RISC-V

Safety Function (Frequency Detection)

Introduction

This application note describes the frequency detection function which is one of the safety features offered by the RISC-V.

The frequency detection function compares the high-speed on-chip oscillator clock or the external X1 oscillation clock and the low-speed on-chip oscillator clock. This allows detection of any abnormal clock frequencies.

Target Device

RISC-V

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

1.1 Overview of Specifications

The frequency detection function described in this application note compares the frequencies of the high-speed on-chip oscillator clock and the low-speed on-chip oscillator clock. This allows detection of any abnormal clock frequencies.

More specifically, this function measures the pulse interval under the conditions below to detect an abnormal frequency.

- The high-speed on-chip oscillator clock (HOCO clock) is selected as a count clock for timer array unit 0 (TAU0).
- The low-speed on-chip oscillator clock (32.768 kHz) should is selected as a timer input for channel 5 of TAU0.

To judge whether the frequency is normal or abnormal, the pulse interval is checked to see if it is within the tolerable range specified by constants. If the frequency is normal, the LED turns off. If not, the LED blinks.

The count clock for TAU0 is selected by setting the value of the high-speed on-chip oscillator frequency select register (SCKDIVCR) to one of predefined constants.

Table 1.1 lists the peripheral functions to be used and their uses.

Table 1.1 Peripheral Functions to be Used and their Uses

Peripheral Function	Use
External interrupt input (IRQ4)	Switch input Changes the count clock (HOCO clock) frequency for TAU0.
Channel 5 of timer array unit 0	Low-speed on-chip oscillator clock Measures pulse intervals.
Bit 7 in port 1	Displays the frequency detection result on the LED.
Bit 0 in port 1	Displays the selected HOCO clock on the LED.

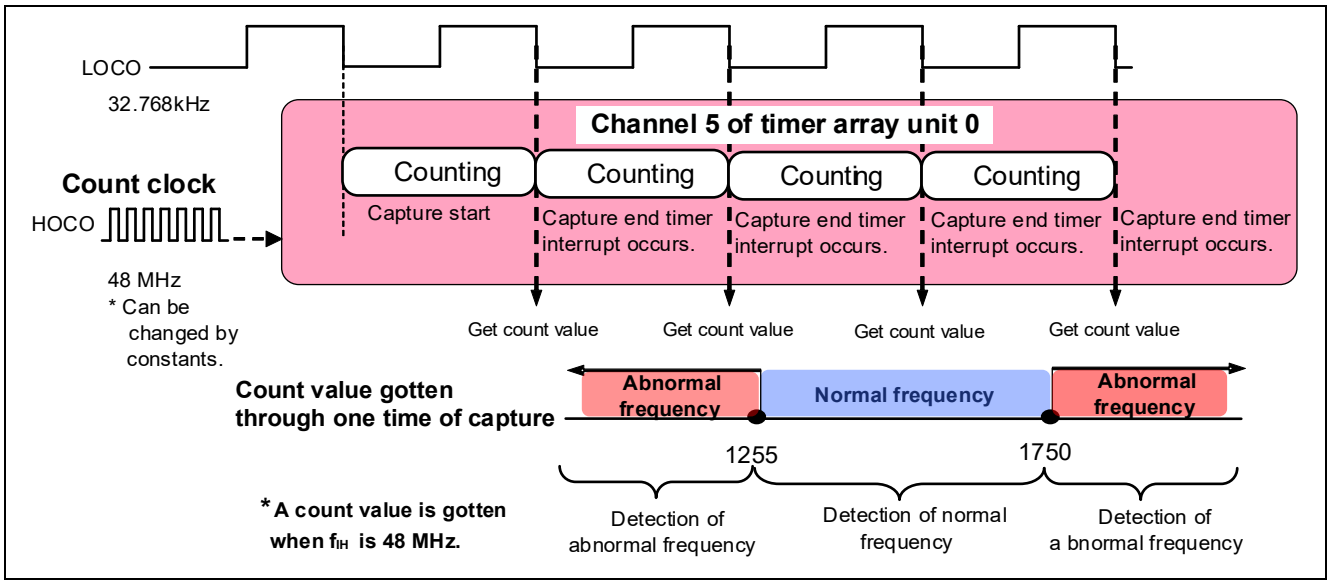

Figure 1.1 Outline of Frequency Detection

1.2 Operation Outline

The sample code shown in this application note compares the frequencies of the high-speed on-chip oscillator clock and the low-speed on-chip oscillator clock. This enables detection of any abnormal clock frequencies.

More specifically, the sample code measures the pulse interval under the conditions below to detect an abnormal frequency.

- The high-speed on-chip oscillator clock (HOCO clock) is selected as a count clock for timer array unit 0 (TAU0).
- The low-speed on-chip oscillator clock (32.768 kHz) is selected as a timer input for channel 5 of TAU0.

To judge whether the frequency is normal or abnormal, the pulse interval is checked to see if it is within the tolerable range specified by constants. If the frequency is normal, the LED1 turns off. If not, the LED1 blinks.

The tolerable range above depends on PULSEWIDTH_RANGE_MIN and PULSEWIDTH_RANGE_MAX in Table 4 2.

The count clock frequency for TAU0 can be changed dynamically by using the switch. Its frequency depends on TAU0_COUNT_CLOCK_1 and TAU0_COUNT_CLOCK_2 in Table 4 2. The LED2 turns off when the device operates with the count clock set by TAU0_COUNT_CLOCK_1. The LED2 turns on when the device operates with the count clock set by TAU0_COUNT_CLOCK_2.

(1) Initialize the TAU

Initialize the TAU.

<Conditions for setting>

— Select the HOCO clock as a count clock for TAU0.
— Select the low-speed on-chip oscillator clock (32.768 kHz) as a timer input for channel 5 of TAU0.

(2) Start the pulse interval measurement

The value captured by a first capture end timer interrupt (TAU0_ENDI5) is invalid. Thus, invalidate the first pulse interval measurement value according to the following processing:

— Set the TS05 bit of timer channel start register 0 (TS0) to 1 to enable counting. This clears the timer count register (TCR05) to 0000H and starts counting.
— Transition to the Sleep mode and wait until a capture end timer interrupt (TAU0_ENDI5) occurs.
— When the timer input enable edge is detected, the timer counter register (TCR05) value is captured into the timer data register (TDR05) and then a capture end timer interrupt (TAU0_ENDI5) will occur. (This results in a return from the Sleep status). Then, clear the timer counter register (TCR05) to 0000H.
— Clear the interrupt request flag for TAU0_ENDI5.

(3) Transition to Sleep mode

— Enter the Sleep mode and wait until the next enable edge input.
— Return from the Sleep mode by the processing of the second or subsequent capture end timer interrupt (TAU0_ENDI5) or by that of switching external interrupt (IRQ4).

(4) Check an interrupt source

The processing differs depending on the interrupt source upon return from the Sleep mode. Check the interrupt source after return from the Sleep mode.

＜When returning from the Sleep mode upon completion of pulse interval measurement＞

— Get a pulse interval measurement value.
— If the measurement value is within the tolerable range, LED1 turns off. Then, return to step (3).
— If it does not, LED1 blinks. Then, return to step (3).

＜When returning from the Sleep mode upon a switching external interrupt＞

— To prevent chattering, take actions (A) through (F) below.

A) Using the interrupt handler for external interrupts to set the EN0 bit in the interval timer control register (ITLCTL0) to 1. Then, start counting.
B) Wait until the value written to the EN0 bit is reflected.
C) Wait until an interval timer interrupt occurs.
D) Check the switch status with the interval timer interrupt handler. That is, check the P108 status.
E) If this status is 1, determine that the switch has not been depressed. Then, return to step (3).
F) If it is 0, determine that the switch has been depressed. Perform the operations below.
   - Stop the timer of the TAU0.
   - Change the HOCO clock which is the count clock for TAU0.
   - If the current count clock for TAU0 is set by TAU0_COUNT_CLOCK_1 in Table 4 2, change to the count clock set by TAU0_COUNT_CLOCK_2 and turn on LED2. Then, return to step (2).
   - If the current count clock for TAU0 is set by TAU0_COUNT_CLOCK_2 in Table 4 2, change to the count clock set by TAU0_COUNT_CLOCK_1 and turn off LED2. Then, return to step (2).

