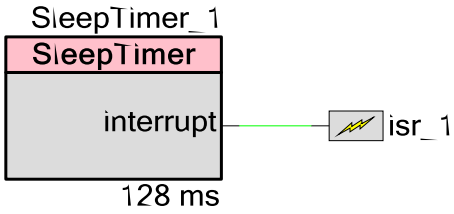
スリープ タイマ

1.60

特徴



- デバイスを低消費電力モードからウェークアップ アクティブとスリープ切り替え
- 構成可能な割り込み発生オプション
- デバイスが動作モード中に定期的に割り込みを生成
- 12 の独立した期間：2、4、8、16、32、64、128、256、512、1024、2048、4096 ms

概要説明

スリープ タイマ コンポーネントは、構成可能な間隔で、アクティブとスリープモードを切り替えることでデバイスをウェイクアップするために使用できます。また構成可能な間隔で、割り込みを発行するように設定することもできます。

スリープ タイマをいつ使用するか

スリープ タイマ コンポーネントは、構成可能な間隔で、アクティブとスリープ切り替えモードからデバイスを定期的にウェイクアップするために使用できます。これには、割り込みが発行されても、されなくとも構いません。また、デバイスが動作モードにあるときに、定期的な割り込みの生成に使用できます。すなわち、カウンタのように動作するということです。

定期的な割り込みはハードウェアカウンタでも実施できます。ただし、これはハードウェアリソースを効率の悪い形で使用することになり、デバイスが動作モードを続けなければなりません。

スリープ タイマは独自のリソースセットを使用するため、設計で 1 つのみ可能です。

割り込み – 出力

スリープ タイマには 1 つの出力接続（割り込み）があり、入力接続はありません。割り込み出力は、セントラル タイムホイール（CTW）割り込みソースを使用します。CTW カウンタが最終カウントに到達したとき、割り込みが発行されます。これはコンポーネントカスタマイザまたは API 関数によって指定されます。

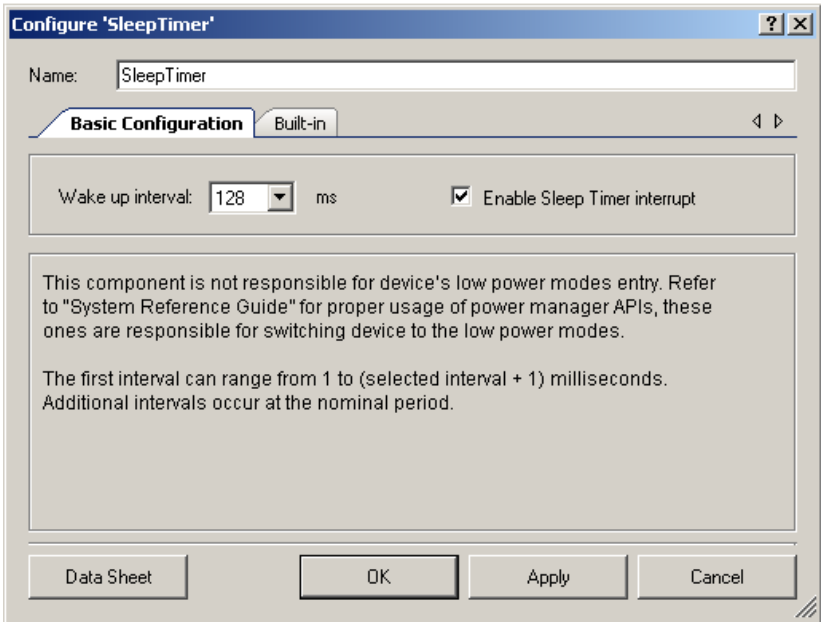
Enable Sleep Timer Interrupt（イネーブルタイマ割り込み）パラメータの選択を外すことにより、シンボル上で出力を隠すことができます。

スケマティックマクロ情報

コンポーネントカタログのデフォルトのスリープ タイマは、デフォルト設定を伴うスリープ タイマ コンポーネントを使用したスケマティックマクロで、デフォルト設定で構成されている割り込みコンポーネントに接続されています。

パラメータおよびセットアップ

スリープタイマ スケマティックマクロを設定上にドラッグし、スリープ タイマ コンポーネントをダブルクリックすると、Configure Sleep Timer（構成スリープ タイマ）ダイアログが開きます。



