# CY8C21x34/B

# CapSense®設計ガイド

文書番号: 001-78915 Rev. *A

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): 800.858.1810
Phone (Intnl): 408.943.2600
http: //www.cypress.com

## 著作権

© Cypress Semiconductor Corporation, 2010–2015. 本文書に記載される情報は、予告なく変更される場合があります。Cypress Semiconductor Corporation (サイプレス セミコンダクタ社) は、サイプレス製品に組み込まれた回路以外のいかなる回路を使用することに対して一切の責任を負いません。サイプレス セミコンダクタ社は、特許またはその他の権利に基づくライセンスを譲渡することも、または含意することもありません。サイプレス製品は、サイプレスとの書面による合意に基づくものでない限り、医療、生命維持、救命、重要な管理、または安全の用途のために使用することを保証するものではなく、また使用することを意図したものでもありません。さらにサイプレスは、誤動作や誤りによって使用者に重大な傷害をもたらすことが合理的に予想される生命維持システムの重要なコンポーネントとしてサイプレス製品を使用することを許可していません。生命維持システムの用途にサイプレス製品を供することは、製造者がそのような使用におけるあらゆるリスクを負うことを意味し、その結果サイプレスはあらゆる責任を免除されることを意味します。

## 商標

PSoC Designer™、Programmable System-on-Chip™、PSoC Creator™、SmartSense™ はサイプレス セミコンダクタ コーポレーションの商標であり、PSoC®、CapSense®はサイプレス セミコンダクタ コーポレーションの登録商標です。本書で言及するその他全ての商標または登録商標は、それぞれの所有者に帰属します。

