To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

H8/300H Tiny Series

Entering Low Power Consumption Mode (H8/3687)

Introduction

The switching function for low power consumption modes of the H8/3687 Group products allows the user to choose from among five different modes: active, sub-active, sleep, sub-sleep, and standby.

Target Device

H8/3687

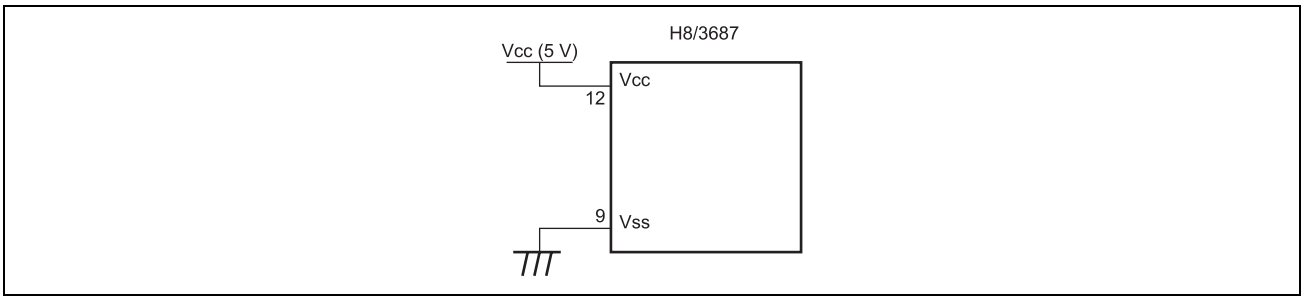
Contents

1. Overview 2
2. Configuration 2
3. Sample Program 3
4. Reference Documents 15

1. Overview

By means of a function for switching between H8/3687 low power consumption modes, switching between active, sub-active, sleep, sub-sleep, and standby modes is performed.

2. Configuration



Specifications

H8/3687 operating frequency: 16 MHz

3. Sample Program

3.1 Functions

- Direct transitions are made into active and sub-active modes.
- Transitions are made into sleep, sub-sleep, and standby modes. Interrupts are used to return to active and sub-active modes at the specified operating frequency.

3.2 Program incorporation

- Sample program 2-A
  Incorporate #define directives.
- Sample program 2-B
  Incorporate prototype declarations.
- Sample program 2-C
  Add the sample program 2-C as the common routine.

3.3 Modifications to sample programs

Without modifications to the sample program, the system may not run. Modifications must be made according to the customer's program and system environment.

- By using a file with definitions of IO register structures which can be obtained free of charge from the Renesas web site, the sample program can be used without further changes. When creating definitions independently, the customer should modify the IO register structures used in the sample program as appropriate.

3.4 Method of use

- The operating frequency is changed.

```
com_frequency_change (int return_mode, int frequency)
```

Argument	Description
return_mode	Specifies the operating mode after the frequency is changed. ACTIVE_MODE(0): Active mode SUB_ACTIVE_MODE(1): Sub-active mode
frequency	Indicates the operating frequency after the change. OSC_PER_1 (0x00): ϕ (In this sample program, setting is 16 MHz) OSC_PER_8 (0x10): ϕ/8 (In this sample program, setting is 2 MHz)* OSC_PER_16 (0x14): ϕ/16 (In this sample program, setting is 1 MHz)* OSC_PER_32 (0x18): ϕ/32 (In this sample program, setting is 0.5 MHz)* OSC_PER_64 (0x1C): ϕ/64 (In this sample program, setting is 0.25 MHz)* OSC_PER_W8 (0x00): ϕw/8 (4.096 kHz) OSC_PER_W4 (0x01): ϕw/4 (8.192 kHz) OSC_PER_W2 (0x02): ϕw/2 (16.384 kHz) Note: * Settings resulting in a frequency of less than 78.125 kHz (the minimum operating frequency) are not possible. Example: When ϕ=4 MHz, ϕ/64=62.5 kHz, and so OSC_PER_64 (0x1C) cannot be specified.