スリープ タイマ コンポーネントには、次のパラメータがあります。

Wake up interval（ウェイクアップの間隔）

スリープ タイマがデバイスをウェイクアップしたり、（構成されている場合は）割り込みを生成したりする間隔を定義します。次のような独立した間隔のみが受付けられます。2、4、8、16、32、64、128、256、512、1024、2048、4096 ms

これらの間隔値は、ILO からの 1-kHz の入力クロックを想定しています。実際には、ILO の周波数が、デバイスデータシートに表記されているように変動するのでその範囲でスリープ タイマ

間隔は、変動します。このパラメータは初期設定を定義します。ソフトウェアは、スリープ タイマが停止したときのみ、この値を再設定できます。

Enable Sleep Timer interrupt

このパラメータは、選択された間隔が過ぎた後に、スリープ タイマ コンポーネントが割り込みを発行するかどうかを定義します。このパラメータは、コンポーネントがデバイスを低消費電力モードからウェイクアップするかどうかには影響しません。

このパラメータは初期設定を定義します。ソフトウェアからこのパラメータの設定を再設定できます。

Clock Select (クロック選択)

スリープ タイマ コンポーネントは CTW を使用し、動作に 1 kHz のクロックを必要とします。このクロックは、内蔵低速発振器によって生成されます。ILO 1 kHz のクロックは、CTW カウンタに直接供給されます。ILO は外部のコンポーネントなしに、また極めて低消費電力でクロックを生成します。

スリープ タイマを開始させる API 関数は、1 kHz のクロックを自動的に有効にし、コンポーネントが停止した後も有効の状態を続けさせます。最初の間隔は、1～（期間＋1）ミリ秒です。追加の間隔が公称期間で発生します。

配置

配置に関する情報はありません。

リソース

スリープ タイマは次のデバイス リソースを使用します。

- 1 kHz の ILO クロックライン
- CTW カウンタ
- CTW カウンタの割り込みライン

モード	デジタル ブロック					API メモリ（バイト）		ピン
	データパス	マクロセル	状態レジスタ	制御レジスタ	Counter7	Flash	RAM	
デフォルト	N/A	N/A	N/A	N/A	N/A	160	1	N/A

アプリケーション プログラミング インタフェース

アプリケーション プログラミング インターフェース (API) ルーチンにより、ソフトウェアを使用してコンポーネントを設定できます。次の表は、各関数へのインターフェースとその説明を示しています。その次のセクションでは、各関数について詳しく説明します。

デフォルトでは、PSoC Creator はインスタンス名「SleepTimer_1」を設計上のコンポーネントの最初のインスタンスに割り当てます。インスタンス名は、識別子の構文ルールに従った固有の値に変更できます。インスタンス名は、すべてのグローバル関数名、変数名、定数名のプリフィックスになります。読みやすいように、下表では「SleepTimer」というインスタンス名を使用しています。

機能

関数	説明
void SleepTimer_Start(void)	スリープ タイマ操作を開始します。
void SleepTimer_Stop(void)	スリープ タイマ操作を停止します。
void SleepTimer_EnableInt(void)	ウェイクアップ時にスリープ タイマ コンポーネントによる割り込み発行を有効にします。
void SleepTimer_DisableInt(void)	ウェイクアップ時にスリープ タイマ コンポーネントによる割り込み発行を無効にします。
void SleepTimer_SetInterval(uint8)	スリープ タイマのウェイクアップの間隔を設定します。
uint8 SleepTimer_GetStatus(void)	パワーマネージャ割り込み状態レジスタ値を返し、このレジスタのすべてのビットをクリアします。
void SleepTimer_Init(void)	カスタマイズされて提供されるデフォルト設定で初期化・復元します。
void SleepTimer_Enable(void)	1 kHz ILOとCTWカウンタを有効にします。

グローバル変数

変数	説明
SleepTimer_initVar	スリープ タイマが初期化したかどうかを示します。変数は、0に初期化され、SleepTimer_Start()が初めて呼び出されたときに1にセットされます。SleepTimer_Start()ルーチンの最初の呼び出し後は、この動作により、コンポーネントは、再初期化なしに再起動できます。 コンポーネントの再初期化が必要な場合、SleepTimer_Start()やSleepTimer_Enable()の前に、関数SleepTimer_Init()を呼び出すことができます。

void SleepTimer_Start(void)

