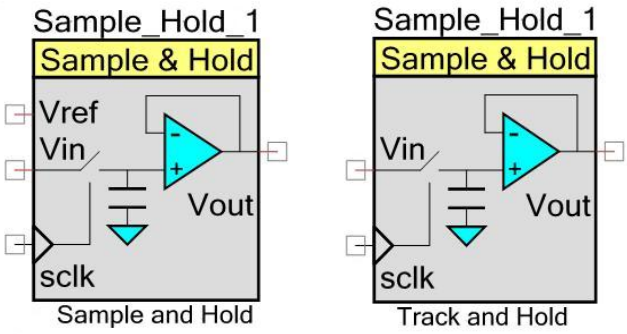
サンプルホールド、トラックホールドコンポーネント

1.10

特長

- 2 つの動作モード: サンプル & ホールド、トラック & ホールド
- 4 つの駆動電力モード設定



概要

サンプルホールド、トラックホールドコンポーネントは、連続的に変化するアナログ信号をサンプリングして、ある一定期間保持する手段を提供します。トラックおよびホールド、ホールドおよびサンプルの両方の関数をサポートしています。Configure ダイアログで選択することができます。

入出力の接続

ここでは、サンプルホールド、トラックホールドコンポーネントのさまざまな入力および出力接続について説明します。I/O 項目のアスタリスク (*) はその I/O が、説明に挙げられた条件において、回路シンボルに表示されない場合があることを示します。

Vin – アナログ

Vin 端子は、サンプルホールド、トラックホールドコンポーネントの入力です。サンプリングまたは追従させたいアナログ信号を接続します。

Vout – アナログ

Vout 端子は、サンプルホールド、トラックホールドコンポーネントの出力です。この信号はピンやアナログ入力 (例えばコンパレータ、ADC) に配線することができます。