2. Operation Check Conditions

The sample code contained in this application note has been checked under the conditions listed in the table below.

Table 2.1 Operation Check Conditions

Item	Description
MCU used	RISC-V (R9A02G021)
Board used	RISC-V-48p Fast Prototyping Board (RTK9FPG021S000W0BJ)
Operating frequency	・High-speed on-chip oscillator clock (HOCO) : 48MHz ・CPU/peripheral hardware clock: The clock can be changed by depressing the target board's switch. Predefine each of the two constants for the clock as 48, 32 or 24MHz.
Operating voltage	3.3 V (can be operated at 1.6 V to 5.5 V) LVD0 detection voltage: Reset mode At rising edge TYP. 1.95 V (1.83 V to 2.07 V) At falling edge TYP. 1.90 V (1.78 V to 2.02 V)
Integrated development environment (e^{2}studio)	e^{2}studio V2024-01.1 (24.1.1) from Renesas Electronics Corp.
C compiler (e^{2}studio)	LLVM for RISC-V 17.0.2.202401
Smart configurator (SC)	Smart Configurator for RISC-V V24.1.1.v20240125-1623
Board support package (BSP)	V1.00 from Renesas Electronics Corp.

3. Description of the Hardware

3.1 Hardware Configuration Example

The example of configuration of the hardware that is used for this application note is shown below.

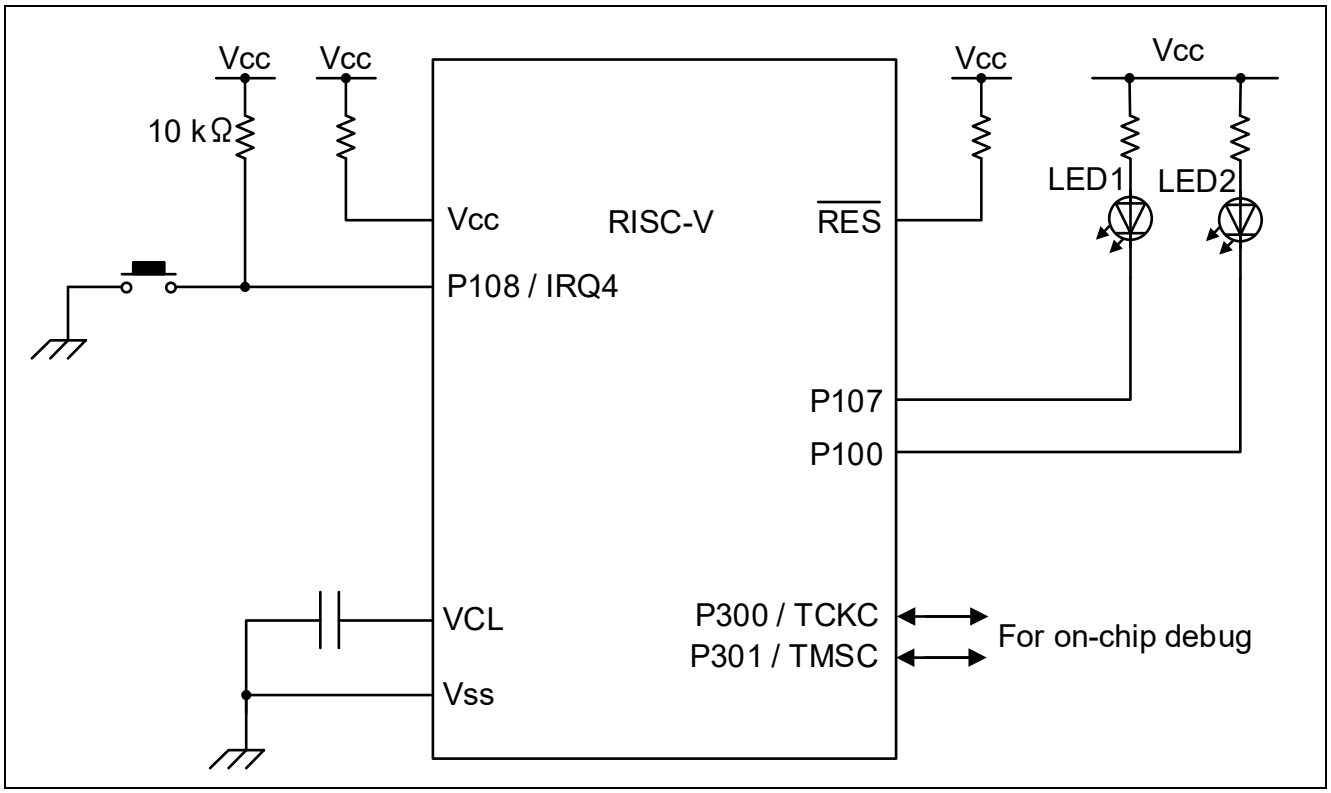

Figure 3.1 Hardware Configuration

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 appropriate pin handling and meets electrical characteristic requirements (connect each input-only port to V_{CC} or V_{SS} through a resistor).

Note 2. V_{CC} must not be lower than the reset release voltage (V_{LVD0}) that is specified for the LVD0.

3.2 List of Pins to be used

Table 3.1 lists the pins to be used and their functions.

Table 3.1 Pins to be Used and their Functions

Pin name	I/O	Function
P108/IRQ4	Input	Switch input Changes the count clock for TAU0.
P107	Output	Displays the frequency detection result on the LED.
P100	Output	Displays the selected HOCO clock on the LED.

4. Description of the Software

4.1 List of Option Byte Settings

Table 4.1 summarizes the settings of the option bytes.

Table 4.1 Option Byte Settings

Address	Setting Value	Contents
0000_0400H	FFFF_FFFFH	Disables the watchdog timer. (Counting stopped after reset)
0000_0404H	FFFF_CFDFH	LVD0 detection voltage: reset mode At rising edge TYP. 1.95 V (1.83 V to 2.07 V) At falling edge TYP. 1.90 V (1.78 V to 2.02 V)
		High-speed on-chip oscillator clock : 48 MHz
0101_0008H	FFFF_FFFFH	Enables on-chip debugging

4.2 List of Constants

Table 4.2 lists the constants that are used in this sample program.

Table 4.2 Constants for Sample Program

Constant	Setting	Description
_0001_TAU_OVERFLOW_OCCURS	0x0001U	Overflow occurrence detection
PULSEWIDTH_RANGE_MIN	1255	Lower limit of the tolerable range for pulse interval measurement
PULSEWIDTH_RANGE_MAX	1750	Upper limit of the tolerable range for pulse interval measurement
TAU0_COUNT_CLOCK_1	0x00	Count clock 1 for TAU0. It is predefined as one of these values: 0x00:24MHz 0x02:32MHz 0x04:48MHz
TAU0_COUNT_CLOCK_2	0x01	Count clock 2 for TAU0. Same as TAU0_COUNTCLOCK_1.

Note 1: The above PULSEWIDTH_RANGE_MIN and PULSEWIDTH_RANGE_MAX are calculated as follows.
PULSEWIDTH_RANGE_MIN = HOCO_FREQUENCY_MIN / LOCO_FREQUENCY_MAX
PULSEWIDTH_RANGE_MAX = HOCO_FREQUENCY_MAX / LOCO_FREQUENCY_MIN
HOCO_FREQUENCY_MIN: Value (47.28MHz) calculated by considering a tolerance of the HOCO frequency.
LOCO_FREQUENCY_MAX: Value (37.68kHz) calculated by considering a tolerance of the low-speed on-chip oscillator frequency.
HOCO_FREQUENCY_MAX: Value (48.72MHz) calculated by considering a tolerance of the HOCO frequency.
LOCO_FREQUENCY_MIN: Value (27.85kHz) calculated by considering a tolerance of the low-speed on-chip oscillator frequency.
Change the upper and lower limits of the tolerable range for pulse interval measurement depending on the system used.

