RZ/A2M Group

RS-CANFD Sample Program Application Note

Introduction

This application note describes a sample program that performs CAN FD communication using the RS-CANFD Driver for RZ/A2M Group MCUs.

Target Device

RZ/A2M

The RS-CANFD sample program described in this application note does not work with original RZ/A2M SUB board (RTK79210XXB00000BE) of Renesas. It is necessary to mount the CAN transceiver and the oscillator for the CAN clock.

Contents

1. Overview 3

2. Operation Confirmation Conditions 4
2.1 System configuration 5
2.1.1 Hardware 5
2.1.2 Software 7

3. Operation explanation 8
3.1 Common setting 8
3.1.1 CAN Modes 8
3.1.2 Communication speed 10
3.2 Operation example 1: transmit / receive operation 12
3.2.1 Overview of operation 12
3.2.2 CAN frame header information setting 17
3.2.3 Transmit/receive FIFO buffer setting 18
3.2.4 Receive rule setting 19
3.2.5 Receive FIFO buffer setting 21
3.3 Operation example 2: gateway operation 22
3.3.1 Overview of operation 22
3.3.2 Gateway settings 26
3.3.3 Receive rule setting 28
3.4 Operation example 3: external Loopback operation 29
3.4.1 Overview of operation 29
3.4.2 CAN frame header information setting 32
3.4.3 Transmit buffer setting 32
3.4.4 Receive rule setting 33
3.4.5 Receive buffer setting 34

4. Restrictions 35

5. Precautions 35

6. Used open source software and licenses 35

7. Reference Documents 36

Revision History 37

1. Overview

RS-CANFD sample program application note (sample program) is an example of CAN FD communication operation using the RZ/A2M CANFD interface (RS-CANFD) driver. The following three items are listed as operation examples.

Table 1-1 List of operation examples

#	Contents	Resources used by RS-CANFD
1	Transmit / receive operation	Transmit : Transmit/receive FIFO Receive : Receive FIFO
2	Gateway operation	Transmit/receive FIFO
3	External Loopback operation	Transmit buffer Receive buffer

Refer to RZ/A2M group RS-CANFD driver application note (R01AN4868) for details of the APIs. This sample program uses only the following APIs.

Table 1-2 List of used APIs

API name	Contents	Operation example 1	Operation example 2	Operation example 3
R_CAN_Create	Initial setting	○	○	○
R_CAN_Control	Operation mode transition	○	○	○
R_CAN_PortSet	Port function / port test settings	○	○	○
R_CAN_SetBitrate(Note)	Bit rate setting	○	○	○
R_CAN_TxSet	Transmit data setting	○		○
R_CAN_Tx	Transmit request setting	○		○
R_CAN_TxCheck	Transmit completion check			○
R_CAN_RxSet	Receive permission setting	○		○
R_CAN_RxPoll	Receive completion check	○		○
R_CAN_RxRead	Receive data read	○		○
R_CAN_Gateway	Enable Gateway operation.		○	

Note: R_CAN_SetBitrate is used inside R_CAN_Create.

Table 1-3 Peripheral function and Usage

Peripheral function	Usage
CANFD interface (RS-CANFD)	CAN FD communication module
Interrupt controller (INTC)	Configuration of RS-CANFD interrupt
General purpose I/O port (GPIO)	RS-CANFD terminal function setting

2. Operation Confirmation Conditions

The sample code of this application has been verified by following conditions.

Table 2-1 Peripheral device used

Item	Contents
MCU used	RZ/A2M
Operating frequency (Note1)	CPU Clock (I ϕ): 528MHz Image processing clock (G ϕ): 264MHz Internal Bus Clock (B ϕ): 132MHz Peripheral Clock 1 (P1 ϕ): 66MHz Peripheral Clock 0 (P0 ϕ): 33MHz QSPI0_SPCLK: 66MHz CKIO: 132MHz CAN_CLK: 32MHz (Note2)
Operating voltage	Power supply voltage (I/O): 3.3V Power supply voltage (either 1.8V or 3.3V I/O (PVcc_SPI)): 3.3V Power supply voltage (internal): 1.2V
Integrated development environment	e2 studio V7.4.0
C compiler	GNU Arm Embedded Toolchain Version8-2018-q4-major
	Compiler options (except directory path) Hardware Debug: -mcpu=cortex-a9 -march=armv7-a –marm -mlittle-endian -mfloat-abi=hard -mfpu=neon -Os -ffunction-sections -fdata-sections -Wnull-dereference -g3 -Wstack-usage=100 -std=gnu99 -fdiagnostics-parseable-fixits
Emulator	J-Link LITE or J-Link BASE (Manufacturer: SEGGER Microcontroller)
Operation mode	Boot mode 3 (Serial Flash boot 3.3V)
Board to be used (Note3)	RZ/A2M CPU board RTK7921053C00000BE RZ/A2M SUB board RTK79210XXB00000BE
Device (functionality to be used on the board)	• Serial flash memory allocated to SPI multi-I/O bus space Manufacturer: Macronix Inc., Model Name: MX25L51245GXD • CAN transceiver Manufacturer: NXP Semiconductors, Model Name: TJF1051T/3

Note1: The operating frequency used in clock mode 1 (Clock input of 24MHz from EXTAL pin)

Note2: External clock (CAN_CLK) is used as CAN communication clock source

Note3: Since the CAN transceiver and the oscillator for the CAN clock are not mounted, they must be mounted when using CAN.

2.1 System configuration

2.1.1 Hardware

Figure 2-1 shows the hardware configuration of this application note. Transmit / receive operation uses only channel 0 of each board. Gateway operation uses both channels 0 and 1 on each board. The external loopback test uses only channel 0 of one board. For details, refer to the operation overview of each operation example.

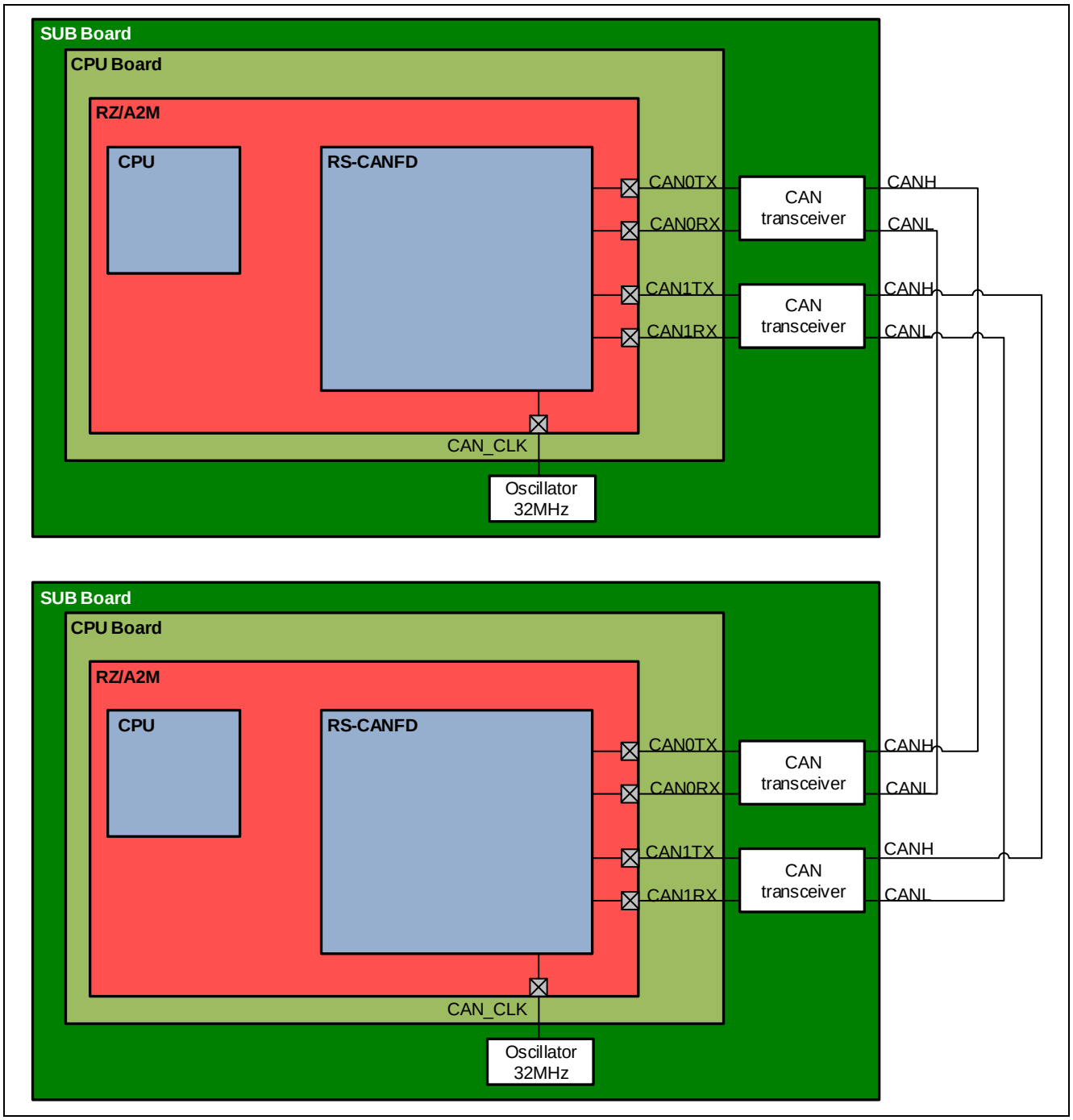


Figure 2-1 Hardware configuration

The RZ/A2M SUB board does not mounted a CAN transceiver and an oscillator for the CAN clock, so it must be mounted when using CAN. Figure 2-2 shows part of the circuit diagram of the RZ/A2M SUB board. The dotted line is the unmounted part.

For details, refer to the RTK79210XXB00000BE (RZ/A2M SUB board) User's Manual.

