOD=1 and SW_SER2_EN=0 in device tree.
- The sci4 channel is enabled when SW_LCD_EN=0 and SW_SER0_PMOD=1 in device tree.
- The sci9 channel is enabled when RSCI9_SEL=0 and SW_LCD_EN=0 in device tree.

Table 1-1 Connected port to UART (RZ/G3E Group) SMARC Evaluation Board Kit

Channel	Connector	Support status	Remark
SCI0	–	No	–
SCI1	–	No	–
SCI2	SMARC module: CN4, SW4 SMARC Carrier-II: SW_OPT_MUX	Yes	Output as M2_KEY_E
SCI3	–	No	–
SCI4	SMARC module: CN4 SMARC Carrier-II: SW_OPT_MUX	Yes	Output as PMOD1_3A
SCI5	–	No	–
SCI6	–	No	–
SCI7	–	No	–
SCI8	–	No	–
SCI9	SMARC module: CN4, SW4	Yes	Output as SER1_UART

1.3.2 Simple I2C (master-only)

- Table 1-2 describes port connected to I2C function on the RZ/G3E SMARC Evaluation Board Kit.
- The rsci_i2c9 channel is enabled when RSCI9_SEL=1 in device tree

Table 1-2 Connected port to I2C (RZ/G3E Group) SMARC Evaluation Board Kit

Channel	Connector	Support status	Remark
rsci_i2c0	–	No	–
rsci_i2c1	–	No	–
rsci_i2c2	–	No	–
rsci_i2c3	–	No	–
rsci_i2c4	–	No	–
rsci_i2c5	–	No	–
rsci_i2c6	–	No	–
rsci_i2c7	–	No	–
rsci_i2c8	–	No	–
rsci_i2c9	SMARC module: CN4, SW4 SMARC Carrier-II: SW_M2_DIS	Yes	Output as PMOD1_6A

1.3.3 Simple SPI

- Table 1-3 describes port connected to SPI function on the RZ/G3E SMARC Evaluation Board Kit.
- The rsci_spi8 channel is enabled when RSPi0_RSCI8_SPI_SEL=1 in device tree
- The rsci_spi9 channel is enabled when RSCI9_SEL=2 in device tree
- On RZ/G3E SMARC Evaluation Board Kit:
 - rsci_spi8 channel output as PMOD0_2A only supports a maximum speed of 12.5MHz. Moreover, PMOD0_2A only supports master mode because TXU0304RUTR IC cannot support bidirectional transfer.
 - rsci_spi9 channel can support a maximum speed of 25MHz with master or slave roles.

Table 1-3 Connected port to SPI (RZ/G3E Group) SMARC Evaluation Board Kit

Channel	Connector	Support status	Remark
rsci_spi0	—	No	—
rsci_spi1	—	No	—
rsci_spi2	—	No	—
rsci_spi3	—	No	—
rsci_spi4	—	No	—
rsci_spi5	—	No	—
rsci_spi6	—	No	—
rsci_spi7	—	No	—
rsci_spi8	SMARC module: CN4	Yes	Output as PMOD0_2A
rsci_spi9	SMARC module: CN4 SMARC Carrier-II: SW_M2_DIS	Yes	Output as PMOD0_2A and PMOD1_6A

1.4 Reference

1.4.1 Standard

There is no reference document on standards.

1.4.2 Related Documents

The following table shows the document related to this module.

Table 1-4 Related documents

Number	Issue	Title	Edition	Date
-	-	-	-	-

1.5 Restrictions

Only one mode can run in a time. No support mode switching in runtime.

1.6 Notice

1.6.1 Asynchronous Communications Interface (UART)

- This driver isn't evaluated with the baud rate of 1Mbps and more.

1.6.2 Simple I2C (master-only)

- This module: Supports enable the rsci_i2c9 of RZ/G3E SMARC Evaluation Board Kit.
- Master-only transfer support.
- In the interrupt sources, the receive data full, transmit end and transmit data empty sources are supported by driver.

1.6.3 Simple SPI

- Master/Slave transfer are both supported.
- Support communication with flash-spi (Pmod SF3).
- Support speed change when transferring.

2. Terminology

The following table shows the terminology related to this module.

Table 2-1 Terminology

Terms	Explanation
SCI	Serial Communications Interface
UART	Universal Asynchronous Receiver-Transmitter
Tx	Transmit
Rx	Receive
I2C	Inter-Integrated Circuit
SCL	Serial Clock
SDA	Serial Data
SPI	Serial Peripheral Interface
SSL	Slave Select
CS	Chip Select
MOSI	Master Out Slave In
MISO	Master In Slave Out

3. Operating Environment

3.1 Hardware Environment

The following table shows the hardware needed to use this module.

Table 3-1 Hardware Environment

Name	Product number
RZ/G3E SMARC Evaluation Board Kit	RTK9947E57S01000BE

3.2 Module Configuration

3.2.1 Asynchronous Communications Interface (UART)

The following Figure 3-1 shows the configuration of serial module.

