RX Family

Read/Write Operations in SDRAM Using the SDRAMC

Introduction

The SDRAM interface in the RX Family can be connected directly to an SDRAM up to 128 Mbytes (1024 Mbits) with a CAS latency of 1 to 3 cycles. This application note describes a method of using the RX65N Group and RX72M Group to read from and write to a 128 Mbit SDRAM (Micron).

This application note describes the methods of read and write operations on the 128 Mbit SDRAM (manufactured by Micron or Alliance Memory) mounted on the following MCUs:

- RX65N
- RX72M
- RX72N
- RX671

Target Device

- RX Family MCU loaded with the SDRAM area controller

Contents

1. Specifications

The following processing is implemented in the sample code of this application note.

- An SDRAMC is used to perform a read from, or a write to, the following 128 Mbit SDRAMs:
  - MT48LC8M16A2P-6A manufactured by Micron (2 M words × 16 bits × 4 banks)
  - MT48LC4M32B2P-6A manufactured by Micron (1 M words × 32 bits × 4 banks)
  - AS4C8M16SA-7TCN manufactured by Alliance Memory (2 M words × 16 bits × 4 banks)
- After the MCU is restored from the reset state, the SDRAM is initialized and increment data is written to the 128-Mbit SDRAM area. After the write to the entire area is complete, the written value is read.
- The operation of the sample code differs as follows depending on whether the read value and expected value match:
  - Renesas Starter Kit+: If the read value and expected value match, the sample code turns on LED0. If they do not match, the sample code turns on LED1.
  - EK-RX671: If the read value and expected value match, the sample code turns on LED1. If they do not match, the sample code turns on LED2.

Table 1.1 shows Peripheral Functions and Their Applications, Table 1.2 shows SDRAM (MT48LC8M16A2P-6A) Specifications, Table 1.3 shows SDRAM (MT48LC4M32B2P-6A) Specifications, and Table 1.4 shows Specifications of SDRAM (AS4C8M16SA-7TCN).

Table 1.1 Peripheral Functions and Their Applications

Peripheral Function	Application
External bus	Connects to the SDRAM.
I/O ports	Turn on LEDs.
CMT0	Timer for wait time

Table 1.2 SDRAM (MT48LC8M16A2P-6A) Specifications

Item	Description
Product	Micron MT48LC8M16A2P-6A
Configuration	2 M-word × 16 bits × 4 banks
Size	128 Mbits
Row addressing	A11 to A0
Column addressing	A8 to A0
Auto refresh cycle	4096 refresh cycles every 64 ms
CAS latency	2 or 3 cycles
Initial auto refresh	2 times
AUTO REFRESH period (tRFC)	60 ns (min.)
WRITE recovery time	22.67 ns (min.) (1)
PRECHARGE command period (tRP)	18 ns (min.)
ACTIVE-to-PRECHARGE command period (tRAS)	42 ns (min.)
ACTIVE-to-READ or WRITE delay (tRCD)	18 ns (min.)

Note: 1. The WRITE recovery time is 1 clock cycle + 6 ns. In this application, SDCLK is 60 MHz. Therefore, 1 clock cycle is 16.67 ns.

Table 1.3 SDRAM (MT48LC4M32B2P-6A) Specifications

Item	Description
Product	Micron MT48LC4M32B2P-6A
Configuration	1 M-word × 32 bits × 4 banks
Size	128 Mbits
Row addressing	A11 to A0
Column addressing	A7 to A0
Auto refresh cycle	4096 refresh cycles every 64 ms
CAS latency	2 or 3 cycles
Initial auto refresh	2 times
AUTO REFRESH period (tRFC)	60 ns (min.)
WRITE recovery time	19.5 ns (min.) (1)
PRECHARGE command period (tRP)	18 ns (min.)
ACTIVE-to-PRECHARGE command period (tRAS)	42 ns (min.)
ACTIVE-to-READ or WRITE delay (tRCD)	18 ns (min.)

Note: 1. The WRITE recovery time is 1 clock cycle + 7 ns. In this application, SDCLK is 80 MHz. Therefore, 1 clock cycle is 12.5 ns.

Table 1.4 Specifications of SDRAM (AS4C8M16SA-7TCN)

Item	Description
Product	AS4C8M16SA-7TCN (manufactured by Alliance Memory)
Configuration	2 M words × 16 bits × 4 banks
Size	128 Mbits
Row addressing	A11 to A0
Column addressing	A8 to A0
Auto refresh cycle	4,096 refresh cycles per 64 ms
CAS latency	2 or 3 cycles
Initial auto refresh	2 times
Row cycle time (tRC)	63 ns (min.) (same bank)
WRITE recovery time (tWR)	14 ns (min.) (1)
PRECHARGE command period (tRP)	21 ns (min.)
ACTIVE-to-PRECHARGE command period (tRAS)	42 ns (min.)
ACTIVE-to-READ or WRITE delay (tRCD)	21 ns (min.)

Note: 1. The WRITE recovery time is 1 clock cycle + 6 ns. In this application, SDCLK is 60 MHz. Therefore, 1 clock cycle is 16.67 ns.

2. Operation Confirmation Conditions

The sample code accompanying this application note has been run and confirmed under the conditions below.

Table 2.1 Operation Confirmation Conditions (RX65N)

Item	Contents
MCU used	R5F565NEDDFC (RX65N Group)
Operating frequencies	- Main clock: 24 MHz - PLL: 240 MHz (main clock divided by 1 and multiplied by 10) - System clock (ICLK): 120 MHz (PLL divided by 2) - Peripheral module clock A (PCLKA): 120 MHz (PLL divided by 2) - Peripheral module clock B (PCLKB): 60 MHz (PLL divided by 4) - Peripheral module clock C (PCLKC): 60 MHz (PLL divided by 4) - Peripheral module clock D (PCLKD): 60 MHz (PLL divided by 4) - SDRAM clock (SDCLK): 60 MHz (PLL divided by 4)
Operating voltage	3.3 V
Integrated development environment	Renesas Electronics Corporation e^2 studio 2020-04
C compiler	Renesas Electronics Corporation C/C++ Compiler Package for RX Family V.3.02
	Compile options Default settings of integrated development environment
iodefine.h version	Version 2.30
Endian	Little endian
Operating mode	On-Chip ROM Enabled Extended Mode
Processor mode	Supervisor mode
Sample code version	Version 1.00
Board used	Renesas Starter Kit+ for RX65N-2MB (product part no.:RTK50565N2S80000BE) (SDRAM: MT48LC8M16A2P-6A)

Table 2.2 Operation Confirmation Conditions (RX72M)

Item	Contents
MCU used	R5F572MNDDBD (RX72M Group)
Operating frequencies	- Main clock: 24 MHz - PLL: 240 MHz (main clock divided by 1 and multiplied by 10) - System clock (ICLK): 240 MHz (PLL divided by 1) - Peripheral module clock A (PCLKA): 120 MHz (PLL divided by 2) - Peripheral module clock B (PCLKB): 60 MHz (PLL divided by 4) - Peripheral module clock C (PCLKC): 60 MHz (PLL divided by 4) - Peripheral module clock D (PCLKD): 60 MHz (PLL divided by 4) - SDRAM clock (SDCLK): 80 MHz (PLL divided by 3)
Operating voltage	3.3 V
Integrated development environment	Renesas Electronics Corporation e^2 studio 2020-04
C compiler	Renesas Electronics Corporation C/C++ Compiler Package for RX Family V.3.02
	Compile options Default settings of integrated development environment
iodefine.h version	Version 1.00C
Endian	Little endian
Operating mode	On-Chip ROM Enabled Extended Mode
Processor mode	Supervisor mode
Sample code version	Version 1.00
Board used	Renesas Starter Kit+ for RX72M (product part no.: RTK5572MNDS10000BE) (SDRAM: MT48LC4M32B2P-6A)

Table 2.3 Operation Confirmation Conditions (RX671: [RSK-RX671])

Item	Description
MCU used	R5F5671EHDFB (RX671 Group)
Operating frequency	- Main clock: 24 MHz - PLL: 240 MHz (main clock divided by 1 and multiplied by 10) - System clock (ICLK): 120 MHz (PLL divided by 2) - Peripheral module clock A (PCLKA): 120 MHz (PLL divided by 2) - Peripheral module clock B (PCLKB): 60 MHz (PLL divided by 4) - Peripheral module clock C (PCLKC): 60 MHz (PLL divided by 4) - Peripheral module clock D (PCLKD): 60 MHz (PLL divided by 4) - SDCLK: 60 MHz (PLL divided by 4)
Operating voltage	3.3 V (from the power supply)
Integrated development environment	Renesas Electronics Corporation e^2 studio Version 2023-10
C compiler	Renesas Electronics Corporation C/C++ Compiler Package for RX Family V.3.05
	Compile options Default settings of the integrated development environment
iodefine.h version	Version 1.00A
Endian	On-chip ROM enabled expanded mode
Operating mode	Supervisor mode
Processor mode	Ver.1.10
Sample code version	3.3 V (from the power supply)
Board used	Renesas Starter Kit+ for RX671 (product part no.: RTK55671EHS1000BE) (SDRAM: MT48LC8M16A2P-6A)

Table 2.4 Operation Confirmation Conditions (RX671: [EK-RX671])

Item	Description
MCU used	R5F5671EHDFB (RX671 Group)
Operating frequency	- Main clock: 24 MHz - PLL: 240 MHz (main clock divided by 1 and multiplied by 10) - System clock (ICLK): 120 MHz (PLL divided by 2) - Peripheral module clock A (PCLKA): 120 MHz (PLL divided by 2) - Peripheral module clock B (PCLKB): 60 MHz (PLL divided by 4) - Peripheral module clock C (PCLKC): 60 MHz (PLL divided by 4) - Peripheral module clock D (PCLKD): 60 MHz (PLL divided by 4) - SDCLK: 60 MHz (PLL divided by 4)
Operating voltage	3.3 V (from the power supply)
Integrated development environment	Renesas Electronics Corporation e^2 studio Version 2023-10
C compiler	Renesas Electronics Corporation C/C++ Compiler Package for RX Family V.3.05
	Compile options Default settings of the integrated development environment
iodefine.h version	Version 1.00A
Endian	On-chip ROM enabled expanded mode
Operating mode	Supervisor mode
Processor mode	Ver.1.10
Sample code version	3.3 V (from the power supply)
Board used	EK-RX671 (product part no.: RTK5EK6710S00001BE) (SDRAM: AS4C8M16SA-7TCN)

Table 2.5 Operation Confirmation Conditions (RX72N)

Item	Description
MCU used	R5F572NNDDBD (RX72N Group)
Operating frequency	- Main clock: 24 MHz - PLL: 240 MHz (main clock divided by 1 and multiplied by 10) - System clock (ICLK): 240 MHz (PLL divided by 1) - Peripheral module clock A (PCLKA): 120 MHz (PLL divided by 2) - Peripheral module clock B (PCLKB): 60 MHz (PLL divided by 4) - Peripheral module clock C (PCLKC): 60 MHz (PLL divided by 4) - Peripheral module clock D (PCLKD): 60 MHz (PLL divided by 4) - SDCLK: 60 MHz (PLL divided by 4)
Operating voltage	3.3 V (from the power supply)
Integrated development environment	Renesas Electronics Corporation e^2 studio Version 2023-10
C compiler	Renesas Electronics Corporation C/C++ Compiler Package for RX Family V.3.05
	Compile options Default settings of the integrated development environment
iodefine.h version	Version 1.00A
Endian	On-chip ROM enabled expanded mode
Operating mode	Supervisor mode
Processor mode	Ver.1.10
Sample code version	3.3 V (from the power supply)
Board used	Renesas Starter Kit+ for RX72N (product part no.: RTK5572NNHS10000BE) (SDRAM: MT48LC8M16A2P-6A)

3. Reference Application Note

For additional information associated with this document, refer to the following application note.

- RX65N Group, RX651 Group Initial Setting (R01AN3034)
- RX72M Group Initial Setting (R01AN4530)
- RX671 Group Initial Setting Example (R01AN5551)
- RX72N Group Initial Setting Example (R01AN4970)

However the latest version is always recommended. Visit the Renesas Electronics Corporation website to check and download the latest version.

4. Peripheral Function

This chapter provides supplementary information on the SDRAMC. Refer to the User's Manual: Hardware for basic information.

4.1 Output Operation in the SDRAMC

With the RX65N SDRAMC, when an SDRAM command is issued, pin states associated with the SDRAM will be changed after a certain time of delay from the rising of SDCLK. The command is determined on the next rising edge of SDCLK. Refer to the Electrical Characteristics chapter in the User's Manual: Hardware for details of the output delay time for each pin.

Figure 4.1 shows the Output on Pins Associated with the SDRAM and the Timing of Command Determination.

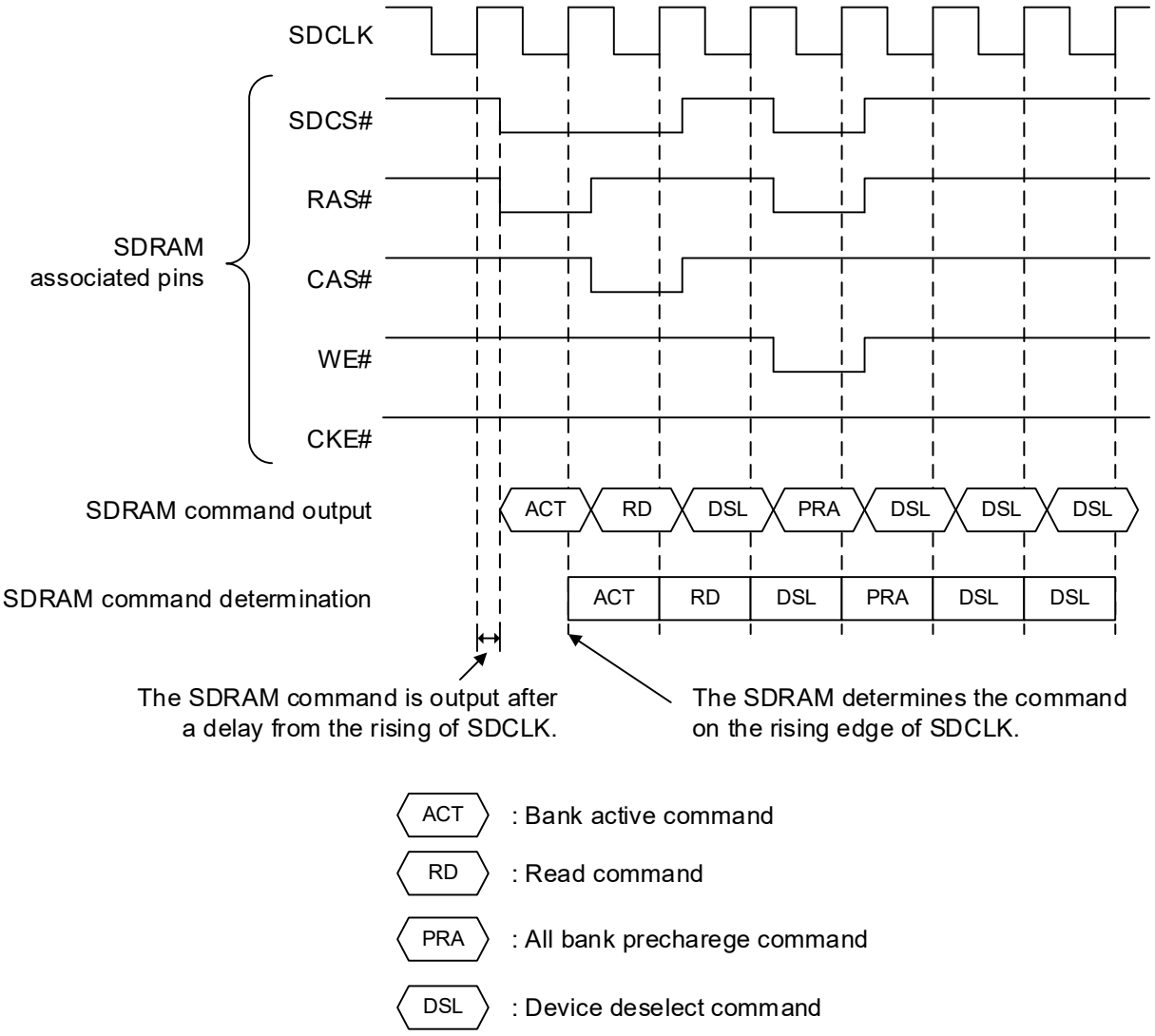
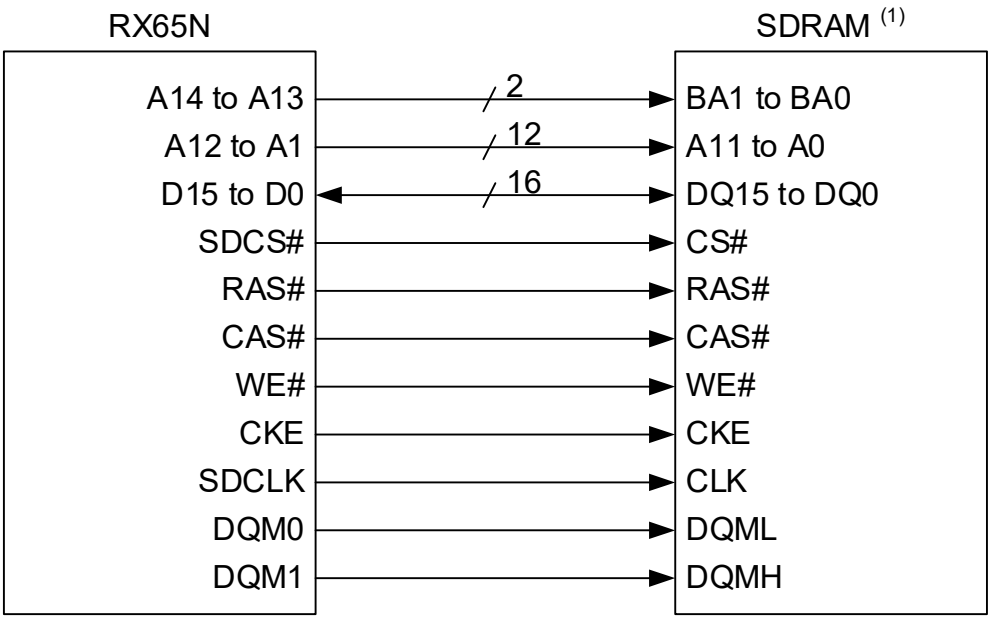


Figure 4.1 Output on Pins Associated with the SDRAM and the Timing of Command Determination

5. Hardware

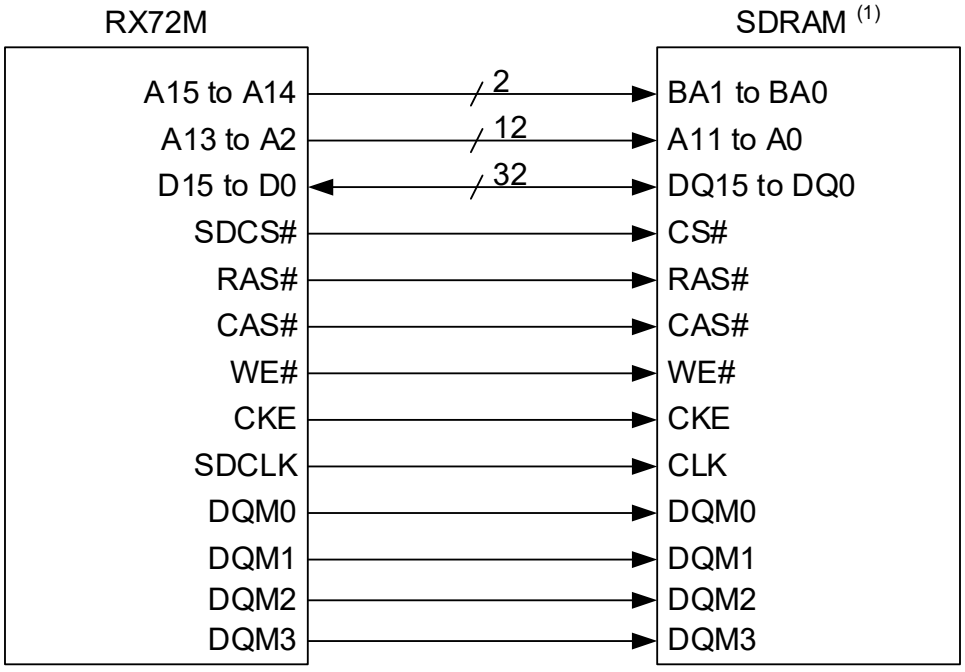
5.1 Hardware Configuration

Figure 5.1 to Figure 5.4 show the Connection Example.



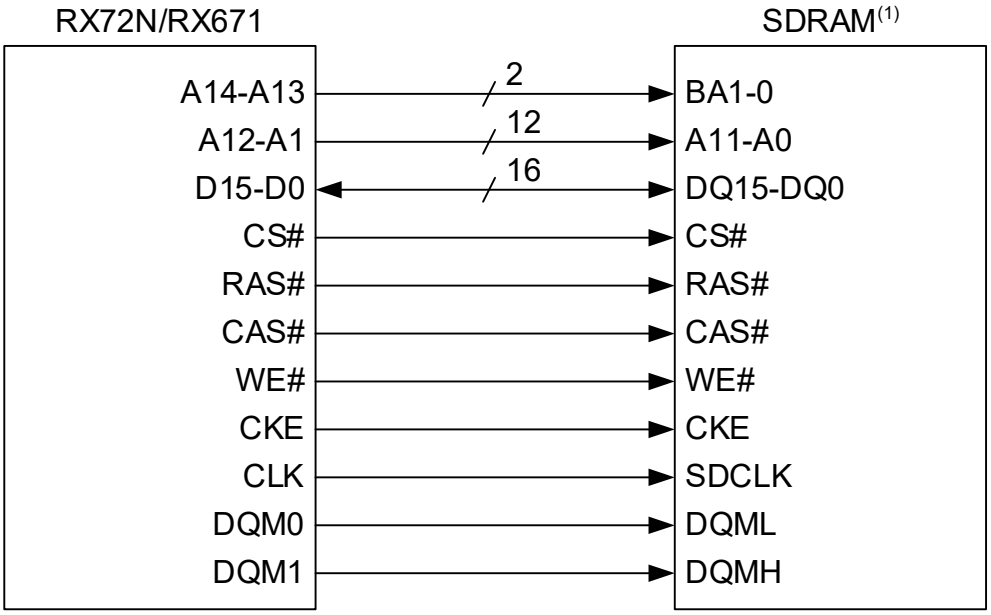
Note:
1. SDRAM: MT48LC8M16A2P-6A (2 M-word × 16 bits × 4 banks)

Figure 5.1 MT48LC8M16A2P-6A Connection Example



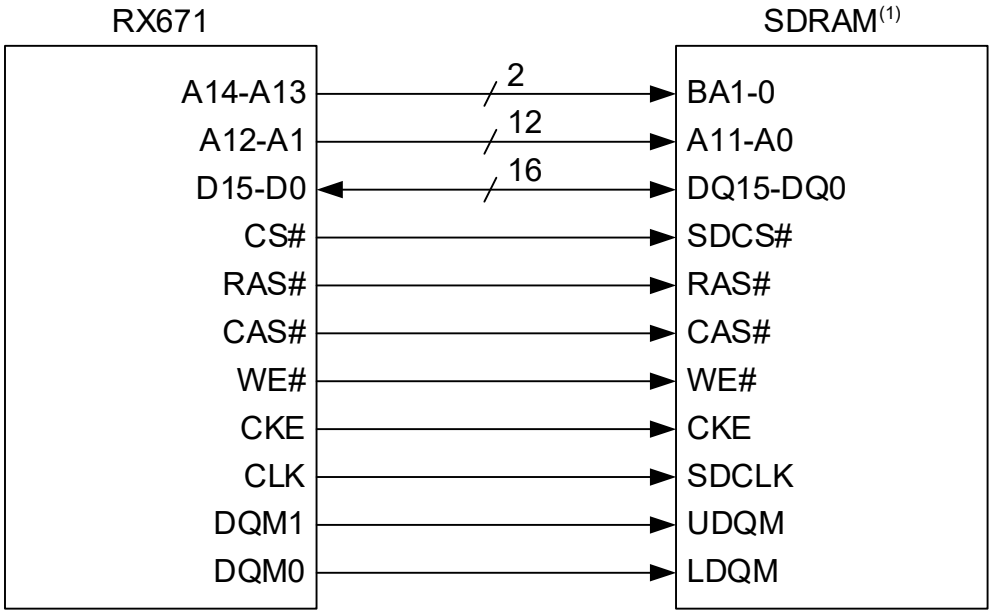
Note:
1. SDRAM: MT48LC4M32B2P-6A (1 M-word × 32 bits × 4 banks)

Figure 5.2 MT48LC4M32B2P-6A Connection Example



Note:
1. MT48LC8M16A2P-6A (2 M words x 16 bits x 4 banks) is used.

Figure 5.3 MT48LC8M16A2P-6A Connection Example



Note:
1. AS4C8M16SA-7TCN (2 M words x 16 bits x 4 banks) is used.

Figure 5.4 AS4C8M16SA-7TCN Connection Example

5.2 Pins Used

Table 5.1 to Table 5.5 list the Pins Used and Their Functions.

Table 5.1 Pins Used and Their Functions(RX65N)

Pin Name	I/O	Function
P73	Output	Outputs for LED0 (verification succeeded).
PG7	Output	Outputs for LED1 (verification error).
PA7 to PA0	Output	Outputs an address (A7 to A0).
PB6 to PB0	Output	Outputs an address (A14 to A8.)
PD7 to PD0	I/O	Outputs data (D7 to D0).
PE7 to PE0	I/O	Outputs data (D15 to D8.)
P70	Output	Outputs the SDCLK signal.
P61	Output	Outputs the SDCS# signal.
P62	Output	Outputs the RAS# signal.
P63	Output	Outputs the CAS# signal.
P64	Output	Outputs the WE# signal.
P65	Output	Outputs the CKE signal.
P66	Output	Outputs the DQM0 signal.
P67	Output	Outputs the DQM1 signal.

