RZ/T1 Group

POE3 & GPTa Sample Driver

R01AN2597EJ0140
Rev.1.40
Jun. 07, 2018

Introduction

This application note explains a sample program that uses the RZ/T1 POE3 and GPTa functions to set the GPTa output pins to high impedance after a certain time.

The POE3 sample driver has the following features:

- FIT-based interfaces (R_POE3_Open() , R_POE3_Close(), R_POE3_Control() , R_POE3_GetVersion())
- Specifying the number of times of sampling
- High-impedance output (including software control)
- Interrupt

The GPTa sample driver has the following features:

- FIT-based interfaces (R_GPTA_Open() , R_GPTA_Close() , R_GPTA_Control(), R_GPTA_GetVersion())
- 16 bits x 4 channels
- Counting methods (count-up/down, triangle waves/sawtooth waves)
- Clock sources (PCLKC, PCLKC /2, PCLKC /4, PCLKC /8)
- Outputs (two per channel)
- Output compare/input capture (two per channel)
- Compare register (four for each output compare or input capture register, capable of operating as a buffer register as well)
- Generating PWM waveforms (including three-phase PWM waveforms)
- Frame period
- Synchronous operation/phase shift
- Generating dead time
- Starting, clearing, and stopping the counter
- Internal trigger
- ELC linkage (set up by the ELC driver)
- Noise filter
- Interrupt

The sample program has the following features:

- Setting POE3 output to high impedance during triangle wave PWM mode 1 operation (GTIOC0A/GTIOC0B output)

Target Devices

RZ/T1

When applying the sample program covered in this application note to another microcomputer, modify the program according to the specifications for the target microcomputer and conduct an extensive evaluation of the modified program.

Table of Contents

1. Specifications 4

2. Operating Environment 5

3. Related Application Note 6

4. Peripheral Functions 7

5. Hardware 8

5.1 Hardware Configuration 8

5.2 Pins 9

6. Software 10

6.1 Operation Outline 10

6.1.1 Project Setup 10

6.1.2 Preparation 10

6.2 Memory Mapping 11

6.2.1 Section Allocation for the Sample program 11

6.2.2 MPU Setup 11

6.2.3 Exception Processing Vector Table 11

6.3 Interrupts 11

6.4 Fixed-Width Integer Types 11

6.5 Constants/Error Codes 12

6.6 Structures, Unions, and Enumerated Types 17

6.7 Global Variables 24

6.8 Functions 27

6.9 Specifications of Functions 28

6.9.1 R_POE3_Open 28

6.9.2 R_POE3_Close 28

6.9.3 R_POE3_Control 28

6.9.4 R_POE3_GetVersion 29

6.9.5 R_GPTA_Open 29

6.9.6 R_GPTA_Close 29

6.9.7 R_GPTA_Control 30

6.9.8 R_GPTA_GetVersion 30

6.9.9 main 30

6.9.10 cmp_match_cb 31

6.10 Flowcharts 32

6.10.1 Main Processing 32

6.10.2 Callback Processing 34

6.11 R_POE3_Control Commands 35

6.11.1 POE3_SELECT_PORT 36

6.11.2 POE3_SET_INPUT_MODE 37

6.11.3 POE3_SET_OSC_MODE 38

6.11.4 POE3_SET_ALV 39

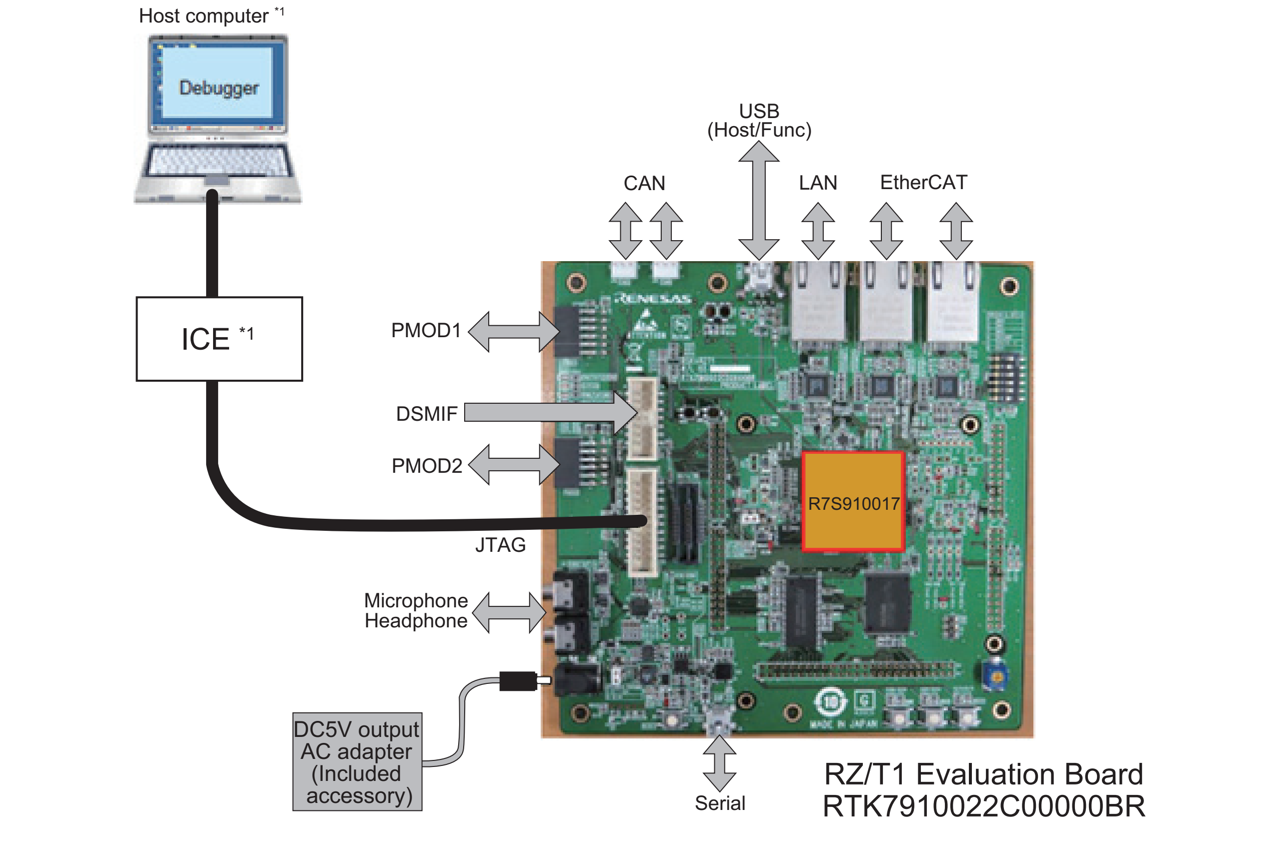
6.11.5 POE3_DETECT_SHORT 40
6.11.6 POE3_CTRL_HIZ 41
6.11.7 POE3_CTRL_HIZ_EX 43
6.11.8 POE3_REFLECT_REG 44
6.11.9 POE3_SOFT_HIZ 45
6.11.10 POE3_SET_CB 46
6.12 R_GPTA_Control Commands 47
6.12.1 GPTA_SET_COUNT_DIR 48
6.12.2 GPTA_SET_TIMER_MODE 49
6.12.3 GPTA_SET_CLK_SRC 50
6.12.4 GPTA_CTRL_IO 51
6.12.5 GPTA_CTRL_IO_NEG 53
6.12.6 GPTA_READ_REG 55
6.12.7 GPTA_WRITE_REG 56
6.12.8 GPTA_SET_BUF_MODE 58
6.12.9 GPTA_SET_AD_BUF_MODE 59
6.12.10 GPTA_SET_CB 60
6.12.11 GPTA_TIMER_START_SW 61
6.12.12 GPTA_CTRL_DTIME 62
6.12.13 GPTA_SET_HW_FACTOR 63
6.12.14 GPTA_SET_CLEAR_FACTOR 64
6.12.15 GPTA_SET_SYNC 65
6.12.16 GPTA_SET_NF 66
6.12.17 GPTA_SET_NF_EX 67
6.12.18 GPTA_SET_INTR_SKIP 68
7. Sample Program 71
8. Related Documents 72

1. Specifications

Table 1.1 Peripheral Functions and Applications lists the peripheral functions to be used and their applications, and Figure 1.1 shows the Operating Environment where the sample code is executed.

Table 1.1 Peripheral Functions and Applications

Peripheral Function	Application
RZ/T1 built-in Port Output Enable 3 (POE3)	Outputs high impedance signals to the GTIOC0A/B pin after a certain time of measurement based on the PWM timer.
RZ/T1 built-in General PWM Timer (GPTa)	Activates triangle wave PWM mode 1 and outputs PWM waveforms to the GTIOC0A/B pin.
RZ/T1 built-in interrupt controller (ICUA)	Uses the following interrupt source due to time measurement using the PWM timer: ch0 input capture/compare match A interrupt (ID: 178)
RZ/T1 built-in power saving function	Controls clock signal supply to the GPTa module used by the sample program.
RZ/T1 built-in I/O port RZ/T1 built-in Multi-function Pin Controller (MPC)	Sets up the pins related to the following modules used by the sample program: POE3 and GPTa Sets up the following ports (LEDs) to display the operating status of the sample program: LED10(PM7), LED 8(PM4), LED9(PM6), LED 6(PM5)



Note 1. Indicates the device that the user needs to prepare.

Figure 1.1 Operating Environment

2. Operating Environment

The sample code covered in this application note is for the environment below.

Table 2.1 Operating Environment

Item	Description
Microcomputer	RZ/T1 Group
Operating frequency	CPUCLK = 450 MHz
Operating voltage	3.3 V
Integrated Development Environment	Manufactured by IAR Systems Embedded Workbench® for Arm Version 8.20.2 Manufactured by Arm DS-5^{TM} 5.26.2 Manufactured by RENESAS e2studio 6.1.0
Operating mode	SPI boot mode 16-bit bus boot mode
Board	RZ/T1 Evaluation Board (RTK7910022C00000BR)
Device (functions to be used on the board)	• NOR flash memory (connected to CS0 and CS1 spaces) Manufacturer: Macronix International Co., Ltd. Model: MX29GL512FLT2I-10Q • SDRAM (connected to CS2 and CS3 spaces) Manufacturer: Integrated Silicon Solution Inc. Model: IS42S16320D-7TL • Serial flash memory Manufacturer: Macronix International Co., Ltd. Model: MX25L51245G • LED LED10(PM7), LED8(PM4), LED9(PM6), LED6(PM5)

3. Related Application Note

The application note related to this application note is listed below for reference.

- Application Note: RZ/T1 Group Initial Settings (R01A2554EJ)

Note: For any registers not covered by this application note, use the values specified in the Application Note: RZ/T1 Group Initial Settings.

4. Peripheral Functions

The basics of the operating modes, Port Output Enable 3 (POE3), General PWM Timer (GPTa), interrupt controller (ICUA), power saving function, and general I/O port are described in the RZ/T1 Group User's Manual: Hardware.

5. Hardware

5.1 Hardware Configuration

Figure 5.1 shows the hardware configuration.

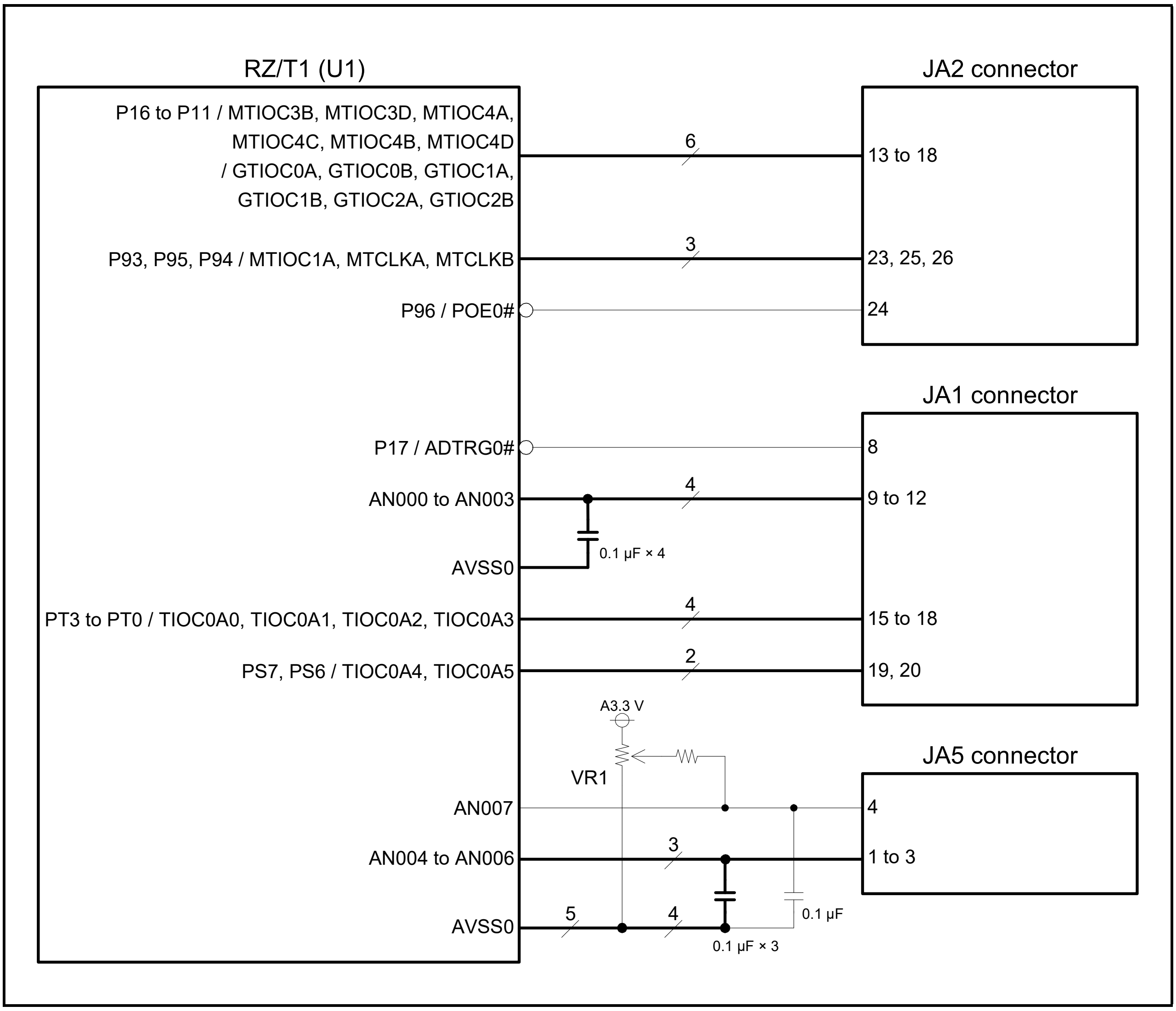

RZ/T1 (U1)
JA2 connector
P16 to P11 / MTIOC3B, MTIOC3D, MTIOC4A, MTIOC4C, MTIOC4B, MTIOC4D / GTIOC0A, GTIOC0B, GTIOC1A, GTIOC1B, GTIOC2A, GTIOC2B
6
13 to 18
P93, P95, P94 / MTIOC1A, MTCLKA, MTCLKB
3
23, 25, 26
P96 / POE0#
24
JA1 connector
P17 / ADTRG0#
8
AN000 to AN003
4
9 to 12
0.1 µF × 4
AVSS0
PT3 to PT0 / TIOC0A0, TIOC0A1, TIOC0A2, TIOC0A3
4
15 to 18
PS7, PS6 / TIOC0A4, TIOC0A5
2
19, 20
A3.3 V
VR1
JA5 connector
AN007
4
AN004 to AN006
3
1 to 3
AVSS0
5
4
0.1 µF
0.1 µF × 3

Figure 5.1 Hardware Configuration

5.2 Pins

Table 5.1 shows the pins and functions.

Table 5.1 Pins and Functions

Pin Name	Input/Output	Function
PM7	Output	Outputs the operating status of the sample program (LED10)
PM4	Output	Outputs the operating status of the sample program (LED8)
PM6	Output	Outputs the operating status of the sample program (LED9)
PM5	Output	Outputs the operating status of the sample program (LED6)
POE0#	Input	POE3 input pins (P71, P96*1, and PB5)
POE4#	Input	POE3 input pin (PR1*1)
POE8#	Input	POE3 input pins (PE7 and PP0*1)
POE10#	Input	POE3 input pins (P71, P96*1, and PB5)
GTETRG	Input	External trigger input pins for GPT0 to GPT3 (PC0 and PA3)
GTIOC0A*2	Output	GPT0 pins (P16, PB7, and PA2)
GTIOC0B*2	Output	GPT0 pins (P15, PF6, and PA1)
GTIOC1A*2	Output	GPT1 pins (P14, PF5, and PA0)
GTIOC1B*2	Output	GPT1 pins (P13, P77, and P87)
GTIOC2A*2	Output	GPT2 pins (P12*1, P76, and P86)
GTIOC2B*2	Output	GPT2 pins (P11*1, P75, and PD7)
GTIOC3A	Output	GPT3 pins (P66*1 and PA6)
GTIOC3B	Output	GPT3 pins (P67*1 and PA7)
MTIOC0A	Output	MTU0 pins (PE6 and PP4)
MTIOC0B	Output	MTU0 pins (PP3 and PE4)
MTIOC0C	Output	MTU0 pins (PE5 and PP2)
MTIOC0D	Output	MTU0 pins (PE3 and PP1)
MTIOC3B*2	Output	MTU complementary PWM output pins (P16, PB7, and PA2)
MTIOC3D*2	Output	MTU complementary PWM output pins (P15, PF6, and PA1)
MTIOC4A*2	Output	MTU complementary PWM output pins (P14, PF5, and PA0)
MTIOC4B*2	Output	MTU complementary PWM output pins (P12*1, P76, and P86)
MTIOC4C*2	Output	MTU complementary PWM output pins (P13, P77, and P87)
MTIOC4D*2	Output	MTU complementary PWM output pins (P11*1, P75, and PD7)
MTIOC6B	Output	MTU complementary PWM output pins (PA7 and PS5*1)
MTIOC6D	Output	MTU complementary PWM output pins (P70 and PS4*1)
MTIOC7A	Output	MTU complementary PWM output pins (PE7 and PS3*1)
MTIOC7B	Output	MTU complementary PWM output pins (P22 and PS1*1)
MTIOC7C	Output	MTU complementary PWM output pins (P42 and PS2*1)
MTIOC7D	Output	MTU complementary PWM output pins (PH6 and PS0*1)

Note 1. Only 320-pin type is available.
Note 2. Colored cells are the pins shared by GPTa and MTU3a.