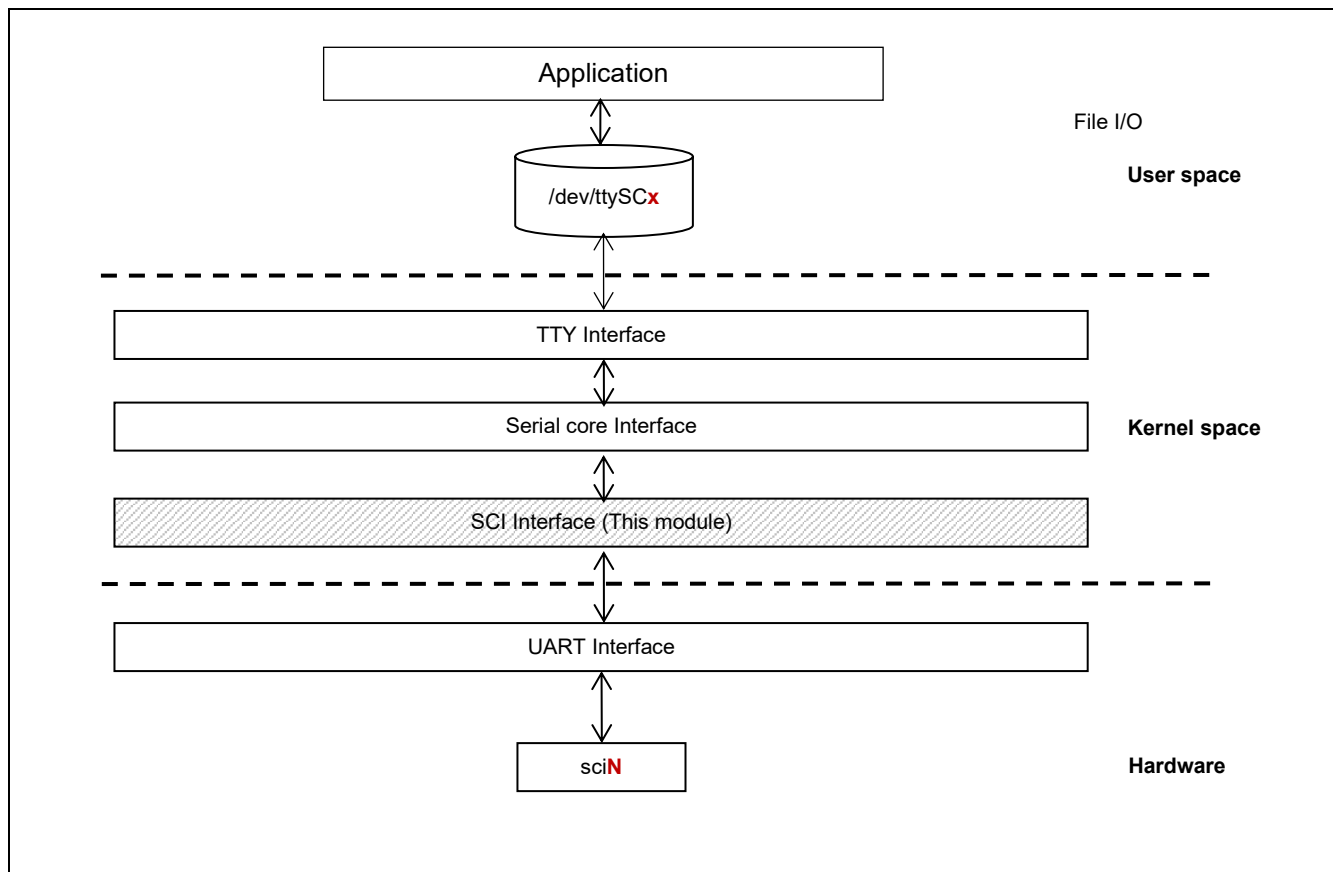


Figure 3-1 The UART from the user space to the hardware space

Note)

x: the serial device which enabled in device tree. (start: 1, 2, ...)

N: the number of sci channels (0, 1, 2, ..., 10)

