お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。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。
3. 当社製品を改造、改変、複製等しないでください。
4. 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。
5. 輸出に際しては、「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。本資料に記載されている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の目的で使用しないでください。また、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に使用することができません。
6. 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ものですが、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。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されてい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生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   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

   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、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社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。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、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、社会的損害など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、機器またはシステムとしての出荷保証をお願いいたします。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、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、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て、当社は、一切その責任を負いません。
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固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

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が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。

User's Manual

Renesas Embedded Application Programming Interface

ユーザーズマニュアル

SH/Tiny版

ルネサス エレクトロニクス
www.renesas.com

Rev.1.03　2008.08

本資料ご利用に際しての留意事項

1. 本資料は、お客様に用途に応じた適切な弊社製品をご購入いただくための参考資料であり、本資料中に記載の技術情報について弊社または第三者の知的財産権その他の権利の実施、使用を許諾または保証するものではありません。
2. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例など全ての情報の使用に起因する損害、第三者の知的財産権その他の権利に対する侵害に関し、弊社は責任を負いません。
3. 本資料に記載の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、あるいはその他軍事用途の目的で使用しないでください。また、輸出に際しては、「外国為替および外国貿易法」その他輸出関連法令を遵守し、それらの定めるところにより必要な手続を行ってください。
4. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例などの全ての情報は本資料発行時点のものであり、弊社は本資料に記載した製品または仕様等を予告なしに変更することがあります。弊社の半導体製品のご購入およびご使用に当たりましては、事前に弊社営業窓口で最新の情報をご確認いただきますとともに、弊社ホームページ（http://www.renesas.com）などを通じて公開される情報に常にご注意ください。
5. 本資料に記載した情報は、正確を期すため慎重に制作したものですが、万一本資料の記述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、弊社はその責任を負いません。
6. 本資料に記載の製品データ、図、表などに示す技術的な内容、プログラム、アルゴリズムその他応用回路例などの情報を流用する場合は、流用する情報を単独で評価するだけでなく、システム全体で十分に評価し、お客様の責任において適用可否を判断してください。弊社は、適用可否に対する責任は負いません。
7. 本資料に記載された製品は、各種安全装置や運輸・交通用、医療用、燃焼制御用、航空宇宙用、原子力、海底中継用の機器・システムなど、その故障や誤動作が直接人命を脅かしあるいは人体に危害を及ぼすおそれのあるような機器・システムや特に高度な品質・信頼性が要求される機器・システムでの使用を意図して設計、製造されたものではありません（弊社が自動車用と指定する製品を自動車に使用する場合を除きます）。これらの用途に利用されることをご検討の際には、必ず事前に弊社営業窓口へご照会ください。なお、上記用途に使用されたことにより発生した損害等について弊社はその責任を負いかねますのでご了承願います。
8. 第7項にかかわらず、本資料に記載された製品は、下記の用途には使用しないでください。これらの用途に使用されたことにより発生した損害等につきましては、弊社は一切の責任を負いません。
   1）生命維持装置。
   2）人体に埋め込み使用するもの。
   3）治療行為（患部切り出し、薬剤投与等）を行うもの。
   4）その他、直接人命に影響を与えるもの。
9. 本資料に記載された製品のご使用につき、特に最大定格、動作電源電圧範囲、放熱特性、実装条件およびその他諸条件につきましては、弊社保証範囲内でご使用ください。弊社保証値を越えて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、弊社はその責任を負いません。
10. 弊社は製品の品質および信頼性の向上に努めておりますが、特に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。弊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、社会的損害など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、誤動作防止設計などの安全設計（含むハードウェアおよびソフトウェア）およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。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、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
11. 本資料に記載の製品は、これを搭載した製品から剥がれた場合、幼児が口に入れて誤飲する等の事故の危険性があります。お客様の製品への実装後に容易に本製品が剥がれることがなきよう、お客様の責任において十分な安全設計をお願いします。お客様の製品から剥がれた場合の事故につきましては、弊社はその責任を負いません。
12. 本資料の全部または一部を弊社の文書による事前の承諾なしに転載または複製することを固くお断りいたします。
13. 本資料に関する詳細についてのお問い合わせ、その他お気付きの点等がございましたら弊社営業窓口までご照会ください。

D039444

目次

1. はじめに

Renesas Embedded Application Programming Interface（Renesas Embedded API）は、Renesas System Solutions 社製のマイクロコンピュータ用統合 API ライブラリです（以下、本ライブラリとする）。

2. ドライバ

2.1. 概要

本ライブラリは、マイクロコンピュータの周辺機能制御プログラム（ペリフェラルドライバ）を提供します。本ライブラリを使用することで、ユーザプログラムにペリフェラルドライバを組み込むことが可能になります。

2.2. 制御機能

本ライブラリは、ペリフェラルドライバとして以下に示す制御機能を内蔵しています。

(1) シリアル通信インタフェース制御機能

シリアル通信インタフェースドライバにより、シリアル通信条件の設定／クリア、および送受信するデータの制御・管理を行います。

(2) タイマ制御機能

タイマドライバにより、タイマ動作条件の設定／クリア、およびタイマの動作制御を行います。

(3) I/O ポート制御機能

I/O ポートドライバにより、I/O ポート使用条件の設定／クリア、およびデータのリード／ライト制御を行います。

(4) 外部割り込み制御機能

外部割り込みドライバにより、外部割り込み使用条件の設定／クリア、および割り込みの動作制御を行います。

(5) A/D コンバータ制御機能

A/D コンバータドライバにより、A/D コンバータ使用条件の設定／クリア、および A/D コンバータの動作制御を行います。

2.3. シリアル通信インタフェースドライバ

シリアル通信インタフェースドライバは、シリアル通信の設定／クリア、データの送受信、およびシリアル通信のステータス制御を行います。

2.4. タイマドライバ

タイマドライバは、タイマの設定／クリア、およびタイマ動作制御を行います。また以下のモード時において、カウンタ値を取得します。

- タイマモード
- イベントカウンタモード
- パルス幅変調モード（PWM モード）
- パルス周期測定モード
- パルス幅測定モード
- インプットキャプチャモード
- アウトプットコンペアモード

2.4.1. タイマモード

タイマモードは、内部で生成されたカウントソースをカウントするモードです。コンペアマッチ割り込み、またはオーバフロー割り込み発生時に、設定したコールバック関数を呼び出します。

2.4.2. イベントカウンタモード

イベントカウンタモードは、入力端子から入力される外部信号、または他のカウンタのオーバフロー／アンダフローをカウントするモードです。コンペアマッチ割り込み、インプットキャプチャ割り込み、またはオーバフロー／アンダフロー割り込み発生時に、設定したコールバック関数を呼び出します。

2.4.3. パルス幅変調モード（PWMモード）

パルス幅変調モード（PWM モード）は、任意の幅のパルスを連続して出力するモードです。コンペアマッチ割り込み発生時に、設定したコールバック関数を呼び出します。

2.4.4. パルス周期測定モード

パルス周期測定モードは、入力端子から入力される外部信号のパルス周期を測定するモードです。インプットキャプチャ割り込み、またはオーバフロー割り込み発生時に、設定したコールバック関数を呼び出します。

2.4.5. パルス幅測定モード

パルス幅測定モードは、入力端子から入力される外部信号のパルス幅を測定するモードです。インプットキャプチャ割り込み、またはオーバフロー割り込み発生時に、設定したコールバック関数を呼び出します。

2.4.6. インプットキャプチャモード

インプットキャプチャモードは、入力端子におけるアクティブ信号のエッジまたはクロックパルスでタイマ値をラッチすることにより、割り込み要求が発生するモードです。インプットキャプチャ割り込み、またはオーバフロー割り込み発生時に、設定したコールバック関数を呼び出します。

2.4.7. アウトプットコンペアモード

アウトプットコンペアモードは、タイマカウンタと比較値が一致したときに、割り込み要求が発生するモードです。コンペアマッチ割り込み発生時に、設定したコールバック関数を呼び出します。

2.5. I/Oポートドライバ

I/O ポートドライバは、I/O ポートの入出力設定、I/O ポートへのデータ書き込み、および I/O ポートからのデータ読み出しを行います。

2.6. 外部割り込みドライバ

外部割り込みドライバは、外部割り込みの設定・制御、および外部割り込みフラグの状態取得・クリアを行います。

2.7. A/Dコンバータ ドライバ

A/D コンバータ ドライバは、A/D コンバータの設定・制御・設定クリア、値の取得、および状態の取得・クリアを行います。

3. 標準型

表 3.1 に、本ライブラリが定義する標準型の内容について示します。設定値の詳細については、「4.2 API関数の解説」を参照して下さい。

表 3.1 標準型の内容

標準型	内容
Boolean	Boolean 型は、真（RAPI_TRUE（= 1））または偽（RAPI_FALSE（= 0））を示す enum 型のデータを表します。
VoidFuncNotify	VoidFuncNotify 型は、登録する通知関数の型を表します。

4. ライブラリ・リファレンス

4.1. 機能別API関数一覧

表 4.1 および 表 4.2 に、本ライブラリを周辺機能別に分類したAPI関数一覧表を示します。

表 4.1 Renesas Embedded API 関数一覧表（1/2）

周辺機能	No.	API 関数名	機能
シリアル通信インタフェース（SCI）	1	__CreateSCI	SCI の設定
	2	__DestroySCI	シリアルポートのクローズ
	3	__StartSCIReceiving	シリアル受信の開始
	4	__StartSCISending	シリアル送信の開始
	5	__StopSCIReceiving	シリアル受信の停止
	6	__StopSCISending	シリアル送信の停止
	7	__PollingSCIReceiving	シリアルポーリング受信
	8	__PollingSCISending	シリアルポーリング送信
	9	__GetSCIStatus	SCI のステータス取得
	10	__ClearSCIStatus	SCI のステータスクリア
	11	__OutputSCISck	SCK 端子の出力制御
	12	__OutputSCITxd	TXD 端子の出力制御
タイマ MTU2	1	__CreateTimer	タイマモードの設定
	2	__EnableTimer	タイマモードの動作制御
	3	__DestroyTimer	タイマモードの設定クリア
	4	__CreateEventCounter	イベントカウンタモードの設定
	5	__EnableEventCounter	イベントカウンタモードの動作制御
	6	__DestroyEventCounter	イベントカウンタモードの設定クリア
	7	__GetTimerCounter	タイマモード／イベントカウンタモードのカウンタ値取得
	8	__CreatePWM	パルス幅変調モードの設定
	9	__SetPWMPin	パルス幅変調用タイマジェネラルレジスタの設定
	10	__EnablePWM	パルス幅変調モードの動作制御
	11	__DestroyPWM	パルス幅変調モードの設定クリア
	12	__CreatePulsePeriodMeasurementMode	パルス周期測定モードの設定
	13	__SetInputPeriodPin	パルス周期測定用入力端子の設定
	14	__EnablePulsePeriodMeasurementMode	パルス周期測定モードの動作制御
	15	__DestroyPulsePeriodMeasurementMode	パルス周期測定モードの設定クリア
	16	__GetPulsePeriodMeasurementMode	パルス周期測定モードの測定値取得
	17	__CreatePulseWidthMeasurementMode	パルス幅測定モードの設定
	18	__SetInputWidthPin	パルス幅測定用入力端子の設定
	19	__EnablePulseWidthMeasurementMode	パルス幅測定モードの動作制御
	20	__DestroyPulseWidthMeasurementMode	パルス幅測定モードの設定クリア
	21	__GetPulseWidthMeasurementMode	パルス幅測定モードの測定値取得
	22	__CreateInputCapture	インプットキャプチャモードの設定
	23	__SetInputCapturePin	インプットキャプチャ用入力端子の設定
	24	__EnableInputCapture	インプットキャプチャモードの動作制御
	25	__DestroyInputCapture	インプットキャプチャモードの設定クリア
	26	__GetCaptureValue	インプットキャプチャモードのカウンタ値取得
	27	__CreateOutputCompare	アウトプットコンペアモードの設定
	28	__SetOutputPin	アウトプットコンペア用タイマジェネラルレジスタの設定
	29	__EnableOutputCompare	アウトプットコンペアモードの動作制御
	30	__DestroyOutputCompare	アウトプットコンペアモードの設定クリア
	31	__GetTimerFlag	タイマのステータス取得
	32	__ClearTimerFlag	タイマのステータスクリア

表 4.2　Renesas Embedded API 関数一覧表（2/2）

周辺機能	No.	API 関数名	機能
I/O ポート	1	__SetIOPort	I/O ポートの設定
	2	__ReadIOPort	I/O ポートからの読み出し
	3	__WriteIOPort	I/O ポートへの書き込み
外部割り込み	1	__CreateInterrupt	外部割り込みの設定
	2	__EnableInterrupt	外部割り込みの動作制御
	3	__GetInterruptAndPinInfo	外部割り込み入力端子のレベルおよび割り込み要求のステータス取得
	4	__ClearInterruptFlag	割り込み要求のステータスクリア
A/D コンバータ	1	__CreateADC	A/D コンバータの設定
	2	__EnableADC	A/D コンバータの動作制御
	3	__DestroyADC	A/D コンバータの設定クリア
	4	__GetADC	A/D 変換の結果取得
	5	__GetADCFlag	A/D コンバータのステータス取得
	6	__ClearADCFlag	A/D コンバータのステータスクリア

4.2. API関数の解説

本節では、API 関数の書式およびプログラム例について、以下の項目に分けて解説します。

概要	関数の処理内容を概説します。次に、関数の書式を示し、引数について簡単に説明します。
解説	関数の使い方を説明します。次に、設定可能なパラメータについて、引数ごとに“**[引数名]**”で区切って示します。
戻り値	関数の戻り値について説明します。
周辺機能	関数の周辺機能分類を示します。
参照	関連する関数を示します。
備考	関数を使用する際の注意事項を示します。
プログラム例	関数の使い方をプログラム例で示します。

4.2.1. シリアル通信インタフェース（SCI）

1) __CreateSCI

概要

＜SCI の設定＞

Boolean __CreateSCI(unsigned long data1, unsigned short data2)

data1	設定データ 1
data2	設定データ 2

解説（1/2）

指定したパラメータによって、SCI の設定を行います。

[data1]

data1 には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

●チャネルの指定：（複数指定可）

RAPI_COM1	SCI チャネル 0
RAPI_COM2	SCI チャネル 1
RAPI_COM3	SCI チャネル 2

●SCI の動作モード：（いずれか 1 つ）

RAPI_SM_SYNC	クロック同期式モード
RAPI_SM_ASYNC	調歩同期式モード

●調歩同期式モードのデータ長フォーマット：（いずれか 1 つ）

RAPI_7_BIT_LENGTH	データ長 7 ビット
RAPI_8_BIT_LENGTH	データ長 8 ビット

【注】クロック同期式モードで使用する場合、これらのパラメータは設定しないで下さい。

●シリアル／パラレル変換フォーマット：（いずれか 1 つ）

RAPI_LSB_SEL	LSB ファースト
RAPI_MSB_SEL	MSB ファースト

【注】調歩同期式モードのデータ長 7 ビットで使用する場合、これらのパラメータは
設定しないで下さい。

●SCI のクロックソース選択、および SCK 端子からのクロック出力の許可／禁止：（いずれか 1 つ）

RAPI_CKDIR_INT	内部クロック／SCK 端子は入力端子（CKE[1:0] = 00） （クロック同期式モードでは SCK 端子は同期クロック出力）
RAPI_CKDIR_EXT	外部クロック／SCK 端子はクロック入力（CKE[1:0] = 10） （調歩同期式モードではビットレートの 16 倍の周波数クロックを入力。クロック同期式モードでは同期クロックを入力）

●調歩同期式モードのストップビット長：（いずれか 1 つ）

RAPI_STPB_1	1 ストップビット
RAPI_STPB_2	2 ストップビット

【注】クロック同期式モードで使用する場合、これらのパラメータは設定しないで下さい。

●内蔵ボーレートジェネレータのクロックソース：（いずれか 1 つ）

RAPI_BCSS_P1	Pφクロック
RAPI_BCSS_P4	Pφ/4 クロック
RAPI_BCSS_P16	Pφ/16 クロック
RAPI_BCSS_P64	Pφ/64 クロック

解説（2/2）

●調歩同期式モードのパリティモード：（いずれか 1 つ）

RAPI_PARITY_NON	パリティビットなし
RAPI_PARITY_EVEN	偶数パリティビット
RAPI_PARITY_ODD	奇数パリティビット

【注】
- マルチプロセッサモードでは、これらの設定値は無効です。
- クロック同期式モードで使用する場合、これらのパラメータは設定しないで下さい。

●調歩同期式モードのマルチプロセッサ通信機能（使用許可／禁止）：（いずれか 1 つ）

RAPI_MULTI_ENA	マルチプロセッサモードを許可
RAPI_MULTI_DIS	マルチプロセッサモードを禁止

【注】クロック同期式モードで使用する場合、これらのパラメータは設定しないで下さい。

●調歩同期式モードのマルチプロセッサビット：（いずれか 1 つ）

RAPI_MULTIPRO_BIT_H	マルチプロセッサビットの値を 1 に指定
RAPI_MULTIPRO_BIT_L	マルチプロセッサビットの値を 0 に指定

【注】クロック同期式モードで使用する場合、これらのパラメータは設定しないで下さい。

●割り込みの要因：（複数指定可）

RAPI_INT_TX_ENA	送信データエンプティ割り込みを許可
RAPI_INT_TEND_ENA	送信終了割り込みを許可
RAPI_INT_RX_ERR_ENA	受信データフルおよび受信エラー割り込みを許可
RAPI_INT_ERR_ENA	受信エラー割り込みを許可
RAPI_INT_MULTI_ENA	マルチプロセッサ割り込みを許可

●送受信割り込み優先レベル：（いずれか 1 つ）

RAPI_SCI_INT_LV_0	割り込み優先レベル 0
RAPI_SCI_INT_LV_1	割り込み優先レベル 1
RAPI_SCI_INT_LV_2	割り込み優先レベル 2
RAPI_SCI_INT_LV_3	割り込み優先レベル 3
RAPI_SCI_INT_LV_4	割り込み優先レベル 4
RAPI_SCI_INT_LV_5	割り込み優先レベル 5
RAPI_SCI_INT_LV_6	割り込み優先レベル 6
RAPI_SCI_INT_LV_7	割り込み優先レベル 7
RAPI_SCI_INT_LV_8	割り込み優先レベル 8
RAPI_SCI_INT_LV_9	割り込み優先レベル 9
RAPI_SCI_INT_LV_10	割り込み優先レベル 10
RAPI_SCI_INT_LV_11	割り込み優先レベル 11
RAPI_SCI_INT_LV_12	割り込み優先レベル 12
RAPI_SCI_INT_LV_13	割り込み優先レベル 13
RAPI_SCI_INT_LV_14	割り込み優先レベル 14
RAPI_SCI_INT_LV_15	割り込み優先レベル 15

[data2]

data2 には、ボーレートジェネレータのビットレートレジスタ（SCBRR）の値（N；0≦N≦255）を設定して下さい。

戻り値

SCI の設定に成功した場合は RAPI_TRUE を返し、設定に失敗した場合は RAPI_FALSE を返します。

周辺機能

SCI

参照

なし

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include ”rapi_sif_sh_7125.h”

void func( void )
{
   /* SCI チャネル 0 をクロック同期式モードに設定 */
   __CreateSCI( RAPI_COM1 | RAPI_SM_SYNC | RAPI_CKDIR_INT |
          RAPI_BCSS_P1 | RAPI_LSB_SEL | RAPI_INT_RX_ERR_ENA |
          RAPI_INT_LV_6, 20 );
}

/* または */

void func( void )
{
   /* SCI チャネル 0 を調歩同期式モードに設定 */
   __CreateSCI( RAPI_COM1 | RAPI_SM_ASYNC | RAPI_8_BIT_LENGTH |
          RAPI_LSB_SEL | RAPI_STPB_1 | RAPI_CKDIR_INT |
          RAPI_BCSS_P1 | RAPI_PARITY_NON | RAPI_MULTI_DIS |
          RAPI_INT_RX_ERR_ENA | RAPI_SCI_INT_LV_6, 20 );
}
```

2) __DestroySCI

概要

＜シリアルポートのクローズ＞

Boolean __DestroySCI(unsigned long data)

data	設定データ

解説

指定したシリアルポートへのクロック供給を停止します。

[data]

data には、チャネルの指定として次のいずれかのパラメータを設定して下さい。

RAPI_COM1	SCI チャネル 0
RAPI_COM2	SCI チャネル 1
RAPI_COM3	SCI チャネル 2

戻り値

シリアルポートの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

SCI

参照

__CreateSCI

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include ”rapi_sif_sh_7125.h”

void func( void )
{
    /* SCI チャネル 0 のシリアルポートをクローズ */
    __DestroySCI( RAPI_COM1 );
}
```

3) __StartSCIReceiving

概要

＜シリアル受信の開始＞

Boolean __StartSCIReceiving(unsigned long data, unsigned char *RcvDtBuf, unsigned short byteNum, unsigned short *RcvNum)

data	設定データ
RcvDtBuf	受信データを格納するバッファへのポインタ
byteNum	受信バイト数
RcvNum	受信したデータ数を格納するアドレスへのポインタ

解説

シリアル受信を開始し、受信したデータを指定したバイト数分だけ取得します。

[data]

data には、チャネルの指定として次のいずれかのパラメータを設定して下さい。

RAPI_COM1	SCI チャネル 0
RAPI_COM2	SCI チャネル 1
RAPI_COM3	SCI チャネル 2

戻り値

シリアル受信開始に成功した場合はRAPI_TRUE を返し、失敗した場合はRAPI_FALSE を返します。

周辺機能

SCI

参照

__CreateSCI，__StopSCIReceiving，__GetSCIStatus

備考

- __CreateSCI関数を実行後、少なくとも 1 ビット期間待ってから本API関数を実行して下さい。
- 第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。
- 次の値が受信バッファに格納されます。
  - 上位 8 ビット：シリアルステータスレジスタ（SCSSR）から読み込んだ値
  - 下位 8 ビット：レシーブデータレジスタ（SCRDR）から読み込んだ値

  【注】受信エラー発生時は読み込みません。

プログラム例

```
#include ”rapi_sif_sh_7125.h”

unsigned char ReceiveBuf[10];
unsigned short ReceiveNum;

void func( void )
{
    /* SCI チャネル 0 のシリアル受信データを 5 バイト分取得 */
    __StartSCIReceiving( RAPI_COM1, ReceiveBuf, 5, &ReceiveNum );
}
```

4) __StartSCISending

概要

＜シリアル送信の開始＞

Boolean __StartSCISending(unsigned long data, unsigned char *SndDtBuf, unsigned short byteNum, unsigned short *SndNum)

data	設定データ
SndDtBuf	送信データへのポインタ
byteNum	送信バイト数
SndNum	送信したデータ数を格納するバッファへのポインタ

解説

シリアル送信を開始し、指定したバイト数分だけデータを送信します。

[data]

data には、チャネルの指定として次のいずれかのパラメータを設定して下さい。

RAPI_COM1	SCI チャネル 0
RAPI_COM2	SCI チャネル 1
RAPI_COM3	SCI チャネル 2

戻り値

シリアル送信開始に成功した場合はRAPI_TRUE を返し、失敗した場合はRAPI_FALSE を返します。

周辺機能

SCI

参照

__CreateSCI，__StopSCISending，__GetSCIStatus

備考

- __CreateSCI関数実行後、少なくとも 1 ビット期間待ってから本API関数を実行して下さい。
- 第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include ”rapi_sif_sh_7125.h”

unsigned char SendBuf[10];
unsigned short SendNum;

void func( void )
{
    /* 5 バイトのデータをシリアル送信開始 */
    __StartSCISending( RAPI_COM1, SendBuf, 5, &SendNum );
}
```

5) __StopSCIReceiving

概要

＜シリアル受信の停止＞

Boolean __StopSCIReceiving(unsigned long data1, unsigned short data2)

data1	設定データ 1
data2	設定データ 2

解説

シリアル受信を停止します。

[data1]

data1 には、チャネルの指定として次のいずれかのパラメータを設定して下さい。

RAPI_COM1	SCI チャネル 0
RAPI_COM2	SCI チャネル 1
RAPI_COM3	SCI チャネル 2

[data2]

data2 には、シリアル受信停止までのウェイト時間を設定して下さい。

戻り値

シリアル受信停止に成功し、なおかつ受信エラーが発生していない場合は RAPI_TRUE を返します。
それ以外の場合は RAPI_FALSE を返します。

周辺機能

SCI

参照

__StartSCIReceiving

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include ”rapi_sif_sh_7125.h”

void func( void )
{
    /* SCI チャネル 0 のシリアル受信を停止 */
    __StopSCIReceiving( RAPI_COM1, 50000 );
}
```

