

Linux Interface Specification Device Driver I2C

User's Manual: Software

RZ/G2L Group, RZ/V2L Group, RZ/V2N Group,
RZ/G3E Group and RZ/Five

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 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. 1.13 Jul. 22, 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 Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing 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 microprocessing 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 manual is designed to provide the user with an understanding of the hardware functions and electrical characteristics of the MPU. It is intended for users designing application systems incorporating the MPU.. It is intended for users developing software incorporating the processors. A basic knowledge of software development and Linux systems is necessary in order 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/G2L Group, RZ/V2L Group, RZ/Five Group, RZ/G3E Group and RZ/V2N 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/G2L Group User's Manual: Hardware	---
		RZ/V2L Group User's Manual: Hardware	---
		RZ/Five Group User's Manual: Hardware	---
		RZ/V2N Group User's Manual: Hardware	---
		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 I2C	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

Table of Contents

1. Overview	1
1.1 Overview	1
1.2 Function	1
1.2.1 Driver Function	1
1.2.2 Transfer Speed	2
1.2.3 Connected Device	2
1.3 Reference	5
1.3.1 Standards	5
1.3.2 Related Documents	5
1.4 Restrictions	6
1.5 Notice	7
2. Terminology	8
3. Operating Environment	9
3.1 Hardware Environment	9
3.2 Module Configuration	10
3.3 State Transition Diagram	14
4. External Interface	15
4.1 Device Node	15
4.2 External Function	15
4.3 Structure	17
4.3.1 struct i2c_adapter	17
4.3.2 struct i2c_client	18
4.3.3 struct i2c_board_info	18
4.3.4 struct i2c_msg	19
4.3.5 struct i2c_rdwr_ioctl_data	19
4.3.6 struct i2c_smbus_ioctl_data	19
4.3.7 union i2c_smbus_data	19
4.4 Global Variables and Constants	20
4.4.1 Global Variables	20
4.4.2 Global Constants	20
4.5 Definitions	21
4.5.1 Device information in Device Tree	21
5. Integration	25
5.1 Directory Configuration	25
5.2 Integration Procedure	25
5.3 Option Setting	26
5.3.1 Module Parameters	26
5.3.1.1 DT	26
5.3.1.2 Guideline to support Slave mode testing	26
5.3.2 Kernel Parameters	27
5.3.3 Device tree bindings	27

1. Overview

1.1 Overview

This manual explains the Linux I2C device driver in RZ/G2L Group, RZ/V2L Group 2nd Generation, RZ/Five Group, RZ/V2N Group, RZ/G3E Group.

Note: Currently, this device is supported in two kernel versions v5.10 and v6.1 with the information below:

- v5.10: RZ/G2L Group, RZ/V2L group and RZ/Five.
- v6.1: RZ/G2L, RZ/G2LC, RZ/V2N and RZ/G3E.

1.2 Function

This module transmits/receives data to/from a device connected to the I2C interface on RZ/G2L, RZ/V2L, RZ/G2LC, RZ/G2UL, RZ/Five, RZ/V2N, RZ/G3E.

1.2.1 Driver Function

The following table lists the functions of this module.

Table 1-1 Driver Function

Function	RZ/G2L, RZ/V2L, RZ/Five	RZ/V2N, RZ/G3E
Number of channels	4	9
Channel	Ch0, Ch1, Ch2, Ch3	Ch0, Ch1, Ch2, Ch3, Ch4, Ch5, Ch6, Ch7, Ch8
Master Mode	Support	Support
Slave Mode	Support up to 3 devices	Support up to 3 devices
7-bit address	Support	Support
10-bit address	Not support	Not support

1.2.2 Transfer Speed

The transfer rate is up to 1 Mbps. For master operation, the duty cycle of the SCL clock is selectable in the range from 0% to 100%.

Table 1-2 Transfer speed (RZ/G2L, RZ/V2L, RZ/Five, RZ/V2N group, RZ/G3E group)

Interface mode	Real transfer speed	Support
Slow speed transfer (<100kHz)	Less than 100Kbit/s	yes
Standard mode (100KHz)	100Kbit/s	yes
Fast mode (400KHz)	400Kbit/s	yes
Fast mode plus (1MHz)	1000kbit/s	yes
High Speed (3.4MHz)	3.4Mbit/s	no

1.2.3 Connected Device

This module connects the following device RZ/G2L, RZ/G2LC, RZ/G2UL, RZ/V2L, RZ/Five, RZ/V2N, RZ/G3E System Evaluation Board.

Table 1-3 Connected device (RZ/G2L and RZ/V2L)

Channel	Device	Category	Address	Remark
I2C0	OV5645	CAMERA	0x3c	-
I2C1	ADV7535	HDMI	0x3d	-
I2C2	-	-	-	-
I2C3	WM8978	Audio	0x1a	-
	VERSA3	Clock	0x68	

Table 1-4 Connected device (RZ/G2LC)

Channel	Device	Category	Address	Remark
I2C0	OV5654	CAMERA	0x3c	-
I2C1	ADV7535	HDMI	0x3d	-
I2C2	VERSA3	CLOCK	0x68	-
	WM8978	AUDIO	0x1a	
I2C3	-	-	-	-

Table 1-5 Connected device (RZ/G2UL)

Channel	Device	Category	Address	Remark
I2C0	DA9062	PMIC	0x58	-
	VERSA3	CLOCK	0x68	
	OV5645	CAMERA	0x3c	
I2C1	ADV7535	HDMI	0x39	-
	WM8978	AUDIO	0x1a	
I2C2	-	-	-	-
I2C3	-	-	-	-

Table 1-6 Connected device (RZ/Five)

Channel	Device	Category	Address	Remark
I2C0	DA9062	PMIC	0x58	-
	VERSA3	CLOCK	0x68	
	OV5645	CAMERA	0x3c	
I2C1	ADV7535	HDMI	0x39	-
	WM8978	AUDIO	0x1a	
I2C2	-	-	-	-
I2C3	-	-	-	-

Table 1-7 Connected devices (RZ/V2N)

Channel	Device	Category	Address	Remark
I2C0	OV5645	CAMERA	0x3c	-
I2C1	OV5645	CAMERA	0x3c	-
I2C2	DA7212	CODEC	0x1a	-
I2C3	ADV7535	HDMI	0x3d	-
I2C4	-	-	-	-
I2C5	-	-	-	-
I2C6	-	-	-	-
I2C7	-	-	-	-
I2C8	5P35023	versa clock generator	0x69	-

Table 1-8 Connected devices (RZ/G3E)

Channel	Device	Category	Address	Remark
I2C0	OV5645	CAMERA	0x3c	-
I2C1	DA7212	CODEC	0x1a	-
I2C2	-	-	-	-
I2C3	-	-	-	-
I2C4	-	-	-	-
I2C5	-	-	-	-
I2C6	-	-	-	-
I2C7	ADV7535	HDMI	0x3d	-
I2C8	5P35023	versa clock generator	0x68	-

1.3 Reference

1.3.1 Standards

The following table shows the standard that this module corresponds.

