RX110 Group

Initial Setting

Abstract

This document describes settings required after a reset such as clock settings, stop processing for active peripheral functions after a reset, and nonexistent port initialization according to usage conditions selected in the header files.

Products

- RX110 Group 64-pin package with a ROM size between 16 KB and 128 KB
- RX110 Group 48-pin package with a ROM size between 16 KB and 128 KB
- RX110 Group 40-pin package with a ROM size between 16 KB and 64 KB
- RX110 Group 36-pin package with a ROM size between 16 KB and 64 KB

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Contents

1. Specifications ... 3
1.1 Stopping Peripheral Functions Operating after a Reset ... 3
1.2 Configuring Nonexistent Ports ... 3
1.3 Setting Clocks ... 4
1.3.1 Overview ... 4
1.3.2 Clock Specifications Used in the Sample Code ... 4
1.3.3 Selecting Clocks ... 5

2. Operation Confirmation Conditions ... 6

3. Reference Application Note ... 7

4. Software ... 7
4.1 Stop Processing for Active Peripheral Functions after a Reset ... 7
4.2 Nonexistent Port Initialization ... 8
4.2.1 Overview ... 8
4.2.2 Selecting the Number of Pins ... 8
4.3 Clock Settings ... 9
4.3.1 Clock Setting Procedure ... 9
4.3.2 Sub-Clock Oscillation Stabilization Time ... 10
4.4 Section Composition ... 11
4.5 File Composition ... 11
4.6 Option-Setting Memory ... 11
4.7 Constants ... 12
4.8 Functions ... 19
4.9 Function Specifications ... 20
4.10 Flowcharts ... 24
4.10.1 Main Processing ... 24
4.10.2 Stop Processing for Active Peripheral Functions after a Reset ... 24
4.10.3 Nonexistent Port Initialization ... 25
4.10.4 Clock Initialization ... 26
4.10.5 Main Clock Oscillation Setting ... 27
4.10.6 HOCO Clock Oscillation Setting ... 27
4.10.7 Sub-Clock Oscillation Setting ... 28
4.10.8 Sub-Clock Stop Setting ... 31
4.10.9 CMT0 Wait Start Setting, and CMT0 Wait Completion Check and Initialization ... 31

5. Importing a Project ... 33
5.1 Importing a Project in the e^2 studio ... 33
5.2 Importing a Project in CS+ ... 34

6. Sample Code ... 35

7. Reference Documents ... 35

Revision History ... 36

1. Specifications

In the sample code, peripheral functions operating after a reset are stopped, and nonexistent port and clock settings are configured. The application note assumes processing at power-on (cold start).

1.1 Stopping Peripheral Functions Operating after a Reset

Some peripheral functions operate at power-on, and the module-stop function is disabled for others. These include the DTC and RAM0. Although the sample code includes processing for stopping these peripheral functions, it is not executed in the sample code. Change the constant as required to execute processing.

1.2 Configuring Nonexistent Ports

Port direction registers which have nonexistent ports need to be specified with determined values. In the sample code, initial values are set for port direction registers in 64-pin products. Change the values according to the product used.

1.3 Setting Clocks

1.3.1 Overview

Clocks are configured in the following steps:

1. Sub-clock setting (including the associated RTC settings)
2. Main clock setting
3. HOCO clock setting
4. System clock switching

In this application note, the clock settings are switched by changing the constants defined in r_init_clock.h.

In the sample code, the main clock is used as the system clock, and the sub-clock and RTC are not used. Change the constant to select the required clock setting.

1.3.2 Clock Specifications Used in the Sample Code

Table 1.1 lists the Clock Specifications Used in the Sample Code. Values such as the oscillation stabilization time are calculated using values listed in Table 1.1.

Table 1.1 Clock Specifications Used in the Sample Code

Clock	Oscillation Frequency	Oscillation Stabilization Time	Remarks
Crystal/ceramic resonator for the main clock	16 MHz	4.2 ms (2)	Crystal used
Crystal for the sub-clock	32.768 kHz (1)	1.3 sec. (2)	For low clock loads
HOCO clock	32 MHz (1)	56 μs (3)	

Notes:

1. The clock is disabled in the sample code.
2. The oscillation stabilization time of a crystal/ceramic resonator differs depending on the traces, conditions of oscillation parameters, and other settings in the user system. Contact the crystal/ceramic resonator manufacturer to evaluate the user system and provide an appropriate oscillation stabilization time.
3. Refer to the Electrical Characteristics in the User's Manual: Hardware.

1.3.3 Selecting Clocks

In the sample code, users can select the system clock source, whether clocks are oscillating or stopped, and other settings by changing constants defined in r_init_clock.h. Refer to Table 4.7 and Table 4.8 for constants that can be changed.

Table 1.2 lists Examples of Clock Selections and Table 1.3 lists Examples of the Sub-Clock and RTC Selections.

Table 1.2 Examples of Clock Selections

No.		1	2	3
System clock		Main clock	HOCO clock	Sub-clock
Main clock		Oscillating	Stopped	Stopped
HOCO clock		Stopped	Oscillating	Stopped
Sub-clock		Stopped (1)	Stopped (1)	Oscillating
Operating power control mode		High-speed operating mode	High-speed operating mode	High-speed operating mode
Constants	SEL_SYSCLK	CLK_MAIN	CLK_HOCO	CLK_SUB
	SEL_MAIN	B_USE	B_NOT_USE	B_NOT_USE
	SEL_HOCO	B_NOT_USE	B_USE	B_NOT_USE
	SEL_SUB	B_NOT_USE (1)	B_NOT_USE (1)	B_ USE
	REG_OPCCR	OPCM_HIGH	OPCM_HIGH	OPCM_HIGH

Note:

1. When not using the sub-clock for the system clock, clock frequency accuracy measurement circuit (CAC), or the realtime clock (RTC), set the value of the SEL_SUB constant to B_NOT_USE. When using the sub-clock, refer to Table 1.3.

Table 1.3 Examples of the Sub-Clock and RTC Selections

Pattern	Sub-Clock	System Clock (2)		RTC	
	Crystal	Used/ Not Used	Value in SEL_SUB (1)	Used/ Not Used	Value in SEL_RTC (1)
Sub-clock not used	None	—	B_NOT_USE	—	B_NOT_USE
Sub-clock used as the system clock	Used	Used	B_USE	Not used	B_NOT_USE
Sub-clock used for the RTC	Used	Not used	B_NOT_USE	Used	B_USE
Sub-clock used for the system clock and the RTC	Used	Used	B_USE	Used	B_USE

Notes:

1. When setting B_USE to both the SEL_SUB and SEL_RTC constants or either of them, the sub-clock oscillates.
2. The sub-clock oscillation is controlled by bits SOSCCR.SOSTP and RCR3.RTCEN. When the sub-clock is used as the system clock, it is controlled by the SOSCCR.SOSTP bit, and when the sub-clock is used as the RTC count source, it is controlled by the RCR3.RTCEN bit. Therefore the initial setting for the sub-clock differs depending on whether the sub-clock is used as the system clock or not. Also the sub-clock starts oscillating at power-on. Thus processing to stop the sub-clock is performed even when the sub-clock is not used.

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

<table>
<tr><th colspan="2">Item</th><th>Contents</th></tr>
<tr><td colspan="2">MCU used</td><td>RF51105ADFM (RX110 Group)</td></tr>
<tr><td rowspan="2">Operating frequencies</td><td>When the main clock is selected as the system clock</td><td>- Main clock: 16 MHz
- Sub-clock: 32.768 kHz (stopped when the sub-clock is not used)
- LOCO: 4 MHz
- HOCO: Stopped
- System clock (ICLK): 16 MHz (main clock divided by 1)
- Peripheral module clock B (PCLKB): 16 MHz (main clock divided by 1)</td></tr>
<tr><td>When the HOCO clock is selected as the system clock</td><td>- Main clock: Stopped
- Sub-clock: 32.768 kHz (stopped when the sub-clock is not used)
- LOCO: 4 MHz
- HOCO: 32 MHz
- System clock (ICLK): 32 MHz (HOCO divided by 1)
- Peripheral module clock B (PCLKB): 32 MHz (HOCO divided by 1)</td></tr>
<tr><td colspan="2">Operating voltage</td><td>3.3 V</td></tr>
<tr><td colspan="2">Integrated development environment</td><td>Renesas Electronics Corporation
e^2 studio Version 7.5.0</td></tr>
<tr><td colspan="2" rowspan="2">C compiler</td><td>Renesas Electronics Corporation
C/C++ Compiler Package for RX Family V.3.01.00</td></tr>
<tr><td>Compile options
The default setting is used in the integrated development environment.</td></tr>
<tr><td colspan="2">iodefine.h version</td><td>Version 1.0B</td></tr>
<tr><td colspan="2">Endian</td><td>Little endian</td></tr>
<tr><td colspan="2">Operating mode</td><td>Single-chip mode</td></tr>
<tr><td colspan="2">Processor mode</td><td>Supervisor mode</td></tr>
<tr><td colspan="2">Sample code version</td><td>Version 1.02</td></tr>
</table>

3. Reference Application Note

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

- RX Family Coding Example for Wait Processing by Software Rev. 1.00 (R01AN1852EJ).

The wait function in the reference application note is used in the sample code accompanying this application note. The revision number of the reference application note is as of when this application note was made. However the latest version is always recommended. Visit the Renesas Electronics Corporation website to check and download the latest version.

4. Software

In the sample code, peripheral functions operating after a reset are stopped, nonexistent ports are configured, and then clock settings are configured.

4.1 Stop Processing for Active Peripheral Functions after a Reset

Peripheral functions that are operating after a reset are stopped in this processing.

The module-stop state is canceled after a reset only for modules listed in Table 4.1. To enter the module-stop state, set the module stop bit to 1 (transition to the module-stop state is made). Power consumption can be reduced by entering the module-stop state.

