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

リファレンスマニュアル

ルネサス エレクトロニクス
www.renesas.com

Rev.1.12　2009.05

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

変更履歴

Rev.	変更内容	作成
1.00	・ 0.21 から 1.00 にバージョンを変更 ・ タイマに指定可能な定義値の組合せの項目を追加 ・ 外部割り込みに指定可能な定義値の組合せの項目を追加 ・ A/D 変換器に指定可能な定義値の組合せの項目を追加 ・ 各定義値の説明の見直し・修正を実施 ・ 定義値 RAPI_TG_TAITG を削除。RAPI_TG_TAIIN と同値のため ・ プログラム例のヘッダファイル修正 ・ RAPI_NULL の追加修正 ・ __CreatePulseWidthMeasurementMode の H8/300H の表の不要部分削除 ・ Tiny API の記述を削除 ・ __CreatePulseWidthModulationMode の注意事項追加 ・ __StartSerialReceiving の注意事項追加	06.04.07
1.01	・ H8/36049、H8/36077、H8/36109、および R8C/24,25 用ライブラリ対応のための追加修正 ・ 「1.はじめに」の余分な句点を削除 ・ H8 のポート P73 を削除 ・ __CreateInputCapture のチャネル A～D の表記をインプットキャプチャ A～D へ訂正 ・ __CreateOutputCompare のチャネル A～D の表記をアウトプットコンペア A～D へ訂正 ・ seroal→serial 誤記訂正 ・ RAPI_70_STATES の説明誤記修正 ・ タイマチャネル名の誤記を修正 ・ インプットキャプチャモードの誤記修正 ・ アウトプットコンペアモードの誤記修正 ・ RAPI_OVERFLOW_BIT15 の説明の誤記修正 ・ R8C UART1 クロック同期モード対応 ・ __CreateInputCapture と__CreateOutputCompare の R8C と H8/300H の第 1 引数に RAPI_COUNT_CLEAR を追加 ・ __CreateInputCapture と__CreateOutputCompare の R8C のタイマ RD のカウントソースから「外部信号」を削除 ・ __SetTimerRegister と__GetTimerRegister のタイマ RD の項目に次の記載を追加 [15]：タイマ RD アウトプットマスタ許可レジスタ 1 [16]：タイマ RD アウトプットマスタ許可レジスタ 2 [17]：タイマ RD アウトプット制御レジスタ ・ __Create...、__Open...および、BasicOpen...の説明に、H8/300H のモジュール解除の説明を追加 ・ __Destroy...、__Close...および、__BasicClose...の説明に、H8/300H のモジュールスタンバイ設定の説明を追加 ・ __PollingSerialReceiving の戻り値を Boolean から unsigned int へ変更し、戻り値の説明とプログラム例を変更 ・ __PollingSerialSending の戻り値を Boolean から unsigned int へ変更し、戻り値の説明とプログラム例を変更 ・ __ReadIOPort および、__ReadIOPortRegister に、H8/300H のポート B の使用に関する備考を追加 ・ __CreateInputCapture の第 3 引数の説明を修正 ・ 「指定された」を「指定した」へ表記を統一 ・ __CreateEventCounter の H8/300H のタイマ B1 のカウントエッジの説明から RAPI_BOTH を削除 ・ __CreatePulsePeriodMeasurementMode および__CreatePulseWidthMeasurementMode に RAPI_FTIOA と RAPI_FTIOB を追加 ・ 「Tiny」の表記をすべて削除 ・ __GetPulsePeriodMeasurementMode、__GetPulseWidthMeasurementMode および、__GetEventCounter に第 2 引数の説明を追加 ・ __CreateTimer、__CreateEventCounter および、__CreatePulseWidthModulationMode の	06.12.25

	H8 のタイマ V のトリガに関する記述を修正 ・ __SetSerialInterrupt の H8 の割り込み設定と割り込みコントロールレジスタへの設定の説明を修正 ・ __EnableTimerRegister の H8/300H のタイマ V の備考を追加 ・ __CreatePulsePeriodMeasurementMode および__CreatePulseWidthMeasurementMode の R8C のタイマ RA 入力端子指定、RAPI_TIOSEL_P1_7 および、RAPI_TIOSEL_P1_5 を追加 ・ プログラム例のファイル名を修正 ・ __BasicSetSerialFormat の「参照」項目を削除 ・ __SetSerialFormat と__SetSerialInterrupt の「参照」に__BasicSetSerialFormat を追加 ・ __CreateInputCapture と__CreateOutputCompare の誤記訂正 ・ 「2.1 概要」に説明追加 ・ __CreateInputCapture のタイマ W とタイマ RC のカウンタクリア要因を削除 ・ __CreateOutputCompare の R8C の項目に、RAPI_COMPARE_MATCH_A_STOP および、RAPI_STOP を追加 ・ __SetSerialInterrupt の H8/300H の RAPI_INT_LV_0 および RAPI_INT_LV_1 へ変更 ・ __CreateTimer のクロックに関する記述を修正 ・ __CreatePulsePeriodMeasurementMode および__CreatePulseWidthMeasurementMode の H8 のフィルタ指定に、RAPI_TRC_FILTER_F および RAPI_TRD_FILTER_F を追加 ・ H8/36109 のタイマ RD に RAPI_TIMER_RD2 および RAPI_TIMER_RD3 を追加 ・ __CreateTimer の H8/300H のタイマ V のトリガ入力に関する記述を削除 ・ __CreateEventCounter の R8C の項目に、パルス出力機能の記述を追加 ・ __CreateOutputCompare の R8C の data3 と data5 の項目に、タイマ RE 使用時の記述を追加 ・ __CreatePulsePeriodMeasurementMode、__CreatePulseWidthMeasurementMode および、__CreateInputCapture のタイマ RC およびタイマ RD のフィルタ機能の項目を修正 ・ __EnableInterrupt の data1 に複数の定義値を指定可能の記述を追加 ・ __CreateADC の M16C の遅延トリガモード 0 と遅延トリガモード 1 で RAPI_WITHOUT_SAMPLE_HOLD が指定可能になっていたのを修正 ・ __CreateADC の R8C の繰り返しモードで RAPI_FOCOF が指定可能になっていたのを修正 ・ __CreateADC の M16C の data1 の項目に RAPI_AN30、RAPI_AN31、RAPI_AN32、RAPI_P9_GROUP を追加 ・ _EnableADC の M16C の data1 の項目に RAPI_AN30、RAPI_AN31、RAPI_AN32、RAPI_P9_GROUP を追加 ・ __CreateOutputCompare の H8/300H の割り込みの設定値の誤記を修正 ・ タイマ RD とタイマ RD のシンボル名の誤記を訂正 ・ __CreateEventCounter の M16C のゲート機能の記述を削除 ・ __BasicSetSerialFormat の H8/300H ノイズ除去機能の説明を追加 ・ __CreatePulsePeriodMeasurementMode と__CreatePulseWidthMeasurementMode の H8/300H タイマ W の入力端子の記述を削除	
1.02	・ R8C/26,27,28,29,2A,2B,2C,2D 用ライブラリ対応のための追加修正	07.02.26
1.03	・ 「Renesas API」の表記を「Renesas Embedded API」へ訂正 ・ 「4.1 機能別 API 一覧」の「Renesas Embedded」を「Renesas Embedded API」へ訂正 ・ __BasicOpenSerialDriver、__BasicCloseSerialDriver、__OpenSerialDriver および、__CloseSerialDriver のプログラム例がコンパイルエラーになるのを修正 ・ __BasicStartSerialSending の要約、備考およびプログラム例の誤記を訂正 ・ __BasicStopSerialReceiving の要約の誤記を訂正 ・ __CreatePulseWidthModulationMode の M16C のカウント開始条件の省略値を RAPI_TG_TAIIN から RAPI_TG_TAIS へ訂正 ・ __SetInterrupt の M16C のキー入力割り込み使用時の省略値の記述を削除 ・ __CreateADC の R8C の繰り返しモード使用時、単掃引モード使用時および、繰り返し掃引モード 0 の設定項目に分解能が抜けていたのを追加修正	07.11.30
1.04	・ H8S/20103,20203 用ライブラリ対応のための追加修正 ・ __GetADCAll の R8C/2C,2D 用の記述の誤記を訂正 ・ __GetInputCapture の R8C のタイマ RC 使用時の data2 の説明を訂正 ・ __CreateOutputCompare の R8C のタイマ RE のカウントソースと出力機能の省略値の説明を	08.03.03

	訂正	
1.05	・ H8S の RAPI_TMR_TRIGGER を削除 ・ A/D 変換器の H8S の繰り返しモードの割り込み要求の項目を削除 ・ A/D 変換器の H8S の指定できる端子数を訂正 ・ H8S の RAPI_TIMER_ON および RAPI_TIMER_OFF の説明で API と一致していない箇所を修正 ・ __CreateOutputCompare の H8S のタイマ RG の data3 の説明を修正 ・ H8S の TimerMode のゲート機能を削除 ・ H8S の EventCounterMode、PulsePeriodMeasurementMode および、PulseWidthMeasurementMode にゲート機能を追加 ・ H8、H8S の__BasicSetSerialFormat の RAPI_BCSS_F32 を RAPI_BCSS_F64 に修正 ・ H8S のタイマの設定で、カウントソースφ/40 となっている説明をφ40 へ修正 ・ H8S のタイマ設定で、出力機能およびフィルタ機能の周波数の誤記を訂正 ・ __CreatePulseWidthModulationMode の H8S で RAPI_H_L_L と RAPI_L_H_H の説明が抜けていたのを追加修正	08.05.08
1.06	・ A/D 変換器の API に DestroyADCUnit、GetADCUnit、GetADCUnitStatus および、ClearADCUnitStatus を追加 ・ __GetADC および、__GetADCAll の R8C/2C,2D 用の記述を修正 ・ __ConfigurePMC および、PMC に関する項目を追加 ・ __SetEventSequence、__CreateEventGenerateTimer、__EnableEventGenerateTimer、__DestroyEventGenerateTimer、__WritePortBufferRegister、__ReadPortBufferRegister および、ELC に関する項目を追加 ・ H8S の AD の変換速度 330 ステートと 166 ステートを削除	08.06.18
1.07	・ PMC および ELC のプログラム例のヘッダファイル名を訂正 ・ __SetEventSequence→__SetEventLink へ変更 ・ 「2.1 概要」に PMC および ELC 用のヘッダファイルの説明を追加 ・ 「2.1 概要」の Interrupt_*xxx_yyy*.h→Interrupt_*xxx_yyy*.c へ訂正 ・ H8S の__CreateADC にユニットの定義値と説明を追加 ・ H8S の__CreateADC に定義値 RAPI_ADTRG2_TRIGGER を追加 ・ H8S の__EnableADC にユニットの定義値と説明を追加 ・ H8S の__GetADC にユニットの定義値と説明を追加 ・ H8S のタイマ RD のユニット名修正 ・ H8S の__CreateInputCapture に割り込み制御モードの説明抜けを修正 ・ PMC の定義値および説明を修正 ・ ELC の定義値および説明を修正	08.07.14
1.08	・ __SetEventLink の解説の説明を追加	08.07.18
1.09	・ タイマ ELC イベント信号の番号を修正	08.07.29
1.10	・ ELC の入出力ポートグループ番号を訂正(グループ 2→グループ 1、グループ 3→グループ 2)	08.11.10
1.11	・ データトランスファコントローラ機能追加	08.12.15
1.12	・ __ConfigurePMC の定義値の説明に I/O ポート指定方法を追加。	08.03.06

目次

1. はじめに 1
2. ドライバ 2
2.1 概要 2
2.2 ドライバの機能 3
2.3 シリアルI/Fドライバ 4
2.4 タイマドライバ 5
2.4.1 タイマモード 5
2.4.2 イベントカウンタモード 5
2.4.3 パルス幅変調モード（PWMモード） 5
2.4.4 パルス周期測定モード 5
2.4.5 パルス幅測定モード 5
2.4.6 インプットキャプチャモード 5
2.4.7 アウトプットコンペアモード 6
2.5 I/Oポートドライバ 7
2.6 外部割り込みドライバ 8
2.7 A/D変換器ドライバ 9
2.8 周辺機能マッピングコントローラドライバ 11
2.9 イベントリンクコントローラドライバ 12
2.10 データトランスファコントローラドライバ 13
3. 標準の型 14
4. ライブラリリファレンス 15
4.1 機能別API一覧 15
4.2 各APIの解説 18
4.2.1 シリアルI/O 19
_BasicOpenSerialDriver 19
_BasicCloseSerialDriver 20
_BasicSetSerialFormat 21
_BasicStartSerialReceiving 26
_BasicStartSerialSending 27
_BasicReceivingStatusRead 28
_BasicSendingStatusRead 30
_BasicStopSerialReceiving 31
_BasicStopSerialSending 32
_OpenSerialDriver 33
_CloseSerialDriver 34
_ConfigSerialDriverNotify 35
_SetSerialFormat 38
_SetSerialInterrupt 39
_StartSerialReceiving 42
_StartSerialSending 44
_StopSerialReceiving 46
_StopSerialSending 47

_PollingSerialReceiving....48
_PollingSerialSending....50
4.2.2 タイマ....52
_CreateTimer....52
_EnableTimer....59
_DestroyTimer....61
_CreateEventCounter....63
_EnableEventCounter....69
_DestroyEventCounter....71
_GetEventCounter....73
_CreatePulseWidthModulationMode....75
_EnablePulseWidthModulationMode....81
_DestroyPulseWidthModulationMode....83
_CreatePulsePeriodMeasurementMode....84
_EnablePulsePeriodMeasurementMode....90
_DestroyPulsePeriodMeasurementMode....92
_GetPulsePeriodMeasurementMode....94
_CreatePulseWidthMeasurementMode....96
_EnablePulseWidthMeasurementMode....102
_DestroyPulseWidthMeasurementMode....104
_GetPulseWidthMeasurementMode....106
_CreateInputCapture....108
_EnableInputCapture....124
_DestroyInputCapture....126
_GetInputCapture....128
_CreateOutputCompare....131
_EnableOutputCompare....149
_DestroyOutputCompare....151
_SetTimerRegister....153
_EnableTimerRegister....162
_ClearTimerRegister....164
_GetTimerRegister....166
4.2.3 I/Oポート....175
_SetIOPort....175
_ReadIOPort....181
_WriteIOPort....186
_SetIOPortRegister....191
_ReadIOPortRegister....194
_WriteIOPortRegister....196
4.2.4 外部割り込み....198
_SetInterrupt....198
_EnableInterrupt....203
_GetInterruptFlag....205
_ClearInterruptFlag....207
4.2.5 A/D変換器....209

_CreateADC 209
_EnableADC 221
_DestroyADC 227
_DestroyADCUnit 228
_GetADC 229
_GetADCAll 231
_GetADCUnit 233
_GetADCStatus 235
_GetADCUnitStatus 236
_ClearADCStatus 238
_ClearADCUnitStatus 239
4.2.6 周辺機能マッピングコントローラ 241
_ConfigurePMC 241
4.2.7 イベントリンクコントローラ 245
_SetEventLink 245
_CreateEventGenerateTimer 254
_EnableEventGenerateTimer 258
_DestroyEventGenerateTimer 259
_ReadPortBufferRegister 260
_WritePortBufferRegister 261
4.2.8 データトランスファコントローラ 262
_CreateDTC 262
_EnableDTC 264
_DisableDTC 266
_EnableSWDTC 268

1. はじめに

Renesas Embedded Application Programming Interface(API)は、㈱ルネサステクノロジ製マイクロコンピュータ用統一 API です。

2. ドライバ

2.1 概要

本ライブラリでは、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。Renesas Embedded API を利用することで、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できます。

Renesas Embedded API の構成を次に示します。

ファイル名	説明
rapi_*xxx_yyy*.lib (*xxx* = ファミリ名、*yyy* = シリーズ名)	Renesas Embedded API ライブラリファイルです。Renesas Embedded API を使用する場合は、本ファイルをリンカの入力ファイルとして指定してください。
rapi_ad_*xxx_yyy*.h (*xxx* = ファミリ名、*yyy* = シリーズ名)	A/D 変換器ドライバ API 用ヘッダファイルです。A/D 変換器ドライバ API を使用する場合は、本ファイルをインクルードする必要があります。
rapi_io_port_*xxx_yyy*.h (*xxx* = ファミリ名、*yyy* = シリーズ名)	I/O ポートドライバ API 用ヘッダファイルです。I/O ポートドライバ API を使用する場合は、本ファイルをインクルードする必要があります。
rapi_sif_*xxx_yyy*.h (*xxx* = ファミリ名、*yyy* = シリーズ名)	シリアル I/F ドライバ API 用ヘッダファイルです。シリアル I/F ドライバ API を使用する場合は、本ファイルをインクルードする必要があります。
rapi_timer_*xxx_yyy*.h (*xxx* = ファミリ名、*yyy* = シリーズ名)	タイマドライバ API 用ヘッダファイルです。タイマドライバ API を使用する場合は、本ファイルをインクルードする必要があります。
rapi_pmc_*xxx_yyy*.h (*xxx* = ファミリ名、*yyy* = シリーズ名)	周辺機能マッピングコントローラドライバ API 用ヘッダファイルです。周辺機能マッピングコントローラドライバ API を使用する場合は、本ファイルをインクルードする必要があります。
rapi_elc_*xxx_yyy*.h (*xxx* = ファミリ名、*yyy* = シリーズ名)	イベントリンクコントローラドライバ API 用ヘッダファイルです。イベントリンクコントローラドライバ API を使用する場合は、本ファイルをインクルードする必要があります。
rapi_io_*xxx_yyy*.h (*xxx* = ファミリ名、*yyy* = シリーズ名)	Renesas Embedded API 用 CPU 制御レジスタ定義ヘッダファイルです。
Interrupt_*xxx_yyy*.c (*xxx* = ファミリ名、*yyy* = シリーズ名)	Renesas Embedded API 用割り込み関数ソースファイルです。割り込みを使用する Renesas Embedded API を使用する場合は、本ファイルをユーザプログラムに追加してください。

2.2 ドライバの機能

本ライブラリでは、ペリフェラルドライバとして次の機能を用意しています。

(1) シリアル I/O の設定、制御

シリアル I/F ドライバでは、シリアル通信について、通信条件の設定・解除および、通信データの送受信制御や管理をおこなう機能を提供します。

(2) タイマの設定、制御

タイマドライバでは、タイマについて、動作条件の設定・解除および、タイマの動作制御をおこなう機能を提供します。

(3) I/O ポートの設定、制御

I/O ポートドライバでは、I/O ポートについて、使用条件の設定・解除および、データのリード・ライト制御をおこなう機能を提供します。

(4) 外部割り込みの設定、制御

外部割り込みドライバでは、外部割り込みについて、使用条件の設定・解除および、割り込みの動作制御をおこなう機能を提供します。

(5) A/D 変換器の設定、制御

A/D 変換器ドライバでは、A/D 変換器について、使用条件の設定・解除および、A/D 変換器の動作制御をおこなう機能を提供します。

(6) 周辺機能マッピングコントローラの設定、制御

周辺機能マッピングコントローラドライバでは、周辺機能マッピングコントローラについて、各端子への機能の割り当ての設定をおこなう機能を提供します。

(7) イベントリンクコントローラの設定、制御

イベントリンクコントローラドライバでは、イベントリンクコントローラについて、各周辺モジュールへのリンクするイベント信号の設定、イベント発生タイマの動作条件の設定・解除および、タイマの動作制御および、ポートバッファレジスタのリード・ライト制御をおこなう機能を提供します。

(8) データトランスファコントローラの設定、制御

データトランスファコントローラドライバでは、データトランスファコントローラについて、データ転送機能を提供します。

2.3 シリアルI/Fドライバ

シリアル I/F ドライバでは、シリアル通信の設定、設定破棄、送信、受信および、シリアル通信の状態制御の機能を提供します。

シリアル I/F ドライバは 1 データ送受信用 API と複数データ送受信用 API の 2 種類があります。

2.4 タイマドライバ

タイマドライバでは、次のモードについて、設定、設定破棄、動作制御および、カウンタ値取得などの機能を提供します。

- 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。アンダフロー割り込み発生時または、オーバフロー割り込み発生時には、設定したコールバック関数をコールします。

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、入力端子から入力した外部信号、他のタイマのオーバフローまたは、他のタイマのアンダフローをカウントするモードです。アンダフロー割り込み発生時または、オーバフロー割り込み発生時には、設定したコールバック関数をコールします。

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。アンダフロー割り込み発生時または、オーバフロー割り込み発生時には、設定したコールバック関数をコールします。

2.4.4 パルス周期測定モード

パルス周期測定モードは、入力端子から入力した外部信号のパルス周期を測定するモードです。アンダフロー割り込み発生時または、オーバフロー割り込み発生時には、設定したコールバック関数をコールします。

2.4.5 パルス幅測定モード

パルス幅測定モードは、入力端子から入力した外部信号のパルス幅を測定するモードです。アンダフロー割り込み発生時または、オーバフロー割り込み発生時には、設定したコールバック関数をコールします。

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、入力端子へのエッジ入力または、クロックをトリガとしてタイマの値をラッチし、割り込み要求を発生するモードです。インプットキャプチャ割り込み発生時、アンダフロー割り込み発生時または、オーバフロー割り込み発生時には、設定したコールバック関数をコールします。

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、割り込み要求を発生するモードです。コンペアマッチ割り込み発生時、アンダフロー割り込み発生時または、オーバフロー割り込み発生時には、設定したコールバック関数をコールします。

2.5 I/Oポートドライバ

I/O ポートドライバでは、I/O ポートの入出力設定、I/O ポートへのデータの書き込みおよび、I/O ポートからのデータの読み出しの機能を提供します。

2.6 外部割り込みドライバ

外部割り込みドライバでは、外部割り込みの設定、外部割り込みの制御、外部割り込みフラグの状態取得および、外部割り込みフラグのクリアの機能を提供します。

2.7 A/D変換器ドライバ

A/D 変換器ドライバでは、A/D 変換器の設定、A/D 変換器の制御、A/D 変換器の設定破棄、A/D 変換値取得、A/D 変換器の状態取得および、A/D 変換器の状態クリアの機能を提供します。

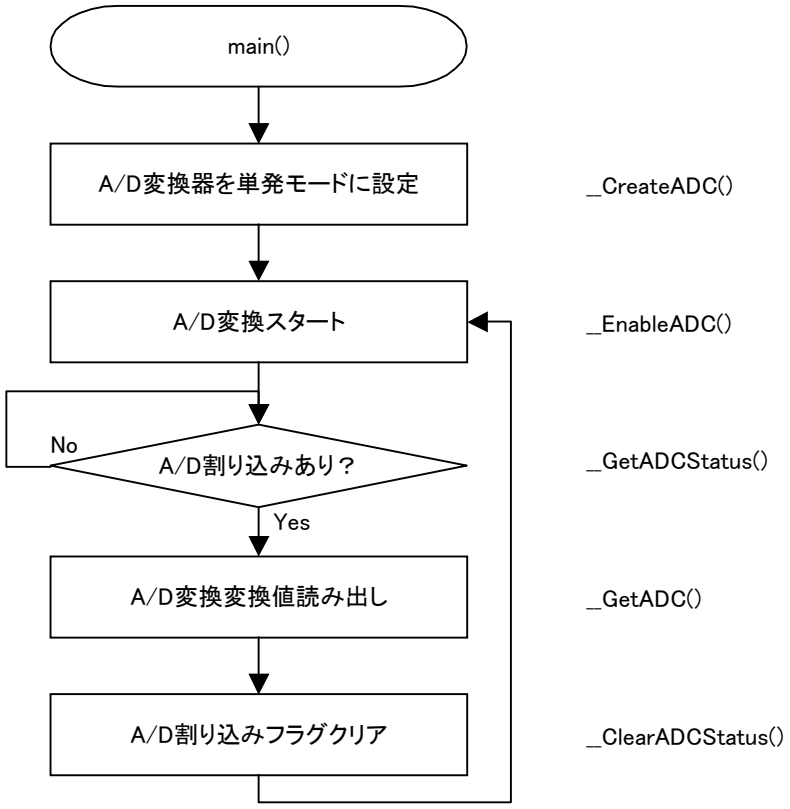
[A/D 変換器ドライバ使用例(単発モード)]

ここでは、以下の条件の A/D 変換器の単発モードについて、プログラム例を示します。

CPU	: R8C
動作クロック	: fAD の 2 分周
分解能	: 10 ビット
アナログ入力端子	: AN0 端子
A/D 変換開始条件	: ソフトウェアトリガ
サンプル&ホールド	: あり

以下に、プログラムの流れとプログラム例を示します。

(プログラムの流れ)



プログラム例

```
#include "rapi_ad_r8c_13.h"

void main( void )
{
    unsigned int  status, data;

    /* Set up A/D converter as one short mode */
    __CreateADC( RAPI_ONE_SHOT|RAPI_AN0|RAPI_FAD2| RAPI_WITH_SAMPLE_HOLD|
                 RAPI_AD_OFF|RAPI_10BIT, 1, 0, 0 );

    while( 1 ){
        /* Disable A/D converter */
        __EnableADC( RAPI_AN0| RAPI_AD_ON, 1 );

        /* Check a flag bit of A/D converter interrupt */
        do{
            __GetADCStatus( &status );
        } while( (*status & 0x0001) == 0 )

        /* Get A/D convertered datas of A/D register */
        __GetADC( &data );

        /* Clear status of A/D convertered */
        __ClearADCStatus( 0 );
    }
}
```

2.8 周辺機能マッピングコントローラドライバ

周辺機能マッピングコントローラドライバでは、各端子への機能の割り当ての設定をおこなう機能を提供します。

2.9 イベントリンクコントローラドライバ

イベントリンクコントローラドライバでは、各周辺モジュールへのリンクするイベント信号の設定、イベント発生タイマの動作条件の設定、イベント発生タイマの動作条件の設定破棄、イベント発生タイマの動作制御、ポートバッファデータの読み出しおよび、ポートバッファへのデータの書き込みをおこなう機能を提供します。

2.10 データトランスファコントローラドライバ

データトランスファコントローラドライバでは、データ転送機能を提供します。

3. 標準の型

本ライブラリで定義されている標準の型について説明します。設定値については各 API の説明を参照してください。

標準の型	説明
Boolean	Boolean 型は、成功(RAPI_TRUE(=1))および失敗(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一覧

次に、Renesas Embedded を構成する API を機能別に示します。

NO	機能分類	API	API 機能
1	シリアル I/O 1 データ用	__BasicOpenSerialDriver	シリアルポートのオープン
2		__BasicCloseSerialDriver	シリアルポートのクローズ
3		__BasicSetSerialFormat	シリアル通信設定
4		__BasicStartSerialReceiving	1 データの受信
5		__BasicStartSerialSending	1 データの送信
6		__BasicReceivingStatusRead	受信状態の読み込み
7		__BasicSendingStatusRead	送信状態の読み込み
8		__BasicStopSerialReceiving	受信の停止
9		__BasicStopSerialSending	送信の停止
10	シリアル I/O 複数データ用	__OpenSerialDriver	シリアルポートのオープン
11		__CloseSerialDriver	シリアルポートのクローズ
12		__ConfigSerialDriverNotify	通知関数の登録
13		__SetSerialFormat	シリアル通信設定
14		__SetSerialInterrupt	送受信割り込み設定
15		__StartSerialReceiving	受信の開始
16		__StartSerialSending	送信の開始
17		__StopSerialReceiving	受信の停止
18		__StopSerialSending	送信の停止
19		__PollingSerialReceiving	ポーリング受信
20		__PollingSerialSending	ポーリング送信
21	タイマ	__CreateTimer	タイマモード設定
22		__EnableTimer	タイマモード動作制御
23		__DestroyTimer	タイマモード設定破棄
24		__CreateEventCounter	イベントカウンタモード設定
25		__EnableEventCounter	イベントカウンタモード動作制御
26		__DestroyEventCounter	イベントカウンタモード設定破棄
27		__GetEventCounter	イベントカウンタモード カウンタ値取得
28		__CreatePulseWidthModulationMode	パルス幅変調モード設定
29		__EnablePulseWidthModulationMode	パルス幅変調モード動作制御
30		__DestroyPulseWidthModulationMode	パルス幅変調モード設定破棄
31		__CreatePulsePeriodMeasurementMode	パルス周期測定モード設定
32		__EnablePulsePeriodMeasurementMode	パルス周期測定モード動作制御
33		__DestroyPulsePeriodMeasurementMode	パルス周期測定モード設定破棄
34		__GetPulsePeriodMeasurementMode	パルス周期測定モード測定値取得
35		__CreatePulseWidthMeasurementMode	パルス幅測定モード設定
36		__EnablePulseWidthMeasurementMode	パルス幅測定モード動作制御

37	タイマ	__DestroyPulseWidthMeasurementMode	パルス幅測定モード設定破棄
38		__GetPulseWidthMeasurementMode	パルス幅測定モード測定値取得
39		__CreateInputCapture	インプットキャプチャモード設定
40		__EnableInputCapture	インプットキャプチャモード動作制御
41		__DestroyInputCapture	インプットキャプチャモード設定破棄
42		__GetInputCapture	インプットキャプチャモード カウンタ値取得
43		__CreateOutputCompare	アウトプットコンペアモード設定
44		__EnableOutputCompare	アウトプットコンペアモード動作制御
45		__DestroyOutputCompare	アウトプットコンペアモード設定破棄
46		__SetTimerRegister	タイマレジスタ設定
47		__EnableTimerRegister	タイマレジスタ動作制御
48		__ClearTimerRegister	タイマレジスタクリア
49		__GetTimerRegister	タイマレジスタ値取得
50	I/O ポート	__SetIOPort	I/O ポートの設定
51		__ReadIOPort	I/O ポートからの読み出し
52		__WriteIOPort	I/O ポートへの書き込み
53		__SetIOPortRegister	I/O ポートレジスタの設定
54		__ReadIOPortRegister	I/O ポートレジスタからの読み出し
55		__WriteIOPortRegister	I/O ポートレジスタへの書き込み
56	外部 割り込み	__SetInterrupt	外部割り込みの設定
57		__EnableInterrupt	外部割り込みの制御
58		__GetInterruptFlag	外部割り込みフラグの状態取得
59		__ClearInterruptFlag	外部割り込みフラグのクリア
60	A/D 変換器	__CreateADC	A/D 変換器の設定
61		__EnableADC	A/D 変換器の動作制御
62		__DestroyADC	A/D 変換器の設定破棄
63		__DestroyADCUnit	A/D 変換器の設定破棄（ユニット指定）
64		__GetADC	A/D 変換値の取得（レジスタ指定）
65		__GetADCAll	A/D 変換値の取得（全レジスタ）
66		__GetADCUnit	A/D 変換値の取得（ユニット指定）
67		__GetADCStatus	A/D 変換器の状態の取得
68		__GetADCUnitStatus	A/D 変換器の状態の取得 （ユニット指定）
69		__ClearADCStatus	A/D 変換器の状態のクリア
70		__ClearADCUnitStatus	A/D 変換器の状態のクリア （ユニット指定）
71	周辺機能マッピングコントローラ	__ConfigurePMC	各端子への機能の割り当ての設定
72	イベントリンクコントローラ	__SetEventLink	各周辺モジュールへリンクする イベント信号の設定
73		__CreateEventGenerateTimer	イベント発生タイマの動作条件の設定

74	イベントリンクコントローラ	__EnableEventGenerateTimer	イベント発生タイマの動作制御
75		__DestroyEventGenerateTimer	イベント発生タイマの動作条件の設定破棄
76		__ReadPortBufferRegister	ポートバッファデータの読み出し
77		__WritePortBufferRegister	ポートバッファへのデータの書き込み
78	データトランスファコントローラ	__CreateDTC	DTC レジスタ情報の設定
79		__EnableDTC	DTC 割り込み起動有効設定
80		__DisableDTC	DTC 割り込み起動無効設定
81		__EnableSWDTC	DTC ソフトウェア起動設定

4.2 各APIの解説

各 API の解説、使い方および、プログラム例について説明します。
各 API の解説は、以下の項目に分けられます。

- **要約**　　：関数の処理内容について要約します。また構文を示し、引数について簡単に説明します。
- **解説**　　：関数とその使い方についてさらに詳しく説明します。
- **戻り値**　　：関数の戻り値について解説します。
- **機能**　　：関数の機能分類を示します。
- **参照**　　：関連する関数を示します。
- **備考**　　：API の使用上の注意等を示します。
- **プログラム例**　：関数の使い方を示すプログラムを示します。

4.2.1 シリアルI/O
__BasicOpenSerialDriver

要約

〈 シリアルポートのオープン 〉

Boolean __BasicOpenSerialDriver(unsigned long data)

data	設定データ

解説

指定したシリアルポートのオープンと初期化をおこないます。

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S) (H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

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

機能

シリアル I/O

参照

__BasicCloseSerialDriver

備考

・ 指定できるシリアルポートは CPU により異なります。
・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ H8S および H8/300H で使用する場合、指定したシリアルポートのモジュールスタンバイ解除後に、シリアルポートのオープンと初期化をおこないます。

プログラム例

```
#include ” rapi_sif_r8c_13.h”

void func( void )
{
    ..........
    /* Open serial driver */
    __BasicOpenSerialDriver( RAPI_COM1 );
    ..........
}
```

__BasicCloseSerialDriver

要約

〈 シリアルポートのクローズ 〉

Boolean __BasicCloseSerialDriver(unsigned long data)

data	設定データ

解説

指定したシリアルポートをクローズします。data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

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

機能

シリアル I/O

参照

__BasicOpenSerialDriver

備考

・ 指定できるシリアルポートは CPU により異なります。
・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ H8S および H8/300H で使用する場合、指定したシリアルポートをクローズ後、モジュールスタンバイ状態に設定します。

プログラム例

```
#include  " rapi_sif_r8c_13.h"

void func( void )
{
    ..........
    /* Close serial driver */
    __BasicCloseSerialDriver( RAPI_COM1 );
    ..........
}
```

__BasicSetSerialFormat

要約

< シリアル通信設定 >

Boolean __BasicSetSerialFormat(unsigned long data1, unsigned char data2)

data1	設定データ 1
data2	設定データ 2

解説

指定したパラメータにしたがい、シリアル通信の設定をおこないます。

[data1]

data1 には次の値を設定することができます。複数の定義値を同時に設定する場合は、”|”記号を使用して指定します。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

シリアル通信モードとして、次の値を設定することができます。

(M16C)(UART0,UART1,UART2)

RAPI_SM_SYNC	クロック同期シリアル通信モード
RAPI_SM_ASYNC	クロック非同期シリアル通信モード

(M16C)(SI/O3,SI/O4)

RAPI_SIO_SM_SYNC	クロック同期シリアル通信モード

クロック非同期シリアル通信のデータ長フォーマットとして、次の値を設定することができます。

クロック同期シリアル通信モードで使用する場合は、設定しないでください。

(M16C)(UART0,UART1,UART2)

RAPI_BIT_7	転送データ長 7 ビット	RAPI_BIT_8	転送データ長 8 ビット
RAPI_BIT_9	転送データ長 9 ビット		

シリアル通信のクロックソースとして、次の値を設定することができます。

(M16C)(UART0,UART1,UART2)

RAPI_CKDIR_INT	シリアル通信に内部クロックをクロックソースとして使用
RAPI_CKDIR_EXT	シリアル通信に外部クロックをクロックソースとして使用

(M16C)(SI/O3,SI/O4)

RAPI_SIO_CKDIR_INT	シリアル通信に内部クロックをクロックソースとして使用
RAPI_SIO_CKDIR_EXT	シリアル通信に外部クロックをクロックソースとして使用

クロック非同期シリアル通信のストップビット長として、次の値を設定することができます。

クロック同期シリアル通信モードで使用する場合は、設定しないでください。

(M16C)(UART0,UART1,UART2)

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

クロック非同期シリアル通信のパリティビットについて、次の値を設定することができます。

クロック同期シリアル通信モードで使用する場合は、設定しないでください。

(M16C)(UART0,UART1,UART2)

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		

シリアル通信のクロック極性として、次の値を設定することができます。
クロック非同期シリアル通信モードで使用する場合は、設定しないでください。
(M16C) (UART0, UART1, UART2)

RAPI_DPOL_NON	極性反転なし	RAPI_DPOL_INV	極性反転あり

(M16C) (SI/O3, SI/O4)

RAPI_SIO_DPOL_NON	極性反転なし	RAPI_SIO_DPOL_INV	極性反転あり

内蔵ボーレートジェネレータのカウントソースとして、次の値を設定することができます。
(M16C) (UART0, UART1, UART2)

RAPI_BCSS_F1	f1SIO	RAPI_BCSS_F2	f2SIO
RAPI_BCSS_F8	f8SIO	RAPI_BCSS_F32	f32SIO

(M16C) (SI/O3, SI/O4)

RAPI_SIO_BCSS_F1	f1SIO	RAPI_SIO_BCSS_F2	f2SIO
RAPI_SIO_BCSS_F8	f8SIO	RAPI_SIO_BCSS_F32	f32SIO

_CTS/_RTS 機能について、次の値を設定することができます。
クロック同期シリアル通信モードの内部クロック選択時は、_RTS 機能は無効です。
(M16C) (UART0, UART1, UART2)

RAPI_CTSRTS_DIS	_CTS/_RTS 機能を使用しない
RAPI_CTS_SEL	_CTS 機能を選択
RAPI_RTS_SEL	_RTS 機能を選択

転送フォーマットとして、次の値を設定することができます。
クロック非同期シリアル通信モードのデータ長 7 ビットまたはデータ長 9 ビットで
使用する場合は、設定しないでください。
(M16C) (UART0, UART1, UART2)

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

(M16C) (SI/O3, SI/O4)

RAPI_SIO_LSB_SEL	LSB ファースト	RAPI_SIO_MSB_SEL	MSB ファースト

シリアルデータ論理切り替えとして、次の値を設定することができます。
(M16C) (UART2)

RAPI_LOGIC_NO_REV	送信バッファレジスタに書いた値の論理の反転はおこないません
RAPI_LOGIC_REV	送信バッファレジスタに書いた値の反転を送信します

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

シリアル通信モードとして、次の値を設定することができます。
(R8C) (UART0, UART1, UART2)

RAPI_SM_SYNC	クロック同期シリアル通信モード
RAPI_SM_ASYNC	クロック非同期シリアル通信モード

クロック非同期シリアル通信のデータ長フォーマットとして、次の値を設定することが
できます。
クロック同期シリアル通信モードで使用する場合は、設定しないでください。

(R8C) (UART0, UART1, UART2)

RAPI_BIT_7	転送データ長 7 ビット	RAPI_BIT_8	転送データ長 8 ビット
RAPI_BIT_9	転送データ長 9 ビット		

シリアル通信のクロックソースとして、次の値を設定することができます。

(R8C) (UART0, UART1, UART2)

RAPI_CKDIR_INT	シリアル通信に内部クロックをクロックソースとして使用
RAPI_CKDIR_EXT	シリアル通信に外部クロックをクロックソースとして使用

クロック非同期シリアル通信のストップビット長として、次の値を設定することができます。

クロック同期シリアル通信モードで使用する場合は、設定しないでください。

(R8C) (UART0, UART1, UART2)

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

クロック非同期シリアル通信のパリティビットについて、次の値を設定することができます。

クロック同期シリアル通信モードで使用する場合は、設定しないでください。

(R8C) (UART0, UART1, UART2)

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		

シリアル通信のクロック極性として、次の値を設定することができます。

クロック非同期シリアル通信モードで使用する場合は、設定しないでください。

(R8C) (UART0, UART1, UART2)

RAPI_DPOL_NON	極性反転なし	RAPI_DPOL_INV	極性反転あり

内蔵ボーレートジェネレータのカウントソースとして、次の値を設定することができます。

(R8C) (UART0, UART1, UART2)

RAPI_BCSS_F1	f1SIO	RAPI_BCSS_F8	f8SIO
RAPI_BCSS_F32	f32SIO		

転送フォーマットとして、次の値を設定することができます。

クロック非同期シリアル通信モードのデータ長 7 ビットまたはデータ長 9 ビットで使用する場合は、設定しないでください。

(R8C) (UART0, UART1, UART2)

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

シリアル通信で使用する端子として、次の値を設定することができます。

(R8C) (UART1)

RAPI_TX_P00_RX_P37	TXD1 を P0_0 に、RXD1 を P3_7 に設定
RAPI_TX_P37_RX_P45	TXD1 を P3_7 に、RXD1 を P4_5 に設定
RAPI_TX_P00_RX_P36	TXD1 を P0_0 に、RXD1 を P3_6 に設定
RAPI_CLK1_P0_5	CLK1 を P0_5 に設定
RAPI_CLK1_P6_5	CLK1 を P6_5 に設定

(H8S)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

シリアル通信モードとして、次の値を設定することができます。

(H8S) (SCI3 チャネル 1, SCI3 チャネル 2, SCI3 チャネル 3)

RAPI_SM_SYNC	クロック同期シリアル通信モード
RAPI_SM_ASYNC	クロック非同期シリアル通信モード

クロック非同期シリアル通信のデータ長フォーマットとして、次の値を設定することができます。
クロック同期シリアル通信モードで使用する場合は、設定しないでください。
(H8S)(SCI3 チャネル 1,SCI3 チャネル 2,SCI3 チャネル 3)

RAPI_BIT_7	転送データ長 7 ビット	RAPI_BIT_8	転送データ長 8 ビット

シリアル通信のクロックソースとして、次の値を設定することができます。
(H8S)(SCI3 チャネル 1,SCI3 チャネル 2,SCI3 チャネル 3)

RAPI_CKDIR_INT	シリアル通信に内部クロックをクロックソースとして使用
RAPI_CKDIR_EXT	シリアル通信に外部クロックをクロックソースとして使用

クロック非同期シリアル通信のストップビット長として、次の値を設定することができます。
クロック同期シリアル通信モードで使用する場合は、設定しないでください。
(H8S)(SCI3 チャネル 1,SCI3 チャネル 2,SCI3 チャネル 3)

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

クロック非同期シリアル通信のパリティビットについて、次の値を設定することができます。
クロック同期シリアル通信モードで使用する場合は、設定しないでください。
(H8S)(SCI3 チャネル 1,SCI3 チャネル 2,SCI3 チャネル 3)

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		

内蔵ボーレートジェネレータのカウントソースとして、次の値を設定することができます。
(H8S)(SCI3 チャネル 1,SCI3 チャネル 2,SCI3 チャネル 3)

RAPI_BCSS_F1	φクロック	RAPI_BCSS_F4	φ/4 クロック
RAPI_BCSS_F16	φ/16 クロック	RAPI_BCSS_F64	φ/64 クロック

ノイズ除去機能のあり/なしを設定することができます。
クロック同期シリアル通信モードで使用する場合は、ノイズ除去機能は無効です。
(H8S)(SCI3 チャネル 1,SCI3 チャネル 2,SCI3 チャネル 3)

RAPI_NOISE_CANCEL_ON	ノイズ除去機能あり
RAPI_NOISE_CANCEL_OFF	ノイズ除去機能なし

(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

シリアル通信モードとして、次の値を設定することができます。
(H8/300H)(SCI3 チャネル 1,SCI3 チャネル 2,SCI3 チャネル 3)

RAPI_SM_SYNC	クロック同期シリアル通信モード
RAPI_SM_ASYNC	クロック非同期シリアル通信モード

クロック非同期シリアル通信のデータ長フォーマットとして、次の値を設定することができます。
クロック同期シリアル通信モードで使用する場合は、設定しないでください。
(H8/300H)(SCI3 チャネル 1,SCI3 チャネル 2,SCI3 チャネル 3)

RAPI_BIT_7	転送データ長 7 ビット	RAPI_BIT_8	転送データ長 8 ビット

シリアル通信のクロックソースとして、次の値を設定することができます。
(H8/300H)(SCI3 チャネル 1,SCI3 チャネル 2,SCI3 チャネル 3)

RAPI_CKDIR_INT	シリアル通信に内部クロックをクロックソースとして使用
RAPI_CKDIR_EXT	シリアル通信に外部クロックをクロックソースとして使用

クロック非同期シリアル通信のストップビット長として、次の値を設定することができます。
クロック同期シリアル通信モードで使用する場合は、設定しないでください。
(H8/300H)(SCI3 チャネル 1,SCI3 チャネル 2,SCI3 チャネル 3)

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

クロック非同期シリアル通信のパリティビットについて、次の値を設定することができます。
クロック同期シリアル通信モードで使用する場合は、設定しないでください。
(H8/300H)(SCI3 チャネル 1,SCI3 チャネル 2,SCI3 チャネル 3)

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		

内蔵ボーレートジェネレータのカウントソースとして、次の値を設定することができます。
(H8/300H)(SCI3 チャネル 1,SCI3 チャネル 2,SCI3 チャネル 3)

RAPI_BCSS_F1	φクロック	RAPI_BCSS_F4	φ/4 クロック
RAPI_BCSS_F16	φ/16 クロック	RAPI_BCSS_F64	φ/64 クロック

ノイズ除去機能のあり/なしを設定することができます。
クロック同期シリアル通信モードで使用する場合は、ノイズ除去機能は無効です。
(H8/300H)(SCI3 チャネル 3)

RAPI_NOISE_CANCEL_ON	ノイズ除去機能あり
RAPI_NOISE_CANCEL_OFF	ノイズ除去機能なし

[data2]
通信速度の分周値を設定します。

戻り値

シリアル通信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。

機能

シリアル I/O

備考

・ 指定できるシリアルポートは CPU により異なります。
・ 第 1 引数に定義値以外の値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_r8c_13.h"

Boolean func( void )
{
    /* Set the data of RAPI_COM1 to serial driver */
    return __BasicSetSerialFormat(RAPI_COM1 | RAPI_SM_SYNC | RAPI_CKDIR_INT |
                                  RAPI_BCSS_F1 | RAPI_DPOL_NON | RAPI_LSB_SEL, 20);
}
```

__BasicStartSerialReceiving

要約

〈 1データの受信 〉

Boolean __BasicStartSerialReceiving(unsigned long data)

data	設定データ

解説

シリアル通信の1データ受信を開始します。

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S) (H8/300H)

RAPI_COM1	SCI3 チャネル1	RAPI_COM2	SCI3 チャネル2
RAPI_COM3	SCI3 チャネル3		

戻り値

シリアル通信の受信開始を成功した場合は RAPI_TRUE を返し、失敗した場合は RAPI_FALSE を返します。

機能

シリアル I/O

参照

__BasicReceivingStatusRead, __BasicStopSerialReceiving

備考

- H8S および H8/300H では__BasicSetSerialFormat をコール後、少なくとも1ビット期間待ってから、本 API をコールしてください。
- 指定できるシリアルポートは CPU により異なります。
- 引数に定義値以外の値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_r8c_13.h”

void func( void )
{
    ..........
    __BasicStartSerialReceiving( RAPI_COM1 );
    ..........
}
```

__BasicStartSerialSending

要約

〈 1データの送信 〉

Boolean __BasicStartSerialSending(unsigned long data1, unsigned int data2)

data1	設定データ
data2	送信データ

解説

シリアル通信の1データ送信を開始します。

data1には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S)(H8/300H)

RAPI_COM1	SCI3チャネル1	RAPI_COM2	SCI3チャネル2
RAPI_COM3	SCI3チャネル3		

戻り値

シリアル通信の送信開始を成功した場合はRAPI_TRUEを返し、失敗した場合はRAPI_FALSEを返します。

機能

シリアルI/O

参照

__BasicSendingStatusRead, __BasicStopSerialSending

備考

・H8SおよびH8/300Hでは__BasicSetSerialFormatをコール後、少なくとも1ビット期間待ってから、本APIをコールしてください。
・指定できるシリアルポートはCPUにより異なります。
・第1引数に定義値以外の値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_r8c_13.h"

void func( void )
{
    ..........
    __BasicStartSerialSending( RAPI_COM1, 0x00AA );
    ..........
}
```

__BasicReceivingStatusRead

要約

< 受信状態の読み込み >

unsigned int __BasicReceivingStatusRead(unsigned long data)

data	設定データ

解説

シリアル通信の受信状態を返します。

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM1	SCI3 チャネル 3		

戻り値

シリアル通信の受信状態を返します。次の値を返します。

(M16C)(UART0,UART1,UART2)

RAPI_RX_INCOMPLETE	受信未完了。
上記以外	受信完了。UARTi 受信バッファレジスタ(i=0 to 2)を読み込んだ値。

(M16C)(SI/O3,SI/O4)

RAPI_RX_INCOMPLETE	受信未完了。
上記以外	受信完了。下位 8 ビット:SI/Oi 送受信レジスタ(i=3,4)を読み込んだ値。

(R8C)

RAPI_RX_INCOMPLETE	受信未完了。
上記以外	受信完了。UARTi 受信バッファレジスタ(i=0,1)を読み込んだ値。

(H8S)(H8/300H)

RAPI_RX_INCOMPLETE	受信未完了。
上記以外	受信完了。 上位 8 ビット：シリアルステータスレジスタを読み込んだ値。 下位 8 ビット：レシーブデータレジスタを読み込んだ値。 （エラー発生時は読み込みません。）

機能

シリアル I/O

参照

__BasicStartSerialReceiving, __BasicStopSerialReceiving

備考

・ 指定できるシリアルポートは CPU により異なります。

・ 引数に定義値以外の値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_r8c_13.h"

void func( void )
{
    unsigned int rcv_data;

    ..........
    rcv_data = __BasicReceivingStatusRead( RAPI_COM1 );
    ..........
}
```

__BasicSendingStatusRead

要約

< 送信状態の読み込み >

Boolean __BasicSendingStatusRead(unsigned long data)

data	設定データ

解説

シリアル通信の送信状態を返します。data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

送信バッファにデータなしの場合は RAPI_TRUE を返し、データありの場合は RAPI_FALSE を返します。

機能

シリアル I/O

参照

__BasicStartSerialSending, __BasicStopSerialSending

備考

・ 指定できるシリアルポートは CPU により異なります。
・ 引数に定義値以外の値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_r8c_13.h"

void func( void )
{
    ..........
    if (__BasicSendingStatusRead( RAPI_COM1 ) == RAPI_TRUE) {
        /* Transmission completion */
    }
    ..........
}
```

__BasicStopSerialReceiving

要約

< 受信の停止 >

Boolean __BasicStopSerialReceiving(unsigned long data)

data	設定データ

解説

シリアル通信の受信を停止します。

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

シリアル通信の受信の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。

機能

シリアル I/O

参照

__BasicStartSerialReceiving

備考

・M16C の SI/O3 および、SI/O4 では、本 API は使用できません。
・指定できるシリアルポートは CPU により異なります。
・引数に定義値以外の値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_r8c_13.h"

void func( void )
{
    ..........
    /* Stop receiving data in serial communication */
    __BasicStopSerialReceiving ( RAPI_COM1 );
    ..........
}
```

__BasicStopSerialSending

要約

< 送信の停止 >

Boolean __BasicStopSerialSending(unsigned long data)

data	設定データ

解説

シリアル通信の送信を停止します。

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

シリアル通信の送信の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。

機能

シリアル I/O

参照

__BasicStartSerialSending

備考

・M16C の SI/O3 および、SI/O4 では、本 API は使用できません。
・指定できるシリアルポートは CPU により異なります。
・クロック同期シリアル通信モード使用時は、本 API により受信も停止します。
・引数に定義値以外の値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_r8c_13.h"

void func( void )
{
    ..........
    /* Stop sending data in serial communication */
    __BasicStopSerialSending ( RAPI_COM1 );
    ..........
}
```

__OpenSerialDriver

要約

〈 シリアルポートのオープン 〉

Boolean __OpenSerialDriver(unsigned long data)

data	設定データ

解説

指定したシリアルポートのオープンと初期化をおこないます。

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

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

機能

シリアル I/O

参照

__CloseSerialDriver

備考

・ 指定できるシリアルポートは CPU により異なります。
・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ H8S および H8/300H で使用する場合、指定したシリアルポートのモジュールスタンバイ解除後、シリアルポートのオープンと初期化をおこないます。

プログラム例

```
#include  " rapi_sif_r8c_13.h"

void func( void )
{
    ..........
    /* Open serial driver */
    __OpenSerialDriver( RAPI_COM1 );
    ..........
}
```

__CloseSerialDriver

要約

〈 シリアルポートのクローズ 〉

Boolean __CloseSerialDriver(unsigned long data)

data	設定データ

解説

指定したシリアルポートのクローズをおこないます。

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

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

機能

シリアル I/O

参照

__OpenSerialDriver

備考

・ 指定できるシリアルポートは CPU により異なります。
・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ H8S および H8/300H で使用する場合、指定したシリアルポートをクローズ後、モジュールスタンバイ状態に設定します。

プログラム例

```
#include  " rapi_sif_r8c_13.h"

void func( void )
{
    ..........
    /* Close serial driver */
    __CloseSerialDriver( RAPI_COM1 );
    ..........
}
```

__ConfigSerialDriverNotify

要約

< 通知関数の登録 >

Boolean __ConfigSerialDriverNotify(unsigned long data, VoidFuncNotify *func)

data	設定データ
func	登録する関数ポインタ

解説

シリアル通信の各種送受信情報を取得するための通知関数を登録します。

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

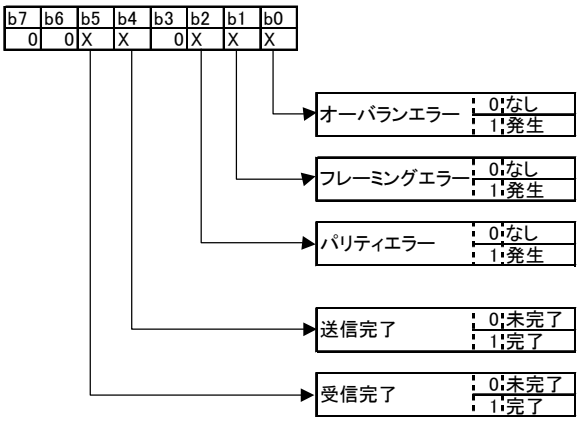
(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

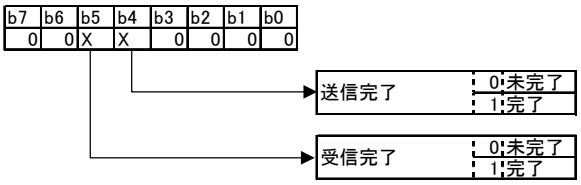
[func]

func に登録する関数は、シリアル I/O ドライバに対してユーザが
提供しなければなりません。
シリアル I/O ドライバは、func に登録された関数を呼び出します。
シリアル I/O ドライバは、送受信の状態を引数によりユーザに知らせます。
次に登録する関数の型を示します。
void “任意の関数名”(unsigned char notify);
次に引数の詳細を示します。

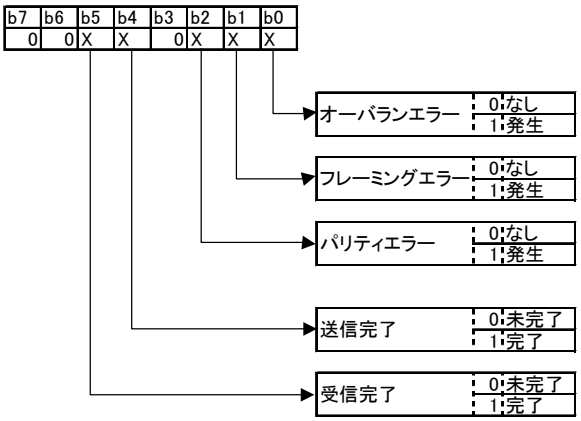
(M16C)(UART0,UART1,UART2)



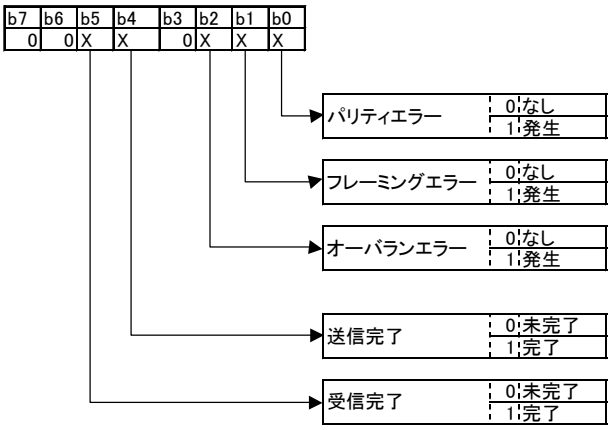
(M16C)(SI/03,SI/04)



(R8C)



(H8S)(H8/300H)



戻り値　シリアルポート指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能　シリアル I/O

参照　__StartSerialReceiving, __StartSerialSending

備考
- 指定できるシリアルポートは CPU により異なります。
- 第 1 引数に定義値以外の値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_r8c_13.h”
```

```
void Notify(unsigned char result) {
    if ((result&RAPI_OVER_ERR) == RAPI_OVER_ERR) {
        /* Overrun error */
    }
    if ((result&RAPI_FRAMING_ERR) == RAPI_FRAMING_ERR) {
        /* Framing error */
    }
    if ((result&RAPI_PARITY_ERR) == RAPI_PARITY_ERR) {
        /* Parity error */
    }
    if ((result&RAPI_TX_END) == RAPI_TX_END) {
        /* Transmission completion */
    }
    if ((result&RAPI_RX_END) == RAPI_RX_END) {
        /* Reception completion */
    }
}

Boolean func( void )
{
    ..........
    /* Set callback functions of RAPI_COM1 to serial driver */
    return __ConfigSerialDriverNotify( RAPI_COM1, Notify );
    ..........
}
```

__SetSerialFormat

要約

< シリアル通信設定 >

Boolean __SetSerialFormat(unsigned long data1, unsigned char data2)

data1	設定データ 1
data2	設定データ 2

解説

指定したパラメータにしたがい、シリアル通信の設定をおこないます。
パラメータの詳細は、__BasicSetSerialFormat の解説を参照してください。

戻り値

シリアル通信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。

機能

シリアル I/O

参照

__BasicSetSerialFormat

備考

・ 指定できるシリアルポートは CPU により異なります。
・ 第 1 引数に定義値以外の値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_r8c_13.h"

Boolean func( void )
{
    ..........
    /* Set the data of RAPI_COM1 to serial driver */
    return __SetSerialFormat(RAPI_COM1 | RAPI_SM_SYNC | RAPI_CKDIR_INT |
                             RAPI_BCSS_F1 | RAPI_DPOL_NON | RAPI_LSB_SEL, 20);
    ..........
}
```

__SetSerialInterrupt

要約

< シリアル割り込み設定 >

Boolean __SetSerialInterrupt(unsigned long data)

data	設定データ

解説

指定したパラメータにしたがい、割り込みの設定をおこないます。

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

割り込み設定として、次の値を設定することができます。

RAPI_INT_TX_DIS	送信割り込み禁止
RAPI_INT_TX_LV_1	送信割り込み優先レベル 1
RAPI_INT_TX_LV_2	送信割り込み優先レベル 2
RAPI_INT_TX_LV_3	送信割り込み優先レベル 3
RAPI_INT_TX_LV_4	送信割り込み優先レベル 4
RAPI_INT_TX_LV_5	送信割り込み優先レベル 5
RAPI_INT_TX_LV_6	送信割り込み優先レベル 6
RAPI_INT_TX_LV_7	送信割り込み優先レベル 7
RAPI_INT_RX_DIS	受信割り込み禁止
RAPI_INT_RX_LV_1	受信割り込み優先レベル 1
RAPI_INT_RX_LV_2	受信割り込み優先レベル 2
RAPI_INT_RX_LV_3	受信割り込み優先レベル 3
RAPI_INT_RX_LV_4	受信割り込み優先レベル 4
RAPI_INT_RX_LV_5	受信割り込み優先レベル 5
RAPI_INT_RX_LV_6	受信割り込み優先レベル 6
RAPI_INT_RX_LV_7	受信割り込み優先レベル 7

(M16C)(UART0,UART1,UART2)

RAPI_INT_TX_DIS	送信割り込み禁止
RAPI_INT_TX_LV_1	送信割り込み優先レベル 1
RAPI_INT_TX_LV_2	送信割り込み優先レベル 2
RAPI_INT_TX_LV_3	送信割り込み優先レベル 3
RAPI_INT_TX_LV_4	送信割り込み優先レベル 4
RAPI_INT_TX_LV_5	送信割り込み優先レベル 5
RAPI_INT_TX_LV_6	送信割り込み優先レベル 6
RAPI_INT_TX_LV_7	送信割り込み優先レベル 7
RAPI_INT_RX_DIS	受信割り込み禁止
RAPI_INT_RX_LV_1	受信割り込み優先レベル 1

RAPI_INT_RX_LV_2	受信割り込み優先レベル 2
RAPI_INT_RX_LV_3	受信割り込み優先レベル 3
RAPI_INT_RX_LV_4	受信割り込み優先レベル 4
RAPI_INT_RX_LV_5	受信割り込み優先レベル 5
RAPI_INT_RX_LV_6	受信割り込み優先レベル 6
RAPI_INT_RX_LV_7	受信割り込み優先レベル 7

(M16C)(SI/O3,SI/O4)

RAPI_INT_SIO_DIS	SI/O 割り込み禁止
RAPI_INT_SIO_LV_1	SI/O 割り込み優先レベル 1
RAPI_INT_SIO_LV_2	SI/O 割り込み優先レベル 2
RAPI_INT_SIO_LV_3	SI/O 割り込み優先レベル 3
RAPI_INT_SIO_LV_4	SI/O 割り込み優先レベル 4
RAPI_INT_SIO_LV_5	SI/O 割り込み優先レベル 5
RAPI_INT_SIO_LV_6	SI/O 割り込み優先レベル 6
RAPI_INT_SIO_LV_7	SI/O 割り込み優先レベル 7

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

割り込み設定として、次の値を設定することができます。

RAPI_INT_TX_DIS	送信割り込み禁止
RAPI_INT_TX_LV_1	送信割り込み優先レベル 1
RAPI_INT_TX_LV_2	送信割り込み優先レベル 2
RAPI_INT_TX_LV_3	送信割り込み優先レベル 3
RAPI_INT_TX_LV_4	送信割り込み優先レベル 4
RAPI_INT_TX_LV_5	送信割り込み優先レベル 5
RAPI_INT_TX_LV_6	送信割り込み優先レベル 6
RAPI_INT_TX_LV_7	送信割り込み優先レベル 7
RAPI_INT_RX_DIS	受信割り込み禁止
RAPI_INT_RX_LV_1	受信割り込み優先レベル 1
RAPI_INT_RX_LV_2	受信割り込み優先レベル 2
RAPI_INT_RX_LV_3	受信割り込み優先レベル 3
RAPI_INT_RX_LV_4	受信割り込み優先レベル 4
RAPI_INT_RX_LV_5	受信割り込み優先レベル 5
RAPI_INT_RX_LV_6	受信割り込み優先レベル 6
RAPI_INT_RX_LV_7	受信割り込み優先レベル 7

(H8S)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

割り込み設定として、次の値を設定することができます。

RAPI_INT_TX_DIS	送信割り込み禁止	RAPI_INT_TX_ENA	送信割り込み許可
RAPI_INT_RX_DIS	受信割り込み禁止	RAPI_INT_RX_ENA	受信割り込み許可

割り込みモードおよびレベルとして、次の値を設定することができます。

RAPI_INT_MODE_0	割り込み制御モード 0
RAPI_INT_LV_0	割り込み制御モード 2　送受信割り込み優先レベル 0
RAPI_INT_LV_1	割り込み制御モード 2　送受信割り込み優先レベル 1
RAPI_INT_LV_2	割り込み制御モード 2　送受信割り込み優先レベル 2
RAPI_INT_LV_3	割り込み制御モード 2　送受信割り込み優先レベル 3

(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

割り込み設定として、次の値を設定することができます。

RAPI_INT_TX_DIS	送信割り込み禁止	RAPI_INT_TX_ENA	送信割り込み許可
RAPI_INT_RX_DIS	受信割り込み禁止	RAPI_INT_RX_ENA	受信割り込み許可

割り込みコントロールレジスタを持つ CPU では、割り込み設定に加えて、
割り込みレベルとして、次の値を設定することができます。

RAPI_INT_LV_0	送受信割り込み優先レベル 0
RAPI_INT_LV_1	送受信割り込み優先レベル 1

戻り値

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

機能

シリアル I/O

備考

・ 指定できるシリアルポートは CPU により異なります。
・ 引数に定義値以外の値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_r8c_13.h”

Boolean func( void )
{
    ..........
    /* Set interrupt of RAPI_COM1 to serial driver */
    return __SetSerialInterrupt( RAPI_COM1 | RAPI_INT_TX_LV_1 | RAPI_INT_RX_LV_2 );
    ..........
}
```

__StartSerialReceiving

要約

< 受信の開始 >

Boolean __StartSerialReceiving(unsigned long data, unsigned char wordNum, unsigned int *RcvDtBuf)

data	設定データ
wordNum	受信ワード数
RcvDtBuf	受信データを格納するバッファポインタ

解説

シリアル通信の受信を開始し、受信したデータを指定したワード数だけ取得します。
受信データの取得完了時には、通知関数をコールします。(通知関数登録時)

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

シリアル通信の受信開始を成功した場合は RAPI_TRUE を返し、失敗した場合は RAPI_FALSE を返します。

機能

シリアル I/O

参照

__ConfigSerialDriverNotify, __StopSerialReceiving

備考

- H8S および H8/300H では__SetSerialFormat をコール後、少なくとも 1 ビット期間待ってから、本 API をコールしてください。
- 指定できるシリアルポートは CPU により異なります。
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- H8S および H8/300H では、次の値を受信バッファへ格納します。
  上位 8 ビット：シリアルステータスレジスタを読み込んだ値。
  下位 8 ビット：レシーブデータレジスタを読み込んだ値。
  （エラー発生時は読み込みません。）

プログラム例

```
#include ” rapi_sif_r8c_13.h”
```

```
unsigned int buffer[10];
void func( void )
{
    ..........
    /* Get 5 word data received in serial communication */
    __StartSerialReceiving( RAPI_COM1, 5, buffer );
    ..........
}
```

__StartSerialSending

要約

< 送信の開始 >

Boolean __StartSerialSending(unsigned long data, unsigned char wordNum, unsigned int *SndDtBuf)

data	設定データ
wordNum	送信バイト数
SndDtBuf	送信データポインタ

解説

シリアル通信の送信を開始し、指定したワード数だけ送信データの送信バッファへ書き込みます。すべての送信データの送信完了時には、通知関数をコールします。

(通知関数登録時)

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART1		

(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

シリアル通信の送信開始を成功した場合は RAPI_TRUE を返し、失敗した場合は RAPI_FALSE を返します。

機能

シリアル I/O

参照

__ConfigSerialDriverNotify, __StopSerialSending

備考

・H8S および H8/300H では__SetSerialFormat をコール後、少なくとも 1 ビット期間待ってから、本 API をコールしてください。
・指定できるシリアルポートは CPU により異なります。
・第 1 引数に定義値以外の値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_r8c_13.h"
```

```
unsigned int buffer[10];
void func( void )
{
    ..........
    /* Set 5 word data to transmit buffer of serial communication */
    __StartSerialSending( RAPI_COM1, 5, buffer );
    ..........
}
```

__StopSerialReceiving

要約

< 受信の停止 >

Boolean __StopSerialReceiving(unsigned long data)

data	設定データ

解説

シリアル通信の受信を停止します。

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

シリアル通信の受信の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。

機能

シリアル I/O

参照

__StartSerialReceiving

備考

・M16C の SI/O3 および、SI/O4 では、本 API は使用できません。
・指定できるシリアルポートは CPU により異なります。
・引数に定義値以外の値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_r8c_13.h"

void func( void )
{
    ..........
    /* Stop receiving data in serial communication */
    __StopSerialReceiving ( RAPI_COM1 );
    ..........
}
```

__StopSerialSending

要約

< 送信の停止 >

Boolean __StopSerialSending(unsigned long data)

data	設定データ

解説

シリアル通信の送信を停止します。

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S) (H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

