



AN75400 – PSoC® 3 および PSoC® 5LP

CapSense®デザイン ガイド

文書番号 001-80490 Rev. *B

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
japan.cypress.com

© Cypress Semiconductor Corporation, 2012-2020. 本書面は、Cypress Semiconductor Corporation 及び Spansion LLC を含むその子会社（以下「Cypress」という。）に帰属する財産である。本書面（本書面に含まれ又は言及されているあらゆるソフトウェア若しくはファームウェア（以下「本ソフトウェア」という。）を含む）は、アメリカ合衆国及び世界のその他の国における知的財産法令及び条約に基づき Cypress が所有する。Cypress はこれらの法令及び条約に基づく全ての権利を留保し、本段落で特に記載されているものを除き、その特許権、著作権、商標権又はその他の知的財産権のライセンスを一切許諾しない。本ソフトウェアにライセンス契約書が伴っておらず、かつ Cypress との間で別途本ソフトウェアの使用方法を定める書面による合意がない場合、Cypress は、(1) 本ソフトウェアの著作権に基づき、(a) ソースコード形式で提供されている本ソフトウェアについて、Cypress ハードウェア製品と共に用いるためにのみ、かつ組織内部でのみ、本ソフトウェアの修正及び複製を行うこと、並びに (b) Cypress のハードウェア製品ユニットに用いるためにのみ、（直接又は再販売者及び販売代理店を介して間接のいずれかで）本ソフトウェアをバイナリーコード形式で外部エンドユーザーに配布すること、並びに (2) 本ソフトウェア（Cypress により提供され、修正がなされていないもの）が抵触する Cypress の特許権のクレームに基づき、Cypress ハードウェア製品と共に用いるためにのみ、本ソフトウェアの作成、利用、配布及び輸入を行うことについての非独占的で譲渡不能な一身専属的ライセンス（サブライセンスの権利を除く）を付与する。本ソフトウェアのその他の使用、複製、修正、変換又はコンパイルを禁止する。

適用される法律により許される範囲内で、Cypress は、本書面又はいかなる本ソフトウェア若しくはこれに伴うハードウェアに関しても、明示又は黙示をとわず、いかなる保証（商品性及び特定の目的への適合性の黙示の保証を含むがこれらに限られない）も行わない。いかなるコンピューティングデバイスも絶対に安全ということはない。従って、Cypress のハードウェアまたはソフトウェア製品に講じられたセキュリティ対策にもかかわらず、Cypress は、Cypress 製品への権限のないアクセスまたは使用といったセキュリティ違反から生じる一切の責任を負わない。加えて、本書面に記載された製品には、エラッタと呼ばれる設計上の欠陥またはエラーが含まれている可能性があり、公表された仕様とは異なる動作をする場合がある。適用される法律により許される範囲内で、Cypress は、別途通知することなく、本書面を変更する権利を留保する。Cypress は、本書面に記載のある、いかなる製品若しくは回路の適用又は使用から生じる一切の責任を負わない。本書面で提供されたあらゆる情報（あらゆるサンプルデザイン情報又はプログラムコードを含む）は、参照目的のためのみに提供されたものである。この情報で構成するあらゆるアプリケーション及びその結果としてのあらゆる製品の機能性及び安全性を適切に設計、プログラム、かつテストすることは、本書面のユーザーの責任において行われるものとする。Cypress 製品は、兵器、兵器システム、原子力施設、生命維持装置若しくは生命維持システム、蘇生の設備及び外科的移植を含むその他の医療機器若しくは医療システム、汚染管理若しくは有害物質管理の運用のために設計され若しくは意図されたシステムの重要な構成部分としての使用、又は装置若しくはシステムの不具合が人身傷害、死亡若しくは物的損害を生じさせるようなその他の使用（以下「本目的外使用」という。）のためには設計、意図又は承認されていない。重要な構成部分とは、その不具合が装置若しくはシステムの不具合を生じさせるか又はその安全性若しくは実効性に影響すると合理的に予想できるような装置若しくはシステムのあらゆる構成部分をいう。Cypress 製品のあらゆる本目的外使用から生じ、若しくは本目的外使用に関連するいかなる請求、損害又はその他の責任についても、Cypress はその全部又は一部をとわず一切の責任を負わず、かつ Cypress はそれら一切から本書により免除される。Cypress は Cypress 製品の本来目的外使用から生じ又は本目的外使用に関連するあらゆる請求、費用、損害及びその他の責任（人身傷害又は死亡に基づく請求を含む）から免責補償される。

Cypress、Cypress のロゴ、Spansion、Spansion のロゴ及びこれらの組み合わせ、WICED、PSoC、CapSense、EZ-USB、F-RAM、及び Traveo は、米国及びその他の国における Cypress の商標又は登録商標である。Cypress のより完全な商標のリストは、cypress.com を参照すること。その他の名称及びブランドは、それぞれの権利者の財産として権利主張がなされている可能性がある。

目次



1. はじめに	6
1.1 概要	6
1.2 サイプレス CapSense の文書体系	6
1.3 PSoC 3 および PSoC 5LP デバイスの特長	9
1.4 本書の表記法	10
2. CapSense 技術	11
2.1 CapSense の原理	11
2.2 静電容量の変換	12
2.2.1 CapSense シグマ デルタ (CSD)	12
2.2.2 CSD の実装	14
2.3 SmartSense 自動チューニング	14
3. PSoC 3 および PSoC 5LP における CapSense	15
3.1 CSD の実装	15
3.2 CapSense 独自の特長	16
3.2.1 2 チャネル デザイン	16
3.2.2 耐水性デザイン	16
3.2.3 電流ソース方式	18
3.2.4 チューニング	21
3.2.5 ノンブロッキング アーキテクチャ	22
3.3 CapSense PLUS	23
4. CapSense 設計ツール	24
4.1 PSoC Creator	24
4.2 ハードウェア キット	25
4.2.1 PSoC 3 および PSoC 5LP 開発キット	25
4.2.2 ユニバーサル CapSense モジュール基板	25
4.2.3 PSoC CapSense 拡張基板キット	25
5. CapSense_CSD コンポーネント	26
5.1 パラメーターのまとめ	27
5.2 グローバル アレイ	29
5.2.1 raw カウント	29
5.2.2 ベースライン	29

5.2.3	差分カウント	30
5.2.4	センサー状態	30
5.3	高レベル パラメーター	30
5.3.1	指閾値	31
5.3.2	ヒステリシス	31
5.3.3	デバウンス	31
5.3.4	ノイズ閾値	31
5.3.5	負のノイズ閾値および低ベースライン リセット	32
5.3.6	センサー自動リセット	33
5.3.7	ウィジェット分解能	34
5.3.8	フィルタ セクション	35
5.3.9	高レベル パラメーターの推奨事項	36
5.4	低レベル パラメーター	36
5.4.1	クロック設定	37
5.4.2	アナログ スイッチ分周器	39
5.4.3	疑似乱数シーケンス (PRS)	42
5.4.4	スキャン分解能	43
5.4.5	IDAC 電流	44
5.4.6	スキャン速度	45
5.4.7	デジタル リソースの実装	47
5.4.8	電流ソース	48
5.4.9	電圧基準ソース	50
5.4.10	シールド電極およびガード センサー	51
5.5	ウィジェットのコンフィギュレーション	54
5.5.1	ウィジェットの追加	54
5.5.2	センサー エLEMENT数	54
5.5.3	API 分解能	54
5.5.4	近接センサー	55
5.6	チューニング方法	57
5.6.1	感度	57
5.7	チャンネル数	58
5.7.1	チャンネル 1 / チャンネル 0 への変更	59
5.7.2	2 チャンネル デザインのピン割り当て	60
5.8	チューナー GUI	61
5.8.1	セットアップ	61
6.	CapSense 性能のチューニング	66
6.1	基本事項	66
6.1.1	信号、ノイズおよび SNR	66
6.1.2	充電 / 放電速度	67
6.2	手動チューニング	68
6.3	SmartSense 自動チューニング	69
6.3.1	ガイドライン	70
6.3.2	パラメーター設定	70

7. 設計上の注意事項	71
7.1 ソフトウェア フィルタリング	71
7.2 消費電力	72
7.2.1 スリープ スキャン方式	73
7.2.2 平均消費電力の測定	74
7.3 応答時間	76
7.3.1 スリープ スキャン モード	76
7.3.2 長いバックグラウンド ループ	76
7.3.3 デバウンス	77
7.3.4 フィルタによる遅延	79
7.3.5 割り込み優先順位	80
7.4 ピン割り当て	80
7.4.1 オペアンプ出力ピン	80
7.4.2 2 チャネル デザインのピン割り当て	80
7.4.3 C _{MOD} ピン割り当て	80
7.5 プリント基板レイアウトのガイドライン	80
8. リソース	81
8.1 ウェブサイト	81
8.2 データシート	81
8.3 テクニカル リファレンス マニュアル	81
8.4 開発キット	81
8.4.1 PSoC 3 および PSoC 5LP 開発キット	81
8.4.2 モジュール ボードを開発キットに接続するインターフェース ボード	81
8.4.3 ユニバーサル CapSense モジュール基板	81
8.4.4 MiniProg3	82
8.5 PSoC Programmer	82
8.6 Multi-Chart	82
8.7 PSoC Creator	82
8.8 サンプル コード	82
8.9 デザイン サポート	83
用語集	84
改版履歴	90
改訂履歴	90

1. はじめに



1.1 概要

本書は、PSoC®3 および PSoC® 5LP ファミリのデバイスを使用した CapSense®設計のガイドを提供します。これは、PSoC 3 および PSoC 5LP ファミリのデバイスを使用してアプリケーションを設計する静電容量センシング技術に精通しているエンジニアの方を対象としています。CapSense 技術の完全な理解については「[Getting Started with CapSense](#)」を参照してください。

1.2 サイプレス CapSense の文書体系

図 1-1 および表 1-1 は、サイプレスの CapSense 文書体系の要約です。これらのリソースにより、CapSense 製品設計を完了する際に必要な情報に迅速にアクセスできます。図 1-1 は、静電容量センシングの標準的な製品設計サイクルを示したものです。本ガイドは緑色でハイライト表示した項目を説明します。表 1-1 には、図 1-1 で番号付けされた各タスクをサポートする文書へのリンクが記載されています。

図 1-1. 標準的な CapSense の製品設計フロー

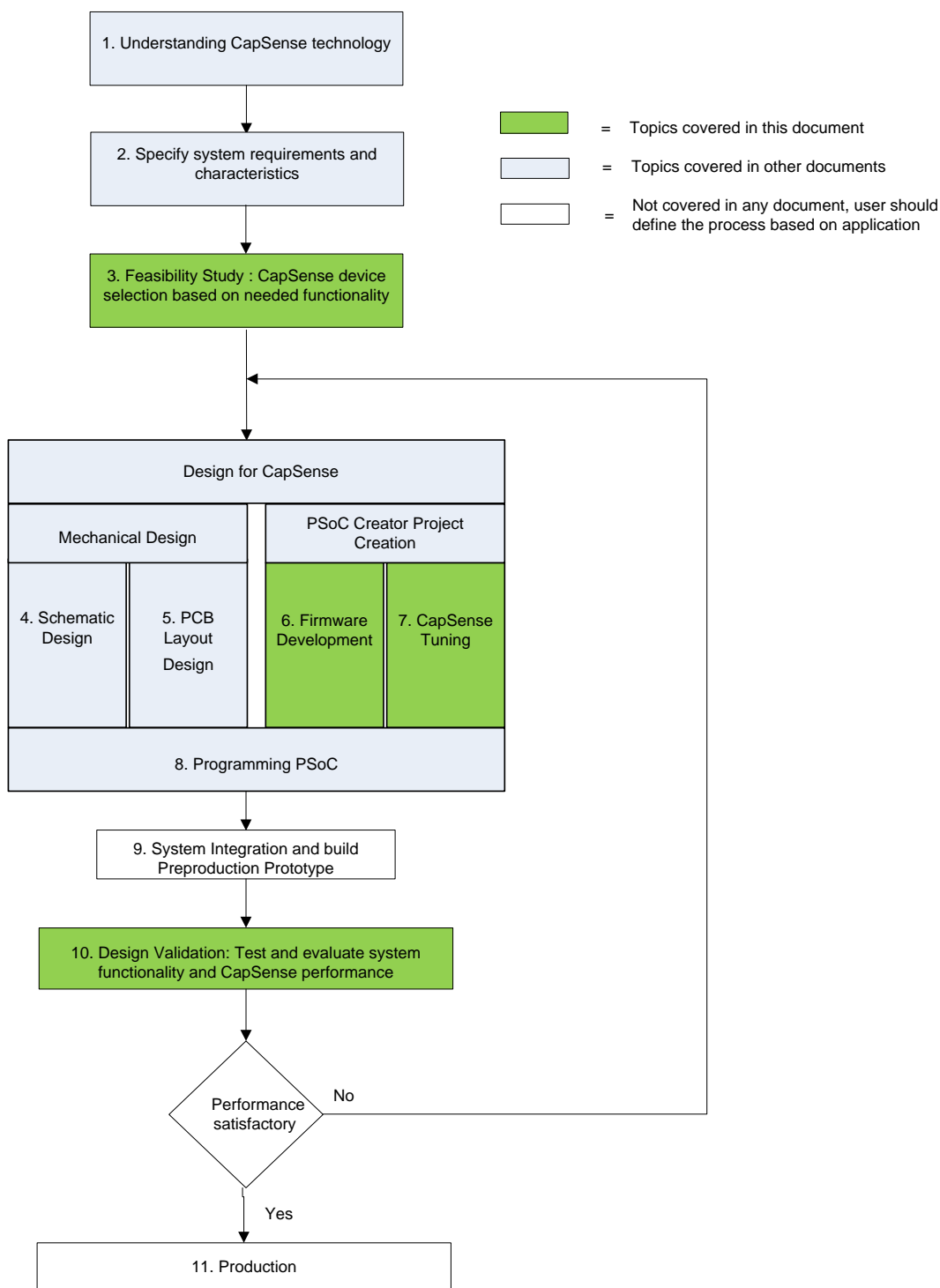


表 1-1. 図 1-1 中に番号付けされた設計タスクをサポートしているサイプレスの文書

番号付けされた設計タスク	サイプレス CapSense 用サポートドキュメント		
	文書名	セクション	説明
1	Getting Started with CapSense	2	CapSense 動作の詳細な理論
2	Getting Started with CapSense	4、5	デバイスの特長を説明し、デバイスを選択する手助けをする
	PSoC 3 Device Datasheets PSoC 5LP Device Datasheets	-	
3	Getting Started with CapSense	4、5	デバイスの特長を説明し、デバイスを選択する手助けをする
	本書	3	
4	Getting Started with CapSense	3	C _{MOD} にのピン割り当ておよび値などの情報を提供
	本書	7	
5	Getting Started with CapSense	3	PCB レイアウト ガイドラインを提供
6	PSoC Creator ヘルプ トピック (PSoC Creator IDE の Help タブにある)	-	PSoC Creator ヘルプ トピックは PSoC Creator IDE を使用するガイドラインを提供
	本書	4、5、6、7	本書および CapSense_CSD コンポーネントデータシートは CapSense アプリケーションを開発するファームウェア ガイドラインを提供
	CapSense_CSD コンポーネントデータシート	Application Programming Interface	
7	本書	6	CapSense システムを調整する方法を説明
8	MiniProg3 User Guide	-	PSoC 3 および PSoC 5LP のプログラミングを詳しく説明
	AN61290 - PSoC 3/PSoC 5LP Hardware Design Considerations	Programming and Debugging	
9	該当なし	-	-
10	Getting Started with CapSense	3	設計上の重要な注意事項
	本書	6、7	
11	該当なし	-	-

1.3 PSoC 3 および PSoC 5LP デバイスの特長

PSoC 3 および PSoC 5LP は、コンフィギュレーション可能なアナログおよびデジタル ペリフェラル機能、メモリ、マイクロコントローラーを搭載したプログラマブル組込みシステムオンチップです。これらのデバイスは非常に柔軟性が高く、CapSense の他に多くの機能を実装することができます。主な特長は次のとおりです。

デバイスの特長

- PSoC 3: 67MHz 8051 CPU
- PSoC 5LP: 67MHz Arm Cortex-M3 CPU
- PSoC 3: 最大 64KB フラッシュ、8KB SRAM、2KB EEPROM
- PSoC 5LP: 最大 256KB フラッシュ、64KB SRAM、2KB EEPROM
- 24 チャンネル DMA
- 最大 72 の I/O ピン
- 低消費電力モード
 - PSoC 3: 1 μ A のスリープ モード電流
 - PSoC 5LP: 2 μ A のスリープ モード電流
 - PSoC 3: 200nA のハイバネート モード電流
 - PSoC 5LP: 300nA のハイバネート モード電流
- アナログ機能
 - 分解能が 8~20 ビットのコンフィギュレーション可能なデルタ シグマ ADC
 - 最大 4 個のコンパレータ、オペアンプ、DAC、多機能アナログ ブロック
 - 0.1%内部バンドギャップ基準電圧
- デジタル機能
 - 最大 4 個の 16 ビットのコンフィギュレーション可能なタイマー、カウンタ、PWM ブロック
 - 最大 24 個のプログラマブル ロジック デバイス (PLD) ベースのユニバーサル デジタル ブロック (UDB)
- 多種のパッケージ: QFN、SSOP、TQFP
- 任意の GPIO で最大 46x16 セグメントの LCD を直接駆動
- I²C、UART、フル スピード USB、SPI、CAN 通信インターフェース

CapSense の特長

- CapSense のボタン、スライダ、マトリクス ボタンおよび近接センサーの組み合わせに対応
- 最大 61 個の静電容量センサーをサポートし、すべての GPIO ピンで CapSense をサポート
- ファームウェアを開発する API を統合
- 防水設計をサポート
- 2 チャンネル デザイン: 2 個のセンサーを同時にスキャンするのに十分なリソース
- SmartSense™ 自動チューニング
 - 電源投入時と実行中のチューニング パラメーターを自動的に設定および監視
 - デザインの移植性のために、ユーザー インターフェース設計の変化に適応
 - 実行時の環境の変化を補正
 - 最小 0.1pF の静電容量でタッチを検知

1.4 本書の表記法

表記法	説明
Courier New フォント	ファイルの場所、ユーザーが入力したテキスト、ソース コードを表示 C:\...cd\icc\
イタリック フォント	ファイル名および参考資料を表示 <i>PSoC Designer User Guide</i> にある <i>sourcefile.hex</i> ファイルを参照してください。
[角括弧、太字]	手順としてキーボードのコマンドを示す [Enter] または [Ctrl] [C]
File > Open	メニュー パスを表示: File > Open > New Project
太字	操作手順でのコマンド、メニューパス、アイコン名を表示: File アイコンをクリックして、 Open をクリック
Times New Roman フォント	式を表示: 2+2=4
灰色のボックス内のテキスト	製品の注意点やユニークな機能を説明

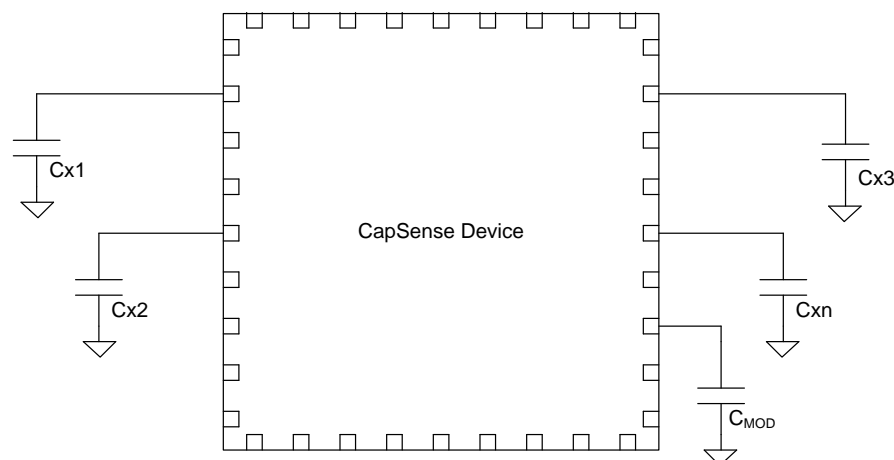
2. CapSense 技術



2.1 CapSense の原理

CapSense は、センサーとして指定された各 I/O ピンの静電容量を計測することによって機能するタッチ センシング技術です。図 2-1 に示すように、各センサー ピンの総合静電容量は、Cx1～Cxn の値を持つ等価集中コンデンサとしてモデル化できます。CapSense 技術は外部変調コンデンサ C_{MOD}を必要とします。C_{MOD}の詳細は「[静電容量の変換](#)」を参照してください。

図 2-1. CapSense デバイス スキャンング センサーCx1～Cxn



各センサーの I/O ピンは、配線、ビアまたは必要に応じてその両方でセンサー パッドに接続されます。非導電性のオーバーレイがセンサー パッドを覆うのに必要であり、システムのタッチ インターフェースを構成しています。指がオーバーレイに接触すると、人体の伝導性と大きさにより、グランドに接続された導体板がセンサー パッドと平行に置かれるのと同じ状況になります。これを図 2-2 に示します。この配置は平行板コンデンサを構成し、その静電容量は以下の式によって与えられます。

$$C_F = \frac{\epsilon_0 \epsilon_r A}{D}$$

式 1

ここで、

C_F = センサーを覆うオーバーレイに接触する指により生じた静電容量

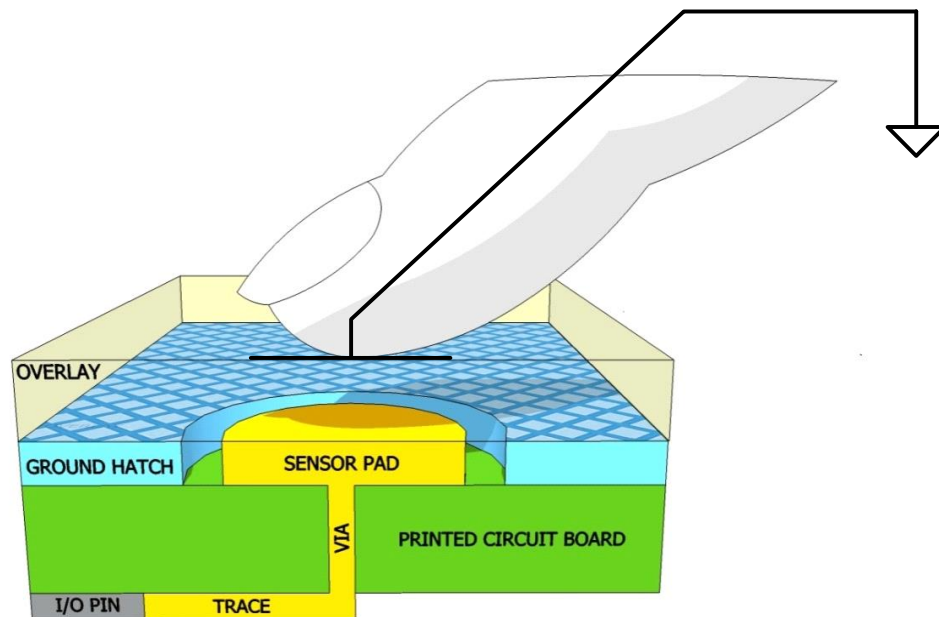
ε₀ = 真空の誘電率

ε_r = オーバーレイの比誘電率

A = 指とセンサー パッドが重なっている面積

D = オーバーレイの厚さ

図 2-2. CapSense システム等価モデル



指がオーバーレイに接触しなくても、センサーの入力ピンには寄生容量 (C_P) があります。 C_P は、CapSense コントローラー内部の寄生容量と電界の組み合わせの結果です。この電界は、センサー パッド、配線、ビアおよびシステム内の他の導体（グラウンド面、他の配線、製品のシャーシまたは封入物内の金属等）間のカップリングにより生じた電界です。 C_P および C_F はお互いに並列です。これは、両者がセンサー ピンとグラウンドの間に接続されているためです。

指がセンサーに触れていない場合

$$C_X = C_P \quad \text{式 2}$$

指がセンサーに触れている場合

$$C_X = C_P + C_F \quad \text{式 3}$$

一般的に、 C_P は C_F より何十倍か大きい値です。 C_P は通常 6pF～15pF の範囲ですが、極端な場合は 45pF まで高くなることもあります。 C_F は通常 0.1pF～0.4pF の範囲です。 C_P の大きさは、CapSense システムのチューニング時に非常に重要です。最適な性能を得るには、 C_P をできる限り低くする必要があります。

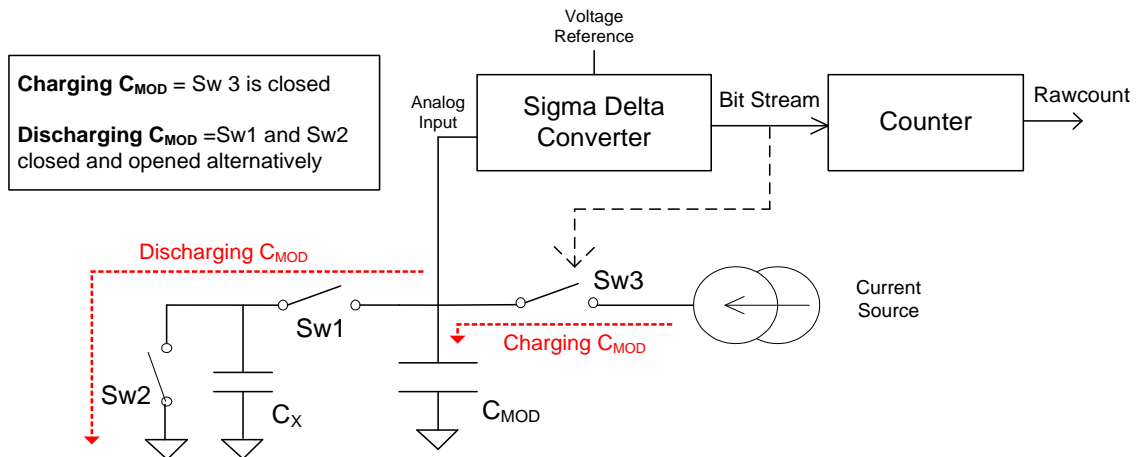
2.2 静電容量の変換

CapSense デバイスは、各 C_X の大きさをデジタル カウントに変換します。このデジタル カウントは、センサー パッド上またはその近くの指の存在を検出するために保存および処理されます。

2.2.1 CapSense シグマ デルタ (CSD)

CapSense シグマ デルタ (CSD) はセンサーの静電容量をデジタル カウントに変換する方式の 1 つです。CSD 方式は他のセンシング方式を凌ぎます。図 2-3 は CSD 方式のブロック図を示します。

図 2-3. CapSense CSD ブロック図



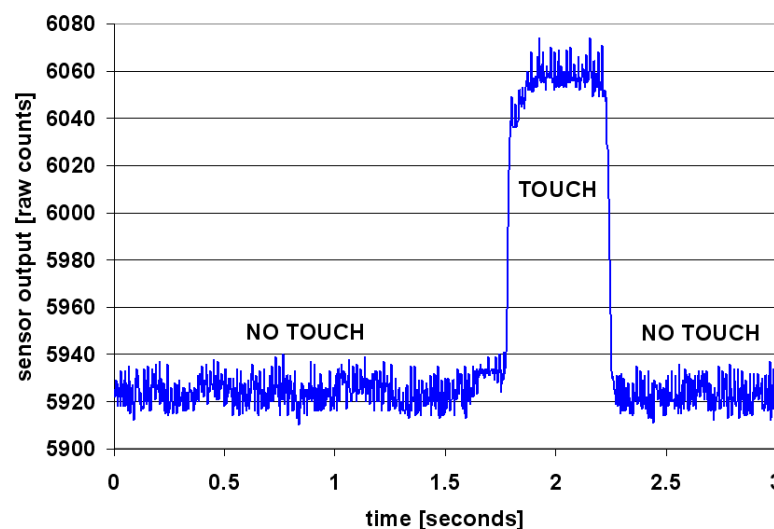
CSD 方式には、変調コンデンサ(C_{MOD}) と呼ばれる大きな積分コンデンサが必要です。センサー コンデンサ C_X は、スイッチ $Sw1$ と $Sw2$ に接続され、スイッチト コンデンサ ブロックを形成します。 $Sw1$ は C_X を C_{MOD} に、 $Sw2$ は C_X をグラウンドに接続します。 $Sw1$ と $Sw2$ は交互に開いたり、閉じたりし (同時に閉じない)、 C_{MOD} の放電パスを提供します。

シグマ デルタ コンバータは、 C_X の静電容量をデジタル カウントに変換します。定電流ソースはスイッチ $Sw3$ を介して C_{MOD} に接続されます。 $Sw3$ が閉じると、電流ソースは C_{MOD} を充電します。 C_{MOD} は入力としてシグマ デルタ コンバータに接続されます。シグマデルタ コンバータはその入力に応じて、 $Sw3$ を制御して C_{MOD} の平均電圧を基準電圧に維持するようにします。シグマ デルタ コントローラーの出力は、 $Sw3$ のデューティ サイクルを示すビット ストリームです。

$Sw3$ のデューティ サイクルは C_X の静電容量に正比例します。たとえば、 C_X の値が高くなると、 C_{MOD} の放電電流が増加します。基準電圧で C_{MOD} 電圧を維持するように、シグマ デルタ コンバータは $Sw3$ のデューティ サイクルを増加します。

デジタル値のビット ストリームは、raw カウントとして知られます。raw カウントは高レベルのアルゴリズムによって解釈され、センサーの状態を判断します。指がセンサーに触れると、 C_X が C_F の分増加し、raw カウントがそれに比例して増加します。定常状態の raw カウントの動きと事前に設定された閾値を比較することで、センサーがオン (接触) 状態であるかオフ (非接触) 状態であるかを高レベル アルゴリズムにより判定できます。図 2-4 は、センサーを指で触れてから離すまで連続スキャンした raw カウントのプロットを示します。

図 2-4. raw カウントと時間



サイプレスの CSD センシング方式の詳しい説明は「[PSoC 3, PSoC 5LP Architecture TRM](#)」を参照してください。

2.2.2 CSD の実装

CSD 方式を実装する方法は多くあります。

IDAC ソース方式: 図 2-3 に示すように、IDAC は C_{MOD} を充電し、センサーは C_{MOD} を放電します。

IDAC シンク方式: IDAC は C_{MOD} を放電し、センサーは C_{MOD} を充電します。

外部抵抗方式: 外部抵抗は C_{MOD} を放電し、センサーは C_{MOD} を充電します。

上記の CSD 方式の実装方法は「[電流ソース方式](#)」に記載されています。

2.3 SmartSense 自動チューニング

CapSense システムを構成するハードウェアとファームウェアには、CapSense システムの動作を決めるいくつかのパラメーターがあります。これらのパラメーターのチューニングは、適切なシステムの動作と快適なユーザーの使用体験のために重要です。残念ながら、チューニングは反復プロセスなので時間がかかります。一般的な開発サイクルでは、インターフェースは初期の設計段階、システム統合中および増産前にチューニングされます。

