

Linux Interface Specification Device Driver CANFD

User's Manual: Software

RZ/G2L Group, RZ/V2L Group, RZ/V2N Group,
RZ/V2H Group, RZ/G3E Group, RZ/G3S Group,
and RZ/Five 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 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. 1.14 Mar 27, 2026)

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/V2H Group, RZ/V2N Group, RZ/G3E Group and RZ/G3S 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/V2N Group User's Manual: Hardware	---
		RZ/V2H Group User's Manual: Hardware	---
		RZ/Five Group User's Manual: Hardware	---
		RZ/G3E Group User's Manual: Hardware	---
		RZ/G3S 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 CANFD Device Driver	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
ACIA	Asynchronous Communications Interface Adapter
bps	bits per second
CRC	Cyclic Redundancy Check
DMA	Direct Memory Access
DMAC	Direct Memory Access Controller
GSM	Global System for Mobile Communications
Hi-Z	High Impedance
IEBus	Inter Equipment Bus
I/O	Input/Output
IrDA	Infrared Data Association
LSB	Least Significant Bit
MSB	Most Significant Bit
NC	Non-Connect
PLL	Phase Locked Loop
PWM	Pulse Width Modulation
SFR	Special Function Register
SIM	Subscriber Identity Module
UART	Universal Asynchronous Receiver/Transmitter
VCO	Voltage Controlled Oscillator

Table of Contents

1. Overview	1
1.1 Overview	1
1.2 Function	1
1.3 CANFD devices	1
1.4 Connected Port	2
1.5 Reference	3
1.5.1 Standard	3
1.5.2 Related documents	3
1.6 Restrictions	3
2. Terminology	4
3. Operating Environment	5
3.1 Hardware Environment	5
3.2 Module Configuration	6
3.3 State Transition Diagram	6
4. External Interface	7
4.1 Device node	7
4.2 Sysfs interface specification	8
4.3 Definitions	8
4.3.1 Device information in Device Tree	8
5. Integration	24
5.1 Directory Configuration	24
5.2 Integration Procedure	24
5.2.1 Kernel Configuration	24
5.2.2 Enable Pin-function control for CANFD driver.	24
5.2.3 Enable CANFD driver for each channel.	30
5.3 Option Setting	36
5.3.1 Module Parameters	36
5.3.2 Kernel Parameters	36
5.3.3 Device tree bindings	36
5.3.4 Userspace API	36

1. Overview

1.1 Overview

This manual explains the driver module that controls the CANFD Interface (RS-CANFD) controller in the CANFD on RZ/G2L Group, RZ/V2L Group, RZ/V2H Group, RZ/V2N Group, RZ/G3E Group, RZ/G3S Group and RZ/Five Group.

Note: Currently, the device can support both Linux kernel versions with the information below:

- v5.10: RZ/G2L Group, RZ/V2L Group, RZ/G3S Group and RZ/Five Group.
- v6.1: RZ/G2L Group, RZ/V2L Group, RZ/G3E Group, RZ/V2H Group, RZ/G3S Group and RZ/V2N Group.

1.2 Function

This module supports the following functions by controlling RS-CANFD on RZ/G2L, RZ/G2LC, RZ/G2UL, RZ/V2L, RZ/V2H, RZ/V2N, RZ/G3E, RZ/G3S and RZ/Five.

- CAN FD mode supports both Classical CAN, FD-only mode and CAN-FD mode (Dual mode) formats. The controller supports ISO 11898-1:2015 CAN FD format.
- Transmission and reception of CANFD frame.
- Interval transmission function: Transmit messages at configurable intervals.
- Transmit queue function: Transmits all stored messages according to the ID priority.
- Error status monitoring.
- Classical CAN mode communication speed is up to 1 Mbps maximum.
- FD-only mode communication speed is up to 1Mbps nominal bitrate and 8Mbps Data bit rate.
- CAN-FD mode (Dual mode) can use both Classical CAN mode and FD-only mode.

The information of available modes in each device:

- Classical CAN mode: RZ/G2L, RZ/G2LC, RZ/G2UL, RZ/V2L, RZ/Five, RZ/V2H, RZ/V2N, RZ/G3E.
- FD-only mode: RZ/V2H, RZ/V2N, RZ/G3E.
- CAN-FD mode (Dual mode): RZ/G2L, RZ/G2LC, RZ/G2UL, RZ/V2L, RZ/Five, RZ/V2H, RZ/V2N, RZ/G3E, RZ/G3S.

1.3 CANFD devices

Supported CAN Transceiver device of this module is as follows.

Table 1-1 Supported CANFD transceiver Devices

Vendor	Product	Interface	Note
-	TCAN1046V-Q1	CAN transceiver	Support CAN transceiver that meets the physical layer requirement of the ISO 11898 high-speed CAN specification

1.4 Connected Port

Table 1-2 Connector for RZ/G2L Group, RZ/V2L, and RZ/Five Promotion Board.

Channel	Connector
can0	CN15
can1	CN16

Table 1-3 Connector for RZ/V2N Evaluation Board Kit.

Channel	Connector
can0	CN1
can1	CN1
can2	CN1
can3	CN1

Table 1-4 Connector for RZ/V2H Evaluation Board Kit.

Channel	Connector
can0	CN1
can1	CN1

Table 1-5 Connector for RZ/G3E Smarc Evaluation Board Kit.

Channel	Connector
can0	CAN0
can1	CAN1

Table 1-6 Connector for RZ/G3S Smarc Evaluation Board Kit.

Channel	Connector
can0	CAN0
can1	CAN1

1.5 Reference

1.5.1 Standard

There are no reference documents on standards.

1.5.2 Related documents

The following table shows the document related to this module.

Table 1-7 Related document

Number	Issue	Title	Edition	Date
-	-	-	-	-

1.6 Restrictions

There is no restriction in this module.

2. Terminology

The following table shows the terminology related to this module.

Table 2-1 Terminology

Terms	Explanation
Classical CAN mode	Only classical CAN frames are handled
FD-only mode	Only CAN FD frames are handled
CAN FD mode	Both the classical CAN frames and CANFD frames are handled

3. Operating Environment

3.1 Hardware Environment

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

Table 3-1 Hardware environment

Name	Version
RZ/G2L Evaluation Board Kit	RTK9744L23S01000BE
RZ/G2LC Evaluation Board Kit	RTK9744C22S01000BE
RZ/G2UL Evaluation Board Kit	RTK9743U11S01000BE
RZ/V2L Evaluation Board Kit	RTK9754L23S01000BE
RZ/Five Evaluation Board Kit	RTK9743F01S01000BE
RZ/G3S Evaluation Board Kit.	RTK9845S33C01000BE
RZ/V2N Evaluation Board Kit V1.0	RTK0EF0186C03000BJ
RZ/V2N Evaluation Board Kit V2.0	RTK0EF0186C03001BJ
RZ/V2H Evaluation Board Kit	RTK0EF0168C04000BJ
RZ/G3E Evaluation Board Kit.	RTK9947E57S01000BE

3.2 Module Configuration

The following figure shows the configuration of this module.

