RX13T Group

HOCO Calibration Using the CAC

Introduction

This application note describes the way of adjusting the frequency of the high-speed on-chip oscillator (HOCO) using the clock frequency accuracy measurement circuit (CAC) on the RX13T Group.

Target Device

RX13T Group

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to match the specifications of the alternative MCU.

Contents

1. Specifications

The HOCO oscillation frequency may differ from the factory-configured frequency due to external factors such as ambient temperature. Calibration needs to be performed to compensate for the frequency error. The high-speed on-chip oscillator trimming register (HOCOTRR0) is used to adjust the HOCO oscillation frequency. The HOCO frequency error can be compensated for by adjusting the value of the HOCOTRR0 register at regular intervals.

The sample program described in this application note uses the CAC and the compare match interrupt occurred from compare match timer to measure the HOCO oscillation frequency and adjusts the value of the HOCOTRR0 register based on the results obtained. It also outputs a clock equivalent to HOCO divided by 4 on the MTIOC0B pin.

Table 1.1 lists the peripheral functions and their applications, and Figure 1.1 shows a block diagram.

The external clock shown in the block diagram is used as a measurement reference clock, and it affects the calibration accuracy. Therefore, a signal generator that is as accurate as possible should be used to produce the external clock. For example, when using a signal generator with absolutely no error for the measurement reference clock, the error from the HOCO frequency of 32 MHz would be ±0.1% (the default setting in the sample code). If the error of the signal generator is large, the calibration error is even larger, so select a signal generator such that the error is within the acceptable range in the actual usage environment.

Table 1.1 Peripheral Functions and Their Applications

Peripheral Function	Application
CAC	Measures the HOCO frequency based on the CACREF pin input.
CMT	Starts HOCO measurement when a compare match interrupt occurs.
MTU	Outputs a clock equivalent to HOCO divided by 4.

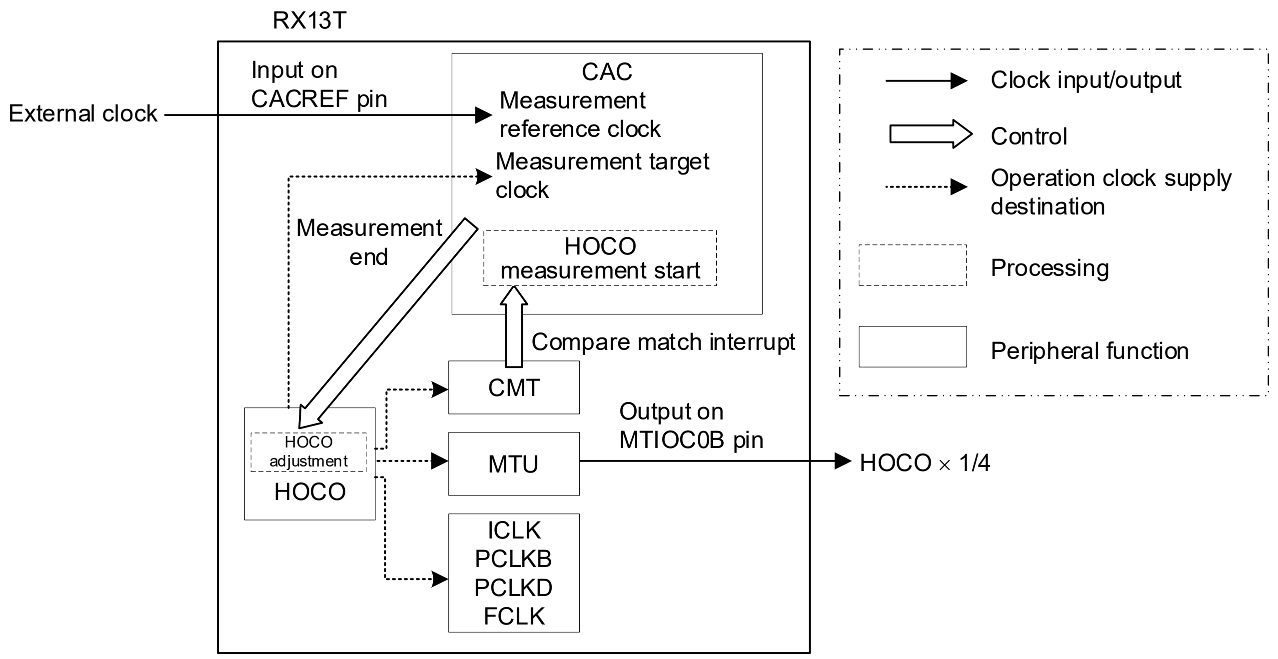


Figure 1.1 Block Diagram

2. Operation Confirmation Environment

2.1 Operation Confirmation Conditions

Table 2.1 lists the conditions under which the operation of the sample code referenced in this application note has been confirmed.

Table 2.1 Operation Confirmation Conditions

Item	Description
MCU used	R5F513T5ADFL (RX13T Group)
Operating frequency	HOCO: 32 MHz System clock (ICLK): 32 MHz (HOCO × 1/1) Peripheral module clock B (PCLKB): 32 MHz (HOCO × 1/1) Peripheral module clock D (PCLKD): 32 MHz (HOCO × 1/1) FlashIF clock (FCLK): 32 MHz (HOCO × 1/1)
Operating voltage	5.0 V
Integrated development environment	Renesas Electronics e² studio Version: 7.5.0
C compiler	Renesas Electronics C/C++ Compiler for RX Family V.3.01.00
	Compiler option Default settings of integrated development environment
iodefine.h version	Version 1.00
Endian order	Little endian or big endian
Operating mode	Single-chip mode
Processor mode	Supervisor mode
Sample code version	Version 1.00
Board used	RX13T CPU card (product number: RTK0EMXA10C00000BJ)
Function generator	Use a signal generator equipped with analog signal output pins and capable of outputting rectangular waves at a frequency accuracy of 2 ppm (±0.0002%, at 18°C to 28°C). The output signal should have a bias of 2.5 V relative to GND. Output a 32 Hz rectangular wave with an amplitude setting of 5.0 Vpp. Figure 2.1 shows the output waveform.