Supplementary note: In addition to these pins, there are also pins that are used to initialize RZ/T1 products. For details, refer to the Application Note: RZ/T1 Group Initial Settings.

6. Software

6.1 Operation Outline

Table 6.1 Operation Outline presents a functional overview of the POE3 & GPTa sample program. Figure 6.1 shows the System Block Diagram for this program.

Table 6.1 Operation Outline

Function	Outline
Operation outline	Sets the GTIOCnA/B pin to high impedance by using the SPOER register of POE3 after a certain time during output from the GTIOCnA/B pin in triangle wave PWM mode 1.
GPT channel	GPT0
GPT operating mode	• Triangle wave PWM mode 1 Sets a period in the GPT0.GTPR register to operate the GPT0.GTCNT counter with triangle waves (full-wave), and outputs PWM waveforms to the GTIOC0A and GTIOC0B pins through compare match operations between the GPT0.GTCCRA and GPT0.GTCCRB registers.

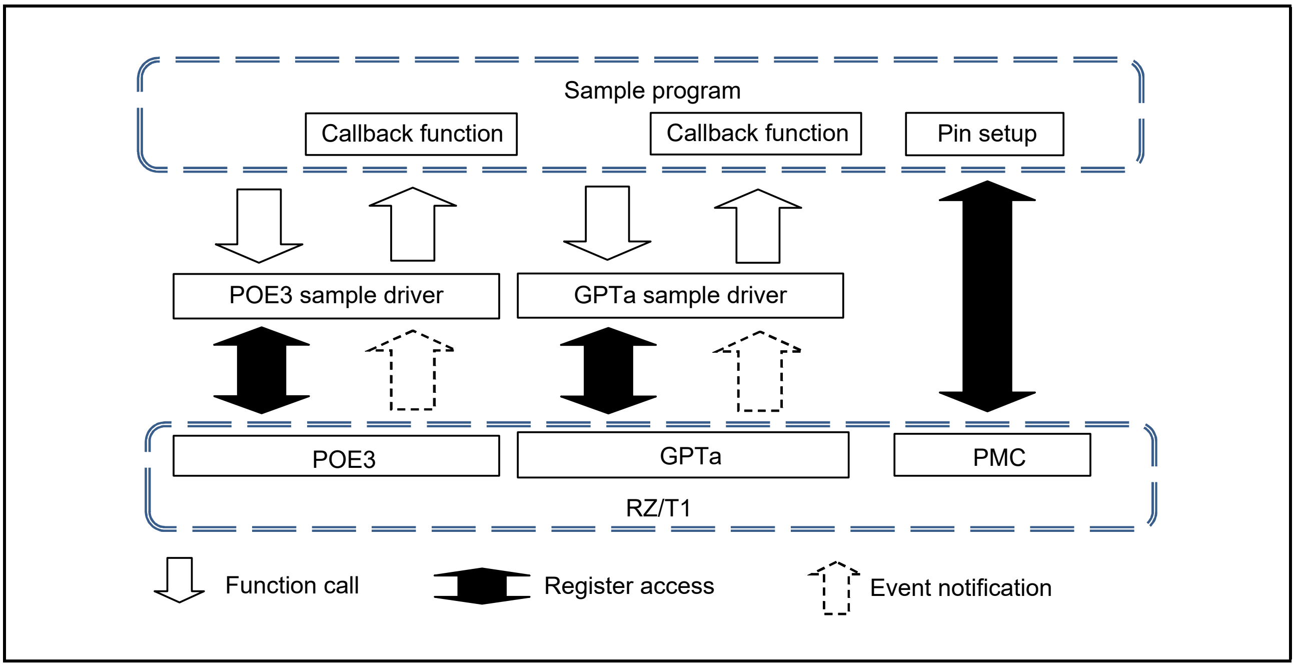


Figure 6.1 System Block Diagram

6.1.1 Project Setup

How to set up projects used in the EWARM development environment is described in the Application Note: RZ/T1 Group Initial Settings.

6.1.2 Preparation

There is no need to prepare for executing this sample program.

6.2 Memory Mapping

Memory mapping for the address spaces in the RZ/T1 Group MCU and the memory in the RZ/T1 Evaluation Board is described in the Application Note: RZ/T1 Group Initial Settings.

6.2.1 Section Allocation for the Sample program

The sections used by the sample program, the section allocation for the sample program in the initial state (load view), and the section allocation for the sample program after the scatter loading function is used (execution view) are described in the Application Note: RZ/T1 Group Initial Settings.

6.2.2 MPU Setup

MPU setup is described in the Application Note: RZ/T1 Group Initial Settings.

6.2.3 Exception Processing Vector Table

Exception processing vector tables are described in the Application Note: RZ/T1 Group Initial Settings.

6.3 Interrupts

Table 6.2 shows the Interrupts for the Sample Driver.

Table 6.2 Interrupts for the Sample Driver

Interrupt (Source ID)	Priority	Process Outline
GPTa ch0 input capture/compare match A interrupt (178)	15	Increments the interrupt count variable.

6.4 Fixed-Width Integer Types

Table 6.3 shows the Fixed-Width Integer Types for the Sample Driver.

Table 6.3 Fixed-Width Integer Types for the Sample Driver

Symbol	Description
int8_t	8-bit signed integer (defined in the standard library)
int16_t	16-bit signed integer (defined in the standard library)
int32_t	32-bit signed integer (defined in the standard library)
int64_t	64-bit signed integer (defined in the standard library)
uint8_t	8-bit unsigned integer (defined in the standard library)
uint16_t	16-bit unsigned integer (defined in the standard library)
uint32_t	32-bit unsigned integer (defined in the standard library)
uint64_t	64-bit unsigned integer (defined in the standard library)

6.5 Constants/Error Codes

Table 6.4 shows the Constants for the Sample Driver, and Table 6.5 shows the Error Codes for the Sample Driver.

Table 6.4 Constants for the Sample Driver (1 / 4)

Constant Name	Setting Value	Description
POE3_GPT3A_PA6	0x00U	Performs high-impedance control for PA6 as the GTIOC3A pin
POE3_GPT3A_P66	0x04U	Performs high-impedance control for P66 as the GTIOC3A pin
POE3_GPT3B_PA7	0x00U	Performs high-impedance control for PA7 as the GTIOC3B pin
POE3_GPT3B_P67	0x04U	Performs high-impedance control for P67 as the GTIOC3B pin
POE3_MTU0A_PE6	0x00U	Performs high-impedance control for PE6 as the MTIOC0A pin
POE3_MTU0A_PP4	0x02U	Performs high-impedance control for PP4 as the MTIOC0A pin
POE3_MTU0B_PP3	0x00U	Performs high-impedance control for PP3 as the MTIOC0B pin
POE3_MTU0B_PE4	0x01U	Performs high-impedance control for PE4 as the MTIOC0B pin
POE3_MTU0C_PE5	0x00U	Performs high-impedance control for PE5 as the MTIOC0C pin
POE3_MTU0C_PP2	0x02U	Performs high-impedance control for PP2 as the MTIOC0C pin
POE3_MTU0D_PE3	0x00U	Performs high-impedance control for PE3 as the MTIOC0D pin
POE3_MTU0D_PP1	0x02U	Performs high-impedance control for PP1 as the MTIOC0D pin
POE3_MTU3B_P16	0x01U	Performs high-impedance control for P16 as the MTIOC3B/GTIOC0B pin
POE3_MTU3B_PB7	0x02U	Performs high-impedance control for PB7 as the MTIOC3B/GTIOC0B pin
POE3_MTU3B_PA2	0x03U	Performs high-impedance control for PA2 as the MTIOC3B/GTIOC0B pin
POE3_MTU3D_P15	0x01U	Performs high-impedance control for P15 as the MTIOC3D/GTIOC0A pin
POE3_MTU3D_PF6	0x03U	Performs high-impedance control for PF6 as the MTIOC3D/GTIOC0A pin
POE3_MTU3D_PA1	0x05U	Performs high-impedance control for PA1 as the MTIOC3D/GTIOC0A pin
POE3_MTU4A_PA0	0x01U	Performs high-impedance control for PA0 as the MTIOC4A/GTIOC1B pin
POE3_MTU4A_PF5	0x03U	Performs high-impedance control for PF5 as the MTIOC4A/GTIOC1B pin
POE3_MTU4A_P14	0x05U	Performs high-impedance control for P14 as the MTIOC4A/GTIOC1B pin
POE3_MTU4C_P77	0x01U	Performs high-impedance control for P77 as the MTIOC4C/GTIOC1A pin
POE3_MTU4C_P87	0x03U	Performs high-impedance control for P87 as the MTIOC4C/GTIOC1A pin
POE3_MTU4C_P13	0x05U	Performs high-impedance control for P13 as the MTIOC4C/GTIOC1A pin
POE3_MTU4B_P86	0x00U	Performs high-impedance control for P86 as the MTIOC4B/GTIOC2B pin
POE3_MTU4B_P76	0x03U	Performs high-impedance control for P76 as the MTIOC4B/GTIOC2B pin
POE3_MTU4B_P12	0x05U	Performs high-impedance control for P12 as the MTIOC4B/GTIOC2B pin
POE3_MTU4D_P75	0x01U	Performs high-impedance control for P75 as the MTIOC4D/GTIOC2A pin
POE3_MTU4D_PD7	0x03U	Performs high-impedance control for PD7 as the MTIOC4D/GTIOC2A pin
POE3_MTU4D_P11	0x05U	Performs high-impedance control for P11 as the MTIOC4D/GTIOC2A pin
POE3_INPUT_MODE_0	0U	Receives a request at a falling edge of a clock pulse.
POE3_INPUT_MODE_4	1U	Receives a request every four clock pulses when sampling is performed 16 times.
POE3_INPUT_MODE_16	2U	Receives a request every 16 clock pulses when sampling is performed 16 times.
POE3_INPUT_MODE_128	3U	Receives a request every 128 clock pulses when sampling is performed 16 times.
POE3_ALV_LOW	0U	Active-low
POE3_ALV_HIGH	1U	Active-high
GPTA_COUNT_DOWN	0U	Count-down
GPTA_COUNT_UP	1U	Count-up
GPTA_SAW_PWM_MODE	0U	Sawtooth wave PWM mode
GPTA_SAW_ONE_SHOT_MODE	1U	Sawtooth wave one-shot pulse mode

Table 6.4 Constants for the Sample Driver (2 / 4)

Constant Name	Setting Value	Description
GPTA_TRIANGLE_PWM_MODE1	4U	Triangle wave PWM mode 1
GPTA_TRIANGLE_PWM_MODE2	5U	Triangle wave PWM mode 2
GPTA_TRIANGLE_PWM_MODE3	6U	Triangle wave PWM mode 3
GPTA_PCLCK_DIV_1	0U	Specifies PCLKC for the clock source
GPTA_PCLCK_DIV_2	1U	Specifies PCLKC/2 for the clock source
GPTA_PCLCK_DIV_4	2U	Specifies PCLKC/4 for the clock source
GPTA_PCLCK_DIV_8	3U	Specifies PCLKC/8 for the clock source
GPTA_IO_CMP_MATCH	0x00U	Specifies compare match operation
GPTA_IO_INPUT_CAPTURE	0x01U	Specifies input capture operation
GPTA_IO_LOW	0x00U	Sets the initial output to low
GPTA_IO_HIGH	0x01U	Sets the initial output to high
GPTA_IO_HOLD_OUTPUT	0x00U	Sets to hold output at the end of the period
GPTA_IO_HOLD_LOW	0x01U	Sets to output a low-level signal at the end of the period
GPTA_IO_HOLD_HIGH	0x02U	Sets to output a high-level signal at the end of the period
GPTA_IO_HOLD_TOGGLE	0x03U	Sets to toggle the output at the end of the period
GPTA_IO_CMP_OUTPUT	0x00U	Sets to hold output by compare match
GPTA_IO_CMP_LOW	0x01U	Sets to output a low-level signal by compare match
GPTA_IO_CMP_HIGH	0x02U	Sets to output a high-level signal by compare match
GPTA_IO_CMP_TOGGLE	0x03U	Sets to toggle the output by compare match
GPTA_IO_TRG_RISING	0x00U	Specifies to perform an input capture operation at a rising edge
GPTA_IO_TRG_FALLING	0x01U	Specifies to perform an input capture operation at a falling edge
GPTA_IO_TRG_BOTH	0x02U	Specifies to perform an input capture operation at both edges
GPTA_DFLT_LOW	0x00U	Sets to output a low-level signal when the counter stops
GPTA_DFLT_HIGH	0x01U	Sets to output a high-level signal when the counter stops
GPTA_HLD_REG	0x00U	Follows the register settings as to the output to be performed when the counter starts or stops
GPTA_HLD_HOLD	0x01U	Sets to retain the output level when the counter starts or stops
GPTA_NEG_CTRL_A_DISABLE	0x0000U	Disables negation control for the GTIOnA pin
GPTA_NEG_CTRL_A_ENABLE	0x0001U	Enables negation control for the GTIOnA pin
GPTA_NEG_CTRL_B_DISABLE	0x0000U	Disables negation control for the GTIOnB pin
GPTA_NEG_CTRL_B_ENABLE	0x0001U	Enables negation control for the GTIOnB pin
GPTA_NEG_OUTPUT_A_LOW	0x0000U	Outputs "0" to the GTIOnA pin during negation control
GPTA_NEG_OUTPUT_A_HIGH	0x0001U	Outputs "1" to the GTIOnA pin during negation control
GPTA_NEG_OUTPUT_B_LOW	0x0000U	Outputs "0" to the GTIOnB pin during negation control
GPTA_NEG_OUTPUT_B_HIGH	0x0001U	Outputs "1" to the GTIOnB pin during negation control
GPTA_NEG_GETTRG	0x0007U	Sets input to the GETTRG pin as the negation source
GPTA_NEG_SWN	0x0001U	Sets a negation source by soft_control
GPTA_NEG_POLARITY_LOW	0x0000U	Sets the polarity of the negation source to "0"
GPTA_NEG_POLARITY_HIGH	0x0001U	Sets the polarity of the negation source to "1"
GPTA_NEG_SWN_LOW	0x0000U	Sets the negation source to "0" by software control
GPTA_NEG_SWN_HIGH	0x0001U	Sets the negation source to "1" by software control
GPTA_NEG_FLG_A_DISABLE	0x0000U	Disables output from the GTIOnA pin
GPTA_NEG_FLG_A_ENABLE	0x0001U	Enables output from the GTIOnA pin
GPTA_NEG_FLG_B_DISABLE	0x0000U	Disables output from the GTIOnB pin
GPTA_NEG_FLG_B_ENABLE	0x0001U	Enables output from the GTIOnB pin
GPTA_BUF_NONE	0x0000U	Sets no-buffer operation

Table 6.4 Constants for the Sample Driver (3 / 4)

Constant Name	Setting Value	Description
GPTA_BUF_SINGLE	0x0001U	Sets single-buffer operation
GPTA_BUF_DOUBLE	0x0002U	Sets double-buffer operation
GPTA_AD_BUF_TRANS_NONE	0x0000U	No buffer operation
GPTA_AD_BUF_TRANS_TOP	0x0001U	Executes buffer transfer at the peak of a clock pulse
GPTA_AD_BUF_TRANS_BOTTOM	0x0002U	Executes buffer transfer at the valley of a clock pulse
GPTA_AD_BUF_TRANS_BOTH	0x0003U	Executes buffer transfer at the peak and valley of a clock pulse
GPTA_AD_BUF_SINGLE	0x0000U	Sets single-buffer operation
GPTA_AD_BUF_DOUBLE	0x0001U	Sets double-buffer operation
GPTA_TIMER_START_0	0x0001U	Sets to start operating the counter for channel 0 (GPT0)
GPTA_TIMER_START_1	0x0002U	Sets to start operating the counter for channel 1 (GPT1)
GPTA_TIMER_START_2	0x0004U	Sets to start operating the counter for channel 2 (GPT2)
GPTA_TIMER_START_3	0x0008U	Sets to start operating the counter for channel 3 (GPT3)
GPTA_TIMER_STOP_0	0x0001U	Sets to stop operating the counter for channel 0 (GPT0)
GPTA_TIMER_STOP_1	0x0002U	Sets to stop operating the counter for channel 1 (GPT1)
GPTA_TIMER_STOP_2	0x0004U	Sets to stop operating the counter for channel 2 (GPT2)
GPTA_TIMER_STOP_3	0x0008U	Sets to stop operating the counter for channel 3 (GPT3)
GPTA_DTIME_MANUAL_SET	0x0000U	Sets dead time manually
GPTA_DTIME_AUTO_SET	0x0001U	Sets dead time with the GTDVU and GTDVD registers
GPTA_GTDVU_BUF_DISABLE	0x0000U	Disables buffer operation for the GTDVU register
GPTA_GTDVU_BUF_ENABLE	0x0001U	Enables buffer operation for the GTDVU register
GPTA_GTDVD_BUF_DISABLE	0x0000U	Disables buffer operation for the GTDVD register
GPTA_GTDVD_BUF_ENABLE	0x0001U	Enables buffer operation for the GTDVD register
GPTA_GTDVD_MANUAL_SET	0x0000U	Sets the GTDVD register manually
GPTA_GTDVD_AUTO_SET	0x0001U	Sets the value of the GTDVU register in the GTDVD register
GPTA_TRG_TYPE_NONE	0x0000U	Does not operate the counter based on hardware sources
GPTA_TRG_TYPE_RISING	0x0001U	Operates the counter at a rising edge based on hardware sources
GPTA_TRG_TYPE_FALLING	0x0002U	Operates the counter at a falling edge based on hardware sources
GPTA_TRG_TYPE_BOTH	0x0003U	Operates the counter at both edges based on hardware sources
GPTA_FACTOR_NONE	0x0000U	Sets no hardware sources
GPTA_FACTOR_GTIOC3A	0x0008U	Sets input to the GTIOC3A pin as the hardware source
GPTA_FACTOR_GTIOC3B	0x0009U	Sets input to the GTIOC3B pin as the hardware source
GPTA_FACTOR_CMP_A	0x000AU	Sets internal output from the GTIOC3A pin as the hardware source (Output compare)
GPTA_FACTOR_CMP_B	0x000BU	Sets internal output from the GTIOC3B pin as the hardware source (Output compare)
GPTA_FACTOR_GETTRG	0x000CU	Sets input to the GETTRG pin as the hardware source
GPTA_CLEAR_NONE	0x0000U	Does not set the source for clearing the counter
GPTA_CLEAR_GTCCRA	0x0001U	Clears the counter by input capture for the GTCCRA register
GPTA_CLEAR_GTCCRB	0x0002U	Clears the counter by input capture for the GTCCRB register
GPTA_CLEAR_SYNC	0x0003U	Clears the counter by synchronous clearing or by another clearing source that operates synchronously
GPTA_CLEAR_GPT0	0x0000U	Clears the counter by the clearing source for GPT0 (channel 0)
GPTA_CLEAR_GPT1	0x0001U	Clears the counter by the clearing source for GPT1 (channel 1)
GPTA_CLEAR_GPT2	0x0002U	Clears the counter by the clearing source for GPT2 (channel 2)
GPTA_CLEAR_GPT3	0x0003U	Clears the counter by the clearing source for GPT3 (channel 3)
GPTA_NF_DISABLE	0x0000U	Disables the noise filter