Table 1-9 Standard

Reference Number	Issue	Title	Edition	Date
-	NXP Semiconductors	THE I2C-BUS SPECIFICATION	-	-
-	SBS Implementers Forum	System Management Bus (SMBus) Specification	Version 2.0	Aug. 03, 2000

1.3.2 Related Documents

The following table shows the document related to this module.

Table 1-10 Related documents

Number	Issue	Title	Edition	Date
-	-	-	-	-

1.4 Restrictions

There is no restriction in this module.

1.5 Notice

- Board status:
 - Support channels 0, 1 and 3 on RZ/G2L and RZ/V2L Smarc.
 - Support channels 0, 1 and 2 on RZ/G2LC Smarc.
 - Support channels 0 and 1 on RZ/G2UL and RZ/Five Smarc.
 - Support channels 0, 1, 2, 3 and 8 on RZ/V2N
 - Support channels 0, 1, 7 and 8 on RZ/G3E.
 - Both master and slave modes are supported.

2. Terminology

The following table shows the terminology related to this module.

Table 2-1 Terminology

Terms	Explanation
I2C	Inter-Integrated Circuit

3. Operating Environment

3.1 Hardware Environment

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

Table 3-1 Hardware specification (RZ/G2L, RZ/V2L, RZ/Five)

Name	Version	Manufacturer
RZ/G2L Evaluation Board Kit	SMARC module: v01 Carrier board: v03	Renesas Electronics Europe GmbH
RZ/G2LC Evaluation Board Kit	SMARC module: v01 Carrier board: v03	Renesas Electronics Europe GmbH
RZ/G2UL Evaluation Board Kit	SMARC module: v01 Carrier board: v03	Renesas Electronics Europe GmbH
RZ/V2L Evaluation Board Kit	SMARC module: v01 Carrier board: v03	Renesas Electronics Europe GmbH
RZ/Five Evaluation Board Kit	SMARC module: v01 Carrier board: v03	Renesas Electronics Europe GmbH

Table 3-2 Hardware specification (RZ/V2N and RZ/G3E)

Name	Product number
RZ/V2N Evaluation Kit	RTK0EF0186C03000BJ
RZ/G3E SMARC Evaluation Board Kit	RTK9947E57S01000BE

3.2 Module Configuration

The following figure shows the configuration of this module.

In the case of RZ/G2L and RZ/V2L, you can control the connected device using the `/dev/i2c-0` or `/dev/i2c-3`.

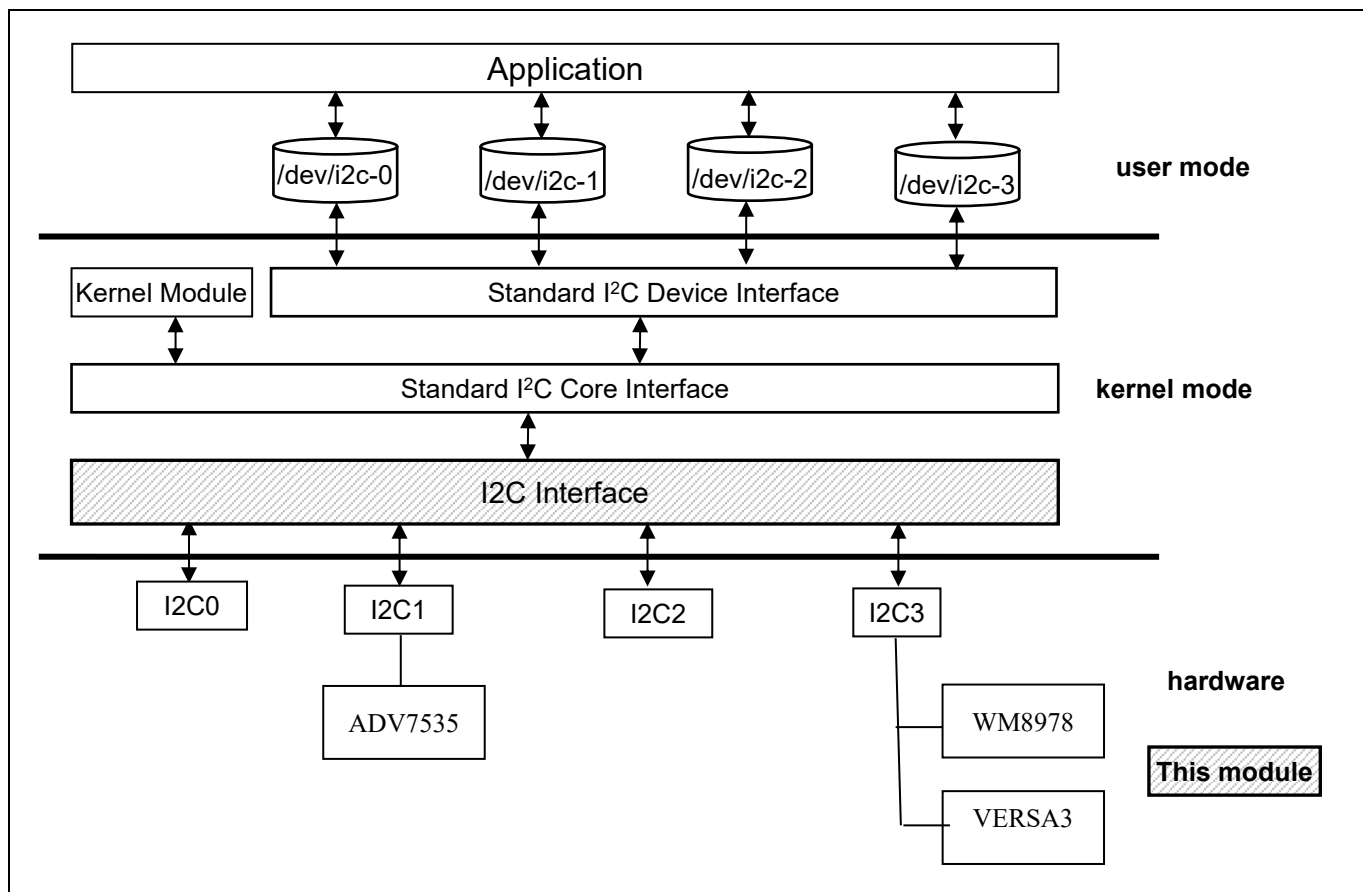


Figure 3-1 I2C Driver Module configuration (RZ/G2L and RZ/V2L)

In the case of RZ/G2LC you can control the connected device using the /dev/i2c-0, /dev/i2c-1 and /dev/i2c-2.

