6 ~ 14-Bit Delta Sigma (デルタ シグマ) ADC データシート DelSigMulti V 1.2

Copyright © 2009-2011 Cypress Semiconductor Corporation. All Rights Reserved.

変調器オーダー	デシメーションレート	分解能	サンプルレート (CLK=2 MHz)	サンプルレート (CLK=8 MHz)	デシメータ数	SC ブロック	Flash	RAM	チャネル (I/O ピン)
CY8C28x45, CY8C28x43, CY8C28x52, CY8C28x33, CY8C28x23									
1	32	6	15625.0	62500.0	2	2	148	3	2
1	64	7.5	7812.5	31250.0	2	2	156	3	2
1	128	9	3906.3	15625.0	2	2	185	5	2
1	256	10.5	1953.1	7812.5	2	2	185	5	2
2	32	8	15625.0	62500.0	2	2	187	5	2
2	64	10	7812.5	31250.0	2	2	216	7	2
2	128	12	3906.3	15625.0	2	2	216	7	2
2	256	14	1953.1	7812.5	2	2	216	7	2
1	32	6	15625.0	62500.0	3	6	180	6	3
1	64	7.5	7812.5	31250.0	3	6	192	6	3
1	128	9	3906.3	15625.0	3	6	234	9	3
1	256	10.5	1953.1	7812.5	3	6	234	9	3
2	32	8	15625.0	62500.0	3	6	215	6	3
2	64	10	7812.5	31250.0	3	6	257	9	3
2	128	12	3906.3	15625.0	3	6	257	9	3
2	256	14	1953.1	7812.5	3	6	257	9	3
1	32	6	15625.0	62500.0	4	8	200	7	4
1	64	7.5	7812.5	31250.0	4	8	216	7	4
1	128	9	3906.3	15625.0	4	8	271	11	4
1	256	10.5	1953.1	7812.5	4	8	271	11	4

変調器オーダー	デシメーションレート	分解能	サンプルレート (CLK=2 MHz)	サンプルレート (CLK=8 MHz)	デシメータ数	SC ブロック	Flash	RAM	チャネル (I/O ピン)
2	32	8	15625.0	62500.0	4	8	243	7	4
2	64	10	7812.5	31250.0	4	8	298	11	4
2	128	12	3906.3	15625.0	4	8	298	11	4
2	256	14	1953.1	7812.5	4	8	298	11	4

その他のコンバータについては、アプリケーション ノート「アナログ - ADC の選択」AN2239 を参照してください。

特徴と概要

- 6-bit ~ 14-bit 分解能
- 2 ~ 4 チャネルの同期サンプリング
- 符号なしまたは符号付き 2 の補数のフォーマットによるデータ
- 最大サンプリング速度は 6 ビット分解能で 65,500 sps、14-bit 分解能で 7812 sps
- ハードウェアに完全実装された $Sinc^N$ フィルタが、CPU オーバヘッドとアンチエイリアスの要件を軽減
- 信号対雑音比を改善する 1 次または 2 次変調器。ユーザ選択可
- 入力範囲は内部及び外部リファレンス オプションによって選択可能
- デジタルブロックが不要
- 構成ウィザードによって、相互に同期しているデルタ - シグマ ADC 測定の 2、3、4 チャネルからの選択が簡単に
- デシメータの内部タイマがデジタルブロックの使用を不要に

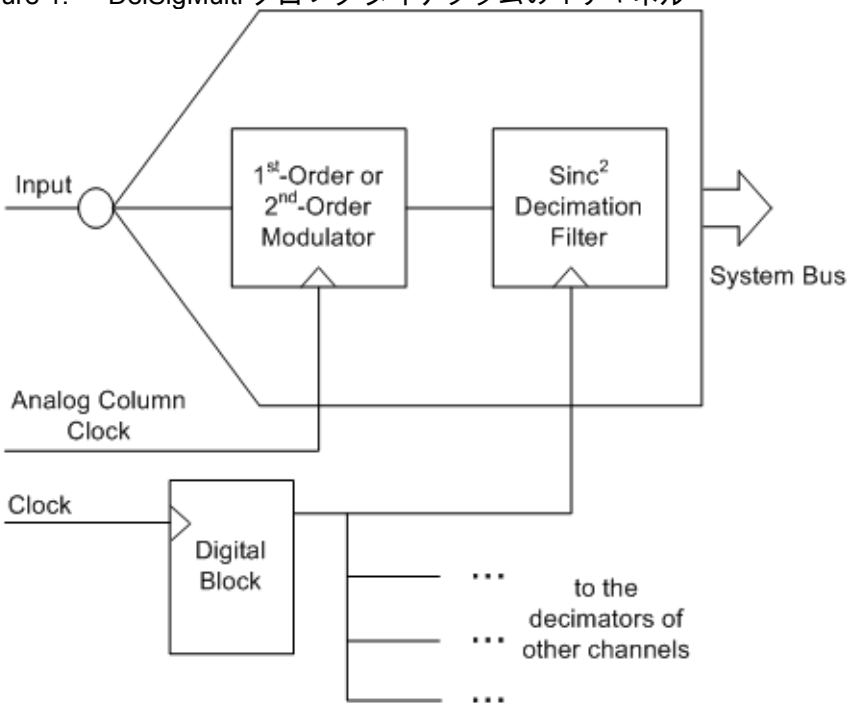
DelSigMulti ユーザ モジュールは、単一の出力サンプルを生成するために 32 ~ 256 の積分サイクルを必要とする、積分型変換器です。マルチプレックス入力を変更すると、変更後最初の 2 つのサンプルが無効になります。DelSigMulti ユーザ モジュールは、最大 4 チャネルの同時、同期のデルタ - シグマ ADC サンプリングをサポートします。

