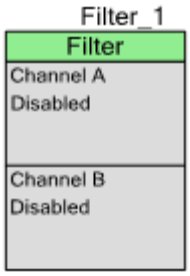
フィルタ
2.0

特長



- 一部の PSoC 3 および PSoC 5 のデバイスで使用可能な Digital Filter Block（DFB）で実行する使いやすいフィルタのユーザ構成。
- 2 つの独立したフィルタ チャンネルをサポートし、それぞれが最大 4 つまでの設計ステージとしてカスケードできます。
- 複数の FIR および IIR（Biquad）フィルタ メソッド（ユーザ係数入力を含む）が優れた柔軟性を実現します。
- 最終係数値は今後の解析用に抽出できます。

概要説明

フィルタ コンポーネントのカスタマイザは、DMA、割り込み、またはデータ フローを管理するポーリングを用いて、Digital Filter Block（DFB）に渡された 1 つまたは 2 つのデータ ストリームにフィルタを構成できます。DFB の 128 のデータおよび係数のロケーションは 2 つのフィルタ チャンネル間で必要に応じて共有されます。カスタマイザは、ユーザ宣言サンプル間隔内でフィルタリングを実行するのに必要な最小バス クロック周波数をレポートします（ただし設定はされていない）。

このコンポーネントは膨大な数のユーザ事例をサポートします。その使用時に何らかの異常に遭遇した場合、psoc_creator_fb@cypress.com に（何が異常を引き起こしたのか適切な説明を添えて）レポートしてください。そうすれば、サイプレスは調査可能となります。

入出力接続

このセクションでは、フィルタのさまざまな入出力接続について説明します。I/O 群リストのアスタリスク (*) は、I/O が、その I/O の説明でリストされている条件において、シンボルに隠されている可能性があることを示します。

割り込み – 出力 *

いずれかのチャンネルが、データ準備イベントに対する割り込み生成用に設定されている場合、割り込み出力がイネーブルになります。ハードウェア信号を ISR コンポーネントに接続して、割り込みルーチンをハンドルします。この端子はチャンネルがデータ準備信号として「割り込み（Interrupt）」を選択している場合にのみ表示されます。端子は両方のチャンネル間で共有されます。

DMA リクエスト – 出力 *

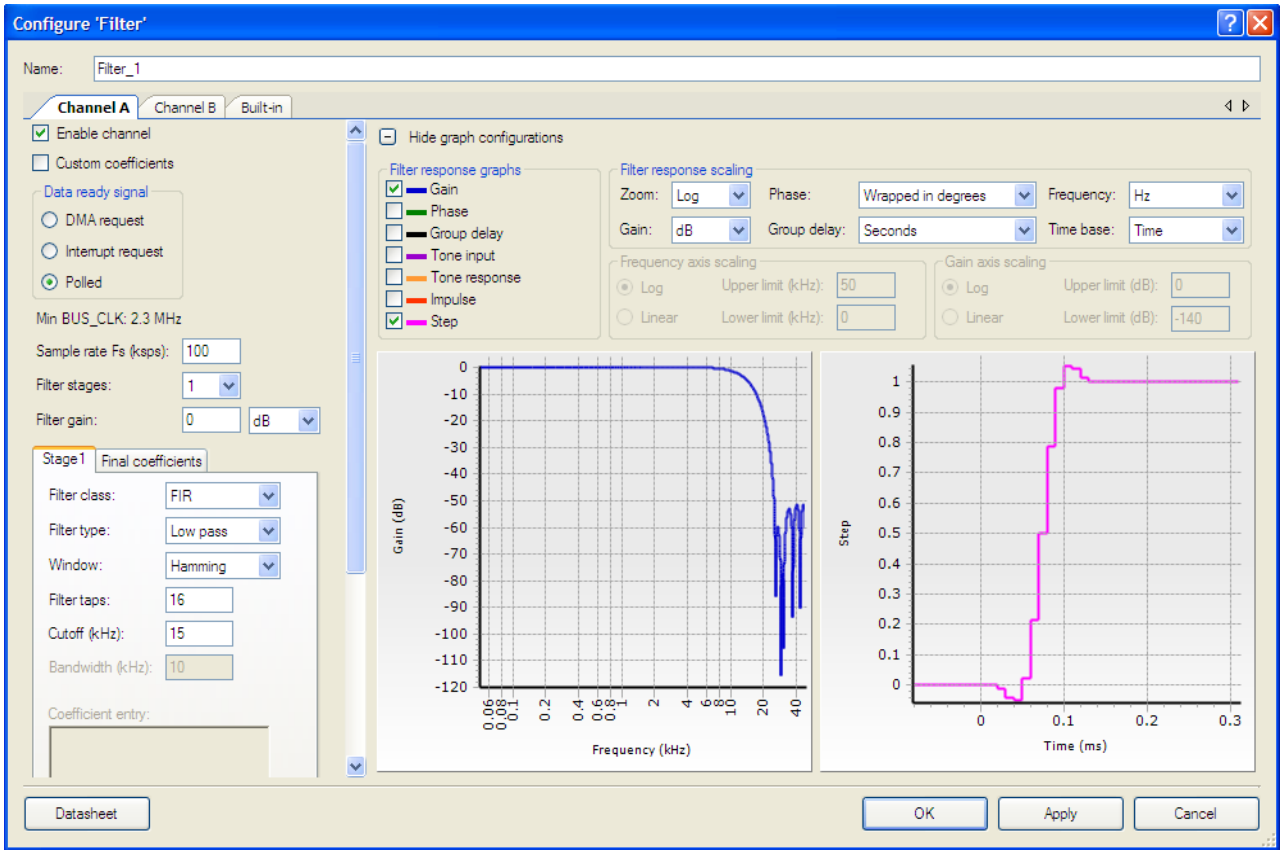
データ準備イベントに対して、どちらかのチャネルを DMA リクエストを発生するように設定した場合、DMA リクエスト出力は *そのチャンネルに対してイネーブルになります。*。各チャンネルには独立した DMA リクエスト出力があります。ハードウェア信号を DMA コンポーネントに接続して、DMA ルーチンをハンドルします。

DMA リクエスト出力信号は読み取りまではハイのままで、その後ローになります。ローカルのバッファがないため、次のサンプルの前に各サンプルが確実に出力から読み込まれるように設計しなければいけません。この期間にサンプルを読み取らないと、次のサンプルのための DMA リクエスト出力が得られません。

コンポーネント・パラメータ

フィルタ コンポーネントをデザイン上にドラッグし、ダブルクリックして **Configure** ダイアログを開きます。**Name** フィールドを変更してフィルタのインスタンス名を調整します。

注 デザインには 1 つのフィルタ コンポーネント（または DFB を使用する他のコンポーネント）だけを配置してください。2 つのコンポーネントの配置を避ける方法はありません。カスタマイザは別々に 2 つのコンポーネントで動作しますが、2 つ以上のコンポーネントがプロジェクトのビルド時に回路図に残されていると、デザインはビルドされず、正しく動作しません。



フィルタ・パラメータ

以下の 2 つのパラメータのグループがあります :

- フィルタ・パラメータは左側にあり、フィルタの応答を決定します。
- パラメータは右上に表示され、どのフィルタのレスポンス グラフを表示するかとフィルタのスケール方法について決定します。