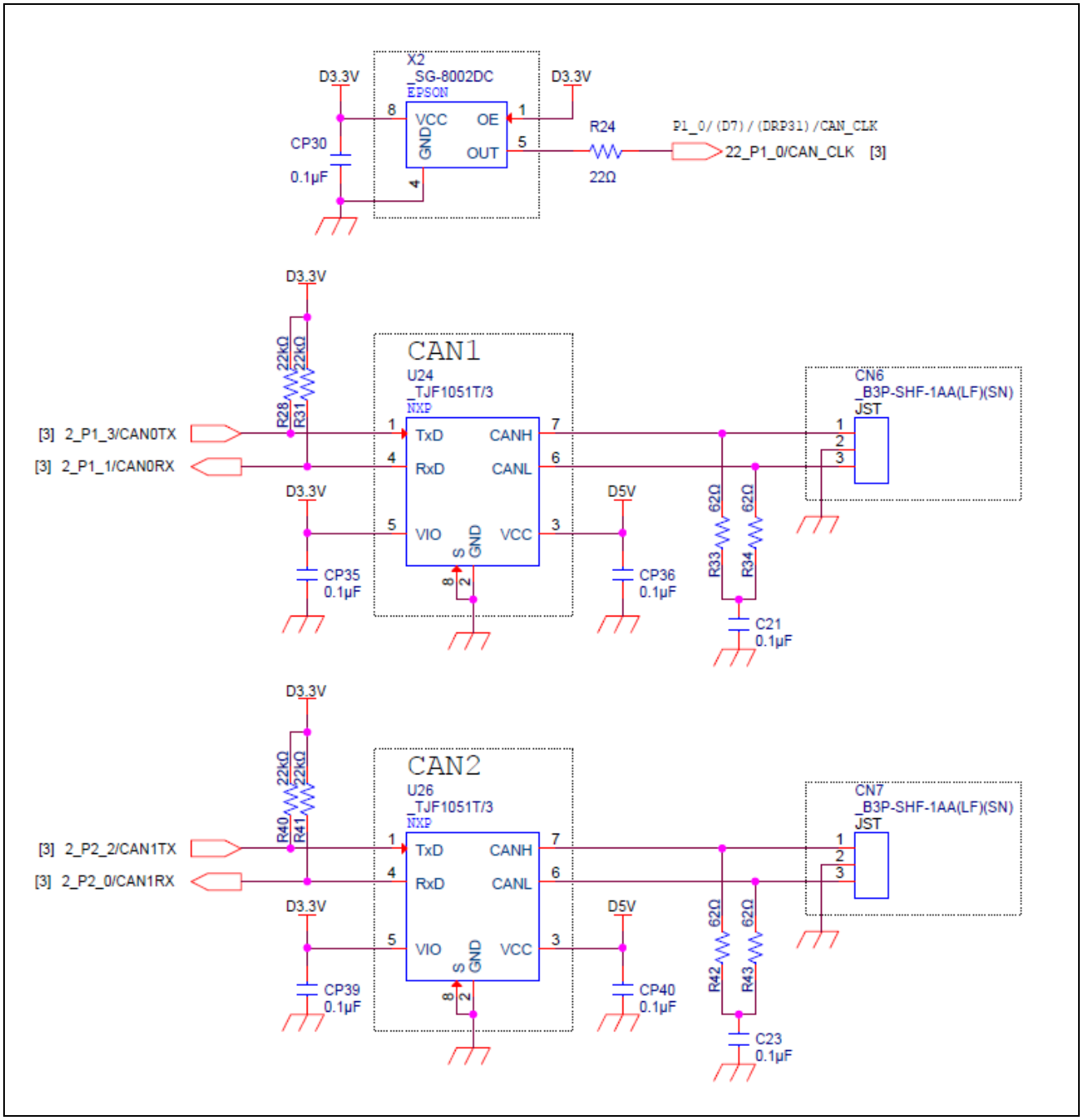


Figure 2-2 Part of the circuit diagram of the RZ/A2M SUB board

2.1.2 Software

Figure 2-3 shows sample program system block diagram. There are 2 tasks in this program. One controls CAN transmit and receive, and another one is executed by an interrupt request.

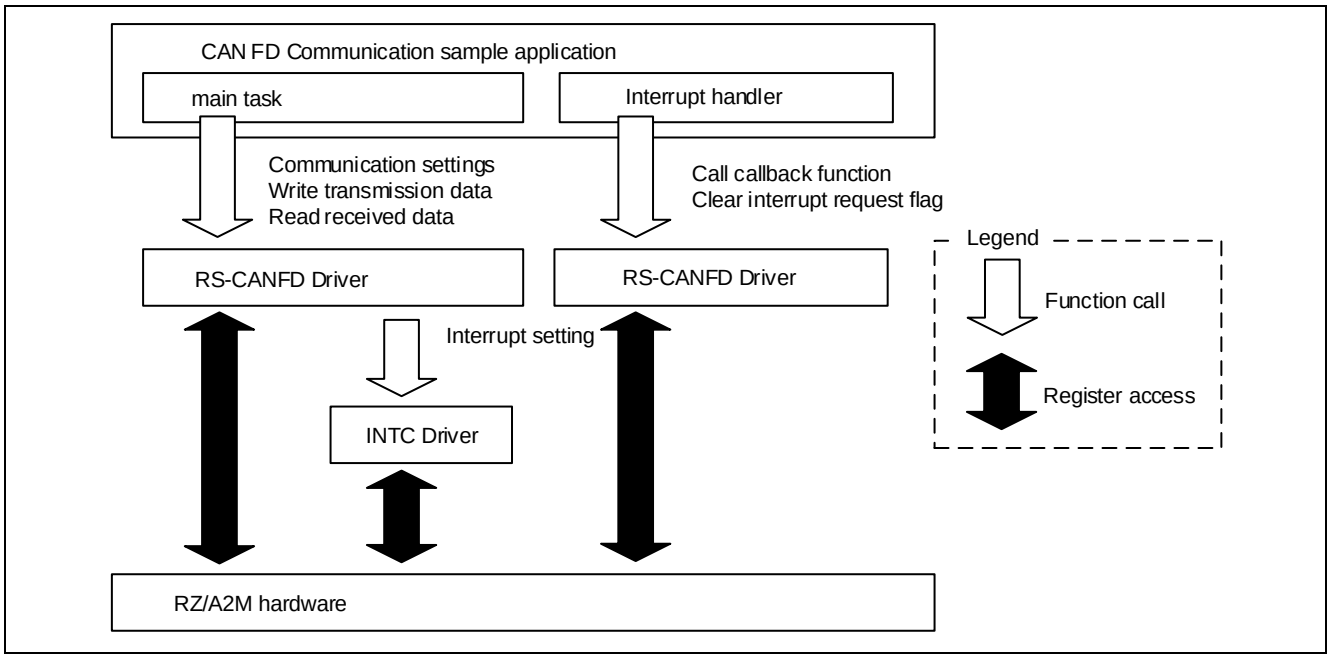


Figure 2-3 Sample program system block diagram

Table 2-2 shows the folder structure of this sample program.

Table 2-2 Folder structure

Folder name	Project name	Executable file name	Debug configuration name	Description
Transmit_64byte	Transmit_64byte	Transmit_64byte.elf	Transmit_64byte Hardware Debug	Operation example 1 Transmit side sample project
Receive_64byte	Receive_64byte	Receive_64byte.elf	Receive_64byte Hardware Debug	Operation example 1 Receive side sample project
Gateway	Gateway	Gateway.elf	Gateway Hardware Debug	Operation example 2 Gateway side sample project
Gateway_Opposite	Gateway_Opposite	Gateway_Opposite.elf	Gateway_Opposite Hardware Debug	Operation example 2 Opposite board sample project
Loopback_Test	Loopback_Test	Loopback_Test.elf	Loopback_Test Hardware Debug	Operation example 3 Sample project

3. Operation explanation

3.1 Common setting

This chapter describes the settings that are commonly made in operation examples 1 to 3 using the RS-CANFD driver.

3.1.1 CAN Modes

The RS-CANFD module has four global modes to control entire RS-CANFD module status and four channel modes to control individual channel status.

- Global stop mode: Stops clocks of the entire module to achieve low power consumption.
- Global reset mode: Performs initial settings for the entire module.
- Global test mode: Performs test settings and performs RAM test.
- Global operating mode: Makes the entire module operable.

- Channel stop mode: Stops the channel clock.
- Channel reset mode: Performs initial settings for the channel.
- Channel halt mode: Stops CAN communication and performs channel test.
- Channel communication mode: Performs CAN communication.

Figure 3-1 shows the state transition diagram of the global mode in this sample program. After MCU reset, global mode is set to global stop mode. When the initial setting (R_CAN_Create function) is executed, it transits to global test mode via global reset mode as an intermediate state. Then, to actually execute transmit / receive, transition to global operating mode. R_CAN_Control function is normally used for transition to the global operating mode (in the case of operation examples 1 and 2), but even in the CAN port test setting (when the port test is specified as an argument of R_CAN_PortSet function), it transits to the global operating mode (in the case of operation example 3).

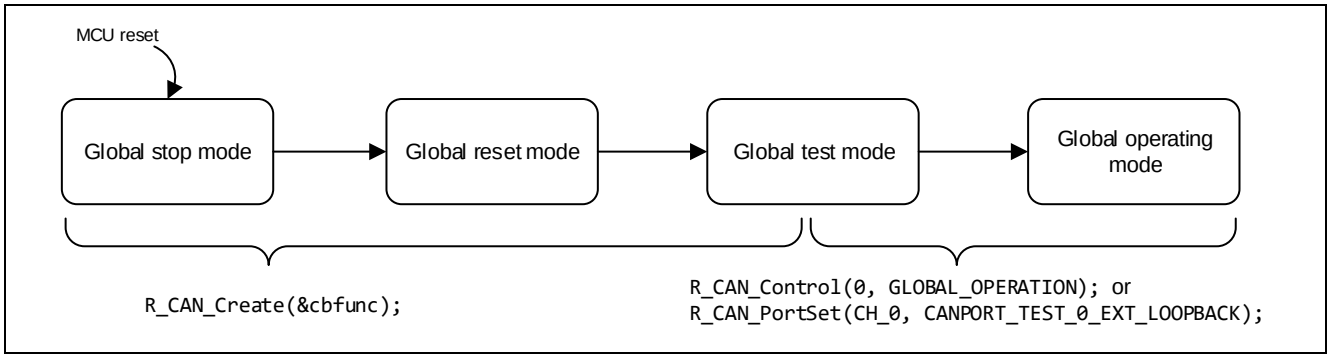


Figure 3-1 State transition diagram of the global mode

Figure 3-2 shows the state transition diagram of the channel mode in this sample program. After MCU reset, the channel mode is set to channel stop mode. When the initial setting (R_CAN_Create function) is executed, it transits to channel halt mode via channel reset mode as an intermediate state. Then, to actually execute transmit / receive, transition to the channel communication mode. R_CAN_Control function is normally used for transition to the channel communication mode (in the case of operation examples 1 and 2), but even in the CAN port test setting (when the port test is specified as an argument of R_CAN_PortSet function), it transits to the channel communication mode (in the case of operation example 3).

