RX210, RX21A, and RX220 Groups

Pulse Width Measurement Using MTU2a

R01AN1010EJ0111
Rev. 1.11
July 1, 2014

Abstract

This document describes methods to measure a high pulse width when an external trigger is detected using multi-function timer pulse unit 2 (hereinafter referred to as MTU) in the RX210, RX21A, and RX220 Groups.

Products

- RX210, RX21A, and RX220 Groups

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Contents

1. Specifications ... 3

2. Operation Confirmation Conditions ... 4

3. Reference Application Note ... 4

4. Hardware ... 5
   - 4.1 Pins Used ... 5

5. Software ... 6
   - 5.1 Sample Code 1 ... 6
     - 5.1.1 Operation Overview ... 7
     - 5.1.2 File Composition ... 9
     - 5.1.3 Option-Setting Memory ... 9
     - 5.1.4 Constants ... 9
     - 5.1.5 Variables ... 10
     - 5.1.6 Functions ... 10
     - 5.1.7 Function Specifications ... 11
     - 5.1.8 Flowcharts ... 14
   - 5.2 Sample Code 2 ... 19
     - 5.2.1 Operation Overview ... 20
     - 5.2.2 File Composition ... 22
     - 5.2.3 Option-Setting Memory ... 22
     - 5.2.4 Constants ... 22
     - 5.2.5 Variables ... 23
     - 5.2.6 Functions ... 23
     - 5.2.7 Function Specifications ... 24
     - 5.2.8 Flowcharts ... 26

6. Applying This Application Note to the RX21A or RX220 Group ... 30

7. Sample Code ... 31

8. Reference Documents ... 31

1. Specifications

A high pulse width is measured when an external trigger is detected using the MTU input capture function. The high pulse width measurement is taken from the rising edge of an input pulse to the subsequent falling edge.

This application note describes two methods listed in Table 1.1 to measure a high pulse.

Table 1.1 Sample Codes for Measuring a High Pulse

Sample Code	Outline	Remarks
Sample code 1	Input pulses to two pins to measure high pulses.	- Use two pins - Low CPU load
Sample code 2	Use the program to measure high pulses.	- Use a single pin - High CPU load

Table 1.2 lists the Peripheral Function and Its Application, Figure 1.1 shows the Connection Diagram of Sample Code 1, and Figure 1.2 shows the Connection Diagram of Sample Code 2.

Table 1.2 Peripheral Function and Its Application

Peripheral Function	Application
MTU2a channel 1 (hereinafter referred to as MTU1)	Measure a pulse width

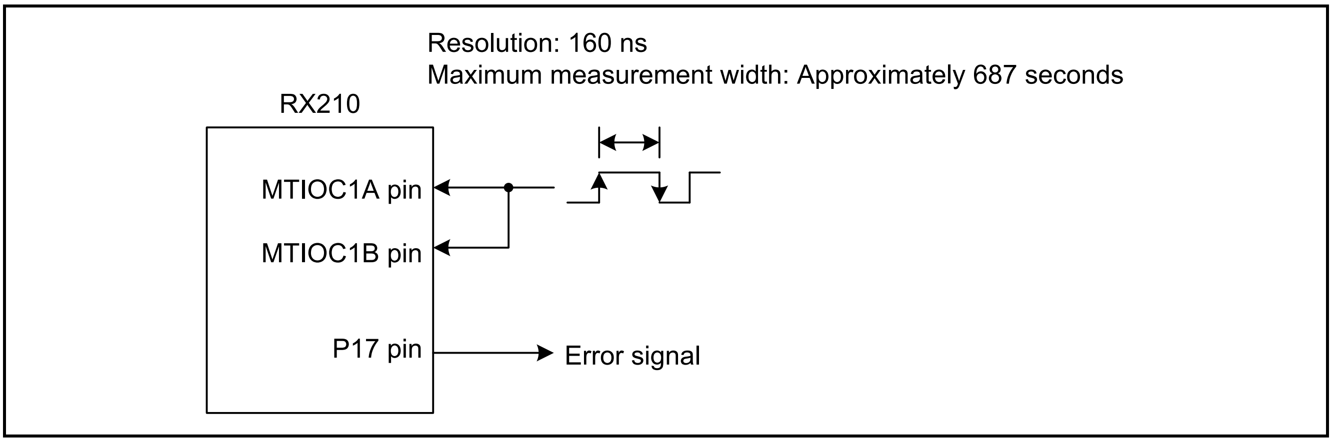

Figure 1.1 Connection Diagram of Sample Code 1

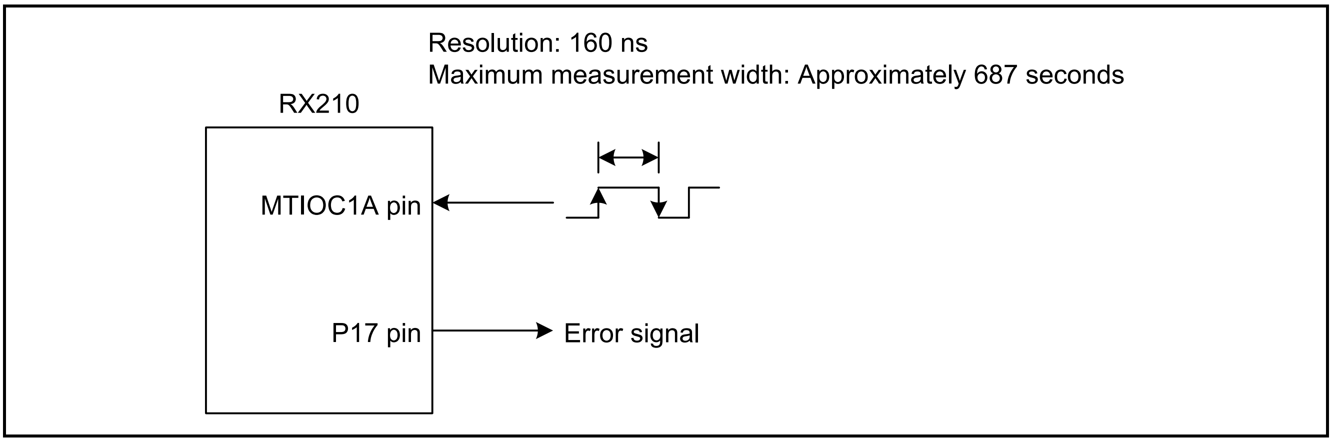

Figure 1.2 Connection Diagram of Sample Code 2

2. Operation Confirmation Conditions

The sample codes accompanying this application note have been run and confirmed under the conditions below.

Table 2.1 Operation Confirmation Conditions

Item	Contents
MCU used	R5F52108ADFP (RX210 Group)
Operating frequencies	- Main clock: 20 MHz - PLL: 100 MHz (main clock divided by 2 and multiplied by 10) - System clock (ICLK): 50 MHz (PLL divided by 2) - Peripheral module clock B (PCLKB): 25 MHz (PLL divided by 4)
Operating voltage	5.0 V
Integrated development environment	Renesas Electronics Corporation High-performance Embedded Workshop Version 4.09.01
C compiler	Renesas Electronics Corporation C/C++ Compiler Package for RX Family V.1.02 Release 01
	Compile options -cpu=rx200 -output=obj="$(CONFIGDIR)\$(FILELEAF).obj" -debug -nologo (The default setting is used in the integrated development environment.)
iodefine.h version	Version 1.2A
Endian	Little endian
Operating mode	Single-chip mode
Processor mode	Supervisor mode
Sample code version	Version 1.10
Board used	Renesas Starter Kit for RX210 (product part no.: R0K505210C000BE)

3. Reference Application Notes

For additional information associated with this document, refer to the following application notes.

- RX210 Group Initial Setting Rev. 2.00 (R01AN1002EJ)
- RX21A Group Initial Setting Rev. 1.10 (R01AN1486EJ)
- RX220 Group Initial Setting Rev. 1.10 (R01AN1494EJ)

The initial setting functions in the reference application notes are used in the sample code in this application note. The revision numbers of the reference application notes are current as of when this application note was made. However the latest version is always recommended. Visit the Renesas Electronics Corporation website to check and download the latest version.

4. Hardware

4.1 Pins Used

Table 4.1 lists the Pins Used and Their Functions – Sample Code 1 and Table 4.2 lists the Pins Used and Their Functions – Sample Code 2.
The pins described here are for 100-pin products. When the product with less than 100-pin is used, select appropriate pins for the product used.

Table 4.1 Pins Used and Their Functions – Sample Code 1

Pin Name	I/O	Function
P20/MTIOC1A	Input	Input a measurement pulse
P21/MTIOC1B	Input	Input a measurement pulse
P17	Output	Output an error signal

Table 4.2 Pins Used and Their Functions – Sample Code 2

Pin Name	I/O	Function
P20/MTIOC1A	Input	Input a measurement pulse
P17	Output	Output an error signal

5. Software

Pulse width measurement starts when the measurement start flag is set to 1. A high pulse width is calculated in the MTU1 input capture B interrupt handler for sample code 1 and in the MTU1 input capture A interrupt handler for sample code 2.

Settings for sample codes 1 and 2 are described in the following sections.

5.1 Sample Code 1

A pulse width from the rising edge of a pulse input to the MTIOC1B pin to the next falling edge is calculated. The number of times the MTU1.TCNT register overflows is counted in the overflow interrupt handler. When the number of overflows exceeds 65,535, an error signal is output and measurement stops.

The pulse width is calculated in the MTU1 input capture B interrupt handler based on the number of overflows and the MTU1.TGRB register value.

Formula for calculating a pulse width: 160 ns × (number of overflows × 10000h + MTU1.TGRB)

Settings are as follows:

MTU1
- Count clock: Rising edge of PCLKB/4 (PCLKB = 25 MHz)
- Operating mode: Normal mode
- Timer general register (TGRB): Use as the input capture register
- MTIOC1A pin: Input capture at the rising edge
- MTIOC1B pin: Input capture at both edges
- Synchronous operation: Not used
- Counter clear: Input capture of TGRB

Interrupts
- Input capture A interrupt (TGIA1)
  Interrupt priority level: 3
  Interrupt source: MTU1.TGRA input capture
- Input capture B interrupt (TGIB1)
  Interrupt priority level: 3
  Interrupt source: MTU1.TGRB input capture
- Overflow interrupt (TCIV1)
  Interrupt priority level: 4
  Interrupt source: MTU1.TCNT overflow

5.1.1 Operation Overview

5.1.1.1 Measuring a Pulse Width

(1) When the TSTR.CST1 bit is set to 1 (count starts), MTU1 starts counting.

