RX Family and M32C/R32C Series

Guide for Migration from the M32C/R32C to the RX: Timers

Abstract

This document describes migration from Timers A and B of the M32C/R32C Series to the MTU3a timer of the RX Family.

Products

- RX Family
- M32C/80 Series
- R32C/100 Series

This document explains migration from the M32C/R32C Series to the RX Family, using the RX660 Group MCU as an example of the RX Family, the M32C/87 Group MCU as an example of the M32C/80 Series MCU, and the R32C/118 Group MCU as an example of the R32C/100 Series MCU. When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

There are differences in terminology between the RX Family MCU and the M32C/R32C Series MCU.

The table below shows the differences in terminology related to timers.

Differences in Terminology Between the RX Family and the M32C/R32C Series

Item	**M32C/R32C Series**	**RX Family**
Timer modules	Timer A Timer B ...	Multi-function timer pulse unit 3 (MTU3a) Compare match timer (CMT) Compare match timer (CMTW) 16-bit timer pulse unit (TPUa) 8-bit timers (TMR and TMRb) (The above items are examples.)
Peripheral function operating clocks	Peripheral function clocks: fC, fC32, fOCO40M, fOCO-F, fOCO-S, f1	Peripheral module clocks: PCLKA, PCLKB, PCLKD
Timer operating clocks (hereinafter count clock)	Count source	Count clock
Function for selecting peripheral function input and output for pins	Function select registers and input function select registers	MPC *1
Registers for peripheral functions	Special function registers (SFRs)	I/O registers

Note: 1. The MPC is not available in some groups.

Contents

1. General Differences in Timers

Table 1.1 shows General Differences Between the Timers A and B of the M32C/87 and the MTU3a of the RX660. Table 1.2 shows General Differences Between Timers A and B of the R32C/118 and MTU3a of the RX660.

Table 1.1 General Differences Between the Timers A and B of the M32C/87 and the MTU3a of the RX660

Item	Timers A and B (M32C/87)	MTU3a (RX660)
Number of channels	11 channels (Timer A: 5 channels, Timer B: 6 channel)	9 channels (16 bits × 8 channels, 32 bits × 1 channel)
Counting	Counting-up Counting-down	Counting-up Counting-down
Maximum count value	FFFFh *1	FFFFh / FFFF FFFFh
Operating mode	• Timer mode • Pulse width modulation mode (PWM mode) • Event counter mode • One-shot timer mode • Pulse period measurement mode • Pulse width measurement mode	• Normal mode • PWM modes 1 to 2 • Phase counting modes 1 to 5 • Reset-synchronized PWM mode • Complementary PWM modes 1 to 3
Start condition	• TAiS and TBiS bits of the TABSR and TBSR registers are set to “1” (start counting). • External trigger from the TAiIN pin *2 • Timer B2, Timer Aj (j = i - 1 (j = 4 if i = 0)), Timer Ak (k = i + 1 (k = 0 if i = 4)) overflows or underflows. *2 • The TAiOS bit of the ONSF register is set to “1” (start the timer). *2	• The CSTn bits of the TSTRA and TSTRB registers are set to “1” (enable counting). • Counting is started by the Event Link Controller (ELC).
Stop condition	The TAiS and TBiS bits of the TABSR and TBSR registers are set to “0” (stop counting).	The CSTn bits of the TSTRA and TSTRB registers are set to “0” (stop counting).
Interrupt request occurrence timing	• When an underflow occurs • When an overflow occurs • When a valid edge of the pulse measured is input. *3 • When a fall of the PWM pulse is detected	• When an underflow occurs • When an overflow occurs • When an input capture occurs • When a compare match occurs
Reading from timers	The count values are read by reading the TAi and TBi registers.	The count value is read by reading the TCNT register.
Writing to timers	• When counting is stopped When a value is written to the TAi and TBi registers, the value is written to both the reload register and the counter. • When counting is in progress *4 When a value is written to the TAi and TBi registers, the value is written to the reload registers (the value is transferred to the counters the next time a reload occurs).	Writing a value to the TCNT register
Function	• Gate function	• Low power consumption function • Event link function • A/D conversion start request delaying function • Interrupt skipping function

Notes: 1. FFFEh applies in the pulse width modulation mode (PWM mode).