シリアル通信の送信の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。

機能

シリアル I/O

参照

__StartSerialSending

備考

・ M16C の SI/O3 および、SI/O4 では、本 API は使用できません。
・ 指定できるシリアルポートは CPU により異なります。
・ クロック同期シリアル通信モード使用時は、本 API により受信も停止します。
・ 引数に定義値以外の値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_r8c_13.h"

void func( void )
{
    ..........
    /* Stop sending data in serial communication */
    __StopSerialSending ( RAPI_COM1 );
    ..........
}
```

__PollingSerialReceiving

要約

< ポーリング受信 >

unsigned int __PollingSerialReceiving(unsigned long data)

data	設定データ

解説

シリアル通信の受信をポーリングで行います。__StartSerialReceiving で指定したデータ数だけ取得します。受信データの取得完了時には、通知関数をコールします。

（通知関数登録時）

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

受信要求したデータ数のうち、未受信のデータ数を返します。

機能

シリアル I/O

参照

__ConfigSerialDriverNotify, __SetSerialInterrupt, __StartSerialReceiving

備考

・ 指定できるシリアルポートは CPU により異なります。

・ 引数に定義値以外の値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_r8c_13.h"
```

```
unsigned int buffer[10], count;

void func( void )
{
    ..........
    /* Reception interrupt disable */
    __SetSerialInterrupt( RAPI_COM1 | RAPI_INT_TX_DIS | RAPI_INT_RX_DIS );
    /* Start reception */
    __StartSerialReceiving( RAPI_COM1, 5, buffer );
    do {
        count = __PollingSerialReceiving( RAPI_COM1 );
    } while( count );
    ..........
}
```

__PollingSerialSending

要約

< ポーリング送信 >

unsigned int __PollingSerialSending(unsigned long data)

data	設定データ

解説

シリアル通信の送信をポーリングで行います。

__StartSerialSending で指定したデータ数だけ、__StartSerialSending で指定した送信バッファのデータを送信します。すべての送信データの送信完了時には、通知関数をコールします。(通知関数登録時)

[data]

data には次の値を設定することができます。

(M16C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2	RAPI_COM4	SI/O3
RAPI_COM5	SI/O4		

(R8C)

RAPI_COM1	UART0	RAPI_COM2	UART1
RAPI_COM3	UART2		

(H8S)(H8/300H)

RAPI_COM1	SCI3 チャネル 1	RAPI_COM2	SCI3 チャネル 2
RAPI_COM3	SCI3 チャネル 3		

戻り値

送信要求したデータ数のうち、未送信のデータ数を返します。

機能

シリアル I/O

参照

__ConfigSerialDriverNotify, __SetSerialInterrupt, __StartSerialSending

備考

・ 指定できるシリアルポートは CPU により異なります。
・ 引数に定義値以外の値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_r8c_13.h”
```

```
unsigned int buffer[10], count;
void func( void )
{
    ..........
    /* Transmission interrupt disable */
    __SetSerialInterrupt( RAPI_COM1 | RAPI_INT_TX_DIS | RAPI_INT_RX_DIS );
    /* Start transmission */
    __StartSerialSending( RAPI_COM1, 5, buffer );
    do {
        count = __PollingSerialSending( RAPI_COM1 );
    } while( count );
    ..........
}
```

4.2.2 タイマ
__CreateTimer

要約

< タイマモード設定 >

Boolean __CreateTimer(unsigned long data1, unsigned int data2, unsigned int data3, unsigned int data4, void* func)

data1	設定データ 1（機種により内容が異なる）
data2	設定データ 2（機種により内容が異なる）
data3	設定データ 3（機種により内容が異なる）
data4	設定データ 4（機種により内容が異なる）
func	コールバック関数ポインタ （コールバック関数を設定しない場合は 0 を指定する）

解説

指定したタイマをタイマモードに指定します。

[data1]

data1 には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を使用する。
RAPI_TIMER_A1	タイマ A チャネル 1 を使用する。
RAPI_TIMER_A2	タイマ A チャネル 2 を使用する。
RAPI_TIMER_A3	タイマ A チャネル 3 を使用する。
RAPI_TIMER_A4	タイマ A チャネル 4 を使用する。
RAPI_TIMER_B0	タイマ B チャネル 0 を使用する。
RAPI_TIMER_B1	タイマ B チャネル 1 を使用する。
RAPI_TIMER_B2	タイマ B チャネル 2 を使用する。
RAPI_F1	カウントソースに f_1 を選択する。
RAPI_F2	カウントソースに f_2 を選択する。
RAPI_F8	カウントソースに f_8 を選択する。
RAPI_F32	カウントソースに f_{32} を選択する。
RAPI_FC32	カウントソースに f_{C32} を選択する。
RAPI_TIMER_ON	__CreateTimer 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateTimer 内ではタイマを動作停止に設定する。
RAPI_GATE_L	TA_{iIN} 端子に"L"が入力されている期間カウントするゲート機能を選択する。
RAPI_GATE_H	TA_{iIN} 端子に"H"が入力されている期間カウントするゲート機能を選択する。
RAPI_PULSE_ON	TA_{iIN} 端子からのパルス出力ありを選択する。
RAPI_PULSE_OFF	TA_{iIN} 端子からのパルス出力なしを選択する。

・タイマ A 使用時（RAPI_TIMER_A0～RAPI_TIMER_A4 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_F32, RAPI_FC32 }から 1 つを指定します。省略値は RAPI_F2 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（パルス出力機能）　{ RAPI_PULSE_ON, RAPI_PULSE_OFF }から 1 つを指定します。

省略値は RAPI_PULSE_OFF です。

（ゲート機能）　{ RAPI_GATE_L, RAPI_GATE_H }から 1 つを指定します。
省略した場合は、「ゲート機能なし」に設定します。

・タイマ B 使用時（RAPI_TIMER_B0～RAPI_TIMER_B2 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_F32, RAPI_FC32 }から 1 つを指定します。省略値は RAPI_F2 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

(R8C)

RAPI_TIMER_X	タイマ X を使用する。
RAPI_TIMER_Y	タイマ Y を使用する。
RAPI_TIMER_Z	タイマ Z を使用する。
RAPI_TIMER_RA	タイマ RA を使用する。
RAPI_TIMER_RB	タイマ RB を使用する。
RAPI_F1	カウントソースに f_1 を選択する。
RAPI_F2	カウントソースに f_2 を選択する。
RAPI_F8	カウントソースに f_8 を選択する。
RAPI_F32	カウントソースに f_{32} を選択する。
RAPI_FRING	カウントソースに f_{RING} を選択する。
RAPI_FOCO	カウントソースに fOCO を選択する。
RAPI_FC32	カウントソースに fC32 を選択する。
RAPI_TIMER_Y_UNDERFLOW	カウントソースにタイマ Y のアンダフローを選択する。
RAPI_TIMER_RA_UNDERFLOW	カウントソースにタイマ RA のアンダフローを選択する。
RAPI_TIMER_ON	__CreateTimer 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateTimer 内ではタイマを動作停止に設定する。
RAPI_WRITE_RELOAD_ONLY	プライマリレジスタ、プリスケーラに書き込むと、 リロードレジスタのみに書き込まれる動作を選択する。
RAPI_WRITE_RELOAD_BOTH	プライマリレジスタ、プリスケーラに書き込むと、それぞれ リロードレジスタとカウンタの両方に書き込まれる動作を選択する。

・タイマ X 使用時（RAPI_TIMER_X を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_F32 }から 1 つを指定します。
省略値は RAPI_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

・タイマ Y 使用時（RAPI_TIMER_Y を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F8, RAPI_FRING }から 1 つを指定します。
省略値は RAPI_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（書き込み制御）　{RAPI_WRITE_RELOAD_ONLY, RAPI_WRITE_RELOAD_BOTH}から 1 つを指定します。省略値は RAPI_WRITE_RELOAD_BOTH です。

・タイマ Z 使用時（RAPI_TIMER_Z を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_TIMER_Y_UNDERFLOW}から１つを指定します。省略値は RAPI_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から１つを指定します。省略値は RAPI_TIMER_OFF です。

（書き込み制御）　{RAPI_WRITE_RELOAD_ONLY, RAPI_WRITE_RELOAD_BOTH}から１つを指定します。省略値は RAPI_WRITE_RELOAD_BOTH です。

・タイマ RA 使用時（RAPI_TIMER_RA を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_FOCO , RAPI_FC32 }から１つを指定します。省略値は RAPI_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から１つを指定します。省略値は RAPI_TIMER_OFF です。

・タイマ RB 使用時（RAPI_TIMER_RB を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_TIMER_RA_UNDERFLOW}から１つを指定します。省略値は RAPI_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から１つを指定します。省略値は RAPI_TIMER_OFF です。

（書き込み制御）　{RAPI_WRITE_RELOAD_ONLY, RAPI_WRITE_RELOAD_BOTH}から１つを指定します。省略値は RAPI_WRITE_RELOAD_BOTH です。

(H8S)

RAPI_TIMER_RA	タイマ RA を使用する。
RAPI_TIMER_RB	タイマ RB を使用する。
RAPI_TRA_F1	タイマ RA のカウントソースにφのクロックを選択する。
RAPI_TRA_F2	タイマ RA のカウントソースにφ/2 のクロックを選択する。
RAPI_TRA_F8	タイマ RA のカウントソースにφ/8 のクロックを選択する。
RAPI_TRA_F32	タイマ RA のカウントソースにφ/32 のクロックを選択する。
RAPI_TRA_F40	タイマ RA のカウントソースにφ40 のクロックを選択する。
RAPI_TRA_F64	タイマ RA のカウントソースにφ/64 のクロックを選択する。
RAPI_TRA_F128	タイマ RA のカウントソースにφ/128 のクロックを選択する。
RAPI_TRA_FSUB	タイマ RA のカウントソースにサブ発信器出力クロックを選択する。
RAPI_TRB_F1	タイマ RB のカウントソースにφのクロックを選択する。
RAPI_TRB_F2	タイマ RB のカウントソースにφ/2 のクロックを選択する。
RAPI_TRB_F4	タイマ RB のカウントソースにφ/4 のクロックを選択する。
RAPI_TRB_F8	タイマ RB のカウントソースにφ/8 のクロックを選択する。
RAPI_TRB_F32	タイマ RB のカウントソースにφ/32 のクロックを選択する。
RAPI_TRB_F64	タイマ RB のカウントソースにφ/64 のクロックを選択する。
RAPI_TRB_F128	タイマ RB のカウントソースにφ/128 のクロックを選択する。
RAPI_TIMER_RA_ UNDERFLOW	タイマ RB のカウントソースにタイマ RA のアンダフローを選択する。
RAPI_TIMER_ON	__CreateTimer 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateTimer 内ではタイマを動作停止に設定する。
RAPI_UNDERFLOW	アンダフロー割り込みを許可する。
RAPI_INT_MODE_0	割り込み制御モード 0 を指定する。
RAPI_INT_MODE_2	割り込み制御モード 2 を指定する。
RAPI_WRITE_	プライマリレジスタ、プリスケーラに書き込むと、

RELOAD_ONLY	リロードレジスタのみに書き込まれる動作を選択する。
RAPI_WRITE_ RELOAD_BOTH	プライマリレジスタ、プリスケーラに書き込むと、それぞれ リロードレジスタとカウンタの両方に書き込まれる動作を選択する。

・タイマ RA 使用時（RAPI_TIMER_RA を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRA_F1, RAPI_TRA_F2, RAPI_TRA_F8, RAPI_TRA_F32, RAPI_TRA_F40, RAPI_TRA_F64, RAPI_TRA_F128, RAPI_TRA_FSUB }から 1 つを指定します。省略値は RAPI_TRA_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　アンダフロー割り込み要求ありの場合、RAPI_UNDERFLOW を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（割り込み制御モード）　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。省略値は RAPI_INT_MODE_0 です。

・タイマ RB 使用時（RAPI_TIMER_RB を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRB_F1, RAPI_TRB_F2, RAPI_TRB_F4, RAPI_TRB_F8, RAPI_TRB_F32, RAPI_TRB_F64, RAPI_TRB_F128, RAPI_TIMER_RA_UNDERFLOW}から 1 つを指定します。省略値は RAPI_TRB_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（書き込み制御）　{RAPI_WRITE_RELOAD_ONLY, RAPI_WRITE_RELOAD_BOTH}から 1 つを指定します。省略値は RAPI_WRITE_RELOAD_BOTH です。

（割り込み）　アンダフロー割り込み要求ありの場合、RAPI_UNDERFLOW を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（割り込み制御モード）　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。省略値は RAPI_INT_MODE_0 です。

(H8/300H)

RAPI_TIMER_A	タイマ A を使用する。
RAPI_TIMER_B1	タイマ B1 を使用する。
RAPI_TIMER_V	タイマ V を使用する。
RAPI_TA_F8	タイマ A にφ/8 のクロックを入力する。
RAPI_TA_F32	タイマ A にφ/32 のクロックを入力する。
RAPI_TA_F128	タイマ A にφ/128 のクロックを入力する。
RAPI_TA_F256	タイマ A にφ/256 のクロックを入力する。
RAPI_TA_F512	タイマ A にφ/512 のクロックを入力する。
RAPI_TA_F2048	タイマ A にφ/2048 のクロックを入力する。
RAPI_TA_F4096	タイマ A にφ/4096 のクロックを入力する。
RAPI_TA_F8192	タイマ A にφ/8192 のクロックを入力する。
RAPI_TB1_F4	タイマ B1 は内部クロックφ/4 でカウントする。
RAPI_TB1_F16	タイマ B1 は内部クロックφ/16 でカウントする。
RAPI_TB1_F64	タイマ B1 は内部クロックφ/64 でカウントする。
RAPI_TB1_F256	タイマ B1 は内部クロックφ/256 でカウントする。
RAPI_TB1_F512	タイマ B1 は内部クロックφ/512 でカウントする。
RAPI_TB1_F2048	タイマ B1 は内部クロックφ/2048 でカウントする。
RAPI_TB1_F8192	タイマ B1 は内部クロックφ/8192 でカウントする。

RAPI_TV_F4	タイマ V は内部クロックφ/4、立ち下がりエッジでカウントする。
RAPI_TV_F8	タイマ V は内部クロックφ/8、立ち下がりエッジでカウントする。
RAPI_TV_F16	タイマ V は内部クロックφ/16、立ち下がりエッジでカウントする。
RAPI_TV_F32	タイマ V は内部クロックφ/32、立ち下がりエッジでカウントする。
RAPI_TV_F64	タイマ V は内部クロックφ/64、立ち下がりエッジでカウントする。
RAPI_TV_F128	タイマ V は内部クロックφ/128、立ち下がりエッジでカウントする。
RAPI_TIMER_ON	__CreateTimer 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateTimer 内ではタイマを動作停止に設定する。
RAPI_OVERFLOW	オーバフロー割り込みを許可する。
RAPI_OUT_F4	φ/4 のクロックを出力する。
RAPI_OUT_F8	φ/8 のクロックを出力する。
RAPI_OUT_F16	φ/16 のクロックを出力する。
RAPI_OUT_F32	φ/32 のクロックを出力する。
RAPI_OUT_8192	φW/4 のクロックを出力する。
RAPI_OUT_4096	φW/8 のクロックを出力する。
RAPI_OUT_2048	φW/16 のクロックを出力する。
RAPI_OUT_1024	φW/32 のクロックを出力する。
RAPI_AUTO_RELOAD	オートリロード機能を使用する。

・タイマ A 使用時（RAPI_TIMER_A を指定時）に指定可能な定義値

（クロック）　{ RAPI_TA_F8, RAPI_TA_F32, RAPI_TA_F128, RAPI_TA_F256, RAPI_TA_F512, RAPI_TA_F2048, RAPI_TA_F4096, RAPI_TA_F8192 }から 1 つを指定します。省略値は RAPI_TA_F8192 です。

（割り込み）　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（出力）　{ RAPI_OUT_F4, RAPI_OUT_F8, RAPI_OUT_F16, RAPI_OUT_F32, RAPI_OUT_8192, RAPI_OUT_4096, RAPI_OUT_2048, RAPI_OUT_1024 }から 1 つを指定します。省略した場合は、「クロック出力」に設定します。

・タイマ B1 使用時（RAPI_TIMER_B1 を指定時）に指定可能な定義値

（クロック）　{ RAPI_TB1_F4, RAPI_TB1_F16, RAPI_TB1_F64, RAPI_TB1_F256, RAPI_TB1_F512, RAPI_TB1_F2048, RAPI_TB1_F8192}から 1 つを指定します。省略値は RAPI_TB1_F8192 です。

（割り込み）　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（リロード）　オートリロード機能を選択する場合、RAPI_AUTO_RELOAD を指定します。RAPI_AUTO_RELOAD を指定しない場合は、「インターバル機能選択」を設定します。

・タイマ V 使用時（RAPI_TIMER_V を指定時）に指定可能な定義値

（クロック）　{ RAPI_TV_F4, RAPI_TV_F8, RAPI_TV_F16, RAPI_TV_F32, RAPI_TV_F64, RAPI_TV_F128}から 1 つを指定します。クロック指定を省略した場合は、「クロック入力禁止」に指定します。

（割り込み）　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

[data2]

(M16C)(R8C)

割り込み制御レジスタへ設定する割り込みレベル(0～7)を指定します。

(H8S)

インタラプトプライオリティレジスタへ設定する割り込み優先レベル(0～3)を指定します。

(H8/300H)

割り込みコントロールレジスタへ設定する割り込みプライオリティレベル(0～1)を指定します。割り込みコントロールレジスタを持たない CPU の場合は 0 を指定します。

[data3]

(M16C)

タイマレジスタへの設定値を 16 ビットで指定します。

(R8C) (H8S)

タイマレジスタまたは、プライマリレジスタへの設定値を 8 ビットで指定します。

(H8/300H)

タイマロードレジスタへの設定値を 8 ビットで指定します。

タイマ B1 使用時のみ設定可能です。タイマ B1 使用時以外は 0 を指定します。

[data4]

(M16C)(H8/300H)

0 を指定します。

(R8C)(H8S)

プリスケーラレジスタへの設定値を 8 ビットで指定します。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(タイマモード)

参照

__EnableTimer, __DestroyTimer

備考

・ 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できるタイマは CPU により異なります。
・ H8S および、H8/300H で使用する場合、指定したタイマのモジュールスタンバイ解除後に設定をおこないます。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void TimerIntFunc( void ){}

void func( void )
{
    /* Set up timer X as timer mode */
    __CreateTimer( RAPI_TIMER_X|RAPI_TIMER_ON|RAPI_F8, 5, 0x80, 0x80, TimerIntFunc );
}
```

__EnableTimer

要約

< タイマモード動作制御 >

Boolean __EnableTimer(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したタイマモードに設定されているタイマの動作開始/動作停止を制御します。

[data]

data には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を選択する。
RAPI_TIMER_A1	タイマ A チャネル 1 を選択する。
RAPI_TIMER_A2	タイマ A チャネル 2 を選択する。
RAPI_TIMER_A3	タイマ A チャネル 3 を選択する。
RAPI_TIMER_A4	タイマ A チャネル 4 を選択する。
RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。
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する。
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する。

(R8C)

RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_Y	タイマ Y を選択する。
RAPI_TIMER_Z	タイマ Z を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。
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する。
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する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。
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する。
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する。

(H8/300H)

RAPI_TIMER_V	タイマ V を選択する。
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する。
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する。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(タイマモード)

参照　__CreateTimer, __DestroyTimer

備考
- 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できるタイマは CPU により異なります。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void func( void )
{
    /* Disable timer Y as timer mode */
    __EnableTimer( RAPI_TIMER_Y | RAPI_TIMER_OFF );
}
```

__DestroyTimer

要約

< タイマモード設定破棄 >

Boolean __DestroyTimer(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したタイマモードに設定されているタイマの設定を破棄します。

[data]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を選択する。
RAPI_TIMER_A1	タイマ A チャネル 1 を選択する。
RAPI_TIMER_A2	タイマ A チャネル 2 を選択する。
RAPI_TIMER_A3	タイマ A チャネル 3 を選択する。
RAPI_TIMER_A4	タイマ A チャネル 4 を選択する。
RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。

(R8C)

RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_Y	タイマ Y を選択する。
RAPI_TIMER_Z	タイマ Z を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。

(H8/300H)

RAPI_TIMER_A	タイマ A を選択する。
RAPI_TIMER_B1	タイマ B1 を選択する。
RAPI_TIMER_V	タイマ V を選択する。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(タイマモード)

参照

__CreateTimer, __EnableTimer

備考

・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できるタイマは CPU により異なります。
・ H8S および、H8/300H で使用する場合、指定したタイマを、設定破棄後、
モジュールスタンバイ状態に設定します。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void func( void )
{
    /* Destroy the setting of timer Z as timer mode */
    __DestroyTimer( RAPI_TIMER_Z );
}
```

__CreateEventCounter

要約

< イベントカウンタモード設定 >

Boolean __CreateEventCounter(unsigned long data1, unsigned int data2,
unsigned int data3, unsigned int data4, void* func)

data1	設定データ 1（機種により内容が異なる）
data2	設定データ 2（機種により内容が異なる）
data3	設定データ 3（機種により内容が異なる）
data4	設定データ 4（機種により内容が異なる）
func	コールバック関数ポインタ （コールバック関数を設定しない場合は 0 を指定する）

解説

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、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を使用する。
RAPI_TIMER_A1	タイマ A チャネル 1 を使用する。
RAPI_TIMER_A2	タイマ A チャネル 2 を使用する。
RAPI_TIMER_A3	タイマ A チャネル 3 を使用する。
RAPI_TIMER_A4	タイマ A チャネル 4 を使用する。
RAPI_TIMER_B0	タイマ B チャネル 0 を使用する。
RAPI_TIMER_B1	タイマ B チャネル 1 を使用する。
RAPI_TIMER_B2	タイマ B チャネル 2 を使用する。
RAPI_EV_EXTERNAL	カウントソースに TA_{iIN} 端子(タイマ Ai 使用時) または、 TB_{iIN} 端子(タイマ Bi 使用時)に入力された外部信号を選択する。
RAPI_EV_TIMER_AJ	カウントソースにタイマ Aj(j=i-1、ただし i=0 のとき j=4)のオーバフロー またはアンダフローを選択する。
RAPI_EV_TIMER_AK	カウントソースにタイマ Ak(k=i+1、ただし i=4 のとき k=0)のオーバフロー またはアンダフローを選択する。
RAPI_EV_TIMER_B2	カウントソースにタイマ B2 のオーバフローまたはアンダフローを 選択する。
RAPI_EV_TIMER_BJ	カウントソースにタイマ Bj(j=i-1、ただし i=0 のとき j=2)のオーバフロー またはアンダフローを選択する。
RAPI_TIMER_ON	__CreateEventCounter 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateEventCounter 内ではタイマを動作停止に設定する。
RAPI_PULSE_ON	TA_{iIN} 端子からのパルス出力ありを選択する。
RAPI_PULSE_OFF	TA_{iIN} 端子からのパルス出力なしを選択する。
RAPI_AUTO_RELOAD	カウントタイプにリロードタイプを選択する。
RAPI_FREE_RUN	カウントタイプにフリーランタイプを選択する。
RAPI_UP_COUNT	カウント動作にアップカウントを選択する。

RAPI_DOWN_COUNT	カウント動作にダウンカウントを選択する。
RAPI_UDF_REGISTER	アップ/ダウン切り替え要因に UDF レジスタを選択する。
RAPI_TAIOUT	アップ/ダウン切り替え要因に TA_{iOUT} 端子の入力信号を選択する。
RAPI_RISING	立ち上がりエッジをカウントソースの有効エッジとする。
RAPI_FALLING	立ち下がりエッジをカウントソースの有効エッジとする。
RAPI_BOTH	立ち下がりエッジと立ち上がりエッジの両方を カウントソースの有効エッジとする。

・タイマ A 使用時（RAPI_TIMER_A0～RAPI_TIMER_A4 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_EV_EXTERNAL, RAPI_EV_TIMER_AJ, RAPI_EV_TIMER_AK, RAPI_EV_TIMER_B2 }から 1 つをを指定します。
省略値は RAPI_EV_EXTERNAL です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（パルス出力機能）　{ RAPI_PULSE_ON, RAPI_PULSE_OFF }から 1 つを指定します。
省略値は RAPI_PULSE_OFF です。

（カウントタイプ）　{ RAPI_AUTO_RELOAD, RAPI_FREE_RUN }から 1 つを指定します。
省略値は RAPI_AUTO_RELOAD です。

（カウント方向）　{ RAPI_UP_COUNT, RAPI_DOWN_COUNT }から 1 つを指定します。
省略値は RAPI_DOWN_COUNT です。カウント方向の設定は、UDF レジスタ使用時のみ有効となります。

（カウント方向切替）　{ RAPI_UDF_REGISTER, RAPI_TAIOUT }から 1 つを指定します。
省略値は RAPI_UDF_REGISTER です。

（カウントエッジ）　{ RAPI_RISING, RAPI_FALLING }から 1 つを指定します。
省略値は RAPI_FALLING です。

・タイマ B 使用時（RAPI_TIMER_B0～RAPI_TIMER_B2 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_EV_EXTERNAL, RAPI_EV_TIMER_BJ }から 1 つを指定します。
省略値は RAPI_EV_EXTERNAL です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（カウントエッジ）　{ RAPI_RISING, RAPI_FALLING, RAPI_BOTH }から 1 つを指定します。
省略値は RAPI_FALLING です。

(R8C)

RAPI_TIMER_X	タイマ X を使用する。
RAPI_TIMER_Y	タイマ Y を使用する。
RAPI_TIMER_RA	タイマ RA を使用する。
RAPI_TIMER_ON	__CreateEventCounter 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateEventCounter 内ではタイマを動作停止に設定する。
RAPI_WRITE_RELOAD_ONLY	プライマリレジスタ、プリスケーラに書き込むと、リロードレジスタのみに書き込まれる動作を選択する。
RAPI_WRITE_RELOAD_BOTH	プライマリレジスタ、プリスケーラに書き込むと、それぞれリロードレジスタとカウンタの両方に書き込まれる動作を選択する。
RAPI_RISING	立ち上がりエッジをカウントソースの有効エッジとする。
RAPI_FALLING	立ち下がりエッジをカウントソースの有効エッジとする。
RAPI_FILTER_F1	サンプリング周波数 f_1 のデジタルフィルタ機能を使用する。

RAPI_FILTER_F8	サンプリング周波数 f_8 のデジタルフィルタ機能を使用する。
RAPI_FILTER_F32	サンプリング周波数 f_{32} のデジタルフィルタ機能を使用する。
RAPI_TIOSEL_P1_7	カウントソース入力端子を P1_7 に設定する。
RAPI_TIOSEL_P1_5	カウントソース入力端子を P1_5 に設定する。
RAPI_PULSE_ON	TRAO 端子からのパルス出力ありを選択する。
RAPI_PULSE_OFF	TRAO 端子からのパルス出力なしを選択する。

・タイマ X 使用時（RAPI_TIMER_X を指定時）に指定可能な定義値

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（カウントエッジ）　{ RAPI_RISING, RAPI_FALLING }から 1 つを指定します。
省略値は RAPI_FALLING です。

・タイマ Y 使用時（RAPI_TIMER_Y を指定時）に指定可能な定義値

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（書き込み制御）　{RAPI_WRITE_RELOAD_ONLY, RAPI_WRITE_RELOAD_BOTH}から 1 つを
指定します。省略値は RAPI_WRITE_RELOAD_BOTH です。

（_INT2/CNTR1 入力）　{ RAPI_RISING, RAPI_FALLING }から 1 つを指定します。
省略値は RAPI_FALLING です。

・タイマ RA 使用時（RAPI_TIMER_RA を指定時）に指定可能な定義値

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（カウントエッジ）　{ RAPI_RISING, RAPI_FALLING }から 1 つを指定します。
省略値は RAPI_FALLING です。

（フィルタ）　{ RAPI_FILTER_F1, RAPI_FILTER_F8, RAPI_FILTER_F32 }から 1 つを
指定します。省略した場合は、「フィルタなし」に設定します。

（入力端子）　{ RAPI_TIOSEL_P1_7, RAPI_TIOSEL_P1_5 }から 1 つを指定します。
省略値は RAPI_TIOSEL_P1_7 です。

（パルス出力機能）　{ RAPI_PULSE_ON, RAPI_PULSE_OFF }から 1 つを指定します。
省略値は RAPI_PULSE_OFF です。

(H8S)

RAPI_TIMER_RA	タイマ RA を使用する。
RAPI_TIMER_ON	__CreateEventCounter 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateEventCounter 内ではタイマを動作停止に設定する。
RAPI_UNDERFLOW	アンダフロー割り込みを許可する。
RAPI_INT_MODE_0	割り込み制御モード 0 を指定する。
RAPI_INT_MODE_2	割り込み制御モード 2 を指定する。
RAPI_TRA_GATE_H	IRQ2 端子に"H"が入力されている期間カウントするゲート機能を選択する。
RAPI_TRA_RISING	立ち上がりエッジをカウントソースの有効エッジとする。
RAPI_TRA_FALLING	立ち下がりエッジをカウントソースの有効エッジとする。
RAPI_TRA_FILTER_F1	タイマ RA のサンプリング周波φのフィルタ機能を使用する。
RAPI_TRA_FILTER_F8	タイマ RA のサンプリング周波数φ/8 フィルタ機能を使用する。
RAPI_TRA_FILTER_F32	タイマ RA のサンプリング周波数φ/32 のフィルタ機能を使用する。
RAPI_TRA_PULSE_ON	TRAO 端子からのパルス出力ありを選択する。
RAPI_TRA_PULSE_OFF	TRAO 端子からのパルス出力なしを選択する。

・タイマ RA 使用時（RAPI_TIMER_RA を指定時）に指定可能な定義値

（割り込み）　アンダフロー割り込み要求ありの場合、RAPI_UNDERFLOW を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（割り込み制御モード）　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。
省略値は RAPI_INT_MODE_0 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（カウントエッジ）　{ RAPI_TRA_RISING, RAPI_TRA_FALLING }から 1 つを指定します。
省略値は RAPI_TRA_RISING です。

（フィルタ）　{ RAPI_TRA_FILTER_F1, RAPI_TRA_FILTER_F8, RAPI_TRA_FILTER_F32 }から 1 つを指定します。省略した場合は、「フィルタなし」に設定します。

（パルス出力機能）　{ RAPI_TRA_PULSE_ON, RAPI_TRA_PULSE_OFF }から 1 つを指定します。
省略値は RAPI_TRA_PULSE_OFF です。

（ゲート機能）　ゲート機能を使用する場合は、RAPI_TRA_GATE_H を指定します。
省略した場合は、「ゲート機能なし」に設定します。

(H8/300H)

RAPI_TIMER_B1	タイマ B1 を使用する。
RAPI_TIMER_V	タイマ V を使用する。
RAPI_TIMER_ON	__CreateEventCounter 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateEventCounter 内ではタイマを動作停止に設定する。
RAPI_OVERFLOW	オーバフロー割り込みを許可する。
RAPI_AUTO_RELOAD	オートリロード機能を使用する。
RAPI_RISING	立ち上がりエッジをカウントソースの有効エッジとする。
RAPI_FALLING	立ち下がりエッジをカウントソースの有効エッジとする。
RAPI_BOTH	立ち下がりエッジと立ち上がりエッジの両方をカウントソースの有効エッジとする。
RAPI_TRGV_RISING	TRGV 端子入力信号の立ち上がりエッジを有効エッジとする。
RAPI_TRGV_FALLING	TRGV 端子入力信号の立ち下がりエッジを有効エッジとする。
RAPI_TRGV_BOTH	TRGV 端子入力信号の立ち上がりと立下りの両エッジを有効エッジとする。
RAPI_TRGV_PROHIBITED	TRGV 端子からのトリガ入力禁止を選択する。

・タイマ B1 使用時（RAPI_TIMER_B1 を指定時）に指定可能な定義値

（割り込み）　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。
RAPI_OVERFLOW を指定しない場合は「オーバーフロー割り込み要求なし」に設定します。

（リロード）　オートリロード機能を選択する場合、RAPI_AUTO_RELOAD を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（カウントエッジ）　{ RAPI_RISING, RAPI_FALLING }から 1 つを指定します。
省略値は RAPI_FALLING です。

・タイマ V 使用時（RAPI_TIMER_V を指定時）に指定可能な定義値

（割り込み）　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（カウントエッジ）　{ RAPI_RISING, RAPI_FALLING, RAPI_BOTH }から 1 つを指定します。
省略値は RAPI_FALLING です。
（TRGV 端子入力）　{ RAPI_TRGV_RISING, RAPI_TRGV_FALLING, RAPI_TRGV_BOTH,
RAPI_TRGV_PROHIBITED }から 1 つを指定します。
省略値は RAPI_TRGV_PROHIBITED です。

[data2]
(M16C)(R8C)
割り込み制御レジスタへ設定する割り込みレベル(0～7)を指定します。
(H8S)
インタラプトプライオリティレジスタへ設定する割り込み優先レベル(0～3)を指定します。
(H8/300H)
割り込みコントロールレジスタへ設定する割り込みプライオリティレベル(0～1)を指定します。割り込みコントロールレジスタを持たない CPU の場合は 0 を指定します。

[data3]
(M16C)
タイマレジスタへの設定値を 16 ビットで指定します。
(R8C)(H8S)
タイマレジスタまたは、プライマリレジスタへの設定値を 8 ビットで指定します。
(H8/300H)
タイマロードレジスタへの設定値を 8 ビットで指定します。
タイマ B1 使用時のみ設定可能です。タイマ B1 使用時以外は 0 を指定します。

[data4]
(M16C)(H8/300H)
0 を指定します。
(R8C)(H8S)
プリスケーラレジスタへの設定値を 8 ビットで指定します。

戻り値　タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能　タイマ(イベントカウンタモード)

参照　__EnableEventCounter, __DestroyEventCounter, __GetEventCounter

備考
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できるタイマは CPU により異なります。
- H8S および、H8/300H で使用する場合、指定したタイマのモジュールスタンバイ解除に設定をおこないます。

プログラム例

```
#include "rapi_timer_r8c_13.h"
```

```
void TimerIntFunc( void ){}

void func( void )
{
    /* Set up timer Y as event counter mode */
    __CreateEventCounter( RAPI_TIMER_Y|RAPI_TIMER_ON|RAPI_FALLING, 5,
                          0x80, 0x80, TimerIntFunc );
}
```

__EnableEventCounter

要約

< イベントカウンタモード動作制御 >

Boolean __EnableEventCounter(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したタイマモードに設定されているタイマの動作開始/動作停止を制御します。

[data]

data には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を選択する。
RAPI_TIMER_A1	タイマ A チャネル 1 を選択する。
RAPI_TIMER_A2	タイマ A チャネル 2 を選択する。
RAPI_TIMER_A3	タイマ A チャネル 3 を選択する。
RAPI_TIMER_A4	タイマ A チャネル 4 を選択する。
RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。
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を動作開始に設定する。
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を動作停止に設定する。

(R8C)

RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_Y	タイマ Y を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。
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を動作開始に設定する。
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を動作停止に設定する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。
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を動作開始に設定する。
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を動作停止に設定する。

(H8/300H)

RAPI_TIMER_V	タイマ V を選択する。
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を動作開始に設定する。
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を動作停止に設定する。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(イベントカウンタモード)

参照

__CreateEventCounter, __DestroyEventCounter, __GetEventCounter

備考

・引数に定義値以外の値を指定した場合の動作は保証されません。
・指定できるタイマは CPU により異なります。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void func( void )
{
    /* Disable timer Y as event counter mode */
    __EnableEventCounter( RAPI_TIMER_Y|RAPI_TIMER_OFF );
}
```

__DestroyEventCounter

要約

〈 イベントカウンタモード設定破棄 〉

Boolean __DestroyEventCounter(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したタイマモードに設定されているタイマの設定を破棄します。

[data]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を選択する。
RAPI_TIMER_A1	タイマ A チャネル 1 を選択する。
RAPI_TIMER_A2	タイマ A チャネル 2 を選択する。
RAPI_TIMER_A3	タイマ A チャネル 3 を選択する。
RAPI_TIMER_A4	タイマ A チャネル 4 を選択する。
RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。

(R8C)

RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_Y	タイマ Y を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。

(H8/300H)

RAPI_TIMER_B1	タイマ B1 を選択する。
RAPI_TIMER_V	タイマ V を選択する。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(イベントカウンタモード)

参照

__CreateEventCounter, __EnableEventCounter, __GetEventCounter

備考

・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できるタイマは CPU により異なります。
・ H8S および、H8/300H で使用する場合、指定したタイマを、設定破棄後、
モジュールスタンバイ状態に設定します。

プログラム例

```
#include "rapi_timer_r8c_13.h"
```

```
void func( void )
{
    /* Destroy the setting of timer Y as event counter mode */
    __DestroyEventCounter( RAPI_TIMER_Y );
}
```

__GetEventCounter

要約

〈 イベントカウンタモードカウンタ値取得 〉

Boolean __GetEventCounter(unsigned long data1, unsigned int *data2)

data1	設定データ 1（機種により内容が異なる）
data2	カウンタ値を格納するバッファポインタ

解説

指定したイベントカウンタモードに設定されているタイマのカウンタ値を取得します。

[data1]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を選択する。
RAPI_TIMER_A1	タイマ A チャネル 1 を選択する。
RAPI_TIMER_A2	タイマ A チャネル 2 を選択する。
RAPI_TIMER_A3	タイマ A チャネル 3 を選択する。
RAPI_TIMER_A4	タイマ A チャネル 4 を選択する。
RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。

(R8C)

RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_Y	タイマ Y を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。

(H8/300H)

RAPI_TIMER_B1	タイマ B1 を選択する。
RAPI_TIMER_V	タイマ V を選択する。

[data2]

取得したカウンタ値を格納する配列のポインタを指定します。

(M16C)

・タイマ A 使用時 (RAPI_TIMER_A0～RAPI_TIMER_A4 を指定時)

[0]：タイマ Ai レジスタ(i=0～4)の値を格納します。

・タイマ B 使用時 (RAPI_TIMER_B0～RAPI_TIMER_B2 を指定時)

[0]：タイマ Bi レジスタ(i=0～2)の値を格納します。

(R8C)

・タイマ X 使用時 (RAPI_TIMER_X を指定時)

[0]：プリスケーラ X レジスタの値を格納します。

[1]：タイマ X レジスタの値を格納します。

・タイマ Y 使用時 (RAPI_TIMER_Y を指定時)

[0]：プリスケーラ Y の値を格納します。

[1]：タイマ Y プライマリレジスタの値を格納します。

・タイマ RA 使用時 (RAPI_TIMER_RA を指定時)

[0]：タイマ RA プリスケーラレジスタの値を格納します。

[1]：タイマ RA レジスタの値を格納します。

(H8S)

・タイマ RA 使用時 (RAPI_TIMER_RA を指定時)

[0]：タイマ RA プリスケーラレジスタの値を格納します。

[1]：タイマ RA レジスタの値を格納します。

(H8/300H)

・タイマ B1 使用時 (RAPI_TIMER_B1 を指定時)

[0]：タイマカウンタ B1 の値を格納します。

・タイマ V 使用時 (RAPI_TIMER_V を指定時)

[0]：タイマカウンタ V の値を格納します。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(イベントカウンタモード)

参照

__CreateEventCounter, __EnableEventCounter, __DestroyEventCounter

備考

・第 1 引数に定義値以外の値を指定した場合の動作は保証されません。

・指定できるタイマは CPU により異なります。

プログラム例

```
#include "rapi_timer_r8c_13.h"

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

    /* Get the counter of timer Y as event counter mode */
    __GetEventCounter(RAPI_TIMER_Y, data );
}
```

__CreatePulseWidthModulationMode

要約

< パルス幅変調モード設定 >

Boolean __CreatePulseWidthModulationMode(unsigned long data1, unsigned int data2, unsigned int* data3, void* data4)

data1	設定データ 1（機種により内容が異なる）
data2	設定データ 2（機種により内容が異なる）
data3	設定データ 3（機種により内容が異なる）
func	コールバック関数ポインタ （コールバック関数を設定しない場合は 0 を指定する）

解説

指定したタイマをパルス幅変調モードに設定します。

[data1]

data1 には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を使用する。
RAPI_TIMER_A1	タイマ A チャネル 1 を使用する。
RAPI_TIMER_A2	タイマ A チャネル 2 を使用する。
RAPI_TIMER_A3	タイマ A チャネル 3 を使用する。
RAPI_TIMER_A4	タイマ A チャネル 4 を使用する。
RAPI_F1	カウントソースに f_1 を選択する。
RAPI_F2	カウントソースに f_2 を選択する。
RAPI_F8	カウントソースに f_8 を選択する。
RAPI_F32	カウントソースに f_{32} を選択する。
RAPI_FC32	カウントソースに f_{C32} を選択する。
RAPI_TIMER_ON	__CreatePulseWidthModulationMode 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreatePulseWidthModulationMode 内ではタイマを動作停止に設定する。
RAPI_TG_TAIN	カウント開始条件に、TA_{iIN} 端子からの外部トリガ入力を選択する。
RAPI_EV_TIMER_AJ	カウント開始条件に、タイマ Aj(j=i-1、ただし i=0 のとき j=4)の オーバフローまたはアンダフローをトリガに選択する。
RAPI_EV_TIMER_AK	カウント開始条件に、タイマ Ak(k=i+1、ただし i=4 のとき k=0)の オーバフローまたはアンダフローをトリガに選択する。
RAPI_EV_TIMER_B2	カウント開始条件に、タイマ B2 のオーバフローまたは アンダフローをトリガに選択する。
RAPI_TG_TAIS	カウント開始条件は、TABSR レジスタの TA_iS ビットへの"1"書き込みのみ。
RAPI_PULSE_ON	TA_{iIN} 端子からのパルス出力ありを選択する。タイマ Ai 使用時のみ選択可能。
RAPI_PULSE_OFF	TA_{iIN} 端子からのパルス出力なしを選択する。タイマ Ai 使用時のみ選択可能。
RAPI_PWM_16	16 ビットパルス幅変調器としての動作を選択。
RAPI_PWM_8	8 ビットパルス幅変調器としての動作を選択。
RAPI_RISING	立ち上がりエッジを TA_{iIN} 端子入力信号の有効エッジとする。
RAPI_FALLING	立ち下がりエッジを TA_{iIN} 端子入力信号の有効エッジとする。

・タイマ A 使用時（RAPI_TIMER_A0～RAPI_TIMER_A4 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_F32, RAPI_FC32 }から 1 つを指定します。省略値は RAPI_F2 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（カウント開始条件）　{ RAPI_TG_TAIS, RAPI_TG_TAIIN, RAPI_EV_TIMER_AJ, RAPI_EV_TIMER_AK, RAPI_EV_TIMER_B2 }から 1 つを指定します。省略値は RAPI_TG_TAIS です。

（パルス出力機能）　{ RAPI_PULSE_ON, RAPI_PULSE_OFF }から 1 つを指定します。省略値は RAPI_PULSE_OFF です。

（変調器）　{ RAPI_PWM_16, RAPI_PWM_8 }から 1 つを指定します。省略値は RAPI_PWM_16 です。

（TA_{iIN}端子入力）　{ RAPI_RISING, RAPI_FALLING }から 1 つを指定します。省略値は RAPI_FALLING です。TA_{iIN}端子入力設定は、RAPI_TG_TAIIN 使用時のみ有効となります。

(R8C)

RAPI_TIMER_Y	タイマ Y を使用する。
RAPI_TIMER_Z	タイマ Z を使用する。
RAPI_TIMER_RB	タイマ RB を使用する。
RAPI_F1	カウントソースに f_1 を選択する。
RAPI_F2	カウントソースに f_2 を選択する。
RAPI_F8	カウントソースに f_8 を選択する。
RAPI_FRING	カウントソースに f_{RING} を選択する。
RAPI_TIMER_Y_UNDERFLOW	カウントソースにタイマ Y のアンダフローを選択する。
RAPI_TIMER_RA_UNDERFLOW	カウントソースにタイマ RA のアンダフローを選択する。
RAPI_TIMER_ON	__CreatePulseWidthModulationMode 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreatePulseWidthModulationMode 内ではタイマを動作停止に設定する。
RAPI_H_L_L	プライマリ期間"H"出力、セカンダリ期間"L"出力、タイマ停止時"L"出力に設定する。
RAPI_L_H_H	プライマリ期間"L"出力、セカンダリ期間"H"出力、タイマ停止時"H"出力に設定する。
RAPI_TRBO_P3_1	出力端子に P3_1 を選択する。
RAPI_TRBO_P1_3	出力端子に P1_3 を選択する。

・タイマ Y 使用時（RAPI_TIMER_Y を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F8, RAPI_FRING }から 1 つを指定します。省略値は RAPI_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（出力）　{ RAPI_H_L_L, RAPI_L_H_H }から 1 つを指定します。省略値は RAPI_H_L_L です。

・タイマ Z 使用時（RAPI_TIMER_Z を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_TIMER_Y_UNDERFLOW}から 1 つを指定します。省略値は RAPI_F1 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。
(出力)　{ RAPI_H_L_L, RAPI_L_H_H }から 1 つを指定します。
省略値は RAPI_H_L_L です。

・タイマ RB 使用時 (RAPI_TIMER_RB を指定時) に指定可能な定義値

(カウントソース)　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_TIMER_RA_UNDERFLOW}から 1 つを指定します。省略値は RAPI_F1 です。
(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。
(出力)　{ RAPI_H_L_L, RAPI_L_H_H }から 1 つを指定します。
省略値は RAPI_H_L_L です。
(出力端子)　{ RAPI_TRBO_P3_1,RAPI_TRBO_P1_3 }から 1 つを指定します。
省略値は RAPI_TRBO_P3_1 です。
R8C/26,27 のみ指定可能です。

(H8S)

RAPI_TIMER_RB	タイマ RB を使用する。
RAPI_TRB_F1	タイマ RB のカウントソースにφのクロックを選択する。
RAPI_TRB_F2	タイマ RB のカウントソースにφ/2 のクロックを選択する。
RAPI_TRB_F4	タイマ RB のカウントソースにφ/4 のクロックを選択する。
RAPI_TRB_F8	タイマ RB のカウントソースにφ/8 のクロックを選択する。
RAPI_TRB_F32	タイマ RB のカウントソースにφ/32 のクロックを選択する。
RAPI_TRB_F64	タイマ RB のカウントソースにφ/64 のクロックを選択する。
RAPI_TRB_F128	タイマ RB のカウントソースにφ/128 のクロックを選択する。
RAPI_TIMER_RA_ UNDERFLOW	タイマ RB のカウントソースにタイマ RA のアンダフローを選択する。
RAPI_TIMER_ON	__CreatePulseWidthModulationMode 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreatePulseWidthModulationMode 内ではタイマを動作停止に設定する。
RAPI_UNDERFLOW	アンダフロー割り込みを許可する。
RAPI_INT_MODE_0	割り込み制御モード 0 を指定する。
RAPI_INT_MODE_2	割り込み制御モード 2 を指定する。
RAPI_H_L_L	プライマリ期間"H"出力、セカンダリ期間"L"出力、 タイマ停止時"L"出力に設定する。
RAPI_L_H_H	プライマリ期間"L"出力、セカンダリ期間"H"出力、 タイマ停止時"H"出力に設定する。

・タイマ RB 使用時 (RAPI_TIMER_RB を指定時) に指定可能な定義値

(カウントソース)　{ RAPI_TRB_F1, RAPI_TRB_F2, RAPI_TRB_F4, RAPI_TRB_F8, RAPI_TRB_F32, RAPI_TRB_F64, RAPI_TRB_F128, RAPI_TIMER_RA_UNDERFLOW}から 1 つを指定します。省略値は RAPI_TRB_F1 です。
(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。
(割り込み)　アンダフロー割り込み要求ありの場合、RAPI_UNDERFLOW を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。
(割り込み制御モード)　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。
省略値は RAPI_INT_MODE_0 です。

（出力）　　{ RAPI_H_L_L, RAPI_L_H_H }から 1 つを指定します。
省略値は RAPI_H_L_L です。

(H8/300H)

RAPI_TIMER_V	タイマ V を使用する。
RAPI_TV_F4	タイマ V は内部クロックφ/4、立ち下がりエッジでカウントする。
RAPI_TV_F8	タイマ V は内部クロックφ/8、立ち下がりエッジでカウントする。
RAPI_TV_F16	タイマ V は内部クロックφ/16、立ち下がりエッジでカウントする。
RAPI_TV_F32	タイマ V は内部クロックφ/32、立ち下がりエッジでカウントする。
RAPI_TV_F64	タイマ V は内部クロックφ/64、立ち下がりエッジでカウントする。
RAPI_TV_F128	タイマ V は内部クロックφ/128、立ち下がりエッジでカウントする。
RAPI_TIMER_ON	__CreatePulseWidthModulationMode 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreatePulseWidthModulationMode 内ではタイマを動作停止に設定する。
RAPI_COMPARE_MATCH_A	コンペアマッチ A 割り込みを許可する。
RAPI_COMPARE_MATCH_B	コンペアマッチ B 割り込みを許可する。
RAPI_OUT_H_L	プライマリ期間"H"出力、セカンダリ期間"L"出力に設定する。
RAPI_OUT_L_H	プライマリ期間"L"出力、セカンダリ期間"H"出力に設定する。
RAPI_TRGV_RISING	TRGV 端子入力信号の立ち上がりエッジを有効エッジとする。
RAPI_TRGV_FALLING	TRGV 端子入力信号の立ち下がりエッジを有効エッジとする。
RAPI_TRGV_BOTH	TRGV 端子入力信号の立ち上がりと立下りの両エッジを有効エッジとする。
RAPI_TRGV_PROHIBITED	TRGV 端子からのトリガ入力禁止を選択する。

・タイマ V 使用時（RAPI_TIMER_V を指定時）に指定可能な定義値

（クロック）　　{ RAPI_TV_F4, RAPI_TV_F8, RAPI_TV_F16, RAPI_TV_F32, RAPI_TV_F64, RAPI_TV_F128}から 1 つを指定します。クロック指定を省略した場合は、「クロック入力禁止」に指定します。

（割り込み）　　コンペアマッチ A 割り込み要求ありの場合 RAPI_COMPARE_MATCH_A を、またコンペアマッチ B 割り込み要求ありの場合 RAPI_COMPARE_MATCH_B を指定します。割り込みを指定しない場合は「割り込み要求なし」に設定します。

（設定時動作状態）　　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（出力）　　{ RAPI_OUT_H_L, RAPI_OUT_L_H }から 1 つを指定します。
出力は必ず指定してください。

（TRGV 端子入力）　　{ RAPI_TRGV_RISING, RAPI_TRGV_FALLING, RAPI_TRGV_BOTH, RAPI_TRGV_PROHIBITED }から 1 つを指定します。
省略値は RAPI_TRGV_PROHIBITED です。

[data2]

(M16C)(R8C)

割り込み制御レジスタへ設定する割り込みレベル(0～7)を指定します。

(H8S)

インタラプトプライオリティレジスタへ設定する割り込み優先レベル(0～3)を

指定します。

(H8/300H)

割り込みコントロールレジスタへ設定する割り込みプライオリティレベル(0～1)を
指定します。割り込みコントロールレジスタを持たない CPU の場合は 0 を指定します。

[data3]

(M16C)

タイマレジスタへの設定値を格納する 16 ビット変数のポインタを指定します。
16 ビット PWM の場合は、"H"幅 n/f_j、周期 $65535/f_j$ の n の値を 16 ビットで指定します。
また、8 ビット PWM の場合は、"H"幅 $n(m+1)/f_j$、周期 $255(m+1)/f_j$ の n の値を
上位 8 ビットに、m の値を下位 8 ビットに指定します。

(R8C)

設定値を格納する配列へのポインタを指定します。
[0]：プライマリレジスタへの設定値を 8 ビットで指定します。
[1]：セカンダリレジスタへの設定値を 8 ビットで指定します。
[2]：プリスケーラへの設定を 8 ビットで指定します。

(H8S)

設定値を格納する配列へのポインタを指定します。
[0]：プライマリレジスタへの設定値を 8 ビットで指定します。
[1]：セカンダリレジスタへの設定値を 8 ビットで指定します。
[2]：プリスケーラへの設定を 8 ビットで指定します。

(H8/300H)

設定値を格納する配列へのポインタを指定します。
[0]：比較値 A を 8 ビットで指定します。
[1]：比較値 B を 8 ビットで指定します。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(パルス幅変調モード(PWM モード))

参照

__EnablePulseWidthModulationMode, __DestroyPulseWidthModulationMode

備考

・ 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できるタイマは CPU により異なります。
・ H8/300H では、比較値 A<比較値 B となるように設定してください。
・ H8S および、H8/300H で使用する場合、指定したタイマのモジュールスタンバイ解除後に
設定をおこないます。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void TimerIntFunc( void ){}
```

```
void func( void )
{
    unsigned int p_tim[] =  {0xAA, 0xBB, 0xCC};

    /* Set up timer Z as pulse width modulation mode */
    __CreatePulseWidthModulationMode( RAPI_TIMER_Z|RAPI_TIMER_ON|RAPI_F8,
                                      5, p_tim, TimerIntFunc);
}
```

__EnablePulseWidthModulationMode

要約

< パルス幅変調モード動作制御 >

Boolean __EnablePulseWidthModulationMode(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したパルス幅変調モードに設定されているタイマの動作開始/動作停止を制御します。

[data]

data には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を選択する。
RAPI_TIMER_A1	タイマ A チャネル 1 を選択する。
RAPI_TIMER_A2	タイマ A チャネル 2 を選択する。
RAPI_TIMER_A3	タイマ A チャネル 3 を選択する。
RAPI_TIMER_A4	タイマ A チャネル 4 を選択する。
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する。
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する。

(R8C)

RAPI_TIMER_Y	タイマ Y を選択する。
RAPI_TIMER_Z	タイマ Z を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。
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する。
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する。

(H8S)

RAPI_TIMER_RB	タイマ RB を選択する。
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する。
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する。

(H8/300H)

RAPI_TIMER_V	タイマ V を選択する。
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する。
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する。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(パルス幅変調モード(PWM モード))

参照

__CreatePulseWidthModulationMode, __DestroyPulseWidthModulationMode

備考

・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できるタイマは CPU により異なります。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void func( void )
{
    /* Enable timer Y as pulse width modulation mode */
    __EnablePulseWidthModulationMode( RAPI_TIMER_Y|RAPI_TIMER_ON );
}
```

__DestroyPulseWidthModulationMode

要約

< パルス幅変調モード設定破棄 >

Boolean __DestroyPulseWidthModulationMode(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したパルス幅変調モードに設定されているタイマの設定を破棄します。

[data]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を選択する。
RAPI_TIMER_A1	タイマ A チャネル 1 を選択する。
RAPI_TIMER_A2	タイマ A チャネル 2 を選択する。
RAPI_TIMER_A3	タイマ A チャネル 3 を選択する。
RAPI_TIMER_A4	タイマ A チャネル 4 を選択する。

(R8C)

RAPI_TIMER_Y	タイマ Y を選択する。
RAPI_TIMER_Z	タイマ Z を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。

(H8S)

RAPI_TIMER_RB	タイマ RB を選択する。

(H8/300H)

RAPI_TIMER_V	タイマ V を選択する。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(パルス幅変調モード(PWM モード))

参照

__CreatePulseWidthModulationMode, __EnablePulseWidthModulationMode

備考

・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できるタイマは CPU により異なります。
・ H8S および、H8/300H で使用する場合、指定したタイマを、設定破棄後、
  モジュールスタンバイ状態に設定します。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void func( void )
{
    /* Destroy the setting of timer Z as pulse width modulation mode */
    __DestroyPulseWidthModulationMode( RAPI_TIMER_Z );
}
```

__CreatePulsePeriodMeasurementMode

要約

< パルス周期測定モード設定 >

Boolean __CreatePulsePeriodMeasurementMode(unsigned long data1, unsigned int data2, unsigned int data3, unsigned int data4, void* func)

data1	設定データ 1（機種により内容が異なる）
data2	設定データ 2（機種により内容が異なる）
data3	設定データ 3（機種により内容が異なる）
data4	設定データ 4（機種により内容が異なる）
func	コールバック関数ポインタ （コールバック関数を設定しない場合は 0 を指定する）

解説

指定したタイマをパルス周期測定モードに設定します。

[data1]

data1 には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_B0	タイマ B チャネル 0 を使用する。
RAPI_TIMER_B1	タイマ B チャネル 1 を使用する。
RAPI_TIMER_B2	タイマ B チャネル 2 を使用する。
RAPI_F1	カウントソースに f_1 を選択する。
RAPI_F2	カウントソースに f_2 を選択する。
RAPI_F8	カウントソースに f_8 を選択する。
RAPI_F32	カウントソースに f_{32} を選択する。
RAPI_FC32	カウントソースに f_{C32} を選択する。
RAPI_TIMER_ON	__CreatePulsePeriodMeasurementMode 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreatePulsePeriodMeasurementMode 内ではタイマを動作停止に設定する。
RAPI_RISING_RISING	測定パルスの立ち上がり－立ち上がり間の測定を選択する。
RAPI_FALLING_ FALLING	測定パルスの立ち下がり－立ち下がり間の測定を選択する。

・タイマ B 使用時（RAPI_TIMER_B0～RAPI_TIMER_B2 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_F32, RAPI_FC32 }から 1 つを指定します。省略値は RAPI_F2 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（測定パルス）　{ RAPI_RISING_RISING, RAPI_FALLING_FALLING }から 1 つを指定します。省略値は RAPI_FALLING_FALLING です。

(R8C)

RAPI_TIMER_X	タイマ X を使用する。
RAPI_TIMER_RA	タイマ RA を使用する。
RAPI_F1	カウントソースに f_1 を選択する。
RAPI_F2	カウントソースに f_2 を選択する。

RAPI_F8	カウントソースに f_8 を選択する。
RAPI_F32	カウントソースに f_{32} を選択する。
RAPI_FC32	カウントソースに f_{C32} を選択する。
RAPI_FOCO	カウントソースに fOCO 選択する。
RAPI_TIMER_ON	__CreatePulsePeriodMeasurementMode 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreatePulsePeriodMeasurementMode 内ではタイマを動作停止に設定する。
RAPI_RISING_RISING	測定パルスの立ち上がり－立ち上がり間の測定を選択する。
RAPI_FALLING_FALLING	測定パルスの立ち下がり－立ち下がり間の測定を選択する。
RAPI_FILTER_F1	サンプリング周波数 f_1 のデジタルフィルタ機能を使用する。
RAPI_FILTER_F8	サンプリング周波数 f_8 のデジタルフィルタ機能を使用する。
RAPI_FILTER_F32	サンプリング周波数 f_{32} のデジタルフィルタ機能を使用する。
RAPI_TIOSEL_P1_7	カウントソース入力端子を P1_7 に設定する。
RAPI_TIOSEL_P1_5	カウントソース入力端子を P1_5 に設定する。

・タイマ X 使用時（RAPI_TIMER_X を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_F32 }から 1 つを指定します。
省略値は RAPI_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（測定パルス）　{ RAPI_RISING_RISING, RAPI_FALLING_FALLING }から 1 つを指定します。
省略値は RAPI_FALLING_FALLING です。

・タイマ RA 使用時（RAPI_TIMER_RA を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_FC32, RAPI_FOCO }から 1 つを指定します。省略値は RAPI_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（測定パルス）　{ RAPI_RISING_RISING, RAPI_FALLING_FALLING }から 1 つを指定します。
省略値は RAPI_FALLING_FALLING です。

（フィルタ）　{ RAPI_FILTER_F1, RAPI_FILTER_F8, RAPI_FILTER_F32 }から 1 つを指定します。省略した場合は、「フィルタなし」に設定します。

（入力端子）　{ RAPI_TIOSEL_P1_7, RAPI_TIOSEL_P1_5 }から 1 つを指定します。
省略値は RAPI_TIOSEL_P1_7 です。

(H8S)

RAPI_TIMER_RA	タイマ RA を使用する。
RAPI_TRA_F1	タイマ RA のカウントソースにφのクロックを選択する。
RAPI_TRA_F2	タイマ RA のカウントソースにφ/2 のクロックを選択する。
RAPI_TRA_F8	タイマ RA のカウントソースにφ/8 のクロックを選択する。
RAPI_TRA_F32	タイマ RA のカウントソースにφ/32 のクロックを選択する。
RAPI_TRA_F40	タイマ RA のカウントソースにφ40 のクロックを選択する。
RAPI_TRA_F64	タイマ RA のカウントソースにφ/64 のクロックを選択する。
RAPI_TRA_F128	タイマ RA のカウントソースにφ/128 のクロックを選択する。
RAPI_TRA_FSUB	タイマ RA のカウントソースにサブ発信器出力クロックを選択する。
RAPI_TIMER_ON	__CreatePulsePeriodMeasurementMode 内でタイマを動作開始に設定する。

RAPI_TIMER_OFF	__CreatePulsePeriodMeasurementMode 内ではタイマを動作停止に設定する。
RAPI_UNDERFLOW	アンダフロー割り込みを許可する。
RAPI_INT_MODE_0	割り込み制御モード 0 を指定する。
RAPI_INT_MODE_2	割り込み制御モード 2 を指定する。
RAPI_TRA_GATE_H	IRQ2 端子に"H"が入力されている期間カウントするゲート機能を選択する。
RAPI_RISING_RISING	測定パルスの立ち上がり－立ち上がり間の測定を選択する。
RAPI_FALLING_ FALLING	測定パルスの立ち下がり－立ち下がり間の測定を選択する。
RAPI_TRA_FILTER_F1	タイマ RA のサンプリング周波φのフィルタ機能を使用する。
RAPI_TRA_FILTER_F8	タイマ RA のサンプリング周波数φ/8 フィルタ機能を使用する。
RAPI_TRA_FILTER_F32	タイマ RA のサンプリング周波数φ/32 のフィルタ機能を使用する。

・タイマ RA 使用時（RAPI_TIMER_RA を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRA_F1, RAPI_TRA_F2, RAPI_TRA_F8, RAPI_TRA_F32, RAPI_TRA_F40, RAPI_TRA_F64, RAPI_TRA_F128, RAPI_TRA_FSUB }から 1 つを指定します。
省略値は RAPI_TRA_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（割り込み）　アンダフロー割り込み要求ありの場合、RAPI_UNDERFLOW を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（割り込み制御モード）　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。
省略値は RAPI_INT_MODE_0 です。

（測定パルス）　{ RAPI_RISING_RISING, RAPI_FALLING_FALLING }から 1 つを指定します。
省略値は RAPI_RISING_RISING です。

（フィルタ）　{ RAPI_TRA_FILTER_F1, RAPI_TRA_FILTER_F8, RAPI_TRA_FILTER_F32 }から 1 つを指定します。省略した場合は、「フィルタなし」に設定します。

（ゲート機能）　ゲート機能を使用する場合は、RAPI_TRA_GATE_H を指定します。
省略した場合は、「ゲート機能なし」に設定します。

(H8/300H)

RAPI_TIMER_W	タイマ W を使用する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を使用する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を使用する。
RAPI_TIMER_RC	タイマ RC を使用する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を使用する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を使用する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を使用する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を使用する。
RAPI_TW_F1	タイマ W は内部クロックφでカウントする。
RAPI_TW_F2	タイマ W は内部クロックφ/2 でカウントする。
RAPI_TW_F4	タイマ W は内部クロックφ/4 でカウントする。
RAPI_TW_F8	タイマ W は内部クロックφ/8 でカウントする。
RAPI_TZ_F1	タイマ Z は内部クロックφでカウントする。
RAPI_TZ_F2	タイマ Z は内部クロックφ/2 でカウントする。
RAPI_TZ_F4	タイマ Z は内部クロックφ/4 でカウントする。
RAPI_TZ_F8	タイマ Z は内部クロックφ/8 でカウントする。

RAPI_TRC_F1	タイマ RC は内部クロックφでカウントする。
RAPI_TRC_F2	タイマ RC は内部クロックφ/2 でカウントする。
RAPI_TRC_F4	タイマ RC は内部クロックφ/4 でカウントする。
RAPI_TRC_F8	タイマ RC は内部クロックφ/8 でカウントする。
RAPI_TRC_F32	タイマ RC は内部クロックφ/32 でカウントする。
RAPI_TRC_F40M	タイマ RC は内部クロックφ40M でカウントする。
RAPI_TRD_F1	タイマ RD は内部クロックφでカウントする。
RAPI_TRD_F2	タイマ RD は内部クロックφ/2 でカウントする。
RAPI_TRD_F4	タイマ RD は内部クロックφ/4 でカウントする。
RAPI_TRD_F8	タイマ RD は内部クロックφ/8 でカウントする。
RAPI_TRD_F32	タイマ RD は内部クロックφ/32 でカウントする。
RAPI_TRD_F40M	タイマ RD は内部クロックφ40M でカウントする。
RAPI_TIMER_ON	__CreatePulsePeriodMeasurementMode 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreatePulsePeriodMeasurementMode 内ではタイマを動作停止に設定する。
RAPI_OVERFLOW	オーバフロー割り込みを許可する。
RAPI_RISING_RISING	測定パルスの立ち上がりー立ち上がり間の測定を選択する。
RAPI_FALLING_FALLING	測定パルスの立ち下がりー立ち下がり間の測定を選択する。
RAPI_TRC_FILTER_F1	タイマ RC でサンプリング周波数 f_1 のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F8	タイマ RC でサンプリング周波数 f_8 のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F32	タイマ RC でサンプリング周波数 f_{32} のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F	タイマ RC でカウントソースと同じサンプリング周波数のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F1	タイマ RD でサンプリング周波数 f_1 のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F8	タイマ RD でサンプリング周波数 f_8 のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F32	タイマ RD でサンプリング周波数 f_{32} のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F	タイマ RD でカウントソースと同じサンプリング周波数のデジタルフィルタ機能を使用する。
RAPI_FTIOA	FTIOA 端子を入力端子として使用する。
RAPI_FTIOB	FTIOB 端子を入力端子として使用する。

・タイマ W 使用時（RAPI_TIMER_W を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TW_F1, RAPI_TW_F2, RAPI_TW_F4, RAPI_TW_F8 }から 1 つを指定します。省略値は RAPI_TW_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（測定パルス）　{ RAPI_RISING_RISING, RAPI_FALLING_FALLING }から 1 つを指定します。省略値は RAPI_RISING_RISING です。

（割り込み）　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

・タイマ Z 使用時（RAPI_TIMER_Z0～RAPI_TIMER_Z1 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TZ_F1, RAPI_TZ_F2, RAPI_TZ_F4, RAPI_TZ_F8 }から 1 つを指定します。省略値は RAPI_TZ_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。

省略値は RAPI_TIMER_OFF です。

(測定パルス)　{ RAPI_RISING_RISING, RAPI_FALLING_FALLING }から 1 つを指定します。
省略値は RAPI_RISING_RISING です。

(割り込み)　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

(入力端子)　{ RAPI_FTIOA, RAPI_FTIOB }から 1 つを指定します。
省略値は RAPI_FTIOA です。

・タイマ RC 使用時 (RAPI_TIMER_RC を指定時) に指定可能な定義値

(カウントソース)　{ RAPI_TRC_F1, RAPI_TRC_F2, RAPI_TRC_F4, RAPI_TRC_F8, RAPI_TRC_F32,
RAPI_TRC_F40M }から 1 つを指定します。省略値は RAPI_TRC_F1 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

(測定パルス)　{ RAPI_RISING_RISING, RAPI_FALLING_FALLING }から 1 つを指定します。
省略値は RAPI_RISING_RISING です。

(割り込み)　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

(入力端子)　{ RAPI_FTIOA, RAPI_FTIOB }から 1 つを指定します。
省略値は RAPI_FTIOA です。

(デジタルフィルタ用クロック)　{ RAPI_TRC_FILTER_F1, RAPI_TRC_FILTER_F8, RAPI_TRC_FILTER_F32,
RAPI_TRC_FILTER_F}から 1 つを指定します。
省略値は RAPI_TRC_FILTER_F32 です。