(2) When the levels of pins MTIOC1A and MTIOC1B change from low to high, the counter is cleared and the input capture B interrupt request is generated due to an edge input to the MTIOC1B pin, and the input capture A interrupt request is generated due to a rising edge input to the MTIOC1A pin.
The measurement start flag is set to 1 (measurement starts) in the input capture A interrupt handler. Also the overflow counter, overflow interrupt request, and input capture B interrupt request are cleared.

(3) When the MTIOC1B pin level changes from high to low, the MTU1.TCNT register value is transferred to the MTU1.TGRB register and the counter is cleared. At the same time, the MTU1 input capture B interrupt request is generated. In the input capture B interrupt handler, a pulse width is calculated based on the number of times the MTU1.TCNT register overflows and the MTU1.TGRB register value. Then the measurement start flag is cleared.

(4) When the levels of pins MTIOC1A and MTIOC1B change from low to high, the same operation as (2) is performed.

(5) When the MTU1.TCNT register overflows, the overflow interrupt request is generated.
The number of overflows is counted in the overflow interrupt handler.

(6) When the level of the MTIOC1B pin changes from high to low, the same operation as (3) is performed.

Figure 5.1 shows the Timing Diagram of the Pulse Width Measurement. (1) to (5) in the figure correspond to (1) to (5) in the description above.

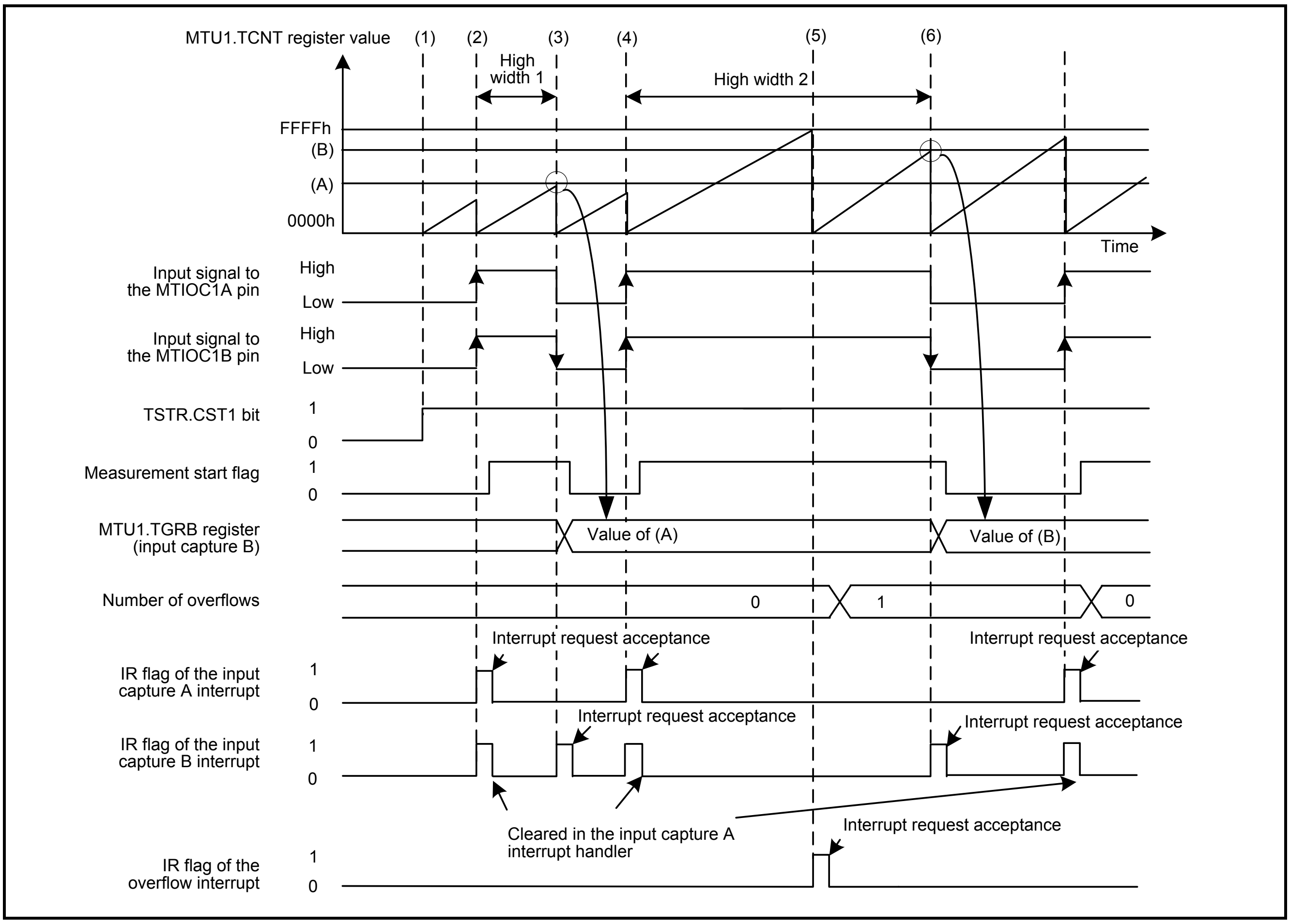

Figure 5.1 Timing Diagram of the Pulse Width Measurement

5.1.1.2 Operation When an Input Capture and Overflow Occur Simultaneously

(1) When a falling edge occurs on the signal input to the MTIOC1B pin while the MTU1.TCNT register value is FFFFh, the MTU1.TCNT register is cleared and the input capture B interrupt request is generated after FFFFh in the MTU1.TCNT register is transferred to the MTU1.TGRB register. If an overflow and counter clear occur simultaneously, the counter clear has higher priority. Thus the overflow interrupt request is not generated.

(2) In the input capture B interrupt handler, the MTU1.TGRB register value is read (FFFFh), and the pulse width is calculated.

(3) When the MTU1.TCNT register value overflows while an interrupt handler (hereinafter referred to as interrupt handler A) other than an overflow interrupt handler and input capture B interrupt handler is being executed, the overflow interrupt handler is delayed.

(4) When a falling edge occurs on the signal input to the MTIOC1B pin while interrupt handler A is being executed, the MTU1.TCNT register value is transferred to the MTU1.TGRB register and the input capture B interrupt request is generated (input capture B interrupt handler is delayed).

(5) When interrupt handler A is completed, the overflow interrupt which has the higher interrupt priority level is executed first. In the overflow interrupt handler, the number of overflows increments by 1. In the input capture B interrupt handler which is subsequently accepted, the pulse width is calculated.

Figure 5.2 shows the Timing Diagram When an Input Capture and Overflow Occur Simultaneously. (1) to (5) in the figure correspond to (1) to (5) in the description above.

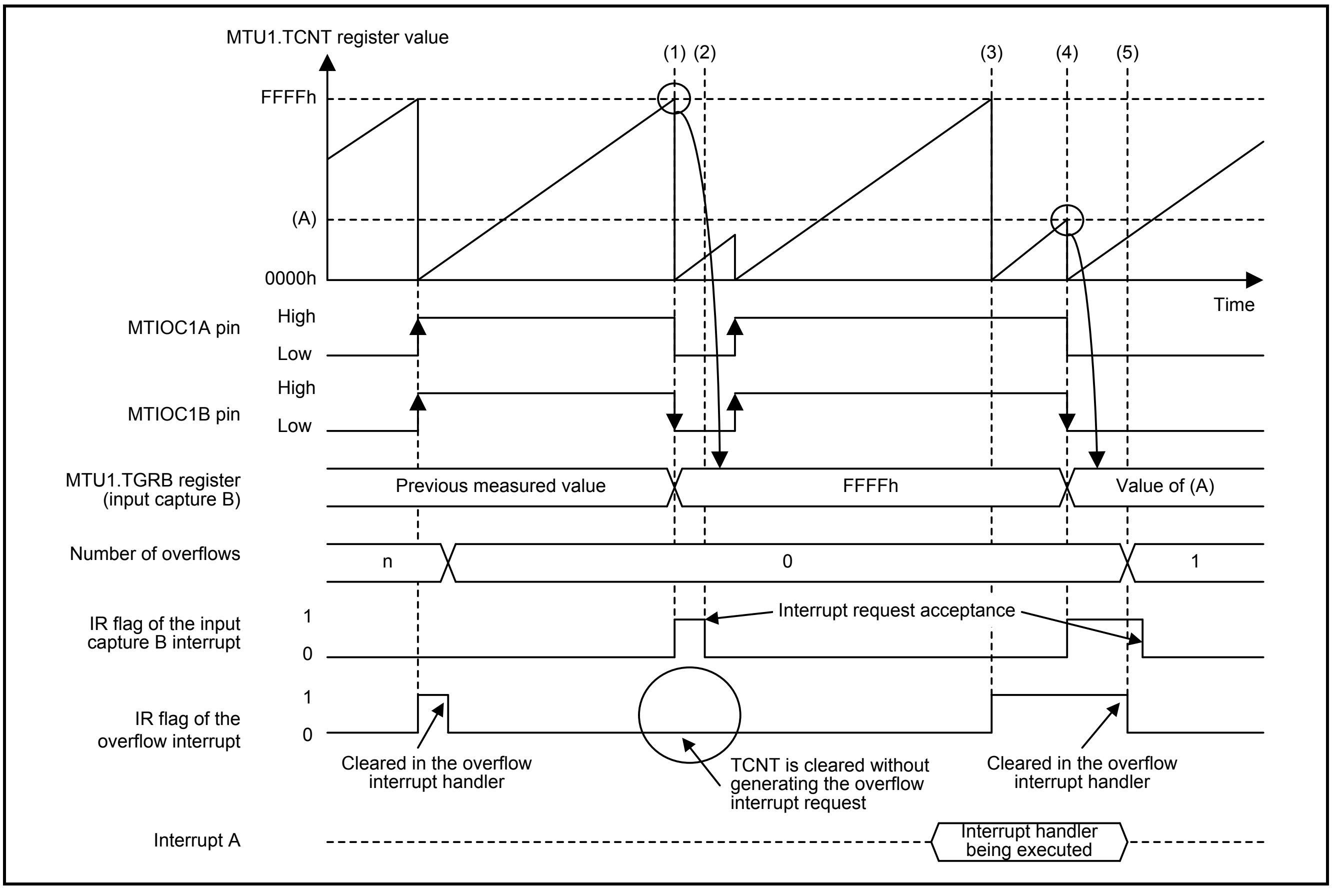

Figure 5.2 Timing Diagram When an Input Capture and Overflow Occur Simultaneously

Notes when embedding the sample codes

When embedding the sample code of this application note in the user system, note the following:

- When an interrupt used in this application note is delayed for a prolonged time due to other interrupt handlers, the sample code may not be executed properly.
- When the measured pulse width is short, the software cannot perform the processes in time and the pulse width cannot be measured properly.

