RX26T Group

Initial Settings Example

Introduction

This application note describes the settings of the RX26T products that must be made in accord with the conditions of usage specified in the header files that become applicable after a reset. These settings include clock settings, stopping any peripheral modules that are still operating after the reset, and the setting of bits corresponding to absent port pins.

Target Device

- 32-pin, 48-pin, 64-pin, 80-pin, or 100-pin products of the RX26T Group with ROM capacities in the range from 128 to 512 KB

Contents

1. Specifications

The sample code provides processing to stop peripheral modules that are operating after a reset, configure ports that are absent on the given device, and make clock settings. This application note is on the assumption that processing is to proceed when power is supplied (cold start).

1.1 Project Description

This application note also covers "r01an6567_rx26t".

"r01an6567_rx26t" is a project that is applicable to the Renesas Flexible Motor Control Kit for RX26T. This project contains files that were automatically generated by the e^2 studio. The settings of this project are adapted for the 100-pin device with a ROM capacity of 512 KB mounted on the Renesas Flexible Motor Control Kit (MCK) for RX26T. If another device is to be used, change the project settings as required. Refer to the following URL for details.
https://en-support.renesas.com/knowledgeBase/18696526

1.2 Stopping Peripheral Modules that Are Still Operating After a Reset

The module stop function is disabled for some peripheral modules after power is supplied. In order to stop these modules, the following processing is provided:

- Processing to stop the functioning of the DMAC, DTC, and RAM modules

Note that this processing is disabled in the sample code. To enable this processing, change the constants as required. Refer to Table 3.11 for details.

1.3 Settings of the Bits Corresponding to Absent Port Pins

The direction control bits corresponding to absent port pins in the given device must be set as described in 20.5.1, Initialization of the Port Direction Register (PDR), in the User’s Manual: Hardware. In the sample code for this application note, the initial values are adapted for a 100-pin device with a ROM capacity of 512 KB.

Change constants so that they are appropriate to the product to be used. Refer to section 3.2 and Tables 3.13 to 3.17 in section 3.5 for details.

1.4 Clock Settings

1.4.1 Overview

Clock settings are made in the following order:

(1) Setting the main clock
(2) Setting the HOCO clock
(3) Setting the PLL clock
(4) Switching the system clock

In this application note, the settings of clocks can be switched by changing constants defined in r_init_clock.h.

The PLL provides the system clock in the sample code. Select the clock to be used by changing the constants as required. Refer to 1.4.3 Clock Selection for details.

1.4.2 Clock Specifications in the Sample Code

Table 1.1 lists Clock Specifications Assumed in the Sample Code.

Table 1.1 Clock Specifications Assumed in the Sample Code

Clock	Oscillation Frequency	Oscillation Stabilization Time	Remarks
Main clock resonator	10 MHz	10 ms (Note 2)	Crystal
PLL clock	240 MHz (Main clock × 1/1 × 24)	— (Note 3)	—
HOCO clock	20 MHz (Note 1)	— (Note 3)	—

Notes: 1. Oscillation is stopped in the sample code.
2. The time required before the oscillation of the resonator becomes stable depends on the conditions, such as the wiring pattern and oscillation parameters, of the actual system. To determine the appropriate oscillation stabilization time, ask the manufacturer of the crystal or ceramic resonator to evaluate the user system.
3. Refer to "Electrical Characteristics" in the User's Manual: Hardware.

1.4.3 Clock Selection

In the sample code, the clock source of the system clock and whether to enable oscillation of each clock can be selected by changing constants defined in r_init_clock.h.

Table 1.2 shows Examples of the Clock Selection. The settings of example 1 are made in the sample code.

Table 1.2 Examples of the Clock Selection

No.		**1**	**2**	**3**	**4**
System clock		PLL	PLL	HOCO	Main clock
PLL clock		Enabled	Enabled	Disabled	Disabled
Main clock		Enabled	Disabled	Disabled	Enabled
HOCO clock		Disabled	Enabled	Enabled	Disabled
Constants (Note 1)	SEL_SYSCLK	CLK_PLL	CLK_PLL	CLK_HOCO	CLK_MAIN
	SEL_PLL	B_USE_PLL_MAIN	B_USE_PLL_HOCO	B_NOT_USE	B_NOT_USE
	SEL_MAIN	B_USE	B_NOT_USE	B_NOT_USE	B_USE
	SEL_HOCO	B_NOT_USE	B_USE	B_USE	B_NOT_USE

Note: 1. For details about the constant settings, refer to Table 3.9, Table 3.10, and Table 3.12.

1.5 Note on the Voltage Level Setting Register (VOLSR)

1.5.1 Note on Setting VOLSR

The effective bit of the voltage level setting register (VOLSR) in the RX26T must be set to the value corresponding to the power-supply voltage (VCC) when the RIIC is to be used.

In this application note, the initial value stated in the following table is used for the voltage level setting register (VOLSR) on the assumption that the Renesas Flexible Motor Control Kit for RX26T is to be used as shipped.

Table 1.3 Initial Value Set in the Voltage Level Setting Register (VOLSR)

Symbol	Bit Name	Function	Initial Value	Reason for Using the Initial Value
RICVLS	RIIC operating voltage setting	0: VCC ≥ 4.5 V 1: VCC < 4.5 V	1	For the Renesas Flexible Motor Control Kit for RX26T as shipped, the power-supply voltage is set to 3.3 V.

Set the voltage level setting register (VOLSR) in accord with the actual condition of usage. Operation with an incorrect setting cannot be guaranteed.

2. Conditions under Which Operation Was Verified

For the four sets of example settings for the sample code covered by this application note (Nos. 1 to 4 in Table 1.2), operation was verified under specific conditions. Table 2.1 lists the Conditions under Which Operation of r01an6567_rx26t Was Verified.

Table 2.1 Conditions under Which Operation of r01an6567_rx26t Was Verified

Item		Contents
MCU used		R5F526TFCDFP (RX26T Group)
Operating frequency	If the PLL is selected as the system clock and the main clock provides the PLL input (No. 1 in Table 1.2)	• Main clock: 10 MHz • PLL: 240 MHz (main clock × 1/1 × 24) • System clock (ICLK): 120 MHz (PLL × 1/2) • Peripheral module clock A (PCLKA): 120 MHz (PLL × 1/2) • Peripheral module clocks B (PCLKB) and D (PCLKD): 60 MHz (PLL × 1/4) • Peripheral module clock C (PCLKC): 120 MHz (PLL × 1/2) • FlashIF clock (FCLK): 60 MHz (PLL × 1/4)
	If the PLL is selected as the system clock and the HOCO provides the PLL input (No. 2 in Table 1.2)	• HOCO: 20 MHz • PLL: 240 MHz (HOCO × 1/1 × 12) • System clock (ICLK): 120 MHz (PLL × 1/2) • Peripheral module clock A (PCLKA): 120 MHz (PLL × 1/2) • Peripheral module clocks B (PCLKB) and D (PCLKD): 60 MHz (PLL × 1/4) • Peripheral module clock C (PCLKC): 120 MHz (PLL × 1/2) • FlashIF clock (FCLK): 60 MHz (PLL × 1/4)
	If the HOCO is selected as the system clock (No. 3 in Table 1.2)	• HOCO: 20 MHz • System clock (ICLK): 20 MHz (HOCO × 1/1) • Peripheral module clocks A to D (PCLKA to PCLKD): 20 MHz (HOCO ×1/1) • FlashIF clock (FCLK): 20MHz (HOCO × 1/1)
	If the main clock is selected as the system clock (No. 4 in Table 1.2)	• Main clock: 10 MHz • System clock (ICLK): 10 MHz (main clock × 1/1) • Peripheral module clocks A to D (PCLKA to PCLKD): 10 MHz (main clock × 1/1) • Flash-IF clock (FCLK): 10 MHz (main clock × 1/1)
Operating voltage		5.0 V
Integrated development environment		Renesas Electronics e^2 studio Version: 2024-07
C compiler		Renesas Electronics C/C++ Compiler Package for RX Family V3.06.00
		Compiler option The default settings in the integrated development environment are used.
Version of iodefine.h		V1.00a
Endian		Little endian, big endian
Operating mode		Single-chip mode
Processor mode		Supervisor mode
Sample code version		Version 1.10
Board used		Renesas Flexible Motor Control Kit for RX26T (Product No.: RTK0EMXE70S00020BJ)

3. Software

In the sample code, peripheral functions operating after a reset are stopped, settings of bits corresponding to absent port pins are made, and then clock settings are made.