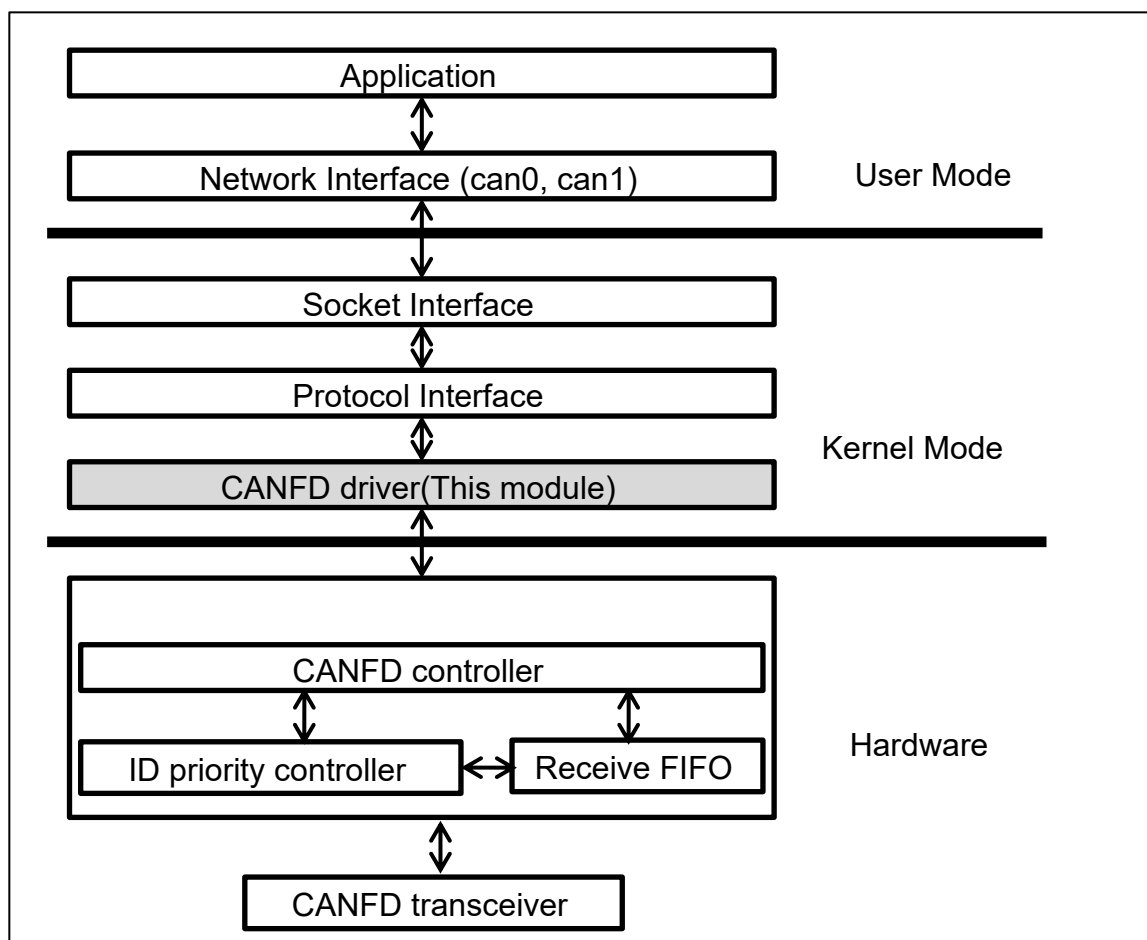


Figure 3-1 RS-CANFD Module Configuration

3.3 State Transition Diagram

There is no state transition diagram for this module.

4. External Interface

4.1 Device node

The following tables show the device nodes of this module.

If hardware settings enable both channel 0 and channel 1 (RZ/G2L, RZ/V2L, RZ/G2UL, RZ/G3S, and RZ/Five):

Table 4-1 CANFD device node (RZ/G2L, RZ/V2L, RZ/G2UL, RZ/G3S and RZ/Five)

Channel	Device node	Device name
channel 0	can0	can0
channel 1	can1	can1

If hardware settings only enable channel 1 (RZ/G2LC):

Table 4-2 CANFD device node RZ/G2LC

Channel	Device node	Device name
channel 1	can1	can1

Table 4-3 CANFD device node RZ/V2H, RZ/V2N and RZ/G3E

channel	device node	device name
channel 0	can0	can0
channel 1	can1	can1
channel 2	can2	can2
channel 3	can3	can3
channel 4	can4	can4
channel 5	can5	can5

4.2 Sysfs interface specification

Please refer <https://www.kernel.org/doc/Documentation/networking/can.txt> to know how to control CANFD driver correctly in user-space.

4.3 Definitions

4.3.1 Device information in Device Tree

The RS-CANFD device properties are shown below.

```
canfd: can@10050000 {
    compatible = "renesas,r9a07g044-canfd", "renesas,rzg2l-canfd";
    reg = <0 0x10050000 0 0x8000>;
    interrupts = <GIC_SPI 426 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 427 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 422 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 424 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 428 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 423 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 425 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 429 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "g_err", "g_recc",
        "ch0_err", "ch0_rec", "ch0_trx",
        "ch1_err", "ch1_rec", "ch1_trx";
    clocks = <&cpg CPG_MOD R9A07G044_CANFD_PCLK>,
        <&cpg CPG_CORE R9A07G044_CLK_P0_DIV2>,
        <&can_clk>;
    clock-names = "fck", "canfd", "can_clk";
    assigned-clocks = <&cpg CPG_CORE R9A07G044_CLK_P0_DIV2>;
    assigned-clock-rates = <50000000>;
    resets = <&cpg R9A07G044_CANFD_RSTP_N>,
        <&cpg R9A07G044_CANFD_RSTC_N>;
    reset-names = "rstp_n", "rstc_n";
    power-domains = <&cpg>;
    status = "disabled";

    channel0 {
        status = "disabled";
    };
    channel1 {
        status = "disabled";
    };
};
```

Figure 4-1 CANFD Node Example for RZ/G2L

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

The RS-CANFD device required properties:

compatible:

Must be set

"renesas,r9a07g044-canfd" for R9A07G044L (RZ/G2L), R9A07G054L (RZ/V2L), R9A07G044C (RZ/G2LC),

"renesas,r9a07g043-canfd" for R9A07G043U (RZ/G2UL),

"renesas,r9a07g043f-canfd" for R9A07G043F (RZ/Five).

reg:

Base address and length of the memory resource used by the CANFD driver.

interrupt:

<GIC_SPI 426 IRQ_TYPE_LEVEL_HIGH>	CAN global error interrupt
<GIC_SPI 422 IRQ_TYPE_LEVEL_HIGH>	CAN0 error interrupt
<GIC_SPI 428 IRQ_TYPE_LEVEL_HIGH>	CAN0 transmit interrupt
<GIC_SPI 423 IRQ_TYPE_LEVEL_HIGH>	CAN1 error interrupt
<GIC_SPI 429 IRQ_TYPE_LEVEL_HIGH>	CAN1 transmit interrupt
<GIC_SPI 424 IRQ_TYPE_LEVEL_HIGH>	CAN0 transmit/receive FIFO receive completion interrupt
<GIC_SPI 425 IRQ_TYPE_LEVEL_HIGH>	CAN1 transmit/receive FIFO receive completion interrupt
<GIC_SPI 427 IRQ_TYPE_LEVEL_HIGH>	CAN receive FIFO interrupt

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

clocks:

slot 1:

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_CANFD_PCLK for (RZ/G2L, RZ/V2L, RZ/G2LC), R9A07G043_CANFD_PCLK for RZ/G2UL, R9A07G043F_CANFD_PCLK for RZ/Five.

slot 2:

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_CLK_P0_DIV2 for (RZ/G2L, RZ/V2L, RZ/G2LC), R9A07G043_CLK_P0_DIV2 for RZ/G2UL, R9A07G043F_CLK_P0_DIV2 for RZ/Five

reset:

slot 1:

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

The 2nd cell must be set to R9A07G044_CANFD_RSTP_N for (RZ/G2L, RZ/V2L, RZ/G2LC) R9A07G043_CANFD_RSTP_N for RZ/G2UL, R9A07G043F_CANFD_RSTP_N for RZ/Five

slot 2:

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

The 2rd cell must be set to R9A07G044_CANFD_RSTC_N for (RZ/G2L, RZ/V2L, RZ/G2LC), R9A07G043_CANFD_RSTC_N for RZ/G2UL, R9A07G043F_CANFD_RSTC_N for RZ/Five

Power domain:

Must be set to always on.

Interrupt-names:

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

Clock-names:

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

reset-names:

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

Please set “output-low” to enable pin standby input for mode control in device tree file *arch/arm64/boot/dts/renesas/rzg2l-smarc-pinfunction.dtsi* for RZ/G2L, RZ/G2LC, and RZ/V2L.

```
can0_stb {
    gpio-hog;
    gpios = <RZG2L_GPIO(42, 2) GPIO_ACTIVE_HIGH>;
    output-low;
    line-name = "can0_stb";
};
can1_stb {
    gpio-hog;
    gpios = <RZG2L_GPIO(42, 3) GPIO_ACTIVE_HIGH>;
    output-low;
    line-name = "can1_stb";
};
```

Figure 4-2 Set output-low to enable pin standby for RZ/G2L, RZ/V2L

```
#if SW_RSPI_CAN
/* SW8 should be at position 2->3 so that GPIO9_CAN1_STB line is activated */
can1-stb {
    gpio-hog;
    gpios = <RZG2L_GPIO(44, 3) GPIO_ACTIVE_HIGH>;
    output-low;
    line-name = "can1_stb";
};
```

Figure 4-3 Set output-low to enable pin standby input for RZ/G2LC

Please set “output-low” to enable pin standby input for mode control in device tree file in *arch/arm64/boot/dts/renesas/rzg2ul-smarc.dtsi* for RZ/G2UL, and *arch/riscv/boot/dts/renesas/rzfive-smarc.dtsi* for RZ/Five

```
#if (SW_ET0_EN_N)
    canfd0_en {
        gpio-hog;
        gpios = <RZG2L_GPIO(2, 2) GPIO_ACTIVE_HIGH>;
        output-low;
        line-name = "canfd0_en";
    };
    canfd1_en {
        gpio-hog;
        gpios = <RZG2L_GPIO(2, 3) GPIO_ACTIVE_HIGH>;
        output-low;
        line-name = "canfd1_en";
    };
};
#endif
```

Figure 4-4 Set output-high to pull up pin standby input for RZ/G2UL

The CAN-FD of RZ/V2N device properties are shown below.

```

canfd: can@12440000 {
    compatible = "renesas,r9a09g047-canfd";
    reg = <0 0x12440000 0 0x40000>;
    interrupts = <GIC_SPI 709 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 710 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 697 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 703 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 711 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 698 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 704 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 712 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 699 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 705 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 713 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 700 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 706 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 714 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 701 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 707 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 715 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 702 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 708 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 716 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "g_err", "g_recc",
        "ch0_err", "ch0_rec", "ch0_trx",
        "ch1_err", "ch1_rec", "ch1_trx",
        "ch2_err", "ch2_rec", "ch2_trx",
        "ch3_err", "ch3_rec", "ch3_trx",
        "ch4_err", "ch4_rec", "ch4_trx",
        "ch5_err", "ch5_rec", "ch5_trx";
    clocks = <&cpwg CPG_MOD 0x9c>,
        <&cpwg CPG_MOD 0x9d>,
        <&cpwg CPG_MOD 0x9e>;
    clock-names = "fck", "clk_ram", "can_clk";
    resets = <&cpwg 0xa1>,
        <&cpwg 0xa2>;
    reset-names = "rstp_n", "rstc_n";
    power-domains = <&cpwg>;
    status = "disabled";

    channel0 {
        status = "disabled";
    };
    channel1 {
        status = "disabled";
    };
    channel2 {
        status = "disabled";
    };
    channel3 {
        status = "disabled";
    };
    channel4 {
        status = "disabled";
    };
    channel5 {
        status = "disabled";
    };
};

```

Figure 4-5 CANFD Node Example for RZ/V2N

Note: Example of CAN-FD device tree setting for RZ/V2N.

The CAN-FD device required properties:

compatible:

Must be set

"renesas,r9a09g047-canfd" for R9A09G056 (RZ/V2N).

reg:

Base address and length of the memory resource used by the CANFD driver.

interrupt:

<GIC_SPI 709 IRQ_TYPE_LEVEL_HIGH>	CAN global error interrupt
<GIC_SPI 710 IRQ_TYPE_LEVEL_HIGH>	CAN RX FIFO interrupt
<GIC_SPI 697 IRQ_TYPE_LEVEL_HIGH>	CAN0 error interrupt
<GIC_SPI 703 IRQ_TYPE_LEVEL_HIGH>	CAN0 common RX FIFO or TXQ interrupt
<GIC_SPI 711 IRQ_TYPE_LEVEL_HIGH>	CAN0 TX interrupt
<GIC_SPI 698 IRQ_TYPE_LEVEL_HIGH>	CAN1 error interrupt
<GIC_SPI 704 IRQ_TYPE_LEVEL_HIGH>	CAN1 common RX FIFO or TXQ interrupt
<GIC_SPI 712 IRQ_TYPE_LEVEL_HIGH>	CAN1 TX interrupt
<GIC_SPI 699 IRQ_TYPE_LEVEL_HIGH>	CAN2 error interrupt
<GIC_SPI 705 IRQ_TYPE_LEVEL_HIGH>	CAN2 common RX FIFO or TXQ interrupt
<GIC_SPI 713 IRQ_TYPE_LEVEL_HIGH>	CAN2 TX interrupt
<GIC_SPI 700 IRQ_TYPE_LEVEL_HIGH>	CAN3 error interrupt
<GIC_SPI 706 IRQ_TYPE_LEVEL_HIGH>	CAN3 common RX FIFO or TXQ interrupt
<GIC_SPI 714 IRQ_TYPE_LEVEL_HIGH>	CAN3 TX interrupt
<GIC_SPI 701 IRQ_TYPE_LEVEL_HIGH>	CAN4 error interrupt
<GIC_SPI 707 IRQ_TYPE_LEVEL_HIGH>	CAN4 common RX FIFO or TXQ interrupt
<GIC_SPI 715 IRQ_TYPE_LEVEL_HIGH>	CAN4 TX interrupt
<GIC_SPI 702 IRQ_TYPE_LEVEL_HIGH>	CAN5 error interrupt
<GIC_SPI 708 IRQ_TYPE_LEVEL_HIGH>	CAN5 common RX FIFO or TXQ interrupt
<GIC_SPI 716 IRQ_TYPE_LEVEL_HIGH>	CAN5 TX interrupt

clocks:

slot 1:

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 CANFD PCLK CLK index 0x9c for RZ/V2N .

slot 2:

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 CANFD RAM CLK index 0x9d for RZ/V2N.

slot 3:

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 CANFD CLKC CLK 0x9e for RZ/V2N.

reset:

slot 1:

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

The 2rd cell must be set to CANFD RSTP_N reset index 0xa1 for RZ/V2N.

slot 2:

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

The 2rd cell must be set to CANFD RSTC_N reset index 0xa2 for RZ/V2N.

power domain:

Must be set to always on.

Interrupt-names:

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

clock-names:

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

reset-names:

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

The CAN-FD of RZ/V2H device properties are shown below.

```
canfd: can@12440000 {
    compatible = "renesas,r9a09g047-canfd";
    reg = <0 0x12440000 0 0x40000>;
    interrupts = <GIC_SPI 709 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 710 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 697 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 703 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 711 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 698 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 704 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 712 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 699 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 705 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 713 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 700 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 706 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 714 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 701 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 707 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 715 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 702 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 708 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 716 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "g_err", "g_recc",
        "ch0_err", "ch0_rec", "ch0_trx",
        "ch1_err", "ch1_rec", "ch1_trx",
        "ch2_err", "ch2_rec", "ch2_trx",
        "ch3_err", "ch3_rec", "ch3_trx",
        "ch4_err", "ch4_rec", "ch4_trx",
        "ch5_err", "ch5_rec", "ch5_trx";
    clocks = <&cpg CPG_MOD 0x9c>,
        <&cpg CPG_MOD 0x9d>,
        <&cpg CPG_MOD 0x9e>;
    clock-names = "fck", "ram_clk", "can_clk";
    resets = <&cpg 0xa1>, <&cpg 0xa2>;
    reset-names = "rstp_n", "rstc_n";
    power-domains = <&cpg>;
    status = "disabled";

    channel0 {
        status = "disabled";
    };
    channel1 {
        status = "disabled";
    };
    channel2 {
        status = "disabled";
    };
    channel3 {
        status = "disabled";
    };
    channel4 {
        status = "disabled";
    };
    channel5 {
        status = "disabled";
    };
};
```

Figure 4-6 CANFD Node Example for RZ/V2H

Note: Example of CAN-FD device tree setting for RZ/V2H.

The CAN-FD device required properties:

compatible:

Must be set

"renesas, r9a09g047-canfd" for R9A09G057 (RZ/V2H).

reg:

Base address and length of the memory resource used by the CANFD driver.

interrupt:

<GIC_SPI 709 IRQ_TYPE_LEVEL_HIGH>	CAN global error interrupt
<GIC_SPI 710 IRQ_TYPE_LEVEL_HIGH>	CAN RX FIFO interrupt
<GIC_SPI 697 IRQ_TYPE_LEVEL_HIGH>	CAN0 error interrupt
<GIC_SPI 703 IRQ_TYPE_LEVEL_HIGH>	CAN0 common RX FIFO or TXQ interrupt
<GIC_SPI 711 IRQ_TYPE_LEVEL_HIGH>	CAN0 TX interrupt
<GIC_SPI 698 IRQ_TYPE_LEVEL_HIGH>	CAN1 error interrupt
<GIC_SPI 704 IRQ_TYPE_LEVEL_HIGH>	CAN1 common RX FIFO or TXQ interrupt
<GIC_SPI 712 IRQ_TYPE_LEVEL_HIGH>	CAN1 TX interrupt
<GIC_SPI 699 IRQ_TYPE_LEVEL_HIGH>	CAN2 error interrupt
<GIC_SPI 705 IRQ_TYPE_LEVEL_HIGH>	CAN2 common RX FIFO or TXQ interrupt
<GIC_SPI 713 IRQ_TYPE_LEVEL_HIGH>	CAN2 TX interrupt
<GIC_SPI 700 IRQ_TYPE_LEVEL_HIGH>	CAN3 error interrupt
<GIC_SPI 706 IRQ_TYPE_LEVEL_HIGH>	CAN3 common RX FIFO or TXQ interrupt
<GIC_SPI 714 IRQ_TYPE_LEVEL_HIGH>	CAN3 TX interrupt
<GIC_SPI 701 IRQ_TYPE_LEVEL_HIGH>	CAN4 error interrupt
<GIC_SPI 707 IRQ_TYPE_LEVEL_HIGH>	CAN4 common RX FIFO or TXQ interrupt
<GIC_SPI 715 IRQ_TYPE_LEVEL_HIGH>	CAN4 TX interrupt
<GIC_SPI 702 IRQ_TYPE_LEVEL_HIGH>	CAN5 error interrupt
<GIC_SPI 708 IRQ_TYPE_LEVEL_HIGH>	CAN5 common RX FIFO or TXQ interrupt
<GIC_SPI 716 IRQ_TYPE_LEVEL_HIGH>	CAN5 TX interrupt

clocks:

slot 1:

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 CANFD PCLK CLK index 0x9c for RZ/V2H.

slot 2:

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 CANFD RAM CLK index 0x9d for RZ/V2H.

slot 3:

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 CANFD CLKC CLK 0x9e for RZ/V2H.

reset:

slot 1:

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

The 2rd cell must be set to CANFD RSTP_N reset index 0xa1 for RZ/V2H.

slot 2:

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

The 2rd cell must be set to CANFD RSTC_N reset index 0xa2 for RZ/V2H.

power domain:

Must be set to always on.

