Date: Jul. 15, 2025

RENESAS TECHNICAL UPDATE

TOYOSU FORESIA, 3-2-24, Toyosu, Koto-ku, Tokyo 135-0061, Japan
Renesas Electronics Corporation

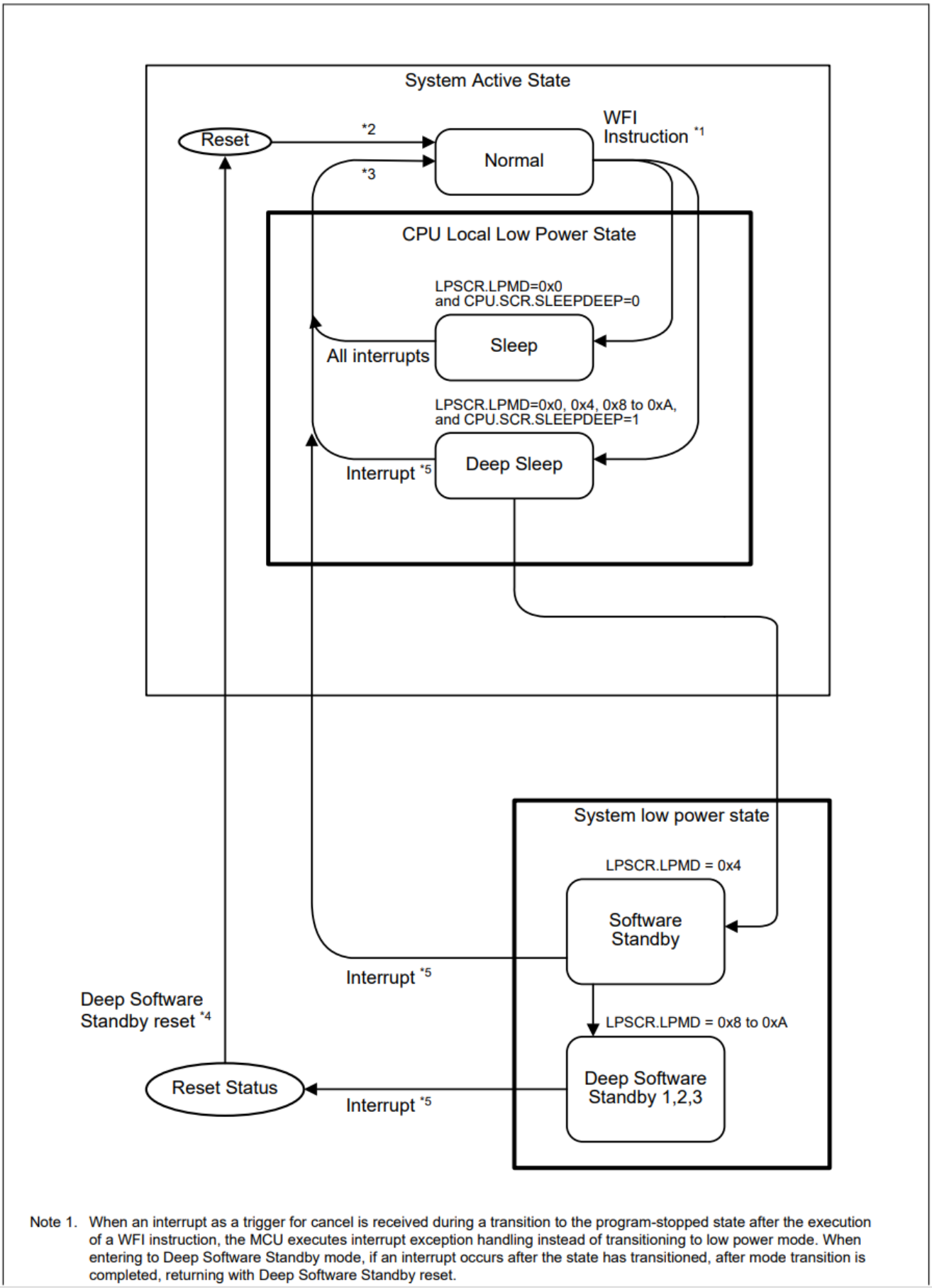
Product Category	MPU/MCU		Document No.	TN-RA*-A0152A/E	Rev.	1.00
Title	Note about the interruption during the transition to low power modes		Information Category	Technical Notification		
Applicable Product	RA8M1, RA8D1, RA8T1, RA8E1, and RA8E2 Group	Lot No. All	Reference Document	Refer table at the end of this document		

Corrections are made to the figures and tables in the user's manual hardware as shown in 1 and 2 below.
If the software meets the applicable condition listed in 4 below, the MCU may not be able to enter the intended low power mode.
If the unintended states described in the following 4 notes cannot be tolerated, use the following 5 workaround.