3.1 Stopping Peripheral Modules that Are Still Operating After a Reset

This processing is to stop peripheral modules that are operating after a reset.

Table 3.1 is a list of the peripheral modules that do not automatically enter the module-stop state after a reset.

To place a module in the module-stop state after a reset, set the module-stop bit to 1. Placing modules in the module-stop state reduces power consumption.

In the sample code, the constants of the form "MSTP_STATE_target module name" in the file r_init_stop_module.h are set to "0 (MODULE_STOP_DISABLE)" so the target modules are not placed in the module stop state. In the case of modules with states that should be changed to "module-stop" in your system, set the corresponding constants to "1 (MODULE_STOP_ENABLE)".

Table 3.1 List of the Peripheral Modules That Do Not Automatically Enter the Module-Stop State After a Reset

Peripheral Module	Module Stop Setting Bit	Value After Reset	Setting When Not Using the Module
DMAC/DTC	MSTPCRA.MSTPA28 bit	0 (Release from the module-stop state)	1 (Transition to the module-stop state is made)
RAM	MSTPCRC.MSTPC0 bit		

3.2 Setting of Bits Corresponding to Absent Port Pins

3.2.1 Overview of Processing

Bits of the PDR registers corresponding to port pins that are absent in the given device should be set to 0 (input) or 1 (output). The R_INIT_Port_Initialize function called from main.c sets values in accord with the table corresponding to the setting for the number of pins (value of PIN_SIZE) in 20.5.1, Initialization of the Port Direction Register (PDR), in the User's Manual: Hardware. If data are to be written in byte units to a PDR register that includes bits corresponding to absent port pins after the call of the R_INIT_Port_Initialize function from main.c, also make sure that the direction control bits for those port pins are set as indicated by the table corresponding to the number of pins on the package in 20.5.1, Initialization of the Port Direction Register (PDR), in the User's Manual: Hardware. When writing to a PODR register in byte units, set the port output data storage bits for absent pins with the output setting to 0.

Table 3.2, Table 3.3, and Table 3.4 list the absent port pins.

Table 3.2 Absent Port Pins in the 100-pin and 80-pin Products

Port Symbol	100-pin products	Number	80-pin products	Number
PORT0	P02 to P07	6	P02 to P07	6
PORT1	P12 to P17	6	P12 to P17	6
PORT2	P25, P26	2	P23 to P26	4
PORT3	P34, P35	2	P32 to P35	4
PORT4	—	—	—	—
PORT5	P56, P57	2	P56, P57	2
PORT6	P66, P67	2	P61 to P63, P66, and P67	5
PORT7	P77	1	P77	1
PORT8	P83 to P87	5	P80 to P87	8
PORT9	P97	1	P97	1
PORTA	PA6, PA7	2	PA0 to PA2, PA4, PA6, and PA7	6
PORTB	—	—	PB7	1
PORTD	—	—	PD0, PD1	2
PORTE	PE6, PE7	2	PE0, PE1, and PE5 to PE7	5
PORTN	PN0 to PN5	6	PN0 to PN5	6

Table 3.3 Absent Port Pins in the 64-pin and 48-pin Products

Port Symbol	64-pin products	Number	48-pin products	Number
PORT0	P02 to P07	6	P01 to P07	7
PORT1	P10 and P12 to P17	7	P12 to P17	6
PORT2	P23 to P27	5	P22 to P27	6
PORT3	P30 to P35	6	P30 to P35	6
PORT4	—	—	P45 to P47	3
PORT5	P50, P51, P55, P56, P57	5	P50, P51, and P54 to P57	6
PORT6	P60 to P63, P66, and P67	6	P60, P61, and P63 to P67	7
PORT7	P77	1	P70, P77	2
PORT8	P80 to P87	8	P80 to P87	8
PORT9	P97	1	P96, P97	2
PORTA	PA0 to PA7	8	PA0 to PA7	8
PORTB	PB7	1	PB7	1
PORTD	PD0 to PD2	3	PD0 to PD2, PD4, and PD6	5
PORTE	PE0, PE1, and PE3 to PE7	7	PE0, PE1, and PE3 to PE7	7
PORTN	PN0 to PN5	6	PN0 to PN5 and PN7	7

Table 3.4 Absent Port Pins in the 32-pin Products

Port Symbol	32-pin products	Number
PORT0	P01 to P07	7
PORT1	P10 to P17	8
PORT2	P20 to P27	8
PORT3	P30 to P35, P37	7
PORT4	P43 to P47	5
PORT5	P50 to P52, P54 to P57	7
PORT6	P60 to P63, P66, P67	6
PORT7	P70, P77	2
PORT8	P80 to P87	8
PORT9	P90 to P97	8
PORTA	PA0 to PA7	8
PORTB	PB0, PB4 to PB7	5
PORTD	PD0 to PD2, PD4, PD6	5
PORTE	PE0, PE1, PE3 to PE7	7
PORTN	PN0 to PN5, PN7	7

3.2.2 Specifying the Number of Pins

In the sample code, use of the 100-pin product is set (PIN_SIZE = 100). The sample code also supports the 80-pin, 64-pin, 48-pin, and 32-pin products. To use a device other than the 100-pin products, set the number of pins of the device for PIN_SIZE in r_init_port_initialize.h.

3.3 Clock Settings

3.3.1 Clock Setting Procedure

Table 3.5 shows Clock Setting Procedure. It also describes the processing of each step and shows the default settings configured in the sample code. The sample code configured with the default settings enables the main clock and the PLL module, and disables the HOCO module.

Table 3.5 Clock Setting Procedure

Step	Processing	Description		Setting in Sample Code
1	Setting the main clock (Note 2)	If not to be used	No setting is required.	The main clock is used.
		If to be used	Set the driving ability of the main clock in the MOFCR register, set the waiting time until the output of the main clock is supplied as the internal clock in the MOSCWTCR register, and then make the main clock start oscillating. After that, waiting until oscillation becomes stable proceeds. (Note 1)	
2	Setting the HOCO clock (Note 2)	If not to be used	The HOCO clock is turned off.	The HOCO clock is not used.
		If to be used	Sets the HOCO frequency and starts HOCO clock oscillation. Then waiting until oscillation becomes stable proceeds. (Note 1)	
3	Setting the PLL clock (Note 2)	If not to be used	No setting is required.	The PLL clock is used.
		If to be used	Set the PLL input frequency division ratio and frequency multiplication factor, and then start the PLL clock oscillating. After that, waiting until oscillation becomes stable proceeds. (Note 1)	
4	Setting the clock frequency division ratios	Sets the clock division ratios.		• ICLK, PCLKA, and PCLKC: × 1/2 • PCLKB, PCLKD, and FCLK: × 1/4
5	Switching the system clock	Switches the system clock in accord with the user system.		Switches to the PLL clock.

Notes: 1. During the wait time, the system checks whether the corresponding bit in the oscillation stabilization flag register (OSCOVFSR) is 1.