Table 6.4 Constants for the Sample Driver (4 / 4)

Constant Name	Setting Value	Description
GPTA_NF_ENABLE	0x0001U	Enables the noise filter
GPTA_SAMP_PCLKC_DIV_1	0x0000U	Specifies PCLKC / 1 for the sampling clock for the noise filter
GPTA_SAMP_PCLKC_DIV_4	0x0001U	Specifies PCLKC / 4 for the sampling clock for the noise filter
GPTA_SAMP_PCLKC_DIV_32	0x0002U	Specifies PCLKC / 32 for the sampling clock for the noise filter
GPTA_SAMP_COUNT_SRC	0x0003U	Specifies the count source for the sampling clock for the noise filter
GPTA_NF_EX_DISABLE	0x0000U	Disables the noise filter for the external trigger input pin
GPTA_NF_EX_ENABLE	0x0001U	Enables the noise filter for the external trigger input pin
GPTA_SAMP_EX_PCLKC_DIV_1	0x0000U	Specifies PCLKC / 1 for the sampling clock for the noise filter of the external trigger input pin
GPTA_SAMP_EX_PCLKC_DIV_2	0x0001U	Specifies PCLKC / 2 for the sampling clock for the noise filter of the external trigger input pin
GPTA_SAMP_EX_PCLKC_DIV_4	0x0002U	Specifies PCLKC / 4 for the sampling clock for the noise filter of the external trigger input pin
GPTA_SAMP_EX_PCLKC_DIV_32	0x0003U	Specifies PCLKC / 32 for the sampling clock for the noise filter of the external trigger input pin
GTCCRA_INTR_SKIP_DISABLE	0x0000U	Does not apply the interrupt skip function to GTCCRA
GTCCRA_INTR_SKIP_ENABLE	0x0001U	Applies the interrupt skip function to GTCCRA
GTCCRB_INTR_SKIP_DISABLE	0x0000U	Does not apply the interrupt skip function to GTCCRB
GTCCRB_INTR_SKIP_ENABLE	0x0001U	Applies the interrupt skip function to GTCCRB
GTCCRC_INTR_SKIP_DISABLE	0x0000U	Does not apply the interrupt skip function to GTCCRC
GTCCRC_INTR_SKIP_ENABLE	0x0001U	Applies the interrupt skip function to GTCCRC
GTCCRD_INTR_SKIP_DISABLE	0x0000U	Does not apply the interrupt skip function to GTCCRD
GTCCRD_INTR_SKIP_ENABLE	0x0001U	Applies the interrupt skip function to GTCCRD
GTCCRE_INTR_SKIP_DISABLE	0x0000U	Does not apply the interrupt skip function to GTCCRE
GTCCRE_INTR_SKIP_ENABLE	0x0001U	Applies the interrupt skip function to GTCCRE
GTCCRF_INTR_SKIP_DISABLE	0x0000U	Does not apply the interrupt skip function to GTCCRF
GTCCRF_INTR_SKIP_ENABLE	0x0001U	Applies the interrupt skip function to GTCCRF
GTADTRA_INTR_SKIP_DISABLE	0x0000U	Does not apply the interrupt skip function to GTADTRA
GTADTRA_INTR_SKIP_ENABLE	0x0001U	Applies the interrupt skip function to GTADTRA
GTADTRB_INTR_SKIP_DISABLE	0x0000U	Does not apply the interrupt skip function to GTADTRB
GTADTRB_INTR_SKIP_ENABLE	0x0001U	Applies the interrupt skip function to GTADTRB
GPTA_INTR_SKIP_NONE	0x0000U	Disables the interrupt skip function
GPTA_INTR_SKIP_TOP	0x0001U	Skips an interrupt by counting both overflows and underflows for sawtooth waves or peaks for triangle waves
GPTA_INTR_SKIP_BOTTOM	0x0002U	Skips an interrupt by counting both overflows and underflows for sawtooth waves or valleys for triangle waves
GPTA_INTR_SKIP_BOTH	0x0003U	Skips an interrupt by counting both overflows and underflows for sawtooth waves or both peaks and valleys for triangle waves
GPTA_INTR_SKIP_0	0x0000U	Disables the interrupt skip function
GPTA_INTR_SKIP_1	0x0001U	Number of interrupt skips: 1
GPTA_INTR_SKIP_2	0x0002U	Number of interrupt skips: 2
GPTA_INTR_SKIP_3	0x0003U	Number of interrupt skips: 3
GPTA_INTR_SKIP_4	0x0004U	Number of interrupt skips: 4
GPTA_INTR_SKIP_5	0x0005U	Number of interrupt skips: 5
GPTA_INTR_SKIP_6	0x0006U	Number of interrupt skips: 6
GPTA_INTR_SKIP_7	0x0007U	Number of interrupt skips: 7

Table 6.5 Error Codes for the Sample Driver

Constant Name	Setting Value	Description
POE3_SUCCESS	0	Processing successful
POE3_ERR_INVALID_ARG	-1	Argument error
POE3_ERR_STATUS	-2	Status error
POE3_ERR_INVALID_CMD	-3	Control type error
POE3_ERR_ALREADY_REFLECTED	-4	Already reflected
GPTA_SUCCESS	0	Processing successful
GPTA_ERR_INVALID_ARG	-1	Argument error
GPTA_ERR_STATUS	-2	Status error
GPTA_ERR_INVALID_CMD	-3	Control type error
GPTA_ERR_TIMER_STARTED	-4	Timer operating

6.6 Structures, Unions, and Enumerated Types

The following figures show the Structures, Unions, and Enumerated Types for the Sample Driver.

```
/* Identifiers for enabling and disabling POE3 */
typedef enum {
        POE3_DISABLE,
        POE3_ENABLE
} poe3_flag_t;

/* Identifiers of POE3 I/O ports */
typedef enum
{
        POE3_IN_POE0,
        POE3_IN_POE4,
        POE3_IN_POE8,
        POE3_IN_POE10,
        POE3_OUT_GPT0A,
        POE3_OUT_GPT0B,
        POE3_OUT_GPT1A,
        POE3_OUT_GPT1B,
        POE3_OUT_GPT2A,
        POE3_OUT_GPT2B,
        POE3_OUT_GPT3A,
        POE3_OUT_GPT3B,
        POE3_OUT_MTU0A,
        POE3_OUT_MTU0B,
        POE3_OUT_MTU0C,
        POE3_OUT_MTU0D,
        POE3_OUT_MTU3B,
        POE3_OUT_MTU3D,
        POE3_OUT_MTU4A,
        POE3_OUT_MTU4B,
        POE3_OUT_MTU4C,
        POE3_OUT_MTU4D,
        POE3_OUT_MTU6B,
        POE3_OUT_MTU6D,
        POE3_OUT_MTU7A,
        POE3_OUT_MTU7B,
        POE3_OUT_MTU7C,
        POE3_OUT_MTU7D,
} poe3_ioport_t;

/* --- POE3_SELECT_PORT --- */
typedef struct ex_poe3_select_port_t
{
        poe3_ioport_t     outport;
        uint16_t          port_select;
} poe3_select_port_t;
```

Figure 6.2 Structures, Unions, and Enumerated Types for the Sample Driver (1/7)

```
/* --- POE3_SET_INPUT_MODE --- */
typedef struct ex_poe3_set_input_t
{
        poe3_ioport_t inport;
        uint8_t mode;
        poe3_flag_t intr_flag;
} poe3_set_input_t;

/* --- POE3_SET_OCS_MODE --- */
typedef struct ex_poe3_set_ocs_mode_t
{
        poe3_flag_t hiz_flag;
} poe3_set_ocs_mode_t;

/* --- POE3_SET_ALV --- */
typedef struct ex_poe3_set_alv_t
{
        poe3_ioport_t outport;
        uint8_t active_level;
} poe3_set_alv_t;

/* --- POE3_DETECT_SHORT --- */
typedef struct ex_poe3_detect_short_t
{
        poe3_port_grp_t group;
        poe3_flag_t hiz_flag;
        poe3_flag_t intr_flag;
} poe3_detect_short_t;

typedef enum
{
        POE3_MTU34_GPT012,
        POE3_MTU67,
        POE3_MTU0,
        POE3_GPT3,
} poe3_port_grp_t;

/* --- POE3_CTRL_HIZ --- */
typedef struct ex_poe3_ctrl_hiz_t
{
        poe3_hiz_grp_t group;
        poe3_flag_t hiz_flag;
} poe3_ctrl_hiz_t;
```

Figure 6.3 Structures, Unions, and Enumerated Types for the Sample Driver (2/7)

```
typedef enum
{
        POE3_HIZ_MTU0A,
        POE3_HIZ_MTU0B,
        POE3_HIZ_MTU0C,
        POE3_HIZ_MTU0D,
        POE3_HIZ_MTU3BD_GPT0AB,
        POE3_HIZ_MTU4AC_GPT1AB,
        POE3_HIZ_MTU4BD_GPT2AB,
        POE3_HIZ_MTU6BD,
        POE3_HIZ_MTU7AC,
        POE3_HIZ_MTU7BD,
        POE3_HIZ_GPT3AB
} poe3_hiz_grp_t;

/* --- POE3_CTRL_HIZ_EX --- */
typedef struct ex_poe3_ctrl_hiz_ex_t
{
        poe3_hiz_grp_t group;
        poe3_ioport_t port;
        poe3_flag_t hiz_flag;
} poe3_ctrl_hiz_ex_t;

/* --- POE3_SOFT_HIZ --- */
typedef struct ex_poe3_soft_hiz_t
{
        poe3_port_grp_t group;
        poe3_flag_t hiz_flag;
} poe3_soft_hiz_t;

/* --- POE3_SET_CB --- */
typedef struct ex_poe3_set_cb_t
{
        poe3_intr_t intr_type;
        poe3_cb_t callback;
        uint8_t intr_pri;
} poe3_set_cb_t;

typedef enum
{
        POE3_INTR_POE0,
        POE3_INTR_POE4,
        POE3_INTR_POE8,
        POE3_INTR_POE10
} poe3_intr_t;

typedef void (*poe3_cb_t)(const uint16_t port_flag, const uint16_t short_flag);

/* --- GPTA_SET_COUNT_DIR --- */
typedef struct ex_gpta_set_count_dir_t
{
        uint8_t ud;
        gpta_flag_t udf_flag;
} gpta_set_count_dir_t;
```

Figure 6.4 Structures, Unions, and Enumerated Types for the Sample Driver (3/7)

```
/* --- GPTA_SET_TIMER_MODE --- */
typedef struct ex_gpta_set_timer_mode_t
{
        uint8_t mode;
} gpta_set_timer_mode_t;

/* --- GPTA_SET_CLK_SRC --- */
typedef struct ex_gpta_set_clk_src_t
{
        uint8_t clock;
} gpta_set_clk_src_t;

/* --- GPTA_CTRL_IO --- */
typedef struct ex_gpta_ctrl_io_t
{
        gpta_port_t port;
        uint8_t io_mode;
        uint8_t io_default;
        uint8_t io_hold;
        uint8_t io_output;
        uint8_t dflt;
        uint8_t hld;
} gpta_ctrl_io;

typedef enum
{
        GPTA_PORT_A,
        GPTA_PORT_B
} gpta_port_t;

/* --- GPTA_CTRL_IO_NEG --- */
typedef struct ex_gpta_ctrl_io_neg_t
{
        uint16_t control_a;
        uint16_t control_b;
        uint16_t output_a;
        uint16_t output_b;
        uint16_t factor;
        uint16_t polarity_select;
        uint16_t soft_control;
        uint16_t flag_a;
        uint16_t flag_b;
} gpta_ctrl_io_neg_t;

/* --- GPTA_READ_REG --- */
typedef struct ex_gpta_read_reg_t
{
        gpta_reg_t register;
        uint16_t *pdata;
} gpta_read_reg_t;
```

Figure 6.5 Structures, Unions, and Enumerated Types for the Sample Driver (4/7)

```
typedef enum
{
        GPTA_REG_GTCNT
        GPTA_REG_GTCCRA
        GPTA_REG_GTCCRB
        GPTA_REG_GTCCRC
        GPTA_REG_GTCCRD
        GPTA_REG_GTCCRE
        GPTA_REG_GTCCRF
        GPTA_REG_GTPR
        GPTA_REG_GTPBR
        GPTA_REG_GTPDBR
        GPTA_REG_GTADTRA
        GPTA_REG_GTADTRB
        GPTA_REG_GTADTBRA
        GPTA_REG_GTADTBRB
        GPTA_REG_GTADTDBRA
        GPTA_REG_GTADTDBRB
        GPTA_REG_GTDVU
        GPTA_REG_GTDVD
        GPTA_REG_GTDBU
        GPTA_REG_GTDBD
} gpta_reg_t;

/* --- GPTA_WRITE_REG --- */
typedef struct ex_gpta_write_reg_t
{
        gpta_reg_t register;
        uint16_t data;
} gpta_write_reg_t;

/* --- GPTA_SET_BUF_MODE --- */
typedef struct ex_gpta_set_buf_mode_t
{
        gpta_buf_t buf_id;
        uint16_t mode;
} gpta_set_buf_mode_t;

typedef enum
{
        GPTA_BUF_GTCCRA,
        GPTA_BUF_GTCCRB,
        GPTA_BUF_GTPR,
        GPTA_AD_BUF_GTADTRA,
        GPTA_AD_BUF_GTADTRB
} gpta_buf_t;

/* --- GPTA_SET_AD_BUF_MODE --- */
typedef struct ex_gpta_set_ad_buf_mode_t
{
        gpta_buf_t buf_id;
        uint16_t timing
        uint16_t mode;
} gpta_set_ad_buf_mode_t;
```

Figure 6.6 Structures, Unions, and Enumerated Types for the Sample Driver (5/7)

```
/* --- GPTA_SET_CB --- */
typedef struct ex_gpta_set_cb_t
{
        gpta_intr_t intr_type;
        gpta_cb callback;
        uint8_t pri;
} gpta_set_cb_t;

typedef enum
{
        GPTA_INTR_GTCIA,
        GPTA_INTR_GTCIB,
        GPTA_INTR_GTCIC,
        GPTA_INTR_GTCID,
        GPTA_INTR_GTCIE,
        GPTA_INTR_GTCIF,
        GPTA_INTR_GDTE,
        GPTA_INTR_GTCIV,
        GPTA_INTR_GTCIU,
        GPTA_INTR_ETGIN,
        GPTA_INTR_ETGIP,
} gpta_intr_t;
;
typedef void (*gpta_cb_t)(void);

/* --- GPTA_TIMER_START_SW --- */
typedef struct ex_gpta_timer_start_t
{
        uint8_t start_ch;
        uint8_t stop_ch;
} gpta_timer_start_t;

/* --- GPTA_CTRL_DTIME --- */
typedef struct ex_gpta_ctrl_dtime_t
{
        uint16_t set_mode;
        uint16_t gtdvu_buf_mode;
        uint16_t gtdvd_buf_mode;
        uint16_t gtdvd_set_mode;
} gpta_ctrl_dtime_t;

/* --- GPTA_SET_HW_FACTOR --- */
typedef struct ex_gpta_set_hw_factor_t
{
        gpta_count_type_t count_type;
        uint16_t trigger_type;
        uint16_t factor;
} gpta_set_hw_factor_t;

typedef enum
{
        GPTA_COUNT_START,
        GPTA_COUNT_STOP,
        GPTA_COUNT_CLEAR,
        GPTA_COUNT_CLEAR_SW
} gpta_count_type_t;
```

Figure 6.7 Structures, Unions, and Enumerated Types for the Sample Driver (6/7)

```
/* --- GPTA_SET_CLEAR_FACTOR --- */
typedef struct ex_gpta_set_clear_factor_t
{
        uint16_t clear_factor;
} gpta_set_clear_factor_t;

/* --- GPTA_SET_SYNC --- */
typedef struct ex_gpta_set_sync_t
{
        uint16_t clear_factor;
} gpta_set_sync_t;

/* --- GPTA_SET_NF --- */
typedef struct ex_gpta_set_nf_t
{
        uint16_t filter_a_flag;
        uint16_t filter_b_flag;
        uint16_t sampling_clock;
} gpta_set_nf_t;

/* --- GPTA_SET_NF_EX --- */
typedef struct ex_gpta_set_nf_ex_t
{
        uint16_t filter_flag;
        uint16_t sampling_clock;
} gpta_set_nf_ex_t;

/* --- GPTA_SET_INTR_SKIP --- */
typedef struct ex_gpta_set_intr_skip_t
{
        uint16_t gtccra_mode;
        uint16_t gtccrb_mode;
        uint16_t gtccrc_mode;
        uint16_t gtccrd_mode;
        uint16_t gtccre_mode;
        uint16_t gtccrf_mode;
        uint16_t gtadtra_mode;
        uint16_t gtadtrb_mode;
        uint16_t func_sel;
        uint16_t skip_num;
} gpta_set_intr_skip_t;
```

Figure 6.8 Structures, Unions, and Enumerated Types for the Sample Driver (7/7)

6.7 Global Variables

Table 6.6 shows the Global Variables.

