SH7786 Group

SH7786 Interrupt Controller Inter-CPU Interrupt Usage Example

R01AN0808EJ0100
Rev.1.00
Jan 20, 2012

Introduction

This document presents a sample program that demonstrates the use of inter-CPU interrupts, a function provided by the SH7786 interrupt controller.

Target Device

SH7786

Contents

1. Introduction

1.1 Specifications

This application note presents the inter-CPU interrupts provided by the SH7786 interrupt controller (INTC), using a program that turns board-mounted LEDs on or off under CPU control each time a timer-generated event occurs.

1.2 Functions Used

- Interrupt controller (INTC inter-CPU interrupts)
- Timer unit (TMU channels 0 and 1)
- General-purpose I/O ports (GPIO port G)

1.3 Applicable Conditions

Evaluation board	AP-AH4AD-0A *1 manufactured by AlphaProject Co., Ltd. External memory Area 0: NOR flash memory: 16 MB S29GL128P90TFIR20 (Spansion Inc.) Areas 2 to 5: DDR3 SDRAM: 256 MB 2 × MT41J64M16LA-187E (Micron Technology)
MCU	SH7786
Operating frequencies	Internal clock: 533 MHz SuperHyway clock: 267 MHz Peripheral clock: 44 MHz DDR3 clock: 533 MHz External bus clock: 89 MHz
Area 0 bus width	16 bits (MD4 = low, MD5 = high, MD6 = low)
Clock operating mode	Clock mode 3 (MD0 = high, MD1 = high, MD2 = low, MD3 = low)
Endian mode	Little endian (MD8 = high)
Addressing mode	29-bit addressing mode (MD10 = low)
Tool chain	Super-H RISC engine Standard Toolchain Ver 9.3.2.0
Compiler options	Other than the include specifications for the High-performance Embedded Workshop, the default settings are used. -cpu=sh4a -endian=little -include="$(PROJDIR)\inc\drv","$(PROJDIR)\inc" -object="$(CONFIGDIR)\$(FILELEAF).obj" -debug -gbr=auto -chgincpath -errorpath -global_volatile=0 -opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1 –nologo
Assembler options	cpu=sh4a -endian=little -round=zero -denormalize=off -include="$(PROJDIR)\inc" -include="$(PROJDIR)\inc\drv" –debug -object="$(CONFIGDIR)\$(FILELEAF).obj" -literal=pool,branch,jump,return -nolist -nologo -chgincpath -errorpath

Note: 1. For detailed information on using the AP-SH4AD-0A, refer to *AP-SH4AD-0A Hardware Manual.*

Table 1.1 lists the section allocations used in this sample program.

Table 1.1 Section Allocations

Section	Section Usage	Area	Allocation Address (Virtual Address)	
INTHandler	Exception/interrupt handler	ROM	0x00000800	P0 area (Can be cached, MMU address conversion not possible)
VECTTBL	Reset vector table Interrupt vector table	ROM		
INTTBL	Interrupt mask table	ROM		
PIntPRG	Interrupt function	ROM		
PResetPRG	Reset program	ROM	0x00002000	
C$BSEC	Uninitialized data area address structure	ROM	0x00004000	
C$DSEC	Initialized data area address structure	ROM		
P	Program area used by CPU0	ROM		
D	Initialized data used by CPU0	ROM		
Pcpu1	Program area used by CPU1	ROM	0x00008000	
Dcpu1	Initialized data used by CPU1	ROM		
RSTHandler_cpu0	CPU0 reset handler	ROM	0xA0000000	P2 area (Can not be cached, MMU address conversion not possible)
RSTHandler_cpu1	CPU1 reset handler	ROM	0xA0000100	
B	Uninitialized data area used by CPU0	RAM	0xE4000000	P4 area (Area mapped to internal resources.)
R	Initialized data area used by CPU0	RAM		
Bcpu1	Uninitialized data area used by CPU1	RAM	0XE4000500	
Rcpu1	Initialized data area used by CPU1	RAM		
S	Stack area used by CPU0	RAM	0xE500E000	
Scpu1	Stack area used by CPU1	RAM	0XE500E400	

Note: Sections for which there is no CPU specification in the Section Usage column are shared between both CPUs.

1.4 Related Application Notes

The reference program used in this document has been verified to operate under the settings and conditions described in the SH7786 Group Application Note SH7786 Initial Settings Example (R01AN0519EJ0101). That application note should be consulted in conjunction with this application note.

2. INTC Inter-CPU Interrupt Usage Example

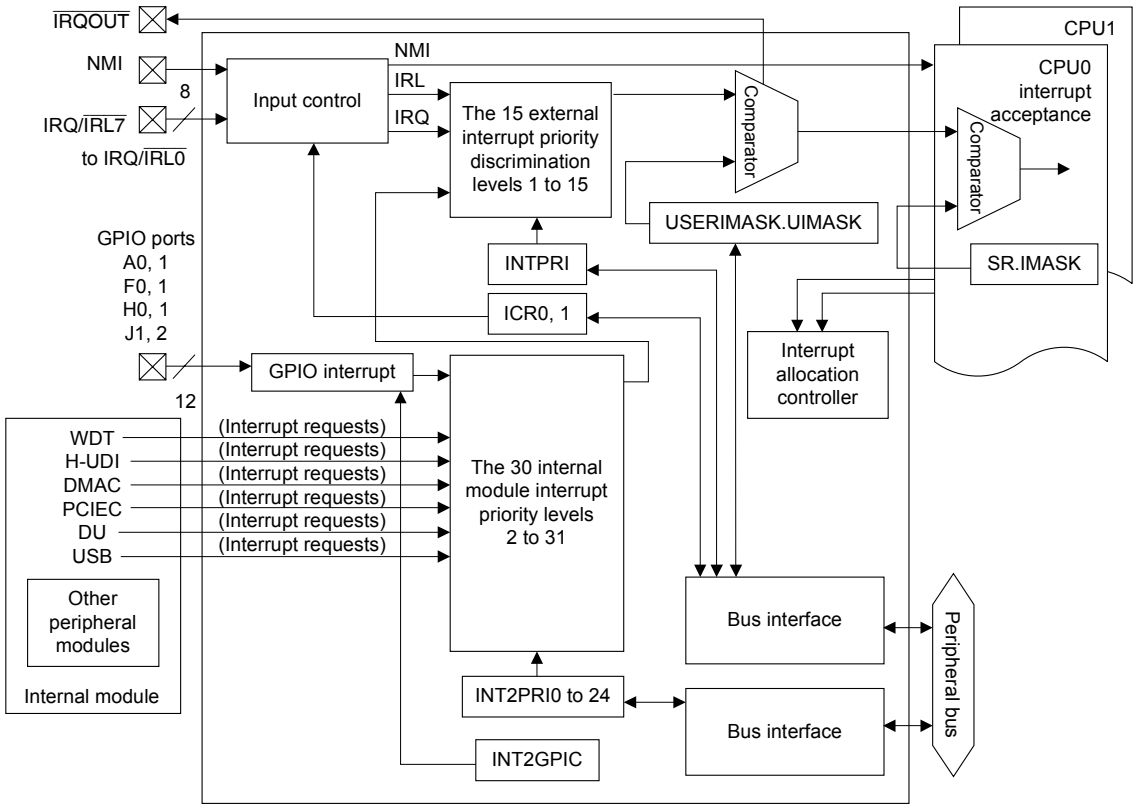
2.1 Application Example

2.1.1 Functions Used and Operation Overview

The interrupt controller (INTC) discriminates the priorities of interrupt factors and controls interrupt requests issued to CPU0 and CPU1 (SH-4A). The INTC has registers to set the priority of each interrupt and interrupt requests are processed according to the priorities set by the user with these registers.

The features of the INTC and its block diagram are presented below.

- External interrupts and internal peripheral module interrupts can be set up in either fixed allocation mode or automatic allocation mode.
- The priority of each external interrupt can be set to one of 15 levels.
- NMI (nonmaskable interrupt) noise cancellation
- NMI requests can be masked by setting the SR.BL bit to 1.
- The SR.IMASK bits can be updated automatically to the level of the accepted interrupt.
- The priority of each internal peripheral module interrupt can be set to one of 30 levels.
- User mode interrupt disable function



Note: The following modules can also output interrupt requests as peripheral modules.
TMU: Timer unit
HPB: HPB bus controller
SCIF: Built-in FIFO serial communications interface
Ether: Ethernet controller
USB: USB controller
I2C: I^2C bus controller
SSI: Serial sound interface
HAC: Audio codec interface
FLCTL: NAND flash memory controller
HSPI: Serial peripheral interface

Abbreviations
INTPRI: Priority setting register
ICR0,1: Interrupt control registers 0 and 1
INT2PRI0 to 24: Priority setting registers 0 to 24
INT2GPIC: GPIO interrupt settings register
USERIMASK.UIMASK: UIMASK bit in the user interrupt mask level setting register
SR.IMASK: IMASK bit in the status register

Figure 2.1 INTC Block Diagram

2.1.2 Inter-CPU Interrupt Generation Factors

Inter-CPU interrupts operate as follows.

- Inter-CPU interrupts are generated by setting an inter-CPU interrupt control register (CnINTICI: n = 0, 1).
- For each CPU, there are 8 factor (8 fields numbered #0 to #7) inputs and for each field an interrupt is generated on the logical AND of 4 bits.
- There are two methods for using these interrupts: (1) with fields allocated to each factor originating CPU or (2) with fields divided by factors that report between CPUs. (Method (1) is used in this application note.)
- Interrupts that are issued to each CPU are handled by the INTICIn interrupt handlers for each field. In this sample program INTICI0 is allocated to handling inter-CPU interrupts from CPU1 to CPU0 and INTICI1 is allocated to handling inter-CPU interrupts from CPU0 to CPU1.
- Cases when there are writes to the same field from multiple CPUs are handled by determining the bit positions in the fields for each CPU.
- A CPU that accepts an interrupt request clears that interrupt request by clearing to 0 the corresponding bit in the CnINTICI register with the inter-CPU interrupt clear register (CnINTICICLR).

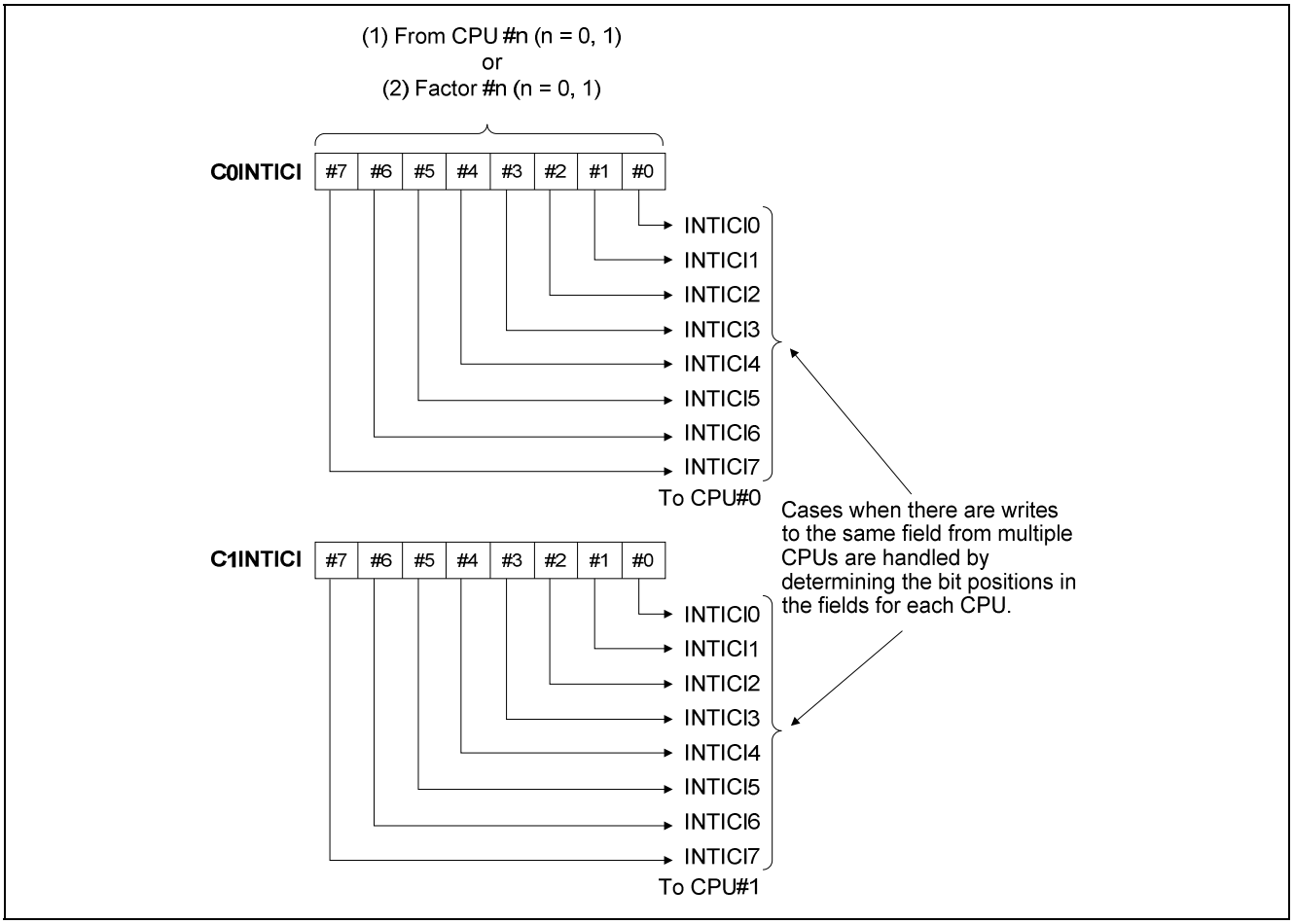


Figure 2.2 Inter-CPU Interrupts

2.1.3 Sample Program

This sample program generates the inter-CPU interrupts using a timer underflow interrupt as the trigger. The sample program also turns the LEDs on the board on and off so that the user can verify that the inter-CPU interrupts were generated.