Table 5.2 Pins Used and Their Functions(RX72M)

Pin Name	I/O	Function
P42	Output	Outputs for LED0 (verification succeeded).
PH0	Output	Outputs for LED1 (verification error).
PA7 to PA2	Output	Outputs an address (A7 to A2).
PB7 to PB0	Output	Outputs an address (A15 to A8.)
PD7 to PD0	I/O	Outputs data (D7 to D0).
PE7 to PE0	I/O	Outputs data (D15 to D8.)
P97 to P90	I/O	Outputs data (D23 to D16).
PG7 to PG0	I/O	Outputs data (D31 to D24.)
P70	Output	Outputs the SDCLK signal.
P61	Output	Outputs the SDCS# signal.
P62	Output	Outputs the RAS# signal.
P63	Output	Outputs the CAS# signal.
P64	Output	Outputs the WE# signal.
P65	Output	Outputs the CKE signal.
P66	Output	Outputs the DQM0 signal.
P67	Output	Outputs the DQM1 signal.
PA0	Output	Outputs the DQM2 signal.
PA1	Output	Outputs the DQM3 signal.

Table 5.3 Pins Used and Their Functions (RSK-RX671)

Pin Name	I/O	Function
P17	Output	Outputs for LED0 (verification succeeded).
PF9	Output	Outputs for LED1 (verification error).
PA7-PA0	Output	Outputs an address (A7 to A0).
PB6-PB0	Output	Outputs an address (A14 to A8).
PD7-PD0	I/O	Inputs/outputs data (D7 to D0).
PE7-PE0	I/O	Inputs/outputs data (D15 to D8).
P70	Output	SDCLK pin output
P61	Output	SDCS# pin output
P62	Output	RAS# pin output
P63	Output	CAS# pin output
P64	Output	WE# pin output
P65	Output	CKE pin output
P66	Output	DQM0 pin output
P67	Output	DQM1 pin output

Table 5.4 Pins Used and Their Functions (EK-RX671)

Pin Name	I/O	Function
P73	Output	Outputs for LED1 (verification succeeded).
PG7	Output	Outputs for LED2 (verification error).
PA7-PA0	Output	Outputs an address (A7 to A0).
PB6-PB0	Output	Outputs an address (A14 to A8).
PD7-PD0	I/O	Inputs/outputs data (D7 to D0).
PE7-PE0	I/O	Inputs/outputs data (D15 to D8).
P70	Output	SDCLK pin output
P61	Output	SDCS# pin output
P62	Output	RAS# pin output
P63	Output	CAS# pin output
P64	Output	WE# pin output
P65	Output	CKE pin output
P66	Output	DQM0 pin output
P67	Output	DQM1 pin output

Table 5.5 Pins Used and Their Functions (RSK-RX72N)

Pin Name	I/O	Function
P71	Output	Outputs for LED0 (verification succeeded).
PH6	Output	Outputs for LED1 (verification error).
PA7-PA0	Output	Outputs an address (A7 to A0).
PB6-PB0	Output	Outputs an address (A14 to A8).
PD7-PD0	I/O	Inputs/outputs data (D7 to D0).
PE7-PE0	I/O	Inputs/outputs data (D15 to D8).
P70	Output	SDCLK pin output
P61	Output	SDCS# pin output
P62	Output	RAS# pin output
P63	Output	CAS# pin output
P64	Output	WE# pin output
P65	Output	CKE pin output
P66	Output	DQM0 pin output
P67	Output	DQM1 pin output

6. Software

6.1 Operation Overview

The initialization sequence, SDRAM mode register, auto refresh cycle, and SDRAM read/write timing are specified according to the SDRAM used.

This operation overview introduces an example of SDRAMC configuration with the Micron SDRAM (MT48LC8M16A2P-6A).

6.1.1 Configuring the SDRAM Initialization Sequence

After a reset, the SDRAM must be initialized before it can be used. Configure the initialization sequence considering AUTO REFRESH period (tRFC), number of times for initial auto refresh, PRECHARGE command period (tRP), and other settings according to the datasheet for the SDRAM.

Figure 6.1 shows the Timing of SDRAM (MT48LC8M16A2P-6A) Initialization and Table 6.1 lists the Example of the SDRAMC Initial Sequence Settings when Connecting to the SDRAM (MT48LC8M16A2P-6A).

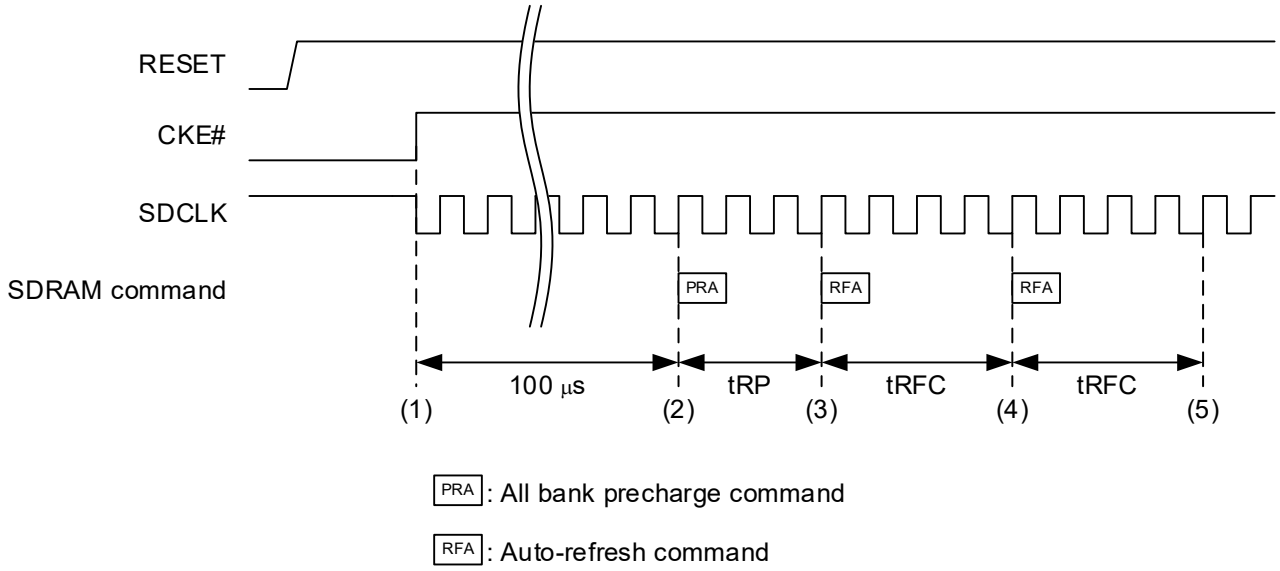


Figure 6.1 Timing of SDRAM (MT48LC8M16A2P-6A) Initialization

(1) After a reset, specifies pins associated with the SDRAM, sets the SYSCR0.EXBE bit to 1 (external bus is enabled) and starts outputting on the SDCLK pin. Then a high level signal is output from the CKE# pin. The CKE# pin is connected to GND through a resistor to drive the CKE# pin low after the power is turned on.

(2) Waits for 100 µs or longer after the clock is started to output. The device deselect command is output during the wait time. Then specifies the initialization timing to the SDIR register and sets the SDICR.INIRQ bit to 1, then the all bank precharge command is output.

(3) After the all bank precharge command is output, the auto-refresh command is output after the number of cycles specified by the SDIR.PRC[2:0] bits elapse. Set the value for the SDIR.PRC[2:0] bits to tRP or greater.

(4) After the auto-refresh command is output, the wait time in cycles specified by the SDIR.ARFI[2:0] bits is inserted. Set the value for SDIR.ARFI[2:0] bits to tRFC or greater.

When the number of initialization auto-refresh is set to 2 times or more by the SDIR.ARFC[3:0] bits, the auto-refresh command is output again.

(5) After the auto-refresh command is output for the number of times specified by the SDIR.ARFC[3:0] bits, the initial sequence is complete.

Table 6.1 Example of the SDRAMC Initial Sequence Settings when Connecting to the SDRAM (MT48LC8M16A2P-6A)

SDRAM Timing	Symbol	Setting Value	Setting in the SDRAMC with RX65N
Wait time until the PRECHARGE command is input after SDCLK is input	—	100 µs	After starting to output SDCLK, waits for 100 µs by the software, and starts the initial sequence.
PRECHARGE command period	tRP	18 ns (min.)	SDIR.PRC[2:0] = 000b: 3 cycles (approx. 50 ns when SDCLK is 60 MHz)
AUTO REFRESH period	tRFC	60 ns (min.)	SDIR.ARFI[3:0] = 0001b: 4 cycles (approx. 67 ns when SDCLK is 60 MHz)
Initial auto refresh	—	2 times	SDIR.ARFC[3:0] = 0010b: 2 times

6.1.2 Specifying the SDRAM Mode Register

After the SDRAM initialization, a mode has to be set for SDRAM. Set the SDRAM mode once after the initialization. When values are written to the SDRAM mode register (SDMOD), the mode register set command is output. For details on setting values to the SDRAM mode register, refer to the datasheet for the SDRAM.

Table 6.2 lists the SDRAM Mode Register of the SDRAM (MT48LC8M16A2P-6A) and Figure 6.2 shows the Timing Diagram of the Mode Register Set Command.

Table 6.2 SDRAM Mode Register of the SDRAM (MT48LC8M16A2P-6A)

Bit	Symbol	Description
b2 to b0	Burst Length	Selection of a burst length 000: 1 001: 2 010: 4 011: 8 111: Full page (only when b3 is 1) Do not set values other than above.
b3	Burst Type	Selection of a burst type 0: Sequential 1: Interleaved
b6 to b4	CAS Latency	Selection of a CAS latency 010: 2 011: 3 Do not set values other than above.
b8, b7	Operating Mode	00: Standard operation Do not set values other than above.
b9	Write Burst Mode	Selection of a write burst mode 0: Programmed burst length 1: Single location access
b11, b10	Reserved	Write 00b.

The burst length is 1 for the RX65N SDRAMC operation. If a value other than 1 is set as the burst length, the operation is not guaranteed.

The value set to the register in this application note is 230h (burst length: 1, CAS latency: 3 cycles).

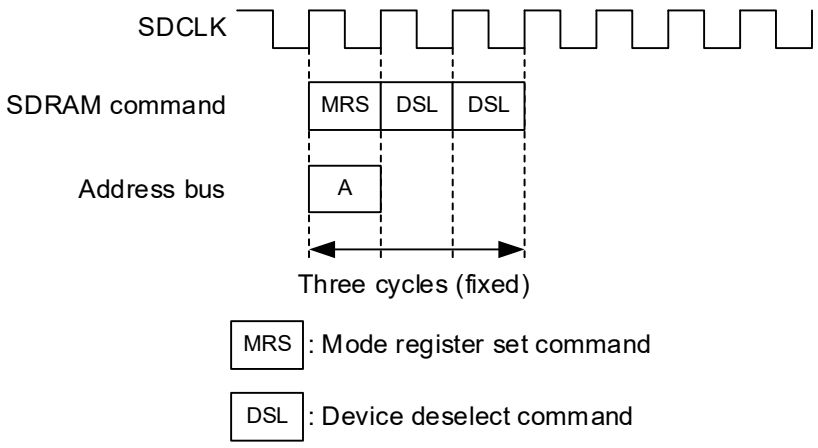


Figure 6.2 Timing Diagram of the Mode Register Set Command

6.1.3 Specifying the Auto-Refresh Cycle

To retain data in the SDRAM, a refresh must be performed for the number of rows during the refresh period (tREF). Auto-refresh must be performed considering the refresh period (tREF), number of rows, AUTO REFRESH period (tRFC), and other settings according to the datasheet for the SDRAM.

Table 6.3 lists the AUTO REFRESH Timing for the SDRAM (MT48LC8M16A2P-6A) and Figure 6.3 shows the AUTO REFRESH Operating Timing.

Table 6.3 AUTO REFRESH Timing for the SDRAM (MT48LC8M16A2P-6A)

SDRAM Timing	Symbol	Setting Value	Setting in the SDRAMC with RX65N
Refresh period	tREF	64 ms	Used for calculating AUTO REFRESH cycle
Number of rows	—	4096	Used for calculating AUTO REFRESH cycle
Auto refresh cycle	—	15.625 µs (tREF ÷ number of rows)	SDRFCR.RFC[11:0] = 03A7h: 936 cycles (approx. 15.6 µs when SDCLK is 60 MHz)
AUTO REFRESH period	tRFC	60 ns (min.)	SDRFCR.REFW[3:0] = 0011b: 4 cycles (approx. 67 ns when SDCLK is 60 MHz)

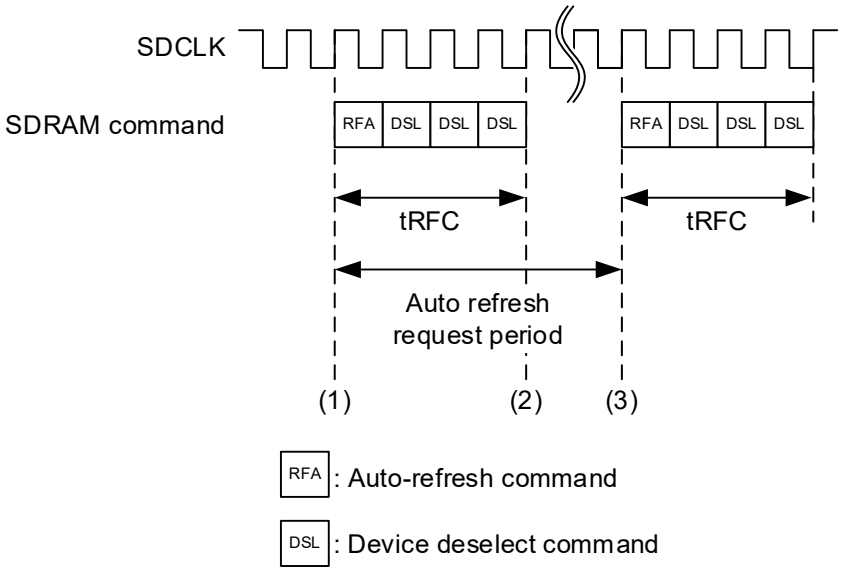


Figure 6.3 AUTO REFRESH Operating Timing

(1) When the SDRFEN.RFEN bit is set to 1 (auto-refresh operation is enabled), the auto-refresh command is output.

(2) After the auto-refresh command is output, the device deselect command is output until the number of cycles specified by the SDRFCR.REFW[3:0] bits elapse. Set a value for the SDRFCR.REFW[3:0] bits to tRFC or greater.

(3) The auto-refresh command is output every number of cycles specified by the SDRFCR.RFC[11:0] bits. Set a value for the SDRFCR.RFC[11:0] bits to be the auto refresh cycle (tREF ÷ number of rows) or less.

6.1.4 Specifying the SDRAM Read/Write Timing

The SDRAM read/write timing is specified considering the SDRAM settings of CAS latency (CL), WRITE recovery time (tWR), PRECHARGE command period (tRP), ACTIVE-to-PRECHARGE command (tRAS), and ACTIVE-to-READ or WRITE delay (tRCD).

Table 6.4 lists the Read/Write Timing when Connecting to the SDRAM (MT48LC8M16A2P-6A), Figure 6.4 shows the Read timing, and Figure 6.5 shows the Write timing.

Table 6.4 Read/Write Timing when Connecting to the SDRAM (MT48LC8M16A2P-6A)

SDRAM Timing	Symbol	Setting Value	Setting in the SDRAMC with RX65N
CAS latency (2)	—	2 or 3 (1)	SDTR.CL[2:0] = 011b: 3 cycles
WRITE recovery time	tWR	22.67 ns (min.)	SDTR.WR = 1: 2 cycles (approx. 33 ns when SDCLK is 60 MHz)
PRECHARGE command period	tRP	18 ns (min.)	SDTR.RP[2:0] = 001b: 2 cycle (approx. 33 ns when SDCLK is 60 MHz)
ACTIVE-to-PRECHARGE command period (2)	tRAS	42 ns (min.)	SDTR.RAS[2:0] = 010b: 3 cycles (approx. 50 ns when SDCLK is 60 MHz)
ACTIVE-to-READ or WRITE delay (2)	tRCD	18 ns (min.)	SDTR.RCD[1:0] = 01b: 2 cycle (approx. 33 ns when SDCLK is 60 MHz)

Notes: 1. Select '3' in the SDRAM mode setting.
2. Set a value for the ACTIVE-to-PRECHARGE command period to less than or equal to ACTIVE-to-READ or WRITE delay (SDTR.RCD[1:0] + SDTR.CL[2:0]).

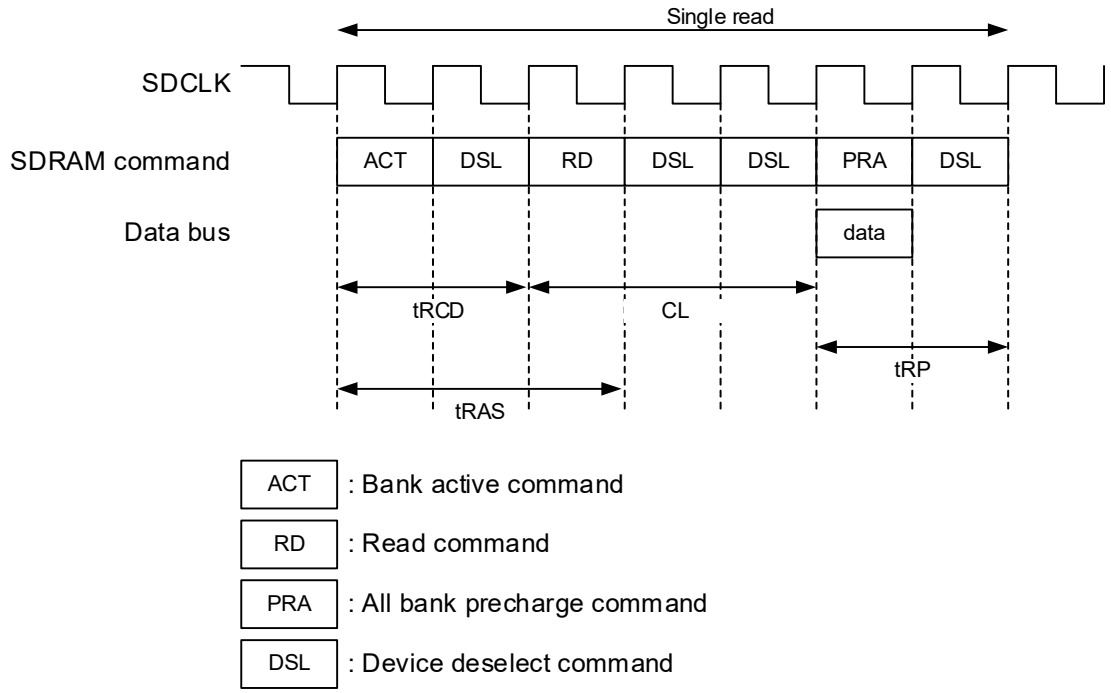


Figure 6.4 Read Timing when the SDRAMC with Setting in Table 6.4 is Used

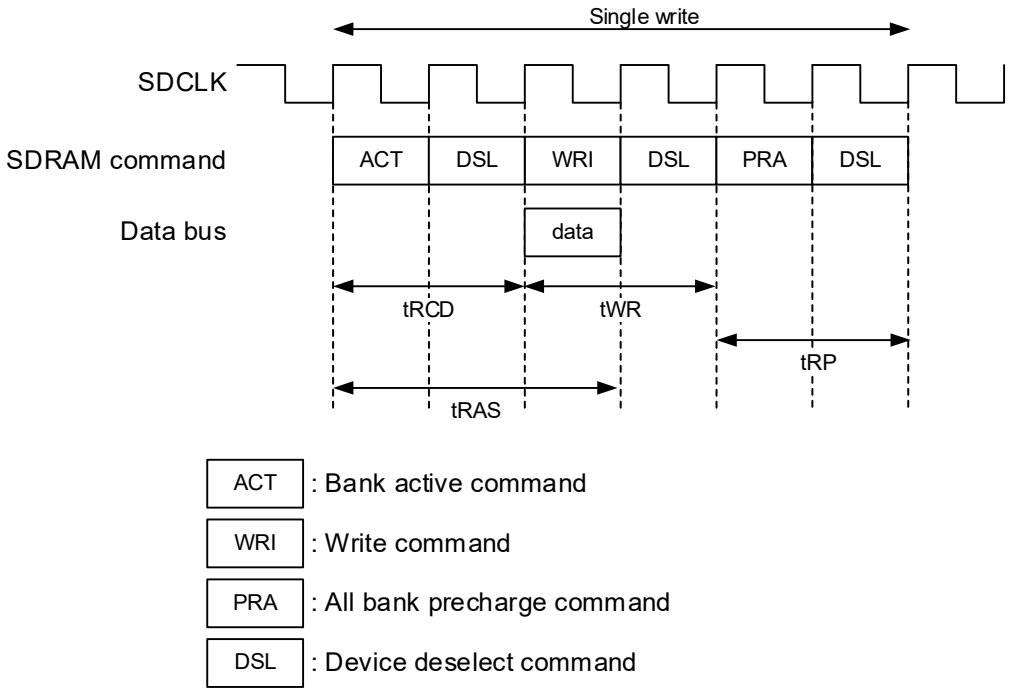


Figure 6.5 Write Timing when the SDRAMC with Setting in Table 6.4 is Used

6.2 File Composition

Table 6.5 lists the Files Used in the Sample Code. In the sample code of this Application Note, The each of projects for RX65N and RX72 that Smart Configurator is used/not used is prepared.

Files generated by the integrated development environment are not included in this table. Files containing unmodified source code generated by the Code Generator function of Smart Configurator have been omitted.

Table 6.5 Files Used in the Sample Code

File Name	Outline	Remarks
main.c	Main processing	Processing differs depending on whether Smart Configurator is used
r_init_stop_module.c	Stop processing for active peripheral functions after a reset	Only defined in projects that Smart Configurator is not used.
r_init_stop_module.h	Header file for r_init_stop_module.c	Only defined in projects that Smart Configurator is not used.
r_init_port_initialize.c	Nonexistent port initialization	Only defined in projects that Smart Configurator is not used.
r_init_port_initialize.h	Header file for r_init_non_existent_port.c	Only defined in projects that Smart Configurator is not used.
r_init_clock.c	Clock initialization	Only defined in projects that Smart Configurator is not used.
r_init_clock.h	Header file for r_init_clock.c	Only defined in projects that Smart Configurator is not used.
r_init_rom_cache.c	Initial ROM cache settings	Only defined in projects that Smart Configurator is not used and RX72M.
r_init_rom_cache.h	Header file of r_init_rom_cache.c	Only defined in projects that Smart Configurator is not used and RX72M.
r_cmt_wait.c	Wait processing using the CMT	Only defined in projects that Smart Configurator is not used.
r_cmt_wait.h	Header file for r_cmt_wait.c	Only defined in projects that Smart Configurator is not used.
r_sdram_api.c	SDRAM initialization.	Only defined in projects that Smart Configurator is not used.
r_sdram_api.h	Header file for r_sdram_api.c	Only defined in projects that Smart Configurator is not used.

6.3 Option-Setting Memory

Table 6.6 and Table 6.7 lists the Option-Setting Memory Configured in the Sample Code. When necessary, set a value suited to the user system.

Table 6.6 RX65N Option-Setting Memory Configured in the Sample Code

Symbol	Address	Setting Value	Contents
OFS0	FE7F 5D07h to FE7F 5D04h	FFFF FFFFh	IWDT is halted after a reset. WDT is stopped after a reset.
OFS1	FE7F 5D0Bh to FE7F 5D08h	FFFF FFFFh	Voltage monitor 0 reset is disabled after a reset. HOCO oscillation is disabled after a reset.
MDE	FE7F 5D03h to FE7F 5D00h	FFFF FFFFh	Little endian

Table 6.7 RX72M Option-Setting Memory Configured in the Sample Code

Symbol	Address	Setting Value	Contents
OFS0	FE7F 5D07h to FE7F 5D04h	FFFF FFFFh	IWDT is halted after a reset. WDT is stopped after a reset.
OFS1	FE7F 5D0Bh to FE7F 5D08h	FFFF FFFFh	Voltage monitor 0 reset is disabled after a reset. HOCO oscillation is disabled after a reset.
MDE	FE7F 5D03h to FE7F 5D00h	FFFF FFFFh	Little endian Linear mode

6.4 Constants

Table 6.8 and Table 6.9 lists the Constants Used in the Sample Code.

Table 6.8 Constants Used in the Sample Code (1/2)

Constant Name	Setting Value	Contents
LED0_REG_PODR(3)	PORT7.PODR.BIT.B3(1) PORT4.PODR.BIT.B2(2)	LED0 output data store bit
LED0_REG_PDR(3)	PORT7.PDR.BIT.B3(1) PORT4.PDR.BIT.B2(2)	LED0 I/O select bit
LED0_REG_PMR(3)	PORT7.PMR.BIT.B3(1) PORT4.PMR.BIT.B2(2)	LED0 pin mode control bit
LED1_REG_PODR(3)	PORTG.PODR.BIT.B7(1) PORTH.PODR.BIT.B0(2)	LED1 output data store bit
LED1_REG_PDR(3)	PORTG.PDR.BIT.B7(1) PORTH.PDR.BIT.B0(2)	LED1 I/O select bit
LED1_REG_PMR(3)	PORTG.PMR.BIT.B7(1) PORTH.PMR.BIT.B0(2)	LED1 pin mode control bit
LED_ON(3)	0	LED output data: Turned on
LED_OFF(3)	1	LED output data: Turned off
SDRAM_TOP	(void*)(0x08000000)	Start address of the SDRAM area
SDRAM_END	(void*)(0x09000000)	End address of the SDRAM area
R_WT_CMT_CLOCK(3)	60000000L	CMT count source frequency (PCLK)
R_WT_CMT_DIVIDE(3)	32L	Division ratio of the CMT count source
R_WT_BASE_US(3)	1000000L	Calculated value for the wait time for 1 µs
R_WT_BASE_MS(3)	1000L	Calculated value for the wait time for 1 ms
SDRAM_REG_MPC_PFAOE0(3)	0x7F(1) 0xFF(2)	MPC.PFAOE0 register set value
SDRAM_REG_MPC_PFAOE1(3)	0x00	MPC.PFAOE1 register set value
SDRAM_REG_MPC_PFBCR0(3)	0x11(1) 0x31(2)	MPC.PFBCR0 register set value
SDRAM_REG_MPC_PFBCR1(3)	0xD0	MPC.PFBCR1 register set value
SDRAM_REG_MPC_PFBCR2(3)	0x00	MPC.PFBCR2 register set value
SDRAM_REG_MPC_PFBCR3(3)	0x00(1) 0x40(2)	MPC.PFBCR3 register set value

Notes: 1. RX65N sample code setting value.
2. RX72M sample code setting value.
3. Only defined in projects that Smart Configurator is not used.