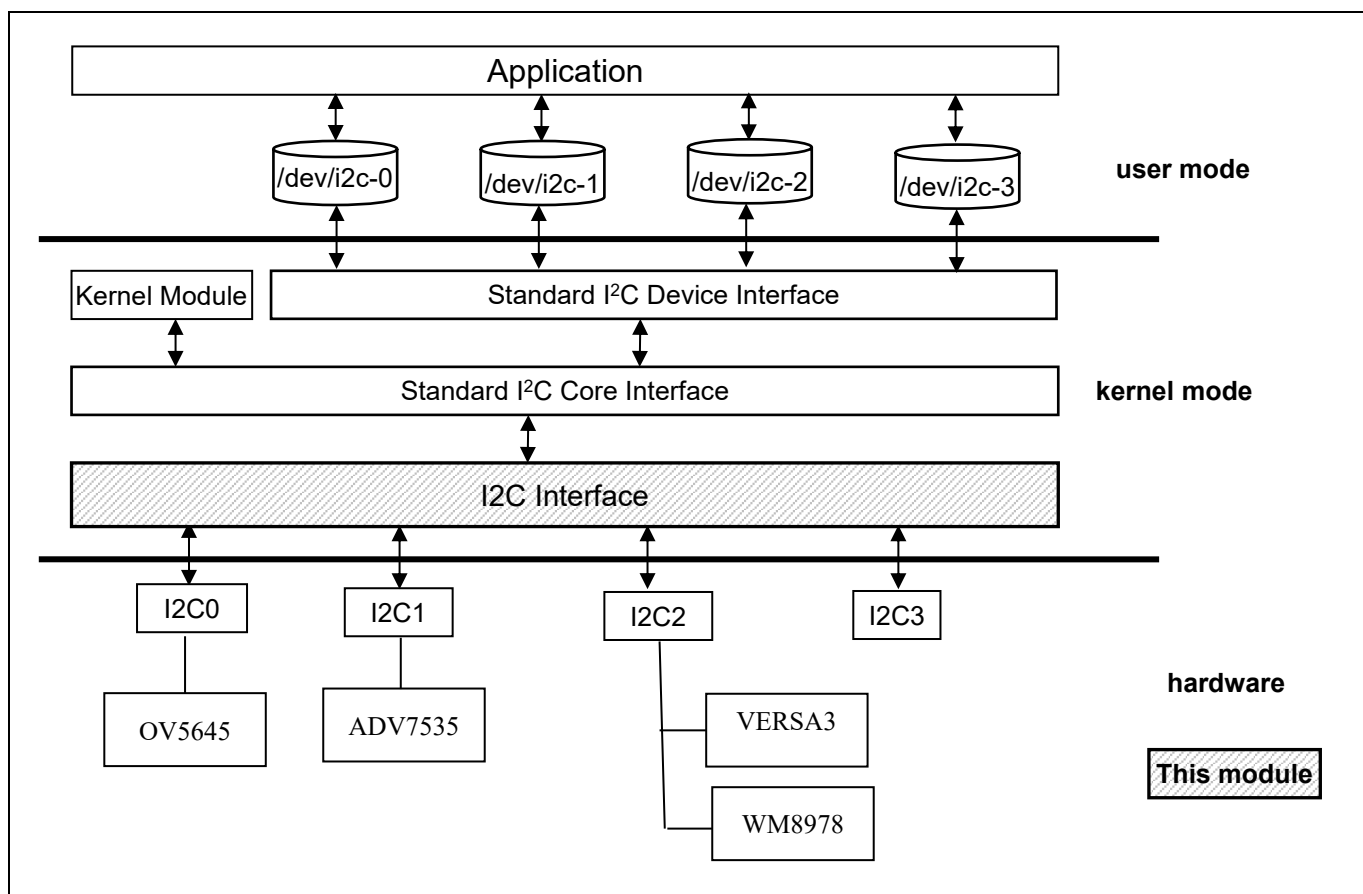


Figure 3-2 I2C Driver Module configuration (RZ/G2LC)

In the case of RZ/G2UL and RZ/Five you can control the connected device using the /dev/i2c-0 or /dev/i2c-1.

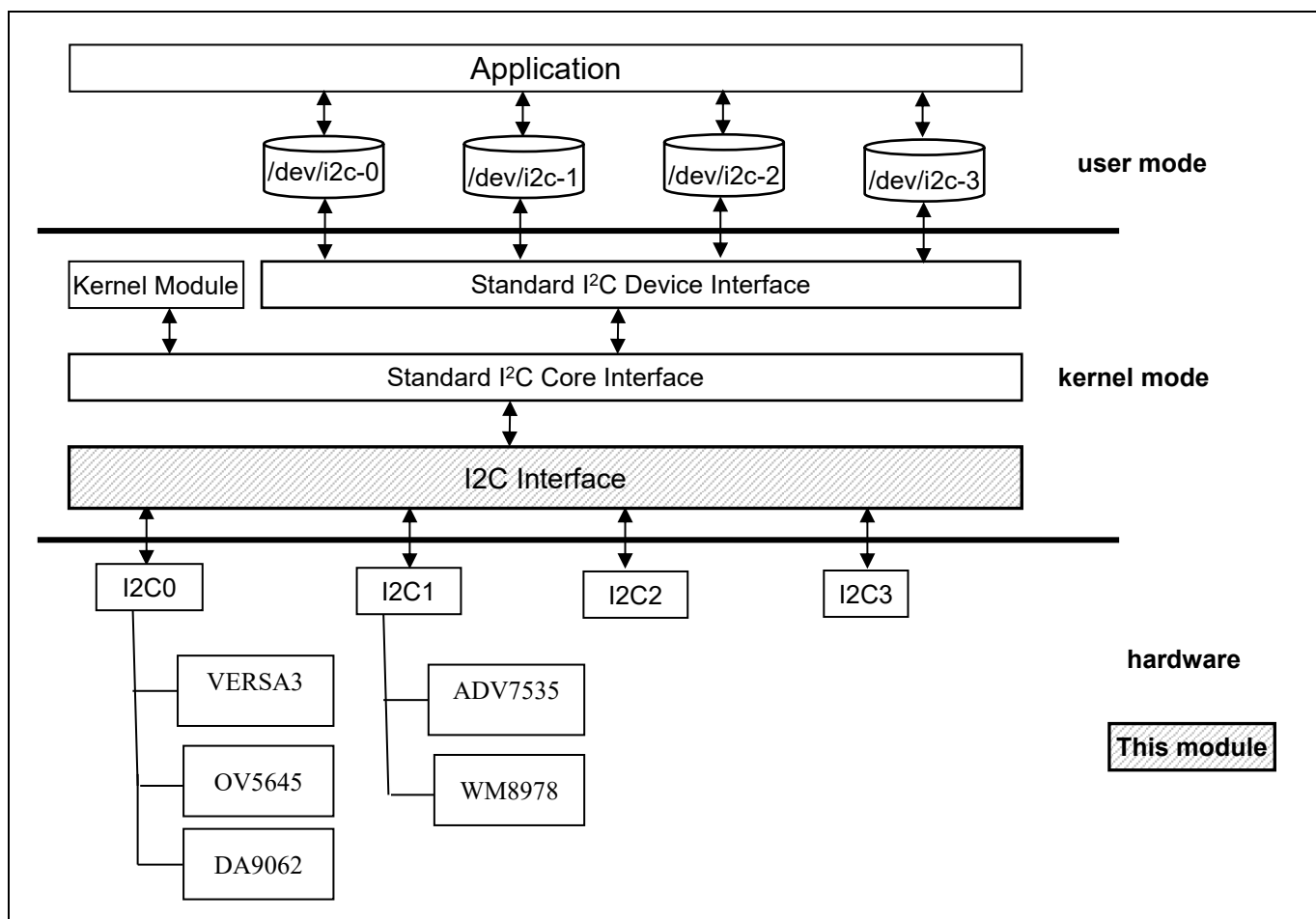


Figure 3-3 I2C Driver Module configuration (RZ/G2UL and RZ/Five)

In the case of RZ/V2N you can control the connected device using the /dev/i2c-0,1,2,3,8.