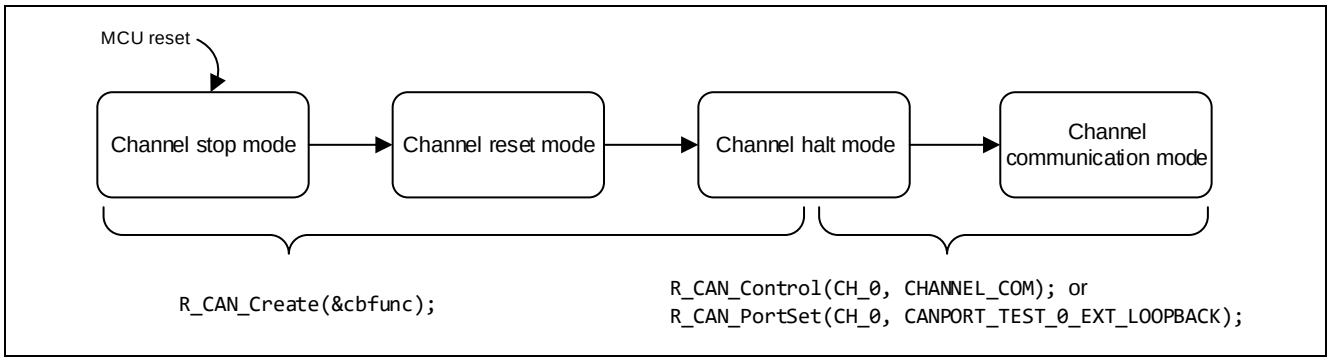


Figure 3-2 State transition diagram of the channel mode

3.1.2 Communication speed

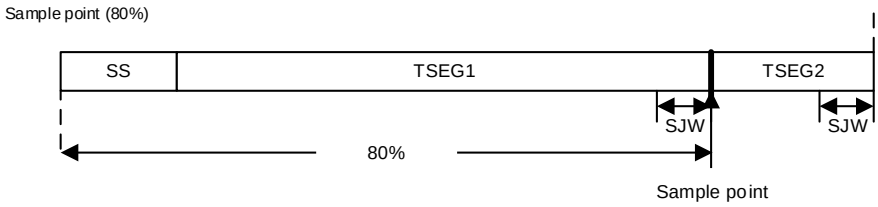
3.1.2.1 Bit Timing Setting

In the CAN bit timing setting of this RS-CANFD module, one bit of the communication frame is composed of three segments. Figure 3-3 shows the bit segments and sample point.

Among these segments, Time Segment 1 (TSEG1) and Time Segment 2 (TSEG2) specify sample point. The sampling timing can be changed by changing these values. The CAN FD mode has two types of bit rates (nominal bit rate and data bit rate), and each needs to be set.

The minimum unit for this timing setting is called 1 Time Quanta (Tq), and is determined by the clock frequency input to the RS-CANFD module and baud rate prescaler division value.

Table 3-1 shows bit timing setting example. Bit timing is set by the macro defined in "r_can_rz_config.h". In this sample program, the nominal bit rate bit timing is set to 1bit = 32Tq, sample point 59.38%, the data bit rate bit timing is set to 1bit = 16Tq, sample point 62.5%.



Classical CAN mode:
SS = 1 Tq fixed
Set TSEG1 to a range of 4 Tq to 16 Tq
Set TSEG2 to a range of 2 Tq to 8 Tq
Set SJW to a range of 1 Tq to 4 Tq
Set SS + TSEG1 + TSEG2 to a range of 8 Tq to 25 Tq
TSEG1 > TSEG2 > SJW

CAN FD mode:
Nominal bit rate
SS = 1 Tq fixed
Set TSEG1 to a range of 4 Tq to 128 Tq
Set TSEG2 to a range of 2 Tq to 32 Tq
Set SJW to a range of 1 Tq to 32 Tq
Set SS + TSEG1 + TSEG2 to a range of 8 Tq to 161 Tq
TSEG1 > TSEG2 > SJW

DATA bit rate
SS = 1 Tq fixed
Set TSEG1 to a range of 2 Tq to 16 Tq
Set TSEG2 to a range of 2 Tq to 8 Tq
Set SJW to a range of 1 Tq to 8 Tq
Set SS + TSEG1 + TSEG2 to a range of 8 Tq to 25 Tq
TSEG1 > TSEG2 > SJW

- SS (synchronization segment):
  The SS is a segment that performs synchronization by monitoring the edge from recessive to dominant bits in the Interframe Space.
  Interframe Space consists of Intermission, Suspend Transmission, and Bus Idle. All nodes can start transmission during Bus Idle.
- TSEG1 (Time segment 1):
  TSEG1 is a segment that absorbs physical delay on the CAN network. The physical delay on the CAN network is twice of the total of the delay on the CAN bus, the delay in the input comparator, and the delay in the output driver.
- TSEG2 (Time segment 2):
  TSEG2 is a segment that compensates phase error due to an error in frequency.
- SJW (Resynchronization jump width)
  The SJW is a length to extend or reduce the time segment to compensate for an error in phase due to phase error.

Figure 3-3 Bit segments and sample point

Table 3-1 Bit timing setting example

1Bit	Set Value (Note)			Sampling Point (%)
	CmNCFG_NTSEG1 CmDCFG_DTSEG1	CmNCFG_NTSEG2 CmDCFG_DTSEG2	CmNCFG_NSJW CmNCFG_DSJW	
8Tq	3	2	1	62.50
	4	1	1	75.00
16Tq	8	5	2	62.50
	9	4	2	68.75
	10	3	2	75.00
32Tq	20	9	2	68.75
	22	7	2	75.00
	17	12	2	59.38

Note: m: Channel number (0, 1)
Nominal bit rate corresponds to the RSCFDnCFDCmNCFG register, and data bit rate corresponds to the RSCFDnCFDCmDCFG register.

3.1.2.2 Communication speed setting

The communication speed is determined by the CAN clock (fCAN) which is the clock source of the RS-CANFD module, the baud rate prescaler division value, and Tq count per bit time. fCAN can use either clkc (internal clock P1 ϕ/2) or clk_xincan (external clock CAN_CLK).

Clock selection is performed with the macro GCFG_DCS (0: clkc, 1: clk_xincan) defined in "r_can_rz_config.h". The nominal bit rate prescaler division value is set with the macro CmNCFG_NBRP (m = 0, 1) and the data bit rate prescaler division value is set with the macro CmDCFG_DBRP (m = 0, 1). The value obtained by adding 1 to the setting value of each macro is the division value. Be sure to set the same value for CmNCFG_NBRP and CmDCFG_DBRP.

Table 3-2 and Table 3-3 show the main communication speed setting examples. This sample program uses clk_xincan 32MHz for fCAN and operates at a nominal bit rate of 1 Mbps and a data bit rate of 2 Mbps.

Table 3-2 Communication speed setting example (nominal bit rate)

Communication speed calculation formula	fCAN Nominal bit rate prescaler division ratio (Note1) × (Tq count of 1 nominal bit time)		
fCAN / Bit rate	32MHz	16MHz	8MHz
1Mbps	16Tq(2) 32Tq(1) (Note2)	8Tq(2) 16Tq(1)	8Tq(1)
500Kbps	8Tq(8) 16Tq(4)	8Tq(4) 16Tq(2)	8Tq(2) 16Tq(1)
250Kbps	8Tq(16) 16Tq(8)	8Tq(8) 16Tq(4)	8Tq(4) 16Tq(2)
125Kbps	8Tq(32) 16Tq(16)	8Tq(16) 16Tq(8)	8Tq(8) 16Tq(4)

Note1: Values in () are baud rate prescaler division values.
Note2: Settings in this sample program.

Table 3-3 Communication speed setting example (data bit rate)

Communication speed calculation formula	fCAN Data bit rate prescaler division ratio (Note1) × (Tq count of 1 data bit time)	
fCAN / Bit rate	32MHz	16MHz
4Mbps	8Tq(1)	None
2Mbps	16Tq(1) (Note2)	8Tq(1)

Note1: Values in () are baud rate prescaler division values.
Note2: Settings in this sample program.

3.2 Operation example 1: transmit / receive operation

3.2.1 Overview of operation

In this operation example, transmit and receive are performed using two RZ/A2M boards. Transmit side uses the transmit/receive FIFO buffer to transmit 64 bytes of data twice in succession, and receive side receives it twice using the receive FIFO buffer.

The transmit data must be set one by one. Therefore, message transmit execute before setting the second transmit data.

Reading of received data is performed in the interrupt processing of the receive FIFO interrupt. The receive FIFO buffer depth is set to 8 messages, and an interrupt request is generated when messages are stored in the FIFO buffer up to 2/8 full.

Figure 3-4 shows the system configuration of this operation example.

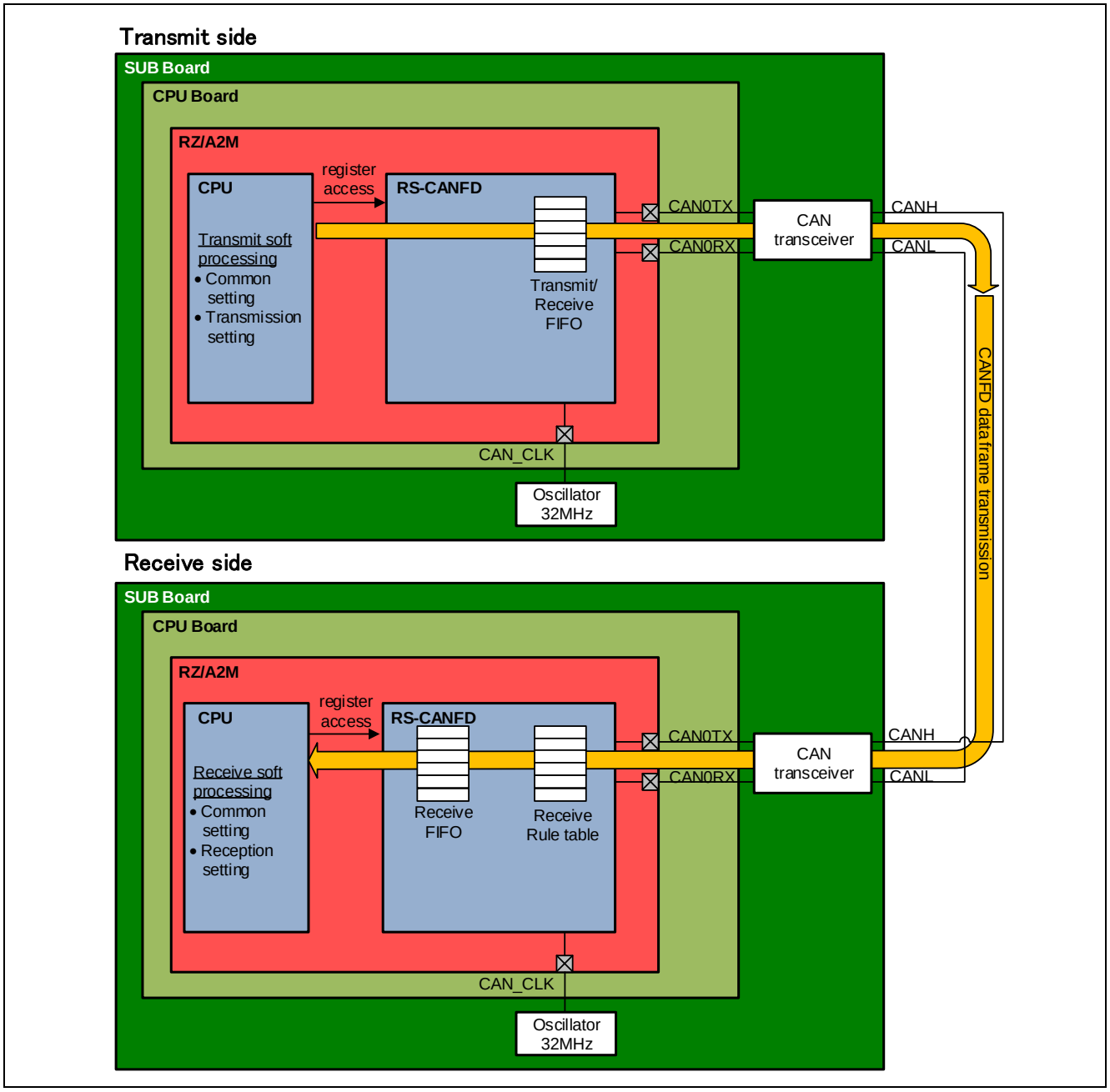


Figure 3-4 System configuration

The following shows the operation check procedure for transmit/receive operation. Confirm the transmit/receive operation by the program on the receive side.