Table 2.1 lists the specifications of the sample program.

Table 2.1 Sample Program

Item	Specification
Timer	Channel 0 (TMU0) Channel 1 (TMU1)
Timer interrupt generation condition	TMU0: An underflow is generated at 1-second intervals TMU1: An underflow is generated at 0.5-second intervals
Inter-CPU interrupt generation condition	CPU0 → CPU1 inter-CPU interrupt: TMU0 interrupt CPU1 → CPU0 inter-CPU interrupt: TMU1 interrupt
Inter-CPU interrupt allocation	CPU0 → CPU1 inter-CPU interrupt: ICI1 bit 1 CPU1 → CPU0 inter-CPU interrupt: ICI0 bit 1 Bits are allocated so that 1 bit is allocated to 1 factor.
Interrupt priority	TMU0: 3 TMU1: 3 CPU0 → CPU1 inter-CPU interrupt (ICIPRI0): 3 CPU1 → CPU0 inter-CPU interrupt (ICIPRI1): 3
LED control	LD1: Turned on and off by CPU1 LD2: Turned on and off by CPU0

Note: See section 2.1.5, Sample program sequence, for the sequence of control operations.

2.1.4 Sample Program Register Settings

The table below lists the register settings used by this sample program.

Table 2.2 INTC Inter-CPU Interrupt Register Settings

Register (abbreviation)	Address	R/W	Size	Operation Specifications
Inter-CPU interrupt register (CPU0) (C0INTICI)	H'FE41 0070	R/W	32	• Generates interrupts for CPU0 After a TMU1 interrupt is generated, set ICI0 (bits[3:0]) to B'0001. (CPU0 → CPU1 interrupt)
Inter-CPU interrupt register (CPU1) (C1INTICI)	H'FE41 0074	R/W	32	• Generates interrupts for CPU1 After a TMU0 interrupt is generated, set ICI1 (bits[7:4]) to B'0001. (CPU1 → CPU0 interrupt)
Inter-CPU interrupt clear register (CPU0) (C0INTICICLR)	H'FE41 0080	—/W	32	• Clears inter-CPU interrupt requests ICICLR0 (bits[3:0]) = B'0001: Clears the CN0INTICI.ICI0 interrupt. (Writing zero has no effect.)
Inter-CPU interrupt clear register (CPU1) (C1INTICICLR)	H'FE41 0084	—/W	32	• Clears inter-CPU interrupt requests ICICLR1 (bits[7:4]) = B'0001: Clears the CN1INTICI.ICI1 interrupt. (Writing zero has no effect.)
Inter-CPU interrupt priority setting register (CPU0) (C0INTICIPRI)	H'FE41 0090	R/W	32	• Sets the priority relative to CPU0 ICIPRE0 (bits[3:0]) = B'0011 (priority 3)
Inter-CPU interrupt priority setting register (CPU1) (C1INTICIPRI)	H'FE41 0094	R/W	32	• Sets the priority relative to CPU1 ICIPRE1 (bits[7:4]) = B'0011 (priority 3)
Inter-CPU interrupt priority clear register (CPU0) (C0INTICIPRICLR)	H'FE41 00A0	R/W	32	• Clears the set priority Clears the C0INTICIPRO.ICIPRI0 interrupt (Writing zero has no effect.)
Inter-CPU interrupt priority clear register (CPU1) (C1INTICIPRICLR)	H'FE41 00A4	R/W	32	• Clears the set priority Clears the C1INTICIPRO.ICIPRI1 interrupt (Writing zero has no effect.)

Table 2.3 INTC Internal Peripheral Module Interrupt Register Settings

Register (abbreviation)	Address	R/W	Size	Operation Specifications
Peripheral interrupt priority setting register 0 (INT2PRI0)	H'FE41 0800	R/W	32	• Sets the priority at each event code TMU-ch0_ch2 (bits[28:24])="B'0011" (Priority 3)
Peripheral interrupt priority setting register 1 (INT2PRI1)	H'FE41 0804	R/W	32	• Sets the priority at each event code TMU-ch0_ch2 (bits[20:16])="B'0011" (Priority 3)
Interrupt factor register (Is not influenced by the mask state)(CPU0) (C0INT2A0_1)	H'FE41 0A04	R	32	• Indicates the interrupt state for each module type (Is not influenced by the mask state) TMU-ch0_ch2 (bit 31) = B'0: No interrupt TMU-ch0_ch2 (bit 31) = B'1: Interrupt occurred
Interrupt factor register (Is influenced by the mask state)(CPU0) (C0INT2A1_1)	H'FE41 0A14	R	32	• Indicates the interrupt state for each module type (Is influenced by the mask state) TMU-ch0_ch2 (bit 31) = B'0: No interrupt TMU-ch0_ch2 (bit 31) = B'1: Interrupt occurred
Interrupt factor register (Is not influenced by the mask state)(CPU1) (C1INT2A0_1)	H'FE41 0B04	R	32	• Indicates the interrupt state for each module type (Is not influenced by the mask state) TMU-ch0_ch2 (bit 30) = B'0: No interrupt TMU-ch0_ch2 (bit 30) = B'1: Interrupt occurred
Interrupt factor register (Is influenced by the mask state)(CPU1) (C1INT2A1_1)	H'FE41 0B14	R	32	• Indicates the interrupt state for each module type (Is influenced by the mask state) TMU-ch0_ch2 (bit 30) = B'0: No interrupt TMU-ch0_ch2 (bit 30) = B'1: Interrupt occurred
Peripheral interrupt mask register 1(CPU0) (C0INT2MSK1)	H'FE41 0A24	R/W	32	• Sets the mask for each module type Write of B'0 to TMU-ch0_ch2 (bit 31): The ch0 mask is cleared.
Peripheral interrupt mask register 1(CPU1) (C1INT2MSK1)	H'FE41 0B24	R/W	32	• Sets the mask for each module type Write of B'0 to TMU-ch0_ch2 (bit 31): The ch0 mask is cleared.
Peripheral interrupt mask clear register (CPU0) (C0INT2MSKCLR1)	H'FE41 0A34	R/W	32	• Clears the mask for each module type TMU-ch0_ch2 (bit 31) = B'1: Clears the TMU-ch0 mask.
Peripheral interrupt mask clear register (CPU1) (C1INT2MSKCLR1)	H'FE41 0B34	R/W	32	• Clears the mask for each module type TMU-ch0_ch2 (bit 30) = B'1: Clears the TMU-ch1 mask.
Peripheral interrupt detailed factor display register 01 (INT2B01)	H'FE41 0C04	R	32	• Indicates the module type even more precisely. Bit 0: Indicates the TMU-ch0_ch2 (ch0) interrupt Bit 1: Indicates the TMU-ch1_ch2 (ch1) interrupt

Table 2.4 TMU0/1 Register Settings

Register (abbreviation)	Address	R/W	Size	Operation Specifications
Timer start register 0 (TSTR0) * Common to channels 0, 1, and 2	H'FFD8 0004	R/W	8	• Starts/stops the timer 0 counter STR0 (bit 0) = B'0: Stops TCNT0 counter operation (Counting starts after the program starts) • Starts/stops the timer 1 counter STR1 (bit 1) = B'0: Stops TCNT1 counter operation (Counting starts after the program starts)
Timer constant register 0 (TCOR0)	H'FFD8 0008	R/W	32	• Resets TCNT to the count value after an underflow occurs Initial value: H'A7D7
Timer constant register 1 (TCOR1)	H'FFD8 0014			
Timer counter 0 (TCNT0/1)	H'FFD8 000C	R/W	32	• Sets the count that causes an underflow to occur (In this program, this value is set so that an underflow occurs after 1s.) Initial value: H'A7D7
Timer counter 1 (TCNT1)	H'FFD8 0018			
Timer control register 0 (TCR1)	H'FFD8 0010	R/W	16	• Underflow flag UNF (bit 0) = B'0: No underflow UNF (bit 0) = B'1: Underflow occurred • Underflow control UNIE (bit 5) = B'0: Enables underflow interrupts (TUNI). • Clock edge selection CKEG0/1 (bits 3/4): B'00: Count on rising edges. • Timer prescaler TPSC (bits 2 to 0) = B'100: Count on 1024/Pck
Timer control register 1 (TCR1)	H'FFD8 001C			

Table 2.5 Low-Power Mode Settings

Register (abbreviation)	Address	R/W	Size	Operation Specifications
CPU1 standby control register (C1STBCR)	H'FE40 1004	R/W	32	• Starts/stops CPU1 MSTP (bit 0) = B'0: CPU1 operates

Note: : Although CPU0 operates at power on, CPU1 is stopped.

Table 2.6 GPIO Register Settings (LED 1/2 control)

Register (abbreviation)	Address	R/W	Size	Operation Specifications
Port G control register (PGCR)	H'FFCC000C	R/W	16	• PG5 mode PG5MD0/1 (bits 10/11) = B'01: Output set • PG6 mode PG6MD0/1= B'01: Output set
Port G data register (PGDR)	H'FFCC002C	R/W	8	• LED 1 control PG5DT (bit 5) = B'0: LED off PG5DT (bit 5) = B'1: LED on • LED 2 control PG6DT (bit 6) = B'0: LED off PG6DT (bit 6) = B'1: LED on

Note: Registers that are not used and bits that are not set are used with their initial values.

2.1.5 Sample Program Sequence

The following figures show the sample program operation sequences.

