RL78/G11

R01AN3519EJ0100
Rev.1.00
Nov. 30, 2016

CPU Clock Changing and Standby Settings (C Language) CC-RL

Introduction

This application note describes how to change the RL78/G11's CPU clock and set it to standby (changing operation modes).

This application uses switch input to change the CPU clock and the operation mode, while controlling 5 LEDs to indicate the CPU clock status and the operation mode.

Target Device

RL78/G11

When applying the sample program covered in this application note to another microcomputer, modify the program according to the specifications for the target microcomputer and conduct an extensive evaluation of the modified program.

Contents

1. Specifications

This application describes how to switch the CPU clock and operation mode using switch input, as shown in Figure 1.1 Operating Mode Status Transition Diagram.

In addition, the application controls five LEDs to indicate the status of the CPU clock and the operation mode.

The Peripheral Functions and Applications used in this application note, Operating Mode Status Transition Diagram, and Operation Modes and Corresponding LED Status are show in Table 1.1, Figure 1.1, and Table 1.2, correspondingly.

Table 1.1 Peripheral Functions and Applications

Peripheral Function	Application
Port output	Controls LEDs (LED1-LED5) connected to pins P00, P01, P54, P55, P56.
External interrupt	Interrupt (INTP0) that detects a pin input edge according to switch input (SW1).
12-bit interval timer	Interrupt (INTIT) that detects an interval signal from the 12-bit interval timer
A/D converter	Converts analog signal input level of the P20/ANI0 pin.

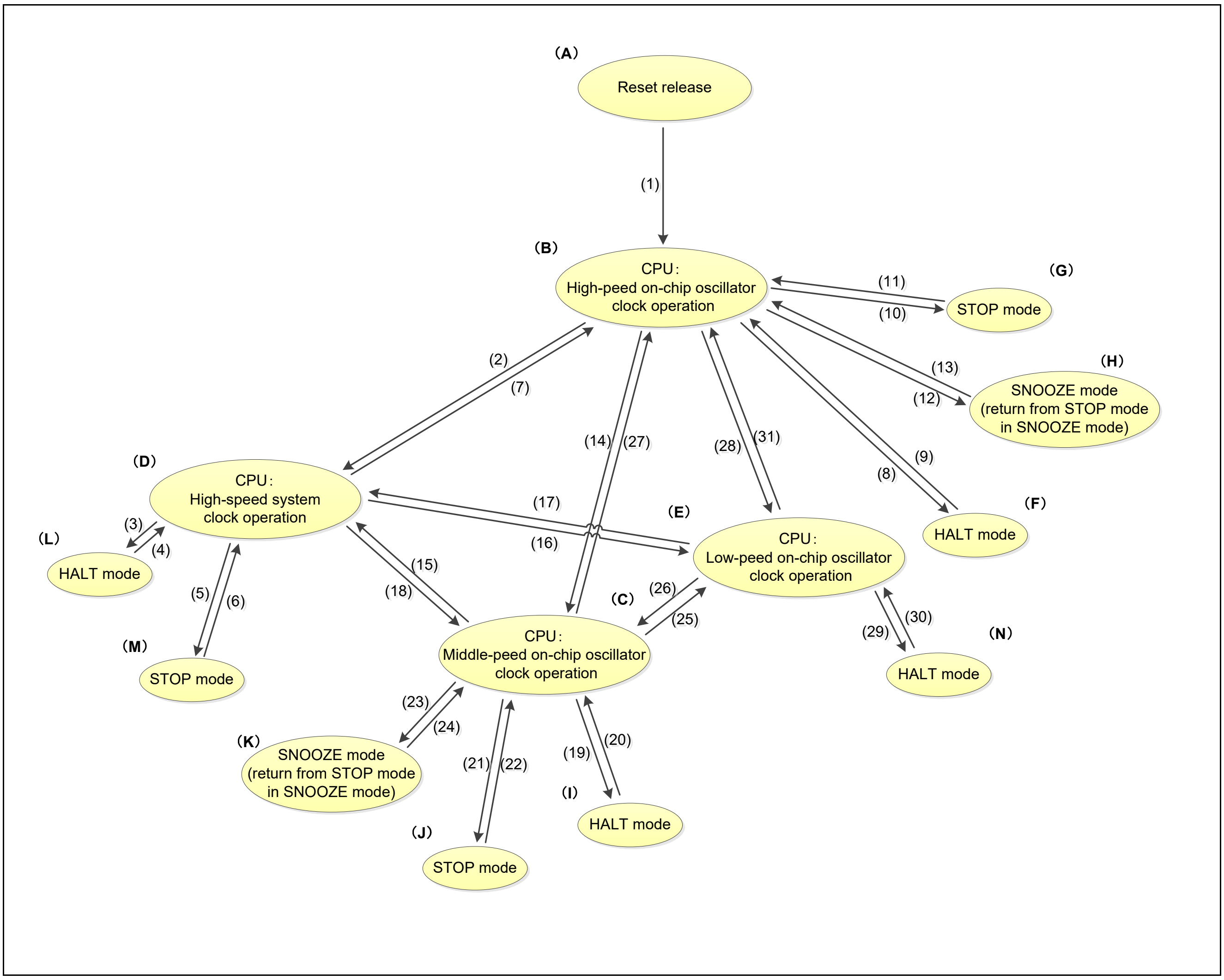


Figure 1.1 Operating Mode Status Transition Diagram

Table 1.2 Operation Modes and Corresponding LED Status

CPU/Peripheral Hardware Clock (f_{CLK})	Operation mode	LED Status				
		LED1	LED2	LED3	LED4	LED5
High-speed on-chip oscillator clock (f_{IH})	Normal operation mode	ON	ON	OFF	OFF	ON
	HALT mode	OFF	ON	OFF	OFF	ON
	SNOOZE mode	ON	OFF	OFF	OFF	ON
	STOP mode	OFF	OFF	OFF	OFF	ON
Middle-speed on-chip oscillator clock (f_{IM})	Normal operation mode	ON	ON	OFF	ON	OFF
	HALT mode	OFF	ON	OFF	ON	OFF
	SNOOZE mode	ON	OFF	OFF	ON	OFF
	STOP mode	OFF	OFF	OFF	ON	OFF
Low-speed on-chip oscillator clock (f_{IL})	Normal operation mode	ON	ON	OFF	ON	ON
	HALT mode	OFF	OFF	OFF	ON	ON
High-speed system clock (f_{MX})	Normal operation mode	ON	ON	ON	OFF	ON
	HALT mode	OFF	ON	ON	OFF	ON
	STOP mode	OFF	OFF	ON	OFF	ON

1.1 CPU Clock Changes

This section describes the special function register (SFR) settings required for changing the CPU clock.

- Changing from high-speed on-chip oscillator clock to middle-speed on-chip oscillator clock
- Changing from high-speed on-chip oscillator clock to low-speed on-chip oscillator clock
- Changing from high-speed on-chip oscillator clock to high-speed system clock
- Changing from middle-speed on-chip oscillator clock to high-speed on-chip oscillator clock
- Changing from middle-speed on-chip oscillator clock to low-speed on-chip oscillator clock
- Changing from middle-speed on-chip oscillator clock to high-speed system clock
- Changing from low-speed on-chip oscillator clock to high-speed on-chip oscillator clock
- Changing from low-speed on-chip oscillator clock to middle-speed on-chip oscillator clock
- Changing from low-speed on-chip oscillator clock to high-speed system clock
- Changing from high-speed system clock to high-speed on-chip oscillator clock
- Changing from high-speed system clock to middle-speed on-chip oscillator clock
- Changing from high-speed system clock to low-speed on-chip oscillator clock

1.1.1 Changing from high-speed on-chip oscillator clock to middle-speed on-chip oscillator clock

When changing the CPU clock from the high-speed on-chip oscillator clock to the middle-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the middle-speed on-chip oscillator clock to f_{CLK} using the system clock control register (CKC). Confirm that the status of the main on-chip oscillator clock status has switched to the middle-speed on-chip oscillator clock, and then stop the high-speed on-chip oscillator.

① Set (1) the MIOEN bit of the CSC register, and then start oscillating the middle-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	x	1	0	0	0	0	**1**	0

② Use a software wait to wait for the oscillation of the middle-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 4µs) using the timer function or another function.

③ Set(1) the MCM1 bit of the CKC register to specify the middle-speed on-chip oscillator clock as the main on-chip oscillator clock.

	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
	0	0	0	0	0	0	0	**1**

④ Confirm that the MCS1 bit of the CKC register has changed to 1, set (1) the HIOSTOP bit and stop the oscillating the high-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	x	1	0	0	0	0	1	**1**

Register setting values:
x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

1.1.2 Changing from high-speed on-chip oscillator clock to low-speed on-chip oscillator clock

When changing the CPU clock from the high-speed on-chip oscillator clock to the low-speed on-chip oscillator clock, start the oscillation using the subsystem clock select register (CKSEL), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the low-speed on-chip oscillator clock to f_{CLK} using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock status has switched to the subsystem clock, and then stop the high-speed on-chip oscillator.

① Set (1) the SELLOSC bit of the CKSEL register, and then start oscillating the low-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CKSEL	0	0	0	0	0	0	0	SELLOSC
	0	0	0	0	0	0	0	**1**

② Use a software wait to wait for the oscillation of the low-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 210µs) using the timer function or another function. In this application note is always to operate the low-speed on-chip oscillator. For this reason, it does not perform oscillation stabilization wait of the low-speed on-chip oscillator.

③ Set(1) the CSS bit of the CKC register to specify the subsystem clock as the CPU/peripheral hardware clock.

	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
	0	**1**	0	0	0	0	0	0

④ Confirm that the CLS bit of the CKC register has changed to 1, set (1) the HIOSTOP bit and stop the oscillating the high-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	x	1	0	0	0	0	x	**1**

Register setting values:
x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

1.1.3 Changing from high-speed on-chip oscillator clock to high-speed system clock

When changing the CPU clock from the high-speed on-chip oscillator clock to the high-speed system clock, set the oscillator and start oscillation using the following registers: clock operation mode control register (CMC), oscillation stabilization time select register (OSTS), clock operation status control register (CSC). Next, wait for the oscillation to stabilize using the oscillation stabilization time counter status register (OSTC).

After the oscillation stabilizes, set the high-speed system clock to f_{CLK} using the system clock control register (CKC).

Confirm that the status of the main system clock has changed to the high-speed system clock, and then stop the high-speed on-chip oscillator.

① Set (1) the OSCSEL bit of the CMC register (when fx > 10MHz, set (1) the AMPH bit) to operate the X1 oscillator. Set (1) the EXCLK bit and OSCSEL bit when using the external clock.

	7	6	5	4	3	2	1	0
CMC	EXCLK	OSCSEL	0	0	0	0	0	AMPH
	0/1	**1**	0	0	0	0	0	**0/1**

AMPH bit: clear to 0 when the X1 oscillation clock is 10 MHZ or lower.

② Using the OSTS register, select the oscillation stabilization time of the X1 oscillation circuit. This setting does not have to exist at the time the external clock is used.
Example: Set the following values for a wait of at least 102µs based on a 10 MHz resonator.

	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0
	0	0	0	0	0	**0**	**1**	**0**

③ Clear (0) the MSTOP bit of the CSC register to start oscillating the X1 oscillator. After the external clock is input to the external clock signal, to clear (0) the MSTOP bit.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	0	1	0	0	0	0	x	0

④ Use the OSTC register to wait for oscillation of the X1 oscillator to stabilize. External clock is not required oscillation stabilization wait.
Example: Wait until the bits reach the following values for a wait of at least 102µs based on a 10 MHz resonator.

	7	6	5	4	3	2	1	0
OSTC	MOST8	MOST9	MOST10	MOST11	MOST13	MOST15	MOST17	MOST18
	1	**1**	**1**	**0**	**0**	**0**	**0**	**0**

Register setting values:
x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

⑤ Set(1) the MCM0 bit of the CKC register to specify the high-speed system clock as the main system clock.

	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
	0	0	0	**1**	0	0	0	0

⑥ Confirm that the MCS bit of the CKC register has changed to 1, set (1) the HIOSTOP bit and stop the oscillating the high-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	0	1	0	0	0	0	x	**1**

Register setting values:
x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

1.1.4 Changing from middle-speed on-chip oscillator clock to high-speed on-chip oscillator clock

When changing the CPU clock from the middle-speed on-chip oscillator clock to the high-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the high-speed on-chip oscillator clock to f_{CLK} using the system clock control register (CKC). Confirm that the status of the main on-chip oscillator clock status has switched to the high-speed on-chip oscillator clock, and then stop the middle-speed on-chip oscillator.

① Clear (0) the HIOSTOP bit of the CSC register, and then start oscillating the high-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	x	1	0	0	0	0	1	**0**

② Use a software wait to wait for the oscillation of the high-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 65µs) using the timer function or another function.

③ Clear(0) the MCM1 bit of the CKC register to specify the high-speed on-chip oscillator clock as the main on-chip oscillator clock.

	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
	0	0	0	0	0	0	1	**0**

④ Confirm that the MCS1 bit of the CKC register has changed to 0, clear (0) the MIOEN bit and stop the oscillating the middle-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	x	1	0	0	0	0	**0**	0

Register setting values:
x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

1.1.5 Changing from middle-speed on-chip oscillator clock to low-speed on-chip oscillator clock

When changing the CPU clock from the middle-speed on-chip oscillator clock to the low-speed on-chip oscillator clock, start the oscillation using the subsystem clock select register (CKSEL), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the low-speed on-chip oscillator clock to f_{CLK} using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock status has switched to the subsystem clock, and then stop the middle-speed on-chip oscillator.

① Set (1) the SELLOSC bit of the CKSEL register, and then start oscillating the low-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CKSEL	0	0	0	0	0	0	0	SELLOSC
	0	0	0	0	0	0	0	**1**

② Use a software wait to wait for the oscillation of the low-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 210μs) using the timer function or another function. In this application note is always to operate the low-speed on-chip oscillator. For this reason, it does not perform oscillation stabilization wait of the low-speed on-chip oscillator.

③ Set(1) the CSS bit of the CKC register to specify the subsystem clock as the CPU/peripheral hardware clock.

	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
	0	**1**	0	0	0	0	1	1

④ Confirm that the CLS bit of the CKC register has changed to 1, clear (0) the MIOEN bit and stop the oscillating the middle-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	x	1	0	0	0	0	**0**	x

Register setting values:

x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

1.1.6 Changing from middle-speed on-chip oscillator clock to high-speed system clock

When changing the CPU clock from the middle-speed on-chip oscillator clock to the high-speed system clock, set the oscillator and start oscillation using the following registers: clock operation mode control register (CMC), oscillation stabilization time select register (OSTS), clock operation status control register (CSC). Next, wait for the oscillation to stabilize using the oscillation stabilization time counter status register (OSTC).

After the oscillation stabilizes, set the high-speed system clock to f_{CLK} using the system clock control register (CKC).

Confirm that the status of the main system clock has changed to the high-speed system clock, and then stop the middle-speed on-chip oscillator.

① Set (1) the OSCSEL bit of the CMC register (when fx > 10MHz, set (1) the AMPH bit) to operate the X1 oscillator. Set (1) the EXCLK bit and OSCSEL bit when using the external clock.

	7	6	5	4	3	2	1	0
CMC	EXCLK	OSCSEL	0	0	0	0	0	AMPH
	0/1	**1**	0	0	0	0	0	**0/1**

AMPH bit: clear to 0 when the X1 oscillation clock is 10 MHZ or lower.

② Using the OSTS register, select the oscillation stabilization time of the X1 oscillation circuit. This setting does not have to exist at the time the external clock is used.
Example: Set the following values for a wait of at least 102μs based on a 10 MHz resonator.

	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0
	0	0	0	0	0	**0**	**1**	**0**

③ Clear (0) the MSTOP bit of the CSC register to start oscillating the X1 oscillator. After the external clock is input to the external clock signal, to clear (0) the MSTOP bit.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	0	1	0	0	0	0	1	x

④ Use the OSTC register to wait for oscillation of the X1 oscillator to stabilize. External clock is not required oscillation stabilization wait.
Example: Wait until the bits reach the following values for a wait of at least 102μs based on a 10 MHz resonator.

	7	6	5	4	3	2	1	0
OSTC	MOST8	MOST9	MOST10	MOST11	MOST13	MOST15	MOST17	MOST18
	1	**1**	**1**	**0**	**0**	**0**	**0**	**0**

Register setting values:
x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

⑤ Set(1) the MCM0 bit of the CKC register to specify the high-speed system clock as the main system clock.

	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
	0	0	0	**1**	0	0	1	1

⑥ Confirm that the MCS bit of the CKC register has changed to 1, clear (0) the MIOEN bit and stop the oscillating the middle-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	0	1	0	0	0	0	**0**	x

Register setting values:
x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

1.1.7 Changing from low-speed on-chip oscillator clock to high-speed on-chip oscillator clock

When changing the CPU clock from the low-speed on-chip oscillator clock to the high-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the high-speed on-chip oscillator clock to f_{CLK} using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock status has switched to the main system clock, and then stop the low-speed on-chip oscillator.

① Clear (0) the HIOSTOP bit of the CSC register, and then start oscillating the high-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	x	1	0	0	0	0	x	**0**

② Use a software wait to wait for the oscillation of the high-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 65µs) using the timer function or another function.

③ Clear(0) the CSS bit of the CKC register to specify the main system clock as the CPU/peripheral hardware clock.

	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
	1	**0**	0	0	0	0	0	0

④ Confirm that the CLS bit of the CKC register has changed to 0, clear (0) the SELLOSC bit of the subsystem clock select register(CKSEL) and stop the oscillating the low-speed on-chip oscillator. In this application note, for WUTMMCK0 bit of the subsystem clock supply mode control register(OSMC) is 1, does not stop the low-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CKSEL	0	0	0	0	0	0	0	SELLOSC
	0	0	0	0	0	0	0	**0**

Note: Changing the value of the MCM0 bit is prohibited while the CPU/peripheral hardware clock is operating with the subsystem clock

Register setting values:
x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

1.1.8 Changing from low-speed on-chip oscillator clock to middle-speed on-chip oscillator clock

When changing the CPU clock from the low-speed on-chip oscillator clock to the middle-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the middle-speed on-chip oscillator clock to f_{CLK} using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock status has switched to the main system clock, and then stop the low-speed on-chip oscillator.

① Set (1) the MIOEN bit of the CSC register, and then start oscillating the middle-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	x	1	0	0	0	0	**1**	x

② Use a software wait to wait for the oscillation of the middle-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 4μs) using the timer function or another function.

③ Clear(0) the CSS bit of the CKC register to specify the main system clock as the CPU/peripheral hardware clock.

	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
	1	**0**	0	0	0	0	1	1

④ Confirm that the CLS bit of the CKC register has changed to 0, clear (0) the SELLOSC bit of the subsystem clock select register(CKSEL) and stop the oscillating the low-speed on-chip oscillator. In this application note, for WUTMMCK0 bit of the subsystem clock supply mode control register(OSMC) is 1, does not stop the low-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CKSEL	0	0	0	0	0	0	0	SELLOSC
	0	0	0	0	0	0	0	**0**

Note: Changing the value of the MCM0 bit is prohibited while the CPU/peripheral hardware clock is operating with the subsystem clock

Register setting values:
x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

1.1.9 Changing from low-speed on-chip oscillator clock to high-speed system clock

When changing the CPU clock from the low-speed on-chip oscillator clock to the high-speed system clock, set the oscillator and start oscillation using the following registers: clock operation mode control register (CMC), oscillation stabilization time select register (OSTS), clock operation status control register (CSC). Next, wait for the oscillation to stabilize using the oscillation stabilization time counter status register (OSTC).

After the oscillation stabilizes, set the high-speed system clock to f_{CLK} using the system clock control register (CKC).

Confirm that the status of the CPU/peripheral hardware clock has changed to the main system clock, and then stop the low-speed on-chip oscillator.

① Set (1) the OSCSEL bit of the CMC register (when fx > 10MHz, set (1) the AMPH bit) to operate the X1 oscillator. Set (1) the EXCLK bit and OSCSEL bit when using the external clock.

	7	6	5	4	3	2	1	0
CMC	EXCLK	OSCSEL	0	0	0	0	0	AMPH
	0/1	**1**	0	0	0	0	0	**0/1**

AMPH bit: clear to 0 when the X1 oscillation clock is 10 MHZ or lower.

② Using the OSTS register, select the oscillation stabilization time of the X1 oscillation circuit. This setting does not have to exist at the time the external clock is used.

Example: Set the following values for a wait of at least 102 μs based on a 10 MHz resonator.

	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0
	0	0	0	0	0	**0**	**1**	**0**

③ Clear (0) the MSTOP bit of the CSC register to start oscillating the X1 oscillator. After the external clock is input to the external clock signal, to clear (0) the MSTOP bit.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	0	1	0	0	0	0	x	x

④ Use the OSTC register to wait for oscillation of the X1 oscillator to stabilize. External clock is not required oscillation stabilization wait.

Example: Wait until the bits reach the following values for a wait of at least 102 μs based on a 10 MHz resonator.

	7	6	5	4	3	2	1	0
OSTC	MOST8	MOST9	MOST10	MOST11	MOST13	MOST15	MOST17	MOST18
	1	**1**	**1**	**0**	**0**	**0**	**0**	**0**

Register setting values:

x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

⑤ Clear(0) the CSS bit of the CKC register to specify the main system clock as the CPU/peripheral hardware clock.

	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
	1	**0**	1	1	0	0	0	0

⑥ Confirm that the CLS bit of the CKC register has changed to 0, clear (0) the SELLOSC bit of the subsystem clock select register(CKSEL) and stop the oscillating the low-speed on-chip oscillator. In this application note, for WUTMMCK0 bit of the subsystem clock supply mode control register(OSMC) is 1, does not stop the low-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CKSEL	0	0	0	0	0	0	0	SELLOSC
	0	0	0	0	0	0	0	**0**

Note: Changing the value of the MCM0 bit is prohibited while the CPU/peripheral hardware clock is operating with the subsystem clock

Register setting values:
x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

1.1.10 Changing from high-speed system clock to high-speed on-chip oscillator clock

When changing the CPU clock from the high-speed system clock to the high-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the high-speed on-chip oscillator clock to f_{CLK} using the system clock control register (CKC). Confirm that the status of the main system clock status has switched to the main on-chip oscillator clock, and then stop the X1 oscillator.

① Clear (0) the HIOSTOP bit of the CSC register, and then start oscillating the high-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	0	1	0	0	0	0	x	**0**

② Use a software wait to wait for the oscillation of the high-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 65μs) using the timer function or another function.

③ Clear(0) the MCM0 bit of the CKC register to specify the main on-chip oscillator clock as the main system clock.

	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
	0	0	1	**0**	0	0	0	0

④ Confirm that the MCS bit of the CKC register has changed to 0, set (1) the MSTOP bit and stop the oscillating the X1 oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	1	1	0	0	0	0	x	0

Register setting values:
x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

1.1.11 Changing from high-speed system clock to middle-speed on-chip oscillator clock

When changing the CPU clock from the high-speed system clock to the middle-speed on-chip oscillator clock, start the oscillation using the clock operation status control register (CSC), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the middle-speed on-chip oscillator clock to f_{CLK} using the system clock control register (CKC). Confirm that the status of the main system clock status has switched to the main on-chip oscillator clock, and then stop the X1 oscillator.

① Set (1) the MIOEN bit of the CSC register, and then start oscillating the middle-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	0	1	0	0	0	0	**1**	x

② Use a software wait to wait for the oscillation of the middle-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 4μs) using the timer function or another function.

③ Clear(0) the MCM0 bit of the CKC register to specify the main on-chip oscillator clock as the main system clock.

	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
	0	0	1	**0**	0	0	1	1

④ Confirm that the MCS bit of the CKC register has changed to 0, set (1) the MSTOP bit and stop the oscillating the X1 oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	1	1	0	0	0	0	1	x

Register setting values:
x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

1.1.12 Changing from high-speed system clock to low-speed on-chip oscillator clock

When changing the CPU clock from the high-speed system clock to the low-speed on-chip oscillator clock, start the oscillation using the subsystem clock select register (CKSEL), then wait for the oscillation to stabilize using the timer function or another function. After the oscillation stabilization time has elapsed, set the low-speed on-chip oscillator clock to f_{CLK} using the system clock control register (CKC). Confirm that the status of the CPU/peripheral hardware clock status has switched to the subsystem clock, and then stop the X1 oscillator.

① Set (1) the SELLOSC bit of the CKSEL register, and then start oscillating the low-speed on-chip oscillator.

	7	6	5	4	3	2	1	0
CKSEL	0	0	0	0	0	0	0	SELLOSC
	0	0	0	0	0	0	0	**1**

② Use a software wait to wait for the oscillation of the low-speed on-chip oscillator to stabilize. Count the wait time (oscillation stabilization time: 210μs) using the timer function or another function. In this application note is always to operate the low-speed on-chip oscillator. For this reason, it does not perform oscillation stabilization wait of the low-speed on-chip oscillator.

③ Set(1) the CSS bit of the CKC register to specify the subsystem clock as the CPU/peripheral hardware clock.

	7	6	5	4	3	2	1	0
CKC	CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
	0	**1**	1	1	0	0	0	0

④ Confirm that the CLS bit of the CKC register has changed to 1, set (1) the MSTOP bit and stop the oscillating the X1 oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
	1	1	0	0	0	0	x	x

Register setting values:

x: unused bit; blank space; unchanged bit; -: reserved bits or unassigned bit

2. Operation Confirmation Conditions

The sample code accompanying this application note has been run and confirmed under the conditions below.

Table 2.1 Operation Confirmation Conditions

Item	Contents
MCU used	RL78/G11 (R5F1056A)
Operating frequencies	● High-speed on-chip oscillator clock: 24MHz ● Middle-speed on-chip oscillator clock: 4MHz ● Low-speed on-chip oscillator clock: 15kHz ● High-speed system clock: 20MHz ● CPU/peripheral hardware clock: 24MHz/20MHz/4MHz/15kHz[Note]
Operating voltage	3.3V (operating range 2.9V to 5.5V) LVD operations (V_{LVD}): reset mode 2.81V (2.76V to 2.87V)
Integrated development environment (CS+)	CS+ for CC V4.01.00 from Renesas Electronics Corp.
C compiler (CS+)	CC-RL V1.02.00 from Renesas Electronics Corp.
Integrated development environment (e^2 studio)	e^2 studio V5.2.0.020 from Renesas Electronics Corp.
C compiler (e^2 studio)	CC-RL V1.02.00 from Renesas Electronics Corp.

Note: CPU/peripheral hardware clock settings are changed in the application.

3. Related Application Notes

Application notes related to this document are shown below. Please refer to these as needed.

RL78/I1D Operation State Switching CC-RL (R01AN3095E) Application Note

4. Hardware Explanation

4.1 Hardware Configuration Example

Figure 4.1 shows an example of the hardware configuration used in this application note.

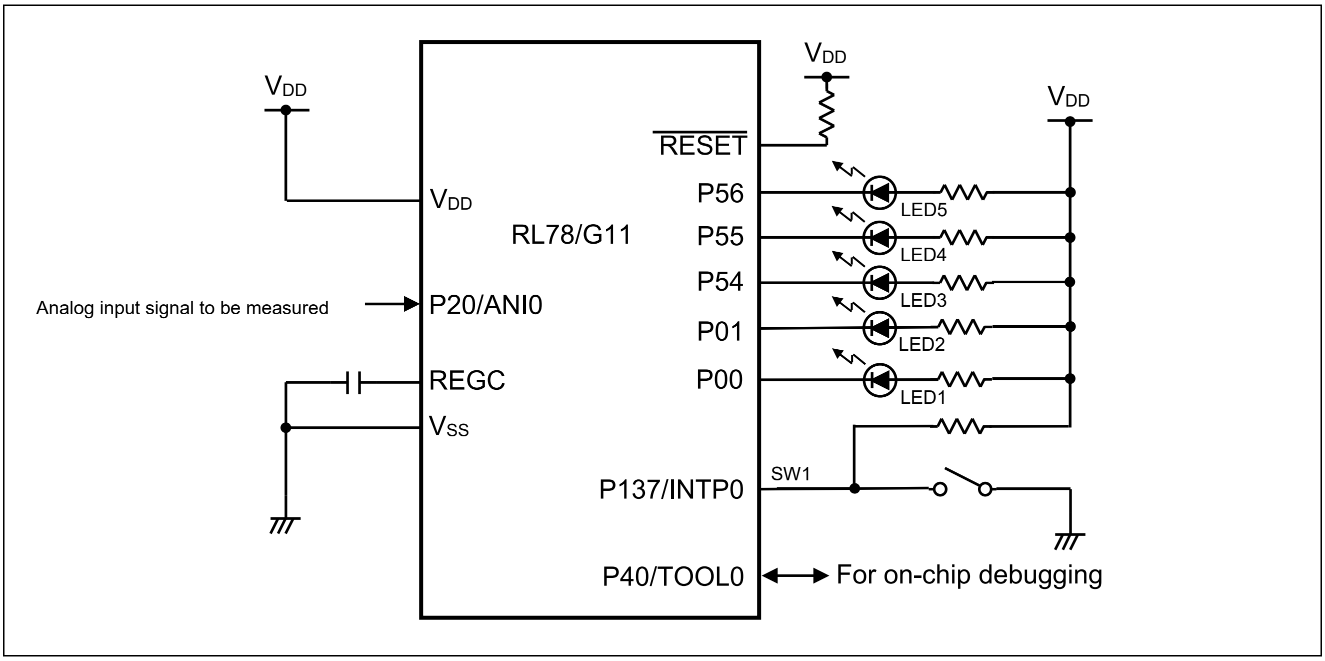


Figure 4.1 Hardware Configuration

Note: 1.This simplified circuit diagram was created to show an overview of connections only.
When actually designing your circuit, make sure the design includes sufficient pin processing and meets electrical characteristic requirements.
(Connect each input-only port to V_{DD} or V_{SS} through a resistor.)
2. If a pin name starts with EV_{SS}, connect the pin to V_{SS}, if it starts with EV_{DD}, connect it to V_{DD}.
3. Make V_{DD} higher than the RESET release voltage (V_{LVD}) set in LVD.

4.2 Used Pin List

Table 4.1 provides List of Pins and Functions

Table 4.1 List of Pins and Functions

Pin Name	Input/Output	Function
P137/INTP0	Input	Switch (SW1) input port
P20/AIN0	Input	A/D converter analog input port
P00	Output	LED (LED1) control port
P01	Output	LED (LED2) control port
P54	Output	LED (LED3) control port
P55	Output	LED (LED4) control port
P56	Output	LED (LED5) control port

5. Software Explanation

5.1 Operation Outline

This application enables the user to change the CPU clock and the operation mode using switch input. The CPU clock and the operating mode is changed in the order of 1 to 31 of Figure 1.1 operation mode status transition diagram.

(1) Input/output port initialization

- P00-P01 and P54-P56 pins: set as output ports (use to control LEDs)
- P137/INTP0 pin: set as input port (use for switch input)
- P20/ANI0 pin: set as analog input port (use as A/D conversion analog input channel)

(2) Clock generator initialization

<Setting conditions>

- Set the flash operation mode to HS (high-speed main) mode using user option byte (000C2H/010C2H).)
- High-speed on-chip oscillator clock frequency: set to 24 MHz
- Set the operation mode of the high-speed system clock pin to X1 oscillation, and connect a crystal resonator to the X1/P121 and X2/EXCLK/P122 pins.
- Select the main system clock (f_{MAIN}) as the CPU/peripheral hardware clock (f_{CLK}).

(3) Interrupt processing initialization

- Set the INTP0 pin valid edge to falling edge and enable switch input.
- Use the 12-bit interval timer to confirm switch input. The voltage level of the pin is checked approximately every 5 ms. If the voltage level matches twice consecutively, the switch input is recognized as valid (prevents chattering).

(4) The CPU clock and operation mode change as follows each time the falling edge of a signal (switch) input to the P137/INTP0 pin is detected. The following is the CPU clock and operation mode after the switch is pressed.

Table 5.1 LED status (after the switch is pressed) (1/2)

	CPU clock	Operation mode	LED1	LED2	LED3	LED4	LED5
(1)	High-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	OFF	ON
(2)	High-speed system clock	Normal operation mode	ON	ON	ON	OFF	ON
(3)	High-speed system clock	HALT mode	OFF	ON	ON	OFF	ON
(4)	High-speed system clock	Normal operation mode	ON	ON	ON	OFF	ON
(5)	High-speed system clock	STOP mode	OFF	OFF	ON	OFF	ON
(6)	High-speed system clock	Normal operation mode	ON	ON	ON	OFF	ON
(7)	High-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	OFF	ON
(8)	High-speed on-chip oscillator clock	HALT mode	OFF	ON	OFF	OFF	ON
(9)	High-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	OFF	ON
(10)	High-speed on-chip oscillator clock	STOP mode	OFF	OFF	OFF	OFF	ON
(11)	High-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	OFF	ON
(12)	High-speed on-chip oscillator clock	SNOOZE mode	ON	OFF	OFF	OFF	ON
(13)	High-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	OFF	ON
(14)	Middle-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	ON	OFF
(15)	High-speed system clock	Normal operation mode	ON	ON	ON	OFF	ON
(16)	Low-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	ON	ON
(17)	High-speed system clock	Normal operation mode	ON	ON	ON	OFF	ON
(18)	Middle-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	ON	OFF

Table 5.2 LED Status (after the switch is pressed) (2/2)

	CPU clock	Operation mode	LED1	LED2	LED3	LED4	LED5
(19)	Middle-speed on-chip oscillator clock	HALT mode	OFF	ON	OFF	ON	OFF
(20)	Middle-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	ON	OFF
(21)	Middle-speed on-chip oscillator clock	STOP mode	OFF	OFF	OFF	ON	OFF
(22)	Middle-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	ON	OFF
(23)	Middle-speed on-chip oscillator clock	SNOOZE mode	ON	OFF	OFF	ON	OFF
(24)	Middle-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	ON	OFF
(25)	Low-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	ON	ON
(26)	Middle-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	ON	OFF
(27)	High-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	OFF	ON
(28)	Low-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	ON	ON
(29)	Low-speed on-chip oscillator clock	HALT mode	OFF	ON	OFF	ON	ON
(30)	Low-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	ON	ON
(31)	High-speed on-chip oscillator clock	Normal operation mode	ON	ON	OFF	OFF	ON

After the CPU clock and operation mode have been changed according to steps 1 to 31 above, the falling edge of a signal (switch) input to the P137/INTP0 pin is detected, all LEDs are turned OFF, and the CPU goes to HALT mode (only RESET input in standby recovery).

In addition, if the CPU clock can't be status transition to a certain period of time such as by oscillation failure of the crystal oscillator is all LEDs are turned OFF and end the status transition in error processing.

Note: Refer to the RL78/G11 User's Manual for usage notes concerning this device.

5.2 Option Byte Settings

Table5.1 lists the option byte settings.

Table 5.1 Option Byte Settings

Address	Setting Value	Contents
000C0H/010C0H	01101110B	Watchdog timer operation is stopped (count is stopped after reset)
000C1H/010C1H	01111111B	LVD operation (V_{LVD}): reset mode Detection voltage: Rising edge 2.81 V/falling edge 2.75 V
000C2H/010C2H	11100000B	HS mode, HOCO: 24 MHz
000C3H/010C3H	10000100B	On-chip debugging enabled

5.3 Variables

Table5.2 lists the global variables.

Table 5.2 Variables

Type	Variable Name	Contents	Function Used
8-bit	g_int_cnt	Number of interval signal detection interrupts for12-bit interval timer	R_MAIN_BtoD, R_MAIN_DtoB, R_MAIN_BtoC, R_MAIN_CtoD, R_MAIN_DtoE, R_MAIN_EtoD, R_MAIN_DtoC, R_MAIN_CtoE, R_MAIN_EtoC, R_MAIN_CtoB, R_MAIN_BtoE, R_MAIN_EtoB, r_it_interrupt
8-bit	g_int_flg	Confirm the external interrupt generation detection flag	R_MAIN_NOP_Loop, r_intc0_interrupt

5.4 Functions (subroutines)

Table 5.3 lists the functions (subroutines).

Table 5.3 Functions

Function Name	Outline	Number of operating mode status transition diagram
R_MAIN_AtoB	Status transition processing from (A) to (B)	(1)
R_MAIN_BtoD	Status transition processing from (B) to (D)	(2)
R_MAIN_DtoL	Status transition processing from (D) to (L)	(3)
R_MAIN_LtoD	Status transition processing from (L) to (D)	(4)
R_MAIN_DtoM	Status transition processing from (D) to (M)	(5)
R_MAIN_MtoC	Status transition processing from (M) to (D)	(6)
R_MAIN_DtoB	Status transition processing from (D) to (B)	(7)
R_MAIN_BtoF	Status transition processing from (B) to (F)	(8)
R_MAIN_FtoB	Status transition processing from (F) to (B)	(9)
R_MAIN_BtoG	Status transition processing from (B) to (G)	(10)
R_MAIN_GtoB	Status transition processing from (G) to (B)	(11)
R_MAIN_BtoH	Status transition processing from (B) to (H)	(12)
R_MAIN_HtoB	Status transition processing from (H) to (B)	(13)
R_MAIN_BtoC	Status transition processing from (B) to (C)	(14)
R_MAIN_CtoD	Status transition processing from (C) to (D)	(15)
R_MAIN_DtoE	Status transition processing from (D) to (E)	(16)
R_MAIN_EtoD	Status transition processing from (E) to (D)	(17)
R_MAIN_DtoC	Status transition processing from (D) to (C)	(18)
R_MAIN_CtoI	Status transition processing from (C) to (I)	(19)
R_MAIN_ItoC	Status transition processing from (I) to (C)	(20)
R_MAIN_CtoJ	Status transition processing from (C) to (J)	(21)
R_MAIN_JtoC	Status transition processing from (J) to (C)	(22)
R_MAIN_CtoK	Status transition processing from (C) to (K)	(23)
R_MAIN_KtoC	Status transition processing from (K) to (C)	(24)
R_MAIN_CtoE	Status transition processing from (C) to (E)	(25)
R_MAIN_EtoC	Status transition processing from (E) to (C)	(26)
R_MAIN_CtoB	Status transition processing from (C) to (B)	(27)
R_MAIN_BtoE	Status transition processing from (B) to (E)	(28)
R_MAIN_EtoN	Status transition processing from (E) to (N)	(29)
R_MAIN_NtoE	Status transition processing from (N) to (E)	(30)
R_MAIN_EtoB	Status transition processing from (E) to (B)	(31)
R_MAIN_NOP_Loop	Continuous NOP instruction execution processing	—
R_MAIN_END	A/D converter setting	—
R_MAIN_ERROR	End processing of status transition	—
R_MAIN_Set_SnoozeOn	Error processing of status transition	—
r_intc0_interrupt	Confirm the external interrupt generation detection flag update processing	—
r_it_interrupt	12-bit interval timer interval signal detection interrupt count processing	—
r_adc_interrupt	SNOOZE mode release processing	—

5.5 Function (subroutine) Specifications

The following are the sample code functions (subroutines) used in this application note.