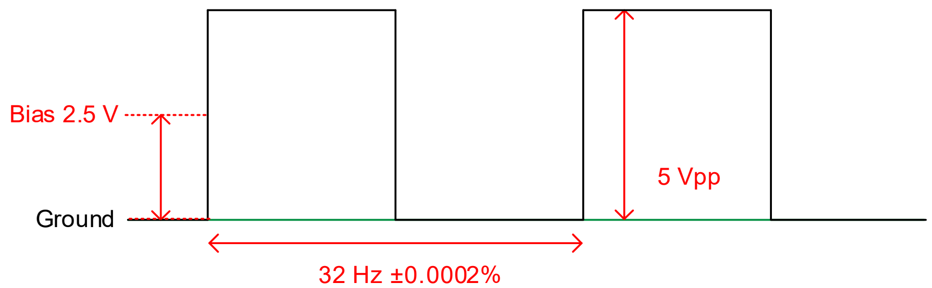
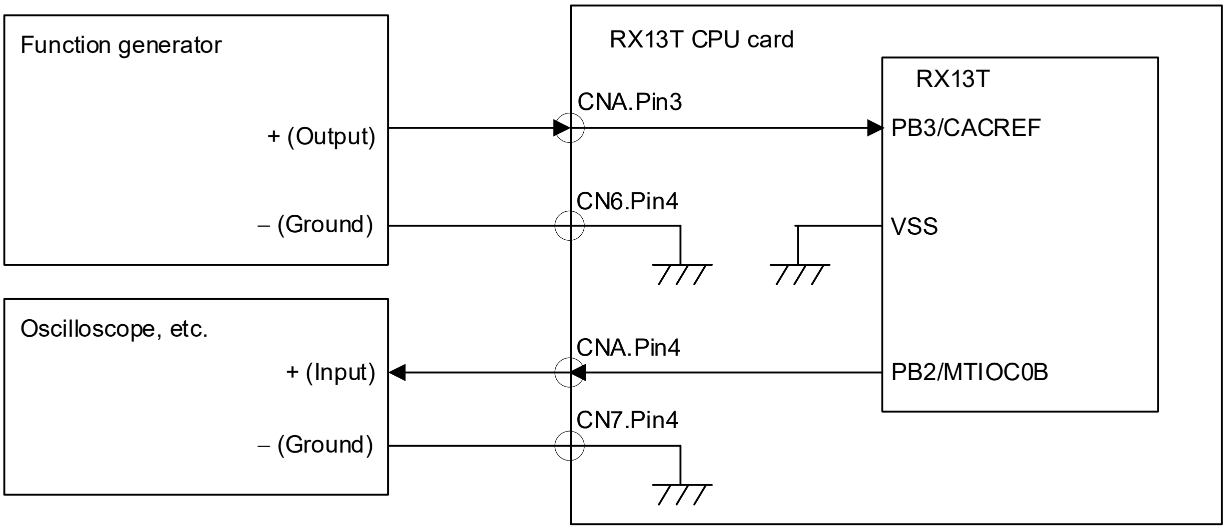


Figure 2.1 Function Generator Output Waveform

2.2 Pin Connections

Figure 2.2 shows the pin connections for this application note.



The arrows in the figure indicate input/output directions.

Figure 2.2 Pin Connections

3. Related Application Note

For additional information related to this document, refer to the following application note:

RX13T Group Initial Settings Example (R01AN4890)

If a newer version is available, use it instead. Visit the Renesas Electronics Corporation website to check and download the latest version.

4. Hardware

4.1 Pin Used

Table 4.1 lists the pins used and their functions.

Table 4.1 Pins Used and Their Functions

Pin Name	I/O	Description
PB3/CACREF	Input	Measurement reference clock input
PB2/MTIOC0B	Output	Outputs a clock equivalent to HOCO divided by 4.

5. Software

5.1 Operation Overview

In the sample program described in this application note a 32 Hz rectangular wave is used as the CAC's measurement reference clock and HOCO (32 MHz) divided by 32 is used as the measurement target clock. The measurement reference clock is produced by a function generator and input on the CACREF pin. The CAC counts valid edges of the measurement target clock in the period from a rising edge to the next rising edge of the measurement reference clock.

Measurement starts when a compare match interrupt is occurred from the compare match timer (CMT) and is performed five times in succession.

Of the five measurements, the second to the fifth are averaged and used as the measurement result. If the difference between the measurement result and the theoretical value is outside the allowable range, addition or subtraction is performed on the HOCOTRR0 register value. If the difference between the measurement result and the theoretical value is within the allowable range, the HOCOTRR0 register value is not changed.

Table 5.1 and Figure 5.1 show the register value adjustment specifications, and Figure 5.2 is a timing diagram of calibration.

Table 5.1 Register Value Adjustment Specifications

Pattern	Condition	Adjustment Value
1	$X \geq (Z + L)$	–1
2	$X \leq (Z – L)$	+1
3	$(Z – L) < X < (Z + L)$	As is

X: Measurement result, Z: Theoretical value*1, L: Allowable error range*2

Notes: 1. The following formula is used to calculate the theoretical value:
Theoretical value (Z) = measurement target clock frequency ÷ measurement reference clock frequency (Digits after the decimal point are discarded.)

2. The following formula is used to calculate the allowable range:
Allowable range (L) = Z × 1.001 − Z (Digits after the decimal point are discarded.)