Table 6.6 Global Variables (1 / 3)

Type	Variable Name	Description	Function
static uint32_t	gb_poe3_sts	POE3 driver state	R_POE3_Open R_POE3_Close R_POE3_Control
static poe3_intr_entry_t	poe3_intr_entry	POE3 callback function entry	poe3_intr_poe0_proc poe3_intr_poe4_proc poe3_intr_poe8_proc poe3_intr_poe10_proc poe3_poe0_isr poe3_poe4_isr poe3_poe8_isr poe3_poe10_isr
const static poe3_cmdtbl_t	poe3_cmd_tbl[]	POE3 command table	poe3_get_cmd_func
const static poe3_port_select_t	poe3_port_select_tbl[]	POE3 command validation table	proc_select_port
const static poe3_ctrl_hiz_ex_local_t	poe3_ctrl_hiz_ex_local_tbl[]	POE3 command validation table	check_hiz_ex_combination
static struct st_poe	reg_array	Temporary storage array for the POE3 register	poe3_open proc_select_port proc_set_input_mode proc_set_osc_mode proc_set_alv proc_detect_short proc_ctrl_hiz proc_ctrl_hiz_ex proc_reflect_reg
static int32_t	refrect_flag	Reflection status storage flag for the POE3 register	check_cmd_executable proc_reflect_reg

Table 6.6 Global Variables (2 / 3)

Type	Variable Name	Description	Function
static gpta_intr_entry_t	intr_entry_tbl[]	GPTa callback function entry	void gpta_intr_proc int32_t get_intr_no gpta_gtcia0_isr gpta_gtcib0_isr gpta_gtcic0_isr gpta_gtcid0_isr gpta_gtcie0_isr gpta_gtcif0_isr gpta_gdte0_isr gpta_gtciv0_isr gpta_gtciu0_isr gpta_gtcia1_isr gpta_gtcib1_isr gpta_gtcic1_isr gpta_gtcid1_isr gpta_gtcie1_isr gpta_gtcif1_isr gpta_gdte1_isr gpta_gtciv1_isr gpta_gtciu1_isr gpta_gtcia2_isr gpta_gtcib2_isr gpta_gtcic2_isr gpta_gtcid2_isr gpta_gtcie2_isr gpta_gtcif2_isr gpta_gdte2_isr gpta_gtciv2_isr gpta_gtciu2_isr gpta_gtcia3_isr gpta_gtcib3_isr gpta_gtcic3_isr gpta_gtcid3_isr gpta_gtcie3_isr gpta_gtcif3_isr gpta_gdte3_isr gpta_gtciv3_isr gpta_gtciu3_isr gpta_etgip_isr gpta_etgin_isr
static volatile struct st_gpt0 * const	pgptn[]	GPTa register address table	gpta_intr_gtcia_proc gpta_intr_gtcib_proc gpta_intr_gtcic_proc gpta_intr_gtcid_proc gpta_intr_gtcie_proc gpta_intr_gtcif_proc gpta_intr_gdte_proc gpta_intr_gtciv_proc gpta_intr_gtciu_proc
const static gpta_cmdtbl_t	gpta_cmd_tbl[]	GPTa command table	gpta_get_cmd_func
const static gpta_register_access_t	gpta_register_access_tbl[]	GPTa register access table	proc_read_reg proc_write_reg
static gpta_drv_info_t	drv_info[]	GPTa driver state	gpta_open gpta_close gpta_get_ch_info proc_set_timer_mode proc_set_clk_src proc_write_reg proc_set_buf_mode proc_set_ad_buf_mode proc_timer_start_sw

Table 6.6 Global Variables (3 / 3)

Type	Variable Name	Description	Function
static volatile struct st_gpt0 * const	pgptn[]	GPTa register address table	proc_set_count_dir proc_set_timer_mode proc_set_clk_src proc_ctrl_io proc_ctrl_io_neg proc_set_buf_mode proc_set_ad_buf_mode proc_ctrl_dtime proc_set_clear_factor proc_set_intr_skip

int32_t	interrupt_count	A variable for counting the number of compare match interrupts	main()
			cb_gpta()

6.8 Functions

Table 6.7 lists the functions to be used.

Table 6.7 Functions

Function Name	Page Number
R_POE3_Open	28
R_POE3_Close	28
R_POE3_Control	28
R_POE3_GetVersion	29
R_GPTA_Open	29
R_GPTA_Close	29
R_GPTA_Control	30
R_GPTA_GetVersion	30
main	30
cmp_match_cb	31

6.9 Specifications of Functions

6.9.1 R_POE3_Open

R_POE3_Open		
Synopsis	POE3 driver open processing	
Header	r_poe3_rzt1_if.h	
Declaration	int32_t R_POE3_Open(void)	
Description	This function opens the POE3 driver.	
Arguments	–	: –
Return values	POE3_SUCCESS	: Opened successfully
	POE3_ERROR_STATUS	: Already opened

6.9.2 R_POE3_Close

R_POE3_Close		
Synopsis	POE3 driver close processing	
Header	r_poe3_rzt1_if.h	
Declaration	int32_t R_POE3_Close(void)	
Description	This function closes the POE3 driver.	
Arguments	–	: –
Return values	POE3_SUCCESS	: Closed successfully
	POE3_ERROR_STATUS	: Already closed

6.9.3 R_POE3_Control

R_POE3_Control		
Synopsis	POE3 driver control processing	
Header	r_poe3_rzt1_if.h	
Declaration	int32_t R_POE3_Control(const poe3_cmd_t cmd, void * pBuf)	
Description	This function performs processing based on the cmd argument. For details on cmd and pbu, see 6.11, respectively. This API can be used after the R_POE3_Open function is called.	
Arguments	const poe3_cmd_t cmd	: Control type
	void * pBuf	: Parameter corresponding to the control
Return values*1	POE3_SUCCESS	: Control processing successful
	POE3_ERR_STATUS	: R_POE3_Open function not called
	POE3_ERR_INVALID_ARG	: Argument error
	POE3_ERR_INVALID_CMD	: Control type error

Note 1. For information about the return value for each processing based on cmd, see 6.11.

6.9.4 R_POE3_GetVersion

R_POE3_GetVersion		
Synopsis	POE3 driver version acquisition processing	
Header	r_poe3_rzt1_if.h	
Declaration	uint32_t R_POE3_GetVersion(void)	
Description	This function returns the version of the POE3 driver.	
Arguments	–	: –
Return values	Version	: Version of the POE3 driver Upper 16 bits: Major version Lower 16 bits: Minor version

6.9.5 R_GPTA_Open

R_GPTA_Open		
Synopsis	GPTa driver open processing	
Header	r_gpta_rzt1_if.h	
Declaration	int32_t R_GPTA_Open(const int32_t ch)	
Description	This function opens the GPTa driver. It releases the stopped state of the module.	
Arguments	int32_t ch	: Sets the channel number to be opened (0 to 3)
Return values	GPTA_SUCCESS	: Opened successfully
	GPTA_ERR_STATUS	: Already opened

6.9.6 R_GPTA_Close

R_GPTA_Close		
Synopsis	GPTa driver close processing	
Header	r_gpta_rzt1_if.h	
Declaration	int32_t R_GPTA_Close(const int32_t ch)	
Description	This function closes the GPTa driver. It causes the module to enter a stopped state.	
Arguments	int32_t ch	: Sets the channel number to be closed (0 to 3)
Return values	GPTA_SUCCESS	: Closed successfully
	GPTA_ERR_STATUS	: Already closed

6.9.7 R_GPTA_Control

R_GPTA_Control		
Synopsis	GPTa driver control processing	
Header	r_gpta_rzt1_if.h	
Declaration	int32_t R_GPTA_Control(const int32_t ch, const gpta_cmd_t cmd, void *pBuf)	
Description	This function performs processing based on the cmd argument for the channel specified by the ch argument. For details on cmd and pbuf, see 6.12. This API can be used after the R_GPTA_Open function is called.	
Arguments	const int32_t ch	: Sets the channel number to be controlled (0 to 3)
	const gpta_cmd_t cmd	: Control type
	void *pBuf	: Parameter corresponding to the control
Return values*1	GPTA_SUCCESS	: Control processing successful
	GPTA_ERR_STATUS	: R_GPTA_Open function not called
	GPTA_ERR_INVALID_ARG	: Argument error
	GPTA_ERR_INVALID_CMD	: Control type error

Note 1. For information about the return value for each processing based on cmd, see 6.11.

6.9.8 R_GPTA_GetVersion

R_GPTA_GetVersion		
Synopsis	GPTa driver version acquisition processing	
Header	r_gpta_rzt1_if.h	
Declaration	uint32_t R_GPTA_GetVersion(void)	
Description	This function returns the version of the GPTa driver.	
Arguments	–	: –
Return values	Version	: Version of the GPTa driver Upper 16 bits: Major version Lower 16 bits: Minor version

6.9.9 main

main	
Synopsis	Main processing for the sample program
Header	–
Declaration	int32_t main(void)
Description	This function sets the GTIOC0A and GTIOC0B pins as the target for high-impedance control. It sets PWM mode 1 for GPT0. (GPTa) The GTIOC0A and GTIOC0B pins are set to high impedance after 10,000 interrupts occur.
Arguments	None
Return values	"0" always returns.

6.9.10 cmp_match_cb

cmp_match_cb	
Synopsis	Callback function for GPT0 compare match interrupts
Header	–
Declaration	void cmp_match_cb(void)
Description	This function counts up the number of interrupts.
Arguments	None
Return values	None

6.10 Flowcharts

6.10.1 Main Processing

The following figures show the flowcharts of main processing.

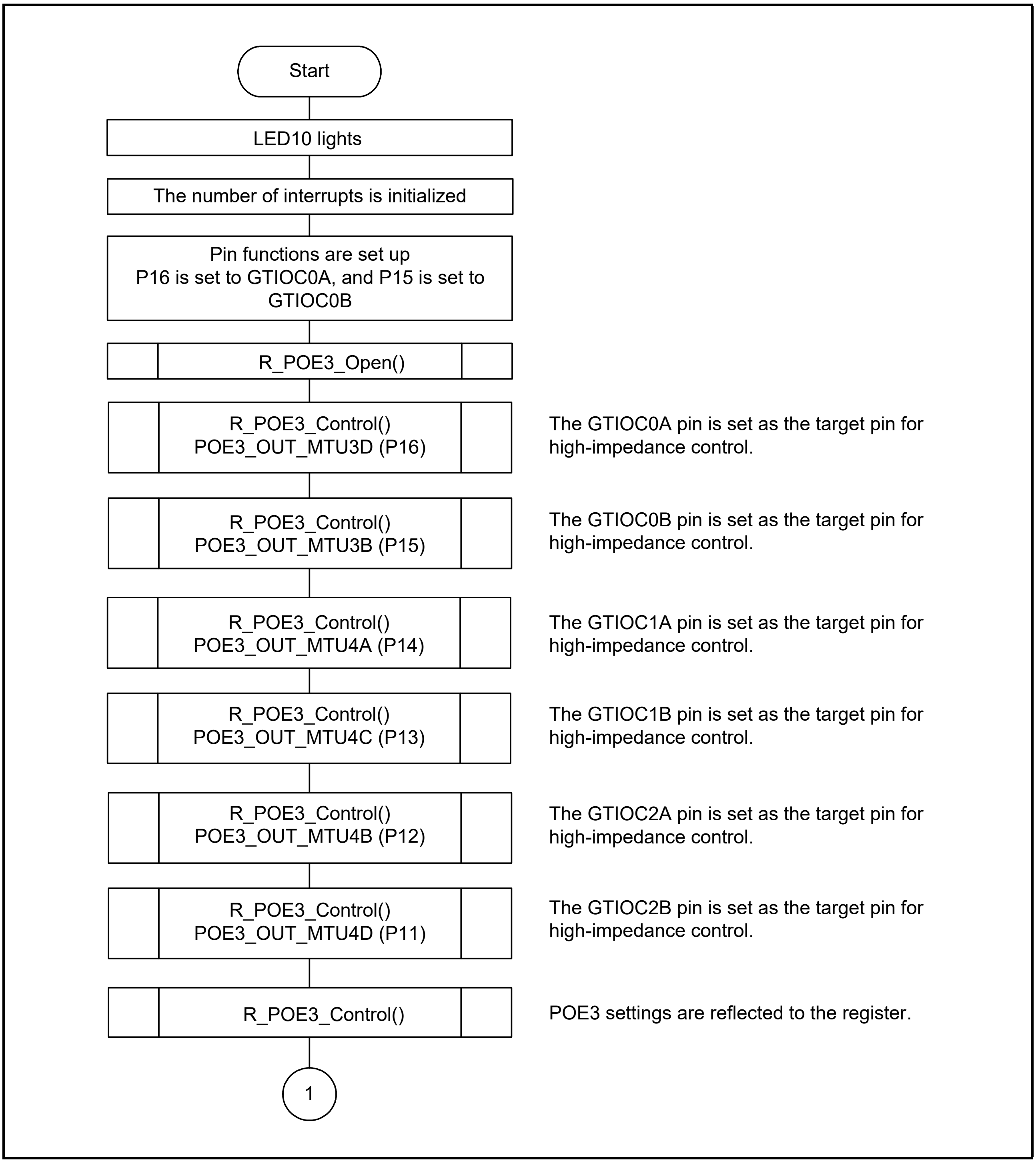

Start
LED10 lights
The number of interrupts is initialized
Pin functions are set up
P16 is set to GTIOC0A, and P15 is set to GTIOC0B
R_POE3_Open()
R_POE3_Control()
POE3_OUT_MTU3D (P16)
The GTIOC0A pin is set as the target pin for high-impedance control.
R_POE3_Control()
POE3_OUT_MTU3B (P15)
The GTIOC0B pin is set as the target pin for high-impedance control.
R_POE3_Control()
POE3_OUT_MTU4A (P14)
The GTIOC1A pin is set as the target pin for high-impedance control.
R_POE3_Control()
POE3_OUT_MTU4C (P13)
The GTIOC1B pin is set as the target pin for high-impedance control.
R_POE3_Control()
POE3_OUT_MTU4B (P12)
The GTIOC2A pin is set as the target pin for high-impedance control.
R_POE3_Control()
POE3_OUT_MTU4D (P11)
The GTIOC2B pin is set as the target pin for high-impedance control.
R_POE3_Control()
POE3 settings are reflected to the register.
1

Figure 6.9 Main Processing (1/3)

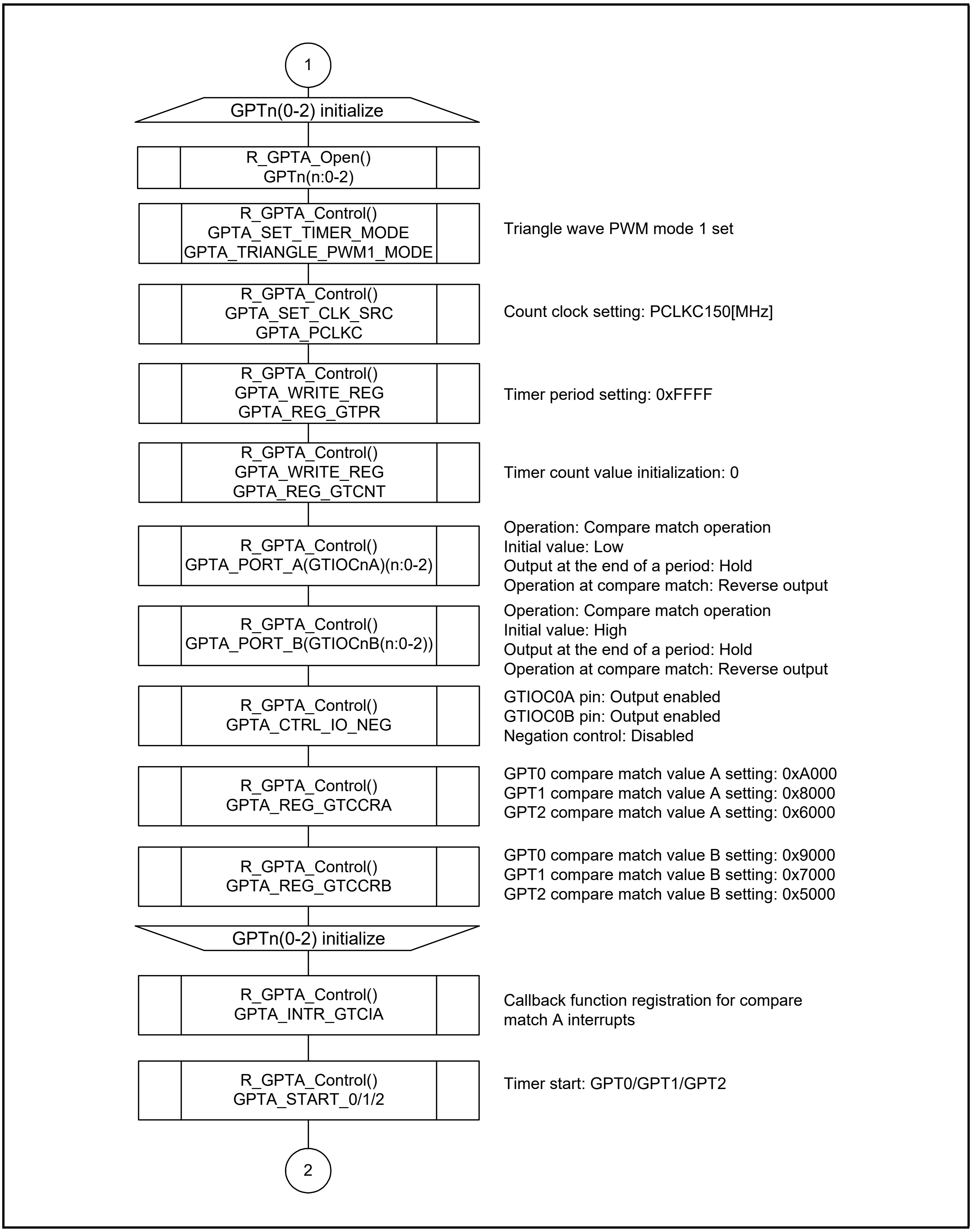
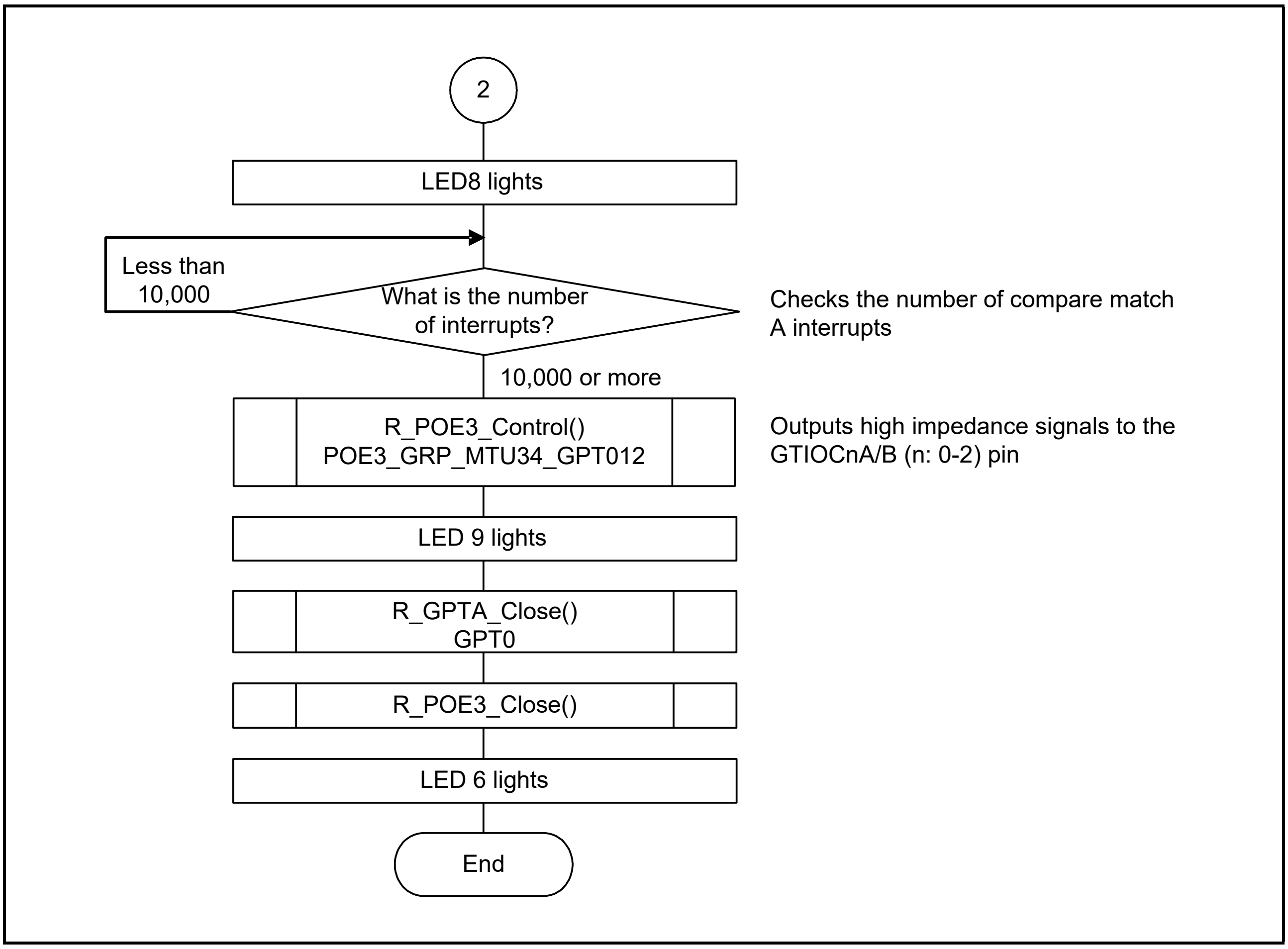