2. Whether to use a clock can be selected by changing the setting of the corresponding constant in r_init_clock.h. Change the constant settings as required. Refer to section 3.7 for constants.

3.4 Section Configuration

Table 3.6 shows Information of the Section Changed in the Sample Code (r01an6567_rx26t).

For details about how to add, change, or delete sections, refer to the latest version of the RX Family CC-RX Compiler User's Manual.

Table 3.6 Information of the Section Changed in the Sample Code (r01an6567_rx26t)

Section Name	Type	Address	Description
End_of_RAM	Addition	0000 FFFCh	End address of the on-chip RAM

3.5 File Composition

Table 3.7 shows Files Used in the Sample Code. Files generated by the integrated development environment are not included in this table.

Table 3.7 Files Used in the Sample Code

File Name	Outline	Remarks
main.c	Main processing	
r_init_stop_module.c	Stopping the peripheral modules that are still operating after a reset	
r_init_stop_module.h	Header file for r_init_stop_module.c	
r_init_port_initialize.c	Initialization of the bits corresponding to absent port pins	
r_init_port_initialize.h	Header file for r_init_port_initialize.c	
r_init_clock.c	Initialization of the system clock	
r_init_clock.h	Header file for r_init_rom_cache.c	

3.6 Option-Setting Memory

Table 3.8 shows Status of the Option-Setting Memory Used in the Sample Code. Set a value that suites the user system as required.

Table 3.8 Status of the Option-Setting Memory Used in the Sample Code

Symbol	Address	Setting Value	Description
OFS0	0012 0068h to 0012 006Bh	FFFF FFFFh	The IWDT is stopped after a reset. The WDT is stopped after a reset.
OFS1	0012 006Ch to 0012 006Fh	FFFF FFFFh	The voltage monitor 0 reset is disabled after a reset. HOCO oscillation is disabled after a reset.
MDE	0012 0064h to 0012 0067h	FFFF FFFFh	Little endian, linear mode

3.7 Constants

Table 3.9 to Table 3.12 shows Constants Used in the Sample Code. Note that the constants in Table 3.12 cannot be changed by the user.

Table 3.13 to Table 3.17 show constants that may vary with the product package. The macro definitions are selected by #ifdef in accord with the PIN_SIZE value set in Table 3.11.

Table 3.9 Constants Used in the Sample Code (User Changeable) (1/4)

Constant Name	Initial Setting	Description
SEL_MAIN (Note 1)	B_USE	Main clock enable/disable selection: B_USE: Enable (enables oscillation of the main clock) B_NOT_USE: Disable (disables the main clock)
MAIN_CLOCK_Hz (Note 1)	10000000L	Oscillation frequency of the resonator for the main clock (Hz)
REG_MOFCR (Note 1)	20h	Setting of the driving ability of the main clock oscillator (setting value of the MOFCR register)
REG_MOSCWTCR (Note 1)	53h	Setting value of the main clock oscillator wait control register
SEL_PLL (Note 1)	B_USE_PLL_MAIN	PLL clock enable/disable selection: B_USE_PLL_MAIN: Used (with the main clock) B_USE_PLL_HOCO: Used (with the HOCO clock) B_NOT_USE: Not used (PLL clock stopped)
REG_PLLCR (Note 1, Note 2, Note 3)	1710h (HOCO as PLL clock source) 2F00h (Other than above)	PLL input frequency division ratio and frequency multiplication factor settings (setting value of the PLLCR register) 1710h: Clock source: HOCO × 1/1 × 12 2F00h: Clock source: Main clock × 1/1 × 24
REG_SCKCR (Note 1, Note 2, Note 3)	2101 1212h (PLL selected) 0000 0000h (HOCO selected) 0000 0000h (Other than above)	Internal clock frequency division ratio (setting value of the SCKCR register)
REG_VOLSR (Note 1, Note 3)	80h	Setting of the voltage level for VCC when RIIC is used (setting value of the VOLSR register)

Notes: 1. Change the constant value in r_init_clock.h in accord with the user system.
2. The value to be set depends on the clock source of the selected system clock. Refer to the setting value column in Table 3.9 for details.
3. When changing the setting of this register, follow the User's Manual: Hardware.

Table 3.10 Constants Used in the Sample Code (User Changeable) (2/4)

Constant Name	Initial Setting	Description
SEL_HOCO (Note 1)	B_NOT_USE	HOCO clock enable/disable selection: B_USE: Enable (Enables oscillation of the HOCO clock.) B_NOT_USE: Disable (Disables the HOCO clock.)
REG_HOCOCR2 (Note 1)	FREQ_20MHz	HOCO clock frequency selection: FREQ_16MHz: 16 MHz FREQ_18MHz: 18 MHz FREQ_20MHz: 20 MHz
SEL_SYSCLK (Note 1)	CLK_PLL	System clock source selection: CLK_PLL: PLL CLK_HOCO: HOCO CLK_MAIN: Main clock

Note: 1. Change the constant value in r_init_clock.h in accord with the user system.

Table 3.11 Constants Used in the Sample Code (User Changeable) (3/4)

Constant Name	Initial Setting	Description
MSTP_STATE_DMACDTC (Note 1)	MODULE_STOP_DISABLE	Selection of the module-stop state for DMAC and DTC: MODULE_STOP_DISABLE: Disables state transition. MODULE_STOP_ENABLE: Enables state transition.
MSTP_STATE_RAM (Note 1)	MODULE_STOP_DISABLE	Selection of the module-stop state for RAM: MODULE_STOP_DISABLE: Disables state transition. MODULE_STOP_ENABLE: Enables state transition.
PIN_SIZE (Note 2)	100	Number of pins for the product used

Notes: 1. Change the setting value in r_init_stop_module.h in accord with the user system.
2. Change the setting value in r_init_port_initialize.h in accord with the number of the pins of the product to be used. Changing the settings of the bits corresponding to the absent port pins in the given device is also required. Refer to section 3.2 for details.

Table 3.12 Constants Used in the Sample Code (Not User Changeable) (4/4)

Constant Name	Setting Value	Description
B_NOT_USE	0	Disable
B_USE	1	Enable
B_USE_PLL_MAIN	2	The PLL clock is used. (clock source: main clock)
B_USE_PLL_HOCO	3	The PLL clock is used. (clock source: HOCO)
FREQ_16MHz	00h	HOCO frequency: 16 MHz
FREQ_18MHz	01h	HOCO frequency: 18 MHz
FREQ_20MHz	02h	HOCO frequency: 20 MHz
CLK_PLL	0400h	Clock source: PLL
CLK_HOCO	0100h	Clock source: HOCO
CLK_MAIN	0200h	Clock source: Main clock
REG_SCKCR2	2011h	Peripheral clock (CANFD clock) frequency division ratio (value set in the SCKCR2 register)
MODULE_STOP_ENABLE	1	Transition to the module-stop state is made.
MODULE_STOP_DISABLE	0	Release from the module-stop state

Table 3.13 Constants for 100-Pin Products (PIN_SIZE = 100)

Constant Name	Setting Value	Description
DEF_P0PDR	00h	Value set in the port direction register for port P0
DEF_P1PDR	00h	Value set in the port direction register for port P1
DEF_P2PDR	00h	Value set in the port direction register for port P2
DEF_P3PDR	00h	Value set in the port direction register for port P3
DEF_P4PDR	00h	Value set in the port direction register for port P4
DEF_P5PDR	00h	Value set in the port direction register for port P5
DEF_P6PDR	00h	Value set in the port direction register for port P6
DEF_P7PDR	00h	Value set in the port direction register for port P7
DEF_P8PDR	00h	Value set in the port direction register for port P8
DEF_P9PDR	00h	Value set in the port direction register for port P9
DEF_PAPDR	00h	Value set in the port direction register for port PA
DEF_PBPDR	00h	Value set in the port direction register for port PB
DEF_PDPDR	00h	Value set in the port direction register for port PD
DEF_PEPDR	00h	Value set in the port direction register for port PE
DEF_PNPDR	00h	Value set in the port direction register for port PN

