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.

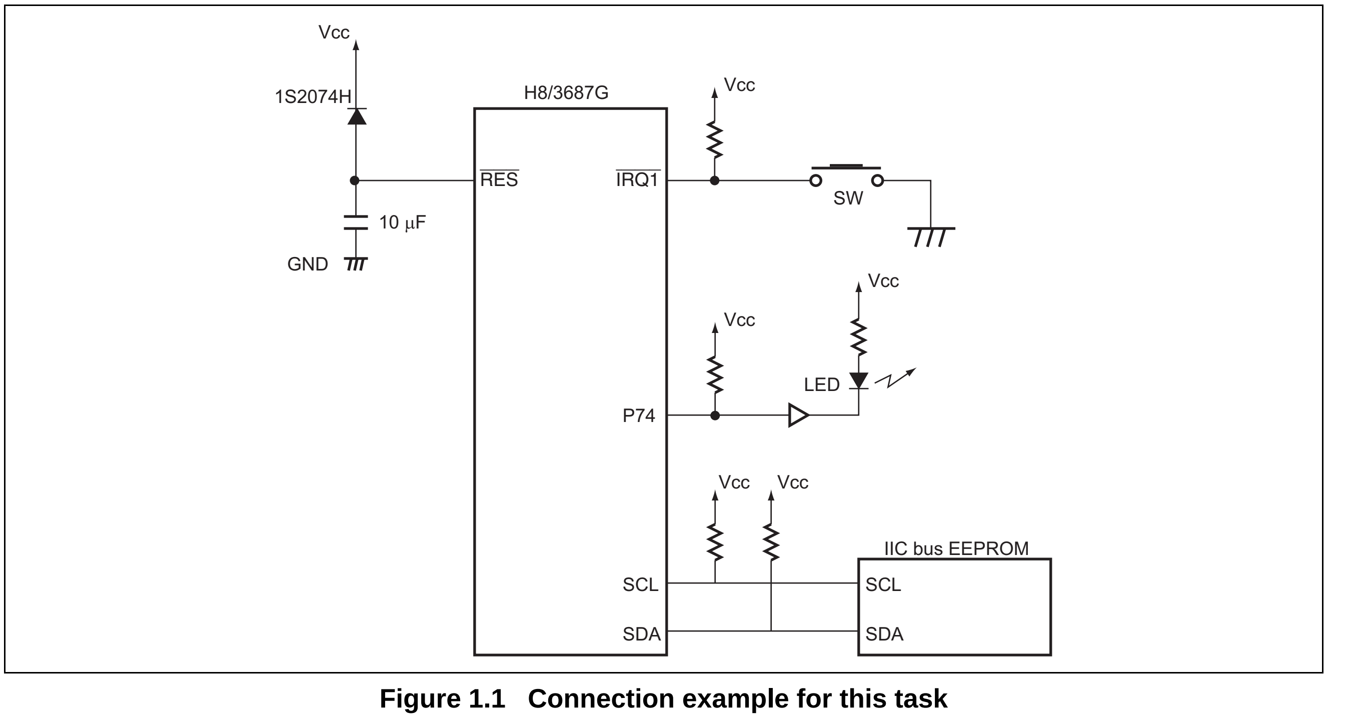


Figure 1.1 Connection example for this task

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.

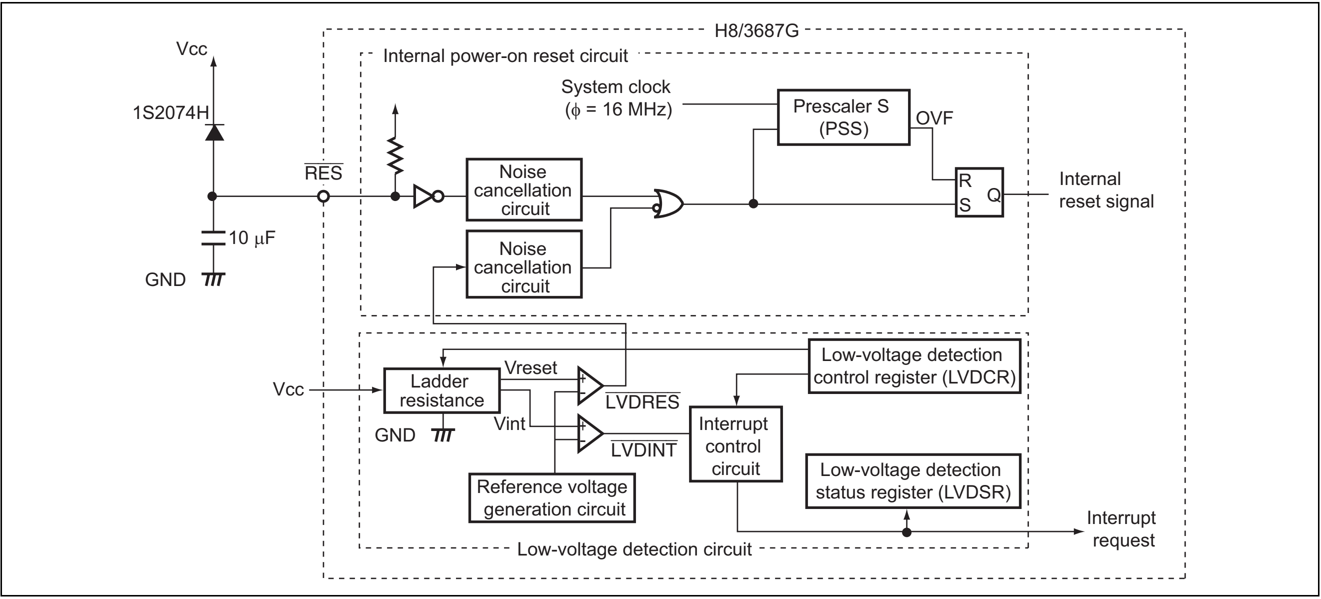
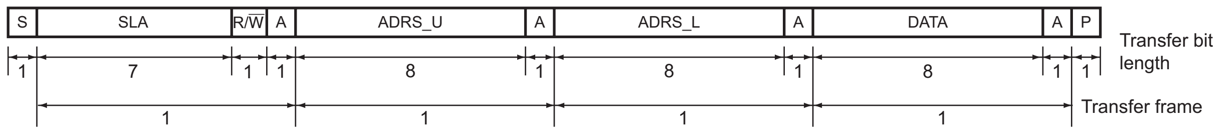

H8/3687G
Vcc
Internal power-on reset circuit
System clock
(ϕ = 16 MHz)
Prescaler S
(PSS)
OVF
1S2074H
RES
Noise cancellation circuit
R
S
Q
Internal reset signal
10 μF
GND
Noise cancellation circuit
Low-voltage detection control register (LVDCR)
Vcc
Ladder resistance
Vreset
LVDRES
GND
Vint
LVDINT
Interrupt control circuit
Low-voltage detection status register (LVDSR)
Reference voltage generation circuit
Interrupt request
Low-voltage detection circuit

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.



