

RZ/T1 グループ

CMSIS-RTOS RTX for Cortex-R4

MTU3a サンプルプログラム

(EWARM / ICCARM, e2studio/RenesasGCC, DS-5/ARMCC)

R01AN3540JJ0100

Rev.1.00

2016.12.27

要旨

本アプリケーションノートでは、RZ/T1 のマルチファンクションタイマパルスユニット (MTU3a) の MTU3、MTU4 の相補 PWM モードを使用して、正と負の 3 相 (6 本) のデッドタイム付き PWM 波形を出力するサンプルプログラムについて説明します。

MTU3a サンプルプログラムの特長を以下に示します。

- ・ MTU3、MTU4 を使ったキャリア周期 ($=100\mu\text{s}$)、デッドタイム ($=2\mu\text{s}$) 付きの相補 PWM 波形を出力します。

動作確認デバイス

RZ/T1

本アプリケーションノートを他のマイコンへ適用する場合、そのマイコンの仕様にあわせて変更し、十分評価してください。

目次

1. 仕様.....	5
2. 動作環境.....	6
3. ドキュメント.....	7
3.1 関連文書一覧.....	7
4. ハードウェア説明.....	8
4.1 ハードウェア構成例.....	8
4.2 使用端子一覧.....	9
5. ソフトウェア説明.....	10
5.1 動作概要.....	10
5.2 プロジェクト設定	11
5.3 メモリマップ	11
5.3.1 サンプルプログラムのセクション配置.....	11
5.3.2 MPU の設定	11
5.3.3 例外処理ベクタテーブル	11
5.3.4 必要メモリサイズ	11
5.4 フォルダ構成.....	11
5.5 使用割り込み一覧	12
5.6 固定幅整数一覧.....	12
5.7 定数一覧.....	12
5.7.1 サンプルプログラムで使用する定数	12
5.7.2 サンプルプログラムで使用するエラーコード定数.....	12
5.8 構造体/共用体/列挙体一覧.....	13
5.9 関数一覧.....	26
5.10 サンプルプログラム関数仕様	27
5.10.1 main	27
5.10.2 SER_Init.....	27
5.10.3 mtu3_setup	27
5.11 MTU ドライバ関数仕様.....	28
5.11.1 R_MTU_Init.....	28
5.11.2 R_MTU_Uninit.....	28
5.11.3 R_MTU_PWM_Compliment_Open.....	28
5.11.4 R_MTU_PWM_Compliment_Close	28
5.11.5 R_MTU_PWM_Compliment_Control	29
5.11.6 R_MTU_Timer_Open.....	29
5.11.7 R_MTU_Capture_Open	30
5.11.8 R_MTU_PWM_Open	30
5.11.9 R_MTU_Close.....	30
5.11.10 R_MTU_Control	31
5.11.11 R_MTU_GetVersion.....	31

5.12	フローチャート	32
5.12.1	サンプルアプリケーションメイン処理	32
5.12.2	相補 PWM モード制御処理	33
5.12.3	MTU コンペア/マッチタイミング処理	34
5.12.4	MTU キャプチャ処理	35
5.12.5	PWM モード制御処理	36
5.13	R_MTU_PWM_Compliment_Open パラメーター一覧	37
5.13.1	clock_src.source	37
5.13.2	clock_src.clock_edge	37
5.13.3	clk_div.clock_div	38
5.13.4	clk_div.cycle_freq	38
5.13.5	dead_time	38
5.13.6	toggle	38
5.13.7	mode	39
5.13.8	p_n	39
5.13.9	p_n_bf	39
5.13.10	d_bf	39
5.13.11	protect	40
5.13.12	pwm_output_X.olsp (X= 1、2、3)	40
5.13.13	pwm_output_X.olsn (X= 1、2、3)	40
5.13.14	pwm_output_X.duty (X= 1、2、3)	40
5.13.15	pwm_output_X.output (X= 1、2、3)	41
5.14	R_MTU_PWM_Compliment_Control パラメーター一覧	42
5.14.1	cmd = MTU_CMD_START 時	42
5.14.2	cmd = MTU_CMD_STOP 時	42
5.14.3	cmd = MTU_CMD_GET_STATUS 時	42
5.15	R_MTU_Timer_Open パラメーター一覧	43
5.15.1	clock_src.source	43
5.15.2	clock_src.clock_edge	43
5.15.3	clear_src	44
5.15.4	timer_X.actions.freq (X = a, b, c, d)	44
5.15.5	timer_X.actions.do_action (X = a, b, c, d)	45
5.15.6	timer_X.actions.output (X = a, b, c, d)	45
5.16	R_MTU_Capture_Open パラメーター一覧	46
5.16.1	clock_src.source	46
5.16.2	clock_src.clock_edge	46
5.16.3	clock_div	47
5.16.4	clear_src	47
5.16.5	capture_X.actions (X = a, b, c, d)	48
5.16.6	capture_X.capture_edge (X = a, b, c, d)	48
5.16.7	capture_X.filter_enable (X = a, b, c, d)	49
5.17	R_MTU_PWM_Open パラメーター一覧	50
5.17.1	clock_src.source	50
5.17.2	clock_src.clock_edge	50
5.17.3	cycle_freq	51
5.17.4	clear_src	51
5.17.5	pwm_mode	51

5.17.6	pwm_X.duty (X = a, b, c, d).....	51
5.17.7	pwm_X.actions (X = a, b, c, d).....	52
5.17.8	pwm_X.outputs (X = a, b, c, d)	52
5.18	R_MTU_Control パラメータ一覧	53
5.18.1	cmd = MTU_CMD_START 時	53
5.18.2	cmd = MTU_CMD_STOP 時	53
5.18.3	cmd = MTU_CMD_SAFE_STOP 時	54
5.18.4	cmd = MTU_CMD_RESTART 時	54
5.18.5	cmd = MTU_CMD_SYNCHRONIZE 時	54
5.18.6	cmd = MTU_CMD_GET_STATUS 時 (timer mode 時)	54
5.18.7	cmd = MTU_CMD_GET_STATUS 時 (input capture mode 時)	54
5.18.8	cmd = MTU_CMD_SET_CAPT_EDGE 時	55
6.	サンプルプログラム	56

1. 仕様

以下に、使用する周辺機能と用途を示します。

表 1.1 使用する周辺機能と用途

周辺機能	用途
クロック発生回路 (CPG)	CPU クロックおよび低速オンチップオシレータで使用
割り込みコントローラ (ICUA)	MTU3a のコンペアマッチ割り込みで使用
マルチファンクションタイマパルスユニット (MTU3a)	MTU3、MTU4 の相補 PWM 出力で使用
エラーコントロールモジュール (ECM)	ERROROUT#端子の初期化

上記の各機能の内容は『RZ/T1 グループ・ユーザズマニュアル ハードウェア編』に記載しています。

2. 動作環境

本アプリケーションのサンプルプログラムは、下記の環境を想定しています。

表 2.1 動作環境

項目	内容
使用マイコン	RZ/T1 グループ
動作周波数	CPUCLK = 450MHz
動作電圧	3.3V
統合開発環境	IAR システムズ製 Embedded Workbench for ARM Version 7.80.2 (以下、EWARM と略す) RENESAS 製 e2studio 5.2.0.020 (以下、e2studio と略す) ARM 製 DS-5 Version: 5.25.0 (以下、DS-5 と略す)
動作モード	SPI ブートモード (シリアルフラッシュからのブート) SW4 : ON/ON/ON RAM ブートモード (JTAG-ICE を用いた内蔵 RAM への直接ダウンロードによるブート)、または、16 ビットバスブートモード (NOR フラッシュからのブート) SW4 : ON/OFF/ON
使用ボード	RZ/T1 CPU ボード (RTK7910018C00000BE)
使用デバイス (ボード上で使用する機能)	<ul style="list-style-type: none"> NOR フラッシュメモリ (CS0、CS1 空間に接続) メーカー名: Macronix International Co., Ltd. 型名: MX29GL512FLT2I-10Q SDRAM (CS2、CS3 空間に接続) メーカー名: Integrated Silicon Solution Inc. 型名: IS42S16320D-7TL シリアルフラッシュメモリ メーカー名: Macronix International Co., Ltd. 型名: MX25L51245GMI-10G
ICE	IAR システムズ製 I-jet JTAG エミュレータ SEGGER 社製 J-Link JTAG エミュレータ ARM 社製 KEIL ULINK2 エミュレータ

3. ドキュメント

3.1 関連文書一覧

本書に関連する文書を以下に示します。

- CMSIS-RTOS compliant Kernel Version 4.74

本システムで使用している RTOS の仕様書です。サンプルアプリケーションは、CMSIS-RTOS RTX の機能を使用します。また、各サンプルアプリケーションは CMSIS-RTOS RTX を初期化します。

- RZ/T1 グループ ユーザーズマニュアル ハードウェア編 (R01UH0483)

RZ/T1 のハードウェア仕様を記載しています。

(最新版をルネサス エレクトロニクスホームページから入手してください。)

- RZ/T1 グループ 初期設定 アプリケーションノート (R01AN2554)

RZ/T1 の初期設定について記載しています。本書に記載のないものは、この「RZ/T1 グループ 初期設定アプリケーションノート」で設定した値を使用します。

(最新版をルネサス エレクトロニクスホームページから入手してください。)

- RZ/T1 グループ CMSIS-RTOS RTX for Cortex-R4 RTX サンプルプログラム アプリケーションノート (R01AN3538JJ)

RZ/T1 上で動作する CMSIS-RTOS RTX サンプルアプリケーションについて記載しています。

- テクニカルアップデート／テクニカルニュース

(最新の情報をルネサス エレクトロニクスホームページから入手してください。)

- 開発環境に関わるユーザーズマニュアル

IAR 統合開発環境 (IAR Embedded Workbench for ARM) に関しては、最新版を IAR ホームページから入手してください。

DS-5 統合開発環境 (ARM Development Studio 5) に関しては、最新版を ARM ホームページから入手してください。

ルネサス エレクトロニクスソフトウェア開発ツール (e2studio 等) に関しては、ルネサスエレクトロニクスホームページから最新版を入手してください。

4. ハードウェア説明

4.1 ハードウェア構成例

以下に、RZ/T1 CPU ボードを用いた基本的な接続例を示します。

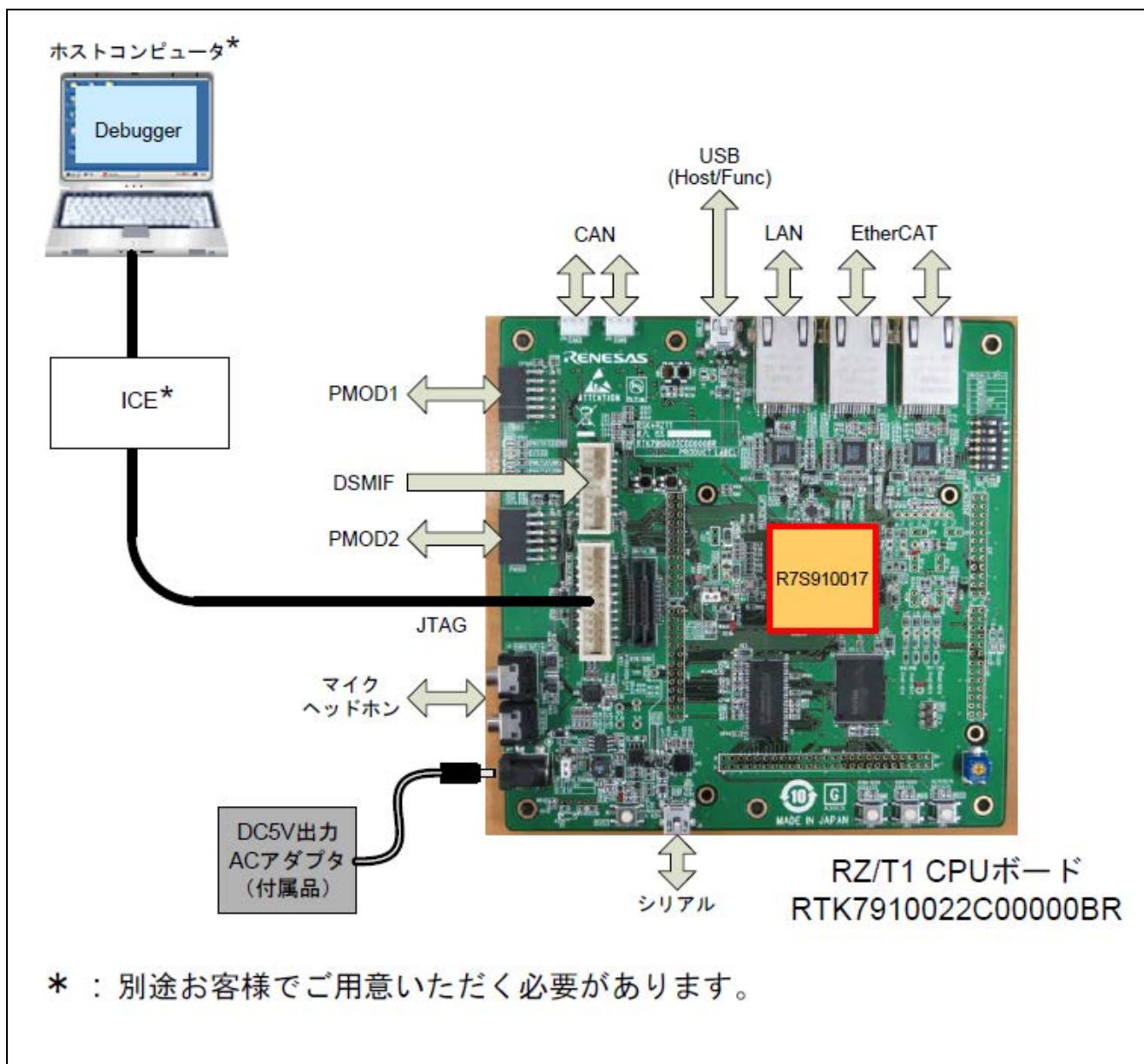


図 4.1 動作環境

以下に、本サンプルプログラムで使用されるハードウェア構成例を示します。

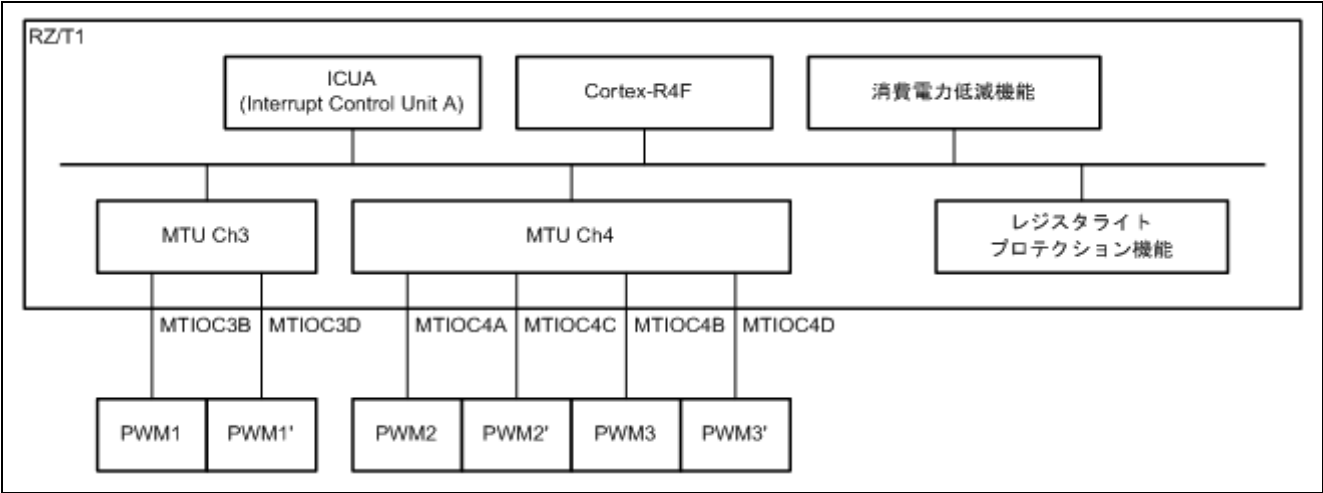


図 4.2 ハードウェア構成例

4.2 使用端子一覧

以下に、使用端子と機能を示します。

表 4.1 使用端子と機能

端子名	入出力	内容
MD0	入力	動作モードの選択 MD0="L"、MD1="L"、MD2="L" (SPI ブートモード) MD0="L"、MD1="H"、MD2="L" (RAM ブートモード、または、16 ビットバスブートモード)
MD1	入力	
MD2	入力	
MTIOC3B	出力	PWM 出力 1
MTIOC3D	出力	PWM 出力 1' (PWM 出力 1 の逆相波形出力)
MTIOC4A	出力	PWM 出力 2
MTIOC4C	出力	PWM 出力 2' (PWM 出力 2 の逆相波形出力)
MTIOC4B	出力	PWM 出力 3
MTIOC4D	出力	PWM 出力 3' (PWM 出力 3 の逆相波形出力)