Enable channel

このパラメータはチャンネルのコード生成をイネーブルまたはディスエーブルにします。各チャンネルはそのチャンネルに十分なリソースがある場合に限り、イネーブルにすることができます。構成の変更をサポート可能なリソースが不足している場合、警告アイコンが表示されます。このエラー状態の追跡は非常に複雑なため、異なる設定間を何度も行ったり来たりしてクリックし、すべてのエラー・フラグの消去が必要になることがあります。また、エラー・フラグが表示されないことは、エラーがないことを保証するものではありません。

Custom coefficients

このパラメータはチャンネルのカスタム係数エントリをイネーブルまたはディスエーブルにします。このパラメータがイネーブルの場合、カスタム係数を **Coefficient entry** テキスト ボックスに入力できます。これらのカスタム係数は、フィルタ応答の計算および DFB マイクロコードの生成に使用されます。フィルタ 2.0 では、カスタム設定はフィルタの全ステージに適用されます。つまり、コンポーネント自身で設計されたステージを用いてカスタム係数を使用するステージを混在させることができなくなります。このパラメータがディスエーブルの場合、カスタマイザは要件からフィルタ係数を計算し、**Coefficient entry** ボックスは空白になります。この設計プロセスから得られる最終係数は、そのチャンネルの 4 つのステージすべての集まりであり、**Final coefficients** タブのボックスに表示されます。これらの係数はクリップボードへ完全にコピーでき、 以降の処理のテキスト ファイルまたはスプレッドシートに貼り付けることができます。カスタム係数モードの係数エントリ ボックスに戻って貼り付けできますから、内部デザイン フィルタがカスタム・デザインで使用できるようになります。

このパラメータがイネーブルの場合、**Filter stages** パラメータは、**Filter class** および **Filter taps** がディスエーブルである場合を除き、使用不可となります。このパラメータが使用不可にすると、**Filter stage** パラメータが再度イネーブルになります。カスタム FIR モードでは、カスタマイザはタップ値(ゼロも可)を新しい行の各数値入力とみなし、タップの総数をカウントします。カスタマイザは、カスタムバイクアッド モードで、エントリ数が 5 倍になることを予測し、これが問題とならない場合以外はエラーをレポートします。またバイクアッドの分母の係数の安定性も確認し、ひとつでもバイクアッドが不安定の場合、係数を受け入れません。

データ準備信号

このブロックは、データ準備時に各チャンネルに特定された DMA リクエスト信号仕様や両チャンネル間でシェアする割り込みリクエスト信号を設定します。またステータス レジスタをポーリングして DFB 出力の新規データをチェックすることもできます。ただし、割り込みはここでポーリングできるようにするため、DFB 処理を開始する前に INT_CTRL レジスタでイネーブルにする必要があります。

INT_CTRL および SR（ステータス レジスタ）レジスタの詳細については、レジスタ セクションを参照してください。出力保持レジスタには二重のバッファがありますが、それが上書きされる前に出力からデータを取得することを忘れないでください。

実際に高速なサンプリング・モードでフィルタを実行している場合、DMA はこれを維持するために十分な速度を有する唯一のメソッドと思われます。なぜなら、DFB は、メイン CPU の処理を含まず、命令セットへの DSP 命令拡張をしたマイクロコントローラよりもはるかに高速なサンプリングレートで、複雑なフィルタを実現できるからです。

サンプリング速度

速度は **Sample rate** ボックス（常に ksps 単位）の中に入力され、ハードウェアの処理には影響しませんが、設計プロセスで結果を表示に使用されるフィルタ係数とグラフのスケーリングのみに影響します。所望の物理サンプリング速度で DFB からデータを取り出すのは設計者の責任になります。カスタマイザは公称サンプリング速度を 1sps から 10Msps の範囲に制限します。これは単にレポートの便宜性のためです。なぜなら PSoC 3 または PSoC 5 デバイスの最大 bus_clk 速度は 80MHz を超えないからです。

フィルタ 2.0 にはデシメーションやインターポレーションがないため、サンプリング速度はマルチステージ フィルタの各ステージと同じになります。2 つのチャンネルのサンプリング速度は同じである必要はありませんが、フィルタ 2.0 の DFB ファームウェアは、2 つの速度が同じか、ほぼ同等の場合、最も効率良く動作します。

フィルタ設計入力の設定後、フィルタ 2.0 は DFB が実行する必要のある最低の PSoC 3/PSoC 5 bus_clk 周波数を計算し、すべてのサンプルが処理されていることを確認します。フィルタは、この値がシステムの最高実行可能クロック周波数を超える場合、実装できません。バッファは提供されていません。これはデジタルフィルタ ブロックコードがより高速なチャンネルの着信サンプル期間に、両フィルタを実行できる必要のあることを意味しています。

システムの最高実行可能 bus_clk 速度、そして 2 つのチャンネルのそれぞれに必要となる係数およびフィルタ ポールのカウントが分かっている場合、フィルタが実行する最大サンプリング速度を計算できます。両方のチャンネルを使用している場合、これはより高速なサンプリング速度で実行するチャンネル（または同じ速度で実行されている場合は両方のチャンネル）での速度になります。可能な最高サンプル レートは、両方のフィルタで要求される DFB サイクル数で使える最高の bus_clk を割ることにより計算されます。そのため、2 つのチャンネルのそれぞれは、以下のようにサイクル カウントを計算します。

FIR のみ : 10 + number_of_taps

バイクアッドの偶数の全オーダーのみ : 13 + 5 × オーダー

バイクアッドの奇数の全オーダーのみ : 18 + 5 × オーダー

バイクアッドの偶数全オーダーおよび FIR の両方 : 20 + number_of_taps + 5 × オーダー

バイクアッドの奇数全オーダーおよび FIR の両方 : 25 + number_of_taps + 5 × オーダー

2 つのチャンネルの計算結果を加算。これは両チャンネルの実行に必要な bus_clk cycles の総数です。これを使用予定の bus_clk 周波数に分割すると、2 つのうち速いほうのサンプリングレートが得られます。

フィルタ・ステージ

各チャンネルには最大 4 つまでのカスケード フィルタ ステージを設定することができます。その数は **Filter stage** のドロップダウン リストから選択されます。適切なリソースが不足して、別のステージの追加をサポートできない場合、警告が表示されます。フィルタ レスポンス グラフは新規パラメータが入力されるたびに全カスケードの動作を表示するために更新されます。各ステージには独自のタブを介してアクセスできます。

各フィルタ・ステージのパラメータを個別に設定できます。従って、例えば、FIR のローパス フィルタはバイクアッド ノッチ フィルタとカスケードできます。カスタマイザは入力のオーダーに関係なく、同様の方法で、4 つのステージすべての係数とポールを合計します。すべての FIR ステージは、1 つの FIR フィルタ用に畳み込み積分され、すべてのバイクアッド ステージがカスケードで実装されます。非常に高度なダイナミック レンジ管理アルゴリズムは全ステージのシーケンスを調べ、オーダーと内部ゲインを調整して、そのフィルタから最高品質の信号処理を生み出します。

Filter gain

このパラメータにより、結果として生成されるフィルタ処理済み信号の DC ゲインを提供します。このパラメータはリニアとデシベルの両方のゲイン値をとります。リニアのゲインは 0.01 から 100 の範囲内にする必要があり、dB ゲインの対応制限は −40 dB から 40 dB です。ゲインをそれらの制限外に設定しようとすると、カスタマイザはエラーのフラグをたてます。必要ならば、リニアのゲインに負の数字を入力して応答を反転させることができます。この符号は dB 表示には反映されません。