2. In one-shot timer mode, the TAiS bit of the TABSR register must be “1” (start counting).
3. No interrupt request occurs when the first valid edge is input after counting has started.
4. Make sure that a value is written at least 1 cycle of the count source clock later after counting has started.

Table 1.2 General Differences Between Timers A and B of the R32C/118 and MTU3a of the RX660

Item	Timers A and B (R32C/118)	MTU3a (RX660)
Number of channels	11 channels (Timer A: 5 channels, Timer B: 6 channel)	9 channels (16 bits × 8 channels, 32 bits × 1 channel)
Counting	Counting-up Counting-down	Counting-up Counting-down
Maximum count value	FFFFh [*1]	FFFFh / FFFF FFFFh
Operating mode	• Timer mode • Pulse width modulation mode (PWM mode) • Event counter mode • One-shot timer mode • Pulse period measurement mode • Pulse width measurement mode	• Normal mode • PWM modes 1 to 2 • Phase counting modes 1 to 5 • Reset-synchronized PWM mode • Complementary PWM modes 1 to 3
Start condition	• TAiS and TBiS bits of the TABSR and TBSR registers are set to “1” (start counting). • External trigger from the TAiIN pin [*2] • Timer B2, Timer Aj (j = i - 1 (j = 4 if i = 0)), Timer Ak (k = i + 1 (k = 0 if i = 4)) overflows or underflows. [*2] • The TAiOS bit of the ONSF register is set to “1” (start the timer). [*2]	• The CSTn bits of the TSTRA and TSTRB registers are set to “1” (enable counting). • Counting is started by the Event Link Controller (ELC).
Stop condition	The TAiS and TBiS bits of the TABSR and TBSR registers are set to “0” (stop counting).	The CSTn bits of the TSTRA and TSTRB registers are set to “0” (stop counting).
Interrupt request occurrence timing	• When an underflow occurs • When an overflow occurs • When a valid edge of the pulse measured is input. [*3] • When a fall of the PWM pulse is detected	• When an underflow occurs • When an overflow occurs • When an input capture occurs • When a compare match occurs
Reading from timers	The count values are read by reading the TAi and TBi registers.	The count value is read by reading the TCNT register.
Writing to timers	• When counting is stopped When a value is written to the TAi and TBi registers, the value is written to both the reload register and the counter. • When counting is in progress [*4] When a value is written to the TAi and TBi registers, the value is written to the reload registers (the value is transferred to the counters the next time a reload occurs).	Writing a value to the TCNT register
Function	• Gate function	• Low power consumption function • Event link function • A/D conversion start request delaying function • Interrupt skipping function

Notes: 1. FFFEh applies in the pulse width modulation mode (PWM mode).
2. In one-shot timer mode, the TAiS bit of the TABSR register must be “1” (start counting).
3. No interrupt request occurs when the first valid edge is input after counting has started.
4. Make sure that a value is written at least 1 cycle of the count source clock later after counting has started.

2. Peripheral Functions Used

This document shows an example of operation that uses MTU3a of the RX Family MCU, and Timer A and Timer B of the M32C/R32C Series MCU.

Table 2.1 shows Peripheral Functions and Modes Used in Relation to the Operating Example.

Table 2.1 Peripheral Functions and Modes Used in Relation to the Operating Example

No.	Operation	M32C/R32C		RX	
		Peripheral Function	Mode	Peripheral Function	Mode
1	Outputting a PWM waveform	Timer A	PWM mode, timer mode [*1]	MTU3a	PWM1 mode
2	One-shot timer (single pulse output)		One-shot timer mode		PWM1 mode
3	Measuring a pulse period	Timer B	Pulse period measurement mode		Normal mode
4	Measuring a pulse width		Pulse width measurement mode		Normal mode

Note: 1. In timer mode, only a pulse with a duty cycle of 50% can be output.

2.1 Outputting a PWM Waveform

This section describes the differences in the operation that outputs a PWM waveform from an output pin between the case where MTU3a is used in PWM mode 1 on the RX Family MCU, and the case where Timer A is used in timer mode on the M32C/R32C Series MCU.

MTU3a of the RX Family MCU has free-running counter (TCNT register). In PWM1 mode, a value is set to the TGR register, and when the values for the TGR register and TCNT register match (compare match occurs), output can be high, low, or inverted.

This section describes an example of a PWM waveform being output. A high is output when the TCNT register and TGRA register values match, and a low is output when the TCNT register and TGRB register values match. This example assumes a duty cycle of 50%, and the M32C/R32C uses timer mode and not PWM mode.