[Legend]

S:	Start condition	ADRS_U:	Upper EEPROM memory address
SLA:	EEPROM slave address	ADRS_L:	Lower EEPROM memory address
R/W:	Reception/ transmission direction	DATA:	Transmission data
A:	Acknowledgement	P:	Stop condition

Figure 2.2 IIC bus interface format

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

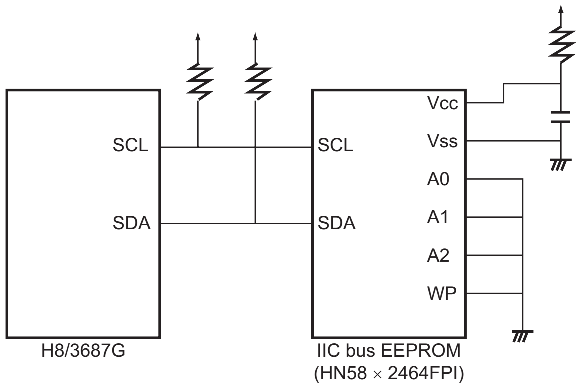


Figure 2.3 IIC Bus EEPROM Connection Example

2. 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

3. 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.

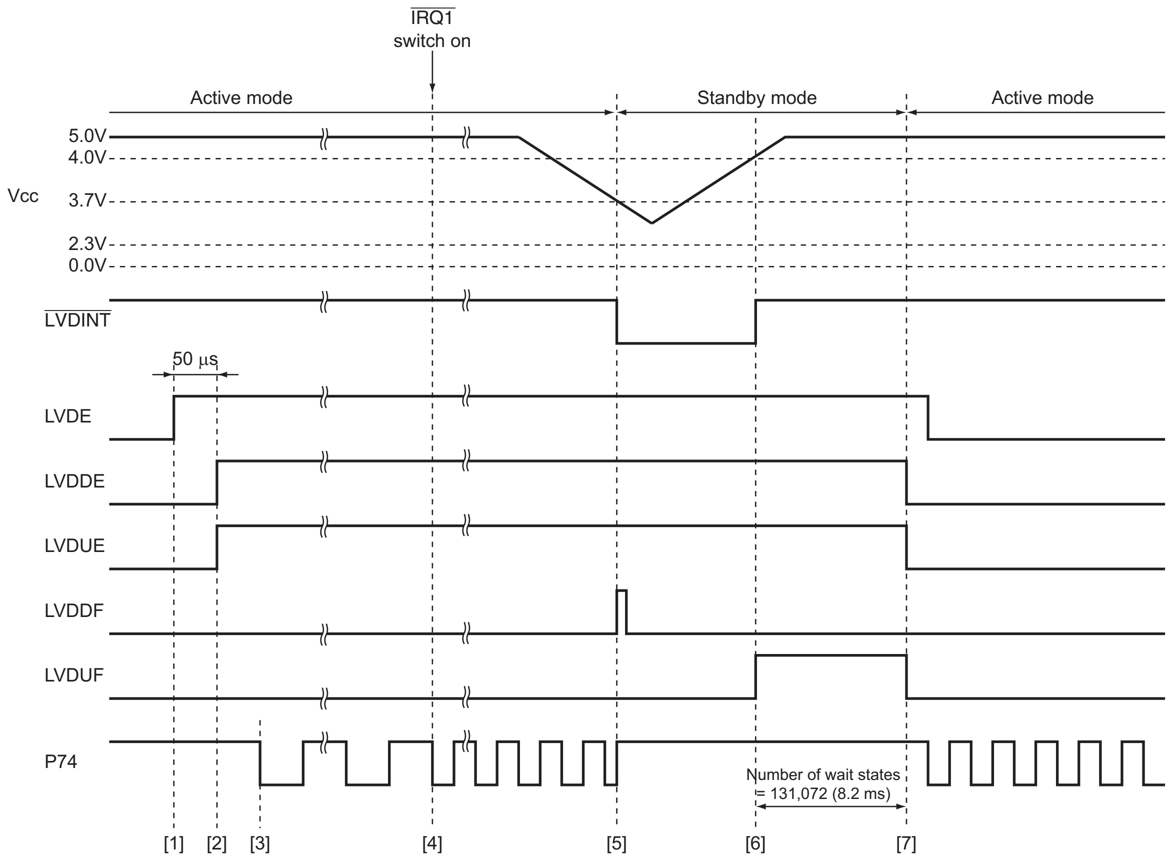
Table 2.2 64-kbit EEPROM manufactured by Renesas Technology Corp. (HN58X2464FPI)

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 µA (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

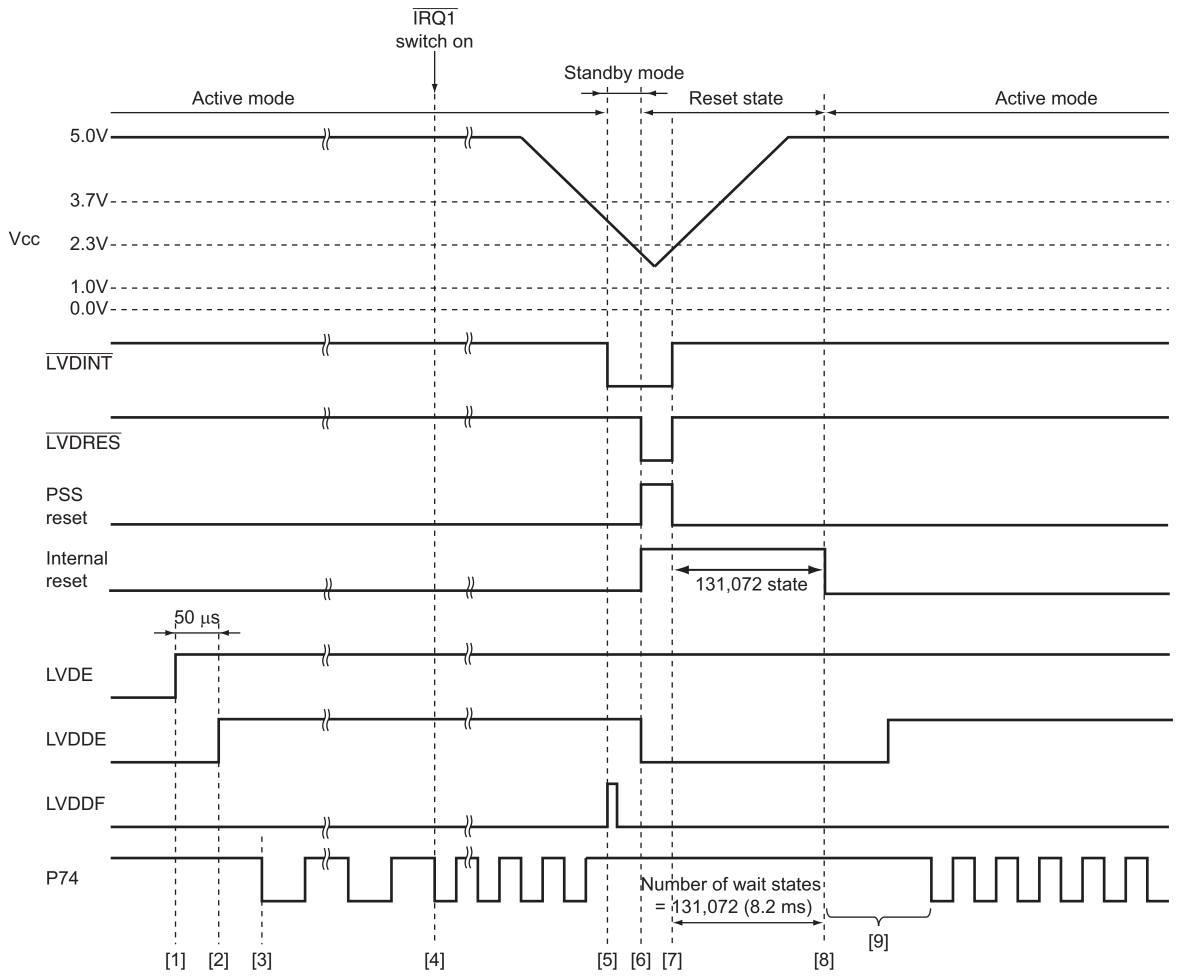
   Figure 3.1 shows the procedure for setting and canceling LVDI, and transitions to standby mode triggered by low-voltage detection interrupts.