Interrupt-names:

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

clock-names:

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

reset-names:

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

The CAN-FD of RZ/G3E device properties are shown below.

```

canfd: can@12440000 {
    compatible = "renesas,r9a09g047-canfd";
    reg = <0 0x12440000 0 0x40000>;
    interrupts = <GIC_SPI 709 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 710 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 697 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 703 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 711 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 698 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 704 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 712 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 699 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 705 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 713 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 700 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 706 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 714 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 701 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 707 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 715 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 702 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 708 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 716 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "g_err", "g_recc",
        "ch0_err", "ch0_rec", "ch0_trx",
        "ch1_err", "ch1_rec", "ch1_trx",
        "ch2_err", "ch2_rec", "ch2_trx",
        "ch3_err", "ch3_rec", "ch3_trx",
        "ch4_err", "ch4_rec", "ch4_trx",
        "ch5_err", "ch5_rec", "ch5_trx";
    clocks = <&cpg CPG_MOD 156>,
        <&cpg CPG_MOD 157>,
        <&cpg CPG_MOD 158>;
    clock-names = "fck", "clk_ram", "can_clk";
    resets = <&cpg 161>,
        <&cpg 162>;
    reset-names = "rstp_n", "rstc_n";
    power-domains = <&cpg>;
    status = "disabled";

    channel0 {
        status = "disabled";
    };
    channel1 {
        status = "disabled";
    };
    channel2 {
        status = "disabled";
    };
    channel3 {
        status = "disabled";
    };
    channel4 {
        status = "disabled";
    };
    channel5 {
        status = "disabled";
    };
};

```

Figure 4-7 CANFD Node example for RZ/G3E

Note: Example of CAN-FD device tree setting for RZ/G3E.

The CAN-FD device required properties:

compatible:

Must be set

"renesas, r9a09g047-canfd" for R9A09G047 (RZ/G3E).

reg:

Base address and length of the memory resource used by the CANFD driver.

interrupt:

<GIC_SPI 709 IRQ_TYPE_LEVEL_HIGH>	CAN global error interrupt
<GIC_SPI 710 IRQ_TYPE_LEVEL_HIGH>	CAN RX FIFO interrupt
<GIC_SPI 697 IRQ_TYPE_LEVEL_HIGH>	CAN0 error interrupt
<GIC_SPI 703 IRQ_TYPE_LEVEL_HIGH>	CAN0 common RX FIFO or TXQ interrupt
<GIC_SPI 711 IRQ_TYPE_LEVEL_HIGH>	CAN0 TX interrupt
<GIC_SPI 698 IRQ_TYPE_LEVEL_HIGH>	CAN1 error interrupt
<GIC_SPI 704 IRQ_TYPE_LEVEL_HIGH>	CAN1 common RX FIFO or TXQ interrupt
<GIC_SPI 712 IRQ_TYPE_LEVEL_HIGH>	CAN1 TX interrupt
<GIC_SPI 699 IRQ_TYPE_LEVEL_HIGH>	CAN2 error interrupt
<GIC_SPI 705 IRQ_TYPE_LEVEL_HIGH>	CAN2 common RX FIFO or TXQ interrupt
<GIC_SPI 713 IRQ_TYPE_LEVEL_HIGH>	CAN2 TX interrupt
<GIC_SPI 700 IRQ_TYPE_LEVEL_HIGH>	CAN3 error interrupt
<GIC_SPI 706 IRQ_TYPE_LEVEL_HIGH>	CAN3 common RX FIFO or TXQ interrupt
<GIC_SPI 714 IRQ_TYPE_LEVEL_HIGH>	CAN3 TX interrupt
<GIC_SPI 701 IRQ_TYPE_LEVEL_HIGH>	CAN4 error interrupt
<GIC_SPI 707 IRQ_TYPE_LEVEL_HIGH>	CAN4 common RX FIFO or TXQ interrupt
<GIC_SPI 715 IRQ_TYPE_LEVEL_HIGH>	CAN4 TX interrupt
<GIC_SPI 702 IRQ_TYPE_LEVEL_HIGH>	CAN5 error interrupt
<GIC_SPI 708 IRQ_TYPE_LEVEL_HIGH>	CAN5 common RX FIFO or TXQ interrupt
<GIC_SPI 716 IRQ_TYPE_LEVEL_HIGH>	CAN5 TX interrupt

clocks:

slot 1:

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 CANFD PCLK CLK index 156 for RZ/G3E.

slot 2:

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 CANFD RAM CLK index 157 for RZ/G3E.

slot 3:

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 CANFD CLKC CLK index 158 for RZ/G3E.

reset:

slot 1:

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

The 2nd cell must be set to CANFD RSTP_N reset index 161 for RZ/G3E.

slot 2:

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

The 2nd cell must be set to CANFD RSTC_N reset index 162 for RZ/G3E.

power domain:

Must be set to always on.

Interrupt-names:

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

clock-names:

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

reset-names:

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

The CAN-FD of RZ/G3S device properties are shown below.

```

canfd: can@100c0000 {
    compatible = "renesas,r9a08g045-canfd", "renesas,rzg3s-canfd";
    reg = <0 0x100c0000 0 0x20000>;
    interrupts = <GIC_SPI 373 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 374 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 375 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 376 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 377 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 378 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 379 IRQ_TYPE_LEVEL_HIGH>,
        <GIC_SPI 380 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "g_err", "g_recc",
        "ch0_rec", "ch1_rec", "ch0_err",
        "ch1_err", "ch0_trx", "ch1_trx";
    clocks = <&cpg CPG_MOD R9A08G045_CANFD_CLK_RAM>,
        <&cpg CPG_MOD R9A08G045_CANFD_PCLK>,
        <&cpg CPG_CORE R9A08G045_CLK_P4_DIV2>,
        <&can_clk>;
    clock-names = "ram_clk", "fck", "canfd", "can_clk";
    assigned-clocks = <&cpg CPG_CORE R9A08G045_CLK_P4_DIV2>;
    assigned-clock-rates = <80000000>;
    resets = <&cpg R9A08G045_CANFD_RSTP_N>,
        <&cpg R9A08G045_CANFD_RSTC_N>;
    reset-names = "rstp_n", "rstc_n";
    power-domains = <&cpg>;
    status = "disabled";

    channel0 {
        status = "disabled";
    };
    channel1 {
        status = "disabled";
    };
};

```

Figure 4-8 CAN FD Node Example for RZ/G3S

Note: Example of CAN-FD device tree setting for RZ/G3S.

