RZ/T1 グループ

マルチファンクションタイマパルスユニット（MTU3a）

2021.05.14
Rev.1.50
R01AN2653JJ0150

要旨

本アプリケーションノートでは、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。

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（= 100µs）、デッドタイム（= 2µs）付きの相補 PWM 波形を出力します。
- SW2 を押下するたびに PWM のデューティ比を 25%、50%、75%（以降繰り返し）と切換えます。

動作確認デバイス

RZ/T1 グループ

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。

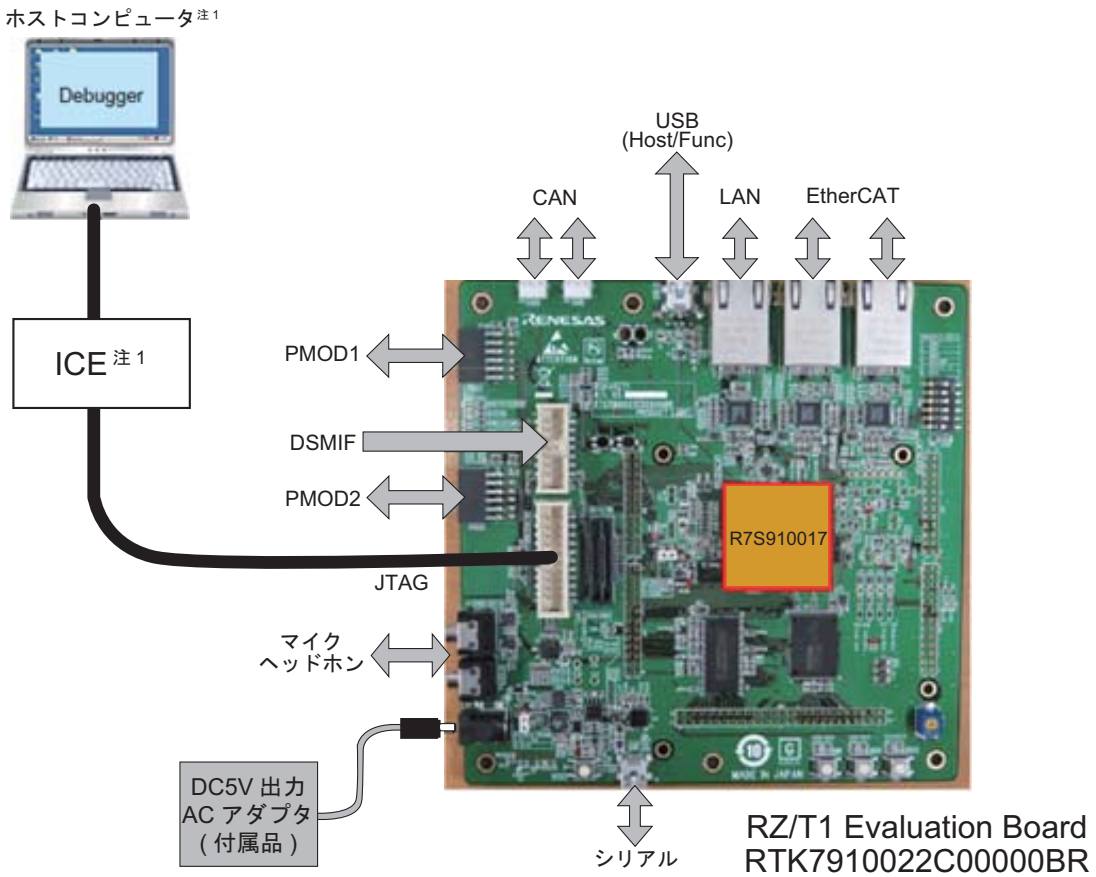
目　次

1.　仕様

表 1.1 に使用する周辺機能と用途を、**図** 1.1 に動作環境を示します。

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）	外部割り込み入力端子（IRQ5）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



注 1.　別途お客様でご用意いただく必要があります。

図 1.1　　動作環境

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 8.20.2 Arm製 DS-5™ 5.26.2 RENESAS製 e2studio 6.1.0
動作モード	SPIブートモード 16ビットバスブートモード
使用ボード	RZ/T1 Evaluation Board （RTK7910022C00000BR）
使用デバイス （ボード上で使用する機能）	• NORフラッシュメモリ （CS0、CS1空間に接続） メーカ名 : Macronix International Co..、型名 : MX29GL512FLT2I-10Q • SDRAM （CS2、CS3空間に接続） メーカ名 : Integrated Silicon Solution Inc、型名 : IS42S16320D-7TL • シリアルフラッシュメモリ メーカ名 : Macronix International Co..、型名 : MX25L51245G

3.　関連アプリケーションノート

本アプリケーションノートに関連するアプリケーションノートを以下に示します。併せて参照してください。

- RZ/T1 グループ 初期設定

4.　周辺機能説明

クロック発生回路（CPG）、マルチファンクションタイマパルスユニット（MTU3a）、割り込みコントローラ（ICUA）、エラーコントロールモジュール（ECM）についての基本的な内容は、RZ/T1 グループ・ユーザーズマニュアル ハードウェア編を参照してください。

5.　ハードウェア説明

5.1　ハードウェア構成例

図 5.1 にハードウェア構成例を示します。

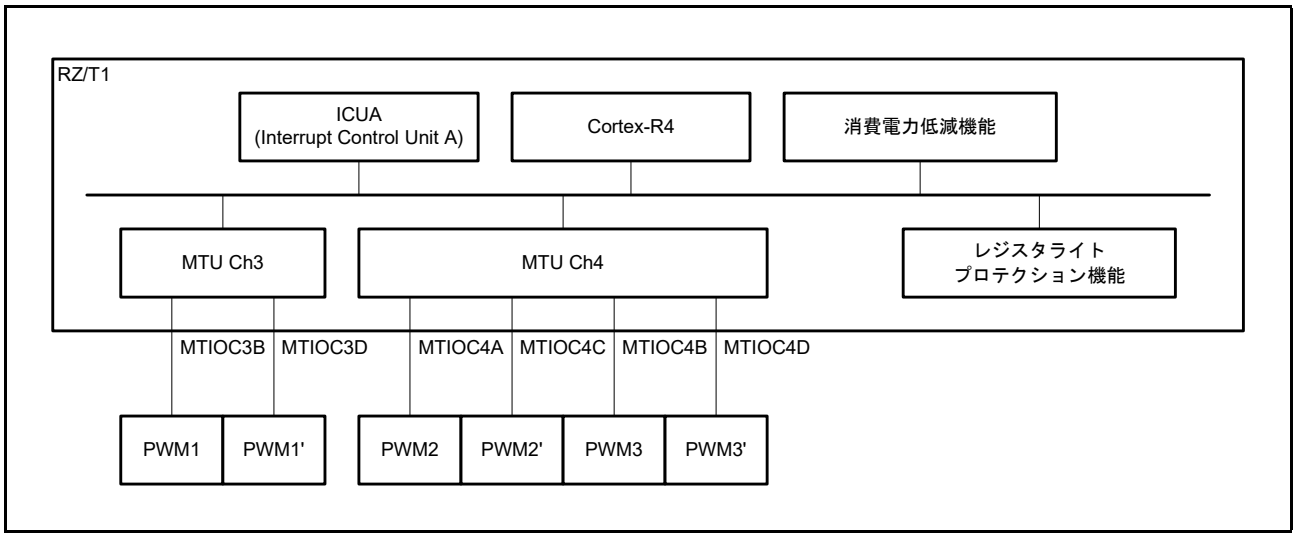


図 5.1　ハードウェア構成例

5.2　使用端子一覧

表 5.1 に使用端子と機能を示します。

表 5.1　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” （16 ビットバスブートモード）
MD1	入力	
MD2	入力	
IRQ5	入力	SW2（IRQ 端子割り込み）
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）

6.　ソフトウェア説明

6.1　動作概要

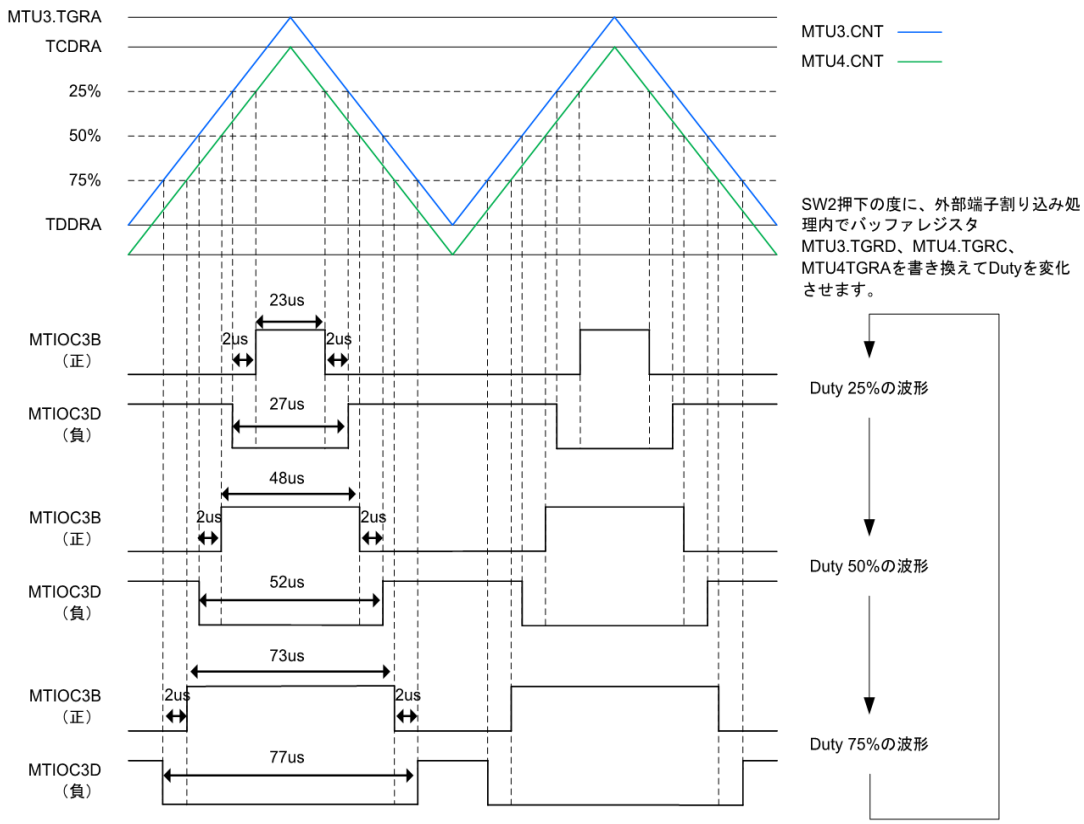
本サンプルプログラムでは、マルチファンクションタイマパルスユニット（MTU3a）の初期設定を行い、相補 PWM 出力を行います。

SW2 が押されると外部端子割り込み 5 が発生し、バッファレジスタ書き換えによりデューティ比（25% → 50% → 75% → 25% →（以降繰り返し））を変更します。

本サンプルプログラムの機能概要を**表 6.1　動作概要**に示します。また、**図 6.1** にタイミング図を示します。

表6.1　　動作概要

機能	概要
チャネル	チャネル3（MTU3）、チャネル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	各相共通で25%、50%、75%



注 .　以下の MTU 出力端子も同じ出力です。
MTIOC4A（正）と MTIOC4C（逆）
MTIOC4B（正）と MTIOC4D（逆）

図 6.1　　タイミング図

6.1.1　プロジェクト設定

開発環境となる EWARM 上で使用されるプロジェクト設定については、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。

6.2　メモリマップ

RZ/T1 グループのアドレス空間と RZ/T1 評価ボードのメモリマッピングについては、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。

6.2.1　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）、スキャ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。

6.2.2　MPU の設定

MPU の設定は、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。

6.2.3　例外処理ベクタテーブル

例外処理のベクタテーブルは、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。

6.3　使用割り込み一覧

表 6.2 にサンプルプログラムで使用する割り込みを示します。

表6.2　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
IRQ9割り込み（IRQ端子割り込み5）	15	SW2が押下される度に、バッファレジスタMTU3.TGRD、MTU4.TGRC、MTU4.TGRDを書き換えてDutyを25%、50%、75%と切換えます。

6.4　固定幅整数一覧

表 6.3 にサンプルプログラムで使用する固定幅整数を示します。

表6.3　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）
int64_t	64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）
uint64_t	64ビット整数、符号なし（標準ライブラリにて定義）