SmartSense 自動チューニングはサイプレスの先進技術です。SmartSense は、広範囲のアプリケーションでシステム性能を自動的に最適化する洗練されたアルゴリズムです。それは使い易く、試作や製造段階で手動チューニングを無くすことで、設計のサイクル時間を短縮します。SmartSense 自動チューニングは電源投入時に各 CapSense ボタンを自動的に調整し、使用中の最適なボタン操作を維持します。これは、PCB 基板とオーバーレイの製造工程のばらつきに適應しており、LCD インバーター、AC ライン、スイッチング電源などのノイズ源を自動的に調整してノイズを取り除きます。

SmartSense 自動チューニングは「[自動チューニング](#)」で詳しく説明されています。

3. PSoC 3 および PSoC 5LP における CapSense



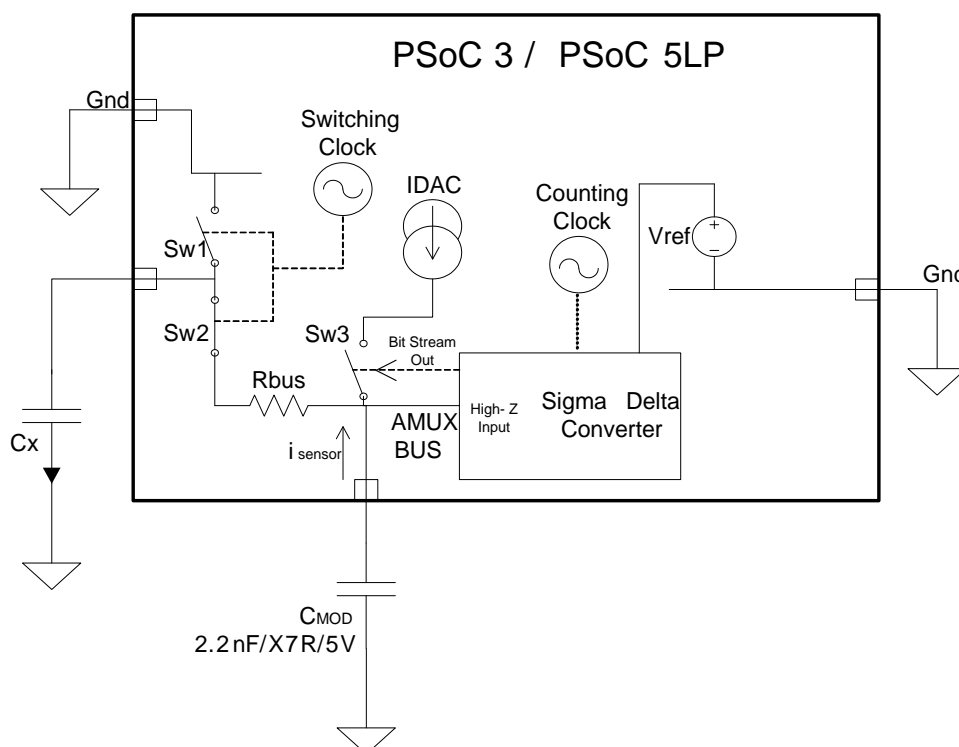
PSoC 3 および PSoC 5LP はプログラム可能なデバイスであり、豊富なアナログおよびデジタル リソースが用意されています。これらのデバイスは、多数のユニークな機能を持った、CSD 方式を用いた CapSense を実装します。これらのデバイスは柔軟性があり、リソースが豊富なため、CapSense やその他多くのシステム機能を実行できます。

3.1 CSD の実装

CSD 方式の詳細な説明は「[静電容量の変換](#)」を参照してください。図 3-1 は、CSD 方式をどのように PSoC 3 および PSoC 5LP デバイスに実装するかを示します。これらのデバイスは最大 4 つの DAC リソースがあります。CapSense はこれらのいずれかを使用して電流ソース (IDAC) を実装します。スイッチは非重複クロック (スイッチング クロック) によって制御されます。アナログバス (AMUXBUS) は C_{MOD} 、 C_X 、IDAC およびシグマ デルタ コンバータの入力を一緒に接続します。シグマ デルタ コンバータは、IDAC を制御して C_{MOD} 電圧が基準電圧 (V_{ref}) に近似するようにします。シグマ デルタ コンバータはカウンタにビット ストリームを出力します。カウンタは raw カウントを出力します。CPU は、raw カウントを処理してセンサーの状態を判断します。

図 3-1 は、CSD の IDAC ソースの実装方式を示します。その他の方式は「[電流ソース方式](#)」を参照してください。

図 3-1. CSD の実装 (IDAC ソース方式)



3.2 CapSense 独自の特長

3.2.12 チャネル デザイン

PSoC 3 および PSoC 5LP デバイスは、2 個のセンサーを同時にスキャンすることが可能です。これにより、スキャン時間は半減されます。スキャン時間を短縮することにより、ボタン オン/オフ検出の応答時間が増え、平均消費電力を削減できます。

2 チャネルを使用する場合、1 チャネルのリソースが 2 倍必要となるので、20 個以上のセンサーを持っているデザインのものに推奨されています。「[チャネル数](#)」では、2 チャネル デザインを選択する方法が説明されています。[表 3-1](#) は、1 チャネル デザインと 2 チャネル デザインに必要なリソースを比較します。

表 3-1. 1 チャネル デザインと 2 チャネル デザインのリソース比較

リソース タイプ	1 チャネル	2 チャネル
アナログ リソース ¹	AMUXBUS 1 または 2 本 コンパレータ 1 個 DAC 1 個	AMUXBUS 2 本 コンパレータ 2 個 DAC 2 個
デジタル リソース ²	データバス 4 本 マクロセル 19 個	データバス 6 本 マクロセル 31 個
外部コンポーネント ³	C _{MOD} コンデンサ 1 個 ブリード抵抗 1 セット	C _{MOD} コンデンサ 2 個 ブリード抵抗 2 セット

3.2.2 耐水性デザイン

一部の CapSense 静電容量タッチ センシング システムは、水が存在する場では信頼性の高い動作を必要とします。家電製品や車載アプリケーション、産業用アプリケーションは、水や氷、湿度の変化を伴う環境で動作する必要があるシステムの例です。そのようなアプリケーションには、シールド電極およびガード センサーで堅牢なタッチ センシングを提供することが可能です。

水滴や水分の耐性が必要な用途の場合は、シールド電極を使用する必要があります。[図 3-2](#) のように、シールド電極はセンサーの周りを囲んでいます。シールド電極はデバイスの I/O ピンに接続され、センサー ピンと同じスイッチング信号で駆動されます。同じ信号でシールド電極とセンサー両方を駆動すると、それらの間の静電容量が無効になります。つまり、センサーとシールドを部分的に覆っている水膜や水滴が効果的に除去されます。シールド電極はセンサーの C_p も軽減します。必要な I/O ピン数は、基板サイズとシールド部分によって異なります。シールド電極の占める面積が大きい場合、複数の I/O ピンで駆動する必要があります。

接触面の水流に対する耐性を必要とする場合、シールド電極と共にガード センサーを使用してください。ガード センサーはタッチ センシング領域の全体を被う必要があります。[図 3-2](#) に示すように、ガード センサーは、通常タッチ センシング領域の境界を覆います。ガード センサーは、その他のセンサーと同じ方法で CapSense デバイスによってスキャンされます。ガード センサーに水が存在すると、それがアクティブになり、誤った指タッチを検出すること防ぐために他のセンサーのスキャンを無効にします。

「[シールド電極およびガード センサー](#)」では、シールド電極とガード センサーを有効にする方法を説明します。

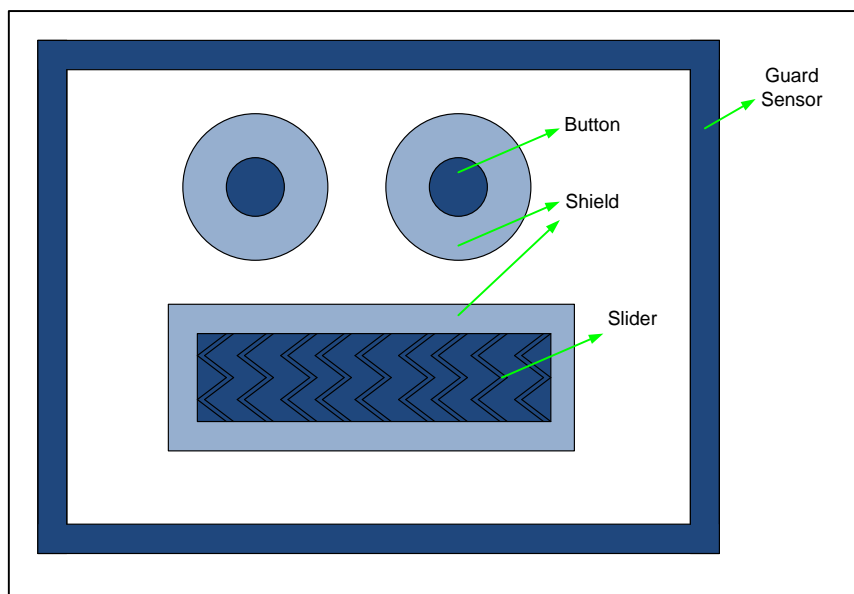
¹ 左 AMUXBUS と右 AMUXBUS があります。1 チャネル デザインは 1 本だけの AMUXBUS が必要で、すべてのセンサーがチップの片側に配置

されています。2 チャネル デザインはセンサーをチップの両側に配置し、左 AMUXBUS と右 AMUXBUS を短絡させます。IDAC ソース方式または IDAC シンク方式が選ばれた場合、少なくとも 1 個の DAC が必要です。「[電流ソース方式](#)」を参照してください。

² データバスとマクロセルは、ユニバーサル デジタル ブロック (UDB) のコンポーネントのブロックです。PSoC 3 および PSoC 5LP デバイスには、最大 24 の UDB ブロックがあります。

³ 外部抵抗方式が選択された場合、ブリード抵抗が必要になります。「[電流ソース方式](#)」を参照してください。

図 3-2. シールド電極およびガード センサー



3.2.2.1 SIO ピン

PSoC 3 および PSoC 5LP デバイスには、プログラマブルな「論理 HIGH」レベルという独自の機能を備えた特別の入力／出力 (SIO) ピンがあります。つまり、SIO の論理 HIGH レベルは V_{DD} に固定されず、ユーザー定義の電圧にプログラムすることができます。

この機能は、SIO ピンがシールド電極として使用される場合に重要です。シールド電極は、その信号が CapSense センサーの信号と一致する場合に最高の結果を生み出します。SIO ピンを使用することにより、CapSense センサーに一致する論理 HIGH レベルを選択することができます。

3.2.3 電流ソース方式

PSoC 3 および PSoC 5LP デバイスは 3 つの CSD 実装方法の中で 1 つを選択できます。デザインにおいて最高の性能を生み出す方式を判定する際、ソース要件、ノイズ感受性、シールド電極の要件などいくつかの要素があります。これらのすべての方式では、Vref の初期設定値はバンドギャップ電圧 1.024V に相当します。

表 3-2 は 3 つの方式の比較を示します。PSoC 3 および PSoC 5LP デバイスには、型番に応じて最大 4 個の DAC があります。設計上の他の機能はすべての DAC を使用する場合、CapSense 用に外部抵抗方式を使用することが可能です。「電流ソース」節は電流ソース方式の選択方法について説明します。

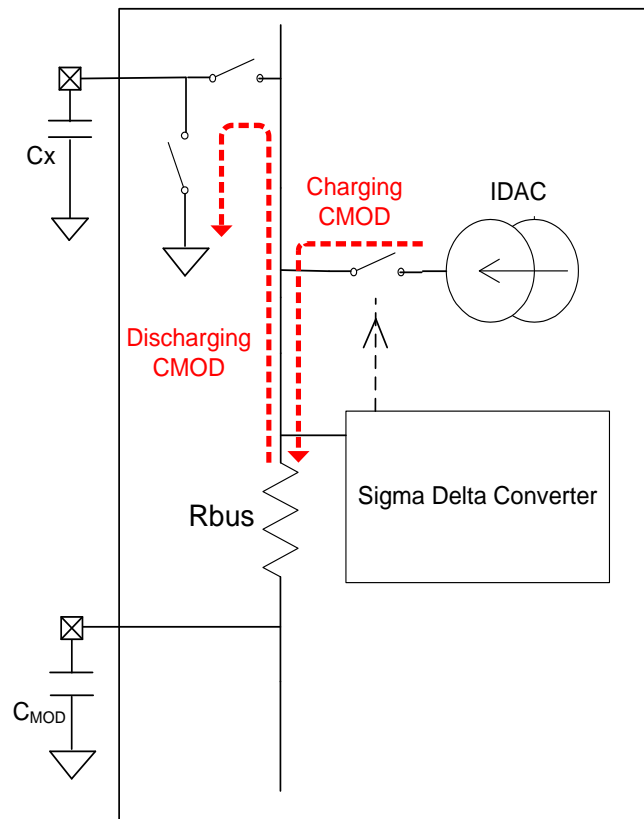
3.2.3.1 IDAC ソース方式

このコンフィギュレーションでは、CapSense センサーは C_{MOD} と GND 間を連続的に切り替え、C_{MOD} を充電します。IDAC はソースモードに設定され、C_{MOD} 電圧が Vref 以下になるとオンになります。図 3-3 は、IDAC ソース方式のブロック図を示します。

この方式は、センサーの電圧振幅が小さい (Vref と GND の間) ため、指伝導ノイズの影響を受けやすいものです。VDAC で Vref を増加させると、ノイズ耐性が高くなりますが、追加の DAC リソースが必要となります。

IDAC ソース方式を使用した場合、SIO ピンをシールド電極に使用します。これにより、シールド電極を Vref と GND 間で切り替えることができ、センサーと同じ信号を取得することを保証します。

図 3-3. IDAC ソース方式

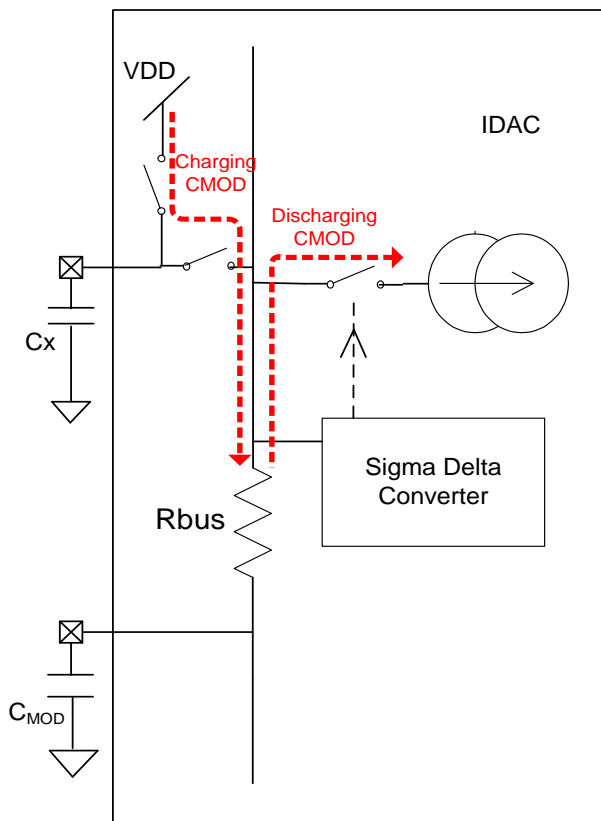


3.2.3.2 IDAC シンク方式

このコンフィギュレーションでは、CapSense センサーは V_{DD} と C_{MOD} 間を連続的に切り替え、 C_{MOD} を充電します。IDAC はシンクモードに設定され、 C_{MOD} 電圧が V_{ref} を超えるとオンになります。図 3-4 は、IDAC シンク方式のブロック図を示します。

この方式は、センサーが V_{DD} と V_{ref} 間を切り替えるため、電源ノイズの影響を受けやすいものです。

図 3-4. IDAC シンク方式



3.2.3.3 外部抵抗方式

このコンフィギュレーションは、IDAC の代わりに外部抵抗（ブリード抵抗）を用いて C_{MOD} を放電することを除き、IDAC シンク方式に似ています。CapSense センサーは、 V_{DD} と C_{MOD} 間を連続的に切り替え、 C_{MOD} を充電します。 C_{MOD} 電圧が V_{ref} を超えると、ブリード抵抗はグラウンドに接続されます。図 3-5 は、外部抵抗方式のブロック図を示します。

この方式は、センサーが V_{DD} と V_{ref} 間を切り替えるため、電源ノイズの影響を受けやすいものです。

図 3-5. 外部抵抗方式

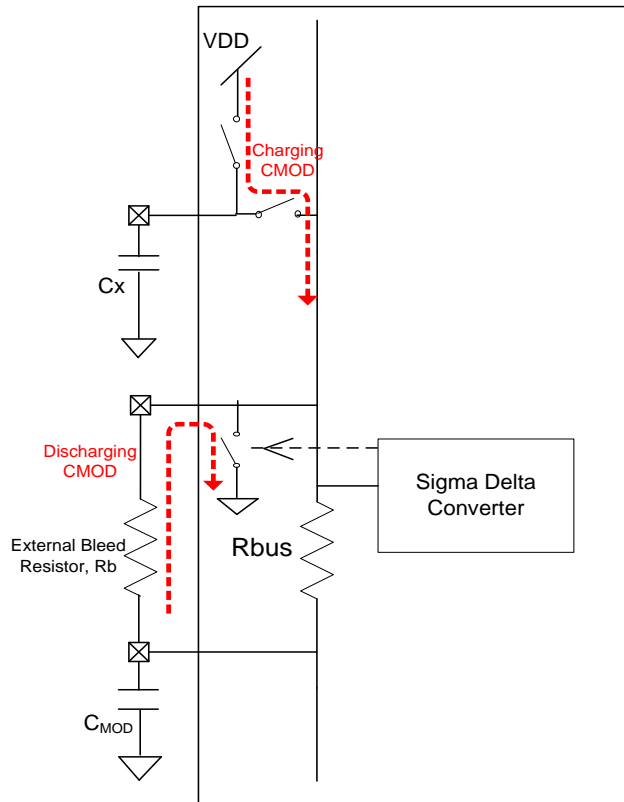


表 3-2. 電流ソース方式の比較

	IDAC ソース	IDAC シンク	外部抵抗
DAC リソース	必要	必要	不要
外部ブリード抵抗	不要	不要	必要
電源ノイズ	影響されない	影響される	影響される
指伝導ノイズ	影響される VDAC で V_{ref} を増加させることが必要	影響されにくい	影響されにくい
シールド電極	SIO ピン	任意の GPIO ピン	任意の GPIO ピン

3.2.4 チューニング

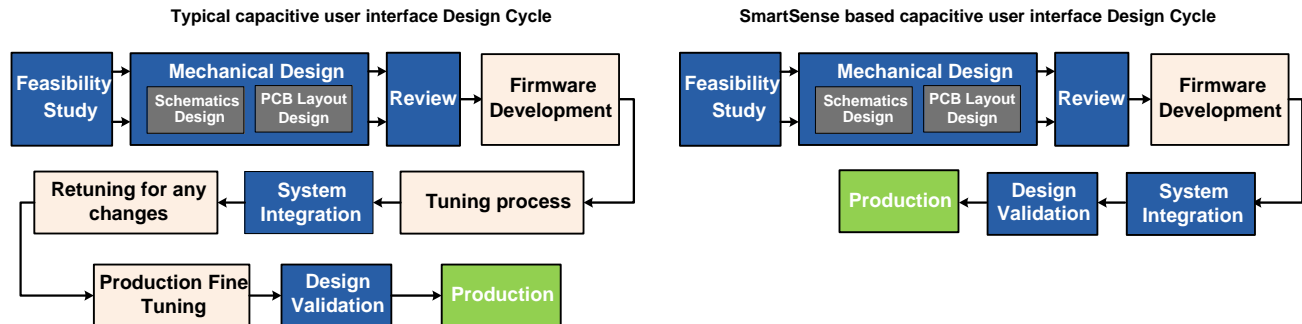
SmartSense 自動チューニングは、電源投入時に各 CapSense ボタンを自動的に調整し、使用中の最適なボタン性能を維持する先進技術です。設計サイクル時間を削減し、プロセスのばらつきに適応します。パラメーターをさらに制御することが必要なとき、または C_P が高い場合は、CapSense パラメーターを手動で調整することもできます。「[チューニング方法](#)」節は、チューニング方法の選択について説明します。「[CapSense 性能のチューニング](#)」節は、自動チューニングおよび手動チューニングのプロセスを説明します。

3.2.4.1 自動チューニング機能

設計サイクルタイムの短縮

図 3-6 は、SmartSense 自動チューニングがどのように設計サイクル タイムを著しく短縮するかを説明します。

図 3-6. 設計サイクルの比較



次の例は、SmartSense 自動チューニングがどのように設計サイクル タイムを短縮させ、プラットフォームの設計を可能にするかを説明します。図 3-7 は 21 インチのラップトップ モデルのマルチメディア キーを示し、図 3-8 は 15 インチ モデルのマルチメディア キーを示します。キーは、同じ機能とサイズを持っている CapSense ボタンです。しかし、モデル間でデザインを直接移植することはできません。理由は、21 インチ モデルは、ボタン間の間隔がより大きく、ボタンと CapSense コントローラー間の配線がより長いからです。デザインを再調整する必要があります。SmartSense 自動チューニングは、開発者が同じデザインを別のモデルへ移植する際にデザイン サイクル時間を大きく削減することを可能にします。

図 3-7. 21 インチ モデル用のラップトップ マルチメディア キー



図 3-8. 15 インチ モデル用のラップトップ マルチメディア キー (同一の機能およびボタン サイズ)



プロセスのばらつきに対する耐性

C_P は、プリント基板のレイアウトおよび配線の長さ、基板製造プロセスのばらつき、または複数購買サプライ チェーン内のベンダー間のプリント基板のばらつきによって異なる可能性があります。ボタンの感度は C_P に依存します。 C_P 値が高くなると感度が低下し、指を触れる信号の振幅が狭くなります。 C_P の変化により、ボタンが高感度になりすぎたり、感度が十分でなくなったり、または動作しなくなってしまうことがあります。そのとき、ユーザーはシステムを再調整し、場合によってはユーザーインターフェースのサブシステムを再検証する必要があります。SmartSense 自動チューニングはこのような問題を解決します。

使いやすさ

SmartSense 自動チューニングでは、すべてのパラメーターに関する深い知識を持っていなくても CapSense アプリケーションを迅速で容易に設計できます。

3.2.4.2 手動チューニング機能

詳細制御

手動チューニングは、各 CapSense パラメーターを選択することができます。これは、ノイズが高いシステムなどの特別な動作条件のシステムに重要です。ノイズの多いシステムで自動チューニングの使用は、誤ったボタン タッチの原因になる可能性があります。手動で指閾値を増加させることで、システムのノイズ耐性を向上させます。

高寄生容量

自動チューニングは、5~45pF 範囲の C_P 値に対応するように設計されています。 C_P が長い配線や大きなボタン サイズのために 45pF を超えた場合、手動チューニングを使用してください。

不良検出アルゴリズム

自動チューニングでは、起動および実行中でパラメーターを変更することが可能です。これにより、不良検出アルゴリズムを書くのは難しくなることがあります。

不良検出アルゴリズムを考えてみましょう。ベースライン カウントが工場出荷時にフラッシュに格納された値の範囲内であるかどうかを確認します。格納された値が 3000 カウント、スキャン分解能が 12 ビットであると仮定します。ベースライン カウントは C_P の平均カウントであり、スキャン分解能は静電容量の計測の分解能です (これら用語の詳細な説明は「[raw カウント](#)」および「[スキャン分解能](#)」を参照してください)。自動チューニングを使用した場合、物理的条件または環境条件が変化するためスキャン分解能パラメーターが 13 ビットに設定されることがあります。スキャン分解能が 13 ビットに設定されると、ベースラインは 6000 カウントになり、システムは不調になります。

RAM とフラッシュの使用の削減

自動チューニング アルゴリズムにはより大量のフラッシュと RAM が必要です。[表 3-3](#) は、4 個のセンサーを使用する CapSense デザインに必要なメモリを示します。

表 3-3. 手動チューニングと自動チューニングの必要なメモリの比較

チューニング方法	PSoC 3		PSoC 5LP	
	RAM (バイト)	フラッシュ (バイト)	RAM (バイト)	フラッシュ (バイト)
手動チューニング	207	6345	384	5104
自動チューニング	292	8282	488	6152

3.2.5 ノンブロッキング アーキテクチャ

CapSense スキャンは 3 つの段階があります。

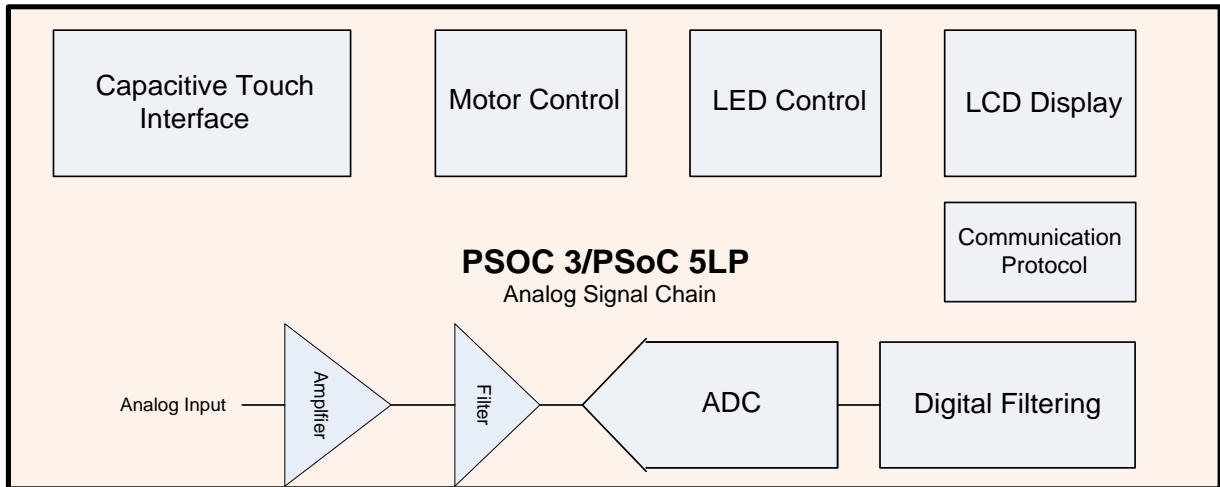
1. **前スキャン:** ハードウェアの準備
2. **ハードウェア スキャン:** 実際のハードウェア スキャン
3. **後スキャン:** 結果の処理と保存

前スキャンと後スキャンのみは、CPU がコードを実行する必要があります。コード アーキテクチャは、ハードウェア スキャンが完了するまで待ちません。代わりに CPU はコードの次のラインを実行し、ハードウェアはスキャンが完了し、割り込みを生成すると、ISR で後スキャン コードを実行します。このノンブロッキング アーキテクチャは、CPU が複数のアプリケーションを処理するデザインに役立ちます。

3.3 CapSense PLUS

CapSense PLUS は、CapSenseに加えて機能を実行する PSoC デバイスを表します。PSoC 3 および PSoC 5LP デバイスはこのカテゴリに入っています。図 3-9 のように、これらのデバイスが提供している様々な機能により、多数のシステム機能を単一のチップに組み込むことができます。これにより、基板のサイズ、BOM コストおよび消費電力を削減できます。

図 3-9. システム内の CapSense PLUS の機能



詳細については以下のリンクを閲覧ください。

- [アナログ機能](#)
- [通信モデム](#)
- [LCD 駆動](#)
- [低消費電力](#)
- [USB 接続](#)

PSoC 3 および PSoC 5LP デバイスを用いて成功したデザインについては、以下のリンクを閲覧ください。

- [血糖値計](#)
- [血圧計](#)
- [排卵日検査器](#)
- [薬物注入ポンプ](#)
- [iPod、iPhone、iPad アクセサリ](#)
- [LED プロジェクター](#)
- [磁気カードリーダー](#)
- [パルス式酸素濃度計](#)
- [3次元 \(3D\) のアクティブ シャッター メガネ](#)

4. CapSense 設計ツール



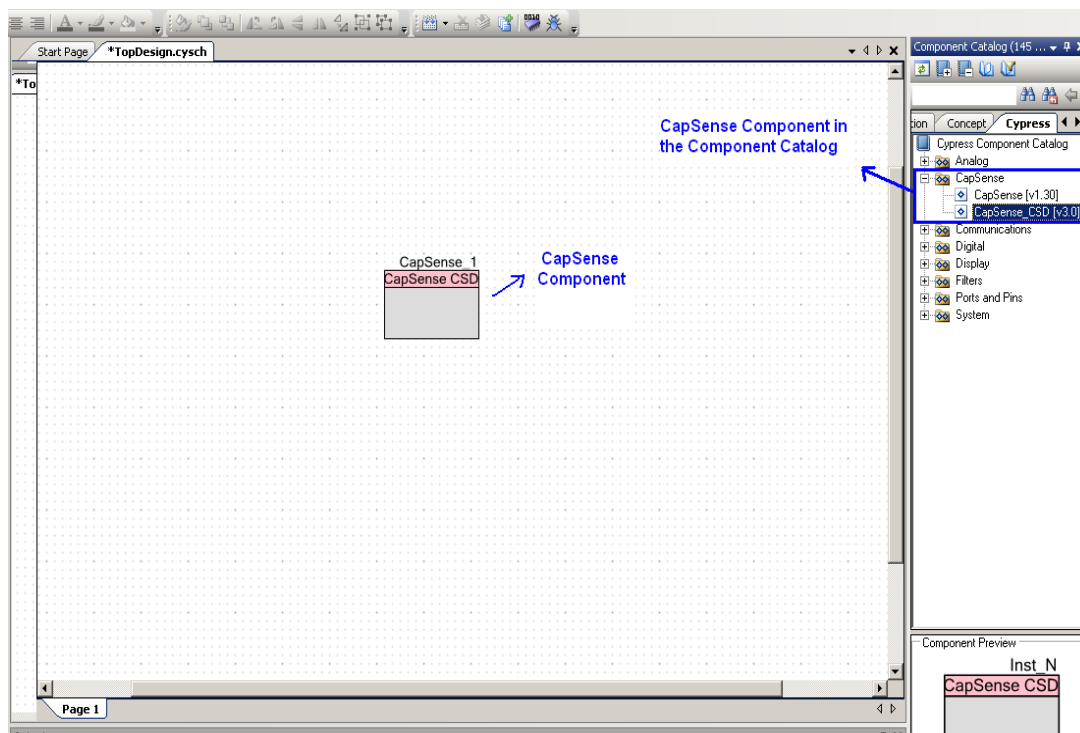
サイプレスは、CapSense 静電容量タッチ センシング アプリケーションの開発用に幅広いハードウェアおよびソフトウェア ツールを提供しています。注文情報は「[リソース](#)」を参照してください。

4.1 PSoC Creator

サイプレスの **PSoC Creator** は統合開発環境です。PSoC Creator は、ハードウェア コンフィギュレーションとソフトウェア開発を単一の統合ツールに組み合わせています。アプリケーションは、コンポーネント ライブラリを使用するドラッグ アンド ドロップ方式の設計環境で開発されます。