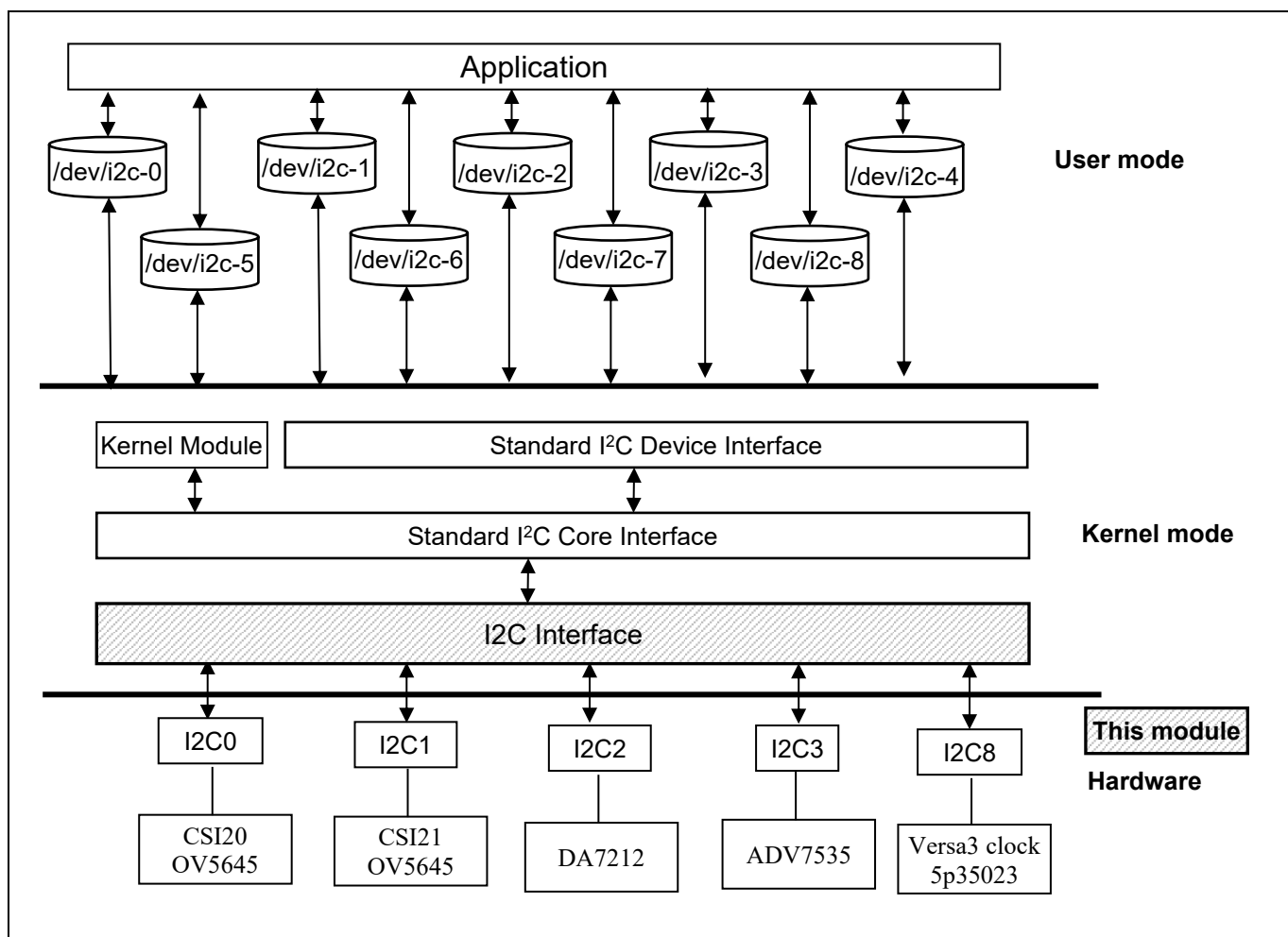


Figure 3-4 I2C Driver Module configuration (RZ/V2N)

In the case of RZ/G3E you can control the connected device using the /dev/i2c-0,1,7,8.