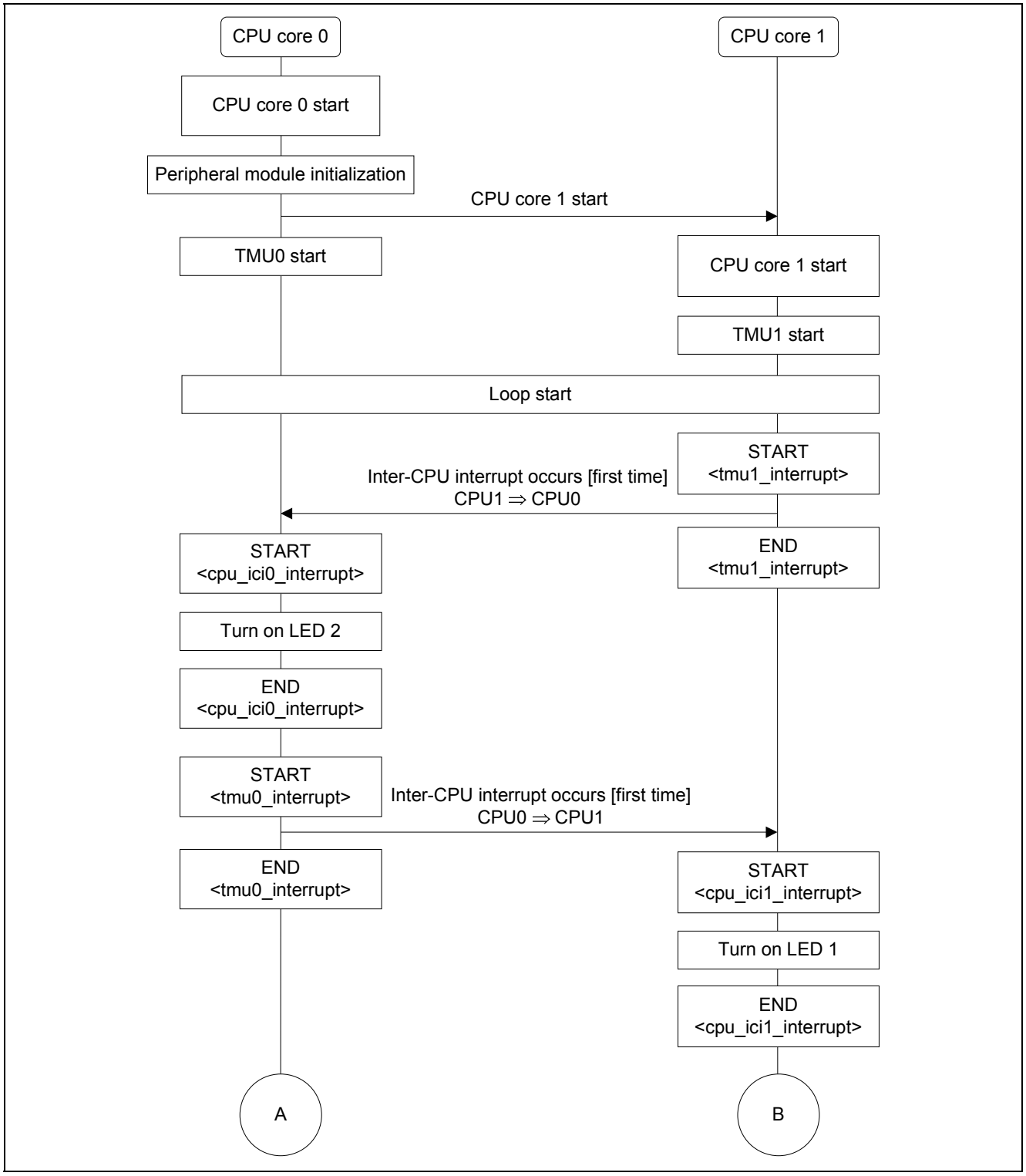


Figure 2.3 Sample Program Sequence 1

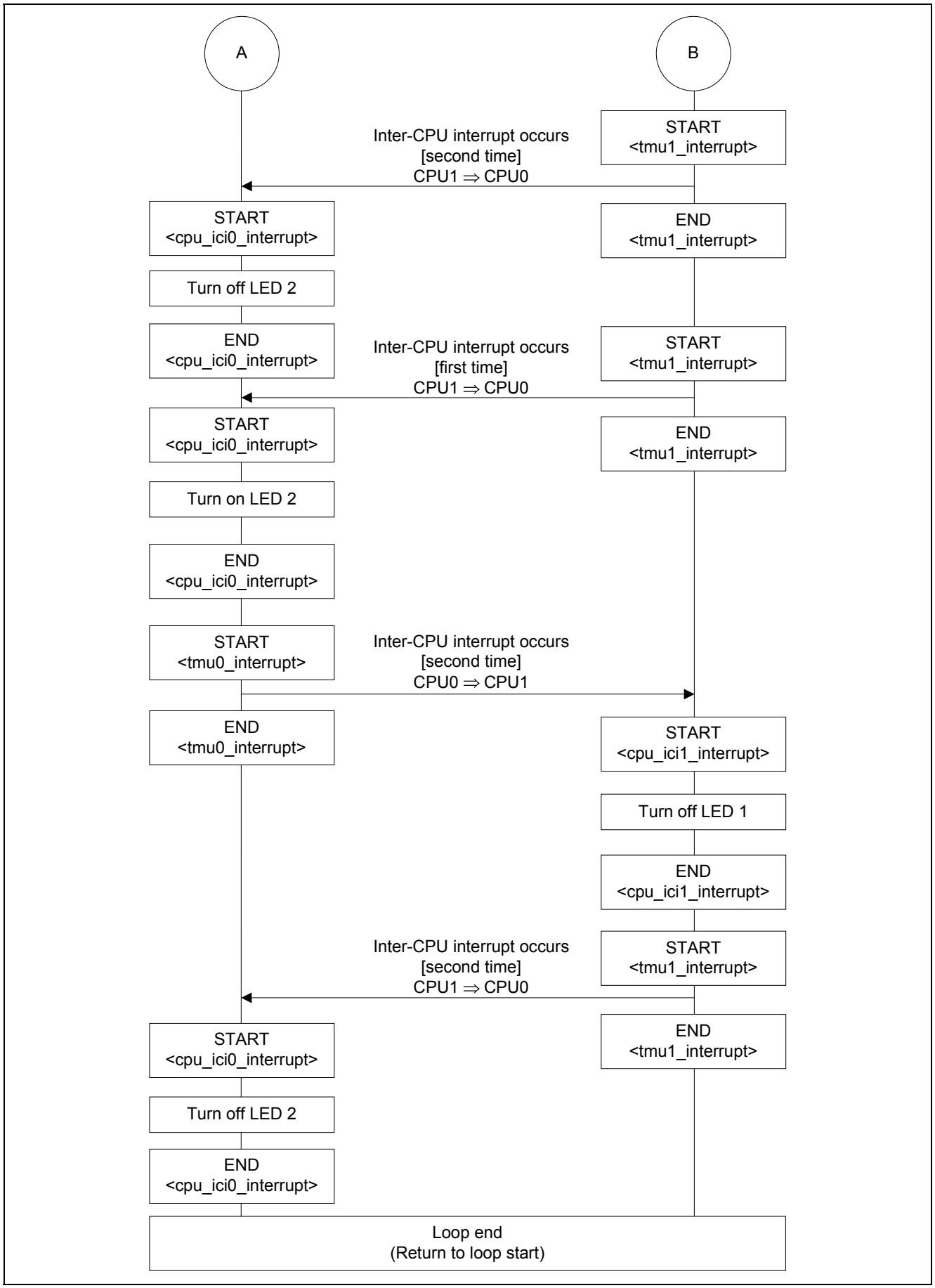


Figure 2.4 Sample Program Sequence 2

2.1.6 Sample Program Processing Order

The following figures show the sample program processing flow.

(1) CPU Core 0 Main Flowchart

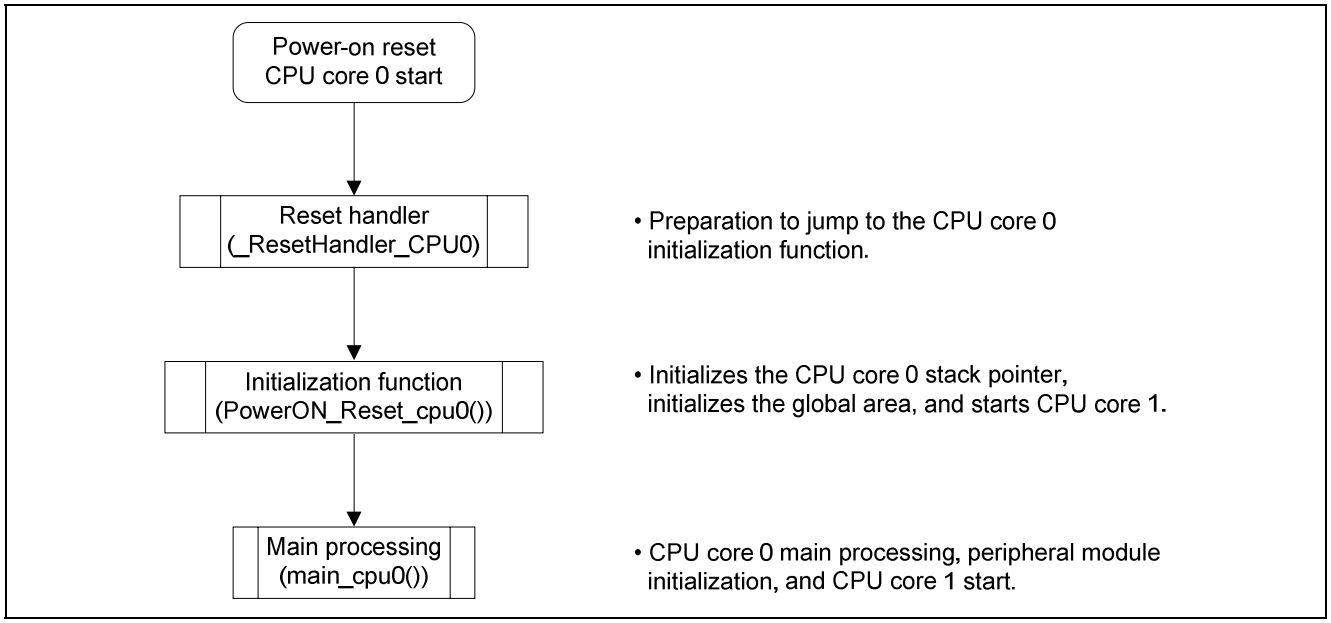


Figure 2.5 CPU Core 0 Main Flowchart

(2) CPU Core 0 Reset Handler (ResetHandler_CPU0)

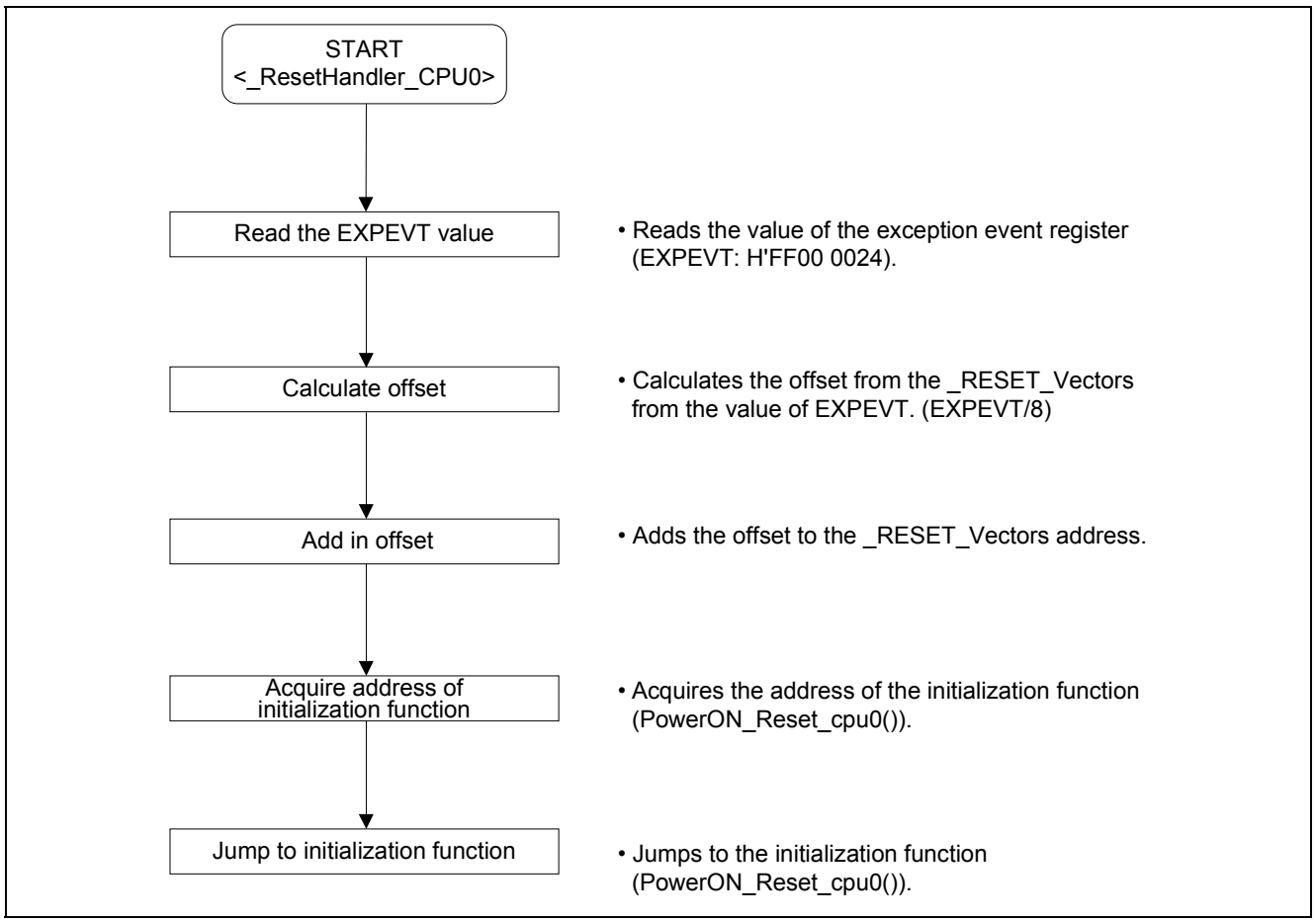


Figure 2.6 CPU Core 0 Reset Handler Flowchart

(3) CPU Core 0 Initialization (PowerON_Reset_CPU0())