・タイマ RD 使用時 (RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時) に指定可能な定義値

(カウントソース)　{ RAPI_TRD_F1, RAPI_TRD_F2, RAPI_TRD_F4, RAPI_TRD_F8, RAPI_TRD_F32,
RAPI_TRD_F40M }から 1 つを指定します。省略値は RAPI_TRD_F1 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

(測定パルス)　{ RAPI_RISING_RISING, RAPI_FALLING_FALLING }から 1 つを指定します。
省略値は RAPI_RISING_RISING です。

(割り込み)　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

(入力端子)　{ RAPI_FTIOA, RAPI_FTIOB }から 1 つを指定します。
省略値は RAPI_FTIOA です。

(デジタルフィルタ用クロック)　{ RAPI_TRD_FILTER_F1, RAPI_TRD_FILTER_F8, RAPI_TRD_FILTER_F32,
RAPI_TRD_FILTER_F }から 1 つを指定します。
省略値は RAPI_TRD_FILTER_F32 です。

[data2]

(M16C)(R8C)
割り込み制御レジスタへ設定する割り込みレベル(0～7)を指定します。

(H8S)
インタラプトプライオリティレジスタへ設定する割り込み優先レベル(0～3)を
指定します。

(H8/300H)
割り込みコントロールレジスタへ設定する割り込みプライオリティレベル(0～1)を
指定します。割り込みコントロールレジスタを持たない CPU の場合は 0 を指定します。

[data3]

(M16C)(H8/300H)

0 を指定します。

(R8C)(H8S)

タイマレジスタまたは、プライマリレジスタへの設定値を 8 ビットで指定します。

[data4]

(M16C)(H8/300H)

0 を指定します。

(R8C)(H8S)

プリスケーラレジスタへの設定値を指定します。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(パルス周期測定モード)

参照

__EnablePulsePeriodMeasurementMode, __DestroyPulsePeriodMeasurementMode, __GetPulsePeriodMeasurementMode

備考

・第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・指定できるタイマは CPU により異なります。
・H8S および、H8/300H で使用する場合、指定したタイマのモジュールスタンバイ解除後に設定をおこないます。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void TimerIntFunc( void ){}

void func( void )
{
    /* Set up timer X as pulse period measurement mode */
    __CreatePulsePeriodMeasurementMode(
        RAPI_TIMER_X|RAPI_TIMER_ON|RAPI_FALLING_FALLING|RAPI_F8,
        5, 0x80, 0x80, TimerIntFunc);
}
```

__EnablePulsePeriodMeasurementMode

要約

< パルス周期測定モード動作制御 >

Boolean __EnablePulsePeriodMeasurementMode(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したパルス周期測定モードに設定されているタイマの動作開始/動作停止を制御します。

[data]

data には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。
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する。
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する。

(R8C)

RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。
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する。
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する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。
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する。
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する。

(H8/300H)

RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
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する。
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する。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(パルス周期測定モード)

参照　__CreatePulsePeriodMeasurementMode, __DestroyPulsePeriodMeasurementMode, __GetPulsePeriodMeasurementMode

備考
- 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できるタイマは CPU により異なります。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void func( void )
{
    /* Enable timer X as pulse period measurement mode */
    __EnablePulsePeriodMeasurementMode( RAPI_TIMER_X|RAPI_TIMER_ON );
}
```

__DestroyPulsePeriodMeasurementMode

要約

< パルス周期測定モード設定破棄 >

Boolean __DestroyPulsePeriodMeasurementMode(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したパルス周期測定モードに設定されいてるタイマの設定を破棄します。

[data]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。

(R8C)

RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。

(H8/300H)

RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(パルス周期測定モード)

参照

__CreatePulsePeriodMeasurementMode, __EnablePulsePeriodMeasurementMode, __GetPulsePeriodMeasurementMode

備考

・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できるタイマは CPU により異なります。
・ H8S および、H8/300H で使用する場合、指定したタイマを、設定破棄後、
  モジュールスタンバイ状態に設定します。

プログラム例

```
#include "rapi_timer_r8c_13.h"
```

```
void func( void )
{
    /* Destroy the setting of timer X as pulse period measurement mode */
    __DestroyPulsePeriodMeasurementMode( RAPI_TIMER_X );
}
```

__GetPulsePeriodMeasurementMode

要約

< パルス周期測定モード測定値取得 >

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

data1	設定データ 1（機種により内容が異なる）
data2	カウンタ値を格納するバッファポインタ

解説

指定したパルス周期測定モードに設定されているタイマのカウンタ値を取得します。

[data1]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。

(R8C)

RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。

(H8/300H)

RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。

[data2]

取得したカウンタ値を格納する配列のポインタを指定します。

(M16C)

・タイマ B 使用時 (RAPI_TIMER_B0～RAPI_TIMER_B2 を指定時)

[0]：タイマ Bi レジスタ(i=0～2)の値を格納します。

(R8C)

・タイマ X 使用時 (RAPI_TIMER_X を指定時)

[0]：プリスケーラ X レジスタの値を格納します。

[1]：タイマ X レジスタの値を格納します。

・タイマ RA 使用時 (RAPI_TIMER_RA を指定時)

[0]：タイマ RA プリスケーラレジスタの値を格納します。

[1]：タイマ RA レジスタの値を格納します。

(H8S)

・タイマ RA 使用時 (RAPI_TIMER_RA を指定時)

[0]：タイマ RA プリスケーラレジスタの値を格納します。

[1]：タイマ RA レジスタの値を格納します。

(H8/300H)

・タイマ W 使用時 (RAPI_TIMER_W を指定時)

[0]：(ジェネラルレジスタ A の値)-(ジェネラルレジスタ C の値)を格納します。

[1]：(ジェネラルレジスタ B の値)-(ジェネラルレジスタ D の値)を格納します。

・タイマ Z 使用時 (RAPI_TIMER_Z0～RAPI_TIMER_Z1 を指定時)

[0]：(ジェネラルレジスタ A_i(i=0,1)の値)-(ジェネラルレジスタ C_i(i=0,1)の値)を格納します。

[1]：(ジェネラルレジスタ B_i(i=0,1)の値)-(ジェネラルレジスタ D_i(i=0,1)の値)を格納します。

・タイマ RC 使用時 (RAPI_TIMER_RC を指定時)

[0]：(ジェネラルレジスタ A の値)-(ジェネラルレジスタ C の値)を格納します。

[1]：(ジェネラルレジスタ B の値)-(ジェネラルレジスタ D の値)を格納します。

・タイマ RD 使用時 (RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時)

[0]：(ジェネラルレジスタ A の値)-(ジェネラルレジスタ C の値)を格納します。

[1]：(ジェネラルレジスタ B の値)-(ジェネラルレジスタ D の値)を格納します。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(パルス周期測定モード)

参照

__CreatePulsePeriodMeasurementMode, __EnablePulsePeriodMeasurementMode, __DestroyPulsePeriodMeasurementMode

備考

・第 1 引数に定義値以外の値を指定した場合の動作は保証されません。

・指定できるタイマは CPU により異なります。

プログラム例

```
#include "rapi_timer_r8c_13.h"

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

    /* Get the measured value of timer X as pulse period measurement mode */
    __GetPulsePeriodMeasurementMode( RAPI_TIMER_X, data );
}
```

__CreatePulseWidthMeasurementMode

要約

< パルス幅測定モード設定 >

Boolean __CreatePulseWidthMeasurementMode(unsigned long data1, unsigned int data2, unsigned int data3, unsigned int data4, void* func)

data1	設定データ 1（機種により内容が異なる）
data2	設定データ 2（機種により内容が異なる）
data3	設定データ 3（機種により内容が異なる）
data4	設定データ 4（機種により内容が異なる）
func	コールバック関数ポインタ （コールバック関数を設定しない場合は 0 を指定する）

解説

指定したタイマをパルス幅測定モードに設定します。

[data1]

data1 には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_B0	タイマ B チャネル 0 を使用する。
RAPI_TIMER_B1	タイマ B チャネル 1 を使用する。
RAPI_TIMER_B2	タイマ B チャネル 2 を使用する。
RAPI_F1	カウントソースに f_1 を選択する。
RAPI_F2	カウントソースに f_2 を選択する。
RAPI_F8	カウントソースに f_8 を選択する。
RAPI_F32	カウントソースに f_{32} を選択する。
RAPI_FC32	カウントソースに f_{C32} を選択する。
RAPI_TIMER_ON	__CreatePulseWidthMeasurementMode 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreatePulseWidthMeasurementMode 内ではタイマを動作停止に設定する。

・タイマ B 使用時（RAPI_TIMER_B0～RAPI_TIMER_B2 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_F32, RAPI_FC32 }から 1 つを指定します。省略値は RAPI_F2 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

(R8C)

RAPI_TIMER_X	タイマ X を使用する。
RAPI_TIMER_RA	タイマ RA を使用する。
RAPI_F1	カウントソースに f_1 を選択する。
RAPI_F2	カウントソースに f_2 を選択する。
RAPI_F8	カウントソースに f_8 を選択する。
RAPI_F32	カウントソースに f_{32} を選択する。
RAPI_FC32	カウントソースに f_{C32} を選択する。
RAPI_FOCO	カウントソースに fOCO 選択する。
RAPI_TIMER_ON	__CreatePulseWidthMeasurementMode 内でタイマを動作開始に設定する。

RAPI_TIMER_OFF	__CreatePulseWidthMeasurementMode 内ではタイマを動作停止に設定する。
RAPI_FALLING_ RISING	測定パルスの立ち下がり－立ち上がり間("L"レベル幅)の測定を選択する。
RAPI_RISING_ FALLING	測定パルスの立ち上がり－立ち下がり間("H"レベル幅)の測定を選択する。
RAPI_FILTER_F1	サンプリング周波数 f_1 のデジタルフィルタ機能を使用する。
RAPI_FILTER_F8	サンプリング周波数 f_8 のデジタルフィルタ機能を使用する。
RAPI_FILTER_F32	サンプリング周波数 f_{32} のデジタルフィルタ機能を使用する。
RAPI_TIOSEL_P1_7	カウントソース入力端子を P1_7 に設定する。
RAPI_TIOSEL_P1_5	カウントソース入力端子を P1_5 に設定する。

・タイマ X 使用時 (RAPI_TIMER_X を指定時) に指定可能な定義値

(カウントソース)　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_F32 }から 1 つを指定します。
省略値は RAPI_F1 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

(測定パルス)　{ RAPI_FALLING_ RISING, RAPI_RISING_ FALLING }から 1 つを指定します。
省略値は RAPI_FALLING_RISING です。

・タイマ RA 使用時 (RAPI_TIMER_RA を指定時) に指定可能な定義値

(カウントソース)　{ RAPI_F1, RAPI_F2, RAPI_F8, RAPI_FC32, RAPI_FOCO }から 1 つを
指定します。省略値は RAPI_F1 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

(測定パルス)　{ RAPI_FALLING_ RISING, RAPI_RISING_ FALLING }から 1 つを指定します。
省略値は RAPI_FALLING_RISING です。

(フィルタ)　{ RAPI_FILTER_F1, RAPI_FILTER_F8, RAPI_FILTER_F32 }から 1 つを
指定します。省略した場合は、「フィルタなし」に設定します。

(入力端子)　{ RAPI_TIOSEL_P1_7, RAPI_TIOSEL_P1_5 }から 1 つを指定します。
省略値は RAPI_TIOSEL_P1_7 です。

(H8S)

RAPI_TIMER_RA	タイマ RA を使用する。
RAPI_TRA_F1	タイマ RA のカウントソースにφのクロックを選択する。
RAPI_TRA_F2	タイマ RA のカウントソースにφ/2 のクロックを選択する。
RAPI_TRA_F8	タイマ RA のカウントソースにφ/8 のクロックを選択する。
RAPI_TRA_F32	タイマ RA のカウントソースにφ/32 のクロックを選択する。
RAPI_TRA_F40	タイマ RA のカウントソースにφ40 のクロックを選択する。
RAPI_TRA_F64	タイマ RA のカウントソースにφ/64 のクロックを選択する。
RAPI_TRA_F128	タイマ RA のカウントソースにφ/128 のクロックを選択する。
RAPI_TRA_FSUB	タイマ RA のカウントソースにサブ発信器出力クロックを選択する。
RAPI_TIMER_ON	__CreatePulseWidthMeasurementMode 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreatePulseWidthMeasurementMode 内ではタイマを動作停止に設定する。
RAPI_UNDERFLOW	アンダフロー割り込みを許可する。
RAPI_INT_MODE_0	割り込み制御モード 0 を指定する。
RAPI_INT_MODE_2	割り込み制御モード 2 を指定する。
RAPI_TRA_GATE_H	IRQ2 端子に"H"が入力されている期間カウントするゲート機能を選択する。

RAPI_FALLING_ RISING	測定パルスの立ち下がり－立ち上がり間("L"レベル幅)の測定を選択する。
RAPI_RISING_ FALLING	測定パルスの立ち上がり－立ち下がり間("H"レベル幅)の測定を選択する。
RAPI_TRA_FILTER_F1	タイマ RA のサンプリング周波φのフィルタ機能を使用する。
RAPI_TRA_FILTER_F8	タイマ RA のサンプリング周波数φ/8 フィルタ機能を使用する。
RAPI_TRA_FILTER_F32	タイマ RA のサンプリング周波数φ/32 のフィルタ機能を使用する。

・タイマ RA 使用時（RAPI_TIMER_RA を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRA_F1, RAPI_TRA_F2, RAPI_TRA_F8, RAPI_TRA_F32, RAPI_TRA_F40, RAPI_TRA_F64, RAPI_TRA_F128, RAPI_TRA_FSUB }から 1 つを指定します。
省略値は RAPI_TRA_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（割り込み）　アンダフロー割り込み要求ありの場合、RAPI_UNDERFLOW を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（割り込み制御モード）　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。
省略値は RAPI_INT_MODE_0 です。

（測定パルス）　{ RAPI_FALLING_RISING, RAPI_RISING_FALLING }から 1 つを指定します。
省略値は RAPI_FALLING_RISING です。

（フィルタ）　{ RAPI_TRA_FILTER_F1, RAPI_TRA_FILTER_F8, RAPI_TRA_FILTER_F32 }から 1 つを指定します。省略した場合は、「フィルタなし」に設定します。

（ゲート機能）　ゲート機能を使用する場合は、RAPI_TRA_GATE_H を指定します。
省略した場合は、「ゲート機能なし」に設定します。

(H8/300H)

RAPI_TIMER_W	タイマ W を使用する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を使用する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を使用する。
RAPI_TIMER_RC	タイマ RC を使用する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を使用する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を使用する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を使用する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を使用する。
RAPI_TW_F1	タイマ W は内部クロックφでカウントする。
RAPI_TW_F2	タイマ W は内部クロックφ/2 でカウントする。
RAPI_TW_F4	タイマ W は内部クロックφ/4 でカウントする。
RAPI_TW_F8	タイマ W は内部クロックφ/8 でカウントする。
RAPI_TZ_F1	タイマ Z は内部クロックφでカウントする。
RAPI_TZ_F2	タイマ Z は内部クロックφ/2 でカウントする。
RAPI_TZ_F4	タイマ Z は内部クロックφ/4 でカウントする。
RAPI_TZ_F8	タイマ Z は内部クロックφ/8 でカウントする。
RAPI_TRC_F1	タイマ RC は内部クロックφでカウントする。
RAPI_TRC_F2	タイマ RC は内部クロックφ/2 でカウントする。
RAPI_TRC_F4	タイマ RC は内部クロックφ/4 でカウントする。
RAPI_TRC_F8	タイマ RC は内部クロックφ/8 でカウントする。

RAPI_TRC_F32	タイマ RC は内部クロックφ/32 でカウントする。
RAPI_TRC_F40M	タイマ RC は内部クロックφ40M でカウントする。
RAPI_TRD_F1	タイマ RD は内部クロックφでカウントする。
RAPI_TRD_F2	タイマ RD は内部クロックφ/2 でカウントする。
RAPI_TRD_F4	タイマ RD は内部クロックφ/4 でカウントする。
RAPI_TRD_F8	タイマ RD は内部クロックφ/8 でカウントする。
RAPI_TRD_F32	タイマ RD は内部クロックφ/32 でカウントする。
RAPI_TRD_F40M	タイマ RD は内部クロックφ40M でカウントする。
RAPI_TIMER_ON	__CreatePulseWidthMeasurementMode 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreatePulseWidthMeasurementMode 内ではタイマを動作停止に設定する。
RAPI_OVERFLOW	オーバフロー割り込みを許可する。
RAPI_TRC_FILTER_F1	タイマ RC でサンプリング周波数 f_1 のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F8	タイマ RC でサンプリング周波数 f_8 のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F32	タイマ RC でサンプリング周波数 f_{32} のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F	タイマ RC でカウントソースと同じサンプリング周波数の デジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F1	タイマ RD でサンプリング周波数 f_1 のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F8	タイマ RD でサンプリング周波数 f_8 のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F32	タイマ RD でサンプリング周波数 f_{32} のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F	タイマ RD でカウントソースと同じサンプリング周波数の デジタルフィルタ機能を使用する。
RAPI_FTIOA	FTIOA 端子を入力端子として使用する。
RAPI_FTIOB	FTIOB 端子を入力端子として使用する。

・タイマ W 使用時（RAPI_TIMER_W を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TW_F1, RAPI_TW_F2, RAPI_TW_F4, RAPI_TW_F8 }から 1 つを指定します。省略値は RAPI_TW_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

・タイマ Z 使用時（RAPI_TIMER_Z0～RAPI_TIMER_Z1 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TZ_F1, RAPI_TZ_F2, RAPI_TZ_F4, RAPI_TZ_F8 }から 1 つを指定します。省略値は RAPI_TZ_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（入力端子）　{ RAPI_FTIOA, RAPI_FTIOB }から 1 つを指定します。省略値は RAPI_FTIOA です。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRC_F1, RAPI_TRC_F2, RAPI_TRC_F4, RAPI_TRC_F8, RAPI_TRC_F32, RAPI_TRC_F40M }から 1 つを指定します。省略値は RAPI_TRC_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。
（入力端子）　　{ RAPI_FTIOA, RAPI_FTIOB }から 1 つを指定します。
省略値は RAPI_FTIOA です。
（デジタルフィルタ用クロック）　　{ RAPI_TRC_FILTER_F1, RAPI_TRC_FILTER_F8, RAPI_TRC_FILTER_F32, RAPI_TRC_FILTER_F }から 1 つを指定します。
省略値は RAPI_TRC_FILTER_F32 です。

・タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時）に指定可能な定義値

（カウントソース）　　{ RAPI_TRD_F1, RAPI_TRD_F2, RAPI_TRD_F4, RAPI_TRD_F8, RAPI_TRD_F32, RAPI_TRD_F40M }から 1 つを指定します。省略値は RAPI_TRD_F1 です。
（設定時動作状態）　　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。
（割り込み）　　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。
（入力端子）　　{ RAPI_FTIOA, RAPI_FTIOB }から 1 つを指定します。
省略値は RAPI_FTIOA です。
（デジタルフィルタ用クロック）　　{ RAPI_TRD_FILTER_F1, RAPI_TRD_FILTER_F8, RAPI_TRD_FILTER_F32, RAPI_TRD_FILTER_F }から 1 つを指定します。
省略値は RAPI_TRD_FILTER_F32 です。

[data2]

(M16C)(R8C)
　割り込み制御レジスタへ設定する割り込みレベル(0～7)を指定します。
(H8S)
　インタラプトプライオリティレジスタへ設定する割り込み優先レベル(0～3)を指定します。
(H8/300H)
　割り込みコントロールレジスタへ設定する割り込みプライオリティレベル(0～1)を指定します。割り込みコントロールレジスタを持たない CPU の場合は 0 を指定します。

[data3]

(M16C)(H8/300H)
　0 を指定します。
(R8C)(H8S)
　タイマレジスタまたは、プライマリレジスタへの設定値を 8 ビットで指定します。

[data4]

(M16C)(H8/300H)
　0 を指定します。
(R8C)(H8S)
　プリスケーラレジスタへの設定値を指定します。

戻り値　　タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能　タイマ(パルス幅測定モード)

参照　__EnablePulseWidthMeasurementMode, __DestroyPulseWidthMeasurementMode, __GetPulseWidthMeasurementMode

備考
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できるタイマは CPU により異なります。
- H8S および、H8/300H で使用する場合、指定したタイマのモジュールスタンバイ解除後に設定をおこないます。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void TimerIntFunc( void ){}

void func( void )
{
    /* Set up timer X as pulse width measurement mode */
    __CreatePulseWidthMeasurementMode(
        RAPI_TIMER_X|RAPI_TIMER_ON|RAPI_RISING_FALLING|RAPI_F8,
        5, 0x80, 0x80, TimerIntFunc);
}
```

__EnablePulseWidthMeasurementMode

要約

< パルス幅測定モード動作制御 >

Boolean __EnablePulseWidthMeasurementMode(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したパルス幅測定モードに設定されているタイマの動作開始/動作停止を制御します。

[data]

data には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。
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する。
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する。

(R8C)

RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。
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する。
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する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。
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する。
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する。

(H8/300H)

RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
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する。
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する。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(パルス幅測定モード)

参照

__CreatePulseWidthMeasurementMode, __DestroyPulseWidthMeasurementMode,

__GetPulseWidthMeasurementMode

備考

・引数に定義値以外の値を指定した場合の動作は保証されません。
・指定できるタイマは CPU により異なります。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void func( void )
{
    /* Disable timer X as pulse width measurement mode */
    __EnablePulseWidthMeasurementMode( RAPI_TIMER_X|RAPI_TIMER_OFF );
}
```

__DestroyPulseWidthMeasurementMode

要約

< パルス幅測定モード設定破棄 >

Boolean __DestroyPulseWidthMeasurementMode(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したパルス周期測定モードに設定されているタイマの設定を破棄します。

[data]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。

(R8C)

RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。

(H8/300H)

RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(パルス幅測定モード)

参照

__CreatePulseWidthMeasurementMode, __EnablePulseWidthMeasurementMode, __GetPulseWidthMeasurementMode

備考

・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できるタイマは CPU により異なります。
・ H8S および、H8/300H で使用する場合、指定したタイマを、設定破棄後、
モジュールスタンバイ状態に設定します。

プログラム例

```
#include "rapi_timer_r8c_13.h"
```

```
void func( void )
{
    /* Destroy the setting of timer X as pulse width measurement mode */
    __DestroyPulseWidthMeasurementMode( RAPI_TIMER_X );
}
```

__GetPulseWidthMeasurementMode

要約

< パルス幅測定モード測定値取得 >

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

data1	設定データ 1 (機種により内容が異なる)
data2	カウンタ値を格納するバッファポインタ

解説

指定したパルス幅測定モードに設定されているタイマのカウンタ値を取得します。

[data1]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。

(R8C)

RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。

(H8/300H)

RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。

[data2]

取得したカウンタ値を格納する配列のポインタを指定します。

(M16C)

・タイマ B 使用時 (RAPI_TIMER_B0～RAPI_TIMER_B2 を指定時)

[0]：タイマ Bi レジスタ(i=0～2)の値を格納します。

(R8C)

・タイマ X 使用時 (RAPI_TIMER_X を指定時)

[0]：プリスケーラ X レジスタの値を格納します。

[1]：タイマ X レジスタの値を格納します。

・タイマ RA 使用時 (RAPI_TIMER_RA を指定時)

[0]：タイマ RA プリスケーラレジスタの値を格納します。

[1]：タイマ RA レジスタの値を格納します。

(H8S)

[0]：タイマ RA プリスケーラレジスタの値を格納します。

[1]：タイマ RA レジスタの値を格納します。

(H8/300H)

・タイマ W 使用時 (RAPI_TIMER_W を指定時)

[0]：(ジェネラルレジスタ A の値)-(ジェネラルレジスタ C の値)を格納します。

[1]：(ジェネラルレジスタ B の値)-(ジェネラルレジスタ D の値)を格納します。

・タイマ Z 使用時 (RAPI_TIMER_Z0～RAPI_TIMER_Z1 を指定時)

[0]：(ジェネラルレジスタ A_i(i=0,1)の値)-(ジェネラルレジスタ C_i(i=0,1)の値)を格納します。

[1]：(ジェネラルレジスタ B_i(i=0,1)の値)-(ジェネラルレジスタ D_i(i=0,1)の値)を格納します。

・タイマ RC 使用時 (RAPI_TIMER_RC を指定時)

[0]：(ジェネラルレジスタ A の値)-(ジェネラルレジスタ C の値)を格納します。

[1]：(ジェネラルレジスタ B の値)-(ジェネラルレジスタ D の値)を格納します。

・タイマ RD 使用時 (RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時)

[0]：(ジェネラルレジスタ A の値)-(ジェネラルレジスタ C の値)を格納します。

[1]：(ジェネラルレジスタ B の値)-(ジェネラルレジスタ D の値)を格納します。

戻り値

タイマ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(パルス幅測定モード)

参照

__CreatePulseWidthMeasurementMode, __EnablePulseWidthMeasurementMode, __DestroyPulseWidthMeasurementMode

備考

・第 1 引数に定義値以外の値を指定した場合の動作は保証されません。

・指定できるタイマは CPU により異なります。

プログラム例

```
#include "rapi_timer_r8c_13.h"

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

    /* Get the measured value of timer X as pulse width measurement mode */
    __GetPulseWidthMeasurementMode( RAPI_TIMER_X, data );
}
```

__CreateInputCapture

要約

〈 インプットキャプチャモード設定 〉

Boolean __CreateInputCapture(unsigned long data1, unsigned int* data2, unsigned int* data3, unsigned int* data4, void data5)**

data1	設定データ 1（機種により内容が異なる）
data2	設定データ 2（機種により内容が異なる）
data3	設定データ 3（機種により内容が異なる）
data4	設定データ 4（機種により内容が異なる）
data5	設定データ 5（機種により内容が異なる）

解説

指定したタイマまたはインテリジェント I/O をインプットキャプチャモードに設定します。

[data1]

data1 には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_S	タイマ S を使用する。
RAPI_F1	カウントソースに f_1 を選択する。
RAPI_F2	カウントソースに f_2 を選択する。
RAPI_TIMER_ON	__CreateInputCapture 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateInputCapture 内ではタイマを動作停止に設定する。
RAPI_OVERFLOW_BIT14	ビット 14 のオーバフローをベースタイマ割り込みに設定する。
RAPI_OVERFLOW_BIT15	ビット 15 のオーバフローをベースタイマ割り込みに設定する。

・タイマ S 使用時（RAPI_TIMER_S を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2 }から 1 つを指定します。省略値は RAPI_F2 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（ベースタイマ）　{ RAPI_OVERFLOW_BIT14, RAPI_OVERFLOW_BIT15 }から 1 つを指定します。
省略値は RAPI_OVERFLOW_BIT15 です。

(R8C)

RAPI_TIMER_C	タイマ C を使用する。
RAPI_TIMER_RC	タイマ RC を使用する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を使用する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を使用する。
RAPI_TIMER_RF	タイマ RF を使用する。
RAPI_F1	カウントソースに f_1 を選択する。
RAPI_F8	カウントソースに f_8 を選択する。
RAPI_F32	カウントソースに f_{32} を選択する。
RAPI_FRING_FAST	カウントソースに $f_{RING-fast}$ を選択する。
RAPI_TRC_F1	タイマ RC はカウントソースに f_1 を選択する。
RAPI_TRC_F2	タイマ RC はカウントソースに f_2 を選択する。
RAPI_TRC_F4	タイマ RC はカウントソースに f_4 を選択する。

RAPI_TRC_F8	タイマ RC はカウントソースに f_8 を選択する。
RAPI_TRC_F32	タイマ RC はカウントソースに f_{32} を選択する。
RAPI_TRC_F40M	タイマ RC はカウントソースに fOCO40M を選択する。
RAPI_TRD_F1	タイマ RD はカウントソースに f_1 を選択する。
RAPI_TRD_F2	タイマ RD はカウントソースに f_2 を選択する。
RAPI_TRD_F4	タイマ RD はカウントソースに f_4 を選択する。
RAPI_TRD_F8	タイマ RD はカウントソースに f_8 を選択する。
RAPI_TRD_F32	タイマ RD はカウントソースに f_{32} を選択する。
RAPI_TRD_F40M	タイマ RD はカウントソースに fOCO40M を選択する。
RAPI_TRF_F1	タイマ RF でカウントソースに f_1 を選択する。
RAPI_TRF_F8	タイマ RF でカウントソースに f_8 を選択する。
RAPI_TRF_F32	タイマ RF でカウントソースに f_{32} を選択する。
RAPI_TIMER_ON	__CreateInputCapture 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateInputCapture 内ではタイマを動作停止に設定する。
RAPI_RISING	測定パルスの有効エッジに立ち上がりエッジを選択する。
RAPI_FALLING	測定パルスの有効エッジに立ち下がりエッジを選択する。
RAPI_BOTH	測定パルスの有効エッジに立ち上がりと立ち下がりの両エッジを選択する。
RAPI_TRC_RISING	タイマ RC で測定パルスの有効エッジに立ち上がりエッジを選択する。
RAPI_TRC_FALLING	タイマ RC で測定パルスの有効エッジに立ち下がりエッジを選択する。
RAPI_TRC_BOTH	タイマ RC で測定パルスの有効エッジに立ち上がりと立ち下がりの両エッジを選択する。
RAPI_TRF_RISING	タイマ RF で測定パルスの有効エッジに立ち上がりエッジを選択する。
RAPI_TRF_FALLING	タイマ RF で測定パルスの有効エッジに立ち下がりエッジを選択する。
RAPI_TRF_BOTH	タイマ RF で測定パルスの有効エッジに立ち上がりと立ち下がりの両エッジを選択する。
RAPI_FILTER_F1	サンプリング周波数 f_1 のデジタルフィルタ機能を使用する。
RAPI_FILTER_F8	サンプリング周波数 f_8 のデジタルフィルタ機能を使用する。
RAPI_FILTER_F32	サンプリング周波数 f_{32} のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F1	タイマ RC でサンプリング周波数 f_1 のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F8	タイマ RC でサンプリング周波数 f_8 のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F32	タイマ RC でサンプリング周波数 f_{32} のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F	タイマ RC でカウントソースと同じサンプリング周波数のデジタルフィルタ機能を使用する。
RAPI_INT3_TRIGGER	TC_{IN} 入力のトリガ機能を選択する。
RAPI_FRING128	$f_{RING128}$ のトリガ機能を選択する。
RAPI_OVERFLOW	オーバフロー割り込みを許可する。
RAPI_INPUT_CAPTURE_A	GRA のインプットキャプチャ割り込みを許可する。
RAPI_INPUT_CAPTURE_B	GRB のインプットキャプチャ割り込みを許可する。
RAPI_INPUT_CAPTURE_C	GRC のインプットキャプチャ割り込みを許可する。

RAPI_INPUT_ CAPTURE_D	GRD のインプットキャプチャ割り込みを許可する。
RAPI_COUNT_CLEAR_A	カウンタクリア要因に GRA のインプットキャプチャを指定する。
RAPI_COUNT_CLEAR_B	カウンタクリア要因に GRB のインプットキャプチャを指定する。
RAPI_COUNT_CLEAR_C	カウンタクリア要因に GRC のインプットキャプチャを指定する。
RAPI_COUNT_CLEAR_D	カウンタクリア要因に GRD のインプットキャプチャを指定する。
RAPI_COUNT_CLEAR_ SYNC	同期動作をしている他のチャネルのカウンタクリアに同期して カウンタをクリアする。
RAPI_TIMER_SYNC	チャネル 0 とチャネル 1 でタイマ同期する。
RAPI_TRD_FILTER_F1	タイマ RD でサンプリング周波数 f_1 のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F8	タイマ RD でサンプリング周波数 f_8 のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F32	タイマ RD でサンプリング周波数 f_{32} のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F	タイマ RD でカウントソースと同じサンプリング周波数の デジタルフィルタ機能を使用する。

・タイマ C 使用時（RAPI_TIMER_C を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F8, RAPI_F32, RAPI_FRING_FAST }から 1 つを指定します。
省略値は RAPI_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（測定パルス）　{ RAPI_RISING, RAPI_FALLING, RAPI_BOTH }から 1 つを指定します。
省略値は RAPI_RISING です。

（フィルタ）　{ RAPI_FILTER_F1, RAPI_FILTER_F8, RAPI_FILTER_F32 }から 1 つを
指定します。省略した場合は、「フィルタなし」に設定します。

（入力）　{ RAPI_INT3_TRIGGER, RAPI_FRING128 }から 1 つを指定します。
省略値は RAPI_INT3_TRIGGER です。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRC_F1, RAPI_TRC_F2, RAPI_TRC_F4, RAPI_TRC_F8, RAPI_TRC_F32,
RAPI_TRC_F40M }から 1 つを指定します。省略値は RAPI_TRC_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、
インプットキャプチャ A 割り込み要求ありの場合は
RAPI_INPUT_CAPTURE_A を、インプットキャプチャ B 割り込み要求ありの
場合 RAPI_INPUT_CAPTURE_B を、インプットキャプチャ C 割り込み
要求ありの場合 RAPI_INPUT_CAPTURE_C を、インプットキャプチャ D
割り込み要求ありの場合 RAPI_INPUT_CAPTURE_D を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（デジタルフィルタ用クロック）　{ RAPI_TRC_FILTER_F1, RAPI_TRC_FILTER_F8, RAPI_TRC_FILTER_F32,
RAPI_TRC_FILTER_F }から 1 つを指定します。
省略値は RAPI_TRC_FILTER_F32 です。

・タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD1 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRD_F1, RAPI_TRD_F2, RAPI_TRD_F4, RAPI_TRD_F8, RAPI_TRD_F32,
RAPI_TRD_F40M }から 1 つを指定します。省略値は RAPI_TRD_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。

省略値は RAPI_TIMER_OFF です。

(割り込み)　オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、インプットキャプチャ A 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_A を、インプットキャプチャ B 割り込み要求ありの場合 RAPI_INPUT_CAPTURE_B を、インプットキャプチャ C 割り込み要求ありの場合 RAPI_INPUT_CAPTURE_C を、インプットキャプチャ D 割り込み要求ありの場合 RAPI_INPUT_CAPTURE_D を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

(カウンタクリア)　カウンタクリア要因に GRA のインプットキャプチャを指定する場合は、RAPI_COUNT_CLEAR_A を、GRB のインプットキャプチャを指定する場合は、RAPI_COUNT_CLEAR_B を、GRC のインプットキャプチャを指定する場合は、RAPI_COUNT_CLEAR_C を、GRD のインプットキャプチャを指定する場合は、RAPI_COUNT_CLEAR_D を、同期動作をしている他のチャネルのカウンタクリアに同期する場合は、RAPI_COUNT_CLEAR_SYNC を指定します。

(同期)　チャネル 0 とチャネル 1 でタイマ同期する場合は、RAPI_TIMER_SYNC を指定します。同期を指定しない場合は、「チャネル 0 とチャネル 1 は独立動作」を設定します。

(デジタルフィルタ用クロック)　{ RAPI_TRD_FILTER_F1, RAPI_TRD_FILTER_F8, RAPI_TRD_FILTER_F32, RAPI_TRD_FILTER_F }から 1 つを指定します。
省略値は RAPI_TRD_FILTER_F32 です。

・タイマ RF 使用時 (RAPI_TIMER_RF を指定時) に指定可能な定義値

(カウントソース)　{ RAPI_TRF_F1, RAPI_TRF_F8, RAPI_TRF_F32 }から 1 つを指定します。
省略値は RAPI_TRF_F1 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

(測定パルス)　{ RAPI_TRF_RISING, RAPI_TRF_FALLING, RAPI_TRF_BOTH }から 1 つを指定します。省略値は RAPI_TRF_RISING です。

(フィルタ)　{ RAPI_FILTER_F1, RAPI_FILTER_F8, RAPI_FILTER_F32 }から 1 つを指定します。省略した場合は、「フィルタなし」に設定します。

(H8S)

RAPI_TIMER_RC	タイマ RC を使用する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を使用する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を使用する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を使用する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を使用する。
RAPI_TIMER_RG	タイマ RG を使用する。
RAPI_TRC_F1	タイマ RC は内部クロックφでカウントする。
RAPI_TRC_F2	タイマ RC は内部クロックφ/2 でカウントする。
RAPI_TRC_F4	タイマ RC は内部クロックφ/4 でカウントする。
RAPI_TRC_F8	タイマ RC は内部クロックφ/8 でカウントする。
RAPI_TRC_F32	タイマ RC は内部クロックφ/32 でカウントする。
RAPI_TRC_F40	タイマ RC は内部クロックφ40 でカウントする。
RAPI_TRD_F1	タイマ RD は内部クロックφでカウントする。
RAPI_TRD_F2	タイマ RD は内部クロックφ/2 でカウントする。

RAPI_TRD_F4	タイマ RD は内部クロックφ/4 でカウントする。
RAPI_TRD_F8	タイマ RD は内部クロックφ/8 でカウントする。
RAPI_TRD_F32	タイマ RD は内部クロックφ/32 でカウントする。
RAPI_TRD_F40	タイマ RD は内部クロックφ40 でカウントする。
RAPI_TRG_F1	タイマ RG は内部クロックφでカウントする。
RAPI_TRG_F2	タイマ RG は内部クロックφ/2 でカウントする。
RAPI_TRG_F4	タイマ RG は内部クロックφ/4 でカウントする。
RAPI_TRG_F8	タイマ RG は内部クロックφ/8 でカウントする。
RAPI_TRG_F32	タイマ RG は内部クロックφ/32 でカウントする。
RAPI_TRG_F40	タイマ RG は内部クロックφ40 でカウントする。
RAPI_TIMER_ON	__CreateInputCapture 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateInputCapture 内ではタイマを動作停止に設定する。
RAPI_OVERFLOW	オーバフロー割り込みを許可する。
RAPI_INPUT_ CAPTURE_A	GRA のインプットキャプチャ割り込みを許可する。
RAPI_INPUT_ CAPTURE_B	GRB のインプットキャプチャ割り込みを許可する。
RAPI_INPUT_ CAPTURE_C	GRC のインプットキャプチャ割り込みを許可する。
RAPI_INPUT_ CAPTURE_D	GRD のインプットキャプチャ割り込みを許可する。
RAPI_INT_MODE_0	割り込み制御モード 0 を指定する。
RAPI_INT_MODE_2	割り込み制御モード 2 を指定する。
RAPI_TRC_COUNT_ CLEAR_A	タイマ RC のカウンタクリア要因に GRA のインプットキャプチャを指定する。
RAPI_TRD_COUNT_ CLEAR_A	タイマ RD のカウンタクリア要因に GRA のインプットキャプチャを指定する。
RAPI_TRD_COUNT_ CLEAR_B	タイマ RD のカウンタクリア要因に GRB のインプットキャプチャを指定する。
RAPI_TRD_COUNT_ CLEAR_C	タイマ RD のカウンタクリア要因に GRC のインプットキャプチャを指定する。
RAPI_TRD_COUNT_ CLEAR_D	タイマ RD のカウンタクリア要因に GRD のインプットキャプチャを指定する。
RAPI_TRD_COUNT_ CLEAR_SYNC	タイマ RD と同期動作をしている他のチャネルのカウンタクリアに 同期してカウンタをクリアする。
RAPI_TRG_COUNT_ CLEAR_A	タイマ RG のカウンタクリア要因に GRA のインプットキャプチャを指定する。
RAPI_TRG_COUNT_ CLEAR_B	タイマ RG のカウンタクリア要因に GRB のインプットキャプチャを指定する。
RAPI_TIMER_SYNC	チャネル 0 とチャネル 1 でタイマ同期する。
RAPI_TRC_FILTER_F1	タイマ RC でサンプリング周波数φのデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F8	タイマ RC でサンプリング周波数φ/8 のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F32	タイマ RC でサンプリング周波数φ/32 のデジタルフィルタ機能を使用する。

RAPI_TRC_FILTER_F	タイマ RC でカウントソースと同じサンプリング周波数の デジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F1	タイマ RD でサンプリング周波数φのデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F8	タイマ RD でサンプリング周波数φ/8 のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F32	タイマ RD でサンプリング周波数φ/32 のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F	タイマ RD でカウントソースと同じサンプリング周波数の デジタルフィルタ機能を使用する。
RAPI_TRG_FILTER_F1	タイマ RG でサンプリング周波数φのデジタルフィルタ機能を使用する。
RAPI_TRG_FILTER_F8	タイマ RG でサンプリング周波数φ/8 のデジタルフィルタ機能を使用する。
RAPI_TRG_FILTER_F32	タイマ RG でサンプリング周波数φ/32 のデジタルフィルタ機能を使用する。
RAPI_TRG_FILTER_F	タイマ RG でカウントソースと同じサンプリング周波数の デジタルフィルタ機能を使用する。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRC_F1, RAPI_TRC_F2, RAPI_TRC_F4, RAPI_TRC_F8, RAPI_TRC_F32, RAPI_TRC_F40 }から 1 つを指定します。省略値は RAPI_TRC_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、インプットキャプチャ A 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_A を、インプットキャプチャ B 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_B を、インプットキャプチャ C 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_C を、インプットキャプチャ D 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_D を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（割り込み制御モード）　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。省略値は RAPI_INT_MODE_0 です。

（カウンタクリア）　カウンタクリア要因に GRA のインプットキャプチャを指定する場合は、RAPI_TRC_COUNT_CLEAR_A を指定します。

（デジタルフィルタ用クロック）　{ RAPI_TRC_FILTER_F1, RAPI_TRC_FILTER_F8, RAPI_TRC_FILTER_F32, RAPI_TRC_FILTER_F }から 1 つを指定します。省略値は RAPI_TRC_FILTER_F32 です。

・タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRD_F1, RAPI_TRD_F2, RAPI_TRD_F4, RAPI_TRD_F8, RAPI_TRD_F32, RAPI_TRD_F40 }から 1 つを指定します。省略値は RAPI_TRD_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、インプットキャプチャ A 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_A を、インプットキャプチャ B 割り込み要求ありの場合 RAPI_INPUT_CAPTURE_B を、インプットキャプチャ C 割り込み要求ありの場合 RAPI_INPUT_CAPTURE_C を、インプットキャプチャ D 割り込み要求ありの場合 RAPI_INPUT_CAPTURE_D を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（割り込み　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。

制御モード)　　省略値は RAPI_INT_MODE_0 です。

(カウンタクリア)　　カウンタクリア要因に GRA のインプットキャプチャを指定する場合は、RAPI_TRD_COUNT_CLEAR_A を、GRB のインプットキャプチャを指定する場合は、RAPI_TRD_COUNT_CLEAR_B を、GRC のインプットキャプチャを指定する場合は、RAPI_TRD_COUNT_CLEAR_C を、GRD のインプットキャプチャを指定する場合は、RAPI_TRD_COUNT_CLEAR_D を、同期動作をしている他のチャネルのカウンタクリアに同期する場合は、RAPI_TRD_COUNT_CLEAR_SYNC を指定します。

(同期)　　チャネル 0 とチャネル 1 でタイマ同期する場合は、RAPI_TIMER_SYNC を指定します。同期を指定しない場合は、「チャネル 0 とチャネル 1 は独立動作」を設定します。

(デジタルフィルタ用クロック)　　{ RAPI_TRD_FILTER_F1, RAPI_TRD_FILTER_F8, RAPI_TRD_FILTER_F32, RAPI_TRD_FILTER_F }から 1 つを指定します。
省略値は RAPI_TRD_FILTER_F32 です。

・タイマ RG 使用時 (RAPI_TIMER_RG を指定時) に指定可能な定義値

(カウントソース)　　{ RAPI_TRG_F1, RAPI_TRG_F2, RAPI_TRG_F4, RAPI_TRG_F8, RAPI_TRG_F32, RAPI_TRG_F40 }から 1 つを指定します。省略値は RAPI_TRG_F1 です。

(設定時動作状態)　　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

(割り込み)　　オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、インプットキャプチャ A 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_A を、インプットキャプチャ B 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_B を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

(割り込み制御モード)　　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。
省略値は RAPI_INT_MODE_0 です。

(カウンタクリア)　　カウンタクリア要因に GRA のインプットキャプチャを指定する場合は、RAPI_TRG_COUNT_CLEAR_A を、GRB のインプットキャプチャを指定する場合は、RAPI_TRG_COUNT_CLEAR_B を指定します。

(デジタルフィルタ用クロック)　　{ RAPI_TRG_FILTER_F1, RAPI_TRG_FILTER_F8, RAPI_TRG_FILTER_F32, RAPI_TRG_FILTER_F }から 1 つを指定します。
省略値は RAPI_TRG_FILTER_F32 です。

(H8/300H)

RAPI_TIMER_W	タイマ W を使用する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を使用する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を使用する。
RAPI_TIMER_RC	タイマ RC を使用する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を使用する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を使用する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を使用する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を使用する。
RAPI_TW_F1	タイマ W は内部クロックφでカウントする。
RAPI_TW_F2	タイマ W は内部クロックφ/2 でカウントする。
RAPI_TW_F4	タイマ W は内部クロックφ/4 でカウントする。

RAPI_TW_F8	タイマ W は内部クロックφ/8 でカウントする。
RAPI_TZ_F1	タイマ Z は内部クロックφでカウントする。
RAPI_TZ_F2	タイマ Z は内部クロックφ/2 でカウントする。
RAPI_TZ_F4	タイマ Z は内部クロックφ/4 でカウントする。
RAPI_TZ_F8	タイマ Z は内部クロックφ/8 でカウントする。
RAPI_TRC_F1	タイマ RC は内部クロックφでカウントする。
RAPI_TRC_F2	タイマ RC は内部クロックφ/2 でカウントする。
RAPI_TRC_F4	タイマ RC は内部クロックφ/4 でカウントする。
RAPI_TRC_F8	タイマ RC は内部クロックφ/8 でカウントする。
RAPI_TRC_F32	タイマ RC は内部クロックφ/32 でカウントする。
RAPI_TRC_F40M	タイマ RC は内部クロックφ40M でカウントする。
RAPI_TRD_F1	タイマ RD は内部クロックφでカウントする。
RAPI_TRD_F2	タイマ RD は内部クロックφ/2 でカウントする。
RAPI_TRD_F4	タイマ RD は内部クロックφ/4 でカウントする。
RAPI_TRD_F8	タイマ RD は内部クロックφ/8 でカウントする。
RAPI_TRD_F32	タイマ RD は内部クロックφ/32 でカウントする。
RAPI_TRD_F40M	タイマ RD は内部クロックφ40M でカウントする。
RAPI_TIMER_ON	__CreateInputCapture 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateInputCapture 内ではタイマを動作停止に設定する。
RAPI_OVERFLOW	オーバフロー割り込みを許可する。
RAPI_INPUT_ CAPTURE_A	GRA のインプットキャプチャ割り込みを許可する。
RAPI_INPUT_ CAPTURE_B	GRB のインプットキャプチャ割り込みを許可する。
RAPI_INPUT_ CAPTURE_C	GRC のインプットキャプチャ割り込みを許可する。
RAPI_INPUT_ CAPTURE_D	GRD のインプットキャプチャ割り込みを許可する。
RAPI_COUNT_CLEAR_A	カウンタクリア要因に GRA のインプットキャプチャを指定する。
RAPI_COUNT_CLEAR_B	カウンタクリア要因に GRB のインプットキャプチャを指定する。
RAPI_COUNT_CLEAR_C	カウンタクリア要因に GRC のインプットキャプチャを指定する。
RAPI_COUNT_CLEAR_D	カウンタクリア要因に GRD のインプットキャプチャを指定する。
RAPI_COUNT_CLEAR_ SYNC	同期動作をしている他のチャネルのカウンタクリアに同期して カウンタをクリアする。
RAPI_TIMER_SYNC	チャネル 0 とチャネル 1 でタイマ同期する。
RAPI_TRC_FILTER_F1	タイマ RC でサンプリング周波数φのデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F8	タイマ RC でサンプリング周波数φ/2 のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F32	タイマ RC でサンプリング周波数φ/32 のデジタルフィルタ機能を使用する。
RAPI_TRC_FILTER_F	タイマ RC でカウントソースと同じサンプリング周波数の デジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F1	タイマ RD でサンプリング周波数φのデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F8	タイマ RD でサンプリング周波数φ/2 のデジタルフィルタ機能を使用する。
RAPI_TRD_FILTER_F32	タイマ RD でサンプリング周波数φ/32 のデジタルフィルタ機能を使用する。

RAPI_TRD_FILTER_F	タイマ RD でカウントソースと同じサンプリング周波数の デジタルフィルタ機能を使用する。

・タイマ W 使用時（RAPI_TIMER_W を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TW_F1, RAPI_TW_F2, RAPI_TW_F4, RAPI_TW_F8 }から 1 つを指定します。省略値は RAPI_TW_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、インプットキャプチャ A 割り込み要求ありの場合は RAPI_INPUT_ CAPTURE_A を、インプットキャプチャ B 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_B を、インプットキャプチャ C 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_C を、インプットキャプチャ D 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_D を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

・タイマ Z 使用時（RAPI_TIMER_Z0～RAPI_TIMER_Z1 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TZ_F1, RAPI_TZ_F2, RAPI_TZ_F4, RAPI_TZ_F8 }から 1 つを指定します。省略値は RAPI_TZ_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、インプットキャプチャ A 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_A を、インプットキャプチャ B 割り込み要求ありの場合 RAPI_INPUT_CAPTURE_B を、インプットキャプチャ C 割り込み要求ありの場合 RAPI_INPUT_CAPTURE_C を、インプットキャプチャ D 割り込み要求ありの場合 RAPI_INPUT_CAPTURE_D を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（カウンタクリア）　カウンタクリア要因に GRA のインプットキャプチャを指定する場合は、RAPI_COUNT_CLEAR_A を、GRB のインプットキャプチャを指定する場合は、RAPI_COUNT_CLEAR_B を、GRC のインプットキャプチャを指定する場合は、RAPI_COUNT_CLEAR_C を、GRD のインプットキャプチャを指定する場合は、RAPI_COUNT_CLEAR_D を、同期動作をしている他のチャネルのカウンタクリアに同期する場合は、RAPI_COUNT_CLEAR_SYNC を指定します。

（同期）　チャネル 0 とチャネル 1 でタイマ同期する場合は、RAPI_TIMER_SYNC を指定します。同期を指定しない場合は、「チャネル 0 とチャネル 1 は独立動作」を設定します。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRC_F1, RAPI_TRC_F2, RAPI_TRC_F4, RAPI_TRC_F8, RAPI_TRC_F32, RAPI_TRC_F40M }から 1 つを指定します。省略値は RAPI_TRC_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、インプットキャプチャ A 割り込み要求ありの場合は RAPI_INPUT_ CAPTURE_A を、インプットキャプチャ B 割り込み要求ありの場合は RAPI_INPUT_CAPTURE_B を、インプットキャプチャ C 割り込み

要求ありの場合は RAPI_INPUT_CAPTURE_C を、インプットキャプチャ D
割り込み要求ありの場合は RAPI_INPUT_CAPTURE_D を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

(デジタルフィルタ用クロック) { RAPI_TRC_FILTER_F1, RAPI_TRC_FILTER_F8, RAPI_TRC_FILTER_F32, RAPI_TRC_FILTER_F }から 1 つを指定します。
省略値は RAPI_TRC_FILTER_F32 です。

・タイマ RD 使用時 (RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時) に指定可能な定義値

(カウントソース) { RAPI_TRD_F1, RAPI_TRD_F2, RAPI_TRD_F4, RAPI_TRD_F8, RAPI_TRD_F32, RAPI_TRD_F40M }から 1 つを指定します。省略値は RAPI_TRD_F1 です。

(設定時動作状態) { RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

(割り込み) オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、
インプットキャプチャ A 割り込み要求ありの場合は
RAPI_INPUT_CAPTURE_A を、インプットキャプチャ B 割り込み要求ありの
場合 RAPI_INPUT_CAPTURE_B を、インプットキャプチャ C 割り込み
要求ありの場合 RAPI_INPUT_CAPTURE_C を、インプットキャプチャ D
割り込み要求ありの場合 RAPI_INPUT_CAPTURE_D を指定します。
割り込みを指定しない場合は、「割り込み要求なし」に設定します。

(カウンタクリア) カウンタクリア要因に GRA のインプットキャプチャを指定する場合は、
RAPI_COUNT_CLEAR_A を、GRB のインプットキャプチャを指定する場合は、
RAPI_COUNT_CLEAR_B を、GRC のインプットキャプチャを指定する場合は、
RAPI_COUNT_CLEAR_C を、GRD のインプットキャプチャを指定する場合は、
RAPI_COUNT_CLEAR_D を、同期動作をしている他のチャネルのカウンタ
クリアに同期する場合は、RAPI_COUNT_CLEAR_SYNC を指定します。

(同期) チャネル 0 とチャネル 1 でタイマ同期する場合は、RAPI_TIMER_SYNC を
指定します。同期を指定しない場合は、「チャネル 0 とチャネル 1 は
独立動作」を設定します。

(デジタルフィルタ用クロック) { RAPI_TRD_FILTER_F1, RAPI_TRD_FILTER_F8, RAPI_TRD_FILTER_F32, RAPI_TRD_FILTER_F }から 1 つを指定します。
省略値は RAPI_TRD_FILTER_F32 です。

[data2]

(M16C)

割り込みレベルを格納する配列のポインタを指定します。
[0] : IC/OC ベースタイマ割り込みのレベル(0～7)を指定します。
[1] : IC/OC 割り込み 0 のレベル(0～7)を指定します。
[2] : IC/OC 割り込み 1 のレベル(0～7)を指定します。

(R8C)

・タイマ C 使用時 (RAPI_TIMER_C を指定時) 、タイマ RC 使用時 (RAPI_TIMER_RC を指定時)
または、タイマ RD 使用時 (RAPI_TIMER_RD0～RAPI_TIMER_RD1 を指定時)

割り込み制御レジスタへ設定する割り込みレベル(0～7)を格納した変数のポインタを
指定します。

・タイマ RF 使用時 (RAPI_TIMER_RF を指定時)

割り込みレベルを格納する配列のポインタを指定します。
[0] : タイマ RF 割り込みのレベル(0～7)を指定します。

[1]：インプットキャプチャー割り込みのレベル(0～7)を指定します。

(H8S)

インタラプトプライオリティレジスタへ設定する割り込み優先レベル(0～3)を格納した変数のポインタを指定します。

(H8/300H)

割り込みコントロールレジスタへ設定する割り込みプライオリティレベル(0～1)を格納した変数のポインタを指定します。

割り込みコントロールレジスタを持たない CPU の場合は 0 を指定します。

[data3]

(M16C)

時間計測制御レジスタへの設定値を格納する配列のポインタを指定します。

[0]：時間計測制御レジスタ 0 への設定値を指定します。

[1]：時間計測制御レジスタ 1 への設定値を指定します。

[2]：時間計測制御レジスタ 2 への設定値を指定します。

[3]：時間計測制御レジスタ 3 への設定値を指定します。

[4]：時間計測制御レジスタ 4 への設定値を指定します。

[5]：時間計測制御レジスタ 5 への設定値を指定します。

[6]：時間計測制御レジスタ 6 への設定値を指定します。

[7]：時間計測制御レジスタ 7 への設定値を指定します。

配列の各要素には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

RAPI_IC_RISING	測定パルスの有効エッジに立ち上がりエッジを選択する。
RAPI_IC_FALLING	測定パルスの有効エッジに立ち下がりエッジを選択する。
RAPI_IC_BOTH	測定パルスの有効エッジに立ち上がりと立ち下がりの両エッジを選択する。
RAPI_FILTER_F1_F2	サンプリング周波数 f_1 または f_2 のデジタルフィルタ機能を使用する。
RAPI_FILTER_FBT1	サンプリング周波数 f_{BT1} のデジタルフィルタ機能を使用する。
RAPI_GATE	ゲート機能を使用する。
RAPI_GATE_CLEAR	ベースタイマと G1POk レジスタの一致により、ゲート機能を解除する。
RAPI_PRESCALER	プリスケーラ機能を使用する。

・時間計測レジスタ 0～7 に指定可能な定義値

(測定パルス)　{ RAPI_IC_RISING, RAPI_IC_FALLING, RAPI_IC_BOTH }から 1 つを指定します。測定パルスを指定しない場合は、時間計測レジスタを「時間計測しない」に設定します。

(フィルタ)　{ RAPI_FILTER_F1_F2, RAPI_FILTER_FBT1 }から 1 つを指定します。フィルタを指定しない場合は、「デジタルフィルタなし」に設定します。

・時間計測レジスタ 6～7 に指定可能な定義値

(ゲート)　ゲート機能を使用する場合は、RAPI_GATE を指定します。RAPI_GATE を指定しない場合は、「ゲート機能不使用」に設定します。
RAPI_GATE_CLEAR は RAPI_GATE と同時に指定してください。

(プリスケーラ)　プリスケーラ機能を使用する場合は、RAPI_PRESCALER を指定します。RAPI_PRESCALER を指定しない場合は、「プリスケーラ機能なし」に設定します。

(R8C)

・タイマ C 使用時（RAPI_TIMER_C を指定時）、タイマ RF 使用時（RAPI_TIMER_RF を指定時）

0 を指定します。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）

各有効エッジの設定を格納する配列のポインタを指定します。

[0]：TRCIOA 端子の有効エッジを指定します。

[1]：TRCIOB 端子の有効エッジを指定します。

[2]：TRCIOC 端子の有効エッジを指定します。

[3]：TRCIOD 端子の有効エッジを指定します。

配列の各要素には、測定パルスの有効エッジとして{ RAPI_TRC_RISING, RAPI_TRC_FALLING, RAPI_TRC_BOTH }をから 1 つを設定することができます。

また、デジタルフィルタ機能ありの場合は、RAPI_FILTER_ON を設定します。

使用しないインプットキャプチャに対応する要素には、0 を設定してください。

RAPI_TRC_RISING	タイマ RC の測定パルスの有効エッジに立ち上がりエッジを選択する。
RAPI_TRC_FALLING	タイマ RC は測定パルスの有効エッジに立ち下がりエッジを選択する。
RAPI_TRC_BOTH	タイマ RC は測定パルスの有効エッジに立ち上がりと立ち下がりの両エッジを選択する。
RAPI_FILTER_ON	デジタルフィルタ機能ありを選択する。 RAPI_FILTER_ON を指定しない場合は、ディジタルフィルタ機能なし。

・タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD1 を指定時）

各有効エッジの設定を格納する配列のポインタを指定します。

[0]：TRDIOAi(i=0,1)端子の有効エッジを指定します。

[1]：TRDIOBi(i=0,1)端子の有効エッジを指定します。

[2]：TRDIOCi(i=0,1)端子の有効エッジを指定します。

[3]：TRDIODi(i=0,1)端子の有効エッジを指定します。

配列の各要素には、測定パルスの有効エッジとして{ RAPI_TRD_RISING, RAPI_TRD_FALLING, RAPI_TRD_BOTH }をから 1 つを設定することができます。

また、デジタルフィルタ機能ありの場合は、RAPI_FILTER_ON を設定します。

チャネル 0 の TRDIOA0 端子入力に fOCO128 信号を設定する場合は、RAPI_FOCO128 を指定します。使用しないチャネルに対応する要素には、0 を設定してください。

RAPI_TRD_RISING	タイマ RD の測定パルスの有効エッジに立ち上がりエッジを選択する。
RAPI_TRD_FALLING	タイマ RD の測定パルスの有効エッジに立ち下がりエッジを選択する。
RAPI_TRD_BOTH	タイマ RD の測定パルスの有効エッジに立ち上がりと立ち下がりの両エッジを選択する。
RAPI_FILTER_ON	デジタルフィルタ機能ありを選択する。 RAPI_FILTER_ON を指定しない場合は、ディジタルフィルタ機能なし。
RAPI_FOCO128	TRDIOA0 端子入力に fOCO128 信号を選択する。チャネル 0 の TRDIOA0 端子の有効エッジのみ指定可能。

(H8S)

各有効エッジの設定を格納する配列のポインタを指定します。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）または、タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD1 を指定時）

[0]：FTIOA 端子の有効エッジを指定します。

[1]：FTIOB 端子の有効エッジを指定します。

[2]：FTIOC 端子の有効エッジを指定します。

[3]：FTIOD 端子の有効エッジを指定します。

・タイマ RG 使用時（RAPI_TIMER_RG を指定時）

[0]：FTIOA 端子の有効エッジを指定します。

[1]：FTIOB 端子の有効エッジを指定します。

配列の各要素には、次の定義値{ RAPI_RISING, RAPI_FALLING, RAPI_BOTH }をから 1 つを設定することができます。

使用しないインプットキャプチャに対応する要素には、0 を設定してください。

RAPI_RISING	測定パルスの有効エッジに立ち上がりエッジを選択する。
RAPI_FALLING	測定パルスの有効エッジに立ち下がりエッジを選択する。
RAPI_BOTH	測定パルスの有効エッジに立ち上がりと立ち下がりの両エッジを選択する。
RAPI_FILTER_ON	デジタルフィルタ機能ありを選択する。 RAPI_FILTER_ON を指定しない場合は、ディジタルフィルタ機能なし。

(H8/300H)

各有効エッジの設定を格納する配列のポインタを指定します。

[0]：FTIOA 端子の有効エッジを指定します。

[1]：FTIOB 端子の有効エッジを指定します。

[2]：FTIOC 端子の有効エッジを指定します。

[3]：FTIOD 端子の有効エッジを指定します。

配列の各要素には、次の定義値{ RAPI_RISING, RAPI_FALLING, RAPI_BOTH }をから 1 つを設定することができます。

使用しないインプットキャプチャに対応する要素には、0 を設定してください。

RAPI_RISING	測定パルスの有効エッジに立ち上がりエッジを選択する。
RAPI_FALLING	測定パルスの有効エッジに立ち下がりエッジを選択する。
RAPI_BOTH	測定パルスの有効エッジに立ち上がりと立ち下がりの両エッジを選択する。
RAPI_FILTER_ON	デジタルフィルタ機能ありを選択する。 RAPI_FILTER_ON を指定しない場合は、ディジタルフィルタ機能なし。

[data4]

(M16C)

タイマ S の各レジスタへの設定値を格納する配列のポインタを指定します。

[0]：機能選択レジスタおよび機能許可レジスタへの設定値を指定します。
　　時間計測機能を許可するチャネルを指定します。

[1]：割り込み許可レジスタ 0 への設定値を指定します。IO/CO 割り込み 0 要求を
　　許可するチャネルを指定します。

[2]：割り込み許可レジスタ 1 への設定値を指定します。IO/CO 割り込み 1 要求を
　　許可するチャネルを指定します。

[3]：カウントソース分周レジスタへの設定値を指定します。カウントソースの
　　(n+1)分周の n の値を 8 ビットで指定します。

[4]：時間計測プリスケーラレジスタ 6 への設定値を指定します。
　　プリスケーラ周期 n+1 の n の値を 8 ビットで指定します。

[5]：時間計測プリスケーラレジスタ 7 への設定値を指定します。
　　プリスケーラ周期 n+1 の n の値を 8 ビットで指定します。

配列の各要素で指定するチャネルには、次の定義値を使用します。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。0 を指定した場合は、対応する

レジスタへ 0 を設定します。

RAPI_CHANNEL0	チャネル 0 を選択する。
RAPI_CHANNEL1	チャネル 1 を選択する。
RAPI_CHANNEL2	チャネル 2 を選択する。
RAPI_CHANNEL3	チャネル 3 を選択する。
RAPI_CHANNEL4	チャネル 4 を選択する。
RAPI_CHANNEL5	チャネル 5 を選択する。
RAPI_CHANNEL6	チャネル 6 を選択する。
RAPI_CHANNEL7	チャネル 7 を選択する。

(R8C)(H8S)(H8/300H)

0 を指定します。

[data5]

(M16C)

コールバック関数を格納する配列のポインタを指定します。

[0]：IC/OC ベースタイマ割り込みのコールバック関数のポインタを指定します。
IC/OC ベースタイマ割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。

[1]：IC/OC 割り込み 0 のコールバック関数のポインタを指定します。
IC/OC 割り込み 0 のコールバック関数のポインタを指定しない場合は、0 を設定します。

[2]：IC/OC 割り込み 1 のコールバック関数のポインタを指定します。
IC/OC 割り込み 1 のコールバック関数のポインタを指定しない場合は、0 を設定します。

(R8C)

・タイマ C 使用時（RAPI_TIMER_C を指定時）、タイマ RC 使用時（RAPI_TIMER_RC を指定時）または、タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD1 を指定時）

コールバック関数を格納する変数のポインタを指定します。コールバック関数を格納する変数のポインタを指定しない場合は、RAPI_NULL を設定します。

・タイマ RF 使用時（RAPI_TIMER_RF を指定時）

コールバック関数を格納する配列のポインタを指定します。

[0]：タイマ RF 割り込みのコールバック関数のポインタを指定します。
タイマ RF 割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。

[1]：インプットキャプチャ割り込みのコールバック関数のポインタを指定します。
インプットキャプチャ割り込みのコールバック関数のポインタを指定しない場合は、0 を設定します。

(H8S)

コールバック関数を格納する配列のポインタを指定します。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）または、タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時）

[0]：カウンタオーバフロー割り込みのコールバック関数のポインタを指定します。
カウンタオーバフロー割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。

[1]：インプットキャプチャ A 割り込みのコールバック関数のポインタを指定します。
インプットキャプチャ A 割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。
[2]：インプットキャプチャ B 割り込みのコールバック関数のポインタを指定します。
インプットキャプチャ B 割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。
[3]：インプットキャプチャ C 割り込みのコールバック関数のポインタを指定します。
インプットキャプチャ C 割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。
[4]：インプットキャプチャ D 割り込みのコールバック関数のポインタを指定します。
インプットキャプチャ D 割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。

・タイマ RG 使用時（RAPI_TIMER_RG を指定時）

[0]：カウンタオーバフロー割り込みのコールバック関数のポインタを指定します。
カウンタオーバフロー割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。
[1]：インプットキャプチャ A 割り込みのコールバック関数のポインタを指定します。
インプットキャプチャ A 割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。
[2]：インプットキャプチャ B 割り込みのコールバック関数のポインタを指定します。
インプットキャプチャ B 割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。

(H8/300H)

コールバック関数を格納する変数のポインタを指定します。コールバック関数を格納する変数のポインタを指定しない場合は、RAPI_NULL を設定します。

戻り値　タイマまたはインテリジェント I/O 指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能　タイマ(インプットキャプチャモード)

参照　__EnableInputCapture, __DestroyInputCapture, __GetInputCapture

備考
- 第 1 引数、第 3 引数および、第 4 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できるタイマは CPU により異なります。
- デジタルフィルタ機能を有しない CPU を使用する場合、デジタルフィルタ機能に関する設定はできません。
- H8S および、H8/300H で使用する場合、指定したタイマのモジュールスタンバイ解除後に設定をおこないます。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void TimerIntFunc( void ){}
```

```
void func( void )
{
    /* Set up timer C as imput capture mode */
    __CreateInputCapture(
        RAPI_TIMER_C|RAPI_TIMER_ON|RAPI_BOTH|RAPI_F32| RAPI_FRING128,
        5, TimerIntFunc);
}
```

__EnableInputCapture

要約

< インプットキャプチャモード動作制御 >

Boolean __EnableInputCapture(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したインプットキャプチャモードに設定されているタイマまたはインテリジェント I/O の動作開始/動作停止を制御します。

[data]

data には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_S	タイマ S を選択する。
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する。
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する。

(R8C)

RAPI_TIMER_C	タイマ C を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RF	タイマ RF を選択する。
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する。
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する。

(H8S)

RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
RAPI_TIMER_RG	タイマ RG を選択する。
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する。
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する。

(H8/300H)

RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。

RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
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 動作開始に設定する。
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 動作停止に設定する。

戻り値　タイマまたはインテリジェント I/O 指定が正しくない場合は RAPI_FALSE を返し、
それ以外は RAPI_TRUE を返します。

機能　タイマ(インプットキャプチャモード)

参照　CreateInputCapture, DestroyInputCapture, GetInputCapture

備考
- 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できるタイマは CPU により異なります。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void func( void )
{
    /* Enable timer C as input capture mode */
    __EnableInputCapture( RAPI_TIMER_C|RAPI_TIMER_ON );
}
```