出力結果のオーバーフローがないことを保証するには、ユニティより小さいゲインのフィルタを実装するとよいことに着目してください。なぜならほとんどのフィルタ応答は、信号の時間領域ではオーバーシュートを示すからです。デフォルトのゲインはユニティに設定されますが、サイプレスは入力信号がデジタル フルスケール に達する典型的な設計では、0.25x (− 12.04 dB) のゲインを選択することをお勧めします。カスタマイザのフィルタ設計ルーチンは、このゲインが選択されている場合、フィルタの内部ステージは、どのような実現可能なデジタル入力信号にもオーバーロードしません。

「Final coefficients（最終係数）」タブ

フィルタ・ステージのセットアップ用タブに加えて、**Final coefficients** タブ内に、フィルタで使用される最終係数を表示するウィンドウがあります。これはカスタマイザが性能を最適化するために実行するすべての係数の端数の切り捨て、ゲインのスケーリング、およびバイクアッドの再オーダリングを含みます。このウィンドウを右クリックして内容を選択しコピーできます。次にそれらを以降の解析用にスプレッドシートへ貼り付けることができます。フィルタ・ステージ：係数入力ウィンドウ セクションでは、FIR および IIR の係数の入出力、両方に使用されるフォーマットについて説明します。

フィルタ・ステージ：フィルタ クラス

選択可能なオプションは **FIR** および **Biquad** です。バイクアッド フィルタは 2 次フィルタ セクションの一連のカスケードとして実装されています。奇数次オーダー伝達関数の 1 次のセクションは z^{-2} のゼロの係数で 2 次のバイクアッドとして実装されます。

フィルタ・ステージ:フィルタ・タイプ(FIR)

FIR フィルタの場合、利用可能なオプションは **ローパス** 、**ハイパス**、**バンドパス**、**バンドストップ**、**Sinc^4**、および **Hilbert** フィルタです。

sinc^4 フィルタは緩やかな立ち上がりの通過帯域を持つ専用のローパス フィルタです。4 次の sinc デシメータ(16 ビット以下の分解能で実行時、PSoC 3 および PSoC 5 の ADC など)を使用するデルタシグマ ADC のドローピング周波数応答を補償するために設計されています。

ヒルベルト フィルタは、さらに 90 度の位相シフトを有するハイパス フィルタの特殊ケースです。それは、いくつかの通信信号処理に使用されます。

バイクアッド フィルタの場合、利用可能なオプションは **ローパス**、**ハイパス**、**バンドパス**、 および バンドストップ です。

フィルタ・ステージ:ウィンドウ(FIR)

FIR フィルタ使用時に提供されるウィンドウ メソッドはいくつかあります。それらの違いをニーズに対してバランスをとる必要があります。パスバンド リップル、トランジション帯域幅、およびストップバンド アッテネーション、ウィンドウ メソッドそれぞれによって異なった影響を受けます。

- **長方形**:このメソッドはウィンドウのない代表例であり、理想的なローパス フィルタの sinc インパルス応答は使用するタップ数を越えるとゼロに切り捨てられます。これらのフィルタは大きな通過帯域リップル、鋭いロールオフ、および貧弱なストップバンド アッテネーションを示します。このウィンドウは、ギブス現象に起因する大きなリップル効果があるため、まれにしか使用されません。
- **ハミング**:フィルタ 2.0 で、使用されるウィンドウは実際には少し修正を加えた Albrecht によるハミング ウィンドウです。それは、ある程度の平坦性とより均一な阻止帯域を示します。これは優れた汎用 FIR フィルタであり、新たに配置されたコンポーネントはデフォルトの選択となります。
- **ブラックマン**:修正 ブラックマン ウィンドウ(Blackman-Nuttal ウィンドウに近く、これも Albrecht による) これはウィンドウの ハミング クラスより大きなストップバンド アッテネーションを提供しますが、トランジション帯域はより広くなります。

フィルタ・ステージ:フィルタ・タップ(FIR)

このエントリボックスのバージョンは FIR クラスのフィルタ・ステージが利用できます。FIR フィルタのみを使用する場合、全フィルタの総結合サイズは最大 128 タップまで可能です。FIR フィルタの次数はタップの数よりも 1 少なくなります。

フィルタ・ステージ:シェイプ(バイクアッド)

フィルタ 2.0 でバイクアッド フィルタ クラスを使用時、シェイプは バタワース、ベッセル、および チェビシェフ となります。フィルタ 2.0 のベッセル実装は古典的な線形バージョンのバイリニア変換を使用します。これはサンプリングレートのかなりの割合を占めるカットオフ周波数に対するフラット群遅延のビヘイビアをプリザーブできません。

フィルタ・ステージ:オーダー(バイクアッド)

エントリボックスの、このバージョンはバイクアッド クラスのフィルタ ステージ用です。ローパスおよびハイパス フィルタは偶数または奇数のいずれかになりますが、1 ポールが DFB 内に同じバイクアッド トポロジで実装されるため、実装は常に最も近い偶数で端数を切り捨てられます。バンドパスおよびバンドストップ フィルタは偶数の次数だけが可能で、この場合に奇数を入力するとエラーが表示されます。チェビシェフ および バタワース フィルタは最高 50 の次数が可能です。非常に高いオーダー、および狭いバンドパスまたはバンドストップの帯域幅では、カスタマイザの数値制限は予期しない結果を生み出すことがあります。

ベッセル ローパスおよびハイパス フィルタは最高 25 オーダーまで設計可能です。

バイクアッド カスケード フィルタのオーダー N で必要とされるメモリロケーションの数は N が偶数の場合は 2.5N、また N が奇数の場合は 2.5(N + 1) となります。バイクアッド フィルタは追加変数の 3 つの内部メモリ ロケーションを必要とするため、バイクアッドまたはバイクアッド+FIR 使用時は、すべてのフィルタの総結合サイズは 125 メモリ ロケーションを超えることはできません。

フィルタ・ステージ:カットオフ

Sinc^4、ロー パス、ハイ パス、および ヒルベルト フィルタの通過帯域周波数のエッジを入力します。この周波数でのフィルタの予想ゲインはフィルタのクラスとタイプによって決まります。FIR のローパス、ハイパスおよびヒルベルト フィルタの場合、応答は入力したカットオフ周波数で名目上 −6 dB です。バイクアッド ベッセル および バタワース フィルタの場合、応答は −3 dB です。チェビシェフ フィルタの場合、通過帯域で最も高いゲインについて、応答は入力リップル値に等しくなります。

FIR sinc^4 フィルタのゲインの応答は実験によって最適に評価されます。

フィルタ・ステージ:中心周波数および帯域幅(バンドパスおよびバンドストップ)

バンドパス フィルタの帯域幅は、フィルタ クラスおよびタイプのために以前提供されたカットオフ基準を満たす、上下限周波数の周波数差です。中心周波数は上下限周波数の間にありますが、それらの周波数との正確な関係はフィルタ・クラスおよびタイプで決まります。

FIR クラスのフィルタの場合、バンドパス フィルタまたはバンドストップ フィルタの中心周波数は上下限カットオフ周波数の算術平均です。言い換えれば、顧客によって作成された応答は中心周波数まわりに等差級数的に対称になります。10 kHz の中心周波数および 10 kHz の帯域幅を持つ FIR バンドパス フィルタまたはバンドストップ フィルタは、5 kHz および 15 kHz で −6 dB のポイントです。

