RX Family

Synchronous Operation Using MTU3/GPTW

Introduction

This application note describes how to perform synchronous operation using the MTU3d and GPTW (start, stop, clearing (restart)).

RX66T Group microcontrollers (MCUs) are equipped with the Multi-Function Timer Pulse Unit 3 (MTU3d) and the General-Purpose PWM Timer (GPTW).

The descriptions in this application note target RX Family devices equipped with the MTU3 and the GPTW. When using this application note with Renesas MCUs other than the RX66T Group, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Target Device

RX Family devices equipped with the MTU3 and GPTW

Confirmed Devices

RX66T Group

The Multi-Function Timer Pulse Unit 3 is referred to as “MTU” throughout this document.

Contents

1. MTU and GPTW Specifications

This application note describes synchronous operation of the MTU and GPTW (start, stop, clearing (restart)).

The following terms are used throughout the application note.

- Inter-channel — Signals, registers, etc. in the MTU and GPTW are used for synchronous operation within the MTU or GPTW.
- Inter-module — Modules, pins, etc. outside the MTU or GPTW are used for synchronous operations with modules, pins, etc. outside the MTU or GPTW
  External trigger input and ELC event input are included in this classification
- Software — Sets registers
- Hardware — Pins, internal/external module interrupts, internal signals, etc.

The following sections describe the differences between MTU and GPTW synchronous operation.

1.1 Differences in Synchronous Operations

The following lists the MTU and GPTW synchronous operations (start, stop, clearing (restart)).

Table 1.1 Synchronous Operation Functions (1/2)

Item	MTU	GPTW
Inter-channel synchronous start	Synchronous start by software • Use TSTRA and TSTRB registers — When the bits of the target channels are set to 1b, the TCNT counters of the set channels start counting — MTU0 to MTU4 and MTU9 cannot be synchronized with MTU6 and MTU7 • Use TCSYSTR register — When the bits of the target channels are set to 1b, the TCNT counters of the set channels start counting — MTU0 to MTU4 and MTU9 can be synchronized with MTU6 and MTU7	Synchronous start by software • Use GTSTR register — When the bits of the target channels are set to 1b, the GTCNT counters of the set channels start counting — Updating the GTSTR register of any channel enables counting starts of GTCNT counters of the channels set to 1b
Inter-channel synchronous stop	Synchronous stop by software • Use TSTRA and TSTRB registers — When the bits of the target channels are set to 0b, the TCNT counters of the set channels stop counting — MTU0 to MTU4 and MTU9 cannot be synchronized with MTU6 and MTU7	Synchronous stop by software • Use GTSTP register — When the bits of the target channels are set to 1b, the GTCNT counters of the set channels stop counting — Updating the GTSTP register of any channel enables counting stop for the GTCNT counters of the channels set to 1b
Inter-channel synchronous clearing	Synchronous clearing by hardware • Use compare match — Use the CCLR bits of TSYRA, TSYRB and TCR to set the clearing generation source and channels — When a clearing generation source is generated, the TCNT counters of the channels to be cleared are count cleared — MTU0 to MTU4 and MTU9 cannot be synchronized with MTU6 and MTU7 • Use TGImn interrupt (m = A to D, n = 0 to 2) — When a clearing generation source is set in the TSYCR register, the TCNT counters of MTU6 and MTU7 clear the counts at the set interrupt generation timing	Synchronous clearing by software • Use GTCLR register — When the bits of the target channels are set to 1b, the GTCNT counters of the set channels are cleared — Count stop is possible for the GTCNT counter on channels set to 1b when updating the GRCLR register of any channel

Table 1.2 Synchronous Operation Functions (2/2)

Item	MTU	GPTW
Inter-module synchronous start	Synchronous start by hardware • ELC event input — Set “counting starts” to the bits for the target channels of ELOPA, ELOPB, and ELOPE in the ELC — When the event sources selected in the ELSRn register are generated, the CSTn bits of the TSTRA and TSTRB registers are set to 1b and counting starts. — If the same event sources are selected in the ELSRn register, MTU0, MTU3, MTU4, MTU6, MTU7, and MTU9 can be synchronized	Synchronous start by hardware • Select sources with GTSSR register The following sources can be selected: — External trigger input (GTETRGA, GTETRGB, GTETRGC and GTETRGD) — ELC event input — Pin input (GTIOCnA and GTIOCnB)
Inter-module synchronous stop	—	Synchronous stop by hardware • Select sources with GTPSR register The following sources can be selected: — External trigger input (GTETRGA, GTETRGB, GTETRGC and GTETRGD) — ELC event input — Pin input (GTIOCnA and GTIOCnB)
Inter-module synchronous clearing	Synchronous clearing by hardware • ELC event input — Set “counting restarts” to the bits for the target channels of ELOPA, ELOPB, and ELOPE in the ELC — When the event sources selected in the ELSRn register are generated, the TCNT counts of the selected channels are cleared — If the same event sources are selected in the ELSRn register, MTU0, MTU3, MTU4, MTU6, MTU7, and MTU9 can be synchronized	Synchronous clearing by hardware • Select sources with GTCSR register The following sources can be selected: — External trigger input (GTETRGA、GTETRGB, GTETRGC and GTETRGD) — ELC event input — Pin input (GTIOCnA and GTIOCnB)

2. Operation Confirmation Conditions

The sample codes included in this application note have been confirmed under the following operating conditions.

Table 2.1 Operation Confirmation Environments

Item	Description
MCU	R5F566TEADFP (included in Renesas Starter Kit for RX66T)
Operating frequency	Main clock: 8MHz PLL: 160MHz (Main clock x 1/1 x 20) HOCO: Stopped LOCO: Stopped System clock (ICLK): 160MHz (PLL x 1/1) Peripheral module clock A (PCLKA): 80MHz (PLL x 1/2) Peripheral module clock B (PCLKB): 40MHz (PLL x 1/4) Peripheral module clock C (PCLKC): 160MHz (PLL x 1/1) Peripheral module clock D (PCLKD): 40MHz (PLL x 1/4) FlashIF clock (FCLK): 40MHz (PLL x 1/4)
Operating voltage	3.3V
Integrated development environment (IDE)	Renesas Electronics e^2 studio Version 2022-01
C compiler[Note]	Renesas Electronics C/C++ Compiler Package for RX Family v3.04.00
	Compiler option The integrated development environment default settings are used.
iodefine.h version	V1.00
Endian	Little endian
Operation mode	Single-chip mode
Processor mode	Supervisor mode
Sample code version	V1.00
Board	Renesas Starter Kit for RX66T (Product number: RTK50566T0CxxxxxBE)
Emulator	E2-Lite

Note: Import the same version of the toolchain (C compiler) as specified in the original project. If the same toolchain is not located in the import destination, the toolchain cannot be selected, and an error will occur. Check the toolchain selection status on the project settings screen.

Refer to FAQ 3000404 for setting methods.

FAQ 3000404: ‘Program "make" not found in PATH’ error when attempting to build an imported project (e^2 studio)

3. MCU Sample Codes

3.1 Common

3.1.1 Sample Code List

This application note provides the following sample codes created with the Smart Configurator.

Sample codes can be downloaded from the Renesas Electronics website.

Table 3.1 MTU Sample Code List

Name	Sample Code Usage Conditions	Ref.
12-Phase PWM Output in PWM Mode 2 r01an6282_rx66t_mtu3_pwm2_sync.zip	• PWM mode 2 • Software (TCSYSTR register) synchronous start • Hardware (compare match) synchronous clearing	3.2
Inter-Channel Synchronous Clearing Using Compare Match r01an6282_rx66t_mtu3_cmp_sync.zip	• PWM mode 1 • Software (TCSYSTR register) synchronous start • Hardware (compare match) synchronous clearing	3.3
5-Phase Complementary PWM Output r01an6282_rx66t_mtu3_complementary_sync.zip	• Complementary PWM mode 2 (transfer at trough) • Software (TCSYSTR register) synchronous start	3.4
MTU6/MTU7 Counter Synchronous Clearing by Interrupt r01an6282_rx66t_mtu3_int_sync.zip	• PWM mode 1 • Software (TCSYSTR register) synchronous start • Hardware (TGImn interrupt) synchronous clearing	3.5
Synchronous Operation by Event Input from ELC r01an6282_rx66t_mtu3_elc_sync.zip	• PWM mode 1 • Hardware (ELC) synchronous start • Software (TSTRA register) synchronous stop • Hardware (compare match) synchronous clearing	3.6

3.1.2 Folder Structure

The main folder structure of a sample code is as follows.

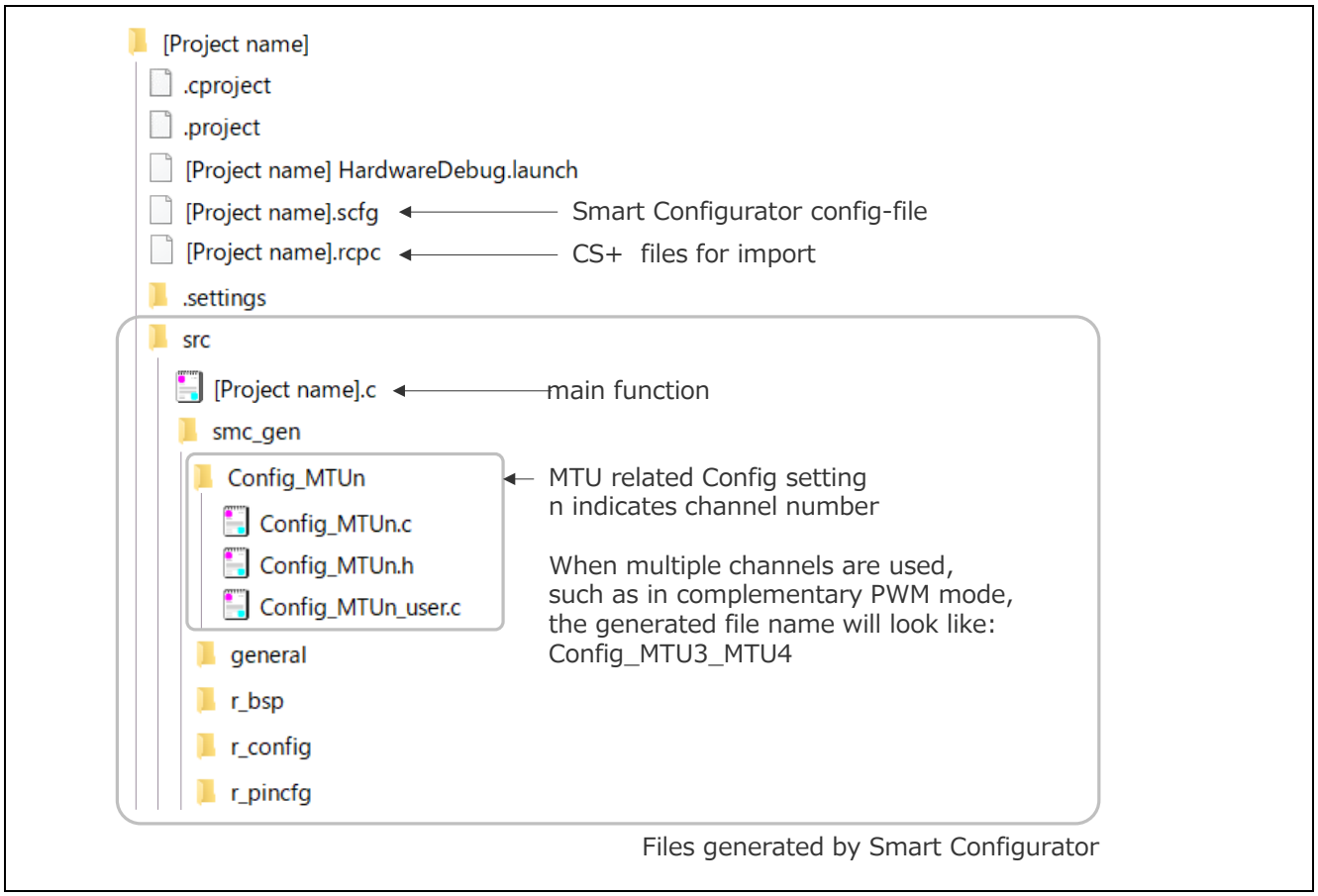


Figure 3.1 MTU Folder Structure

3.1.3 File Structure

The main file structure of a sample code is as follows.

Table 3.2 MTU File Structure

File Name	Description
[Project name].c	main function This is the main function. The Smart Configurator generates an empty function. The necessary processing for each sample code is described here.
Config_MTUn.c*	R_Config_MTUn_Create function This is the MTU's initialization function. The initialization function based on the settings in the Smart Configurator is generated by the Smart Configurator. The call for this function is generated by the Smart Configurator. This function is called in the R_SystemInit function executed before the main function.
	R_Config_MTUn_Start function This is the MTU's count start function. This function is generated by the Smart Configurator. In the sample codes, this function is called from the main function.
	R_Config_MTUn_Stop function This is the MTU's count stop function. This function is generated by the Smart Configurator. This function is not used in the sample codes.
Config_MTUn_user.c*	r_Config_MTUn_Create_UserInit function This is the MTU's user initialization function. The Smart Configurator generates an empty function. The necessary processing for each sample code is described here. This is the last function to be called in the R_Config_MTUn_Create function generated by the Smart Configurator.
	r_Config_MTUn_[interrupt name]_interrupt function This is the interrupt handler function. The Smart Configurator generates an empty function. The necessary processing for each sample code is described here.
Config_MTUn.h*	This is the header file that defines MTU related functions. This file is included in the r_smc_entry.h file generated by the Smart Configurator. To use MTU related functions, be sure to include the r_smc_entry.h file.

*: n indicates channel number

3.1.4 Adding Components

The sample codes use the Smart Configurator to add the MTU as described below.

Table 3.3 Adding Components

Item	Description
Component	Reference the section for each sample code ((1) in figure below)
Configuration Name	Sample codes use the default setting name
Operation	Reference the section for each sample code ((2) in figure below)
Resource	Reference the section for each sample code ((3) in figure below)

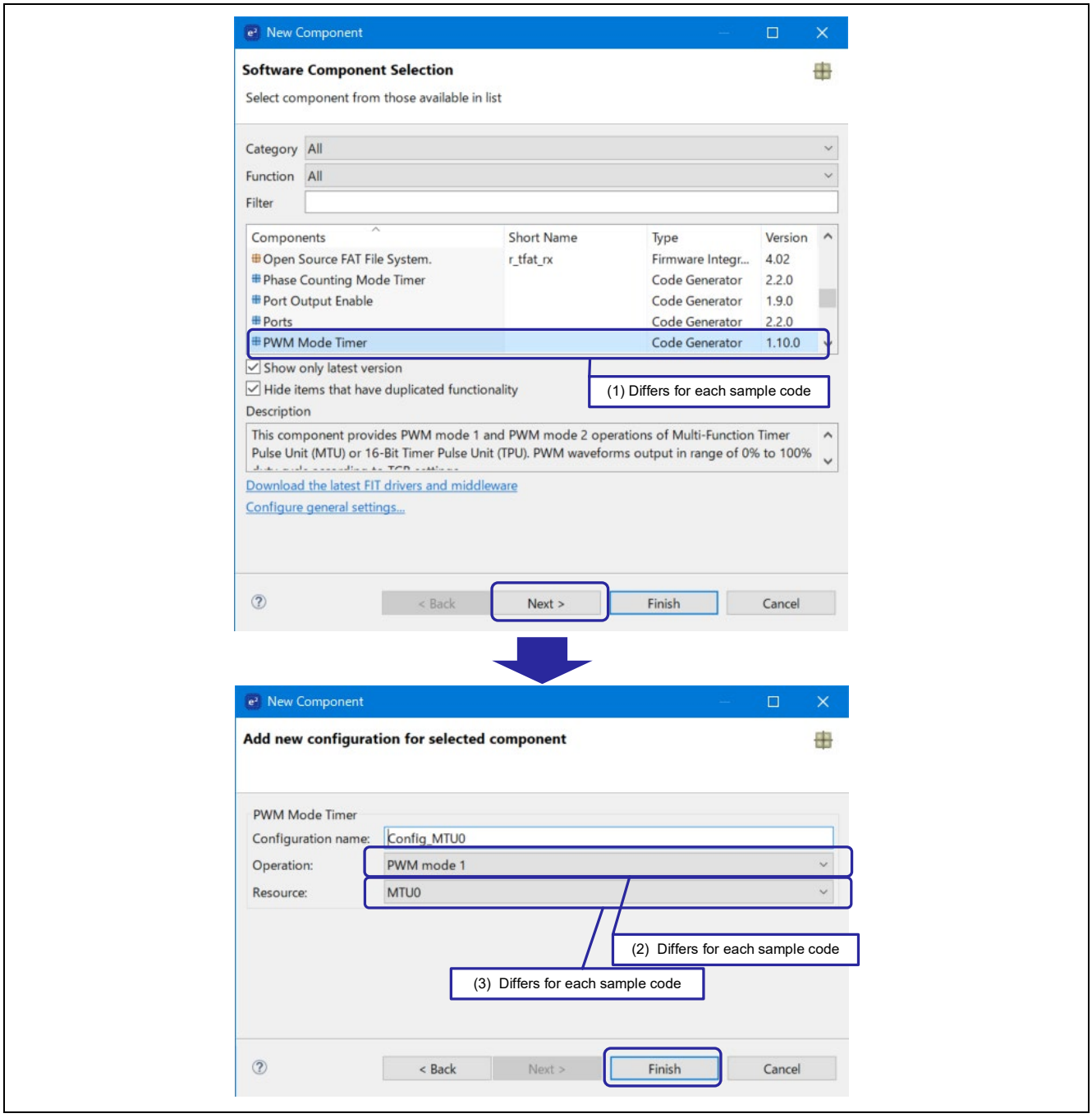


Figure 3.2 Adding Components

3.1.5 Pin Settings

Figure 3.3 shows an example of pin settings using the Smart Configurator.

Configure the pins after setting the MTU. For MTU settings, refer to “Smart Configurator Settings” for each sample code.

Pin settings are carried out in the R_Config_MTUn_Create function generated by the Smart Configurator.

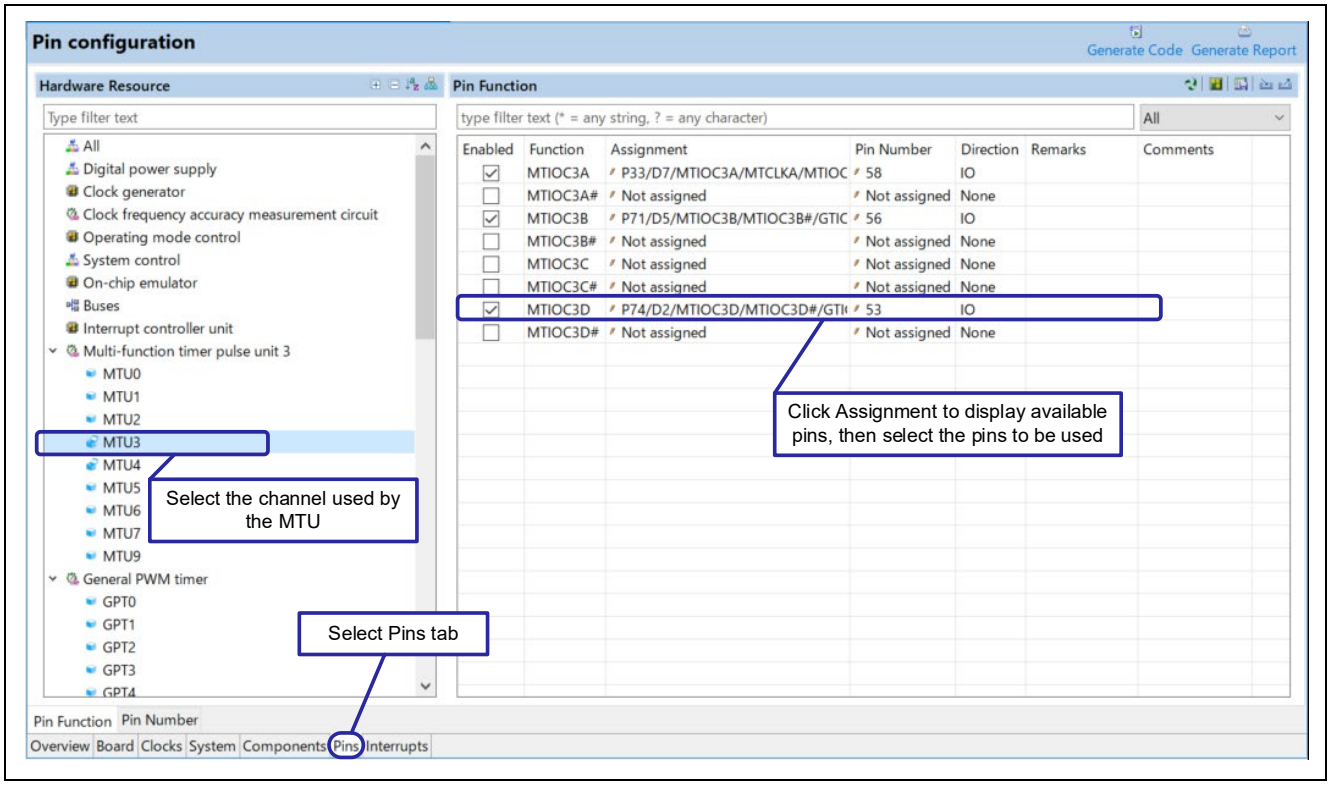


Figure 3.3 Pin Settings

3.1.6 Interrupt Settings

Figure 3.4 shows an example of interrupt settings using the Smart Configurator. For details on Software Configurable Interrupt A, refer to Renesas RX66T Group User's Manual Hardware, section 14.4.5.1 Software Configurable Interrupt A.

Configure interrupts after setting the MTU settings. For MTU settings, refer to "Smart Configurator Settings" for each sample code.

Interrupt settings can be configured in the R_Config_MTUn_Create function, R_Config_MTUn_Start function, and R_Config_MTUn_Stop function, all of which are generated by the Smart Configurator.

The interrupt handler function is created with the name r_Config_MTUn_[interrupt name]_interrupt in the Config_MTUn_user.c file generated by the Smart Configurator.

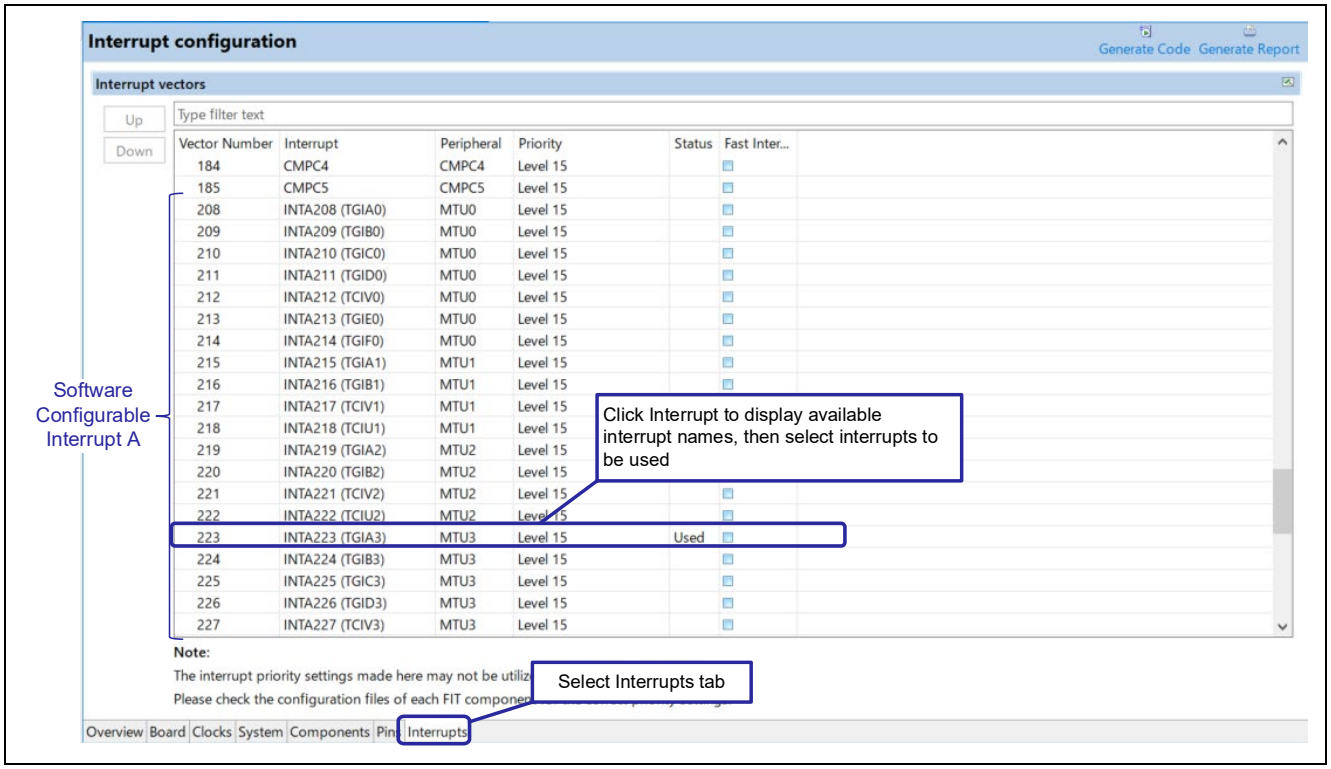


Figure 3.4 Interrupt Settings

3.2 12-Phase PWM Output in PWM Mode 2

- Target sample code file name: r01an6282_rx66t_mtu3_pwm2_sync.zip

3.2.1 Overview

In MTU PWM mode 2, up to 12 phases of PWM waveforms can be output in synchronization.

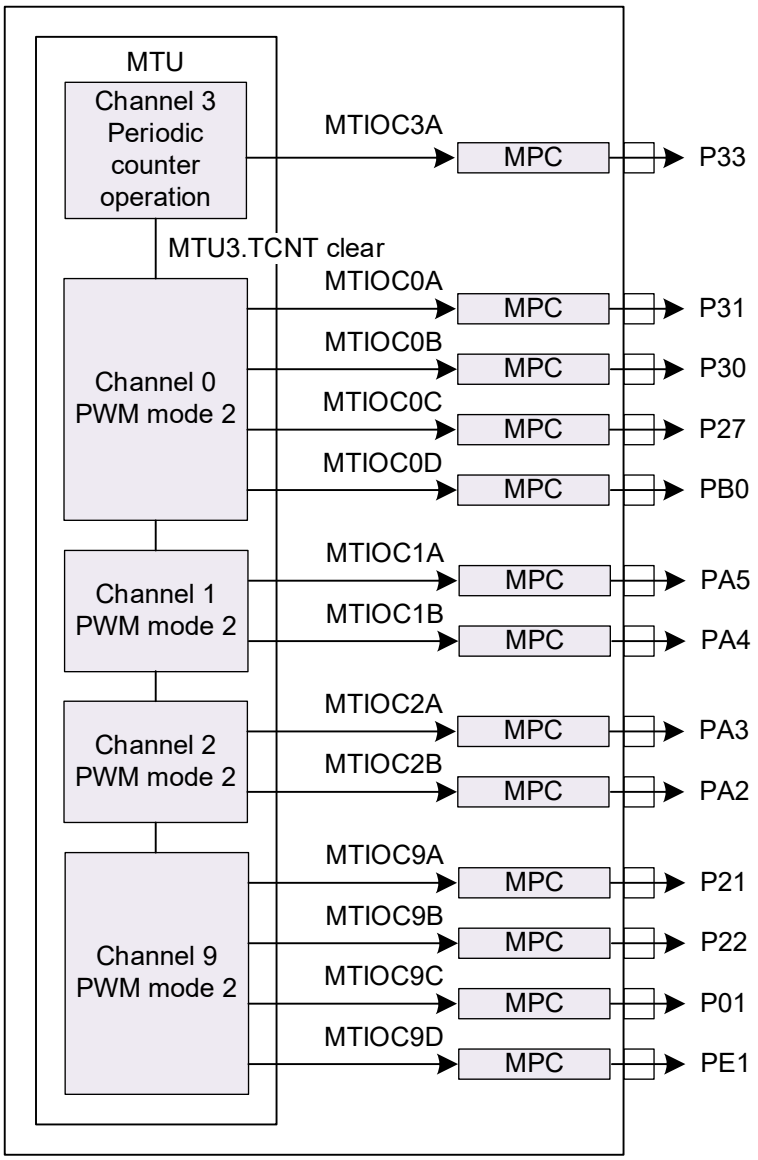
This sample code describes a case in which 12-phase PWM is output by using the TCSYSTR register to perform software synchronous start for MTU0 to MTU3 and MTU9, and using the timer counter (TCNT) clear of MTU3 (channel 3), which cannot be set to PWM mode 2, as synchronous clearing for MTU0 to MTU2 and MTU9 (channels 0 to 2 and 9).

The following list provides the MTU settings used in the sample code.

- MTU3 (channel 3)
  - Use normal mode timer
  - Set to synchronous operation
  - Carrier period = 1ms
  - Timer count clock = 40MHz (PCLKC/4)
  - Use MTU3.TGRA as period register
    - Timer counter clear source = MTU3.TGRA compare match
  - Toggle output at TGRA compare match
- MTU0 to MTU2 and MTU9 (channels 0 to 2 and 9)
  - Use PWM mode 2
  - Set to synchronous operation
  - Initial output value = low
  - Timer count clock = 40MHz (PCLKC/4)
  - Counter clear source = counter clear of channel 3 in synchronous operation
  - Use MTU0.TGRA as duty register
    - High output at MTU0.TGRA compare match
  - Use MTU0.TGRB as duty register
    - High output at MTU0.TGRB compare match
  - Use MTU0.TGRC as duty register
    - High output at MTU0.TGRC compare match
  - Use MTU0.TGRD as duty register
    - High output at MTU0.TGRD compare match
  - Use MTU1.TGRA as duty register
    - High output at MTU1.TGRA compare match
  - Use MTU1.TGRB as duty register
    - High output at MTU1.TGRB compare match
  - Use MTU2.TGRA as duty register
    - High output at MTU2.TGRA compare match
  - Use MTU2.TGRB as duty register
    - High output at MTU2.TGRB compare match
  - Use MTU9.TGRA as duty register
    - High output at MTU9.TGRA compare match
  - Use MTU9.TGRB as duty register
    - High output at MTU9.TGRB compare match
  - Use MTU9.TGRC as duty register
    - High output at MTU9.TGRC compare match
  - Use MTU9.TGRD as duty register
    - High output at MTU9.TGRD compare match

Set in Smart Configurator.
For Setting Methods,
refer to section 3.2.3.

The structure of this sample code is shown below.

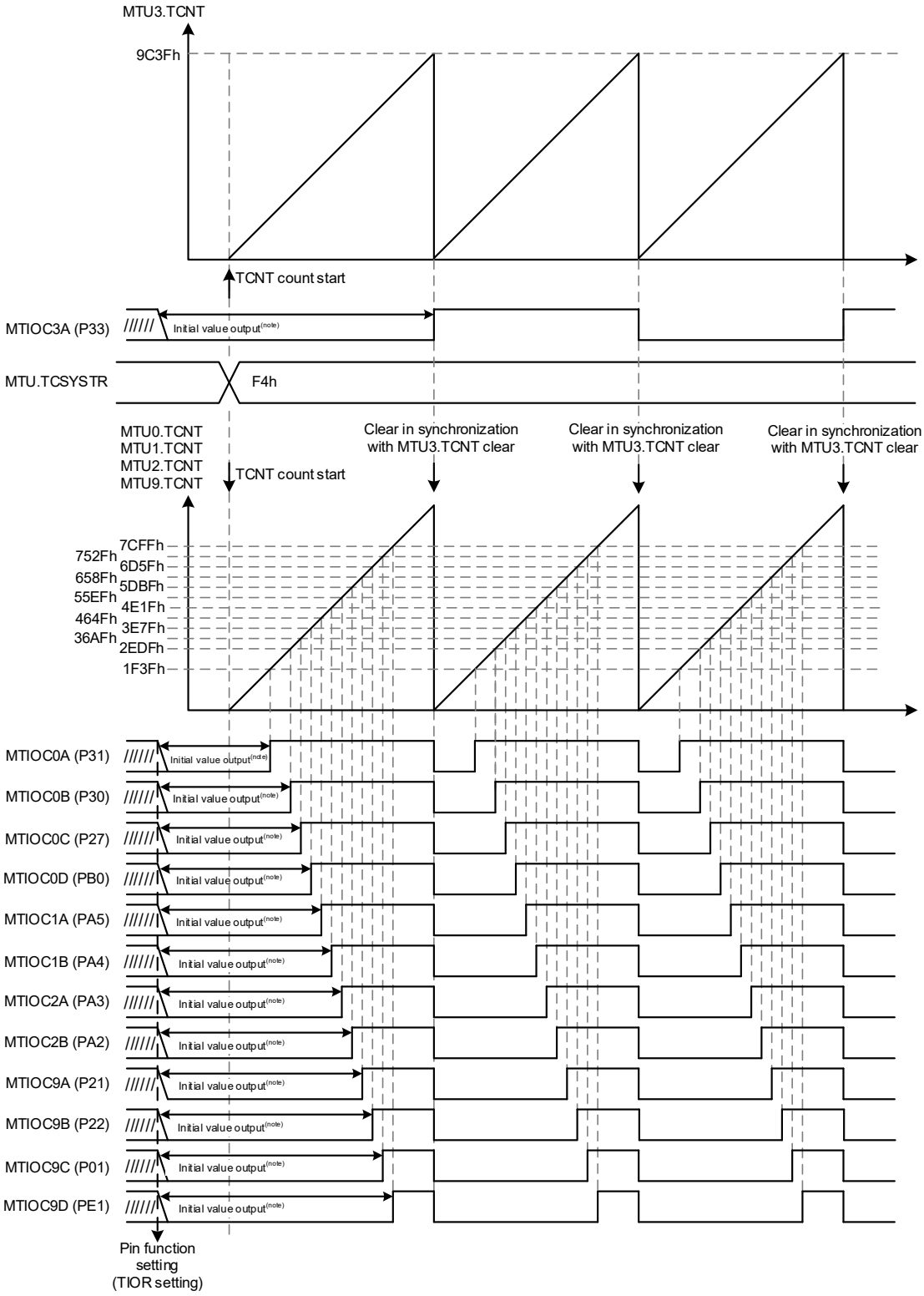


MTU3: Periodic counter operation
MTU0 to MTU2 and MTU9: Generate PWM mode 2 waveform
MPC: Sets pins to be used from general purpose I/O port to peripheral function I/O port

Figure 3.5 Sample Code Structure

3.2.2 Operation Details

The sample code operations are shown below. Use the TGRA of MTU3 as the period register, which cannot be set to PWM mode 2. The TCNT of MTU3 is cleared by a TGRA compare match. The TCNTs of MTU0 to MTU2 and MTU9 are cleared in synchronization with the clearing of the TCNT of MTU3 clearing.



Note: The sample code waveform starts outputting the initial values when the PMR register is set.

Figure 3.6 Sample Code Operations

3.2.3 Smart Configurator Settings

The sample code uses the Smart Configurator to add the MTU as described below. For details on how to add components, refer to section 3.1.4 Adding Components.

Table 3.4 Adding Components (MTU3)

Item	Description
Component	Normal Mode Timer
Configuration name	Config_MTU3
Input capture/ Output compare pins	4 pins
Resource	MTU3

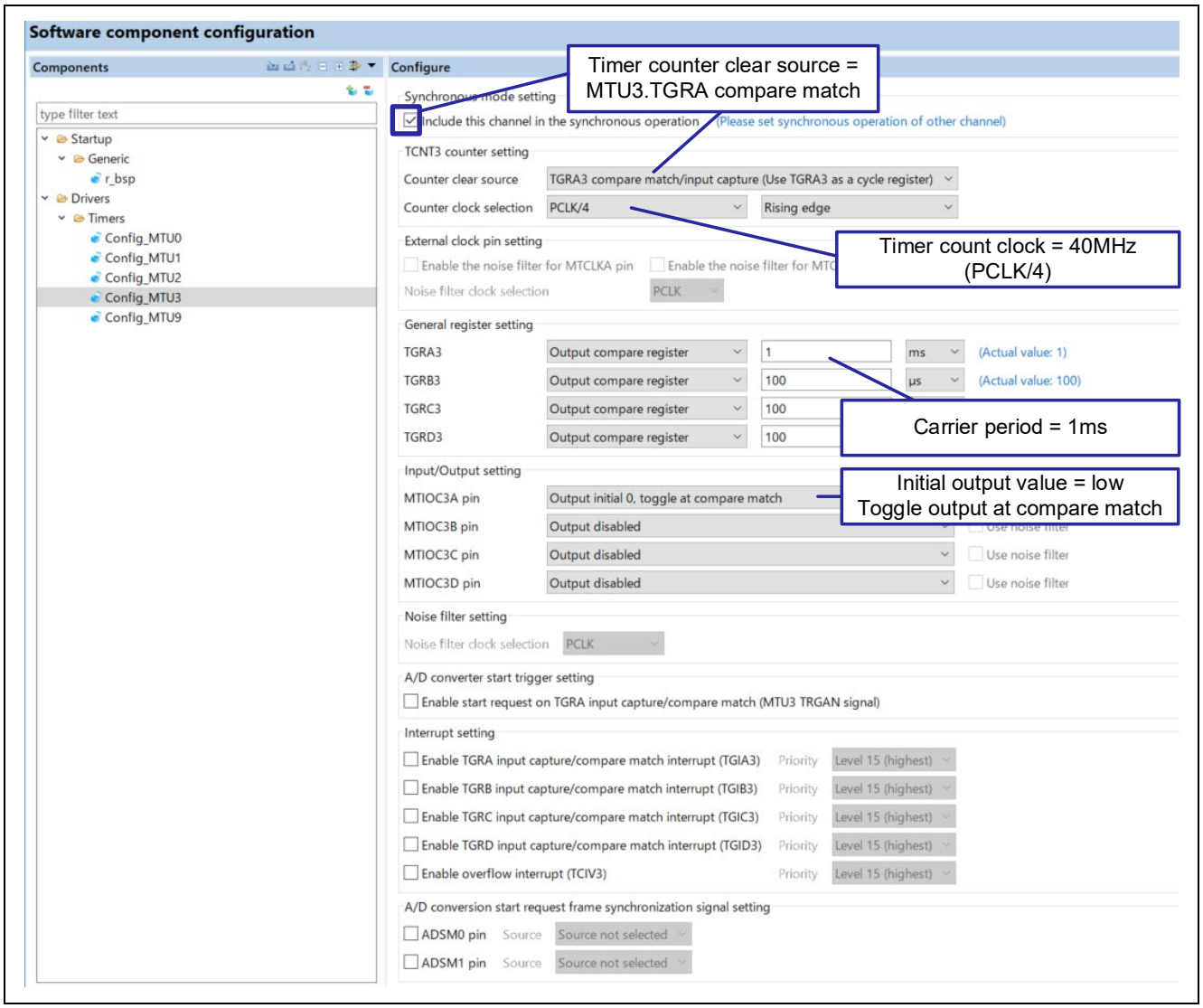


Figure 3.7 MTU3 Settings

Table 3.5 Adding Components (MTU0 to MTU2 and MTU9)

Item	Description			
Component	PWM Mode Timer			
Configuration name	Config_MTU0	Config_MTU1	Config_MTU2	Config_MTU9
Operation	PWM Mode 2			
Resource	MTU0	MTU1	MTU2	MTU9

Figure 3.8 shows the Config_MTU0 settings. The settings for MTU1, MTU2, and MTU9 are basically the same. As duty cycles differ, refer to Figure 3.9 to Figure 3.11 for TGRA, TGRB, TGRC and TGRD settings.

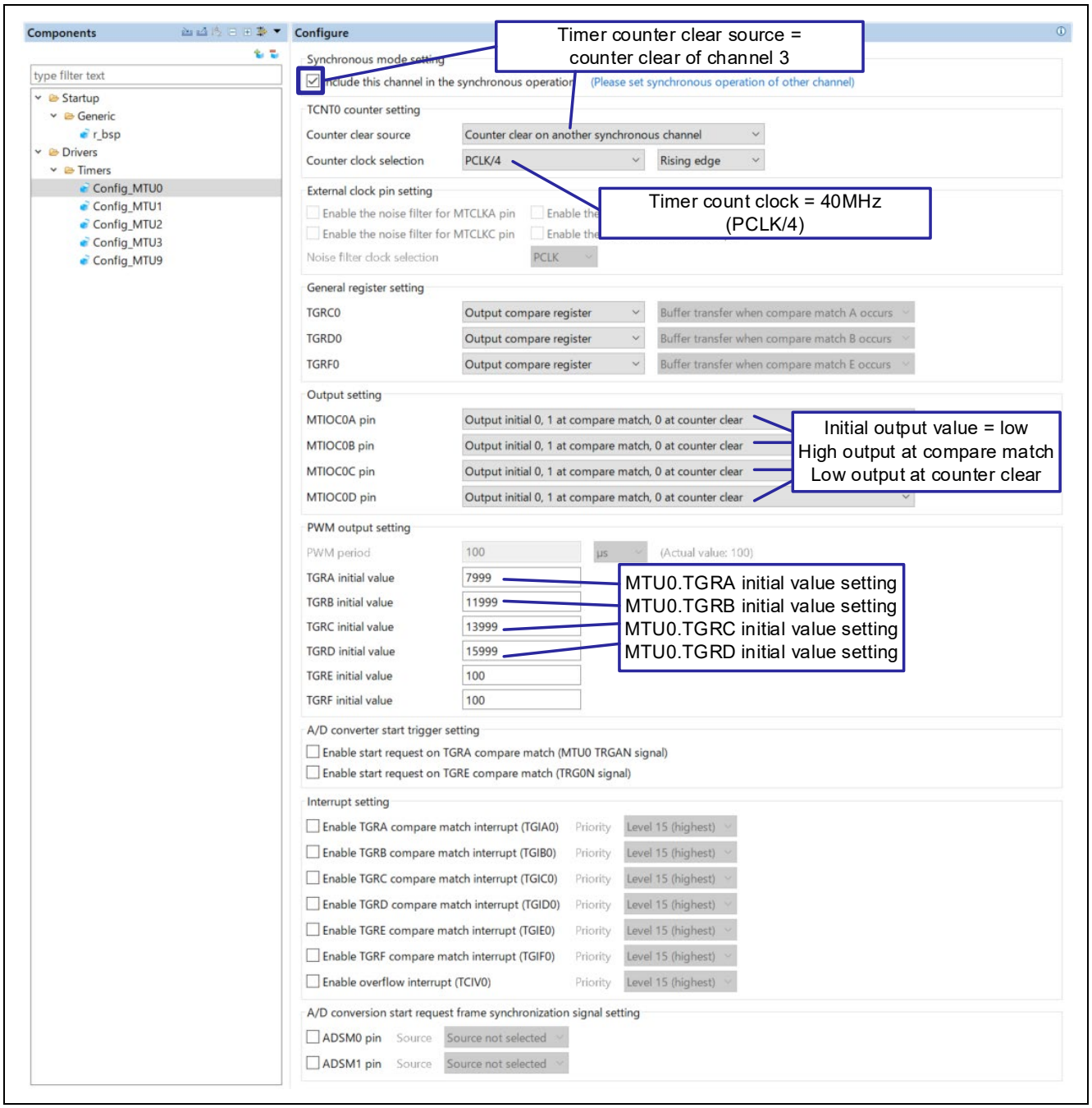


Figure 3.8 MTU0 Settings

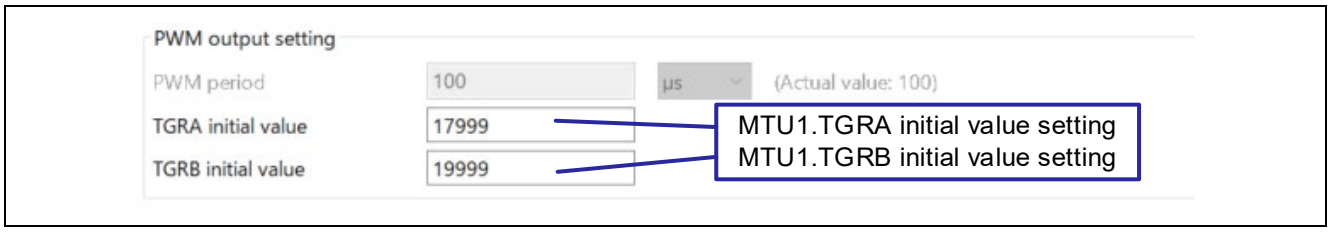


Figure 3.9 MTU1 Settings (TGRA and TGRB Compare Match Register Settings)

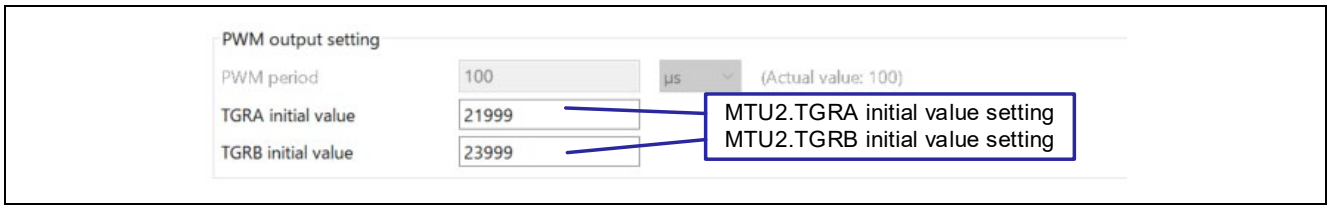


Figure 3.10 MTU2 Settings (TGRA and TGRB Compare Match Register Settings)

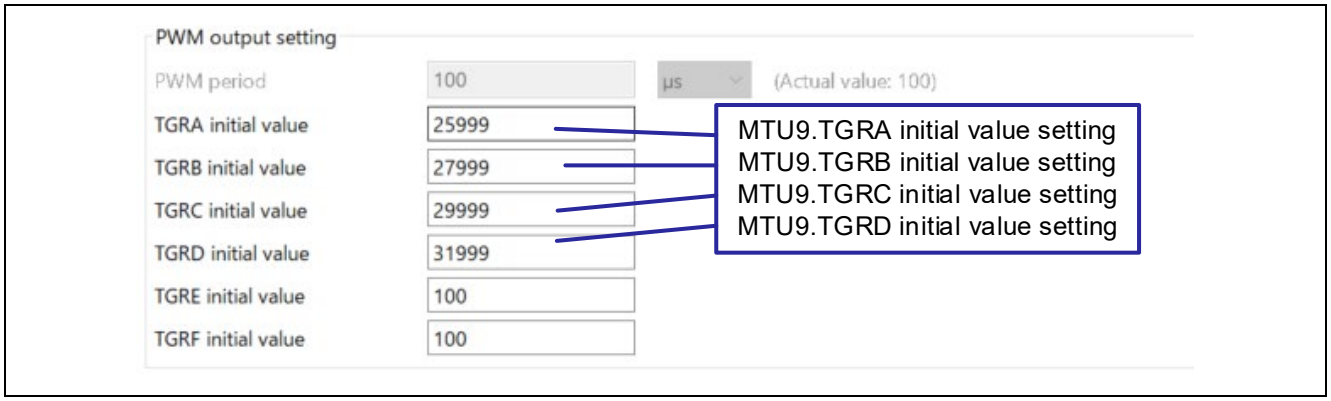


Figure 3.11 MTU9 Settings (TGRA, TGRB, TGRC and TGRD Compare Match Register Settings)

3.2.4 Flowcharts

The following shows the processing of a component added after code generation by the Smart Configurator.

In the main function, count start function mtu_start is read and counting is started.

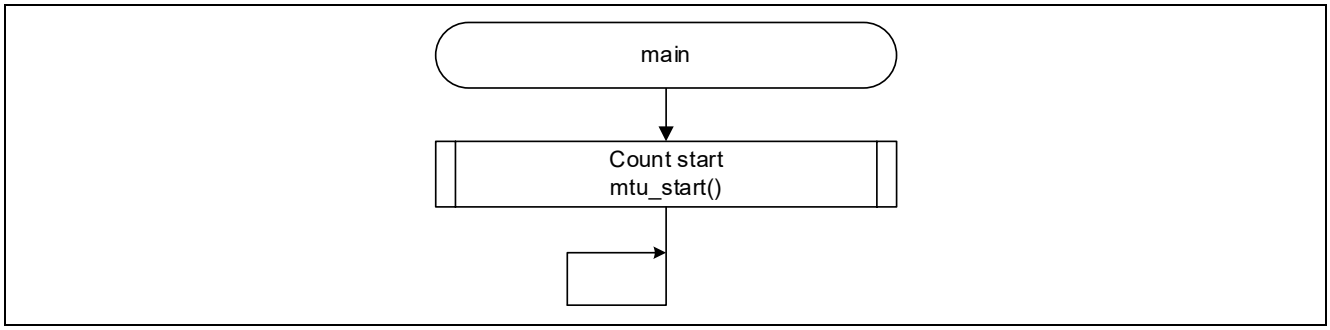


Figure 3.12 main Function

Counting is started for MTU0 to MTU3 and MTU9 in the count start function.