Example of use:

```
/* disable SCI3 reception interrupts */
SCI3.SCR3.BYTE = 0x10;
/* invalidate TimerZ interrupts */
TZ0.TIER.BIT.IMIEA = 0;
/* change the frequency */
com_frequency_change (ACTIVE_MODE, OSC_PER_8);
// return using a direct transition interrupt
// make transition to active mode, operating frequency ϕ/8
```

- Make a transition to sleep/sub-sleep/standby mode. When an interrupt occurs, return to active/sub-active mode at the specified operating frequency.

```
com_mode_change (int stop_mode, int return_mode, int freuqncy)
```

Argument	Description
stop_mode	Specifies a stop mode. SLEEP_MODE(0): Sleep mode SUB_SLEEP_MODE(1): Sub-sleep mode STANDBY_MODE(2): Standby mode
return_mode	Specifies the operating mode after the frequency is changed. ACTIVE_MODE(0): Active mode SUB_ACTIVE_MODE(1): Sub-active mode
frequency	Indicates the operating frequency after the change. OSC_PER_1 (0x00): ϕ (In this sample program, setting is 16 MHz) OSC_PER_8 (0x10): ϕ/8 (In this sample program, setting is 2 MHz)* OSC_PER_16 (0x14): ϕ/16 (In this sample program, setting is 1 MHz)* OSC_PER_32 (0x18): ϕ/32 (In this sample program, setting is 0.5 MHz)* OSC_PER_64 (0x1C): ϕ/64 (In this sample program, setting is 0.25 MHz)* OSC_PER_W8 (0x00): ϕw/8 (4.096 kHz) OSC_PER_W4 (0x01): ϕw/4 (8.192 kHz) OSC_PER_W2 (0x02): ϕw/2 (16.384 kHz) Note: * Settings resulting in a frequency of less than 78.125 kHz (the minimum operating frequency) are not possible. Example: When ϕ = 4 MHz, ϕ/64 = 62.5 kHz, and so OSC_PER_64 (0x1C) cannot be specified.

Example of use:

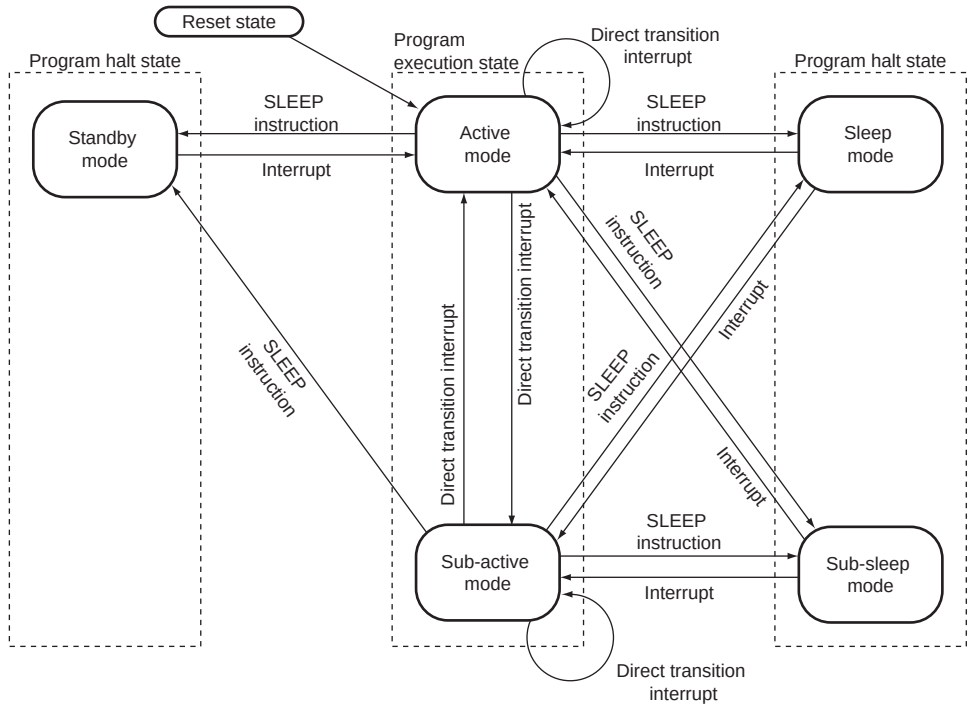
```
/* disable SCI3 reception interrupts */
SCI3.SCR3.BYTE = 0x10;
/* invalidate TimerZ interrupts */
TZ0.TIER.BIT.IMIEA = 0;
/* enable int 0*/
IENR1.BIT.IEN0 = 1;
com_mode_change (SLEEP_MODE, ACTIVE_MODE, OSC_PER_1);
// enter sleep mode; return on int0 interrupt
// active mode, return at frequency ϕ
```