	Hardware processing	Software processing
[1]	None	Use low-voltage detection circuit
[2]	None	Set LVDR
[3]	None	Read LED blinking interval from EEPROM and start blinking of LED
[4]	Set IRRI1 to 1	Begin $\overline{\text{IRQ1}}$ interrupt processing Change LED blinking interval
[5]	Set IRRI0 to 1	Begin voltage drop ($\overline{\text{IRQ0}}$) interrupt processing Store LED blinking interval data in RAM to EEPROM Execute SLEEP instruction
[6]	Set IRRI0 to 1	Begin voltage rise ($\overline{\text{IRQ0}}$) interrupt processing
[7]	Cancel standby mode	None

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.



	Hardware processing	Software processing
[1]	None	Use low-voltage detection circuit
[2]	None	Set LVDR
[3]	None	Read LED blinking interval from EEPROM and start blinking of LED
[4]	Set IRRI1 to 1	Begin $\overline{\text{IRQ1}}$ interrupt processing Change LED blinking interval
[5]	Set IRRI0 to 1	Begin voltage drop ($\overline{\text{IRQ0}}$) interrupt processing Store LED blinking interval data in RAM to EEPROM Execute SLEEP instruction
[6]	Internal reset occurs	None
[7]	Begin PSS reset count-up	None
[8]	Cancel internal reset	None
[9]	None	Perform steps [1] through [3]

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 µs

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	LVDSEL	0	LVDR detection level selection LVDSEL = 0: Sets reset detection voltage to 2.3 V LVDSEL = 1: Sets reset detection voltage to 3.6 V

2	LVDRE	1	LVDR enable
			LVDRE = 0: Disables reset by LVDR
			LVDRE = 1: Enables reset by LVDR

1	LVDDE	1	LVDR enable LVDDE = 0: Disables interrupt requests when voltage falls LVDDE = 1: Enables interrupt requests when voltage falls
0	LVDUE	1	LVDR enable LVDUE = 0: Disables interrupt requests when voltage rises LVDUE = 1: Enables interrupt requests when voltage rises

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

Bit	Bit name	Setting	Function
1	LVDDF	0	LVD power supply voltage drop flag LVDDF = 0: Cleared to 0 state LVDDF = 1: Power supply voltage has fallen to 3.7 V or below
0	LVDUF	0	LVD power supply voltage rise flag LVDUF = 0: Cleared to 0 state LVDUF = 1: While the LVDUE 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: 0xFFF0

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
5
4
	STS2
	STS1
	STS0
		STS2 = 1
		STS1 = 0
		STS0 = 0
			Standby timer select 2 to 0
			When STS2 = 1, STS1 = 0 and STS0 = 0, the number of wait states is set to 131,072 states

- SYSCR2 System control register 2 Address: 0xFFF1

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
3
2
	MA2
	MA1
	MA0
		MA2 = 0
		MA1 = x
		MA0 = x
			Active mode clock select 2 to 0
			MA2 = 0, MA1 = x, MA0 = x:
			Sets active mode/sleep mode operating clock to ϕosc
			(x: don't care)

- IEGR1 Interrupt edge select register 1 Address: 0xFFF2

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: 0xFFF4

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: 0xFFF6

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
2
1
0
	CKS3
	CKS2
	CKS1
	CKS0
		CKS3 = 0
		CKS2 = 0
		CKS1 = 0
		CKS0 = 1
			Transfer clock select 3 to 0
			CKS3 = 0, CKS2 = 0, CKS1 = 0, CKS0 = 1: Set transfer rate to 400 kHz when ϕ = 16

- 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.