This function is newly created after code generation by the Smart Configurator.

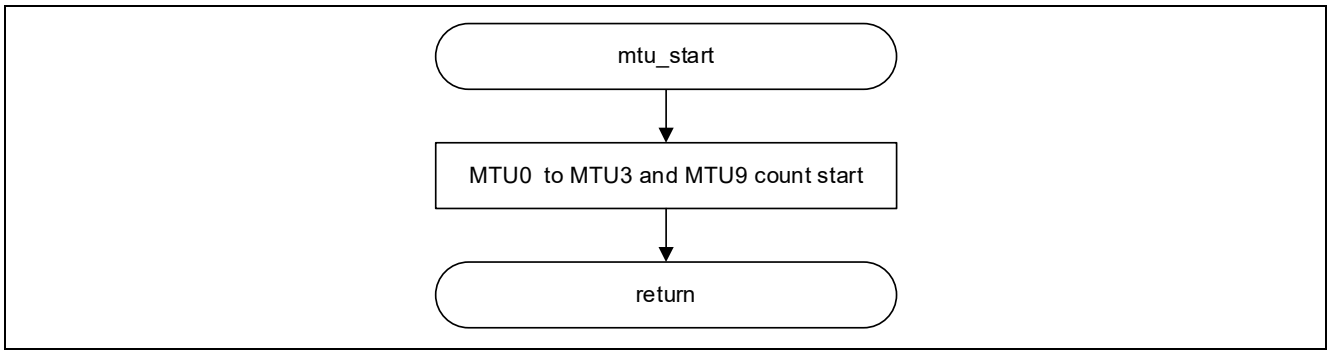


Figure 3.13 Count Start Function

3.2.5 Usage Notes

3.2.5.1 Counting Starts for Multiple Channels

In this sample code, the SCH0 to SCH3 and SCH9 bits of timer counter synchronous start register TCSYSTR are set at the same time in the mtu_start function to start counting multiple channels at the same time.

When using the R_Config_MTUm_Start (m = 0 to 3, 9) function generated by the Smart Configurator, the counting starts timing may not be the same because each of the functions are read.

The MTU0 to MTU3 and MTU9 counting can be started simultaneously by setting the CST0 to CST3 and CST9 bits of timer start register TSTRA at the same time.

For details, refer to RX66T Group User's Manual: Hardware, section 22.2.17 Timer Start Registers (TSTRA, TSTRB, TSTR) and 22.2.19 Timer Counter Synchronous Start Register (TCSYSTR).

3.2.5.2 Contention between Overflow/Underflow and Counter Clearing

If an overflow/underflow and a counter clearing occur simultaneously, neither a TCIV interrupt nor a TCIU interrupt is generated and the TCNT clearing takes precedence.

For details, refer to RX66T Group User's Manual: Hardware, section 22.6.17 Contention between Overflow/Underflow and Counter Clearing.

3.3 Inter-Channel Synchronous Clearing Using Compare Match

- Target sample code file name: r01an6282_rx66t_mtu3_cmp_sync.zip

3.3.1 Overview

Synchronous operation can be used to modify multiple TCNT values at the same time (synchronous setting). Multiple TCNT values can be set to 0000h at the same time by the TCR register setting (synchronous clearing).

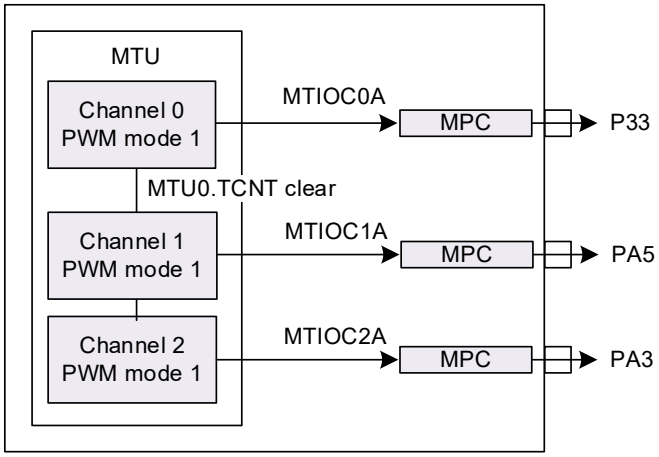
This section describes a sample code in which MTU0 to MTU2 are set to synchronous operation and PWM mode 1, and MTU1 and MTU2 are synchronously cleared by counter clear source MTU0. MTU0 to MTU2 use the TCSYSTR register to perform synchronous start by software.

The following list provides the MTU settings used in the sample code.

- MTU0 (channel 0)
  - Use PWM mode 1
  - Set to synchronous operation
  - Initial output value = low
  - Carrier period = 1ms
  - Timer count clock = 40MHz (PCLKC/4)
  - Use MTU0.TGRB as period register
    - Timer counter clear source = MTU0.TGRB compare match
    - Toggle output at TGRB compare match
  - Use MTU0.TGRA as duty register
    - Toggle output at TGRA compare match
- MTU1 and MTU2 (channels 1 and 2)
  - Use PWM mode 1
  - Set to synchronous operation
  - Initial output value = low
  - Timer count clock = 40MHz (PCLKC/4)
  - Counter clear source = counter clear of channel 0 in synchronous operation
  - Use MTU1.TGRA as duty register
    - Toggle output at TGRA compare match
  - Use MTU1.TGRB as duty register
    - Toggle output at TGRB compare match
  - Use MTU2.TGRA as duty register
    - Toggle output at TGRA compare match
  - Use MTU2.TGRB as duty register
    - Toggle output at TGRB compare match

Set in Smart Configurator.
For Setting Methods,
refer to section 3.3.3.

The structure of this sample code is shown below.

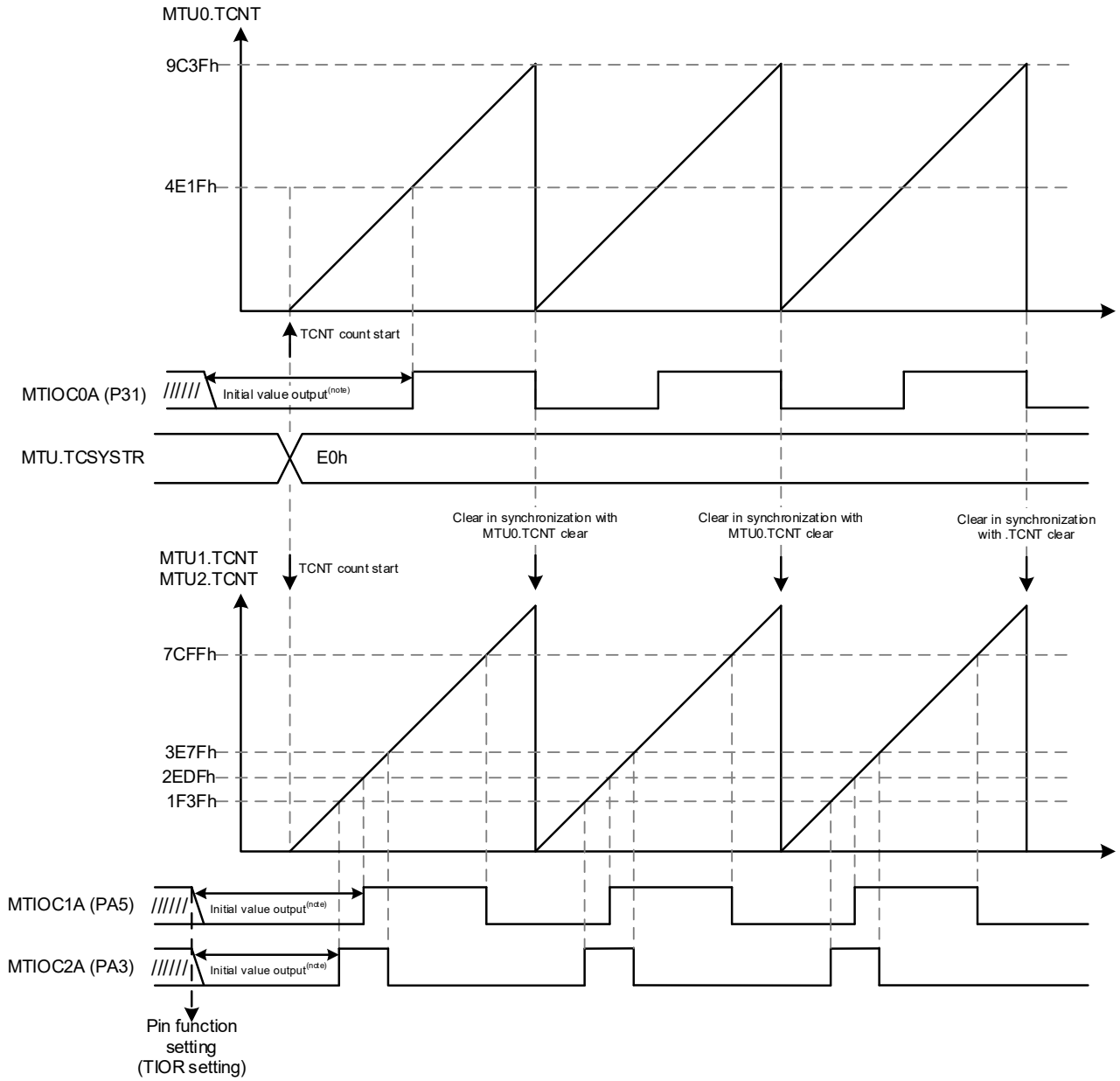


MTU0 to MTU2: Generate PWM mode 1 waveform
MPC: Sets pins to be used from general purpose I/O port to peripheral function I/O port

Figure 3.14 Sample Code Structure

3.3.2 Operation Details

The sample code operations are shown below. MTU0 to MTU2 are set to synchronous operation and PWM mode 1, the MTU0 counter clear source is set to MTU0.TGRB compare match, and MTU1 and MTU2 counter clear sources are set to synchronous clearing. Three-phase PWM waveform is output from the MTIOC0A, MTIOC1A, and MTIOC2A pins.



Note: The sample code waveform starts outputting the initial values when the PMR register is set.

Figure 3.15 Sample Code Operations

3.3.3 Smart Configurator Settings

The sample code uses the Smart Configurator to add the MTU as described below. For details on how to add components, refer to section 3.1.4 Adding Components.

Table 3.6 Adding Components (MTU0)

Item	Description
Component	PWM Mode Timer
Configuration name	Config_MTU0
Operation	PWM Mode 1
Resource	MTU0

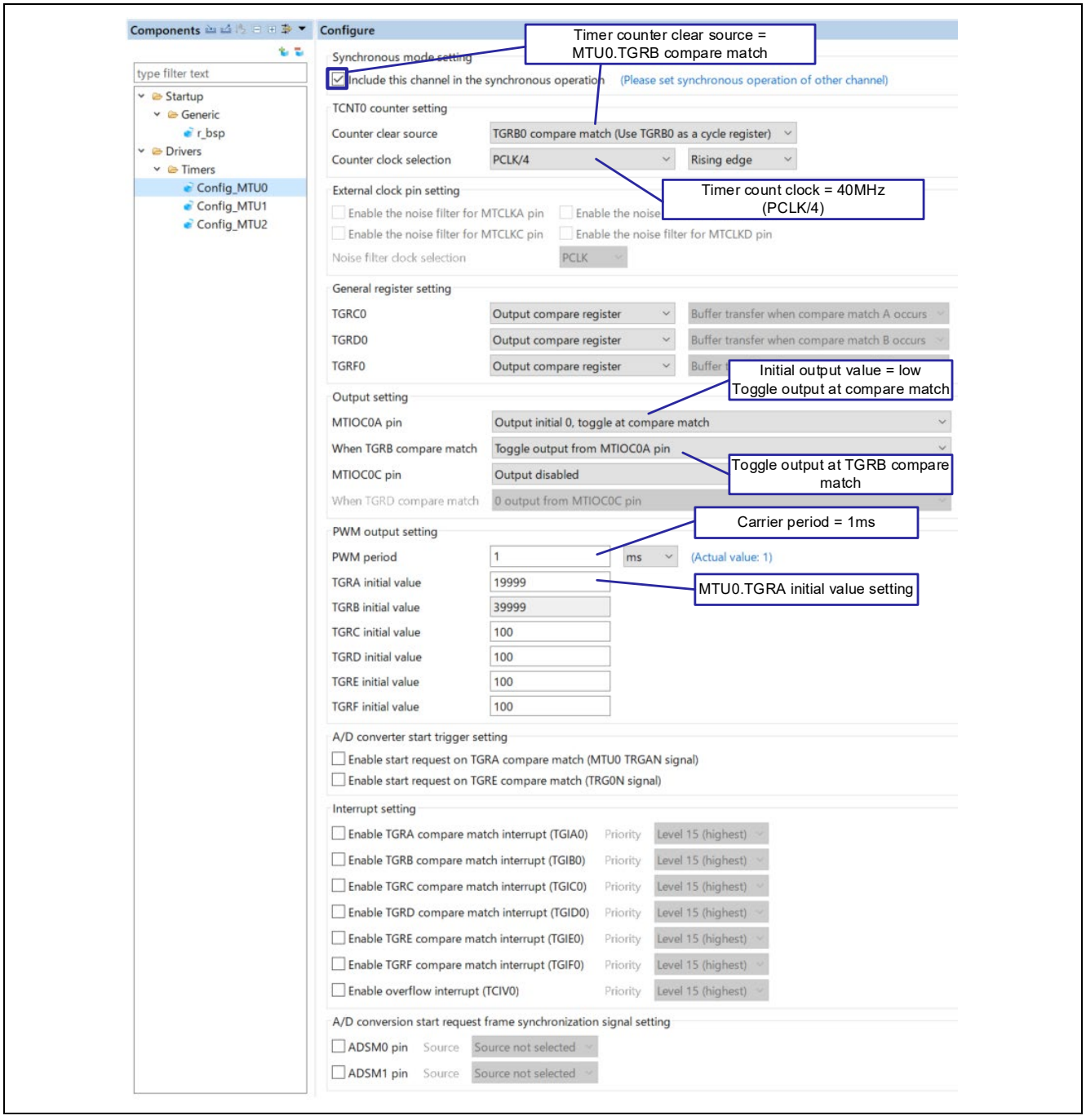


Figure 3.16 MTU0 Settings

Table 3.7 Adding Components (MTU1 and MTU2)

Item	Description	
Component	PWM Mode Timer	
Configuration name	Config_MTU1	Config_MTU2
Operation	PWM Mode 1	
Resource	MTU1	MTU2

Figure 3.17 shows the Config_MTU1 settings. The settings for MTU2 are basically the same. As duty cycles differ, refer to Figure 3.18 for TGRA and TGRB settings.

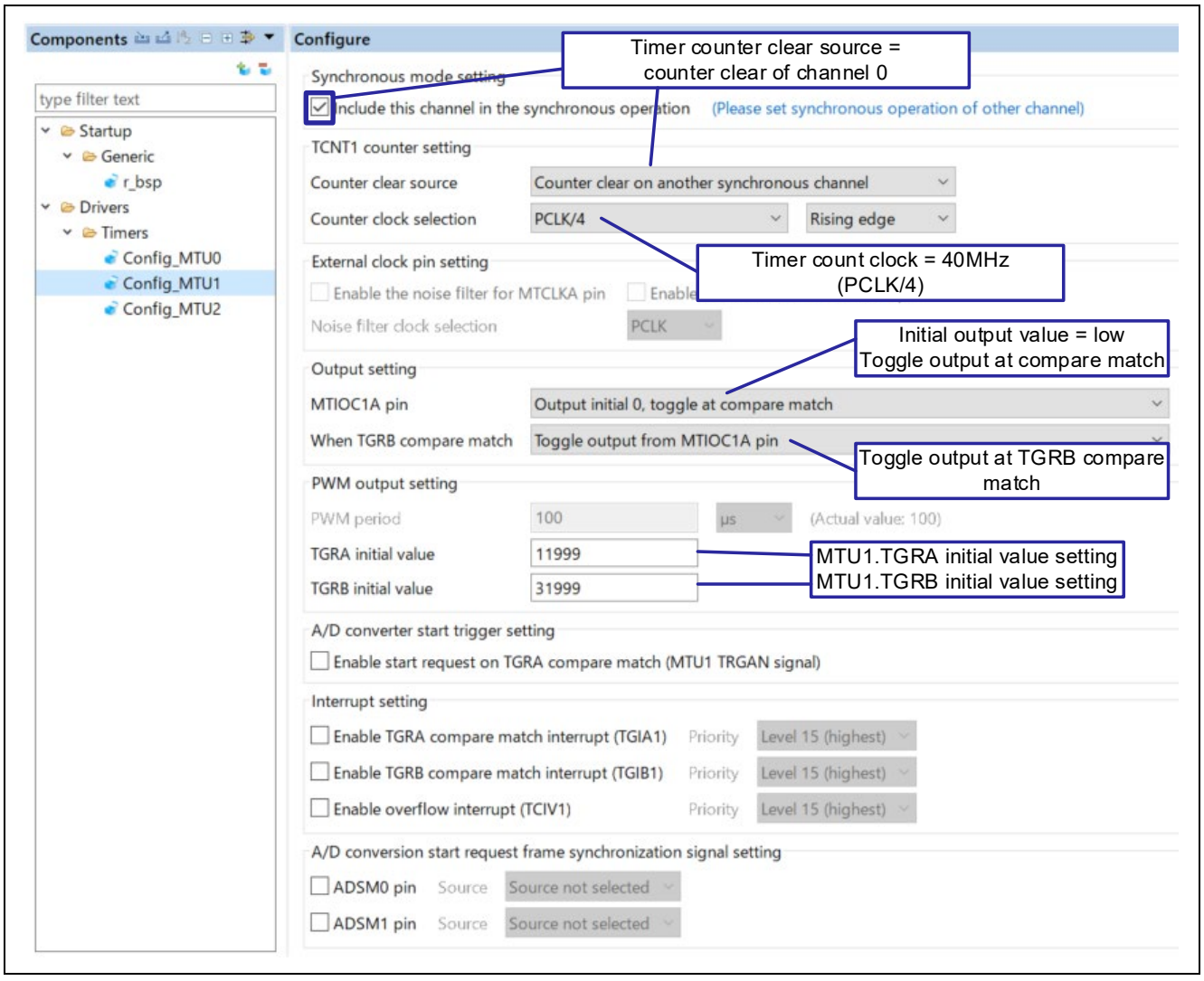


Figure 3.17 MTU1 Settings

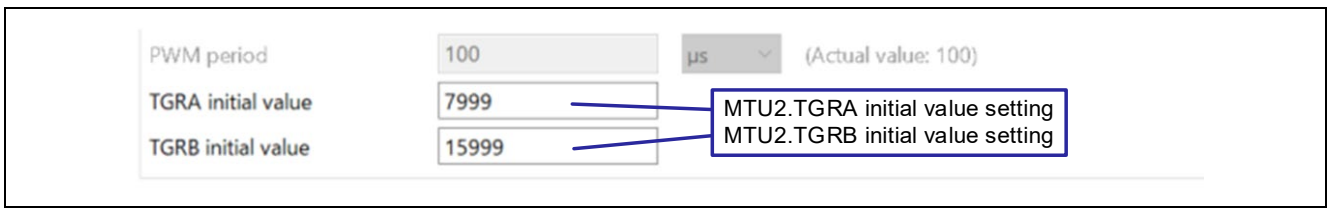


Figure 3.18 MTU2 Settings (TGRA and TGRB Compare Match Register Settings)

3.3.4 Flowcharts

The following shows the processing of a component added after code generation by the Smart Configurator.

In the main function, count start function mtu_start is read and counting is started.

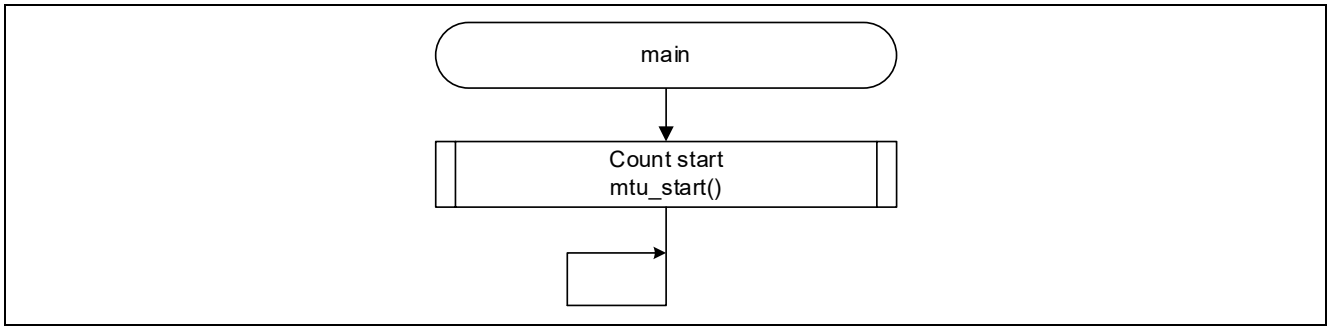


Figure 3.19 main Function

Counting is started for MTU0 to MTU2 in the count start function.

This function is newly created after code generation by the Smart Configurator.

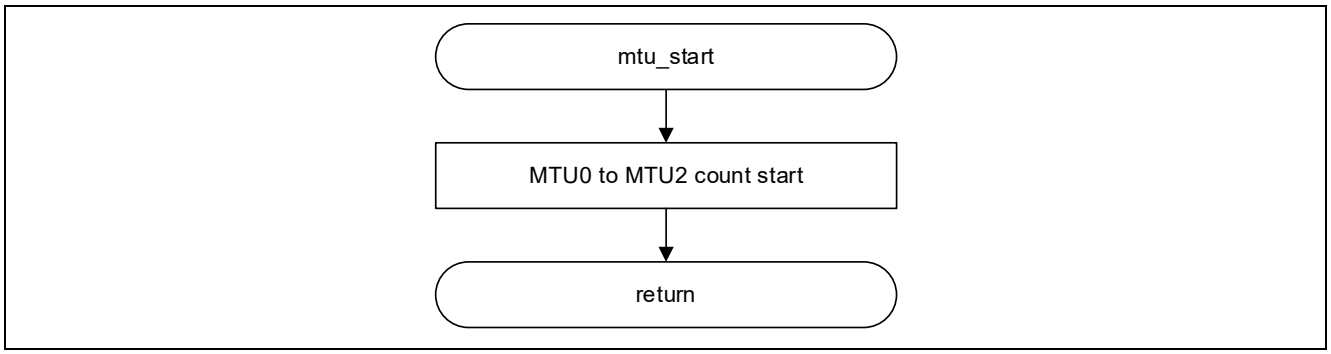


Figure 3.20 Count Start Function

3.3.5 Usage Notes

3.3.5.1 Counting Starts for Multiple Channels

In this sample code, the SCH0 to SCH2 bits of timer counter synchronous start register TCSYSTR are set at the same time in the mtu_start function in order to start counting multiple channels at the same time.

When using the R_Config_MTUm_Start (m = 0 to 2) function generated by the Smart Configurator, the counting starts timing may not be the same because each of the functions are read.

The MTU0 to MTU2 counting can be started simultaneously by setting the CST0 to CST2 bits of timer start register TSTRA at the same time.

For details, refer to RX66T Group User's Manual: Hardware, section 22.2.17 Timer Start Registers (TSTRA, TSTRB, TSTR) and 22.2.19 Timer Counter Synchronous Start Register (TCSYSTR).

3.3.5.2 Synchronous Operation Group

Synchronous operation (set/clearing) can be performed in the following two groups. Synchronous operation can be carried out within the group, but not with another group. Also, MTU5 does not support synchronous operation.

- Group A: MTU0, MTU1, MTU2, MTU3, MTU4, and MTU9
- Group B: MTU6 and MTU7

The following table shows the relationships between synchronous operation and synchronous start/stop and the registers.

Table 3.8 Relationships between Group and Register

Operation	Group A (MTU0 to MTU4, MTU9)	Group B (MTU6, MTU7)
Setting of intended channel of synchronous operation	TSYRA	TSYRB
Count synchronous start/stop	TSTRA	TSTRB
Count synchronous start	TCSYSTR	

3.3.5.3 Contention between Overflow/Underflow and Counter Clearing

If an overflow/underflow and a counter clearing occur simultaneously, neither a TCIV interrupt nor a TCIU interrupt is generated and the TCNT clearing takes precedence.

For details, refer to RX66T Group User's Manual: Hardware, section 22.6.17. Contention between Overflow/Underflow and Counter Clearing.

3.4 5-Phase Complementary PWM Output

- Target sample code file name: r01an6282_rx66t_mtu3_complementary_sync.zip

3.4.1 Overview

6-phase complementary PWM can be output by synchronizing operations of MTU3 and MTU4 with MTU6 and MTU7.

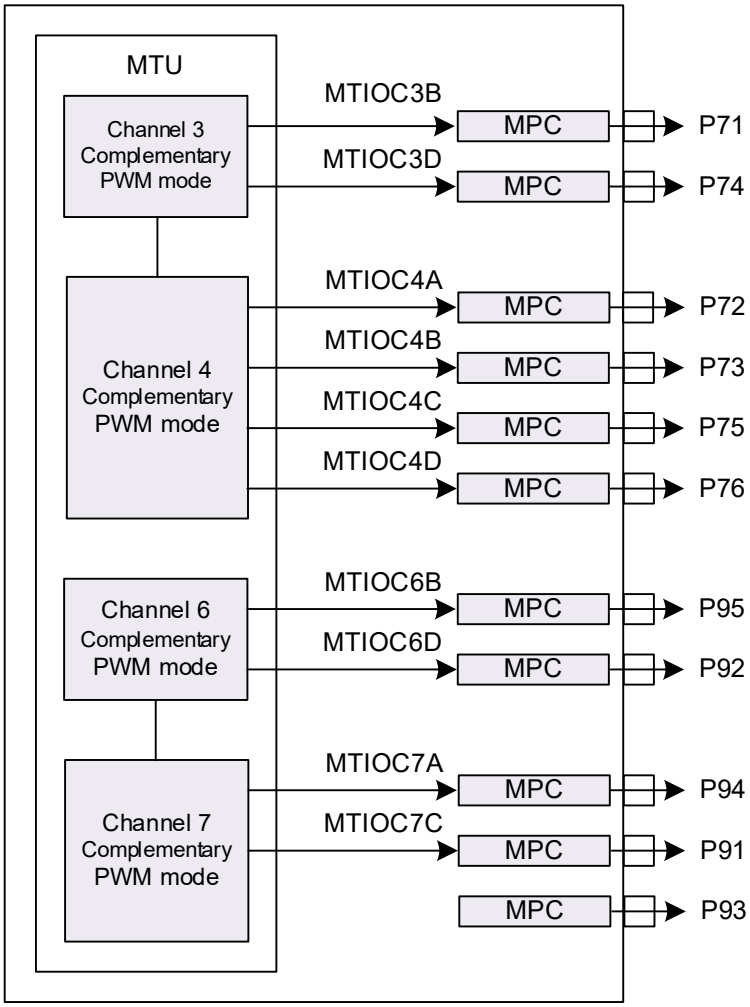
This section describes a sample code in which MTU3 and MTU4 are synchronously started with MTU6 and MTU7 to provide 5-phase complementary PWM output. The unused 1-phase pin (P93) of MTU7 is used as a general purpose I/O port.

The following list provides the MTU and PORT settings used in the sample code

- MTU3, MTU4, MTU6, and MTU7 (channels 3, 4, 6, and 7)
  - Use complementary PWM mode 2 (transfer at trough)
  - Carrier period = 1ms
  - Dead time = 30µs
  - Timer count clock = 40MHz (PCLKC/4)
  - Set MTUn.TGRA to MTUn.TCNT upper limit value (n = 3, 6) (1/2 carrier period + dead time)
  - Set buffer transfer timing
    - Transfers data at the trough of the count
  - Initial output value is high, active level is low
  - Use MTUn.TGRB (n = 3, 6) as U-phase duty register
    - Positive-phase:
      Low output at up-counting compare match
      High output at down-counting compare match
    - Negative-phase:
      High output at up-counting compare match
      Low output at down-counting compare match
  - Use MTUn.TGRA (n = 4, 7) as V-phase duty register
    - Positive-phase:
      Low output at up-counting compare match
      High output at down-counting compare match
    - Negative-phase:
      High output at up-counting compare match
      Low output at down-counting compare match
  - Use MTUn.TGRB (n = 4) as W-phase duty register
    - Positive-phase:
      Low output at up-counting compare match
      High output at down-counting compare match
    - Negative-phase:
      High output at up-counting compare match
      Low output at down-counting compare match
  - Enable underflow interrupt
- PORT
  - Use P93 as general purpose I/O port

Set in Smart Configurator.
For Setting Methods,
refer to section 3.4.3.

The structure of this sample code is shown below.



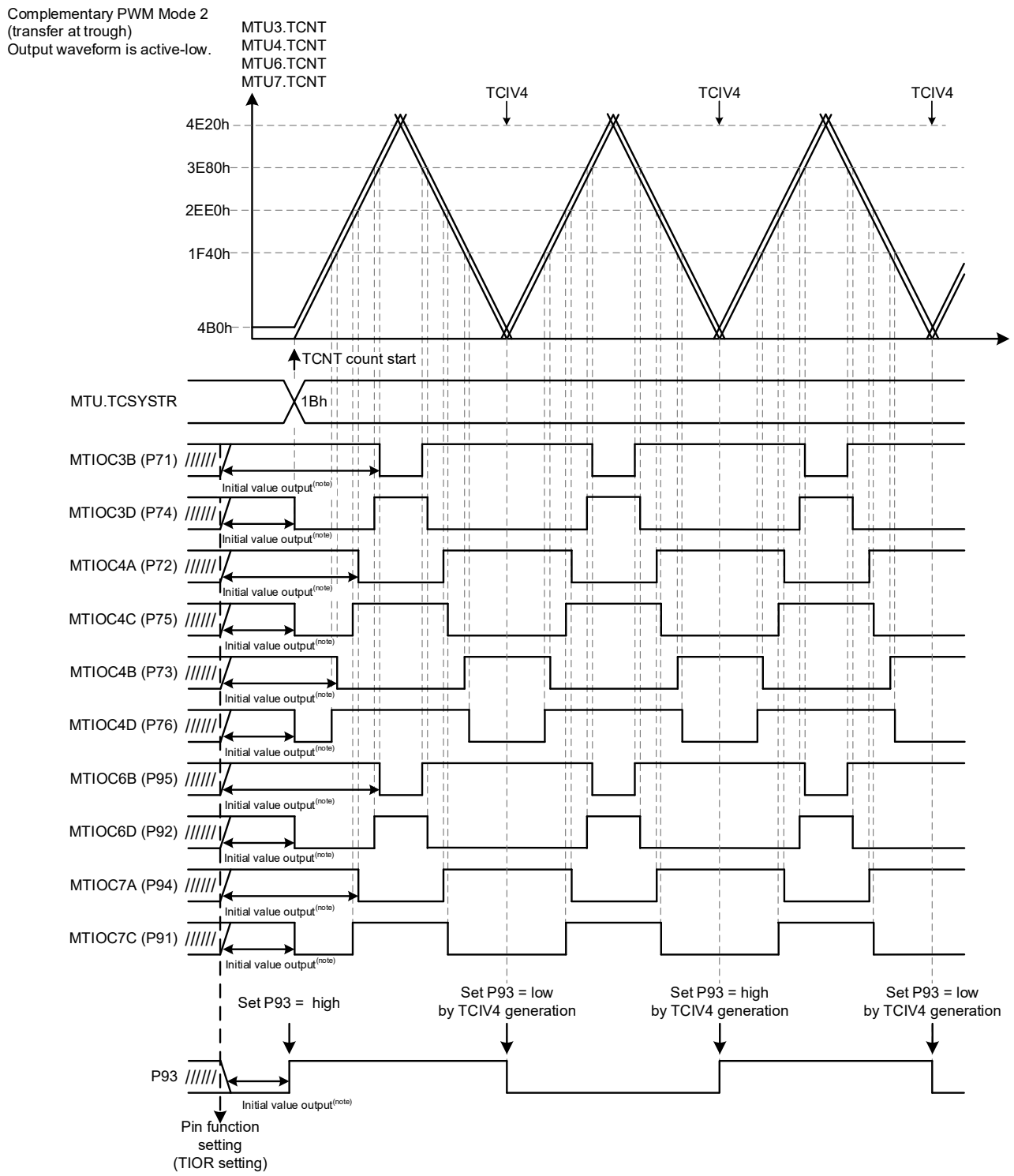
MTU3, MTU4, MTU6, and MTU7: Generate complementary PWM mode waveform
MPC: Sets pins to be used, other than P93, from general purpose I/O port to peripheral function I/O port
Sets only P93 to general purpose I/O port

Figure 3.21 Sample Code Structure

3.4.2 Operation Details

The sample code operations are shown below. Synchronous start is performed by setting MTU3, MTU4, MTU6, and MTU7 to complementary PWM mode 2 (transfer at trough) and setting 1Bh to the MTU.TCSYSTR register.

P93 is set to high just before the MTU counting starts, and output is toggled for each underflow interrupt (TCIV4) generation.



Note: The sample code waveform starts outputting the initial values when the PMR register is set.

Figure 3.22 Sample Code Operations

3.4.3 Smart Configurator Settings

The sample code uses the Smart Configurator to add the MTU as described below. For details on how to add components, refer to section 3.1.4 Adding Components.

Table 3.9 Adding Components (MTU3 and MTU4)

Item	Description
Component	Complementary PWM Mode Timer
Configuration name	Config_MTU3_MTU4
Operation	Complementary PWM Mode 2 (transfer at trough)
Resource	MTU3_MTU4

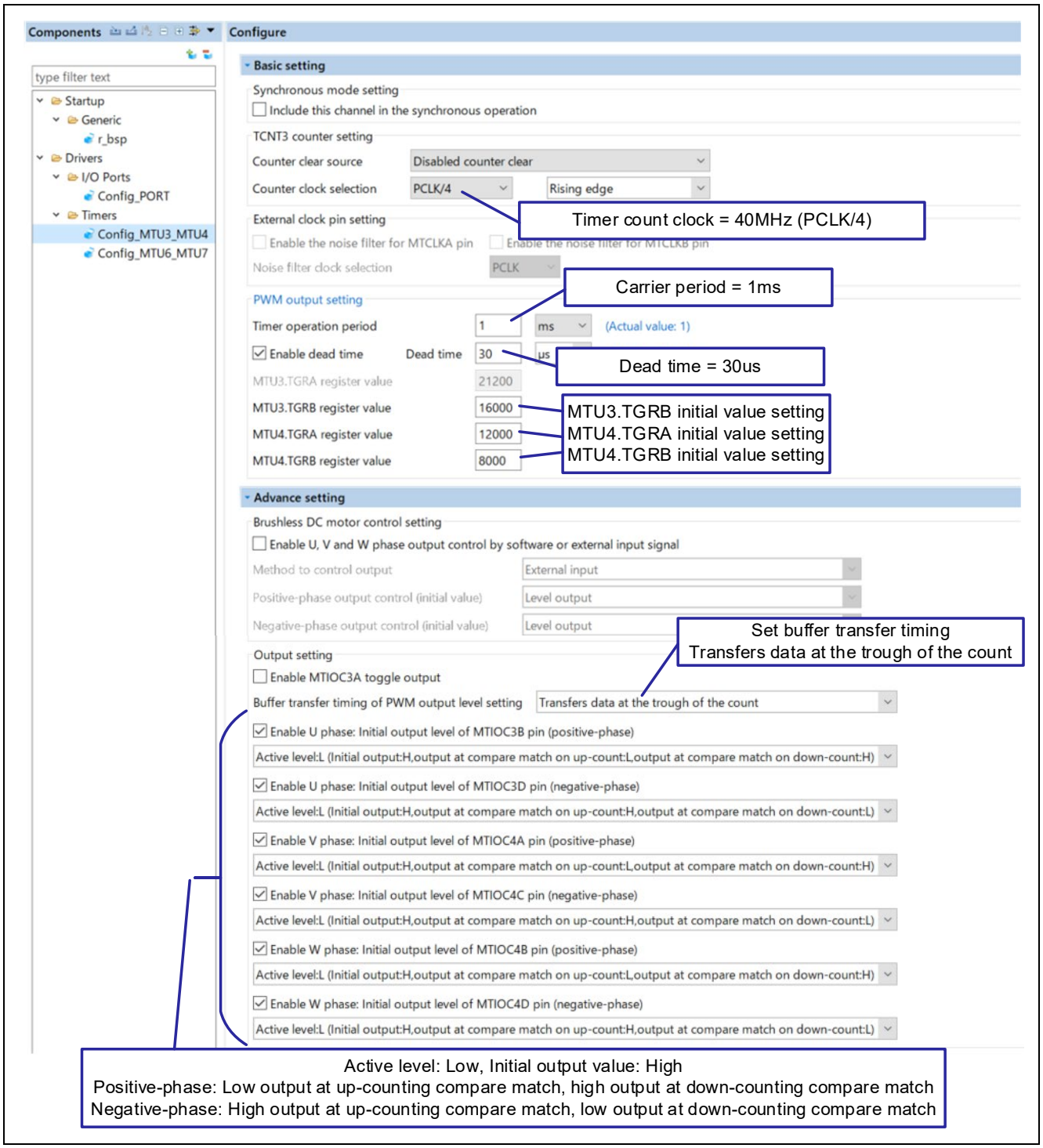


Figure 3.23 MTU3 and MTU4 Settings (1/2)

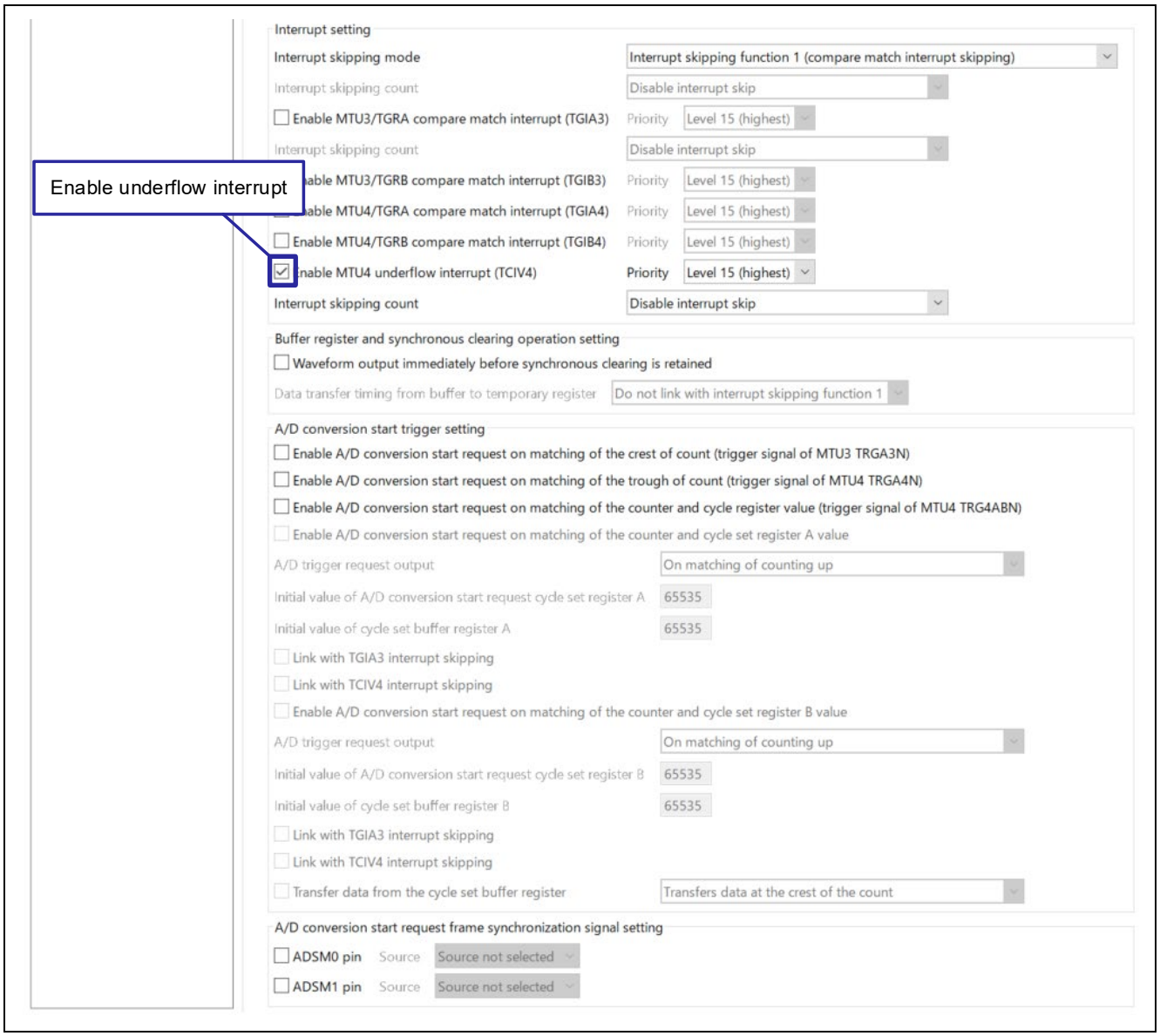


Figure 3.24 MTU3 and MTU4 Settings (2/2)

Table 3.10 Adding Components (MTU6 and MTU7)

Item	Description
Component	Complementary PWM Mode Timer
Configuration name	Config_MTU6_MTU7
Operation	Use complementary PWM mode 2 (transfer at trough)
Resource	MTU6_MTU7

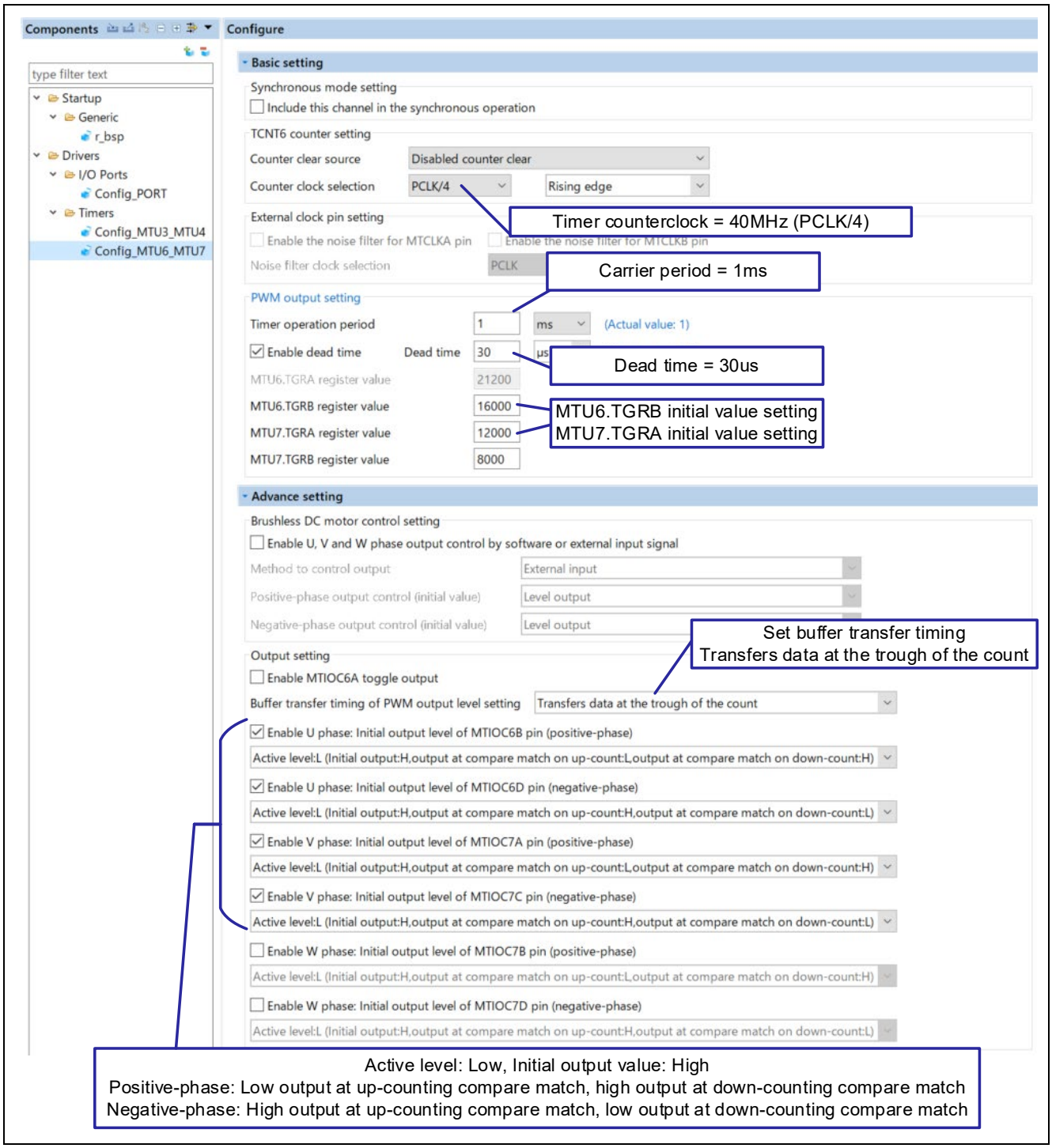


Figure 3.25 MTU6 and MTU7 Settings

When using P93 as a general purpose I/O port, add the PORT as shown below.

Table 3.11 Adding Components (PORT)

Item	Description
Component	Port
Configuration name	Config_PORT
Resource	PORT

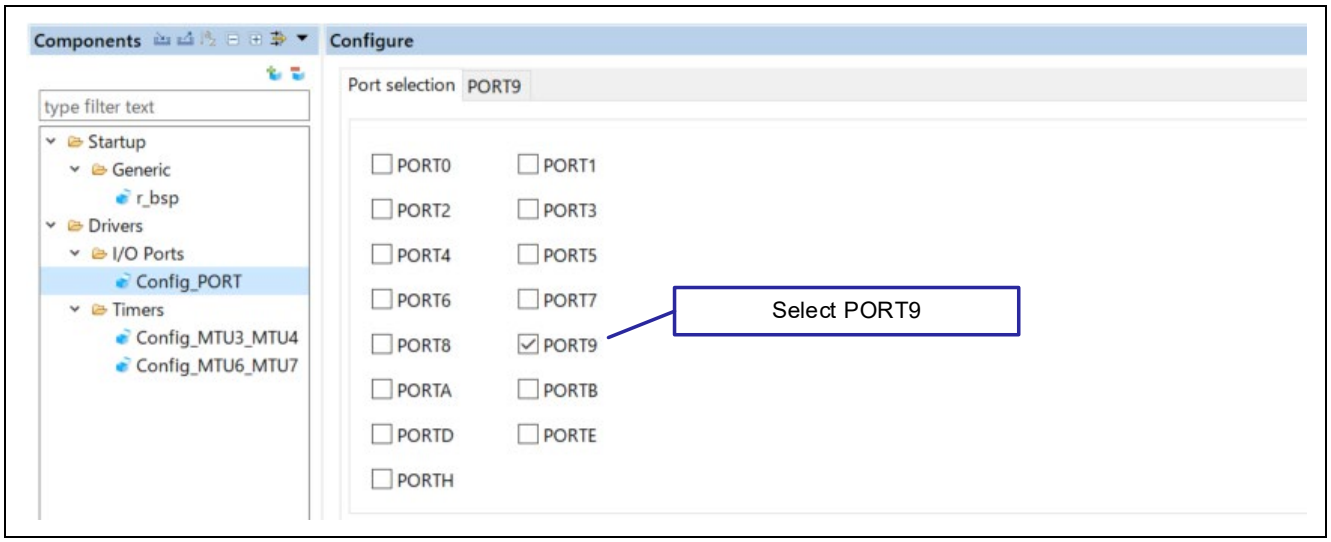


Figure 3.26 Settings for P93 (1/2)

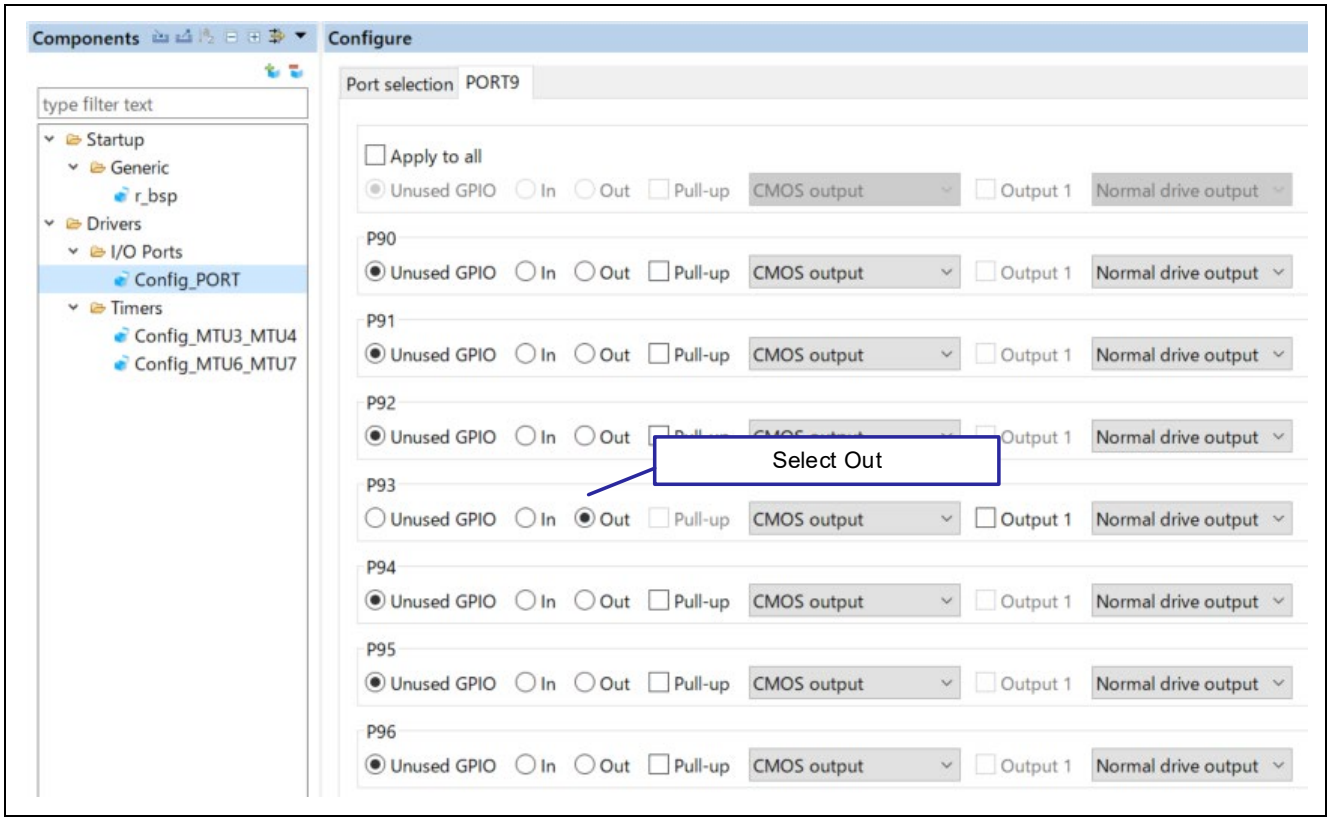


Figure 3.27 Settings for P93 (2/2)

3.4.4 Flowcharts

The following shows the processing of a component added after code generation by the Smart Configurator.