3.2.2 Simple I2C (master-only)

The following Figure 3-2 shows the configuration of I2C module

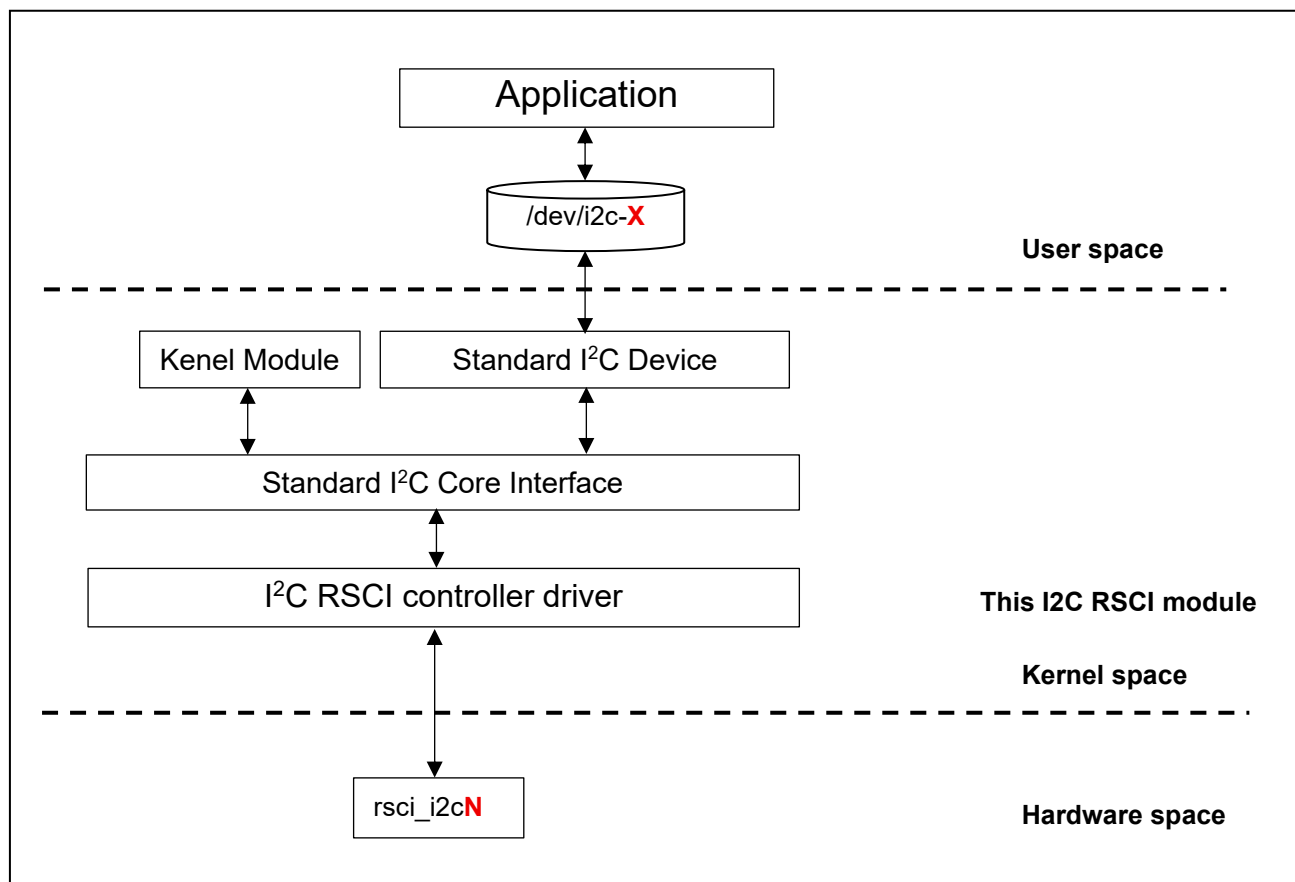


Figure 3-2 The I2C from the user space to the hardware space.

Note)

X: the I2C device number (start: 9, 10, 11, ...)

N: the number of rsci_i2c channel. (0, 1, 2, ..., 10)

3.2.3 Simple SPI

The following Figure 3-3 shows the configuration of SPI module.