## ソース コード

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

## 免責条項

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

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

# 目次


**1. はじめに ..... 7**

1.1 概要 ..... 7

1.2 サイプレス CapSense のドキュメント体系 ..... 7

1.3 CY8C21x34/B CapSense プラスファミリの特長 ..... 9

1.3.1 高性能タッチ検知機能 ..... 9

1.3.2 デバイスの特長 ..... 9

1.4 本書の表記法 ..... 10

**2. CapSense の技術 ..... 11**

2.1 CapSense の原理 ..... 11

2.2 CY8C21x34/B の CapSense モデル ..... 13

2.2.1 CapSense シグマ デルタ (CSD) ..... 13

2.2.2 ADC 機能を備えた CSD (CSDADC) ..... 14

2.2.3 SmartSense 自動チューニング ..... 14

**3. CapSense 設計ツール ..... 16**

3.1 概要 ..... 16

3.1.1 PSoC Designer およびユーザー モジュール ..... 16

3.1.2 汎用 CapSense コントローラー キット ..... 17

3.1.3 汎用 CapSense コントローラー モジュール基板 ..... 18

3.1.4 CapSense データ表示ツール ..... 18

3.2 ユーザー モジュール概要 ..... 18

3.3 CapSense ユーザー モジュール グローバル アレイ ..... 19

3.3.1 Raw カウント ..... 19

3.3.2 ベースライン ..... 19

3.3.3 差分カウント (信号) ..... 19

3.3.4 センサー状態 ..... 19

3.3.5 信号 ..... 20

3.4 CSD ユーザー モジュールのコンフィギュレーション ..... 20

3.4.1 クロックプリスケーラなしの CSD ..... 20

3.4.2 クロックプリスケーラありの CSD ..... 20

3.5 CSD ユーザー モジュール パラメーター ..... 20

3.6 CSD ユーザー モジュール高レベルのパラメーター ..... 21

3.6.1 指閾値 ..... 21

3.6.2 ノイズ閾値 ..... 21

3.6.3 ベースライン更新閾値 ..... 21
3.6.4 センサーの自動リセット ..... 21
3.6.5 ヒステリシス ..... 21
3.6.6 デバウンス ..... 22
3.6.7 負のノイズ閾値 ..... 22
3.6.8 低ベースライン値リセット ..... 22
3.6.9 高レベル パラメーターの推奨事項 ..... 22
3.7 CSD ユーザー モジュール低レベル パラメーター ..... 22
3.7.1 スキャン速度 ..... 23
3.7.2 分解能 ..... 23
3.7.3 リファレンス ..... 23
3.7.4 リファレンス値 ..... 24
3.7.5 プリスケーラ期間 ..... 24
3.7.6 シールド電極出力 ..... 24
3.8 CSDADC ユーザー モジュールのコンフィギュレーション ..... 24
3.8.1 PRS16 クロック源が備わった CSDADC ..... 24
3.8.2 PRS8 クロックソースが備わった CSDADC ..... 24
3.8.3 PWM8 クロックソースが備わった CSDADC ..... 24
3.8.4 VC2 クロックソースが備わった CSDADC ..... 25
3.9 CSDADC ユーザー モジュール パラメーター ..... 25
3.10 低レベル パラメーター ..... 25
3.10.1 スキャン速度 ..... 25
3.10.2 分解能 ..... 25
3.10.3 リファレンス ..... 25
3.10.4 リファレンス値 ..... 26
3.10.5 プリスケーラ期間 ..... 26
3.10.6 シールド電極出力 ..... 26
3.10.7 ADC Enabled ..... 26
3.11 SmartSense ユーザー モジュール パラメーター ..... 26
3.12 低レベル パラメーター ..... 27
3.12.1 シールド電極出力 ..... 27
3.12.2 変調器キャパシタ ピン ..... 27
3.12.3 フィードバック抵抗ピン ..... 27
3.12.4 閾値設定モード ..... 27
3.12.5 感度レベル ..... 27
3.12.6 指閾値 ..... 27

**4. ユーザー モジュールによる CapSense の性能のチューニング ..... 28**

4.1 一般的な注意事項 ..... 28
4.1.1 信号、ノイズ、および SNR (信号対ノイズ比) ..... 28
4.1.2 充電／放電速度 ..... 29
4.1.3 ベースライン値更新閾値の検証の重要性 ..... 30
4.2 CSD および CSDADC ユーザー モジュールのチューニング ..... 31
4.2.1 チューニングを行うためのハードウェアおよびソフトウェア セットアップ ..... 32

4.2.2 プリスケーラの選択 ..... 32
4.2.3 $R_b$ に応じて Raw カウント範囲を設定 ..... 33
4.2.4 高レベルパラメーターを設定する ..... 33
4.3 SmartSense ユーザー モジュールのコンフィギュレーション ..... 33

**5. 設計の注意事項 ..... 36**

5.1 オーバーレイの選択 ..... 36
5.2 ESD 保護 ..... 37
5.2.1 予防 ..... 37
5.2.2 転向 ..... 37
5.2.3 クランプ ..... 37
5.3 EMC (電磁環境適合性) の注意点 ..... 37
5.3.1 放射性干渉 ..... 37
5.3.2 放射妨害波 ..... 38
5.3.3 伝導イミュニティおよびエミッション ..... 38
5.4 ソフトウェアのフィルター処理 ..... 38
5.5 消費電力 ..... 39
5.5.1 システム デザインの推奨事項 ..... 39
5.5.2 スリープ スキャン方式 ..... 39
5.5.3 応答時間と消費電力の比較 ..... 40
5.5.4 平均消費電力の測定 ..... 40
5.6 ピンの割り当て ..... 41
5.7 プリント基板レイアウトのガイドライン ..... 41

**6. 耐水性 ..... 42**

6.1 シールド電極とガード センサー ..... 42
6.1.1 シールド基板 ..... 42
6.1.2 ガード センサー ..... 45
6.2 設計に関する推奨事項 ..... 47

**7. 近接検知 ..... 48**

7.1 近接センサーの種類 ..... 48
7.1.1 ボタン ..... 48
7.1.2 ワイヤ ..... 48
7.1.3 プリント基板配線 ..... 48
7.1.4 センサー連結 ..... 49
7.2 設計に関する推奨事項 ..... 49

**8. 低消費電力設計の注意事項 ..... 50**

8.1 追加的な省電力技術 ..... 50
8.1.1 駆動モードをアナログ HI-Z にセットする ..... 50
8.1.2 まとめ ..... 51
8.1.3 スリープ モードの混乱 ..... 51
8.1.4 保留中の割り込み ..... 51
8.1.5 グローバル割り込みのイネーブル: ..... 51