フィルタ 2.0 で使用されるバンドパスおよびバンドストップ 変換のタイプは、幾何学的に対称な周波数応答を提供します。バイクアッド クラスのバンドパス フィルタは、上下カットオフ周波数で定義され、それらはユーザが入力した中心周波数まわりに*数値的に* 対称で、FIR の場合と同様に設置されます。その結果、バタワース バイクアッド バンドパスフィルタでは 10 kHz の中心周波数と 10 kHz の帯域幅を入力すると、5 kHz および 15 kHz で −3 dB のポイントを得ることができます。該フィルタの「真」の中心周波数は 10 kHz ではなく、この場合、SQRT(5 kHz × 15 kHz) = 8.66 kHz となります。

バイクアッド クラスのバンドストップ フィルタは、入力した中心周波数で最大の減衰量になるように設計されています。これは帯域通過カットオフ周波数が、FIR の場合と一致するようには設置できないことを意味しています。カットオフ周波数するために、実際には次の小さな計算を行う必要があります。

$$F_{lower} = SQRT(0.25 \times BW^2 + F_{center}^2) - 0.5 \times BW$$

また当然 $F_{upper} = F_{lower} + BW$

10 kHz の中心周波数と 10 kHz の帯域幅の例で、この計算は F_{lower} = 6.18 kHz および F_{upper} = 16.18 kHz となります。

フィルタ・ステージ:リップル(IIR チェビシェフ)

このパラメータはバイクアッド クラスの チェビシェフ フィルタの場合のみ有効です。これはフィルタの理論通過帯域リップルを決定します。許容範囲は 0.00001 dB から 3 dB です。

フィルタ・ステージ:係数入力ウィンドウ

選択したフィルタ ステージにカスタム係数を入力する場合には、このテキスト ボックスを使用します。**Custom Coefficients** チェック ボックスが選択されている必要がありますから、詳細は、Custom coefficients を参照してください。

各行に係数を浮動小数点値で入力してください。空白は無視されますが、ゼロは有効な入力となります。数値入力はフィルタ 2.0 では受付られません。

バイクアッド フィルタの場合、最初に 3 つの分子係数(z^0、z^{-1}、および z^{-2})を入力してください。続いて 2 つの分母係数(z^0 の分母係数がユニティであることが前提)を入力してください。入力した係数がバリデートされても、もし係数が無効である(または総数が 5 で割り切れない)ことがわかったときは、エラーが表示されます。入力した分母係数は安定性もテストされます。もしバイクアッドが不安定であると分かったときは、警告が表示されます。バイクアッド係数は同じゲイン調整およびシーケンス アルゴリズムに適用されますが、これらには内部生成係数が使われるので、実装上のオーダーは入力と同じにならない場合があります。また、通過帯域のフィルタのピーク ゲインは、ユーザ ゲイン ボックスに入力された値になるよう調整されます。任意の分子スケーリング値を入力でき、アルゴリズムはそれらを適切にスケーリングします。フィルタ 2.0 を PID 実装または他の制御ループに使用している場合、独自のゲイン値を計算し、ゲイン・ボックスに入れて、ゲイン全体が所望の値になっていることを確認する必要があります。

FIR フィルタの場合、入力した最初の値は z^0 の係数、遅延のないタップとして取り扱われます。FIR のみのフィルタの場合、セパレート ゲイン調整アルゴリズムは適用されません。FIR の場合は、係数値の最大許容範囲は−1 から 1-2-23 です。範囲外の値の入力は可能ですが、その場合は許容範囲内に入るように数値を落としたフィルタ ゲイン値とする必要があります。このゲイン値は、無効な係数の入力で得られます。

FIR および バイクアッド フィルタが結合されている場合、ゲインのスケーリングが自動的に FIR 部分に適用され、バイクアッドが実行される前に実装されます。**Final coefficients** タブに表示されたカスケード全体をスケーリングした結果を表示できます。そこから、必要に応じて、以降の解析用に別のアプリケーションへ結果を貼り付けることができます。

パラメータの表示

パラメータの表示は、フィルタの応答を構成ウィンドウに表示する方法にのみ影響します。フィルタ設定のコード生成には影響しません。

パラメータ選択のセットアップは、このセクションの上部にあるグラフ構成ボタンで表示または非表示することができます。これによりグラフをより簡単に読み込めるようになります。多数のウィンドウ枠がグラフ領域に表示される場合、プロット軸は自動プロット ルーチンの制限のため、思い通りに番号付けされないことがあります。

プロットは、周波数およびパラメータの 2 つのサブプロット領域に分割されます。それぞれのサブプロットを右クリックすると、ビットマップ形式でクリップボード側をコピーして、自分のレポートに貼り付けることができます。

フィルタ応答グラフ

ゲイン – これがイネーブルの場合、周波数全域のフィルタ応答の振幅を表示します。

位相 – これがイネーブルの場合、周波数全域のフィルタ応答の位相シフトを表示します。

群遅延 – これがイネーブルの場合、周波数全域のフィルタ応答の群遅延を表示します。

トーン入力 – これがイネーブルの場合、フィルタへの入力として使用する、中心での正弦波信号またはフィルタのカットオフ周波数を表示します。複数のフィルタ・ステージが使用されている場合、トーンはアベレージ センター またはカットオフ周波数と等しくなります。フィルタ 2.0 では、この正弦波の周波数は個別に設定できません。

トーン応答 – これがイネーブルの場合、あらかじめ定められたトーン入力波へのフィルタ応答を表示します（トーン入力 を参照）。

インパルス – これがイネーブルの場合、単一サンプルの立上りインパルスへのフィルタ応答を表示します。

ステップ – これがイネーブルの場合、立上り単位ステップ関数へのフィルタ応答を表示します。

フィルタ応答のスケーリング：ズーム

使用可能なオプションは **Log**、**Linear**、および **Custom** です。

対数ズーム オプションは、DC からナイキスト周波数までの対数周波数軸によるフィルタ応答の表示を提供します。

線形オプションは、DC からナイキスト周波数までの通過帯域周波数を線形周波数軸で表示します。

カスタム オプションの選択をイネーブルにすると、ゲインおよび周波数の最大最小制限の入力設定が可能になります。カスタムビューで、周波数およびゲイン軸に独自の制限を定義することができます。このモードで、周波数およびゲイン軸を線形モードと対数モードの両方に設定できます。

フィルタ応答のスケーリング：ゲイン

dB ゲインを選択すると、ゲイン応答に対しデシベルでゲイン値を表示します。**Linear** 表示を設定すると、線形スケールでゲイン値を表示します。

フィルタ応答のスケーリング：位相

このパラメータにより、ラジアンまたは度表現で、ラップまたはアンラップ位相を選択表示できます。使用可能なオプションは **Unwrapped in degrees**、**Wrapped in degrees**、**Unwrapped in radians**、および **Wrapped in radians** です。ゼロを送信するいくつかのフィルタ（例えば、バンドストップ フィルタ）はアンラップ位相を選択した場合でも、不連続な位相応答を表示します。

フィルタ応答のスケーリング：群遅延

このパラメータにより、マイクロ秒での時間またはサンプル数で群遅延応答を選択できるようになります。

フィルタ応答のスケーリング：周波数

このパラメータにより、kHz またはサンプル数の値で x 軸の周波数応答を選択できます。

フィルタ応答のスケーリング：基準時間

このパラメータにより、ミリ秒またはサンプル数の値で x 軸の時間応答を選択できるようになります。

周波数軸のスケーリング：対数/線形選択

このオプションは **Custom** ズームが選択されている場合に限り有効です。**Log** を選択すると、x 軸の周波数応答を対数スケールに設定します。**Linear** を選択すると、x 軸を線形スケールで設定します。

