# SH7231 Group

## Using Deep Software Standby Mode

R01AN0822EJ0100
Rev.1.00
Jan. 11, 2012


## Abstract

This application note describes an example to use one of the features as power saving mode for the SH7231, the deep software standby mode.


## Products

SH7231

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

2. Operation Confirmation Conditions ........ 4

3. Reference Application Note ........ 4

4. Peripheral Functions ........ 5
4.1 Low Power Consumption Mode ........ 5
4.2 Interrupt Controller (INTC) ........ 5
4.3 Compare Match Timer (CMT) ........ 5

5. Hardware ........ 6
5.1 Pins Used ........ 6

6. Software ........ 7
6.1 Operation Overview ........ 7
6.2 File Composition ........ 8
6.3 Constants ........ 8
6.4 Sections ........ 8
6.5 Variables ........ 9
6.6 Functions ........ 9
6.7 Function Specifications ........ 10
6.8 Flowcharts ........ 12
6.8.1 Main Processing ........ 12
6.8.2 Initial Setting for Pin Function Controller ........ 13
6.8.3 Initial Setting for Compare Match Timer ........ 14
6.8.4 Initial Setting for IRQ ........ 15
6.8.5 Transition Processing to Deep Software Standby Mode ........ 16
6.8.6 Return Processing from Deep Software Standby Mode by Interrupts ........ 17
6.8.7 IRQ0 Interrupt Processing (Transition to Deep Software Standby Mode) ........ 18
6.8.8 CMI0 Interrupt Processing (Port G Control) ........ 19

7. Sample Code ........ 20

8. Reference Documents ........ 20

## 1. Specifications

In this application note, the following two processing are carried out; transition to the deep software standby mode using the external interrupt pin IRQ0, and cancellation of it with the external interrupt pin IRQ1.

In the sample code, the port G is controlled by the CMI0 interrupt of a compare match timer (hereinafter called CMT), which blinks LED0 to LED3.

Table 1.1 lists the peripheral functions and their applications. Figure 1.1 shows the block diagram of the peripheral functions.

**Table 1.1 Peripheral Functions and Their Applications**

| Peripheral Function | Application |
|---|---|
| Control of low power consumption mode | Transitions to /Cancels the deep software standby mode |
| Pin function controller (PFC) | Sets functions and input-output to the used pins at port G |
| I/O port | Sets the port G output |
| Interrupt controller (INTC) | Controls IRQ (IRQ0 and IRQ1) and CMT at channel 0 |
| Compare match timer (CMT) | Controls the port G in a cycle of blinking LEDs |

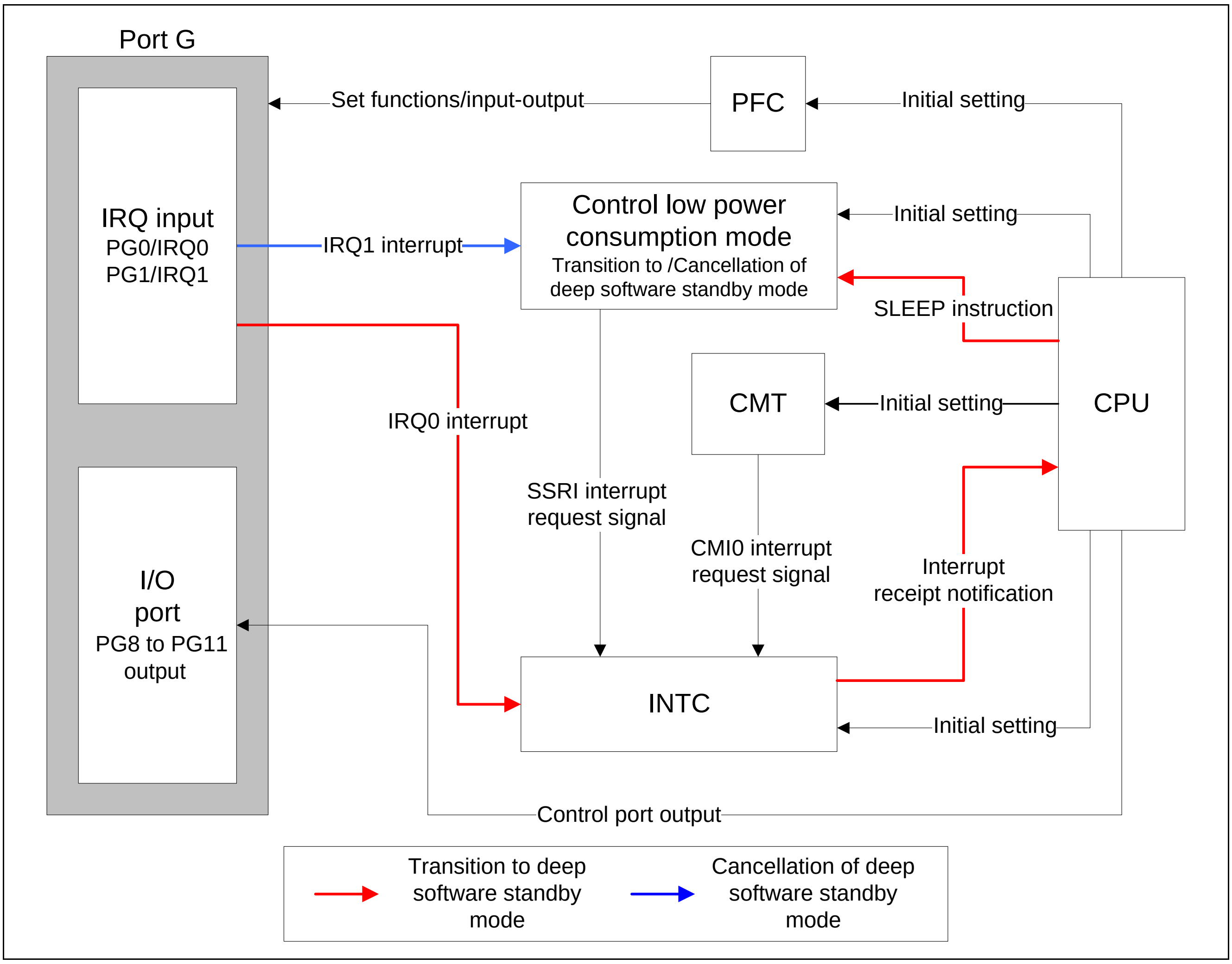


**Figure 1.1 Block Diagram of Peripheral Functions**

## 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**

| Item | Contents |
|---|---|
| MCU used | SH7231 |
| Devices used | None |
| Operating frequency | Internal clock: 100MHz<br>Bus clock: 50MHz<br>Peripheral clock: 50MHz |
| Operating voltage | Vcc: 3.3V |
| Integrated development environment | Renesas Electronics Corporation<br>High-performance Embedded Workshop Ver.4.08.00 |
| C compiler | Renesas Electronics Corporation<br>SuperH RISC engine family C/C++ compiler package V.9.04 Release 00 |
| | Compile options:<br>-cpu=sh2afpu -fpu=single<br>-include="$(WORKSPDIR)\inc","$(WORKSPDIR)\src\common"<br>-object="$(CONFIGDIR)\$(FILELEAF).obj" -debug -gbr=auto -chgincpath<br>-errorpath -global_volatile=0 -opt_range=all -infinite_loop=0<br>-del_vacant_loop=0 -struct_alloc=1 -nologo |
| Operating mode | Single chip mode |
| Sample code version | 1.00 |
| Board used | R0K572310C000BR |