構成ウィザードによって、各チャネルで使用されるアナログブロック数と、各チャネルのデシメータオーバーサンプル速度を簡単に選択できます。

モジュールを配置する前に、「パラメータ」のセクションをお読みください。

Note 1 チャネルのみ、または複数の非同期チャネルが必要なときは、通常の 「DelSig」または「DelSigPlus」ユーザ モジュールを使います。

Figure 1. DelSigMulti ブロック ダイアグラムの 1 チャネル



機能説明

図 1 に示すように、DelSigMulti ユーザ モジュールは 3 つの主要機能で構成されています。

- 変調器
- $Sinc^2$ デシメーション フィルタ
- タイミング信号発生器

各コンポーネントは、アプリケーションにおいてパフォーマンスとリソース利用の適正なバランスを実現するために選択･ 調整できるオプションを提供します。

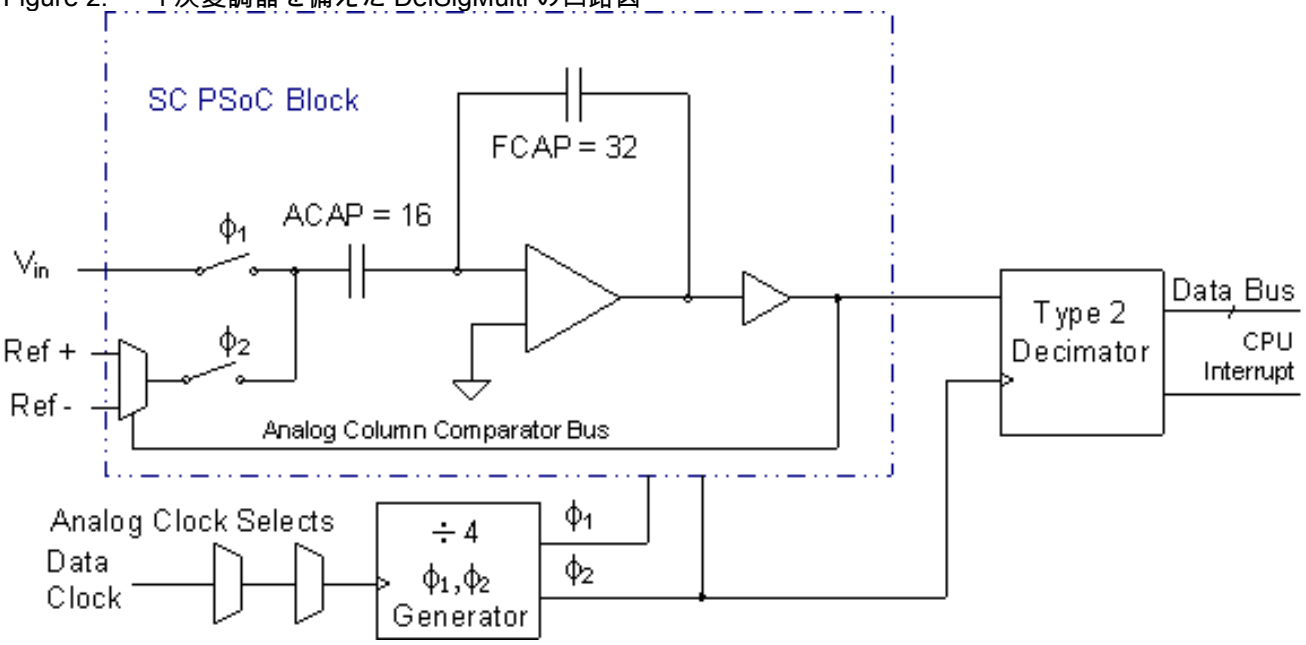
変調器

変調器は 1-bit オーバーサンプリング回路で、それが生成する 1 及び 0 の密度を単位として入力電圧を表します。変調器の出力は、複数の 1-bit サンプルをより高い分解能のサンプルに変換するローパス デシメーション フィルタにより、最終のサンプリング速度に低減されます。一般に、より高い復号化レート (つまりより高いオーバーサンプリング速度) は、より高い分解能の結果を生成することができますが、変調器の次数などその他の要因も関係します。

デルタシグマ変換器の主な利点は、変調器が行う 「ノイズ シェーピング」です。通常、信号サンプリングに特有の量子化ノイズは、「DC」とサンプル周波数の半分またはナイキスト周波数の間の周波数で、ほぼ均等に分散しています (「白」)。簡単に言うと、デルタシグマ変調器は、量子化ノイズの一部を低周波数から高周波数にシフトします。これは後で、デシメーション フィルタによって減衰されます。2 つのスイッチド キャパシタ アナログ PSoC ブロックを必要とする 2 次変調器は、1 つのアナログ PSoC ブロックのみを必要とする 1 次変調器よりも、ノイズ シェーピングで優れています。256X のより高い復号化レートで 2 次変調器は、1 次変調器に比べ、実効分解能において 3.5-bit の増加が見られます。

[+] Feedback

Figure 2. 1 次変調器を備えた DelSigMulti の回路図



アナログ ブロックは、積分器として構成されます。コンパレータの出力極性はリファレンスマルチプレクサを構成し、それによってリファレンス電圧が入力に加算または入力から減算され、積分器に置かれるように構成されます。リファレンス制御は、積分器の出力をゼロに向かって引き戻そうとします。単一ビットコンパレータの出力もまた、デシメータ $sinc^2$ フィルタに入力されます。

注：1-bit オーバーサンプリング速度は、スイッチド キャパシタ PSoC ブロックを制御する φ_1 と φ_2 を生成する四分周信号発生器によって決定します。出力レートは、1-bit オーバーサンプリング速度を得るためにデータを 4 で割り、最終サンプリング速度を得るためにさらにデシメーションレートで割ることによって決定します。サンプリング速度は次の式で求められます。

Equation 1

$$SampleRate = \frac{DataClockFrequency}{4 \times DecimationRate} \text{ samples per second}$$

使用できるもっとも高いデータクロック周波数は、仕様表で示されています。データクロックが 8 MHz で、デシメーションレートが 256 の場合、サンプリング速度は次のようになります。