周波数軸のスケーリング：上限

このオプションは **Custom** ズームが選択されている場合に限り有効です。周波数レスポンス グラフの x 軸の上限を設定します。最上限はサンプリング速度の半分以下にする必要があります。

周波数軸のスケーリング：下限

このオプションは **Custom** ズームが選択されている場合に限り有効です。周波数レスポンス グラフの x 軸の下限を設定します。最下限は上限値よりも小さく、かつゼロよりも大きくする必要があります。

ゲイン軸のスケーリング：対数/線形選択

このオプションは **Custom** ズームが選択されている場合に限り有効です。**Log** を選択すると、ゲイン軸を対数スケールに設定します。**Linear** を選択すると、ゲイン軸を線形スケールに設定します。

ゲイン軸のスケーリング：上限

このオプションは **Custom** ズームが選択されている場合に限り有効です。**Log** または **Linear** ボタンを選択したかどうかに基づき、dB または線形スケールのいずれかでゲイン応答の上限を設定します。

ゲイン軸のスケーリング : 下限

このオプションは **Custom** ズームが選択されている場合に限り有効です。**Log** または **Linear** ボタンを選択したかどうかに基づき、dB または線形スケールのいずれかでゲイン応答の下限を設定します。下限は上限値よりも小さくする必要があります。

配置

フィルタ 2.0 コンポーネントは DFB すべてを消費するため、動作中のプロジェクトは配置済みフィルタ(または DFB を必要とする他のコンポーネント)は 1 つだけしか入れることができません。複数のフィルタ コンポーネントを配置する場合、それぞれを独自にカスタマイザでセットアップし、そのプロパティーを保存することができます。これによりプロジェクトのビルド前の最初の回路図に、複数のフィルタのセットアップを既定値として保存しておく方法がとれます。

リソース

Analog Blocks	Digital Blocks					API メモリ(バイト)		ピン(外部入出力ごと)
	データパス	Macro セル	Status Registers	Control Registers	Counter7	フラッシュ	RAM	
該当なし	該当なし	該当なし	該当なし	該当なし	該当なし	3116	69	1

フィルタ 2.0 コンポーネントは DFB 以外のリソーセスは消費しません。最高のスループットを実現するには、データ管理に DMA が必要になると思われます。

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とやりとりできます。次の表に、各関数に対するインタフェースを「include」ファイルによって提供される関連定数とともに示します。続くセクションでは、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「Filter_1」を、特定のプロジェクトにおける最初のコンポーネントのインスタンスに割り当てます。コンポーネントのインスタンス名は、識別子の文法ルールに従ってユニークな名前に変更できます。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「Filter.」というインスタンス名を使用しています。

関数	説明
Filter_Start()	割り込み、DMA およびフィルタ設定に対し、フィルタ・コンポーネントのハードウェアを設定し、イネーブルにします。
Filter_Stop()	フィルタを動作状態から停止し、ハードウェアの電源を切ります。

関数	説明
Filter_Read8()	フィルタの出力保持レジスタの電流値を読み取ります。最上位のバイトのバイトを読み取ります。
Filter_Read16()	フィルタの出力保持レジスタの電流値を読み取ります。最上位のバイトの 2 バイトを読み取ります。
Filter_Read24()	フィルタの出力保持レジスタの電流値を読み取ります。データ出力保持レジスタの 3 バイトを読み取ります。
Filter_Write8()	新しい 8 ビットのサンプルをフィルタの入力ステージング・レジスタに書き込みます。
Filter_Write16()	新しい 16 ビットのサンプルをフィルタの入力ステージング・レジスタに書き込みます。
Filter_Write24()	新しい 24 ビットのサンプルをフィルタの入力ステージング・レジスタに書き込みます。
Filter_ClearInterruptSource()	Filter_ALL_INTR マスクをステータス・レジスタに書き込み、アクティブな割り込みをすべてクリアします。
Filter_IsInterruptChannelA()	チャンネル A がデータ準備割り込みでトリガされたかどうかを識別します。
Filter_IsInterruptChannelB()	チャンネル B がデータ準備割り込みでトリガされたかどうかを識別します。
Filter_Sleep()	動作を停止し、ユーザ構成を保存します。
Filter_Wakeup()	ユーザ構成を復元し、イネーブルにします。
Filter_Init()	初期化、もしくはデフォルトのフィルタ構成を復元します。
Filter_Enable()	フィルタをイネーブルにします。
Filter_SaveConfig()	フィルタのノンリテンション レジスタの構成を保存します。
Filter_RestoreConfig()	フィルタのノンリテンション レジスタの構成を復元します。
Filter_SetCoherency()	コヒーレンス・レジスタのキー・コヒーレンス・バイトを設定します。

グローバル変数

変数	説明
Filter_initVar	フィルタが初期化されたかどうかを表示します。変数は、0 に初期化され、Filter_Start() が初めて呼び出されたときに 1 に設定されます。これにより、コンポーネントは Filter_Start() ルーチンへの最初の呼び出し後、再初期化せずにリスタートできます。 コンポーネントの再初期化が必要な場合、関数 Filter_Start() や Filter_Enable() の前に、関数 Filter_Init() が呼び出されます。

定義

- **Filter_CHANNEL_x** – Filter_CHANNEL_A または Filter_CHANNEL_B. 関数呼び出しで発生する処理をチャンネル指定する際に使用してください。
- **Filter_CHANNEL_x_INTR** – ステータス・レジスタの CHANNEL_A または CHANNEL_B の割り込みをマスクします。
- **Filter_ALL_INTR** – ステータス・レジスタの可能性がある割り込みをマスクします。

void Filter_Start(void)

説明: これは、コンポーネントのオペレーションを開始する際に好ましいメソッドです。割り込み、DMA およびフィルタ設定に対し、フィルタ・コンポーネントのハードウェアを設定し、イネーブルにします。

引数: なし

戻り値: なし

副作用: なし

void Filter_Stop(void)

説明: フィルタ ハードウェアを動作状態から停止し、電源を切ります。

引数: なし

戻り値: なし

副作用: なし

uint8 Filter_Read8(uint8 channel)

説明: チャンネル A またはチャンネル B の出力保持レジスタの最上位バイトを読み取ります。

引数: uint8 channel: どのフィルタ チャンネルを読み取る必要があるか。Filter_CHANNEL_A および Filter_CHANNEL_B はオプションです。

戻り値: 8 ビット フィルタ出力値

副作用: なし

uint16 Filter_Read16(uint8 channel)

説明: チャンネル A またはチャンネル B の出力保持レジスタの最上位 2 バイトを読み取ります。

引数: uint8 channel: どのフィルタ チャンネルを読み取る必要があるか。Filter_CHANNEL_A および Filter_CHANNEL_B はオプションです。

戻り値: 16 ビット フィルタ出力値

副作用: なし

uint32 Filter_Read24(uint8 channel)

説明: チャンネル A またはチャンネル B の出力保持レジスタの 3 バイトすべてを読み取ります。

引数: uint8 channel: どのフィルタ チャンネルを読み取る必要があるのか。Filter_CHANNEL_A および Filter_CHANNEL_B はオプションです。

戻り値: 符号拡張 24 ビット出力値の符号なし 32 ビット表現

副作用: なし

void Filter_Write8(uint8 channel, uint8 sample)