- Method of operation
  1. Connect the channel 0 CANH pins (pin 1 of CAN1 connector) of each board. Also connect the channel 0 CANL pins (pin 3 of CAN1 connector) of each board.
  2. Set a breakpoint in the last while statement of the main function in the program on the receive side. Also, execute the program on the receive side to put it in a receive waiting state.
  3. Execute to transmit.

- What to check
  1. Make sure that the receiving program reaches the breakpoint and exits.
  2. In the program on the receive side, check the contents of the received data storage array (rx_buf_table) and confirm that it matches the transmitted data.

- Check result

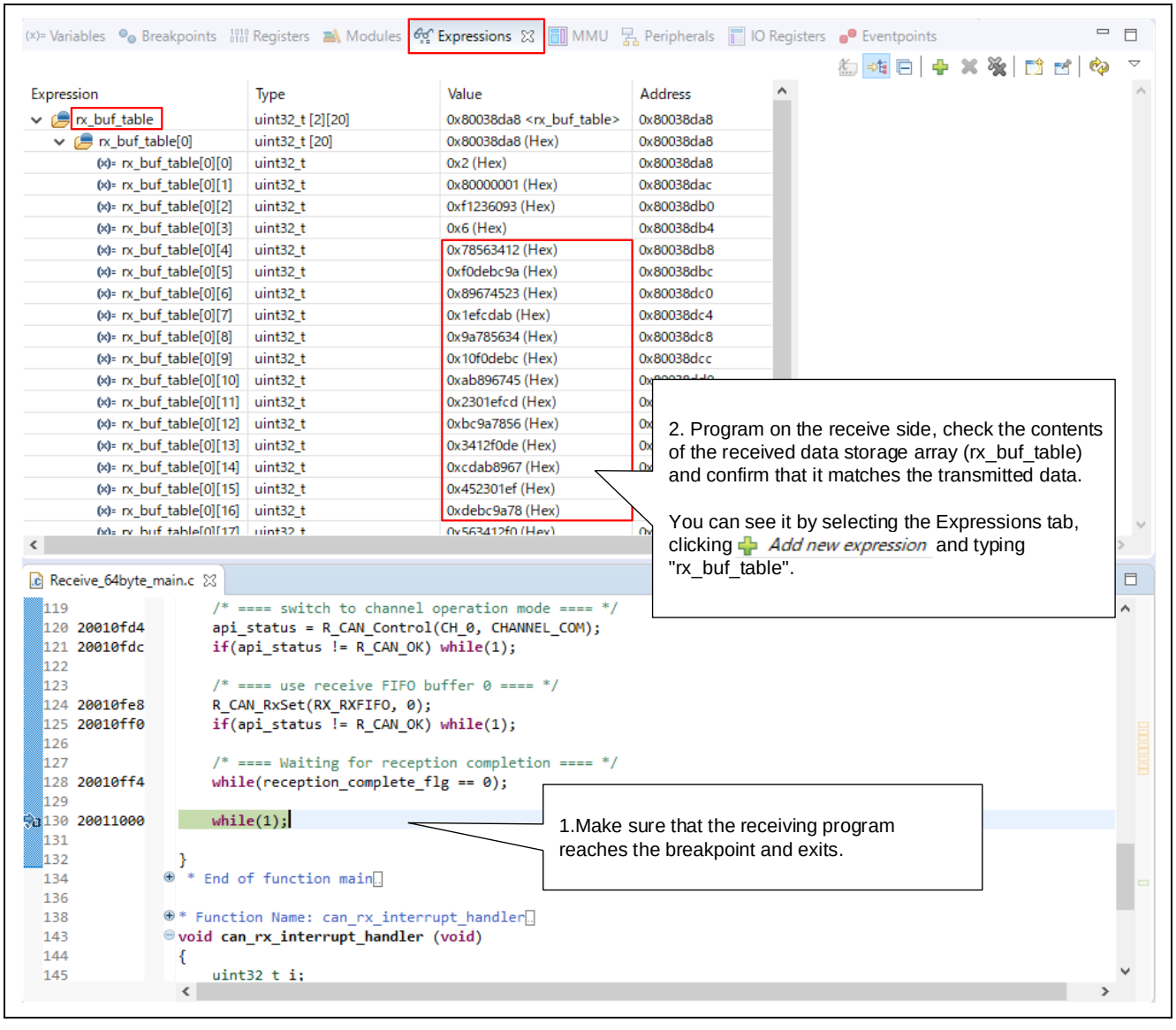


Figure 3-5 Check result

Expression	Type	Value	Address
rx_buf_table	uint32_t [2][20]	0x80038da8 <rx_buf_table>	0x80038da8
rx_buf_table[0]	uint32_t [20]	0x80038da8 (Hex)	0x80038da8
(x)= rx_buf_table[0][0]	uint32_t	0x2 (Hex)	0x80038da8
(x)= rx_buf_table[0][1]	uint32_t	0x80000001 (Hex)	0x80038dac
(x)= rx_buf_table[0][2]	uint32_t	0xf1236093 (Hex)	0x80038db0
(x)= rx_buf_table[0][3]	uint32_t	0x6 (Hex)	0x80038db4
(x)= rx_buf_table[0][4]	uint32_t	0x78563412 (Hex)	0x80038db8
(x)= rx_buf_table[0][5]	uint32_t	0xf0debc9a (Hex)	0x80038dbc
(x)= rx_buf_table[0][6]	uint32_t	0x89674523 (Hex)	0x80038dc0
(x)= rx_buf_table[0][7]	uint32_t	0x1efcdab (Hex)	0x80038dc4
(x)= rx_buf_table[0][8]	uint32_t	0x9a785634 (Hex)	0x80038dc8
(x)= rx_buf_table[0][9]	uint32_t	0x10f0debc (Hex)	0x80038dcc
(x)= rx_buf_table[0][10]	uint32_t	0xab896745 (Hex)	0x80038dd0
(x)= rx_buf_table[0][11]	uint32_t	0x2301efcd (Hex)	0x80038dd4
(x)= rx_buf_table[0][12]	uint32_t	0xbc9a7856 (Hex)	0x80038dd8
(x)= rx_buf_table[0][13]	uint32_t	0x3412f0de (Hex)	0x80038ddc
(x)= rx_buf_table[0][14]	uint32_t	0xcdab8967 (Hex)	0x80038de0
(x)= rx_buf_table[0][15]	uint32_t	0x452301ef (Hex)	0x80038de4
(x)= rx_buf_table[0][16]	uint32_t	0xdebc9a78 (Hex)	0x80038de8
(x)= rx_buf_table[0][17]	uint32_t	0x563412f0 (Hex)	0x80038dec
(x)= rx_buf_table[0][18]	uint32_t	0xefcdab89 (Hex)	0x80038df0
(x)= rx_buf_table[0][19]	uint32_t	0x67452301 (Hex)	0x80038df4
rx_buf_table[1]	uint32_t [20]	0x80038df8 (Hex)	0x80038df8
(x)= rx_buf_table[1][0]	uint32_t	0x80000002 (Hex)	0x80038df8
(x)= rx_buf_table[1][1]	uint32_t	0xf1248e69 (Hex)	0x80038dfc
(x)= rx_buf_table[1][2]	uint32_t	0x6 (Hex)	0x80038e00
(x)= rx_buf_table[1][3]	uint32_t	0xefcdab89 (Hex)	0x80038e04
(x)= rx_buf_table[1][4]	uint32_t	0x67452301 (Hex)	0x80038e08
(x)= rx_buf_table[1][5]	uint32_t	0xdebc9a78 (Hex)	0x80038e0c
(x)= rx_buf_table[1][6]	uint32_t	0x563412f0 (Hex)	0x80038e10
(x)= rx_buf_table[1][7]	uint32_t	0xcdab8967 (Hex)	0x80038e14
(x)= rx_buf_table[1][8]	uint32_t	0x452301ef (Hex)	0x80038e18
(x)= rx_buf_table[1][9]	uint32_t	0xbc9a7856 (Hex)	0x80038e1c
(x)= rx_buf_table[1][10]	uint32_t	0x3412f0de (Hex)	0x80038e20
(x)= rx_buf_table[1][11]	uint32_t	0xab896745 (Hex)	0x80038e24
(x)= rx_buf_table[1][12]	uint32_t	0x2301efcd (Hex)	0x80038e28
(x)= rx_buf_table[1][13]	uint32_t	0x9a785634 (Hex)	0x80038e2c
(x)= rx_buf_table[1][14]	uint32_t	0x10f0debc (Hex)	0x80038e30
(x)= rx_buf_table[1][15]	uint32_t	0x89674523 (Hex)	0x80038e34
(x)= rx_buf_table[1][16]	uint32_t	0x1efcdab (Hex)	0x80038e38
(x)= rx_buf_table[1][17]	uint32_t	0x78563412 (Hex)	0x80038e3c
(x)= rx_buf_table[1][18]	uint32_t	0xf0debc9a (Hex)	0x80038e40
(x)= rx_buf_table[1][19]	uint32_t	0x0 (Hex)	0x80038e44