5.1.2 File Composition

Table 5.1 lists the Files Used in the Sample Code. Files generated by the integrated development environment are not included in this table.

Table 5.1 Files Used in the Sample Code

File Name	Outline	Remarks
main.c	Main processing	
r_init_stop_module.c	Stop processing for active peripheral functions after a reset	
r_init_stop_module.h	Header file for r_init_stop_module.c	
r_init_non_existent_port.c	Nonexistent port initialization	
r_init_non_existent_port.h	Header file for r_init_non_existent_port.c	
r_init_clock.c	Clock initialization	
r_init_clock.h	Header file for r_init_clock.c	

5.1.3 Option-Setting Memory

Table 5.2 lists the Option-Setting Memory Configured in the Sample Code. When necessary, set a value suited to the user system.

Table 5.2 Option-Setting Memory Configured in the Sample Code

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

5.1.4 Constants

Table 5.3 lists the Constants Used in the Sample Code.

Table 5.3 Constants Used in the Sample Code

Constant Name	Setting Value	Contents
P_OVF_ERR	PORT1.PODR.BIT.B7	Port output data register for error signal output
PD_OVF_ERR	PORT1.PDR.BIT.B7	Port direction register for error signal output

5.1.5 Variables

Table 5.4 lists the Global Variables.

Table 5.4 Global Variables

Type	Variable Name	Contents	Function Used
unsigned short	mtu1_ovf_cnt	Overflow counter of the MTU1.TCNT register	Excep_MTU1_TCIV1, Excep_MTU1_TGIA1, Excep_MTU1_TGIB1
unsigned long	pulse_cnt	Pulse measurement counter	Excep_MTU1_TGIB1
unsigned char	start_flag	Measurement start flag 0: Measurement stopped 1: Measurement starts	Excep_MTU1_TCIV1, Excep_MTU1_TGIA1, Excep_MTU1_TGIB1
unsigned char	error_flag	Measurement error flag 0: Normal 1: Error	Excep_MTU1_TCIV1, Excep_MTU1_TGIB1

5.1.6 Functions

Table 5.5 lists the Functions Used in the Sample Code.

Table 5.5 Functions Used in the Sample Code

Function Name	Outline
main	Main processing
port_init	Port initialization
R_INIT_StopModule	Stop processing for active peripheral functions after a reset
R_INIT_NonExistentPort	Nonexistent port initialization
R_INIT_Clock	Clock initialization
peripheral_init	Peripheral function initialization
error_proc	Error processing
Excep_MTU1_TGIA1	MTU1 input capture A interrupt handler
Excep_MTU1_TGIB1	MTU1 input capture B interrupt handler
Excep_MTU1_TCIV1	MTU1 overflow interrupt handler

5.1.7 Function Specifications

The following tables list the sample code function specifications.

main	
Outline	Main processing
Header	None
Declaration	void main(void)
Description	Start the count operation for MTU1 after initialization.
Arguments	None
Return Value	None

port_init	
Outline	Port initialization
Header	None
Declaration	void port_init(void)
Description	Initialize ports.
Arguments	None
Return Value	None

R_INIT_StopModule	
Outline	Stop processing for active peripheral functions after a reset
Header	r_init_stop_module.h
Declaration	void R_INIT_StopModule(void)
Description	Configure the setting to enter the module-stop state.
Arguments	None
Return Value	None
Remarks	Transition to the module-stop state is not performed in the sample code. For details on this function, refer to the Initial Setting application note for the product used.

R_INIT_NonExistentPort	
Outline	Nonexistent port initialization
Header	r_init_non_existent_port.h
Declaration	void R_INIT_NonExistentPort(void)
Description	Initialize port direction registers for ports that do not exist in products with less than 100 pins.
Arguments	None
Return Value	None
Remarks	The number of pins in the sample code is set for the 100-pin package (PIN_SIZE=100). After this function is called, when writing in byte units to the PDR registers or PODR registers which have nonexistent ports, set the corresponding bits for nonexistent ports as follows: set the I/O select bits in the PDR registers to 1 and set the output data store bits in the PODR registers to 0. For details on this function, refer to the Initial Setting application note for the product used.

R_INIT_Clock	
Outline	Clock initialization
Header	r_init_clock.h
Declaration	void R_INIT_Clock(void)
Description	Initialize the clock.
Arguments	None
Return Value	None
Remarks	The sample code selects processing which uses PLL as the system clock without using the sub-clock. For details on this function, refer to the Initial Setting application note for the product used.

peripheral_init	
Outline	Peripheral function initialization
Header	None
Declaration	void peripheral_init(void)
Description	Initialize peripheral functions used.
Arguments	None
Return Value	None

error_proc	
Outline	Error processing
Header	None
Declaration	void error_proc(void)
Description	Output an error signal and enter an infinite loop.
Arguments	None
Return Value	None

Excep_MTU1_TGIA1	
Outline	MTU1 input capture A interrupt handler
Header	None
Declaration	void Excep_MTU1_TGIA1(void)
Description	Set the measurement start flag to 1 (measurement starts) and start pulse width calculation. Also clear the input capture B interrupt request and the overflow interrupt request, and reset the overflow counter.
Arguments	None
Return Value	None

Excep_MTU1_TGIB1	
Outline	MTU1 input capture B interrupt handler
Header	None
Declaration	void Excep_MTU1_TGIB1(void)
Description	When the measurement start flag is 1 (measurement starts), calculate the pulse width. The measurement start flag is cleared.
Arguments	None
Return Value	None

Excep_MTU1_TCIV1	
Outline	MTU1 overflow interrupt handler
Header	None
Declaration	void Excep_MTU1_TCIV1 (void)
Description	When the measurement start flag is 1 (measurement starts), the number of overflows is counted. When the number of overflows exceeds 65,535, the MCU enters error processing.
Arguments	None
Return Value	None

5.1.8 Flowcharts

5.1.8.1 Main Processing

Figure 5.3 shows the Main Processing.

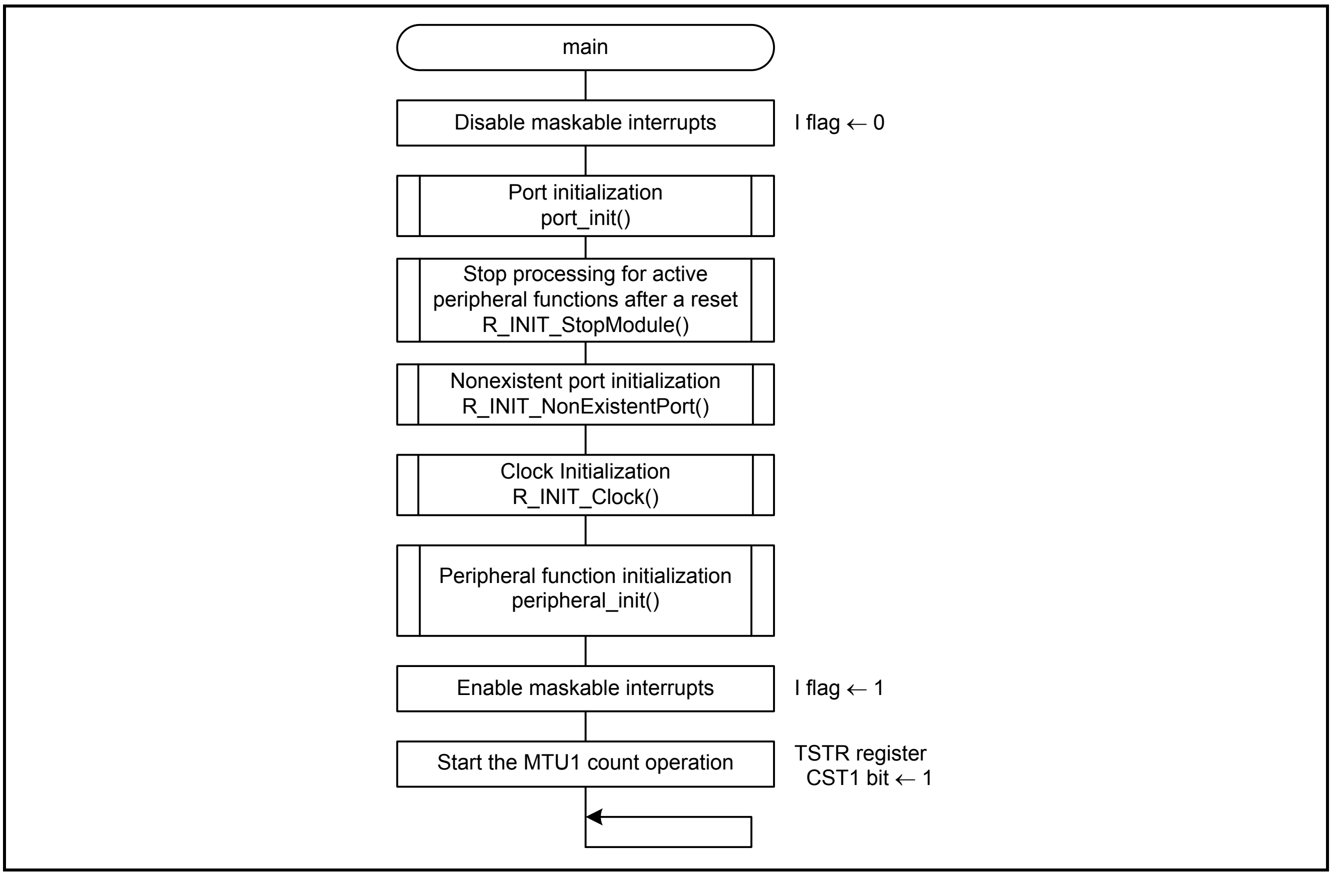

Figure 5.3 Main Processing

5.1.8.2 Port Initialization

Figure 5.4 shows the Port Initialization.