6.5　定数／エラーコード一覧

表 6.4 にサンプルプログラムで使用する定数を示します。

表6.4　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_DUTY_25	((uint16_t) MTU_C_CYCLE * 3/4 + MTU_DEAD_TIME * 1/2) 注1	デューティ比25%を設定するための定数です。
MTU_DUTY_50	((uint16_t) MTU_C_CYCLE * 2/4 + MTU_DEAD_TIME * 1/2) 注1	デューティ比50%を設定するための定数です。
MTU_DUTY_75	((uint16_t) MTU_C_CYCLE * 1/4 + MTU_DEAD_TIME * 1/2) 注1	デューティ比75%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。
DUTY_CNT_MAX	(3)	デューティ比切換えを行う際に用いる最大値です。

注1 デューティ比 n% の設定値は MTU_DUTY_n = MTU_C_CYCLE * (1 - n/100) + MTU_DEAD_TIME * 1/2

6.6　構造体／共用体／列挙型一覧

図 6.2 ～図 6.14 にサンプルプログラムで使用する構造体／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 channnel 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 6.2　サンプルプログラムで使用する構造体／共用体／列挙体（1）

```
/* The possible return codes from the API functions. */
typedef enum mtu_pwm_err_e
{
    MTU_SUCCESS = 0,
    MTU_ERR_BAD_CHAN,                   // Invalid channel number.
    MTU_ERR_CH_NOT_OPENED,              // Channel not yet opened.
    MTU_ERR_CH_NOT_CLOSED,              // Channel still open from previous open.
    MTU_ERR_UNKNOWN_CMD,                // Control command is not recognized.
    MTU_ERR_INVALID_ARG,                // Argument is not valid for parameter.
    MTU_ERR_ARG_RANGE,                  // Argument is out of range for parameter.
    MTU_ERR_NULL_PTR,                   // Received null pointer; missing required argument.
    MTU_ERR_LOCK,                       // The lock procedure failed.
    MTU_ERR_UNDEF                       // Undefined/unknown error
} mtu_err_t;

/* 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_TOGGLE = 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_TOGGLE = 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 6.3　　サンプルプログラムで使用する構造体／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_4,      // PCLK/4
    MTU_SRC_CLK_DIV_16,     // PCLK/16
    MTU_SRC_CLK_DIV_64,     // PCLK/64
    MTU_SRC_CLK_DIV_256,    // PCLK/256
    MTU_SRC_CLK_DIV_1024,   // PCLK/1024
    MTU_SRC_CLK_DIV_2,      // PCLK/2
    MTU_SRC_CLK_DIV_8,      // PCLK/8
    MTU_SRC_CLK_DIV_32      // PCLK/32
} 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 6.4　　サンプルプログラムで使用する構造体／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 channnel not support */
    MTU_GRP_CH6 = 0x4000,
    MTU_GRP_CH7 = 0x8000,
    MTU_GRP_CH8 = 0x0008
} mtu_group_t;
```

図 6.5　　サンプルプログラムで使用する構造体／共用体／列挙体（4）

```
typedef struct mtu_timer_status_s
{
      uint32_t timer_count;              // The current channel counter value.
      bool     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.
      uint8_t  capture_flags;             // 1 if a capture event occurred, 0 if still waiting.
} mtu_capture_status_t;

typedef struct mtu_pwm_status_s
{
      bool    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 6.6　　サンプルプログラムで使用する構造体／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_NUM_TIMERS
} 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 6.7　　サンプルプログラムで使用する構造体／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                    filter_enable;     // Noise filter on or off.
} mtu_capture_settings_t;
```

図 6.8　　サンプルプログラムで使用する構造体／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 6.9　　サンプルプログラムで使用する構造体／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 6.10　サンプルプログラムで使用する構造体／共用体／列挙体（9）

```
/************ Type defines used with the R_MTU_PWM_Complement_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,  // Complementary PWM mode 1
    MTU_CMPL_PWM_MODE_2 = 0x0E,  // Complementary PWM mode 2
    MTU_CMPL_PWM_MODE_3 = 0x0F,  // Complementary 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 6.11　サンプルプログラムで使用する構造体／共用体／列挙体（10）

```
typedef enum
{
    MTU_CMPL_PWM_D_BF_OFF = 0,  // OFF
    MTU_CMPL_PWM_D_BF_ON_SYNC,   // SYNC
    MTU_CMPL_PWM_D_BF_ON_ASYNC,  // ASYNC
}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 6.12　サンプルプログラムで使用する構造体／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 6.13　サンプルプログラムで使用する構造体／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;             // Complementary 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 6.14　サンプルプログラムで使用する構造体／共用体／列挙体（13）

6.7　大域変数一覧

表 6.5 に大域変数一覧を示します。

表6.5　大域変数一覧

型	変数名	内容	使用関数
uint16_t	g_duty_rate[DUTY_CNT_MAX]	デューティ比データ	R_IRQ9_isr
uint8_t	g_duty_cnt	デューティ比切換えカウンタ	R_IRQ9_isr

6.8　関数一覧

表 6.6 に関数一覧を示します。

表6.6　関数一覧

関数名	ページ番号
main	26
init_mtu3	26
R_MTU_PWM_Complement_Open	27
R_MTU_PWM_Complement_Close	27
R_MTU_PWM_Complement_Control	28
R_IRQ9_isr	28
R_MTU_Timer_Open	29
R_MTU_Capture_Open	30
R_MTU_PWM_Open	31
R_MTU_Close	32
R_MTU_Control	33

6.9　関数仕様

6.9.1　main

main	
概　要	メイン処理
宣　言	int 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
リターン値	なし
補足	なし

6.9.2　init_mtu3

init_mtu3	
概　要	MTU 初期化処理
宣　言	void init_mtu3(void)
説　明	MTU3、MTU4 を初期化します。
引　数	なし
リターン値	なし
補足	なし

6.9.3　R_MTU_PWM_Complement_Open

R_MTU_PWM_Complement_Open		
概　要	相補 PWM モード設定処理	
ヘッダ	r_mtu3_if.h	
宣　言	mtu_err_t R_MTU_PWM_Complement_Open(mtu_cmpl_pwm_channel_t channel, mtu_cmpl_pwm_chnl_settings_t * pconfig)	
説　明	指定 MTU チャネル毎の相補 PWM 出力の設定を行います。	
引　数	mtu_cmpl_pwm_channel_t channel	相補 PWM チャネルを指定します。 MTU_CHANNEL_3_4 MTU_CHANNEL_6_7
	mtu_cmpl_pwm_chnl_settings_t * pconfig	相補 PWM 出力するためのコンフィグ設定のポインタです。詳細は 6.11　R_MTU_PWM_Complement_Open パラメーター覧を参照してください。
リターン値	相補 PWM オープン関数の実行結果を返します。 MTU_SUCCESS : コマンドは正常に終了しました MTU_ERR_BAD_CHAN : チャネル指定が無効です MTU_ERR_CH_NOT_CLOSED : チャネルは動作中です MTU_ERR_NULL_PTR : pconfig ポインタが NULL です MTU_ERR_INVALID_ARG : 引数の値が無効です	
補足	r_mtu3_config.h で定義される MTU3_CFG_PARAM_CHECKING_ENABLE を 1 にすることで、引数パラメータのチェック処理を有効にします。 関連関数 : R_MTU_PWM_Complement_Close()、R_MTU_PWM_Complement_Control()	

6.9.4　R_MTU_PWM_Complement_Close

R_MTU_PWM_Complement_Close		
概　要	相補 PWM 終了処理	
ヘッダ	r_mtu3_if.h	
宣　言	mtu_err_t R_MTU_PWM_Complement_Close(mtu_cmpl_pwm_channel_t channel)	
説　明	指定 MTU チャネル毎の相補 PWM 設定の終了処理を行います。	
引　数	mtu_cmpl_pwm_channel_t channel	相補 PWM チャネルを指定します。 MTU_CHANNEL_3_4 MTU_CHANNEL_6_7
リターン値	相補 PWM クローズ関数の実行結果を返します。 MTU_SUCCESS : コマンドは正常に終了しました MTU_ERR_CH_NOT_OPEN : チャネルは動作停止中です MTU_ERR_BAD_CHAN : チャネル指定が無効です	
補足	r_mtu3_config.h で定義される MTU3_CFG_PARAM_CHECKING_ENABLE を 1 にすることで、引数パラメータのチェック処理を有効にします。	

6.9.5 R_MTU_PWM_Complement_Control

R_MTU_PWM_Complement_Control		
概　要	相補 PWM 制御処理	
ヘッダ	r_mtu3_if.h	
宣　言	mtu_err_t R_MTU_PWM_Complement_Control(mtu_cmpl_pwm_channel_t channel, mtu_cmd_t cmd, void * pcmd_data)	
説　明	相補 PWM の開始、停止、PWM 動作中の情報取得を行います。	
引　数	mtu_cmpl_pwm_channel_t channel	相補 PWM チャネルを指定します。 MTU_CHANNEL_3_4 MTU_CHANNEL_6_7
	mtu_cmd_t cmd	制御するコマンドを指定します。 MTU_CMD_START MTU_CMD_STOP MTU_CMD_GET_STATUS
	void * pcmd_data	取得した情報の格納位置へのポインタです。詳細は 6.12　R_MTU_PWM_Complement_Control パラメーター一覧を参照してください。
リターン値	相補 PWM コントロール関数の実行結果を返します。 MTU_SUCCESS : コマンドは正常に終了しました MTU_ERR_BAD_CHAN : チャネル指定が無効です MTU_ERR_CH_NOT_OPEN : チャネルは動作停止中です MTU_ERR_UNKNOWN_CMD : コマンドは無効です MTU_ERR_NULL_PTR : pcmd_data ポインタが NULL です	
補足	r_mtu3_config.h で定義される MTU3_CFG_PARAM_CHECKING_ENABLE を 1 にすることで、引数パラメータのチェック処理を有効にします。	

6.9.6 R_IRQ9_isr

R_IRQ9_isr	
概　要	IRQ9 割り込み（IRQ 端子割り込み 5）処理
宣　言	void R_IRQ9_isr(void)
説　明	バッファレジスタ MTU3.TGRD、MTU4.TGRC、MTU4.TGRD を書き換えて Duty を 25%、50%、75% と切換えます。
引　数	なし
リターン値	なし
補足	なし

6.9.7　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	mtu_err_t R_MTU_Timer_Open(mtu_channel_t channel, mtu_timer_chnl_settings_t *pconfig, void (*pcallback)(void *pdata))	
説　明	指定した MTU チャネル毎のコンペア／マッチタイミング操作のための設定を行います。	
引　数	mtu_channel_t channel	コンペア／マッチする MTU チャネルを指定します。 MTU_CHANNEL_0 MTU_CHANNEL_1 MTU_CHANNEL_2 MTU_CHANNEL_3 MTU_CHANNEL_4 MTU_CHANNEL_6 MTU_CHANNEL_7 MTU_CHANNEL_8
	mtu_timer_chnl_settings_t *pconfig	コンペア／マッチするためのコンフィグ設定のポインタです。詳細は 6.13　R_MTU_Timer_Open パラメーター覧を参照してください。
	void (*pcallback)(void *pdata)	コールバック関数へのポインタを指定します。
リターン値	コンペア／マッチオープン関数の実行結果を返します。 MTU_SUCCESS : コマンドは正常に終了しました MTU_TIMERS_ERR_BAD_CHAN : チャネル指定が無効です MTU_TIMERS_ERR_CH_NOT_CLOSED : チャネルは動作中です MTU_TIMERS_ERR_NULL_PTR : pconfig ポインタが NULL です MTU_ERR_ARG_RANGE : 引数の指定した範囲が無効です MTU_TIMERS_ERR_INVALID_ARG : 引数の値が無効です	
補足	r_mtu3_config.h で定義される MTU3_CFG_PARAM_CHECKING_ENABLE を 1 にすることで、引数パラメータのチェック処理を有効にします。 関連関数 : R_MTU_Close()、R_MTU_Control()	

6.9.8　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	mtu_err_t R_MTU_Capture_Open(mtu_channel_t channel, mtu_capture_chnl_settings_t *pconfig, void (*pcallback)(void *pdata))	
説　明	指定した MTU チャネル毎の入力キャプチャの設定を行います。	
引　数	mtu_channel_t channel	キャプチャする MTU チャネルを指定します。 MTU_CHANNEL_0 MTU_CHANNEL_1 MTU_CHANNEL_2 MTU_CHANNEL_3 MTU_CHANNEL_4 MTU_CHANNEL_6 MTU_CHANNEL_7 MTU_CHANNEL_8
	mtu_capture_chnl_settings_t *pconfig	キャプチャするためのコンフィグ設定のポインタです。詳細は 6.14　R_MTU_Capture_Open パラメーター覧を参照してください。
	void (*pcallback)(void *pdata)	コールバック関数へのポインタを指定します。
リターン値	キャプチャオープン関数の実行結果を返します。 MTU_SUCCESS : コマンドは正常に終了しました MTU_TIMERS_ERR_BAD_CHAN : チャネル指定が無効です MTU_TIMERS_ERR_CH_NOT_CLOSED : チャネルは動作中です MTU_TIMERS_ERR_NULL_PTR : pconfig ポインタが NULL です MTU_TIMERS_ERR_INVALID_ARG : 引数の値が無効です	
補足	r_mtu3_config.h で定義される MTU3_CFG_PARAM_CHECKING_ENABLE を 1 にすることで、引数パラメータのチェック処理を有効にします。 関連関数 : R_MTU_Close()、R_MTU_Control()	