Using PWM1 mode of MTU3a as an example, by making the TGRA register and TGRB register values the same, a pulse can be output with a duty cycle of 0% or 100%. Because Timer A in the M32C/R32C Series cannot output a pulse with a duty cycle of 100%, timer output is stopped, and control must be performed by ports.

Figure 2.1 shows Example of Outputting a PWM Waveform. Table 2.2 shows Operational Overview of Outputting a PWM Waveform.

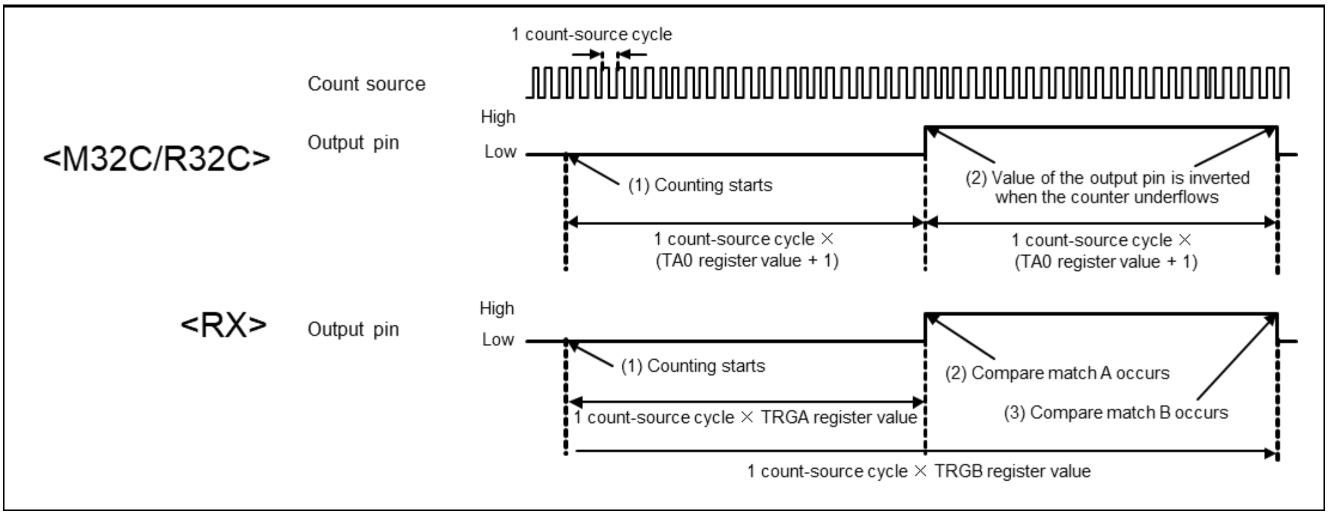

Figure 2.1 Example of Outputting a PWM Waveform

Table 2.2 Operational Overview of Outputting a PWM Waveform

Item	M32C/R32C (in the case of Timer A on the M32C/87 or Timer A on the R32C/118)	RX (in the case of MTU3a on the RX660)
Operating mode	Timer mode	PWM mode 1
Overview of operations	(1) Counting starts The count operation is started. (2) Counter underflows Each time the counter underflows, output from the pin is inverted.	(1) Counting starts The count operation is started. (2) Compare match A occurs When the TCNT and TGRA values match, compare match A occurs, and output from the pin changes from low to high. (3) Compare match B occurs When the TCNT and TGRB values match, compare match B occurs, and output from the pin changes from high to low. When compare match B occurs, the TCNT value is cleared.

2.2 One-shot timer

This section explains the difference in the operation that outputs a pulse only once from an output pin between the case where MTU3a is used in PWM mode 1 on the RX, and the case where Timer A is used in one-shot timer mode on the M32C/R32C.

MTU3a of the RX Family MCU has free-running counter (TCNT register). In PWM1 mode, a value is set to the TGR register, and when the values for the TGR register and TCNT register match (compare match occurs), output can be high, low, or inverted.

This section describes an example of a pulse being output only once. A high is output when the TCNT register and TGRA register values match, and a low is output when the TCNT register and TGRB register values match. In addition, when the TCNT register and TGRA register values match, buffer operation is used to transfer FFFFh to the TGRA, when the TCNT register and TGRB register values match, buffer operation is used to transfer FFFEh to the TGRB register. Furthermore, the TCNT register is cleared. Therefore, by preventing a second compare match with the TGRA register, changes in the state of the output pin can be avoided.