Contents

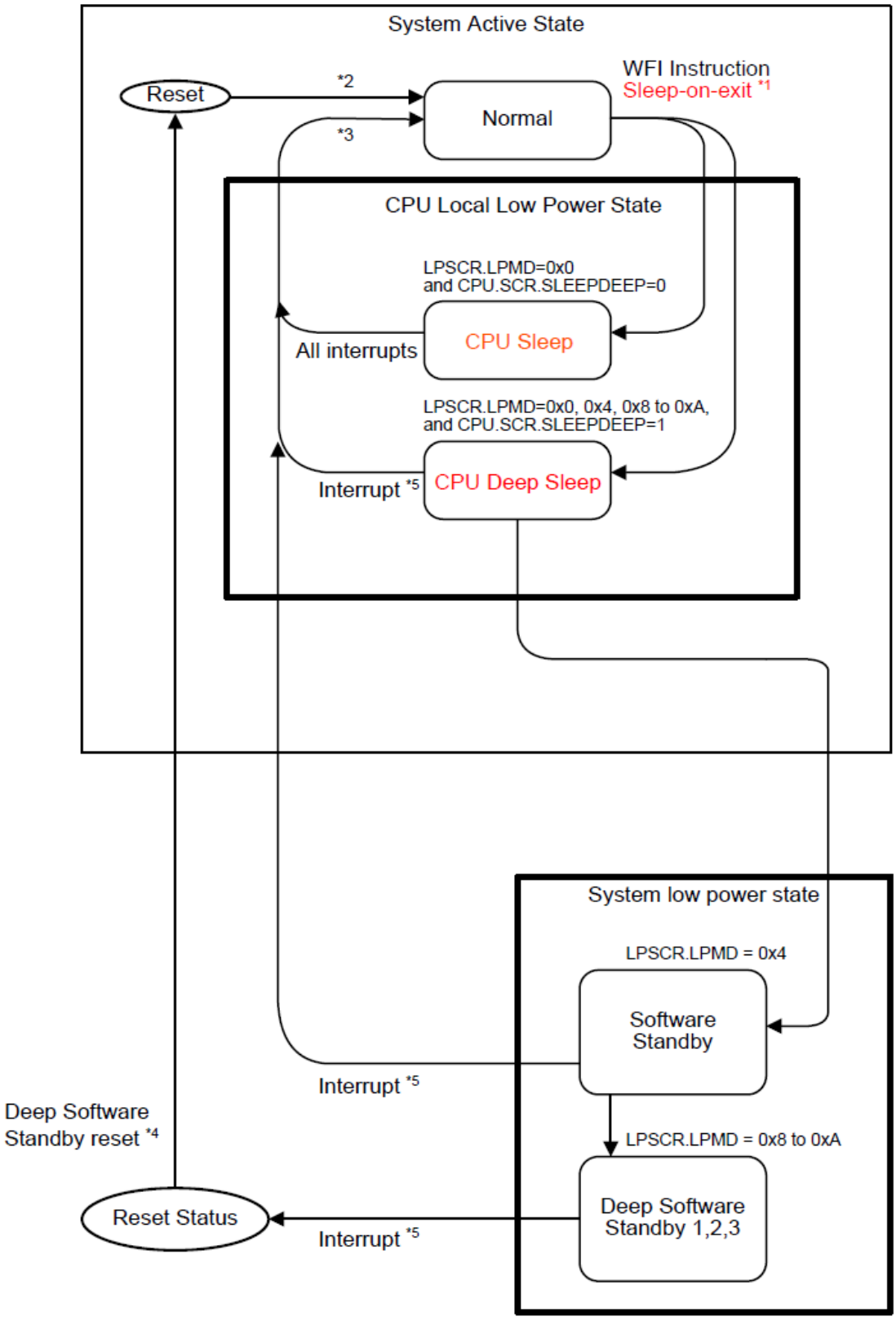
1. Correction of the Figure10.1 “Mode transitions”

Before correction



Note 1. When an interrupt as a trigger for cancel is received during a transition to the program-stopped state after the execution of a WFI instruction, the MCU executes interrupt exception handling instead of transitioning to low power mode. When entering to Deep Software Standby mode, if an interrupt occurs after the state has transitioned, after mode transition is completed, returning with Deep Software Standby reset.