## 3. Reference Application Note

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

- SH7231 Group Example of Initialization (R01AN0322EJ)

# 4. Peripheral Functions

This chapter provides information on low power consumption mode, interrupt controller (hereinafter called INTC) and compare match timer (hereinafter called CMT) used in this application note. Refer to the "SH7231 Group User’s Manual: Hardware" for basic information.

## 4.1 Low Power Consumption Mode

Following four types are available as low consumption modes and a function.

1. Sleep mode
2. Software standby mode
3. Deep software standby mode
4. Module standby function

Once transitioned to the deep software standby mode described in this application note, CPU, internal peripheral functions except key scan controller (hereinafter, KEYC), and 32kHz timer, high-speed on-chip RAM, and oscillator are halted and internal power supply is turned off. Therefore significant lower power supply is achieved.

In deep software standby mode, the registered contents and data in halted functions are all undefined. Only the data in the on-chip RAM (for data retention) can be kept by setting so.

Following means are available for canceling deep software standby mode; interrupts by external interrupt pins such as (hereinafter, NMI, IRQ0 to IRQ9), KEYC, and TIM32C, and the RES pin.

The interrupt by a cancellation source is delivered to the INTC side as the request signal for low power consumption interrupt (SSRI). Interrupt priority level can be set.

## 4.2 Interrupt Controller (INTC)

The INTC determines the priority on interrupt sources to control the interrupt requests to CPU.

According to the user-set priority level to the allocated register in each interrupt, the interrupt requests are processed.

Interrupts by IRQ0, IRQ1, and CMI0 are available in this application note.

## 4.3 Compare Match Timer (CMT)

The SH7231 includes a CMT consisting of a two-channel 16-bit timer. The CMT has a 16-bit counter, and can generate interrupts at set intervals.

In this application note, the compare match interrupt 0 (hereinafter called CMI0) allows blinking the LED periodically. The CMI0 is generated when the compare match counter 0 (CMCNT_0) and the compare match constant register 0 (CMCOR_0) match.

# 5. Hardware

## 5.1 Pins Used

Table 5.1 lists the pins used and their functions.

**Table 5.1 Pins Used and Their Functions**

| Pin Name | I/O | Function |
|---|---|---|
| PG0/IRQ0 | Input | Transition trigger to deep software standby mode (SW2 on the board used) |
| PG1/IRQ1 | Input | Cancel of deep software standby mode (SW3 on the board used) |
| PG8 | Output | LED0 blinking on the board used |
| PG9 | Output | LED1 blinking on the board used |
| PG10 | Output | LED2 blinking on the board used |
| PG11 | Output | LED3 blinking on the board used |

# 6. Software

## 6.1 Operation Overview

Figure 6.1 shows the relation between operation and data transfer.

1. The start is determined if it is a return from deep software standby mode by an interrupt at the timing of entering the main loop on start-up.
2. When determined as the restart by a cancellation interrupt, the cancellation processing function clears the setting of deep software standby mode control, and the setting of the cancellation source Then, the LED blinking pattern data in the on-chip RAM (for data retention) is copied to the high-speed on-chip RAM.
3. When determined not by a cancellation interrupt, the LED blinking pattern data in the on-chip ROM is copied to the high-speed on-chip RAM.
4. Afterward, the processing enters the main loop via the initial setting. By the compare match interrupt processing in CMI0, the port G control to blink LED repeats using the pattern data from the high speed on-chip RAM.
5. The transition to deep software standby mode starts when detecting the IRQ0 input as a falling edge. The LED blinking pattern data in the high speed on-chip RAM is copied to the on-chip RAM (for data retention), and the processing enters deep software standby mode.
6. Detecting the IRQ1 input at a falling edge during deep software standby mode, the mode is cancelled and the processing returns to power on reset processing.