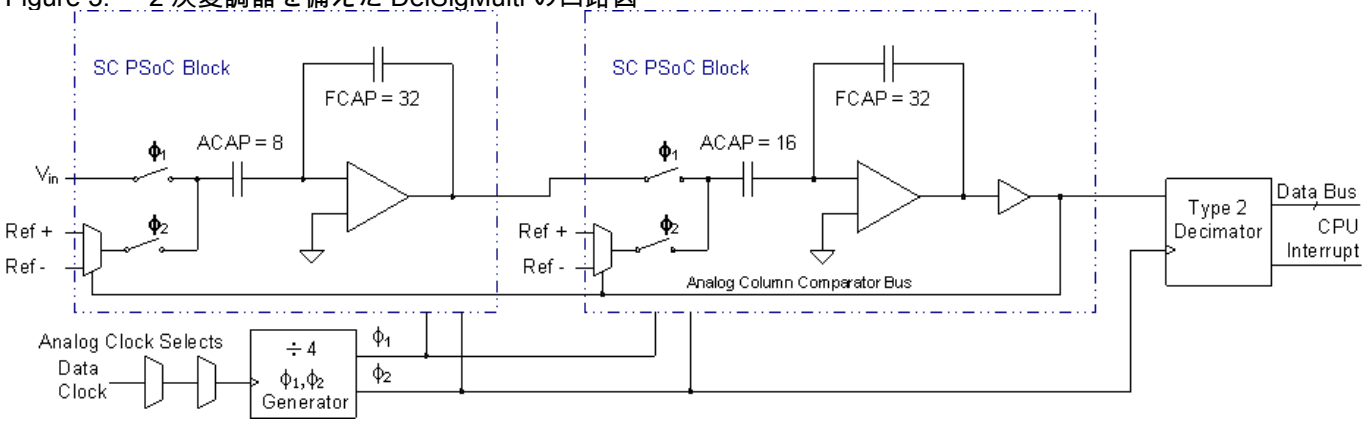
Equation 2

$$8 \times 10^6 / (4 \times 256) = 7812.5 sps$$

[+] Feedback

2 次変調器は、1 次変調器のアナログ出力を同様の PSoC ブロックにフィードし、2 つ目のブロックの 1-bit コンパレータ出力がここに示すように両方のブロックに戻るよう、フィードバック構成を変更することによって作成します。

Figure 3. 2 次変調器を備えた DelSigMulti の回路図



アナログ コンパレータのバスは、アナログ PSoC ブロック アレイのコラムで垂直に走るため、2 次変調器のブロックは、ほかのものより 1 つ上に配置する必要があります。

DelSigMulti の範囲は、$\pm V_{Ref}$ によって規定されます。V_{Ref} は PSoC Designer のグローバル リソース ウィンドウで設定します。固定スケールの場合、V_{Ref} は $\pm V_{Bandgap}$ に設定されます。PSoC デバイスの CY8C29x66 ファミリでは $\pm 1.6\ V_{Bandgap}$ です。調整可能なスケールの場合、V_{Ref} は ±Port 2[6] に設定されます。レシオメトリック スケールでは、V_{Ref} は $\pm V_{DD}/2$ に設定されます。オプションの完全なリストを表 1 に示します。

Table 1. Ref Mux グローバル パラメータ設定の入力電圧範囲

RefMux の設定	Vdd = 5V	Vdd = 3.3V
(Vdd/2) ± BandGap	$1.2 < V_{in} < 3.8$	$0.35 < V_{in} < 2.95$
(Vdd/2) ± (Vdd/2)	$0 < V_{in} < 5$	$0 < V_{in} < 3.3$
BandGap ± BandGap	$0 < V_{in} < 2.6$	$0 < V_{in} < 2.6$
(1.6*BandGap) ± (1.6*BandGap)	$0 < V_{in} < 4.16$	適用外
(2*BandGap) ± BandGap	$1.3 < V_{in} < 3.9$	適用外
(2*BandGap) ± P2[6]	$(2.6 - V_{P2[6]}) < V_{in} < (2.6 + V_{P2[6]})$	適用外
P2[4] ± BandGap	$(V_{P2[4]} - 1.3) < V_{in} < (V_{P2[4]} + 1.3)$	$(V_{P2[4]} - 1.3) < V_{in} < (V_{P2[4]} + 1.3)$
P2[4] ± P2[6]	$(V_{P2[4]}-V_{P2[6]}) < V_{in} < (V_{P2[4]}+V_{P2[6]})$	$(V_{P2[4]}-V_{P2[6]}) < V_{in} < (V_{P2[4]}+V_{P2[6]})$

[+] Feedback

Sinc^2 デシメーション フィルタ

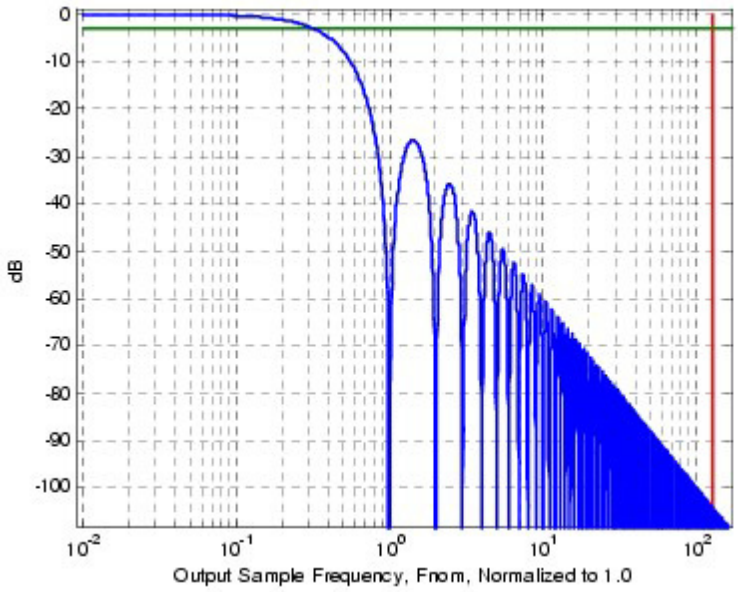
デシメーション フィルタの応答は、この z ドメイン関係によって得られます。

Equation 3

$$H(z) = \left[\frac{1 - z^{-n}}{1 - z^{-1}}\right]^2, \text{ where n is the decimation level.}$$

このセクションでプロットした周波数ドメイン伝達関数は、出力サンプリング速度 F_{nom} が 1.0 になるように正規化されます。-3 dB のポイントは 0.318×F_{nom} のすぐ上で発生し、関数は F_{nom} の各整数倍でゼロになります。1-bit サンプリング速度は定格出力速度よりも 32 ~ 256 高いため、ナイキスト制限は F_{nom} の 4 ~ 7 オクターブ上となり、アンチエイリアス フィルタの要件を著しく低減します。256 の復号化レートに対する 1-bit ナイキスト周波数は、グラフの右側に太い垂直線で示されています。さらに高い復号化レートも可能ですが、デバイスのノイズフロアのため、利点はほとんどありません。14-bit トポロジ、256 の復号化レートを持つ 2 次変調器の場合は、分解能が信号対雑音比により制限されます。DC 測定または移動速度の遅い信号でリピート可能な 14-bit 分解能を得るには、複数の出力サンプルの平均を取るか、高度な信号処理技法を採用することが必要となります。