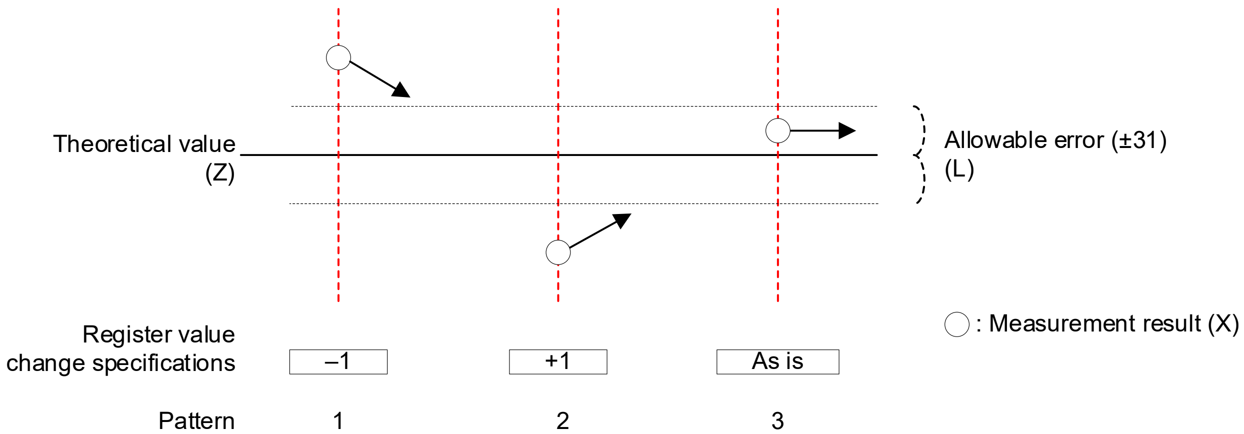


Figure 5.1 Register Value Adjustment Specifications

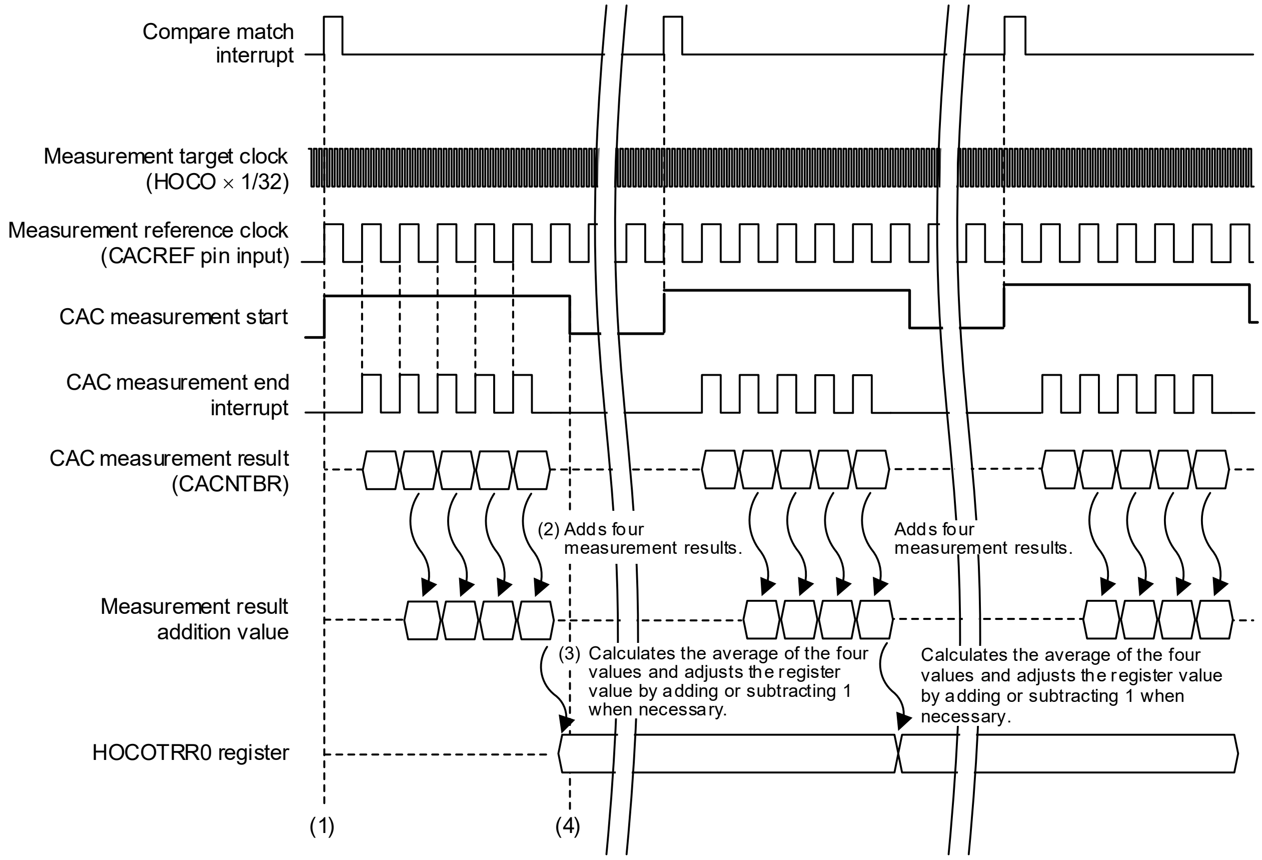


Figure 5.2 Timing Diagram of Calibration

(1) When a compare match interrupt is occurred from CMT, CAC startup settings are made and measurement starts.

(2) Valid edges of the measurement target clock are counted in the period from a rising edge to the next rising edge of the measurement reference clock. When an interrupt is occurred at measurement end, the measurement results are obtained. The measurement results consist of a set of five measurements.

(3) The second to the fifth of the five measurements in the set are averaged, and this value is used to adjust the value of the HOCOTRR0 register.