1
GPTn(0-2) initialize
R_GPTA_Open()
GPTn(n:0-2)
R_GPTA_Control()
GPTA_SET_TIMER_MODE
GPTA_TRIANGLE_PWM1_MODE
Triangle wave PWM mode 1 set
R_GPTA_Control()
GPTA_SET_CLK_SRC
GPTA_PCLKC
Count clock setting: PCLKC150[MHz]
R_GPTA_Control()
GPTA_WRITE_REG
GPTA_REG_GTPR
Timer period setting: 0xFFFF
R_GPTA_Control()
GPTA_WRITE_REG
GPTA_REG_GTCNT
Timer count value initialization: 0
R_GPTA_Control()
GPTA_PORT_A(GTIOCnA)(n:0-2)
Operation: Compare match operation
Initial value: Low
Output at the end of a period: Hold
Operation at compare match: Reverse output
R_GPTA_Control()
GPTA_PORT_B(GTIOCnB(n:0-2))
Operation: Compare match operation
Initial value: High
Output at the end of a period: Hold
Operation at compare match: Reverse output
R_GPTA_Control()
GPTA_CTRL_IO_NEG
GTIOC0A pin: Output enabled
GTIOC0B pin: Output enabled
Negation control: Disabled
R_GPTA_Control()
GPTA_REG_GTCCRA
GPT0 compare match value A setting: 0xA000
GPT1 compare match value A setting: 0x8000
GPT2 compare match value A setting: 0x6000
R_GPTA_Control()
GPTA_REG_GTCCRB
GPT0 compare match value B setting: 0x9000
GPT1 compare match value B setting: 0x7000
GPT2 compare match value B setting: 0x5000
GPTn(0-2) initialize
R_GPTA_Control()
GPTA_INTR_GTCIA
Callback function registration for compare match A interrupts
R_GPTA_Control()
GPTA_START_0/1/2
Timer start: GPT0/GPT1/GPT2
2

Figure 6.10 Main Processing (2/3)



Checks the number of compare match A interrupts

Outputs high impedance signals to the GTIOCnA/B (n: 0-2) pin

Figure 6.11 Main Processing (3/3)

6.10.2 Callback Processing

The following figure shows the flowchart of callback processing.

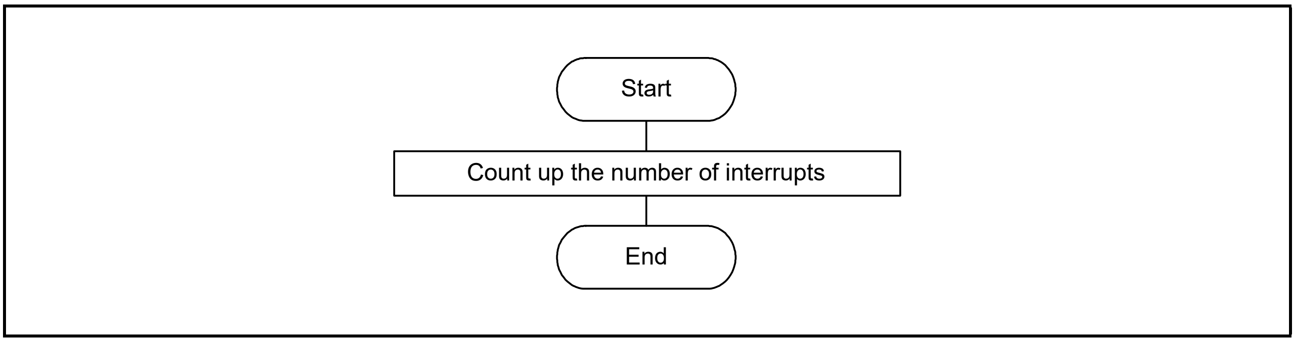


Figure 6.12 Callback Processing

6.11 R_POE3_Control Commands

The following table lists the commands for the R_POE3_Control function.

Table 6.8 R_POE3_Control Commands

Command	Outline
POE3_SELECT_PORT	Selects a port for the output pin
POE3_SET_INPUT_MODE	Specifies request reception conditions for the input port to which high impedance signals are to be output
POE3_SET_OSC_MODE	Specifies whether to output high impedance signals when an oscillation error is detected
POE3_SET_ALV	Sets the active level of the output pin
POE3_DETECT_SHORT	Specifies the pins for which short circuits are to be detected
POE3_CTRL_HIZ	Specifies high-impedance output conditions for each output pin
POE3_CTRL_HIZ_EX	Extends high-impedance output conditions for each output pin
POE3_REFLECT_REG	Controls high-impedance output for pins by software
POE3_SOFT_HIZ	Sets the callback function
POE3_SET_CB	Reflects the settings specified in each command to the POE3 register

6.11.1 POE3_SELECT_PORT

POE3_SELECT_PORT

Synopsis: Selecting a port for the output pin

Header: r_poe3_if.h

Description: This command selects a port for the output pin that is set to high impedance by using the POE3 function.
This pin must also be separately set in the register for the Multi-function Pin Controller (MPC).
Note that the pin set in the register for POE3 and the pin set in the register for the MPC are not detached.
The settings of this command are reflected in the register when the command is issued.
Parameters are transferred in the form of poe3_select_port_t * type variables.

Parameters:

poe3_ioport outport	Specifies the output pin to be set.
uint16_t port_select	Specifies the port for which high impedance control is to be performed.

Return values:

POE3_ERR_INVALID_ARG	: Invalid argument
POE3_ERR_ALREADY_REFLECTED	: Command already issued

Remarks: The values that can be set in the "port_select" parameter are predetermined for each output pin specified in "port".

Output pin name : Actual pin name
POE3_OUT_GPT3A: POE3_GPT3A_PA6 or POE3_GPT3A_P66
POE3_OUT_GPT3B: POE3_GPT3B_PA7 or POE3_GPT3B_P67
POE3_OUT_MTU0A: POE3_MTU0A_PE6 or POE3_MTU0A_PP4
POE3_OUT_MTU0B: POE3_MTU0B_PP3 or POE3_MTU0B_PE4
POE3_OUT_MTU0C: POE3_MTU0C_PE5 or POE3_MTU0C_PP2
POE3_OUT_MTU0D: POE3_MTU0D_PE3 or POE3_MTU0D_PP1
POE3_OUT_MTU3B: POE3_MTU3B_P16 or POE3_MTU3B_PB7 or POE3_MTU3B_PA2
POE3_OUT_MTU3D: POE3_MTU3D_P15 or POE3_MTU3D_PF6 or POE3_MTU3D_PA1
POE3_OUT_MTU4A: POE3_MTU4A_PA0 or POE3_MTU4A_PF5 or POE3_MTU4A_P14
POE3_OUT_MTU4C: POE3_MTU4C_P77 or POE3_MTU4C_P87 or POE3_MTU4C_P13
POE3_OUT_MTU4B: POE3_MTU4B_P86 or POE3_MTU4B_P76 or POE3_MTU4B_P12
POE3_OUT_MTU4D: POE3_MTU4D_P75 or POE3_MTU4D_PD7 or POE3_MTU4D_P11

Example: Setting GPT3-A (GTIOC3A) to PA6
outport←POE3_OUT_GPT3A, port_select←POE3_GPT3A_PA6

If this command is not called, the following default values will be used:
POE3_OUT_GPT3A: POE3_GPT3A_PA6
POE3_OUT_GPT3B: POE3_GPT3B_PA7
POE3_OUT_MTU0A: POE3_MTU0A_PE6
POE3_OUT_MTU0B: POE3_MTU0B_PP3
POE3_OUT_MTU0C: POE3_MTU0C_PE5
POE3_OUT_MTU0D: POE3_MTU0D_PE3
POE3_OUT_MTU3B: POE3_MTU3B_PA2
POE3_OUT_MTU3D: POE3_MTU3D_PA1
POE3_OUT_MTU4A: POE3_MTU4A_P14
POE3_OUT_MTU4C: POE3_MTU4C_P13
POE3_OUT_MTU4B: POE3_MTU4B_P12
POE3_OUT_MTU4D: POE3_MTU4D_P11

6.11.2 POE3_SET_INPUT_MODE

POE3_SET_INPUT_MODE

Synopsis: Specifying request reception conditions for the input port to which high impedance signals are to be output

Header: r_poe3_if.h

Description: This command sets the request reception conditions specified in "mode" for the port specified in "port".
The settings of this command are reflected in the register when the command is issued.
Parameters are transferred in the form of poe3_set_input_t * type variables.

Parameters	poe3_ioport_t inport	Sets an input port
	uint8_t mode	Sets an input mode
	poe3_flag_t intr_flag	Sets the interrupt-enabled flag
Return values	POE3_ERR_INVALID_ARG	: Invalid argument
	POE3_ERR_ALREADY_REFLECTED	: Command already issued

Remarks: The following values can be set in the "inport" parameter:
POE3_IN_POE0: Specifies the request reception conditions for the POE0# port
POE3_IN_POE4: Specifies the request reception conditions for the POE4# port
POE3_IN_POE8: Specifies the request reception conditions for the POE8# port
POE3_IN_POE10: Specifies the request reception conditions for the POE10# port

The following values can be set in the "mode" parameter:
POE3_INPUT_MODE_0:
Accepts the request at the falling edge of an input signal to the specified port
POE3_INPUT_MODE_4:
Accepts the request when a low-level input signal to the specified port is sampled every four PCLKD clock signals and a total of 16 times, and if all the signals are at low level.
POE3_INPUT_MODE_16:
Accepts the request when a low-level input signal to the specified port is sampled every 16 PCLKD clock signals and a total of 16 times, and if all the signals are at low level.
POE3_INPUT_MODE_128:
Accepts the request when a low-level input signal to the specified port is sampled every 128 PCLKD clock signals and a total of 16 times, and if all the signals are at low level.

The following values can be set in the "intr_flag" parameter:
Even if POE3_ENABLE is specified in this parameter, no interrupt will be generated unless callback is registered by using POE3_SET_CB (6.11.10).
POE3_DISABLE: Disables an interrupt request when a request is received
POE3_ENABLE: Enables an interrupt request when a request is received

If this command is not called, the following default values will be used:
mode: POE3_INPUT_MODE_0
intr_flag: POE3_DISABLE

6.11.3 POE3_SET_OSC_MODE

POE3_SET_OSC_MODE

Synopsis	Specifying whether to output high impedance signals when an oscillation error is detected	
Header	r_poe3_if.h	
Description	This command specifies whether to set the MTU complementary PWM output pin, MTU0 pin, and GPT pin to high impedance when oscillation stoppage or a PLL oscillation error (CLMA0/1) is detected. The settings of this command are reflected in the register when the command is issued. Parameters are transferred in the form of poe3_set_osc_t * type variables.	
Parameters	poe3_flag_t hiz_flag	Specifies whether to output high impedance signals when an OSC error occurs.
Return values	POE3_ERR_INVALID_ARG POE3_ERR_ALREADY_REFLECTED	: Invalid argument : Command already issued
Remarks	The following values can be set in the "hiz_flag" parameter: POE3_DISABLE: Does not set the MTU complementary PWM output pin, MTU0 pin, or GPT pin to high impedance when oscillation stoppage or a PLL oscillation error (CLMA0/1) is detected POE3_ENABLE: Sets the MTU complementary PWM output pin, MTU0 pin, and GPT pin to high impedance when oscillation stoppage or a PLL oscillation error (CLMA0/1) is detected If this command is not called, the following default value will be used: hiz_flag: POE3_DISABLE	

6.11.4 POE3_SET_ALV

POE3_SET_ALV

Synopsis Setting the active level of the output pin

Header r_poe3_if.h

Description This command sets the active level of the pin specified in "outport". (This is used to detect short circuits.)
The pins that cannot be set in this command (MTU6 and MTU7) follow the settings for MTU3a.
The settings of this command are reflected in the register when the command is issued.
Parameters are transferred in the form of poe3_set_alv_t type variables.

Parameters	poe3_ioport_t outport	Specifies the pin for which the active level is to be set
	uint8_t active_level	Specifies the active level
Return values	POE3_ERR_INVALID_ARG	: Invalid argument
	POE3_ERR_ALREADY_REFLECTED	: Command already issued

Remarks The following values can be set in the "outport" parameter:
POE3_OUT_MTU3B / POE3_OUT_GPT0A*1
POE3_OUT_MTU3D / POE3_OUT_GPT0B*1
POE3_OUT_MTU4A / POE3_OUT_GPT1A*1
POE3_OUT_MTU4C / POE3_OUT_GPT1B*1
POE3_OUT_MTU4B / POE3_OUT_GPT2A*1
POE3_OUT_MTU4D / POE3_OUT_GPT2B*1
Note 1. These are double-duty pins and so actually the same pins.

The following values can be set in the "active_level" parameter:
POE3_ALV_LOW: Sets the active level to low
POE3_ALV_HIGH: Sets the active level to high

If this command is not called, the following default value will be used:
acteive_level: POE3_DISABLE

6.11.5 POE3_DETECT_SHORT

POE3_DETECT_SHORT

Synopsis	Specifying the pins for which short circuits are to be detected	
Header	r_poe3_if.h	
Description	This command detects short circuits for the pins specified in the argument. If a short circuit is detected, the pins will be set to high impedance. Because this API needs to have short-circuit detection set for each group, to enable all short-circuit detections, this API must be called twice with POE3_MTU32_GPTA012 and POE3_MTU67 specified in the "group" argument. The settings of this command are reflected in the register when the command is issued. Parameters are transferred in the form of poe3_detect_short_t type variables.	
Parameters	poe3_port_grp_t group	Specifies the group for which short circuits are to be detected
	poe3_flag_t hiz_flag	Specifies whether to output high impedance signals when a short circuit is detected
	poe3_flag_t intr_flag	Sets the interrupt-enabled flag
Return values	POE3_ERR_INVALID_ARG	: Invalid argument
	POE3_ERR_ALREADY_REFLECTED	: Command already issued

Remarks

The following values can be set in the "group" parameter:

POE3_MTU34_GPT012:

Detects whether even one of the three pairs of two-phase outputs (to be compared) from the following pins are simultaneously set at the active level: MTU complementary PWM output pins (MTU3 and MTU4 pins) or GPT output pins (GPT0 to GPT2 pins).

Setting this value detects short circuits for the following pin combinations:

MTIOC3B and MTIOC3D
MTIOC4A and MTIOC4B
MTIOC4C and MTIOC4D
GTIOC0A and GTIOC0B
GTIOC1A and GTIOC1B
GTIOC2A and GTIOC2B

POE3_MTU67:

Detects whether even one of the three pairs of two-phase outputs (to be compared) from the following pins are simultaneously set at the active level: MTU complementary PWM output pins (MTU6 and MTU7 pins).

Setting this value detects short circuits for the following pin combinations:

MTIOC6B and MTIOC6D
MTIOC7A and MTIOC7B
MTIOC7C and MTIOC7D

The following values can be set in the "hiz_flag" parameter:

POE3_DISABLE: Does not output high impedance signals when a short circuit is detected
POE3_ENABLE: Outputs high impedance signals when a short circuit is detected

The following values can be set in the "intr_flag" parameter:
Even if POE3_ENABLE is specified in this parameter, no interrupt will be generated unless callback is registered by using POE3_SET_CB (6.11.10).

POE3_DISABLE: Disables an interrupt request when a short circuit is detected
POE3_ENABLE: Enables an interrupt request when a short circuit is detected

If this command is not called, the following default values will be used:

hiz_flag: POE3_DISABLE
intr_flag: POE3_ENABLE

6.11.6 POE3_CTRL_HIZ

POE3_CTRL_HIZ

Synopsis Controlling high-impedance output for each output pin

Header r_poe3_if.h

Description This command specifies whether to output high impedance signals to the pins specified in the argument when the specified conditions are met.
The settings of this command are reflected in the register when the command is issued.
Parameters are transferred in the form of poe3_ctrl_hiz_t type variables.

Parameters

poe3_hiz_grp_t group	Specifies the pins for which high-impedance output control is to be performed
poe3_flag_t hiz_flag	Specifies whether to perform high-impedance output control

Return values

POE3_ERR_INVALID_ARG	: Invalid argument
POE3_ERR_ALREADY_REFLECTED	: Command already issued

Remarks The following values can be set in the "group" parameter:
The values enclosed in parentheses are high-impedance output pins.