5. ソフトウェア説明

5.1 動作概要

本サンプルプログラムでは、マルチファンクションタイマパルスユニット（MTU3a）の初期設定を行い、相補 PWM 出力を行います。

本サンプルプログラムの動作概要を以下に示します。また、図 5.1 に波形を示します。

表 5.1 動作概要

機能	概要
チャンネル	チャンネル 3（MTU3）、チャンネル 4（MTU4）
PWM 出力	正と負の 3 相（6 本）、キャリア周期 100us、デッドタイム 2us
動作モード	相補 PWM モード 1（山で転送）
クロック	PCLKC/4（=37.5MHz）、立ち上がりエッジ
デューティ比	25%

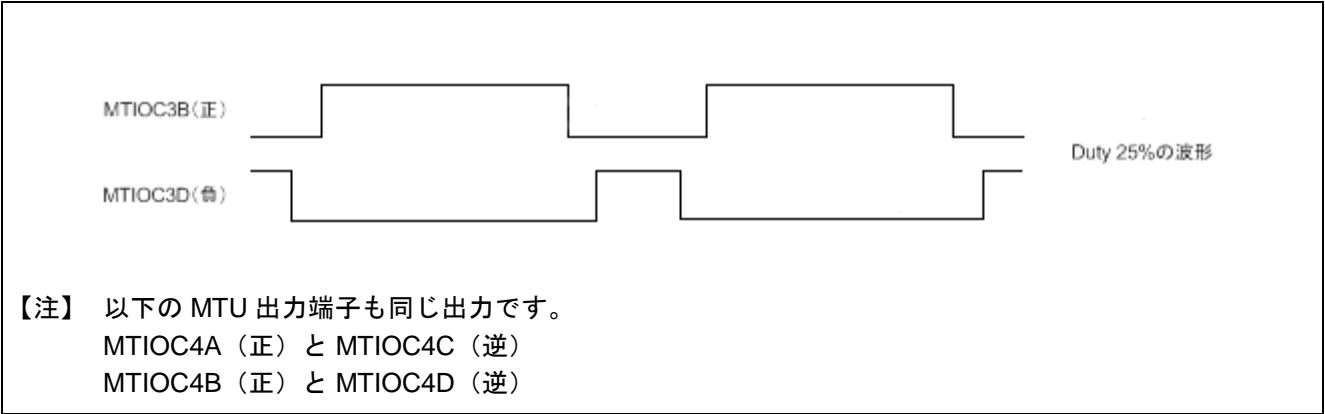


図 5.1 波形

5.2 プロジェクト設定

開発環境となるEWARM / e2studio / DS-5上で使用されるプロジェクト設定については、「RZ/T1グループ 初期設定 アプリケーションノート」に記載しています。

5.3 メモリマップ

RZ/T1 グループのアドレス空間と RZ/T1 CPU ボードのメモリマッピングについては「RZ/T1 グループ 初期設定 アプリケーションノート」に記載しています。

5.3.1 サンプルプログラムのセクション配置

サンプルプログラムで使用するセクションおよびサンプルプログラムの初期状態のセクション配置（ロードビュー）、スキッタローディング機能を使用後のセクション配置（実行ビュー）は、アプリケーションノート「RZ/T1 グループ初期設定」に記載しています。

5.3.2 MPU の設定

MPU の設定は、「RZ/T1 グループ 初期設定 アプリケーションノート」に記載しています。

5.3.3 例外処理ベクタテーブル

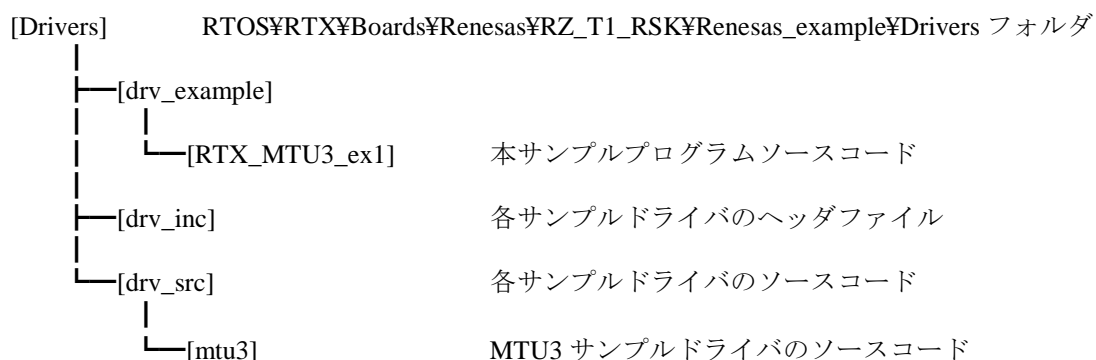
例外処理のベクタテーブルは、「RZ/T1 グループ 初期設定 アプリケーションノート」に記載しています。

5.3.4 必要メモリサイズ

必要メモリサイズは、RTX_ex1 サンプルプログラムの例が、「RZ/T1 グループ CMSIS-RTOS RTX for Cortex-R4 RTX サンプルプログラム アプリケーションノート」に記載されています。詳細な必要メモリサイズは、実際のプロジェクトを参照してください。

5.4 フォルダ構成

サンプルプログラム、各サンプルドライバのフォルダ構成を以下に示します。



5.5 使用割り込み一覧

表 5.2 サンプルプログラムで使用する割り込み

割り込み(要因 ID)	優先度	処理概要
コンペアマッチ割り込み(25)	7	コールバック関数をコール

5.6 固定幅整数一覧

表 5.3 サンプルプログラムで使用する固定幅整数

シンボル	内容
int8_t	8 ビット整数、符号あり（標準ライブラリにて定義）
int16_t	16 ビット整数、符号あり（標準ライブラリにて定義）
int32_t	32 ビット整数、符号あり（標準ライブラリにて定義）
uint8_t	8 ビット整数、符号なし（標準ライブラリにて定義）
uint16_t	16 ビット整数、符号なし（標準ライブラリにて定義）
uint32_t	32 ビット整数、符号なし（標準ライブラリにて定義）

5.7 定数一覧

以下に、サンプルプログラム / サンプルドライバで使用する定数、エラーコードを記します。

5.7.1 サンプルプログラムで使用する定数

表 5.4 サンプルプログラムで使用する定数

定数名	設定値	内容
MTU3_CFG_PARAM_CHECKING_ENABLE	1	MTU の API 関数にてパラメータチェック許可の有効（1）／無効（0）を表します。
MTU_COUNT_STOP	0	MTU カウンタ動作を停止させるための定数です。
MTU_CMPLMT_PWM_START	0xC0	相補 PWM を開始するための定数です。
MTU_C_SET_CYCLE	0xEA6	キャリア周期を設定するための定数です。
MTU_C_CYCLE	0x753	キャリア周期の 1/2 を設定するための定数です。
MTU_DEAD_TIME	0x4B	デッドタイムを設定するための定数です。
MTU_TGRA_CYCLE	MTU_C_CYCLE + MTU_DEAD_TIME	キャリア周期の 1/2+デッドタイムを設定するための定数です。
MTU_ENABLE_OUTPUT	1	MTU 出力を許可に設定するための定数です。
MTU_DISABLE_OUTPUT	0	MTU 出力を禁止に設定するための定数です。

5.7.2 サンプルプログラムで使用するエラーコード定数

表 5.5 サンプルプログラムで使用するエラーコード定数

定数名	設定値	内容
ESUCCESS	0	成功
EINVAL	-28	引数の不正
EACCES	-64	アクセス拒否
EFAULT	-97	失敗

5.8 構造体/共用体/列挙体一覧

図 5.2～図 5.14 にサンプルドライバで使用する構造体/共用体/列挙体を示します。

```
/* Enumeration of MTU channel numbers. */
typedef enum
{
    MTU_CHANNEL_0 = 0,
    MTU_CHANNEL_1,
    MTU_CHANNEL_2,
    MTU_CHANNEL_3,
    MTU_CHANNEL_4,
    MTU_CHANNEL_5,    /* This channel not support */
    MTU_CHANNEL_6,
    MTU_CHANNEL_7,
    MTU_CHANNEL_8,
    MTU_CHANNEL_MAX
} mtu_channel_t;

/* Clocking source selections. Index into register settings table. */
typedef enum mtu_clk_sources_e
{
    MTU_CLK_SRC_EXT_MTCLKA = 0x00,    // External clock input on MTCLKA pin
    MTU_CLK_SRC_EXT_MTCLKB = 0x01,    // External clock input on MTCLKB pin
    MTU_CLK_SRC_EXT_MTCLKC = 0x02,    // External clock input on MTCLKC pin
    MTU_CLK_SRC_EXT_MTCLKD = 0x03,    // External clock input on MTCLKD pin
    MTU_CLK_SRC_CASCADE = 0x04,    // Clock by overflow from other channel counter. (only on
    certain channels)
    MTU_CLK_SRC_INTERNAL              // Use internal clock (PCLK)
} mtu_clk_sources_t;
```

図 5.2 サンプルドライバで使用する構造体/共用体/列挙体 (1)

```
/* The possible settings for MTU output pins. Register setting values. */
typedef enum mtu_output_states_e
{
    MTU_PIN_NO_OUTPUT = 0x0,    // Output high impedance.
    MTU_PIN_LO_GOLO    = 0x1,    // Initial output is low. Low output at compare match.
    MTU_PIN_LO_GOHI    = 0x2,    // Initial output is low. High output at compare match.
    MTU_PIN_LO_TOGGGLE = 0x3,    // Initial output is low. Toggle (alternate) output at compare match.
    MTU_PIN_HI_GOLO    = 0x5,    // Initial output is high. Low output at compare match.
    MTU_PIN_HI_GOHI    = 0x6,    // Initial output is high. High output at compare match.
    MTU_PIN_HI_TOGGGLE = 0x7     // Initial output is high. Toggle (alternate) output at compare match.
} mtu_output_states_t;

/* The possible settings for counting clock active edge. Register setting values. */
typedef enum mtu_clk_edges_e
{
    MTU_CLK_RISING_EDGE  = 0x00,
    MTU_CLK_FALLING_EDGE = 0x08,
    MTU_CLK_ANY_EDGE     = 0x10,
} mtu_clk_edges_t;
```

図 5.3 サンプルドライバで使用する構造体/共用体/列挙体 (2)

```

/* The possible counter clearing source selections. Index into register settings table. */
typedef enum mtu_clear_src_e
{
    MTU_CLR_TIMER_A = 0,    // Clear the channel counter on the "A" compare or capture event.
    MTU_CLR_TIMER_B,        // Clear the channel counter on the "B" compare or capture event.
    MTU_CLR_TIMER_C,        // Clear the channel counter on the "C" compare or capture event.
    MTU_CLR_TIMER_D,        // Clear the channel counter on the "D" compare or capture event.
    MTU_CLR_SYNC,           // Clear the channel counter when another sync'ed channel clears.
    MTU_CLR_DISABLED        // Never clear the channel counter.
} mtu_clear_src_t;

/* PCLK divisor for internal clocking source. Index into register settings table. */
typedef enum mtu_pclk_divisor_e
{
    MTU_SRC_CLK_DIV_1 = 0, // PCLK/1
    MTU_SRC_CLK_DIV_2,     // PCLK/2
    MTU_SRC_CLK_DIV_4,     // PCLK/4
    MTU_SRC_CLK_DIV_8,     // PCLK/8
    MTU_SRC_CLK_DIV_16,    // PCLK/16
    MTU_SRC_CLK_DIV_32,    // PCLK/32
    MTU_SRC_CLK_DIV_64,    // PCLK/64
    MTU_SRC_CLK_DIV_256,   // PCLK/256
    MTU_SRC_CLK_DIV_1024  // PCLK/1024
} mtu_src_clk_divisor_t;

/* Actions to be done upon timer or capture event. Multiple selections to be ORed together. */
typedef enum mtu_actions_e
{
    MTU_ACTION_NONE      = 0x00,    // Do nothing with this timer.
    MTU_ACTION_OUTPUT     = 0x01,    // Change state of output pin.
    MTU_ACTION_INTERRUPT  = 0x02,    // Generate interrupt request.
    MTU_ACTION_CALLBACK   = 0x04,    // Generate interrupt request and execute user-defined callback on
                                     // interrupt.
    MTU_ACTION_REPEAT     = 0x10,    // Continuously repeat the timer cycle and actions
    MTU_ACTION_TRIGGER_ADC = 0x20,    // Trigger ADC on this event. Timer A events only.
    MTU_ACTION_CAPTURE    = 0x40,    // Default input capture action. Placeholder value, does not to be
    specified.
} mtu_actions_t;