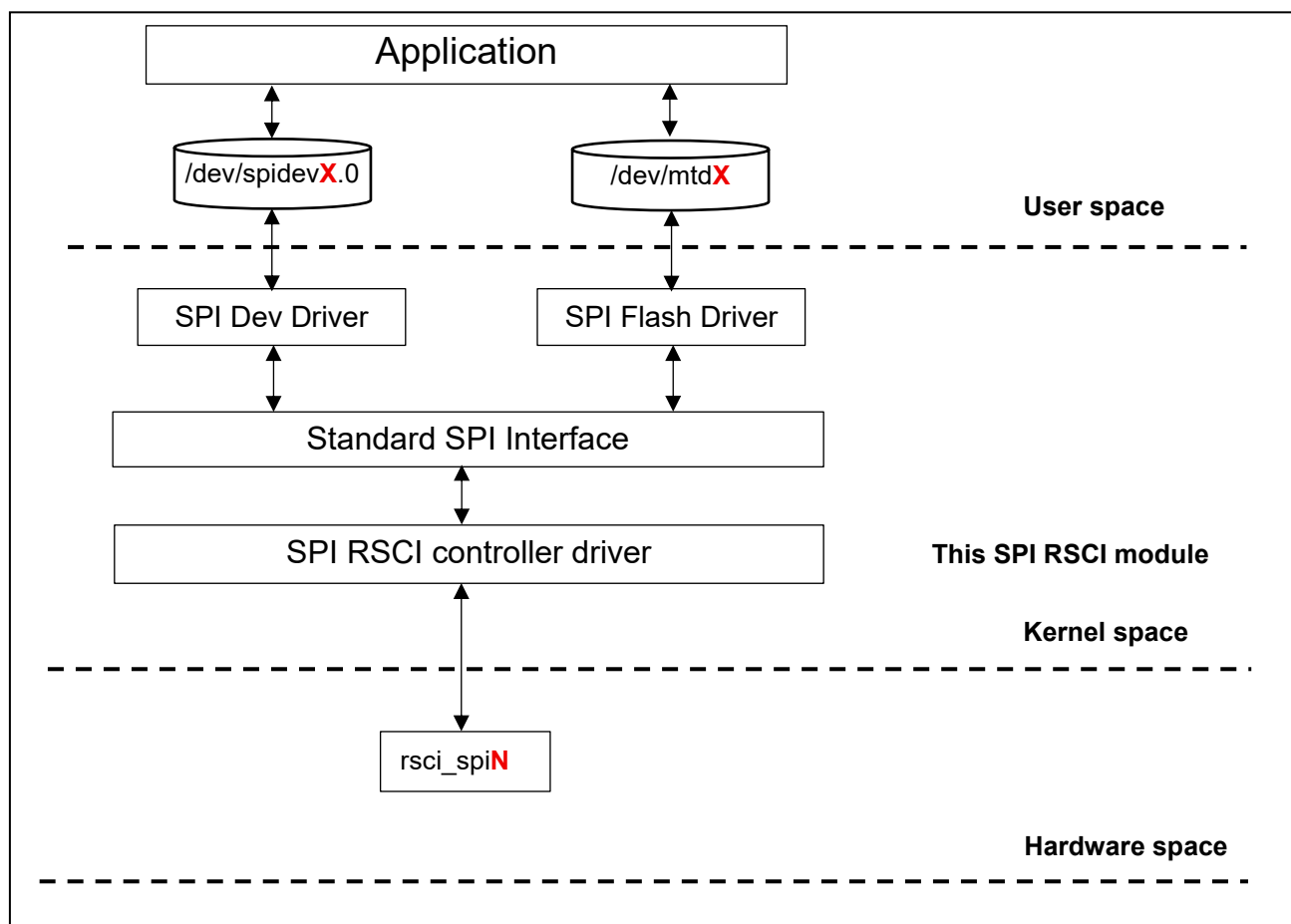


Figure 3-3 The SPI from the user space to the hardware space.

Note)

X: the SPI device number. (start: 0, 1, 2, ...)

For example: in case X=0,

- The /dev/spidev1.0 can be master or slave role if you configure the role in device tree.
- The /dev/mtd0 can communicate with flash device if you configure in device tree.

N: the number of rsci_spi channel. (0, 1, 2, ..., 10)

3.3 State Transition Diagram

There is no state transition diagram for this module.

4. External Interface

4.1 Device Node

Device node of this module is shown below.

Table 4-1 Device Node

Protocol	Device node	Remark
UART	/dev/ttySC x x : 1, 2, 3.	RZ/G3E SMARC Evaluation Board Kit.
I2C	/dev/i2c-9	RZ/G3E SMARC Evaluation Board Kit.
SPI	/dev/spidev0.0 /dev/spidev1.0 /dev/mtd0	RZ/G3E SMARC Evaluation Board Kit.

4.2 External Interface

Detailed explanation is skipped because the external interface of this module is based on Linux

5. Integration

5.1 Directory Configuration

5.1.1 Asynchronous Communications Interface (UART)

The directory configuration of UART is shown below.

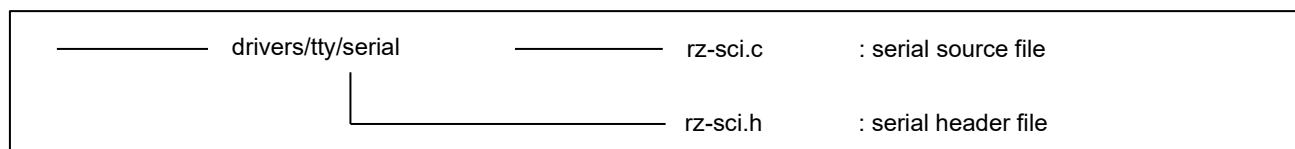


Figure 5-1 Directory Configuration of UART function

5.1.2 Simple I2C (master-only)

The directory configuration of I2C is shown below.

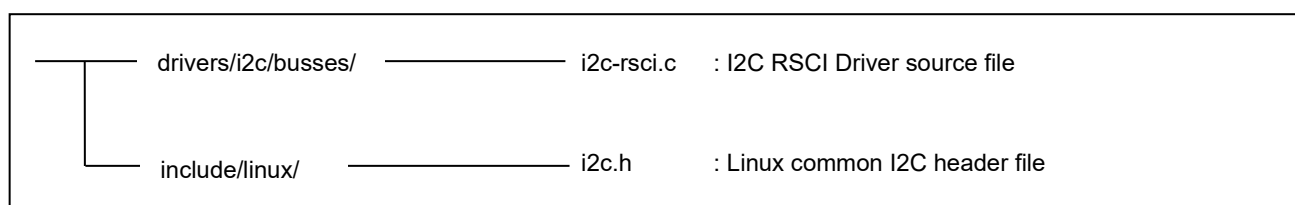


Figure 5-2 Directory Configuration of I2C function

5.1.3 Simple SPI

The directory configuration of SPI is shown below.