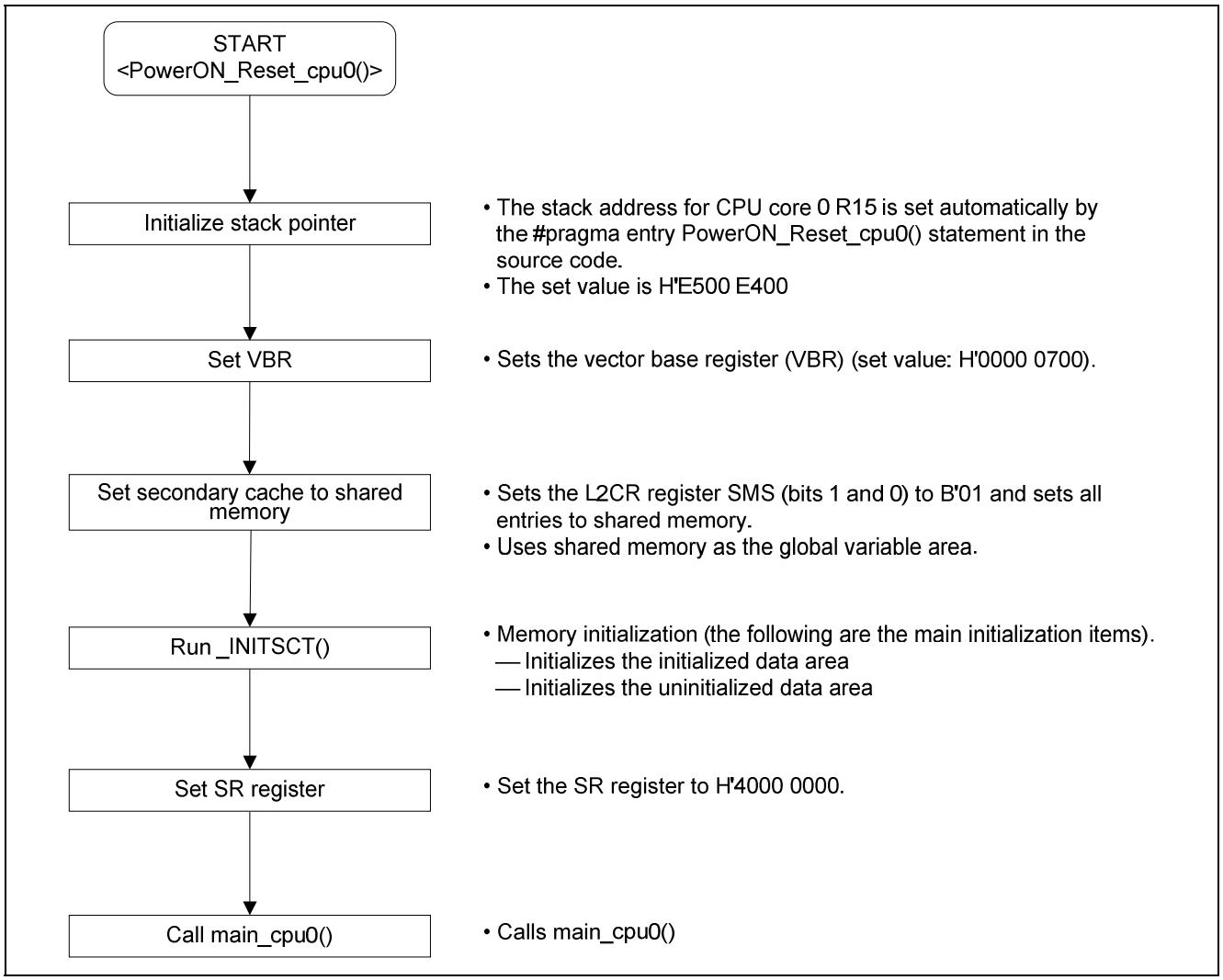
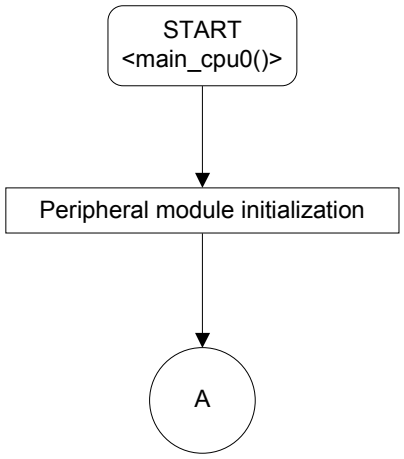


Figure 2.7 CPU Core 0 Initialization Flowchart

(4) CPU Core 0 Main Processing 1 (main_cpu0())



• TMU initialization
* Channel 0 *
[TCR0]:
UNIE (bit 5): B'1
[Function]
Enables underflow interrupts (TUNI)
TPSC2-0 (bits 2 to 0): B'100
[Function]
Selects counting by Pck/1024.

[TCNT0]: H'FE50 (Sets a 1-second interval.)
[TCOR0]: H'FE50 (Sets a 1-second interval.)
* Channel 1 *
[TCR1]:
UNIE (bit 5): B'1
[Function]
Enables underflow interrupts (TUNI)
TPSC2-0 (bits 2 to 0): B'100
[Function]
Selects counting by Pck/1024.

[TCNT1]: H'7F28 (Sets a 0.5-second interval.)
[TCOR1]: H'7F28 (Sets a 0.5-second interval.)

• INTC initialization
[INT2PRI01]:
TMU_ch0_2 (bits 28 to 24): B'00011 (channel 0)
TMU_ch0_2 (bits 20 to 16): B'00011 (channel 1)
[Function]
Sets the TMU channel 0 and 1 priorities to 3.
[C0INT2MSKCLR1]: MU_ch0_2 (bit 31): B'1
[Function]
Clears the CPU core 0 TMU channel 0 mask.
[C1INT2MSKCLR1]: TMU_ch0_2 (bit 30): B'1
[Function]
Clears the CPU core 1 TMU channel 1 mask.
[C0INTICIPRI]: ICIPRI0 (bits 3 to 0): B'0011
[Function]
Sets the CPU core 0 ICI0 priority to 3.
[C1INTICIPRI]: ICIPRI1 (bits 7 to 4): B'0011
[Function]
Sets the CPU core 1 ICI1 priority to 3.

• GPIO initialization
[PGCR]: H'D7FF
[Function] Sets PG5 and PG6 to output (LED output).

Figure 2.8 CPU Core 0 Main Processing 1 Flowchart

(5) CPU Core 0 Main Processing 2 (main_cpu0())

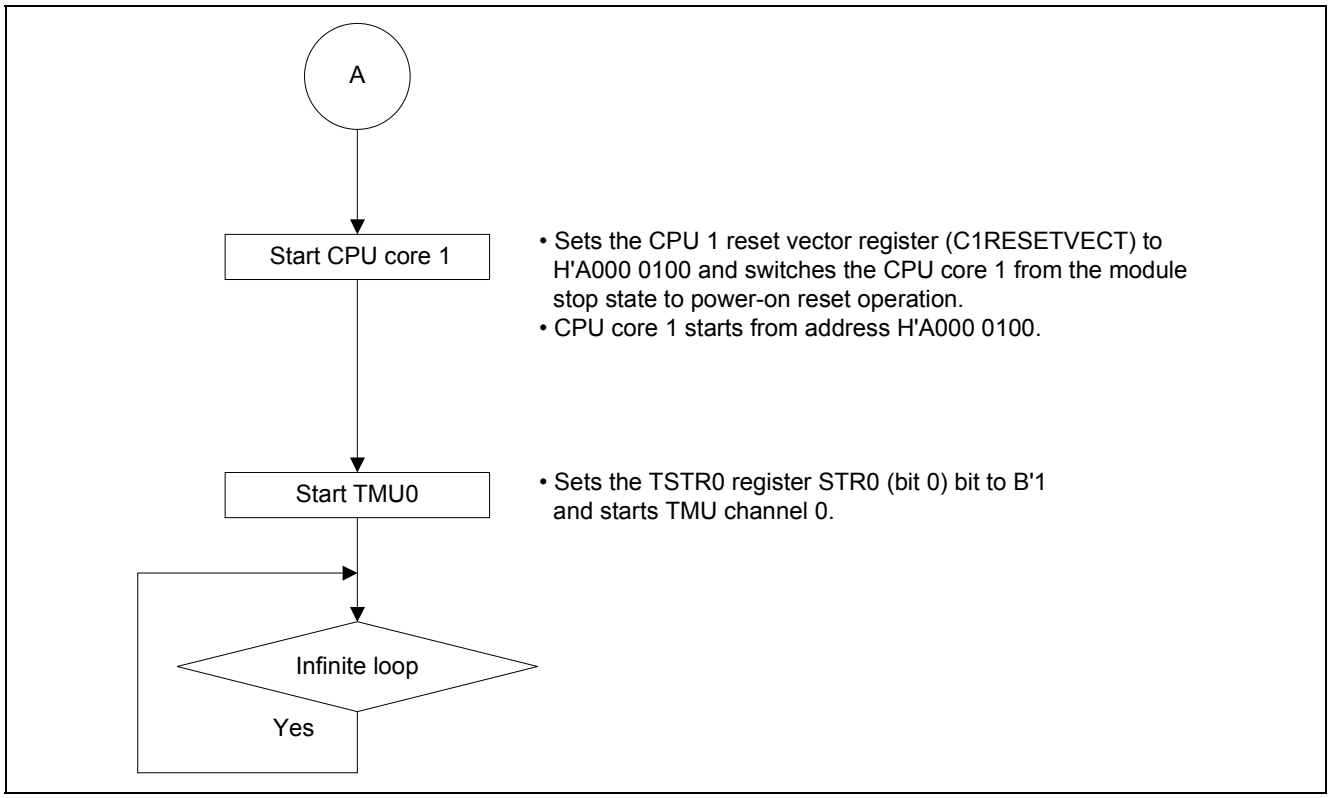


Figure 2.9 CPU Core 0 Main Processing 2 Flowchart

(6) CPU Core 0 TMU0 Interrupt Handling 1 (tmu0_interrupt())