In the main function, count start function mtu_start is read and counting is started.

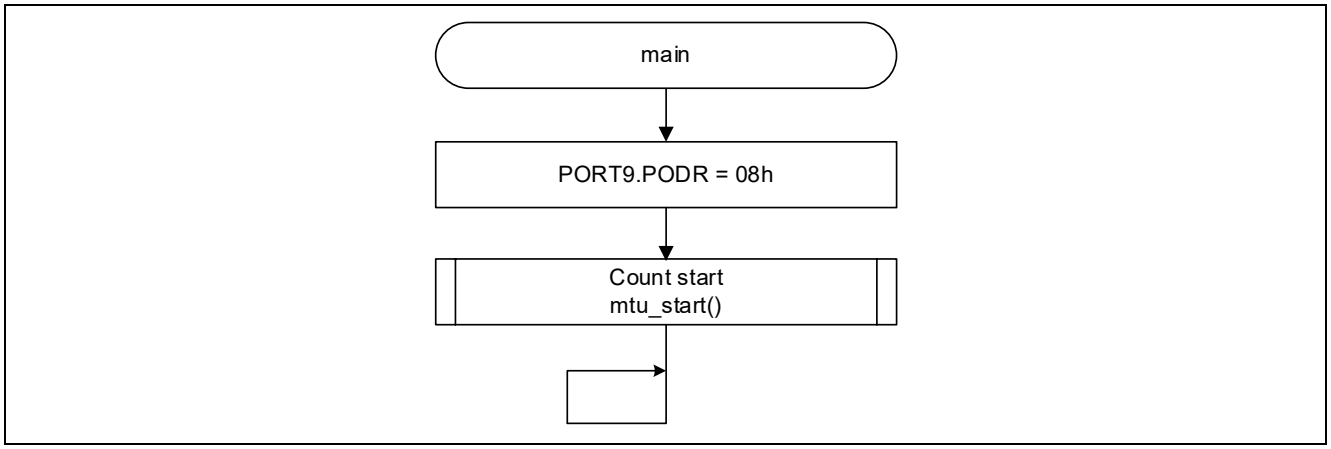


Figure 3.28 main Function

The MTU3, MTU4, MTU6 and MTU7 counting is started in the count start function.

This function is newly created after code generation by the Smart Configurator.

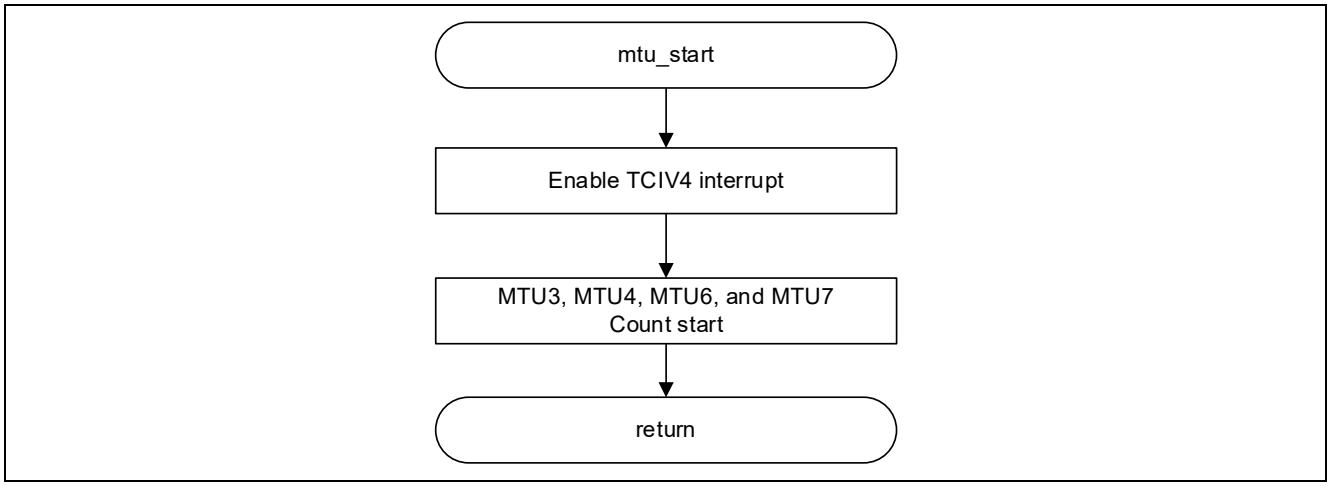


Figure 3.29 Count Start Function

The TCIV4 interrupt handler function changes the value of P93 according to the value of the current P93.

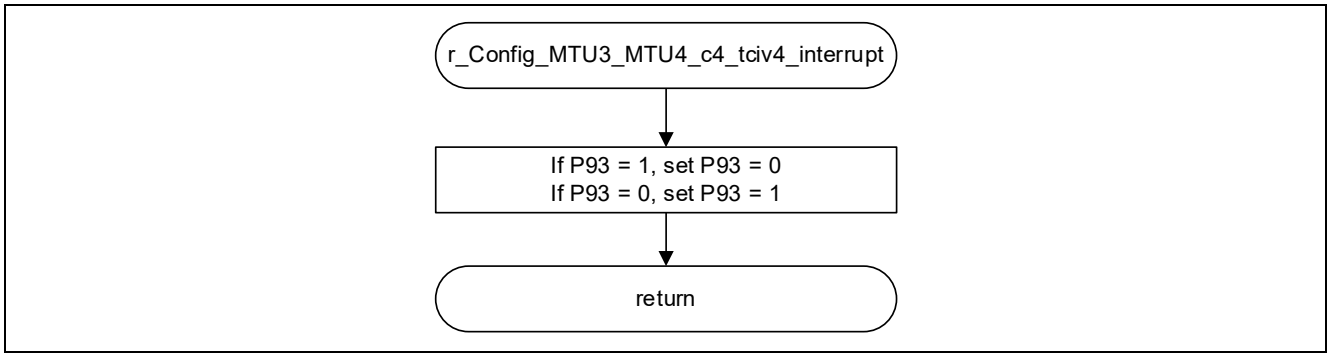


Figure 3.30 TCIV4 Interrupt Handler Function

3.4.5 Related Operations

3.4.5.1 Stopping during Synchronous Operation

The TSTRA and TSTRB registers are used to stop MTU3, MTU4, MTU6, and MTU7. The CST3 and CST4 bits of the TSTRA register are set to 0b to stop MTU3 and MTU4, and the CST6 and CST7 bits of the TSTRB register are set to 0b to stop MTU6 and MTU7.

The stop timing may not be the same between MTU3/MTU4 and MTU6/MTU7.

3.4.6 Usage Notes

3.4.6.1 Counting Starts for Multiple Channels

In this sample code, the SCH3, SCH4, SCH6, and SCH7 bits of timer counter synchronous start register TCSYSTR are set at the same time in the mtu_start function in order to start counting multiple channels at the same time.

When using the R_Config_MTU3_MTU4_Start and R_Config_MTU6_MTU7_Start functions generated by the Smart Configurator, the counting starts timing may not be the same because each of the functions are read.

For details, refer to RX66T Group User's Manual: Hardware, section 22.2.19 Timer Counter Synchronous Start Register (TCSYSTR).

3.4.6.2 Counter Value when Count Operation is Stopped in Complementary PWM Mode

When the counting operation is stopped while operating in complementary PWM mode, MTU3.TCNT (MTU6.TCNT) goes to the value of timer dead time register TDDRA (TDDRB) and MTU4.TCNT (MTU7.TCNT) goes to 0000h.

For details, refer to RX66T Group User's Manual: Hardware, section 22.6.13 Counter Value When Count Operation is Stopped in Complementary PWM Mode.

3.4.6.3 Notes to Prevent Malfunctions in Synchronous Clearing for Complementary PWM Mode

Operational malfunctions may occur when performing synchronous clearing in complementary PWM mode.

For details, refer to RX66T Group User's Manual: Hardware, section 22.6.25 Notes to Prevent Malfunctions in Synchronous Clearing for Complementary PWM Mode.

3.4.6.4 Notes on Buffer Register Updates

When updating the buffer registers in complementary PWM mode, the transfer of the updated buffer values is triggered by writing to MTU4.TGRD on the MTU3 and MTU4 sides, and by writing to MTU7.TGRD on the MTU6 and MTU7 sides. Note that if the write timing to both registers straddle the transfer timing, simultaneous transfer will not occur.

3.5 MTU6/MTU7 Counter Synchronous Clearing by Interrupt

- Target sample code file name: r01an6282_rx66t_mtu3_int_sync.zip

3.5.1 Overview

MTU6 and MTU7 can perform counter clearing using TGImn interrupt generation timing (m = A to D, n = 0 to 2) by setting the TSYCR register.

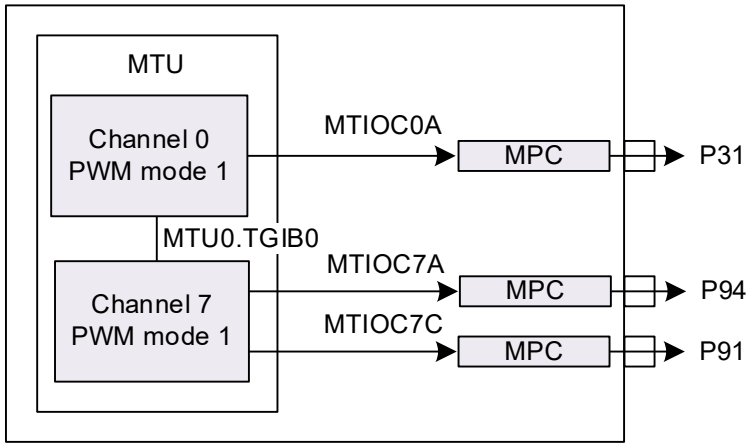
This sample code describes how to perform counter clearing for MTU7 at the MTU0's TGIB0 interrupt generation timing using the timer synchronous clear register (TSYCR). MTU0 and MTU7 use the TCSYSTR register to perform synchronous start by software.

The following list provides the MTU settings used in the sample code.

- MTU0 (channel 0)
  - Use PWM mode 1
  - Initial output value = low
  - Carrier period = 1ms
  - Timer count clock = 40MHz (PCLKC/4)
  - Use MTU0.TGRB as period register
    - Timer counter clear source = MTU0.TGRB compare match
    - Low output at TGRB compare match
  - MTU0.TGRA as duty register
    - High output at TGRA compare match
- MTU7 (channel 7)
  - Use PWM mode 1
  - Initial output value = low
  - Timer count clock = 40MHz (PCLKC/4)
  - No counter clearing
  - Use MTU7.TGRA as duty register
    - Toggle output at TGRA compare match
  - Use MTU7.TGRB as duty register
    - Toggle output at TGRB compare match
  - Use MTU7.TGRC as duty register
    - Toggle output at TGRC compare match
  - Use MTU7.TGRD as duty register
    - Toggle output at TGRD compare match
  - Enable clearing by MTU0.TGIB0 interrupt generation timing

Set in Smart Configurator.
For Setting Methods,
refer to section 3.5.3.

The structure of this sample code is shown below.



MTU0 and MTU7: Generate PWM mode 1 waveform
MPC: Sets pins to be used from general purpose I/O port to peripheral function I/O port

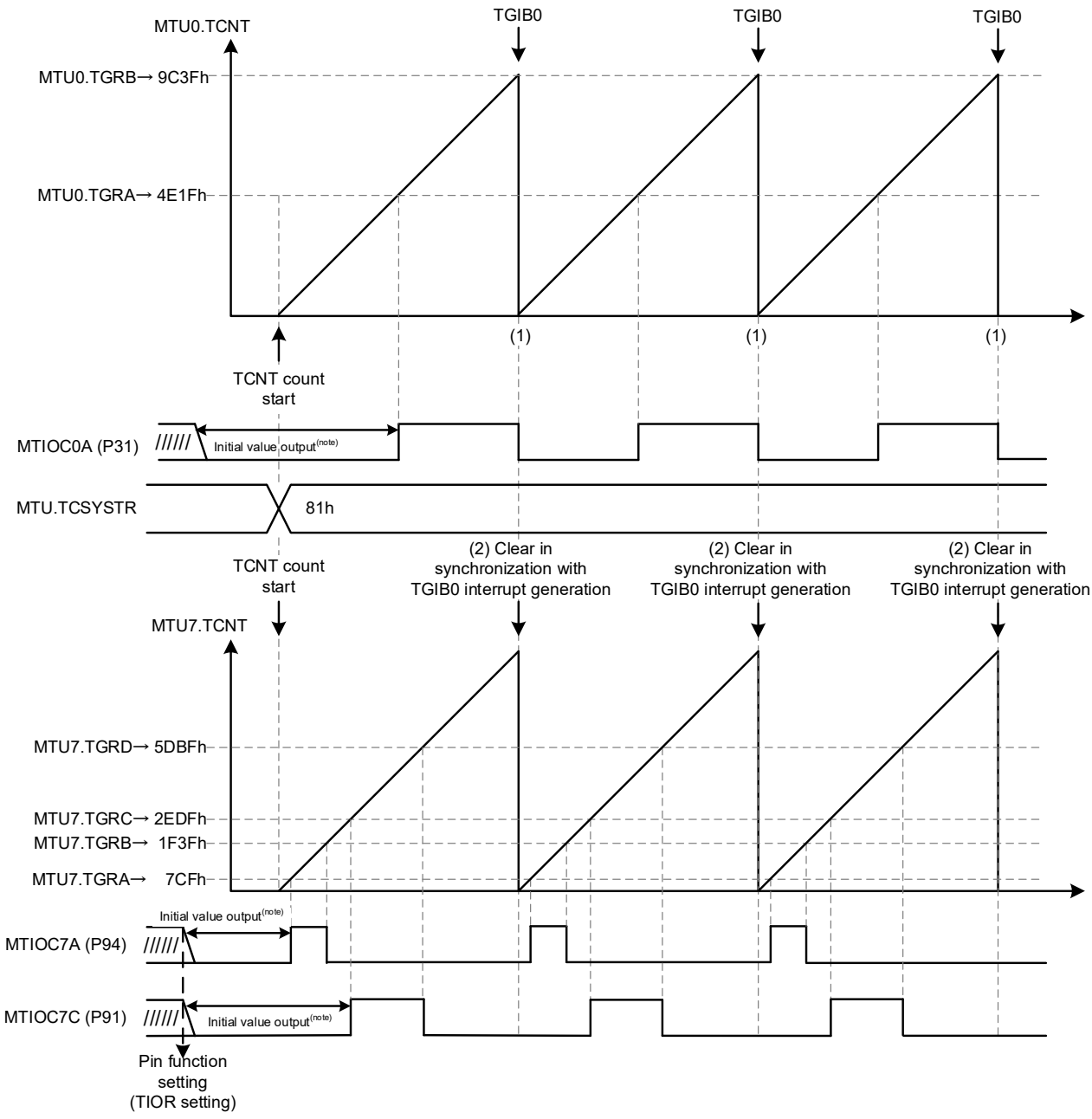
Figure 3.31 Sample Code Structure

3.5.2 Operation Details

The sample code operations are shown below. Synchronous operation is started by setting MTU0 and MTU7 to PWM mode 1 and setting 81h to the MTU.TCSYSTR register.

MTU0.TGRB is used as a period register, and MTU0.TCNT is count cleared by the compare match of TGRB ((1) in the figure below). MTU7.TCNT is count cleared at the timing the compare match interrupt (TGIB0) of MTU0.TGRB is generated ((2) in the figure below). Note that (1) and (2) do not occur simultaneously. For details, refer to 3.5.6.2.

MTIOC0A outputs high at an MTU0.TGRA compare match and low at an MTU0.TGRB compare match. MTIOC7A toggles output each time an MTU7.TGRA compare match or MTU7.TGRB compare match occurs. MTIOC7C toggles output each time an MTU7.TGRC compare match or MTU7.TGRD compare match occurs.



Note: The sample code waveform starts outputting the initial values when the PMR register is set.

Figure 3.32 Sample Code Operations

3.5.3 Smart Configurator Settings

The sample code uses the Smart Configurator to add the MTU as described below. For details on how to add components, refer to section 3.1.4 Adding Components.

Table 3.12 Adding Components (MTU0)

Item	Description	
Component	PWM Mode Timer	
Configuration name	Config_MTU0	Config_MTU7
Operation	PWM Mode 1	
Resource	MTU0	MTU7

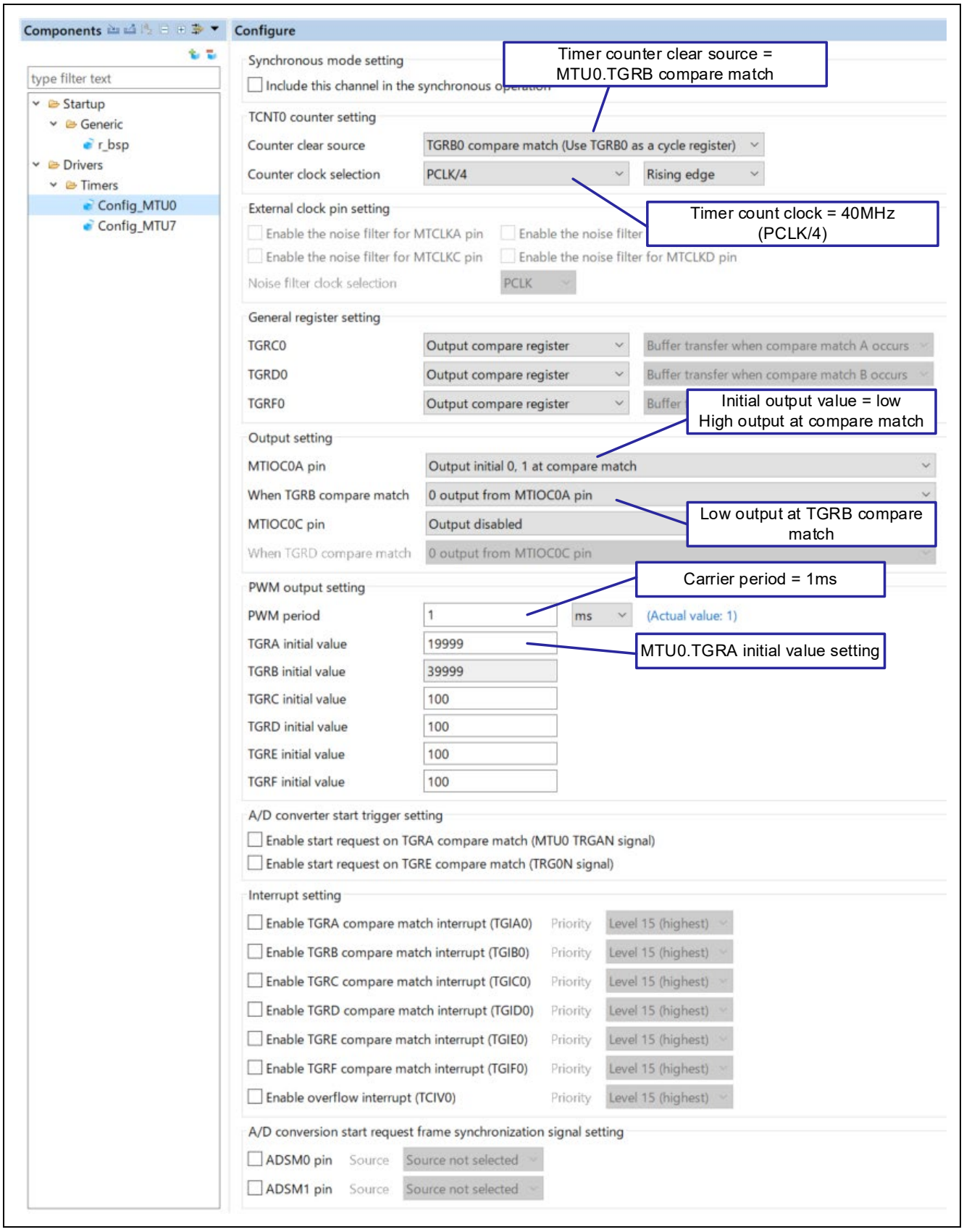


Figure 3.33 MTU0 Settings

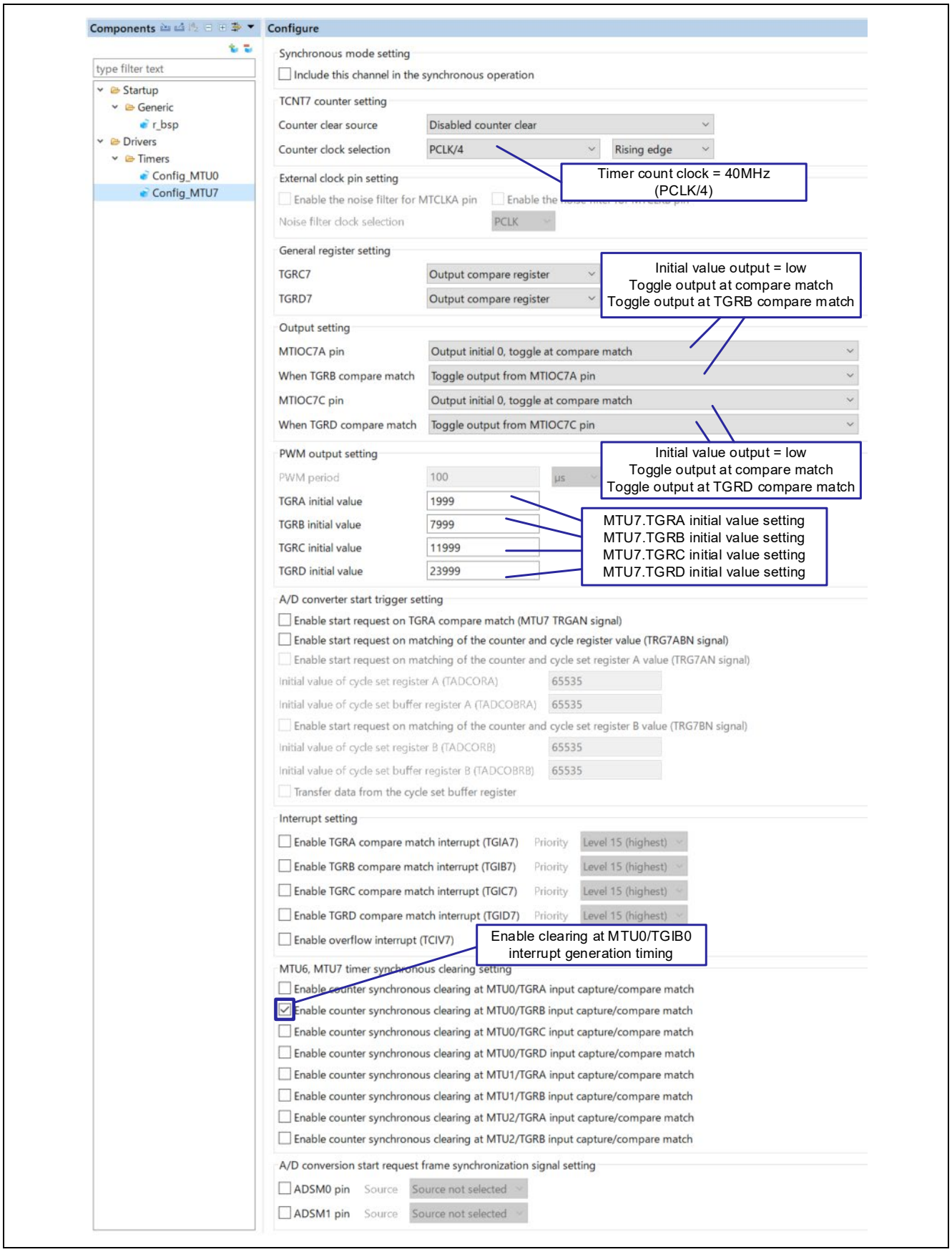


Figure 3.34 MTU7 Settings

3.5.4 Flowcharts

The following shows the processing of a component added after code generation by the Smart Configurator.

In the main function, count start function mtu_start is read and counting is started.

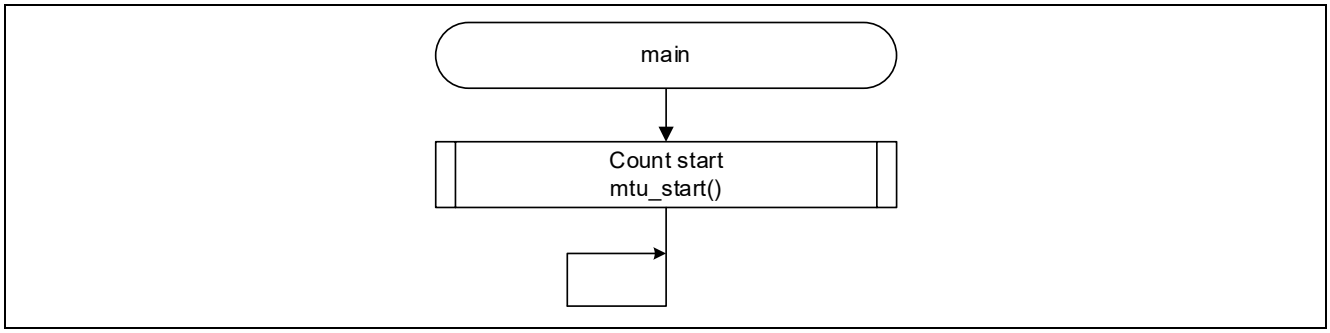


Figure 3.35 main Function

The MTU0 and MTU7 counting is started in the count start function.

This function is newly created after code generation by the Smart Configurator.

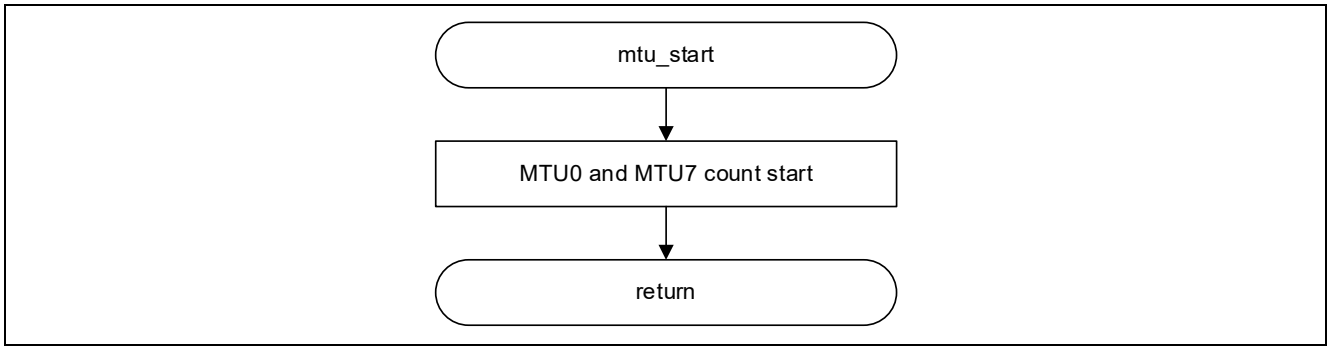


Figure 3.36 Count Start Function

3.5.5 Related Operations

3.5.5.1 Using Multiple Synchronous Clearing

This sample code is used to describe an operation in which multiple synchronous clearing is performed.

The MTU7's Smart Configurator settings should be changed as follows.

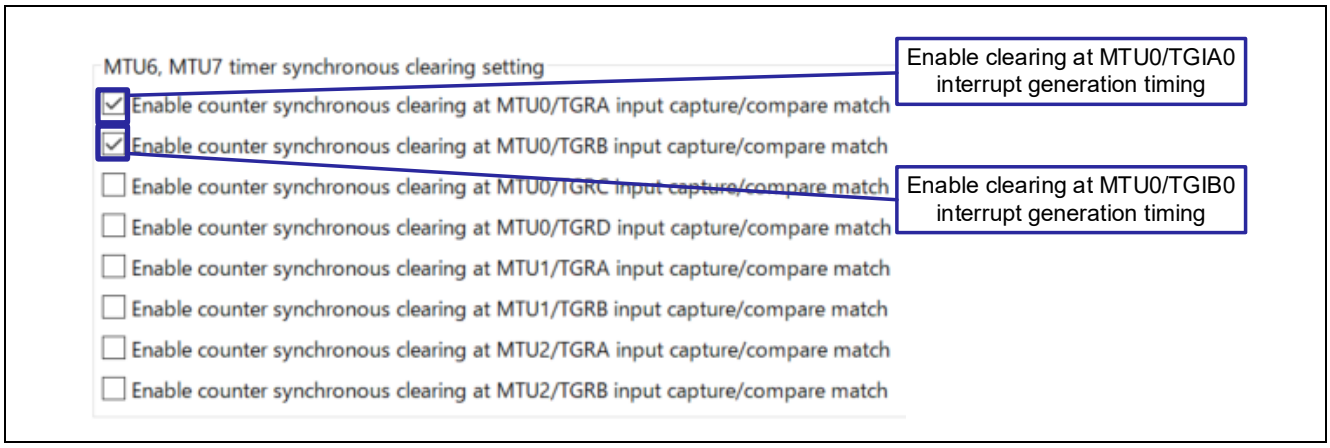


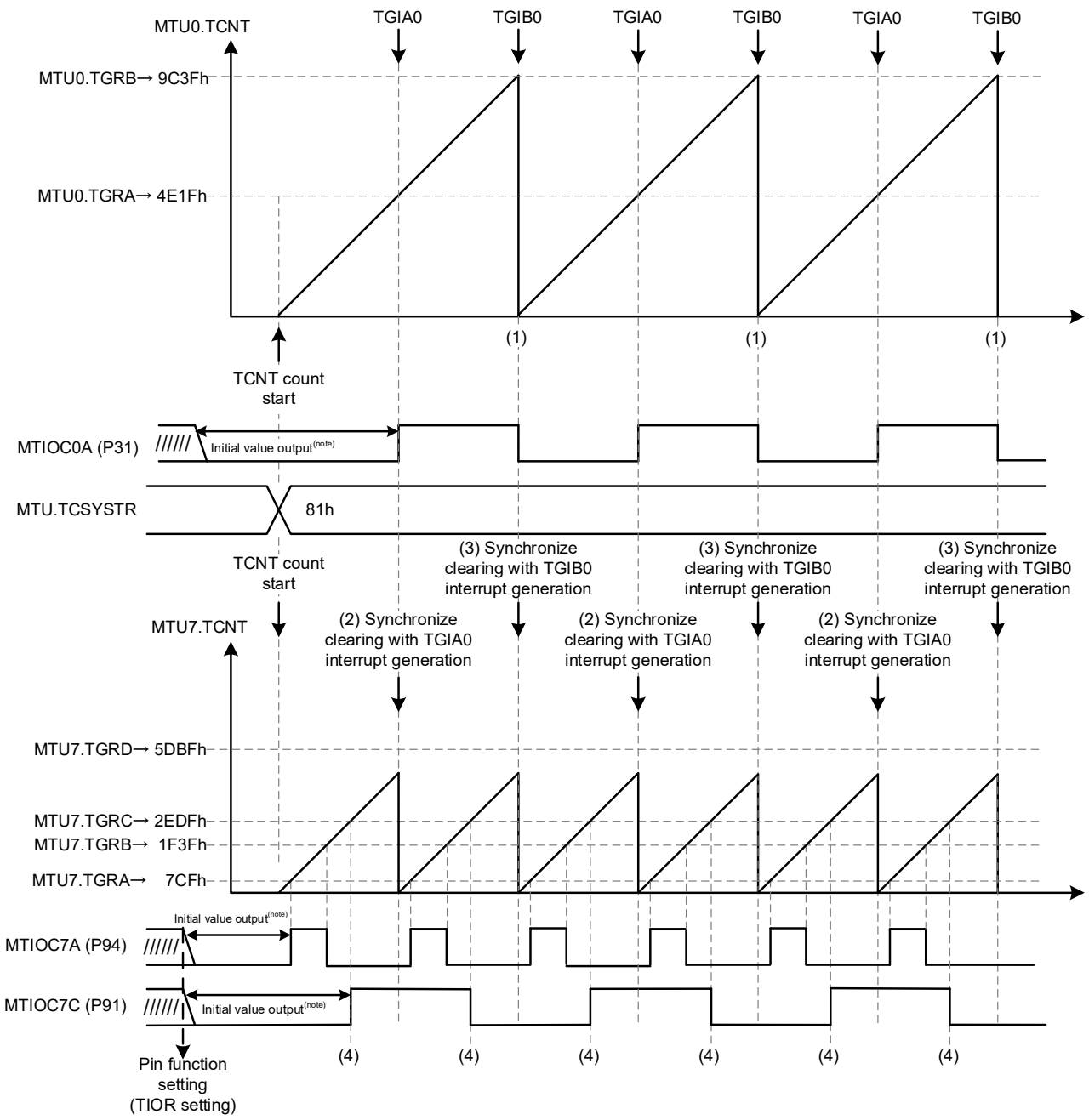
Figure 3.37 MTU7 Settings

Figure 3.38 shows operations after the settings have been changed.

MTU0.TGRB is used as the period register and MTU0.TCNT is counter cleared at a TGRB compare match ((1) in Figure 3.38). MTU7.TCNT is counter cleared at the MTU0.TGRA compare match interrupt (TGIA0) generation timing ((2) in Figure 3.38) and the MTU0.TGRB compare match interrupt (TGIB0) generation timing ((3) in Figure 3.38). Note that the timing of the MTU0.TGRA compare match and (2), and the timing of (1) and (3), are not simultaneous.

MTIOC0A outputs high at an MTU0.TGRA compare match and low at an MTU0.TGRB compare match. MTIOC7A toggles output each time an MTU7.TGRA compare match or an MTU7.TGRB compare occurs.

In the same manner as MTIOC7A, MTIOC7C toggles output each time an MTU7.TGRC compare match occurs ((4) in Figure 3.38). Because MTU7.TCNT is cleared before the MTU7.TGRD compare match occurs, the MTU7.TGRD compare match does not occur and output does not change.



Note: The sample code waveform starts outputting the initial values when the PMR register is set.

Figure 3.38 Operations After Setting Changes

3.5.6 Usage Notes

3.5.6.1 Counting Starts for Multiple Channels

In this sample code, the SCH0 and SCH7 bits of timer counter synchronous start register TCSYSTR are set at the same time in the mtu_start function to start counting multiple channels at the same time.

When using the R_Config_MTU0_Start and R_Config_MTU7_Start functions generated by the Smart Configurator, the counting start timings may not be the same because each of the functions are read.

For details, refer to RX66T Group User's Manual: Hardware, section 22.2.19 Timer Counter Synchronous Start Register (TCSYSTR).

3.5.6.2 Interrupt Signal Timing

The timing of the compare match signal and the timing of compare match interrupt (TGInm (m = A to D, n = 0 to 7, 9)) are not simultaneous.

For details, refer to RX66T Group User's Manual: Hardware, section 22.5.2 Interrupt Signal Timing.

3.5.6.3 Regarding MTU6

Although this sample code confirms the counter clear operation for MTU7, the same counter clear operation can be performed for MTU6. However, because MTIOC6A (PA1) and MTIOC6C (PA0) are connected to the CAN transceiver on the board used in this application note (Renesas Starter Kit for RX66T), the counter clear operation on MTU6 will not output the correct results. To confirm operations in MTU6, either refrain from using MTIOC6A (PA1) and MTIOC6C (PA0) or disconnect the CAN transceiver.

3.6 Synchronous Operation by Event Input from ELC

- Target sample code file name: r01an6282_rx66t_mtu3_elc_sync.zip

3.6.1 Overview

The MTU can use the ELC (event link controller) to perform synchronous operation (start and stop).

This sample code describes how to perform counting start for MTU0 and MTU1 by selecting the GTCCRA compare match of GPTW0 as the event source of ELC. MTU1 is synchronously cleared by counter clear source of MTU0. GPTW1 starts simultaneously with GPTW0, and outputs PWM in the same cycle as MTU0.

The following list provides the GPTW, MTU, and ELC settings used in the sample code.

- GPTW0 and GPTW1 (channels 0 and 1)
  - Use Sawtooth-wave PWM mode
  - Initial output value = low
  - Carrier period = 100µs
  - Timer counter clock = 160MHz (PCLKC)
  - Use GTPR as period register
    - Count direction = up-counting
    - Counter initial value = 0
  - Use GPTW0.GTCCRA as duty register
    - Use GTIOC0A pin as PWM output pin
    - Toggle output at GTCCRA compare match
    - Retain output at cycle end
  - Use GPTW1.GTCCRA as duty register
    - Use GTIOC1A pin as PWM output pin
    - Toggle output at GTCCRA compare match
    - Toggle output at cycle end
  - Software source count start enabled
- MTU0 (channel 0)
  - Use PWM mode 1
  - Initial output value = low
  - Set to synchronous operation
  - Carrier period = 100µs
  - Timer counter clock = 160MHz (PCLKC)
  - Use MTU0.TGRB as period register
    - Timer counter clear source = MTU0.TGRB compare match
    - Toggle output at TGRB compare match
  - Use MTU0.TGRA as duty register
    - Toggle output at TGRA compare match
- MTU3 (channel 3)
  - Use PWM mode 1
  - Initial output value = low
  - Set to synchronous operation
  - Counter clear source = counter clear of channel 0 in synchronous operation
  - Timer counter clock = 160MHz (PCLKC)
  - Use MTU3.TGRA as duty register
    - Toggle output at TGRA compare match
  - Use MTU3.TGRB as duty register
    - Toggle output at TGRB compare match
- ELC
  - Select GPT0 compare match A as event
  - Select MTU0 as destination resource and count start as operation
  - Select MTU3 as destination resource and count start as operation

Set in Smart Configurator.
For Setting Methods,
refer to section 3.6.3.

The structure of this sample code is shown below.

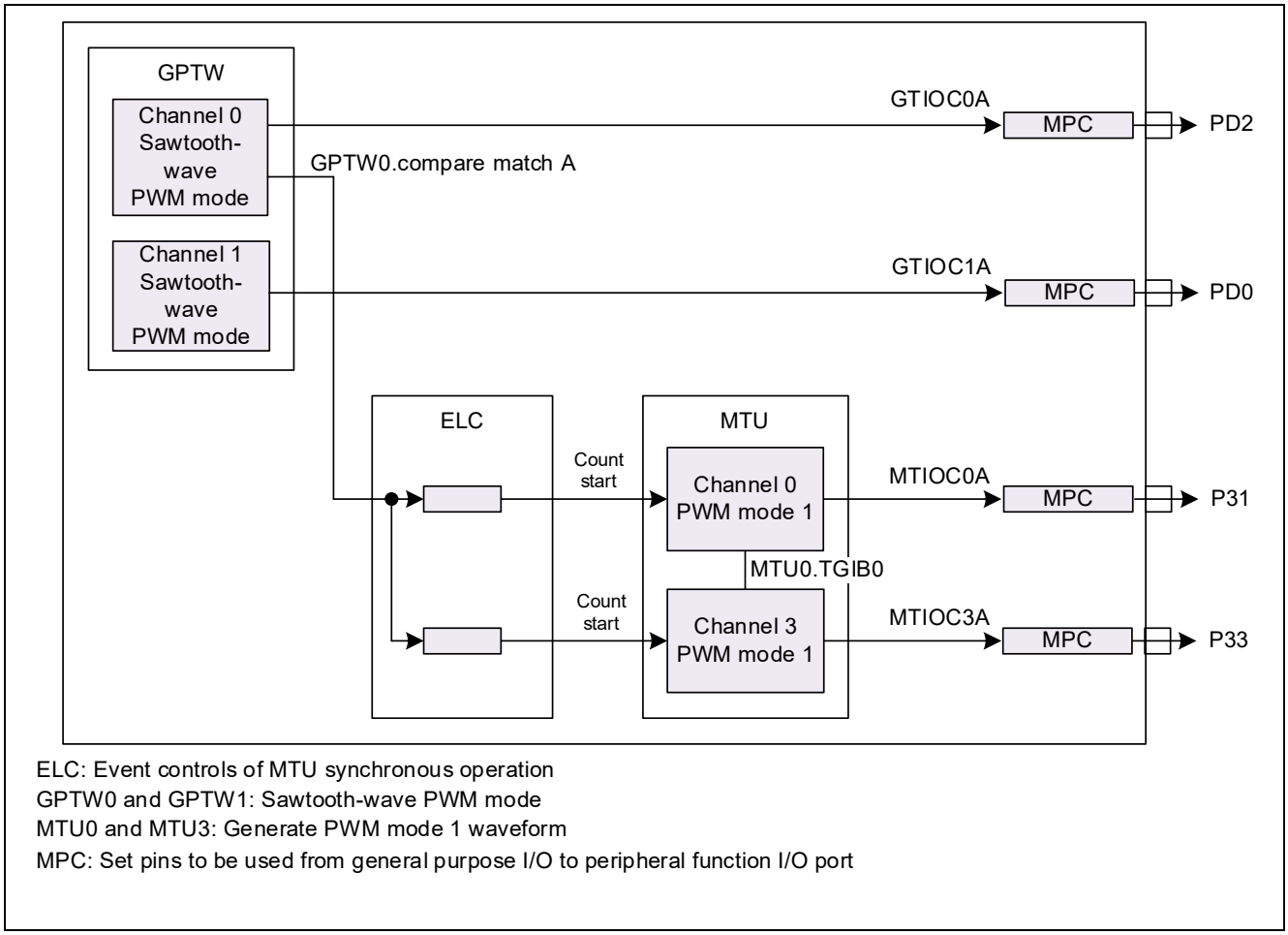


Figure 3.39 Sample Code Structure

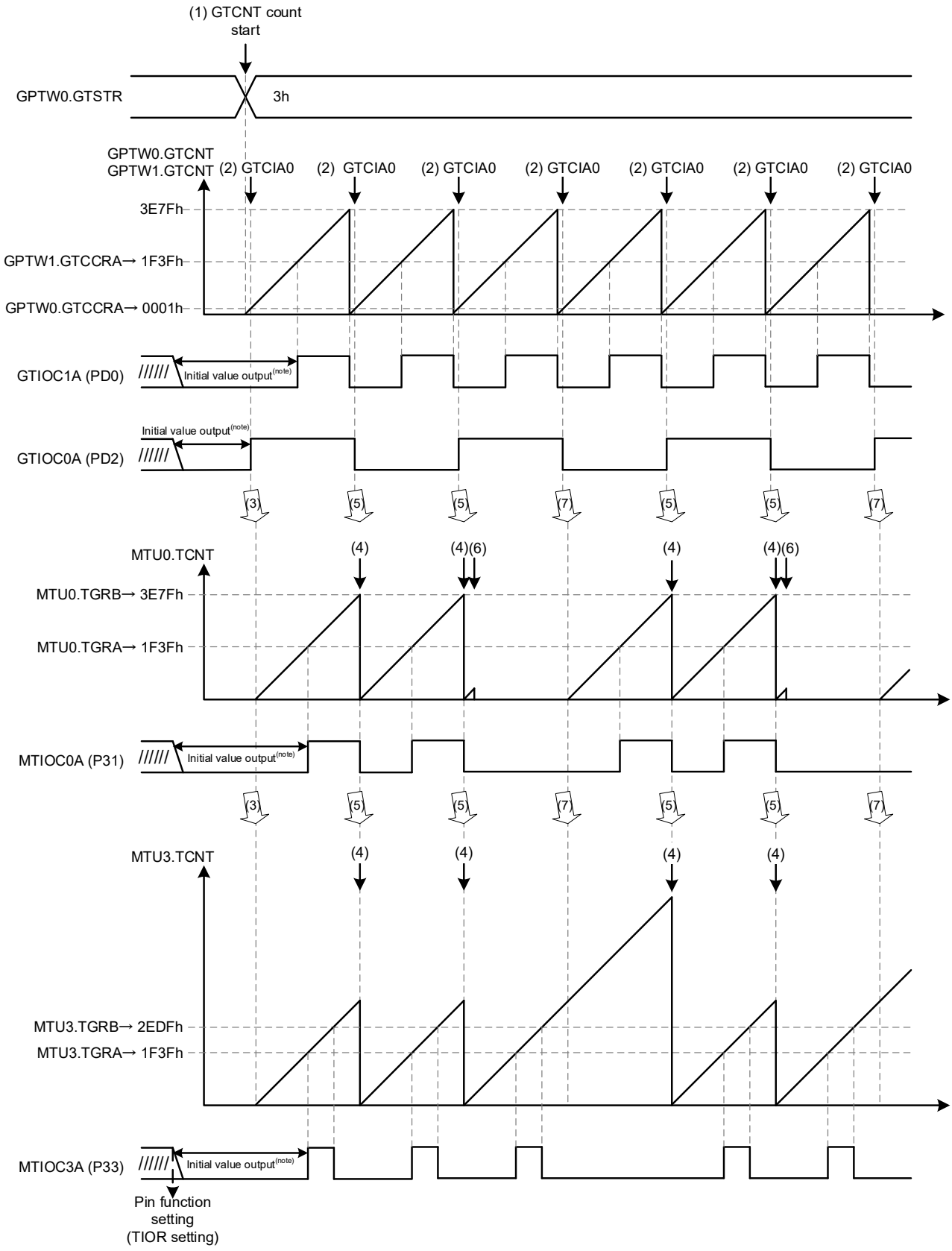
3.6.2 Operation Details

The sample code operations are shown in Figure 3.40. GPTW0 and GPTW1 are set to sawtooth-wave PWM mode and MTU0 and MTU3 are set to PWM mode 1. GPTW0 compare match A is set to the event source of ELC, and MTU0 and MTU3 are set to counting start when the event is generated.

The operation is executed in the following order, steps 1 to 8.