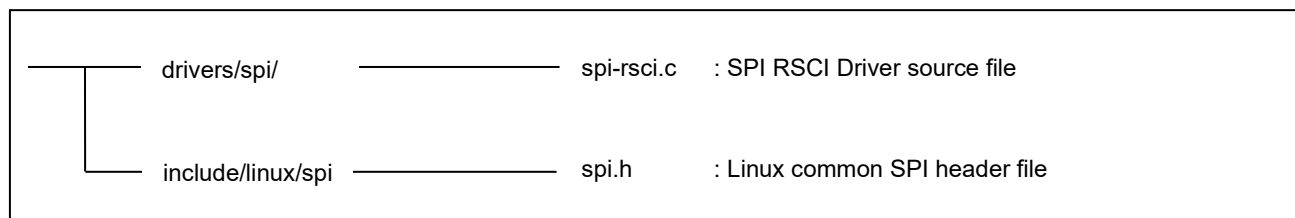


Figure 5-3 Directory Configuration of SPI function

5.2 Integration Procedure

5.2.1 Enable channel of RSCI module

5.2.1.1 Asynchronous Communications Interface (UART)

The following description explains how to enable sci2, sci4 and sci9 channel in RZ/G3E SMARC Evaluation Board Kit.

arch/arm64/boot/dts/renesas/rzg3e-smarc.dtsi

```
aliases {
    #if (!SW_LCD_EN)
        serial1 = &sci4;
    #if (RSCI9_SEL == RSCI_UART)
        serial2 = &sci9;
    #endif
    #endif
    #if ((SW_SER0_PMOD) && (!SW_SER2_EN))
        serial3 = &sci2;
    #endif
};
```

Figure 5-4 Aliases of sci channels on RZ/G3E SMARC

- To enable sci2 channel. We need to set macro of SW_SER2_EN=0 and SW_SER0_PMOD=1. Additionally, we need to make sure that the corresponding switch on the board is correct.
- To enable sci4 channel. We need to set macro of SW_LCD_EN=0. Additionally, we need to make sure that the corresponding switch on the board is correct.
- To enable sci9 channel. We need to set macro of RSCI9_SEL=0 and SW_LCD_EN=0. Additionally, we need to make sure that the corresponding switch on the board is correct.

For example, Figure 5-5 enable 3 channels above.

arch/arm64/boot/dts/renesas/r9a09g047e57-smarc.dts

```
/* Switch selection settings */
#define SW_LCD_EN            0
#define SW_SER0_PMOD        1
#define SW_SER2_EN          0
#define RSCI9_SEL            0
```

Figure 5-5 Enable SCI channels

5.2.1.2 Simple I2C (master-only)

The following description explains how to enable rsci_i2c9 channel in RZ/G3E SMARC Evaluation Board Kit.

- To enable rsci_i2c9 channel. We need to set macro of RSCI9_SEL=1 and SW_LCD_EN=0. Additionally, we need to make sure that the corresponding switch on the board is correct.

For example, Figure 5-6 enable rsci_i2c9 channel.

arch/arm64/boot/dts/renesas/r9a09g047e57-smarc.dts

```
/* Switch selection settings */  
  
#define SW_LCD_EN          0  
  
#define RSCI9_SEL          1
```

Figure 5-6 Enable rsci_i2c9 channel

5.2.1.3 Simple SPI

The following description explains how to enable rsci_spi8 and rsci_spi9 channel in RZ/G3E SMARC Evaluation Board Kit.

- To enable rsci_i2c8 channel. We need to set macro of RSPI0_RSCI8_SPI_SEL=1. Additionally, we need to make sure that the corresponding switch on the board is correct.
- To enable rsci_spi9 channel. We need to set macro of RSCI9_SEL=2 and SW_LCD_EN=0. Additionally, we need to make sure that the corresponding switch on the board is correct.

For example, Figure 5-7 enable rsci_spi8 and rsci_spi9 channels.

arch/arm64/boot/dts/renesas/r9a09g047e57-smarc.dts

```
/* Switch selection settings */  
  
#define RSPI0_RSCI8_SPI_SEL 1  
  
#define SW_LCD_EN          0  
  
#define RSCI9_SEL          2
```

Figure 5-7 Enable rsci_spi channels

5.2.2 Enable function

5.2.2.1 Asynchronous Communications Interface (UART)

UART functions can work normally when we connect serial to this module through Tx and Rx pin.