(4) When adjustment of the HOCO oscillation frequency ends, CAC measurement stops. The HOCO clock is supplied as ICLK, etc., to the CPU and peripheral functions until the next CMT compare match interrupt occurs.

5.2 File Composition

Table 5.2 lists the files used in the sample code. Files generated by the integrated development environment are not included in this table.

Table 5.2 Files Used in the Sample Code

File Name	Outline	Remarks
main.c	Main processing routine	
r_init_stop_module.c	Disable peripheral functions running after a reset	
r_init_stop_module.h	Header file of r_init_stop_module.c	
r_init_port_initialize.c	Initial nonexistent port settings	
r_init_port_initialize.h	Header file of r_init_port_initialize.c	
r_init_clock.c	Initial clock settings	
r_init_clock.h	Header file of r_init_clock.c	
r_cmt_func.c	CMT settings and periodic interrupt handling	
r_cmt_func.h	Header file of r_cmt_func.c	
r_cac_func.c	CAC settings and measurement end interrupt handling	
r_cac_func.h	Header file of r_cac_func.c	
r_mtu_func.c	MTU settings for PCLK output	
r_mtu_func.h	Header file of r_mtu_func.c	

5.3 Option-Setting Memory

Table 5.3 lists the option-setting memory configuration used in the sample code. If necessary, set values optimized for the user system.

Table 5.3 Option-Setting Memory Configuration in Sample Code

Symbol	Address	Setting Value	Description
MDE	FFFF FF80h to FFFF FF83h	FFFF FFFFh	Little endian
OFS1	FFFF FF88h to FFFF FF8Bh	FFFF FFFFh	The voltage monitor 0 reset is disabled after a reset. HOCO oscillation is disabled after a reset.
OFS0	FFFF FF8Ch to FFFF FF8Fh	FFFF FFFFh	The IWDT is stopped after a reset.

5.4 Constants

Table 5.4 and Table 5.5 lists the constants used by the sample code. Note that the clock settings used are those listed under No. 1 in Table 1.2, Clock Selection Example, in the Initial Settings Example application note.

Table 5.4 Constants (User Changeable) Used by Sample Code

Constant Name	Setting Value	Description
CMT_CNT	62,500	Compare match timer count value
CACREF_CLOCK_HZ	32	Measurement reference clock frequency
TARGET_CLOCK_DIV	32	Measurement target clock frequency dividing ratio
ACCEPTABLE_PERCENT	10	Allowable range percentage (in 0.01% increments) relative to the measurement result. A setting of 10 corresponds to an allowable range of 0.1%. If this setting is changed, use a setting value of 5 (0.05%) or greater.
CHECK_CNT	4	Number of measurement results
MTU_ENABLE	1	MTU operation enabled.

Table 5.5 Constants (Non-User Changeable) Used by Sample Code

Constant Name	Setting Value	Description
REG_HOCOTRR	SYSTEM.HOCOTRR0.BYTE	HOCOTRR0 register
HOCO_HZ	32,000,000L	HOCO frequency
TYP_CAC_RESULT	(HOCO_HZ / (TARGET_CLOCK_DIV * CACREF_CLOCK_HZ))	Theoretical measurement value relative to HOCO frequency
MEND_FINISH	0	Frequency measurement is stopped
MEND_START	1	Frequency measurement is started
MEND_CHECK_FINISH	(CHECK_CNT + 1)	Frequency measurement is completed
ACCEPTABLE_RANGE	(((TYP_CAC_RESULT * (10000 + ACCEPTABLE_PERCENT)) / 10000) - TYP_CAC_RESULT)	Allowable error range of the measurement results

5.5 Variables

Table 5.5 lists global variables.

Table 5.6 Global Variables

Type	Variable Name	Description	Used by Function
static uint8_t	buffer_counter	Frequency measurement counter	R_CAC_Init Excep_CAC_MENDF
static uint32_t	result_buffer	Storage buffer for frequency measurement results	R_CAC_Init Excep_CAC_MENDF
static int32_t	result_diff	Storage buffer for result	R_CAC_Init Excep_CAC_MENDF

5.6 Functions

Table 5.6 lists the functions.

Table 5.7 Functions

Function Name	Outline
main	Main processing routine
R_INIT_StopModule	Disable peripheral functions running after a reset
R_INIT_Port_Initialize	Initial nonexistent port settings
R_INIT_Clock	Initial clock settings
R_MTU_Init	MTU settings
R_CMT_Init	CMT settings
Excep_CMT0_CMI0	CMT compare match interrupt handling
R_CAC_Init	CAC settings
Excep_CAC_MENDF	CAC measurement end interrupt handling

5.7 Function Specifications

The following tables list the sample code function specifications.

main	
Outline	Main processing routine
Header	None
Declaration	void main (void)
Description	Calls the following functions: stop processing of active peripheral functions after a reset, nonexistent port initialization, clock initialization, CMT settings, and MTU settings.
Arguments	None
Return Value	None

R_INIT_StopModule	
Outline	Disable peripheral functions running after a reset
Header	r_init_stop_module.h
Declaration	void R_INIT_StopModule(void)
Description	Makes settings to transition to the module stop state.
Arguments	None
Return Value	None
Remarks	In the sample code, no transition to the module stop state occurs. For details of this function, refer to the Initial Settings Example application note.

R_INIT_Port_Initialize	
Outline	Initial nonexistent port settings
Header	r_init_port_initialize.h
Declaration	void R_INIT_Port_Initialize(void)
Description	Makes initial settings to the port direction registers corresponding to the pins of nonexistent port.
Arguments	None
Return Value	None
Remarks	The setting in the sample code (PIN_SIZE=48) is for 48-pin products. When writing in byte units to PDR or PODR registers containing nonexistent ports after this function has been called, set the direction control bits corresponding to the nonexistent ports as described in 17.4, Initialization of Port Direction Register (PDR), in User’s Manual: Hardware. Also, set the port output data storage bits corresponding to ports set to output to 0. For details of this function, refer to the Initial Settings Example application note.