SCLK – 入力 *

SCLK 入力は、サンプルホールド、トラックホールドコンポーネントのサンプリングクロック入力を定義します。

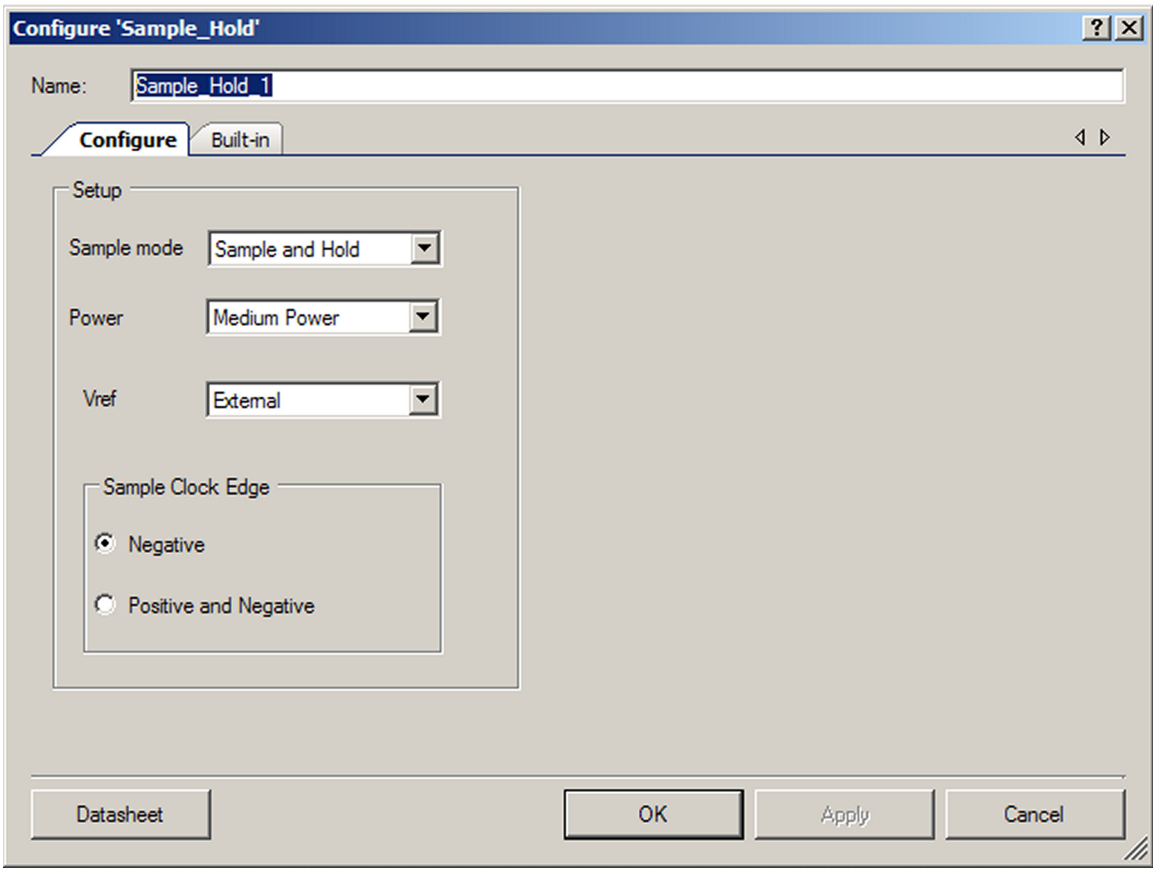
Vref – 入力 *

Vref 入力はオプションの入力で、**Sample mode** パラメータで選択されます。

- **Sample mode** が **Sample and Hold** で、**Vref** が **External** の場合このピンは表示され、有効な Vref 信号源に接続されます。
- **Sample mode** が **Track and Hold** の場合は、このピンがシンボルから消えます。

パラメータおよび設定

サンプルホールド、トラックホールドコンポーネントを回路図の上にドラッグし、ダブルクリックして **Configure** ダイアログを開きます。



サンプルホールド、トラックホールドコンポーネントには以下のパラメータがあります。

Sample mode

Sample and Hold 設定は、クロックのエッジで信号をサンプリングします。エッジは立ち下がりエッジまたは両方のエッジを選択します。

Track and Hold 設定は、サンプルクロックの立ち下がりエッジの信号をサンプルしますが、サンプルクロックがローレベルの期間は入力信号に追従します。

Power

このパラメータは、サンプルホールド、トラックホールドコンポーネントの駆動電力の初期値を設定します。駆動電力は、サンプルホールド、トラックホールドコンポーネントが入力信号の変化に応答する速度を決定します。以下 4 つの電力設定があります。**Minimum Power**(最小)、**Low Power**(低)、**Medium Power** (中、初期設定)、**High Power**(高)。**Minimum Power** 設定は、最も遅い応答時間になり、**High Power** 設定は最も速い応答時間になります。