Figure 4. -3 dB 及びナイキスト周波数を使った Sinc^2 デシメーション フィルタの振幅特性



以前の DELSIG8 と DELSIG11 とは異なり、このユーザ モジュールは伝達関数の分子と分母をすべてハードウェアで実行します。これには、改良型「タイプ 2」のデシメータが必要です。これは 1 次および 2 次の変調器トポロジに使用されます。デシメータは、1-bit サンプリング速度で動作する二重積分器により、伝達関数の分母を実装します。分子は、定格出力サンプリング速度で動作する二重微分器 (2 つ目の差分演算子) によって実装されます。DelSigMulti ユーザ モジュールが消費する CPU のオーバーヘッドと割り込みレイテンシは、I/O スペースのデシメータレジスタからサンプルデータを回復するために必要な約 80 サイクル以下に限定されています。タイプ 2 のデシメータは、n ビットコンバータ用に 0 ~ 2^n-1 の範囲の符号無しの値を生成します。割込みサービス ルーチンは、これを -2^{n-1} ~ $+2^{n-1}$-1 の範囲の 2 の補数に変換するように構成できます。

[+] Feedback

Table 2.　デルタ シグマ ADC の機能表

機能	デルタ シグマ ADC			
	DELSIG8, DELSIG11	DelSig	DelSigPlus	DelSigMulti
分解能	8, 11	6-14	6-14	6-14
デジタルブロック	1	1-2	0	0
アナログブロック	1-2	1-2	1-2	2-8
サポートされているコンポーネント	CY8C24/27/29。CY8C24x94、CY8C28x45 は未サポート	CY8C24x94, CY8C29xxx, CY8C28x45	CY8C24x94, CY8C28x45	CY8C28x45
CPU オーバーヘットと割り込みレイテンシ	高	低	低	低

タイミング信号発生器と要件

アナログ変調器に φ_1 と φ_2 クロックを供給する四分周信号発生器は、デシメータにビットクロックも供給します。出力サンプリング速度に対応するデシメーションファクタは、ワードクロックによって決定します。ワードクロックは、デシメータの内部タイマによって生成されます。

タイプ 2 デシメータは、$Sinc^2$ フィルタの完全にハードウェア化されたバージョンです。そのアーキテクチャによって、デシメーションと割り込みの目的に使用できる内部タイマのオプションが可能になります。効果的な分解能の計算には、次の式を使用します。

一段変調器：(log2(DecimatorRate) - 1) × 1.5

二段変調器：(log2(DecimatorRate) - 1) × 2

DataFormat ビットは、符号付き（2 の補正出力）と符号無し（オフセットバイナリデータ）として加重することができます。

Table 3.　デシメータデータの出力シフト

デシメーションレート	変調器タイプ	有効な分解能	シフト
32	一段	6	4
32	二段	8	2
64	一段	8 (7.5)	4
64	二段	10	2
128	一段	9	5
128	二段	12	2
256	一段	11(10.5)	5
256	二段	14	2

DC および AC の電気的特徴

以下の値は初期の特性データを基に、予測される性能を入れておきます。別途記載がない限り、下記では T_A = -40、25、85、125°C および V_{dd} = 5.0V を想定しています。

Table 4. 5.0V 結果の概要

パラメータ	標準値	制限	単位	摘要
8 ビット、24MHz CPU Clk、1 MHz データクロック、高出力				
ゲイン	-2.6482	2	%FSR	
オフセット	-47.0072	13	mV	
DNL	0.161	<1	LSB	
INL	0.27	--	LSB	
SNR	45.86	--	dB	
8 ビット、24MHz CPU Clk、2 MHz データクロック、高出力				
ゲイン	-2.3168	2	%FSR	
オフセット	-62.3507	13	mV	
DNL	0.069	<1	LSB	
INL	0.172	--	LSB	
SNR	45.86	--	dB	

以下の値は初期の特性データを基に、予測される性能を入れておきます。別途記載がない限り、T_A = -40、25、85、125°C および V_{dd} = 3.3V を想定しています。

Table 5. 3.3V 結果の概要

パラメータ	標準値	制限	単位	摘要
8 ビット、24MHz CPU Clk、1 MHz データクロック、高出力				
ゲイン	-2.7182	2	%FSR	
オフセット	-40.1334	5	mV	
DNL		<1	LSB	
INL		--	LSB	
SNR		--	dB	
8 ビット、24MHz CPU Clk、2 MHz データクロック、高出力				
ゲイン	-2.8219	2	%FSR	
オフセット	-42.8073	5	mV	
DNL	0.064	<1	LSB	
INL	0.161	--	LSB	
SNR	46.02	--	dB	

配置

ツールバーで DelSigMulti ユーザ モジュールを選択するか、セレクタービューでそのアイコンをダブルクロックすると、選択ウィンドウが開き、適切なトポロジを選択するガイダンスが表示されます。トポロジは、配置ビューでユーザ モジュールを右クリックして、コンテンツメニューで [User Module Selection Options...] （ユーザ モジュール選択オプション）を選択すると変更できます。

1 次変調器を設計するには、2 つの PSoC ブロック、1 つのデジタルブロック、および 1 つのアナログブロックが必要です。アナログ ブロックは、どのスイッチド キャパシタ PSoC ブロックにも配置できます。

2 次変調器デザインは、2 つのスイッチド キャパシタ PSoC ブロック、ADC0、ADC1 を利用しています。これらを連結するアナログコンパレータバスがアナログアレイの各コラムを縦に通っているため、スイッチド キャパシタ PSoC ブロックは縦に上下させて並べなければなりません。

アナログ ブロックでは多数の配置が可能ですが、DelSigMulti は、PSoC デバイス唯一のハードウェアデシメータフィルタも使用します。デシメータは、アナログブロックが配置されたときに自動的に割り当てられます。ユーザによる操作は不要です。このため、1 つの構成では、DelSigMulti ユーザ モジュールのインスタンスは 1 つだけ配置できます。ダイナミックな再構成の場合、一度に 2 つ以上の構成を読み込む（有効にする）ことが可能で、2 つの DelSigMulti ユーザ モジュールが同時に稼動することを抑止するためのチェックは行われません。この状態では、両方のインスタンスが稼動しているように見えますが、一番最近読み込んだインスタンスがデシメーションフィルタを制御します。ただし、両方の割り込みが実行され、干渉をきたす場合もあります。

パラメータおよびリソース

DelSigMulti のインスタンスが配置されると、適正に作動するように幾つかのパラメータを構成する必要がでてきます（入力信号マルチプレクサ選択、クロック位相、ポーリング選択）。

DataFormat0/1/2/3

このパラメータは、符号無し（デフォルト）または符号付きの値をとることがあります。n ビットの分解能の場合、符号無しデータはゼロから 2^n-1 の値です。符号付きデータは、-2^{n-1} ~ $+2^{n-1}$-1 の間の値です。0/1/2/3 は、使用されるデルタ - シグマ ADC チャネル数を示します。