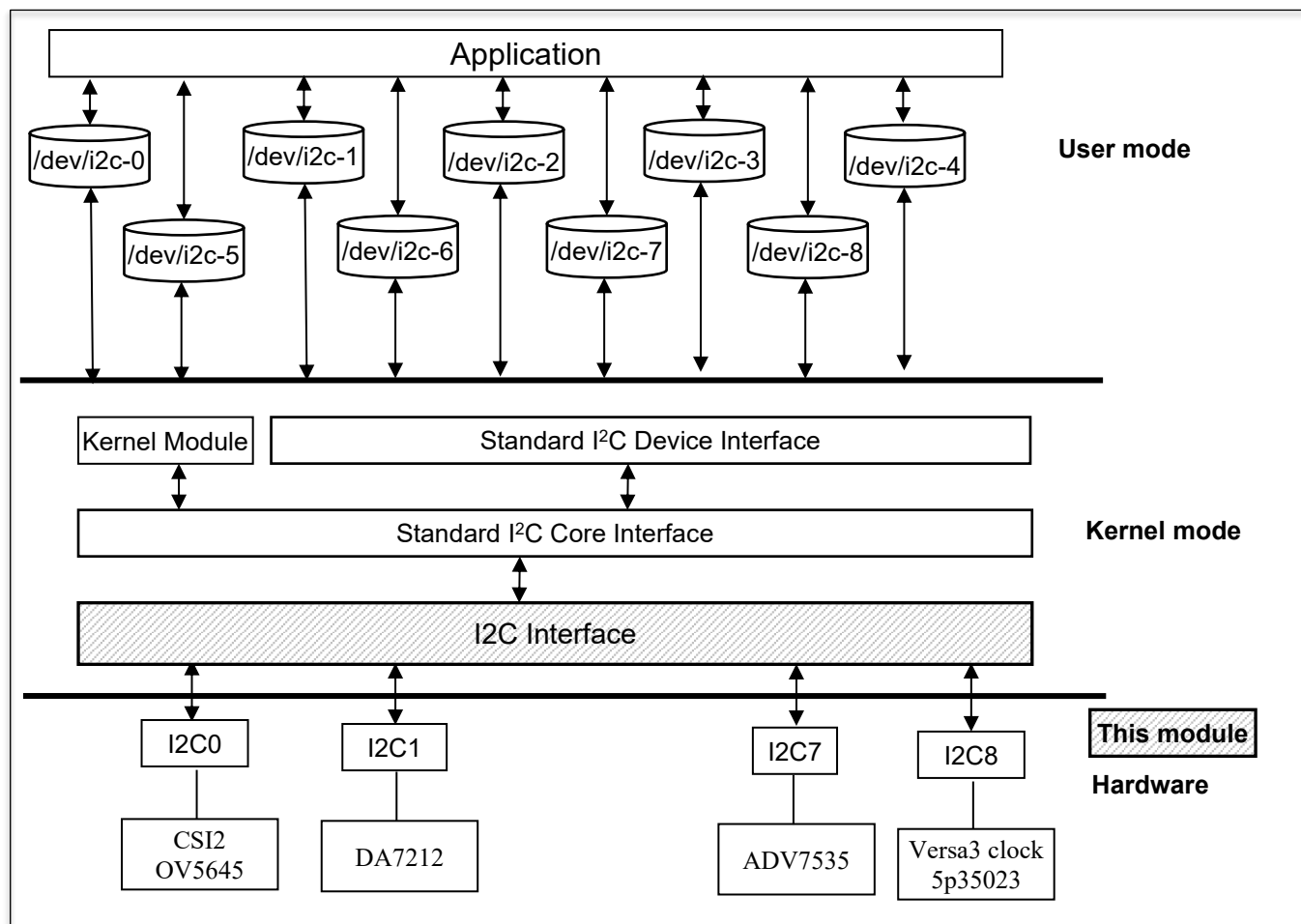


Figure 3-5 I2C Driver Module configuration (RZ/G3E)

3.3 State Transition Diagram

There is no state transition diagram for this module.

4. External Interface

4.1 Device Node

The following table shows the device node of this module.

Table 4-1 I2C device node (RZ/G2L, RZ/G2LC, RZ/G2UL, RZ/V2L, RZ/Five Group)

Channel	Device node	Major number	Minor number
I2C0	/dev/i2c-0	89	0
I2C1	/dev/i2c-1	89	1
I2C2	/dev/i2c-2	89	2
I2C3	/dev/i2c-3	89	3

Table 4-2 I2C device node (RZ/V2N and RZ/G3E group)

Channel	Device node	Major number	Minor number
I2C0	/dev/i2c-0	89	0
I2C1	/dev/i2c-1	89	1
I2C2	/dev/i2c-2	89	2
I2C3	/dev/i2c-3	89	3
I2C6	/dev/i2c-6	89	6
I2C7	/dev/i2c-7	89	7
I2C8	/dev/i2c-8	89	8

4.2 External Function

This section explains in the following format about the functions this module supplies.

[Overview]	Presents an overview of a function.
[Function Name]	Explains the name of the function.
[Calling format]	Explains the format for calling the function.
[Argument]	Explains the argument(s) of the function.
[Return value]	Explains the return value(s) of the function.
[Error number]	Explains the error number(s) of the function.
[Feature]	Explains the features of the function.
[Remark]	Explains points to be noted when using the function.

The following table lists the interface functions in this module, and Standard I2C core Interface.

Table 4-3 System calls

Chapter	Function name	Description	File
4.2.1	i2cdev_open	Open I2C.	drivers/i2c/i2c-dev.c
4.2.2	i2cdev_release	Close I2C.	drivers/i2c/i2c-dev.c
4.2.3	i2cdev_read	Read I2C (8bit data is received).	drivers/i2c/i2c-dev.c
4.2.4	i2cdev_write	Write I2C (8bit data is sent).	drivers/i2c/i2c-dev.c
4.2.5	i2cdev_ioctl	Setting some specific operations of I2C	drivers/i2c/i2c-dev.c

Note: Supported in both v5.10 and v6.1 kernel revision.

Table 4-4 Standard I2C device interface (RZ/G2L, RZ/V2L, RZ/Five, RZ/V2N group, RZ/G3E group)

Chapter	Function name	Description	File
4.2.6	riic_i2c_probe	Probe function to start the driver	drivers/i2c/busses/i2c-riic.c
4.2.7	riic_init_hw	Initialize the settings from the hardware	drivers/i2c/busses/i2c-riic.c
4.2.8	riic_clear_set_bit	Clear all bits which is set	drivers/i2c/busses/i2c-riic.c
4.2.9	riic_func	Choose the specific bus type system	drivers/i2c/busses/i2c-riic.c
4.2.10	riic_xfer	Transfer necessary information	drivers/i2c/busses/i2c-riic.c
4.2.11	riic_xfer_atomic	Transfer using polling	drivers/i2c/busses/i2c-riic.c
4.2.11	riic_tdre_isr	Use TIE interrupt (Transmit interrupt enable)	drivers/i2c/busses/i2c-riic.c
4.2.12	riic_rdrf_isr	Use RIE interrupt (Receive interrupt enable)	drivers/i2c/busses/i2c-riic.c
4.2.13	riic_tend_isr	Determine the transfer is completed	drivers/i2c/busses/i2c-riic.c
4.2.14	riic_stop_isr	Stop the Interrupt	drivers/i2c/busses/i2c-riic.c
4.2.15	riic_bus_barrier	Verify bus availability before initiating an I2C transfer.	drivers/i2c/busses/i2c-riic.c
4.2.16	riic_recover_bus	Implement bus recovery	drivers/i2c/busses/i2c-riic.c
4.2.17	i2c_of_match_device	Match and connect the ID of Device	drivers/i2c/i2c-core-of.c
4.2.18	i2c_of_match_device_sysfs	Adding devices through the i2c interface	drivers/i2c/i2c-core-of.c
4.2.19	of_i2c_register_device	Register the device	drivers/i2c/i2c-core-of.c
4.2.20	of_i2c_register_devices	Register the child devices	drivers/i2c/i2c-core-of.c
4.2.21	of_i2c_get_board_info	Get information from board using	drivers/i2c/i2c-core-of.c
4.2.22	i2c_transfer	Execute a single or combined I2C message	drivers/i2c/i2c-core-base.c
4.2.23	i2c_get_functionality	Return the functionality.	include/linux/i2c.h

Note: Supported in both v5.10 and v6.1 kernel revision.

Function riic_xfer_atomic() is not supported in RZ/V2N Group, RZ/G3E Group.

4.3 Structure

The all below structures of this module are supported on Linux kernel v5.10 and v6.1.

4.3.1 struct i2c_adapter

Table 4-5 Struct i2c_adapter

Structure name	Member			
	Type	Member name	Overview	Note
i2c_adapter	struct module *	owner	The owner of this module	Support both v5.10 and v6.1
	unsigned int	class	The type of I2C device supported by this driver	Support both v5.10 and v6.1
	const struct i2c_algorithm *	algo	The pointer of the algorithm to access the bus	Support both v5.10 and v6.1
	void *	algo_data	The algorithm data	Support both v5.10 and v6.1
	const struct i2c_lock_operations *	lock_ops	Lock operations	Support both v5.10 and v6.1
	struct rt_mutex	bus_lock	The structure specified rt_mutex	Support both v5.10 and v6.1
	struct rt_mutex	mux_lock	The structure specified rt_mutex	Support both v5.10 and v6.1
	int	timeout	Timeout value	Support both v5.10 and v6.1
	int	retries	The retry number	Support both v5.10 and v6.1
	struct device	dev	The adapter device	Support both v5.10 and v6.1
	int	nr	The adapter ID	Support both v5.10 and v6.1
	char	name[48]	The name of I2C device driver	Support both v5.10 and v6.1
	struct completion	dev_released	The structure used to maintain the state of "completion"	Support both v5.10 and v6.1
	struct mutex	userspace_clients_lock	The mutex of client	Support both v5.10 and v6.1
	struct list_head	userspace_clients	The list of client	Support both v5.10 and v6.1
	struct i2c_bus_recovery_info *	bus_recovery_info	The pointer of the information for bus recovery	Support both v5.10 and v6.1
	const struct i2c_adapter_quirks *	quirks	describe flaws of the i2c adapter	Support both v5.10 and v6.1
	struct irq_domain *	host_notify_domain	recovery	Support both v5.10 and v6.1
	struct regulator *	bus_regulator	control bulk regulator	Only support in v6.1