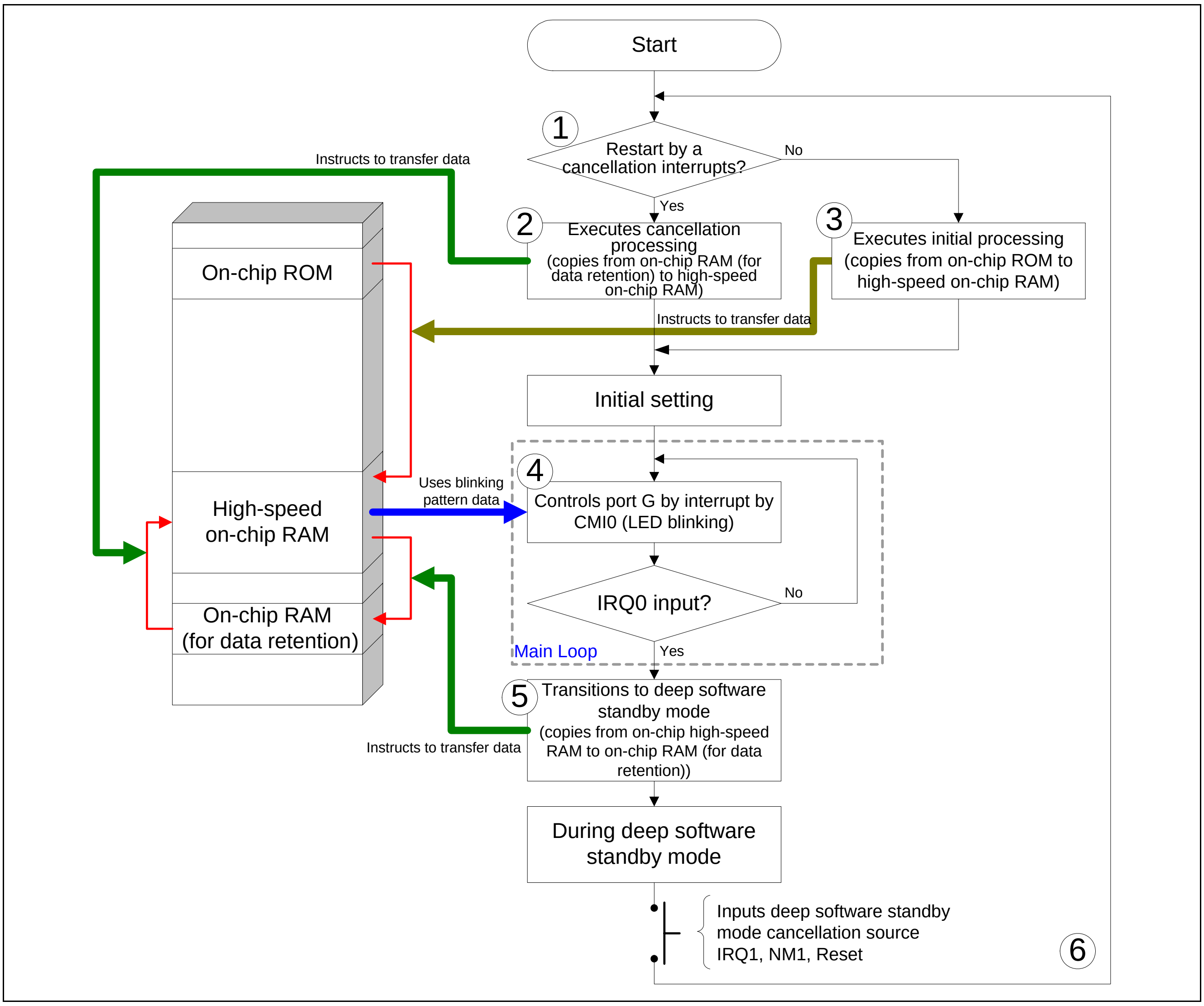


**Figure 6.1 Relation between Operation and Data Transfer**

## 6.2 File Composition

Table 6.1 lists the file used in the sample code. The files generated automatically in the integrated development environment are excluded in this table.

**Table 6.1 File Used in the Sample Code**

| File | Outline | Remarks |
|---|---|---|
| main.c | Main module | Initial setting, interrupt exception handling |

## 6.3 Constants

Table 6.2 lists the constants used in the sample code.

**Table 6.2 Constants Used in the Sample Code**

| Constant Name | Setting Value | Contents |
|---|---|---|
| LED_DATA_NUM | 8 | Used for the numbers of LED blinking data, and also used for elements of array and maximum value in the counter |
| BACKUP_RAM_TOP | H'FFFD 8000 | The start address in the on-chip RAM (for data retention) |

## 6.4 Sections

Table 6.3 lists the sections added in the sample code.

**Table 6.3 Sections Added in the Sample Code**

| Section Name | Setting Value | Contents |
|---|---|---|
| WriteData | H'FFF8 0000 | Variable for displaying LED-blinking pattern data in high-speed on-chip RAM |
| BackupRAM | H'FFFD 8000 | Variable for displaying LED-blinking pattern data in on-chip RAM (for data retention) |

## 6.5 Variables

Table 6.4 lists the global variables. Table 6.5 lists the const-type variables.

**Table 6.4 Global Variables**

| Type | Variable Name | Contents | Function Used |
|---|---|---|---|
| unsigned char | g_cnt_led_blink | Counts of LED blinking data | INT_CMT_CMI0 |
| unsigned short | g_buf_cmt_LedPatternWork [LED_DATA_NUM] | Storage area for LED blinking data in high-speed on-chip RAM | main, INT_CMT_CMI0, cancel_deep_standby, INT_IRQ0 |
| unsigned short | g_buf_cmt_LedPatternMem [LED_DATA_NUM] | Storage area for LED blinking data in on-chip RAM (for data retention) | cancel_deep_standby, INT_IRQ0 |

**Table 6.5 const-Type Variables**

| Type | Variable Name | Contents | Function Used |
|---|---|---|---|
| unsigned short | g_const_cmt_LedPattern [LED_DATA_NUM] | LED blinking pattern data | main, cancel_deep_standby, INT_IRQ0 |

## 6.6 Functions

Table 6.6 lists the functions.

**Table 6.6 List of Functions**

| Function Name | Outline |
|---|---|
| main | Main processing |
| io_init_pfc | Initial setting for pin function controller |
| io_init_cmt | Initial setting for compare match timer |
| io_init_irq | Initial setting for IRQ |
| goto_deep_standby | Transition processing to deep software standby mode |
| cancel_deep_standby | Return processing from deep software standby mode by interrupts |
| INT_IRQ0 | IRQ0 interrupt processing (transition to deep software standby mode) |
| INT_CMT_CMI0 | CMI0 interrupt processing (port G control) |

## 6.7 Function Specifications

The following tables list the sample code function specifications.

| Main | |
|---|---|
| **Outline** | Main processing |
| **Header** | |
| **Declaration** | void main(void) |
| **Description** | Checks if the restart is at returning from deep software standby mode by an interrupt. If so, executes a cancellation processing function. If not, transfers the LED blinking pattern data in the on-chip ROM to the high-speed on-chip RAM. Then, executes initial setting processing functions to enter the main loop. |
| **Arguments** | None |
| **Returned Value** | None |

| io_init_pfc | |
|---|---|
| **Outline** | Initial setting for pin function controller |
| **Header** | |
| **Declaration** | void io_init_pfc(void) |
| **Description** | Sets input I/O setting at port G. Sets input to IRQ0 and IRQ1. |
| **Arguments** | None |
| **Returned Value** | None |

| io_init_cmt | |
|---|---|
| **Outline** | Initial setting for compare match timer |
| **Header** | |
| **Declaration** | void io_init_cmt(void) |
| **Description** | Sets the registers for compare match timer (channel 0). |
| **Arguments** | None |
| **Returned Value** | None |

| io_init_irq | |
|---|---|
| **Outline** | Initial setting for IRQ |
| **Header** | |
| **Declaration** | void io_init_irq(void) |
| **Description** | Sets the detection edge and interrupt priority of IRQ0. |
| **Arguments** | None |
| **Returned Value** | None |

| goto_deep_standby | |
|---|---|
| **Outline** | Transition processing to deep software standby mode |
| **Header** | |
| **Declaration** | void goto_deep_standby(void * dest, void * src, size_t size) |
| **Description** | After transferring the LED blinking pattern data from the high-speed on-chip RAM to the on-chip RAM (for data retention), sets the registers for deep software standby mode and transitions to deep software standby mode. |
| **Arguments** | void * dest : First address in the area of transfer destination |
| | void * src : First address in the area of transfer source |
| | size_t size : Data size to transfer |
| **Returned Value** | None |