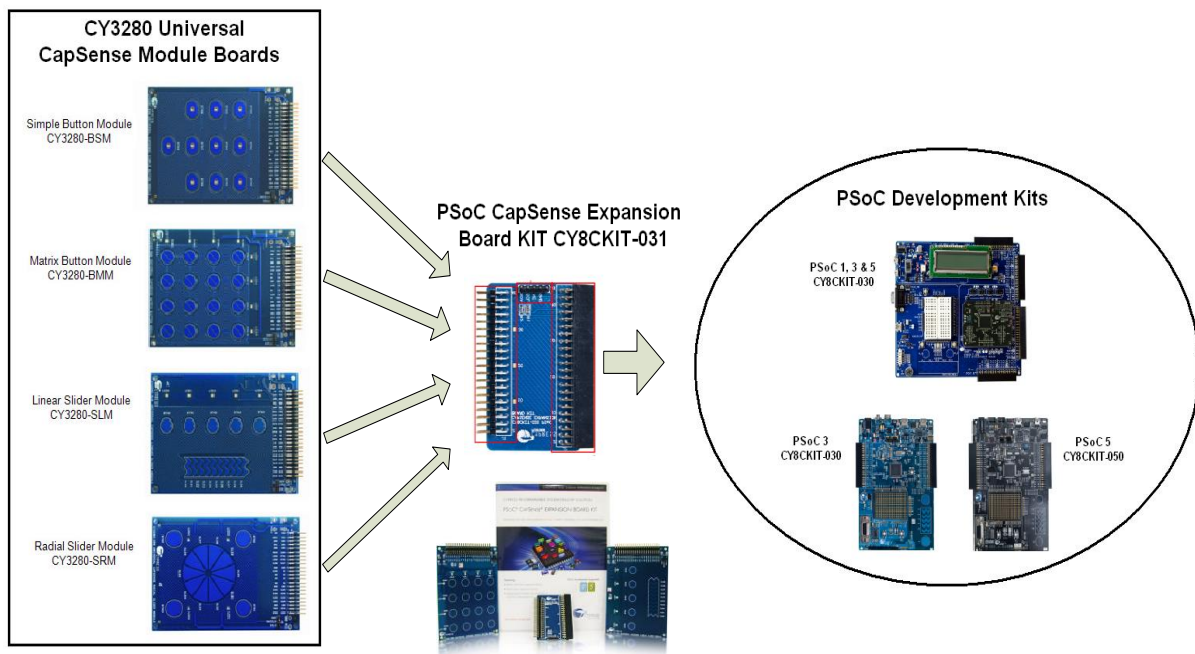
PSoC Creator には CapSense_CSD コンポーネントがあります。このコンポーネントは、スイッチト コンデンサ回路、アナログ バス、コンパレータ、デジタル カウント機能、高レベル ソフトウェア ルーチン (API) を使用して静電容量タッチ センサーを実装します。I²C、SPI、UART、タイマー、PWM、アンプ、ADC、LCD など他の機能を実装するために使用できるその他デジタルおよびアナログコンポーネントがあります。図 4-1 は、コンポーネント カタログからドラッグし、TopDesign に配置した CapSense_CSD コンポーネントです。

図 4-1. PSoC Creator の TopDesign



4.2 ハードウェア キット

図 4-2. CapSense ハードウェア



4.2.1 PSoC 3 および PSoC 5LP 開発キット

PSoC 3 および PSoC 5LP 開発キットは CapSense 機能に対応しています。

- [CY8CKIT-001 PSoC®開発キット](#)
- [CY8CKIT-030 PSoC® 3 開発キット](#)
- [CY8CKIT-050B PSoC® 5LP 開発キット](#)

4.2.2 ユニバーサル CapSense モジュール基板

サイプレスは、ニーズを満たす各種の CapSense センサー、LED およびインターフェースを備えたユニバーサル CapSense モジュール基板を提供しています。

- [CY3280-BSM](#) シンプル ボタン モジュール
- [CY3280-BMM](#) マトリクス ボタン モジュール
- [CY3280-SLM](#) リニア スライダー モジュール
- [CY3280-SRM](#) ラジアル スライダー モジュール
- [CY3280-BBM](#) ユニバーサル CapSense プロトタイプ モジュール

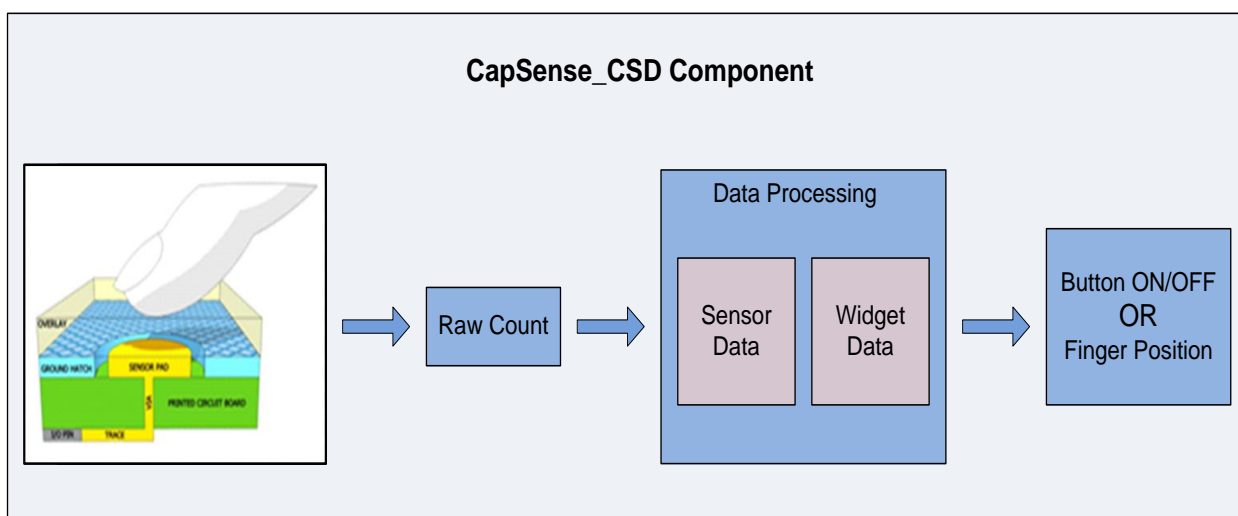
4.2.3 PSoC CapSense 拡張基板キット

[CY8CKIT-031](#) PSoC CapSense の拡張基板キットは、任意の PSoC 3 と PSoC 5LP 開発キットを任意のユニバーサル CapSense モジュール基板に接続します。CY8CKIT-031 は、インターフェース基板および 2 個のモジュール基板 (CY3280-SLM と CY3280-BMM) を提供します。

5. CapSense_CSD コンポーネント



図 5-1. CapSense_CSD コンポーネント ブロック図



CapSense_CSD コンポーネントは、完全な CapSense システムを提供します。コンポーネントには、高レベルと低レベルに分類されたいくつかのパラメーターがあります。パラメーターは、グローバル アレイを使用してファームウェア内で相互に通信します。

高レベルのパラメーターは、CapSense システムが raw カウントを、センサーのオン/オフ状態などのような、便利な情報に変換する処理方法を制御します。高レベル パラメーターの例は、指閾値、ノイズ閾値、デバウンス カウンターなどです。「[高レベル パラメーター](#)」を参照してください。

低レベルのパラメーターは、物理層での CapSense システム動作を制御します。物理層で静電容量は raw カウントに変換されます。低レベルのパラメーターの例は、IDAC 範囲、IDAC 値、スキャン クロックなどです。「[低レベル パラメーター](#)」を参照してください。

CapSense_CSD コンポーネントは、「ウィジェット」と呼ばれているタッチ インターフェースの各種を提供します。ウィジェットは、1 個以上のセンサーから構成されています。それらはスライダーやタッチ パッドなどのインターフェース オブジェクトを表します。ウィジェットのタイプは、ボタン、スライダー、ラジアル スライダー、マトリクス ボタン、タッチ パッド、近接センサーを含んでいます。[図 5-2](#) はスライダーの例を示します。スライダー ウィジェットを形成するには、7 個のセンサーを狭い間隔で直線に配置します。

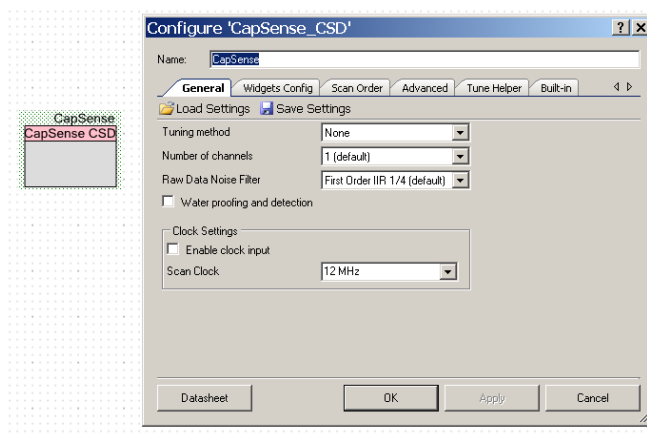
図 5-2. 7 セグメント スライダー



5.1 パラメーターのまとめ

図 5-3 は、PSoC Creator の CapSense_CSD コンポーネントのスクリーンショットおよびコンフィギュレーション ウィンドウを示します。また、コンポーネントをダブル クリックするか、右クリックして「Configure」を選択することでコンフィギュレーション ウィンドウを開きます。

図 5-3. PSoC Creator の CapSense コンポーネント



コンフィギュレーション ウィンドウには、いくつかのパラメーターが異なるタブに配置されています。表 5-1 は、このウィンドウにあるパラメーターをまとめ、関連するリンクを提供します。

表 5-1. CapSense_CSD コンポーネントのコンフィギュレーション ウィンドウのパラメーター

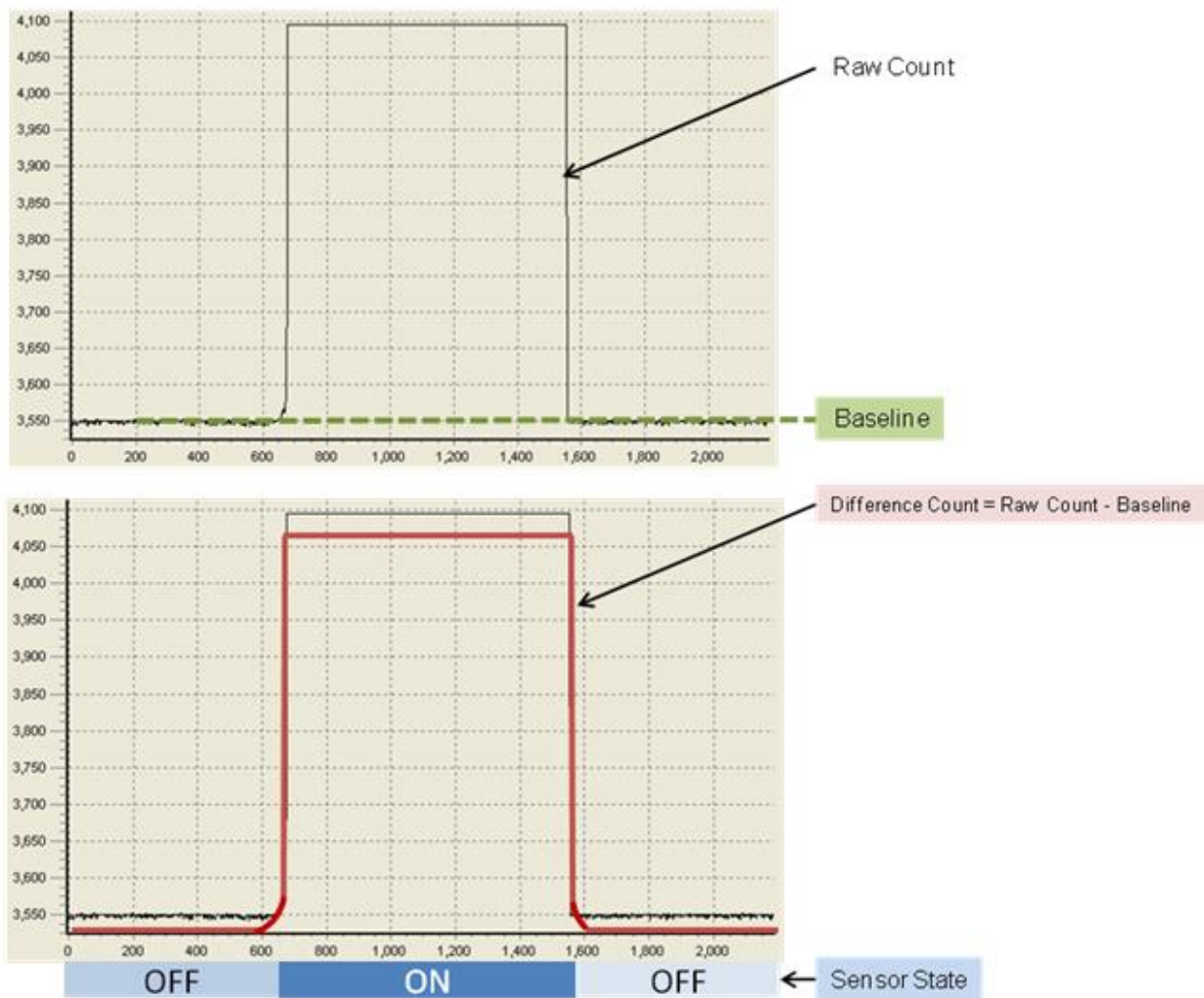
汎用	ウィジェット設定	スキャン順序	詳細設定		チューナー ヘルパー
Tuning Method (チューニング方法)	Add Widget (ウィジェット追加)	Analog Switch Divider (アナログ スイッチ分周器)	Analog Switch Drive Source (ア ナログ スイッチド ライブ ソース)	Shield (シールド)	Tune Helper (チューナー ヘルパー) の イネーブル
Number of channels (チャンネル数)	Finger Threshold (指閾値)	IDAC Value (IDAC 値)	Multiple Analog Switch Divider (複数のアナログ スイッチ分周器)	Inactive Sensor Connection (非アクティブな センサー接続)	EZ12C コンポーネントの インスタンス名
Raw Data Noise Filter (raw データノ イズ フィルター)	Noise Threshold (ノイズ閾値)	Move to Channel 1/ Move to Channel 0 (チャンネル 1/ チャンネル 0 に変更)	Analog Switch Divider (アナログ スイッチ分周器)	Guard Sensor (ガード センサー)	
Water Proofing and Detection (耐水性および検知)	Hysteresis (ヒステリシス)	Sensitivity (感度)	Scan Speed (スキャン速度)	Current Source (電流ソース)	
Enable Clock Input (クロック入力の イネーブル)	Debounce (デバウンス)		PRS EMI Reduction (PRS EMI 減少)	IDAC Range (IDAC 範囲)	
Scan Clock (スキャン クロック)	Scan Resolution (スキャン分解能)		Sensor Auto Reset (センサー 自動リセット)	Number of Bleed Resistors, channel 0/ channel 1 (ブリード抵抗、 チャンネル 0/ チャンネル 1 の数)	
	Number of Sensor Elements (センサー エレメント数)		Widget Resolution (ウィジェット 分解能)	Digital Resource Implementation, channel 0/ channel 1 (デジタル リソース 実行、チャンネル 0/ チャンネル 1)	
	API Resolution (API 分解能)		Negative Noise Threshold (負のノイズ閾値)	Voltage Reference Source (電圧基準ソース)	
	Position Noise Filter (位置ノイズ フィルタ)		Low Baseline Reset (低ベース ライン リセット)		
	No of Dedicated Sensor Elements (専用センサー エレメント数)				

低レベル パラメーター
高レベル パラメーター
ウィジェット パラメーター
その他のシステム パラメーター

5.2 グローバル アレイ

CapSense システムは、環境の変化がある場合の適切な検出と動作を確保するために、いくつかのグローバル アレイを使用します。これらのアレイは手動で変更できませんが、デバッグ用に検査できます。

図 5-4. グローバル パラメーター



5.2.1 raw カウント

CapSense コントローラー ハードウェアは静電容量を計測し、raw カウントと呼ばれるデジタル形式で結果を提供します。raw カウントの値は、センサーの静電容量が増加するとともに増加します。

5.2.2 ベースライン

センサーの raw カウント値は、温度や湿度などの環境変化により、徐々に変わります。このような徐々に現われる変動はベースライン値によって補正されます。ベースラインは、ソフトウェア アルゴリズムを使用して、raw カウントの徐々に起こる変化を追跡します。これは raw カウントの突然の変化に影響されにくいローパス フィルタです。ベースライン値は、差分カウントを計算するための基準レベルを提供します。

5.2.3 差分カウント

差分カウントとは、センサーの raw カウントとベースラインの差です。通常は、センサーが非アクティブの場合、差分カウントはゼロです。センサーにタッチすると、raw カウントが増加するため、正の差分カウント値が発生します。

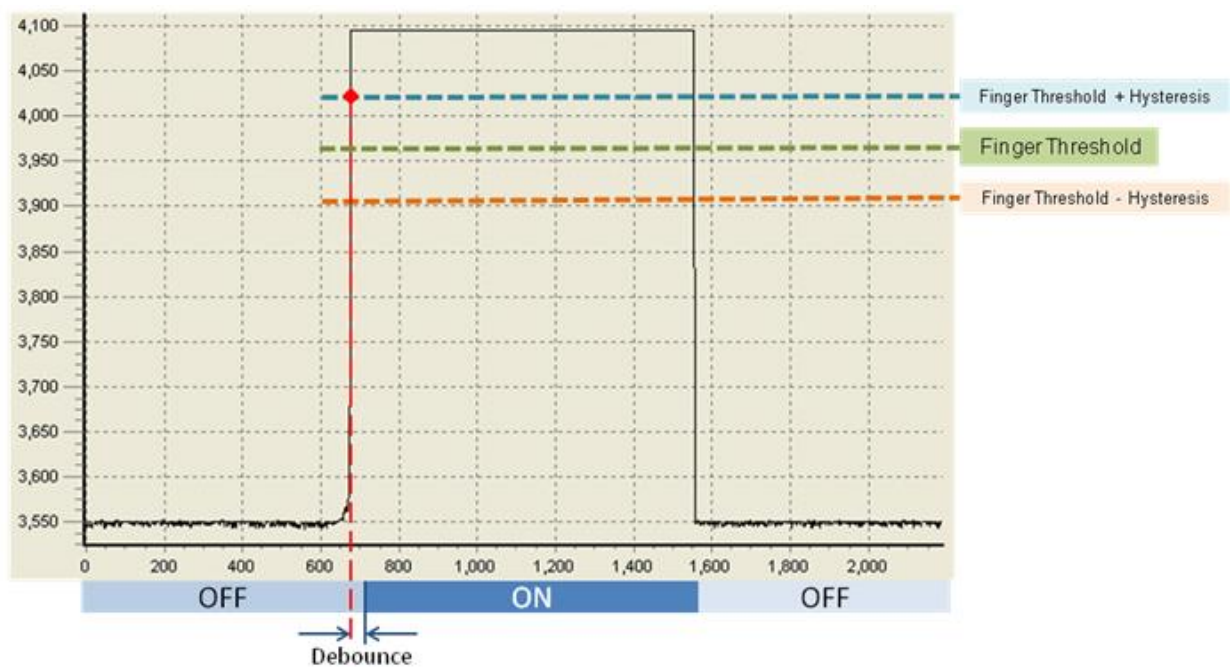
5.2.4 センサー状態

各センサーの状態は、ボタンがオンの場合に 1、ボタンがオフの場合に 0 として表します。すべてのセンサーのオン/オフ状態はバイトアレイに格納されます。各アレイ エLEMENTは、8 個のセンサーのオン/オフ状態を保持できます。

5.3 高レベル パラメーター

高レベル パラメーターでは、センサーのオン/オフ状態やスライダー上の指の評価位置などの情報を生成するために raw カウントを処理する方法を定義します。図 5-5 および式 4 は、高レベル パラメーターの概要を提供します。

図 5-5.高レベル パラメーター



式 4

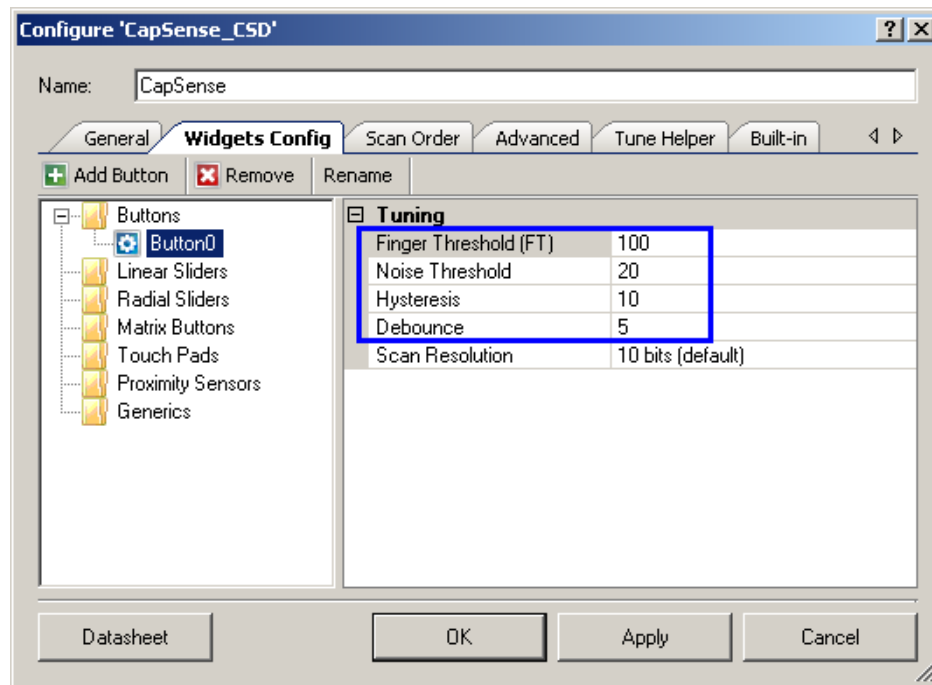
(差分カウント \geq 指閾値 + ヒステリシス) & (サンプルカウント \geq デバウンス)の場合、センサー状態 = ON

if (差分カウント \leq 指閾値 - ヒステリシス)の場合、センサー状態 = OFF

ここで、

サンプル カウント=指閾値+ヒステリシスより大きい測定値のサンプル数

図 5-6. 高レベル パラメーター



5.3.1 指閾値

指閾値パラメーターは、指のタッチに対するセンサーの感度を定義します。これは式 4 で定義されたように、ヒステリシス パラメーターと併用してセンサーの状態を判断します。値の範囲は 0～255 です。

5.3.2 ヒステリシス

ヒステリシス パラメーターは式 4 で定義されたように、指閾値と併用してセンサーの状態を判断します。ヒステリシスはノイズの多い遷移の耐性を向上させます。これはボタンのデバウンス機能です。タッチ状態は、差分カウントが指閾値より少し大きくなるまで、オフのままになります。タッチ状態は、差分カウントがノイズ閾値より少し小さくなるまで、オンのままになります。これにより、差分カウントがノイズを含んでおり、ノイズと指閾値の間にある場合、タッチ／非タッチのステート マシンがトリガすることを防ぎます。

値の範囲は 0～255 です。ヒステリシス パラメーターは、指閾値パラメーターより低く設定する必要があります。

5.3.3 デバウンス

デバウンス パラメーターは、センサーのオフからオンへの遷移にカウンターを加えます。センサーがオフからオンになるには、特定のサンプル数の間、差分カウントは (指閾値+ヒステリシス レベル) より大きい値を維持する必要があります。

値の範囲は 1～255 です。1 をセットするとデバウンスはありません。

5.3.4 ノイズ閾値

個々のセンサーに対して、ノイズ閾値パラメーターは raw カウントの上限を設定してベースライン値を更新します。スライダ センサーに対しては、raw カウントの下限を設定して重心計算で結果をカウントします。

値の範囲は 3～255 です。ユーザー モジュールが正常に動作するためには、ノイズ閾値を (指閾値-ヒステリシス値) より高く設定しないでください。

5.3.5 負のノイズ閾値および低ベースライン リセット

図 5-7. ノイズ閾値とベースライン更新パラメーター

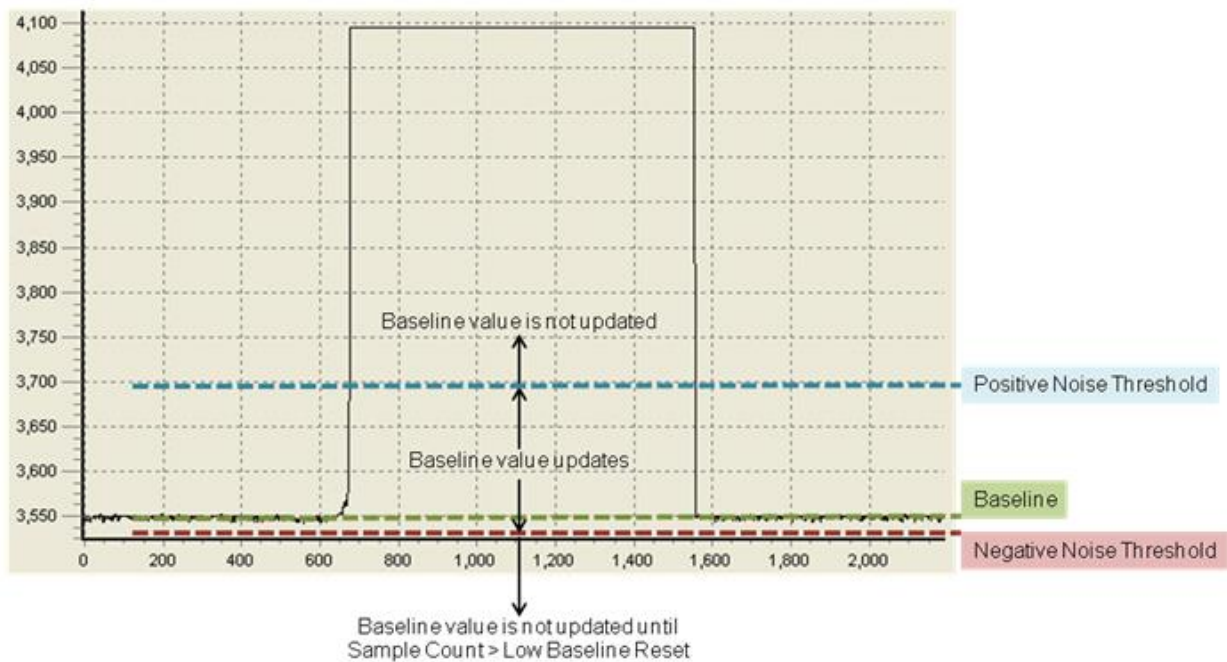
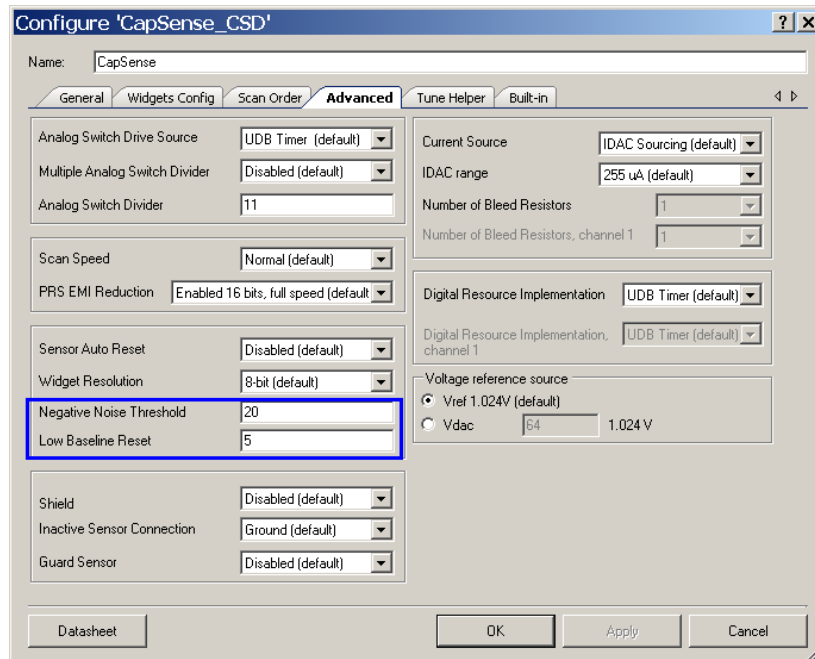


図 5-8. 負のノイズ閾値と低ベースライン リセット パラメーター

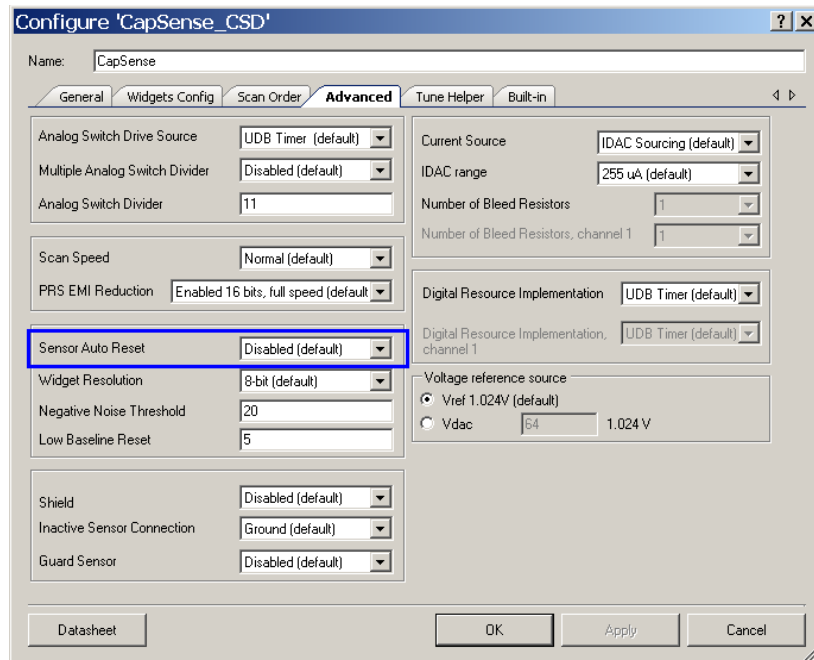


負のノイズ閾値パラメーターは、負の差分カウント閾値として使用されます。raw カウントが、低ベースライン リセット パラメーターによって指定されたサンプル数の間、(ベースライン-負のノイズ閾値) より小さい場合、ベースラインは新しい raw カウント値に設定されます。値の範囲は 0~255 です。

低ベースライン リセット パラメーターは、負のノイズ閾値パラメーターと併用されます。これは、ベースライン値のリセットに必要な異常に低いサンプルの数をカウントするものです。起動時に指が置かれている状態の補正するために使用されます。値の範囲は 0~255 です。

5.3.6 センサー自動リセット

図 5-9. センサー自動リセット パラメーター



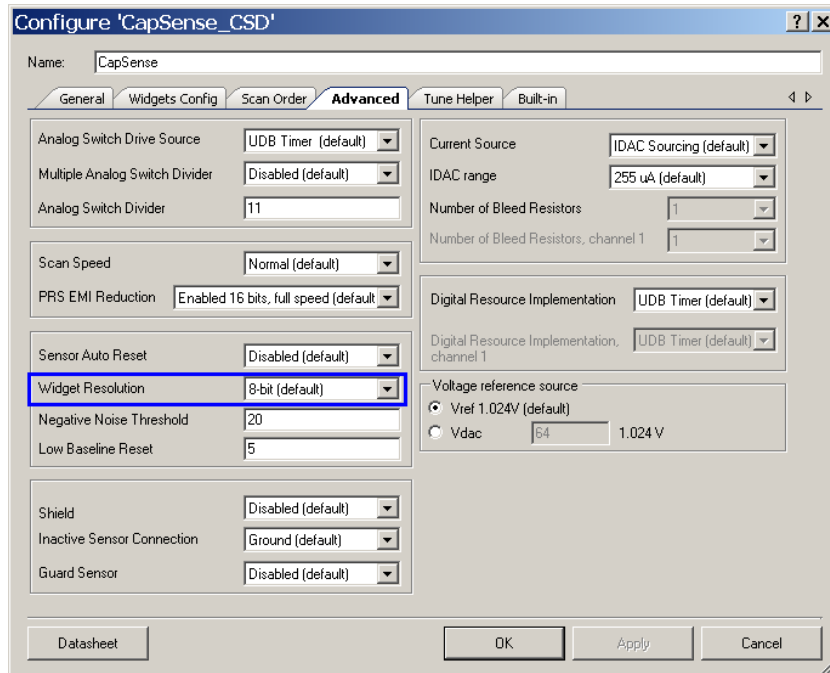
このパラメーターは、ベースライン値が常に更新されるか、差分カウントがノイズ閾値より低い場合にのみ更新されるかを指定します。

Enabled (有効): ベースラインは常に更新されます。これは、センサーの最大時間（標準値は 5~10 秒）を制限しますが、センサーに触れずに raw カウントが誤って上がった場合に、センサーが永久的にオンになることを防ぎます。この突然の上昇の原因には、大幅な電源電圧の変化や高エネルギーRF ノイズ ソース、急速な温度変化などがあります。

Disabled (無効): ベースライン値は差分カウントがノイズ閾値パラメーターを下回った場合にのみ更新されます。

5.3.7 ウィジェット分解能

図 5-10. ウィジェット分解能パラメーター



The screenshot shows the 'Configure CapSense_CSD' dialog box with the 'Advanced' tab selected. The 'Name' field is set to 'CapSense'. The 'Widget Resolution' dropdown menu is highlighted with a blue box, showing '8-bit (default)'. Other parameters include 'Analog Switch Drive Source' (UDB Timer), 'Current Source' (IDAC Sourcing), 'IDAC range' (255 uA), 'Number of Bleed Resistors' (1), 'Scan Speed' (Normal), 'PRS EMI Reduction' (Enabled 16 bits, full speed), 'Sensor Auto Reset' (Disabled), 'Negative Noise Threshold' (20), 'Low Baseline Reset' (5), 'Shield' (Disabled), 'Inactive Sensor Connection' (Ground), and 'Guard Sensor' (Disabled). The 'Voltage reference source' section shows 'Vref 1.024V (default)' selected.