8.2 ウェイクアップ後の実行シーケンス ..... 52
8.2.1 PLL モードの有効化 ..... 52
8.2.2 グローバル割り込み有効化の実行 ..... 52
8.2.3 スレーブ モードを持った $I^2C$ スレーブ ..... 52
8.2.4 スリープ タイマー ..... 52

**9. リソース ..... 53**

9.1 ウェブサイト ..... 53
9.2 データシート ..... 53
9.3 テクニカル リファレンス マニュアル ..... 53
9.4 開発キット ..... 53
9.4.1 汎用 CapSense コントローラー キット ..... 53
9.4.2 汎用 CapSense モジュール基板 ..... 53
9.4.3 インサーキット エミュレーション(ICE)キット ..... 54
9.5 PSoC Programmer ..... 54
9.6 MultiChart ..... 54
9.7 PSoC Designer ..... 54
9.8 サンプル コード ..... 54
9.9 設計サポート ..... 55
9.10 改訂履歴 ..... 55

# 1. はじめに

## 1.1 概要

本書は、静電容量タッチ検知 (CapSense®) の機能を CapSense Plus™コントローラーの CY8C21x34/B ファミリに使用する設計ガイダンスを提供します。このガイドでは以下の項目について紹介します。

- CapSense プラスコントローラーの CY8C21x34/B ファミリの特長
- CapSense の動作原理
- CapSense 設計ツールの概要
- 最適な性能を得るための CapSense タッチ検知システムの詳細チューニングガイド
- 耐水性と近接検出などの優れた特長
- 電気的および機械的なシステム設計の考慮点
- CapSense をシステムに組み込むための追加リソースおよびサポート

## 1.2 サイプレス CapSense のドキュメント体系

図 1-1 と表 1-1 でサイプレス CapSense 文書体系をまとめます。これらのリソースにより、CapSense 製品の設計を正常に完了するために必要な情報へ迅速にアクセスできます。

図 1-1 に静電容量検知の製品設計サイクルの一般的なフローを示します。本ガイドに記載されている情報は、緑色でハイライト表示されたトピックに最も関連があります。表 1-1 には、図 1-1 で付番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