Clock Phase0/1/2/3

クロック位相の選択は、あるアナログ PSoC ブロックの出力を別のアナログ PSoC ブロックの入力と同期させるために使用します。スイッチド キャパシタ アナログ PSoC ブロックは、2 相クロック (φ_1, φ_2) を使用して、信号を取得および転送します。通常、DelSigMulti への入力は、φ_1 でサンプリングされます。ユーザ モジュールの多くは、φ_1 中に出力を自動的にゼロに設定し、φ_2 中しか有効な出力を供給しないため、問題が発生します。そのようなモジュールの出力が DelSigMulti 入力に配信された場合、DelSigMulti は不定値をサンプリングします。クロック位相を選択することで、入力信号を $\varphi_2$2 中に取得するように、位相を交換できます。0/1/2/3 は、使用されるデルタ - シグマ ADC チャネル数を示します。

PosInput0/1/2/3

このパラメータは、シングルエンド入力の信号ソース、または差分入力の非反転入力を決定します。0/1/2/3 は、使用されるデルタ - シグマ ADC チャネル数を示します。

NegInput0/1/2/3 および NegInputGain0/1/2/3

NegInput は、差分信号ペアの反転入力用ソースを選択します。シングルエンドの入力を使用した場合、このパラメータには任意の有効値を設定できます。NegInputGain パラメータを [Disconnected]（接続解除）（ゼロゲイン）に設定すると、コンバータからの接続を解除できます。

NegInputGain は非反転入力と比例して、反転入力のゲインを調整します（上述の「NegInput パラメータ」セクションを参照してください）。シングルエンドの入力の場合、このパラメータは [Disconnected]（接続解除）となります。差分入力では、NegInputGain は 1.000 に設定できます。必要に応じて、反転入力に適用されたゲインは、非反転入力と比例して、0.0625 ~ 1.9375 の範囲で 1/16th の増分を用いて調整できます。0/1/2/3 は、使用されるデルタ - シグマ ADC チャネル数を示します。

割り込み生成制御

PSoC Designer の "Enable interrupt generation control"（割り込み生成制御を有効にする）チェックボックスがチェックされている時に、さらに二つのパラメータが設定できます。**Enable interrupt generation control**（割り込み生成制御をイネーブルにする）チェックボックスがチェックされている場合だけです。これらは、**[Project]（プロジェクト）> > [Settings]（設定）> > [Chip Editor]（チップ エディタ）>** でアクセスできます。「割り込み生成の制御」は、オーバーレイ全体で複数のユーザ モジュールにより共有される割り込みとともに、複数のオーバーレイが使用される場合に重要です。

- 割り込み API
- IntDispatchMode

[+] Feedback

InterruptAPI

InterruptAPI パラメータを使うと、ユーザ モジュールの割り込みハンドラと割り込みベクタ テーブル エントリの状況に応じた生成が可能になります。「Enable (有効)」を選択すると、割り込みハンドラと割り込みベクトル テーブル エントリが生成されます。「Disable (無効)」を選択すると、割り込みハンドラと割り込みベクトル テーブル エントリがバイパスされます。

複数のオーバーレイがあり、1 つのブロック リソースが異なるオーバーレイで使用されているプロジェクトの場合、特に注意が必要です。コードスペースを節約するために実際にこれを必要とするオーバーレイの場合のみ、割り込みを生成するよう選択します。

IntDispatchMode

IntDispatchMode パラメータを使用して、同一ブロック内の異なるオーバーレイ内に存在する複数のユーザ モジュールで共用されている割り込みについて、割り込み要求をどのように処理するかを指定します。ActiveStatus を選択すると、ファームウェアが、共有される割り込みリクエストに応答する前に、どちらのオーバーレイがアクティブであるかをテストします。このテストは、共用割り込みが要求されるたびに行われます。このためにレイテンシが付加され、共用割り込み要求を処理する非決定性のプロセスも生じますが、RAM は不要です。OffsetPreCalc を選択すると、ファームウェアが、オーバーレイが最初にロードされるときだけ、共有割り込みリクエストのソースを計算するようになります。この計算によって割り込みレイテンシは減少し、共用割り込み要求を処理する決定性のプロセスが生じますが、RAM のバイトを消費します。

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) ルーチンは、設計者が高級言語でモジュールを操作できるようにユーザ モジュールをコンポーネントとして提供します。このセクションでは、各機能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。

ユーザ モジュールを配置するたびに、インスタンス名がアサインされます。デフォルトでは、PSoC Designer は指定されたプロジェクトで、DelSigMulti_1 をこのユーザ モジュールの最初の例として指定されています。これは識別子の構文ルールに従って、独自のルールに変更できます。割り当てられたインスタンスは、グローバル関数名、変数、定数記号のプレフィクスとなります。次の説明では、簡単にするために、インスタンス名は省略されて単に 「DelSigMulti」となっています。

Note すべてのユーザ モジュール API の場合と同じように、API 関数を呼び出すことで A と X レジスタの値が変更されることがあります。関数を呼び出した後で A および X レジスタの値が必要になるのであれば、関数を呼び出す側の責任において、これらのレジスタの値を関数呼び出し前に退避させてください。この 「レジスタは脆弱」というポリシーは、効率面の理由から採用され、PSoC デザイナのバージョン 1.0 から発効しています。C コンパイラは、自動的にこの条件で処理されています。アセンブラ言語のプログラマは、コードがこのポリシーを遵守していることも確認する必要があります。一部のユーザ モジュール API 関数では、A と X は変更されないこともありますが、将来も変更されないという保証はありません。

DelSigMulti_Start

説明：

このユーザ モジュールで必要なすべての初期化を実行し、スイッチド キャパシタ PSoC ブロックの出力レベルを設定します。すべてのチャネルが同じ出力レベルで構成されています。

C プロトタイプ：

```
void  DelSigMulti_Start(BYTE bfPowerSetting)
```

アセンブラ：

```
mov   A, bfPowerSetting
lcall  DelSigMulti_Start
```

パラメータ：

bPowerSetting: 4 つの 2-bit ビットフィールドを持つ 1 バイトは、各チャネルの出力レベルを個別に指定します。リセットとコンフィグレーションの後、DelSigMulti に割り当てられたアナログ PSoC ブロックの電力が遮断されます。C およびアセンブラで記号名が登録されていますが、関連する値を以下の表に示します。