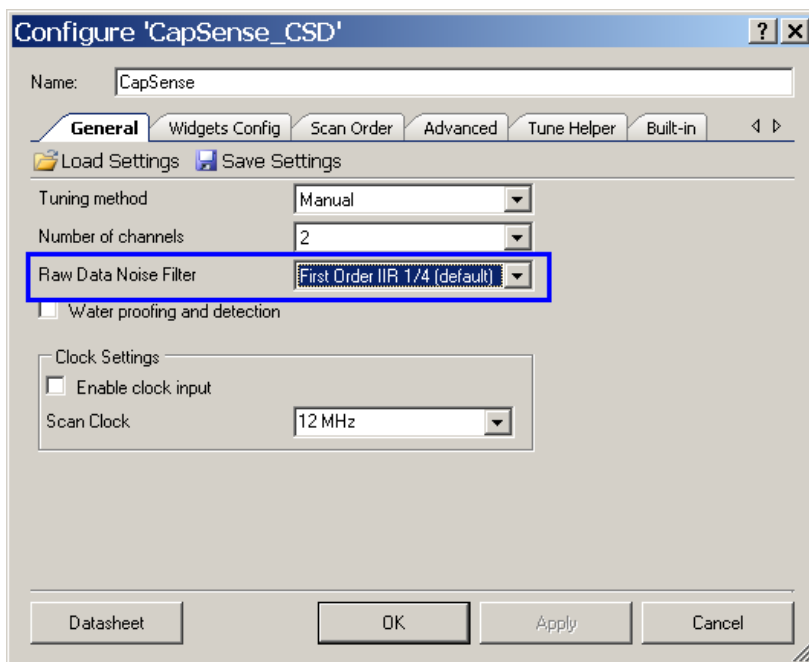
このパラメーターは、信号カウントに 8 ビット変数か 16 ビット変数を使用するかを決めます。信号カウントは図 5-5 に示すように、raw カウントとベースラインの差です。このパラメーターはウィジェット レベルで設定されます。ウィジェットのすべてのセンサーは同じ分解能を持ちます。

ほとんどの場合、適切な初期設定値が 8 ビットです。ただし、薄いオーバーレイまたは大きいセンサー領域による高信号レベルの設計では、16 ビットの分解能を選択する必要があります。スライダーが隣接センサーの相対信号レベルを用いて指の位置を計算し、信号が飽和して円滑でない動作を引き起こすため、16 ビットの分解能は必要になることがあります。

5.3.8 フィルタ セクション

5.3.8.1 raw データ ノイズ フィルタ

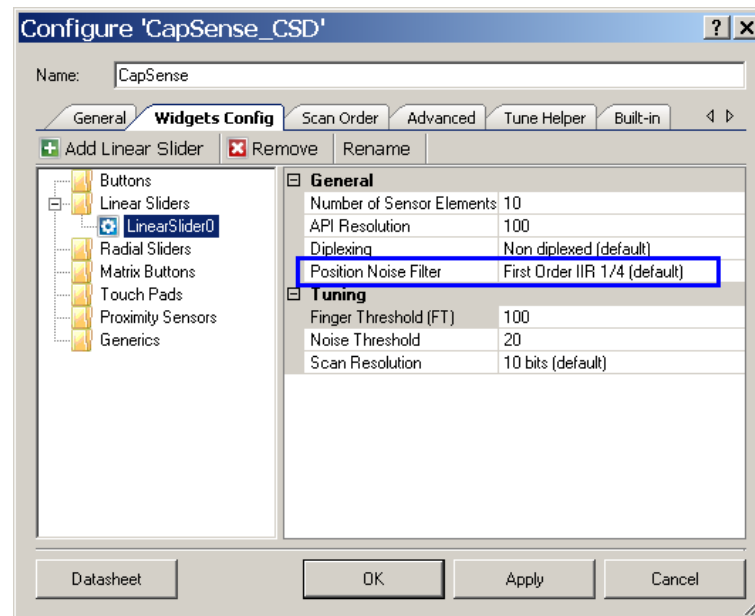
図 5-11. raw データ ノイズ フィルタ パラメーター



raw カウントは CapSense システムの未加工出力で、固有のノイズがあります。raw カウントのノイズは、ソフトウェア フィルタを使って最小限にすることができます。raw データ ノイズ フィルタ パラメーターは、いくつかのソフトウェア フィルタを選んで raw カウントおよび指の位置のデータを整理することができます。各フィルタ タイプの使い方は「[ソフトウェア フィルタリング](#)」を参照してください。

5.3.8.2 位置ノイズ フィルタ

図 5-12. 位置ノイズ フィルタ パラメーター



リニア スライダーやラジアル スライダーなどの一部のウィジェット タイプには、指位置出力があります。位置ノイズ フィルタ パラメーターは、いくつかのソフトウェア フィルタを選んで指位置出力を整理することができます。

5.3.9 高レベル パラメーターの推奨事項

次の推奨事項は最適なパラメーター設定を選択するための出発点となります。

- **指閾値:** センサーがオンの状態で raw カウントの 75%に設定
- **ノイズ閾値:** センサーがオフの状態で raw カウントの 40%に設定
- **負のノイズ閾値:** ノイズ閾値に等しく設定
- **ヒステリシス:** センサーがオンの状態で raw カウントの 15%に設定
- **低ベースライン リセット:** 10 に設定
- **センサー自動リセット:** 設計要件に応じて設定
- **デバウンス:** 設計要件に応じて設定
- **ウィジェット分解能:** スライダーを使用し、薄いオーバーレイまたは大きいセンサー領域により信号が飽和する場合は 16 ビットを選択し、それ以外の場合は 8 ビットを選択
- **フィルタ選択:** 適切なフィルタを選択するために、「ソフトウェア フィルタリング」を参照してください。

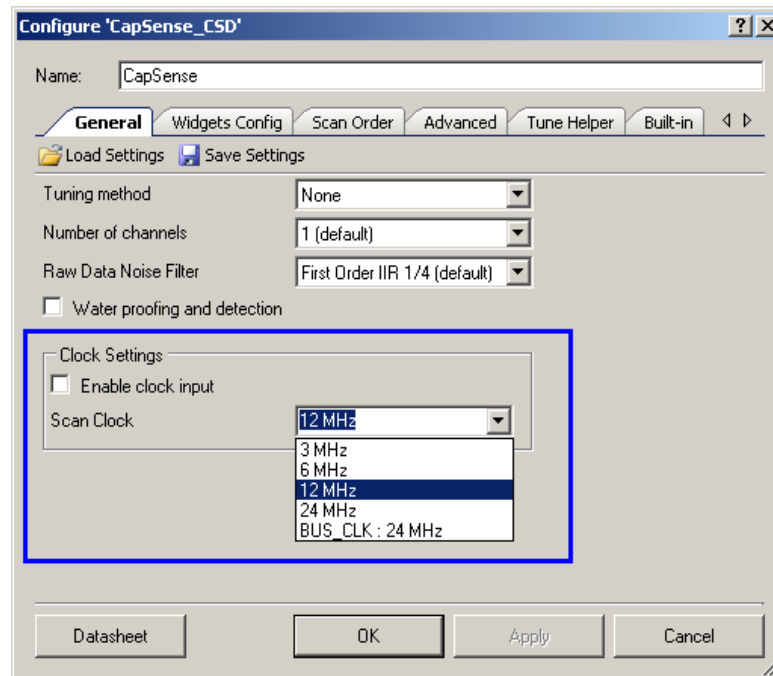
5.4 低レベル パラメーター

低レベル パラメーターは、物理層で CapSense システムの動作を定義し、静電容量から raw カウントへの変換に関連しています。

5.4.1 クロック設定

5.4.1.1 スキャン クロック

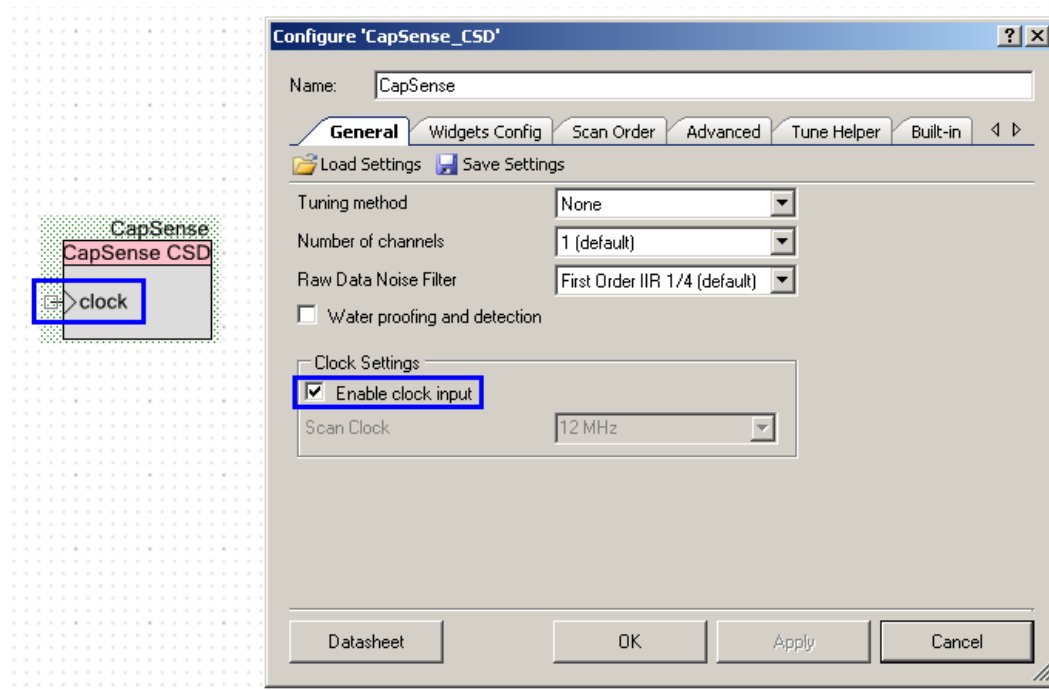
図 5-13. スキャン クロック パラメーター



スキャン クロック パラメーターは、CapSense_CSD ブロックのクロック ソースを選択します。スキャン クロックは CPU 周波数に依存しません。これはマスター クロックから派生するため、マスター クロックの周波数はスキャン クロック パラメーター以上にする必要があります。

5.4.1.2 クロック入力のイネーブル

図 5-14. 外部クロック入力パラメーター

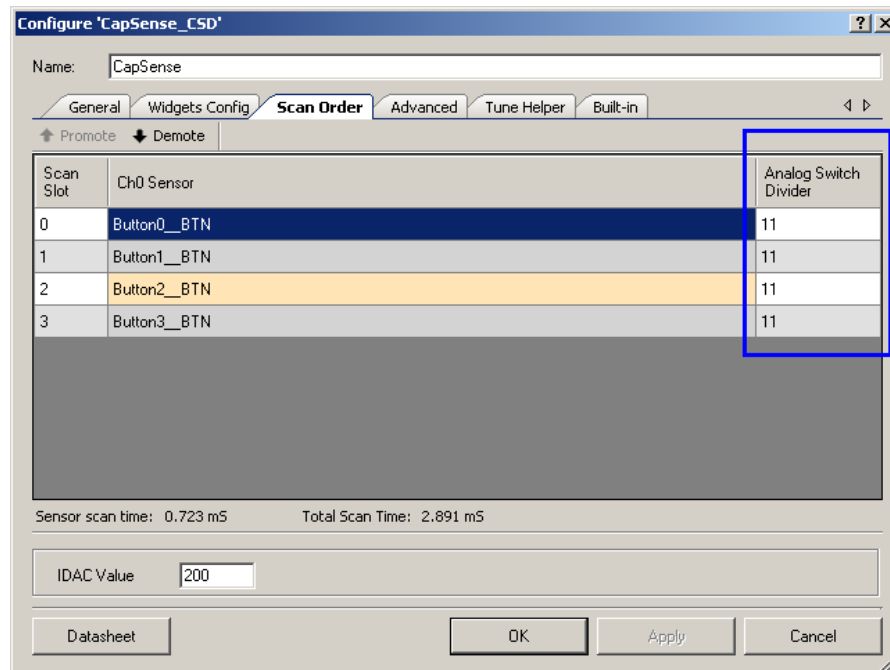


クロック入力のイネーブル パラメーターを選択することは、クロック ソースが CapSense_CSD ブロックの外部から供給されることを意味します。このオプションを選択した場合、図 5-14 に示すように、スキャン クロックの周波数は無効になり、端子が CapSense_CSD コンポーネントに現れます。システム内のデジタル信号および GPIO ピン経由で送られる外部信号は、この端子に接続できます。

5.4.2 アナログ スイッチ分周器

5.4.2.1 アナログ スイッチ分周器 (プリスケaler)

図 5-15. アナログ スイッチ分周期パラメーター



アナログ スイッチ分周器パラメーターは CapSense センサーのスイッチング周波数を設定します。CapSense センサーは、IDAC ソース モードの場合に、C_{MOD}と GND 間を継続的に切り替えます。IDAC シンクと外部抵抗モードの場合に、C_{MOD}と V_{DD}間を切り替えます。スイッチング クロック周波数は以下のように定義されます。

$$\text{スイッチング クロック} = \frac{\text{スキャン クロック}}{\text{アナログスイッチ分周器}} \quad \text{式 5}$$

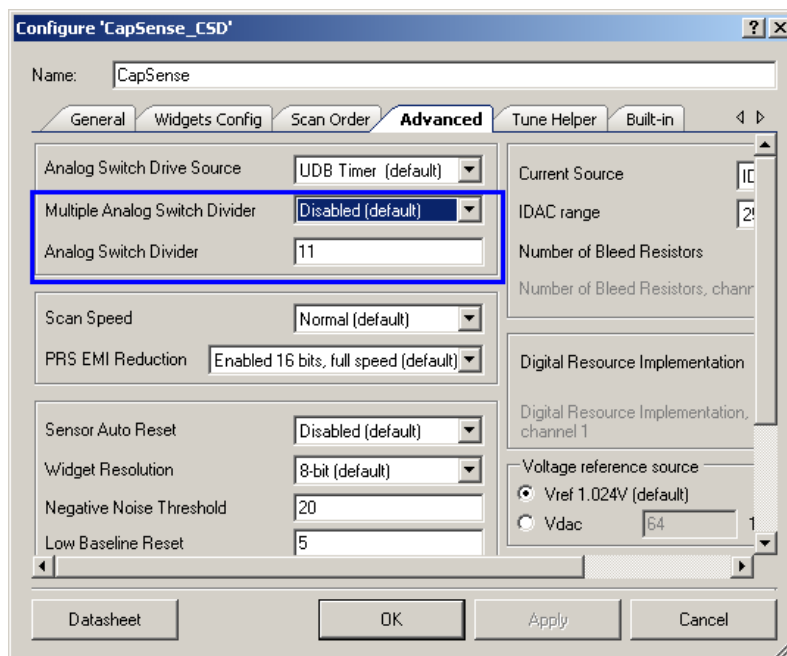
ここで、

スキャン クロック=CapSense_CSD ブロックへのソース クロック (「[クロック設定](#)」で説明)

スイッチング クロック=CapSense センサーが切り替わる周波数

5.4.2.2 複数のアナログ スイッチ分周器

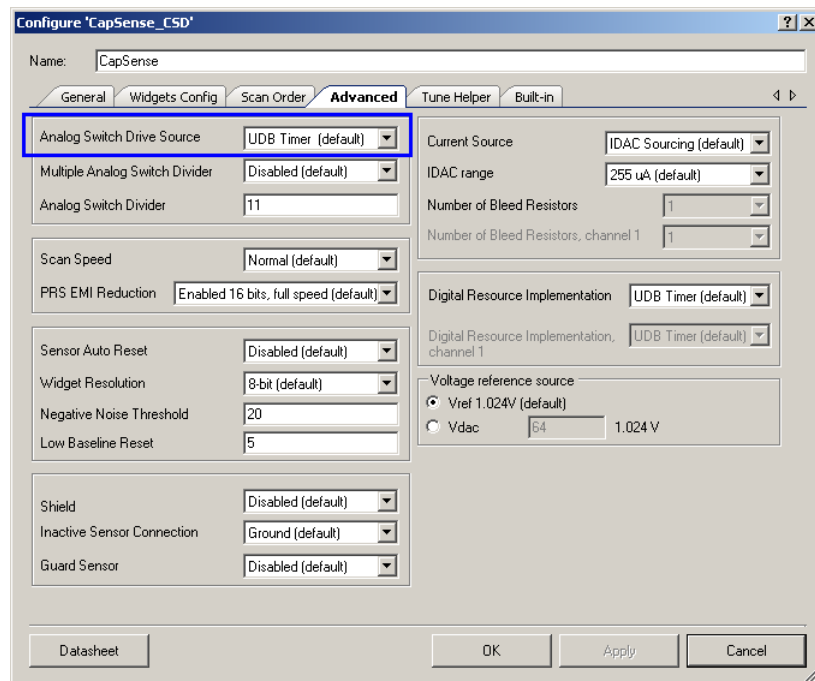
図 5-16. 複数のアナログ スイッチ分周器



CapSense_CSD には、各 CapSense センサーに対して個別のアナログ スイッチ分周器の値を選択するオプション、またはすべてのセンサーに共通の 1 つのアナログ スイッチ分周器の値を選択するオプションがあります。すべてのセンサーの C_P が同じになる場合は、共通のアナログ スイッチ分周器の値は適切になります。各センサーの C_P 間に大きなばらつきがある場合、複数のアナログ スイッチ分周器パラメーターを有効にする必要があります。複数のアナログ スイッチ分周器パラメーターを有効にした場合、図 5-15 に示したようにその値は「Scan Order」タブで設定されます。

5.4.2.3 アナログ スイッチ ドライブ ソース

図 5-17. アナログ スイッチ ドライブ ソース パラメーター



このパラメーターは、アナログ スイッチ分周器の実装方法を選択します。アナログ スイッチ分周器はタイマー ベースの分周器です。このタイマーの実装に使用されるリソースは次の 3 つのオプションから選択できます。

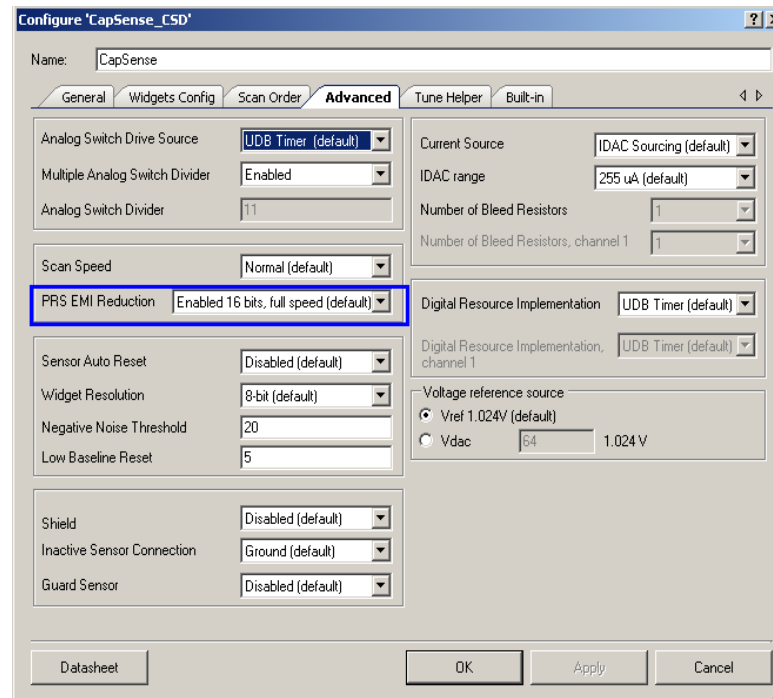
Direct (直接): このオプションを選択すると、アナログ スイッチ分周器が使用されず、スキャン クロック自身はスイッチング クロックになります。重要なリソースの使用が必要になる場合はこのオプションを選択しますが、設計が正常に実行されていることを確認しないといけません。

UDB Timer (UDB タイマー): このオプションを選択すると、アナログ スイッチ分周器のタイマーは UDB を使用して実装します。複数の UDB ブロックがあるため、これはデフォルト オプションです。

FF Timer (FF タイマー): このオプションを選択すると、アナログ スイッチ分周器のタイマーは固定機能デジタル ブロックを使用し実装します。

5.4.3 疑似乱数シーケンス (PRS)

図 5-18. PRS EMI 減少パラメーター



このパラメーターは、スイッチング クロックで疑似乱数シーケンス (PRS) を使用することができます。PRS は、スイッチングクロック周波数をその中心の周波数を上回ったり、下回ったりし、そのスペクトルを拡散して EMI を減少させます。次のオプションから選択できます。

Disabled (無効): このオプションは最高の SNR を提供しますので、アプリケーションに電磁干渉の減少が必要でない場合にお勧めします。

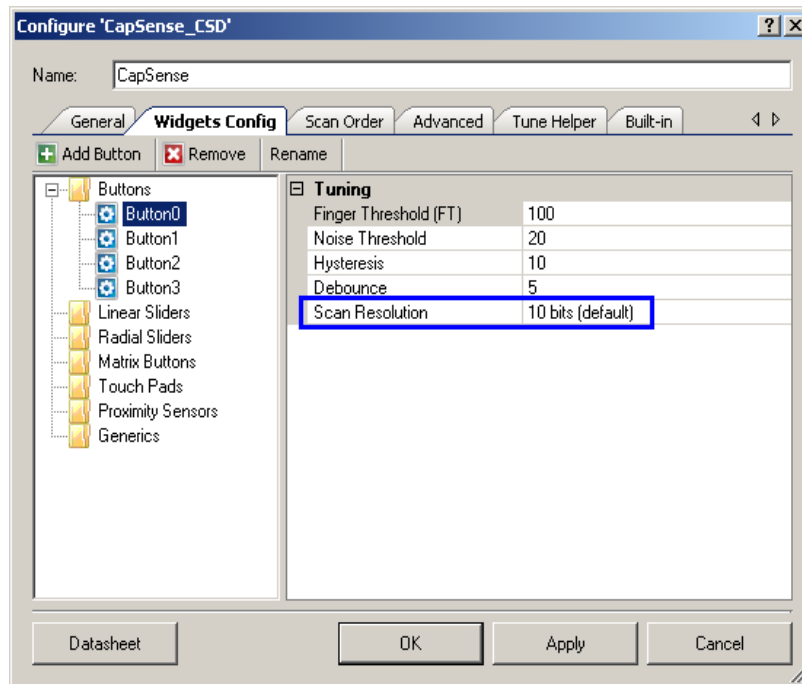
Enabled 8 bits (有効、8 ビット): 8 ビットでは 16 ビットより SNR が改善されますが、繰り返し期間が短くなるため、EMI が増加します。

Enabled 16 bits, full speed (default) (有効、16 ビット、フル スピード – デフォルト): 16 ビットはより低い SNR をもたらしませんが、EMI 減少は優れています。EMI の減少のために、このオプションをお勧めします。

Enabled 16 bits, 1/4 speed (有効、16 ビット、1/4 スピード): 「Enable 16 bits, full speed」より 4 倍速い PRS クロック出力が必要です。

5.4.4 スキャン分解能

図 5-19. スキャン分解能パラメーター

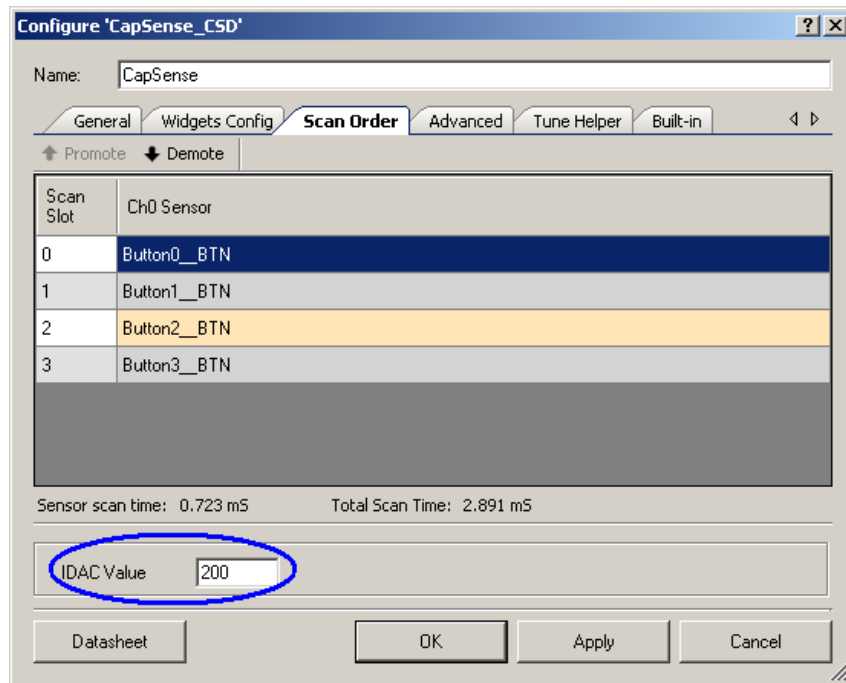


このパラメーターではスキャン分解能をビット数の単位で設定します。N ビットが 2 のスキャン分解能の場合、最大 raw カウント値は 2^{N-1} です。分解能を高くすると感度が向上しますが、センサーをスキャンするのに要する時間も長くなります。値の範囲は 8～16 です。

5.4.5 IDAC 電流

5.4.5.1 IDAC 値

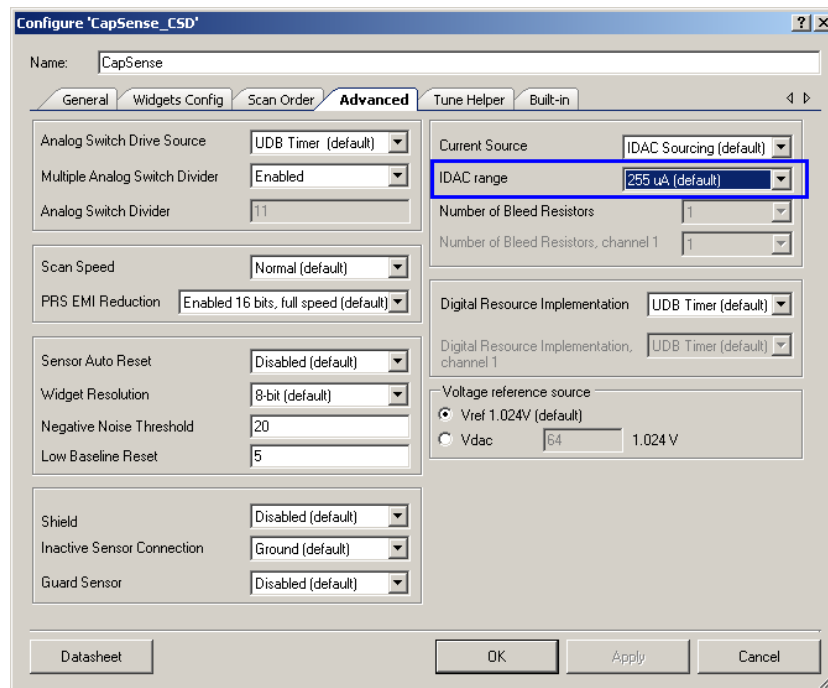
図 5-20. IDAC 値パラメーター



このパラメーターは IDAC 電流を設定します。IDAC 電流が低下すると、raw カウントは感度とともに増加します。N ビットが 2 のスキャン分解能の場合、最大 raw カウント値は 2^{N-1} です。IDAC 電流は、raw カウントが最大値の 50%~80%になるように設定する必要があります。