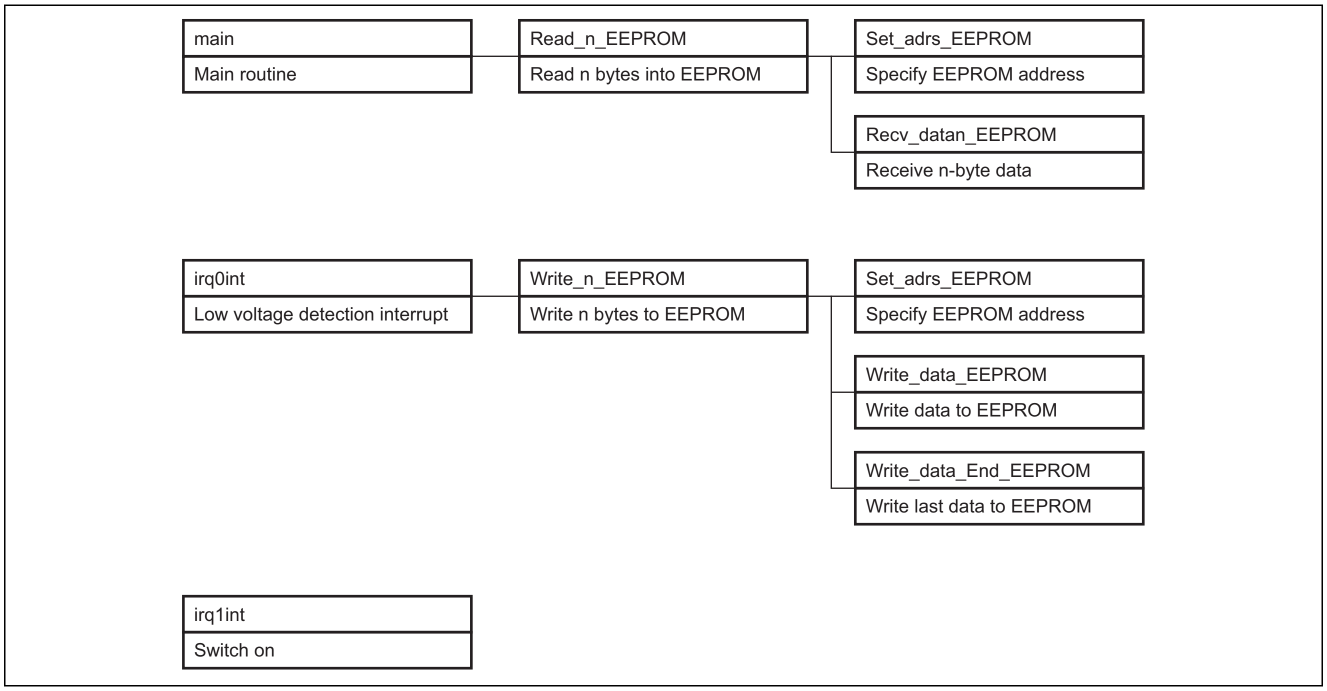
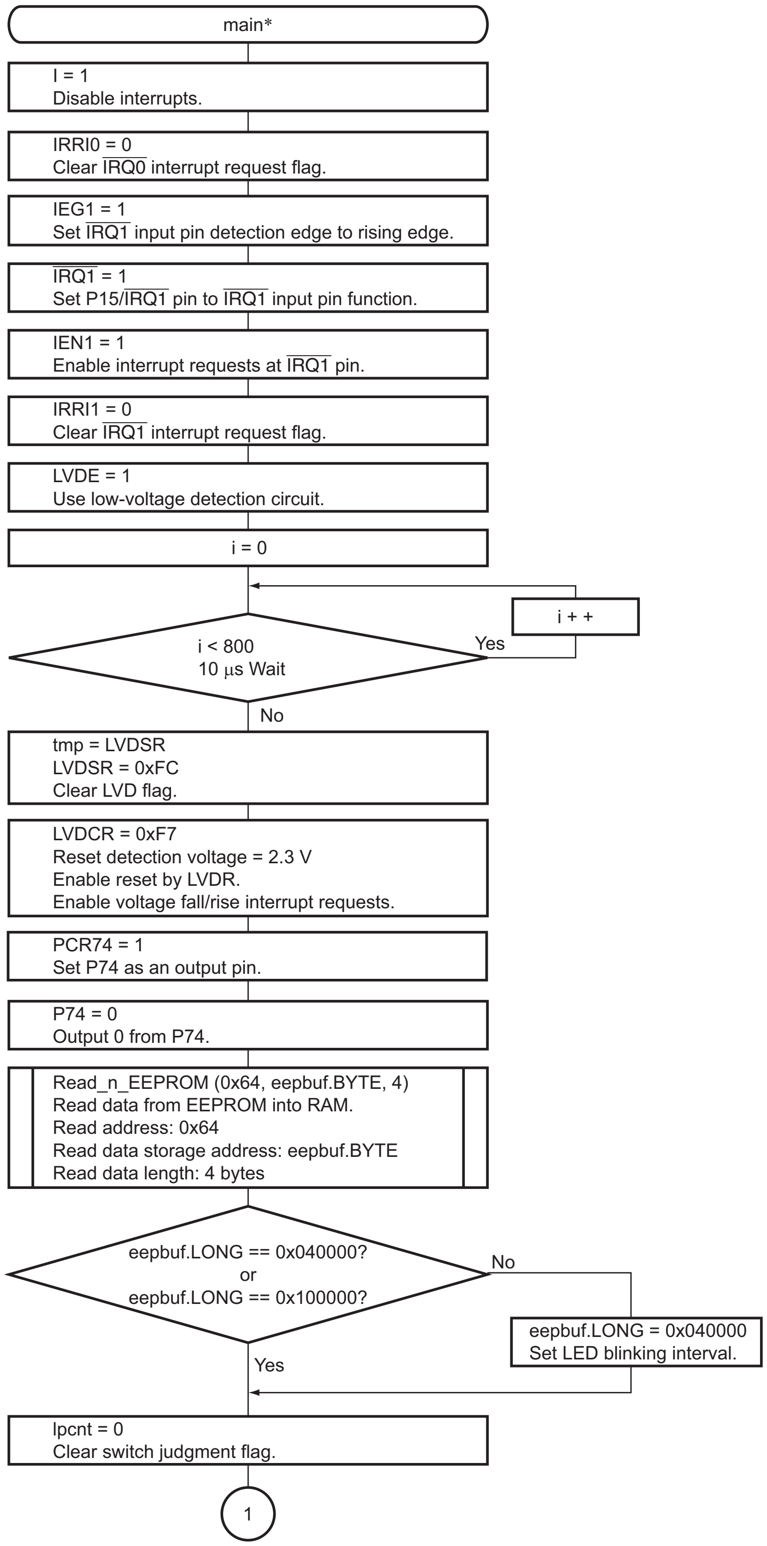