The CAN-FD device required properties:

compatible:

Must be set

"renesas, r9a08g045-canfd" for R9A08G045 (RZ/G3S).

reg:

Base address and length of the memory resource used by the CANFD driver.

interrupt:

<GIC_SPI 373 IRQ_TYPE_LEVEL_HIGH>	CAN global error interrupt
<GIC_SPI 374 IRQ_TYPE_LEVEL_HIGH>	CAN RX FIFO interrupt
<GIC_SPI 375 IRQ_TYPE_LEVEL_HIGH>	CAN0 error interrupt
<GIC_SPI 376 IRQ_TYPE_LEVEL_HIGH>	CAN1 error interrupt
<GIC_SPI 377 IRQ_TYPE_LEVEL_HIGH>	CAN0 common RX FIFO or TXQ interrupt
<GIC_SPI 378 IRQ_TYPE_LEVEL_HIGH>	CAN1 common RX FIFO or TXQ interrupt
<GIC_SPI 379 IRQ_TYPE_LEVEL_HIGH>	CAN0 TX interrupt
<GIC_SPI 380 IRQ_TYPE_LEVEL_HIGH>	CAN1 TX interrupt

clocks:

slot 1:

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 R9A08G045_CANFD_CLK_RAM for RZ/G3S .

slot 2:

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 R9A08G045_CANFD_PCLK for RZ/G3S.

slot 3:

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 R9A08G045_CLK_P4_DIV2 for RZ/G3S.

reset:

slot 1:

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

The 2nd cell must be set to R9A08G045_CANFD_RSTP_N for RZ/G3S.

slot 2:

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

The 2nd cell must be set to R9A08G045_CANFD_RSTC_N for RZ/G3S.

power domain:

Must be set to always on.

Interrupt-names:

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

clock-names:

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

reset-names:

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

Please configure in arch/arm64/boot/dts/renesas/rzg3s-smarc.dtsi to enable pin standby input mode as below:

```
#if SW_GPIO_CAN_PMOD1 == SW_ON
    /* GPIO8_CAN0_STB */
    can0-stb-hog {
        gpio-hog;
        gpios = <RZG2L_GPIO(13, 0) GPIO_ACTIVE_HIGH>;
        output-low;
        line-name = "can0_stb";
    };
#endif

#if SW_GPIO_CAN_PMOD2 == SW_ON
    /* GPIO9_CAN0_STB */
    can1-stb-hog {
        gpio-hog;
        gpios = <RZG2L_GPIO(13, 1) GPIO_ACTIVE_HIGH>;
        output-low;
        line-name = "can1_stb";
    };
#endif
```

Figure 4-9 Standby input mode

5. Integration

5.1 Directory Configuration

The directory configuration is shown below.

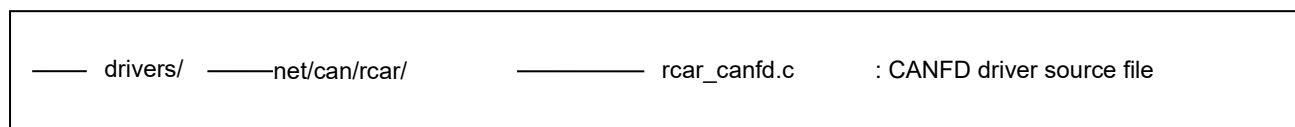


Figure 5-1 Directory Configuration

5.2 Integration Procedure

5.2.1 Kernel Configuration

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

Set kernel config for CANFD driver following as below of kernel config file (defconfig) in arch/arm64/configs/defconfig directory.

```
CONFIG_CAN=y
CONFIG_CAN_RCAR_CANFD=y
```

Figure 5-2 Kernel configuration

5.2.2 Enable Pin-function control for CANFD driver.

Enable pin function control for RZ/G2L and RZ/V2L.

```

can0_pins: can0 {
    pinmux = <RZG2L_PORT_PINMUX(10, 1, 2)>, /* TX */
             <RZG2L_PORT_PINMUX(11, 0, 2)>; /* RX */
};

can1_pins: can1 {
    pinmux = <RZG2L_PORT_PINMUX(12, 1, 2)>, /* TX */
             <RZG2L_PORT_PINMUX(13, 0, 2)>; /* RX */
};
  
```

Figure 5-3 Enable Pin function control for RZ/G2L and RZ/V2L

Enable pin function control for RZ/G2LC.

Set "SW_RSPI_CAN = 1" in device-tree

Setting on Smarc Module board RZ/G2LC SW1-3 = "ON" , SW1,4 = "OFF"

```
#if SW_RSPI_CAN
    can1_pins: can1 {
        pinmux = <RZG2L_PORT_PINMUX(44, 0, 3)>, /* TXD */
                <RZG2L_PORT_PINMUX(44, 1, 3)>; /* RxD */
    };
#endif
```

Figure 5-4 Enable Pin function control for RZ/G2LC

Enable pin function control for RZ/G2UL and RZ/Five.

Set "SW_ET0_EN_N = 1" in device-tree

Setting on Smarc Module board RZ/G2UL and RZ/Five SW1-3 = "OFF"

```
#if (SW_ET0_EN_N)
    canfd0_pins: can0 {
        pinmux = <RZG2L_PORT_PINMUX(1, 1, 3)>, /* TX */
                <RZG2L_PORT_PINMUX(1, 2, 3)>; /* RX */
    };

    canfd1_pins: can1 {
        pinmux = <RZG2L_PORT_PINMUX(2, 0, 3)>, /* TX */
                <RZG2L_PORT_PINMUX(2, 1, 3)>; /* RX */
    };
#endif
```

Figure 5-5 Enable Pin function control for RZ/G2UL and RZ/Five

Enable pin function control for RZ/V2N.

Enable pin function control for RZ/V2N in devicetree file (arch/arm64/boot/dts/renesas/r9a09g056n48-rzv2n-evk.dts)

```
&pinctrl {
    ....
    canfd_pins: can_pins {
#ifdef (CAN_SPDIF_SCI_SEL == SCI_SEL || CAN_SPDIF_SCI_SEL == CAN_SEL)
        can0_pins: can0 {
            pinmux = <RZV2N_PORT_PINMUX(8, 0, 5)>, /* TX */
                <RZV2N_PORT_PINMUX(8, 1, 5)>; /* RX */

        };

        can1_pins: can1 {
            pinmux = <RZV2N_PORT_PINMUX(8, 2, 5)>, /* TX */
                <RZV2N_PORT_PINMUX(8, 3, 5)>; /* RX */

        };
#ifdef
#ifdef (CAN_SPDIF_SCI_SEL == SPDIF_SEL || CAN_SPDIF_SCI_SEL == CAN_SEL)
        can2_pins: can2 {
            pinmux = <RZV2N_PORT_PINMUX(8, 4, 5)>, /* TX */
                <RZV2N_PORT_PINMUX(8, 5, 5)>; /* RX */

        };

        can3_pins: can3 {
            pinmux = <RZV2N_PORT_PINMUX(8, 6, 5)>, /* TX */
                <RZV2N_PORT_PINMUX(8, 7, 5)>; /* RX */

        };
#ifdef
```

Figure 5-6 Enable Pin function control for RZ/V2N

Enable pin function control for RZ/V2H.