5.4.5.2 IDAC 範囲

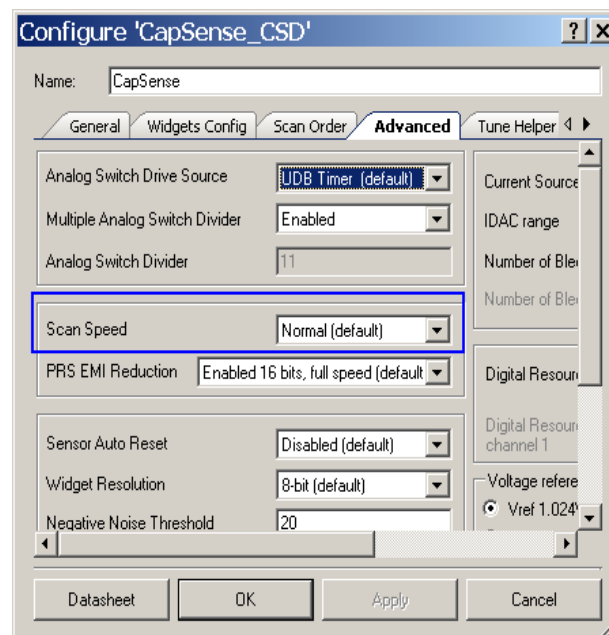
図 5-21. IDAC 範囲パラメーター



このパラメーターは IDAC 電流の範囲を選択します。必要な IDAC 電流は C_P に正比例します。高い C_P の場合は広い範囲を選択し、逆の場合もまた同様となります。値は 32 μ A、255 μ A、2048 μ A です。

5.4.6 スキャン速度

図 5-22. スキャン速度パラメーター



このパラメーターはセンサーのスキャン速度を設定します。速いスキャン速度は短い応答時間をもたらします。スキャン速度が遅いと、SNR が改善され、電源や温度の変化による影響を受けにくくなります。値は Very Fast (非常に速い)、Fast (速い)、Normal (通常)、Slow (遅い) です。

スキャン速度クロックは以下の式でカウント クロックを設定します。

$$\text{カウント クロック} = \frac{\text{スキャン クロック}}{\text{スキャン速度分周期}} \quad \text{式 6}$$

ここで、

スキャン クロック=CapSense_CSD ブロックのソース クロック

スキャン速度分周器=1 (非常に速い)、2 (速い)、3 (通常)、4 (遅い)

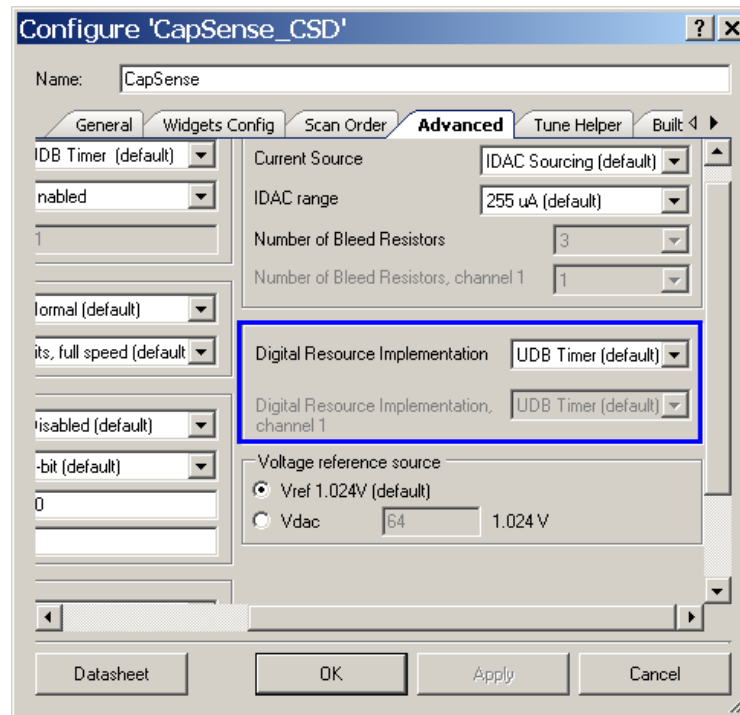
スキャン時間は、スキャン速度、スキャン分解能、スキャン クロックおよび CPU クロックによって異なります。表 5-2 は、スキャンクロックが 24MHz および CPU クロックが 48MHz で、単一のセンサーをスキャンする時間を示します。

表 5-2. 単一センサーのスキャン時間 (単位: μs) 対スキャン速度と分解能

分解能 (単位: ビット)	スキャン速度			
	非常に速い	速い	通常	遅い
8	58	80	122	208
9	80	122	208	377
10	122	208	377	718
11	208	377	718	1400
12	377	718	1400	2770
13	718	1400	2770	5500
14	1400	2770	5500	10950
15	2770	5500	10950	21880
16	5500	10950	21880	43720

5.4.7 デジタル リソースの実装

図 5-23. デジタル リソース実装パラメーター



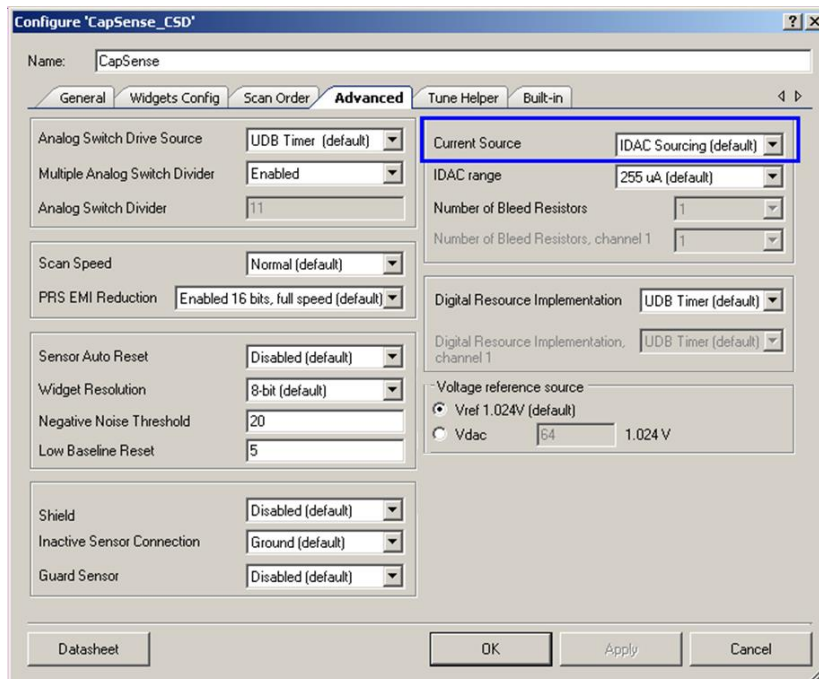
スキャン速度設定は、タイマー ベースの分周器を使ってカウント クロックを作成します。このパラメーターでは、この分周器の実装用のリソースを選択できます。デジタル リソース実装をチャンネルごとに選択することができます。オプションは以下のとおりです。

UDB Timer (UDB タイマー): このオプションを選択すると、タイマーは UDB を使用して実装します。複数の UDB ブロックがあるため、これはデフォルト オプションになります。

FF Timer (FF タイマー): このオプションを選択すると、タイマーは固定機能デジタル ブロックを使用して実装します。

5.4.8 電流ソース

図 5-24. 電流ソース パラメーター



Configure 'CapSense_CSD'

Name: CapSense

General Widgets Config Scan Order **Advanced** Tune Helper Built-in

Analog Switch Drive Source: UDB Timer (default)

Multiple Analog Switch Divider: Enabled

Analog Switch Divider: 11

Scan Speed: Normal (default)

PRS EMI Reduction: Enabled 16 bits, full speed (default)

Sensor Auto Reset: Disabled (default)

Widget Resolution: 8-bit (default)

Negative Noise Threshold: 20

Low Baseline Reset: 5

Shield: Disabled (default)

Inactive Sensor Connection: Ground (default)

Guard Sensor: Disabled (default)

Current Source: IDAC Sourcing (default)

IDAC range: 255 uA (default)

Number of Bleed Resistors: 1

Number of Bleed Resistors, channel 1: 1

Digital Resource Implementation: UDB Timer (default)

Digital Resource Implementation, channel 1: UDB Timer (default)

Voltage reference source

☒ Vref 1.024V (default)

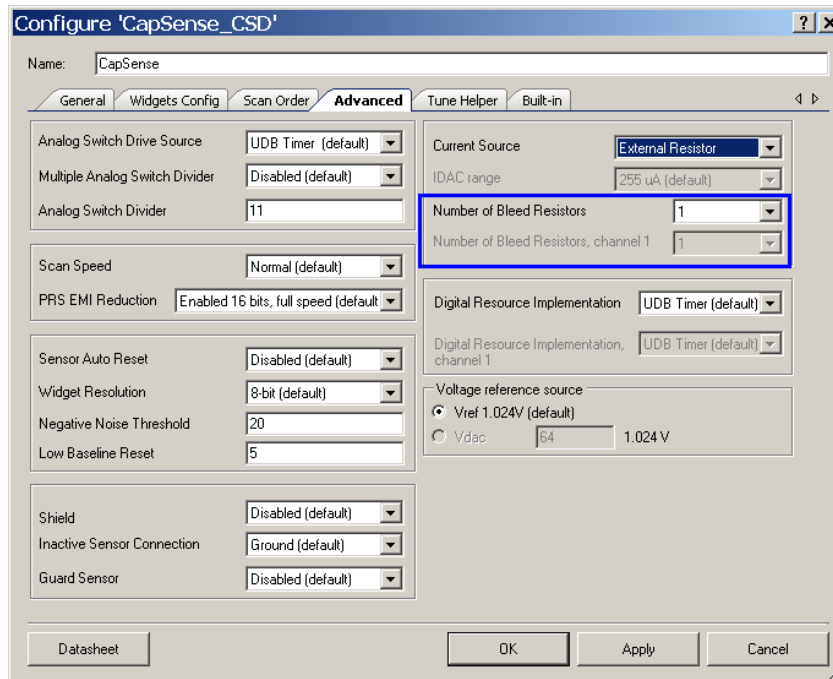
☐ Vdac: 64 1.024 V

Datasheet OK Apply Cancel

このパラメーターは電流ソース方式を選択します。値は IDAC Sourcing (IDAC ソース)、IDAC Sinking (IDAC シンク) および External Resistor (外部抵抗) です。これらの電流ソース方式の詳細説明は「[電流ソース方式](#)」を参照してください。

5.4.8.1 ブリード抵抗数

図 5-25. ブリード抵抗数パラメーター

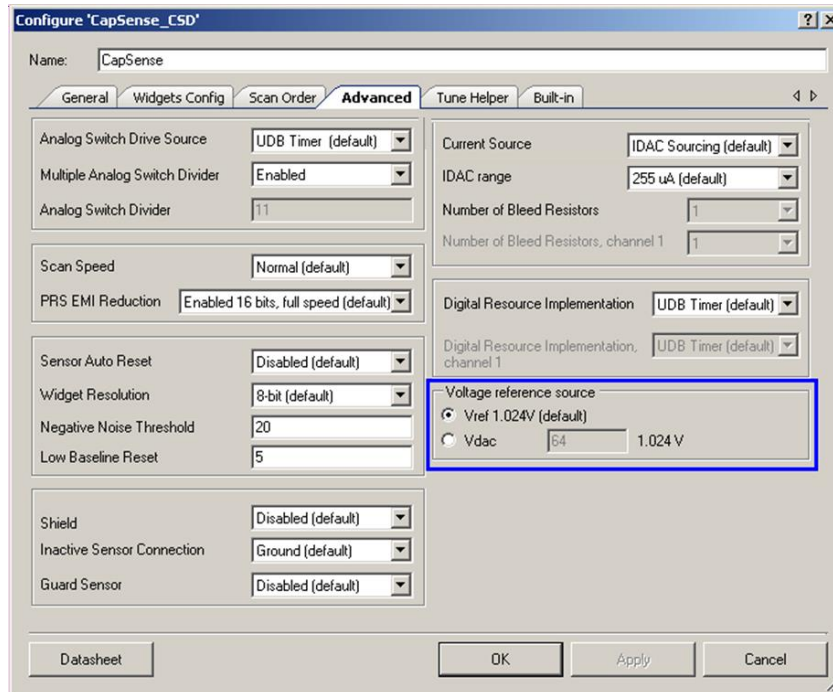


The screenshot shows the 'Configure CapSense_CSD' dialog box with the 'Advanced' tab selected. The 'Name' field is 'CapSense'. The 'Current Source' is set to 'External Resistor'. The 'IDAC range' is '255 uA (default)'. The 'Number of Bleed Resistors' is set to '1', and 'Number of Bleed Resistors, channel 1' is set to '1'. These two fields are highlighted with a blue rectangle. Other settings include 'Analog Switch Drive Source' as 'UDB Timer (default)', 'Multiple Analog Switch Divider' as 'Disabled (default)', 'Analog Switch Divider' as '11', 'Scan Speed' as 'Normal (default)', 'PRS EMI Reduction' as 'Enabled 16 bits, full speed (default)', 'Sensor Auto Reset' as 'Disabled (default)', 'Widget Resolution' as '8-bit (default)', 'Negative Noise Threshold' as '20', 'Low Baseline Reset' as '5', 'Shield' as 'Disabled (default)', 'Inactive Sensor Connection' as 'Ground (default)', and 'Guard Sensor' as 'Disabled (default)'. The 'Voltage reference source' is set to 'Vref 1.024V (default)'. The 'Digital Resource Implementation' and 'Digital Resource Implementation, channel 1' are both set to 'UDB Timer (default)'. The 'Datasheet' button is at the bottom left, and 'OK', 'Apply', and 'Cancel' buttons are at the bottom right.

外部抵抗方式は、PSoC デバイスの外部ブリード抵抗を必要とします。このパラメーターでは外部ブリード抵抗の数を指定します。最大 3 個のブリード抵抗が接続できます。2 チャネルの静電容量センシングの場合、それぞれのチャンネルには個別のブリード抵抗一式が必要です。

5.4.9 電圧基準ソース

図 5-26. 電圧基準ソース パラメーター



The screenshot shows the 'Configure CapSense_CSD' dialog box with the 'Advanced' tab selected. The 'Voltage reference source' section is highlighted with a blue box. It contains two radio buttons: 'Vref 1.024V (default)' which is selected, and 'Vdac' which is unselected. Next to 'Vdac' is a text box containing the value '64' and a label '1.024 V'.

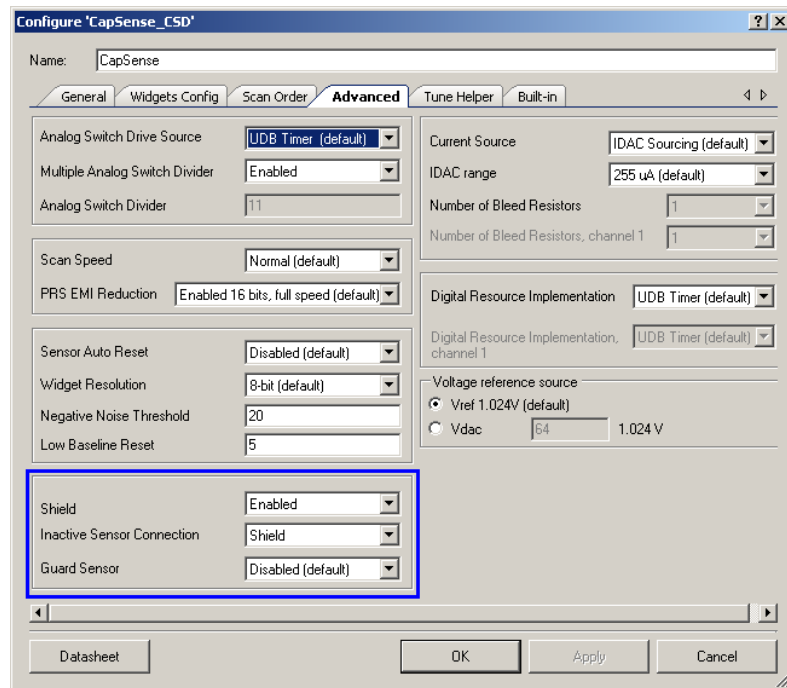
このパラメーターで電圧基準ソースを指定できます。オプションは以下のとおりです。

Vref 1.024V: これはバンドギャップ電圧です。

Vdac: これは 0~4V の範囲内のいずれの電圧を選択できます。IDAC ソース方式を使用する場合に役立ちます。電圧基準を増加させることにより、センサーの電圧振幅を増加させることが可能です。これは指伝導ノイズに対してより良い耐性を与えます。この設定は 1 つの DAC リソースを使用します。

5.4.10 シールド電極およびガード センサー

図 5-27. シールド電極およびガード センサー パラメーター



5.4.10.1 シールド

シールド電極を使用するデザインでは、このパラメーターを有効にする必要があります。「[耐水性デザイン](#)」を参照してください。

5.4.10.2 非アクティブなセンサー接続

このオプションは、センサーがスキャンされていないときのセンサー接続を決定します。値は以下のとおりです。

GND: この設定はノイズが少なくなるので、推奨されている設定です。

Shield (シールド): シールド電極を使用し、スライダーなどのセンサーに隣接したウィジェットを含んでいるデザインは、この設定を選択します。非アクティブなセンサーをシールド電極に接続することで、水がセンサー上にある場合の誤トリガを防ぎます。

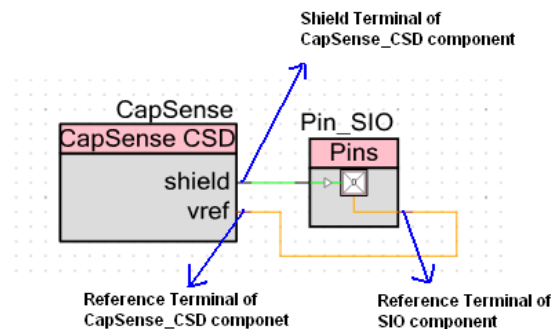
HI-Z: 非アクティブなセンサーを高インピーダンスのままにします。この設定を選択すると、センサーに接続されているグラウンドによる総寄生容量が減少します。

5.4.10.3 ガード センサー

ガード センサーを使用するデザインでは、このパラメーターを有効にする必要があります。「[耐水性デザイン](#)」を参照してください。

5.4.10.4 シールド電極用に SIO ピンの使用

図 5-28. シールド電極で使用する SIO ピン

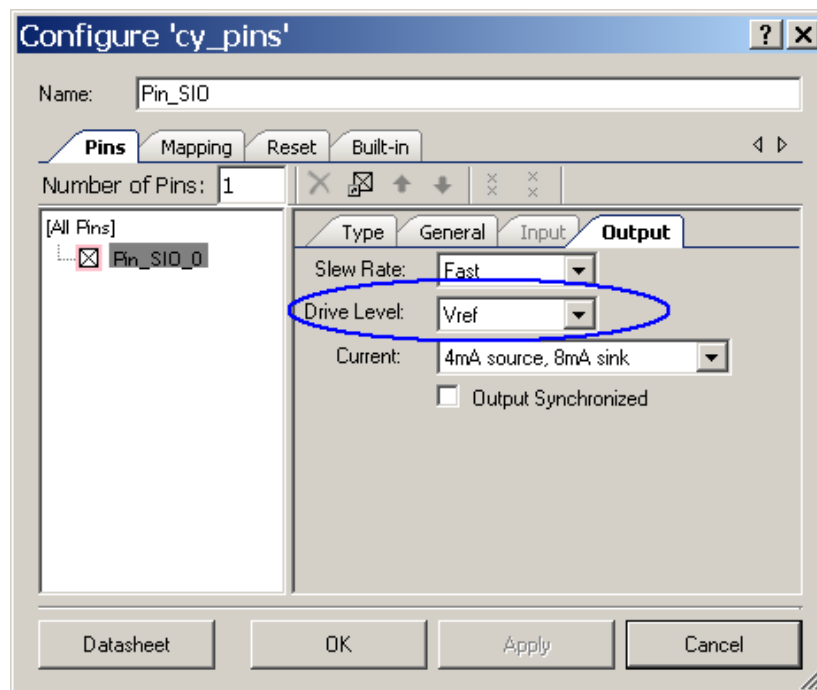


シールドが有効になると、図 5-28 に示すようにシールド端子がコンポーネントに現れます。

I/O ピンをこの端子に接続する必要があります。電流ソースに IDAC シンクまたは外部抵抗方式を使用する場合、通常の GPIO ピンを使用します。IDAC ソーシング方式の使用の場合は、シールドを SIO ピンに接続する必要があります。SIO ピンを使用して、シールド信号をセンサー信号に整合します。

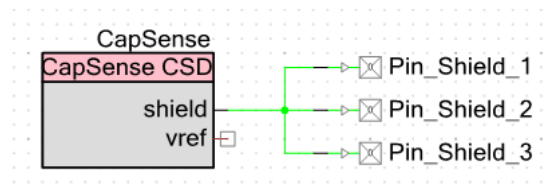
シールド電極として SIO ピンを使用するには、コンポーネント カタログからデジタル出力ピンをドラッグします。コンフィギュレーションウィンドウを開いて、以下のように「Drive Level」(駆動レベル) オプションを「Vref」に設定します。

図 5-29. ピン コンポーネントを SIO にするために「Drive Level」を設定



5.4.10.5 複数のシールド電極ピン

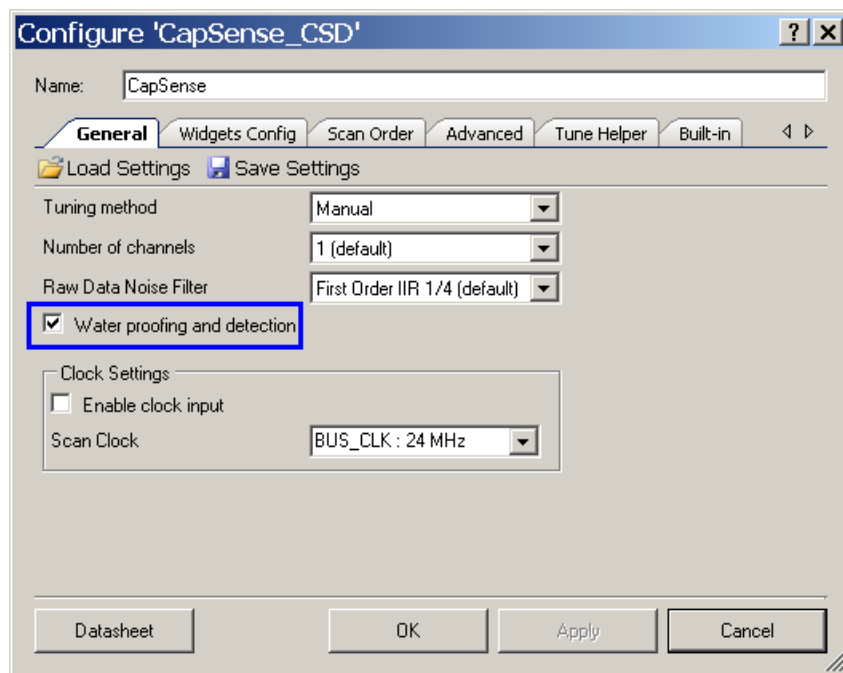
図 5-30. 複数のシールド電極ピン



シールド領域が大きい場合、1 本の電極ピンでシールド電極を駆動するのに十分ではないことがあります。シールド電極が完全に充電および放電されることを確認する必要があります。そのために、シールド電極ピンを探り、オシロスコープで監視します。通常のプロープがピンに大きい静電容量を加えることがありますので、10x モードに設定する必要があります。FET プロープをお勧めします。シールド電極が完全に充電および放電されない場合は、複数の I/O ピンを使用してシールド電極を駆動します。I/O ピン (GPIO と SIO の両方) には、4mA の駆動力があります。

5.4.10.6 耐水性および検知

図 5-31. 耐水性および検知パラメーター



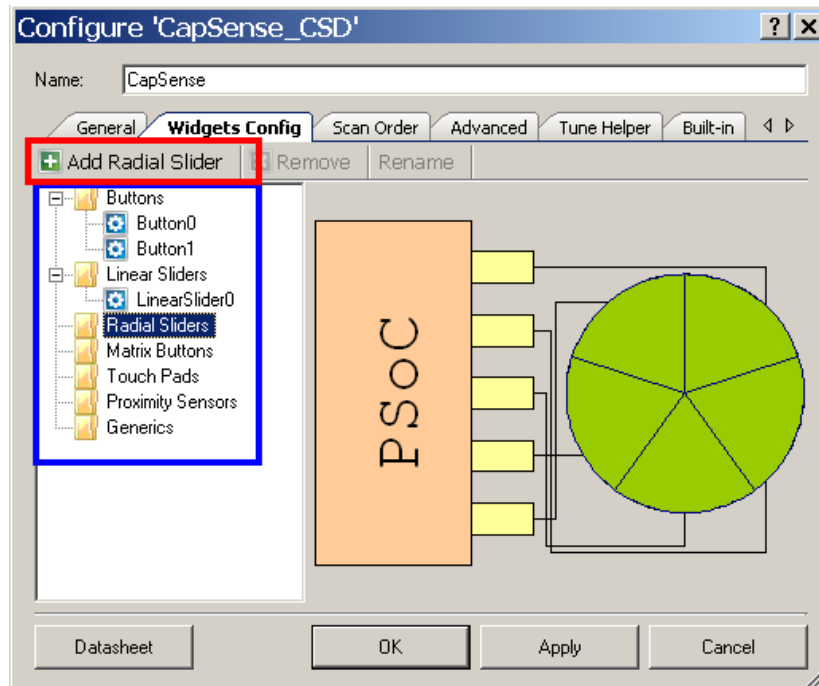
シールド電極とガードセンサーを有効にするもう 1 つの方法は、「General」タブの「Water Proofing and Detection」パラメーターを選択することです。このボックスをチェックすると、シールド電極とガードセンサーの両方は「Advanced」タブで有効にされます。

5.5 ウィジェットのコンフィギュレーション

「Widgets Config」タブでは、ウィジェットを追加し、設定できます。

5.5.1 ウィジェットの追加

図 5-32. ウィジェットの追加



ウィジェットを追加するには、ウィジェットの種類を選択して、「Add」をクリックします。

5.5.2 センサー エLEMENT 数

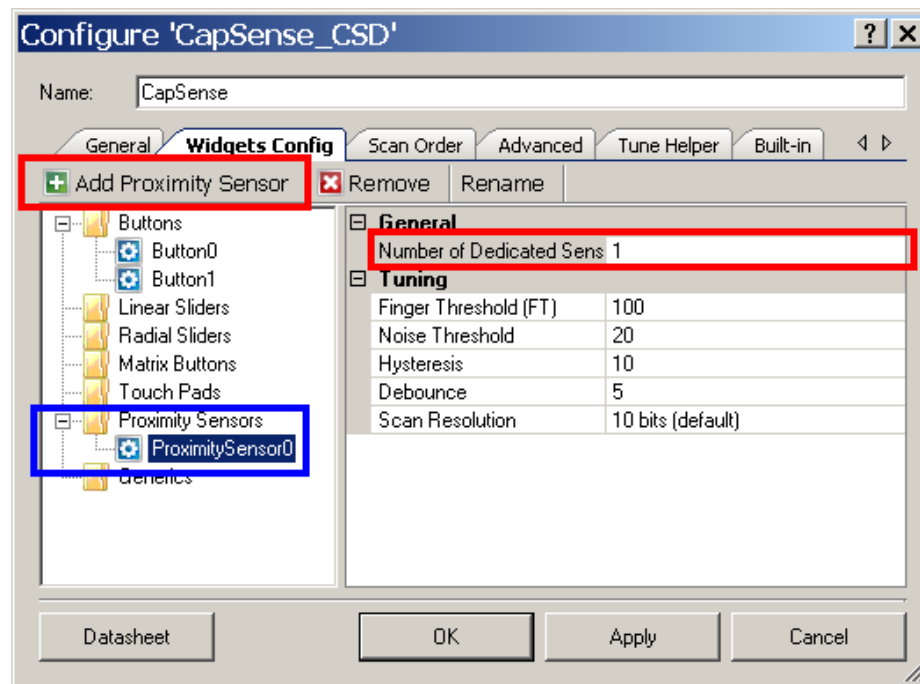
各ウィジェットのセンサー数は、このタブで設定されます。

5.5.3 API 分解能

スライダーやタッチ パッドなどのいくつかのウィジェット タイプは、センサーのオン/オフ状態の代わりに指位置のデータを使用します。指位置のデータの分解能は、API 分解能と呼ばれます。

5.5.4 近接センサー

図 5-33. 近接センサー パラメーター



近接センサーは、ユーザーがセンサーに触れなくてもセンサー近くの手を検知します。近接センサーは、プリント基板のユーザー インターフェースの周辺に配置している長い配線を含むことがあります。または、センサーを短絡させ、それらを単一のセンサーとしてスキャンすることが可能です。これら 2 つの方法を組み合わせ、ユーザー インターフェースの周辺に配置されている近接センサーの配線を他のセンサーと短絡させ、それらを単一のセンサーとしてスキャンすることができます。

注: PSoC Creator の初期設定では、近接センサーは無効にされ、その他すべてのセンサーは有効にされています。次のコードの一部のように、スキャンする前に有効にする必要があります。

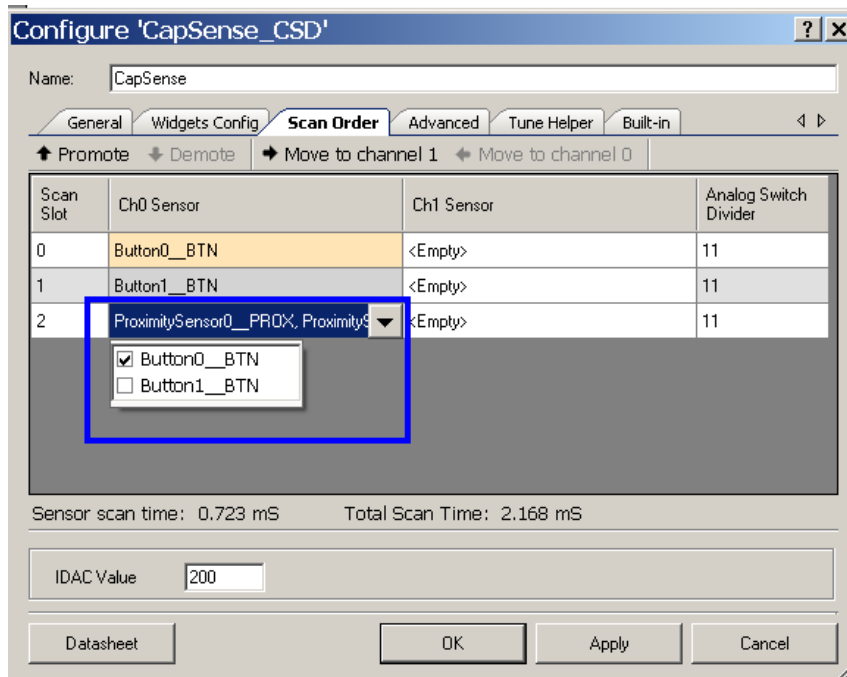
```

17
18 CyGlobalIntEnable; /* Uncomment this line to enable global interrupts. */
19 CapSense_Start();
20 CapSense_EnableWidget(CapSense_PROXIMITYSENSOR0_PROX); /*Enabling proximity sensor explicitly*/
21 CapSense_InitializeAllBaselines();
22
23 for(;;)
24 {
25     if ( CapSense_IsBusy() == 0 )
26     {
27         CapSense_UpdateEnabledBaselines();
28         CapSense_CheckIsAnyWidgetActive();
29         CapSense_ScanEnabledWidgets();
30     }
31 }
32
--