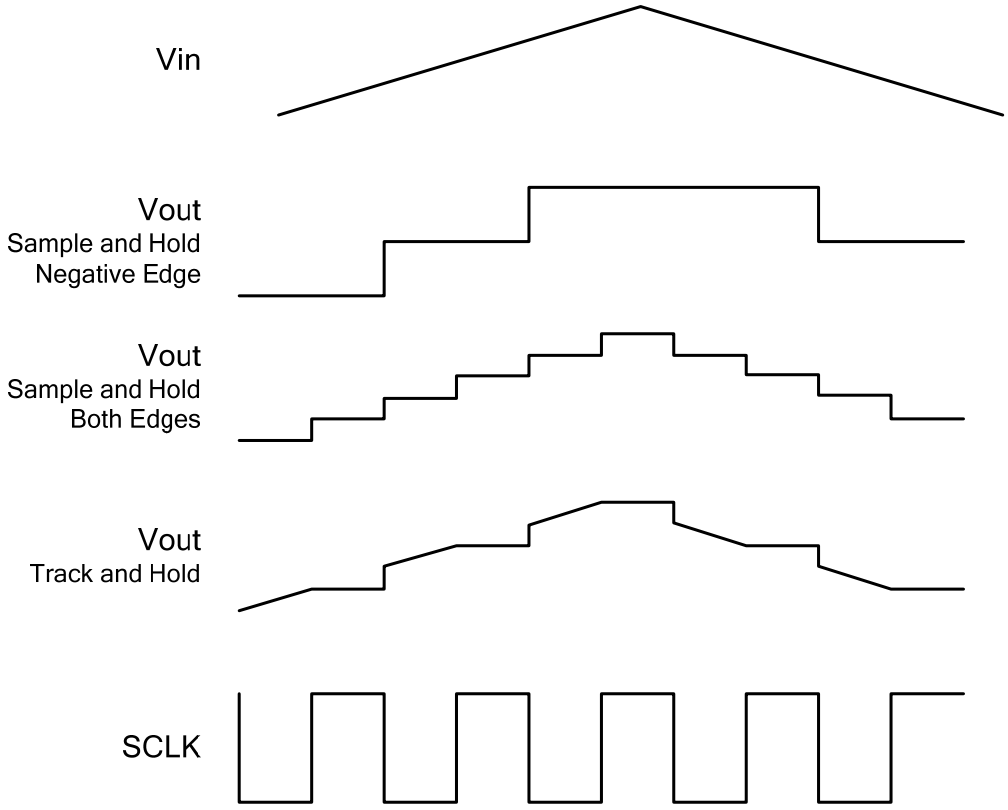
Vref

Vref モードは、**Internal**(内部) または **External**(外部) から基準電圧を選択するのに使用されます。**Vref** が **External** の場合、外部基準電圧が サンプルホールド、トラックホールドコンポーネントに適用されます。Vref モードが **Internal** の場合、コンポーネントは内部ソース Vss (増幅器の基準電位を与えるコンポーネントの内部グランド信号) から基準電圧ををを取得します。

Sample Clock Edge

このパラメータはクロックエッジを設定します。このパラメータは、 Sample and Hold モードでのみ有効です。エッジ設定には次の 2 種類があります。**Negative** (立ち下がり) 、**Positive and Negative**(立ち上がりおよび立ち下がり)。

図 1. サンプルホールド、トラックホールドコンポーネントの波形



リソース

サンプルホールド、トラックホールドコンポーネントは 1 つの SC/CT ブロックを使用します。

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。続くセクションでは、各関数について詳しく説明します。

初期設定では、PSoC Creator は、ユーザの回路図に最初に配置されたコンポーネントのインスタンス名として"Sample_Hold_1"を割り当てます。インスタンスの名称は、識別子の文法ルールに従って固有の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、定数名の接頭辞になります。便宜上、以下の表では"Sample_Hold"というインスタンス名を使用します。

関数	機能
Sample_Hold_Start()	サンプルホールド、トラックホールドコンポーネントを設定して電源を有効にします。
Sample_Hold_Stop()	サンプルホールド、トラックホールドコンポーネントブロックをオフにします。
Sample_Hold_SetPower()	サンプルホールド、トラックホールドコンポーネントの駆動電力を設定します。
Sample_Hold_Sleep()	サンプルホールド、トラックホールドコンポーネントをスリープモードにします。
Sample_Hold_Wakeup()	サンプルホールド、トラックホールドコンポーネントをスリープモードから復帰させます。
Sample_Hold_Init()	サンプルホールド、トラックホールドコンポーネントを初期化します。
Sample_Hold_Enable()	ハードウェアを起動し、コンポーネントの動作を開始します。
Sample_Hold_SaveConfig()	空の関数。将来使用するために予約されています。
Sample_Hold_RestoreConfig()	空の関数。将来使用するために予約されています。

void Sample_Hold_Start(void)

機能: コンポーネントの初期化に必要な処理を実行し、ブロックに電源を供給します。最初にこのルーチンが実行される時は、サンプル モード、クロックエッジ、駆動電力が各々の初期値に設定されます。Sample_Hold_Stop() 呼び出し後、ミキサの再起動のためにこの関数を呼び出すと、現在のコンポーネントパラメータ設定が保持されます。

パラメータ: なし

返り値: なし

注意事項: なし

void Sample_Hold_Stop(void)

機能: サンプルホールド、トラックホールドブロックをオフにします。

パラメータ: なし

返り値: なし

注意事項: Sample and Hold モードと駆動電力には影響しません。

void Sample_Hold_SetPower(uint8 power)

機能: 駆動電力を 4つのレベル (最小、低、中、高) のいずれかに設定します。

パラメータ: uint8 power: コンポーネントに駆動電力を設定します。設定は以下の表を参照してください。

パラメータ	電力設定
Sample_Hold_MINPOWER	最低の駆動電力と最も遅い応答時間
Sample_Hold_LOWPOWER	低駆動電力と速度
Sample_Hold_MEDPOWER	中駆動電力と速度
Sample_Hold_HIGHPOWER	最高の駆動電力と最も速い応答時間

返り値: なし

注意事項: なし

void Sample_Hold_Sleep(void)

機能: これは、コンポーネントのスリープを準備するのに推奨される API です。Sample_Hold_Sleep() API は、現在のコンポーネントの状態を保存します。次に、Sample_Hold_Stop() 関数を呼び出し、Sample_Hold_SaveConfig() を呼び出してハードウェア設定を保存します。