1. The GPTW0.GTSTR register is set to 0003h and GPTW0 and GPTW1 are started synchronously. ((1) in Figure 3.40)
2. GPTW0.GTCCRA compare match occurs and GTCIA0 is generated ((2) in Figure 3.40).
3. Using GPTW0 compare match A (GTCIA0) as the event source, the TSTRA.CST0 and CST3 bits of the MTU go to 1b and the MTU0 and MTU3 TCNT start counting ((3) in Figure 3.40)
4. MTU0.TCNT is cleared at an MTU0.TGRB compare match, and MTU3.TCNT, which was set for synchronous clearing at an MTU0.TGRB compare match, is also cleared ((4) in Figure 3.40).
5. GPTW0 compare match A is generated repeatedly but is disabled when the TSTRA.CST0 and the CST3 bit of the MTU are 1b ((5) in Figure 3.40).
6. Every third generation of GTCIA0 sets 0b in the TSTRA.CST0 bit of the MTU and stops the MTU0.TCNT count ((6) in Figure 3.40). MTU3.TCNT continues to count until the next MTU0.TGRB compare match occurs.
7. When GPTW0 compare match A occurs while the MTU0.TCNT counting is stopped, TSTRA.CST0 bit of the MTU goes to 1b and the MTU0.TCNT starts counting. The TSTRA.CST3 remains at 1b and the GTCIA0 generation is disabled ((7) in Figure 3.40).
8. Steps 4 to 7 are repeated.

Note that in synchronous operation using ELC, the operation timing may not be simultaneous for the event generation module (GPTW0 in the sample code) and the modules that receive generated events and perform interlinked operations (MTU0 and MTU3 in the sample code).



Note: The sample code waveform starts outputting the initial values when the PMR register is set.

Figure 3.40 Sample Code Operations

3.6.3 Smart Configurator Settings

The sample code uses the Smart Configurator to add the GPTW, MTU, and ELC as described below. For details on how to add components, refer to section 3.1.4 Adding Components.

Table 3.13 Adding Components (GPTW0 and GPTW1)

Item	Description	
Component	General PWM Timer	
Configuration name	Config_GPT0	Config_GPT1
Work mode	Sawtooth-Wave PWM Mode	
Resource	GPT0	GPT1

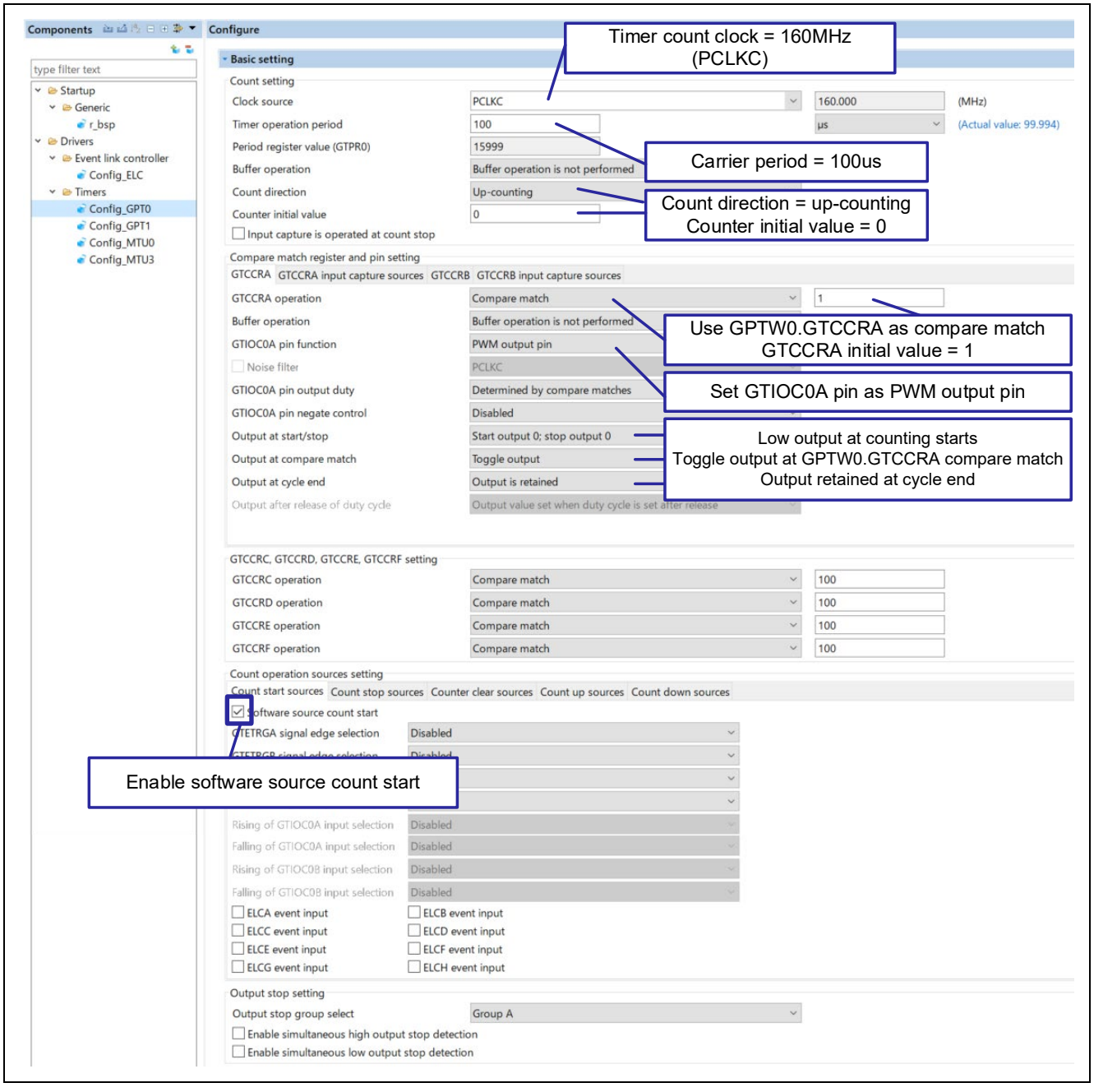


Figure 3.41 GPT0 Settings (1/2)

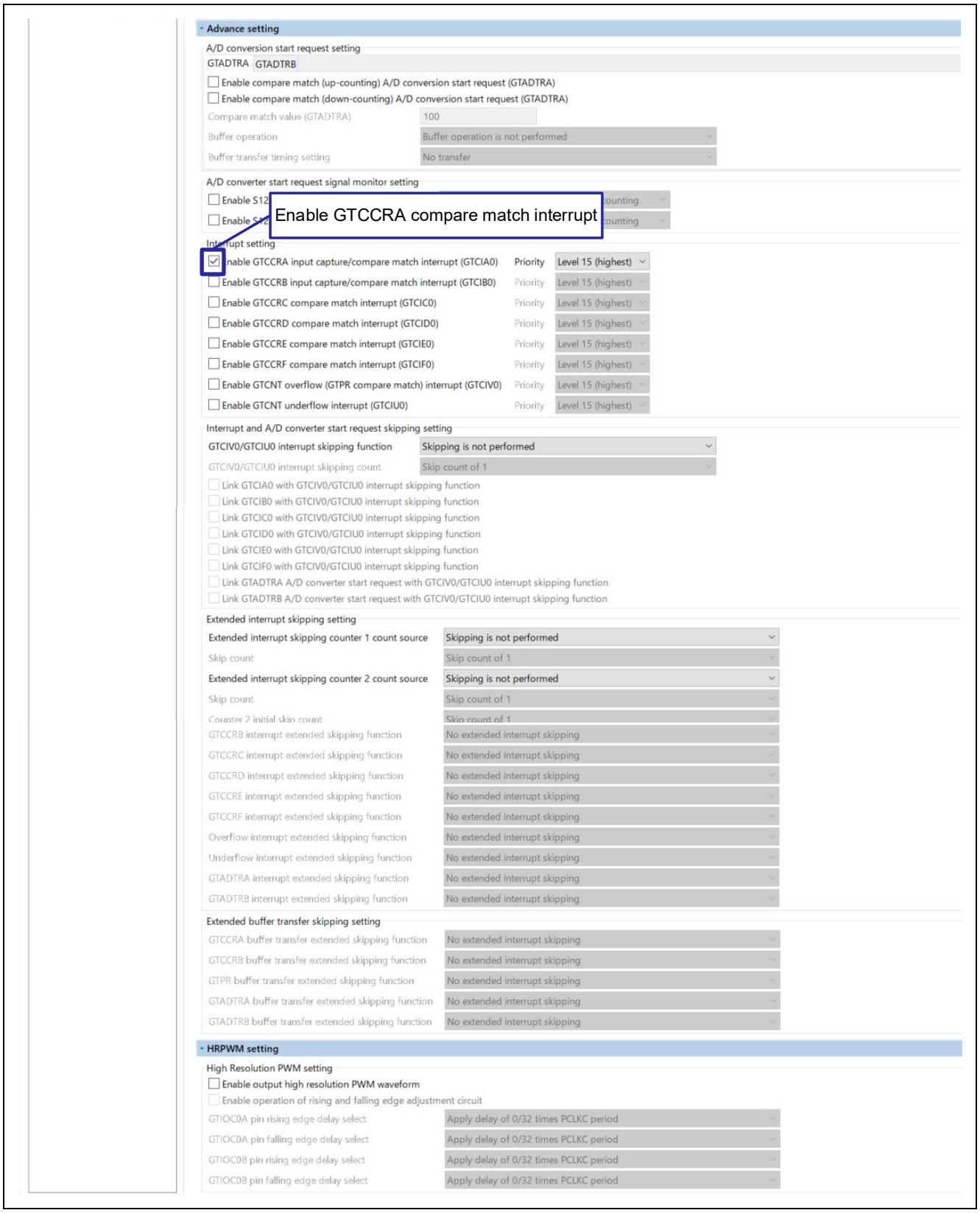


Figure 3.42 GPT0 Settings (2/2)

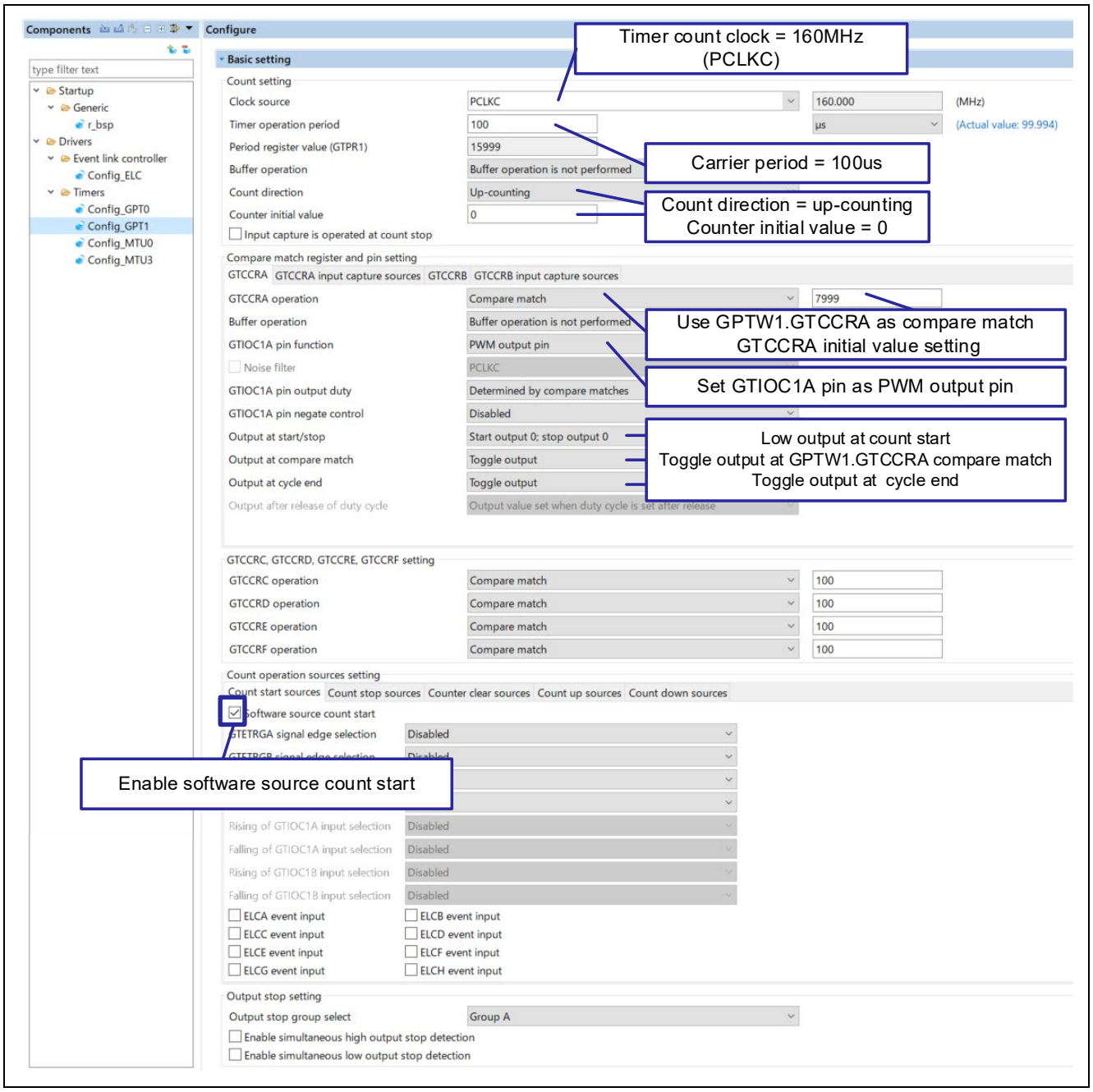


Figure 3.43 GPT1 Settings

Table 3.14 Adding Components (MTU0 and MTU3)

Item	Description	
Component	PWM Mode Timer	
Configuration name	Config_MTU0	Config_MTU3
Operation	PWM Mode 1	
Resource	MTU0	MTU3

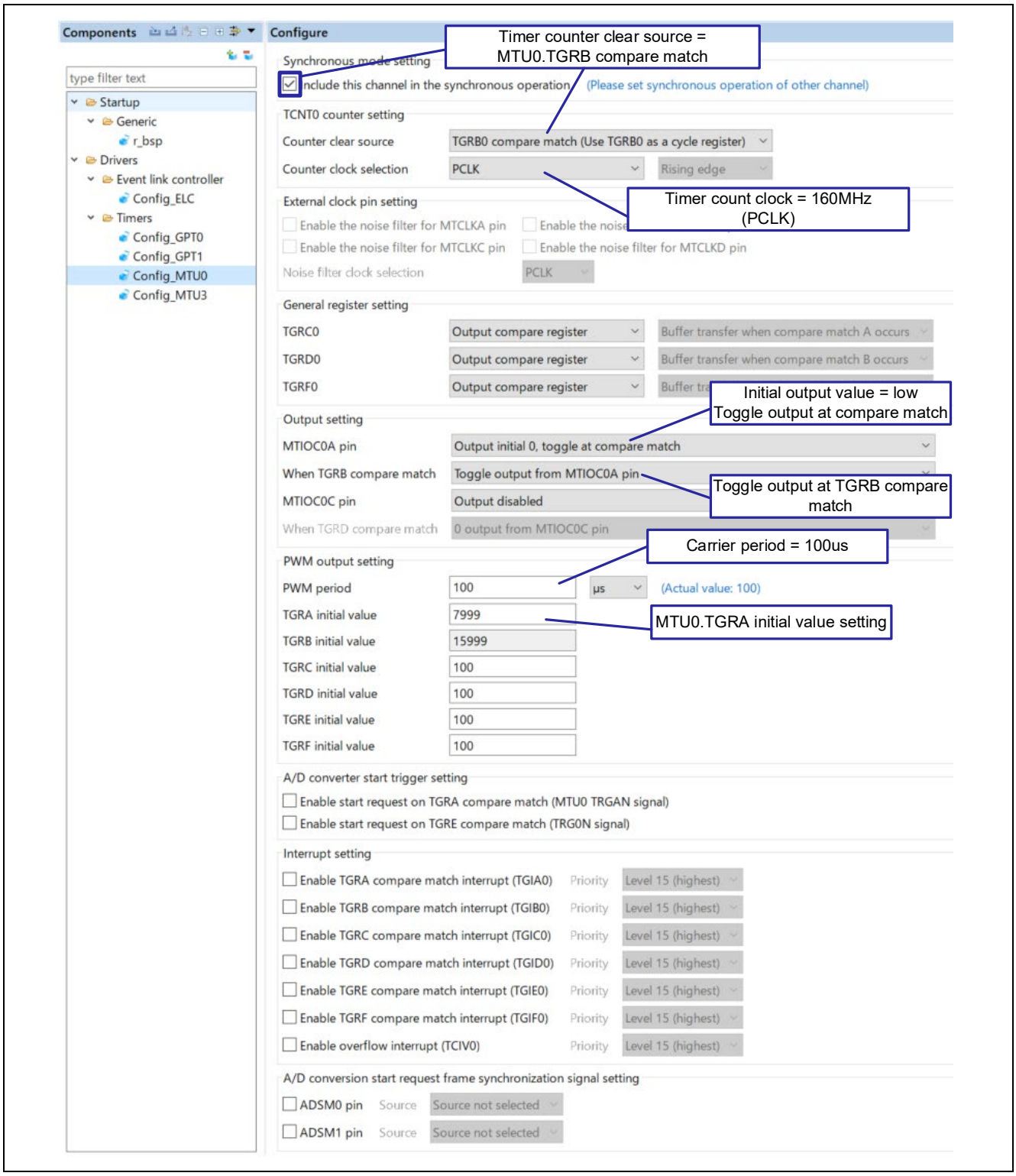


Figure 3.44 MTU0 Settings

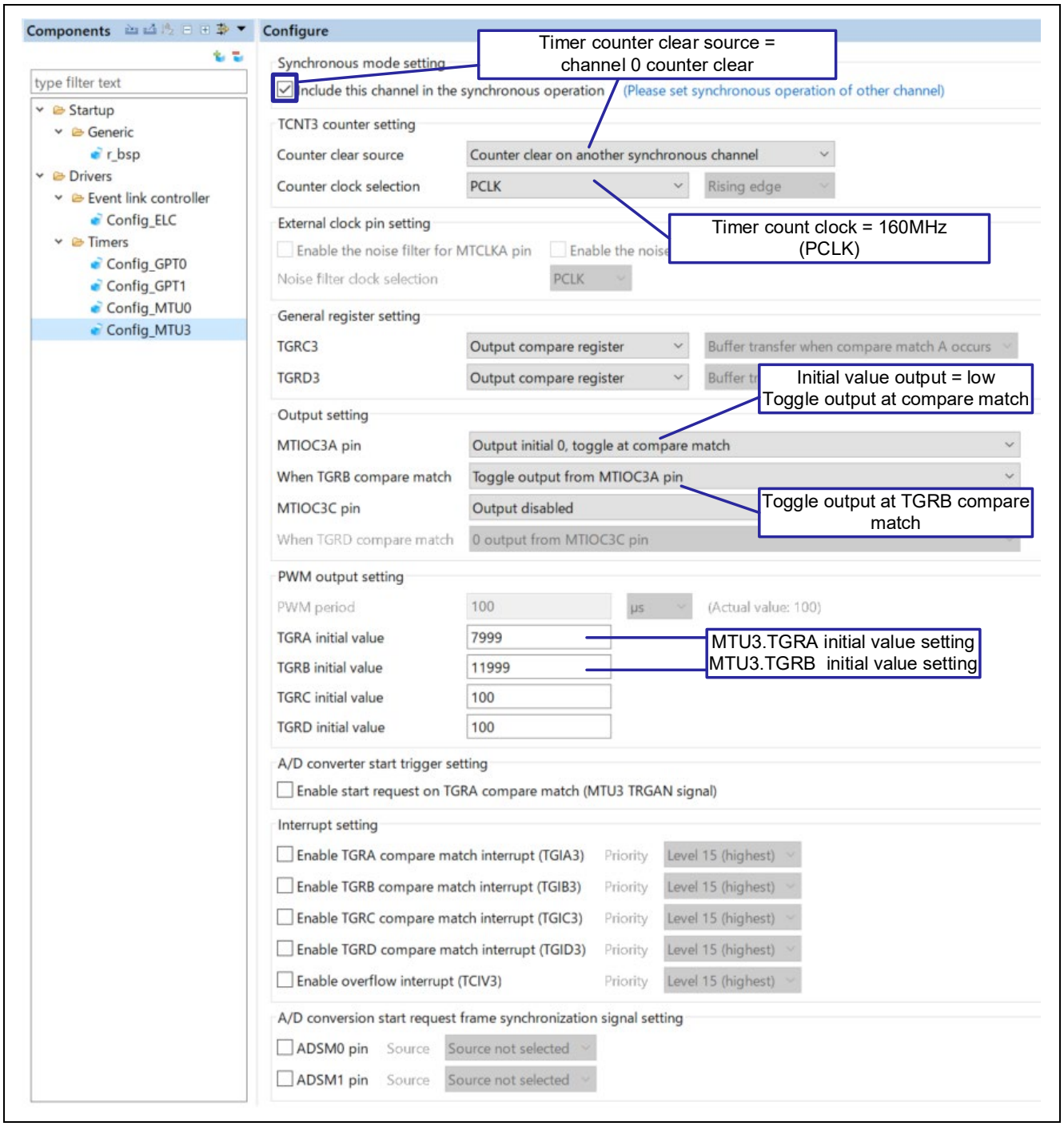


Figure 3.45 MTU3 Settings

Table 3.15 Adding Components (ELC)

Item	Description
Component	Event Link Controller
Configuration name	Config_ELC
Resource	ELC

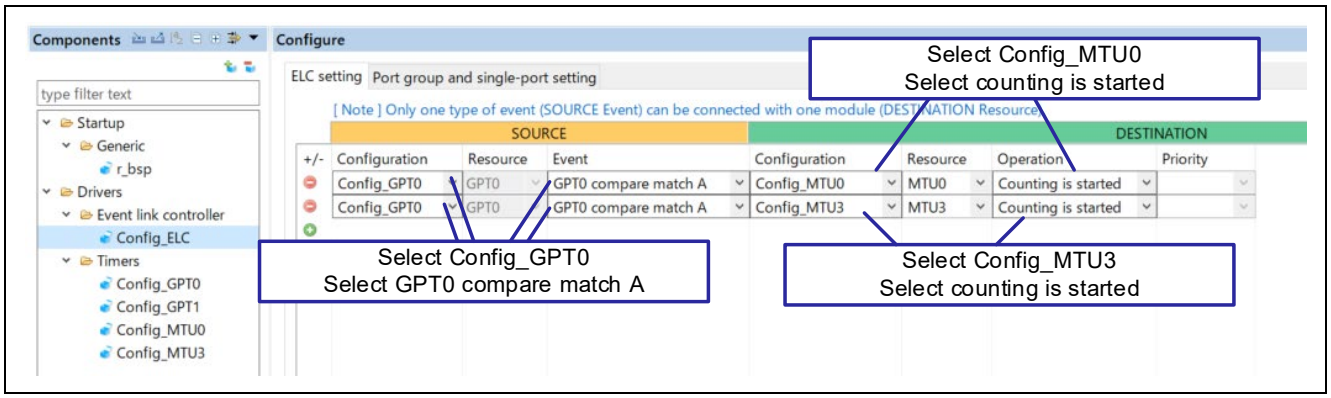


Figure 3.46 ELC Settings

3.6.4 Flowcharts

The following shows the processing of a component added after code generation by the Smart Configurator.

In the main function, the ELC is enabled, the count start function gpt_start is read, and counting is started. If the value of the interrupt generation count flag (g_int_cnt) is equal to STOP_INTERVAL (set to 3 in the sample code), MTU0 counting is stopped, MTU0.TCNT is cleared, and the interrupt generation count flag is updated to 0.

This sample code uses the following variables.

- g_int_cnt: variable for retaining interrupt generation count

This sample code uses the following constant.

- STOP_INTERVAL: setting value that determines the interval at which MTU0 is stopped

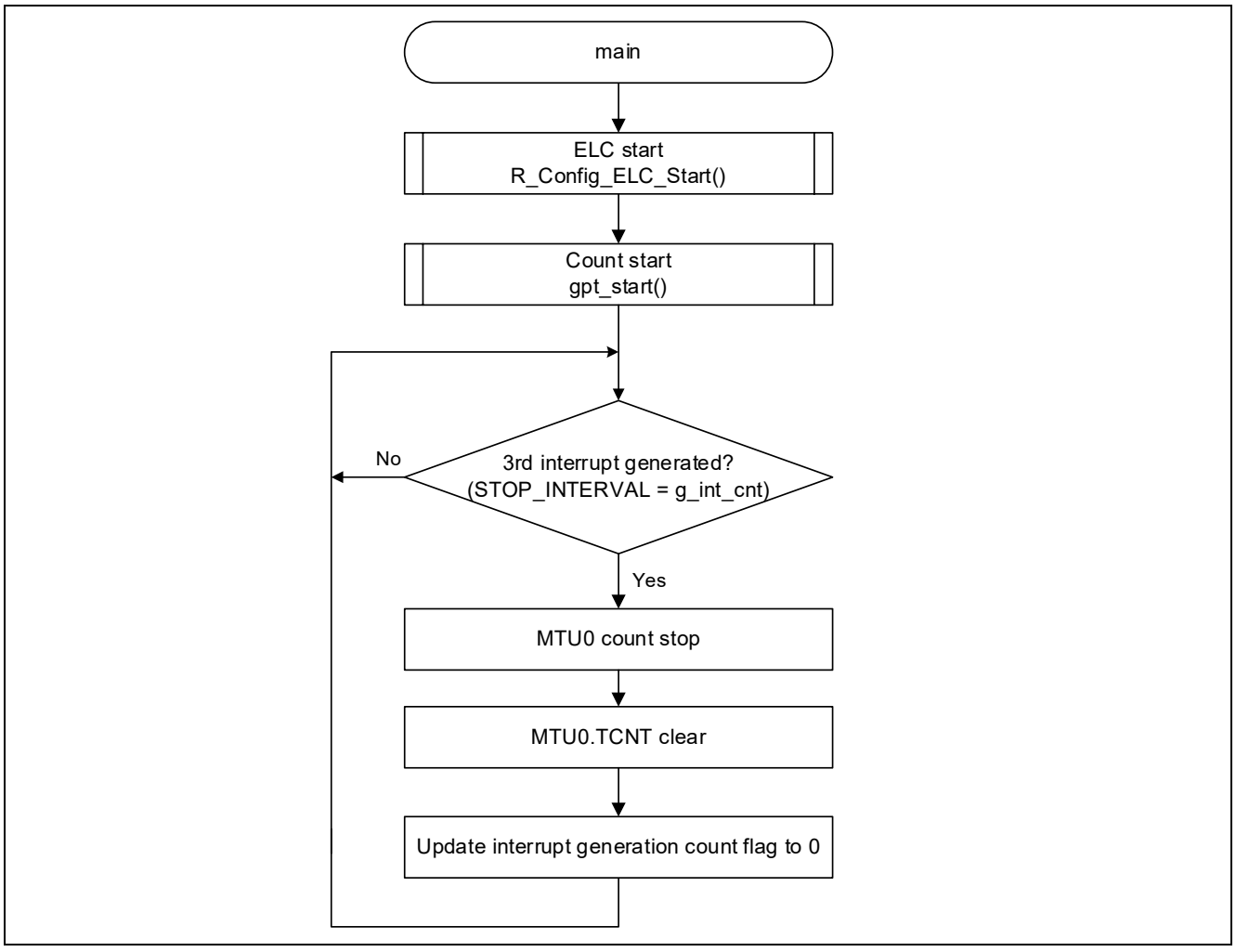


Figure 3.47 main Function

In the count start function, the GTCIA0 interrupt is enabled and the GPT0 and GPT1 counting is started.

This function is newly created after code generation by the Smart Configurator.

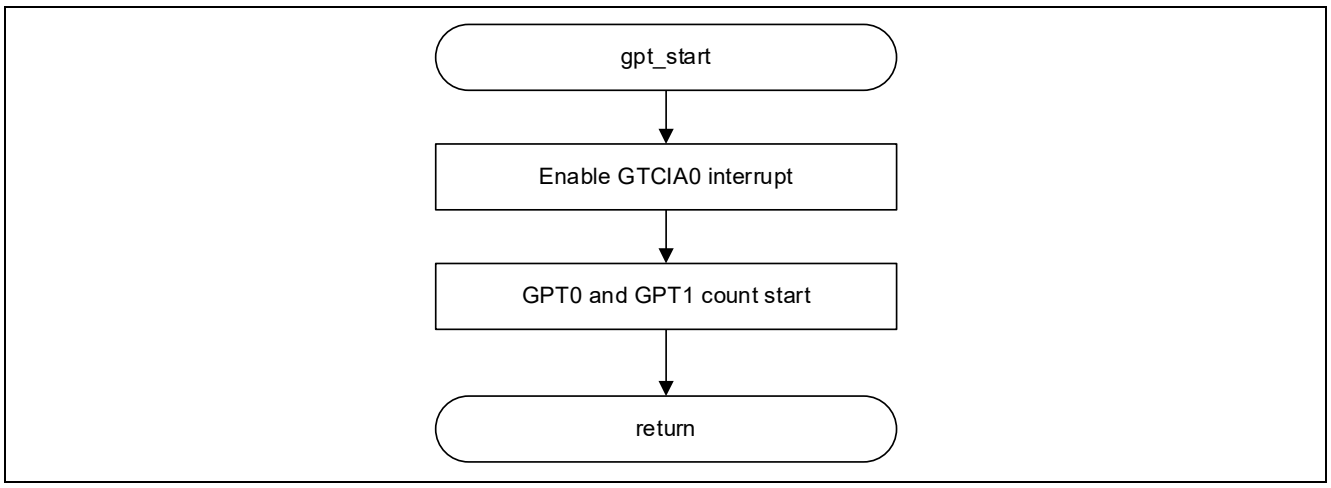


Figure 3.48 Count Start Function

In the GTCIA0 interrupt handler function, the interrupt generation count flag is increased by 1.

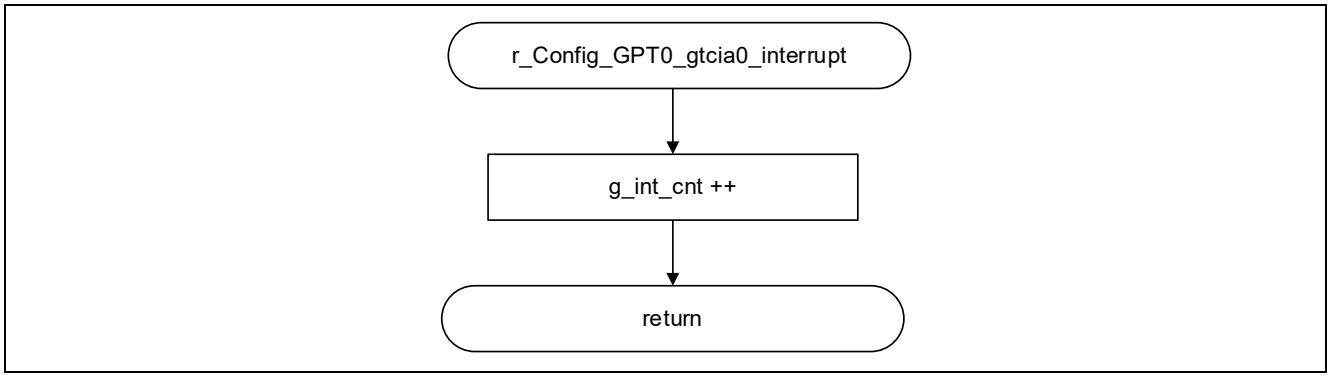


Figure 3.49 GTCIA0 Interrupt Handler Function

3.6.5 Usage Notes

3.6.5.1 Counting Starts for Multiple Channels

In this sample code, the CSTRT0 and CSTRT1 bits of timer software start register GTSTR are set at the same time in the gpt_start function in order to start counting GPTW0 and GPTW1 at the same time.

When using the R_Config_GPTm_Start (m = 0, 1) function generated by the Smart Configurator, the counting starts timing may not be the same because each of the functions are read.

For details, refer to RX66T Group User's Manual: Hardware, section 24.3.8.1 Synchronous Operation by Software.

3.6.5.2 Operations in Response to Event Signal from ELC

This sample code explains how to perform a count start operation when an event signal is received from the ELC. The MTU can also perform a count restart (counter clearing) operation and input capture operation as interlinked operations.

For details, refer to RX66T Group User's Manual: Hardware, section 22.8.2 MTU Operations in Response to Receiving Event Signals from the ELC.

3.6.5.3 Notes on Timer Mode Register Settings for ELC Event Input

When setting the MTU as the destination resource for the ELC, set the timer mode register (TMDR) for the corresponding channel to the initial value (00h).

For details, refer to RX66T Group User's Manual: Hardware, section 22.6.26 Notes on Timer Mode Register Setting for ELC Event Input.

3.6.5.4 Event Signal Output to ELC

Although this sample code describes the operation of receiving an event signal from the ELC, it is possible to output an interrupt request signal from the MTU to the ELC as an event signal.

For details, refer to RX66T Group User's Manual: Hardware, sections 19. Event Link Controller (ELC) and 22.8.1 Event Signal Output to the ELC.

3.6.5.5 Usage Notes on MTU Operation by Event Signal Reception from the ELC

Precautions must be taken when the MTU is used in count start operation or count restart (count clearing) operation by event link.

For details, refer to RX66T Group User's Manual: Hardware, section 22.8.3 Usage Notes on MTU Operation by Event Signal Reception from the ELC.

4. GPTW Sample Codes

4.1 Common

4.1.1 Sample Code List

This application note provides the following sample codes created with the Smart Configurator.

Sample codes can be downloaded from the Renesas Electronics website.

Table 4.1 GPTW Sample Code List

Name	Sample Code Usage Conditions	Ref.
Synchronous Operation by Software in Sawtooth-Wave PWM Mode r01an6282_rx66t_gptw_sawtooth_pwm_sync.zip	• Sawtooth-wave PWM mode • Software (GTSTR register) synchronous start • Software (GTSTP and GTCLR register) synchronous stop/clearing	4.2
Synchronous Operation (Phase Shift) by Software in Sawtooth-Wave PWM Mode r01an6282_rx66t_gptw_sawtooth_pwm_sync_shift.zip	• Sawtooth-wave PWM mode • Software (GTSTR register) synchronous start • Software (GTSTP and GTCLR register) synchronous stop/clearing • Phase shift by GTCNT counter value	4.3
Synchronous Operation (Phase Shift) by Software in Triangle-Wave PWM Mode r01an6282_rx66t_gptw_triangle_sync_shift.zip	• Triangle-wave PWM mode 1 • Software (GTSTR register) synchronous start • Phase shift by GTCNT counter value	4.4
Synchronous Operation by Event Input from ELC r01an6282_rx66t_gptw_sawtooth_1st_elc_sync.zip	• Sawtooth-wave PWM mode • Hardware (ELC) synchronous start • Hardware (ELC) synchronous stop/clearing	4.5
Synchronous Operation by External Trigger Input r01an6282_rx66t_gptw_sawtooth_1st_trigger_sync.zip	• Sawtooth-wave PWM mode • Hardware (external trigger) synchronous start • Hardware (external trigger) synchronous stop/clearing	4.6

4.1.2 Folder Structure

The main folder structure of a sample code is as follows.

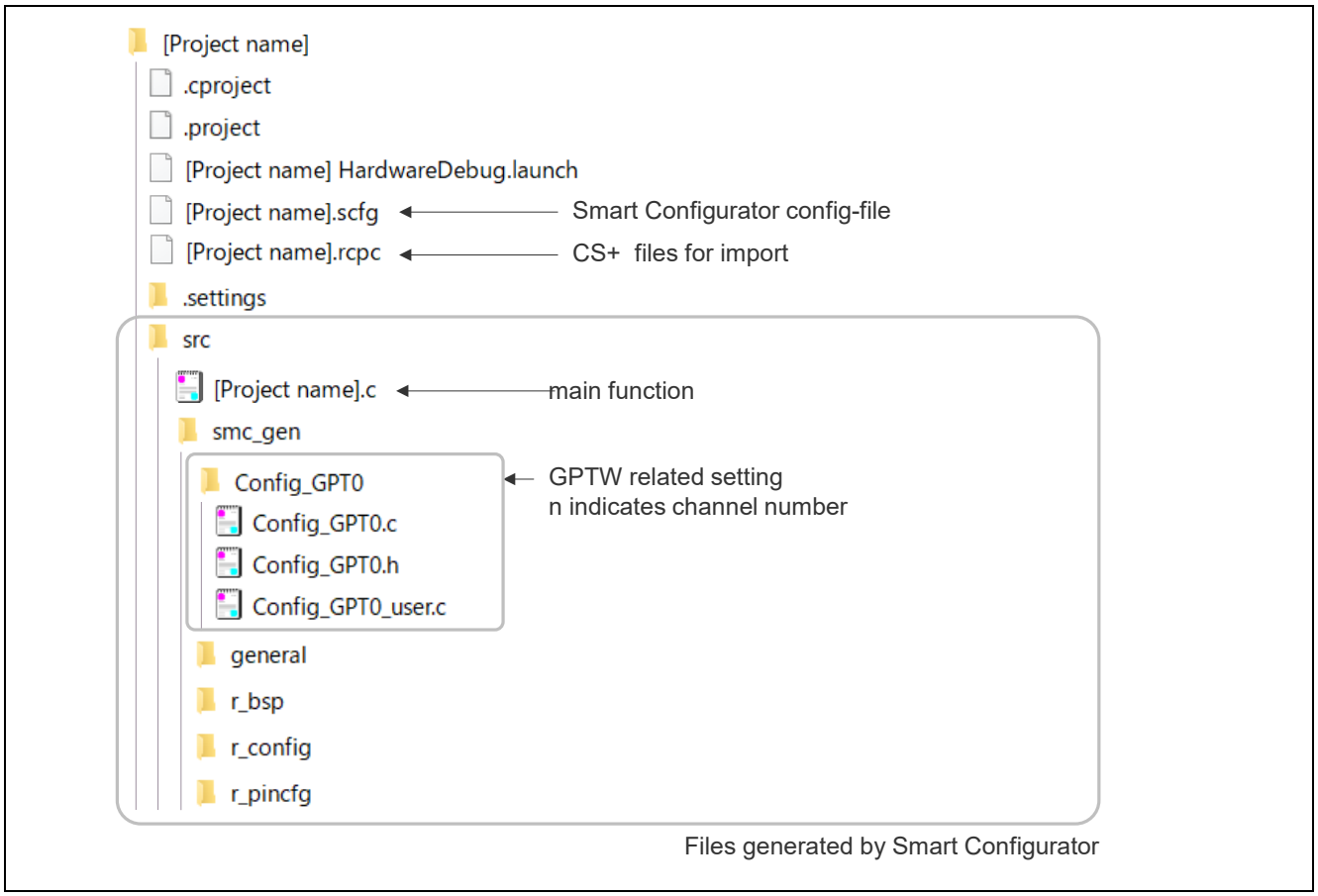


Figure 4.1 GPTW Folder Structure

4.1.3 File Structure

The main file structure of a sample code is as follows.

Table 4.2 GPTW File Structure

File Name	Description
[Project name].c	main Function This is the main function. The Smart Configurator generates an empty function. The necessary processing for each sample code is described here.
Config_GPTn.c*	R_Config_GPTn_Create Function This is the GPTW's initialization function. The initialization function based on the settings in the Smart Configurator is generated by the Smart Configurator. The call for this function is generated by the Smart Configurator. This function is called in the R_SystemInit function executed before the main function.
	R_Config_GPTn_Start Function This is the GPTW's count start function. This function is generated by the Smart Configurator. In the sample codes, this function is called from the main function
	R_Config_GPTn_Stop Function This is the GPTW's count stop function. This function is generated by the Smart Configurator. This function is not used in the sample codes.
Config_GPTn_user.c*	r_Config_GPTn_Create_UserInit Function This is the GPTW's user initialization function. The Smart Configurator generates an empty function. The necessary processing for each sample code is described here. This is the last function to be called in the R_Config_GPTn_Create function generated by the Smart Configurator.
	r_Config_GPTn_[interrupt name]_interrupt Function This is the interrupt handler function. The Smart Configurator generates an empty function. The necessary processing for each sample code is described here.
Config_GPTn.h*	This is the header file that defines GPTW related functions. This file is included in the r_smc_entry.h file generated by the Smart Configurator. To use GPTW related functions, be sure to include the r_smc_entry.h file.

*: n indicates channel number

4.1.4 Adding Components

The sample codes use the Smart Configurator to add the GPTW as described below.

Table 4.3 Adding Components

Item	Description
Component	General PWM Timer ((1) in figure below)
Configuration name	Sample codes use the default setting name
Work mode	Reference the section for each sample code ((2) in figure below)
Resource	Reference the section for each sample code ((3) in figure below)

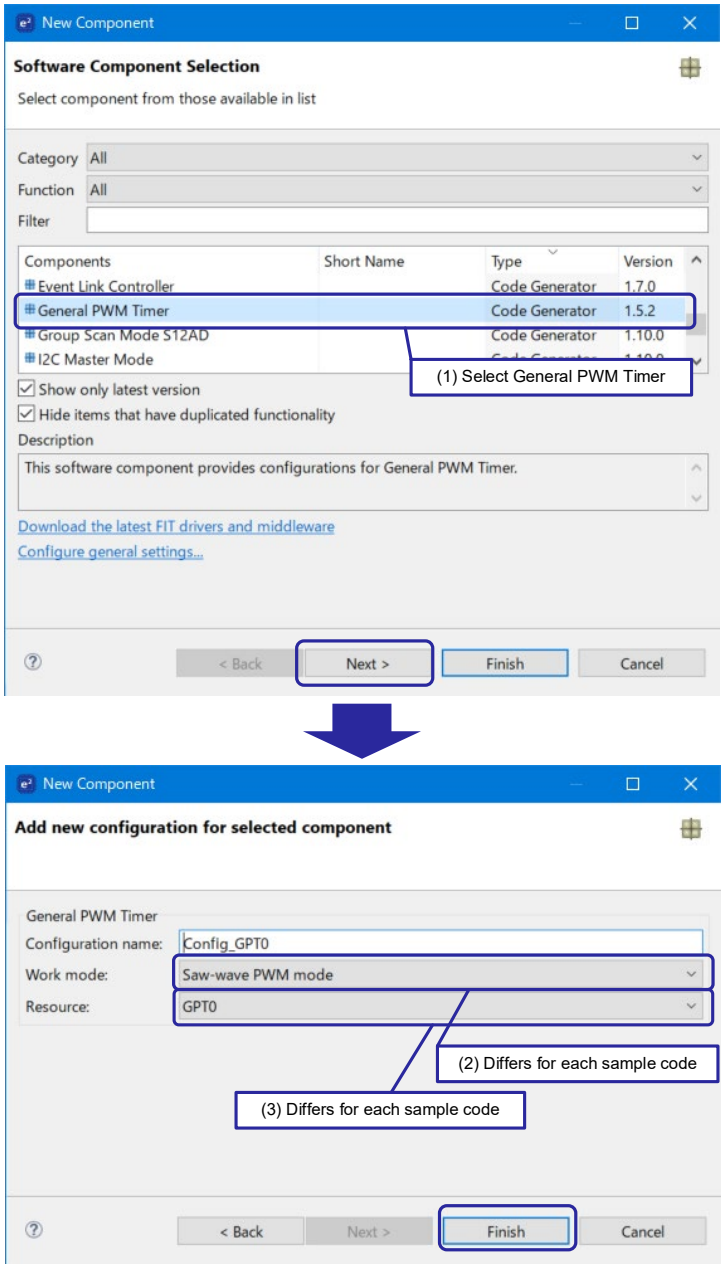


Figure 4.2 Adding Components

4.1.5 Pin Settings

Figure 4.3 shows an example of pin settings using the Smart Configurator.

Configure the pins after setting the GPTW. For GPTW settings, refer to “Smart Configurator Settings” for each sample code.

Pin settings are carried out in the R_Config_GPTn_Create function generated by the Smart Configurator.

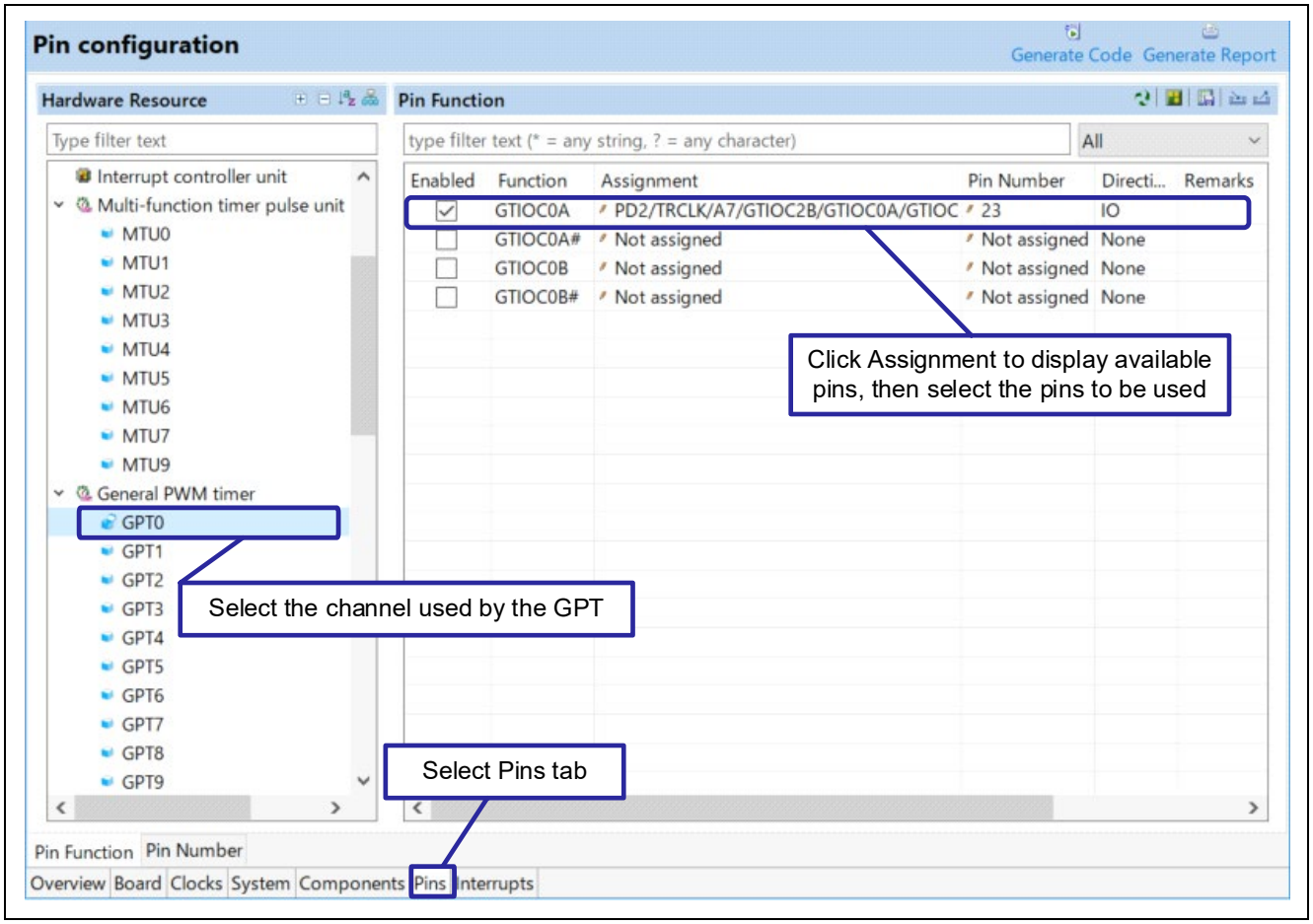


Figure 4.3 Pin Settings

4.1.6 Interrupt Settings

Figure 4.4 shows an example of interrupt settings using the Smart Configurator. For details on Software Configurable Interrupt A, refer to Renesas RX66T Group User's Manual Hardware, section 14.4.5.1 Software Configurable Interrupt A.

Configure interrupts after setting the GPTW settings. For GPTW settings, refer to "Smart Configurator Settings" for each sample code.

Interrupt settings can be configured in the R_Config_GPTn_Create function, R_Config_GPTn_Start function, and R_Config_GPTn_Stop function, all of which are generated by the Smart Configurator.

The interrupt hander function is created with the name r_Config_GPTn_[interrupt name]_interrupt in the Config_GPTn_user.c file generated by the Smart Configurator.