```

図 5.4 サンプルドライバで使用する構造体/共用体/列挙体 (3)

```
/****** Type defines used with the R_MTU_Control function. *****/
/* Control function command codes. */
typedef enum mtu_cmd_e
{
    MTU_CMD_START,           // Activate clocking
    MTU_CMD_STOP,            // Pause clocking
    MTU_CMD_SAFE_STOP,       // Stop clocking and set outputs to safe state
    MTU_CMD_RESTART,         // Zero the counter then resume clocking
    MTU_CMD_SYNCHRONIZE,     // Specify channels to group for synchronized clearing.
    MTU_CMD_GET_STATUS,      // Retrieve the current status of the channel
    MTU_CMD_SET_CAPT_EDGE,   // Sets the detection edge polarity for input capture.
    MTU_CMD_UNKNOWN         // Not a valid command.
} mtu_cmd_t;

/* Used as bit-field identifiers to identify channels assigned to a group for group operations.
 * Add multiple channels to group by ORing these values together. */
typedef enum
{
    MTU_GRP_CH0 = 0x0001,
    MTU_GRP_CH1 = 0x0002,
    MTU_GRP_CH2 = 0x0004,
    MTU_GRP_CH3 = 0x0040,
    MTU_GRP_CH4 = 0x0080,
    PROHIBTID = 0x0000,    /* This channel not support */
    MTU_GRP_CH6 = 0x4000,
    MTU_GRP_CH7 = 0x8000,
    MTU_GRP_CH8 = 0x0008
} mtu_group_t;
```

図 5.5 サンプルドライバで使用する構造体/共用体/列挙体 (4)


```
typedef struct mtu_timer_status_s
{
    uint32_t timer_count;           // The current channel counter value.
    Bool_t   timer_running;        // True = timer currently counting, false = counting stopped.
} mtu_timer_status_t;

typedef struct mtu_capture_status_s
{
    uint32_t capt_a_count;          // The count at input capture A event.
    uint32_t capt_b_count;          // The count at input capture B event.
    uint32_t capt_c_count;          // The count at input capture C event.
    uint32_t capt_d_count;          // The count at input capture D event.
    uint32_t timer_count;           // The current channel counter value.
} mtu_capture_status_t;

typedef struct mtu_pwm_status_s
{
    bool_t   running;
    uint16_t pwm_timer_count;        // The current channel counter value.
    uint16_t pwm_a_value;            // The count at input capture A event.
    uint16_t pwm_b_value;            // The count at input capture B event.
    uint16_t pwm_c_value;            // The count at input capture C event.
    uint16_t pwm_d_value;            // The count at input capture D event.
} mtu_pwm_status_t;
```

図 5.6 サンプルドライバで使用する構造体/共用体/列挙体 (5)

```

/***** Type defines used for callback functions. *****/
/* Specifies the timer to which an operation is associated. Returned in callback data structure. */
typedef enum
{
    MTU_TIMER_A = 0,    //Corresponds to MTU TGRA register operations
    MTU_TIMER_B,        //Corresponds to MTU TGRB register operations
    MTU_TIMER_C,        //Corresponds to MTU TGRC register operations
    MTU_TIMER_D,        //Corresponds to MTU TGRD register operations
    MTU_TIMERS_MAX
} mtu_timer_num_t;

/***** Type defines used for callback functions. *****/
/* Data structure passed to User callback upon pwm interrupt. */
typedef struct mtu_callback_data_s
{
    mtu_channel_t    channel;
    mtu_timer_num_t timer_num;
    uint32_t         count;
} mtu_callback_data_t;

/***** Type defines used with the R_MTU_Timer_Open and R_MTU_Capture_Open functions. *****/
typedef struct mtu_timer_clk_src_s
{
    mtu_clk_sources_t    source;    // Internal clock or external clock input
    mtu_clk_edges_t      clock_edge; // Specify the clock active edge.
} mtu_clk_src_t;

```

図 5.7 サンプルドライバで使用する構造体/共用体/列挙体 (6)

```

/***** Type defines used with the R_MTU_Capture_Open function. *****/
typedef enum
{
    MTU_CAP_SRC_A = 0,
    MTU_CAP_SRC_B,
    MTU_CAP_SRC_C,
    MTU_CAP_SRC_D
} mtu_cap_src_t;

/* The possible settings for input capture signal active edge. Register setting values. */
typedef enum mtu_cap_edges_e
{
    MTU_CAP_RISING_EDGE = 0x08,
    MTU_CAP_FALLING_EDGE = 0x09,
    MTU_CAP_ANY_EDGE = 0x0A,
} mtu_cap_edges_t;

typedef struct mtu_capture_set_edge_s // Used with the MTU_TIMER_CMD_SET_CAPT_EDGE command.
{
    mtu_cap_src_t    capture_src;        // The capture source.
    mtu_cap_edges_t  capture_edge;       // Specify transition polarities.
} mtu_capture_set_edge_t;

typedef struct mtu_capture_settings_s
{
    mtu_actions_t    actions;
    mtu_cap_edges_t  capture_edge;       // Specify transition polarities.
    bool_t           filter_enable;      // Noise filter on or off.
} mtu_capture_settings_t;

```

図 5.8 サンプルドライバで使用する構造体/共用体/列挙体 (7)

```

typedef struct mtu_capture_chnl_settings_s
{
    mtu_clk_src_t      clock_src;    // Specify clocking source.
    mtu_src_clk_divisor_t  clock_div; // Internal clock divisor selection.
    mtu_clear_src_t      clear_src;  // Specify the counter clearing source.
    mtu_capture_settings_t capture_a;
    mtu_capture_settings_t capture_b;
    mtu_capture_settings_t capture_c;
    mtu_capture_settings_t capture_d;
} mtu_capture_chnl_settings_t;

/***** Type defines used with the R_MTU_Timer_Open function. *****/
typedef struct mtu_timer_actions_config_s
{
    mtu_actions_t      do_action; // Various actions that can be done at timer event.
    mtu_output_states_t output;    // Output pin transition type when output action is selected.
} mtu_timer_actions_cfg_t;

typedef struct mtu_timer_settings_s
{
    uint32_t  freq; // If internal clock source, the desired event frequency, or if external the Compare-match count.
    mtu_timer_actions_cfg_t actions;
} mtu_timer_settings_t;

typedef struct mtu_timer_chnl_settings_s
{
    mtu_clk_src_t  clock_src;    // Specify clocking source.
    mtu_clear_src_t  clear_src;  // Specify the counter clearing source.
    mtu_timer_settings_t timer_a;
    mtu_timer_settings_t timer_b;
    mtu_timer_settings_t timer_c;
    mtu_timer_settings_t timer_d;
} mtu_timer_chnl_settings_t;

```

図 5.9 サンプルドライバで使用する構造体/共用体/列挙体 (8)

```

/***** Type defines used with the R_MTU_PWM_Open function. *****/
/* Available PWM operating modes. */
typedef enum mtu_pwm_mode_e
{
    MTU_PWM_MODE_1 = 0x02,
    MTU_PWM_MODE_2 = 0x03
} mtu_pwm_mode_t;

typedef struct mtu_pwm_settings_s
{
    uint16_t          duty;
    mtu_actions_t     actions;
    mtu_output_states_t outputs;    // Specify transition polarities.
} mtu_pwm_settings_t;

typedef struct mtu_pwm_chnl_settings_s
{
    mtu_clk_src_t      clock_src;    // Specify clocking source.
    uint32_t           cycle_freq;    // Cycle frequency for the channel
    mtu_clear_src_t     clear_src;    // Specify the counter clearing source.
    mtu_pwm_mode_t      pwm_mode;    // Specify mode 1 or mode 2
    mtu_pwm_settings_t  pwm_a;
    mtu_pwm_settings_t  pwm_b;
    mtu_pwm_settings_t  pwm_c;
    mtu_pwm_settings_t  pwm_d;
} mtu_pwm_chnl_settings_t;

```

図 5.10 サンプルドライバで使用する構造体/共用体/列挙体 (9)

```

/***** Type defines used with the R_MTU_PWM_Compliment_Open function. *****/
typedef enum
{
    MTU_CHANNEL_3_4 = 0,
    MTU_CHANNEL_6_7,
    MTU_CMPL_PWM_CHANNEL_MAX
} mtu_cmpl_pwm_channel_t;

typedef enum
{
    MTU_TOGGLE_OFF = 0x00, // Output toggle OFF
    MTU_TOGGLE_ON  = 0x01, // Output toggle ON
} mtu_cmpl_pwm_toggle_t;

typedef enum
{
    MTU_CMPL_PWM_MODE_1 = 0x0D, // Complimentary PWM mode 1
    MTU_CMPL_PWM_MODE_2 = 0x0E, // Complimentary PWM mode 2
    MTU_CMPL_PWM_MODE_3 = 0x0F, // Complimentary PWM mode 3
} mtu_cmpl_pwm_mode_t;

typedef enum
{
    MTU_PIN_P_N_1 = 0x00, // TOCR1
    MTU_PIN_P_N_2 = 0x01, // TOCR2
} mtu_cmpl_pwm_p_n_t;

typedef enum
{
    MTU_PIN_P_N_BF_OFF          = 0x00, // Does not transfer
    MTU_PIN_P_N_BF_CREST        = 0x01, // Transfer in TCNT Crest
    MTU_PIN_P_N_BF_TROUGH       = 0x02, // Transfer in TCNT trough
    MTU_PIN_P_N_BF_CREST_TROUGH = 0x03, // Transfer in TCNT crest and trough
} mtu_cmpl_pwm_p_n_bf_t;

```

図 5.11 サンプルドライバで使用する構造体/共用体/列挙体 (10)

```
typedef enum
{
    MTU_CMPL_PWM_D_BF_OFF = 0, // OFF
    MTU_CMPL_PWM_D_BF_ON      // ON
} mtu_cmpl_pwm_d_bf_t;

typedef enum
{
    MTU_PIN_OLSN_HI_UPHI_DNLO = 0x00, // Init High Active Low Up High Down Low
    MTU_PIN_OLSN_LO_UPLO_DNHI = 0x01, // Init Low Active High Up Low Down High
} mtu_cmpl_pwm_olsn_t;

typedef enum
{
    MTU_PIN_OLSP_HI_UPLO_DNHI = 0x00, // Init High Active Low Up Low Down High
    MTU_PIN_OLSP_LO_UPHI_DNLO = 0x01, // Init Low Active High Up High Down Low
} mtu_cmpl_pwm_olsp_t;

typedef enum
{
    MTU_CMPL_PWM_ST_COUNT_OFF = 0,
    MTU_CMPL_PWM_ST_COUNT_ON,
} mtu_cmpl_pwm_count_st_t;

typedef enum
{
    MTU_CMPL_PWM_DIRECT_DOWN = 0,
    MTU_CMPL_PWM_DIRECT_UP,
} mtu_cmpl_pwm_direction_t;
```

図 5.12 サンプルドライバで使用する構造体/共用体/列挙体 (11)

```
typedef enum
{
    MTU_CMPL_PWM_OUTPUT_OFF    = 0,
    MTU_CMPL_PWM_OUTPUT_ON,
} mtu_cmpl_pwm_output_st_t;

typedef enum
{
    MTU_CMPL_PWM_PROTECT_OFF    = 0,
    MTU_CMPL_PWM_PROTECT_ON,
} mtu_cmpl_pwm_reg_protect_t;

typedef struct mtu_cmpl_pwm_clk_div_s
{
    mtu_src_clk_divisor_t clock_div;    // Internal clock divisor selection.
    uint16_t               cycle_freq;   // Cycle
} mtu_cmpl_pwm_clk_div_t;

typedef struct mtu_cmpl_pwm_settings_s
{
    mtu_cmpl_pwm_olsp_t      olsp; // Output Level Select P
    mtu_cmpl_pwm_olsn_t      olsn; // Output Level Select N
    uint16_t                 duty; // Duty cycle (Unit 0.1%)
    mtu_cmpl_pwm_output_st_t output; // PWM output
} mtu_cmpl_pwm_settings_t;
```

図 5.13 サンプルドライバで使用する構造体/共用体/列挙体 (12)


```
typedef struct mtu_cmpl_pwm_chnl_settings_s
{
    mtu_clk_src_t          clock_src;    // Specify clocking source.
    mtu_cmpl_pwm_clk_div_t  clk_div;     // Internal clock divisor selection.
    uint16_t               dead_time;    // Dead time
    mtu_cmpl_pwm_toggle_t   toggle;      // Output toggle
    mtu_cmpl_pwm_mode_t     mode;        // Complimentary PWM mode
    mtu_cmpl_pwm_p_n_t      p_n;        // TOC Select
    mtu_cmpl_pwm_p_n_bf_t   p_n_bf;     // Buffer output level
    mtu_cmpl_pwm_d_bf_t     d_bf;       // Double buffer select
    mtu_cmpl_pwm_reg_protect_t protect;   // register protect
    mtu_cmpl_pwm_settings_t pwm_output_1; // PWM output 1
    mtu_cmpl_pwm_settings_t pwm_output_2; // PWM output 2
    mtu_cmpl_pwm_settings_t pwm_output_3; // PWM output 3
} mtu_cmpl_pwm_chnl_settings_t;

typedef struct mtu_cmpl_pwm_chnl_status_s
{
    mtu_cmpl_pwm_count_st_t  c_st;
    mtu_cmpl_pwm_direction_t d_st;
} mtu_cmpl_pwm_chnl_status_t;
```

図 5.14 サンプルドライバで使用する構造体/共用体/列挙体 (13)

5.9 関数一覧

以下に、関数一覧を示します。

表 5.6 関数一覧

関数名	ページ番号
main	27
SER_Init	27
mtu3_setup	27
R_MTU_Init	28
R_MTU_Uninit	28
R_MTU_PWM_Compliment_Open	28
R_MTU_PWM_Compliment_Close	28
R_MTU_PWM_Compliment_Control	29
R_MTU_Timer_Open	29
R_MTU_Capture_Open	30
R_MTU_PWM_Open	30
R_MTU_Close	30
R_MTU_Control	31
R_MTU_GetVersion	31

5.10 サンプルプログラム関数仕様

5.10.1 main

main	
ヘッダ	
宣言	int32_t main(void)
説明	相補 PWM モード 1 の設定を行い、MTU3、MTU4 を使用して 3 相 PWM を出力します。
引数	なし
戻り値	本関数は無限ループによりリターンしません。
補足	-

5.10.2 SER_Init

SER_init	
ヘッダ	Serial.h
宣言	void SERInit(void)
説明	シリアルの初期化を行います。
引数	なし
戻り値	なし
補足	-

5.10.3 mtu3_setup

mtu3setup	
ヘッダ	
宣言	int32_t mtu3_setup(void)
説明	相補 PWM の設定を行います。
引数	なし
戻り値	ESUCCESS、EACCESS、EFAULT、EBUSY
補足	-

5.11 MTU ドライバ関数仕様

5.11.1 R_MTU_Init

R_MTU_Init	
ヘッダ	r_mtu3_if.h
宣言	void R_MTU_Init(void)
説明	MTU ドライバの初期化を行います。
引数	なし
戻り値	なし
補足	-

5.11.2 R_MTU_Uninit

R_MTU_Uninit	
ヘッダ	r_mtu3_if.h
宣言	int32_t R_MTU_Uninit(void)
説明	MTU ドライバの終了を行い、MTU3a デバイスを停止します。
引数	なし
戻り値	ESUCCESS、EACCESS、EFAULT
補足	-