Important information: When making a transition to sub-active mode, operation is limited as follows.

- The RTC interval timer cannot be used.
- The watchdog timer, I^2C, timer B1, and timer Z cannot be used.
- The timer V, SCI3, and A/D converter are reset, and so the various registers should be rewritten as necessary.

3.5 Description of operation

Figure 1 shows the possible transitions between modes. A transition from a program execution state to a program halt state is made by executing a SLEEP instruction. An interrupt is used to return from the program halt state to the program execution state. Direct transitions can also be made between active mode and sub-active mode, which are program execution states, without stopping program execution. And, by making direct transitions from active mode to active mode, and from sub-active mode to sub-active mode, the operating frequency can be changed in the same mode. A transition from all modes to the reset state is made through RES input. Table 1 indicates transition conditions to various modes when the SLEEP instruction is executed, as well as return destinations resulting from an interrupt; Table 2 describes the internal states of the LSI device in each of the operating modes.



Notes: 1. In transitions between modes using interrupts, the interrupt request must be accepted, and interrupt processing performed.
2. For details of conditions for transitions between modes, refer to Table 1.

Figure 1 Mode transition diagram

Table 1 States upon SLEEP instruction execution and return destinations on interrupt

SYSCR1	SYSCR2			States upon SLEEP Instruction Execution	Return Destination on Interrupt
DTON	SSBY	SMSEL	LSON		
0	0	0	0	Sleep mode	Active mode
0	0	0	1	Sleep mode	Sub-active mode
0	0	1	0	Sub-sleep mode	Active mode
0	0	1	1	Sub-sleep mode	Sub-active mode
0	1	×	×	Standby mode	Active mode
1	×	0*	0	Active mode (direct transition)	
1	×	×	1	Sub-active mode (direct transition)	

[Legend] ×: Don't care

Note: * When making a state transition with SMSEL=1, the timer V, SCI3, and A/D converter are set, and the values in registers are returned to the initial values. After a transition to active mode, when using these functions the register values must be rewritten.

Table 2 LSI states in different operating modes

Function		Active	Sleep	Sub-active	Sub-sleep	Standby
System clock oscillator		Operates	Operates	Stop	Stop	Stop
Sub-clock oscillator		Operates	Operates	Operates	Operates	Operates
CPU	Instruction execution	Operates	Stop	Operates	Stop	Stop
	Register	Operates	Held	Operates	Held	Held
RAM		Operates	Held	Operates	Held	Held
I/O port		Operates	Held	Operates	Held	Registers held; output at high impedance
External interrupt	IRQ3 to IRQ0	Operates	Operates	Operates	Operates	Operates
	WKP5 to WKP0	Operates	Operates	Operates	Operates	Operates
Peripheral module	RTC	Operates	Operates	Operates when clock time base function selected; held when interval timer selected		
	Timer V	Operates	Operates	Reset	Reset	Reset
	Watchdog timer	Operates	Operates	Held (however, operates when internal oscillator selected for counter clock*)		
	SCI3, SCI3_2	Operates	Operates	Reset	Reset	Reset
	IIC2	Operates	Operates	Held*	Held	Held
	Timer B1	Operates	Operates	Held*	Held	Held
	Timer Z	Operates	Operates	Held*	Held	Held
	A/D converter	Operates	Operates	Reset	Reset	Reset

Note: * In sub-active mode, register read/write is possible.

