

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

EEPROM Back-Up Processing upon Detecting Low Voltage

Introduction

An internal low-voltage detection circuit is used to back up data stored in RAM into EEPROM.

Target Device

H8/3687G

Contents

1. Specifications	2
2. Description of Functions	3
3. Description of Operation	5
4. Description of Software	8
5. Flowcharts	15
6. Program Listing	22

1. Specifications

1. An internal low-voltage detection circuit is used, and the operating state is changed.
2. In active mode, the LED is lit according to the blinking interval data read from the external EEPROM.
3. The LED blinking interval is changed through the IRQ1 switch, and the blinking interval data is stored in RAM.
4. While in active mode, when the voltage falls to 3.7 V or lower, the blinking interval data in RAM is written to the external EEPROM, and a transition to standby mode is made.
5. If, while in standby mode, the voltage rises to 4.0 V or higher, the system is returned to active mode.
6. When the voltage falls to 2.3 V or below, an internal reset signal is generated.
7. A connection example for this task is shown in figure 1.1.

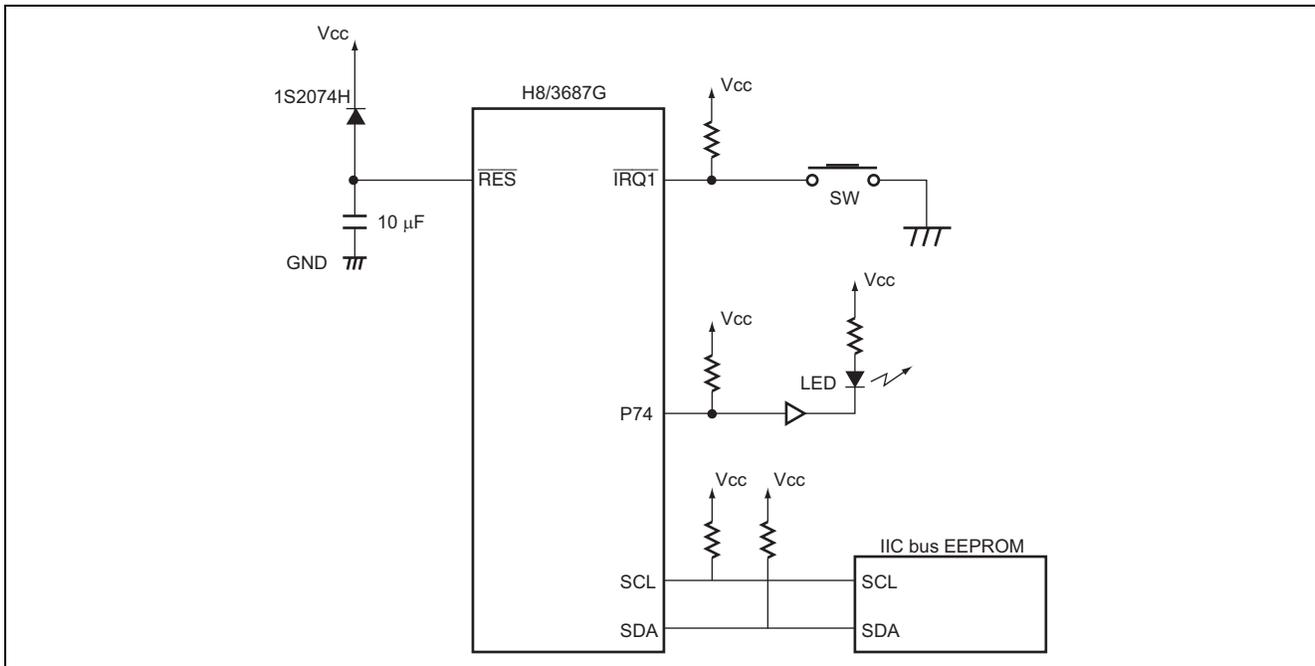


Figure 1.1 Connection example for this task

2. Description of Functions

In this sample task, the optional internal low-voltage detection circuit is used to control the operating state at low voltages. A block diagram of the low-voltage detection circuit is shown in figure 2.1. Below, the block diagram of the low-voltage detection circuit is described.

- System clock (ϕ) is a 16 MHz clock which serves as the reference clock for operation of the CPU and peripheral functions.
- Prescaler S (PSS) is functions as a 13-bit counter with ϕ as an input, counting up one each cycle.
- Low-voltage detection control register (LVDCR) is controls the low-voltage detection circuit. In this sample task, the low-voltage detection circuit is used to generate an IRQ0 interrupt when the voltage rises or falls, and sets the reset detection voltage to 2.3 V.
- Low-voltage detection status register (LVDSR) is flags indicating whether the power supply voltage has risen or fallen from a constant voltage.

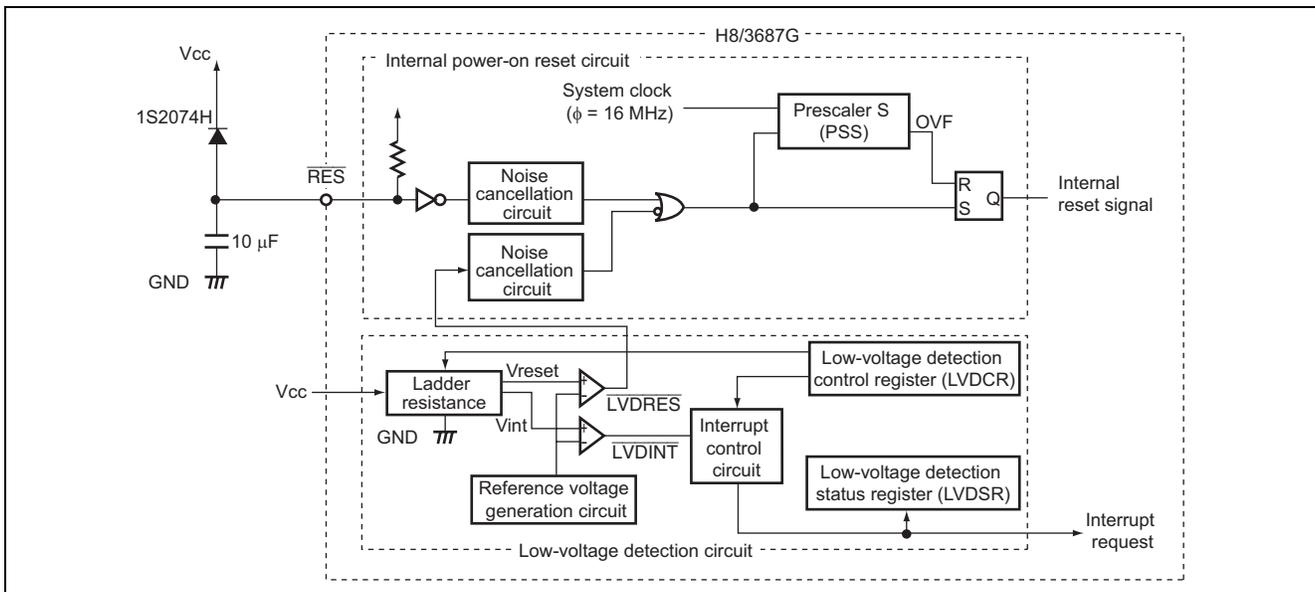


Figure 2.1 Block diagram of the low-voltage detection circuit

A standard IIC bus interface format (EEPROM byte write) is shown in figure 2.2.

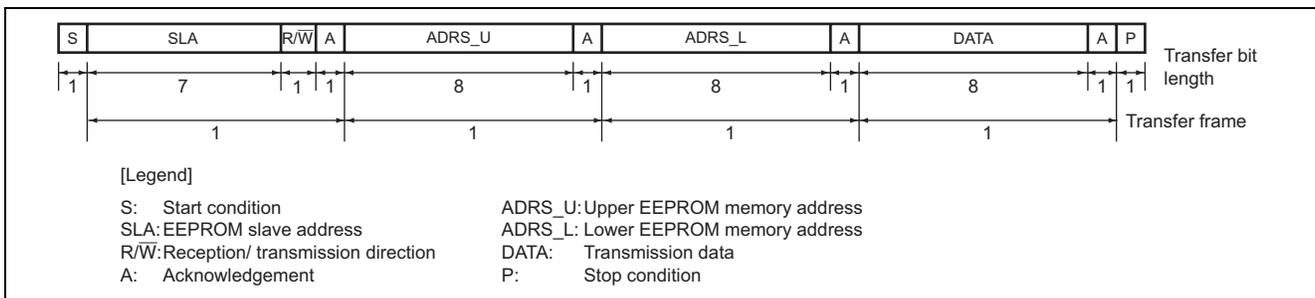


Figure 2.2 IIC bus interface format

A connection example of the IIC bus EEPROM is shown in figure 2.3.