6.9.9　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	mtu_err_t R_MTU_PWM_Open(mtu_channel_t channel, mtu_pwm_chnl_settings_t *pconfig, void (*pcallback)(void *pdata))	
説　明	指定した MTU チャネル毎の基本的な PWM 動作のための設定を行います。	
引　数	mtu_channel_t channel	PWM 出力するポートを指定します。 MTU_CHANNEL_0 注 1 MTU_CHANNEL_1 注 1 MTU_CHANNEL_2 注 1 MTU_CHANNEL_3 MTU_CHANNEL_4 MTU_CHANNEL6 MTU_CHANNEL7 注 1.　PWM モード 2 を指定した場合はチャネル 0、1、2 のみが設定可能です。
	mtu_pwm_chnl_settings_t * pconfig	基本 PWM 出力するためのコンフィグ設定のポインタです。詳細は 6.15　R_MTU_PWM_Open パラメーター覧を参照してください。
	void (*pcallback)(void *pdata)	コールバック関数へのポインタを指定します。
リターン値	PWM オープン関数の実行結果を返します。 MTU_SUCCESS : コマンドは正常に終了しました MTU_ERR_BAD_CHAN : チャネル指定が無効です MTU_ERR_CH_NOT_CLOSED : チャネルは動作中です MTU_ERR_NULL_PTR : pconfig ポインタは NULL です MTU_ERR_ARG_RANGE : 引数の指定した範囲が無効です MTU_ERR_INVALID_ARG : 引数の値が無効です	
補足	r_mtu3_config.h で定義される MTU3_CFG_PARAM_CHECKING_ENABLE を 1 にすることで、引数パラメータのチェック処理を有効にします。 関連関数 : R_MTU_Close()、R_MTU_Control()	

6.9.10　R_MTU_Close

R_MTU_Close		
概　要	MTU 終了処理	
ヘッダ	r_mtu3_if.h	
宣　言	mtu_err_t R_MTU_Close(mtu_channel_t channel)	
説　明	コンペア／マッチ、入力キャプチャ、PWM で設定した MTU チャネル毎の設定を無効にします。	
引　数	mtu_channel_t channel	無効にする MTU チャネルを指定します。 MTU_CHANNEL_0 MTU_CHANNEL_1 MTU_CHANNEL_2 MTU_CHANNEL_3 MTU_CHANNEL_4 MTU_CHANNEL_6 MTU_CHANNEL_7 MTU_CHANNEL_8
リターン値	PWM クローズ関数の実行結果を返します。 MTU_SUCCESS : コマンドは正常に終了しました MTU_TIMERS_ERR_CH_NOT_OPEN : チャネルは動作停止中です MTU_TIMERS_ERR_BAD_CHAN : チャネル指定が無効です	
補足	r_mtu3_config.h で定義される MTU3_CFG_PARAM_CHECKING_ENABLE を 1 にすることで、引数パラメータのチェック処理を有効にします。 関連関数 : R_MTU_Timer_Open()、R_MTU_Capture_Open()、R_MTU_PWM_Open()	

6.9.11　R_MTU_Control

R_MTU_Control		
概　要	MTU 制御処理	
ヘッダ	r_mtu3_if.h	
宣　言	mtu_err_t R_MTU_Control(mtu_channel_t channel, mtu_cmd_t cmd, void * pcmd_data)	
説　明	指定した MTU チャネル毎の特別なハードウェアまたはソフトウェアの動作の処理を行います。	
引　数	mtu_channel_t channel	MTU チャネルを指定します。 MTU_CHANNEL_0 MTU_CHANNEL_1 MTU_CHANNEL_2 MTU_CHANNEL_3 MTU_CHANNEL_4 MTU_CHANNEL_6 MTU_CHANNEL_7 MTU_CHANNEL_8
	mtu_cmd_t cmd	制御するコマンドを指定します。 MTU_CMD_START MTU_CMD_STOP MTU_CMD_SAFE_STOP MTU_CMD_RESTART MTU_CMD_SYNCHRONIZE MTU_CMD_GET_STATUS MTU_CMD_CLEAR_EVENTS MTU_CMD_SET_CAPT_EDGE
	void * pcmd_data	cmd で設定したコマンドに合わせた形式で指定する必要があります。詳細は 6.16　R_MTU_Control パラメーター一覧を参照してください。
リターン値	コントロール関数の実行結果を返します。 MTU_SUCCESS : コマンドは正常に終了しました MTU_TIMERS_ERR_CH_NOT_OPEN : チャネルは動作停止中です MTU_TIMERS_ERR_BAD_CHAN : チャネル指定が無効です MTU_TIMERS_ERR_UNKNOWN_CMD : コマンドは無効です MTU_TIMERS_ERR_NULL_PTR : pcmd_data ポインタが NULL です MTU_TIMERS_ERR_INVALID_ARG : 引数の値が無効です	
補足	r_mtu3_config.h で定義される MTU3_CFG_PARAM_CHECKING_ENABLE を 1 にすることで、引数パラメータのチェック処理を有効にします。 関連関数 : R_MTU_Timer_Open()、R_MTU_Capture_Open()、R_MTU_PWM_Open()	