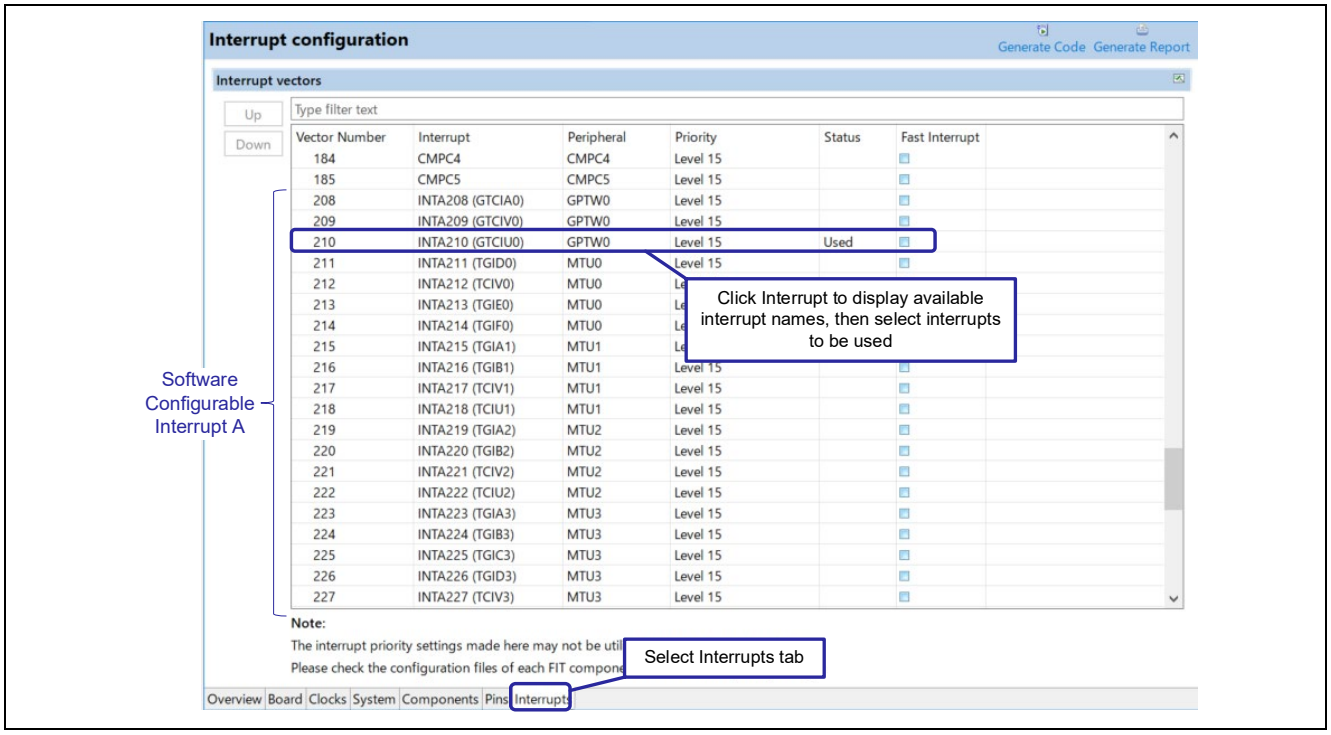


Figure 4.4 Interrupt Settings

Only GTCIE0, GTCIF0 and GDTE0 are selected for GPTW interrupts by default in the Interrupts tab of the Smart Configurator. To use interrupts configured in the Components tab, the interrupts must be selected in the Interrupts tab. The following shows the status and error message when a selection is missing.

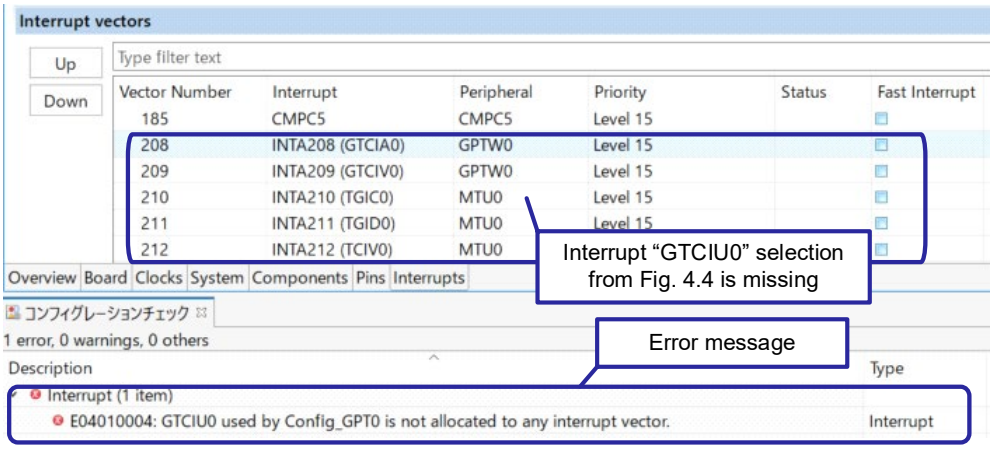


Figure 4.5 Interrupt Settings (Interrupt Selection Missing)

4.2 Synchronous Operation by Software in Sawtooth-Wave PWM Mode

- Target sample code file name: r01an6282_rx66t_gptw_sawtooth_pwm_sync.zip

4.2.1 Overview

In the GPTW sawtooth-wave PWM mode, synchronous start can be performed using the GTSTR register, synchronous stop using the GTSTP register and synchronous clearing using the GTCLR register.

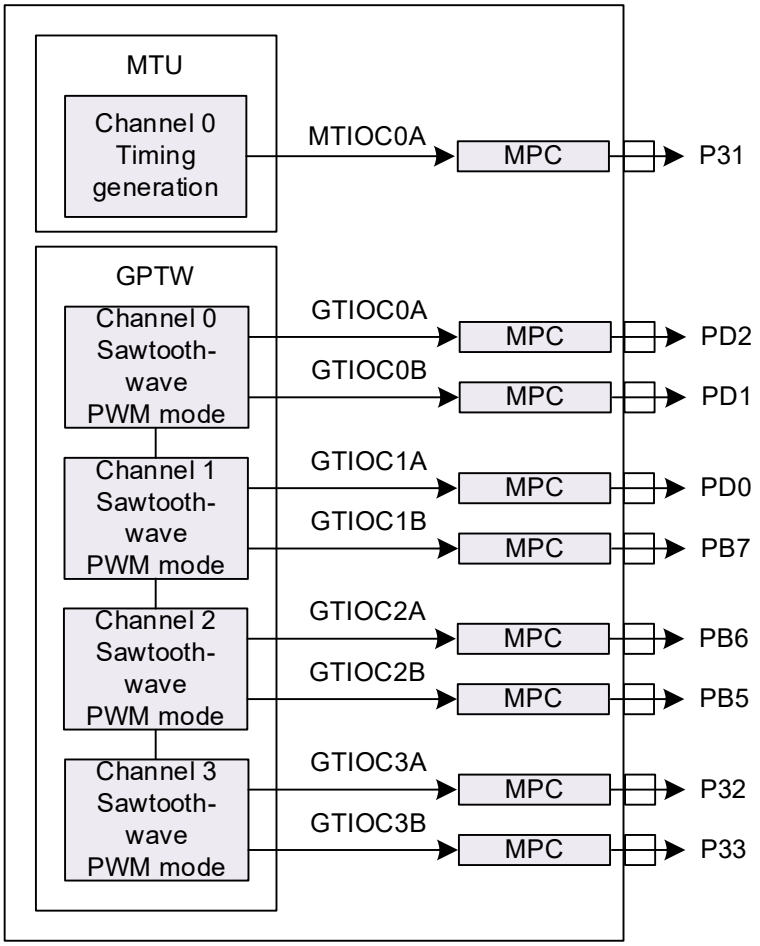
This section describes a sample code that repeatedly performs synchronous count start and synchronous stop/clearing for GPTW0 to GPTW3 (channels 0 to 3) by software source according to the compare match interrupt of MTU0 (channel 0).

The following list provides the MTU and GPTW settings used in the sample code.

- MTU0 (channel 0)
  - Use normal mode timer
  - Initial output value = low
  - Carrier period = 200µs
  - Timer counter clock = 160MHz (PCLKC)
  - Use TGRA as period register
    - Timer counter clear source = TGRA compare match
    - Toggle output at TGRA compare match
- GPTW0 to GPTW3 (channels 0 to 3)
  - Use sawtooth-wave PWM mode
  - Low output at counting starts, low output at counting stops
  - Low output at cycle end
  - Carrier period = 400µs
  - Timer counter clock = 160MHz (PCLKC)
  - Use GTPR as period register
    - Count direction = up-counting
    - Counter initial value = 0
  - Use GPTWn.GTCCRA as duty register (n = 0 to 3)
    - Use GTIOCnA pin as PWM output pin
    - High output at GPTWn.GTCCRA compare match
  - Use GPTWn.GTCCRB as duty register (n = 0 to 3)
    - Use GTIOCnB pin as PWM output pin
    - High output at GPTWn.GTCCRB compare match
  - Software source count start,
    software source count stop, and
    software source count clear enabled

Set in Smart Configurator.
For Setting Methods,
refer to section 4.2.3.

The structure of this sample code is shown below.



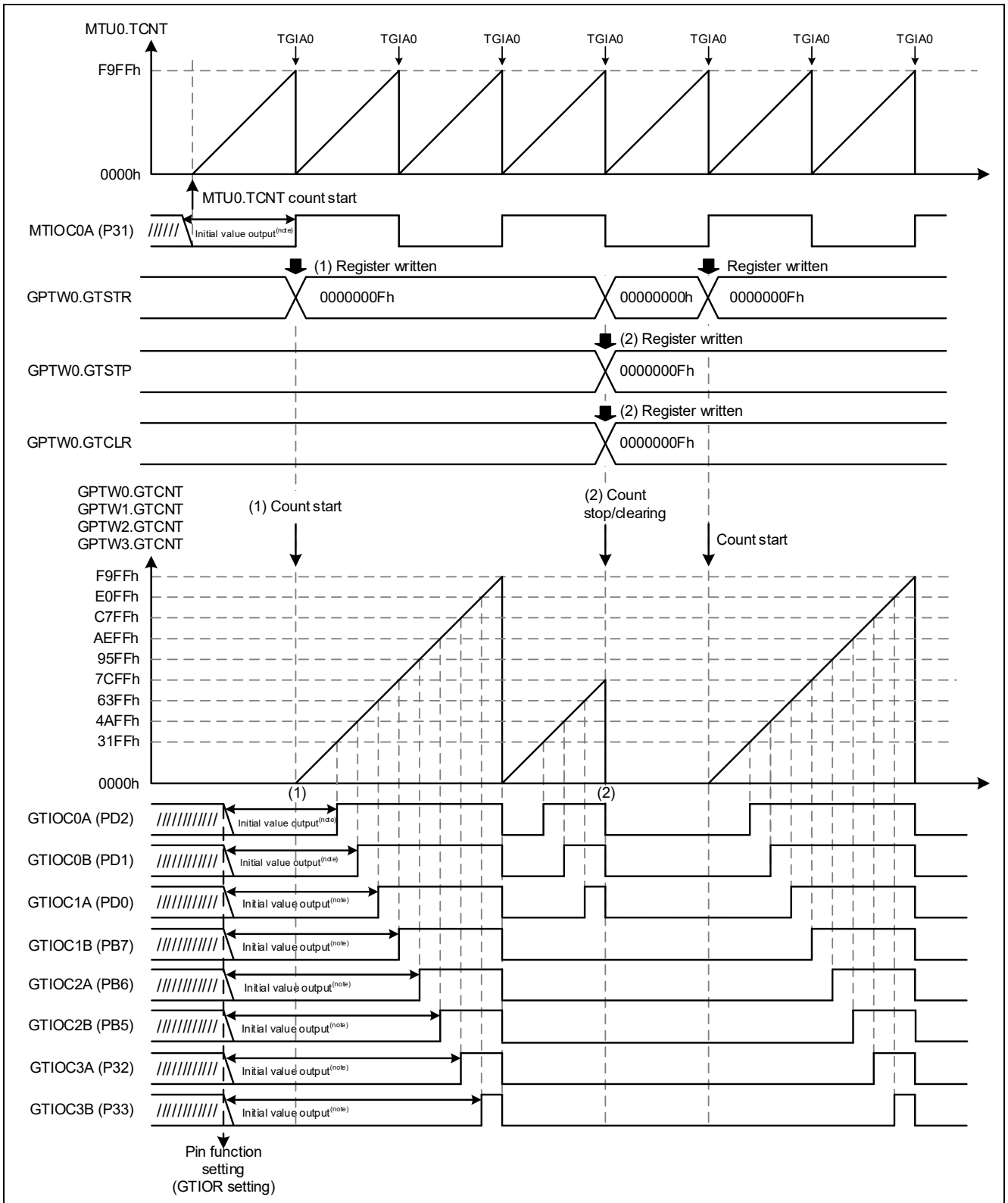
MTU0: Generates timing for synchronous operations
GPTW0 to GPTW3: Generates sawtooth-wave PWM mode waveform
MPC: Sets pins to be used from general purpose I/O port to peripheral function I/O port

Figure 4.6 Sample Code Structure

4.2.2 Operation Details

The sample code operations are shown in Figure 4.7. Use the TGRA of MTU0 as the period register to set the count synchronous start or synchronous stop/clearing for GPTW0 to GPTW3 when a TGRA compare match interrupt is generated.

- Synchronous start
  When the first TGRA compare match interrupt is generated, software start register GPTW0.GTSTR is set and GPTW0 to GPTW3 start counting in synchronization ((1) in Figure 4.7).
- Synchronous stop/clearing
  When the third TGRA compare match interrupt is generated, software stop register GPTW0.GTSTP and software clear register GPTW0.GTCLR are set, and counting stops and the counters are cleared in synchronization ((2) in Figure 4.7).



Note: The sample code waveform starts outputting the initial values when the PMR register is set.

Figure 4.7 Sample Code Operations

4.2.3 Smart Configurator Settings

The sample codes use the Smart Configurator to add the MTU and GPTW as described below. For details on how to add MTU components, refer to section 3.1.4 Adding Components, and for details on how to add GPTW components, refer to section 4.1.4 Adding Components.

Table 4.4 Adding Components (MTU0)

Item	Description
Component	Normal Mode Timer
Configuration name	Config_MTU0
Input capture/ Output compare pins	2 pins
Resource	MTU0

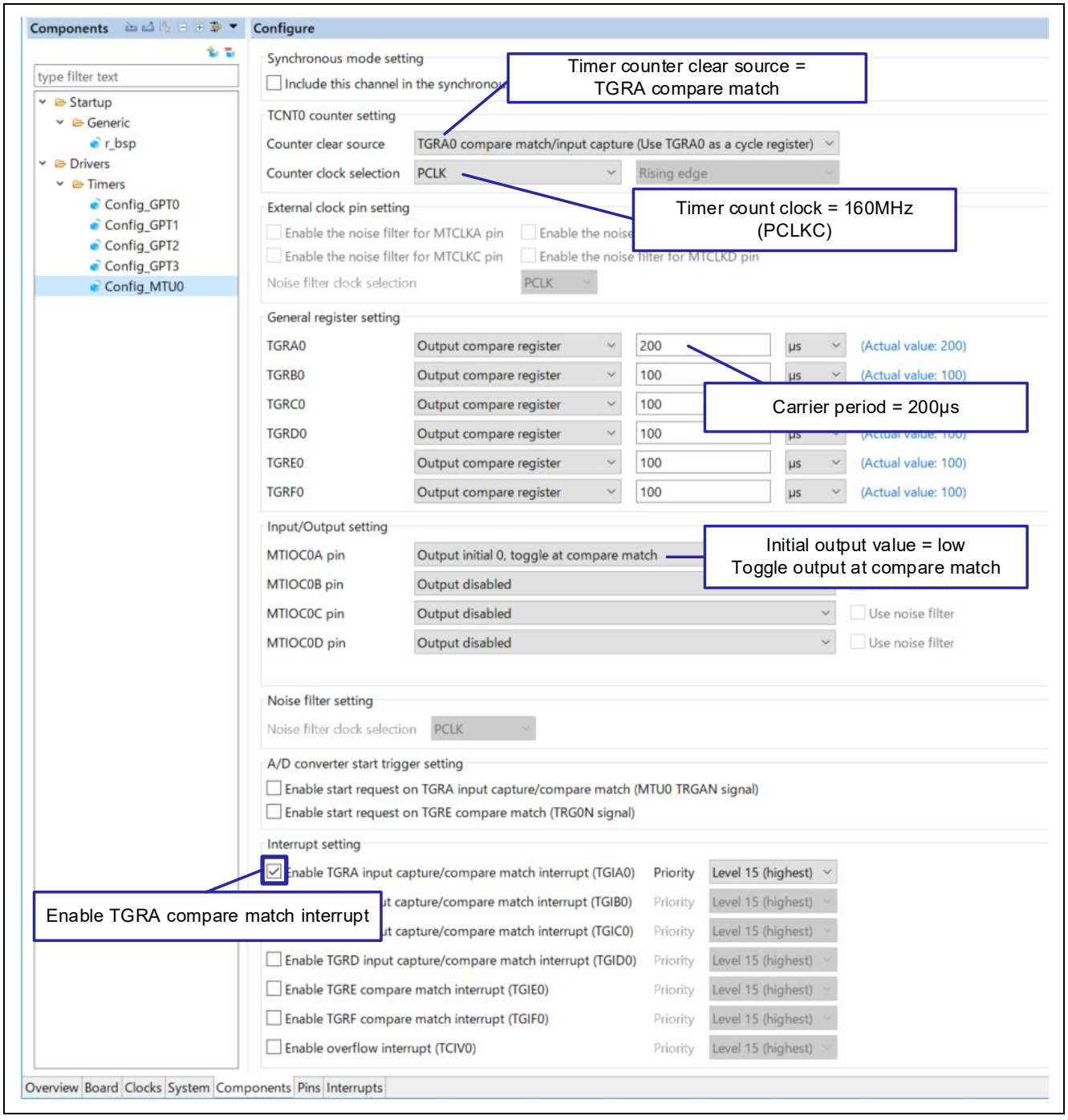


Figure 4.8 MTU0 Settings

Table 4.5 Adding Components (GPTW0 to GPTW3)

Item	Description			
Component	General PWM Timer			
Configuration name	Config_GPT0	Config_GPT1	Config_GPT2	Config_GPT3
Work mode	Sawtooth-Wave PWM Mode			
Resource	GPT0	GPT1	GPT2	GPT3

Figure 4.9 to Figure 4.12 show the Config_GPT0 settings. The same settings apply to GPT1 to GPT3. As the output duty cycles differ, the GTCCRA and GTCCRB setting values for each channel also differ.

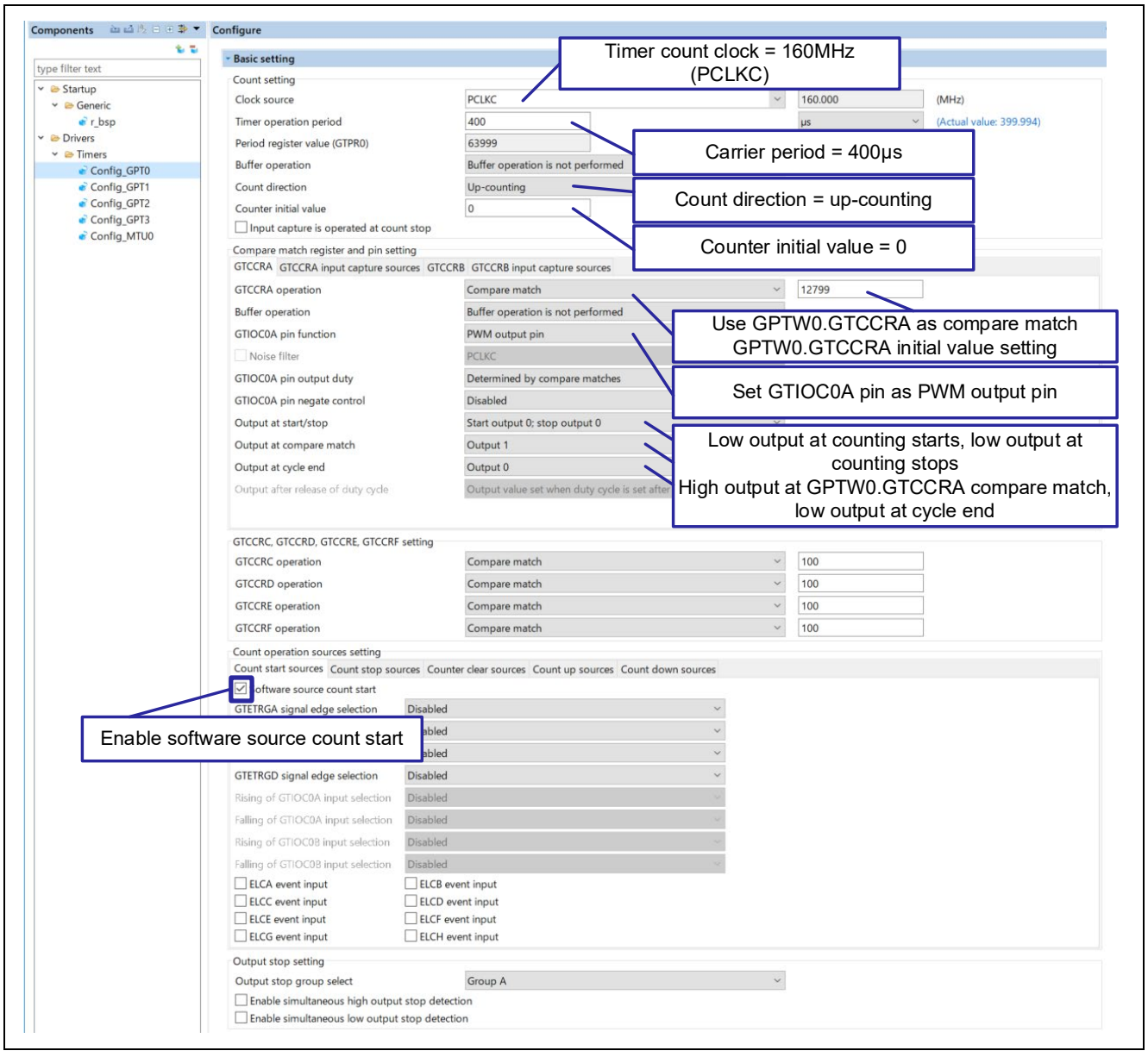


Figure 4.9 GPT0 Settings (1/4)

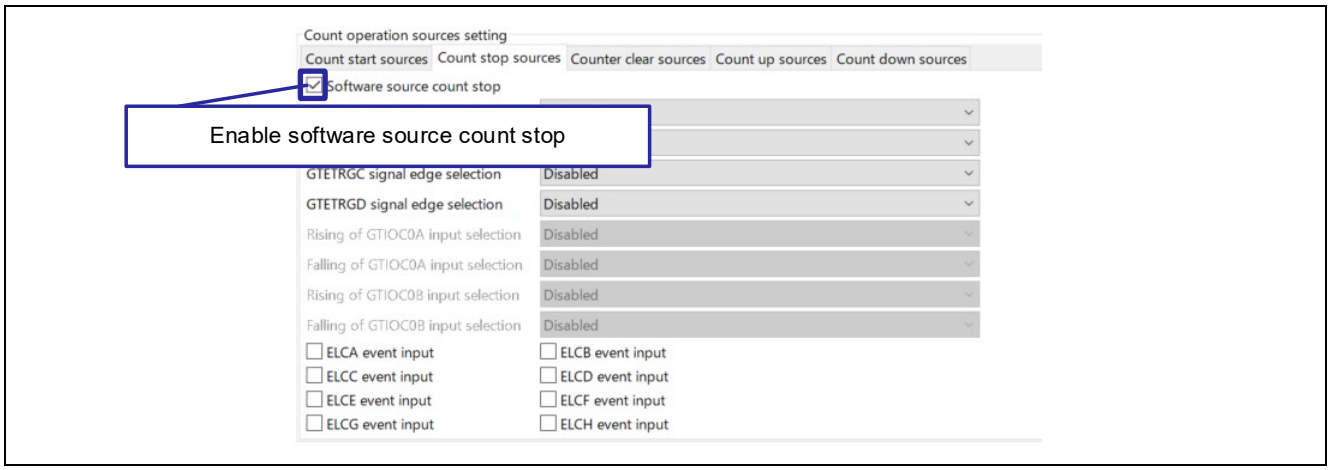


Figure 4.10 GPT0 Settings (2/4)

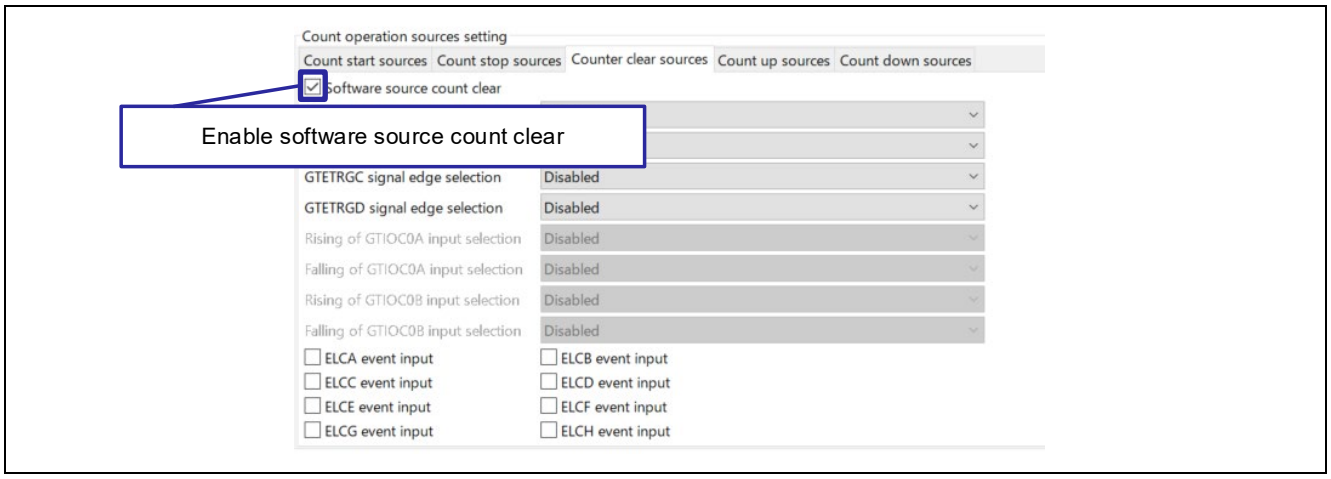


Figure 4.11 GPT0 Settings (3/4)

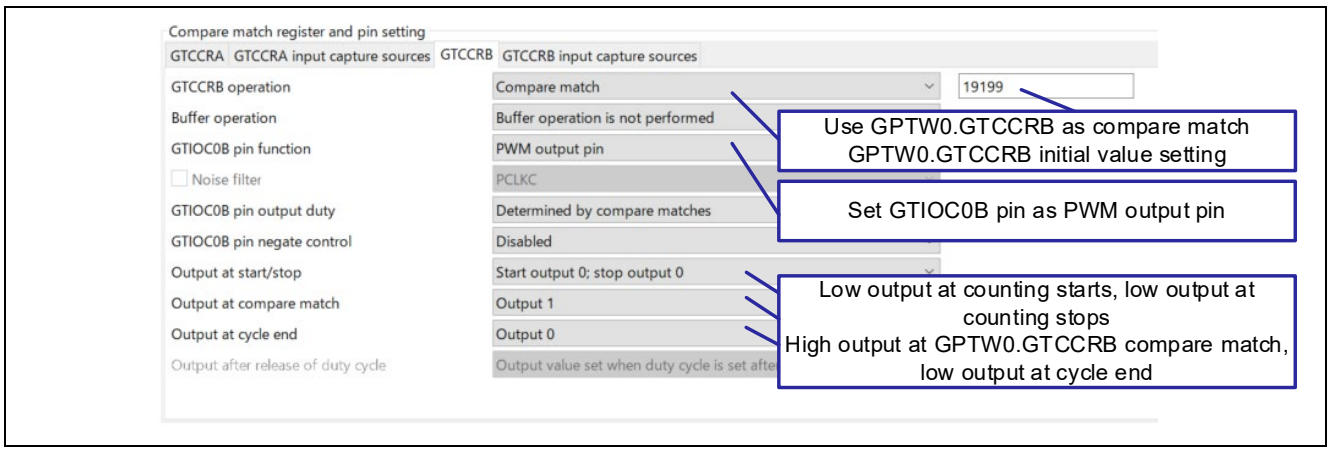


Figure 4.12 GPT0 Settings (4/4)

4.2.4 Flowcharts

The following flowchart shows the main function processing of a function added after code generation by the Smart Configurator.

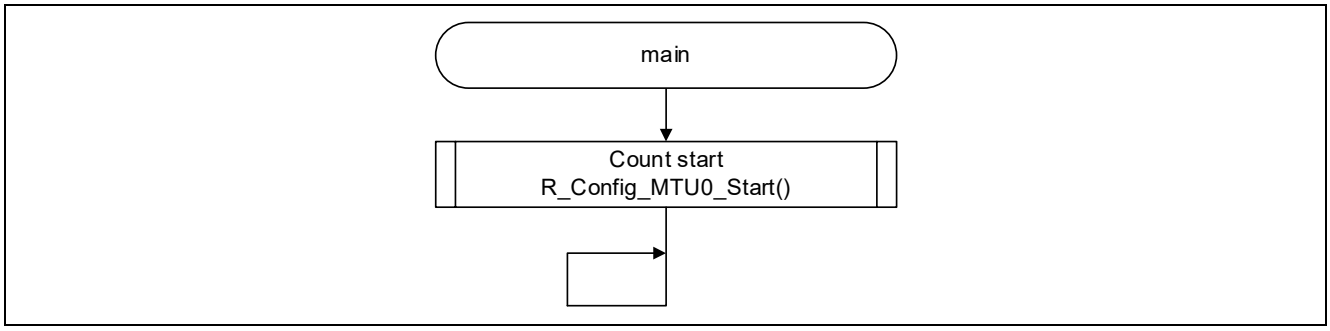


Figure 4.13 main Function

In the TGIA0 interrupt handler function, the control register for GPTW synchronous start or synchronous stop/clearing is set according to the current interrupt generation count.

This sample code uses the following variable.

- s_int_cnt: interrupt generation count variable for repeating synchronous start and synchronous stop/clearing operations

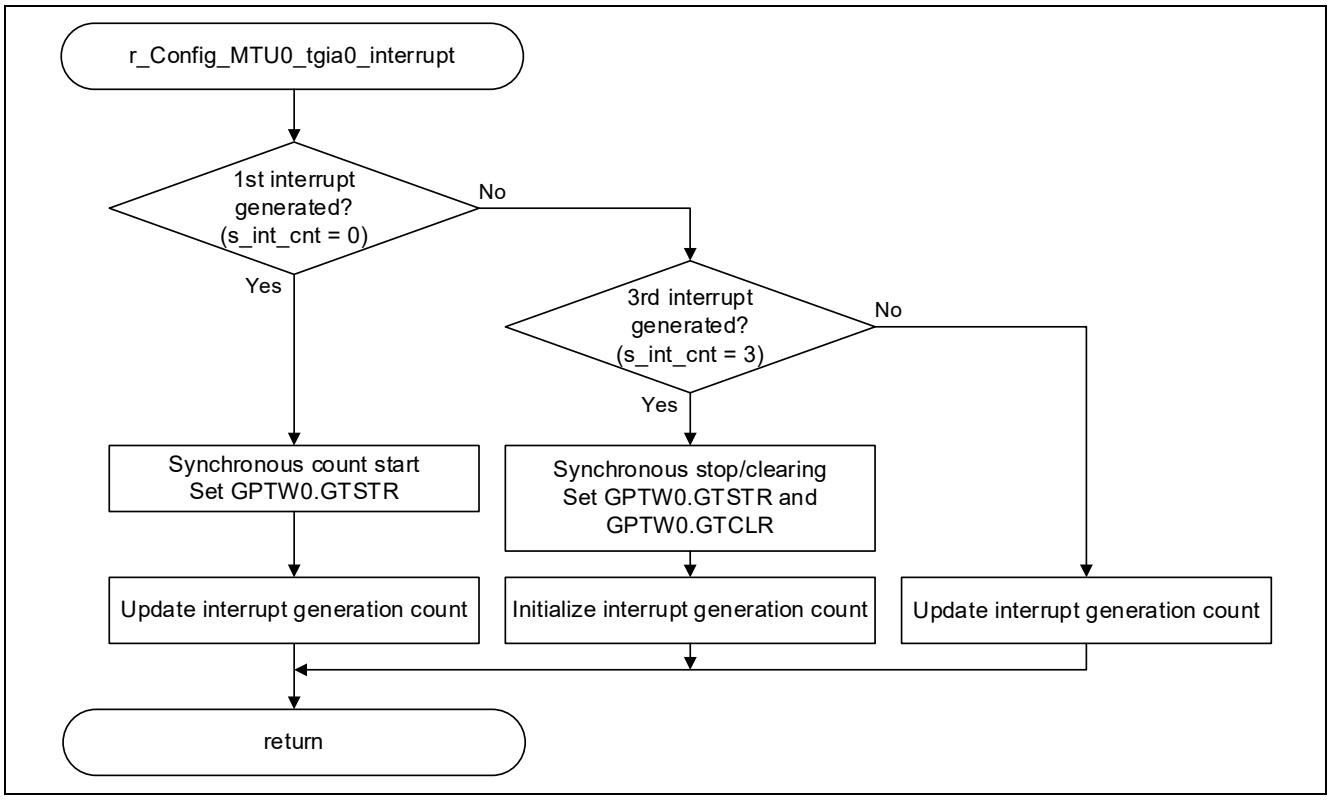


Figure 4.14 TGIA0 Interrupt Handler Function

4.2.5 Usage Notes

4.2.5.1 Counting Starts for Multiple Channels

In this sample code, the CSTRT0 to CSTRT3 bits of timer software start register GTSTR are set at the same time in the r_Config_MTU0_tgia0_interrupt function to start counting the GPTW0 to GPTW3 channels at the same time.

When using the R_Config_GPTm_Start (m = 0 to 3) function generated by the Smart Configurator, the counting starts timing may not be the same because each of the functions are read.

For details, refer to RX66T Group User's Manual: Hardware, section 24.3.8.1 Synchronous Operation by Software.

4.2.5.2 Inter-Channel Synchronous Operation Setting Register by Software

The GTSTR, GTSTP and GTCLR registers of each channel are common registers, and the channel operation at the bit position written to 1b can be performed, regardless of which channel register is updated. Writing 0b does not cause any change in counter operation or register value.

For details, refer to RX66T Group User's Manual: Hardware, sections 24.2.2 General PWM Timer Software Start Register (GTSTR), 24.2.3 General PWM Timer Software Stop Register (GTSTP), and 24.2.4 General PWM Timer Software Clear Register (GTCLR).

4.2.5.3 Order of Priority in Events

In this sample code, synchronous start/stop by software can be realized by setting multiple bits of the GTSTR and GRSTP registers to 1b at the same time.

When start/stop by a hardware source set in GTSSR/GTPSR conflicts with the CPU writing (GTSTR writing/GTSTP writing), the CPU writing takes priority.

For details, refer to RX66T Group User's Manual: Hardware, section 24.10.5 Order of Priority in Events, (2) The GTCR.CST Bit.

4.3 Synchronous Operation (Phase Shift) by Software in Sawtooth-Wave PWM Mode

- Target sample code file name: r01an6282_rx66t_gptw_sawtooth_pwm_sync_shift.zip

4.3.1 Overview

In GPTW sawtooth-wave PWM mode, synchronous start can be performed using the GTSTR register, and synchronous stop using the GTSTP register. Count start can be performed with phase differences among channels by setting the GTCNT counter value for each channel before the count start.

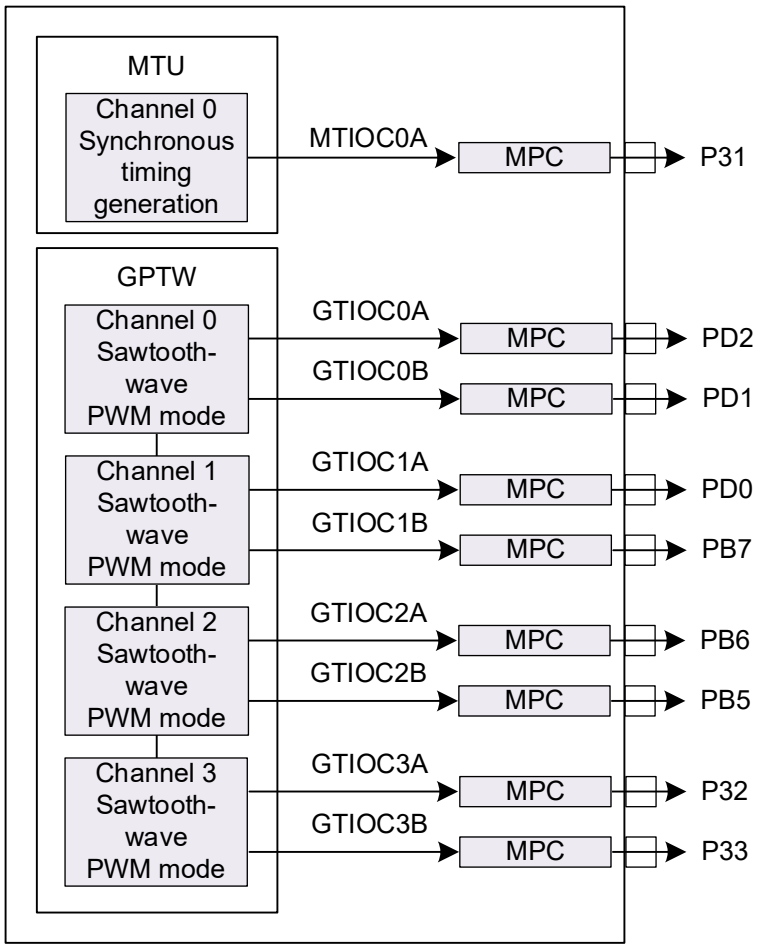
This section describes a sample code that repeatedly performs synchronous start/stop/clearing for GPTW0 to GPTW3 (channels 0 to 3) by software source, by setting the counter initial value with phase differences among channels using the Smart Configurator.

The following list provides the MTU and GPTW settings used in the sample code.

- MTU0 (channel 0)
  - Use normal mode timer
  - Initial output value = low
  - Carrier period = 200µs
  - Timer counter clock = 160MHz (PCLKC)
  - Use TGRA as period register
    - Timer counter clear source = TGRA compare match
    - Toggle output at TGRA compare match
- GPTW0 to GPTW3 (channels 0 to 3)
  - Use sawtooth-wave PWM mode
  - Low output at counting starts, low output at counting stops
  - Low output at cycle end
  - Carrier period = 400µs
  - Timer counter clock = 160MHz (PCLKC)
  - Use GTPR as period register
    - Count direction = up-counting
    - GPTW0 counter initial value = 9599 (15% of cycle)
    - GPTW1 counter initial value = 6399 (10% of cycle)
    - GPTW2 counter initial value = 3199 (5% of cycle)
    - GPTW3 counter initial value = 0
  - Use GPTWn.GTCCRA as duty register (n = 0 to 3)
    - Use GTIOCnA pin as PWM output pin
    - High output at GPTWn.GTCCRA compare match
  - Use GPTWn.GTCCRB as duty register (n = 0 to 3)
    - Use GTIOCnB pin as PWM output pin
    - High output at GPTWn.GTCCRB compare match
  - Software source count start,
    software source count stop, and
    software source count clearing enabled

Set in Smart Configurator.
For Setting Methods,
refer to section 4.3.3.

The structure of this sample code is shown below.



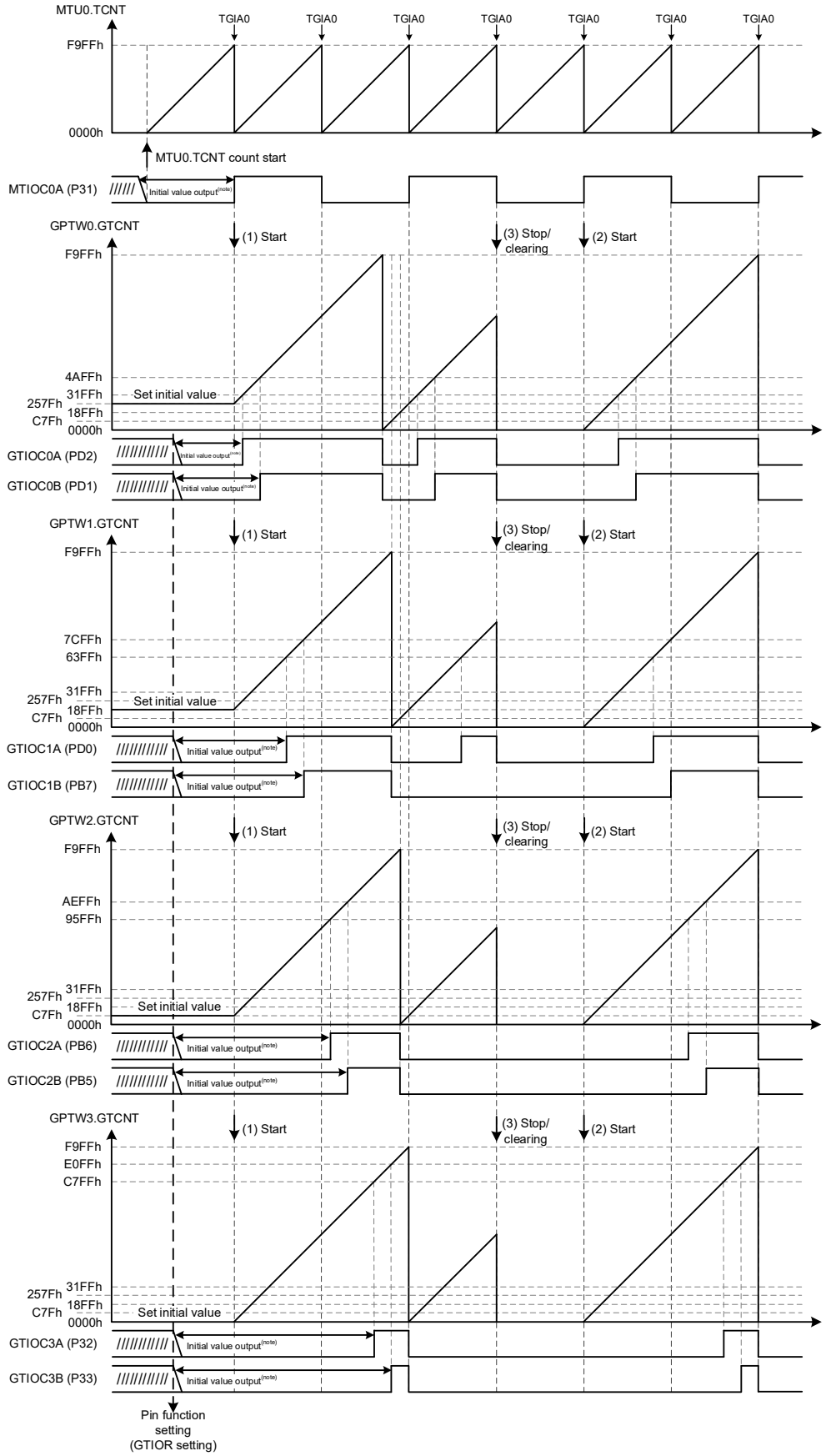
MTU0: Generates synchronous timing
GPTW0 to GPTW3: Generate sawtooth-wave PWM mode waveform
MPC: Sets pins to be used from general purpose I/O port to peripheral function I/O port

Figure 4.15 Sample Code Structure

4.3.2 Operation Details

The sample code operations are shown in Figure 4.16. Use the TGRA of MTU0 as the period register to set the count synchronous start or synchronous stop/clearing for GPTW0 to GPTW3 when a TGRA compare match interrupt is generated.

- Synchronous start
  When the first TGRA compare match interrupt (TGIA0) is generated, GPTW0.GTSTR is set and GPTW0 to GPTW3 counting starts in synchronization from the counter initial value of each channel, enabling phase shift start (count start with phase shift differences among channels) ((1) in Figure 4.16).
  After the second count start, the phase shift does not occur because the counter initial value set for each channel is initialized by synchronous clearing ((2) in Figure 4.16).
- Synchronous stop/clearing
  When the 4th TGRA compare match interrupt is generated, software stop register GPTW0.GTSTP and software clear register GPTW0.GTCLR are set, GPTW0 to GPTW3 counting stops in synchronization, and the counter is cleared ((3) in Figure 4.16).



Note: The sample code waveform starts outputting the initial values when the PMR register is set.

Figure 4.16 Sample Code Operations

4.3.3 Smart Configurator Settings

The sample codes use the Smart Configurator to add the MTU and GPTW as described below. For details on how to add MTU components, refer to section 3.1.4 Adding Components, and for details on how to add GPTW components, refer to section 4.1.4 Adding Components.

Table 4.6 Adding Components (MTU0)

Item	Description
Component	Normal Mode Timer
Configuration name	Config_MTU0
Input capture/ Output compare pins	2 pins
Resource	MTU0

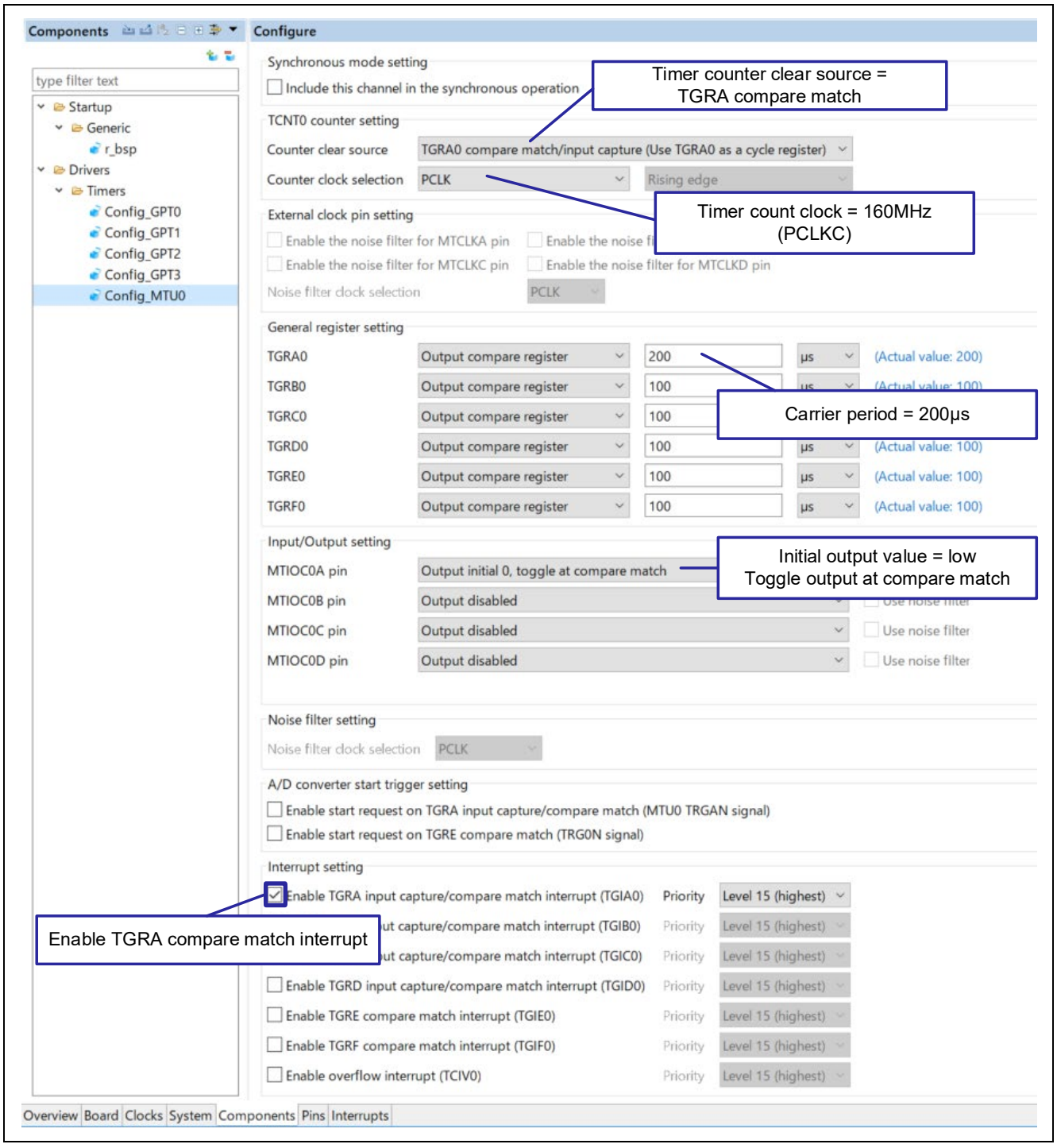


Figure 4.17 MTU0 Settings

Table 4.7 Adding Components (GPTW0 to GPTW3)

Item	Description			
Component	General PWM Timer			
Configuration name	Config_GPT0	Config_GPT1	Config_GPT2	Config_GPT3
Work mode	Sawtooth-Wave PWM Mode			
Resource	GPT0	GPT1	GPT2	GPT3

Figure 4.18 to Figure 4.21 show the Config_GPT0 settings. The settings for GPT1 to GPT3 are basically the same. Due to the phase differences among channels, the counter initial value for each channel also differs. As the output duty cycles differ, the GTCCRA and GTCCRB setting values for each channel also differ.