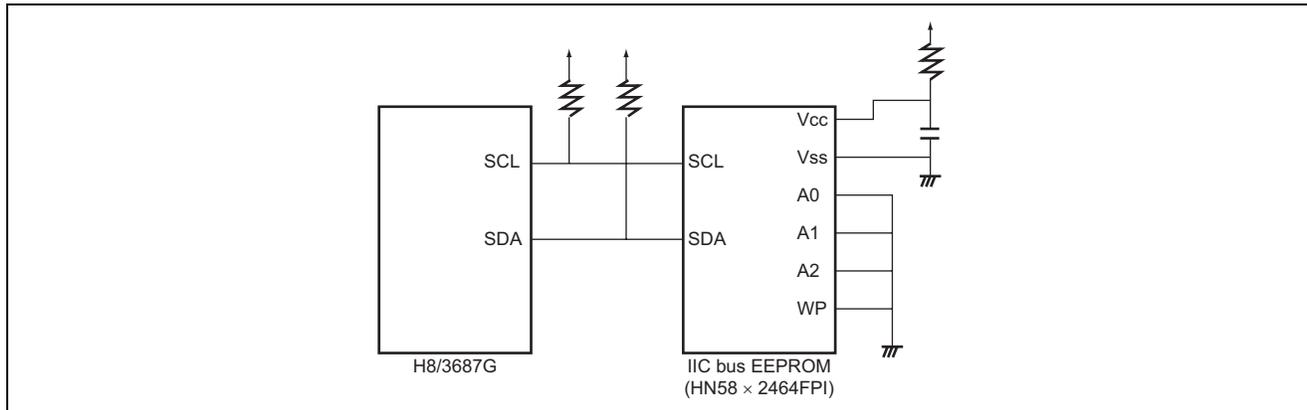


Figure 2.3 IIC Bus EEPROM Connection Example

- Function allocations in this sample task are shown in table 2.1. Functions are allocated as shown in table 2.1, and EEPROM back-up processing is performed upon low voltage detection.

Table 2.1 Function allocations

Function	Function allocation
PSS	A 13-bit counter with the system clock used as an input signal
LVDCR	Controls operation/cancellation of the low-voltage detection circuit
LVDSR	Flags indicating whether the power supply voltage has risen or fallen from a certain constant voltage
PDR7	In order to confirm the operating mode, an LED connected to pin P74 is lit
PCR7	Pin P74 is set to an output pin
SYSCR1	Controls low-power consumption modes
SYSCR2	Controls low-power consumption modes
IRQ1	LED blinking interval modification switch
SCL	Contact pin to EEPROM
SDA	Contact pin to EEPROM

- Specifications of the IIC bus EEPROM used in this sample task are described below.

The IIC bus EEPROM is a two-wired serial interface EEPROM (electrically erasable/programmable ROM). In this sample task, 64-kbit EEPROM (HN58X2464FPI) manufactured by Renesas Technology Corp. is used. The features of the EEPROM used in this sample task are shown in table 2.2.

Table 2.2 64-kbit EEPROM manufactured by Renesas Technology Corp. (HN58X2464FPI)

Single power supply	1.8 to 5.5 V
Two-wired serial interface	IIC bus interface
Operating frequency	400 kHz
Current consumption	At standby 3 μ A (max.)
	At reading 1 mA (max.)
	At writing 3 mA (max.)
Page rewriting	Page size: 32 bytes
Rewrite time	10 ms (2.7 to 5.5 V or higher)/15ms (1.8 to 2.7 V)
Number of times for rewriting	10^5 (during page rewriting)

3. Description of Operation

1. Timing Charts

Figure 3.1 shows the procedure for setting and canceling LVDI, and transitions to standby mode triggered by low-voltage detection interrupts.

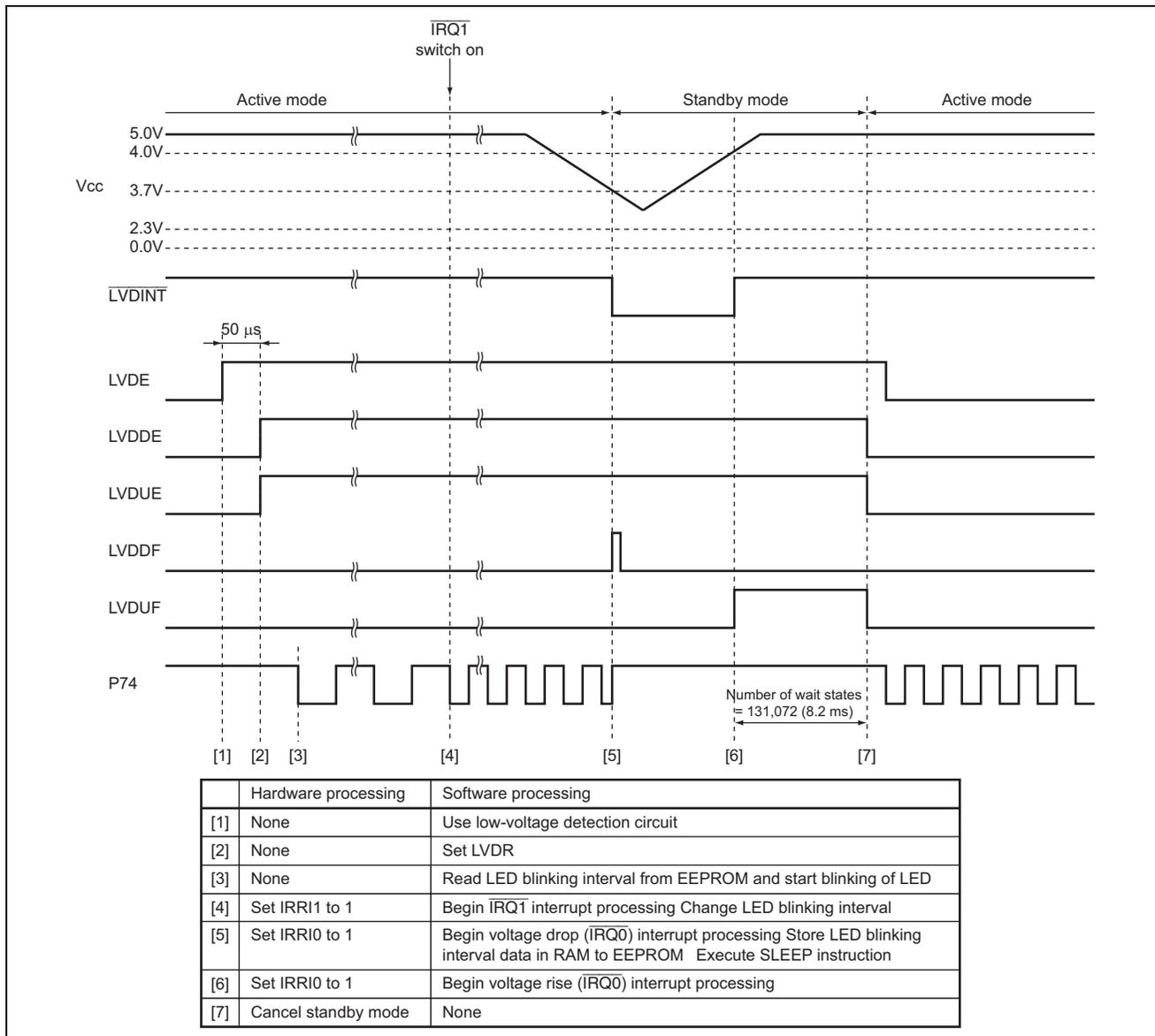


Figure 3.1 Description of operation (1)

Figure 3.2 illustrates a transition to standby mode triggered by a low-voltage detection interrupt, and reset operation on low voltage detection.

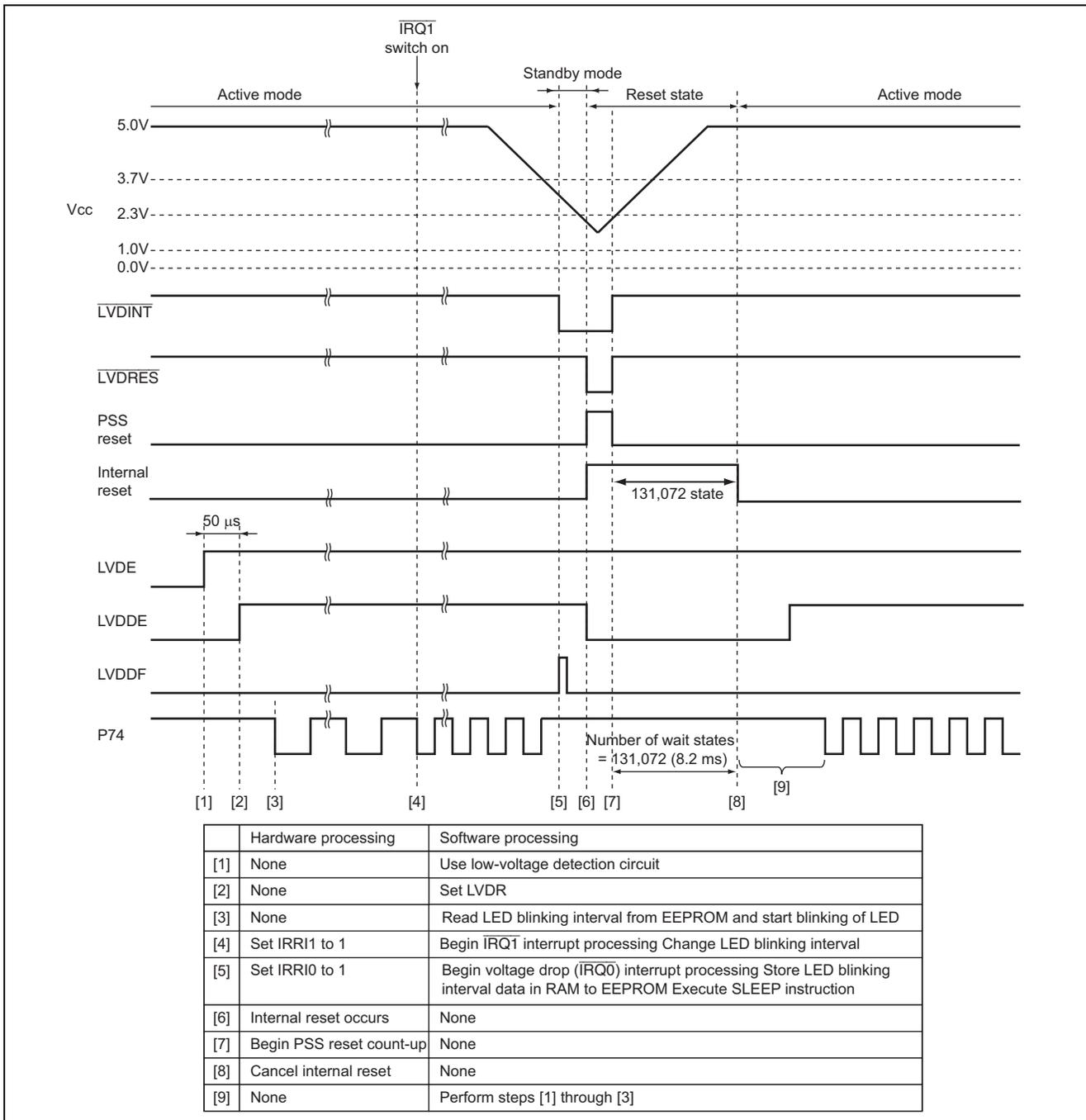


Figure 3.2 Description of operation (2)