6) __StopSCISending

概要

＜シリアル送信の停止＞

Boolean __StopSCISending(unsigned long data1, unsigned short data2)

data1	設定データ 1
data2	設定データ 2

解説

シリアル送信を停止します。

[data1]

data1 には、チャネルの指定として次のいずれかのパラメータを設定して下さい。

RAPI_COM1	SCI チャネル 0
RAPI_COM2	SCI チャネル 1
RAPI_COM3	SCI チャネル 2

[data2]

data2 には、シリアル送信停止までのウェイト時間を設定して下さい。

戻り値

シリアル送信停止に成功した場合はRAPI_TRUE を返し､失敗した場合はRAPI_FALSE を返します。

周辺機能

SCI

参照

__StartSCISending

備考

- クロック同期式モード設定時は、シリアルデータ受信も停止します。
- 第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include ”rapi_sif_sh_7125.h”

void func( void )
{
    /* SCI チャネル 0 のシリアル送信を停止 */
    __StopSCISending( RAPI_COM1, 50000 );
}
```

7) __PollingSCIReceiving

概要

＜シリアルポーリング受信＞

Boolean __PollingSCIReceiving(unsigned long data)

data	設定データ

解説

ポーリングを用いたシリアル受信を行います。__StartSCIReceivingで指定した受信データ数分のみ取得します。

[data]

data には、チャネルの指定として次のいずれかのパラメータを設定して下さい。

RAPI_COM1	SCI チャネル 0
RAPI_COM2	SCI チャネル 1
RAPI_COM3	SCI チャネル 2

戻り値

シリアルポートの指定が正しくない場合または受信データが間違っている場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

SCI

参照

__StartSCIReceiving，__GetSCIStatus

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include ”rapi_sif_sh_7125.h”

unsigned char ReceiveBuf[10];
unsigned short ReceiveNum;

void INT_SCI0_RX( void );
void ErrorOpe( void );

/* メイン処理 */
void func( void )
{
    /* SCI チャネル 0 をクロック同期式モードに設定
        （受信データフルおよび受信エラー割り込みを許可） */
    __CreateSCI( RAPI_COM1 | RAPI_SM_SYNC | RAPI_CKDIR_INT |
           RAPI_BCSS_P1 | RAPI_LSB_SEL | RAPI_INT_RX_ERR_ENA |
           RAPI_SCI_INT_LV_6, 20 );

    /* 受信データ量を 5 バイトに設定し、シリアル受信開始 */
    __StartSCIReceiving( RAPI_COM1, ReceiveBuf, 5, &ReceiveNum );
}

/* 割り込み処理 */
void INT_SCI0_RX( void )
{
    /* 5 バイトのデータ受信 */
    if( __PollingSCIReceiving( RAPI_COM1 ) == RAPI_TRUE ){   /* 受信成功 */
        if( ReceiveNum == 5 ){
            /* 5 バイトのデータ受信終了 */
            __StopSCIReceiving( RAPI_COM1 );
        }
    }
    else{    /* 受信失敗 */
        ErrorOpe();    /* エラー処理 */
    }
}
```

8) __PollingSCISending

概要

＜シリアルポーリング送信＞

Boolean __PollingSCISending(unsigned long data)

data	設定データ

解説

ポーリングを用いたシリアル送信を行います。__StartSCISendingで指定した送信用バッファ内のデータ数分のみ送信します。

[data]

data には、チャネルの指定として次のいずれかのパラメータを設定して下さい。

RAPI_COM1	SCI チャネル 0
RAPI_COM2	SCI チャネル 1
RAPI_COM3	SCI チャネル 2

戻り値

シリアルポートの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

SCI

参照

__StartSCISending

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include ”rapi_sif_sh_7125.h”

unsigned char SendBuf[10];
unsigned short SendNum;

void INT_SCI0_TX( void );

/* メイン処理 */
void func( void )
{
    /* SCI チャネル 0 をクロック同期式モードに設定
        （送信データエンプティ割り込みを許可） */
    __CreateSCI( RAPI_COM1 | RAPI_SM_SYNC | RAPI_CKDIR_INT |
           RAPI_BCSS_P1 | RAPI_LSB_SEL | RAPI_INT_TX_ENA |
           RAPI_SCI_INT_LV_6, 20 );

    /* 送信データ量を 5 バイトに設定し、シリアル送信開始 */
    __StartSCISending( RAPI_COM1, SendBuf, 5, &SendNum );
}

/* 割り込み処理 */
void INT_SCI0_TX( void )
{
    if( SendNum == 5 ){
        /* 5 バイトのデータ送信終了 */
        __StopSCIsending( RAPI_COM1, 5000 );
    }
    else{
        /* 5 バイトのデータ送信 */
        __PollingSCISending( RAPI_COM1 );
    }
}
```

9) __GetSCIStatus

概要

＜SCI のステータス取得＞

Boolean __GetSCIStatus(unsigned long data, unsigned char *status)

data	設定データ
status	受信エラーフラグを格納するために使用するバイトアドレス

解説

SCI の各種送受信ステータスを取得します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを選択する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_COM1	SCI チャネル 0
RAPI_COM2	SCI チャネル 1
RAPI_COM3	SCI チャネル 2
RAPI_TDRE	送信データレジスタエンプティフラグ
RAPI_RDRF	受信データレジスタフルフラグ
RAPI_ORER	オーバランエラーフラグ
RAPI_FER	フレーミングエラーフラグ
RAPI_PER	パリティエラーフラグ
RAPI_TEND	送信終了フラグ
RAPI_MPB	受信用マルチプロセッサビットフラグ
RAPI_MPBT	送信用マルチプロセッサビットフラグ
RAPI_RECV_ERROR	すべての受信エラーフラグ （オーバラン、フレーミング、およびパリティエラーフラグ）
RAPI_ALL_FLAG	SCI のすべてのステータスフラグ

戻り値

シリアルポートの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

SCI

参照

__ClearSCIStatus

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include ”rapi_sif_sh_7125.h”

unsigned char statusvalue;

void func( void )
{
    /* SCI チャネル 0 のパリティエラーフラグ取得 */
    return __GetSCIStatus( RAPI_COM1 | RAPI_PER, &statusvalue );
}
```

10) __ClearSCIStatus

概要

＜SCI のステータスクリア＞

Boolean __ClearSCIStatus(unsigned long data)

data	設定データ

解説

SCI の各種送受信ステータスをクリアします。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを選択する場合は、パラメータ間に “|” 記号を入れて下さい。

RAPI_COM1	SCI チャネル 0
RAPI_COM2	SCI チャネル 1
RAPI_COM3	SCI チャネル 2
RAPI_TDRE	送信データレジスタエンプティフラグ
RAPI_RDRF	受信データレジスタフルフラグ
RAPI_ORER	オーバランエラーフラグ
RAPI_FER	フレーミングエラーフラグ
RAPI_PER	パリティエラーフラグ
RAPI_TEND	送信終了フラグ
RAPI_MPB	受信用マルチプロセッサビットフラグ
RAPI_MPBT	送信用マルチプロセッサビットフラグ
RAPI_RECV_ERROR	すべての受信エラーフラグ （オーバラン、フレーミング、およびパリティエラーフラグ）
RAPI_ALL_FLAG	SCI のすべてのステータスフラグ

戻り値

シリアルポートの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

SCI

参照

__GetSCIStatus

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include ”rapi_sif_sh_7125.h”

void func( void )
{
    /* SCI チャネル 0 のパリティエラーフラグクリア */
    return __ClearSCIStatus( RAPI_COM1 | RAPI_PER );
}
```

11) __OutputSCISck

概要

＜SCK 端子の出力制御＞

Boolean __OutputSCISck(unsigned long data)

data	設定データ

解説

調歩同期式モード時に、SCK 端子の出力を制御します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に “|” 記号を入れて下さい。

RAPI_COM1	SCI チャネル 0
RAPI_COM2	SCI チャネル 1
RAPI_COM3	SCI チャネル 2
RAPI_SCK_OUTPUT	ビットレートの 16 倍の周波数クロックを出力
RAPI_SCK_NO_OUTPUT	SCK 端子にシリアルポートレジスタ（SCSPTR）の SPB1DT ビット値を出力しない
RAPI_SCK_OUTPUT_L	SCK 端子にシリアルポートレジスタ（SCSPTR）の SPB1DT ビット値をローレベルで出力
RAPI_SCK_OUTPUT_H	SCK 端子にシリアルポートレジスタ（SCSPTR）の SPB1DT ビット値をハイレベルで出力

戻り値

シリアルポートの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

SCI

参照

__CreateSCI

備考

- SCK端子をポート出力端子として使用する場合は、__CreateSCIでクロックソースを内部クロック（RAPI_CKDIR_INT）に設定して下さい。
- 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include ”rapi_sif_sh_7125.h”

void func( void )
{
    /* チャネル 0 の SCK 端子への出力データをハイレベルに指定 */
    return __OutputSCISck( RAPI_COM1 | RAPI_SCK_OUTPUT_H );
}
```

12) __OutputSCITxd

概要

＜TXD 端子の出力制御＞

Boolean __OutputSCITxd(unsigned long data)

data	設定データ

解説

調歩同期式モード時に、TXD 端子の出力を制御します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_COM1	SCI チャネル 0
RAPI_COM2	SCI チャネル 1
RAPI_COM3	SCI チャネル 2
RAPI_TXD_BREAK_L	TXD 端子をローレベル出力に設定
RAPI_TXD_BREAK_H	TXD 端子をハイレベル出力に設定

戻り値

シリアルポートの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

SCI

参照

なし

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include ”rapi_sif_sh_7125.h”

void func( void )
{
    /* チャネル 0 の TXD 端子をハイレベル出力に設定 */
    return __OutputSCITxd( RAPI_COM1 | RAPI_TXD_BREAK_H );
}
```

4.2.2. タイマMTU2

1) __CreateTimer

概要

＜タイマモードの設定＞

Boolean __CreateTimer(unsigned long data1, unsigned short data2, void *func)

data1	設定データ 1
data2	設定データ 2
func	コールバック関数ポインタ

解説（1/7）

指定したタイマをタイマモードに設定します。

[data1]

data1 には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_CMT_0	CMT チャネル 0
RAPI_CMT_1	CMT チャネル 1
RAPI_FREE_RUNNING	フリーランニングカウント動作
RAPI_PERIODIC	周期カウント動作
RAPI_MP1	内部クロック：MPφ/1 でカウント
RAPI_MP4	内部クロック：MPφ/4 でカウント
RAPI_MP16	内部クロック：MPφ/16 でカウント
RAPI_MP64	内部クロック：MPφ/64 でカウント
RAPI_MP256_1	内部クロック：MPφ/256 でカウント（チャネル 1 の場合）
RAPI_MP256_34	内部クロック：MPφ/256 でカウント （チャネル 3 または 4 の場合）
RAPI_MP1024_2	内部クロック：MPφ/1024 でカウント（チャネル 2 の場合）
RAPI_MP1024_34	内部クロック：MPφ/1024 でカウント （チャネル 3 または 4 の場合）
RAPI_P8	内部クロック：Pφ/8 でカウント
RAPI_P32	内部クロック：Pφ/32 でカウント
RAPI_P128	内部クロック：Pφ/128 でカウント
RAPI_P512	内部クロック：Pφ/512 でカウント
RAPI_RISING_EDGE	立ち上がりエッジでカウント
RAPI_FALLING_EDGE	立ち下がりエッジでカウント
RAPI_BOTH_EDGE	両エッジでカウント
【注】カウントソースに MPφ/1 を選択した場合、カウントエッジは 立ち上がりエッジのみの動作となります。	

解説（2/7）

RAPI_TCNT_CLEAR_DIS	TCNT のクリアを禁止 （チャネル 0 の TGRE，TGRF のみ使用可）
RAPI_TCNT_CLEAR_TGRA	TGRA コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRB	TGRB コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRC	TGRC コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRD	TGRD コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_DIS_TGRE	TGRE レジスタを使用、ただし TGRE コンペアマッチで TCNT のクリアを禁止
RAPI_TCNT_CLEAR_DIS_TGRF	TGRF レジスタを使用するが、TGRF コンペアマッチで TCNT のクリアを禁止
RAPI_TCNT_CLEAR_TGRU	TGRU_5 コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRV	TGRV_5 コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRW	TGRW_5 コンペアマッチで TCNT をクリア
RAPI_OUTPUT_RETAIN	出力保持
RAPI_OUTPUT_0_0	初期出力 0，コンペアマッチで 0 を出力
RAPI_OUTPUT_0_1	初期出力 0，コンペアマッチで 1 を出力
RAPI_OUTPUT_0_TOG	初期出力 0，コンペアマッチでトグル出力
RAPI_OUTPUT_1_0	初期出力 1，コンペアマッチで 0 を出力
RAPI_OUTPUT_1_1	初期出力 1，コンペアマッチで 1 を出力
RAPI_OUTPUT_1_TOG	初期出力 1，コンペアマッチでトグル出力
RAPI_OVERFLOW_ENA	オーバフロー割り込みを許可
RAPI_OVERFLOW_DIS	オーバフロー割り込みを禁止
RAPI_COMPARE_MATCH_ENA	コンペアマッチ割り込みを許可
RAPI_COMPARE_MATCH_DIS	コンペアマッチ割り込みを禁止
RAPI_AD_START_REQ	TGRA インプットキャプチャ／コンペアマッチで A/D コンバータ開始要求の生成を許可
RAPI_NO_AD_START_REQ	TGRA インプットキャプチャ／コンペアマッチで A/D コンバータ開始要求の生成を禁止
RAPI_TIMER_INT_LV_0	割り込み優先レベル 0
RAPI_TIMER_INT_LV_1	割り込み優先レベル 1
RAPI_TIMER_INT_LV_2	割り込み優先レベル 2
RAPI_TIMER_INT_LV_3	割り込み優先レベル 3
RAPI_TIMER_INT_LV_4	割り込み優先レベル 4
RAPI_TIMER_INT_LV_5	割り込み優先レベル 5
RAPI_TIMER_INT_LV_6	割り込み優先レベル 6
RAPI_TIMER_INT_LV_7	割り込み優先レベル 7
RAPI_TIMER_INT_LV_8	割り込み優先レベル 8
RAPI_TIMER_INT_LV_9	割り込み優先レベル 9
RAPI_TIMER_INT_LV_10	割り込み優先レベル 10
RAPI_TIMER_INT_LV_11	割り込み優先レベル 11
RAPI_TIMER_INT_LV_12	割り込み優先レベル 12
RAPI_TIMER_INT_LV_13	割り込み優先レベル 13
RAPI_TIMER_INT_LV_14	割り込み優先レベル 14
RAPI_TIMER_INT_LV_15	割り込み優先レベル 15

解説（3/7）

●RAPI_MTU2_0 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，RAPI_MP64 } から 1 つを指定します。初期値は RAPI_MP1 です。

（動作方式） { RAPI_FREE_RUNNING，RAPI_PERIODIC } から 1 つを指定します。初期値は RAPI_FREE_RUNNING です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP1 を指定した場合、カウントエッジは初期値のみの動作となります。

• RAPI_FREE_RUNNING 設定時は、次のパラメータを選択できます：

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

• RAPI_PERIODIC 設定時は、次のパラメータを選択できます：

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_TGRC，RAPI_TCNT_CLEAR_TGRD，RAPI_TCNT_CLEAR_DIS_TGRE，RAPI_TCNT_CLEAR_DIS_TGRF } から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（カウント出力） { RAPI_OUTPUT_RETAIN，RAPI_OUTPUT_0_0，RAPI_OUTPUT_0_1，RAPI_OUTPUT_0_TOG，RAPI_OUTPUT_1_0，RAPI_OUTPUT_1_1，RAPI_OUTPUT_1_TOG } から 1 つを指定します。
初期値は RAPI_OUTPUT_RETAIN です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA，RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。
初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つ指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（4/7）

●RAPI_MTU2_1 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，
RAPI_MP64，RAPI_MP256_1 } から 1 つを指定します。
初期値は RAPI_MP1 です。

（動作方式） { RAPI_FREE_RUNNING，RAPI_PERIODIC } から 1 つを
指定します。初期値は RAPI_FREE_RUNNING です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，
RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP1 を指定した場合、
カウントエッジは初期値のみの動作となります。

• RAPI_FREE_RUNNING 設定時は、次のパラメータを選択できます：

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から
1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，
RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，
RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，
RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，
RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，
RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，
RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，
RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から
1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

• RAPI_PERIODIC 設定時は、次のパラメータを選択できます：

（カウントクリアソース） { RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB }
から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

（カウント出力） { RAPI_OUTPUT_RETAIN，RAPI_OUTPUT_0_0，
RAPI_OUTPUT_0_1，RAPI_OUTPUT_0_TOG，
RAPI_OUTPUT_1_0，RAPI_OUTPUT_1_1，
RAPI_OUTPUT_1_TOG } から 1 つを指定します。
初期値は RAPI_OUTPUT_RETAIN です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から
1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA，
RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。
初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，
RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，
RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，
RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，
RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，
RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，
RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，
RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から
1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（5/7）

●RAPI_MTU2_2 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，
RAPI_MP64，RAPI_MP1024_2 } から 1 つを指定します。
初期値は RAPI_MP1 です。

（動作方式） { RAPI_FREE_RUNNING，RAPI_PERIODIC } から 1 つを
指定します。初期値は RAPI_FREE_RUNNING です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，
RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP1 を指定した場合、
カウントエッジは初期値のみの動作となります。

• RAPI_FREE_RUNNING 設定時は、次のパラメータを選択できます：

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から
1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，
RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，
RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，
RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，
RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，
RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，
RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，
RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から
1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

• RAPI_PERIODIC 設定時は、次のパラメータを選択できます：

（カウントクリアソース） { RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB }
から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

（カウント出力） { RAPI_OUTPUT_RETAIN，RAPI_OUTPUT_0_0，
RAPI_OUTPUT_0_1，RAPI_OUTPUT_0_TOG，
RAPI_OUTPUT_1_0，RAPI_OUTPUT_1_1，
RAPI_OUTPUT_1_TOG } から 1 つを指定します。
初期値は RAPI_OUTPUT_RETAIN です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から
1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA，
RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。
初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，
RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，
RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，
RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，
RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，
RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，
RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，
RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から
1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（6/7）

●**RAPI_MTU2_3 または RAPI_MTU2_4 選択時に指定可能なパラメータ**：

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，RAPI_MP64，RAPI_MP256_34，RAPI_MP1024_34 } から 1 つを指定します。初期値は RAPI_MP1 です。

（動作方式） { RAPI_FREE_RUNNING，RAPI_PERIODIC } から 1 つを指定します。初期値は RAPI_FREE_RUNNING です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP1 を指定した場合、カウントエッジは初期値のみの動作となります。

• **RAPI_FREE_RUNNING 設定時は、次のパラメータを選択できます**：

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

• **RAPI_PERIODIC 設定時は、次のパラメータを選択できます**：

（カウントクリアソース） { RAPI_ TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB，RAPI_ TCNT_CLEAR_TGRC，RAPI_TCNT_CLEAR_TGRD } から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（カウント出力） { RAPI_OUTPUT_RETAIN，RAPI_OUTPUT_0_0，RAPI_OUTPUT_0_1，RAPI_OUTPUT_0_TOG，RAPI_OUTPUT_1_0，RAPI_OUTPUT_1_1，RAPI_OUTPUT_1_TOG } から 1 つを指定します。
初期値は RAPI_OUTPUT_RETAIN です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA，RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。
初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（7/7）

●RAPI_MTU2_5 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，RAPI_MP64 } から 1 つを指定します。初期値は RAPI_MP1 です。

（動作方式） RAPI_PERIODIC のみ指定可能です。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRU，RAPI_TCNT_CLEAR_TGRV，RAPI_TCNT_CLEAR_TGRW } から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA，
RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。
初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，
RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，
RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，
RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，
RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，
RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，
RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，
RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

●RAPI_CMT_0 または RAPI_CMT_1 選択時に指定可能なパラメータ：

（チャネル） { RAPI_CMT_0，RAPI_CMT_1 } から 1 つを指定します。

（カウントソース） { RAPI_P8，RAPI_P32，RAPI_P128，RAPI_P512 } から 1 つを指定します。初期値は RAPI_P8 です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA，
RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。
初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，
RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，
RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，
RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，
RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，
RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，
RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，
RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

[data2]
data2 には、タイマジェネラルレジスタ（TGR）の設定値を 16 ビット単位で設定して下さい。

[func]
func には、コールバック関数を指すポインタを指定して下さい。コールバック関数を指定しない場合は、RAPI_NULL を指定してください。

戻り値

タイマの指定が正しくない場合はRAPI_FALSEを返し、それ以外の場合はRAPI_TRUEを返します。

周辺機能

タイマ MTU2（タイマモード）

参照

__EnableTimer，__DestroyTimer

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

/* コールバック関数宣言 */
void TimerIntFunc( void );

void func( void )
{
   /* MTU2 チャネル 0 をタイマモードに設定 */
   __CreateTimer( RAPI_MTU2_0 | RAPI_MP16 | RAPI_PERIODIC |
          RAPI_RISING_EDGE | RAPI_TCNT_CLEAR_TGRA |
          RAPI_OUTPUT_0_TOG | RAPI_NO_AD_START_REQ |
          RAPI_COMPARE_MATCH_ENA | RAPI_TIMER_INT_LV_5,
          0x8000, TimerIntFunc );
}
```

2) __EnableTimer

概要

＜タイマモードの動作制御＞

Boolean __EnableTimer(unsigned long data)

data	設定データ

解説

タイマモードに設定されたタイマの動作開始／停止を制御します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、複数のタイマを同時に動作設定できます。（下記注意事項参照）

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU20_4	MTU2 チャネル 0～4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_MTU2_5	MTU2 チャネル 5U，5V，5W
RAPI_CMT_0	CMT チャネル 0
RAPI_CMT_1	CMT チャネル 1
RAPI_CMT_ALL	CMT のすべてのチャネル
RAPI_TIMER_ON	タイマモードに設定されているタイマを動作開始に設定
RAPI_TIMER_OFF	タイマモードに設定されているタイマを動作停止に設定

【注】 • RAPI_MTU2_0～4 において、複数のタイマを同時設定できます。
• RAPI_MTU2_5U，5V，および 5W において、複数のタイマを同時設定できます。
• RAPI_CMT_0，RAPI_CMT_1 において、複数のタイマを同時設定できます。

戻り値

タイマの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（タイマモード）

参照

__CreateTimer，__DestroyTimer

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* MTU2 チャネル 0 およびチャネル 1 をタイマモードで動作開始 */
    __EnableTimer( RAPI_MTU2_0 | RAPI_MTU2_1 | RAPI_TIMER_ON );
}
```

3) __DestroyTimer

概要

＜タイマモードの設定クリア＞

Boolean __DestroyTimer(unsigned long data)

data	設定データ

解説

タイマモードに設定されたタイマの設定をクリアします。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数のタイマの設定をクリアできます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU20_4	MTU2 チャネル 0～4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_MTU2_5	MTU2 チャネル 5U，5V，5W
RAPI_MTU2_ALL	MTU2 のすべてのチャネル
RAPI_CMT_0	CMT チャネル 0
RAPI_CMT_1	CMT チャネル 1
RAPI_CMT_ALL	CMT のすべてのチャネル
RAPI_TIMER_ALL	MTU2 および CMT のすべてのチャネル

戻り値

タイマの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（タイマモード）

参照

__CreateTimer，__EnableTimer

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* MTU2 チャネル 2 およびチャネル 3 に対し、タイマモードの設定クリア */
    __DestroyTimer( RAPI_MTU2_2 | RAPI_MTU2_3 );
}
```

4) __CreateEventCounter

概要

＜イベントカウンタモードの設定＞

Boolean __CreateEventCounter(unsigned long data1, unsigned short data2, void *func1, void *func2)

data1	設定データ 1
data2	設定データ 2
func1	コンペアマッチ割り込みまたはオーバフロー割り込み用のコールバック関数
func2	アンダフロー割り込み用のコールバック関数

解説（1/7）

指定したタイマをイベントカウンタモードに設定します。

[data1]

data1 には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_EXTER_TCLKA_0	外部クロック：TCLKA 端子入力でカウント （チャネル 0～2 の場合）
RAPI_EXTER_TCLKB_0	外部クロック：TCLKB 端子入力でカウント （チャネル 0～2 の場合）
RAPI_EXTER_TCLKC_0	外部クロック：TCLKC 端子入力でカウント （チャネル 0 または 2 の場合）
RAPI_EXTER_TCLKD_0	外部クロック：TCLKD 端子入力でカウント （チャネル 0 の場合）
RAPI_EXTER_TCLKA_3	外部クロック：TCLKA 端子入力でカウント （チャネル 3 または 4 の場合）
RAPI_EXTER_TCLKB_3	外部クロック：TCLKB 端子入力でカウント （チャネル 3 または 4 の場合）
RAPI_TCNT2_FLOW	TCNT_2 のオーバフロー／アンダフローでカウント
RAPI_NORMAL	通常動作
RAPI_PHASE_COUNTING_1	位相計数モード 1（チャネル 1 または 2 のみ対応）
RAPI_PHASE_COUNTING_2	位相計数モード 2（チャネル 1 または 2 のみ対応）
RAPI_PHASE_COUNTING_3	位相計数モード 3（チャネル 1 または 2 のみ対応）
RAPI_PHASE_COUNTING_4	位相計数モード 4（チャネル 1 または 2 のみ対応）
RAPI_FREE_RUNNING	フリーランニングカウント動作
RAPI_PERIODIC	周期カウント動作
RAPI_RISING_EDGE	立ち上がりエッジでカウント
RAPI_FALLING_EDGE	立ち下がりエッジでカウント
RAPI_BOTH_EDGE	両エッジでカウント
【注】カウントソースに TCNT_2 のオーバフロー／アンダフローを選択した場合、 カウントエッジは立ち上がりエッジのみの動作となります。	
RAPI_TCNT_CLEAR_DIS	TCNT のクリアを禁止 （チャネル 0 の TGRE および TGRF のみ使用可）
RAPI_TCNT_CLEAR_TGRA	TGRA コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRB	TGRB コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRC	TGRC コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRD	TGRD コンペアマッチで TCNT をクリア