Enable pin function control for RZ/V2H in devicetree file (arch/arm64/boot/dts/renesas/rzv2h-evk-common.dtsi)

```
#if (SEL_CAN01_HEADER && !SEL_SCI4_CAN01)
    can0_pins: can0 {
        pinmux = <RZV2H_PORT_PINMUX(8, 0, 5)>, /* TX */
                 <RZV2H_PORT_PINMUX(8, 1, 5)>; /* RX */
    };

    can1_pins: can1 {
        pinmux = <RZV2H_PORT_PINMUX(8, 2, 5)>, /* TX */
                 <RZV2H_PORT_PINMUX(8, 3, 5)>; /* RX */
    };
#endif
```

Figure 5-7 Enable Pin function control for RZ/V2H

Enable pin function control for RZ/G3E.

Enable pin function control for RZ/G3E in devicetree file (arch/arm64/boot/dts/renesas/rzg3e-smarc.dtsi)

```

&pinctrl {
#if (!SW_LCD_EN) && (SW_GPIO8_CAN0_STB))
    can0-stb-hog {
        gpio-hog;
        gpios = <RZV2H_GPIO(5, 4) GPIO_ACTIVE_HIGH>;
        output-low;
        line-name = "can0_stb";
    };
#endif
#if (!SW_LCD_EN) && (SW_GPIO9_CAN1_STB))
    can1-stb-hog {
        gpio-hog;
        gpios = <RZV2H_GPIO(5, 5) GPIO_ACTIVE_HIGH>;
        output-low;
        line-name = "can1_stb";
    };
#endif

    canfd_pins: canfd {
#if (ISW_PDM_EN)
        can1_pins: can1 {
            pinmux = <RZV2H_PORT_PINMUX(L, 2, 3)>, /* RX */
                    <RZV2H_PORT_PINMUX(L, 3, 3)>; /* TX */
        };
#endif
#if (ISW_LCD_EN)
        can4_pins: can4 {
            pinmux = <RZV2H_PORT_PINMUX(5, 2, 3)>, /* RX */
                    <RZV2H_PORT_PINMUX(5, 3, 3)>; /* TX */
        };
#endif
    };
};

```

Figure 5-8 Enable Pin function control for RZ/G3E

Enable pin function control for RZ/G3S.

Enable pin function control for RZ/G3S in devicetree file (arch/arm64/boot/dts/renesas/rzg3s-smarc.dtsi)

```
&pinctrl {  
    ....  
    can0_pins: can0 {  
        pinmux = <RZG2L_PORT_PINMUX(6, 1, 3)>, /* TX */  
                <RZG2L_PORT_PINMUX(6, 2, 3)>; /* RX */  
    };  
  
    can1_pins: can1 {  
        pinmux = <RZG2L_PORT_PINMUX(17, 0, 3)>, /* TX */  
                <RZG2L_PORT_PINMUX(17, 1, 3)>; /* RX */  
    };  
    ....  
}
```

Figure 5-9 Enable Pin function control for RZ/G3S

5.2.3 Enable CANFD driver for each channel.

Enable both channel 0 and channel 1 for RZ/G2L, RZ/V2L, RZ/G2LC, RZ/G2UL, RZ/Five.

```
&can {
    pinctrl-0 = <&canfd0_pins &canfd1_pins>;
    pinctrl-names = "default";
    status = "okay";

    channel0 {
        status = "okay";
    };

    channel1 {
        status = "okay";
    };
};
```

Figure 5-10 Enable channel CANFD

Enable channel 1 only for RZ/G2LC by using delete-node

```
#if (SW_SCIF_CAN || SW_RSPI_CAN)
&canfd {
    pinctrl-0 = <&can1_pins>;
    /delete-node/ channel0;
};
#else
&canfd {
    /delete-property/ pinctrl-0;
    /delete-property/ pinctrl-names;
    status = "disabled";
};
#endif
```

Figure 5-11 Enable channel CANFD RZ/G2LC

Disabled CANFD for G2UL using config SW_ET0_EN_N

```
#if (!SW_ET0_EN_N)
&canfd {
    /delete-property/ pinctrl-0;
    /delete-property/ pinctrl-names;
    status = "disabled";
};
#endif
```

Figure 5-12 Disabled CANFD on RZ/G2UL

Enable both channels 0, 1 for RZ/V2H in directory (arch/arm64/boot/dts/renesas/rzv2h-evk-common.dtsi)

```
#if (SEL_CAN01_HEADER && !SEL_SCI4_CAN01)
&canfd {
    pinctrl-0 = <&can0_pins &can1_pins>;
    pinctrl-names = "default";
    status = "okay";
    channel0 {
        status = "okay";
    };
    channel1 {
        status = "okay";
    };
};
#endif
```

Figure 5-13 Enable 2 channels CAN-FD (RZ/V2H)

Hardware design of CANFD for RZ/V2H interface have some pins function control conflict with others function. Define switch macro to choose CANFD function for pins (arch/arm64/boot/dts/renesas/r9a09g057h44-rzv2h-evk-ver2.dts)

```
#define SEL_CAN01_HEADER 1
#define SEL_SCI4_CAN01 0
```

Figure 5-14 Macro switch support 2 channels CAN-FD (RZ/V2H)

Enable 4 channels 0, 1, 2, 3 for RZ/V2N in directory (arch/arm64/boot/dts/renesas/r9a09g056n48-rzv2n-evk.dts)

```
&canfd {
    pinctrl-0 = <&canfd_pins>;
    pinctrl-names = "default";

    status = "okay";

    #if (CAN_SPDIF_SCI_SEL == SCI_SEL || CAN_SPDIF_SCI_SEL == CAN_SEL)
        channel0 {
            status = "okay";
        };

        channel1 {
            status = "okay";
        };
    #endif
    #if (CAN_SPDIF_SCI_SEL == SPDIF_SEL || CAN_SPDIF_SCI_SEL == CAN_SEL)
        channel2 {
            status = "okay";
        };

        channel3 {
            status = "okay";
        };
    #endif
};
```

Figure 5-15 Enable 4 channels CAN-FD (RZ/V2N)

Hardware design of CANFD for RZ/V2N interface have some pins function control conflict with others function. Define switch macro to choose CANFD function for pins (arch/arm64/boot/dts/renesas/r9a09g056n48-rzv2n-evk.dts)

```
/* Select among CAN/SPDIF/SCI pins
 * CAN_SPDIF_SCI_SEL value:
 * 0: CAN is selected (default)
 * 1: SPDIF is selected
 * 2: SCI is selected
 */
```

Figure 5-16 Define macro switch for CAN-FD (RZ/V2N)

If you want to enable all 4 channels, set the macro in the device-tree (arch/arm64/boot/dts/renesas/r9a09g056n48-rzv2n-evk.dts) as shown below and delete the unused nodes.