6.10　フローチャート

6.10.1　メイン処理

図 6.15 にメイン処理のフローチャートを示します。

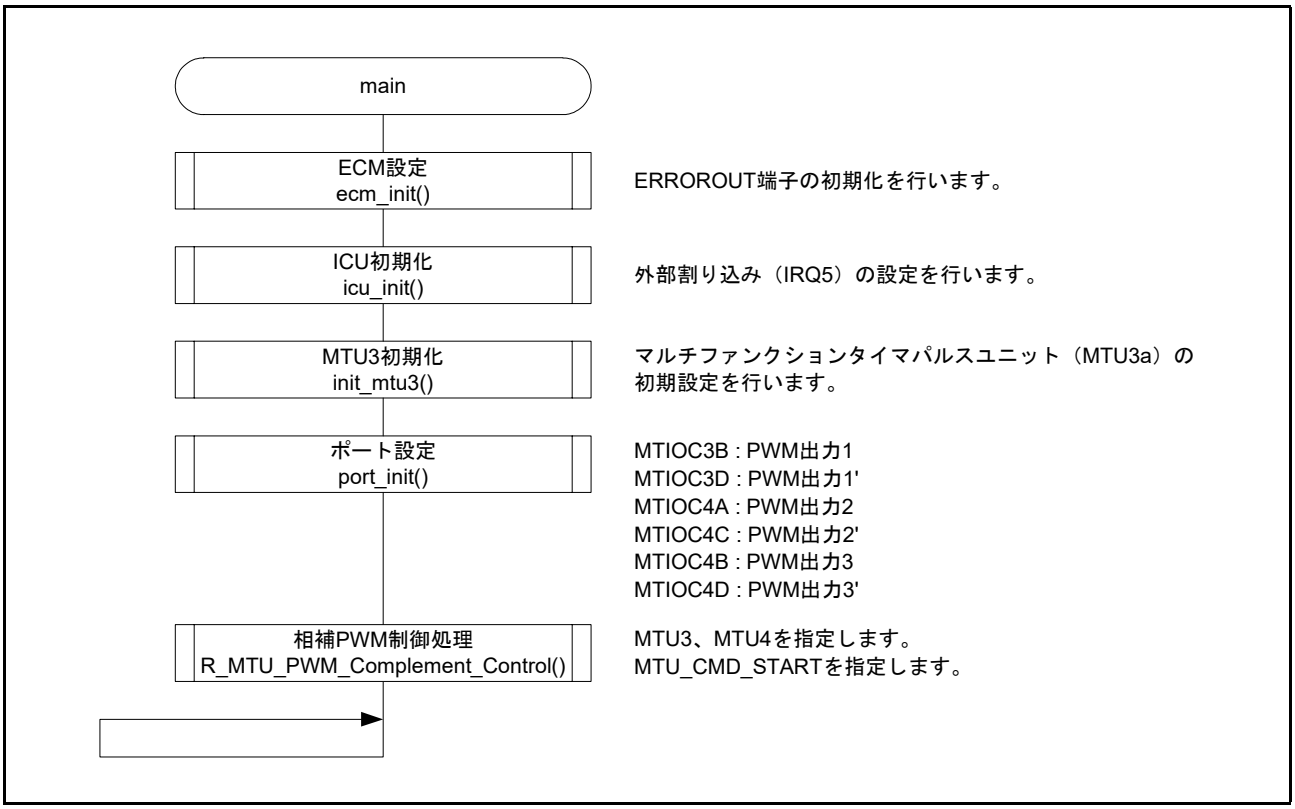


図 6.15　メイン処理

6.10.2　MTU 初期化処理

図 6.16 に MTU 初期化処理のフローチャートを示します。

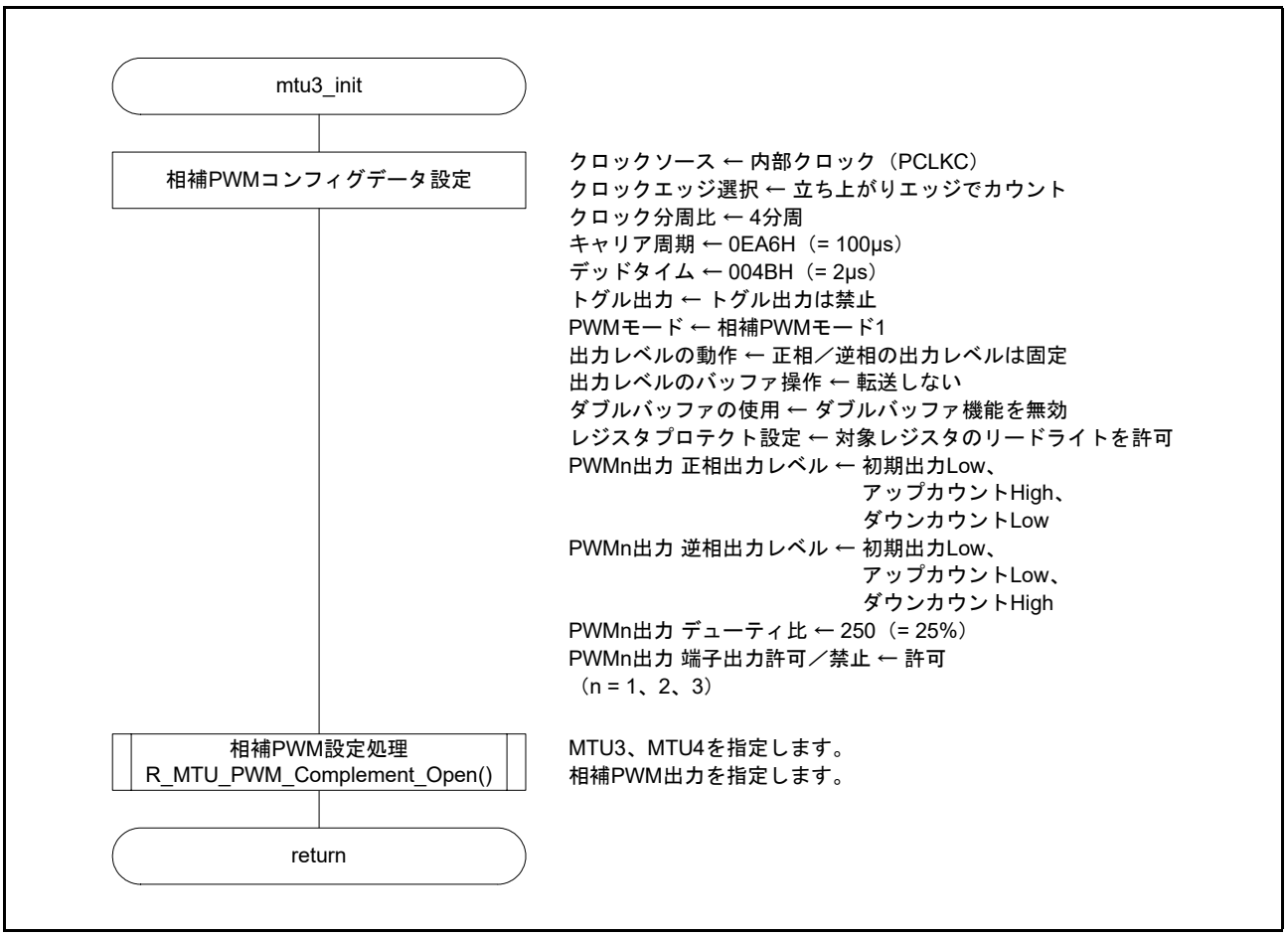
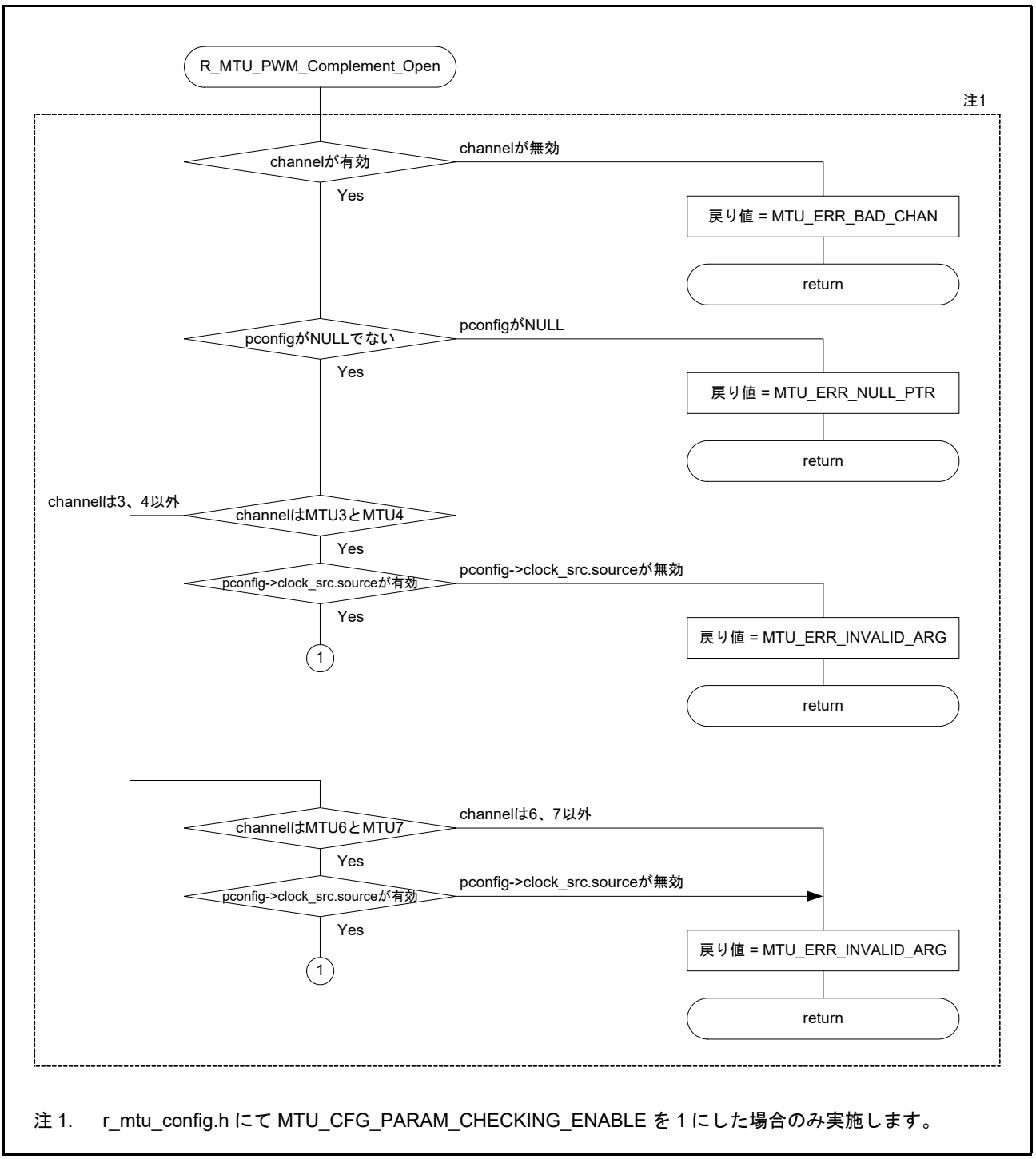


図 6.16　MTU 初期化処理

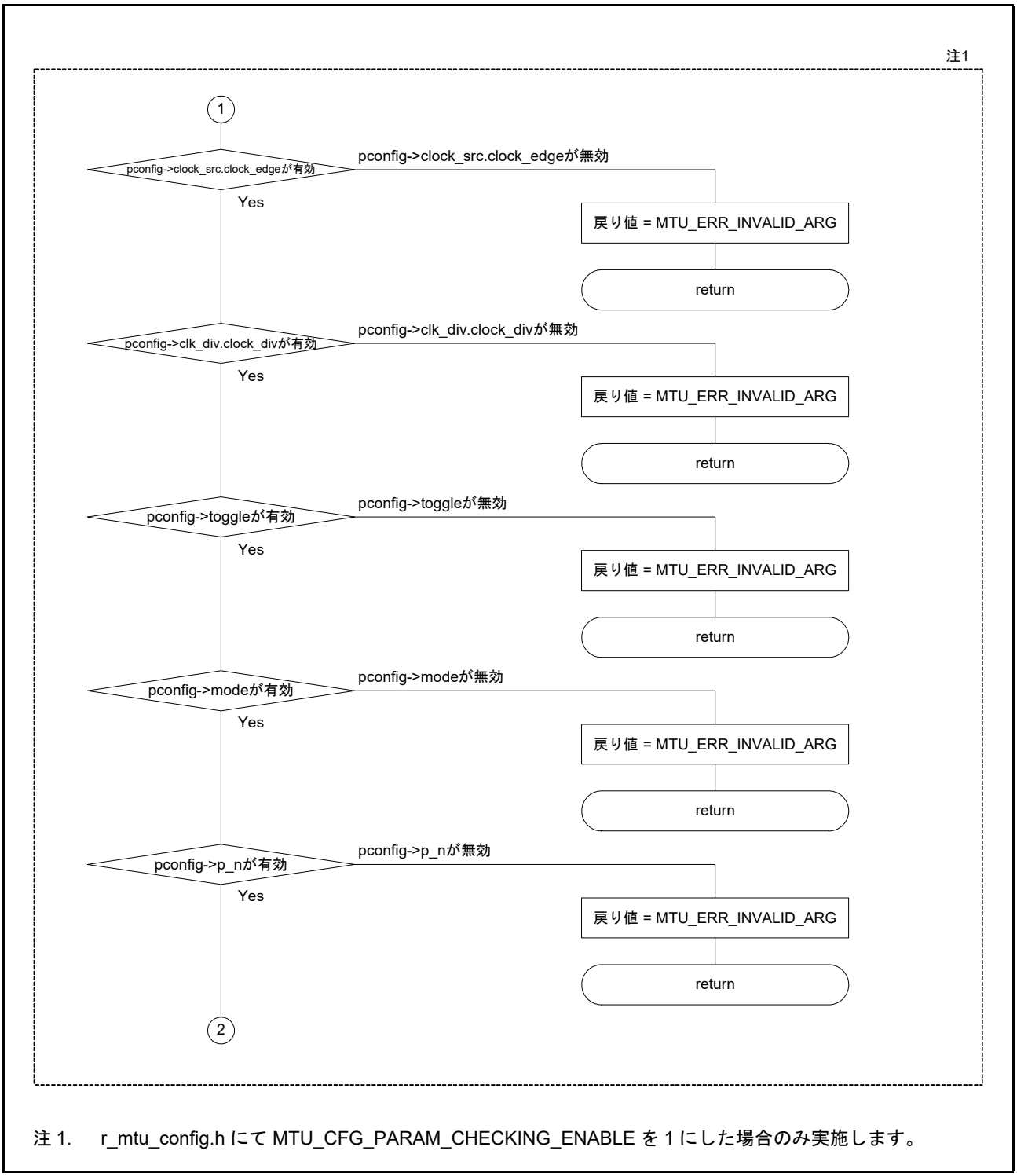
6.10.3　相補 PWM モード設定処理

図 6.17 ～図 6.21 に相補 PWM モード設定処理のフローチャートを示します。



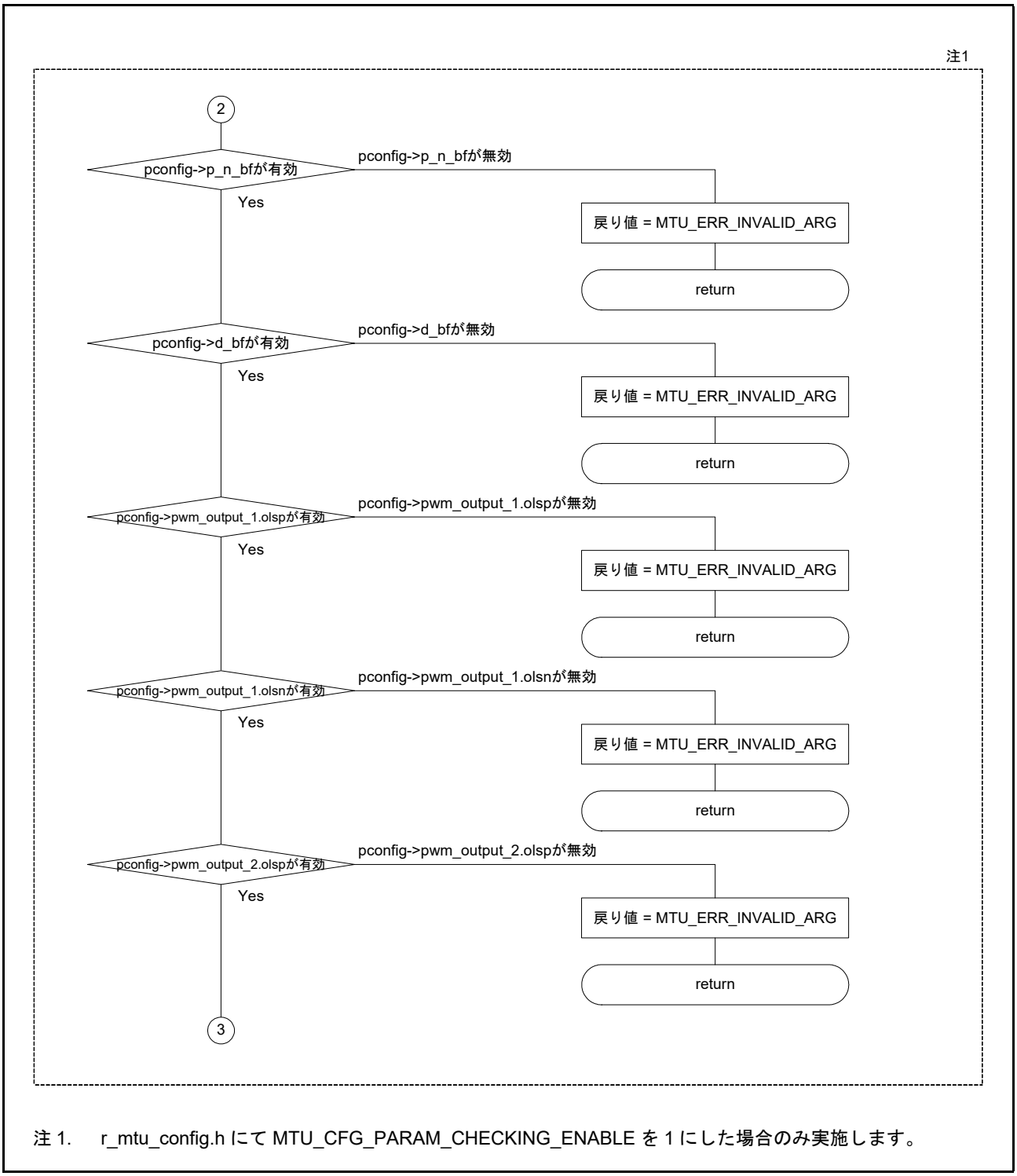
注 1.　r_mtu_config.h にて MTU_CFG_PARAM_CHECKING_ENABLE を 1 にした場合のみ実施します。