4.3.2 struct i2c_client

Table 4-6 Struct i2c_client

Structure name	Member		
	Type	Member name	Overview
i2c_client	unsigned short	flags	The support function flag of client
	unsigned short	addr	The slave address
	char	name[I2C_NAME_SIZE]	The client name
	struct i2c_adapter *	adapter	The adapter information
	struct device	dev	The driver model device node for the slave
	int	irq	The interrupt number used by the device
	struct list_head	detected	The member of i2c_driver.clients list

4.3.3 struct i2c_board_info

Table 4-7 Struct i2c_board_info

Structure name	Member		
	Type	Member name	Overview
i2c_board_info	char	type[I2C_NAME_SIZE]	The chip type to initialize i2c_client.name
	unsigned short	flags	The flag to initialize i2c_client.flags
	unsigned short	addr	The device address
	void *	platform_data	The Platform Data of the device
	struct dev_archdata *	archdata	The information of board dependent part
	struct device_node *	of_node	The information of the device node
	struct fwnode_handle *	fwnode	Device node supplied by the platform firmware
	const struct property_entry *	properties	Additional device properties for the device
	const struct resource *	resources	Resources associated with the device
	unsigned int	num_resources	Number of resources in the @resources array
	int	irq	The interrupt number used by the device

4.3.4 struct i2c_msg

Table 4-8 Struct i2c_msg

Structure name	Member		
	Type	Member name	Overview
i2c_msg	__u16	addr	The slave address
	__u16	flags	Specify R/W flag
	__u16	len	The message length
	__u8*	buf	The pointer to the message data

4.3.5 struct i2c_rdwr_ioctl_data

Table 4-9 Struct i2c_rdwr_ioctl_data

Structure name	Member		
	Type	Member name	Overview
i2c_rdwr_ioctl_data	struct i2c_msg *	msgs	The pointer to i2c_msg structure
	__u32	nmsgs	A number of i2c_msg

4.3.6 struct i2c_smbus_ioctl_data

Table 4-10 Struct i2c_smbus_ioctl_data

Structure name	Member		
	Type	Member name	Overview
i2c_smbus_ioctl_data	__u8	read_write	Specify R/W flag
	__u8	command	The slave address
	__u32	size	Data type (size)
	union i2c_smbus_data *	data	The pointer to the data

4.3.7 union i2c_smbus_data

Table 4-11 Union i2c_smbus_data

Union name	Member		
	Type	Member name	Overview
i2c_smbus_data	__u8	byte	8bit data buffer
	__u16	word	16bit data buffer
	__u8	block[I2C_SMBUS_BLOCK_MAX + 2]	Block data buffer

4.4 Global Variables and Constants

4.4.1 Global Variables

There are no global variables for this module.

4.4.2 Global Constants

The following table shows the global constants used by standard I2C core.

Table 4-12 List of Global constants

Global Constant Name	Value	Remark
I2C_CLIENT_END	0xfffeU	-
I2C_CLIENT_PEC	0x04	-
I2C_CLIENT_TEN	0x10	-
I2C_CLIENT_WAKE	0x80	-
I2C_FUNC_SMBUS_QUICK	0x00010000	-
I2C_FUNC_SMBUS_READ_BYTE	0x00020000	-
I2C_MODULE_PREFIX	"i2c:"	-
I2C_M_RD	0x0001	-
I2C_M_RECV_LEN	0x0400	-
I2C_M_TEN	0x0010	-
I2C_NAME_SIZE	20	-
I2C_SMBUS_BYTE	1	-
I2C_SMBUS_BYTE_DATA	2	-
I2C_SMBUS_BLOCK_DATA	5	-
I2C_SMBUS_BLOCK_MAX	32	-
I2C_SMBUS_BLOCK_PROC_CALL	7	-
I2C_SMBUS_I2C_BLOCK_DATA	8	-
I2C_SMBUS_PROC_CALL	4	-
I2C_SMBUS_QUICK	0	-
I2C_SMBUS_READ	1	-
I2C_SMBUS_WRITE	0	-
I2C_SMBUS_WORD_DATA	3	-
I2C_RDWR	0x0707	-
I2C_FUNCS	0x0705	-
I2C_SLAVE	0x0703	-
I2C_SMBUS	0x0720	-

4.5 Definitions

4.5.1 Device information in Device Tree

The I2C device properties are showed below.

```
i2c0: i2c@10058000 {
    #address-cells = <1>;
    #size-cells = <0>;
    compatible = "renesas,riic-r9a07g044", "renesas,riic-rz";
    reg = <0 0x10058000 0 0x400>;
    interrupts = <GIC_SPI 350 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 348 IRQ_TYPE_EDGE_RISING>,
                <GIC_SPI 349 IRQ_TYPE_EDGE_RISING>,
                <GIC_SPI 352 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 353 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 351 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 354 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 355 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "tei", "ri", "ti", "spi", "sti",
                    "naki", "ali", "tmoi";
    clocks = <&cpg CPG_MOD R9A07G044_I2C0_PCLK>;
    clock-frequency = <100000>;
    resets = <&cpg R9A07G044_I2C0_MRST>;
    power-domains = <&cpg>;
    status = "disabled";
};
```

Note: All of the information in above device tree is used for both RZ/G2L, RZ/V2L, RZ/G2LC, RZ/G2UL and RZ/Five except: clocks and resets.

```
i2c0: i2c@14400400 {
    #address-cells = <1>;
    #size-cells = <0>;
    compatible = "renesas,riic-r9a09g056";
    reg = <0 0x14400400 0 0x400>;
    interrupts = <GIC_SPI 174 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 507 IRQ_TYPE_EDGE_RISING>,
                <GIC_SPI 506 IRQ_TYPE_EDGE_RISING>,
                <GIC_SPI 176 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 177 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 175 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 178 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 179 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "tei", "ri", "ti", "spi", "sti",
                    "naki", "ali", "tmoi";
    clocks = <&cpg CPG_MOD R9A09G056_ACPU_RIIC0_CKM>;
    clock-frequency = <100000>;
    resets = <&cpg R9A09G056_ACPU_RIIC0_MRST>;
    power-domains = <&cpg>;
    status = "disabled";
};
```

Note: This example is applied for RZ/V2N.

```

i2c0: i2c@14400400 {
    compatible = "renesas,riic-r9a09g047";
    reg = <0 0x14400400 0 0x400>;
    interrupts = <GIC_SPI 174 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 507 IRQ_TYPE_EDGE_RISING>,
                <GIC_SPI 506 IRQ_TYPE_EDGE_RISING>,
                <GIC_SPI 176 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 177 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 175 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 178 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 179 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "tei", "ri", "ti", "spi", "sti",
                    "naki", "ali", "tmoi";
    clocks = <&cpg CPG_MOD 0x94>;
    resets = <&cpg 0x98>;
    power-domains = <&cpg>;
    #address-cells = <1>;
    #size-cells = <0>;
    status = "disabled";
};

```

Note: This example is applied for RZ/G3E.