2. EEPROM Write-in Time

In this sample task, data (4 bytes) in RAM is backed up in EEPROM at low voltage. Processing time from the beginning of interrupts to the end of write-in to EEPROM at low voltage is shown in table 3.1. The power supply voltage should be kept at the operation guaranteed lower limit voltage (3.0 V) or higher until back-up processing completes.

Table 3.1 EEPROM backup processing time

Data size to be written	Processing time
4 bytes	330 μ s

4. Description of Software

4.1 Description of modules

Modules in this sample task are listed in table 4.1.

Table 4.1 Description of modules

Module name	Label name	Function
Main routine	main	Set low-voltage detection circuit, enable interrupts, read EEPROM data, control LED (P74), and judge switch connected to IRQ0
Low-voltage detection interrupt	irq0int	IRQ0 interrupt processing Clear LVD flag, set lpcnt to 0 or 1, and EEPROM back-up processing
Switch on	irq1int	IRQ1 interrupt processing Set lpcnt to 2
EEPROM access routine	Read_n_EEPROM	Read n bytes from EEPROM
	Write_n_byte	Write n bytes to EEPROM
	Write_data_EEPROM	Write data to EEPROM
	Write_data_End_EEPROM	Write last data to EEPROM
	Set_adrs_EEPROM	Specify EEPROM address
	Recv_datan_EEPROM	Receive n-byte data using IIC

4.2 Description of arguments

Arguments for respective functions are described below.

- Read_n_EEPROM function

Argument	Function	Data length
adrs	Specify read address	2 bytes
*rd_ptr	Read data storage address	1 byte
no	Read data length	2 bytes

- Write_n_EEPROM function

Argument	Function	Data length
adrs	Specifies write address	2 bytes
*wr_ptr	Write data storage address	1 byte
no	Write data length	2 bytes

- Write_data_EEPROM function

Argument	Function	Data length
wr_data	Write data	1 byte

- Write_data_End_EEPROM function

Argument	Function	Data length
wr_data	Write data	1 byte

- Set_adrs_EEPROM function

Argument	Function	Data length
adrs	Write/read address	2 bytes

Recv_datan_EEPROM function

Argument	Function	Data length
*rd_ptr	Read data storage address	1 byte
no	Read byte length 1 to 64	2 bytes

4.3 Description of Internal Registers Used

Internal registers used in this sample task are indicated below.

- LVDCR Low-voltage detection control register Address: 0xF730

Bit	Bit name	Setting	Function
7	LVDE	1	LVD enable LVDE = 0: Low-voltage detection circuit is not used (standby state) LVDE = 1: Low-voltage detection circuit is used
3	LVDSSEL	0	LVDR detection level selection LVDSSEL = 0: Sets reset detection voltage to 2.3 V LVDSSEL = 1: Sets reset detection voltage to 3.6 V
2	LVDRRE	1	LVDR enable LVDRRE = 0: Disables reset by LVDR LVDRRE = 1: Enables reset by LVDR
1	LVDDDE	1	LVDR enable LVDDDE = 0: Disables interrupt requests when voltage falls LVDDDE = 1: Enables interrupt requests when voltage falls
0	LVDDUE	1	LVDR enable LVDDUE = 0: Disables interrupt requests when voltage rises LVDDUE = 1: Enables interrupt requests when voltage rises

- LVDSR Low-voltage detection status register Address: 0xF731

Bit	Bit name	Setting	Function
1	LVDDDF	0	LVD power supply voltage drop flag LVDDDF = 0: Cleared to 0 state LVDDDF = 1: Power supply voltage has fallen to 3.7 V or below
0	LVDDUF	0	LVD power supply voltage rise flag LVDDUF = 0: Cleared to 0 state LVDDUF = 1: While the LVDDUE flag of LVDCR is set to 1, the power supply voltage has fallen to 3.7 V or below, and risen again to 4.0 V or above before falling to Vreset (2.3 V) or below

- PDR7 Port data register 7 Address: 0xFFDA

Bit	Bit name	Setting	Function
4	P74	0	Port data register 74 P74 = 0: Pin P74 output level Low P74 = 1: Pin P74 output level High

- PMR1 Port mode register 1 Address: 0xFFE0

Bit	Bit name	Setting	Function
5	IRQ1	1	Selects function of pin P15/IRQ1 IRQ1 = 0: Sets pin P15/IRQ1 to P15 I/O pin function IRQ1 = 1: Sets pin P15/IRQ1 to P15 output pin function

- PCR7 Port control register 7

Address: 0xFFEA

Bit	Bit name	Setting	Function
4	PCR74	0	Port control register 74 PCR74 = 0: Sets pin P74 to P74 input pin function PCR74 = 1: Sets pin P74 to P74 output pin function

- SYSCR1 System control register 1

Address: 0xFFFF0

Bit	Bit name	Setting	Function
7	SSBY	1	Software standby DTON = 0, SSBY = 1: After executing SLEEP instruction in active mode, makes transition to standby mode
6	STS2	STS2 = 1	Standby timer select 2 to 0
5	STS1	STS1 = 0	When STS2 = 1, STS1 = 0 and STS0 = 0, the number of wait states is set to
4	STS0	STS0 = 0	131,072 states

- SYSCR2 System control register 2

Address: 0xFFFF1

Bit	Bit name	Setting	Function
5	DTON	0	Direct transfer on flag DTON = 0, SSBY = 1: After executing SLEEP instruction in active mode, makes transition to standby mode
4	MA2	MA2 = 0	Active mode clock select 2 to 0
3	MA1	MA1 = x	MA2 = 0, MA1 = x, MA0 = x:
2	MA0	MA0 = x	Sets active mode/sleep mode operating clock to ϕ_{osc} (x: don't care)

- IEGR1 Interrupt edge select register 1

Address: 0xFFFF2

Bit	Bit name	Setting	Function
0	IEG1	1	IRQ1 edge select IEG1 = 0: Selects falling edge as IRQ1 pin input detection edge IEG1 = 1: Selects rising edge as IRQ1 pin input detection edge

- IENR1 Interrupt enable register 1

Address: 0xFFFF4

Bit	Bit name	Setting	Function
1	IEN1	1	IRQ1 interrupt request enable IEN1 = 0: Disables interrupt requests at pin IRQ1 IEN1 = 1: Enables interrupt requests at pin IRQ1

- IRR1 Interrupt flag register 1

Address: 0xFFFF6

Bit	Bit name	Setting	Function
1	IRRI1	0	IRQ1 interrupt request flag IRR1 = 0: IRQ1 pin interrupt not requested IRR1 = 1: IRQ1 pin interrupt requested
0	IRRI0	0	IRQ0 interrupt request flag IRR0 = 0: IRQ0 pin interrupt not requested IRR0 = 1: IRQ0 pin interrupt requested

• ICCR1 IIC bus control register 1

Address: 0xF748

Bit	Bit name	Setting	Function
7	ICE	1	IIC bus interface enable ICE = 0: IIC2 module enters function stop status (SCL/SDA pin functions as a port) ICE = 1: IIC2 module enters transfer enabled status (SCL/SDA pin functions as a bus drive pin)
6	RCVD	0	Receive disable RCVD = 0: Enables subsequent reception operation RCVD = 1: Disables subsequent reception operation
5	MST	0	Master/slave select MST = 0: Selects slave MST = 1: Selects master
4	TRS	0	Transmission/reception select TRS = 0: Reception mode TRS = 1: Transmission mode
3	CKS3	CKS3 = 0	Transfer clock select 3 to 0
2	CKS2	CKS2 = 0	CKS3 = 0, CKS2 = 0, CKS1 = 0, CKS0 = 1: Set transfer rate to 400 kHz when $\phi = 16$
1	CKS1	CKS1 = 0	
0	CKS0	CKS0 = 1	

• ICCR2 IIC bus control register 2

Address: 0xF749

Bit	Bit name	Setting	Function
7	BBSY	1	Bus busy BBSY = 0: IIC bus is being used BBSY = 1: IIC bus is not used
6	SCP	0	Start/stop condition issue prohibited RCVD = 0: Permits issuance RCVD = 1: Prohibits issuance
5	SDAO	1	SDA output value control SDAO = 0: Low level SDAO = 1: High level
4	SDAOP	1	SDAO write protection SDAOP = 0: Writing enabled SDAOP = 1: Writing disabled
3	SCLO	1	SCL output level monitor SCLO = 0: SCL outputs low level signal SCLO = 1: SCL outputs high level signal
1	IICRST	0	IIC controller reset IICRST = 0: Normal termination IICRST = 1: Reset