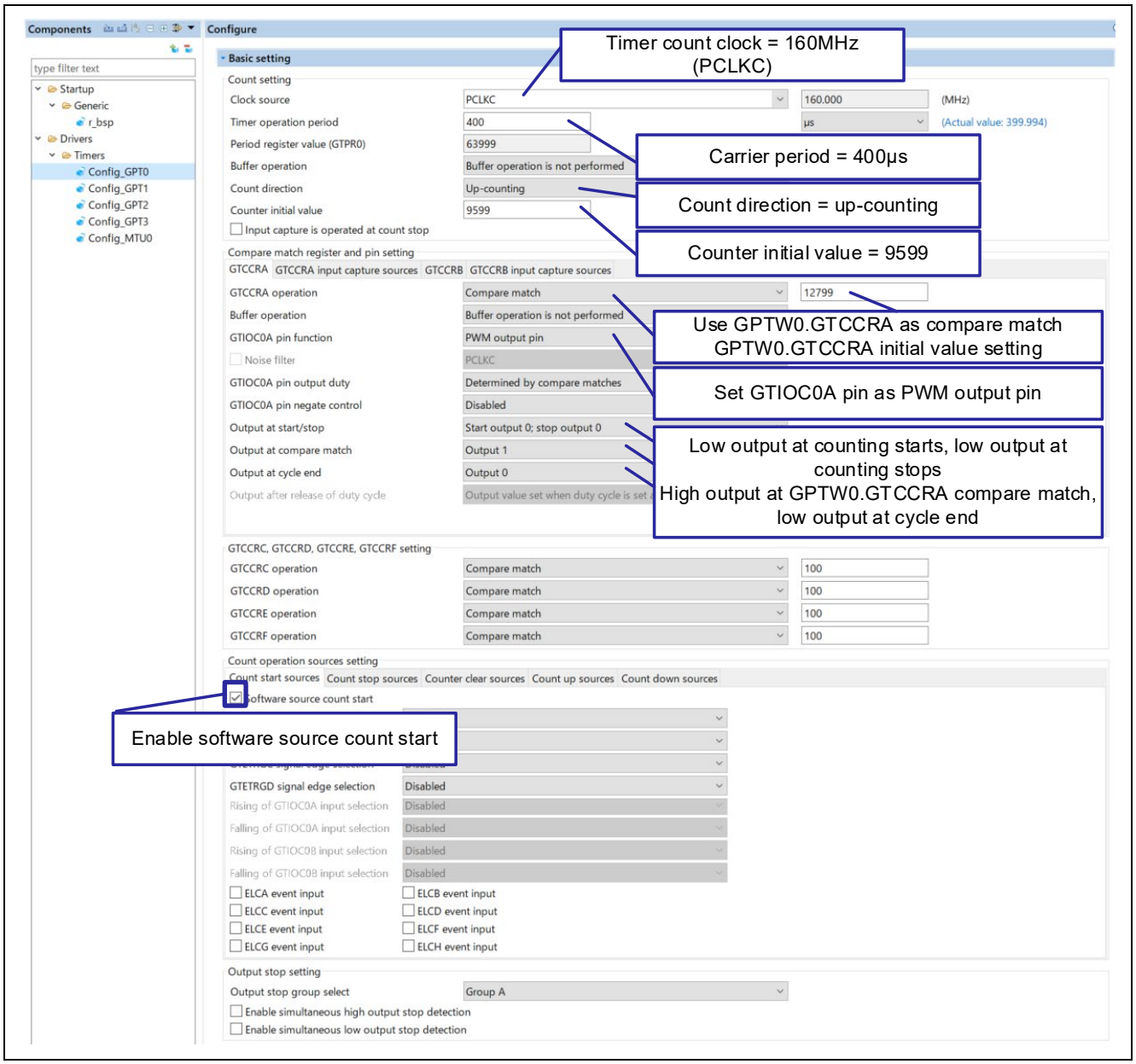


Figure 4.18 GPT0 Settings (1/4)

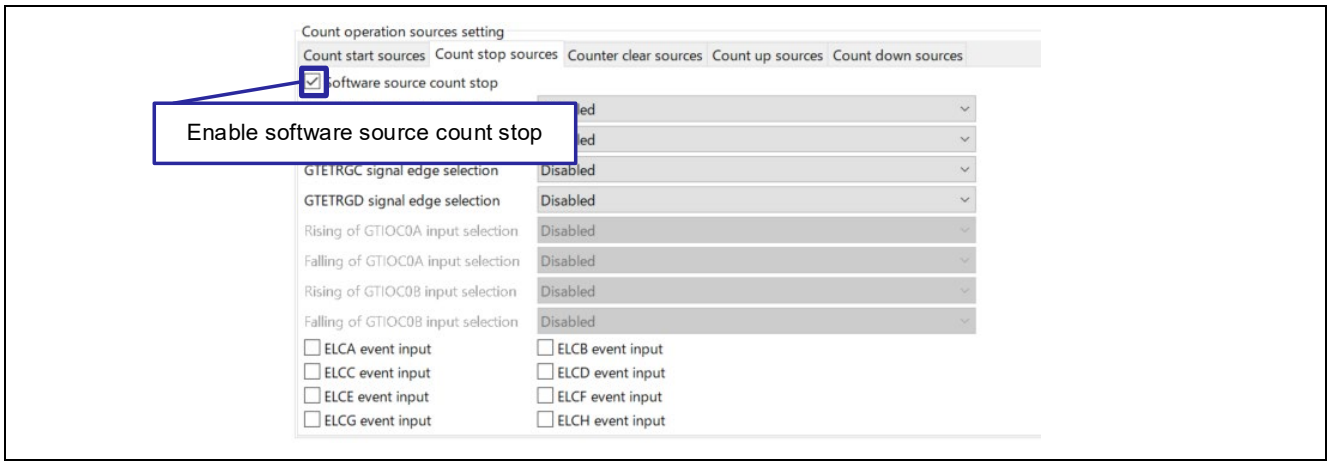


Figure 4.19 GPT0 Settings (2/4)

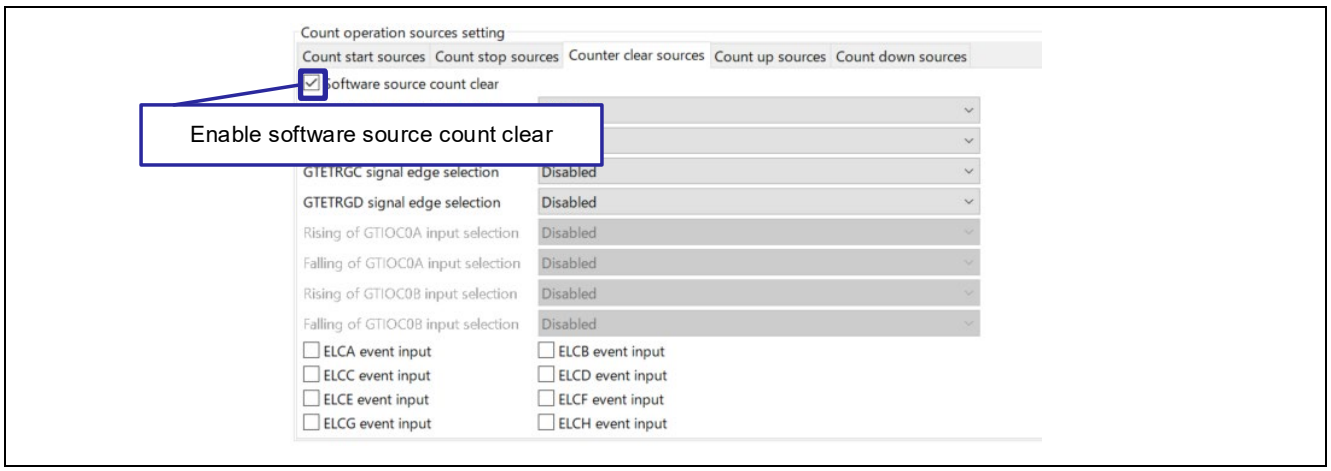


Figure 4.20 GPT0 Settings (3/4)

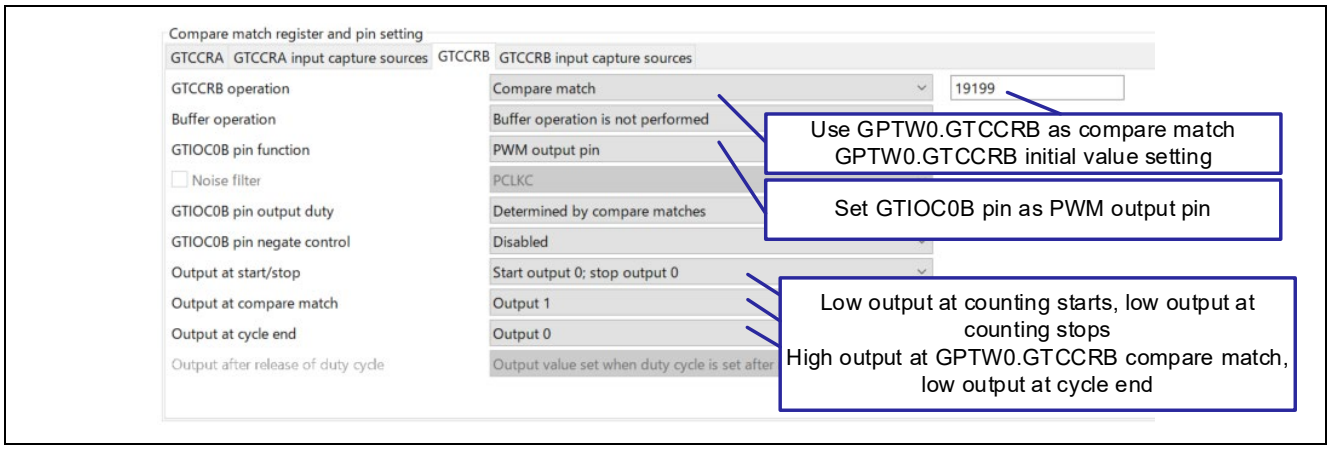


Figure 4.21 GPT0 Settings (4/4)

4.3.4 Flowcharts

The following flowchart shows the main function processing of a function added after code generation by the Smart Configurator.

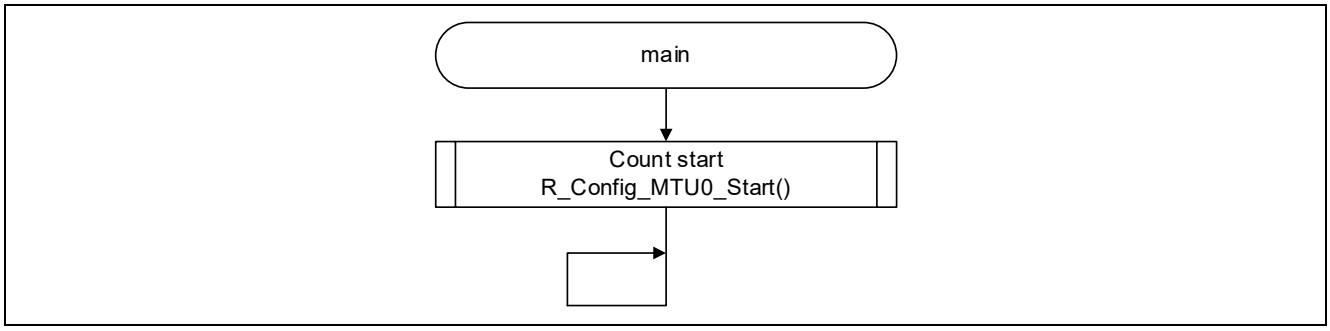


Figure 4.22 main Function

In the TGIA0 interrupt handler function, the control register for GPTW synchronous start or synchronous stop/clearing is set according to the current interrupt generation count.

This sample code uses the following variable.

- s_int_cnt: interrupt generation count variable for repeating synchronous start and synchronous stop/clearing operations

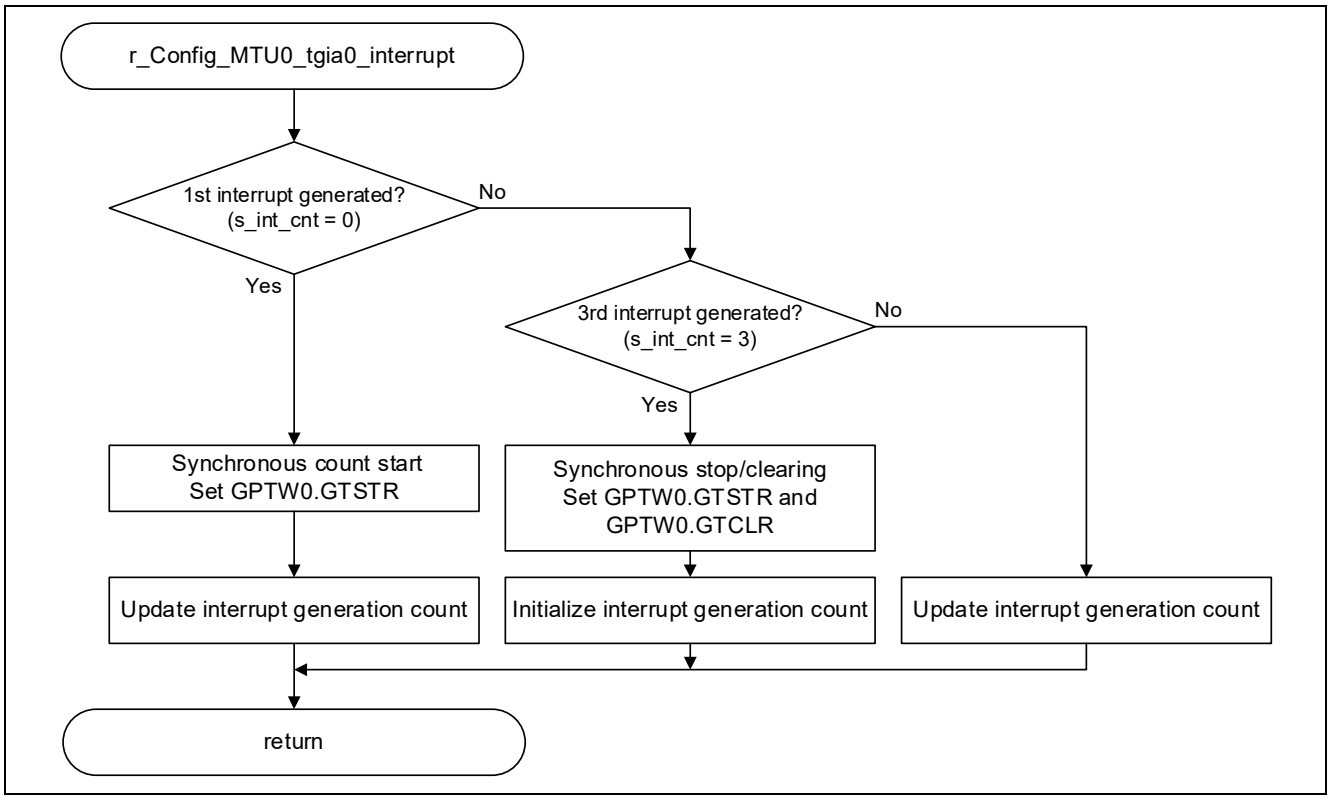


Figure 4.23 TGIA0 Interrupt Handler Function

4.3.5 Related Operations

4.3.5.1 When a Phase Difference is Set so Counter Initial Value > Compare Value

The following describes an example of phase shift start operation when the counter initial value is set to a value higher than the compare value.

- GPTW0: when counter initial value > GTCCRA and GTCCRB compare values
  In the first cycle, counting starts from 577Fh, which is greater than the compare value, and because neither GTCCRA nor GTCCRB compare match occurs, the waveform output does not change ((1) in figure below). In the second cycle, counting starts from counter value 0000h, so both GTCCRA and GTCCRB compare matches occur and output goes high ((2) in figure below).
- GPTW1: when counter initial value < GTCCRA and GTCCRB compare values
  In the first cycle, counting starts from 18FFh, which is less than the compare value, so GTCCRA and GTCCRB compare matches occur and output goes high ((3) in figure below).

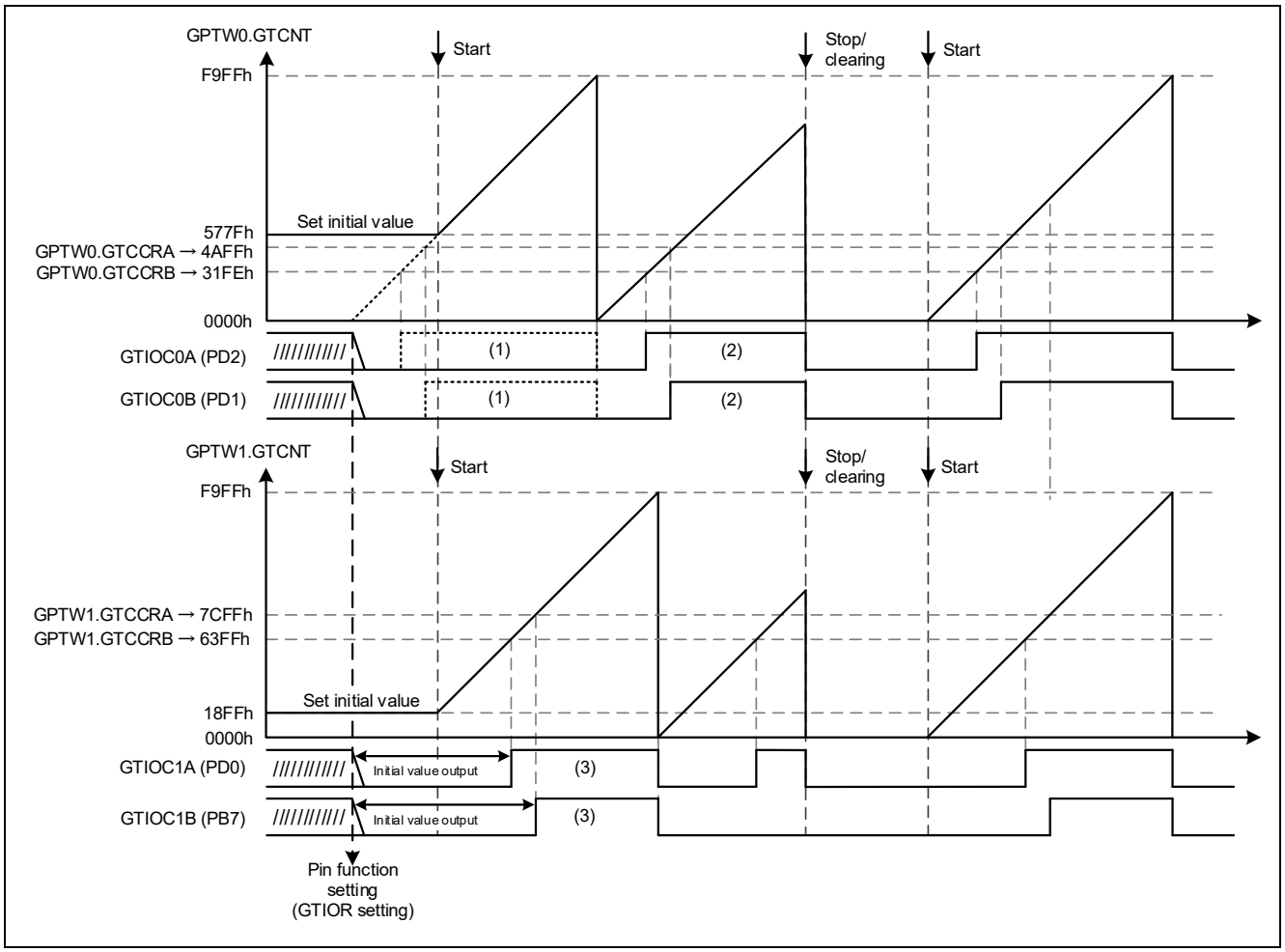


Figure 4.24 Operation Example of Phase Shift Start (GPTW0: Counter Initial Value > Compare Value)

4.3.5.2 Output Change by Synchronous Stop/Clearing Operation After Phase Shift Start

The following describes an example of operation when synchronous stop or synchronous clearing is performed after the phase shift start of GPTW0 and GPTW1.

- Phase shift start → synchronous stop
  GPTW0.GTSTP is set during count operation after the phase shift starts and GPTW0 and GPTW1 are synchronously stopped ((1) in figure below). From the next synchronous start ((2) in figure below), the counter is incremented from the counter value at the time the count stops, so the set phase difference is retained.
  The following operation example describes a case in which the GTIOR.OADFLT (OBDFLT) bit is set to low output when counting stops. At (1) stop, the GPTW0 pin output level changes from high to low, and the GPTW1 remains at low ((3) in figure below).

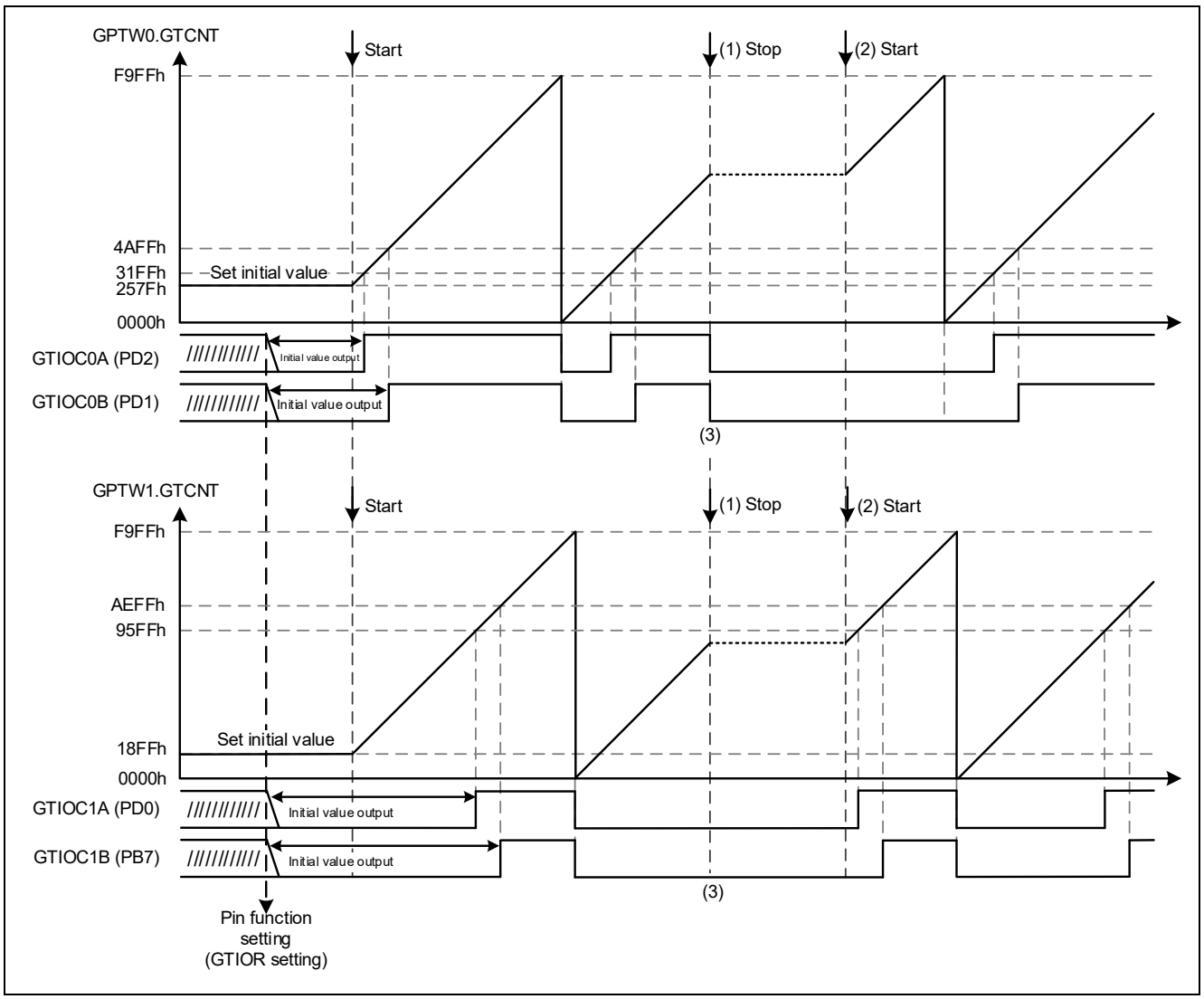


Figure 4.25 Operation Example of Synchronous Stop After Phase Shift Starts

- Phase shift start → synchronous clearing
  When GPTW0.GTCLR is set during count operation after the phase shift starts and GPTW0 and GPTW1 are synchronously cleared ((1) in figure below), the set phase difference is lost because the counter is incremented from counter value 0000h.
  The following operation example describes a case in which the GTIOR.GTIOA (GTIOB) bit is set to low output at cycle end. At (1) clearing, the GPTW0 pin output level changes from high to low, and the GPTW1 remains at low ((2) in figure below).

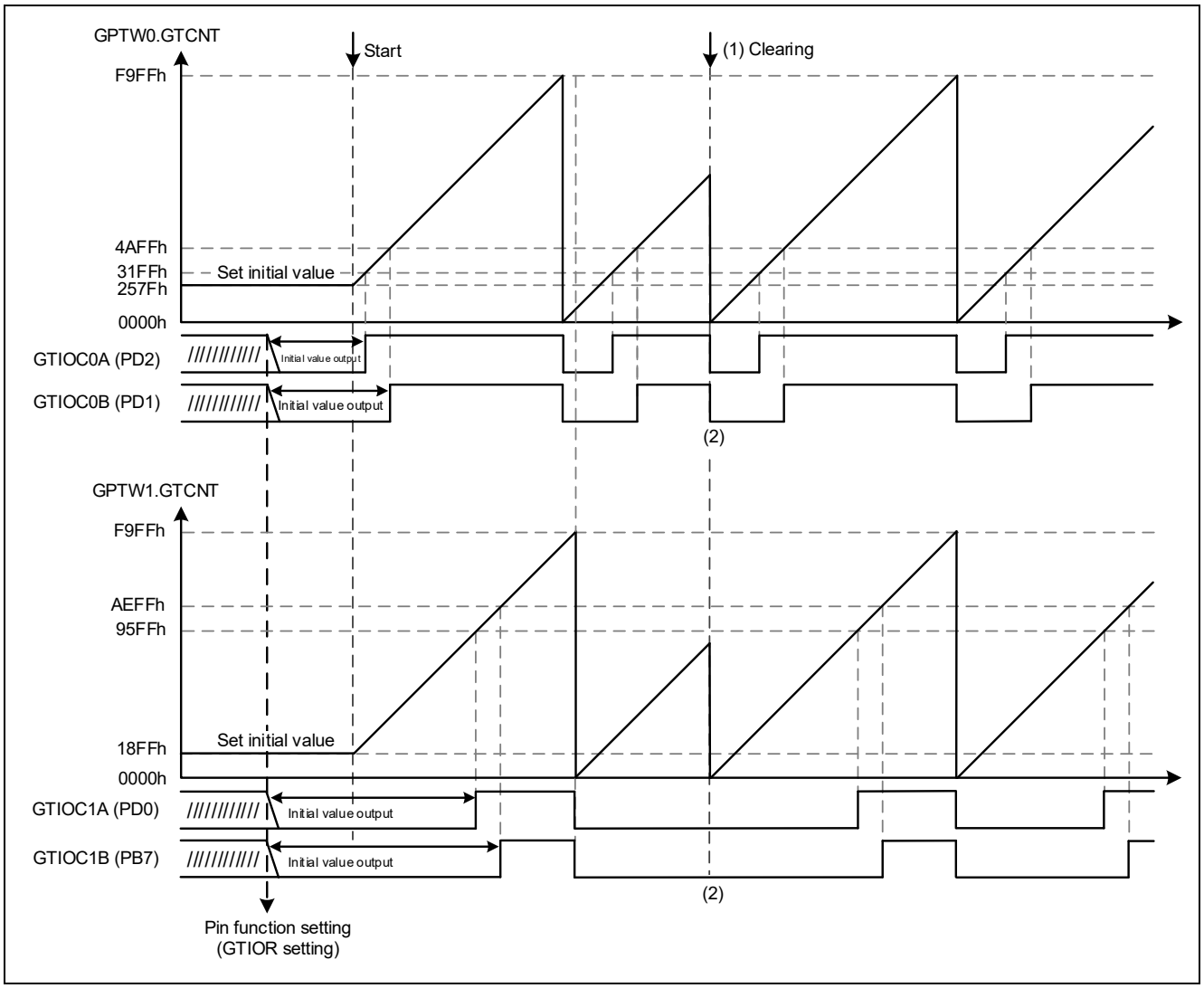


Figure 4.26 Operation Example of Synchronous Clearing After Phase Shift Starts

4.3.6 Usage Notes

4.3.6.1 Counting Starts for Multiple Channels

In this sample code, the CSTRT0 to CSTRT3 bits of timer software start register GTSTR are set at the same time in the r_Config_MTU0_tgia0_interrupt function to start counting GPTW0 to GPTW3 at the same time.

When using the R_Config_GPTm_Start (m = 0 to 3) function generated by the Smart Configurator, the counting starts timing may not be the same because each of the functions are read.

For details, refer to RX66T Group User's Manual: Hardware, section 24.3.8.1 Synchronous Operation by Software.

4.3.6.2 Inter-channel Synchronous Operation Setting Register by Software

The GTSTR, GTSTP, and GTCLR registers of each channel are common registers, and the channel operation at the bit position written to 1b can be performed, regardless of which channel register is updated. Writing 0b does not cause any change in counter operation or register value.

For details, refer to RX66T Group User's Manual: Hardware, sections 24.2.2 General PWM Timer Software Start Register (GTSTR), 24.2.3 General PWM Timer Software Stop Register (GTSTP), and 24.2.4 General PWM Timer Software Clear Register (GTCLR).

4.3.6.3 Order of Priority in Events

In this sample code, synchronous start/stop by software can be realized by setting multiple bits of the GTSTR and GRSTP registers to 1b at the same time.

When a start/stop by a hardware source set in GTSSR and GTPSR conflicts with a CPU write (GTSTR write/GTSTP write), the CPU write takes priority.

For details, refer to RX66T Group User's Manual: Hardware, section 24.10.5 Order of Priority in Events (2) The GTCR.CST Bit.

4.3.6.4 Setting a Value Greater than the Compare Value as the Counter Initial Value

In this sample code, the counter initial value for all channels is set to a value less than the first compare value.

If the counter initial value is set to a value greater than the first compare value, the first compare match may not occur, and the output waveform may appear inverted.

For details, refer to section 4.3.5.1 When a Phase Difference is Set so Counter Initial Value > Compare Value

4.4 Synchronous Operation (Phase Shift) by Software in Triangle-Wave PWM Mode

- Target sample code file name: r01an6282_rx66t_gptw_triangle_sync_shift.zip

4.4.1 Overview

In triangle-wave PWM mode, count start can be performed at the same time with phase differences among channels, as described for sawtooth-wave PWM mode in section 4.3, by setting the GTCNT counter value for each channel before the count start and performing synchronous start.

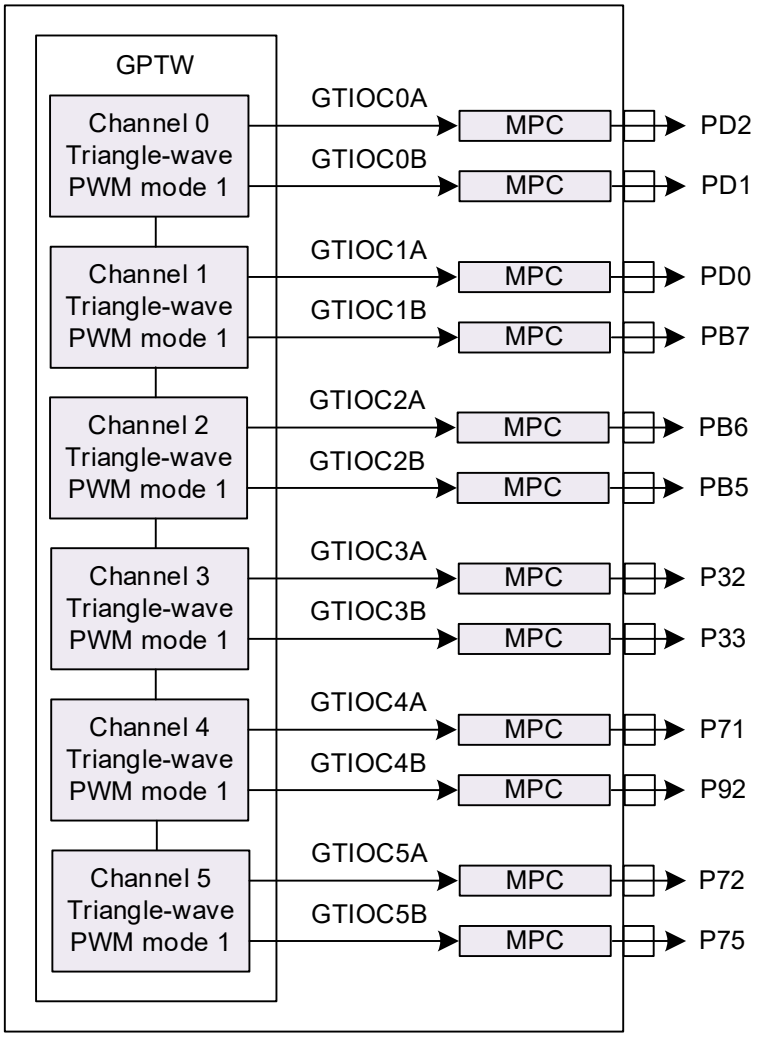
This section describes a sample code that performs synchronous start for GPTW0 to GPTW5 (channels 0 to 5) by software source and outputs 6-phase complementary PWM, by setting the counter initial value with phase differences among channels using the Smart Configurator.

The following list provides the GPTW settings used in the sample code.

GPTW0 to GPTW5 (channels 0 to 5)
- Use triangle-wave PWM mode 1 (32-bit transfer at trough)
- Retain output at cycle end
- Carrier period = 1ms
- Timer counter clock = 160MHz (PCLKC)
- Use GTPR as period register
  - Count direction = up-counting
  - GPTW0 counter initial value = 40000 (50% of cycle)
  - GPTW1 counter initial value = 32000 (40% of cycle)
  - GPTW2 counter initial value = 24000 (30% of cycle)
  - GPTW3 counter initial value = 16000 (20% of cycle)
  - GPTW4 counter initial value = 8000 (10% of cycle)
  - GPTW5 counter initial value = 0
- Use GPTWn.GTCCRA as duty register (n = 0 to 5)
  - Use GTIOCnA pin as PWM output pin
  - High output at counting starts, high output at counting stops
  - Toggle output at GPTWn.GTCCRA compare match
- Use GPTWn.GTCCRB as duty register (n = 0 to 5)
  - Use GTIOCnB pin as PWM output pin
  - Low output at counting starts, low output at counting stops
  - Toggle output at GPTWn.GTCCRB compare match
- Use automatic dead time generation
- Software source count start enabled

Set in Smart Configurator.
For Setting Methods,
refer to section 4.4.3.

The structure of this sample code is shown below.



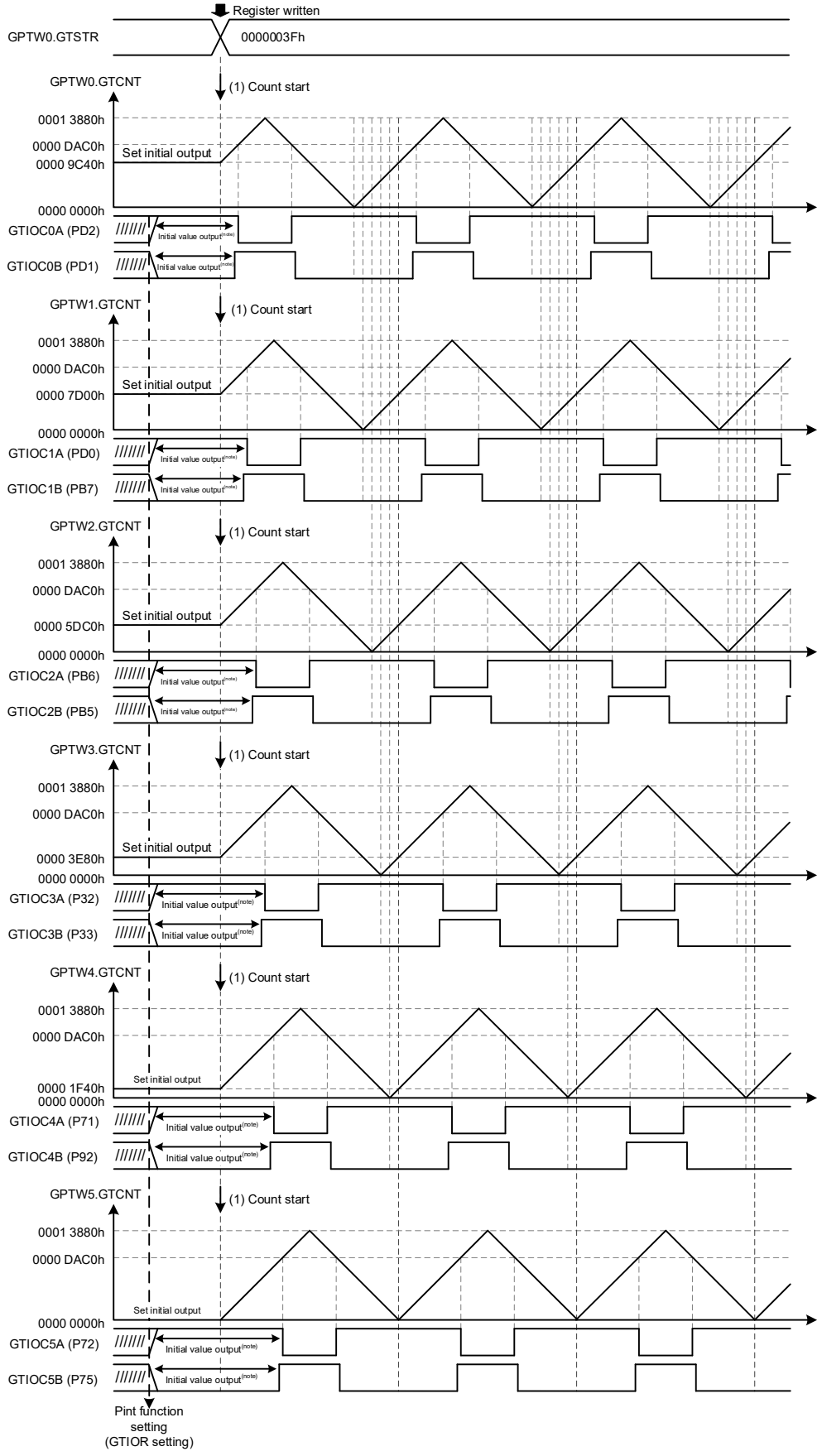
GPTW0 to 5: Generate triangle-wave PWM mode 1 waveform
MPC: Sets pins to be used from general purpose I/O port to peripheral function I/O port

Figure 4.27 Sample Code Structure

4.4.2 Operation Details

The sample code operations are shown in Figure 4.28. Synchronous count for GPTW0 to GPTW5 starts from the counter initial value of each channel and phase shift start is enabled (count start with phase differences among channels) by setting software start register GPTW0.GTSTR ((1) in Figure 4.16).

In addition, pin outputs of the sample code are set to a uniform duty cycle to clarify the phase difference.



Note: The sample code waveform starts outputting the initial values when the PMR register is set.

Figure 4.28 Sample Code Operations

4.4.3 Smart Configurator Settings

The sample codes use the Smart Configurator to add the GPTW as described below. For details on how to add GPTW components, refer to section 4.1.4 Adding Components.

Table 4.8 Adding Components (GPTW0 to 5)

Item	Description		
Component	General PWM Timer		
Configuration name	Config_GPT0	・・・	Config_GPT5
Work mode	Triangle-Wave PWM Mode 1		
Resource	GPT0	・・・	GPT5

Figure 4.29 and Figure 4.30 show the Config_GPT0 settings. The settings for GPT1 to GPT5 are basically the same. Due to the phase differences among channels, the counter initial value for each channel also differs.

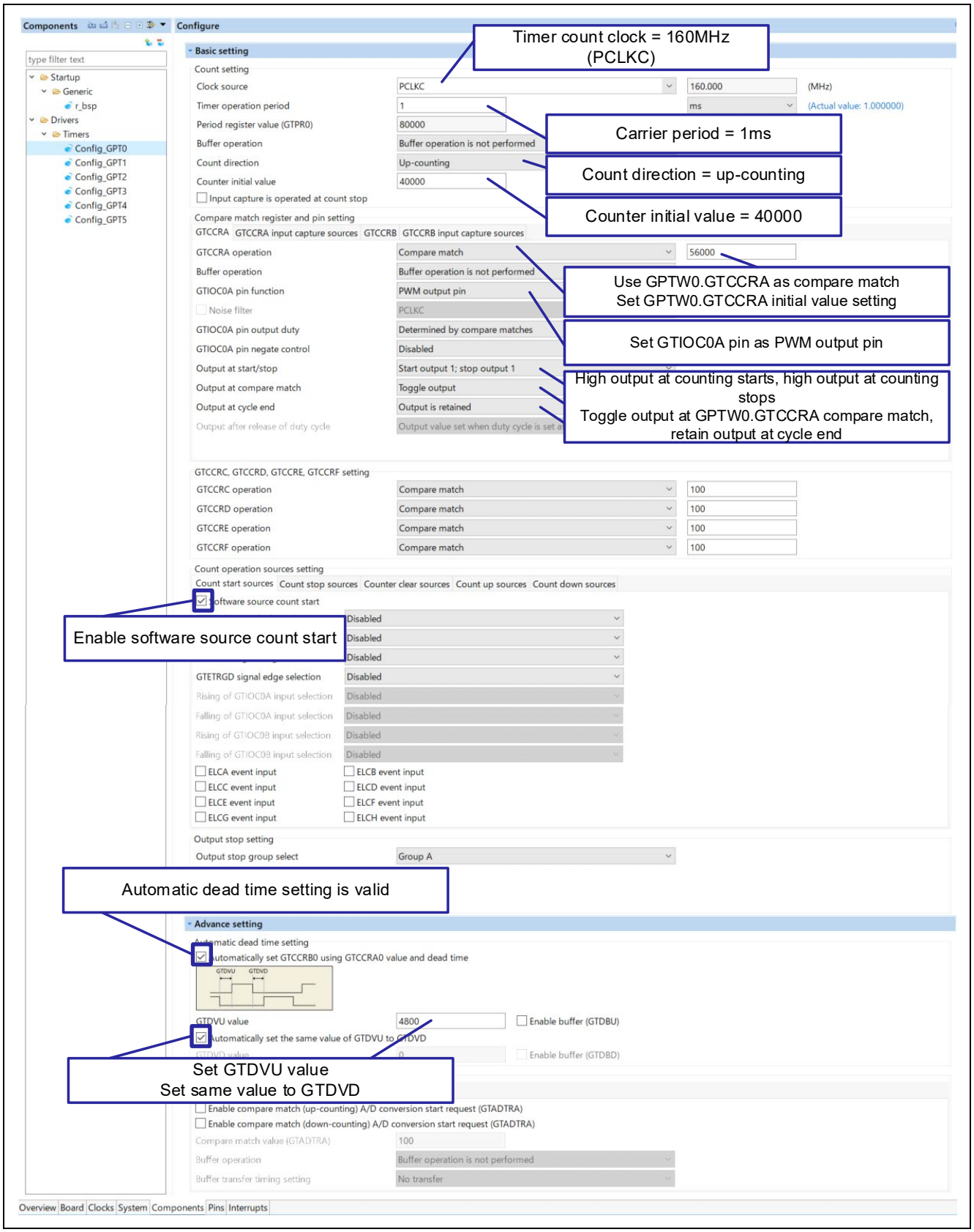


Figure 4.29 GPT0 Settings (1/2)

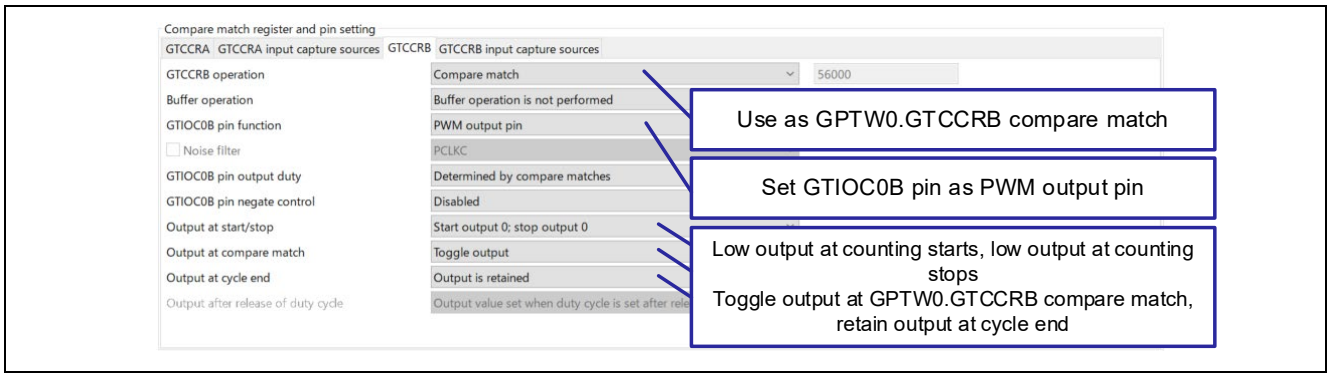


Figure 4.30 GPT0 Settings (2/2)

4.4.4 Flowcharts

The following flowchart shows the main function processing of a function added after code generation by the Smart Configurator.

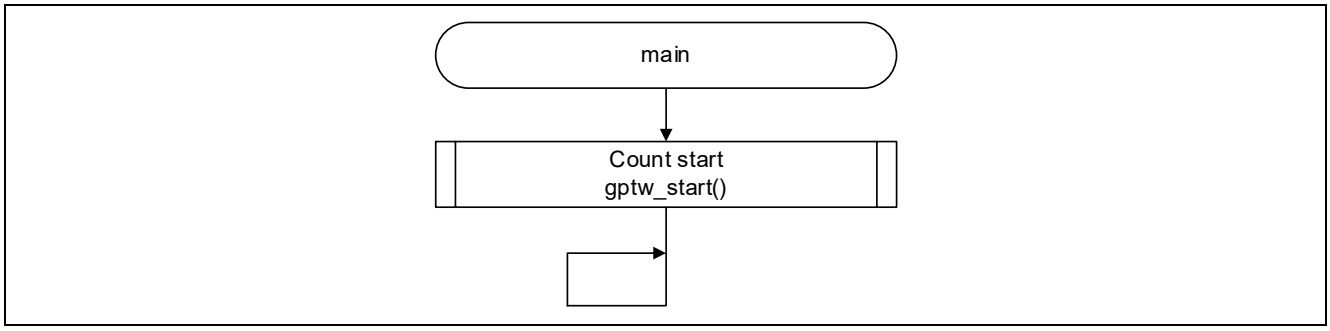


Figure 4.31 main Function

Counting is started for GPTW0 to GPTW5 in the count start function.

This function is newly created after code generation by the Smart Configurator.

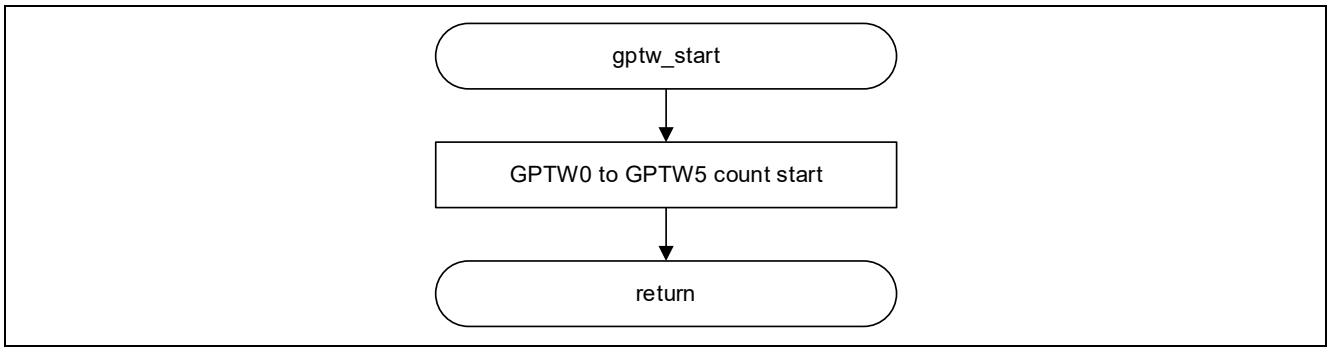


Figure 4.32 Count Start Function

4.4.5 Related Operations

4.4.5.1 When a Phase Difference is Set so Counter Initial Value > Compare Value

The following describes an example of phase shift start operation when the counter initial value is set to a value higher than the compare value.