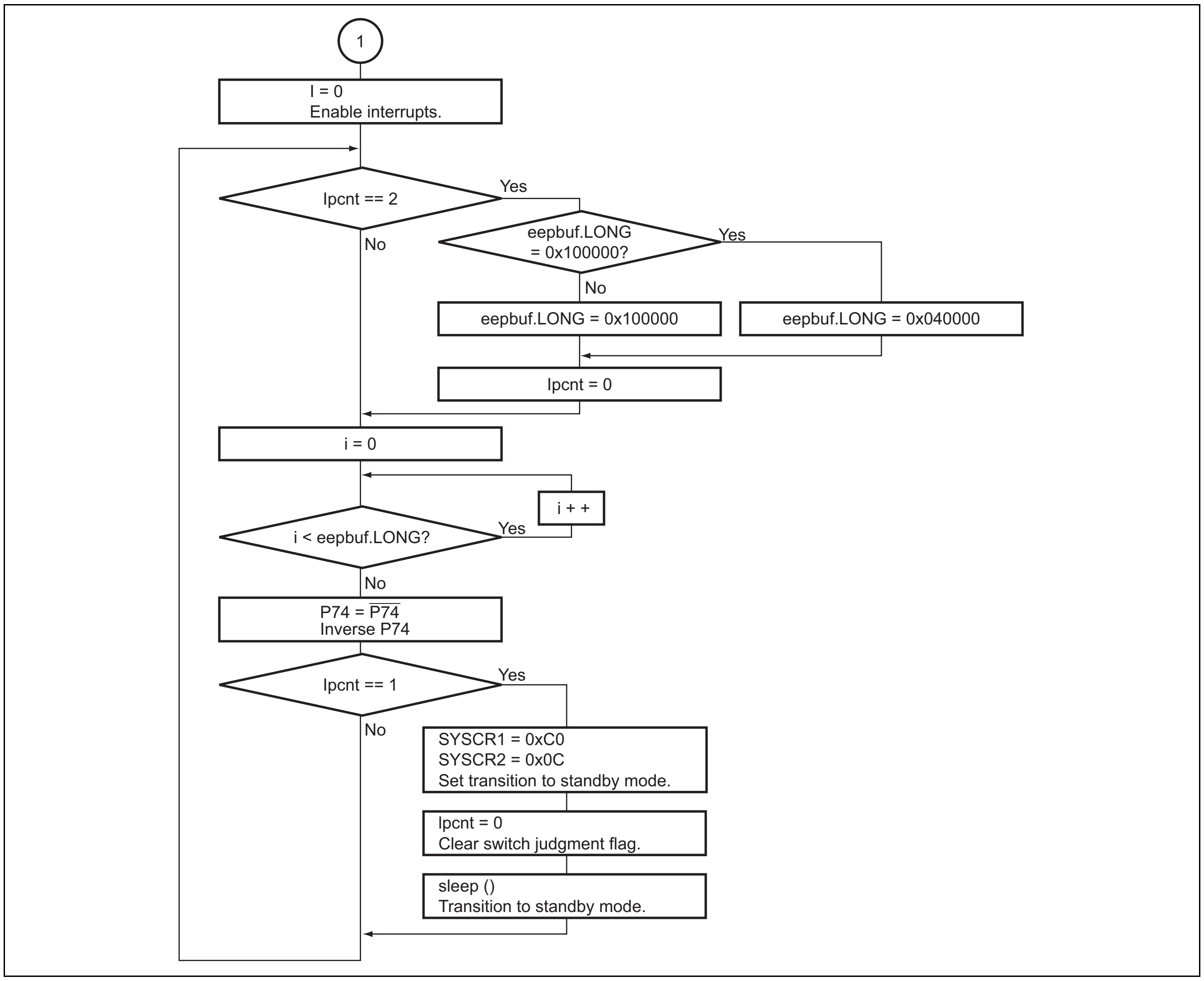
Figure 4.1 Module Hierarchical Diagram

5. Flowcharts

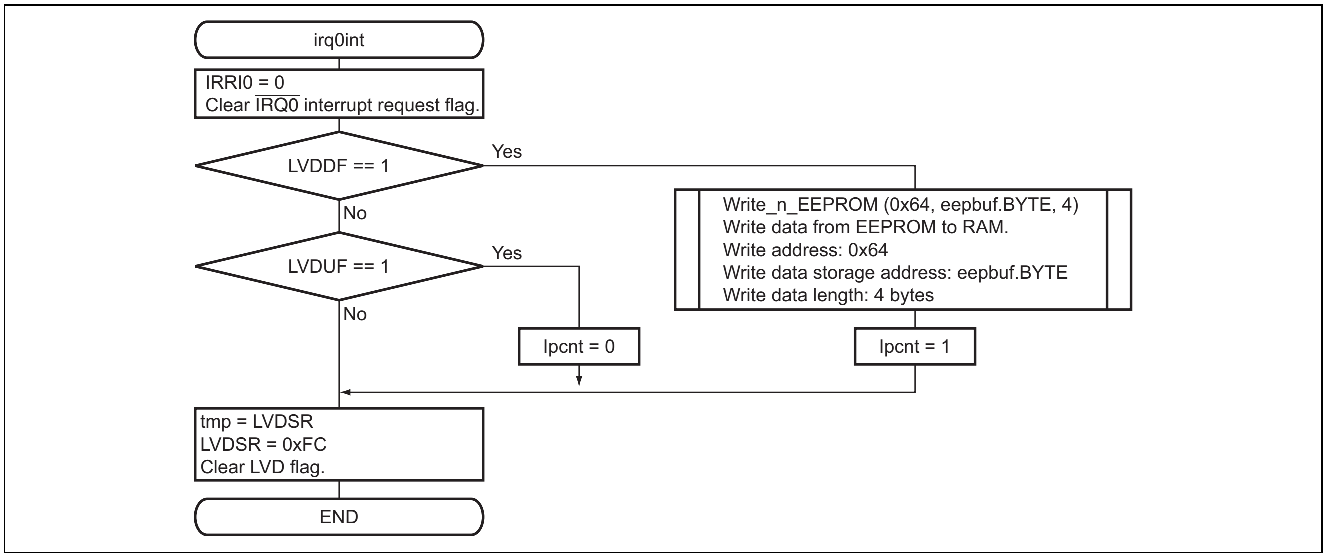
1. Main routine


main*
I = 1
Disable interrupts.
IRRI0 = 0
Clear IRQ0 interrupt request flag.
IEG1 = 1
Set IRQ1 input pin detection edge to rising edge.
IRQ1 = 1
Set P15/IRQ1 pin to IRQ1 input pin function.
IEN1 = 1
Enable interrupt requests at IRQ1 pin.
IRRI1 = 0
Clear IRQ1 interrupt request flag.
LVDE = 1
Use low-voltage detection circuit.
i = 0
i < 800
10 µs Wait
Yes
i + +
No
tmp = LVDSR
LVDSR = 0xFC
Clear LVD flag.
LVDCR = 0xF7
Reset detection voltage = 2.3 V
Enable reset by LVDR.
Enable voltage fall/rise interrupt requests.
PCR74 = 1
Set P74 as an output pin.
P74 = 0
Output 0 from P74.
Read_n_EEPROM (0x64, eepbuf.BYTE, 4)
Read data from EEPROM into RAM.
Read address: 0x64
Read data storage address: eepbuf.BYTE
Read data length: 4 bytes
eepbuf.LONG == 0x040000?
or
eepbuf.LONG == 0x100000?
No
eepbuf.LONG = 0x040000
Set LED blinking interval.
Yes
lpcnt = 0
Clear switch judgment flag.
1

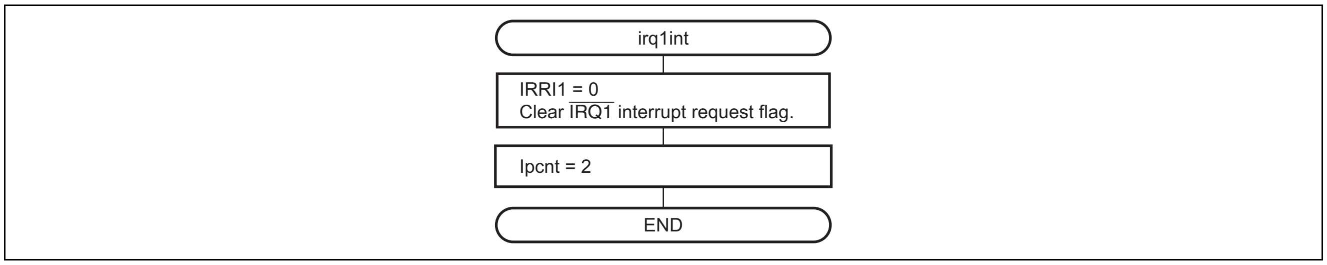
Note: * The stack pointer is set using INIT.SRC (assembly language).