Table 6.9 Constants Used in the Sample Code (2/2)

Constant Name	Setting Value	Contents
SDRAM_REG_BSC_SDCCR(3)	0x00(1) 0x10(2)	BSC.SDCCR register set value
SDRAM_REG_BSC_SDCMOD(3)	0x00	BSC.SDCMOD register set value
SDRAM_REG_BSC_SDAMOD(3)	0x00	BSC.SDAMOD register set value
SDRAM_REG_BSC_SDRFCR(3)	0x33A8(1) 0x44E0(2)	BSC.SDRFCR register set value
SDRAM_REG_BSC_SDIR(3)	0x0021(1) 0x0022(2)	BSC.SDIR register set value
SDRAM_REG_BSC_SDADR(3)	0x01(1) 0x00(2)	BSC.SDADR register set value
SDRAM_REG_BSC_SDTR(3)	0x00021303(1) 0x00031303(2)	BSC.SDTR register set value
SDRAM_REG_BSC_SDMOD(3)	0x0230	BSC.SDMOD register set value
LED0_PORT_PIN(4)	GPIO_PORT_7_PIN_3(1) GPIO_PORT_4_PIN_2(2) GPIO_PORT_1_PIN_7(5) GPIO_PORT_7_PIN_1(7)	Enumerator of pin corresponding to LED0
LED1_PORT_PIN(4)	GPIO_PORT_G_PIN_7(1) GPIO_PORT_H_PIN_0(2) GPIO_PORT_F_PIN_5(5) GPIO_PORT_8_PIN_2(6) GPIO_PORT_H_PIN_6(7)	Enumerator of pin corresponding to LED1
LED2_PORT_PIN(4) (8)	GPIO_PORT_7_PIN_3	Enumerator of pin corresponding to LED2

Notes: 1. RX65N sample code setting value.
2. RX72M sample code setting value.
3. Only defined in projects that Smart Configurator is not used.
4. Only defined in projects that Smart Configurator is used.
5. RX671 (RSK-RX671) sample code setting value.
6. RX671 (EK-RX671) sample code setting value.
7. RX72N sample code setting value.
8. Only defined in projects for the EK-RX671.

6.5 Variables

Table 6.10 lists the Global Variables.

Table 6.10 Global Variables

Type	Variable Name	Description	Used by Function
static volatile uint8_t	s_g_cmt_event_flag	Compare match event occurrence flag	main cmt_event_cb

Note: Only defined in projects that Smart Configurator is used.

6.6 Functions

Table 6.11 lists the Functions. Functions consisting of unmodified source code generated by the Code Generator function of Smart Configurator have been omitted.

Table 6.11 Functions

Function Name	Outline
main	Main processing
port_init	Port initialization
R_INIT_StopModule(3)	Stop processing for active peripheral functions after a reset
R_INIT_Port_Initialize(3)	Nonexistent port initialization
R_INIT_Clock(3)	Clock initialization
R_INIT_ROM_Cache(2)(3)	Initial ROM cache settings
R_INIT_CMT_Wait(3)	Timer initialization for wait time
R_CMT_Wait(3)	Wait processing using the CMT
R_WAIT_US(3)	Wait processing using the CMT (unit: µs) (1)
R_WAIT_MS(3)	Wait processing using the CMT (unit: ms) (1)
R_SDRAM_Init(3)	SDRAMC initialization
sdram_verify_err	SDRAM verification error processing
cmt_oneshot_cb(4)	Compare match event callback processing.

Notes: 1. This function is a function-like macro.
2. Only defined in projects that RX72M sample code.
3. Only defined in projects that Smart Configurator is not used.
4. Only defined in projects that Smart Configurator is used.

6.7 Function Specifications

The following tables list the sample code function specifications.

main	
Outline	Main processing
Header	None
Declaration	void main(void)
Description	After initialization, initializes the SDRAM and performs read/write operation in the SDRAM.
Arguments	None
Return Value	None
Remarks	Processing differs depending on whether Smart Configurator is used.

port_init	
Outline	Port initialization
Header	None
Declaration	static void port_init(void)
Description	Initializes the ports.
Arguments	None
Return Value	None
Remarks	Processing differs depending on whether Smart Configurator is used.

R_INIT_StopModule	
Outline	Stop processing for active peripheral functions after a reset
Header	r_init_stop_module.h
Declaration	void R_INIT_StopModule(void)
Description	Configures the setting to enter the module-stop state.
Arguments	None
Return Value	None
Remarks	Transition to the module-stop state is not performed in the sample code. Refer to the RX65N Group, RX651 Group Initial Setting and RX72M Group Initial Setting application note for details on this function. This function is only defined in the projects that Smart Configurator are not used.

R_INIT_Port_Initialize	
Outline	Nonexistent port initialization
Header	r_init_port_initialize.h
Declaration	void R_INIT_Port_Initialize(void)
Description	Initializes port direction registers for ports that do not exist.
Arguments	None
Return Value	None
Remarks	RX65N sample code the number of pins in the sample code is set for the 176-pin package (PIN_SIZE=176). RX72M sample code the number of pins in the sample code is set for the 224-pin package (PIN_SIZE=224). After this function is called, when writing in byte units to the PDR registers or PODR registers which have nonexistent ports, set the corresponding bits for nonexistent ports as follows: set the I/O select bits in the PDR registers to 1 and set the output data store bits in the PODR registers to 0. Refer to the RX65N Group, RX651 Group Initial Setting and RX72M Group Initial Setting application note for details on this function. This function is only defined in the projects that Smart Configurator are not used.

R_INIT_Clock	
Outline	Clock initialization
Header	r_init_clock.h
Declaration	void R_INIT_Clock(void)
Description	Initializes the clock.
Arguments	None
Return Value	None
Remarks	The sample code selects processing which uses PLL as the system clock without using the sub-clock. Refer to the RX65N Group, RX651 Group Initial Setting and RX72M Group Initial Setting application note for details on this function. This function is only defined in the projects that Smart Configurator are not used.

R_INIT_ROM_Cache	
Outline	Initial ROM cache settings
Header	r_init_ROM_Cache.h
Declaration	void R_INIT_ROM_Cache(void)
Description	After specifying the non-cacheable areas, enables the ROM cache.
Arguments	None
Return Value	None
Remarks	In the sample code, this function only makes it possible for the ROM cache to operate. It is assumed that this function will be called while the ROM cache is in the disabled state after the system starts. To specify non-cacheable areas after the ROM cache has been enabled, first disable the ROM cache and then call this function. Refer to the RX65N Group, RX651 Group Initial Setting and RX72M Group Initial Setting application note for details on this function. This function is only defined in the projects that Smart Configurator are not used and RX72M.

R_SDRAM_Init	
Outline	SDRAMC initialization
Header	None
Declaration	void R_SDRAM_Init(void)
Description	Initializes SDRAMC used.
Arguments	None
Return Value	None
Remarks	This function is only defined in the projects that Smart Configurator are not used.

sdram_verify_err	
Outline	SDRAM verification error processing
Header	None
Declaration	static void sdram_verify_err(void)
Description	When the SDRAM verification error occurs, turns on LED1 and executes loop processing.
Arguments	None
Return Value	None
Remarks	Processing differs depending on whether Smart Configurator is used.

R_INIT_CMT_Wait	
Outline	Timer initialization for wait time
Header	r_cmt_wait.h
Declaration	void R_INIT_CMT_Wait (void)
Description	Initializes the timer (CMT0) for wait time.
Arguments	None
Return Value	None
Remarks	This function is only defined in the projects that Smart Configurator are not used.

R_CMT_Wait	
Outline	Wait processing using the CMT
Header	r_cmt_wait.h
Declaration	void R_CMT_Wait (uint16_t cnt)
Description	Waits for the time specified by the argument.
Arguments	uint16_t cnt: Wait time
Return Value	None
Remarks	This function is used in the R_CMT_WAIT_US(t_us) or R_CMT_WAIT_MS(t_ms) function. This function is only defined in the projects that Smart Configurator are not used.

R_CMT_WAIT_US	
Outline	Wait processing using the CMT (unit: µs)
Header	r_cmt_wait.h
Declaration	R_CMT_WAIT_US(t_us)
Description	Waits for the time (µs) specified by the argument.
Arguments	u_int16 t_us: Wait time (µs)
Return Value	None
Remarks	This function is a function-like macro. #define R_CMT_WAIT_US(t_us) R_CMT_Wait(t_us * (R_WT_CMT_CLOCK / R_WT_BASE_US) / R_WT_CMT_DIVIDE) This function is only defined in the projects that Smart Configurator are not used.

R_CMT_WAIT_MS	
Outline	Wait processing using the CMT (unit: ms)
Header	r_cmt_wait.h
Declaration	R_CMT_WAIT_MS(t_ms)
Description	Waits for the time (ms) specified by the argument.
Arguments	u_int16 t_ms: Wait time (ms)
Return Value	None
Remarks	This function is a function-like macro. #define R_CMT_WAIT_MS(t_ms) R_CMT_Wait(t_ms * (R_WT_CMT_CLOCK / R_WT_BASE_MS) / R_WT_CMT_DIVIDE) This function is only defined in the projects that Smart Configurator are not used.

cmt_event_cb	
Outline	Compare match event callback processing.
Header	None
Declaration	void cmt_oneshot_cb (void)
Description	Set 1 to s_g_cmt_event_flag when a compare match event occurs.
Arguments	CMT channel number
Return Value	None
Remarks	This function is only defined in the projects that Smart Configurator are not used.

6.8 Flowcharts

6.8.1 Sample Codes that Smart Configurator is not used

6.8.1.1 Main Processing

Figure 6.6 shows the Main processing of projects that Smart Configurator is not used.

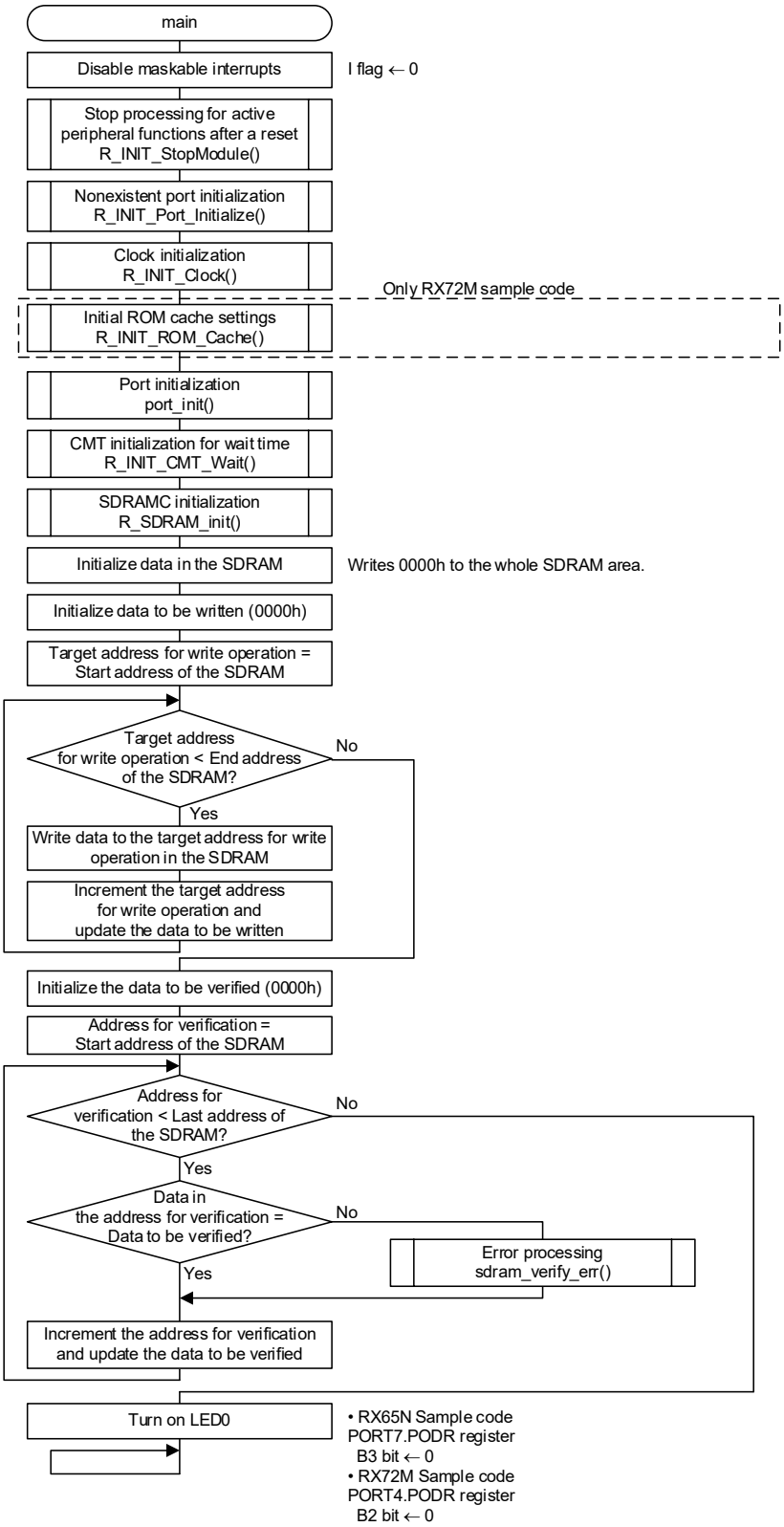


Figure 6.6 Main processing of projects that Smart Configurator is not used

6.8.1.2 SDRAMC Initialization

Figure 6.7 and Figure 6.8 show the RX65N SDRAMC Initialization. Figure 6.9 and Figure 6.10 show the RX72M SDRAMC Initialization.

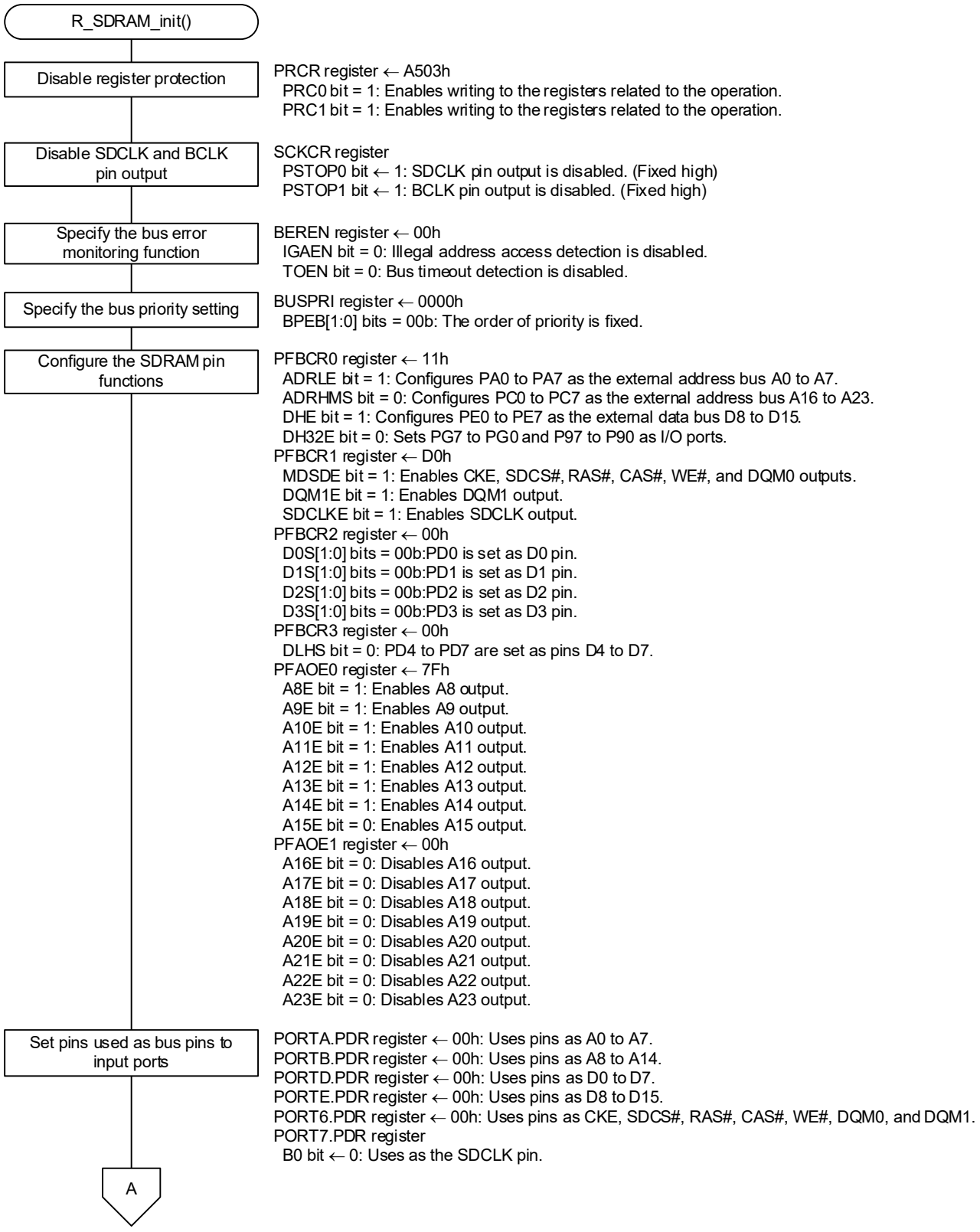


Figure 6.7 RX65N sample SDRAM Initialization (1/2)

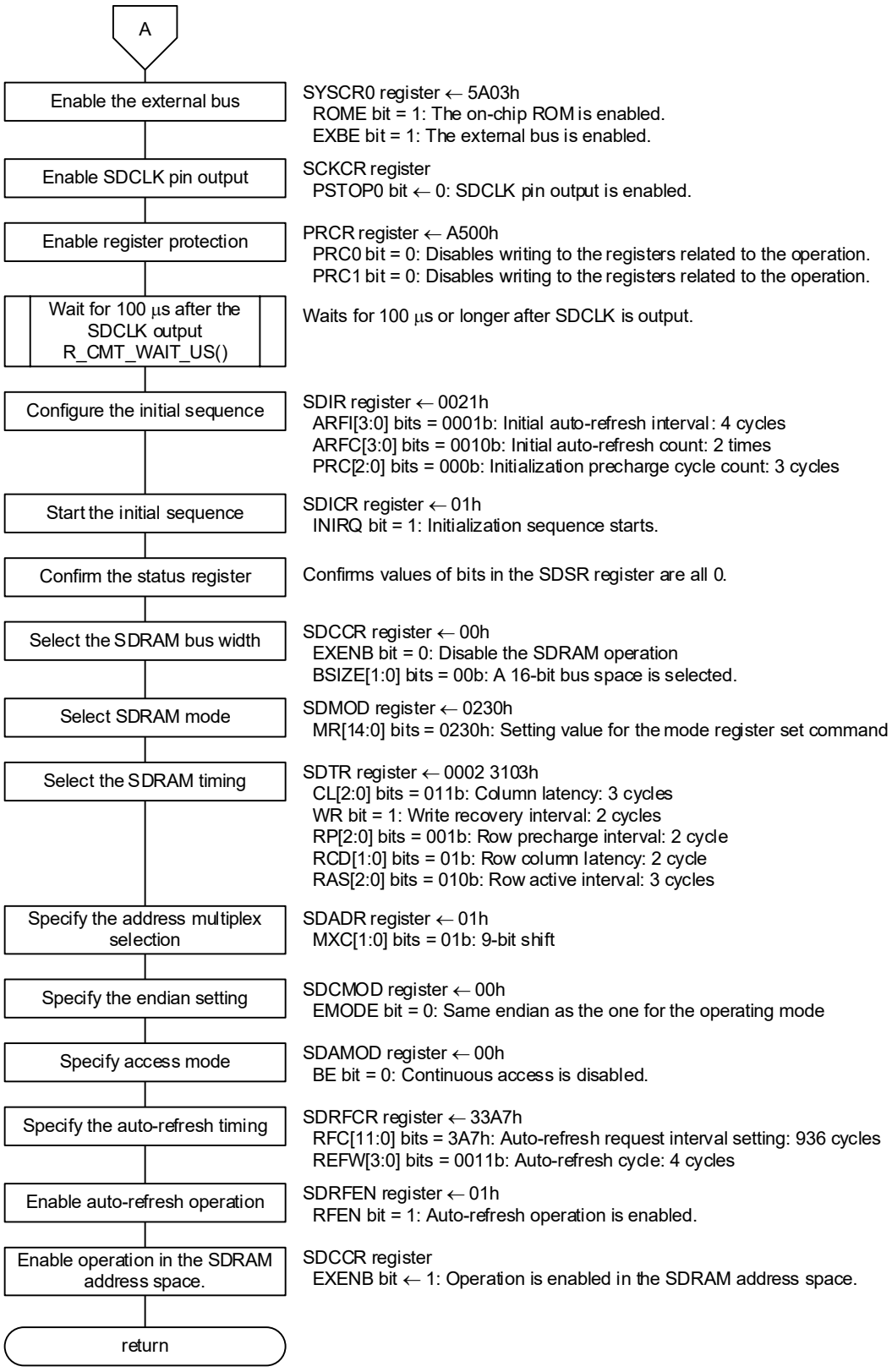


Figure 6.8 RX65N sample SDRAM Initialization (2/2)

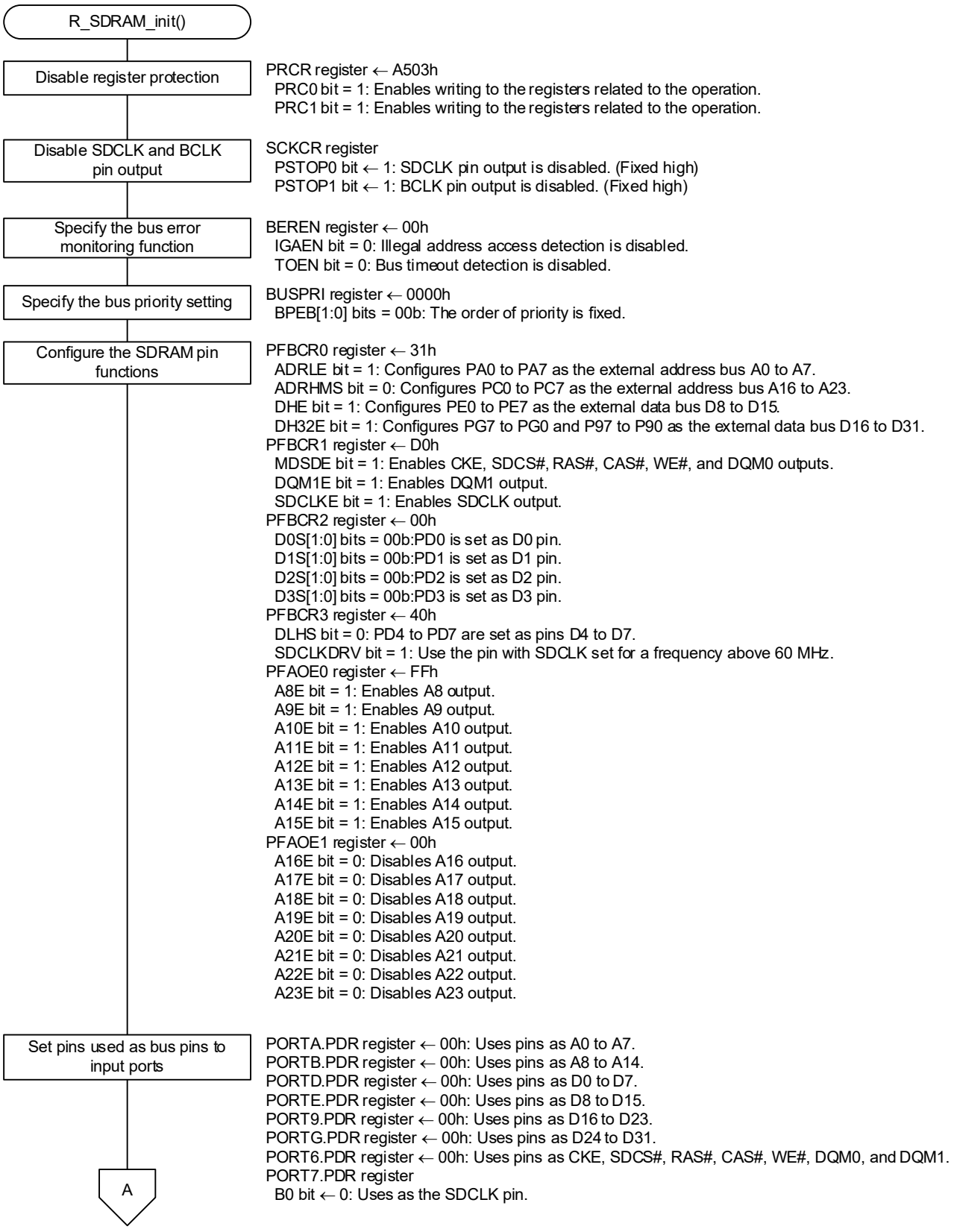


Figure 6.9 RX72M sample SDRAM Initialization (1/2)

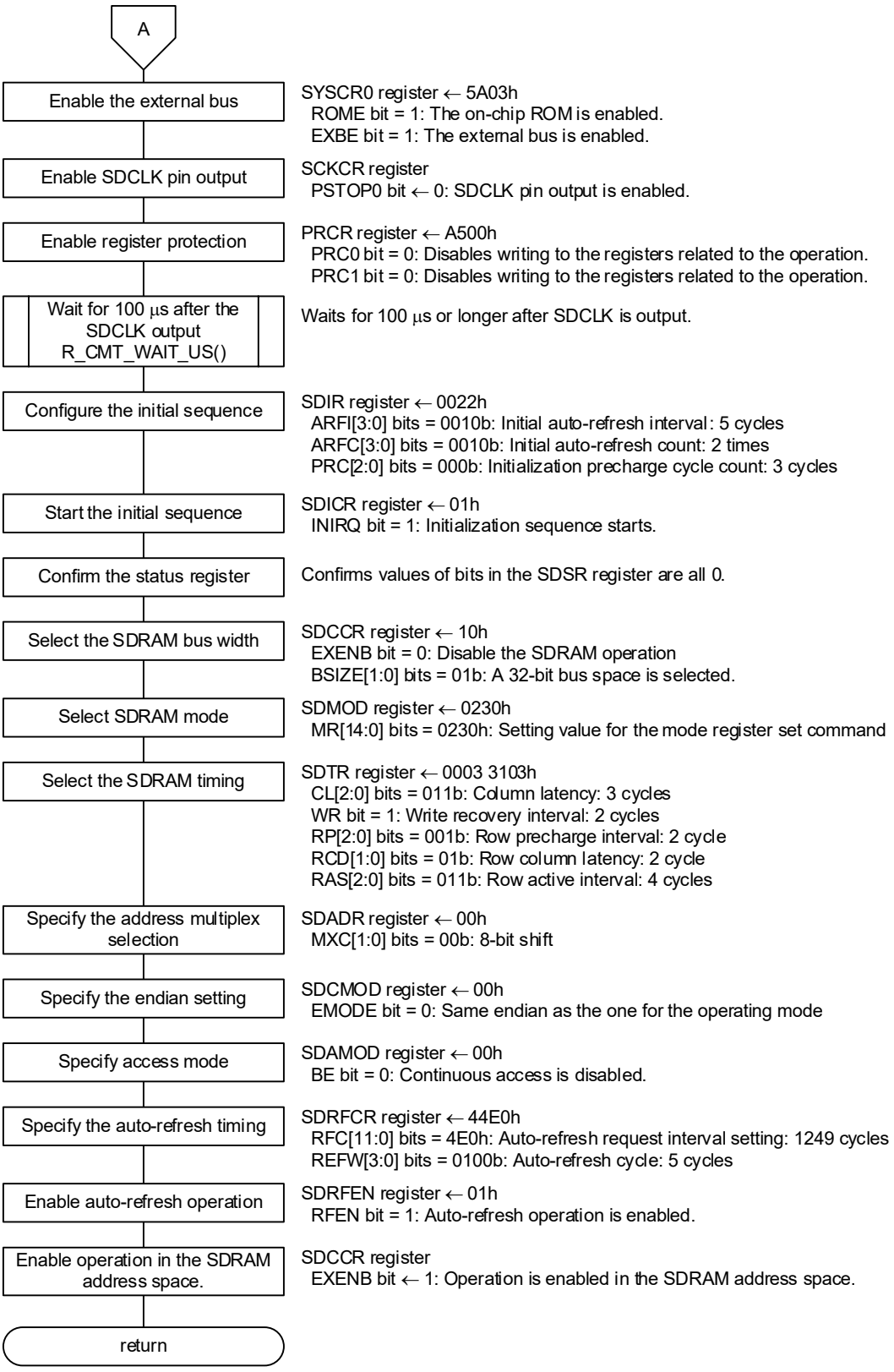


Figure 6.10 RX72M sample SDRAM Initialization (2/2)

6.8.1.3 Port Initialization

Figure 6.11 shows the Port Initialize of RX65N projects that Smart Configurator is not used. Figure 6.12 shows the Port Initialize of RX72M projects that Smart Configurator is not used.