4.3 List of Variables

Table 4.3 lists the global variables.

Table 4.3 Global Variables

Type	Variable Name	Contents	Function Used
static uint8_t	g_MeasureEndFlag	Pulse interval measurement end flag	r_Config_TAU0_5_interrupt r_tau0_channel5_get_measure_status r_tau0_channel5_clear_measure_status
static uint8_t	g_ICLKChangeFlag	Count clock change request flag	R_Config_ITL000_Callback_Shared_Interrupt r_it_get_iclk_change_flag r_it_clear_iclk_change_flag
static uint8_t	g_irq4_flag	IRQ4 external interrupt occurrence flag	r_Config_ICU_irq4_interrupt r_intc0_get_irq4_flag r_intc0_clear_irq4_flag
static uint8_t	g_intit_flag	Interval timer interrupt occurrence flag	R_Config_ITL000_Callback_Shared_Interrupt r_it_get_intit_flag r_it_clear_intit_flag

4.4 List of Functions

Table 4.4 lists the functions that are used in this sample program.

Table 4.4 List of Functions

Function name	Outline
r_main_iclk_change()	Changes the count clock.
r_main_iclk_change()	Changes the HOCO clock.
r_main_get_pulsewidth_measure_result()	Judges the pulse interval measurement value.
r_main_start_led_blink()	Starts blinking the LED.
r_main_stop_led_blink()	Stops blinking the LED.
r_intc0_get_irq4_flag()	Gets the IRQ4 external interrupt occurrence flag
r_intc0_clear_irq4_flag()	Clears the IRQ4 external interrupt occurrence flag
r_it_get_intit_flag()	Gets the interval timer interrupt occurrence flag
r_it_clear_intit_flag()	Clears the interval timer interrupt occurrence flag
r_tau0_channel5_measure_start()	Starts pulse interval measurement
r_tau0_channel5_get_measure_status()	Gets the pulse interval measurement end flag
r_tau0_channel5_clear_measure_status()	Clears the pulse interval measurement end flag
r_it_get_iclk_change_flag()	Gets the count clock change request flag
r_it_clear_iclk_change_flag()	Clears the count clock change request flag.

4.5 Function Specifications

This section describes the specifications for the functions that are used in the sample code.

r_main_iclk_change()	
Outline	Changes the count clock.
Header	r_cg_userdefine.h
Declaration	void r_main_iclk_change(void)
Description	This function stops channel 5 of TAU0. Then, it changes the HOCO clock which is a count clock for TAU0.
Argument	None
Return Value	None

r_main_hoco_change()	
Outline	Changes the HOCO clock.
Header	r_cg_userdefine.h
Declaration	void r_main_hoco_change(uint8_t clock)
Description	This function changes the HOCO clock to the clock set by TAU0_COUNT_CLOCK_1 or the clock set by TAU0_COUNT_CLOCK_2. These constants are listed in Table 4 2.
Argument	None
Return Value	None

r_main_get_pulsewidth_measure_result()	
Outline	Judges the pulse interval measurement value.
Header	r_cg_userdefine.h
Declaration	uint8_t r_main_get_pulsewidth_measure_result(void)
Description	This function judges the pulse interval measurement value. That is, it judges whether the global variable (g_tau0_ch5_width) containing the pulse interval measurement value is within the tolerable range for pulse interval measurement in Table 4 2.
Argument	None
Return Value	When the pulse interval measurement value is within the tolerable range :0x00 When the pulse interval measurement value is not within the tolerable range :0x01

r_main_start_led_blink()	
Outline	Starts blinking the LED.
Header	r_cg_userdefine.h
Declaration	void r_main_start_led_blink(void)
Description	This function starts blinking the LED.
Argument	None
Return Value	None

r_main_stop_led_blink()	
Outline	Stops blinking the LED.
Header	r_cg_userdefine.h
Declaration	void r_main_stop_led_blink(void)
Description	This function stops blinking the LED.
Argument	None
Return Value	None

r_intc0_get_irq4_flag()	
Outline	Gets the IRQ4 external interrupt occurrence flag.
Header	r_cg_macrodriver.h r_cg_userdefine.h Config_ICU.h Config_ITL000.h
Declaration	uint8_t r_intc0_get_irq4_flag(void)
Description	This function gets the IRQ4 external interrupt occurrence flag.
Argument	None
Return Value	None

r_intc0_clear_irq4_flag()	
Outline	Clears the INTP0 external interrupt occurrence flag.
Header	r_cg_macrodriver.h r_cg_userdefine.h Config_ICU.h Config_ITL000.h
Declaration	void r_intc0_clear_irq4_flag(void)
Description	This function clears the IRQ4 external interrupt occurrence flag.
Argument	None
Return Value	None

r_it_get_intit_flag()	
Outline	Gets the interval timer interrupt occurrence flag.
Header	r_cg_macrodriver.h r_cg_userdefine.h Config_ICU.h
Declaration	uint8_t r_it_get_intit_flag(void)
Description	This function gets the interval timer interrupt occurrence flag.
Argument	None
Return Value	When an interval timer interrupt has not occurred :0x00 When an interval timer interrupt has occurred :0x01

r_it_clear_intit_flag()	
Outline	Clears the interval timer interrupt occurrence flag.
Header	r_cg_macrodriver.h r_cg_userdefine.h Config_ICU.h
Declaration	void r_it_clear_intit_flag(void)
Description	This function clears the interval timer interrupt occurrence flag.
Argument	None
Return Value	None

r_tau0_channel5_measure_start()	
Outline	Discards pulse interval measurement.
Header	Config_TAU0_5.h
Declaration	void r_tau0_channel5_measure_start(void)
Description	This function waits until the first pulse interval measurement after the start of the timer (TM05). Then, it clears the interrupt flag.
Argument	None
Return Value	None

r_tau0_channel5_get_measure_status()	
Outline	Gets the pulse interval measurement end flag.
Header	r_cg_macrodriver.h r_cg_userdefine.h Config_TAU0_5.h
Declaration	void r_tau0_channel5_clear_measure_status(void)
Description	This function gets the pulse interval measurement end flag.
Argument	None
Return Value	When pulse interval measurement is not ended :0x00 When pulse interval measurement is ended :0x01

r_tau0_channel5_clear_measure_status()	
Outline	Clears the pulse interval measurement end flag.
Header	r_cg_macrodriver.h r_cg_userdefine.h Config_TAU0_5.h
Declaration	void r_tau0_channel5_clear_measure_status(void)
Description	This function clears the pulse interval measurement end flag to 0.
Argument	None
Return Value	None

r_it_get_iclk_change_flag()		
Outline	Gets the count clock change request flag.	
Header	r_cg_macrodriver.h r_cg_userdefine.h Config_ITL000.h	
Declaration	uint8_t r_it_get_iclk_change_flag(void)	
Description	This function gets the count clock change request flag.	
Argument	None	
Return Value	When count clock change is not requested When count clock change is requested	:0x00 :0x01

r_it_clear_iclk_change_flag()	
Outline	Clears the count clock change request flag.
Header	r_cg_macrodriver.h r_cg_userdefine.h Config_ITL000.h
Declaration	void r_it_clear_iclk_change_flag(void)
Description	This function clears the count clock change request flag to 0.
Argument	None
Return Value	None

4.6 Flowcharts

4.6.1 Main Processing

Figure 4.1 to Figure 4.3 shows the flowchart for the main processing.