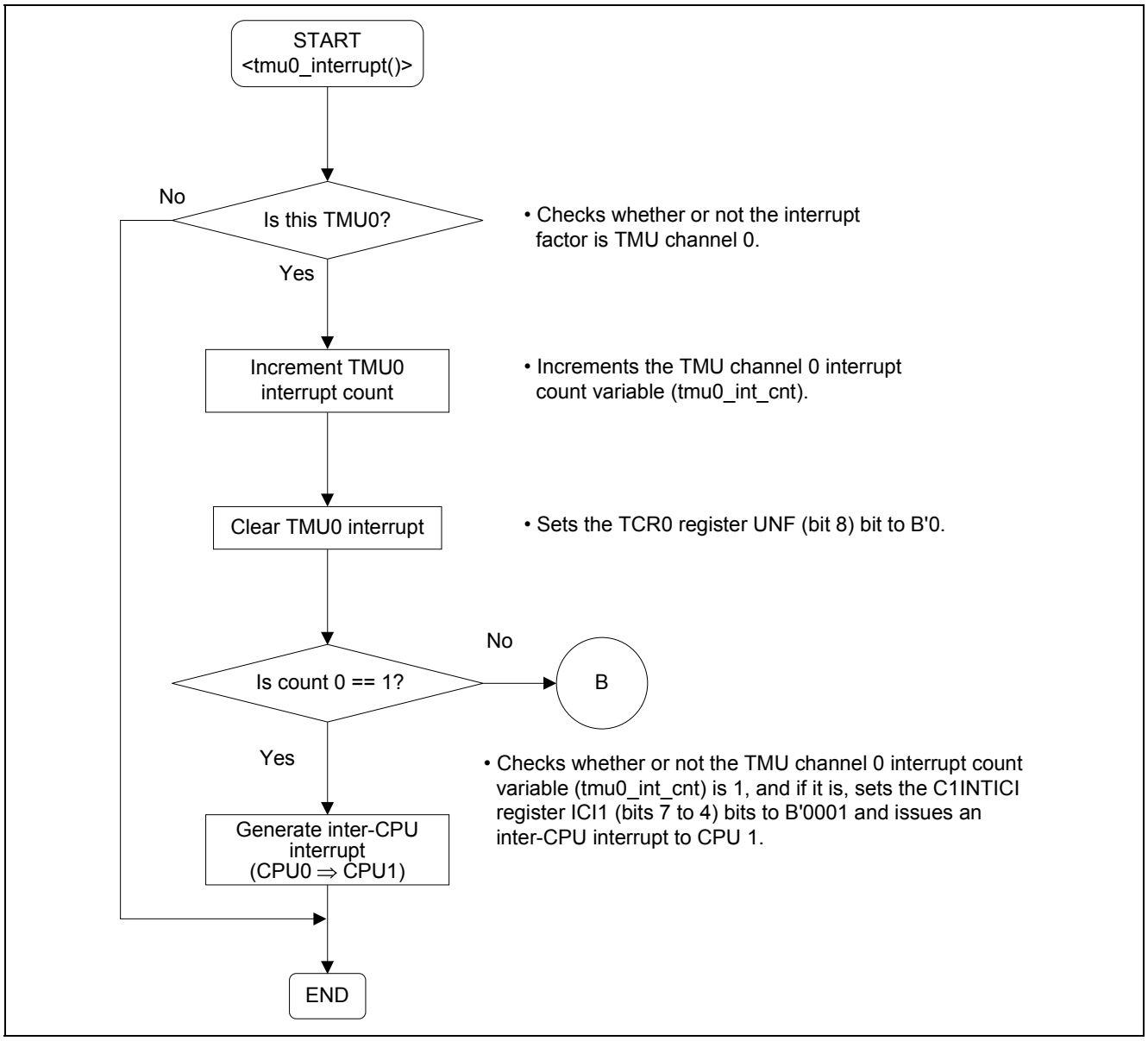


Figure 2.10 CPU Core 0 Main Processing 3 - TMU0 Interrupt 1 Flowchart

(7) CPU Core 0 TMU0 Interrupt Handling 2 (tmu0_interrupt())

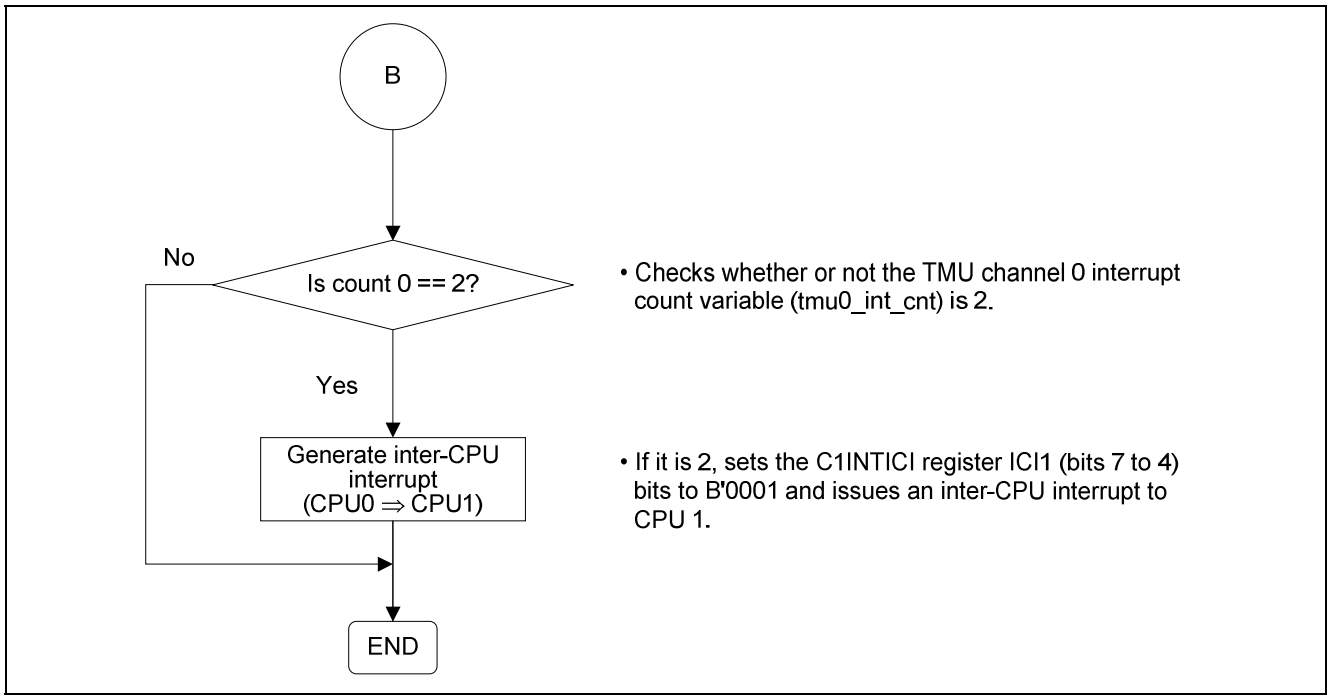


Figure 2.11 CPU Core 0 Main Processing - TMU0 Interrupt 2 Flowchart

(8) CPU Core 0 Inter-CPU Interrupt Handling 1 (cpu_ici0_interrupt())

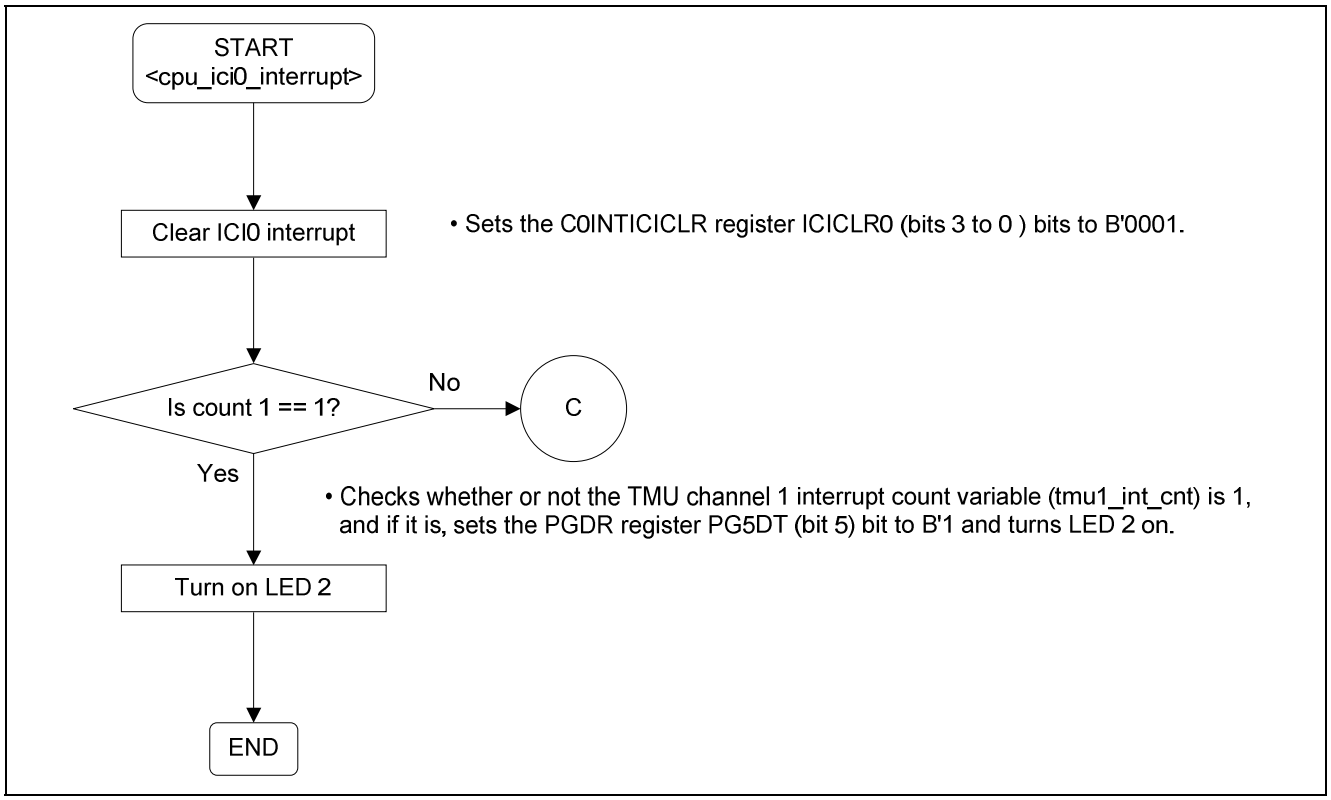


Figure 2.12 CPU Core 0 Main Processing - Inter-CPU Interrupt 1 Flowchart

(9) CPU core 0 Inter-CPU Interrupt Handling 2 (cpu_ici0_interrupt())