POE3_HIZ_MTU0A: (MTIOC0A)
POE3_HIZ_MTU0B: (MTIOC0B)
POE3_HIZ_MTU0C: (MTIOC0C)
POE3_HIZ_MTU0D: (MTIOC0D)

High-impedance output conditions

- If the conditions specified in "POE3_IN_POE8" in the command are met
- If POE3_GRP_MTU0 is specified in the command
- If POE3_ENABLE is specified in the command, and oscillation stoppage or an oscillation error occurs
- If the high-impedance output conditions specified in "POE3_MTU0" in the command are met

POE3_HIZ_MTU3BD_GPT0AB: (MTIOC3B, MTIOC3D, GTIOC0A, GTIOC0B)
POE3_HIZ_MTU4AC_GPT1AB: (MTIOC4A, MTIOC4C, GTIOC1A, GTIOC1B)
POE3_HIZ_MTU4BD_GPT2AB: (MTIOC4B, MTIOC4D, GTIOC2A, GTIOC2B)

High-impedance output conditions

- If a short circuit occurs on the MTU complementary PWM output pins (MTU3 and MTU4 pins) or GPT output pins (GPT0 to GPT2 pins) with POE3_GRP_MTU34_GPT012 enabled in the command
- If the conditions specified in "POE3_IN_POE0" in the command are met
- If POE3_GRP_MTU34_GPT012 is specified in the command
- If POE3_ENABLE is specified in the command, and oscillation stoppage or an oscillation error occurs
If the high-impedance output conditions specified in "POE3_MTU34_GPT012" in the command are met

POE3_HIZ_MTU6BD: (MTIOC6B, MTIOC6D)
POE3_HIZ_MTU7AC: (MTIOC7A, MTIOC7C)
POE3_HIZ_MTU7BD: (MTIOC7B, MTIOC7D)

High-impedance output conditions

- If a short circuit occurs on the MTU complementary PWM output pins (MTU6 and MTU7 pins) with POE3_GRP_MTU67 enabled in the command
- If the conditions specified in "POE3_IN_POE4" in the command are met
- If POE3_GRP_MTU67 is specified in the command
- If POE3_ENABLE is specified in the command, and oscillation stoppage or an oscillation error occurs
- If the high-impedance output conditions specified in "POE3_MTU67" in the command are met

POE3_HIZ_GPT3AB:

High-impedance output conditions

- If the conditions specified in "POE3_IN_POE10" in the command are met
- If POE3_GRP_GPT3 is specified in the command
- If POE3_ENABLE is specified in the command, and oscillation stoppage or an oscillation error occurs
- If the high-impedance output conditions specified in "POE3_GPT3" in the command are met

The following values can be set in the "hiz_flag" parameter:

POE3_ENABLE

Outputs high impedance signals to the applicable pins when even one of the conditions for the output pins (group) specified in "group" is met.

POE3_DISABLE

Does not output high impedance signals

If this command is not called, the following default values will be used:
(The default value differs according to the value specified in "group".)
group:
POE3_HIZ_MTU0A
POE3_HIZ_MTU0B
POE3_HIZ_MTU0C
POE3_HIZ_MTU0D
hiz_flag: POE3_DISABLE

group:
POE3_HIZ_MTU3BD_GPT0AB
POE3_HIZ_MTU4AC_GPT1AB
POE3_HIZ_MTU4BD_GPT2AB
POE3_HIZ_MTU6BD
POE3_HIZ_MTU7AC
POE3_HIZ_MTU7BD
POE3_HIZ_GPT3AB
hiz_flag: POE3_ENABLE

6.11.7 POE3_CTRL_HIZ_EX

POE3_CTRL_HIZ_EX

Synopsis	Extending high-impedance output conditions for each output pin	
Header	r_poe3_if.h	
Description	This command specifies whether to output high impedance signals to the pins specified in the argument when the specified conditions are met. The settings of this command are reflected in the register when the command is issued. Parameters are transferred in the form of poe3_ctrl_hiz_ex_t type variables.	
Parameters	poe3_port_grp_t group	Specifies the pins for which high-impedance output conditions are to be extended
	poe3_ioport_t port	Specifies the input port to be used as the high-impedance output condition.
	poe3_flag_t hiz_flag	Specifies whether to add to the high-impedance output conditions.
Return values	POE3_ERR_INVALID_ARG	: Invalid argument
	POE3_ERR_ALREADY_REFLECTED	: Command already issued

Remarks

The following values can be set in the "group" parameter.
The values that can be specified in the "port" parameter vary according to the value specified in "group".

POE3_MTU34_GPT012:
Input ports ("port") that can be set as high-impedance output conditions:
- POE3_IN_POE4
- POE3_IN_POE8
- POE3_IN_POE10

POE3_MTU67:
Input ports ("port") that can be set as high-impedance output conditions:
- POE3_IN_POE0
- POE3_IN_POE8
- POE3_IN_POE10

POE3_MTU0:
Input ports ("port") that can be set as high-impedance output conditions:
- POE3_IN_POE0
- POE3_IN_POE4
- POE3_IN_POE10

POE3_GPT3:
Input ports ("port") that can be set as high-impedance output conditions:
- POE3_IN_POE0
- POE3_IN_POE4
- POE3_IN_POE8
- POE3_IN_POE10

The following values can be set in the "hiz_flag" parameter:

POE3_DISABLE
Deletes the conditions for outputting high impedance signals to the pins in the specified group

POE3_ENABLE
Adds to the conditions for outputting high impedance signals to the pins in the specified group

Multiple ports can be set for a single group as the high impedance output conditions.

If this command is not called, the following default value will be used for the "port" settings for all "group" settings:
hiz_flag: POE3_DISABLE

6.11.8 POE3_REFLECT_REG

POE3_REFLECT_REG		
Synopsis	Reflecting the settings specified in each command to the POE3 register	
Header	r_poe3_if.h	
Description	This command reflects the settings specified in the following commands to the POE3 register: POE3_SELECT_PORT POE3_SET_INPUT_MODE POE3_SET_OSC_MODE POE3_SET_ALV POE3_DETECT_SHORT POE3_CTRL_HIZ POE3_CTRL_HIZ_EX This command can be used only once. Once these five commands are specified, they cannot be changed until RZ/T1 is reset. Parameters for this command are ignored.	
Parameters	None	–
Return values	POE3_ERR_ALREADY_REFLECTED	: Command already issued
Remarks	Before issuing this command, specify the necessary settings in the above commands (POE3_SELECT_PORT, POE3_SET_INPUT_MODE, POE3_SET_OSC_MODE, POE3_SET_ALV, POE3_DETECT_SHORT, POE3_CTRL_HIZ, POE3_CTRL_HIZ_EX). If these commands are issued after this command, processing will fail. (POE3_ERR_ALREADY_REFLECTED will return as the return value.)	

6.11.9 POE3_SOFT_HIZ

POE3_SOFT_HIZ

Synopsis	Controlling high-impedance output for pins by software	
Header	r_poe3_if.h	
Description	This command outputs high impedance signals to the pin group specified in the argument. Parameters are transferred in the form of poe3_soft_hiz_t type variables.	
Parameters	poe3_port_grp_t group	Specifies the group for which high-impedance output control is to be performed
	poe3_flag_t hiz_flag	Specifies whether to perform high-impedance output
Return values	POE3_ERR_INVALID_ARG	: Invalid argument

Remarks

The following values can be set in the "group" parameter:

POE3_MTU34_GPT012:

Specifies the MTU complementary PWM output pins (MTIOC3B, MTIOC3D, MTIOC4A, MTIOC4B, MTIOC4C, and MTIOC4D pins) or GPT output pins (GTIOC0A, GTIOC0B, GTIOC1A, GTIOC1B, GTIOC2A, and GTIOC2B pins)

POE3_MTU67:

Specifies the MTU complementary PWM output pins (MTIOC6B, MTIOC6D, MTIOC7A, MTIOC7B, MTIOC7C, MTIOC7D).

POE3_MTU0:

Specifies the MTU0 pins (MTIOC0A, MTIOC0B, MTIOC0C, MTIOC0D).

POE3_GPT3:

Specifies the GPT3 pins (GTIOC3A, GTIOC3B).

The following values can be set in the "hiz_flag" parameter:

POE3_DISABLE:

Does not output high impedance signals to the pins in the specified group

POE3_ENABLE:

Outputs high impedance signals to the pins in the specified group

If this command is not called, the following default value will be used for all "group" settings:

hiz_flag: POE3_DISABLE

6.11.10 POE3_SET_CB

POE3_SET_CB

Synopsis Registering the callback function to be called when an interrupt occurs

Header r_poe3_if.h

Description This command registers the callback function to be called when the interrupt specified in the argument occurs.
Parameters are transferred in the form of poe3_set_cb_t type variables.

Parameters	poe3_intr_t intr_type	Specifies an interrupt type
	poe3_cb callback	Specifies a callback function
	uint8_t intr_pri	Specifies the interrupt priority level
Return values	POE3_ERR_INVALID_ARG	: Invalid argument

Remarks The following values can be set in the "intr_type" parameter:
POE3_INTR_POE0:
POE0# high-impedance request or output short-circuit (MTU ch 3, 4 or GPT ch 0, 1, 2) interrupt (OEI1)
POE3_INTR_POE4:
POE4# high-impedance request or output short-circuit (MTU ch 6, 7) interrupt (OEI2)
POE3_INTR_POE8:
POE8# high-impedance request interrupt (OEI3)
POE3_INTR_POE10:
POE10# high-impedance request interrupt (OEI4)

The following values can be set in the "callback" parameter.
For other than NULL: Sets the entry address for the callback function
NULL: Cancels the registration of the callback function
The following values are set to the arguments of the callback function to be called when an interrupt is notified:
First argument
0: POE0, 4, 8, 10# high-impedance request interrupts do not occur
1: POE0, 4, 8, 10# high-impedance request interrupts occur
Second argument
0: Output short-circuit interrupts do not occur
1: Output short-circuit interrupts occur

The following values can be set in the "intr_pri" argument:
0~15: Interrupt priority level (0 is the highest priority)

If this command is not called, the following default values will be used for all "intr_type" settings:
callback: NULL(interrupt disabled)
intr_pri: 0

6.12 R_GPTA_Control Commands

The following table lists the commands for the R_GPTA_Control function.

Table 6.9 R_GPTA_Control Commands

Command	Outline
GPTA_SET_COUNT_DIR	Specifies the timer counting direction
GPTA_SET_TIMER_MODE	Sets the timer mode
GPTA_SET_CLK_SRC	Sets the clock source for the timer
GPTA_CTRL_IO	Controls I/O port output
GPTA_CTRL_IO_NEG	Performs negation control for the output pin
GPTA_READ_REG	Reads from the GPTa register
GPTA_WRITE_REG	Writes to the GPTa register
GPTA_SET_BUF_MODE	Sets buffer operations
GPTA_SET_AD_BUF_MODE	Sets buffer operations for the A/D conversion start request timing register
GPTA_SET_CB	Registers the callback function to be called when an interrupt occurs
GPTA_TIMER_START_SW	Starts the counter
GPTA_CTRL_DTIME	Controls the dead time
GPTA_SET_HW_FACTOR	Sets a hardware source that starts, stops, and clears the counter
GPTA_SET_CLEAR_FACTOR	Sets a source that clears the counter
GPTA_SET_SYNC	Sets a source that clears the counter synchronously
GPTA_SET_NF	Sets the noise filter
GPTA_SET_NF_EX	Sets the noise filter for the external trigger input pin
GPTA_SET_INTR_SKIP	Sets the interrupt skip function

6.12.1 GPTA_SET_COUNT_DIR

GPTA_SET_COUNT_DIR		
Synopsis	Specifying the timer counting direction	
Header	r_gpta_if.h	
Description	• For sawtooth waves If GPTA_COUNT_DOWN is set in "ud" during count-up operations, the counting direction will switch when an overflow occurs. If GPTA_COUNT_UP is set in "ud" during count-down operations, the counting direction will switch when an underflow occurs. If the "ud" setting is changed from GPTA_COUNT_UP to GPTA_COUNT_DOWN with udf_flag set to GPTA_DISABLE when the counting operation is stopped, the counting direction will be count-up first and then switched to count-down when an overflow occurs. If the "ud" setting is changed from GPTA_COUNT_DOWN to GPTA_COUNT_UP with udf_flag set to GPTA_DISABLE when the counting operation is stopped, the counting direction will be count-down first and then switched to count-up when an underflow occurs. If udf_flag is set to GPTA_ENABLE when the counting operation is stopped, the counting direction will be based on the "ud" setting used when counting started, • For triangle waves Even if the "ud" value is changed during counting operations, the counting direction will not change. Even if the "ud" value is changed with udf_flag set to GPTA_DISABLE when the counting operation is stopped, the counting direction will not change after counting starts. If udf_flag is set to GPTA_ENABLE when the counting operation is stopped, the current setting of "ud" will determine the counting direction when counting starts, Use the command to switch between sawtooth waves and triangle waves. Parameters are transferred in the form of gpta_set_count_dir_t * type variables.	
Parameters	uint8_t ud	Sets the counting direction
	gpta_flag_t udf_flag	Forcibly sets the counting direction
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument
Remarks	The following values can be set in the "ud" parameter: GPTA_COUNT_DOWN: Sets the counting direction to count-down GPTA_COUNT_UP: Sets the counting direction to count-up The following values can be set in the "udf_flag" parameter: GPTA_DISABLE: Does not forcibly set the counting direction GPTA_ENABLE: Forcibly sets the counting direction If this command is not called, the following default values will be used: ud: GPTA_COUNT_UP udf_flag: GPTA_DISABLE	

6.12.2 GPTA_SET_TIMER_MODE

GPTA_SET_TIMER_MODE

Synopsis Setting the timer mode

Header r_gpta_if.h

Description This command sets the timer operating mode.
This command can be used only when the timer is stopped.
Parameters are transferred in the form of gpta_set_timer_mode_t * type variables.

Parameters	uint8_t mode	Sets the timer operating mode
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument
	GPTA_ERR_TIMER_STARTED	: Timer operating

Remarks The following values can be set in the "mode" parameter:

GPTA_SAW_PWM_MODE:
Sawtooth wave PWM mode (single or double buffers can be used)

GPTA_SAW_ONE_SHOT_MODE:
Sawtooth wave one-shot pulse mode (buffer operation is fixed)

GPTA_TRIANGLE_PWM_MODE1:
Triangle wave PWM mode 1 (valleys, 16-bit transfer) (single or double buffers can be used)

GPTA_TRIANGLE_PWM_MODE2:
Triangle wave PWM mode 2 (peaks/valleys, 16-bit transfer) (single or double buffers can be used)

GPTA_TRIANGLE_PWM_MODE3:
Triangle wave PWM mode 3 (valleys, 32-bit transfer) (buffer operation is fixed)

If this command is not called, the following default value will be used:
mode: GPTA_SAW_PWM_MODE

6.12.3 GPTA_SET_CLK_SRC

GPTA_SET_CLK_SRC

Synopsis	Setting the clock source for the timer	
Header	r_gpta_if.h	
Description	This command sets the clock source for the timer. This command can be used only when the timer is stopped. Parameters are transferred in the form of gpta_set_clk_src_t * type variables.	
Parameters	uint8_t clock	Sets the clock for the timer
Return values	GPTA_ERR_INVALID_ARG GPTA_ERR_TIMER_STARTED	: Invalid argument : Timer operating

Remarks

The following values can be set in the "clock" parameter:

GPTA_PCLKC_DIV_1:
PCLKC (system clock)
GPTA_PCLKC_DIV_2:
PCLKC / 2 (system clock / 2)
GPTA_PCLKC_DIV_4:
PCLKC / 4 (system clock / 4)
GPTA_PCLKC_DIV_8:
PCLKC / 8 (system clock / 8)

If this command is not called, the following default value will be used:
clock: GPTA_PCLKC_DIV_1

6.12.4 GPTA_CTRL_IO

GPTA_CTRL_IO

Synopsis	Controlling I/O port output	
Header	r_gpta_if.h	
Description	This command controls the function of the GTIOCnA/B(n=0-3) pin. Parameters are transferred in the form of gpta_ctrl_io_t * type variables.	
Parameters	uint8_t port	Sets the port to be controlled (A or B)
	uint8_t io_mode	Sets compare match or input capture operation
	uint8_t io_default	Sets initial output
	uint8_t io_hold	Sets the output to be produced at the end of the timer period
	uint8_t io_output	Sets the output to be produced when compare match occurs
	uint8_t dflt	Sets the output to be produced when counting stops
	uint8_t hld	Sets the output to be produced when counting starts or stops
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument

Remarks

The following values can be set in the "port" parameter:
- GPTA_PORT_A:
  - Sets the GTIOCnA pin (n=0-3)
  - "n" is specified in the "ch" argument for the R_GPTA_Control function.
- GPTA_PORT_B:
  - Sets the GTIOCnB pin (n=0-3)

The following values can be set in the "io_mode" parameter:
- GPTA_CMP_MATCH:
  - Controls the output of the GTIOCnA/B pin by compare match (n=0-3)
- GPTA_INPUT_CAPTURE:
  - Sets the GTIOCnA/B pin as the trigger for input capture (n=0-3)

The following values can be set in the "io_default" parameter.
"io_default" is valid only when "io_mode" is set to GPTA_CMP_MATCH.
This parameter is ignored when "io_mode" is set to GPTA_INPUT_CAPTURE.
- GPTA_IO_LOW:
  - Sets the initial output of the GTIOCnA/B pin to low (n=0-3)
- GPTA_IO_HIGH:
  - Sets the initial output of the GTIOCnA/B pin to high (n=0-3)

The following values can be set in the "io_hold" parameter.
"io_hold" is valid only when "io_mode" is set to GPTA_CMP_MATCH.
This parameter is ignored when "io_mode" is set to GPTA_INPUT_CAPTURE.
- GPTA_IO_HOLD_OUTPUT:
  - Holds the output of the GTIOCnA/B pin at the end of a period (n=0-3)
- GPTA_IO_HOLD_LOW:
  - Sets the output of the GTIOCnA/B pin to low at the end of a period (n=0-3)
- GPTA_IO_HOLD_HIGH:
  - Sets the output of the GTIOCnA/B pin to high at the end of a period (n=0-3)
- GPTA_IO_HOLD_TOGGLE:
  - Toggles the output of the GTIOCnA/B pin at the end of a period (n=0-3)

The following values can be set in the "io_output" parameter.
The values that can be set in "io_output" differ depending on whether "io_mode" is set to GPTA_CMP_MATCH or GPTA_INPUT_CAPTURE.

io_mode: GPTA_IO_CMP_MATCH
  GPTA_IO_CMP_OUTPUT:
    Holds the output of the GTIOCnA/B pin when compare match occurs (n=0-3)
  GPTA_IO_CMP_LOW:
    Sets the output of the GTIOCnA/B pin to low when compare match occurs (n=0-3)
  GPTA_IO_CMP_HIGH:
    Sets the output of the GTIOCnA/B pin to high when compare match occurs (n=0-3)
  GPTA_IO_CMP_TOGGLE:
    Toggles the output of the GTIOCnA/B pin when compare match occurs (n=0-3)

io_mode: GPTA_IO_INPUT_CAPTURE
  GPTA_IO_TRG_RISING:
    Performs input capture at a rising edge of a clock pulse on the GTIOCnA/B pin (n=0-3)
  GPTA_IO_TRG_FALLING:
    Performs input capture at a falling edge of a clock pulse on the GTIOCnA/B pin (n = 0-3)
  GPTA_IO_TRG_BOTH:
    Performs input capture at both edges of a clock pulse on the GTIOCnA/B pin (n = 0-3)

The following values can be set in the "dflt" parameter:
  GPTA_DFLT_LOW:
    Sets the output of the GTIOCnA/B pin to low when counting stops (n = 0-3)
  GPTA_DFLT_HIGH:
    Sets the output of the GTIOCnA/B pin to high when counting stops (n = 0-3)

The following values can be set in the "hld" parameter:
  GPTA_HLD_REG:
    Outputs the value specified in "io_default" to the GTIOCnA/B pin when counting starts (n = 0-3)
    Outputs the value specified in "dflt" to the GTIOCnA/B pin when counting stops (n = 0-3)
    Reflects the value of "dflt" to the GTIOCnA/B pin immediately if the value is changed when counting is stopped (n = 0-3)
  GPTA_HLD_HOLD:
    Holds the output of the GTIOCnA/B pin when counting starts or stops (n = 0-3)

If this command is not called, the following default values will be used:
  io_mode: GPTA_CMP_MATCH
  io_default: GPTA_IO_LOW
  io_hold: GPTA_IO_HOLD_OUTPUT
  io_output: GPTA_IO_CMP_OUTPUT
  dflt: GPTA_DFLT_LOW
  hld: GPTA_HLD_REG

Caution "The end of a period" means an overflow or underflow for sawtooth waves and a valley for triangle waves.
If the end of a period coincides with the timing of compare match when compare match occurs, the setting of "io_hold" takes precedence in sawtooth wave PWM mode, and the setting of "io_output" takes precedence in other modes.
Even if compare match is only set in this command, nothing will be output to the GTIOCnA/B pin.
It is necessary to set up output negation control separately by using the command.