解説（2/7）

RAPI_OUTPUT_RETAIN	出力保持
RAPI_OUTPUT_0_0	初期出力 0，コンペアマッチで 0 を出力
RAPI_OUTPUT_0_1	初期出力 0，コンペアマッチで 1 を出力
RAPI_OUTPUT_0_TOG	初期出力 0，コンペアマッチでトグル出力
RAPI_OUTPUT_1_0	初期出力 1，コンペアマッチで 0 を出力
RAPI_OUTPUT_1_1	初期出力 1，コンペアマッチで 1 を出力
RAPI_OUTPUT_1_TOG	初期出力 1，コンペアマッチでトグル出力
RAPI_OVERFLOW_ENA	オーバフロー割り込みを許可
RAPI_OVERFLOW_DIS	オーバフロー割り込みを禁止
RAPI_COMPARE_MATCH_ENA	コンペアマッチ割り込みを許可
RAPI_COMPARE_MATCH_DIS	コンペアマッチ割り込みを禁止
RAPI_UNDERFLOW_ENA	アンダフロー割り込みを許可 （チャネル 1 または 2 が位相計数モードに 設定されている場合のみ選択可）
RAPI_UNDERFLOW_DIS	アンダフロー割り込みを禁止 （チャネル 1 または 2 が位相計数モードに 設定されている場合のみ選択可）
RAPI_AD_START_REQ	TGRA のインプットキャプチャ／コンペアマッチによる A/D 変換開始要求の発生を許可
RAPI_NO_AD_START_REQ	TGRA のインプットキャプチャ／コンペアマッチによる A/D 変換開始要求の発生を禁止
RAPI_TIMER_INT_LV_0	割り込み優先レベル 0
RAPI_TIMER_INT_LV_1	割り込み優先レベル 1
RAPI_TIMER_INT_LV_2	割り込み優先レベル 2
RAPI_TIMER_INT_LV_3	割り込み優先レベル 3
RAPI_TIMER_INT_LV_4	割り込み優先レベル 4
RAPI_TIMER_INT_LV_5	割り込み優先レベル 5
RAPI_TIMER_INT_LV_6	割り込み優先レベル 6
RAPI_TIMER_INT_LV_7	割り込み優先レベル 7
RAPI_TIMER_INT_LV_8	割り込み優先レベル 8
RAPI_TIMER_INT_LV_9	割り込み優先レベル 9
RAPI_TIMER_INT_LV_10	割り込み優先レベル 10
RAPI_TIMER_INT_LV_11	割り込み優先レベル 11
RAPI_TIMER_INT_LV_12	割り込み優先レベル 12
RAPI_TIMER_INT_LV_13	割り込み優先レベル 13
RAPI_TIMER_INT_LV_14	割り込み優先レベル 14
RAPI_TIMER_INT_LV_15	割り込み優先レベル 15

解説（3/7）

●RAPI_MTU2_0 または RAPI_NORMAL 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_EXTER_TCLKA_0, RAPI_EXTER_TCLKB_0, RAPI_EXTER_TCLKC_0, RAPI_EXTER_TCLKD_0 } から 1 つを指定します。

（動作方式） { RAPI_FREE_RUNNING, RAPI_PERIODIC } から 1 つを指定します。 初期値は RAPI_FREE_RUNNING です。

（カウントエッジ） { RAPI_RISING_EDGE, RAPI_FALLING_EDGE, RAPI_BOTH_EDGE } から 1 つを指定します。初期値は RAPI_RISING_EDGE です。

• RAPI_FREE_RUNNING 選択時は、次のパラメータを設定できます。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA, RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0, RAPI_TIMER_INT_LV_1, RAPI_TIMER_INT_LV_2, RAPI_TIMER_INT_LV_3, RAPI_TIMER_INT_LV_4, RAPI_TIMER_INT_LV_5, RAPI_TIMER_INT_LV_6, RAPI_TIMER_INT_LV_7, RAPI_TIMER_INT_LV_8, RAPI_TIMER_INT_LV_9, RAPI_TIMER_INT_LV_10, RAPI_TIMER_INT_LV_11, RAPI_TIMER_INT_LV_12, RAPI_TIMER_INT_LV_13, RAPI_TIMER_INT_LV_14, RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

• RAPI_PERIODIC 選択時は、次のパラメータを設定できます。

（カウントクリアソース） { RAPI_TCNT_CLEAR_TGRA, RAPI_TCNT_CLEAR_TGRB, RAPI_TCNT_CLEAR_TGRC, RAPI_TCNT_CLEAR_TGRD, RAPI_TCNT_CLEAR_OTHER } から 1 つを指定します。

（カウント出力） { RAPI_OUTPUT_RETAIN, RAPI_OUTPUT_0_0, RAPI_OUTPUT_0_1, RAPI_OUTPUT_0_TOG, RAPI_OUTPUT_1_0, RAPI_OUTPUT_1_1, RAPI_OUTPUT_1_TOG } から 1 つを指定します。初期値は RAPI_OUTPUT_RETAIN です。

（A/D 変換開始要求） { RAPI_AD_START_REQ, RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA, RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0, RAPI_TIMER_INT_LV_1, RAPI_TIMER_INT_LV_2, RAPI_TIMER_INT_LV_3, RAPI_TIMER_INT_LV_4, RAPI_TIMER_INT_LV_5, RAPI_TIMER_INT_LV_6, RAPI_TIMER_INT_LV_7, RAPI_TIMER_INT_LV_8, RAPI_TIMER_INT_LV_9, RAPI_TIMER_INT_LV_10, RAPI_TIMER_INT_LV_11, RAPI_TIMER_INT_LV_12, RAPI_TIMER_INT_LV_13, RAPI_TIMER_INT_LV_14, RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（4/7）

●RAPI_MTU2_1 および RAPI_NORMAL 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_EXTER_TCLKA_0, RAPI_EXTER_TCLKB_0, RAPI_TCNT2_FLOW } から 1 つを指定します。

（動作方式） { RAPI_FREE_RUNNING, RAPI_PERIODIC } から 1 つを指定します。初期値は RAPI_FREE_RUNNING です。

（カウントエッジ） { RAPI_RISING_EDGE, RAPI_FALLING_EDGE, RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_TCNT2_FLOW を指定した場合、カウントエッジは初期値のみの動作となります。

• RAPI_FREE_RUNNING 選択時は、次のパラメータを設定できます。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA, RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0, RAPI_TIMER_INT_LV_1, RAPI_TIMER_INT_LV_2, RAPI_TIMER_INT_LV_3, RAPI_TIMER_INT_LV_4, RAPI_TIMER_INT_LV_5, RAPI_TIMER_INT_LV_6, RAPI_TIMER_INT_LV_7, RAPI_TIMER_INT_LV_8, RAPI_TIMER_INT_LV_9, RAPI_TIMER_INT_LV_10, RAPI_TIMER_INT_LV_11, RAPI_TIMER_INT_LV_12, RAPI_TIMER_INT_LV_13, RAPI_TIMER_INT_LV_14, RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

• RAPI_PERIODIC 選択時は、次のパラメータを設定できます。

（カウントクリアソース） { RAPI_TCNT_CLEAR_TGRA, RAPI_TCNT_CLEAR_TGRB, RAPI_TCNT_CLEAR_OTHER } から 1 つを指定します。

（カウント出力） { RAPI_OUTPUT_RETAIN, RAPI_OUTPUT_0_0, RAPI_OUTPUT_0_1, RAPI_OUTPUT_0_TOG, RAPI_OUTPUT_1_0, RAPI_OUTPUT_1_1, RAPI_OUTPUT_1_TOG } から 1 つを指定します。
初期値は RAPI_OUTPUT_RETAIN です。

（A/D 変換開始要求） { RAPI_AD_START_REQ, RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA, RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。
初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0, RAPI_TIMER_INT_LV_1, RAPI_TIMER_INT_LV_2, RAPI_TIMER_INT_LV_3, RAPI_TIMER_INT_LV_4, RAPI_TIMER_INT_LV_5, RAPI_TIMER_INT_LV_6, RAPI_TIMER_INT_LV_7, RAPI_TIMER_INT_LV_8, RAPI_TIMER_INT_LV_9, RAPI_TIMER_INT_LV_10, RAPI_TIMER_INT_LV_11, RAPI_TIMER_INT_LV_12, RAPI_TIMER_INT_LV_13, RAPI_TIMER_INT_LV_14, RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（5/7）

●RAPI_MTU2_2 および RAPI_NORMAL 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_EXTER_TCLKA_0, RAPI_EXTER_TCLKB_0, RAPI_EXTER_TCLKC_0 } から 1 つを指定します。

（動作方式） { RAPI_FREE_RUNNING, RAPI_PERIODIC } から 1 つを指定します。初期値は RAPI_FREE_RUNNING です。

（カウントエッジ） { RAPI_RISING_EDGE, RAPI_FALLING_EDGE, RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_TCNT2_FLOW を指定した場合、カウントエッジは初期値のみの動作となります。

• RAPI_FREE_RUNNING 選択時は、次のパラメータを設定できます。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA, RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0, RAPI_TIMER_INT_LV_1, RAPI_TIMER_INT_LV_2, RAPI_TIMER_INT_LV_3, RAPI_TIMER_INT_LV_4, RAPI_TIMER_INT_LV_5, RAPI_TIMER_INT_LV_6, RAPI_TIMER_INT_LV_7, RAPI_TIMER_INT_LV_8, RAPI_TIMER_INT_LV_9, RAPI_TIMER_INT_LV_10, RAPI_TIMER_INT_LV_11, RAPI_TIMER_INT_LV_12, RAPI_TIMER_INT_LV_13, RAPI_TIMER_INT_LV_14, RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

• RAPI_PERIODIC 選択時は、次のパラメータを設定できます。

（カウントクリアソース） { RAPI_TCNT_CLEAR_TGRA, RAPI_TCNT_CLEAR_TGRB, RAPI_TCNT_CLEAR_OTHER } から 1 つを指定します。

（カウント出力） { RAPI_OUTPUT_RETAIN, RAPI_OUTPUT_0_0, RAPI_OUTPUT_0_1, RAPI_OUTPUT_0_TOG, RAPI_OUTPUT_1_0, RAPI_OUTPUT_1_1, RAPI_OUTPUT_1_TOG } から 1 つを指定します。
初期値は RAPI_OUTPUT_RETAIN です。

（A/D 変換開始要求） { RAPI_AD_START_REQ, RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA, RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。
初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0, RAPI_TIMER_INT_LV_1, RAPI_TIMER_INT_LV_2, RAPI_TIMER_INT_LV_3, RAPI_TIMER_INT_LV_4, RAPI_TIMER_INT_LV_5, RAPI_TIMER_INT_LV_6, RAPI_TIMER_INT_LV_7, RAPI_TIMER_INT_LV_8, RAPI_TIMER_INT_LV_9, RAPI_TIMER_INT_LV_10, RAPI_TIMER_INT_LV_11, RAPI_TIMER_INT_LV_12, RAPI_TIMER_INT_LV_13, RAPI_TIMER_INT_LV_14, RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（6/7）

●“RAPI_MTU2_3 または RAPI_MTU2_4”，および RAPI_NORMAL 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_EXTER_TCLKA_3，RAPI_EXTER_TCLKB_3 } から 1 つを指定します。

（動作方式） { RAPI_FREE_RUNNING，RAPI_PERIODIC } から 1 つを指定します。初期値は RAPI_FREE_RUNNING です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。初期値は RAPI_RISING_EDGE です。

• RAPI_FREE_RUNNING 選択時は、次のパラメータを設定できます。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

• RAPI_PERIODIC 選択時は、次のパラメータを設定できます。

（カウントクリアソース） { RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_TGRC，RAPI_TCNT_CLEAR_TGRD，RAPI_TCNT_CLEAR_OTHER } から 1 つを指定します。

（カウント出力） { RAPI_OUTPUT_RETAIN，RAPI_OUTPUT_0_0，RAPI_OUTPUT_0_1，RAPI_OUTPUT_0_TOG，RAPI_OUTPUT_1_0，RAPI_OUTPUT_1_1，RAPI_OUTPUT_1_TOG } から 1 つを指定します。初期値は RAPI_OUTPUT_RETAIN です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA，RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（7/7）

●RAPI_MTU2_1 または RAPI_MTU2_2，および RAPI_PHASE_COUNTING_1，RAPI_PHASE_COUNTING_2，RAPI_PHASE_COUNTING_3，RAPI_PHASE_COUNTING_4 のいずれかを選択時に指定可能なパラメータ：

（チャネル）	{ RAPI_MTU2_1，RAPI_MTU2_2 } から 1 つを指定します。
（割り込みイネーブル）	{ RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } または { RAPI_UNDERFLOW_ENA，RAPI_UNDERFLOW_DIS } の両方，またはいずれか一方から 1 つを指定します。初期値はそれぞれ RAPI_OVERFLOW_DIS，RAPI_UNDERFLOW_DIS です。
（割り込み優先レベル）	{ RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

[data2]
data2 には、タイマレジスタへの設定値を 16 ビット単位で設定して下さい。

[func1]
func1 には、コールバック関数を指すポインタを指定して下さい。本コールバック関数は、通常のイベントカウンタモード時のコンペアマッチ割り込みまたはオーバフロー割り込み用です。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。

[func2]
func2 には、コールバック関数を指すポインタを指定して下さい。本コールバック関数は、位相計数モード時のアンダフロー割り込み用です。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。

戻り値

タイマの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（イベントカウンタモード）

参照

__EnableEventCounter，__DestroyEventCounter，__GetTimerCounter

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

/* コールバック関数宣言 */
void TimerIntFunc( void );

void func( void )
{
   /* MTU2 チャネル 0 をイベントカウンタモードに設定 */
   __CreateEventCounter( RAPI_MTU2_0 | RAPI_EXTER_TCLKA_0 |
          RAPI_PERIODIC | RAPI_RISING_EDGE | RAPI_TCNT_CLEAR_TGRA |
          RAPI_OUTPUT_RETAIN | RAPI_COMPARE_MATCH_ENA |
          RAPI_TIMER_INT_LV_5, 0x8000, TimerIntFunc, RAPI_NULL );
}
```

5) __EnableEventCounter

概要

＜イベントカウンタモードの動作制御＞

Boolean __EnableEventCounter(unsigned long data)

data	設定データ

解説

イベントカウンタモードに設定されたタイマの動作開始／停止を制御します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数のタイマを動作制御できます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU20_4	MTU2 チャネル 0～4
RAPI_TIMER_ON	イベントカウンタモードに設定されているタイマ動作を開始
RAPI_TIMER_OFF	イベントカウンタモードに設定されているタイマ動作を停止

戻り値

タイマの指定が正しくない場合はRAPI_FALSE を返し、それ以外の場合はRAPI_TRUE を返します。

周辺機能

タイマ MTU2（イベントカウンタモード）

参照

__EnableTimer

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* MTU2 チャネル 0 およびチャネル 1 をイベントカウンタモードに設定し、
       タイマ動作を開始する */
    __EnableEventCounter( RAPI_MTU2_0 | RAPI_MTU2_1 | RAPI_TIMER_ON );
}
```

6) __DestroyEventCounter

概要

＜イベントカウンタモードの設定クリア＞

Boolean __DestroyEventCounter(unsigned long data)

data	設定データ

解説

タイマモードに設定されたタイマの設定をクリアします。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数のタイマの設定をクリアできます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU20_4	MTU2 チャネル 0～4

戻り値

タイマの指定が正しくない場合はRAPI_FALSE を返し、それ以外の場合はRAPI_TRUE を返します。

周辺機能

タイマ MTU2（イベントカウンタモード）

参照

__DestroyTimer

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* MTU2 チャネル 1 およびチャネル 2 に対し、イベントカウンタモードの設定クリア */
    __DestroyEventCounter( RAPI_MTU2_1 | RAPI_MTU2_2 );
}
```

7) __GetTimerCounter

概要

＜タイマモード／イベントカウンタモードのカウンタ値取得＞

Boolean __GetTimerCounter(unsigned long data1, unsigned short *data2)

data1	設定データ 1
data2	設定データ 2

解説

タイマモードまたはイベントカウンタモードに設定されたタイマのカウンタ値を取得します。

[data1]

data1 には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、複数のタイマのカウンタ値を同時に取得できます。（下記注意事項参照）

RAPI_MTU2_0	MTU2 チャネル 0（TCNT_0）
RAPI_MTU2_1	MTU2 チャネル 1（TCNT_1）
RAPI_MTU2_2	MTU2 チャネル 2（TCNT_2）
RAPI_MTU2_3	MTU2 チャネル 3（TCNT_3）
RAPI_MTU2_4	MTU2 チャネル 4（TCNT_4）
RAPI_MTU20_4	MTU2 チャネル 0～4
RAPI_MTU2_5U	MTU2 チャネル 5U（TCNTU_5）
RAPI_MTU2_5V	MTU2 チャネル 5V（TCNTV_5）
RAPI_MTU2_5W	MTU2 チャネル 5W（TCNTW_5）
RAPI_MTU2_5	MTU2 チャネル 5U，5V，および 5W
RAPI_CMT_0	CMT チャネル 0（CMCNT_0）
RAPI_CMT_1	CMT チャネル 1（CMCNT_1）
RAPI_CMT_ALL	CMT のすべてのチャネル

●タイマモード選択時に指定可能なパラメータ：

（カウントチャネル） { RAPI_MTU2_0，RAPI_MTU2_1，RAPI_MTU2_2，RAPI_MTU2_3，RAPI_MTU2_4，RAPI_MTU2_5U，RAPI_MTU2_5V，RAPI_MTU2_5W，RAPI_CMT_0，RAPI_CMT_1 } から 1 つを指定します。

【注】 • MTU2 チャネル 0～5 において、同時指定可能です。
• CMT チャネル 0 および 1 において、同時指定可能です。

●イベントカウンタモード選択時に指定可能なパラメータ：

（カウントチャネル） { RAPI_MTU2_0，RAPI_MTU2_1，RAPI_MTU2_2，RAPI_MTU2_3，RAPI_MTU2_4 } から 1 つを指定します。

[data2]

data2 には、タイマカウンタ値の格納用バッファを指すポインタを指定して下さい。

戻り値

タイマの指定が正しくない場合 RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（タイマモード／イベントカウンタモード）

参照

__CreateTimer

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    unsigned short data[2];

    /* MTU2 チャネル 0 およびチャネル 1 のタイマカウンタ値取得 */
    __GetTimerCounter( RAPI_MTU2_0 | RAPI_MTU2_1, data );
}
```

8) __CreatePWM

概要

＜パルス幅変調モードの設定＞

Boolean __CreatePWM(unsigned long data, void *func)

data	設定データ
func	コールバック関数ポインタ

解説（1/6）

指定したタイマをパルス幅変調モードに設定します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MP1	内部クロック：MPφ/1 でカウント
RAPI_MP4	内部クロック：MPφ/4 でカウント
RAPI_MP16	内部クロック：MPφ/16 でカウント
RAPI_MP64	内部クロック：MPφ/64 でカウント
RAPI_MP256	内部クロック：MPφ/256 でカウント
RAPI_MP256_1	内部クロック：MPφ/256 でカウント（チャネル 1 の場合）
RAPI_MP256_34	内部クロック：MPφ/256 でカウント （チャネル 3 または 4 の場合）
RAPI_MP1024_2	内部クロック：MPφ/1024 でカウント（チャネル 2 の場合）
RAPI_MP1024_34	内部クロック：MPφ/1024 でカウント （チャネル 3 または 4 の場合）
RAPI_EXTER_TCLKA_0	外部クロック：TCLKA 端子入力でカウント （チャネル 0～2 の場合）
RAPI_EXTER_TCLKB_0	外部クロック：TCLKB 端子入力でカウント （チャネル 0～2 の場合）
RAPI_EXTER_TCLKC_0	外部クロック：TCLKC 端子入力でカウント （チャネル 0 または 2 の場合）
RAPI_EXTER_TCLKD_0	外部クロック：TCLKD 端子入力でカウント （チャネル 0 の場合）
RAPI_EXTER_TCLKA_3	外部クロック：TCLKA 端子入力でカウント （チャネル 3 または 4 の場合）
RAPI_EXTER_TCLKB_3	外部クロック：TCLKB 端子入力でカウント （チャネル 3 または 4 の場合）
RAPI_RISING_EDGE	立ち上がりエッジでカウント
RAPI_FALLING_EDGE	立ち下がりエッジでカウント
RAPI_BOTH_EDGE	両エッジでカウント
【注】カウントソースに MPφ/1 を選択した場合、カウントエッジは 立ち上がりエッジのみの動作となります。	

解説（2/6）

RAPI_PWM_MODE1	PWM モード 1
RAPI_PWM_MODE2	PWM モード 2（チャネル 0～2 のみ対応）
RAPI_TCNT_CLEAR_DIS	TCNT クリアを禁止
RAPI_TCNT_CLEAR_TGRA	TGRA コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRB	TGRB コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRC	TGRC コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRD	TGRD コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_OTHER	同期クリア中または同期動作中の他のチャネルのカウンタクリアにより TCNT をクリア
RAPI_OVERFLOW_ENA	オーバフロー割り込み許可
RAPI_OVERFLOW_DIS	オーバフロー割り込み禁止
RAPI_AD_START_REQ	TGRA のインプットキャプチャ／コンペアマッチによる A/D 変換開始要求の発生を許可
RAPI_NO_AD_START_REQ	TGRA のインプットキャプチャ／コンペアマッチによる A/D 変換開始要求の発生を禁止
RAPI_TIMER_INT_LV_0	割り込み優先レベル 0
RAPI_TIMER_INT_LV_1	割り込み優先レベル 1
RAPI_TIMER_INT_LV_2	割り込み優先レベル 2
RAPI_TIMER_INT_LV_3	割り込み優先レベル 3
RAPI_TIMER_INT_LV_4	割り込み優先レベル 4
RAPI_TIMER_INT_LV_5	割り込み優先レベル 5
RAPI_TIMER_INT_LV_6	割り込み優先レベル 6
RAPI_TIMER_INT_LV_7	割り込み優先レベル 7
RAPI_TIMER_INT_LV_8	割り込み優先レベル 8
RAPI_TIMER_INT_LV_9	割り込み優先レベル 9
RAPI_TIMER_INT_LV_10	割り込み優先レベル 10
RAPI_TIMER_INT_LV_11	割り込み優先レベル 11
RAPI_TIMER_INT_LV_12	割り込み優先レベル 12
RAPI_TIMER_INT_LV_13	割り込み優先レベル 13
RAPI_TIMER_INT_LV_14	割り込み優先レベル 14
RAPI_TIMER_INT_LV_15	割り込み優先レベル 15
RAPI_TIMER_SYNC	チャネル 0～4 で同期動作させる
RAPI_TIMER_NO_SYNC	チャネル 0～4 で同期動作させない

解説（3/6）

●RAPI_MTU2_0 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，RAPI_MP64，
RAPI_EXTER_TCLKA_0，RAPI_EXTER_TCLKB_0，
RAPI_EXTER_TCLKC_0，RAPI_EXTER_TCLKD_0 }
から 1 つを指定します。初期値は RAPI_MP1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，
RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（動作モード） { RAPI_PWM_MODE1，RAPI_PWM_MODE2 } から
1 つを指定します。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，
RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_TGRC，
RAPI_TCNT_CLEAR_TGRD，RAPI_TCNT_CLEAR_OTHER }
から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から
1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から
1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，
RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，
RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，
RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，
RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，
RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，
RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，
RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 }
から 1 つを指定します。
初期値は RAPI_TIMER_INT_LV_0 です。

解説（4/6）

●RAPI_MTU2_1 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，
RAPI_MP64，RAPI_MP256_1，RAPI_EXTER_TCLKA_0，
RAPI_EXTER_TCLKB _0 } から 1 つを指定します。
初期値は RAPI_MP1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，
RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（動作モード） { RAPI_PWM_MODE1，RAPI_PWM_MODE2 } から
1 つを指定します。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，
RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_OTHER }
から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から
1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から
1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，
RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，
RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，
RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，
RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，
RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，
RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，
RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 }
から 1 つを指定します。
初期値は RAPI_TIMER_INT_LV_0 です。

解説（5/6）

●RAPI_MTU2_2 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，
RAPI_MP64，RAPI_MP1024_2，RAPI_EXTER_TCLKA_0，
RAPI_EXTER_TCLKB_0，RAPI_EXTER_TCLKC_0 }
から 1 つを指定します。初期値は RAPI_MP1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，
RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（動作モード） { RAPI_PWM_MODE1，RAPI_PWM_MODE2 } から
1 つを指定します。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，
RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_OTHER }
から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から
1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から
1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，
RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，
RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，
RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，
RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，
RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，
RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，
RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 }
から 1 つを指定します。
初期値は RAPI_TIMER_INT_LV_0 です。

解説（6/6）

●RAPI_MTU2_3 または RAPI_MTU2_4 選択時に指定可能なパラメータ：
（ただし、PWM モード 1（RAPI_PWM_MODE1）のみ対応）

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，RAPI_MP64，
RAPI_MP256_34，RAPI_MP1024_34，
RAPI_EXTER_TCLKA_3，RAPI_EXTER_TCLKB_3 }
から 1 つを指定します。初期値は RAPI_MP1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，
RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，
RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_TGRC，
RAPI_TCNT_CLEAR_TGRD，RAPI_TCNT_CLEAR_OTHER }
から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から
1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から
1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，
RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，
RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，
RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，
RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，
RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，
RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，
RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から
1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

●同期動作設定時に指定するパラメータについて：
チャネル 0～4 のいずれかで同期動作を設定する場合、RAPI_TIMER_SYNC を指定して下さい。
同期動作を設定しない場合（各チャネルで独立に動作）は、RAPI_TIMER_NO_SYNC を指定して下さい。初期値は RAPI_TIMER_NO_SYNC です。

[func]
func には、コールバック関数を指すポインタを指定して下さい。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。