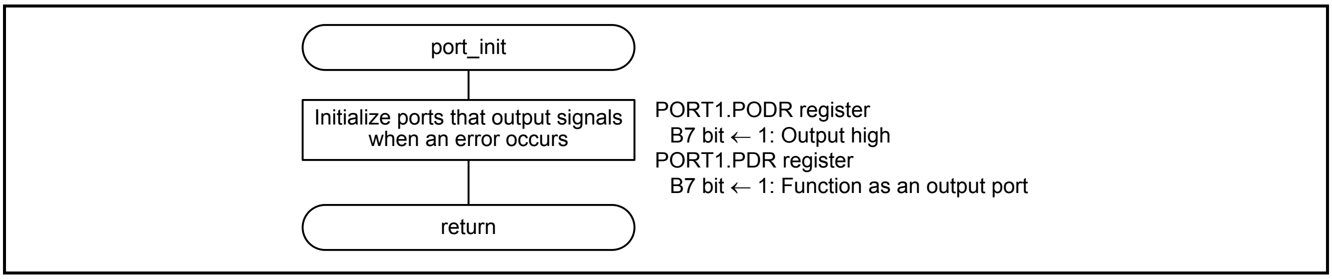

Figure 5.4 Port Initialization

5.1.8.3 Peripheral Function Initialization

Figure 5.5 and Figure 5.6 show the Peripheral Function Initialization.

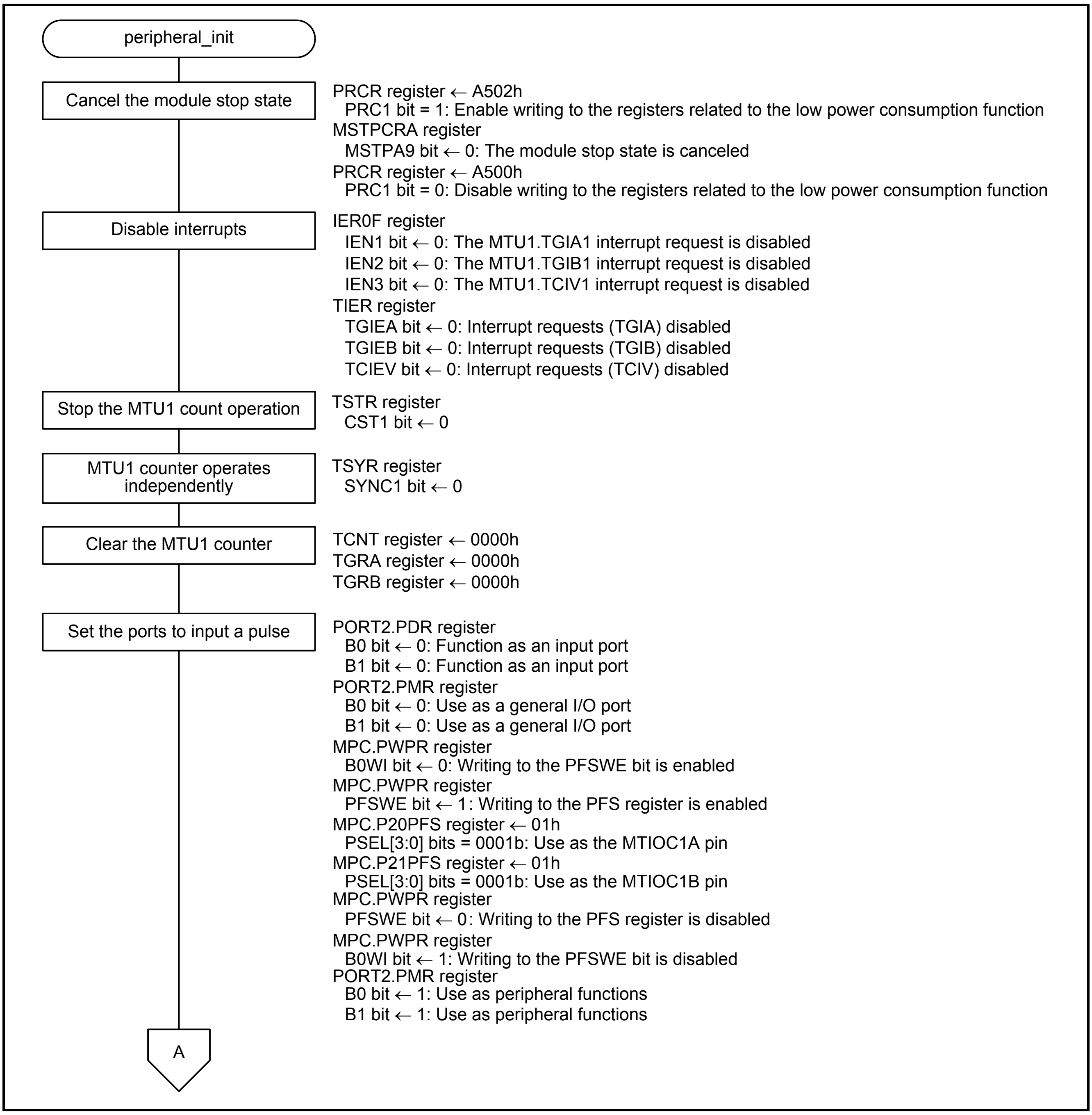

Figure 5.5 Peripheral Function Initialization (1/2)

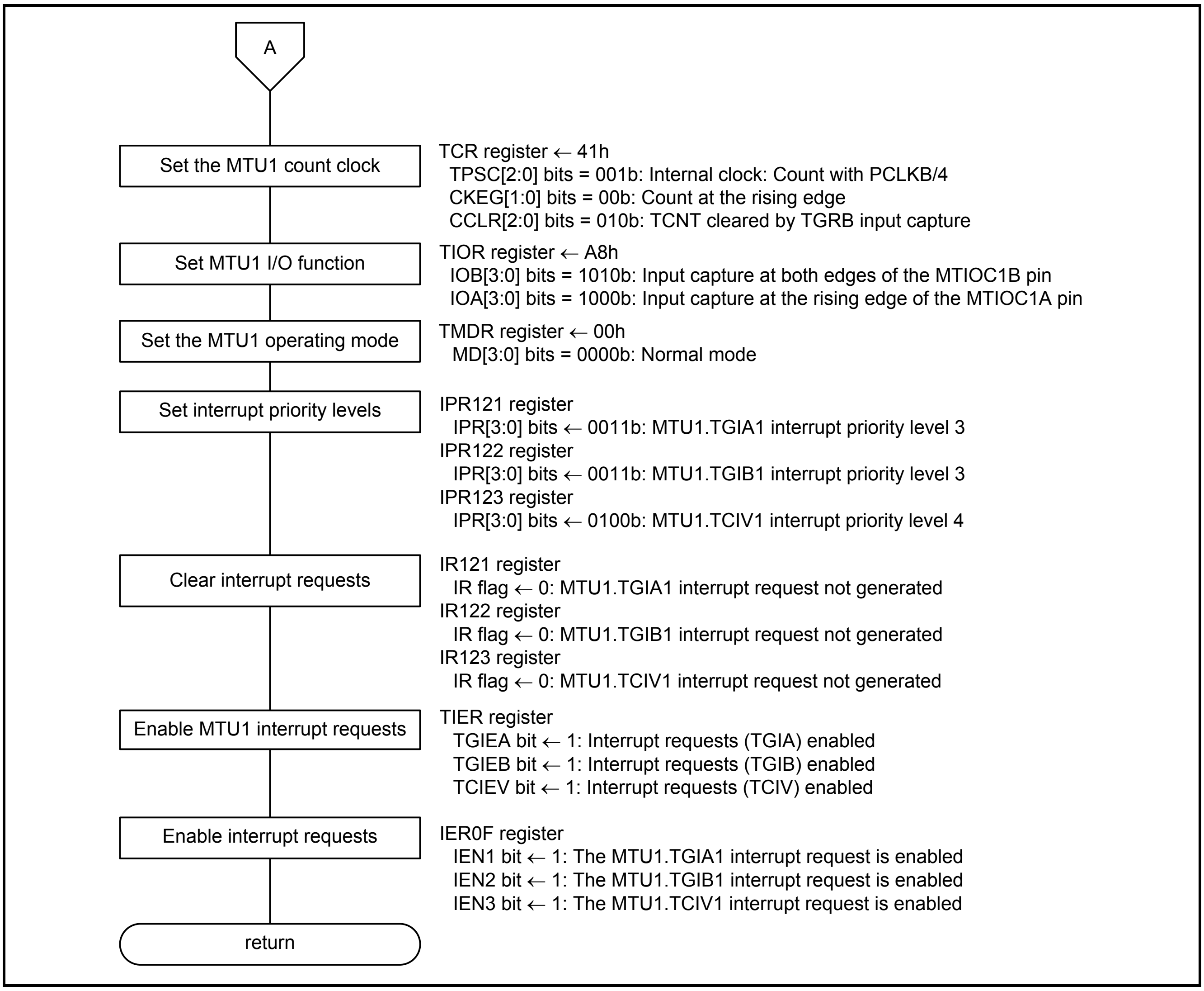

Figure 5.6 Peripheral Function Initialization (2/2)

5.1.8.4 Error Processing

Figure 5.7 shows the Error Processing.

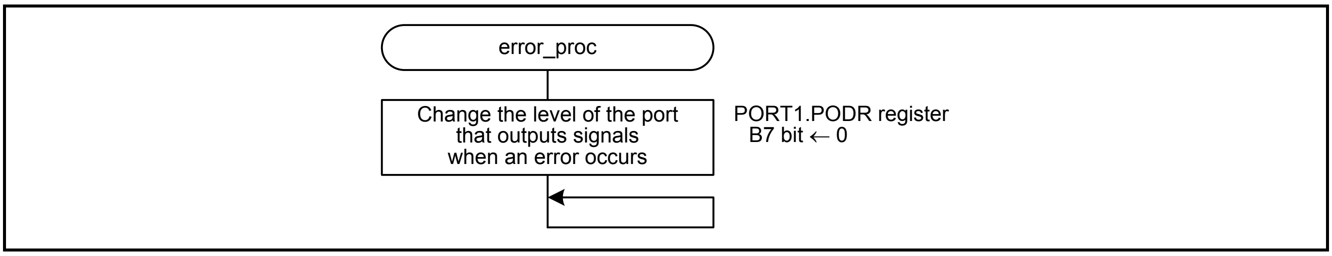

Figure 5.7 Error Processing

5.1.8.5 MTU1 Input Capture A Interrupt Handler

Figure 5.8 shows the MTU1 Input Capture A Interrupt Handler.