図 6.17　相補 PWM モード設定処理（1）



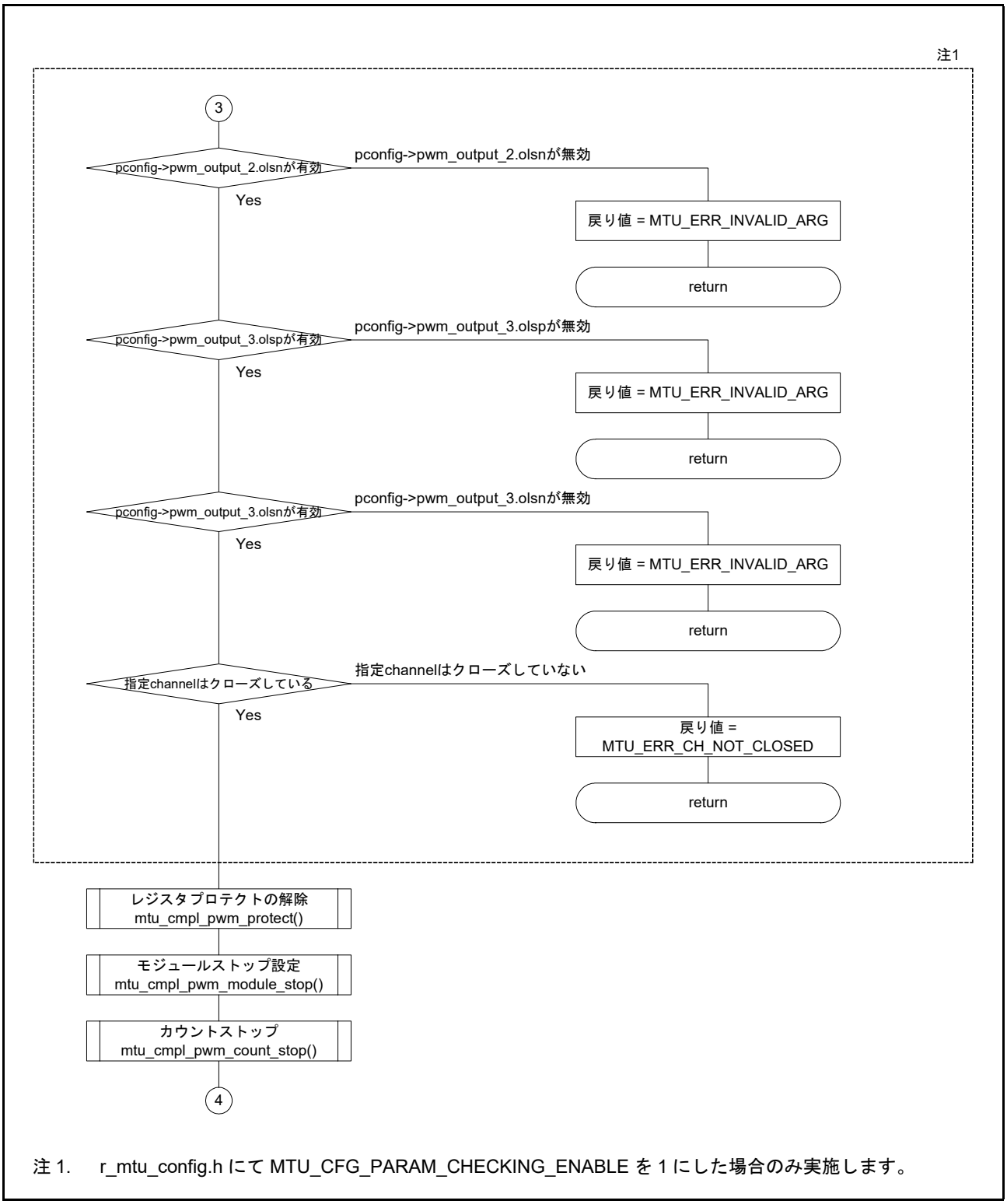
注 1.　r_mtu_config.h にて MTU_CFG_PARAM_CHECKING_ENABLE を 1 にした場合のみ実施します。

図 6.18　相補 PWM モード終了処理（2）



注 1.　r_mtu_config.h にて MTU_CFG_PARAM_CHECKING_ENABLE を 1 にした場合のみ実施します。

図 6.19　　相補 PWM モード終了処理（3）



注 1.　r_mtu_config.h にて MTU_CFG_PARAM_CHECKING_ENABLE を 1 にした場合のみ実施します。

図 6.20　相補 PWM モード終了処理（4）

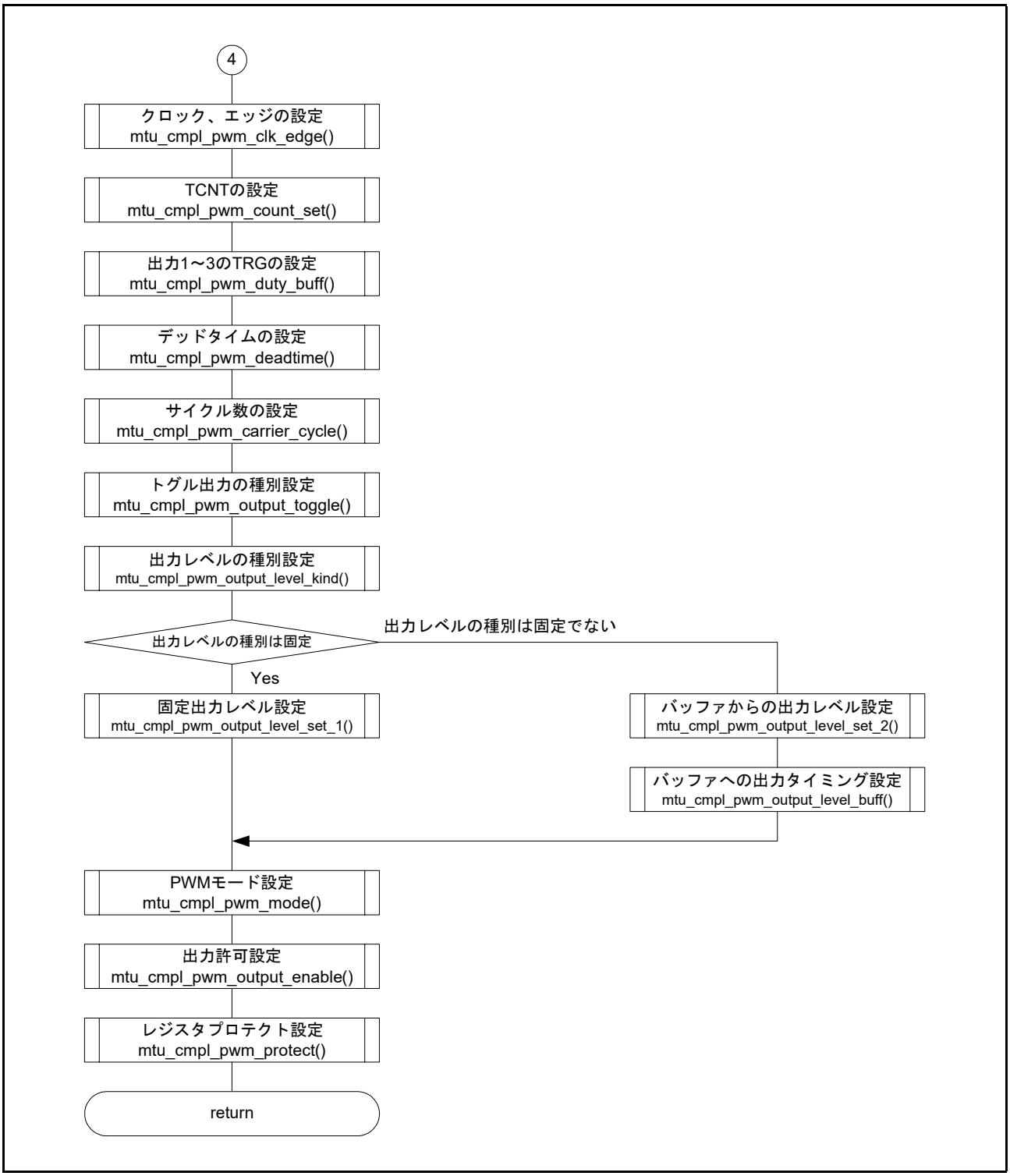
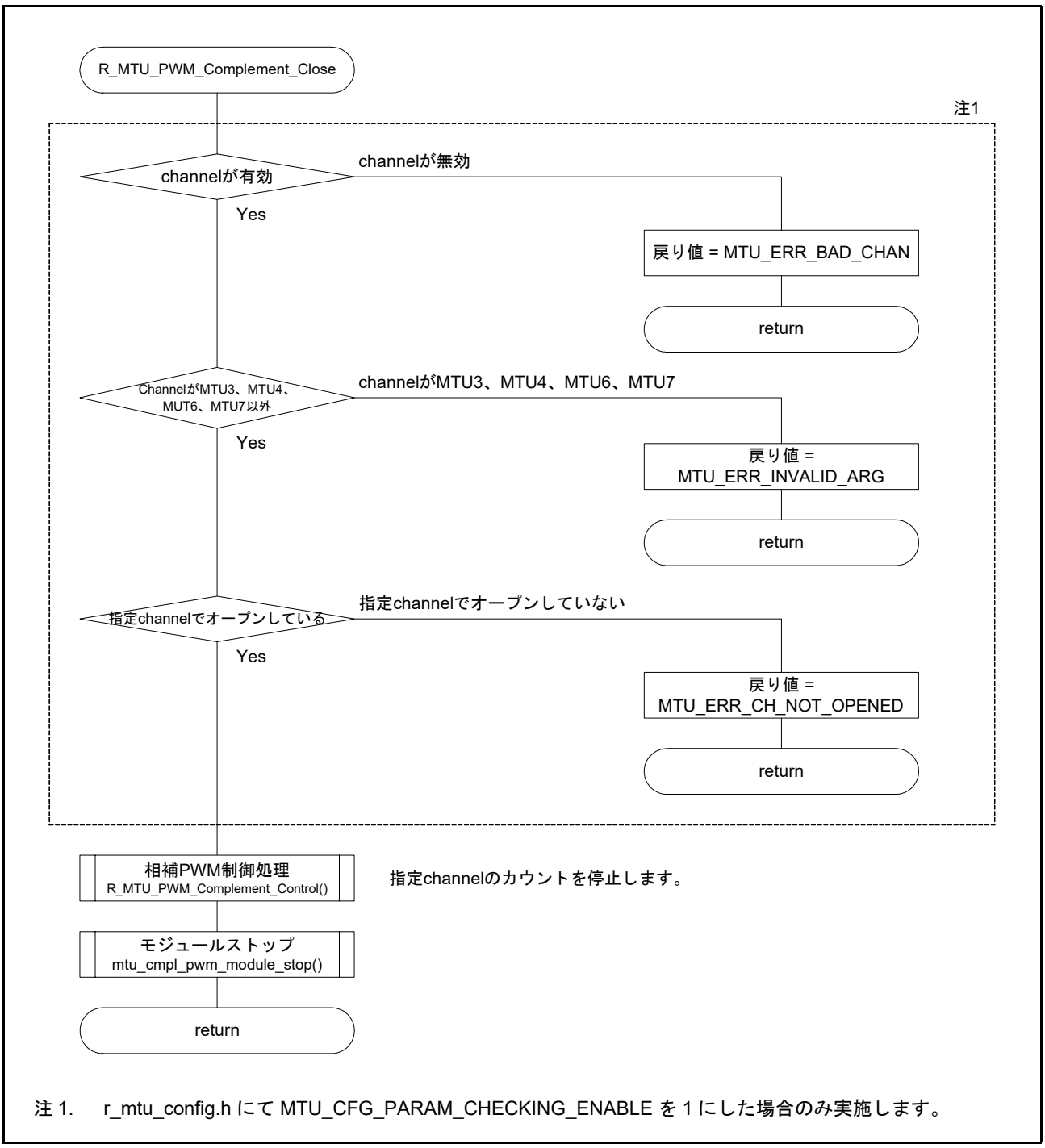