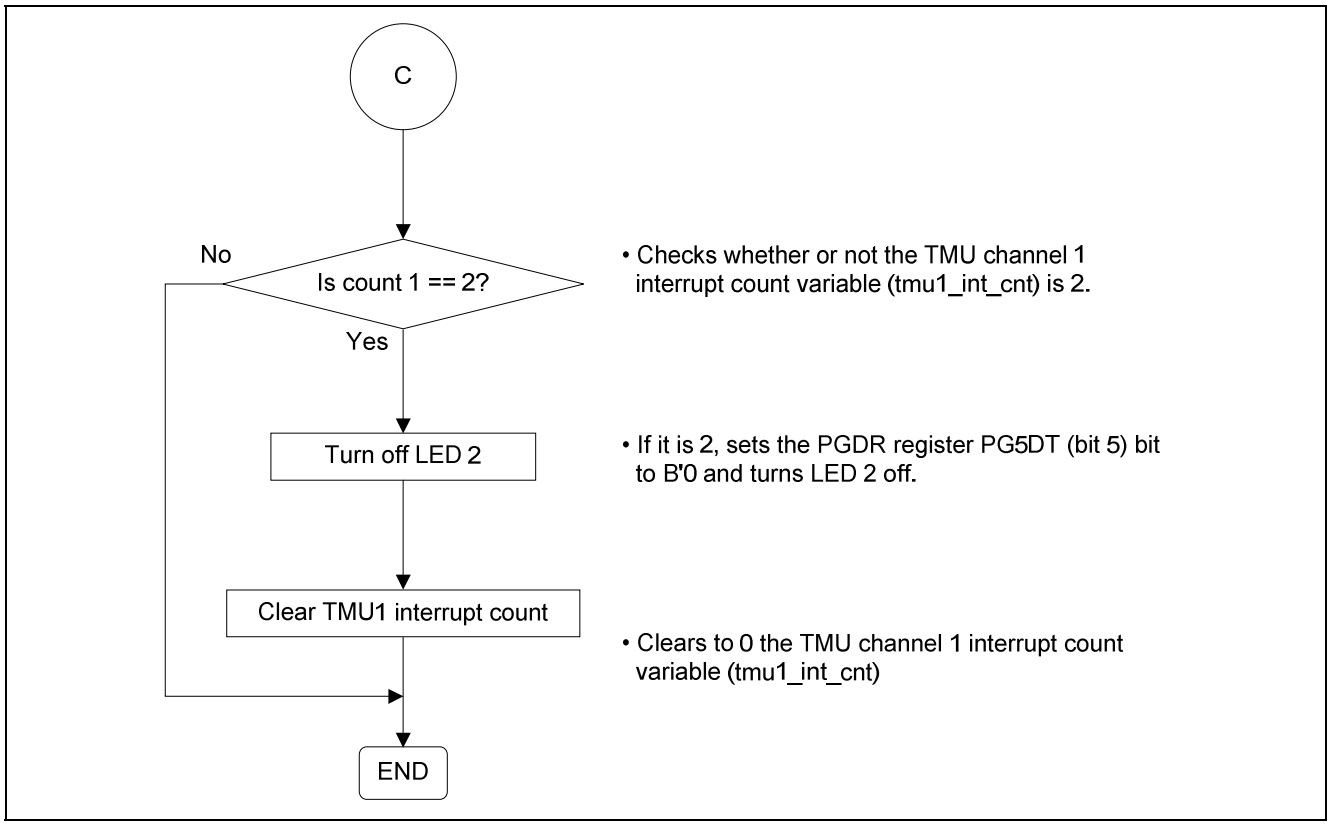


Figure 2.13 CPU Core 0 Main Processing - Inter-CPU Interrupt 2 Flowchart

(10) CPU Core 1 Main Flowchart

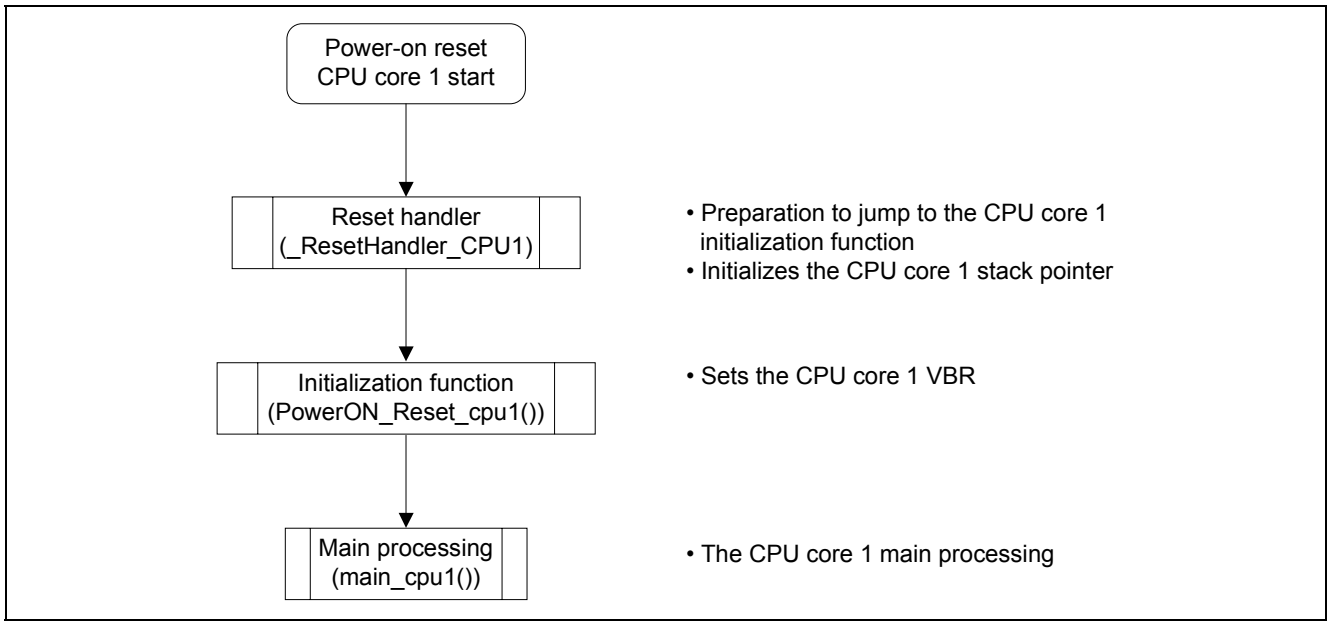


Figure 2.14 CPU Core 1 Main Flowchart

(11) CPU Core 1 Reset Handler (ResetHandler_CPU1)

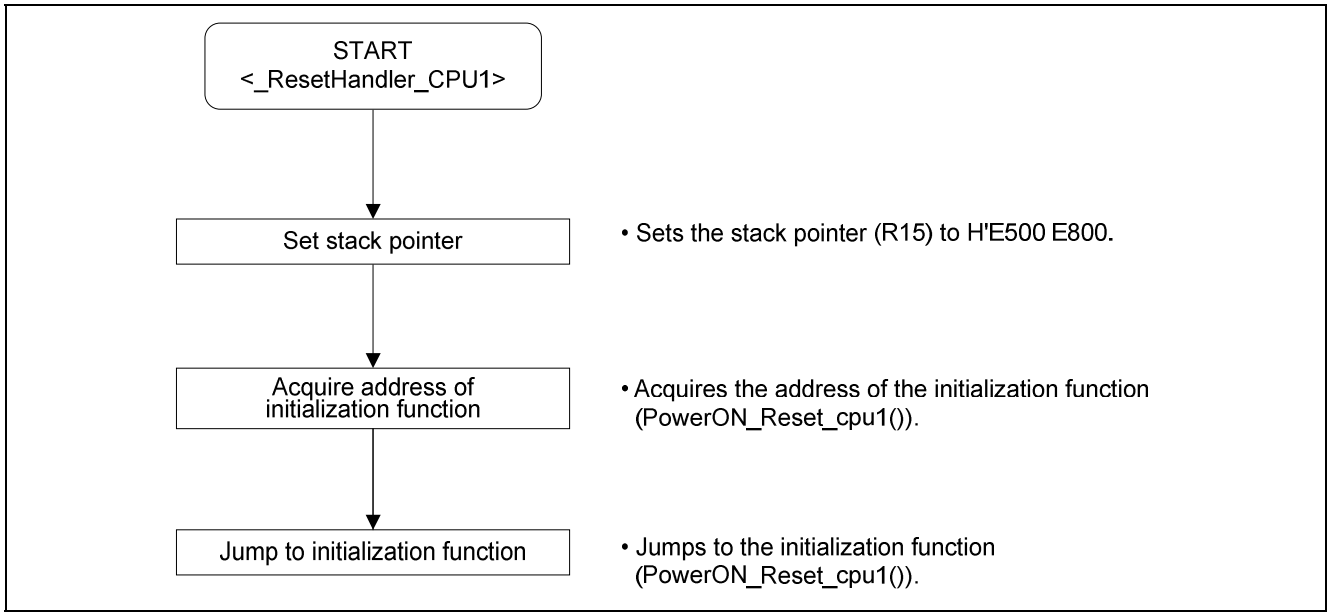


Figure 2.15 CPU Core 1 Reset Handler Flowchart

(12) CPU Core 1 Initialization (PowerON_Reset_CPU1())

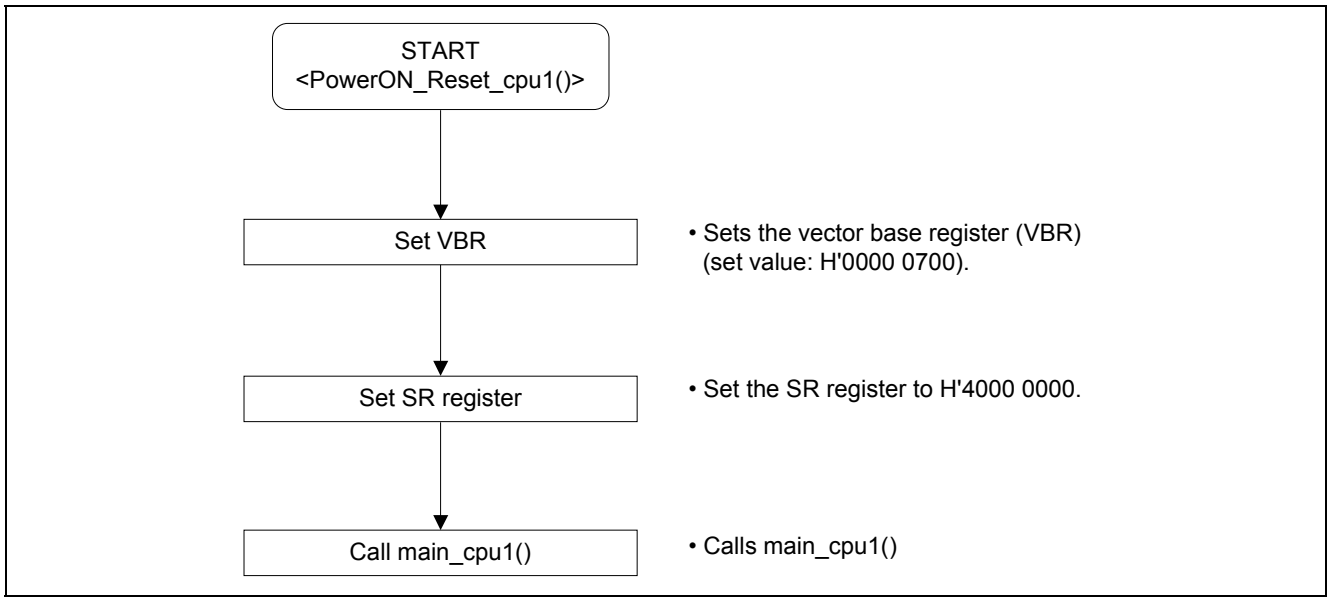


Figure 2.16 CPU Core 1 Initialization Flowchart

(13) CPU Core 1 Main Processing (main_cpu1())

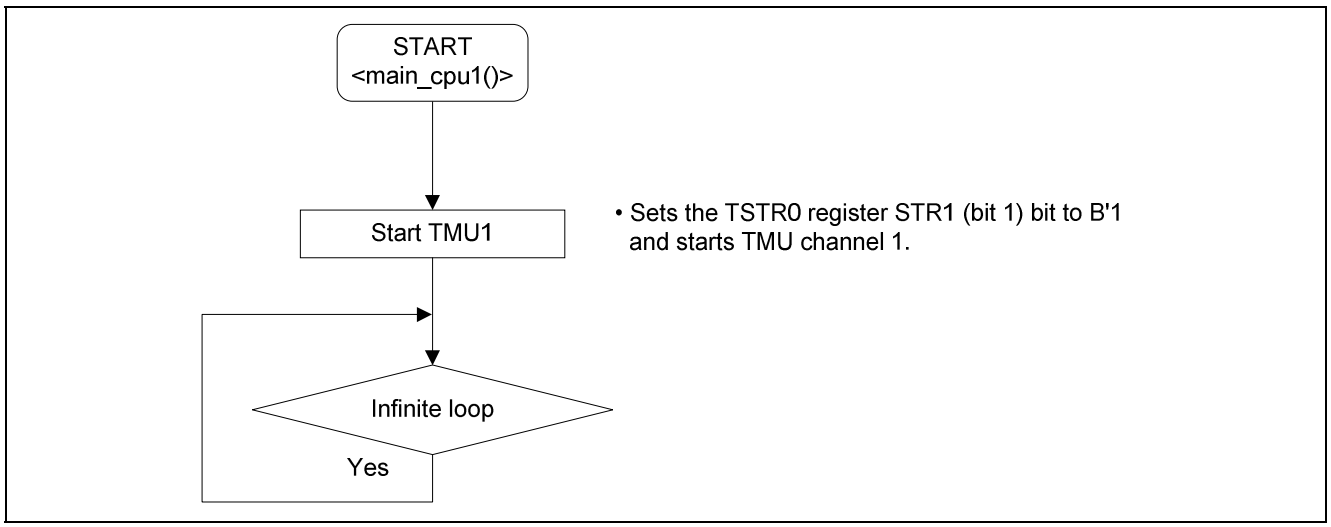


Figure 2.17 CPU Core 1 Main Processing Flowchart

(14) CPU Core 1 TMU1 Interrupt Handling 1 (tmu1_interrupt())