| cancel_deep_standby | |
|---|---|
| **Outline** | Return processing from deep software standby mode by interrupts |
| **Header** | |
| **Declaration** | void cancel_deep_standby(void) |
| **Description** | When determined IRQ1 as the deep software standby mode cancellation source, transfers the LED blinking pattern data from the on-chip RAM (for data retention) to the high-speed on-chip RAM. Then clears the cancellation source register. |
| **Arguments** | None |
| **Returned Value** | None |

| INT_IRQ0 | |
|---|---|
| **Outline** | IRQ0 interrupt processing (transition to deep software standby mode) |
| **Header** | |
| **Declaration** | void INT_IRQ0(void) |
| **Description** | Interrupt handling by IRQ0 input.<br>Executes the function goto_deep_standby. |
| **Arguments** | None |
| **Returned Value** | None |

| INT_CMT_CMI0 | |
|---|---|
| **Outline** | CMI0 interrupt processing (port G control) |
| **Header** | |
| **Declaration** | void INT_CMT_CMI0(void) |
| **Description** | Controls port G using the pattern data (g_buf_cmt_LedPatternWork[]) read from the counter value (g_cnt_led_blink) and blinks the LEDs. |
| **Arguments** | None |
| **Returned Value** | None |

## 6.8 Flowcharts

### 6.8.1 Main Processing

Figure 6.2 shows the procedure of main processing.

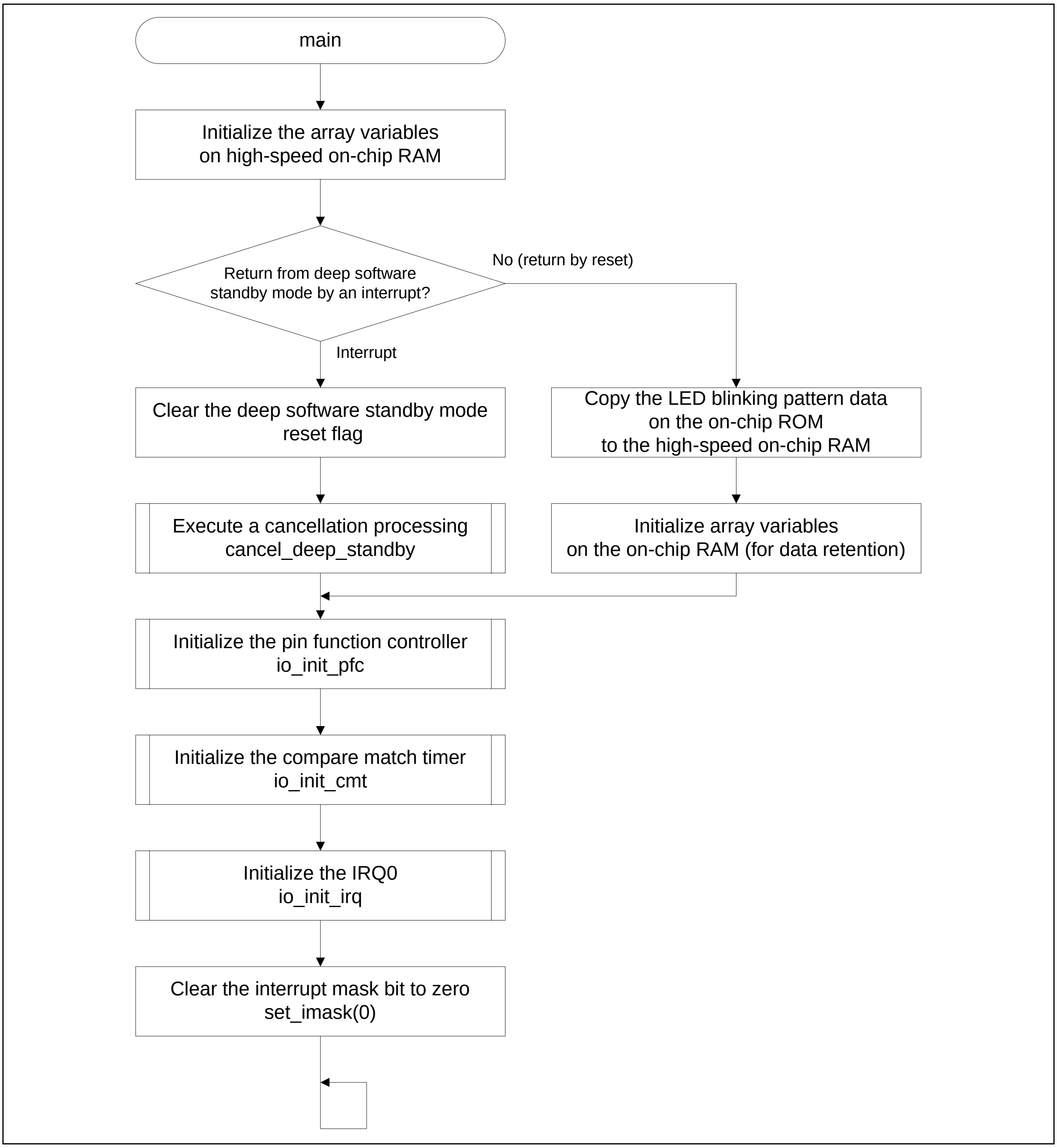


**Figure 6.2 Procedure of Main Processing**

### 6.8.2 Initial Setting for Pin Function Controller

Figure 6.3 shows the procedure of initial setting for the pin function controller.