R_INIT_Clock	
Outline	Initial clock settings
Header	r_init_clock.h
Declaration	void R_INIT_Clock(void)
Description	Makes initial clock settings and sets the power control mode.
Arguments	None
Return Value	None
Remarks	The sample code processing sets the HOCO as the system clock and selects high-speed operating mode as the power control mode. For details of this function, refer to the Initial Settings Example application note.

R_MTU_Init	
Outline	MTU settings
Header	r_mtu_func.h
Declaration	void R_MTU_Init(void)
Description	Makes settings for channel 0 of the MTU to output PCLK divided by 4.
Arguments	None
Return Value	None

R_CMT_Init	
Outline	CMT settings
Header	r_cmt_func.h
Declaration	void R_CMT_Init(void)
Description	Makes settings for compare match operation on channel 0 of the CMT.
Arguments	None
Return Value	None

Excep_CMT0_CMI0	
Outline	CMT compare match interrupt handling
Header	r_cmt_func.h
Declaration	static void Excep_CMT0_CMI0 (void)
Description	Periodically makes CAC settings when the CMT compare match interrupt occurs.
Arguments	None
Return Value	None

R_CAC_Init	
Outline	CAC settings
Header	r_cac_func.h
Declaration	void R_CAC_Init(void)
Description	Makes CAC settings.
Arguments	None
Return Value	None

Excep_CAC_MENDF	
Outline	CAC measurement end interrupt handling
Header	r_cac_func.h
Declaration	static void Excep_CAC_MENDF(void)
Description	Adjusts HOCO frequencies based on the measurement results.
Arguments	None
Return Value	None

5.8 Flowcharts

5.8.1 Main Processing

Figure 5.3 shows the main processing.

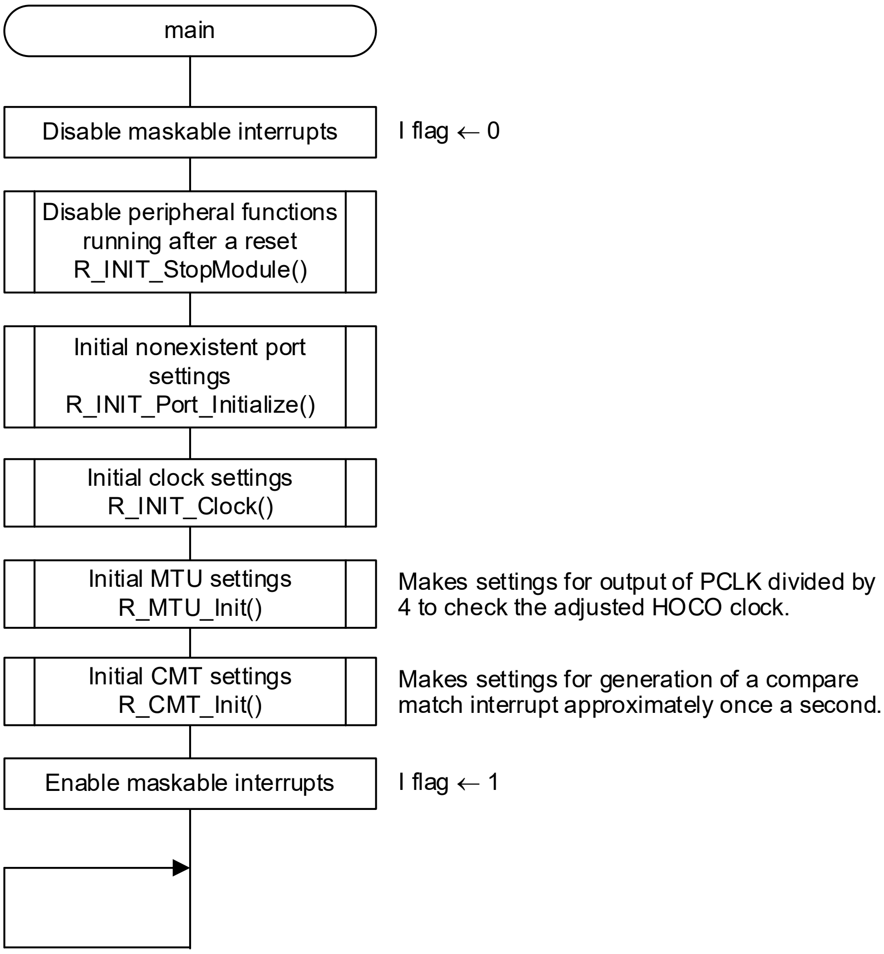
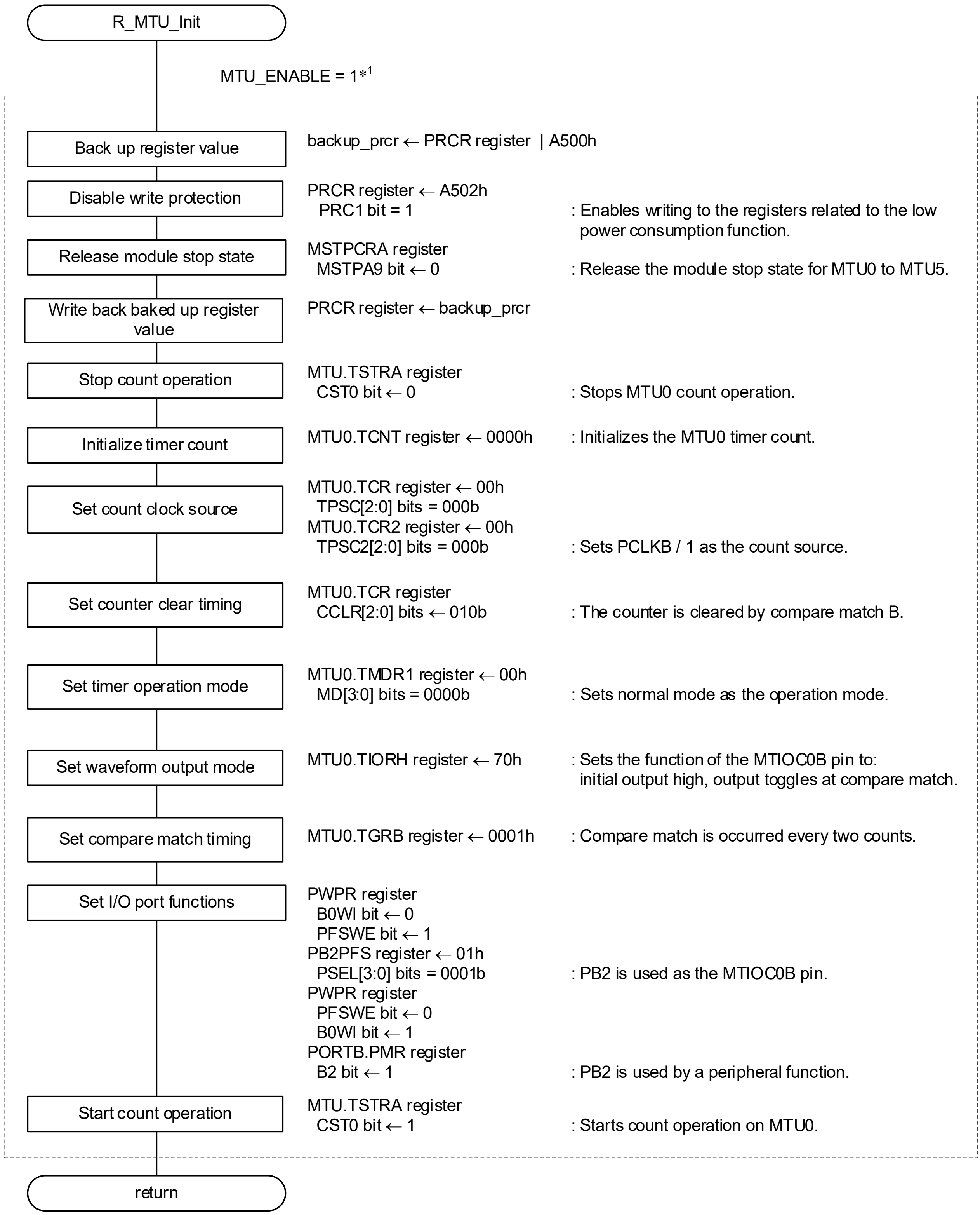