```

5.5.4.1 専用センサー エLEMENT数

図 5-34. 近接センサーと既存センサーの組み合わせ

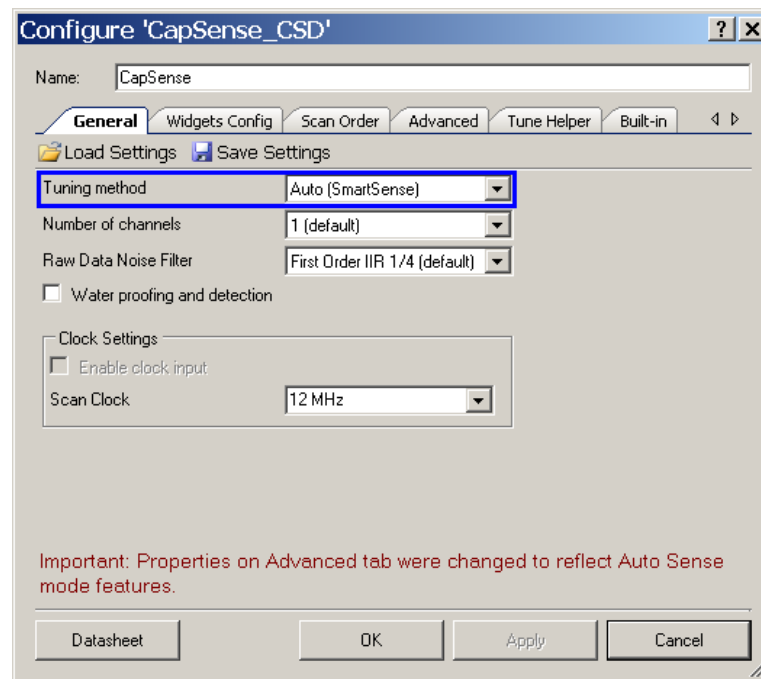


既存センサーと専用の近接センサーを組み合わせるには、図 5-33 のように「Widgets Config」タブの「Number of Dedicated Sens」を設定します。その後、「Scan Order」タブのドロップダウン メニューで近接センサーと組み合わせる既存のセンサーを選択します。

センサーの組み合わせはそれらの動作に影響を与えませんが、結果として近接センサーは C_P が高く、スキャン時間が長くなります。従って長いスキャン間隔を防ぐために、近接センサーは他のセンサーより少ない割合でスキャンする必要があります。

5.6 チューニング方法

図 5-35. チューニング方法パラメーター



このパラメーターはで CapSense システムの調整方法を選択できます。すべての調整方法では、チューナ GUI を使用して CapSense 信号を監視することができます。値は以下のとおりです。

Auto (自動): SmartSense 自動調整アルゴリズムはすべてのパラメータを最適な値に設定します。自動調整モードを選ぶと、次のパラメーターを変更することができません。

表 5-3. 自動調整モードの場合の固定パラメーター

パラメーター	設定
電流ソース	IDAC ソース/IDAC シンク 外部抵抗方式は対応していない
IDAC 範囲	255 μ A
複数アナログ スイッチ分周器	有効
スキャン速度	通常
PRS	有効、16 ビット、フル スピード

Manual (手動): すべてのパラメーターを手動で調整する必要があります。

None (なし): パラメーターが固定で、フラッシュに保存されています。手動チューニングか自動チューニングのいずれかの方法を使用して、固定パラメーターを調整する必要があります。

5.6.1 感度

感度パラメーターは、SmartSense 自動調整アルゴリズムのために C_F 値を設定します。感度値は、 C_F を決めるために 0.1pF の倍数で設定されます。感度値が 1 の場合は C_F が 0.1pF に、感度値が 4 の場合は C_F が 0.4pF になります。

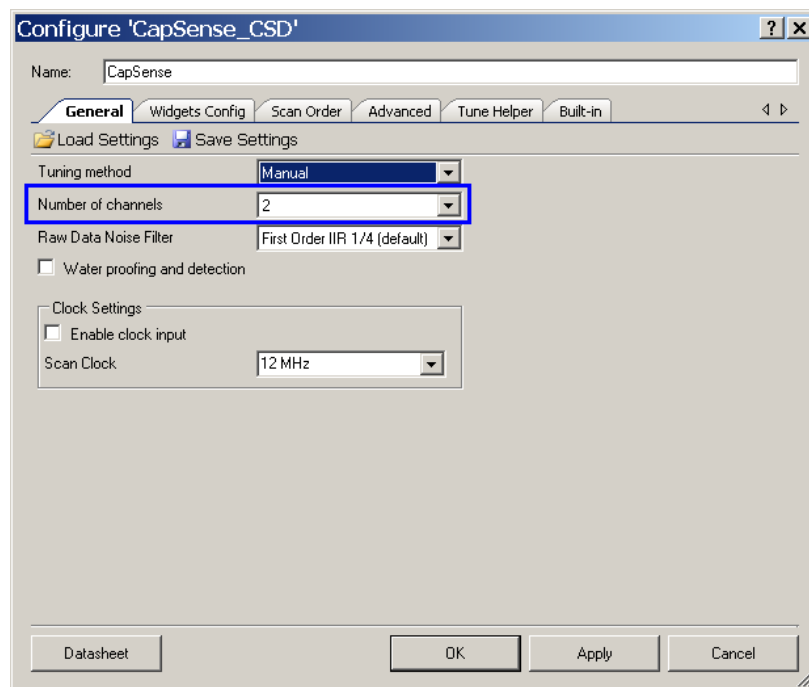
SmartSense 自動調整は感度パラメーターを用いて表 5-4.に示すパラメーターを計算します。

表 5-4. 感度を用いて自動調整で設定するパラメーター

パラメーター	計算タイミング
アナログ スイッチ分周器	CapSense_CSD の起動時に 1 度
IDAC 値	CapSense_CSD の起動時に 1 度
スキャン分解能	CapSense_CSD の起動時に 1 度
指閾値	センサーのスキャン中、継続的に計算
ノイズ閾値	センサーのスキャン中、継続的に計算
ヒステリシス	センサーのスキャン中、継続的に計算

5.7 チャネル数

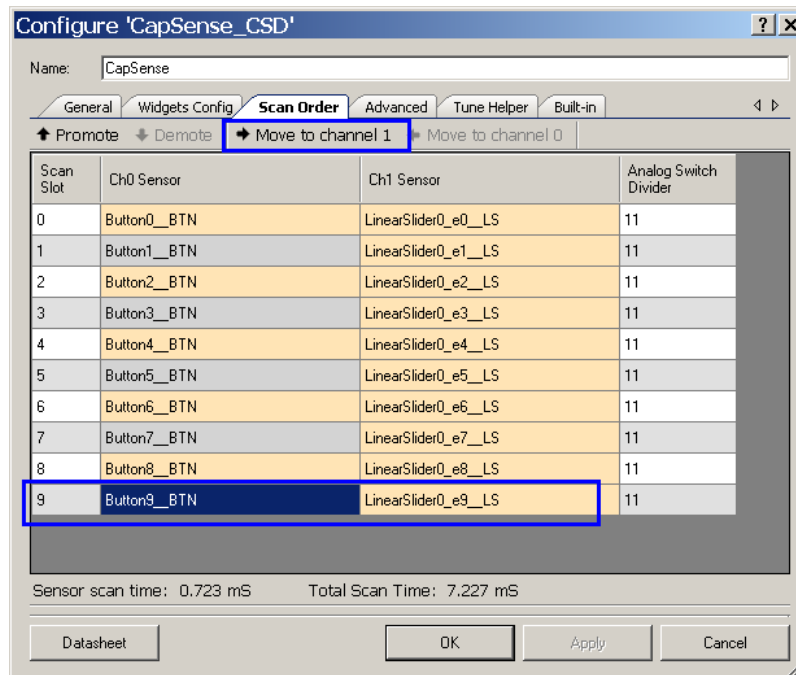
図 5-36. チャネル数



このパラメーターでは 2 チャネル デザインを実装できます。

5.7.1 チャンネル 1 / チャンネル 0 への変更

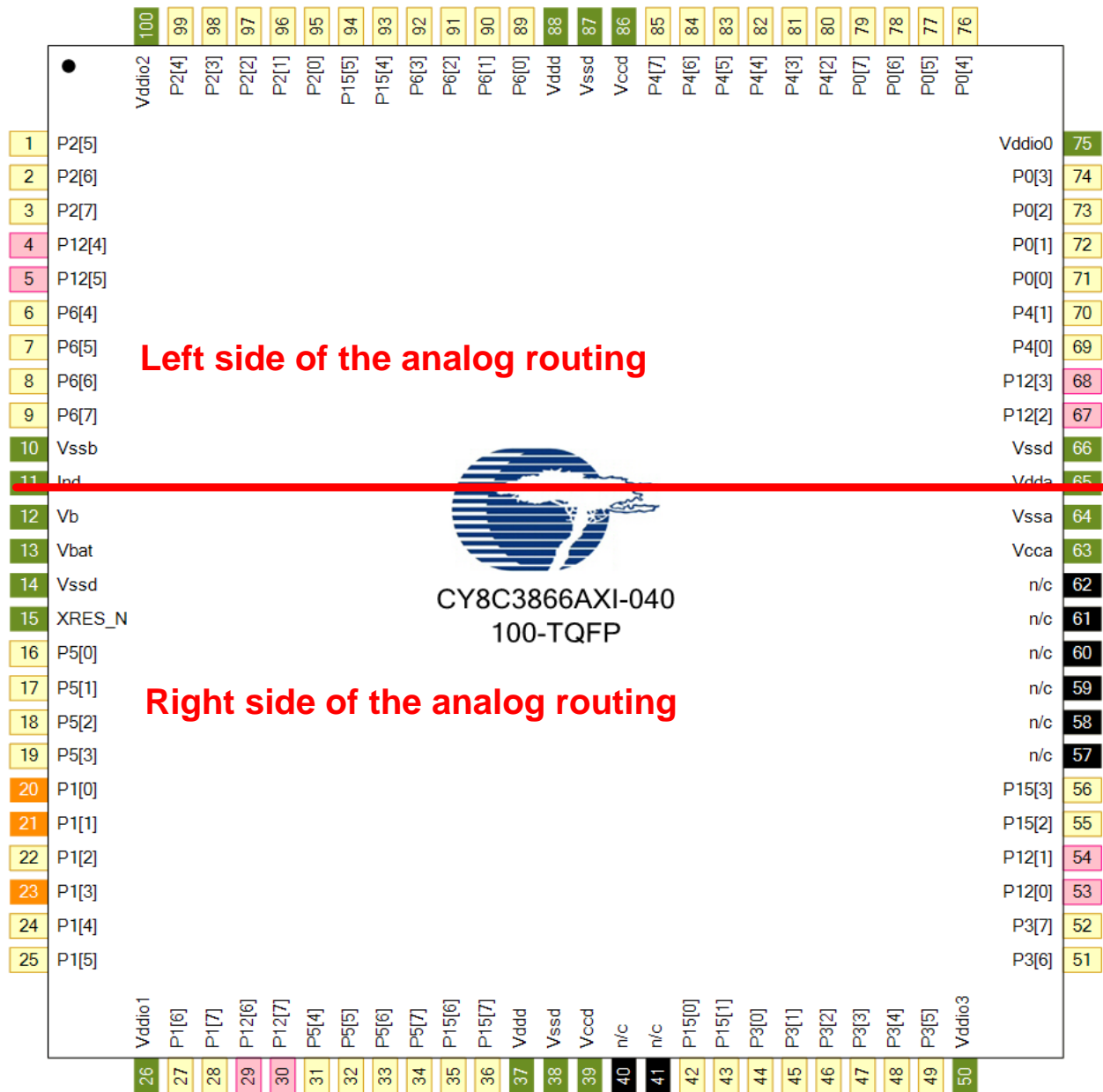
図 5-37. センサーのチャンネル割り当ての設定



このパラメーターでは、各センサーにチャンネル割り当てを指定できます。「Scan Order」タブで所望するセンサーを選んで、「Move to Channel 1」または「Move to Channel 0」をクリックします。

5.7.22 チャンネル デザインのピン割り当て

図 5-38. 2 チャンネル設計のピン割り当て



PSoC の各 GPIO は、内部ルーティング バスとの接続に応じて、アナログ ルーティングの左側か右側のどちらかの一部です。詳細はデバイス データシートの「Analog Routing」節を参照してください。図 5-38 は PSoC のピン配置図であり、GPIO およびそれに対応するアナログ ルーティングでの側を示します。

2 チャンネル デザインでは、1 本のチャンネルに割り当てられたすべてのセンサーをルーティングの片側に配置し、もう 1 本のチャンネルに割り当てられたすべてのセンサーをルーティングの反対側に配置する必要があります。デザインに 1 本だけのチャンネルを使用する場合、C_{MOD} およびブリード抵抗ピンをすべてのセンサーと同じルーティングの片側に配置する必要があります。

5.8 チューナーGUI

CapSense_CSD コンポーネントの便利な機能の一つは、CapSense 信号を監視する組み込み GUI です。チューナーGUI は自動チューニングと手動チューニング両方に対応しています。「[CapSense 性能のチューニング](#)」では、チューニング プロセスを説明します。チューナーGUI は MiniProg3 を介して PC の USB に接続し、I²C を使用してデバイスと通信します。

図 5-39. I²C-USB ブリッジとしての MiniProg3

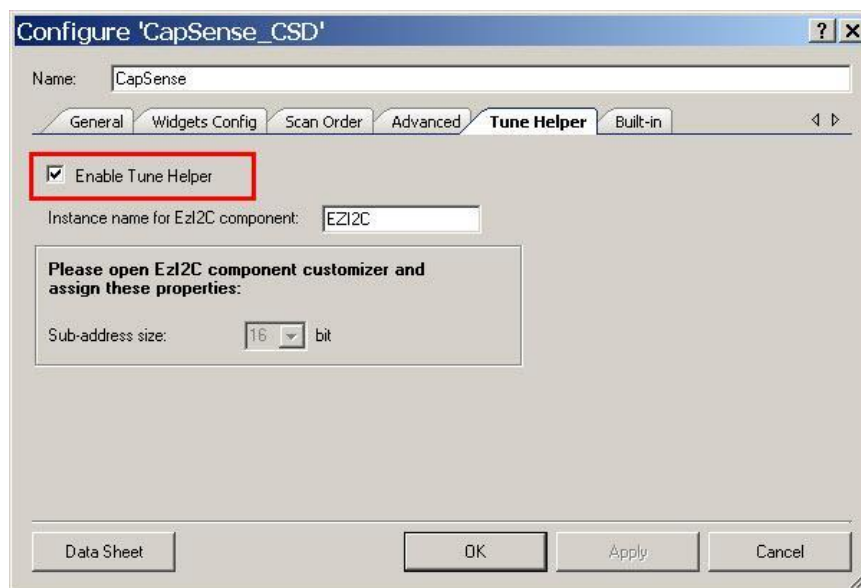


5.8.1 セットアップ

注: PSoC Creator プロジェクトをビルドする前に、ステップ 1~5 を完了してください。

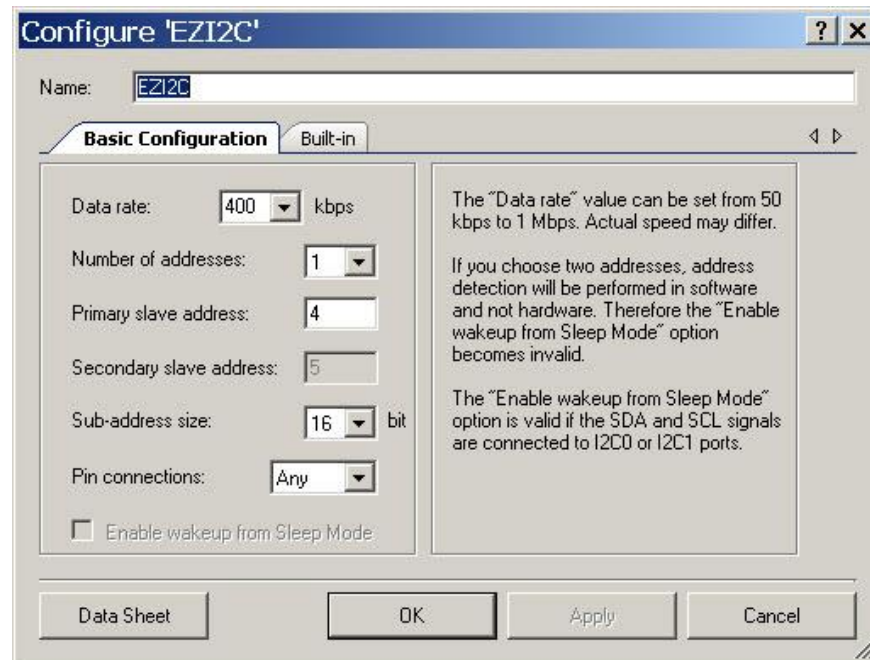
1. [チューニング方法](#)を自動と手動に設定します。「None」を選択すると、チューナーGUI は動作しません。
2. [ウィジェットを追加](#)します。
3. 「Tune Helper」タブで「Enable Tuner Helper」を選択します。

図 5-40. チューナーのイネーブル



4. EZI2C コンポーネントを配置して設定します。

図 5-41. EZI2C コンポーネントの名前を変更



5. 以下のコード セクションをプロジェクトに追加します。

```
#include <device.h>
void main()
{
  CyGlobalIntEnable; /* Global Interrupt enable */

  CapSense_TunerStart(); /* CapSense Block and Tuner Communication power up and
  Initialization */

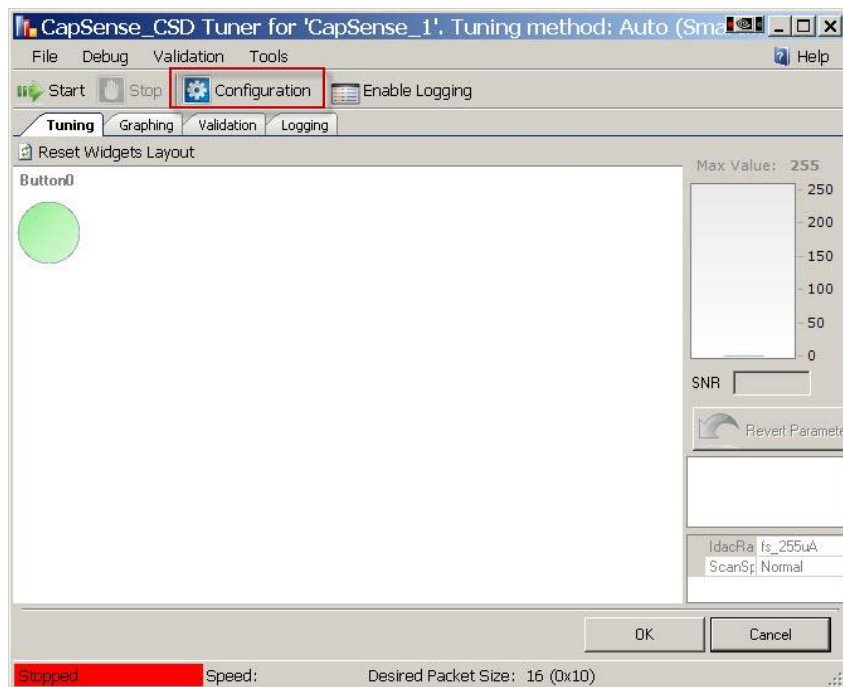
  While(1)
  {
    CapSense_TunerComm(); /*Scan all the sensors and send the result to PC */
  }
}
```

注: デバイスをプログラムして、MiniProg3 を正しく接続させた後に、以下の手順を実行してください。

1. CapSense_CSD コンポーネントを右クリックして、「Launch Tuner」をクリックします。

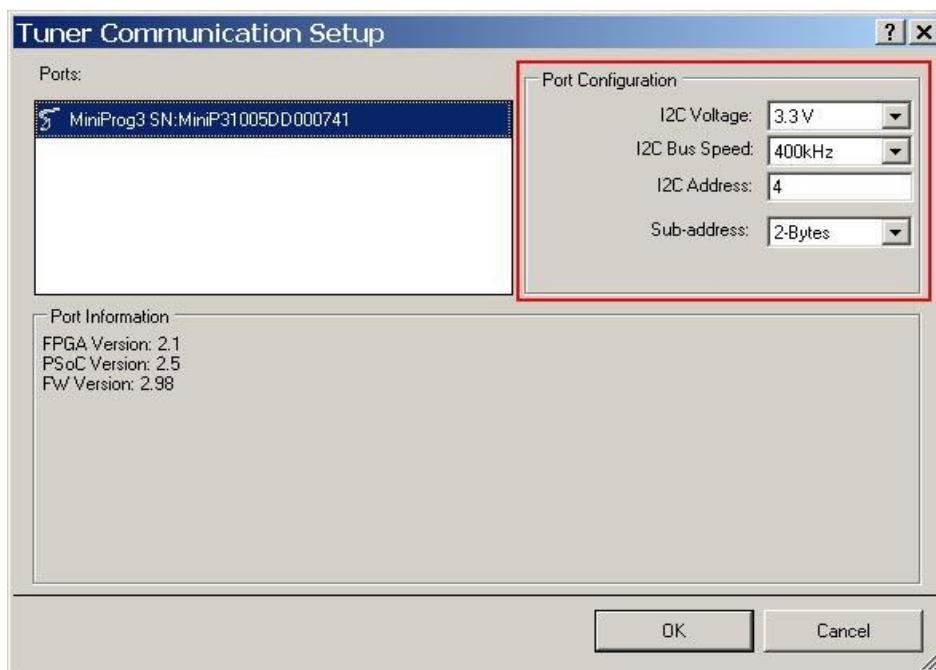
2. チューナーのウィンドウで、「Configuration」をクリックします。

図 5-42. チューナーGUI



3. 「Tuner Communication Setup」ウィンドウで MiniProg3 をクリックして、下図に示すようにパラメーターを設定します。

図 5-43. 「Tuner Communication Setup」ウィンドウ

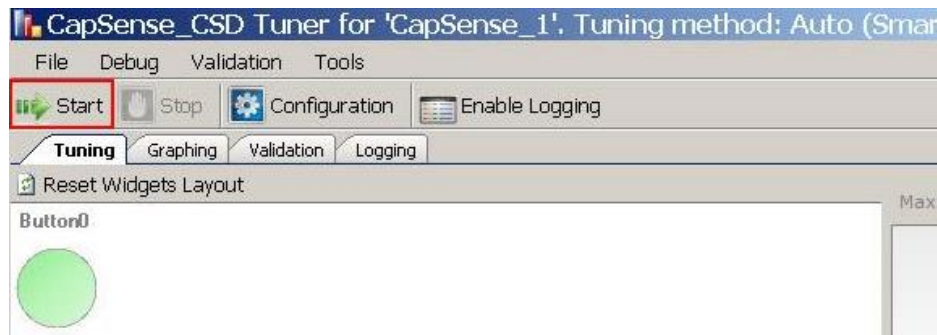


注: MiniProg3 の電圧設定は、基板上の電圧設定と一致する必要があります。

4. 「OK」をクリックすると、「Tuner Communication Setup」ウィンドウが閉じます。

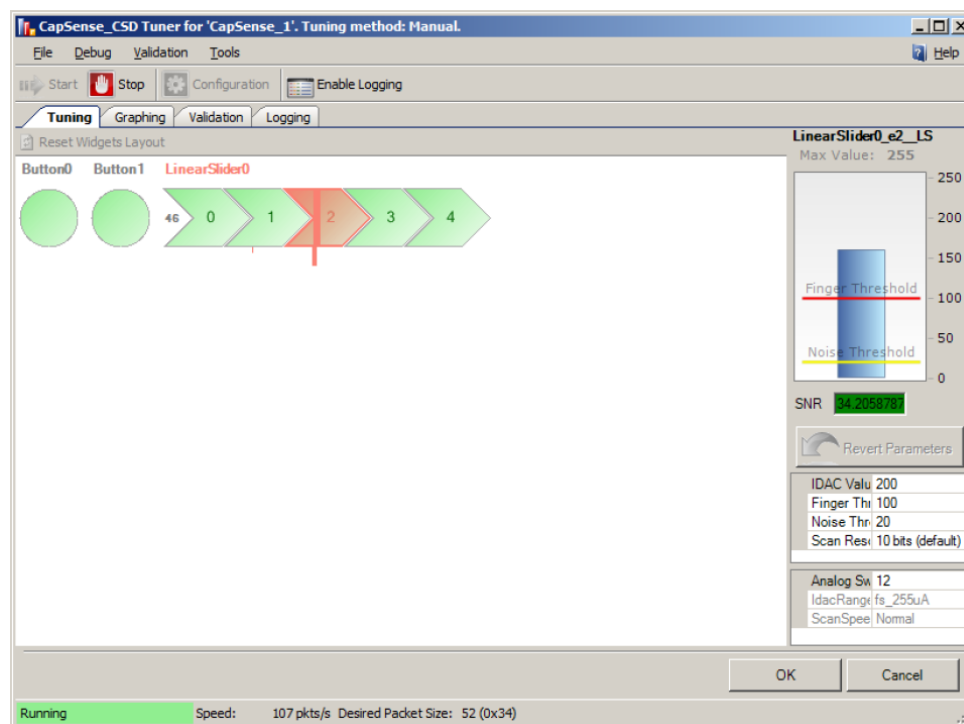
5. チューナー ウィンドウで「Start」をクリックして I²C 通信を開始し、CapSense パラメーターの監視を始めます。

図 5-44. チューナー通信の開始



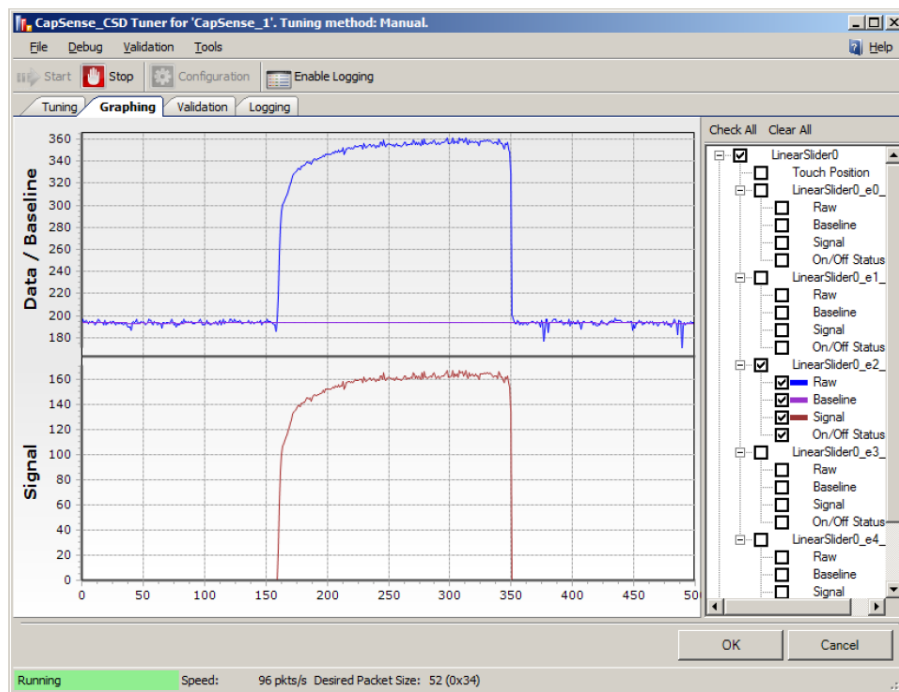
6. 各センサーのパラメーターを設定します。チューナー ウィンドウの右下にある「OK」をクリックした後、チューニングが完了すると、コンポーネントのパラメーターが設定されます。

図 5-45. センサーのパラメーターの設定



7. 「Graphing」タブでは、信号を監視できます。データ ログは「Logging」タブで取得できます。

図 5-46. CapSense 結果の監視



注: 「Graphing」タブに表示された raw カウント信号はフィルタをかけられていない raw カウントです。raw カウントにフィルタをかけた場合、SNR はフィルタをかけられた raw カウントを使って計算する必要があります。「[raw データ ノイズ フィルタ](#)」を参照してください。

6. CapSense 性能のチューニング



チューニングとは、CapSense パラメーターの最適な値を決めることです。チューニングは、タッチに対する高い感度を保ち、センサー基板やオーバーレイの素材、環境条件によるプロセスのばらつきを補正するために必要です。

6.1 基本事項

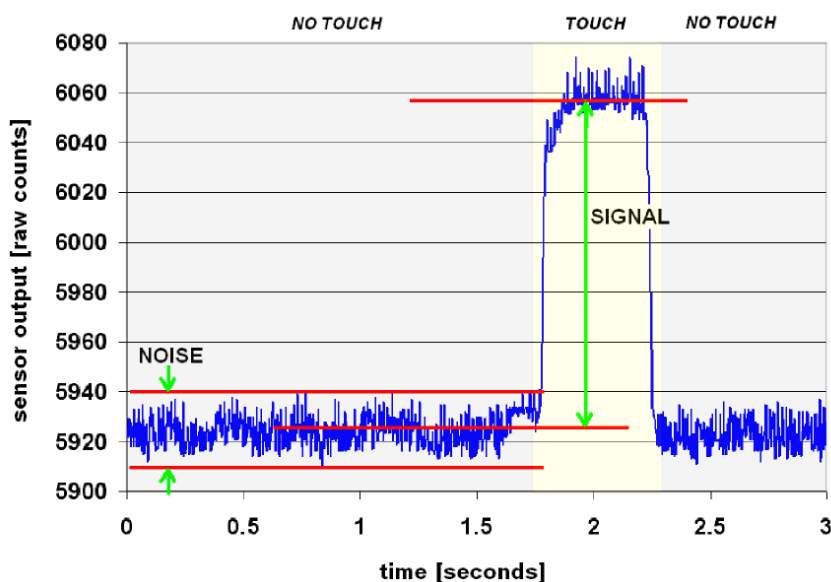
6.1.1 信号、ノイズおよび SNR

適切に調整された CapSense システムでは、オンとオフのセンサー状態が確実に識別されます。この性能レベルを得るには、CapSense 信号は CapSense ノイズよりはるかに大きくする必要があります。信号とノイズを比較する量が、信号対雑音比 (SNR) です。CapSense の SNR の意味を説明する前に、まずタッチ センシングにおいて信号とノイズとは何かを定義します。

6.1.1.1 CapSense 信号

CapSense 信号とは図 6-1 に示すように、センサー上に指を置いたときのセンサー応答の変化です。センサーの出力は、センサー静電容量を追跡する値を表示するデジタル カウンターです。この例では、センサーに指を置かない場合の平均レベルは 5925 カウントです。センサーに指を置いた場合の平均出力は 6060 カウントに増えます。CapSense 信号は指接触によるカウントの変化 (信号=6060-5925=135 カウント) を追跡します。