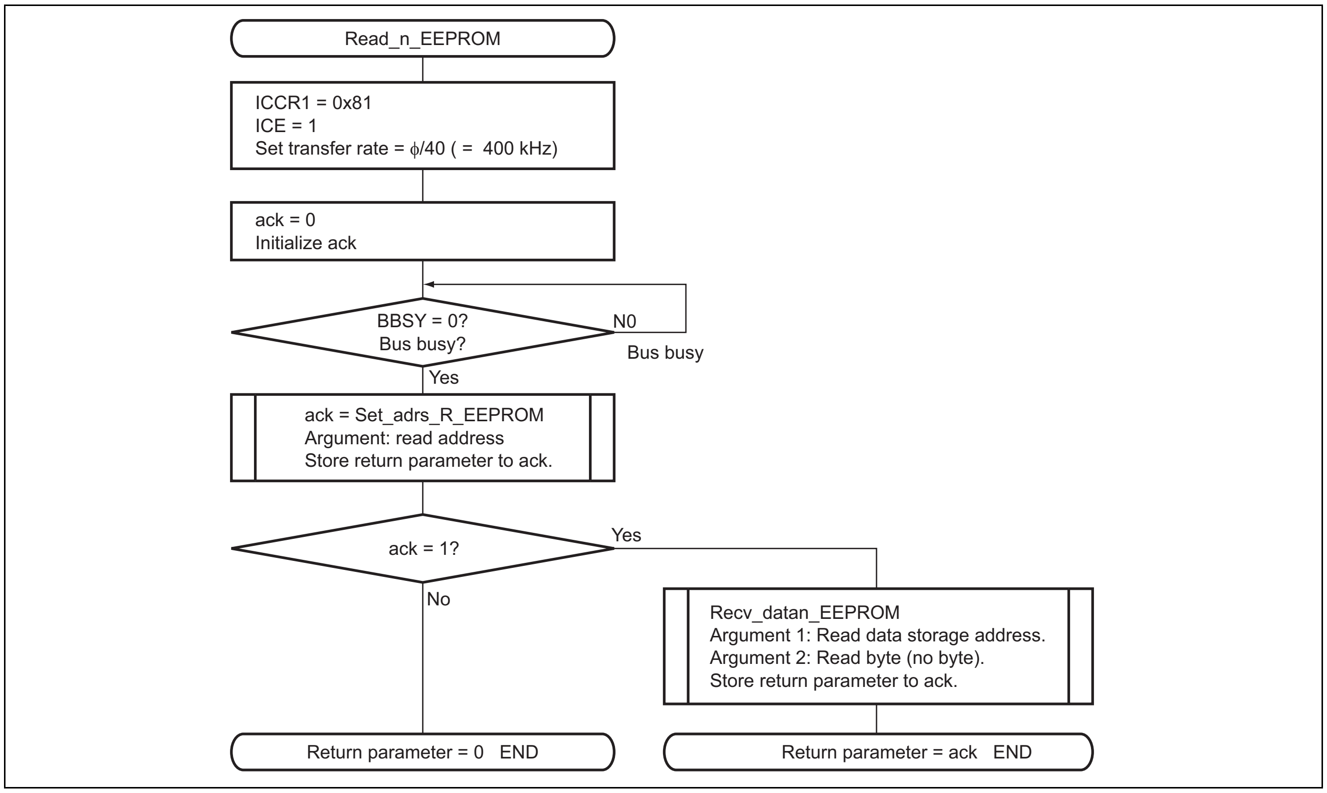
2. Low-voltage Detection Interrupts

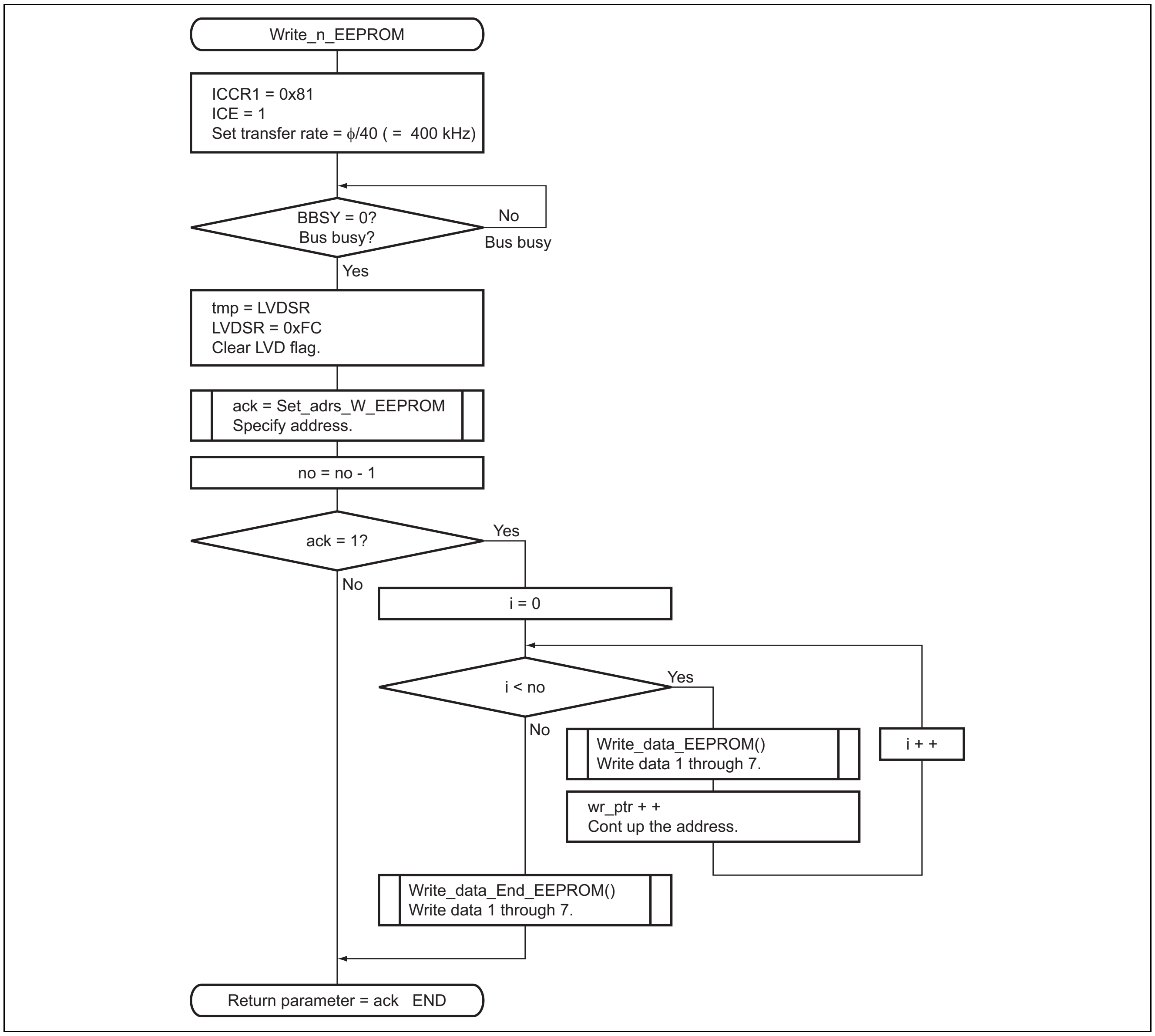

irq0int
IRRI0 = 0
Clear $\overline{IRQ0}$ interrupt request flag.
LVDDF == 1
Yes
No
Write_n_EEPROM (0x64, eepbuf.BYTE, 4)
Write data from EEPROM to RAM.
Write address: 0x64
Write data storage address: eepbuf.BYTE
Write data length: 4 bytes
LVDUF == 1
Yes
No
lpcnt = 0
lpcnt = 1
tmp = LVDSR
LVDSR = 0xFC
Clear LVD flag.
END