Figure 3-6 Received data

Table 3-4 Transmit side specifications list

Contents			Specification
Channel used			channel 0
Port used			P1_0: CAN_CLK P1_1: CAN0RX P1_3: CAN0TX
Buffer used for transmit			Transmit/receive FIFO buffer 0
Bit rate			Nominal bit rate: 1Mbps, Data bit rate: 2Mbps
Transmit History			OFF
Transmit data setting			
	data 1	ID bit	ID: 0x00000001
		RTR bit	Data frame
		IDE bit	Extended ID
		DLC bit	64Byte
		FDSTS bit	CAN FD frame
		BRS bit	The bit rate in the data area changes
		Transmit data	0x12,0x34,0x56,0x78,0x9a,0xbc,0xde,0xf0, 0x23,0x45,0x67,0x89,0xab,0xcd,0xef,0x01, 0x34,0x56,0x78,0x9a,0xbc,0xde,0xf0,0x10, 0x45,0x67,0x89,0xab,0xcd,0xef,0x01,0x23, 0x56,0x78,0x9a,0xbc,0xde,0xf0,0x12,0x34, 0x67,0x89,0xab,0xcd,0xef,0x01,0x23,0x45, 0x78,0x9a,0xbc,0xde,0xf0,0x12,0x34,0x56, 0x89,0xab,0xcd,0xef,0x01,0x23,0x45,0x67
	data 2	ID bit	ID: 0x00000002
		RTR bit	Data frame
		IDE bit	Extended ID
		DLC bit	64Byte
		FDSTS bit	CAN FD frame
		BRS bit	The bit rate in the data area changes
		Transmit data	0x89,0xab,0xcd,0xef,0x01,0x23,0x45,0x67, 0x78,0x9a,0xbc,0xde,0xf0,0x12,0x34,0x56, 0x67,0x89,0xab,0xcd,0xef,0x01,0x23,0x45, 0x56,0x78,0x9a,0xbc,0xde,0xf0,0x12,0x34, 0x45,0x67,0x89,0xab,0xcd,0xef,0x01,0x23, 0x34,0x56,0x78,0x9a,0xbc,0xde,0xf0,0x10, 0x23,0x45,0x67,0x89,0xab,0xcd,0xef,0x01, 0x12,0x34,0x56,0x78,0x9a,0xbc,0xde,0xf0
Interrupt used			None

Table 3-5 Receive side specifications list

Contents			Specification
Channel used			channel 0
Port used			P1_0: CAN_CLK P1_1: CAN0RX P1_3: CAN0TX
Buffer used for receive			Receive FIFO buffer 0
Bit rate			Nominal bit rate: 1Mbps, Data bit rate: 2Mbps
Number of receive rules			2
Receive rules			
	rule 0	IDE bit	Extended ID
		RTR bit	Data frame
		LB bit	Receive rule target: When a message transmitted from another CAN node is received
		ID bit	Message ID to receive: 0x00000001
		IDE mask bit	The IDE bit is compared
		RTR mask bit	The RTR bit is compared
		ID mask bit	All ID bits are compared
		DLC bit	64Byte
		PTR bit	Receive rule label: 0x0123
		RMV bit	No receive buffer is used
		RMDP bit	No receive buffer selection
		FDP_TRFIFO bit	No transmit/receive FIFO buffer selection
		FDP_RXFIFO bit	Receive FIFO buffer 0 selection
	rule 1	IDE bit	Extended ID
		RTR bit	Data frame
		LB bit	Receive rule target: When a message transmitted from another CAN node is received
		ID bit	Message ID to receive: 0x00000002
		IDE mask bit	The IDE bit is compared
		RTR mask bit	The RTR bit is compared
		ID mask bit	All ID bits are compared
		DLC bit	64Byte
		PTR bit	Receive rule label: 0x0124
		RMV bit	No receive buffer is used
		RMDP bit	No receive buffer selection
		FDP_TRFIFO bit	No transmit/receive FIFO buffer selection
		FDP_RXFIFO bit	Receive FIFO buffer 0 selection
Interrupt used			Receive FIFO interrupt

3.2.2 CAN frame header information setting

When transmit, set header information such as ID and data length for each transmit data. The header information is set by R_CAN_TxSet function argument. As an example of header information setting, header information setting value in this operation example is described below.

Table 3-6 Header information setting value (data 1)

Argument	Corresponding register name	Set value	Description
p_frame-> ID	RSCFDnCFDCFIDk	0x00000001	Set ID
p_frame-> THDSE		0x00	Transmit history data is not stored in the buffer
p_frame-> RTR		0x00	Data frame
p_frame-> IDE		0x01	Extended ID
p_frame-> LBL	RSCFDnCFDCFPTRk	0x0000	Set label data Since the transmit history is not stored, the setting value of this bit is not used
p_frame-> DLC		0x0F	64 data bytes
p_frame-> FDSTS	RSCFDnCFDCFFDCSTSk	0x01	CAN FD frame
p_frame-> BRS		0x01	The bit rate of the data area changes
p_frame-> ESI		0x00	The setting value of this bit is not used because the channel error state is transmitted as the ESI bit

Table 3-7 Header information setting value (data 2)

Argument	Corresponding register name	Set value	Description
p_frame-> ID	RSCFDnCFDCFIDk	0x00000002	Set ID
p_frame-> THDSE		0x00	Transmit history data is not stored in the buffer
p_frame-> RTR		0x00	Data frame
p_frame-> IDE		0x01	Extended ID
p_frame-> LBL	RSCFDnCFDCFPTRk	0x0000	Set label data Since the transmit history is not stored, the setting value of this bit is not used
p_frame-> DLC		0x0F	64 data bytes
p_frame-> FDSTS	RSCFDnCFDCFFDCSTSk	0x01	CAN FD frame
p_frame-> BRS		0x01	The bit rate of the data area changes
p_frame-> ESI		0x00	The setting value of this bit is not used because the channel error state is transmitted as the ESI bit

3.2.3 Transmit/receive FIFO buffer setting

In this operation example, the transmit/receive FIFO buffer is used as a buffer for transmit. The transmit/receive FIFO buffer is set in "r_can_rz_config.h". As an example of transmit/receive FIFO buffer setting, transmit/receive FIFO buffer setting value in this operation example is described below.

Table 3-8 Transmit/receive FIFO buffer 0 setting macro setting value

Definition	Corresponding register name	Set value	Description
CFCC0_CFITT	RSCFDnCFDCFCCk	0x00	Set message transmission interval Set to 0 because the interval transmission function is not used
CFCC0_CFTML		0x00	Link to transmit buffer 0
CFCC0_CFITR		0x00	Transmit/receive FIFO interval timer resolution Set the initial value because the interval transmission function is not used
CFCC0_CFITSS		0x00	Transmit/receive FIFO interval timer clock source Set the initial value because the interval transmission function is not used
CFCC0_CFM		0x01	Transmit/receive FIFO mode: Transmit mode
CFCC0_CFIGCV		0x00	Set initial value because receive interrupt is disabled
CFCC0_CFIM		0x00	Set initial value because transmit interrupt is disabled
CFCC0_CFDC		0x01	Transmit/receive FIFO buffer depth: 4 messages
CFCC0_CFPLS		0x07	Transmit/receive FIFO buffer payload storage size: 64 bytes
CFCC0_CFTXIE		0x00	Disable transmit/receive FIFO transmit interrupt
CFCC0_CFRXIE		0x00	Disable transmit/receive FIFO receive interrupt

3.2.4 Receive rule setting

The receive rule is set in “r_can_rz_config.h”. As an example of receive rule setting, receive rule setting value in this operation example is described below.

Table 3-9 Receive rule count setting macro setting value

Definition	Corresponding register name	Set value	Description
CAN0_RX_RULE_NUM	RSCFDnCFDGAFLCFG0	0x02	Number of rules for channel 0: 2
CAN1_RX_RULE_NUM		0x00	Number of rules for channel 1: 0

Table 3-10 Receive rule related macro setting value (Receive rule 0)

Definition	Corresponding register name	Set value	Description
GAFLID_GAFLIDE_000	RSCFDnCFDGAFLIDj	0x01	Extended ID
GAFLID_GAFLRTR_000		0x00	Data frame
GAFLID_GAFLLB_000		0x00	Receive rule target: When a message transmitted from another CAN node is received
GAFLID_GAFLID_000		0x00000001	Message ID to receive: 0x00000001
GAFLM_GAFLIDEM_000	RSCFDnCFDGAFLMj	0x01	The IDE bit is compared
GAFLM_GAFLRTRM_000		0x01	The RTR bit is compared
GAFLM_GAFLIDM_000		0x1FFFFFFF	All ID bits are compared
GAFLP0_GAFLDLC_000	RSCFDnCFDGAFLP0_j	0x0F	64 data bytes
GAFLP0_GAFLPTR_000		0x0123	Set the label information
GAFLP0_GAFLRMV_000		0x00	No receive buffer is used
GAFLP0_GAFLRMDP_000		0x00	No receive buffer selection
GAFLP1_GAFLFDP_TRFIFO_000	RSCFDnCFDGAFLP1_j	0x00	No transmit/receive FIFO buffer selection
GAFLP1_GAFLFDP_RXFIFO_000		0x01	Receive FIFO buffer 0 selection

Table 3-11 Receive rule related macro setting value (Receive rule 1)

Definition	Corresponding register name	Set value	Description
GAFLID_GAFLIDE_001	RSCFDnCFDGAFLIDj	0x01	Extended ID
GAFLID_GAFLRTR_001		0x00	Data frame
GAFLID_GAFLLB_001		0x00	Receive rule target: When a message transmitted from another CAN node is received
GAFLID_GAFLID_001		0x00000002	Message ID to receive: 0x00000002
GAFLM_GAFLIDEM_001	RSCFDnCFDGAFLMj	0x01	The IDE bit is compared
GAFLM_GAFLRTRM_001		0x01	The RTR bit is compared
GAFLM_GAFLIDM_001		0x1FFFFFFF	All ID bits are compared
GAFLP0_GAFLDLC_001	RSCFDnCFDGAFLP0_j	0x0F	64 data bytes
GAFLP0_GAFLPTR_001		0x0124	Set the label information
GAFLP0_GAFLRMV_001		0x00	No receive buffer is used
GAFLP0_GAFLRMDP_001		0x00	No receive buffer selection
GAFLP1_GAFLFDP_TRFIFO_001	RSCFDnCFDGAFLP1_j	0x00	No transmit/receive FIFO buffer selection
GAFLP1_GAFLFDP_RXFIFO_001		0x01	Receive FIFO buffer 0 selection

3.2.5 Receive FIFO buffer setting

In this operation example, the receive FIFO buffer is used as a buffer for receive. The receive FIFO buffer is set in "r_can_rz_config.h". As an example of receive FIFO buffer setting, receive FIFO buffer setting value in this operation example is described below.