- GPTW0: when counter initial value > GTCCRA and GTCCRB compare values
  Counting starts from 9C40h, which is greater than the compare value, and because neither GTCCRA nor GTCCRB compare match occur during up-counting, the waveform output does not change ((1) in figure below). During down counting, both GTCCRA and GTCCRB compare matches occur, so the GTIOC0A pin outputs low and the GTIOC0B pin outputs high ((2) in figure below).
- GPTW1: when counter initial value < GTCCRA and GTCCRB compare values
  Counting starts from 7D00h, which is less than the compare value, and because both GTCCRA and GTCCRB compare matches occur, the GTIOC0A pin outputs low and the GTIOC0B pin outputs high ((3) in figure below).

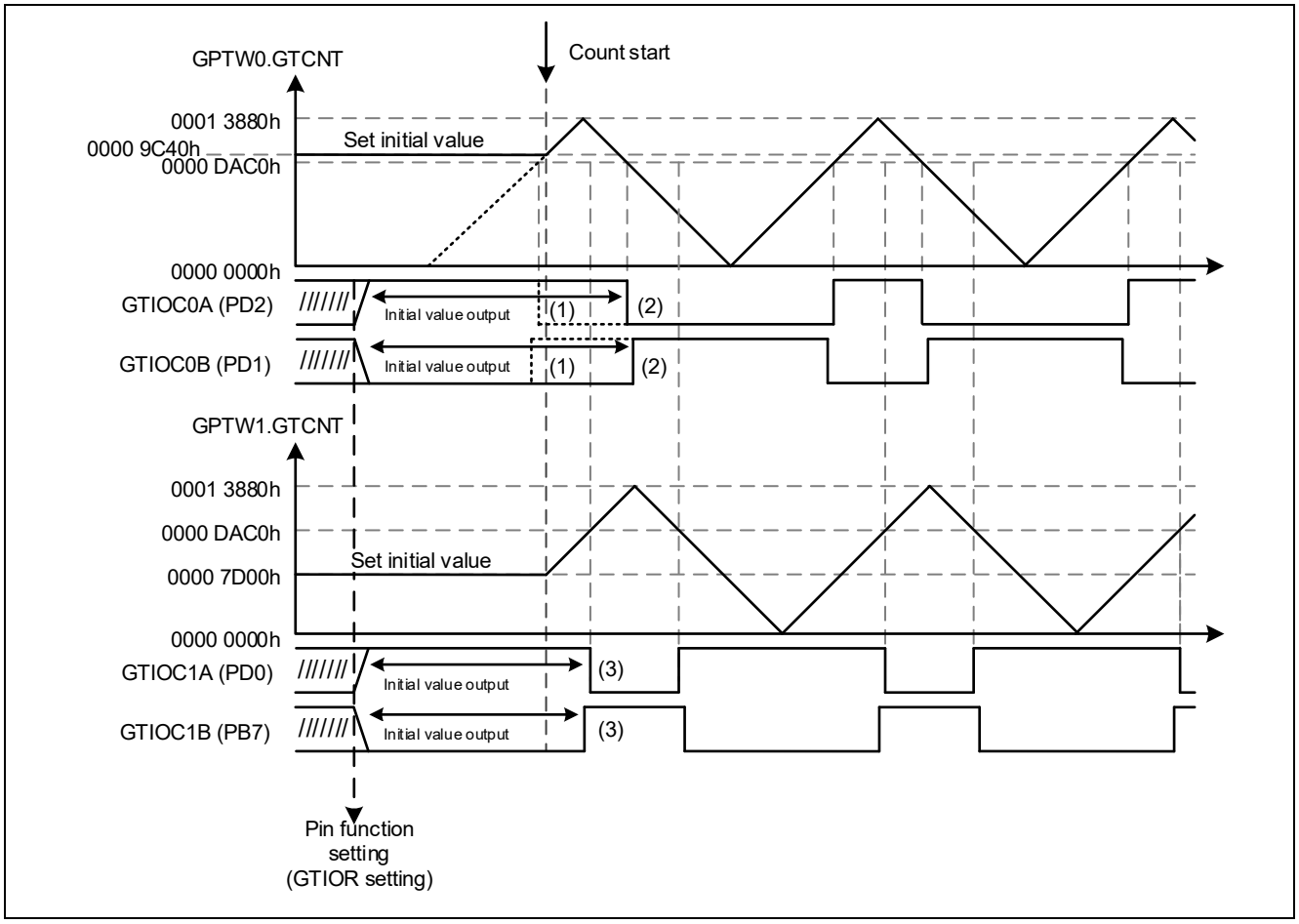


Figure 4.33 Operation Example of Phase Shift Start (GPTW0: Counter Initial Value > Compare Value)

4.4.6 Usage Notes

4.4.6.1 Counting Starts for Multiple Channels

In this sample code, the CSTRT0 to CSTRT5 bits of timer software start register GTSTR are set at the same time in the gptw_start function to start counting GPTW0 to GPTW5 at the same time.

When using the R_Config_GPTm_Start (m = 0 to 5) function generated by the Smart Configurator, the counting starts timing may not be the same because each of the functions are read.

For details, refer to RX66T Group User's Manual: Hardware, section 24.3.8.1 Synchronous Operation by Software.

4.4.6.2 Order of Priority in Events

In this sample code, synchronous start by software can be realized by setting multiple bits of the GTSTR register to 1b at the same time.

When a start/stop by a hardware source set in GTSSR and GTPSR conflicts with a CPU write (GTSTR write), the CPU write takes priority.

For details, refer to RX66T Group User's Manual: Hardware, section 24.10.5 Order of Priority in Events (2) The GTCR.CST Bit.

4.4.6.3 Setting a Value Greater than the Compare Value as the Counter Initial Value

In this sample code, the counter initial value for all channels is set to a value less than the first compare value.

If the counter initial value is set to a value greater than the first compare value, the first compare match may not occur, and the output waveform may appear inverted.

For details, refer to section 4.4.5.1 When a Phase Difference is Set so Counter Initial Value > Compare Value.

4.5 Synchronous Operation by Event Input from ELC

- Target sample code file name: r01an6282_rx66t_gptw_sawtooth_1st_elc_sync.zip

4.5.1 Overview

GPTW can perform synchronous operation (start, stop, clearing) using the ELC (event link controller) event input of a hardware source.

This section describes a sample code that repeatedly performs synchronous count start and synchronous stop/clearing for GPTW0 and GPTW1 (channels 0 and 1) according to the compare match interrupt of MTU0 (channel 0), using ELC event input of a hardware source.

The following list provides the MTU, GPTW and ELC settings used in the sample code.

- MTU0 (channel 0)
  - Use PWM mode 1
  - Initial output value = low
  - Carrier period = 400µs
  - Timer counter clock = 160MHz (PCLKC)
  - Use TGRA as duty register
    - High output at TGRA compare match
  - Use TGRB as period register
    - Timer counter clear source = TGRB compare match
    - Low output at TGRB compare match
- GPTW0 and GPTW1 (channels 0 and 1)
  - Use sawtooth-wave one-shot pulse mode
  - Low output at counting starts, low output at counting stops
  - Retain output at cycle end
  - Carrier period = 200µs
  - Timer counter clock = 160MHz (PCLKC)
  - Use GTPR as period register
    - Count direction = up-counting
    - Counter initial value = 0
  - Use GPTWn.GTCCRA as duty register (n = 0, 1)
    - Use GTIOCnA pin as PWM output pin
    - Toggle output at GPTWn.GTCCRA compare match
  - Use GPTWn.GTCCRB as duty register (n = 0, 1)
    - Use GTIOCnB pin as PWM output pin
    - Toggle output at GPTWn.GTCCRB compare match
  - Use double buffer registers
    - GTCCRC and GTCCRD operate as buffer registers of GTCCRA
    - GTCCRE and GTCCRF operate as buffer registers of GTCCRB
  - Enable sources:
    Count start source = ELCA event input
    Count stop source = ELCB event input
    Counter clear source = ELCB event input
- ELC
  - Select MTU0 compare match 0A as ELC event
  - Select MTU0 compare match 0B as ELC event
  - Select GPTW event source A as destination resource
  - Select GPTW event source B as destination resource

Set in Smart Configurator.
For Setting Methods,
refer to section 4.5.3.

The structure of this sample code is shown below.

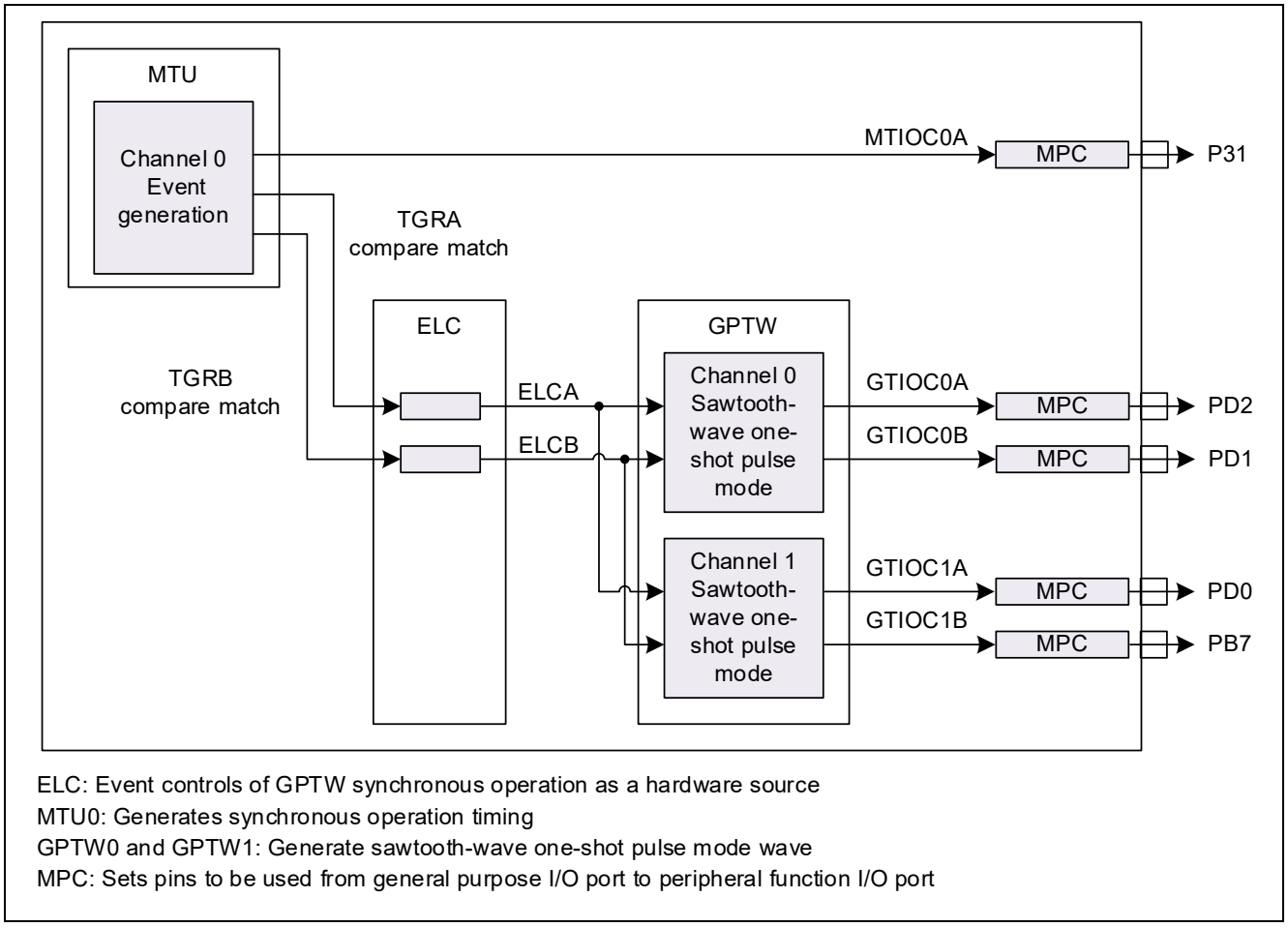


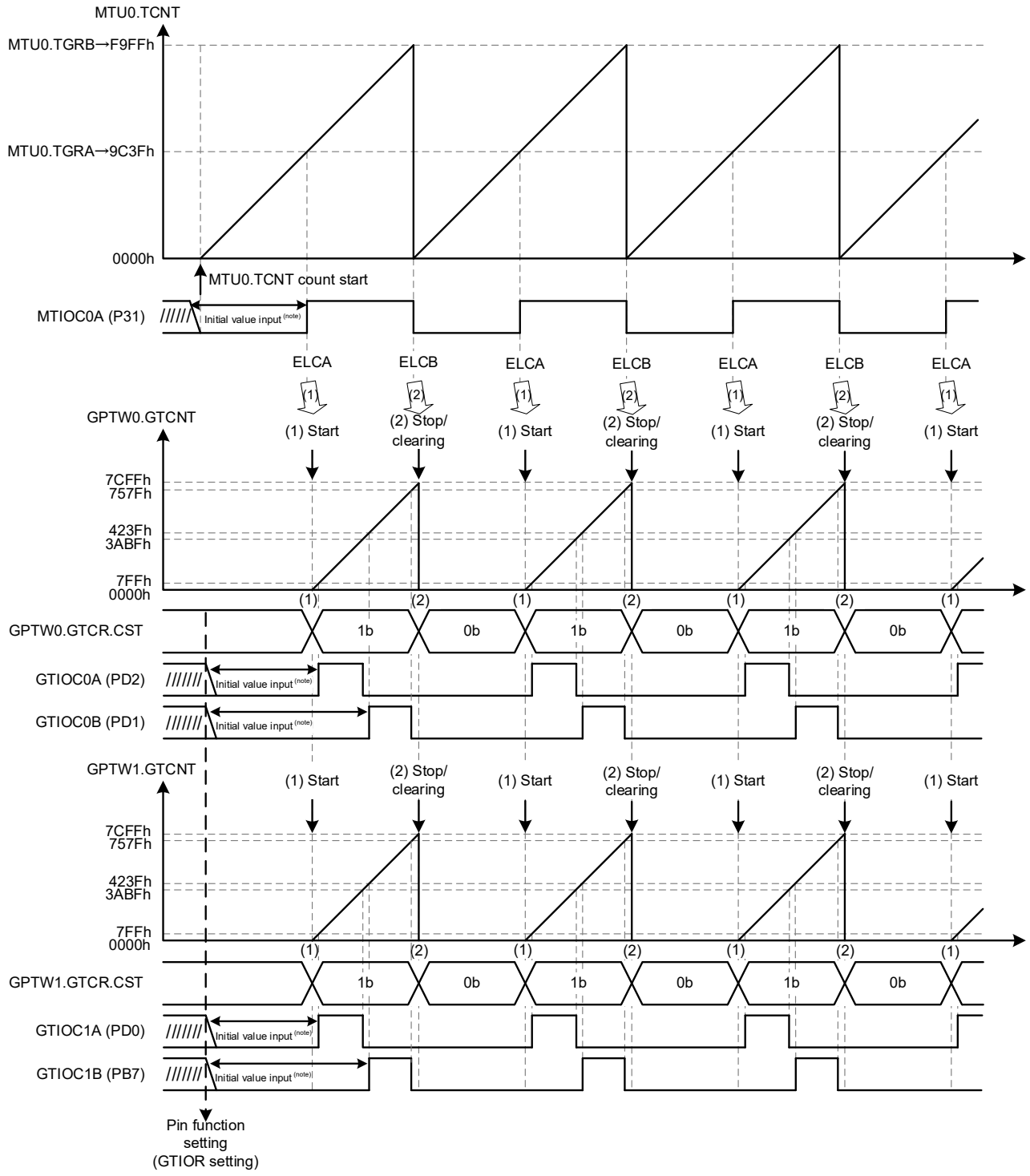
Figure 4.34 Sample Code Structure

4.5.2 Operation Details

The sample code operations are shown in Figure 4.35. The event sources of ELC are set as follows: the MTU0's TGRA compare match is set as the ELCA event and TGRB compare match as the ELCB event.

- Synchronous start
  GPTW0 and GPTW1 synchronous count starts by ELCA event input when the MTU0.TGRA compare match occurs ((1) in Figure 4.35).
- Synchronous stop/clearing
  GPTW0 and GPTW1 synchronous count is stopped/cleared by ELCB event input when the MTU0.TGRB compare match occurs ((2) in Figure 4.35).

For details on sawtooth-wave one-shot pulse mode, refer to RX Family PWM Output Methods Using MTU3/GPTW Application Note, section 4.5.2 Operation Details.



Note: The sample code waveform starts outputting the initial values when the PMR register is set.

Figure 4.35 Sample Code Operations

4.5.3 Smart Configurator Settings

The sample codes use the Smart Configurator to add the MTU and GPTW as described below. For details on how to add MTU components, refer to section 3.1.4 Adding Components, and for details on how to add GPTW components, refer to section 4.1.4 Adding Components.

Table 4.9 Adding Components (MTU0)

Item	Description
Component	PWM Mode Timer
Configuration name	Config_MTU0
Operation	PWM Mode 1
Resource	MTU0

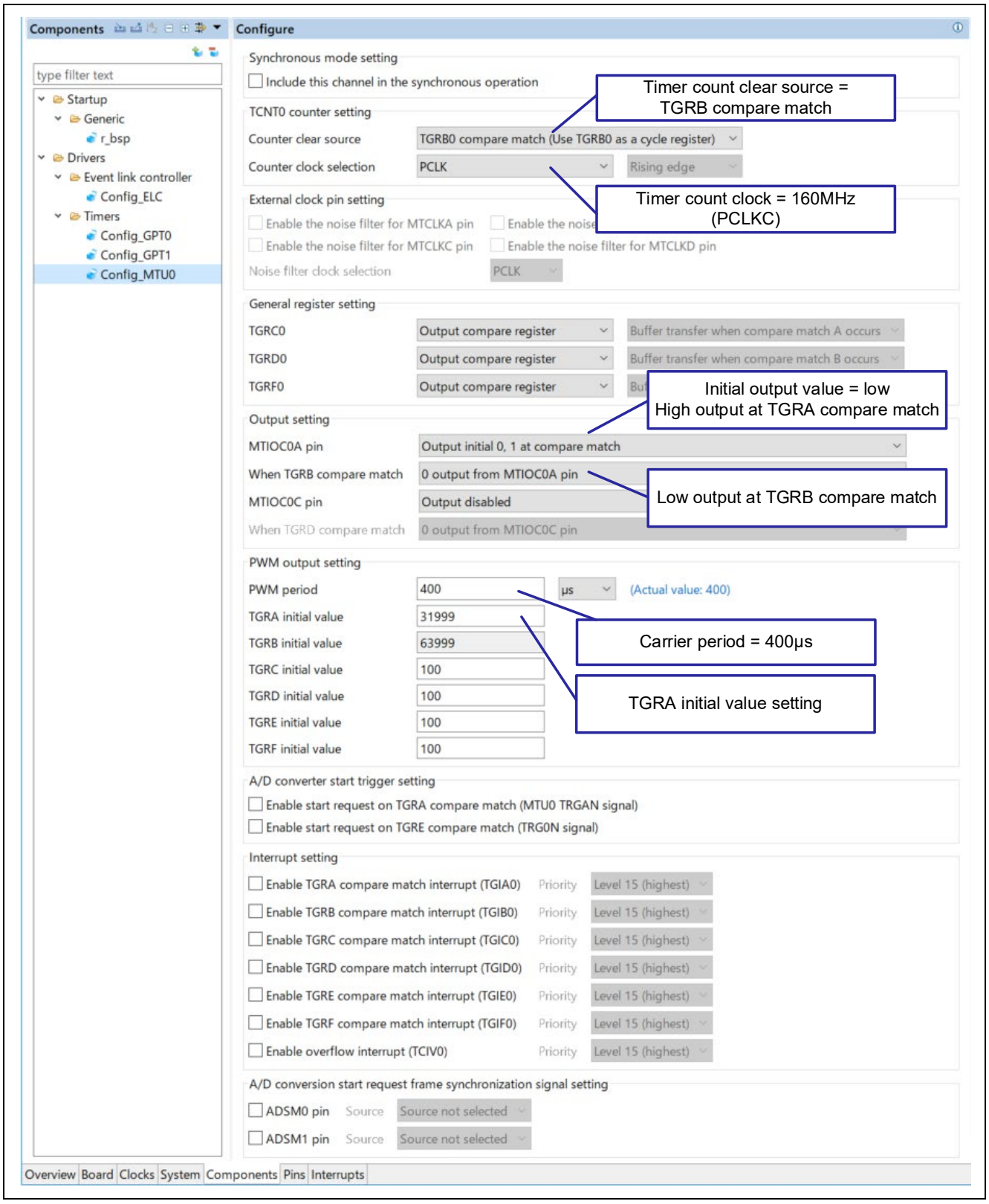


Figure 4.36 MTU0 Settings

Table 4.10 Adding Components (GPTW0 and GPTW1)

Item	Description	
Component	General PWM Timer	
Configuration name	Config_GPT0	Config_GPT1
Work mode	Sawtooth-wave One-Shot Pulse Mode	
Resource	GPT0	GPT1

Figure 4.37 to Figure 4.40 show the Config_GPT0 settings. The settings for GPT1 are basically the same.

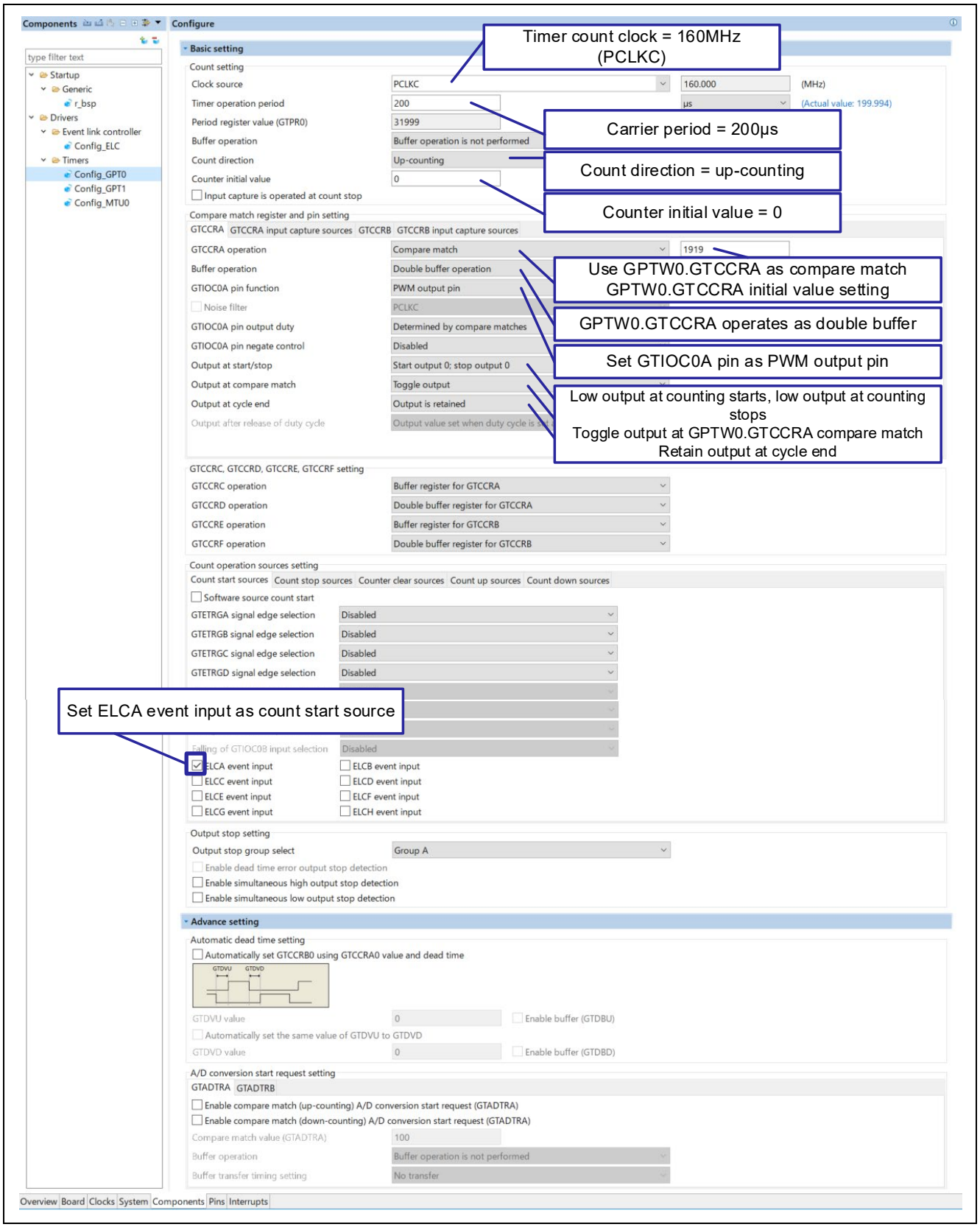


Figure 4.37 GPT0 Settings (1/4)

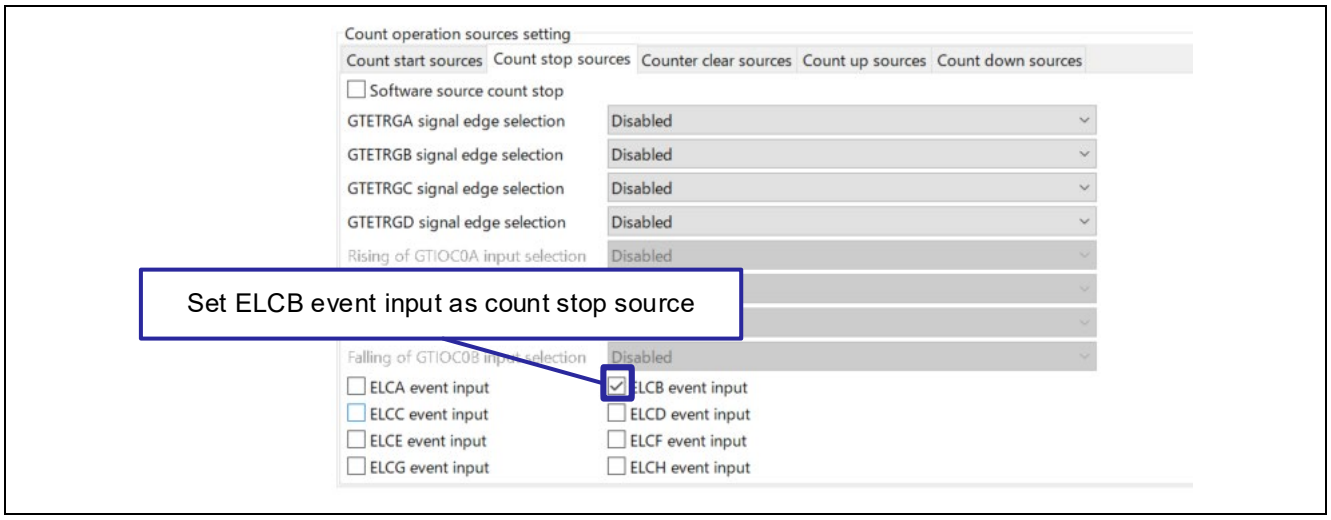


Figure 4.38 GPT0 Settings (2/4)

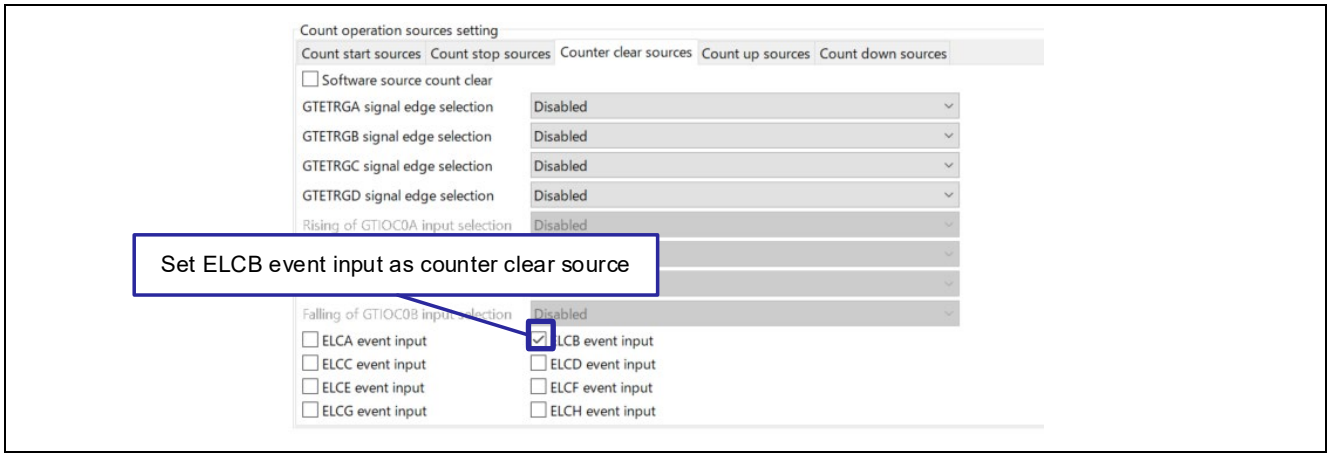


Figure 4.39 GPT0 Settings (3/4)

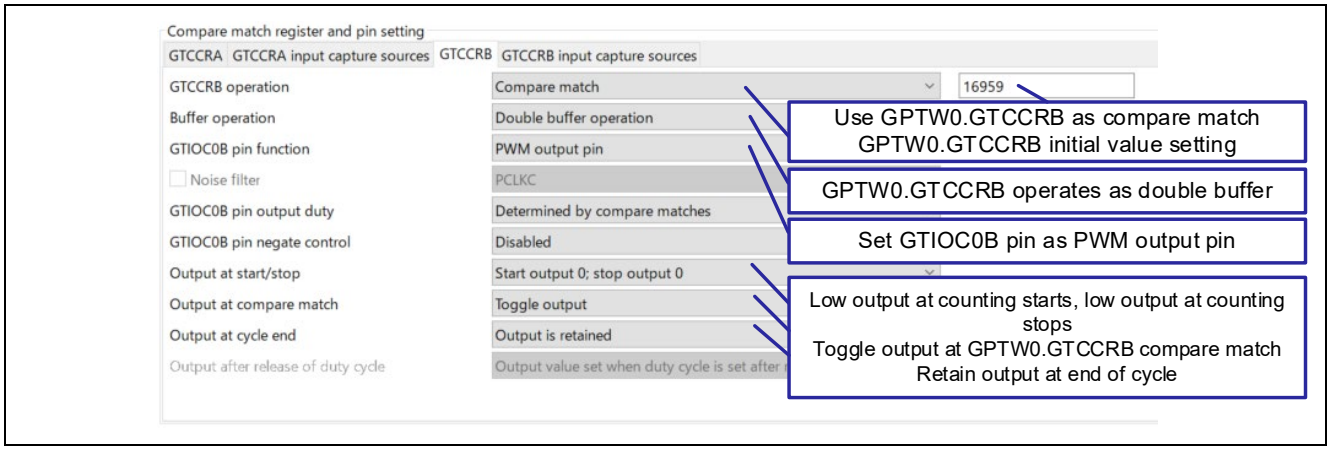
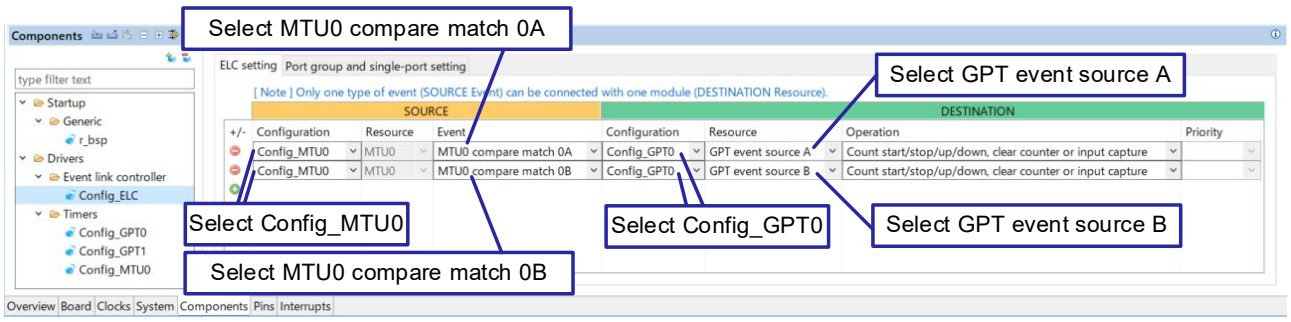


Figure 4.40 GPT0 Settings (4/4)

ELC component settings are as follows.

Table 4.11 Adding Components (ELC)

Item	Description
Component	Event Link Controller
Configuration name	Config_ELC1
Resource	ELC



Note: GPTW event sources are common to all channels, so both GPTW0 and GPTW1 channels can be used.

Figure 4.41 ELC Settings

4.5.4 Flowcharts

The following flowchart shows the main function processing of a function added after code generation by the Smart Configurator.

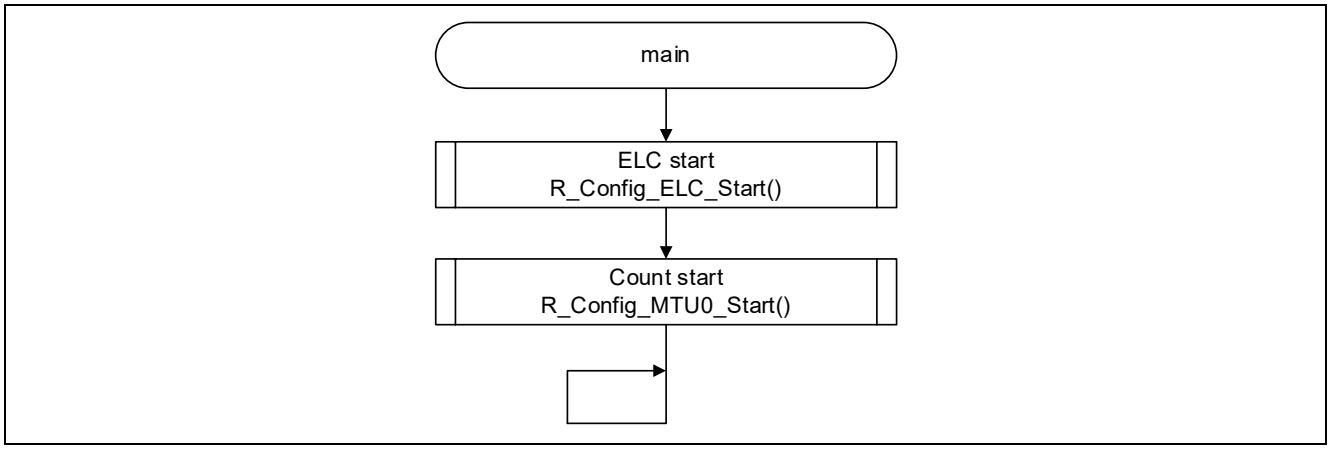


Figure 4.42 main Function

The user initialization function R_Config_GPT0_Create_UserInit, which is executed before the main function, sets the values of the buffer registers. In order to set the second compare match register value in the first cycle, a forced buffer transfer is performed after setting the buffer register values, and then the temporary register and compare register values are set. This function is called from within the R_Config_GPT0_Create function.

R_Config_GPT1_Create_UserInit also performs the same processes.

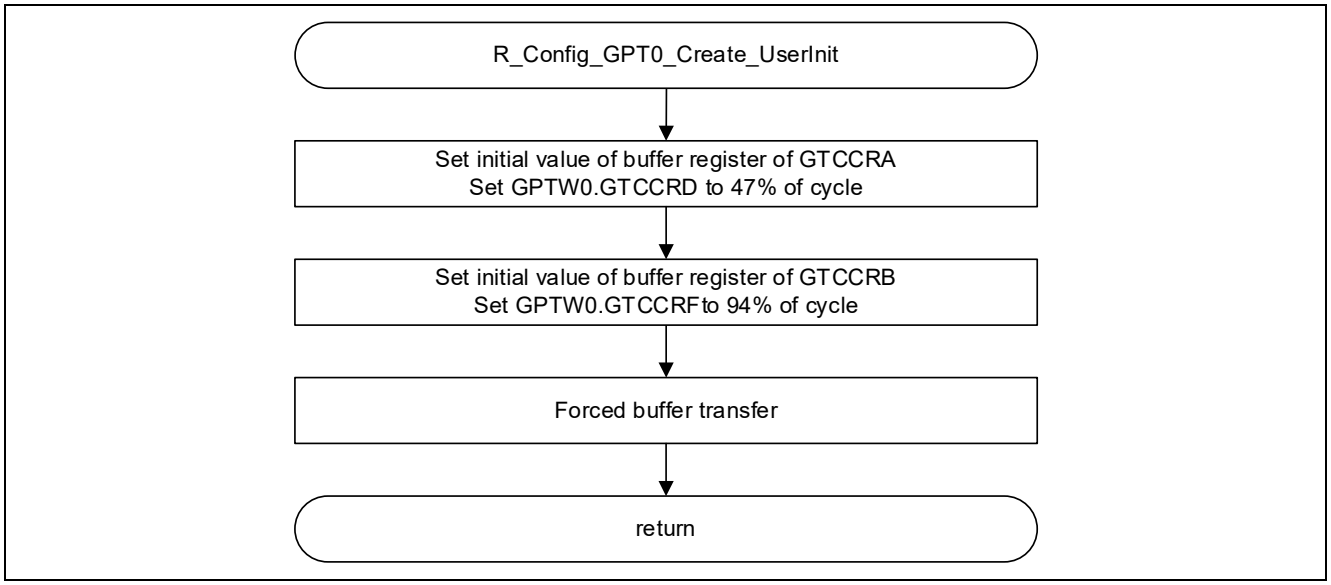


Figure 4.43 User Initialization Function

4.5.5 Related Operations

4.5.5.1 MTU and GPTW Synchronous Count Start using ELC Event Input

Figure 4.44 shows the synchronous count start operation for GPTW0 and GPTW1 when counting starts in MTU0.

- Synchronous start
  Synchronous count starts for GPTW0 and GPTW1 at the MUT0.TGRA compare match (TCNT count value: 0000h) by ELCA event input ((1) in Figure 4.44).

Note that in synchronous operation using ELC, the operation timing differs for the event generation module (MTU0 in the sample code) and the modules that receive generated events and perform interlinked operations (GPTW0 and GPTW1 in the sample code).

For details, refer to section 4.5.6.3.

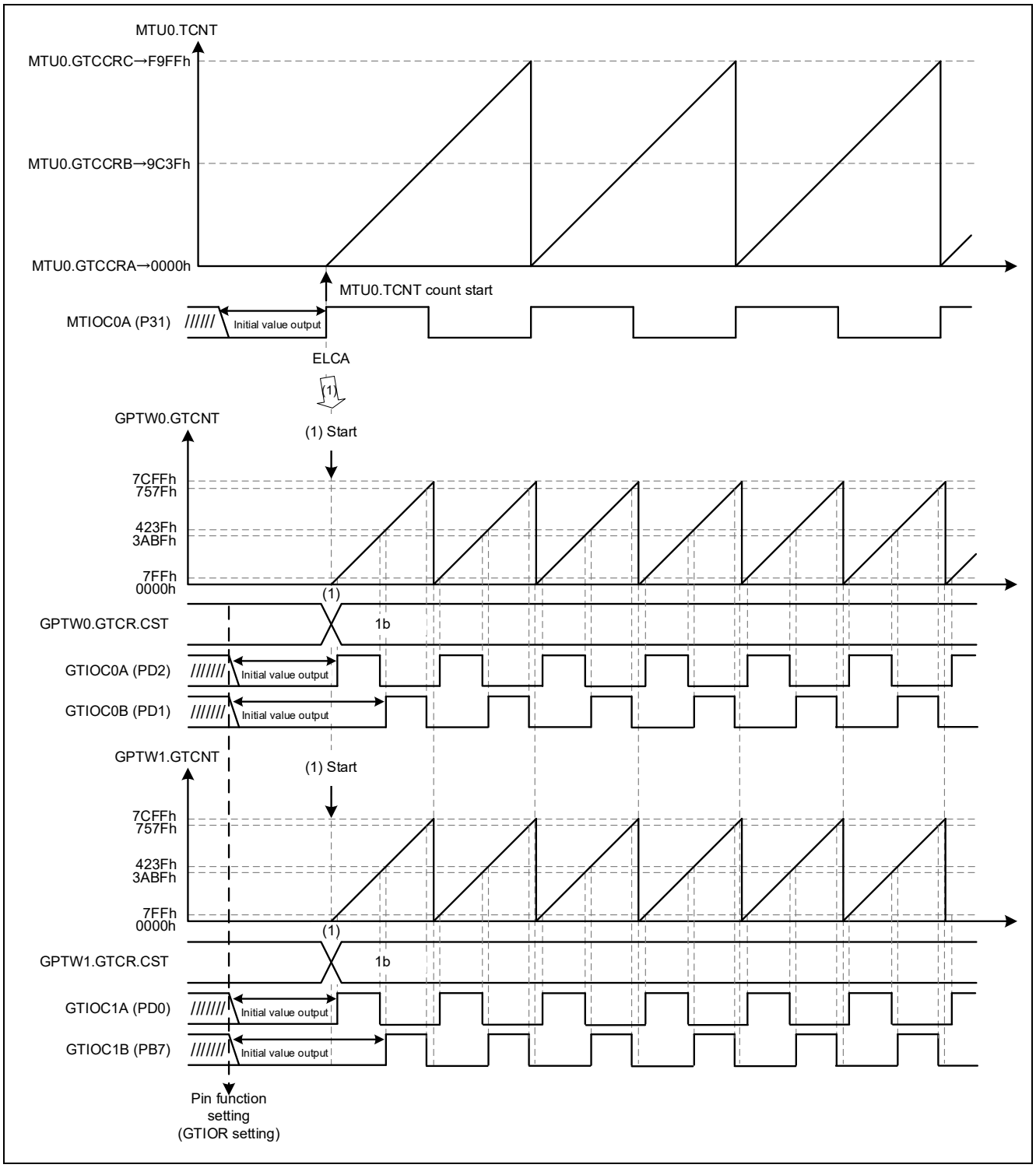


Figure 4.44 Example of MTU and GPTW Synchronous Operation by ELC Event Input

4.5.6 Usage Notes

4.5.6.1 Order of Priority in Events

In this sample code, synchronous start/stop can be realized by using ELC event input of a hardware source.

When a start/stop by a hardware source set in GTSSR and GTPSR conflicts with a CPU write (GTSTR write/GTSTP write), the CPU write takes priority.

For details, refer to RX66T Group User's Manual: Hardware, section 24.10.5 Order of Priority in Events (2) The GTCR.CST Bit.

4.5.6.2 GTCNT Counter Start/Stop

In synchronous count start/stop operation using a hardware source, when a start source event input occurs, the GTCR.CST bit is set to 1b, and when a stop source event input occurs, the GTCR.CST bit is set to 0b.

Since the GTCNT counter starts/stops after the count clock is selected by TPCS[3:0] bits following the GTCR.CST bit update, events are ignored until the GTCNT counter actually starts, and events may be accepted, or interrupts may be generated after the CST bit is set to 0b.

For details, refer to RX66T Group User's Manual: Hardware, section 24.10.4 The GTCNT Counter Start/Stop.

4.5.6.3 Timing of Hardware Count Start, Stop, and Clear Operations

The start timing of synchronous operation differs according to the hardware source and clock used.

For details, refer to RX66T Group User's Manual: Hardware, section 24.3.7 Hardware Count Start/Count Stop and Clear Operation.

4.5.6.4 Operation of GPTW When Event Signal is Input

Eight event signals specified by the ELSR48 to ELSR55 registers are connected to every channel of GPTW as GPTW event sources A to H.

For details, refer to RX66T Group User's Manual: Hardware, section 19.3.4 Operation of GPTW When Event Signal is Input.

4.6 Synchronous Operation by External Trigger Input

- Target sample code file name: r01an6282_rx66t_gptw_sawtooth_1st_trigger_sync.zip

4.6.1 Overview

GPTW can perform synchronous operation (start, stop, clearing) using an external trigger input of a hardware source.

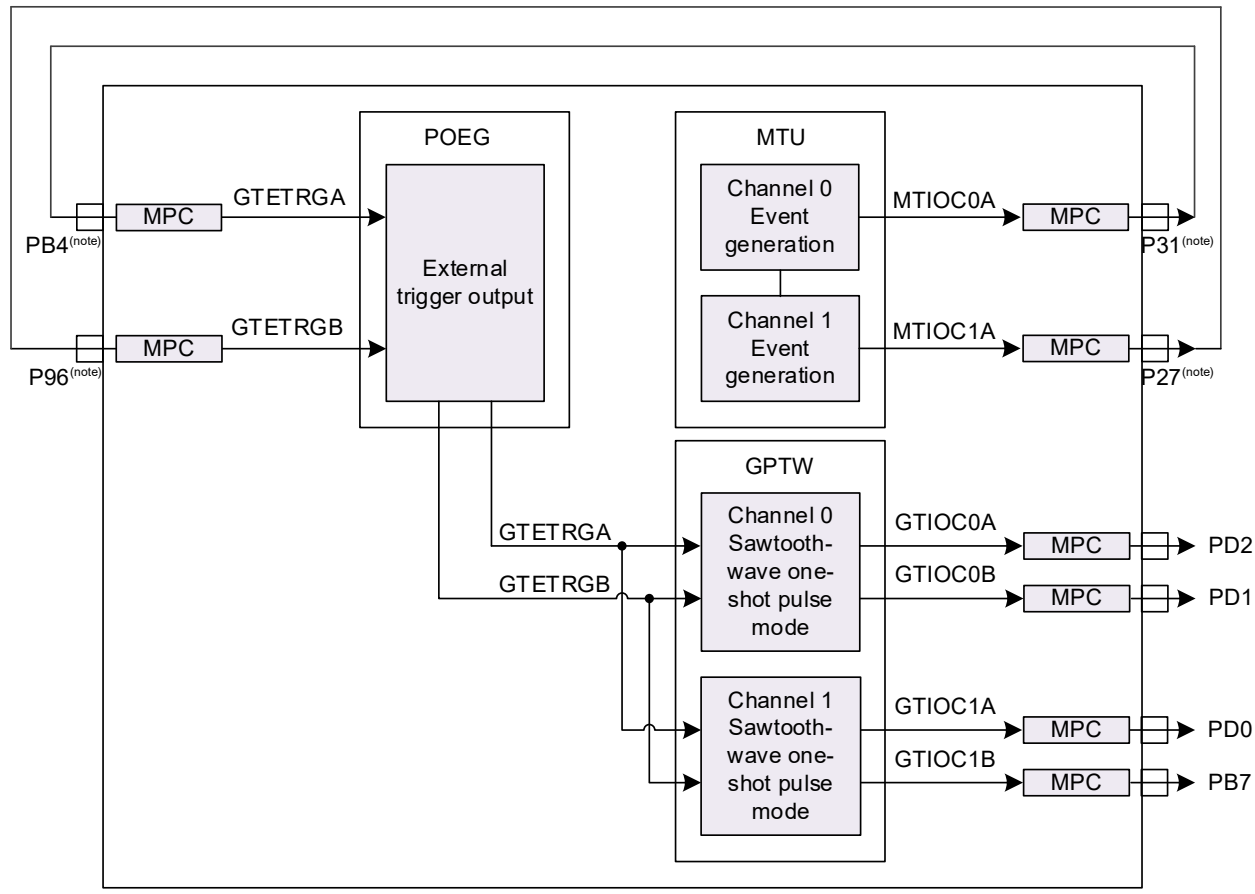
This section describes a sample code that repeatedly performs synchronous count start and synchronous stop/clearing for GPTW0 and GPTW1 (channels 0 and 1) using external trigger input of a hardware source.

The following list provides the MTU, GPTW and POEG settings used in the sample code.

- MTU0 and MTU1 (channels 0 and 1)
  - Use normal mode timer
  - Set to synchronous operation
  - Carrier period = 200µs
  - Timer counter clock = 160MHz (PCLKC)
  - Use MTU0.TGRA as period register
    - Timer counter clear source = MTU0.TGRA compare match
  - MTIOCnA pin initial output is low,
    Toggle output at MTUn.TGRA compare match (n = 0, 1)
- GPTW0 and GPTW1 (channels 0 and 1)
  - Use sawtooth-wave one-shot pulse mode
  - Low output at counting starts, low output at counting stops
  - Retain output at cycle end
  - Carrier period = 200µs
  - Timer counter clock = 160MHz (PCLKC)
  - Use GTPR as period register
    - Count direction = up-counting
    - Counter initial value = 0
  - Use GPTWn.GTCCRA as duty register (n = 0, 1)
    - Use GTIOCnA pin as PWM output pin
    - Toggle output at GPTWn.GTCCRA compare match
  - Use GPTWn.GTCCRB as duty register (n = 0, 1)
    - Use GTIOCnB pin as PWM output pin
    - Toggle output at GPTWn.GTCCRB compare match
  - Set sources:
    Count start source = GTETRGA pin input rising edge detection
    Count stop source = GTETRGB pin input rising edge detection
    Counter clear source = GTETRGB pin input rising edge detection
- POEG
  - Enable GTETRGA pin settings
  - Enable GTETRGB pin settings

Set in Smart Configurator.
For Setting Methods,
refer to section 4.6.3.