For the example used in this section, the value to be set for the TGRA register must be 0000h and the value to be set for the TGRB register must be in the range from 0001h to FFFEh. In addition, after outputting a single one-shot time pulse, when you want to output another pulse, stop the count and reset registers like the TCNT register and TGR register.

Figure 2.2 shows Example of Outputting a One-Shot Pulse. Table 2.3 shows Operational Overview of the One-Shot Timer.

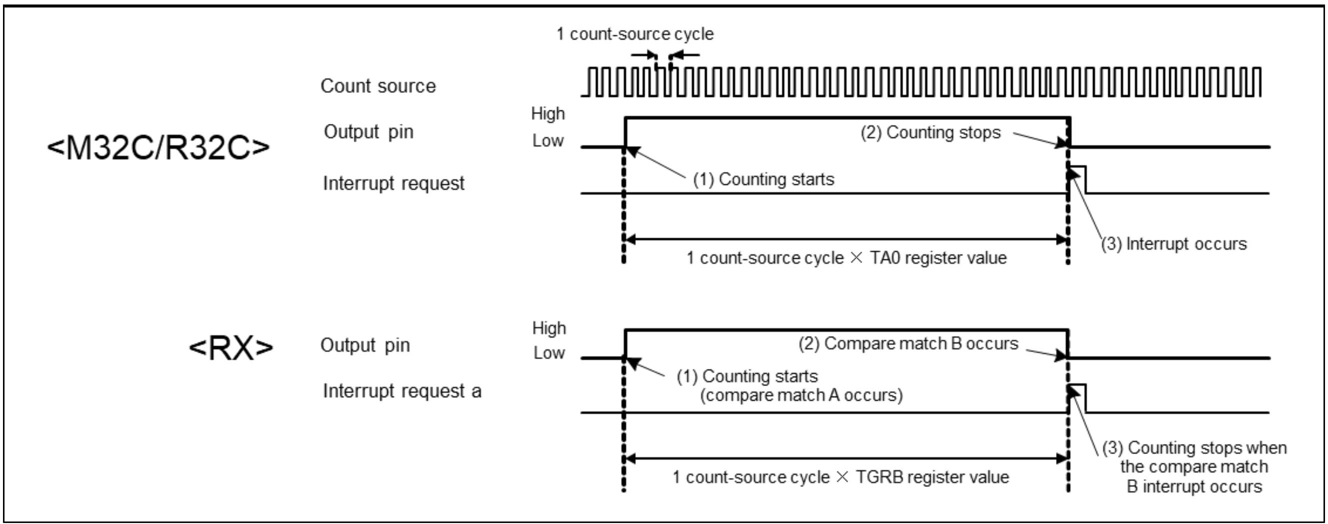

Figure 2.2 Example of Outputting a One-Shot Pulse

Table 2.3 Operational Overview of the One-Shot Timer

Item	**M32C/R32C (in the case of Timer A on the M32C/87 or Timer A on the R32C/118)**	**RX (in the case of MTU3a on the RX660)**
Operating mode	One-shot timer mode	PWM mode 1
Overview of operations	(1) Counting starts At the same time the count starts, the output pin changes from low to high. (2) Counting stops When the TA0 count value becomes 0000h, the count stops, and the output pin changes from high to low. (3) Interrupt occurs A timer A0 interrupt occurs when the TA0 count value becomes 0000h.	(1) Counting starts (compare match A occurs) When the TCNT and TGRA values match at the same time the count operation starts, compare match A occurs, and the output pin changes from low to high. Also, buffer operation is performed when compare match A occurs, and the value in TGRC (FFFFh) is transferred to TGRA. (2) Compare match B occurs When the TCNT and TGRB values match, compare match B occurs, and output from the pin changes from high to low. Also, when compare match B occurs, the TCNT value is cleared, buffer operation is performed, and the value in TGRD (FFFEh) is transferred to TGRB. (3) Counting stops when the compare match B interrupt occurs The count stops by the compare match B interrupt handling program.

2.3 Measuring a Pulse Period

This section describes the differences in the operation that measures the period from one rising edge to the next rising edge of a pulse input to an external input pin between the case where MTU3a are used in normal mode on the RX, and the case where Timer B is used in pulse period measurement mode on the M32C/R32C.