5.11.3 R_MTU_PWM_Compliment_Open

R_MTU_PWM_Compliment_Open	
概要	相補 PWM モード設定処理
ヘッダ	r_mtu3_if.h
宣言	int32_t R_MTU_PWM_Compliment_Open (mtu_cmpl_pwm_channel_t const channel, mtu_cmpl_pwm_chnl_settings_t * const pconfig)
説明	相補 PWM モード出力動作開始する為の設定を行います。
引数	channel 相補 PWM モードチャンネル MTU_CHANNEL_3_4 MTU_CHANNEL_6_7 pconfig 相補 PWM モード設定情報のポインタ
戻り値	ESUCCESS、EACCESS、EFAULT、EINVAL、EBUSY
補足	R_MTU_Init を実行し、MTU ドライバを初期化して下さい。 相補 PWM モード機能の終了後、R_MTU_PWM_Compliment_Close 関数を呼び出して終了して下さい。

5.11.4 R_MTU_PWM_Compliment_Close

R_MTU_PWM_Compliment_Close	
概要	相補 PWM 終了処理
ヘッダ	r_mtu3_if.h
宣言	int32_t R_MTU_PWM_Compliment_Close(mtu_cmpl_pwm_channel_t const channel)
説明	相補 PWM モード出力動作を終了します。
引数	channel 相補 PWM モードチャンネル MTU_CHANNEL_3_4 MTU_CHANNEL_6_7
戻り値	ESUCCESS、EACCESS、EFAULT、EINVAL、EBUSY
補足	R_MTU_PWM_Compliment_Open 関数を実行してチャンネルがオープン状態以上である必要があります。

5.11.5 R_MTU_PWM_Compliment_Control

R_MTU_PWM_Compliment_Control	
概 要	相補 PWM 制御処理
ヘッダ	r_mtu3_if.h
宣 言	int32_t R_MTU_PWM_Compliment_Control(mtu_cmpl_pwm_channel_t const channel, mtu_cmd_t const cmd, void * const pcmd_data)
説 明	指定チャンネルの相補 PWM モードの機能設定を行います。 コマンドの詳細については、5.14 章を参照下さい。
引 数	channel 相補 PWM モードチャンネル MTU_CHANNEL_3_4 MTU_CHANNEL_6_7 cmd 相補 PWM コントロールコマンド MTU_CMD_START MTU_CMD_STOP MTU_CMD_GET_STATUS pcmd_data 相補 PWM コントロールコマンドパラメータ情報のポインタ
戻り値	ESUCCESS、EINVAL、EACCESS、EFAULT、EBUSY
補足	R_MTU_PWM_Compliment_Open 関数を実行してチャンネルがオープン状態以上である必要があります。

5.11.6 R_MTU_Timer_Open

R_MTU_Timer_Open	
概 要	MTU コンペア/マッチ設定処理
ヘッダ	r_mtu3_if.h
宣 言	int32_t R_MTU_Timer_Open(mtu_channel_t const channel, mtu_timer_chnl_settings_t * const pconfig, void (* const pcallback)(void * pdata))
説 明	指定した MTU チャンネル毎のコンペア/マッチタイミグ操作の為の設定を行います。設定情報のポインタ (pconfig) の詳細については、5.15 章を参照ください。
引 数	channel MTU チャンネル : MTU_CHANNEL_0 ~ MTU_CHANNEL_8 pconfig コンペア/マッチタイミグ設定情報のポインタ pcallback コールバック関数のポインタ
リターン値	ESUCCESS、EINVAL、EACCESS、EFAULT、EBUSY
補足	R_MTU_Init を実行し、MTU ドライバを初期化して下さい。 MTU コンペア/マッチタイミグ機能の終了後、R_MTU_Close 関数を呼び出して終了して下さい。

5.11.7 R_MTU_Capture_Open

R_MTU_Capture_Open	
概 要	MTU キャプチャ設定処理
ヘッダ	r_mtu3_if.h
宣 言	int32_t R_MTU_Capture_Open(mtu_channel_t const channel, mtu_capture_chnl_settings_t * const pconfig, void (* const pcallback)(void * pdata))
説 明	指定した MTU チャンネル毎のキャプチャ操作の為の設定を行います。キャプチャ設定情報のポインタ (pconfig) の詳細は、5.16 章を参照下さい。
引 数	channel MTU チャンネル : MTU_CHANNEL_0 ~ MTU_CHANNEL_8 pconfig キャプチャ設定情報のポインタ pcallback コールバック関数のポインタ
戻り値	ESUCCESS、EINVAL、EACCESS、EFAULT、EBUSY
補足	R_MTU_Init を実行し、MTU ドライバを初期化して下さい。 MTU キャプチャ機能の終了後、R_MTU_Close 関数を呼び出して終了して下さい。

5.11.8 R_MTU_PWM_Open

R_MTU_PWM_Open	
概 要	MTU PWM 設定処理
ヘッダ	r_mtu3_if.h
宣 言	int32_t R_MTU_PWM_Open(mtu_channel_t const channel, mtu_pwm_chnl_settings_t * const pconfig, void (* const pcallback)(void * pdata))
説 明	指定した MTU チャンネル毎の基本的な PWM モード動作の為の設定を行います。PWM モード設定情報 (pconfig) については、5.13 章を参照ください。
引 数	channel MTU チャンネル : MTU_CHANNEL_0 ~ MTU_CHANNEL_8 注 PWM モード 2 を指定した場合はチャンネル 0、1、2 のみが設定可能です。 pconfig PWM モード設定情報のポインタ pcallback コールバック関数のポインタ
戻り値	ESUCCESS、EINVAL、EACCESS、EFAULT、EBUSY
補足	R_MTU_Init を実行し、MTU ドライバを初期化して下さい。 MTU PWM 機能の終了後、R_MTU_Close 関数を呼び出して終了して下さい。

5.11.9 R_MTU_Close

R_MTU_Close	
概 要	MTU 終了処理
ヘッダ	r_mtu3_if.h
宣 言	int32_t R_MTU_Close (mtu_channel_t const channel)
説 明	指定した MTU チャンネルの動作を終了します。
引 数	channel MTU チャンネル : MTU_CHANNEL_0 ~ MTU_CHANNEL_8
戻り値	ESUCCESS、EINVAL、EACCESS、EFAULT、EBUSY
補足	R_MTU_Timer_Open 関数、R_MTU_Capture_Open 関数、R_MTU_PWM_Open 関数を実行して MTU チャンネルがオープン状態以上である必要があります。

5.11.10 R_MTU_Control

R_MTU_Control	
概 要	MTU 制御処理
ヘッダ	r_mtu3_if.h
宣 言	int32_t R_MTU_Control(mtu_channel_t const channel, mtu_cmd_t const cmd, void * const pcmd_data)
説 明	指定した MTU チャンネル毎の動作設定を行います。 コマンドの詳細については、5.18 章を参照ください。
引 数	channel MTU チャンネル：MTU_CHANNEL_0 ~ MTU_CHANNEL_8
	cmd MTU コントロールコマンド MTU_CMD_START MTU_CMD_STOP MTU_CMD_SAFE_STOP MTU_CMD_RESTART MTU_CMD_SYNCHRONIZE MTU_CMD_GET_STATUS MTU_CMD_SET_CAPT_EDGE
	pcmd_data MTU コントロールコマンドパラメータ情報のポインタ
戻り値	ESUCCESS、EINVAL、EACCESS、EFAULT、EBUSY
補足	R_MTU_Timer_Open 関数、R_MTU_Capture_Open 関数及び、R_MTU_PWM_Open 関数を実行して MTU チャンネルがオープン状態以上である必要があります。

5.11.11 R_MTU_GetVersion

R_MTU_GetVersion	
概 要	バージョンの取得
ヘッダ	r_mtu3_if.h
宣 言	uint32_t R_MTU_GetVersion(void)
説 明	MTU ドライバのバージョン情報を取得します。
引 数	なし
戻り値	バージョン情報 0-15 :マイナーバージョン情報
	16-31 :メジャーバージョン情報
補足	-