戻り値

タイマの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（パルス幅変調モード）

参照

__EnablePWM，__DestroyPWM

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

/* コールバック関数宣言 */
void TimerIntFunc( void );

void func( void )
{
   /* MTU2 チャネル 1 をパルス幅変調モードに設定 */
   __CreatePWM( RAPI_MTU2_1 | RAPI_MP_4 | RAPI_RISING_EDGE |
          RAPI_PWM_MODE1 | RAPI_TCNT_CLEAR_TGRB | RAPI_OVERFLOW_DIS |
          RAPI_TIMER_INT_LV_0 | RAPI_TIMER_NO_SYNC, TimerIntFunc );
}
```

9) __SetPWMPin

概要

＜パルス幅変調用タイマジェネラルレジスタの設定＞

Boolean __SetPWMPin(unsigned long data1, unsigned short data2, void *func)

data1	設定データ 1
data2	設定データ 2
func	コールバック関数ポインタ

解説（1/3）

指定したチャネルのタイマジェネラルレジスタをパルス幅変調用に設定します。

[data1]

data1 には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_TGRA	タイマジェネラルレジスタ A
RAPI_TGRB	タイマジェネラルレジスタ B
RAPI_TGRC	タイマジェネラルレジスタ C
RAPI_TGRD	タイマジェネラルレジスタ D
RAPI_OUTPUT_RETAIN	出力保持
RAPI_OUTPUT_0_0	初期出力 0，コンペアマッチで 0 を出力
RAPI_OUTPUT_0_1	初期出力 0，コンペアマッチで 1 を出力
RAPI_OUTPUT_0_TOG	初期出力 0，コンペアマッチでトグル出力
RAPI_OUTPUT_1_0	初期出力 1，コンペアマッチで 0 を出力
RAPI_OUTPUT_1_1	初期出力 1，コンペアマッチで 1 を出力
RAPI_OUTPUT_1_TOG	初期出力 1，コンペアマッチでトグル出力
RAPI_COMPARE_MATCH_ENA	コンペアマッチ割り込みを許可
RAPI_COMPARE_MATCH_DIS	コンペアマッチ割り込みを禁止
RAPI_TIMER_INT_LV_0	割り込み優先レベル 0
RAPI_TIMER_INT_LV_1	割り込み優先レベル 1
RAPI_TIMER_INT_LV_2	割り込み優先レベル 2
RAPI_TIMER_INT_LV_3	割り込み優先レベル 3
RAPI_TIMER_INT_LV_4	割り込み優先レベル 4
RAPI_TIMER_INT_LV_5	割り込み優先レベル 5
RAPI_TIMER_INT_LV_6	割り込み優先レベル 6
RAPI_TIMER_INT_LV_7	割り込み優先レベル 7
RAPI_TIMER_INT_LV_8	割り込み優先レベル 8
RAPI_TIMER_INT_LV_9	割り込み優先レベル 9
RAPI_TIMER_INT_LV_10	割り込み優先レベル 10
RAPI_TIMER_INT_LV_11	割り込み優先レベル 11
RAPI_TIMER_INT_LV_12	割り込み優先レベル 12
RAPI_TIMER_INT_LV_13	割り込み優先レベル 13
RAPI_TIMER_INT_LV_14	割り込み優先レベル 14
RAPI_TIMER_INT_LV_15	割り込み優先レベル 15

解説（2/3）

●RAPI_MTU2_0，RAPI_MTU2_3 または RAPI_MTU2_4 選択時に指定可能なパラメータ：

（タイマジェネラルレジスタ） { RAPI_TGRA，RAPI_TGRB，RAPI_TGRC，RAPI_TGRD } から 1 つを指定します。

（カウンタ出力） { RAPI_OUTPUT_RETAIN，RAPI_OUTPUT_0_0，RAPI_OUTPUT_0_1，RAPI_OUTPUT_0_TOG，RAPI_OUTPUT_1_0，RAPI_OUTPUT_1_1，RAPI_OUTPUT_1_TOG } から 1 つを指定します。
初期値は RAPI_OUTPUT_RETAIN です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA，RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。
初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。
初期値は RAPI_TIMER_INT_LV_0 です。

●RAPI_MTU2_1 または RAPI_MTU2_2 選択時に指定可能なパラメータ：

（タイマジェネラルレジスタ） { RAPI_TGRA，RAPI_TGRB } から 1 つを指定します。

（カウンタ出力） { RAPI_OUTPUT_RETAIN，RAPI_OUTPUT_0_0，RAPI_OUTPUT_0_1，RAPI_OUTPUT_0_TOG，RAPI_OUTPUT_1_0，RAPI_OUTPUT_1_1，RAPI_OUTPUT_1_TOG } から 1 つを指定します。
初期値は RAPI_OUTPUT_RETAIN です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA，RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。
初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。
初期値は RAPI_TIMER_INT_LV_0 です。

解説（3/3）

[data2]
data2 には、デューティレジスタ値または周期レジスタ値（16 ビットカウンタ値）を指定して下さい。
【注】「デューティレジスタ値 ＜ 周期レジスタ値」となります。
- PWM モード 1 では、TGRA および TGRB，また TGRC および TGRD で一組となります。
- PWM モード 2 では、1 つが周期レジスタで、その他はデューティレジスタになります。

[func]
func には、コールバック関数を指すポインタを指定して下さい。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。既に指定したコールバック関数を保持する場合は、RAPI_HOLD を指定して下さい。これにより、data2 の値のみ変更されます。

戻り値

チャネルの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（パルス幅変調モード）

参照

__EnablePWM，__DestroyPWM

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

/* コールバック関数宣言 */
void TimerIntBFunc( void );
void TimerIntAFunc( void );

void func( void )
{
   /* MTU2 チャネル 1 の TGRB（TGRB_1）を周期レジスタに設定 */
   __SetPWMPin( RAPI_MTU2_1 | RAPI_TGRB | RAPI_OUTPUT_0_0 |
          RAPI_COMPARE_MATCH_ENA | RAPI_TIMER_INT_LV_3,
          50000, TimerIntBFunc );

   /* MTU2 チャネル 1 の TGRA（TGRA_1）をデューティレジスタに設定 */
   __SetPWMPin( RAPI_MTU2_1 | RAPI_TGRA | RAPI_OUTPUT_0_1 |
          RAPI_COMPARE_MATCH_ENA | RAPI_TIMER_INT_LV_3,
          30000, TimerIntAFunc );
}
```

10) __EnablePWM

概要

＜パルス幅変調モードの動作制御＞

Boolean __EnablePWM(unsigned long data)

data	設定データ

解説

パルス幅変調モードに設定されたタイマの動作開始／動作停止を制御します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数のタイマを動作制御できます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU20_4	MTU2 チャネル 0～4
RAPI_TIMER_ON	パルス幅変調モードに設定されているタイマを動作開始に設定
RAPI_TIMER_OFF	パルス幅変調モードに設定されているタイマを動作停止に設定

戻り値

タイマの指定が正しくない場合はRAPI_FALSE を返し、それ以外の場合はRAPI_TRUE を返します。

周辺機能

タイマ MTU2（パルス幅変調モード）

参照

__CreatePWM，__DestroyPWM

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* パルス幅変調モードに設定した MTU2 チャネル 1 およびチャネル 2 に対し、
       タイマ動作開始 */
    __EnablePWM( RAPI_MTU2_1 | RAPI_MTU2_2 | RAPI_TIMER_ON );
}
```

11) __DestroyPWM

概要

＜パルス幅変調モードの設定クリア＞

Boolean __DestroyPWM(unsigned long data)

data	設定データ

解説

パルス幅変調モードに設定されたタイマの設定をクリアします。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数のタイマの設定をクリアすることができます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU20_4	MTU2 チャネル 0～4

戻り値

タイマの指定が正しくない場合はRAPI_FALSE を返し、それ以外の場合はRAPI_TRUE を返します。

周辺機能

タイマ MTU2（パルス幅変調モード）

参照

__CreatePWM，__EnablePWM

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* MTU2 チャネル 1 およびチャネル 2 に対し、パルス幅変調モードの設定をクリア */
    __DestroyPWM( RAPI_MTU2_1 | RAPI_MTU2_2 );
}
```

12) __CreatePulsePeriodMeasurementMode

概要

＜パルス周期測定モードの設定＞

Boolean __CreatePulsePeriodMeasurementMode(unsigned long data, void *func)

data	設定データ
func	コールバック関数ポインタ

解説（1/5）

指定したタイマをパルス周期測定モードに設定します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを設定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_MP_1	内部クロック：MP φ/1 でカウント
RAPI_MP_4	内部クロック：MP φ/4 でカウント
RAPI_MP_16	内部クロック：MP φ/16 でカウント
RAPI_MP_64	内部クロック：MP φ/64 でカウント
RAPI_MP_256_1	内部クロック：MP φ/256 でカウント（チャネル 1 の場合）
RAPI_MP_256_34	内部クロック：MP φ/256 でカウント （チャネル 3 または 4 の場合）
RAPI_MP_1024_2	内部クロック：MP φ/1024 でカウント（チャネル 2 の場合）
RAPI_MP_1024_34	内部クロック：MP φ/1024 でカウント （チャネル 3 または 4 の場合）
RAPI_RISING_EDGE	立ち上がりエッジでカウント
RAPI_FALLING_EDGE	立ち下がりエッジでカウント
RAPI_BOTH_EDGE	両エッジでカウント
【注】カウントソースに MPφ/1 を選択した場合、カウントエッジは 立ち上がりエッジのみの動作となります。	
RAPI_AD_START_REQ	TGRA のインプットキャプチャ／コンペアマッチによる A/D 変換開始要求の発生を許可
RAPI_NO_AD_START_REQ	TGRA のインプットキャプチャ／コンペアマッチによる A/D 変換開始要求の発生を禁止
RAPI_TCNT_CLEAR_DIS	TCNT クリアを禁止
RAPI_TCNT_CLEAR_TGRA	TGRA インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRB	TGRB インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRC	TGRC インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRD	TGRD インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRU	TGRU インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRV	TGRV インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRW	TGRW インプットキャプチャで TCNT をクリア

解説（2/5）

RAPI_OVERFLOW_ENA	オーバフロー割り込みを許可
RAPI_OVERFLOW_DIS	オーバフロー割り込みを禁止
RAPI_TIMER_INT_LV_0	割り込み優先レベル 0
RAPI_TIMER_INT_LV_1	割り込み優先レベル 1
RAPI_TIMER_INT_LV_2	割り込み優先レベル 2
RAPI_TIMER_INT_LV_3	割り込み優先レベル 3
RAPI_TIMER_INT_LV_4	割り込み優先レベル 4
RAPI_TIMER_INT_LV_5	割り込み優先レベル 5
RAPI_TIMER_INT_LV_6	割り込み優先レベル 6
RAPI_TIMER_INT_LV_7	割り込み優先レベル 7
RAPI_TIMER_INT_LV_8	割り込み優先レベル 8
RAPI_TIMER_INT_LV_9	割り込み優先レベル 9
RAPI_TIMER_INT_LV_10	割り込み優先レベル 10
RAPI_TIMER_INT_LV_11	割り込み優先レベル 11
RAPI_TIMER_INT_LV_12	割り込み優先レベル 12
RAPI_TIMER_INT_LV_13	割り込み優先レベル 13
RAPI_TIMER_INT_LV_14	割り込み優先レベル 14
RAPI_TIMER_INT_LV_15	割り込み優先レベル 15

●RAPI_MTU2_0 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64 } から 1 つを指定します。初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_TGRC，RAPI_TCNT_CLEAR_TGRD } から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（3/5）

●RAPI_MTU2_1 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64，RAPI_MP_256_1 } から 1 つを指定します。
初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB } から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（4/5）

●RAPI_MTU2_2 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64，RAPI_MP_1024_2 } から 1 つを指定します。
初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB } から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（5/5）

●RAPI_MTU2_3 または RAPI_MTU2_4 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64，RAPI_MP_256_34，RAPI_MP_1024_34 } から 1 つを指定します。
初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_TGRC，RAPI_TCNT_CLEAR_TGRD } から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

●RAPI_MTU2_5 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64 } から 1 つを指定します。初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRU，RAPI_TCNT_CLEAR_TGRV，RAPI_TCNT_CLEAR_TGRW } から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

[func]
func には、コールバック関数を格納しているポインタを指定して下さい。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。

戻り値

タイマの指定が正しくない場合はRAPI_FALSEを返し、それ以外の場合はRAPI_TRUEを返します。

周辺機能

タイマ MTU2（パルス周期測定モード）

参照

__CreateInputCapture

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

/* コールバック関数宣言 */
void TimerIntFunc( void );

void func( void )
{
   /* MTU2 チャネル 4 をパルス周期測定モードに設定 */
   __CreatePulsePeriodMeasurementMode( RAPI_MTU2_4 | RAPI_MP_1 |
          RAPI_RISING_EDGE | RAPI_TCNT_CLEAR_DIS | RAPI_OVERFLOW_ENA |
          RAPI_TIMER_INT_LV_1, TimerIntFunc );
}
```

13) __SetInputPeriodPin

概要

＜パルス周期測定用入力端子の設定＞

Boolean __SetInputPeriodPin(unsigned long data, void *func)

data	設定データ
func	コールバック関数ポインタ

解説（1/3）

指定した端子をパルス周期測定用の入力端子に設定します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_TIOCA	TIOCiA 端子（i = 0～4）
RAPI_TIOCB	TIOCiB 端子（i = 0～4）
RAPI_TIOCC	TIOCiC 端子（i = 0，3，4）
RAPI_TIOCD	TIOCiD 端子（i = 0，3，4）
RAPI_RISING_CAP_0	立ち上がりエッジでインプットキャプチャ（チャネル 0～4 の場合）
RAPI_FALLING_CAP_0	立ち下がりエッジでインプットキャプチャ（チャネル 0～4 の場合）
RAPI_RISING_CAP_5	立ち上がりエッジでインプットキャプチャ（チャネル 5 の場合）
RAPI_FALLING_CAP_5	立ち下がりエッジでインプットキャプチャ（チャネル 5 の場合）
RAPI_CAPTURE_ENA	TGRi インプットキャプチャ割り込みを許可（i = A，B，C，D，U，V，W）
RAPI_CAPTURE_DIS	インプットキャプチャ割り込みを禁止
RAPI_TIMER_INT_LV_0	割り込み優先レベル 0
RAPI_TIMER_INT_LV_1	割り込み優先レベル 1
RAPI_TIMER_INT_LV_2	割り込み優先レベル 2
RAPI_TIMER_INT_LV_3	割り込み優先レベル 3
RAPI_TIMER_INT_LV_4	割り込み優先レベル 4
RAPI_TIMER_INT_LV_5	割り込み優先レベル 5
RAPI_TIMER_INT_LV_6	割り込み優先レベル 6
RAPI_TIMER_INT_LV_7	割り込み優先レベル 7
RAPI_TIMER_INT_LV_8	割り込み優先レベル 8
RAPI_TIMER_INT_LV_9	割り込み優先レベル 9
RAPI_TIMER_INT_LV_10	割り込み優先レベル 10
RAPI_TIMER_INT_LV_11	割り込み優先レベル 11
RAPI_TIMER_INT_LV_12	割り込み優先レベル 12
RAPI_TIMER_INT_LV_13	割り込み優先レベル 13
RAPI_TIMER_INT_LV_14	割り込み優先レベル 14
RAPI_TIMER_INT_LV_15	割り込み優先レベル 15

解説（2/3）

●RAPI_MTU2_0，RAPI_MTU2_3 または RAPI_MTU2_4 選択時に指定可能なパラメータ：

（キャプチャチャネル） { RAPI_TIOCA，RAPI_TIOCB，RAPI_TIOCC，RAPI_TIOCD } から 1 つを指定します。

（キャプチャエッジ） { RAPI_RISING_CAP_0，RAPI_FALLING_CAP_0 } から 1 つを指定します。

（割り込みイネーブル） { RAPI_CAPTURE_ENA，RAPI_CAPTURE_DIS } から 1 つを指定します。初期値は RAPI_CAPTURE_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

●RAPI_MTU2_1 または RAPI_MTU2_2 選択時に指定可能なパラメータ：

（キャプチャチャネル） { RAPI_TIOCA，RAPI_TIOCB } から 1 つを指定します。

（キャプチャエッジ） { RAPI_RISING_CAP_0，RAPI_FALLING_CAP_0 } から 1 つを指定します。

（割り込みイネーブル） { RAPI_CAPTURE_ENA，RAPI_CAPTURE_DIS } から 1 つを指定します。初期値は RAPI_CAPTURE_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

●RAPI_MTU2_5 選択時に指定可能なパラメータ：

（キャプチャチャネル） { RAPI_MTU2_5U，RAPI_MTU2_5V，RAPI_MTU2_5W } から 1 つを指定します。

（キャプチャエッジ） { RAPI_RISING_CAP_5，RAPI_FALLING_CAP_5 } から 1 つを指定します。

（割り込みイネーブル） { RAPI_CAPTURE_ENA，RAPI_CAPTURE_DIS } から 1 つを指定します。初期値は RAPI_CAPTURE_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（3/3）

[func]
func には、コールバック関数を格納しているポインタを指定して下さい。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。

戻り値

タイマの指定が正しくない場合はRAPI_FALSE を返し、それ以外の場合はRAPI_TRUE を返します。

周辺機能

タイマ MTU2（パルス周期測定モード）

参照

__SetInputCapturePin

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void TimerIntFunc( void );

void func( void )
{
    /* MTU2 チャネル 0 の TGRA をパルス周期測定用に設定 */
    __SetInputPeriodPin( RAPI_MTU2_0 | RAPI_TIOCA | RAPI_RISING_CAP_0 |
           RAPI_CAPTURE_ENA | RAPI_TIMER_INT_LV_3, TimerIntFunc );
}
```

14) __EnablePulsePeriodMeasurementMode

概要

＜パルス周期測定モードの動作制御＞

Boolean __EnablePulsePeriodMeasurementMode(unsigned long data)

data	設定データ

解説

パルス周期測定モードに設定されたタイマの動作開始／動作停止を制御します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数のタイマを動作制御できます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_TIMER_ON	パルス周期測定モードに設定されているタイマを動作開始に設定
RAPI_TIMER_OFF	パルス周期測定モードに設定されているタイマを動作停止に設定

戻り値

タイマの指定が正しくない場合はRAPI_FALSE を返し、それ以外の場合はRAPI_TRUE を返します。

周辺機能

タイマ MTU2（パルス周期測定モード）

参照

__EnableInputCapture

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* パルス周期測定モードに設定した MTU2 チャネル 4 に対し、タイマ動作を開始 */
    __EnablePulsePeriodMeasurementMode( RAPI_MTU2_4 | RAPI_TIMER_ON );
}
```

15) __DestroyPulsePeriodMeasurementMode

概要

＜パルス周期測定モードの設定クリア＞

Boolean __DestroyPulsePeriodMeasurementMode(unsigned long data)

data	設定データ

解説

パルス周期測定モードに設定されたタイマの設定をクリアします。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数のタイマの設定をクリアすることができます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU20_4	MTU2 チャネル 0～4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_MTU2_5	MTU2 チャネル 5U，5V，および 5W
RAPI_MTU2_ALL	すべての MTU2 チャネル

戻り値

タイマの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（パルス周期測定モード）

参照

__DestroyInputCapture

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* MTU2 チャネル 4 に対し、パルス周期測定モードの設定をクリア */
    __DestroyPulsePeriodMeasurementMode( RAPI_MTU2_4 );
}
```

16) __GetPulsePeriodMeasurementMode

概要

＜パルス周期測定モードの測定値取得＞

Boolean __GetPulsePeriodMeasurementMode(unsigned long data1, unsigned short *data2)

data1	設定データ 1
data2	カウンタ値を格納するバッファへのポインタ

解説

パルス周期測定モードに設定されたタイマのカウンタ値を取得します。

[data1]

data1 には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数の端子を選択することができます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_TIOCA	TIOCiA 端子（i = 0～4）
RAPI_TIOCB	TIOCiB 端子（i = 0～4）
RAPI_TIOCC	TIOCiC 端子（i = 0，3，4）
RAPI_TIOCD	TIOCiD 端子（i = 0，3，4）

戻り値

タイマの指定が正しくない場合はRAPI_FALSE を返し、それ以外の場合はRAPI_TRUE を返します。

周辺機能

タイマ MTU2（パルス周期測定モード）

参照

__GetCaptureValue

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    unsigned short data[2];

    /* パルス周期測定モードに設定した MTU2 チャネル 4 に対し、
       TGRA および TGRB の測定値を取得 */
    __GetPulsePeriodMeasurementMode( RAPI_MTU2_4 | RAPI_TIOCA |
            RAPI_TIOCB, data );
}
```

17) __CreatePulseWidthMeasurementMode

概要

＜パルス幅測定モードの設定＞

Boolean __CreatePulseWidthMeasurementMode(unsigned long data, void *func)

data	設定データ
func	コールバック関数ポインタ

解説（1/5）

指定したタイマをパルス幅測定モードに設定します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_MP_1	内部クロック：MPφ/1 でカウント
RAPI_MP_4	内部クロック：MPφ/4 でカウント
RAPI_MP_16	内部クロック：MPφ/16 でカウント
RAPI_MP_64	内部クロック：MPφ/64 でカウント
RAPI_MP_256_1	内部クロック：MPφ/256 でカウント（チャネル 1 の場合）
RAPI_MP_256_34	内部クロック：MPφ/256 でカウント （チャネル 3 または 4 の場合）
RAPI_MP_1024_2	内部クロック：MPφ/256 でカウント（チャネル 2 の場合）
RAPI_MP_1024_34	内部クロック：MPφ/256 でカウント （チャネル 3 または 4 の場合）
RAPI_RISING_EDGE	立ち上がりエッジでカウント
RAPI_FALLING_EDGE	立ち下がりエッジでカウント
RAPI_BOTH_EDGE	両エッジでカウント
【注】カウントソースに MPφ/1 を選択した場合、カウントエッジは 立ち上がりエッジのみの動作となります。	
RAPI_TCNT_CLEAR_DIS	TCNT クリアを禁止
RAPI_TCNT_CLEAR_TGRA	TGRA インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRB	TGRB インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRC	TGRC インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRD	TGRD インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRU	TGRU インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRV	TGRV インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRW	TGRW インプットキャプチャで TCNT をクリア

解説（2/5）

RAPI_OVERFLOW_ENA	オーバフロー割り込みを許可
RAPI_OVERFLOW_DIS	オーバフロー割り込みを禁止
RAPI_AD_START_REQ	TGRA のインプットキャプチャ／コンペアマッチによる A/D 変換開始要求の発生を許可
RAPI_NO_AD_START_REQ	TGRA のインプットキャプチャ／コンペアマッチによる A/D 変換開始要求の発生を禁止
RAPI_TIMER_INT_LV_0	割り込み優先レベル 0
RAPI_TIMER_INT_LV_1	割り込み優先レベル 1
RAPI_TIMER_INT_LV_2	割り込み優先レベル 2
RAPI_TIMER_INT_LV_3	割り込み優先レベル 3
RAPI_TIMER_INT_LV_4	割り込み優先レベル 4
RAPI_TIMER_INT_LV_5	割り込み優先レベル 5
RAPI_TIMER_INT_LV_6	割り込み優先レベル 6
RAPI_TIMER_INT_LV_7	割り込み優先レベル 7
RAPI_TIMER_INT_LV_8	割り込み優先レベル 8
RAPI_TIMER_INT_LV_9	割り込み優先レベル 9
RAPI_TIMER_INT_LV_10	割り込み優先レベル 10
RAPI_TIMER_INT_LV_11	割り込み優先レベル 11
RAPI_TIMER_INT_LV_12	割り込み優先レベル 12
RAPI_TIMER_INT_LV_13	割り込み優先レベル 13
RAPI_TIMER_INT_LV_14	割り込み優先レベル 14
RAPI_TIMER_INT_LV_15	割り込み優先レベル 15

●RAPI_MTU2_0 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64 } から 1 つを指定します。初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_TGRC，RAPI_TCNT_CLEAR_TGRD } から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（3/5）

●RAPI_MTU2_1 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64，RAPI_MP_256_1 } から 1 つを指定します。
初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB } から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（4/5）

●RAPI_MTU2_2 選択時に指定可能なパラメータ：

（カウントソース）
{ RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64，RAPI_MP_1024_2 } から 1 つを指定します。
初期値は RAPI_MP_1 です。

（カウントエッジ）
{ RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース）
{ RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB } から 1 つを指定します。
初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求）
{ RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル）
{ RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル）
{ RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（5/5）

●RAPI_MTU2_3 および RAPI_MTU2_4 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64，RAPI_MP_256_34, RAPI_MP_1024_34 } から 1 つを指定します。初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_TGRC，RAPI_TCNT_CLEAR_TGRD } から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

●RAPI_MTU2_5 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64 } から 1 つを指定します。初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRU，RAPI_TCNT_CLEAR_TGRV，RAPI_TCNT_CLEAR_TGRW } から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

[func]
func には、コールバック関数を指すポインタを指定して下さい。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。

戻り値

タイマの指定が正しくない場合はRAPI_FALSEを返し、それ以外の場合はRAPI_TRUEを返します。

周辺機能

タイマ MTU2（パルス幅測定モード）

参照

__CreateInputCapture

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

/* コールバック関数宣言 */
void TimerIntFunc( void );

void func( void )
{
   /* MTU2 チャネル 4 をパルス幅測定モードに設定 */
   __CreatePulseWidthMeasurementMode( RAPI_MTU2_4 | RAPI_MP_1 |
          RAPI_RISING_EDGE | RAPI_TCNT_CLEAR_TGRB | RAPI_OVERFLOW_DIS |
          RAPI_TIMER_INT_LV_0, TimerIntFunc );
}
```