__DestroyInputCapture

要約

< インプットキャプチャモード設定破棄 >

Boolean __DestroyInputCapture(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したインプットキャプチャモードに設定されているタイマまたはインテリジェント I/O の設定を破棄します。

[data]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_S	タイマ S を選択する。

(R8C)

RAPI_TIMER_C	タイマ C を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RF	タイマ RF を選択する。

(H8S)

RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
RAPI_TIMER_RG	タイマ RG を選択する。

(H8/300H)

RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。

戻り値

タイマまたはインテリジェント I/O 指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(インプットキャプチャモード)

参照

__CreateInputCapture, __EnableInputCapture, __GetInputCapture

備考

・引数に定義値以外の値を指定した場合の動作は保証されません。
・指定できるタイマは CPU により異なります。
・H8S および、H8/300H で使用する場合、指定したタイマを、設定破棄後、
モジュールスタンバイ状態に設定します。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void func( void )
{
    /* Destroy the setting of timer C as input capture mode */
    __DestroyInputCapture( RAPI_TIMER_C );
}
```

__GetInputCapture

要約

< インプットキャプチャモードカウンタ値取得 >

Boolean __GetInputCapture(unsigned long data1, unsigned int *data2)

data1	設定データ 1（機種により内容が異なる）
data2	設定データ 2（機種により内容が異なる）

解説

指定したインプットキャプチャモードに設定されているタイマまたはインテリジェント I/O のカウンタ値を取得します。

[data1]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_S	タイマ S を選択する。

(R8C)

RAPI_TIMER_C	タイマ C を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RF	タイマ RF を選択する。

(H8S)

RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
RAPI_TIMER_RG	タイマ RG を選択する。

(H8/300H)

RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。

[data2]

(M16C)

取得したカウンタ値を格納する配列のポインタを指定します。

[0]：ベースタイマレジスタの値を格納します。

[1]：時間計測レジスタ 0 の値を格納します。

[2]：時間計測レジスタ 1 の値を格納します。

[3]：時間計測レジスタ 2 の値を格納します。
[4]：時間計測レジスタ 3 の値を格納します。
[5]：時間計測レジスタ 4 の値を格納します。
[6]：時間計測レジスタ 5 の値を格納します。
[7]：時間計測レジスタ 6 の値を格納します。
[8]：時間計測レジスタ 7 の値を格納します。

(R8C)

取得したカウンタ値を格納する配列のポインタを指定します。

・タイマ C 使用時（RAPI_TIMER_C を指定時）または、タイマ RF 使用時（RAPI_TIMER_RF を指定時）

[0]：タイマレジスタの値を格納します。
[1]：キャプチャ、コンペア 0 レジスタの値を格納します。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）または、タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD1 を指定時）

[0]：タイマカウンタの値を格納します。
[1]：ジェネラルレジスタ A の値を格納します。
[2]：ジェネラルレジスタ B の値を格納します。
[3]：ジェネラルレジスタ C の値を格納します。
[4]：ジェネラルレジスタ D の値を格納します。

(H8S)

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）または、タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時）

[0]：タイマカウンタの値を格納します。
[1]：ジェネラルレジスタ A の値を格納します。
[2]：ジェネラルレジスタ B の値を格納します。
[3]：ジェネラルレジスタ C の値を格納します。
[4]：ジェネラルレジスタ D の値を格納します。

・タイマ RG 使用時（RAPI_TIMER_RG を指定時）

[0]：タイマカウンタの値を格納します。
[1]：ジェネラルレジスタ A の値を格納します。
[2]：ジェネラルレジスタ B の値を格納します。

(H8/300H)

[0]：タイマカウンタの値を格納します。
[1]：ジェネラルレジスタ A の値を格納します。
[2]：ジェネラルレジスタ B の値を格納します。
[3]：ジェネラルレジスタ C の値を格納します。
[4]：ジェネラルレジスタ D の値を格納します。

戻り値

タイマまたはインテリジェント I/O 指定が正しくない場合は RAPI_FALSE を返し、
それ以外は RAPI_TRUE を返します。

機能

タイマ(インプットキャプチャモード)

参照

__CreateInputCapture, __EnableInputCapture, __DestroyInputCapture

備考

・第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・指定できるタイマは CPU により異なります。

プログラム例

```
#include "rapi_timer_r8c_13.h"

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

    /* Get the counter of timer C as input capture mode */
    __GetInputCapture( RAPI_TIMER_C, data );
}
```

__CreateOutputCompare

要約

< アウトプットコンペアモード設定 >

Boolean __CreateOutputCompare(unsigned long data1, unsigned int* data2, unsigned int* data3, unsigned int* data4, void data5)**

data1	設定データ 1（機種により内容が異なる）
data2	設定データ 2（機種により内容が異なる）
data3	設定データ 3（機種により内容が異なる）
data4	設定データ 4（機種により内容が異なる）
data5	設定データ 5（機種により内容が異なる）

解説

指定したタイマまたはインテリジェント I/O をアウトプットコンペアモードに設定します。

[data1]

data1 には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_S	タイマ S を使用する。
RAPI_F1	カウントソースに f_1 を選択する。
RAPI_F2	カウントソースに f_2 を選択する。
RAPI_TIMER_ON	__CreateOutputCompare 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateOutputCompare 内ではタイマを動作停止に設定する。
RAPI_OVERFLOW_BIT14	ビット 14 のオーバフローをベースタイマ割り込みに選択する。
RAPI_OVERFLOW_BIT15	ビット 15 のオーバフローをベースタイマ割り込みに選択する。

・タイマ S 使用時（RAPI_TIMER_S を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_F1, RAPI_F2 }から 1 つを指定します。省略値は RAPI_F2 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（ベースタイマ）　{ RAPI_OVERFLOW_BIT14, RAPI_OVERFLOW_BIT15 }から 1 つを指定します。
省略値は RAPI_OVERFLOW_BIT15 です。

(R8C)

RAPI_TIMER_C	タイマ C を使用する。
RAPI_TIMER_RC	タイマ RC を使用する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を使用する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を使用する。
RAPI_TIMER_RE	タイマ RE を使用する。
RAPI_TIMER_RF	タイマ RF を使用する。
RAPI_F1	カウントソースに f_1 を選択する。
RAPI_F8	カウントソースに f_8 を選択する。
RAPI_F32	カウントソースに f_{32} を選択する。
RAPI_FRING_FAST	カウントソースに $f_{RING-fast}$ を選択する。
RAPI_TRC_F1	タイマ RC はカウントソースに f_1 を選択する。
RAPI_TRC_F2	タイマ RC はカウントソースに f_2 を選択する。

RAPI_TRC_F4	タイマ RC はカウントソースに f_4 を選択する。
RAPI_TRC_F8	タイマ RC はカウントソースに f_8 を選択する。
RAPI_TRC_F32	タイマ RC はカウントソースに f_{32} を選択する。
RAPI_TRC_F40M	タイマ RC はカウントソースに fOCO40M を選択する。
RAPI_TRD_F1	タイマ RD はカウントソースに f_1 を選択する。
RAPI_TRD_F2	タイマ RD はカウントソースに f_2 を選択する。
RAPI_TRD_F4	タイマ RD はカウントソースに f_4 を選択する。
RAPI_TRD_F8	タイマ RD はカウントソースに f_8 を選択する。
RAPI_TRD_F32	タイマ RD はカウントソースに f_{32} を選択する。
RAPI_TRD_F40M	タイマ RD はカウントソースに fOCO40M を選択する。
RAPI_TRE_F4	タイマ RE はカウントソースに f_4 を選択する。
RAPI_TRE_F8	タイマ RE はカウントソースに f_8 を選択する。
RAPI_TRE_F32	タイマ RE はカウントソースに f_{32} を選択する。
RAPI_TRE_FC4	タイマ RD はカウントソースに fC4 を選択する。
RAPI_TRF_F1	タイマ RF はカウントソースに f_1 を選択する。
RAPI_TRF_F8	タイマ RF はカウントソースに f_8 を選択する。
RAPI_TRF_F32	タイマ RF はカウントソースに f_{32} を選択する。
RAPI_TIMER_ON	__CreateOutputCompare 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateOutputCompare 内ではタイマを動作停止に設定する。
RAPI_OVERFLOW	オーバフロー割り込みを許可する。
RAPI_COMPARE_MATCH	コンペアマッチ割り込みを許可する。
RAPI_COMPARE_ MATCH_A	GRA のコンペアマッチ割り込みを許可する。
RAPI_COMPARE_ MATCH_B	GRB のコンペアマッチ割り込みを許可する。
RAPI_COMPARE_ MATCH_C	GRC のコンペアマッチ割り込みを許可する。
RAPI_COMPARE_ MATCH_D	GRD のコンペアマッチ割り込みを許可する。
RAPI_COUNT_CLEAR_A	カウンタクリア要因に GRA のコンペアマッチを指定する。
RAPI_COUNT_CLEAR_B	カウンタクリア要因に GRB のコンペアマッチを指定する。
RAPI_COUNT_CLEAR_C	カウンタクリア要因に GRC のコンペアマッチを指定する。
RAPI_COUNT_CLEAR_D	カウンタクリア要因に GRD のコンペアマッチを指定する。
RAPI_COUNT_CLEAR_ SYNC	同期動作をしている他のチャネルのカウンタクリアに同期して カウンタをクリアする。
RAPI_COUNT_CLEAR _CMP1	コンペア 1 一致時に TRF レジスタを”0x0000”にする。
RAPI_TIMER_SYNC	チャネル 0 とチャネル 1 でタイマ同期する。
RAPI_CMPO0_DISABLE	$CMPO_0$ からの CMP 出力を禁止する。
RAPI_CMPO1_DISABLE	$CMPO_1$ からの CMP 出力を禁止する。
RAPI_CMPO2_DISABLE	$CMPO_2$ からの CMP 出力を禁止する。
RAPI_CMP10_DISABLE	$CMP1_0$ からの CMP 出力を禁止する。
RAPI_CMP11_DISABLE	$CMP1_1$ からの CMP 出力を禁止する。

RAPI_CMP12_DISABLE	$CMP1_2$からの CMP 出力を禁止する。
RAPI_CMP00_ENABLE	$CMP0_0$からの CMP 出力を許可する。
RAPI_CMP01_ENABLE	$CMP0_1$からの CMP 出力を許可する。
RAPI_CMP02_ENABLE	$CMP0_2$からの CMP 出力を許可する。
RAPI_CMP10_ENABLE	$CMP1_0$からの CMP 出力を許可する。
RAPI_CMP11_ENABLE	$CMP1_1$からの CMP 出力を許可する。
RAPI_CMP12_ENABLE	$CMP1_2$からの CMP 出力を許可する。
RAPI_TRF000_DISABLE	$TRF00_0$からの TRF0 出力を禁止する。
RAPI_TRF001_DISABLE	$TRF00_1$からの TRF0 出力を禁止する。
RAPI_TRF002_DISABLE	$TRF00_2$からの TRF0 出力を禁止する。
RAPI_TRF010_DISABLE	$TRF01_0$からの TRF0 出力を禁止する。
RAPI_TRF011_DISABLE	$TRF01_1$からの TRF0 出力を禁止する。
RAPI_TRF012_DISABLE	$TRF01_2$からの TRF0 出力を禁止する。
RAPI_TRF000_ENABLE	$TRF00_0$からの TRF0 出力を許可する。
RAPI_TRF001_ENABLE	$TRF00_1$からの TRF0 出力を許可する。
RAPI_TRF002_ENABLE	$TRF00_2$からの TRF0 出力を許可する。
RAPI_TRF010_ENABLE	$TRF01_0$からの TRF0 出力を許可する。
RAPI_TRF011_ENABLE	$TRF01_1$からの TRF0 出力を許可する。
RAPI_TRF012_ENABLE	$TRF01_2$からの TRF0 出力を許可する。
RAPI_OUTPUT_ REVERSE_0	タイマ C は $CMP0_0$～$CMP0_2$からの CMP 出力を反転する。 タイマ RF は $TRF00_0$～$TRF00_2$からの TRF0 出力を反転する。
RAPI_OUTPUT_ REVERSE_1	タイマ C は $CMP1_0$～$CMP1_2$からの CMP 出力を反転する。 タイマ RF は $TRF01_0$～$TRF01_2$からの TRF0 出力を反転する。
RAPI_RELOAD	コンペア 1 一致時に TC レジスタを"0x0000"にする。
RAPI_UNCHANGE_0	タイマ C はコンペア 0 で一致しても CMP0 出力は変化しない タイマ RF はコンペア 0 で一致しても CMP0 出力は変化しない
RAPI_REVERSE_0	タイマ C はコンペア 0 の一致信号で CMP0 出力を反転する。 タイマ RF はコンペア 0 の一致信号で TRF00 出力を反転する
RAPI_L_0	タイマ C はコンペア 0 の一致信号で CMP0 出力を"L"に設定する。 タイマ RF はコンペア 0 の一致信号で TRF00 出力を"L"に設定する。
RAPI_H_0	タイマ C はコンペア 0 の一致信号で CMP0 出力を"H"に設定する。 タイマ RF はコンペア 0 の一致信号で TRF00 出力を"H"に設定する。
RAPI_UNCHANGE_1	タイマ C はコンペア 1 で一致しても CMP1 出力は変化しない。 タイマ RF はコンペア 1 で一致しても TRF01 出力は変化しない。
RAPI_REVERSE_1	タイマ C はコンペア 1 の一致信号で CMP1 出力を反転する。 タイマ RF はコンペア 1 の一致信号で TRF01 出力を反転する。
RAPI_L_1	タイマ C はコンペア 1 の一致信号で CMP1 出力を"L"に設定する。 タイマ RF はコンペア 1 の一致信号で TRF01 出力を"L"に設定する。
RAPI_H_1	タイマ C はコンペア 1 の一致信号で CMP1 出力を"H"に設定する。 タイマ RF はコンペア 1 の一致信号で TRF01 出力を"H"に設定する。
RAPI_OUTPUT_DISABLE	出力機能を出力禁止に設定する。
RAPI_OUTPUT_F2	出力機能を f_2出力に設定する。
RAPI_OUTPUT_F4	出力機能を f_4出力に設定する。

RAPI_OUTPUT_F8	出力機能を f_8 出力に設定する。
RAPI_OUTPUT_COMPARE	出力機能をコンペア出力に設定する。
RAPI_4BIT_COUNTER	4bit カウンタを使用する。
RAPI_COMPARE_MATCH_A_STOP	GRA のコンペアマッチでカウント停止します。
RAPI_STOP	TSTART ビットへの"0"書き込みでカウントを停止します。
RAPI_STOP_OUTPUT	カウント停止時の出力レベルをアウトプットにする。
RAPI_STOP_H	カウント停止時の出力レベルを H にする。
RAPI_STOP_L	カウント停止時の出力レベルを L にする。

・タイマ C 使用時（RAPI_TIMER_C を指定時）に指定可能な定義値

(カウントソース)　{ RAPI_F1, RAPI_F8, RAPI_F32, RAPI_FRING_FAST }から 1 つを指定します。省略値は RAPI_F1 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

(CMPO$_0$出力)　{ RAPI_CMPO0_DISABLE, RAPI_CMPO0_ENABLE }から 1 つを指定します。省略値は RAPI_CMPO0_DISABLE です。

(CMPO$_1$出力)　{ RAPI_CMPO1_DISABLE, RAPI_CMPO1_ENABLE }から 1 つを指定します。省略値は RAPI_CMPO1_DISABLE です。

(CMPO$_0$出力)　{ RAPI_CMPO2_DISABLE, RAPI_CMPO2_ENABLE }から 1 つを指定します。省略値は RAPI_CMPO2_DISABLE です。

(CMP1$_0$出力)　{ RAPI_CMP10_DISABLE, RAPI_CMP10_ENABLE }から 1 つを指定します。省略値は RAPI_CMP10_DISABLE です。

(CMP1$_1$出力)　{ RAPI_CMP11_DISABLE, RAPI_CMP11_ENABLE }から 1 つを指定します。省略値は RAPI_CMP11_DISABLE です。

(CMP1$_0$出力)　{ RAPI_CMP12_DISABLE, RAPI_CMP12_ENABLE }から 1 つを指定します。省略値は RAPI_CMP12_DISABLE です。

(CMPO 出力反転)　CMPO 出力を反転する場合、RAPI_OUTPUT_REVERSE_0 を指定します。RAPI_OUTPUT_REVERSE_0 を指定しない場合は、「CMPO 出力反転しない」に設定します。

(CMP1 出力反転)　CMP1 出力を反転する場合、RAPI_OUTPUT_REVERSE_1 を指定します。RAPI_OUTPUT_REVERSE_1 を指定しない場合は、「CMP1 出力反転しない」に設定します。

(TC リロード)　TC レジスタをリロードする場合、RAPI_RELOAD を指定します。RAPI_RELOAD を指定しない場合は、「リロードなし」に設定します。

(CMPO 出力モード)　{ RAPI_UNCHANGE_0, RAPI_REVERSE_0, RAPI_L_0, RAPI_H_0 }から 1 つを指定します。省略値は RAPI_UNCHANGE_0 です。

(CMP1 出力モード)　{ RAPI_UNCHANGE_1, RAPI_REVERSE_1, RAPI_L_1, RAPI_H_1 }から 1 つを指定します。省略値は RAPI_UNCHANGE_0 です。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）に指定可能な定義値

(カウントソース)　{ RAPI_TRC_F1, RAPI_TRC_F2, RAPI_TRC_F4, RAPI_TRC_F8, RAPI_TRC_F32, RAPI_TRC_F40M }から 1 つを指定します。省略値は RAPI_TRC_F1 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

(割り込み)　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を、

コンペアマッチ A 割り込み要求ありの場合は RAPI_COMPARE_MATCH_A を、
コンペアマッチ B 割り込み要求ありの場合は RAPI_COMPARE_MATCH_B を、
コンペアマッチ C 割り込み要求ありの場合は RAPI_COMPARE_MATCH_C を、
コンペアマッチ D 割り込み要求ありの場合は RAPI_COMPARE_MATCH_D を、
指定します。割り込みを指定しない場合は、「割り込み要求なし」に
設定します。

(カウンタクリア)　カウンタクリア要因に GRA のコンペアマッチを指定する場合は、
RAPI_COUNT_CLEAR_A を指定します。

・タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD1 を指定時）に指定可能な定義値

(カウントソース)　{ RAPI_TRD_F1, RAPI_TRD_F2, RAPI_TRD_F4, RAPI_TRD_F8, RAPI_TRD_F32,
RAPI_TRD_F40M }から 1 つを指定します。省略値は RAPI_TRD_F1 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

(割り込み)　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を、
コンペアマッチ A 割り込み要求ありの場合は RAPI_COMPARE_MATCH_A を、
コンペアマッチ B 割り込み要求ありの場合は RAPI_COMPARE_MATCH_B を、
コンペアマッチ C 割り込み要求ありの場合は RAPI_COMPARE_MATCH_C を、
コンペアマッチ D 割り込み要求ありの場合は RAPI_COMPARE_MATCH_D を、
指定します。割り込みを指定しない場合は、「割り込み要求なし」に
設定します。

(カウンタクリア)　カウンタクリア要因に GRA のコンペアマッチを指定する場合は、
RAPI_COUNT_CLEAR_A を、GRB のコンペアマッチを指定する場合は、
RAPI_COUNT_CLEAR_B を、GRC のコンペアマッチを指定する場合は、
RAPI_COUNT_CLEAR_C を、GRD のコンペアマッチを指定する場合は、
RAPI_COUNT_CLEAR_D を、同期動作をしている他のチャネルのカウンタ
クリアに同期する場合は、RAPI_COUNT_CLEAR_SYNC を指定します。

(同期)　チャネル 0 とチャネル 1 でタイマ同期する場合は、RAPI_TIMER_SYNC を
指定します。同期を指定しない場合は、「チャネル 0 とチャネル 1 は
独立動作」を設定します。

(カウント停止)　{ RAPI_COMPARE_MATCH_A_STOP, RAPI_STOP }から 1 つを指定します。
省略値は RAPI_STOP です。

・タイマ RE 使用時（RAPI_TIMER_RE を指定時）に指定可能な定義値

(カウントソース)　{ RAPI_TRE_F4, RAPI_TRE_F8, RAPI_TRE_F32, RAPI_TRE_FC4 }から 1 つを
指定します。省略値は RAPI_TRE_F4 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

(割り込み)　コンペアマッチ割り込み要求ありの場合は RAPI_COMPARE_MATCH を
指定します。割り込みを指定しない場合は、「割り込み要求なし」に
設定します。

(出力機能)　{ RAPI_OUTPUT_DISABLE, RAPI_OUTPUT_F2, RAPI_OUTPUT_F4,
RAPI_OUTPUT_F8, RAPI_OUTPUT_COMPARE }から 1 つを指定します。
省略値は RAPI_OUTPUT_DISABLE です。
R8C/28,29 では指定できません。

(カウンタ)　4bit カウンタを使用する場合は、RAPI_4BIT_COUNTER を指定します。

RAPI_4BIT_COUNTER を指定しない場合は、4bit カウンタを使用せず、
8bit カウンタのみで動作します。

・タイマ RF 使用時（RAPI_TIMER_RF を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRF_F1, RAPI_TRF_F8, RAPI_TRF_F32, }から 1 つを指定します。
省略値は RAPI_TRF_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

（$TRFO0_0$ 出力）　{ RAPI_TRFO00_DISABLE, RAPI_TRFO00_ENABLE }から 1 つを指定します。
省略値は RAPI_TRFO00_DISABLE です。

（$TRFO0_1$ 出力）　{ RAPI_TRFO01_DISABLE, RAPI_TRFO01_ENABLE }から 1 つを指定します。
省略値は RAPI_TRFO01_DISABLE です。

（$TRFO0_0$ 出力）　{ RAPI_TRFO02_DISABLE, RAPI_TRFO02_ENABLE }から 1 つを指定します。
省略値は RAPI_TRFO02_DISABLE です。

（$TRFO1_0$ 出力）　{ RAPI_TRFO10_DISABLE, RAPI_TRFO10_ENABLE }から 1 つを指定します。
省略値は RAPI_TRFO10_DISABLE です。

（$TRFO1_1$ 出力）　{ RAPI_TRFO11_DISABLE, RAPI_TRFO11_ENABLE }から 1 つを指定します。
省略値は RAPI_TRFO11_DISABLE です。

（$TRFO1_0$ 出力）　{ RAPI_TRFO12_DISABLE, RAPI_TRFO12_ENABLE }から 1 つを指定します。
省略値は RAPI_TRFO12_DISABLE です。

（TRFO0 出力反転）　TRFO0 出力を反転する場合、RAPI_OUTPUT_REVERSE_0 を指定します。
RAPI_OUTPUT_REVERSE_0 を指定しない場合は、「TRFO0 出力反転しない」に
設定します。

（TRFO1 出力反転）　TRFO1 出力を反転する場合、RAPI_OUTPUT_REVERSE_1 を指定します。
RAPI_OUTPUT_REVERSE_1 を指定しない場合は、「TRFO1 出力反転しない」に
設定します。

（カウントクリア）　コンペアマッチ 1 でカウント値をクリアする場合、RAPI_COUNT_CLEAR_CMP1
を指定します。RAPI_COUNT_CLEAR_CMP1 を指定しない場合は、
「クリアなし」に設定します。

（TRFO0 出力モード）　{ RAPI_UNCHANGE_0, RAPI_REVERSE_0, RAPI_L_0, RAPI_H_0 }から 1 つを
指定します。省略値は RAPI_UNCHANGE_0 です。

（TRFO1 出力モード）　{ RAPI_UNCHANGE_1, RAPI_REVERSE_1, RAPI_L_1, RAPI_H_1 }から 1 つを
指定します。省略値は RAPI_UNCHANGE_0 です。

（カウント停止時出力）　{ RAPI_STOP_OUTPUT, RAPI_STOP_H, RAPI_STOP_L }から 1 つを
指定します。省略値は RAPI_STOP_OUTPUT です。

(H8S)

RAPI_TIMER_RC	タイマ RC を使用する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を使用する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を使用する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を使用する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を使用する。
RAPI_TIMER_RE	タイマ RE を使用する。
RAPI_TIMER_RG	タイマ RG を使用する。
RAPI_TRC_F1	タイマ RC は内部クロックφでカウントする。
RAPI_TRC_F2	タイマ RC は内部クロックφ/2 でカウントする。

RAPI_TRC_F4	タイマ RC は内部クロックφ/4 でカウントする。
RAPI_TRC_F8	タイマ RC は内部クロックφ/8 でカウントする。
RAPI_TRC_F32	タイマ RC は内部クロックφ/32 でカウントする。
RAPI_TRC_F40	タイマ RC は内部クロックφ40 でカウントする。
RAPI_TRD_F1	タイマ RD は内部クロックφでカウントする。
RAPI_TRD_F2	タイマ RD は内部クロックφ/2 でカウントする。
RAPI_TRD_F4	タイマ RD は内部クロックφ/4 でカウントする。
RAPI_TRD_F8	タイマ RD は内部クロックφ/8 でカウントする。
RAPI_TRD_F32	タイマ RD は内部クロックφ/32 でカウントする。
RAPI_TRD_F40	タイマ RD は内部クロックφ40 でカウントする。
RAPI_TRE_F4	タイマ RE はカウントソースに f_4 を選択する。
RAPI_TRE_F8	タイマ RE はカウントソースに f_8 を選択する。
RAPI_TRE_F32	タイマ RE はカウントソースに f_{32} を選択する。
RAPI_TRE_FSUB	タイマ RE のカウントソースにサブ発信器出力クロックを選択する。
RAPI_TRG_F1	タイマ RG は内部クロックφでカウントする。
RAPI_TRG_F2	タイマ RG は内部クロックφ/2 でカウントする。
RAPI_TRG_F4	タイマ RG は内部クロックφ/4 でカウントする。
RAPI_TRG_F8	タイマ RG は内部クロックφ/8 でカウントする。
RAPI_TRG_F32	タイマ RG は内部クロックφ32 でカウントする。
RAPI_TRG_F40	タイマ RG は内部クロックφ40 でカウントする。
RAPI_TIMER_ON	__CreateOutputCompare 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateOutputCompare 内ではタイマを動作停止に設定する。
RAPI_OVERFLOW	オーバフロー割り込みを許可する。
RAPI_COMPARE_MATCH	コンペアマッチ割り込みを許可する。
RAPI_COMPARE_ MATCH_A	GRA のコンペアマッチ割り込みを許可する。
RAPI_COMPARE_ MATCH_B	GRB のコンペアマッチ割り込みを許可する。
RAPI_COMPARE_ MATCH_C	GRC のコンペアマッチ割り込みを許可する。
RAPI_COMPARE_ MATCH_D	GRD のコンペアマッチ割り込みを許可する。
RAPI_INT_MODE_0	割り込み制御モード 0 を指定する。
RAPI_INT_MODE_2	割り込み制御モード 2 を指定する。
RAPI_TRC_COUNT_ CLEAR_A	タイマ RC のカウンタクリア要因に GRA のコンペアマッチを指定する。
RAPI_TRD_COUNT_ CLEAR_A	タイマ RD のカウンタクリア要因に GRA のコンペアマッチを指定する。
RAPI_TRD_COUNT _CLEAR_B	タイマ RD のカウンタクリア要因に GRB のコンペアマッチを指定する。
RAPI_TRD_COUNT _CLEAR_C	タイマ RD のカウンタクリア要因に GRC のコンペアマッチを指定する。

RAPI_TRD_COUNT_CLEAR_D	タイマ RD のカウンタクリア要因に GRD のコンペアマッチを指定する。
RAPI_TRD_COUNT_CLEAR_SYNC	タイマ RD の同期動作をしている他のチャネルのカウンタクリアに同期してカウンタをクリアする。
RAPI_TRG_COUNT_CLEAR_A	タイマ RG のカウンタクリア要因に GRA のコンペアマッチを指定する。
RAPI_TRG_COUNT_CLEAR_B	タイマ RG のカウンタクリア要因に GRB のコンペアマッチを指定する。
RAPI_TIMER_SYNC	チャネル 0 とチャネル 1 でタイマ同期する。
RAPI_OUTPUT_DISABLE	出力機能を出力禁止に設定する。
RAPI_OUTPUT_F2	出力機能をφ/2 出力に設定する。
RAPI_OUTPUT_F4	出力機能をφ/4 出力に設定する。
RAPI_OUTPUT_F8	出力機能をφ/8 出力に設定する。
RAPI_OUTPUT_COMPARE	出力機能をコンペア出力に設定する。
RAPI_4BIT_COUNTER	4bit カウンタを使用する。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRC_F1, RAPI_TRC_F2, RAPI_TRC_F4, RAPI_TRC_F8, RAPI_TRC_F32, RAPI_TRC_F40 }から 1 つを指定します。省略値は RAPI_TRC_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、コンペアマッチ A 割り込み要求ありの場合は RAPI_COMPARE_MATCH_A を、コンペアマッチ B 割り込み要求ありの場合は RAPI_COMPARE_MATCH_B を、コンペアマッチ C 割り込み要求ありの場合は RAPI_COMPARE_MATCH_C を、コンペアマッチ D 割り込み要求ありの場合は RAPI_COMPARE_MATCH_D を、指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（割り込み制御モード）　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。省略値は RAPI_INT_MODE_0 です。

（カウンタクリア）　カウンタクリア要因に GRA のコンペアマッチを指定する場合は、RAPI_TRC_COUNT_CLEAR_A を指定します。

・タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRD_F1, RAPI_TRD_F2, RAPI_TRD_F4, RAPI_TRD_F8, RAPI_TRD_F32, RAPI_TRD_F40 }から 1 つを指定します。省略値は RAPI_TRD_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を、コンペアマッチ A 割り込み要求ありの場合は RAPI_COMPARE_MATCH_A を、コンペアマッチ B 割り込み要求ありの場合は RAPI_COMPARE_MATCH_B を、コンペアマッチ C 割り込み要求ありの場合は RAPI_COMPARE_MATCH_C を、コンペアマッチ D 割り込み要求ありの場合は RAPI_COMPARE_MATCH_D を、指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（割り込み　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。

制御モード)　省略値は RAPI_INT_MODE_0 です。

(カウンタクリア)　カウンタクリア要因に GRA のコンペアマッチを指定する場合は、RAPI_TRD_COUNT_CLEAR_A を、GRB のコンペアマッチを指定する場合は、RAPI_TRD_COUNT_CLEAR_B を、GRC のコンペアマッチを指定する場合は、RAPI_TRD_COUNT_CLEAR_C を、GRD のコンペアマッチを指定する場合は、RAPI_TRD_COUNT_CLEAR_D を、同期動作をしている他のチャネルのカウンタクリアに同期する場合は、RAPI_TRD_COUNT_CLEAR_SYNC を指定します。

(同期)　チャネル 0 とチャネル 1 でタイマ同期する場合は、RAPI_TIMER_SYNC を指定します。同期を指定しない場合は、「チャネル 0 とチャネル 1 は独立動作」を設定します。

・タイマ RE 使用時 (RAPI_TIMER_RE を指定時) に指定可能な定義値

(カウントソース)　{ RAPI_TRE_F4, RAPI_TRE_F8, RAPI_TRE_F32, RAPI_TRE_FSUB }から 1 つを指定します。省略値は RAPI_TRE_F4 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

(割り込み)　コンペアマッチ割り込み要求ありの場合は RAPI_COMPARE_MATCH を指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

(割り込み　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。

制御モード)　省略値は RAPI_INT_MODE_0 です。

(出力機能)　{ RAPI_OUTPUT_DISABLE, RAPI_OUTPUT_F2, RAPI_OUTPUT_F4, RAPI_OUTPUT_F8, RAPI_OUTPUT_COMPARE }から 1 つを指定します。省略値は RAPI_OUTPUT_DISABLE です。

(カウンタ)　4bit カウンタを使用する場合は、RAPI_4BIT_COUNTER を指定します。RAPI_4BIT_COUNTER を指定しない場合は、4bit カウンタを使用せず、8bit カウンタのみで動作します。

・タイマ RG 使用時 (RAPI_TIMER_RG を指定時) に指定可能な定義値

(カウントソース)　{ RAPI_TRG_F1, RAPI_TRG_F2, RAPI_TRG_F4, RAPI_TRG_F8, RAPI_TRG_F32, RAPI_TRG_F40 }から 1 つを指定します。省略値は RAPI_TRG_F1 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

(割り込み)　オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、コンペアマッチ A 割り込み要求ありの場合は RAPI_COMPARE_MATCH_A を、コンペアマッチ B 割り込み要求ありの場合は RAPI_COMPARE_MATCH_B を、指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

(割り込み　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。

制御モード)　省略値は RAPI_INT_MODE_0 です。

(カウンタクリア)　カウンタクリア要因に GRA のインプットキャプチャを指定する場合は、RAPI_TRG_COUNT_CLEAR_A を、GRB のインプットキャプチャを指定する場合は、RAPI_TRG_COUNT_CLEAR_B を指定します。

(H8/300H)

RAPI_TIMER_W	タイマ W を使用する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を使用する。

RAPI_TIMER_Z1	タイマ Z チャネル 1 を使用する。
RAPI_TIMER_RC	タイマ RC を使用する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を使用する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を使用する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を使用する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を使用する。
RAPI_TW_F1	タイマ W は内部クロックφでカウントする。
RAPI_TW_F2	タイマ W は内部クロックφ/2 でカウントする。
RAPI_TW_F4	タイマ W は内部クロックφ/4 でカウントする。
RAPI_TW_F8	タイマ W は内部クロックφ/8 でカウントする。
RAPI_TZ_F1	タイマ Z は内部クロックφでカウントする。
RAPI_TZ_F2	タイマ Z は内部クロックφ/2 でカウントする。
RAPI_TZ_F4	タイマ Z は内部クロックφ/4 でカウントする。
RAPI_TZ_F8	タイマ Z は内部クロックφ/8 でカウントする。
RAPI_TRC_F1	タイマ RC は内部クロックφでカウントする。
RAPI_TRC_F2	タイマ RC は内部クロックφ/2 でカウントする。
RAPI_TRC_F4	タイマ RC は内部クロックφ/4 でカウントする。
RAPI_TRC_F8	タイマ RC は内部クロックφ/8 でカウントする。
RAPI_TRC_F32	タイマ RC は内部クロックφ/32 でカウントする。
RAPI_TRC_F40M	タイマ RC は内部クロックφ40M でカウントする。
RAPI_TRD_F1	タイマ RD は内部クロックφでカウントする。
RAPI_TRD_F2	タイマ RD は内部クロックφ/2 でカウントする。
RAPI_TRD_F4	タイマ RD は内部クロックφ/4 でカウントする。
RAPI_TRD_F8	タイマ RD は内部クロックφ/8 でカウントする。
RAPI_TRD_F32	タイマ RD は内部クロックφ/32 でカウントする。
RAPI_TRD_F40M	タイマ RD は内部クロックφ40M でカウントする。
RAPI_TIMER_ON	__CreateOutputCompare 内でタイマを動作開始に設定する。
RAPI_TIMER_OFF	__CreateOutputCompare 内ではタイマを動作停止に設定する。
RAPI_OVERFLOW	オーバフロー割り込みを許可する。
RAPI_COMPARE_MATCH_A	GRA のコンペアマッチ割り込みを許可する。
RAPI_COMPARE_MATCH_B	GRB のコンペアマッチ割り込みを許可する。
RAPI_COMPARE_MATCH_C	GRC のコンペアマッチ割り込みを許可する。
RAPI_COMPARE_MATCH_D	GRD のコンペアマッチ割り込みを許可する。
RAPI_COUNT_CLEAR_A	カウンタクリア要因に GRA のコンペアマッチを指定する。
RAPI_COUNT_CLEAR_B	カウンタクリア要因に GRB のコンペアマッチを指定する。
RAPI_COUNT_CLEAR_C	カウンタクリア要因に GRC のコンペアマッチを指定する。
RAPI_COUNT_CLEAR_D	カウンタクリア要因に GRD のコンペアマッチを指定する。
RAPI_COUNT_CLEAR_SYNC	同期動作をしている他のチャネルのカウンタクリアに同期してカウンタをクリアする。

RAPI_TIMER_SYNC	チャネル 0 とチャネル 1 でタイマ同期する。

・タイマ W 使用時（RAPI_TIMER_W を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TW_F1, RAPI_TW_F2, RAPI_TW_F4, RAPI_TW_F8 }から 1 つを指定します。省略値は RAPI_TW_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、コンペアマッチ A 割り込み要求ありの場合は RAPI_COMPARE_MATCH_A を、コンペアマッチ B 割り込み要求ありの場合は RAPI_COMPARE_MATCH_B を、コンペアマッチ C 割り込み要求ありの場合は RAPI_COMPARE_MATCH_C を、コンペアマッチ D 割り込み要求ありの場合は RAPI_COMPARE_MATCH_D を、指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（カウンタクリア）　カウンタクリア要因に GRA のコンペアマッチを指定する場合は、RAPI_COUNT_CLEAR_A を指定します。

・タイマ Z 使用時（RAPI_TIMER_Z0～RAPI_TIMER_Z1 を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TZ_F1, RAPI_TZ_F2, RAPI_TZ_F4, RAPI_TZ_F8 }から 1 つを指定します。省略値は RAPI_TZ_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を、コンペアマッチ A 割り込み要求ありの場合は RAPI_COMPARE_MATCH_A を、コンペアマッチ B 割り込み要求ありの場合は RAPI_COMPARE_MATCH_B を、コンペアマッチ C 割り込み要求ありの場合は RAPI_COMPARE_MATCH_C を、コンペアマッチ D 割り込み要求ありの場合は RAPI_COMPARE_MATCH_D を、指定します。割り込みを指定しない場合は、「割り込み要求なし」に設定します。

（カウンタクリア）　カウンタクリア要因に GRA のコンペアマッチを指定する場合は、RAPI_COUNT_CLEAR_A を、GRB のコンペアマッチを指定する場合は、RAPI_COUNT_CLEAR_B を、GRC のコンペアマッチを指定する場合は、RAPI_COUNT_CLEAR_C を、GRD のコンペアマッチを指定する場合は、RAPI_COUNT_CLEAR_D を、同期動作をしている他のチャネルのカウンタクリアに同期する場合は、RAPI_COUNT_CLEAR_SYNC を指定します。

（同期）　チャネル 0 とチャネル 1 でタイマ同期する場合は、RAPI_TIMER_SYNC を指定します。同期を指定しない場合は、「チャネル 0 とチャネル 1 は独立動作」を設定します。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）に指定可能な定義値

（カウントソース）　{ RAPI_TRC_F1, RAPI_TRC_F2, RAPI_TRC_F4, RAPI_TRC_F8, RAPI_TRC_F32, RAPI_TRC_F40M }から 1 つを指定します。省略値は RAPI_TRC_F1 です。

（設定時動作状態）　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。省略値は RAPI_TIMER_OFF です。

（割り込み）　オーバーフロー割り込み要求ありの場合は RAPI_OVERFLOW を、コンペアマッチ A 割り込み要求ありの場合は RAPI_COMPARE_MATCH_A を、コンペアマッチ B 割り込み要求ありの場合は RAPI_COMPARE_MATCH_B を、

コンペアマッチ C 割り込み要求ありの場合は RAPI_COMPARE_MATCH_C を、
コンペアマッチ D 割り込み要求ありの場合は RAPI_COMPARE_MATCH_D を、
指定します。割り込みを指定しない場合は、「割り込み要求なし」に
設定します。

(カウンタクリア)　カウンタクリア要因に GRA のコンペアマッチを指定する場合は、
RAPI_COUNT_CLEAR_A を指定します。

・タイマ RD 使用時 (RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時) に指定可能な定義値

(カウントソース)　{ RAPI_TRD_F1, RAPI_TRD_F2, RAPI_TRD_F4, RAPI_TRD_F8, RAPI_TRD_F32,
RAPI_TRD_F40M }から 1 つを指定します。省略値は RAPI_TRD_F1 です。

(設定時動作状態)　{ RAPI_TIMER_ON, RAPI_TIMER_OFF }から 1 つを指定します。
省略値は RAPI_TIMER_OFF です。

(割り込み)　オーバーフロー割り込み要求ありの場合、RAPI_OVERFLOW を、
コンペアマッチ A 割り込み要求ありの場合は RAPI_COMPARE_MATCH_A を、
コンペアマッチ B 割り込み要求ありの場合は RAPI_COMPARE_MATCH_B を、
コンペアマッチ C 割り込み要求ありの場合は RAPI_COMPARE_MATCH_C を、
コンペアマッチ D 割り込み要求ありの場合は RAPI_COMPARE_MATCH_D を、
指定します。割り込みを指定しない場合は、「割り込み要求なし」に
設定します。

(カウンタクリア)　カウンタクリア要因に GRA のコンペアマッチを指定する場合は、
RAPI_COUNT_CLEAR_A を、GRB のコンペアマッチを指定する場合は、
RAPI_COUNT_CLEAR_B を、GRC のコンペアマッチを指定する場合は、
RAPI_COUNT_CLEAR_C を、GRD のコンペアマッチを指定する場合は、
RAPI_COUNT_CLEAR_D を、同期動作をしている他のチャネルのカウンタ
クリアに同期する場合は、RAPI_COUNT_CLEAR_SYNC を指定します。

(同期)　チャネル 0 とチャネル 1 でタイマ同期する場合は、RAPI_TIMER_SYNC を
指定します。同期を指定しない場合は、「チャネル 0 とチャネル 1 は
独立動作」を設定します。

[data2]

(M16C)

割り込みレベルを格納する配列のポインタを指定します。
[0]：IC/OC ベースタイマ割り込みのレベル(0～7)を指定します。
[1]：IC/OC 割り込み 0 のレベル(0～7)を指定します。
[2]：IC/OC 割り込み 1 のレベル(0～7)を指定します。

(R8C)

・タイマ C 使用時 (RAPI_TIMER_C を指定時) または、タイマ RF 使用時 (RAPI_TIMER_RF を指定時)

割り込みレベルを格納する配列のポインタを指定します。
[0]：タイマ割り込みのレベル(0～7)を指定します。
[1]：コンペアマッチ 0 割り込みのレベル(0～7)を指定します。
[2]：コンペアマッチ 1 割り込みのレベル(0～7)を指定します。

**・タイマ RC 使用時 (RAPI_TIMER_RC を指定時) 、タイマ RD 使用時 (RAPI_TIMER_RD0～
RAPI_TIMER_RD1 を指定時)または、タイマ RE 使用時 (RAPI_TIMER_RE を指定時)**

割り込み制御レジスタへ設定する割り込みレベル(0～7)を格納した変数のポインタを
指定します。

(H8S)

インタラプトプライオリティレジスタへ設定する割り込み優先レベル(0～3)を
格納した変数のポインタを指定します。

(H8/300H)

割り込みコントロールレジスタへ設定する割り込みプライオリティレベル(0～1)を
格納する変数のポインタを指定します。
割り込みコントロールレジスタを持たない CPU の場合は 0 を指定します。

[data3]

(M16C)

波形生成制御レジスタおよび波形生成レジスタへの設定値を格納する配列のポインタを
指定します。
[0]：波形生成制御レジスタ 0 への設定値を指定します。
[1]：波形生成制御レジスタ 1 への設定値を指定します。
[2]：波形生成制御レジスタ 2 への設定値を指定します。
[3]：波形生成制御レジスタ 3 への設定値を指定します。
[4]：波形生成制御レジスタ 4 への設定値を指定します。
[5]：波形生成制御レジスタ 5 への設定値を指定します。
[6]：波形生成制御レジスタ 6 への設定値を指定します。
[7]：波形生成制御レジスタ 7 への設定値を指定します。
[8]：波形生成レジスタ 0 への設定値を 16 ビットで指定します。
チャネル 0 の比較値を 16 ビットで指定します。
[9]：波形生成レジスタ 1 への設定値を 16 ビットで指定します。
チャネル 1 の比較値を 16 ビットで指定します。
[10]：波形生成レジスタ 2 への設定値を 16 ビットで指定します。
チャネル 2 の比較値を 16 ビットで指定します。
[11]：波形生成レジスタ 3 への設定値を 16 ビットで指定します。
チャネル 3 の比較値を 16 ビットで指定します。
[12]：波形生成レジスタ 4 への設定値を 16 ビットで指定します。
チャネル 4 の比較値を 16 ビットで指定します。
[13]：波形生成レジスタ 5 への設定値を 16 ビットで指定します。
チャネル 5 の比較値を 16 ビットで指定します。
[14]：波形生成レジスタ 6 への設定値を 16 ビットで指定します。
チャネル 6 の比較値を 16 ビットで指定します。
[15]：波形生成レジスタ 7 への設定値を 16 ビットで指定します。
チャネル 7 の比較値を 16 ビットで指定します。
配列の要素の各波形生成制御レジスタへの設定値には、次の定義値を設定することが
できます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

RAPI_SINGLE	単相波形出力モードを選択する。
RAPI_SR	SR 波形出力モードを選択します。
RAPI_PHASE_DELAYED	反転波形出力モードを選択する。
RAPI_OUT_INIT_0	出力初期値として"0"を出力する。
RAPI_OUT_INIT_1	出力初期値として"1"を出力する。
RAPI_OUTPUT_ REVERSED	出力反転機能を使用する。

RAPI_RELOAD_WITH_RESET	G1POj レジスタ値リロードのタイミングとして、ベースタイマのリセット時のリロードを選択する。
RAPI_RELOAD_WITH_WRITE	G1POj レジスタ値リロードのタイミングとして、書き込み時のリロードを選択する。

・波形生成制御レジスタ 0～7 に指定可能な定義値

(出力モード)　{ RAPI_SINGLE, RAPI_SR, RAPI_PHASE_DELAYED }から 1 つを指定します。省略値は RAPI_SINGLE です。

(出力初期値)　{ RAPI_OUT_INIT_0, RAPI_OUT_INIT_1 }から 1 つを指定します。省略値は RAPI_OUT_INIT_0 です。

(出力反転)　出力反転する場合、RAPI_OUTPUT_REVERSED を指定します。出力反転を指定しない場合は、「出力反転しない」に設定します。

(リロード)　{ RAPI_RELOAD_WITH_RESET, RAPI_RELOAD_WITH_WRITE }から 1 つを指定します。省略値は RAPI_RELOAD_WITH_WRITE です。

(R8C)

比較値を格納する配列のポインタを指定します。

・タイマ C 使用時 (RAPI_TIMER_C を指定時)、タイマ RF 使用時 (RAPI_TIMER_RF を指定時)

[0]：比較値 0 を 16 ビットで指定します。

[1]：比較値 1 を 16 ビットで指定します。

・タイマ RC 使用時 (RAPI_TIMER_RC を指定時)

[0]：TRCIOA 端子の出力動作を指定します。

[1]：TRCIOB 端子の出力動作を指定します。

[2]：TRCIOC 端子の出力動作を指定します。

[3]：TRCIOD 端子の出力動作を指定します。

[4]：ジェネラルレジスタ A へ設定する比較値を 16 ビットで指定します。

[5]：ジェネラルレジスタ B へ設定する比較値を 16 ビットで指定します。

[6]：ジェネラルレジスタ C へ設定する比較値を 16 ビットで指定します。

[7]：ジェネラルレジスタ D へ設定する比較値を 16 ビットで指定します。

・タイマ RD 使用時 (RAPI_TIMER_RD0～RAPI_TIMER_RD1 を指定時)

[0]：TRDIOAi(i=0,1)端子の出力動作を指定します。

[1]：TRDIOBi(i=0,1)端子の出力動作を指定します。

[2]：TRDIOCi(i=0,1)端子の出力動作を指定します。

[3]：TRDIODi(i=0,1)端子の出力動作を指定します。

[4]：ジェネラルレジスタ A へ設定する比較値を 16 ビットで指定します。

[5]：ジェネラルレジスタ B へ設定する比較値を 16 ビットで指定します。

[6]：ジェネラルレジスタ C へ設定する比較値を 16 ビットで指定します。

[7]：ジェネラルレジスタ D へ設定する比較値を 16 ビットで指定します。

・タイマ RE 使用時 (RAPI_TIMER_RE を指定時) に指定可能な定義値

[0]：タイマ RE コンペアデータレジスタへ設定する比較値を 8 ビットで指定します。

配列の要素の各アウトプットコンペアの出力動作の指定には、次の定義値を使用することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。使用しないチャネルに対応する出力動作の要素には、0 を設定してください。

RAPI_OUT_0	出力波形に 0 出力を選択する。
RAPI_OUT_1	出力波形に 1 出力を選択する。
RAPI_OUT_TOGGLE	出力波形にトグル出力を選択する。

RAPI_OUT_INIT_0	初期出力に 0 出力を選択する。
RAPI_OUT_INIT_1	初期出力に 1 出力を選択する。

(H8S)

比較値を格納する配列のポインタを指定します。

・タイマ RC 使用時 (RAPI_TIMER_RC を指定時) または、タイマ RD 使用時 (RAPI_TIMER_RD0～ RAPI_TIMER_RD3 を指定時)

[0]：FTIOA 端子の出力動作を指定します。

[1]：FTIOB 端子の出力動作を指定します。

[2]：FTIOC 端子の出力動作を指定します。

[3]：FTIOD 端子の出力動作を指定します。

[4]：ジェネラルレジスタ A へ設定する比較値を 16 ビットで指定します。

[5]：ジェネラルレジスタ B へ設定する比較値を 16 ビットで指定します。

[6]：ジェネラルレジスタ C へ設定する比較値を 16 ビットで指定します。

[7]：ジェネラルレジスタ D へ設定する比較値を 16 ビットで指定します。

配列の要素の各アウトプットコンペアの出力動作の指定には、次の定義値を使用することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。使用しないチャネルに対応する出力動作の要素には、0 を設定してください。

RAPI_OUT_0	出力波形に 0 出力を選択する。
RAPI_OUT_1	出力波形に 1 出力を選択する。
RAPI_OUT_TOGGLE	出力波形にトグル出力を選択する。
RAPI_OUT_INIT_0	初期出力に 0 出力を選択する。
RAPI_OUT_INIT_1	初期出力に 1 出力を選択する。
RAPI_AD_TRIGGER_ON	コンペアマッチで A/D 変換開始トリガを発生する。

・タイマ RE 使用時 (RAPI_TIMER_RE を指定時)

[0]：タイマ RE コンペアデータレジスタへ設定する比較値を 8 ビットで指定します。

・タイマ RG 使用時 (RAPI_TIMER_RG を指定時)

[0]：FTIOA 端子の出力動作を指定します。

[1]：FTIOB 端子の出力動作を指定します。

[2]：ジェネラルレジスタ A へ設定する比較値を 16 ビットで指定します。

[3]：ジェネラルレジスタ B へ設定する比較値を 16 ビットで指定します。

配列の要素の各アウトプットコンペアの出力動作の指定には、次の定義値を使用することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。使用しないチャネルに対応する出力動作の要素には、0 を設定してください。

RAPI_OUT_0	出力波形に 0 出力を選択する。
RAPI_OUT_1	出力波形に 1 出力を選択する。
RAPI_OUT_TOGGLE	出力波形にトグル出力を選択する。

(H8/300H)

比較値を格納する配列のポインタを指定します。

[0]：FTIOA 端子の出力動作を指定します。

[1]：FTIOB 端子の出力動作を指定します。

[2]：FTIOC 端子の出力動作を指定します。

[3]：FTIOD 端子の出力動作を指定します。

[4]：ジェネラルレジスタ A へ設定する比較値を 16 ビットで指定します。

[5]：ジェネラルレジスタ B へ設定する比較値を 16 ビットで指定します。

[6]：ジェネラルレジスタ C へ設定する比較値を 16 ビットで指定します。
[7]：ジェネラルレジスタ D へ設定する比較値を 16 ビットで指定します。
配列の要素の各アウトプットコンペアの出力動作の指定には、次の定義値を
使用することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して
指定します。使用しないチャネルに対応する出力動作の要素には、0 を設定してください。

RAPI_OUT_0	出力波形に 0 出力を選択する。
RAPI_OUT_1	出力波形に 1 出力を選択する。
RAPI_OUT_TOGGLE	出力波形にトグル出力を選択する。
RAPI_OUT_INIT_0	初期出力に 0 出力を選択する。
RAPI_OUT_INIT_1	初期出力に 1 出力を選択する。

[data4]

(M16C)

タイマ S の各レジスタへの設定値を格納する配列のポインタを指定します。
[0]：機能選択レジスタおよび機能許可レジスタへの設定値を指定します。
波形生成機能を許可するチャネルを指定します。
[1]：割り込み許可レジスタ 0 への設定値を指定します。IO/CO 割り込み 0 要求を
許可するチャネルを指定します。
[2]：割り込み許可レジスタ 1 への設定値を指定します。IO/CO 割り込み 1 要求を
許可するチャネルを指定します。
[3]：カウントソース分周レジスタへの設定値を指定します。カウントソースの
(n+1)分周の n の値を 8 ビットで指定します。
配列の各要素で指定するチャネルには、次の定義値を使用します。複数の定義値を
同時に指定する場合は、"|"記号を使用して指定します。0 を指定した場合は、
対応するレジスタへ 0 を設定します。

RAPI_CHANNEL0	チャネル 0 を選択する。
RAPI_CHANNEL1	チャネル 1 を選択する。
RAPI_CHANNEL2	チャネル 2 を選択する。
RAPI_CHANNEL3	チャネル 3 を選択する。
RAPI_CHANNEL4	チャネル 4 を選択する。
RAPI_CHANNEL5	チャネル 5 を選択する。
RAPI_CHANNEL6	チャネル 6 を選択する。
RAPI_CHANNEL7	チャネル 7 を選択する。

(R8C)(H8S)(H8/300H)

0 を指定します。

[data5]

(M16C)

コールバック関数を格納する配列のポインタを指定します。
[0]：IC/OC ベースタイマ割り込みのコールバック関数のポインタを指定します。
IC/OC ベースタイマ割り込みのコールバック関数のポインタを指定しない場合は、
0 を設定します。
[1]：IC/OC 割り込み 0 のコールバック関数のポインタを指定します。
IC/OC 割り込み 0 のコールバック関数のポインタを指定しない場合は、

0 を設定します。

[2]：IC/OC 割り込み 1 のコールバック関数のポインタを指定します。
IC/OC 割り込み 1 のコールバック関数のポインタを指定しない場合は、
0 を設定します。

(R8C)

・タイマ C 使用時（RAPI_TIMER_C を指定時）または、タイマ RF 使用時（RAPI_TIMER_RF を指定時）

コールバック関数を格納する配列のポインタを指定します。

[0]：タイマ割り込みのコールバック関数のポインタを指定します。
タイマ割り込みのコールバック関数のポインタを指定しない場合は、
0 を設定します。

[1]：コンペアマッチ割り込み 0 のコールバック関数のポインタを指定します。
コンペアマッチ割り込み 0 のコールバック関数のポインタを指定しない場合は、
0 を設定します。

[2]：コンペアマッチ割り込み 1 のコールバック関数のポインタを指定します。
コンペアマッチ割り込み 1 のコールバック関数のポインタを指定しない場合は、
0 を設定します。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）、タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD1 を指定時）または、タイマ RE 使用時（RAPI_TIMER_RE を指定時）

コールバック関数を格納する変数のポインタを指定します。
コールバック関数を格納する変数のポインタを指定しない場合は、RAPI_NULL を
設定します。

(H8S)

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）またはタイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時）

[0]：カウンタオーバフロー割り込みのコールバック関数のポインタを指定します。
カウンタオーバフロー割り込みのコールバック関数のポインタを指定しない
場合は 0 を設定します。

[1]：コンペアマッチ A 割り込みのコールバック関数のポインタを指定します。
コンペアマッチ A 割り込みのコールバック関数のポインタを指定しない
場合は 0 を設定します。

[2]：コンペアマッチ B 割り込みのコールバック関数のポインタを指定します。
コンペアマッチ B 割り込みのコールバック関数のポインタを指定しない
場合は 0 を設定します。

[3]：コンペアマッチ C 割り込みのコールバック関数のポインタを指定します。
コンペアマッチ C 割り込みのコールバック関数のポインタを指定しない
場合は 0 を設定します。

[4]：コンペアマッチ D 割り込みのコールバック関数のポインタを指定します。
コンペアマッチ D 割り込みのコールバック関数のポインタを指定しない
場合は 0 を設定します。

・タイマ RE 使用時（RAPI_TIMER_RE を指定時）

コールバック関数を格納する変数のポインタを指定します。
コールバック関数を格納する変数のポインタを指定しない場合は、RAPI_NULL を
設定します。

・タイマ RG 使用時（RAPI_TIMER_RG を指定時）

[0]：カウンタオーバフロー割り込みのコールバック関数のポインタを指定します。

カウンタオーバフロー割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。

[1]：コンペアマッチ A 割り込みのコールバック関数のポインタを指定します。
コンペアマッチ A 割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。

[2]：コンペアマッチ B 割り込みのコールバック関数のポインタを指定します。
コンペアマッチ B 割り込みのコールバック関数のポインタを指定しない場合は 0 を設定します。

(H8/300H)

コールバック関数を格納する変数のポインタを指定します。
コールバック関数を格納する変数のポインタを指定しない場合は、RAPI_NULL を設定します。

戻り値

タイマまたはインテリジェント I/O 指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(アウトプットコンペアモード)

参照

__EnableOutputCompare, __DestroyOutputCompare

備考

- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- 第 3 引数の各アウトプットコンペアの出力動作の定義値以外の値を指定した場合の動作は保証されません。
- M16C 使用時に第 4 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できるタイマは CPU により異なります。
- H8S および、H8/300H で使用する場合、指定したタイマのモジュールスタンバイ解除後に設定をおこないます。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void TimerIntFunc0( void ){}
void TimerIntFunc1( void ){}
void TimerIntFunc2( void ){}

void func( void )
{
    unsigned int *p_func[] =  {(void*) TimerIntFunc0,(void*) TimerIntFunc1,
                               (void*) TimerIntFunc2};
    unsigned char p_ic[] =  {1,2,3};
    unsigned int p_cmp[] =  {0x1234, 0x9876};

    /* Set up timer C as output compare mode */
    __CreateOutputCompare(
        RAPI_TIMER_C|RAPI_TIMER_ON|RAPI_RELOAD|RAPI_L_1|RAPI_L_0|RAPI_F32|
        RAPI_CMP02_ENABLE|RAPI_CMP12_ENABLE, p_cmp, p_ic, p_func );
}
```

__EnableOutputCompare

要約

< アウトプットコンペアモード動作制御 >

Boolean __EnableOutputCompare(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したアウトプットコンペアモードに設定されているタイマまたはインテリジェント I/O の動作開始/動作停止を制御します。

[data]

data には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_S	タイマ S を選択する。
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する。
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する。

(R8C)

RAPI_TIMER_C	タイマ C を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RE	タイマ RE を選択する。
RAPI_TIMER_RF	タイマ RF を選択する。
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する。
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する。

(H8S)

RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
RAPI_TIMER_RE	タイマ RE を選択する。
RAPI_TIMER_RG	タイマ RG を選択する。
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する。
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する。

(H8/300H)

RAPI_TIMER_W	タイマ W を選択する。

RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
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 動作開始に設定する。
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 動作停止に設定する。

戻り値

タイマまたはインテリジェント I/O 指定が正しくない場合は RAPI_FALSE を返し、
それ以外は RAPI_TRUE を返します。

機能

タイマ(アウトプットコンペアモード)

参照

__CreateOutputCompare, __DestroyOutputCompare

備考

・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できるタイマは CPU により異なります。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void func( void )
{
    /* Enable timer C as output compare mode */
    __EnableOutputCompare( RAPI_TIMER_C|RAPI_TIMER_ON );
}
```

__DestroyOutputCompare

要約

< アウトプットコンペアモード設定破棄 >

Boolean __DestroyOutputCompare(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したアウトプットコンペアモードに設定されているタイマまたはインテリジェント I/O の設定を破棄します。

[data]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_S	タイマ S を選択する。

(R8C)

RAPI_TIMER_C	タイマ C を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RE	タイマ RE を選択する。
RAPI_TIMER_RF	タイマ RF を選択する。

(H8S)

RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
RAPI_TIMER_RE	タイマ RE を選択する。
RAPI_TIMER_RG	タイマ RG を選択する。

(H8/300H)

RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。

戻り値

タイマまたはインテリジェント I/O 指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

タイマ(アウトプットコンペアモード)

参照　__CreateOutputCompare, __EnableOutputCompare

備考
- 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できるタイマは CPU により異なります。
- H8S および、H8/300H で使用する場合、指定したタイマを、設定破棄後、モジュールスタンバイ状態に設定します。

プログラム例