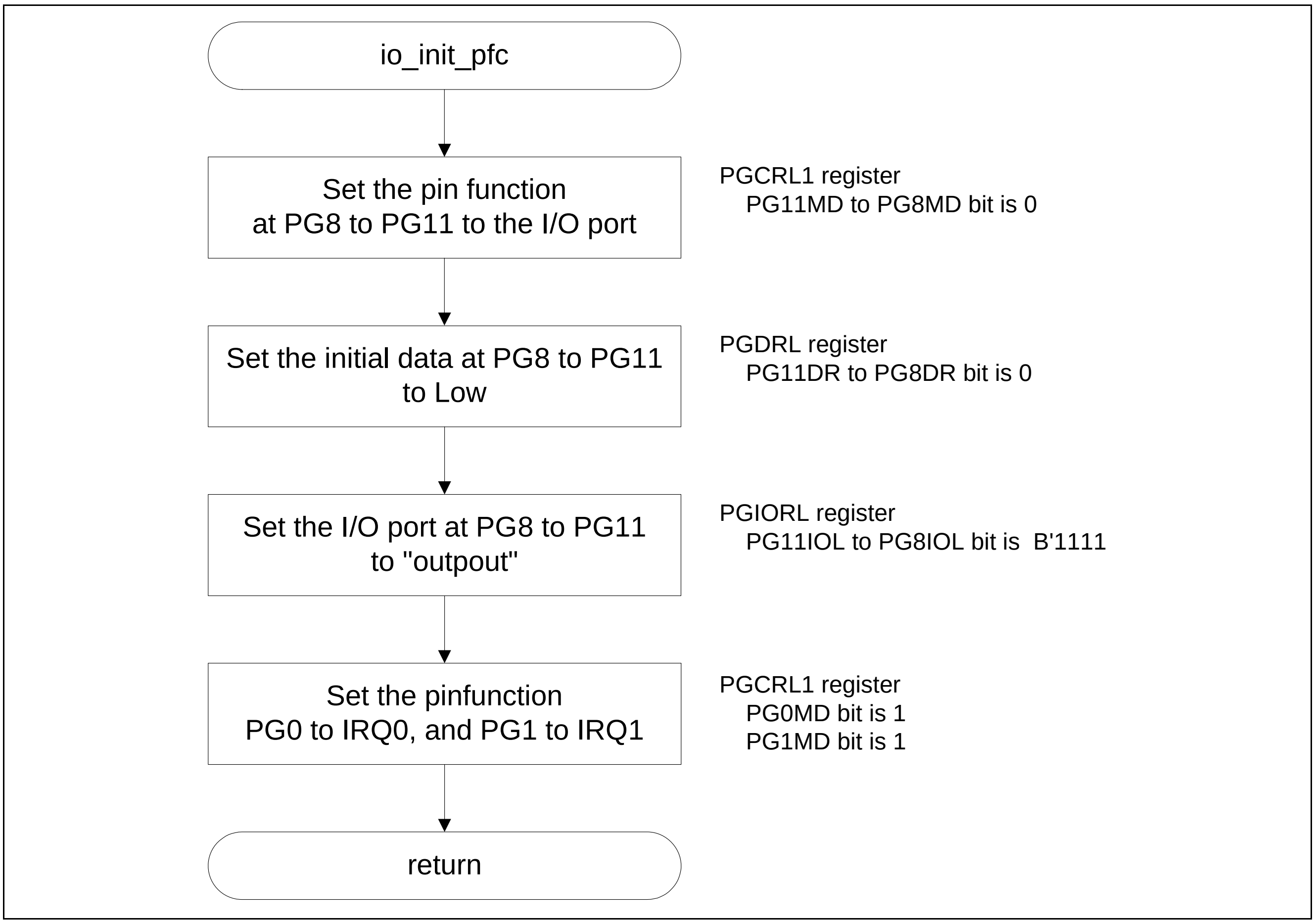


**Figure 6.3 Procedure of Initial Setting for Pin Function Controller**

### 6.8.3 Initial Setting for Compare Match Timer

Figure 6.4 shows the procedure of initial setting for the compare match timer.

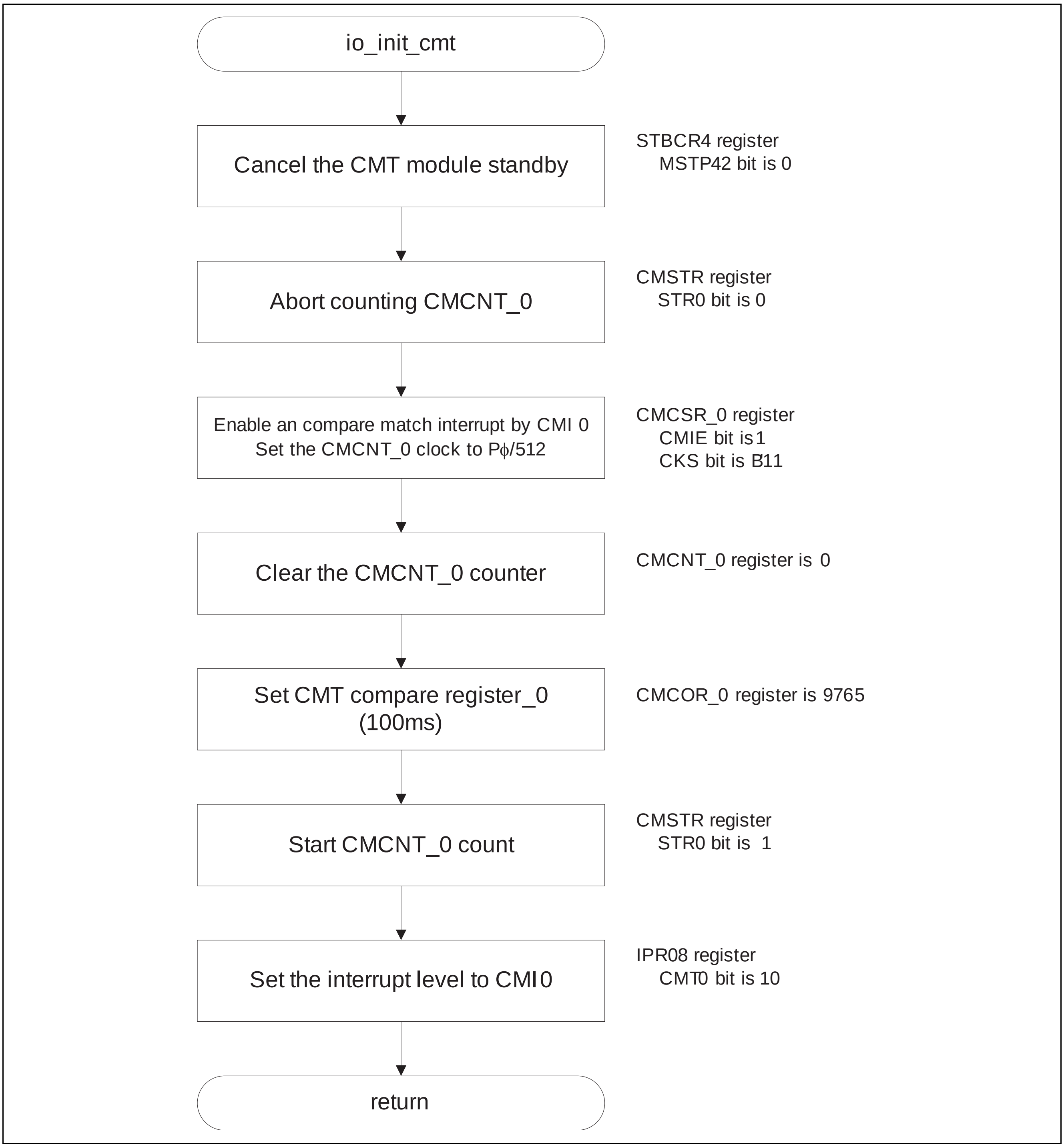


**Figure 6.4 Procedure of Initial Setting for Compare Match Timer**

### 6.8.4 Initial Setting for IRQ

Figure 6.5 shows the procedure of initial setting for the IRQ.