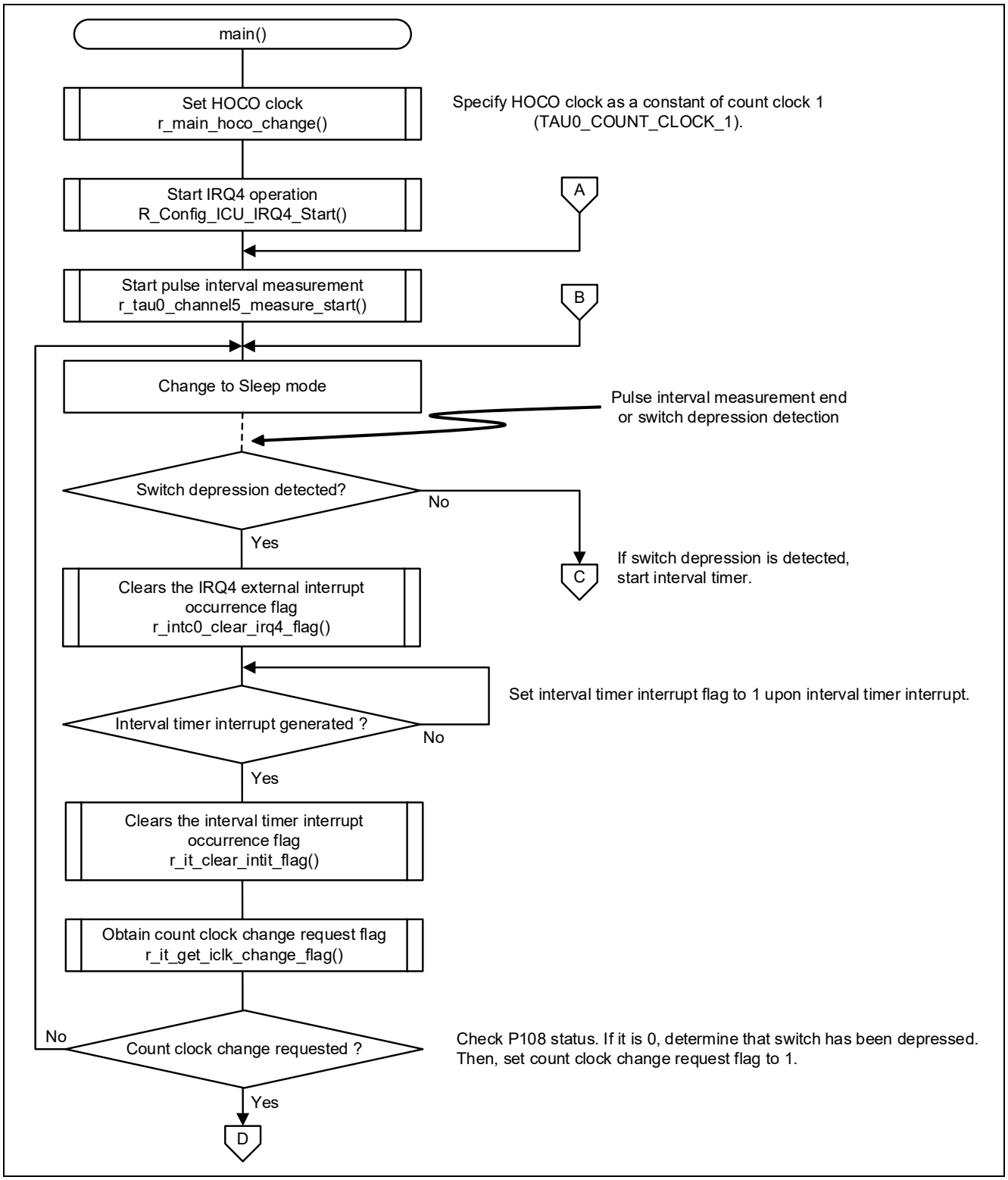

Figure 4.1 Main Processing (1/3)

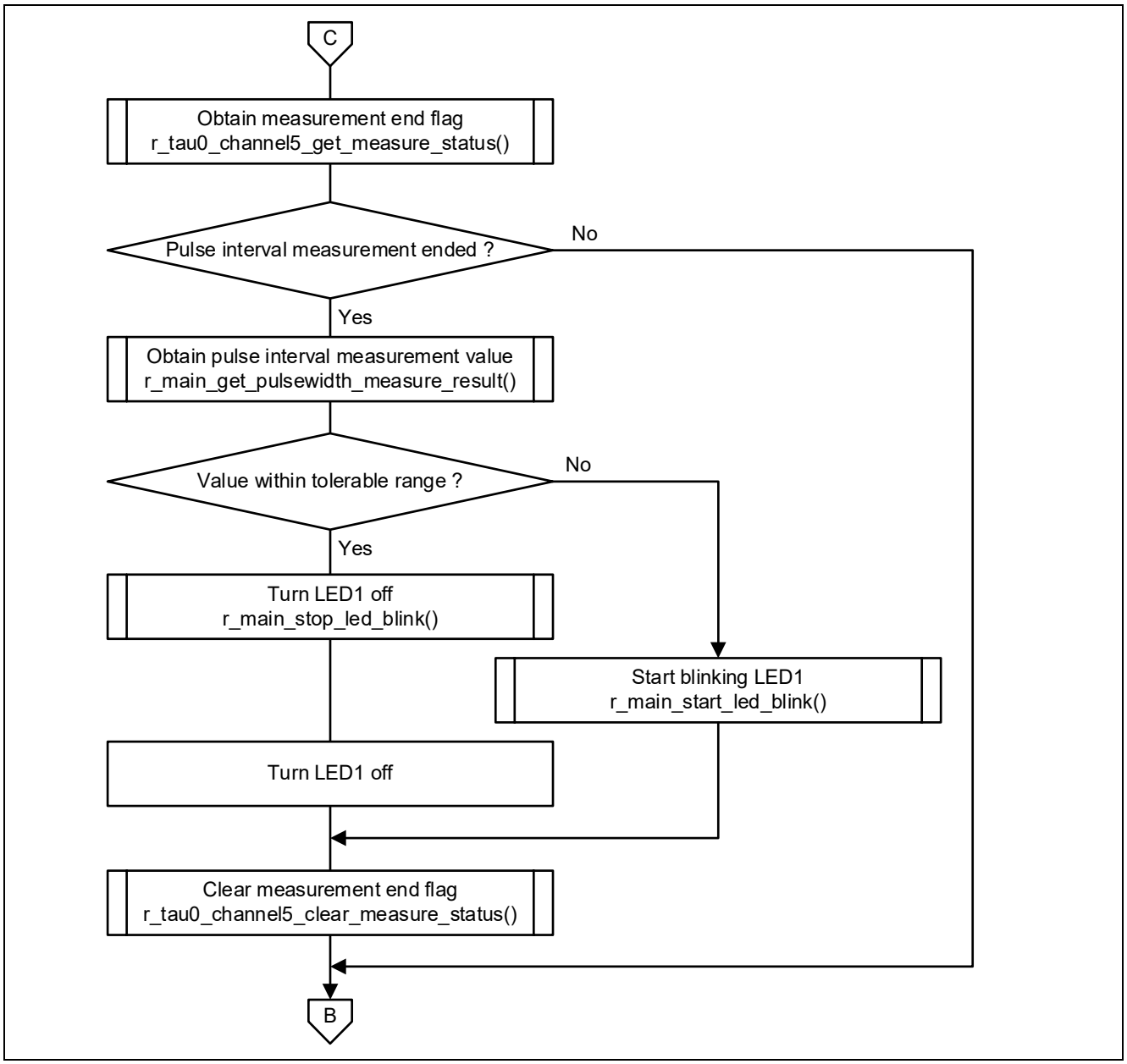

Figure 4.2 Main Processing (2/3)

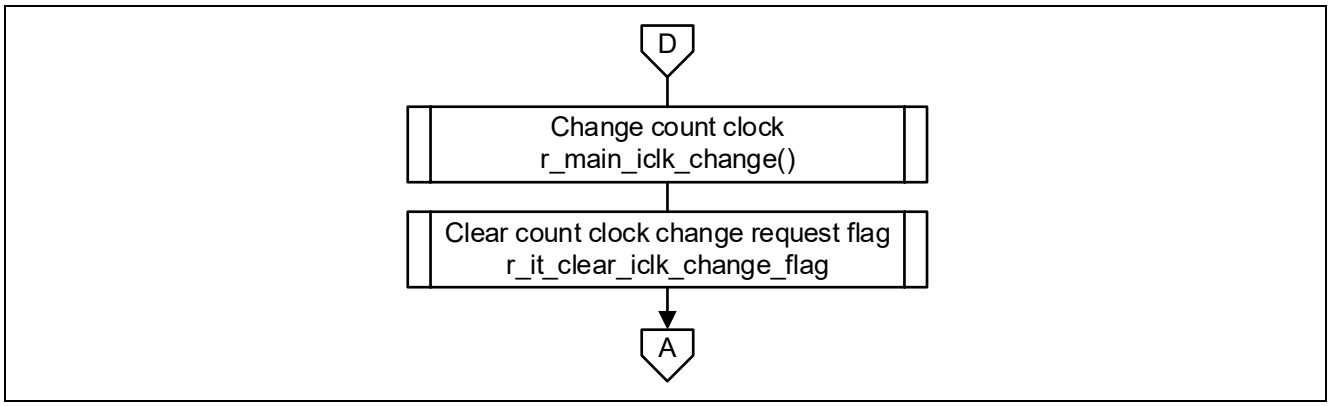

Figure 4.3 Main Processing (3/3)

4.6.2 Changing Count Clock

Figure 4.4 shows the flowchart for changing the count clock.