```
#include "rapi_timer_r8c_13.h"

void func( void )
{
    /* Destroy the setting of timer C as output compare mode */
    __DestroyOutputCompare( RAPI_TIMER_C );
}
```

__SetTimerRegister

要約

< タイマレジスタ設定 >

Boolean __SetTimerRegister(unsigned long data1, unsigned int *data2)

data1	設定データ 1（機種により内容が異なる）
data2	レジスタ値を格納するバッファポインタ

解説

指定したタイマまたはインテリジェント I/O のレジスタを設定します。

[data1]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を選択する。
RAPI_TIMER_A1	タイマ A チャネル 1 を選択する。
RAPI_TIMER_A2	タイマ A チャネル 2 を選択する。
RAPI_TIMER_A3	タイマ A チャネル 3 を選択する。
RAPI_TIMER_A4	タイマ A チャネル 4 を選択する。
RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。
RAPI_TIMER_S	タイマ S を選択する。

(R8C)

RAPI_TIMER_C	タイマ C を選択する。
RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_Y	タイマ Y を選択する。
RAPI_TIMER_Z	タイマ Z を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RE	タイマ RE を選択する。
RAPI_TIMER_RF	タイマ RF を選択する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
RAPI_TIMER_RE	タイマ RE を選択する。
RAPI_TIMER_RG	タイマ RG を選択する。

(H8/300H)

RAPI_TIMER_A	タイマ A を選択する。
RAPI_TIMER_B1	タイマ B1 を選択する。
RAPI_TIMER_V	タイマ V を選択する。
RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。

[data2]

レジスタ値を格納するバッファポインタの内容は、次に示すとおりに指定してください。
各レジスタへの値の設定は、バッファポインタの要素順におこないます。

(M16C)

・タイマ A 使用時 (RAPI_TIMER_A0～RAPI_TIMER_A4 を指定時)

[0]：タイマ Ai モードレジスタ(i=0～4)への設定値を指定します。
[1]：タイマ Ai レジスタ(i=0～4)への設定値を指定します。
[2]：アップダウンフラグレジスタへの設定値を指定します。
[3]：ワンショット開始フラグレジスタへの設定値を指定します。
[4]：トリガ選択レジスタへの設定値を指定します。
[5]：時計用プリスケーラリセットレジスタへの設定値を指定します。
[6]：カウント開始フラグレジスタへの設定値を指定します。

・タイマ B 使用時 (RAPI_TIMER_B0～RAPI_TIMER_B2 を指定時)

[0]：タイマ Bi モードレジスタ(i=0～2)への設定値を指定します。
[1]：タイマ Bi レジスタ(i=0～2)への設定値を指定します。
[3]：時計用プリスケーラリセットレジスタへの設定値を指定します。
[4]：カウント開始フラグレジスタへの設定値を指定します。

・タイマ S 使用時 (RAPI_TIMER_S を指定時)

[0]：ベースタイマレジスタへの設定値を指定します。
[1]：ベースタイマリセットレジスタへの設定値を指定します。
[2]：ベースタイマ制御レジスタ 0 への設定値を指定します。
[3]：ベースタイマ制御レジスタ 1 への設定値を指定します。
[4]：カウントソース分周レジスタへの設定値を指定します。
[5]：時間計測制御レジスタ 0 への設定値を指定します。
[6]：時間計測制御レジスタ 1 への設定値を指定します。
[7]：時間計測制御レジスタ 2 への設定値を指定します。
[8]：時間計測制御レジスタ 3 への設定値を指定します。
[9]：時間計測制御レジスタ 4 への設定値を指定します。
[10]：時間計測制御レジスタ 5 への設定値を指定します。
[11]：時間計測制御レジスタ 6 への設定値を指定します。
[12]：時間計測制御レジスタ 7 への設定値を指定します。

[13]：時間計測プリスケーラレジスタ 6 への設定値を指定します。
[14]：時間計測プリスケーラレジスタ 7 への設定値を指定します。
[15]：波形生成制御レジスタ 0 への設定値を指定します。
[16]：波形生成制御レジスタ 1 への設定値を指定します。
[17]：波形生成制御レジスタ 2 への設定値を指定します。
[18]：波形生成制御レジスタ 3 への設定値を指定します。
[19]：波形生成制御レジスタ 4 への設定値を指定します。
[20]：波形生成制御レジスタ 5 への設定値を指定します。
[21]：波形生成制御レジスタ 6 への設定値を指定します。
[22]：波形生成制御レジスタ 7 への設定値を指定します。
[23]：波形生成レジスタ 0 への設定値を指定します。
[24]：波形生成レジスタ 1 への設定値を指定します。
[25]：波形生成レジスタ 2 への設定値を指定します。
[26]：波形生成レジスタ 3 への設定値を指定します。
[27]：波形生成レジスタ 4 への設定値を指定します。
[28]：波形生成レジスタ 5 への設定値を指定します。
[29]：波形生成レジスタ 6 への設定値を指定します。
[30]：波形生成レジスタ 7 への設定値を指定します。
[31]：機能選択レジスタへの設定値を指定します。
[32]：機能許可レジスタへの設定値を指定します。
[33]：割り込み要求レジスタへの設定値を指定します。
[34]：割り込み許可レジスタ 0 への設定値を指定します。
[35]：割り込み許可レジスタ 1 への設定値を指定します。

(R8C)

・タイマ C 使用時 (RAPI_TIMER_C を指定時)

[0]：タイマ C 出力制御レジスタへの設定値を指定します。
[1]：タイマ C 制御レジスタ 0 への設定値を指定します。
[2]：タイマ C 制御レジスタ 1 への設定値を指定します。
[3]：キャプチャ、コンペア 0 レジスタへの設定値を指定します。
[4]：コンペア 1 レジスタへの設定値を指定します。

・タイマ X 使用時 (RAPI_TIMER_X を指定時)

[0]：タイマカウントソース設定レジスタへの設定値を指定します。
[1]：プリスケーラ X レジスタへの設定値を指定します。
[2]：タイマ X レジスタへの設定値を指定します。
[3]：タイマ X モードレジスタへの設定値を指定します。

・タイマ Y 使用時 (RAPI_TIMER_Y を指定時)

[0]：タイマカウントソース設定レジスタへの設定値を指定します。
[1]：プリスケーラ Y レジスタへの設定値を指定します。
[2]：タイマ Y プライマリレジスタへの設定値を指定します。
[3]：タイマ Y セカンダリレジスタへの設定値を指定します。
[4]：タイマ Y、Z 波形出力制御レジスタへの設定値を指定します。
[5]：タイマ Y、Z 出力制御レジスタへの設定値を指定します。
[6]：タイマ Y、Z モードレジスタへの設定値を指定します。

・タイマ Z 使用時 (RAPI_TIMER_Z を指定時)

[0]：タイマカウントソース設定レジスタへの設定値を指定します。

[1]：プリスケーラ Z レジスタへの設定値を指定します。
[2]：タイマ Z プライマリレジスタへの設定値を指定します。
[3]：タイマ Z セカンダリレジスタへの設定値を指定します。
[4]：タイマ Y、Z 波形出力制御レジスタへの設定値を指定します。
[5]：タイマ Y、Z 出力制御レジスタへの設定値を指定します。
[6]：タイマ Y、Z モードレジスタへの設定値を指定します。

・タイマ RA 使用時（RAPI_TIMER_RA を指定時）

[0]：タイマ RA I/O 制御レジスタへの設定値を指定します。
[1]：タイマ RA プリスケーラレジスタへの設定値を指定します。
[2]：タイマ RA レジスタへの設定値を指定します。
[3]：タイマ RA モードレジスタへの設定値を指定します。
[4]：タイマ RA 制御レジスタへの設定値を指定します。

・タイマ RB 使用時（RAPI_TIMER_RB を指定時）

[0]：タイマ RB ワンショット制御レジスタへの設定値を指定します。
[1]：タイマ RB I/O 制御レジスタへの設定値を指定します。
[2]：タイマ RB プリスケーラレジスタへの設定値を指定します。
[3]：タイマ RB プライマリレジスタへの設定値を指定します。
[4]：タイマ RB セカンダリレジスタへの設定値を指定します。
[5]：タイマ RB モードレジスタへの設定値を指定します。
[6]：タイマ RB 制御レジスタへの設定値を指定します。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）

[0]：タイマ RC ジェネラルレジスタ A への設定値を指定します。
[1]：タイマ RC ジェネラルレジスタ B への設定値を指定します。
[2]：タイマ RC ジェネラルレジスタ C への設定値を指定します。
[3]：タイマ RC ジェネラルレジスタ D への設定値を指定します。
[4]：タイマ RC デジタルフィルタ機能選択レジスタへの設定値を指定します。
[5]：タイマ RC 制御レジスタ 1 への設定値を指定します。
[6]：タイマ RC 制御レジスタ 2 への設定値を指定します。
[7]：タイマ RC I/O 制御レジスタ 0 への設定値を指定します。
[8]：タイマ CD I/O 制御レジスタ 1 への設定値を指定します。
[9]：タイマ RC ステータスレジスタへの設定値を指定します。
[10]：タイマ RC 割り込み許可レジスタへの設定値を指定します。
[11]：タイマ RC カウンタへの設定値を指定します。
[12]：タイマ RC アウトプットマスタ許可レジスタを指定します。
[13]：タイマ RC モードレジスタへの設定値を指定します。

・タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD1 を指定時）

[0]：タイマ RD モードレジスタへの設定値を指定します。
[1]：タイマ RD PWM モードレジスタへの設定値を指定します。
[2]：タイマ RD 機能制御レジスタへの設定値を指定します。
[3]：タイマ RD ジェネラルレジスタ A i(i=0,1)への設定値を指定します。
[4]：タイマ RD ジェネラルレジスタ B i(i=0,1)への設定値を指定します。
[5]：タイマ RD ジェネラルレジスタ C i(i=0,1)への設定値を指定します。
[6]：タイマ RD ジェネラルレジスタ D i(i=0,1)への設定値を指定します。
[7]：タイマ RD デジタルフィルタ機能選択レジスタ i(i=0,1)への設定値を指定します。
[8]：タイマ RD 制御レジスタ i(i=0,1)への設定値を指定します。

[9]：タイマ RD I/O 制御レジスタ Ai(i=0,1)への設定値を指定します。
[10]：タイマ RD I/O 制御レジスタ Ci(i=0,1)への設定値を指定します。
[11]：タイマ RD ステータスレジスタ i(i=0,1)への設定値を指定します。
[12]：タイマ RD 割り込み許可レジスタ i(i=0,1)への設定値を指定します。
[13]：タイマ RD カウンタ i(i=0,1)への設定値を指定します。
[14]：タイマ RD スタートレジスタへの設定値を指定します。
[15]：タイマ RD アウトプットマスタ許可レジスタ 1 を指定します。
[16]：タイマ RD アウトプットマスタ許可レジスタ 2 を指定します。
[17]：タイマ RD アウトプット制御レジスタを指定します。

・タイマ RE 使用時 (RAPI_TIMER_RE を指定時)

[0]：タイマ RE 秒データレジスタへの設定値を指定します。
[1]：タイマ RE 分データレジスタへの設定値を指定します。
[2]：タイマ RE 時データレジスタへの設定値を指定します。
[3]：タイマ RE 曜日データレジスタへの設定値を指定します。
[4]：タイマ RE 制御レジスタ 2 への設定値を指定します。
[5]：タイマ RE カウントソース選択レジスタへの設定値を指定します。
[6]：タイマ RE 制御レジスタ 1 への設定値を指定します。

・タイマ RF 使用時 (RAPI_TIMER_RF を指定時)

[0]：タイマ RF 制御レジスタ 0 への設定値を指定します。
[1]：キャプチャ、コンペア 0 レジスタへの設定値を指定します。
[2]：コンペア 1 レジスタへの設定値を指定します。
[3]：タイマ RF 出力制御レジスタ 0 への設定値を指定します。
[4]：タイマ RF 制御レジスタ 1 への設定値を指定します。

(H8S)

・タイマ RA 使用時 (RAPI_TIMER_RA を指定時)

[0]：タイマ RA I/O コントロールレジスタへの設定値を指定します。
[1]：タイマ RA プリスケーラレジスタへの設定値を指定します。
[2]：タイマ RA タイマレジスタへの設定値を指定します。
[3]：タイマ RA モードレジスタへの設定値を指定します。
[4]：タイマ RA 割り込み要求ステータスレジスタへの設定値を指定します。
[5]：タイマ RA コントロールレジスタへの設定値を指定します。

・タイマ RB 使用時 (RAPI_TIMER_RB を指定時)

[0]：タイマ RB ワンショットコントロールレジスタへの設定値を指定します。
[1]：タイマ RB I/O コントロールレジスタへの設定値を指定します。
[2]：タイマ RB プリスケーラレジスタへの設定値を指定します。
[3]：タイマ RB プライマリレジスタへの設定値を指定します。
[4]：タイマ RB セカンダリレジスタへの設定値を指定します。
[5]：タイマ RB モードレジスタへの設定値を指定します。
[6]：タイマ RB 割り込み要求ステータスレジスタへの設定値を指定します。
[7]：タイマ RB コントロールレジスタへの設定値を指定します。

・タイマ RC 使用時 (RAPI_TIMER_RC を指定時)

[0]：タイマ RC コントロールレジスタ 1 への設定値を指定します。
[1]：タイマ RC コントロールレジスタ 2 への設定値を指定します。
[2]：タイマ RC インタラプトイネーブルレジスタへの設定値を指定します。
[3]：タイマ RC ステータスレジスタへの設定値を指定します。

[4]：タイマ RC I/O コントロールレジスタ 0 への設定値を指定します。
[5]：タイマ RC I/O コントロールレジスタ 1 への設定値を指定します。
[6]：タイマ RC アウトプットイネーブルレジスタへの設定値を指定します。
[7]：タイマ RC デジタルフィルタ機能選択レジスタへの設定値を指定します。
[8]：タイマ RC A/D 変換トリガコントロールレジスタへの設定値を指定します。
[9]：タイマ RC カウンタへの設定値を指定します。
[10]：ジェネラルレジスタ A への設定値を指定します。
[11]：ジェネラルレジスタ B への設定値を指定します。
[12]：ジェネラルレジスタ C への設定値を指定します。
[13]：ジェネラルレジスタ D への設定値を指定します。
[14]：タイマ RC モードレジスタへの設定値を指定します。

・タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時）

[0]：タイマ RD モードレジスタへの設定値を指定します。
[1]：タイマ RD PWM モードレジスタへの設定値を指定します。
[2]：タイマ RD ファンクションコントロールレジスタへの設定値を指定します。
[3]：タイマ RD アウトプットマスタイネーブルレジスタ 1 への設定値を指定します。
[4]：タイマ RD アウトプットマスタイネーブルレジスタ 2 への設定値を指定します。
[5]：タイマ RD アウトプットコントロールレジスタへの設定値を指定します。
[6]：タイマ RD A/D 変換開始トリガコントロールレジスタへの設定値を指定します。
[7]：タイマ RD カウンタ_i(i=0,1)への設定値を指定します。
[8]：ジェネラルレジスタ A_i(i=0,1)への設定値を指定します。
[9]：ジェネラルレジスタ B_i(i=0,1)への設定値を指定します。
[10]：ジェネラルレジスタ C_i(i=0,1)への設定値を指定します。
[11]：ジェネラルレジスタ D_i(i=0,1)への設定値を指定します。
[12]：タイマ RD コントロールレジスタ_i(i=0,1)への設定値を指定します。
[13]：タイマ RD I/O コントロールレジスタ A_i(i=0,1)への設定値を指定します。
[14]：タイマ RD I/O コントロールレジスタ C_i(i=0,1)への設定値を指定します。
[15]：タイマ RD ステータスレジスタ_i(i=0,1)への設定値を指定します。
[16]：タイマ RD インタラプトイネーブルレジスタ_i(i=0,1)への設定値を指定します。
[17]：PWM モードアウトプットレベルコントロールレジスタ_i(i=0,1)への設定値を指定します。
[18]：タイマ RD デジタルフィルタ機能選択レジスタ i(i=0,1)への設定値を指定します。
[19]：タイマ RD スタートレジスタへの設定値を指定します。

・タイマ RE 使用時（RAPI_TIMER_RE を指定時）

[0]：タイマ RE 秒データレジスタ/カウンタデータレジスタへの設定値を指定します。
[1]：タイマ RE 分データレジスタ/コンペアデータレジスタへの設定値を指定します。
[2]：タイマ RE 時データレジスタへの設定値を指定します。
[3]：タイマ RE 曜日データレジスタへの設定値を指定します。
[4]：タイマ RE コントロールレジスタ 2 への設定値を指定します。
[5]：タイマ RE クロックソースソースセレクトレジスタへの設定値を指定します。
[6]：タイマ RE 割り込みフラグレジスタへの設定値を指定します。
[7]：タイマ RE コントロールレジスタ 1 への設定値を指定します。

・タイマ RG 使用時（RAPI_TIMER_RG を指定時）

[0]：タイマ RG カウンタコントロールレジスタへの設定値を指定します。
[1]：タイマ RG コントロールレジスタへの設定値を指定します。
[2]：タイマ RG I/O コントロールレジスタへの設定値を指定します。

[3]：タイマ RG ステータスレジスタへの設定値を指定します。
[4]：タイマ RG インタラプトイネーブルレジスタへの設定値を指定します。
[5]：タイマ RG カウンタへの設定値を指定します。
[6]：ジェネラルレジスタ A への設定値を指定します。
[7]：ジェネラルレジスタ B への設定値を指定します。
[8]：GRA バッファレジスタへの設定値を指定します。
[9]：GRB バッファレジスタへの設定値を指定します。
[10]：タイマ RG モードレジスタへの設定値を指定します。

(H8/300H)

・タイマ A 使用時（RAPI_TIMER_A を指定時）

[0]：タイマモードレジスタ A への設定値を指定します。
[1]：タイマカウンタ A への設定値を指定します。

・タイマ B1 使用時（RAPI_TIMER_B1 を指定時）

[0]：タイマモードレジスタ B1 への設定値を指定します。
[1]：タイマロードレジスタ B1 への設定値を指定します。

・タイマ V 使用時（RAPI_TIMER_V を指定時）

[0]：タイマカウンタ V への設定値を指定します。
[1]：タイムコンスタントレジスタ A への設定値を指定します。
[2]：タイムコンスタントレジスタ B への設定値を指定します。
[3]：タイマコントロールレジスタ V0 への設定値を指定します。
[4]：タイマコントロールレジスタ V1 への設定値を指定します。
[5]：タイマコントロール/ステータスレジスタ V への設定値を指定します。

・タイマ W 使用時（RAPI_TIMER_W を指定時）

[0]：タイマコントロールレジスタ W への設定値を指定します。
[1]：タイマインタラプトイネーブルレジスタ W への設定値を指定します。
[2]：タイマステータスレジスタ W への設定値を指定します。
[3]：タイマ I/O コントロールレジスタ 0 への設定値を指定します。
[4]：タイマ I/O コントロールレジスタ 1 への設定値を指定します。
[5]：タイマカウンタへの設定値を指定します。
[6]：ジェネラルレジスタ A への設定値を指定します。
[7]：ジェネラルレジスタ B への設定値を指定します。
[8]：ジェネラルレジスタ C への設定値を指定します。
[9]：ジェネラルレジスタ D への設定値を指定します。
[10]：タイマモードレジスタ W への設定値を指定します。

・タイマ Z 使用時（RAPI_TIMER_Z0～RAPI_TIMER_Z1 を指定時）

[0]：タイマモードレジスタへの設定値を指定します。
[1]：タイマ PWM モードレジスタへの設定値を指定します。
[2]：タイマファンクションコントロールレジスタへの設定値を指定します。
[3]：タイマアウトプットマスタイネーブルレジスタへの設定値を指定します。
[4]：タイマアウトプットコントロールレジスタへの設定値を指定します。
[5]：タイマカウンタ_i(i=0,1)への設定値を指定します。
[6]：ジェネラルレジスタ A_i(i=0,1)への設定値を指定します。
[7]：ジェネラルレジスタ B_i(i=0,1)への設定値を指定します。
[8]：ジェネラルレジスタ C_i(i=0,1)への設定値を指定します。
[9]：ジェネラルレジスタ D_i(i=0,1)への設定値を指定します。

[10]：タイマコントロールレジスタ_i(i=0,1)への設定値を指定します。
[11]：タイマ I/O コントロールレジスタ A_i(i=0,1)への設定値を指定します。
[12]：タイマ I/O コントロールレジスタ C_i(i=0,1)への設定値を指定します。
[13]：タイマステータスレジスタ_i(i=0,1)への設定値を指定します。
[14]：タイマインタラプトイネーブルレジスタ_i(i=0,1)への設定値を指定します。
[15]：PWM モードアウトプットレベルコントロールレジスタ_i(i=0,1)への設定値を指定します。
[16]：タイマスタートレジスタへの設定値を指定します。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）

[0]：タイマ RC コントロールレジスタ 1 への設定値を指定します。
[1]：タイマ RC コントロールレジスタ 2 への設定値を指定します。
[2]：タイマ RC インタラプトイネーブルレジスタへの設定値を指定します。
[3]：タイマ RC ステータスレジスタへの設定値を指定します。
[4]：タイマ RC I/O コントロールレジスタ 0 への設定値を指定します。
[5]：タイマ RC I/O コントロールレジスタ 1 への設定値を指定します。
[6]：タイマ RC アウトプットイネーブルレジスタへの設定値を指定します。
[7]：タイマ RC デジタルフィルタ機能選択レジスタへの設定値を指定します。
[8]：タイマ RC カウンタへの設定値を指定します。
[9]：ジェネラルレジスタ A への設定値を指定します。
[10]：ジェネラルレジスタ B への設定値を指定します。
[11]：ジェネラルレジスタ C への設定値を指定します。
[12]：ジェネラルレジスタ D への設定値を指定します。
[13]：タイマ RC モードレジスタへの設定値を指定します。

・タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時）

[0]：タイマ RD モードレジスタへの設定値を指定します。
[1]：タイマ RD PWM モードレジスタへの設定値を指定します。
[2]：タイマ RD ファンクションコントロールレジスタへの設定値を指定します。
[3]：タイマ RD アウトプットマスタイネーブルレジスタ 1 への設定値を指定します。
[4]：タイマ RD アウトプットマスタイネーブルレジスタ 2 への設定値を指定します。
[5]：タイマ RD アウトプットコントロールレジスタへの設定値を指定します。
[6]：タイマ RD カウンタ_i(i=0,1)への設定値を指定します。
[7]：ジェネラルレジスタ A_i(i=0,1)への設定値を指定します。
[8]：ジェネラルレジスタ B_i(i=0,1)への設定値を指定します。
[9]：ジェネラルレジスタ C_i(i=0,1)への設定値を指定します。
[10]：ジェネラルレジスタ D_i(i=0,1)への設定値を指定します。
[11]：タイマ RD コントロールレジスタ_i(i=0,1)への設定値を指定します。
[12]：タイマ RD I/O コントロールレジスタ A_i(i=0,1)への設定値を指定します。
[13]：タイマ RD I/O コントロールレジスタ C_i(i=0,1)への設定値を指定します。
[14]：タイマ RD ステータスレジスタ_i(i=0,1)への設定値を指定します。
[15]：タイマ RD インタラプトイネーブルレジスタ_i(i=0,1)への設定値を指定します。
[16]：PWM モードアウトプットレベルコントロールレジスタ_i(i=0,1)への設定値を指定します。
[17]：タイマ RD デジタルフィルタ機能選択レジスタ i(i=0,1)への設定値を指定します。
[18]：タイマ RD スタートレジスタへの設定値を指定します。

戻り値　タイマまたはインテリジェント I/O 指定が正しくない場合は RAPI_FALSE を返し、

それ以外は RAPI_TRUE を返します。

機能　タイマ(レジスタ操作)

参照　__EnableTimerRegister, __ClearTimerRegister, __GetTimerRegister

備考
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できるタイマは CPU により異なります。

プログラム例

```
#include " rapi_timer_r8c_13.h"

void func( void )
{
    unsigned char data[] =  {0,0,0,0,0,0,0};

    /* Set up timer Z register */
    __SetTimerRegister( RAPI_TIMER_Z, data );
}
```

__EnableTimerRegister

要約

< タイマレジスタ動作制御 >

Boolean __EnableTimerRegister(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したタイマまたはインテリジェント I/O の動作開始/動作停止を制御します。

[data]

data には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を選択する。
RAPI_TIMER_A1	タイマ A チャネル 1 を選択する。
RAPI_TIMER_A2	タイマ A チャネル 2 を選択する。
RAPI_TIMER_A3	タイマ A チャネル 3 を選択する。
RAPI_TIMER_A4	タイマ A チャネル 4 を選択する。
RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。
RAPI_TIMER_S	タイマ S を選択する。
RAPI_TIMER_ON	選択したタイマを動作開始に設定する。
RAPI_TIMER_OFF	選択したタイマを動作停止に設定する。

(R8C)

RAPI_TIMER_C	タイマ C を選択する。
RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_Y	タイマ Y を選択する。
RAPI_TIMER_Z	タイマ Z を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RE	タイマ RE を選択する。
RAPI_TIMER_RF	タイマ RF を選択する。
RAPI_TIMER_ON	選択したタイマを動作開始に設定する。
RAPI_TIMER_OFF	選択したタイマを動作停止に設定する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。

RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
RAPI_TIMER_RE	タイマ RE を選択する。
RAPI_TIMER_RG	タイマ RG を選択する。
RAPI_TIMER_ON	選択したタイマを動作開始に設定する。
RAPI_TIMER_OFF	選択したタイマを動作停止に設定する。

(H8/300H)

RAPI_TIMER_V	タイマ V を選択する。
RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
RAPI_TIMER_ON	選択したタイマを動作開始に設定する。
RAPI_TIMER_OFF	選択したタイマを動作停止に設定する。

戻り値

タイマまたはインテリジェント I/O 指定が正しくない場合は RAPI_FALSE を返し、
それ以外は RAPI_TRUE を返します。

機能

タイマ(レジスタ操作)

参照

__SetTimerRegister, __ClearTimerRegister, __GetTimerRegister

備考

・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できるタイマは CPU により異なります。
・ H8/300H のタイマ V の使用でタイマの動作開始を設定する場合、本 API の直前の __SetTimerRegister で指定された、TCNTV に入力するクロックとカウント条件を設定します。

プログラム例

```
#include " rapi_timer_r8c_13.h"

void func( void )
{
    /* Activate timer C */
    __EnableTimerRegister( RAPI_TIMER_C|RAPI_TIMER_ON );
}
```

__ClearTimerRegister

要約

< タイマレジスタクリア >

Boolean __ClearTimerRegister(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定したタイマまたはインテリジェント I/O のレジスタをリセット後の値に設定します。

[data]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を選択する。
RAPI_TIMER_A1	タイマ A チャネル 1 を選択する。
RAPI_TIMER_A2	タイマ A チャネル 2 を選択する。
RAPI_TIMER_A3	タイマ A チャネル 3 を選択する。
RAPI_TIMER_A4	タイマ A チャネル 4 を選択する。
RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。
RAPI_TIMER_S	タイマ S を選択する。

(R8C)

RAPI_TIMER_C	タイマ C を選択する。
RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_Y	タイマ Y を選択する。
RAPI_TIMER_Z	タイマ Z を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RE	タイマ RE を選択する。
RAPI_TIMER_RF	タイマ RF を選択する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
RAPI_TIMER_RE	タイマ RE を選択する。
RAPI_TIMER_RG	タイマ RG を選択する。

(H8/300H)

RAPI_TIMER_A	タイマ A を選択する。
RAPI_TIMER_B1	タイマ B1 を選択する。
RAPI_TIMER_V	タイマ V を選択する。
RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。

戻り値

タイマまたはインテリジェント I/O 指定が正しくない場合は RAPI_FALSE を返し、
それ以外は RAPI_TRUE を返します。

機能

タイマ(レジスタ操作)

参照

__SetTimerRegister, __EnableTimerRegister, __GetTimerRegister

備考

・ 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できるタイマは CPU により異なります。

プログラム例

```
#include " rapi_timer_r8c_13.h"

void func( void )
{
    /* Clear the setting of timer C */
    __ClearTimerRegister( RAPI_TIMER_C );
}
```

__GetTimerRegister

要約

< タイマレジスタ値取得 >

Boolean __GetTimerRegister(unsigned long data1, unsigned int *data2)

data1	設定データ（機種により内容が異なる）
data2	レジスタ値を格納するバッファポインタ

解説

指定したタイマまたはインテリジェント I/O のレジスタ値を取得します。

[data]

data には、次の定義値を設定することができます。

(M16C)

RAPI_TIMER_A0	タイマ A チャネル 0 を選択する。
RAPI_TIMER_A1	タイマ A チャネル 1 を選択する。
RAPI_TIMER_A2	タイマ A チャネル 2 を選択する。
RAPI_TIMER_A3	タイマ A チャネル 3 を選択する。
RAPI_TIMER_A4	タイマ A チャネル 4 を選択する。
RAPI_TIMER_B0	タイマ B チャネル 0 を選択する。
RAPI_TIMER_B1	タイマ B チャネル 1 を選択する。
RAPI_TIMER_B2	タイマ B チャネル 2 を選択する。
RAPI_TIMER_S	タイマ S を選択する。

(R8C)

RAPI_TIMER_C	タイマ C を選択する。
RAPI_TIMER_X	タイマ X を選択する。
RAPI_TIMER_Y	タイマ Y を選択する。
RAPI_TIMER_Z	タイマ Z を選択する。
RAPI_TIMER_RA	タイマ RA を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RE	タイマ RE を選択する。
RAPI_TIMER_RF	タイマ RF を選択する。

(H8S)

RAPI_TIMER_RA	タイマ RA を選択する。
RAPI_TIMER_RB	タイマ RB を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。
RAPI_TIMER_RE	タイマ RE を選択する。
RAPI_TIMER_RG	タイマ RG を選択する。

(H8/300H)

RAPI_TIMER_A	タイマ A を選択する。
RAPI_TIMER_B1	タイマ B1 を選択する。
RAPI_TIMER_V	タイマ V を選択する。
RAPI_TIMER_W	タイマ W を選択する。
RAPI_TIMER_Z0	タイマ Z チャネル 0 を選択する。
RAPI_TIMER_Z1	タイマ Z チャネル 1 を選択する。
RAPI_TIMER_RC	タイマ RC を選択する。
RAPI_TIMER_RD0	タイマ RD チャネル 0 を選択する。
RAPI_TIMER_RD1	タイマ RD チャネル 1 を選択する。
RAPI_TIMER_RD2	タイマ RD チャネル 2 を選択する。
RAPI_TIMER_RD3	タイマ RD チャネル 3 を選択する。

[data2]

取得したレジスタ値を格納する配列のポインタを指定します。

配列の内容は次の通りです。

(M16C)

・タイマ A 使用時 (RAPI_TIMER_A0～RAPI_TIMER_A4 を指定時)

[0]：タイマ Ai モードレジスタ(i=0～4)の値を格納します。

[1]：タイマ Ai レジスタ(i=0～4)の値を格納します。

[2]：アップダウンフラグレジスタの値を格納します。

[3]：ワンショット開始フラグレジスタの値を格納します。

[4]：トリガ選択レジスタの値を格納します。

[5]：時計用プリスケーラリセットフラグレジスタの値を格納します。

[6]：カウント開始フラグレジスタの値を格納します。

・タイマ B 使用時 (RAPI_TIMER_B0～RAPI_TIMER_B2 を指定時)

[0]：タイマ Bi モードレジスタ(i=0～2)の値を格納します。

[1]：タイマ Bi レジスタ(i=0～2)の値を格納します。

[2]：時計用プリスケーラリセットフラグレジスタの値を格納します。

[3]：カウント開始フラグレジスタの値を格納します。

・タイマ S 使用時 (RAPI_TIMER_S を指定時)

[0]：ベースタイマレジスタの値を格納します。

[1]：ベースタイマリセットレジスタの値を格納します。

[2]：ベースタイマ制御レジスタ 0 の値を格納します。

[3]：ベースタイマ制御レジスタ 1 の値を格納します。

[4]：カウントソース分周レジスタの値を格納します。

[5]：時間計測制御レジスタ 0 の値を格納します。

[6]：時間計測制御レジスタ 1 の値を格納します。

[7]：時間計測制御レジスタ 2 の値を格納します。

[8]：時間計測制御レジスタ 3 の値を格納します。

[9]：時間計測制御レジスタ 4 の値を格納します。

[10]：時間計測制御レジスタ 5 の値を格納します。

[11]：時間計測制御レジスタ 6 の値を格納します。

[12]：時間計測制御レジスタ 7 の値を格納します。

[13]：時間計測プリスケーラレジスタ 6 の値を格納します。
[14]：時間計測プリスケーラレジスタ 7 の値を格納します。
[15]：波形生成制御レジスタ 0 の値を格納します。
[16]：波形生成制御レジスタ 1 の値を格納します。
[17]：波形生成制御レジスタ 2 の値を格納します。
[18]：波形生成制御レジスタ 3 の値を格納します。
[19]：波形生成制御レジスタ 4 の値を格納します。
[20]：波形生成制御レジスタ 5 の値を格納します。
[21]：波形生成制御レジスタ 6 の値を格納します。
[22]：波形生成制御レジスタ 7 の値を格納します。
[23]：時間計測レジスタ 0/波形生成レジスタ 0 の値を格納します。
[24]：時間計測レジスタ 1/波形生成レジスタ 1 の値を格納します。
[25]：時間計測レジスタ 2/波形生成レジスタ 2 の値を格納します。
[26]：時間計測レジスタ 3/波形生成レジスタ 3 の値を格納します。
[27]：時間計測レジスタ 4/波形生成レジスタ 4 の値を格納します。
[28]：時間計測レジスタ 5/波形生成レジスタ 5 の値を格納します。
[29]：時間計測レジスタ 6/波形生成レジスタ 6 の値を格納します。
[30]：時間計測レジスタ 7/波形生成レジスタ 7 の値を格納します。
[31]：機能選択レジスタの値を格納します。
[32]：機能許可レジスタの値を格納します。
[33]：割り込み要求レジスタの値を格納します。
[34]：割り込み許可レジスタ 0 の値を格納します。
[35]：割り込み許可レジスタ 1 の値を格納します。

(R8C)

・タイマ C 使用時 (RAPI_TIMER_C を指定時)

[0]：タイマ C レジスタの値を格納します。
[1]：キャプチャ、コンペア 0 レジスタの値を格納します。
[2]：コンペア 1 レジスタの値を格納します。
[3]：タイマ C 出力制御レジスタの値を格納します。
[4]：タイマ C 制御レジスタ 1 の値を格納します。
[5]：タイマ C 制御レジスタ 0 の値を格納します。

・タイマ X 使用時 (RAPI_TIMER_X を指定時)

[0]：タイマカウントソース設定レジスタの値を格納します。
[1]：プリスケーラ X レジスタの値を格納します。
[2]：タイマ X レジスタの値を格納します。
[3]：タイマ X モードレジスタの値を格納します。

・タイマ Y 使用時 (RAPI_TIMER_Y を指定時)

[0]：タイマカウントソース設定レジスタの値を格納します。
[1]：プリスケーラ Y の値を格納します。
[2]：タイマ Y プライマリレジスタの値を格納します。
[3]：タイマ Y セカンダリレジスタの値を格納します。
[4]：タイマ Y、Z 波形出力制御レジスタの値を格納します。
[5]：タイマ Y、Z 出力制御レジスタの値を格納します。
[6]：タイマ Y、Z モードレジスタの値を格納します。

・タイマ Z 使用時 (RAPI_TIMER_Z を指定時)

[0]：タイマカウントソース設定レジスタの値を格納します。
[1]：プリスケーラ Z レジスタの値を格納します。
[2]：タイマ Z プライマリレジスタの値を格納します。
[3]：タイマ Z セカンダリレジスタの値を格納します。
[4]：タイマ Y、Z 波形出力制御レジスタの値を格納します。
[5]：タイマ Y、Z 出力制御レジスタの値を格納します。
[6]：タイマ Y、Z モードレジスタの値を格納します。

・タイマ RA 使用時（RAPI_TIMER_RA を指定時）

[0]：タイマ RA I/O 制御レジスタの値を格納します。
[1]：タイマ RA プリスケーラレジスタの値を格納します。
[2]：タイマ RA レジスタの値を格納します。
[3]：タイマ RA モードレジスタの値を格納します。
[4]：タイマ RA 制御レジスタの値を格納します。

・タイマ RB 使用時（RAPI_TIMER_RB を指定時）

[0]：タイマ RB ワンショット制御レジスタの値を格納します。
[1]：タイマ RB I/O 制御レジスタの値を格納します。
[2]：タイマ RB プリスケーラレジスタの値を格納します。
[3]：タイマ RB プライマリレジスタの値を格納します。
[4]：タイマ RB セカンダリレジスタの値を格納します。
[5]：タイマ RB モードレジスタの値を格納します。
[6]：タイマ RB 制御レジスタの値を格納します。

・タイマ RC 使用時（RAPI_TIMER_RC を指定時）

[0]：タイマ RC ジェネラルレジスタ A の値を格納します。。
[1]：タイマ RC ジェネラルレジスタ B の値を格納します。。
[2]：タイマ RC ジェネラルレジスタ C の値を格納します。。
[3]：タイマ RC ジェネラルレジスタ D の値を格納します。。
[4]：タイマ RC デジタルフィルタ機能選択レジスタの値を格納します。。
[5]：タイマ RC 制御レジスタ 1 の値を格納します。。
[6]：タイマ RC 制御レジスタ 2 の値を格納します。。
[7]：タイマ RC I/O 制御レジスタ 0 の値を格納します。。
[8]：タイマ CD I/O 制御レジスタ 1 の値を格納します。。
[9]：タイマ RC ステータスレジスタの値を格納します。。
[10]：タイマ RC 割り込み許可レジスタの値を格納します。。
[11]：タイマ RC カウンタの値を格納します。。
[12]：タイマ RC アウトプットマスタ許可レジスタを指定します。
[13]：タイマ RC モードレジスタの値を格納します。。

・タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD1 を指定時）

[0]：タイマ RD モードレジスタの値を格納します。
[1]：タイマ RD PWM モードレジスタの値を格納します。
[2]：タイマ RD 機能制御レジスタの値を格納します。
[3]：タイマ RD ジェネラルレジスタ A i(i=0,1)の値を格納します。
[4]：タイマ RD ジェネラルレジスタ B i(i=0,1)の値を格納します。
[5]：タイマ RD ジェネラルレジスタ C i(i=0,1)の値を格納します。
[6]：タイマ RD ジェネラルレジスタ D i(i=0,1)の値を格納します。
[7]：タイマ RD デジタルフィルタ機能選択レジスタ i(i=0,1)の値を格納します。

[8]：タイマ RD 制御レジスタ i(i=0,1)の値を格納します。
[9]：タイマ RD I/O 制御レジスタ Ai(i=0,1)の値を格納します。
[10]：タイマ RD I/O 制御レジスタ Ci(i=0,1)の値を格納します。
[11]：タイマ RD ステータスレジスタ i(i=0,1)の値を格納します。
[12]：タイマ RD 割り込み許可レジスタ i(i=0,1)の値を格納します。
[13]：タイマ RD カウンタ i(i=0,1)の値を格納します。
[14]：タイマ RD スタートレジスタの値を格納します。
[15]：タイマ RD アウトプットマスタ許可レジスタ 1 の値を格納します。
[16]：タイマ RD アウトプットマスタ許可レジスタ 2 の値を格納します。
[17]：タイマ RD アウトプット制御レジスタの値を格納します。

・タイマ RE 使用時 (RAPI_TIMER_RE を指定時)

[0]：タイマ RE 秒データレジスタの値を格納します。
[1]：タイマ RE 分データレジスタの値を格納します。
[2]：タイマ RE 時データレジスタの値を格納します。
[3]：タイマ RE 曜日データレジスタの値を格納します。
[4]：タイマ RE 制御レジスタ 2 の値を格納します。
[5]：タイマ RE カウントソース選択レジスタの値を格納します。
[6]：タイマ RE 制御レジスタ 1 の値を格納します。

・タイマ RF 使用時 (RAPI_TIMER_RF を指定時)

[0]：タイマ RF 制御レジスタ 0 の値を格納します。。
[1]：キャプチャ、コンペア 0 レジスタの値を格納します。。
[2]：コンペア 1 レジスタの値を格納します。。
[3]：タイマ RF 出力制御レジスタ 0 の値を格納します。。
[4]：タイマ RF 制御レジスタ 1 の値を格納します。。

(H8S)

・タイマ RA 使用時 (RAPI_TIMER_RA を指定時)

[0]：タイマ RA I/O コントロールレジスタの値を格納します。
[1]：タイマ RA プリスケーラレジスタの値を格納します。
[2]：タイマ RA タイマレジスタの値を格納します。
[3]：タイマ RA モードレジスタの値を格納します。
[4]：タイマ RA 割り込み要求ステータスレジスタの値を格納します。
[5]：タイマ RA コントロールレジスタの値を格納します。

・タイマ RB 使用時 (RAPI_TIMER_RB を指定時)

[0]：タイマ RB ワンショットコントロールレジスタの値を格納します。
[1]：タイマ RB I/O コントロールレジスタの値を格納します。
[2]：タイマ RB プリスケーラレジスタの値を格納します。
[3]：タイマ RB プライマリレジスタの値を格納します。
[4]：タイマ RB セカンダリレジスタの値を格納します。
[5]：タイマ RB モードレジスタの値を格納します。
[6]：タイマ RB 割り込み要求ステータスレジスタの値を格納します。
[7]：タイマ RB コントロールレジスタの値を格納します。

・タイマ RC 使用時 (RAPI_TIMER_RC を指定時)

[0]：タイマ RC コントロールレジスタ 1 の値を格納します。
[1]：タイマ RC コントロールレジスタ 2 の値を格納します。
[2]：タイマ RC インタラプトイネーブルレジスタの値を格納します。

[3]：タイマ RC ステータスレジスタの値を格納します。
[4]：タイマ RC I/O コントロールレジスタ 0 の値を格納します。
[5]：タイマ RC I/O コントロールレジスタ 1 の値を格納します。
[6]：タイマ RC アウトプットイネーブルレジスタの値を格納します。
[7]：タイマ RC デジタルフィルタ機能選択レジスタの値を格納します。
[8]：タイマ RC A/D 変換トリガコントロールレジスタの値を格納します。
[9]：タイマ RC カウンタの値を格納します。
[10]：ジェネラルレジスタ A の値を格納します。
[11]：ジェネラルレジスタ B の値を格納します。
[12]：ジェネラルレジスタ C の値を格納します。
[13]：ジェネラルレジスタ D の値を格納します。
[14]：タイマ RC モードレジスタの値を格納します。

・タイマ RD 使用時（RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時）

[0]：タイマ RD モードレジスタの値を格納します。
[1]：タイマ RD PWM モードレジスタの値を格納します。
[2]：タイマ RD ファンクションコントロールレジスタの値を格納します。
[3]：タイマ RD アウトプットマスタイネーブルレジスタ 1 の値を格納します。
[4]：タイマ RD アウトプットマスタイネーブルレジスタ 2 の値を格納します。
[5]：タイマ RD アウトプットコントロールレジスタの値を格納します。
[6]：タイマ RD A/D 変換開始トリガコントロールレジスタの値を格納します。
[7]：タイマ RD カウンタ_i(i=0,1)の値を格納します。
[8]：ジェネラルレジスタ A_i(i=0,1)の値を格納します。
[9]：ジェネラルレジスタ B_i(i=0,1)の値を格納します。
[10]：ジェネラルレジスタ C_i(i=0,1)の値を格納します。
[11]：ジェネラルレジスタ D_i(i=0,1)の値を格納します。
[12]：タイマ RD コントロールレジスタ_i(i=0,1)の値を格納します。
[13]：タイマ RD I/O コントロールレジスタ A_i(i=0,1)の値を格納します。
[14]：タイマ RD I/O コントロールレジスタ C_i(i=0,1)の値を格納します。
[15]：タイマ RD ステータスレジスタ_i(i=0,1)の値を格納します。
[16]：タイマ RD インタラプトイネーブルレジスタ_i(i=0,1)の値を格納します。
[17]：PWM モードアウトプットレベルコントロールレジスタ_i(i=0,1)の値を格納します。
[18]：タイマ RD デジタルフィルタ機能選択レジスタ i(i=0,1)の値を格納します。
[19]：タイマ RD スタートレジスタの値を格納します。

・タイマ RE 使用時（RAPI_TIMER_RE を指定時）

[0]：タイマ RE 秒データレジスタ/カウンタデータレジスタの値を格納します。
[1]：タイマ RE 分データレジスタ/コンペアデータレジスタの値を格納します。
[2]：タイマ RE 時データレジスタの値を格納します。
[3]：タイマ RE 曜日データレジスタの値を格納します。
[4]：タイマ RE コントロールレジスタ 2 の値を格納します。
[5]：タイマ RE クロックソースセレクトレジスタの値を格納します。
[6]：タイマ R 割り込みフラグレジスタの値を格納します。
[7]：タイマ RE コントロールレジスタ 1 の値を格納します。

・タイマ RG 使用時（RAPI_TIMER_RG を指定時）

[0]：タイマ RG カウンタコントロールレジスタの値を格納します。
[1]：タイマ RG コントロールレジスタの値を格納します。

[2]：タイマ RG I/O コントロールレジスタの値を格納します。

[3]：タイマ RG ステータスレジスタの値を格納します。

[4]：タイマ RG インタラプトイネーブルレジスタの値を格納します。

[5]：タイマ RG カウンタの値を格納します。

[6]：ジェネラルレジスタ A の値を格納します。

[7]：ジェネラルレジスタ B の値を格納します。

[8]：GRA バッファレジスタの値を格納します。

[9]：GRB バッファレジスタの値を格納します。

[10]：タイマ RG モードレジスタの値を格納します。

(H8/300H)

・タイマ A 使用時 (RAPI_TIMER_A を指定時)

[0]：タイマモードレジスタ A の値を格納します。

[1]：タイマカウンタ A の値を格納します。

・タイマ B1 使用時 (RAPI_TIMER_B1 を指定時)

[0]：タイマモードレジスタ B1 の値を格納します。

[1]：タイマカウンタ B1 の値を格納します。

・タイマ V 使用時 (RAPI_TIMER_V を指定時)

[0]：タイマカウンタ V の値を格納します。

[1]：タイムコンスタントレジスタ A の値を格納します。

[2]：タイムコンスタントレジスタ B の値を格納します。

[3]：タイムコントロールレジスタ V0 の値を格納します。

[4]：タイムコントロールレジスタ V1 の値を格納します。

[5]：タイマコントロール/ステータスレジスタ V の値を格納します。

・タイマ W 使用時 (RAPI_TIMER_W を指定時)

[0]：タイマモードレジスタ W の値を格納します。

[1]：タイマコントロールレジスタ W の値を格納します。

[2]：タイマインタラプトマスタイネーブルレジスタ W の値を格納します。

[3]：タイマステータスレジスタ W の値を格納します。

[4]：タイマ I/O コントロールレジスタ 0 の値を格納します。

[5]：タイマ I/O コントロールレジスタ 1 の値を格納します。

[6]：タイマカウンタの値を格納します。

[7]：ジェネラルレジスタ A の値を格納します。

[8]：ジェネラルレジスタ B の値を格納します。

[9]：ジェネラルレジスタ C の値を格納します。

[10]：ジェネラルレジスタ D の値を格納します。

・タイマ Z 使用時 (RAPI_TIMER_Z0～RAPI_TIMER_Z1 を指定時)

[0]：タイマスタートレジスタの値を格納します。

[1]：タイマモードレジスタの値を格納します。

[2]：タイマ PWM モードレジスタの値を格納します。

[3]：タイマファンクションコントロールレジスタの値を格納します。

[4]：タイマアウトプットマスタイネーブルレジスタの値を格納します。

[5]：タイマアウトプットコントロールレジスタの値を格納します。

[6]：タイマカウンタ_i(i=0,1)の値を格納します。

[7]：ジェネラルレジスタ A_i(i=0,1)の値を格納します。

[8]：ジェネラルレジスタ B_i(i=0,1)の値を格納します。

[9]：ジェネラルレジスタ C_i(i=0,1)の値を格納します。
[10]：ジェネラルレジスタ D_i(i=0,1)の値を格納します。
[11]：タイマコントロールレジスタ_i(i=0,1)の値を格納します。
[12]：タイマ I/O コントロールレジスタ A_i(i=0,1)の値を格納します。
[13]：タイマ I/O コントロールレジスタ B_i(i=0,1)の値を格納します。
[14]：タイマステータスレジスタ_i(i=0,1)の値を格納します。
[15]：タイマインタラプトイネーブルレジスタ_i(i=0,1)の値を格納します。
[16]：PWM モードアウトプットレベルコントロールレジスタ_i(i=0,1)の値を格納します。

・タイマ RC 使用時 (RAPI_TIMER_RC を指定時)

[0]：タイマ RC コントロールレジスタ 1 の値を格納します。
[1]：タイマ RC コントロールレジスタ 2 の値を格納します。
[2]：タイマ RC インタラプトイネーブルレジスタの値を格納します。
[3]：タイマ RC ステータスレジスタの値を格納します。
[4]：タイマ RC I/O コントロールレジスタ 0 の値を格納します。
[5]：タイマ RC I/O コントロールレジスタ 1 の値を格納します。
[6]：タイマ RC アウトプットイネーブルレジスタの値を格納します。
[7]：タイマ RC デジタルフィルタ機能選択レジスタの値を格納します。
[8]：タイマ RC カウンタの値を格納します。
[9]：ジェネラルレジスタ A の値を格納します。
[10]：ジェネラルレジスタ B の値を格納します。
[11]：ジェネラルレジスタ C の値を格納します。
[12]：ジェネラルレジスタ D の値を格納します。
[13]：タイマ RC モードレジスタの値を格納します。

・タイマ RD 使用時 (RAPI_TIMER_RD0～RAPI_TIMER_RD3 を指定時)

[0]：タイマ RD モードレジスタの値を格納します。
[1]：タイマ RD PWM モードレジスタの値を格納します。
[2]：タイマ RD ファンクションコントロールレジスタの値を格納します。
[3]：タイマ RD アウトプットマスタイネーブルレジスタ 1 の値を格納します。
[4]：タイマ RD アウトプットマスタイネーブルレジスタ 2 の値を格納します。
[5]：タイマ RD アウトプットコントロールレジスタの値を格納します。
[6]：タイマ RD カウンタ_i(i=0,1)の値を格納します。
[7]：ジェネラルレジスタ A_i(i=0,1)の値を格納します。
[8]：ジェネラルレジスタ B_i(i=0,1)の値を格納します。
[9]：ジェネラルレジスタ C_i(i=0,1)の値を格納します。
[10]：ジェネラルレジスタ D_i(i=0,1)の値を格納します。
[11]：タイマ RD コントロールレジスタ_i(i=0,1)の値を格納します。
[12]：タイマ RD I/O コントロールレジスタ A_i(i=0,1)の値を格納します。
[13]：タイマ RD I/O コントロールレジスタ C_i(i=0,1)の値を格納します。
[14]：タイマ RD ステータスレジスタ_i(i=0,1)の値を格納します。
[15]：タイマ RD インタラプトイネーブルレジスタ_i(i=0,1)の値を格納します。
[16]：PWM モードアウトプットレベルコントロールレジスタ_i(i=0,1)の値を格納します。
[17]：タイマ RD デジタルフィルタ機能選択レジスタ i(i=0,1)の値を格納します。
[18]：タイマ RD スタートレジスタの値を格納します。

戻り値

タイマまたはインテリジェント I/O 指定が正しくない場合は RAPI_FALSE を返し、
それ以外は RAPI_TRUE を返します。

機能

タイマ(レジスタ操作)

参照

__SetTimerRegister, __EnableTimerRegister, __ClearTimerRegister

備考

・第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・指定できるタイマは CPU により異なります。

プログラム例

```
#include " rapi_timer_r8c_13.h"

void func( void )
{
   unsigned int data[7];

   /* Get the value of timer Z registers */
    __GetTimerRegister( RAPI_TIMER_Z, data );
}
```

4.2.3 I/Oポート
__SetIOPort

要約

< I/O ポートの設定 >

Boolean __SetIOPort(unsigned long data1, unsigned int data2)

data1	設定データ 1（機種により内容が異なる）
data2	設定データ 2（機種により内容が異なる）

解説

指定した I/O ポートの動作条件設定をおこないます。

[data1]

data1 には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。ただし、複数のポートを同時に指定することはできません。

(M16C)

各 I/O ポートに対応する定義値は次の通りです。

RAPI_PORT_0_0	ポート $P0_0$	RAPI_PORT_0_1	ポート $P0_1$
RAPI_PORT_0_2	ポート $P0_2$	RAPI_PORT_0_3	ポート $P0_3$
RAPI_PORT_0_4	ポート $P0_4$	RAPI_PORT_0_5	ポート $P0_5$
RAPI_PORT_0_6	ポート $P0_6$	RAPI_PORT_0_7	ポート $P0_7$
RAPI_PORT_1_0	ポート $P1_0$	RAPI_PORT_1_1	ポート $P1_1$
RAPI_PORT_1_2	ポート $P1_2$	RAPI_PORT_1_3	ポート $P1_3$
RAPI_PORT_1_4	ポート $P1_4$	RAPI_PORT_1_5	ポート $P1_5$
RAPI_PORT_1_6	ポート $P1_6$	RAPI_PORT_1_7	ポート $P1_7$
RAPI_PORT_2_0	ポート $P2_0$	RAPI_PORT_2_1	ポート $P2_1$
RAPI_PORT_2_2	ポート $P2_2$	RAPI_PORT_2_3	ポート $P2_3$
RAPI_PORT_2_4	ポート $P2_4$	RAPI_PORT_2_5	ポート $P2_5$
RAPI_PORT_2_6	ポート $P2_6$	RAPI_PORT_2_7	ポート $P2_7$
RAPI_PORT_3_0	ポート $P3_0$	RAPI_PORT_3_1	ポート $P3_1$
RAPI_PORT_3_2	ポート $P3_2$	RAPI_PORT_3_3	ポート $P3_3$
RAPI_PORT_3_4	ポート $P3_4$	RAPI_PORT_3_5	ポート $P3_5$
RAPI_PORT_3_6	ポート $P3_6$	RAPI_PORT_3_7	ポート $P3_7$
RAPI_PORT_6_0	ポート $P6_0$	RAPI_PORT_6_1	ポート $P6_1$
RAPI_PORT_6_2	ポート $P6_2$	RAPI_PORT_6_3	ポート $P6_3$
RAPI_PORT_6_4	ポート $P6_4$	RAPI_PORT_6_5	ポート $P6_5$
RAPI_PORT_6_6	ポート $P6_6$	RAPI_PORT_6_7	ポート $P6_7$
RAPI_PORT_7_0	ポート $P7_0$	RAPI_PORT_7_1	ポート $P7_1$
RAPI_PORT_7_2	ポート $P7_2$	RAPI_PORT_7_3	ポート $P7_3$
RAPI_PORT_7_4	ポート $P7_4$	RAPI_PORT_7_5	ポート $P7_5$
RAPI_PORT_7_6	ポート $P7_6$	RAPI_PORT_7_7	ポート $P7_7$
RAPI_PORT_8_0	ポート $P8_0$	RAPI_PORT_8_1	ポート $P8_1$
RAPI_PORT_8_2	ポート $P8_2$	RAPI_PORT_8_3	ポート $P8_3$
RAPI_PORT_8_4	ポート $P8_4$	RAPI_PORT_8_5	ポート $P8_5$
RAPI_PORT_8_6	ポート $P8_6$	RAPI_PORT_8_7	ポート $P8_7$

RAPI_PORT_9_0	ポート $P9_0$	RAPI_PORT_9_1	ポート $P9_1$
RAPI_PORT_9_2	ポート $P9_2$	RAPI_PORT_9_3	ポート $P9_3$
RAPI_PORT_9_5	ポート $P9_5$	RAPI_PORT_9_6	ポート $P9_6$
RAPI_PORT_9_7	ポート $P9_7$	RAPI_PORT_10_0	ポート $P10_0$
RAPI_PORT_10_1	ポート $P10_1$	RAPI_PORT_10_2	ポート $P10_2$
RAPI_PORT_10_3	ポート $P10_3$	RAPI_PORT_10_4	ポート $P10_4$
RAPI_PORT_10_5	ポート $P10_5$	RAPI_PORT_10_6	ポート $P10_6$
RAPI_PORT_10_7	ポート $P10_7$		

設定に関する定義値は次の通りです。

RAPI_PORT_INPUT	選択したポートを入力ポートに設定する。
RAPI_PORT_OUTPUT	選択したポートを出力ポートに設定する。
RAPI_PULLED_HIGH	選択したポートをプルアップありに設定する。
RAPI_NOT_PULLED _HIGH	選択したポートをプルアップなしに設定する。
RAPI_LATCH	入力ポート設定、出力ポート設定にかかわらず、ポートラッチを読むように設定する。ポート P1 使用時のみ指定可能。

(R8C)

各 I/O ポートに対応する定義値は次の通りです。

RAPI_PORT_0_0	ポート $P0_0$	RAPI_PORT_0_1	ポート $P0_1$
RAPI_PORT_0_2	ポート $P0_2$	RAPI_PORT_0_3	ポート $P0_3$
RAPI_PORT_0_4	ポート $P0_4$	RAPI_PORT_0_5	ポート $P0_5$
RAPI_PORT_0_6	ポート $P0_6$	RAPI_PORT_0_7	ポート $P0_7$
RAPI_PORT_1_0	ポート $P1_0$	RAPI_PORT_1_1	ポート $P1_1$
RAPI_PORT_1_2	ポート $P1_2$	RAPI_PORT_1_3	ポート $P1_3$
RAPI_PORT_1_4	ポート $P1_4$	RAPI_PORT_1_5	ポート $P1_5$
RAPI_PORT_1_6	ポート $P1_6$。	RAPI_PORT_1_7	ポート $P1_7$
RAPI_PORT_2_0	ポート $P2_0$	RAPI_PORT_2_1	ポート $P2_1$
RAPI_PORT_2_2	ポート $P2_2$	RAPI_PORT_2_3	ポート $P2_3$
RAPI_PORT_2_4	ポート $P2_4$	RAPI_PORT_2_5	ポート $P2_5$
RAPI_PORT_2_6	ポート $P2_6$。	RAPI_PORT_2_7	ポート $P2_7$
RAPI_PORT_3_0	ポート $P3_0$	RAPI_PORT_3_1	ポート $P3_1$
RAPI_PORT_3_2	ポート $P3_2$	RAPI_PORT_3_3	ポート $P3_3$
RAPI_PORT_3_4	ポート $P3_4$	RAPI_PORT_3_5	ポート $P3_5$
RAPI_PORT_3_6	ポート $P3_6$	RAPI_PORT_3_7	ポート $P3_7$
RAPI_PORT_4_3	ポート $P4_3$	RAPI_PORT_4_4	ポート $P4_4$
RAPI_PORT_4_5	ポート $P4_5$	RAPI_PORT_5_0	ポート $P5_0$
RAPI_PORT_5_1	ポート $P5_1$	RAPI_PORT_5_2	ポート $P5_2$
RAPI_PORT_5_3	ポート $P5_3$	RAPI_PORT_5_4	ポート $P5_4$
RAPI_PORT_5_5	ポート $P5_5$	RAPI_PORT_5_6	ポート $P5_6$
RAPI_PORT_5_7	ポート $P5_7$	RAPI_PORT_6_0	ポート $P6_0$
RAPI_PORT_6_1	ポート $P6_1$	RAPI_PORT_6_2	ポート $P6_2$
RAPI_PORT_6_3	ポート $P6_3$	RAPI_PORT_6_4	ポート $P6_4$
RAPI_PORT_6_5	ポート $P6_5$	RAPI_PORT_6_6	ポート $P6_6$

RAPI_PORT_6_7	ポート $P6_7$	RAPI_PORT_7_0	ポート $P7_0$
RAPI_PORT_7_1	ポート $P7_1$	RAPI_PORT_7_2	ポート $P7_2$
RAPI_PORT_7_3	ポート $P7_3$	RAPI_PORT_7_4	ポート $P7_4$
RAPI_PORT_7_5	ポート $P7_5$	RAPI_PORT_7_6	ポート $P7_6$
RAPI_PORT_7_7	ポート $P7_7$	RAPI_PORT_8_0	ポート $P8_0$
RAPI_PORT_8_1	ポート $P8_1$	RAPI_PORT_8_2	ポート $P8_2$
RAPI_PORT_8_3	ポート $P8_3$	RAPI_PORT_8_4	ポート $P8_4$
RAPI_PORT_8_5	ポート $P8_5$	RAPI_PORT_8_6	ポート $P8_6$
RAPI_PORT_8_7	ポート $P8_7$	RAPI_PORT_9_0	ポート $P9_0$
RAPI_PORT_9_1	ポート $P9_1$	RAPI_PORT_9_2	ポート $P9_2$
RAPI_PORT_9_3	ポート $P9_3$		

設定に関する定義値は次の通りです。

RAPI_PORT_INPUT	選択したポートを入力ポートに設定する。
RAPI_PORT_OUTPUT	選択したポートを出力ポートに設定する。
RAPI_PULLED_HIGH	選択したポートをプルアップありに設定する。
RAPI_NOT_PULLED_HIGH	選択したポートをプルアップなしに設定する。
RAPI_DRIVE_CAPACITY_H	選択したポートの N チャネル出力トランジスタの駆動能力を High に設定する。
RAPI_DRIVE_CAPACITY_L	選択したポートの N チャネル出力トランジスタの駆動能力を Low に設定する。

(H8S)

各 I/O ポートに対応する定義値は次の通りです。

RAPI_PORT_1_0	ポート P10	RAPI_PORT_1_1	ポート P11
RAPI_PORT_1_2	ポート P12	RAPI_PORT_1_3	ポート P13
RAPI_PORT_1_4	ポート P14	RAPI_PORT_1_5	ポート P15
RAPI_PORT_1_6	ポート P16	RAPI_PORT_1_7	ポート P17
RAPI_PORT_2_0	ポート P20	RAPI_PORT_2_1	ポート P21
RAPI_PORT_2_2	ポート P22	RAPI_PORT_2_3	ポート P23
RAPI_PORT_2_4	ポート P24	RAPI_PORT_2_5	ポート P25
RAPI_PORT_2_6	ポート P26	RAPI_PORT_2_7	ポート P27
RAPI_PORT_3_0	ポート P30	RAPI_PORT_3_1	ポート P31
RAPI_PORT_3_2	ポート P32	RAPI_PORT_3_3	ポート P33
RAPI_PORT_3_4	ポート P34	RAPI_PORT_3_5	ポート P35
RAPI_PORT_3_6	ポート P36	RAPI_PORT_3_7	ポート P37
RAPI_PORT_5_0	ポート P50	RAPI_PORT_5_1	ポート P51
RAPI_PORT_5_2	ポート P52	RAPI_PORT_5_3	ポート P53
RAPI_PORT_5_4	ポート P54	RAPI_PORT_5_5	ポート P55
RAPI_PORT_5_6	ポート P56	RAPI_PORT_5_7	ポート P57
RAPI_PORT_6_0	ポート P60	RAPI_PORT_6_1	ポート P61
RAPI_PORT_6_2	ポート P62	RAPI_PORT_6_3	ポート P63
RAPI_PORT_6_4	ポート P64	RAPI_PORT_6_5	ポート P65
RAPI_PORT_6_6	ポート P66	RAPI_PORT_6_7	ポート P67
RAPI_PORT_8_5	ポート P85	RAPI_PORT_8_6	ポート P86

RAPI_PORT_8_7	ポート P87	RAPI_PORT_9_0	ポート P90
RAPI_PORT_9_1	ポート P91	RAPI_PORT_9_2	ポート P92
RAPI_PORT_9_3	ポート P93	RAPI_PORT_9_4	ポート P94
RAPI_PORT_9_5	ポート P95	RAPI_PORT_9_6	ポート P96
RAPI_PORT_9_7	ポート P97	RAPI_PORT_A_0	ポート PA0
RAPI_PORT_A_1	ポート PA1	RAPI_PORT_A_2	ポート PA2
RAPI_PORT_A_3	ポート PA3	RAPI_PORT_A_4	ポート PA4
RAPI_PORT_A_5	ポート PA5	RAPI_PORT_A_6	ポート PA6
RAPI_PORT_A_7	ポート PA7	RAPI_PORT_B_0	ポート PB0
RAPI_PORT_B_1	ポート PB1	RAPI_PORT_B_2	ポート PB2
RAPI_PORT_B_3	ポート PB3	RAPI_PORT_B_4	ポート PB4
RAPI_PORT_B_5	ポート PB5	RAPI_PORT_B_6	ポート PB6
RAPI_PORT_B_7	ポート PB7	RAPI_PORT_J_0	ポート PJ0
RAPI_PORT_J_1	ポート PJ1		

設定に関する定義値は次の通りです。

RAPI_PORT_INPUT	選択したポートを入力ポートに設定する。
RAPI_PORT_OUTPUT	選択したポートを出力ポートに設定する。
RAPI_PULLED_HIGH	選択したポートをプルアップありに設定する。
RAPI_NOT_PULLED_HIGH	選択したポートをプルアップなしに設定する。
RAPI_DRIVE_CAPACITY_H	選択したポートの駆動能力を高駆動出力に設定する。

(H8/300H)

各 I/O ポートに対応する定義値は次の通りです。

RAPI_PORT_1_0	ポート P10	RAPI_PORT_1_1	ポート P11
RAPI_PORT_1_2	ポート P12	RAPI_PORT_1_4	ポート P14
RAPI_PORT_1_5	ポート P15	RAPI_PORT_1_6	ポート P16
RAPI_PORT_1_7	ポート P17	RAPI_PORT_2_0	ポート P20
RAPI_PORT_2_1	ポート P21	RAPI_PORT_2_2	ポート P22
RAPI_PORT_2_3	ポート P23	RAPI_PORT_2_4	ポート P24
RAPI_PORT_2_5	ポート P25	RAPI_PORT_2_6	ポート P26
RAPI_PORT_2_7	ポート P27	RAPI_PORT_3_0	ポート P30
RAPI_PORT_3_1	ポート P31	RAPI_PORT_3_2	ポート P32
RAPI_PORT_3_3	ポート P33	RAPI_PORT_3_4	ポート P34
RAPI_PORT_3_5	ポート P35	RAPI_PORT_3_6	ポート P36
RAPI_PORT_3_7	ポート P37	RAPI_PORT_5_0	ポート P50
RAPI_PORT_5_1	ポート P51	RAPI_PORT_5_2	ポート P52
RAPI_PORT_5_3	ポート P53	RAPI_PORT_5_4	ポート P54
RAPI_PORT_5_5	ポート P55	RAPI_PORT_5_6	ポート P56
RAPI_PORT_5_7	ポート P57	RAPI_PORT_6_0	ポート P60
RAPI_PORT_6_1	ポート P61	RAPI_PORT_6_2	ポート P62
RAPI_PORT_6_3	ポート P63	RAPI_PORT_6_4	ポート P64
RAPI_PORT_6_5	ポート P65	RAPI_PORT_6_6	ポート P66
RAPI_PORT_6_7	ポート P67	RAPI_PORT_7_0	ポート P70
RAPI_PORT_7_1	ポート P71	RAPI_PORT_7_2	ポート P72

RAPI_PORT_7_4	ポート P74	RAPI_PORT_7_5	ポート P75
RAPI_PORT_7_6	ポート P76	RAPI_PORT_7_7	ポート P77
RAPI_PORT_8_0	ポート P80	RAPI_PORT_8_1	ポート P81
RAPI_PORT_8_2	ポート P82	RAPI_PORT_8_3	ポート P83
RAPI_PORT_8_4	ポート P84	RAPI_PORT_8_5	ポート P85
RAPI_PORT_8_6	ポート P86	RAPI_PORT_8_7	ポート P87
RAPI_PORT_9_0	ポート P90	RAPI_PORT_9_1	ポート P91
RAPI_PORT_9_2	ポート P92	RAPI_PORT_9_3	ポート P93
RAPI_PORT_9_4	ポート P94	RAPI_PORT_9_5	ポート P95
RAPI_PORT_9_6	ポート P96	RAPI_PORT_9_7	ポート P97
RAPI_PORT_C_0	ポート PC0	RAPI_PORT_C_1	ポート PC1
RAPI_PORT_C_2	ポート PC2	RAPI_PORT_C_3	ポート PC3
RAPI_PORT_D_0	ポート PD0	RAPI_PORT_D_1	ポート PD1
RAPI_PORT_D_2	ポート PD2	RAPI_PORT_D_3	ポート PD3
RAPI_PORT_D_4	ポート PD4	RAPI_PORT_D_5	ポート PD5
RAPI_PORT_D_6	ポート PD6	RAPI_PORT_D_7	ポート PD7
RAPI_PORT_E_0	ポート PE0	RAPI_PORT_E_1	ポート PE1
RAPI_PORT_E_2	ポート PE2	RAPI_PORT_E_3	ポート PE3
RAPI_PORT_E_4	ポート PE4	RAPI_PORT_E_5	ポート PE5
RAPI_PORT_E_6	ポート PE6	RAPI_PORT_E_7	ポート PE7
RAPI_PORT_F_0	ポート PF0	RAPI_PORT_G_0	ポート PG0
RAPI_PORT_G_1	ポート PG1	RAPI_PORT_G_2	ポート PG2
RAPI_PORT_G_3	ポート PG3	RAPI_PORT_G_4	ポート PG4
RAPI_PORT_G_5	ポート PG5	RAPI_PORT_G_6	ポート PG6
RAPI_PORT_G_7	ポート PG7	RAPI_PORT_H_0	ポート PH0
RAPI_PORT_H_1	ポート PH1	RAPI_PORT_H_2	ポート PH2
RAPI_PORT_H_3	ポート PH3	RAPI_PORT_H_4	ポート PH4
RAPI_PORT_H_5	ポート PH5	RAPI_PORT_H_6	ポート PH6
RAPI_PORT_H_7	ポート PH7	RAPI_PORT_J_0	ポート PJ0
RAPI_PORT_J_1	ポート PJ1		

設定に関する定義値は次の通りです。

RAPI_PORT_INPUT	選択したポートを入力ポートに設定する。
RAPI_PORT_OUTPUT	選択したポートを出力ポートに設定する。
RAPI_PULLED_HIGH	選択したポートをプルアップありに設定する。
RAPI_NOT_PULLED_HIGH	選択したポートをプルアップなしに設定する。

[data2]

(M16C)

_NMI/_SD に割り当てられたデジタルデバウンス機能のデジタルフィルタ幅を指定します。ポート P85 使用時のみ指定可能。

INPC17/_INT5 に割り当てられたデジタルデバウンス機能のデジタルフィルタ幅を指定します。ポート P17 使用時のみ指定可能。

上記以外のポート使用時は 0 を設定してください。

(R8C) (H8S) (H8/300H)
　0 を指定します。

戻り値　I/O ポート指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能　I/O ポート

参照　__ReadIOPort, __WriteIOPort, __SetIOPortRegister, __ReadIOPortRegister, __WriteIOPortRegister

備考
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できる I/O ポートは CPU により異なります。
- 指定した I/O ポートが有しない機能を設定することはできません。

プログラム例