6.12.5 GPTA_CTRL_IO_NEG

GPTA_CTRL_IO_NEG

Synopsis	Performing negation control for the output pin	
Header	r_gpta_if.h	
Description	This command performs output negation control for the GTIOCnA/B pin. (n = 0-3) Parameters are transferred in the form of gpta_ctrl_io_neg_t * type variables.	
Parameters	uint16_t control_a	Specifies whether to perform negation control for the GTIOCnA pin
	uint16_t control_b	Specifies whether to perform negation control for the GTIOCnB pin
	uint16_t output_a	Sets the value to be output to the GTIOCnA pin when negation control is performed
	uint16_t output_b	Sets the value to be output to the GTIOCnB pin when negation control is performed
	uint16_t factor	Sets the negation output source
	uint16_t polarity_select	Selects the polarity of the negation source
	uint16_t soft_control	Specifies whether to perform negation control by software
	uint16_t flag_a	Specifies whether to output from the GTIOCnA pin
	uint16_t flag_b	Specifies whether to output from the GTIOCnB pin
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument

Remarks

The following values can be set in the "control_a" parameter:
- GPTA_NEG_CTRL_A_DISABLE:
  - Disables negation control
- GPTA_NEG_CTRL_A_ENABLE:
  - Enables negation control

The following values can be set in the "control_b" parameter:
- GPTA_NEG_CTRL_B_DISABLE:
  - Disables negation control
- GPTA_NEG_CTRL_B_ENABLE:
  - Enables negation control

The following values can be set in the "output_a" parameter:
- GPTA_NEG_OUTPUT_A_LOW:
  - Outputs "0" to the GTIOCnA pin during negation control (n = 0-3)
- GPTA_NEG_OUTPUT_A_HIGH:
  - Outputs "1" to the GTIOCnA pin during negation control (n = 0-3)

The following values can be set in the "output_b" parameter:
- GPTA_NEG_OUTPUT_B_LOW:
  - Outputs "0" to the GTIOCnB pin during negation control (n = 0-3)
- GPTA_NEG_OUTPUT_B_HIGH:
  - Outputs "1" to the GTIOCnB pin during negation control (n = 0-3)

The following values can be set in the "factor" parameter:
- GPTA_NEG_GETTRG:
  - Sets input to the GETTRG pin as the negation source.
- GPTA_NEG_SWN:
  - Sets soft_control as the negation source.

The following values can be set in the "polarity_select" parameter:
GPTA_NEG_POLARITY_LOW:
Performs negation control when the negation source is set to "0"
GPTA_NEG_POLARITY_HIGH:
Performs negation control when the negation source is set to "1"

The following values can be set in the "soft_control" parameter:
This parameter is valid only when factor is set to GPTA_NEG_SWN.
This parameter is ignored when factor is set to a value other than GPTA_NEG_SWN.
The behavior of soft_control varies according to the value of polarity_select.
polarity_select: GPTA_NEG_POLARITY_LOW
GPTA_NEG_SWN_LOW:
Performs negation control
GPTA_NEG_SWN_HIGH:
Does not perform negation control

polarity_select: GPTA_NEG_POLARITY_HIGH
GPTA_NEG_SWN_LOW:
Does not perform negation control
GPTA_NEG_SWN_HIGH:
Performs negation control

The following values can be set in the "flag_a" parameter:
GPTA_NEG_FLAG_A_DISABLE:
Disables output from the GTIOCnA pin (n = 0-3)
GPTA_NEG_FLAG_A_ENABLE:
Enables output from the GTIOCnA pin (n = 0-3)

The following values can be set in the "flag_b" parameter:
GPTA_NEG_FLAG_B_DISABLE:
Disables output from the GTIOCnB pin (n = 0-3)
GPTA_NEG_FLAG_B_ENABLE:
Enables output from the GTIOCnB pin (n = 0-3)

If this command is not called, the following default values will be used:
control_a: GPTA_NEG_CTRL_A_DISABLE
control_b: GPTA_NEG_CTRL_B_DISABLE
output_a: GPTA_NEG_OUTPUT_A_LOW
output_b: GPTA_NEG_OUTPUT_B_LOW
factor: GPTA_NEG_GETTRG
polarity_select: GPTA_NEG_POLARITY_HIGH
soft_control: GPTA_NEG_SWN_LOW(no negation control)
flag_a: GPTA_NEG_FLAG_A_DISABLE
flag_b: GPTA_NEG_FLAG_B_DISABLE

6.12.6 GPTA_READ_REG

GPTA_READ_REG

Synopsis Reading from the GPTa registers

Header r_gpta_if.h

Description This command reads from the GPTa timer counter, compare-capture register, period setting register, and A/D conversion start request timing register.
Parameters are transferred in the form of gpta_read_reg_t * type variables.

Parameters	gpta_reg_t register	Sets the type of the register to read from
	uint16_t *pdata	Specifies the area for storing the register value that is read
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument

Remarks The following values can be set in the "reg_id" parameter:

GPTA_REG_GTCNT:
Specifies a general PWM timer counter register
GPTA_REG_GTCCRA:
Specifies general PWM timer compare-capture register A
GPTA_REG_GTCCRB:
Specifies general PWM timer compare-capture register B
GPTA_REG_GTCCRC:
Specifies general PWM timer compare-capture register C
GPTA_REG_GTCCRD:
Specifies general PWM timer compare-capture register D
GPTA_REG_GTCCRE:
Specifies general PWM timer compare-capture register E
GPTA_REG_GTCCRF:
Specifies general PWM timer compare-capture register F
GPTA_REG_GTPR:
Specifies a general PWM timer period setting register
GPTA_REG_GTPBR:
Specifies a general PWM timer period setting buffer register
GPTA_REG_GTPDBR:
Specifies a general PWM timer period setting double-buffer register
GPTA_REG_GTADTRA:
Specifies A/D conversion start request timing register A
GPTA_REG_GTADTRB:
Specifies A/D conversion start request timing register B
GPTA_REG_GTADTBRA:
Specifies A/D conversion start request timing buffer register A
GPTA_REG_GTADTBRB:
Specifies A/D conversion start request timing buffer register B
GPTA_REG_GTADTDBRA:
Specifies A/D conversion start request timing double-buffer register A
GPTA_REG_GTADTDBRB:
Specifies A/D conversion start request timing double-buffer register B
GPTA_REG_GTDVU:
Specifies general PWM timer dead time value register U
GPTA_REG_GTDVD:
Specifies general PWM timer dead time value register D
GPTA_REG_GTDBU:
Specifies general PWM timer dead time buffer register U
GPTA_REG_GTDBD:
Specifies general PWM timer dead time buffer register D

The value of the specified register is written to the area specified in the "pdata" parameter.
The area must be allocated by the caller of this command.

6.12.7 GPTA_WRITE_REG

GPTA_WRITE_REG

Synopsis Writing to the GPTa register

Header r_gpta_if.h

Description This command writes to the GPTa timer counter, compare-capture register, period setting register, and A/D conversion start request timing register.
Parameters are transferred in the form of gpta_read_reg_t * type variables.

Parameters	gpta_reg_t reg_id	Sets the type of the register to write to
	uint16_t data	Sets the value to be written to the register
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument
	GPTA_ERR_TIMER_STARTED	: Timer operating

Remarks The following values can be set in the "reg_id" parameter:

GPTA_REG_GTCNT:
Specifies a general PWM timer counter register
This register cannot be written to during counting operations.

GPTA_REG_GTCCRA:
Specifies general PWM timer compare-capture register A

GPTA_REG_GTCCRB:
Specifies general PWM timer compare-capture register B

GPTA_REG_GTCCRC:
Specifies general PWM timer compare-capture register C

GPTA_REG_GTCCRD:
Specifies general PWM timer compare-capture register D

GPTA_REG_GTCCRE:
Specifies general PWM timer compare-capture register E

GPTA_REG_GTCCRF:
Specifies general PWM timer compare-capture register F

GPTA_REG_GTPR:
Specifies a general PWM timer period setting register

GPTA_REG_GTPBR:
Specifies a general PWM timer period setting buffer register

GPTA_REG_GTPDBR:
Specifies a general PWM timer period setting double-buffer register

GPTA_REG_GTADTRA:
Specifies A/D conversion start request timing register A

GPTA_REG_GTADTRB:
Specifies A/D conversion start request timing register B

GPTA_REG_GTADTBRA:
Specifies A/D conversion start request timing buffer register A

GPTA_REG_GTADTBRB:
Specifies A/D conversion start request timing buffer register B

GPTA_REG_GTADTDBRA:
Specifies A/D conversion start request timing double-buffer register A

GPTA_REG_GTADTDBRB:
Specifies A/D conversion start request timing double-buffer register B

GPTA_REG_GTDVU:
Specifies general PWM timer dead time value register U

GPTA_REG_GTDVD:
Specifies general PWM timer dead time value register D

GPTA_REG_GTDBU:
Specifies general PWM timer dead time buffer register U

GPTA_REG_GTDBD:
Specifies general PWM timer dead time buffer register D

The value to be written to the register is specified in the "pdata" parameter.

The value to be written to the register by this command must be changed according to the clock source.

6.12.8 GPTA_SET_BUF_MODE

GPTA_SET_BUF_MODE

Synopsis	Setting buffer operations	
Header	r_gpta_if.h	
Description	This command sets buffer operations. This command can be used only when the timer is stopped. No buffer, single buffer, or double buffer operations can be specified. Parameters are transferred in the form of gpta_set_buf_mode_t * type variables.	
Parameters	gpta_buf_t buf_id	Sets a target buffer
	uint16_t mode	Sets an operating mode
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument
	GPTA_ERR_TIMER_STARTED	: Timer operating

Remarks

The following values can be set in the "buf_id" parameter:

- GPTA_BUF_GTCCRA:
  - Specifies GTCCRA buffer operations
- GPTA_BUF_GTCCRB:
  - Specifies GTCCRB buffer operations
- GPTA_BUF_GTPR
  - Specifies GTPR buffer operations

The following values can be set in the "mode" parameter:

- GPTA_BUF_NONE:
  - Performs no buffer operation
- GPTA_BUF_SINGLE:
  - Performs single-buffer operations
- GPTA_BUF_DOUBLE:
  - Performs double-buffer operations

If this command is not called, the following default value will be used:
mode: GPTA_BUF_NONE

6.12.9 GPTA_SET_AD_BUF_MODE

GPTA_SET_AD_BUF_MODE

Synopsis	Setting buffer operations for the A/D conversion start request timing register	
Header	r_gpta_if.h	
Description	This command sets a buffer operation mode for the A/D conversion start request timing register. This command can be used only when the timer is stopped. Parameters are transferred in the form of gpta_set_ad_buf_mode_t * type variables.	
Parameters	gpta_buf_t buf_id	Sets a target buffer
	uint16_t timing	Sets the buffer transfer timing
	uint16_t mode	Sets an operating mode
Return values	GPTA_ERR_INVALID_ARG GPTA_ERR_TIMER_STARTED	: Invalid argument : Timer operating

Remarks

The following values can be set in the "buf_id" parameter:
- GPTA_AD_BUF_GTADTRA:
  - Specifies GTADTRA buffer operations
- GPTA_AD_BUF_GTADTRB:
  - Specifies GTADTRB buffer operations

The following values can be set in the "timing" parameter:
- For triangle waves:
  - GPTA_AD_BUF_TRANS_NONE:
    - Executes no buffer operation
  - GPTA_AD_BUF_TRANS_TOP:
    - Executes buffer transfer at the peak of a clock pulse
  - GPTA_AD_BUF_TRANS_BOTTOM:
    - Executes buffer transfer at the valley of a clock pulse
  - GPTA_AD_BUF_TRANS_BOTH:
    - Executes buffer transfer at both the peak and valley of a clock pulse
- For sawtooth waves:
  - GPTA_AD_BUF_TRANS_NONE:
    - Executes no buffer operation
  - GPTA_AD_BUF_TRANS_TOP:
  - GPTA_AD_BUF_TRANS_BOTTOM:
  - GPTA_AD_BUF_TRANS_BOTH:
    - Executes buffer transfer when an underflow (for count-down) or overflow (for count-up) occurs.

The following values can be set in the "mode" parameter:
- GPTA_AD_BUF_SINGLE:
  - Performs single-buffer operations
- GPTA_AD_BUF_DOUBLE:
  - Performs double-buffer operations

If this command is not called, the following default values will be used:
timing: GPTA_AD_BUF_TRANS_NONE
mode: GPTA_AD_BUF_SINGLE (Because no transfer is selected as above, no buffer operation is performed.)

6.12.10 GPTA_SET_CB

GPTA_SET_CB

Synopsis	Registering the callback function to be called when an interrupt occurs	
Header	r_gpta_if.h	
Description	This command registers the callback function to be called when the interrupt specified in the parameter occurs. GPTA_INTR_ETGIN and GPTA_INTR_ETGIP are common to all channels. The channel specified in the ch argument of the R_GPTA_Control function is ignored. Parameters are transferred in the form of gpta_set_cb_t type variables.	
Parameters	gpta_intr_t intr_type	Specifies an interrupt type
	gpta_cb callback	Specifies a callback function
	uint8_t intr_pri	Specifies the interrupt priority level
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument

Remarks

The following values can be set in the "intr_type" parameter:

GPTA_INTR_GTCIA:
Input capture/compare match A interrupt (GTCIAn)
GPTA_INTR_GTCIB:
Input capture/compare match B interrupt (GTCIBn)
GPTA_INTR_GTCIC:
Compare match C interrupt (GTCICn)
GPTA_INTR_GTCID:
Compare match D interrupt (GTCIDn)
GPTA_INTR_GTCIE:
Compare match E interrupt (GTCIEn)
GPTA_INTR_GTCIF:
Compare match F interrupt (GTCIFn)
GPTA_INTR_GDTE:
Dead time error interrupt (GTDEn)
GPTA_INTR_GTCIV:
Overflow interrupt (GTCIVn)
Sawtooth wave: Overflow interrupt
Triangle wave: Interrupt at a peak of a pulse
GPTA_INTR_GTCIU:
Underflow interrupt (GTCIUn)
Sawtooth wave: Underflow interrupt
Triangle wave: Interrupt at a valley of a pulse
GPTA_INTR_ETGIN:
Input interrupt at a falling edge of an external trigger signal (ETGIN)
GPTA_INTR_ETGIP:
Input interrupt at a rising edge of an external trigger signal (ETGIP)

The following values can be set in the "callback" parameter:
For other than NULL: Sets the entry address for the callback function
NULL: Cancels the registration of the callback function

The following values can be set in the "intr_pri" argument:
0~15: Interrupt priority level (0 is the highest priority)

If this command is not called, the following default values will be used for all "intr_type" settings:
callback: NULL(interrupt disabled)
intr_pri: 0

6.12.11 GPTA_TIMER_START_SW

GPTA_TIMER_START_SW

Synopsis Starting the counter

Header r_gpta_if.h

Description This command starts or stops the counters for the GPTn.GTCNT (n: 0-3) registers.
This command is common to all channels.
This command ignores the channels specified in the ch argument of the R_GPTA_Control function, and starts and stops the counters for the channels specified in the start_ch and stop_ch parameters, respectively. (Counters for multiple channels can be started or stopped at the same time.)
Parameters are transferred in the form of gpta_timer_start_t type variables.

Parameters	uint8_t start_ch	Sets the channel whose counter is to be started.
	uint8_t stop_ch	Sets the channel whose counter is to be stopped.
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument
	GPTA_ERR_STATUS	: The channel specified in the parameter is not open

Remarks The following values can be set in the "start_ch" parameter:

GPTA_TIMER_START_0:
Starts the counter for channel 0
GPTA_TIMER_START_1:
Starts the counter for channel 1
GPTA_TIMER_START_2:
Starts the counter for channel 2
GPTA_TIMER_START_3:
Starts the counter for channel 3

The following values can be set in the "stop_ch" parameter:

GPTA_TIMER_STOP_0:
Stops the counter for channel 0
GPTA_TIMER_STOP_1:
Stops the counter for channel 1
GPTA_TIMER_STOP_2:
Stops the counter for channel 2
GPTA_TIMER_STOP_3:
Stops the counter for channel 3

To start counters for multiple channels at the same time, specify as below.
Example: To start the counters for channels 0 and 2 at the same time

```
start_ch = GPTA_START_0 | GPTA_START_2;
```

To stop counters for multiple channels at the same time, specify as below.
Example: To stop the counters for channels 1 and 3 at the same time

```
stop_ch = GPTA_STOP_1 | GPTA_STOP_3
```

Before starting counters for multiple channels at the same time, use the command to set the initial value of counting in GPTA_REG_GTCNT so that counting operations can be performed for each channel with phases.

This command does not influence the counters for the channels that are not specified in the parameters of the command.
If the same channel is specified for both the "start_ch" and "stop_ch" parameters, an error will return. (Invalid arguments)

The counters for the channels that are not specified in "start_ch" remain stopped.