説明: チャンネル A またはチャンネル B の入力ステージング・レジスタの最上位バイトに書き込みます。

引数: uint8 channel: どのフィルタ チャンネルを読み取る必要があるか。Filter_CHANNEL_A および Filter_CHANNEL_B はオプションです。

uint8 sample: 入力レジスタに書き込まれる値

戻り値: なし

副作用: この関数は最上位のバイトのみを書き込みます。これは、最下位バイトがゼロに設定されていなかった場合、入力サンプルに追加されてノイズの原因となることがあります。

void Filter_Write16(uint8 channel, uint16 sample)

説明: チャンネル A またはチャンネル B の入力ステージング・レジスタの最上位 2 バイトを書き込みます。

引数: uint8 channel: どのフィルタ チャンネルを読み取る必要があるか。Filter_CHANNEL_A および Filter_CHANNEL_B はオプションです。

uint16 sample: 入力レジスタに書き込まれる値

戻り値: なし

副作用: なし

void Filter_Write24(uint8 channel, uint32 sample)

説明: チャンネル A またはチャンネル B の入力ステージング・レジスタの 3 バイトすべてに書き込みます。

引数: uint8 channel: どのフィルタ チャンネルを読み取る必要があるか。Filter_CHANNEL_A および Filter_CHANNEL_B はオプションです。

uint32 sample: 入力レジスタに書き込まれる値

戻り値: なし

副作用: なし

void Filter_ClearInterruptSource(void)

説明: Filter_ALL_INTR マスクをステータス・レジスタに書き込み、アクティブな割り込みをすべてクリアします。このマスクの定義については、前の 定義 セクションを参照してください。

引数: なし

戻り値: なし

副作用: なし

uint8 Filter_IsInterruptChannelA(void)

説明: チャンネル A がデータ準備割り込みでトリガされたかどうかを識別します。

引数: なし

戻り値: チャンネルAに割り込み無しの場合は0、その他の場合は正の値。

副作用: なし

uint8 Filter_IsInterruptChannelB(void)

説明: チャンネル B がデータ準備割り込みでトリガされたかどうかを識別します。

引数: なし

戻り値: チャンネル B に割り込み無しの場合は0、その他の場合は正値。

副作用: なし

void Filter_SetCoherency(uint8 channel, unit8 byte_select)

説明: DFB コヒーレンス・レジスタの値を設定します。この値はキー・コヒーレンス・バイトを決定します。キー・コヒーレンス・バイトはフィールドの更新が求められたとき、ソフトウェアが最後にどのバイトが読み書きされたかをハードウェアをうまく伝えるソフトウェアの方法です。

引数: uint8 channel: そのオプションは Filter_Channel_A および Filter_Channel_B です。

uint8 byte_select: どの 3 バイトをキー・コヒーレンス・バイトとして設定するか決定します。オプションは High バイト、Med バイトおよび Low バイトです。

戻り値: なし

副作用: デフォルトでは、High バイトはキー コヒーレンス バイトです。この API を使用してデフォルトの動作を変更して、次に前に述べた Filter_Read() and Filter_Write() API を使用すると、予期しない動作を引き起こすことがあります。

void Filter_Sleep(void)

説明: DFB 処理を停止します。構成レジスタおよびコンポーネントのイネーブル状態を保存します。スリープの入力直前に呼び出しておく必要があります。

引数: なし

戻り値: なし

副作用: フィルタ出力レジスタはノンリテンションであり、スリープしようとしている間は保存されません。そのため、スリープしようとする前に、ペンディングの変換がないことを確認してください。

void Filter_Wakeup(void)

説明: これは、コンポーネントを Filter_Sleep() が呼び出されたときの状態に復元するための推奨 API です。Filter_Wakeup() 関数は、設定を復旧させるために Filter_RestoreConfig() 関数を呼び出します。Filter_Sleep() 関数が呼び出される前にコンポーネントがイネーブルになった場合、Filter_Wakeup() 関数がコンポーネントを再度イネーブルにします。

引数: なし

戻り値: なし

副作用: 最初に Filter _Sleep() または Filter _SaveConfig() 関数を呼び出さずにFilter_Wakeup() 関数を呼び出すと、予期しない動作につながる可能性があります。

void Filter_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。Filter_Start() API が Filter_Init() 関数を呼び出すので、この関数を呼び出す必要はありません。これはコンポーネントのオペレーションを開始する際に好ましいメソッドです。

引数: なし

戻り値: なし

副作用: すべてのレジスタがリセットされ、初期値になります。これによって、コンポーネントが再初期化されます。

void Filter_Enable(void)

説明: ハードウェアの使用を開始し、コンポーネントの動作を開始します。Filter_Start() API が Filter_Enable() 関数を呼び出すので、この関数を呼び出す必要はありません。これはコンポーネントのオペレーションを開始する際に好ましいメソッドです。

引数: なし

戻り値: なし

副作用: なし

void Filter_SaveConfig(void)

説明: この関数は、コンポーネントの設定と保持されないレジスタを保存します。この関数は、[Configure] ダイアログで定義されている、または該当する API で変更される、現在のコンポーネント・パラメータ値も保存します。この関数は、Filter_Sleep() 関数に呼び出されます。

引数: なし

戻り値: なし

副作用: なし

void Counter_RestoreConfig(void)

説明: この関数は、コンポーネントの設定とノンリテンション レジスタを復元します。この関数はまた、コンポーネントのパラメータ値を Filter_Sleep() 関数を呼び出す前の状態に復旧します。

引数: なし

戻り値: なし

副作用: Filter_SaveConfig() または Filter_Sleep() を事前に呼び出さず、この関数を呼び出した場合、予期しない動作を引き起こすことがあります。

ファームウェア・ソースコードのサンプル

PSoC Creator は、[Find Example Project（サンプルプロジェクトを検索）] ダイアログに多数のサンプルプロジェクトを提供しており、そこには回路図およびサンプル コードが含まれています。コンポーネント固有の例を見るには、[Component Catalog] 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] 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 **Filter Options** 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（サンプルプロジェクトを検索）」を参照してください。

機能説明

フィルタ・コンポーネントは DFB のコプロセッサに必要なコードを生成し、フィルタコンポーネントを設定します。マルチステージ フィルタはコンボリューションによって数学的に単一フィルタにまとめあげて、各チャンネルを 1 つの大きなフィルタにします。フィーチャ・モードは各チャンネルからの IIR-FIR ストリームのサポートを含みます。

DMA

DMA コンポーネントはフィルタ出力から RAM へ変換結果を転送するのに使用できます。DMA コンポーネントはフィルタ入力レジスタへ入力サンプル値を転送するのにも使用できます。フィルタ データ READY 信号は DMA のデータ リクエスト信号に接続する必要があります。DMA Wizard は次に示した表のように DMA オプションを設定するのに使用できます。