[Function Name] R_MAIN_AtoB

Outline	Status transition processing from (A) to (B)
Declaration	void R_MAIN_AtoB(void)
Description	Control LED lighting. (CPU clock: high-speed on-chip oscillator clock)
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_BtoD

Outline	Status transition processing from (B) to (D)
Declaration	void R_MAIN_BtoD(void)
Description	Change the CPU clock from high-speed on-chip oscillator clock to high-speed system clock. After the clock is switched, control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_DtoL

Outline	Status transition processing from (D) to (L)
Declaration	void R_MAIN_DtoL(void)
Description	Control LED lighting, then transition to HALT mode. (CPU clock stopped (when using high-speed system clock))
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_LtoD

Outline	Status transition processing from (L) to (D)
Declaration	void R_MAIN_LtoD(void)
Description	Control LED lighting. (CPU clock: high-speed system clock)
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_DtoM

Outline	Status transition processing from (D) to (M)
Declaration	void R_MAIN_DtoM(void)
Description	Control LED lighting, then transition to STOP mode. (Stop CPU clock (when using high-speed system clock))
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_MtoD

Outline	Status transition processing from (M) to (D)
Declaration	void R_MAIN_MtoD(void)
Description	Control LED lighting. (CPU clock: high-speed system clock)
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_DtoB

Outline	Status transition processing from (D) to (B)
Declaration	void R_MAIN_DtoB(void)
Description	Change the CPU clock from high-speed system clock to high-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_BtoF

Outline	Status transition processing from (B) to (F)
Declaration	void R_MAIN_BtoF(void)
Description	Control LED lighting, then transition to HALT mode. (CPU clock stopped (when using high-speed on-chip oscillator clock))
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_FtoB

Outline	Status transition processing from (F) to (B)
Declaration	void R_MAIN_FtoB(void)
Description	Control LED lighting. (CPU clock: high-speed on-chip oscillator clock)
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_BtoG

Outline	Status transition processing from (B) to (G)
Declaration	void R_MAIN_BtoG(void)
Description	Control LED lighting, then transition to STOP mode. (CPU clock stopped (when using high-speed on-chip oscillator clock))
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_GtoB

Outline	Status transition processing from (G) to (B)
Declaration	void R_MAIN_GtoB(void)
Description	Control LED lighting. (CPU clock: high-speed on-chip oscillator clock)
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_BtoH

Outline	Status transition processing from (B) to (H)
Declaration	void R_MAIN_BtoH(void)
Description	Set A/D converter and control LED lighting. Then, transition to SNOOZE mode.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_HtoB

Outline	Status transition processing from (H) to (B)
Declaration	void R_MAIN_HtoB(void)
Description	Set SNOOZE release and stop A/D converter. Then control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_BtoC

Outline	Status transition processing from (B) to (C)
Declaration	void R_MAIN_BtoC(void)
Description	Change the CPU clock from high-speed on-chip oscillator clock to middle-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_CtoD

Outline	Status transition processing from (C) to (D)
Declaration	void R_MAIN_CtoD(void)
Description	Change the CPU clock from middle-speed on-chip oscillator clock to high-speed system clock. After the clock is switched, control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_DtoE

Outline	Status transition processing from (D) to (E)
Declaration	void R_MAIN_DtoE(void)
Description	Change the CPU clock from high-speed system clock to low-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_EtoD

Outline	Status transition processing from (E) to (D)
Declaration	void R_MAIN_EtoD(void)
Description	Change the CPU clock from low-speed on-chip oscillator clock to high-speed system clock. After the clock is switched, control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_DtoC

Outline	Status transition processing from (D) to (C)
Declaration	void R_MAIN_DtoC(void)
Description	Change the CPU clock from high-speed system clock to middle-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_CtoI

Outline	Status transition processing from (C) to (I)
Declaration	void R_MAIN_CtoI(void)
Description	Control LED lighting, then transition to HALT mode. (CPU clock stopped (when using middle-speed on-chip oscillator clock))
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_ItoC

Outline	Status transition processing from (I) to (C)
Declaration	void R_MAIN_ItoC(void)
Description	Control LED lighting. (CPU clock: middle-speed on-chip oscillator clock)
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_CtoJ

Outline	Status transition processing from (C) to (J)
Declaration	void R_MAIN_CtoJ(void)
Description	Control LED lighting, then transition to STOP mode. (CPU clock stopped (when using middle-speed on-chip oscillator clock))
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_JtoC

Outline	Status transition processing from (J) to (C)
Declaration	void R_MAIN_JtoC(void)
Description	Control LED lighting. (CPU clock: middle-speed on-chip oscillator clock)
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_CtoK

Outline	Status transition processing from (C) to (K)
Declaration	void R_MAIN_CtoK(void)
Description	Set A/D converter and control LED lighting. Then, transition to SNOOZE mode.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_KtoC

Outline	Status transition processing from (K) to (C)
Declaration	void R_MAIN_KtoC(void)
Description	Set SNOOZE release and stop A/D converter. Then control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_CtoE

Outline	Status transition processing from (C) to (E)
Declaration	void R_MAIN_CtoE(void)
Description	Change the CPU clock from middle-speed on-chip oscillator clock to low-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_EtoC

Outline	Status transition processing from (E) to (C)
Declaration	void R_MAIN_EtoC(void)
Description	Change the CPU clock from low-speed on-chip oscillator clock to middle-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_CtoB

Outline	Status transition processing from (C) to (B)
Declaration	void R_MAIN_CtoB(void)
Description	Change the CPU clock from middle-speed on-chip oscillator clock to high-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_BtoE

Outline	Status transition processing from (B) to (E)
Declaration	void R_MAIN_BtoE(void)
Description	Change the CPU clock from high-speed on-chip oscillator clock to low-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_EtoN

Outline	Status transition processing from (E) to (N)
Declaration	void R_MAIN_EtoN(void)
Description	Control LED lighting, then transition to HALT mode. (CPU clock stopped (when using low-speed on-chip oscillator clock))
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_NtoE

Outline	Status transition processing from (N) to (E)
Declaration	void R_MAIN_NtoE(void)
Description	Control LED lighting. (CPU clock: low-speed on-chip oscillator clock)
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_EtoB

Outline	Status transition processing from (E) to (B)
Declaration	void R_MAIN_EtoB(void)
Description	Change the CPU clock from low-speed on-chip oscillator clock to high-speed on-chip oscillator clock. After the clock is switched, control LED lighting.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_NOP_Loop

Outline	Continuous NOP instruction execution processing
Declaration	void R_MAIN_NOP_Loop(void)
Description	Execute NOP instruction continuously. End processing when external interrupt generation detection flag is confirmed.
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_END

Outline	End processing of status transition
Declaration	void R_MAIN_END(void)
Description	Disable interrupts. Control LED lighting (all off).
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_ERROR

Outline	Error processing of status transition
Declaration	void R_MAIN_ERROR(void)
Description	Disable interrupts. Control LED lighting (all off). Loop processing with in the function(Return is only reset input).
Argument	None
Return Value	None
Notes	None

[Function Name] R_MAIN_AD_SnoozeOn

Outline	A/D converter setting
Declaration	void R_MAIN_AD_SnoozeOn(void)
Description	Set A/D converter to hardware trigger wait mode with 12-bit interval timer interrupt signal. Enable SNOOZE mode and transition to A/D conversion wait status.
Argument	None
Return Value	None
Notes	None

[Function Name] r_intc0_interrupt

Outline	External interrupt generation detection flag confirmation processing
Declaration	static void __near r_intc0_interrupt(void)
Description	Confirm external interrupt generation detection flag with generation of external interrupt. End processing when switch input level changes to high.
Argument	None
Return Value	None
Notes	None

[Function Name] r_it_interrupt

Outline	12-bit interval timer interval signal detection interrupt count processing
Declaration	static void __near r_it_interrupt(void)
Description	Increment the RITCOUNT each time the 12-bit interval timer interrupt signal detection interrupt is generated.
Argument	None
Return Value	None
Notes	None

[Function Name] r_adc_interrupt

Outline	SNOOZE mode release processing
Declaration	static void __near r_adc_interrupt(void)
Description	Clear the AWC bit of the ADM2 register and release the SNOOZE mode.
Argument	None
Return Value	None
Notes	None

5.6 Flowcharts

Figure 5.1 shows the entire flow for this application note.

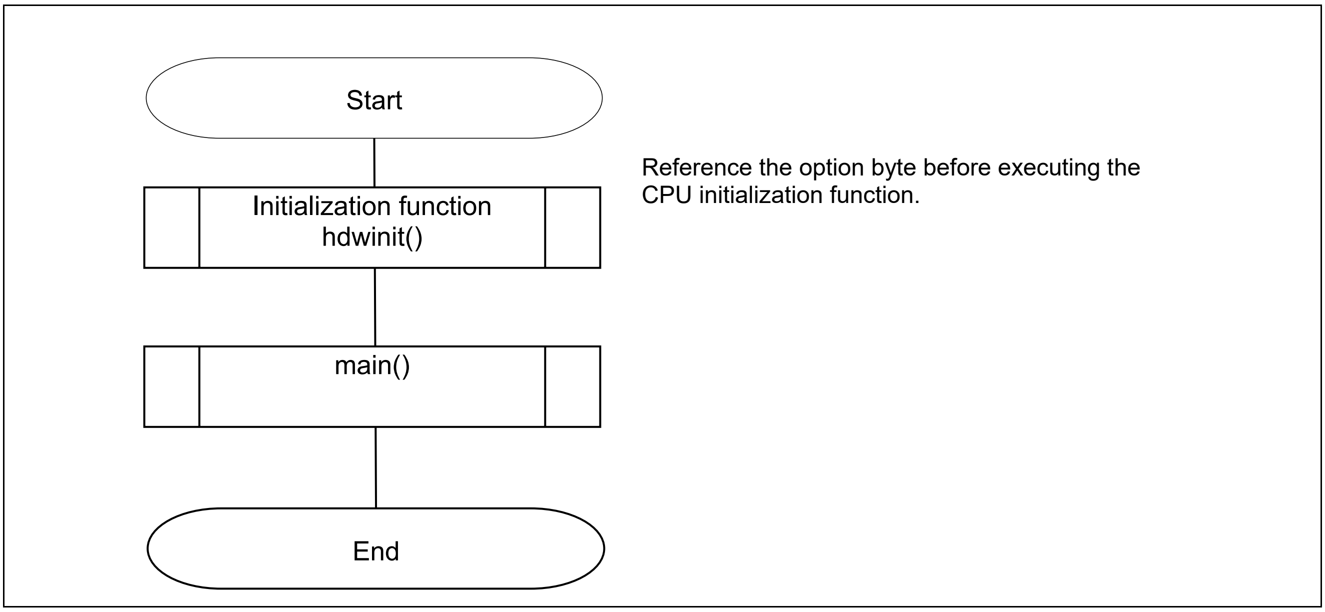

Start
Initialization function
hdwinit()
main()
End
Reference the option byte before executing the CPU initialization function.

Figure 5.1 Overall Flowchart

5.6.1 Initialization Function

Figure 5.2 shows the flowchart for the initialization function.

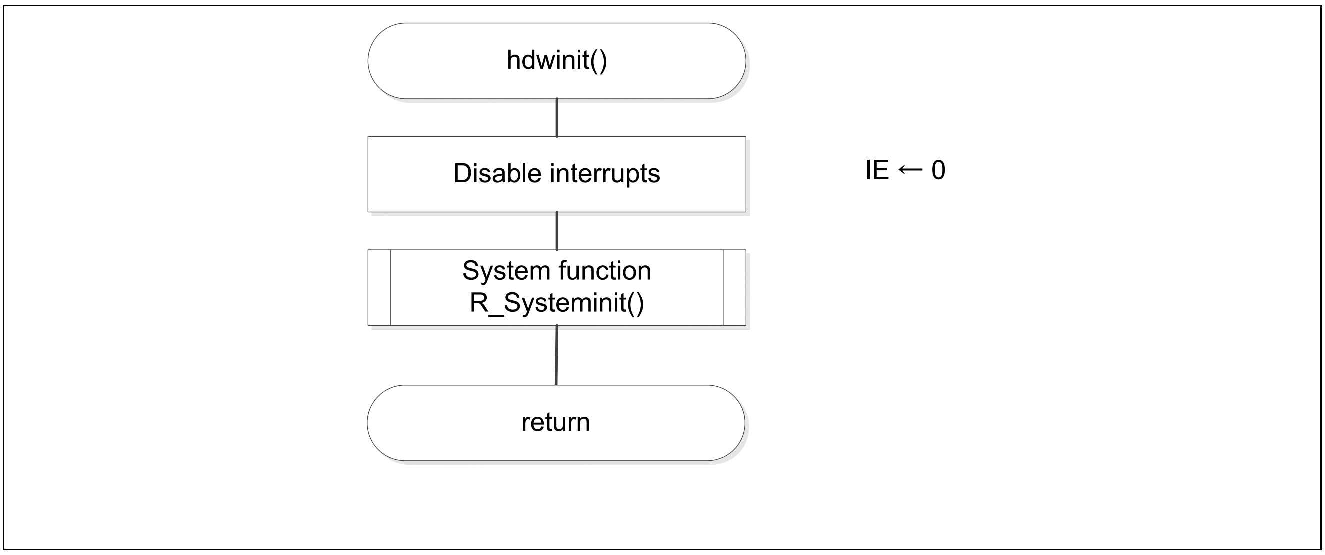


Figure 5.2 Initialization Function

5.6.2 System Function

Figure 5.3 shows the flowchart for system function.

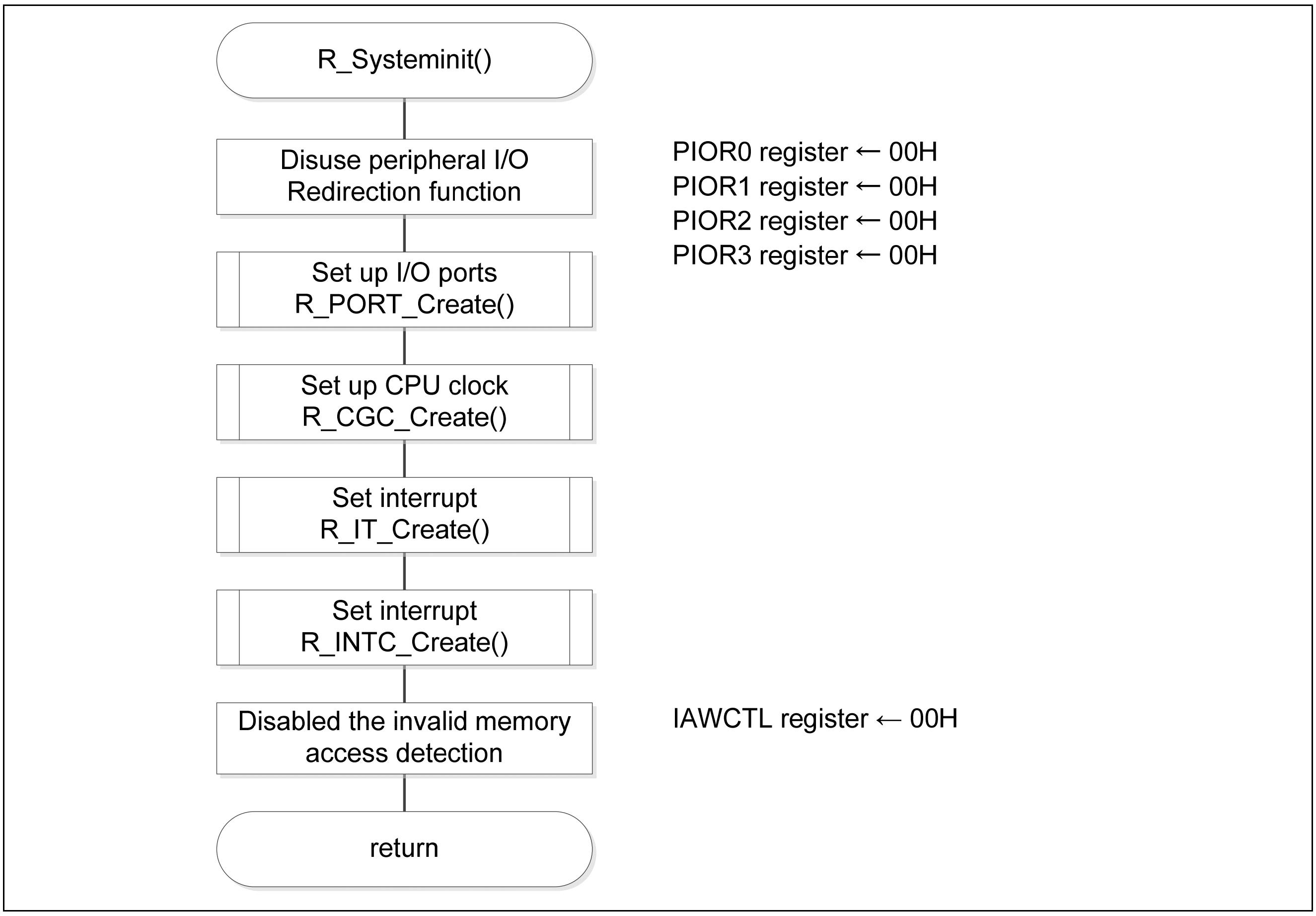
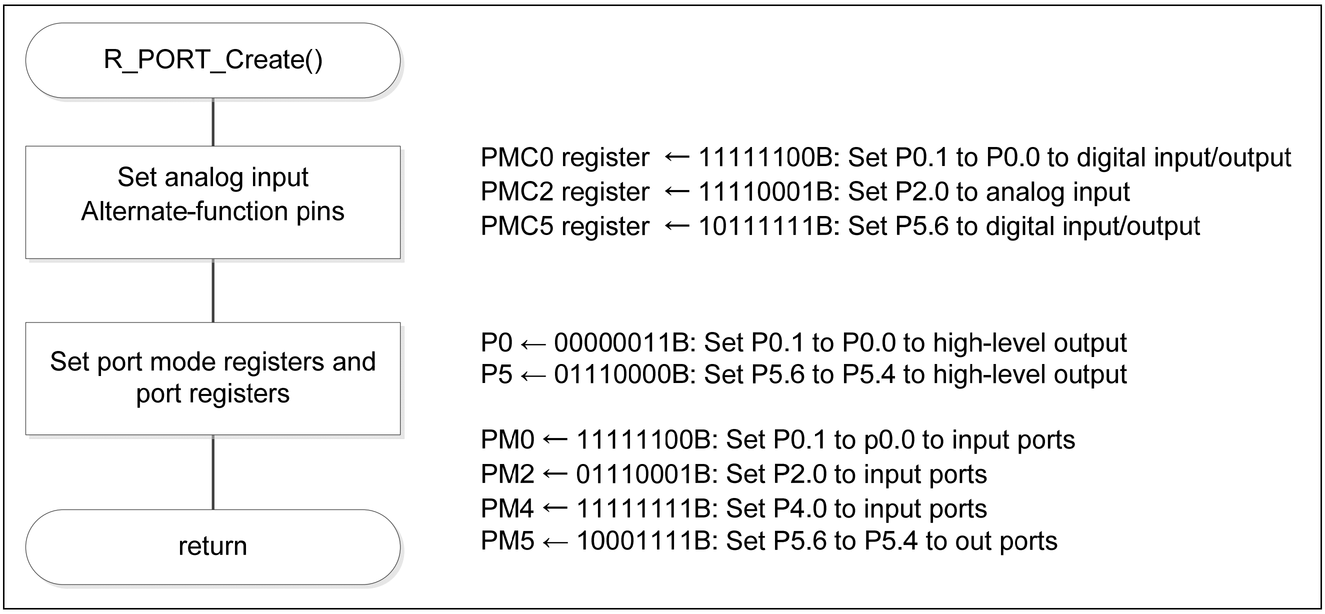

R_Systeminit()
Disuse peripheral I/O Redirection function
Set up I/O ports R_PORT_Create()
Set up CPU clock R_CGC_Create()
Set interrupt R_IT_Create()
Set interrupt R_INTC_Create()
Disabled the invalid memory access detection
return
PIOR0 register ← 00H
PIOR1 register ← 00H
PIOR2 register ← 00H
PIOR3 register ← 00H
IAWCTL register ← 00H

Figure 5.3 System Function

5.6.3 Input/Output Port Settings

Figure 5.4 shows the flowchart for the input/output port settings.



PMC0 register ← 11111100B: Set P0.1 to P0.0 to digital input/output
PMC2 register ← 11110001B: Set P2.0 to analog input
PMC5 register ← 10111111B: Set P5.6 to digital input/output

P0 ← 00000011B: Set P0.1 to P0.0 to high-level output
P5 ← 01110000B: Set P5.6 to P5.4 to high-level output

PM0 ← 11111100B: Set P0.1 to p0.0 to input ports
PM2 ← 01110001B: Set P2.0 to input ports
PM4 ← 11111111B: Set P4.0 to input ports
PM5 ← 10001111B: Set P5.6 to P5.4 to out ports

Figure 5.4 Input/Output Port Settings

Note: Refer to the initialization flowchart in the RL78/G13 Initialization (R01AN2575J) Application Note for details on how to set unused ports.

Caution: When designing circuits, always make sure unused ports are properly processed and all electrical characteristics are met. Also make sure each unused input-only port is connected to V_{DD} or V_{SS} through a resister.

5.6.4 Clock Generator Setting

Figure 5.5 shows the flowchart for setting the clock generator.

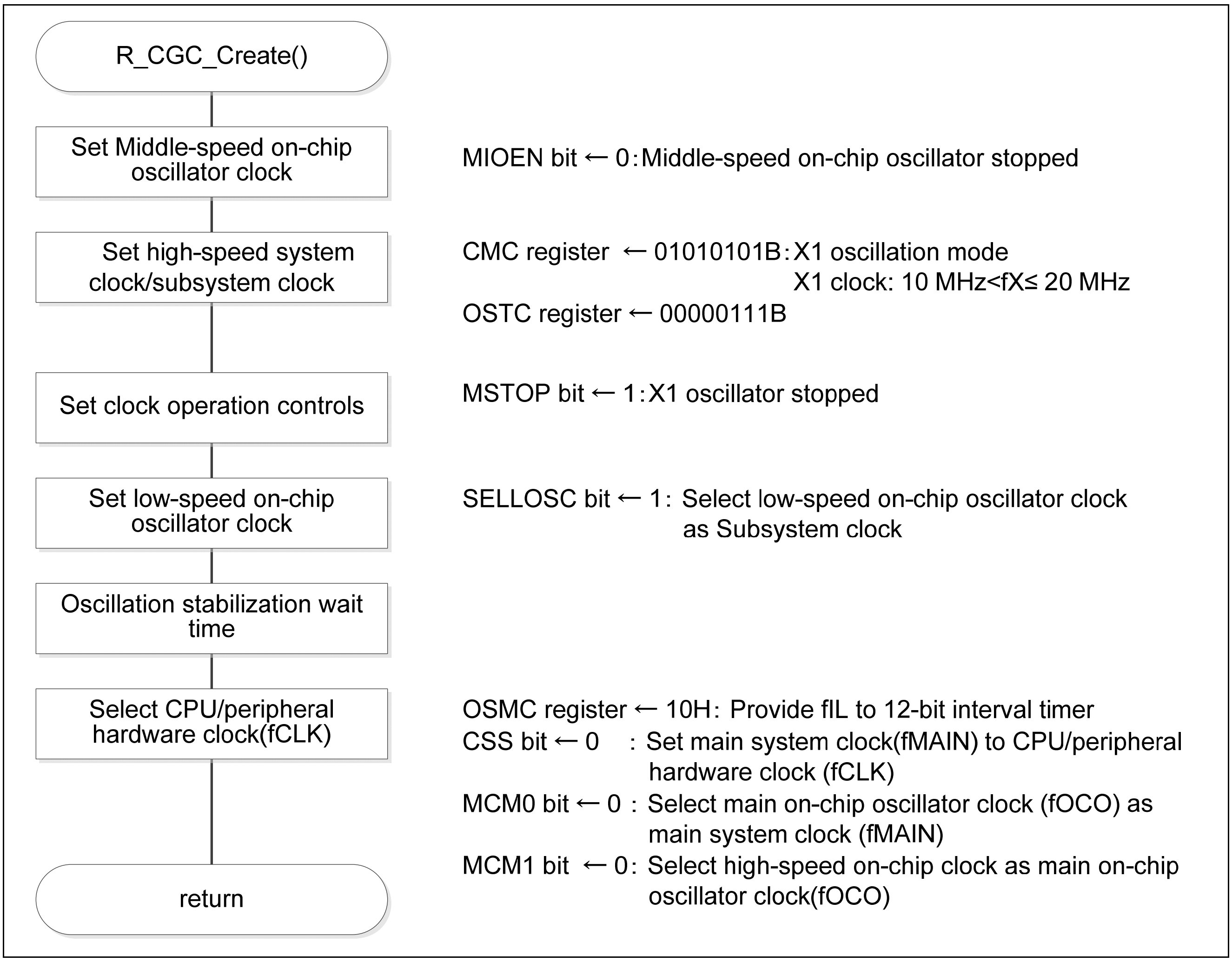

R_CGC_Create()
Set Middle-speed on-chip oscillator clock
MIOEN bit ← 0:Middle-speed on-chip oscillator stopped
Set high-speed system clock/subsystem clock
CMC register ← 01010101B:X1 oscillation mode
X1 clock: 10 MHz<fX≤ 20 MHz
OSTC register ← 00000111B
Set clock operation controls
MSTOP bit ← 1:X1 oscillator stopped
Set low-speed on-chip oscillator clock
SELLOSC bit ← 1: Select low-speed on-chip oscillator clock as Subsystem clock
Oscillation stabilization wait time
Select CPU/peripheral hardware clock(fCLK)
OSMC register ← 10H: Provide fIL to 12-bit interval timer
CSS bit ← 0 : Set main system clock(fMAIN) to CPU/peripheral hardware clock (fCLK)
MCM0 bit ← 0 : Select main on-chip oscillator clock (fOCO) as main system clock (fMAIN)
MCM1 bit ← 0: Select high-speed on-chip clock as main on-chip oscillator clock(fOCO)
return

Figure 5.5 Clock Generator Setting

Clock operation mode setting

- Clock operation mode control register (CMC)
  High-speed system clock pin operation mode: X1 oscillation mode
  X1 clock oscillation frequency control: 10MHz < fMX ≤ 20MHz

Symbol: CMC

7	6	5	4	3	2	1	0
EXCLK	OSCSEL	0	0	0	0	0	AMPH
0	**1**	0	0	0	0	0	**1**

Bits 7-6

EXCLK	OSCSEL	High-seed oscillation clock pin operation mode	X1/P121 Port	X2/EXCLK/P122 Port
0	0	Input port mode	Input port	
0	**1**	**X1 oscillation mode**	**Crystal/ceramic resonator connection**	
1	0	Input port mode	Input port	
1	1	External clock input mode	Input port	External clock input

Bit 0

AMPH	Control of X1 clock oscillation frequency
0	1MHz ≤ f_X ≤ 10MHz
1	**10MHz < f_X ≤ 20MHz**

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.

Operation control of clocks

- Clock operation status control register (CSC)
  High-speed system clock operation control: X1 oscillator stopped
  Middle-speed on-chip oscillator clock operation control: Middle-speed on-chip oscillator stopped
  High-speed on-chip oscillator clock operation control: High-speed on-chip oscillator operating

Symbol: CSC

7	6	5	4	3	2	1	0
MSTOP	1	0	0	0	0	MIOEN	HIOSTOP
1	1	0	0	0	0	0	**0**

Bit 7

MSTOP	High-speed system clock operation control		
	X1 oscillation mode	External clock input mode	Input port mode
0	X1 oscillator operating	External clock from EXCLK pin is valid	Input port
1	**X1 oscillator stopped**	**External clock from EXCLK pin is invalid**	

Bit 1

MIOEN	Middle-speed on-chip oscillator clock operation control
0	**Middle-speed on-chip oscillator stopped**
1	Middle-speed on-chip oscillator operating

Bit 0

HIOSTOP	High-speed on-chip oscillator clock operation control
0	**High-speed on-chip oscillator operating**
1	High-speed on-chip oscillator stopped

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.

CPU/peripheral hardware clock (f_{CLK}) setting

- System clock control register (CKC)
  Status of f_{CLK}: main system clock
  Selection of f_{CLK}: high-speed on-chip oscillator clock （f_{IH}）

Symbol: CKC

7	6	5	4	3	2	1	0
CLS	CSS	MCS	MCM0	0	0	MCS1	MCM1
0	**0**	**0**	**0**	0	0	0	0

Bit 7

CLS	Status of CPU/peripheral hardware clock (f_{CLK})
0	**Main system clock (f_{MAIN})**
1	Subsystem clock (f_{SUB})

Bit 6

CSS	Selection of CPU/peripheral hardware clock (f_{CLK})
0	**Main system clock (f_{MAIN})**
1	Subsystem clock (f_{SUB})

Bit 5

MCS	Status of main system clock (f_{MAIN})
0	**High-speed on-chip oscillator clock f_{IH})**
1	High-speed system clock (f_{MX})

Bit 4

MCM0	Main system clock (f_{MAIN}) operation control
0	**Selects high-speed on-chip oscillator clock (f_{IH}) as main system clock (f_{MAIN})**
1	Selects high-speed system clock (f_{MX}) as main system clock (f_{MAIN}).

Bit 1

MCS1	Status of main on-chip oscillator clock (f_{oco})
0	**High-speed on-chip oscillator clock (f_{IH})**
1	Middle-speed on-chip oscillator clock (f_{IM})

Bit 0

MCS1	Main on-chip oscillator clock (f_{oco}) operation control
0	**High-speed on-chip oscillator clock (f_{IH})**
1	Middle-speed on-chip oscillator clock (f_{IM})

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.

Operation speed mode control

- Operation speed mode control register (OSMC)
  Selection of operation clock for 12-bit interval timer: low-speed on-chip oscillator clock

Symbol: OSMC

7	6	5	4	3	2	1	0
0	0	0	WUTMMCK0	0	0	0	0
0	0	0	**1**	0	0	0	0

Bit 4

WUTMMCK0	Selection of operation clock for 12-bit interval timer
0	Do not select low-speed on-chip oscillator clock
1	**Select low-speed on-chip oscillator clock**

Subsystem clock select

- Subsystem clock select register(CKSEL)
  Subsystem clock select :Select low-speed on-chip oscillator clock

Symbol: CKSEL

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	SELLOSC
0	0	0	0	0	0	0	**1**

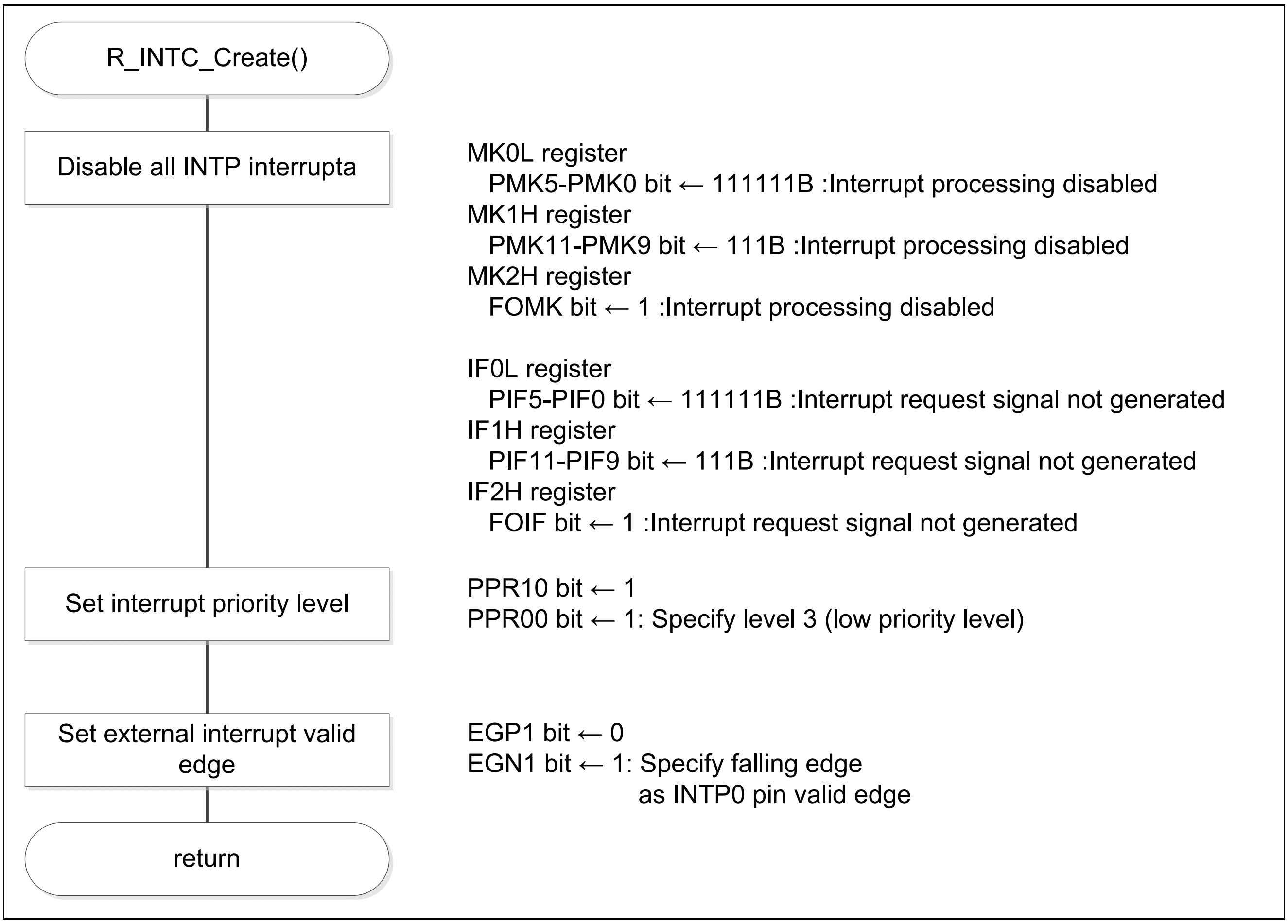
Bit 0

SELLOSC	Selection of low-speed on-chip oscillator clock
0	Do not select low-speed on-chip oscillator clock
1	**Select low-speed on-chip oscillator clock**

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.

5.6.5 External Interrupt Setting

Figure 5.6 shows the flowchart for setting the external interrupt.


R_INTC_Create()
Disable all INTP interrupta
Set interrupt priority level
Set external interrupt valid edge
return

MK0L register
  PMK5-PMK0 bit ← 111111B :Interrupt processing disabled
MK1H register
  PMK11-PMK9 bit ← 111B :Interrupt processing disabled
MK2H register
  FOMK bit ← 1 :Interrupt processing disabled

IF0L register
  PIF5-PIF0 bit ← 111111B :Interrupt request signal not generated
IF1H register
  PIF11-PIF9 bit ← 111B :Interrupt request signal not generated
IF2H register
  FOIF bit ← 1 :Interrupt request signal not generated

PPR10 bit ← 1
PPR00 bit ← 1: Specify level 3 (low priority level)

EGP1 bit ← 0
EGN1 bit ← 1: Specify falling edge
as INTP0 pin valid edge

Figure 5.6 External Interrupt Setting

Control of external interrupt valid edge

- External interrupt rising edge enable register (EGP0)
  Select valid edge for INTP0 pin: falling edge

Symbol: EGP0

7	6	5	4	3	2	1	0
EGP7	EGP6	EGP5	EGP4	EGP3	EGP2	EGP1	EGP0
x	x	x	x	x	x	x	**0**

Symbol: EGN0

7	6	5	4	3	2	1	0
EGN7	EGN6	EGN5	EGN4	EGN3	EGN2	EGN1	EGN0
x	x	x	x	x	x	x	**1**

Bit 0

EGP1	EGN1	INTP0 pin valid edge selection
0	0	Edge detection disabled
0	**1**	**Falling edge**
1	0	Rising edge
1	1	Both rising and falling edges

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.

5.6.6 12-bit Interval Timer Setting

Figure 5.7 shows the flowchart for setting the 12-bit interval timer.