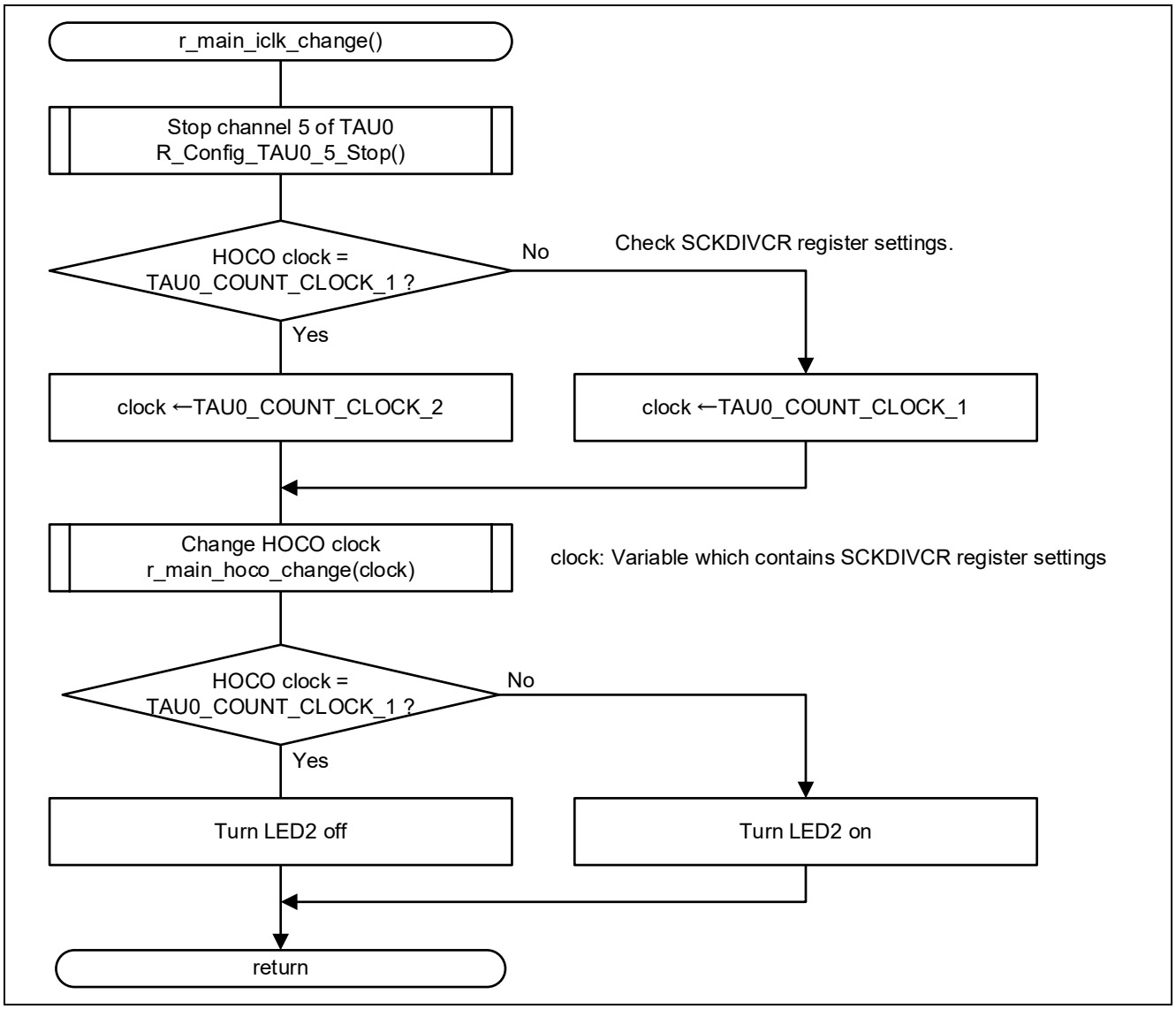

Figure 4.4 Changing Count Clock

4.6.3 Changing HOCO Clock

Figure 4.5 shows the flowchart for changing the HOCO clock.

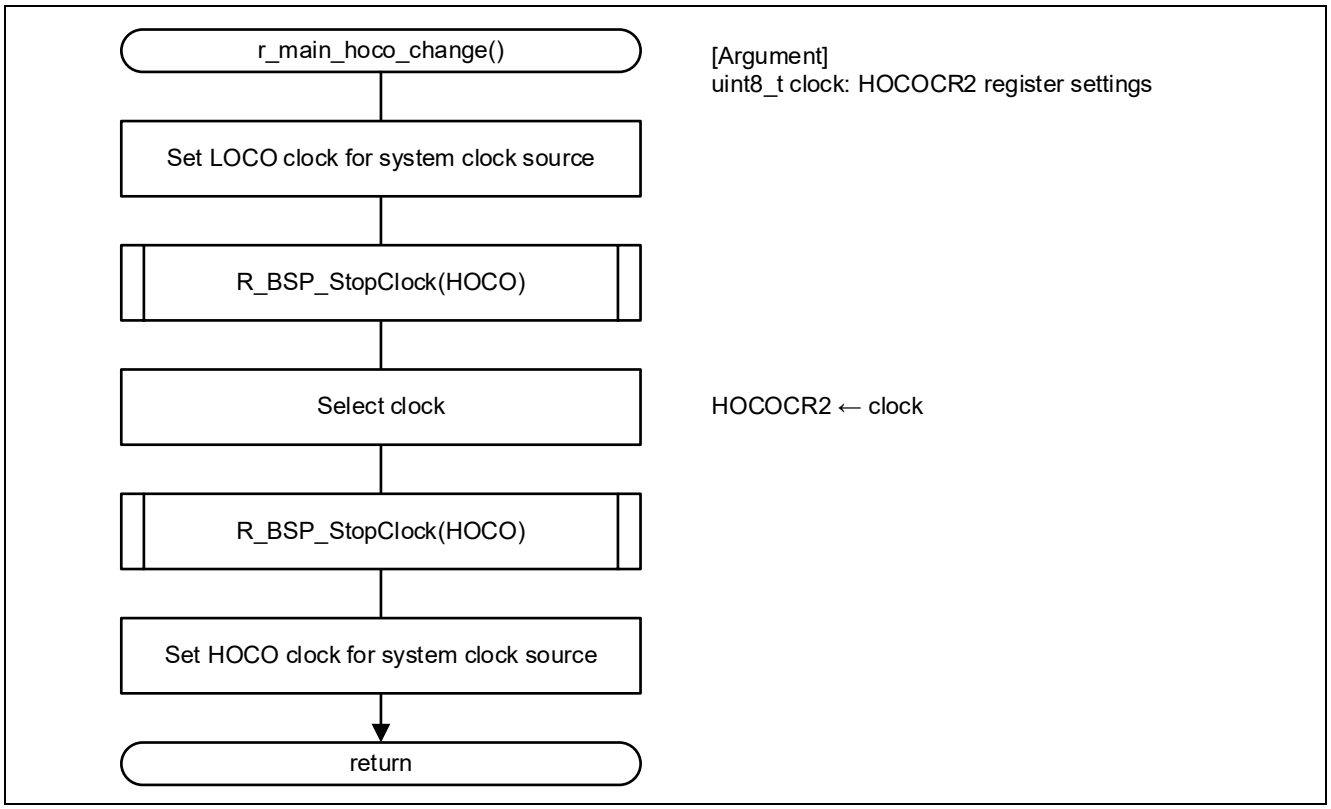

Figure 4.5 Changing HOCO Clock

4.6.4 Judging Pulse Interval Measurement Value

Figure 4.6 shows the flowchart for judging the pulse interval measurement value.