(1) Kernel configuration settings

To enable the UART function of RSCI module, make the following setting with Kernel Configuration.

```
Device Drivers --->
  [*] Character devices --->
    Serial drivers --->
      <*> Renesas SCI serial port support
```

Figure 5-8 Kernel configuration of UART function driver

Figure 5-9 lists the kernel config symbols to support UART function driver for RZ/G3E Group.

```
CONFIG_SERIAL_RZ_SCI=y
CONFIG_SERIAL_RZ_SCI_CONSOLE=y
```

Figure 5-9 Kenel config symbols of UART function driver

5.2.2.2 Simple I2C (master-only)

The I2C function can work normally when we connect serial to this module through SCL and SDA pin.

(1) Kernel configuration settings

```
Device Drivers --->
  I2C support --->
    <*> I2C device interface
  I2C Hardware Bus support --->
    <*> Renesas RSCI I2C adapter
```

Figure 5-10 Kernel configuration of I2C function driver

Figure 5-11 lists the kernel config symbols to support I2C function driver for RZ/G3E Group

```
CONFIG_I2C_RSCI=y
```

Figure 5-11 Kenel config symbols of I2C function driver

5.2.2.3 Simple SPI

The SPI function can work normally when we connect serial to this module through MOSI, MISO, SCLK, and SSL/CS pin.

(1) Add User Space Interface

Add the compatible value of the Subnode to the file: drivers/spi/spidev.c

“maker name, slave device name”

```
static const struct of_device_id spidev_dt_ids[] = {
    { .compatible = "maker,slavedev" },      ← Add this line.
    {}
};
```

Figure 5-12 Add slavedev compatible

(2) Add Subnode setting in device tree

(a) Master or Slave roles

Use the slave device name of the connection destination as the compatible value. The following form is ideal.
“maker name, slave device name”.

Support transfer board to board through /dev/spidevX.0 on sysfs and spidev_test tool.

After adding compatible above, we will add subnode to rsci_spi node in device tree.

Master mode: adding content as below to rsci_spi node in: **arch/arm64/boot/dts/renesas/rzg3e-smarc.dtsi**

```
/* Add a subnode. */

    slavedev {

        compatible = "maker,slavedev";

        reg = <0>;

        spi-max-frequency = <25000000>;

        spi-cpha;

        spi-cpol;

    };
```

Figure 5-13 The setting for master mode in rsci_spi enable node

Slave mode: adding content as below to rsci_spi node in: **arch/arm64/boot/dts/renesas/rzg3e-smarc.dtsi**

```
spi-slave;

    slave {

        compatible = "maker,slavedev";

        spi-cpha;

        spi-cpol;

    };
```

Figure 5-14 The setting for slave mode in rsci_spi enable node

(b) Connect to flash-SPI device

Add Subnode of flash-spi device to rsci_spi node in device tree.

For example, using Pmod SF3 connect to rsci_spi pin. After adding Subnode as below:

```
flash0 {
    compatible = "micron,mt25ql256a","jedec,spi-nor";

    reg = <0>;

    spi-max-frequency = <12500000>; // => up to 25MHz with rsci_spi9

    spi-tx-bus-width = <1>;

    spi-rx-bus-width = <1>;

    m25p,fast-read;

    partitions {
        compatible = "fixed-partitions";

        #address-cells = <1>;

        #size-cells = <1>;

        partition@0000000 {
            label = "test-area0";

            reg = <0x00000000 0x00080000>;

        };

    };

};
```

Figure 5-15 The setting for flash-spi device for rsci_spi node

(3) Make spidev_test tool

Steps to create **spidev_test** tool are as below:

```
cd {CIP_WORK_DIR}/tools/spi$
source {SDK_DIR}/environment-setup-aarch64-poky-linux
{CIP_WORK_DIR}/tools/spi$ make
```

Figure 5-16 Step make spidev_test tool

Then copy **spidev_test** binary to rootfs. (Master or Slave Board).

Note: Please send the test command on Slave board before sending it on Master board.

(4) Kernel configuration control

```
Device Drivers --->

[*] SPI support --->

    <*> Renesas RSCI RZ/V2H controller
```

Figure 5-17 Kenel config symbols of SPI function driver

To use file system driver JFF2, make the following setting with Kernel Configuration.

```
File Systems --->
    [*] Miscellaneous filesystems -->
        <*> Journaling Flash File System v2 (JFFS2) support
        [*] JFFS2 write-buffering support
```

Figure 5-18 File system configuration support for flash SPI

Figure 5-19 lists the kernel config symbols to support SPI function driver for RZ/G3E Group

```
CONFIG_SPI_RZ_RSCI=y
```

Figure 5-19 Kenel config symbols of SPI function driver

Add the configuration to enable slave mode in: **arch/arm64/configs/defconfig**

```
CONFIG_SPI_SLAVE=y
```

Figure 5-20 Configuration support for slave mode

5.3 Device Tree Setting

5.3.1 SoC Device Tree

Basic configuration information of sci, rsci_i2c, rsci_spi node is defined at device tree for RZ/G3E Group