記号名	マスク
DelSigMulti_CH0_OFF	00h
DelSigMulti_CH0_LOWPOWER	01h
DelSigMulti_CH0_MEDPOWER	02h
DelSigMulti_CH0_HIGHPOWER	03h
DelSigMulti_CH1_OFF	00h
DelSigMulti_CH1_LOWPOWER	04h
DelSigMulti_CH1_MEDPOWER	08h
DelSigMulti_CH1_HIGHPOWER	0Ch
DelSigMulti_CH2_OFF	00h
DelSigMulti_CH2_LOWPOWER	10h
DelSigMulti_CH2_MEDPOWER	20h
DelSigMulti_CH2_HIGHPOWER	30h
DelSigMulti_CH3_OFF	00h
DelSigMulti_CH3_LOWPOWER	40h
DelSigMulti_CH3_MEDPOWER	80h
DelSigMulti_CH3_HIGHPOWER	C0h

戻り値：

なし

副作用：

A および X レジスタは、この関数のこの実装または将来の実装で変更される場合があります。大容量メモリ モデル (c) のすべての RAM ページ ポインタ レジスタにも同じことが言えます。必要に応じて、fastcall16 関数の呼び出しでこれらの値を保存してください。

DelSigMulti_Stop

説明：

スイッチド キャパシタ PSoC ブロックの出力レベルをオフに設定します。

C プロトタイプ：

```
void  DelSigMulti_Stop(void)
```

アセンブラ：

```
lcall  DelSigMulti_Stop
```

パラメータ：

なし

戻り値：

なし

副作用：

A および X レジスタは、この関数のこの実装または将来の実装で変更される場合があります。大容量メモリ モデル (c) のすべての RAM ページ ポインタ レジスタにも同じことが言えます。必要に応じて、fastcall16 関数の呼び出しでこれらの値を保存してください。

DelSigMulti_SetPower

説明：

UM の各チャネルに対するスイッチド キャパシタ PSoC ブロックの出力レベルを設定します。

C プロトタイプ：

```
void  DelSigMulti_SetPower(BYTE bPowerSetting)
```

アセンブラ：

```
mov   A, bPowerSetting
lcall  DelSigMulti_SetPower
```

パラメータ：

bPowerSetting: 4 つの 2-bit ビットフィールドを持つ 1 バイトは、各チャネルの出力レベルを個別に指定します。リセットとコンフィグレーションの後、DelSigMulti に割り当てられたアナログ PSoC ブロックの電力が遮断されます。C およびアセンブラで記号名が登録されていますが、関連する値を以下の表に示します。

記号名	マスク
DelSigMulti_CH0_OFF	00h
DelSigMulti_CH0_LOWPOWER	01h
DelSigMulti_CH0_MEDPOWER	02h
DelSigMulti_CH0_HIGHPOWER	03h
DelSigMulti_CH1_OFF	00h
DelSigMulti_CH1_LOWPOWER	04h
DelSigMulti_CH1_MEDPOWER	08h
DelSigMulti_CH1_HIGHPOWER	0Ch
DelSigMulti_CH2_OFF	00h
DelSigMulti_CH2_LOWPOWER	10h
DelSigMulti_CH2_MEDPOWER	20h
DelSigMulti_CH2_HIGHPOWER	30h
DelSigMulti_CH3_OFF	00h
DelSigMulti_CH3_LOWPOWER	40h
DelSigMulti_CH3_MEDPOWER	80h
DelSigMulti_CH3_HIGHPOWER	C0h

戻り値：

なし

副作用：

A および X レジスタは、この関数のこの実装または将来の実装で変更される場合があります。大容量メモリ モデル (c) のすべての RAM ページ ポインタ レジスタにも同じことが言えます。必要に応じて、fastcall16 関数の呼び出しでこれらの値を保存してください。

DelSigMulti_StartAD

説明：

このユーザ モジュール用に割り込みを有効にし、サンプリングを開始します。

C プロトタイプ：

```
void  DelSigMulti_StartAD(void)
```

アセンブリ：

```
lcall  DelSigMulti_StartAD
```

パラメータ：

なし

戻り値：

なし

副作用：

A および X レジスタは、この関数のこの実装または将来の実装で変更される場合があります。大容量メモリ モデル (c) のすべての RAM ページ ポインタ レジスタにも同じことが言えます。必要に応じて、fastcall16 関数の呼び出しでこれらの値を保存してください。

DelSigMulti_StopAD

説明：

割り込みの無効化により、A/D をシャットダウンします。ただし、アナログブロックへの電力供給は続行されます。

C プロトタイプ：

```
void  DelSigMulti_StopAD(void)
```

アセンブラ：

```
lcall  DelSigMulti_StopAD
```

パラメータ：

なし

戻り値：

なし

副作用：

A および X レジスタは、この関数のこの実装または将来の実装で変更される場合があります。大容量メモリ モデル (c) のすべての RAM ページ ポインタ レジスタにも同じことが言えます。必要に応じて、fastcall16 関数の呼び出しでこれらの値を保存してください。

DelSigMulti_fIsDataAvailable

説明：

新しい ADC サンプルデータの準備ができているかどうかをチェックします。ADC からの新しい結果が出ると、フラグがセットされます。この API 関数によって、ユーザがフラグがチェックできます。UM 全体のデータ可用性をチェックします。すべてのチャネルが同期されるため、すべてのデータが新しいか、新しくないかのいずれかになります。従って、たった 1 つのチェックがすべてのチャネルのすべてのサンプルデータに適用できるよう作られます。ユーザは、ほかの API 関数を使用してこのフラグをクリアしなければなりません。

C プロトタイプ：

```
BYTE  DelSigMulti_fIsDataAvailable(void)
```

アセンブラ：

```
lcall  DelSigMulti_fIsDataAvailable
cmp   A, 0
jz    .DataNotAvailable
```

パラメータ：

なし

戻り値：

各チャネルのデータが変換され、読み出しの準備ができていると、ゼロ以外の値が返されます。各チャネルのデータが変換されず、読み出しの準備ができていないと、ゼロが返されます。

副作用：

A および X レジスタは、この関数のこの実装または将来の実装で変更される場合があります。大容量メモリ モデル (c) のすべての RAM ページ ポインタ レジスタにも同じことが言えます。必要に応じて、fastcall16 関数の呼び出しでこれらの値を保存してください。現時点では、CUR_PP ページ ポインタ レジスタだけが変更されています。