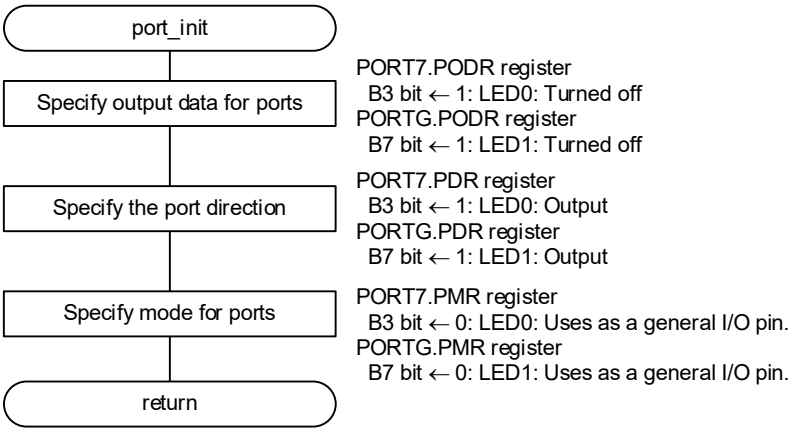


Figure 6.11 Port Initialize of RX65N projects that Smart Configurator is not used

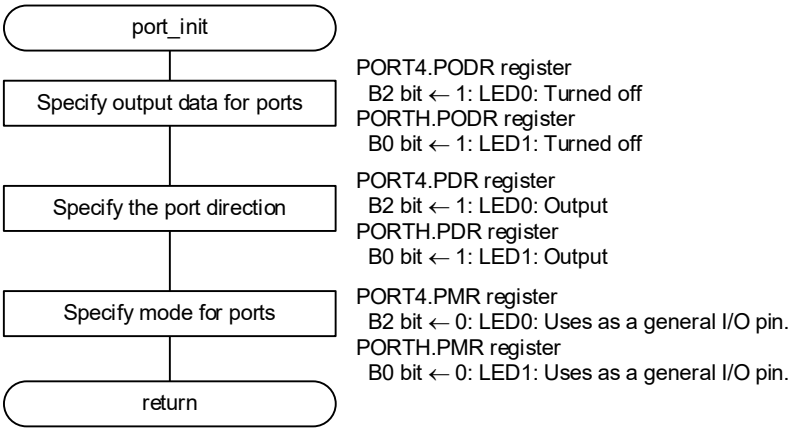


Figure 6.12 Port Initialize of RX72M projects that Smart Configurator is not used

6.8.1.4 Timer Initialization for Wait Time

Figure 6.13 shows the Timer Initialization for Wait Time.

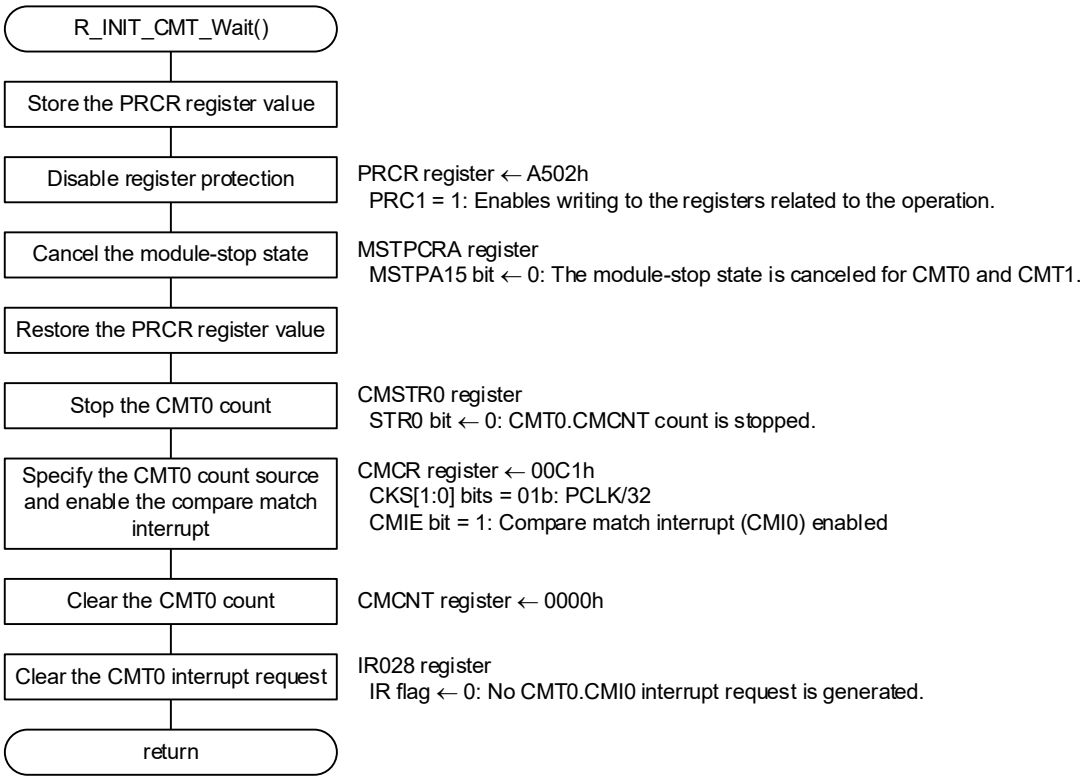


Figure 6.13 Timer Initialization for Wait Time

6.8.1.5 Wait Processing Using the CMT

Figure 6.14 shows the Wait Processing Using the CMT.

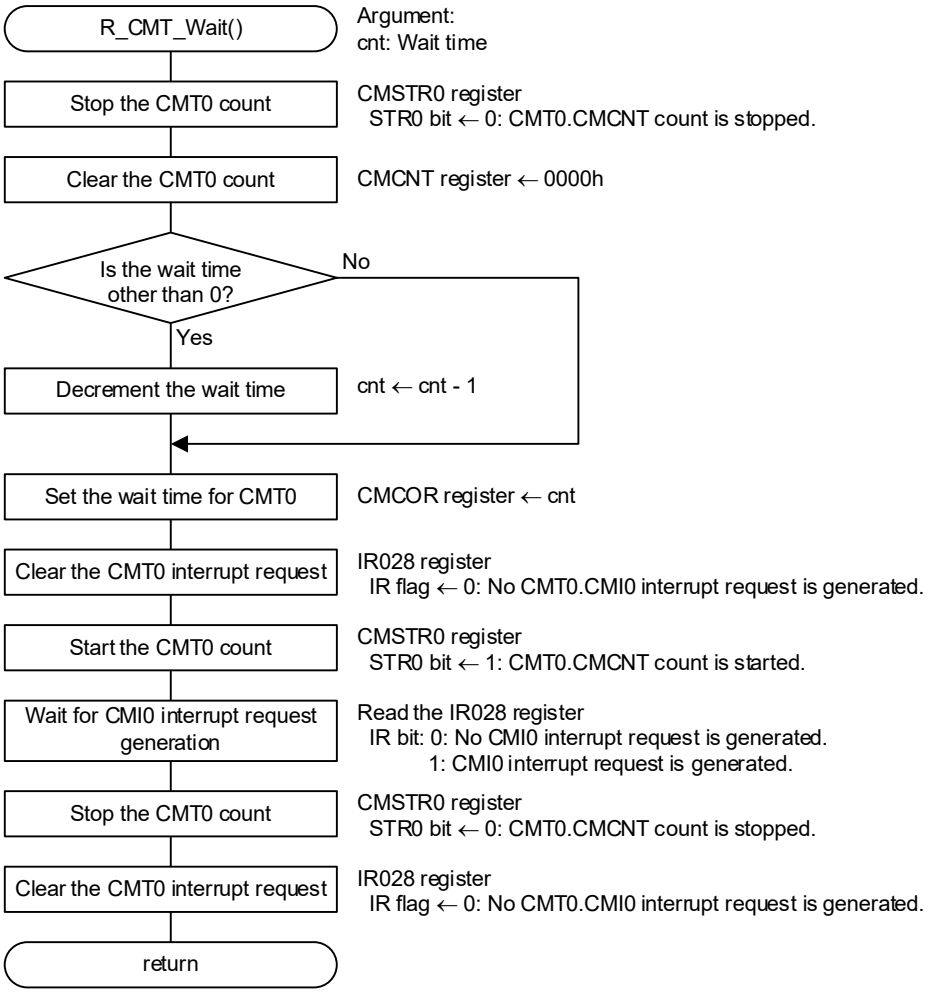


Figure 6.14 Wait Processing Using the CMT

6.8.1.6 SDRAM Verification Error Processing

Figure 6.15 shows the SDRAM Verification Error Processing.

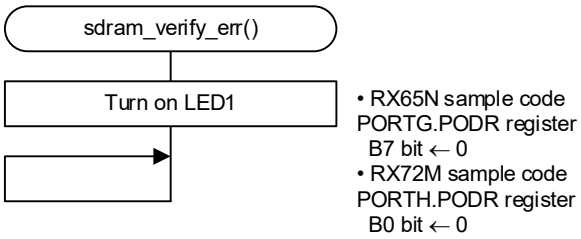


Figure 6.15 SDRAM Verification Error Processing of projects that Smart Configurator is not used

6.8.2 Sample Codes that Smart Configurator is used

6.8.2.1 Main Processing

Figure 6.16 shows Main processing of projects that Smart Configurator is used. Table 6.12 shows Data Settings in R_GPIO_PinWrite(). In the sample code generated by Smart Configurator, registers are rewritten by using a function of the GPIO FIT module.

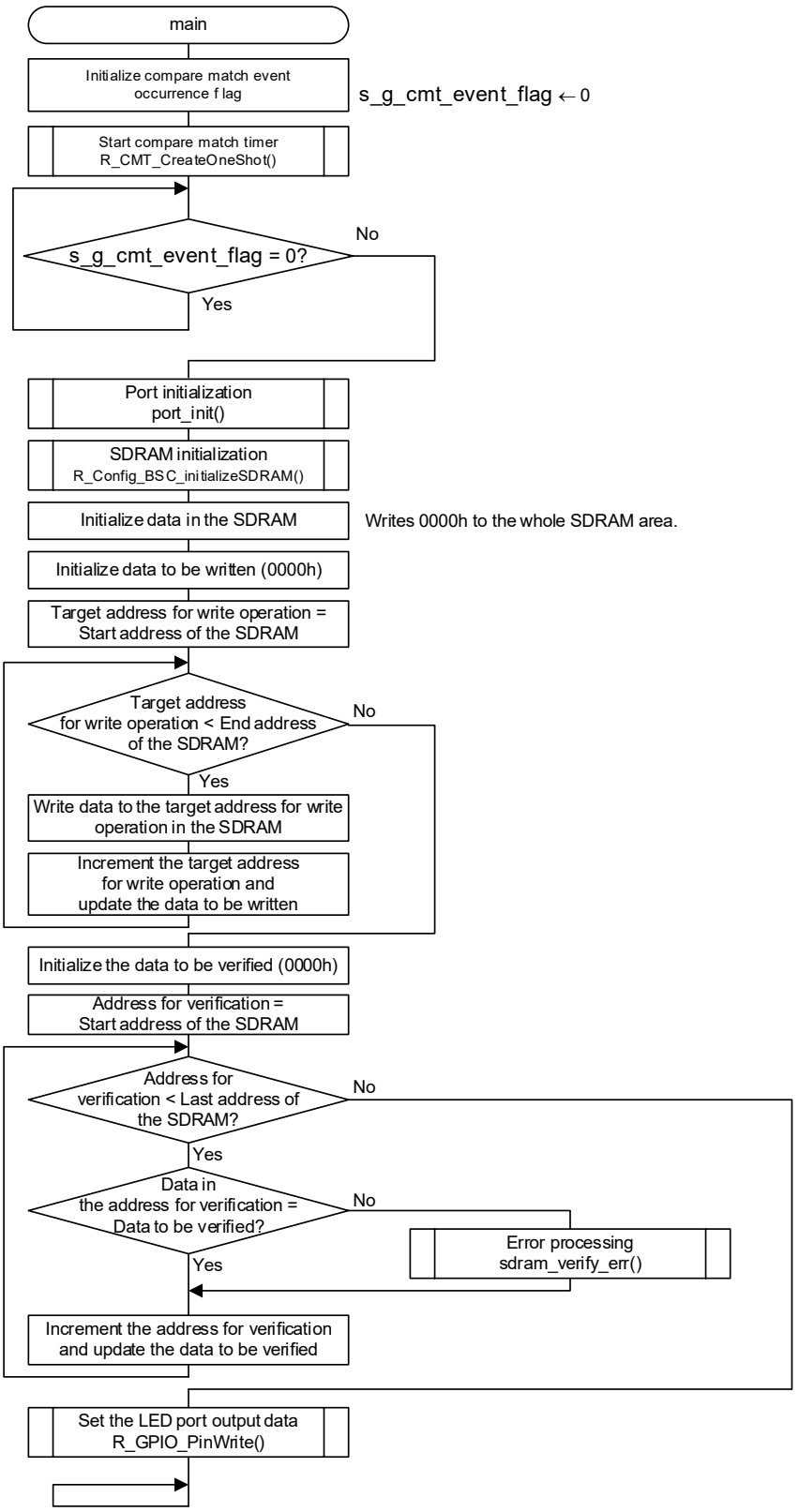


Figure 6.16 Main processing of projects that Smart Configurator is used

Table 6.12 Data Settings in R_GPIO_PinWrite()

Board Name	Bit to Be Set	Value to Be Set	Operation
RSK- RX65N	B3 bit of the PORT7.PODR register	0	Turns LED0 on.
		1	Turns LED0 off.
RSK- RX72M	B2 bit of the PORT4.PODR register	0	Turns LED0 on.
		1	Turns LED0 off.
RSK- RX671	B7 bit of the PORT1.PODR register	0	Turns LED0 on.
		1	Turns LED0 off.
RSK- RX72N	B1 bit of the PORT7.PODR register	0	Turns LED0 on.
		1	Turns LED0 off.
EK-RX671	B6 bit of the PORTH.PODR register	0	Turns LED1 on.
		1	Turns LED1 off.

6.8.2.2 Port Initialization

Figure 6.17 shows Port Initialization Processing in the Sample Code Generated by Smart Configurator (flow chart), Table 6.13 shows Data Settings in R_GPIO_PinDirection(). Table 6.14 shows Data Settings in R_GPIO_PinControl(). In the Sample Codes that Smart Configurator is used, the register is rewritten using the GPIO FIT function.

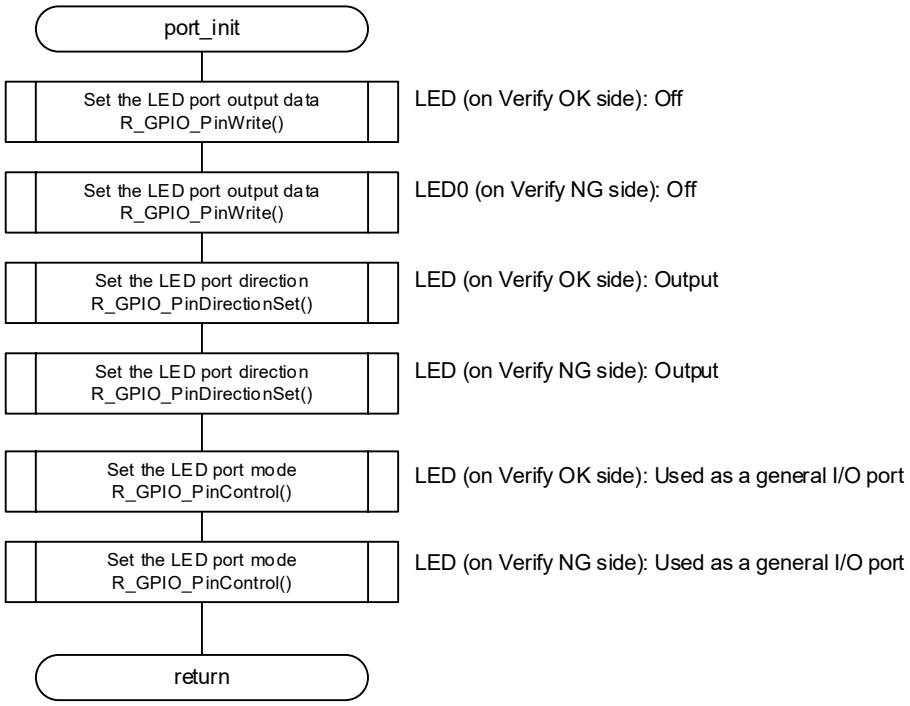


Figure 6.17 Port Initialization Processing in the Sample Code Generated by Smart Configurator

Table 6.13 Data Settings in R_GPIO_PinDirection()

Board Name	Bit to Be Set	Value to Be Set	Operation
RSK- RX65N	B3 bit of the PORT7.PDR register	1	Sets the output port.
	B7 bit of the PORTG.PDR register	1	Sets the output port.
RSK- RX72M	B2 bit of the PORT4.PDR register	1	Sets the output port.
	B0 bit of the PORTH.PDR register	1	Sets the output port.
RSK- RX671	B7 bit of the PORT1.PDR register	1	Sets the output port.
	B5 bit of the PORTF.PDR register	1	Sets the output port.
RSK- RX72N	B1 bit of the PORT7.PDR register	1	Sets the output port.
	B2 bit of the PORT8.PDR register	1	Sets the output port.
EK-RX671	B6 bit of the PORTH.PDR register	1	Sets the output port.
	B3 bit of the PORT7.PDR register	1	Sets the output port.

Table 6.14 Data Settings in R_GPIO_PinControl()

Board Name	Bit to Be Set	Value to Be Set	Operation
RSK- RX65N	B3 bit of the PORT7.PMR register	0	Sets the general I/O port.
	B7 bit of the PORTG.PMR register	0	Sets the general I/O port.
RSK- RX72M	B2 bit of the PORT4.PMR register	0	Sets the general I/O port.
	B0 bit of the PORTH.PMR register	0	Sets the general I/O port.
RSK- RX671	B7 bit of the PORT1.PMR register	0	Sets the general I/O port.
	B5 bit of the PORTF.PMR register	0	Sets the general I/O port.
RSK- RX72N	B1 bit of the PORT7.PMR register	0	Sets the general I/O port.
	B2 bit of the PORT8.PMR register	0	Sets the general I/O port.
EK-RX671	B6 bit of the PORTH.PMR register	0	Sets the general I/O port.
	B3 bit of the PORT7.PMR register	0	Sets the general I/O port.

6.8.2.3 SDRAM Verification Error Processing

Figure 6.18 shows the SDRAM Verification Error Processing of projects that Smart Configurator is used. In the Sample Codes that Smart Configurator is used, the register is rewritten using the GPIO FIT function.

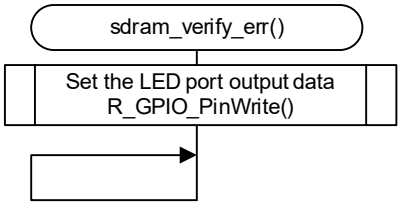


Figure 6.18 SDRAM Verification Error Processing of projects that Smart Configurator is used

6.8.2.4 Compare match event callback processing

Figure 6.19 shows the Compare match event callback processing.

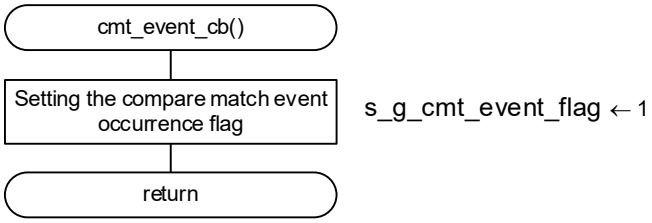


Figure 6.19 Compare match event callback processing

7. Concept of register setting in target device of SDRAM specification

This article explains the concept of register setting in the target device of the SDRAM specification using MT48LC4M32B2P-6A as an example.

For the settings of the pins used in SDRAMC, check the User's Manual: Hardware of the device used, hardware, " List of Output Enable Settings" in the I/O Ports, or "How to Set the External Bus Interface" in the Multi-Function Pin Controller (MPC).

7.1 BCLK (SDCLK) setting

The clock supplied to the CLK pin of SDRAM must be longer than the time specified in AC characteristics in the SDRAM data sheet.

In case of MT48LC4M32B2P-6A, CL = 3, it is necessary to supply the clock of 7ns or more (within 142MHz) per cycle. Also, since the maximum frequency that can be output from the SDCLK pin differs depending on the microcomputer used, check each User's Manual: Hardware.

As an example, if the RX72M and the clock source are 240MHz, the specifications can be met if the BCK[3:0] of the SCKCR register is set to a division of 3 or more and "ICLK≥BCLK".

7.2 SDC Control Register (SDCCR)

Operation enable of the SDRAM address space (EXENB bit) and SDRAM Bus Width Select (BSIZE bit) can be performed.

In the case of MT48LC4M32B2P-6A, since it can be accessed with a 32-bit width, setting BSIZE[1:0] to "01b" (A 32-bit bus space is selected) can meet the specifications. To enable of the SDRAM address space, set EXENB to "1" (Operation is enabled) after the SDRAMC related register settings are completed.

7.3 SDC Mode Register (SDCMOD)

You can select the Endian Mode select (EMODE bit).

In the case of MT48LC4M32B2P-6A, if the device operation mode is little endian, the endian of the SDRAM address space is the same as the endian of the operation mode. In this case, it can be operated by setting EMODE to “0” (Endian of SDRAM address space is the same as the endian of operating mode.).

7.4 SDRAM Access Mode Register (SDAMOD)

You can select the Continuous Access Enable (EB bit).

In the case of MT48LC4M32B2P-6A, continuous access is possible, so you can operate it by setting BE to “1” (Continuous access is enabled) after setting access using EXDMAC.

7.5 SDRAM Refresh Control Register (SDRFCR)

You can set the Auto-Refresh Request Interval Setting (RFC bit) and the Auto-Refresh Cycle/ Self-Refresh Clearing Cycle Count Setting (REFW bit).

The auto refresh request interval can be calculated from the following formula.

- Refresh cycle / number of row addresses

In the case of MT48LC4M32B2P-6A, the refresh cycle is 64 ms and the number of row addresses is “4096”, so the auto refresh request interval is 15.625 µs. The auto refresh request must be made within this time, the auto refresh request interval must be set by the RFC bit. As an example, if SDCLK is 60MHz, set RFC[11:0] to "3A7h" (936 cycles) or less to 15.6µs or less, which can meet the specifications.

Set the auto-refresh request interval setting to tRFC or more. In case of MT48LC4M32B2P-6A, tRFC is 60ns. As an example, if SDCLK is 60MHz, setting REFW[3:0] to "0011b" (4 cycles) or more makes it 67 ns or more, which satisfies the specifications.

7.6 SDRAM Initialization Register (SDIR)

You can set the Initialization Auto-Refresh Interval (ARFI bit), Initialization Auto-Refresh Count (ARFC bit), and Initialization Precharge Cycle Count (PRC bit).

Set the initialization auto-refresh interval to tRFC or more. In case of MT48LC4M32B2P-6A, tRFC is 60ns. As an example, if SDCLK is 60MHz, setting ARFI[3:0] to "0001b" (4 cycles) or more makes it 67 ns or more, which satisfies the specifications.