Table 3-12 Receive FIFO buffer 0 setting macro setting value

Definition	Corresponding register name	Set value	Description
RFCC0_RFIGCV	RSCFDnCFDRFCCx	0x01	Interrupt request generated when FIFO is 2/8 full
RFCC0_RFIM		0x00	An interrupt occurs when the condition set by the RFCC0_RFIGCV bits is met
RFCC0_RFDC		0x02	Receive FIFO buffer depth: 8 messages
RFCC0_RFPLS		0x07	Receive FIFO buffer payload storage size: 64 bytes
RFCC0_RFIE		0x01	Receive FIFO interrupt is enable

3.3 Operation example 2: gateway operation

3.3.1 Overview of operation

When the transmit/receive FIFO buffer is set to gateway mode, received messages can be transmit from any channel without going through the CPU. When using the gateway function in CAN FD mode, the frame to be transmitted can be replaced with a classical CAN frame or CAN FD frame. In this operation example, the classical CAN frame received on channel 0 is replaced with the CAN FD frame and transmitted from channel 1.

The above processing is all done in hardware, so there is no need to implement it in software. Software processing is only for common settings and gateway mode settings.

The opposite board program for operation check transmits classical CAN frame data from channel 0, and receives the converted CANFD frame data on channel 1.

Figure 3-7 shows the system configuration of this operation example.

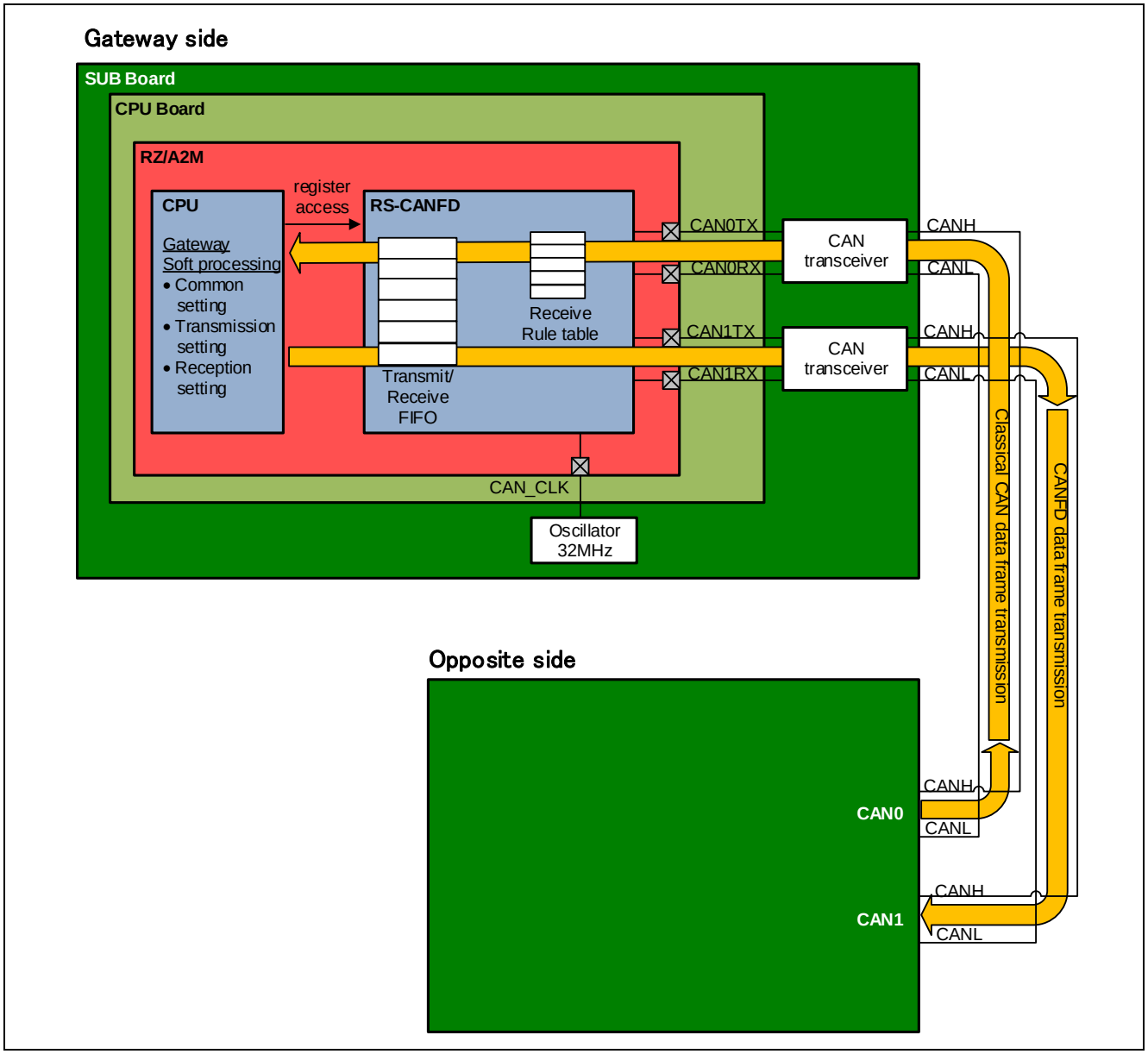


Figure 3-7 System configuration

The following shows the operation check procedure for gateway. Confirm the gateway operation by the program on the opposite board side.

- Method of operation
  1. Connect the channel 0 CANH pins (pin 1 of CAN1 connector) of each board. Also connect the channel 0 CANL pins (pin 3 of CAN1 connector) of each board.
     Connect the channel 1 CANH pins (pin 1 of CAN2 connector) of each board. Also connect the channel 1 CANL pins (pin 3 of CAN2 connector) of each board.
  2. Execute the program on the gateway side to put it in a receive waiting state.
  3. Set a breakpoint in the while statement with /* NG */ and /* OK */ comments at the end of the main function in the opposite board program. Also, execute the program on the opposite board side to transmit and receive.

- What to check
  1. Confirm that the transmitted data and received data match, enter the loop with the comment /* OK */, and exit the program.

- Check result

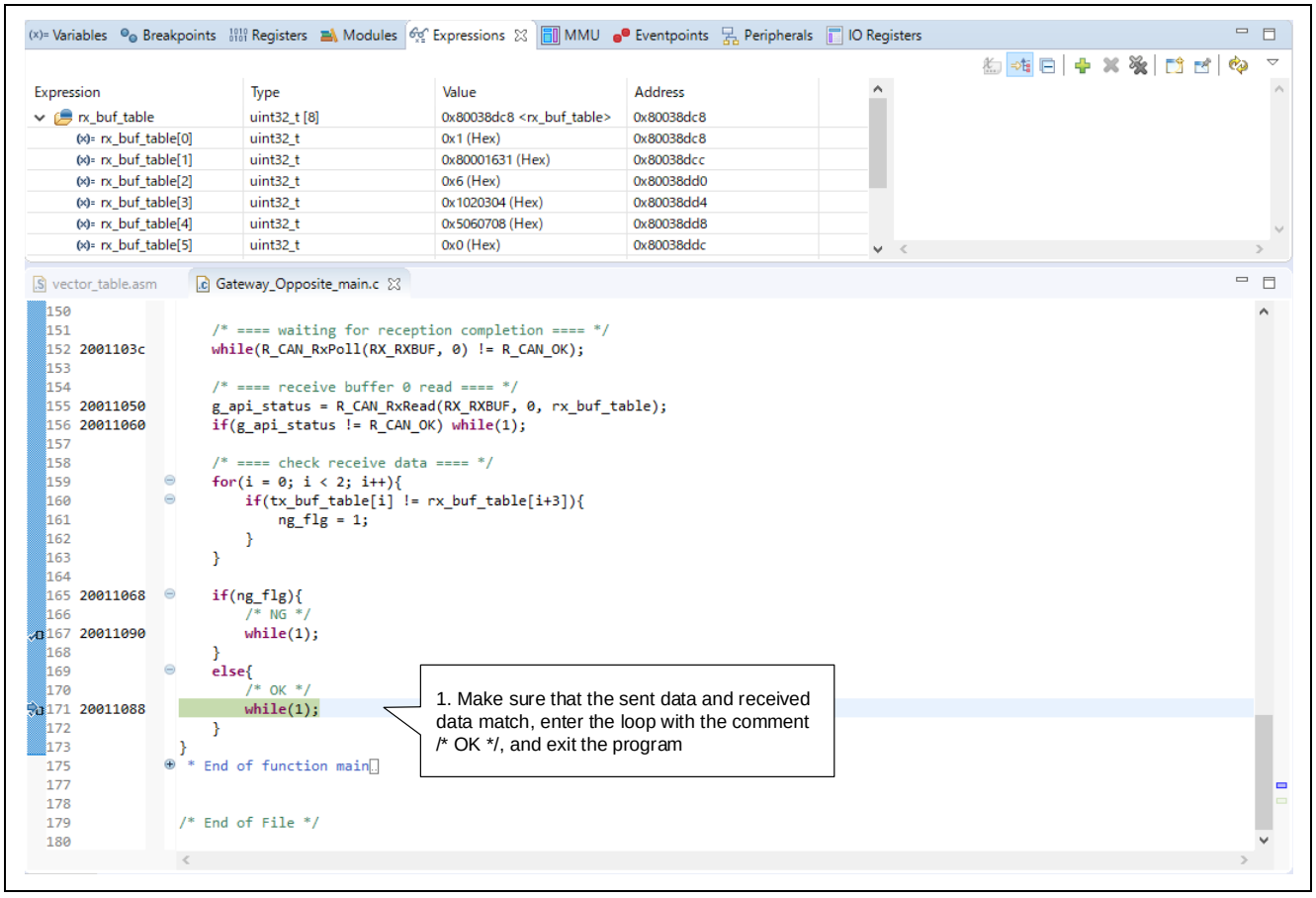


Figure 3-8 Check result

Table 3-13 Gateway specification list

Contents		Specification
Channel used		channel 0, channel 1
Port used		P1_0: CAN_CLK P1_1: CAN0RX P1_3: CAN0TX P2_0: CAN1RX P2_2: CAN1TX
Buffer used for transmit		Transmit/receive FIFO buffer 3 (gateway mode)
Bit rate		• Classical CAN: 1Mbps • CAN FD: Nominal bit rate: 1Mbps, Data bit rate: 2Mbps
Receive rules		
	IDE bit	Standard ID
	RTR bit	Data frame
	LB bit	Receive rule target: When a message transmitted from another CAN node is received
	ID bit	Message ID to receive: 0x00000000
	IDE mask bit	The IDE bit is not compared
	RTR mask bit	The RTR bit is not compared
	ID mask bit	ID is not compared
	DLC bit	The DLC bit is not compared
	PTR bit	Receive rule label: 0x0000
	RMV bit	No receive buffer is used
	RMDP bit	No receive buffer selection
	FDP_TRFIFO bit	Transmit/receive FIFO buffer 3 selection
	FDP_RXFIFO bit	No receive FIFO buffer selection
Interrupt used		None