18) __SetInputWidthPin

概要

＜パルス幅測定用入力端子の設定＞

Boolean __SetInputWidthPin(unsigned long data, void *func)

data	設定データ
func	コールバック関数ポインタ

解説（1/3）

指定した端子をパルス幅測定用の入力端子に設定します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_TIOCA	TIOCiA 端子（i = 0～4）
RAPI_TIOCB	TIOCiB 端子（i = 0～4）
RAPI_TIOCC	TIOCiC 端子（i = 0，3，4）
RAPI_TIOCD	TIOCiD 端子（i = 0，3，4）
RAPI_CAPTURE_ENA	TGRi インプットキャプチャ割り込みを許可 （i = A，B，C，D，U，V，W）
RAPI_CAPTURE_DIS	インプットキャプチャ割り込みを禁止
RAPI_TIMER_INT_LV_0	割り込み優先レベル 0
RAPI_TIMER_INT_LV_1	割り込み優先レベル 1
RAPI_TIMER_INT_LV_2	割り込み優先レベル 2
RAPI_TIMER_INT_LV_3	割り込み優先レベル 3
RAPI_TIMER_INT_LV_4	割り込み優先レベル 4
RAPI_TIMER_INT_LV_5	割り込み優先レベル 5
RAPI_TIMER_INT_LV_6	割り込み優先レベル 6
RAPI_TIMER_INT_LV_7	割り込み優先レベル 7
RAPI_TIMER_INT_LV_8	割り込み優先レベル 8
RAPI_TIMER_INT_LV_9	割り込み優先レベル 9
RAPI_TIMER_INT_LV_10	割り込み優先レベル 10
RAPI_TIMER_INT_LV_11	割り込み優先レベル 11
RAPI_TIMER_INT_LV_12	割り込み優先レベル 12
RAPI_TIMER_INT_LV_13	割り込み優先レベル 13
RAPI_TIMER_INT_LV_14	割り込み優先レベル 14
RAPI_TIMER_INT_LV_15	割り込み優先レベル 15

解説（2/3）

●RAPI_MTU2_0，RAPI_MTU2_3 または RAPI_MTU2_4 選択時に指定可能なパラメータ：

（キャプチャチャネル） { RAPI_TIOCA，RAPI_TIOCB，RAPI_TIOCC，RAPI_TIOCD } から 1 つを指定します。

（割り込みイネーブル） { RAPI_CAPTURE_ENA，RAPI_CAPTURE_DIS } から 1 つを指定します。初期値は RAPI_CAPTURE_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

●RAPI_MTU2_1 または RAPI_MTU2_2 選択時に指定可能なパラメータ：

（キャプチャチャネル） { RAPI_TIOCA，RAPI_TIOCB } から 1 つを指定します。

（割り込みイネーブル） { RAPI_CAPTURE_ENA，RAPI_CAPTURE_DIS } から 1 つを指定します。初期値は RAPI_CAPTURE_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

●RAPI_MTU2_5 選択時に指定可能なパラメータ：

（キャプチャチャネル） { RAPI_MTU2_5U，RAPI_MTU2_5V，RAPI_MTU2_5W } から 1 つを指定します。

（割り込みイネーブル） { RAPI_CAPTURE_ENA，RAPI_CAPTURE_DIS } から 1 つを指定します。初期値は RAPI_CAPTURE_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（3/3）

[func]
func には、コールバック関数を指すポインタを指定して下さい。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。

戻り値

タイマの指定が正しくない場合はRAPI_FALSE を返し、それ以外の場合はRAPI_TRUE を返します。

周辺機能

タイマ MTU2（パルス幅測定モード）

参照

__SetInputCapturePin

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

/* コールバック関数宣言 */
void TimerIntFunc( void );

void func( void )
{
   /* MTU2 チャネル 4 に対し、TGRA をパルス幅測定用に設定 */
   __SetInputWidthPin( RAPI_MTU2_4 | RAPI_TIOCA | RAPI_CAPTURE_ENA |
          RAPI_TIMER_INT_LV_3, TimerIntFunc );
}
```

19) __EnablePulseWidthMeasurementMode

概要

＜パルス幅測定モードの動作制御＞

Boolean __EnablePulseWidthMeasurementMode(unsigned long data)

data	設定データ

解説

パルス幅測定モードに設定されたタイマの動作開始／動作停止を制御します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数のタイマを動作制御できます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_TIMER_ON	パルス幅測定モードに設定されているタイマを動作開始に設定
RAPI_TIMER_OFF	パルス幅測定モードに設定されているタイマを動作停止に設定

戻り値

タイマの指定が正しくない場合はRAPI_FALSE を返し、それ以外の場合はRAPI_TRUE を返します。

周辺機能

タイマ MTU2（パルス幅測定モード）

参照

__EnableInputCapture

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* パルス幅測定モードに設定した MTU2 チャネル 4 に対し、タイマ動作を開始 */
    __EnablePulseWidthMeasurementMode( RAPI_MTU2_4 | RAPI_TIMER_ON );
}
```

20) __DestroyPulseWidthMeasurementMode

概要

＜パルス幅測定モードの設定クリア＞

Boolean __DestroyPulseWidthMeasurementMode(unsigned long data)

data	設定データ

解説

パルス幅測定モードに設定されたタイマの設定をクリアします。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数のタイマの設定をクリアすることができます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU20_4	MTU2 チャネル 0～4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_MTU2_5	MTU2 チャネル 5U，5V，および 5W
RAPI_MTU2_ALL	すべての MTU2 チャネル

戻り値

タイマの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（パルス幅測定モード）

参照

__DestroyInputCapture

備考

引数に上記以外のパラメータを設定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* MTU2 チャネル 4 に対し、パルス幅測定モードの設定をクリア */
    __DestroyPulseWidthMeasurementMode( RAPI_MTU2_4 );
}
```

21) __GetPulseWidthMeasurementMode

概要

＜パルス幅測定モードの測定値取得＞

Boolean __GetPulseWidthMeasurementMode(unsigned long data1, unsigned short *data2)

data1	データ 1
data2	タイマのカウンタ値を格納するバッファへのポインタ

解説

パルス幅測定モードに設定されたタイマのカウンタ値を取得します。

[data1]

data1 には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数の端子を選択することができます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_TIOCA	TIOCiA 端子（i = 0～4）
RAPI_TIOCB	TIOCiB 端子（i = 0～4）
RAPI_TIOCC	TIOCiC 端子（i = 0，3，4）
RAPI_TIOCD	TIOCiD 端子（i = 0，3，4）

戻り値

タイマの指定が正しくない場合はRAPI_FALSE を返し、それ以外の場合はRAPI_TRUE を返します。

周辺機能

タイマ MTU2（パルス幅測定モード）

参照

__GetCaptureValue

備考

- 本 API 関数実行時の端子状態が“H”の場合（“L”の場合）、直前の“L”期間（“H”期間）のカウンタ値が data2 に格納されます。ただし、本 API 関数は端子状態の読み込みは行わないため、端子状態の判別処理については本 API 関数実行前にユーザ側で行って下さい。（下記プログラム例参照）
- 第 1 引数に上記以外のパラメータを設定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    unsigned short H_value,L_value;

    /* パルス幅測定モードに設定した MTU2 チャネル 0 においてパルス幅を取得 */
    if( PE.PRL.BIT.B0 == 1 ){    /* パルス幅測定用端子が 'H' の場合 */
        /* Low パルス幅を取得 */
        __GetPulseWidthMeasurementMode( RAPI_MTU2_0 | RAPI_TIOCA,
               &L_value );
    }
    else{                        /* パルス幅測定用端子が 'L' の場合 */
        /* High パルス幅を取得 */
        __GetPulseWidthMeasurementMode( RAPI_MTU2_0 | RAPI_TIOCA,
               &H_value );
    }
}
```

22) __CreateInputCapture

概要

＜インプットキャプチャモードの設定＞

Boolean __CreateInputCapture(unsigned long data, void *func)

data	設定データ
func	コールバック関数ポインタ

解説（1/6）

指定したタイマをインプットキャプチャモードに設定します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_MP_1	内部クロック：MPφ/1 でカウント
RAPI_MP_4	内部クロック：MPφ/4 でカウント
RAPI_MP_16	内部クロック：MPφ/16 でカウント
RAPI_MP_64	内部クロック：MPφ/64 でカウント
RAPI_MP_256_1	内部クロック：MPφ/256 でカウント（チャネル 1 の場合）
RAPI_MP_256_34	内部クロック：MPφ/256 でカウント （チャネル 3 または 4 の場合）
RAPI_MP_1024_2	内部クロック：MPφ/1024 でカウント（チャネル 2 の場合）
RAPI_MP_1024_34	内部クロック：MPφ/1024 でカウント （チャネル 3 または 4 の場合）
RAPI_EXTER_TCLKA_0	外部クロック：TCLKA 端子入力でカウント （チャネル 0～2 の場合）
RAPI_EXTER_TCLKB_0	外部クロック：TCLKB 端子入力でカウント （チャネル 0～2 の場合）
RAPI_EXTER_TCLKC_0	外部クロック：TCLKC 端子入力でカウント （チャネル 0 または 2 の場合）
RAPI_EXTER_TCLKD_0	外部クロック：TCLKD 端子入力でカウント （チャネル 0 の場合）
RAPI_EXTER_TCLKA_3	外部クロック：TCLKA 端子入力でカウント （チャネル 3 または 4 の場合）
RAPI_EXTER_TCLKB_3	外部クロック：TCLKB 端子入力でカウント （チャネル 3 または 4 の場合）
RAPI_RISING_EDGE	立ち上がりエッジでカウント
RAPI_FALLING_EDGE	立ち下がりエッジでカウント
RAPI_BOTH_EDGE	両エッジでカウント
【注】カウントソースに MPφ/1 を選択した場合、カウントエッジは 立ち上がりエッジのみの動作となります。	

解説（2/6）

RAPI_TCNT_CLEAR_DIS	TCNT クリアを禁止
RAPI_TCNT_CLEAR_TGRA	TGRA インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRB	TGRB インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRC	TGRC インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRD	TGRD インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRU	TGRU インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRV	TGRV インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_TGRW	TGRW インプットキャプチャで TCNT をクリア
RAPI_TCNT_CLEAR_OTHER	同期クリア実行中またはカウンタクリアの原因となる同期動作を行っている他のチャネルを指定
RAPI_OVERFLOW_ENA	オーバフロー割り込みを許可
RAPI_OVERFLOW_DIS	オーバフロー割り込みを禁止
RAPI_AD_START_REQ	TGRA のインプットキャプチャ／コンペアマッチによる A/D 変換開始要求の発生を許可
RAPI_NO_AD_START_REQ	TGRA のインプットキャプチャ／コンペアマッチによる A/D 変換開始要求の発生を禁止
RAPI_TIMER_INT_LV_0	割り込み優先レベル 0
RAPI_TIMER_INT_LV_1	割り込み優先レベル 1
RAPI_TIMER_INT_LV_2	割り込み優先レベル 2
RAPI_TIMER_INT_LV_3	割り込み優先レベル 3
RAPI_TIMER_INT_LV_4	割り込み優先レベル 4
RAPI_TIMER_INT_LV_5	割り込み優先レベル 5
RAPI_TIMER_INT_LV_6	割り込み優先レベル 6
RAPI_TIMER_INT_LV_7	割り込み優先レベル 7
RAPI_TIMER_INT_LV_8	割り込み優先レベル 8
RAPI_TIMER_INT_LV_9	割り込み優先レベル 9
RAPI_TIMER_INT_LV_10	割り込み優先レベル 10
RAPI_TIMER_INT_LV_11	割り込み優先レベル 11
RAPI_TIMER_INT_LV_12	割り込み優先レベル 12
RAPI_TIMER_INT_LV_13	割り込み優先レベル 13
RAPI_TIMER_INT_LV_14	割り込み優先レベル 14
RAPI_TIMER_INT_LV_15	割り込み優先レベル 15
RAPI_TIMER_SYNC	チャネル 0～4 を同期動作する
RAPI_TIMER_NO_SYNC	チャネル 0～4 を同期動作しない

解説（3/6）

●RAPI_MTU2_0 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64，
RAPI_EXTER_TCLKA_0，RAPI_EXTER_TCLKB_0，
RAPI_EXTER_TCLKC_0，RAPI_EXTER_TCLKD_0 } から
1 つを指定します。初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，
RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，
RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_TGRC，
RAPI_TCNT_CLEAR_TGRD，RAPI_TCNT_CLEAR_OTHER }
から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から
1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から
1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，
RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，
RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，
RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，
RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，
RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，
RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，
RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から
1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（4/6）

●RAPI_MTU2_1 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64，
RAPI_MP_256_1，RAPI_EXTER_TCLKA_0，
RAPI_EXTER_TCLKB_0 } から 1 つを指定します。
初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，
RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，
RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_OTHER }
から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から
1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から
1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，
RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，
RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，
RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，
RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，
RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，
RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，
RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から
1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（5/6）

●RAPI_MTU2_2 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64，RAPI_MP_1024_2，RAPI_EXTER_TCLKA_0，RAPI_EXTER_TCLKB_0，RAPI_EXTER_TCLKC_0 } から 1 つを指定します。初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_OTHER } から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI_OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（6/6）

●RAPI_MTU2_3 または RAPI_MTU2_4 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64，RAPI_MP_256_34，RAPI_MP_1024_34，RAPI_EXTER_TCLKA_3，RAPI_EXTER_TCLKB_3 } から 1 つを指定します。初期値は RAPI_MP_1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_TGRC，RAPI_TCNT_CLEAR_TGRD，RAPI_TCNT_CLEAR_OTHER } から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

（割り込みイネーブル） { RAPI_OVERFLOW_ENA，RAPI _OVERFLOW_DIS } から 1 つを指定します。初期値は RAPI_OVERFLOW_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

●RAPI_MTU2_5 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP_1，RAPI_MP_4，RAPI_MP_16，RAPI_MP_64 } から 1 つを指定します。初期値は RAPI_MP_1 です。

（カウントクリアソース） { RAPI_TCNT_CLEAR_DIS，RAPI_TCNT_CLEAR_TGRU，RAPI_TCNT_CLEAR_TGRV，RAPI_TCNT_CLEAR_TGRW } から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

●同期動作設定時に指定するパラメータについて：

チャネル 0～4 のいずれかで同期動作を設定する場合、RAPI_TIMER_SYNC を指定して下さい。同期動作を設定しない場合（各チャネルで独立に動作）は、RAPI_TIMER_NO_SYNC を指定して下さい。初期値は RAPI_TIMER_NO_SYNC です。

[func]

func には、コールバック関数を指すポインタを指定して下さい。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。

戻り値

タイマの指定が正しくない場合はRAPI_FALSEを返し、それ以外の場合はRAPI_TRUEを返します。

周辺機能

タイマ MTU2（インプットキャプチャモード）

参照

__EnableInputCapture，__DestroyInputCapture，__GetCaptureValue

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

/* コールバック関数宣言 */
void TimerIntFunc( void );

void func( void )
{
   /* MTU2 チャネル 0 をインプットキャプチャモードに設定する */
   __CreateInputCapture( RAPI_MTU2_0 | RAPI_MP_4 | RAPI_BOTH_EDGE |
          RAPI_TCNT_CLEAR_TGRB | RAPI_OVERFLOW_DIS |
          RAPI_TIMER_INT_LV_0 | RAPI_TIMER_NO_SYNC, TimerIntFunc );
}
```

23) __SetInputCapturePin

概要

＜インプットキャプチャ用入力端子の設定＞

Boolean __SetInputCapturePin(unsigned long data, void *func)

data	設定データ
func	コールバック関数ポインタ

解説（1/3）

指定した端子をインプットキャプチャ用の入力端子に設定します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_TIOCA	TIOCiA 端子（i = 0～4）
RAPI_TIOCB	TIOCiB 端子（i = 0～4）
RAPI_TIOCC	TIOCiC 端子（i = 0，3，4）
RAPI_TIOCD	TIOCiD 端子（i = 0，3，4）
RAPI_RISING_CAP_0	立ち上がりエッジでインプットキャプチャ（チャネル 0～4 の場合）
RAPI_FALLING_CAP_0	立ち下がりエッジでインプットキャプチャ（チャネル 0～4 の場合）
RAPI_BOTH_CAP_0	両エッジでインプットキャプチャ（チャネル 0～4 の場合）
RAPI_RISING_CAP_5	立ち上がりエッジでインプットキャプチャ（チャネル 5 の場合）
RAPI_FALLING_CAP_5	立ち下がりエッジでインプットキャプチャ（チャネル 5 の場合）
RAPI_BOTH_CAP_5	両エッジでインプットキャプチャ（チャネル 5 の場合）
RAPI_TCNT1_CAP	キャプチャ入力元はチャネル 1 のカウントクロックで、 TCNT_1 のカウントアップ／カウントダウンで インプットキャプチャ（チャネル 0 の場合）
RAPI_TGRC0_CAP	TGRC_0 のコンペアマッチ／インプットキャプチャの発生で インプットキャプチャ（チャネル 1：TIOC1B 端子の場合）
RAPI_TGRA0_CAP	TGRA_0 のコンペアマッチ／インプットキャプチャの発生で インプットキャプチャ（チャネル 1：TIOC1A 端子の場合）
RAPI_CAPTURE_ENA	TGRi インプットキャプチャ割り込みを許可 （i = A，B，C，D，U，V，W）
RAPI_CAPTURE_DIS	インプットキャプチャ割り込みを禁止
RAPI_TIMER_INT_LV_0	割り込み優先レベル 0
RAPI_TIMER_INT_LV_1	割り込み優先レベル 1
RAPI_TIMER_INT_LV_2	割り込み優先レベル 2
RAPI_TIMER_INT_LV_3	割り込み優先レベル 3
RAPI_TIMER_INT_LV_4	割り込み優先レベル 4
RAPI_TIMER_INT_LV_5	割り込み優先レベル 5
RAPI_TIMER_INT_LV_6	割り込み優先レベル 6
RAPI_TIMER_INT_LV_7	割り込み優先レベル 7
RAPI_TIMER_INT_LV_8	割り込み優先レベル 8
RAPI_TIMER_INT_LV_9	割り込み優先レベル 9
RAPI_TIMER_INT_LV_10	割り込み優先レベル 10
RAPI_TIMER_INT_LV_11	割り込み優先レベル 11
RAPI_TIMER_INT_LV_12	割り込み優先レベル 12
RAPI_TIMER_INT_LV_13	割り込み優先レベル 13
RAPI_TIMER_INT_LV_14	割り込み優先レベル 14
RAPI_TIMER_INT_LV_15	割り込み優先レベル 15

解説（2/3）

●RAPI_MTU2_0，RAPI_MTU2_3 または RAPI_MTU2_4 選択時に指定可能なパラメータ：

（キャプチャチャネル） { RAPI_TIOCA，RAPI_TIOCB，RAPI_TIOCC，RAPI_TIOCD } から 1 つを指定します。

（キャプチャエッジ） { RAPI_RISING_CAP_0，RAPI_FALLING_CAP_0，RAPI_BOTH_CAP_0，RAPI_TGRC0_CAP，RAPI_TGRA0_CAP } から 1 つを指定します。

（割り込みイネーブル） { RAPI_CAPTURE_ENA，RAPI_CAPTURE_DIS } から 1 つを指定します。
初期値は RAPI_CAPTURE_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

●RAPI_MTU2_1 または RAPI_MTU2_2 選択時に指定可能なパラメータ：

（キャプチャチャネル） { RAPI_TIOCA，RAPI_TIOCB } から 1 つを指定します。

（キャプチャエッジ） { RAPI_RISING_CAP_0，RAPI_FALLING_CAP_0，RAPI_BOTH_CAP_0，RAPI_TGRC0_CAP，RAPI_TGRA0_CAP } から 1 つを指定します。

（割り込みイネーブル） { RAPI_CAPTURE_ENA，RAPI_CAPTURE_DIS } から 1 つを指定します。初期値は RAPI_CAPTURE_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

解説（3/3）

●RAPI_MTU2_5 選択時に指定可能なパラメータ：

（キャプチャチャネル） { RAPI_MTU2_5U，RAPI_ MTU2_5V，RAPI_ MTU2_5W } から 1 つを指定します。

（キャプチャエッジ） { RAPI_RISING_CAP_5，RAPI_FALLING_CAP_5，RAPI_BOTH_CAP_5，RAPI_TCNT1_CAP } から 1 つを指定します。

（割り込みイネーブル） { RAPI_CAPTURE_ENA，RAPI_CAPTURE_DIS } から 1 つを指定します。初期値は RAPI_CAPTURE_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。初期値は RAPI_TIMER_INT_LV_0 です。

[func]
func には、コールバック関数を指すポインタを指定して下さい。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。

戻り値

タイマの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（インプットキャプチャモード）

参照

__EnableInputCapture，__DestroyInputCapture，__GetCaptureValue

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

/* コールバック関数宣言 */
void TimerIntFunc( void );

void func( void )
{
   /* MTU2 チャネル 0 に対し、TGRA をインプットキャプチャ用に設定 */
   __SetInputCapturePin( RAPI_MTU2_0 | RAPI_TIOCA |
          RAPI_RISING_CAPTURE_0 | RAPI_NO_AD_START_REQ |
          RAPI_CAPTURE_ENA | RAPI_TIMER_INT_LV_3, TimerIntFunc );
}
```

24) __EnableInputCapture

概要

＜インプットキャプチャモードの動作制御＞

Boolean __EnableInputCapture(unsigned long data)

data	設定データ

解説

インプットキャプチャモードに設定されたタイマの動作開始／動作停止を制御します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数のタイマを動作制御できます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_TIMER_ON	インプットキャプチャモードに設定されているタイマを動作開始に設定
RAPI_TIMER_OFF	インプットキャプチャモードに設定されているタイマを動作停止に設定

戻り値

タイマの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（インプットキャプチャモード）

参照

__CreateInputCapture，__DestroyInputCapture，__GetCaptureValue

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* インプットキャプチャモードに設定した MTU2 チャネル 1 およびチャネル 2 に対し、
       タイマ動作を開始 */
    __EnableInputCapture( RAPI_MTU2_1 | RAPI_MTU2_2 | RAPI_TIMER_ON );
}
```

25) __DestroyInputCapture

概要

＜インプットキャプチャモードの設定クリア＞

Boolean __DestroyInputCapture(unsigned long data)

data	設定データ

解説

インプットキャプチャモードに設定されたタイマの設定をクリアします。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数のタイマの設定をクリアすることができます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU20_4	MTU2 チャネル 0～4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_MTU2_5	MTU2 チャネル 5U，5V，および 5W
RAPI_MTU2_ALL	すべての MTU2 チャネル

戻り値

タイマの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（インプットキャプチャモード）

参照

__CreateInputCapture，__EnableInputCapture，__GetCaptureValue

備考

引数に上記以外のパラメータを設定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* MTU2 チャネル 2 およびチャネル 3 に対し、インプットキャプチャモードの設定を
       クリア */
    __DestroyInputCapture( RAPI_MTU2_2 | RAPI_MTU2_3 );
}
```

26) __GetCaptureValue

概要

＜インプットキャプチャモードのカウンタ値取得＞

Boolean __GetCaptureValue(unsigned long data1, unsigned short *data2)

data1	設定データ 1
data2	タイマのカウンタ値を格納するバッファへのポインタ

解説

インプットキャプチャモードに設定されたタイマのカウンタ値を取得します。

[data1]

data1 には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数の端子を設定することができます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_TIOCA	TIOCiA 端子（i = 0～4）
RAPI_TIOCB	TIOCiB 端子（i = 0～4）
RAPI_TIOCC	TIOCiC 端子（i = 0，3，4）
RAPI_TIOCD	TIOCiD 端子（i = 0，3，4）

戻り値

タイマの指定が正しくない場合はRAPI_FALSE を返し、それ以外の場合はRAPI_TRUE を返します。

周辺機能

タイマ MTU2（インプットキャプチャモード）

参照

__CreateInputCapture，__EnableInputCapture，__DestroyInputCapture

備考

第 1 引数に上記以外のパラメータを設定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    unsigned short data;