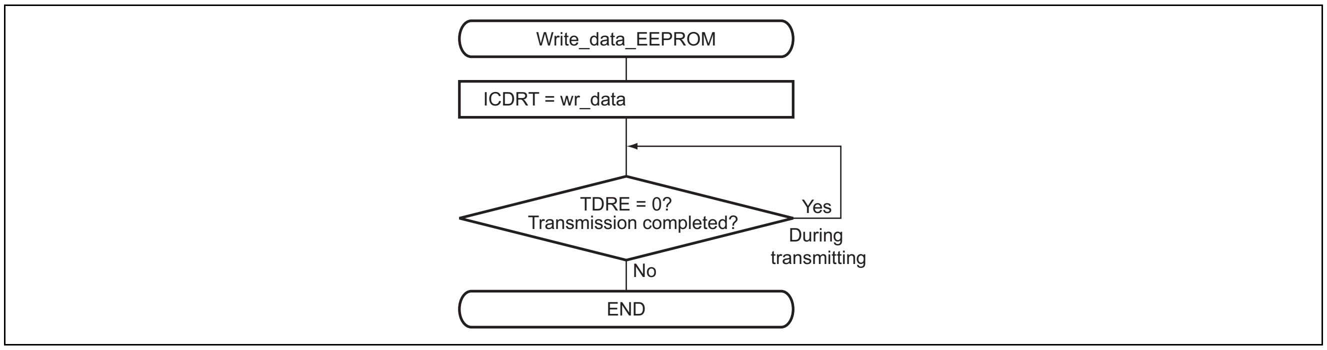
3. Switch-on

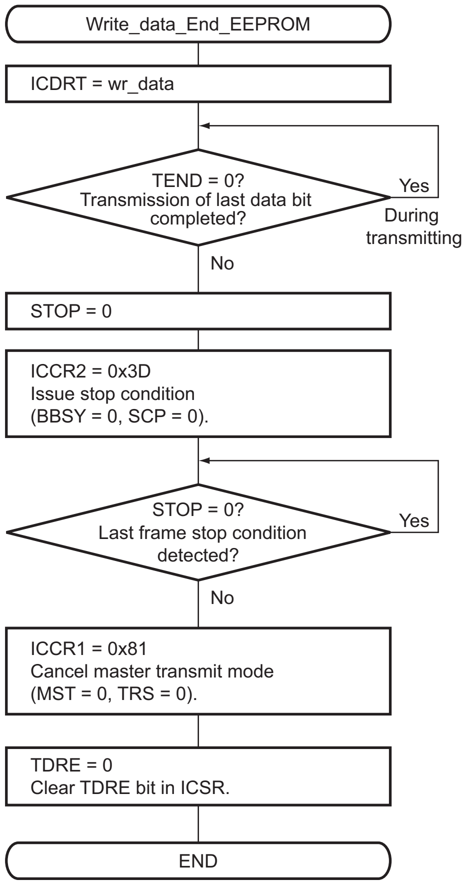


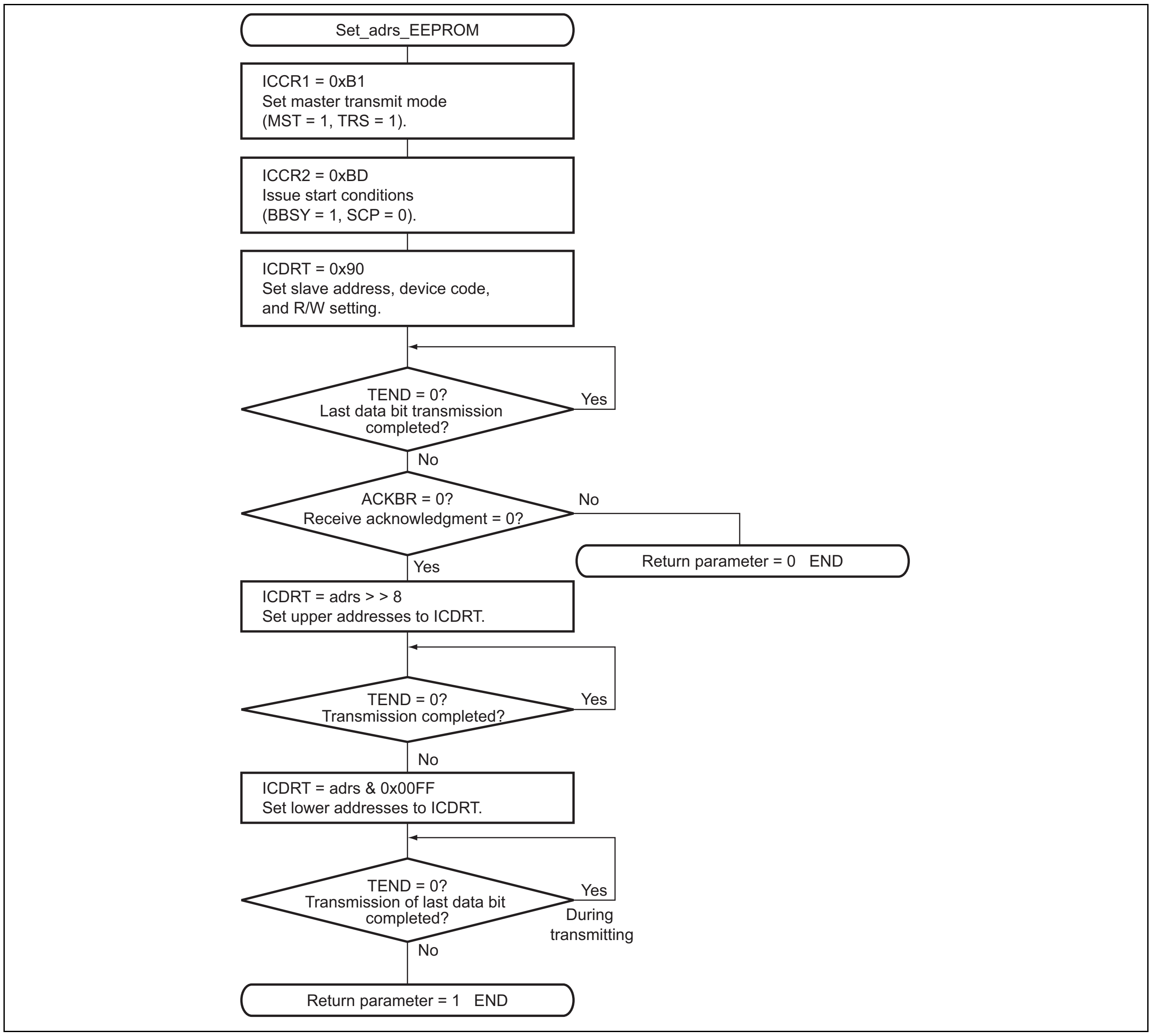
4. EEPROM Access Routine

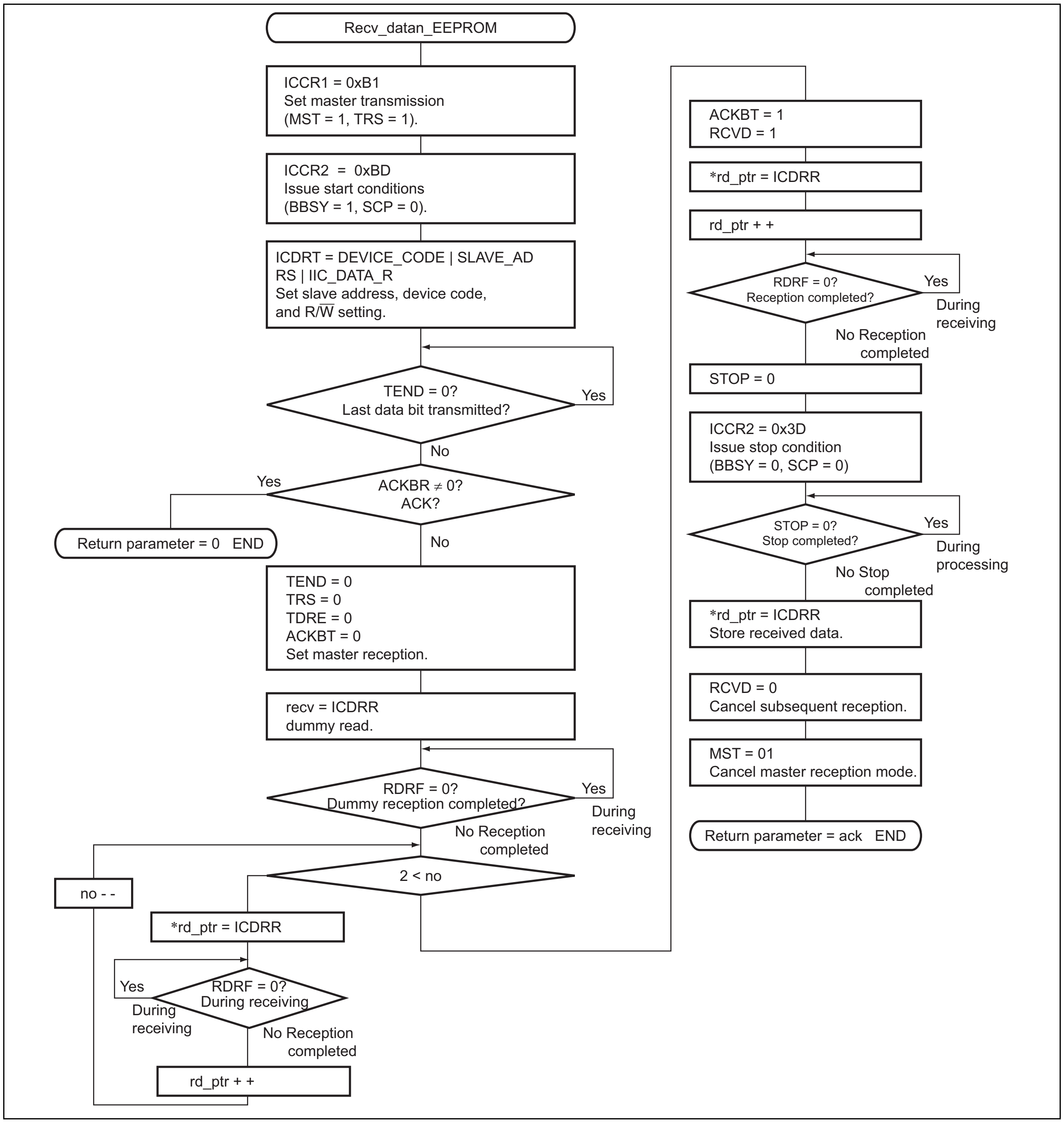

Read_n_EEPROM
ICCR1 = 0x81
ICE = 1
Set transfer rate = ϕ/40 (= 400 kHz)
ack = 0
Initialize ack
BBSY = 0?
Bus busy?
N0
Bus busy
Yes
ack = Set_adrs_R_EEPROM
Argument: read address
Store return parameter to ack.
ack = 1?
Yes
No
Recv_datan_EEPROM
Argument 1: Read data storage address.
Argument 2: Read byte (no byte).
Store return parameter to ack.
Return parameter = 0 END
Return parameter = ack END


Write_n_EEPROM
ICCR1 = 0x81
ICE = 1
Set transfer rate = ϕ/40 (= 400 kHz)
BBSY = 0?
Bus busy?
No
Bus busy
Yes
tmp = LVDSR
LVDSR = 0xFC
Clear LVD flag.
ack = Set_adrs_W_EEPROM
Specify address.
no = no - 1
ack = 1?
Yes
No
i = 0
i < no
Yes
No
Write_data_EEPROM()
Write data 1 through 7.
i + +
wr_ptr + +
Cont up the address.
Write_data_End_EEPROM()
Write data 1 through 7.
Return parameter = ack END


Write_data_EEPROM
ICDRT = wr_data
TDRE = 0?
Transmission completed?
Yes
During
transmitting
No
END


Write_data_End_EEPROM
ICDRT = wr_data
TEND = 0?
Transmission of last data bit completed?
Yes
During transmitting
No
STOP = 0
ICCR2 = 0x3D
Issue stop condition
(BBSY = 0, SCP = 0).
STOP = 0?
Last frame stop condition detected?
Yes
No
ICCR1 = 0x81
Cancel master transmit mode
(MST = 0, TRS = 0).
TDRE = 0
Clear TDRE bit in ICSR.
END


Set_adrs_EEPROM
ICCR1 = 0xB1
Set master transmit mode
(MST = 1, TRS = 1).
ICCR2 = 0xBD
Issue start conditions
(BBSY = 1, SCP = 0).
ICDRT = 0x90
Set slave address, device code,
and R/W setting.
TEND = 0?
Last data bit transmission
completed?
Yes
No
ACKBR = 0?
Receive acknowledgment = 0?
No
Return parameter = 0 END
Yes
ICDRT = adrs > > 8
Set upper addresses to ICDRT.
TEND = 0?
Transmission completed?
Yes
No
ICDRT = adrs & 0x00FF
Set lower addresses to ICDRT.
TEND = 0?
Transmission of last data bit
completed?
Yes
During
transmitting
No
Return parameter = 1 END


Recv_datan_EEPROM
ICCR1 = 0xB1
Set master transmission
(MST = 1, TRS = 1).
ICCR2 = 0xBD
Issue start conditions
(BBSY = 1, SCP = 0).
ICDRT = DEVICE_CODE | SLAVE_AD
RS | IIC_DATA_R
Set slave address, device code,
and R/W setting.
TEND = 0?
Last data bit transmitted?
Yes
No
ACKBR ≠ 0?
ACK?
Yes
Return parameter = 0 END
No
TEND = 0
TRS = 0
TDRE = 0
ACKBT = 0
Set master reception.
recv = ICDRR
dummy read.
RDRF = 0?
Dummy reception completed?
Yes
During
receiving
No Reception
completed
2 < no
no - -
*rd_ptr = ICDRR
RDRF = 0?
During receiving
Yes
During
receiving
No Reception
completed
rd_ptr + +
ACKBT = 1
RCVD = 1
*rd_ptr = ICDRR
rd_ptr + +
RDRF = 0?
Reception completed?
Yes
During
receiving
No Reception
completed
STOP = 0
ICCR2 = 0x3D
Issue stop condition
(BBSY = 0, SCP = 0)
STOP = 0?
Stop completed?
Yes
During
processing
No Stop
completed
*rd_ptr = ICDRR
Store received data.
RCVD = 0
Cancel subsequent reception.
MST = 01
Cancel master reception mode.
Return parameter = ack END

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     LVDSEL      LVDCR_BIT.b3                                          /* LVDI Detection Level Select              */
#define     LVDRE       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     LVDUF       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 *)0xFFF0                     /* System Control Register 1                */
#define     SYSCR2      *(volatile unsigned char *)0xFFF1                     /* System Control Register 2                */