MTU3a of the RX Family MCU has free-running counter (TCNT register). When using the input capture function in normal mode, the input edge of the pin is detected, and the TCNT register value can be transferred to the TGR register. The input capture interrupt is generated when the input edge of the pin is detected, the overflow interrupt is generated when the TCNT register overflows, and these interrupts can be used independently.

In the pulse period measurement example in this section, each time a rising edge is detected on the external input pin, the value in the TCNT register is transferred to the TGRA register, at which time the TCNT register is set to be cleared. Each time the overflow interrupt is generated, the variable counts the number of overflows, and when the input capture interrupt is generated, the TGRA register value and the number of overflows counted are used to calculate the pulse period.

Figure 2.3 shows Example of Measuring the Pulse Period. Table 2.4 shows Operational Overview of Measuring the Pulse Period.

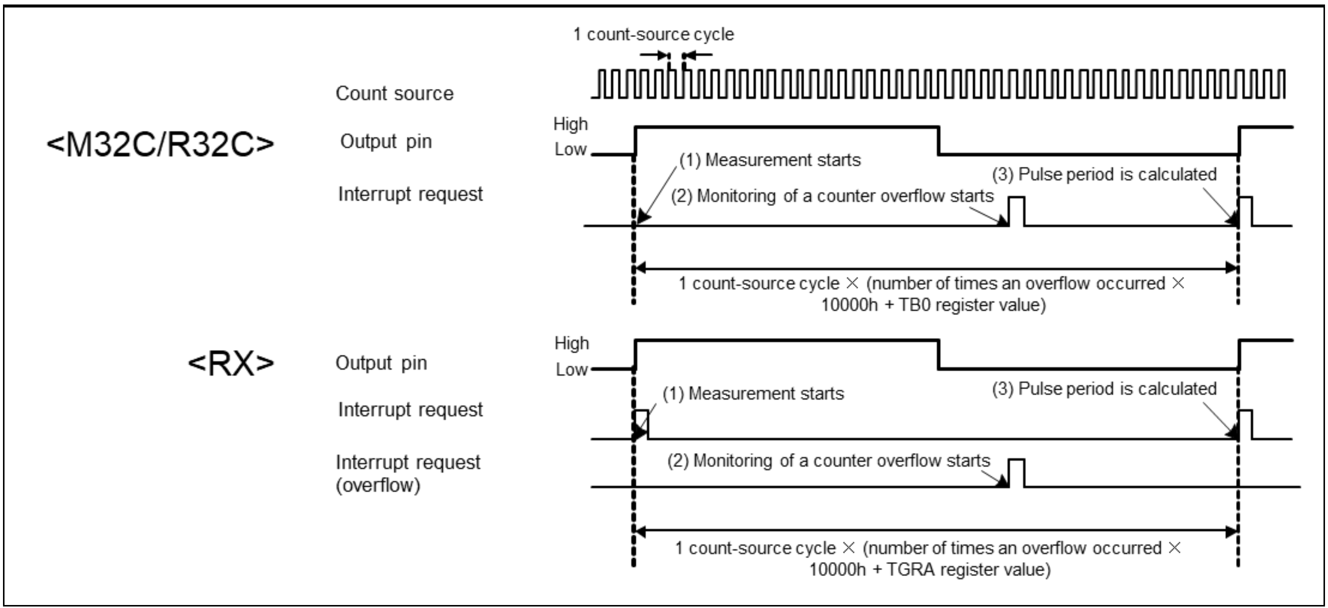

Figure 2.3 Example of Measuring the Pulse Period

Table 2.4 Operational Overview of Measuring the Pulse Period

Item	M32C/R32C (in the case of Timer B on the M32C/87 or Timer B on the R32C/118)	RX (in the case of MTU3a on the RX660)
Operating mode	Pulse period measurement mode	Normal mode
Overview of operations	(1) Measurement starts When a rising edge of a pulse input on an external input pin is detected, the timer B0 interrupt occurs. (2) Monitoring of a counter overflow starts When TB0 overflows, the timer B0 interrupt occurs. Read the overflow flag in the interrupt handler, and count the number of overflows. (3) Pulse period is calculated The pulse period can be calculated using the number of overflows and the TB0 register value.	(1) Measurement starts When a rising edge of a pulse input on an external input pin is detected, the input capture interrupt occurs. (2) Monitoring of a counter overflow starts When TCNT overflows, the overflow interrupt occurs. Count the number of overflows in the interrupt handler. (3) Pulse period is calculated The pulse period can be calculated using the number of overflows and the TGRB register value.