For initialization auto-refresh, execute the number of times specified in the data sheet. In the case of MT48LC4M32B2P-6A, it is twice or more, so by setting ARFC[3:0] to “0010b” (twice) or more. The specifications can be met more than twice.

Set the initialization precharge cycle count to tRP or more. In case of MT48LC4M32B2P-6A, tRP is 18ns. As an example, when SDCLK is 60MHz, PRC[3:0] is set to "0000b" (3 cycles) or more and it becomes 50ns or more, which can meet the specifications.

7.7 SDRAM Address Register (SDADR)

You can select the Address Multiplex Select (MXC bit). Match the shift amount of address multiplex with the width of column addressing. The MT48LC4M32B2P-6A has an 8-bit width, so the specifications can be met by setting MXC [1:0] to “00b” (8-bit shift).

7.8 SDRAM Timing Register (SDTR)

The following settings can be made in this register.

- SDRAMC Column Latency (CL bit)
- Write Recovery Interval(WR bit)
- Row Precharge Interval(RP bit)
- Row Column Latency(RCD bit)
- Row Active Interval(RAS bit)

Set the SDRAMC column latency setting so that it is the same as the column latency set in SDRAM. As an example, if the SDRAM column latency setting is "3", CL[2:0] can be set to "011b" (3 cycles) to meet the specifications.

The write recovery Interval setting should be set to tWR or more. For MT48LC4M32B2P-6A, when SDCLK is 60MHz, tWR is 23.67ns. As an example, if SDCLK is 60MHz, setting the WR bit to “1” (2 cycles) makes it 33ns, which satisfies the specifications.

Set the row precharge Interval to tRP or more. In case of MT48LC4M32B2P-6A, tRP is 18ns. As an example, if SDCLK is 60MHz, setting RP[2:0] to "001b" (2 cycles) or more will result in 33ns or more, which satisfies the specifications.

Set the row column latency setting to be tRCD or more. For MT48LC4M32B2P-6A, tRCD is 18ns. As an example, if SDCLK is 60MHz, setting RCD[1:0] to "01b" (2 cycles) or more will result in 33ns or more, which can meet the specifications.

Set the row active Interval to tRAS or more. In case of MT48LC4M32B2P-6A, tRAS is 42ns. As an example, if SDCLK is 60MHz, setting [2:0] to “010b” (3 cycles) or more will make it 50 ns or more, which can meet the specifications.

7.9 SDRAM Mode Register (SDMOD)

You can write the setting to the mode register of SDRAM. In case of MT48LC4M32B2P-6A, if you want to set column latency to "3" and burst length to "1", by setting "0x0230" in the SDRAM mode register, you can make the expected settings in SDRAM.

8. Porting Sample Codes that Smart Configurator is not used to Other RX Family

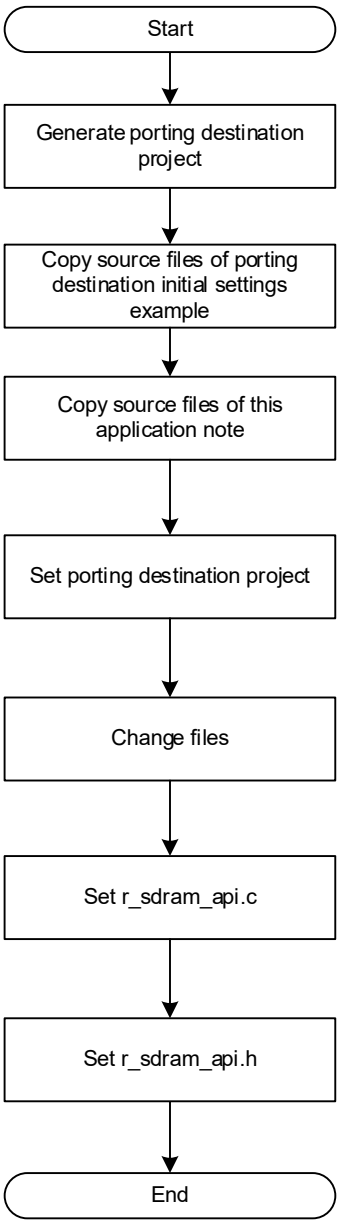
Sample codes included in this application note can be ported to other RX Family loaded with the exception vector table and software configurable interrupts. This section shows an example of porting sample code that RX65N and does not used the Smart Configurator to the RX72M (Renesas Starter Kit+ for RX72M).

8.1 Before Porting

Confirm the following specifications before porting sample codes. If there is a difference in the specifications, the method described in this section may not be used. After making sure of the specifications, use this application.

The SDRAMC specification of the porting source and porting destination
The CMT specification of the porting source and porting destination

8.2 Porting Procedure Flow

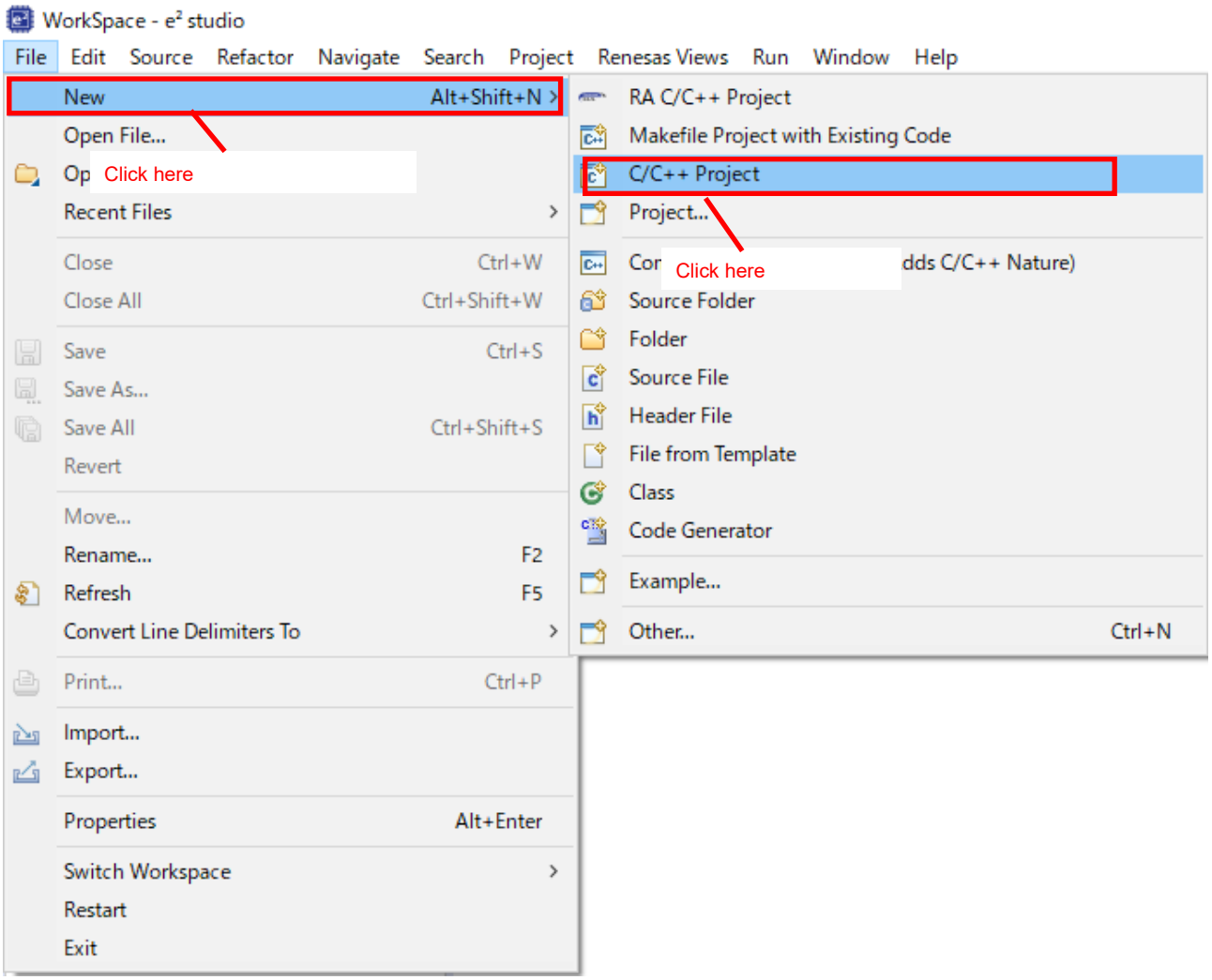


8.3 Porting Procedure

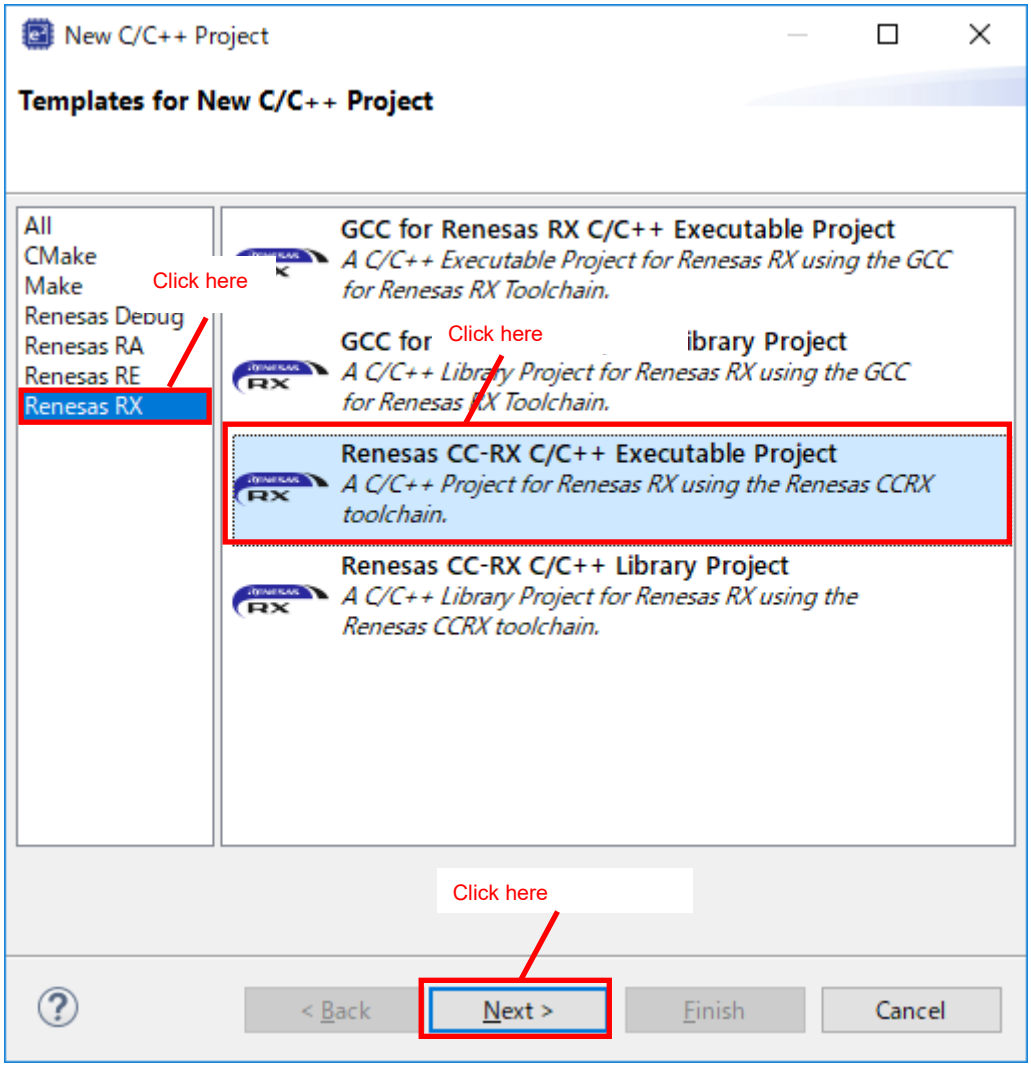
8.3.1 Generating a Porting Destination Project

Start e² studio and create a new project.

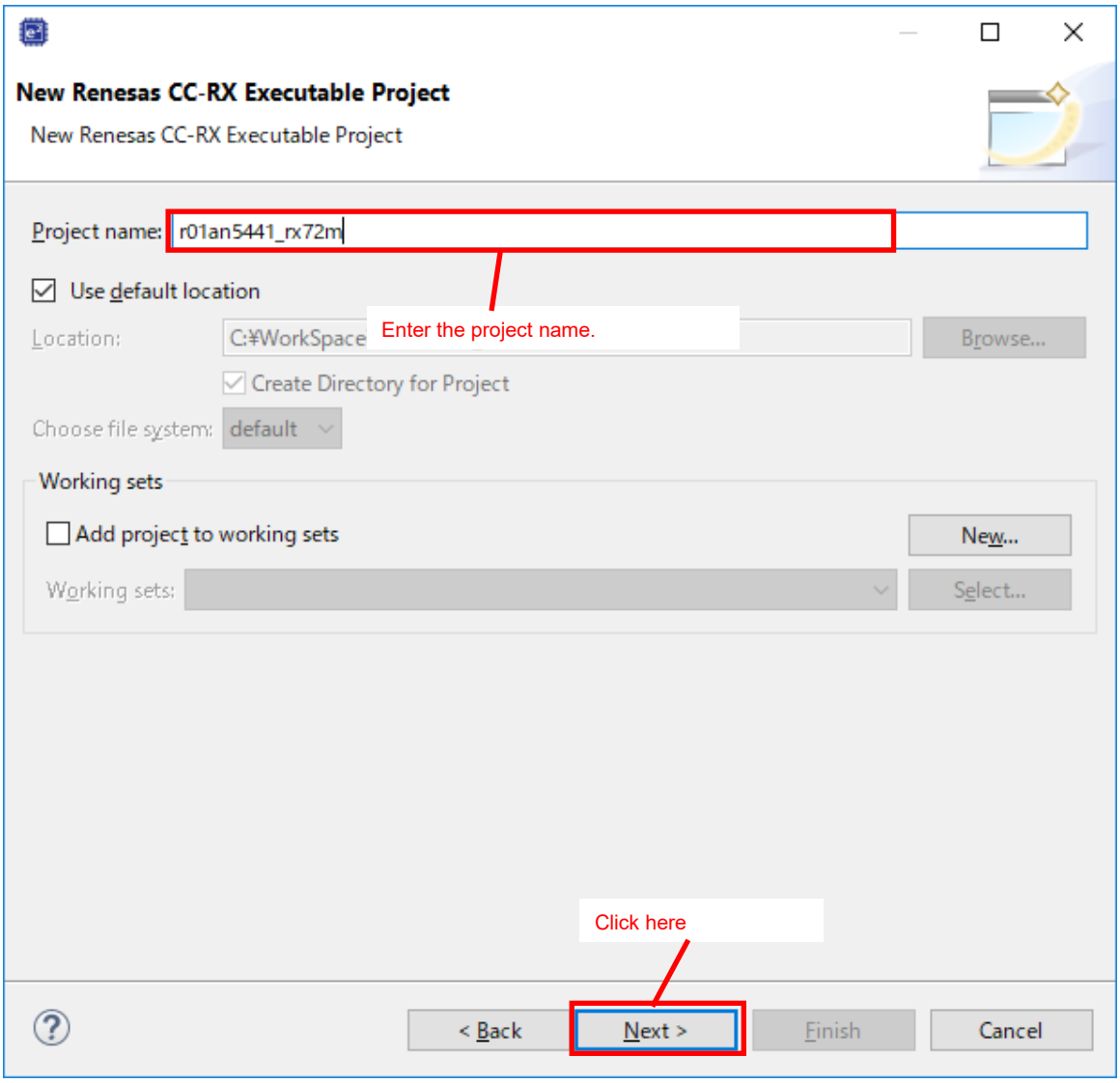
1) Generating a porting destination project
   1-1) Start e² studio and click [File].
   1-2) Click [C/C++ Project] of [New] to start the New C/C++ Project wizard.



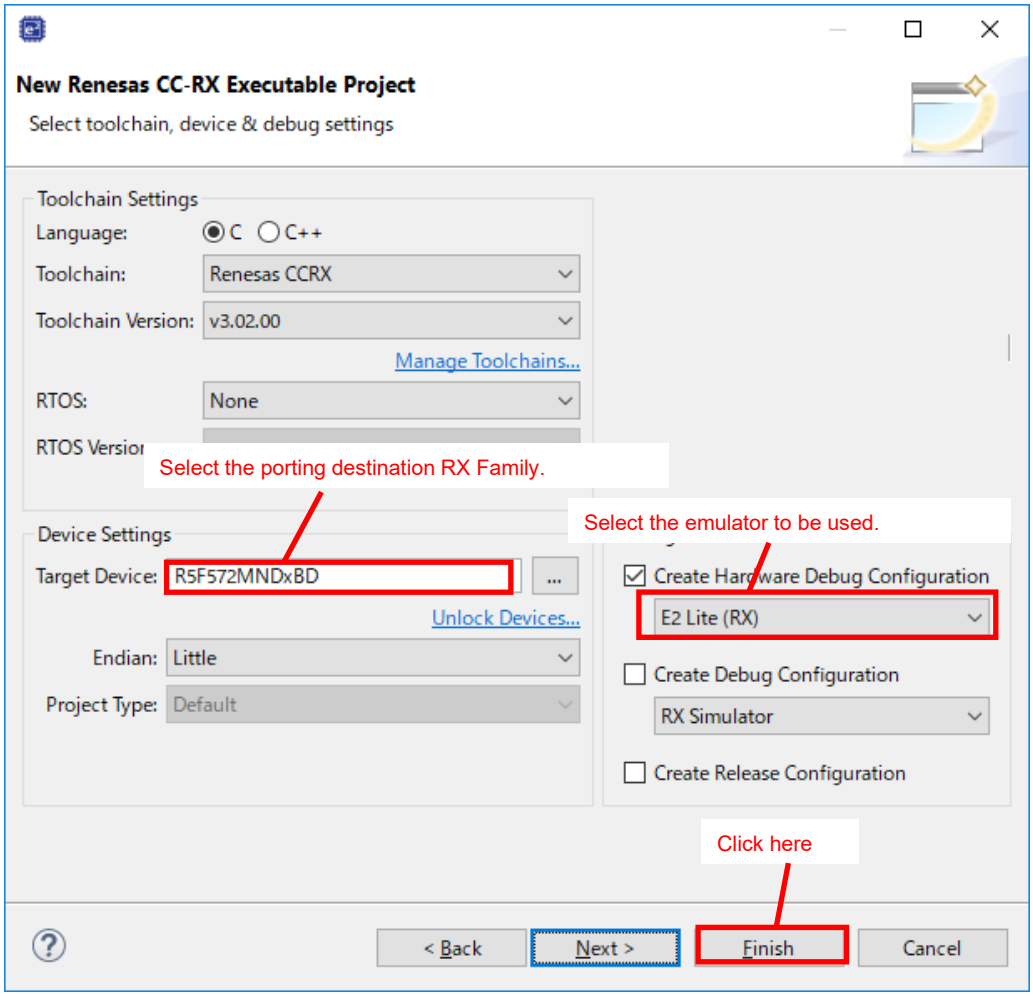
1-3) Click [Renesas RX].
1-4) Click [Renesas CC-RX C/C++ Executable Project].
1-5) Click [Next >].



1-6) Enter the project name.
1-7) Click [Next >].



1-8) Change [Target Device:] to [R5F572MNDxBD].
(When porting to another RX Family, change to the porting destination RX Family.)
1-9) Select the emulator to be used.
1-10) Click [Finish].



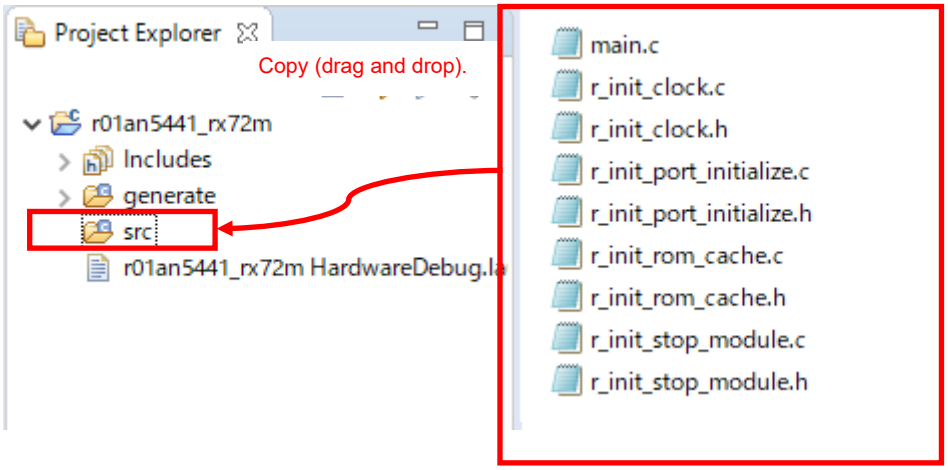
1-11) Delete [<Project name>.c] in the generated project.

8.3.2 Copying the Source Files of Porting Destination Initial Settings Example

Copy the source files of the initial settings example application note of the porting destination RX Family to the newly generated project.

1) Downloading the initial settings example application note
   1-1) From the Renesas Electronics website, download [RX72M Group Initial Settings (R01AN4530)]. (When porting to another RX Family, download the initial settings example application note corresponding to the porting destination RX Family.)
   1-2) Extract the downloaded zip file to the desired folder.

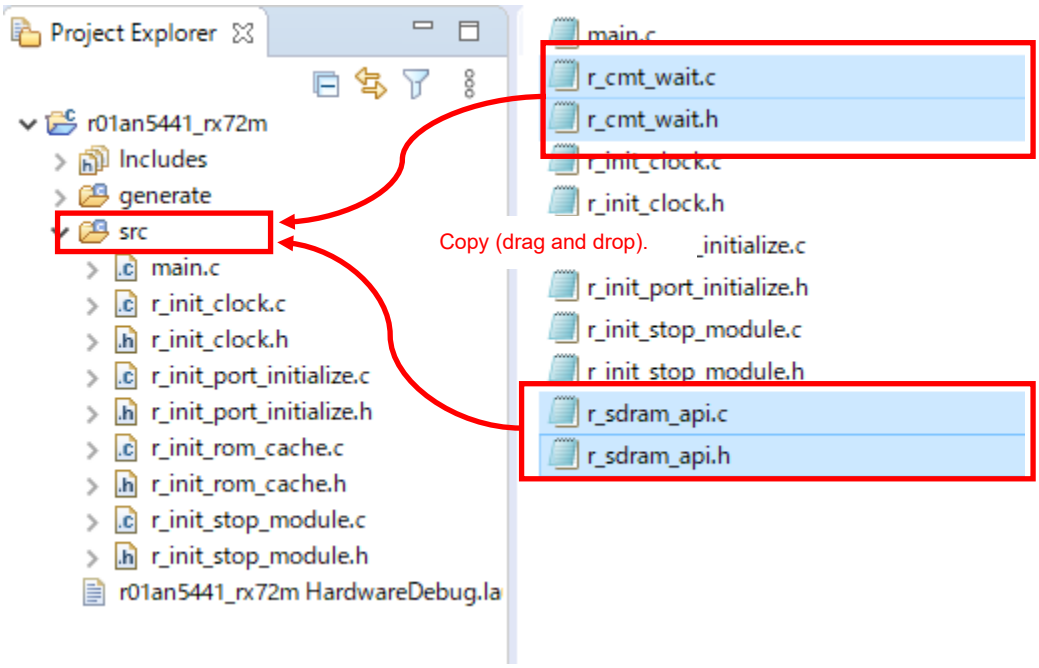
2) Copying the source files of the initial settings example application note to the project
   2-1) Use Explorer to open the extracted folder and copy all files from [r01an4530_rx72m] -> [r01an4530_src] to the generated project.



8.3.3 Copying the Source Files of the Application Note

Copy the source files of the application to the generated project.

1) Copy [r_cmt_wait.c], [r_cmt_wait.h], [r_sdram_api.c], and [r_sdram_api.h] from [r01an5441_rx65n_sdram] -> [r01an5441_src] of this application to the project.