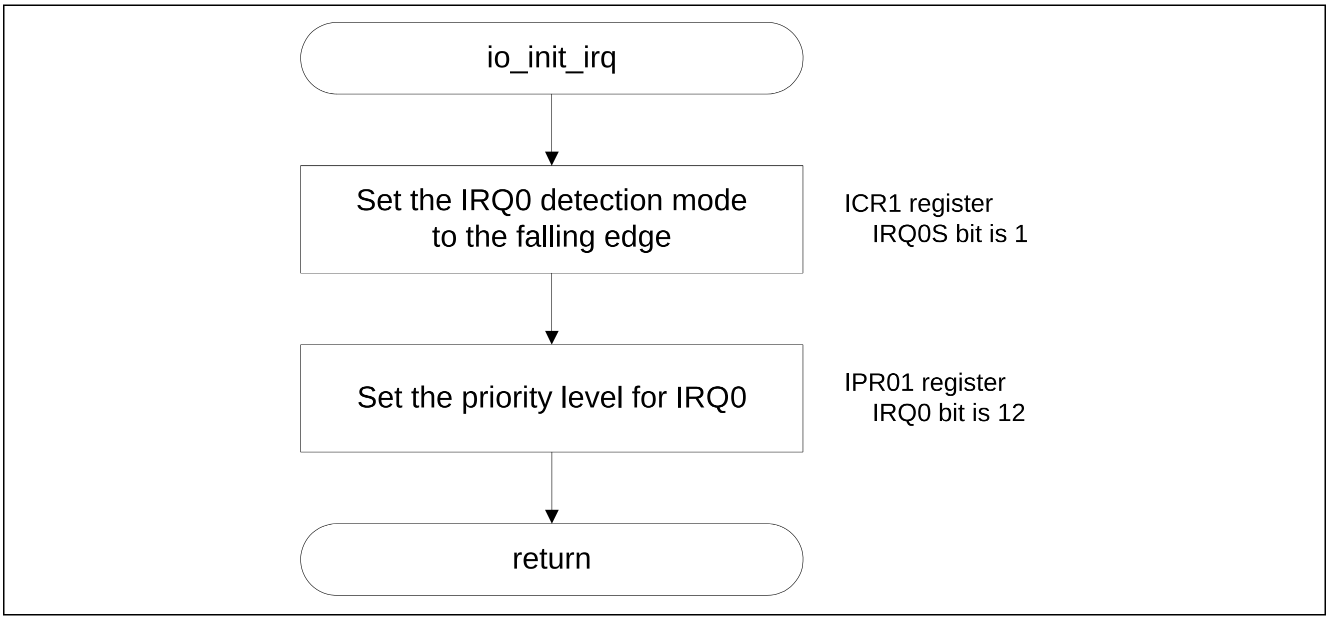


**Figure 6.5 Procedure of Initial Setting for IRQ**

### 6.8.5 Transition Processing to Deep Software Standby Mode

Figure 6.6 shows the procedure of transition processing to the deep software standby mode.

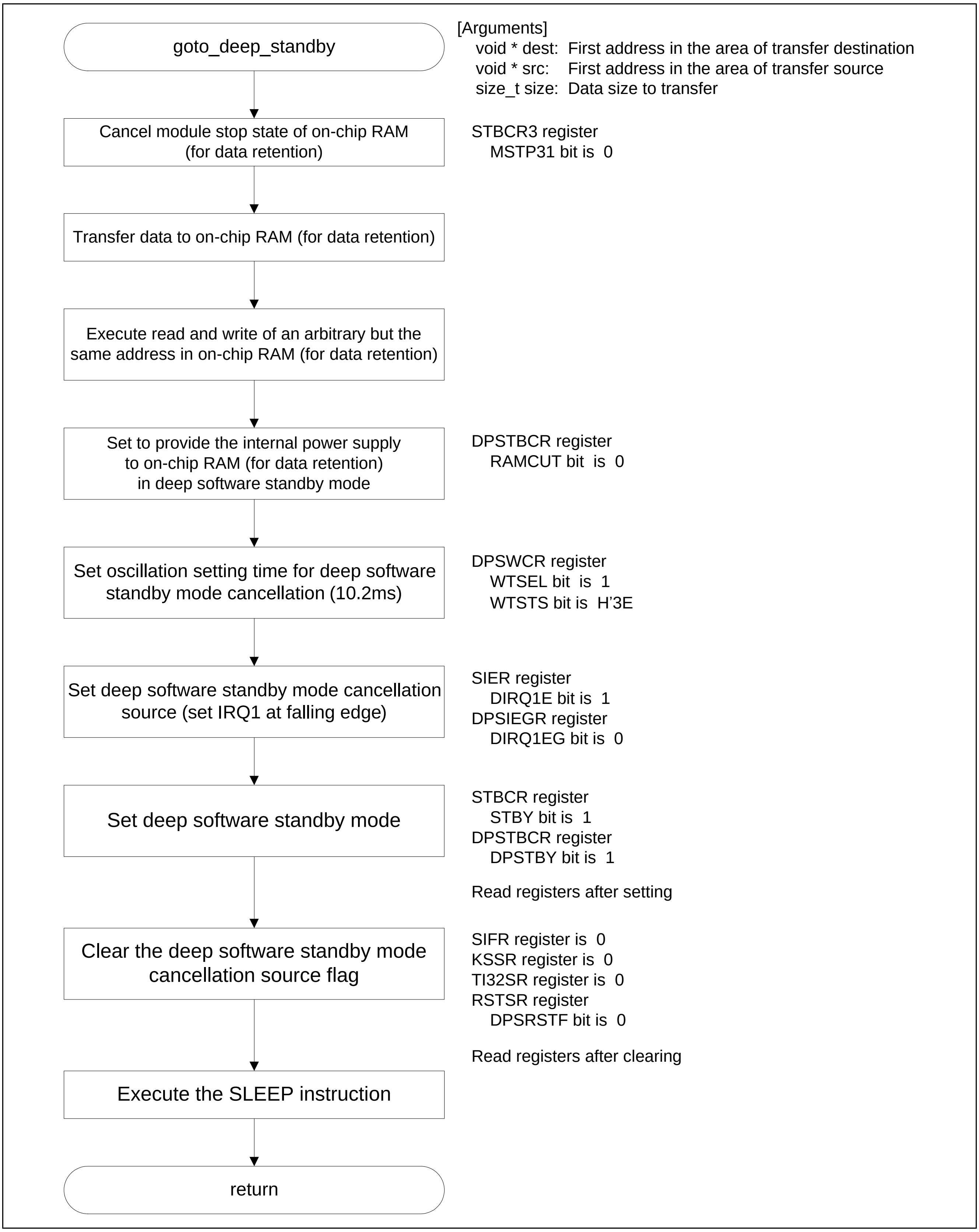


**Figure 6.6 Procedure of Transition Processing to Deep Software Standby Mode**

### 6.8.6 Return Processing from Deep Software Standby Mode by Interrupts

First, clear the register set for transition to the deep software standby mode.