図 6.21　相補 PWM モード終了処理（5）

6.10.4　相補 PWM モード終了処理

図 6.22 に相補 PWM モード終了処理のフローチャートを示します。

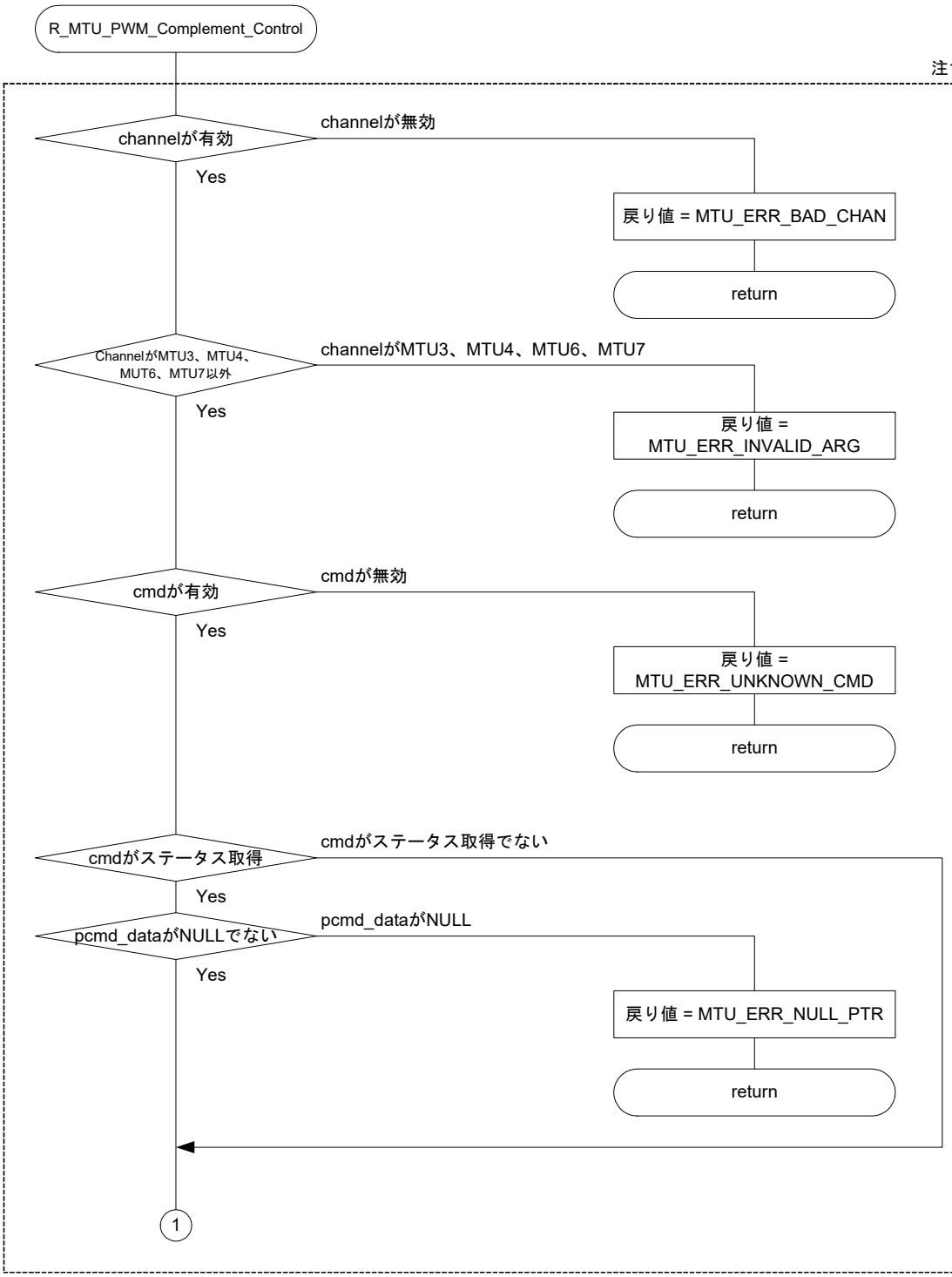


注 1.　r_mtu_config.h にて MTU_CFG_PARAM_CHECKING_ENABLE を 1 にした場合のみ実施します。

図 6.22　相補 PWM モード終了処理

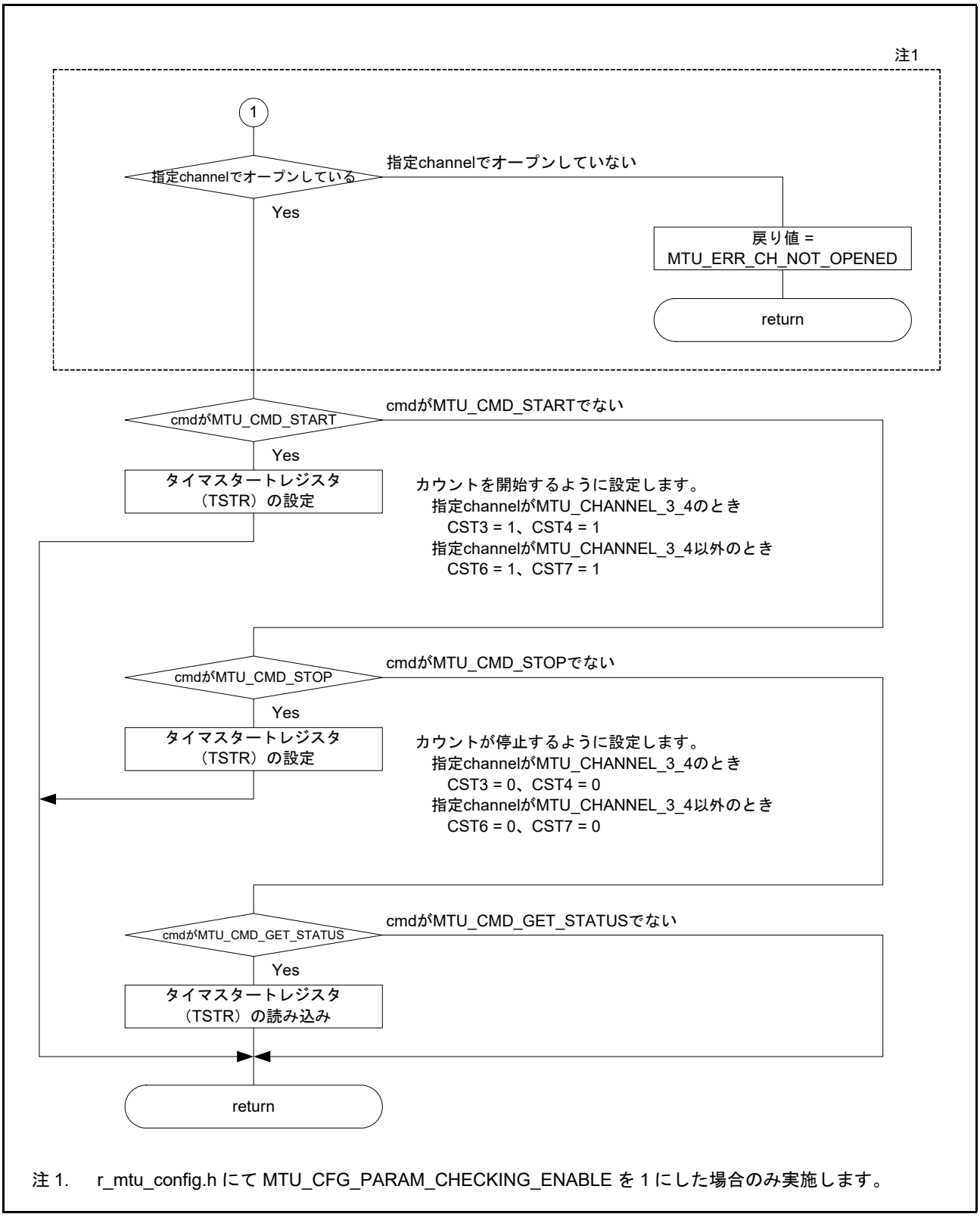
6.10.5　相補 PWM 制御処理

図 6.23 に相補 PWM 制御処理のフローチャートを示します。



注 1.　r_mtu_config.h にて MTU_CFG_PARAM_CHECKING_ENABLE を 1 にした場合のみ実施します。

図 6.23　相補 PWM 制御処理（1）



注 1.　r_mtu_config.h にて MTU_CFG_PARAM_CHECKING_ENABLE を 1 にした場合のみ実施します。

図 6.24　　相補 PWM 制御処理（2）

6.10.6　IRQ9 割り込み（IRQ 端子割り込み 5）処理

図 6.25 に IRQ9 割り込み（IRQ 端子割り込み 5）処理のフローチャートを示します。

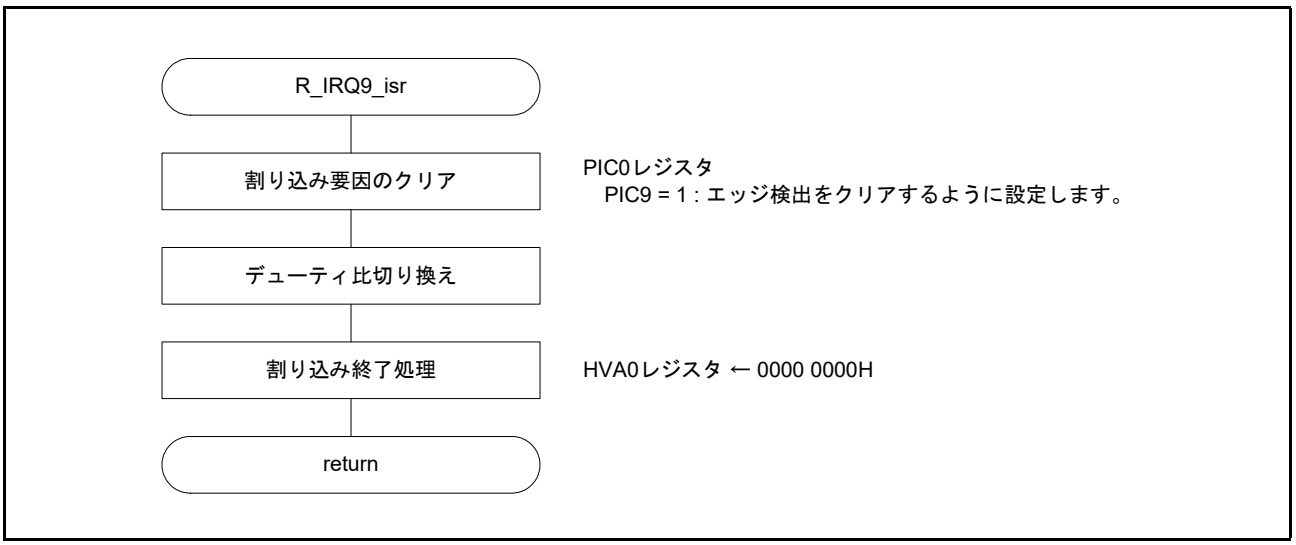
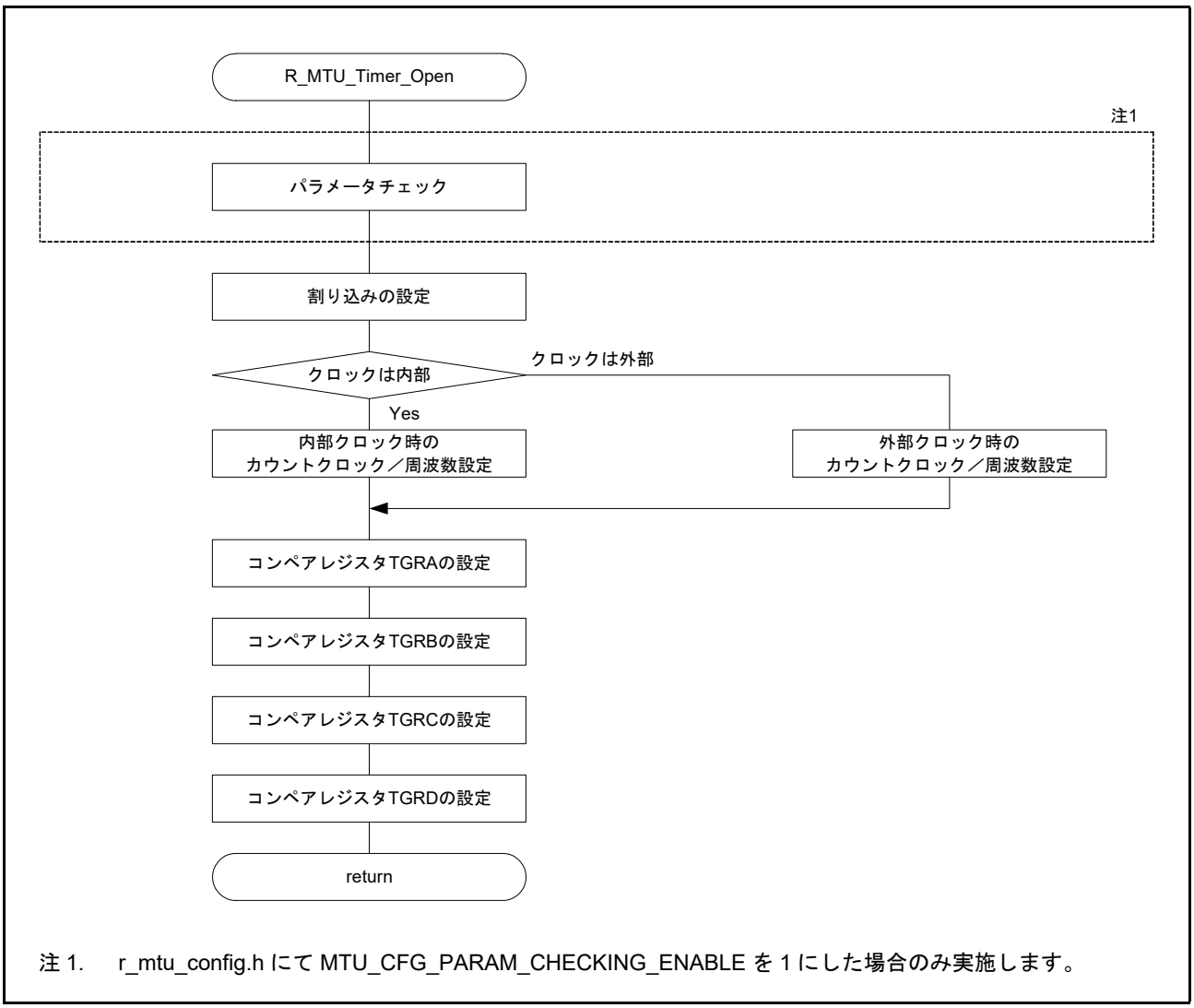