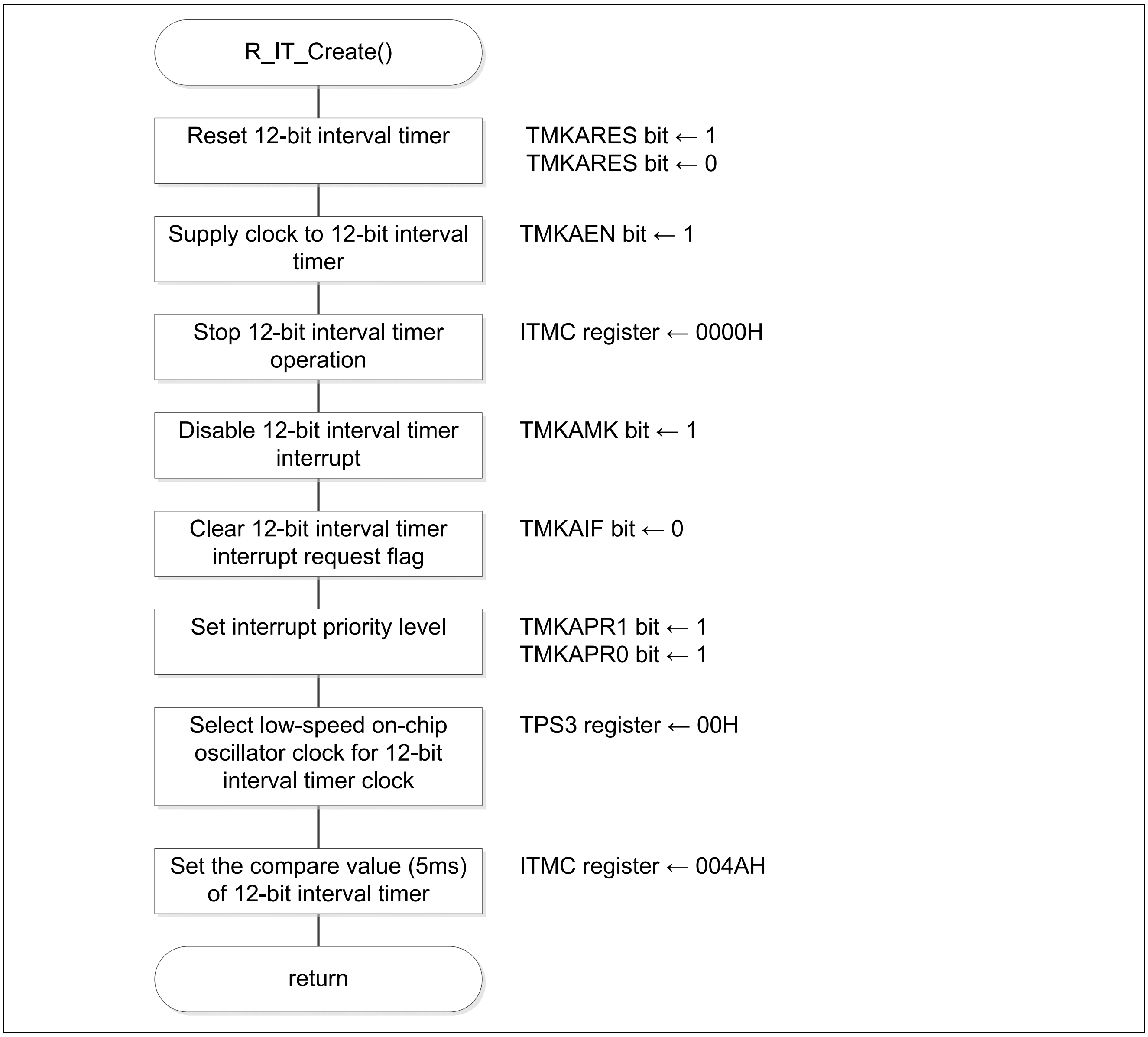

R_IT_Create()
Reset 12-bit interval timer
TMKARES bit ← 1
TMKARES bit ← 0
Supply clock to 12-bit interval timer
TMKAEN bit ← 1
Stop 12-bit interval timer operation
ITMC register ← 0000H
Disable 12-bit interval timer interrupt
TMKAMK bit ← 1
Clear 12-bit interval timer interrupt request flag
TMKAIF bit ← 0
Set interrupt priority level
TMKAPR1 bit ← 1
TMKAPR0 bit ← 1
Select low-speed on-chip oscillator clock for 12-bit interval timer clock
TPS3 register ← 00H
Set the compare value (5ms) of 12-bit interval timer
ITMC register ← 004AH
return

Figure 5.7 12-bit Interval Timer Setting

12-bit interval timer clock supply setting

- Peripheral enable register 2 (PER2)
  Enable clock supply to 12-bit interval timer.

Symbol: PER2

7	6	5	4	3	2	1	0
TMKAEN	0	DOCEN	0	0	0	0	TKB0EN
1	0	x	0	0	0	0	x

Bit 7

TMKAEN	Control of 12-bit interval timer input clock supply
0	Stops input clock supply.
1	**Enables input clock supply.**

12-bit interval timer interval signal detection interrupt (INTIT) setting

- Interrupt request flag register (IF1H)
  Clear TMKAIF interrupt source flag.
- Interrupt mask flag register (MK1H)
  Set TMKAMK interrupt mask.

Symbol: IF1H

7	6	5	4	3	2	1	0
PIF11	PIF10	PIF9	PIF8	PIF7	KRIF	TMKAIF	ADIF
x	x	x	x	x	x	**0**	x

Bit 1

TMKAIF	Interrupt request flag
0	**No interrupt request signal is generated**
1	Interrupt request signal is generated, interrupt request status

Symbol: MK1H

7	6	5	4	3	2	1	0
PMK11	PMK10	PMK9	PMK8	PMK7	KRMK	TMKAMK	ADMK
x	x	x	x	x	x	**0**	x

Bit 1

TMKAMK	Interrupt servicing control
0	Interrupt servicing enabled
1	**Interrupt servicing disabled**

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.

12-bit interval timer interval signal detection interrupt (INTIT) setting

- Interval timer control register (ITMC)
  Start 12-bt interval timer count operation.

Symbol: ITMC

15	14	13	2	11-0
RINTE	0	0	0	ITCMP11-ITCMP0
1	0	0	0	**04AH**

Bit 15

RINTE	12-bit interval timer operation control
0	Count operation stopped (count clear)
1	**Count operation started**

Bits 11-0

ITCMP11-ITCMP0	Specification of 12-bit interval timer compare value
04AH	**These bits generate an interrupt at the fixed cycle (count clock cycles x (ITCMP setting value 04AH + 1)).**
000H	Setting prohibited

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.

5.6.7 Main Processing

Figure 5.8, Figure 5.9, Figure 5.10, Figure 5.11 shows the flowchart for the main processing.

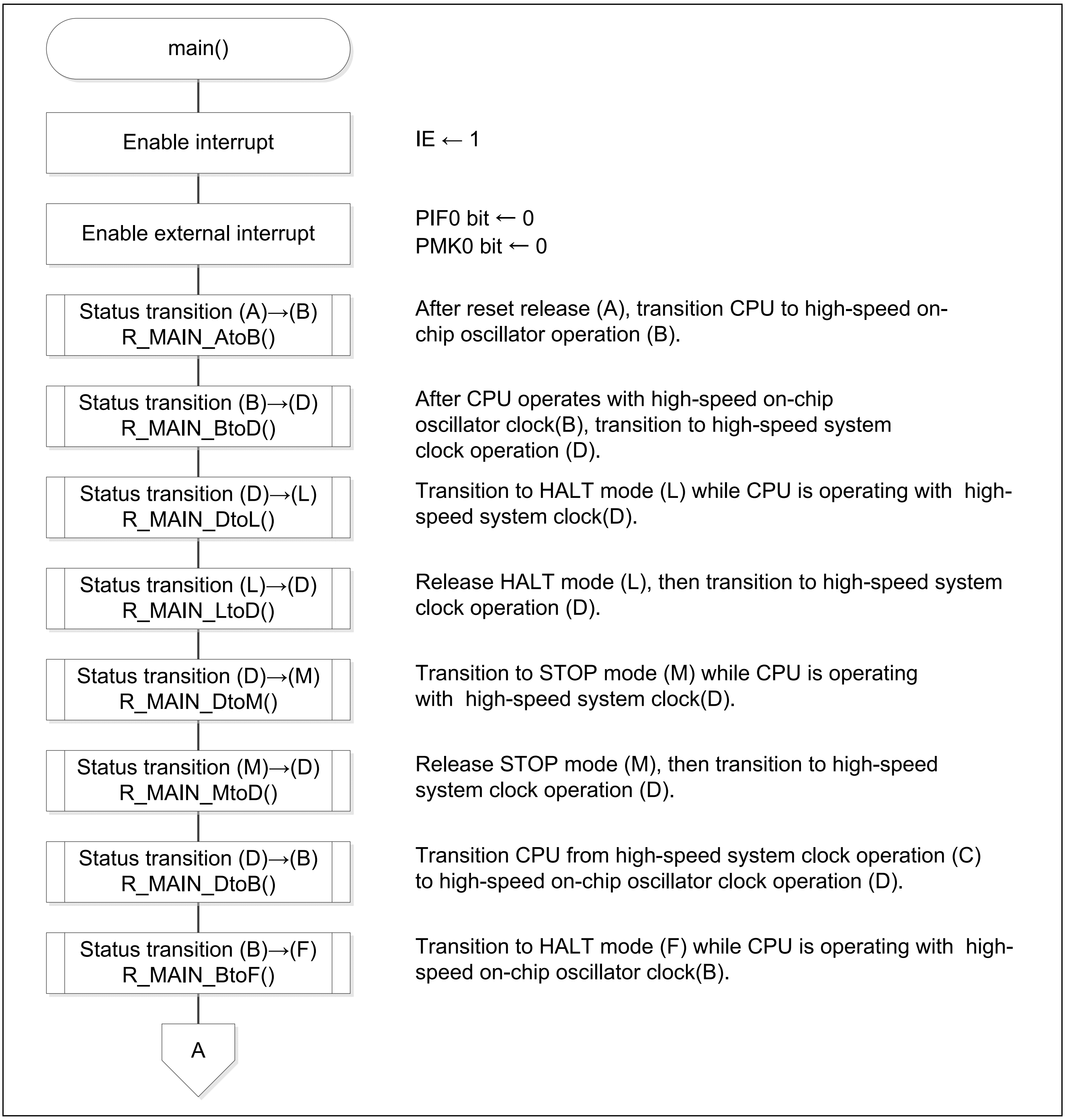

main()
Enable interrupt
IE ← 1
Enable external interrupt
PIF0 bit ← 0
PMK0 bit ← 0
Status transition (A)→(B)
R_MAIN_AtoB()
After reset release (A), transition CPU to high-speed on-chip oscillator operation (B).
Status transition (B)→(D)
R_MAIN_BtoD()
After CPU operates with high-speed on-chip oscillator clock(B), transition to high-speed system clock operation (D).
Status transition (D)→(L)
R_MAIN_DtoL()
Transition to HALT mode (L) while CPU is operating with high-speed system clock(D).
Status transition (L)→(D)
R_MAIN_LtoD()
Release HALT mode (L), then transition to high-speed system clock operation (D).
Status transition (D)→(M)
R_MAIN_DtoM()
Transition to STOP mode (M) while CPU is operating with high-speed system clock(D).
Status transition (M)→(D)
R_MAIN_MtoD()
Release STOP mode (M), then transition to high-speed system clock operation (D).
Status transition (D)→(B)
R_MAIN_DtoB()
Transition CPU from high-speed system clock operation (C) to high-speed on-chip oscillator clock operation (D).
Status transition (B)→(F)
R_MAIN_BtoF()
Transition to HALT mode (F) while CPU is operating with high-speed on-chip oscillator clock(B).
A

Figure 5.8 Main Processing (1/4)

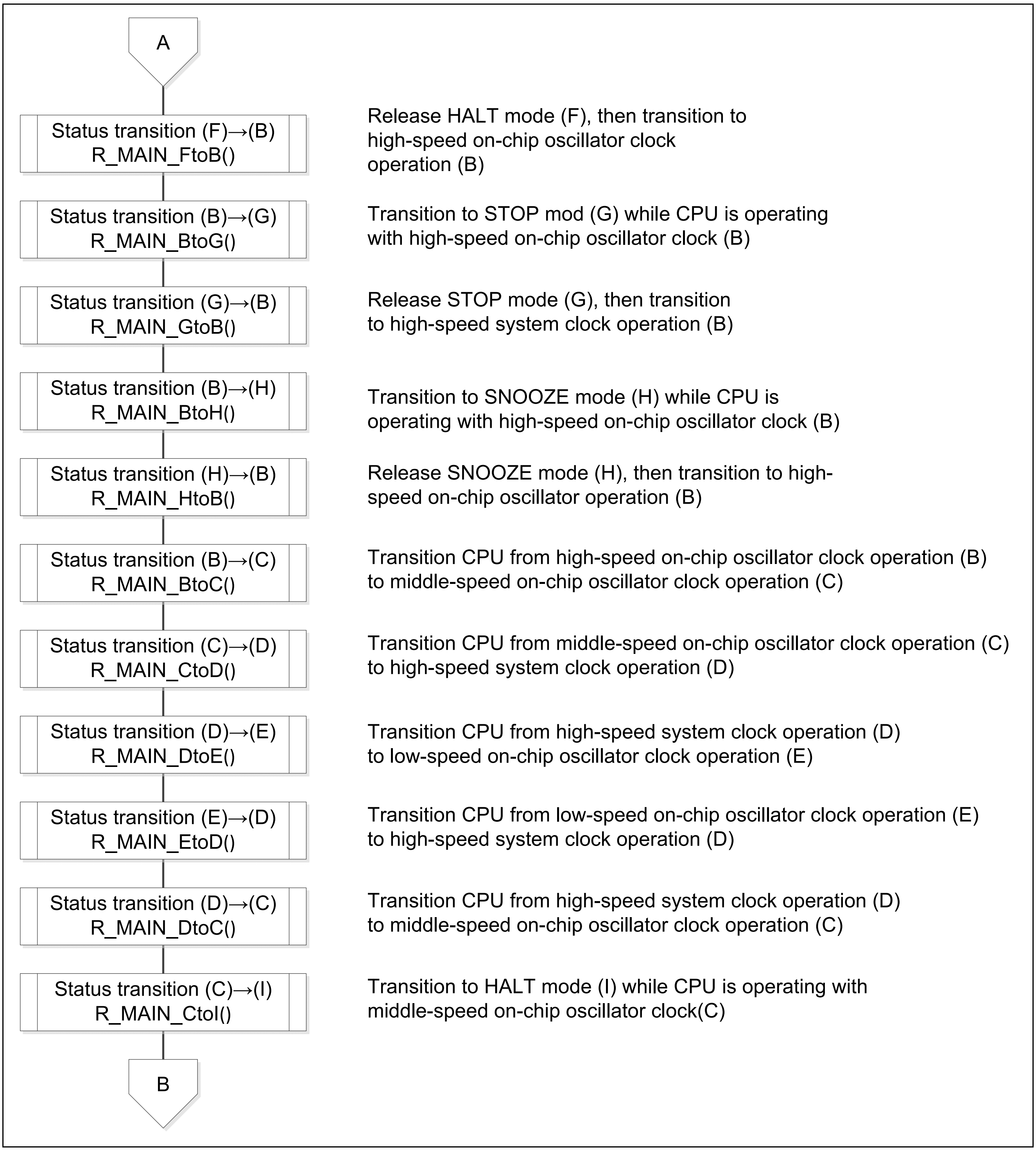
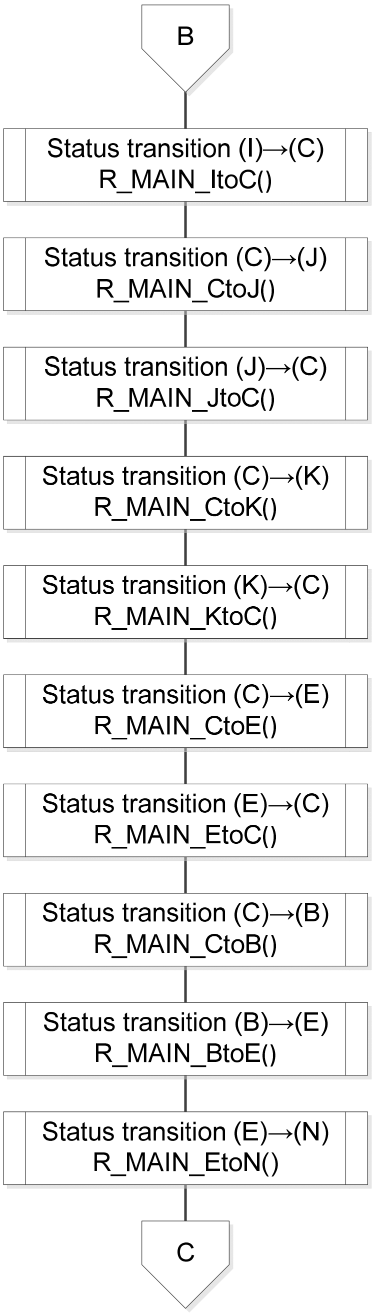

A
Status transition (F)→(B)
R_MAIN_FtoB()
Release HALT mode (F), then transition to high-speed on-chip oscillator clock operation (B)
Status transition (B)→(G)
R_MAIN_BtoG()
Transition to STOP mod (G) while CPU is operating with high-speed on-chip oscillator clock (B)
Status transition (G)→(B)
R_MAIN_GtoB()
Release STOP mode (G), then transition to high-speed system clock operation (B)
Status transition (B)→(H)
R_MAIN_BtoH()
Transition to SNOOZE mode (H) while CPU is operating with high-speed on-chip oscillator clock (B)
Status transition (H)→(B)
R_MAIN_HtoB()
Release SNOOZE mode (H), then transition to high-speed on-chip oscillator operation (B)
Status transition (B)→(C)
R_MAIN_BtoC()
Transition CPU from high-speed on-chip oscillator clock operation (B) to middle-speed on-chip oscillator clock operation (C)
Status transition (C)→(D)
R_MAIN_CtoD()
Transition CPU from middle-speed on-chip oscillator clock operation (C) to high-speed system clock operation (D)
Status transition (D)→(E)
R_MAIN_DtoE()
Transition CPU from high-speed system clock operation (D) to low-speed on-chip oscillator clock operation (E)
Status transition (E)→(D)
R_MAIN_EtoD()
Transition CPU from low-speed on-chip oscillator clock operation (E) to high-speed system clock operation (D)
Status transition (D)→(C)
R_MAIN_DtoC()
Transition CPU from high-speed system clock operation (D) to middle-speed on-chip oscillator clock operation (C)
Status transition (C)→(I)
R_MAIN_CtoI()
Transition to HALT mode (I) while CPU is operating with middle-speed on-chip oscillator clock(C)
B

Figure 5.9 Main Processing (2/4)


B
Status transition (I)→(C)
R_MAIN_ItoC()
Status transition (C)→(J)
R_MAIN_CtoJ()
Status transition (J)→(C)
R_MAIN_JtoC()
Status transition (C)→(K)
R_MAIN_CtoK()
Status transition (K)→(C)
R_MAIN_KtoC()
Status transition (C)→(E)
R_MAIN_CtoE()
Status transition (E)→(C)
R_MAIN_EtoC()
Status transition (C)→(B)
R_MAIN_CtoB()
Status transition (B)→(E)
R_MAIN_BtoE()
Status transition (E)→(N)
R_MAIN_EtoN()
C

Release HALT mode (I), then transition to middle-speed on-chip oscillator clock operation (C)

Transition to STOP mod (J) while CPU is operating with middle-speed on-chip oscillator clock (C)

Release STOP mode (J), then transition to middle-speed system clock operation (C)

Transition to SNOOZE mode (K) while CPU is operating with middle-speed on-chip oscillator clock (C)

Release SNOOZE mode (K), then transition to midle-speed on-chip oscillator operation (C)

Transition CPU from middle-speed on-chip oscillator clock operation (C) to low-speed on-chip oscillator clock operation (E)

Transition CPU from low-speed on-chip oscillator clock operation (E) to middle-speed on-chip oscillator clock operation (C)

Transition CPU from middle-speed on-chip oscillator clock operation (C) to high-speed on-chip oscillator clock operation (B)

Transition CPU from high-speed on-chip oscillator clock operation (B) to low-speed on-chip oscillator clock operation (E)

Transition to HALT mode (N) while CPU is operating with low-speed on-chip oscillator clock(E)

Figure 5.10 Main Processing (3/4)

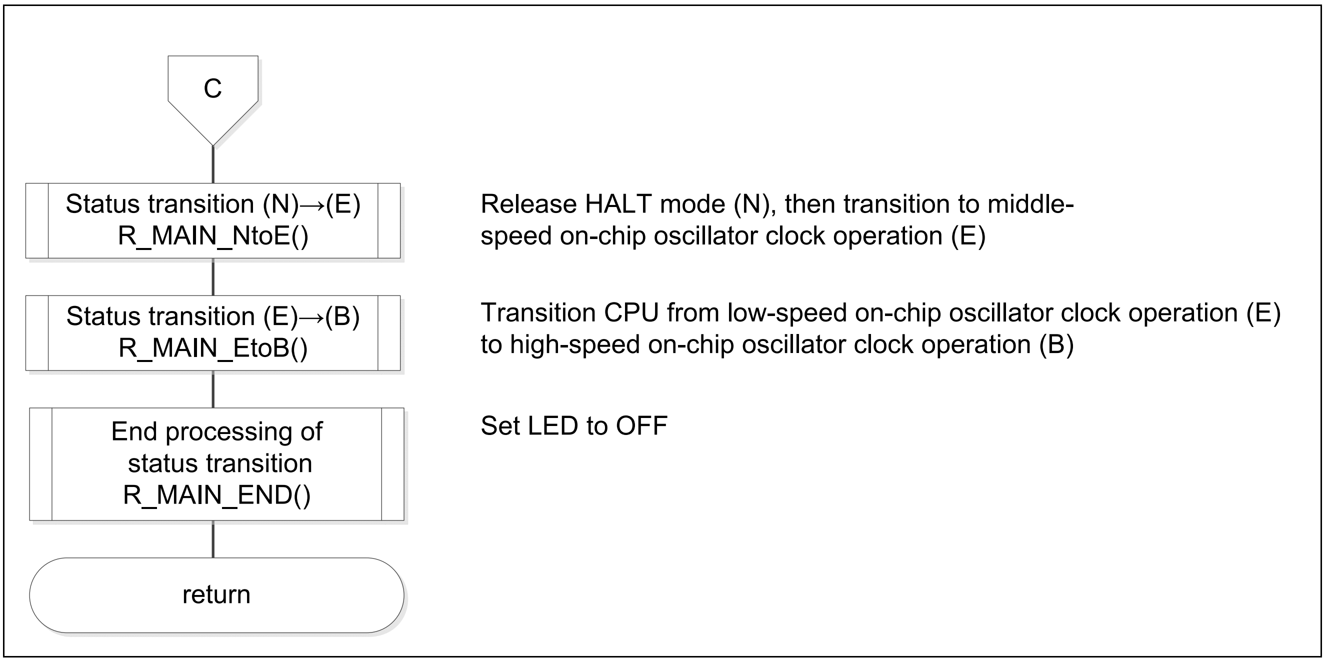

C
Status transition (N)→(E)
R_MAIN_NtoE()
Release HALT mode (N), then transition to middle-speed on-chip oscillator clock operation (E)
Status transition (E)→(B)
R_MAIN_EtoB()
Transition CPU from low-speed on-chip oscillator clock operation (E) to high-speed on-chip oscillator clock operation (B)
End processing of status transition
R_MAIN_END()
Set LED to OFF
return

Figure 5.11 Main Processing (4/4)

Pin input edge detection interrupt (INTP0) setting

- Interrupt request flag register (IF0L)
  Clear the PIF0 interrupt source flag.
- Interrupt mask flag register (MK0L)
  Set PMK0 interrupt mask.

Symbol: IF0L

7	6	5	4	3	2	1	0
PIF5	PIF4	PIF3	PIF2	PIF1	PIF0	LVIIF	WDTIF
x	x	x	x	x	**0**	x	x

Bit 2

PIF0	Interrupt request flag
0	**No interrupt request signal is generated**
1	Interrupt request signal is generated, interrupt request status

Symbol: MK0L

7	6	5	4	3	2	1	0
PMK5	PMK4	PMK3	PMK2	PMK1	PMK0	LVIMK	WDTMK
x	x	x	x	x	**0**	x	x

Bit 2

PMK0	Interrupt servicing control
0	**Interrupt servicing enabled**
1	Interrupt servicing disabled

Note: Refer to the RL78/G11 User’s Manual (Hardware version) for details on how to set registers.

5.6.8 Status Transition AtoB

Figure5.12 shows the flowchart for status transition AtoB.

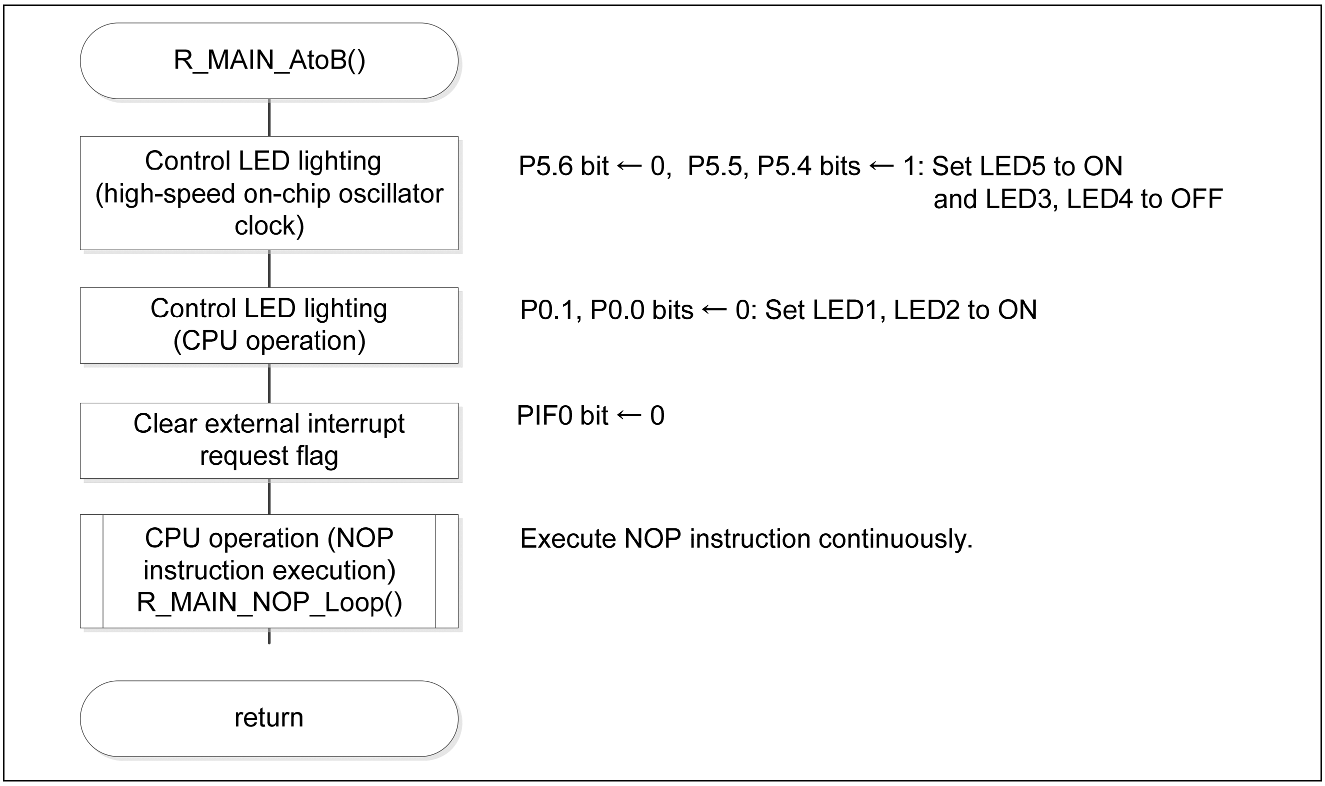

R_MAIN_AtoB()
Control LED lighting (high-speed on-chip oscillator clock)
P5.6 bit ← 0, P5.5, P5.4 bits ← 1: Set LED5 to ON and LED3, LED4 to OFF
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously.
return

Figure 5.12 Status Transition AtoB

5.6.9 CPU operation (NOP instruction execution)

Figure5.13 shows the flowchart for the CPU operation (NOP instruction execution)

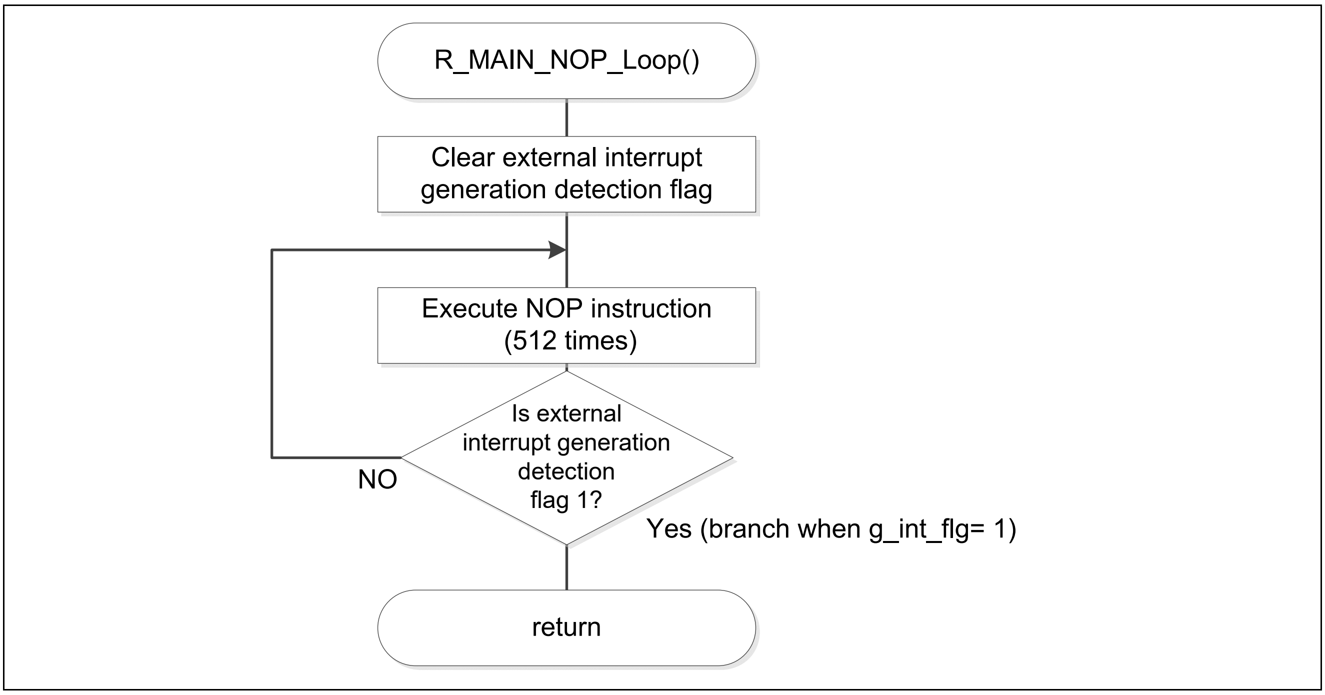

R_MAIN_NOP_Loop()
Clear external interrupt generation detection flag
Execute NOP instruction (512 times)
Is external interrupt generation detection flag 1?
NO
Yes (branch when g_int_flg= 1)
return

Figure 5.13 CPU Operation (NOP instruction execution)

5.6.10 Status Transition BtoD

Figure5.14 and Figure5.15 shows the flowchart for status transition BtoD.

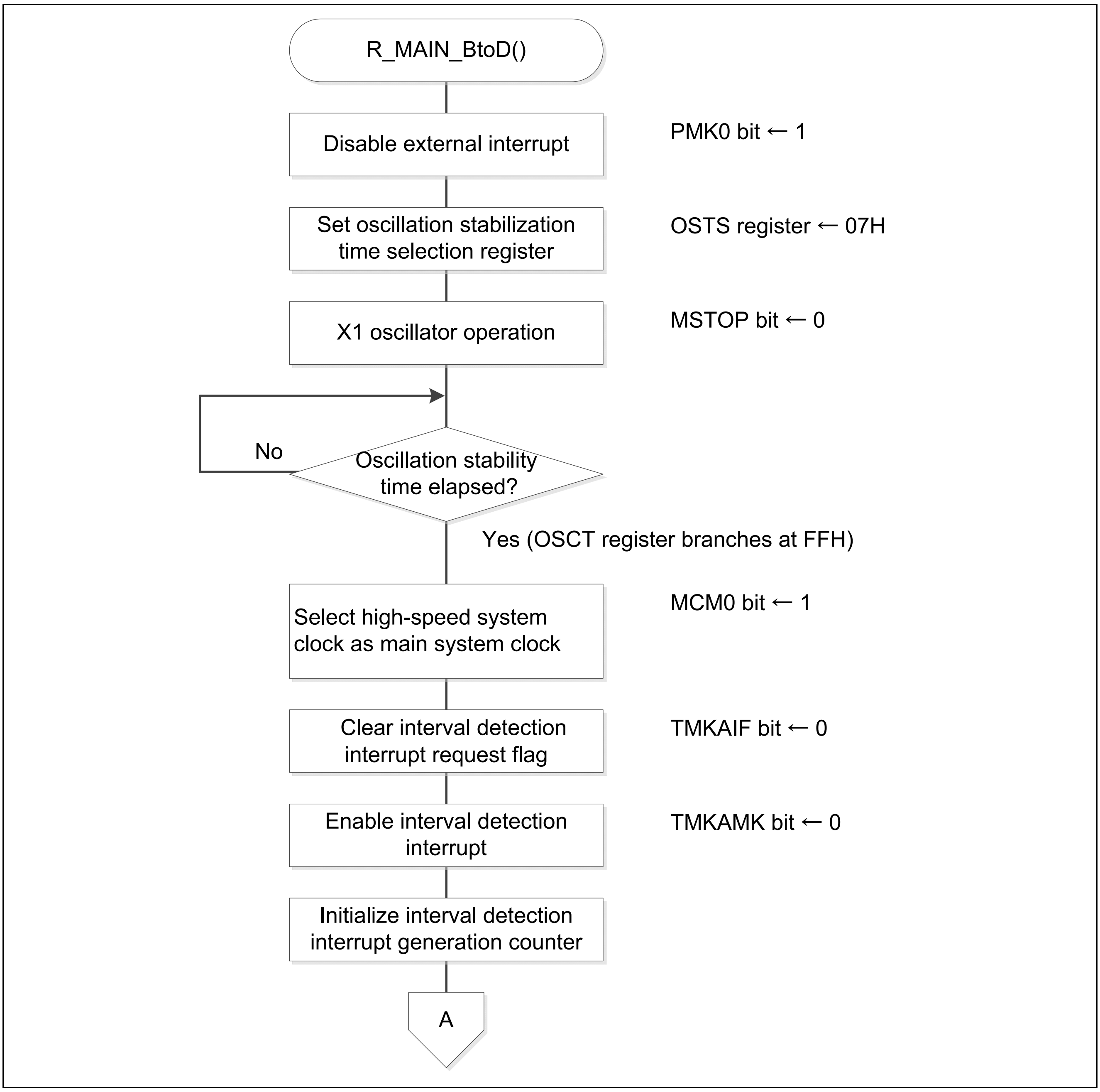

R_MAIN_BtoD()
Disable external interrupt
PMK0 bit ← 1
Set oscillation stabilization time selection register
OSTS register ← 07H
X1 oscillator operation
MSTOP bit ← 0
No
Oscillation stability time elapsed?
Yes (OSCT register branches at FFH)
Select high-speed system clock as main system clock
MCM0 bit ← 1
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Enable interval detection interrupt
TMKAMK bit ← 0
Initialize interval detection interrupt generation counter
A

Figure 5.14 Status Transition BtoD (1/2)

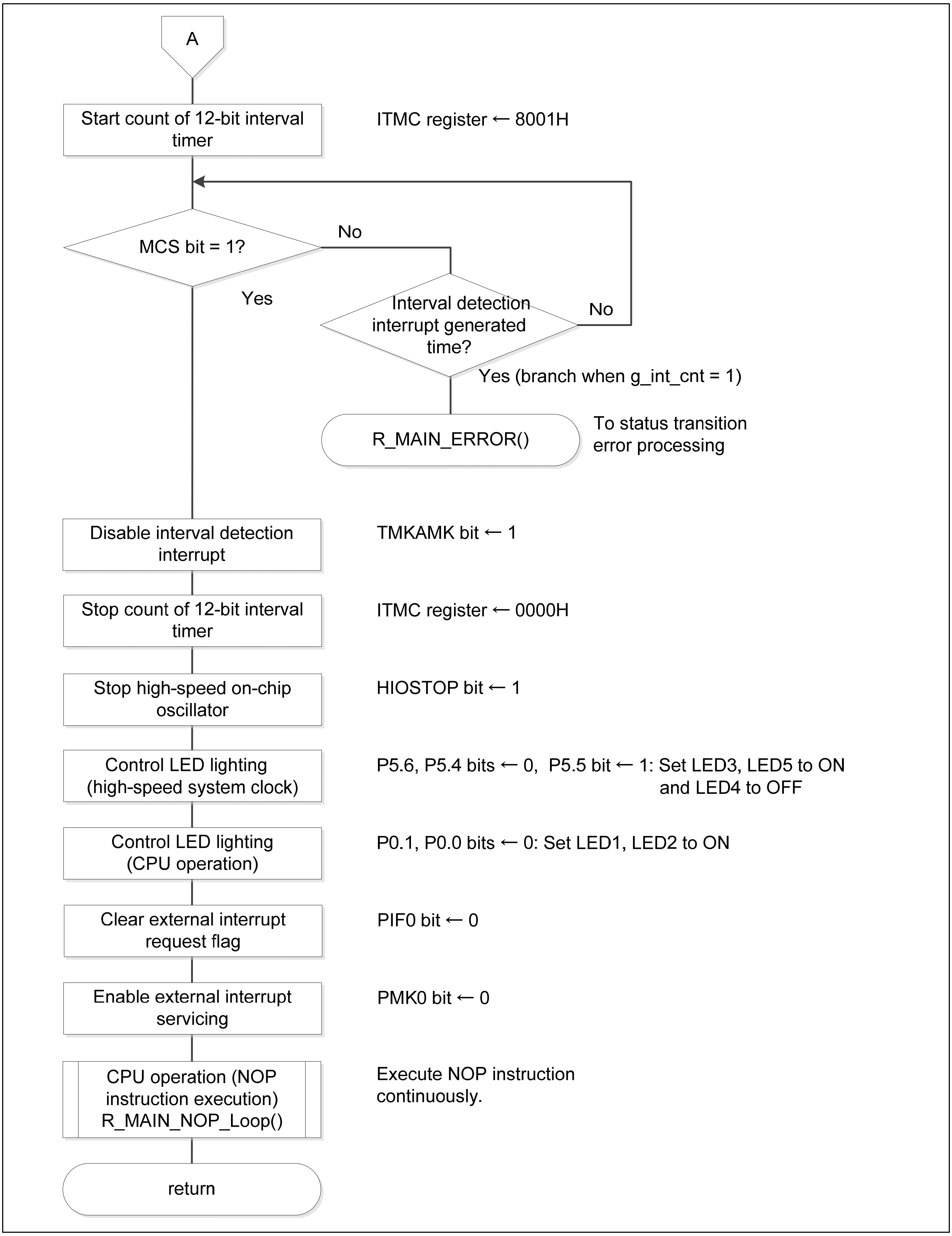

A
Start count of 12-bit interval timer
ITMC register ← 8001H
MCS bit = 1?
No
Yes
Interval detection interrupt generated time?
No
Yes (branch when g_int_cnt = 1)
R_MAIN_ERROR()
To status transition error processing
Disable interval detection interrupt
TMKAMK bit ← 1
Stop count of 12-bit interval timer
ITMC register ← 0000H
Stop high-speed on-chip oscillator
HIOSTOP bit ← 1
Control LED lighting (high-speed system clock)
P5.6, P5.4 bits ← 0, P5.5 bit ← 1: Set LED3, LED5 to ON and LED4 to OFF
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously.
return

Figure 5.15 Status Transition BtoD (2/2)

5.6.11 Error Processing of Status Transition

Figure5.16 shows the flowchart for error processing of status transition

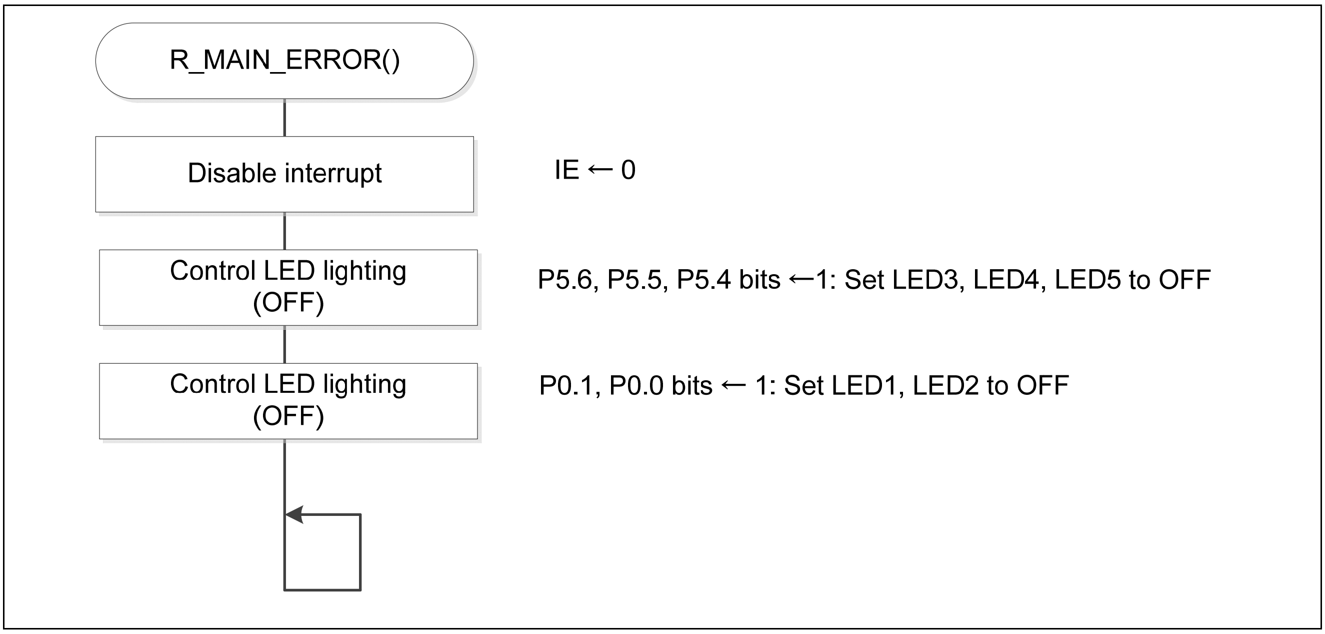

R_MAIN_ERROR()
Disable interrupt
IE ← 0
Control LED lighting (OFF)
P5.6, P5.5, P5.4 bits ←1: Set LED3, LED4, LED5 to OFF
Control LED lighting (OFF)
P0.1, P0.0 bits ← 1: Set LED1, LED2 to OFF

Figure 5.16 Error Processing of Status Transition

5.6.12 Status Transition DtoL

Figure5.17 shows the flowchart for status transition DtoL.