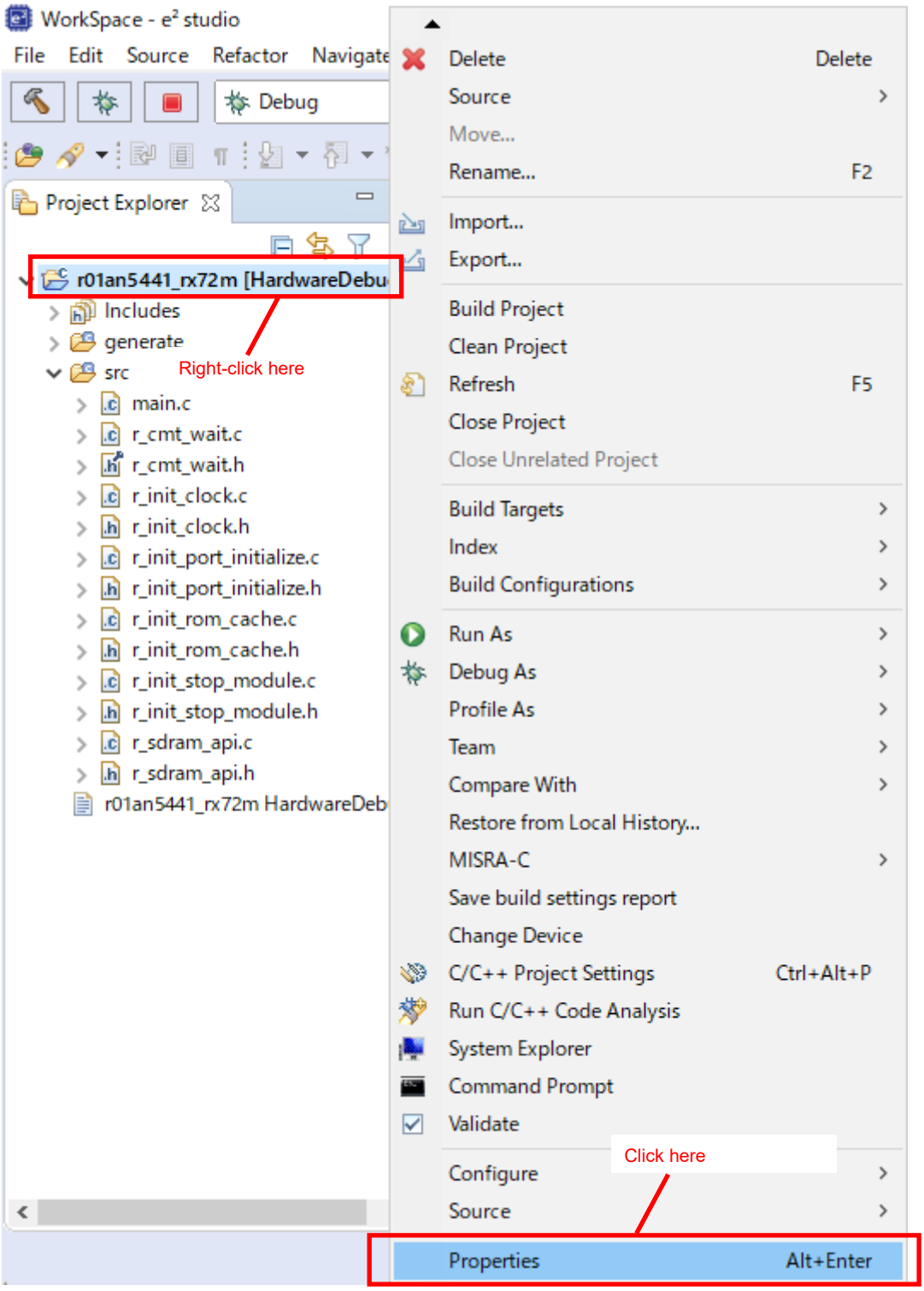


8.3.4 Setting Porting Destination Project

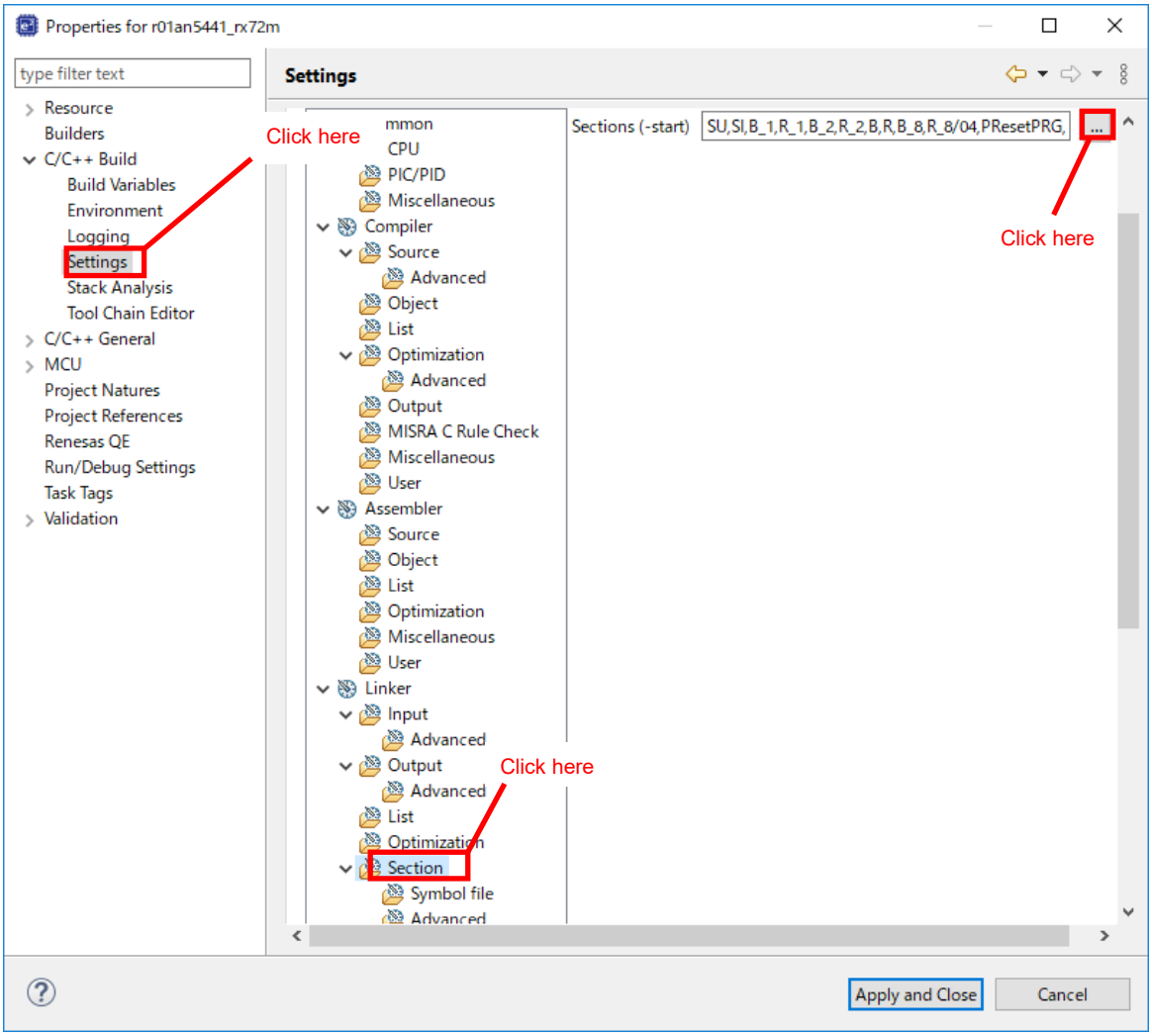
Change the build settings of the generated project.

1) Adding the final address section of the RAM

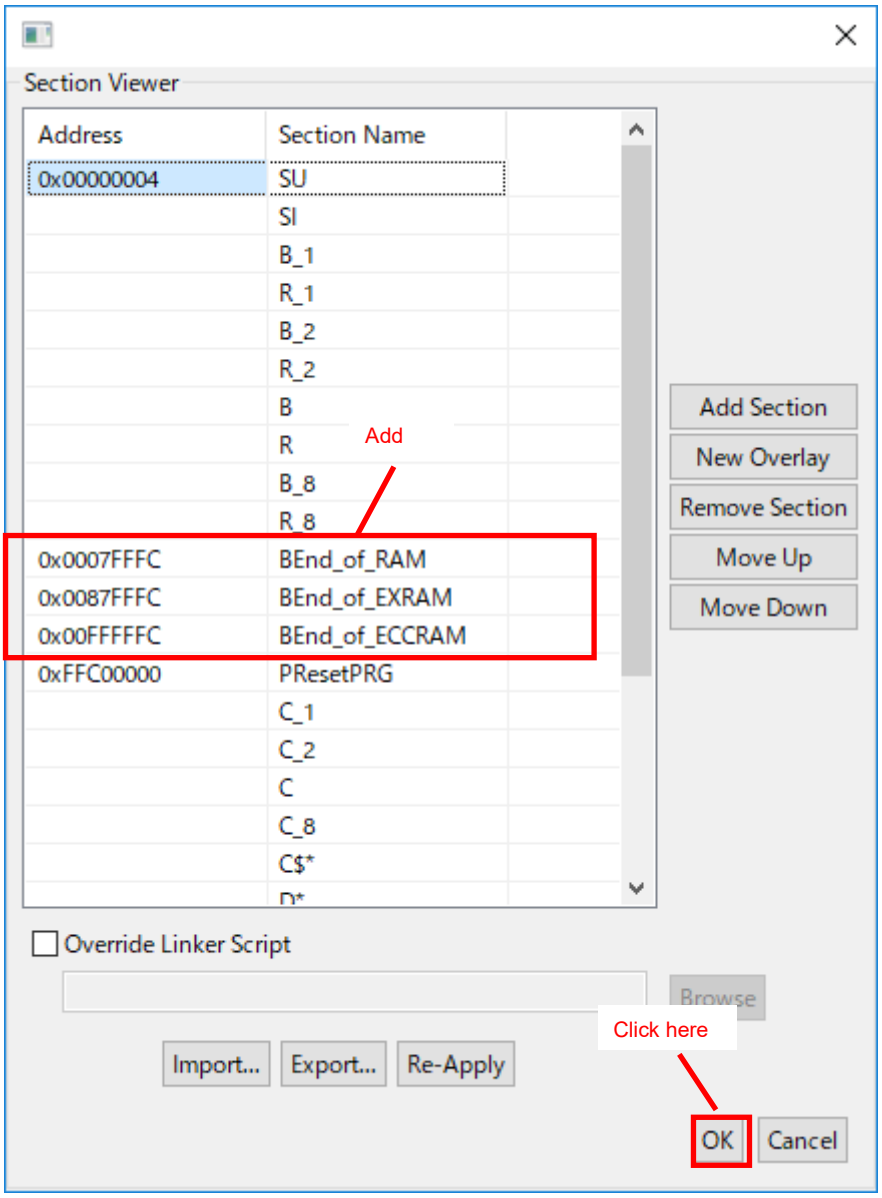
1-1) Right-click the generated project and click [Properties].



1-2) Click [C/C++ Build] -> [Settings].
1-3) Click [Tool Settings] -> [Linker] -> [Section].
1-4) Click […] at the right end of [Section].



1-5) Add the [End_of_RAM] section and [End_of_EXRAM] section and the [End_of_ECCRAM] section.
1-6) Click [OK].



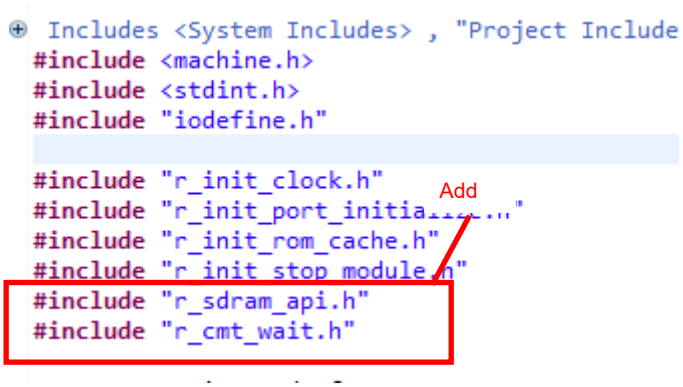
1-7) Click [Apply and Close].

8.3.5 Changing Files

Change each source file copied in order to run the sample code of the application.

1) Changing the path to the include file.
   1-1) The include file path of the source file differs depending on the initial settings example; review and change the include file path according to the porting destination project.

2) Changing [main.c]
   2-1) To [main.c], add the include path to [r_sdram_api.h] [r_cmt_wait.h].



   2-2) Add the following define under [#include “r_cmt_wait”] in [main.c] and change the define related to LED0 and LED1 to the port used by LED0 and LED1. This time, LED0 is set to [P42] and LED1 is set to [PH0] to match LED0 and LED1 of Renesas Starter Kit+ for RX72M.

```
/***********************************************************************
Macro definitions
***********************************************************************/
/* **** LEDs **** */
/* ==== LED0 (SDRAM verify OK) ==== */
#define LED0_REG_PODR   PORT4.PODR.BIT.B2   /* LED0 Output data store bit */
#define LED0_REG_PDR    PORT4.PDR.BIT.B2    /* LED0 I/O select bit */
#define LED0_REG_PMR    PORT4.PMR.BIT.B2    /* LED0 Pin mode control bit */
/* ==== LED1 (SDRAM verify error) ==== */
#define LED1_REG_PODR   PORTH.PODR.BIT.B0   /* LED1 Output data store bit */
#define LED1_REG_PDR    PORTH.PDR.BIT.B0    /* LED1 I/O select bit */
#define LED1_REG_PMR    PORTH.PMR.BIT.B0    /* LED1 Pin mode control bit */
/* ==== LEDs output data ==== */
#define LED_ON          (0)             /* LED on */
#define LED_OFF         (1)             /* LED off */
```

2-3) Define [SDRAM TOP] and [SDRAM END] following to [#define LED_OFF (1)] in [main.c], based on the connected SDRAM and the specified External address space. This time, [SDRAM_TOP] is set to [(void*)(0x08000000)] and [SDRAM_END] is set to [(void*)(0x09000000)] to match the specifications of this application note.

```
/* **** SDRAM address **** */
#define SDRAM_TOP   (void*)(0x08000000)          /* SDRAM top address 0x0800 0000 */
#define SDRAM_END   (void*)(0x09000000)          /* SDRAM end address 0x0900 0000 */
```

2-4) Add prototype function declarations for the [port_init] function and [sdram_verify_err] function in [main.c].

```
 Exported global variables and functions (to be accessed b
void main(void);

static void port_init(void);
static void sdram_verify_err(void);
```

Add

2-5) Define the [*sp_sdram_adr], [s_sdram_data], and [s_sdram_data] variables in the [main] function of [main.c] with a type that matches the data bus width of the connected SDRAM. This time, it is defined in [uint32_t] to match the specifications of the SDRAM installed in Renesas Starter Kit+ for RX72M.

```
 * Function Name: main
void main (void)
{
    volatile static uint32_t *sp_sdram_adr;      /* address pointer(SDRAM) */
    volatile static uint32_t s_sdram_data;       /* sdram write data (SDRAM) */
    volatile static uint32_t s_sdram_cmp_data;   /* compare dat      ) */

    /* ---- Disable maskable interrupts ---- */
    clrpsw_i();
```

Add

2-6) In the [main] function of [main.c], call the [port_init] function, [R_INIT_CMT_Wait] function, and [R_SDRAM_Init] function before while statement. Also, add the processing to change the [External bus clock (BCLK) select bit] according to the specifications of the SDRAM connected. This time, it does not need to be added because it matches the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M.

```
    /* ---- Initialization of the ROM Cache ---- */
    R_INIT_ROM_Cache();

    /* ---- BCLK bit select division 3 ---- */
    SYSTEM.PRCR.WORD = 0xA501;
    SYSTEM.SCKCR.BIT.BCK = 0x09;
    SYSTEM.PRCR.WORD = 0xA500;

    /* ---- Initialize ports ---- */
    port_init();

    /* ---- Initialize WAIT timer(CMT0) ---- */
    R_INIT_CMT_Wait();

    /* ---- Initialize SDRAMC ---- */
    R_SDRAM_Init();
```

Add if needed

Add

2-7) Add the following SDRAM access processing and LED0 lighting processing under the [main] function and [R_SDRAM_Init] function of [main.c]. This process is the same as that used in the application note.

```
/*********************************
 *  Memory access (SDRAM)
 *********************************/

/* ---- Data initialize for SDRAM  ---- */
for(sp_sdram_adr = SDRAM_TOP; sp_sdram_adr < SDRAM_END; sp_sdram_adr++)
{
    *sp_sdram_adr = 0x00000000;         /* Initialize data */
}

/* ---- Write data for SDRAM  ---- */
s_sdram_data = 0x00000000;            /* SDRAM write data initialize */
for(sp_sdram_adr = SDRAM_TOP; sp_sdram_adr < SDRAM_END; sp_sdram_adr++)
{
    *sp_sdram_adr = s_sdram_data++;    /* Write increment data */
}

/* ---- Verify SDRAM data  ---- */
s_sdram_cmp_data = 0x00000000;            /* SDRAM verify data initialize */
for(sp_sdram_adr = SDRAM_TOP; sp_sdram_adr<SDRAM_END; sp_sdram_adr++)
{
    /* ---- Verify error check  ---- */
    if(s_sdram_cmp_data != (*sp_sdram_adr))
    {
        /* ---- Verify error  ---- */
        sdram_verify_err();
    }
    s_sdram_cmp_data++;              /* Verify data increment */

}

/* LED0 ON (SDRAM verify OK) */
LED0_REG_PODR = LED_ON;
```

2-8) Define the following [port_init] function and [sdram_verify_err] function entities in [main.c].

```
static void port_init(void)
{

    /* ---- Initialize LEDs ---- */

    /* Set port output data - LEDs OFF */
    LED0_REG_PODR = LED_OFF;
    LED1_REG_PODR = LED_OFF;

    /* Set port direction - Output */
    LED0_REG_PDR = 1;
    LED1_REG_PDR = 1;

    /* Set port mode - Use pin as general I/O port */
    LED0_REG_PMR = 0;
    LED1_REG_PMR = 0;
}

static void sdram_verify_err(void)
{

    /* LED1 ON (SDRAM verify error) */
    LED1_REG_PODR = LED_ON;

    while (1)
    {
    }

}
```

8.3.6 Setting r_sdram_api.c

Change the port direction of the pin used for SDRAM connection to input. This time, to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M, PORT9 and PORTG corresponding to D16 to D31 are set as input ports.

```
/* ---- Set up pins to be used as input port pins. ---- */
PORTA.PDR.BYTE = 0x00;              /* PORTA(A0-A7)  input */
PORTB.PDR.BYTE = 0x00;              /* PORTB(A8-A14) input */
PORTD.PDR.BYTE = 0x00;              /* PORTD(D0-D7)  input */
PORTE.PDR.BYTE = 0x00;              /* PORTE(D8-D15) input */
PORT9.PDR.BYTE = 0x00;              /* PORT9(D16-D23) input */
PORTG.PDR.BYTE = 0x00;              /* PORTG(D24-D31) input */
PORT6.PDR.BYTE = 0x00;              /* SDCS#, RAS#,CAS#, WE#, (
PORT7.PDR.BIT.B0 = 0;               /* SDCLK input */
```

Add

8.3.7 Setting r_sdram_api.h

Change [r_sdram_api.h] in accordance with the porting destination environment.

The setting value when porting to RX72M (Renesas Starter Kit+ for RX72M) is shown below. When porting to another RX Family, change to the setting value corresponding to the porting destination environment. Also, see 7.Concept of register setting in target device of SDRAM specification.

1) Setting Address Output Enable Register

Change the settings of [SDRAM_REG_MPC_PFAOE0] (Address Output Enable Register 0) and [SDRAM_REG_MPC_PFAOE1] (Address Output Enable Register 1) for the specifications of the connected SDRAM. This time, [SDRAM_REG_MPC_PFAOE0] is changed to “0xFF” to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M. No need to change [SDRAM_REG_MPC_PFAOE1].

```
#define SDRAM_REG_MPC_PFAOE0  (0xFF)        /* Addr  tput Enable Register 0
#define SDRAM_REG_MPC_PFAOE1  (0x00)        /* Addr  tput Enable Register 1
```

Set

2) Setting External Bus Control Register

Change the settings of [SDRAM_REG_MPC_PFBCR0] (External Bus Control Register 0), [SDRAM_REG_MPC_PFBCR1] (External Bus Control Register 1), [SDRAM_REG_MPC_PFBCR2] (External Bus Control Register 2) and [SDRAM_REG_MPC_PFBCR3] (External Bus Control Register 3) for the specifications of the connected SDRAM. This time, [SDRAM_REG_MPC_PFBCR0] is changed to “0x31” and [SDRAM_REG_MPC_PFBCR3] is changed to “0x40” to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M. No need to change [SDRAM_REG_MPC_PFBCR1] and [SDRAM_REG_MPC_PFBCR2].

```
#define SDRAM_REG_MPC_PFBCR0  (0x31)        /* Externa  ontrol Register 0 set value */
#define SDRAM_REG_MPC_PFBCR1  (0xD0)        /* Externa  ontrol Register 1 set value */
#define SDRAM_REG_MPC_PFBCR2  (0x00)        /* External Bus Control Register 2 set value */
#define SDRAM_REG_MPC_PFBCR3  (0x40)        /* External Bus Control Register 3 set value */
```

Set

Set

3) Setting SDC Control Register

Change the setting of [SDRAM_REG_BSC_SDCCR] (SDC Control Register) for the specifications of the connected SDRAM. This time, it is changed to "0x10" to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M.

```
#define SDRAM_REG_BSC_SDCCR   (0x10)        /*  ontrol Register set value
```

Set

4) Setting SDC Mode Register
Change the setting of [SDRAM_REG_BSC_SDCMOD] (SDC Mode Register) for the specifications of the connected SDRAM. This time, we have not changed it to meet the specifications of the SDRAM embedded with the Renesas Starter Kit + for RX72M.

5) SDRAM Access Mode Register
Change the setting of [SDRAM_REG_BSC_SDAMOD] (SDRAM Access Mode Register) for the specifications of the connected SDRAM. This time, we have not changed it to meet the specifications of the SDRAM embedded with the Renesas Starter Kit + for RX72M.

6) Setting SDRAM Refresh Control Register
Change the setting of [SDRAM_REG_BSC_SDRFCR] (SDRAM Refresh Control Register) for the specifications of the connected SDRAM. This time, it is changed to "0x44E0" to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M.

```
#define SDRAM_REG_BSC_SDRFCR  (0x44E0)      /* SDRAM Refresh Control Register set value */
```
Set

7) Setting SDRAM Initialization Register
Change the setting of [SDRAM_REG_BSC_SDIR] (SDRAM Initialization Register) for the specifications of the connected SDRAM. This time, it is changed to "0x0022" to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M.

```
#define SDRAM_REG_BSC_SDIR    (0x0022)      /* SDRAM Initialization Register set value */
```
Set

8) Setting SDRAM Address Register
Change the setting of [SDRAM_REG_BSC_SDADR] (SDRAM address register) for the specifications of the connected SDRAM. This time, it is changed to "0x00" to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M.

```
#define SDRAM_REG_BSC_SDADR   (0x00)      /* SDRAM Address Register set
```
Set

9) Setting SDRAM Timing Register
Change the setting of [SDRAM_REG_BSC_SDTR] (SDRAM Timing Register) for the specifications of the connected SDRAM This time, it is changed to "0x00031303" to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M.

```
#define SDRAM_REG_BSC_SDTR    (0x00031303)    /* SDRAM Timing Register set value */
```
Set

10)Setting SDRAM Mode Register
Change the setting of [SDRAM_REG_BSC_SDMOD] (SDRAM Mode Register) for the specifications of the connected SDRAM. This time, we have not changed it to meet the specifications of the SDRAM embedded with the Renesas Starter Kit + for RX72M.

9. Porting Sample Codes that Smart Configurator is used to Other RX Family

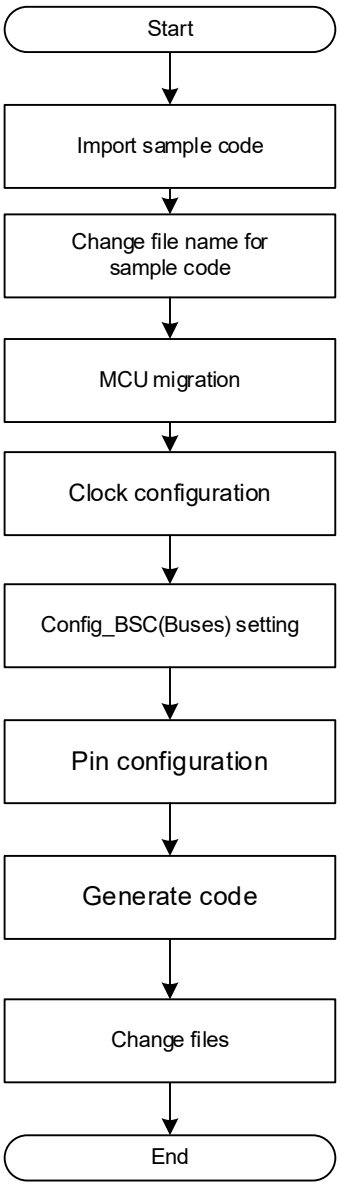
Sample codes included in this application note can be ported to other RX Family loaded with the exception vector table and software configurable interrupts. This section shows an example of porting sample code that RX65N and does used the Smart Configurator to the RX72M (Renesas Starter Kit+ for RX72M).

9.1 Before Porting

Confirm the following specifications before porting sample codes. If there is a difference in the specifications, the method described in this section may not be used. After making sure of the specifications, use this application.