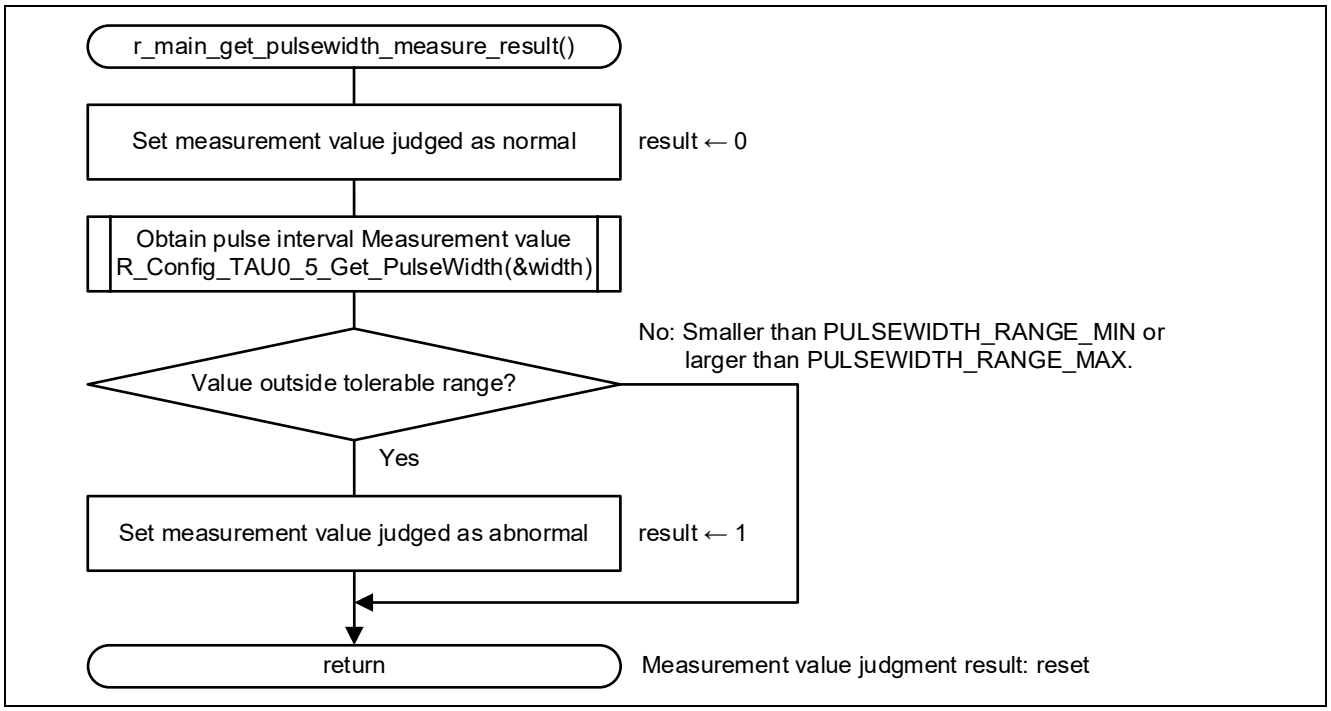

Figure 4.6 Judging Pulse Interval Measurement Value

4.6.5 Starting LED Blink

Figure 4.7 shows the flowchart for starting the LED blink.

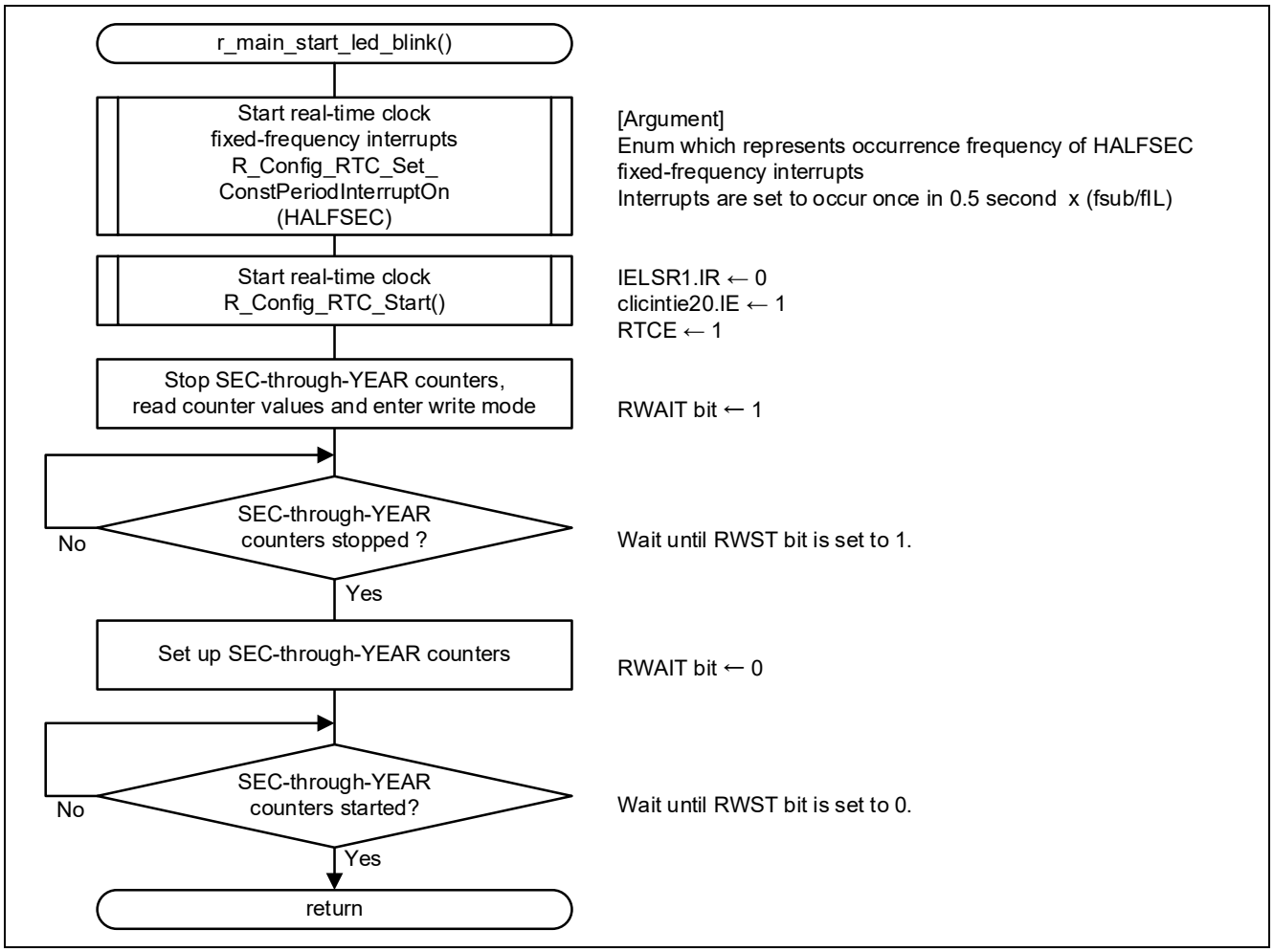

Figure 4.7 Starting LED Blink

4.6.6 Stopping LED Blink

Figure 4.8 shows the flowchart for stopping the LED blink.