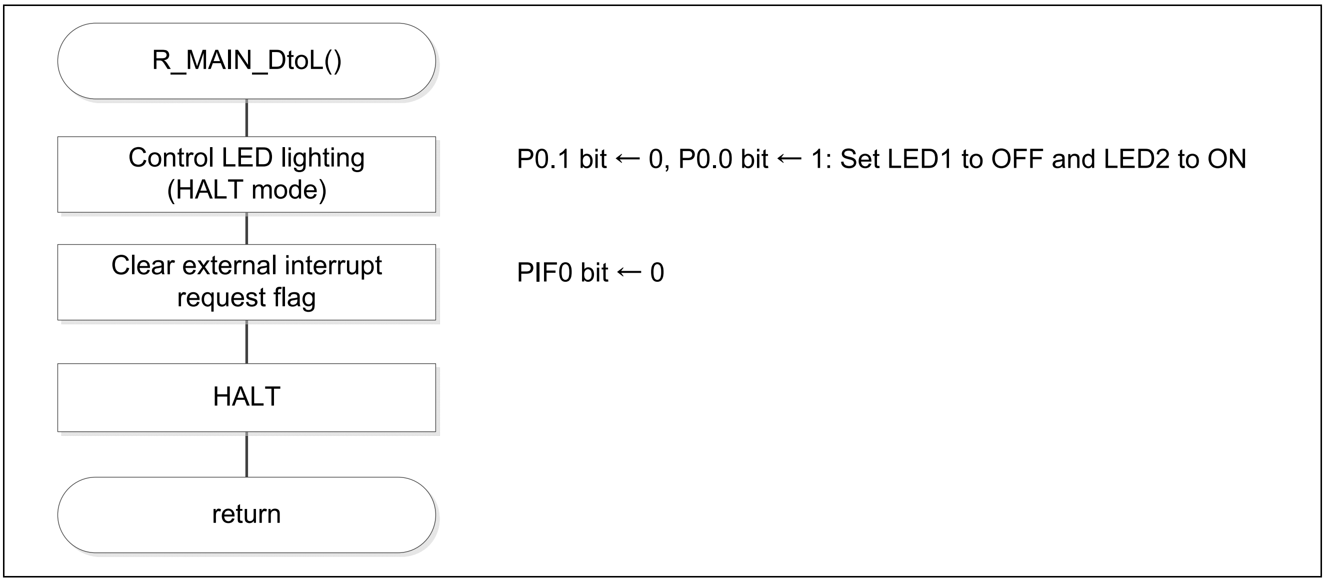

R_MAIN_DtoL()
Control LED lighting (HALT mode)
P0.1 bit ← 0, P0.0 bit ← 1: Set LED1 to OFF and LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
HALT
return

Figure 5.17 Status Transition DtoL

5.6.13 Status Transition LtoD

Figure5.18 shows the flowchart for status transition LtoD.

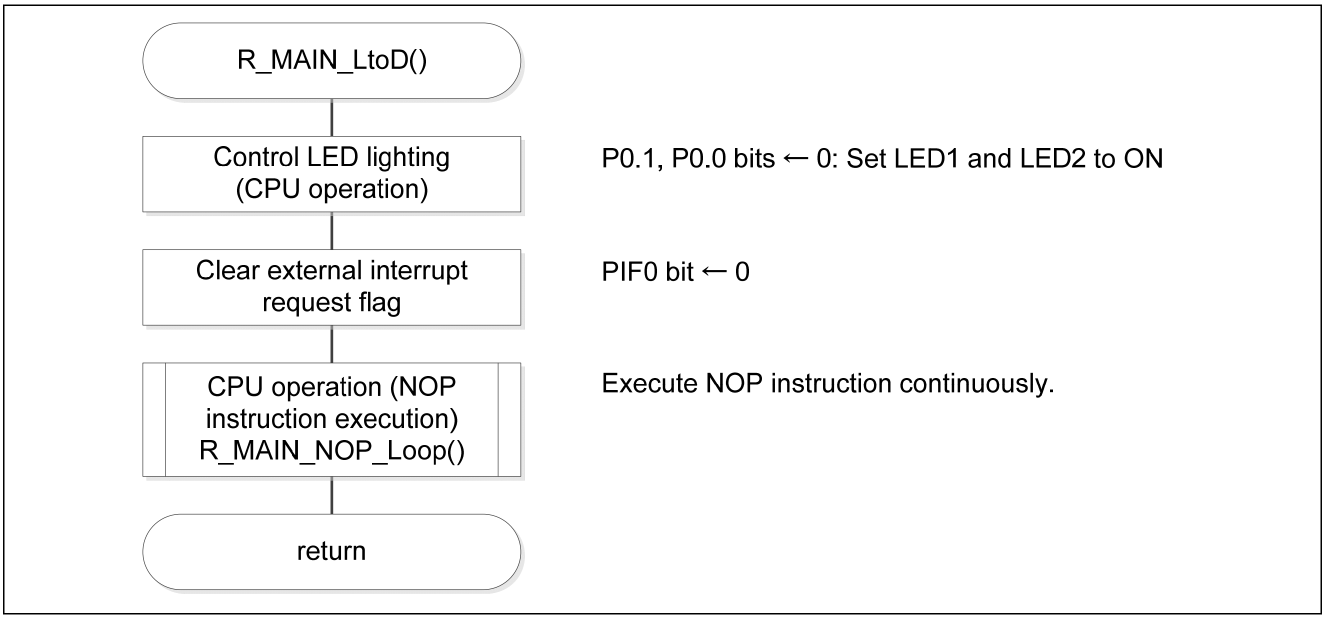

R_MAIN_LtoD()
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1 and LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously.
return

Figure 5.18 Status Transition LtoD

5.6.14 Status Transition DtoM

Figure5.19 shows the flowchart for status transition DtoM.

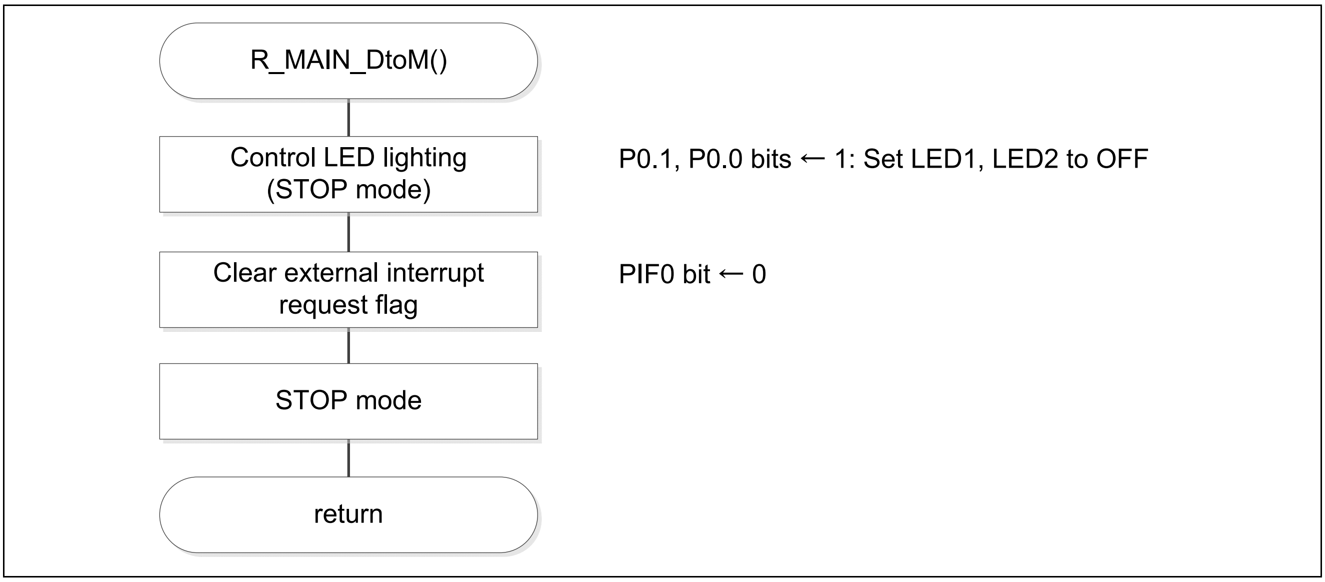

R_MAIN_DtoM()
Control LED lighting (STOP mode)
P0.1, P0.0 bits ← 1: Set LED1, LED2 to OFF
Clear external interrupt request flag
PIF0 bit ← 0
STOP mode
return

Figure 5.19 Status Transition DtoM

5.6.15 Status Transition MtoD

Figure5.20 shows the flowchart for status transition MtoD.

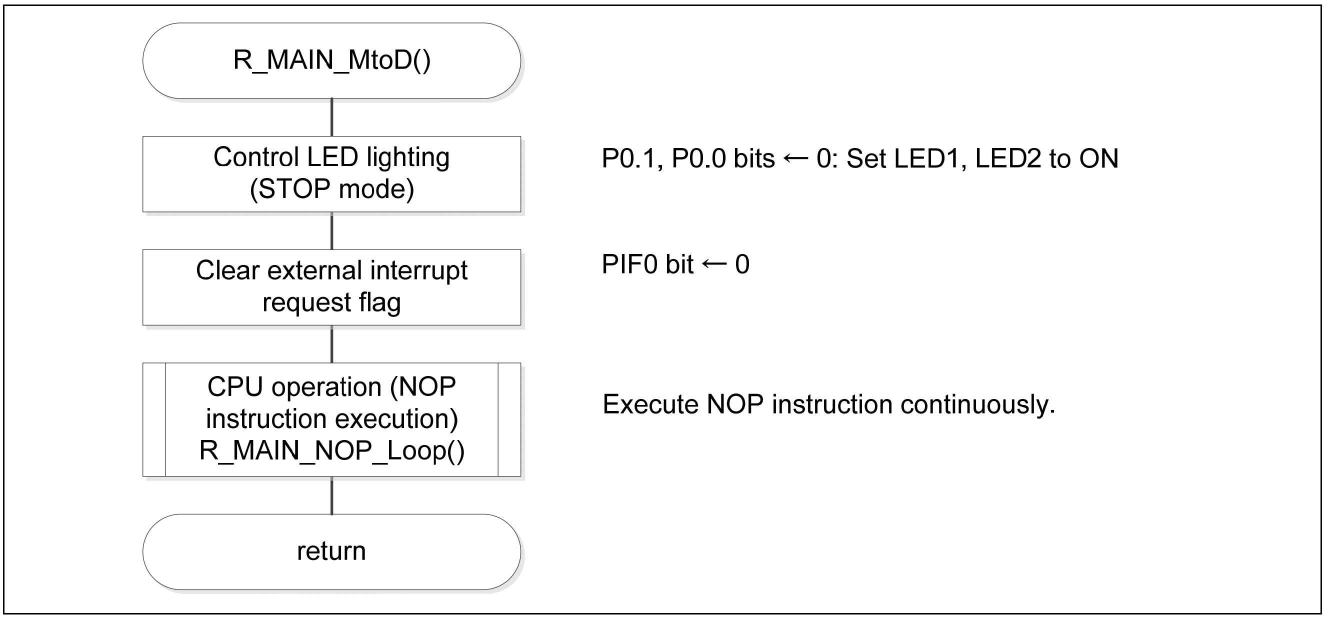

R_MAIN_MtoD()
Control LED lighting (STOP mode)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously.
return

Figure 5.20 Status Transition MtoD

5.6.16 Status Transition DtoB

Figure 5.21 and Figure 5.22 shows the flowchart for status transition DtoB.

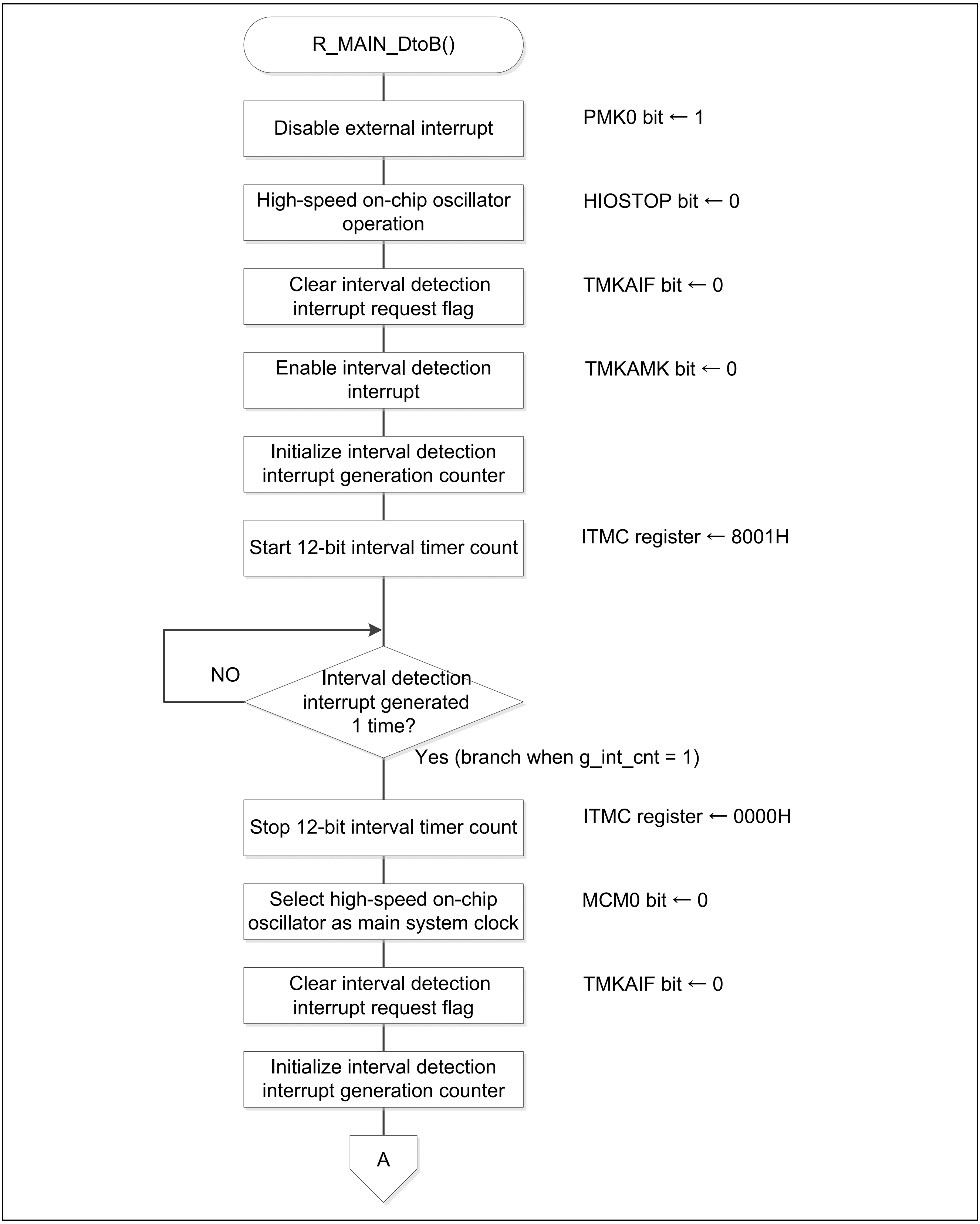

R_MAIN_DtoB()
Disable external interrupt
PMK0 bit ← 1
High-speed on-chip oscillator operation
HIOSTOP bit ← 0
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Enable interval detection interrupt
TMKAMK bit ← 0
Initialize interval detection interrupt generation counter
Start 12-bit interval timer count
ITMC register ← 8001H
NO
Interval detection interrupt generated 1 time?
Yes (branch when g_int_cnt = 1)
Stop 12-bit interval timer count
ITMC register ← 0000H
Select high-speed on-chip oscillator as main system clock
MCM0 bit ← 0
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Initialize interval detection interrupt generation counter
A

Figure 5.21 Status Transition DtoB (1/2)

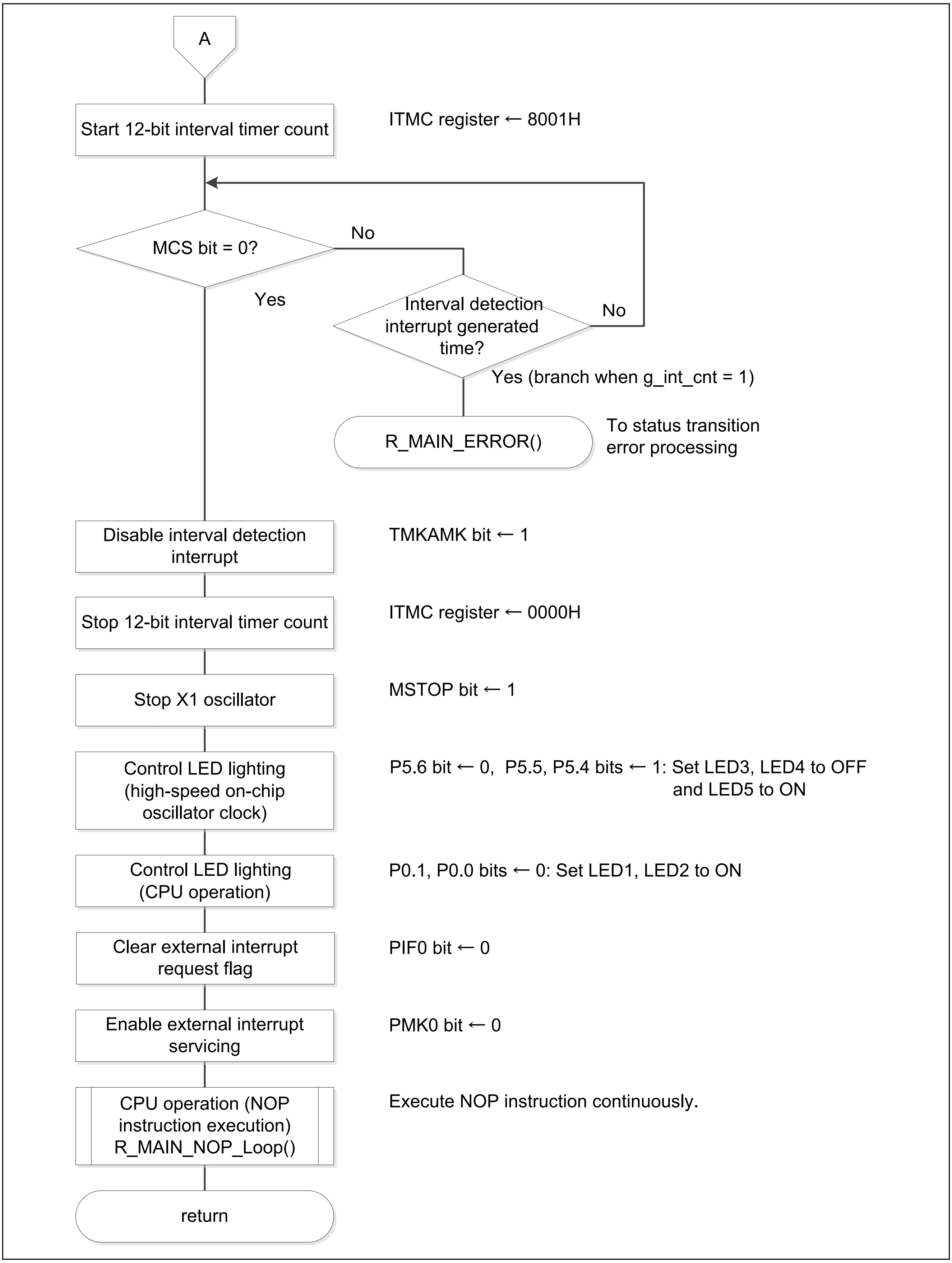

A
Start 12-bit interval timer count
ITMC register ← 8001H
MCS bit = 0?
No
Yes
Interval detection interrupt generated time?
No
Yes (branch when g_int_cnt = 1)
R_MAIN_ERROR()
To status transition error processing
Disable interval detection interrupt
TMKAMK bit ← 1
Stop 12-bit interval timer count
ITMC register ← 0000H
Stop X1 oscillator
MSTOP bit ← 1
Control LED lighting (high-speed on-chip oscillator clock)
P5.6 bit ← 0, P5.5, P5.4 bits ← 1: Set LED3, LED4 to OFF and LED5 to ON
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously.
return

Figure 5.22 Status Transition DtoB(2/2)

5.6.17 Status Transition BtoF

Figure 5.23 shows the flowchart for status transition BtoF.

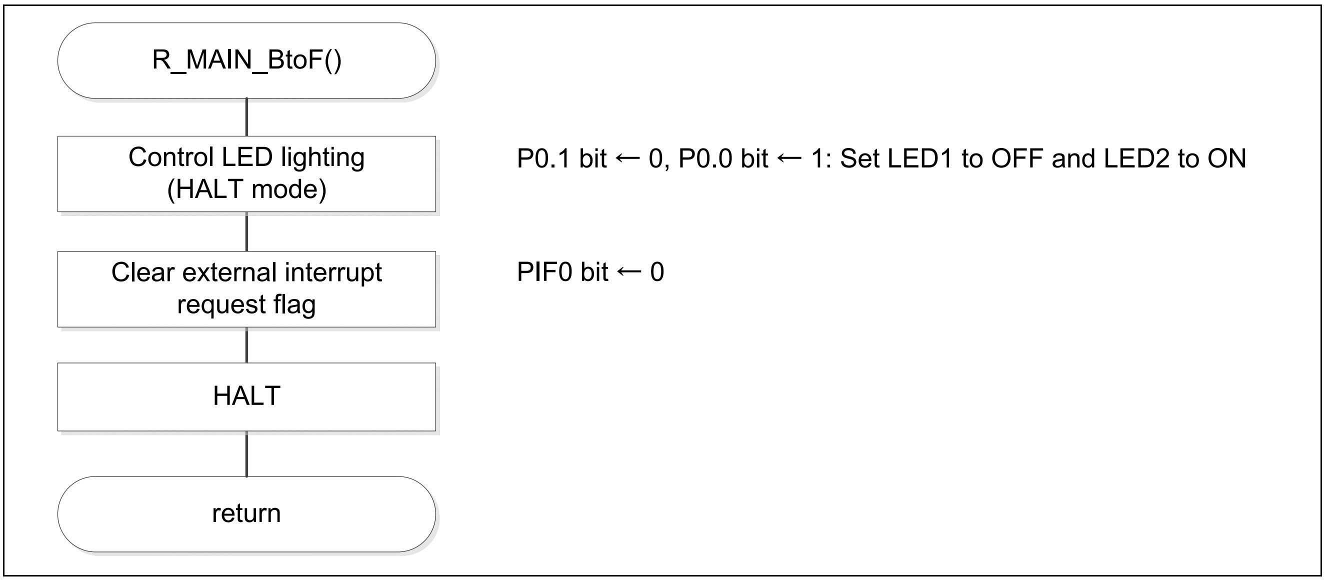

R_MAIN_BtoF()
Control LED lighting (HALT mode)
P0.1 bit ← 0, P0.0 bit ← 1: Set LED1 to OFF and LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
HALT
return

Figure 5.23 Status Transition BtoF

5.6.18 Status Transition FtoB

Figure 5.24 shows the flowchart for status transition FtoB.

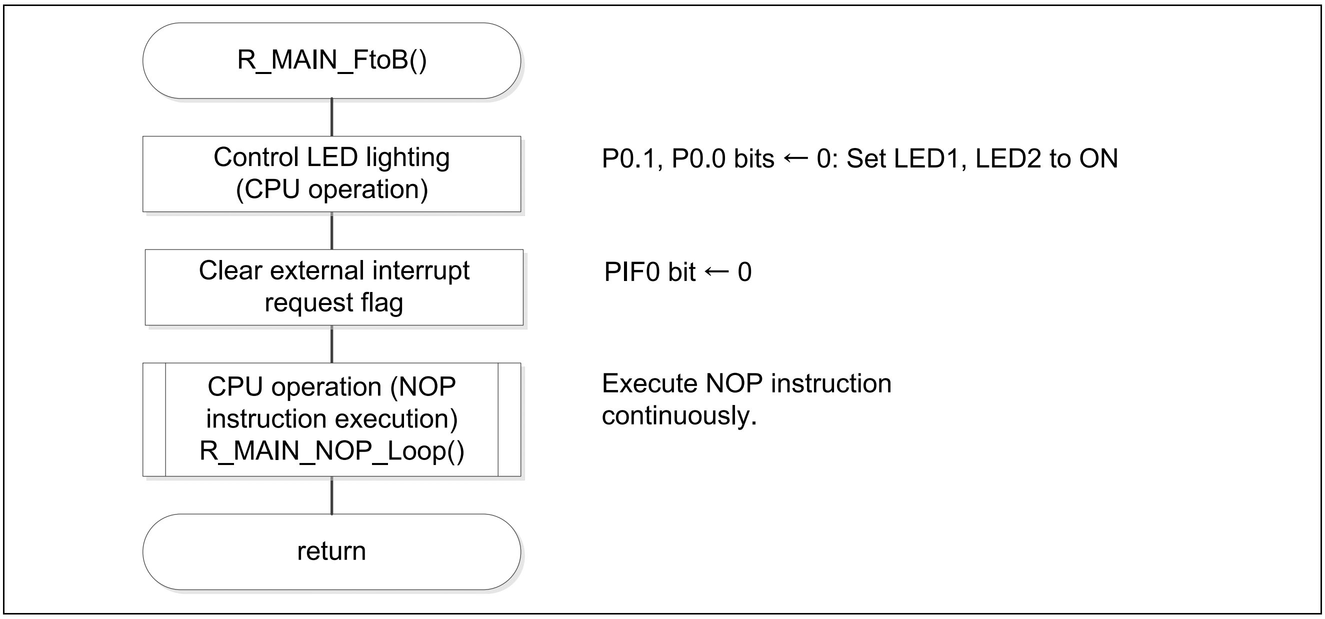

R_MAIN_FtoB()
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously.
return

Figure 5.24 Status Transition FtoB

5.6.19 Status Transition BtoG

Figure 5.25 shows the flowchart for status transition BtoG.

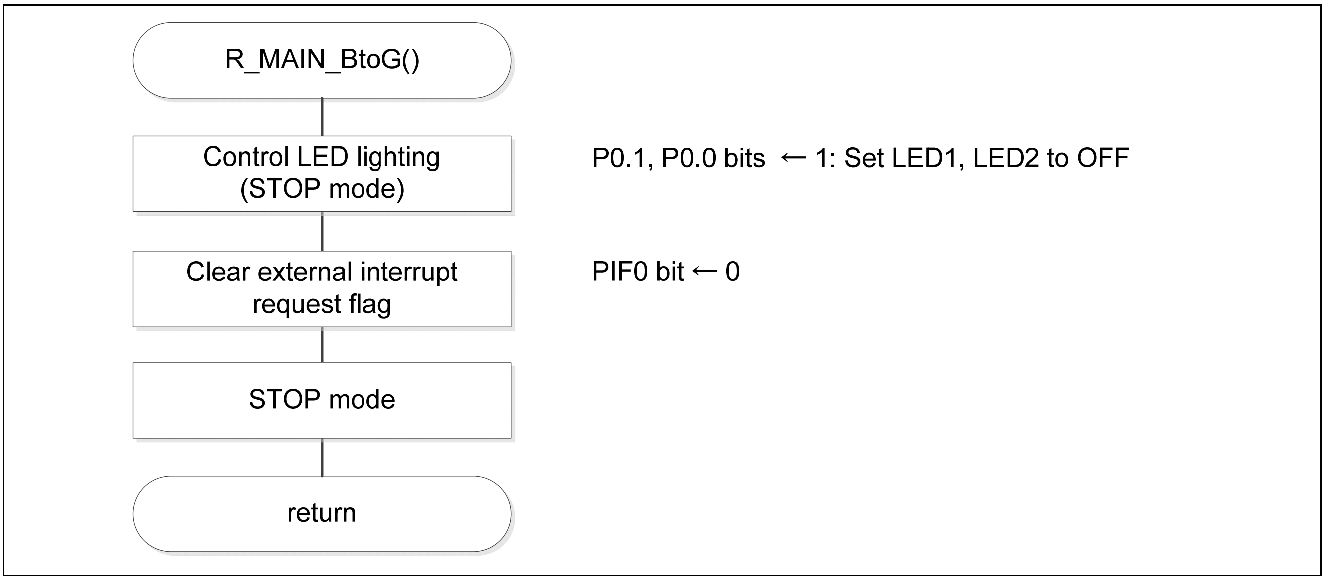

R_MAIN_BtoG()
Control LED lighting (STOP mode)
P0.1, P0.0 bits ← 1: Set LED1, LED2 to OFF
Clear external interrupt request flag
PIF0 bit ← 0
STOP mode
return

Figure 5.25 Status Transition BtoG

5.6.20 Status Transition GtoB

Figure 5.26 shows the flowchart for status transition GtoB.

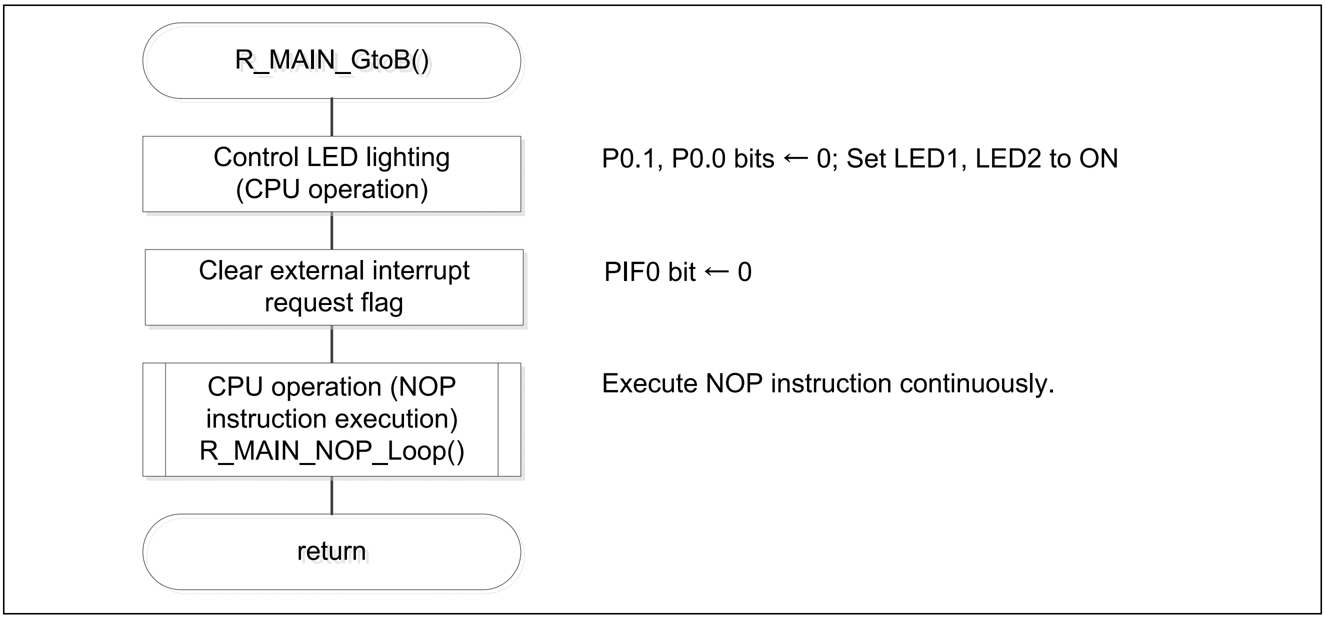

R_MAIN_GtoB()
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0; Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously.
return

Figure 5.26 Status Transition GtoB

5.6.21 Status Transition BtoH

Figure 5.27 shows the flowchart for status transition BtoH.