Table 3.14 Constants for 80-Pin Products (PIN_SIZE = 80)

Constant Name	Setting Value	Description
DEF_P0PDR	00h	Value set in the port direction register for port P0
DEF_P1PDR	00h	Value set in the port direction register for port P1
DEF_P2PDR	18h	Value set in the port direction register for port P2
DEF_P3PDR	0Ch	Value set in the port direction register for port P3
DEF_P4PDR	00h	Value set in the port direction register for port P4
DEF_P5PDR	00h	Value set in the port direction register for port P5
DEF_P6PDR	0Eh	Value set in the port direction register for port P6
DEF_P7PDR	00h	Value set in the port direction register for port P7
DEF_P8PDR	07h	Value set in the port direction register for port P8
DEF_P9PDR	00h	Value set in the port direction register for port P9
DEF_PAPDR	17h	Value set in the port direction register for port PA
DEF_PBPDR	80h	Value set in the port direction register for port PB
DEF_PDPDR	03h	Value set in the port direction register for port PD
DEF_PEPDR	23h	Value set in the port direction register for port PE
DEF_PNPDR	00h	Value set in the port direction register for port PN

Table 3.15 Constants for 64-Pin Products (PIN_SIZE = 64)

Constant Name	Setting Value	Description
DEF_P0PDR	00h	Value set in the port direction register for port P0
DEF_P1PDR	01h	Value set in the port direction register for port P1
DEF_P2PDR	98h	Value set in the port direction register for port P2
DEF_P3PDR	0Fh	Value set in the port direction register for port P3
DEF_P4PDR	00h	Value set in the port direction register for port P4
DEF_P5PDR	23h	Value set in the port direction register for port P5
DEF_P6PDR	0Fh	Value set in the port direction register for port P6
DEF_P7PDR	00h	Value set in the port direction register for port P7
DEF_P8PDR	07h	Value set in the port direction register for port P8
DEF_P9PDR	00h	Value set in the port direction register for port P9
DEF_PAPDR	3Fh	Value set in the port direction register for port PA
DEF_PBPDR	80h	Value set in the port direction register for port PB
DEF_PDPDR	07h	Value set in the port direction register for port PD
DEF_PEPDR	3Bh	Value set in the port direction register for port PE
DEF_PNPDR	00h	Value set in the port direction register for port PN

Table 3.16 Constants for 48-Pin Products (PIN_SIZE = 48)

Constant Name	Setting Value	Description
DEF_P0PDR	02h	Value set in the port direction register for port P0
DEF_P1PDR	00h	Value set in the port direction register for port P1
DEF_P2PDR	9Ch	Value set in the port direction register for port P2
DEF_P3PDR	0Fh	Value set in the port direction register for port P3
DEF_P4PDR	E0h	Value set in the port direction register for port P4
DEF_P5PDR	33h	Value set in the port direction register for port P5
DEF_P6PDR	3Bh	Value set in the port direction register for port P6
DEF_P7PDR	01h	Value set in the port direction register for port P7
DEF_P8PDR	07h	Value set in the port direction register for port P8
DEF_P9PDR	41h	Value set in the port direction register for port P9
DEF_PAPDR	3Fh	Value set in the port direction register for port PA
DEF_PBPDR	80h	Value set in the port direction register for port PB
DEF_PDPDR	57h	Value set in the port direction register for port PD
DEF_PEPDR	3Bh	Value set in the port direction register for port PE
DEF_PNPDR	00h	Value set in the port direction register for port PN

Table 3.17 Constants for 32-Pin Products (PIN_SIZE = 32)

Constant Name	Setting Value	Description
DEF_P0PDR	02h	Value set in the port direction register for port P0
DEF_P1PDR	03h	Value set in the port direction register for port P1
DEF_P2PDR	9Fh	Value set in the port direction register for port P2
DEF_P3PDR	8Fh	Value set in the port direction register for port P3
DEF_P4PDR	F8h	Value set in the port direction register for port P4
DEF_P5PDR	37h	Value set in the port direction register for port P5
DEF_P6PDR	0Fh	Value set in the port direction register for port P6
DEF_P7PDR	01h	Value set in the port direction register for port P7
DEF_P8PDR	07h	Value set in the port direction register for port P8
DEF_P9PDR	7Fh	Value set in the port direction register for port P9
DEF_PAPDR	3Fh	Value set in the port direction register for port PA
DEF_PBPDR	F1h	Value set in the port direction register for port PB
DEF_PDPDR	57h	Value set in the port direction register for port PD
DEF_PEPDR	3Bh	Value set in the port direction register for port PE
DEF_PNPDR	80h	Value set in the port direction register for port PN

3.8 Functions

Table 3.18 shows Functions.

Table 3.18 Functions

Function Name	Outline
main	Main Processing
R_INIT_StopModule	Stopping Peripheral Modules that Are Still Operating After a Reset
R_INIT_Port_Initialize	Initialization of the Bits Corresponding to Absent Port Pins
R_INIT_Clock	Clock Initialization
CGC_oscillation_main	Main Clock Oscillation Setting
CGC_oscillation_PLL	PLL Clock Oscillation Setting
CGC_oscillation_HOCO	HOCO Clock Oscillation Setting

3.9 Function Specifications

The following tables list the specifications of sample code functions.

main	
Outline	Main processing
Header	None
Declaration	void main(void)
Description	Calls the functions that stop peripheral modules that are still operating after a reset, initialize the bits corresponding to absent port pins, and initialize clocks.
Arguments	None
Return Value	None

R_INIT_StopModule	
Outline	Stopping the peripheral modules that are still operating after a reset
Header	r_init_stop_module.h
Declaration	void R_INIT_StopModule(void)
Description	Configures the settings for modules to make transitions to the module-stop state.
Arguments	None
Return Value	None
Remarks	Transitions to the module-stop state do not proceed in the sample code.

R_INIT_Port_Initialize	
Outline	Initialization of the bits corresponding to absent port pins
Header	r_init_port_initialize.h
Declaration	void R_INIT_Port_Initialize (void)
Description	Initializes port direction register bits that correspond to absent port pins.
Arguments	None
Return Value	None
Remarks	In the sample code, use of the 100-pin product is set (PIN_SIZE = 100). If data are written in byte units to the PDR register that include bits corresponding to absent port pins after the call of this function, make sure that the direction-control bits for the absent port pins are set as indicated in 20.5, "Initialization of the Port Direction Register (PDR)", in the User's Manual: Hardware. Also, set the output data storage bits corresponding to port pins set as outputs to 0.

R_INIT_Clock	
Outline	Clock initialization
Header	r_init_clock.h
Declaration	void R_INIT_Clock(void)
Description	Initializes clocks.
Arguments	None
Return Value	None
Remarks	In the sample code, the PLL is selected as the system clock.

CGC_oscillation_main	
Outline	Main Clock Oscillation Setting
Header	r_init_clock.h
Declaration	void CGC_oscillation_main (void)
Description	Sets the driving ability of the main clock and sets the MOSCWTCR register, and then starts oscillation of the main clock. After that, waiting until oscillation of the main clock becomes stable proceeds.
Arguments	None
Return Value	None

CGC_oscillation_PLL	
Outline	PLL Clock Oscillation Setting
Header	r_init_clock.h
Declaration	void CGC_oscillation_PLL (void)
Description	Sets the PLL input frequency division ratio and frequency multiplication factor, and then starts oscillation of the PLL clock. After that, waiting until oscillation of the PLL becomes stable proceeds.
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 HOCO frequency and then starts oscillation of the HOCO clock. After that, waiting until oscillation of the HOCO clock becomes stable proceeds.
Arguments	None
Return Value	None

3.10 Flowcharts

3.10.1 Main Processing

Figure 3.1 shows the flowchart of Main Processing.