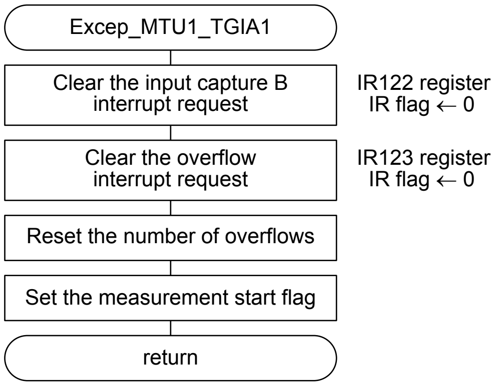

Figure 5.8 MTU1 Input Capture A Interrupt Handler

5.1.8.6 MTU1 Input Capture B Interrupt Handler

Figure 5.9 shows the MTU1 Input Capture B Interrupt Handler.

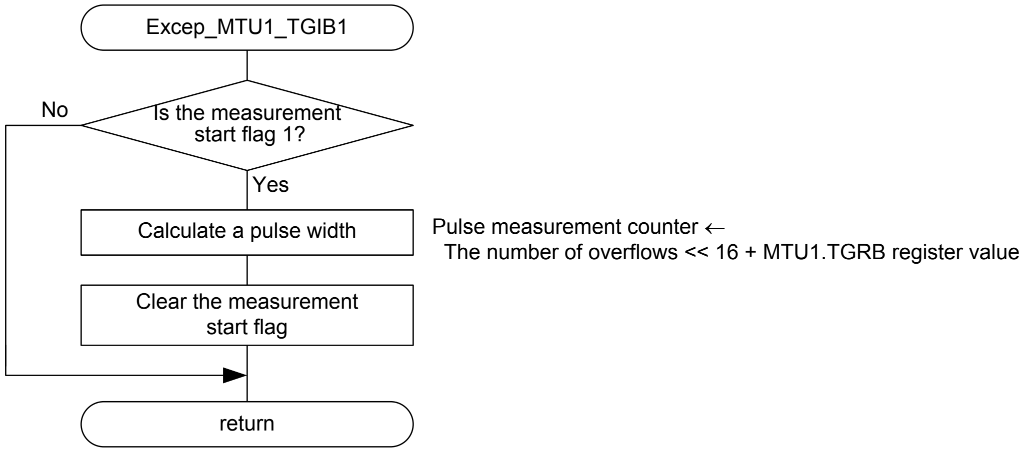

Figure 5.9 MTU1 Input Capture B Interrupt Handler

5.1.8.7 MTU1 Overflow Interrupt Handler

Figure 5.10 shows the MTU1 Overflow Interrupt Handler.

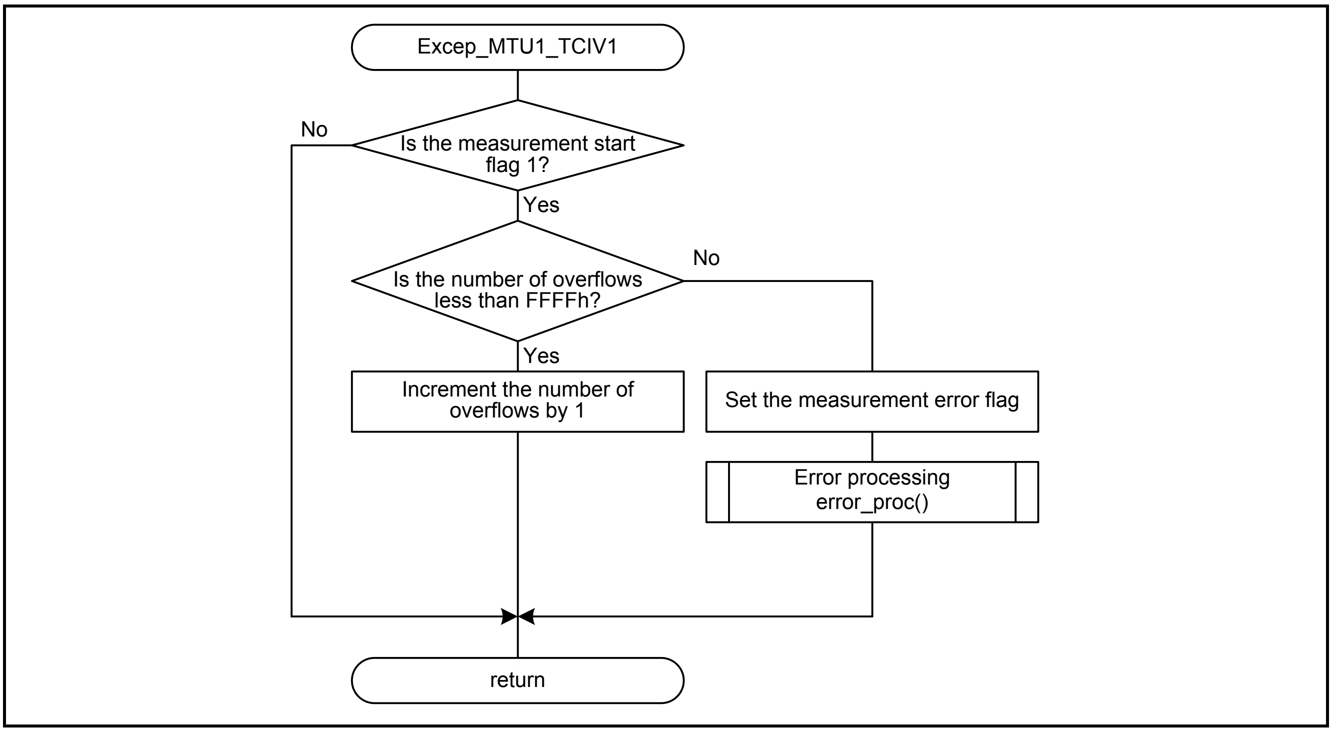

Figure 5.10 MTU1 Overflow Interrupt Handler

5.2 Sample Code 2

A pulse width from the rising edge of a pulse input to the MTIOC1A pin to the next falling edge is calculated. The number of times the MTU1.TCNT register overflows is counted in the overflow interrupt handler. When the number of overflows exceeds 65,535, an error signal is output and measurement stops.

The pulse width is calculated in the MTU1 input capture A interrupt handler based on the number of overflows and the MTU1.TGRA register value.

Formula for calculating the pulse width: 160 ns × (number of overflows × 10000h + MTU1.TGRA)

Settings are as follows:

MTU1
- Count clock: Rising edge of PCLKB/4 (PCLKB = 25 MHz)
- Operating mode: Normal mode
- Timer general register (TGRA): Use as the input capture register
- MTIOC1A pin: Input capture at both edges
- Synchronous operation: Not used
- Counter clear: Input capture of TGRA
- Noise filter: Noise filter of the MTIOC1A pin enabled
- Noise filter clock: PCLKB/1 (PCLKB = 25 MHz)

Interrupts
- Input capture A interrupt (TGIA1)
  - Interrupt priority level: 3
  - Interrupt source: MTU1.TGRA input capture
- Overflow interrupt (TCIV1)
  - Interrupt priority level: 4
  - Interrupt source: MTU1.TCNT overflow

5.2.1 Operation Overview

5.2.1.1 Measuring a Pulse Width

(1) When the TSTR.CST1 bit is set to 1 (count starts), MTU1 starts counting.

(2) When an edge occurs on the signal input to the MTIOC1A pin, the MTU1.TCNT register value is transferred to the MTU1.TGRA register and the counter is cleared. At the same time, the MTU1 input capture A interrupt request is generated.
The MTIOC1A pin status is verified in the input capture A interrupt handler. If the status is high, the software determines that the high pulse width measurement is started. Then the measurement start flag is set to 1 (measurement starts) and the number of overflows is cleared.

(3) When an edge occurs on the signal input to the MTIOC1A pin again, the MTU1 input capture A interrupt request is generated. The MTIOC1A pin status is verified in the input capture A interrupt handler. If the status is low, the software determines that the high pulse width measurement is completed. Then a pulse width is calculated based on the number of overflows of the MTU1.TCNT register and the MTU1.TGRA register value. The measurement start flag is cleared.

(4) When a rising edge occurs on the signal input to the MTIOC1A pin again, the same operation as (2) is performed.

(5) When the MTU1.TCNT register overflows, the overflow interrupt request is generated.
The number of overflows is counted in the overflow interrupt handler.

(6) When a rising edge occurs on the signal input to the MTIOC1A pin again, the same operation as (3) is performed.

Figure 5.11 shows the Timing Diagram of the Pulse Width Measurement. (1) to (6) in the figure correspond to (1) to (6) in the description above.