#define     IEGR1_BIT   (*(struct BIT *)0xFFF2)                               /* Interrupt Edge Select Register 1         */
#define     IEG1        IEGR1_BIT.b1                                          /* IRQ0 Edge Select                         */
#define     IEG0        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 *)0xFFF6)                        /* Interrupt Request Register 1      */
#define     IRRI1        IRR1_BIT.b1                                         /* IRQ1 Interrupt Request Flag       */
#define     IRRI0        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];
}eepbuf;

/*****************************************************************************************************************/
/*   Vector Address                                                                                              */
/*****************************************************************************************************************/
#pragma section     V1                                                       /* VECTOR SECTOIN SET                */
void (*const VEC_TBL1[])(void) = {                                           /* 0x00 - 0x0f                       */
     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                      */

     IRRI0 = 0;                                                               /* Clear IRRI0                            */
     IEG1 = 1;                                                                /* Rising Edge of IRQ1 Input              */
     IRQ1 = 1;                                                                /* Initialize IRQ1 Terminal Input         */
     IEN1 = 1;                                                                /* IRQ1 Interrupt Enable                  */
     IRRI1 = 0;                                                               /* Clear IRRI1                            */

     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;

     IRRI0 = 0;                                                                   /* Clear IRRI0                      */
     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 write1-7                           */
               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)                           */


                                                                         /* <1>                                     */
     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);
                                                                         /* <2>                                     */
     IIC2.ICDRT = (unsigned char)(adrs >> 8);                            /* <2>Set high address                     */
     while(IIC2.ICSR.BIT.TDRE == 0);                                     /* <2>send end?                            */
                                                                         /* <3>                                     */
     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 1to64(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)                    */

                                                                             /* <1>                              */
     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);
                                                                             /* Set Master recv                  */
     IIC2.ICSR.BIT.TEND = 0;                                                 /* TEND = 0                         */
     IIC2.ICCR1.BIT.TRS = 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