説明： **これが推奨されるコンポーネントの動作を開始する方法です。**SleepTimer_Start()**は変数** initVar **を設定し、関数** SleepTimer_Init()**を呼び出し、次に関数** SleepTimer_Enable()**を呼び出します。**1 kHz ILO **クロックを有効にし、この有効である状態をスリープタイマ コンポーネントが停止した後にも保ち続けます。**

パラメータ： なし

戻り値： なし

副作用： 変数initVarがすでに設定されている場合、この関数は単に関数SleepTimer_Enable()を呼び出します。

void SleepTimer_Stop(void)

説明： スリープ タイマ操作を停止し、ウェイクアップと割り込みを無効にします。CTWカウンタが最終カウントに達しても、デバイスはウェイクアップしません。また割り込みも発行されません。

パラメータ： なし

戻り値： なし

副作用： 1 kHz ILOクロックが有効である状態を、スリープタイマ コンポーネントが停止した後にも保ち続けます。

void SleepTimer_EnableInt (void)

説明： CTW最終カウント割り込みを有効にします。

パラメータ： なし

戻り値： なし

副作用： なし

void SleepTimer_DisableInt (void)

説明： CTW最終カウント割り込みを無効にします。

パラメータ： なし

戻り値： なし

副作用： なし

void SleepTimer_SetInterval (uint8 interval)

説明： CTW間隔期間を設定します。最初の間隔は、1～（期間＋1）ミリ秒です。追加の間隔が公称期間で発生します。間隔値はCTWが無効にされた場合のみ変更できます。CTWを無効にするには、コンポーネントを停止します。

パラメータ： uint8 interval: CTW用の間隔値。

名前	値	公称期間
SleepTimer__CTW_2_MS	4'b0001	2 ms
SleepTimer__CTW_4_MS	4'b0010	4 ms
SleepTimer__CTW_8_MS	4'b0011	8 ms
SleepTimer__CTW_16_MS	4'b0100	16 ms
SleepTimer__CTW_32_MS	4'b0101	32 ms
SleepTimer__CTW_64_MS	4'b0110	64 ms
SleepTimer__CTW_128_MS	4'b0111	128 ms
SleepTimer__CTW_256_MS	4'b1000	256 ms
SleepTimer__CTW_512_MS	4'b1001	512 ms
SleepTimer__CTW_1024_MS	4'b1010	1024 ms
SleepTimer__CTW_2048_MS	4'b1011	2048 ms
SleepTimer__CTW_4096_MS	4'b1100	4096 ms

戻り値： なし

副作用： なし

uint8 SleepTimer_GetStatus (void)

説明： スリープタイマの状態レジスタの状態を返し、保留中の割り込み状態ビットをクリアします。この関数は、ctw_int状態ビットをクリアするために、いつでもウェイクアップ後に呼び出さなければなりません。この関数は、スリープタイマの割り込みを無効または有効にするために呼び出します。

パラメータ： なし

戻り値： 対応するイベントが発生した場合、イベントに対応するビットをセットした8ビット値（uint8）を返します。下記の定数は、この戻り値を持つことができる2つのイベント用のビットマスクを示しています。

定数	説明
SleepTimer_PM_INT_SR_ONEPPSP	oneppsイベントが発生しました。
SleepTimer_PM_INT_SR_CTW	セントラルタイムホイール イベントが発生しました。

副作用： SleepTimerに関連した割り込みで関数SleepTimer_GetStatus()が呼び出されない場合、割り込みはクリアされないので、割り込みから抜けても即座に再度割り込みに入ります。

スリープ タイマが時間切れになると、スリープ期間は機能的には0 msとなります。これは、ctw_int flagが関数GetStatus()によってクリアされるまで、割り込みが呼び出されるためです。

レジスタの読み出し/クリアと同時に割り込みが生成された場合、このビットはセットされた状態を続けます（それによって別の割り込みが発生します）。

パワーマネージャ割り込み状態レジスタのすべての割り込み状態ビットをレポートし、次にクリアします。これらのビットの一部には、このコンポーネントの操作と関係ないものもあります。

この関数は、ctw_int状態ビットをクリアするために、いつもウェイクアップの後に呼び出さなければなりません（スリープタイマの割り込みが無効または有効にされたとき）。CTWイベント発生後は1ms（ILOの1クロックサイクル）以内にSleepTimer_GetStatus()を呼び出さなければなりません。

void SleepTimer_Init (void)