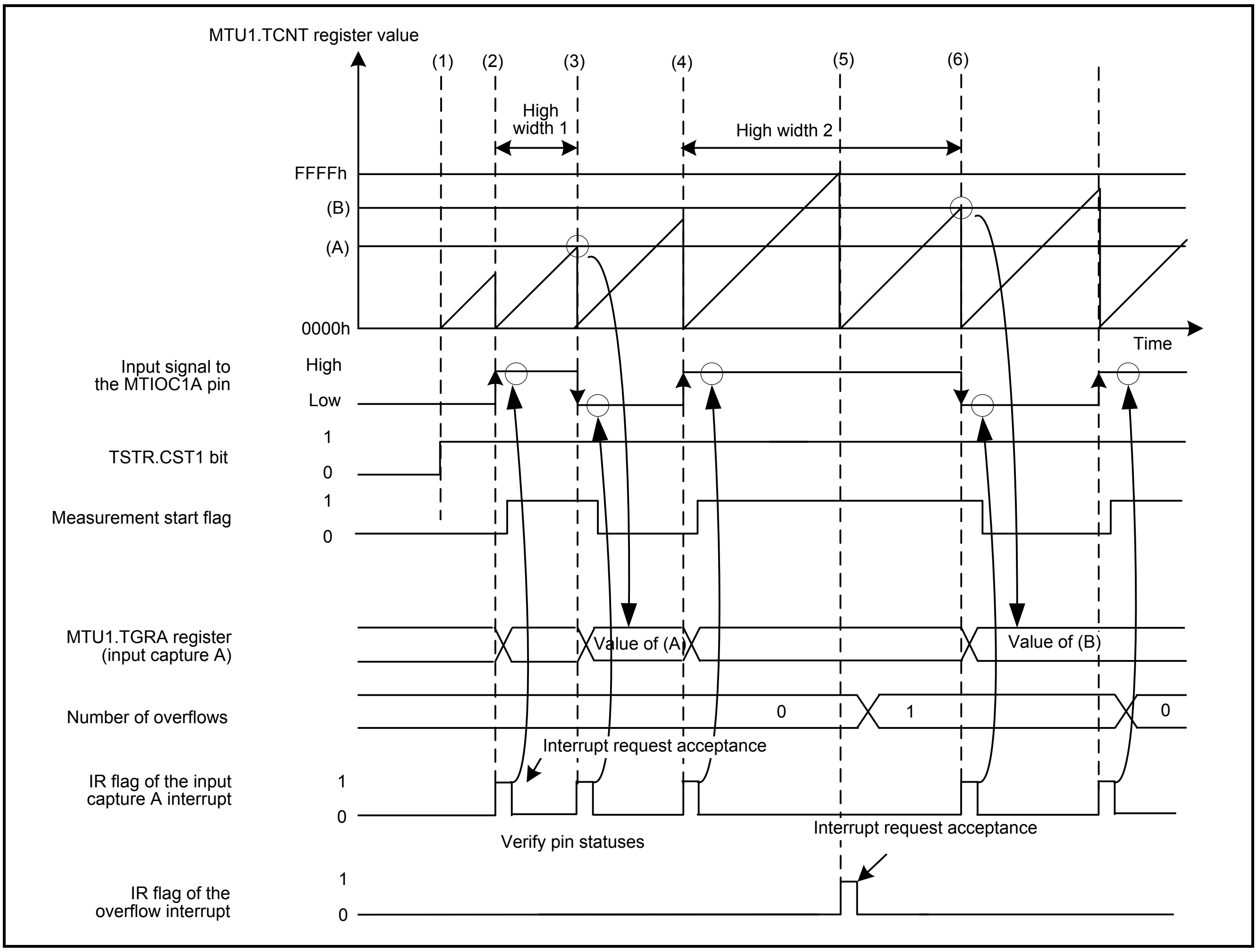

Figure 5.11 Timing Diagram of the Pulse Width Measurement

5.2.1.2 Operation When an Input Capture and Overflow Occur Simultaneously

(1) When a falling edge occurs on the signal input to the MTIOC1A pin while the MTU1.TCNT register value is FFFFh, the MTU1.TCNT register is cleared and the input capture A interrupt request is generated after FFFFh in the MTU1.TCNT register is transferred to the MTU1.TGRA register. If an overflow and counter clear occur simultaneously, the counter clear has a higher priority. Thus the overflow interrupt request is not generated.

(2) In the input capture A interrupt handler, the MTU1.TGRA register value (FFFFh) is read, and the pulse width is calculated.

(3) When the MTU1.TCNT register value overflows while an interrupt handler (hereinafter referred to as interrupt handler A) other than an overflow interrupt handler and input capture A interrupt handler is being executed, the overflow interrupt handler is delayed.

(4) When a falling edge occurs on the signal input to the MTIOC1A pin while interrupt handler A is being executed, the MTU1.TCNT register value is transferred to the MTU1.TGRA register and an input capture A interrupt request is generated (input capture A interrupt handler is delayed).

(5) When interrupt handler A is completed, the overflow interrupt which has a higher interrupt priority level is executed first. In the overflow interrupt handler, the number of overflows increments by 1. In the input capture A interrupt handler which is subsequently accepted, the pulse width is calculated.

Figure 5.12 shows the Timing Diagram When an Input Capture and Overflow Occur Simultaneously. (1) to (5) in the figure correspond to (1) to (5) in the description above.

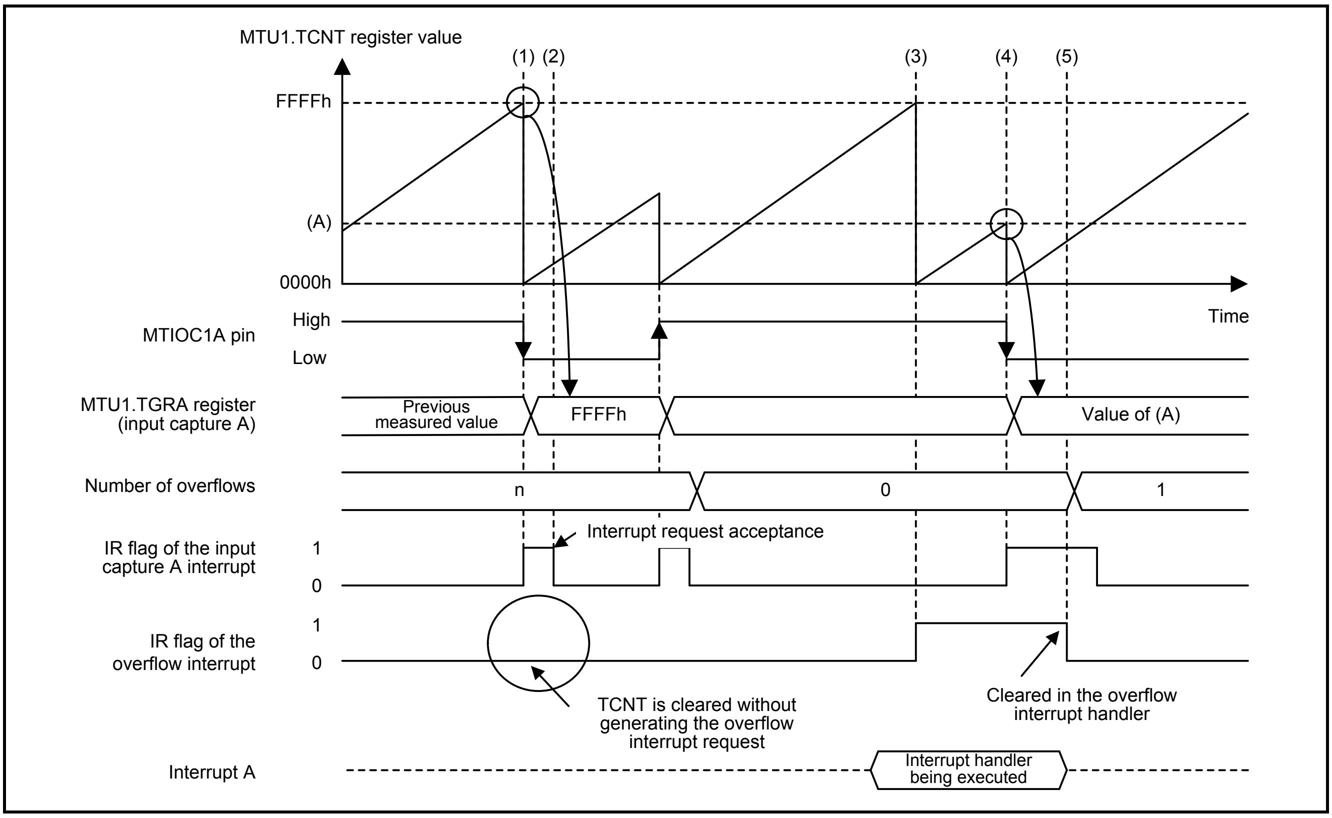

Figure 5.12 Timing Diagram When an Input Capture and Overflow Occur Simultaneously

Notes when embedding the sample codes

When embedding the sample code of this application note in the user system, note the following:

- When an interrupt used in this application note is delayed for a prolonged time due to other interrupt handlers, the sample code may not be executed properly.
- When the measured pulse width is short, the software cannot perform the processes in time and the pulse width cannot be measured properly.

5.2.2 File Composition

Table 5.6 lists the Files Used in the Sample Code. Files generated by the integrated development environment are not included in this table.

Table 5.6 Files Used in the Sample Code

File Name	Outline	Remarks
main.c	Main processing	
r_init_stop_module.c	Stop processing for active peripheral functions after a reset	
r_init_stop_module.h	Header file for r_init_stop_module.c	
r_init_non_existent_port.c	Nonexistent port initialization	
r_init_non_existent_port.h	Header file for r_init_non_existent_port.c	
r_init_clock.c	Clock initialization	
r_init_clock.h	Header file for r_init_clock.c	

5.2.3 Option-Setting Memory

Table 5.7 lists the Option-Setting Memory Configured in the Sample Code. When necessary, set a value suited to the user system.

Table 5.7 Option-Setting Memory Configured in the Sample Code

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

5.2.4 Constants

Table 5.8 lists the Constants Used in the Sample Code.

Table 5.8 Constants Used in the Sample Code

Constant Name	Setting Value	Contents
P_OVF_ERR	PORT1.PODR.BIT.B7	Port output data register for error signal output
PD_OVF_ERR	PORT1.PDR.BIT.B7	Port direction register for error signal output
PI_MTIOC1A	PORT2.PIDR.BIT.B0	Port input data register for MTU1.MTIOC1A
HIGH	1	Port input data is high
LOW	0	Port input data is low

5.2.5 Variables

Table 5.9 lists the Global Variables.

Table 5.9 Global Variables

Type	Variable Name	Contents	Function Used
unsigned short	mtu1_ovf_cnt	Overflow counter of the MTU1.TCNT register	Excep_MTU1_TCIV1, Excep_MTU1_TGIA1
unsigned long	pulse_cnt	Pulse measurement counter	Excep_MTU1_TGIA1
unsigned char	start_flag	Measurement start flag 0: Measurement stopped 1: Measurement starts	Excep_MTU1_TCIV1, Excep_MTU1_TGIA1
unsigned char	error_flag	Measurement error flag 0: Normal 1: Error	Excep_MTU1_TCIV1, Excep_MTU1_TGIA1

5.2.6 Functions

Table 5.10 lists the Functions Used in the Sample Code.

Table 5.10 Functions Used in the Sample Code

Function Name	Outline
main	Main processing
port_init	Port initialization
R_INIT_StopModule	Stop processing for active peripheral functions after a reset
R_INIT_NonExistentPort	Nonexistent port initialization
R_INIT_Clock	Clock initialization
peripheral_init	Peripheral function initialization
error_proc	Error processing
Excep_MTU1_TGIA1	MTU1 input capture A interrupt handler
Excep_MTU1_TCIV1	MTU1 overflow interrupt handler

5.2.7 Function Specifications

The following tables list the sample code function specifications.

main	
Outline	Main processing
Header	None
Declaration	void main(void)
Description	Start the count operation for MTU1 after initialization.
Arguments	None
Return Value	None

port_init	
Outline	Port initialization
Header	None
Declaration	void port_init(void)
Description	Initialize ports.
Arguments	None
Return Value	None

R_INIT_StopModule	
Outline	Stop processing for active peripheral functions after a reset
Header	r_init_stop_module.h
Declaration	void R_INIT_StopModule(void)
Description	Configure the setting to enter the module-stop state.
Arguments	None
Return Value	None
Remarks	Transition to the module-stop state is not performed in the sample code. For details on this function, refer to the Initial Setting application note for the product used.