6.12.12 GPTA_CTRL_DTIME

GPTA_CTRL_DTIME

Synopsis	Controlling the dead time	
Header	r_gpta_if.h	
Description	This command automatically sets compare match values for reversed-phase waveforms with dead time. Parameters are transferred in the form of gpta_ctrl_dtime_t type variables.	
Parameters	uint16_t set_mode	Specifies the method for setting dead time
	uint16_t gtdvu_buf_mode	Specifies a buffer operation mode for the GTDVU register
	uint16_t gtdvd_buf_mode	Specifies a buffer operation mode for the GTDVD register
	uint16_t gtdvd_set_mode	Specifies the method for setting the GTDVD register
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument

Remarks

The following values can be set in the "set_mode" parameter:
  GPTA_DTIME_MANUAL_SET:
    Sets the GTCCRB register manually without using the GTDVU and GTDVD registers
  GPTA_DTIME_AUTO_SET:
    Uses the GTDVU and GTDVD registers to automatically set compare match values for reversed-phase waveforms with dead time into the GTCCRB register

The following values can be set in the "gtdvu_buf_mode" parameter:
  GPTA_GTDVU_BUF_DISABLE:
    Disables buffer operations for the GTDVU register
  GPTA_GTDVU_BUF_ENABLE:
    Enables buffer operations for the GTDVU register

The following values can be set in the "gtdvd_buf_mode" parameter:
  GPTA_GTDVD_BUF_DISABLE:
    Disables buffer operations for the GTDVD register
  GPTA_GTDVD_BUF_ENABLE:
    Enables buffer operations for the GTDVD register

The following values can be set in the "dtdvd_set_mode" parameter:
  GPTA_GTDVD_MANUAL_SET:
    Sets the GTDVU register and the GTDVD register separately
  GPTA_GTDVD_AUTO_SET:
    Automatically sets the value written to the GTDVU register into the GTDVD register

The command must be used to write to the GTDVU, GTDVD, GTDBU, or GTDBD register.

In sawtooth wave PWM mode, this command is ignored and does not automatically set compare match values.

If this command is not called, the following default values will be used:
  set_mode: GPTA_DTIME_MANUAL_SET
  gtdvu_buf_mode: GPTA_GTDVU_BUF_DISABLE
  gtdvd_buf_mode: GPTA_GTDVD_BUF_DISABLE
  dtdvd_set_mode: GPTA_GTDVD_MANUAL_SET

6.12.13 GPTA_SET_HW_FACTOR

GPTA_SET_HW_FACTOR

Synopsis	Setting a hardware source that starts, stops, or clears the counter	
Header	r_gpta_if.h	
Description	This command sets a hardware source that starts, stops, or clears the counters for the GPTn.GTCNT (n: 0-3) registers. Parameters are transferred in the form of gpta_set_fw_factor_t type variables.	
Parameters	gpta_count_type_t count_type	Sets a counter operation type (start, stop, or clear)
	uint16_t trigger_type	Sets an effective edge for a hardware source
	uint16_t factor	Sets a hardware source
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument

Remarks

The following values can be set in the "count_type" parameter:

- GPTA_COUNT_START:
  - Sets a hardware source that starts the counter
- GPTA_COUNT_STOP:
  - Sets a hardware source that stops the counter
- GPTA_COUNT_CLEAR:
  - Sets a hardware source that clears the counter
- GPTA_COUNT_CLEAR_SW:
  - Clears the counter. If this value is specified, the other parameters (trigger_type, factor) are ignored.

The following values can be set in the "trigger_type" parameter:
This parameter specifies an effective edge for the hardware source that triggers the counter operation specified in "count_type".

- GPTA_TRG_TYPE_NONE:
  - Sets no effective edge for the hardware source
- GPTA_TRG_TYPE_RISING:
  - Sets a rising edge as the effective edge for the hardware source
- GPTA_TRG_TYPE_FALLING:
  - Sets a falling edge as the effective edge for the hardware source
- GPTA_TRIG_TYPE_BOTH:
  - Sets both edges as the effective edge for the hardware source

The following values can be set in the "factor" parameter.
This parameter specifies the hardware source that triggers the counter operation specified in "count_type".

- GPTA_FACTOR_NONE:
  - Sets no hardware source
- GPTA_FACTOR_GTIOC3A:
  - Sets input to the GTIOC3A pin as the hardware source
- GPTA_FACTOR_GTIOC3B:
  - Sets input to the GTIOC3B pin as the hardware source
- GPTA_FACTOR_CMP_A:
  - Sets internal output from the GTIOC3A pin (output compare) as the hardware source. This value cannot be specified for channel 3.
- GPTA_FACTOR_CMP_B:
  - Sets internal output from the GTIOC3B pin (output compare) as the hardware source. This value cannot be specified for channel 3.
- GPTA_FACTOR_GETTRG:
  - Sets input to the GTETRGB pin as the hardware source

If this command is not called, the following default values will be used:
trigger_type: GPTA_TRG_TYPE_NONE
count_type: GPTA_FACTOR_NONE

6.12.14 GPTA_SET_CLEAR_FACTOR

GPTA_SET_CLEAR_FACTOR

Synopsis	Setting a source that clears the counter	
Header	r_gpta_if.h	
Description	This command sets a source that clears the counter for each channel of the GPTn.GTCNT (n: 0-3) register. Parameters are transferred in the form of gpta_set_clear_factor_t type variables.	
Parameters	uint16_t clear_factor	Sets a source that clears the counter
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument

Remarks
The following values can be set in the "clear_factor" parameter:
GPTA_CLEAR_NONE:
Does not set a source that clears the counter
GPTA_CLEAR_GTCCRA:
Clears the counter by input capture for the GTCCRA register
GPTA_CLEAR_GTCCRB:
Clears the counter by input capture for the GTCCRB register
GPTA_CLEAR_SYNC:
Clears the counter by synchronous clearing or by another counter clearing source that operates synchronously

If this command is not called, the following default value will be used:
clear_factor: GPTA_CLEAR_NONE

6.12.15 GPTA_SET_SYNC

GPTA_SET_SYNC		
Synopsis	Setting a source that clears the counter synchronously	
Header	r_gpta_if.h	
Description	This command sets a source that synchronously clears the counter for each channel of the GPTn.GTCNT (n: 0-3) register. Parameters are transferred in the form of gpta_set_sync_t type variables.	
Parameters	uint16_t clear_factor	Sets a source that clears the counter
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument
Remarks	The following values can be set in the "clear_factor" parameter: GPTA_CLEAR_GPT0: Clears the counter in sync with the clearing source for GPT0 (channel 0) GPTA_CLEAR_GPT1: Clears the counter in sync with the clearing source for GPT1 (channel 1) GPTA_CLEAR_GPT2: Clears the counter in sync with the clearing source for GPT2 (channel 2) GPTA_CLEAR_GPT3: Clears the counter in sync with the clearing source for GPT3 (channel 3) Synchronous clearing does not occur if the channel specified in the ch argument of the R_GPTA_Control function is the same as the channel specified in clear_factor. If this command is not called, the following default value will be used: clear_factor: GPTA_CLEAR_GPT0	

6.12.16 GPTA_SET_NF

GPTA_SET_NF

Synopsis	Setting the noise filter	
Header	r_gpta_if.h	
Description	This command enables or disables the noise filter, and sets a sampling clock for the noise filter for each channel. Parameters are transferred in the form of gpta_set_nf_t type variables.	
Parameters	uint16_t filter_a_flag	Specifies whether to enable or disable the noise filter for the GTIOCnA pin. (n: Channel number)
	uint16_t filter_b_flag	Specifies whether to enable or disable the noise filter for the GTIOCnB pin. (n: Channel number)
	uint16_t sampling_clock	Specifies a sampling clock for the noise filter
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument

Remarks

The following values can be set in the "filter_a_flag" parameter:
  GPTA_NF_DISABLE:
    Disables the noise filter for the GTIOCnA pin
  GPTA_NF_ENABLE:
    Enables the noise filter for the GTIOCnA pin

The following values can be set in the "filter_b_flag" parameter:
  GPTA_NF_DISABLE:
    Disables the noise filter for the GTIOCnB pin
  GPTA_NF_ENABLE:
    Enables the noise filter for the GTIOCnB pin

The following values can be set in the "sampling_clock" parameter:
  GPTA_SAMP_PCLKC_DIV_1:
    Specifies PCLKC / 1 for the sampling clock
  GPTA_SAMP_PCLKC_DIV_4:
    Specifies PCLKC / 4 for the sampling clock
  GPTA_SAMP_PCLKC_DIV_32:
    Specifies PCLKC / 32 for the sampling clock
  GPTA_SAMP_COUNT_SRC:
    Specifies the count source for the sampling clock

If this command is not called, the following default values will be used:
  filter_a_flag: GPTA_NF_DISABLE
  filter_b_flag: GPTA_NF_DISABLE
  sampling_clock: GPTA_SAMP_PCLKC_DIV_1

6.12.17 GPTA_SET_NF_EX

GPTA_SET_NF_EX

Synopsis Setting the noise filter for the external trigger input pin

Header r_gpta_if.h

Description This command sets the noise filter for the external trigger input pin (GTETRG).
This command is common to all channels.
Parameters are transferred in the form of gpta_set_nf_ex_t type variables.

Parameters	uint16_t filter_flag	Specifies whether to enable or disable the noise filter for the GTETRG pin.
	uint16_t sampling_clock	Specifies a sampling clock for the noise filter
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument

Remarks The following values can be set in the "filter_flag" parameter:
GPTA_NF_EX_DISABLE:
Disables the noise filter for the GTETRG pin
GPTA_NF_EX_DISABLE:
Enables the noise filter for the GTETRG pin

The following values can be set in the "sampling_clock" parameter:
GPTA_SAMP_EX_PCLKC_DIV_1:
Specifies PCLKC / 1 for the sampling clock
GPTA_SAMP_EX_PCLKC_DIV_2:
Specifies PCLKC / 2 for the sampling clock
GPTA_SAMP_EX_PCLKC_DIV_4:
Specifies PCLKC / 4 for the sampling clock
GPTA_SAMP_EX_PCLKC_DIV_32:
Specifies PCLKC / 32 for the sampling clock

If this command is not called, the following default values will be used:
filter_flag: GPTA_NF_EX_DISABLE
sampling_clock: GPTA_SAMP_EX_PCLKC_DIV_1

6.12.18 GPTA_SET_INTR_SKIP

GPTA_SET_INTR_SKIP

Synopsis	Setting the interrupt skip function	
Header	r_gpta_if.h	
Description	This command specifies whether to link the settings for the overflow interrupt (GTCIV)/underflow interrupt (GTCIU) skip function of the GTCNT counter to other interrupts, and whether to link A/D conversion start requests with the GTCIV/GTCIU interrupt skip function. These specifications are set for each channel. Parameters are transferred in the form of gpta_set_intr_skip_t variables.	
Parameters	uint16_t gtccra_mode	Sets whether to link with the interrupt skip function
	uint16_t gtccrb_mode	Sets whether to link with the interrupt skip function
	uint16_t gtccrc_mode	Sets whether to link with the interrupt skip function
	uint16_t gtccrd_mode	Sets whether to link with the interrupt skip function
	uint16_t gtccre_mode	Sets whether to link with the interrupt skip function
	uint16_t gtccrf_mode	Sets whether to link with the interrupt skip function
	uint16_t gtadtra_mode	Sets whether to link with the interrupt skip function
	uint16_t gtadtrb_mode	Sets whether to link with the interrupt skip function
	uint16_t func_sel	Selects the interrupt skip function
	uint16_t skip_num	Skips the number of interrupt skips
Return values	GPTA_ERR_INVALID_ARG	: Invalid argument

Remarks

The following values can be set in the "gtccra_mode" parameter:

GTCCRA_INTR_SKIP_DISABLE:

Does not link GTCCRA compare match/input capture interrupts with the GTCIV/GTCIU interrupt skip function

GTCCRA_INTR_SKIP_ENABLE:

Links GTCCRA compare match/input capture interrupts with the GTCIV/GTCIU interrupt skip function

The following values can be set in the "gtccrb_mode" parameter:

GTCCRB_INTR_SKIP_DISABLE:

Does not link GTCCRB compare match/input capture interrupts with the GTCIV/GTCIU interrupt skip function

GTCCRB_INTR_SKIP_ENABLE:

Links GTCCRB compare match/input capture interrupts with the GTCIV/GTCIU interrupt skip function

The following values can be set in the "gtccrc_mode" parameter:

GTCCRC_INTR_SKIP_DISABLE:

Does not link GTCCRC compare match interrupts with the GTCIV/GTCIU interrupt skip function

GTCCRC_INTR_SKIP_ENABLE:

Links GTCCRC compare match interrupts with the GTCIV/GTCIU interrupt skip function

The following values can be set in the "gtccrd_mode" parameter:

GTCCRD_INTR_SKIP_DISABLE:

Does not link GTCCRD compare match interrupts with the GTCIV/GTCIU interrupt skip function

GTCCRD_INTR_SKIP_ENABLE:

Links GTCCRD compare match interrupts with the GTCIV/GTCIU interrupt skip function

The following values can be set in the "gtccre_mode" parameter:

GTCCRE_INTR_SKIP_DISABLE:

Does not link GTCCRE compare match interrupts with the GTCIV/GTCIU interrupt skip function

GTCCRE_INTR_SKIP_ENABLE:

Links GTCCRE compare match interrupts with the GTCIV/GTCIU interrupt skip function

The following values can be set in the "gtccrf_mode" parameter:

GTCCRF_INTR_SKIP_DISABLE:

Does not link GTCCRF compare match interrupts with the GTCIV/GTCIU interrupt skip function

GTCCRF_INTR_SKIP_ENABLE:

Links GTCCRF compare match interrupts with the GTCIV/GTCIU interrupt skip function

The following values can be set in the "gtadtra_mode" parameter:

GTADTRA_INTR_SKIP_DISABLE:

Does not link GTADTRA A/D conversion start requests with the GTCIV/GTCIU interrupt skip function

GTADTRA_INTR_SKIP_ENABLE:

Links GTADTRA A/D conversion start requests with the GTCIV/GTCIU interrupt skip function

The following values can be set in the "gtadtrb_mode" parameter:

GTADTRB_INTR_SKIP_DISABLE:

Does not link GTADTRB A/D conversion start requests with the GTCIV/GTCIU interrupt skip function

GTADTRB_INTR_SKIP_ENABLE:

Links GTADTRB A/D conversion start requests with the GTCIV/GTCIU interrupt skip function

The following values can be set in the "func_sel" parameter:

GPTA_INTR_SKIP_NONE:

Does not skip interrupts

GPTA_INTR_SKIP_TOP:

Skips an interrupt by counting both overflows and underflows for sawtooth waves or peaks for triangle waves

GPTA_INTR_SKIP_BOTTOM:

Skips an interrupt by counting both overflows and underflows for sawtooth waves or valleys for triangle waves

GPTA_INTR_SKIP_BOTH:

Skips an interrupt by counting both overflows and underflows for sawtooth waves or both peaks and valleys for triangle waves

The following values can be set in the "skip_num" parameter:

GPTA_INTR_SKIP_0:

Skips the number of skips to 0 (No interrupt will be skipped)

GPTA_INTR_SKIP_1:

Skips the number of skips to 1

GPTA_INTR_SKIP_2:

Skips the number of skips to 2

GPTA_INTR_SKIP_3:

Skips the number of skips to 3

GPTA_INTR_SKIP_4:

Skips the number of skips to 4

GPTA_INTR_SKIP_5:

Skips the number of skips to 5

GPTA_INTR_SKIP_6:

Skips the number of skips to 6

GPTA_INTR_SKIP_7:

Skips the number of skips to 7

If this command is not called, the following default values will be used:
gtccra_mode: GTCCRA_INTR_SKIP_DISABLE
gtccrb_mode: GTCCRB_INTR_SKIP_DISABLE
gtccrc_mode: GTCCRC_INTR_SKIP_DISABLE
gtccrd_mode: GTCCRD_INTR_SKIP_DISABLE
gtccre_mode: GTCCRE_INTR_SKIP_DISABLE
gtccrf_mode: GTCCRF_INTR_SKIP_DISABLE
gtadtra_mode: GTADTRA_INTR_SKIP_DISABLE
gtadtrb_mode: GTADTRB_INTR_SKIP_DISABLE
func_sel: GPTA_INTR_SKIP_NONE

7. Sample Program

Download the sample code from the Renesas Electronics website.

8. Related Documents

- User's Manuals: Hardware
  RZ/T1 Group User's Manual: Hardware
  (Download the latest edition from the Renesas Electronics website.)

  RZ/T1 Evaluation Board RTK7910022C00000BR User's Manual
  (Download the latest edition from the Renesas Electronics website.)

- Technical Update and Technical News
  (Download the latest information from the Renesas Electronics website.)

- User's Manuals: Development Environment
  For the IAR Embedded Workbench® for Arm, download the user's manual from the IAR website.
  (Download the latest edition from the IAR website.)

Website and Support

Renesas Electronics website

http://www.renesas.com/

Inquiries

http://www.renesas.com/inquiry

Revision History	Application Note: POE3 & GPTa Sample Driver

Rev.	Date	Description	
		Page	Summary
0.10	Apr. 02, 2015	—	First Edition issued
1.00	Apr. 10, 2015	—	Only the revision number was changed to be posted on a website.
1.10	Jul. 16, 2015	2. Operating Environment	
		5	Table 2.1 Operating Environment: Description added to Integrated Development Environment
		6. Software	
		11	6.2.4 Required Memory Size: Description and reference added
		11	Table 6.2: Table title and size description were partially amended
		11	Table 6.2 Memory Requirements: Description on the Note, changed
		12	Table 6.3 added
		12	Table 6.4 added
1.20	Dec. 04, 2015	2. Operating Environment	
		5	Table 2.1 Operating Environment: Integrated Development Environment, information partially amended
1.30	Apr. 05, 2017	2. Operating Environment	
		5	Table 2.1 Operating Environment: Integrated Development Environment, modified
		6. Software	
		—	6.2.4 Required Memory Size, deleted
1.40	Jun. 07, 2018	2. Operating Environment	
		5	Table 2.1 Operating Environment: The description on the integrated development environment, modified
		8. Related Documents	
		72	The name of IAR Embedded Workbench, modified

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. Handling of Unused Pins

   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

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

2. Processing at Power-on

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

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

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

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

4. Clock Signals

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

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

5. Differences between Products

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

   — The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

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 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.
5. 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.
6. 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.
7. 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.
8. 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.
9. 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.
10. 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.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

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

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

Renesas Electronics America Inc.
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 P. R. China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, 200333 P. R. China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 7.0