図 6-1. CapSense の信号とノイズの例



6.1.1.2 CapSense ノイズ

CapSense ノイズは、図 6-1 に示すように、センサー上に指を置かないときのセンサー応答のピークツーピーク変化です。この例では、指を置かない場合の出力波形は、5912～5938 の範囲内に制限されています。ノイズはこの波形の最大値と最小値の差 (ノイズ=5938-5912=26 カウント) です。

注: システムで **raw データ ノイズ フィルタ** を使用した場合、フィルタされた raw カウントを使ってノイズを計算してください。**チューナー GUI** は「Graphing」タブでフィルタをかけていない raw カウントを通知するため、ノイズを計算するためにデータを記録する必要があります。

6.1.1.3 CapSense SNR

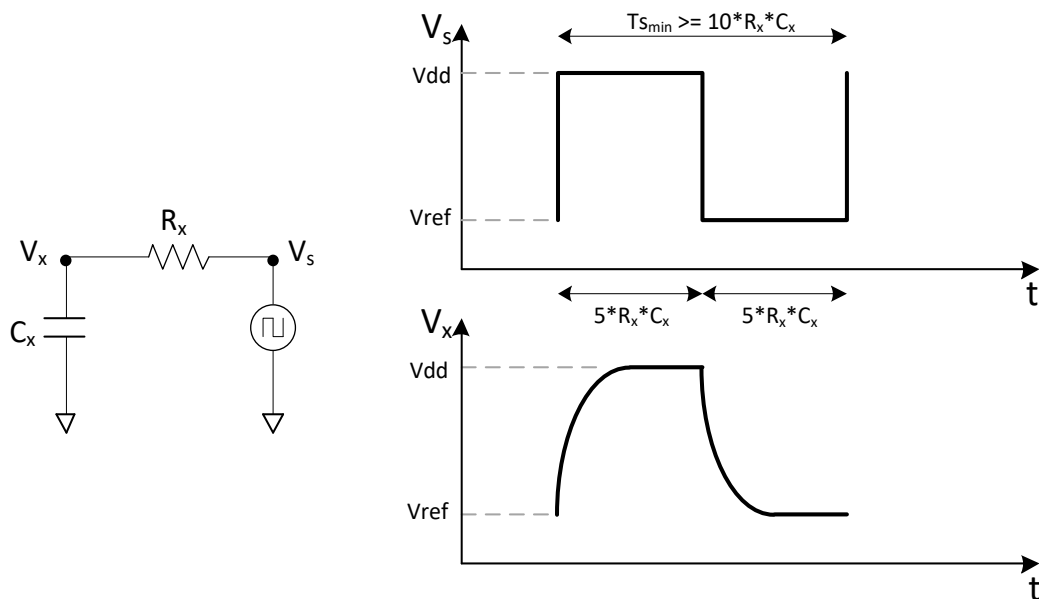
CapSense SNR とは信号とノイズの単純比です。この例では、信号が 135 カウントであり、ノイズが 26 カウントである場合、SNR は (135:26) で、簡単にすると (5.2:1) です。CapSense の推奨最小 SNR は 5:1 で、すなわち信号値がノイズ値より 5 倍大きいです。一般的に、フィルタはノイズを減少させるためにファームウェアに実装されます。詳細は「**ソフトウェア フィルタリング**」を参照してください。

6.1.2 充電／放電速度

チューニング プロセスにおいて最高感度を達成するには、各サイクルの間センサー コンデンサを完全に充電および放電しなければなりません。充電／放電パスは、式 5 で定義するスイッチング クロックに等しいレートで 2 つの状態間で切り替わります。

充電／放電パスには、電荷移動を減速させる直列抵抗が含まれます。電荷移動の変化速度は図 6-2 に示すように、センサー コンデンサと直列抵抗から計算する RC 時定数で決まります。

図 6-2. 充電／放電波形



充電／放電速度を、RC 時定数に適合するレベルに設定します。経験則として、各遷移には 5RC の期間を合わせて、毎期間 2 回の遷移 (充電 1 回、放電 1 回) とする必要があります。次の式に最小期間と最大周波数を示します。

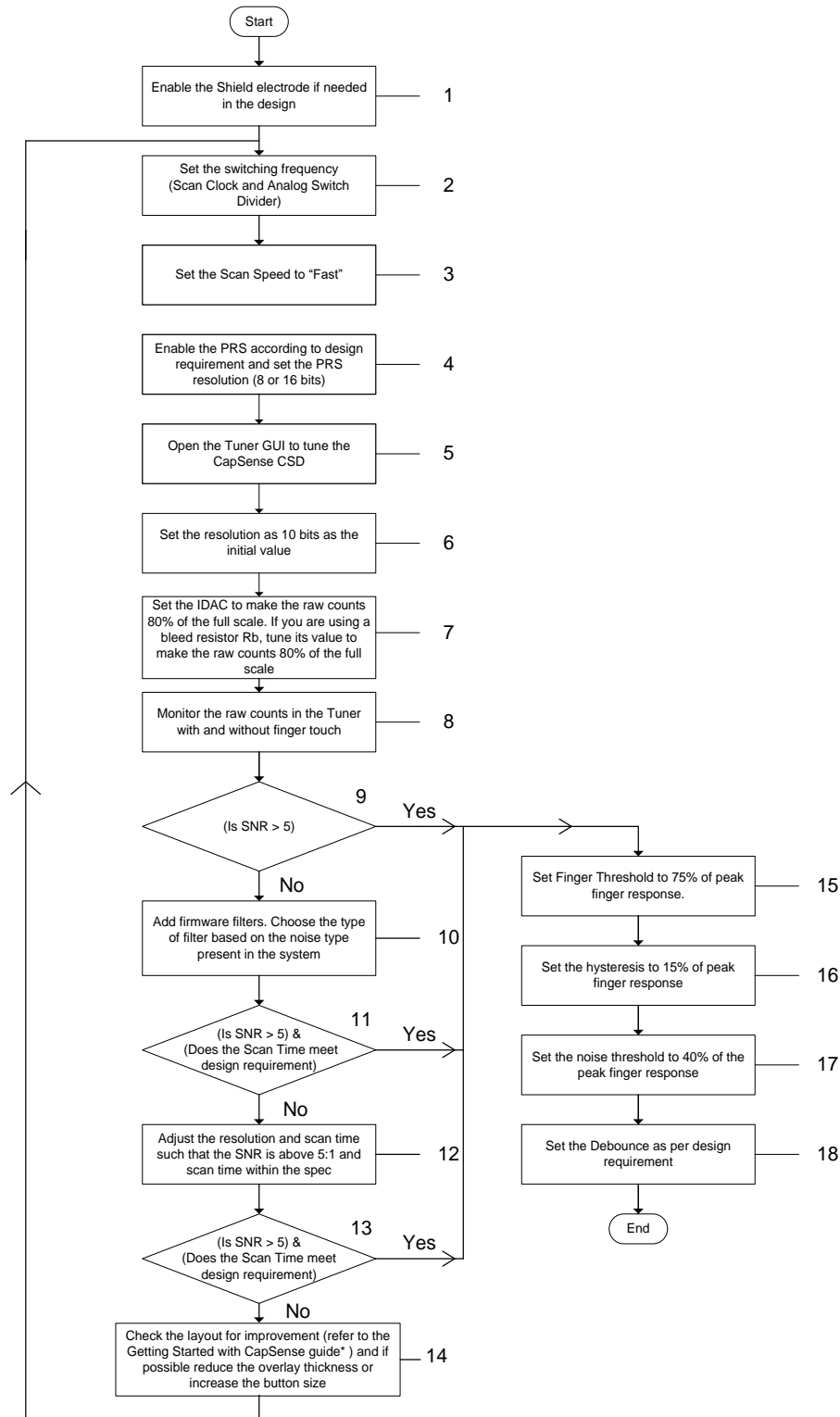
$$T_{SMin} = 10 \times R_X C_X \quad \text{式 7}$$

$$f_{SMax} = \frac{1}{10 \times R_X C_X} \quad \text{式 8}$$

6.2 手動チューニング

図 6-3 に示すフロー チャートは、CapSense_CSD コンポーネントの手動チューニングの手順を提供します。

図 6-3. CapSense_CSD コンポーネントのチューニング



1. シールド電極を使用する場合に関する情報は「[耐水性デザイン](#)」を参照してください。
2. センサーが完全に充電および放電されるために、各静電容量センサーのスイッチング周波数は「[アナログ スイッチ分周器](#)」で説明したように設定する必要があります。あるいは、CapSense_CSD コンフィギュレーション ウィンドウでアナログ スイッチ分周器の値を増やします。この値を変更すると、プロジェクトをリビルドし、デバイスをリプログラムする必要があります。
3. SNR またはスキャン時間が不適切でない場合、後でスキャン速度を変更できます。
4. この機能をいつ、どのように使用するかについては「[疑似乱数シーケンス \(PRS\)](#)」を参照してください。
5. GUI を開くための手順は「[チューナーGUI](#)」を参照してください。
6. デザインのオーバーレイが 1mm 未満である場合、より低い分解能 (8 か 9) を初期値として使用することができます。
7. raw カウントがフルスケール値の 80%に達するようにチューナーGUI の [IDAC 値](#) を調整します。チューナーGUI でフルスケール値の 80%を達成できない場合は、CapSense_CSD コンポーネントのコンフィギュレーション ウィンドウで IDAC 範囲を調整します。この値を変更すると、プロジェクトをリビルドし、デバイスをリプログラムする必要があります。

IDAC シンク/IDAC ソース方式の代わりにブリード抵抗 Rb を使用した場合、raw カウントがフルスケール値の 80%に達するように抵抗値を変更します。
8. ピークツーピークのノイズおよび指のピーク応答を記録します。
9. SNR は単に信号とノイズの比率です。適切な CapSense 設計では、SNR は 5:1 以上であることが必要です。この要件を満たした場合、ステップ 15 に移ってください。
10. システムにあるノイズのフィルタを正しく選択するには「[フィルタ セクション](#)」を参照してください。「First Order IIR 1/4」フル タ は、必要な SRAM を最小限にし、速い応答を提供するため、良い最初の選択となります。
11. SNR が 5:1 以上であることを確認します。そうである場合、デザインの要件を満たすスキャン時間を確認します。スキャン時間を確認するためには、
 - チューナーGUI で OK をクリックして、CapSense_CSD コンポーネント パラメーターを更新します。
 - 「Scan Order」タブを選択して、スキャン時間を確認します。
 SNR とスキャン速度両方の要件が満たされた場合、ステップ 15 に移ってください。
12. SNR を向上するために分解能を増加し、スキャン速度を下げることができます。ただし、両方の変更はスキャン時間を増加させます。
13. ステップ 11 を参照してください。
14. SNR がまだ 5:1 に達していない場合、プリント基板のレイアウトやオーバーレイ設計を改善することを検討してください。オーバーレイ厚さを薄くし、ボタン直径を大きくすると、感度が高くなります。プリント基板の設計ガイドラインは「[Getting Started with CapSense](#)」の第 3 章を参照してください。
15. SNR を確認するとき、測定した指のピーク応答の 75%を使用します。
16. SNR を確認するとき、測定した指のピーク応答の 15%を使用します。
17. SNR を確認するとき、測定した指のピーク応答の 40%を使用します。
18. 「[デバウンス](#)」を参照してください。一般的に、デバウンス値を 2 に設定する必要があります。スキャン速度の高いデザインでは、デバウンス値をより高く (5 などに) 設定してください。

注: チューニング方法が「Auto」に設定された場合でも、パラメーターを手動で調整することができます。チューナーGUI でパラメーターを変更して OK をクリックすると、手動で入力したパラメーターがコンポーネントに適用されます。

6.3 SmartSense 自動チューニング

チューニング方法に「Auto」を選択すると、SmartSense アルゴリズムは CapSense_CSD コンポーネント パラメーターを選択します。性能を最大限にしなが、CapSense の安定した動作を確保するために、SmartSense は各センサーの SNR を (5:1~11:1) の範囲に維持します。「[パラメーター設定](#)」で説明した 4 つのみの低レベル パラメーターを設定する必要があります。SmartSense は IDAC シンク/IDAC ソース方式にのみ対応しています。ブリード抵抗方式には対応していません。

6.3.1 ガイドライン

SmartSense 自動チューニングを使用するために、以下のことが必要です。

- すべてのセンサー C_P は 5~45pF です。
- C_F は少なくとも 0.1pF です。
- C_{MOD} コンデンサは X7R タイプ 2.2nF、定格電圧が 5V 以上です。
- SmartSense は手動チューニングと同じ API をサポートしています。ただし、デザイン性能への影響が確かに分からない限り、ファームウェアで CapSense_CSD コンポーネント ファームウェアを変更する API を使用しないでください。

6.3.2 パラメーター設定

6.3.2.1 センサー自動リセット

このパラメーターでは、ベースラインを常に更新されるか、または信号差がノイズ閾値以下になったときにのみ更新するかを決めます。「Enabled」に設定した場合、ベースラインは常に更新されます。この設定では、センサーがオンとなる最大時間を制限します（一般的に 5~10 秒）が、システムの故障のためタッチがなくても raw カウントが急に上昇した場合、センサーが無限にオンのままになることを防ぎます。「[センサー自動リセット](#)」を参照してください。

6.3.2.2 デバウンス

デバウンス パラメーターでは、デバウンス カウンターをセンサーの有効状態への遷移に追加します。センサーが無効状態から有効状態になるには、指タッチ信号がデバウンス数の連続したスキャンで検知される必要があります。このパラメーターは、同じ程度ですべてのセンサーへ影響を及ぼします。「[デバウンス](#)」を参照してください。

6.3.2.3 変調コンデンサ ピン

このパラメーターでは、 C_{MOD} コンデンサが接続するピンを選択します。P0[1]と P0[3]ピンは使用可能です。

注: SmartSense が正常に動作するには、2.2nF の外付けコンデンサが必須です。

6.3.2.4 感度レベル

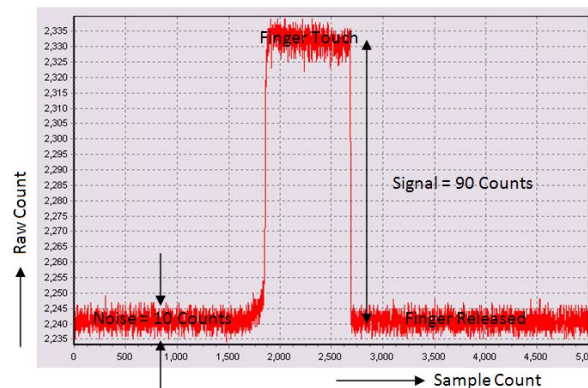
センサー信号の強さは感度により増減されます。高 (0.1pF)、中高 (0.2pF)、中低 (0.3pF)、低 (0.4pF) の感度が選択可能です。

より厚いオーバーレイの場合、正常に実装するには、より強い信号が必要です。センサーからより強い信号を生成するために、SmartSense のセンサーをスキャンする時間が増えます。つまり、高感度のセンサーは中高感度のセンサーよりスキャン時間が長くなります。

このパラメーターを設定する際、可能な限り低い感度値を使用してください。堅牢な性能を保証するために以下を実施します。

1. [チューナーGUI](#)を開きます。
2. 感度レベルを低 (0.4pF) に設定して、SNR を計算します。[図 6-4](#) に、指タッチがある場合の一般的な raw カウント グラフを示します。SNR が 5:1~10:1 の範囲内にあるように感度レベルを調整します。

図 6-4. 標準センサーへの指タッチがあるときの raw カウント グラフ



7. 設計上の注意事項



アプリケーションに静電容量タッチ センシング技術を組み込む場合、CapSense デバイスはより大きなフレームワークを必要とすることに注意してください。プリント基板レイアウトからエンドアプリ環境へのユーザー インターフェースにいたるまで、すべての設計レベルに慎重に設計すると、堅牢で信頼性の高いシステム性能の実現が可能になります。「[Getting Started with CapSense](#)」には、オーバーレイ選択や ESD 保護、電磁環境適合性 (EMC) などの詳しい情報があります。本節では、PSoC 3 および PSoC 5LP 固有の設計注意事項について説明します。

7.1 ソフトウェア フィルタリング

CapSense_CSD コンポーネントには、CapSense のスキャン結果からノイズを除去するために、いくつかの既成ファームウェア フィルタが用意されています。下表では、フィルタおよびその使用する場合を説明します。

表 7-1. コンポーネント用に用意されている CapSense フィルタの表

フィルター	説明	用途
メジアン	サイズ 3 のバッファからメジアン入力値を計算する非線形フィルタ	モーターやスイッチング電源からのスパイク ノイズ
平均	等しく加重された係数を持つ、有限インパルス応答フィルタ (フィードバックなし)	電源からの周期的ノイズ
IIR	RC ローパス フィルタに類似したステップ応答を備えた有限インパルス応答フィルタ (フィードバック付き)	高周波数ノイズ
ジッタ	前の入力に基づいて現行の入力を制限する非線形フィルタ	厚いオーバーレイからのノイズ (SNR<5:1)、特にスライダの重心データに有用

注意:

- コンポーネントのメジアン フィルタはサイズ 2 のバッファを使用します。このバッファには前のサンプルが 2 つ含まれています。バッファに格納されている現時点のサンプルと前の 2 つのサンプルは整えられ、中間値がメジアン値として取得されます。
- コンポーネントの平均フィルタはサイズ 2 のバッファを使用します。このバッファには前のサンプルが 2 つ含まれています。フィルタ出力は、2 つの前のサンプルと現時点のサンプルの平均値です。

図 7-1. raw カウントのフィルタリング
 黒 = フィルタをかけられていない raw カウント
 緑 = フィルタをかけられた raw カウント

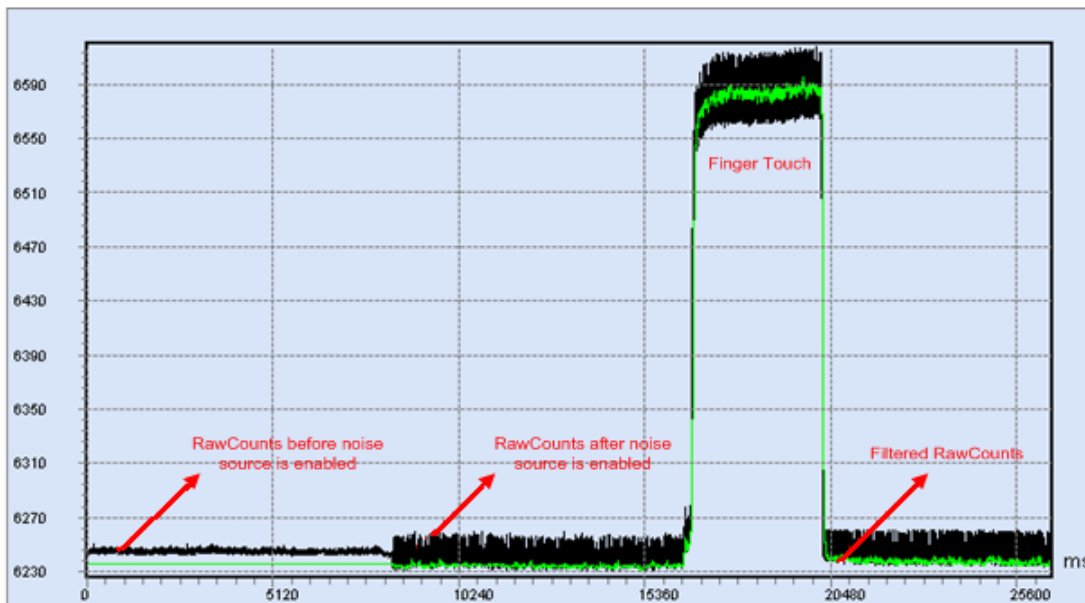


図 7-1 に、ノイズとフィルタリングの raw カウントへの影響を示します。センサーの近くの高スイッチング ラインによるノイズをシミュレートするために 500kHz で近隣のセンサーを切り替えることでノイズを作りました。この例は以下のとおりです。

- ノイズ源のないノイズ=N=8 カウント (ピークツーピーク)
- ノイズ源のあるノイズ=N1=40 カウント (ピークツーピーク)
- フィルタをかけた後のノイズ=N2=12 カウント
- 信号 (指のピーク応答) =S=320 カウント
- フィルタを適用する前の SNR=SNR1=S/N1=320/40=8
- フィルタを適用した後の SNR=SNR2=S/N2=320/12=26.7

7.2 消費電力

バッテリー駆動アプリケーションでは、電流レベルを下げると、バッテリー寿命が長くなります。この点を考慮して、システム設計者はシステムの平均消費電流を最小限にする必要があります。

表 7-2. PSoC3 のセンサー1 個のスキャン用の消費電力

CPU (MHz)	mA (V _{DD} =3.3V)	mA (V _{DD} =5V)
3	3.7	4.7
6	4.4	5.5
12	5.9	6.9
24	8.4	9.5
48	14.5	16.3

CapSense 静電容量タッチセンシング システムの消費電力を削減する方法はいくつかあります。

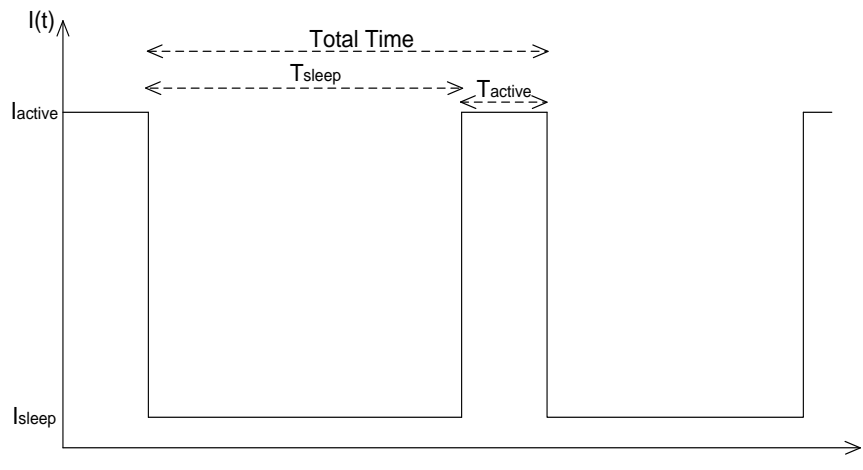
- **GPIO ドライブ モードを HI-Z に設定:** 初期設定では、すべての GPIO ピンは HI-Z に設定されています。ただし、プロジェクトがいくつかのピンのドライブ モード コンフィギュレーションを変更した場合は、それらを使用しないときは HI-Z にする必要があります。
- **低消費電力用に CPU 速度を最適化:** CapSense 動作では、専用のハードウェアがセンサー スキャンを実施するため、CPU は高速である必要はありません。ただし、CPU はスキャンを開始する前にハードウェアを設定し、ハードウェアがスキャンを完了すると、CPU は結果を処理します。そのため、CPU の速度を遅くすると、スキャン時間が増加します。総スキャン時間を前スキャン、ハードウェア スキャン、後スキャンに分割する方法は「[ノンブロッキング アーキテクチャ](#)」を参照してください。
- **低 V_{DD} で動作**

これらの提案に加えて、スリープ スキャン方式の適用も効果的です。

7.2.1 スリープ スキャン方式

一般的なアプリケーションでは、CapSense コントローラーは常にアクティブな状態である必要はありません。デバイスをスリープ状態にして、デバイスの CPU と主要ブロックを停止することができます。スリープ状態のデバイスが消費する電流はアクティブ時の電流よりもはるかに小さいです。

図 7-2. スリープ スキャン方式



デバイスの長期的な平均消費電流は、以下の式を使って計算できます。

$$I_{average} = \frac{(I_{active} \times T_{active}) + (I_{sleep} \times T_{sleep})}{\text{合計時間}} \quad \text{式 9}$$

ここで、

$I_{average}$ = デバイスの平均電流

I_{active} = デバイスのアクティブ電流

T_{active} = デバイスのアクティブ時間

I_{sleep} = デバイスのスリープ電流

T_{sleep} = デバイスのスリープ時間

合計時間 = $T_{active} + T_{sleep}$

デバイスの平均消費電力は、次のように計算します。

$$P_{average} = V_{DD} \times I_{average} \quad \text{式 10}$$

ここで、

$P_{average}$ = デバイスの平均消費電力

V_{DD} =デバイスの電源電圧

$I_{average}$ =デバイスの平均電流

7.2.2 平均消費電力の測定

7.2.2.1 アクティブ電流

アクティブ電流を測定するために、デバイスをプログラムしてすべての電源ピン (V_{DDD} 、 V_{DDA} 、 V_{DDIO}) の電流を測定します。このために、電源ピンを短絡させて、共通電源ポイントで電流を測定することは最適な方法です。

7.2.2.2 アクティブ時間

スキャン ループの前と後に GPIO ピンをトグルすることでアクティブ時間を測定することができます。次のコードの一部は、ピンがスキャンの開始前に設定され、スキャンの完了後にリセットされた一般的な CapSense スキャン ループを示します。

```

22     for (;;)
23     {
24         if (CapSense_IsBusy() ==0 )
25         {
26             Pin_Write(0);
27             CapSense_UpdateEnabledBaselines();
28             if (CapSense_CheckIsWidgetActive())
29             {
30                 /* Do required */
31             }
32
33             /*Check other CapSense sensors active and do the required */
34
35             Pin_Write(1);
36             CapSense_ScanEnabledWidgets();
37         }
38         /* Other application code than CapSense */
39     }
40
41
42
43 }
44

```

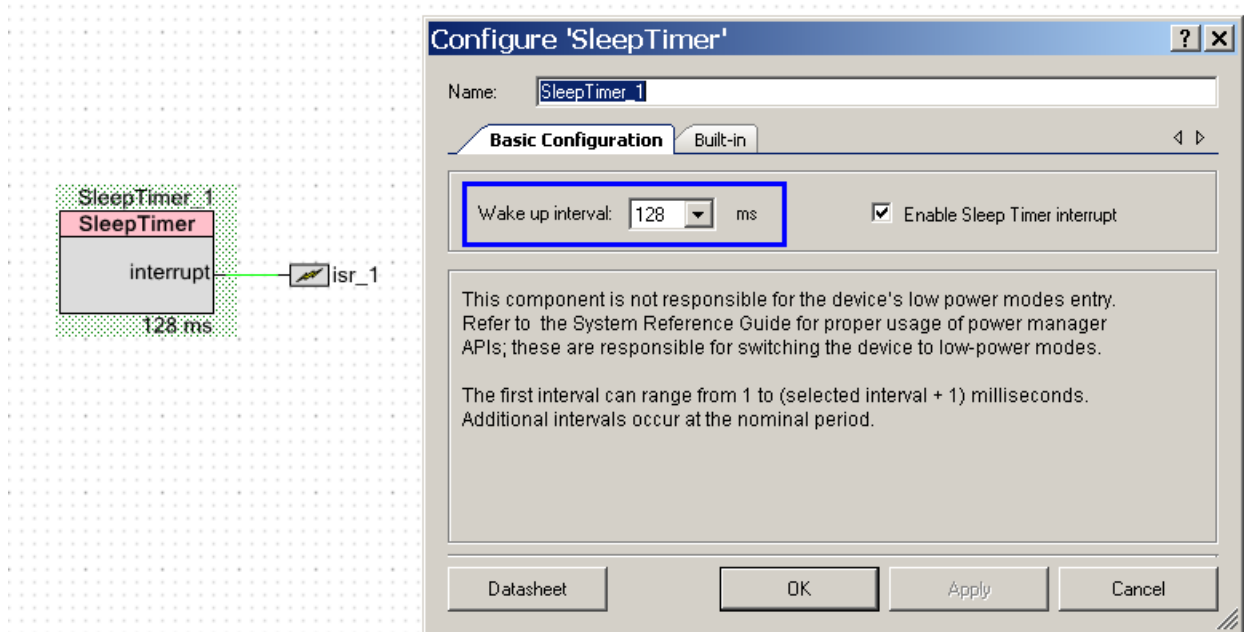
7.2.2.3 スリープ電流

PSoC 3 のスリープ電流は 1 μ A、PSoC 5LP のスリープ電流は 2 μ A です。スリープ電流を測定するために、デバイスを永久的にスリープ モードに維持するプロジェクトを作成し、「[アクティブ電流](#)」で説明した方法を使用します。

7.2.2.4 スリープ時間

API「CyPmSleep()」ではデバイスをスリープモードに移行させます。図 7-3 に示すように、スリープタイマーは特定の間隔後に割り込みを生成してデバイスをウェイクアップします。

図 7-3. スリープタイマー

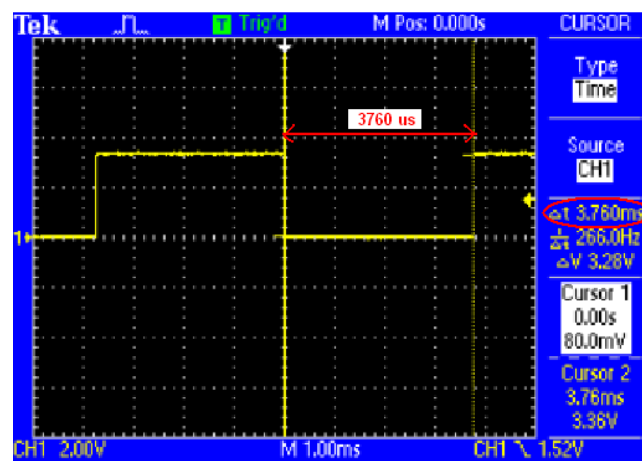


7.2.2.5 平均消費電力の計算

アクティブ電流 (表 7-1 より、CPU=24MHz、動作電圧=3V)

$I_{\text{active}}=8.4\text{mA}$

図 7-4. アクティブ時間の測定



アクティブ時間 (PSoC 3、CPU=24MHz、分解能=12ビット、スキャン速度=通常)

$T_{\text{active}}=3760\mu\text{s}$

スリープ電流 (データシートより)

$$I_{\text{sleep}} = 1\mu\text{A}$$

スリープ時間 (スリープ期間がスリープ タイマーで 128ms に設定されたことを想定)

スリープ時間=合計時間-アクティブ時間

合計時間=スキャン間隔=128ms

$$T_{\text{sleep}} = 128,000\mu\text{s} - 3760\mu\text{s} = 124240\mu\text{s}$$

平均電流 (式 9 より)

$$I_{\text{average}} = 247.7\mu\text{A}$$

平均消費電力

$$P_{\text{average}} = V_{\text{DD}} \times I_{\text{average}}$$

$$P_{\text{average}} = 3.3\text{V} \times 247.7\mu\text{A} = 817.4\mu\text{W}$$

スリープ スキャン モードを使用する平均消費電力は 817.4μW になります。一方、スリープ モードを使用しないシステムの平均電力は 27.7mW になります。