R_INIT_NonExistentPort	
Outline	Nonexistent port initialization
Header	r_init_non_existent_port.h
Declaration	void R_INIT_NonExistentPort(void)
Description	Initialize port direction registers for ports that do not exist in products with less than 100 pins.
Arguments	None
Return Value	None
Remarks	The number of pins in the sample code is set for the 100-pin package (PIN_SIZE=100). After this function is called, when writing in byte units to the PDR registers or PODR registers which have nonexistent ports, set the corresponding bits for nonexistent ports as follows: set the I/O select bits in the PDR registers to 1 and set the output data store bits in the PODR registers to 0. For details on this function, refer to the Initial Setting application note for the product used.

R_INIT_Clock	
Outline	Clock initialization
Header	r_init_clock.h
Declaration	void R_INIT_Clock(void)
Description	Initialize the clock.
Arguments	None
Return Value	None
Remarks	The sample code selects processing which uses PLL as the system clock without using the sub-clock. For details on this function, refer to the Initial Setting application note for the product used.

peripheral_init	
Outline	Peripheral function initialization
Header	None
Declaration	void peripheral_init(void)
Description	Initialize peripheral functions used.
Arguments	None
Return Value	None

error_proc	
Outline	Error processing
Header	None
Declaration	void error_proc(void)
Description	Output an error signal and enter an infinite loop.
Arguments	None
Return Value	None

Excep_MTU1_TGIA1	
Outline	MTU1 input capture A interrupt handler
Header	None
Declaration	void Excep_MTU1_TGIA1(void)
Description	When the status of the MTIOC1A pin is high, set the measurement start flag to 1 (measurement starts) and start pulse width calculation. Also clear the overflow counter. When the status of the MTIOC1A pin is low, calculate a pulse width and clear the measurement start flag.
Arguments	None
Return Value	None

Excep_MTU1_TCIV1	
Outline	MTU1 overflow interrupt handler
Header	None
Declaration	void Excep_MTU1_TCIV1 (void)
Description	When the measurement start flag is 1 (measurement starts), the number of overflows is counted. When the number of overflows exceeds 65,535, the MCU enters error processing.
Arguments	None
Return Value	None

5.2.8 Flowcharts

5.2.8.1 Main Processing

Figure 5.13 shows the Main Processing.

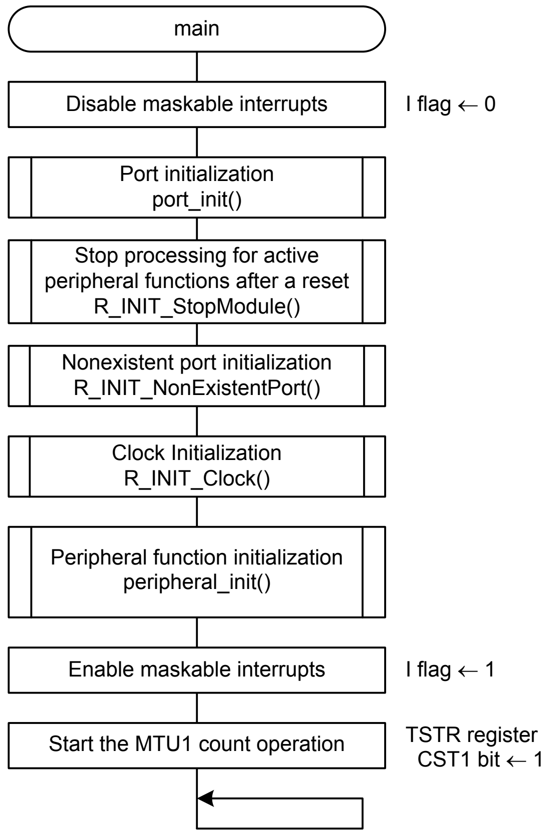

Figure 5.13 Main Processing

5.2.8.2 Port Initialization

Figure 5.14 shows the Port Initialization.

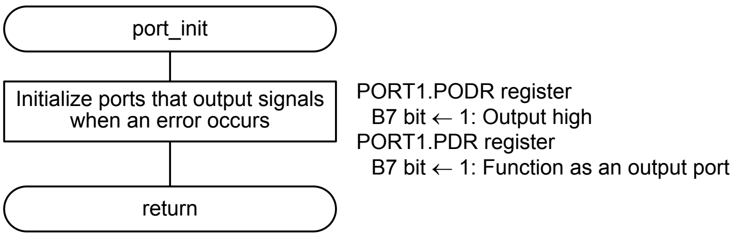

Figure 5.14 Port Initialization

5.2.8.3 Peripheral Function Initialization

Figure 5.15 shows the Peripheral Function Initialization.

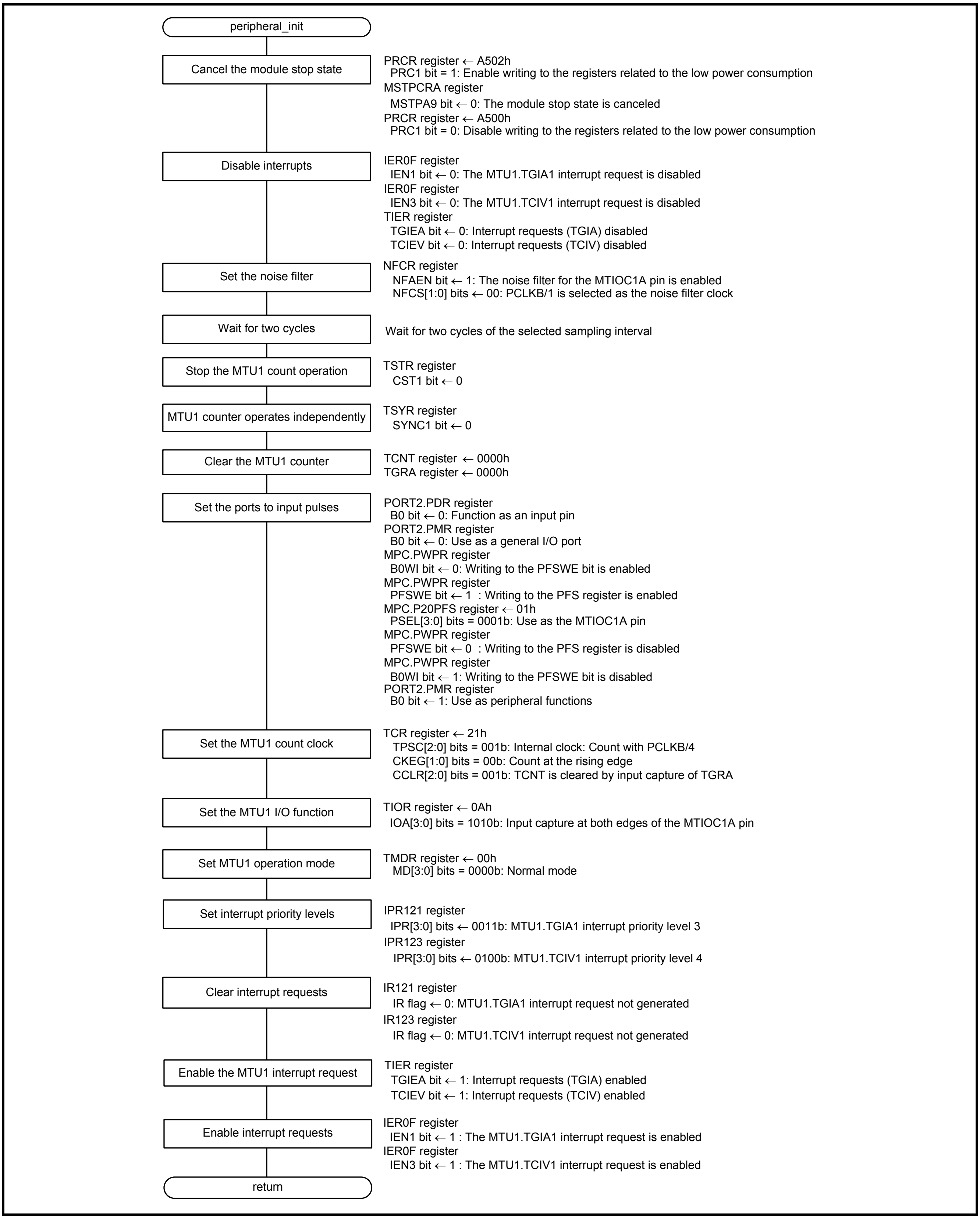

Figure 5.15 Peripheral Function Initialization

5.2.8.4 Error Processing

Figure 5.16 shows the Error Processing.

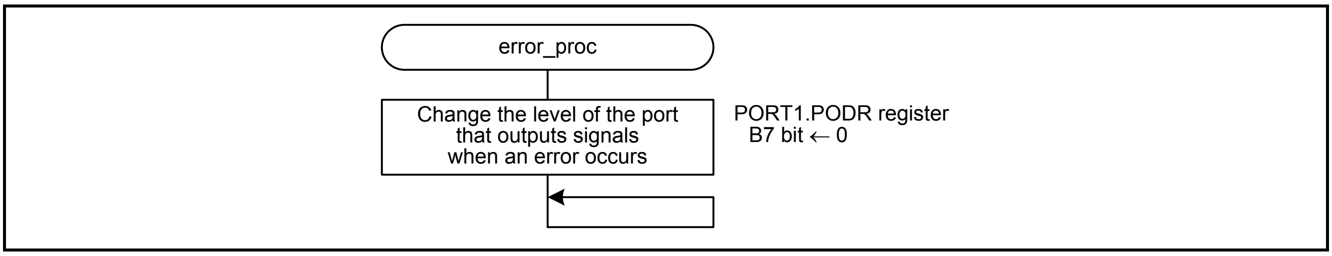

Figure 5.16 Error Processing

5.2.8.5 MTU1 Input Capture A Interrupt Handler

Figure 5.17 shows the MTU1 Input Capture A Interrupt Handler.