#define CAN_SPDIF_SCI_SEL	0
#define CAN_SEL	0
#define SPDIF_SEL	1
#define SCI_SEL	2

Figure 5-17 Macro switch support 4 channels CAN-FD (RZ/V2N)

Enable 2 channels 1, 4 for RZ/G3E in directory (arch/arm64/boot/dts/renesas/rzg3e-smarc.dtsi)

```

&canfd {
    #if (!ISW_PDM_EN) || (!ISW_LCD_EN)

        pinctrl-0 = <&canfd_pins>;
        pinctrl-names = "default";
        status = "okay";

    #endif

    #if (!ISW_PDM_EN)

        channel1 {

            status = "okay";

        };

    #endif

    #if (!ISW_LCD_EN)

        channel4 {

            status = "okay";

        };

    #endif

};

```

Figure 5-18 Enable 2 channels CAN-FD (RZ/G3E)

Hardware design of CANFD for RZ/G3E interface have some pins function control conflict with others function. Define switch macro to choose CANFD function for pins (arch/arm64/boot/dts/renesas/rzg3e-smarc.dtsi)

```

* SW_GPIO_CAN_PMOD states:

* Please change the corresponding selection to below Macros:

* SW_GPIO8_CAN0_STB:

*     0 - Connect to GPIO8 PMOD (default)
*     1 - Connect to CAN0 transceiver STB pin

* SW_GPIO9_CAN1_STB:

*     0 - Connect to GPIO9 PMOD (default)
*     1 - Connect to CAN1 transceiver STB pin

```

Figure 5-19 Define macro switch for CAN-FD (RZ/G3E)

```

#define SW_LCD_EN            0
#define SW_PDM_EN           0

```

Figure 5-20 Macro switch support 2 channels CAN-FD (RZ/G3E)

Enable 2 channels 0, 1 for RZ/G3S in directory (arch/arm64/boot/dts/renesas/rzg3s-smarc.dtsi)

```
&canfd {
    pinctrl-0 = <&can0_pins &can1_pins>;
    pinctrl-names = "default";
    status = "okay";

    channel0 {
        status = "okay";
    };

    channel1 {
        status = "okay";
    };
};
```

Figure 5-21 Enable 2 channels CAN-FD (RZ/G3S)

Hardware design of CAN-FD for RZ/G3S that supports to switch between CAN and PMOD pins. Define Switch in (arch/arm64/boot/dts/renesas/r9a08g045s33-smarc.dts)

```
/*
 * SW_GPIO_CAN_PMOD[x]: switch between CAN and PMOD pins
 * @SW_GPIO_CAN_PMOD1:
 *     SW_OFF - GPIO8 is connected to GPIO8_PMOD
 *     SW_ON  - GPIO8 is connected to GPIO8_CAN0_STB
 * @SW_GPIO_CAN_PMOD2:
 *     SW_OFF - GPIO9 is connected to GPIO9_PMOD
 *     SW_ON  - GPIO9 is connected to GPIO9_CAN1_STB
 */
#define SW_GPIO_CAN_PMOD1      SW_OFF
#define SW_GPIO_CAN_PMOD2      SW_OFF
```

Figure 5-22 Macro Switch for switching between CAN and PMOD pins

5.3 Option Setting

5.3.1 Module Parameters

There are no module parameters.

5.3.2 Kernel Parameters

There are no kernel parameters.

5.3.3 Device tree bindings

Demonstrates an example of device tree setting which can be found in Documentation/devicetree/bindings/net/can/rcar-canfd.txt

Property to set classical CAN in device tree:

renesas,no-can-fd;

Property to set CAN-FD dual mode in device tree:

renesas,can-fd-dualmode;

5.3.4 Userspace API

Configure CAN as CANFD or Classical CAN:

By default, the CAN driver operates in CANFD mode.

Classical CAN mode: Add the property **renesas,no-can-fd;** to canfd node in arch/arm64/boot/dts/renesas/rz-smarc-common.dtsi for G2L series and RZ/V2L.

```
&canfd {
    pinctrl-0 = <&can0_pins &can1_pins>;
    pinctrl-names = "default";
    renesas,no-can-fd;
    status = "okay";

    channel0 {
        status = "okay";
    };

    channel1 {
        status = "okay";
    };
};
```

Figure 5-23 Enable Classical CAN for RZ/G2L series and RZ/V2L

Configure CAN-FD driver to run in FD-only mode (RZ/G3E and RZ/V2N). This is default mode so there is no need to add property.

Configure CAN-FD driver to run in CAN-FD dual mode (RZ/G3E, RZ/V2H and RZ/V2N). Add property **renesas,can-fd-dualmode**; to canfd node in (arch/arm64/boot/dts/renesas/r9a09g056n48-rzv2n-evk.dts) for RZ/V2N.

```
&canfd {
    pinctrl-0 = <&can0_pins &can1_pins>;
    pinctrl-names = "default";
    renesas,can-fd-dualmode;
    status = "okay";

    channel0 {
        status = "okay";
    };

    channel1 {
        status = "okay";
    };
};
```

Figure 5-24 Enable CAN-FD Dual mode example for RZ/V2N

RZ/G3S supported CAN-FD dual mode (cannot configure to Classical only mode or FD-only mode).

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

Rev.	Date	Description	
		Page	Summary
0.50	Apr. 31, 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	7,8,9	Update device-tree information
1.4	May. 31, 2022	-	No modification, change version to keep consistent with other documents
1.5	Jun. 24, 2022	-	Add support RZ/Five and update device-tree information
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	May. 30, 2025	1	- Add MPU information support for both kernel versions v5.10 and v6.1.
		3	- Correct remove redundant blank line at the top page.
		6	- Correct format of this page.
		10	- Correct remove redundant blank line at the top page.
		11	- Correct figure 5.3 enable PFC for G2L, V2L.
1.10	Jun. 30, 2025	-	Add RZ/V2N support information
1.11	Jul. 22, 2025	-	Add RZ/G3E support information
1.12	Nov. 28, 2025	1	Add information of RZ/G2UL and RZ/V2L support for kernel v6.1
		21	Add figure 5.9 for G2UL disable config
		24	Add guide how to config CAN as CANFD or classical CAN
		-	Add RZ/G3S support information. Include the names of all figures to improve clarity and understanding. Add information and guide to configure three modes: Dual, FD-only and Classical only.
1.13	Dec. 19, 2025	-	Add RZ/V2H support information
1.14	Mar 27, 2026	-	Change RZ/V2N device tree information

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

Publication Date: Rev. 1.14 Mar 27, 2026

Published by: Renesas Electronics Corporation

RZ/G2L Group, RZ/V2L Group, RZ/V2H Group,
RZ/V2N Group, RZ/G3E Group, RZ/G3S Group
and RZ/Five Group



Renesas Electronics Corporation