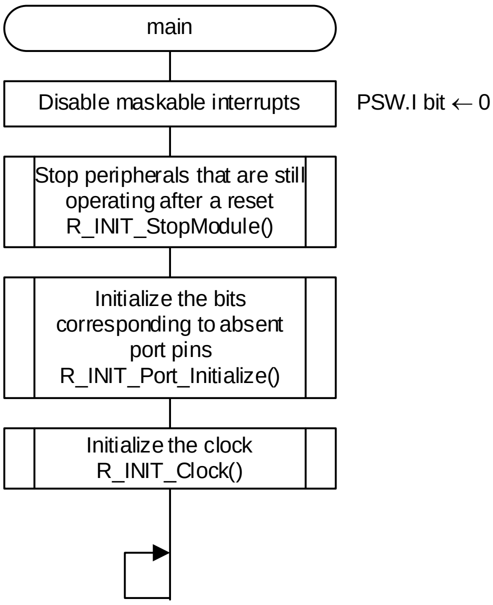
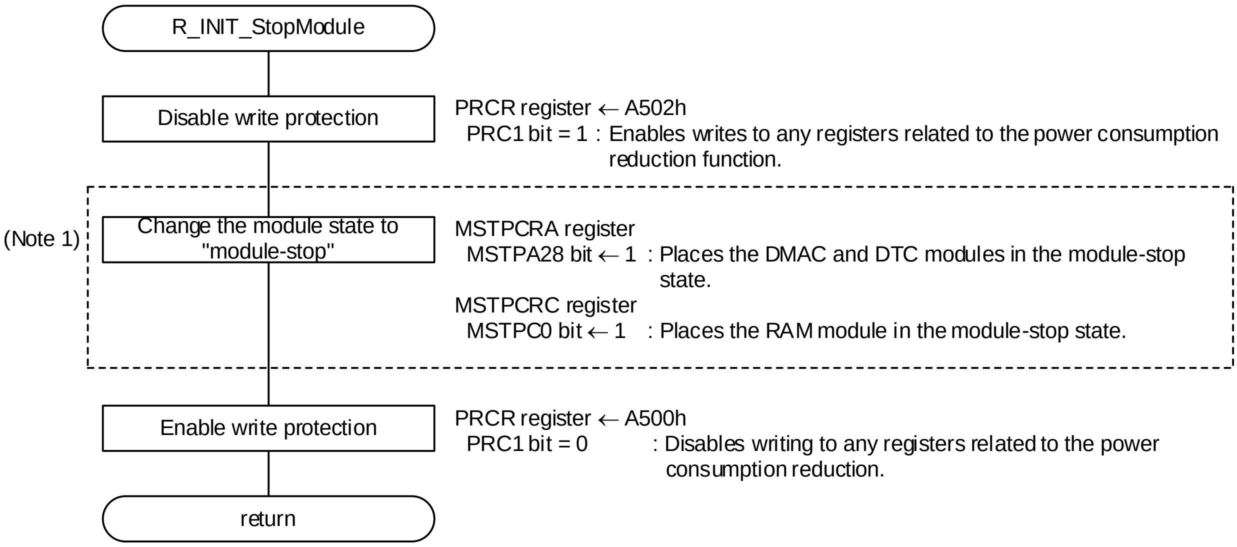


Figure 3.1 Main Processing

3.10.2 Stopping Peripheral Modules that Are Still Operating After a Reset

Figure 3.2 shows the flowchart of Stopping Peripheral Modules that Are Still Operating After a Reset.

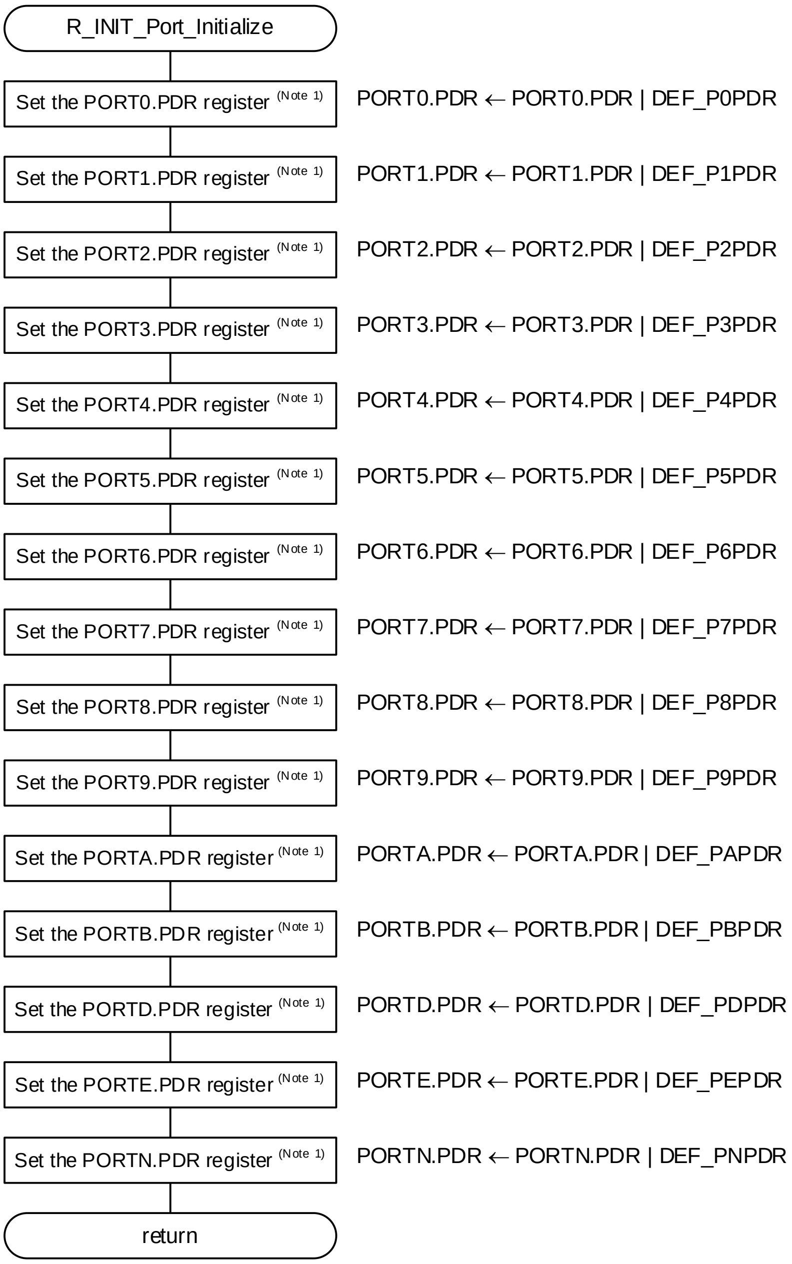


Note 1. In the sample code, none of these modules is placed in the module-stop state. To place these modules in the module-stop state, set the corresponding constants to MODULE_STOP_ENABLE (1).

Figure 3.2 Stopping Peripheral Modules that Are Still Operating After a Reset

3.10.3 Initialization of the Bits Corresponding to Absent Port Pins

Figure 3.3 shows the flowchart of Initialization of the Bits Corresponding to Absent Port Pins.