2.4 Measuring a Pulse Width

This section describes the differences in the operation that measures the pulse width from one rising edge to the next falling edge of a pulse input to an external input pin between the case where MTU3a is used in normal mode on the RX, and the case where Timer B is used in pulse width measurement mode on the M32C/R32C.

MTU3a of the RX Family MCU has free-running counter (TCNT register). When using the input capture function in normal mode, the input edge of the pin is detected, and the TCNT register value can be transferred to the TGR register. The input capture interrupt is generated when the input edge of the pin is detected, the overflow interrupt is generated when the TCNT register overflows, and these interrupts can be used independently.

In the pulse width measurement example in this section, each time an edge is detected (rising edge or falling edge) on the external input pin, the value in the TCNT register is transferred to the TGRA register, at which time the TCNT register is set to be cleared. Each time the overflow interrupt is generated, the variable counts the number of overflows, and when the input capture interrupt is generated, the TGRA register value and the number of overflows counted are used to calculate the pulse width.

Figure 2.4 shows Example of Measuring the Pulse Width. Table 2.5 shows Operational Overview of the Example of Measuring the Pulse Width.

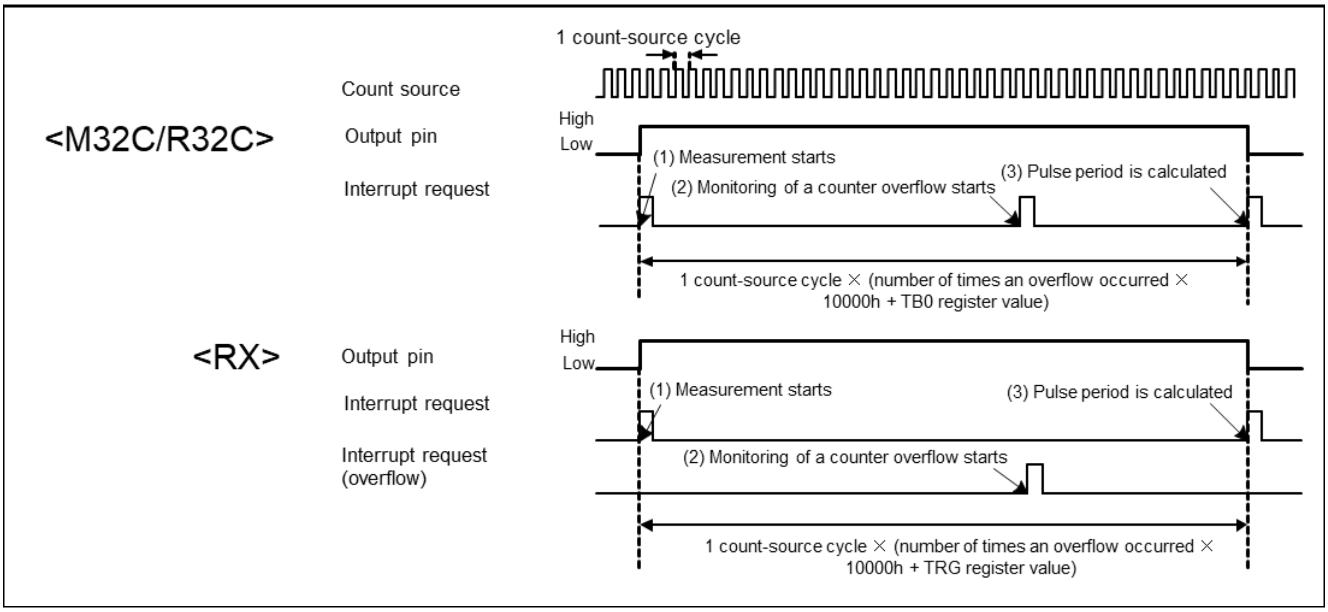

Figure 2.4 Example of Measuring the Pulse Width

Table 2.5 Operational Overview of the Example of Measuring the Pulse Width

Item	M32C/R32C (in the case of Timer B on the M32C/87 or Timer B on the R32C/118)	RX (in the case of MTU3a on the RX660)
Operating mode	Pulse width measurement mode	Normal mode
Overview of operations	(1) Measurement starts When a rising edge or falling edge of a pulse input on an external input pin is detected, the timer B0 interrupt occurs. (2) Monitoring of a counter overflow starts When TB0 overflows, the timer B0 interrupt occurs. Read the overflow flag in the interrupt handler, and count the number of overflows. (3) Pulse width is calculated The pulse width can be calculated using the number of overflows and the TB0 register value	(1) Measurement starts When a rising edge or falling edge of a pulse input on an external input pin is detected, the input capture interrupt occurs. (2) Monitoring of a counter overflow starts When TCNT overflows, the overflow interrupt occurs. Count the number of overflows in the interrupt handler. (3) Pulse width is calculated The pulse width can be calculated using the number of overflows and the TGR register value.

3. Related Sections in the RX User's Manual: Hardware

When migrating from the M32C/R32C Series MCU to the RX Family MCU, refer to the following sections of the user's manual for hardware:

- Multi-Function Timer Pulse Unit 3
- Clock Generation Circuit
- Low Power Consumption
- Interrupt Controller, CPU
- I/O Ports, MPC
- Register Write Protection Function

4. Appendix

4.1 Points on Migration from the M32C/R32C to the RX

This section explains points on migration from the M32C/R32C to the RX.

4.1.1 Interrupts