The I2C device required properties:

compatible:

Must be set to "renesas,riic-r9a07g044" for R9A07G044L (RZ/G2L), R9A07G054L (RZ/V2L), R9A07G044C (RZ/G2LC) as a fallback.

Must be set to "renesas,riic-r9a07g043" for R9A07G043 (RZ/G2UL) as a fallback.

Must be set to "renesas,riic-r9a07g043f" for R9A07G043F (RZ/Five) as a fallback.

Must be set to "renesas,riic-r9a09g056" for R9A09G056 (RZ/V2N).

Must be set to "renesas,riic-r9a09g047" for R9A09G047 (RZ/G3E).

reg:

Base address and length of the memory resource used by the I2C CH0.

clocks:

The 1st cell is a node or label of CPG clock to be used.

The 2nd cell must be set to CPG_MOD.

The 3rd cell must be set to R9A07G044_I2C0_PCLK for RZ/G2L, RZ/V2L, RZ/G2LC, R9A07G043_I2C0_PCLK for RZ/G2UL, R9A07G043F_I2C0_PCLK for RZ/Five, R9A09G056_I2C0_PCLK for RZ/V2N, R9A09G047_I2C0_PCLK for RZ/G3E.

Clocks-frequency:

Must be set 100000 as default.

address-cells:

Must be set to 1.

Size-cells:

Must be set to 0.

interrupt:

<GIC_SPI 350 IRQ_TYPE_LEVEL_HIGH>	Transmission completed
<GIC_SPI 348 IRQ_TYPE_LEVEL_HIGH>	Receive data full
<GIC_SPI 349 IRQ_TYPE_LEVEL_HIGH>	Transmit data empty
<GIC_SPI 352 IRQ_TYPE_LEVEL_HIGH>	Detection of a stop condition
<GIC_SPI 353 IRQ_TYPE_LEVEL_HIGH>	Detection of a start condition
<GIC_SPI 351 IRQ_TYPE_LEVEL_HIGH>	Reception of a NACK
<GIC_SPI 354 IRQ_TYPE_LEVEL_HIGH>	Arbitration lost
<GIC_SPI 355 IRQ_TYPE_LEVEL_HIGH>	Timeout

Note: +32 into each Interrupt ID of RZ/Five.

<GIC_SPI 174 IRQ_TYPE_LEVEL_HIGH>	Transmission completed
<GIC_SPI 507 IRQ_TYPE_LEVEL_HIGH>	Receive data full
<GIC_SPI 506 IRQ_TYPE_LEVEL_HIGH>	Transmit data empty
<GIC_SPI 176 IRQ_TYPE_LEVEL_HIGH>	Detection of a stop condition
<GIC_SPI 177 IRQ_TYPE_LEVEL_HIGH>	Detection of a start condition
<GIC_SPI 175 IRQ_TYPE_LEVEL_HIGH>	Reception of a NACK
<GIC_SPI 178 IRQ_TYPE_LEVEL_HIGH>	Arbitration lost
<GIC_SPI 179 IRQ_TYPE_LEVEL_HIGH>	Timeout

Note: Interrupt IDs of RZ/V2N, RZ/G3E.

Interrupt-name:

Name of each interrupt source, driver get interrupt by interrupt-name

reset:

release reset state: R9A07G044_I2C0_MRST is used for R9A07G044L (RZ/G2L), R9A07G054L (RZ/V2L), R9A07G044C (RZ/G2LC) and R9A07G043_I2C0_MRST is used for R9A07G043 (RZ/G2UL),

R9A07G043F_I2C0_MRST is used for R9A07G043F (RZ/Five).

R9A09G056_I2C0_MRST is used for R9A09G056 (RZ/V2N).

R9A09G047_I2C0_MRST is used for R9A09G047 (RZ/G3E).

power domain:

Must be set to always on.

5. Integration

5.1 Directory Configuration

The directory configuration is shown below.

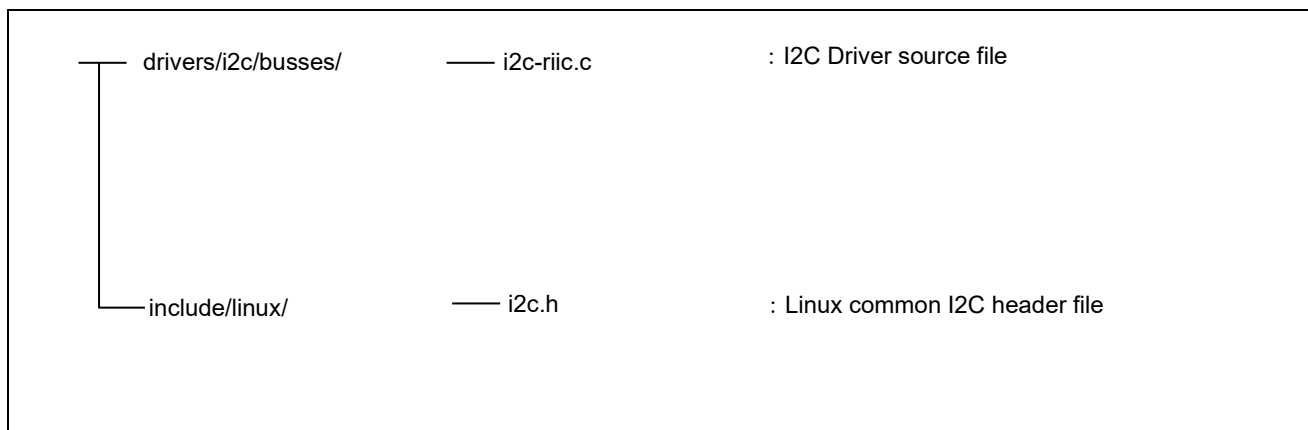


Figure 5-1 Directory configuration

5.2 Integration Procedure

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

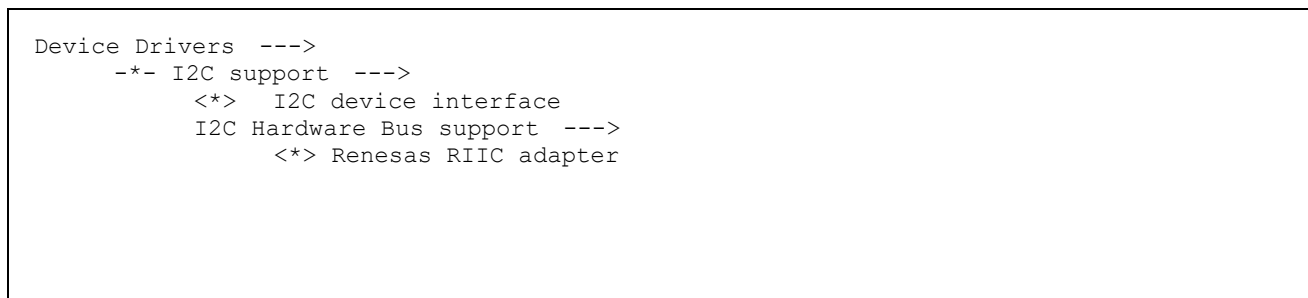


Figure 5-2 Kernel configuration

Lists the kernel config symbols to support I2C for RZ/G2L, RZ/V2L, RZ/V2N, RZ/G3E Group, RZ/Five.