図 6.25　IRQ9 割り込み（IRQ 端子割り込み 5）処理

6.10.7　MTU コンペア／マッチ設定処理

図 6.26 に MTU コンペア／マッチ設定処理のフローチャートを示します。

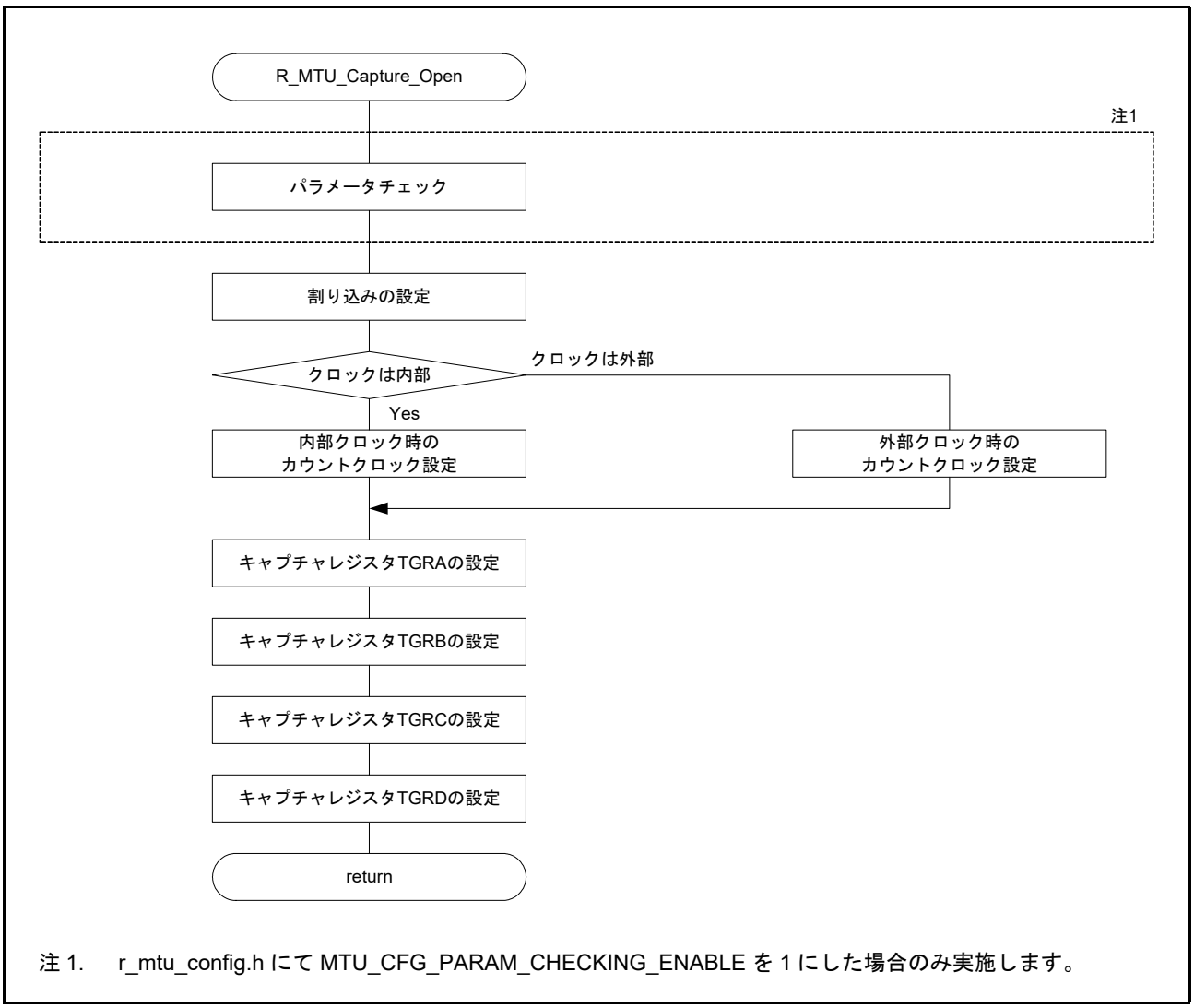


注 1.　r_mtu_config.h にて MTU_CFG_PARAM_CHECKING_ENABLE を 1 にした場合のみ実施します。

図 6.26　MTU コンペア／マッチ設定処理

6.10.8　MTU キャプチャ処理

図 6.27 に MTU キャプチャ処理のフローチャートを示します。

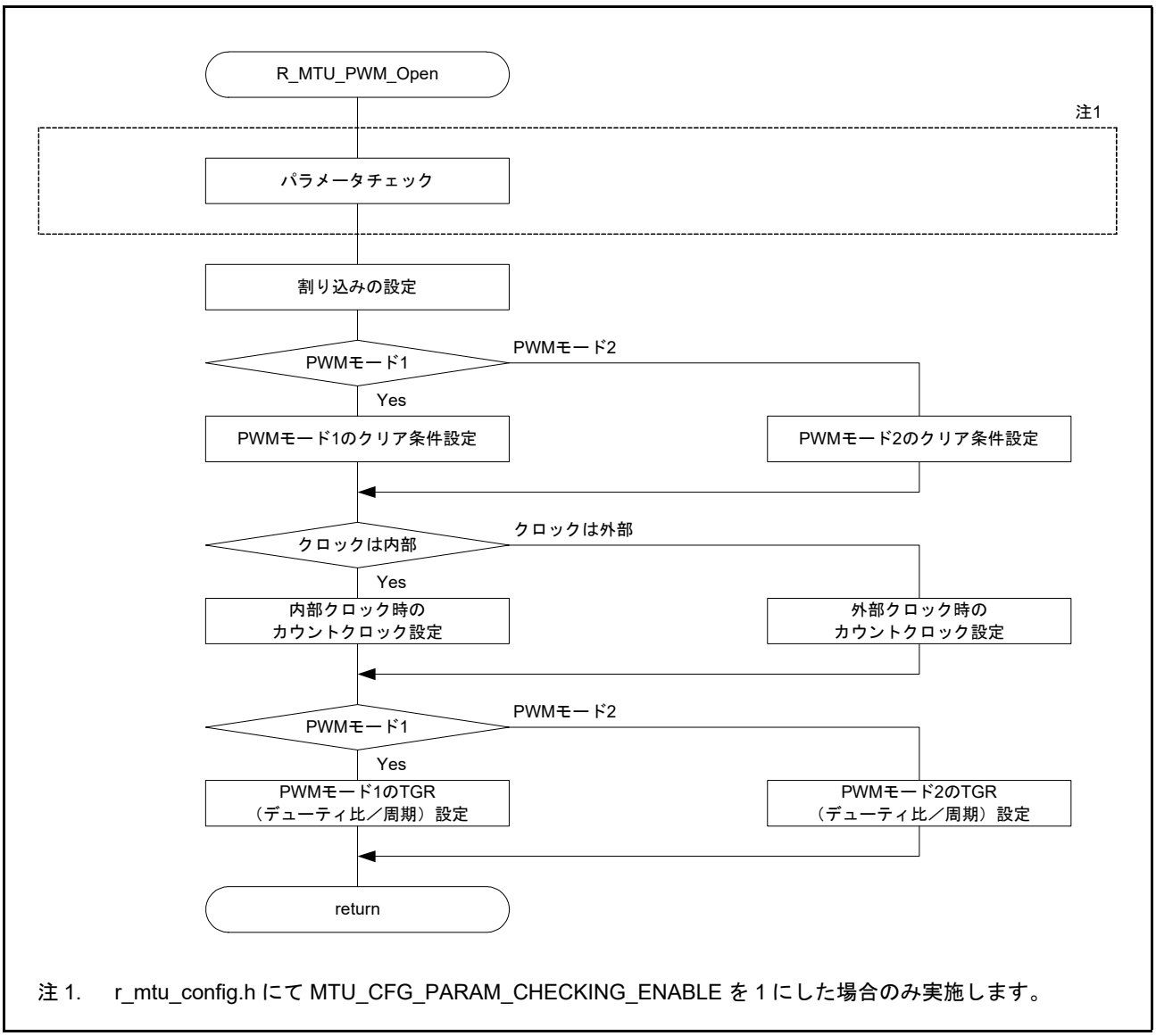


注 1.　r_mtu_config.h にて MTU_CFG_PARAM_CHECKING_ENABLE を 1 にした場合のみ実施します。

図 6.27　MTU キャプチャ処理

6.10.9　MTU PWM 処理

図 6.28 に MTU PWM 処理のフローチャートを示します。

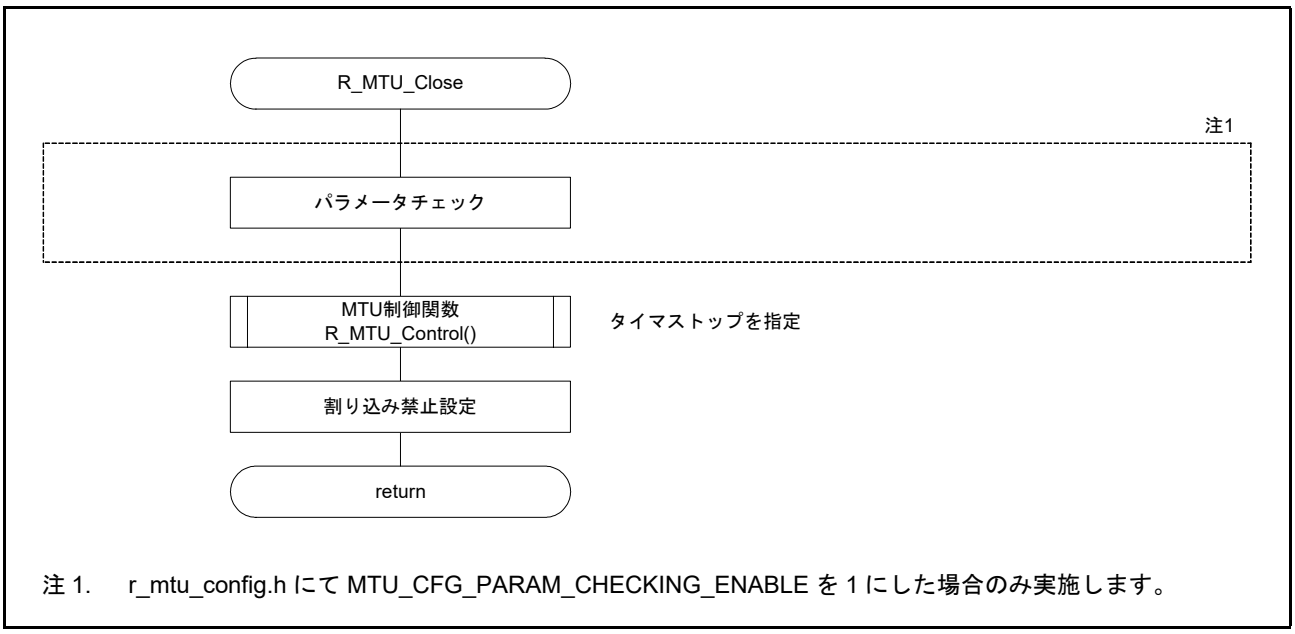


注 1.　r_mtu_config.h にて MTU_CFG_PARAM_CHECKING_ENABLE を 1 にした場合のみ実施します。

図 6.28　MTU PWM 処理

6.10.10　MTU 終了処理

図 6.29 に MTU 終了処理のフローチャートを示します。

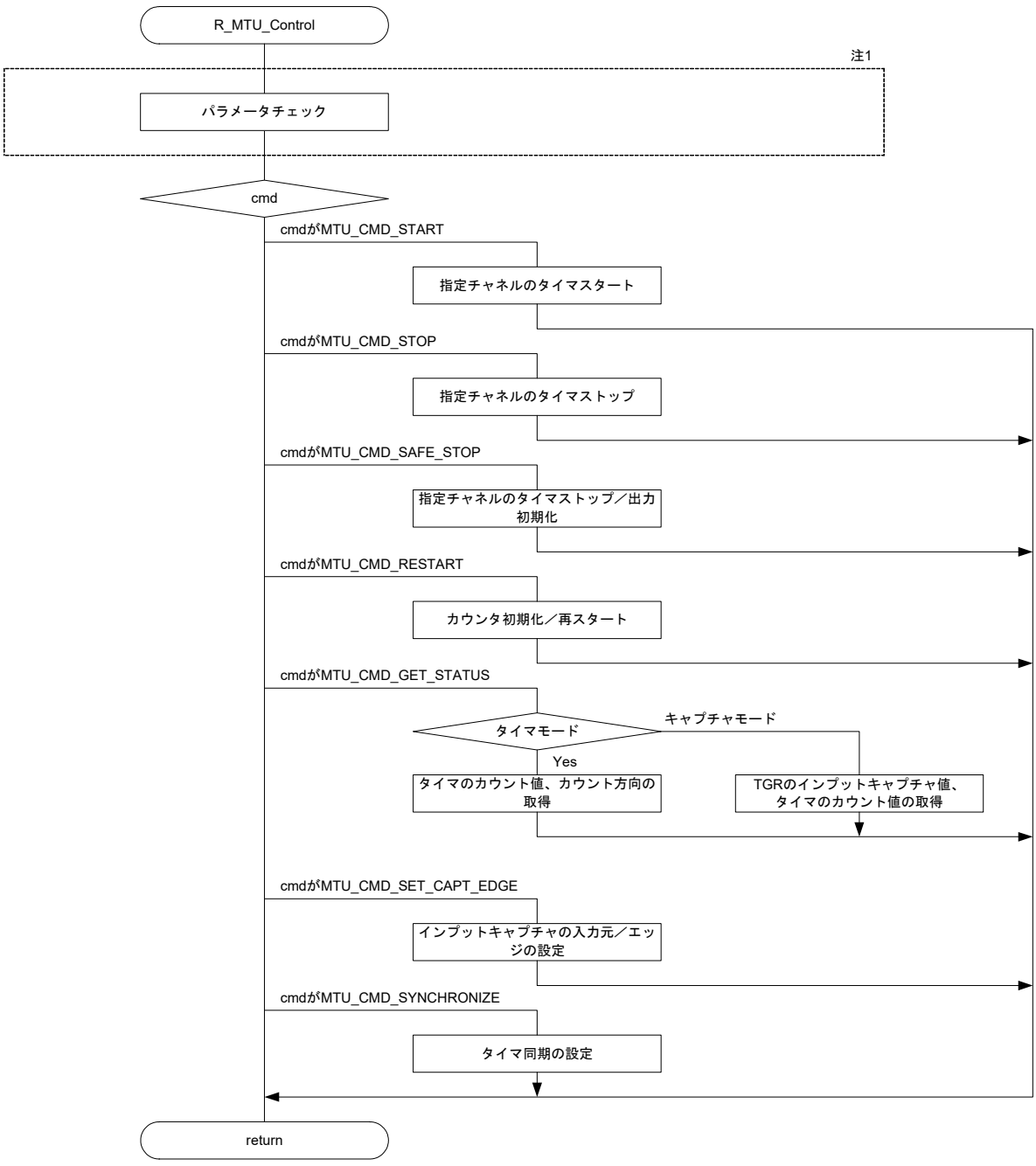


注 1.　r_mtu_config.h にて MTU_CFG_PARAM_CHECKING_ENABLE を 1 にした場合のみ実施します。

図 6.29　MTU 終了処理

6.10.11　MTU 制御処理

図 6.30 に MTU 制御処理のフローチャートを示します。



注 1.　r_mtu_config.h にて MTU_CFG_PARAM_CHECKING_ENABLE を 1 にした場合のみ実施します。

図 6.30　MTU 制御処理

6.11　R_MTU_PWM_Complement_Open パラメーター覧

R_MTU_PWM_Complement_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6.7　R_MTU_PWM_Complement_Openパラメーター覧

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。

6.11.1　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。

6.11.2 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。	

6.11.3 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	なし	

6.11.4 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	なし	

6.11.5 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	なし	

6.11.6 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。	

6.11.7 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	なし	

6.11.8　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。	

6.11.9　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。	

6.11.10　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。	

6.11.11　protect

protect		
概　要	MTU3、MTU4 の誤書き込み防止設定をします。	
ヘッダ	r_mtu3_if.h	
説　明	MTU3、MTU4 の対象レジスタおよび対象カウンタに対する誤書き込み防止設定をします。	
パラメータ	MTU_CMPL_PWM_PROTECT_OFF	リードライトを許可する
	MTU_CMPL_PWM_PROTECT_ON	リードライトを禁止する
補足	誤書き込み防止の対象レジスタおよび対象カウンタの詳細についてはユーザーズマニュアルのタイマリードライトイネーブルレジスタ（TRWERA、TRWERB）を参照してください。	

6.11.12　pwm_output_X.olsp（X = 1、2、3）

pwm_output_X.olsp		
概　要	PWM 出力 X（X = 1、2、3）の正相の出力レベルを設定します。	
ヘッダ	r_mtu3_if.h	
説　明	相補 PWM 出力 X（X = 1、2、3）の正相の出力レベルを設定します。	
パラメータ	MTU_PIN_OLSP_HI_UPLO_DNHI	初期出力 High、アップカウント Low、ダウンカウント High
	MTU_PIN_OLSP_LO_UPHI_DNLO	初期出力 Low、アップカウント High、ダウンカウント Low
補足	なし	

6.11.13　pwm_output_X.olsn（X = 1、2、3）

pwm_output_X.olsn		
概　要	PWM 出力 X（X = 1、2、3）の逆相の出力レベルを設定します。	
ヘッダ	r_mtu3_if.h	
説　明	相補 PWM 出力 X（X = 1、2、3）の逆相の出力レベルを選択します。	
パラメータ	MTU_PIN_OLSN_HI_UPHI_DNLO	初期出力 High、アップカウント High、ダウンカウント Low
	MTU_PIN_OLSN_LO_UPLO_DNHI	初期出力 Low、アップカウント Low、ダウンカウント High
補足	なし	

6.11.14　pwm_output_X.duty（X = 1、2、3）

pwm_output_X.duty		
概　要	PWM 出力 X（X = 1, 2, 3）のデューティ比を設定します。	
ヘッダ	r_mtu3_if.h	
説　明	相補 PWM 出力 X（X = 1、2、3）のデューティ比を設定します。 デューティ比は、0.1%単位で指定します。 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）	デューティ比を 0.1%単位で指定します （0 ～ 1000）。
補足	なし	

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

pwm_output_X.output		
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	なし	

6.12　R_MTU_PWM_Complement_Control パラメーター一覧

R_MTU_PWM_Complement_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。

6.12.1　cmd = MTU_CMD_START 時

本コマンド指定時は、R_MTU_PWM_Complement_Control 関数の第一引数 channel によるチャ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。

6.12.2　cmd = MTU_CMD_STOP 時

本コマンド指定時は、R_MTU_PWM_Complement_Control 関数の第一引数 channel によるチャ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。

6.12.3　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6.8　cmd = MTU_CMD_GET_STATUS時パラメーター覧

パラメータ	概要
c_st	タイマのカウント状態（動作、停止）
d_st	タイマのカウント方向

6.13　R_MTU_Timer_Open パラメーター覧

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6.9　R_MTU_Timer_Openパラメーター覧

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。

6.13.1　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 の各チャ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

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.　チャネル 0、2 のみ設定可能です。
注 2.　チャネル 0 のみ設定可能です。
注 3.　チャネル 1 のみ設定可能です。

6.13.2 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 の各チャ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。	

6.13.3 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.　チャネル 0、3、4、6、7、8 のみ設定可能です。	

6.13.4 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。	

6.13.5 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。	

6.13.6　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。
	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	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。	

6.14　R_MTU_Capture_Open パラメーター一覧

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6.10　R_MTU_Capture_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。
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）。