フィルタ分解能	DMA ウィザードでの DMA ソース/デスティネーションの名前	方向	DMA Req Signal	DMA Req Type	説明
8 ビット	Filter_HOLDBH_PTR	source (ソース)	DMA_Req_B	レベル	8 ビット・フィルタ出力値をフィルタ出力レジスタ Filter_HOLDB から受信します。
	Filter_HOLDAH_PTR	source (ソース)	DMA_Req_A	レベル	8 ビット・フィルタ出力値をフィルタ出力レジスタ Filter_HOLDA から受信します。
	Filter_STAGEAH_PTR	destination (転送先)	DMA_Rea_A または該当なし	Level または該当なし	8 ビット入力サンプル値を RAM からフィルタ入力レジスタ Filter_STAGEA へ受信します。
	Filter_STAGEBH_PTR	destination (転送先)	DMA_Req_B または該当なし	Level または該当なし	8 ビット入力サンプル値を RAM からフィルタ入力レジスタ Filter_STAGEB へ受信します。
16 ビット	Filter_HOLDB_PTR	source (ソース)	DMA_Req_B	レベル	16 ビット・フィルタ出力値をフィルタ出力レジスタ Filter_HOLDB から受信します。

フィルタ分解能	DMA ウィザードでの DMA ソース/デスティネーションの名前	方向	DMA Req Signal	DMA Req Type	説明
	Filter_HOLDA_PTR	source (ソース)	DMA_Req_A	レベル	16 ビット・フィルタ出力値をフィルタ出力レジスタ Filter_HOLDA から受信します。
	Filter_STAGEA_PTR	destination (転送先)	DMA_Rea_A または該当なし	Level または該当なし	16 ビット入力サンプル値を RAM からフィルタ入力レジスタ Filter_STAGEA へ受信します。
	Filter_STAGEB_PTR	destination (転送先)	DMA_Req_B または該当なし	Level または該当なし	16 ビット入力サンプル値を RAM からフィルタ入力レジスタ Filter_STAGEB へ受信します。
24ビット	Filter_HOLDB_PTR	source (ソース)	DMA_Req_B	レベル	24 ビット・フィルタ出力値をフィルタ出力レジスタ Filter_HOLDB から受信します。
	Filter_HOLDA_PTR	source (ソース)	DMA_Req_A	レベル	24 ビット・フィルタ出力値をフィルタ出力レジスタ Filter_HOLDA から受信します。
	Filter_STAGEA_PTR	destination (転送先)	DMA_Rea_A または該当なし	Level または該当なし	24 ビット入力サンプル値を RAM からフィルタ入力レジスタ Filter_STAGEA へ受信します。
	Filter_STAGEB_PTR	destination (転送先)	DMA_Req_B または該当なし	Level または該当なしw	24 ビット入力サンプル値を RAM からフィルタ入力レジスタ Filter_STAGEB へ受信します。

レジスタ

ステージング

フィルタ コンポーネントの 2 つのチャンネルは、それぞれ 24 ビット専用の入力ステージングレジスタを備えています。データ処理を行なわない場合は、フィルタはウェイト状態に入ります。このときフィルタはデザインの新しいパススルーを始めるために、これらのひとつのレジスタへの書き込みを待ちます。

保持

最新の出力サンプルは、入力データ処理後、24 ビット出力保持レジスタに配置されます。出力 サンプル準備のシステム通知については、以下のように 3 つのオプションがあります。割り込み、DMA リクエスト、またはポーリング。

Data Align (Filter_DALIGN) Register

DFB は入力データが MSB 整列となることを必要とし、提供された出力結果も MSB 整列となります。DFB ハードウェアは、入力ステージング レジスタおよび出力ホールディング レジスタのデータ アライメント機能を提供します。これは、システムソフトウェアにとって便利です。

ステージングおよびホールディング レジスタはともにバイト アクセスをサポートし、8 ビット以下の入出力サンプルのアライメント処理に対応します。同様に、これら 4 つのレジスタはすべて 32 ビット レジスタとしてマッピングされるため、(4 バイトのうち 3 バイトが使われるため)17 から 24 ビット間にわたるサンプルではアライメント問題が発生しません。しかし、サンプル サイズが、9 から 16 のサンプル サイズの場合、それらはホールディング/ステージング レジスタのビット 23:8 に source、sink しますが、これらのサンプルをバスのビット 15:0 で読み/書きできれば便利です。

データ アライン レジスタのビットは、システム バスのビット 15:0 がホールディング レジスタのビット 23:8 からの source か、または ステージング レジスタのビット 23:8 への sink のいずれかにできるアライメント機能を提供します。ステージング および ホールディングレジスタのそれぞれは個々に DALIGN レジスタのビットで設定可能です。ビットがハイに設定されていると、効果的なバイト シフトが起こります。

例 デルタ-シグマ ADC からの出力サンプルが 12 ビット幅で ADC の出力 サンプル レジスタの 23 ビットに配置されていて、この値を DFB に流したい場合、ADC のデータ アライメント機能をイネーブルにすればいいです。これにより、ADC の 出力サンプル レジスタの 16 ビットはバスのビット 15:0 で読み取られます DFB のアライメント機能を ステージング A 入力レジスタに設定すると、この 16 ビットはバスのビット 15:0 に書き込みできますが、必要な場合、ステージング A レジスタのビット 23:8 にも書き込みできます。

Coherency (Filter_COHER) Register

コヒーレンスは、レジスタ・フィールドがバス アクセスより広い場合にブロック障害を保護するために、DFB に含まれるハードウェアのことをさします。このケースはフィールドが部分的に書き/読み される時間に合わせてインターバルをおくことができます（インコヒーレント）。

コヒーレント チェックはオプションで、COHER レジスタでイネーブルです。ハードウェアはコヒーレント チェックを Staging および Holding レジスタの両方に提供します。

システム・ソフトウェアがフィールドを部分的に更新するだけの場合、ハードウェアがそのフィールドを使用しないように、ステージング レジスタは、書き込み保護されます。フィールドが、システムソフトウェアや DMA によって部分的に読み出される場合、ハードウェアがフィールドを更新しないように、ホールディング レジスタは読み込み保護されます。ブロックの構成によっては、ステージングおよび ホールディング レジスタのすべてのバイトが必要になるとは限りません。コヒーレント メソッドによって、いかなるサイズの出力フィールドも可能となり、正しくハンドルできます。

レジスタのビット・フィールドは Key Coherency バイトとして使用される ステージング およびホールディング レジスタの 3 つのバイトのうち、どれを選択するのかに使用されます。COHER レジスタでは、コヒーレントはイネーブルで、Key Coherency Byte が選択されます。Key Coherency Byte は、フィールドの更新が求められたときに、ハードウェアに、フィールドのどのバイトが最後に書き/読みされたかを知らせる、ユーザ(ソフトウェア）です。.

Filter Register および DMA Wizard Settings

フィルタ分解能	バースト あたりの バイト数 *	Filter_COHER および Filter_DALIGN レジスタ設定	Endian Flag *	
			PSoC 3	PSoC 5
8 ビット	1	Filter_COHER_REG = 0xAA Filter_DALIGN_REG = 0x00	オフ	オフ
16 ビット	2	Filter_COHER_REG = 0x00 Filter_DALIGN_REG = 0x0F	TD_SWAP_EN | TD_INC_DST_ADR	オフ
24ビット	4	Filter_COHER_REG = 0xAA Filter_DALIGN_REG = 0x00	TD_SWAP_EN | TD_SWAP_SIZE4 | TD_INC_DST_ADR	TD_SWAP_SIZE4

* DMA ウィザード ツールで設定

Filter_SR_REG

Filter Status Register。これを読み取り割り込みソースを取得します。Filter_ClearInterruptSource() マクロを使用して、それをクリアします。