- **ICIER** IIC bus interrupt enable register Address: 0xF74B

Bit	Bit name	Setting	Function
1	ACKBR	—	Receive acknowledgment ACKBR = 0: Receive acknowledgment = 0 ACKBR = 1: Receive acknowledgment = 1
0	ACKBT	0	Transmit acknowledgement ACKBT = 0: Transmit acknowledgment = 0 ACKBT = 1: Transmit acknowledgment = 1

- **ICSR** IIC bus status register Address: 0xF74C

Bit	Bit name	Setting	Function
7	TDRE	—	Transmit data empty TDRE = 0: Cleared to 0 TDRE = 1: data is transferred from ICDRT to ICDRS
6	TEND	—	Transmit end TEND = 0: Cleared to 0 TEND = 1: Ninth SCL clock rises when TDRE is 1 in the IIC bus format
5	RDRF	—	Receive data register full RDRF = 0: Cleared to 0 RDRF = 1: Received data is transferred from ICDRS to ICDRR
3	STOP	0	Stop condition detection flag STOP = 0: Cleared to 0 STOP = 1: Stop condition is detected upon completion of frame transfer

- **ICDRT** IIC bus transmit data register Address: 0xF74E
Function: Stores transmitted data. When detecting a fact that ICDRS is available, transfers data from ICDRT to ICDRS to start data transmission.

Setting:

- **ICDRR** IIC bus receive data register Address: 0xF74F
Function: Stores received data. When 1-byte data reception is completed, transfers data from ICDRS to ICDRR to allow subsequent data reception.

Setting:

4.4 Description of RAM Used

The RAM used in this sample task is described in table 4.2.

Table 4.2 Description of RAM used

Label name	Function	Size	Used in
lpcnt	Flag to discriminate low-voltage detection states Lpcnt = 0: Returned to normal mode Lpcnt = 1: At low power voltage, backs up data in RAM to EEPROM, and transits to module standby Lpcnt = 2: IRQ1 interrupt, low-voltage detection circuit disabled	1 byte	Main routine Low-voltage detection interrupt Switch on

4.5 Module Hierarchical Diagram

The module hierarchical diagram of this sample task is shown in figure 4.1.

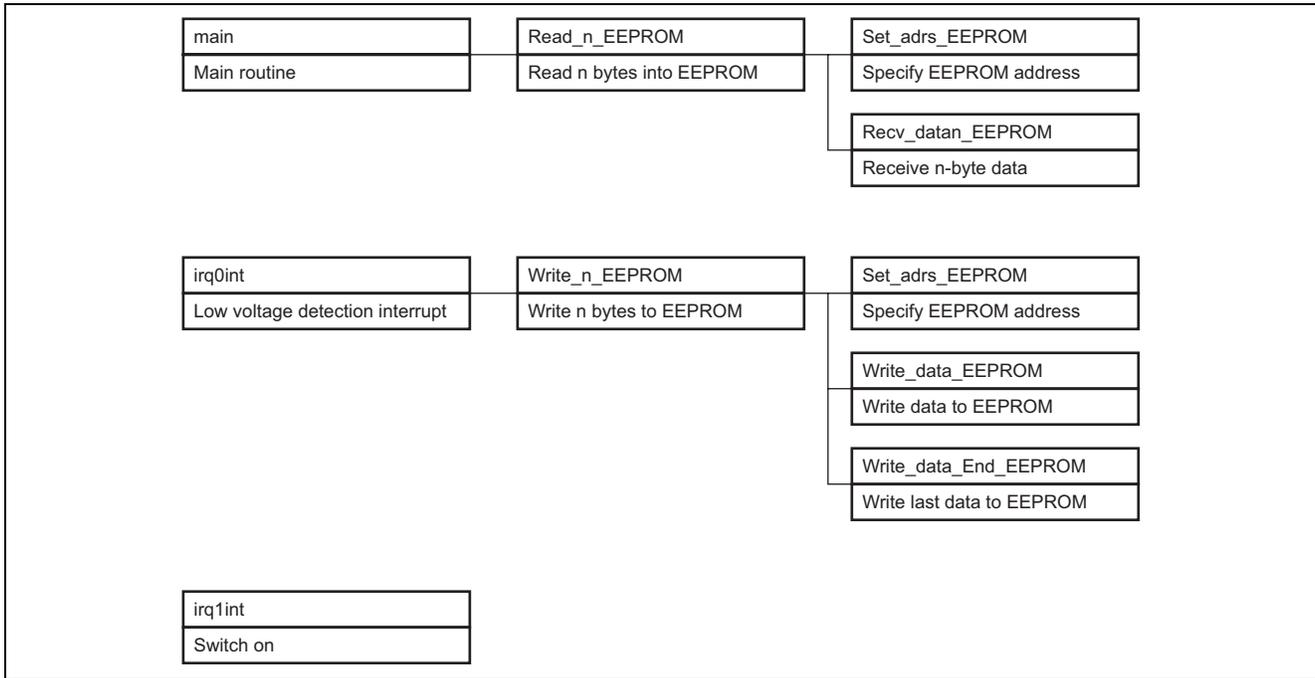
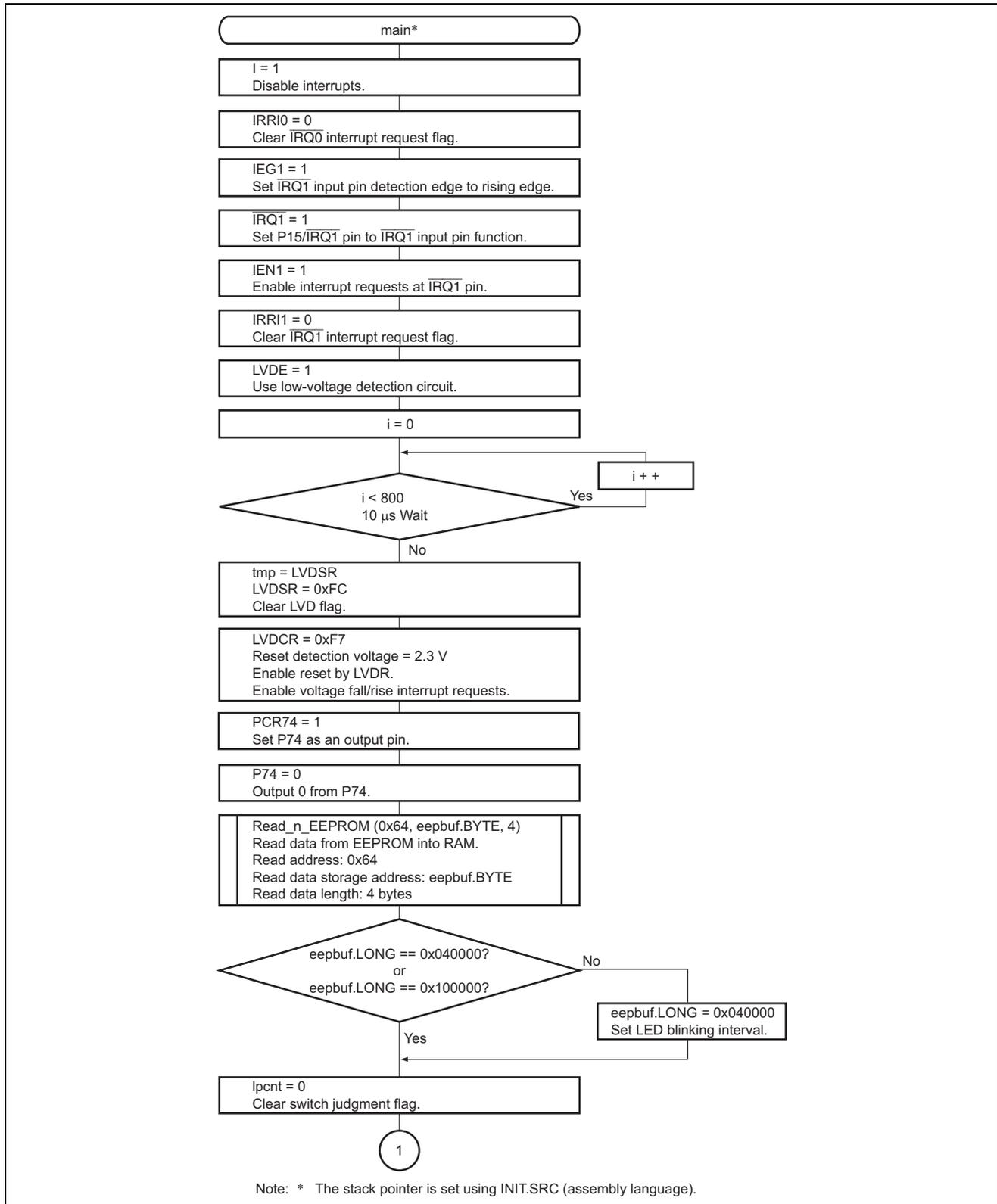