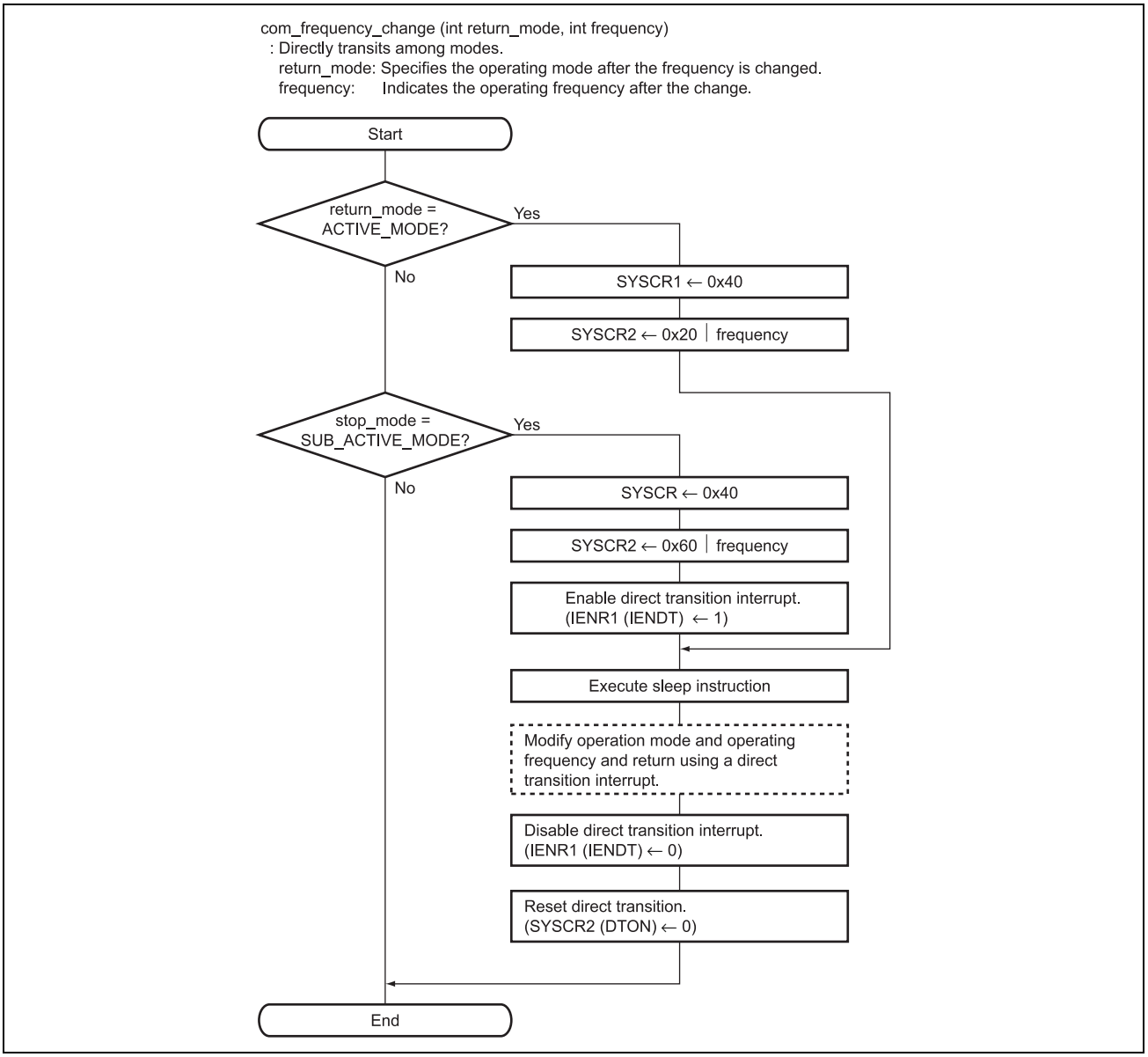
3.6 List of registers used

The H8 microcomputer internal registers used in this sample program are listed. For details, refer to the H8/3687 Group Hardware Manual.