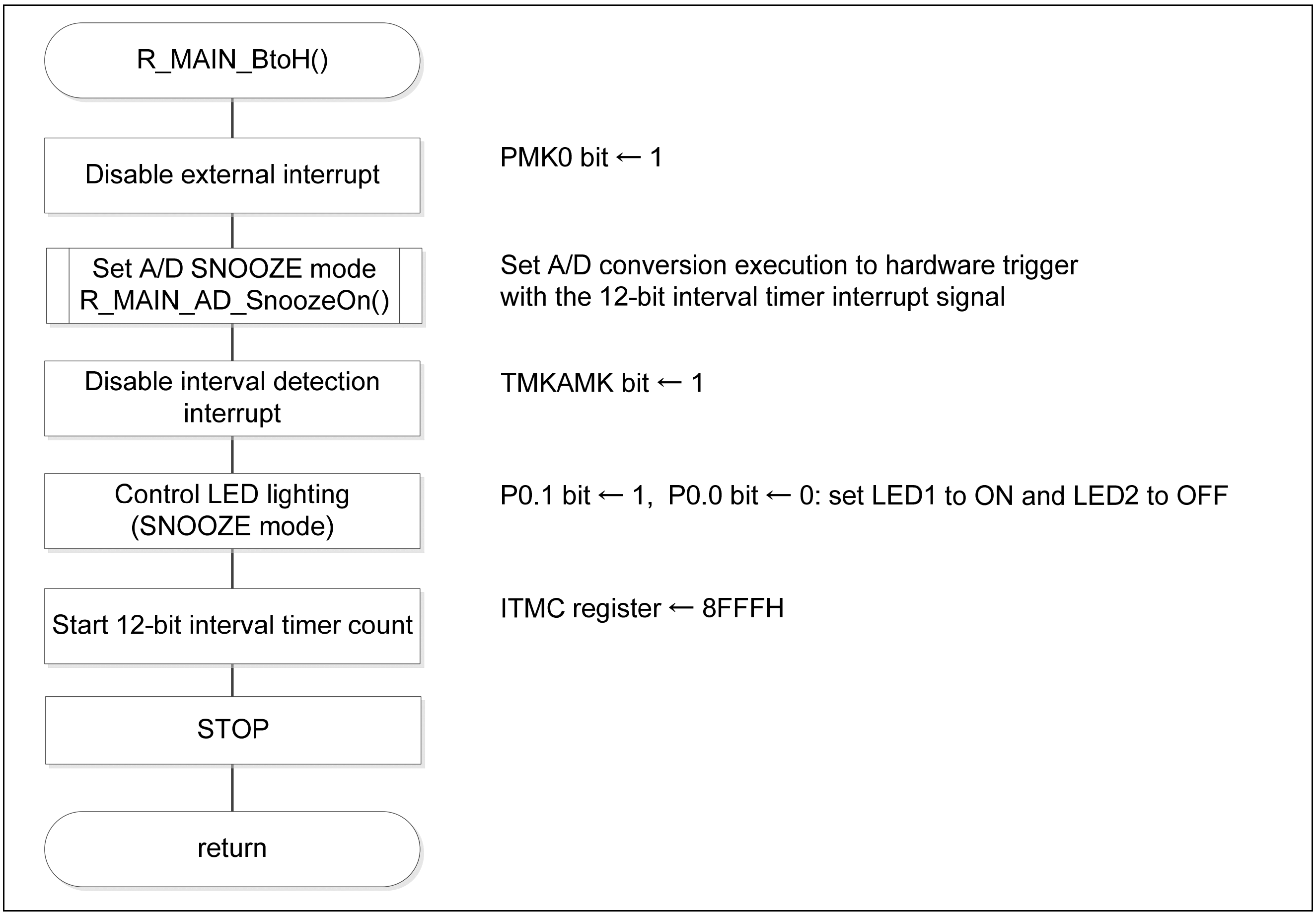

R_MAIN_BtoH()
Disable external interrupt
PMK0 bit ← 1
Set A/D SNOOZE mode R_MAIN_AD_SnoozeOn()
Set A/D conversion execution to hardware trigger with the 12-bit interval timer interrupt signal
Disable interval detection interrupt
TMKAMK bit ← 1
Control LED lighting (SNOOZE mode)
P0.1 bit ← 1, P0.0 bit ← 0: set LED1 to ON and LED2 to OFF
Start 12-bit interval timer count
ITMC register ← 8FFFH
STOP
return

Figure 5.27 Status Transition BtoH

5.6.22 A/D Converter Setting

Figure 5.28 shows the flowchart for setting the A/D converter.

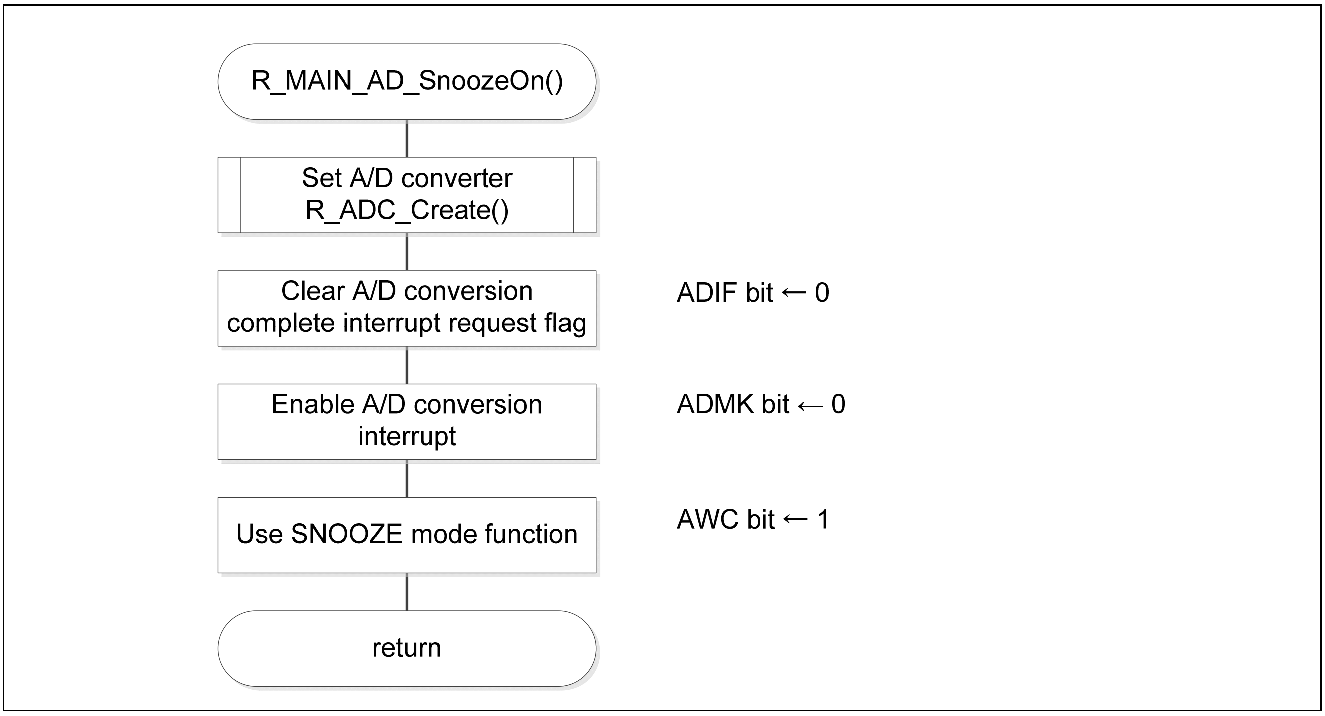

R_MAIN_AD_SnoozeOn()
Set A/D converter R_ADC_Create()
Clear A/D conversion complete interrupt request flag
ADIF bit ← 0
Enable A/D conversion interrupt
ADMK bit ← 0
Use SNOOZE mode function
AWC bit ← 1
return

Figure 5.28 A/D Converter Setting

SNOOZE mode setting

- A/D converter mode register 2 (ADM2)
  Set SNOOZE mode.

Symbol: ADM2

7	6	5	4	3	2	1	0
ADREFP1	ADREFP0	ADREFM	0	ADCRK	AWC	0	ADTYP
x	x	x	0	x	**1**	0	x

Bit 2

AWC	Specification of SNOOZE mode
0	Do not use the SNOOZE mode function
1	**Use the SNOOZE mode function**

Note: Refer to the RL78/G11 User's Manual (hardware version) for detailed explanations on how to set registers.

5.6.23 A/D Converter Initial Setting

Figure 5.29 shows the flowchart for setting the A/D converter.

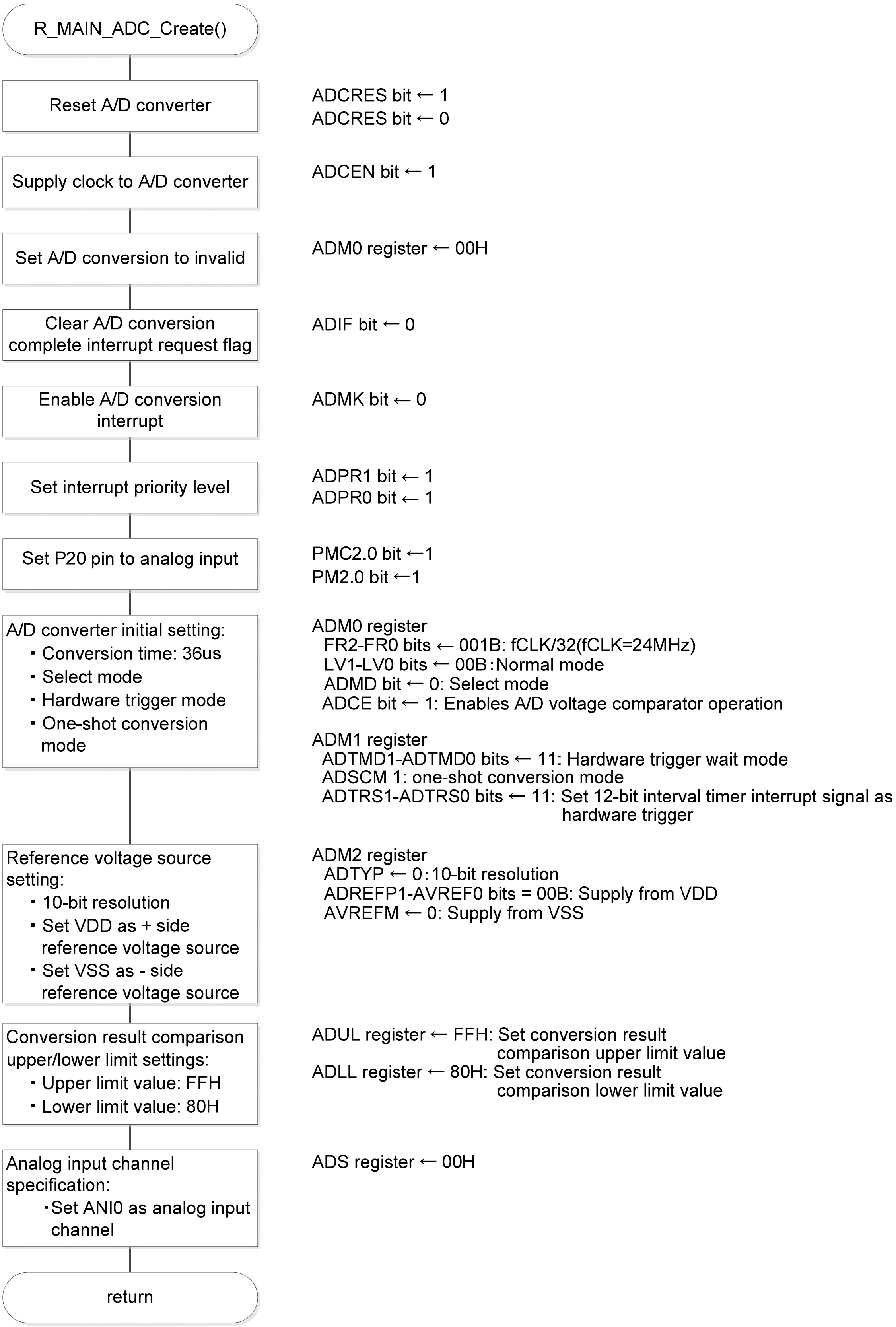

R_MAIN_ADC_Create()
Reset A/D converter
ADCRES bit ← 1
ADCRES bit ← 0
Supply clock to A/D converter
ADCEN bit ← 1
Set A/D conversion to invalid
ADM0 register ← 00H
Clear A/D conversion complete interrupt request flag
ADIF bit ← 0
Enable A/D conversion interrupt
ADMK bit ← 0
Set interrupt priority level
ADPR1 bit ← 1
ADPR0 bit ← 1
Set P20 pin to analog input
PMC2.0 bit ←1
PM2.0 bit ←1
A/D converter initial setting:
・Conversion time: 36us
・Select mode
・Hardware trigger mode
・One-shot conversion mode
ADM0 register
FR2-FR0 bits ← 001B: fCLK/32(fCLK=24MHz)
LV1-LV0 bits ← 00B:Normal mode
ADMD bit ← 0: Select mode
ADCE bit ← 1: Enables A/D voltage comparator operation
ADM1 register
ADTMD1-ADTMD0 bits ← 11: Hardware trigger wait mode
ADSCM 1: one-shot conversion mode
ADTRS1-ADTRS0 bits ← 11: Set 12-bit interval timer interrupt signal as hardware trigger
Reference voltage source setting:
・10-bit resolution
・Set VDD as + side reference voltage source
・Set VSS as - side reference voltage source
ADM2 register
ADTYP ← 0:10-bit resolution
ADREFP1-AVREF0 bits = 00B: Supply from VDD
AVREFM ← 0: Supply from VSS
Conversion result comparison upper/lower limit settings:
・Upper limit value: FFH
・Lower limit value: 80H
ADUL register ← FFH: Set conversion result comparison upper limit value
ADLL register ← 80H: Set conversion result comparison lower limit value
Analog input channel specification:
・Set ANI0 as analog input channel
ADS register ← 00H
return

Figure 5.29 A/D Converter Initial Setting

A/D conversion time and operation mode settings

- A/D converter mode register 0 (ADM0)

  Control the A/D conversion operation.

  Specify the A/D conversion channel selection mode.

Symbol: ADM0

7	6	5	4	3	2	1	0
ADCS	ADMD	FR2	FR1	FR0	LV1	LV0	ADCE
x	**0**	**0**	**0**	**1**	**0**	**0**	**1**

Bit 6

ADMD	Specification of A/D channel selection mode
0	**Select mode**
1	Scan mode

Bits 5-1

ADM0					Mode	Conversion Time					Conversion clock (f_{AD})
FR2	FR1	FR0	LV1	LV0		f_{CLK}= 1MHz	f_{CLK}= 4MHz	f_{CLK}= 8MHz	f_{CLK}= 16MHz	**f_{CLK}= 24MHz**	
0	0	0	**0**	**0**	**Normal 1**	Setting prohibited	Setting prohibited	Setting prohibited	108 µs	72 µs	$f_{CLK}/64$
0	**0**	**1**						108 µs	54 µs	**36 µs**	$f_{CLK}/32$
0	1	0					108 µs	54 µs	27 µs	18 µs	$f_{CLK}/16$
0	1	1					54 µs	27 µs	13.5 µs	9 µs	$f_{CLK}/8$
1	0	0					40.5 µs	20.25 µs	10.125 µs	6.75 µs	$f_{CLK}/6$
1	0	1				135 µs	33.75 µs	16.875 µs	8.4375 µs	5.625 µs	$f_{CLK}/5$
1	1	0				108 µs	27 µs	13.5 µs	6.75 µs	4.5 µs	$f_{CLK}/4$
1	1	1				54 µs	13.5 µs	6.75 µs	3.375 µs	2.25 µs	$f_{CLK}/2$

Bit 0

ADCE	A/D voltage comparator operation control
0	Stops A/D enables comparator operation
1	**Enables A/D voltage comparator operation**

Note: Refer to the RL78/G11 User’s Manual (hardware version) for detailed explanations on how to set registers.

A/D conversion trigger mode setting

- A/D converter mode register 1 (ADM1)

  Select the A/D conversion trigger mode.

  Specify the A/D conversion operation mode.

Symbol: ADM1

7	6	5	4	3	2	1	0
ADTMD1	ADTMD0	ADSCM	0	0	0	ADTRS1	ADTRS0
1	**1**	**1**	0	0	0	**1**	**1**

Bits 7-6

ADTMD1	ADTMD0	Selection of the A/D conversion trigger mode
0	—	Software trigger mode
1	0	Hardware trigger no-wait mode
1	1	**Hardware trigger wait mode**

Bit 5

ADSCM	Specification of the A/D conversion mode
0	Sequential conversion mode
1	**One-shot conversion mode**

Bits 1-0

ADTRS1	ADTRS0	Selection of the hardware trigger signal
0	0	End of timer channel 01 count or capture interrupt signal (INTTM01)
0	1	Event signal selected by ELC
1	0	Setting prohibited
1	**1**	**12-bit interval timer interrupt signal (INTIT)**

Note: Refer to the RL78/G11 User's Manual (hardware version) for detailed explanations on how to set registers.

Reference voltage source setting

- A/D converter mode register 2 (ADM2)
  Set the reference voltage source.

Symbol: ADM2

7	6	5	4	3	2	1	0
ADREFP1	ADREFP0	ADREFM	0	ADCRK	AWC	0	ADTYP
0	**0**	**0**	0	**0**	**0**	0	**0**

Bits 7-6

ADREFP1	ADREFP0	Selection of the + side reference voltage source of the A/D converter
0	**0**	**Supplied from V_{DD}**
0	1	Supplied fromP20/AV_{REFP}/ANI0
1	0	Supplied from the internal reference voltage (1.45 V)
1	1	Setting prohibited

Bit 5

ADREFM	Selection of the - side reference voltage source of the A/D converter
0	**Supplied from V_{SS}**
1	Supplied from P21/AV_{REFM}/ANI1

Bit 3

ADCRK	Checking the upper and lower limit conversion result values
0	**The interrupt signal (INTAD) is output when ADLL register ≤ ADCR register ≤ register ADUL.**
1	The interrupt signal (INTAD) is output when ADCR register ＜ADLL register, and ADUL register ＜ ADCR register.

Bit 2

AWC	Specification of the SNOOZE mode
0	**Do not use the SNOOZE mode function.**
1	Use the SNOOZE mode function.

Bit 0

ADTYP	Selection of the A/D conversion resolution
0	**10-bit resolution**
1	8-bit resolution

Note: Refer to the RL78/G11 User's Manual (hardware version) for detailed explanations on how to set registers.

Conversion result comparison upper/lower limit settings

- Conversion result comparison upper limit setting register (ADUL)
- Conversion result comparison lower limit setting register (ADLL)

Set conversion result comparison upper/lower limit values.

Symbol: ADUL

7	6	5	4	3	2	1	0
ADUL7	ADUL6	ADUL5	ADUL4	ADUL3	ADUL2	ADUL1	ADUL0
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**

Symbol: ADLL

7	6	5	4	3	2	1	0
ADLL7	ADLL6	ADLL5	ADLL4	ADLL3	ADLL2	ADLL1	ADLL0
1	**0**	**0**	**0**	**0**	**0**	**0**	**0**

Input channel specification

- Analog input channel specification register (ADS)

Specify the input channel of the analog voltage to be A/D converted.

Symbol: ADS

7	6	5	4	3	2	1	0
ADISS	0	0	ADS4	ADS3	ADS2	ADS1	ADS0
0	0	0	**0**	**0**	**0**	**0**	**0**

Bits 7, 4-0

ADISS	ADS4	ADS3	ADS2	ADS1	ADS0	Analog input channel	Input source
0	**0**	**0**	**0**	**0**	**0**	**ANI0**	**P20/ANI0 /AV$_{REFP}$ pin**
0	0	0	0	0	1	ANI1	P21/ANI1 /AV$_{REFM}$ pin
0	0	0	0	1	0	ANI2	P22/ANI2 pin
0	0	0	0	1	1	ANI3	P23/ANI3 pin
0	1	0	0	0	0	ANI16	P01/ANI16 pin
0	1	0	0	0	1	ANI17	P00/ANI17 pin
0	1	0	0	1	0	ANI18	P33/ANI18 pin
0	1	0	0	1	1	ANI19	P32/ANI19 pin
0	1	0	1	0	0	ANI20	P31/ANI20 pin
0	1	0	1	0	1	ANI21	P30/ANI21 pin
0	1	0	1	1	0	ANI22	P56/ANI22 pin
0	1	0	1	1	1	—	PGAOUT(PGA output)
1	0	0	0	0	0	—	Temperature sensor 0 output
1	0	0	0	0	1	—	Internal reference voltage output (1.45 V)
Other than the above						Setting prohibited	

Note: Refer to the RL78/G11 User's Manual (hardware version) for detailed explanations on how to set registers.

5.6.24 Status Transition HtoB

Figure 5.30 shows the status transition HtoB.

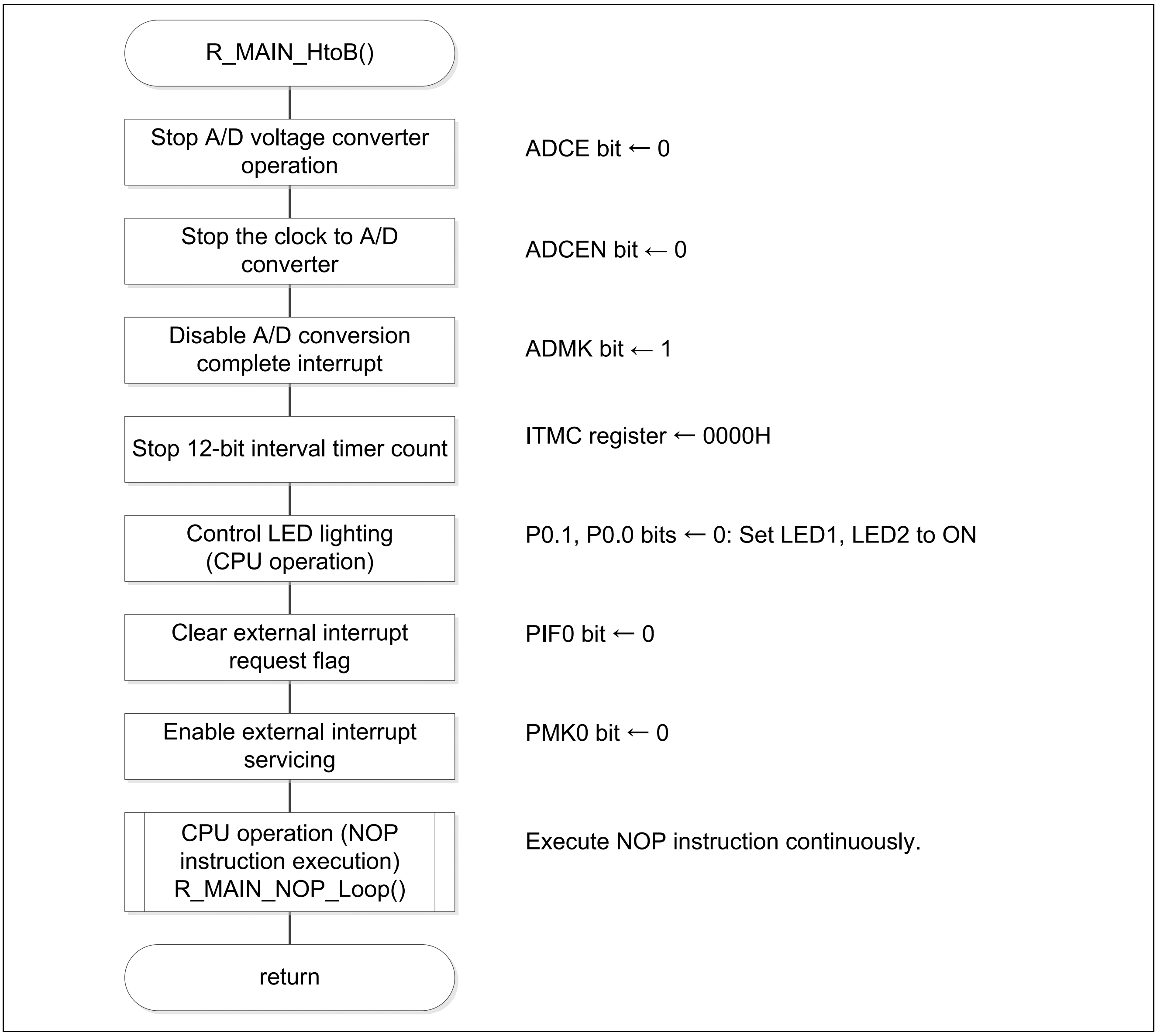

R_MAIN_HtoB()
Stop A/D voltage converter operation
ADCE bit ← 0
Stop the clock to A/D converter
ADCEN bit ← 0
Disable A/D conversion complete interrupt
ADMK bit ← 1
Stop 12-bit interval timer count
ITMC register ← 0000H
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously.
return

Figure 5.30 Status Transition HtoB

5.6.25 Status Transition BtoC

Figure 5.31 and Figure 5.32 shows the status transition BtoC.

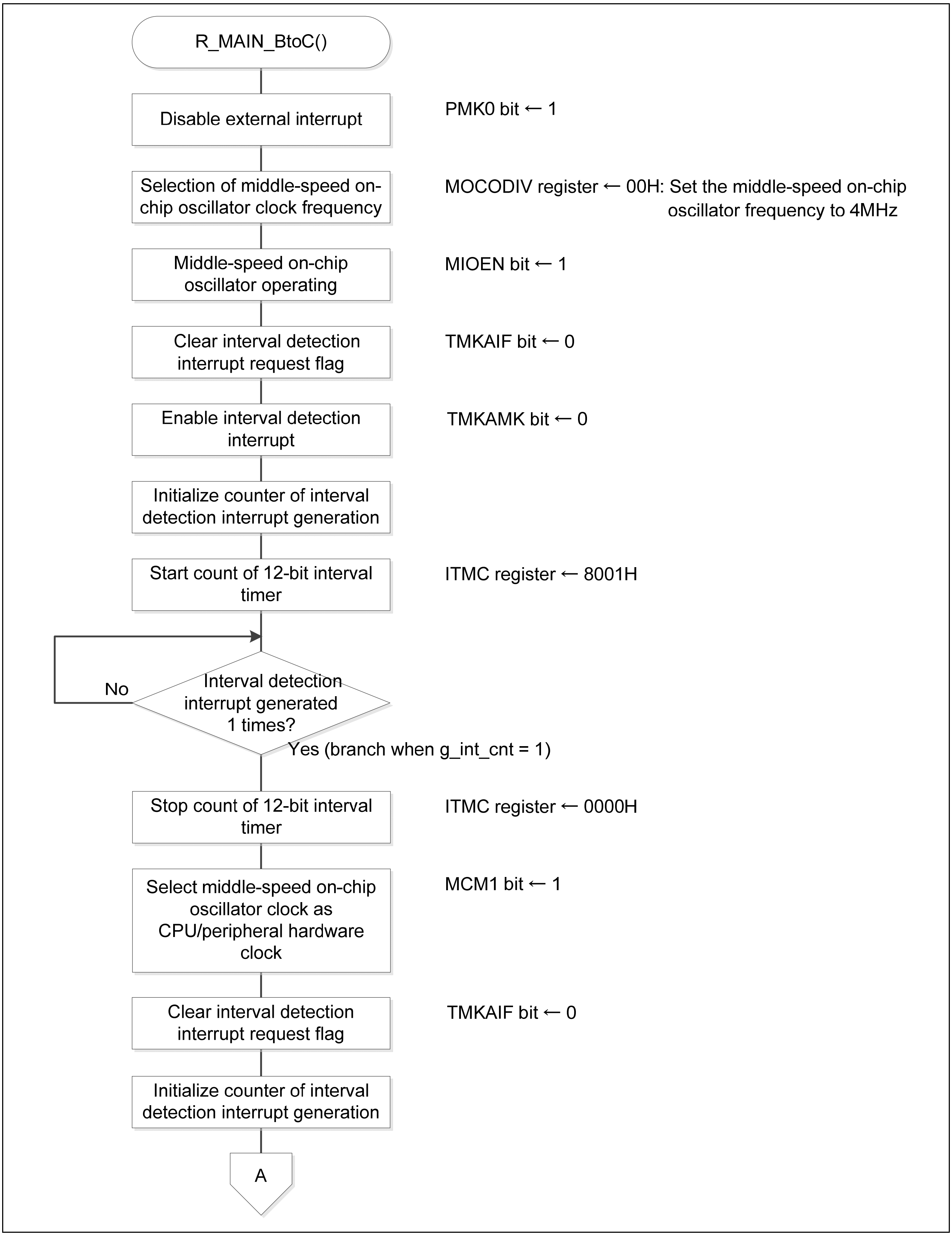

R_MAIN_BtoC()
Disable external interrupt
PMK0 bit ← 1
Selection of middle-speed on-chip oscillator clock frequency
MOCODIV register ← 00H: Set the middle-speed on-chip oscillator frequency to 4MHz
Middle-speed on-chip oscillator operating
MIOEN bit ← 1
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Enable interval detection interrupt
TMKAMK bit ← 0
Initialize counter of interval detection interrupt generation
Start count of 12-bit interval timer
ITMC register ← 8001H
No
Interval detection interrupt generated 1 times?
Yes (branch when g_int_cnt = 1)
Stop count of 12-bit interval timer
ITMC register ← 0000H
Select middle-speed on-chip oscillator clock as CPU/peripheral hardware clock
MCM1 bit ← 1
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Initialize counter of interval detection interrupt generation
A

Figure 5.31 Status Transition BtoC(1/2)

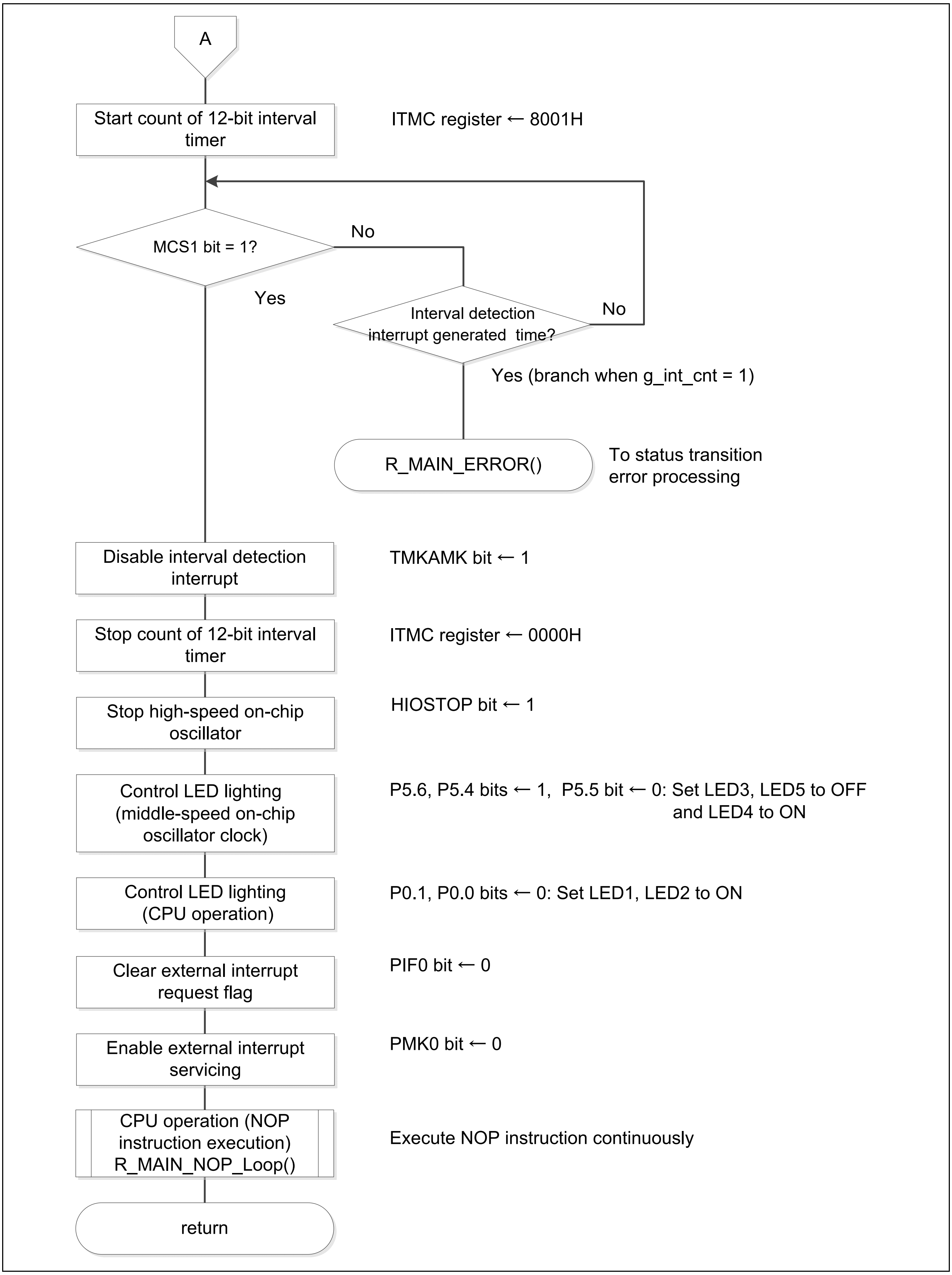

A
Start count of 12-bit interval timer
ITMC register ← 8001H
MCS1 bit = 1?
No
Yes
Interval detection interrupt generated time?
No
Yes (branch when g_int_cnt = 1)
R_MAIN_ERROR()
To status transition error processing
Disable interval detection interrupt
TMKAMK bit ← 1
Stop count of 12-bit interval timer
ITMC register ← 0000H
Stop high-speed on-chip oscillator
HIOSTOP bit ← 1
Control LED lighting (middle-speed on-chip oscillator clock)
P5.6, P5.4 bits ← 1, P5.5 bit ← 0: Set LED3, LED5 to OFF and LED4 to ON
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously
return

Figure 5.32 Status Transition BtoC(2/2)

5.6.26 Status Transition CtoD

Figure 5.33 and Figure 5.34 shows the status transition CtoD.

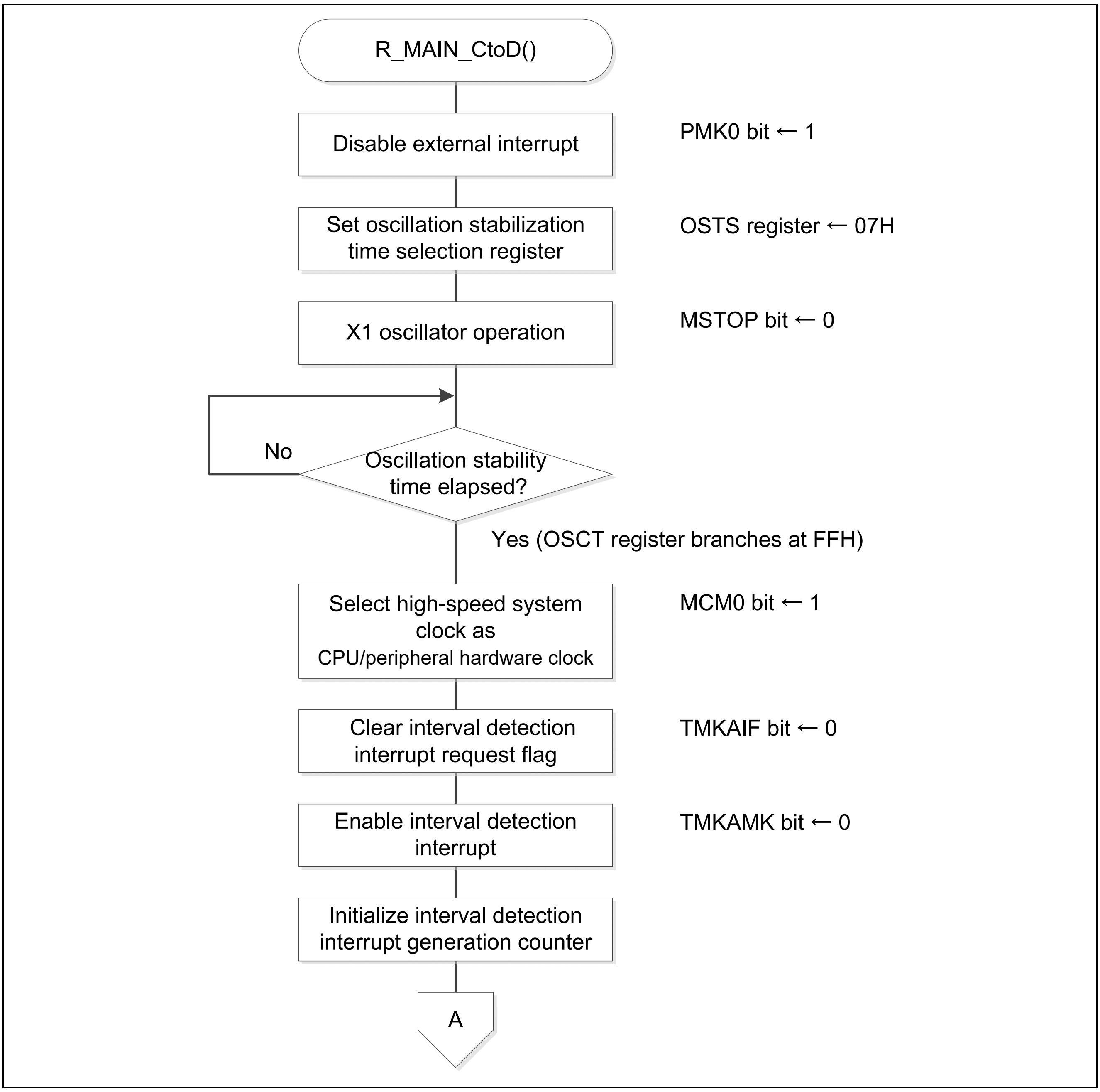

R_MAIN_CtoD()
Disable external interrupt
PMK0 bit ← 1
Set oscillation stabilization time selection register
OSTS register ← 07H
X1 oscillator operation
MSTOP bit ← 0
No
Oscillation stability time elapsed?
Yes (OSCT register branches at FFH)
Select high-speed system clock as CPU/peripheral hardware clock
MCM0 bit ← 1
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Enable interval detection interrupt
TMKAMK bit ← 0
Initialize interval detection interrupt generation counter
A

Figure 5.33 Status Transition CtoD(1/2)

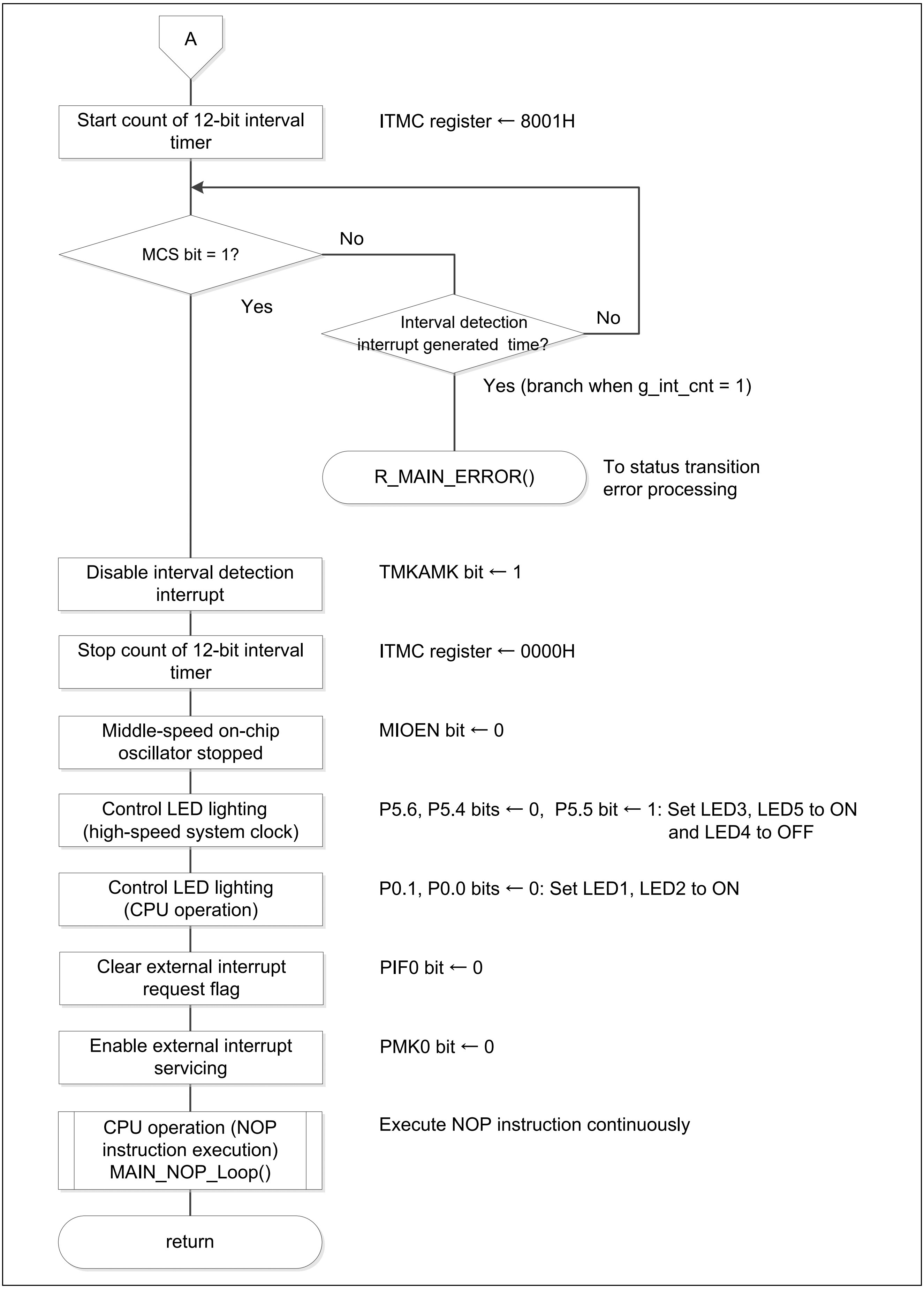

A
Start count of 12-bit interval timer
ITMC register ← 8001H
MCS bit = 1?
No
Yes
Interval detection interrupt generated time?
No
Yes (branch when g_int_cnt = 1)
R_MAIN_ERROR()
To status transition error processing
Disable interval detection interrupt
TMKAMK bit ← 1
Stop count of 12-bit interval timer
ITMC register ← 0000H
Middle-speed on-chip oscillator stopped
MIOEN bit ← 0
Control LED lighting (high-speed system clock)
P5.6, P5.4 bits ← 0, P5.5 bit ← 1: Set LED3, LED5 to ON and LED4 to OFF
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) MAIN_NOP_Loop()
Execute NOP instruction continuously
return

Figure 5.34 Status Transition CtoD(2/2)

5.6.27 Status Transition DtoE

Figure 5.35 and Figure5.36 shows the status transition DtoE.