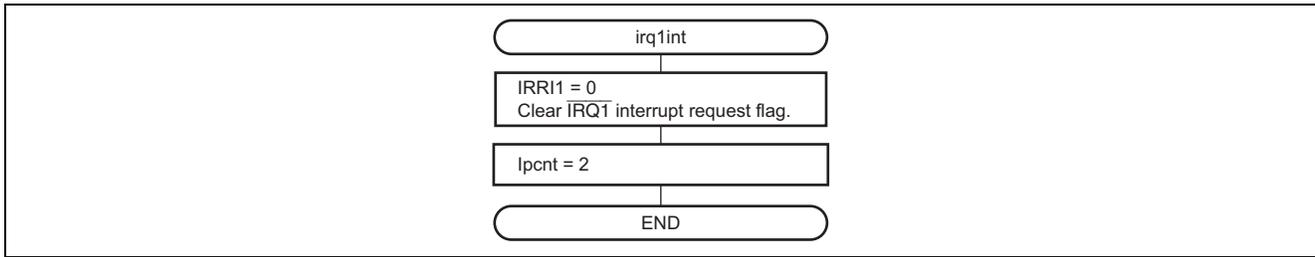
Figure 4.1 Module Hierarchical Diagram

5. Flowcharts

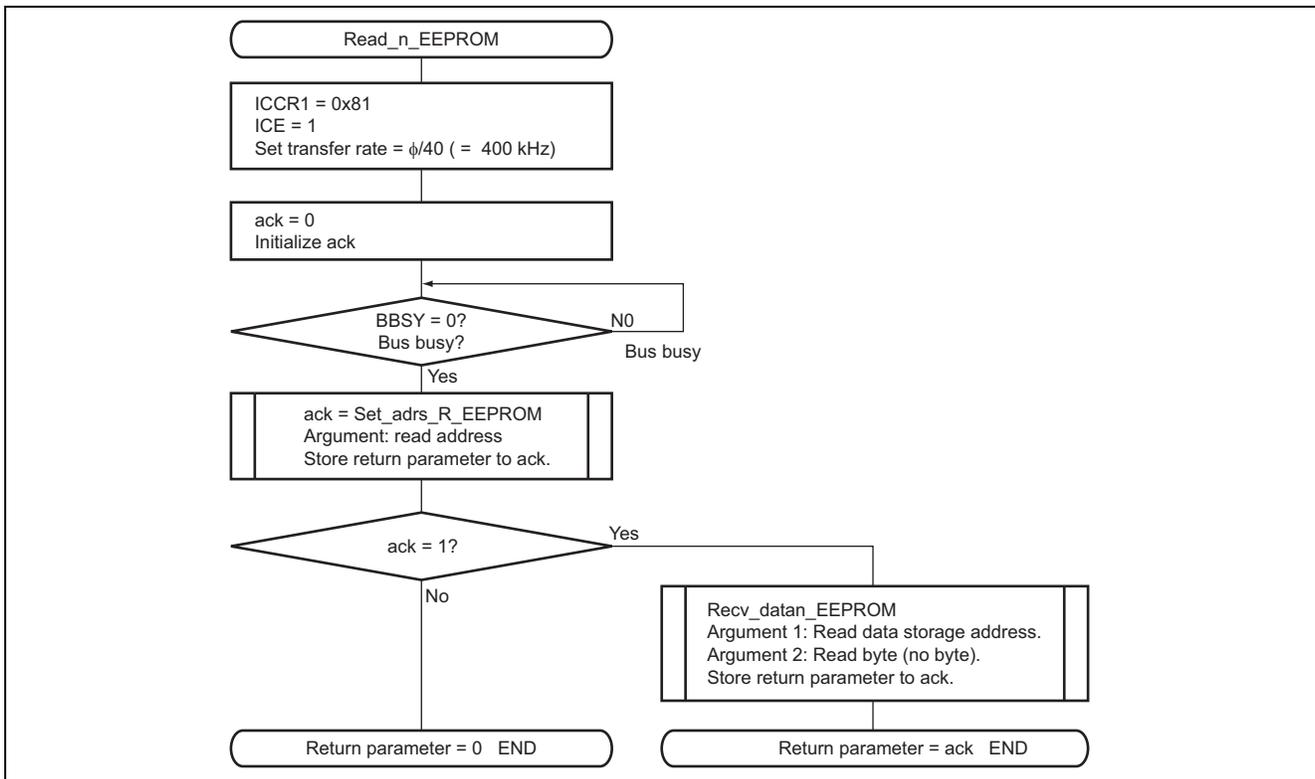
1. Main routine

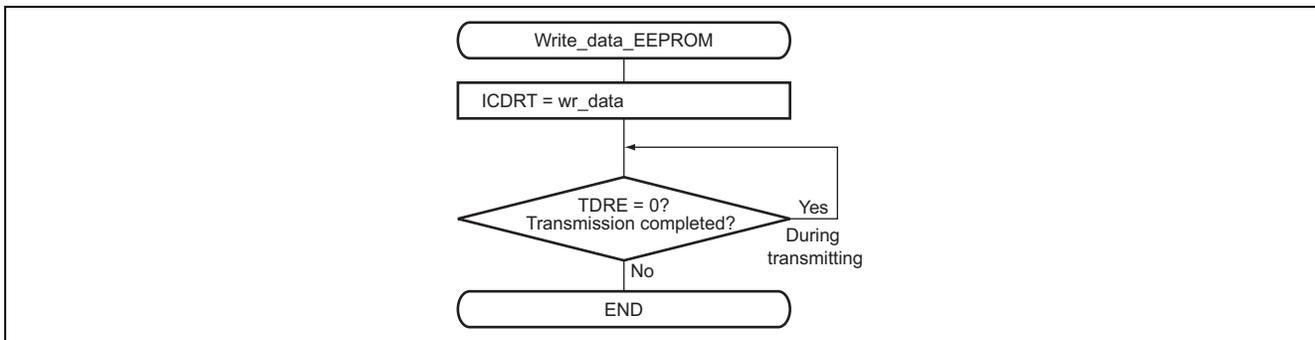
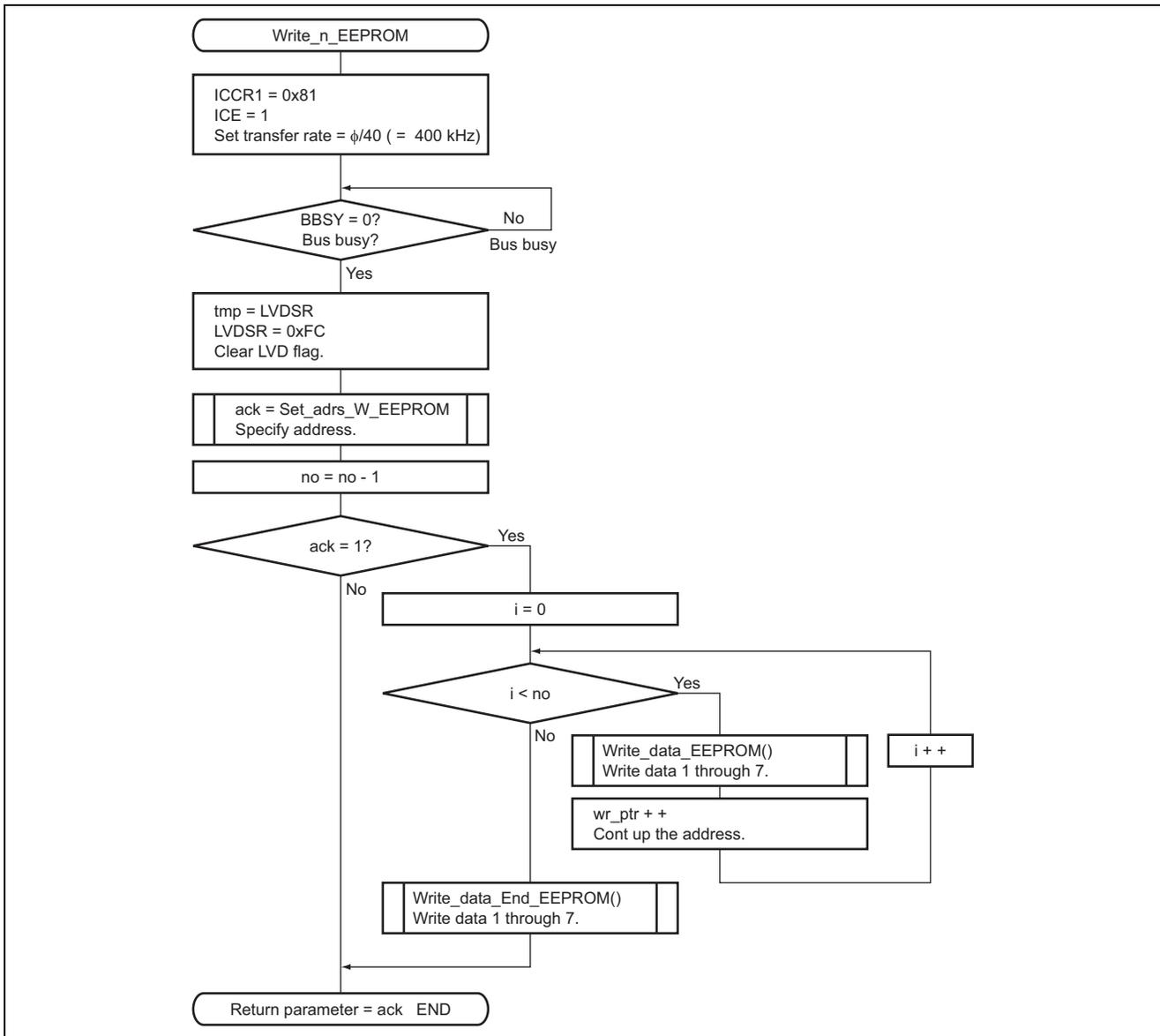