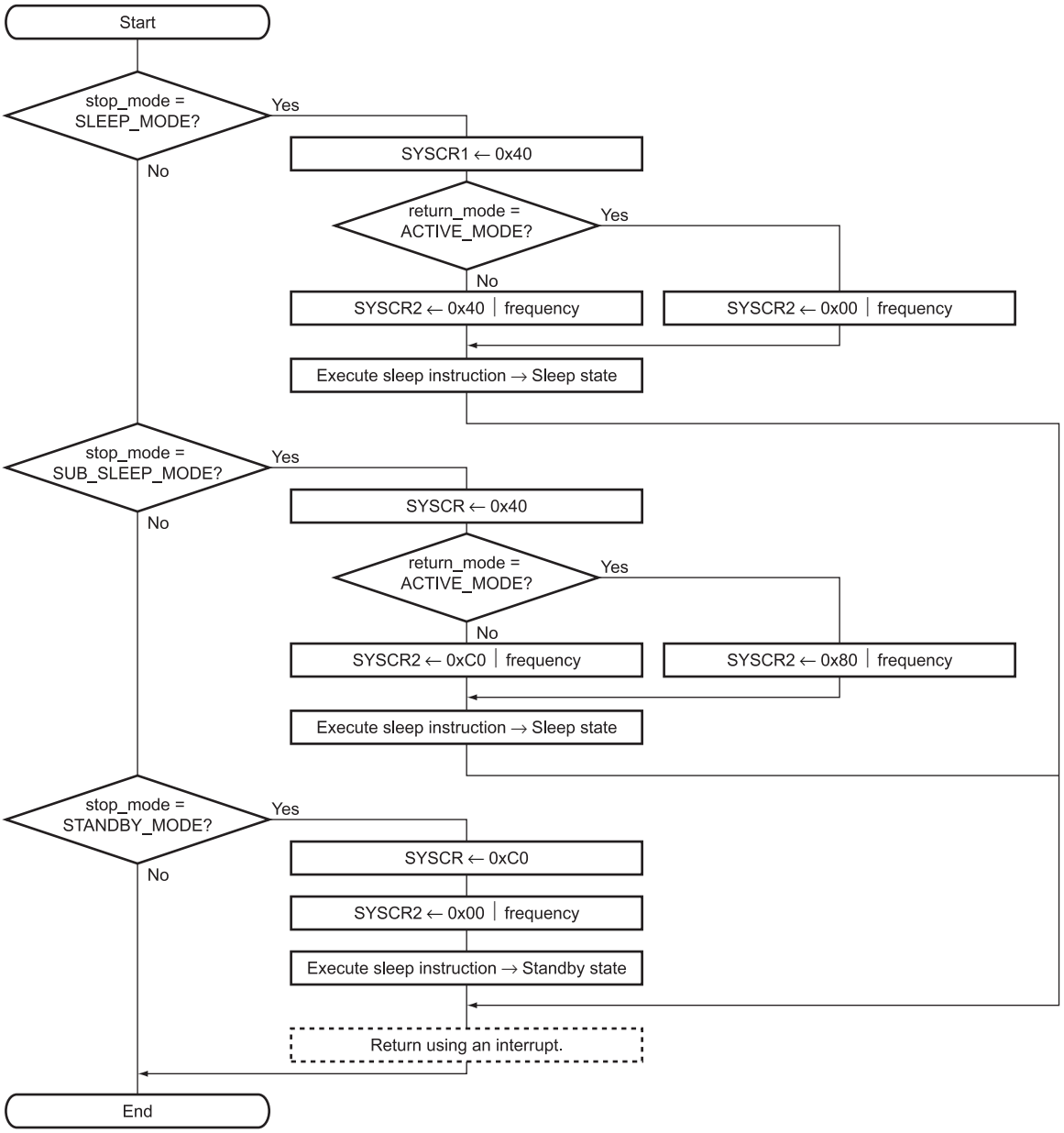
Name	Description
System control register 1 (SYSCR1)	Controls low power consumption modes
System control register 2 (SYSCR2)	Controls low power consumption modes

3.7 Flowcharts

The flow of program execution is indicated below.



com_mode_change (int stop_mode, int return_mode, int frequency)
: Makes transition among modes.
stop_mode: Specifies a stop mode.
return_mode: Specifies the operating mode after the frequency is changed.
frequency: Indicates the operating frequency after the change.