DelSigMulti_GetAllDataClearFlag

説明：

この関数は、1 つの関数を伴うチャネルデータを取得するために使用されます。RAM アレイポインタがこの関数に渡され、関数は ADC の結果をこの RAM アレイに配置します。ユーザは、RAM アレイが ADC チャネルの番号と分解能に対応する正しいサイズであることを確認してください。またこの関数は、これらの信号について、ADC データの準備が整っているというフラグをクリアします。関数が呼び出される前にデータサンプルの準備が整っていることを検証するために、DelSigMulti_fIsDataAvailable() を呼び出す場合もあります。

C プロトタイプ：

```
void DelSigMulti_GetAllDataClearFlag(BYTE* pbRamBuffer)
```

アセンブラ：

```
mov   A, >pbRamBuffer
mov   X, <pbRamBuffer
lcall  DelSigMulti_GetAllDataClearFlag
```

パラメータ：

pbRamBuffer: このパラメータは、ADC サンプルデータがコピーされる RAM 位置を特定します。コピーされるデータは ADC の結果です。ユーザは、RAM アレイタイプが ADC 結果のデータタイプに適合することを確認する必要があります。ADC チャネルデータは、チャネル 0 データから開始するユーザアレイに配置され、バイトの順番は MSB が最初です。例えば、UM に 3 チャネルの符号無しの 10-bit データがある場合、このアレイは次の順番で 6 バイトをユーザアレイにコピーします。Ch0_Result_MSB, Ch0_Result_LSB, Ch1_Result_MSB, Ch1_Result_LSB, Ch2_Result_MSB, Ch2_Result_LSB. ユーザのアレイタイプが BYTE タイプと異なる場合、ユーザはアレイポインタを関数に渡し、アレイポインタは BYTE ポインタになります。

戻り値：

なし

副作用：

A および X レジスタは、この関数のこの実装または将来の実装で変更される場合があります。大容量メモリ モデル (c) のすべての RAM ページ ポインタ レジスタにも同じことが言えます。必要に応じて、fastcall16 関数の呼び出しでこれらの値を保存してください。現時点では、CUR_PP ページ ポインタ レジスタだけが変更されています。

DelSigMulti_iGetData

DelSigMulti_cGetData

説明：

変換されたデータを符号付き 8-bit または 16-bit2 の補数形式で返します。注：ユーザ モジュールの DataFormat パラメータが内部表現を決定します。内部表現が符号無しの無しの場合、符号付きの形式関数を呼び出しても、データの値は変りません。データサンプルの準備ができていることを確認するために、DelSigMulti_fIsDataAvailable() が呼び出されることもあります。チャネル番号はこの関数に渡されます。

C プロトタイプ：

```
CHAR DelSigMulti_cGetData(BYTE bChannelNumber) // use for 8-bit resolution or lower
INT  DelSigMulti_iGetData(BYTE bChannelNumber) // use for 9-bit resolution or higher
```

アセンブラ：

```
mov   A, [bChannelNumber]
lcall  DelSigMulti_cGetData      ; Result will be in A
- or -
mov   A, [bChannelNumber]
lcall  DelSigMulti_iGetData      ; LSB will be in A, MSB in X upon return
```

パラメータ：

bChannelNumber: このパラメータは返されるチャネルのデータを指定します。

戻り値：

変換したデータ サンプルを 8-bit または 16-bit の 2 の補数の形で返します。

副作用：

A および X レジスタは、この関数のこの実装または将来の実装で変更される場合があります。大容量メモリ モデル (c) のすべての RAM ページ ポインタ レジスタにも同じことが言えます。必要に応じて、fastcall16 関数の呼び出しでこれらの値を保存してください。現時点では、CUR_PP ページ ポインタ レジスタだけが変更されています。

DelSigMulti_bGetData

DelSigMulti_wGetData

説明：

変換されたデータを 8-bit または 16-bit 符号無し形式で返します。注：ユーザ モジュールの DataFormat パラメータが内部表現を決定します。内部表現が符号無しの無しの場合、符号付きの形式関数を呼び出しても、データの値は変りません。データサンプルの準備ができていることを確認するために、DelSigMulti_fIsDataAvailable() が呼び出されることもあります。チャネル番号はこの関数に渡されます。

C プロトタイプ：

```
BYTE DelSigMulti_bGetData(BYTE bChannelNumber) // use for 8-bit resolution or lower
WORD DelSigMulti_wGetData(BYTE bChannelNumber) // use for 9-bit resolution or higher
```

アセンブラ：

```
mov   A, [bChannelNumber]
lcall  DelSigMulti_bGetData       ; Result will be in A
```

```
- or -
mov   A, [bChannelNumber]
lcall  DelSigMulti_wGetData       ; LSB will be in A, MSB in X upon return
```

パラメータ：

bChannelNumber: このパラメータは返されるチャネルのデータを指定します。

戻り値：

関数に従い、8-bit または 16-bit の変換されたデータ サンプルを符号なしの形式で返します。

副作用：

A および X レジスタは、この関数のこの実装または将来の実装で変更される場合があります。大容量メモリ モデル (c) のすべての RAM ページ ポインタ レジスタにも同じことが言えます。必要に応じて、fastcall16 関数の呼び出しでこれらの値を保存してください。現時点では、CUR_PP ページ ポインタ レジスタだけが変更されています。

DelSigMulti_ClearFlag

説明：

データ利用可能フラグをリセットします。

C プロトタイプ：

```
void  DelSigMulti_ClearFlag(void)
```

アセンブラ：

```
lcall  DelSigMulti_ClearFlag
```

パラメータ：

なし

戻り値：

なし

副作用

A および X レジスタは、この関数のこの実装または将来の実装で変更される場合があります。大容量メモリ モデル (c) のすべての RAM ページ ポインタ レジスタにも同じことが言えます。必要に応じて、fastcall16 関数の呼び出しでこれらの値を保存してください。現時点では、CUR_PP ページ ポインタ レジスタだけが変更されています。

ファームウェア ソースコードの例

この例では前のシナリオを繰り返しますが、API 変数ではなく、グローバル変数への直接リファレンスを使用しています。

アセンブリ言語の例をここに示します。

```
include "DelSigMulti.inc"
include "m8c.inc"

export _main

_main:
   M8C_EnableGInt                 ; enable global interrupts
```

```
   mov   A,DelSigMulti_CH0_HIGHPOWER | DelSigMulti_CH1_HIGHPOWER        ; Establish
power setting...
   call  DelSigMulti_Start              ; and initialize
   call  DelSigMulti_StartAD            ; Commence sampling process
mainloop:
   call  DelSigMulti_fIsDataAvailable   ; Retrieve the status byte
   cmp   A, 0
   jz    mainloop                  ; spin lock until(data is Available)
   mov   A, >RamBuffer
   mov   X, <RamBuffer
   call  DelSigMulti_GetAllDataClearFlag   ; fastcall convention places ADC results in
RAM array defined by RamBuffer pointer
   call  ProcessSample              ; pass the sample to the dummy fcn
   jmp   mainloop


ProcessSample:
   ...                              ; (do something useful with the data)
   ret
```