説明： カスタマイザの [Configure] (設定) ダイアログの設定に従って、コンポーネントを初期化または復元します。SleepTimer_Init()を呼び出す必要はありません。それはSleepTimer_Start() APIがこの関数を呼び出すからです。またこの関数はコンポーネント操作を開始するための推奨手段となっています。CTW間隔期間を設定し、（カスタイマザの設定に従って）CTW割り込みを有効または無効にします。

パラメータ： なし

戻り値： なし

副作用： なし

void SleepTimer_Enable(void)

説明： 1 kHz ILOとCTWを起動し、コンポーネント操作を開始します。SleepTimer_Enable()を呼び出す必要はありません。それはSleepTimer_Start() APIが、このコンポーネント操作を開始するための推奨手段であるこの関数を呼び出すためです。

パラメータ： なし

戻り値： なし

副作用： なし

ファームウェア ソースコードの例

PSoC Creator は、[Find Example Project (プロジェクト例を検索)] ダイアログに数多くのプロジェクト例を提供しており、そこには回路図およびコード例が含まれています。コンポーネント固有の例を見るには、[Component Catalog (コンポーネント カタログ)] または回路図に置いたコンポーネント インスタンスからダイアログを開きます。一般例については、[Start Page (スタート ページ)] または **[File (ファイル)]** メニューからダイアログを開きます。必要に応じてダイアログにある [Fileter Options (フィルタのオプション)] を使用し、選択できるプロジェクトのリストを絞り込みます。

詳しくは、PSoC Creator ヘルプの「Find Example Project (プロジェクト例を検索)」を参照してください。

機能説明

スリープ タイマ コンポーネントは、デバイスを低消費電力モードに入れる責任はありません。詳細については、 *システム リファレンス ガイド*の「電源管理 API」セクションを参照してください。このガイドは PSoC Creator のヘルプメニューにあります。

スリープ タイマ コンポーネントは、セントラルタイムホイール（CTW）を使用します。CTW は 1 kHz ILO によってクロックを供給される、独立して実行できる 1 kHz の 13 ビットカウンタです。

CTW とウォッチドッグタイマ（WDT）の関係が理解できるように、デバイス データシートも参照してください。

前述のように、スリープ タイマは次の間隔で設定できます。2、4、8、16、32、64、128、256、512、1024、2048、4096 ms。ただし、スリープタイマのクロックソース、すなわち ILO の周波数はバリエーションを　持っていますので、スリープタイマの間隔に影響を及ぼします。このバリエーションはデバイスデータシートに記載されています。

スリープ タイマの正しい操作では、デバイスがウェイクアップするたび、またスリープ タイマ割り込みが発行されるたびに、関数 SleepTimer_GetStatus を呼び出してください。

DC 電気的特性と AC 電気的特性

以下の値は、期待されるパフォーマンスを示しており、初期のキャラクターライゼーションデータを基にしています。

スリープ タイマ仕様

パラメータ	説明	条件	最小値	典型値	最大値	単位
	スリープ タイマ期間		-45%	---	+100%	

コンポーネントの変更

ここでは、前のバージョンからコンポーネントに加えられた主な変更を示します。

バージョン	変更の説明	変更の理由 / 影響
1.60	タイムホイール構成レジスタ2のクロバリング問題を修正。ソースコードコメントを更新した。	クロバリング問題の可能性を排除し、より明確なコメントを挿入。
	データシートのマイナーな編集と更新	
1.50.a	SleepTimer コンポーネントのバージョン1.50でファームウェアの欠陥を発見。この欠陥には、共有レジスタを上書きする可能性あり。この欠陥は、SleepTimer コンポーネントの新しいバージョンで修正されたため、バージョン1.50は使用しないこと。	
	シリコン改定との整合性をアドバタイズするコンポーネントへの情報を追加	このツールは、コンポーネントが不整合のシリコン上で使用された場合にエラー/警告を発します。その場合、ターゲットデバイスをサポートする改訂版を更新してください。
	データシートのマイナーな編集と更新	
1.50	Keil reentrancyサポートを追加。	Keil社のコンパイラを用いたPSoC 3サポート。複数のコントロールフローから呼び出される関数の機能。
	APIフローを変更。SleepTimer_Start()がカスタマイザ設定に従ってハードウェアを構成。関数SleepTimer_Init()を追加。	すべてのコンポーネントが同じ実行フローを持つこと。コンポーネントのパラメータを変更するには、SleepTimer_Stop()を呼び出し、パラメータを変更する関数を呼び出し、次にSleepTimer_Start()を呼び出すことにより、コンポーネントを再起動する。後でカスタマイザ設定を復元するには、（コンポーネント停止中に）グローバル変数SleepTimer_initVarの値を0にセットし、再起動する。