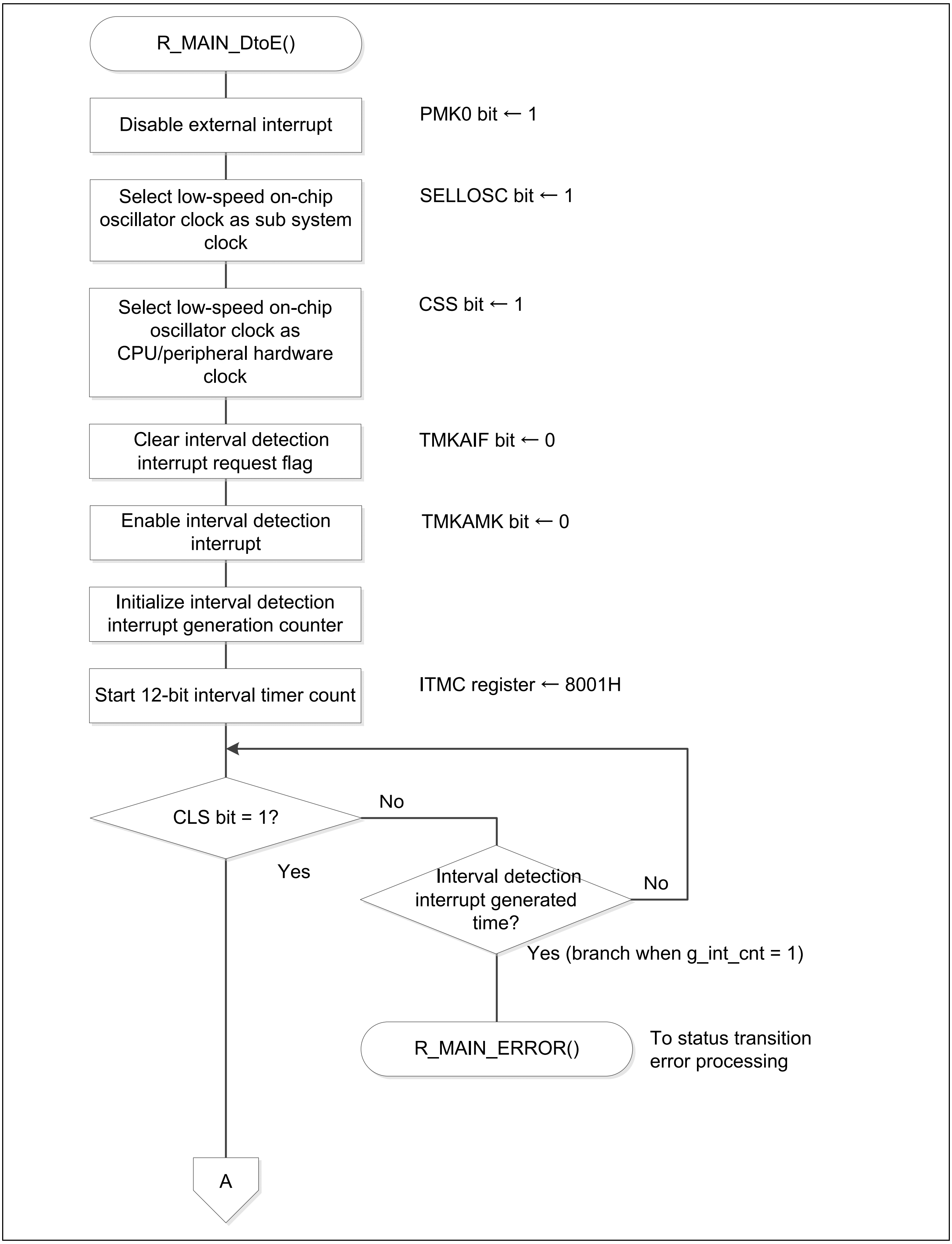

R_MAIN_DtoE()
Disable external interrupt
PMK0 bit ← 1
Select low-speed on-chip oscillator clock as sub system clock
SELLOSC bit ← 1
Select low-speed on-chip oscillator clock as CPU/peripheral hardware clock
CSS bit ← 1
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Enable interval detection interrupt
TMKAMK bit ← 0
Initialize interval detection interrupt generation counter
Start 12-bit interval timer count
ITMC register ← 8001H
CLS bit = 1?
No
Yes
Interval detection interrupt generated time?
No
Yes (branch when g_int_cnt = 1)
R_MAIN_ERROR()
To status transition error processing
A

Figure 5.35 Status Transition DtoE(1/2)

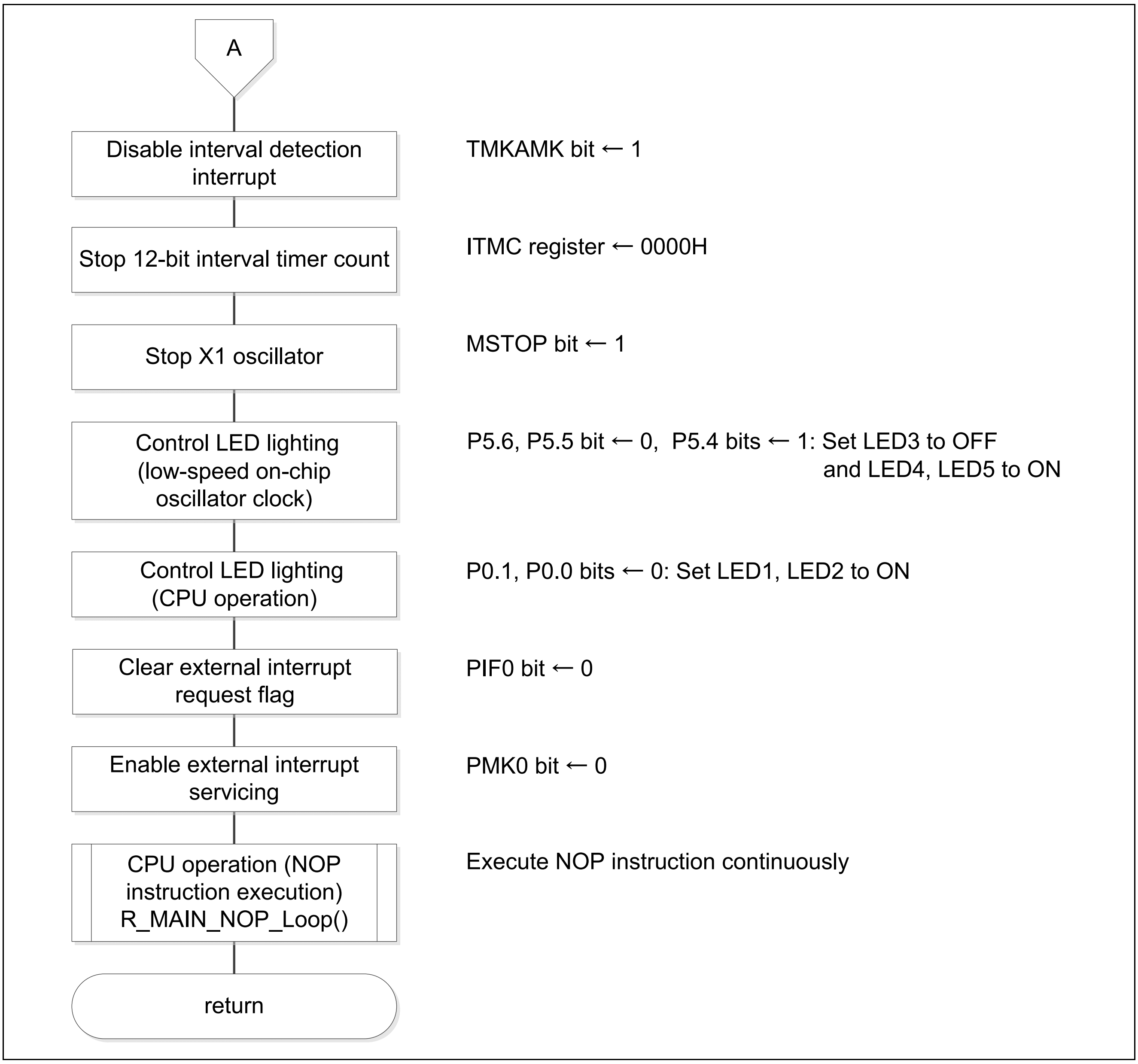

A
Disable interval detection interrupt
Stop 12-bit interval timer count
Stop X1 oscillator
Control LED lighting (low-speed on-chip oscillator clock)
Control LED lighting (CPU operation)
Clear external interrupt request flag
Enable external interrupt servicing
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
return
TMKAMK bit ← 1
ITMC register ← 0000H
MSTOP bit ← 1
P5.6, P5.5 bit ← 0, P5.4 bits ← 1: Set LED3 to OFF and LED4, LED5 to ON
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
PIF0 bit ← 0
PMK0 bit ← 0
Execute NOP instruction continuously

Figure 5.36 Status Transition DtoE(2/2)

5.6.28 Status Transition EtoD

Figure 5.37 and Figure 5.38 shows the status transition EtoD.

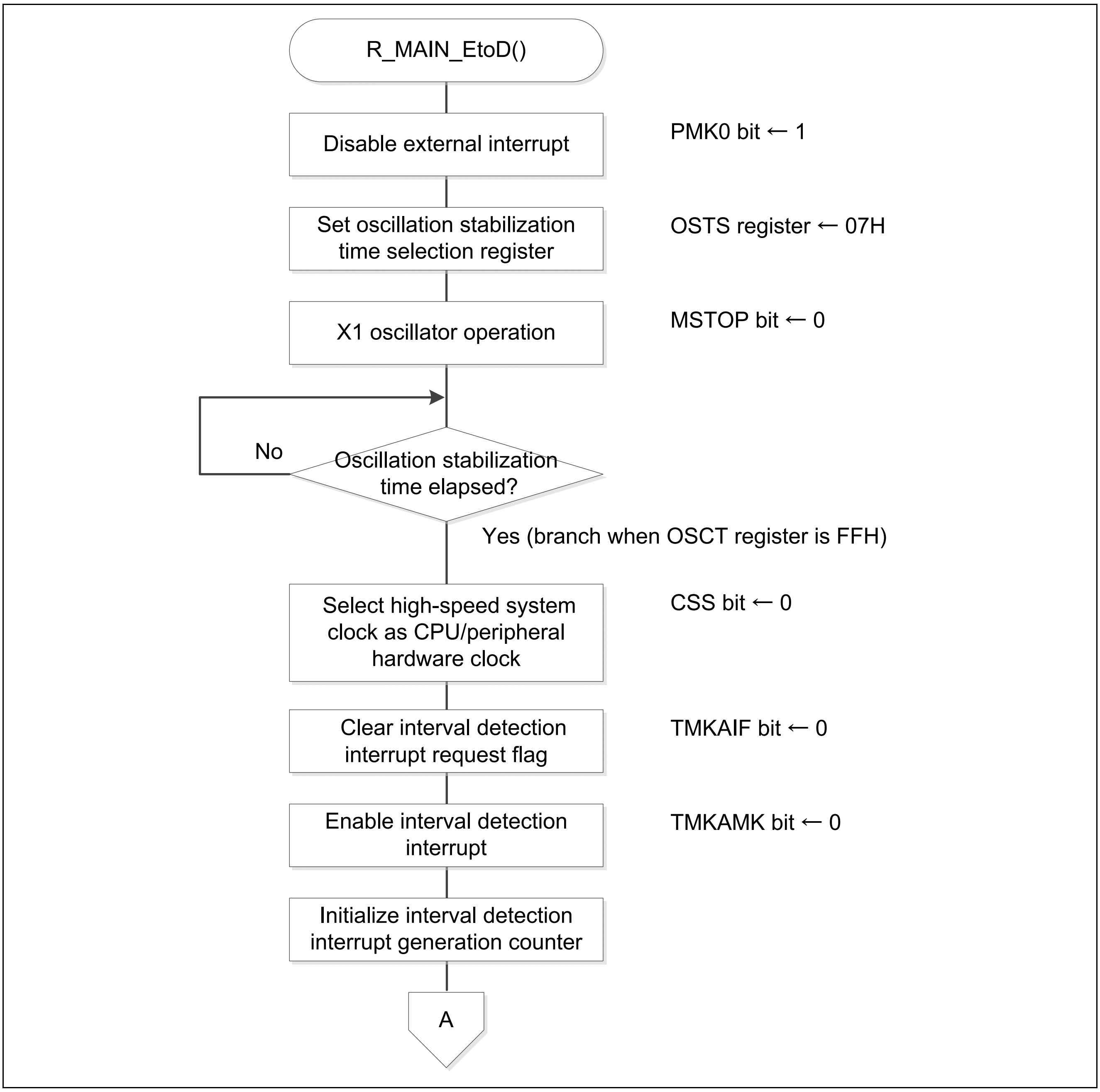

R_MAIN_EtoD()
Disable external interrupt
PMK0 bit ← 1
Set oscillation stabilization time selection register
OSTS register ← 07H
X1 oscillator operation
MSTOP bit ← 0
No
Oscillation stabilization time elapsed?
Yes (branch when OSCT register is FFH)
Select high-speed system clock as CPU/peripheral hardware clock
CSS bit ← 0
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Enable interval detection interrupt
TMKAMK bit ← 0
Initialize interval detection interrupt generation counter
A

Figure 5.37 Status Transition EtoD(1/2)

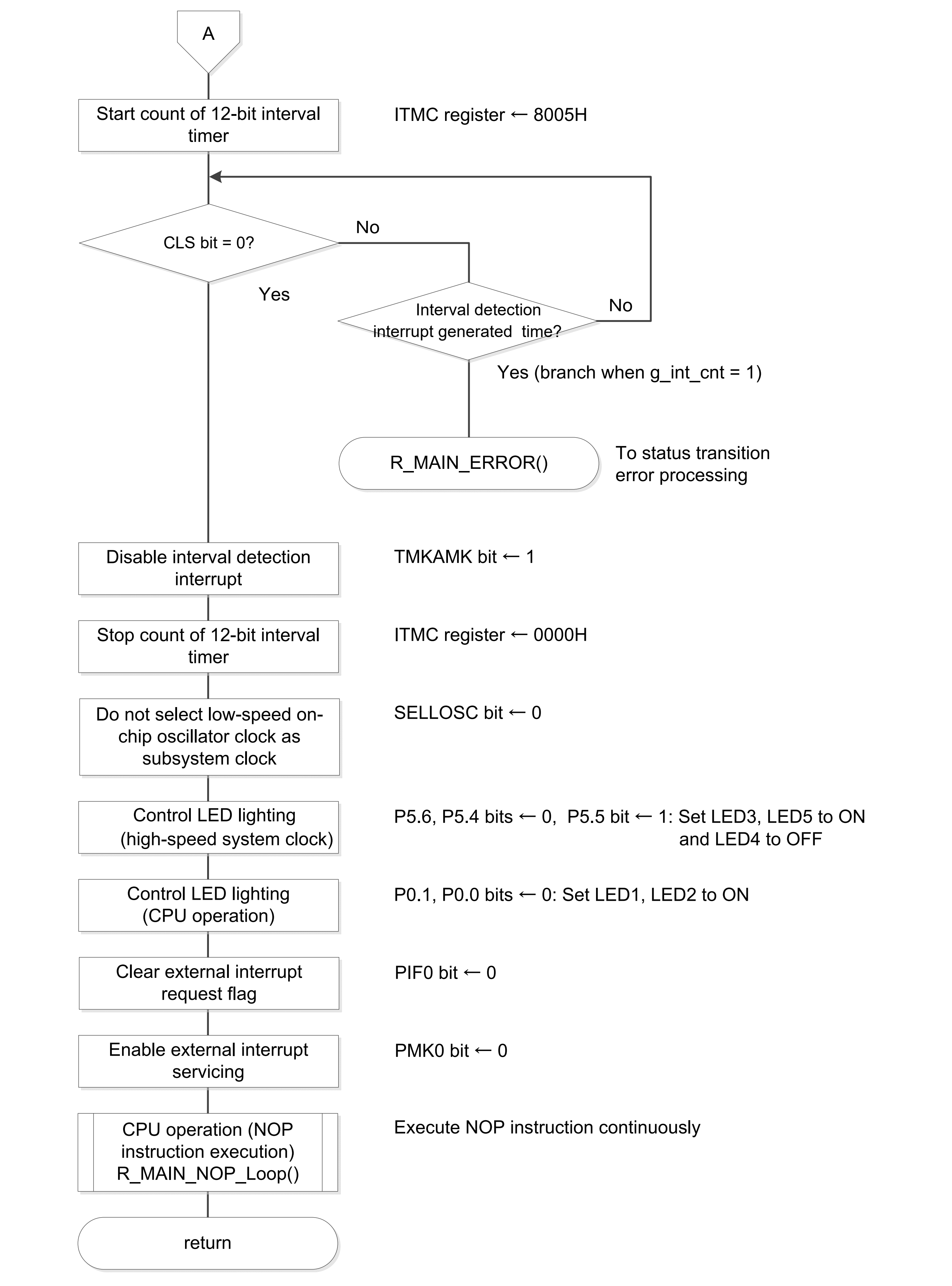
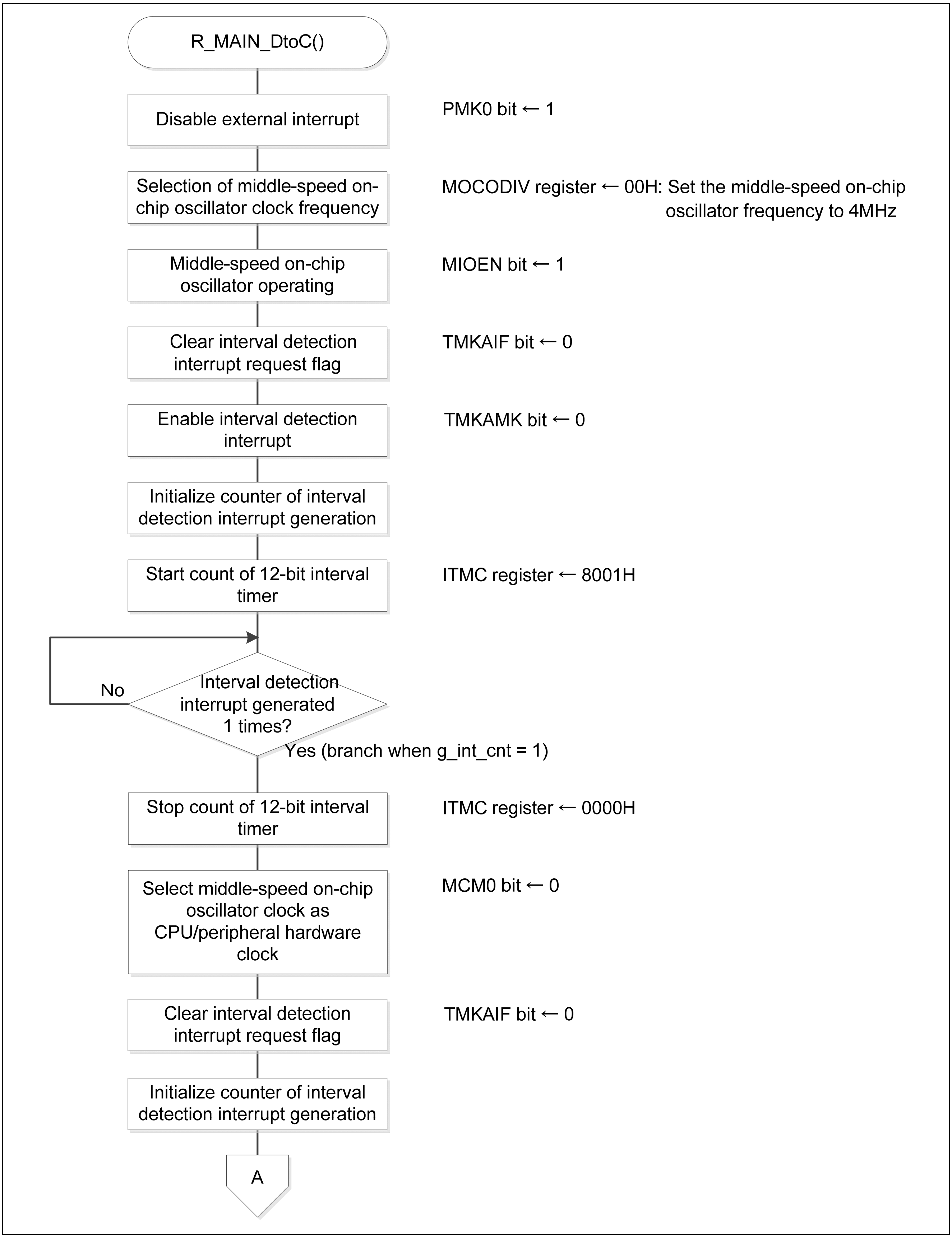

A
Start count of 12-bit interval timer
ITMC register ← 8005H
CLS bit = 0?
No
Yes
Interval detection interrupt generated time?
No
Yes (branch when g_int_cnt = 1)
R_MAIN_ERROR()
To status transition error processing
Disable interval detection interrupt
TMKAMK bit ← 1
Stop count of 12-bit interval timer
ITMC register ← 0000H
Do not select low-speed on-chip oscillator clock as subsystem clock
SELLOSC bit ← 0
Control LED lighting (high-speed system clock)
P5.6, P5.4 bits ← 0, P5.5 bit ← 1: Set LED3, LED5 to ON and LED4 to OFF
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously
return

Figure 5.38 Status Transition EtoD(2/2)

5.6.29 Status Transition DtoC

Figure 5.39 and Figure 5.40 shows the status transition DtoC.



R_MAIN_DtoC()
Disable external interrupt
PMK0 bit ← 1
Selection of middle-speed on-chip oscillator clock frequency
MOCODIV register ← 00H: Set the middle-speed on-chip oscillator frequency to 4MHz
Middle-speed on-chip oscillator operating
MIOEN bit ← 1
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Enable interval detection interrupt
TMKAMK bit ← 0
Initialize counter of interval detection interrupt generation
Start count of 12-bit interval timer
ITMC register ← 8001H
Interval detection interrupt generated 1 times?
No
Yes (branch when g_int_cnt = 1)
Stop count of 12-bit interval timer
ITMC register ← 0000H
Select middle-speed on-chip oscillator clock as CPU/peripheral hardware clock
MCM0 bit ← 0
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Initialize counter of interval detection interrupt generation
A

Figure 5.39 Status Transition DtoC(1/2)

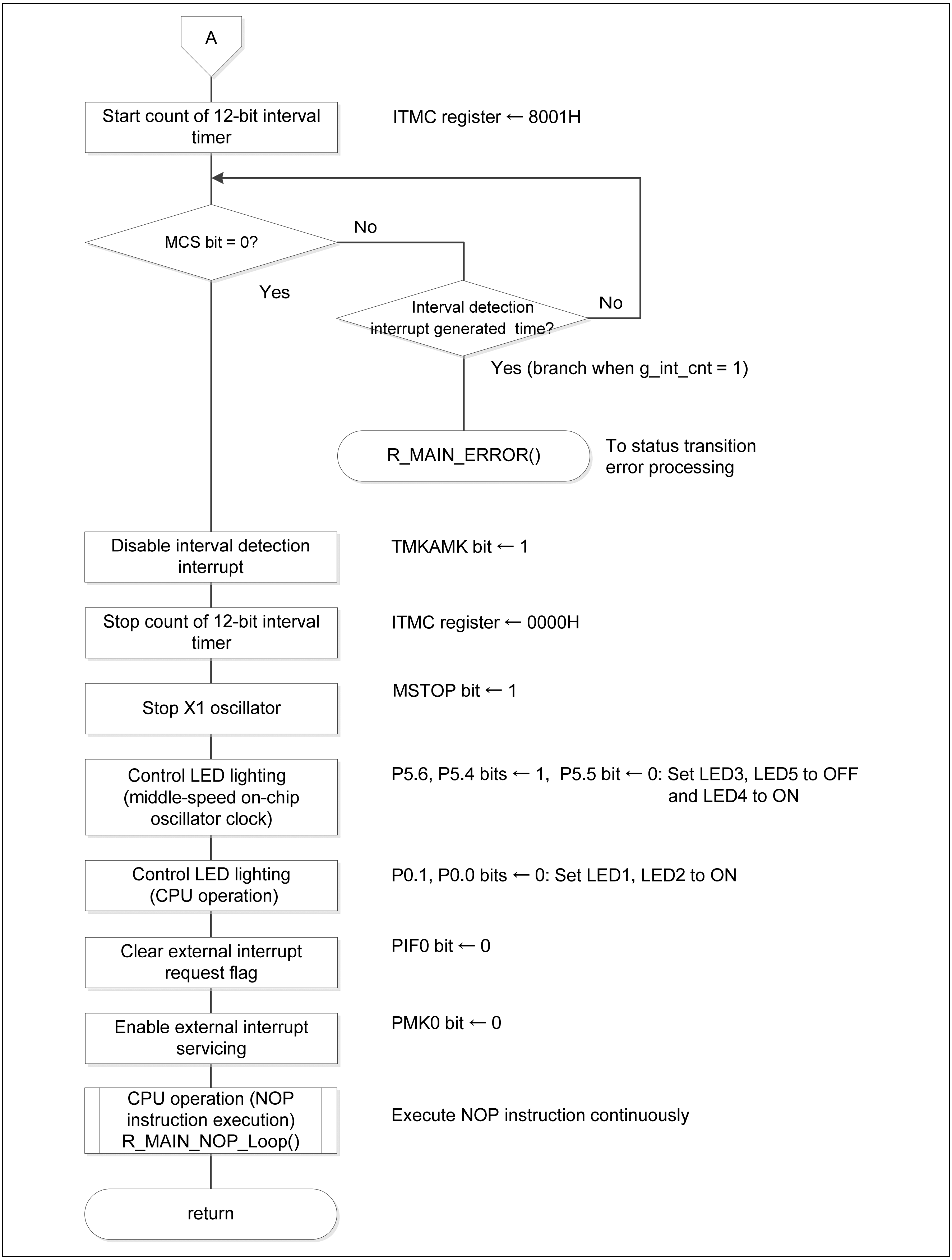

A
Start count of 12-bit interval timer
ITMC register ← 8001H
MCS bit = 0?
No
Yes
Interval detection interrupt generated time?
No
Yes (branch when g_int_cnt = 1)
R_MAIN_ERROR()
To status transition error processing
Disable interval detection interrupt
TMKAMK bit ← 1
Stop count of 12-bit interval timer
ITMC register ← 0000H
Stop X1 oscillator
MSTOP bit ← 1
Control LED lighting (middle-speed on-chip oscillator clock)
P5.6, P5.4 bits ← 1, P5.5 bit ← 0: Set LED3, LED5 to OFF and LED4 to ON
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously
return

Figure 5.40 Status Transition DtoC(2/2)

5.6.30 Status Transition CtoI

Figure 5.41 shows the status transition CtoI.

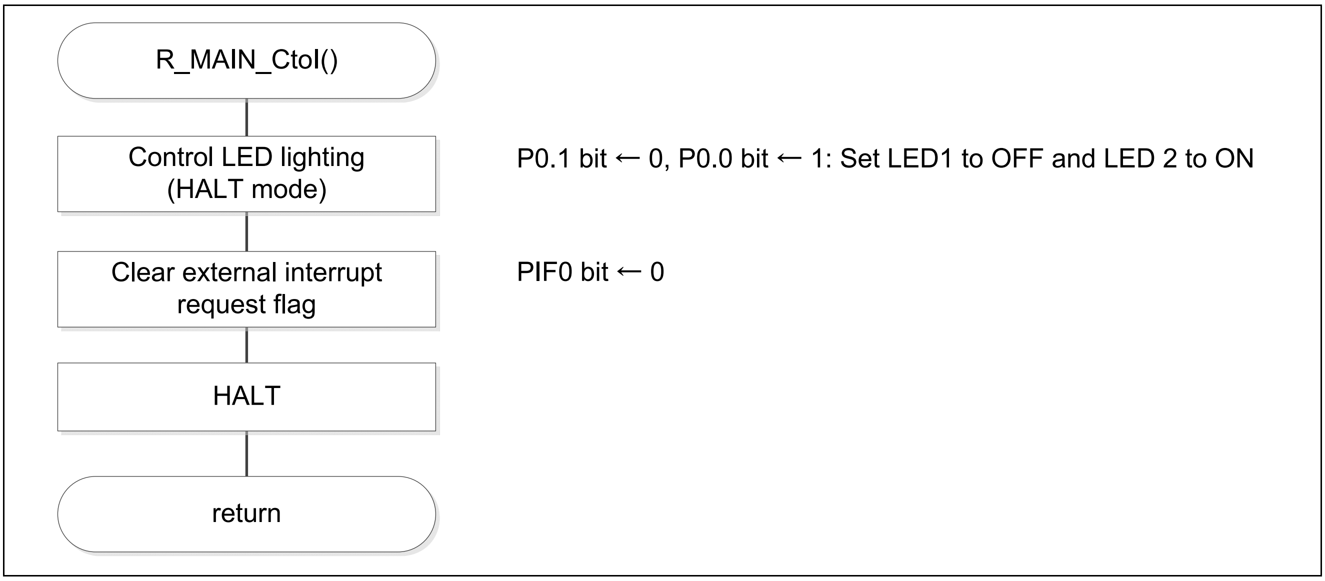

R_MAIN_CtoI()
Control LED lighting (HALT mode)
P0.1 bit ← 0, P0.0 bit ← 1: Set LED1 to OFF and LED 2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
HALT
return

Figure 5.41 Status Transition CtoI

5.6.31 Status Transition ItoC

Figure 5.42 shows the status transition ItoC.

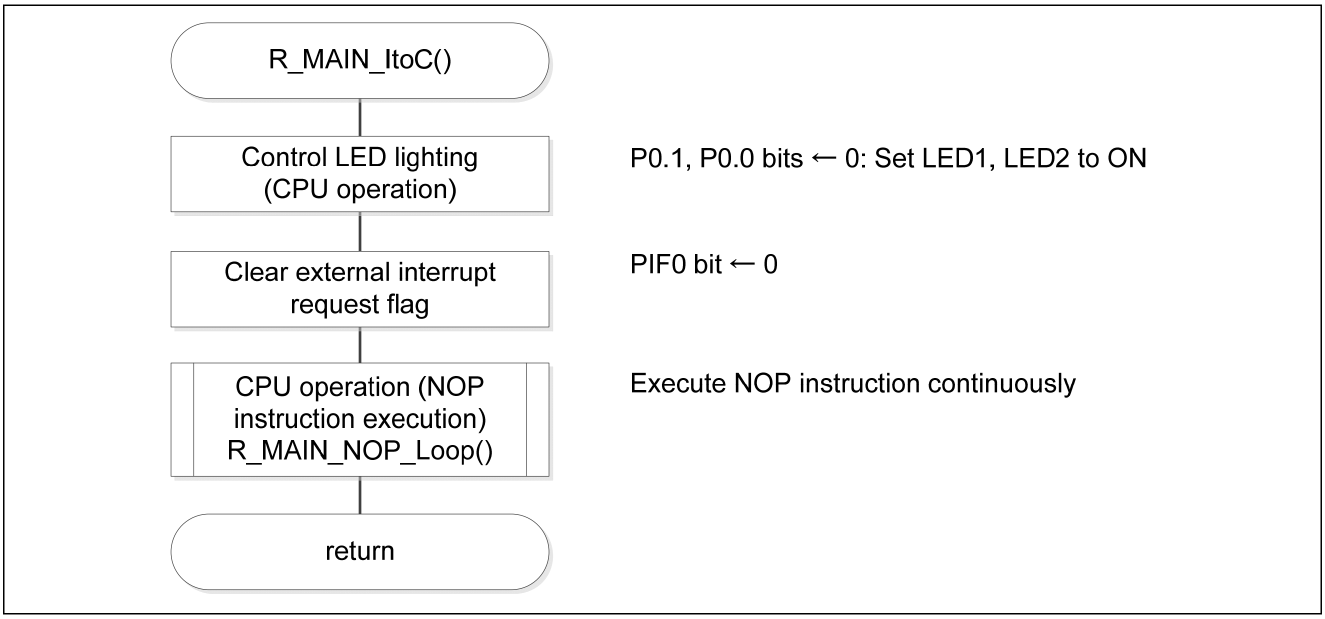

R_MAIN_ItoC()
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously
return

Figure 5.42 Status Transition ItoC

5.6.32 Status Transition CtoJ

Figure 5.43 shows the status transition CtoJ.