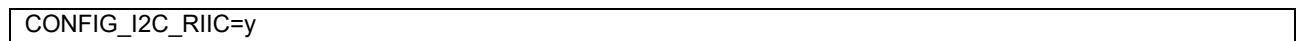


Figure 5-3 Kernel config symbols

5.3 Option Setting

5.3.1 Module Parameters

5.3.1.1 DT

We can set I2C transfer speed 400000 or 1000000 to "clock-frequency" of device tree file in arch/arm64/boot/dts/ directory. No setting means 100000 Hz.

```
&i2c0 {
    status = "okay";
    clock-frequency = <100000>;
    ...
}

&i2c1 {
    status = "okay";
    clock-frequency = <100000>;
    ...
}

&i2c3 {
    status = "okay";
    clock-frequency = <400000>;
    ...
}
```

Figure 5-4 Example of setting I2C transfer speed

5.3.1.2 Guideline to support Slave mode testing

List the board support I2C slave for RZ/G2L, RZ/V2L, RZ/V2N, RZ/G3E Group, RZ/Five.

Step to test i2c slave:

- Connect slave and master in the same I2C bus lines.
- Add a new virtual eeprom I2C device in Slave board(used I2C0) by running command:

```
$ echo slave-24c02 0x1064 > /sys/bus/i2c/devices/i2c-0/new_device
```

- Get detection for this device from master board (i2c3) by running command:

```
root@smarc-rzg2l:~# i2cdetect -y -r 3
  0 1 2 3 4 5 6 7 8 9 a b c d e f
00:  -----
10:  ----- UU -----
20:  -----
30:  -----
40:  -----
50:  -----
60:  ----- 64 ----- UU -----
70:  -----
```

- ⇒ Device 64 is detected
- In master board use i2cget/i2cset to read/write the data

```
root@smarc-rzg2l:~# i2cset -f -y 3 0x64 0xaa 0x10    => Set 0x10 to 0xaa address
root@smarc-rzg2l:~# i2cget -f -y 3 0x64 0xaa        => Confirm the value of 0xaa
0x10
```

```

root@smarc-rzg2l:~# i2cset -f -y 3 0x64 0x00 0x10
root@smarc-rzg2l:~# i2cset -f -y 3 0x64 0x10 0x20
root@smarc-rzg2l:~# i2cget -f -y 3 0x64 0x00
0x10
root@smarc-rzg2l:~# i2cget -f -y 3 0x64 0x10
0x20

```

⇒ Can transmit and receive true value between master and slave board

Step to test multiple slave:

- Connect slave and master in the same I2C bus lines.
- Add a new virtual eeprom I2C device in Slave board(used I2C0) by running command:

```

$ echo slave-24c02 0x1064 > /sys/bus/i2c/devices/i2c-0/new_device
$ echo slave-24c02 0x1065 > /sys/bus/i2c/devices/i2c-0/new_device
$ echo slave-24c02 0x1066 > /sys/bus/i2c/devices/i2c-0/new_device

```

- Get detection for this device from master board (i2c3) by running command:

```

root@smarc-rzg2l:~# i2cdetect -y -r 3
    0 1 2 3 4 5 6 7 8 9 a b c d e f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
40:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
50:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60:  --  --  --  -- 64 65 66 --  --  --  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --

```

- ⇒ Device 64, 65, 66 is detected
- In master board use i2cget/i2cset to read/write the data

```

root@smarc-rzg2l:~# i2cset -f -y 3 0x64 0xaa 0x10    => Set 0x10 to 0xaa address
root@smarc-rzg2l:~# i2cget -f -y 3 0x64 0xaa        => Confirm the value of 0xaa
0x10
root@smarc-rzg2l:~# i2cset -f -y 3 0x64 0x00 0x10
root@smarc-rzg2l:~# i2cset -f -y 3 0x65 0x00 0x20
root@smarc-rzg2l:~# i2cset -f -y 3 0x66 0x00 0x30
root@smarc-rzg2l:~# i2cget -f -y 3 0x64 0x00
0x10
root@smarc-rzg2l:~# i2cget -f -y 3 0x65 0x00
0x20
root@smarc-rzg2l:~# i2cget -f -y 3 0x66 0x00
0x30

```

⇒ Can transmit and receive true value between master and multi slave device.

5.3.2 Kernel Parameters

There are no module parameters.

5.3.3 Device tree bindings

Demonstrates an example of device tree setting which can be found in
Documentation/devicetree/bindings/i2c/i2c-riic.txt

REVISION HISTORY	Linux Interface Specification Device Driver I2C User's Manual: Software
-------------------------	--

Rev.	Date	Description	
		Page	Summary
0.50	Apr. 30, 2021	—	First Edition issued
1.0	Jul. 15, 2021	—	No modification, keep version to keep consistent with other documents
1.1	Sep. 15, 2021	—	Merge RZ/G2L driver manual with RZ/V2L
1.2	Feb. 15, 2022	—	Add RZ/G2UL, RZ/G2LC device
1.3	Mar. 31, 2022	17,18	Update information for device-tree
1.4	May. 31, 2022	—	No modification, change version to keep consistent with other documents
1.5	Jun. 24, 2022	—	Add support RZ/Five
1.6	Sep. 15, 2022	—	No modification, change version to keep consistent with other documents
1.7	Dec. 15, 2022	—	No modification, change version to keep consistent with other documents
1.8	Mar. 15, 2023	—	No modification, change version to keep consistent with other documents
1.9	Mar. 31, 2024	1, 20	Add support Fast Mode Plus interface mode
1.10	Mar. 31, 2025	1	Update table 1.2-2
		12	Update table 4.2-2
		20, 21	Add guidelines to support Slave mode testing
1.11	May. 30, 2025	1	Add MPU information support for both kernel versions v5.10 and v6.1. Update table 1.2-1
		2	Update table 1.2-3
		12	Update table 4.2-1 Update table 4.2-2
		13	Update table 4.3-1
		19	Update figure 5-3
1.12	Jun. 30, 2025	—	Add support RZ/V2N
1.13	Jul. 22, 2025	—	Add support RZ/G3E

Linux Interface Specification Device Driver I2C
User's Manual: Software

Publication Date: Rev. 1.13 Jul. 22, 2025

Published by: Renesas Electronics Corporation

RZ/G2L Group, RZ/V2L Group, RZ/V2N Group,
RZ/G3E Group and RZ/Five



Renesas Electronics Corporation