The SDRAMC specification of the porting source and porting destination
The CMT specification of the porting source and porting destination

9.2 Porting Procedure Flow

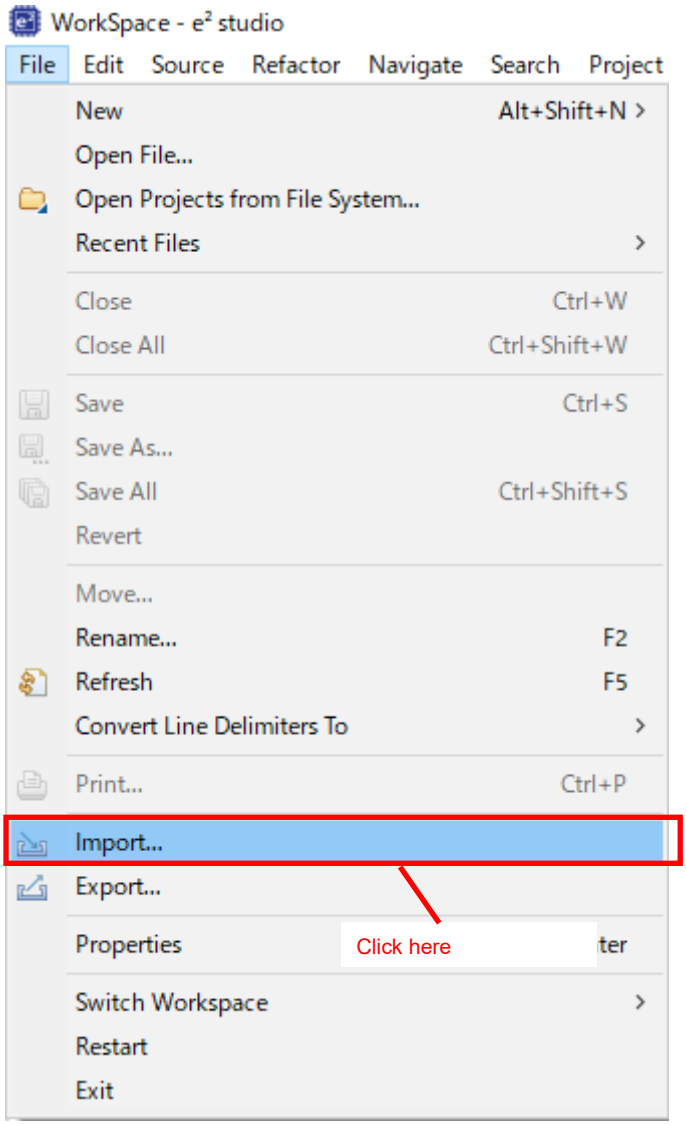


9.3 Porting Procedure

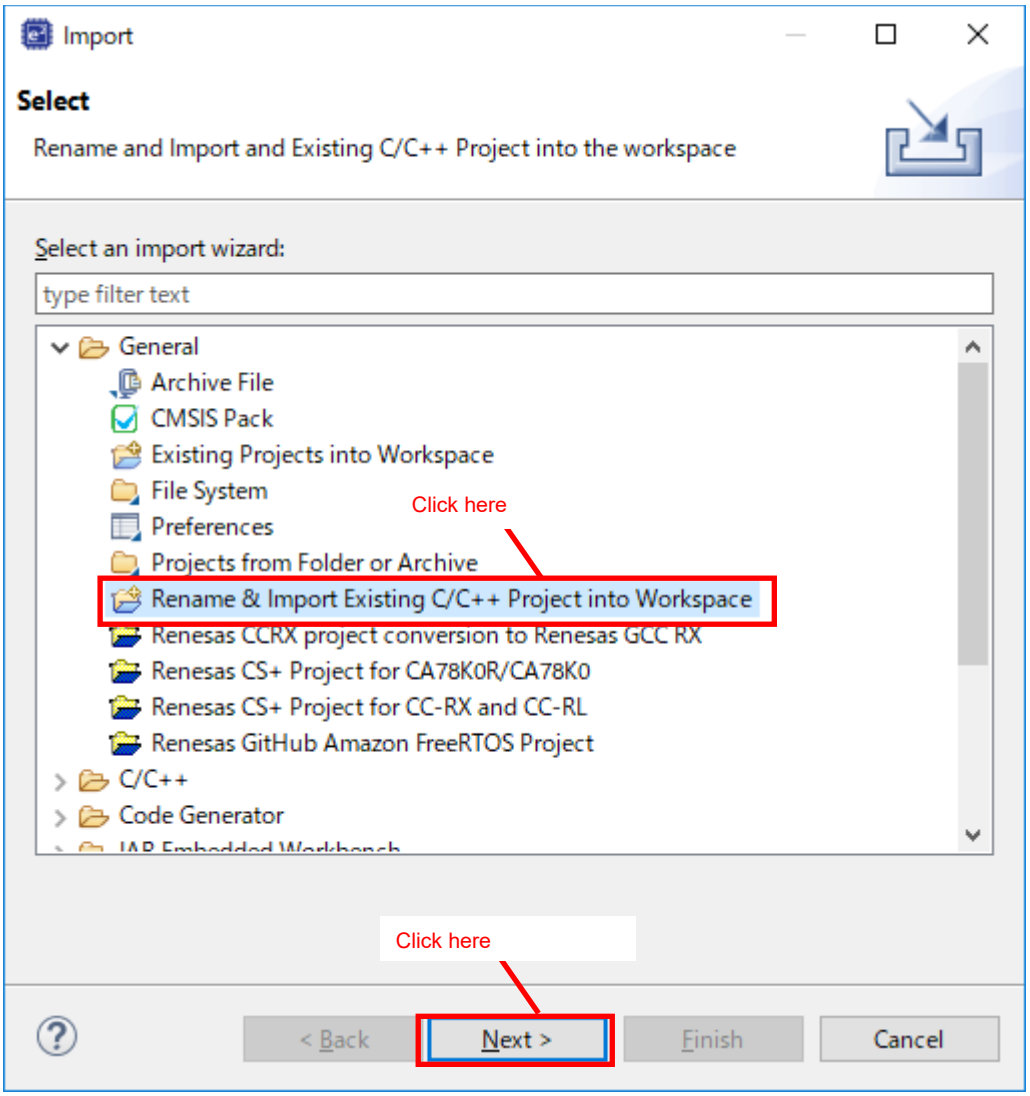
9.3.1 Import sample code

Import the sample code of this application

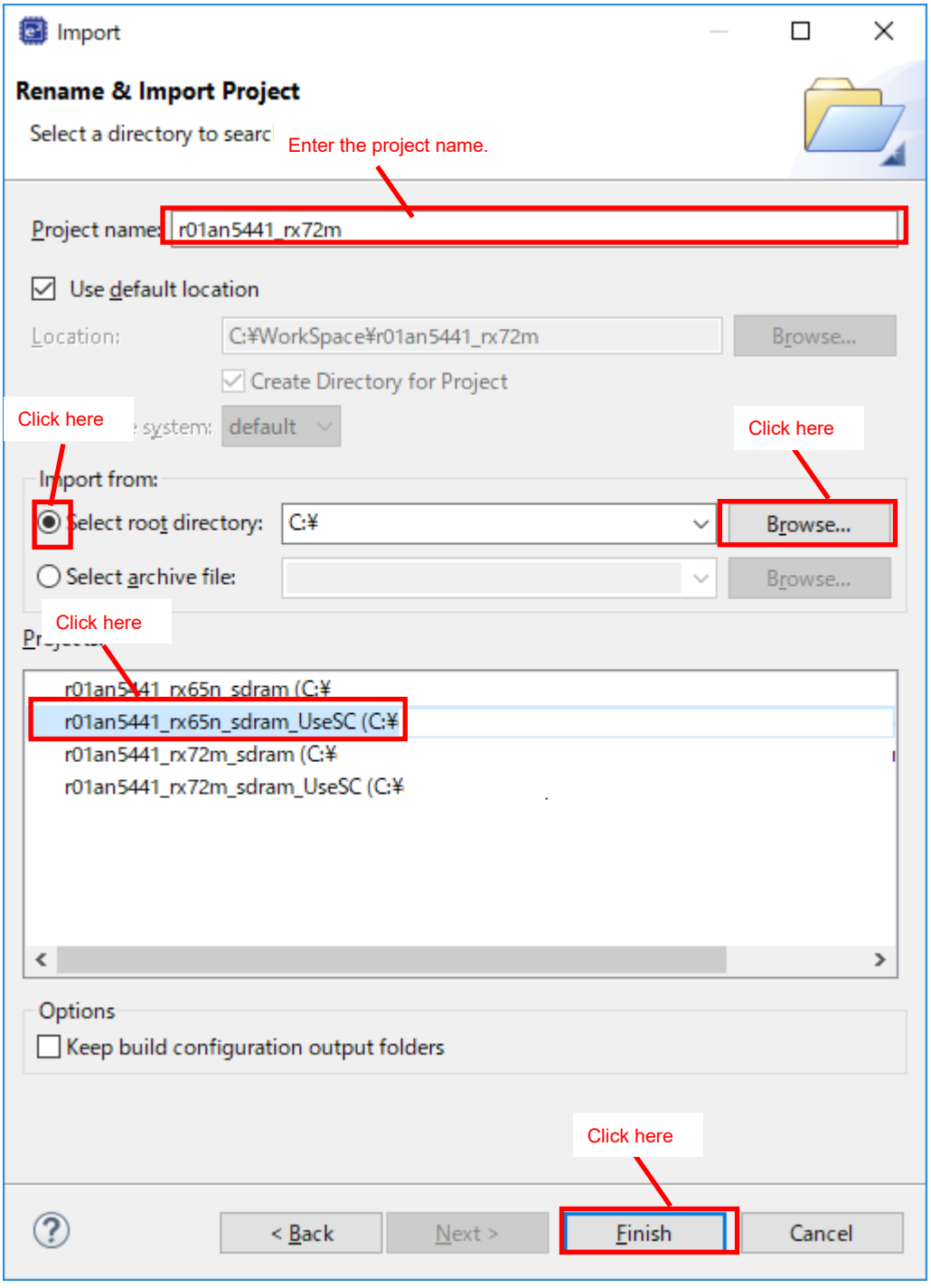
1) Start e² studio and click [File].
2) Click [Import]



3) Click [Rename & Import Existing C/C++ Project into Workspace].
4) Click [Next >].



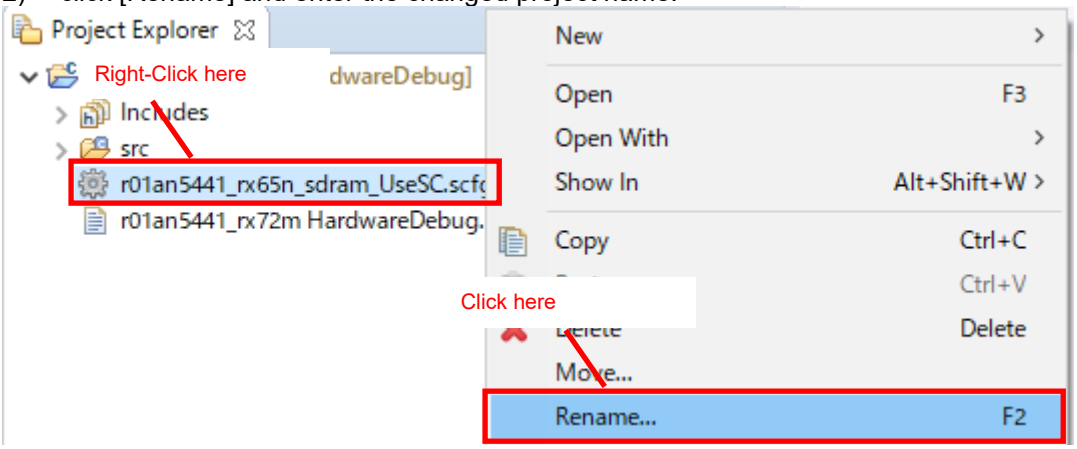
5) Enter the project name.
6) Click [Select root directory:].
7) Click [Browse] and Select the folder that contains the downloaded sample code.
8) Click [r01an5441_rx65n_sdram_SC].
9) Click [Finish].



9.3.2 Change file name for sample code

Change the file name of the imported sample code.

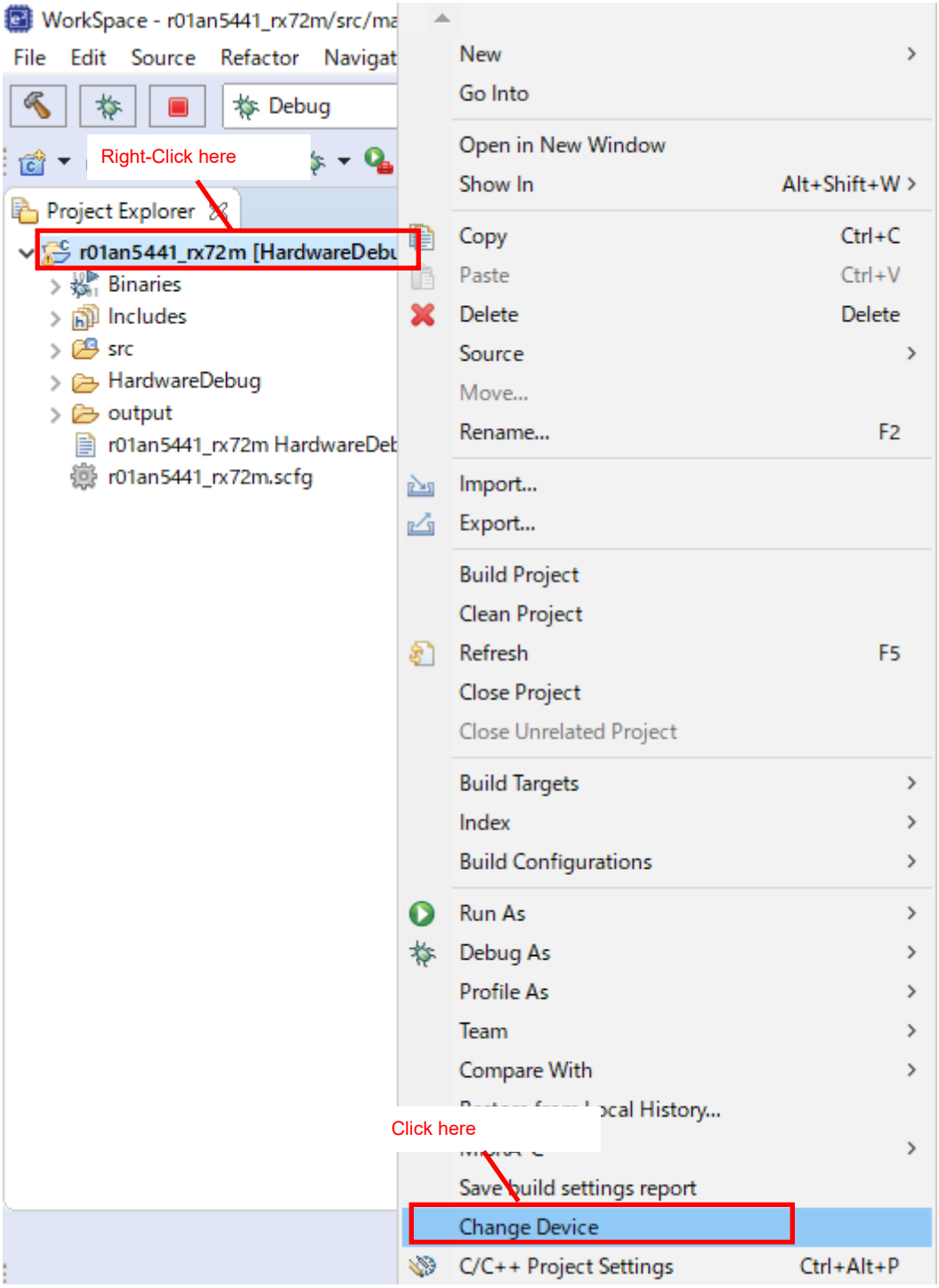
1) Right-click [r01an5441_rx65n_sdram_UseSC.scfg]
2) click [Rename] and enter the changed project name.



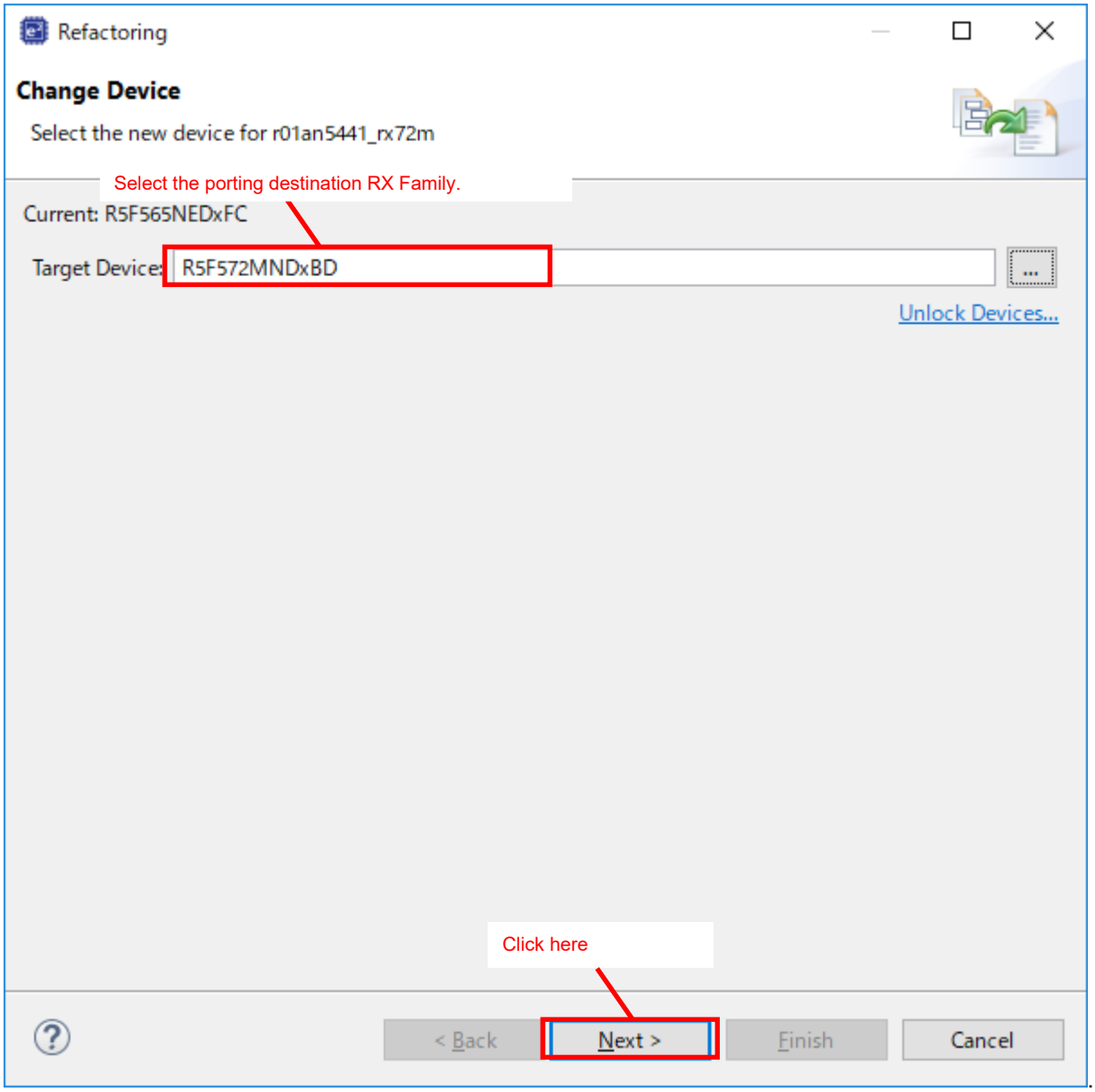
9.3.3 MCU migration

Perform MCU migration of the imported sample code.

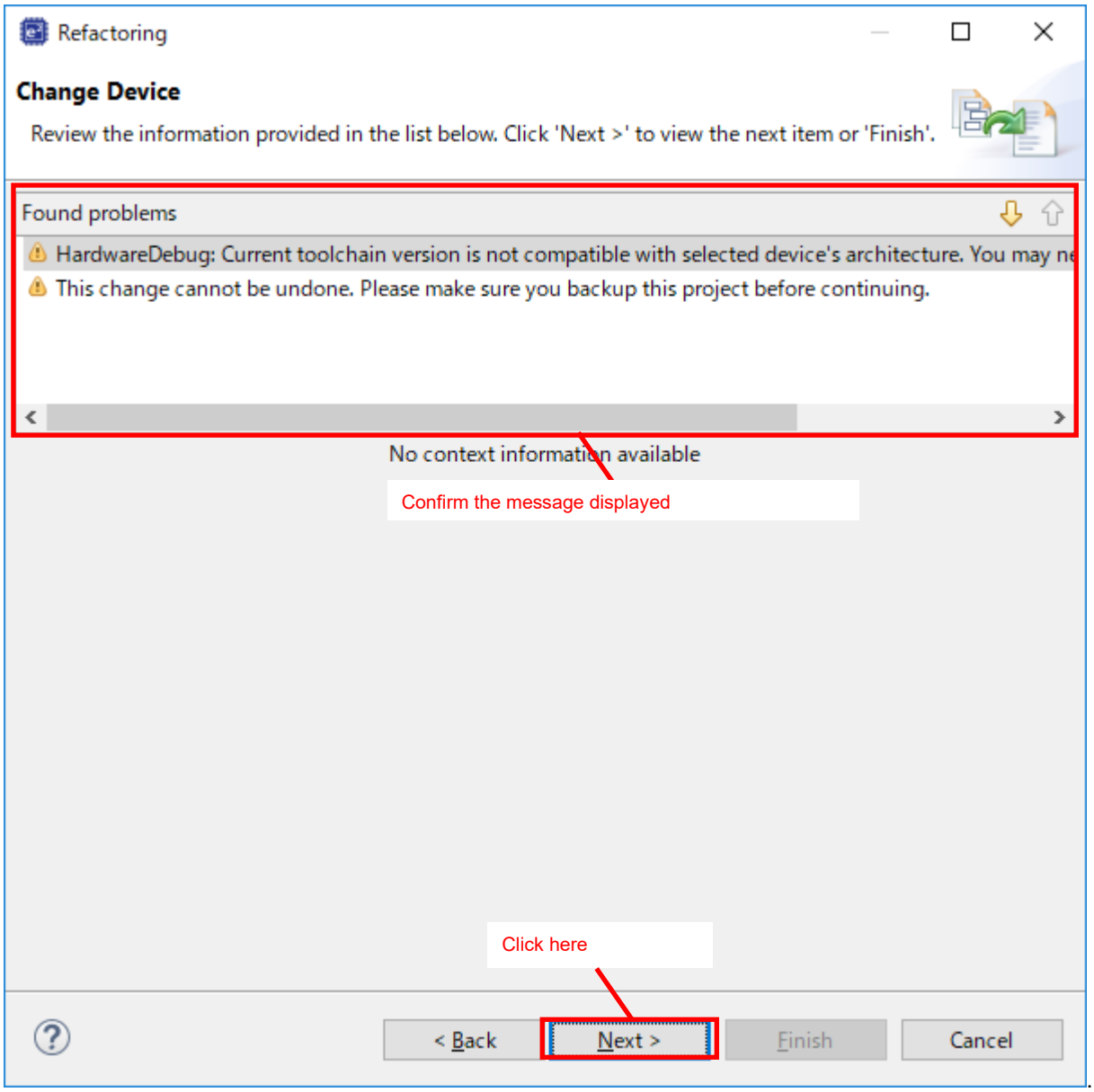
1) Right-click [<Project name>]
2) click [Change Device]



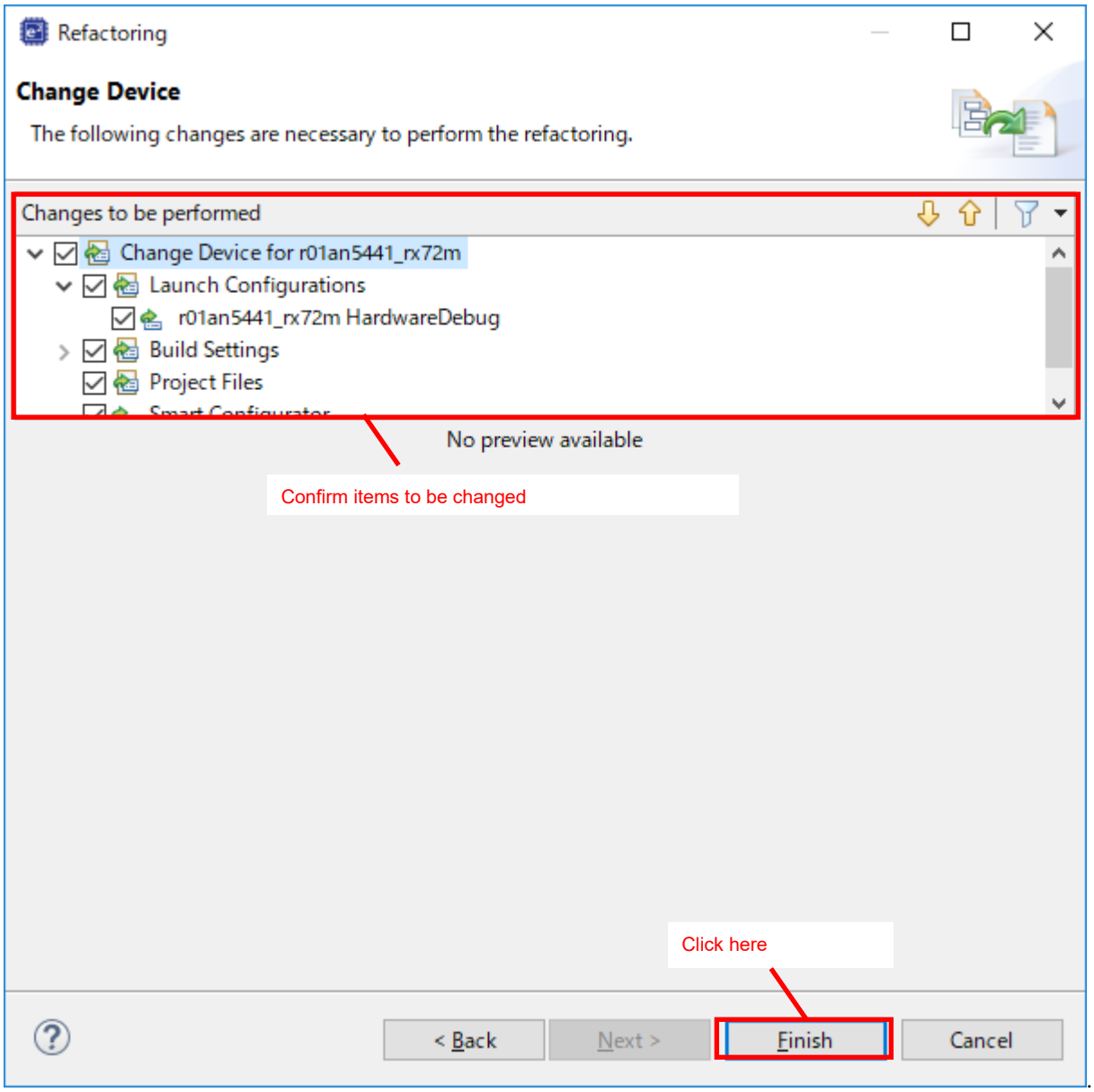
3) Change [Target Device:] to [R5F572MNDxBD].
(When porting to another RX Family, change to the porting destination RX Family.)
4) Click [Next >]



5) Confirm the message displayed in [Discovered Issues]
6) Click [Next >]



7) Confirm items to be changed
8) Click [Finish]

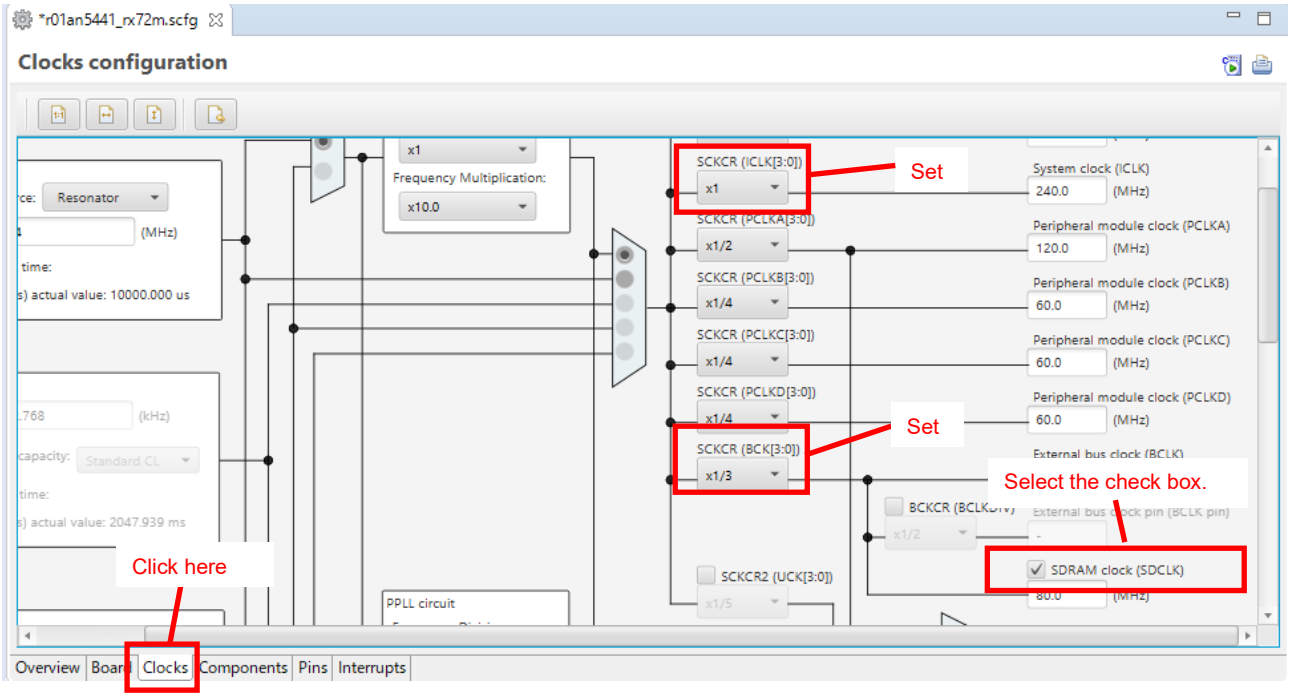


9.3.4 Clock configuration

Use Smart Configurator to change the clock settings.