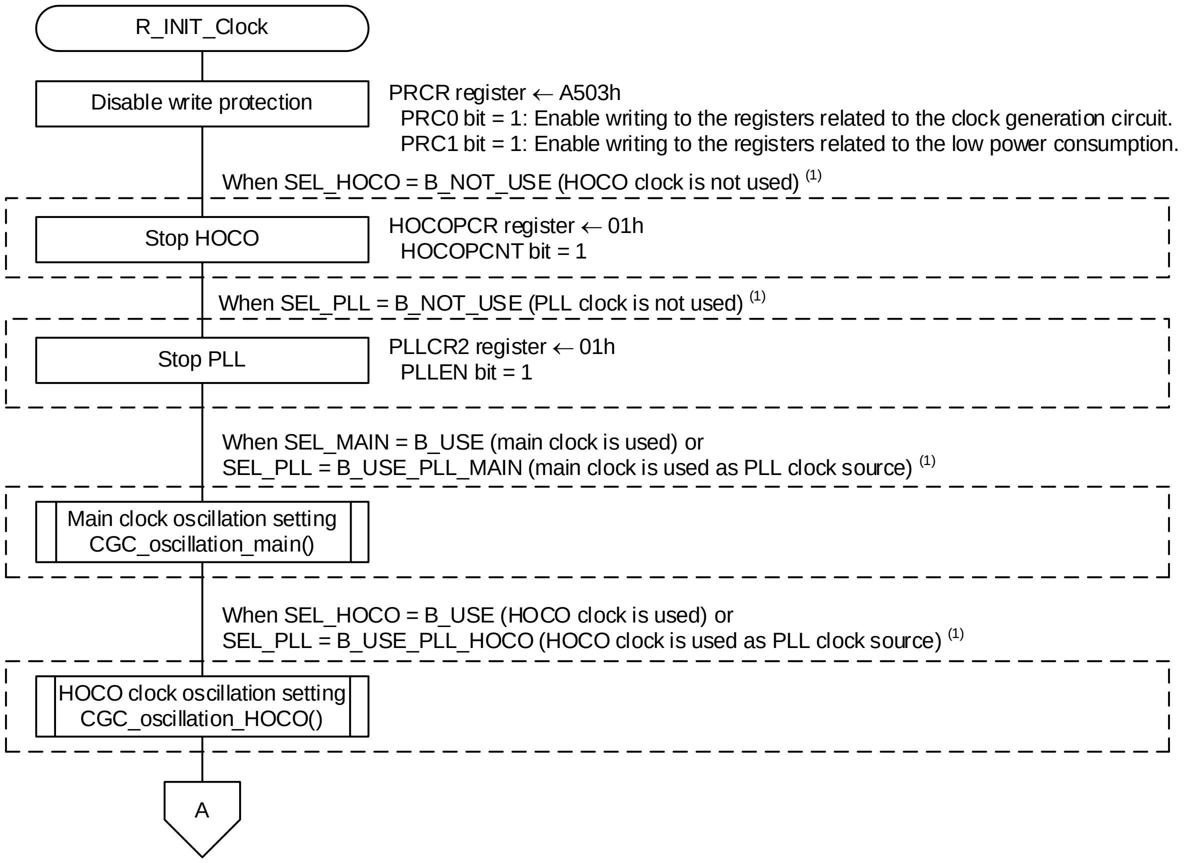


Note 1. Processing is skipped (omitted during compilation) if all pins in the register are present.

Figure 3.3 Initialization of the Bits Corresponding to Absent Port Pins

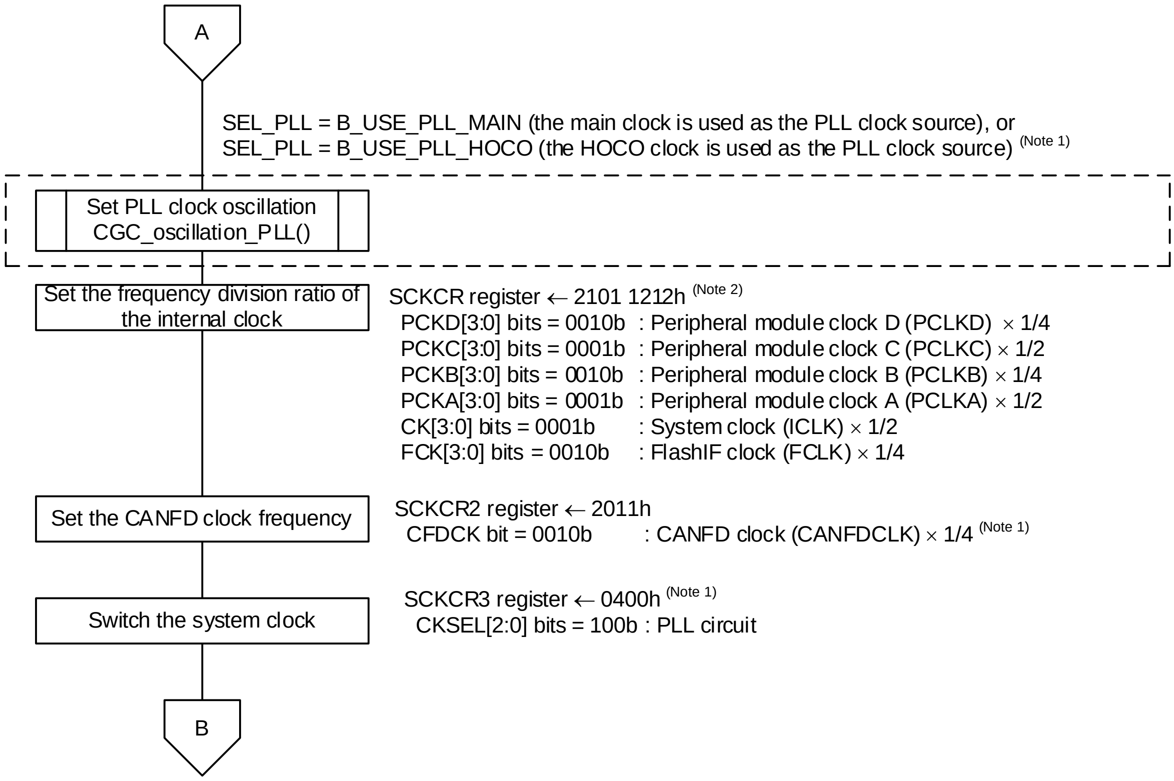
3.10.4 Clock Initialization

Figures 3.4 to 3.6 shows the clock setting flowchart.



Note 1. Change the SEL_MAIN, SEL_PLL, and SEL_HOCO constant settings in "r_init_clock.h" in accord with the user system. Refer to Tables 3.8 and 3.9 for details.

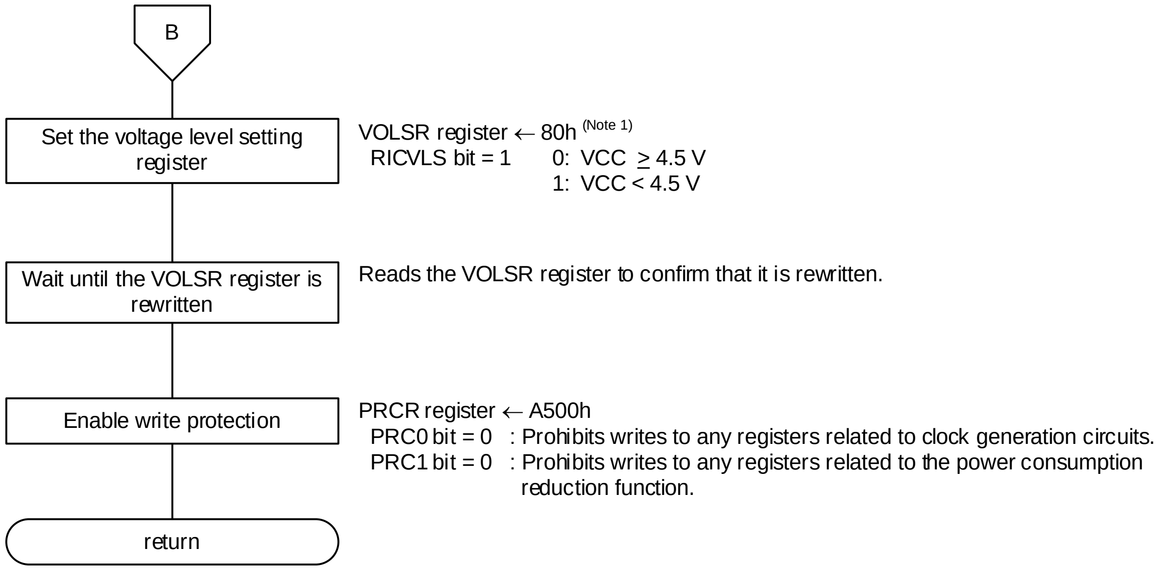
Figure 3.4 Clock Initialization (1/3)



Notes 1. Change the REG_SCKCR2 and REG_SCKCR3 constant settings in "r_init_stop_module.h" in accord with the user system.