7.3 応答時間

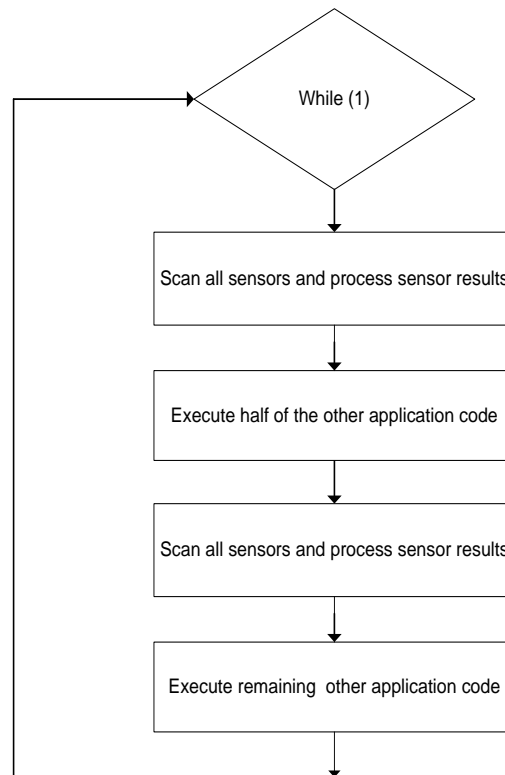
7.3.1 スリープ スキャン モード

スキャン間にデバイスをスリープ モードに移行させることでかなりの電力を節約することができます。ただし、スリープ時間を非常に高い値に増加すると、応答時間が遅くなります。デザインで消費電力と応答時間の両方が重要である場合、連続スキャンとスリープスキャン両方のモードを組み合わせた方法を使用できます。この方法を使うと、デバイスは大部分の時間にセンサーをスキャンしてからスリープ状態になるスリープ スキャン モードにあります。センサーに触れると、デバイスはセンサーが常にスキャンされる連続スキャン モードに遷移します。デバイスは特定の期間の間、連続スキャン モードのままになります。この期間中にセンサーに触れない場合、デバイスはスリープ スキャン モードに戻ります。

7.3.2 長いバックグラウンド ループ

バックグラウンド ループが CapSense 以外のアプリケーションに使用する実行コードが長すぎると、応答時間が遅くなることもあります。図 7-5 に示すように、ループごとに CapSense スキャンを複数回実行できます。

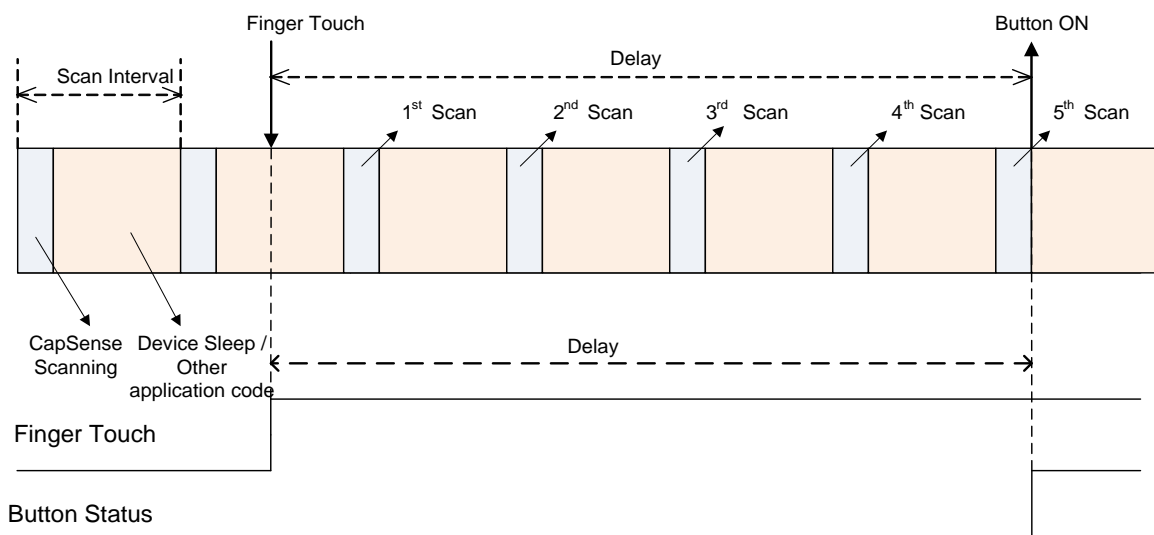
図 7-5. 応答時間を改善するループ内の複数回のスキャン



7.3.3 デバウンス

デバウンスは、システムが誤った指タッチを検知することを防ぎますが、図 7-6 に示すように指検知を遅延させます。

図 7-6. スキャン間隔および検知遅延、デバウンス カウント=5



デバイスに起因する最大遅延は以下のとおりです。

$$\text{最大遅延} = \text{デバウンス} \times \text{スキャン間隔}$$

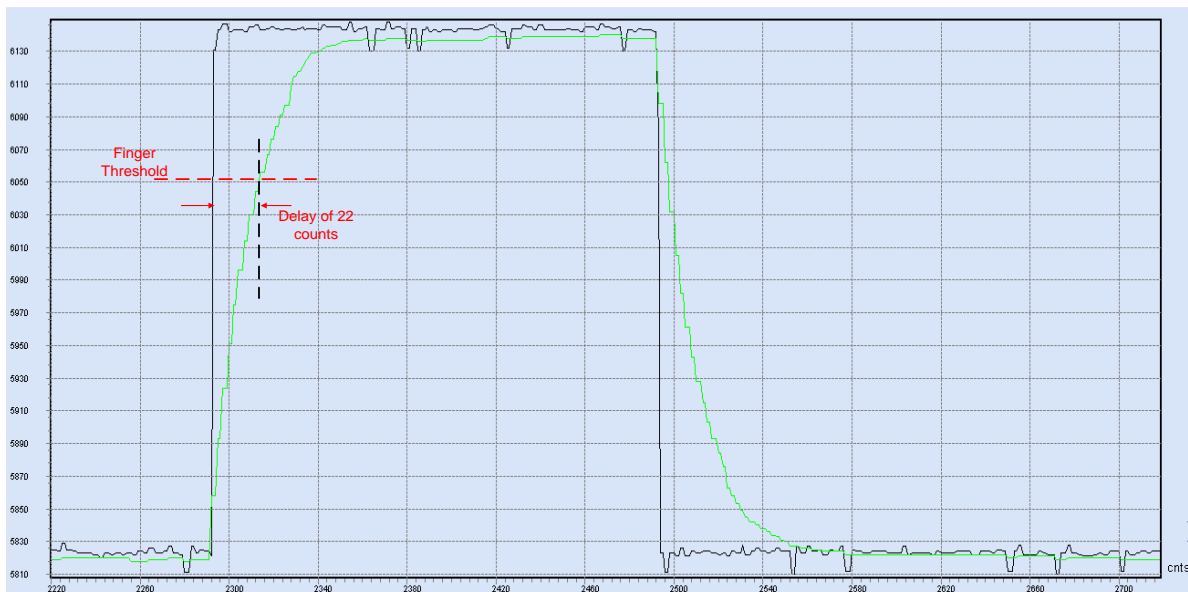
スリープ モードを使用する場合、デバウンスを 1 に設定してください。スリープ モードは応答時間を長くするため、効果的に信号をデバウンスします。これが適切な方法でない場合は以下のコードの一部に示すように指タッチを検知するとセンサーを複数回スキャンすることができます。

```
if (CapSense_CheckIsWidgetActive(CapSense_BUTTON0__BTN))
{
    /* Scan the sensor multiple times */
    for (i=0; i < DEBOUNCE ; i++)
    {
        if ( (CapSense_IsBusy() == 0 ) && CapSense_CheckIsWidgetActive(CapSense_BUTTON0__BTN) )
        {
            CapSense_ScanEnabledWidgets();
        }
    }
    /* After number of scans equal to debounce count is complete, check for sensor active.
    * If it is still active, then sensor can be considered as ON.
    */
    if (CapSense_CheckIsWidgetActive(CapSense_BUTTON0__BTN))
    {
        /* Process the code which is based on sensor active */
    }
}
```

7.3.4 フィルタによる遅延

フィルタをかけることも指タッチの検知を遅延させます。IIR16 フィルタの使用の影響は次のとおりです。IIR16 フィルタは新しいデータの 1/16 および古いデータの 15/16 を追加して、新しいフィルタ サンプルを計算することに注意してください。

図 7-7. 指タッチでのフィルタによる遅延



指によるステップ変更=320 カウント

指閾値=指のピーク応答の 75%=240 カウント

フィルタにより発生する遅延の計算は以下のとおりです。

$$T_n = a \times r^{n-1}$$

ここで、

a=1 (ステップ変更または指のピーク応答)

Tn=0.25 (ステップ変更の 75%は閾値)

r=1/16 (フィルタ IIR16)

n=22

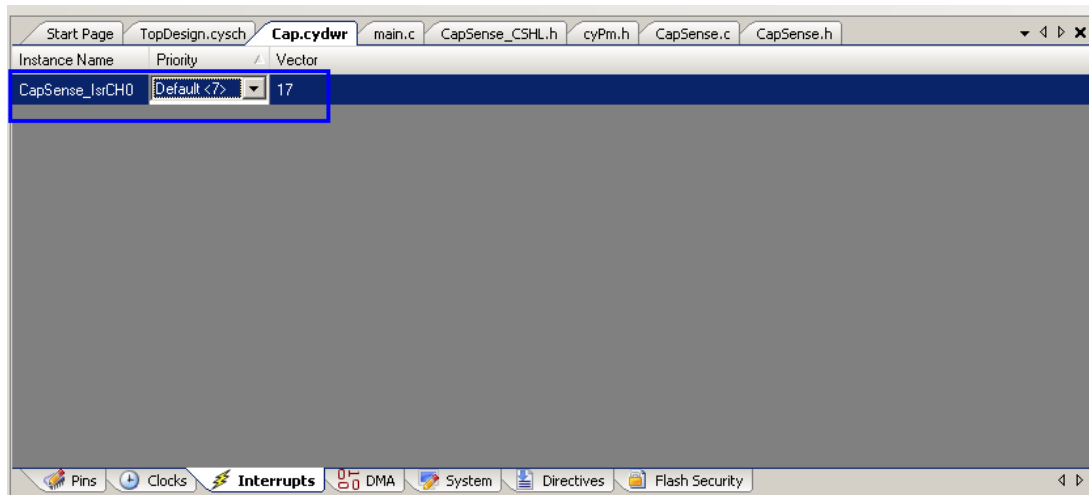
つまり、指タッチは 22 回のスキャンの後にのみ検知されます。フィルタによる遅延を最小限にするために、適切なフィルタを選択してください。例えば、IIR4 フィルタは 7 回スキャンのみの遅延を発生させます。

IIR16 フィルタを使用する必要がある場合、「デバウンス」で説明したように複数回のスキャンを使用してください。ただし、センサーはオン/オフ状態にかかわらずループごとに複数回スキャンする必要があります。

7.3.5 割り込み優先順位

「[ノンブロッキング アーキテクチャ](#)」で説明したように、CapSense_CSD コンポーネントのコード アーキテクチャは元々ノンブロッキングです。つまり、ハードウェアが CapSense をスキャンしている間、CPU はその他のアプリケーション コードを実行することができます。ハードウェアがスキャンを完了すると、CPU への割り込みが発生します。CapSense 割り込みの優先順位を変更するために、<プロジェクト名>.cydwr ファイルを開いて「Interrupts」タブをクリックします。

図 7-8. 割り込み優先順位の変更



通信プロトコルなど優先順位がより高い割り込みがある場合、CapSense 割り込みの優先順位を低くすることができます。優先順位の高い割り込みの処理中に CapSense 割り込みがアサートされる場合、CapSense は割り込みは遅れて処理されます。この遅延によりデータ破損が起こりませんが、次のセンサーのスキャンを遅らせ、総スキャン時間を増加させます。

7.4 ピン割り当て

7.4.1 オペアンプ出力ピン

オペアンプ出力は P0[0]、P0[1]、P3[6]、P3[7]ピンに直接接続されます。これらのピンを CapSense に使用した場合、オペアンプは使用できなくなります。これらのピンはオペアンプ出力に直接接続しているため、 C_P が高くなります。そのため、これら 4 ピンを CapSense に使用しないでください。使用する必要がある場合は、スライダやタッチ パッドに使用せず、ボタンに使用してください。また、 C_P を低くするためにピンの配線を短くする必要があります。

7.4.2.2 チャンネル デザインのピン割り当て

2 チャンネル デザインでは、特定のチャンネルに属するピンがチップと同じ側にあるようにします。2 チャンネル デザインの詳細は「[チャンネル数](#)」を参照してください。

7.4.3 C_{MOD} ピン割り当て

最も効果的なアナログ ルーティングを実現するには、以下のピンを C_{MOD} に使用してください。

- 左側: P2[0]、P2[4]、P6[0]、P6[4]、P15[4]
- 右側: P1[0]、P1[4]、P5[0]、P5[4]

7.5 プリント基板レイアウトのガイドライン

詳細なプリント基板のレイアウト ガイドラインは「[Getting Started with CapSense](#)」を参照してください。

8. リソース



8.1 ウェブサイト

PSoC 3 および PSoC 5LP における CapSense アプリケーションを示す [CapSense® PLUS](#) をご覧ください。

8.2 データシート

PSoC 3 および PSoC 5LP デバイス ファミリのデータシートは www.cypress.com から入手できます。

- [PSoC 3 データシート](#)
- [PSoC 5LP データシート](#)

8.3 テクニカル リファレンス マニュアル

次のテクニカル リファレンス マニュアルでは、トップレベル アーキテクチャ図、レジスタのまとめ、タイミング図など PSoC 3 および PSoC 5LP アーキテクチャに関する情報にすばやく、簡単にアクセスできます。

- [PSoC 3 TRM](#)
- [PSoC 5LP TRM](#)

8.4 開発キット

8.4.1 PSoC 3 および PSoC 5LP 開発キット

- [CY8CKIT-001 PSoC®開発キット](#)
- [CY8CKIT-030 PSoC® 3 開発キット](#)
- [CY8CKIT-050 PSoC® 5LP 開発キット](#)

8.4.2 モジュール ボードを開発キットに接続するインターフェース ボード

- [CY8CKIT-031 PSoC CapSense 拡張基板キット](#)

8.4.3 ユニバーサル CapSense モジュール基板

8.4.3.1 シンプル ボタン モジュール基板

[CY3280-BSM](#) シンプル ボタン モジュールは、10 個の CapSense ボタンと 10 個の LED から成ります。このモジュールはあらゆる CY3280 ユニバーサル CapSense コントローラー基板と接続します。

8.4.3.2 マトリックス ボタン モジュール基板

CY3280-BMM マトリックス ボタン モジュールは、4x4 マトリックス形式として構成される 8 個の LED と 8 個の CapSense センサーから成ります(すなわち、物理的ボタン 16 個が形成されます)。このモジュールはあらゆる CY3280 ユニバーサル CapSense コントローラー基板と接続します。

8.4.3.3 リニア スライダー モジュール基板

CY3280-SLM リニア スライダー モジュールは 5 個の CapSense ボタン、1 個のリニア スライダー (10 個のセンサー付) および 5 個の LED から成ります。このモジュールはあらゆる CY3280 ユニバーサル CapSense コントローラー基板と接続します。

8.4.3.4 ラジアル スライダー モジュール基板

CY3280-SRM ラジアル スライダー モジュールは 4 個の CapSense ボタン、1 個のラジアル スライダー (10 個のセンサー付) および 4 個の LED から成ります。このモジュールはあらゆる CY3280 ユニバーサル CapSense コントローラー基板と接続します。

8.4.3.5 ユニバーサル CapSense プロトタイプ モジュール

CY3280-BBM ユニバーサル CapSense プロトタイプ モジュールで、付属コントローラー基板上の 44 ピン コネクタに接続されたすべての信号へのアクセスが可能です。プロトタイプ モジュール基板はユニバーサル CapSense コントローラー基板と併用して、その他の専用のユニバーサル CapSense モジュール基板に備えていない追加機能を実行することができます。

8.4.4 MiniProg3

MiniProg3 は、PSoC 3 および PSoC 5LP 製品をプログラムおよびデバッグするために使用されます。これはまた、チューナー GUI の I²C-USB 通信ブリッジとしても使用されます。

- **CY8CKIT-002 PSoC® MiniProg3 プログラムおよびデバッグ キット:** <http://www.cypress.com/?rID=39045>

8.5 PSoC Programmer

PSoC Programmer は PSoC デバイスをプログラムするための柔軟性のある統合プログラム アプリケーションです。PSoC Designer と PSoC Creator を併用して PSoC デバイスに様々なデザインをプログラムすることが可能です。

PSoC Programmer は API を備えたハードウェア層を提供し、プログラマとブリッジ デバイスを使用して特定のアプリケーションを設計します。PSoC Programmer ハードウェア層は、COM ガイド文書だけでなく、C#、C、Perl および Python の言語でサンプルコードとしても説明されています。

8.6 Multi-Chart

I²C 通信を使用するチューナー GUI に加えて、CapSense 結果を監視するために UART を使用できます。**Multi-Chart** は、リアルタイムに CapSense データ表示および記録を行うための UART 通信ベースのシンプルな PC ツールです。このアプリケーションを使えば、48 のセンサーまでのデータの閲覧、チャートの保存や印刷、後で分析するためにデータをスプレッドシートに保存することも可能です。

8.7 PSoC Creator

サイプレスは、第一級の統合開発環境である **PSoC Creator** を提供しています。PSoC Designer により、単一のツールでハードウェアをコンフィギュレーションし、ファームウェアを開発できます。

8.8 サンプル コード

サイプレスは、設計を速やかに完了するために大量のコード実例を提供しています。PSoC Creator を起動し、スタート ページで「find example project」リンクをクリックします。CapSense に精通するには、以下 2 つのサンプル プロジェクトをお勧めします。

- CapSense_CSD_Design
- CapSense_CSD_With Tuner

8.9 デザイン サポート

サイプレスには様々なデザイン サポート チャンネルがあり、CapSense ソリューション適用に成功しています。

- **知識ベース記事**: 製品ファミリ別の技術情報記事を閲覧したり、CapSense についての様々なトピックを検索できます。
- **CapSense アプリケーション ノート**: 本書に記載された情報に基づく広範囲なアプリケーション ノート。
- **ホワイト ペーパー**: 高度な静電容量タッチ インターフェースに関するトピック。
- **サイプレス開発コミュニティ**: サイプレス技術コミュニティに参加し、情報交換できます。
- **ビデオ ライブラリ**: チュートリアル ビデオで素早く学習できます。
- **品質および信頼性**: サイプレスは顧客満足を第一に考えます。当社の品質ウェブサイトでは、信頼性および品質レポートをご覧いただけます。
- **技術サポート**: 世界一流の技術サポートがオンラインで利用可能です。

AMUXBUS

入出力ピンを複数の内部アナログ信号に接続する PSoC 内にあるアナログ マルチプレクサ バスです。

SmartSense™ 自動チューニング

設計フェーズの後で最適な性能のために、センシング パラメーターを自動設定し、システム、製造および環境変化に対し連続的に補正する CapSense アルゴリズムです。

ベースライン

センサーに人の指で触らないときの raw カウントの傾向を推定し、ファームウェア アルゴリズムから得られる値です。ベースラインは raw カウントの突然の変化に敏感性が低く、差分カウントを計算するための基準点を提供します。

ボタンまたはボタン ウィジェット

関連したセンサーを持って、センサーのアクティブ状態または非アクティブ状態 (すなわち、2 つだけの状態) を報告するウィジェットです。例えば、センサー上の指の「タッチ有り」または「タッチ無し」の状態を検出できます。

差分カウント

raw カウントとベースラインの差です。差分が負数であるか、またはノイズ閾値未満である場合、差分カウントは常に 0 に設定されます。

静電容量センサー

静電容量の変化によってタッチまたは近づいている物体に反応する導電体および基板 (プリント回路基板 (PCB) 上の銅ボタンなど) です。

CapSense®

サイプレスのタッチセンシング ユーザー インターフェース ソリューション。業界 2 位に対して、4 倍の販売実績がある業界 No.1 ソリューションです。

CapSense メカニカル ボタン リプレースメント (MBR)

メカニカル ボタンを静電容量ボタンにアップグレードするサイプレスの構成可能なソリューションであり、センサー パラメーターの設定に必要な設計工数を最小限に抑え、ファームウェアの開発も不要とします。これらのデバイスは CY8CMBR3XXX および CY8CMBR2XXX のファミリを含んでいます。

重心または重心位置

スライダー分解能の指定した範囲内のスライダー上の指の位置を示す数です。この数は CapSense 重心計算アルゴリズムにより算出されます。

補正 IDAC

過剰なセンサー C_P を補正するために CSD により使用されるプログラム可能な定電流源です。この IDAC は、変調 IDAC と違って、CSD ブロックでシグマ-デルタ変調器によって制御されません。

CSD

CapSense シグマ デルタ (CSD) は、静電容量センシング アプリケーション用に自己容量を測定するサイプレスの特許取得済み方法です。

CSD モードでは、センシング システムは電極の自己容量を測定し、指の有無を識別するために自己容量の変化が検出されます。

デバウンス

有効なタッチとなるためにタッチがある必要な連続スキャン サンプル数を定義するパラメーターです。このパラメーターは怪しいタッチ信号を排除するために役立ちます。

指のタッチは、差分カウントがスキャン サンプルの連続デバウンス数で (指閾値+ヒステリシス) を超える場合にのみ報告されます。

被駆動シールド

シールド電極がセンサー スwitching 信号と同じ位相および振幅を持つ信号によって駆動され、耐液性を有効にするために CSD によって使用される技術です。

電極

プリント基板、ITO または FPCB 上のパッドまたは層などの導電材料です。電極は CapSense デバイス上のポートピンに接続し、CapSense センサーとして使用されるか、または CapSense 機能に関する特定の信号を駆動するために使用されます。

指閾値

センサーの状態を確定するためにヒステリシスと一緒に使用されるパラメーターです。センサーの状態は、差分カウントが (指閾値+ヒステリシス) を上回る場合はオンとして報告され、差分カウントが (指閾値-ヒステリシス) を下回る場合はオフとして報告されます。

センサー連結

複数のセンサーを連動して、単一のセンサーとしてスキャンする方法です。近接センシング用のセンサーの領域を増やし、電力消費量を減少するために用いられます。

システムが低消費電力モードにある場合の電力を削減するために、センサーは個別にスキャンされずに、これらのすべてを連動して単一のセンサーとしてスキャンされ、時間を短縮させます。ユーザーがいずれかのセンサーに触った場合、システムはアクティブ モードに移行し、全てのセンサーを個別にスキャンしてアクティブになったセンサーを検出します。

PSoC はファームウェアでセンサー連結をサポートしており、複数のセンサーは同時に AMUXBUS に接続してスキャンできます。

ジェスチャー

ジェスチャーはスワイプやピンチズームなどのユーザーの行動です。CapSense は事前に定義されたタッチ パターンに基づいて異なるジェスチャーを識別するジェスチャー検出機能を備えています。CapSense コンポーネントでは、ジェスチャー機能はタッチパッド ウィジェットのみにサポートされます。

ガード センサー

ボタン センサーと同様に PCB 上の全てのセンサーを取り囲み、液体流を検出するために使用される銅配線です。ガード センサーがトリガされると、ファームウェアは誤ったタッチを防ぐために、すべての他のセンサーのスキャンを無効にすることができます。

ハッチ フィル／ハッチ グランド／ハッチド グランド

静電容量センシングの PCB を設計するとき、良好なノイズ耐性のために接地した銅面をセンサーの周りに配置する必要があります。しかし、ベタ グランドはセンサーの期待されない寄生静電容量を増加させます。そのため、グランドは特別なハッチ パターンで充填する必要があります。ハッチ パターンはメッシュのように密接に配置され、交差されるラインがあり、ラインの幅および 2 本のライン間の間隔は充填率を指定します。耐液性の場合、シールド電極 として呼ばれるこのハッチ フィルはグランドの代わりにシールド信号で駆動されます。

ヒステリシス

システム ノイズに起因してセンサー状態の出力がランダムにトグルすることを回避し、センサーの状態を決定するために指閾値と一緒に使用されるパラメーターです。[指閾値](#)を参照してください。

IDAC (電流出力デジタル-アナログ変換器)

CapSense および ADC 動作の PSoC 内のプログラマブルな定電流源です。

耐液性

水滴、液体流や霧が存在する環境でも確実に動作する静電容量センシング システムの能力です。

リニア スライダー

指の物理的な位置 (単一の軸で) を検出するために特定の直線状で配置された複数のセンサーを含むウィジェットです。

低ベースライン リセット

raw カウントが負のノイズ閾値を異常的に下回るスキャン サンプルの最大数を表すパラメーターです。低ベースラインリセットの値を超える場合、ベースラインは現時点の raw カウントにリセットされます。

手動チューニング

CapSense パラメーターの手動設定 (または手動チューニング) プロセスです。

マトリックス ボタン

マトリクス状で配置された 2 つ以上のセンサーを含んで、垂直方向および水平方向に配置されるセンサーの交差部に人の指 (タッチ) の有無を検出するために使用されるウィジェットです。

M を水平軸上のセンサーの数と、N を垂直軸上のセンサーの数とすれば、マトリクス ボタン ウィジェットは (M+N) 本のポートピンだけを使用して合計で (M×N) 個の交差部を監視することができます。

CSD センシング方式 (自己容量) を使用する場合、このウィジェットは同時に 1 点のみの交差位置で有効なタッチを検出できます。

変調コンデンサ (CMOD)

自己容量センシング モードでの CSD ブロックの動作のために必要な外部コンデンサです。

変調器クロック

センサーのスキャン間に CSD ブロックから変調器出力をサンプリングするために使用されるクロック ソースです。このクロックが raw カウント カウンターにも供給されます。スキャン時間 (事前および事後処理時間を除く) は $((2^N - 1) / \text{変調器クロック周波数})$ (N はスキャンの分解能) により計算されます。

変調 IDAC

変調 IDAC はプログラマブルな定電流源であり、この出力は V_{REF} の AMUXBUS 電圧を維持するために、CSD ブロックのシグマ デルタ変調器の出力によって制御 (オン／オフ) されます。この IDAC によって供給される平均電流は、センサー コンデンサが引き出した平均電流に等しいです。

相互容量

ある電極 (例えば、TX) と他の電極 (例えば RX) 間の静電容量は相互容量として知られています。

負のノイズ閾値

負の方向に出るスプリアス信号から通常のノイズを識別するために使用される閾値です。このパラメーターは、低ベースライン リセット パラメーターと共に使用されます。

raw カウントが負のノイズ閾値を超えない (すなわち、ベースラインと raw カウントの差 (ベースライン-raw カウント) が負のノイズ閾値未満である) 限り、ベースラインは raw カウントの変化を追跡するために更新されます。

負の方向でスプリアス信号をトリガーする可能性があるシナリオは次のとおりです。電源投入時にセンサーに指が触れるとき、センサーの近く配置される金属の物体を除去するとき、耐液性のある CapSense 製品の水分を除去するとき、および他の急激な環境変化があるときです。

ノイズ (CapSense ノイズ)

センサーがオフ状態 (タッチなし) で、ピークツーピーク カウントとして測定される raw カウントの変化です。

ノイズ閾値

センサー用にノイズから信号を識別するために使用されるパラメーターです。(raw カウント-ベースライン) の差がノイズ閾値を超える場合、おそらく有効な信号を示します。差分がノイズ閾値に下回る場合、raw カウントはノイズのみを含みます。

オーバーレイ

静電容量センサーをカバーしタッチ面として機能するプラスチックやガラスなどの非導電材料です。センサーがある PCB はオーバーレイの下に直接配置されるか、またはスプリングを介して接続されます。製品の筐体は常にオーバーレイになります。

寄生容量 (C_P)

寄生容量は PCB 配線、センサパッド、ビアおよびエアギャップによって与えられるセンサー電極の固有容量です。寄生容量は CSD の感度を減らすため、望ましくないものです。

近接センサー

あらゆる物理的な接触なしに近くの物体の存在を検知できるセンサーです。

ラジアル スライダー

指の物理的な位置を検出するために特定の円形状に配置された複数のセンサーを含むウィジェットです。

raw カウント

センサーの物理的静電容量を示す CapSense ハードウェア ブロックの未処理デジタル カウント出力です。

リフレッシュ間隔

センサーの 2 回の連続スキャンの間の時間です。

スキャン分解能

CSD ブロックによって生成される raw カウントの分解能 (ビット数) です。

スキャン時間

センサーのスキャンを完了するために必要な時間です。

自己容量

回路のグラウンドと電極間の静電容量です。

感度

センサー静電容量の変化に応じる raw カウントの変化であり、(単位: カウント/pF) で表します。センサーの感度は基板レイアウト、オーバーレイ特性、センシング方式およびチューニング パラメーターに依存します。

センス クロック

CSD センシング方式用のスイッチト コンデンサ回路のフロント エンドを実装するために使用されるクロック ソースです。

センサー

[静電容量センサー](#)を参照してください。

センサー自動リセット

センサーがシステム故障によって誤ったタッチ状態の報告から防ぎ、または金属物体がセンサーの近くに連続的に存在する場合の設定。

センサー自動リセット機能が有効になった場合、ベースラインは差分カウントがノイズ閾値を超えても常に更新されます。このように、センサーが無期限のオン状態を報告しないようにします。センサー自動リセットが無効化されていると、ベースラインは差分カウントがノイズ閾値を下回った場合にのみ更新されます。

センサー連結

[センサー連結](#)を参照してください。

シールド電極

センサーの周囲を覆う銅トレースで水または他の液体による誤タッチを防止します。シールド電極は CSD ブロックからのシールド信号出力によって駆動されます。[被駆動シールド](#)を参照してください。

シールド タンク コンデンサ (C_{SH})

高い寄生容量を持つ広いシールド層がある場合、CSD シールドの駆動能力を強化するために使用されるオプションの外部コンデンサ (C_{SH} タンク コンデンサ) です。

信号 (CapSense 信号)

差分カウントは信号とも呼ばれます。差分カウントを参照してください。

信号対雑音比 (SNR)

タッチしたときのセンサーの信号とタッチしないときのセンサーのノイズ信号との比率です。

スライダ分解能

スライダが分解された指の位置の総数を示すパラメーターです。

タッチパッド

特定の水平と垂直な様式で配置された複数のセンサーからなり、タッチの X および Y 位置を検出するウィジェットです。

トラックパッド

[タッチパッド](#)を参照してください。

チューニング

CapSense の動作に必要な様々なハードウェアおよびソフトウェアまたは閾値パラメーターの最適値を決定するプロセスです。

V_{REF}

PSoC 内にあるプログラマブル基準電圧ブロックであり、CapSense および ADC の動作に使用されます。

ウィジェット

単一センサーまたは同様のセンサー グループで構成される CapSense コンポーネントのユーザー インターフェース要素です。ボタン、近接センサー、リニア スライダー、ラジアル スライダー、マトリックス ボタンおよびタッチパッドはサポートされているウィジェットです。

改版履歴



改訂履歴

文書名: AN75400 - PSoC® 3 および PSoC® 5LP CapSense®デザイン ガイド		
文書番号: 001-80490		
版	発行日	変更内容
**	06/13/2012	これは英語版 001-75400 Rev. **を翻訳した日本語版 001-80490 Rev. **です。
*A	06/30/2015	これは英語版 001-75400 Rev. *B を翻訳した日本語版 001-80490 Rev. *A です。
*B	01/08/2020	これは英語版 001-75400 Rev. *E を翻訳した日本語版 001-80490 Rev. *B です。