5.12 フローチャート

5.12.1 サンプルアプリケーションメイン処理

図 5.15 にサンプルアプリケーションのメイン処理のフローチャートを示します。



図 5.15 サンプルアプリケーションメイン処理

5.12.2 相補 PWM モード制御処理

図 5.16 に相補 PWM モード設定処理のフローチャートを示します。

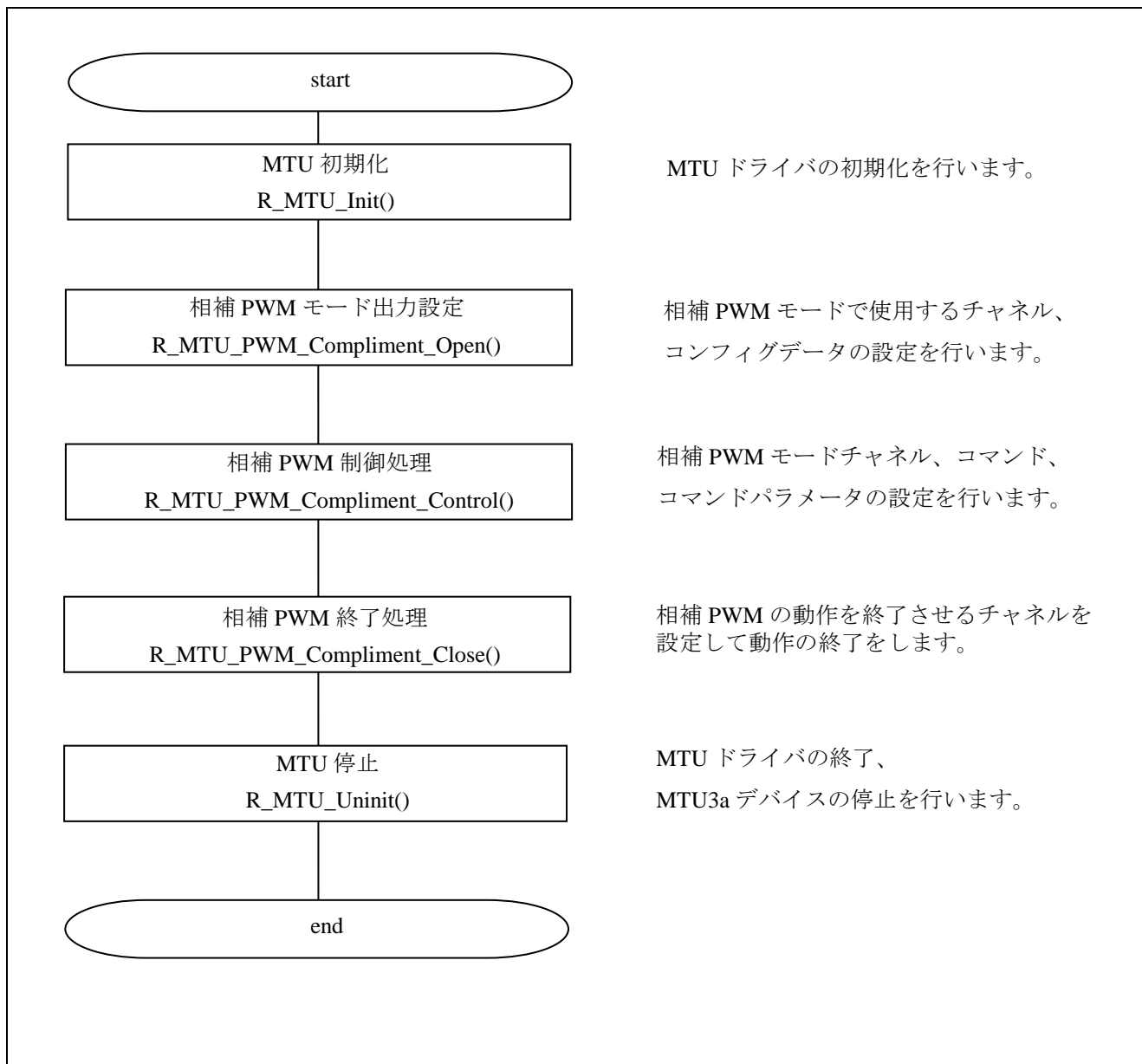


図 5.16 相補 PWM モード設定処理

5.12.3 MTU コンペア／マッチタイミング処理

図 5.17 にコンペア／マッチタイミング処理のフローチャートを示します。

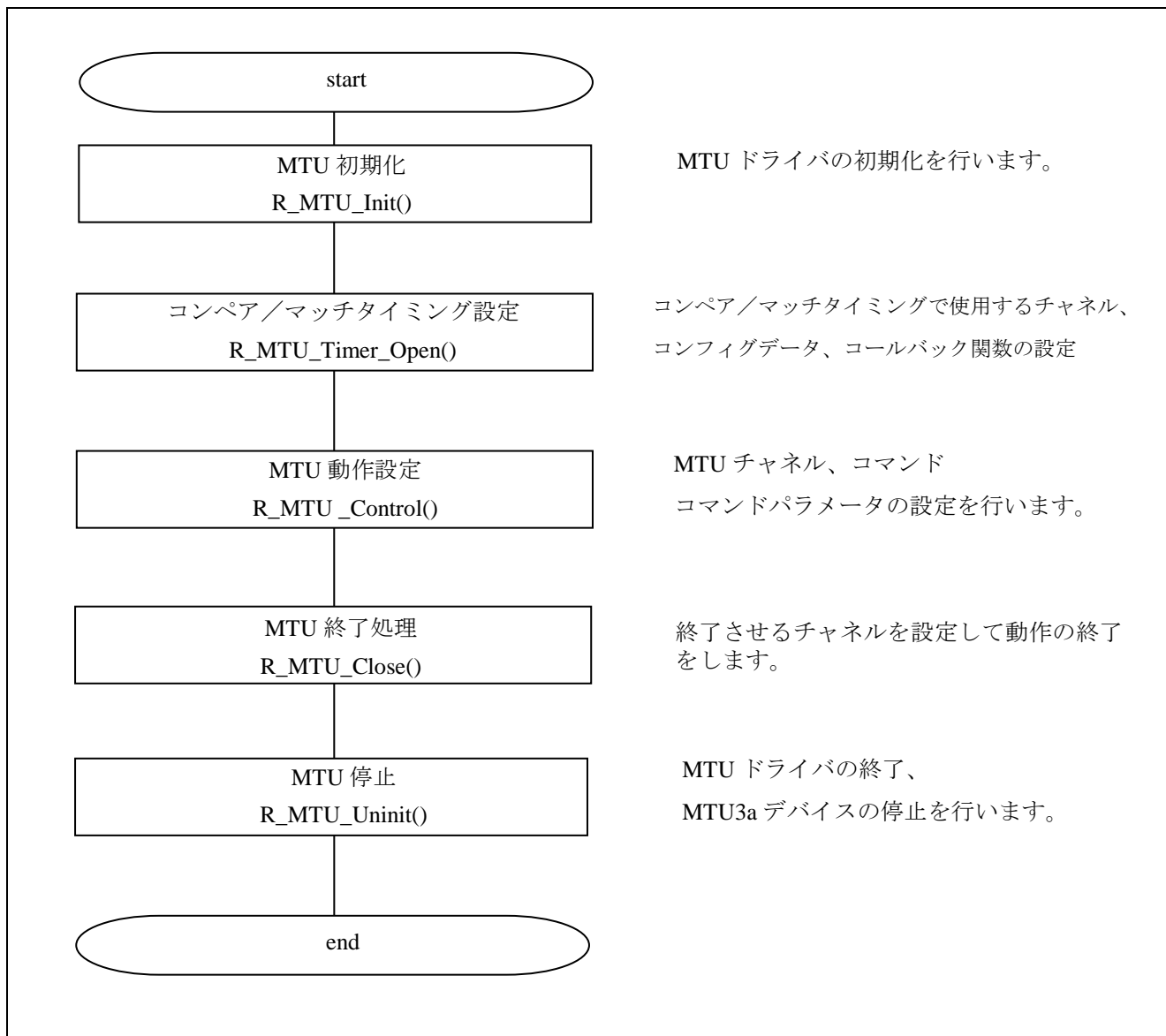


図 5.17 コンペア／マッチタイミング処理

5.12.4 MTU キャプチャ処理

図 5.18 キャプチャ処理のフローチャートを示します。

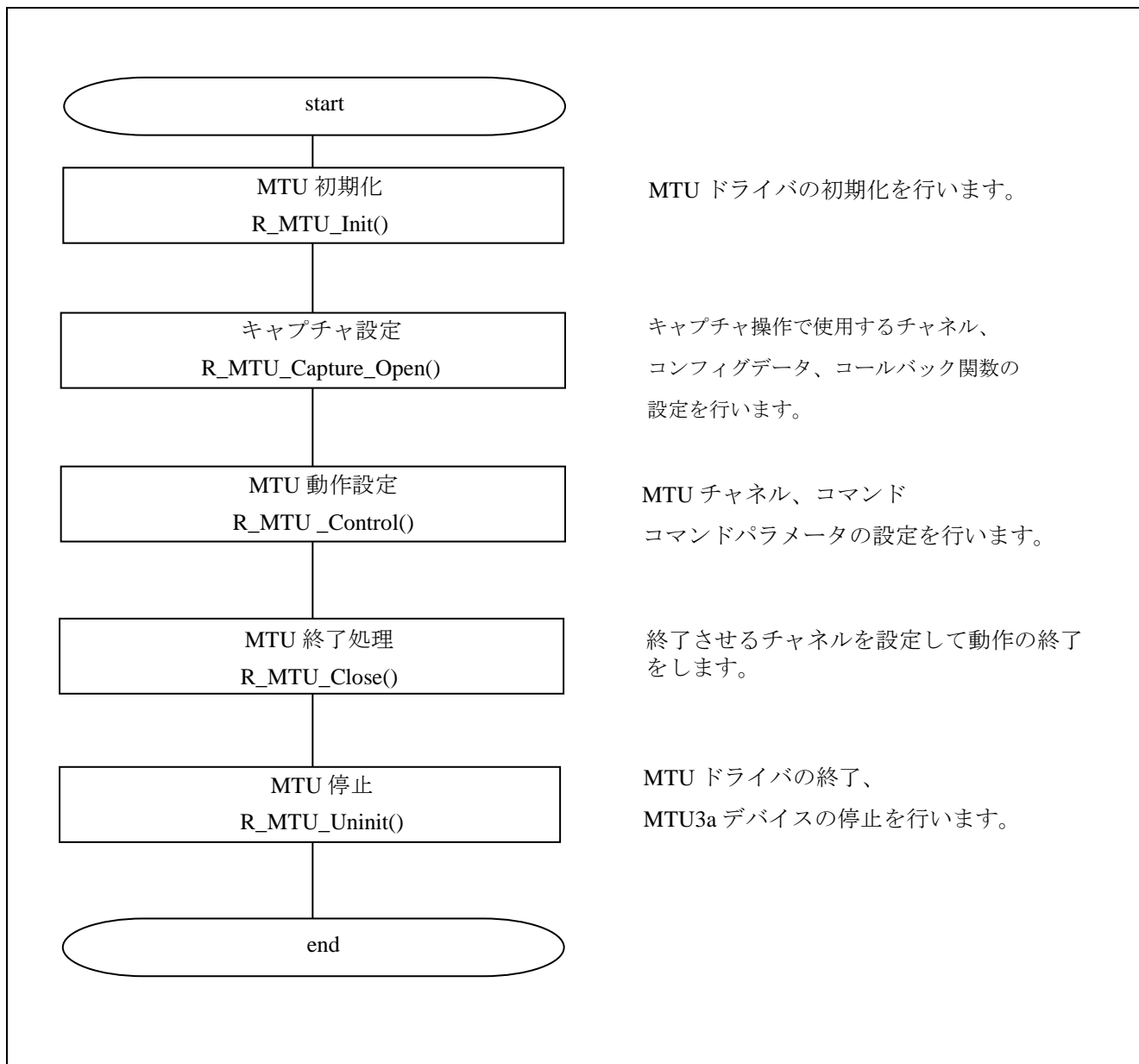


図 5.18 キャプチャ処理

5.12.5 PWM モード制御処理

図 5.19 に PWM モード制御処理のフローチャートを示します。

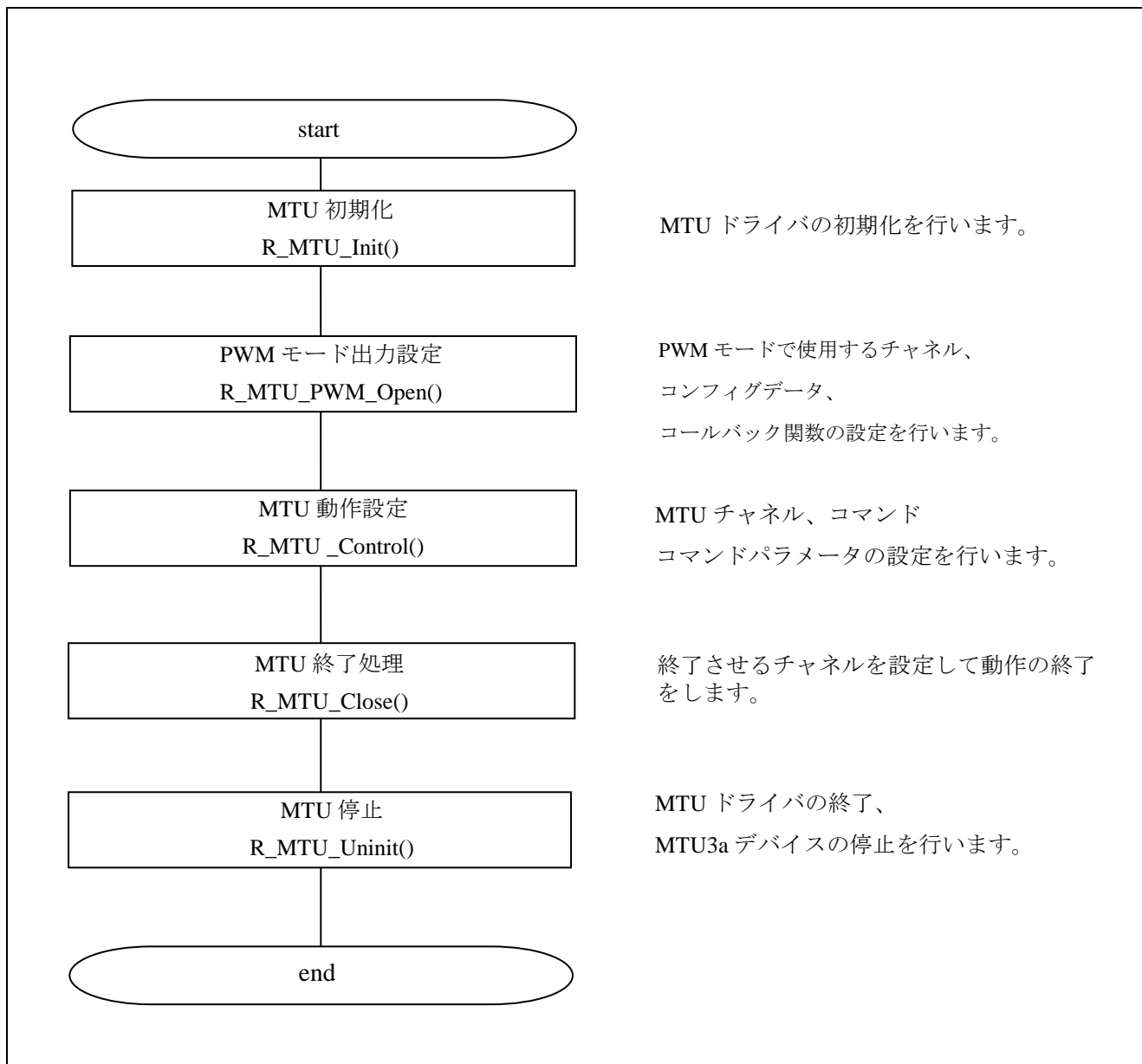


図 5.19 PWM モード制御処理

5.13 R_MTU_PWM_Compliment_Open パラメータ一覧

R_MTU_PWM_Compliment_Open 関数で使用するパラメータ (*pconfig) 一覧を以下に示します。

表 5.7 R_MTU_PWM_Compliment_Open パラメータ一覧

パラメータ	概要
clock_src.source	カウントクロックを設定します。
clock_src.clock_edge	クロックエッジを設定します。
clk_div.clock_div	カウントクロックの分周比を設定します。
clk_div.cycle_freq	クロックカウント数を設定します。
dead_time	デッドタイムを設定します。
toggle	キャリア周期に同期したトグル出力を設定します。
mode	相補 PWM の動作モードを設定します。
p_n	出力レベルの動作を設定します。
p_n_bf	出力レベルのバッファ操作を設定します。
d_bf	ダブルバッファ機能の使用有無を設定します。
protect	MTU3、MTU4 の誤書き込み防止設定をします。
pwm_output_X.olsp (X = 1, 2, 3)	PWM 出力 X (X = 1, 2, 3) の正相の出力レベルを設定します。
pwm_output_X.olsn (X = 1, 2, 3)	PWM 出力 X (X = 1, 2, 3) の逆相の出力レベルを設定します。
pwm_output_X.duty (X = 1, 2, 3)	PWM 出力 X (X = 1, 2, 3) のデューティ比を設定します。
pwm_output_X.output (X = 1, 2, 3)	PWM 出力 X (X = 1, 2, 3) の端子出力の許可／禁止を設定します。

各パラメータの詳細を以下に記します。

5.13.1 clock_src.source

clock_src.source	
概 要	カウントクロックを設定します。
ヘッダ	r_mtu3_if.h
説 明	MTU のチャネル(MTU3 と MTU4、または MTU6 と MTU7)におけるカウントクロックを設定します。 外部クロックを選択する場合は、あらかじめ該当端子の I/O ポート設定およびマルチファンクションピンコントローラ (MPC) の設定が必要です。
パラメータ	MTU_CLK_SRC_EXT_MTCLKA ^{注1} 外部クロック (MTCLKA 端子入力) を指定します。 MTU_CLK_SRC_EXT_MTCLKB ^{注1} 外部クロック (MTCLKB 端子入力) を指定します。 MTU_CLK_SRC_INTERNAL 内部クロック (PCLKC) を指定します。
補足	注 1: チャネル 3 と 4 選択時のみ設定可能です。

5.13.2 clock_src.clock_edge

clock_src.clock_edge	
概 要	クロックエッジを設定します。
ヘッダ	r_mtu3_if.h
説 明	MTU のチャネル (MTU3 と MTU4、または MTU6 と MTU7) における入力クロックをカウントするエッジを設定します。
パラメータ	MTU_CLK_RISING_EDGE 立ち上がりエッジでカウントに指定します。 MTU_CLK_FALLING_EDGE 立ち下がりエッジでカウントに指定します。 MTU_CLK_ANY_EDGE 両エッジでカウントに指定します。
補足	指定したタイマ周期 (clk_div) によりカウントクロックが PCLKC/1 となる場合、エッジの指定は無視され初期値 (立ち上がりエッジ) となります。

5.13.3 clk_div.clock_div

clk_div.clock_div		
概要	カウントクロックのクロックの分周比を設定します。	
ヘッダ	r_mtu3_if.h	
説明	MTU のチャネル（MTU3 と MTU4、または MTU6 と MTU7）のクロック分周比を選択します。	
パラメータ	MTU_SRC_CLK_DIV_1	PCLKC/1 でカウント
	MTU_SRC_CLK_DIV_2	PCLKC/2 でカウント
	MTU_SRC_CLK_DIV_4	PCLKC/4 でカウント
	MTU_SRC_CLK_DIV_8	PCLKC/8 でカウント
	MTU_SRC_CLK_DIV_16	PCLKC/16 でカウント
	MTU_SRC_CLK_DIV_32	PCLKC/32 でカウント
	MTU_SRC_CLK_DIV_64	PCLKC/64 でカウント
	MTU_SRC_CLK_DIV_256	PCLKC/256 でカウント
	MTU_SRC_CLK_DIV_1024	PCLKC/1024 でカウント
補足	-	

5.13.4 clk_div.cycle_freq

clk_div.cycle_freq		
概要	相補 PWM のキャリア周期を設定します。	
ヘッダ	r_mtu3_if.h	
説明	MTU のチャネル（MTU3 と MTU4、または MTU6 と MTU7）のキャリア周期の設定を行います。 設定した値の 1/2 がタイマ周期データレジスタ（TCDRA, TCDRB）に設定されます。	
パラメータ	数値	キャリア周期の設定値(2~1FFFEh)
補足	-	

5.13.5 dead_time

dead_time		
概要	相補 PWM のデッドタイムを設定します。	
ヘッダ	r_mtu3_if.h	
説明	相補 PWM のデッドタイムとして、デッドタイムデータレジスタ（TDDRA、TDDRB）への設定値を指定します。	
パラメータ	数値	デッドタイムデータレジスタへの設定値 (0~FFFFh)
補足	-	

5.13.6 toggle

toggle		
概要	キャリア周期に同期したトグル出力を設定します。	
ヘッダ	r_mtu3_if.h	
説明	MTIOC3A または MTIOC6A からキャリア周期に同期したトグル出力を行うかを選択します。 MTIOC3A、MTIOC6A を使用する場合は、あらかじめ該当端子の I/O ポート設定およびマルチファンクションピンコントローラ（MPC）の設定が必要です。	
パラメータ	MTU_TOGGLE_OFF	トグル出力 OFF
	MTU_TOGGLE_ON	トグル出力 ON
補足	基本的には相補 PWM 出力として本出力は使用しません。評価などでご使用頂くことが可能です。	

5.13.7 mode

mode	
概 要	相補 PWM の動作モードを設定します。
ヘッダ	r_mtu3_if.h
説 明	相補 PWM の動作モードを 1～3 より選択します。
パラメータ	MTU_CMPL_PWM_MODE_1 相補 PWM モード 1(山で転送) MTU_CMPL_PWM_MODE_2 相補 PWM モード 2(谷で転送) MTU_CMPL_PWM_MODE_3 相補 PWM モード 3(山谷で転送)
補足	-