In the sample code, the MSTP_STATE_"target module" constant is set to 0 (MODULE_STOP_DISABLE), so the target module does not enter the module-stop state. When the system requires a module to enter the module-stop state, set the constant in r_init_stop_module.h to 1 (MODULE_STOP_ENABLE).

Table 4.1 lists the Peripheral Modules whose Module-Stop States are Canceled after a Reset.

Table 4.1 Peripheral Modules whose Module-Stop States are Canceled after a Reset

Peripheral Module	Module Stop Bit	Value after a Reset	Value when not Using the Module
DTC	MSTPCRA.MSTPA28 bit	0 (module-stop state is canceled)	1 (transition to the module-stop state is made)
RAM0	MSTPCRC.MSTPC0 bit		

4.2 Nonexistent Port Initialization

4.2.1 Overview

The port direction registers which have nonexistent ports need to be specified with determined values. After the nonexistent port initialization 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 values listed in Table 4.2 and Table 4.3 to the I/O select bits in the PDR registers, and set the output data store bits in the PODR registers to 0.

Table 4.2 and Table 4.3 list Setting Values in the Port Direction Registers.

Table 4.2 Setting Values in the Port Direction Registers (1/2)

Port Symbol	64-Pin Package								48-Pin Package							
	b7	b6	b5	b4	b3	b2	b1	b0	b7	b6	b5	b4	b3	b2	b1	b0
PORT0	0	0		0		0	0	0	0	0	1	0	1	0	0	0
PORT1					0	0	0	0					0	0	0	0
PORT2			0	0	0	0	0	0			0	0	0	0	0	0
PORT3	0	0	0	0	0				0	0	0	0	0	1	1	1
PORT4	0		0						0		0					
PORT5	0	0			0	0	0	0	0	0	1	1	0	0	0	0
PORTA	0		0			0			0		0			0		1
PORTB				0		0				1		0		0		
PORTC																
PORTE										1	1					
PORTH	0	0	0	0					0	0	0	0				
PORTJ			0	0	0	0	0	0			0	0	0	0	0	0

Table 4.3 Setting Values in the Port Direction Registers (2/2)

Port Symbol	40-Pin Package								36-Pin Package							
	b7	b6	b5	b4	b3	b2	b1	b0	b7	b6	b5	b4	b3	b2	b1	b0
PORT0	0	0	1	0	1	0	0	0	0	0	1	0	1	0	0	0
PORT1					0	0	0	0					0	0	0	0
PORT2			0	0	0	0	0	0		1	0	0	0	0	0	0
PORT3	0	0	0	0	0		1	1	0	0	0	0	0	1	1	1
PORT4	0		0	1	1			1	0	1	0	1	1			1
PORT5	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0
PORTA	0		0			0		1	0		0			0	1	1
PORTB	1	1	1	0		0	1		1	1	1	0		0	1	
PORTC	1	1	1		1	1	1	1	1	1	1		1	1	1	1
PORTE	1	1	1						1	1	1					
PORTH	0	0	0	0					0	0	0	0				
PORTJ			0	0	0	0	0	0			0	0	0	0	0	0

4.2.2 Selecting the Number of Pins

The number of pins in the sample code is set for the 64-pin package (PIN_SIZE=64). This application note covers 64-pin, 48-pin, 40-pin, and 36-pin packages. When using products other than 64 pin-package, change PIN_SIZE in r_init_port_initialize.h to the number of pins on the package used.

4.3 Clock Settings

4.3.1 Clock Setting Procedure

Table 4.4 lists the Clock Setting Procedure with each processing and setting in the sample code. In the sample code, the main clock is operating, and the sub-clock and HOCO are stopped.

Table 4.4 Clock Setting Procedure

Step	Processing	Details		Setting in the Sample Code
1	Sub-clock setting (1)	Not used	The sub-clock control circuit is initialized.	Sub-clock is not used.
		Used	The sub-clock control circuit is initialized and the sub-clock oscillation is enabled. Then wait for the oscillation stabilization time (2) by software is processed.	
2	Main clock setting (1)	Not used	No setting is required.	Main clock is used.
		Used	The main clock drive capability is set, the MOSCWTCR register is set with a wait time until the main clock output is provided to the internal clock, and then the main clock oscillation is enabled. Then wait for the oscillation stabilization time is processed.	
3	HOCO clock setting (1)	Not used	No setting is required.	HOCO clock is not used.
		Used	The HOCOWTCR register is set with a wait time until the HOCO clock output is provided to the internal clock, and then the HOCO clock oscillation is enabled. Then wait for the oscillation stabilization time is processed.	
4	Operating power control mode setting	The operating power control mode is set according to the operating frequency and operating voltage in the user system.		High-speed operating mode is set
5	Clock division ratio setting	The clock division ratio is changed.		ICLK, PCLKB, PCLKD, FCLK: Divided by 1
6	System clock switching	The system clock is switched according to the user system.		Switched to the main clock.

Notes:

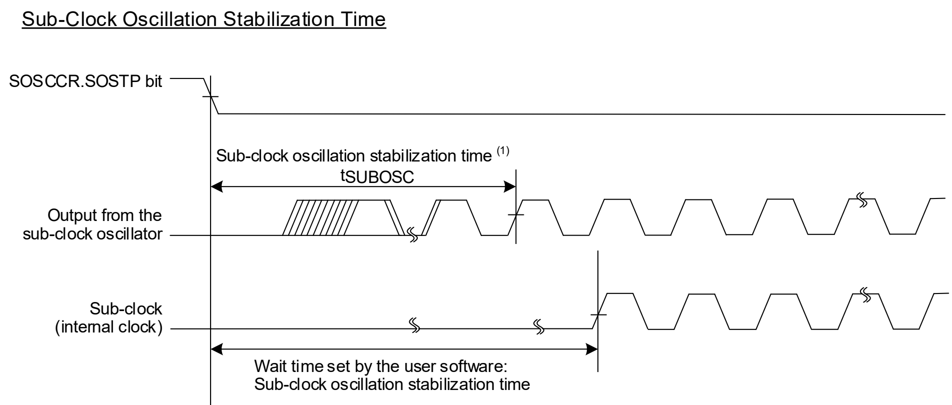
1. When selecting each clock usage, change the appropriate constant in r_init_clock.h as required.
2. Refer to 4.3.2 Sub-Clock Oscillation Stabilization Time for details on the sub-clock oscillation stabilization time.

4.3.2 Sub-Clock Oscillation Stabilization Time

This section describes the sub-clock oscillation stabilization time shown in Figure 4.1.

The sub-clock oscillation stabilization time (tSUBOSC) is set to the sub-clock oscillation stabilization time recommended by the crystal/ceramic resonator manufacturer. The wait time by software is set to a value greater than or equal to tSUBOSC.

tSUBOSC used in the sample code is 1.3 seconds, thus the wait time by software is 1.31 seconds here.



Note:
1. Contact the crystal/ceramic resonator manufacturer to determine the oscillation stabilization time of a crystal/ceramic resonator for the user system. The oscillation stabilization time is not a condition for MCU operation, but for a crystal/ceramic resonator to start oscillation.

Figure 4.1 Sub-Clock Oscillation Stabilization Time

4.4 Section Composition

Table 4.5 lists the section data changed in the sample code. For details on adding, changing, and deleting section, refer to the RX Family C/C++ Compiler Package User's Manual.

Table 4.5 Section Data Changed in the Sample Code

Section Name	Change	Address	Function
End_of_RAM	Addition	0000 3FFCh*	Last address of internal RAM

Notes: The capacity of the internal RAM is different every product. Change the address according to the product to be used.

4.5 File Composition

Table 4.6 lists the Files Used in the Sample Code. Files generated by the integrated development environment should not be listed in this table.

Table 4.6 Files Used in the Sample Code

File Name	Outline	Remarks
main.c	Main processing	
r_init_stop_module.c	Stop processing for active peripheral functions after a reset	
r_init_stop_module.h	Header file for r_init_stop_module.c	
r_init_port_initialize.c	Nonexistent port initialization	
r_init_port_initialize.h	Header file for r_init_port_initialize.c	
r_init_clock.c	Clock initialization	
r_init_clock.h	Header file for r_init_clock.c	
r_delay.c	Wait processing by software	
r_delay.h	Header file for r_delay.c	

4.6 Option-Setting Memory

Table 4.7 lists the Option-Setting Memory Configured in the Sample Code. When necessary, set a value suited to the user system.

Table 4.7 Option-Setting Memory Configured in the Sample Code

Symbol	Address	Setting Value	Contents
OFS0	FFFF FF8Fh to FFFF FF8Ch	FFFF FFFFh	The IWDT is stopped after a reset.
OFS1	FFFF FF8Bh to FFFF FF88h	FFFF FFFFh	Fast startup time at power-on is disabled. The voltage monitor 1 reset is disabled after a reset. HOCO oscillation is disabled after a reset.
MDE	FFFF FF83h to FFFF FF80h	FFFF FFFFh	Little endian

4.7 Constants

Table 4.8 and Table 4.9 list the constants used in the sample code, which can be changed by users. Table 4.10 lists the constants used in the sample code, which cannot be changed by users. Table 4.11 lists the Constants when a 64-Pin Package is Used (PIN_SIZE=64), Table 4.12 lists the Constants when a 48-Pin Package is Used (PIN_SIZE=48), Table 4.13 lists the Constants when a 40-Pin Package is Used (PIN_SIZE=40), and Table 4.14 lists the Constants when a 36-Pin Package is Used (PIN_SIZE=36).

Table 4.8 Constants Used in the Sample Code (1/2)
(Users can change the constants listed in this table.)