バージョン	変更の説明	変更の理由 / 影響
	1 kHz ILOクロックをつねに有効に保つために、関数SleepTimer_Start()を再設計。以前は、関数SleepTimer_Init()の中で一度有効にされた。	これにより、コンポーネント操作と1 kHz ILOが停止し、コンポーネントを再開したときの問題可能性を修正。
	コンポーネントのXML記述を追加。	これにより、PSoC Creatorがこのコンポーネント用に新規デバッガツールウィンドウの作成メカニズムを提供することが可能に。
	Microsoft Windows 7用にオートスクロールを最適化。	不要なスクロールバーの表示を避けるため。
1.10	関数SleepTimer_Reset()を削除し、関数SleepTimer_GetStatus()を追加。 デフォルトで、割り込み出力端末が、設計に配置されている割り込みコンポーネントに設定される。	機能性が低かった前バージョンについて、問題修正のための様々な変更を実行。

© Cypress Semiconductor Corporation, 2011. 本文書に記載される情報は、予告なく変更される場合があります。Cypress Semiconductor Corporation (サイプレス セミコンダクタ社) は、サイプレス 製品に組み込まれた回路以外のいかなる回路を使用することに対しても一切の責任を負いません。特許またはその他の権限下で、ライセンスを譲渡または暗示することもありません。サイプレス 製品は、サイプレスとの書面による合意に基づくものでない限り、医療、生命維持、救命、重要な管理、または安全の用途のために使用することを保証するものではなく、また使用することを意図したものでもありません。さらにサイプレスは、誤動作や故障によって使用者に重大な傷害をもたらすことが合理的に予想される、生命維持システムの重要なコンポーネントとしてサイプレス製品を使用することを許可していません。生命維持システムの用途にサイプレス 製品を供することは、製造者がそのような使用におけるあらゆるリスクを負うことを意味し、その結果サイプレスはあらゆる責任を免除されることを意味します。

PSoC® は、サイプレス セミコンダクタ社の登録商標であり、PSoC Creator™ およびプログラマブル System-on-Chip™ は、サイプレス セミコンダクタ社の商標です。本書で言及するその他すべての商標または登録商標は、各社の所有物です。

全てのソース コード (ソフトウェアおよび / またはファームウェア) はサイプレス セミコンダクタ社 (以下「サイプレス」) が所有し、全世界の特許権保護 (米国およびその他の国)、米国の著作権法ならびに国際協定の条項により保護され、かつそれらに従います。サイプレスが本書面によりライセンシーに付与するライセンスは、個人的、非独占的かつ譲渡不能のライセンスであって、適用される契約で指定されたサイプレスの集積回路と併用されるライセンシーの製品のみをサポートするカスタム ソフトウェアおよび / またはカスタムファームウェアを作成する目的に限って、サイプレスのソース コードの派生著作物をコピー、使用、変更そして作成するためのライセンス、ならびにサイプレスのソース コードおよび派生著作物をコンパイルするためのライセンスです。上記で指定された場合を除き、サイプレスの書面による明示的な許可なくして本ソース コードを複製、変更、 変換、コンパイル、または表示することは全て禁止されます。

免責条項: サイプレス は、明示的または黙示的を問わず、本資料に関するいかなる種類の保証も行いません。これには、商品性または特定目的への適合性の黙示的な保証が含まれますが、これに限定されません。サイプレスは、本文書に記載される資料に対して今後予告なく変更を加える権利を留保します。サイプレスは、本文書に記載されるいかなる製品または回路を適用または使用したことによって生ずるいかなる責任も負いません。サイプレスは、誤動作や故障によって使用者に重大な傷害をもたらすことが合理的に予想される生命維持システムの重要なコンポーネントとしてサイプレス製品を使用することを許可していません。生命維持システムの用途にサイプレス製品を供することは、製造者がそのような使用におけるあらゆるリスクを負うことを意味し、その結果サイプレスはあらゆる責任を免除されることを意味します。

ソフトウェアの使用は、適用されるサイプレス ソフトウェア ライセンス契約によって制限され、かつ制約される場合があります。