6.14.1　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.　チャネル 0、2 のみ指定可能です。
注 2.　チャネル 0 のみ指定可能です。
注 3.　チャネル 1 のみ指定可能です。

6.14.2 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。	

6.14.3 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	なし	

6.14.4 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.　チャネル 0、3、4、6、7、8 のみ指定可能です。	

6.14.5　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。	

6.14.6　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	なし	

6.14.7　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。	

6.15　R_MTU_PWM_Open パラメーター覧

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6.11　R_MTU_PWM_Openパラメーター覧

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。

6.15.1　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

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.　チャネル 0、2 のみ指定可能です。
注 2.　チャネル 0 のみ指定可能です。

6.15.2 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 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。	

6.15.3 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 TGRC 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。	

6.15.4 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.　チャネル 0、3、4、6、7、8 のみ指定可能です。	

6.15.5　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.　チャネル 0 ～ 4、6、7 のみ指定可能です。 注 2.　チャネル 0 ～ 2 のみ指定可能です。

6.15.6　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。	

6.15.7　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。	

6.15.8　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が、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。	

6.16　R_MTU_Control パラメーター覧

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。

6.16.1　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

pcmd_data		
概　要	使用するチャ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
	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);	

パラメータ	MTU_GRP_CH0	チャネル番号 0
	MTU_GRP_CH1	チャネル番号 1
	MTU_GRP_CH2	チャネル番号 2
	MTU_GRP_CH3	チャネル番号 3
	MTU_GRP_CH4	チャネル番号 4
	MTU_GRP_CH6	チャネル番号 6
	MTU_GRP_CH7	チャネル番号 7
	MTU_GRP_CH8	チャネル番号 8
補足	注 .　1 チャ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 で複数チャネルを指定した場合は、第一引数の値は無効になります。	

6.16.2　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

pcmd_data		
概　要	使用するチャ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
	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);	

パラメータ	MTU_GRP_CH0	チャネル番号 0
	MTU_GRP_CH1	チャネル番号 1
	MTU_GRP_CH2	チャネル番号 2
	MTU_GRP_CH3	チャネル番号 3
	MTU_GRP_CH4	チャネル番号 4
	MTU_GRP_CH6	チャネル番号 6
	MTU_GRP_CH7	チャネル番号 7
	MTU_GRP_CH8	チャネル番号 8
補足	注 .　1 チャ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 で複数チャネルを指定した場合は、第一引数の値は無効となります。	

6.16.3　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。

6.16.4　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。

6.16.5　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

pcmd_data		
概　要	使用するチャ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
	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);	

パラメータ	MTU_GRP_CH0	チャネル番号 0
	MTU_GRP_CH1	チャネル番号 1
	MTU_GRP_CH2	チャネル番号 2
	MTU_GRP_CH3	チャネル番号 3
	MTU_GRP_CH4	チャネル番号 4
	MTU_GRP_CH6	チャネル番号 6
	MTU_GRP_CH7	チャネル番号 7
	MTU_GRP_CH8	チャネル番号 8
補足	注 .　1 チャ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 で複数チャネルを指定した場合は、第一引数の値は無効となります。	

6.16.6　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6.12　cmd = MTU_CMD_GET_STATUS時（timer mode時）パラメーター覧

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

6.16.7　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6.13　cmd = MTU_CMD_GET_STATUS時（input capture mode時）パラメーター一覧

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

6.16.8　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6.14　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

7.　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。

8. 参考ドキュメント

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

  評価ボード RZ/T1 Evaluation Board ユーザーズマニュアル
  （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。）

- 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。）

- ユーザーズマニュアル：開発環境
  IAR 統合開発環境（IAR Embedded Workbench® for Arm）に関しては、IAR ホームページから入手してください。
  （最新版を IAR 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/

改訂記録	マルチファンクションタイマパルスユニット（MTU3a）アプリケーションノート

Rev.	発行日	改訂内容	
		ページ	ポイント
0.10	2015.03.16	—	初版発行
1.00	2015.04.10	—	Web掲載に際しRevのみ変更
1.10	2015.07.06	2. 動作環境	
		6	表2.1 動作環境　統合開発環境　表記一部修正、追加
		6. ソフトウェア説明	
		11	6.2.4 説明文 参照を追加
		11	表6.2 タイトルを一部追加
		12	表6.3 追加
		12	表6.4 追加
1.20	2015.12.03	2. 動作環境	
		6	表2.1 動作環境　統合開発環境　一部修正
1.30	2017.04.05	2. 動作環境	
		6	表2.1 動作環境　統合開発環境の内容変更
		6. ソフトウェア説明	
		—	6.2.4 必要メモリサイズ　削除
1.40	2018.06.07	2. 動作環境	
		6	表2.1 動作環境　統合開発環境の内容変更
		5. ハードウェア説明	
		9	図5.1 ハードウェア構成例 モジュール名変更
		8. 参考ドキュメント	
		77	IAR 統合開発環境名変更
1.50	2021.05.14	6. ソフトウェア説明	
		10	図6.1タイミング図　波形変更
		12	表6.4 サンプルプログラムで使用する定数　デューティ比の定数変更

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うと、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。