Constant Name	Setting Value	Contents
SEL_MAIN (1)	B_USE	Selection of the main clock operation: - B_USE: Used (main clock oscillating) - B_NOT_USE: Not used (main clock stopped)
MAIN_CLOCK_HZ (1)	16,000,000L	Oscillation frequency of a crystal/ceramic resonator for the main clock (Hz)
REG_MOFCR (1)	20h	Setting for the drive capability of the main clock oscillator (setting value in the MOFCR register)
REG_MOSCWTCR (1)	06h	Setting value in the main clock wait control register
SEL_HOCO	B_NOT_USE	Selection of the HOCO clock operation: - B_USE: Used (HOCO clock oscillating) - B_NOT_USE: Not used (HOCO clock stopped)
REG_HOCOWTCR	06h	Setting value in the HOCO wait control register
SEL_SUB (1, 2)	B_NOT_USE	Selection of the sub-clock usage for the system clock: - B_USE: Used - B_NOT_USE: Not used
SEL_RTC (1, 2)	B_NOT_USE	Selection of the sub-clock usage for the RTC count source: - B_USE: Used - B_NOT_USE: Not used
SUB_CLOCK_HZ (1)	32,768 L	Oscillation frequency of a crystal for the sub-clock (Hz)
WAIT_TIME_FOR_SUB_OSCILLATION (1)	1,310,000,000L	Sub-clock oscillation stabilization time (ns)
REG_RCR3 (1)	DRIVE_MID	Selection of the sub-clock oscillator drive capability: - DRIVE_MID: Medium drive capacity (4.4 pF type) - DRIVE_HIGH: High drive capacity (6.0 pF type) - DRIVE_LOW: Low drive capacity (3.7 pF type) - DRIVE_TYP: Drive capacity for standard CL
SEL_CNTMD (1)	CNTMD_CAL	Selection of the real-time clock count mode - CNTMD_CAL: Calendar count mode - CNTMD_BIN: Binary count mode

Notes:
1. Change the setting value in r_init_clock.h according to the user system.
2. The sub-clock operation is set to be oscillating by setting B_USE (sub-clock used) to either of the SEL_SUB constant or SEL_RTC constant, or both.

Table 4.9 Constants Used in the Sample Code (2/2)
(Users can change the constants listed in this table.)

Constant Name	Setting Value	Contents
SEL_SYSCLK (1)	CLK_MAIN	Clock source selection for the system clock - CLK_HOCO: HOCO clock - CLK_MAIN: Main clock - CLK_SUB: Sub-clock
REG_OPCCR (1)	OPCM_HIGH	Selection of the operating power control mode (5) - OPCM_HIGH: High-speed operating mode - OPCM_MID: Middle-speed operating mode - OPCM_LOW: Low-speed operating mode (4)
REG_SOPCCR (1)	SOPCM_HIGH	Selection of the sub operating power control mode (5) - SOPCM_HIGH: High-speed operating mode - SOPCM_MID: Middle-speed operating mode - SOPCM_LOW: Low-speed operating mode (4)
MSTP_STATE_DTC (2)	MODULE_STOP_DISABLE	Selection of the module-stop state for DTC - MODULE_STOP_DISABLE: Module-stop state canceled - MODULE_STOP_ENABLE: Entering the module-stop state
MSTP_STATE_RAM0 (2)	MODULE_STOP_DISABLE	Selection of the module-stop state for RAM0 - MODULE_STOP_DISABLE: Operating - MODULE_STOP_ENABLE: Stopped
PIN_SIZE (3)	64	Number of pins on the product used
SUB_CLOCK_CYCLE	(1000000L/SUB_CLOCK_HZ)	Sub-clock cycle (μs)
LOCO_CLOCK_KHZ	(4560L)	LOCO frequency (kHz)
FOR_CMT0_TIME	(7018*8)	Count period (ns) of the timer (CMT0) to wait for oscillation to be stabilized (LOCO = 4.56 MHz (max.) divided by 8 and PCLK divided by 32)

Notes:

1. Change the setting value in r_init_clock.h according to the user system.
2. Change the setting value in r_init_stop_module.h according to the user system.
3. Change the setting value in r_init_port_initialize.h according to the user system.
4. Low-speed operating mode can be selected only when the sub-clock is used as the system clock.
5. The ranges of the operating frequency and operating voltage differ depending on operating modes. Refer to the User's Manual: Hardware for details.

Table 4.10 Constants Used in the Sample Code
(Users cannot change the constants listed in this table.)

Constant Name	Setting Value	Contents
B_NOT_USE	0	Not used
B_USE	1	Used
DRIVE_MID	00h	Sub-clock: Medium drive capacity (4.4 pF type)
DRIVE_HIGH	02h	Sub-clock: High drive capacity (6.0 pF type)
DRIVE_LOW	04h	Sub-clock: Low drive capacity (3.7 pF type)
DRIVE_TYP	08h	Sub-clock: Drive capacity for standard CL
CNTMD_CAL	0	RTC: Calendar count mode
CNTMD_BIN	1	RTC: Binary count mode
CLK_MAIN	0200h	Clock source: Main clock
CLK_HOCO	0100h	Clock source: HOCO
CLK_SUB	0300h	Clock source: Sub-clock
REG_SCKCR (1)	0000 0000h	Setting for the internal clock division ratio (setting value in the SCKCR register)
SOPCM_MID	00h	Sub-operating power control mode: Middle-speed operating mode
OPCM_MID	02h	Operating power control mode: Middle-speed operating mode
SOPCM_HIGH	00h	Sub-operating power control mode: High-speed operating mode
OPCM_HIGH	00h	Operating power control mode: High-speed operating mode
SOPCM_LOW	01h	Sub-operating power control mode: Low-speed operating mode
OPCM_LOW	00h	Operating power control mode: Low-speed operating mode
MODULE_STOP_ENABLE	1	Transition to the module stop-state is made
MODULE_STOP_DISABLE	0	Module stop-state is canceled

Note:

1. The setting value varies depending on the clock source of the system clock selected.

Table 4.11 Constants when a 64-Pin Package is Used (PIN_SIZE=64)

Constant Name	Setting Value	Contents
DEF_P0PDR_MASK	28h	Mask value for the port P0 direction register
DEF_P1PDR_MASK	F0h	Mask value for the port P1 direction register
DEF_P2PDR_MASK	C0h	Mask value for the port P2 direction register
DEF_P3PDR_MASK	07h	Mask value for the port P3 direction register
DEF_P4PDR_MASK	5Fh	Mask value for the port P4 direction register
DEF_P5PDR_MASK	30h	Mask value for the port P5 direction register
DEF_PAPDR_MASK	5Bh	Mask value for the port PA direction register
DEF_PBPDR_MASK	EBh	Mask value for the port PB direction register
DEF_PCPDR_MASK	FFh	Mask value for the port PC direction register
DEF_PEPDR_MASK	FFh	Mask value for the port PE direction register
DEF_PHPDR_MASK	0Fh	Mask value for the port PH direction register
DEF_PJPDR_MASK	C0h	Mask value for the port PJ direction register
DEF_P0PDR	00h	Setting value for the port P0 direction register
DEF_P1PDR	00h	Setting value for the port P1 direction register
DEF_P2PDR	00h	Setting value for the port P2 direction register
DEF_P3PDR	00h	Setting value for the port P3 direction register
DEF_P4PDR	00h	Setting value for the port P4 direction register
DEF_P5PDR	00h	Setting value for the port P5 direction register
DEF_PAPDR	00h	Setting value for the port PA direction register
DEF_PBPDR	00h	Setting value for the port PB direction register
DEF_PCPDR	00h	Setting value for the port PC direction register
DEF_PEPDR	00h	Setting value for the port PE direction register
DEF_PHPDR	00h	Setting value for the port PH direction register
DEF_PJPDR	00h	Setting value for the port PJ direction register

Table 4.12 Constants when a 48-Pin Package is Used (PIN_SIZE=48)

Constant Name	Setting Value	Contents
DEF_P0PDR_MASK	00h	Mask value for the port P0 direction register
DEF_P1PDR_MASK	F0h	Mask value for the port P1 direction register
DEF_P2PDR_MASK	C0h	Mask value for the port P2 direction register
DEF_P3PDR_MASK	00h	Mask value for the port P3 direction register
DEF_P4PDR_MASK	5Fh	Mask value for the port P4 direction register
DEF_P5PDR_MASK	00h	Mask value for the port P5 direction register
DEF_PAPDR_MASK	5Ah	Mask value for the port PA direction register
DEF_PBPDR_MASK	ABh	Mask value for the port PB direction register
DEF_PCPDR_MASK	FFh	Mask value for the port PC direction register
DEF_PEPDR_MASK	9Fh	Mask value for the port PE direction register
DEF_PHPDR_MASK	0Fh	Mask value for the port PH direction register
DEF_PJPDR_MASK	C0h	Mask value for the port PJ direction register
DEF_P0PDR	28h	Setting value for the port P0 direction register
DEF_P1PDR	00h	Setting value for the port P1 direction register
DEF_P2PDR	00h	Setting value for the port P2 direction register
DEF_P3PDR	07h	Setting value for the port P3 direction register
DEF_P4PDR	00h	Setting value for the port P4 direction register
DEF_P5PDR	30h	Setting value for the port P5 direction register
DEF_PAPDR	01h	Setting value for the port PA direction register
DEF_PBPDR	40h	Setting value for the port PB direction register
DEF_PCPDR	00h	Setting value for the port PC direction register
DEF_PEPDR	60h	Setting value for the port PE direction register
DEF_PHPDR	00h	Setting value for the port PH direction register
DEF_PJPDR	00h	Setting value for the port PJ direction register

Table 4.13 Constants when a 40-Pin Package is Used (PIN_SIZE=40)