```
#include " rapi_io_port_r8c_13.h"

void func( void )
{
    /* Set up port P03 as input port */
    __SetIOPort(RAPI_PORT_0_3| RAPI_PORT_INPUT| RAPI_PULLED_HIGH, 0, 0 );
}
```

__ReadIOPort

要約

< I/O ポートからの読み出し >

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

data1	設定データ 1（機種により内容が異なる）
data2	I/O ポートから読み出した値を格納する変数のポインタ

解説

指定した I/O ポートの値を取得します。

[data1]

データを読み出す I/O ポートを指定します。各 I/O ポートに対応する定義値は次の通りです。

(M16C)

RAPI_PORT_0_0	ポート $P0_0$	RAPI_PORT_0_1	ポート $P0_1$
RAPI_PORT_0_2	ポート $P0_2$	RAPI_PORT_0_3	ポート $P0_3$
RAPI_PORT_0_4	ポート $P0_4$	RAPI_PORT_0_5	ポート $P0_5$
RAPI_PORT_0_6	ポート $P0_6$	RAPI_PORT_0_7	ポート $P0_7$
RAPI_PORT_1_0	ポート $P1_0$	RAPI_PORT_1_1	ポート $P1_1$
RAPI_PORT_1_2	ポート $P1_2$	RAPI_PORT_1_3	ポート $P1_3$
RAPI_PORT_1_4	ポート $P1_4$	RAPI_PORT_1_5	ポート $P1_5$
RAPI_PORT_1_6	ポート $P1_6$	RAPI_PORT_1_7	ポート $P1_7$
RAPI_PORT_2_0	ポート $P2_0$	RAPI_PORT_2_1	ポート $P2_1$
RAPI_PORT_2_2	ポート $P2_2$	RAPI_PORT_2_3	ポート $P2_3$
RAPI_PORT_2_4	ポート $P2_4$	RAPI_PORT_2_5	ポート $P2_5$
RAPI_PORT_2_6	ポート $P2_6$	RAPI_PORT_2_7	ポート $P2_7$
RAPI_PORT_3_0	ポート $P3_0$	RAPI_PORT_3_1	ポート $P3_1$
RAPI_PORT_3_2	ポート $P3_2$	RAPI_PORT_3_3	ポート $P3_3$
RAPI_PORT_3_4	ポート $P3_4$	RAPI_PORT_3_5	ポート $P3_5$
RAPI_PORT_3_6	ポート $P3_6$	RAPI_PORT_3_7	ポート $P3_7$
RAPI_PORT_6_0	ポート $P6_0$	RAPI_PORT_6_1	ポート $P6_1$
RAPI_PORT_6_2	ポート $P6_2$	RAPI_PORT_6_3	ポート $P6_3$
RAPI_PORT_6_4	ポート $P6_4$	RAPI_PORT_6_5	ポート $P6_5$
RAPI_PORT_6_6	ポート $P6_6$	RAPI_PORT_6_7	ポート $P6_7$
RAPI_PORT_7_0	ポート $P7_0$	RAPI_PORT_7_1	ポート $P7_1$
RAPI_PORT_7_2	ポート $P7_2$	RAPI_PORT_7_3	ポート $P7_3$
RAPI_PORT_7_4	ポート $P7_4$	RAPI_PORT_7_5	ポート $P7_5$
RAPI_PORT_7_6	ポート $P7_6$	RAPI_PORT_7_7	ポート $P7_7$
RAPI_PORT_8_0	ポート $P8_0$	RAPI_PORT_8_1	ポート $P8_1$
RAPI_PORT_8_2	ポート $P8_2$	RAPI_PORT_8_3	ポート $P8_3$
RAPI_PORT_8_4	ポート $P8_4$	RAPI_PORT_8_5	ポート $P8_5$
RAPI_PORT_8_6	ポート $P8_6$	RAPI_PORT_8_7	ポート $P8_7$
RAPI_PORT_9_0	ポート $P9_0$	RAPI_PORT_9_1	ポート $P9_1$
RAPI_PORT_9_2	ポート $P9_2$	RAPI_PORT_9_3	ポート $P9_3$
RAPI_PORT_9_5	ポート $P9_5$	RAPI_PORT_9_6	ポート $P9_6$

RAPI_PORT_9_7	ポート $P9_7$	RAPI_PORT_10_0	ポート $P10_0$
RAPI_PORT_10_1	ポート $P10_1$	RAPI_PORT_10_2	ポート $P10_2$
RAPI_PORT_10_3	ポート $P10_3$	RAPI_PORT_10_4	ポート $P10_4$
RAPI_PORT_10_5	ポート $P10_5$	RAPI_PORT_10_6	ポート $P10_6$
RAPI_PORT_10_7	ポート $P10_7$		

(R8C)

RAPI_PORT_0_0	ポート $P0_0$	RAPI_PORT_0_1	ポート $P0_1$
RAPI_PORT_0_2	ポート $P0_2$	RAPI_PORT_0_3	ポート $P0_3$
RAPI_PORT_0_4	ポート $P0_4$	RAPI_PORT_0_5	ポート $P0_5$
RAPI_PORT_0_6	ポート $P0_6$	RAPI_PORT_0_7	ポート $P0_7$
RAPI_PORT_1_0	ポート $P1_0$	RAPI_PORT_1_1	ポート $P1_1$
RAPI_PORT_1_2	ポート $P1_2$	RAPI_PORT_1_3	ポート $P1_3$
RAPI_PORT_1_4	ポート $P1_4$	RAPI_PORT_1_5	ポート $P1_5$
RAPI_PORT_1_6	ポート $P1_6$。	RAPI_PORT_1_7	ポート $P1_7$
RAPI_PORT_2_0	ポート $P2_0$	RAPI_PORT_2_1	ポート $P2_1$
RAPI_PORT_2_2	ポート $P2_2$	RAPI_PORT_2_3	ポート $P2_3$
RAPI_PORT_2_4	ポート $P2_4$	RAPI_PORT_2_5	ポート $P2_5$
RAPI_PORT_2_6	ポート $P2_6$。	RAPI_PORT_2_7	ポート $P2_7$
RAPI_PORT_3_0	ポート $P3_0$	RAPI_PORT_3_1	ポート $P3_1$
RAPI_PORT_3_2	ポート $P3_2$	RAPI_PORT_3_3	ポート $P3_3$
RAPI_PORT_3_4	ポート $P3_4$	RAPI_PORT_3_5	ポート $P3_5$
RAPI_PORT_3_6	ポート $P3_6$	RAPI_PORT_3_7	ポート $P3_7$
RAPI_PORT_4_2	ポート $P4_2$	RAPI_PORT_4_3	ポート $P4_3$
RAPI_PORT_4_4	ポート $P4_4$	RAPI_PORT_4_5	ポート $P4_5$
RAPI_PORT_4_6	ポート $P4_4$	RAPI_PORT_4_7	ポート $P4_7$
RAPI_PORT_5_0	ポート $P5_0$	RAPI_PORT_5_1	ポート $P5_1$
RAPI_PORT_5_2	ポート $P5_2$	RAPI_PORT_5_3	ポート $P5_3$
RAPI_PORT_5_4	ポート $P5_4$	RAPI_PORT_5_5	ポート $P5_5$
RAPI_PORT_5_6	ポート $P5_6$	RAPI_PORT_5_7	ポート $P5_7$
RAPI_PORT_6_0	ポート $P6_0$	RAPI_PORT_6_1	ポート $P6_1$
RAPI_PORT_6_2	ポート $P6_2$	RAPI_PORT_6_3	ポート $P6_3$
RAPI_PORT_6_4	ポート $P6_4$	RAPI_PORT_6_5	ポート $P6_5$
RAPI_PORT_6_6	ポート $P6_6$	RAPI_PORT_6_7	ポート $P6_7$
RAPI_PORT_7_0	ポート $P7_0$	RAPI_PORT_7_1	ポート $P7_1$
RAPI_PORT_7_2	ポート $P7_2$	RAPI_PORT_7_3	ポート $P7_3$
RAPI_PORT_7_4	ポート $P7_4$	RAPI_PORT_7_5	ポート $P7_5$
RAPI_PORT_7_6	ポート $P7_6$	RAPI_PORT_7_7	ポート $P7_7$
RAPI_PORT_8_0	ポート $P8_0$	RAPI_PORT_8_1	ポート $P8_1$
RAPI_PORT_8_2	ポート $P8_2$	RAPI_PORT_8_3	ポート $P8_3$
RAPI_PORT_8_4	ポート $P8_4$	RAPI_PORT_8_5	ポート $P8_5$
RAPI_PORT_8_6	ポート $P8_6$	RAPI_PORT_8_7	ポート $P8_7$
RAPI_PORT_9_0	ポート $P9_0$	RAPI_PORT_9_1	ポート $P9_1$
RAPI_PORT_9_2	ポート $P9_2$	RAPI_PORT_9_3	ポート $P9_3$

(H8S)

RAPI_PORT_1_0	ポート P10	RAPI_PORT_1_1	ポート P11
RAPI_PORT_1_2	ポート P12	RAPI_PORT_1_3	ポート P13
RAPI_PORT_1_4	ポート P14	RAPI_PORT_1_5	ポート P15
RAPI_PORT_1_6	ポート P16	RAPI_PORT_1_7	ポート P17
RAPI_PORT_2_0	ポート P20	RAPI_PORT_2_1	ポート P21
RAPI_PORT_2_2	ポート P22	RAPI_PORT_2_3	ポート P23
RAPI_PORT_2_4	ポート P24	RAPI_PORT_2_5	ポート P25
RAPI_PORT_2_6	ポート P26	RAPI_PORT_2_7	ポート P27
RAPI_PORT_3_0	ポート P30	RAPI_PORT_3_1	ポート P31
RAPI_PORT_3_2	ポート P32	RAPI_PORT_3_3	ポート P33
RAPI_PORT_3_4	ポート P34	RAPI_PORT_3_5	ポート P35
RAPI_PORT_3_6	ポート P36	RAPI_PORT_3_7	ポート P37
RAPI_PORT_5_0	ポート P50	RAPI_PORT_5_1	ポート P51
RAPI_PORT_5_2	ポート P52	RAPI_PORT_5_3	ポート P53
RAPI_PORT_5_4	ポート P54	RAPI_PORT_5_5	ポート P55
RAPI_PORT_5_6	ポート P56	RAPI_PORT_5_7	ポート P57
RAPI_PORT_6_0	ポート P60	RAPI_PORT_6_1	ポート P61
RAPI_PORT_6_2	ポート P62	RAPI_PORT_6_3	ポート P63
RAPI_PORT_6_4	ポート P64	RAPI_PORT_6_5	ポート P65
RAPI_PORT_6_6	ポート P66	RAPI_PORT_6_7	ポート P67
RAPI_PORT_8_5	ポート P85	RAPI_PORT_8_6	ポート P86
RAPI_PORT_8_7	ポート P87	RAPI_PORT_9_0	ポート P90
RAPI_PORT_9_1	ポート P91	RAPI_PORT_9_2	ポート P92
RAPI_PORT_9_3	ポート P93	RAPI_PORT_9_4	ポート P94
RAPI_PORT_9_5	ポート P95	RAPI_PORT_9_6	ポート P96
RAPI_PORT_9_7	ポート P97	RAPI_PORT_A_0	ポート PA0
RAPI_PORT_A_1	ポート PA1	RAPI_PORT_A_2	ポート PA2
RAPI_PORT_A_3	ポート PA3	RAPI_PORT_A_4	ポート PA4
RAPI_PORT_A_5	ポート PA5	RAPI_PORT_A_6	ポート PA6
RAPI_PORT_A_7	ポート PA7	RAPI_PORT_B_0	ポート PB0
RAPI_PORT_B_1	ポート PB1	RAPI_PORT_B_2	ポート PB2
RAPI_PORT_B_3	ポート PB3	RAPI_PORT_B_4	ポート PB4
RAPI_PORT_B_5	ポート PB5	RAPI_PORT_B_6	ポート PB6
RAPI_PORT_B_7	ポート PB7	RAPI_PORT_J_0	ポート PJ0
RAPI_PORT_J_1	ポート PJ1		

(H8/300H)

RAPI_PORT_1_0	ポート P10	RAPI_PORT_1_1	ポート P11
RAPI_PORT_1_2	ポート P12	RAPI_PORT_1_4	ポート P14
RAPI_PORT_1_5	ポート P15	RAPI_PORT_1_6	ポート P16
RAPI_PORT_1_7	ポート P17	RAPI_PORT_2_0	ポート P20
RAPI_PORT_2_1	ポート P21	RAPI_PORT_2_2	ポート P22
RAPI_PORT_2_3	ポート P23	RAPI_PORT_2_4	ポート P24

RAPI_PORT_2_5	ポート P25	RAPI_PORT_2_6	ポート P26
RAPI_PORT_2_7	ポート P27	RAPI_PORT_3_0	ポート P30
RAPI_PORT_3_1	ポート P31	RAPI_PORT_3_2	ポート P32
RAPI_PORT_3_3	ポート P33	RAPI_PORT_3_4	ポート P34
RAPI_PORT_3_5	ポート P35	RAPI_PORT_3_6	ポート P36
RAPI_PORT_3_7	ポート P37	RAPI_PORT_5_0	ポート P50
RAPI_PORT_5_1	ポート P51	RAPI_PORT_5_2	ポート P52
RAPI_PORT_5_3	ポート P53	RAPI_PORT_5_4	ポート P54
RAPI_PORT_5_5	ポート P55	RAPI_PORT_5_6	ポート P56
RAPI_PORT_5_7	ポート P57	RAPI_PORT_6_0	ポート P60
RAPI_PORT_6_1	ポート P61	RAPI_PORT_6_2	ポート P62
RAPI_PORT_6_3	ポート P63	RAPI_PORT_6_4	ポート P64
RAPI_PORT_6_5	ポート P65	RAPI_PORT_6_6	ポート P66
RAPI_PORT_6_7	ポート P67	RAPI_PORT_7_0	ポート P70
RAPI_PORT_7_1	ポート P71	RAPI_PORT_7_2	ポート P72
RAPI_PORT_7_4	ポート P74	RAPI_PORT_7_5	ポート P75
RAPI_PORT_7_6	ポート P76	RAPI_PORT_7_7	ポート P77
RAPI_PORT_8_0	ポート P80	RAPI_PORT_8_1	ポート P81
RAPI_PORT_8_2	ポート P82	RAPI_PORT_8_3	ポート P83
RAPI_PORT_8_4	ポート P84	RAPI_PORT_8_5	ポート P85
RAPI_PORT_8_6	ポート P86	RAPI_PORT_8_7	ポート P87
RAPI_PORT_9_0	ポート P90	RAPI_PORT_9_1	ポート P91
RAPI_PORT_9_2	ポート P92	RAPI_PORT_9_3	ポート P93
RAPI_PORT_9_4	ポート P94	RAPI_PORT_9_5	ポート P95
RAPI_PORT_9_6	ポート P96	RAPI_PORT_9_7	ポート P97
RAPI_PORT_B_0	ポート PB0	RAPI_PORT_B_1	ポート PB1
RAPI_PORT_B_2	ポート PB2	RAPI_PORT_B_3	ポート PB3
RAPI_PORT_B_4	ポート PB4	RAPI_PORT_B_5	ポート PB5
RAPI_PORT_B_6	ポート PB6	RAPI_PORT_B_7	ポート PB7
RAPI_PORT_C_0	ポート PC0	RAPI_PORT_C_1	ポート PC1
RAPI_PORT_C_2	ポート PC2	RAPI_PORT_C_3	ポート PC3
RAPI_PORT_D_0	ポート PD0	RAPI_PORT_D_1	ポート PD1
RAPI_PORT_D_2	ポート PD2	RAPI_PORT_D_3	ポート PD3
RAPI_PORT_D_4	ポート PD4	RAPI_PORT_D_5	ポート PD5
RAPI_PORT_D_6	ポート PD6	RAPI_PORT_D_7	ポート PD7
RAPI_PORT_E_0	ポート PE0	RAPI_PORT_E_1	ポート PE1
RAPI_PORT_E_2	ポート PE2	RAPI_PORT_E_3	ポート PE3
RAPI_PORT_E_4	ポート PE4	RAPI_PORT_E_5	ポート PE5
RAPI_PORT_E_6	ポート PE6	RAPI_PORT_E_7	ポート PE7
RAPI_PORT_F_0	ポート PF0	RAPI_PORT_F_1	ポート PF1
RAPI_PORT_F_2	ポート PF2	RAPI_PORT_F_3	ポート PF3
RAPI_PORT_F_4	ポート PF4	RAPI_PORT_F_5	ポート PF5
RAPI_PORT_F_6	ポート PF6	RAPI_PORT_F_7	ポート PF7

RAPI_PORT_G_0	ポート PG0	RAPI_PORT_G_1	ポート PG1
RAPI_PORT_G_2	ポート PG2	RAPI_PORT_G_3	ポート PG3
RAPI_PORT_G_4	ポート PG4	RAPI_PORT_G_5	ポート PG5
RAPI_PORT_G_6	ポート PG6	RAPI_PORT_G_7	ポート PG7
RAPI_PORT_H_0	ポート PH0	RAPI_PORT_H_1	ポート PH1
RAPI_PORT_H_2	ポート PH2	RAPI_PORT_H_3	ポート PH3
RAPI_PORT_H_4	ポート PH4	RAPI_PORT_H_5	ポート PH5
RAPI_PORT_H_6	ポート PH6	RAPI_PORT_H_7	ポート PH7
RAPI_PORT_J_0	ポート PJ0	RAPI_PORT_J_1	ポート PJ1

戻り値

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

機能

I/O ポート

参照

__SetIOPort, __WriteIOPort, __SetIOPortRegister, __ReadIOPortRegister, __WriteIOPortRegister

備考

・第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・指定できる I/O ポートは CPU により異なります。
・H8/300H のポート B では、アナログ入力端子として使用されているポートは、
入力ポートとして使用することはできません。

プログラム例

```
#include " rapi_io_port_r8c_13.h"

void func( void )
{
    /* Get the value of port P12 */
    __ReadIOPort(RAPI_PORT_1_2, &data );
}
```

__WriteIOPort

要約

< I/O ポートへの書き込み >

Boolean __WriteIOPort(unsigned long data1, unsigned int data2)

data1	設定データ 1（機種により内容が異なる）
data2	I/O ポートに書き込むデータ

解説

指定した I/O ポートへ値の書き込みをおこないます。

[data1]

データを書き込む I/O ポートを指定します。各 I/O ポートに対応する定義値は次の通りです。

(M16C)

RAPI_PORT_0_0	ポート $P0_0$	RAPI_PORT_0_1	ポート $P0_1$
RAPI_PORT_0_2	ポート $P0_2$	RAPI_PORT_0_3	ポート $P0_3$
RAPI_PORT_0_4	ポート $P0_4$	RAPI_PORT_0_5	ポート $P0_5$
RAPI_PORT_0_6	ポート $P0_6$	RAPI_PORT_0_7	ポート $P0_7$
RAPI_PORT_1_0	ポート $P1_0$	RAPI_PORT_1_1	ポート $P1_1$
RAPI_PORT_1_2	ポート $P1_2$	RAPI_PORT_1_3	ポート $P1_3$
RAPI_PORT_1_4	ポート $P1_4$	RAPI_PORT_1_5	ポート $P1_5$
RAPI_PORT_1_6	ポート $P1_6$	RAPI_PORT_1_7	ポート $P1_7$
RAPI_PORT_2_0	ポート $P2_0$	RAPI_PORT_2_1	ポート $P2_1$
RAPI_PORT_2_2	ポート $P2_2$	RAPI_PORT_2_3	ポート $P2_3$
RAPI_PORT_2_4	ポート $P2_4$	RAPI_PORT_2_5	ポート $P2_5$
RAPI_PORT_2_6	ポート $P2_6$	RAPI_PORT_2_7	ポート $P2_7$
RAPI_PORT_3_0	ポート $P3_0$	RAPI_PORT_3_1	ポート $P3_1$
RAPI_PORT_3_2	ポート $P3_2$	RAPI_PORT_3_3	ポート $P3_3$
RAPI_PORT_3_4	ポート $P3_4$	RAPI_PORT_3_5	ポート $P3_5$
RAPI_PORT_3_6	ポート $P3_6$	RAPI_PORT_3_7	ポート $P3_7$
RAPI_PORT_6_0	ポート $P6_0$	RAPI_PORT_6_1	ポート $P6_1$
RAPI_PORT_6_2	ポート $P6_2$	RAPI_PORT_6_3	ポート $P6_3$
RAPI_PORT_6_4	ポート $P6_4$	RAPI_PORT_6_5	ポート $P6_5$
RAPI_PORT_6_6	ポート $P6_6$	RAPI_PORT_6_7	ポート $P6_7$
RAPI_PORT_7_0	ポート $P7_0$	RAPI_PORT_7_1	ポート $P7_1$
RAPI_PORT_7_2	ポート $P7_2$	RAPI_PORT_7_3	ポート $P7_3$
RAPI_PORT_7_4	ポート $P7_4$	RAPI_PORT_7_5	ポート $P7_5$
RAPI_PORT_7_6	ポート $P7_6$	RAPI_PORT_7_7	ポート $P7_7$
RAPI_PORT_8_0	ポート $P8_0$	RAPI_PORT_8_1	ポート $P8_1$
RAPI_PORT_8_2	ポート $P8_2$	RAPI_PORT_8_3	ポート $P8_3$
RAPI_PORT_8_4	ポート $P8_4$	RAPI_PORT_8_5	ポート $P8_5$
RAPI_PORT_8_6	ポート $P8_6$	RAPI_PORT_8_7	ポート $P8_7$
RAPI_PORT_9_0	ポート $P9_0$	RAPI_PORT_9_1	ポート $P9_1$
RAPI_PORT_9_2	ポート $P9_2$	RAPI_PORT_9_3	ポート $P9_3$
RAPI_PORT_9_5	ポート $P9_5$	RAPI_PORT_9_6	ポート $P9_6$

RAPI_PORT_9_7	ポート $P9_7$	RAPI_PORT_10_0	ポート $P10_0$
RAPI_PORT_10_1	ポート $P10_1$	RAPI_PORT_10_2	ポート $P10_2$
RAPI_PORT_10_3	ポート $P10_3$	RAPI_PORT_10_4	ポート $P10_4$
RAPI_PORT_10_5	ポート $P10_5$	RAPI_PORT_10_6	ポート $P10_6$
RAPI_PORT_10_7	ポート $P10_7$		

(R8C)

RAPI_PORT_0_0	ポート $P0_0$	RAPI_PORT_0_1	ポート $P0_1$
RAPI_PORT_0_2	ポート $P0_2$	RAPI_PORT_0_3	ポート $P0_3$
RAPI_PORT_0_4	ポート $P0_4$	RAPI_PORT_0_5	ポート $P0_5$
RAPI_PORT_0_6	ポート $P0_6$	RAPI_PORT_0_7	ポート $P0_7$
RAPI_PORT_1_0	ポート $P1_0$	RAPI_PORT_1_1	ポート $P1_1$
RAPI_PORT_1_2	ポート $P1_2$	RAPI_PORT_1_3	ポート $P1_3$
RAPI_PORT_1_4	ポート $P1_4$	RAPI_PORT_1_5	ポート $P1_5$
RAPI_PORT_1_6	ポート $P1_{6}$。	RAPI_PORT_1_7	ポート $P1_7$
RAPI_PORT_2_0	ポート $P2_0$	RAPI_PORT_2_1	ポート $P2_1$
RAPI_PORT_2_2	ポート $P2_2$	RAPI_PORT_2_3	ポート $P2_3$
RAPI_PORT_2_4	ポート $P2_4$	RAPI_PORT_2_5	ポート $P2_5$
RAPI_PORT_2_6	ポート $P2_{6}$。	RAPI_PORT_2_7	ポート $P2_7$
RAPI_PORT_3_0	ポート $P3_0$	RAPI_PORT_3_1	ポート $P3_1$
RAPI_PORT_3_2	ポート $P3_2$	RAPI_PORT_3_3	ポート $P3_3$
RAPI_PORT_3_4	ポート $P3_4$	RAPI_PORT_3_5	ポート $P3_5$
RAPI_PORT_3_6	ポート $P3_6$	RAPI_PORT_3_7	ポート $P3_7$
RAPI_PORT_4_3	ポート $P4_3$	RAPI_PORT_4_4	ポート $P4_4$
RAPI_PORT_4_5	ポート $P4_5$	RAPI_PORT_5_0	ポート $P5_0$
RAPI_PORT_5_1	ポート $P5_1$	RAPI_PORT_5_2	ポート $P5_2$
RAPI_PORT_5_3	ポート $P5_3$	RAPI_PORT_5_4	ポート $P5_4$
RAPI_PORT_5_5	ポート $P5_5$	RAPI_PORT_5_6	ポート $P5_6$
RAPI_PORT_5_7	ポート $P5_7$	RAPI_PORT_6_0	ポート $P6_0$
RAPI_PORT_6_1	ポート $P6_1$	RAPI_PORT_6_2	ポート $P6_2$
RAPI_PORT_6_3	ポート $P6_3$	RAPI_PORT_6_4	ポート $P6_4$
RAPI_PORT_6_5	ポート $P6_5$	RAPI_PORT_6_6	ポート $P6_6$
RAPI_PORT_6_7	ポート $P6_7$	RAPI_PORT_7_0	ポート $P7_0$
RAPI_PORT_7_1	ポート $P7_1$	RAPI_PORT_7_2	ポート $P7_2$
RAPI_PORT_7_3	ポート $P7_3$	RAPI_PORT_7_4	ポート $P7_4$
RAPI_PORT_7_5	ポート $P7_5$	RAPI_PORT_7_6	ポート $P7_6$
RAPI_PORT_7_7	ポート $P7_7$	RAPI_PORT_8_0	ポート $P8_0$
RAPI_PORT_8_1	ポート $P8_1$	RAPI_PORT_8_2	ポート $P8_2$
RAPI_PORT_8_3	ポート $P8_3$	RAPI_PORT_8_4	ポート $P8_4$
RAPI_PORT_8_5	ポート $P8_5$	RAPI_PORT_8_6	ポート $P8_6$
RAPI_PORT_8_7	ポート $P8_7$	RAPI_PORT_9_0	ポート $P9_0$
RAPI_PORT_9_1	ポート $P9_1$	RAPI_PORT_9_2	ポート $P9_2$
RAPI_PORT_9_3	ポート $P9_3$		

(H8S)

RAPI_PORT_1_0	ポート P10	RAPI_PORT_1_1	ポート P11
RAPI_PORT_1_2	ポート P12	RAPI_PORT_1_3	ポート P13
RAPI_PORT_1_4	ポート P14	RAPI_PORT_1_5	ポート P15
RAPI_PORT_1_6	ポート P16	RAPI_PORT_1_7	ポート P17
RAPI_PORT_2_0	ポート P20	RAPI_PORT_2_1	ポート P21
RAPI_PORT_2_2	ポート P22	RAPI_PORT_2_3	ポート P23
RAPI_PORT_2_4	ポート P24	RAPI_PORT_2_5	ポート P25
RAPI_PORT_2_6	ポート P26	RAPI_PORT_2_7	ポート P27
RAPI_PORT_3_0	ポート P30	RAPI_PORT_3_1	ポート P31
RAPI_PORT_3_2	ポート P32	RAPI_PORT_3_3	ポート P33
RAPI_PORT_3_4	ポート P34	RAPI_PORT_3_5	ポート P35
RAPI_PORT_3_6	ポート P36	RAPI_PORT_3_7	ポート P37
RAPI_PORT_5_0	ポート P50	RAPI_PORT_5_1	ポート P51
RAPI_PORT_5_2	ポート P52	RAPI_PORT_5_3	ポート P53
RAPI_PORT_5_4	ポート P54	RAPI_PORT_5_5	ポート P55
RAPI_PORT_5_6	ポート P56	RAPI_PORT_5_7	ポート P57
RAPI_PORT_6_0	ポート P60	RAPI_PORT_6_1	ポート P61
RAPI_PORT_6_2	ポート P62	RAPI_PORT_6_3	ポート P63
RAPI_PORT_6_4	ポート P64	RAPI_PORT_6_5	ポート P65
RAPI_PORT_6_6	ポート P66	RAPI_PORT_6_7	ポート P67
RAPI_PORT_8_5	ポート P85	RAPI_PORT_8_6	ポート P86
RAPI_PORT_8_7	ポート P87	RAPI_PORT_9_0	ポート P90
RAPI_PORT_9_1	ポート P91	RAPI_PORT_9_2	ポート P92
RAPI_PORT_9_3	ポート P93	RAPI_PORT_9_4	ポート P94
RAPI_PORT_9_5	ポート P95	RAPI_PORT_9_6	ポート P96
RAPI_PORT_9_7	ポート P97	RAPI_PORT_A_0	ポート PA0
RAPI_PORT_A_1	ポート PA1	RAPI_PORT_A_2	ポート PA2
RAPI_PORT_A_3	ポート PA3	RAPI_PORT_A_4	ポート PA4
RAPI_PORT_A_5	ポート PA5	RAPI_PORT_A_6	ポート PA6
RAPI_PORT_A_7	ポート PA7	RAPI_PORT_B_0	ポート PB0
RAPI_PORT_B_1	ポート PB1	RAPI_PORT_B_2	ポート PB2
RAPI_PORT_B_3	ポート PB3	RAPI_PORT_B_4	ポート PB4
RAPI_PORT_B_5	ポート PB5	RAPI_PORT_B_6	ポート PB6
RAPI_PORT_B_7	ポート PB7	RAPI_PORT_J_0	ポート PJ0
RAPI_PORT_J_1	ポート PJ1		

(H8/300H)

RAPI_PORT_1_0	ポート P10	RAPI_PORT_1_1	ポート P11
RAPI_PORT_1_2	ポート P12	RAPI_PORT_1_4	ポート P14
RAPI_PORT_1_5	ポート P15	RAPI_PORT_1_6	ポート P16
RAPI_PORT_1_7	ポート P17	RAPI_PORT_2_0	ポート P20
RAPI_PORT_2_1	ポート P21	RAPI_PORT_2_2	ポート P22
RAPI_PORT_2_3	ポート P23	RAPI_PORT_2_4	ポート P24
RAPI_PORT_2_5	ポート P25	RAPI_PORT_2_6	ポート P26

RAPI_PORT_2_7	ポート P27	RAPI_PORT_3_0	ポート P30
RAPI_PORT_3_1	ポート P31	RAPI_PORT_3_2	ポート P32
RAPI_PORT_3_3	ポート P33	RAPI_PORT_3_4	ポート P34
RAPI_PORT_3_5	ポート P35	RAPI_PORT_3_6	ポート P36
RAPI_PORT_3_7	ポート P37	RAPI_PORT_5_0	ポート P50
RAPI_PORT_5_1	ポート P51	RAPI_PORT_5_2	ポート P52
RAPI_PORT_5_3	ポート P53	RAPI_PORT_5_4	ポート P54
RAPI_PORT_5_5	ポート P55	RAPI_PORT_5_6	ポート P56
RAPI_PORT_5_7	ポート P57	RAPI_PORT_6_0	ポート P60
RAPI_PORT_6_1	ポート P61	RAPI_PORT_6_2	ポート P62
RAPI_PORT_6_3	ポート P63	RAPI_PORT_6_4	ポート P64
RAPI_PORT_6_5	ポート P65	RAPI_PORT_6_6	ポート P66
RAPI_PORT_6_7	ポート P67	RAPI_PORT_7_0	ポート P70
RAPI_PORT_7_1	ポート P71	RAPI_PORT_7_2	ポート P72
RAPI_PORT_7_4	ポート P74	RAPI_PORT_7_5	ポート P75
RAPI_PORT_7_6	ポート P76	RAPI_PORT_7_7	ポート P77
RAPI_PORT_8_0	ポート P80	RAPI_PORT_8_1	ポート P81
RAPI_PORT_8_2	ポート P82	RAPI_PORT_8_3	ポート P83
RAPI_PORT_8_4	ポート P84	RAPI_PORT_8_5	ポート P85
RAPI_PORT_8_6	ポート P86	RAPI_PORT_8_7	ポート P87
RAPI_PORT_9_0	ポート P90	RAPI_PORT_9_1	ポート P91
RAPI_PORT_9_2	ポート P92	RAPI_PORT_9_3	ポート P93
RAPI_PORT_9_4	ポート P94	RAPI_PORT_9_5	ポート P95
RAPI_PORT_9_6	ポート P96	RAPI_PORT_9_7	ポート P97
RAPI_PORT_C_0	ポート PC0	RAPI_PORT_C_1	ポート PC1
RAPI_PORT_C_2	ポート PC2	RAPI_PORT_C_3	ポート PC3
RAPI_PORT_D_0	ポート PD0	RAPI_PORT_D_1	ポート PD1
RAPI_PORT_D_2	ポート PD2	RAPI_PORT_D_3	ポート PD3
RAPI_PORT_D_4	ポート PD4	RAPI_PORT_D_5	ポート PD5
RAPI_PORT_D_6	ポート PD6	RAPI_PORT_D_7	ポート PD7
RAPI_PORT_E_0	ポート PE0	RAPI_PORT_E_1	ポート PE1
RAPI_PORT_E_2	ポート PE2	RAPI_PORT_E_3	ポート PE3
RAPI_PORT_E_4	ポート PE4	RAPI_PORT_E_5	ポート PE5
RAPI_PORT_E_6	ポート PE6	RAPI_PORT_E_7	ポート PE7
RAPI_PORT_G_0	ポート PG0	RAPI_PORT_G_1	ポート PG1
RAPI_PORT_G_2	ポート PG2	RAPI_PORT_G_3	ポート PG3
RAPI_PORT_G_4	ポート PG4	RAPI_PORT_G_5	ポート PG5
RAPI_PORT_G_6	ポート PG6	RAPI_PORT_G_7	ポート PG7
RAPI_PORT_H_0	ポート PH0	RAPI_PORT_H_1	ポート PH1
RAPI_PORT_H_2	ポート PH2	RAPI_PORT_H_3	ポート PH3
RAPI_PORT_H_4	ポート PH4	RAPI_PORT_H_5	ポート PH5
RAPI_PORT_H_6	ポート PH6	RAPI_PORT_H_7	ポート PH7
RAPI_PORT_J_0	ポート PJ0	RAPI_PORT_J_1	ポート PJ1

戻り値　I/O ポート指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能　I/O ポート

参照　__SetIOPort, __ReadIOPort, __SetIOPortRegister, __ReadIOPortRegister, __WriteIOPortRegister

備考
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できる I/O ポートは CPU により異なります。

プログラム例

```
#include " rapi_io_port_r8c_13.h"

void func( void )
{
    unsigned int data;

    /* Set the data to port P05 */
    __WriteIOPort( RAPI_PORT_0_5, 0 );
}
```

__SetIOPortRegister

要約

< I/O ポートレジスタの設定 >

Boolean __SetIOPortRegister(unsigned long data1, unsigned int data2, unsigned int data3, unsigned int data4)

data1	設定データ 1（機種により内容が異なる）
data2	設定データ 2（機種により内容が異なる）
data3	設定データ 3（機種により内容が異なる）
data4	設定データ 4（機種により内容が異なる）

解説

[data1]

指定した I/O ポートの動作条件設定をレジスタ単位でおこないます。

(M16C)

各 I/O ポートレジスタに対応する定義値は次の通りです。

RAPI_PORT_0	ポート P0 レジスタ	RAPI_PORT_1	ポート P1 レジスタ
RAPI_PORT_2	ポート P2 レジスタ	RAPI_PORT_3	ポート P3 レジスタ
RAPI_PORT_6	ポート P6 レジスタ	RAPI_PORT_7	ポート P7 レジスタ
RAPI_PORT_8	ポート P8 レジスタ	RAPI_PORT_9	ポート P9 レジスタ
RAPI_PORT_10	ポート P10 レジスタ		

設定に関する定義値は次の通りです。

RAPI_LATCH	入力ポート設定、出力ポート設定にかかわらず、ポートラッチを読むように設定する。ポート P1 使用時のみ指定可能。

(R8C)

各 I/O ポートレジスタに対応する定義値は次の通りです。

RAPI_PORT_0	ポート P0 レジスタ	RAPI_PORT_1	ポート P1 レジスタ
RAPI_PORT_2	ポート P2 レジスタ	RAPI_PORT_3	ポート P3 レジスタ
RAPI_PORT_4	ポート P4 レジスタ	RAPI_PORT_5	ポート P5 レジスタ
RAPI_PORT_6	ポート P6 レジスタ	RAPI_PORT_7	ポート P7 レジスタ
RAPI_PORT_8	ポート P8 レジスタ	RAPI_PORT_9	ポート P9 レジスタ

(H8S)

各 I/O ポートレジスタに対応する定義値は次の通りです。

RAPI_PORT_1	ポート P1 レジスタ	RAPI_PORT_2	ポート P2 レジスタ
RAPI_PORT_3	ポート P3 レジスタ	RAPI_PORT_5	ポート P5 レジスタ
RAPI_PORT_6	ポート P6 レジスタ	RAPI_PORT_8	ポート P8 レジスタ
RAPI_PORT_9	ポート P9 レジスタ	RAPI_PORT_A	ポート PA レジスタ
RAPI_PORT_B	ポート PB レジスタ	RAPI_PORT_J	ポート PJ レジスタ

(H8/300H)

各 I/O ポートレジスタに対応する定義値は次の通りです。

RAPI_PORT_1	ポート P1 レジスタ	RAPI_PORT_2	ポート P2 レジスタ
RAPI_PORT_3	ポート P3 レジスタ	RAPI_PORT_5	ポート P5 レジスタ
RAPI_PORT_6	ポート P6 レジスタ	RAPI_PORT_7	ポート P7 レジスタ
RAPI_PORT_8	ポート P8 レジスタ	RAPI_PORT_9	ポート P9 レジスタ

RAPI_PORT_C	ポート PC レジスタ	RAPI_PORT_D	ポート PD レジスタ
RAPI_PORT_E	ポート PE レジスタ	RAPI_PORT_F	ポート PF レジスタ
RAPI_PORT_G	ポート PG レジスタ	RAPI_PORT_H	ポート PH レジスタ
RAPI_PORT_J	ポート PJ レジスタ		

[data2]

(M16C)(R8C)

選択したポートに対応するポート方向レジスタへの設定値を指定します。

(H8S)(H8/300H)

選択したポートに対応するポートコントロールレジスタへの設定値を指定します。

[data3]

(M16C)(R8C)

選択したポートに対応するプルアップ制御レジスタへの設定値を指定します。

(H8S)

選択したポートに対応するポートプルアップコントロールレジスタへの設定値を
指定します。

(H8/300H)

選択したポートに対応するプルアップコントロールレジスタへの設定値を指定します。

[data4]

(M16C)

_NMI/_SD に割り当てられたデジタルデバウンス機能のデジタルフィルタ幅を
指定します。ポート P8 使用時のみ指定可能。
INPC17/_INT5 に割り当てられたデジタルデバウンス機能のデジタルフィルタ幅を
指定します。ポート P1 使用時のみ指定可能。
上記以外のポート使用時は 0 を設定してください。

(R8C)

ポート P1 駆動能力レジスタへの設定値を指定します。ポート P1 選択時のみ指定可能。
ポート P1 使用時以外は 0 を設定してください。

(H8S)

選択したポートに対応するポートドライブコントロールレジスタへの設定値を
指定します。

(H8/300H)

0 を指定します。

戻り値

I/O ポートレジスタ指定が正しくない場合は RAPI_FALSE を返し、それ以外は
RAPI_TRUE を返します。

機能

I/O ポート

参照

__SetIOPort, __ReadIOPort, __WriteIOPort, __ReadIOPortRegister,
__WriteIOPortRegister

備考

・第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・指定できる I/O ポートレジスタは CPU により異なります。
・指定した I/O ポートが有しない機能を設定することはできません。

プログラム例

```
#include " rapi_io_port_r8c_13.h"

void func( void )
{
    /* Set inputs/outputs of port P1 register */
    __SetIOPortRegister(RAPI_PORT_1, 0xAA, 0, 0 );
}
```

__ReadIOPortRegister

要約

< I/O ポートレジスタからの読み出し >

Boolean __ReadIOPortRegister(unsigned long data1, unsigned int *data2)

data1	設定データ 1（機種により内容が異なる）
data2	I/O ポートレジスタから読み出した値を格納する変数のポインタ

解説

指定した I/O ポートの値をレジスタ単位で取得します。

[data1]

データを読み出す I/O ポートレジスタを指定します。各 I/O ポートレジスタに対応する定義値は次の通りです。

(M16C)

RAPI_PORT_0	ポート P0 レジスタ	RAPI_PORT_1	ポート P1 レジスタ
RAPI_PORT_2	ポート P2 レジスタ	RAPI_PORT_3	ポート P3 レジスタ
RAPI_PORT_6	ポート P6 レジスタ	RAPI_PORT_7	ポート P7 レジスタ
RAPI_PORT_8	ポート P8 レジスタ	RAPI_PORT_9	ポート P9 レジスタ
RAPI_PORT_10	ポート P10 レジスタ		

(R8C)

RAPI_PORT_0	ポート P0 レジスタ	RAPI_PORT_1	ポート P1 レジスタ
RAPI_PORT_2	ポート P2 レジスタ	RAPI_PORT_3	ポート P3 レジスタ
RAPI_PORT_4	ポート P4 レジスタ	RAPI_PORT_5	ポート P5 レジスタ
RAPI_PORT_6	ポート P6 レジスタ	RAPI_PORT_7	ポート P7 レジスタ
RAPI_PORT_8	ポート P8 レジスタ	RAPI_PORT_9	ポート P9 レジスタ

(H8S)

RAPI_PORT_1	ポート P1 レジスタ	RAPI_PORT_2	ポート P2 レジスタ
RAPI_PORT_3	ポート P3 レジスタ	RAPI_PORT_5	ポート P5 レジスタ
RAPI_PORT_6	ポート P6 レジスタ	RAPI_PORT_8	ポート P8 レジスタ
RAPI_PORT_9	ポート P9 レジスタ	RAPI_PORT_A	ポート PA レジスタ
RAPI_PORT_B	ポート PB レジスタ	RAPI_PORT_J	ポート PJ レジスタ

(H8/300H)

RAPI_PORT_1	ポート P1 レジスタ	RAPI_PORT_2	ポート P2 レジスタ
RAPI_PORT_3	ポート P3 レジスタ	RAPI_PORT_5	ポート P5 レジスタ
RAPI_PORT_6	ポート P6 レジスタ	RAPI_PORT_7	ポート P7 レジスタ
RAPI_PORT_8	ポート P8 レジスタ	RAPI_PORT_9	ポート P9 レジスタ
RAPI_PORT_B	ポート PB レジスタ	RAPI_PORT_C	ポート PC レジスタ
RAPI_PORT_D	ポート PD レジスタ	RAPI_PORT_E	ポート PE レジスタ
RAPI_PORT_F	ポート PF レジスタ	RAPI_PORT_G	ポート PG レジスタ
RAPI_PORT_H	ポート PH レジスタ	RAPI_PORT_J	ポート PJ レジスタ

戻り値

I/O ポートレジスタ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能　I/O ポート

参照　__SetIOPort, __ReadIOPort, __WriteIOPort, __SetIOPortRegister, __WriteIOPortRegister

備考
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できる I/O ポートレジスタは CPU により異なります。
- H8/300H のポート B では、アナログ入力端子として使用されているポートは、入力ポートとして使用することはできません。

プログラム例

```
#include " rapi_io_port_r8c_13.h"

void func( void )
{
    unsigned int data;

    /* Get the value of port P1 register */
    __ReadIOPortRegister( RAPI_PORT_1, &data );
}
```

__WriteIOPortRegister

要約

< I/O ポートレジスタへの書き込み >

Boolean __WriteIOPortRegister(unsigned long data1, unsigned int data2)

data1	設定データ 1（機種により内容が異なる）
data2	I/O ポートレジスタに書き込むデータ

解説

指定した I/O ポートへ値の書き込みをレジスタ単位でおこないます。

[data1]

データを書き込む I/O ポートレジスタを指定します。各 I/O ポートレジスタに対応する定義値は次の通りです。

(M16C)

RAPI_PORT_0	ポート P0 レジスタ	RAPI_PORT_1	ポート P1 レジスタ
RAPI_PORT_2	ポート P2 レジスタ	RAPI_PORT_3	ポート P3 レジスタ
RAPI_PORT_6	ポート P6 レジスタ	RAPI_PORT_7	ポート P7 レジスタ
RAPI_PORT_8	ポート P8 レジスタ	RAPI_PORT_9	ポート P9 レジスタ
RAPI_PORT_10	ポート P10 レジスタ		

(R8C)

RAPI_PORT_0	ポート P0 レジスタ	RAPI_PORT_1	ポート P1 レジスタ
RAPI_PORT_2	ポート P2 レジスタ	RAPI_PORT_3	ポート P3 レジスタ
RAPI_PORT_4	ポート P4 レジスタ	RAPI_PORT_5	ポート P5 レジスタ
RAPI_PORT_6	ポート P6 レジスタ	RAPI_PORT_7	ポート P7 レジスタ
RAPI_PORT_8	ポート P8 レジスタ	RAPI_PORT_9	ポート P9 レジスタ

(H8S)

RAPI_PORT_1	ポート P1 レジスタ	RAPI_PORT_2	ポート P2 レジスタ
RAPI_PORT_3	ポート P3 レジスタ	RAPI_PORT_5	ポート P5 レジスタ
RAPI_PORT_6	ポート P6 レジスタ	RAPI_PORT_8	ポート P8 レジスタ
RAPI_PORT_9	ポート P9 レジスタ	RAPI_PORT_A	ポート PA レジスタ
RAPI_PORT_B	ポート PB レジスタ	RAPI_PORT_J	ポート PJ レジスタ

(H8/300H)

RAPI_PORT_1	ポート P1 レジスタ	RAPI_PORT_2	ポート P2 レジスタ
RAPI_PORT_3	ポート P3 レジスタ	RAPI_PORT_5	ポート P5 レジスタ
RAPI_PORT_6	ポート P6 レジスタ	RAPI_PORT_7	ポート P7 レジスタ
RAPI_PORT_8	ポート P8 レジスタ	RAPI_PORT_9	ポート P9 レジスタ
RAPI_PORT_C	ポート PC レジスタ	RAPI_PORT_D	ポート PD レジスタ
RAPI_PORT_E	ポート PE レジスタ	RAPI_PORT_G	ポート PG レジスタ
RAPI_PORT_H	ポート PH レジスタ	RAPI_PORT_J	ポート PJ レジスタ

戻り値

I/O ポートレジスタ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

I/O ポート

参照

__SetIOPort, __ReadIOPort, __WriteIOPort, __SetIOPortRegister, __ReadIOPortRegister

備考

・ 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できる I/O ポートレジスタは CPU により異なります。

プログラム例

```
#include " rapi_io_port_r8c_13.h"

void func( void )
{
    /* Set the data to port P1 register */
    __WriteIOPortRegister( RAPI_PORT_1, 0xFF );
}
```

4.2.4 外部割り込み
__SetInterrupt

要約

< 外部割り込みの設定 >

Boolean __SetInterrupt(unsigned long data1, unsigned int data2, void* func)

data1	設定データ 1（機種により内容が異なる）
data2	設定データ 2（機種により内容が異なる）
func	コールバック関数ポインタ （コールバック関数を設定しない場合は 0 を指定する）

解説

指定した外部割り込みの設定をおこないます。

[data1]

data1 には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。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。

(M16C)

RAPI_INT0	_INT0 割り込みを使用する。
RAPI_INT1	_INT1 割り込みを使用する。
RAPI_INT2	_INT2 割り込みを使用する。
RAPI_INT3	_INT3 割り込みを使用する。
RAPI_INT4	_INT4 割り込みを使用する。
RAPI_INT5	_INT5 割り込みを使用する。
RAPI_KEY	キー入力割り込みを使用する。
RAPI_INT_RISING	選択した外部割り込みの有効エッジに立ち上がりエッジを指定する。
RAPI_INT_FALLING	選択した外部割り込みの有効エッジに立ち下がりエッジを指定する。
RAPI_INT_BOTH	選択した外部割り込みの有効エッジに両エッジを指定する。
RAPI_KI0_ENABLE	_KI0 端子入力を使用する。
RAPI_KI1_ENABLE	_KI1 端子入力を使用する。
RAPI_KI2_ENABLE	_KI2 端子入力を使用する。
RAPI_KI3_ENABLE	_KI3 端子入力を使用する。

・_INT0～5 割り込み使用時 (RAPI_INT0～RAPI_INT5 を指定時) に指定可能な定義値

(極性)　{ RAPI_INT_RISING, RAPI_INT_FALLING, RAPI_INT_BOTH }から 1 つを指定します。省略値は RAPI_INT_FALLING です。

・キー入力割り込み使用時 (RAPI_KEY を指定時) に指定可能な定義値

(入力端子)　_KI0 端子入力を使用する場合は RAPI_KI0_ENABLE を、_KI1 端子入力を使用する場合は RAPI_KI1_ENABLE を、_KI2 端子入力を使用する場合は RAPI_KI2_ENABLE を、_KI3 端子入力を使用する場合は RAPI_KI3_ENABLE を指定します。

(R8C)

RAPI_INT0	_INT0 割り込みを使用する。
RAPI_INT1	_INT1 割り込みを使用する。
RAPI_INT2	_INT2 割り込みを使用する。
RAPI_INT3	_INT3 割り込みを使用する。
RAPI_KEY	キー入力割り込みを使用する。

RAPI_INT_RISING	選択した外部割り込みの有効エッジに立ち上がりエッジを指定する。
RAPI_INT_FALLING	選択した外部割り込みの有効エッジに立ち下がりエッジを指定する。
RAPI_INT_BOTH	選択した外部割り込みの有効エッジに両エッジを指定する。
RAPI_FILTER_F1	サンプリング周波数 f_1 のデジタルフィルタ機能を使用する。
RAPI_FILTER_F8	サンプリング周波数 f_8 のデジタルフィルタ機能を使用する。
RAPI_FILTER_F32	サンプリング周波数 f_{32} のデジタルフィルタ機能を使用する。
RAPI_KI0_FALLING	立ち下がりエッジを有効エッジとする_KI0 端子入力を許可する。
RAPI_KI0_RISING	立ち上がりエッジを有効エッジとする_KI0 端子入力を許可する。
RAPI_KI1_FALLING	立ち下がりエッジを有効エッジとする_KI1 端子入力を許可する。
RAPI_KI1_RISING	立ち上がりエッジを有効エッジとする_KI1 端子入力を許可する。
RAPI_KI2_FALLING	立ち下がりエッジを有効エッジとする_KI2 端子入力を許可する。
RAPI_KI2_RISING	立ち上がりエッジを有効エッジとする_KI2 端子入力を許可する。
RAPI_KI3_FALLING	立ち下がりエッジを有効エッジとする_KI3 端子入力を許可する。
RAPI_KI3_RISING	立ち上がりエッジを有効エッジとする_KI3 端子入力を許可する。
RAPI_INT1_P1_5	INT1 端子に P1_5 を指定する。
RAPI_INT1_P1_7	INT1 端子に P1_7 を指定する。
RAPI_INT1_P3_6	INT1 端子に P3_6 を指定する。
RAPI_INT2_P6_6	INT2 端子に P6_6 を指定する。
RAPI_INT2_P3_2	INT2 端子に P3_2 を指定する。

・_INT0～_INT3 割り込み使用時（RAPI_INT0～RAPI_INT3 を指定時）に指定可能な定義値

（極性）　{ RAPI_INT_RISING, RAPI_INT_FALLING, RAPI_INT_BOTH }から 1 つを指定します。省略値は RAPI_INT_FALLING です。

（フィルタ）　{ RAPI_FILTER_F1, RAPI_FILTER_F8, RAPI_FILTER_F32 }から 1 つを指定します。フィルタを指定しない場合は、フィルタなしに設定します。

・キー入力割り込み使用時（RAPI_KEY を指定時）に指定可能な定義値

（_KI0 端子入力）　{ RAPI_KI0_FALLING, RAPI_KI0_RISING }から 1 つを指定します。
_KI0 端子入力を指定しない場合は、「_KI0 端子入力禁止」を設定します。

（_KI1 端子入力）　{ RAPI_KI1_FALLING, RAPI_KI1_RISING }から 1 つを指定します。
_KI1 端子入力を指定しない場合は、「_KI1 端子入力禁止」を設定します。

（_KI2 端子入力）　{ RAPI_KI2_FALLING, RAPI_KI2_RISING }から 1 つを指定します。
_KI2 端子入力を指定しない場合は、「_KI2 端子入力禁止」を設定します。

（_KI3 端子入力）　{ RAPI_KI3_FALLING, RAPI_KI3_RISING }から 1 つを指定します。
_KI3 端子入力を指定しない場合は、「_KI3 端子入力禁止」を設定します。

(H8S)

RAPI_IRQ0	IRQ0 割り込みを使用する。
RAPI_IRQ1	IRQ1 割り込みを使用する。
RAPI_IRQ2	IRQ2 割り込みを使用する。
RAPI_IRQ3	IRQ3 割り込みを使用する。
RAPI_IRQ4	IRQ4 割り込みを使用する。
RAPI_IRQ5	IRQ5 割り込みを使用する。
RAPI_IRQ6	IRQ6 割り込みを使用する。
RAPI_IRQ7	IRQ7 割り込みを使用する。
RAPI_NOT_INT_REQUEST	選択した外部割り込みの割り込み要求をしない。

RAPI_INT_REQUEST	選択した外部割り込みの割り込み要求をする。
RAPI_INT_RISING	選択した外部割り込みの有効エッジに立ち上がりエッジを指定する。
RAPI_INT_FALLING	選択した外部割り込みの有効エッジに立ち下がりエッジを指定する。
RAPI_INT_BOTH	選択した外部割り込みの有効エッジに両エッジを指定する。
RAPI_FILTER_F1	サンプリング周波数 f_1 のノイズキャンセル機能を使用する。
RAPI_FILTER_F2	サンプリング周波数 f_2 のノイズキャンセル機能を使用する。
RAPI_FILTER_F4	サンプリング周波数 f_4 のノイズキャンセル機能を使用する。
RAPI_FILTER_F8	サンプリング周波数 f_8 のノイズキャンセル機能を使用する。
RAPI_INT_MODE_0	割り込み制御モード 0 を指定する。
RAPI_INT_MODE_2	割り込み制御モード 2 を指定する。

・IRQ0～7 割り込み使用時（RAPI_IRQ0～RAPI_IRQ7 を指定時）に指定可能な定義値

（極性）　{ RAPI_INT_RISING, RAPI_INT_FALLING, RAPI_INT_BOTH }から 1 つを指定します。

（割り込み要求）　{ RAPI_NOT_INT_REQUEST, RAPI_INT_REQUEST }から 1 つを指定します。省略値は RAPI_NOT_INT_REQUEST です。

（割り込み制御モード）　{ RAPI_INT_MODE_0, RAPI_INT_MODE_2 }から 1 つを指定します。省略値は RAPI_INT_MODE_0 です。

（ノイズキャンセル機能）　{ RAPI_FILTER_F1, RAPI_FILTER_F2, RAPI_FILTER_F4, RAPI_FILTER_F8 }から 1 つを指定します。省略値は RAPI_FILTER_F1 です。

(H8/300H)

RAPI_IRQ0	IRQ0 割り込みを使用する。
RAPI_IRQ1	IRQ1 割り込みを使用する。
RAPI_IRQ2	IRQ2 割り込みを使用する。
RAPI_IRQ3	IRQ3 割り込みを使用する。
RAPI_WKP	WKP 割り込みを使用する。
RAPI_NOT_INT_REQUEST	選択した外部割り込みの割り込み要求をしない。
RAPI_INT_REQUEST	選択した外部割り込みの割り込み要求をする。
RAPI_INT_RISING	選択した外部割り込みの有効エッジに立ち上がりエッジを指定する。
RAPI_INT_FALLING	選択した外部割り込みの有効エッジに立ち下がりエッジを指定する。
RAPI_WKP0_FALLING	立ち下がりエッジを有効エッジとする_WKP0 端子入力を許可する。
RAPI_WKP0_RISING	立ち上がりエッジを有効エッジとする_WKP0 端子入力を許可する。
RAPI_WKP1_FALLING	立ち下がりエッジを有効エッジとする_WKP1 端子入力を許可する。
RAPI_WKP1_RISING	立ち上がりエッジを有効エッジとする_WKP1 端子入力を許可する。
RAPI_WKP2_FALLING	立ち下がりエッジを有効エッジとする_WKP2 端子入力を許可する。
RAPI_WKP2_RISING	立ち上がりエッジを有効エッジとする_WKP2 端子入力を許可する。
RAPI_WKP3_FALLING	立ち下がりエッジを有効エッジとする_WKP3 端子入力を許可する。
RAPI_WKP3_RISING	立ち上がりエッジを有効エッジとする_WKP3 端子入力を許可する。
RAPI_WKP4_FALLING	立ち下がりエッジを有効エッジとする_WKP4 端子入力を許可する。
RAPI_WKP4_RISING	立ち上がりエッジを有効エッジとする_WKP4 端子入力を許可する。
RAPI_WKP5_FALLING	立ち下がりエッジを有効エッジとする_WKP5 端子入力を許可する。
RAPI_WKP5_RISING	立ち上がりエッジを有効エッジとする_WKP5 端子入力を許可する。

・IRQ0～3 割り込み使用時（RAPI_ IRQ0～RAPI_IRQ3 を指定時）に指定可能な定義値

(極性)　{ RAPI_INT_RISING, RAPI_INT_FALLING }から 1 つを指定します。
省略値は RAPI_INT_FALLING です。

(割り込み要求)　{ RAPI_NOT_INT_REQUEST, RAPI_INT_REQUEST }から 1 つを指定します。
省略値は RAPI_NOT_INT_REQUEST です。

・WKP 割り込み使用時（RAPI_WKP を指定時）に指定可能な定義値

(_WKP0 端子入力)　{ RAPI_WKP0_FALLING, RAPI_WKP0_RISING }から 1 つを指定します。
_WKP0 端子入力を指定しない場合は、「_WKP0 端子入力禁止」を設定します。

(_WKP1 端子入力)　{ RAPI_WKP1_FALLING, RAPI_WKP1_RISING }から 1 つを指定します。
_WKP1 端子入力を指定しない場合は、「_WKP1 端子入力禁止」を設定します。

(_WKP2 端子入力)　{ RAPI_WKP2_FALLING, RAPI_WKP2_RISING }から 1 つを指定します。
_WKP2 端子入力を指定しない場合は、「_WKP2 端子入力禁止」を設定します。

(_WKP3 端子入力)　{ RAPI_WKP3_FALLING, RAPI_WKP3_RISING }から 1 つを指定します。
_WKP3 端子入力を指定しない場合は、「_WKP3 端子入力禁止」を設定します。

(_WKP4 端子入力)　{ RAPI_WKP4_FALLING, RAPI_WKP4_RISING }から 1 つを指定します。
_WKP4 端子入力を指定しない場合は、「_WKP4 端子入力禁止」を設定します。

(_WKP5 端子入力)　{ RAPI_WKP5_FALLING, RAPI_WKP5_RISING }から 1 つを指定します。
_WKP5 端子入力を指定しない場合は、「_WKP5 端子入力禁止」を設定します。

[data2]

(M16C)(R8C)
割り込み制御レジスタへ設定する割り込みレベル(0～7)を指定します。

(H8S)
インタラプトプライオリティレジスタへ設定する割り込み優先レベル(0～3)を指定します。

(H8/300H)
割り込みコントロールレジスタへ設定する割り込みプライオリティレベル(0～1)を指定します。割り込みコントロールレジスタを持たない CPU の場合は 0 を指定します。

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、それ以外は RAPI_TRUE を返します。

機能

外部割り込み

参照

__EnableInterrupt, __GetInterruptFlag, __ClearInterruptFlag

備考

・第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・指定できる外部割り込みは CPU により異なります。
・R8C で INT2 および INT3 へ割り込み極性に両エッジを指定できない CPU の場合は、INT2 および INT3 への RAPI_INT_BOTH の指定はできません。

プログラム例

```
#include " rapi_interrupt_r8c_13.h"

void IntFunc( void ){}
```

```
void func( void )
{
    /* Set up _INT0 interrupt */
    __SetInterrupt( RAPI_INT0|RAPI_INT_FALLING, 0, IntFunc );
}
```

__EnableInterrupt

要約

< 外部割り込みの制御 >

Boolean __EnableInterrupt(unsigned long data1, unsigned int data2)

data1	設定データ 1（機種により内容が異なる）
data2	設定データ 2（機種により内容が異なる）

解説

指定した外部割り込みの動作条件の変更をおこないます。

[data1]

data1 には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。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。

(M16C)

RAPI_INT0	_INT0 割り込みを選択する。
RAPI_INT1	_INT1 割り込みを選択する。
RAPI_INT2	_INT2 割り込みを選択する。
RAPI_INT3	_INT3 割り込みを選択する。
RAPI_INT4	_INT4 割り込みを選択する。
RAPI_INT5	_INT5 割り込みを選択する。
RAPI_KEY	キー入力割り込みを選択する。

(R8C)

RAPI_INT0	_INT0 割り込みを選択する。
RAPI_INT1	_INT1 割り込みを選択する。
RAPI_INT2	_INT2 割り込みを選択する。
RAPI_INT3	_INT3 割り込みを選択する。
RAPI_KEY	キー入力割り込みを選択する。

(H8S)

RAPI_IRQ0	IRQ0 割り込みを選択する。
RAPI_IRQ1	IRQ1 割り込みを選択する。
RAPI_IRQ2	IRQ2 割り込みを選択する。
RAPI_IRQ3	IRQ3 割り込みを選択する。
RAPI_IRQ4	IRQ4 割り込みを選択する。
RAPI_IRQ5	IRQ5 割り込みを選択する。
RAPI_IRQ6	IRQ6 割り込みを選択する。
RAPI_IRQ7	IRQ7 割り込みを選択する。
RAPI_NOT_INT_REQUEST	選択した外部割り込みの割り込み要求をしない。（省略値）
RAPI_INT_REQUEST	選択した外部割り込みの割り込み要求をする。
RAPI_INT_MODE_0	割り込み制御モード 0 を指定する。
RAPI_INT_MODE_2	割り込み制御モード 2 を指定する。

(H8/300H)

RAPI_IRQ0	IRQ0 割り込みを選択する。

RAPI_IRQ1	IRQ1 割り込みを選択する。
RAPI_IRQ2	IRQ2 割り込みを選択する。
RAPI_IRQ3	IRQ3 割り込みを選択する。
RAPI_WKP	WKP 割り込みを選択する。
RAPI_NOT_INT_REQUEST	選択した外部割り込みの割り込み要求をしない。(省略値)
RAPI_INT_REQUEST	選択した外部割り込みの割り込み要求をする。

[data2]

(M16C)(R8C)

割り込み制御レジスタへ設定する割り込みレベル(0～7)を指定します。

(H8S)

インタラプトプライオリティレジスタへ設定する割り込み優先レベル(0～3)を指定します。

(H8/300H)

割り込みコントロールレジスタへ設定する割り込みプライオリティレベル(0～1)を指定します。割り込みコントロールレジスタを持たない CPU の場合は 0 を指定します。

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、それ以外は RAPI_TRUE を返します。

機能

外部割り込み

参照

__SetInterrupt, __GetInterruptFlag, __ClearInterruptFlag

備考

・ 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・ 指定できる外部割り込みは CPU により異なります。

プログラム例

```
#include " rapi_interrupt_r8c_13.h"

void func( void )
{
    /* Activate _INT1 interrupt ( interrupt priority level 5 ) */
    __EnableInterrupt(RAPI_INT1, 5 );
}
```

__GetInterruptFlag

要約

〈外部割り込みフラグの状態取得 〉

Boolean __GetInterruptFlag(unsigned long data1, unsigned int *data2))

data1	設定データ 1（機種により内容が異なる）
data2	取得したフラグデータを格納するバッファポインタ

解説

指定した外部割り込みの割り込み要求フラグの値を取得します。

[data1]

(M16C)

RAPI_INT0	_INT0 割り込みを選択する。
RAPI_INT1	_INT1 割り込みを選択する。
RAPI_INT2	_INT2 割り込みを選択する。
RAPI_INT3	_INT3 割り込みを選択する。
RAPI_INT4	_INT4 割り込みを選択する。
RAPI_INT5	_INT5 割り込みを選択する。
RAPI_KEY	キー入力割り込みを選択する。

(R8C)

RAPI_INT0	_INT0 割り込みを選択する。
RAPI_INT1	_INT1 割り込みを選択する。
RAPI_INT2	_INT2 割り込みを選択する。
RAPI_INT3	_INT3 割り込みを選択する。
RAPI_KEY	キー入力割り込みを選択する。

(H8S)

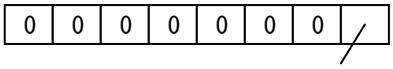
RAPI_IRQ0	IRQ0 割り込みを選択する。
RAPI_IRQ1	IRQ1 割り込みを選択する。
RAPI_IRQ2	IRQ2 割り込みを選択する。
RAPI_IRQ3	IRQ3 割り込みを選択する。
RAPI_IRQ4	IRQ4 割り込みを選択する。
RAPI_IRQ5	IRQ5 割り込みを選択する。
RAPI_IRQ6	IRQ6 割り込みを選択する。
RAPI_IRQ7	IRQ7 割り込みを選択する。

(H8/300H)

RAPI_IRQ0	IRQ0 割り込みを選択する。
RAPI_IRQ1	IRQ1 割り込みを選択する。
RAPI_IRQ2	IRQ2 割り込みを選択する。
RAPI_IRQ3	IRQ3 割り込みを選択する。
RAPI_WKP0	WKP0 割り込みを選択する。
RAPI_WKP1	WKP1 割り込みを選択する。
RAPI_WKP2	WKP2 割り込みを選択する。
RAPI_WKP3	WKP3 割り込みを選択する。
RAPI_WKP4	WKP4 割り込みを選択する。

RAPI_WKP5	WKP5 割り込みを選択する。

[data2]



割り込み要求フラグの値（0:割り込み要求なし，1:割り込み要求あり）

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、それ以外は RAPI_TRUE を返します。

機能　外部割り込み

参照　__SetInterrupt, __EnableInterrupt, __ClearInterruptFlag

備考
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できる外部割り込みは CPU により異なります。

プログラム例

```
#include " rapi_interrupt_r8c_13.h"

void func( void )
{
    unsigned char data;

    /* Get flag of _INT2 interrupt */
    __GetInterruptFlag( IS_INT2, &data );
}
```

__ClearInterruptFlag

要約

< 外部割り込みフラグのクリア >

Boolean __ClearInterruptFlag(unsigned long data)

data	設定データ（機種により内容が異なる）

解説

指定した外部割り込みの割り込み要求フラグをクリアします。

[data]

(M16C)

RAPI_INT0	_INT0 割り込みを選択する。
RAPI_INT1	_INT1 割り込みを選択する。
RAPI_INT2	_INT2 割り込みを選択する。
RAPI_INT3	_INT3 割り込みを選択する。
RAPI_INT4	_INT4 割り込みを選択する。
RAPI_INT5	_INT5 割り込みを選択する。
RAPI_KEY	キー入力割り込みを選択する。

(R8C)

RAPI_INT0	_INT0 割り込みを選択する。
RAPI_INT1	_INT1 割り込みを選択する。
RAPI_INT2	_INT2 割り込みを選択する。
RAPI_INT3	_INT3 割り込みを選択する。
RAPI_KEY	キー入力割り込みを選択する。

(H8S)

RAPI_IRQ0	IRQ0 割り込みを選択する。
RAPI_IRQ1	IRQ1 割り込みを選択する。
RAPI_IRQ2	IRQ2 割り込みを選択する。
RAPI_IRQ3	IRQ3 割り込みを選択する。
RAPI_IRQ4	IRQ4 割り込みを選択する。
RAPI_IRQ5	IRQ5 割り込みを選択する。
RAPI_IRQ6	IRQ6 割り込みを選択する。
RAPI_IRQ7	IRQ7 割り込みを選択する。

(H8/300H)

RAPI_IRQ0	IRQ0 割り込みを選択する。
RAPI_IRQ1	IRQ1 割り込みを選択する。
RAPI_IRQ2	IRQ2 割り込みを選択する。
RAPI_IRQ3	IRQ3 割り込みを選択する。
RAPI_WKP0	WKP0 割り込みを選択する。
RAPI_WKP1	WKP1 割り込みを選択する。
RAPI_WKP2	WKP2 割り込みを選択する。
RAPI_WKP3	WKP3 割り込みを選択する。
RAPI_WKP4	WKP4 割り込みを選択する。
RAPI_WKP5	WKP5 割り込みを選択する。

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、それ以外は RAPI_TRUE を返します。

機能　外部割り込み

参照　__SetInterrupt, __EnableInterrupt, __GetInterruptFlag

備考
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できる外部割り込みは CPU により異なります。

プログラム例