Figure 5.3 Main Processing

5.8.2 MTU Settings

Figure 5.4 is a flowchart of the MTU setting processing.



Note: 1. Change the setting as needed to match the system used.

Figure 5.4 MTU Setting Processing

5.8.3 CMT Settings

Figure 5.5 is a flowchart of the CMT setting processing.

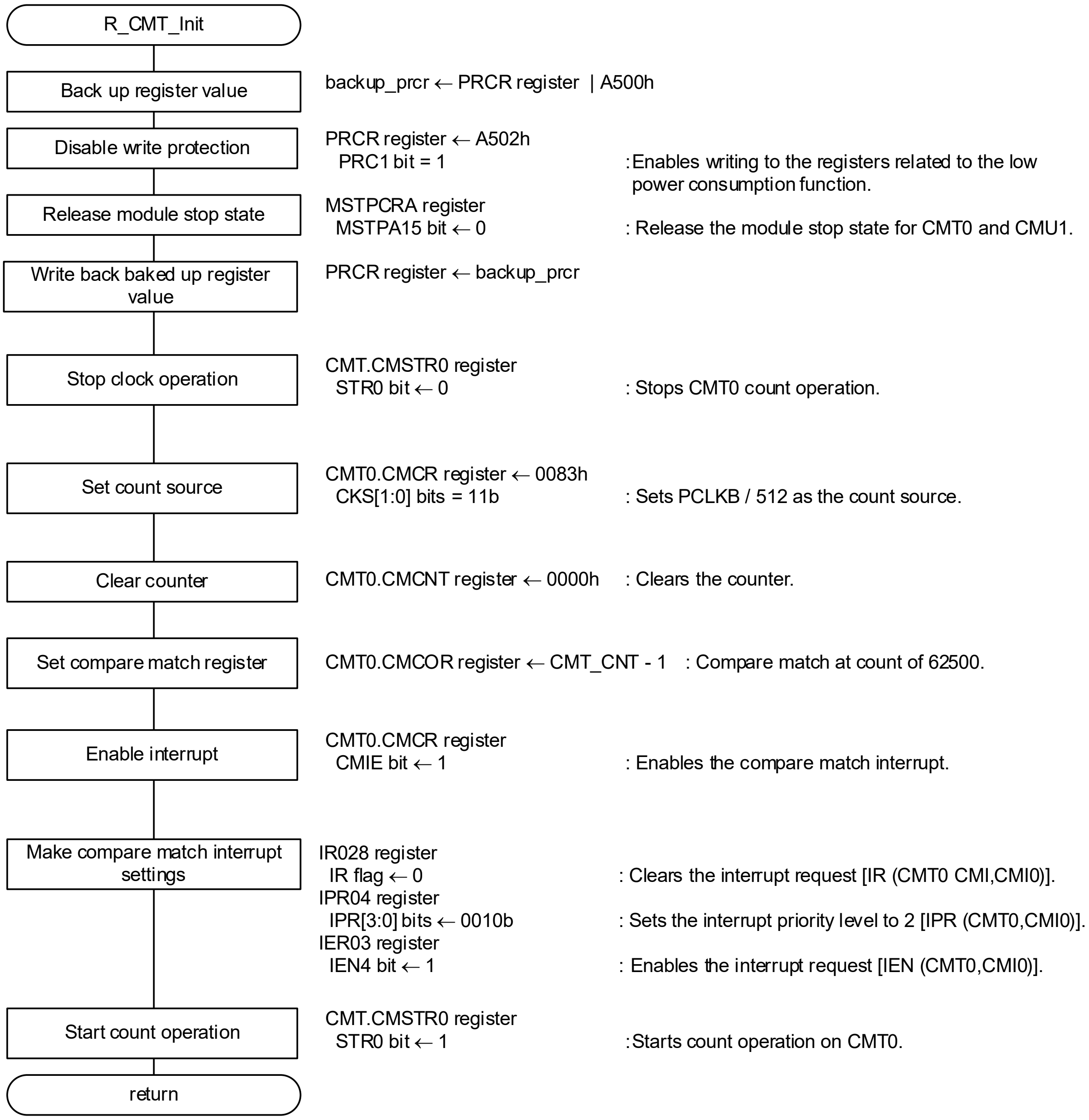