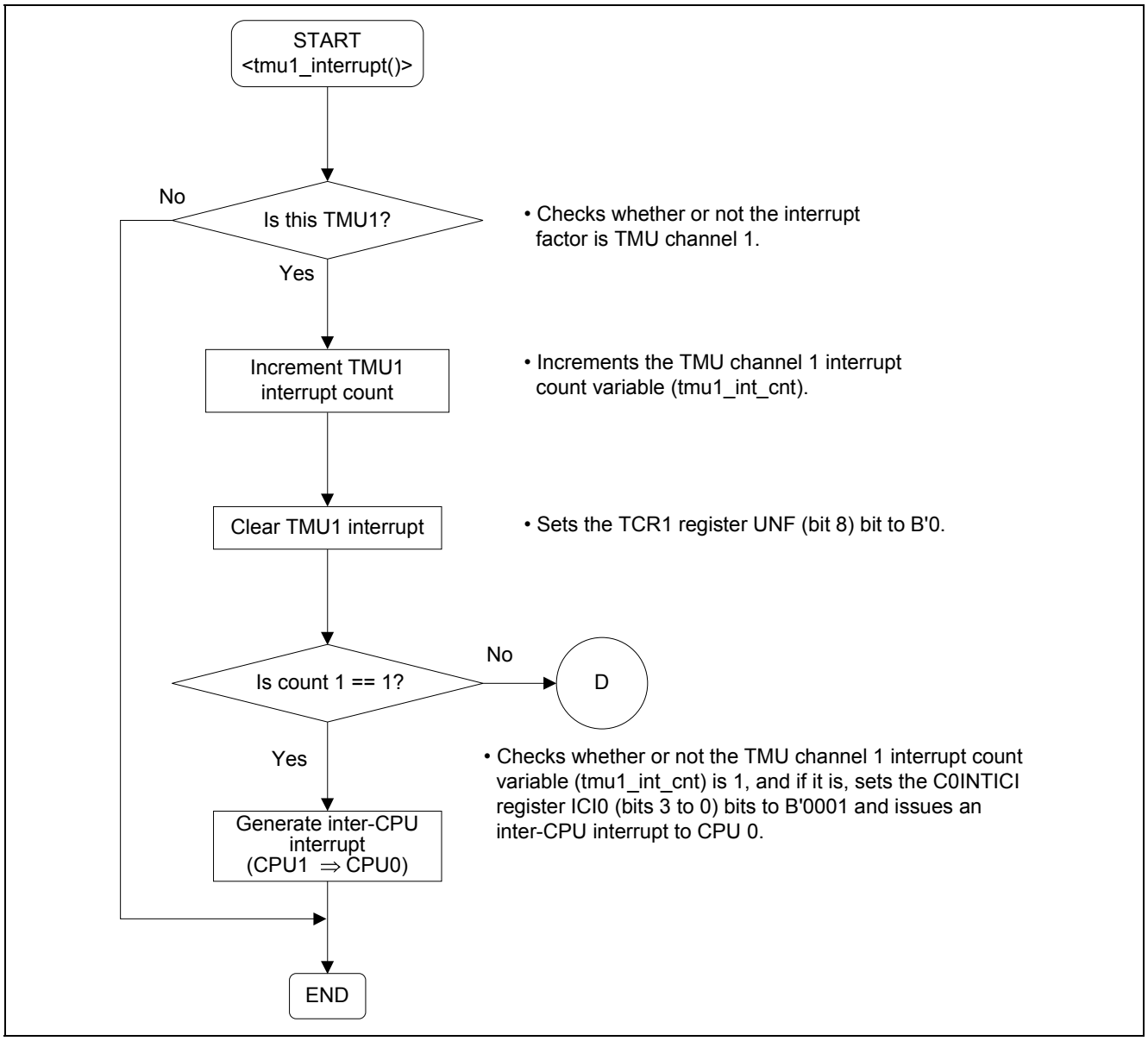


Figure 2.18 CPU Core 1 Main Processing Flowchart

(15) CPU Core 1 TMU1 Interrupt Handling 2 (tmu1_interrupt())

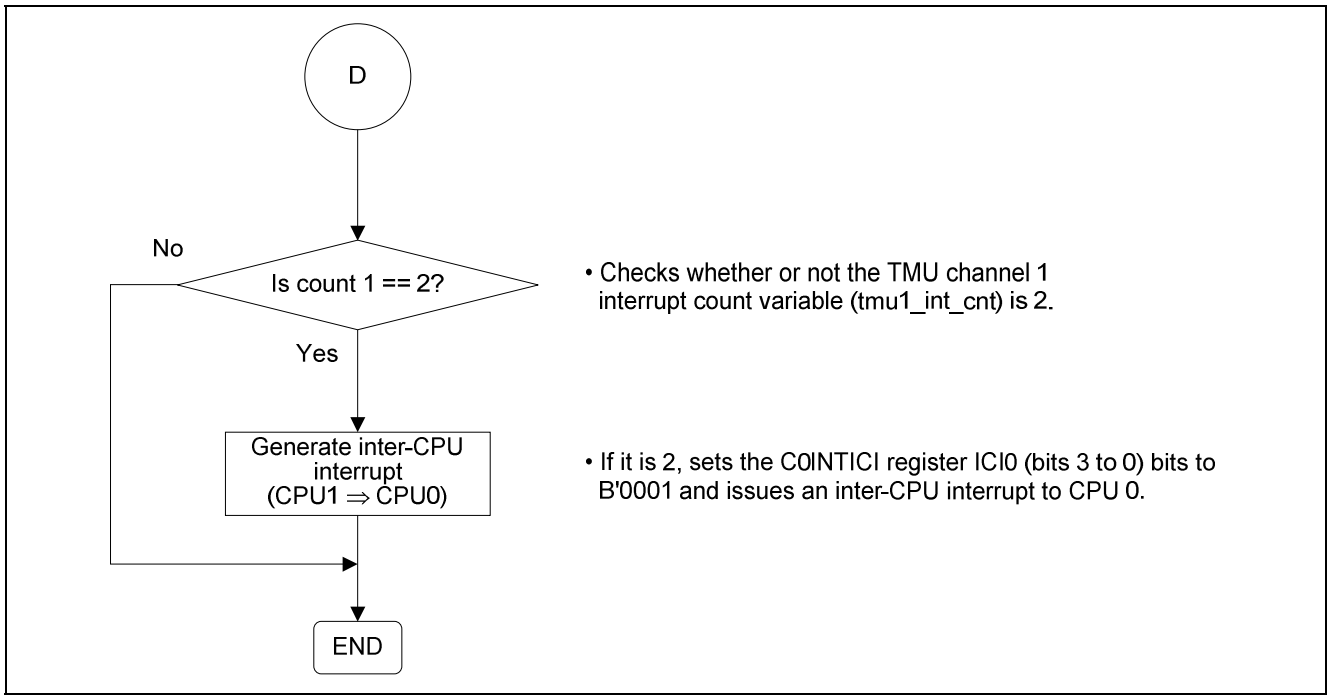


Figure 2.19 CPU Core 1 Main Processing 2 – TMU1 Interrupt 2 Flowchart

(16) CPU core 1 Inter-CPU Interrupt Handling 1 (cpu_ici1_interrupt())

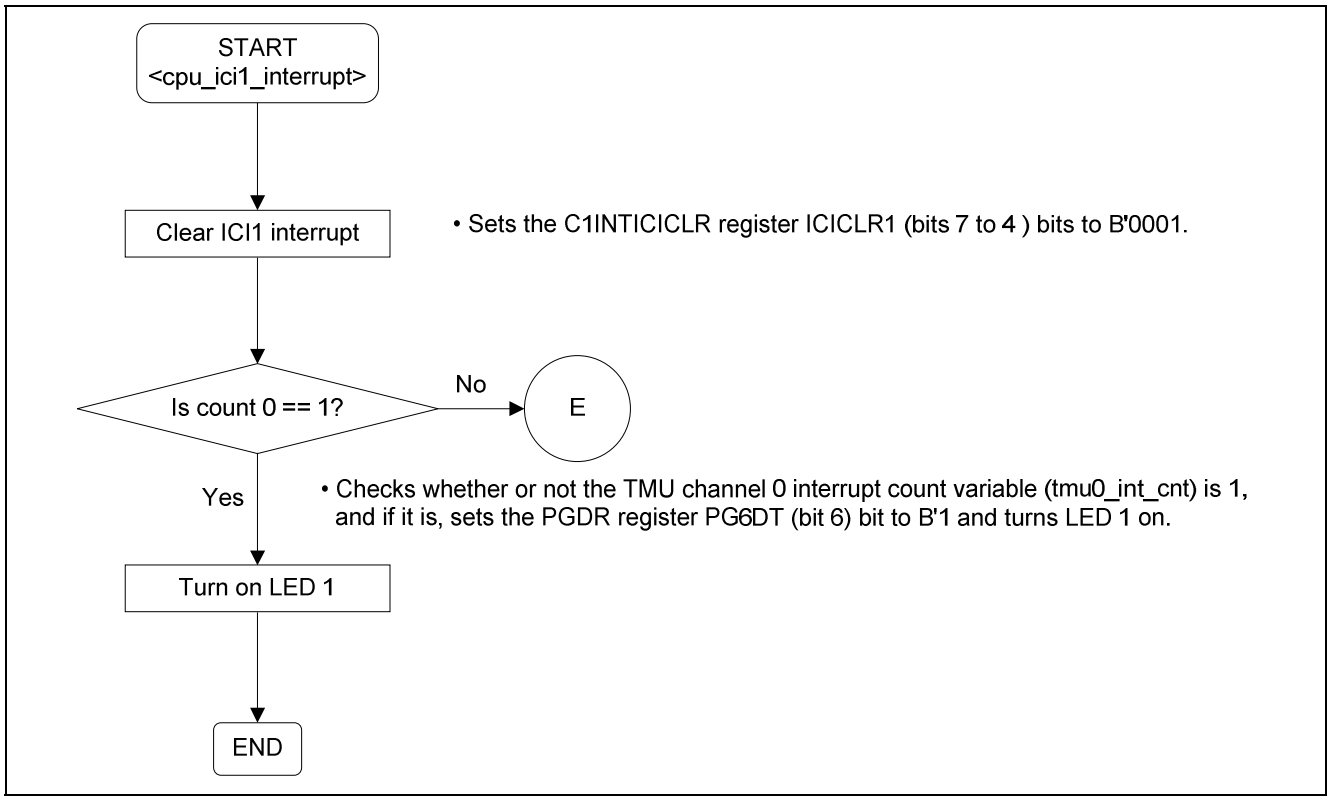


Figure 2.20 CPU Core 1 Main Processing 3 - Inter-CPU Interrupt 1 Flowchart

(17) CPU core 1 Inter-CPU Interrupt Handling 2 (cpu_ici1_interrupt())