```
#include " rapi_interrupt_r8c_13.h"

void func( void )
{
    /* Clear status of _INT0 interrupt */
    __ClearInterruptFlag( RAPI_INT0 );
}
```

4.2.5 A/D変換器
__CreateADC

要約

< A/D 変換器の設定 >

Boolean __CreateADC(unsigned long data1, unsigned int data2, unsigned int data3, void* func)

data1	設定データ 1（機種により内容が異なる）
data2	A/D 変換で使用するアナログ入力端子数（機種により内容が異なる）
data3	設定データ 3（機種により内容が異なる）
func	コールバック関数ポインタ （コールバック関数を設定しない場合は 0 を指定する）

解説

A/D 変換器を指定したモード・動作条件に設定します。

[data1]

data1 には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。ただし、複数のアナログ入力端子の定義値を同時に指定することはできません。

(M16C)

RAPI_ONE_SHOT	単発モードを選択する。
RAPI_REPEAT	繰り返しモードを選択する。
RAPI_SINGLE_SWEEP	単掃引モードを選択する。
RAPI_REPEAT_SWEEP0	繰り返し掃引モード 0 を選択する。
RAPI_REPEAT_SWEEP1	繰り返し掃引モード 1 を選択する。
RAPI_SIMULTANEOUS_SAMPLE_SWEEP	同時サンプル掃引モードを選択する。
RAPI_DELAYED_TRIGGER0	遅延トリガモード 0 を選択する。
RAPI_DELAYED_TRIGGER1	遅延トリガモード 1 を選択する。
RAPI_AN0	アナログ入力端子に AN_0 端子を使用する。
RAPI_AN1	アナログ入力端子に AN_1 端子を使用する。
RAPI_AN2	アナログ入力端子に AN_2 端子を使用する。
RAPI_AN3	アナログ入力端子に AN_3 端子を使用する。
RAPI_AN4	アナログ入力端子に AN_4 端子を使用する。
RAPI_AN5	アナログ入力端子に AN_5 端子を使用する。
RAPI_AN6	アナログ入力端子に AN_6 端子を使用する。
RAPI_AN7	アナログ入力端子に AN_7 端子を使用する。
RAPI_AN00	アナログ入力端子に AN_{00} 端子を使用する。
RAPI_AN01	アナログ入力端子に AN_{01} 端子を使用する。
RAPI_AN02	アナログ入力端子に AN_{02} 端子を使用する。
RAPI_AN03	アナログ入力端子に AN_{03} 端子を使用する。
RAPI_AN04	アナログ入力端子に AN_{04} 端子を使用する。
RAPI_AN05	アナログ入力端子に AN_{05} 端子を使用する。

RAPI_AN06	アナログ入力端子に AN_{06} 端子を使用する。
RAPI_AN07	アナログ入力端子に AN_{07} 端子を使用する。
RAPI_AN20	アナログ入力端子に AN_{20} 端子を使用する。
RAPI_AN21	アナログ入力端子に AN_{21} 端子を使用する。
RAPI_AN22	アナログ入力端子に AN_{22} 端子を使用する。
RAPI_AN23	アナログ入力端子に AN_{23} 端子を使用する。
RAPI_AN24	アナログ入力端子に AN_{24} 端子を使用する。
RAPI_AN25	アナログ入力端子に AN_{25} 端子を使用する。
RAPI_AN26	アナログ入力端子に AN_{26} 端子を使用する。
RAPI_AN27	アナログ入力端子に AN_{27} 端子を使用する。
RAPI_AN30	アナログ入力端子に AN_{30} 端子を使用する。
RAPI_AN31	アナログ入力端子に AN_{31} 端子を使用する。
RAPI_AN32	アナログ入力端子に AN_{32} 端子を使用する。
RAPI_P0_GROUP	アナログ入力端子にポート P_0 グループを使用する。
RAPI_P10_GROUP	アナログ入力端子にポート P_{10} グループを使用する。
RAPI_P1P9_GROUP	アナログ入力端子にポート P_1 /P_9 グループを使用する。
RAPI_P9_GROUP	アナログ入力端子にポート P_9 グループを使用する。
RAPI_FAD	A/D 変換器の動作周波数を f_{AD} に設定する。
RAPI_FAD2	A/D 変換器の動作周波数を f_{AD} の 2 分周に設定する。
RAPI_FAD3	A/D 変換器の動作周波数を f_{AD} の 3 分周に設定する。
RAPI_FAD4	A/D 変換器の動作周波数を f_{AD} の 4 分周に設定する。
RAPI_FAD6	A/D 変換器の動作周波数を f_{AD} の 6 分周に設定する。
RAPI_FAD12	A/D 変換器の動作周波数を f_{AD} の 12 分周に設定する。
RAPI_10BIT	A/D 変換器の分解能を 10 ビットに設定する。
RAPI_8BIT	A/D 変換器の分解能を 8 ビットに設定する。
RAPI_AD_ON	A/D 変換器を動作開始に設定します。
RAPI_AD_OFF	A/D 変換器を動作停止に設定します。
RAPI_WITH_SAMPLE_HOLD	サンプル＆ホールドありに設定する。
RAPI_WITHOUT_SAMPLE_HOLD	サンプル＆ホールドなしに設定する。
RAPI_SOFTWARE_TRIGGER	ソフトウェアトリガを選択する。
RAPI_EXTERNAL_TRIGGER	外部トリガを選択する。
RAPI_TIMER_B0_TRIGGER	タイマ B0 のアンダフローをトリガに選択する。
RAPI_TIMER_B2_TRIGGER	タイマ B2 のアンダフローをトリガに選択する。
RAPI_ICTB2_TRIGGER	タイマ B2 割り込み発生頻度設定カウンタの アンダフローをトリガに選択する。

・単発モード使用時（RAPI_ONE_SHOT を指定時）に指定可能な定義値

（入力端子）　{ RAPI_AN0, RAPI_AN1, RAPI_AN2, RAPI_AN3, RAPI_AN4, RAPI_AN5,

RAPI_AN6, RAPI_AN7, RAPI_AN00, RAPI_AN01, RAPI_AN02, RAPI_AN03,
RAPI_AN04, RAPI_AN05, RAPI_AN06, RAPI_AN07, RAPI_AN20, RAPI_AN21,
RAPI_AN22, RAPI_AN23, RAPI_AN24, RAPI_AN25, RAPI_AN26, RAPI_AN27,
RAPI_AN30, RAPI_AN31, RAPI_AN32 }
から１つを指定します。省略値は RAPI_AN0 です。

(動作周波数) { RAPI_FAD, RAPI_FAD2, RAPI_FAD3, RAPI_FAD4, RAPI_FAD6, RAPI_FAD12 }
から１つを指定します。省略値は RAPI_FAD4 です。

(分解能) { RAPI_10BIT, RAPI_8BIT }から１つを指定します。
省略値は RAPI_8BIT です。

(設定時動作状態) { RAPI_AD_ON, RAPI_AD_OFF }から１つを指定します。
省略値は RAPI_AD_OFF です。

(変換方式) { RAPI_WITH_SAMPLE_HOLD, RAPI_WITHOUT_SAMPLE_HOLD }から１つを
指定します。省略値は RAPI_WITHOUT_SAMPLE_HOLD です。

(トリガ) { RAPI_SOFTWARE_TRIGGER, RAPI_EXTERNAL_TRIGGER }から１つを
指定します。省略値は RAPI_SOFTWARE_TRIGGER です。

・繰り返しモード使用時（RAPI_REPEAT を指定時）に指定可能な定義値

(入力端子) { RAPI_AN0, RAPI_AN1, RAPI_AN2, RAPI_AN3, RAPI_AN4, RAPI_AN5,
RAPI_AN6, RAPI_AN7, RAPI_AN00, RAPI_AN01, RAPI_AN02, RAPI_AN03,
RAPI_AN04, RAPI_AN05, RAPI_AN06, RAPI_AN07, RAPI_AN20, RAPI_AN21,
RAPI_AN22, RAPI_AN23, RAPI_AN24, RAPI_AN25, RAPI_AN26, RAPI_AN27,
RAPI_AN30, RAPI_AN31, RAPI_AN32 }
から１つを指定します。省略値は RAPI_AN0 です。

(動作周波数) { RAPI_FAD, RAPI_FAD2, RAPI_FAD3, RAPI_FAD4, RAPI_FAD6, RAPI_FAD12 }
から１つを指定します。省略値は RAPI_FAD4 です。

(分解能) { RAPI_10BIT, RAPI_8BIT }から１つを指定します。
省略値は RAPI_8BIT です。

(設定時動作状態) { RAPI_AD_ON, RAPI_AD_OFF }から１つを指定します。
省略値は RAPI_AD_OFF です。

(変換方式) { RAPI_WITH_SAMPLE_HOLD, RAPI_WITHOUT_SAMPLE_HOLD }から１つを
指定します。省略値は RAPI_WITHOUT_SAMPLE_HOLD です。

(トリガ) { RAPI_SOFTWARE_TRIGGER, RAPI_EXTERNAL_TRIGGER }から１つを
指定します。省略値は RAPI_SOFTWARE_TRIGGER です。

・単掃引モード使用時（RAPI_SINGLE_SWEEP を指定時）に指定可能な定義値

(入力端子) { RAPI_P0_GROUP, RAPI_P10_GROUP, RAPI_P1P9_GROUP, RAPI_P9_GROUP }
から１つを指定します。省略値は RAPI_P10_GROUP です。

(動作周波数) { RAPI_FAD, RAPI_FAD2, RAPI_FAD3, RAPI_FAD4, RAPI_FAD6, RAPI_FAD12 }
から１つを指定します。省略値は RAPI_FAD4 です。

(分解能) { RAPI_10BIT, RAPI_8BIT }から１つを指定します。
省略値は RAPI_8BIT です。

(設定時動作状態) { RAPI_AD_ON, RAPI_AD_OFF }から１つを指定します。
省略値は RAPI_AD_OFF です。

(変換方式) { RAPI_WITH_SAMPLE_HOLD, RAPI_WITHOUT_SAMPLE_HOLD }から１つを
指定します。省略値は RAPI_WITHOUT_SAMPLE_HOLD です。

(トリガ) { RAPI_SOFTWARE_TRIGGER, RAPI_EXTERNAL_TRIGGER }から１つを
指定します。省略値は RAPI_SOFTWARE_TRIGGER です。

・繰り返し掃引モード０使用時（RAPI_REPEAT_SWEEP0 を指定時）に指定可能な定義値

（入力端子）　{ RAPI_P0_GROUP, RAPI_P10_GROUP, RAPI_P1P9_GROUP, RAPI_P9_GROUP }
から１つを指定します。省略値は RAPI_P10_GROUP です。

（動作周波数）　{ RAPI_FAD, RAPI_FAD2, RAPI_FAD3, RAPI_FAD4, RAPI_FAD6, RAPI_FAD12 }
から１つを指定します。省略値は RAPI_FAD4 です。

（分解能）　{ RAPI_10BIT, RAPI_8BIT }から１つを指定します。
省略値は RAPI_8BIT です。

（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から１つを指定します。
省略値は RAPI_AD_OFF です。

（変換方式）　{ RAPI_WITH_SAMPLE_HOLD, RAPI_WITHOUT_SAMPLE_HOLD }から１つを
指定します。省略値は RAPI_WITHOUT_SAMPLE_HOLD です。

（トリガ）　{ RAPI_SOFTWARE_TRIGGER, RAPI_EXTERNAL_TRIGGER }から１つを
指定します。省略値は RAPI_SOFTWARE_TRIGGER です。

・繰り返し掃引モード１使用時（RAPI_REPEAT_SWEEP1 を指定時）に指定可能な定義値

（入力端子）　{ RAPI_P0_GROUP, RAPI_P10_GROUP, RAPI_P1P9_GROUPP, RAPI_P9_GROUP }
から１つを指定します。省略値は RAPI_P10_GROUP です。

（動作周波数）　{ RAPI_FAD, RAPI_FAD2, RAPI_FAD3, RAPI_FAD4, RAPI_FAD6, RAPI_FAD12 }
から１つを指定します。省略値は RAPI_FAD4 です。

（分解能）　{ RAPI_10BIT, RAPI_8BIT }から１つを指定します。
省略値は RAPI_8BIT です。

（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から１つを指定します。
省略値は RAPI_AD_OFF です。

（変換方式）　{ RAPI_WITH_SAMPLE_HOLD, RAPI_WITHOUT_SAMPLE_HOLD }から１つを
指定します。省略値は RAPI_WITHOUT_SAMPLE_HOLD です。

（トリガ）　{ RAPI_SOFTWARE_TRIGGER, RAPI_EXTERNAL_TRIGGER }から１つを
指定します。省略値は RAPI_SOFTWARE_TRIGGER です。

・同時サンプル掃引モード使用時（RAPI_SIMULTANEOUS_SAMPLE_SWEEP を指定時）に指定可能な定義値

（入力端子）　{ RAPI_P0_GROUP, RAPI_P10_GROUP, RAPI_P1P9_GROUPP, RAPI_P9_GROUP }
から１つを指定します。省略値は RAPI_P10_GROUP です。

（動作周波数）　{ RAPI_FAD, RAPI_FAD2, RAPI_FAD3, RAPI_FAD4, RAPI_FAD6, RAPI_FAD12 }
から１つを指定します。省略値は RAPI_FAD4 です。

（分解能）　{ RAPI_10BIT, RAPI_8BIT }から１つを指定します。
省略値は RAPI_8BIT です。

（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から１つを指定します。
省略値は RAPI_AD_OFF です。

（トリガ）　{ RAPI_SOFTWARE_TRIGGER, RAPI_EXTERNAL_TRIGGER,
RAPI_TIMER_B0_TRIGGER, RAPI_TIMER_B2_TRIGGER, RAPI_ICTB2_TRIGGER }
から１つを指定します。省略値は RAPI_SOFTWARE_TRIGGER です。

・遅延トリガモード０使用時（RAPI_DELAYED_TRIGGER0 を指定時）に指定可能な定義値

（入力端子）　{ RAPI_P0_GROUP, RAPI_P10_GROUP, RAPI_P1P9_GROUPP, RAPI_P9_GROUP }
から１つを指定します。省略値は RAPI_P10_GROUP です。

（動作周波数）　{ RAPI_FAD, RAPI_FAD2, RAPI_FAD3, RAPI_FAD4, RAPI_FAD6, RAPI_FAD12 }
から１つを指定します。省略値は RAPI_FAD4 です。

（分解能）　{ RAPI_10BIT, RAPI_8BIT }から１つを指定します。

省略値は RAPI_8BIT です。

（変換方式）　RAPI_WITH_SAMPLE_HOLD を指定します。

・遅延トリガモード１使用時（RAPI_DELAYED_TRIGGER1 を指定時）に指定可能な定義値

（入力端子）　{ RAPI_P0_GROUP, RAPI_P10_GROUP, RAPI_P1P9_GROUPP, RAPI_P9_GROUP }
から１つを指定します。省略値は RAPI_P10_GROUP です。

（動作周波数）　{ RAPI_FAD, RAPI_FAD2, RAPI_FAD3, RAPI_FAD4, RAPI_FAD6, RAPI_FAD12 }
から１つを指定します。省略値は RAPI_FAD4 です。

（分解能）　{ RAPI_10BIT, RAPI_8BIT }から１つを指定します。
省略値は RAPI_8BIT です。

（変換方式）　RAPI_WITH_SAMPLE_HOLD を指定します。

(R8C)

RAPI_ONE_SHOT	単発モードを選択する。
RAPI_REPEAT	繰り返しモードを選択する。
RAPI_SINGLE_SWEEP	単掃引モードを選択する。
RAPI_REPEAT_SWEEP	繰り返し掃引モード 0 を選択する。
RAPI_AN0	アナログ入力端子に AN0 端子を使用する。
RAPI_AN1	アナログ入力端子に AN1 端子を使用する。
RAPI_AN2	アナログ入力端子に AN2 端子を使用する。
RAPI_AN3	アナログ入力端子に AN3 端子を使用する。
RAPI_AN4	アナログ入力端子に AN4 端子を使用する。
RAPI_AN5	アナログ入力端子に AN5 端子を使用する。
RAPI_AN6	アナログ入力端子に AN6 端子を使用する。
RAPI_AN7	アナログ入力端子に AN7 端子を使用する。
RAPI_AN8	アナログ入力端子に AN8 端子を使用する。
RAPI_AN9	アナログ入力端子に AN9 端子を使用する。
RAPI_AN10	アナログ入力端子に AN10 端子を使用する。
RAPI_AN11	アナログ入力端子に AN11 端子を使用する。
RAPI_AN12	アナログ入力端子に AN12 端子を使用する。
RAPI_AN13	アナログ入力端子に AN13 端子を使用する。
RAPI_AN14	アナログ入力端子に AN14 端子を使用する。
RAPI_AN15	アナログ入力端子に AN15 端子を使用する。
RAPI_AN16	アナログ入力端子に AN16 端子を使用する。
RAPI_AN17	アナログ入力端子に AN17 端子を使用する。
RAPI_AN18	アナログ入力端子に AN18 端子を使用する。
RAPI_AN19	アナログ入力端子に AN19 端子を使用する。
RAPI_AN12_GROUP	アナログ入力端子に AN12 グループを使用する。
RAPI_AN16_GROUP	アナログ入力端子に AN16 グループを使用する。
RAPI_FAD	A/D 変換器の動作周波数を f_{AD} に設定する。
RAPI_FAD2	A/D 変換器の動作周波数を f_{AD} の 2 分周に設定する。
RAPI_FAD4	A/D 変換器の動作周波数を f_{AD} の 4 分周に設定する。
RAPI_FOCOF	A/D 変換器の動作周波数を fOCO-F に設定する。
RAPI_10BIT	A/D 変換器の分解能を 10 ビットに設定する。
RAPI_8BIT	A/D 変換器の分解能を 8 ビットに設定する。

RAPI_AD_ON	__CreateADC 内で A/D 変換器を動作開始に設定します。
RAPI_AD_OFF	__CreateADC 内では A/D 変換器を動作停止に設定します。
RAPI_WITH_SAMPLE_HOLD	サンプル＆ホールドありに設定する。
RAPI_WITHOUT_SAMPLE_HOLD	サンプル＆ホールドなしに設定する。
RAPI_SOFTWARE_TRIGGER	ソフトウェアトリガを選択する。
RAPI_TIMER_RD_TRIGGER	タイマ RD をトリガに選択する。

・単発モード使用時（RAPI_ONE_SHOT を指定時）に指定可能な定義値

（入力端子） { RAPI_AN0, RAPI_AN1, RAPI_AN2, RAPI_AN3, RAPI_AN4, RAPI_AN5, RAPI_AN6, RAPI_AN7, RAPI_AN8, RAPI_AN9, RAPI_AN10, RAPI_AN11, RAPI_AN12, RAPI_AN13, RAPI_AN14, RAPI_AN15, RAPI_AN16, RAPI_AN17, RAPI_AN18, RAPI_AN19}から 1 つを指定します。
入力端子は必ず指定してください。

（動作周波数） { RAPI_FAD, RAPI_FAD2, RAPI_FAD4, RAPI_FOCOF }から 1 つを指定します。
省略値は RAPI_FAD4 です。

（分解能） { RAPI_10BIT, RAPI_8BIT }から 1 つを指定します。
省略値は RAPI_8BIT です。

（設定時動作状態） { RAPI_AD_ON, RAPI_AD_OFF }から 1 つを指定します。
省略値は RAPI_AD_OFF です。

（変換方式） { RAPI_WITH_SAMPLE_HOLD, RAPI_WITHOUT_SAMPLE_HOLD }から 1 つを指定します。省略値は RAPI_WITHOUT_SAMPLE_HOLD です。

（トリガ） {RAPI_SOFTWARE_TRIGGER, RAPI_TIMER_RD_TRIGGER}から 1 つを指定します。
省略値は RAPI_SOFTWARE_TRIGGER です。
R8C/13、R8C/26,27, R8C/28,29 は指定できません。

・繰り返しモード使用時（RAPI_REPEAT を指定時）に指定可能な定義値

（入力端子） { RAPI_AN0, RAPI_AN1, RAPI_AN2, RAPI_AN3, RAPI_AN4, RAPI_AN5, RAPI_AN6, RAPI_AN7, RAPI_AN8, RAPI_AN9, RAPI_AN10, RAPI_AN11, RAPI_AN12, RAPI_AN13, RAPI_AN14, RAPI_AN15, RAPI_AN16, RAPI_AN17, RAPI_AN18, RAPI_AN19}から 1 つを指定します。
入力端子は必ず指定してください。

（動作周波数） { RAPI_FAD, RAPI_FAD2, RAPI_FAD4 }から 1 つを指定します。
省略値は RAPI_FAD4 です。

（分解能） { RAPI_10BIT, RAPI_8BIT }から 1 つを指定します。
省略値は RAPI_8BIT です。

（設定時動作状態） { RAPI_AD_ON, RAPI_AD_OFF }から 1 つを指定します。
省略値は RAPI_AD_OFF です。

（変換方式） { RAPI_WITH_SAMPLE_HOLD, RAPI_WITHOUT_SAMPLE_HOLD }から 1 つを指定します。省略値は RAPI_WITHOUT_SAMPLE_HOLD です。

（トリガ） {RAPI_SOFTWARE_TRIGGER, RAPI_TIMER_RD_TRIGGER}から 1 つを指定します。
省略値は RAPI_SOFTWARE_TRIGGER です。
R8C/13、R8C/26,27, R8C/28,29 は指定できません。

・単掃引モード使用時（RAPI_SINGLE_SWEEP を指定時）に指定可能な定義値

（入力端子）　{ RAPI_AN12_GROUP, RAPI_AN16_GROUP}から１つを指定します。
入力端子は必ず指定してください。

（動作周波数）　{ RAPI_FAD, RAPI_FAD2, RAPI_FAD4 }から１つを指定します。
省略値は RAPI_FAD4 です。

（分解能）　{ RAPI_10BIT, RAPI_8BIT }から１つを指定します。
省略値は RAPI_8BIT です。

（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から１つを指定します。
省略値は RAPI_AD_OFF です。

（トリガ）　{RAPI_SOFTWARE_TRIGGER, RAPI_TIMER_RD_TRIGGER}から１つを指定します。
省略値は RAPI_SOFTWARE_TRIGGER です。
R8C/13、R8C/26,27, R8C/28,29 は指定できません。

（変換方式）　{ RAPI_WITH_SAMPLE_HOLD, RAPI_WITHOUT_SAMPLE_HOLD }から１つを
指定します。省略値は RAPI_WITHOUT_SAMPLE_HOLD です。

・繰り返し掃引モード０使用時（RAPI_REPEAT_SWEEP を指定時）に指定可能な定義値

（入力端子）　{ RAPI_AN12_GROUP, RAPI_AN16_GROUP}から１つを指定します。
入力端子は必ず指定してください。

（動作周波数）　{ RAPI_FAD, RAPI_FAD2, RAPI_FAD4 }から１つを指定します。
省略値は RAPI_FAD4 です。

（分解能）　{ RAPI_10BIT, RAPI_8BIT }から１つを指定します。
省略値は RAPI_8BIT です。

（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から１つを指定します。
省略値は RAPI_AD_OFF です。

（変換方式）　{ RAPI_WITH_SAMPLE_HOLD, RAPI_WITHOUT_SAMPLE_HOLD }から１つを
指定します。省略値は RAPI_WITHOUT_SAMPLE_HOLD です。

（トリガ）　{RAPI_SOFTWARE_TRIGGER, RAPI_TIMER_RD_TRIGGER}から１つを指定します。
省略値は RAPI_SOFTWARE_TRIGGER です。
R8C/13、R8C/26,27, R8C/28,29 は指定できません。

(H8S)

RAPI_ONE_SHOT	単発モードを選択する。
RAPI_REPEAT	繰り返しモードを選択する。
RAPI_SINGLE_SWEEP	単掃引モードを選択する。
RAPI_REPEAT_SWEEP0	繰り返し掃引モード０を選択する。
RAPI_AD_UNIT_1	A/D 変換器ユニット１を使用する。
RAPI_AD_UNIT_2	A/D 変換器ユニット２を使用する。
RAPI_AN0	アナログ入力端子に AN0 端子を使用する。
RAPI_AN1	アナログ入力端子に AN1 端子を使用する。
RAPI_AN2	アナログ入力端子に AN2 端子を使用する。
RAPI_AN3	アナログ入力端子に AN3 端子を使用する。
RAPI_AN4	アナログ入力端子に AN4 端子を使用する。
RAPI_AN5	アナログ入力端子に AN5 端子を使用する。
RAPI_AN6	アナログ入力端子に AN6 端子を使用する。
RAPI_AN7	アナログ入力端子に AN7 端子を使用する。
RAPI_AN8	アナログ入力端子に AN8 端子を使用する。

RAPI_AN9	アナログ入力端子に AN9 端子を使用する。
RAPI_AN10	アナログ入力端子に AN10 端子を使用する。
RAPI_AN11	アナログ入力端子に AN11 端子を使用する。
RAPI_AN0_GROUP	アナログ入力端子に AN0～AN3 のグループを使用する。
RAPI_AN4_GROUP	アナログ入力端子に AN4～AN7 のグループを使用する。
RAPI_AN8_GROUP	アナログ入力端子に AN8～AN11 のグループを使用する。
RAPI_84_STATES	変換時間 = 84 ステート
RAPI_43_STATES	変換時間 = 43 ステート
RAPI_AD_ON	__CreateADC 内で A/D 変換器を動作開始に設定します。
RAPI_AD_OFF	__CreateADC 内では A/D 変換器を動作停止に設定します。
RAPI_ADTRG1_TRIGGER	A/D 変換開始条件に ADTRG1 端子からの外部トリガを含める。
RAPI_ADTRG2_TRIGGER	A/D 変換開始条件に ADTRG2 端子からの外部トリガを含める。
RAPI_TIMER_RC_ TRIGGER	A/D 変換開始条件にタイマ RC からのトリガを含める。
RAPI_TIMER_RD_0_ TRIGGER	A/D 変換開始条件にタイマ RD ユニット 0 からのトリガを含める。
RAPI_TIMER_RD_1_ TRIGGER	A/D 変換開始条件にタイマ RD ユニット 1 からのトリガを含める。
RAPI_AD_INT_REQUEST	A/D 変換割り込みの割り込み要求をする。
RAPI_NOT_AD_INT_ REQUEST	A/D 変換割り込みの割り込み要求をしない。

・単発モード使用時（RAPI_ONE_SHOT を指定時）に指定可能な定義値

（ユニット）　{ RAPI_AD_UNIT_1, RAPI_AD_UNIT_2 }から 1 つを指定します。
省略値は RAPI_AD_UNIT_1 です。

（入力端子）　{ RAPI_AN0, RAPI_AN1, RAPI_AN2, RAPI_AN3, RAPI_AN4, RAPI_AN5, RAPI_AN6, RAPI_AN7, RAPI_AN8, RAPI_AN9, RAPI_AN10, RAPI_AN11 }
から 1 つを指定します。省略値は RAPI_AN0 です。

（変換時間）　{ RAPI_84_STATES, RAPI_43_STATES }から 1 つを指定します。

（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から 1 つを指定します。
省略値は RAPI_AD_OFF です。

（トリガ）　{ RAPI_ADTRG1_TRIGGER, RAPI_ADTRG2_TRIGGER, RAPI_TIMER_RC_TRIGGER, RAPI_TIMER_RD_0_TRIGGER, RAPI_TIMER_RD_1_TRIGGER }
から 1 つを指定します。トリガを指定しない場合は、「トリガ入力禁止」に設定します。

（割り込み要求）　{RAPI_AD_INT_REQUEST, RAPI_NOT_AD_INT_REQUEST}から 1 つを指定します。
省略値は RAPI_NOT_AD_INT_REQUEST です。

・繰り返しモード使用時（RAPI_REPEAT を指定時）に指定可能な定義値

（ユニット）　{ RAPI_AD_UNIT_1, RAPI_AD_UNIT_2 }から 1 つを指定します。
省略値は RAPI_AD_UNIT_1 です。

（入力端子）　{ RAPI_AN0, RAPI_AN1, RAPI_AN2, RAPI_AN3, RAPI_AN4, RAPI_AN5, RAPI_AN6, RAPI_AN7, RAPI_AN8, RAPI_AN9, RAPI_AN10, RAPI_AN11 }
から 1 つを指定します。省略値は RAPI_AN0 です。

（変換時間）　{ RAPI_84_STATES, RAPI_43_STATES }から 1 つを指定します。

（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から 1 つを指定します。

省略値は RAPI_AD_OFF です。

（トリガ）　{ RAPI_ADTRG1_TRIGGER, RAPI_ADTRG2_TRIGGER, RAPI_TIMER_RC_TRIGGER, RAPI_TIMER_RD_0_TRIGGER, RAPI_TIMER_RD_1_TRIGGER }
から 1 つを指定します。トリガを指定しない場合は、「トリガ入力禁止」に設定します。

・単掃引モード使用時（RAPI_SINGLE_SWEEP を指定時）に指定可能な定義値

（ユニット）　{ RAPI_AD_UNIT_1, RAPI_AD_UNIT_2 }から 1 つを指定します。
省略値は RAPI_AD_UNIT_1 です。

（入力端子）　{ RAPI_AN0_GROUP, RAPI_AN4_GROUP, RAPI_AN8_GROUP }から 1 つを指定します。省略値は RAPI_AN0_GROUP です。

（変換時間）　{ RAPI_84_STATES, RAPI_43_STATES }から 1 つを指定します。

（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から 1 つを指定します。
省略値は RAPI_AD_OFF です。

（トリガ）　{ RAPI_ADTRG1_TRIGGER, RAPI_ADTRG2_TRIGGER, RAPI_TIMER_RC_TRIGGER, RAPI_TIMER_RD_0_TRIGGER, RAPI_TIMER_RD_1_TRIGGER }
から 1 つを指定します。トリガを指定しない場合は、「トリガ入力禁止」に設定します。

（割り込み要求）　{RAPI_AD_INT_REQUEST, RAPI_NOT_AD_INT_REQUEST}から 1 つを指定します。
省略値は RAPI_NOT_AD_INT_REQUEST です。

・繰り返し掃引モード 0 使用時（RAPI_REPEAT_SWEEP0 を指定時）に指定可能な定義値

（ユニット）　{ RAPI_AD_UNIT_1, RAPI_AD_UNIT_2 }から 1 つを指定します。
省略値は RAPI_AD_UNIT_1 です。

（入力端子）　{ RAPI_AN0_GROUP, RAPI_AN4_GROUP , RAPI_AN8_GROUP }
から 1 つを指定します。省略値は RAPI_AN0_GROUP です。

（変換時間）　{ RAPI_84_STATES, RAPI_43_STATES }から 1 つを指定します。

（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から 1 つを指定します。
省略値は RAPI_AD_OFF です。

（トリガ）　{ RAPI_ADTRG1_TRIGGER, RAPI_ADTRG2_TRIGGER, RAPI_TIMER_RC_TRIGGER, RAPI_TIMER_RD_0_TRIGGER, RAPI_TIMER_RD_1_TRIGGER }
から 1 つを指定します。トリガを指定しない場合は、「トリガ入力禁止」に設定します。

（割り込み要求）　{RAPI_AD_INT_REQUEST, RAPI_NOT_AD_INT_REQUEST}から 1 つを指定します。
省略値は RAPI_NOT_AD_INT_REQUEST です。

(H8/300H)

RAPI_ONE_SHOT	単発モードを選択する。
RAPI_REPEAT	繰り返しモードを選択する。
RAPI_SINGLE_SWEEP	単掃引モードを選択する。
RAPI_REPEAT_SWEEP0	繰り返し掃引モード 0 を選択する。
RAPI_AN0	アナログ入力端子に AN0 端子を使用する。
RAPI_AN1	アナログ入力端子に AN1 端子を使用する。
RAPI_AN2	アナログ入力端子に AN2 端子を使用する。
RAPI_AN3	アナログ入力端子に AN3 端子を使用する。
RAPI_AN4	アナログ入力端子に AN4 端子を使用する。
RAPI_AN5	アナログ入力端子に AN5 端子を使用する。

RAPI_AN6	アナログ入力端子に AN6 端子を使用する。
RAPI_AN7	アナログ入力端子に AN7 端子を使用する。
RAPI_AN8	アナログ入力端子に AN8 端子を使用する。
RAPI_AN9	アナログ入力端子に AN9 端子を使用する。
RAPI_AN10	アナログ入力端子に AN10 端子を使用する。
RAPI_AN11	アナログ入力端子に AN11 端子を使用する。
RAPI_AN12	アナログ入力端子に AN12 端子を使用する。
RAPI_AN13	アナログ入力端子に AN13 端子を使用する。
RAPI_AN14	アナログ入力端子に AN14 端子を使用する。
RAPI_AN15	アナログ入力端子に AN15 端子を使用する。
RAPI_AN0_GROUP	アナログ入力端子に AN0～AN3 のグループを使用する。
RAPI_AN4_GROUP	アナログ入力端子に AN4～AN7 のグループを使用する。
RAPI_AN8_GROUP	アナログ入力端子に AN8～AN11 のグループを使用する。
RAPI_AN12_GROUP	アナログ入力端子に AN12～AN15 のグループを使用する。
RAPI_134_STATES	変換時間 = 134 ステート
RAPI_70_STATES	変換時間 = 70 ステート
RAPI_AD_ON	__CreateADC 内で A/D 変換器を動作開始に設定します。
RAPI_AD_OFF	__CreateADC 内では A/D 変換器を動作停止に設定します。
RAPI_RISING_TRIGGER	A/D 変換開始条件に立ち上がりエッジの外部トリガを含める。
RAPI_FALLING_ TRIGGER	A/D 変換開始条件に立ち下がりエッジの外部トリガを含める。
RAPI_AD_INT_REQUEST	A/D 変換割り込みの割り込み要求をする。
RAPI_NOT_AD_INT_ REQUEST	A/D 変換割り込みの割り込み要求をしない。

・単発モード使用時（RAPI_ONE_SHOT を指定時）に指定可能な定義値

（入力端子）　{ RAPI_AN0, RAPI_AN1, RAPI_AN2, RAPI_AN3, RAPI_AN4, RAPI_AN5, RAPI_AN6, RAPI_AN7, RAPI_AN8, RAPI_AN9, RAPI_AN10, RAPI_AN11 RAPI_AN12, RAPI_AN13, RAPI_AN14, RAPI_AN15}から 1 つを指定します。
省略値は RAPI_AN0 です。

（変換時間）　{ RAPI_134_STATES, RAPI_70_STATES }から 1 つを指定します。
省略値は RAPI_134_STATES です。

（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から 1 つを指定します。
省略値は RAPI_AD_OFF です。

（トリガ）　{ RAPI_RISING_TRIGGER, RAPI_FALLING_TRIGGER }から 1 つを指定します。
トリガを指定しない場合は、「トリガ入力禁止」に設定します。

（割り込み要求）　{RAPI_AD_INT_REQUEST, RAPI_NOT_AD_INT_REQUEST}から 1 つを指定します。
省略値は RAPI_NOT_AD_INT_REQUEST です。

・繰り返しモード使用時（RAPI_REPEAT を指定時）に指定可能な定義値

（入力端子）　{ RAPI_AN0, RAPI_AN1, RAPI_AN2, RAPI_AN3, RAPI_AN4, RAPI_AN5, RAPI_AN6, RAPI_AN7, RAPI_AN8, RAPI_AN9, RAPI_AN10, RAPI_AN11 RAPI_AN12, RAPI_AN13, RAPI_AN14, RAPI_AN15}から 1 つを指定します。
省略値は RAPI_AN0 です。

（変換時間）　{ RAPI_134_STATES, RAPI_70_STATES }から 1 つを指定します。
省略値は RAPI_134_STATES です。

（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から１つを指定します。
省略値は RAPI_AD_OFF です。
（トリガ）　{ RAPI_RISING_TRIGGER, RAPI_FALLING_TRIGGER }から１つを指定します。
トリガを指定しない場合は、「トリガ入力禁止」に設定します。

・単掃引モード使用時（RAPI_SINGLE_SWEEP を指定時）に指定可能な定義値

（入力端子）　{ RAPI_AN0_GROUP, RAPI_AN4_GROUP , RAPI_AN8_GROUP, RAPI_AN12_GROUP}
から１つを指定します。省略値は RAPI_AN0_GROUP です。
（変換時間）　{ RAPI_134_STATES, RAPI_70_STATES }から１つを指定します。
省略値は RAPI_134_STATES です。
（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から１つを指定します。
省略値は RAPI_AD_OFF です。
（トリガ）　{ RAPI_RISING_TRIGGER, RAPI_FALLING_TRIGGER }から１つを指定します。
トリガを指定しない場合は、「トリガ入力禁止」に設定します。
（割り込み要求）　{RAPI_AD_INT_REQUEST, RAPI_NOT_AD_INT_REQUEST}から１つを指定します。
省略値は RAPI_NOT_AD_INT_REQUEST です。

・繰り返し掃引モード０使用時（RAPI_REPEAT_SWEEP0 を指定時）に指定可能な定義値

（入力端子）　{ RAPI_AN0_GROUP, RAPI_AN4_GROUP , RAPI_AN8_GROUP, RAPI_AN12_GROUP}
から１つを指定します。省略値は RAPI_AN0_GROUP です。
（変換時間）　{ RAPI_134_STATES, RAPI_70_STATES }から１つを指定します。
省略値は RAPI_134_STATES です。
（設定時動作状態）　{ RAPI_AD_ON, RAPI_AD_OFF }から１つを指定します。
省略値は RAPI_AD_OFF です。
（トリガ）　{ RAPI_RISING_TRIGGER, RAPI_FALLING_TRIGGER }から１つを指定します。
トリガを指定しない場合は、「トリガ入力禁止」に設定します。

[data2]

(M16C)

使用する A/D 変換モードにより設定値が異なります。

単発モード 繰り返しモード	1 を指定します。
単掃引モード 繰り返し掃引モード 0 同時サンプル掃引モード 遅延トリガモード 0 遅延トリガモード 1	2,4,6,8 を指定します。
繰り返し掃引モード 1	1,2,3,4 を指定します。

(R8C)

単発モード 繰り返しモード	1 を指定します。
単掃引モード 繰り返し掃引モード 0	2,4 を指定します。

(H8S)

単発モード	1 を指定します。

繰り返しモード	
単掃引モード 繰り返し掃引モード 0	1,2,3,4,5,6,7,8 を指定します。

(H8/300H)

単発モード 繰り返しモード	1 を指定します。
単掃引モード 繰り返し掃引モード 0	1,2,3,4 を指定します。

[data3]

(M16C)(R8C)

割り込み制御レジスタへ設定する割り込みレベル(0～7)を指定します。

(H8S)

インタラプトプライオリティレジスタへ設定する割り込み優先レベル(0～3)を指定します。

(H8/300H)

割り込みコントロールレジスタへ設定する割り込みプライオリティレベル(0～1)を指定します。割り込みコントロールレジスタを持たない CPU の場合は 0 を指定します。

戻り値

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。

機能

A/D 変換器

参照

__EnableADC, __DestroyADC, __DestroyADCUnit, __GetADC, __GetADCAll, __GetADCUnit

備考

- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できる指定できるアナログ入力端子は CPU により異なります。
- R8C で動作周波数に fOCO-F を指定できない CPU の場合は、動作周波数への RAPI_FOCOF の指定はできません。
- R8C でトリガにタイマ RD を指定できない CPU の場合は、トリガへの RAPI_TIMER_RD_TRIGGER の指定はできません。
- H8S および、H8/300H で使用する場合、A/D 変換器のモジュールスタンバイ解除後に設定をおこないます。

プログラム例

```
#include " rapi_ad_r8c_13.h"

void AdIntFunc( void ){}

void func( void )
{
    /* Set up A/D converter as one short mode */
    __CreateADC( RAPI_ONE_SHOT|RAPI_AN2|RAPI_FAD2| RAPI_WITH_SAMPLE_HOLD |
                 RAPI_EXTERNAL_TRIGGER |RAPI_AD_ON|RAPI_10BIT, 1, 5, AdIntFunc );
}
```

__EnableADC

要約

< A/D 変換器の動作制御 >

Boolean __EnableADC (unsigned long data1, unsigned int data2)

data1	設定データ 1（機種により内容が異なる）
data2	A/D 変換で使用するアナログ入力端子数（機種により内容が異なる）

解説

A/D 変換器の動作開始/動作停止を制御します。

[data1]

data1 には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。ただし、複数のアナログ入力端子の定義値を同時に指定することはできません。

(M16C)

RAPI_AN0	アナログ入力端子に AN_0 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN1	アナログ入力端子に AN_1 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN2	アナログ入力端子に AN_2 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN3	アナログ入力端子に AN_3 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN4	アナログ入力端子に AN_4 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN5	アナログ入力端子に AN_5 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN6	アナログ入力端子に AN_6 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN7	アナログ入力端子に AN_7 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN00	アナログ入力端子に AN_{00} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN01	アナログ入力端子に AN_{01} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN02	アナログ入力端子に AN_{02} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN03	アナログ入力端子に AN_{03} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN04	アナログ入力端子に AN_{04} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN05	アナログ入力端子に AN_{05} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN06	アナログ入力端子に AN_{06} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。

R8PI_AN07	アナログ入力端子に AN_{07} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
R8PI_AN20	アナログ入力端子に AN_{20} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
R8PI_AN21	アナログ入力端子に AN_{21} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
R8PI_AN22	アナログ入力端子に AN_{22} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
R8PI_AN23	アナログ入力端子に AN_{23} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
R8PI_AN24	アナログ入力端子に AN_{24} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
R8PI_AN25	アナログ入力端子に AN_{25} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
R8PI_AN26	アナログ入力端子に AN_{26} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
R8PI_AN27	アナログ入力端子に AN_{27} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
R8PI_AN30	アナログ入力端子に AN_{30} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
R8PI_AN31	アナログ入力端子に AN_{31} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
R8PI_AN32	アナログ入力端子に AN_{32} 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
R8PI_P0_GROUP	アナログ入力端子にポート P_0 グループを使用する。単掃引モード、 繰り返し掃引モード 0、同時サンプル掃引モード、遅延トリガモード 0、 遅延トリガモード 1 および、繰り返し掃引モード 1 使用時のみ選択可能。
R8PI_P10_GROUP	アナログ入力端子にポート P_{10} グループを使用する。単掃引モード、 繰り返し掃引モード 0、同時サンプル掃引モード、遅延トリガモード 0、 遅延トリガモード 1 および、繰り返し掃引モード 1 使用時のみ選択可能。
R8PI_P1P9_GROUP	アナログ入力端子にポート P_1 /P_9 グループを使用する。単掃引モード、 繰り返し掃引モード 0、同時サンプル掃引モード、遅延トリガモード 0、 遅延トリガモード 1 および、繰り返し掃引モード 1 使用時のみ選択可能。
R8PI_P9_GROUP	アナログ入力端子にポート P_9 グループを使用する。単掃引モード、 繰り返し掃引モード 0、同時サンプル掃引モード、遅延トリガモード 0、 遅延トリガモード 1 および、繰り返し掃引モード 1 使用時のみ選択可能。
R8PI_AD_ON	A/D 変換器を動作開始に設定します。
R8PI_AD_OFF	A/D 変換器を動作停止に設定します。

(R8C)

R8PI_AN0	アナログ入力端子に AN_0 端子を使用する。
R8PI_AN1	アナログ入力端子に AN_1 端子を使用する。
R8PI_AN2	アナログ入力端子に AN_2 端子を使用する。
R8PI_AN3	アナログ入力端子に AN_3 端子を使用する。
R8PI_AN4	アナログ入力端子に AN_4 端子を使用する。

RAPI_AN5	アナログ入力端子に AN_5 端子を使用する。
RAPI_AN6	アナログ入力端子に AN_6 端子を使用する。
RAPI_AN7	アナログ入力端子に AN_7 端子を使用する。
RAPI_AN8	アナログ入力端子に AN_8 端子を使用する。
RAPI_AN9	アナログ入力端子に AN_9 端子を使用する。
RAPI_AN10	アナログ入力端子に AN_{10} 端子を使用する。
RAPI_AN11	アナログ入力端子に AN_{11} 端子を使用する。
RAPI_AN12	アナログ入力端子に AN_{12} 端子を使用する。
RAPI_AN13	アナログ入力端子に AN_{13} 端子を使用する。
RAPI_AN14	アナログ入力端子に AN_{14} 端子を使用する。
RAPI_AN15	アナログ入力端子に AN_{15} 端子を使用する。
RAPI_AN16	アナログ入力端子に AN_{16} 端子を使用する。
RAPI_AN17	アナログ入力端子に AN_{17} 端子を使用する。
RAPI_AN18	アナログ入力端子に AN_{18} 端子を使用する。
RAPI_AN19	アナログ入力端子に AN_{19} 端子を使用する。
RAPI_AN12_GROUP	アナログ入力端子に AN12 グループを使用する。
RAPI_AN16_GROUP	アナログ入力端子に AN16 グループを使用する。
RAPI_AD_ON	A/D 変換器を動作開始に設定します。
RAPI_AD_OFF	A/D 変換器を動作停止に設定します。

(H8S)

RAPI_AD_UNIT_1	A/D 変換器ユニット 1 を選択する。
RAPI_AD_UNIT_2	A/D 変換器ユニット 2 を選択する。
RAPI_AN0	アナログ入力端子に AN0 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN1	アナログ入力端子に AN1 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN2	アナログ入力端子に AN2 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN3	アナログ入力端子に AN3 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN4	アナログ入力端子に AN4 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN5	アナログ入力端子に AN5 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN6	アナログ入力端子に AN6 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN7	アナログ入力端子に AN7 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN8	アナログ入力端子に AN0 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN9	アナログ入力端子に AN9 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN10	アナログ入力端子に AN10 端子を使用する。

	単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN11	アナログ入力端子に AN11 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN0_GROUP	アナログ入力端子に AN0～AN3 のグループを使用する。 単掃引モードおよび、繰り返し掃引モード 0 使用時のみ選択可能。
RAPI_AN4_GROUP	アナログ入力端子に AN4～AN7 のグループを使用する。 単掃引モードおよび、繰り返し掃引モード 0 使用時のみ選択可能。
RAPI_AN8_GROUP	アナログ入力端子に AN8～AN11 のグループを使用する。 単掃引モードおよび、繰り返し掃引モード 0 使用時のみ選択可能。
RAPI_AD_ON	A/D 変換器を動作開始に設定します。
RAPI_AD_OFF	A/D 変換器を動作停止に設定します。

(H8/300H)

RAPI_AN0	アナログ入力端子に AN0 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN1	アナログ入力端子に AN1 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN2	アナログ入力端子に AN2 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN3	アナログ入力端子に AN3 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN4	アナログ入力端子に AN4 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN5	アナログ入力端子に AN5 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN6	アナログ入力端子に AN6 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN7	アナログ入力端子に AN7 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN8	アナログ入力端子に AN0 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN9	アナログ入力端子に AN9 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN10	アナログ入力端子に AN10 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN11	アナログ入力端子に AN11 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN12	アナログ入力端子に AN12 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN13	アナログ入力端子に AN13 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN14	アナログ入力端子に AN14 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。
RAPI_AN15	アナログ入力端子に AN15 端子を使用する。 単発モードおよび繰り返しモード使用時のみ選択可能。

RAPI_AN0_GROUP	アナログ入力端子に AN0～AN3 のグループを使用する。 単掃引モードおよび、繰り返し掃引モード 0 使用時のみ選択可能。
RAPI_AN4_GROUP	アナログ入力端子に AN4～AN7 のグループを使用する。 単掃引モードおよび、繰り返し掃引モード 0 使用時のみ選択可能。
RAPI_AN8_GROUP	アナログ入力端子に AN8～AN11 のグループを使用する。 単掃引モードおよび、繰り返し掃引モード 0 使用時のみ選択可能。
RAPI_AN12_GROUP	アナログ入力端子に AN12～AN15 のグループを使用する。 単掃引モードおよび、繰り返し掃引モード 0 使用時のみ選択可能。
RAPI_AD_ON	A/D 変換器を動作開始に設定します。
RAPI_AD_OFF	A/D 変換器を動作停止に設定します。

[data2]

(M16C)

使用する A/D 変換モードにより設定値が異なります。

単発モード 繰り返しモード	1 を指定します。
単掃引モード 繰り返し掃引モード 0 同時サンプル掃引モード 遅延トリガモード 0 遅延トリガモード 1	2,4,6,8 を指定します。
繰り返し掃引モード 1	1,2,3,4 を指定します。

(R8C)

使用する A/D 変換モードにより設定値が異なります。

単発モード 繰り返しモード	1 を指定します。
単掃引モード 繰り返し掃引モード 0	2,4 を指定します。

(H8S)

使用する A/D 変換モードにより設定値が異なります。

単発モード 繰り返しモード	1 を指定します。
単掃引モード 繰り返し掃引モード 0	1,2,3,4,5,6,7,8 を指定します。

(H8/300H)

使用する A/D 変換モードにより設定値が異なります。

単発モード 繰り返しモード	1 を指定します。
単掃引モード 繰り返し掃引モード 0	1,2,3,4 を指定します。

戻り値　制御に成功した場合は RAPI_TRUE を返し、制御に失敗した場合は RAPI_FALSE を返します。

機能　A/D 変換器

参照　__CreateADC, __DestroyADC, __DestroyADCUnit, __GetADC, __GetADCAll, __GetADCUnit

備考
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- 指定できる指定できるアナログ入力端子は CPU により異なります。

プログラム例

```
#include "rapi_ad_r8c_13.h"

void func( void )
{
    /* Disable A/D converter */
    __EnableADC( RAPI_AN0|RAPI_AD_OFF, 1 );
}
```

__DestroyADC

要約

< A/D 変換器の設定破棄 >

Boolean __DestroyADC(void)

解説

指定した A/D 変換器の設定を破棄します。

戻り値

設定破棄に成功した場合は RAPI_TRUE を返し、設定破棄に失敗した場合は RAPI_FALSE を返します。

機能

A/D 変換器

参照

__CreateADC, __EnableADC, __DestroyADCUnit, __GetADC, __GetADCAll, __GetADCUnit

備考

・ H8S および、H8/300H で使用する場合、A/D 変換器を設定破棄後、
モジュールスタンバイ状態に設定します。
・ 複数の A/D 変換器ユニットがある場合は、ユニット番号が最も若い
A/D 変換器ユニットの設定を破棄します。

プログラム例

```
#include "rapi_ad_r8c_13.h"

void func( void )
{
    /* Destroy A/D converter */
    __DestroyADC();
}
```

__DestroyADCUnit

要約

< A/D 変換器の設定破棄（ユニット指定）>

Boolean __DestroyADCUnit(unsigned long unit)

unit	設定を破棄する A/D 変換器のユニット

解説

指定した A/D 変換器ユニットの設定を破棄します。

[unit]

unit には、次の定義値を設定することができます。

複数の A/D 変換器ユニットを有する CPU で 0 を指定した場合は、ユニット番号が最も若い A/D 変換器ユニットの設定を破棄します。

(M16C)(R8C)(H8S)(H8/300H)

0 を指定します。

(H8S)

RAPI_AD_UNIT_1	A/D 変換器ユニット 1 を選択する。
RAPI_AD_UNIT_2	A/D 変換器ユニット 2 を選択する。

戻り値

設定破棄に成功した場合は RAPI_TRUE を返し、設定破棄に失敗した場合は RAPI_FALSE を返します。

機能

A/D 変換器

参照

__CreateADC, __EnableADC, __DestroyADC, __GetADC, __GetADCAll, __GetADCUnit

備考

・H8S および、H8/300H で使用する場合、A/D 変換器を設定破棄後、
モジュールスタンバイ状態に設定します。

プログラム例

```
#include "rapi_ad_h8s_20223.h"

void func( void )
{
    /* Destroy A/D converter */
    __DestroyADCUnit( RAPI_AD_UNIT_1 );
}
```

__GetADC

要約

< A/D 変換値の取得（レジスタ指定）>

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

data1	設定データ 1（機種により内容が異なる）
data2	A/D 変換値を格納するポインタ

解説

指定した A/D レジスタの A/D 変換値を取得します。

data1 には、次の値を設定することができます。

(M16C)

RAPI_AD0	A/D レジスタ 0 を選択する。
RAPI_AD1	A/D レジスタ 1 を選択する。
RAPI_AD2	A/D レジスタ 2 を選択する。
RAPI_AD3	A/D レジスタ 3 を選択する。
RAPI_AD4	A/D レジスタ 4 を選択する。
RAPI_AD5	A/D レジスタ 5 を選択する。
RAPI_AD6	A/D レジスタ 6 を選択する。
RAPI_AD7	A/D レジスタ 7 を選択する。

(R8C)

複数の A/D データレジスタを有する R8C では次の定義値を指定します。

RAPI_ADDR0	A/D データレジスタ 0 を選択する。
RAPI_ADDR1	A/D データレジスタ 1 を選択する。
RAPI_ADDR2	A/D データレジスタ 2 を選択する。
RAPI_ADDR3	A/D データレジスタ 3 を選択する。

それ以外は 0 を指定します。

(H8S)

複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

ただし、複数のアナログ入力端子の定義値を同時に指定することはできません。

RAPI_AD_UNIT_1	A/D 変換器ユニット 1 を選択する。
RAPI_AD_UNIT_2	A/D 変換器ユニット 2 を選択する。
RAPI_ADDR0	A/D データレジスタ 0 を選択する。
RAPI_ADDR1	A/D データレジスタ 1 を選択する。
RAPI_ADDR2	A/D データレジスタ 2 を選択する。
RAPI_ADDR3	A/D データレジスタ 3 を選択する。
RAPI_ADDR4	A/D データレジスタ 4 を選択する。
RAPI_ADDR5	A/D データレジスタ 5 を選択する。
RAPI_ADDR6	A/D データレジスタ 6 を選択する。
RAPI_ADDR7	A/D データレジスタ 7 を選択する。

(H8/300H)

RAPI_ADDRA	A/D データレジスタ A を選択する。
RAPI_ADDRB	A/D データレジスタ B を選択する。
RAPI_ADDRC	A/D データレジスタ C を選択する。

RAPI_ADDRD	A/D データレジスタ D を選択する。

戻り値

A/D 変換値取得に成功した場合は RAPI_TRUE を返し、A/D 変換値取得に失敗した場合は RAPI_FALSE を返します。

機能

A/D 変換器

参照

__CreateADC, __EnableADC, __DestroyADC, __DestroyADCUnit, __GetADCAll, __GetADCUnit

備考

・第 1 引数に定義値以外の値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_r8c_13.h"

void func( void )
{
    unsigned int data;

    /* Get an A/D convertered data of A/D register 0 */
    __GetADC( 0, &data );
}
```

__GetADCAll

要約

< A/D 変換値の取得（全レジスタ）>

Boolean __GetADCAll(unsigned int *data)

data	A/D 変換値を格納するポインタ

解説

すべての A/D レジスタの A/D 変換値を取得します。

取得する A/D レジスタを以下に示します。

[M16C] (16 バイト)	:	[0] A/D レジスタ 0 [1] A/D レジスタ 1 [2] A/D レジスタ 2 [3] A/D レジスタ 3 [4] A/D レジスタ 4 [5] A/D レジスタ 5 [6] A/D レジスタ 6 [7] A/D レジスタ 7
[R8C] <複数の A/D データ レジスタを有する 場合> (8 バイト)	:	[0] A/D データレジスタ 0 [1] A/D データレジスタ 1 [2] A/D データレジスタ 2 [3] A/D データレジスタ 3
<上記以外> (2 バイト)	:	A/D レジスタ
[H8S] <ユニット 1> (16 バイト)	:	[0] A/D データレジスタ 0 [1] A/D データレジスタ 1 [2] A/D データレジスタ 2 [3] A/D データレジスタ 3 [4] A/D データレジスタ 4 [5] A/D データレジスタ 5 [6] A/D データレジスタ 6 [7] A/D データレジスタ 7
[H8/300H] (8 バイト)	:	[0] A/D データレジスタ A [1] A/D データレジスタ B [2] A/D データレジスタ C [3] A/D データレジスタ D

戻り値

A/D 変換値取得に成功した場合は RAPI_TRUE を返し、A/D 変換値取得に失敗した場合は RAPI_FALSE を返します。

機能

A/D 変換器

参照

__CreateADC, __EnableADC, __DestroyADC, __DestroyADCUnit, __GetADC, __GetADCUnit

備考

・複数の A/D 変換器ユニットがある場合は、ユニット番号が最も若い A/D 変換器ユニットの A/D 変換値を取得します。

プログラム例

```
#include "rapi_ad_r8c_13.h"

void func( void )
{
    unsigned int data;

    /* Get A/D convertered datas of A/D register */
    __GetADCAll( &data );
}
```

__GetADCUnit

要約

< A/D 変換値の取得（ユニット指定）>

Boolean __GetADCUnit(unsigned long unit, unsigned int *data)

unit	A/D 変換値を取得する A/D 変換器のユニット
data	A/D 変換値を格納するポインタ

解説

指定した A/D 変換器ユニットのすべての A/D レジスタの A/D 変換値を取得します。

[unit]

unit には、次の定義値を設定することができます。

複数の A/D 変換器ユニットを有する CPU で 0 を指定した場合は、ユニット番号が最も若い A/D 変換器ユニットのすべての A/D レジスタの A/D 変換値を取得します。

(M16C)(R8C)(H8S)(H8/300H)

0 を指定します。

(H8S)

RAPI_AD_UNIT_1	A/D 変換器ユニット 1 を選択する。
RAPI_AD_UNIT_2	A/D 変換器ユニット 2 を選択する。

[data]

取得する A/D レジスタを以下に示します。

[M16C] (16 バイト)	:	[0] A/D レジスタ 0 [1] A/D レジスタ 1 [2] A/D レジスタ 2 [3] A/D レジスタ 3 [4] A/D レジスタ 4 [5] A/D レジスタ 5 [6] A/D レジスタ 6 [7] A/D レジスタ 7
[R8C] <複数の A/D データレジスタを有する場合> (8 バイト)	:	[0] A/D データレジスタ 0 [1] A/D データレジスタ 1 [2] A/D データレジスタ 2 [3] A/D データレジスタ 3
<上記以外> (2 バイト)	:	A/D レジスタ
[H8S] <ユニット 1> (16 バイト)	:	[0] A/D データレジスタ 0 [1] A/D データレジスタ 1 [2] A/D データレジスタ 2 [3] A/D データレジスタ 3 [4] A/D データレジスタ 4 [5] A/D データレジスタ 5 [6] A/D データレジスタ 6 [7] A/D データレジスタ 7

<ユニット 2> (8 バイト)	:	[0] A/D データレジスタ 0 [1] A/D データレジスタ 1 [2] A/D データレジスタ 2 [3] A/D データレジスタ 3
[H8/300H] (8 バイト)	:	[0] A/D データレジスタ A [1] A/D データレジスタ B [2] A/D データレジスタ C [3] A/D データレジスタ D

戻り値

A/D 変換値取得に成功した場合は RAPI_TRUE を返し、A/D 変換値取得に失敗した場合は RAPI_FALSE を返します。

機能

A/D 変換器

参照

__CreateADC, __EnableADC, __DestroyADC, __DestroyADCUnit, __GetADC, __GetADCAll

プログラム例

```
#include "rapi_ad_h8s_20223.h"

void func( void )
{
    unsigned int data;

    /* Get A/D convertered datas of A/D register */
    __GetADCUnit( RAPI_AD_UNIT_1, &data );
}
```

__GetADCStatus

要約

< A/D 変換器の状態の取得 >

Boolean __GetADCStatus(unsigned int *status)

status	A/D 変換器の状態を示すレジスタの内容を格納するポインタ

解説

A/D 変換器の状態を取得します。
*status の下位 1 ビット目に割り込みビットの状態(M16C または R8C 使用時)
または A/D エンドフラグの値(H8S および、H8/300H 使用時)を格納します。
また*status の下位 2 ビット目に A/D 変換開始フラグの状態を格納します。
M16C で使用する場合には、*status の上位 8 ビットに A/D 変換ステータス
レジスタ 0 の値を格納します。

[*status の構成]

戻り値

A/D 変換器の状態取得に成功した場合は RAPI_TRUE を返し、A/D 変換器の状態取得取得に
失敗した場合は RAPI_FALSE を返します。

機能

A/D 変換器

参照

__GetADCUnitStatus, __ClearADCStatus, __ClearADCUnitStatus

備考

・ 複数の A/D 変換器ユニットがある場合は、ユニット番号が最も若い
A/D 変換器ユニットの状態を取得します。

プログラム例

```
#include " rapi_ad_r8c_13.h"

void func( void )
{
    unsigned int status;

    /* Get status of A/D convertered */
    __GetADCStatus( &status );
}
```

__GetADCUnitStatus

要約

< A/D 変換器の状態の取得（ユニット指定）>

Boolean __GetADCUnitStatus(unsigned long unit, unsigned int *status)

unit	状態を取得する A/D 変換器のユニット
status	A/D 変換器の状態を示すレジスタの内容を格納するポインタ

解説

指定した A/D 変換器ユニットの状態を取得します。

[unit]

unit には、次の定義値を設定することができます。

複数の A/D 変換器ユニットを有する CPU で 0 を指定した場合は、ユニット番号が最も若い A/D 変換器ユニットの状態を取得します。

(M16C)(R8C)(H8S)(H8/300H)

0 を指定します。

(H8S)

RAPI_AD_UNIT_1	A/D 変換器ユニット 1 を選択する。
RAPI_AD_UNIT_2	A/D 変換器ユニット 2 を選択する。

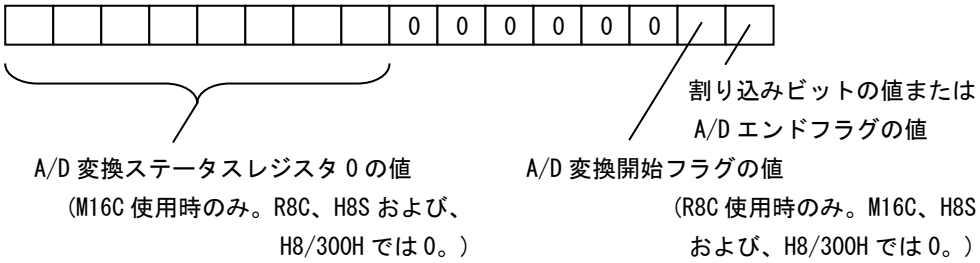
[status]

*status の下位 1 ビット目に割り込みビットの状態(M16C または R8C 使用時)または A/D エンドフラグの値(H8S および、H8/300H 使用時)を格納します。

また*status の下位 2 ビット目に A/D 変換開始フラグの状態を格納します。

M16C で使用する場合には、*status の上位 8 ビットに A/D 変換ステータスレジスタ 0 の値を格納します。

[*status の構成]



戻り値

A/D 変換器の状態取得に成功した場合は RAPI_TRUE を返し、A/D 変換器の状態取得取得に失敗した場合は RAPI_FALSE を返します。

機能

A/D 変換器

参照

__GetADCStatus, __ClearADCStatus, __ClearADCUnitStatus

プログラム例

```
#include " rapi_ad_h8s_20223.h"
```

```
void func( void )
{
    unsigned int status;

    /* Get status of A/D convertered */
    __GetADCUnitStatus( RAPI_AD_UNIT_1, &status );
}
```

__ClearADCStatus

要約

< A/D 変換器の状態のクリア >

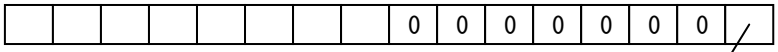
Boolean __ClearADCStatus(unsigned int status)

status	A/D 変換器の状態

解説

A/D 変換器の状態フラグをクリアします。
status の下位 1 ビット目に割り込みビットの状態(M16C または R8C 使用時)
または A/D エンドフラグの値(H8S または H8/300H 使用時)を指定します。
M16C で使用する場合には、status の上位 8 ビットに A/D 変換ステータスレジスタ 0 の値を指定します。
クリアするビットには 0 を設定し、クリアしないビットには 1 を設定します。

[status の構成]



A/D 変換ステータスレジスタ 0 の値
(M16C 使用時のみ。R8C、H8S および、
H8/300H では 0。)

割り込みビットの値
または A/D エンドフラグの値

戻り値

A/D 変換器の状態フラグのクリアに成功した場合は RAPI_TRUE を返し、A/D 変換器の状態フラグのクリア取得に失敗した場合は RAPI_FALSE を返します。

機能

A/D 変換器

参照

__GetADCStatus, __GetADCUnitStatus, __ClearADCUnitStatus

備考

・ 複数の A/D 変換器ユニットがある場合は、ユニット番号が最も若い A/D 変換器ユニットの状態フラグをクリアします。

プログラム例

```
#include "rapi_ad_r8c_13.h"

void func( void )
{
    /* Clear status of A/D convertered */
    __ClearADCStatus( 0 );
}
```

__ClearADCUnitStatus

要約

< A/D 変換器の状態のクリア（ユニット指定）>

Boolean __ClearADCUnitStatus(unsigned long unit, unsigned int status)

unit	状態フラグをクリアする A/D 変換器のユニット
status	A/D 変換器の状態

解説

指定した A/D 変換器ユニットの状態フラグをクリアします。

[unit]

unit には、次の定義値を設定することができます。

複数の A/D 変換器ユニットを有する CPU で 0 を指定した場合は、ユニット番号が最も若い A/D 変換器ユニットの状態をクリアします。

(M16C) (R8C) (H8S) (H8/300H)

0 を指定します。

(H8S)

RAPI_AD_UNIT_1	A/D 変換器ユニット 1 を選択する。
RAPI_AD_UNIT_2	A/D 変換器ユニット 2 を選択する。

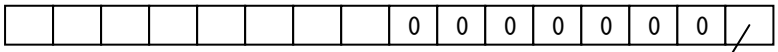
[status]

status の下位 1 ビット目に割り込みビットの状態(M16C または R8C 使用時)または A/D エンドフラグの値(H8S または H8/300H 使用時)を指定します。

M16C で使用する場合には、status の上位 8 ビットに A/D 変換ステータスレジスタ 0 の値を指定します。

クリアするビットには 0 を設定し、クリアしないビットには 1 を設定します。

[status の構成]



A/D 変換ステータスレジスタ 0 の値
(M16C 使用時のみ。R8C、H8S および、H8/300H では 0。)

割り込みビットの値
または A/D エンドフラグの値

戻り値

A/D 変換器の状態フラグのクリアに成功した場合は RAPI_TRUE を返し、A/D 変換器の状態フラグのクリア取得に失敗した場合は RAPI_FALSE を返します。

機能

A/D 変換器

参照

__GetADCStatus, __GetADCUnitStatus, __ClearADCStatus

備考

・複数の A/D 変換器ユニットがある場合は、ユニット番号が最も若い A/D 変換器ユニットの状態フラグをクリアします。

プログラム例

```
#include "rapi_ad_h8s_20223.h"

void func( void )
{
    /* Clear status of A/D convertered */
    __ClearADCUnitStatus( RAPI_AD_UNIT_1, 0 );
}
```

4.2.6 周辺機能マッピングコントローラ __ConfigurePMC

要約

〈 各端子への機能の割り当ての設定 〉

Boolean __ConfigurePMC(unsigned long port, unsigned int *function)

port	機能を設定する端子の I/O ポート
function	機能を示す定義値を格納する配列のポインタ

解説

指定した端子に周辺機能を設定します。

[port]

機能を設定する端子の I/O ポートを指定します。各 I/O ポートに対応する定義値は次の通りです。

(H8S)

RAPI_GR1_PORT_1	グループ 1・ポート 1 を選択する。
RAPI_GR1_PORT_2	グループ 1・ポート 2 を選択する。
RAPI_GR1_PORT_3	グループ 1・ポート 3 を選択する。
RAPI_GR1_PORT_5	グループ 1・ポート 5 を選択する。
RAPI_GR1_PORT_6	グループ 1・ポート 6 を選択する。
RAPI_GR2_PORT_8	グループ 2・ポート 8 を選択する。
RAPI_GR2_PORT_9	グループ 2・ポート 9 を選択する。
RAPI_GR2_PORT_A	グループ 2・ポート A を選択する。

[function]

機能を示す定義値を格納する配列のポインタを指定します。本配列の要素数は、第 1 引数 port で指定したポートレジスタのビット数と一致し、本配列の要素番号は、第 1 引数 port で指定したポートレジスタのポート番号と一致します。

本配列で設定する機能を示す定義値は次のとおりです。I/O ポートとして使用する場合は 0 を設定してください。

(H8S)

RAPI_GR1_IRQ0	グループ 1・_IRQ0 入力を選択する。
RAPI_GR1_IRQ1	グループ 1・_IRQ1 入力を選択する。
RAPI_GR1_IRQ2	グループ 1・_IRQ2 入力を選択する。
RAPI_GR1_IRQ3	グループ 1・_IRQ3 入力を選択する。
RAPI_GR1_IRQ4	グループ 1・_IRQ4 入力を選択する。
RAPI_GR1_IRQ5	グループ 1・_IRQ5 入力を選択する。
RAPI_GR1_IRQ6	グループ 1・_IRQ6 入力を選択する。
RAPI_GR1_IRQ7	グループ 1・_IRQ7 入力を選択する。
RAPI_GR1_ADTRG1	グループ 1・_ADTRG1 入力を選択する。
RAPI_GR1_ADTRG2	グループ 1・_ADTRG2 入力を選択する。
RAPI_GR1_SCK3_1	グループ 1・SCK3 入出力を選択する。
RAPI_GR1_RXD_1	グループ 1・RXD 入力を選択する。
RAPI_GR1_TXD_1	グループ 1・TXD 出力を選択する。
RAPI_GR1_SCK3_2	グループ 1・SCK3_2 入出力を選択する。