Next, when the cancellation source is IRQ1, the LED blinking pattern data in the on-chip RAM (for data retention) is transferred to the high-speed on-chip RAM.

In case the cancellation source is other than IRQ1, the LED blinking pattern data is transferred from the on-chip ROM to the high-speed on-chip RAM.

Then, clear the cancellation source, standby interrupt flag register (SIFR).

Figure 6.7 shows the procedure of return processing from the deep software standby mode.

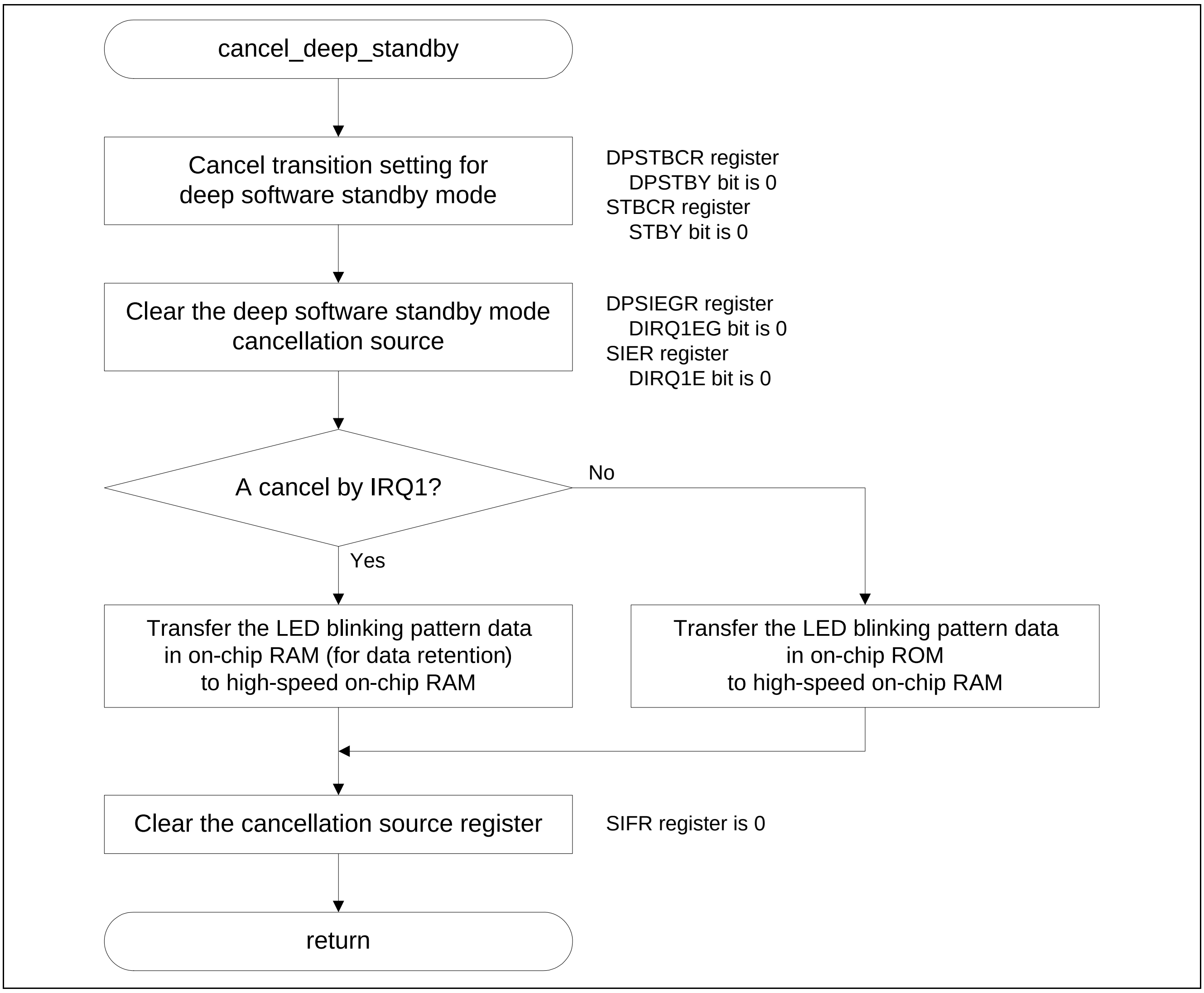


**Figure 6.7 Procedure of Return Processing from Deep Software Standby Mode**

### 6.8.7 IRQ0 Interrupt Processing (Transition to Deep Software Standby Mode)

Figure 6.8 shows the procedure of IRQ0 interrupt processing for transition to the deep software standby mode.

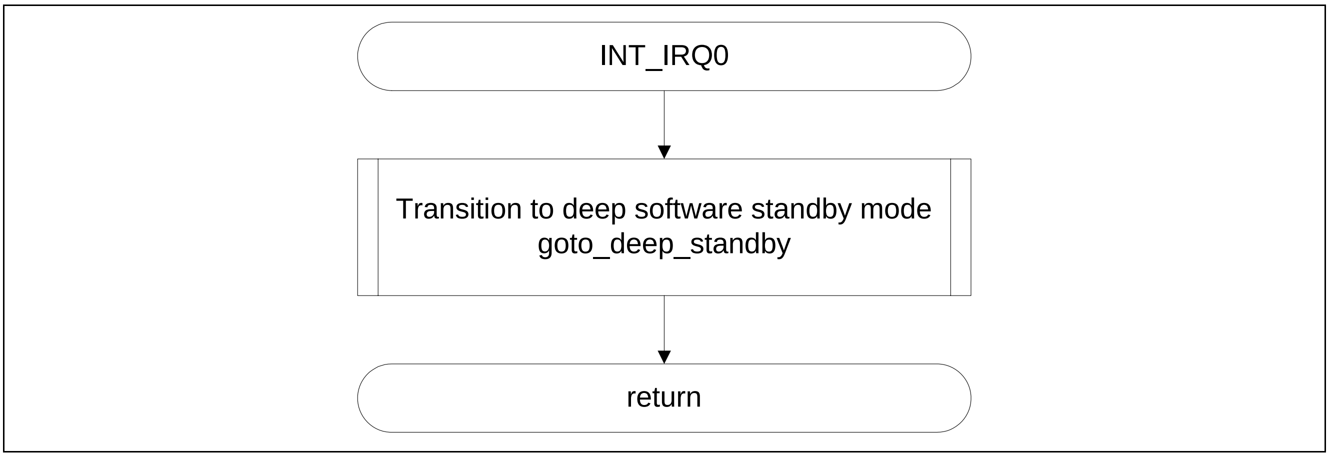


**Figure 6.8 Procedure of IRQ0 Interrupt Processing (Transition to Deep Software standby Mode)**

### 6.8.8 CMI0 Interrupt Processing (Port G Control)

Figure 6.9 shows the procedure of CMI0 interrupt processing for port G control.