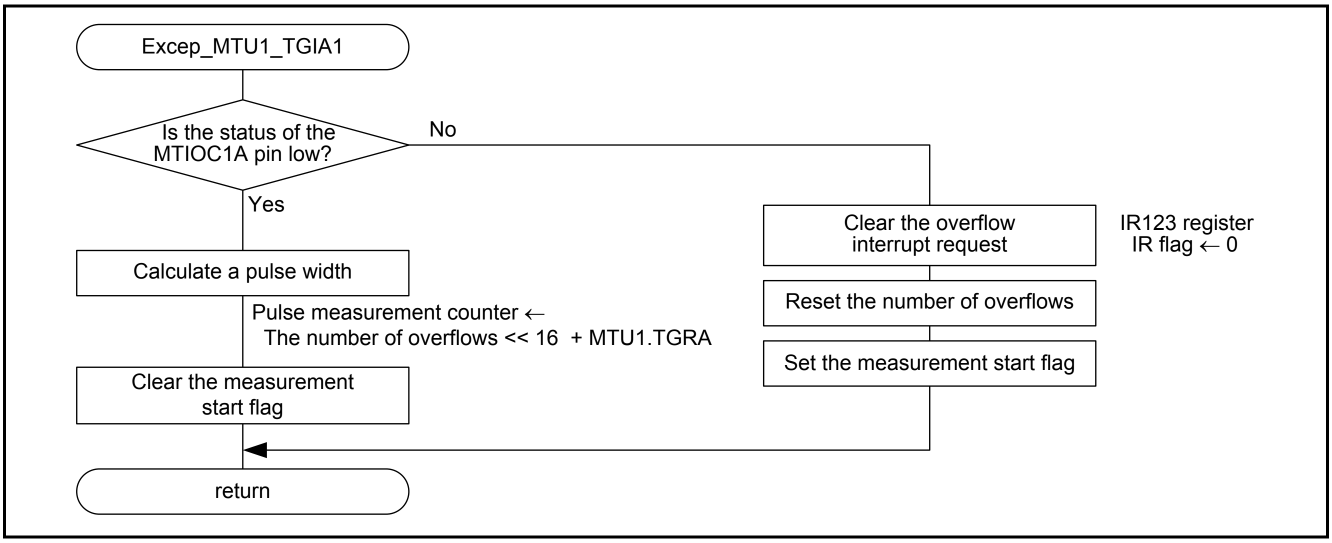

Figure 5.17 MTU1 Input Capture A Interrupt Handler

5.2.8.5 MTU1 Overflow Interrupt Handler

Figure 5.18 shows the MTU1 Overflow Interrupt Handler.

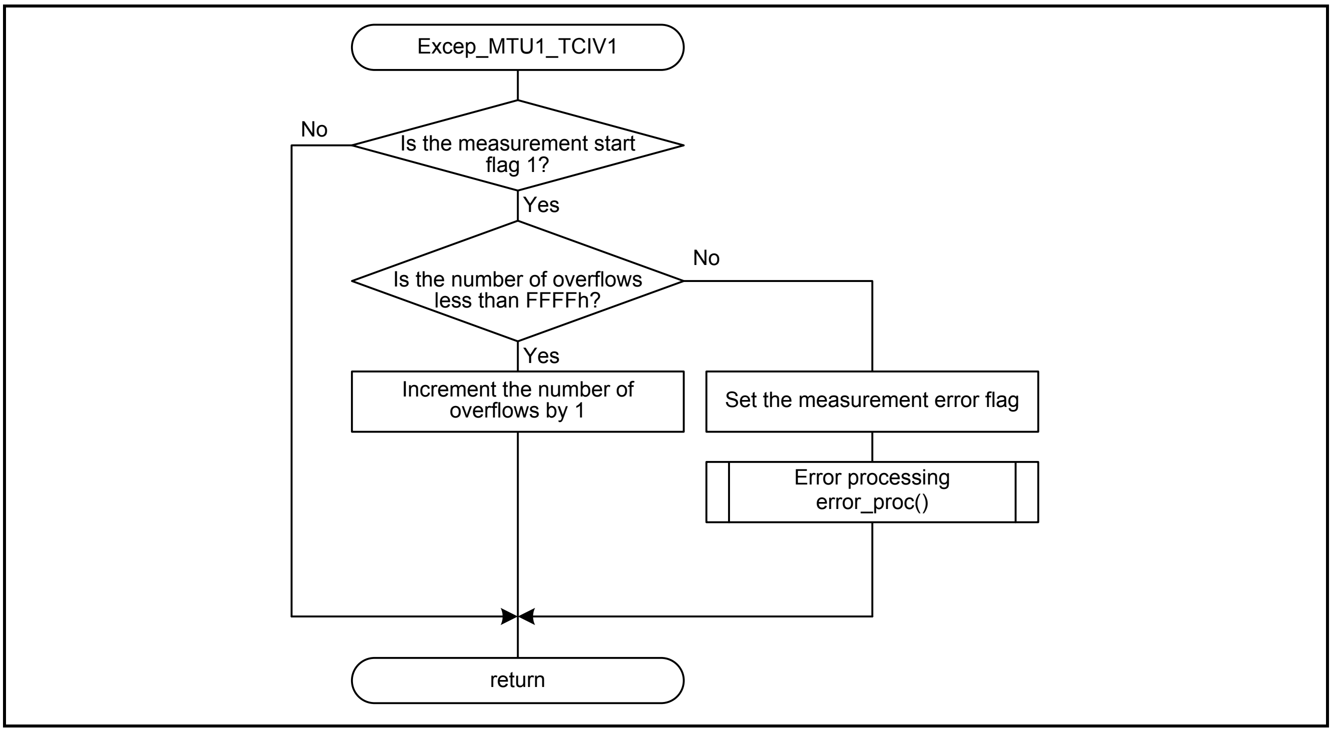

Figure 5.18 MTU1 Overflow Interrupt Handler

6. Applying This Application Note to the RX21A or RX220 Group

The sample code accompanying this application note has been confirmed to operate with the RX210 Group. To make the sample code operate with the RX21A or RX220 Group, use this application note in conjunction with the Initial Setting application note for each group.

For details on using this application note with the RX21A and RX220 Groups, refer to "5. Applying the RX210 Group Application Note to the RX21A Group" in the RX21A Group Initial Setting application note, and "4. Applying the RX210 Group Application Note to the RX220 Group" in the RX220 Group Initial Setting application note.

7. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

8. Reference Documents

User's Manual: Hardware

RX210 Group User's Manual: Hardware Rev.1.50 (R01UH0037EJ)
RX21A Group User's Manual: Hardware Rev.1.00 (R01UH0251EJ)
RX220 Group User's Manual: Hardware Rev.1.10 (R01UH0292EJ)
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.

User's Manual: Development Tools

RX Family C/C++ Compiler Package V.1.01 User's Manual Rev.1.00 (R20UT0570EJ)
The latest version can be downloaded from the Renesas Electronics website.

Website and Support

Renesas Electronics website

http://www.renesas.com

Inquiries

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

REVISION HISTORY	RX210, RX21A, and RX220 Groups Application Note Pulse Width Measurement Using MTU2a

Rev.	Date	Description	
		Page	Summary
1.00	Sep. 1, 2012	—	First edition issued
1.10	Mar. 1, 2013	1	Added details in the Products.
		4	Table 2.1 Operation Confirmation Conditions: Updated information
		5	4.1 Pins Used: Added the explanation.
		6	5.1 Sample Code 1: Changed interrupt priority levels of all interrupts in the Interrupts setting.
		7	5.1.1.1 Measuring a Pulse Width: Changed the last sentence in (2).
		8	5.1.1.2 Operation When an Input Capture and Overflow Occur Simultaneously: - Changed explanation in (1), (2), and (3). - Changed timing descriptions of IR flags in Figure 5.2.
		9	Table 5.1 Files Used in the Sample Code: Added and modified files.
		10	Table 5.4 Global Variables: Changed function names in the Function Used.
		10	Table 5.5 Functions Used in the Sample Code: - Added the R_INIT_StopModule function. - Changed the following function names: - non_existent_port_init to R_INIT_NonExistentPort - clock_init to R_INIT_Clock - Excep_MTU21_TGIA1 to Excep_MTU1_TGIA1 - Excep_MTU21_TGIB1 to Excep_MTU21_TGIB1 - Excep_MTU21_TGIV1 to Excep_MTU1_TCIV1
		11	5.1.7 Function Specifications: - R_INIT_StopModule: Added the specification. - Changed the explanations of the following functions: - R_INIT_NonExistentPort - R_INIT_Clock - Excep_MTU1_TGIA1
		14	5.1.8.1 Main Processing: - Added the process for the R_INIT_StopModule function.
		15	5.1.8.3 Peripheral Function Initialization: - Divided the flowchart into two as Figures 5.5 and 5.6. - Changed interrupt priority levels for the Set interrupt priority process in Figure 5.6.
		17	5.1.8.6 MTU1 Input Capture B Interrupt Handler: Omit processing when an overflow occurs.
		19	5.2 Sample Code 2: - Changed the division of PCLKB for the noise filter clock in MTU1 setting. - Changed interrupt priority levels of all interrupts in the Interrupts setting.

Rev.	Date	Description	
		Page	Summary
1.10	Mar. 1, 2013	21	5.2.1.2 Operation When an Input Capture and Overflow Occur Simultaneously: - Changed explanation in (1), (2), and (3). - Changed timing descriptions of IR flags in Figure 5.12.
		22	Table 5.6 Files Used in the Sample Code: Added and modified files.
		23	Table 5.9 Global Variables: Changed function names in the Function Used.
		23	Table 5.10 Functions Used in the Sample Code: - Added the R_INIT_StopModule function. - Changed the following function names: - non_existent_port_init to R_INIT_NonExistentPort - clock_init to R_INIT_Clock - Excep_MTU21_TGIA1 to Excep_MTU1_TGIA1 - Excep_MTU21_TGIV1 to Excep_MTU1_TCIV1
		24	5.2.7 Function Specifications: - R_INIT_StopModule: Added the specification. - Changed the explanations of the following functions: - R_INIT_NonExistentPort - R_INIT_Clock
		26	5.2.8.1 Main Processing: - Added the process for the R_INIT_StopModule function.
		27	5.2.8.3 Peripheral Function Initialization: - Changed the division of PCLKB for the explanation of the Set the noise filter process. - Changed interrupt priority levels for the Set interrupt priority process in Figure 5.15.
		28	5.2.8.5 MTU1 Input Capture A Interrupt Handler: Omit processing when an overflow occurs.
		30	7. Reference Documents: Updated the revision of the User's Manual: Hardware
1.11	July 1, 2014	1	Products: Added the RX21A and RX220 Groups.
		4	3. Reference Application Notes: Added the Initial Setting application notes for the RX21A and RX220 Groups.
		11, 12, 24, 25	Modified the description of reference application note in the following functions: R_INIT_StopModule, R_INIT_NonExistentPort, and R_INIT_Clock.
		30	6. Applying This Application Note to the RX21A or RX220 Group: Added.
		31	8. Reference Documents: Added the User's Manual: Hardware for the RX21A and RX220 Groups.

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this 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.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of an MPU or MCU 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.

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

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

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

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

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

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, 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 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022/9044

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

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Ku, Seoul, 135-920, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 4.0