3.8 Program Listing

```
/* ------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */
/* 1. Sample Program 11-A #define directives ------------------------------------------------------------------------------  */
/* ------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */

/************************************************************************************************************************/
/*   For frequency and mode modification                                                                                    */
/************************************************************************************************************************/
#define   OSC_PER_1           0x00
#define   OSC_PER_8           0x10
#define   OSC_PER_16          0x14
#define   OSC_PER_32          0x18
#define   OSC_PER_64          0x1C
#define   OSC_PER_W8          0x00
#define   OSC_PER_W4          0x01
#define   OSC_PER_W2          0x02

#define   ACTIVE_MODE         0
#define   SUB_ACTIVE_MODE     1

#define   SLEEP_MODE          0
#define   SUB_SLEEP_MODE      1
#define   STANDBY_MODE        2



/* ------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */
/* 2. Sample program 11-B  Prototype declaration ------------------------------------------------------------------------- */
/* ------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */

/* Mode modification processing                                                                                             */
void com_mode_change (int stop_mode , int return_mode , int frequency) ;
void com_frequency_change (int return_mode , int frequency ) ;
```

```
/* ------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */
/* 4. Sample program 11-C  Common source codes ---------------------------------------------------------------------------- */
/* ------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */

/************************************************************************************************************************/
/************************************************************************************************************************/
/************************************************************************************************************************/
/*                                                                                                                      */
/*                           Low power consumption mode                                                                 */
/*                                                                                                                      */
/************************************************************************************************************************/
/************************************************************************************************************************/
/************************************************************************************************************************/

/************************************************************************************************************************/
/*   1. Module name: com_mode_change                                                                                    */
/*   2. Function overview:' Performs a mode transition                                                                  */
/*     Parameter combination                                                                                            */
/*     stop_mode         return_mode        frequency                                                                   */
/*     SLEEP_MODE   →    ACTIVE_MODE        OSC_PER_1/8/16/32/64                                                        */
/*     SLEEP_MODE   →    SUB_ACTIVE_MODE    OSC_PER_W8/4/2                                                              */
/*     SUB_SLEEP_MODE    →ACTIVE_MODE       OSC_PER_1/8/16/32/64                                                        */
/*     SUB_SLEEP_MODE    →SUB_ACTIVE_MODE   OSC_PER_W8/4/2                                                              */
/*     STANDBY_MODE      →ACTIVE_MODE       OSC_PER_1/8/16/32/64                                                        */
/*                                                                                                                      */
/************************************************************************************************************************/
void com_mode_change (int stop_mode , int return_mode , int frequency    )
{

     switch (stop_mode){
          case SLEEP_MODE :
               /************************************************************************************************************/
               /*   Sets SYSCR1 :                                                                                          */
               /*        SSBSY     = 0 Enters sleep mode during sleep instruction execution                                */
               /*        STS2:0    = 100 specifies the wait time for clock stabilization from standby to sleep as 8.2 ms.  */
               /*        NESEL     = 0 specifies the noise elimination sampling frequency as 8 MHzk/16.                    */
               /************************************************************************************************************/
               SYSCR1.BYTE    = 0x40 ;

               /************************************************************************************************************/
               /*   Sets SYSCR2 :                                                                                          */
               /*        SMSEL     = 0 Enters sleep mode                                                                   */
               /*        LSON      = 0/1                                                                                   */
               /*        DTON      = 0 Disables direct transition during sleep instruction execution                       */
               /*        MA2:0     = 000 Selects the main clock operating frequency (Specified by argument frequency)      */
               /*        SA1:0     = 00 Selects the sub-clock operating frequency (Specified by argument frequency)        */
               /************************************************************************************************************/
               if (return_mode == ACTIVE_MODE){
                    SYSCR2.BYTE    = 0x00 |  frequency ;
               }
               else{
                    SYSCR2.BYTE    = 0x40 |  frequency ;
               }
```

```
        /*****************************************************************************************************************/
        /*   Makes the device sleep and enters sleep mode.                                                               */
        /*****************************************************************************************************************/