Constant Name	Setting Value	Contents
DEF_P0PDR_MASK	00h	Mask value for the port P0 direction register
DEF_P1PDR_MASK	F0h	Mask value for the port P1 direction register
DEF_P2PDR_MASK	C0h	Mask value for the port P2 direction register
DEF_P3PDR_MASK	04h	Mask value for the port P3 direction register
DEF_P4PDR_MASK	46h	Mask value for the port P4 direction register
DEF_P5PDR_MASK	00h	Mask value for the port P5 direction register
DEF_PAPDR_MASK	5Ah	Mask value for the port PA direction register
DEF_PBPDR_MASK	09h	Mask value for the port PB direction register
DEF_PCPDR_MASK	10h	Mask value for the port PC direction register
DEF_PEPDR_MASK	1Fh	Mask value for the port PE direction register
DEF_PHPDR_MASK	0Fh	Mask value for the port PH direction register
DEF_PJPDR_MASK	C0h	Mask value for the port PJ direction register
DEF_P0PDR	28h	Setting value for the port P0 direction register
DEF_P1PDR	00h	Setting value for the port P1 direction register
DEF_P2PDR	00h	Setting value for the port P2 direction register
DEF_P3PDR	03h	Setting value for the port P3 direction register
DEF_P4PDR	19h	Setting value for the port P4 direction register
DEF_P5PDR	30h	Setting value for the port P5 direction register
DEF_PAPDR	01h	Setting value for the port PA direction register
DEF_PBPDR	E2h	Setting value for the port PB direction register
DEF_PCPDR	EFh	Setting value for the port PC direction register
DEF_PEPDR	E0h	Setting value for the port PE direction register
DEF_PHPDR	00h	Setting value for the port PH direction register
DEF_PJPDR	00h	Setting value for the port PJ direction register

Table 4.14 Constants when a 36-Pin Package is Used (PIN_SIZE=36)

Constant Name	Setting Value	Contents
DEF_P0PDR_MASK	00h	Mask value for the port P0 direction register
DEF_P1PDR_MASK	F0h	Mask value for the port P1 direction register
DEF_P2PDR_MASK	80h	Mask value for the port P2 direction register
DEF_P3PDR_MASK	00h	Mask value for the port P3 direction register
DEF_P4PDR_MASK	06h	Mask value for the port P4 direction register
DEF_P5PDR_MASK	00h	Mask value for the port P5 direction register
DEF_PAPDR_MASK	58h	Mask value for the port PA direction register
DEF_PBPDR_MASK	09h	Mask value for the port PB direction register
DEF_PCPDR_MASK	10h	Mask value for the port PC direction register
DEF_PEPDR_MASK	1Fh	Mask value for the port PE direction register
DEF_PHPDR_MASK	0Fh	Mask value for the port PH direction register
DEF_PJPDR_MASK	C0h	Mask value for the port PJ direction register
DEF_P0PDR	28h	Setting value for the port P0 direction register
DEF_P1PDR	00h	Setting value for the port P1 direction register
DEF_P2PDR	40h	Setting value for the port P2 direction register
DEF_P3PDR	07h	Setting value for the port P3 direction register
DEF_P4PDR	59h	Setting value for the port P4 direction register
DEF_P5PDR	30h	Setting value for the port P5 direction register
DEF_PAPDR	03h	Setting value for the port PA direction register
DEF_PBPDR	E2h	Setting value for the port PB direction register
DEF_PCPDR	EFh	Setting value for the port PC direction register
DEF_PEPDR	E0h	Setting value for the port PE direction register
DEF_PHPDR	00h	Setting value for the port PH direction register
DEF_PJPDR	00h	Setting value for the port PJ direction register

4.8 Functions

Table 4.15 lists the Functions Used in the Sample Code.

Table 4.15 Functions Used in the Sample Code

Function Name	Outline
main	Main processing
R_INIT_StopModule	Stop processing for active peripheral functions after a reset
R_INIT_Port_Initialize	Nonexistent port initialization
R_INIT_Clock	Clock initialization
CGC_oscillation_main	Main clock oscillation setting
CGC_oscillation_HOCO	HOCO clock oscillation setting
CGC_oscillation_sub	Sub-clock oscillation setting
CGC_disable_subclk	Sub-clock stop setting
oscillation_subclk	Enabling sub-clock oscillation
init_RTC	RTC initialization
no_use_subclk_as_sysclk	Setting when the sub-clock is not used as the system clock
cmt0_countstart	CMT0 wait start setting (wait for sub-clock oscillation stabilization)
cmt0_endcheck	CMT0 wait (wait for sub-clock oscillation stabilization) completion check and initialization
R_DELAY	Inline function to specify the number of loops
R_DELAY_Us	Function to specify the execution time

4.9 Function Specifications

The following tables list the sample code function specifications.

main	
Outline	Main processing
Header	None
Declaration	void main(void)
Description	Calls the following functions: Stop processing for active peripheral functions after a reset, nonexistent port initialization, and clock initialization.
Arguments	None
Return Value	None

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.

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 in products.
Arguments	None
Return Value	None
Remarks	The number of pins in the sample code is set for the 64-pin package (PIN_SIZE=64). 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 value listed in Table 4.2 and Table 4.3 to the I/O select bits in the PDR registers, and set the output data store bits in the PODR registers to 0.

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 the main clock as the system clock without using the sub-clock and RTC.

CGC_oscillation_main	
Outline	Main clock oscillation setting
Header	r_init_clock.h
Declaration	void CGC_oscillation_main(void)
Description	Sets the main clock drive capability, sets the MOSCWTCR register, and enables main clock oscillation. Then waits for the main clock oscillation stabilization time.
Arguments	None
Return Value	None

CGC_oscillation_HOCO	
Outline	HOCO clock oscillation setting
Header	r_init_clock.h
Declaration	void CGC_oscillation_HOCO(void)
Description	Sets the HOCOWTCR register, and enables HOCO oscillation. Then waits for the HOCO clock oscillation stabilization time.
Arguments	None
Return Value	None

CGC_oscillation_sub	
Outline	Sub-clock oscillation setting
Header	r_init_clock.h
Declaration	void CGC_oscillation_sub(void)
Description	Configures the setting when the sub-clock is used as either the system clock or the RTC count source, or both.
Arguments	None
Return Value	None

CGC_disable_subclk	
Outline	Sub-clock stop setting
Header	r_init_clock.h
Declaration	void CGC_disable_subclk(void)
Description	Configure the sub-clock setting for when it is not used as the system clock or RTC count source.
Arguments	None
Return Value	None

oscillation_subclk	
Outline	Enabling the sub-clock oscillation
Header	None
Declaration	static void oscillation_subclk(void)
Description	Configures settings for sub-clock oscillation.
Arguments	None
Return Value	None

init_RTC	
Outline	RTC initialization
Header	None
Declaration	static void init_RTC (void)
Description	Initializes the RTC (setting for clock provision and RTC software reset).
Arguments	None
Return Value	None

no_use_subclk_as_sysclk	
Outline	Processing when the sub-clock is not used as the system clock
Header	None
Declaration	static void no_use_subclk_as_sysclk (void)
Description	Stops the sub-clock as the system clock when the sub-clock is used only as the RTC count source.
Arguments	None
Return Value	None

cmt0_countstart	
Outline	CMT0 wait start setting (wait for sub-clock oscillation stabilization)
Header	None
Declaration	static void cmt0_countstart(uint16_t cnt)
Description	When using the sub-clock oscillator, waits for the sub-clock oscillation stabilization time with CMT0. When starting to wait for the oscillation stabilization, CMT0 count starts.
Arguments	uint32_t cnt: Oscillation stabilization time cnt = oscillation stabilization time (ns) (1) ÷ FOR_CMT0_TIME (2)
Return Value	None
Remarks	1. The oscillation stabilization time varies depending on the crystal/ceramic resonator. Set the value referring to 4.3.2 Sub-Clock Oscillation Stabilization Time. 2. The value of FOR_CMT0_TIME is calculated with 4.56 MHz (max.) of LOCO. The actual wait time may differ depending on the LOCO frequency.

cmt0_endcheck	
Outline	CMT0 wait (wait for sub-clock oscillation stabilization) completion check and initialization
Header	None
Declaration	static void cmt0_endcheck(void)
Description	When using the sub-clock oscillator, checks whether the wait processing for the sub-clock oscillation stabilization is completed. If completed, initializes CMT0.
Arguments	None
Return Value	None

R_DELAY	
Outline	Inline function to specify the number of loops
Header	r_delay.h
Declaration	static void R_DELAY (unsigned long loop_cnt)
Description	Wait processing which performs loops for the specified number of times (a loop is fixed at five cycles).
Arguments	loop_cnt: The number of loops
Return Value	None

R_DELAY_Us	
Outline	Function to specify the execution time
Header	r_delay.h
Declaration	void R_DELAY_Us (unsigned long us, unsigned long khz)
Description	Calculates the number of loops based on the execution time (μs) and the system clock (ICLK) frequency, and calls the inline function to specify the number of loops.
Arguments	us: Execution time khz: System clock (ICLK) frequency when the function is called.
Return Value	None

4.10 Flowcharts

4.10.1 Main Processing

Figure 4.2 shows the Main Processing.