2. The value to be set depends on the system clock selected by the REG_SCKCR constant in "r_init_clock.h".

Figure 3.5 Clock Initialization (2/3)

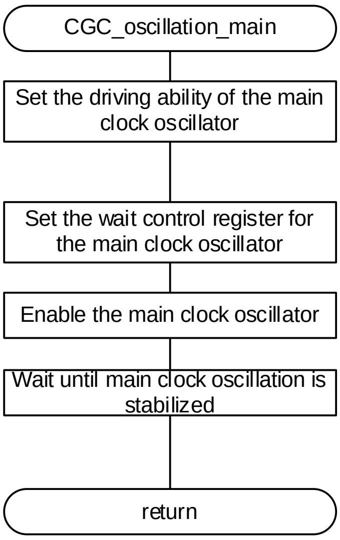


Note 1. Change the REG_VOLSR comstant settings in "r_init_clock.h" in accord with the user system. Refer toTable 3.8 for details.

Figure 3.6 Clock Initialization (3/3)

3.10.5 Main Clock Oscillation Setting

Figure 3.7 shows the flowchart of Main Clock Oscillation Setting.



MOFCR register ← 20h (Note 1)
MODRV2[1:0] bits = 10b : 8.1 to 16 MHz
MOSEL bit = 0 : Resonator

MOSCWTCR register ← 53h (Note 1)
MSTS[7:0] bits = 01010011b : When the max. frequency of fLOCO is 264 kHz, the wait time is approx.10 ms. (Note 2)

MOSCCR register ← 00h
MOSTP bit = 0 : Enables the main clock oscillator.

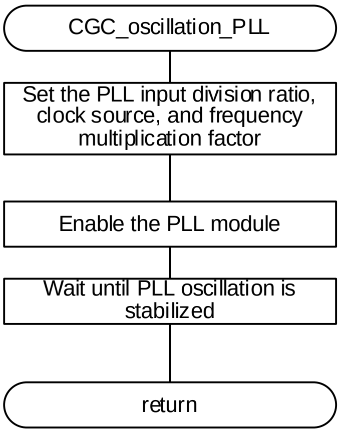
Reads the OSCOVFSR register.
MOOVF bit: 0: MOSTP = 1 (stopped) or waiting until oscillation is stabilized
1: Oscillation is stabilized and can be used as the system clock.

Notes 1. Change the REG_MOFCR and REG_MOSCWTCR constant settings in "r_init_clock.h" in accord with the user system.
2. In this application note, the initial value of the register is used.

Figure 3.7 Main Clock Oscillation Setting

3.10.6 PLL Clock Oscillation Setting

Figure 3.8 shows the flowchart of PLL Clock Oscillation Setting.



PLLCR register ← 2F00h (Note 1)
PLIDIV[1:0] bits = 00b : PLL input frequency division ratio = ×1/1
PLLSRCSEL bit = 0 : Main clock is selected as the PLL clock source
STC[5:0] bits = 101111b : Frequency multiplication factor = ×24

PLLCR2 register ← 00h
PLLEN bit = 0 : Enables the PLL module.

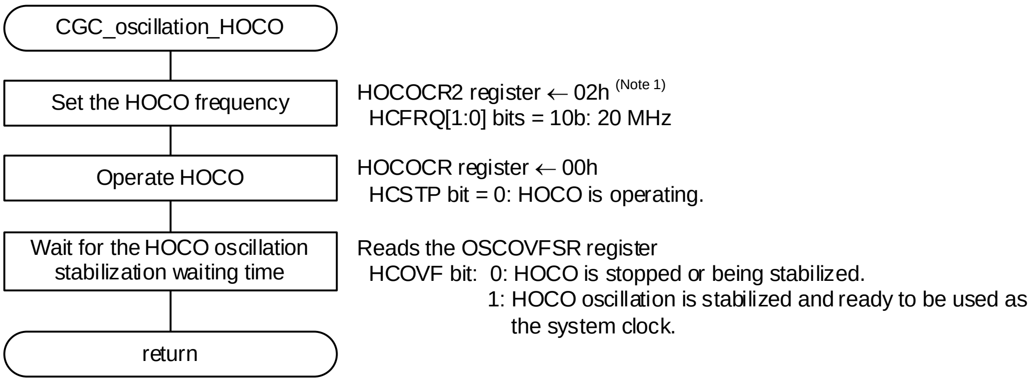
Reads the OSCOVFSR register
PLOVF bit: 0: PLL is stopped or waiting until oscillation is stabilized.
1: Oscillation is stabilized and can be used as the system clock.

Note 1. Change the REG_PLLCR constant settings in "r_init_clock.h" in accord with the user system. Refer to Table 3.8 for details.

Figure 3.8 PLL Clock Oscillation Setting

3.10.7 HOCO Clock Oscillation Setting

Figure 3.9 shows the flowchart of HOCO Clock Oscillation Setting.



Note 1. Change the REG_HOCOCR2 constant settings in "r_init_clock.h" in accord with the user system. Refer to Table 3.9 for details.

Figure 3.9 HOCO Clock Oscillation Setting

4. Importing a Project

The sample code is provided in the form of an e^2 studio project. This section describes the procedures for importing a project into the e^2 studio and CS+. After importing a project, confirm that the build settings and the debug settings are correct.

4.1 Importing a Project into the e^2 studio

If you use the project with e^2 studio, use the procedure described below to import the project to e^2 studio.

(The windows and dialogs shown in the following procedure may slightly differ from the actually displayed ones, depending on the version of e^2 studio you use.)