同じコードを C 言語で示します。

```
#include <m8c.h>        // part specific constants and macros
#include "PSoCAPI.h"    // PSoC API definitions for all User Modules

BYTE * RamBuffer;

void main(void)
{
   M8C_EnableGInt;
   DelSigMulti_Start(DelSigMulti_CH0_HIGHPOWER|DelSigMulti_CH1_HIGHPOWER);
   DelSigMulti_StartAD();
   while(1) {
      if( DelSigMulti_fIsDataAvailable() ) {
DelSigMulti_GetAllDataClearFlag(RamBuffer); // copy ADC data to
// buffer before processing
}
   }
}
```

設定レジスタ

次のセクションでは、このユーザ モジュールで変更されたレジスタについて詳細に説明しています。

アナログレジスタ、1 次変調器

Table 6. 「ADC」アナログ スイッチド キャパシタ PSoC ブロックによって使用されるレジスタ

レジスタ	7	6	5	4	3	2	1	0
ADC_CH0/1/2/3_CR0	1	0	0	1	0	0	0	0
ADC_CH0/1/2/3_CR1	PosInput0/1/2/3			InvertingGain0/1/2/3				
ADC_CH0/1/2/3_CR2	0	1	0	0	0	0	0	0
ADC_CH0/1/2/3_CR3	1	1	1	0	NegInput0/1/2/3		Power (出力)	

[+] Feedback

PosInput0/1/2/3 はシングルエンドの入力信号または差分入力信号の非反転入力を選択します。NegInput0/1/2/3 は、差分信号の反転入力を選択します。InvertingGain0/1/2/3 フィールドがゼロに設定されているとき、反転入力は接続が解除されます。出力は、DelSigMulti_Start と DelSigMulti_SetPowerAPI 関数によって設定されます。0/1/2/3 は、使用されるデルタ - シグマ ADC チャネル数を示します。

アナログレジスタ、2 次変調器

Table 7. 「ADC0」と「ADC1」アナログ スイッチド キャパシタ PSoC ブロックによって使用されるレジスタ

レジスタ	7	6	5	4	3	2	1	0
ADC0_CH0/1/2/3_CR0	1	0	0	0	1	0	0	0
ADC0_CH0/1/2/3_CR1	PosInput0/1/2/3			InvertingGain0/1/2/3				
ADC0_CH0/1/2/3_CR2	0	1	0	0	0	0	0	0
ADC0_CH0/1/2/3_CR3	1	1	1	0	NegInput0/1/2/3		Power (出力)	
ADC1_CH0/1/2/3_CR0	1	0	0	1	0	0	0	0
ADC1_CH0/1/2/3_CR1	LinkToADC0			0	0	0	0	0
ADC1_CH0/1/2/3_CR2	0	0	0	0	0	0	0	0
ADC1_CH0/1/2/3_CR3	1	1	1	0	0	0	Power (出力)	

PosInput0/1/2/3 はシングルエンドの入力信号または差分入力信号の非反転入力を選択します。NegInput0/1/2/3 は、差分信号の反転入力を選択します。InvertingGain0/1/2/3 フィールドがゼロに設定されているとき、反転入力は接続が解除されます。LinktoADC0 は、ブロック配置によって決定し、ADC0 ブロックの出力を ADC1 PSoC ブロックの「A」入力キャパシタに接続します。出力は、DelSigMulti_Start と DelSigMulti_SetPowerAPI 関数によって設定されます。0/1/2/3 は、使用されるデルタ - シグマ ADC チャネル数を示します。

デシメータ制御レジスタ

Table 8. デシメーション制御レジスタ

ビット	7	6	5	4	3	2	1	0
DECx_DH	データの上位バイト							
DECx_DL	データの下位バイト							
DEC_CR0	ACC_IGEN				ICLKS	ACE_IGEN		DCLKS0
DEC_CR1	0	IDEC	ICLKS3	ICLKS2	ICLKS1	DCLKS3	DCLKS2	DCLKS1
DECx_CR0	POL	GOOO	GOOE	0		DATA_IN		

[+] Feedback

ビット	7	6	5	4	3	2	1	0
DEC_CR3	DEC1_EN	CLK_IN1			DEC0_EN	CLK_IN0		
DEC_CR4	DEC3_EN	CLK_IN3			DEC2_EN	CLK_IN2		
DEC_CR5	0				DSCLK			
DECx_CR	モード		データ出力シフト		データフォーマット	デシメーションレート		

デシメータは、Sinc2 フィルタの実装に使用される専用ハードウェアです。3 つの制御レジスタと 2 つのデータ出力レジスタから構成されています。DCol は接続するコラムコンパレータを選択します。DCLKSEL は、デシメータタイミングを制御するデジタルブロックを選択します。どちらのパラメータも、デバイス エディタで設定します。DEC_CR2 のシフトは、ソフトウェアで達成しなければならないアラインされたデータを最小化するために、デシメーションレートに基づいて設定します。また DEC_CR2 でも指定できます。

デジタルブロックレジスタ

Table 9. デジタルブロック制御レジスタ

ビット	7	6	5	4	3	2	1	0
DxCxxCR0								Enable

このレジスタは、関数 DelSigMulti_StartAD. を呼び出すことにより、この UM の生成クロックを開始する追加のデジタルブロックとともに使用されます。また、関数 DelSigMulti _StopAD を呼び出すことにより、生成クロックを停止するためにも使用されます。

バージョン履歴

バージョン	著者	説明
1.2	DHA	デジタル リソースとアナログ リソース間でソース クロックが異なっているかどうかをチェックする DRC を追加。 GetAllDataClearFlagAPI の実現を改良。 クロック選択 DRC を改良。

Note PSoC Designer 5.1 では、すべてのユーザ モジュール Data Sheet にバージョン履歴を導入しました。このセクションには、ユーザ モジュールの現行バージョンと旧バージョンとの相違に関する高度な説明が記載されています。

Copyright © 2009-2011 Cypress Semiconductor Corporation. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC Designer ™ and Programmable System-on-Chip ™ are trademarks and PSoC® is a registered trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.