After correction



Note 1. When an interrupt that acts as a trigger for cancellation of CPU Sleep mode state is received during a transition to the CPU Sleep mode state or CPU Deep Sleep mode state, the MCU does not transit to CPU Sleep mode state or CPU Deep Sleep mode state but goes back to Normal mode state.

2. Correction of the table “Operating conditions of each low power mode”

1) Tables 10.2

Before correction

Parameter	CPU Sleep mode	CPU Deep Sleep mode
Transition condition	WFI instruction after set CPU0.SCR. SLEEPDEEP = 0	WFI instruction after set CPU0.SCR. SLEEPDEEP = 1
State after cancellation by an interrupt	Program execution state (interrupt processing)	Program execution state (interrupt processing)

After correction

Parameter	CPU Sleep mode	CPU Deep Sleep mode
Transition condition	When [Condition 1] or [Condition 2] while CPU.SCR.SLEEPDEEP=0 [Condition 1] ・WFI instruction ・A valid interrupt request(*1) cannot be accepted to CPU. (including a transition from the time WFI instruction is executed to the time the transition to CPU Sleep mode is completed) [Condition 2] ・CPU.SCR.SLEEPONEXIT=1 ・Complete execution of all exception handlers ・A valid interrupt request(*1) cannot be accepted to CPU. (including a transition from the time WFI instruction is executed to the time the transition to CPU Sleep mode is completed)	When [Condition 1] or [Condition 2] while CPU.SCR.SLEEPDEEP=0 [Condition 1] ・WFI instruction ・A valid interrupt request(*1) cannot be accepted to CPU. (including a transition from the time WFI instruction is executed to the time the transition to CPU Deep Sleep mode is completed) [Condition 2] ・CPU.SCR.SLEEPONEXIT=1 ・Complete execution of all exception handlers ・A valid interrupt request(*1) cannot be accepted to CPU. (including a transition from the time WFI instruction is executed to the time the transition to CPU Deep Sleep mode is completed)
State after cancellation by an interrupt	Program execution state	Program execution state

(*1) Valid interrupt requests are any interrupt/exception that are not masked by the priority level of current exception and the priority level set by BASEPRI. In addition, if the interrupt request is based on IELSRn, the interrupt must be enabled by NVIC_ISERn.

2) Table 10.3

Before correction

Item	Software Standby Mode (SSTBY)	Deep Software Standby mode (DSTBY)		
	SSTBY	DSTBY1	DSTBY2	DSTBY3
Transition condition	WFI instruction after set LPSCR and CPU0.SCR.SLEEPDEEP=1	WFI instruction after set LPSCR and CPU0.SCR.SLEEPDEEP=1		
State after cancellation by an interrupt	Program execution state (interrupt processing)	Reset state		

After correction

Item	Software Standby Mode (SSTBY)	Deep Software Standby mode (DSTBY)		
	SSTBY	DSTBY1	DSTBY2	DSTBY3
Transition condition	When [Condition 1] or [Condition 2] while CPU.SCR.SLEEPDEEP=1 and LPSCR.LPMD is 0x4 [Condition 1] ・WFI instruction ・A valid interrupt request(*1) cannot be accepted to CPU. (including a transition from the time WFI instruction is executed to the time the transition to CPU Deep Sleep mode is completed) [Condition 2] ・CPU.SCR.SLEEPONEXIT=1 ・Complete execution of all exception handlers ・A valid interrupt request(*1) cannot be accepted to CPU. (including a transition from the time WFI instruction is executed to the time the transition to CPU Deep Sleep mode is completed)	When [Condition 1] or [Condition 2] while CPU.SCR.SLEEPDEEP=1 and LPSCR.LPMD is either 0x8 or, 0x9 or 0xA. [Condition 1] ・WFI instruction ・A valid interrupt request(*1) cannot be accepted to CPU. (including a transition from the time WFI instruction is executed to the time the transition to CPU Deep Sleep mode is completed) [Condition 2] ・CPU.SCR.SLEEPONEXIT=1 ・Complete execution of all exception handlers ・A valid interrupt request(*1) cannot be accepted to CPU. (including a transition from the time WFI instruction is executed to the time the transition to CPU Deep Sleep mode is completed)		
State after cancellation by an interrupt	Program execution state	Reset state		