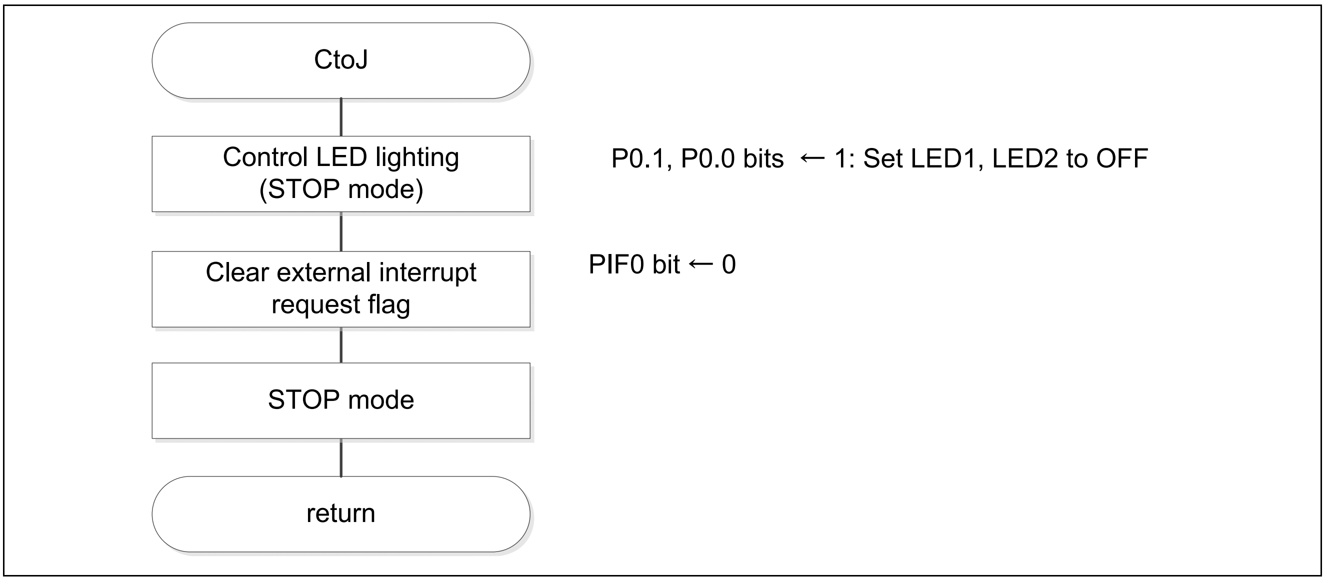

CtoJ
Control LED lighting (STOP mode)
P0.1, P0.0 bits ← 1: Set LED1, LED2 to OFF
Clear external interrupt request flag
PIF0 bit ← 0
STOP mode
return

Figure 5.43 Status Transition CtoJ

5.6.33 Status Transition JtoC

Figure 5.44 shows the status transition JtoC.

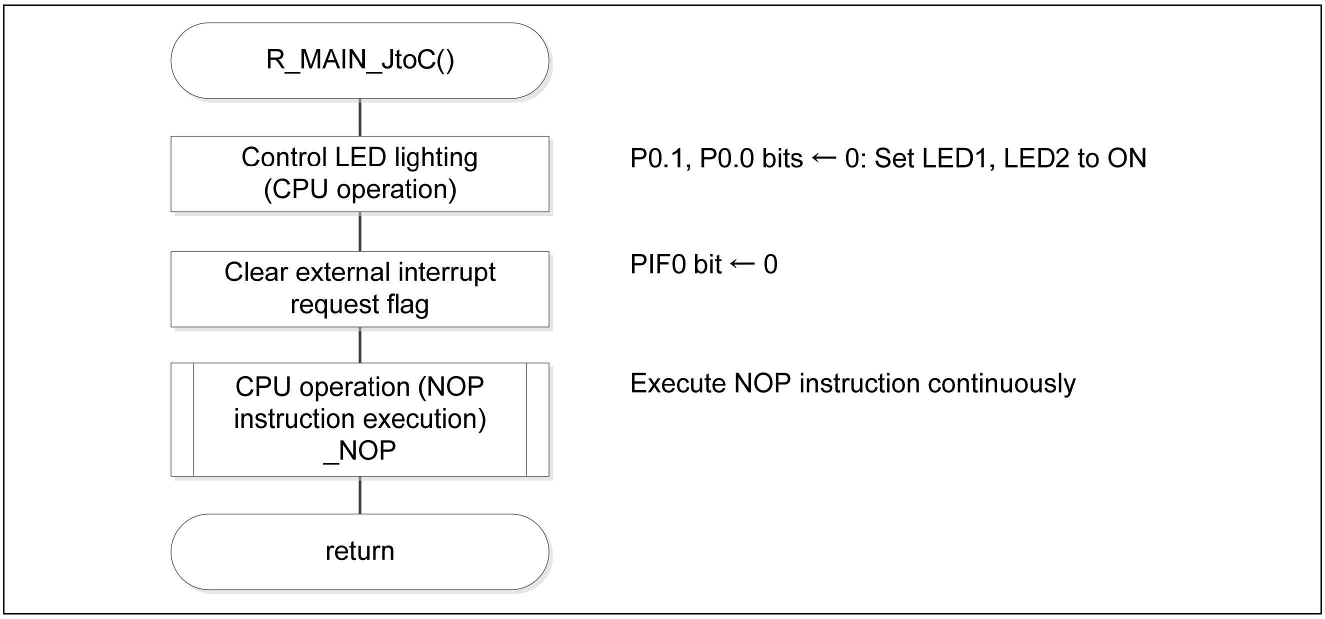

R_MAIN_JtoC()
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
CPU operation (NOP instruction execution) _NOP
Execute NOP instruction continuously
return

Figure 5.44 Status Transition JtoC

5.6.34 Status Transition CtoK

Figure 5.45 shows the status transition CtoK.

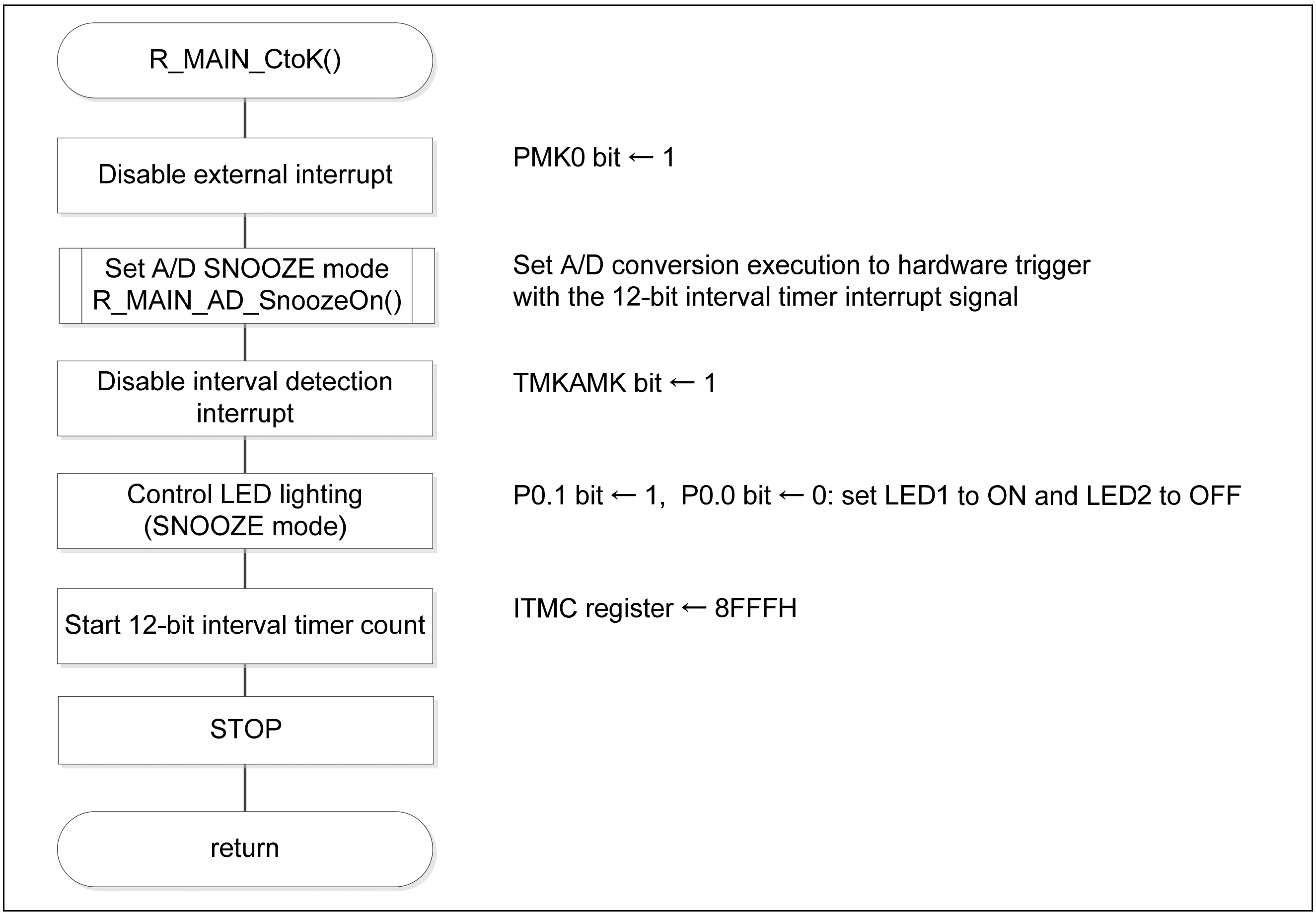

R_MAIN_CtoK()
Disable external interrupt
PMK0 bit ← 1
Set A/D SNOOZE mode
R_MAIN_AD_SnoozeOn()
Set A/D conversion execution to hardware trigger
with the 12-bit interval timer interrupt signal
Disable interval detection
interrupt
TMKAMK bit ← 1
Control LED lighting
(SNOOZE mode)
P0.1 bit ← 1, P0.0 bit ← 0: set LED1 to ON and LED2 to OFF
Start 12-bit interval timer count
ITMC register ← 8FFFH
STOP
return

Figure 5.45 Status Transition CtoK

5.6.35 Status Transition KtoC

Figure5. 46 shows the status transition KtoC.

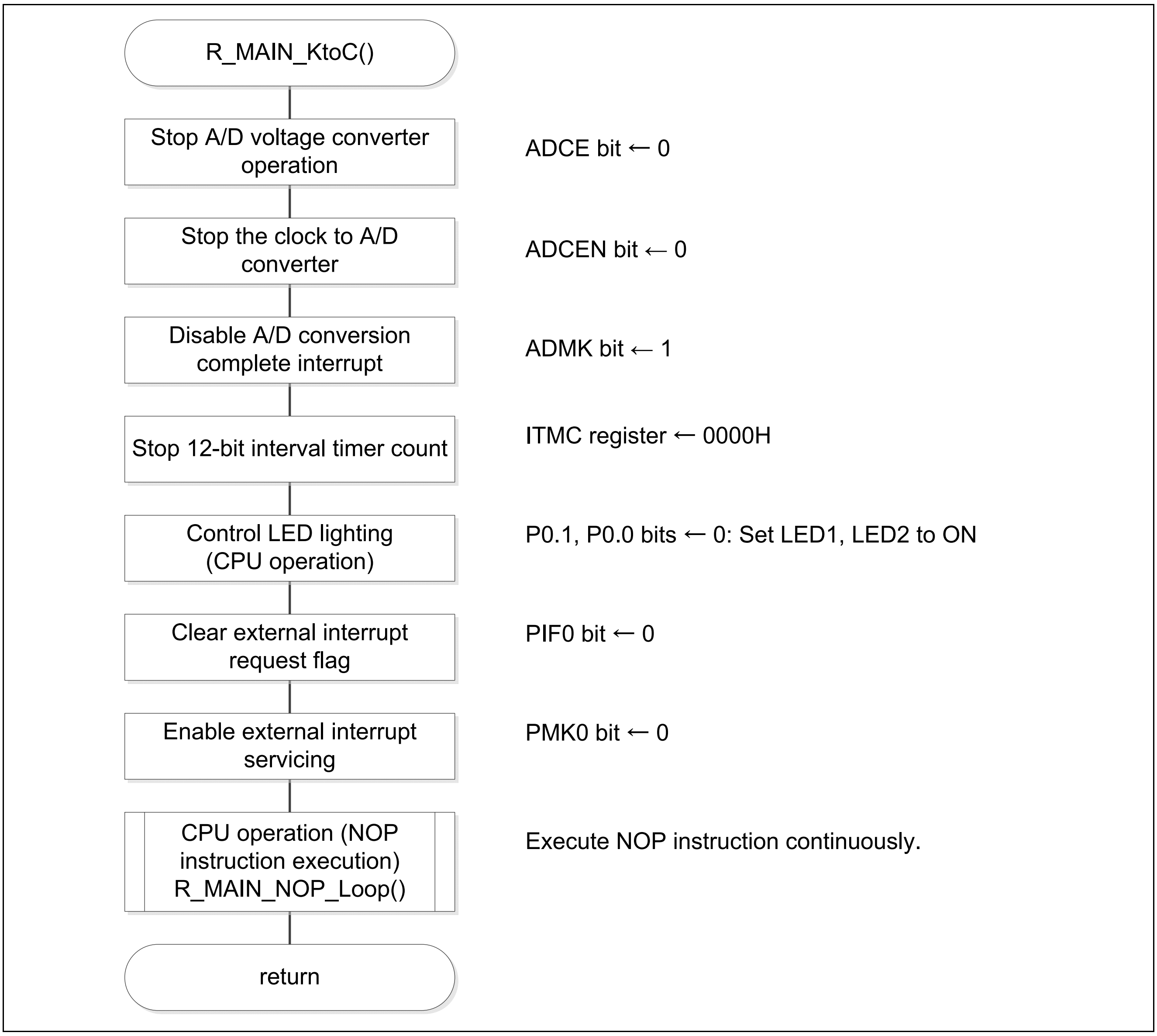

R_MAIN_KtoC()
Stop A/D voltage converter operation
ADCE bit ← 0
Stop the clock to A/D converter
ADCEN bit ← 0
Disable A/D conversion complete interrupt
ADMK bit ← 1
Stop 12-bit interval timer count
ITMC register ← 0000H
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously.
return

Figure 5.46 Status Transition KtoC

5.6.36 Status Transition CtoE

Figure 5.47 and Figure 5.48 shows the status transition CtoE.

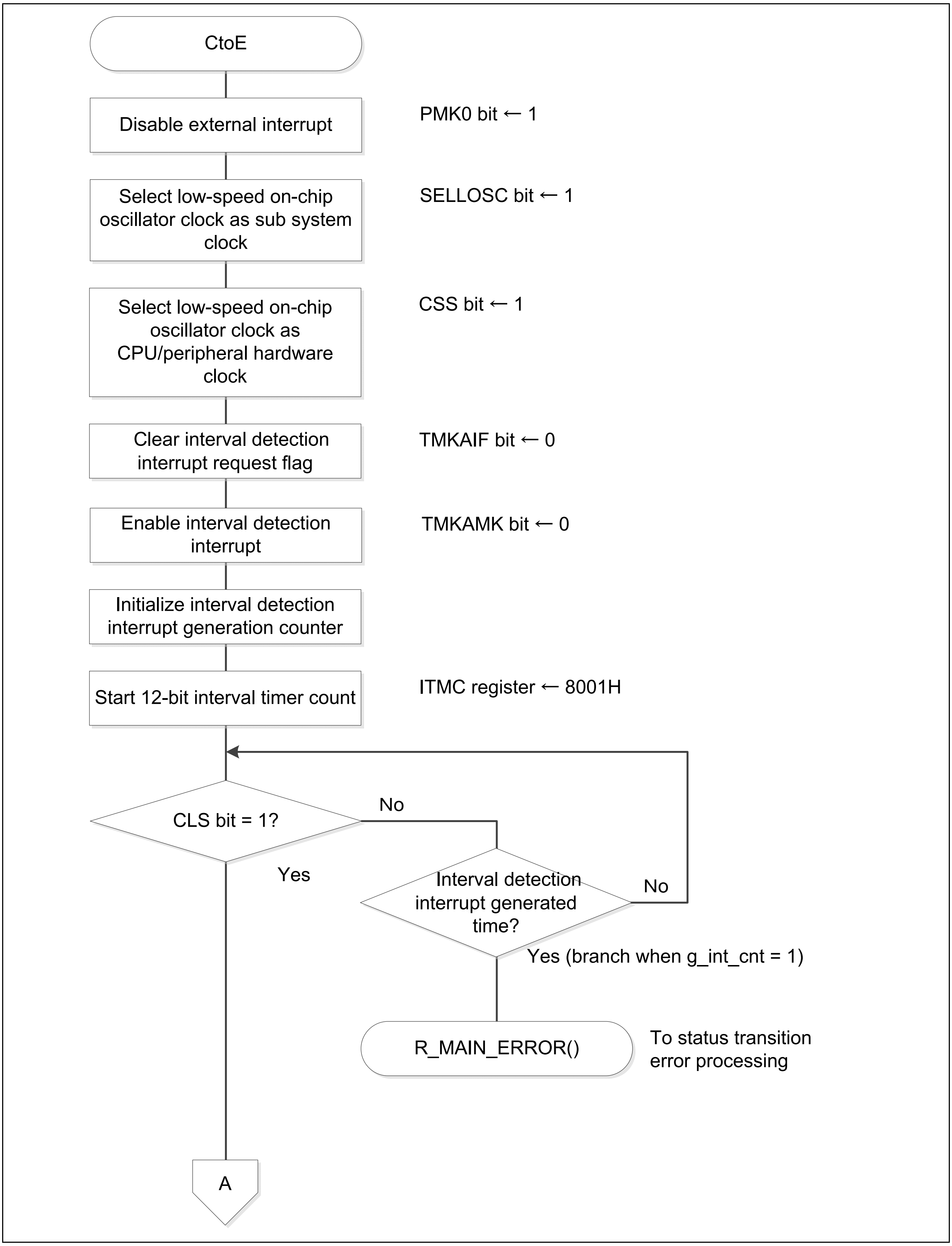

CtoE
Disable external interrupt
PMK0 bit ← 1
Select low-speed on-chip oscillator clock as sub system clock
SELLOSC bit ← 1
Select low-speed on-chip oscillator clock as CPU/peripheral hardware clock
CSS bit ← 1
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Enable interval detection interrupt
TMKAMK bit ← 0
Initialize interval detection interrupt generation counter
Start 12-bit interval timer count
ITMC register ← 8001H
CLS bit = 1?
No
Yes
Interval detection interrupt generated time?
No
Yes (branch when g_int_cnt = 1)
R_MAIN_ERROR()
To status transition error processing
A

Figure 5.47 Status Transition CtoE(1/2)

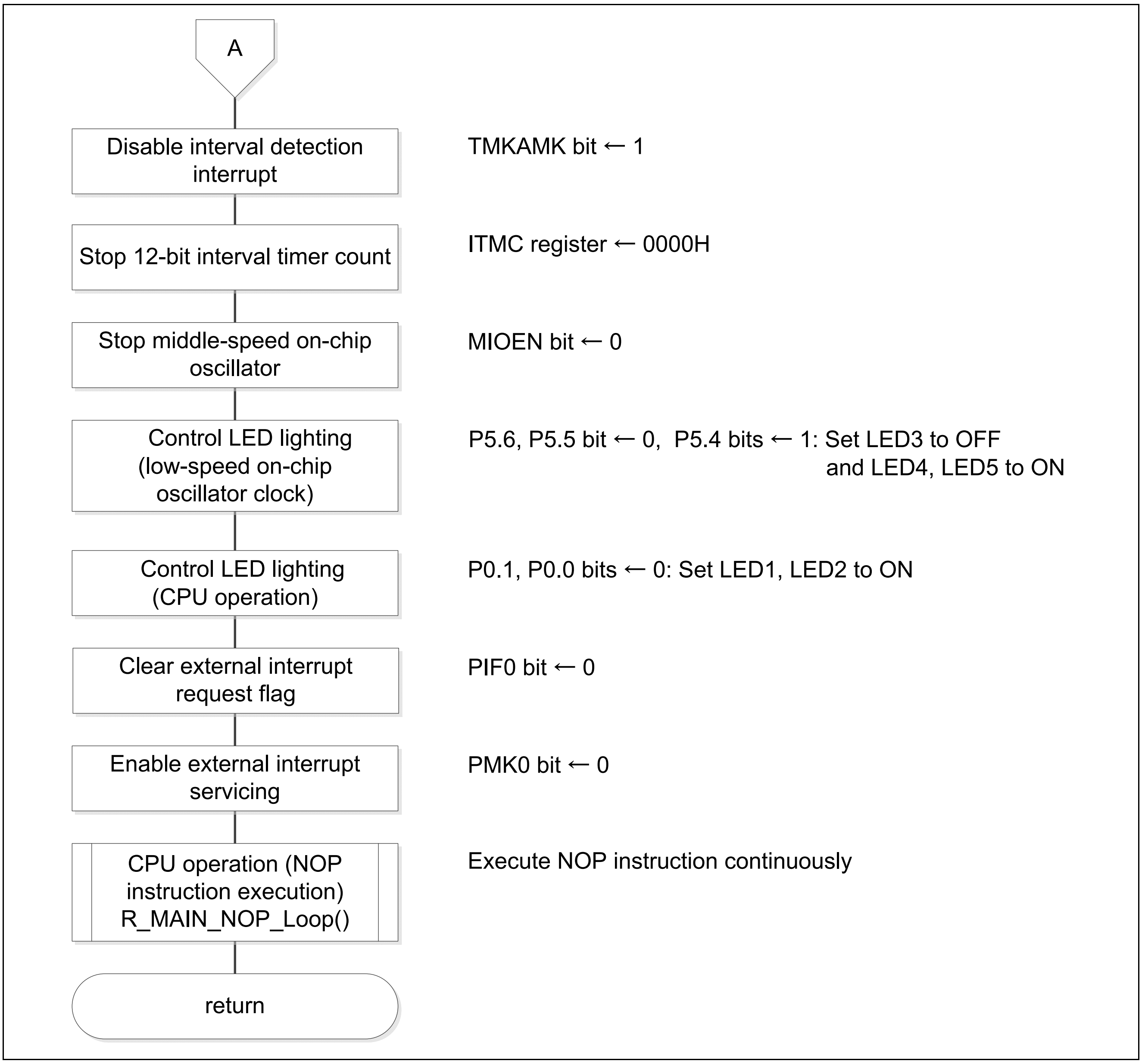

A
Disable interval detection interrupt
TMKAMK bit ← 1
Stop 12-bit interval timer count
ITMC register ← 0000H
Stop middle-speed on-chip oscillator
MIOEN bit ← 0
Control LED lighting (low-speed on-chip oscillator clock)
P5.6, P5.5 bit ← 0, P5.4 bits ← 1: Set LED3 to OFF and LED4, LED5 to ON
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously
return

Figure 5.48 Status Transition CtoE(2/2)

5.6.37 Status Transition EtoC

Figure 5.49 and Figure 5.50 shows the status transition EtoC.

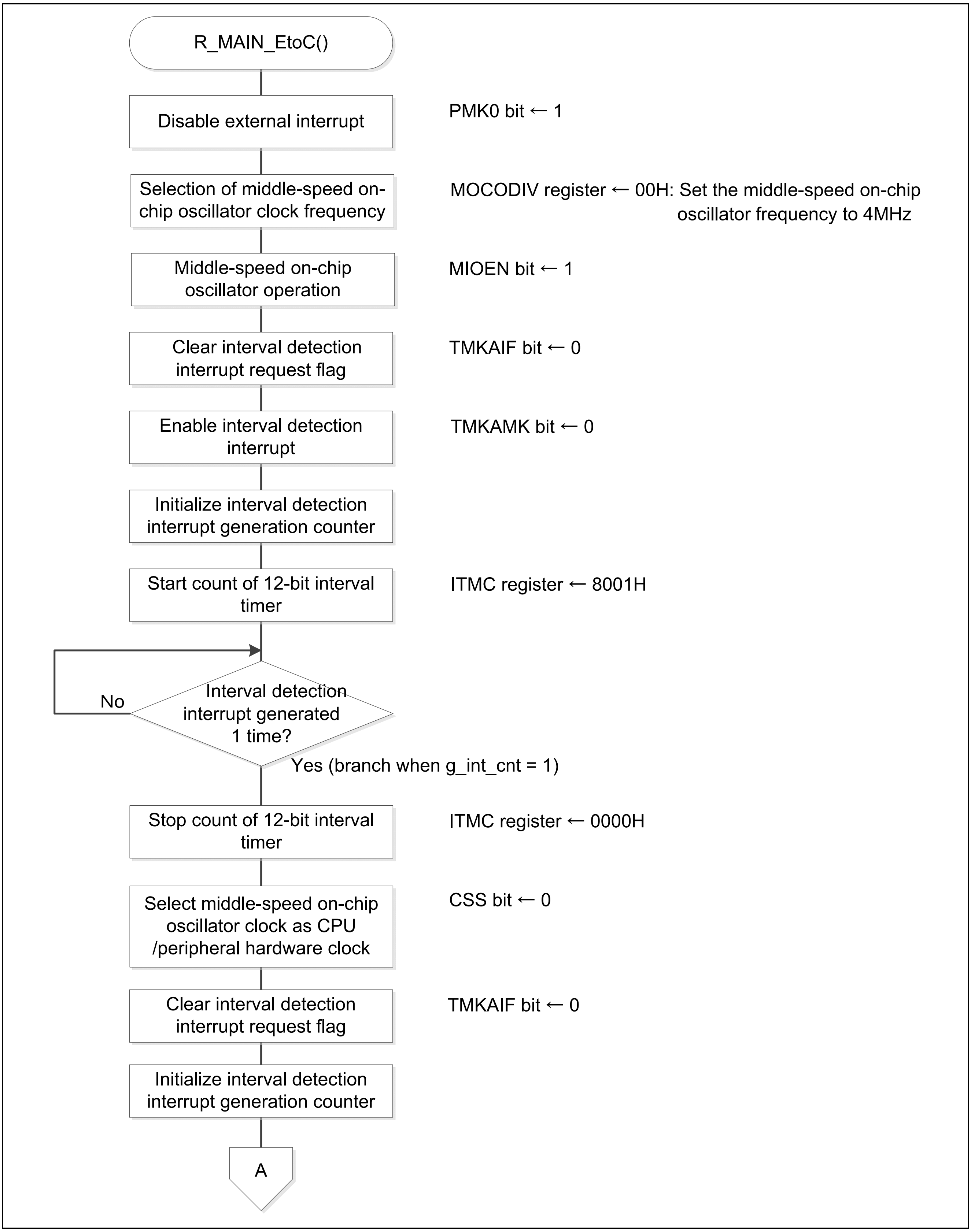

R_MAIN_EtoC()
Disable external interrupt
PMK0 bit ← 1
Selection of middle-speed on-chip oscillator clock frequency
MOCODIV register ← 00H: Set the middle-speed on-chip oscillator frequency to 4MHz
Middle-speed on-chip oscillator operation
MIOEN bit ← 1
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Enable interval detection interrupt
TMKAMK bit ← 0
Initialize interval detection interrupt generation counter
Start count of 12-bit interval timer
ITMC register ← 8001H
Interval detection interrupt generated 1 time?
No
Yes (branch when g_int_cnt = 1)
Stop count of 12-bit interval timer
ITMC register ← 0000H
Select middle-speed on-chip oscillator clock as CPU /peripheral hardware clock
CSS bit ← 0
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Initialize interval detection interrupt generation counter
A

Figure 5.49 Status Transition EtoC(1/2)

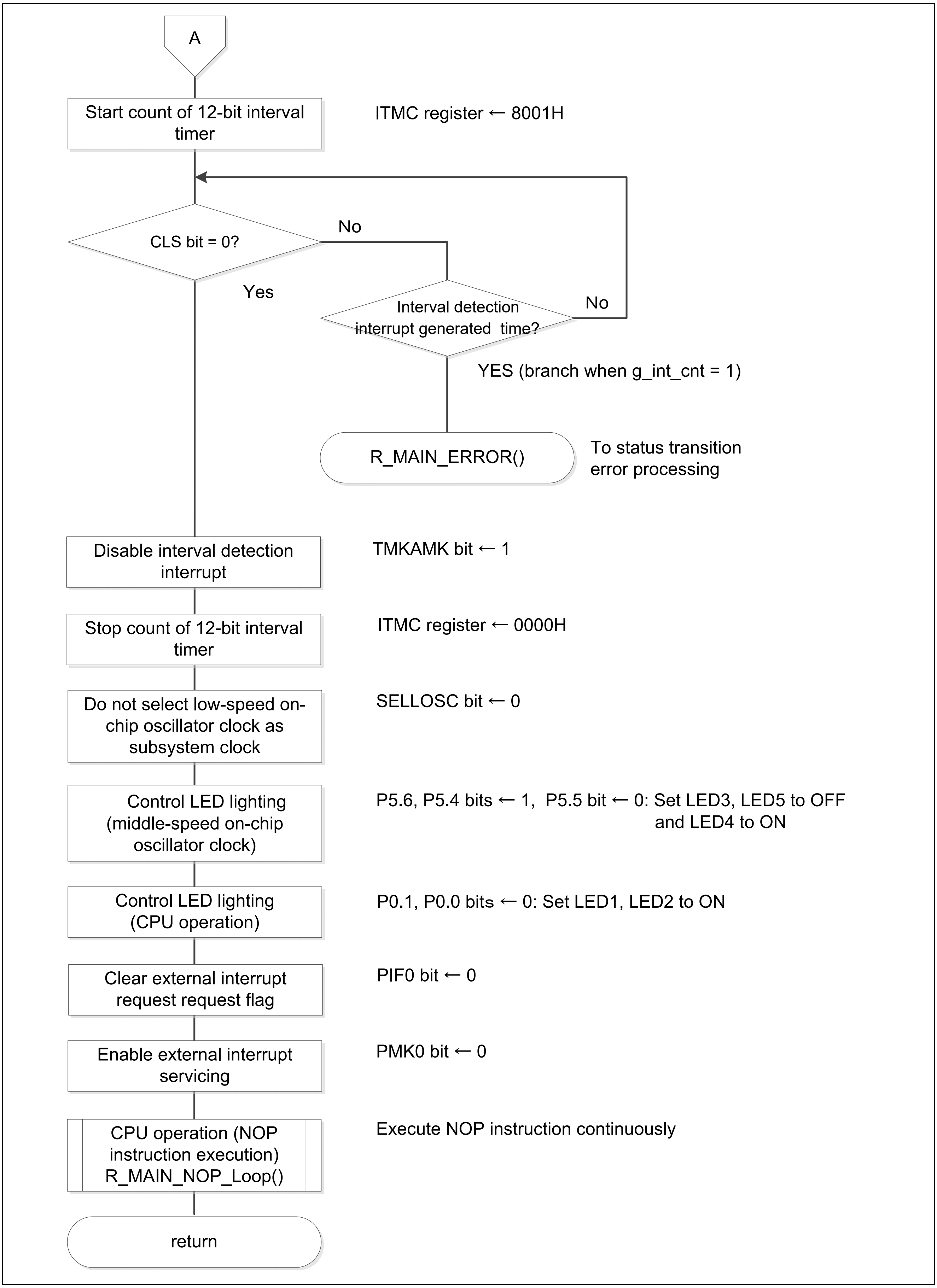

A
Start count of 12-bit interval timer
ITMC register ← 8001H
CLS bit = 0?
No
Yes
Interval detection interrupt generated time?
No
YES (branch when g_int_cnt = 1)
R_MAIN_ERROR()
To status transition error processing
Disable interval detection interrupt
TMKAMK bit ← 1
Stop count of 12-bit interval timer
ITMC register ← 0000H
Do not select low-speed on-chip oscillator clock as subsystem clock
SELLOSC bit ← 0
Control LED lighting (middle-speed on-chip oscillator clock)
P5.6, P5.4 bits ← 1, P5.5 bit ← 0: Set LED3, LED5 to OFF and LED4 to ON
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously
return

Figure 5.50 Status Transition EtoC(2/2)

5.6.38 Status Transition CtoB

Figure 5.51 and Figure 5.52 shows the status transition CtoB.

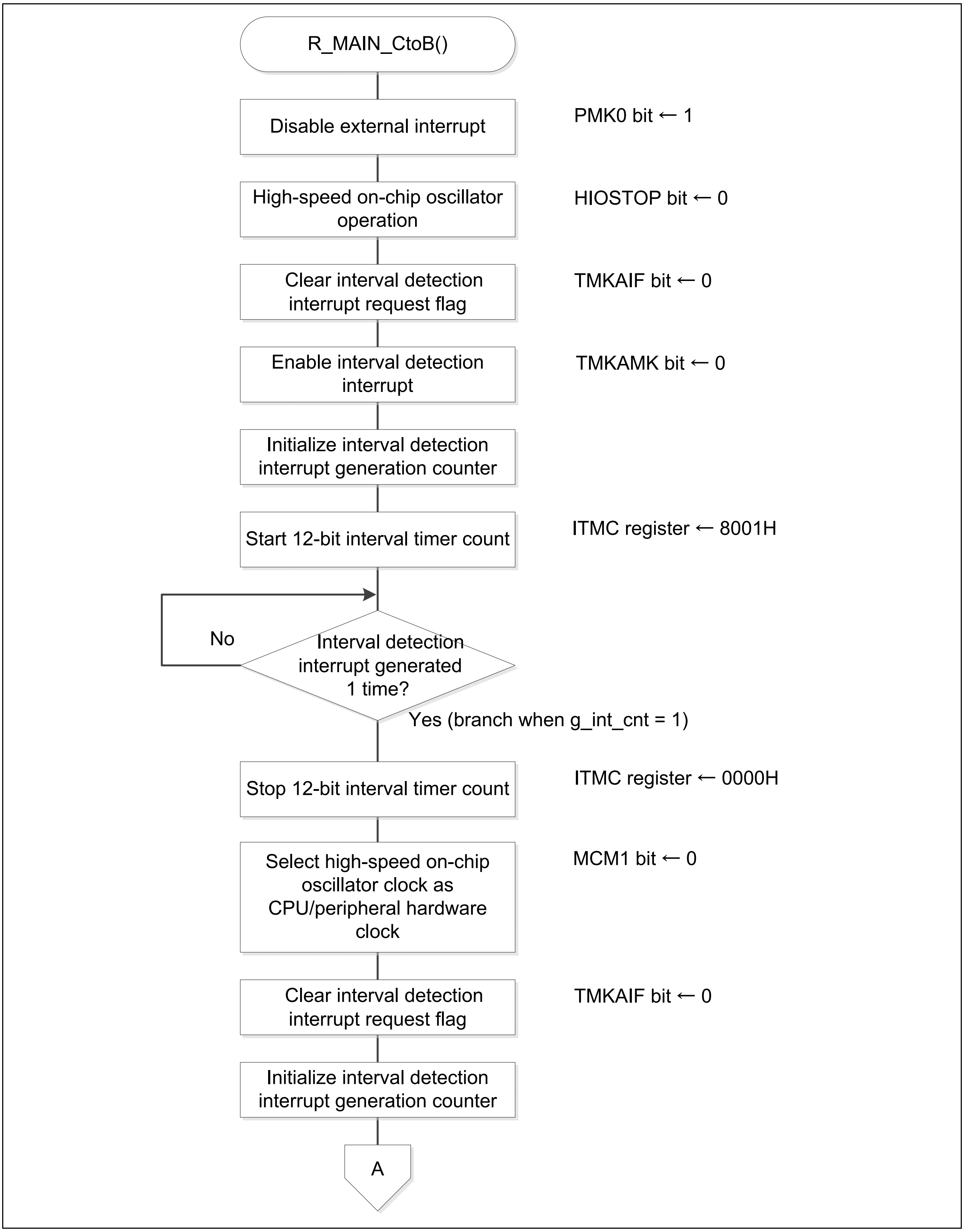

R_MAIN_CtoB()
Disable external interrupt
PMK0 bit ← 1
High-speed on-chip oscillator operation
HIOSTOP bit ← 0
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Enable interval detection interrupt
TMKAMK bit ← 0
Initialize interval detection interrupt generation counter
Start 12-bit interval timer count
ITMC register ← 8001H
No
Interval detection interrupt generated 1 time?
Yes (branch when g_int_cnt = 1)
Stop 12-bit interval timer count
ITMC register ← 0000H
Select high-speed on-chip oscillator clock as CPU/peripheral hardware clock
MCM1 bit ← 0
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Initialize interval detection interrupt generation counter
A

Figure 5.51 Status Transition CtoB(1/2)

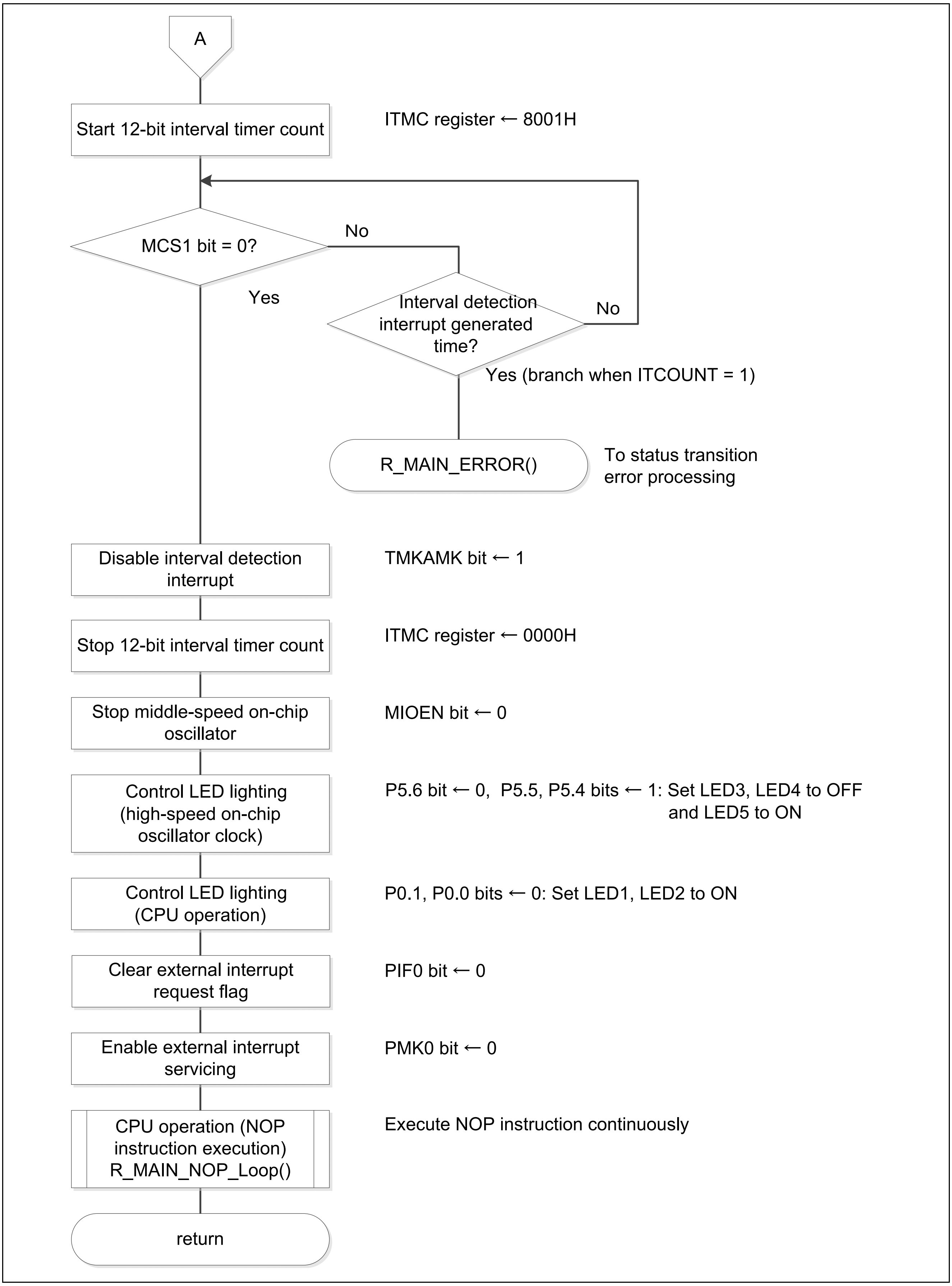

A
Start 12-bit interval timer count
ITMC register ← 8001H
MCS1 bit = 0?
No
Yes
Interval detection interrupt generated time?
No
Yes (branch when ITCOUNT = 1)
R_MAIN_ERROR()
To status transition error processing
Disable interval detection interrupt
TMKAMK bit ← 1
Stop 12-bit interval timer count
ITMC register ← 0000H
Stop middle-speed on-chip oscillator
MIOEN bit ← 0
Control LED lighting (high-speed on-chip oscillator clock)
P5.6 bit ← 0, P5.5, P5.4 bits ← 1: Set LED3, LED4 to OFF and LED5 to ON
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously
return

Figure 5.52 Status Transition CtoB(2/2)

5.6.39 Status Transition BtoE

Figure 5.53 and Figure 5.54 shows the status transition BtoE.