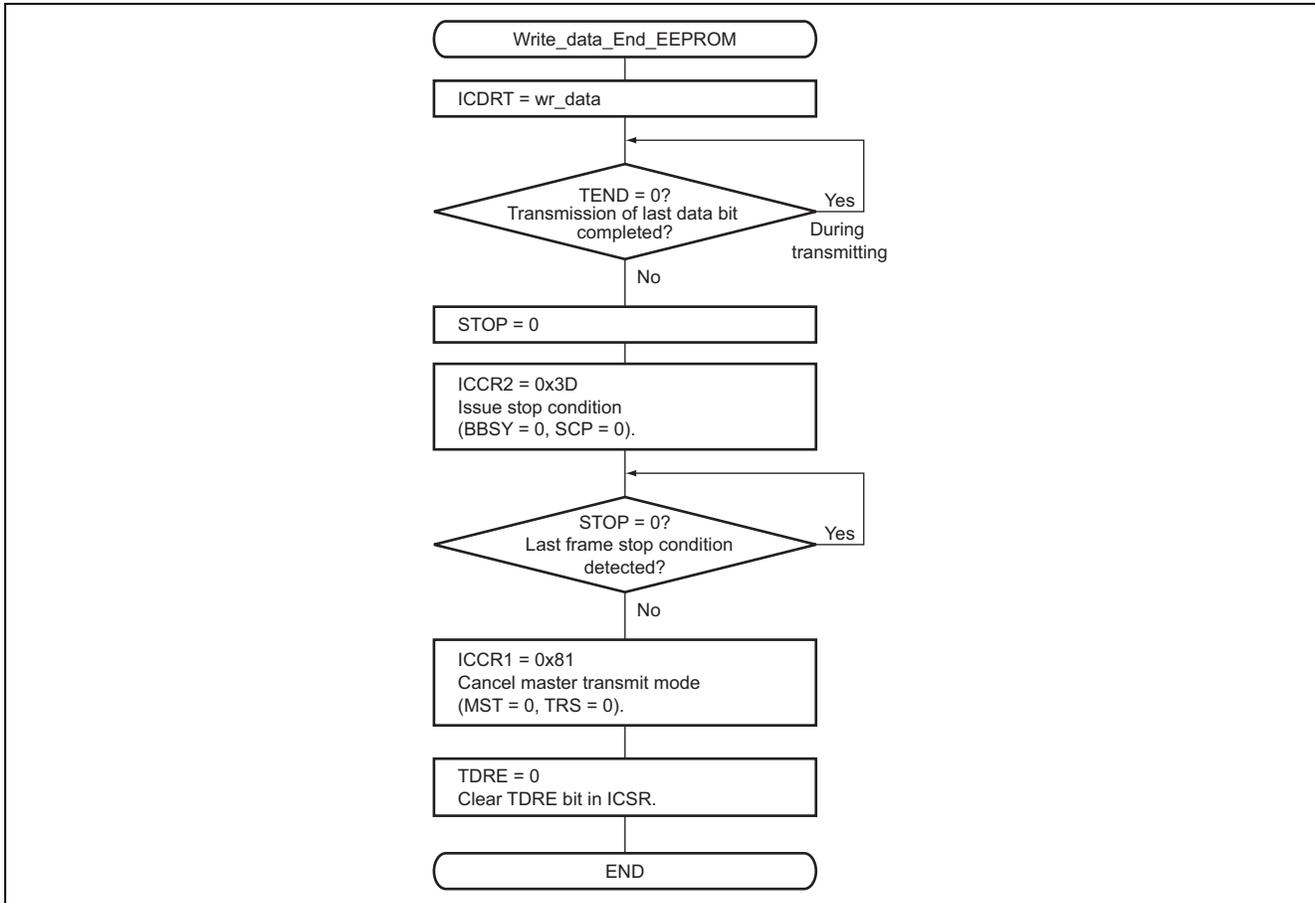
3. Switch-on

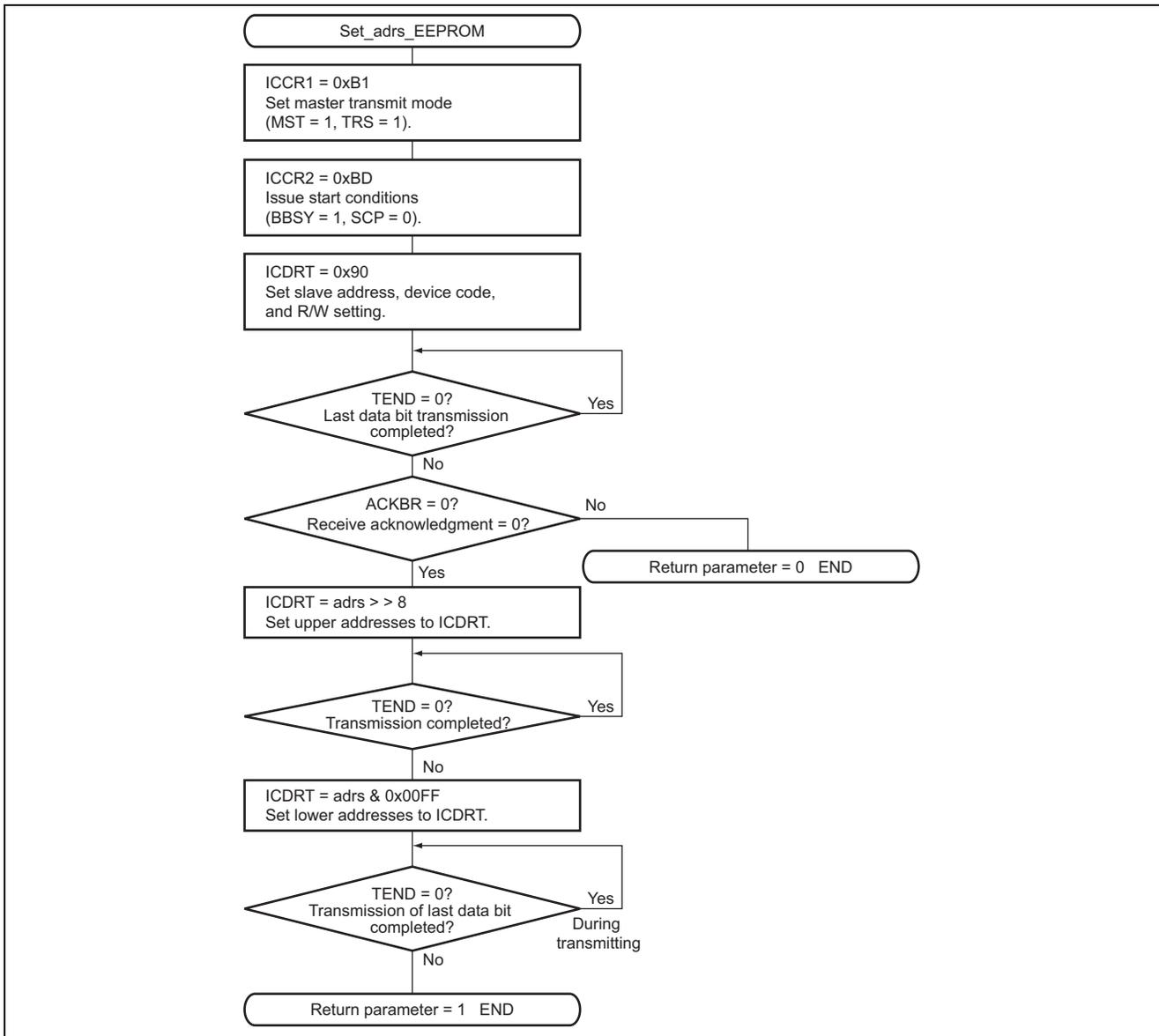


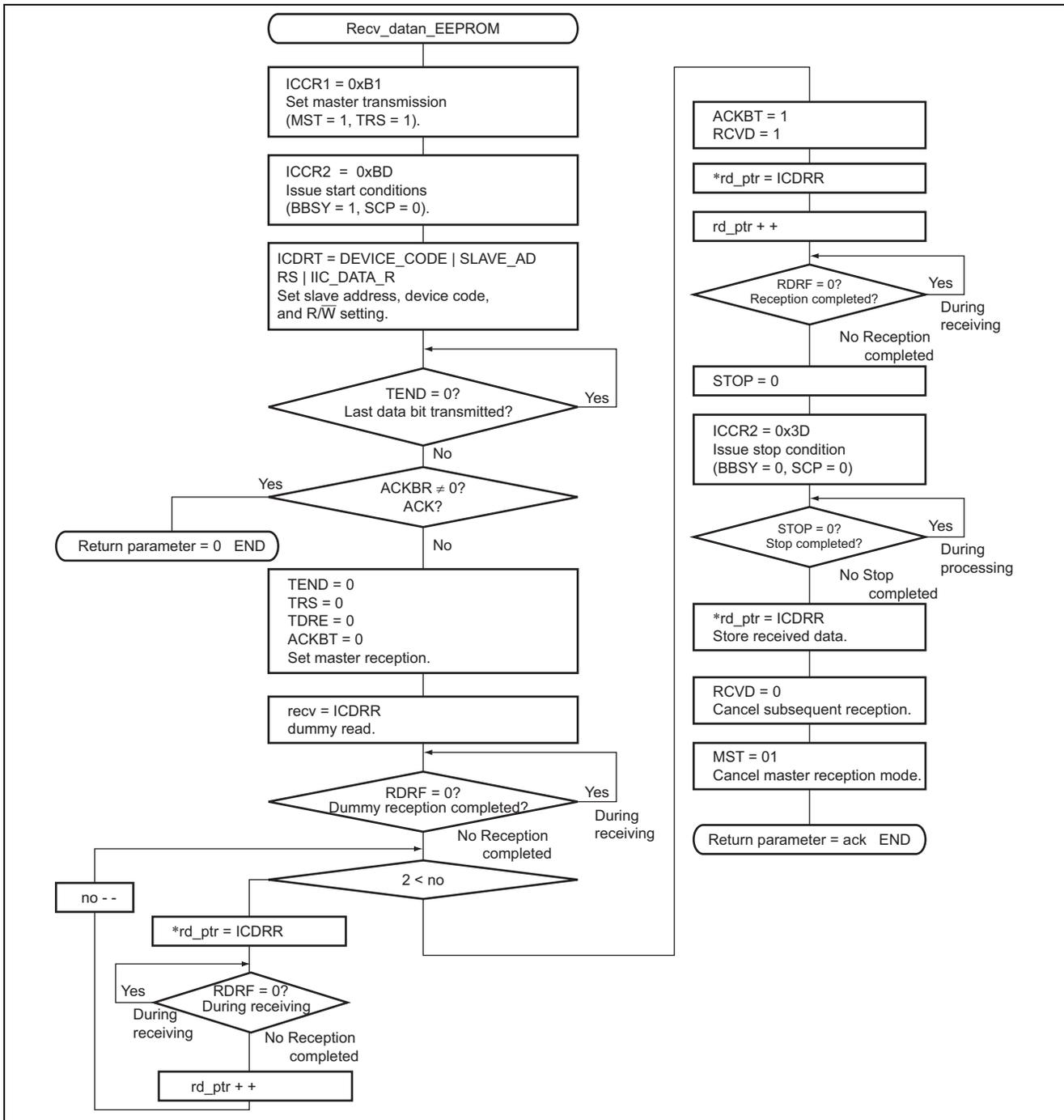
4. EEPROM Access Routine











6. Program Listing

```

/*****
/*
/* H8/300HN Series -H8/3687G-
/* Application Note
/*
/* 'Memory Backup to EEPROM by LVDI'
/*
/* Function
/* : LVD (Interrupt by lowvoltage detect)
/* : IIC Bus Interface 2 (EEPROM Access)
/*
/* External Clock : 16MHz
/* Internal Clock : 16MHz
/* Sub Clock      : 32.768kHz
/*
/*****

#include <machine.h>

/*****
/* Symbol Definition
/*****

struct BIT {
    unsigned char  b7:1;      /* bit7 */
    unsigned char  b6:1;      /* bit6 */
    unsigned char  b5:1;      /* bit5 */
    unsigned char  b4:1;      /* bit4 */
    unsigned char  b3:1;      /* bit3 */
    unsigned char  b2:1;      /* bit2 */
    unsigned char  b1:1;      /* bit1 */
    unsigned char  b0:1;      /* bit0 */
};

#define LVDCR      *(volatile unsigned char *)0xF730      /* Low-voltage-detection control Register */
#define LVDCR_BIT  (*(struct BIT *)0xF730)               /* Low-voltage-detection control Register */
#define LVDE       LVDCR_BIT.b7                          /* LVD Enable */
#define LVDSSEL    LVDCR_BIT.b3                          /* LVDI Detection Level Select */
#define LVDSRE     LVDCR_BIT.b2                          /* LVDR Enable */
#define LVDSR      *(volatile unsigned char *)0xF731      /* Low-Voltage-Detection Status Register */
#define LVDSR_BIT  (*(struct BIT *)0xF731)               /* Low-Voltage-Detection Status Register */
#define LVDDF      LVDSR_BIT.b1                          /* LVD Power-Supply Voltage Fall */
#define LVDDUF     LVDSR_BIT.b0                          /* LVD Power-Supply Voltage Rise */
#define PDR7_BIT   (*(struct BIT *)0xFFDA)               /* Port Data Register 7 */
#define P74        PDR7_BIT.b4                          /* Port Data Register 7 bit4 */
#define PMR1_BIT   (*(struct BIT *)0xFFE0)               /* Port mode Register 1 */
#define IRQ1       PMR1_BIT.b5                          /* P15/IRQ1 Pin Function Switch */
#define IRQ0       PMR1_BIT.b4                          /* P14/IRQ0 Pin Function Switch */
#define PCR7_BIT   (*(struct BIT *)0xFFEA)               /* Port Control Register 7 */
#define PCR74      PCR7_BIT.b4                          /* Port Control Register 7 bit4 */
#define SYSCR1     *(volatile unsigned char *)0xFFFD     /* System Control Register 1 */
#define SYSCR2     *(volatile unsigned char *)0xFFFD     /* System Control Register 2 */
#define IEGR1_BIT  (*(struct BIT *)0xFFF2)               /* Interrupt Edge Select Register 1 */
#define IEGR1      IEGR1_BIT.b1                          /* IRQ0 Edge Select */
#define IEGR0      IEGR1_BIT.b0                          /* IRQ0 Edge Select */