(arch/arm64/boot/dts/renesas/r9a09g047.dtsi)

5.3.1.1 Asynchronous Communications Interface (UART)

The initial reference information is shown below. An example here is sci9 channel

```
sci9: serial@12803000 {
    compatible = "renesas,r9a09g047-rz-rscif",
                "renesas,rz-rscif";

    reg = <0 0x12803000 0 0x400>;

    interrupts = <GIC_SPI 168 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 169 IRQ_TYPE_EDGE_RISING>,
                <GIC_SPI 170 IRQ_TYPE_EDGE_RISING>,
                <GIC_SPI 171 IRQ_TYPE_LEVEL_HIGH>;

    interrupt-names = "eri", "rx", "tx", "te";

    clocks = <&cpg CPG_MOD 138>,
            <&cpg CPG_MOD 139>;

    clock-names = "fck", "tclk";

    power-domains = <&cpg>;

    resets = <&cpg 147>,
            <&cpg 148>;

    reset-names = "presn", "tresn";

    status = "disabled";
};
```

Figure 5-21 Device tree initial references information of SCI9

Required properties:

compatible: must be "renesas,r9a09g047-rz-rscif", "renesas,rz-rscif"

reg: need two resources, first one mapped to register base address, second one mapped to extension resource registers base address.

interrupts: interrupt specifier for the sci9 channel

clocks, resets: phandle + clock/reset specifier pairs

5.3.1.2 Simple I2C (master-only)

The initial reference information is shown below. An example here is rsci_i2c9 channel

```
rsci_i2c9: i2c@12803000 {
    compatible = "renesas,r9a09g047-rz-rsci-i2c",
               "renesas,rsci-i2c";

    reg = <0 0x12803000 0 0x400>;

    interrupts = <GIC_SPI 168 IRQ_TYPE_LEVEL_HIGH>,
               <GIC_SPI 169 IRQ_TYPE_EDGE_RISING>,
               <GIC_SPI 170 IRQ_TYPE_EDGE_RISING>,
               <GIC_SPI 171 IRQ_TYPE_LEVEL_HIGH>;

    interrupt-names = "eri", "rx", "tx", "te";

    clocks = <&cpg CPG_MOD 138>,
           <&cpg CPG_MOD 139>;

    clock-names = "fck", "tclk";

    clock-frequency = <1000000>;

    power-domains = <&cpg>;

    resets = <&cpg 147>,
           <&cpg 148>;

    reset-names = "presetsn", "tresetsn";

    status = "disabled";
};
```

Figure 5-22 Device tree initial references information of rsci_i2c9

Required properties:

compatible: must be "renesas,r9a09g047-rz-rsci-i2c" or "renesas,rsci-i2c";

reg: need two resources, first one mapped to register base address, second one mapped to extension resource registers base address.

interrupts: interrupt specifier for the rsci_i2c9 channel

clocks, resets: phandle + clock/reset specifier pairs

5.3.1.3 Simple SPI

The initial reference information is shown below. An example here is rsci_spi9 channel

```
rsci_spi9: spi@12803000 {
    compatible = "renesas,r9a09g047-rz-rsci-spi",
                "renesas,rsci-spi";

    reg = <0 0x12803000 0 0x400>;

    interrupts = <GIC_SPI 168 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 169 IRQ_TYPE_EDGE_RISING>,
                <GIC_SPI 170 IRQ_TYPE_EDGE_RISING>,
                <GIC_SPI 171 IRQ_TYPE_LEVEL_HIGH>;

    interrupt-names = "eri", "rx", "tx", "te";

    clocks = <&cpg CPG_MOD 138>,
            <&cpg CPG_MOD 139>;

    clock-names = "fck", "tclk";

    dmas = <&dmac1 0x7F44AE>, <&dmac1 0x7F44AD>;

    dma-names = "tx", "rx";

    power-domains = <&cpg>;

    resets = <&cpg 147>,
            <&cpg 148>;

    reset-names = "presetrn", "tresetrn";

    num-cs = <1>;

    #address-cells = <1>;

    #size-cells = <0>;

    status = "disabled";
};
```

Figure 5-23 Device tree initial references information of rsci_spi9

Required properties:

compatible: must be "renesas,r9a09g047-rz-rsci-spi", or "renesas,rsci-spi";

reg: need two resources, first one mapped to register base address, second one mapped to extension resource registers base address.

interrupts: interrupt specifier for the rsci_spi9 channel

clocks, resets: phandle + clock/reset specifier pairs

To enable DMA mode on SPI function, we need to add **dmass** and **dma-names** property to **rsci_spi9** node in device tree as below:

```
&rsci_spi9{
    dmas = <&dmac1 0x7F44AE>, <&dmac1 0x7F44AD>;
    dma-names = "tx", "rx";
};
```

Figure 5-24 Example of definition about DMA mode in SPI function

5.3.2 Board Device tree

The following describes an example of UART, I2C, SPI function driver of RSCI module on RZ/G3E SMARC Evaluation Board Kit in **arch/arm64/boot/dts/renesas/rzg3e-smarc.dtsi**