        sleep() ;

        /* ============================================================================================================= */
        /* === Returns here by an interrupt                                                                          === */
        /* ============================================================================================================= */

        break ;

    case SUB_SLEEP_MODE :
        /*****************************************************************************************************************/
        /*   Sets SYSCR1 :                                                                                               */
        /*        SSBSY     = 0 Enters sleep mode during sleep instruction execution                                     */
        /*        STS2:0    = 100 specifies the wait time for clock stabilization from standby to sleep as 8.2 ms.       */
        /*        NESEL     = 0 specifies the noise elimination sampling frequency as 8 MHzk/16.                         */
        /*****************************************************************************************************************/
        SYSCR1.BYTE    = 0x40 ;

        /*****************************************************************************************************************/
        /*   Sets SYSCR2 :                                                                                               */
        /*        SMSEL     = 1 Enters sub-sleep mode                                                                    */
        /*        LSON      = 1/0                                                                                        */
        /*        DTON      = 0 Disables direct transition during sleep instruction execution                            */
        /*        MA2:0     = 000 Selects the main clock operating frequency (Specified by argument frequency)           */
        /*        SA1:0     = 00 Selects the sub-clock operating frequency (Specified by argument frequency)             */
        /*****************************************************************************************************************/
        if (return_mode == ACTIVE_MODE){
            SYSCR2.BYTE    = 0x80 |  frequency ;
        }
        else{
            SYSCR2.BYTE    = 0xC0 |  frequency ;
        }

        /*****************************************************************************************************************/
        /*   Makes the device sleep and enters sleep mode.                                                               */
        /*****************************************************************************************************************/
        sleep() ;
        /* ============================================================================================================= */
        /* === Returns here by an interrupt                                                                          === */
        /* ============================================================================================================= */

        break ;

    case STANDBY_MODE :
        /*****************************************************************************************************************/
        /*   Sets SYSCR1 :                                                                                               */
        /*        SSBSY     = 1 Makes a transition from sleep mode to standby mode                                       */
        /*        STS2:0    = 100 specifies the wait time for clock stabilization from standby to sleep as 8.2 ms.       */
        /*        NESEL     = 0 specifies the noise elimination sampling frequency as 8 MHzk/16.                         */
        /*****************************************************************************************************************/
        SYSCR1.BYTE    = 0xC0 ;
```

```
                /*****************************************************************************************************************/
                /*   Sets SYSCR2 :                                                                                               */
                /*        SMSEL     = x                                                                                          */
                /*        LSON      = x                                                                                          */
                /*        DTON      = 0 Disables direct transition during sleep instruction execution                            */
                /*        MA2:0     = 000 Selects the main clock operating frequency (Specified by argument frequency)           */
                /*        SA1:0     = 00 Selects the sub-clock operating frequency (Specified by argument frequency)             */
                /*****************************************************************************************************************/
                SYSCR2.BYTE    = 0x00 |  frequency ;

                /*****************************************************************************************************************/
                /*   Makes the device sleep and enters sleep mode.                                                               */
                /*****************************************************************************************************************/
                sleep() ;
                /* ============================================================================================================= */
                /* === Returns here by an interrupt                                                                          === */
                /* ============================================================================================================= */

                break ;


        }
}

/*****************************************************************************************************************************/
/*   1. Module name: com_frequency_change                                                                                        */
/*   2. Function overview: Performs a mode transition                                                                            */
/*   Parameter combination                                                                                                       */
/*   return_mode              frequency                                                                                          */
/*   ACTIVE_MODE              OSC_PER_1/8/16/32/64                                                                               */
/*   SUB_ACTIVE_MODE          OSC_PER_W8/4/2                                                                                     */