CyPmSleep() および CyPmHibernate() 関数を呼び出す前に、Sample_Hold_Sleep() 関数を呼び出してください、。

パラメータ: なし

返り値: なし

注意事項: なし

void Sample_Hold_Wakeup(void)

機能: これは、コンポーネントを Sample_Hold_Sleep() が呼び出されたときの状態に復元するための推奨される API です。Sample_Hold_Wakeup() 関数は Sample_Hold_RestoreConfig() 関数を呼び出して、設定を復元します。Sample_Hold_Sleep() 関数が呼び出される前にコンポーネントが有効になっている場合、Sample_Hold_Wakeup() 関数もコンポーネントを再び有効にします。

パラメータ: なし

返り値: なし

注意事項: あらかじめ Sample_Hold_Sleep() または Sample_Hold_SaveConfig() 関数を呼び出すことなく Sample_Hold_Wakeup() 関数を呼び出すと、予期しない動作をする可能性があります。

void Sample_Hold_Init(void)

機能: Configure 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。Sample_Hold_Start() API が Sample_Hold_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動作を開始する際に推奨される方法です。

パラメータ: なし

返り値: なし

注意事項: 全レジスタは、Configure ダイアログの設定に従って、値が設定されます。

void Sample_Hold_Enable(void)

機能: 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。Sample_Hold_Start() API が Sample_Hold_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動作を開始する際に推奨される方法です。

パラメータ: なし

返り値: なし

注意事項: なし

void Sample_Hold_SaveConfig(void)

機能: 空の関数。将来使用するために予約されています。

パラメータ: なし

返り値: なし

注意事項: なし

void Sample_Hold_RestoreConfig(void)

機能: 空の関数。将来使用するために予約されています。

パラメータ: なし

返り値: なし

注意事項: なし

ファームウェアソースコードのサンプル

PSoC Creator は、Find Example Project ダイアログに、回路図およびサンプルコードを含む多くのサンプルプロジェクトを提供しています。コンポーネント特有のサンプルを見るには、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。一般的なサンプルについては、Start Page または **File** 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 を参照してください。

ユーザレジスタ

API が、ほとんどの用途に必要とされる機能を提供しています。次のレジスタリファレンスに上級者向けの簡単な説明を記します。

表 1. SCx_CR0

ビット	7	6	5	4	3	2	1	0
値	RSVD		dft		mode			適用なし

- mode: SC ブロックの設定モード。

表 2. SCx_CR1

ビット	7	6	5	4	3	2	1	0
値	RSVD		gain	div2	comp		drive	

- gain: S/H Mixer モードと PGA モードの帰還容量比率を制御します。
- div2: 0 の場合、サンプルクロックは S/H Mixer モードの目的のサンプル周波数の 1/2 にする必要があります。
- Comp[1:0]: 補償用コンデンササイズを選択します。
- Drive[1:0]: 出力バッファの電流設定を選択します。

表 3. SCx_CR2

ビット	7	6	5	4	3	2	1	0
値	gndvref	rval			redc		R20_40b	bias_cntl

- gndvref: 反転入力のグランド直接接続を有効にします。
- Rval[2:0]: 帰還レジスタ コントロール。
- Redc[1:0]: 出力段と初段間の静電容量を調整します。
- r20_40b: 入力インピーダンス: 20k または 40k
- bias_cntl: バイアスコントロール、標準、½ (低)

DC/ AC電気的特性

5.0V/3.3V DC/ AC電気的特性

項目	条件および注記	Min	Typ	Max	単位
入力オフセット電圧		–	–	–	mV
無信号時電流	Minimum パワー	–	–	–	mA

	Low パワー	–	–	–	mA
	Medium パワー	–	–	–	mA
	High パワー	–	–	–	mA
fsclk、サンプルクロック周波数	ダウンミキサモード	–	–	4	MHz
fvin、入力信号周波数	ダウンミキサモード	–	–	14	MHz
SR					
スルーレート		–	–	3	V/µs
アクイジション時間	A 5.5V、1% ステップ	–	–	1	µs
ドループレート		–	–	–	mV/ms

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
1.10	データシートの変更: ■ 上記リソース セクションに Sample/Track and Hold 波形を追加 ■ トラック およびホールド モードのコンポーネント シンボルを追加 ■ 多少の変更と更新	
	内部 Vref 問題を修正しました。Sample_Hold_Init() API 内の C コードの変更更新。	サンプル/トラックおよびホールド モード両方に内部 Vref 問題が存在していた。

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。