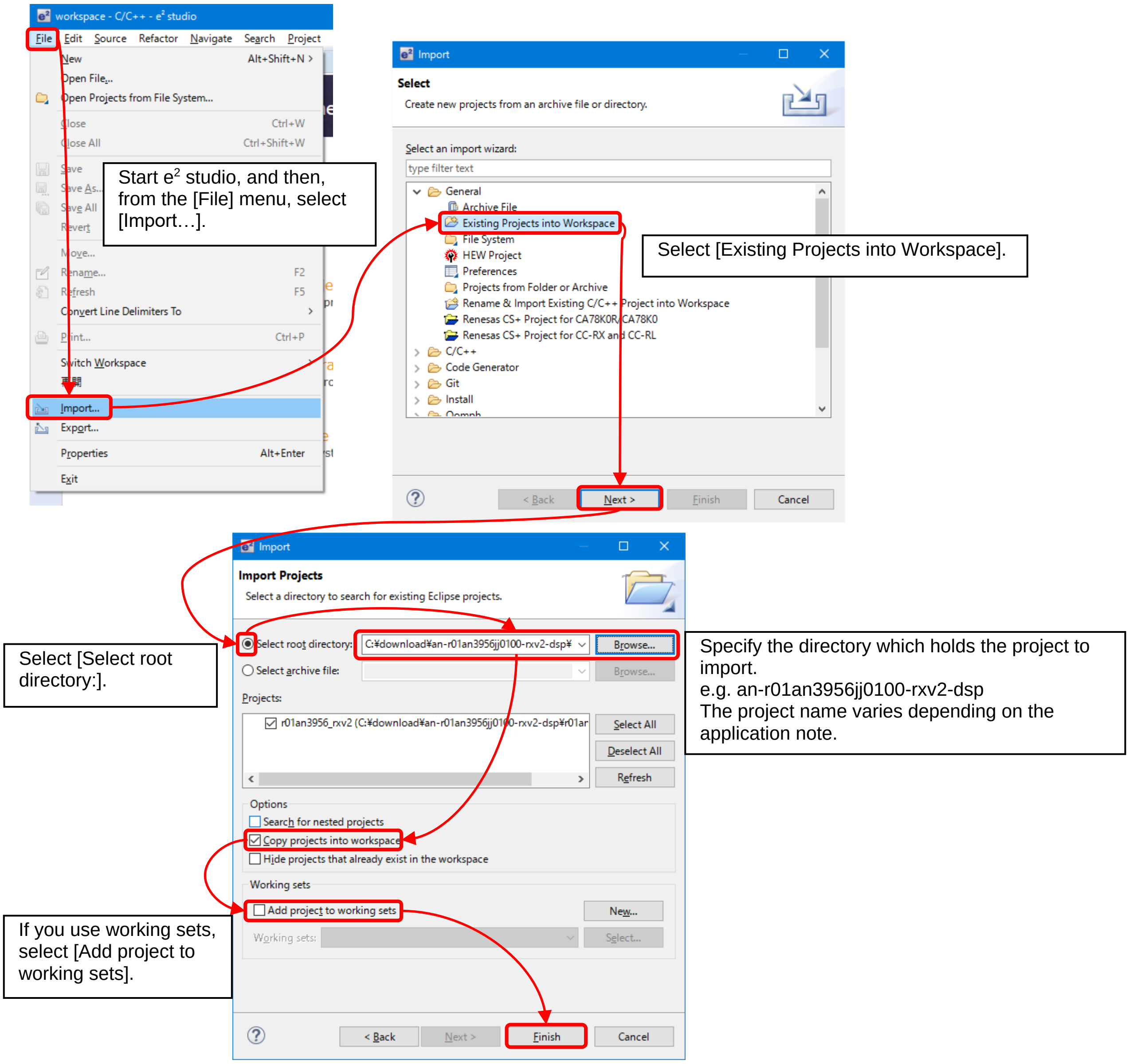


Figure 4.1 Importing a Project into the e^2 studio

4.2 Importing a Project into CS+

If you use the project with CS+, use the procedure described below to import the project to CS+.

(The windows and dialogs shown in the following procedure may slightly differ from the actually displayed ones, depending on the version of CS+ you use.)

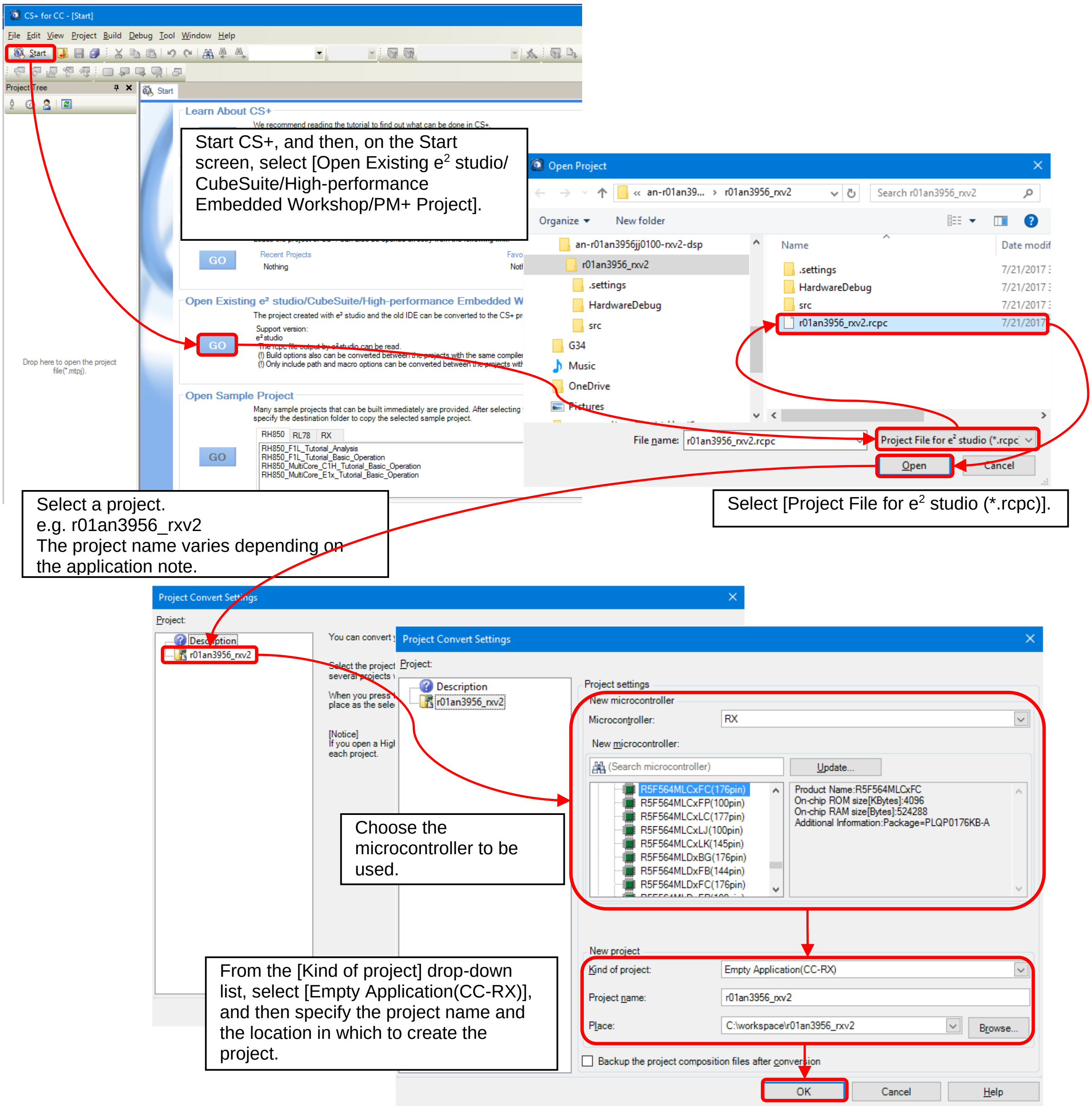


Figure 4.2 Importing a Project into CS+

5. Sample Code

Download the desired sample code from the Renesas Electronics website.

6. Reference Documents

User's Manual: Hardware

RX26T Group User's Manual: Hardware (R01UH0979)
(The latest versions of these documents are available at the Renesas Electronics website.)

Technical Update/Technical News

(The latest information is available at the Renesas Electronics website.)

User's Manual: Development environment

RX Family CC-RX Compiler User's Manual (R20UT3248)
(The latest versions of these documents are available at the Renesas Electronics website.)

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Apr.27.23	—	First edition issued
1.10	Mar.31.25	7	The versions in Table 2.1 were modified.
		10	Table 3.4 Absent Port Pins for 32-pin Products was added.
		17	Table 3.17 Constants for 32-Pin Products (PIN_SIZE = 32) was added.

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/.

Trademarks