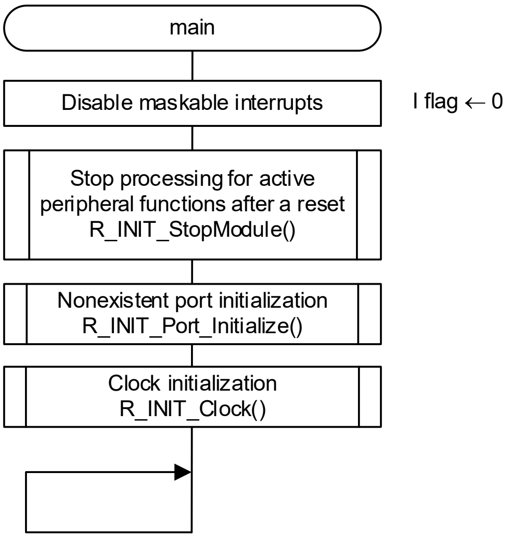
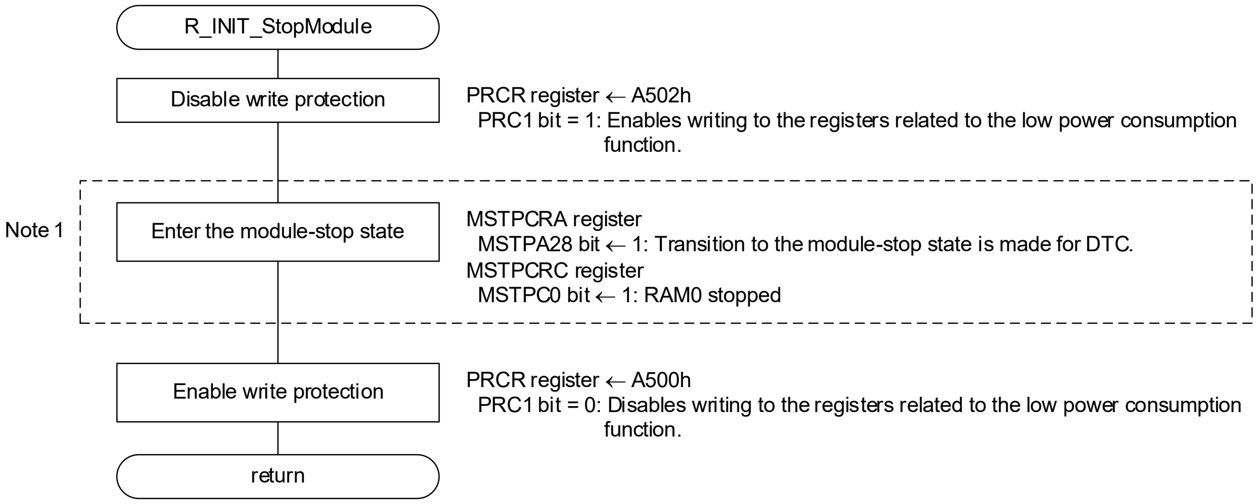


Figure 4.2 Main Processing

4.10.2 Stop Processing for Active Peripheral Functions after a Reset

Figure 4.3 shows the Stop Processing for Active Peripheral Functions after a Reset.



Note:

1. The module-stop state is canceled in the sample code. When entering the module-stop state for any peripheral functions, set the #define MSTP_STATE_"target module" constant to 1.

Figure 4.3 Stop Processing for Active Peripheral Functions after a Reset

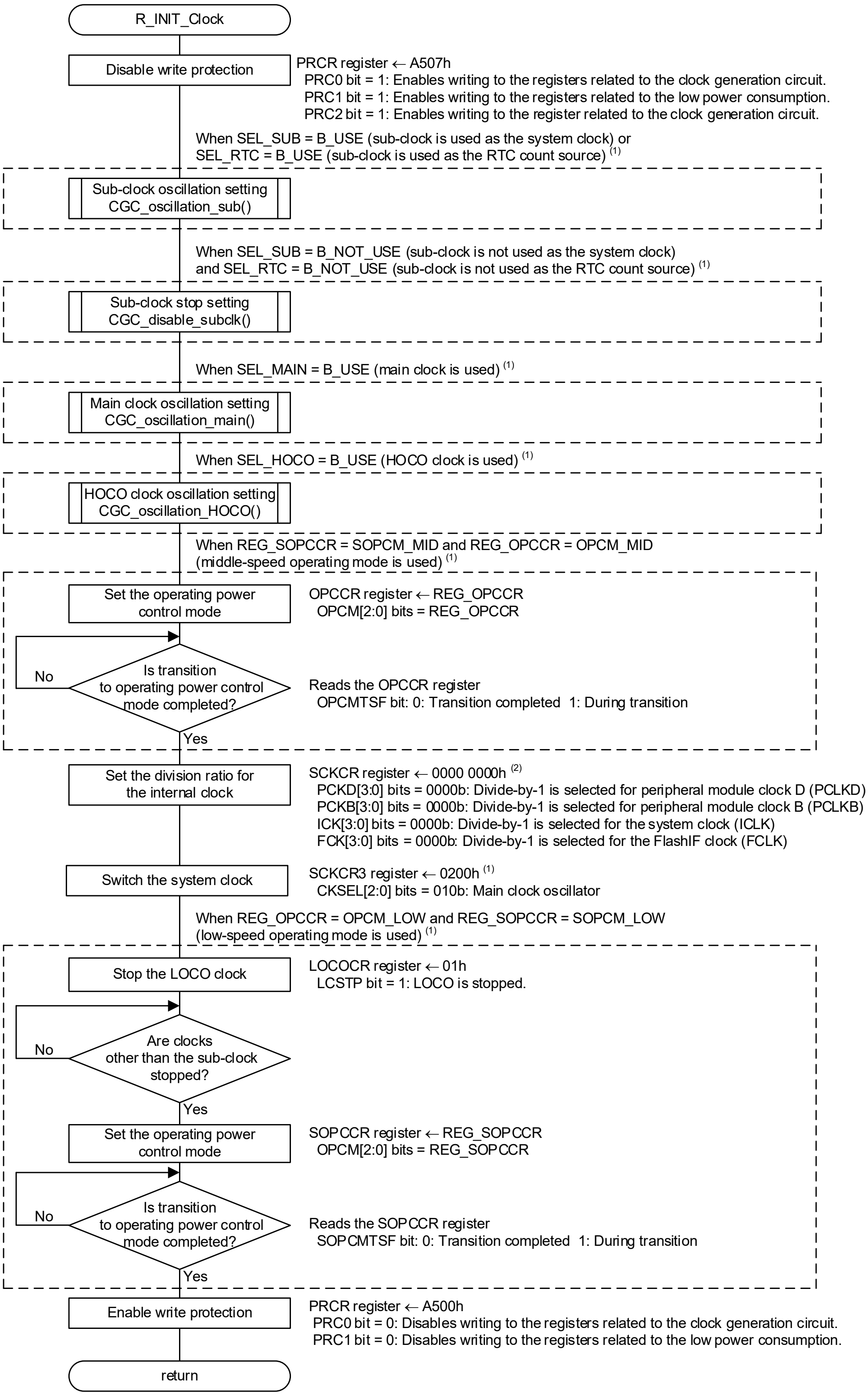
4.10.3 Nonexistent Port Initialization

Figure 4.4 shows the Nonexistent Port Initialization.

Figure 4.4 Nonexistent Port Initialization

4.10.4 Clock Initialization

Figure 4.5 shows the clock initialization.



Notes:
1. Change the constant value according to the user system.
2. The setting value varies depending on the system clock selected by the constant.

Figure 4.5 Clock Initialization

4.10.5 Main Clock Oscillation Setting

Figure 4.6 shows the Main Clock Oscillation Setting.

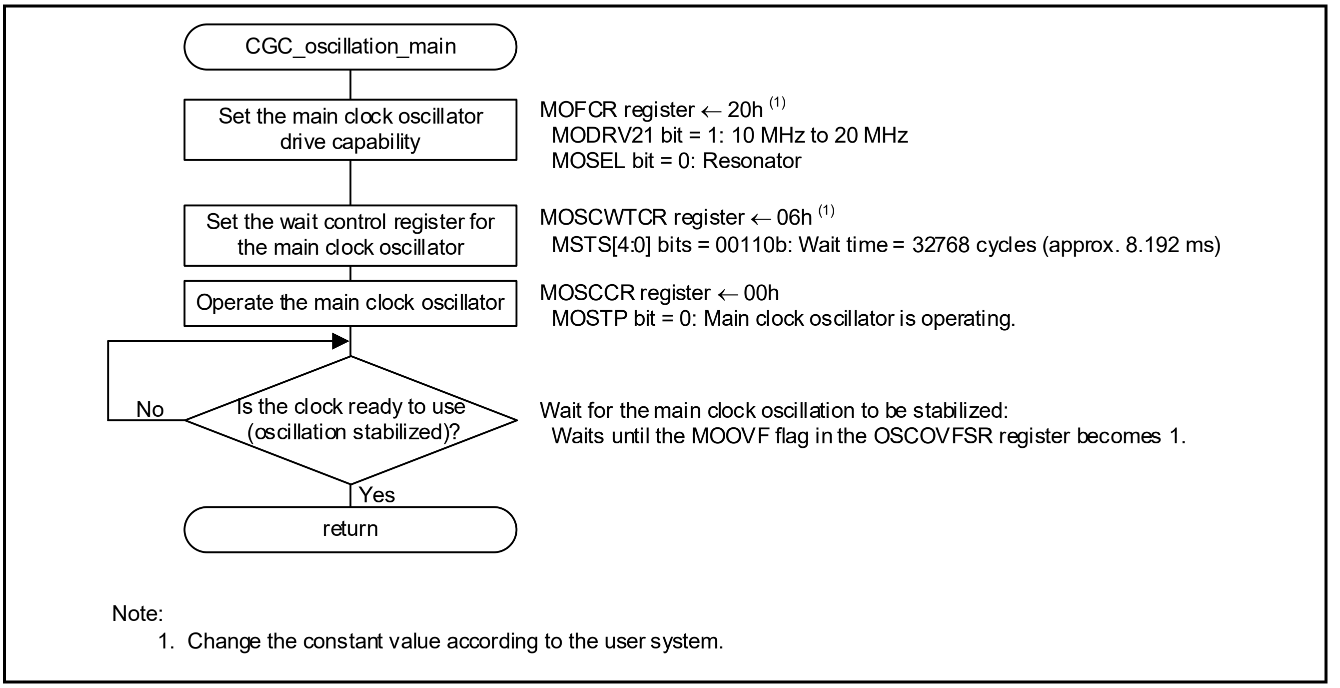


Figure 4.6 Main Clock Oscillation Setting

4.10.6 HOCO Clock Oscillation Setting

Figure 4.7 shows the HOCO Clock Oscillation Setting.