/*                                                                                                                               */
/*****************************************************************************************************************************/
void com_frequency_change (int return_mode , int frequency    )
{
     int i , j ;
     unsigned int    h8_addr;

     switch (return_mode){
          case ACTIVE_MODE :

                /*****************************************************************************************************************/
                /*   Sets SYSCR1 :                                                                                               */
                /*        SSBSY     = 0                                                                                          */
                /*        STS2:0    = 100 specifies the wait time for clock stabilization from standby to sleep as 8.2 ms.       */
                /*        NESEL     = 0 specifies the noise elimination sampling frequency as 8 MHzk/16.                         */
                /*****************************************************************************************************************/
                SYSCR1.BYTE    = 0x40 ;
```

```
            /*****************************************************************************************************************/
            /*   Sets SYSCR2 :                                                                                               */
            /*        SMSEL     = 0                                                                                          */
            /*        LSON      = 0                                                                                          */
            /*        DTON      = 1 Specifies a direct transition during sleep instruction execution                         */
            /*        MA2:0     = 000 Selects the main clock operating frequency (Specified by argument frequency)           */
            /*        SA1:0     = 00 Selects the sub-clock operating frequency (Specified by argument frequency)             */
            /*****************************************************************************************************************/
            SYSCR2.BYTE    = 0x20 |  frequency ;

            break ;

        case SUB_ACTIVE_MODE :

            /*****************************************************************************************************************/
            /*   Sets SYSCR1 :                                                                                               */
            /*        SSBSY     = 0                                                                                          */
            /*        STS2:0    = 100 specifies the wait time for clock stabilization from standby to sleep as 8.2 ms.       */
            /*        NESEL     = 0 specifies the noise elimination sampling frequency as 8 MHzk/16.                         */
            /*****************************************************************************************************************/
            SYSCR1.BYTE    = 0x40 ;

            /*****************************************************************************************************************/
            /*   Sets SYSCR2 :                                                                                               */
            /*        SMSEL     = 0 Not used                                                                                 */
            /*        LSON      = 1 Low-speed flag                                                                           */
            /*        DTON      = 1 Specifies a direct transition during sleep instruction execution                         */
            /*        MA2:0     = 000 Selects the main clock operating frequency (Specified by argument frequency)           */
            /*        SA1:0     = 00 Selects the sub-clock operating frequency (Specified by argument frequency)             */
            /*****************************************************************************************************************/
            SYSCR2.BYTE    = 0x60 |  frequency ;

            break ;

    }

    /* Enables direct transition interrupts                                                                                  */
    IENR1.BIT.IENDT     = 1 ;

    /*****************************************************************************************************************/
    /*   Makes the device sleep and generates a direct transition interrupt.                                         */
    /*   Prepare the interrupt vector routine for direct transition .                                                */
    /*****************************************************************************************************************/
    sleep() ;

    /* ============================================================================================================= */
    /* === A direct transition interrupt occurs and the frequency switches to the specified frequency.          === */
    /* ============================================================================================================= */

    /*****************************************************************************************************************/
    /*   Cancels interrupt disable                                                                                   */
    /*****************************************************************************************************************/

    /* Disables direct transition interrupts                                                                                 */
    IENR1.BIT.IENDT     = 0 ;

    /* Resets direct transitions                                                                                             */
    SYSCR2.BIT.DTON     = 0 ;

}
```

4. Reference Documents

- H8/3687 Group Hardware Manual (published by Renesas Technology Corp.)

Website and Support

Renesas Technology Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/inquiry
csc@renesas.com

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Sep.29.03	—	First edition issued
2.00	Sep.01.06	1, 3	New addition and content correction

Keep safety first in your circuit designs!

1. Renesas Technology Corp. puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corp. product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corp. or a third party.
2. Renesas Technology Corp. assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors.
   Renesas Technology Corp. assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corp. by various means, including the Renesas Technology Corp. Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corp. is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corp. for further details on these materials or the products contained therein.