    /* インプットキャプチャモードに設定した MTU2 チャネル 5U のカウンタ値を取得 */
    __GetCaptureValue( RAPI_MTU2_5U, &data );
}
```

27) __CreateOutputCompare

概要

＜アウトプットコンペアモードの設定＞

Boolean __CreateOutputCompare(unsigned long data)

data	設定データ

解説（1/3）

指定したタイマをアウトプットコンペアモードに設定します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MP1	内部クロック：MPφ/1 でカウント
RAPI_MP4	内部クロック：MPφ/4 でカウント
RAPI_MP16	内部クロック：MPφ/16 でカウント
RAPI_MP64	内部クロック：MPφ/64 でカウント
RAPI_MP256_1	内部クロック：MPφ/256 でカウント（チャネル 1 の場合）
RAPI_MP256_34	内部クロック：MPφ/256 でカウント （チャネル 3 または 4 の場合）
RAPI_MP1024_2	内部クロック：MPφ/1024 でカウント（チャネル 2 の場合）
RAPI_MP1024_34	内部クロック：MPφ/1024 でカウント （チャネル 3 または 4 の場合）
RAPI_EXTER_TCLKA_0	外部クロック：TCLKA 端子入力でカウント （チャネル 0～2 の場合）
RAPI_EXTER_TCLKB_0	外部クロック：TCLKB 端子入力でカウント （チャネル 0～2 の場合）
RAPI_EXTER_TCLKC_0	外部クロック：TCLKC 端子入力でカウント （チャネル 0 または 2 の場合）
RAPI_EXTER_TCLKD_0	外部クロック：TCLKD 端子入力でカウント （チャネル 0 または 2 の場合）
RAPI_EXTER_TCLKA_3	外部クロック：TCLKA 端子入力でカウント （チャネル 3 または 4 の場合）
RAPI_EXTER_TCLKB_3	外部クロック：TCLKB 端子入力でカウント （チャネル 3 または 4 の場合）
RAPI_RISING_EDGE	立ち上がりエッジでカウント
RAPI_FALLING_EDGE	立ち下がりエッジでカウント
RAPI_BOTH_EDGE	両エッジでカウント
【注】カウントソースに MPφ/1 を選択した場合、カウントエッジは 立ち上がりエッジのみの動作となります。	
RAPI_TCNT_CLEAR_TGRA	TGRA コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRB	TGRB コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRC	TGRC コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_TGRD	TGRD コンペアマッチで TCNT をクリア
RAPI_TCNT_CLEAR_OTHER	同期クリア実行中またはカウンタクリアの原因となる同期動作を行っている他のチャネルを指定
RAPI_AD_START_REQ	TGRA のインプットキャプチャ／コンペアマッチによる A/D 変換開始要求の発生を許可
RAPI_NO_AD_START_REQ	TGRA のインプットキャプチャ／コンペアマッチによる A/D 変換開始要求の発生を禁止
RAPI_TIMER_SYNC	チャネル 0～4 を同期動作する
RAPI_TIMER_NO_SYNC	チャネル 0～4 を同期動作しない

解説（2/3）

●RAPI_MTU2_0 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，RAPI_MP64，
RAPI_EXTER_TCLKA_0，RAPI_EXTER_TCLKB_0，
RAPI_EXTER_TCLKC_0，RAPI_EXTER_TCLKD_0 } から
1 つを指定します。初期値は RAPI_MP1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，
RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB，
RAPI_TCNT_CLEAR_TGRC，RAPI_TCNT_CLEAR_TGRD }
から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から
1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

●RAPI_MTU2_1 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，RAPI_MP64，
RAPI_MP256_1，RAPI_EXTER_TCLKA_0，
RAPI_EXTER_TCLKB_0 } から 1 つを指定します。
初期値は RAPI_MP1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，
RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB }
から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から
1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

●RAPI_MTU2_2 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，RAPI_MP64，
RAPI_MP1024_2，RAPI_EXTER_TCLKA_0，
RAPI_EXTER_TCLKB_0，RAPI_EXTER_TCLKC_0 } から
1 つを指定します。初期値は RAPI_MP1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，
RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB }
から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から
1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

解説（3/3）

●RAPI_MTU2_3 または RAPI_MTU2_4 選択時に指定可能なパラメータ：

（カウントソース） { RAPI_MP1，RAPI_MP4，RAPI_MP16，RAPI_MP64，RAPI_MP256_34，RAPI_MP1024_34，RAPI_EXTER_TCLKA_3，RAPI_EXTER_TCLKB_3 } から 1 つを指定します。初期値は RAPI_MP1 です。

（カウントエッジ） { RAPI_RISING_EDGE，RAPI_FALLING_EDGE，RAPI_BOTH_EDGE } から 1 つを指定します。
初期値は RAPI_RISING_EDGE です。
【注】カウントソースに RAPI_MP_1 を指定した場合、
カウントエッジは初期値のみの動作となります。

（カウントクリアソース） { RAPI_TCNT_CLEAR_TGRA，RAPI_TCNT_CLEAR_TGRB，RAPI_TCNT_CLEAR_TGRC，RAPI_TCNT_CLEAR_TGRD } から 1 つを指定します。初期値は RAPI_TCNT_CLEAR_DIS です。

（A/D 変換開始要求） { RAPI_AD_START_REQ，RAPI_NO_AD_START_REQ } から 1 つを指定します。初期値は RAPI_NO_AD_START_REQ です。

●同期動作設定時に指定するパラメータについて：

チャネル 0～4 のいずれかで同期動作を設定する場合、RAPI_TIMER_SYNC を指定して下さい。同期動作を設定しない場合（各チャネルで独立に動作）は、RAPI_TIMER_NO_SYNC を指定して下さい。初期値は RAPI_TIMER_NO_SYNC です。

戻り値

タイマの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（アウトプットコンペアモード）

参照

__CreatePWM

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

/* コールバック関数宣言 */
void TimerIntFunc( void );

void func( void )
{
   /* MTU2 チャネル 0 をアウトプットコンペアモードに設定 */
   __CreateOutputCompare( RAPI_MTU2_0 | RAPI_MP_4 |
          RAPI_FALLING_EDGE | RAPI_TCNT_CLEAR_TGRA |
          RAPI_TIMER_NO_SYNC );
}
```

28) __SetOutputPin

概要

＜アウトプットコンペア用タイマジェネラルレジスタの設定＞

Boolean __SetOutputPin(unsigned long data1, unsigned short data2, void *func)

data1	設定データ 1
data2	設定データ 2
func	コールバック関数ポインタ

解説（1/2）

指定したチャネルのタイマジェネラルレジスタをアウトプットコンペア用に設定します。

[data1]

data1 には、以下のパラメータを指定して下さい。複数のパラメータを設定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_TGRA	タイマジェネラルレジスタ A
RAPI_TGRB	タイマジェネラルレジスタ B
RAPI_TGRC	タイマジェネラルレジスタ C
RAPI_TGRD	タイマジェネラルレジスタ D
RAPI_OUTPUT_RETAIN	出力保持
RAPI_OUTPUT_0_0	初期出力 0，コンペアマッチで 0 を出力
RAPI_OUTPUT_0_1	初期出力 0，コンペアマッチで 1 を出力
RAPI_OUTPUT_0_TOG	初期出力 0，コンペアマッチでトグル出力
RAPI_OUTPUT_1_0	初期出力 1，コンペアマッチで 0 を出力
RAPI_OUTPUT_1_1	初期出力 1，コンペアマッチで 1 を出力
RAPI_OUTPUT_1_TOG	初期出力 1，コンペアマッチでトグル出力
RAPI_COMPARE_MATCH_ENA	コンペアマッチ割り込みを許可
RAPI_COMPARE_MATCH_DIS	コンペアマッチ割り込みを禁止
RAPI_TIMER_INT_LV_0	割り込み優先レベル 0
RAPI_TIMER_INT_LV_1	割り込み優先レベル 1
RAPI_TIMER_INT_LV_2	割り込み優先レベル 2
RAPI_TIMER_INT_LV_3	割り込み優先レベル 3
RAPI_TIMER_INT_LV_4	割り込み優先レベル 4
RAPI_TIMER_INT_LV_5	割り込み優先レベル 5
RAPI_TIMER_INT_LV_6	割り込み優先レベル 6
RAPI_TIMER_INT_LV_7	割り込み優先レベル 7
RAPI_TIMER_INT_LV_8	割り込み優先レベル 8
RAPI_TIMER_INT_LV_9	割り込み優先レベル 9
RAPI_TIMER_INT_LV_10	割り込み優先レベル 10
RAPI_TIMER_INT_LV_11	割り込み優先レベル 11
RAPI_TIMER_INT_LV_12	割り込み優先レベル 12
RAPI_TIMER_INT_LV_13	割り込み優先レベル 13
RAPI_TIMER_INT_LV_14	割り込み優先レベル 14
RAPI_TIMER_INT_LV_15	割り込み優先レベル 15

解説（2/2）

●RAPI_MTU2_0，RAPI_MTU2_3 または RAPI_MTU2_4 選択時に指定可能なパラメータ：

（タイマジェネラルレジスタ） { RAPI_TGRA，RAPI_TGRB，RAPI_TGRC，RAPI_TGRD } から 1 つを指定します。

（カウント出力） { RAPI_OUTPUT_RETAIN，RAPI_OUTPUT_0_0，RAPI_OUTPUT_0_1，RAPI_OUTPUT_0_TOG，RAPI_OUTPUT_1_0，RAPI_OUTPUT_1_1，RAPI_OUTPUT_1_TOG } から 1 つを指定します。
初期値は RAPI_OUTPUT_RETAIN です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA，RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。
初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。
初期値は RAPI_TIMER_INT_LV_0 です。

●RAPI_MTU2_1 または RAPI_MTU2_2 選択時に指定可能なパラメータ：

（タイマジェネラルレジスタ） { RAPI_TGRA，RAPI_TGRB } から 1 つを指定します。

（カウント出力） { RAPI_OUTPUT_RETAIN，RAPI_OUTPUT_0_0，RAPI_OUTPUT_0_1，RAPI_OUTPUT_0_TOG，RAPI_OUTPUT_1_0，RAPI_OUTPUT_1_1，RAPI_OUTPUT_1_TOG } から 1 つを指定します。
初期値は RAPI_OUTPUT_RETAIN です。

（割り込みイネーブル） { RAPI_COMPARE_MATCH_ENA，RAPI_COMPARE_MATCH_DIS } から 1 つを指定します。
初期値は RAPI_COMPARE_MATCH_DIS です。

（割り込み優先レベル） { RAPI_TIMER_INT_LV_0，RAPI_TIMER_INT_LV_1，RAPI_TIMER_INT_LV_2，RAPI_TIMER_INT_LV_3，RAPI_TIMER_INT_LV_4，RAPI_TIMER_INT_LV_5，RAPI_TIMER_INT_LV_6，RAPI_TIMER_INT_LV_7，RAPI_TIMER_INT_LV_8，RAPI_TIMER_INT_LV_9，RAPI_TIMER_INT_LV_10，RAPI_TIMER_INT_LV_11，RAPI_TIMER_INT_LV_12，RAPI_TIMER_INT_LV_13，RAPI_TIMER_INT_LV_14，RAPI_TIMER_INT_LV_15 } から 1 つを指定します。
初期値は RAPI_TIMER_INT_LV_0 です。

[data2]
data2 には、出力カウンタ値（16 ビット）を指定して下さい。

[func]
func には、コールバック関数を指すポインタを指定して下さい。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。既に指定したコールバック関数を保持する場合は、RAPI_HOLD を指定して下さい。この場合、data2 の値のみ変更されます。

戻り値

チャネルの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（アウトプットコンペアモード）

参照

__EnableOutputCompare，__DestroyOutputCompare

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

/* コールバック関数宣言 */
void TimerIntBFunc( void );
void TimerIntAFunc( void );

void func( void )
{
   /* MTU2 チャネル 1 に対し、デューティレジスタおよび周期レジスタを設定 */
   __SetOutputPin( RAPI_MTU2_1 | RAPI_TGRB | RAPI_OUTPUT_0_0 |
          RAPI_COMPARE_MATCH_ENA | RAPI_TIMER_INT_LV_3,
          50000, TimerIntBFunc );

   __SetOutputPin( RAPI_MTU2_1 | RAPI_TGRA | RAPI_OUTPUT_0_1 |
          RAPI_COMPARE_MATCH_ENA | RAPI_TIMER_INT_LV_3,
          30000, TimerIntAFunc );
}
```

29) __EnableOutputCompare

概要

＜アウトプットコンペアモードの動作制御＞

Boolean __EnableOutputCompare(unsigned long data)

data	設定データ

解説

アウトプットコンペアモードに設定されたタイマの動作開始／動作停止を制御します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。これにより、一度に複数のタイマを動作制御できます。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU20_4	MTU2 チャネル 0～4
RAPI_TIMER_ON	アウトプットコンペアモードに設定されているタイマを動作開始に設定
RAPI_TIMER_OFF	アウトプットコンペアモードに設定されているタイマを動作停止に設定

戻り値

タイマの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

タイマ MTU2（アウトプットコンペアモード）

参照

__EnableTimer

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* アウトプットコンペアモードに設定した MTU2 チャネル 0 に対し、
       タイマ動作を開始 */
    __EnableOutputCompare( RAPI_MTU2_0 | RAPI_MTU2_1 | RAPI_TIMER_ON );
}
```

30) __DestroyOutputCompare

概要

＜アウトプットコンペアモードの設定クリア＞

Boolean __DestroyOutputCompare(unsigned long data)

data	設定データ

解説

アウトプットコンペアモードに設定されたタイマの設定をクリアします。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU20_4	MTU2 チャネル 0～4

戻り値

タイマの指定が正しくない場合はRAPI_FALSE を返し、それ以外の場合はRAPI_TRUE を返します。

周辺機能

タイマ MTU2（アウトプットコンペアモード）

参照

__DestroyTimer

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* MTU2 チャネル 1 およびチャネル 2 に対し、アウトプットコンペアモードの設定を
       クリア */
    __DestroyOutputCompare( RAPI_MTU2_1 | RAPI_MTU2_2 );
}
```

31) __GetTimerFlag

概要

＜タイマのステータス取得＞

Boolean __GetTimerFlag(unsigned long data1, unsigned char *data2)

data1	設定データ 1
data2	タイマステータスフラグを格納するバッファへのポインタ

解説

指定したタイマの各種ステータスフラグを取得します。

[data1]

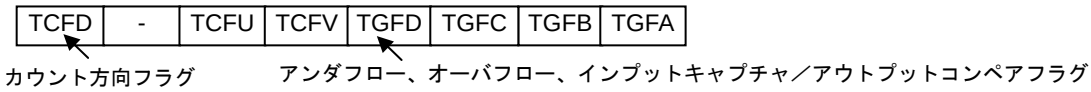
data1 には、以下のパラメータを指定して下さい。複数のフラグを取得する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_CMT_0	CMT チャネル 0
RAPI_CMT_1	CMT チャネル 1
RAPI_TGFA	インプットキャプチャ／アウトプットコンペアフラグ A
RAPI_TGFB	インプットキャプチャ／アウトプットコンペアフラグ B
RAPI_TGFC	インプットキャプチャ／アウトプットコンペアフラグ C
RAPI_TGFD	インプットキャプチャ／アウトプットコンペアフラグ D
RAPI_TCFD	カウント方向フラグ
RAPI_TCFV	オーバフローフラグ
RAPI_TCFU	アンダフローフラグ
RAPI_TGFE	コンペアマッチフラグ E
RAPI_TGFF	コンペアマッチフラグ F
RAPI_CMF	コンペアマッチ／インプットキャプチャフラグ

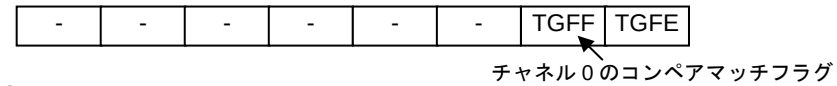
【注】MTU2 チャネル 5U，5V，5W 選択時は、それぞれのコンペアマッチ／インプットキャプチャフラグ（CMFU5，CMFV5，CMFW5）を取得します。

タイマ MTU2 ステータスフラグを格納しているタイマステータスレジスタ（TSR）の構成を以下に示します。

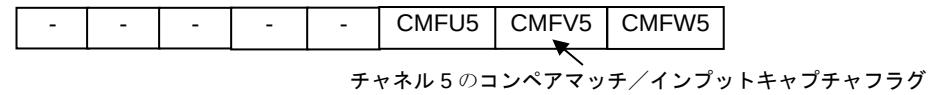
●TSR_0～TSR_4：



●TSR2_0：



●TSR_5：



戻り値

タイマの指定が正しくない場合はRAPI_FALSEを返し、それ以外の場合はRAPI_TRUEを返します。

周辺機能

タイマ MTU2

参照

__ClearTimerFlag

備考

第1引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    unsigned char data;

    /* MTU2 チャネル 0 に対し、カウント方向フラグおよびオーバフローフラグを取得 */
    __GetTimerFlag( RAPI_MTU2_0 | RAPI_TCFV | RAPI_TGFD, &data );
}
```

32) __ClearTimerFlag

概要

＜タイマのステータスクリア＞

Boolean __ClearTimerFlag(unsigned long data)

data	設定データ

解説

指定したタイマの各種ステータスフラグをクリアします。

[data]

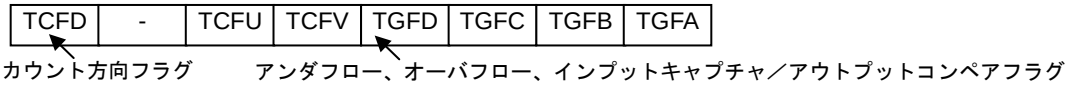
data には、以下のパラメータを指定して下さい。複数のフラグをクリアする場合は、パラメータ間に “|” 記号を入れて下さい。

RAPI_MTU2_0	MTU2 チャネル 0
RAPI_MTU2_1	MTU2 チャネル 1
RAPI_MTU2_2	MTU2 チャネル 2
RAPI_MTU2_3	MTU2 チャネル 3
RAPI_MTU2_4	MTU2 チャネル 4
RAPI_MTU2_5U	MTU2 チャネル 5U
RAPI_MTU2_5V	MTU2 チャネル 5V
RAPI_MTU2_5W	MTU2 チャネル 5W
RAPI_CMT_0	CMT チャネル 0
RAPI_CMT_1	CMT チャネル 1
RAPI_TGFA	インプットキャプチャ／アウトプットコンペアフラグ A
RAPI_TGFB	インプットキャプチャ／アウトプットコンペアフラグ B
RAPI_TGFC	インプットキャプチャ／アウトプットコンペアフラグ C
RAPI_TGFD	インプットキャプチャ／アウトプットコンペアフラグ D
RAPI_TCFV	オーバフローフラグ
RAPI_TCFU	アンダフローフラグ
RAPI_TGFE	コンペアマッチフラグ E
RAPI_TGFF	コンペアマッチフラグ F
RAPI_CMF	コンペアマッチ／インプットキャプチャフラグ

【注】MTU2 チャネル 5U，5V，5W 選択時は、それぞれのコンペアマッチ／インプットキャプチャフラグ（CMFU5，CMFV5，CMFW5）をクリアします。

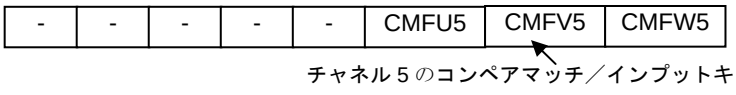
タイマ MTU2 ステータスフラグを格納しているタイマステータスレジスタ（TSR）の構成を以下に示します。

●TSR_0～TSR_4：



●TSR2_0：

●TSR_5：



戻り値

タイマの指定が正しくない場合はRAPI_FALSEを返し、それ以外の場合はRAPI_TRUEを返します。

周辺機能

タイマ MTU2

参照

__GetTimerFlag

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_timer_sh_7125.h"

void func( void )
{
    /* MTU2 チャネル 0 に対し、カウント方向フラグおよびオーバフローフラグをクリア */
    __ClearTimerFlag( RAPI_MTU2_0 | RAPI_TCFV | RAPI_TGFD );
}
```

4.2.3. I/Oポート

1) __SetIOPort

概要

＜I/O ポートの設定＞

Boolean __SetIOPort(unsigned long data)

data	設定データ

解説

指定した I/O ポートの動作条件を設定します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

●ポートの指定：（いずれか 1 つ）

RAPI_FUNC_A	ポート A
RAPI_FUNC_B_L	ポート B の下位 16 ビット
RAPI_FUNC_B_H	ポート B の上位 16 ビット
RAPI_FUNC_E	ポート E
RAPI_FUNC_F	ポート F

【注】ポート F は、8 ビットの入力専用ポートです。

●ビットの指定：（複数指定可）

RAPI_BIT_0	指定したポートのビット 0
RAPI_BIT_1	指定したポートのビット 1
RAPI_BIT_2	指定したポートのビット 2
RAPI_BIT_3	指定したポートのビット 3
RAPI_BIT_4	指定したポートのビット 4
RAPI_BIT_5	指定したポートのビット 5
RAPI_BIT_6	指定したポートのビット 6
RAPI_BIT_7	指定したポートのビット 7
RAPI_BIT_8	指定したポートのビット 8
RAPI_BIT_9	指定したポートのビット 9
RAPI_BIT_10	指定したポートのビット 10
RAPI_BIT_11	指定したポートのビット 11
RAPI_BIT_12	指定したポートのビット 12
RAPI_BIT_13	指定したポートのビット 13
RAPI_BIT_14	指定したポートのビット 14
RAPI_BIT_15	指定したポートのビット 15
RAPI_LOWER_8_BITS	指定したポートの下位 8 ビット
RAPI_HIGHER_8_BITS	指定したポートの上位 8 ビット
RAPI_16_BITS	指定したポートの 16 ビット

●端子機能の設定：（いずれか 1 つ）

RAPI_FUNC_IO	I/O ポート
RAPI_FUNC_MTU2	タイマ MTU2
RAPI_FUNC_SCI	シリアル通信インタフェース
RAPI_FUNC_AD	A/D コンバータ
RAPI_FUNC_POE	ポートアウトプットイネーブル
RAPI_FUNC_INT	外部割り込み

●ポートの入出力方向設定：（いずれか 1 つ）

RAPI_PORT_INPUT	指定したポートを入力に設定
RAPI_PORT_OUTPUT	指定したポートを出力に設定

戻り値

I/O ポートの設定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

I/O ポート

参照

__ReadIOPort, __WriteIOPort

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_io_port_sh_7125.h"

void func( void )
{
    /* ポート A の下位 8 ビットを出力に設定 */
    __SetIOPort( RAPI_FUNC_A | RAPI_LOWER_8_BITS | RAPI_FUNC_IO |
           RAPI_PORT_OUTPUT );
}
```

2) __ReadIOPort

概要

＜I/O ポートからの読み出し＞

Boolean __ReadIOPort(unsigned long data1, unsigned short *data2)

data1	設定データ 1
data2	I/O ポートから読み出した値を格納する変数へのポインタ

解説

指定した I/O ポートの値を取得します。

[data1]

data1 には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

●ポートの指定：（いずれか 1 つ）

RAPI_FUNC_A	ポート A
RAPI_FUNC_B_L	ポート B の下位 16 ビット
RAPI_FUNC_B_H	ポート B の上位 16 ビット
RAPI_FUNC_E	ポート E
RAPI_FUNC_F	ポート F

【注】ポート F は、8 ビットの入力専用ポートです。

●ビットの指定：（複数指定可）

RAPI_BIT_0	指定したポートのビット 0
RAPI_BIT_1	指定したポートのビット 1
RAPI_BIT_2	指定したポートのビット 2
RAPI_BIT_3	指定したポートのビット 3
RAPI_BIT_4	指定したポートのビット 4
RAPI_BIT_5	指定したポートのビット 5
RAPI_BIT_6	指定したポートのビット 6
RAPI_BIT_7	指定したポートのビット 7
RAPI_BIT_8	指定したポートのビット 8
RAPI_BIT_9	指定したポートのビット 9
RAPI_BIT_10	指定したポートのビット 10
RAPI_BIT_11	指定したポートのビット 11
RAPI_BIT_12	指定したポートのビット 12
RAPI_BIT_13	指定したポートのビット 13
RAPI_BIT_14	指定したポートのビット 14
RAPI_BIT_15	指定したポートのビット 15
RAPI_LOWER_8_BITS	指定したポートの下位 8 ビット
RAPI_HIGHER_8_BITS	指定したポートの上位 8 ビット
RAPI_16_BITS	指定したポートの 16 ビット

●データ読み出し時のアクセスサイズの指定：（いずれか 1 つ）

RAPI_BITS_OPE	ビット操作モード
RAPI_BYTE_OPE	バイト操作モード
RAPI_WORD_OPE	ワード操作モード

●読み出し対象レジスタの指定：（いずれか 1 つ）

RAPI_DATA_REGISTER	データ格納用レジスタ（データレジスタ）
RAPI_PORT_REGISTER	端子状態読み出し用レジスタ（ポートレジスタ）

- ポート F 選択時に指定可能な対象レジスタ：
  ポート F は、8 ビットの入力専用ポートです。
  読み出し対象レジスタは、RAPI_DATA_REGISTER のみ選択可能です。

戻り値

I/O ポートの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

I/O ポート

参照

__SetIOPort，__WriteIOPort

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_io_port_sh_7125.h"

unsigned short data[2];

void func( void )
{
    /* ポート A のビット 1 およびビット 3 の値を取得 */
    __ReadIOPort( RAPI_FUNC_A | RAPI_BIT_1 | RAPI_BIT_3 |
           RAPI_PORT_REGISTER | RAPI_BITS_OPE, &data );
}
```

3) __WriteIOPort

概要

＜I/O ポートへの書き込み＞

Boolean __WriteIOPort(unsigned long data1, unsigned short data2)

data1	設定データ 1
data2	I/O ポートへ書き込むデータ

解説（1/2）

指定した I/O ポートにデータを書き込みます。

[data1]

data1 には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

●ポートの指定：（いずれか 1 つ）

RAPI_FUNC_A	ポート A
RAPI_FUNC_B_L	ポート B の下位 16 ビット
RAPI_FUNC_B_H	ポート B の上位 16 ビット
RAPI_FUNC_E	ポート E
RAPI_FUNC_F	ポート F

●ビットの指定：（複数指定可）

RAPI_BIT_0	指定したポートのビット 0
RAPI_BIT_1	指定したポートのビット 1
RAPI_BIT_2	指定したポートのビット 2
RAPI_BIT_3	指定したポートのビット 3
RAPI_BIT_4	指定したポートのビット 4
RAPI_BIT_5	指定したポートのビット 5
RAPI_BIT_6	指定したポートのビット 6
RAPI_BIT_7	指定したポートのビット 7
RAPI_BIT_8	指定したポートのビット 8
RAPI_BIT_9	指定したポートのビット 9
RAPI_BIT_10	指定したポートのビット 10
RAPI_BIT_11	指定したポートのビット 11
RAPI_BIT_12	指定したポートのビット 12
RAPI_BIT_13	指定したポートのビット 13
RAPI_BIT_14	指定したポートのビット 14
RAPI_BIT_15	指定したポートのビット 15
RAPI_LOWER_8_BITS	指定したポートの下位 8 ビット
RAPI_HIGHER_8_BITS	指定したポートの上位 8 ビット
RAPI_16_BITS	指定したポートの 16 ビット

●データ書き込み時のアクセスサイズの指定：（いずれか 1 つ）

RAPI_BITS_OPE	ビット操作モード
RAPI_BYTE_OPE	バイト操作モード
RAPI_WORD_OPE	ワード操作モード

解説（2/2）

[data2]
data2 には、I/O ポートへ書き込む値を設定して下さい。

●RAPI_BITS_OPE 選択時に指定可能なパラメータ：（いずれか 1 つ）

RAPI_L	1 ビットまたは複数ビットに 0 を書き込む
RAPI_H	1 ビットまたは複数ビットに 1 を書き込む

【注】RAPI_BITS_OPE 選択時は、指定したビットすべてに対し同じ値（RAPI_L または RAPI_H）を書き込みます。

●RAPI_BYTE_OPE または RAPI_WORD_OPE 選択時に指定可能な値：
指定したポートに書き込む値をバイトサイズまたはワードサイズで指定して下さい。

戻り値

I/O ポートの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

I/O ポート

参照

__SetIOPort，__ReadIOPort

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_io_port_sh_7125.h"

void func( void )
{
    /* ポート A のビット 0～7 にデータ H'AA を書き込む */
    __WriteIOPort( RAPI_FUNC_A | RAPI_LOWER_8_BITS | RAPI_BYTE_OPE, 0xAA );
}
```