5.3.2.1 Asynchronous Communications Interface (UART)

An example here is sci9 node

Note: Ensure sci9 node macro is enabled.

```
aliases {
    serial2 = &sci9;
};

&pinctrl {
    sci9_pins: sci9 {
        pinmux = <RZV2H_PORT_PINMUX(8, 3, 5)>, /* SCI9_TXD_MOSI_SDA */
                <RZV2H_PORT_PINMUX(8, 2, 5)>; /* SCI9_RXD_MISO_SCL */

        bias-pull-up;
    };
};

&sci9 {
    pinctrl-0 = <&sci9_pins>;
    pinctrl-names = "default";
    status = "okay";
};
```

Figure 5-25 Example of setting sci9 node in device tree

5.3.2.2 Simple I2C (master-only)

An example here is rsci_i2c9 node

Note: Ensure rsci_i2c9 node macro is enabled.

```
&pinctrl {  
    rsci_i2c9_pins: rsci_i2c9 {  
        pinmux = <RZV2H_PORT_PINMUX(6, 6, 5)>, /* SCI9_TXD_MOSI_SDA */  
        <RZV2H_PORT_PINMUX(6, 5, 5)>; /* SCI9_RXD_MISO_SCL */  
        drive-open-drain = <1>;  
    };  
};  
  
&rsci_i2c9 {  
    pinctrl-0 = <&rsci_i2c9_pins>;  
    pinctrl-names = "default";  
    status = "okay";  
};
```

Figure 5-26 Example of setting rsci_i2c9 node in device tree

Notice that RSCI module has multiple functions, so if we use I2C function, we need to add **drive-open-drain = <1>;** property to rsci_i2c9_pins node to module can work normally.

5.3.2.3 Simple SPI

An example here is `rsci_spi9` node. Furthermore, it can support master or slave role.

To use master or slave role of `rsci_spi9`, we will add Subnode to `rsci_spi9` node listed in **Add Subnode setting in device tree** section.

Master or Slave role will have **master_sci9_pins** or **slave_sci9_pins**. Default is **master_sci9_pins** when adding subnode Slave mode (**spi-slave** property has been added) will use **slave_sci9_pins**

Note: Ensure `rsci_spi9` node macro is enabled.

```
&pinctrl {
    master_sci9_pins: master_sci9 {
        pinmux = <RZV2H_PORT_PINMUX(6, 6, 5)>, /* SCI9_TXD_MOSI_SDA */
                <RZV2H_PORT_PINMUX(6, 5, 5)>, /* SCI9_RXD_MISO_SCL */
                <RZV2H_PORT_PINMUX(8, 4, 5)>; /* SCI9_SCLK */
    };
    slave_sci9_pins: slave_sci9 {
        pinmux = <RZV2H_PORT_PINMUX(6, 6, 5)>, /* SCI9_TXD_MOSI_SDA */
                <RZV2H_PORT_PINMUX(6, 5, 5)>, /* SCI9_RXD_MISO_SCL */
                <RZV2H_PORT_PINMUX(8, 4, 5)>, /* SCI9_SCLK */
                <RZV2H_PORT_PINMUX(8, 5, 5)>; /* SCI9_SSL */
    };
};

&rsci_spi9 {
    pinctrl-0 = <&master_sci9_pins>;
    pinctrl-1 = <&slave_sci9_pins>;
    pinctrl-names = "master", "slave";
    status = "okay";

    /* Using cs-gpios property when is master mode */
    cs-gpios = <&pinctrl RZV2H_GPIO(6, 4) GPIO_ACTIVE_HIGH>;

    /* Add subnode at here */
};
```

Figure 5-27 Example of setting `rsci_spi9` node in device tree

Note)

In the SPI function driver of RSCI module support master/slave pins. Moreover, in master mode of SPI function don't support chip select pinfunction, so we use chip select pin of gpio with the name of property is **cs-gpios**.

5.4 Option Setting

5.4.1 Module Parameters

5.4.1.1 Asynchronous Communications Interface (UART)

There are no parameters.

5.4.1.2 Simple I2C (master-only)

We can set I2C transfer speed from 20KHz to 400KHz in "**clock-frequency**" property of device tree file in arch/arm64/boot/dts/r9a09g047.dtsi file. No setting means 100KHz.

5.4.1.3 Simple SPI

There are no parameters.

5.4.2 Kernel Parameters

There are no kernel parameters.

REVISION HISTORY	Linux Interface Specification Device Driver RSCI User's Manual: Software
------------------	---

Rev.	Date	Description	
		Page	Summary
1.00	Jul. 22, 2025	—	First Edition issued

Linux Interface Specification Device Driver RSCI
User's Manual: Software

Publication Date: Rev.1.00 Jul. 22, 2025

Published by: Renesas Electronics Corporation

RZ/G3E Group