5.13.8 p_n

p_n	
概 要	出力レベルの動作を設定します。
ヘッダ	r_mtu3_if.h
説 明	相補 PWM 出力の出力レベル(正相、逆相)を固定設定か、バッファ動作設定に選択します。 固定設定とした場合は、タイマアウトプットコントロールレジスタ 1 (TOCR1A、TOCR1B) の設定で出力レベルが決まります。 バッファ動作設定とした場合は、バッファ動作によりタイマアウトプットコントロールレジスタ 2 (TOCR2A、TOCR2B) とバッファレジスタ (TOLBRA, TOLBRB) の設定で出力レベルが決まります。
パラメータ	MTU_PIN_P_N_1 ^{注1} 正相の出力レベル、逆相の出力レベルを固定にする。 MTU_PIN_P_N_2 正相の出力レベル、逆相の出力レベルをバッファ動作に設定した動作にする。
補足	注 1 : 固定設定の場合は、指定した正相および逆相の出力レベルが、PWM 出力 1～3 で共通の設定になります。

5.13.9 p_n_bf

p_n_bf	
概 要	出力レベルのバッファ操作を設定します。
ヘッダ	r_mtu3_if.h
説 明	相補 PWM 出力のバッファからの出力レベル (正相、逆相) の設定の転送トリガを選択します。 ^{注1}
パラメータ	MTU_PIN_P_N_BF_OFF 転送しない。 MTU_PIN_P_N_BF_CREST 山で転送する。 MTU_PIN_P_N_BF_TROUGH 谷で転送する。 MTU_PIN_P_N_BF_CREST_TROUGH 山谷で転送する。
補足	注 1 : p_n で MTU_PIN_P_N_2 を選択した時のみ有効になります。

5.13.10 d_bf

d_bf	
概 要	ダブルバッファ機能の使用有無を設定します。
ヘッダ	r_mtu3_if.h
説 明	相補 PWM 出力のダブルバッファ機能を有効／無効に設定します。 ^{注1} 設定を有効にすることで PWM 変更時の PWM 出力の最小分解能を±2 から±1 にすることが可能です。
パラメータ	MTU_CMPL_PWM_D_BF_OFF ダブルバッファ機能を無効に指定します。 MTU_CMPL_PWM_D_BF_ON ダブルバッファ機能を有効に指定します。
補足	注 1 : ダブルバッファ機能は、相補 PWM モード 3 (山谷で転送) 時のみ指定可能です。

5.13.15 pwm_output_X.output (X= 1、2、3)

pwm_output_X.duty

概 要 PWM 出力 X (X= 1、2、3) の端子出力の許可／禁止を設定します。

ヘッダ r_mtu3_if.h

説 明 相補 PWM 出力 X (X= 1、2、3) の端子出力を正相、逆相の組み合わせごとに許可／禁止に設定します。

MTU3、4 選択時

PWM 出力 1 : MTIOC3B (正相) MTIOC3D (逆相)

PWM 出力 2 : MTIOC4A (正相) MTIOC4C (逆相)

PWM 出力 3 : MTIOC4B (正相) MTIOC4D (逆相)

MTU6、7 選択時

PWM 出力 1 : MTIOC6B (正相) MTIOC6D (逆相)

PWM 出力 2 : MTIOC7A (正相) MTIOC7C (逆相)

PWM 出力 3 : MTIOC7B (正相) MTIOC7D (逆相)

PWM 出力端子を使用する場合は、あらかじめ該当端子の I/O ポート設定およびマルチファンクションピンコントローラ (MPC) の設定が必要です。

パラメータ MTU_CMPL_PWM_OUTPUT_OFF 相補 PWM 出力として使用しない

MTU_CMPL_PWM_OUTPUT_ON 相補 PWM 出力として使用する

補足 -

5.14 R_MTU_PWM_Compliment_Control パラメータ一覧

R_MTU_PWM_Compliment_Control 関数で使用するパラメータ (*pcmd_data) 一覧を以下に示します。

pcmd_data は、cmd で指定したコマンドに合わせた形式で記述する必要があります。

5.14.1 cmd = MTU_CMD_START 時

本コマンド指定時は、R_MTU_PWM_Compliment_Control 関数の第一引数 channel によるチャンネル指定のみが有効です。パラメータ (*pcmd_data) は設定せずに NULL に設定してください。

5.14.2 cmd = MTU_CMD_STOP 時

本コマンド指定時は、R_MTU_PWM_Compliment_Control 関数の第一引数 channel によるチャンネル指定のみが有効です。パラメータ (*pcmd_data) は設定せずに NULL に設定してください。

5.14.3 cmd = MTU_CMD_GET_STATUS 時

パラメータ (*pcmd_data) に mtu_cmpl_pwm_chnl_status_t 構造体の先頭アドレスを指定します。コマンド実行時に以下のパラメータ情報が取得されて指定した構造体変数に値を返します。

表 5.8 cmd = MTU_CMD_GET_STATUS 時パラメータ一覧

パラメータ	概要
c_st	タイマのカウンタ状態（動作、停止）
d_st	タイマのカウンタ方向

5.15 R_MTU_Timer_Open パラメータ一覧

R_MTU_Timer_Open 関数で使用するパラメータ (*pconfig) 一覧を以下に示します。

表 5.9 R_MTU_Timer_Open パラメータ一覧

パラメータ	概要
clock_src.source	カウントクロックを設定します。
clock_src.clock_edge	クロックエッジを設定します。
clear_src	カウンタクリア要因を設定します。
timer_X.actions.freq (X= a、b、c、d)	タイマジェネラルレジスタ TGRX (X= a、b、c、d) のコンペアマッチ周期 (Hz) を設定します。
timer_X.actions.do_action (X= a、b、c、d)	タイマジェネラルレジスタ TGRX (X= a、b、c、d) の動作を設定します。
timer_X.actions.output (X= a、b、c、d)	タイマジェネラルレジスタ TGRX (X= a、b、c、d) の端子出力レベルを設定します。

5.15.1 clock_src.source

clock_src.source	
概 要	カウントクロックを設定します。
ヘッダ	r_mtu3_if.h
説 明	MTU の各チャンネルにおけるカウントクロックを設定します。 外部クロックを選択する場合は、あらかじめ該当端子の I/O ポート設定およびマルチファンクションピンコントローラ (MPC) の設定が必要です。 内部クロック (PCLKC) を選択する場合は、指定したタイマ周期 (timer_X.actions.freq) より適切な分周比を自動で設定します。
パラメータ	<div> <div>MTU_CLK_SRC_EXT_MTCLKA</div> <div>外部クロック (MTCLKA 端子入力) を指定します。</div> </div> <div> <div>MTU_CLK_SRC_EXT_MTCLKB</div> <div>外部クロック (MTCLKB 端子入力) を指定します。</div> </div> <div> <div>MTU_CLK_SRC_EXT_MTCLKC ^{注1}</div> <div>外部クロック (MTCLKC 端子入力) を指定します。</div> </div> <div> <div>MTU_CLK_SRC_EXT_MTCLKD ^{注2}</div> <div>外部クロック (MTCLKD 端子入力) を指定します。</div> </div> <div> <div>MTU_CLK_SRC_CASCADE ^{注3}</div> <div>MTU2.TCNT のオーバフロー／アンダフローを指定します。</div> </div> <div> <div>MTU_CLK_SRC_INTERNAL</div> <div>内部クロック (PCLKC) を指定します。</div> </div>
補足	注 1：チャンネル 0、2 のみ設定可能です。 注 2：チャンネル 0 のみ設定可能です。 注 3：チャンネル 1 のみ設定可能です。

5.15.2 clock_src.clock_edge

clock_src.clock_edge	
概 要	クロックエッジを設定します。
ヘッダ	r_mtu3_if.h
説 明	MTU の各チャンネルにおける入力クロックをカウントするエッジを設定します。
パラメータ	<div> <div>MTU_CLK_RISING_EDGE</div> <div>立ち上がりエッジでカウントに指定します。</div> </div> <div> <div>MTU_CLK_FALLING_EDGE</div> <div>立ち下がりエッジでカウントに指定します。</div> </div> <div> <div>MTU_CLK_ANY_EDGE</div> <div>両エッジでカウントに指定します。</div> </div>
補足	指定したタイマ周期 (timer_X.actions.freq) によりカウントクロックが PCLKC/1 となる場合、またはカウントクロックに MTU2.TCNT のオーバフロー／アンダフローを指定した場合は、エッジの指定は無視され初期値 (立ち上がりエッジ) となります。

5.15.3 clear_src

clear_src	
概 要	カウンタクリア要因を設定します。
ヘッダ	r_mtu3_if.h
説 明	MTU の各チャンネルにおける TCNT のカウンタクリア要因を設定します。
パラメータ	MTU_CLR_TIMER_A TGRA のコンペアマッチを指定します。
	MTU_CLR_TIMER_B TGRB のコンペアマッチを指定します。
	MTU_CLR_TIMER_C ^{注1} TGRC のコンペアマッチを指定します。
	MTU_CLR_TIMER_D ^{注1} TGRD のコンペアマッチを指定します。
	MTU_CLR_SYNC 同期クリア／同期動作をしている他のチャンネルのカウンタクリアを指定します。
	MTU_CLR_DISABLED TCNT のクリア禁止を指定します。
補足	注 1：チャンネル 0、3、4、6、7、8 のみ設定可能です。

5.15.4 timer_X.actions.freq (X = a, b, c, d)

timer_X.actions.freq	
概 要	タイマジェネラルレジスタ TGRX (X = a, b, c, d) のコンペアマッチ周期 (Hz) を設定します。
ヘッダ	r_mtu3_if.h
説 明	MTU の各チャンネルにおける TGR の周期 (Hz) を指定します。 カウントクロックに内部クロック (PCLKC) を指定している場合は、適切な PCLKC の分周比や TGR 設定値 (カウント値) を自動で計算します。 カウントクロックに外部クロックを指定している場合は、本パラメータの値は単に TGR に設定されコンペアマッチまでのカウント値として扱われます。
パラメータ	数値 (Hz 単位) Hz 単位で周期を設定します。
	外部クロックの場合は TGR カウント値 (最大 FFFFh) を指定します。
補足	設定可能範囲は 3~100000000Hz です (チャンネル 8 のみ、1~100000000Hz です) また、clock_src.clock_edge で両エッジを設定した場合は上限が 60000000Hz となります。

5.15.5 timer_X.actions.do_action (X = a, b, c, d)

timer_X.actions.do_action

概要	タイマジェネラルレジスタ TGRX (X = a, b, c, d) の動作を設定します。	
ヘッダ	r_mtu3_if.h	
説明	MTU の各チャンネルにおける TGR の動作を指定します。 MTU 端子を出力に指定する場合は、あらかじめ該当端子の I/O ポート設定およびマルチファンクションピンコントローラ (MPC) の設定が必要です。 またパラメータは以下のように複数の動作を同時に指定することが可能です。 例) *pconfig として my_timer_cfg を使用する場合 my_timer_cfg.timer_a.actions.do_action = (mtu_actions_t)(MTU_ACTION_OUTPUT MTU_ACTION_INTERRUPT);	
パラメータ	MTU_ACTION_OUTPUT	MTU 端子を出力に指定します。
	MTU_ACTION_INTERRUPT 注1	コンペアマッチ割り込みを許可に指定します。
	MTU_ACTION_CALLBACK	コンペアマッチ割り込みを許可にして、割り込みサービスルーチンにてユーザ指定のコールバック関数を実行するよう指定します。
	MTU_ACTION_TRIGGER_ADC 注2	TGRA のコンペアマッチによる A/D コンバータのトリガ起動を指定します。
	MTU_ACTION_REPEAT 注3	最初のコンペアマッチ後もタイマカウント動作を継続するよう指定します。
	MTU_ACTION_NONE	TGR を使用しない場合に指定します。他のパラメータと組み合わせて指定することは禁止です。
補足	注 1 : 割り込み優先順位はあらかじめ r_mtu3_config.h で指定する必要があります。 注 2 : チャンネル 0~4、6、7 を指定可能です。 注 3 : 本パラメータを指定しない場合は、割り込み処理ルーチン内でタイマカウントを停止します。このため割り込み許可 (MTU_ACTION_INTERRUPT または MTU_ACTION_CALLBACK) を指定しない場合は、常にタイマカウントは継続動作します。	

5.15.6 timer_X.actions.output (X = a, b, c, d)

timer_X.actions.output

概要	タイマジェネラルレジスタ TGRX (X = a, b, c, d) の端子出力レベルを設定します。	
ヘッダ	r_mtu3_if.h	
説明	MTU の各チャンネルにおける TGR の端子出力レベルを設定します。	
パラメータ	MTU_PIN_NO_OUTPUT	出力禁止を指定します。
	MTU_PIN_LO_GOLO	初期出力は Low、コンペアマッチで Low に指定します。
	MTU_PIN_LO_GOHI	初期出力は Low、コンペアマッチで High に指定します。
	MTU_PIN_LO_TOGGLE	初期出力は Low、コンペアマッチでトグル出力に指定します。
	MTU_PIN_HI_GOLO	初期出力は High、コンペアマッチで Low に指定します。
	MTU_PIN_HI_GOHI	初期出力は High、コンペアマッチで High に指定します。
補足	MTU_PIN_HI_TOGGLE	初期出力は High、コンペアマッチでトグル出力に指定します。
	パラメータで _GOLO、_GOHI を指定した場合は、最初のコンペアマッチで MTU 端子出力を Low、High に設定して以後は出力レベルを保持します。以降はコンペアマッチが発生しても出力レベルは変化しません。 また _TOGGLE を指定した場合は、コンペアマッチが発生する度に MTU 端子の出力レベルを切り替えてトグル出力を行います。	

5.16 R_MTU_Capture_Open パラメータ一覧

R_MTU_Capture_Open 関数で使用するパラメータ (*pconfig) 一覧を以下に示します。

表 5.10 R_MTU_Capture_Open パラメータ一覧

パラメータ	概要
clock_src.source	カウントクロックを設定します。
clock_src.clock_edge	クロックエッジを設定します。
clock_div	内部クロックの分周比を設定します。
clear_src	カウンタクリア要因を設定します。
capture_X.actions (X= a、b、c、d)	タイマジェネラルレジスタ TGRX (X= a、b、c、d) の動作を設定します。
capture_X.capture_edge (X= a、b、c、d)	タイマジェネラルレジスタ TGRX (X= a、b、c、d) のインプットキャプチャ入力信号の有効エッジを設定します。
capture_X.filter_enable (X= a、b、c、d)	対応する MTU 入力端子 MTIOCnX (X= a、b、c、d) のノイズフィルタを設定します (n=0~4、6、7、8)。

5.16.1 clock_src.source

clock_src.source	
概 要	カウントクロックを設定します。
ヘッダ	r_mtu3_if.h
説 明	MTU の各チャンネルにおけるカウントクロックを設定します。 外部クロックを選択する場合は、あらかじめ該当端子の I/O ポート設定およびマルチファンクションピンコントローラ (MPC) の設定が必要です。
パラメータ	MTU_CLK_SRC_EXT_MTCLKA 外部クロック (MTCLKA 端子入力) を指定します。 MTU_CLK_SRC_EXT_MTCLKB 外部クロック (MTCLKB 端子入力) を指定します。 MTU_CLK_SRC_EXT_MTCLKC ^{注1} 外部クロック (MTCLKC 端子入力) を指定します。 MTU_CLK_SRC_EXT_MTCLKD ^{注2} 外部クロック (MTCLKD 端子入力) を指定します。 MTU_CLK_SRC_CASCADE ^{注3} MTU2.TCNT のオーバフロー／アンダフローを指定します。 MTU_CLK_SRC_INTERNAL 内部クロック (PCLKC) を指定します。
補足	注 1：チャンネル 0、2 のみ指定可能です。 注 2：チャンネル 0 のみ指定可能です。 注 3：チャンネル 1 のみ指定可能です。