#define IENR1_BIT  (*(struct BIT *)0xFFF4)               /* Interrupt Enable Register 1 */
#define IEN1       IENR1_BIT.b1                          /* IRQ0 Interrupt Enable */

```

```

#define IEN0 IENR1_BIT.b0 /* IRQ0 Interrupt Enable */
#define IRR1_BIT (*(struct BIT *)0xFFFF6) /* Interrupt Request Register 1 */
#define IRR1 IRR1_BIT.b1 /* IRQ1 Interrupt Request Flag */
#define IRR10 IRR1_BIT.b0 /* IRQ0 Interrupt Request Flag */

#pragma interrupt (irq0int)
#pragma interrupt (irq1int)
/*****
/* Function define */
/*****
extern void INIT ( void ); /* SP Set */
extern unsigned char Write_byte_EEPROM
( unsigned short adrs , unsigned char wr_data );
extern unsigned char Read_byte_EEPROM ( unsigned short adrs );
void main ( void );
void irq0int ( void );
void irq1int ( void );
void sleep ( void );

/*****
/* RAM define */
/*****
volatile unsigned char lpcnt;
union addt{
    unsigned long LONG;
    unsigned char BYTE[4];
}eeplib;

/*****
/* Vector Address */
/*****
#pragma section V1 /* VECTOR SECTOIN SET */
void (*const VEC_TBL1[])(void) = {
    INIT /* 00 Reset */
};
#pragma section V2 /* VECTOR SECTOIN SET */
void (*const VEC_TBL2[])(void) = {
    irq0int /* 1C IRQ0 Interrupt */
};
#pragma section V3 /* VECTOR SECTOIN SET */
void (*const VEC_TBL3[])(void) = {
    irq1int /* 1E IRQ1 Interrupt */
};

#pragma section /* P */

```

```

/*****
/* Main Program
/*****
void main ( void )
{
    unsigned long i;
    unsigned char tmp;

    set_imask_ccr(1); /* Interrupt Disable */

    IRRIO = 0; /* Clear IRRIO */
    IEG1 = 1; /* Rising Edge of IRQ1 Input */
    IRQ1 = 1; /* Initialize IRQ1 Terminal Input */
    IEN1 = 1; /* IRQ1 Interrupt Enable */
    IRR11 = 0; /* Clear IRR11 */

    LVDE = 1; /* LVD Enable */
    for(i=0; i<800; i++); /* 50us Wait */
    tmp = LVDSR;
    LVDSR = 0xFC; /* Clear LVDDF, LVDUF */
    LVDCR = 0xF7; /* Set LVDRE, LVDDE, LVDUE */

    PCR74 = 1; /* P74 Output Pin */
    P74 = 0; /* P74 is Low */

    Read_n_EEPROM(0x64, eepbuf.BYTE, 4); /* Read EEPROM DATA -> eepbuf */
    if(!((eepbuf.LONG==0x040000)|(eepbuf.LONG==0x100000)))
        eepbuf.LONG = 0x040000; /* Set eepbuf */

    lpcnt = 0; /* Clear lpcnt */
    set_imask_ccr(0); /* Interrupt Enable */
    while(1){
        if(lpcnt == 2){ /* IRQ1 SW On? */
            if(eepbuf.LONG == 0x100000) /* Change eepbuf data */
                eepbuf.LONG = 0x040000;
            else
                eepbuf.LONG = 0x100000;
            lpcnt = 0; /* Clear lpcnt */
        }

        for(i=0; i<eepbuf.LONG; i++); /* Wait Loop */
        P74 = ~P74;

        if(lpcnt == 1){ /* Lowvoltage detect ? */
            SYSCR1 = 0xC0;
            SYSCR2 = 0x0C;
            lpcnt = 0; /* Clear lpcnt */
            sleep(); /* Transition to Standby Mode */
        }
    }
}