For the RX Family, when an interrupt request is received while all of the following conditions are met, the interrupt occurs.

- The I flag (PSW.I bit) is 1.
- Registers IER and IPR in the ICU are set to enable interrupts.
- The interrupt request is enabled by the interrupt request enable bits for peripheral functions.

Table 4.1 shows Comparison of Conditions for Interrupt Generation Between the M32C/R32C and the RX.

Table 4.1 Comparison of Conditions for Interrupt Generation Between the M32C/R32C and the RX

Item	M32C/R32C	RX
I flag	When the I flag is set to 1 (enabled), the maskable interrupt request can be accepted.	
Interrupt request flag	When there is an interrupt request from a peripheral function, the interrupt request flag becomes 1 (interrupt requested).	
Interrupt priority level	Selected by setting bits ILVL2 to ILVL0.	Selected by setting the IPR[3:0] bits.
Interrupt request enable	—	Specified by setting the IER register.
Interrupt enable for peripheral functions	—	Interrupts can be enabled or disabled in each peripheral function.

For more information, refer to sections Interrupt Controller (ICU), CPU, and sections for other peripheral functions used in the user's manual for hardware.

4.1.2 Module Stop Function

The RX Family has the ability to stop each peripheral module individually.

By transitioning unused peripheral modules to the module stop state, power consumption can be reduced.

After a reset is released, all modules (with a few exceptions) are in the module stop state.

Registers for modules in the module stop state cannot be written to or read.

For more information, refer to the Low Power Consumption section in the user’s manual for hardware.

4.1.3 I/O Ports

In the RX Family, the MPC must be configured in order to assign I/O signals from peripheral functions to pins.

Before performing pin I/O control in the RX Family, perform the following two operations:

- In the MPC.PFS register, select the peripheral functions that are assigned to the appropriate pins.
- In the PMR register for I/O ports, select the function for the pin to be used as a general I/O port or I/O port for a peripheral function.

The M32C/R32C provides a function select register that allows the user to select whether to use the pin as an I/O port or for the output port for a specific peripheral function.

Before performing pin I/O control in the M32C, perform the following two operations:

- Function select registers B to E: Use these registers to select the peripheral function that can be assigned to the target pin.
- Function select register A: Use this register to select whether the target pin is to be used as a general I/O port or for the selected peripheral function.

Before performing pin I/O control in the R32C, perform the following operation:

- Function select register: Use this register to select the peripheral function that can be assigned and to select whether the target pin is to be used as a general I/O port or for the selected peripheral function.

Table 4.2 shows Comparison of I/O Settings for Peripheral Function Pins Between the M32C and the RX.
Table 4.3 shows Comparison of I/O Settings for Peripheral Function Pins Between the R32C and the RX.