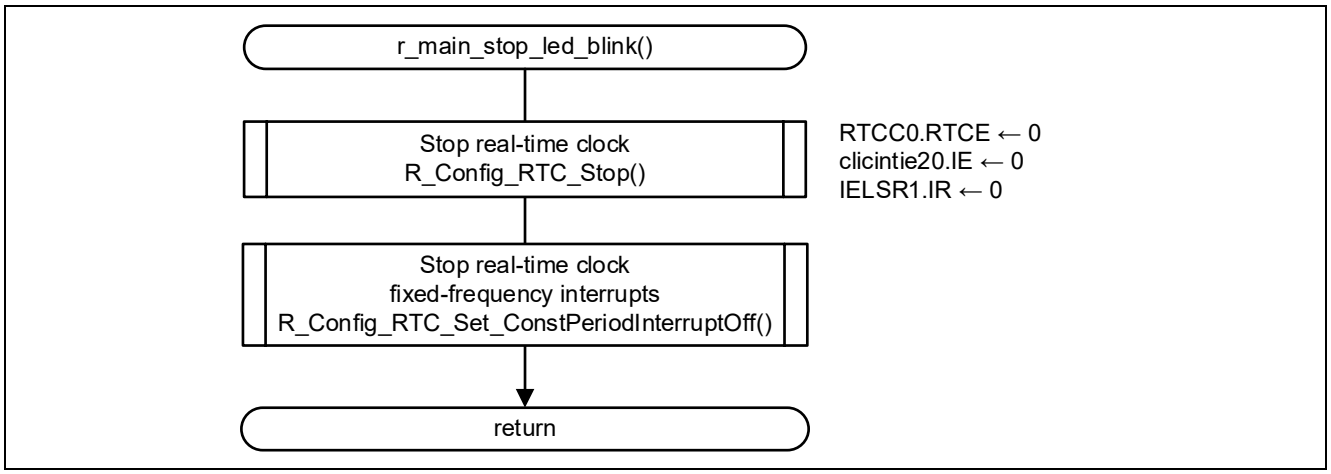

Figure 4.8 Stopping LED Blink

4.6.7 Getting IRQ4 External Interrupt Occurrence Flag

Figure 4.9 shows the flowchart for getting the IRQ4 external interrupt occurrence flag. This function does nothing but returns global variable g_irq4_flag as a return value.

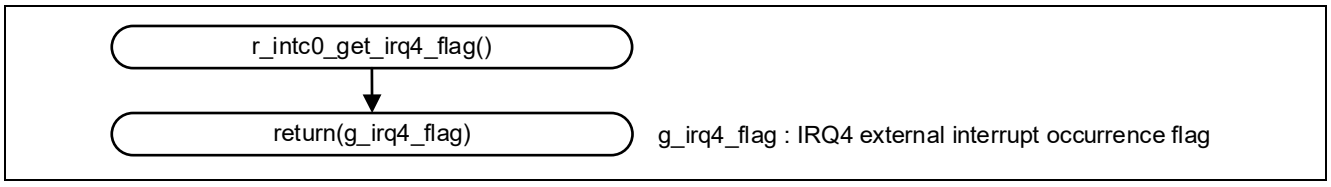

Figure 4.9 Getting IRQ4 External Interrupt Occurrence Flag

4.6.8 Clearing IRQ4 External Interrupt Occurrence Flag

Figure 4.10 shows the flowchart for clearing the IRQ4 external interrupt occurrence flag.

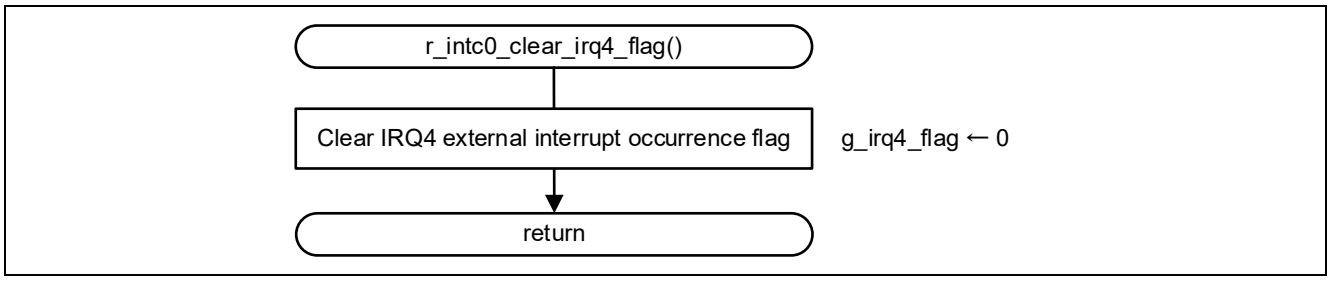

Figure 4.10 Clearing IRQ4 External Interrupt Occurrence Flag

4.6.9 Getting Interval Timer Interrupt Occurrence Flag

Figure 4.11 shows the flowchart for getting the interval timer interrupt occurrence flag. This function does nothing but returns global variable g_intit_flag as a return value.

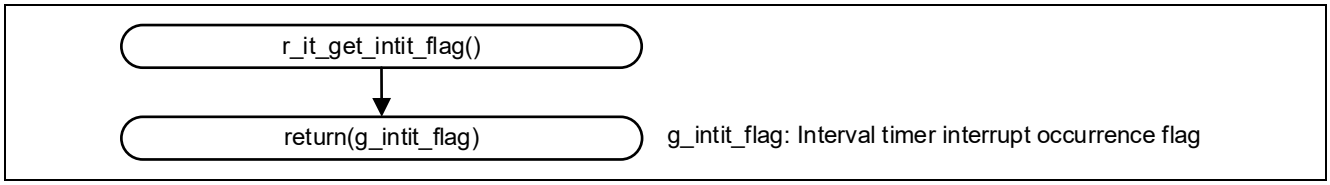

Figure 4.11 Getting Interval Timer Interrupt Occurrence Flag

4.6.10 Clearing Interval Timer Interrupt Occurrence Flag

Figure 4.12 shows the flowchart for clearing the interval timer interrupt occurrence flag.

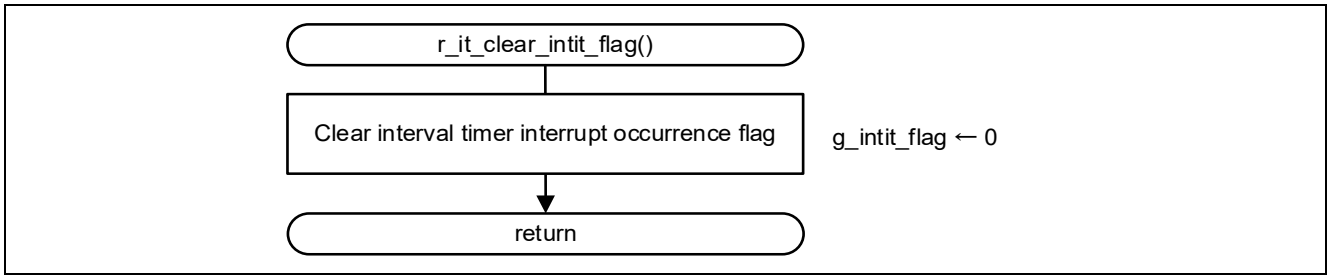

Figure 4.12 Clearing Interval Timer Interrupt Occurrence Flag

4.6.11 Starting Pulse Interval Measurement

Figure 4.13 shows the flowchart for starting pulse interval measurement.