RAPI_GR1_RXD_2	グループ 1・RXD_2 入力を選択する。
RAPI_GR1_TXD_2	グループ 1・TXD_2 出力を選択する。
RAPI_GR1_SCK3_3	グループ 1・SCK3_3 入出力を選択する。
RAPI_GR1_RXD_3	グループ 1・RXD_3 入力を選択する。
RAPI_GR1_TXD_3	グループ 1・TXD_3 出力を選択する。
RAPI_GR1_SSO	グループ 1・SSO 入出力を選択する。
RAPI_GR1_SSCK	グループ 1・SSCK 入出力を選択する。
RAPI_GR1_SCS_SDA	グループ 1・SDA/_SCS 入出力を選択する。
RAPI_GR1_SSI_SCL	グループ 1・SCL/SSI 入出力を選択する。
RAPI_GR1_TRAIO	グループ 1・TRAIO 入出力を選択する。
RAPI_GR1_TRAO	グループ 1・TRAO 出力を選択する。
RAPI_GR1_TRGB	グループ 1・TRGB 入力を選択する。
RAPI_GR1_TRBO	グループ 1・TRBO 出力を選択する。
RAPI_GR1_TRCOI	グループ 1・_TRCOI 入力を選択する。
RAPI_GR1_FTIOA	グループ 1・FTIOA 入出力を選択する。
RAPI_GR1_FTIOB	グループ 1・FTIOB 入出力を選択する。
RAPI_GR1_FTIOC	グループ 1・FTIOC 入出力を選択する。
RAPI_GR1_FTIOD	グループ 1・FTIOD 入出力を選択する。
RAPI_GR1_FTCI	グループ 1・FTCI 入力を選択する。
RAPI_GR1_TRDOI_0	グループ 1・_TRDOI_0 入力を選択する。
RAPI_GR1_FTIOA0	グループ 1・FTIOA0 入出力を選択する。
RAPI_GR1_FTIOB0	グループ 1・FTIOB0 入出力を選択する。
RAPI_GR1_FTIOC0	グループ 1・FTIOC0 入出力を選択する。
RAPI_GR1_FTIOD0	グループ 1・FTIOD0 入出力を選択する。
RAPI_GR1_FTIOA1	グループ 1・FTIOA1 入出力を選択する。
RAPI_GR1_FTIOB1	グループ 1・FTIOB1 入出力を選択する。
RAPI_GR1_FTIOC1	グループ 1・FTIOC1 入出力を選択する。
RAPI_GR1_FTIOD1	グループ 1・FTIOD1 入出力を選択する。
RAPI_GR1_TRDOI_1	グループ 1・_TRDOI_1 入力を選択する。
RAPI_GR1_TREO	グループ 1・TREO 出力を選択する。
RAPI_GR1_TCLKA	グループ 1・TCLKA 入力を選択する。
RAPI_GR1_TCLKB	グループ 1・TCLKB 入力を選択する。
RAPI_GR1_TGIOA	グループ 1・IGIOA 入出力を選択する。
RAPI_GR1_TGIOB	グループ 1・TGIOB 入出力を選択する。
RAPI_GR2_IRQ0	グループ 2・_IRQ0 入力を選択する。
RAPI_GR2_IRQ1	グループ 2・_IRQ1 入力を選択する。
RAPI_GR2_IRQ2	グループ 2・_IRQ2 入力を選択する。
RAPI_GR2_IRQ3	グループ 2・_IRQ3 入力を選択する。
RAPI_GR2_IRQ4	グループ 2・_IRQ4 入力を選択する。
RAPI_GR2_IRQ5	グループ 2・_IRQ5 入力を選択する。
RAPI_GR2_IRQ6	グループ 2・_IRQ6 入力を選択する。
RAPI_GR2_IRQ7	グループ 2・_IRQ7 入力を選択する。
RAPI_GR2_FTIOA2	グループ 2・FTIOA2 入出力を選択する。

RAPI_GR2_FTIOB2	グループ 2・FTIOB2 入出力を選択する。
RAPI_GR2_FTIOC2	グループ 2・FTIOC2 入出力を選択する。
RAPI_GR2_FTIOD2	グループ 2・FTIOD2 入出力を選択する。
RAPI_GR2_FTIOA3	グループ 2・FTIOA3 入出力を選択する。
RAPI_GR2_FTIOB3	グループ 2・FTIOB3 入出力を選択する。
RAPI_GR2_FTIOC3	グループ 2・FTIOC3 入出力を選択する。
RAPI_GR2_FTIOD3	グループ 2・FTIOD3 入出力を選択する。
RAPI_GR2_SCK3_1	グループ 2・SCK3 入出力を選択する。
RAPI_GR2_RXD_1	グループ 2・RXD 入力を選択する。
RAPI_GR2_TXD_1	グループ 2・TXD 出力を選択する。
RAPI_GR2_SCK3_3	グループ 2・SCK3_3 入出力を選択する。
RAPI_GR2_RXD_3	グループ 2・RXD_3 入力を選択する。
RAPI_GR2_TXD_3	グループ 2・TXD_3 出力を選択する。
RAPI_GR2_TRAIO	グループ 2・TRAIO 入出力を選択する。
RAPI_GR2_TRAO	グループ 2・TRAO 出力を選択する。
RAPI_GR2_TRGB	グループ 2・TRGB 入力を選択する。
RAPI_GR2_TRBO	グループ 2・TRBO 出力を選択する。
RAPI_GR2_TREO	グループ 2・TREO 出力を選択する。

戻り値

I/O ポートレジスタ指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

周辺機能マッピングコントローラ

備考

・第 1 引数および、第 2 引数に定義値以外の値を指定した場合の動作は保証されません。
・指定できる I/O ポートおよび、機能は CPU により異なります。
・周辺機能マッピングコントローラを有しない CPU では、本 API を使用することはできません。
・第 1 引数と第 2 引数に指定する定義値のグループは、すべて同一のグループになるように指定してください。

プログラム例

```
#include " rapi_pmc_h8s_20223.h"
```

```
void func( void )
{
    unsigned int function[8];

    function[0] = RAPI_GR1_IRQ0;
    function[1] = RAPI_GR1_IRQ1;
    function[2] = RAPI_GR1_IRQ2;
    function[3] = RAPI_GR1_IRQ3;
    function[4] = RAPI_GR1_IRQ4;
    function[5] = RAPI_GR1_IRQ5;
    function[6] = RAPI_GR1_IRQ6;
    function[7] = RAPI_GR1_IRQ7;

    /* Set function of port 3 */
    __ConfigurePMC( RAPI_GR1_PORT_3, function );
}
```

4.2.7 イベントリンクコントローラ
__SetEventLink

要約

〈 各周辺モジュールへのリンクするイベント信号の設定 〉

Boolean __SetEventLink(unsigned long data1, unsigned long data2, void* func)

data1	周辺モジュール、リンクするイベント信号および、割り込み制御モード
data2	data1 で指定した周辺モジュールの動作条件
func	コールバック関数ポインタ (data1 で指定した周辺モジュールに対応する割り込みがない場合および、 コールバック関数を設定しない場合は 0 を指定する)

解説

各周辺モジュールへのリンクするイベント信号および、イベント信号入力時の
動作条件を設定します。"|"記号を使用して複数の定義値を指定します。

[data1]

(H8S)

各周辺モジュールへのリンクするイベント信号を指定します。各周辺モジュールに
対応する定義値は次の通りです。

RAPI_ELC_TIMER_RA	タイマ RA を選択する。
RAPI_ELC_TIMER_RB	タイマ RB を選択する。
RAPI_ELC_TIMER_RC	タイマ RC を選択する。
RAPI_ELC_TIMER_RD0	タイマ RD_0 チャネル 0 を選択する。
RAPI_ELC_TIMER_RD1	タイマ RD_0 チャネル 1 を選択する。
RAPI_ELC_TIMER_RG	タイマ RG を選択する。
RAPI_ELC_ADC1	A/D 変換器ユニット 1 を選択する。
RAPI_ELC_ADC2	A/D 変換器ユニット 2 を選択する。
RAPI_ELC_INT1	ELC 割り込み 1 を選択する。
RAPI_ELC_ OUT_PORT_G2	出力ポートグループ 2 を選択する。
RAPI_ELC_ OUT_PORT_G3	出力ポートグループ 3 を選択する。
RAPI_ELC_IN_PORT_G2	入力ポートグループ 2 を選択する。
RAPI_ELC_IN_PORT_G3	入力ポートグループ 3 を選択する。
RAPI_ELC_ SINGLE_PORT_1	シングルポート 1 を選択する。
RAPI_ELC_ SINGLE_PORT_2	シングルポート 2 を選択する。
RAPI_ELC_ SINGLE_PORT_3	シングルポート 3 を選択する。
RAPI_ELC_ SINGLE_PORT_4	シングルポート 4 を選択する。
RAPI_ELC_CLOCK	クロック発振器を選択する。
RAPI_ELC_INT2	割り込み 2 を選択する。
RAPI_ELC_DAC0	D/A 変換器 0 を選択する。

RAPI_ELC_DAC1	D/A 変換器 1 を選択する。

リンクするイベント信号に対応する定義値は次の通りです。

RAPI_EV_ TIMER_RA_UDF	タイマ RA・アンダフロー信号を選択する。
RAPI_EV_ TIMER_RB_UDF	タイマ RB・アンダフロー信号を選択する。
RAPI_EV_ TIMER_RC_OVF	タイマ RC・オーバフロー信号を選択する。
RAPI_EV_ TIMER_RC_CMA	タイマ RC・コンペアマッチ A 信号を選択する。
RAPI_EV_ TIMER_RC_CMB	タイマ RC・コンペアマッチ B 信号を選択する。
RAPI_EV_ TIMER_RC_CMC	タイマ RC・コンペアマッチ C 信号を選択する。
RAPI_EV_ TIMER_RC_CMD	タイマ RC・コンペアマッチ D 信号を選択する。
RAPI_EV_ TIMER_RD0_OVF	タイマ RD0・チャネル 0・オーバフロー信号を選択する。
RAPI_EV_ TIMER_RD0_CMA	タイマ RD0・チャネル 0・コンペアマッチ A 信号を選択する。
RAPI_EV_ TIMER_RD0_CMB	タイマ RD0・チャネル 0・コンペアマッチ B 信号を選択する。
RAPI_EV_ TIMER_RD0_CMC	タイマ RD0・チャネル 0・コンペアマッチ C 信号を選択する。
RAPI_EV_ TIMER_RD0_CMD	タイマ RD0・チャネル 0・コンペアマッチ D 信号を選択する。
RAPI_EV_ TIMER_RD1_OVF	タイマ RD0・チャネル 1・オーバフロー信号を選択する。
RAPI_EV_ TIMER_RD1_UDF	タイマ RD0・チャネル 1・アンダフロー信号を選択する。
RAPI_EV_ TIMER_RD1_CMA	タイマ RD0・チャネル 1・コンペアマッチ A 信号を選択する。
RAPI_EV_ TIMER_RD1_CMB	タイマ RD0・チャネル 1・コンペアマッチ B 信号を選択する。
RAPI_EV_ TIMER_RD1_CMC	タイマ RD0・チャネル 1・コンペアマッチ C 信号を選択する。
RAPI_EV_ TIMER_RD1_CMD	タイマ RD0・チャネル 1・コンペアマッチ D 信号を選択する。
RAPI_EV_ TIMER_RG_OVF	タイマ RG・オーバフロー信号を選択する。
RAPI_EV_ TIMER_RG_UDF	タイマ RG・アンダフロー信号を選択する。
RAPI_EV_ TIMER_RG_CMA	タイマ RG・コンペアマッチ A 信号を選択する。

RAPI_EV_ TIMER_RG_CMB	タイマ RG・コンペアマッチ B 信号を選択する。
RAPI_EV_AD1	AD 変換器ユニット 1・AD 変換終了信号を選択する。
RAPI_EV_AD2	AD 変換器ユニット 2・AD 変換終了信号を選択する。
RAPI_EV_IN_PORT_G1	入力ポートグループ 1・端子入力値エッジ検出信号を選択する。
RAPI_EV_IN_PORT_G2	入力ポートグループ 2・端子入力値エッジ検出信号を選択する。
RAPI_EV_ SINGLE_PORT_1	シングル入力ポート 1・端子入力値エッジ検出信号を選択する。
RAPI_EV_ SINGLE_PORT_2	シングル入力ポート 2・端子入力値エッジ検出信号を選択する。
RAPI_EV_ SINGLE_PORT_3	シングル入力ポート 3・端子入力値エッジ検出信号を選択する。
RAPI_EV_ SINGLE_PORT_4	シングル入力ポート 4・端子入力値エッジ検出信号を選択する。
RAPI_EV_LVD	LVD・電圧降下検出信号を選択する。
RAPI_EV_LVD_RESET	LVD・電圧降下リセット検出信号を選択する。
RAPI_EV_CPG	CPG・バックアップ動作開始を選択する。
RAPI_EV_WDT	WDT・カウントアップ信号を選択する。
RAPI_EV_WDT_RESET	WDT・リセット信号を選択する。
RAPI_EV_TIMER_RE	タイマ RE・周期信号(周・日・時・分・秒から選択)を選択する。
RAPI_EV_DTC	DTC・転送完了を選択する。
RAPI_EV_ IIC2_SSU_T_EMPTY	IIC2/SSU・送信データエンプティを選択する。
RAPI_EV_ IIC2_SSU_T_END	IIC2/SSU・送信終了を選択する。
RAPI_EV_ IIC2_SSU_R_FULL	IIC2/SSU・受信データフルを選択する。
RAPI_EV_ IIC2_SSU_STOP	IIC2/SSU・停止条件検出を選択する。
RAPI_EV_ IIC2_SSU_ARB_OVE	IIC2/SSU・アービトレーションロスト/オーバランエラーを選択する。
RAPI_EV_ IIC2_SSU_NACK_CNF	IIC2/SSU・NACK 検出/コンクリクトエラーを選択する。
RAPI_EV_ SCI3_1_T_EMPTY	SCI3_1・送信データエンプティを選択する。
RAPI_EV_ SCI3_1_T_END	SCI3_1・送信終了を選択する。
RAPI_EV_ SCI3_1_R_FULL	SCI3_1・受信データフルを選択する。
RAPI_EV_SCI3_1_ERR	SCI3_1・転送エラーを選択する。
RAPI_EV_ SCI3_2_T_EMPTY	SCI3_2・送信データエンプティを選択する。
RAPI_EV_ SCI3_2_T_END	SCI3_2・送信終了を選択する。

RAPI_EV_ SCI3_2_R_FULL	SCI3_2・受信データフルを選択する。
RAPI_EV_SCI3_2_ERR	SCI3_2・転送エラーを選択する。
RAPI_EV_ SCI3_3_T_EMPTY	SCI3_3・送信データエンプティを選択する。
RAPI_EV_ SCI3_3_T_END	SCI3_3・送信終了を選択する。
RAPI_EV_ SCI3_3_R_FULL	SCI3_3・受信データフルを選択する。
RAPI_EV_SCI3_3_ERR	SCI3_3・転送エラーを選択する。
RAPI_EV_TIMER_ELC_0	タイマ ELC イベント信号 0 を選択する。
RAPI_EV_TIMER_ELC_1	タイマ ELC イベント信号 1 を選択する。
RAPI_EV_TIMER_ELC_2	タイマ ELC イベント信号 2 を選択する。
RAPI_EV_TIMER_ELC_3	タイマ ELC イベント信号 3 を選択する。

割り込み制御モードに対応する定義値は次の通りです。

RAPI_INT_MODE_0	割り込み制御モード 0 を指定する。
RAPI_INT_MODE_2	割り込み制御モード 2 を指定する。

全イベントリンクの有効/無効に対応する定義値は次のとおりです。

RAPI_ELC_DISABLE	全イベントリンクを無効にする。
RAPI_ELC_ENABLE	全イベントリンクを有効にする。

イベントリンク割り込み要求のあり/なしに対応する定義値は次のとおりです。

RAPI_NOT_INT_ REQUEST	選択したイベントリンク割り込みの割り込み要求をしない。
RAPI_INT_REQUEST	選択したイベントリンク割り込みの割り込み要求をする。

[data2]

第 1 引数 data1 で指定した周辺モジュールのイベント信号入力時の動作条件を指定します。
data2 には次の定義値、インタラプトプライオリティレジスタへ設定する割り込み
優先レベル(0～3)または、0 を指定します。複数の定義値を同時に指定する場合は、
"|"記号を使用して指定します。

(H8S)

RAPI_OP_TIMER_RA_ON	タイマ RA 動作・カウントスタート
RAPI_OP_TIMER_RA_EV	タイマ RA 動作・イベントカウンタ
RAPI_OP_ TIMER_RA_INVALID	タイマ RA 動作・イベント無効
RAPI_OP_TIMER_RB_ON	タイマ RB 動作・カウントスタート
RAPI_OP_TIMER_RB_EV	タイマ RB 動作・イベントカウンタ
RAPI_OP_ TIMER_RB_INVALID	タイマ RB 動作・イベント無効
RAPI_OP_TIMER_RC_ON	タイマ RC 動作・カウントスタート
RAPI_OP_TIMER_RC_EV	タイマ RC 動作・イベントカウンタ
RAPI_OP_TIMER_RC_IC	タイマ RC 動作・インプットキャプチャ

RAPI_OP_ TIMER_RC_INVALID	タイマ RC 動作・イベント無効
RAPI_OP_ TIMER_RD0_ON	タイマ RD_0 チャネル 0 動作・カウントスタート
RAPI_OP_ TIMER_RD0_EV	タイマ RD_0 チャネル 0 動作・イベントカウンタ
RAPI_OP_ TIMER_RD0_IC	タイマ RD_0 チャネル 0 動作・インプットキャプチャ
RAPI_OP_ TIMER_RD0_INVALID	タイマ RD_0 チャネル 0 動作・イベント無効
RAPI_OP_ TIMER_RD1_ON	タイマ RD_0 チャネル 1 動作・カウントスタート
RAPI_OP_ TIMER_RD1_EV	タイマ RD_0 チャネル 1 動作・イベントカウンタ
RAPI_OP_ TIMER_RD1_IC	タイマ RD_0 チャネル 1 動作・インプットキャプチャ
RAPI_OP_ TIMER_RD1_INVALID	タイマ RD_0 チャネル 1 動作・イベント無効
RAPI_OP_TIMER_RG_ON	タイマ RG 動作・カウントスタート
RAPI_OP_TIMER_RG_EV	タイマ RG 動作・イベントカウンタ
RAPI_OP_TIMER_RG_IC	タイマ RG 動作・インプットキャプチャ
RAPI_OP_ TIMER_RG_INVALID	タイマ RG 動作・イベント無効
RAPI_GR_PORT3_0	ポート P30 をポートグループ指定する。
RAPI_GR_PORT3_1	ポート P31 をポートグループ指定する。
RAPI_GR_PORT3_2	ポート P32 をポートグループ指定する。
RAPI_GR_PORT3_3	ポート P33 をポートグループ指定する。
RAPI_GR_PORT3_4	ポート P34 をポートグループ指定する。
RAPI_GR_PORT3_5	ポート P35 をポートグループ指定する。
RAPI_GR_PORT3_6	ポート P36 をポートグループ指定する。
RAPI_GR_PORT3_7	ポート P37 をポートグループ指定する。
RAPI_GR_PORT6_0	ポート P60 をポートグループ指定する。
RAPI_GR_PORT6_1	ポート P61 をポートグループ指定する。
RAPI_GR_PORT6_2	ポート P62 をポートグループ指定する。
RAPI_GR_PORT6_3	ポート P63 をポートグループ指定する。
RAPI_GR_PORT6_4	ポート P64 をポートグループ指定する。
RAPI_GR_PORT6_5	ポート P65 をポートグループ指定する。
RAPI_GR_PORT6_6	ポート P66 をポートグループ指定する。
RAPI_GR_PORT6_7	ポート P67 をポートグループ指定する。
RAPI_PGR3_OUT_0	出力ポート P3 グループ動作・イベント入力時、0 を出力を選択
RAPI_PGR3_OUT_1	出力ポート P3 グループ動作・イベント入力時、1 を出力を選択
RAPI_PGR3_ OUT_TOGGLE	出力ポート P3 グループ動作・イベント入力時、トグル(反転)出力を選択

RAPI_PGR3_ OUT_BUFFER	出力ポート P3 グループ動作・イベント入力時、バッファ値を出力を選択
RAPI_PGR3_ OUT_BIT_SHIFT	出力ポート P3 グループ動作・イベント入力時、グループ内で ビットシフト出力(MSB→LSB へシフト)を選択
RAPI_PGR6_OUT_0	出力ポート P6 グループ動作・イベント入力時、0 を出力を選択
RAPI_PGR6_OUT_1	出力ポート P6 グループ動作・イベント入力時、1 を出力を選択
RAPI_PGR6_ OUT_TOGGLE	出力ポート P6 グループ動作・イベント入力時、トグル(反転)出力を選択
RAPI_PGR6_ OUT_BUFFER	出力ポート P6 グループ動作・イベント入力時、バッファ値を出力を選択
RAPI_PGR6_ OUT_BIT_SHIFT	出力ポート P6 グループ動作・イベント入力時、グループ内で ビットシフト出力(MSB→LSB へシフト)を選択
RAPI_PGR3_ PDBF_OVER	入力ポート P3 グループ動作・PDBF レジスタへの上書き有効を選択
RAPI_PGR3_ IN_RISING	入力ポート P3 グループ動作・外部入力信号の立ち上がりエッジを検出して、 イベント発生を選択
RAPI_PGR3_ IN_FALLING	入力ポート P3 グループ動作・外部入力信号の立ち下がりエッジを検出して、 イベント発生を選択
RAPI_PGR3_ IN_BOTH	入力ポート P3 グループ動作・外部入力信号の立ち上がり／立ち下がりの 両エッジを検出して、イベント発生を選択
RAPI_PGR6_ PDBF_OVER	入力ポート P6 グループ動作・PDBF レジスタへの上書き有効を選択
RAPI_PGR6_ IN_RISING	入力ポート P6 グループ動作・外部入力信号の立ち上がりエッジを検出して、 イベント発生を選択
RAPI_PGR6_ IN_FALLING	入力ポート P6 グループ動作・外部入力信号の立ち下がりエッジを検出して、 イベント発生を選択
RAPI_PGR6_IN_BOTH	入力ポート P6 グループ動作・外部入力信号の立ち上がり／立ち下がりの 両エッジを検出して、イベント発生を選択
RAPI_SP_OUT_0	出力シングルポート動作・イベント入力時、0 を出力
RAPI_SP_OUT_1	出力シングルポート動作・イベント入力時、1 を出力
RAPI_SP_OUT_TOGGLE	出力シングルポート動作・イベント入力時、トグル(反転)出力
RAPI_SP_IN_RISING	入力シングルポート動作・立ち上がりエッジを検出して、イベント出力
RAPI_SP_IN_FALLING	入力シングルポート動作・立ち下がりエッジを検出して、イベント出力
RAPI_SP_IN_BOTH	入力シングルポート動作・立ち上がり/立ち下がりの両エッジを検出して、 イベント出力
RAPI_SP_P3_0	ポート P30 をシングルポートに選択する。
RAPI_SP_P3_1	ポート P31 をシングルポートに選択する。
RAPI_SP_P3_2	ポート P32 をシングルポートに選択する。
RAPI_SP_P3_3	ポート P33 をシングルポートに選択する。
RAPI_SP_P3_4	ポート P34 をシングルポートに選択する。
RAPI_SP_P3_5	ポート P35 をシングルポートに選択する。
RAPI_SP_P3_6	ポート P36 をシングルポートに選択する。
RAPI_SP_P3_7	ポート P37 をシングルポートに選択する。

RAPI_SP_P6_0	ポート P60 をシングルポートに選択する。
RAPI_SP_P6_1	ポート P61 をシングルポートに選択する。
RAPI_SP_P6_2	ポート P62 をシングルポートに選択する。
RAPI_SP_P6_3	ポート P63 をシングルポートに選択する。
RAPI_SP_P6_4	ポート P64 をシングルポートに選択する。
RAPI_SP_P6_5	ポート P65 をシングルポートに選択する。
RAPI_SP_P6_6	ポート P66 をシングルポートに選択する。
RAPI_SP_P6_7	ポート P67 をシングルポートに選択する。

・タイマ RA（RAPI_ELC_TIMER_RA を指定時）のイベントリンク時の動作指定

（動作条件）　{ RAPI_OP_TIMER_RA_ON, RAPI_OP_TIMER_RA_EV, RAPI_OP_TIMER_RA_INVALID }から 1 つを指定します。

（コールバック関数ポインタ）　0 を指定します。

・タイマ RB（RAPI_ELC_TIMER_RB を指定時）のイベントリンク時の動作指定

（動作条件）　{ RAPI_OP_TIMER_RB_ON, RAPI_OP_TIMER_RB_EV, RAPI_OP_TIMER_RB_INVALID }から 1 つを指定します。

（コールバック関数ポインタ）　0 を指定します。

・タイマ RC（RAPI_ELC_TIMER_RC を指定時）のイベントリンク時の動作指定

（動作条件）　{ RAPI_OP_TIMER_RC_ON, RAPI_OP_TIMER_RC_EV, RAPI_OP_TIMER_RC_IC, RAPI_OP_TIMER_RC_INVALID }から 1 つを指定します。

（コールバック関数ポインタ）　0 を指定します。

・タイマ RD_0 チャネル 0（RAPI_ELC_TIMER_RD0 を指定時）のイベントリンク時の動作指定

（動作条件）　{ RAPI_OP_TIMER_RD0_ON, RAPI_OP_TIMER_RD0_EV, RAPI_OP_TIMER_RD0_IC, RAPI_OP_TIMER_RD0_INVALID }から 1 つを指定します。

（コールバック関数ポインタ）　0 を指定します。

・タイマ RD_0 チャネル 1（RAPI_ELC_TIMER_RD1 を指定時）のイベントリンク時の動作指定

（動作条件）　{ RAPI_OP_TIMER_RD1_ON, RAPI_OP_TIMER_RD1_EV, RAPI_OP_TIMER_RD1_IC, RAPI_OP_TIMER_RD1_INVALID }から 1 つを指定します。

（コールバック関数ポインタ）　0 を指定します。

・タイマ RG（RAPI_ELC_TIMER_RG を指定時）のイベントリンク時の動作指定

（動作条件）　{ RAPI_OP_TIMER_RG_ON, RAPI_OP_TIMER_RG_EV, RAPI_OP_TIMER_RG_IC, RAPI_OP_TIMER_RG_INVALID }から 1 つを指定します。

（コールバック関数ポインタ）　0 を指定します。

・A/D 変換器ユニット(RAPI_ELC_ADC1～RAPI_ELC_ADC2 を指定時）のイベントリンク時の動作指定

（動作条件）　0 を指定します。

（コールバック関数ポインタ）　0 を指定します。

・D/A 変換器（RAPI_ELC_DAC0～RAPI_ELC_DAC1 を指定時）のイベントリンク時の動作指定

（動作条件）　　0 を指定します。

（コールバック関数ポインタ）　　0 を指定します。

・クロック発振器(RAPI_ELC_CLOCK を指定時）のイベントリンク時の動作指定

（動作条件）　　0 を指定します。

（コールバック関数ポインタ）　　0 を指定します。

・ELC 割り込み（RAPI_ELC_INT1～RAPI_ELC_INT2 を指定時）のイベントリンク時の動作指定

（動作条件）　　0 を指定します。

（コールバック関数ポインタ）　　コールバック関数ポインタを指定します。

・出力ポートグループ 1（RAPI_ELC_OUT_PORT_G2 を指定時）のイベントリンク時の動作指定

（動作条件）　　{ RAPI_GR_PORT3_0, RAPI_GR_PORT3_1, RAPI_GR_PORT3_2,
RAPI_GR_PORT3_3, RAPI_GR_PORT3_4, RAPI_GR_PORT3_5,
RAPI_GR_PORT3_6, RAPI_GR_PORT3_7 }から使用するポートを
すべて指定します。
{ RAPI_PGR3_OUT_0, RAPI_PGR3_OUT_1, RAPI_PGR3_OUT_TOGGLE,
RAPI_PGR3_OUT_BUFFER, RAPI_PGR3_OUT_BIT_SHIFT }から出力動作を
1 つ指定します。

（コールバック関数ポインタ）　　0 を指定します。

・出力ポートグループ 2（RAPI_ELC_OUT_PORT_G3 を指定時）のイベントリンク時の動作指定

（動作条件）　　{ RAPI_GR_PORT6_0, RAPI_GR_PORT6_1, RAPI_GR_PORT6_2,
RAPI_GR_PORT6_3, RAPI_GR_PORT6_4, RAPI_GR_PORT6_5,
RAPI_GR_PORT6_6, RAPI_GR_PORT6_7 }から使用するポートを
すべて指定します。
{ RAPI_PGR6_OUT_0, RAPI_PGR6_OUT_1, RAPI_PGR6_OUT_TOGGLE,
RAPI_PGR6_OUT_BUFFER, RAPI_PGR6_OUT_BIT_SHIFT }から出力動作を
1 つ指定します。

（コールバック関数ポインタ）　　0 を指定します。

・入力ポートグループ 1（RAPI_ELC_IN_PORT_G2 を指定時）のイベントリンク時の動作指定

（動作条件）　　{ RAPI_GR_PORT3_0, RAPI_GR_PORT3_1, RAPI_GR_PORT3_2,
RAPI_GR_PORT3_3, RAPI_GR_PORT3_4, RAPI_GR_PORT3_5,
RAPI_GR_PORT3_6, RAPI_GR_PORT3_7 }から使用するポートを
すべて指定します。
{ RAPI_PGR3_PDBF_OVER, RAPI_PGR3_IN_RISING, RAPI_PGR3_IN_FALLING,
RAPI_PGR3_IN_BOTH }から出力動作を 1 つ指定します。

（コールバック関数ポインタ）　　0 を指定します。

・入力ポートグループ 2（RAPI_ELC_OUT_PORT_G3 を指定時）のイベントリンク時の動作指定

（動作条件）　　{ RAPI_GR_PORT6_0, RAPI_GR_PORT6_1, RAPI_GR_PORT6_2,
RAPI_GR_PORT6_3, RAPI_GR_PORT6_4, RAPI_GR_PORT6_5,
RAPI_GR_PORT6_6, RAPI_GR_PORT6_7 }から使用するポートを

すべて指定します。
{ RAPI_PGR6_PDBF_OVER, RAPI_PGR6_IN_RISING, RAPI_PGR6_IN_FALLING, RAPI_PGR6_IN_BOTH }から出力動作を 1 つ指定します。

（コールバック関数ポインタ）　0 を指定します。

・シングルポート 1～4（RAPI_ELC_SINGLE_PORT_1～RAPI_ELC_SINGLE_PORT_4 を指定時）のイベントリンク時の動作指定

（動作条件）　{ RAPI_SP_PORT3_0, RAPI_SP_PORT3_1, RAPI_SP_PORT3_2, RAPI_SP_PORT3_3, RAPI_SP_PORT3_4, RAPI_SP_PORT3_5, RAPI_SP_PORT3_6, RAPI_SP_PORT3_7, RAPI_SP_PORT6_0, RAPI_SP_PORT6_1, RAPI_SP_PORT6_2, RAPI_SP_PORT6_3, RAPI_SP_PORT6_4, RAPI_SP_PORT6_5, RAPI_SP_PORT6_6, RAPI_SP_PORT6_7 }から使用するポートを 1 つ指定します。
{ RAPI_SP_OUT_0, RAPI_SP_OUT_1, RAPI_SP_OUT_TOGGLE, RAPI_SP_IN_RISING, RAPI_SP_IN_FALLING, RAPI_SP_IN_BOTH }から出力または入力動作を 1 つ指定します。

（コールバック関数ポインタ）　0 を指定します。

戻り値

周辺モジュールの指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

イベントリンクコントローラ

備考

・第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・指定できる周辺モジュールポートおよび、リンクするイベント信号は CPU により異なります。
・イベントリンクコントローラを有しない CPU では、本 API を使用することはできません。

プログラム例

```
#include " rapi_elc_h8s_20223.h"

void func( void )
{
    /* Set event of timer RA */
    __SetEventLink( RAPI_ELC_TIMER_RA|RAPI_EV_TIMER_RB_UDF, RAPI_OP_TIMER_RA_ON, 0 );
}
```

__CreateEventGenerateTimer

要約

〈 イベント発生タイマの動作条件の設定 〉

Boolean __CreateEventGenerateTimer(unsigned long data1, unsigned int data2)

data1	イベント発生タイマの動作条件
data2	タイマ ELC カウンタへの設定値

解説

指定したタイマを設定します。

[data1]

data1 には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(H8S)

RAPI_TIMER_ELC	イベント発生タイマを使用する。
RAPI_ELC_TIMER_ON	__CreateEventGenerateTimer 内でタイマを動作開始に設定する。
RAPI_ELC_TIMER_OFF	__CreateEventGenerateTimer 内ではタイマを動作停止に設定する。
RAPI_ELC_TIMER_F1	カウントソースにφを選択する。
RAPI_ELC_TIMER_F2	カウントソースにφ/2 を選択する。
RAPI_ELC_TIMER_F4	カウントソースにφ/4 を選択する。
RAPI_ELC_TIMER_F8	カウントソースにφ/8 を選択する。
RAPI_ELC_TIMER_F16	カウントソースにφ/16 を選択する。
RAPI_ELC_TIMER_F32	カウントソースにφ/32 を選択する。
RAPI_ELC_TIMER_F64	カウントソースにφ/64 を選択する。
RAPI_ELC_TIMER_F128	カウントソースにφ/128 を選択する。
RAPI_ELC_TIMER_F256	カウントソースにφ/256 を選択する。
RAPI_ELC_TIMER_F512	カウントソースにφ/512 を選択する。
RAPI_ELC_ TIMER_F1024	カウントソースにφ/1024 を選択する。
RAPI_ELC_ TIMER_F2048	カウントソースにφ/2048 を選択する。
RAPI_ELC_ TIMER_F4096	カウントソースにφ/4096 を選択する。
RAPI_ELC_ TIMER_F8192	カウントソースにφ/8192 を選択する。
RAPI_ELC_CH0_F1	チャネル 0 イベント発生周期にクロックソースφELC/1 を選択する。
RAPI_ELC_CH0_F2	チャネル 0 イベント発生周期にクロックソースφELC/2 を選択する。
RAPI_ELC_CH0_F4	チャネル 0 イベント発生周期にクロックソースφELC/4 を選択する。
RAPI_ELC_CH0_F8	チャネル 0 イベント発生周期にクロックソースφELC/8 を選択する。
RAPI_ELC_CH0_F16	チャネル 0 イベント発生周期にクロックソースφELC/16 を選択する。
RAPI_ELC_CH0_F32	チャネル 0 イベント発生周期にクロックソースφELC/32 を選択する。
RAPI_ELC_CH0_F64	チャネル 0 イベント発生周期にクロックソースφELC/64 を選択する。
RAPI_ELC_CH0_F128	チャネル 0 イベント発生周期にクロックソースφELC/128 を選択する。
RAPI_ELC_CH0_F256	チャネル 0 イベント発生周期にクロックソースφELC/256 を選択する。
RAPI_ELC_CH0_F512	チャネル 0 イベント発生周期にクロックソースφELC/512 を選択する。

RAPI_ELC_CH0_F1024	チャネル 0 イベント発生周期にクロックソースφELC/1024 を選択する。
RAPI_ELC_CH0_F2048	チャネル 0 イベント発生周期にクロックソースφELC/2048 を選択する。
RAPI_ELC_CH0_F4096	チャネル 0 イベント発生周期にクロックソースφELC/4096 を選択する。
RAPI_ELC_CH0_F8192	チャネル 0 イベント発生周期にクロックソースφELC/8192 を選択する。
RAPI_ELC_CH0_F16384	チャネル 0 イベント発生周期にクロックソースφELC/16384 を選択する。
RAPI_ELC_CH0_F32768	チャネル 0 イベント発生周期にクロックソースφELC/32768 を選択する。
RAPI_ELC_CH1_F1	チャネル 1 イベント発生周期にクロックソースφELC/1 を選択する。
RAPI_ELC_CH1_F2	チャネル 1 イベント発生周期にクロックソースφELC/2 を選択する。
RAPI_ELC_CH1_F4	チャネル 1 イベント発生周期にクロックソースφELC/4 を選択する。
RAPI_ELC_CH1_F8	チャネル 1 イベント発生周期にクロックソースφELC/8 を選択する。
RAPI_ELC_CH1_F16	チャネル 1 イベント発生周期にクロックソースφELC/16 を選択する。
RAPI_ELC_CH1_F32	チャネル 1 イベント発生周期にクロックソースφELC/32 を選択する。
RAPI_ELC_CH1_F64	チャネル 1 イベント発生周期にクロックソースφELC/64 を選択する。
RAPI_ELC_CH1_F128	チャネル 1 イベント発生周期にクロックソースφELC/128 を選択する。
RAPI_ELC_CH1_F256	チャネル 1 イベント発生周期にクロックソースφELC/256 を選択する。
RAPI_ELC_CH1_F512	チャネル 1 イベント発生周期にクロックソースφELC/512 を選択する。
RAPI_ELC_CH1_F1024	チャネル 1 イベント発生周期にクロックソースφELC/1024 を選択する。
RAPI_ELC_CH1_F2048	チャネル 1 イベント発生周期にクロックソースφELC/2048 を選択する。
RAPI_ELC_CH1_F4096	チャネル 1 イベント発生周期にクロックソースφELC/4096 を選択する。
RAPI_ELC_CH1_F8192	チャネル 1 イベント発生周期にクロックソースφELC/8192 を選択する。
RAPI_ELC_CH1_F16384	チャネル 1 イベント発生周期にクロックソースφELC/16384 を選択する。
RAPI_ELC_CH1_F32768	チャネル 1 イベント発生周期にクロックソースφELC/32768 を選択する。
RAPI_ELC_CH2_F1	チャネル 2 イベント発生周期にクロックソースφELC/1 を選択する。
RAPI_ELC_CH2_F2	チャネル 2 イベント発生周期にクロックソースφELC/2 を選択する。
RAPI_ELC_CH2_F4	チャネル 2 イベント発生周期にクロックソースφELC/4 を選択する。
RAPI_ELC_CH2_F8	チャネル 2 イベント発生周期にクロックソースφELC/8 を選択する。
RAPI_ELC_CH2_F16	チャネル 2 イベント発生周期にクロックソースφELC/16 を選択する。
RAPI_ELC_CH2_F32	チャネル 2 イベント発生周期にクロックソースφELC/32 を選択する。
RAPI_ELC_CH2_F64	チャネル 2 イベント発生周期にクロックソースφELC/64 を選択する。
RAPI_ELC_CH2_F128	チャネル 2 イベント発生周期にクロックソースφELC/128 を選択する。
RAPI_ELC_CH2_F256	チャネル 2 イベント発生周期にクロックソースφELC/256 を選択する。
RAPI_ELC_CH2_F512	チャネル 2 イベント発生周期にクロックソースφELC/512 を選択する。
RAPI_ELC_CH2_F1024	チャネル 2 イベント発生周期にクロックソースφELC/1024 を選択する。
RAPI_ELC_CH2_F2048	チャネル 2 イベント発生周期にクロックソースφELC/2048 を選択する。
RAPI_ELC_CH2_F4096	チャネル 2 イベント発生周期にクロックソースφELC/4096 を選択する。
RAPI_ELC_CH2_F8192	チャネル 2 イベント発生周期にクロックソースφELC/8192 を選択する。
RAPI_ELC_CH2_F16384	チャネル 2 イベント発生周期にクロックソースφELC/16384 を選択する。
RAPI_ELC_CH2_F32768	チャネル 2 イベント発生周期にクロックソースφELC/32768 を選択する。
RAPI_ELC_CH3_F1	チャネル 3 イベント発生周期にクロックソースφELC/1 を選択する。
RAPI_ELC_CH3_F2	チャネル 3 イベント発生周期にクロックソースφELC/2 を選択する。
RAPI_ELC_CH3_F4	チャネル 3 イベント発生周期にクロックソースφELC/4 を選択する。
RAPI_ELC_CH3_F8	チャネル 3 イベント発生周期にクロックソースφELC/8 を選択する。
RAPI_ELC_CH3_F16	チャネル 3 イベント発生周期にクロックソースφELC/16 を選択する。

R_API_ELC_CH3_F32	チャネル 3 イベント発生周期にクロックソースφELC/32 を選択する。
R_API_ELC_CH3_F64	チャネル 3 イベント発生周期にクロックソースφELC/64 を選択する。
R_API_ELC_CH3_F128	チャネル 3 イベント発生周期にクロックソースφELC/128 を選択する。
R_API_ELC_CH3_F256	チャネル 3 イベント発生周期にクロックソースφELC/256 を選択する。
R_API_ELC_CH3_F512	チャネル 3 イベント発生周期にクロックソースφELC/512 を選択する。
R_API_ELC_CH3_F1024	チャネル 3 イベント発生周期にクロックソースφELC/1024 を選択する。
R_API_ELC_CH3_F2048	チャネル 3 イベント発生周期にクロックソースφELC/2048 を選択する。
R_API_ELC_CH3_F4096	チャネル 3 イベント発生周期にクロックソースφELC/4096 を選択する。
R_API_ELC_CH3_F8192	チャネル 3 イベント発生周期にクロックソースφELC/8192 を選択する。
R_API_ELC_CH3_F16384	チャネル 3 イベント発生周期にクロックソースφELC/16384 を選択する。
R_API_ELC_CH3_F32768	チャネル 3 イベント発生周期にクロックソースφELC/32768 を選択する。
R_API_ELC_ CH0_NO_DELAY	チャネル 0 遅延時間に遅延なしを選択する。
R_API_ELC_ CH0_DELAY_1	チャネル 0 遅延時間に 1 クロックサイクルを選択する。
R_API_ELC_ CH0_DELAY_2	チャネル 0 遅延時間に 2 クロックサイクルを選択する。
R_API_ELC_ CH0_DELAY_3	チャネル 0 遅延時間に 3 クロックサイクルを選択する。
R_API_ELC_ CH1_NO_DELAY	チャネル 1 遅延時間に遅延なしを選択する。
R_API_ELC_ CH1_DELAY_1	チャネル 1 遅延時間に 1 クロックサイクルを選択する。
R_API_ELC_ CH1_DELAY_2	チャネル 1 遅延時間に 2 クロックサイクルを選択する。
R_API_ELC_ CH1_DELAY_3	チャネル 1 遅延時間に 3 クロックサイクルを選択する。
R_API_ELC_ CH2_NO_DELAY	チャネル 2 遅延時間に遅延なしを選択する。
R_API_ELC_ CH2_DELAY_1	チャネル 2 遅延時間に 1 クロックサイクルを選択する。
R_API_ELC_ CH2_DELAY_2	チャネル 2 遅延時間に 2 クロックサイクルを選択する。
R_API_ELC_ CH2_DELAY_3	チャネル 2 遅延時間に 3 クロックサイクルを選択する。
R_API_ELC_ CH3_NO_DELAY	チャネル 3 遅延時間に遅延なしを選択する。
R_API_ELC_ CH3_DELAY_1	チャネル 3 遅延時間に 1 クロックサイクルを選択する。
R_API_ELC_ CH3_DELAY_2	チャネル 3 遅延時間に 2 クロックサイクルを選択する。
R_API_ELC_ CH3_DELAY_3	チャネル 3 遅延時間に 3 クロックサイクルを選択する。

戻り値　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。

機能　イベントリンクコントローラ

参照　__EnableEventGenerateTimer, __DestroyEventGenerateTimer

備考
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- イベントリンクコントローラを有しない CPU では、本 API を使用することはできません。

プログラム例

```
#include " rapi_elc_h8s_20223.h"

void func( void )
{
    /* Set up ELC timer */
    __CreateEventGenerateTimer( RAPI_TIMER_ELC|RAPI_ELC_TIMER_ON|RAPI_ELC_TIMER_F8,
                          RAPI_ELC_CH0_F16|RAPI_ELC_CH3_F2|RAPI_ELC_CH0_DELAY_2 );
}
```

__EnableEventGenerateTimer

要約

< イベント発生タイマの動作制御 >

Boolean __EnableEventGenerateTimer(unsigned long data)

data	イベント発生タイマの動作条件

解説

イベント発生タイマの動作開始/動作停止を制御します。

[data]

data には、次の定義値を設定することができます。複数の定義値を同時に指定する場合は、"|"記号を使用して指定します。

(H8S)

RAPI_TIMER_ELC	イベント発生タイマを選択する。
RAPI_ELC_TIMER_ON	タイマ ELC を動作開始に設定する。
RAPI_ELC_TIMER_OFF	タイマ ELC を動作停止に設定する。

戻り値

動作制御に成功した場合は RAPI_TRUE を返し、動作制御に失敗した場合は RAPI_FALSE を返します。

機能

イベントリンクコントローラ

参照

__CreateEventGenerateTimer, __DestroyEventGenerateTimer

備考

・引数に定義値以外の値を指定した場合の動作は保証されません。
・イベントリンクコントローラを有しない CPU では、本 API を使用することはできません。

プログラム例

```
#include " rapi_elc_h8s_20223.h"

void func( void )
{
    /* Disable ELC timer */
    __EnableEventGenerateTimer( RAPI_TIMER_ELC|RAPI_ELC_TIMER_OFF );
}
```

__DestroyEventGenerateTimer

要約

〈 イベント発生タイマの動作条件の設定破棄 〉

Boolean __DestroyEventGenerateTimer(unsigned long data)

data	設定を破棄するイベント発生タイマ

解説

イベント発生タイマの設定を破棄します。

[data]

data には、次の定義値を設定することができます。

(H8S)

RAPI_TIMER_ELC	イベント発生タイマを選択する。

戻り値

設定破棄に成功した場合は RAPI_TRUE を返し、設定破棄に失敗した場合は RAPI_FALSE を返します。

機能

イベントリンクコントローラ

参照

__CreateEventCounter, __EnableEventCounter, __GetEventCounter

備考

・引数に定義値以外の値を指定した場合の動作は保証されません。
・イベントリンクコントローラを有しない CPU では、本 API を使用することはできません。

プログラム例

```
#include " rapi_elc_h8s_20223.h"

void func( void )
{
    /* Destroy the setting of ELC timer */
    __DestroyEventGenerateTimer( RAPI_TIMER_ELC );
}
```

__ReadPortBufferRegister

要約

< ポートバッファデータの読み出し >

Boolean __ReadPortBufferRegister(unsigned long data1, unsigned int *data2)

data1	ポートバッファデータを読み出すポート
data2	ポートバッファレジスタから読み出した値を格納する変数のポインタ

解説

指定したポートのポートバッファレジスタの値を取得します。

[data1]

ポートバッファレジスタのデータを読み出すポートを指定します。各ポートに対応する定義値は次の通りです。

(H8S)

RAPI_PORT_3	ポート P3 レジスタ
RAPI_PORT_6	ポート P6 レジスタ

戻り値

ポート指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

イベントリンクコントローラ

参照

__WritePortBufferRegister

備考

・第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・イベントリンクコントローラを有しない CPU では、本 API を使用することはできません。

プログラム例

```
#include " rapi_elc_h8s_20223.h"

void func( void )
{
    unsigned int data;

    /* Get the value of port buffer register 2 */
    __ReadPortBufferRegister( RAPI_PORT_6, &data );
}
```

__WritePortBufferRegister

要約

〈 ポートバッファへのデータの書き込み 〉

Boolean __WritePortBufferRegister(unsigned long data1, unsigned int data2)

data1	ポートバッファへのデータを書き込むポート
data2	ポートバッファレジスタに書き込むデータ

解説

指定したポートのポートバッファレジスタへ値を設定します。

[data1]

ポートバッファレジスタにデータを書き込むポートを指定します。各ポートに対応する定義値は次の通りです。

(H8S)

RAPI_PORT_3	ポート P3 レジスタ
RAPI_PORT_6	ポート P6 レジスタ

戻り値

ポート指定が正しくない場合は RAPI_FALSE を返し、それ以外は RAPI_TRUE を返します。

機能

イベントリンクコントローラ

参照

__ReadPortBufferRegister

備考

・ 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
・ イベントリンクコントローラを有しない CPU では、本 API を使用することはできません。

プログラム例

```
#include " rapi_elc_h8s_20223.h"

void func( void )
{
    /* Set the data to port buffer register 1 */
    __WritePortBufferRegister( RAPI_PORT_3, 0xFF );
}
```

4.2.8 データトランスファコントローラ
__CreateDTC

要約

< DTC レジスタ情報の設定 >

Boolean __CreateDTC(unsigned long RegInfoAddr, unsigned short mra_mrb, unsigned long sar, unsigned long dar, unsigned short cra, unsigned short crb)

RegInfoAddr	DTC レジスタ情報配置アドレス（内臓 RAM 上）
mra_mrb	転送モード A,B 設定データ
sar	ソースアドレス（転送元）
dar	ディスティネーションアドレス（転送先）
cra	転送カウントデータ A
crb	転送カウントデータ B

解説

指定したアドレス（内臓 RAM 上）へ DTC レジスタ情報を設定します。

[RegInfoAddr]

内臓 RAM 上の任意のアドレス(H’ FFDF80-H’ FFFF7F)を指定します。

[mra_mrb]

転送モード A,B を指定します。各モードの定義値は以下の通りです。

(H8S)

RAPI_DTC_MRA_SAR_FIX	SAR は固定
RAPI_DTC_MRA_SAR_INC	転送後 SAR をインクリメント(Sz=0 のとき+1、Sz=1 のとき+2)
RAPI_DTC_MRA_SAR_DEC	転送後 SAR をディクリメント(Sz=0 のとき−1、Sz=1 のとき−2)
RAPI_DTC_MRA_DAR_FIX	DAR は固定
RAPI_DTC_MRA_DAR_INC	転送後 DAR をインクリメント(Sz=0 のとき+1、Sz=1 のとき+2)
RAPI_DTC_MRA_DAR_DEC	転送後 DAR をディクリメント(Sz=0 のとき−1、Sz=1 のとき−2)
RAPI_DTC_MRA_NORMAL	ノーマルモード
RAPI_DTC_MRA_REPEAT	リピートモード
RAPI_DTC_MRA_BLOCK	ブロック転送モード
RAPI_DTC_MRA_SEL_SAR	ソース側がリピート領域またはブロック領域
RAPI_DTC_MRA_SEL_DAR	デスティネーション側がリピート領域またはブロック領域
RAPI_DTC_MRA_BYTE_SIZE	バイトサイズ転送
RAPI_DTC_MRA_WORD_SIZE	ワードサイズ転送
RAPI_DTC_MRB_NO_CHAIN	チェイン転送を行わない
RAPI_DTC_MRB_ENABLE_CHAIN	チェイン転送を行う
RAPI_DTC_MRB_COMPLETE_IRQ	指定されたデータ転送を終了したときだけ CPU に対して割り込み要求を発生
RAPI_DTC_MRB_EVERY_IRQ	DTC 転送を行うたびに CPU に対して割り込みを発生
RAPI_DTC_MRB_SEQ_CHAIN	連続してチェイン転送を行う
RAPI_DTC_MRB_ZERO_CHAIN	転送カウンタ=0 のときのみチェイン転送を行う

[sar]

ソースアドレス（転送元）を指定します。

[dar]

ディスティネーションアドレス（転送先）を指定します。

[cra]

DTC 転送カウントレジスタ A を指定します。

[crb]

DTC 転送カウントレジスタ B を指定します。

戻り値

常に RAPI_TRUE を返します。

機能

データトランスファコントローラ

備考

・ 第 1 引数、第 2 引数、第 3 引数、第 4 引数、第 5 引数、第 6 引数に定義値以外の値を指定した場合の動作は保証されません。

・ cra は転送モードにより 16 ビット/8 ビットになります。
ノーマルモード：16 ビット　リピート/ブロック転送モード：8 ビット×2

・ crb は転送モードにより使用有無が異なる。
ノーマル/リピートモード：使用しない　ブロック転送モード：使用する

プログラム例

```
#include " rapi_dtc_h8s_20203.h"

void main(void)
{
          unsigned long MRA_MRB;
          Boolean RESULT;

          MRA_MRB = (RAPI_DTC_MRA_SAR_INC | RAPI_DTC_MRA_DAR_INC | RAPI_DTC_MRA_BLOCK |
RAPI_DTC_MRA_SEL_SAR | RAPI_DTC_MRA_BYTE_SIZE |
RAPI_DTC_MRB_NO_CHAIN | RAPI_DTC_MRB_COMPLETE_IRQ | RAPI_DTC_MRB_SEQ_CHAIN);
         /*H' FFF000 に DTC レジスタ情報配置
           ソースアドレス：インクリメント
           ディスティネーションアドレス：インクリメント
           ブロック転送モード
           ブロック領域はソースアドレス
           バイトサイズ転送
           チェイン転送無し
           転送終了時割り込み
           連続してチェイン転送を行う
           ソースアドレス:H' 00003000
           ディスティネーションアドレス:H' 00FFE000
           ブロックサイズ H' FF
           転送回数:H' 0001*/
           RESULT = __CreateDTC( 0xfff000, MRA_MRB, 0x00003000, 0x00FFE000, 0x0000, 0x0001);
}
```

__EnableDTC

要約

<割り込みによるDTC起動の有効化>

Boolean __EnableDTC(unsigned long signal)

signal	DTC 起動要因

解説

指定されたDTC起動要因に対応するDTC イネーブルレジスタのDTCEビットを有効に設定します。

[signal]

各要因に対応する定義値は以下の通りです。

(H8S)

RAPI_DTCERA_IRQ0	IRQ0 をDTC起動要因有効に設定
RAPI_DTCERA_IRQ1	IRQ1 をDTC起動要因有効に設定
RAPI_DTCERA_IRQ2	IRQ2 をDTC起動要因有効に設定
RAPI_DTCERA_IRQ3	IRQ3 をDTC起動要因有効に設定
RAPI_DTCERA_IRQ4	IRQ4 をDTC起動要因有効に設定
RAPI_DTCERA_IRQ5	IRQ5 をDTC起動要因有効に設定
RAPI_DTCERA_IRQ6	IRQ6 をDTC起動要因有効に設定
RAPI_DTCERA_IRQ7	IRQ7 をDTC起動要因有効に設定
RAPI_DTCERA_IADEND_1	IADEND_1 をDTC起動要因有効に設定
RAPI_DTCERA_IADCMP_1	IADCMP_1 をDTC起動要因有効に設定
RAPI_DTCERA_IADEND_2	IADEND_2 をDTC起動要因有効に設定(*1)
RAPI_DTCERA_IADCMP_2	IADCMP_2 をDTC起動要因有効に設定(*1)
RAPI_DTCERA_ELC1FP	ELC1FP をDTC起動要因有効に設定
RAPI_DTCERA_ELC2FP	ELC2FP をDTC起動要因有効に設定
RAPI_DTCERA_SCI3_1_RXI	SCI3_1_RXI をDTC起動要因有効に設定
RAPI_DTCERA_SCI3_1_TXI	SCI3_1_TXI をDTC起動要因有効に設定
RAPI_DTCERA_SCI3_2_RXI	SCI3_2_RXI をDTC起動要因有効に設定
RAPI_DTCERA_SCI3_2_TXI	SCI3_2_TXI をDTC起動要因有効に設定
RAPI_DTCERA_SCI3_3_RXI	SCI3_3_RXI をDTC起動要因有効に設定
RAPI_DTCERA_SCI3_3_TXI	SCI3_3_TXI をDTC起動要因有効に設定
RAPI_DTCERA_IIC2_SSU_RXI	IIC2_SSU_RXI をDTC起動要因有効に設定
RAPI_DTCERA_IIC2_SSU_TXI	IIC2_SSU_TXI をDTC起動要因有効に設定
RAPI_DTCERA_ITCMA	ITCMA をDTC起動要因有効に設定(*2)
RAPI_DTCERA_ITCMB	ITCMB をDTC起動要因有効に設定(*2)
RAPI_DTCERA_ITCMC	ITCMC をDTC起動要因有効に設定(*2)
RAPI_DTCERA_ITCMD	ITCMD をDTC起動要因有効に設定(*2)
RAPI_DTCERA_ITDMA0_0	ITDMA0_0 をDTC起動要因有効に設定
RAPI_DTCERA_ITDMB0_0	ITDMB0_0 をDTC起動要因有効に設定
RAPI_DTCERA_ITDMC0_0	ITDMC0_0 をDTC起動要因有効に設定
RAPI_DTCERA_ITDMD0_0	ITDMD0_0 をDTC起動要因有効に設定
RAPI_DTCERA_ITDMA0_1	ITDMA0_1 をDTC起動要因有効に設定

RAPI_DTCERA_ITDMB0_1	ITDMB0_1 を DTC 起動要因有効に設定
RAPI_DTCERA_ITDMC0_1	ITDMC0_1 を DTC 起動要因有効に設定
RAPI_DTCERA_ITDMD0_1	ITDMD0_1 を DTC 起動要因有効に設定
RAPI_DTCERA_ITDMA1_2	ITDMA1_2 を DTC 起動要因有効に設定(*3)
RAPI_DTCERA_ITDMB1_2	ITDMB1_2 を DTC 起動要因有効に設定(*3)
RAPI_DTCERA_ITDMC1_2	ITDMC1_2 を DTC 起動要因有効に設定(*3)
RAPI_DTCERA_ITDMD1_2	ITDMD1_2 を DTC 起動要因有効に設定(*3)
RAPI_DTCERA_ITDMA1_3	ITDMA1_3 を DTC 起動要因有効に設定(*3)
RAPI_DTCERA_ITDMB1_3	ITDMB1_3 を DTC 起動要因有効に設定(*3)
RAPI_DTCERA_ITDMC1_3	ITDMC1_3 を DTC 起動要因有効に設定(*3)
RAPI_DTCERA_ITDMD1_3	ITDMD1_3 を DTC 起動要因有効に設定(*3)
RAPI_DTCERA_ITESC	ITESC を DTC 起動要因有効に設定
RAPI_DTCERA_ITEMI	ITEMI を DTC 起動要因有効に設定
RAPI_DTCERA_ITEHR	ITEHR を DTC 起動要因有効に設定
RAPI_DTCERA_ITEDY	ITEDY を DTC 起動要因有効に設定
RAPI_DTCERA_ITEWK	ITEWK を DTC 起動要因有効に設定
RAPI_DTCERA_ITGMA	ITGMA を DTC 起動要因有効に設定
RAPI_DTCERA_ITGMB	ITGMB を DTC 起動要因有効に設定
RAPI_SWDTC	

戻り値　常に RAPI_TRUE を返します。

機能　データトランスファコントローラ

備考
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- *印のモジュールに関しては製品によりサポート内容が異なります。

  *1 H8S/20223 グループのみ搭載しています。

  *2 H8S/20103 グループのみ搭載しています。

  *3 H8S/20103 グループにはありません。

プログラム例

```
#include " rapi_dtc_h8s_20203.h"

void main(void)
{
          unsigned long MRA_MRB;
          Boolean RESULT;

          MRA_MRB = (RAPI_DTC_MRA_SAR_INC | RAPI_DTC_MRA_DAR_INC | RAPI_DTC_MRA_BLOCK |
RAPI_DTC_MRA_SEL_SAR | RAPI_DTC_MRA_BYTE_SIZE |
RAPI_DTC_MRB_NO_CHAIN | RAPI_DTC_MRB_COMPLETE_IRQ | RAPI_DTC_MRB_SEQ_CHAIN);

          RESULT = __CreateDTC( 0xfff000, MRA_MRB, 0x00003000, 0x00004000, 0x0000, 0x0001);
          /*IRQ0 を DTC 起動要因有効に設定*/
          RESULT = __EnableDTC(RAPI_DTCERA_IRQ0);
}
```

__DisableDTC

要約

<DTC 起動の無効化>

Boolean __DisableDTC(unsigned long signal)

signal	DTC 起動要因

解説

指定された DTC 起動要因に対応する DTC イネーブルレジスタの DTCE ビットを無効に設定します。

[signal]

各要因に対応する定義値は以下の通りです。

(H8S)

RAPI_DTCERA_IRQ0	IRQ0 を DTC 起動要因無効に設定
RAPI_DTCERA_IRQ1	IRQ1 を DTC 起動要因無効に設定
RAPI_DTCERA_IRQ2	IRQ2 を DTC 起動要因無効に設定
RAPI_DTCERA_IRQ3	IRQ3 を DTC 起動要因無効に設定
RAPI_DTCERA_IRQ4	IRQ4 を DTC 起動要因無効に設定
RAPI_DTCERA_IRQ5	IRQ5 を DTC 起動要因無効に設定
RAPI_DTCERA_IRQ6	IRQ6 を DTC 起動要因無効に設定
RAPI_DTCERA_IRQ7	IRQ7 を DTC 起動要因無効に設定
RAPI_DTCERA_IADEND_1	IADEND_1 を DTC 起動要因無効に設定
RAPI_DTCERA_IADCMP_1	IADCMP_1 を DTC 起動要因無効に設定
RAPI_DTCERA_IADEND_2	IADEND_2 を DTC 起動要因無効に設定(*1)
RAPI_DTCERA_IADCMP_2	IADCMP_2 を DTC 起動要因無効に設定(*1)
RAPI_DTCERA_ELC1FP	ELC1FP を DTC 起動要因無効に設定
RAPI_DTCERA_ELC2FP	ELC2FP を DTC 起動要因無効に設定
RAPI_DTCERA_SCI3_1_RXI	SCI3_1_RXI を DTC 起動要因無効に設定
RAPI_DTCERA_SCI3_1_TXI	SCI3_1_TXI を DTC 起動要因無効に設定
RAPI_DTCERA_SCI3_2_RXI	SCI3_2_RXI を DTC 起動要因無効に設定
RAPI_DTCERA_SCI3_2_TXI	SCI3_2_TXI を DTC 起動要因無効に設定
RAPI_DTCERA_SCI3_3_RXI	SCI3_3_RXI を DTC 起動要因無効に設定
RAPI_DTCERA_SCI3_3_TXI	SCI3_3_TXI を DTC 起動要因無効に設定
RAPI_DTCERA_IIC2_SSU_RXI	IIC2_SSU_RXI を DTC 起動要因無効に設定
RAPI_DTCERA_IIC2_SSU_TXI	IIC2_SSU_TXI を DTC 起動要因無効に設定
RAPI_DTCERA_ITCMA	ITCMA を DTC 起動要因無効に設定(*2)
RAPI_DTCERA_ITCMB	ITCMB を DTC 起動要因無効に設定(*2)
RAPI_DTCERA_ITCMC	ITCMC を DTC 起動要因無効に設定(*2)
RAPI_DTCERA_ITCMD	ITCMD を DTC 起動要因無効に設定(*2)
RAPI_DTCERA_ITDMA0_0	ITDMA0_0 を DTC 起動要因無効に設定
RAPI_DTCERA_ITDMB0_0	ITDMB0_0 を DTC 起動要因無効に設定
RAPI_DTCERA_ITDMC0_0	ITDMC0_0 を DTC 起動要因無効に設定
RAPI_DTCERA_ITDMD0_0	ITDMD0_0 を DTC 起動要因無効に設定
RAPI_DTCERA_ITDMA0_1	ITDMA0_1 を DTC 起動要因無効に設定
RAPI_DTCERA_ITDMB0_1	ITDMB0_1 を DTC 起動要因無効に設定

RAPI_DTCERA_ITDMC0_1	ITDMC0_1 を DTC 起動要因無効に設定
RAPI_DTCERA_ITDMD0_1	ITDMD0_1 を DTC 起動要因無効に設定
RAPI_DTCERA_ITDMA1_2	ITDMA1_2 を DTC 起動要因無効に設定(*3)
RAPI_DTCERA_ITDMB1_2	ITDMB1_2 を DTC 起動要因無効に設定(*3)
RAPI_DTCERA_ITDMC1_2	ITDMC1_2 を DTC 起動要因無効に設定(*3)
RAPI_DTCERA_ITDMD1_2	ITDMD1_2 を DTC 起動要因無効に設定(*3)
RAPI_DTCERA_ITDMA1_3	ITDMA1_3 を DTC 起動要因無効に設定(*3)
RAPI_DTCERA_ITDMB1_3	ITDMB1_3 を DTC 起動要因無効に設定(*3)
RAPI_DTCERA_ITDMC1_3	ITDMC1_3 を DTC 起動要因無効に設定(*3)
RAPI_DTCERA_ITDMD1_3	ITDMD1_3 を DTC 起動要因無効に設定(*3)
RAPI_DTCERA_ITESC	ITESC を DTC 起動要因無効に設定
RAPI_DTCERA_ITEMI	ITEMI を DTC 起動要因無効に設定
RAPI_DTCERA_ITEHR	ITEHR を DTC 起動要因無効に設定
RAPI_DTCERA_ITEDY	ITEDY を DTC 起動要因無効に設定
RAPI_DTCERA_ITEWK	ITEWK を DTC 起動要因無効に設定
RAPI_DTCERA_ITGMA	ITGMA を DTC 起動要因無効に設定
RAPI_DTCERA_ITGMB	ITGMB を DTC 起動要因無効に設定
RAPI_SWDTC	DTVECR の SWDTE ビットを 0 クリアし、ソフトウェアで DTC を起動させないに設定

戻り値　常に RAPI_TRUE を返します。

機能　データトランスファコントローラ

備考
- 第 1 引数に定義値以外の値を指定した場合の動作は保証されません。
- *印のモジュールに関しては製品によりサポート内容が異なります。

  *1 H8S/20223 グループのみ搭載しています。

  *2 H8S/20103 グループのみ搭載しています。

  *3 H8S/20103 グループにはありません。

プログラム例

```
#include " rapi_dtc_h8s_20203.h"

void main(void)
{
          unsigned long MRA_MRB;
          Boolean RESULT;
          MRA_MRB = (RAPI_DTC_MRA_SAR_INC | RAPI_DTC_MRA_DAR_INC | RAPI_DTC_MRA_BLOCK |
RAPI_DTC_MRA_SEL_SAR     |     RAPI_DTC_MRA_BYTE_SIZE     |     RAPI_DTC_MRB_NO_CHAIN     |
RAPI_DTC_MRB_COMPLETE_IRQ | RAPI_DTC_MRB_SEQ_CHAIN);
          RESULT = __CreateDTC( 0xfff000, MRA_MRB, 0x00003000, 0x00004000, 0x0000, 0x0001);
          RESULT = __EnableDTC(RAPI_DTCERA_IRQ0);
          /*IRQ0 を DTC 起動要因無効に設定*/
          RESULT = __DisableDTC(RAPI_DTCERA_IRQ0);
}
```

__EnableSWDTC

要約

< DTC ソフトウェア起動の設定 >

Boolean __EnableSWDTC(unsigned long vect, VoidFuncNotify *func)

vect	ソフトウェアによる DTC 起動ベクタ番号
*func	コールバック関数ポインタ (コールバック関数を設定しない場合は 0 を指定する)

解説

DTC ソフトウェア起動の設定をします。

[vect]

DTC 起動ベクタ番号 (VOFR=H' 0000 の場合　H' 400+(ベクタ番号*2))

[*func]

func に登録する関数は、DTC ドライバに対してユーザが提供しなければなりません。

DTC ドライバは、ソフトウェア起動データ転送終了割り込み発生時、func に登録された関数を呼び出します。

DTC ドライバは、送受信の状態を引数によりユーザに知らせます。

関数の宣言は次の通りです。

  void “任意の関数名”(unsigned char notify);

引数の内容は次の通りです。

  DTC 起動ベクタ番号 (DTVECR の値)

戻り値

SWDTE ビットが 1 の時 RAPI_FALSE を、SWDTE ビットが 0 の時 RAPI_TRUE を返します。

機能

データトランスファコントローラ

備考

・ 第 1 引数、第 2 引数に定義値以外の値を指定した場合の動作は保証されません。

・ DTC ソフトウェア起動ベクタの設定及び DTC 起動は同時に処理（ライト）されます。

プログラム例

```
#include " rapi_dtc_h8s_20203.h"

void main(void)
{
          unsigned long MRA_MRB;
          Boolean RESULT;
          MRA_MRB  =  (RAPI_DTC_MRA_SAR_INC  |  RAPI_DTC_MRA_DAR_INC  |  RAPI_DTC_MRA_BLOCK  |
RAPI_DTC_MRA_SEL_SAR     |     RAPI_DTC_MRA_BYTE_SIZE     |     RAPI_DTC_MRB_NO_CHAIN     |
RAPI_DTC_MRB_COMPLETE_IRQ | RAPI_DTC_MRB_SEQ_CHAIN);
          RESULT = __CreateDTC( 0xfff000, MRA_MRB, 0x00003000, 0x00004000, 0x0000, 0x0001);
          /*起動ベクタ H' 400 + (H' 10 * 2) = H' 420  コールバック関数なし*/
          RESULT = __EnableSWDTC( 0x00000010, 0 );
}
```

Renesas Embedded Application Programming Interface
リファレンスマニュアル

発行年月日　2009年5月29日　　Rev.1.12

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　汎用MCU応用技術部

© 2009. Renesas Technology Corp. and Renesas Solutions Corp., All rights reserved. Printed in Japan.

Renesas Embedded Application Programming Interface
リファレンスマニュアル

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

RJJ10J2520-0100