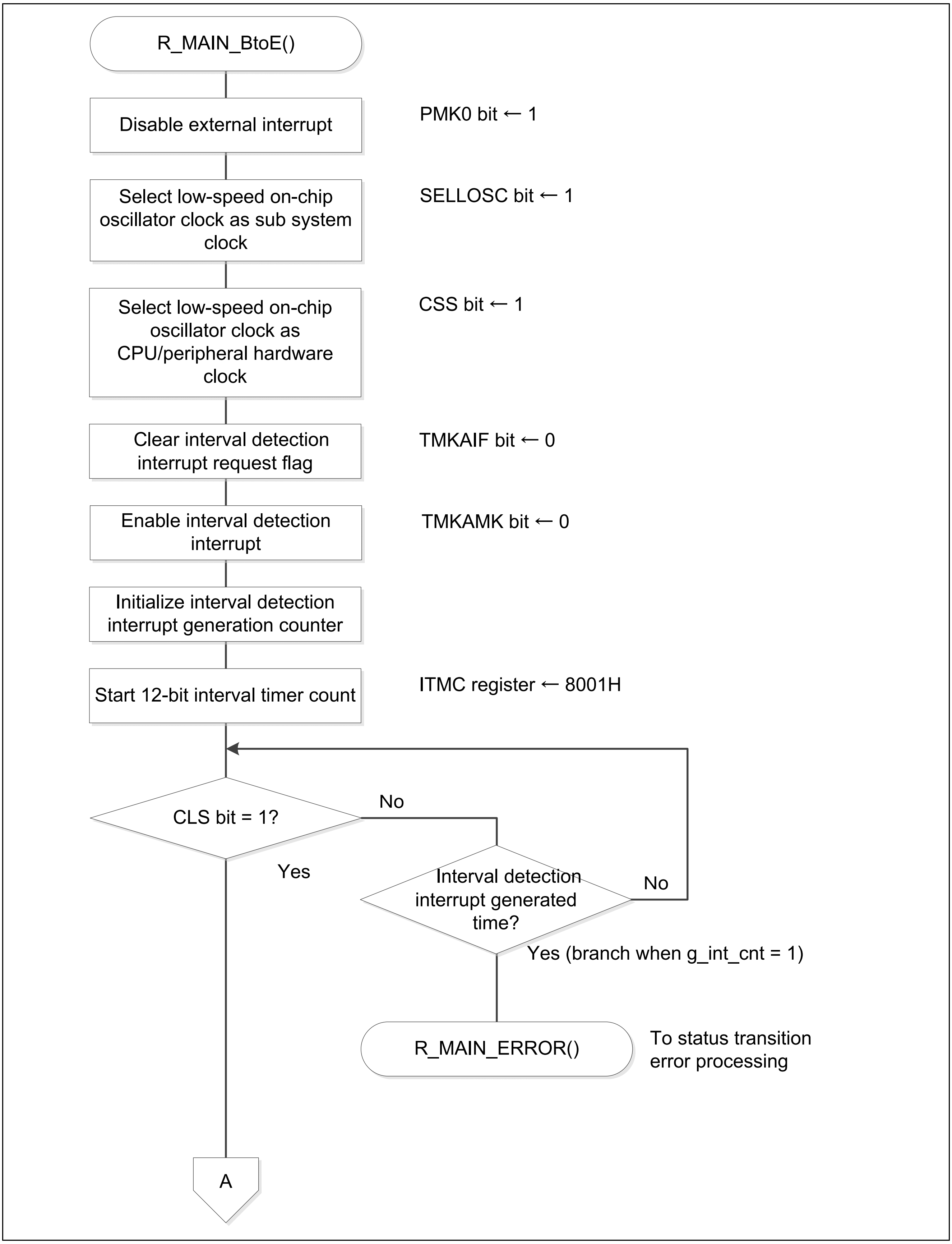

R_MAIN_BtoE()
Disable external interrupt
PMK0 bit ← 1
Select low-speed on-chip oscillator clock as sub system clock
SELLOSC bit ← 1
Select low-speed on-chip oscillator clock as CPU/peripheral hardware clock
CSS bit ← 1
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Enable interval detection interrupt
TMKAMK bit ← 0
Initialize interval detection interrupt generation counter
Start 12-bit interval timer count
ITMC register ← 8001H
CLS bit = 1?
No
Yes
Interval detection interrupt generated time?
No
Yes (branch when g_int_cnt = 1)
R_MAIN_ERROR()
To status transition error processing
A

Figure 5.37 Status Transition BtoE(1/2)

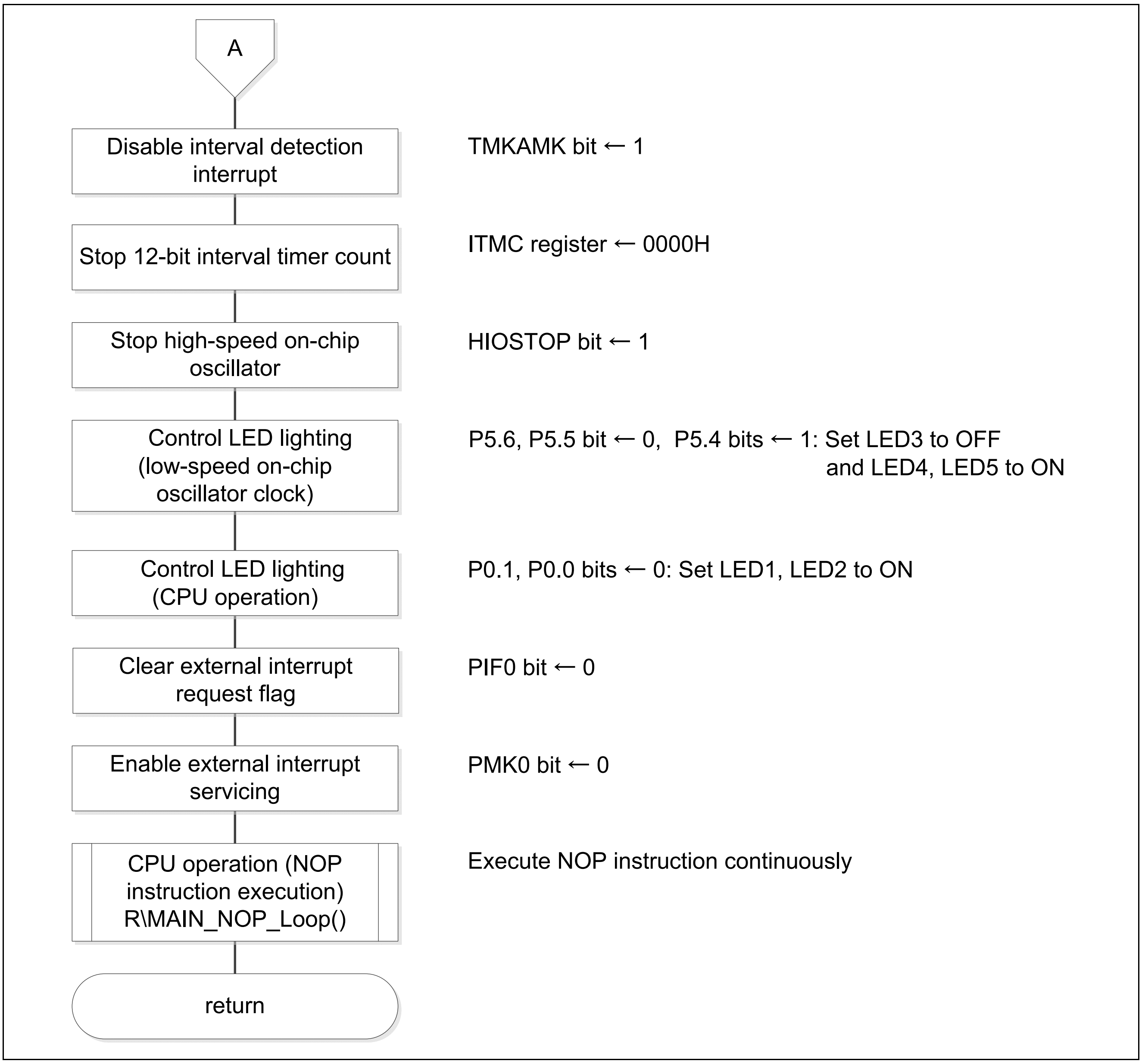

A
Disable interval detection interrupt
TMKAMK bit ← 1
Stop 12-bit interval timer count
ITMC register ← 0000H
Stop high-speed on-chip oscillator
HIOSTOP bit ← 1
Control LED lighting (low-speed on-chip oscillator clock)
P5.6, P5.5 bit ← 0, P5.4 bits ← 1: Set LED3 to OFF and LED4, LED5 to ON
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) R\MAIN_NOP_Loop()
Execute NOP instruction continuously
return

Figure 5.37 Status Transition BtoE(2/2)

5.6.40 Status Transition EtoN

Figure 5.55 shows the status transition EtoN.

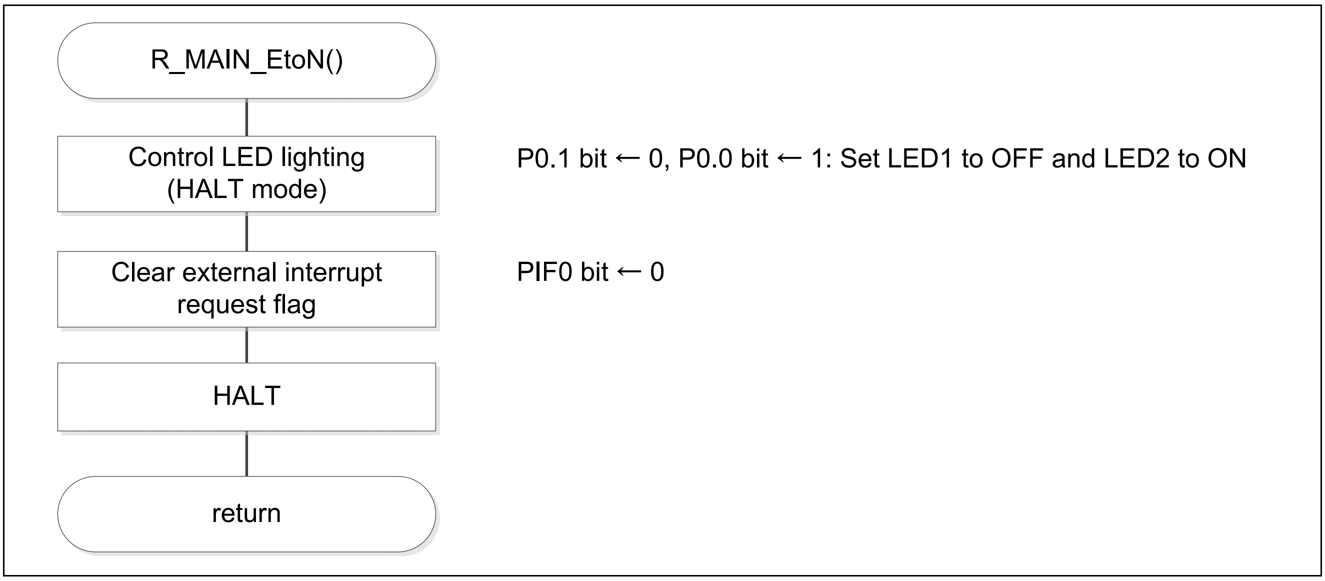

R_MAIN_EtoN()
Control LED lighting (HALT mode)
P0.1 bit ← 0, P0.0 bit ← 1: Set LED1 to OFF and LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
HALT
return

Figure 5.55 Status Transition EtoN

5.6.41 Status Transition NtoE

Figure5.56 shows the status transition NtoE.

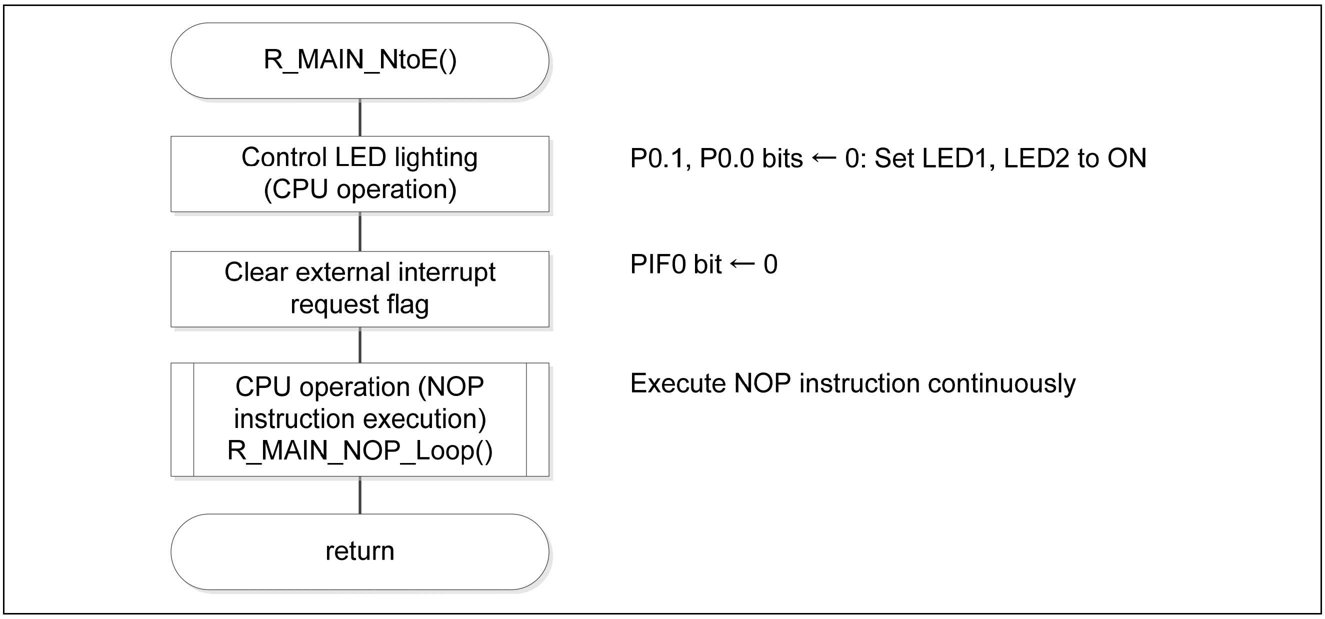

R_MAIN_NtoE()
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request flag
PIF0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously
return

Figure 5.56 Status Transition NtoE

5.6.42 Status Transition EtoB

Figure 5.57 and Figure 5.58 shows the status transition EtoB.

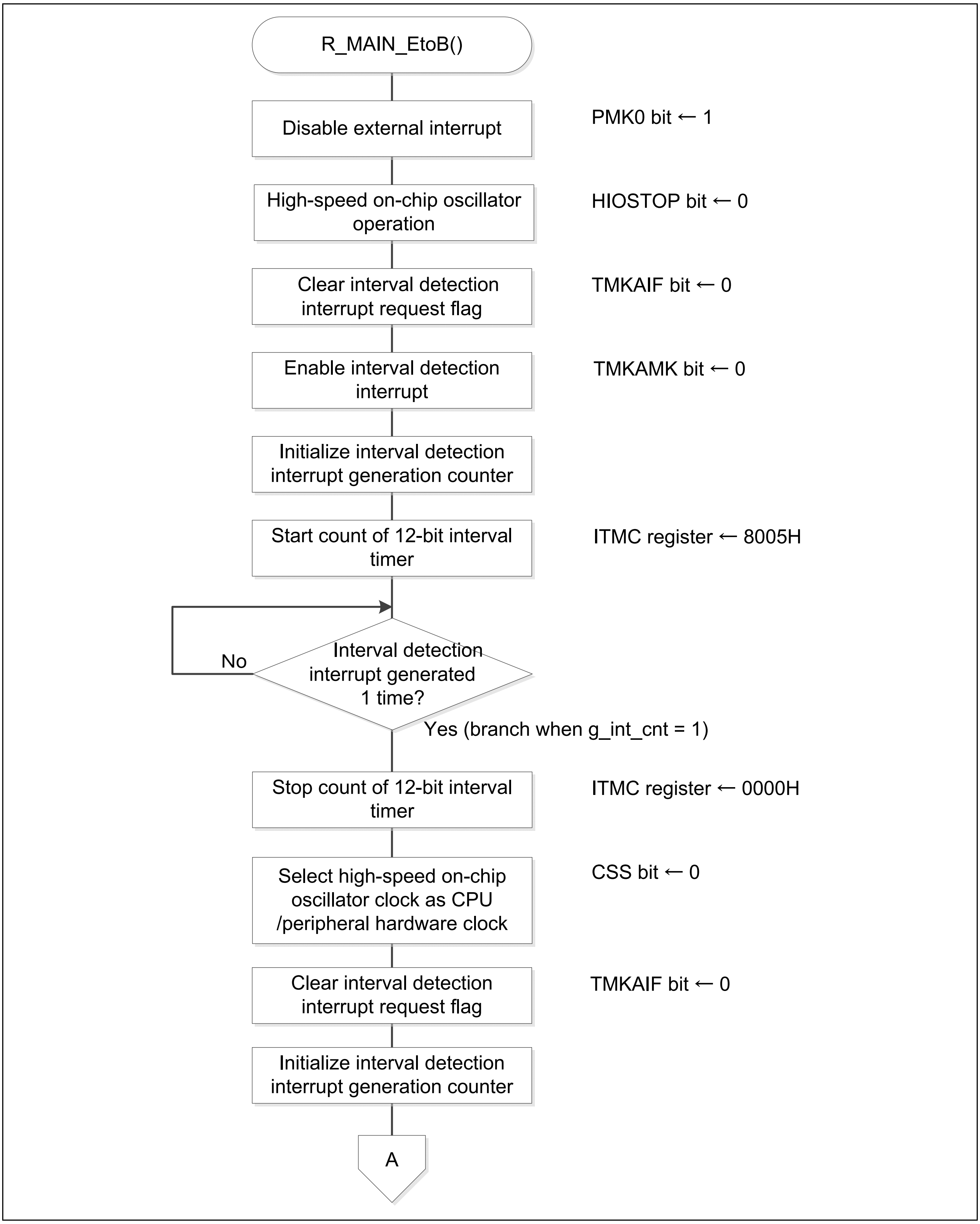

R_MAIN_EtoB()
Disable external interrupt
PMK0 bit ← 1
High-speed on-chip oscillator operation
HIOSTOP bit ← 0
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Enable interval detection interrupt
TMKAMK bit ← 0
Initialize interval detection interrupt generation counter
Start count of 12-bit interval timer
ITMC register ← 8005H
Interval detection interrupt generated 1 time?
No
Yes (branch when g_int_cnt = 1)
Stop count of 12-bit interval timer
ITMC register ← 0000H
Select high-speed on-chip oscillator clock as CPU /peripheral hardware clock
CSS bit ← 0
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Initialize interval detection interrupt generation counter
A

Figure 5.57 Status Transition EtoB(1/2)

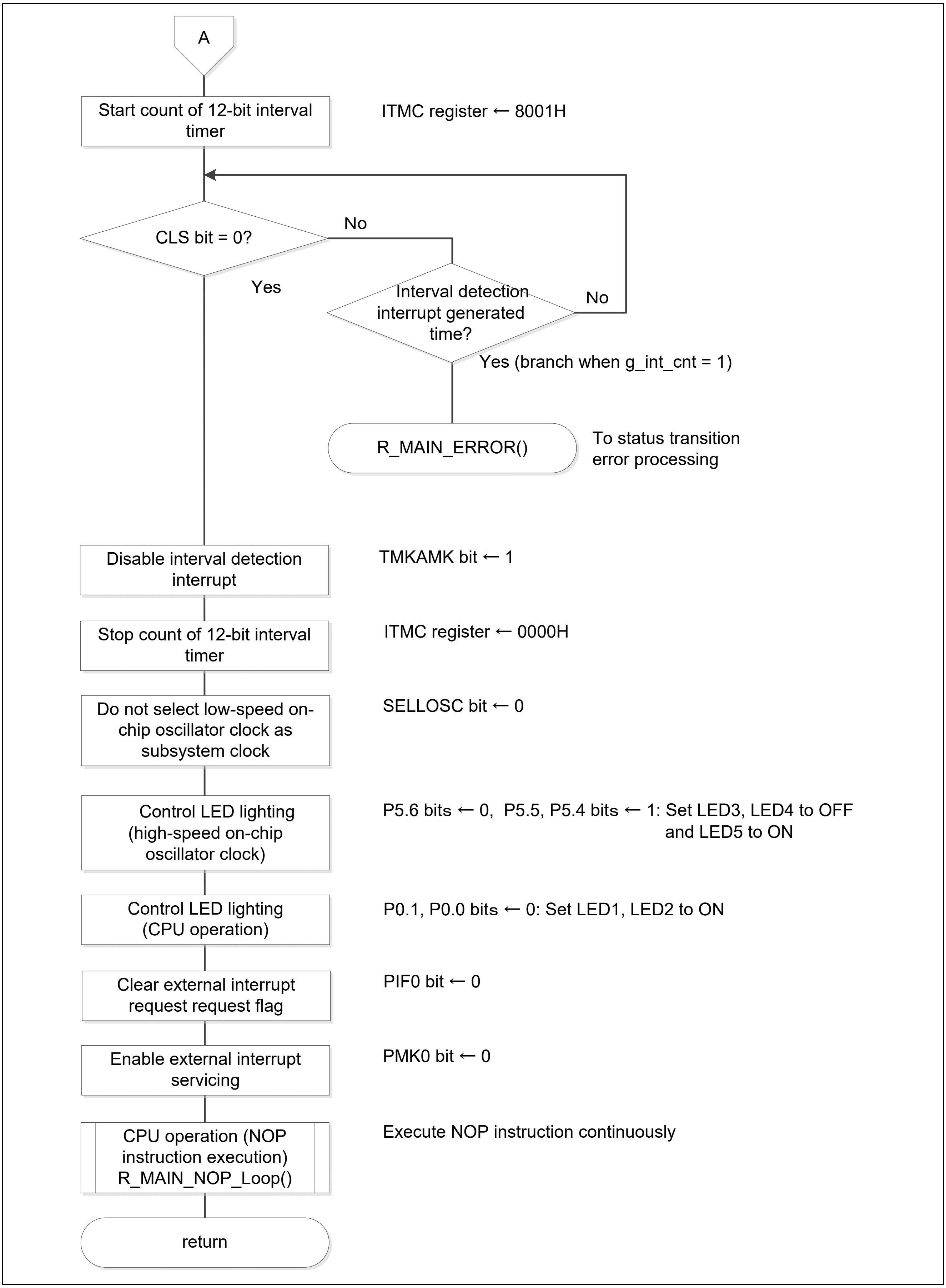

A
Start count of 12-bit interval timer
ITMC register ← 8001H
CLS bit = 0?
No
Yes
Interval detection interrupt generated time?
No
Yes (branch when g_int_cnt = 1)
R_MAIN_ERROR()
To status transition error processing
Disable interval detection interrupt
TMKAMK bit ← 1
Stop count of 12-bit interval timer
ITMC register ← 0000H
Do not select low-speed on-chip oscillator clock as subsystem clock
SELLOSC bit ← 0
Control LED lighting (high-speed on-chip oscillator clock)
P5.6 bits ← 0, P5.5, P5.4 bits ← 1: Set LED3, LED4 to OFF and LED5 to ON
Control LED lighting (CPU operation)
P0.1, P0.0 bits ← 0: Set LED1, LED2 to ON
Clear external interrupt request request flag
PIF0 bit ← 0
Enable external interrupt servicing
PMK0 bit ← 0
CPU operation (NOP instruction execution) R_MAIN_NOP_Loop()
Execute NOP instruction continuously
return

Figure 5.58 Status Transition EtoB(2/2)

5.6.43 Status Transition End Processing

Figure 5.59 shows the flowchart for status transition end processing.

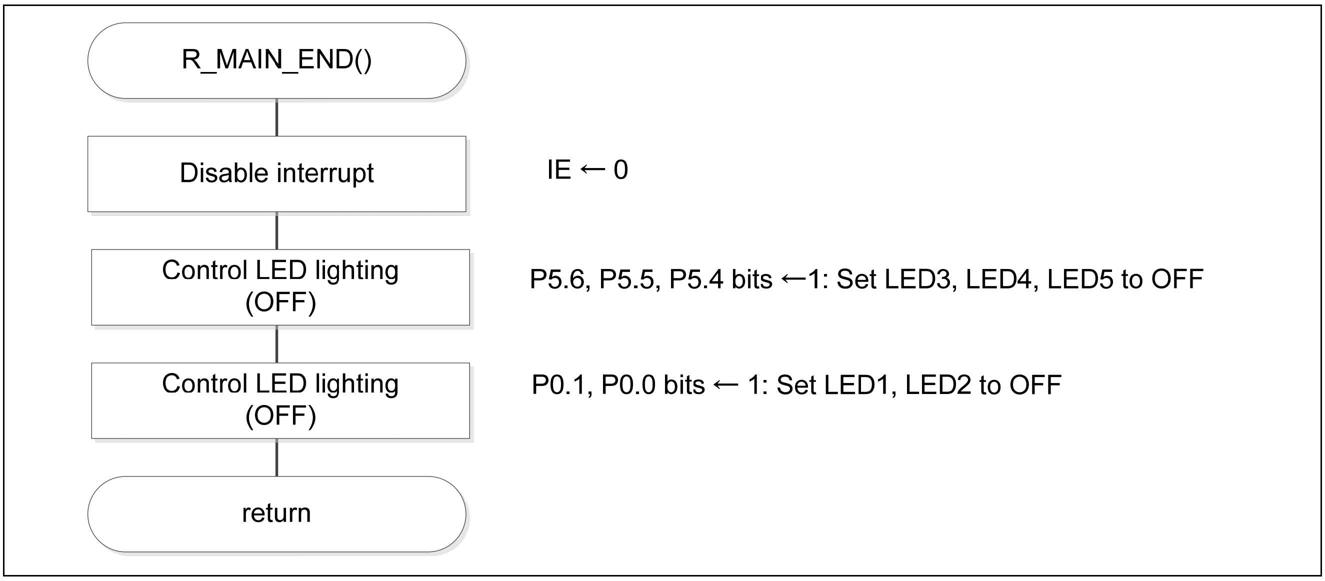

R_MAIN_END()
Disable interrupt
IE ← 0
Control LED lighting (OFF)
P5.6, P5.5, P5.4 bits ←1: Set LED3, LED4, LED5 to OFF
Control LED lighting (OFF)
P0.1, P0.0 bits ← 1: Set LED1, LED2 to OFF
return

Figure 5.59 Status Transition End Processing

5.6.44 External Interrupt Servicing

Figure 5.60 shows the flowchart for external interrupt servicing.

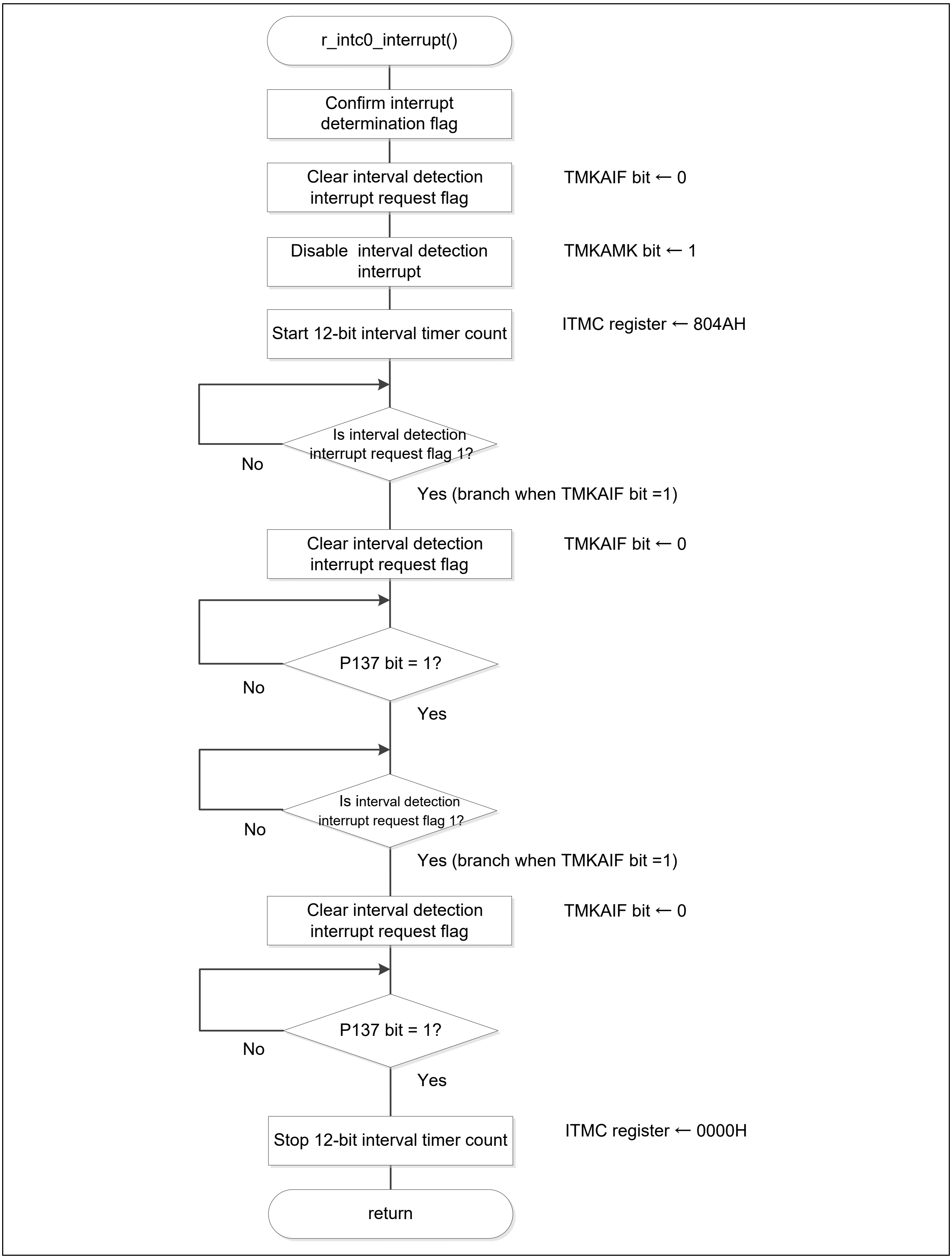

r_intc0_interrupt()
Confirm interrupt determination flag
Clear interval detection interrupt request flag
TMKAIF bit ← 0
Disable interval detection interrupt
TMKAMK bit ← 1
Start 12-bit interval timer count
ITMC register ← 804AH
Is interval detection interrupt request flag 1?
No
Yes (branch when TMKAIF bit =1)
Clear interval detection interrupt request flag
TMKAIF bit ← 0
P137 bit = 1?
No
Yes
Is interval detection interrupt request flag 1?
No
Yes (branch when TMKAIF bit =1)
Clear interval detection interrupt request flag
TMKAIF bit ← 0
P137 bit = 1?
No
Yes
Stop 12-bit interval timer count
ITMC register ← 0000H
return

Figure 5.60 External Interrupt Servicing

5.6.45 12-bit Interval Timer Interrupt Servicing

Figure 5.61 shows the flowchart for 12-bit interval timer interrupt servicing.

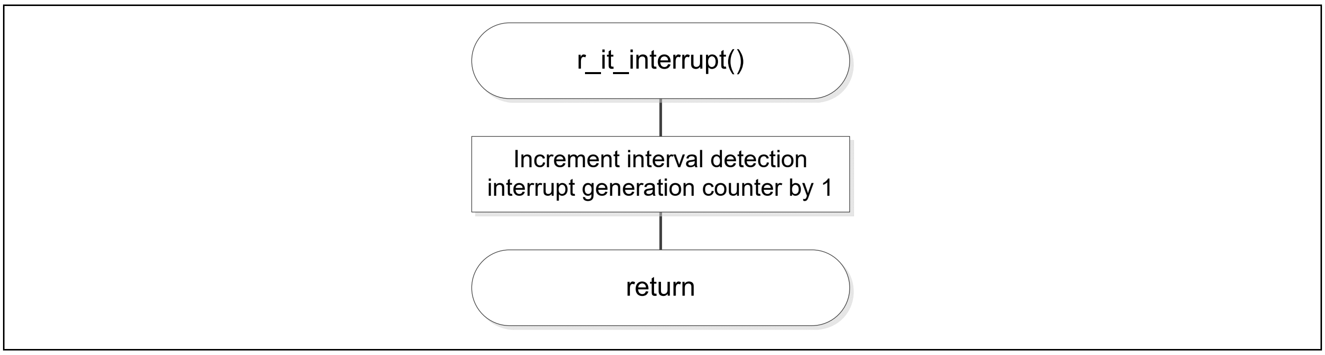

r_it_interrupt()
Increment interval detection interrupt generation counter by 1
return

Figure 5.61 12-bit Interval Timer Interrupt Servicing

5.6.46 A/D Conversion Completion Interrupt Servicing

Figure 5.62 shows the flowchart for A/D conversion completion interrupt servicing.

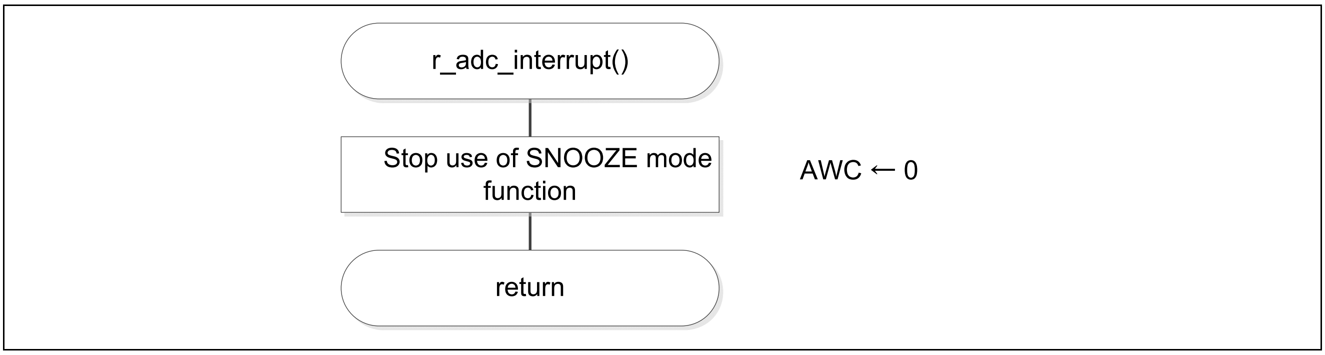


Figure 5.62 A/D Conversion Completion Interrupt Servicing

6. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

7. Reference Documents

RL78/G11 User’s Manual: Hardware (R01UH0637J)

RL78 Family User’s Manual: Software (R01US0015J)

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

Technical Update/Technical News

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

Website and Support

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

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

s

REVISION HISTORY	RL78/G11 CPU Clock Changing and Standby Settings (C Language) CC-RL

Rev.	Date	Description	
		Page	Summary
1.00	Nov. 30, 2016	—	First edition issued

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

   ¾ 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.

   ¾ 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.

   ¾ 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.

   ¾ 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 a product with a different part number, confirm that the change will not lead to problems.

   ¾ The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. 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.
2. 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.
3. 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.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.
   "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 etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.
   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). 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 for which it is not intended. 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 which the product is not intended by Renesas Electronics.
6. 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.
7. 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 systems manufactured by you.
8. 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.
9. 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. You should not use Renesas Electronics products or 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. When exporting the Renesas Electronics 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.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
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.

California Eastern Laboratories, Inc.
4590 Patrick Henry Drive, Santa Clara, California 95054-1817, U.S.A.
Tel: +1-408-919-2500, Fax: +1-408-988-0279

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: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, 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 India Pvt. Ltd.
No.777C, 100 Feet Road, HAL II Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 5.0