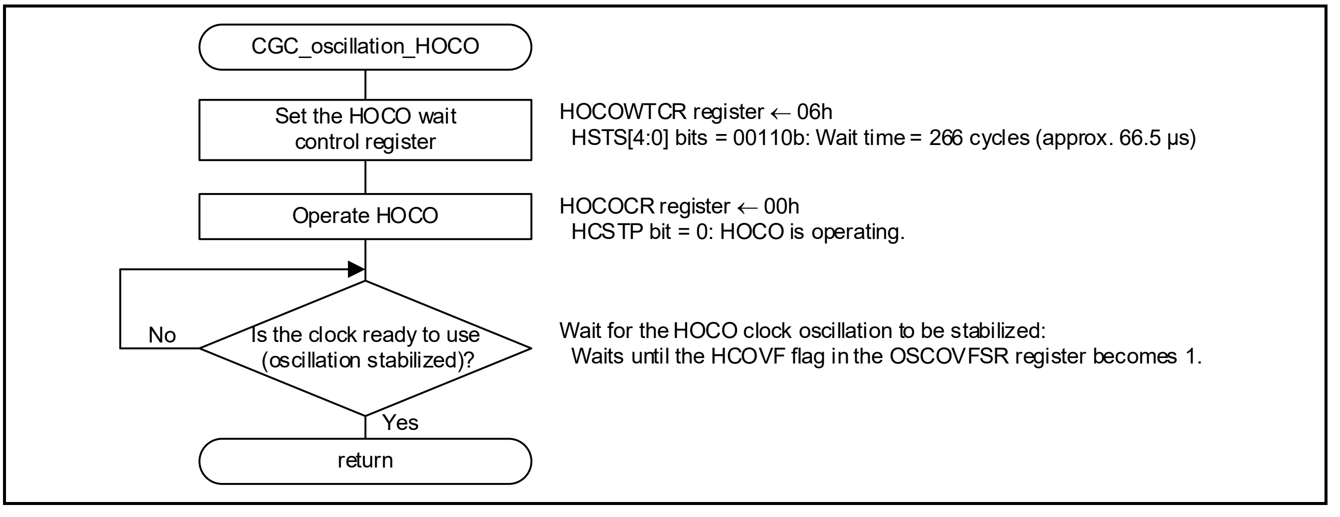


Figure 4.7 HOCO Clock Oscillation Setting

4.10.7 Sub-Clock Oscillation Setting

Figure 4.8 to Figure 4.11 show the sub-clock oscillation setting.

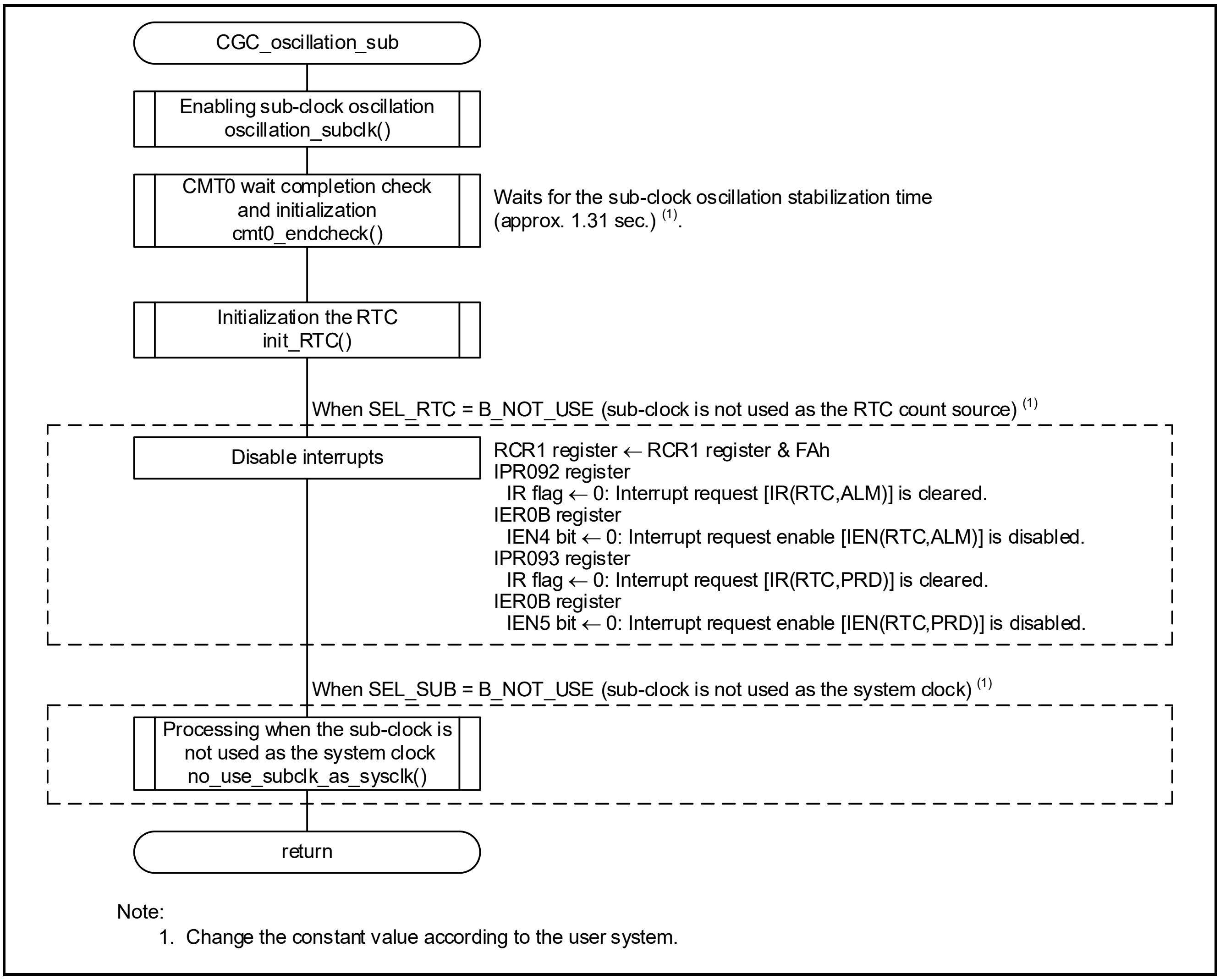
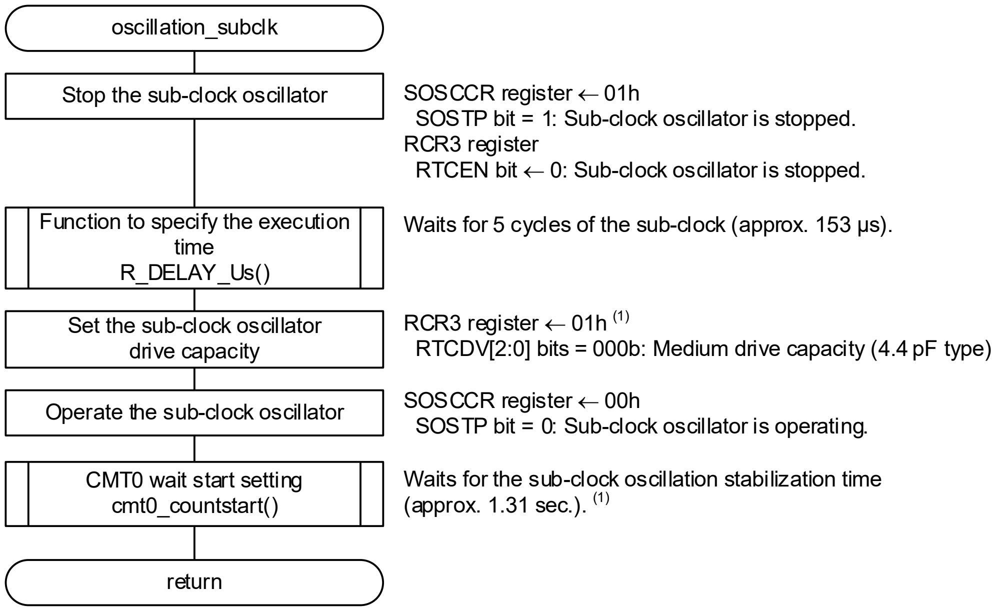


Figure 4.8 Sub-Clock Oscillation Setting



Note:

1. Change the constant value according to the user system.

Figure 4.9 Enabling Sub-Clock Oscillation

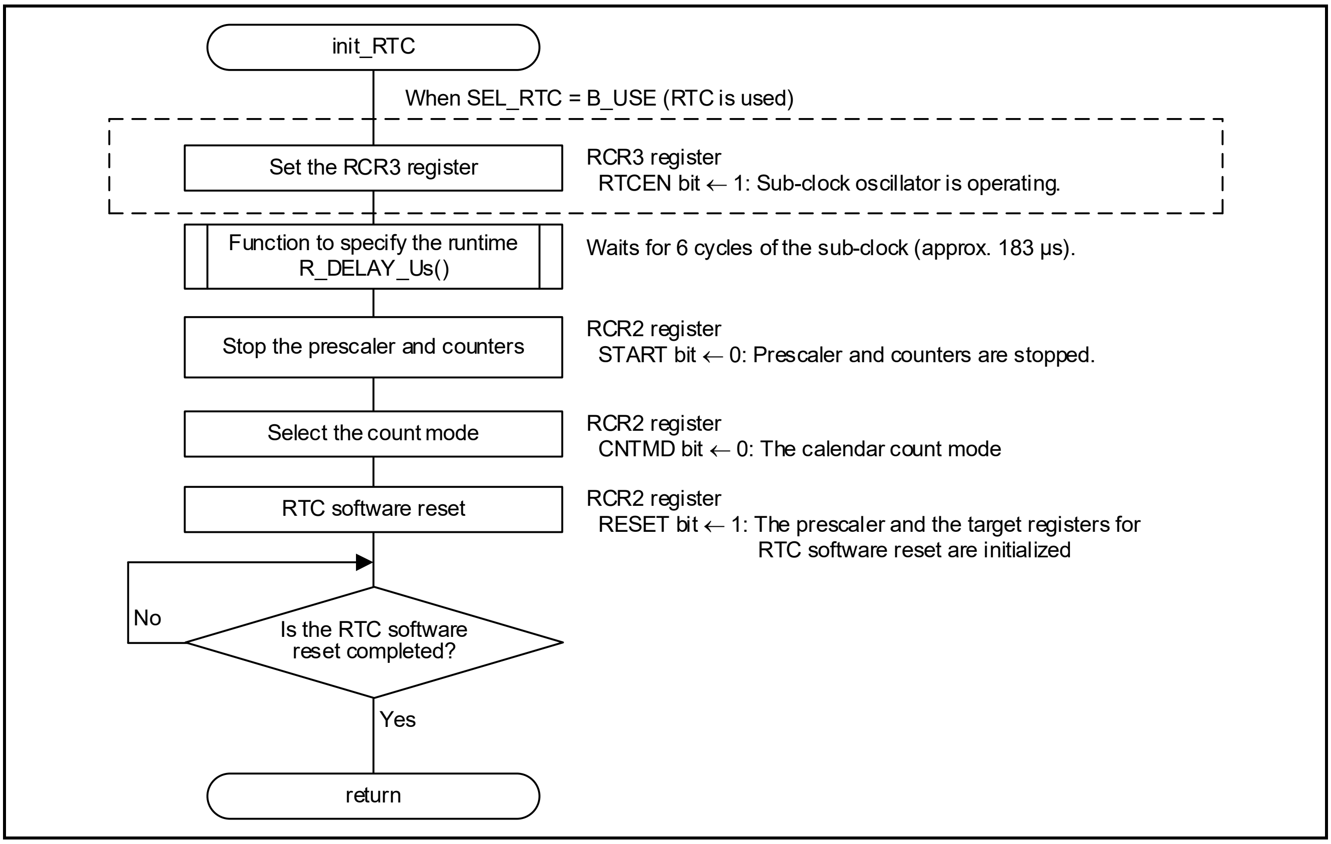


Figure 4.10 RTC Initialization

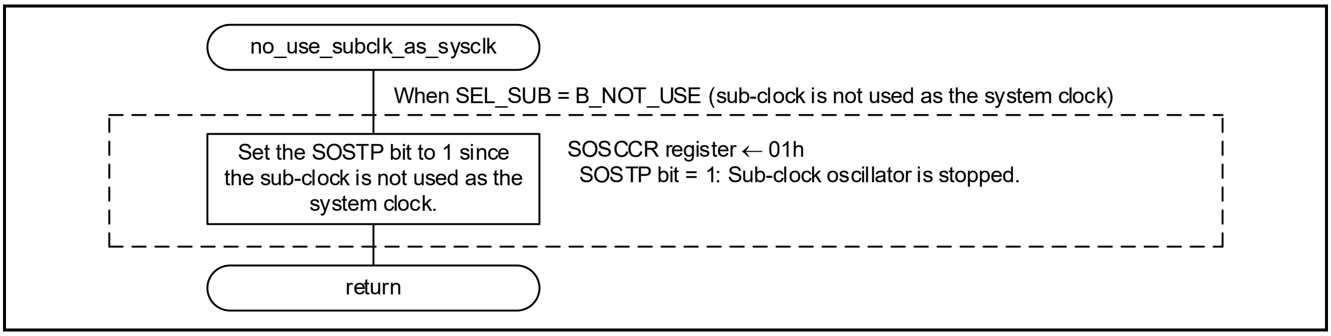


Figure 4.11 Processing when the Sub-Clock is not Used as the System Clock

4.10.8 Sub-Clock Stop Setting

shows the .

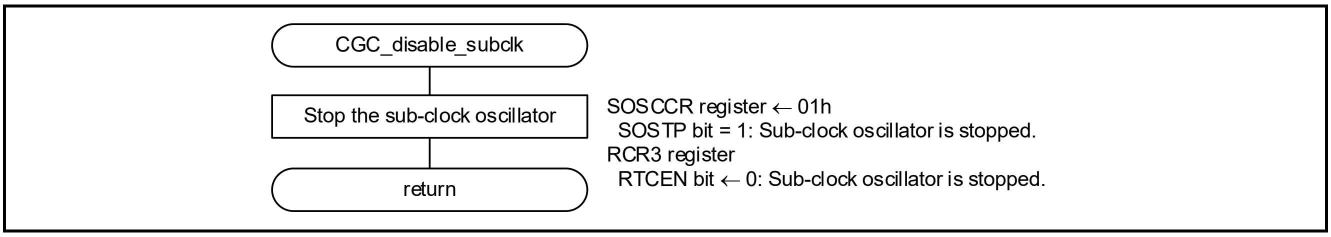


Figure 4.12 Sub-Clock Stop Setting

4.10.9 CMT0 Wait Start Setting, and CMT0 Wait Completion Check and Initialization

Figure 4.13 and Figure 4.14 show the CMT0 Wait Start Setting, and CMT0 Wait Completion Check and Initialization.

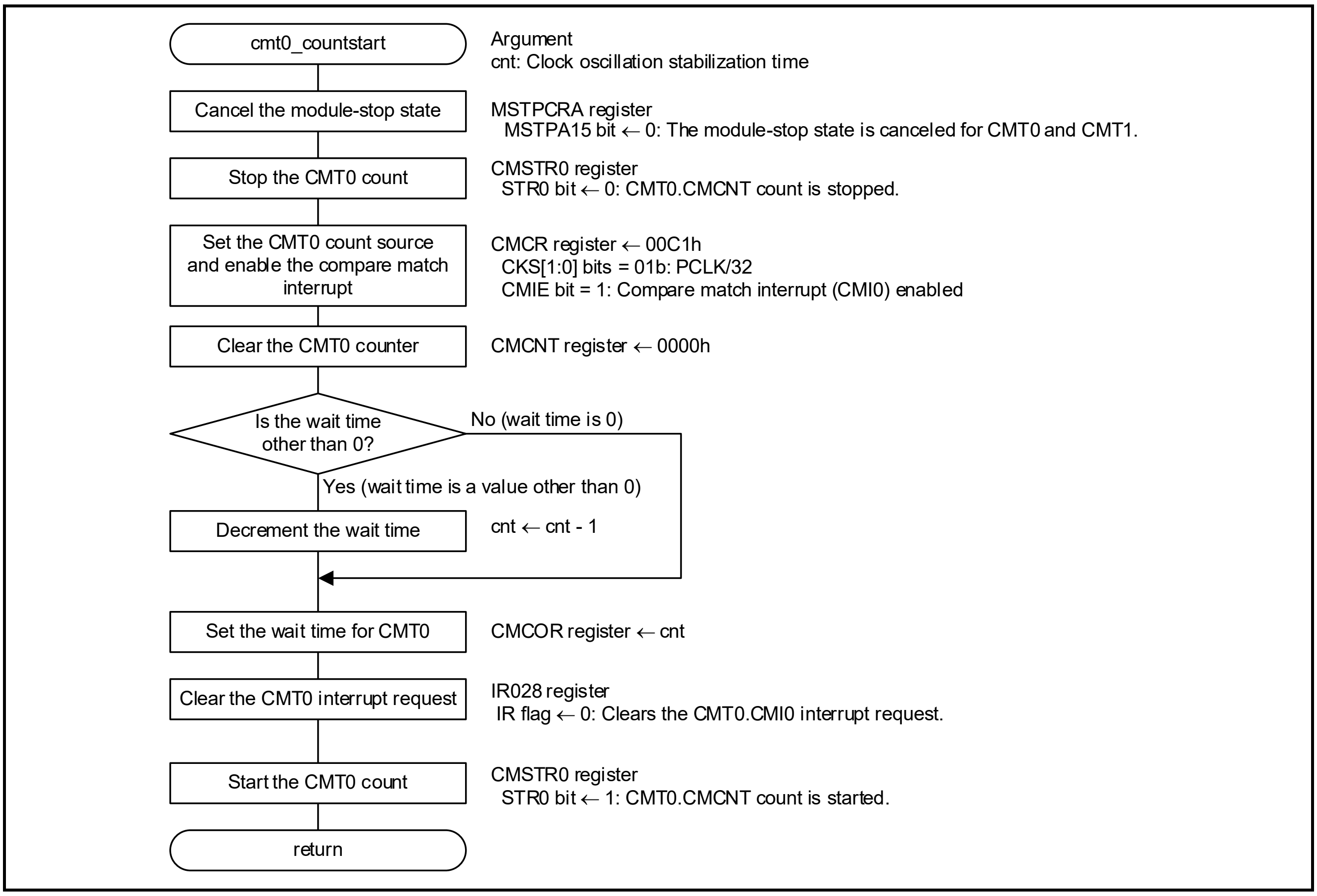
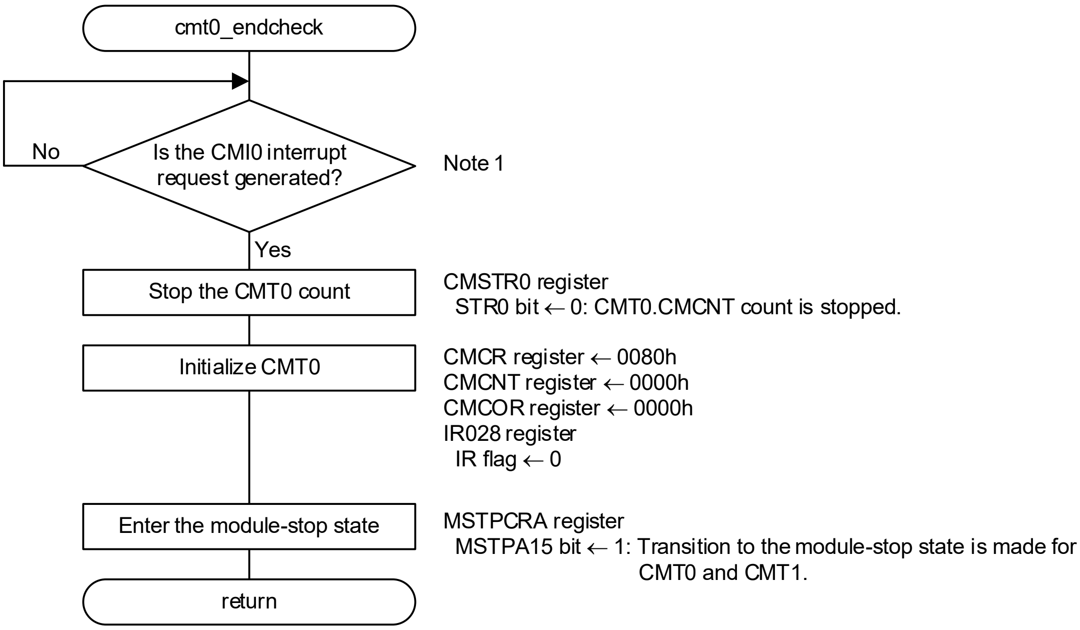


Figure 4.13 CMT0 Wait Start Setting



Note:

1. When the counter of the independent watchdog timer (IWDT) is operating, refresh the counter in this loop as required.

Figure 4.14 CMT0 Wait Completion Check and Initialization

5. Importing a Project

5.1 Importing a Project in the e^2 studio

When using the e^2 studio, follow the procedure shown below to import a project into the e^2 studio.