```

```

/*****
/*  IRQ0 Interrupt
/*****
void irq0int ( void )
{
    unsigned char  tmp;

    IRRIO = 0;                               /* Clear IRRIO                */
    if(LVDDF == 1){                          /* LVD Power-Supply Voltage Fall? */
        Write_n_EEPROM(0x64, eepbuf.BYTE, 4); /* Memory Backup to EEPROM      */
        lpcnt = 1;                            /* Set Standby Mode flag       */
    }
    else if(LVDUF == 1)                      /* LVD Power-Supply Voltage Rise? */
        lpcnt = 0;                            /* IRQ0 Interrupt / Active Mode */

    tmp = LVDSR;
    LVDSR = 0xFC;                            /* Clear LVDDF,LVDUF          */
}

/*****
/*  IRQ1 Interrupt
/*****
void irq1int ( void )
{
    IRRI1 = 0;                               /* Clear IRRI1                */
    lpcnt = 2;                               /* Set lpcnt                  */
}

/*****
/*
/*  IIC2 EEPROM read/write
/*      H8/3687,H8/3694 EEPROM Function
/*
/*****
#include <machine.h>
#include "H8_3687_IIC2.H"

/*****
/*  Symbol Definition
/*****
#define  DEVICE_CODE 0xA0                    /* EEPROM DEVICE CODE:1010    */
#define  SLAVE_ADRS  0x00                    /* SLAVE ADRS:0               */
#define  IIC_DATA_W  0x00                    /* WRITE_DATA                  */
#define  IIC_DATA_R  0x01                    /* READ_DATA                   */

/*****
/*  Function define
/*****
unsigned char Set_adrs_EEPROM ( unsigned short adrs );
void Write_data_EEPROM ( unsigned char wr_data );
void Write_data_End_EEPROM ( unsigned char wr_data );
unsigned char Recv_datan_EEPROM ( unsigned char *rd_ptr , unsigned short no );

```

```

/*****
/*  Main Program
/*  Read_n_EEPROM    (n:2-512 byte)
/*      argument1:read address(unsigned short)
/*      argument2:read data address(unsigned char *)
/*      argument3:read data number(unsigned short)
/*      return: 1:OK/0:NG EEPROM NOACK    (unsigned char)
/*****
unsigned char Read_n_EEPROM( unsigned short adrs , unsigned char *rd_ptr , unsigned short no )
{
    unsigned char    ack;

    IIC2.ICCR1.BYTE = 0x81;                /* Initialize (ICE=1,CKS=0001)    */

    ack = 0;
    while(IIC2.ICCR2.BIT.BBSY != 0);      /* Bus busy?                      */

    ack = Set_adrs_EEPROM(adrs);          /* Set address (dummy write)     */
    if(ack == 1)
        ack = Recv_datan_EEPROM(rd_ptr,no); /* Data read n byte              */

    return(ack);
}

/*****
/*  Write_page_EEPROM    (8byte)
/*      argument1:write address(unsigned short)
/*      argument2:write data address(unsigned char *)
/*      argument3:write data number(unsigned short)
/*      return: 1:OK/0:NG EEPROM NOACK    (unsigned char)
/*****
unsigned char Write_n_EEPROM( unsigned short adrs , unsigned char *wr_ptr , unsigned short no )
{
    unsigned short    i;
    unsigned char    ack;

    IIC2.ICCR1.BYTE = 0x81;                /* Initialize (ICE=1,CKS=0001)    */

    while(IIC2.ICCR2.BIT.BBSY != 0);      /* Bus busy?                      */

    ack = Set_adrs_EEPROM(adrs);          /* Set address (dummy write)     */
    no = no-1;
    if(ack == 1){
        for(i = 0; i < no; i++){
            Write_data_EEPROM(*wr_ptr);    /* Data writel-7                  */
            wr_ptr++;
        }
        Write_data_End_EEPROM(*wr_ptr);    /* Data write8 (last data)       */
    }

    return(ack);
}

```

```

/*****
/*  Write_data_EEPROM                                     */
/*      argument1:write data(unsigned char)             */
/*      return: none                                    */
/*****
void Write_data_EEPROM( unsigned char wr_data )
{
    IIC2.ICDRT = wr_data;                               /* < >Data set          */
    while(IIC2.ICSR.BIT.TDRE == 0);                    /* < >Finish Send?     */
}

/*****
/*  Write_data_End_EEPROM                               */
/*      argument1:write data(unsigned char)             */
/*      return: none                                    */
/*****
void Write_data_End_EEPROM( unsigned char wr_data )
{
    IIC2.ICDRT = wr_data;                               /* <3>Set low address   */
    while(IIC2.ICSR.BIT.TEND == 0);                    /* <3>send end?        */

    IIC2.ICSR.BIT.STOP = 0;                             /* (STOP=0)           */
    IIC2.ICCR2.BYTE = 0x3D;                             /* (BSY=0,SCP=0)      */

    while(IIC2.ICSR.BIT.STOP == 0);                    /* STOP end?          */

    IIC2.ICCR1.BYTE = 0x81;                             /* End Master send(MST=0,TRS=0) */
    IIC2.ICSR.BIT.TDRE = 0;                             /* TDRE = 0           */
}

/*****
/*  Set_adrs_EEPROM                                     */
/*      argument1:write/read address (unsigned short)   */
/*      return: 1:OK/0:NG EEPROM NOACK (unsigned char) */
/*****
/*****
/*  (ADDRESS SET ACTION / DUMMY WRITE ACTION)          */
/*  <1>          <2>          <3>                      */
/*  123456789    123456789    123456789              */
/*  101000000    000000000    000000000              */
/*  slave WA    addressHI A    addressLO A            */
/*****
unsigned char Set_adrs_EEPROM( unsigned short adrs )
{
    IIC2.ICCR1.BYTE = 0xB1;                             /* Set Master send(MST=1,TRS=1) */
    IIC2.ICCR2.BYTE = 0xBD;                             /* (BSY=1,SCP=0)          */

    IIC2.ICDRT = (unsigned char)(DEVICE_CODE | SLAVE_ADRS | IIC_DATA_W); /* <1>Set slave address */

    while(IIC2.ICSR.BIT.TEND == 0);                    /* <1>send end?          */

    if(IIC2.ICIER.BIT.ACKBR != 0)                      /* ACK?                  */
        return(0);

    IIC2.ICDRT = (unsigned char)(adrs >> 8);          /* <2>Set high address   */
    while(IIC2.ICSR.BIT.TDRE == 0);                    /* <2>send end?          */
    IIC2.ICDRT = (unsigned char)(adrs >> 16);          /* <3>Set high address   */
    while(IIC2.ICSR.BIT.TDRE == 0);                    /* <3>send end?          */

    IIC2.ICDRT = (unsigned char)(adrs & 0x00FF);      /* <3>Set low address    */
    while(IIC2.ICSR.BIT.TEND == 0);                    /* <3>send end?          */

    return(1);
}

```

```

/*****
/*  Recv_datan_EEPROM
/*      argument1:read data address(unsigned char *)
/*      argument2:read byte lto64(unsigned char)
/*      return: 1:OK/0:NG EEPROM NOACK      (unsigned char)
/*****
/*****
/*  (CURRENT ADDRESS READ ACTION)
/*  <1>      <2>      <3>      <n>
/*  123456789    123456789    123456789    123456789
/*  101000010    000000000    000000000    000000000
/*  slave RA     readdataA    readdataA    readdataA
/*****
unsigned char Recv_datan_EEPROM( unsigned char *rd_ptr , unsigned short no )
{
    unsigned char    recv;

    IIC2.ICCR1.BYTE = 0xB1;          /* (ICE=1,TRS=1)
    IIC2.ICCR2.BYTE = 0xBD;          /* (BSY=1,SCP=0)

    IIC2.ICDRT = (unsigned char)(DEVICE_CODE | SLAVE_ADRS | IIC_DATA_R);    /* <1>Set slave address
    while(IIC2.ICSR.BIT.TEND == 0);    /* <1>send end?
    if(IIC2.ICIER.BIT.ACKBR != 0)    /* ACK?
        return(0);

    IIC2.ICSR.BIT.TEND = 0;          /* Set Master recv
    IIC2.ICCR1.BIT.TRIS = 0;        /* TEND = 0
    IIC2.ICSR.BIT.TDRE = 0;        /* Set Master recv(TRS = 0)
    IIC2.ICSR.BIT.TDRE = 0;        /* TDRE = 0
    IIC2.ICIER.BIT.ACKBT = 0;      /* (ACKBT = 0)

    recv = IIC2.ICDRR;             /* dummy read
    while(IIC2.ICSR.BIT.RDRF == 0); /* dummy recv end?
    -----
    while(2 < no){
        *rd_ptr = IIC2.ICDRR;      /* <>read
        while(IIC2.ICSR.BIT.RDRF == 0);    /* <>recv end?
        rd_ptr++;                  /* address increment
        no--;
    }
    -----
    IIC2.ICIER.BIT.ACKBT = 1;      /* (ACKBT = 1)
    IIC2.ICCR1.BIT.RCVD = 1;      /* (RCVD = 1)

    *rd_ptr = IIC2.ICDRR;          /* <no-1>read
    rd_ptr++;                      /* address increment
    while(IIC2.ICSR.BIT.RDRF == 0);    /* <no-1>recv end?
    -----
    IIC2.ICSR.BIT.STOP = 0;        /* (STOP=0)
    IIC2.ICCR2.BYTE = 0x3D;        /* (BSY=0,SCP=0)
    while(IIC2.ICSR.BIT.STOP == 0);    /* STOP end?

    *rd_ptr = IIC2.ICDRR;          /* <no>read
    IIC2.ICCR1.BIT.RCVD = 0;      /* (RCVD = 0)
    IIC2.ICCR1.BIT.MST = 0;      /* (MST=0)

    return(1);
}

```

Link address specifications

Section Name	Address
CV1	0x0000
CV2	0x001C
CV3	0x001E
P	0x0100
B	0xFB80

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Sep.29.03	—	First edition issued
2.00	May.07.04	—	Clerical error 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.