5.16.2 clock_src.clock_edge

clock_src.clock_edge	
概 要	クロックエッジを選択します。
ヘッダ	r_mtu3_if.h
説 明	MTU の各チャンネルにおける入力クロックをカウントするエッジを設定します。
パラメータ	MTU_CLK_RISING_EDGE 立ち上がりエッジでカウントに指定します。 MTU_CLK_FALLING_EDGE 立ち下がりエッジでカウントに指定します。 MTU_CLK_ANY_EDGE 両エッジでカウントに指定します。
補足	指定した内部クロックの分周比 (clock_div) が PCLKC/1 となる場合、またはカウントクロックに MTU2.TCNT のオーバフロー／アンダフローを指定した場合は、エッジの指定は無視され初期値 (立ち上がりエッジ) となります。

5.16.3 clock_div

clock_div		
概 要	内部クロックの分周比を設定します。	
ヘッダ	r_mtu3_if.h	
説 明	MTU の各チャンネルにおける内部クロックの分周比を設定します。 clock_src.clock_edge で外部クロックを選択した場合は本パラメータの設定は無効となります。	
パラメータ	MTU_SRC_CLK_DIV_1	内部クロック (PCLKC/1) を指定します。
	MTU_SRC_CLK_DIV_2	内部クロック (PCLKC/2) を指定します。
	MTU_SRC_CLK_DIV_4	内部クロック (PCLKC/4) を指定します。
	MTU_SRC_CLK_DIV_8	内部クロック (PCLKC/8) を指定します。
	MTU_SRC_CLK_DIV_16	内部クロック (PCLKC/16) を指定します。
	MTU_SRC_CLK_DIV_32	内部クロック (PCLKC/32) を指定します。
	MTU_SRC_CLK_DIV_64	内部クロック (PCLKC/64) を指定します。
	MTU_SRC_CLK_DIV_256	内部クロック (PCLKC/256) を指定します。
	MTU_SRC_CLK_DIV_1024	内部クロック (PCLKC/1024) を指定します。
補足	-	

5.16.4 clear_src

clear_src		
概 要	カウンタクリア要因を選択します。	
ヘッダ	r_mtu3_if.h	
説 明	MTU の各チャンネルにおける TCNT のカウンタクリア要因を設定します。	
パラメータ	MTU_CLR_TIMER_A	TGRA のインプットキャプチャを指定します。
	MTU_CLR_TIMER_B	TGRB のインプットキャプチャを指定します。
	MTU_CLR_TIMER_C ^{注1}	TGRC のインプットキャプチャを指定します。
	MTU_CLR_TIMER_D ^{注1}	TGRD のインプットキャプチャを指定します。
	MTU_CLR_SYNC	同期クリア／同期動作をしている他のチャンネルのカウンタクリアを指定します。
	MTU_CLR_DISABLED	TCNT のクリア禁止を指定します。
補足	注 1 : チャンネル 0、3、4、6、7、8 のみ指定可能です。	

5.16.5 capture_X.actions (X = a, b, c, d)

capture_X.actions

概 要	タイマジェネラルレジスタ TGRX (X = a, b, c, d) の動作を設定します。
ヘッダ	r_mtu3_if.h
説 明	MTU の各チャンネルにおける TGR の動作を指定します。 MTIOCNm 端子 (n=0~4, 6, 7, 8, m=A, B, C, D) をインプットキャプチャ入力に使用する場合 は、あらかじめ該当端子の I/O ポート設定およびマルチファンクションピンコントローラ (MPC) の設定が必要です。 またパラメータは以下のように複数の動作を同時に指定することが可能です。

例) *pconfig として my_capture_cfg を使用する場合

```
my_capture_cfg.capture_a.actions
= (mtu_actions_t)(MTU_ACTION_CAPTURE | MTU_ACTION_REPEAT);
```

パラメータ	MTU_ACTION_CAPTURE 注1	TGR をインプットキャプチャ動作に設定します。
	MTU_ACTION_INTERRUPT 注2	インプットキャプチャ割り込みを許可に指定します。
	MTU_ACTION_CALLBACK	インプットキャプチャ割り込みを許可にして、割り込みサービスルーチンにて、ユーザ指定のコールバック関数を実行するよう指定します。
	MTU_ACTION_TRIGGER_ADC 注3	TGRA のインプットキャプチャによる A/D コンバータのトリガ起動を指定します。
	MTU_ACTION_REPEAT 注4	最初のインプットキャプチャ後もタイマカウント動作を継続するよう指定します。
	MTU_ACTION_NONE	TGR を使用しない場合に指定します。他のパラメータと組み合わせて指定することは禁止です。

補足	注 1 : インプットキャプチャ機能を使用する場合は、必ず MTU_ACTION_CAPTURE は指定する必要があります。	
	注 2 : 割り込み優先順位はあらかじめ r_mtu3_config.h で指定する必要があります。	
	注 3 : チャンネル 0~4、6、7 を指定可能です。	
	注 4 : 本パラメータを指定しない場合は、割り込み処理ルーチン内でタイマカウントを停止します。このため割り込み許可 (MTU_ACTION_INTERRUPT または MTU_ACTION_CALLBACK) を指定しない場合は、常にタイマカウントは継続動作します。	

5.16.6 capture_X.capture_edge (X = a, b, c, d)

capture_X.capture_edge

概 要	タイマジェネラルレジスタ TGRX (X = a, b, c, d) のインプットキャプチャ入力信号の有効エッジを設定します。		
ヘッダ	r_mtu3_if.h		
説 明	MTU の各チャンネルにおけるインプットキャプチャ信号に対する有効エッジの設定をします。		
パラメータ	MTU_CAP_RISING_EDGE	立ち上がりエッジでカウントに指定します。	
	MTU_CAP_FALLING_EDGE	立ち下がりエッジでカウントに指定します。	
	MTU_CAP_ANY_EDGE	両エッジでカウントに指定します。	
補足	-		

5.16.7 capture_X.filter_enable (X = a, b, c, d)

capture_X.filter_enable

概 要	対応する MTU 入力端子 MTIOCnX (X=A, B, C, D) のノイズフィルタを設定します (n=0~4, 6, 7, 8)。	
ヘッダ	r_mtu3_if.h	
説 明	MTU の各チャンネルにおけるインプットキャプチャ信号に対するデジタルノイズフィルタを設定します。	
パラメータ	true	ノイズフィルタを有効に設定します。
	false	ノイズフィルタを無効に設定します。
補足	ノイズフィルタのサンプリングクロックは、あらかじめ r_mtu3_config.h で指定する必要があります。	

5.17 R_MTU_PWM_Open パラメーター一覧

R_MTU_PWM_Open 関数で使用するパラメータ (*pconfig) 一覧を以下に示します。

表 5.11 R_MTU_PWM_Open パラメーター一覧

パラメータ	概要
clock_src.source	カウントクロックを設定します。
clock_src.clock_edge	クロックエッジを設定します。
cycle_freq	PWM 周期 (Hz) を設定します。
clear_src	カウンタクリア要因を設定します。
pwm_mode	PWM モードを設定します。
pwm_X.duty (X = a、b、c、d)	タイマジェネラルレジスタ TGRX (X = a、b、c、d) に対するデューティ比を設定します。
pwm_X.actions (X = a、b、c、d)	タイマジェネラルレジスタ TGRX (X = a、b、c、d) の動作を設定します。
pwm_X.outputs (X = a、b、c、d)	タイマジェネラルレジスタ TGRX (X = a、b、c、d) に対する PWM 出力の端子レベルを設定します。

5.17.1 clock_src.source

clock_src.source											
概 要	カウントクロックを設定します。										
ヘッダ	r_mtu3_if.h										
説 明	MTU の各チャンネルにおけるカウントクロックを設定します。 外部クロックを選択する場合は、あらかじめ該当端子の I/O ポート設定およびマルチファンクションピンコントローラ (MPC) の設定が必要です。 内部クロック (PCLKC) を選択する場合は、指定した PWM 周期 (cycle_freq) より適切な分周比を自動で設定します。										
パラメータ	<table> <tr> <td>MTU_CLK_SRC_EXT_MTCLKA</td><td>外部クロック (MTCLKA 端子入力) を指定します。</td></tr> <tr> <td>MTU_CLK_SRC_EXT_MTCLKB</td><td>外部クロック (MTCLKB 端子入力) を指定します。</td></tr> <tr> <td>MTU_CLK_SRC_EXT_MTCLKC ^{注1}</td><td>外部クロック (MTCLKC 端子入力) を指定します。</td></tr> <tr> <td>MTU_CLK_SRC_EXT_MTCLKD ^{注2}</td><td>外部クロック (MTCLKD 端子入力) を指定します。</td></tr> <tr> <td>MTU_CLK_SRC_INTERNAL</td><td>内部クロック (PCLKC) を指定します。</td></tr> </table>	MTU_CLK_SRC_EXT_MTCLKA	外部クロック (MTCLKA 端子入力) を指定します。	MTU_CLK_SRC_EXT_MTCLKB	外部クロック (MTCLKB 端子入力) を指定します。	MTU_CLK_SRC_EXT_MTCLKC ^{注1}	外部クロック (MTCLKC 端子入力) を指定します。	MTU_CLK_SRC_EXT_MTCLKD ^{注2}	外部クロック (MTCLKD 端子入力) を指定します。	MTU_CLK_SRC_INTERNAL	内部クロック (PCLKC) を指定します。
MTU_CLK_SRC_EXT_MTCLKA	外部クロック (MTCLKA 端子入力) を指定します。										
MTU_CLK_SRC_EXT_MTCLKB	外部クロック (MTCLKB 端子入力) を指定します。										
MTU_CLK_SRC_EXT_MTCLKC ^{注1}	外部クロック (MTCLKC 端子入力) を指定します。										
MTU_CLK_SRC_EXT_MTCLKD ^{注2}	外部クロック (MTCLKD 端子入力) を指定します。										
MTU_CLK_SRC_INTERNAL	内部クロック (PCLKC) を指定します。										
補足	注 1 : チャンネル 0、2 のみ指定可能です。 注 2 : チャンネル 0 のみ指定可能です。										

5.17.2 clock_src.clock_edge

clock_src.clock_edge							
概 要	クロックエッジを設定します。						
ヘッダ	r_mtu3_if.h						
説 明	MTU の各チャンネルにおける入力クロックをカウントするエッジを設定します。						
パラメータ	<table> <tr> <td>MTU_CLK_RISING_EDGE</td><td>立ち上がりエッジでカウントに指定します。</td></tr> <tr> <td>MTU_CLK_FALLING_EDGE</td><td>立ち下がりエッジでカウントに指定します。</td></tr> <tr> <td>MTU_CLK_ANY_EDGE</td><td>両エッジでカウントに指定します。</td></tr> </table>	MTU_CLK_RISING_EDGE	立ち上がりエッジでカウントに指定します。	MTU_CLK_FALLING_EDGE	立ち下がりエッジでカウントに指定します。	MTU_CLK_ANY_EDGE	両エッジでカウントに指定します。
MTU_CLK_RISING_EDGE	立ち上がりエッジでカウントに指定します。						
MTU_CLK_FALLING_EDGE	立ち下がりエッジでカウントに指定します。						
MTU_CLK_ANY_EDGE	両エッジでカウントに指定します。						
補足	指定した PWM 周期 (cycle_freq) によりカウントクロックが PCLKC/1 となる場合は、エッジの指定は無視され初期値 (立ち上がりエッジ) となります。						

5.17.3 cycle_freq

cycle_freq	
概要	PWM 周期 (Hz) を設定します。
ヘッダ	r_mtu3_if.h
説明	MTU の各チャンネルにおける PWM 周期 (Hz) を指定します。 カウントクロックに内部クロック (PCLKC) を指定している場合は、適切な PCLKC の分周比を自動で計算します。 カウントクロックに外部クロックを指定している場合は、本パラメータの設定値は単に TGR に設定され PWM 周期のカウント値として扱われます。
パラメータ	数値 (Hz 単位) Hz 単位で周期を指定します。 外部クロックの場合は TGR カウント値 (最大 FFFFh) を指定します。
補足	PWM モード 1 の場合は、TGRA と TGRB が周期設定レジスタになります。 PWM モード 2 の場合は、clear_src で指定した TGR が周期設定レジスタになります。

5.17.4 clear_src

clear_src	
概要	カウンタクリア要因を設定します。
ヘッダ	r_mtu3_if.h
説明	MTU の各チャンネルにおける TCNT のカウンタクリア要因を設定します。
パラメータ	MTU_CLR_TIMER_A TGRA のコンペアマッチを指定します。 MTU_CLR_TIMER_B TGRB のコンペアマッチを指定します。 MTU_CLR_TIMER_C ^{注1} TGRB のコンペアマッチを指定します。 MTU_CLR_TIMER_D ^{注1} TGRD のコンペアマッチを指定します。 MTU_CLR_SYNC 同期クリア/同期動作をしている他のチャンネルのカウンタクリアを指定します。 MTU_CLR_DISABLED TCNT のクリア禁止を指定します。
補足	注 1 : チャンネル 0、3、4、6、7、8 のみ指定可能です。

5.17.5 pwm_mode

pwm_mode	
概要	PWM モードを設定します。
ヘッダ	r_mtu3_if.h
説明	MTU の各チャンネルにおける PWM モードを設定します。
パラメータ	MTU_PWM_MODE_1 ^{注1} PWM モード 1 を設定します。 MTU_PWM_MODE_2 ^{注2} PWM モード 2 を設定します。
補足	注 1 : チャンネル 0~4、6、7 のみ指定可能です。 注 2 : チャンネル 0~2 のみ指定可能です。

5.17.6 pwm_X.duty (X = a, b, c, d)

pwm_X.duty	
概要	タイマジェネラルレジスタ TGRX (X = a, b, c, d) に対するデューティ比を設定します。
ヘッダ	r_mtu3_if.h
説明	MTU の各チャンネルにおける PWM 波形のデューティ比 (0.1%単位) を設定します。 デューティ比には 0%~100%までを設定することが可能です。
パラメータ	数値 (0.1%単位) 0.1%単位でデューティ比を指定します。 (0~1000)
補足	PWM モード 1 の場合は、TGRB と TGRD がデューティ設定レジスタになります。 PWM モード 2 の場合は、clear_src で指定した TGR 以外がデューティ設定レジスタになります。