ビット	7	6	5	4	3	2	1	0
値	INTR SEM2	INTR SEM1	INTR SEM0	INTR HOLDING REG B	INTR HOLDING REG A	RND MODE	SAT MODE	RAM SEL

このレジスタはブロック生成割り込みのステータスを示す 5 つのビットおよび データパス ユニットからの 3 ビットのステータスを含みます。

注 システム ソフトウェアがイベントをポールしたい場合、生成された割り込みがない場合、割り込みを INT_CTRL レジスタでイネーブルにする必要があります。そうすれば、ここでポールができます。

- ビット 7: Semaphore 2 Interrupt – このビットがハイの場合、セマフォ・レジスタのビット 2 は現在の割り込みのソースです。クリアするには、このビットに ‘1’ を書き込みます。
- ビット 6: Semaphore 1 Interrupt – このビットがハイの場合、セマフォ・レジスタのビット 1 は現在の割り込みのソースです。クリアするには、このビットに ‘1’ を書き込みます。
- ビット 5: Semaphore 0 Interrupt – このビットがハイの場合、セマフォ・レジスタのビット 0 は現在の割り込みのソースです。クリアするには、このビットに ‘1’ を書き込みます。
- ビット 4: Holding Register B Interrupt – このビットがハイの場合、ホールディング レジスタ B は現在の割り込みのソースです。クリアするには、このビットに ‘1’ を書き込みます。ホールディング レジスタ B を読み取り、このビットもクリアします。
- ビット 3: Holding Register A Interrupt – このビットがハイの場合、ホールディング レジスタ A は現在の割り込みのソースです。クリアするには、このビットに ‘1’ を書き込みます。ホールディング レジスタ A を読み取り、このビットもクリアします。
- ビット 2: Round Mode – DP が ラウンド モードであることを示します。これは DP ユニット外をパスする結果すべてが 16 ビット値に丸められることを意味します。
- ビット 1: Saturation Mode – DP ユニットが Saturation モードであることを示します。これは、すべての数値計算において、24 ビットの 2 の補数外の数は、許容される正または負の数に抑えられることを意味します。サチュレーション(Saturation)モードのセット、アンセットは、DFB コントローラの Assembly で制御されます。
- ビット 0: RAM Select – どの CS RAM が使用されているのかを表示します。

Filter_INT_CTRL_REG

このレジスタはイベントの割り込み生成を制御します。このレジスタのビットでイネーブルされたイベント群は、論理和がとれ、dfb_intr signal を生成します。

ビット	7	6	5	4	3	2	1	0
値	resvd	resvd	resvd	EN SEM2	EN SEM1	EN SEM0	EN HOLDING REG B	EN HOLDING REG A

ポーリング メソッドを使用する場合、選択したフィルタ チャンネルに対して、このレジスタのビット 0 または 1 のどちらかをイネーブルにしてください。これにより、データが フィルタ 出力レジスタで準備になると、割り込みを生成します。対応するステータス ビットはステータス レジスタに設定されます。ファームウェアはステータス レジスタの対応するビットをポーリングして フィルタ 出力データを読み取ることができます。

- 7 ～ 5 ビット 予約
- ビット 4: ENABLE Semaphore 2 – このビットがハイの場合、'1'がセマフォ レジスタのビット 2 に書き込まれるたびに割り込みが生成されます。
- ビット 3: ENABLE Semaphore 1 – このビットがハイの場合、'1' がセマフォ レジスタのビット 1 に書き込まれるたびに割り込みが生成されます。
- ビット 2: ENABLE Semaphore 0 – このビットがハイの場合、'1' がセマフォ レジスタのビット 0 に書き込まれるたびに割り込みが生成されます。
- ビット 1: ENABLE HOLDING Register B – このビットがハイの場合、新しい有効データが出力 ホールディング レジスタ B に書き込まれるたびに割り込みが生成されます。
- ビット 0: ENABLE HOLDING Register A – このビットがハイの場合、新しい有効データが出力 ホールディング レジスタ A に書き込まれるたびに割り込みが生成されます。

PSoC 3 の DC および AC 電気的特性

フィルタ DC 仕様

パラメータ	説明	条件	Min	Typ	Max	単位
	DFBの動作電流	F_{DFB} で 64 タップ FIR				
		100 kHz (1.3 ksps)	--	0.03	0.05	mA
		500 kHz (6.7 ksps)	--	0.16	0.27	mA
		1 MHz (13.4 ksps)	--	0.33	0.53	mA
		10 MHz (134 ksps)	--	3.3	5.3	mA
		48 MHz (644 ksps)	--	15.7	25.5	mA
		67 MHz (900 ksps)	--	21.8	35.6	mA

フィルタの AC 仕様

パラメータ	説明	条件	Min	Typ	Max	単位
F_{DFB}	DFBの動作周波数		DC	--	67	MHz

PSoC 5 の DC および AC 電気特性

フィルタ DC 仕様

パラメータ	説明	条件	Min	Typ	Max	単位
	DFBの動作電流	F_{DFB} で 64 タップ FIR				
		100 kHz (1.3 ksps)	--	0.03	0.075	mA
		500 kHz (6.7 ksps)	--	0.16	0.3	mA
		1 MHz (13.4 ksps)	--	0.33	0.57	mA
		10 MHz (134 ksps)	--	3.3	5.5	mA
		48 MHz (644 ksps)	--	15.7	26	mA
		67 MHz (900 ksps)	--	21.8	35.6	mA

フィルタの AC 仕様

パラメータ	説明	条件	Min	Typ	Max	単位
F_{DFB}	DFBの動作周波数		DC	--	67	MHz

コンポーネントの変更

ここでは、過去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
2.0	IIR フィルタをサポート	IIR フィルタは旧バージョンではサポートされていませんでした。
	FIR フィルタの再設計	旧バージョンの FIR フィルタのウィンドウ表示バージョンでは問題がありました。
	更新に関連した Filter DMA ウィザード	Filter はデータ READY 信号として DMA を設定せずに、DMA を使用してデータ転送の転送先として設定できます。
	フィルタ・パラメータ設定に基づく要求バス・クロック周波数を表示するため、構成ウインドウに表示フィールドを追加しました。	パラメータの選択に基づくバス・クロック要件が分かります。
	フィルタ構成ウインドウの更新	カスタム係数のオプションを提供し、見栄えと印象も改善されています。
1.50.a	データシートに特性データを追加	
	データシートのマイナーな編集と更新	
1.50	Sleep/Wakeup (スリープ/ウェイクアップ) と Init/Enable (初期化/イネーブル) API を追加。	ローパワー・モードをサポートし、ほとんどのコンポーネントの初期化とイネーブル化の制御を分離する共通インターフェースを提供するため。
	コンポーネントに DMA 能力ファイルを追加しました。	このファイルにより、PSoC Creator で Filter が DMA Wizard ツールでサポートされるようになります。

Copyright © 2005-2012 Cypress Semiconductor Corporation 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 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。特許又はその他の権限下で、ライセンスを譲渡又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、又は安全の用途のために仕様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を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ツ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 Designer™及び Programmable System-on-Chip™は、Cypress Semiconductor Corp.の商標、PSoC®は同社の登録商標です。本文書で言及するその他全ての商標又は登録商標は各社の所有物です。

全てのソースコード(ソフトウェア及び/又はファームウェア)は Cypress Semiconductor Corporation (以下「サイプレス」)が所有し、全世界(米国及びその他の国)の特許権保護、米国の著作権法並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る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及び/又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複製、使用、変更、そして作成するためのライセンス、並びにサイプレスのソースコード及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、又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、明示的又は黙示的を問わず、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、商品性又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又は回路を適用又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ツ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。