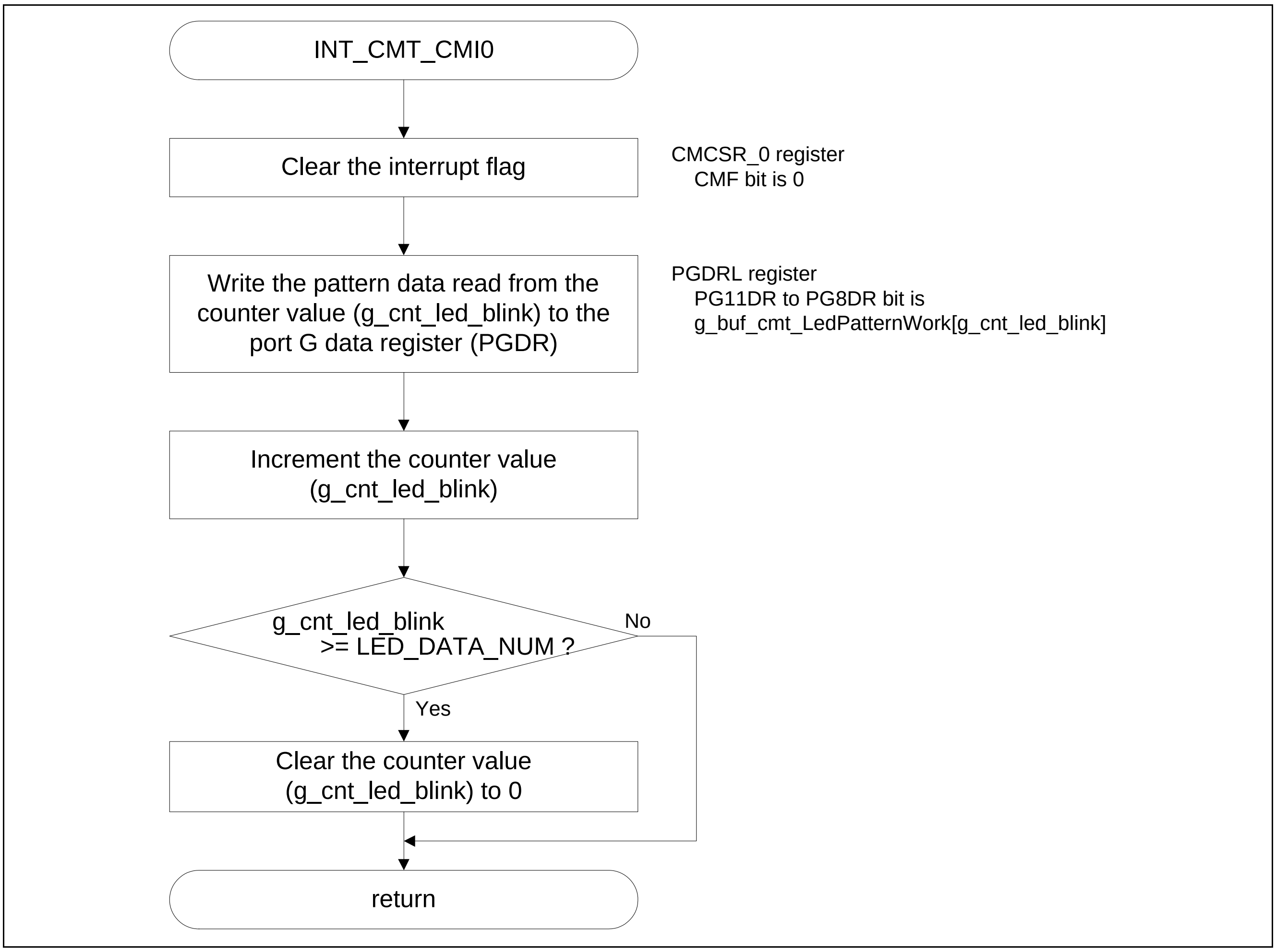


**Figure 6.9 Procedure of CMI0 Interrupt Processing**

## 7. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

## 8. Reference Documents

User's Manual: Hardware

SH7231 Group User's Manual: Hardware Rev.1.00 (R01UH0073EJ)
The latest version can be downloaded from the Renesas Electronics website.

Technical Update/Technical News

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

User's Manual: Development Tools

SuperH C/C++ Compiler Package V.9.04 User's Manual Rev.1.01 (R20UT0704EJ)
The latest version can be downloaded from the Renesas Electronics website.

## Website and Support

Renesas Electronics website

http://www.renesas.com

Inquiries

http://www.renesas.com/contact/

| **Revision History** | SH7231 Group Application Note Using Deep Software Standby Mode |
|---|---|

| **Rev.** | **Date** | **Description** | |
|---|---|---|---|
| | | **Page** | **Summary** |
| 1.00 | Jan. 11, 2012 | — | First edition issued |
| | | | |

## General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins
   Handle unused pins in accord 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 LSI, an associated shoot-through current flow internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.
2. Processing at Power-on
   The state of the product is undefined at the moment 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 moment 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 moment 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 moment when power is supplied until the power reaches the level at which resetting has been specified.
3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.
4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has 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. Moreover, 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.
5. Differences between Products
   Before changing from one product to another, i.e. to one with a different type number, confirm that the change will not lead to problems.
   — The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

**SALES OFFICES** Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

**Renesas Electronics America Inc.**
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

**Renesas Electronics Canada Limited**
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

**Renesas Electronics Europe Limited**
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

**Renesas Electronics Europe GmbH**
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

**Renesas Electronics (China) Co., Ltd.**
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

**Renesas Electronics (Shanghai) Co., Ltd.**
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

**Renesas Electronics Hong Kong Limited**
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

**Renesas Electronics Taiwan Co., Ltd.**
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

**Renesas Electronics Singapore Pte. Ltd.**
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

**Renesas Electronics Malaysia Sdn.Bhd.**
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

**Renesas Electronics Korea Co., Ltd.**
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141


Colophon 1.1