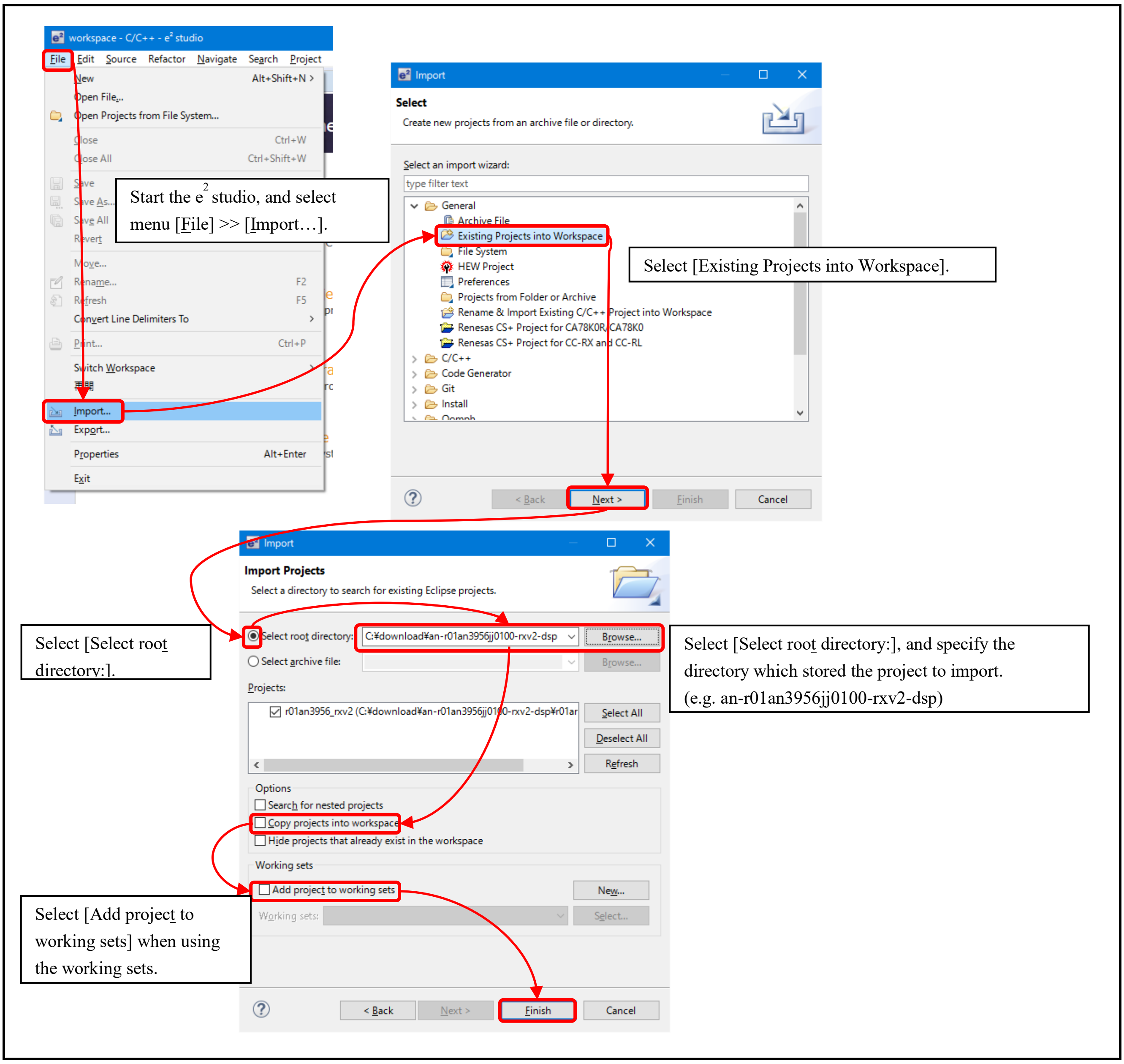


Figure 5.1 Importing a Project in the e^2 studio

5.2 Importing a Project in CS+

When using CS+, follow the procedure shown below to import a project into CS+.

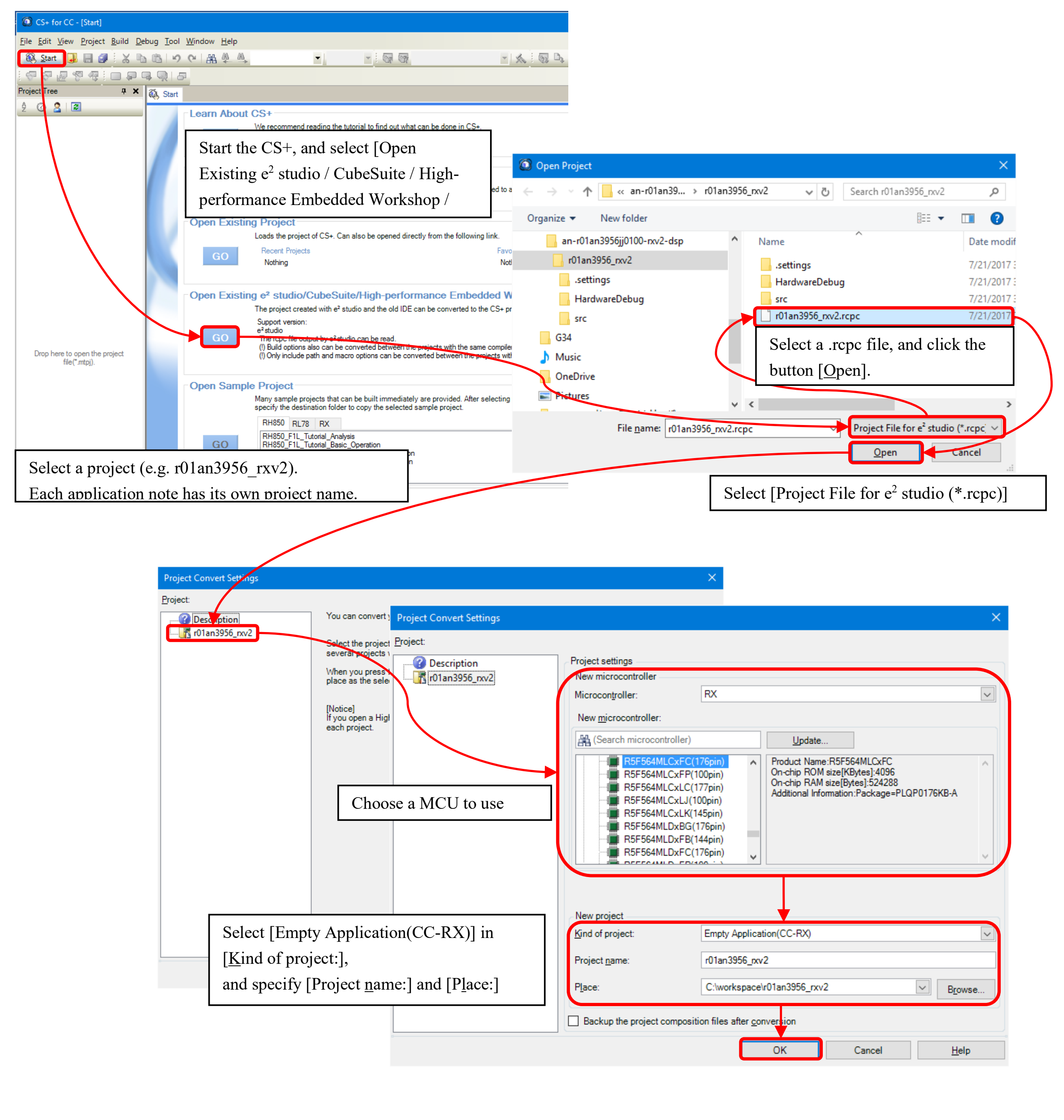


Figure 5.2 Importing a Project in CS+

6. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

7. Reference Documents

User's Manual: Hardware
RX110 Group User's Manual: Hardware (R01UH0421EJ)
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.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Feb.03.14	—	First edition issued
1.01	Mar.02.15	12,14	Added the bit setting for the standard CL in the sub-clock oscillator drive capacity control bit.
		26	Changed the setting value for the PRCR register.
		35	Updated the version of the reference document.
1.02	Sept.19.19	6	Table 2.1 Operation Confirmation Conditions, changed.
		11	4.4 Section Composition is added.
		12	Table 4.8 Constants Used in the Sample Code (1/2) (Users can change the constants listed in this table.), changed.
		13	Table 4.9 Constants Used in the Sample Code (2/2)(Users can change the constants listed in this table.), changed.
		19	Table 4.15 Functions Used in the Sample Code, changed.
		22	In the 4.8 Function Specifications, enable_RTC is deleted and init_RTC is added.
		26	Figure 4.5 Clock Initialization, changed.
		28	Figure 4.8 Sub-Clock Oscillation Setting, changed.
		30	Figure 4.10 RTC Initialization, changed.
		30	Figure 4.11 Processing when the Sub-Clock is not Used as the System Clock, changed.
		35	7.Reference Documents, changed.

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.

Corporate Headquarters

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

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/.