Figure 5.5 CMT Setting Processing

5.8.4 CMT Compare Match Interrupt

Figure 5.6 is a flowchart of CMT compare match interrupt handling.

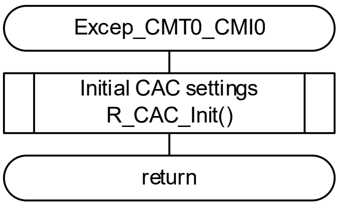
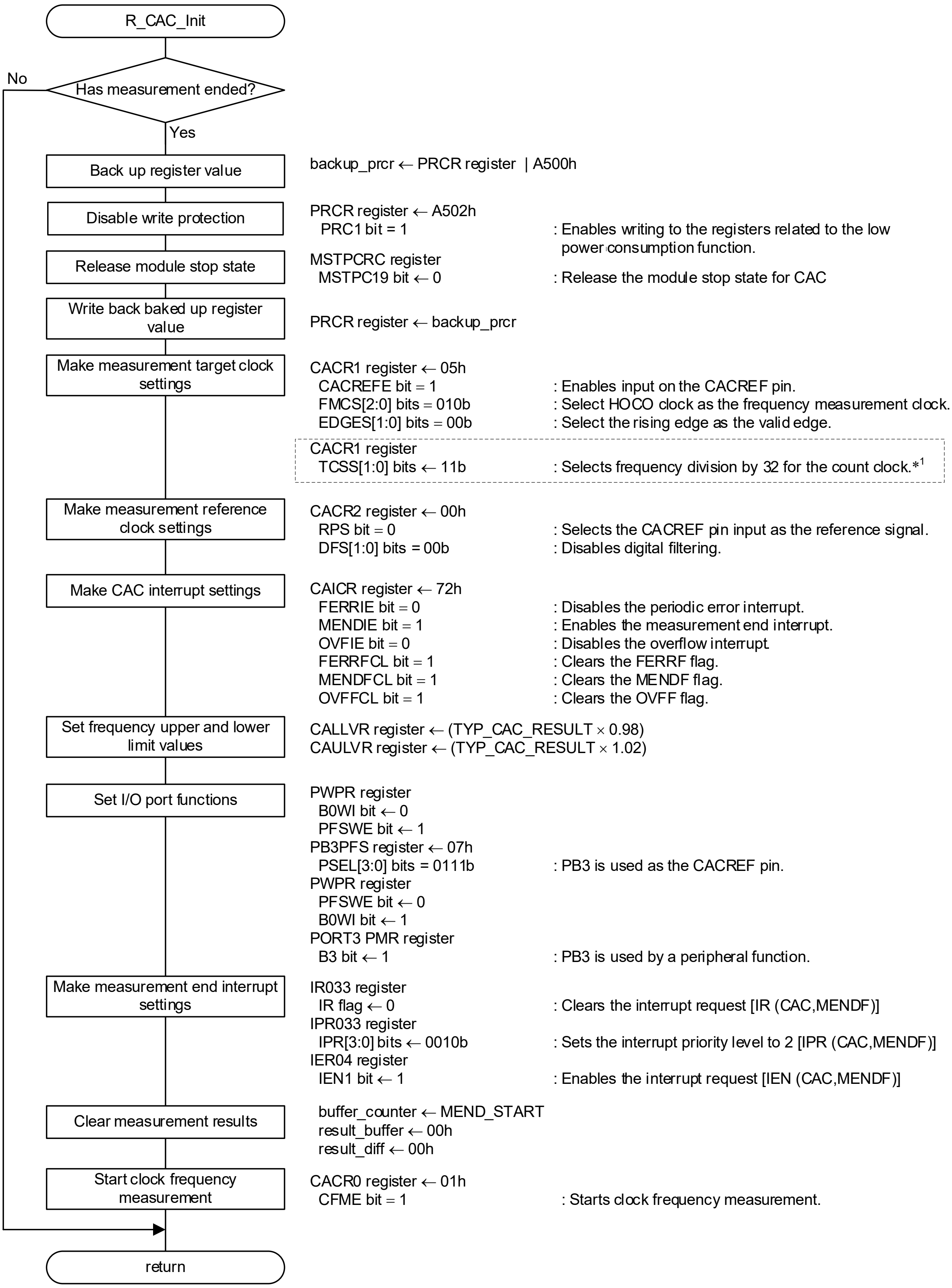


Figure 5.6 CMT Compare Match Interrupt Handling

5.8.5 CAC Settings

Figure 5.7 is a flowchart of the CAC setting processing.