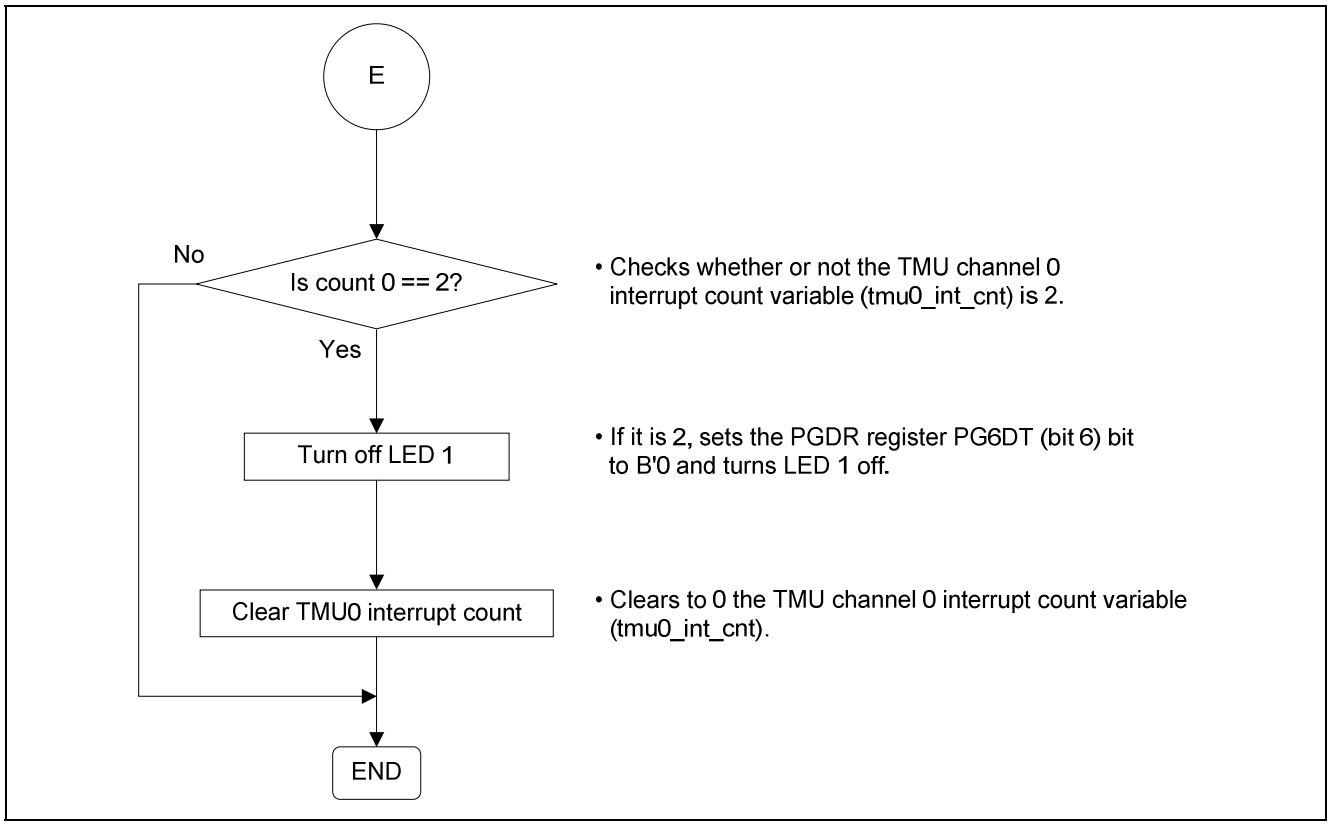


Figure 2.21 CPU Core 1 Main Processing 3 - Inter-CPU Interrupt 2 Flowchart

2.1.7 Sample Program Listing

The sample program is shown below.

```
001 /***********************************************************************
002 ;* DISCLAIMER
003 ;
004 ;* This software is supplied by Renesas Electronics Corporation. and is only
005 ;* intended for use with Renesas products. No other uses are authorized.
006 ;
007 ;* This software is owned by Renesas Electronics Corporation. and is protected under
008 ;* all applicable laws, including copyright laws.
009 ;
010 ;* THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
011 ;* REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
012 ;* INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
013 ;* PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
014 ;* DISCLAIMED.
015 ;
016 ;* TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
017 ;* ELECTRONICS CORPORATION. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
018 ;* FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
019 ;* FOR ANY REASON RELATED TO THE THIS SOFTWARE, EVEN IF RENESAS OR ITS
020 ;* AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
021 ;
022 ;* Renesas reserves the right, without notice, to make changes to this
023 ;* software and to discontinue the availability of this software.
024 ;* By using this software, you agree to the additional terms and
025 ;* conditions found by accessing the following link:
026 ;* http://www.renesas.com/disclaimer
027 ;***********************************************************************/
028 /* Copyright (C) 2011. Renesas Electronics Corporation., All Rights Reserved.*/
029 /*""FILE COMMENT""************ Technical reference data ****************
030 ;* System Name  : SH7786 INTC Sample Program
031 ;* File Name    : sh7786_intc_sample.c
032 ;* Abstract     : Main Program
033 ;* Version      : Ver 1.00
034 ;* Device       : SH7786
035 ;* Tool-Chain   : High-performance Embedded Workshop (Version 4.09.00.007)
036 ;*              : C/C++ Compiler Package for SuperH Family (V.9.3.2.0)
037 ;* OS           : None
038 ;* H/W Platform : SH-4A Board P/N:AP-SH4AD-0A (Manufacturer:ALPHA PROJECT)
039 ;* Description  : Main routines and TMU and ICI interrupt handlers
040 ;*              : for CPU CORE0 and CORE1
041 ;* Operation    :
042 ;* Limitation   :
043 ;*              :
044 ;***********************************************************************
045 ;* History      : 5.Aug.2011 Ver. 1.00 First Release
046 ;*""FILE COMMENT END""*************************************************/
047 /***********************************************************************/
048 /*                                                                     */
049 /*  FILE        :sh7786_intc_sample.c                                  */
050 /*  DATE        :Wed, Jun 22, 2011                                     */
051 /*  DESCRIPTION :Main Program                                          */
052 /*  CPU TYPE    :Other                                                 */
053 /*                                                                     */
054 /*  This file is generated by Renesas Project Generator (Ver.4.16).    */
```

```
055 /*                                                                      */
056 /************************************************************************/
057
058 //#include "typedefine.h"
059 #include "iodefine.h"
060 #include "typedefine.h"
061 #ifdef __cplusplus
062 //#include <ios>                    // Remove the comment when you use ios
063 //_SINT ios_base::Init::init_cnt;   // Remove the comment when you use ios
064 #endif
065
066 void main_cpu0(void);
067 void main_cpu1(void);
068 void tmu0_interrupt(void);
069 void tmu1_interrupt(void);
070 void cpu_ici0_interrupt(void);
071 void cpu_ici1_interrupt(void);
072 static void init_devices(void);
073
074 #ifdef __cplusplus
075 extern "C" {
076 void abort(void);
077 }
078 #endif
079
080 /*********************************/
081 /*                               */
082 /*     Sections for CPU Core 0   */
083 /*                               */
084 /*********************************/
085
086 #pragma section
087
088 static int tmu0_int_cnt = 0;
089 extern int tmu1_int_cnt;
090
091 /* main() for CPU CORE0 */
092 void main_cpu0(void)
093 {
094   tmu0_int_cnt = 0;
095   /* initialize peripherals */
096   init_devices();
097
098   /* Start CPU1 */
099   *(volatile _UINT *)0xFE401008 = 0xA0000100;      /* Set C1RESETVEC reg */
100   *(volatile _UINT *)0xFE401004 = 0x2; /* Set C1STBCR reg to CPU1 active mode */
101
102   TMU0.TSTR0.BIT.STR0 = 1;                         /* Start TMU0 */
103
104   /* Infinite loop */
105   while (1);
106 }
107
108 /* TMU channel0 interrupt handler */
109 void tmu0_interrupt(void)
110 {
111   if (INTC.C0INT2A0_1.BIT._TMU00 == 0)         /* If not TMU0 interrupt then just return */
```

```
112      return;
113
114   tmu0_int_cnt++;
115   TMU0.TCR0.BIT.UNF = 0;                  /* Clear TMU0 interrupt flag */
116
117   if (tmu0_int_cnt == 1)
118      INTC.C1INTICI.BIT.ICI1 = 0x1;        /* Set ICI1 bits */
119   else if (tmu0_int_cnt == 2) {
120      INTC.C1INTICI.BIT.ICI1 = 0x1;        /* Set ICI1 bits */
121   }
122 }
123
124 /* INTC ICI0 interrupt handler */
125 void cpu_ici0_interrupt(void)
126 {
127   INTC.C0INTICICLR.BIT.ICICLR0 = 0x1;     /* Clear ICICLR0 bits */
128
129   if (tmu1_int_cnt == 1)
130      GPIO.PGDR.BIT.PG5DT = 1;             /* Turn on LED2 */
131   else if (tmu1_int_cnt == 2) {
132      GPIO.PGDR.BIT.PG5DT = 0;             /* Turn off LED2 */
133      tmu1_int_cnt = 0;
134   }
135 }
136
137 /* initialize peripherals */
138 static void init_devices(void)
139 {
140   /* initialize TMU */
141   TMU0.TCR0.WORD = 0x0024;/* Set TMU0 interrupt enabled and select 1/1024 prescalar */
142   TMU0.TCNT0 = 0xFE50;                    /* Set TMU0 TCNT to 1 second */
143   TMU0.TCOR0 = 0xFE50;                    /* Set TMU0 TCOR to 1 second */
144   TMU0.TCR1.WORD = 0x0024;/* Set TMU1 interrupt enabled and select 1/1024 prescalar */
145   TMU0.TCNT1 = 0x7F28;                    /* Set TMU1 TCNT to 0.5 second */
146   TMU0.TCOR1 = 0x7F28;                    /* Set TMU1 TCOR to 0.5 second */
147
148   /* initialize INTC */
149   INTC.INT2PRI1.BIT.TMU00 = 0x3;          /* Set TMU0 priority to 3 */
150   INTC.C0INT2MSKCLR1.BIT._TMU00 = 1;      /* Clear CPU0 TMU0 interrupt mask */
151   INTC.INT2PRI1.BIT.TMU01 = 0x3;          /* Set TMU1 priority to 3 */
152   INTC.C1INT2MSKCLR1.BIT._TMU01 = 1;      /* Clear CPU1 TMU1 interrupt mask */
153   INTC.C0ICIPRI.BIT.ICIPRI0 = 0x3;        /* Set cpu0 ICI0 priority to 3 */
154   INTC.C1ICIPRI.BIT.ICIPRI1 = 0x3;        /* Set cpu1 ICI1 priority to 3 */
155
156   /* initialize GPIO */
157   GPIO.PGCR.WORD = 0xD7FF;                /* Set PG5 and PG6 to output mode */
158 }
159
160 /*********************************/
161 /*                               */
162 /*     Sections for CPU Core 1   */
163 /*                               */
164 /*********************************/
165
166 #pragma section cpu1
167
168 static int tmu1_int_cnt = 0;
```

```
169
170 /* main() for CPU CORE1 */
171 void main_cpu1(void)
172 {
173   tmu1_int_cnt = 0;
174
175   TMU0.TSTR0.BIT.STR1 = 1;                /* Start TMU1 */
176
177   /* Infinite loop */
178   while(1);
179 }
180
181 /* TMU channel1 interrupt handler */
182 void tmu1_interrupt(void)
183 {
184   if (INTC.C1INT2A0_1.BIT._TMU01 == 0)        /* If not TMU1 interrupt then just return */
185      return;
186
187   tmu1_int_cnt++;
188   TMU0.TCR1.BIT.UNF = 0;                  /* Clear TMU1 interrupt flag */
189
190   if (tmu1_int_cnt == 1)
191      INTC.C0INTICI.BIT.ICI0 = 0x1;        /* Set ICI0 bits */
192   else if (tmu1_int_cnt == 2) {
193      INTC.C0INTICI.BIT.ICI0 = 0x1;        /* Set ICI0 bits */
194   }
195 }
196
197 /* INTC ICI1 interrupt handler */
198 void cpu_ici1_interrupt(void)
199 {
200   INTC.C1INTICICLR.BIT.ICICLR1 = 0x1;     /* Clear ICICLR1 bits */
201
202   if (tmu0_int_cnt == 1)
203      GPIO.PGDR.BIT.PG6DT = 1;             /* Turn on LED1 */
204   else if (tmu0_int_cnt == 2) {
205      GPIO.PGDR.BIT.PG6DT = 0;             /* Turn off LED1 */
206      tmu0_int_cnt = 0;
207   }
208 }
209
210 #ifdef __cplusplus
211 void abort(void)
212 {
213
214 }
215 #endif
216
217
```

2.1.8 Programming Notes

Keep the following points in mind when creating programs that use the INTC inter-CPU interrupt.

1. CPU 0 and 1 operating states
   The CPU 0 and 1 operating states can be checked from the states of the CPU0/1 standby control register (C0STBCR and C1STBCR) MSTP0 and MSTP1 bits as described in the SH7786 Group Hardware Manual, section 20., Low Power Modes. Also, note that while CPU 0 operates starting after a power-on reset has been cleared, CPU 1 will be in the module stop state at that point. Set the MSTP bit to 0 to start CPU 1.
2. CPU0/1 standby control register (C0STBCR and C1STBCR) RESET0/1 bits
   While a software reset is issued to start or restart CPU 0 or 1, this reset should be issued with CPU 0 or 1 in the module stop state (when the corresponding MSTP0/1 bit is 1). Also, do not set the RESET0/1 bits to 1 when the corresponding MSTP0/1 bit is 0. See the SH7786 Group Hardware Manual, section 20.5.3, CPU Core Module Stop State, for details.

3. Reference Documents

- Software Manual
  SH4-A Software Manual (REJ09B0003)
  (The latest version can be downloaded from the Renesas Electronics Web site.)
- Hardware Manual
  SH7786 Group Hardware Manual (REJ09B0501)
  (The latest version can be downloaded from the Renesas Electronics Web site.)
- Renesas Microcontroller Development Environment Manuals
  SuperH - Multicore Microcontroller E-10A Emulator - User's Manual (HS0005KCU04H) (REJ10J2095)
  SuperH - Multicore Microcontroller E-10A Emulator - User's Manual Supplement - Supplementary Documentation when Using the SH7786 (REJ10J2598)

Website and Support

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

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

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Jan.20.12	—	First edition issued

General Precautions in the Handling of MPU/MCU Products

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

1. Handling of Unused Pins
   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.
   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.
2. Processing at Power-on
   The state of the product is undefined at the moment when power is supplied.
   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.
3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.
4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.
   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.
5. Differences between Products
   Before changing from one product to another, i.e. to one with a different type number, confirm that the change will not lead to problems.
   — The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

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.

RENESAS

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

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

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

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

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

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

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

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

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

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

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

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

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

Colophon 1.1