Table 3-14 Opposite board side specification list

Contents		Specification
Channel used		channel 0, channel 1
Port used		P1_0: CAN_CLK P1_1: CAN0RX P1_3: CAN0TX P2_0: CAN1RX P2_2: CAN1TX
Buffer used for transmit		Transmit buffer 0
Buffer used for receive		Receive buffer 0
Bit rate		• Classical CAN: 1Mbps • CAN FD: Nominal bit rate: 1Mbps, Data bit rate: 2Mbps
Transmit History		OFF
Transmit data setting		
	ID bit	ID: 0x00000001
	RTR bit	Data frame
	IDE bit	Standard ID
	DLC bit	8Byte
	FDSTS bit	Classical CAN frame
	BRS bit	The bit rate in the data area does not change
	Transmit data	0x01020304,0x05060708
Receive rules		
	IDE bit	Standard ID
	RTR bit	Data frame
	LB bit	Receive rule target: When a message transmitted from another CAN node is received
	ID bit	Message ID to receive: 0x00000000
	IDE mask bit	The IDE bit is not compared
	RTR mask bit	The RTR bit is not compared
	ID mask bit	ID is not compared
	DLC bit	The DLC bit is not compared
	PTR bit	Receive rule label: 0x0000
	RMV bit	Receive buffer is used
	RMDP bit	Receive buffer 0 selection
	FDP_TRFIFO bit	No Transmit/receive FIFO buffer selection
	FDP_RXFIFO bit	No receive FIFO buffer selection
Interrupt used		None

3.3.2 Gateway settings

In this operation example, the transmit/receive FIFO buffer is used in gateway mode. The gateway is set in "r_can_rz_config.h". As an example of gateway setting, gateway setting value in this operation example is described below.

Table 3-15 Transmit/receive FIFO buffer 3 configuration macro setting value

Definition	Corresponding register name	Set value	Description
CFCC3_CFITT	RSCFDnCFDCFCCk	0x00	Set message transmission interval Set to 0 because the interval transmission function is not used
CFCC3_CFTML		0x00	Link to transmit buffer 0
CFCC3_CFITR		0x00	Transmit/receive FIFO interval timer resolution Set the initial value because the interval transmission function is not used
CFCC3_CFITSS		0x00	Transmit/receive FIFO interval timer clock source Set the initial value because the interval transmission function is not used
CFCC3_CFM		0x02	Transmit/receive FIFO mode: Gateway mode
CFCC3_CFIGCV		0x00	Set initial value because receive interrupt is disable
CFCC3_CFIM		0x00	Set initial value because transmit interrupt is disabled
CFCC3_CFDC		0x01	Transmit/receive FIFO buffer depth: 4 messages
CFCC3_CFPLS		0x07	Transmit/receive FIFO buffer payload storage size: 64 bytes
CFCC3_CFTXIE		0x00	Disable transmit/receive FIFO transmit interrupt
CFCC3_CFRXIE		0x00	Disable transmit/receive FIFO receive interrupt

Table 3-16 channel 1 CAN FD configuration macro setting value

Definition	Corresponding register name	Set value	Description
C1FDCFG_REFE	RSCFDnCFDCmFDCFG	0x00	Receive data edge filter prohibited
C1FDCFG_FDOE		0x00	FD only mode prohibited
C1FDCFG_TMME		0x00	Transmit buffer merge mode prohibited
C1FDCFG_GWBRS		0x01	Set the BRS bit of the received frame to “1” and transmit
C1FDCFG_GWFDF		0x01	Transmit received frames as CAN FD frames
C1FDCFG_GWEN		0x01	CAN-CAN FD gateway permission
C1FDCFG_ESIC		0x00	Always transmit channel error state as ESI bit in frame
C1FDCFG_EOCCFG		0x00	Error occurrence count method: All sent and received messages

3.3.3 Receive rule setting

The receive rule is set in “r_can_rz_config.h”. As an example of receive rule setting, receive rule setting value in this operation example is described below. In this operation example, all messages are received using the receive rule mask.

Table 3-17 Receive rule count setting macro setting value

Definition	Corresponding register name	Set value	Description
CAN0_RX_RULE_NUM	RSCFDnCFDGAFLCFG0	0x01	Number of rules for channel 0: 1
CAN1_RX_RULE_NUM		0x00	Number of rules for channel 1: 0

Table 3-18 Receive rule related macro setting value

Definition	Corresponding register name	Set value	Description
GAFLID_GAFLIDE_000	RSCFDnCFDGAFLIDj	0x00	Standard ID
GAFLID_GAFLRTR_000		0x00	Data frame
GAFLID_GAFLLB_000		0x00	Receive rule target: When a message transmitted from another CAN node is received
GAFLID_GAFLID_000		0x00000000	Message ID to receive: 0x00000000
GAFLM_GAFLIDEM_000	RSCFDnCFDGAFLMj	0x00	The IDE bit is not compared
GAFLM_GAFLRTRM_000		0x00	The RTR bit is not compared
GAFLM_GAFLIDM_000		0x00000000	ID is not compared
GAFLP0_GAFLDLC_000	RSCFDnCFDGAFLP0_j	0x00	The DLC bit is not compared
GAFLP0_GAFLPTR_000		0x0000	Receive rule label: 0x0000
GAFLP0_GAFLRMV_000		0x00	No receive buffer is used
GAFLP0_GAFLRMDP_000		0x00	No receive buffer selection
GAFLP1_GAFLFDP_TRFIFO_000	RSCFDnCFDGAFLP1_j	0x08	Transmit/receive FIFO buffer 3 selection
GAFLP1_GAFLFDP_RXFIFO_000		0x00	No receive FIFO buffer selection

3.4 Operation example 3: external Loopback operation

3.4.1 Overview of operation

In this operation example, the external loopback mode is used, and the data transmitted from the channel 0 transmit port is received by the channel 0 receive port. Also, compare the transmitted data with the received data to confirm that there is no abnormality in transmit / receive. A transmit buffer is used for transmit, and a receive buffer is used for receive. Receive completion is monitored by polling. Figure 3-9 shows the system configuration of this operation example.

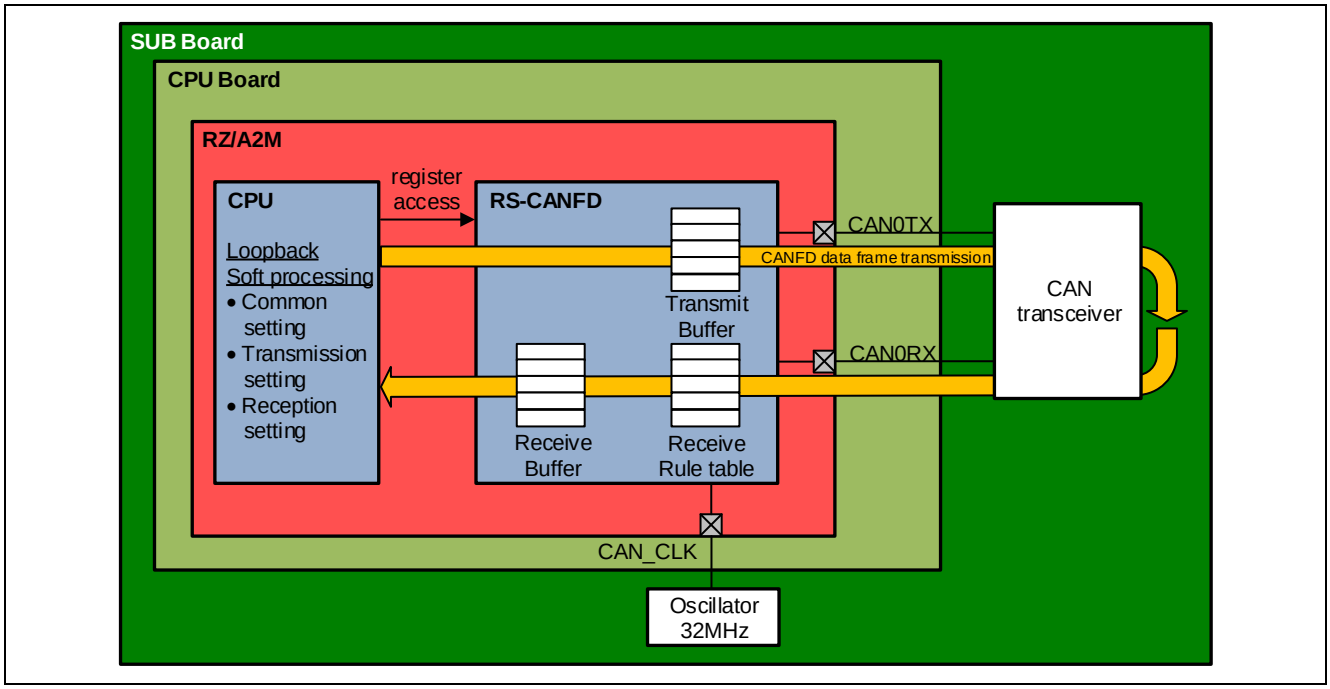


Figure 3-9 System configuration

The procedure for checking the loopback operation is shown below.

- Method of operation
  1. No port connection is required.
  2. Set a breakpoint in the while statement with /* NG */ and /* OK */ comments at the end of the main function in the external loopback test program. Also, execute the program.

- What to check
  1. Confirm that the transmitted data and received data match, enter the loop with the comment /* OK */, and exit the program.