Table 4.2 Comparison of I/O Settings for Peripheral Function Pins Between the M32C and the RX

Function	M32C (in the case of the M32C/87)	RX (in the case of the RX660)
Select the pin function	With the function select registers B to E, I/O ports for peripheral functions can be assigned by selecting from multiple pins.	With the PFS register, I/O ports for peripheral functions can be assigned by selecting from multiple pins.
Switch between general I/O port and peripheral function	With the function select register A, the corresponding pin function can be selected as a general I/O port or a peripheral function.	With the PMR register, the corresponding pin function can be selected as a general I/O port or a peripheral function.

Table 4.3 Comparison of I/O Settings for Peripheral Function Pins Between the R32C and the RX

<table>
<tr><th>Function</th><th>R32C (in the case of the R32C/118)</th><th>RX (in the case of the RX660)</th></tr>
<tr><td>Select the pin function</td><td rowspan="2">With the function select register, the corresponding pin function can be selected as a general I/O port or a peripheral function.
Output ports for peripheral functions can be assigned by selecting from multiple pins.</td><td>With the PFS register, I/O ports for peripheral functions can be assigned by selecting from multiple pins.</td></tr>
<tr><td>Switch between general I/O port and peripheral function</td><td>With the PMR register, the corresponding pin function can be selected as a general I/O port or a peripheral function.</td></tr>
</table>

For details on the RX, refer to the chapters on the multi-function pin controller (MPC) and I/O ports in the user's manual for hardware.

For details on the M32C, refer to the chapter on programmable I/O ports in the user's manual for hardware.

For details on the R32C, refer to the chapter on I/O ports in the user's manual for hardware.

4.2 I/O Register Macros

Macro definitions listed in Table 4.4 can be found in the RX I/O register definitions (iodefine.h).

The readability of programs can be achieved with these macro definitions.

Table 4.4 shows Macro Usage Examples.

Table 4.4 Macro Usage Examples

Macro	Usage Example
IR("module name", "bit name")	**IR(MTU0,TGIA0)** = 0 ; The IR bit corresponding to MTU0.TGIA0 is cleared to 0 (no interrupt request is generated).
DTCE("module name", "bit name")	**DTCE (MTU0, TGIA0)** = 1 ; The DTCE bit corresponding to MTU0.TGIA0 is set to 1 (DTC activation is enabled).
IEN("module name", "bit name")	**IEN(MTU0, TGIA0)** = 1 ; The IEN bit corresponding to MTU0.TGIA0 is set to 1 (interrupt enabled).
IPR("module name", "bit name")	**IPR(MTU0, TGIA0)** = 0x02 ; The IPR bit corresponding to MTU0.TGIA0 is set to 2 (interrupt priority level 2).
MSTP("module name")	**MSTP(MTU)** = 0 ; The MTU0 Module Stop bit is set to 0 (module stop state is canceled).
VECT("module name", "bit name")	#pragma interrupt (Excep_MTU0_TGIA0 (vect=**VECT(MTU0, TGIA0)**) The interrupt function is declared for the corresponding MTU0.TGIA0 register.

4.3 Intrinsic Functions

The RX Family has intrinsic functions for setting control registers and special instructions. When using intrinsic functions, include machine.h.

Table 4.5 shows Examples of Differences in the Settings of Control Registers and Descriptions of Special Instructions Between the M32C/R32C and the RX.

Table 4.5 Examples of Differences in the Settings of Control Registers and Descriptions of Special Instructions Between the M32C/R32C and the RX

Item	Description	
	M32C/R32C	RX
Set the I flag to 1	asm("fset i");	setpsw_i (); [*1]
Set the I flag to 0	asm("fclr i");	clrpsw_i (); [*1]
Expanded into the WAIT instruction	asm("wait");	wait(); [*1]
Expanded into the NOP instruction	asm("nop");	nop(); [*1]

Note: 1. The machine.h file must be included.

5. Reference Documents

User's Manual: Hardware

RX660 Group User's Manual: Hardware (R01UH0937EJ)
M32C/87 Group (M32C/87, M32C/87A, M32C/87B) Hardware Manual (REJ09B0180)
R32C/118 Group User's Manual: Hardware (R01UH0212EJ)
If you are using a product that does not belong to the above groups, refer to the applicable user's manual for hardware.
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 CC-RX Compiler User's Manual (R20UT3248EJ)
C Compiler Package for the M32C Series (M3T-NC308WA)
C Compiler Package for the R32C Series
The latest versions can be downloaded from the Renesas Electronics website.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jan. 10, 24	—	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/.