4.2.4. 外部割り込み

1) __CreateInterrupt

概要

＜外部割り込みの設定＞

Boolean __CreateInterrupt(unsigned long data, void *func)

data	設定データ
func	コールバック関数ポインタ

解説（1/2）

指定した外部割り込みの設定を行います。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。ただし、複数の外部割り込みを同時に指定することはできません。

RAPI_NMI	NMI 割り込み
RAPI_IRQ0	IRQ0 割り込み
RAPI_IRQ1	IRQ1 割り込み
RAPI_IRQ2	IRQ2 割り込み
RAPI_IRQ3	IRQ3 割り込み
RAPI_INT_LOW	割り込み要求をローレベルで検出
RAPI_INT_FALLING	割り込み要求を立ち下がりエッジで検出
RAPI_INT_RISING	割り込み要求を立ち上がりエッジで検出
RAPI_INT_BOTH	割り込み要求を両エッジで検出
RAPI_IRQOUT_INT_OUTPUT	割り込み要求検出信号（IRQOUT）を出力する
RAPI_IRQOUT_HIGH_LEVEL	割り込み要求検出信号（IRQOUT）を出力しない
RAPI_INT_LV_0	割り込み優先レベル 0
RAPI_INT_LV_1	割り込み優先レベル 1
RAPI_INT_LV_2	割り込み優先レベル 2
RAPI_INT_LV_3	割り込み優先レベル 3
RAPI_INT_LV_4	割り込み優先レベル 4
RAPI_INT_LV_5	割り込み優先レベル 5
RAPI_INT_LV_6	割り込み優先レベル 6
RAPI_INT_LV_7	割り込み優先レベル 7
RAPI_INT_LV_8	割り込み優先レベル 8
RAPI_INT_LV_9	割り込み優先レベル 9
RAPI_INT_LV_10	割り込み優先レベル 10
RAPI_INT_LV_11	割り込み優先レベル 11
RAPI_INT_LV_12	割り込み優先レベル 12
RAPI_INT_LV_13	割り込み優先レベルを 13
RAPI_INT_LV_14	割り込み優先レベルを 14
RAPI_INT_LV_15	割り込み優先レベルを 15

解説（2/2）

●IRQ0～IRQ3 割り込み（RAPI_IRQ0～RAPI_IRQ3）選択時に指定可能なパラメータ：

（極性センス） { RAPI_INT_LOW，RAPI_INT_FALLING，RAPI_INT_RISING，RAPI_INT_BOTH } から 1 つを指定します。
初期値は RAPI_INT_LOW です。

（IRQOUT 信号出力） { RAPI_IRQOUT_INT_OUTPUT，RAPI_IRQOUT_HIGH_LEVEL } から 1 つを指定します。
初期値は RAPI_IRQOUT_INT_OUTPUT です。

（割り込み優先レベル） { RAPI_INT_LV_0，RAPI_INT_LV_1，
RAPI_INT_LV_2，RAPI_INT_LV_3，
RAPI_INT_LV_4，RAPI_INT_LV_5，
RAPI_INT_LV_6，RAPI_INT_LV_7，
RAPI_INT_LV_8，RAPI_INT_LV_9，
RAPI_INT_LV_10，RAPI_INT_LV_11，
RAPI_INT_LV_12，RAPI_INT_LV_13，
RAPI_INT_LV_14，RAPI_INT_LV_15 }
から 1 つを指定します。初期値は RAPI_INT_LV_0 です。

●NMI 割り込み（RAPI_NMI）選択時に指定可能なパラメータ：

（極性センス） { RAPI_INT_FALLING，RAPI_INT_RISING } から 1 つを指定します。初期値は RAPI_INT_FALLING です。

（IRQOUT 信号出力） { RAPI_IRQOUT_INT_OUTPUT，RAPI_IRQOUT_HIGH_LEVEL } から 1 つを指定します。
初期値は RAPI_IRQOUT_INT_OUTPUT です。

（割り込み優先レベル） NMI の割り込み優先レベルは 16 固定で設定不可です。

[func]
func には、コールバック関数を指すポインタを指定して下さい。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。

戻り値

外部割り込みの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

外部割り込み

参照

__EnableInterrupt，__GetInterruptAndPinInfo，__ClearInterruptFlag

備考

- 本API関数にて指定した割り込み優先レベルは、__EnableInterrupt関数にてRAPI_IRQ_ENAを指定して実行した後に、インタラプトプライオリティレジスタ（IPR）に設定されます。
- 第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_interrupt_sh_7125.h"

/* コールバック関数宣言 */
void IntFunc( void );

void func( void )
{
    /* IRQ0 割り込みの設定 */
    __CreateInterrupt( RAPI_IRQ0 | RAPI_INT_FALLING |
           RAPI_IRQOUT_INT_OUTPUT | RAPI_INT_LV_7, IntFunc );
}
```

2) __EnableInterrupt

概要

＜外部割り込みの制御＞

Boolean __EnableInterrupt(unsigned long data)

data	設定データ

解説

指定した外部割り込みの動作を制御します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_IRQ0	IRQ0 割り込み
RAPI_IRQ1	IRQ1 割り込み
RAPI_IRQ2	IRQ2 割り込み
RAPI_IRQ3	IRQ3 割り込み
RAPI_INT_REQUEST_CLEAR	IRQi 割り込み要求ステータスフラグ（i = 0～3）をクリア（ローレベル検出に設定している場合は無効）
RAPI_INT_REQUEST_REMAIN	IRQi 割り込み要求ステータスフラグ（i = 0～3）を保持（ローレベル検出に設定している場合は無効）
RAPI_IRQ_DIS	割り込みを禁止
RAPI_IRQ_ENA	割り込みを許可

戻り値

外部割り込みの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

外部割り込み

参照

__CreateInterrupt，__GetInterruptAndPinInfo，__ClearInterruptFlag

備考

- __CreateInterrupt関数にて指定した割り込み優先レベルは、本API関数にてRAPI_IRQ_ENAを指定して実行した後、インタラプトプライオリティレジスタ（IPR）に設定されます。
- 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_interrupt_sh_7125.h"

void func( void )
{
    /* IRQ1 割り込みを許可 */
    __EnableInterrupt( RAPI_IRQ1 | RAPI_INT_REQUEST_CLEAR |
           RAPI_IRQ_ENA );
}
```

3) __GetInterruptAndPinInfo

概要

＜外部割り込み入力端子のレベルおよび割り込み要求のステータス取得＞

Boolean __GetInterruptInfo(unsigned long data1, unsigned char *data2)

data1	設定データ 1
data2	端子レベルおよびステータスフラグを格納するバッファへのポインタ

解説

指定した外部割り込みの割り込み要求入力端子レベルおよび割り込み要求ステータスフラグを取得します。

[data1]

data1 には、以下のパラメータを指定して下さい。複数の IRQi 端子レベルまたは IRQi 割り込み要求ステータスフラグ（i = 0～3）を読み出す場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_NMI_PIN	NMI 端子レベル
RAPI_IRQ0_PIN	IRQ0 端子レベル
RAPI_IRQ1_PIN	IRQ1 端子レベル
RAPI_IRQ2_PIN	IRQ2 端子レベル
RAPI_IRQ3_PIN	IRQ3 端子レベル
RAPI_IRQ0_FLAG	IRQ0 割り込み要求ステータスフラグ
RAPI_IRQ1_FLAG	IRQ1 割り込み要求ステータスフラグ
RAPI_IRQ2_FLAG	IRQ2 割り込み要求ステータスフラグ
RAPI_IRQ3_FLAG	IRQ3 割り込み要求ステータスフラグ

●IRQ ステータスレジスタ（IRQSR）の構成：

1	1	1	1	IRQ3L	IRQ2L	IRQ1L	IRQ0L	0	0	0	0	IRQ3F	IRQ2F	IRQ1F	IRQ0F

IRQi（i=0～3）入力端子レベル（0：Low；1：High）

割り込み要求ステータスフラグの値（0：割り込み要求なし；1：割り込み要求あり）

●割り込みコントロールレジスタ 0（ICR0）の構成：

NMIL	0	0	0	0	0	0	NMIE	0	0	0	0	0	0	0	0

NMI 入力端子レベル（0：Low；1：High）

戻り値

外部割り込みの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

外部割り込み

参照

__CreateInterrupt，__EnableInterrupt，__ClearInterruptFlag

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_interrupt_sh_7125.h"

void func( void )
{
   unsigned char irq[8];
   unsigned char nmi;

   /* IRQ0～3 割り込み要求ステータスフラグおよび各端子レベルを取得 */
   __GetInterruptAndPinInfo( RAPI_IRQ0_FLAG | RAPI_IRQ0_PIN |
          RAPI_IRQ1_FLAG | RAPI_IRQ1_PIN | RAPI_IRQ2_FLAG |
          RAPI_IRQ2_PIN | RAPI_IRQ3_FLAG | RAPI_IRQ3_PIN, irq );

   /* NMI 端子レベルを取得 */
   __GetInterruptAndPinInfo( RAPI_NMI_PIN, &nmi );
}
```

4) __ClearInterruptFlag

概要

＜割り込み要求のステータスクリア＞

Boolean __ClearInterruptFlag(unsigned long data)

data	設定データ

解説

指定した外部割り込みの割り込み要求ステータスフラグをクリアします。

[data]

data には、以下のパラメータを指定して下さい。複数の IRQi 割り込み要求ステータスフラグ（i = 0～3）をクリアするには、パラメータ間に“|”記号を入れて下さい。

RAPI_IRQ0_FLAG	IRQ0 割り込み要求ステータスフラグ
RAPI_IRQ1_FLAG	IRQ1 割り込み要求ステータスフラグ
RAPI_IRQ2_FLAG	IRQ2 割り込み要求ステータスフラグ
RAPI_IRQ3_FLAG	IRQ3 割り込み要求ステータスフラグ

戻り値

外部割り込みの指定が正しくない場合は RAPI_FALSE を返し、それ以外の場合は RAPI_TRUE を返します。

周辺機能

外部割り込み

参照

__CreateInterrupt，__EnableInterrupt，__GetInterruptAndPinInfo

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_interrupt_sh_7125.h"

void func( void )
{
    /* IRQ0～3 割り込み要求ステータスフラグをクリア */
    __ClearInterruptFlag( RAPI_IRQ0_FLAG | RAPI_IRQ1_FLAG |
           RAPI_IRQ2_FLAG | RAPI_IRQ3_FLAG );
}
```

4.2.5. A/Dコンバータ

1) __CreateADC

概要

＜A/D コンバータの設定＞

Boolean __CreateADC(unsigned long data, void *func)

data	設定データ
func	コールバック関数ポインタ

解説（1/6）

A/D コンバータの動作条件を設定します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。ただし、アナログ入力端子用パラメータは複数同時に選択できません。

RAPI_SINGLE	シングルモード
RAPI_4CHANNEL_SCAN	4 チャネルスキャンモード
RAPI_2CHANNEL_SCAN	2 チャネルスキャンモード
RAPI_SINGLE_CYCLE	1 サイクルスキャンモード
RAPI_CONTINUOUS	連続スキャンモード
RAPI_AN0	アナログ入力端子 0
RAPI_AN1	アナログ入力端子 1
RAPI_AN2	アナログ入力端子 2
RAPI_AN3	アナログ入力端子 3
RAPI_AN4	アナログ入力端子 4
RAPI_AN5	アナログ入力端子 5
RAPI_AN6	アナログ入力端子 6
RAPI_AN7	アナログ入力端子 7
RAPI_AN0_AN1	アナログ入力端子 0 および 1
RAPI_AN0_AN2	アナログ入力端子 0～2
RAPI_AN0_AN3	アナログ入力端子 0～3
RAPI_AN4_AN5	アナログ入力端子 4 および 5
RAPI_AN4_AN6	アナログ入力端子 4～6
RAPI_AN4_AN7	アナログ入力端子 4～7
RAPI_AN2_AN3	アナログ入力端子 2 および 3
RAPI_AN6_AN7	アナログ入力端子 6 および 7
RAPI_50_STATES	ステートコントロール：変換時間 50 ステート
RAPI_64_STATES	ステートコントロール：変換時間 64 ステート
RAPI_P_4	Pφ/4 クロック
RAPI_P_3	Pφ/3 クロック
RAPI_P_2	Pφ/2 クロック
RAPI_P_1	Pφクロック（Pφ≦25 MHz の場合）
RAPI_TRIGGER_ENA	ADTRG または MTU2 トリガによる A/D 変換のトリガを許可
RAPI_TRIGGER_DIS	ADTRG または MTU2 トリガによる A/D 変換のトリガを禁止
RAPI_ADF_SOURCE_OR	グループ 0 トリガまたはグループ 1 トリガのいずれかの変換終了時に A/D エンドフラグをセット （2 チャネルスキャンモードにて、A/D 変換のトリガが許可されている場合のみ有効）
RAPI_ADF_SOURCE_AND	グループ 0 トリガおよびグループ 1 トリガの両方の変換終了時に A/D エンドフラグをセット （2 チャネルスキャンモードにて、A/D 変換のトリガが許可されている場合のみ有効） ただし、トリガの順番には影響されません。

解説（2/6）

RAPI_AD_INT_ENA	A/D 割り込みを許可
RAPI_AD_INT_DIS	A/D 割り込みを禁止
RAPI_EXTERNAL_TRIGGER_G0	外部トリガ端子（ADTRG）入力
RAPI_TGRA_TCNT4_TRIGGER_G0	MTU2 各チャネルの TGRA のインプットキャプチャ／コンペアマッチ、相補 PWM モード時、TCNT_4 のアンダフロー（谷）（TRGAN）
RAPI_TGRA_MTU2_CHANNEL0_G0	MTU2 チャネル 0 コンペアマッチ（TRG0N）
RAPI_MTU2_TRG4AN_G0	MTU2 A/D 変換開始要求ディレイド（TRG4AN）
RAPI_MTU2_TRG4BN_G0	MTU2 A/D 変換開始要求ディレイド（TRG4BN）
RAPI_EXTERNAL_TRIGGER_G1	外部トリガ端子（ADTRG）入力 （2 チャネルスキャンモード時のグループ 1 の場合）
RAPI_TGRA_TCNT4_TRIGGER_G1	MTU2 各チャネルの TGRA のインプットキャプチャ／コンペアマッチ、相補 PWM モード時、TCNT_4 のアンダフロー（谷）（TRGAN） （2 チャネルスキャンモード時のグループ 1 の場合）
RAPI_TGRA_MTU2_CHANNEL0_G1	MTU2 チャネル 0 コンペアマッチ（TRG0N） （2 チャネルスキャンモード時のグループ 1 の場合）
RAPI_MTU2_TRG4AN_G1	MTU2 A/D 変換開始要求ディレイド（TRG4AN） （2 チャネルスキャンモード時のグループ 1 の場合）
RAPI_MTU2_TRG4BN_G1	MTU2 A/D 変換開始要求ディレイド（TRG4BN） （2 チャネルスキャンモード時のグループ 1 の場合）
【注】上記 A/D トリガ用パラメータの内、“～_G0”は、シングルモード、4 チャネルスキャンモード、または 2 チャネルスキャンモード時のグループ 0 使用時に指定して下さい。 一方、“～_G1”は、2 チャネルスキャンモード時のグループ 1 使用時に指定して下さい。	
RAPI_ADC_INT_LV_0	割り込み優先レベル 0
RAPI_ADC_INT_LV_1	割り込み優先レベル 1
RAPI_ADC_INT_LV_2	割り込み優先レベル 2
RAPI_ADC_INT_LV_3	割り込み優先レベル 3
RAPI_ADC_INT_LV_4	割り込み優先レベル 4
RAPI_ADC_INT_LV_5	割り込み優先レベル 5
RAPI_ADC_INT_LV_6	割り込み優先レベル 6
RAPI_ADC_INT_LV_7	割り込み優先レベル 7
RAPI_ADC_INT_LV_8	割り込み優先レベル 8
RAPI_ADC_INT_LV_9	割り込み優先レベル 9
RAPI_ADC_INT_LV_10	割り込み優先レベル 10
RAPI_ADC_INT_LV_11	割り込み優先レベル 11
RAPI_ADC_INT_LV_12	割り込み優先レベル 12
RAPI_ADC_INT_LV_13	割り込み優先レベル 13
RAPI_ADC_INT_LV_14	割り込み優先レベル 14
RAPI_ADC_INT_LV_15	割り込み優先レベル 15

解説（3/6）

●シングルモード（RAPI_SINGLE）選択時に指定可能なパラメータ：

（アナログ入力端子） { RAPI_AN0，RAPI_AN1，RAPI_AN2，RAPI_AN3，RAPI_AN4，RAPI_AN5，RAPI_AN6，RAPI_AN7 } から 1 つを指定します。
初期値は RAPI_AN0（A/D モジュール 0 の場合）および RAPI_AN4（A/D モジュール 1 の場合）です。

（ステート制御） { RAPI_50_STATES，RAPI_64_STATES } から 1 つを指定します。
初期値は RAPI_50_STATES です。

（クロック選択） { RAPI_P_4，RAPI_P_3，RAPI_P_2，RAPI_P_1 } から 1 つを指定します。初期値は RAPI_P_4 です。

（トリガイネーブル） { RAPI_TRIGGER_ENA，RAPI_TRIGGER_DIS } から 1 つを指定します。初期値は RAPI_TRIGGER_DIS です。

（トリガ選択） { RAPI_EXTERNAL_TRIGGER_G0,
RAPI_TGRA_TCNT4_TRIGGER_G0,
RAPI_TGRA_MTU2_CHANNEL0_G0,
RAPI_MTU2_TRG4AN_G0,
RAPI_MTU2_TRG4BN_G0 } から 1 つを指定します。
初期値は RAPI_EXTERNAL_TRIGGER_G0 です。

（割り込みイネーブル） { RAPI_AD_INT_ENA，RAPI_AD_INT_DIS } から 1 つを指定します。初期値は RAPI_AD_INT_DIS です。

（割り込み優先レベル） { RAPI_ADC_INT_LV_0，RAPI_ADC_INT_LV_1,
RAPI_ADC_INT_LV_2，RAPI_ADC_INT_LV_3,
RAPI_ADC_INT_LV_4，RAPI_ADC_INT_LV_5,
RAPI_ADC_INT_LV_6，RAPI_ADC_INT_LV_7,
RAPI_ADC_INT_LV_8，API_ADC_INT_LV_9,
RAPI_ADC_INT_LV_10，RAPI_ADC_INT_LV_11,
RAPI_ADC_INT_LV_12，RAPI_ADC_INT_LV_13,
RAPI_ADC_INT_LV_14，RAPI_ADC_INT_LV_15 }
から 1 つを指定します。初期値は RAPI_ADC_INT_LV_0 です。

解説（4/6）

●4 チャネルスキャンモード（RAPI_4CHANNEL_SCAN）選択時に指定可能なパラメータ：

（アナログ入力端子） { RAPI_AN0，RAPI_AN0_AN1，RAPI_AN0_AN2，RAPI_AN0_AN3，RAPI_AN4，RAPI_AN4_AN5，RAPI_AN4_AN6，RAPI_AN4_AN7 } から 1 つを指定します。初期値は RAPI_AN0（A/D モジュール 0 の場合）および RAPI_AN4（A/D モジュール 1 の場合）です。

（ステート制御） { RAPI_50_STATES，RAPI_64_STATES } から 1 つを指定します。初期値は RAPI_50_STATES です。

（クロック選択） { RAPI_P_4，RAPI_P_3，RAPI_P_2，RAPI_P_1 } から 1 つを指定します。初期値は RAPI_P_4 です。

（スキャン設定） { RAPI_SINGLE_CYCLE，RAPI_CONTINUOUS } から 1 つを指定します。初期値は RAPI_SINGLE_CYCLE です。

（トリガイネーブル） { RAPI_TRIGGER_ENA，RAPI_TRIGGER_DIS } から 1 つを指定します。初期値は RAPI_TRIGGER_DIS です。

（トリガ選択） { RAPI_EXTERNAL_TRIGGER_G0，RAPI_TGRA_TCNT4_TRIGGER_G0，RAPI_TGRA_MTU2_CHANNEL0_G0，RAPI_MTU2_TRG4AN_G0，RAPI_MTU2_TRG4BN_G0 } から 1 つを指定します。初期値は RAPI_EXTERNAL_TRIGGER_G0 です。

（割り込みイネーブル） { RAPI_AD_INT_ENA，RAPI_AD_INT_DIS } から 1 つを指定します。初期値は RAPI_AD_INT_DIS です。

（割り込み優先レベル） { RAPI_ADC_INT_LV_0，RAPI_ADC_INT_LV_1，RAPI_ADC_INT_LV_2，RAPI_ADC_INT_LV_3，RAPI_ADC_INT_LV_4，RAPI_ADC_INT_LV_5，RAPI_ADC_INT_LV_6，RAPI_ADC_INT_LV_7，RAPI_ADC_INT_LV_8，API_ADC_INT_LV_9，RAPI_ADC_INT_LV_10，RAPI_ADC_INT_LV_11，RAPI_ADC_INT_LV_12，RAPI_ADC_INT_LV_13，RAPI_ADC_INT_LV_14，RAPI_ADC_INT_LV_15 } から 1 つを指定します。初期値は RAPI_ADC_INT_LV_0 です。

解説（5/6）

●2 チャネルスキャンモード（RAPI_2CHANNEL_SCAN）設定時に指定可能なパラメータ：

（アナログ入力端子） { RAPI_AN0，RAPI_AN0_AN1，RAPI_AN4，RAPI_AN4_AN5 } から 1 つを指定します。
初期値は RAPI_AN0（A/D モジュール 0 の場合）および RAPI_AN4（A/D モジュール 1 の場合）です。

【注】2 チャネルスキャンモードでトリガを使用する場合（トリガイネーブル：RAPI_TRIGGER_ENA）、グループ 1 用のアナログ入力端子（RAPI_AN2, RAPI_AN2_AN3, RAPI_AN6，RAPI_AN6_AN7）の指定は推奨されません。
したがって、グループ 1 のみを選択する場合でも、グループ 0 用の端子（RAPI_AN0，RAPI_AN0_AN1，RAPI_AN4，RAPI_AN4_AN5）を指定して下さい。
これにより、それぞれ対応するグループ 1 用の端子も有効になります。

（ステート制御） { RAPI_50_STATES，RAPI_64_STATES } から 1 つを指定します。
初期値は RAPI_50_STATES です。

（クロック選択） { RAPI_P_4，RAPI_P_3，RAPI_P_2，RAPI_P_1 } から 1 つを指定します。初期値は RAPI_P_4 です。

（スキャン設定） { RAPI_SINGLE_CYCLE，RAPI_CONTINUOUS } から 1 つを指定します。初期値は RAPI_SINGLE_CYCLE です。

（トリガイネーブル） { RAPI_TRIGGER_ENA，RAPI_TRIGGER_DIS } から 1 つを指定します。初期値は RAPI_TRIGGER_DIS です。

（トリガ選択）
●グループ 0 に対しては、
{ RAPI_EXTERNAL_TRIGGER_G0,
RAPI_TGRA_TCNT4_TRIGGER_G0,
RAPI_TGRA_MTU2_CHANNEL0_G0,
RAPI_MTU2_TRG4AN_G0,
RAPI_MTU2_TRG4BN_G0 } から 1 つを指定します。
初期値は RAPI_EXTERNAL_TRIGGER_G0 です。
●グループ 1 に対しては、
{ RAPI_EXTERNAL_TRIGGER_G1,
RAPI_TGRA_TCNT4_TRIGGER_G1,
RAPI_TGRA_MTU2_CHANNEL0_G1,
RAPI_MTU2_TRG4AN_G1,
RAPI_MTU2_TRG4BN_G1 } から 1 つを指定します。
初期値は RAPI_EXTERNAL_TRIGGER_G1 です。
【注】グループ 0 とグループ 1 には、それぞれ別々のトリガを指定して下さい。

（ADF 制御） { RAPI_ADF_SOURCE_OR，RAPI_ADF_SOURCE_AND } から 1 つを指定します。初期値は RAPI_ADF_SOURCE_OR です。

（割り込みイネーブル） { RAPI_AD_INT_ENA，RAPI_AD_INT_DIS } から 1 つを指定します。初期値は RAPI_AD_INT_DIS です。