5.17.7 pwm_X.actions (X = a, b, c, d)

pwm_X.actions	
概 要	タイマジェネラルレジスタ TGRX (X = a, b, c, d) の動作を設定します。
ヘッダ	r_mtu3_if.h
説 明	MTU の各チャンネルにおける TGR の動作を指定します。 MTU 端子を出力に指定する場合は、あらかじめ該当端子の I/O ポート設定およびマルチファンクションピンコントローラ (MPC) の設定が必要です。 またパラメータは以下のように複数の動作を同時に指定することが可能です。
例) *pconfig として simple_pwm_cfg を使用する場合 simple_pwm_cfg.pwm_a.actions = (mtu_actions_t)(MTU_ACTION_OUTPUT MTU_ACTION_INTERRUPT);	
パラメータ	MTU_ACTION_OUTPUT MTU 端子を出力に指定します。 MTU_ACTION_INTERRUPT ^{注1} コンペアマッチ割り込みを許可に指定します。 MTU_ACTION_CALLBACK コンペアマッチ割り込みを許可にして、割り込みサービスルーチンにて、ユーザ指定のコールバック関数を実行するよう指定します。 MTU_ACTION_TRIGGER_ADC ^{注2} TGRA のコンペアマッチによる A/D コンバータのトリガ起動を指定します。 MTU_ACTION_REPEAT ^{注3} 最初の PWM 周期経過後も PWM 出力を継続するよう指定します。 MTU_ACTION_NONE TGR を使用しない場合に指定します。他のパラメータと組み合わせて指定することは禁止です。
補足	注 1 : 割り込み優先順位はあらかじめ r_mtu3_config.h で指定する必要があります。 注 2 : チャンネル 0~4、6、7 を指定可能です。 注 3 : 本パラメータを指定しない場合は、割り込み処理ルーチン内でタイマカウントを停止します。このため割り込み許可 (MTU_ACTION_INTERRUPT または MTU_ACTION_CALLBACK) を指定しない場合は、常にタイマカウントは継続動作します。

5.17.8 pwm_X.outputs (X = a, b, c, d)

pwm_X.outputs	
概 要	タイマジェネラルレジスタ TGRX (X = a, b, c, d) に対する PWM 出力の端子レベルを設定します。
ヘッダ	r_mtu3_if.h
説 明	MTU の各チャンネルにおける PWM 出力端子の初期出力レベルと、コンペアマッチ時の出力レベルを設定します。
パラメータ	MTU_PIN_NO_OUTPUT 出力禁止を指定します。 MTU_PIN_LO_GOLO 初期出力は Low、コンペアマッチで Low に指定します。 MTU_PIN_LO_GOHI 初期出力は Low、コンペアマッチで High に指定します。 MTU_PIN_LO_TOGGLE 初期出力は Low、コンペアマッチでトグル出力に指定します。 MTU_PIN_HI_GOLO 初期出力は High、コンペアマッチで Low に指定します。 MTU_PIN_HI_GOHI 初期出力は High、コンペアマッチで High に指定します。 MTU_PIN_HI_TOGGLE 初期出力は High、コンペアマッチでトグル出力に指定します。
補足	PWM モード 1 では、TGRA と TGRB を使用したときに MTIOCnA 端子から PWM 出力をします。また TGRC と TGRD を使用したときに MTIOCnC 端子から PWM 出力をします。 PWM モード 2 では、周期レジスタに設定した 1 つの TGR とその他のデューティ比レジスタに設定した TGR の組み合わせで PWM 出力をします。 また _TOGGLE を指定した場合は、コンペアマッチが発生する度に MTU 端子の出力レベルを切り替えてトグル出力を行います。基本的には PWM モードでは使用しません。評価などでご使用頂くことが可能です。

5.18 R_MTU_Control パラメータ一覧

R_MTU_Control 関数で使用するパラメータ (*pconfig) 一覧を以下に示します。

pcmd_data は、cmd で指定したコマンドに合わせた形式で記述する必要があります。

5.18.1 cmd = MTU_CMD_START 時

パラメータ (*pcmd_data) には、mtu_group_t 構造体の型の先頭アドレスを指定します。

pcmd_data																	
概 要	使用するチャンネル番号を複数指定します。																
ヘッダ	r_mtu3_if.h																
説 明	MTU のチャンネルを複数指定し、同時に動作させる場合に指定します。 例) pcmd_data として、my_group を使用する場合 <pre>mtu_group_t my_group; my_group = (mtu_group_t)(MTU_GRP_CH0 MTU_GRP_CH3 MTU_GRP_CH4); result = R_MTU_Control(MTU_CHANNEL_0, MTU_CMD_START, &my_group);</pre>																
パラメータ	<table> <tr><td>MTU_GRP_CH0</td><td>チャンネル番号 0</td></tr> <tr><td>MTU_GRP_CH1</td><td>チャンネル番号 1</td></tr> <tr><td>MTU_GRP_CH2</td><td>チャンネル番号 2</td></tr> <tr><td>MTU_GRP_CH3</td><td>チャンネル番号 3</td></tr> <tr><td>MTU_GRP_CH4</td><td>チャンネル番号 4</td></tr> <tr><td>MTU_GRP_CH6</td><td>チャンネル番号 6</td></tr> <tr><td>MTU_GRP_CH7</td><td>チャンネル番号 7</td></tr> <tr><td>MTU_GRP_CH8</td><td>チャンネル番号 8</td></tr> </table>	MTU_GRP_CH0	チャンネル番号 0	MTU_GRP_CH1	チャンネル番号 1	MTU_GRP_CH2	チャンネル番号 2	MTU_GRP_CH3	チャンネル番号 3	MTU_GRP_CH4	チャンネル番号 4	MTU_GRP_CH6	チャンネル番号 6	MTU_GRP_CH7	チャンネル番号 7	MTU_GRP_CH8	チャンネル番号 8
MTU_GRP_CH0	チャンネル番号 0																
MTU_GRP_CH1	チャンネル番号 1																
MTU_GRP_CH2	チャンネル番号 2																
MTU_GRP_CH3	チャンネル番号 3																
MTU_GRP_CH4	チャンネル番号 4																
MTU_GRP_CH6	チャンネル番号 6																
MTU_GRP_CH7	チャンネル番号 7																
MTU_GRP_CH8	チャンネル番号 8																
補足	注 1 : 1 チャンネルのみ使用する場合は、R_MTU_Control 関数の第一引数 channel のみ設定し、第三引数 pcmd_data は NULL としてください。第三引数 pcmd_data で複数チャンネルを指定した場合は、第一引数の値は無効になります。																

5.18.2 cmd = MTU_CMD_STOP 時

パラメータ (*pcmd_data) には、mtu_group_t 構造体の型の先頭アドレスを指定します。

pcmd_data																	
概 要	使用するチャンネル番号を複数指定します。																
ヘッダ	r_mtu3_if.h																
説 明	MTU のチャンネルを複数指定し、同時に動作停止させる場合に指定します。 例) pcmd_data として、my_group を使用する場合 <pre>mtu_group_t my_group; my_group = (mtu_group_t)(MTU_GRP_CH0 MTU_GRP_CH3 MTU_GRP_CH4); result = R_MTU_Control(MTU_CHANNEL_0, MTU_CMD_STOP, &my_group);</pre>																
パラメータ	<table> <tr><td>MTU_GRP_CH0</td><td>チャンネル番号 0</td></tr> <tr><td>MTU_GRP_CH1</td><td>チャンネル番号 1</td></tr> <tr><td>MTU_GRP_CH2</td><td>チャンネル番号 2</td></tr> <tr><td>MTU_GRP_CH3</td><td>チャンネル番号 3</td></tr> <tr><td>MTU_GRP_CH4</td><td>チャンネル番号 4</td></tr> <tr><td>MTU_GRP_CH6</td><td>チャンネル番号 6</td></tr> <tr><td>MTU_GRP_CH7</td><td>チャンネル番号 7</td></tr> <tr><td>MTU_GRP_CH8</td><td>チャンネル番号 8</td></tr> </table>	MTU_GRP_CH0	チャンネル番号 0	MTU_GRP_CH1	チャンネル番号 1	MTU_GRP_CH2	チャンネル番号 2	MTU_GRP_CH3	チャンネル番号 3	MTU_GRP_CH4	チャンネル番号 4	MTU_GRP_CH6	チャンネル番号 6	MTU_GRP_CH7	チャンネル番号 7	MTU_GRP_CH8	チャンネル番号 8
MTU_GRP_CH0	チャンネル番号 0																
MTU_GRP_CH1	チャンネル番号 1																
MTU_GRP_CH2	チャンネル番号 2																
MTU_GRP_CH3	チャンネル番号 3																
MTU_GRP_CH4	チャンネル番号 4																
MTU_GRP_CH6	チャンネル番号 6																
MTU_GRP_CH7	チャンネル番号 7																
MTU_GRP_CH8	チャンネル番号 8																
補足	1 チャンネルのみ使用する場合は、R_MTU_Control 関数の第一引数 channel のみ設定し、第三引数 pcmd_data は NULL にしてください。第三引数 pcmd_data で複数チャンネルを指定した場合は、第一引数の値は無効となります。																

5.18.3 cmd = MTU_CMD_SAFE_STOP 時

本コマンド指定時は、R_MTU_Control 関数の第一引数 channel による 1 チャンネル指定のみが有効です。
パラメータ (*pcmd_data) は設定せずに NULL に設定してください。

5.18.4 cmd = MTU_CMD_RESTART 時

本コマンド指定時は、R_MTU_Control 関数の第一引数 channel による 1 チャンネル指定のみが有効です。
パラメータ (*pcmd_data) は設定せずに NULL を指定してください。

5.18.5 cmd = MTU_CMD_SYNCHRONIZE 時

パラメータ (*pcmd_data) には、mtu_group_t 構造体の型の先頭アドレスを指定します。

pcmd_data	
概 要	使用するチャンネル番号を複数指定します。
ヘッダ	r_mtu3_if.h
説 明	MTU のチャンネルを複数指定し、同時に動作させる場合に指定します。 例) pcmd_data として、my_group を使用する場合 <pre>mtu_group_t my_group; my_group = (mtu_group_t)(MTU_GRP_CH0 MTU_GRP_CH3 MTU_GRP_CH4); result = R_MTU_Control(MTU_CHANNEL_0, MTU_CMD_STOP, &my_group);</pre>
パラメータ	MTU_GRP_CH0 チャンネル番号 0 MTU_GRP_CH1 チャンネル番号 1 MTU_GRP_CH2 チャンネル番号 2 MTU_GRP_CH3 チャンネル番号 3 MTU_GRP_CH4 チャンネル番号 4 MTU_GRP_CH6 チャンネル番号 6 MTU_GRP_CH7 チャンネル番号 7
補足	第三引数 pcmd_data で複数チャンネルを指定した場合は、第一引数の値は無効となります。

5.18.6 cmd = MTU_CMD_GET_STATUS 時 (timer mode 時)

timer_mode 時は、パラメータ (*pcmd_data) に mtu_timer_status_t 構造体の先頭アドレスを指定します。コマンド実行時に以下のパラメータ情報が取得されて指定した構造体変数に値を返します。

表 5.12 cmd = MTU_CMD_GET_STATUS 時(timer mode 時)パラメータ一覧

パラメータ	概要
timer_count	タイマのカウント値
timer_running	タイマのカウント方向

5.18.7 cmd = MTU_CMD_GET_STATUS 時 (input capture mode 時)

input capture mode 時は、パラメータ (*pcmd_data) に mtu_capture_status_t 構造体の先頭アドレスを指定します。コマンド実行時に以下のパラメータ情報が取得されて指定した構造体変数に値を返します。

表 5.13 cmd = MTU_CMD_GET_STATUS 時(input capture mode 時)パラメータ一覧

パラメータ	概要
capt_X_count (X= a, b, c, d)	TGRX(X= a, b, c, d)のインプットキャプチャ値
timer_count	タイマのカウント値

5.18.8 cmd = MTU_CMD_SET_CAPT_EDGE 時

本コマンド指定時は、パラメータ (*pcmd_data) に mtu_capture_set_edge_t 構造体の先頭アドレスを指定します。構造体に指定された以下のパラメータに基づき、インプットキャプチャの入力元とエッジ設定を再設定します。

表 5.14 cmd = MTU_CMD_SET_CAPT_EDGE 時パラメーター一覧

パラメータ	概要
capture_src	インプットキャプチャの入力元設定 MTU_CAP_SRC_A : MTIOCnA 端子 MTU_CAP_SRC_B : MTIOCnB 端子 MTU_CAP_SRC_C : MTIOCnC 端子 MTU_CAP_SRC_D : MTIOCnD 端子
capture_edge	インプットキャプチャのエッジ設定 MTU_CAP_RISING_EDGE : 立ち上がりエッジ MTU_CAP_FALLING_EDGE : 立ち下がりエッジ MTU_CAP_ANY_EDGE : 両エッジ

6. サンプルプログラム

サンプルプログラムは、ルネサス エレクトロニクスホームページから入手してください。

ホームページとサポート窓口

ルネサス エレクトロニクスホームページ

<http://japan.renesas.com/>

お問合せ先

<http://japan.renesas.com/contact/>

すべての商標および登録商標は、それぞれの所有者に帰属します。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2016.12.27	—	新規発行

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 未使用端子の処理

【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。

CMOS製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI周辺のノイズが印加され、LSI内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。

2. 電源投入時の処置

【注意】電源投入時は、製品の状態は不定です。

電源投入時には、LSIの内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。

外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。

同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。

3. リザーブアドレス（予約領域）のアクセス禁止

【注意】リザーブアドレス（予約領域）のアクセスを禁止します。

アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。

4. クロックについて

【注意】リセット時は、クロックが安定した後、リセットを解除してください。

プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。

リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子

（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。

5. 製品間の相違について

【注意】型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。

同じグループのマイコンでも型名が違っていると、内部ROM、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して、お客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
2. 本資料に記載されている情報は、正確を期すため慎重に作成したのですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
3. 本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害に関し、当社は、何らの責任を負うものではありません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を改造、改変、複製等しないでください。かかる改造、改変、複製等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。
標準水準： コンピュータ、OA機器、通信機器、計測機器、AV機器、家電、工作機械、パーソナル機器、産業用ロボット等
高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置等
当社製品は、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（原子力制御システム、軍事機器等）に使用されることを意図しておらず、使用することはできません。たとえ、意図しない用途に当社製品を使用したことによりお客様または第三者に損害が生じても、当社は一切その責任を負いません。なお、ご不明点がある場合は、当社営業にお問い合わせください。
6. 当社製品をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関し、当社は、一切その責任を負いません。
9. 本資料に記載されている当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。また、当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途に使用しないでください。当社製品または技術を輸出する場合は、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。
10. お客様の転売等により、本ご注意書き記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は何らの責任も負わず、お客様にてご負担して頂きますのでご了承ください。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。

- 注1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社とその総株主の議決権の過半数を直接または間接に保有する会社をいいます。
- 注2. 本資料において使用されている「当社製品」とは、注1において定義された当社の開発、製造製品をいいます。



ルネサス エレクトロニクス株式会社

■営業お問合せ窓口

<http://www.renesas.com>

※営業お問合せ窓口の住所は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス株式会社 〒135-0061 東京都江東区豊洲3-2-24（豊洲フォレシア）

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口： <http://japan.renesas.com/contact/>