1) Open [<project name>.scfg] in the created project and click [Clocks] on the tab at the bottom.

2) Change [SCKCR(BCK[3:0])] according to the specifications of the connected SDRAM and check the SDRAM clock (SDCLK). This time, to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M and the specifications of the application note, the [SCKCR(ICLK[3:0])] setting is set to divide by 1, [SCKCR(BCK[3:0])] setting is set to divide by 3.

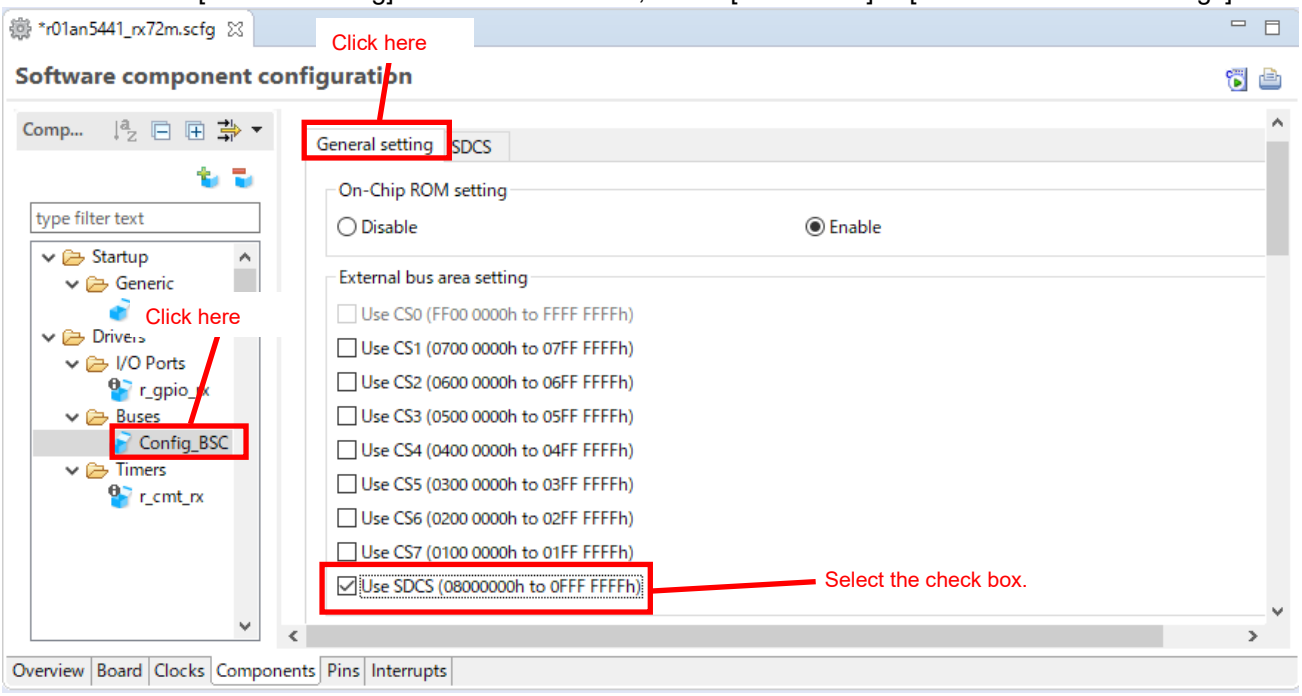


9.3.5 Config_BSC(Buses) setting

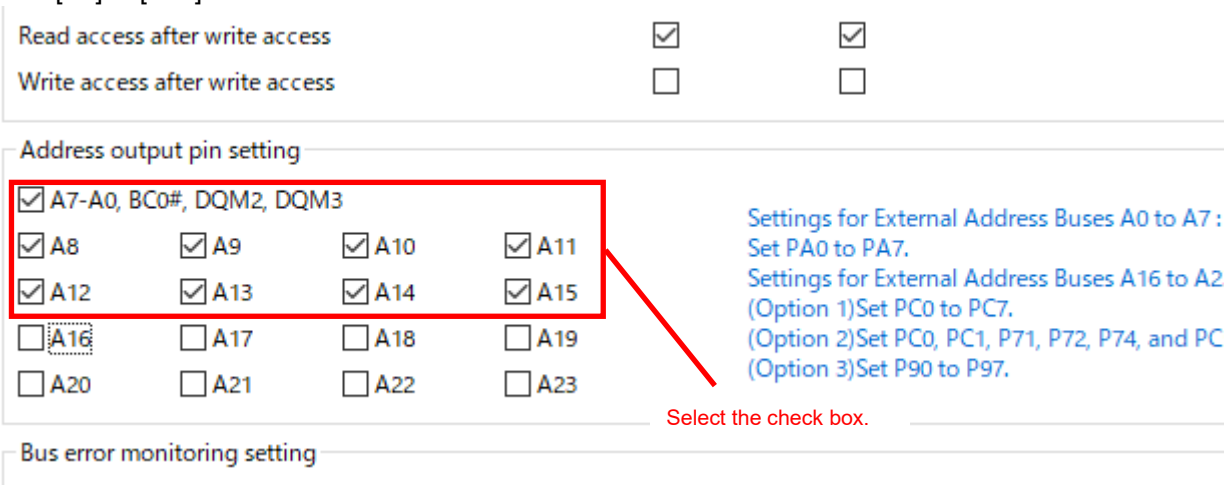
Change [Config_BSC(Buses)] in accordance with the porting destination environment.

The setting value when porting to RX72M (Renesas Starter Kit+ for RX72M) is shown below. When porting to another RX Family, change to the setting value corresponding to the porting destination environment. Also, see 7.Concept of register setting in target device of SDRAM specification.

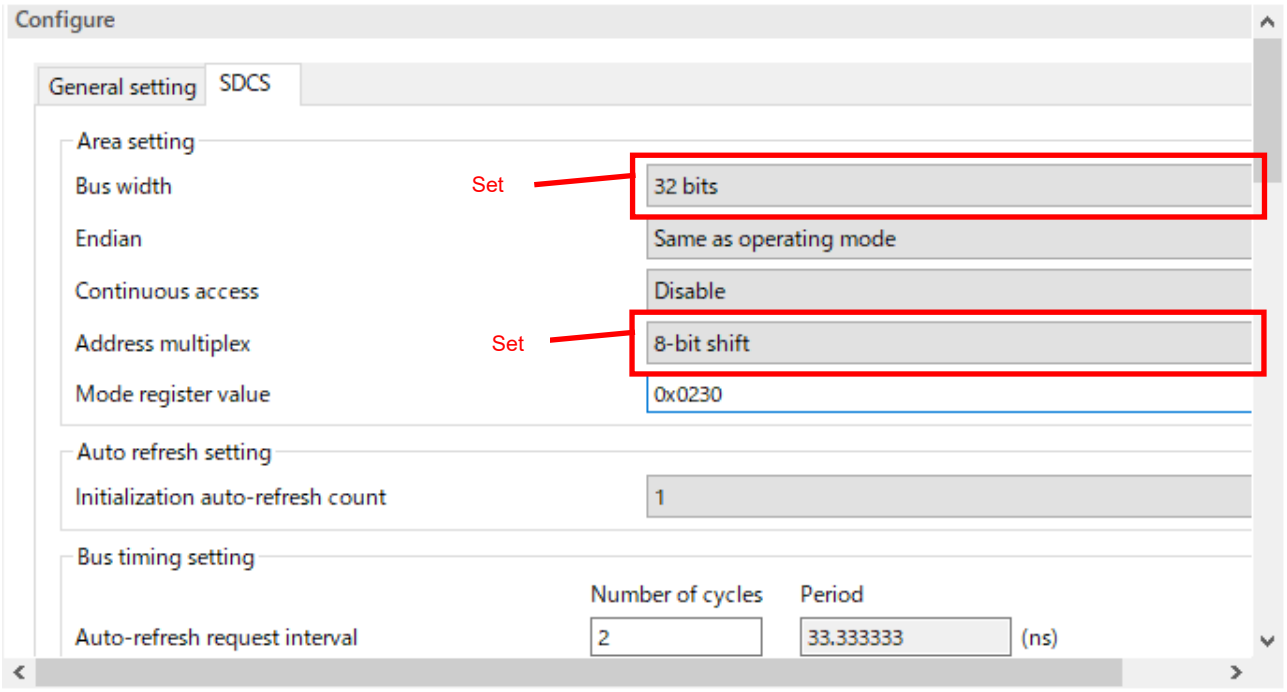
1) External bus area setting
Click [Config_BSC] from the components to display the setting screen. Confirm that the displayed setting screen is the [General setting] tab. To use SDRAM, check [Use SDCS] in [External bus area settings].



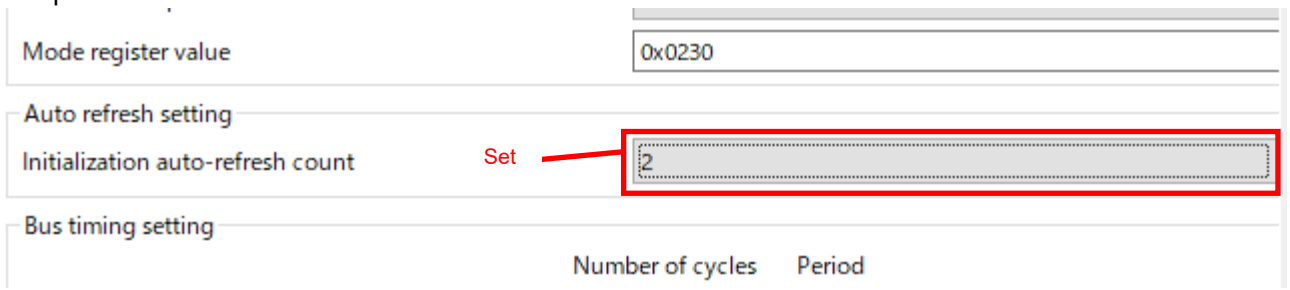
2) External bus area setting
Change the setting of [Address output pin setting] on the [General setting] tab according to the specifications of the connected SDRAM. This time, to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M, we have changed the settings from [A7-A0, BC0#, DQM2, DQM3] and [A8] to [A15].



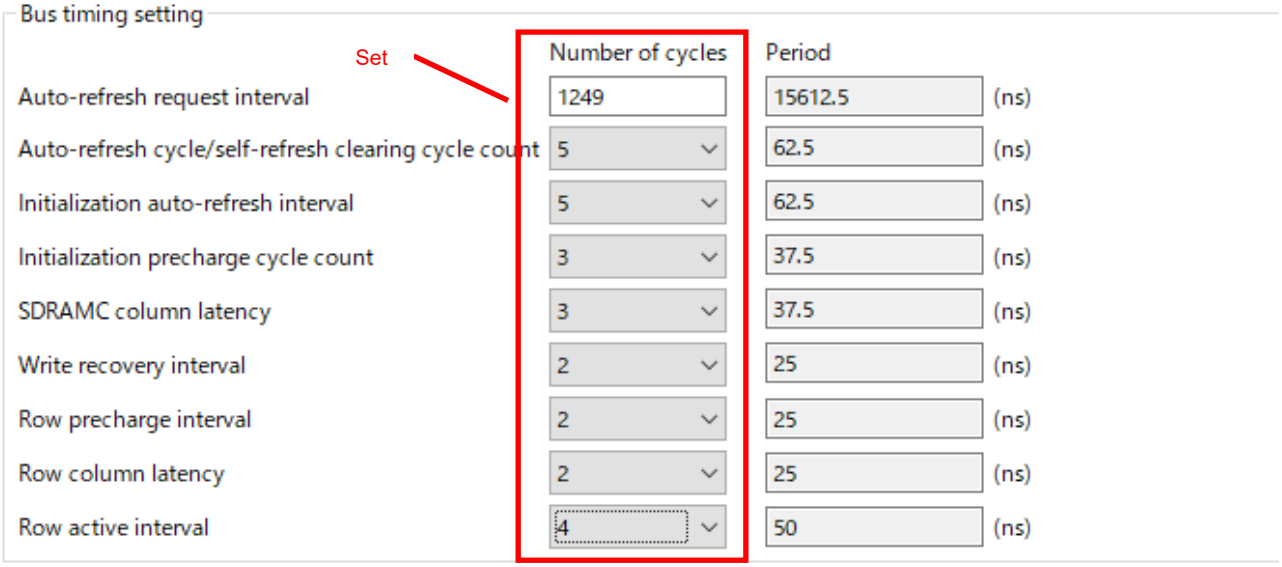
3) Area setting
Click the [SDCS] tab and change the settings for [Area setting] according to the specifications of the connected SDRAM. This time, [Bus width] is changed to [32 bits] and [Address multiplex] is changed to [8-bit shift] to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M.



4) Auto refresh setting
Change the contents of [Auto refresh setting] on the [SDCS] tab according to the specifications of the connected SDRAM. This time, [Initialization auto-refresh count] is changed to “2” to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M.



5) Bus timing setting
Change the setting of [Bus timing setting] on the [SDCS] tab according to the specifications of the connected SDRAM. This time, we changed the settings as shown below to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M.



- The [Auto-refresh request interval] should be [Refresh cycle / number of row addresses] or less. In case of MT48LC4M32B2P-6A, Auto-refresh request interval is 15625ns. This time, it is set to 15612.5ns by setting 1249 cycles.
- Set the [Auto-refresh cycle/ self-refresh clearing cycle count] setting to tRFC or more. In case of MT48LC4M32B2P-6A, tRFC is 60ns. This time, it is set to 62.5ns by setting 5 cycles.
- Set the [Initialization auto-refresh interval] to tRFC or more. In case of MT48LC4M32B2P-6A, tRFC is 60ns. This time, it is set to 62.5ns by setting 5 cycles.
- Set the [Initialization precharge cycle count] to tRP or more. In case of MT48LC4M32B2P-6A, tRP is 18ns. This time, it is set to 37.5ns by setting 3 cycles.
- Set the [SDRAMC column latency] setting so that it is the same as the column latency set in SDRAM. This time, it is set to same as SDRAM by setting 3 cycles.
- The [Write recovery interval] setting should be set to tWR or more. For MT48LC4M32B2P-6A, when SDCLK is 60MHz, tWR is 23.67ns. This time, it is set to 25ns by setting 2 cycles.
- Set the [Row precharge interval] to tRP or more. In case of MT48LC4M32B2P-6A, tRP is 18ns. This time, it is set to 25ns by setting 2 cycles.
- Set the [Row column latency] setting to be tRCD or more. For MT48LC4M32B2P-6A, tRCD is 18ns. This time, it is set to 25ns by setting 2 cycles.
- Set the [Row active Interval] to tRAS or more. In case of MT48LC4M32B2P-6A, tRAS is 42ns. This time, it is set to 50ns by setting 4 cycles.

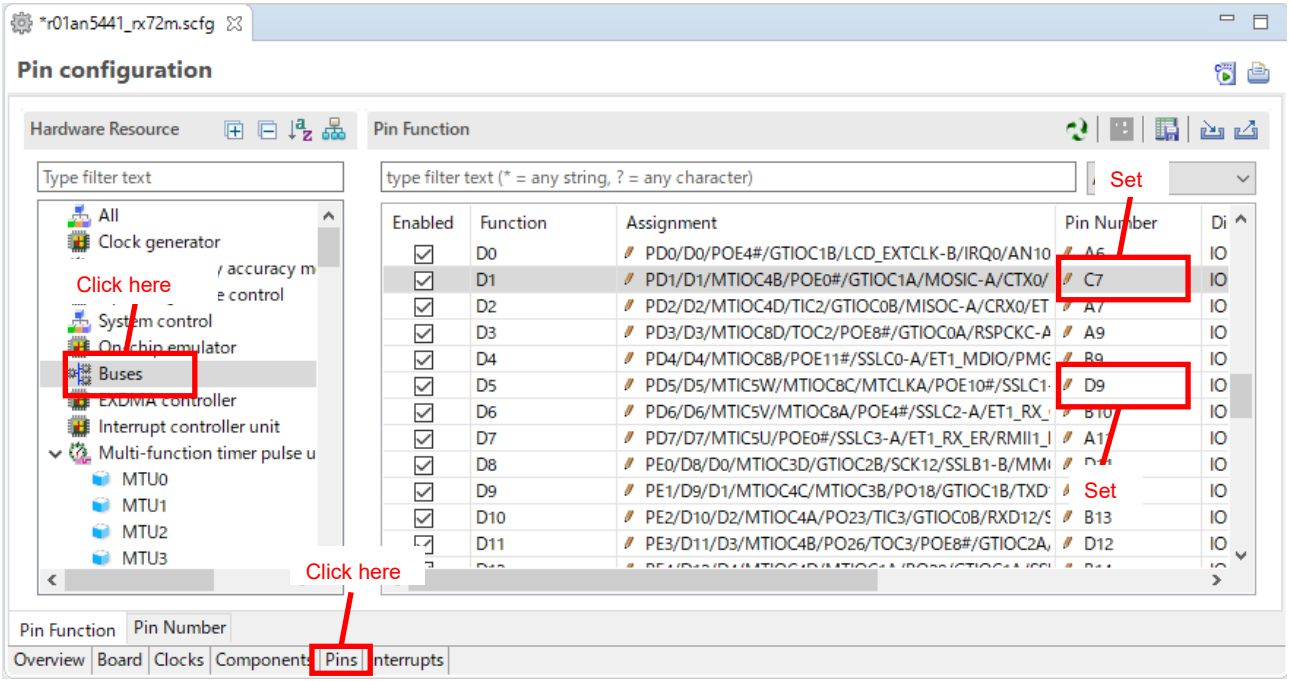
9.3.6 Pin configuration

Use Smart Configurator to set the pins to be used.

1) Open [<project name>.scfg] in the created project and click [Pins] on the tab at the bottom.

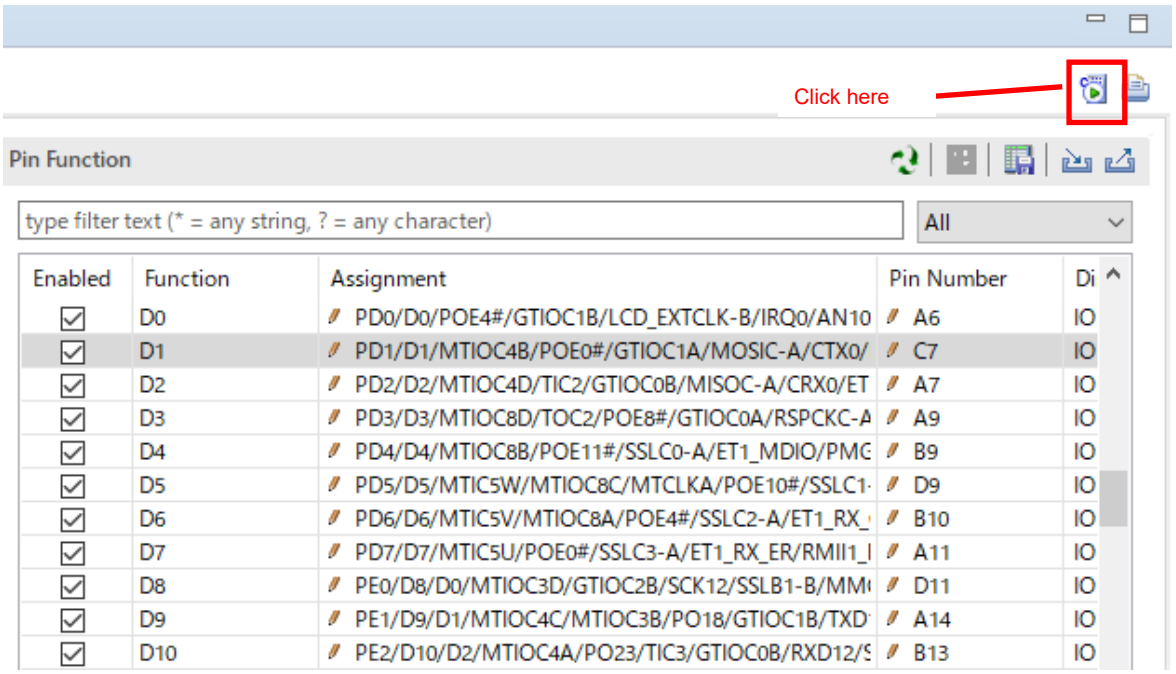
2) Click [Buses] of the hardware resource.

3) Change the pin settings according to the specifications of the connected SDRAM. This time, the terminal number of [D1] is changed to [C7], and the terminal number of [D5] is changed to [D9] to match the specifications of the SDRAM installed in the Renesas Starter Kit+ for RX72M.



9.3.7 Generate code

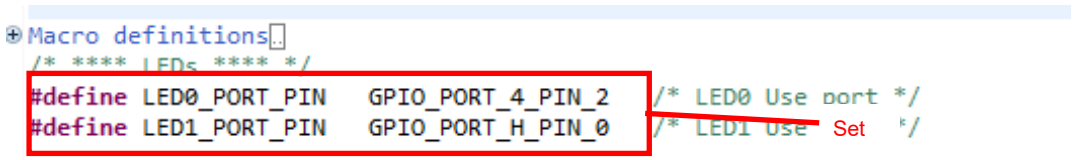
Click the [Generate code] button to generate the code.



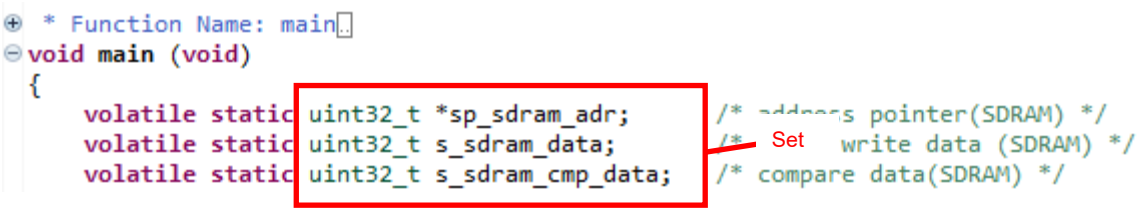
9.3.8 Changing Files

Change [main.c] copied in order to run the sample code of the application.

1) Setting Address Output Enable Register
   Change the settings of [LED0_PORT_PIN] and [LED1_PORT_PIN] according to the LED pin used. This time, [LED0_PORT_PIN] is set to [GPIO_PORT_4_PIN_2] and [LED1_PORT_PIN] is set to [GPIO_PORT_H_PIN_0] to match LED0 and LED1 of Renesas Starter Kit+ for RX72M.



2) Define the [*sp_sdram_adr], [s_sdram_data], and [s_sdram_data] variables of the [main] function with a type that matches the data bus width of the connected SDRAM. This time, it is defined in [uint32_t] to match the specifications of the SDRAM installed in Renesas Starter Kit+ for RX72M.



10. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

11. Reference Documents

User’s Manual: Hardware
RX65N Group, RX651 Group User's Manual: Hardware (R01UH0590)
RX72M Group User's Manual: Hardware (R01UH0804)
RX671 Group User's Manual: Hardware (R01UH0899)
RX72N Group User's Manual: Hardware (R01UH0824)
The latest version can be downloaded from the Renesas Electronics website.

Technical Update/Technical News
The latest information can be downloaded from the Renesas Electronics website.

User’s Manual: Development Tools
RX Family Compiler CC-RX User’s Manual (R20UT3248)
The latest version can be downloaded from the Renesas Electronics website.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jun. 30. 2020	—	First edition issued
1.10	Jan. 10, 2024	All	Added sample programs for the RSK-RX671, EK-RX671, and RSK-RX72N evaluation boards.
		1	Added a list of MCU groups in the “Introduction” section.
		4	Added a description of the SDRAM in Chapter 1, Specifications. Added a description of the operations that turn LED on and off separately for the EK-RX671 and RSK boards.
		5	Added Table 1.4.
		6, 7	Renamed Tables 2.1 and 2.2.
		8, 9, 10	Added Tables 2.3, 2.4, and 2.5.
		10	Added application notes related to the RX671 and RX72N.
		13	Added Figures 5.3 and 5.4.
		15, 16	Added Tables 5.3, 5.4, and 5.5.
		26	Added constant definitions for the RSK-RX671, RSK-RX72N, and EK-RX671.
		41	Modified Figure 6.16.
		42	Added Table 6.12.
		43	Added Figure 6.17.
		44	Added Tables 6.13, and 6.14.
		45	Modified Figure 6.18.
		80	Added documents for reference about the RX671 and RX72N.

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 be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

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

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

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/.

© 2024 Renesas Electronics Corporation. All rights reserved.