- Check result

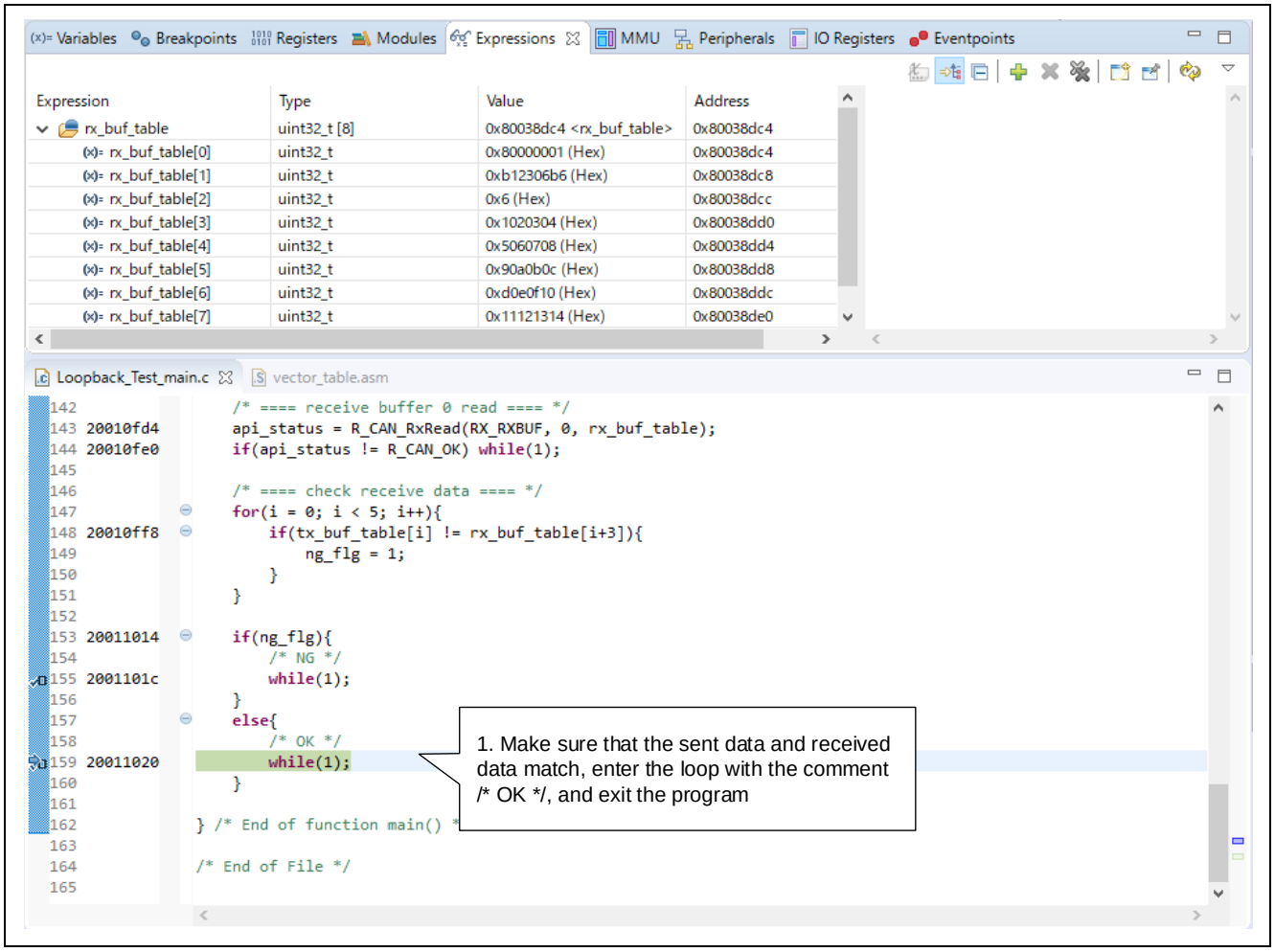


Figure 3-10 Check result

Table 3-19 Loopback specification list

Contents		Specification
Channel used		channel 0 (Self-test mode 0 (external loopback mode))
Port used		P1_0: CAN_CLK P1_1: CAN0RX P1_3: CAN0TX
Buffer used for transmit		Transmit buffer 0
Buffer used for receive		Receive buffer 0
Bit rate		Nominal bit rate: 1Mbps, Data bit rate: 2Mbps
Transmit History		OFF
Transmit data setting		
	ID bit	ID：0x00000001
	RTR bit	Data frame
	IDE bit	Extended ID
	DLC bit	20Byte
	FDSTS bit	CAN FD frame
	BRS bit	The bit rate in the data area changes
	Transmit data	0x01020304,0x05060708, 0x090A0B0C,0x0D0E0F10, 0x11121314
Receive rules		
	IDE bit	Extended ID
	RTR bit	Data frame
	LB bit	Receive rule target: When a message transmitted from another CAN node is received
	ID bit	Message ID to receive: 0x00000001
	IDE mask bit	The IDE bit is compared
	RTR mask bit	The RTR bit is compared
	ID mask bit	All ID bits are compared
	DLC bit	The DLC bit is not compared
	PTR bit	Receive rule label: 0x0123
	RMV bit	Receive buffer is used
	RMDP bit	Receive buffer 0 is used
	FDP_TRFIFO bit	No transmit/receive FIFO buffer selection
	FDP_RXFIFO bit	No receive FIFO buffer selection
Interrupt used		None

3.4.2 CAN frame header information setting

When transmit, set header information such as ID and data length for each transmit data. The header information is set by R_CAN_TxSet function argument. As an example of header information setting, header information setting value in this operation example is described below.

Table 3-20 Header information setting value

Argument	Corresponding register name	Set value	Description
p_frame-> ID	RSCFDnCFDTMIDp	0x00000001	Set ID
p_frame-> THDSE		0x00	Transmit history data is not stored in the buffer
p_frame-> RTR		0x00	Data frame
p_frame-> IDE		0x01	Extended ID
p_frame-> LBL	RSCFDnCFDTMPTRp	0x0000	Set label data Since the transmit history is not stored, the setting value of this bit is not used
p_frame-> DLC		0x0B	20 data bytes
p_frame-> FDSTS	RSCFDnCFDTMFDCTRp	0x01	CAN FD frame
p_frame-> BRS		0x01	The bit rate of the data area changes
p_frame-> ESI		0x00	The setting value of this bit is not used because the channel error state is transmitted as the ESI bit

3.4.3 Transmit buffer setting

In this operation example, the transmit buffer is used as a buffer for transmit. The transmit buffer is set in "r_can_rz_config.h". As an example of transmit buffer setting, transmit buffer setting value in this operation example is described below.

Table 3-21 Transmit buffer configuration macro setting value

Definition	Corresponding register name	Set value	Description
TMIEC0_TMIE0	RSCFDnCFDTMIECy	0x00	Transmit buffer interrupt is disabled

3.4.4 Receive rule setting

The receive rule is set in “r_can_rz_config.h”. As an example of receive rule setting, receive rule setting value in this operation example is described below. In this operation example, since the DLC bit is not compared, all data bytes of the received message are stored in the buffer.

Table 3-22 Receive rule count setting macro setting value

Definition	Corresponding register name	Set value	Description
CAN0_RX_RULE_NUM	RSCFDnCFDGAFLCFG0	0x01	Number of rules for channel 0: 1
CAN1_RX_RULE_NUM		0x00	Number of rules for channel 1: 0

Table 3-23 Receive rule related macro setting value

Definition	Corresponding register name	Set value	Description
GAFLID_GAFLIDE_000	RSCFDnCFDGAFLIDj	0x01	Extended ID
GAFLID_GAFLRTR_000		0x00	Data frame
GAFLID_GAFLLB_000		0x00	Receive rule target: When a message transmitted from another CAN node is received
GAFLID_GAFLID_000		0x00000001	Message ID to receive: 0x00000001
GAFLM_GAFLIDEM_000	RSCFDnCFDGAFLMj	0x01	The IDE bit is compared
GAFLM_GAFLRTRM_000		0x01	The RTR bit is compared
GAFLM_GAFLIDM_000		0x1FFFFFFF	All ID bits are compared
GAFLP0_GAFLDLC_000	RSCFDnCFDGAFLP0_j	0x00	The DLC bit is not compared
GAFLP0_GAFLPTR_000		0x0123	Set the label information
GAFLP0_GAFLRMV_000		0x01	Receive buffer is used
GAFLP0_GAFLRMDP_000		0x00	Receive buffer 0 is used
GAFLP1_GAFLFDP_TRFIFO_000	RSCFDnCFDGAFLP1_j	0x00	No transmit/receive FIFO buffer selection
GAFLP1_GAFLFDP_RXFIFO_000		0x00	No receive FIFO buffer selection

3.4.5 Receive buffer setting

In this operation example, the receive buffer is used as a buffer for receive. The receive buffer is set in "r_can_rz_config.h". As an example of receive buffer setting, receive buffer setting value in this operation example is described below.

Table 3-24 Receive buffer 0 configuration macro setting value

Definition	Corresponding register name	Set value	Description
RMNB_RMPLS	RSCFDnCFDRMNB	0x03	Receive buffer payload storage size: 20bytes
RMNB_NRXMB		0x01	Number of receive buffers: 1

4. Restrictions

There are no restrictions in this application note.

5. Precautions

The Precautions of this application note are shown as follow.

Table 5-1 Precautions

No.	Type	Description
1	Environment	If it is happened a build error while building the project of this application note as it is, the setting of environment may be incorrect. Check following items: • Follow section 3 of “RZ/A2M Software Package Quick Start Guide”(R01QS0027) • Install e2 studio v7.4 or later again.
2	Environment	To avoid build error, expand the project to the folder with short full-path.
3	Environment	To avoid build error, expand the project to the folder without multi-byte character.
4	Environment	This application note includes elf-formatted boot loader. Therefore, the project to generate the boot loader is not bundled. Following application note includes the boot loader project. To get it, please download from Renesas site: RZ/A2M Group Example of Booting from Serial Flash Memory (R01AN4333)
5	Environment	The RZ/A2M SUB board does not mounted a CAN transceiver and an oscillator for the CAN clock, so it must be mounted when using CAN. For details, refer to 2.1 System configuration.

6. Used open source software and licenses

Open source software used in this application note and license of them are shown as following:

- newlib is used under the license described in following site:
  https://www.sourceware.org/newlib/COPYING.NEWLIB

7. Reference Documents

User's Manual: Hardware

RZ/A2M Group User's Manual: Hardware

The latest version can be downloaded from the Renesas Electronics website.

RTK7921053C00000BE (RZ/A2M CPU board) User's Manual

The latest version can be downloaded from the Renesas Electronics website.

RTK79210XXB00000BE (RZ/A2M SUB board) User's Manual

The latest version can be downloaded from the Renesas Electronics website.

ARM Architecture Reference Manual ARMv7-A and ARMv7-R edition Issue C

The latest version can be downloaded from the ARM website.

ARM CortexTM-A9 (Revision: r4p1) Technical Reference Manual

The latest version can be downloaded from the ARM website.

ARM Generic Interrupt Controller Architecture Specification - Architecture version 2.0

The latest version can be downloaded from the ARM website.

ARM CoreLinkTM Level 2 Cache Controller L2C-310 (Revision: r3p3) Technical Reference Manual

The latest version can be downloaded from the ARM website.

Technical Update/Technical News

The latest information can be downloaded from the Renesas Electronics website.

User's Manual: Development Tools

Integrated development environment e2studio User’s Manual can be downloaded from the Renesas Electronics website.

The latest version can be downloaded from the Renesas Electronics website.

Revision History

Rev.	Date	Description	
		Page	Summary
1.0	Jan. 10, 2020	—	First edition issued

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.

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 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.
5. 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.
6. 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.
7. 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.
8. 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.
9. 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.
10. 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.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. 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.4.0-1 November 2017)

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/.