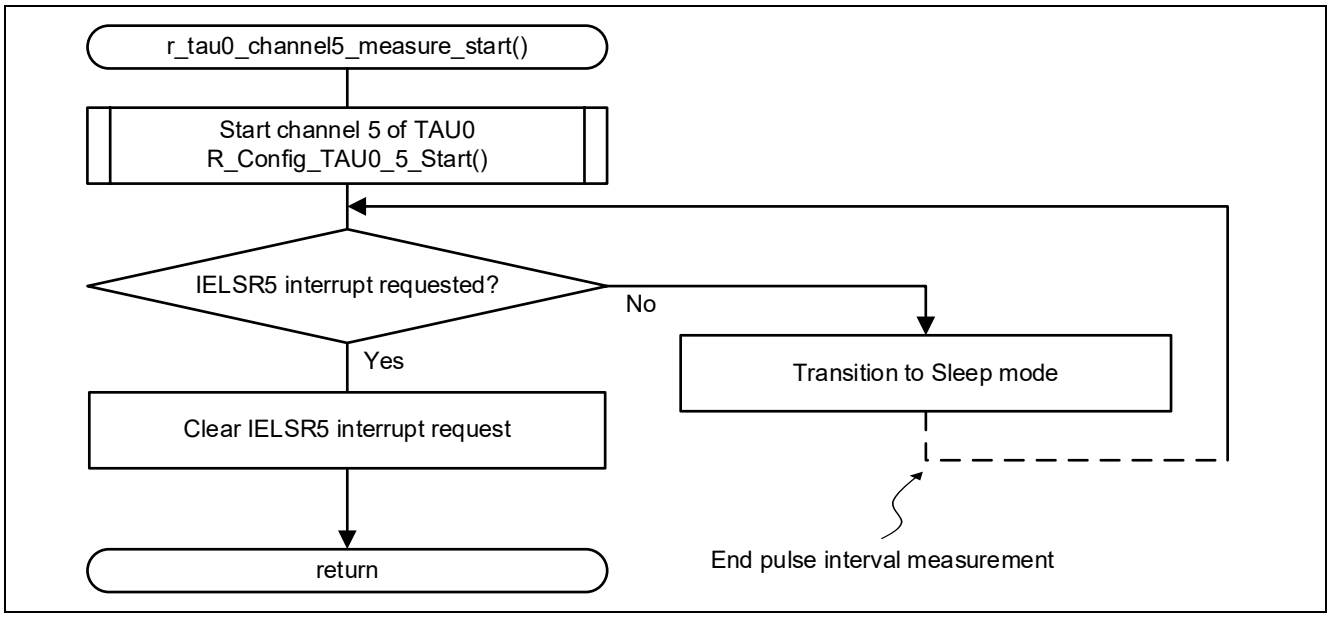

Figure 4.13 Starting Pulse Interval Measurement

4.6.12 Getting Pulse Interval Measurement End Flag

Figure 4.14 shows the flowchart for getting the pulse interval measurement end flag. This function does nothing but returns global variable g_MeasureEndFlag as a return value.

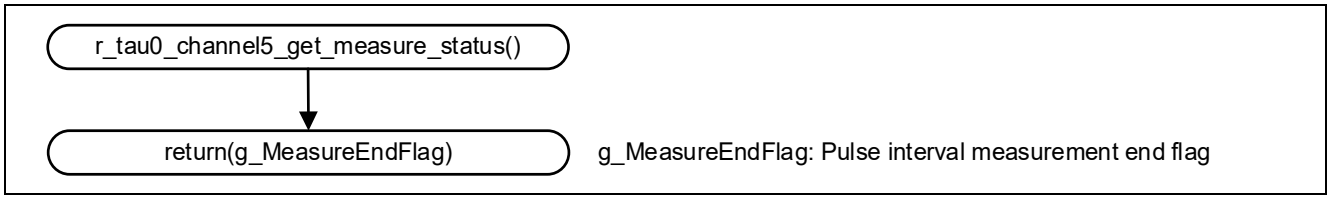

Figure 4.14 Getting Pulse Interval Measurement End Flag

4.6.13 Clearing Pulse Interval Measurement End Flag

Figure 4.15 shows the flowchart for clearing the pulse interval measurement end flag.

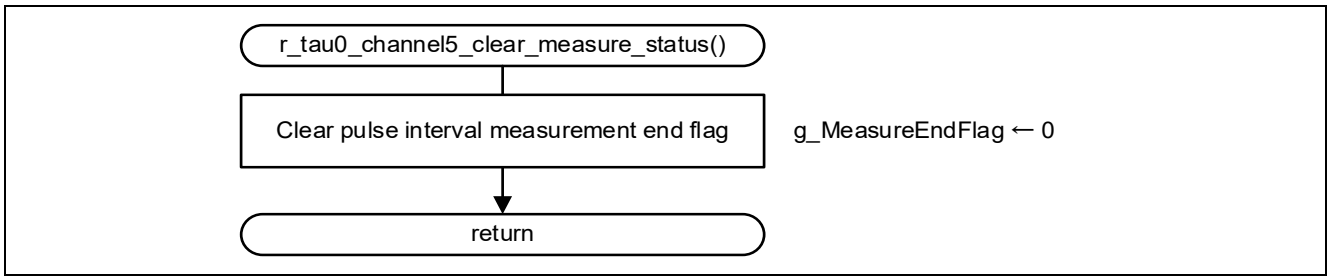

Figure 4.15 Clearing Pulse Interval Measurement End Flag

4.6.14 Getting Count Clock Change Request Flag

Figure 4.16 shows the flowchart for getting the count clock change request flag. This function does nothing but returns global variable g_ICLKChangeFlag as a return value.

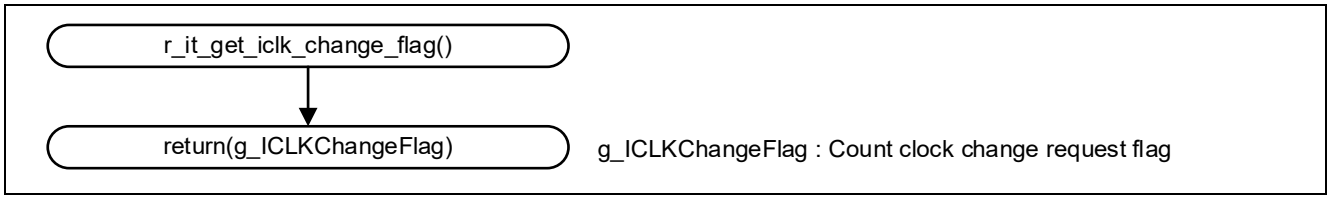

Figure 4.16 Getting Count Clock Change Request Flag

4.6.15 Clearing Count Clock Change Request Flag

Figure 4.17 shows the flowchart for clearing the count clock change request flag.

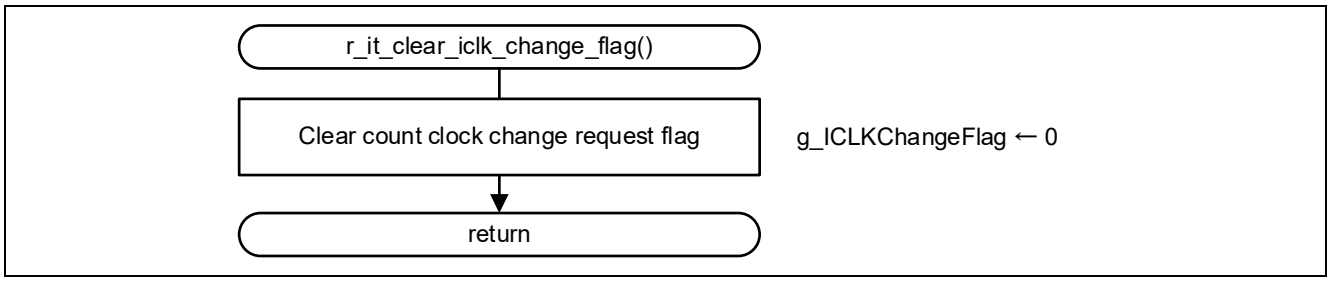

Figure 4.17 Clearing Count Clock Change Request Flag

5. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

6. Reference Documents

RISC-V User's Manual: Hardware (R01UH1036EJ)

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

Technical update

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

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Mar.18.24	—	Initial release

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. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time 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 time 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 time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the 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.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, 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.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of 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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. 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 shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended 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; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or 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 damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall 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 shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.