（割り込み優先レベル） { RAPI_ADC_INT_LV_0，RAPI_ADC_INT_LV_1,
RAPI_ADC_INT_LV_2，RAPI_ADC_INT_LV_3,
RAPI_ADC_INT_LV_4，RAPI_ADC_INT_LV_5,
RAPI_ADC_INT_LV_6，RAPI_ADC_INT_LV_7,
RAPI_ADC_INT_LV_8，API_ADC_INT_LV_9,
RAPI_ADC_INT_LV_10，RAPI_ADC_INT_LV_11,
RAPI_ADC_INT_LV_12，RAPI_ADC_INT_LV_13,
RAPI_ADC_INT_LV_14，RAPI_ADC_INT_LV_15 }
から 1 つを指定します。初期値は RAPI_ADC_INT_LV_0 です。

解説（6/6）

[func]
func には、コールバック関数を指すポインタを指定して下さい。コールバック関数を指定しない場合は、RAPI_NULL を指定して下さい。

戻り値

A/D コンバータの設定に成功した場合は RAPI_TRUE を返し、失敗した場合は RAPI_FALSE を返します。

周辺機能

A/D コンバータ

参照

__EnableADC，__DestroyADC，__GetADC

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_ad_sh_7125.h"

/* コールバック関数宣言 */
void AdIntFunc( void );

void func( void )
{
   /* A/D コンバータをシングルモードに設定 */
   __CreateADC( RAPI_SINGLE | RAPI_AN2 | RAPI_50_STATES |
          RAPI_P_2 | RAPI_TRIGGER_ENA | RAPI_AD_INT_ENA |
          RAPI_EXTERNAL_TRIGGER_G0 | RAPI_ADC_INT_LV_5, AdIntFunc );
}
```

2) __EnableADC

概要

＜A/D コンバータの動作制御＞

Boolean __EnableADC(unsigned long data)

data	設定データ

解説

A/D コンバータの動作開始または動作停止を制御します。

[data]

data には、以下のパラメータを指定して下さい。複数のパラメータを指定する場合は、パラメータ間に“|”記号を入れて下さい。（下記注意事項参照）

RAPI_AD_0	A/D モジュール 0
RAPI_AD_1	A/D モジュール 1
RAPI_AD_ON	A/D コンバータ動作開始
RAPI_AD_OFF	A/D コンバータ動作停止

【注】A/D モジュール 0，1（RAPI_AD_0 および RAPI_AD_1）は同時指定できません。

戻り値

A/D コンバータの制御に成功した場合は RAPI_TRUE を返し、失敗した場合は RAPI_FALSE を返します。

周辺機能

A/D コンバータ

参照

__CreateADC，__DestroyADC，__GetADC

備考

- 本 API 関数は、A/D コントロールレジスタ（ADCR）の ADST ビットのみ操作します。
  したがって、ソフトウェアによって A/D 変換（開始／停止）を行う場合のみ本 API 関数を使用して下さい。外部トリガによる A/D 変換を行う場合、本 API 関数は不要です。
- 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_ad_sh_7125.h"

void func( void )
{
    /* A/D コンバータ・モジュール 0 の動作停止 */
    __EnableADC( RAPI_AD_0 | RAPI_AD_OFF );
}
```

3) __DestroyADC

概要

＜A/D コンバータの設定クリア＞

Boolean __DestroyADC(unsigned long data)

data	設定データ

解説

指定した A/D コンバータの設定をクリアします。

[data]

data には、以下のパラメータを指定して下さい。（いずれか 1 つ）

RAPI_AD_0	A/D モジュール 0
RAPI_AD_1	A/D モジュール 1

戻り値

A/D コンバータの設定クリアに成功した場合は RAPI_TRUE を返し、失敗した場合は RAPI_FALSE を返します。

周辺機能

A/D コンバータ

参照

__CreateADC，__EnableADC，__GetADC

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_ad_sh_7125.h"

void func( void )
{
    /* A/D コンバータ・モジュール 0 の設定クリア */
    __DestroyADC( RAPI_AD_0 );
}
```

4) __GetADC

概要

＜A/D 変換の結果取得＞

Boolean __GetADC(unsigned long data1, unsigned short *data2)

data1	設定データ 1
data2	A/D 変換値を格納するバッファへのポインタ

解説

指定した A/D データレジスタから A/D 変換の結果を取得します。

[data1]

data1 には、以下のパラメータを指定して下さい。複数の A/D データレジスタの値を同時に取得する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_ADDR0	A/D データレジスタ 0
RAPI_ADDR1	A/D データレジスタ 1
RAPI_ADDR2	A/D データレジスタ 2
RAPI_ADDR3	A/D データレジスタ 3
RAPI_ADDR4	A/D データレジスタ 4
RAPI_ADDR5	A/D データレジスタ 5
RAPI_ADDR6	A/D データレジスタ 6
RAPI_ADDR7	A/D データレジスタ 7
RAPI_ADDR_ALL	A/D データレジスタ 0～7 すべて

[data2]

data2 には、A/D 変換値を格納するバッファへのポインタを指定して下さい。

【注】A/D 変換後、変換値は A/D データレジスタには左詰めで格納されますが、
バッファには右詰めで格納されます。

戻り値

A/D 変換結果の取得に成功した場合は RAPI_TRUE を返し、失敗した場合は RAPI_FALSE を返します。

周辺機能

A/D コンバータ

参照

__CreateADC，__EnableADC，__DestroyADC

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_ad_sh_7125.h"

void func( void )
{
    unsigned short data[2];

    /* A/D データレジスタ 0 および 1 の A/D 変換結果を取得 */
    __GetADC( RAPI_ADDR0 | RAPI_ADDR1, &data );
}
```

5) __GetADCFlag

概要

＜A/D コンバータのステータス取得＞

Boolean __GetADCFlag(unsigned long data1, unsigned char *status)

data	設定データ
status	A/D コンバータのステータスを示す A/D エンドフラグを格納するバッファへのポインタ

解説

指定した A/D コンバータのステータスフラグ（A/D エンドフラグ）を取得します。

[data]

data には、以下のパラメータを指定して下さい。複数の A/D モジュールのステータスを取得する場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_AD_0	A/D モジュール 0
RAPI_AD_1	A/D モジュール 1

●A/D エンドフラグ（ADF）は、A/D 変換の終了を示すステータスフラグです。
A/D コントロール／ステータスレジスタ（ADCSR）のビット 15 に相当します。

●ADCSR の構成：

ADF	ADIE	0	0	TRGE	0	CONADF	STC	CKSL[1:0]	ADM[1:0]	ADCS	CH[2:0]

A/D エンドフラグ

戻り値

A/D コンバータのステータス取得に成功した場合は RAPI_TRUE を返し、失敗した場合は RAPI_FALSE を返します。

周辺機能

A/D コンバータ

参照

__ClearADCFlag

備考

第 1 引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_ad_sh_7125.h"

void func( void )
{
    unsigned char status;

    /* A/D コンバータ・モジュール 0 および 1 のステータス取得 */
    __GetADCFlag( RAPI_AD_0 | RAPI_AD_1, &status );
}
```

6) __ClearADCFlag

概要

＜A/D コンバータのステータスクリア＞

Boolean __ClearADCFlag(unsigned long data)

data	設定データ

解説

指定した A/D コンバータのステータスフラグ（A/D エンドフラグ）をクリアします。

[data]

data には、以下のパラメータを指定して下さい。複数の A/D モジュールのステータスをクリアする場合は、パラメータ間に“|”記号を入れて下さい。

RAPI_AD_0	A/D モジュール 0
RAPI_AD_1	A/D モジュール 1

●A/D エンドフラグ（ADF）は、A/D 変換の終了を示すステータスフラグです。
A/D コントロール／ステータスレジスタ（ADCSR）のビット 15 に相当します。

●ADCSR の構成：

ADF	ADIE	0	0	TRGE	0	CONADF	STC	CKSL[1:0]	ADM[1:0]	ADCS	CH[2:0]

A/D エンドフラグ

戻り値

A/D コンバータのステータスクリアに成功した場合は RAPI_TRUE を返し、失敗した場合は RAPI_FALSE を返します。

周辺機能

A/D コンバータ

参照

__GetADCFlag

備考

引数に上記以外のパラメータを指定した場合の動作は保証されません。

プログラム例

```
#include "rapi_ad_sh_7125.h"

void func( void )
{
    /* A/D コンバータ・モジュール 1 のステータスクリア */
    __ClearADCFlag( RAPI_AD_1 );
}
```

5.　使用例

本章では、本ライブラリに内蔵している各周辺ドライバを設定する際のプログラミング使用例を示します。

5.1.　シリアル通信インタフェース（SCI）

(1)　シリアル受信

SCIチャネル 0 を調歩同期式モードに設定し、5 バイト分のデータをシリアル受信する場合の例を 図 5.1 に示します。受信エラー検出後の処理は、必要に応じてユーザ側で行って下さい。

```
#include ”rapi_sif_sh_7125.h”

Boolean ErrorFlag;
unsigned char ReceiveBuf[5];
unsigned short ReceiveNum;
unsigned short SpeedValue = 25000000 / 32 / 38400 – 1;
                /* 周辺クロック(Pφ)：25(MHz)、ビットレート：38400(bit/s) */
unsigned short WaitTimerValue = 30000;

void INT_SCI0_RX( void );
void ErrorOpe( void );

/* メイン処理 */
void func( void )
{
    /* SCI チャネル 0 を調歩同期式モードに設定 */
    __CreateSCI( RAPI_COM1 | RAPI_SM_ASYNC | RAPI_CKDIR_INT |
                 RAPI_BCSS_P1 | RAPI_LSB_SEL | RAPI_8_BIT_LENGTH |
                 RAPI_STPB_1 | RAPI_PARITY_ODD | RAPI_MULTI_DIS |
                 RAPI_INT_RX_ERR_ENA | RAPI_SCI_INT_LV_2, SpeedValue );
    /* 受信データ量を 5 バイトに設定し、シリアル受信開始 */
    __StartSCIReceiving( RAPI_COM1, ReceiveBuf, 5, &ReceiveNum );

    while( RAPI_TRUE ){
        if( ErrorFlag == RAPI_TRUE ){
            /* 受信エラー検出、一定期間ウェイト後に受信停止 */
            __StopSCIReceiving( RAPI_COM1, WaitTimerValue );
            /* エラー処理 */
            ErrorOpe();
            break;
        }
        if( ReceiveNum == 5 ){
            /* 5 バイトのデータ受信終了 */
            __StopSCIReceiving( RAPI_COM1, 0 );
            break;
        }
    }
}

/* 受信割り込み処理 */
void INT_SCI0_RX( void )
{
    /* 5 バイトのデータ受信 */
    if( __PollingSCIReceiving( RAPI_COM1 ) == RAPI_FALSE ){    /* 受信失敗 */
        ErrorFlag = RAPI_TRUE;
    }
}
```

図 5.1　シリアル受信の使用例

(2)　シリアル送信

SCIチャネル 0 を調歩同期式モードに設定し、5 バイト分のデータをシリアル送信する場合の例を 図 5.2 に示します。

```
#include ”rapi_sif_sh_7125.h”

unsigned char  SendBuf[5];
unsigned short SendNum;
unsigned short SpeedValue = 25000000 / 32 / 38400 – 1;
               /* 周辺クロック(Pφ)：25(MHz)、ビットレート：38400(bit/s) */
unsigned short WaitTimerValue = 30000;

void INT_SCI0_TX( void );

/* メイン処理 */
void func( void )
{
    /* SCI チャネル 0 を調歩同期式モードに設定 */
    __CreateSCI( RAPI_COM1 | RAPI_SM_ASYNC | RAPI_CKDIR_INT |
                 RAPI_BCSS_P1 | RAPI_LSB_SEL | RAPI_8_BIT_LENGTH |
                 RAPI_STPB_1 | RAPI_PARITY_ODD | RAPI_MULTI_DIS |
                 RAPI_INT_TX_ENA | RAPI_SCI_INT_LV_2, SpeedValue );
    /* 送信データ量を 5 バイトに設定し、シリアル送信開始 */
    __StartSCISending( RAPI_COM1, SendBuf, 5, &SendNum );

    while( RAPI_TRUE ){
        if( SendNum == 5 ){
            /* 5 バイトのデータ送信終了 */
            __StopSCISending( RAPI_COM1, WaitTimerValue );
            break;
        }
    }
}

/* 送信割り込み処理 */
void INT_SCI0_TX( void )
{
    /* 5 バイトのデータ送信 */
    __PollingSCISending( RAPI_COM1 );
}
```

図 5.2　シリアル送信の使用例

5.2. タイマMTU2

(1) タイマモード

MTU2チャネル0をタイマモードに設定する場合の例を図 5.3に示します。コールバック関数による処理は、必要に応じてユーザ側で行って下さい。

```
#include "rapi_timer_sh_7125.h"

/* コールバック関数宣言 */
void TimerIntFunc( void );

void func( void )
{
    /* MTU2 チャネル 0 をタイマモードに設定 */
    __CreateTimer( RAPI_MTU2_0 | RAPI_MP16 | RAPI_PERIODIC |
                   RAPI_RISING_EDGE | RAPI_TCNT_CLEAR_TGRB |
                   RAPI_OUTPUT_0_TOG | RAPI_COMPARE_MATCH_ENA |
                   RAPI_TIMER_INT_LV_5, 0x8000, TimerIntFunc );
    /* 動作開始 */
    __EnableTimer( RAPI_MTU2_0 | RAPI_TIMER_ON );
}
```

図 5.3 タイマモードの使用例

(2) PWM モード

MTU2 チャネル 0 およびチャネル 1 をPWMモード 2 に設定する場合の例を 図 5.4 に示します。コールバック関数による処理は、必要に応じてユーザ側で行って下さい。また、図 5.4 の設定による動作例およびPWM出力波形を 図 5.5 に示します。

```
#include "rapi_timer_sh_7125.h"

unsigned short PeriodValue = 60000;  /* 0xEA60 */
unsigned short DutyValue1  = 50000;  /* 0xC350 */
unsigned short DutyValue2  = 40000;  /* 0x9C40 */
unsigned short DutyValue3  = 30000;  /* 0x7530 */
unsigned short DutyValue4  = 20000;  /* 0x4E20 */

/* コールバック関数宣言 */
void Timer0IntAFunc( void );
void Timer0IntBFunc( void );
void Timer0IntCFunc( void );
void Timer1IntAFunc( void );
void Timer1IntBFunc( void );

void func( void )
{
    /* MTU2 チャネル 0 を PWM モード 2 に設定 */
    __CreatePWM( RAPI_MTU2_0 | RAPI_MP4 | RAPI_RISING_EDGE |
                 RAPI_PWM_MODE2 | RAPI_TCNT_CLEAR_TGRC |
                 RAPI_OVERFLOW_DIS | RAPI_TIMER_INT_LV_0 |
                 RAPI_TIMER_SYNC, RAPI_NULL );
    /* TGRC_0 を周期レジスタに設定 */
    __SetPWMPin( RAPI_MTU2_0 | RAPI_TGRC | RAPI_OUTPUT_RETAIN |
                 RAPI_COMPARE_MATCH_ENA | RAPI_TIMER_INT_LV_4,
                 PeriodValue, Timer0IntCFunc );
    /* TGRA_0 および TGRB_0 をデューティレジスタに設定 */
    __SetPWMPin( RAPI_MTU2_0 | RAPI_TGRA | RAPI_OUTPUT_0_1 |
                 RAPI_COMPARE_MATCH_ENA | RAPI_TIMER_INT_LV_4,
                 DutyValue1, Timer0IntAFunc );
    __SetPWMPin( RAPI_MTU2_0 | RAPI_TGRB | RAPI_OUTPUT_0_1 |
                 RAPI_COMPARE_MATCH_ENA | RAPI_TIMER_INT_LV_4,
                 DutyValue2, Timer0IntBFunc );

    /* MTU2 チャネル 1 を PWM モード 2 に設定 */
    __CreatePWM( RAPI_MTU2_1 | RAPI_MP4 | RAPI_RISING_EDGE |
                 RAPI_PWM_MODE2 | RAPI_TCNT_CLEAR_OTHER |
                 RAPI_OVERFLOW_DIS | RAPI_TIMER_INT_LV_0 |
                 RAPI_TIMER_SYNC, RAPI_NULL );
    /* TGRA_1 および TGRB_1 をデューティレジスタに設定 */
    __SetPWMPin( RAPI_MTU2_1 | RAPI_TGRA | RAPI_OUTPUT_0_1 |
                 RAPI_COMPARE_MATCH_ENA | RAPI_TIMER_INT_LV_4,
                 DutyValue3, Timer1IntAFunc );
    __SetPWMPin( RAPI_MTU2_1 | RAPI_TGRB | RAPI_OUTPUT_0_1 |
                 RAPI_COMPARE_MATCH_DIS | RAPI_TIMER_INT_LV_4,
                 DutyValue4, Timer1IntBFunc );

    /* 動作開始 */
    __EnablePWM( RAPI_MTU2_0 | RAPI_MTU2_1 | RAPI_TIMER_ON );
}
```

図 5.4 PWM モード 2 の使用例

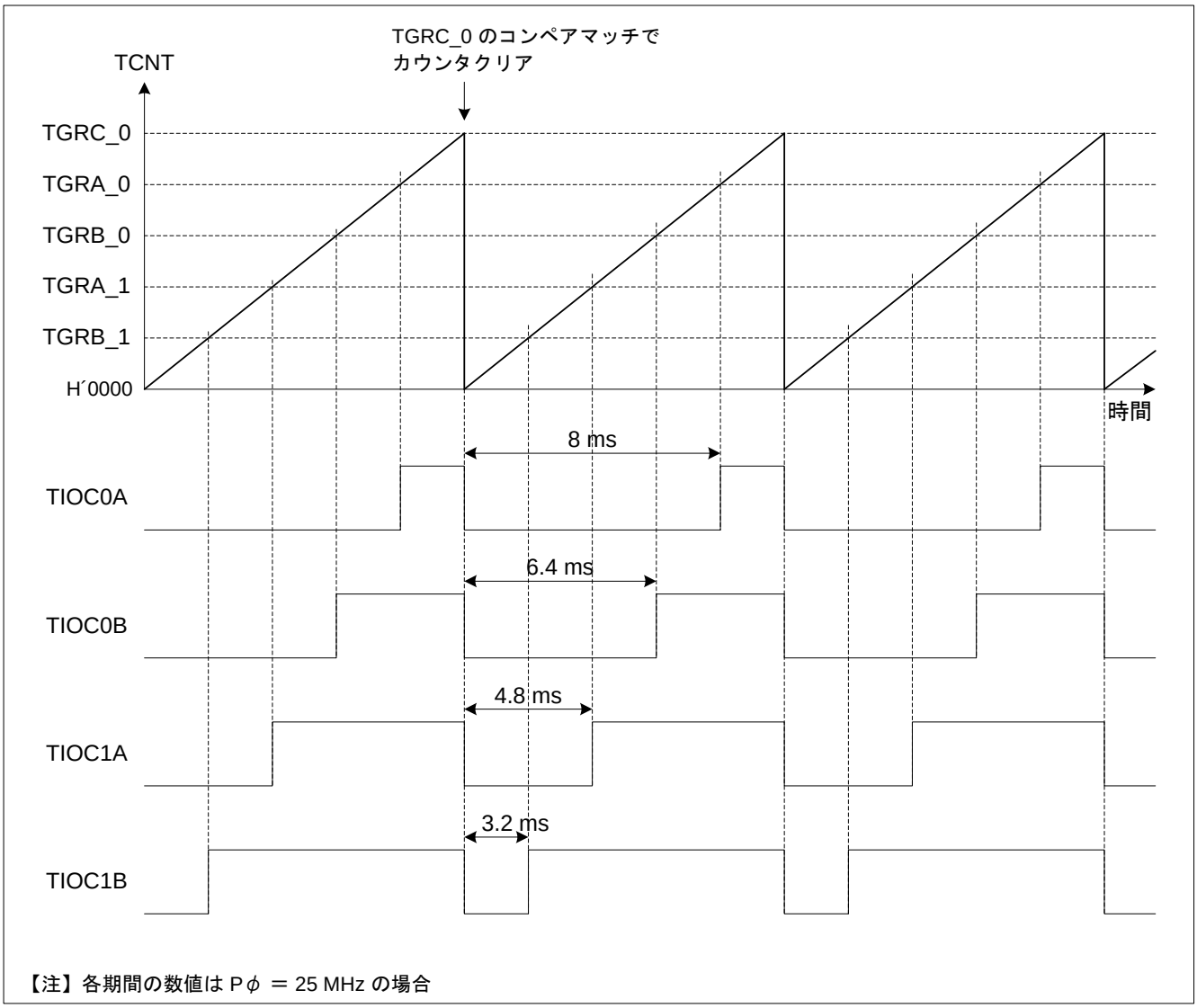


図 5.5　PWM モード 2 の動作例および PWM 出力波形

5.3. I/Oポート

図 5.6 に、I/Oポートの読み出し／書き込み例を示します。

```
#include "rapi_io_port_sh_7125.h"

unsigned short data[2];

void func( void )
{
    /* ポート A の下位 8 ビットの端子機能をポート入力に設定 */
    __SetIOPort( RAPI_FUNC_A | RAPI_LOWER_8_BITS |
                 RAPI_FUNC_IO | RAPI_PORT_INPUT );

    /* ポート A のビット 0 およびビット 3 の値をそれぞれ data[0]および data[1]に読み出す */
    __ReadIOPort( RAPI_FUNC_A | RAPI_BIT_0 | RAPI_BIT_3 |
                  RAPI_BITS_OPE | RAPI_PORT_REGISTER, data );

    /* ポート B のビット 1 およびビット 3 の端子機能をポート出力に設定 */
    __SetIOPort( RAPI_FUNC_B_L | RAPI_BIT_1 | RAPI_BIT_3 |
                 RAPI_FUNC_IO | RAPI_PORT_OUTPUT );

    /* ポート B のビット 1 およびビット 3 に 'H' を書き込む */
    __WriteIOPort( RAPI_FUNC_B_L | RAPI_BIT_1 | RAPI_BIT_3 |
                   RAPI_BITS_OPE, RAPI_H );
}
```

図 5.6 I/O ポートの読み出し／書き込み例

5.4.　外部割り込み

図 5.7 に、外部割り込みの使用例を示します。コールバック関数による処理は、必要に応じてユーザ側で行って下さい。

```
#include "rapi_interrupt_sh_7125.h"

/* コールバック関数宣言 */
void IntFunc( void );

void func( void )
{
    /* IRQ0 割り込みの設定：
        （立ち上がりエッジで検出、割り込み要求検出信号出力なし、割り込み優先レベル 5） */
    __CreateInterrupt( RAPI_IRQ0 | RAPI_INT_RISING | RAPI_IRQOUT_HIGH_LEVEL |
                       RAPI_INT_LV_5, IntFunc );
    /* IRQ0 割り込みイネーブル */
    __EnableInterrupt( RAPI_IRQ0 | RAPI_INT_REQUEST_CLEAR | RAPI_IRQ_ENA );
}
```

図 5.7　外部割り込みの使用例

5.5. A/Dコンバータ

A/Dコンバータの使用例を 図 5.8 に示します。コールバック関数による処理は、必要に応じてユーザ側で行って下さい。

```
#include "rapi_ad_sh_7125.h"

/* コールバック関数宣言 */
void AdIntFunc( void );

void func( void )
{
    /* AN2 端子に対し、外部トリガ入力による A/D 変換（シングルモード）開始 */
    __CreateADC( RAPI_SINGLE | RAPI_AN2 | RAPI_P_2 | RAPI_50_STATES |
                 RAPI_TRIGGER_ENA | RAPI_EXTERNAL_TRIGGER_G0 |
                 RAPI_AD_INT_ENA | RAPI_ADC_INT_LV_3, AdIntFunc );
}
```

図 5.8 A/D コンバータの使用例

改訂履歴	Renesas Embedded Application Programming Interface ユーザーズマニュアル

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2008.06.10	---	初版発行。
1.01	2008.07.08	---	ページヘッダのレイアウト変更。
		4-16	SCK 端子をポート出力で使用する場合の注意事項を備考欄に追記。
		4-69	未定義のパラメータを記載していたため削除。
		4-74	• パルス幅測定時の注意事項を備考欄に追記。 • プログラム例を変更。
		4-107	指定可能な IRQ 割り込み優先レベルの説明を改訂。
		4-114	アナログ入力端子を複数指定できる旨を記述していたため、その部分を削除。
		4-115	• トリガイネーブル時のアナログ入力端子の指定について注意事項を追記。 • トリガ選択時の注意事項を追記。
		4-117	• A/D モジュールチャネルは複数指定できない点を注意事項として追記。 • 関数使用時の留意事項を備考欄に追記。
1.02	2008.07.25	---	ページフッタのレイアウト変更
		4-39	使用しないパラメータを記載していたため削除。
		4-42	同上
		4-75	同上
		4-78	同上
		4-88	同上
		4-89	同上
		4-90	備考欄の記述内容を 1 項目削除。
		4-111	パラメータの説明を一部変更。
1.03	2008.08.04	4-102	第 2 引数の型を変更
		4-103	型の変更に伴うプログラム例の修正
		5-6	型の変更に伴う使用例の修正

SH/Tiny版
ユーザーズマニュアル
Renesas Embedded Application Programming Interface

発行年月日　2008年8月4日　　Rev.1.03

発行　　株式会社ルネサス テクノロジ　営業統括部
〒100-0004　東京都千代田区大手町2-6-2

編集　　株式会社ルネサスソリューションズ
第一応用技術本部　マイコン応用技術部

Renesas Embedded Application Programming Interface
ユーザーズマニュアル

ルネサス エレクトロニクス株式会社
神奈川県川崎市中原区下沼部1753 〒211-8668

RJJ10J2275-0103