(*1) Valid interrupt requests are any interrupt/exception that are not masked by the priority level of current exception and the priority level set by BASEPRI. In addition, if the interrupt request is based on IELSRn, the interrupt must be enabled by NVIC_ISERn.

3. Notes about the Sleep-on-exit function

There are 2 ways to transition to low power modes. One is WFI instruction and the other is Sleep-on-exit. When Sleep-on-exit is used for transition to low power modes, WFI instruction comments written in User's Manual Hardware is applicable to Sleep-on-exit.

4. Applicable condition and notes

[Applicable condition]

When all 1), 2) and 3) conditions are met.

1)Transition to CPU Deep Sleep mode is started by a trigger (WFI instruction or SLEEPONEXIT) with CPU.SCR.SLEEPDEEP=1 and LPSCR.LPMD is either 0x0 or 0x4 to use Software Standby mode, or CPU Deep Sleep mode.

2) When transition to CPU Deep Sleep mode, ICLK frequency is over 90MHz and CPUCLK:ICLK = 2:1.

3) During the specified interval (ICLK 1cycle) of transitioning to CPU Deep Sleep mode, one of the following interrupt requests that is not an interrupt source to return from CPU Deep Sleep mode is accepted by CPU.

a) SysTick interrupt (all of the following are applicable)

・Exception number 15 of Interrupt vector table .

・Interrupt requests are not masked by Base Priority Mask Register (BASEPRI)

(BASEPARI=0 or BASEPRI > SHPR3.PRI_15)

b) Maskable interrupt requests that are not interrupt source to return from CPU Deep Sleep mode (all of the following are

applicable)

- By WUPEN in exception numbers 16 to 111 in the interrupt vector table
  those not permitted to return from CPU Deep Sleep mode
- Interrupt requests are enabled by Interrupt Set-Enable Register (NVIC_ISERn).
- Interrupt requests are not masked by Base Priority Mask Register (BASEPRI)
  (BASEPARI=0 or BASEPRI > NVIC_IPRn.PRI_N)

[Notes]

If the above conditions are met, the MCU will rarely transit to following unintended states.

These unintended states can be resolved by a reset. If this unintended state happens, then the next transition to CPU Deep Sleep mode, Software Standby mode, and Deep Software Standby mode will fail.

Adapt a workaround if these unintended states are not acceptable.

1) When transitioning to CPU Deep Sleep mode (LPSCR.LPMD=0x0 and CPU.SCR.SLEEPDEEP=1)

CLK continues to operate the same as before transitioning to CPU Deep Sleep mode.

• Only CPU stop operating, current consumption becomes same level as CPU Sleep mode.

• As for the interrupt source settings to return from CPU Deep Sleep mode, these interrupt requests can return to Normal mode

2) When transitioning to Software Standby mode (LPSCR.LPMD=0x4 and CPU.SCR.SLEEPDEEP=1)

CLK continues to operate the same as before transitioning to CPU Deep Sleep mode.

• Only CPU stop operating, current consumption becomes same level as CPU Sleep mode.

• As before the transition to Software Standby mode is started, depending on the setting, timer or other peripherals continue to operate, and an interrupt request related to the peripheral is generated.

• Because WDT clock does not stop, a reset or an interrupt for WDT is generated depending on the settings before starting the transition to Software Standby mode.

• Interrupt requests are held in IR flag (IELSRn,).

• As for the interrupt source settings to return from CPU Deep Sleep mode, these interrupt requests can return to Normal mode

3) When transitioning to Deep Software Standby mode (LPSCR.LPMD=0x8 or 0x9 or 0xA and CPU.SCR.SLEEPDEEP=1)

The transition to Deep Software Standby mode is not possible, and the state shown in “1) When transitioning to Software Standby mode” is continued.