Note: 1. This is the setting value when TARGET_CLOCK_DIV = 32. The setting value of the TCSS[1:0] bits varies according to the setting value of TARGET_CLOCK_DIV. Change the setting of this constant as necessary to match the system used.

Figure 5.7 CAC Setting Processing

5.8.6 CAC Measurement End Interrupt

Figure 5.8 is a flowchart of the CAC measurement end interrupt handling.

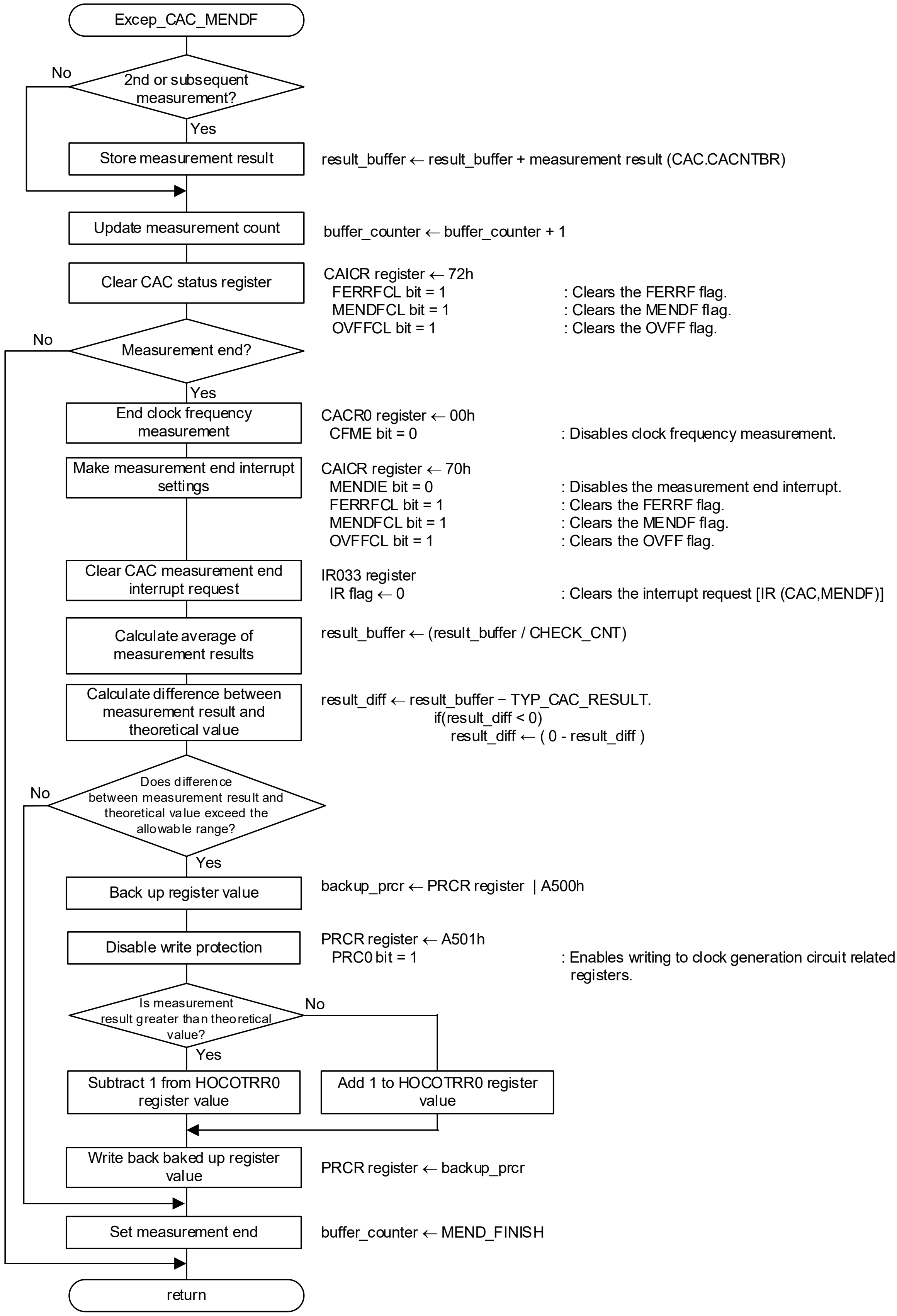


Figure 5.8 CAC Measurement End Interrupt Handling

6. Appendix

6.1 Changing the Measurement Reference Clock Frequency

When using a measurement reference clock frequency other than the default, choose a setting within the range from 32 Hz to 5,000 Hz.

Set the frequency to be used in CACREF_CLOCK_HZ.

Set TARGET_CLOCK_DIV based on the frequency to be used. Refer to Table 6.1 for the appropriate setting value.

If there is more than one possible setting for TARGET_CLOCK_DIV, we recommend using the smallest value.

Table 6.1 TARGET_CLOCK_DIV Setting Values

Measurement Reference Clock Frequency Used		TARGET_CLOCK_DIV Setting Value
Min. Frequency (Hz)	Max. Frequency (Hz)	
600	5,000	1
200	1,200	4
100	600	8
32	160	32

6.2 Changing the Compare Match Timer Count Value

If you decide to change the value of CMT_CNT, make sure to specify a value that does not result in an interrupt occurring while CAC measurement is still in progress.

6.3 Disabling the MTU

Set MTU_ENABLE to 0.

7. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

8. Reference Documents

User’s Manual: Hardware
RX13T Group User’s Manual: Hardware (R01UH0822)
(The latest version can be downloaded from the Renesas Electronics website.)

Technical Update/Technical News
(The latest information can be downloaded from the Renesas Electronics website.)

User’s Manual: Development Tools
RX Family CC-RX Compiler User’s Manual (R20UT3248)
(The latest version can be downloaded from the Renesas Electronics website.)

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Dec. 16, 2019	—	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. 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.

Corporate Headquarters

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

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/.