The structure of this sample code is shown below.



POEG: Outputs GTETRGA and GTETRGB pins as external trigger to GPTW
MTU0: Generates synchronous operation timing
GPTW0 and GPTW1: Generate sawtooth-wave one-shot pulse mode waveform
MPC: Sets pins to be used from general purpose I/O port to peripheral function I/O port

Note: Make connections on board.

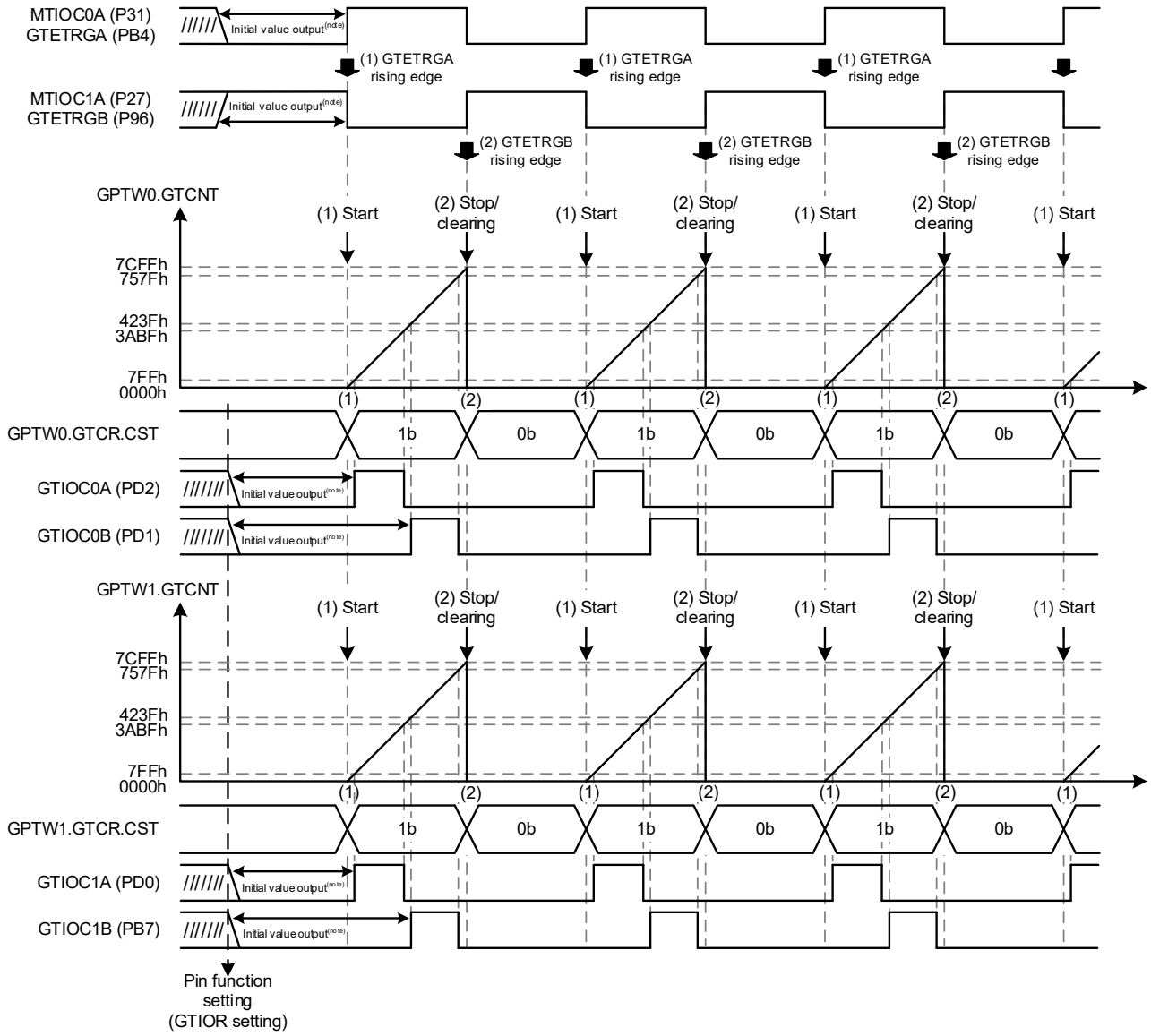
Figure 4.45 Sample Code Structure

4.6.2 Operation Details

The sample code operations are shown below. Output pin MTIOC0A (P31) is connected to external trigger input pin GTETRGA (PB4) and output pin MTIOC1A (P27) is connected to external trigger input pin GTETRGB (P96) on the board.

- Synchronous start
  GPTW0 and GPTW1 synchronous count starts by detecting a rising edge of external trigger input pin GTETRGA ((1) in figure below).
- Synchronous stop/clearing
  GPTW0 and GPTW1 synchronous count is stopped/cleared by detecting a rising edge of external trigger input pin GTETRGB ((2) in figure below).

For details on sawtooth-wave one-shot pulse mode, refer to RX Family PWM Output Methods Using MTU3/GPTW Application Note, section 4.5.2 Operation Details.



Note: The sample code waveform starts outputting the initial values when the PMR register is set.

Figure 4.46 Sample Code Operations (Start at GTETRGA Pin Input Rising Edge, Stop/Clearing at GTETRGB Pin Input Rising Edge)

4.6.3 Smart Configurator Settings

The sample codes use the Smart Configurator to add the MTU and GPTW as described below. For details on how to add MTU components, refer to section 3.1.4 Adding Components, and for details on how to add GPTW components, refer to section 4.1.4 Adding Components.

Table 4.12 Adding Components (MTU0 and MTU1)

Item	Description	
Component	Normal Mode Timer	
Configuration name	Config_MTU0	Config_MTU1
Input capture/ Output compare pins	2 pins	
Resource	MTU0	MTU1

The following figure shows the Config_MTU0 settings. The settings for MTU1 are basically the same. As pin output levels differ, I/O pin settings also differ.

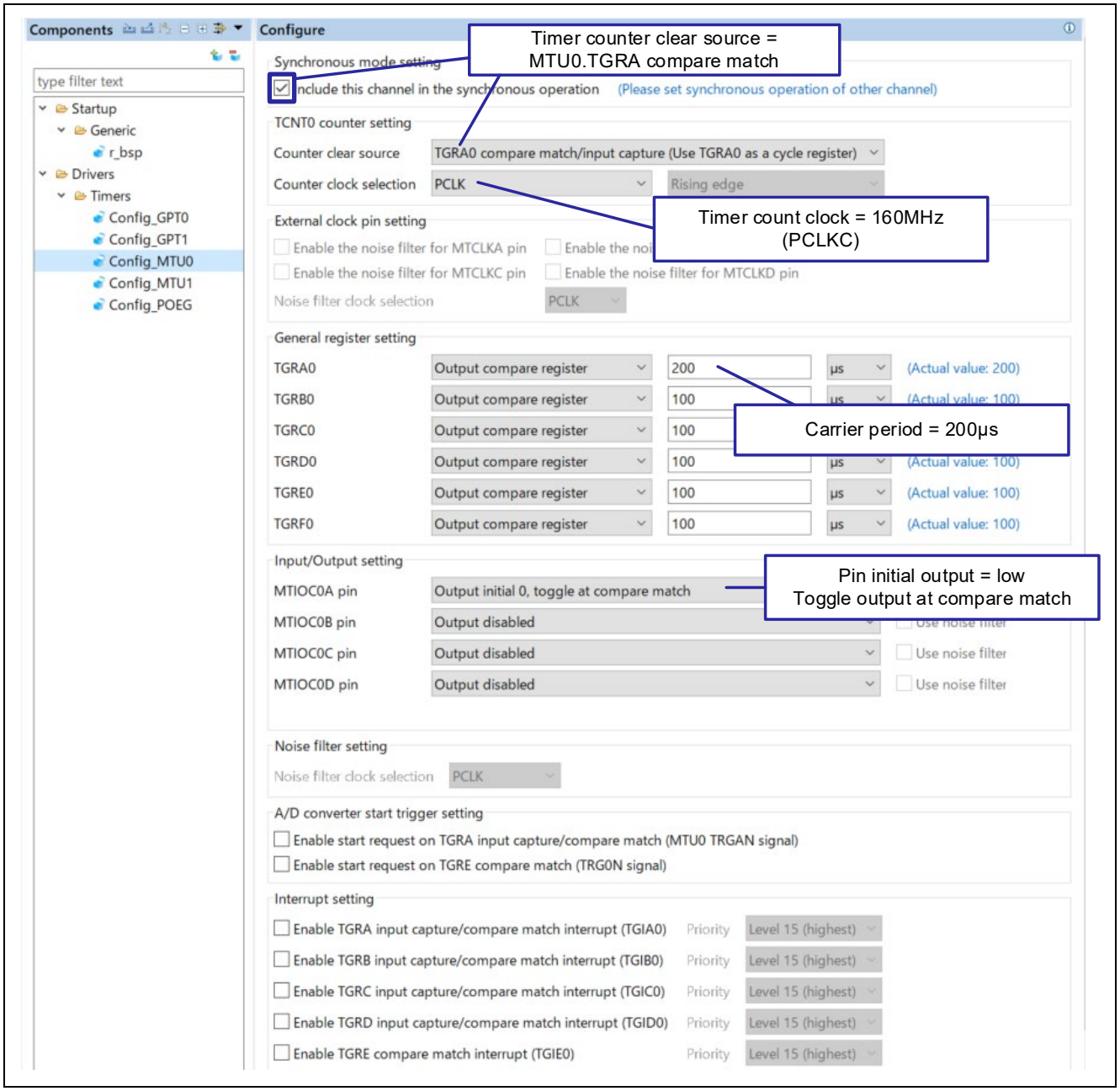


Figure 4.47 MTU0 Settings

Table 4.13 Adding Components (GPTW0 and GPTW1)

Item	Description	
Component	General PWM Timer	
Configuration name	Config_GPT0	Config_GPT1
Work mode	Sawtooth-wave One-shot Pulse Mode	
Resource	GPT0	GPT1

Figure 4.48 to Figure 4.51 show the Config_GPT0 settings. The settings for GPT1 are basically the same.

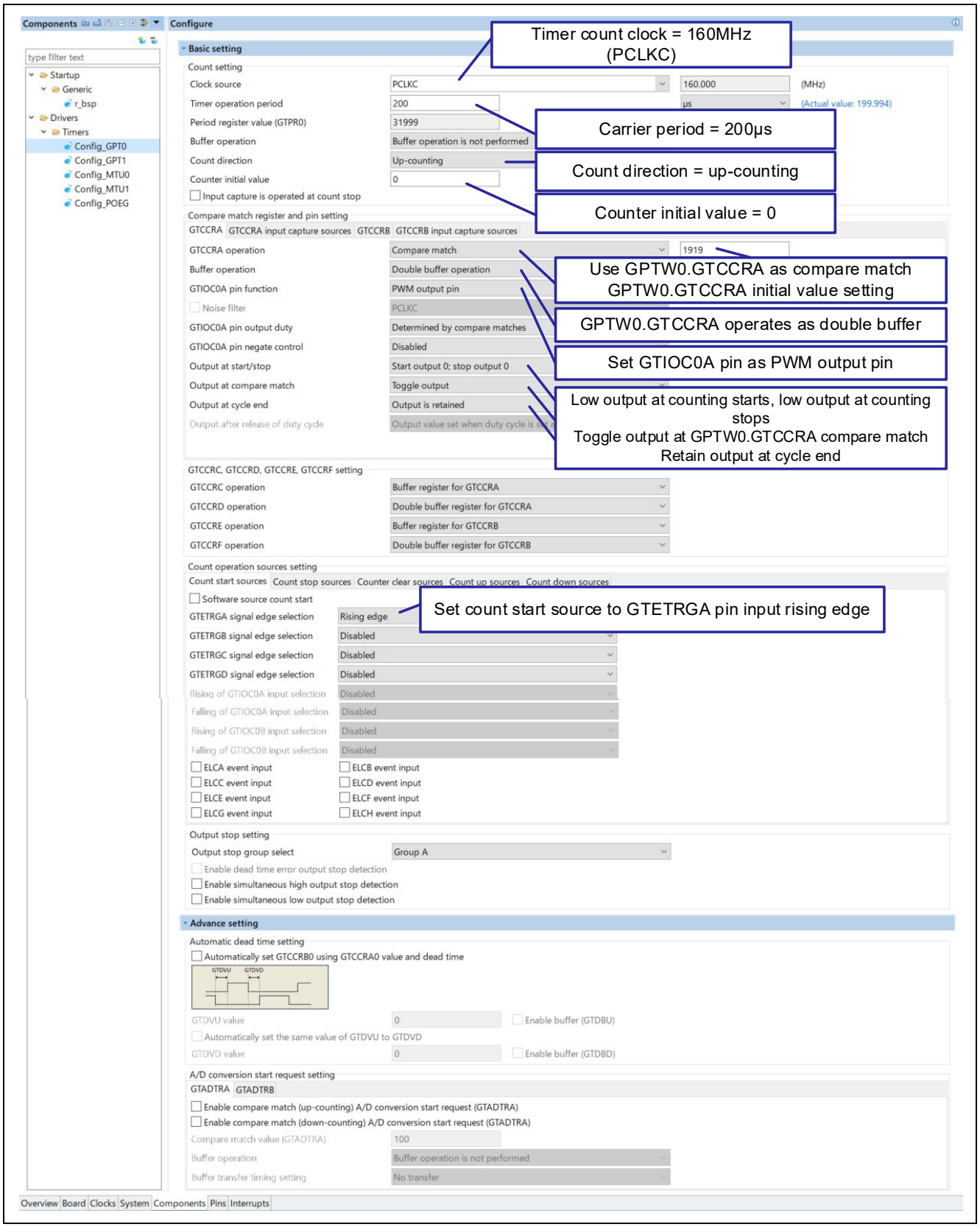


Figure 4.48 GPT0 Settings (1/4)

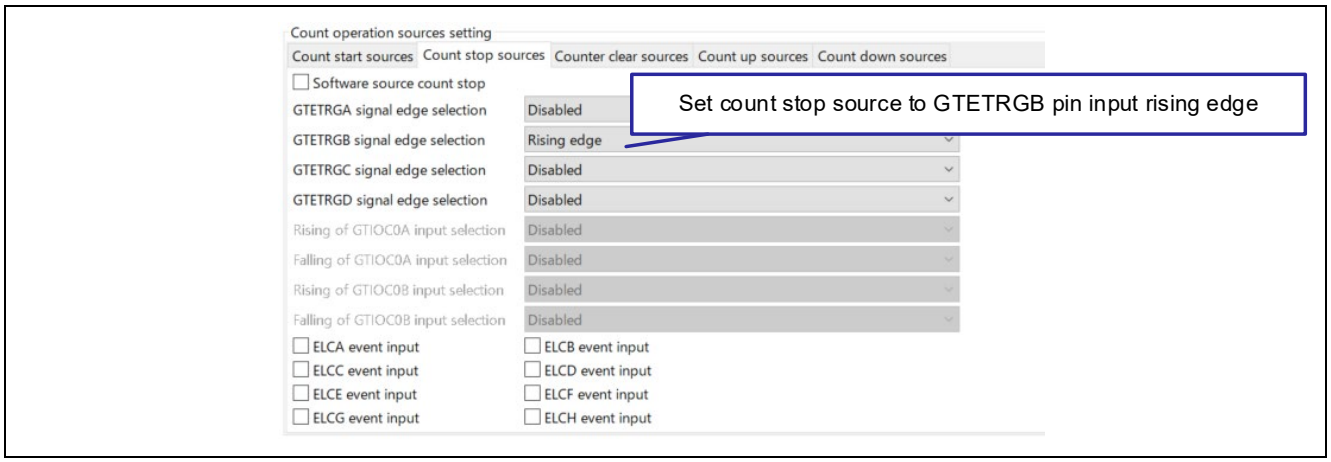


Figure 4.49 GPT0 Settings (2/4)

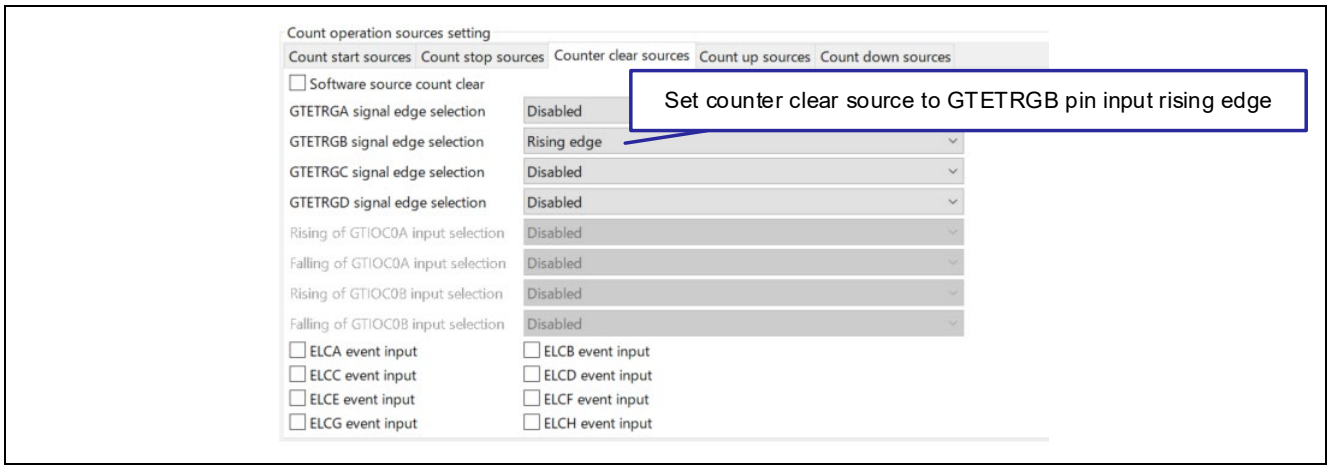


Figure 4.50 GPT0 Settings (3/4)

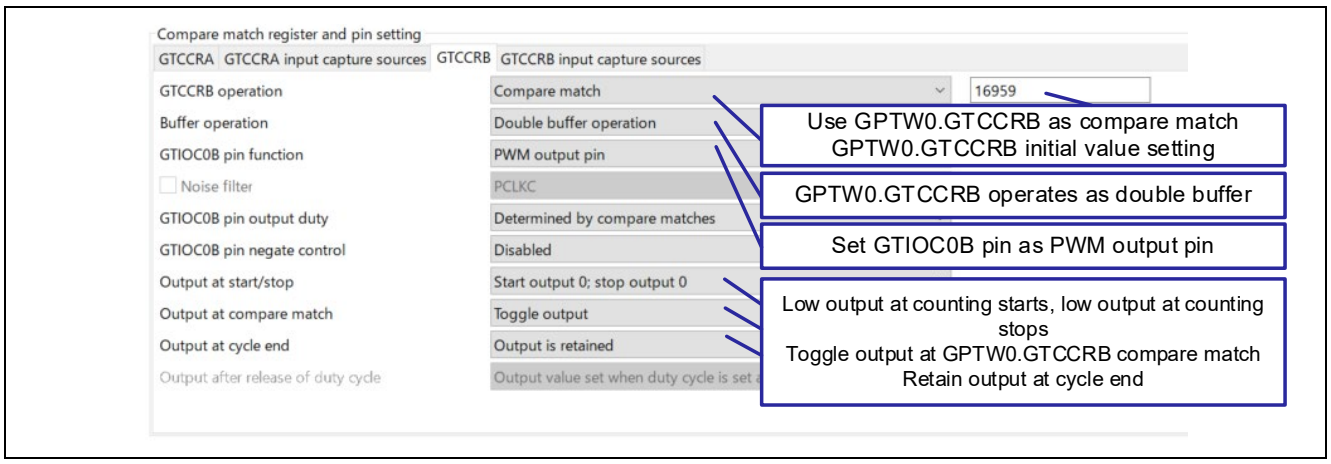


Figure 4.51 GPT0 Settings (4/4)

To use the external trigger input pin, add the POEG component as indicated below.

Table 4.14 Adding Components (POEG)

Item	Description
Component	Port output enable
Configuration name	Config_POEG
Resource	POEG

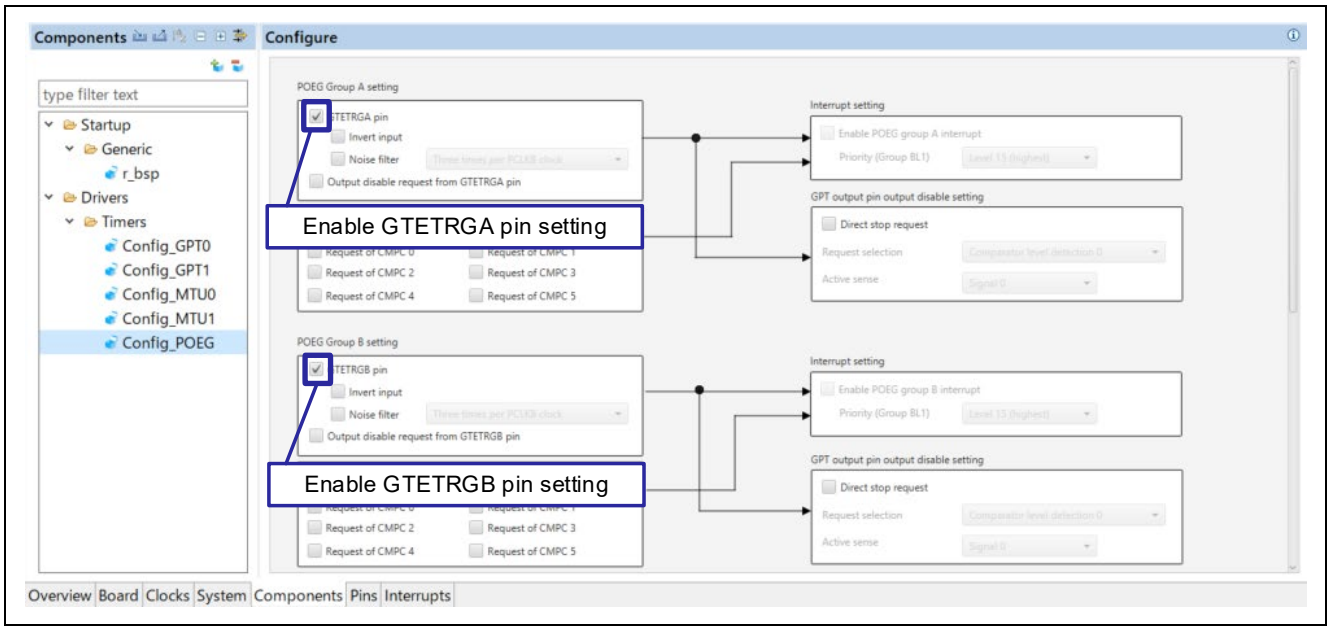


Figure 4.52 POEG Settings

4.6.4 Flowcharts

The following flowchart shows the main function processing of a function added after code generation by the Smart Configurator.

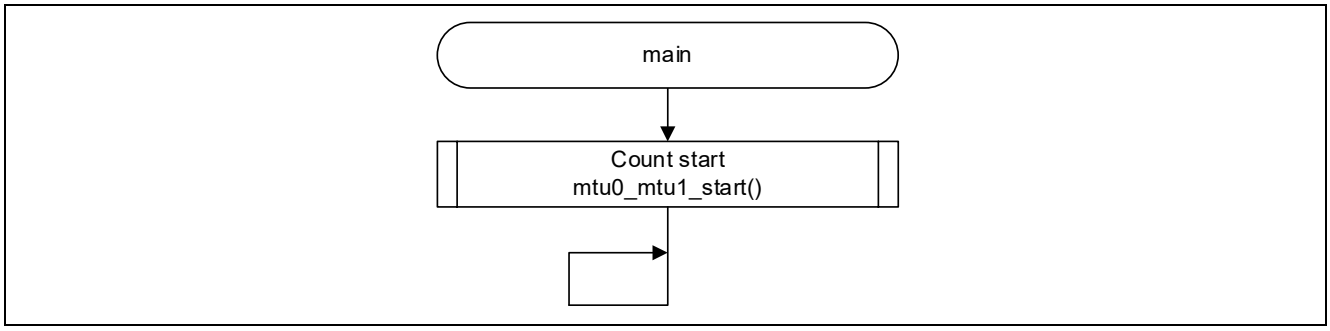


Figure 4.53 main Function

The MTU0 and MTU1 counting is started in the count start function.

This function is newly created after code generation by the Smart Configurator.

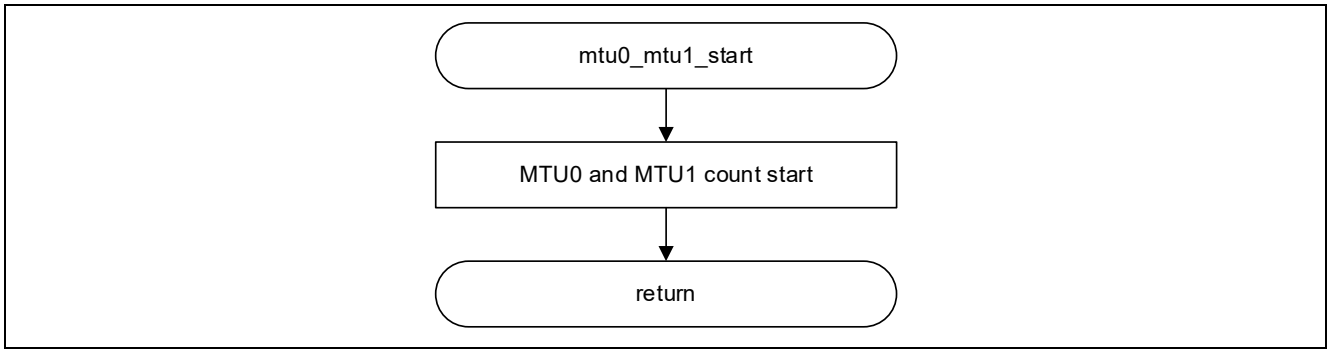


Figure 4.54 Count Start Function

The user initialization function R_Config_GPT0_Create_UserInit, which is executed before the main function, sets the values of the buffer registers. In order to set the second compare match register value in the first cycle, a forced buffer transfer is performed after setting the buffer register values, and then the temporary register and compare register values are set. This function is called from within the R_Config_GPT0_Create function.

R_Config_GPT1_Create_UserInit also performs the same processes.

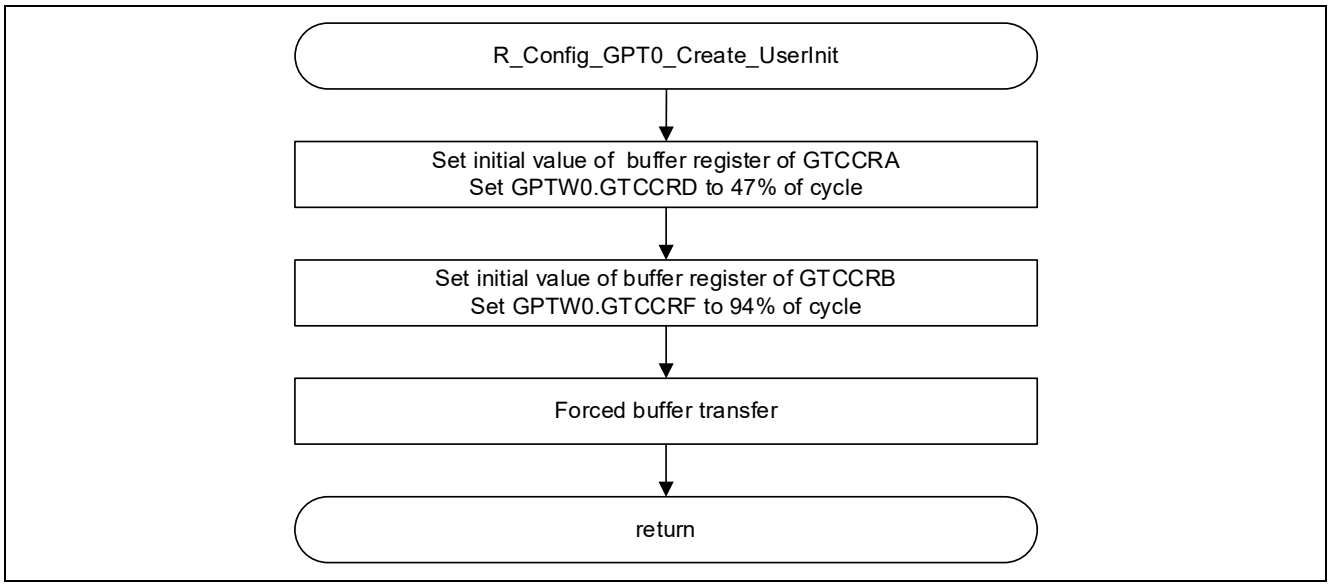


Figure 4.55 User Initialization Function

4.6.5 Related Operations

4.6.5.1 Assigning Start and Stop/Clearing to One External Input Trigger

This sample code describes an operation which performs counter synchronous start and stop/clearing of GPTW0 and GPTW1 by detecting the rising/falling edge of one external trigger input GTETRGA.

Change the count stop source and counter clear source settings for GPTW0 and GPTW1 in the Smart Configurator as follows.

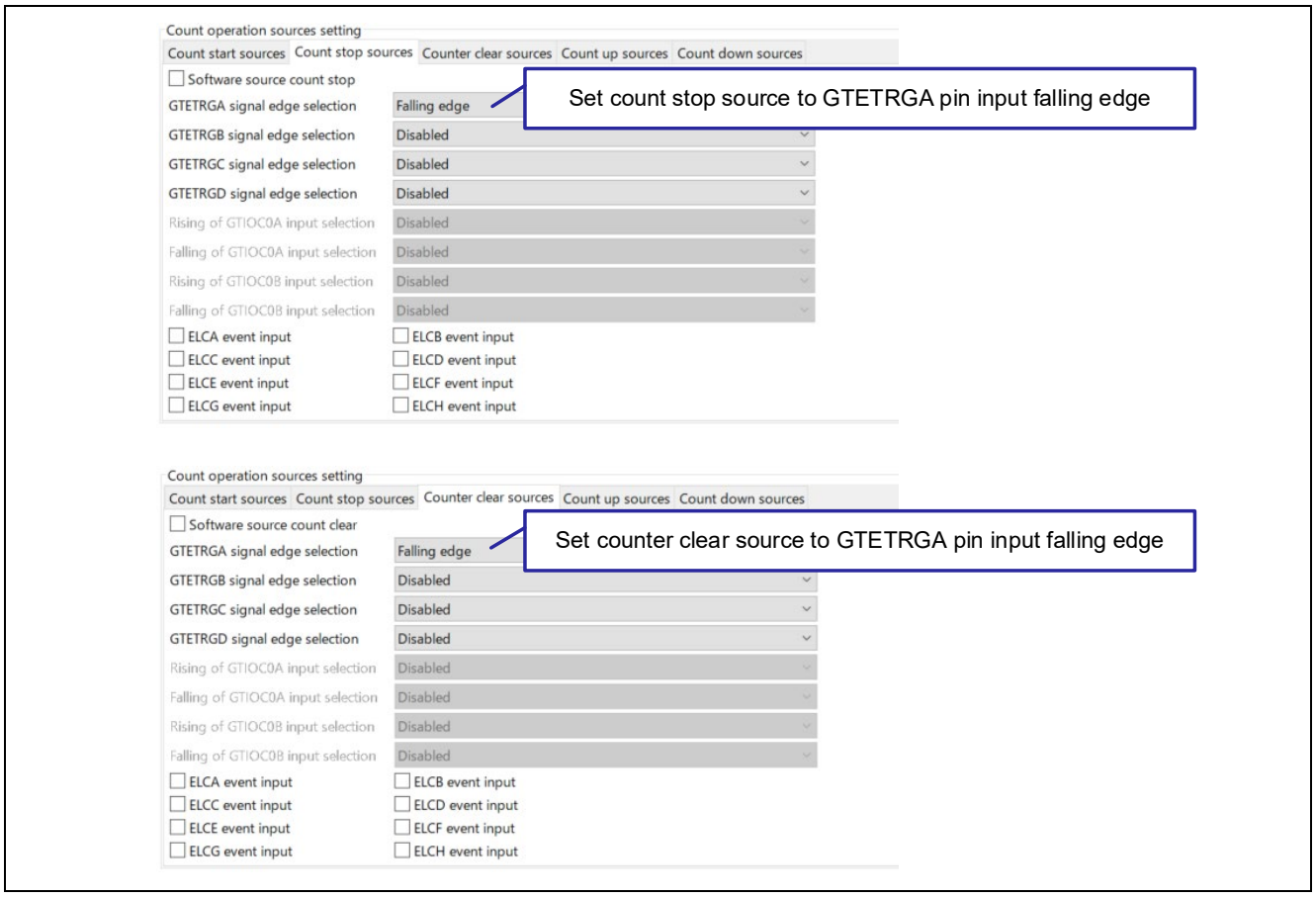


Figure 4.56 GPTW0 and GPTW1 Settings

After changing the settings, the operation will be as shown in Figure 4.57.

- Synchronous start
  GPTW0 and GPTW1 synchronous count starts by detecting a rising edge of external trigger input pin GTETRGA ((1) in Figure 4.57).
- Synchronous stop/clearing
  GPTW0 and GPTW1 synchronous count is stopped/cleared by detecting a falling edge of external trigger input pin GTETRGB ((2) in Figure 4.57).

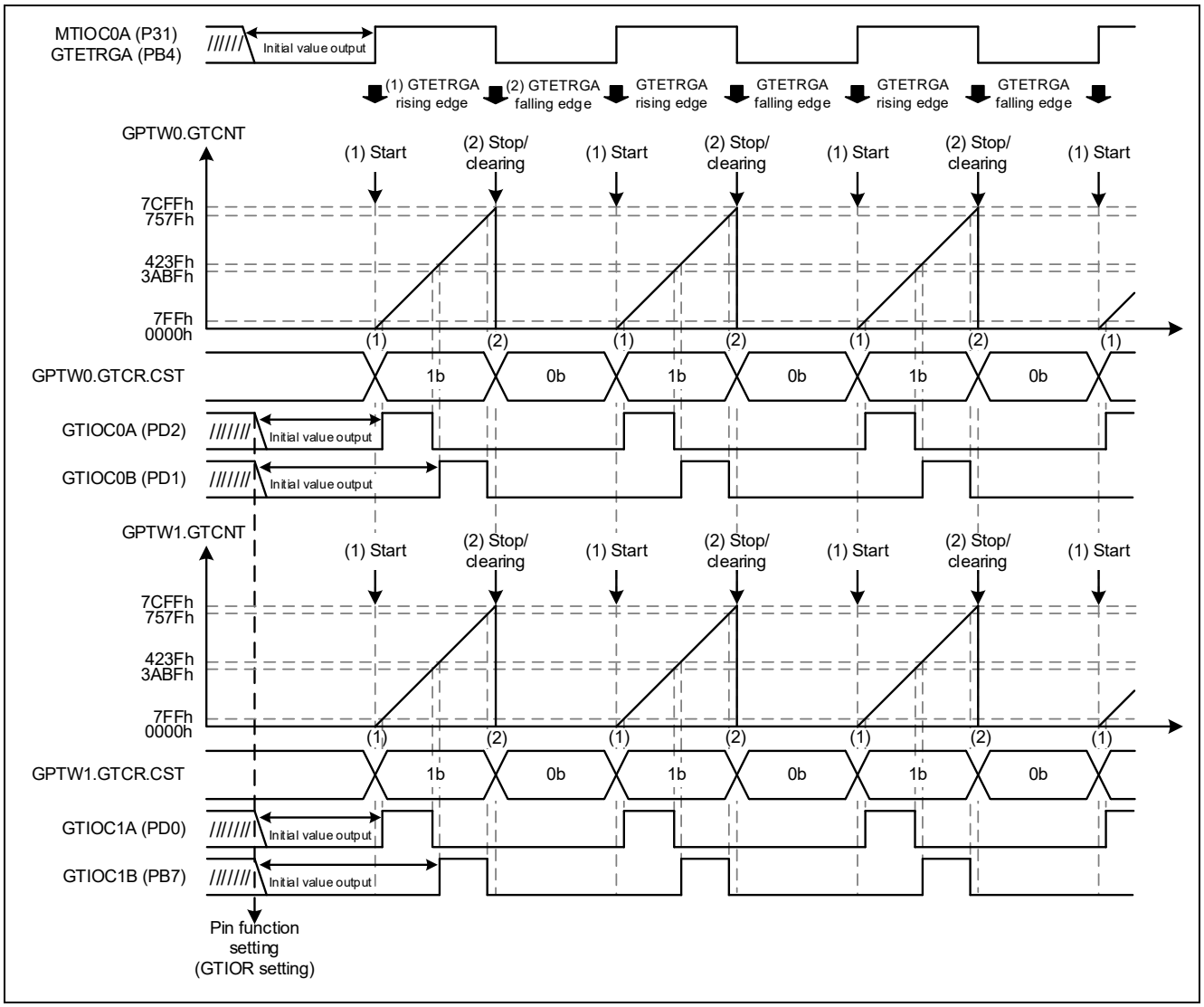


Figure 4.57 Example of MTU and GPTW Synchronous Operation Using One External Input Trigger (Start at GTETRGA Pin Input Rising Edge, Stop/Clearing at Input Falling Edge)

4.6.6 Usage Notes

4.6.6.1 Counting Starts for Multiple Channels

In this sample code, the SCH0 and SCH1 bits of timer counter synchronous start register TCSYSTR are set at the same time in the mtu0_mtu1_start function to start counting MTU0 and MTU1 at the same time.

When using the R_Config_MTUm_Start (m = 0, 1) function generated by the Smart Configurator, the counting starts timing may not be the same because each of the functions are read. The MTU0 and MTU1 counting can be started simultaneously by setting the CST0 and CST1 bits of timer start register TSTRA at the same time.

For details, refer to RX66T Group User's Manual: Hardware, section 22.2.17 Timer Start Registers (TSTRA, TSTRB, TSTR) and 22.2.19 Timer Counter Synchronous Start Register (TCSYSTR).

4.6.6.2 Order of Priority in Events

In this sample code, synchronous start/stop by software can be realized by using an external trigger of a hardware source.

When a start/stop by a hardware source set in GTSSR and GTPSR conflicts with a CPU write (GTSTR write/GTSTP write), the CPU write takes priority.

For details, refer to RX66T Group User's Manual: Hardware, section 24.10.5 Order of Priority in Events (2) The GTCR.CST Bit.

4.6.6.3 GTCNT Counter Start/Stop

In synchronous count start/stop operations using a hardware source, when a start source edge is detected, the GTCR.CST bit is set to 1b, and when a stop source edge is detected, the GTCR.CST bit is set to 0b.

Since the GTCNT counter starts/stops after the count clock is selected by TPCS[3:0] bits following the GTCR.CST bit update, events are ignored until the GTCNT counter actually starts, and events may be accepted, or interrupts may be generated after the CST bit is set to 0b.

For details, refer to RX66T Group User's Manual: Hardware, section 24.10.4 The GTCNT Counter Start/Stop.

4.6.6.4 Timing of Hardware Count Start, Stop, and Clear Operations

The start timing of synchronous operation differs according to the hardware source and clock used.

For details, refer to RX66T Group User's Manual: Hardware, section 24.3.7 Hardware Count Start/Count Stop and Clear Operation.

5. How to Import the Project

The sample code is provided in the format of an e² studio project. This chapter describes how to import a project into e² studio and CS+. After the import is complete, confirm the build and debugger settings.

5.1 Importing with e² studio

When using the sample code in e² studio, import it into e² studio using the following steps.

(The actual screen may vary according to the version of e² studio you are using.)

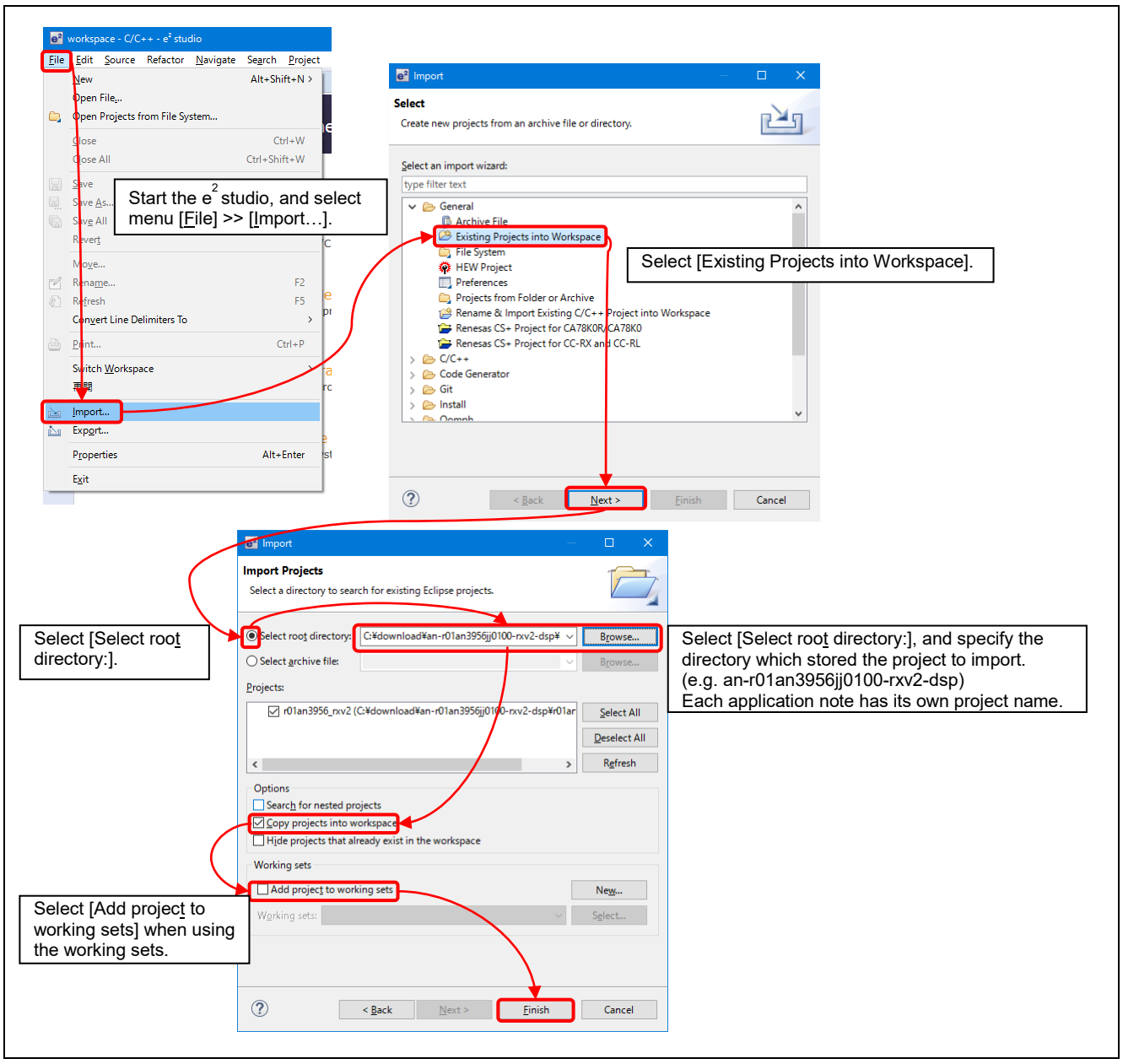


Figure 5.1 How to Import a Project into e² studio

5.2 Importing with CS+

When using the sample code with CS+, import the code to CS+ using the following steps.

(The actual screen may vary according to the version of CS+ you are using.)

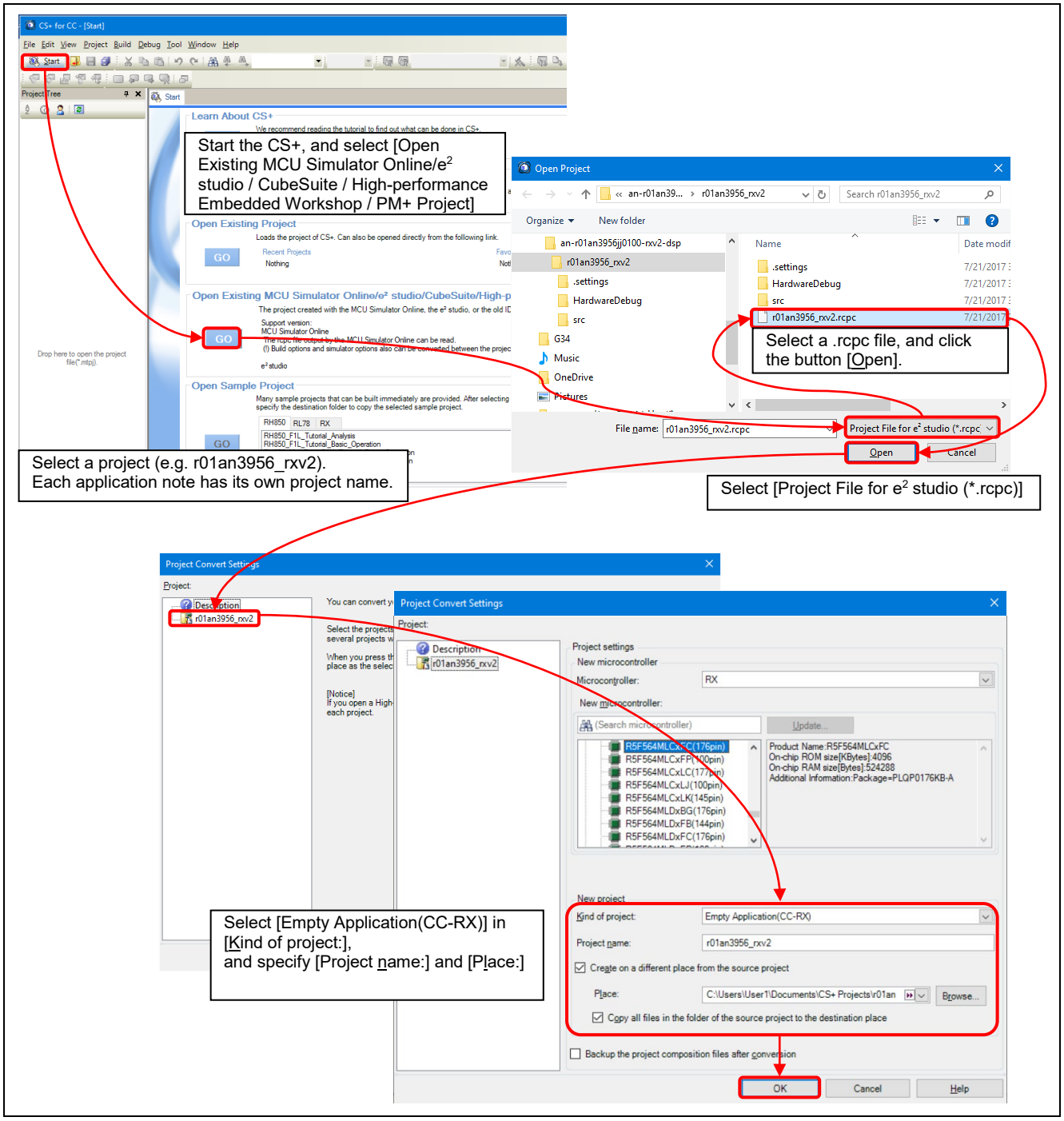


Figure 5.2 How to Import a Project into CS+

6. Reference Documents

- User's Manual: Hardware
  RX66T Group User's Manual: Hardware (R01UH0749)
  (Please obtain the latest version from the Renesas Electronics Corp. website.)

- Technical Updates/Technical News
  (Please obtain the latest version from the Renesas Electronics Corp. website.)

- User's Manual: Development Environment
  RX Family CC-RX Compiler User's Manual (R20UT3248)
  (Please obtain the latest version from the Renesas Electronics Corp. website.)

- User's Manual: Development Environment
  RX66T Group Renesas Starter Kit User's Manual (R20UT4150)
  (Please obtain the latest version from the Renesas Electronics Corp. website.)

- Application Note
  RX Family PWM Output Methods Using MTU3/GPTW (R01AN5995)
  (Please obtain the latest version from the Renesas Electronics Corp. website.)

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jun. 29, 2022	—	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.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

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

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

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

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Trademarks

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

Contact information

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