As for the interrupt source (DPSIERn) settings to returning from Deep Software Standby mode, these interrupt requests can return to Normal mode only when the interrupt source (WUPEN) to return from CPU Deep Sleep mode is set.

5. Workaround

[Workaround]

To avoid the unintended states described above, apply the following 1) or 2) before the condition to transit to CPU Deep Sleep mode, or Software Standby mode are met:

1) Set frequency setting as CPUCLK: ICLK=1:1.
2) Disable interrupt requests that are not interrupt sources to return from CPU Deep Sleep mode. : (For the setting procedure, see "_Setting Procedure for Transition to CPU Deep Sleep mode, or Software Standby Mode")

a. Disable SysTick interrupt requests.

Exception number 15 of Interrupt vector table

b. Disable maskable interrupt requests that are not interrupt sources to return from CPU Deep Sleep mode.

Exception number 16～111 of Interrupt vector table that WUPEN does not allow to return from CPU Deep Sleep mode

Setting Procedure for Transition to CPU Deep Sleep mode, or Software Standby Mode

This section describes procedures for avoiding unintended states.

The handling of interrupt requests after returning from CPU Deep Sleep mode or Software Standby mode varies depending on the method used to disable the maskable interrupt request. Either one or the other should be applied.

Procedure A) Disable maskable interrupt request acceptance.

Any interrupt request that occurs while interrupt request acceptance is disabled is discarded.

Before transitioning to CPU Deep Sleep mode, or Software Standby mode

Step1: Disable the SysTick interrupt request. (*1)

Step2: Clear IELSRn in ICU to disable acceptance of maskable interrupt requests that are not interrupt sources to return from CPU Deep Sleep mode.

Step3: Read IELSRn in ICU to confirm that IELSRn in ICU has been cleared.

Step4: Transition to Software Standby mode (WFI instruction, SLEEPONEXIT)

After returning from CPU Deep Sleep mode or Software Standby mode

Step5: Enable the SysTick interrupt request.

Step6: Set IELSRn in ICU to enable acceptance of maskable interrupt requests that are not interrupt sources to return from CPU Deep Sleep mode.

Step B) Disable the maskable interrupt request

The interrupt request generated while the interrupt request is disabled is retained in IELSRn.IR flag. Therefore, after returning from CPU Deep Sleep mode, Software Standby and enabling the maskable interrupt, it is possible to process the interrupt.

Before transitioning to Software Standby, Snooze, or Deep Software Standby mode

Step1: Disable the SysTick interrupt request. (*1)

Step2: Write 1 to the corresponding bit in NVIC_ICERn in CPU to disable maskable interrupt requests that are not interrupt sources to return from CPU Deep Sleep mode.

Step3: Execute Data Synchronization Barrier (DSB) instruction.

Step4: Transition Software Standby mode (WFI instruction, SLEEPONEXIT)

After returning from CPU Deep Sleep mode or Software Standby mode

Step5: Enable the SysTick interrupt request.

Step6:	Write 1 to the corresponding bit in NVIC_ISERn in CPU to enable maskable interrupt requests that are not interrupt sources to return from CPU Deep Sleep mode.

*1: Disabling a SysTick interrupt request may cause SysTick interrupt request to be delayed by one cycle of SysTick timer without generating the latest SysTick interrupt request.

Reference for each product table number, register name, mode, error name.

1.Interrupt vector table is as follows.

Table 13.3 shows the interrupt vector table for RA8M1, RA8D1, RA8T1, RA8E1, and RA8E2 groups.

2. WUPEN resister name for each product is as follows.

WUPEN0/1 is used for RA8M1, RA8D1, RA8T1, RA8E1, and RA8E2 groups.

Related Documentation

Product	Document Name
RA8M1 Group	Renesas RA8M1 Group User's Manual: Hardware Rev. 1.20
RA8D1 Group	Renesas RA8D1 Group User's Manual: Hardware Rev. 1.20
RA8T1 Group	Renesas RA8T1 Group User's Manual: Hardware Rev. 1.20
RA8E1 Group	Renesas RA8E1 Group User's Manual: Hardware Rev. 1.00
RA8E2 Group	Renesas RA8E2 Group User's Manual: Hardware Rev. 1.00