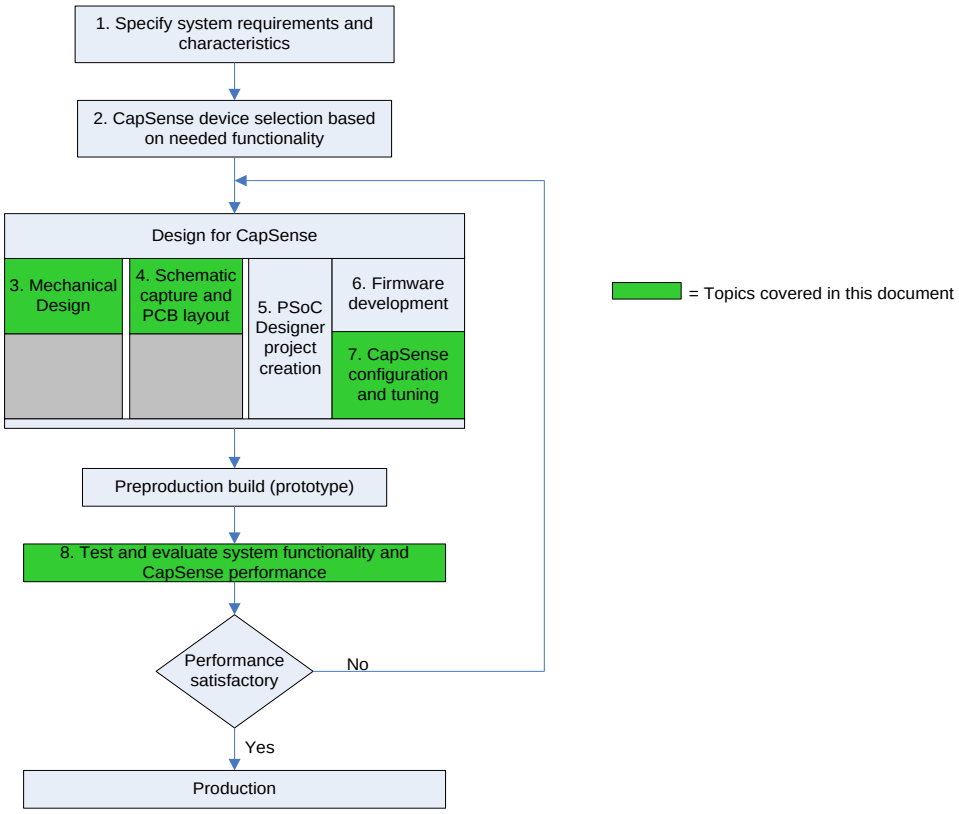



表 1-1.図 1-1 中に番号付けされた設計タスクをサポートしているサイプレスのドキュメント

| 図 1-1 内の設計タスク番号 | サイプレスの **CapSense** 用サポート ドキュメント |
|---|---|
| 1 | CapSense 入門<br>PSoC 1 入門 |
| 2 | CapSense 入門<br>CY8C21x34/B CapSense デバイス データシート |
| 3 | CapSense 入門<br>PSoC ファミリ固有の CapSense 設計ガイド (本書)<br>CapSense アプリケーション ノート |
| 4 | CapSense 入門<br>PSoC ファミリ固有の CapSense®設計ガイド (本書)<br>CapSense アプリケーション ノート |
| 5 | PSoC Designer™ユーザー ガイド |
| 6 | CapSense アプリケーション ノート<br>CapSense コードの例<br>PSoC ファミリ固有のテクニカル リファレンス マニュアル (CY8C21x34/B 用)<br>AN2014 - PSoC® 1 プログラミングの基本<br>AN2015 - PSoC® 1 - フラッシュ＆E2PROM 入門<br>AN44168 - 外部マイクロコントローラー (HSSP) を用いた PSoC® 1 デバイス プログラミング<br>PSoC® 1 ISSP プログラミング仕様書 |

| 図 1-1 内の設計タスク番号 | サイプレスの **CapSense** 用サポート ドキュメント |
|---|---|
| 7 | PSoC ファミリ固有の CapSense 設計ガイド (本書)<br>PSoC ファミリ固有の CapSense ユーザー モジュールデータシート (CSD、CSDADC、および SmartSense)<br>PSoC ファミリ固有のテクニカル リファレンス マニュアル (CY8C21x34/B 用)<br>CapSense コントローラー コード例の設計ガイド |
| 8 | PSoC ファミリ固有の CapSense 設計ガイド (本書)<br>CapSense コードの例 |

## 1.3 CY8C21x34/B CapSense プラスファミリの特長

サイプレスの CY8C21x34/B は低電力、ハイパフォーマンス、プログラマブルな CapSense コントローラー ファミリで次の特長を有します。

### 1.3.1 高性能タッチ検知機能

- プログラム可能な容量検知要素
  - CSD および CSDADC 静電容量センサー技術を用いて CapSense ボタン、スライダー、近接センサーの組合せをサポート
  - SmartSense™ 自動チューニングをサポート (CY8C21x34B)
  - ボタンとスライダーを実装する集積 API を含む
  - 24 個の静電容量ボタンと 4 つのスライダーをサポート
  - 最大 5cm の近接検知をサポート (オンボード プリント基板配線)
  - シールド電極で水耐性性能を提供

### 1.3.2 デバイスの特長

- 高性能、低電力 M8C ハーバード アーキテクチャ プロセッサ
  - 最大 24MHz のクロックで動作する M8C プロセッサ
- 柔軟性のある内蔵メモリ
  - 最大 8KB のフラッシュと 512B の SRAM
  - エミュレート EEPROM
- プログラム可能な高精度クロック供給
  - 内部主発振器 (IMO): 24/48MHz ± 2.5%
  - ウォッチドッグとスリープ タイマー用の 32kHz 内部低速発振器 (ILO)
- 拡張 GPIO の特徴
  - プログラム可能なピン コンフィギュレーションを備えた最大 28 個の GPIO
  - すべての GPIO で 25mA のシンク電流に対応
  - GPIO での内部プルアップ抵抗、HI-Z、オープン ドレイン、ストロング駆動モード
- ペリフェラルの特長
  - カウンター、タイマー、パルス幅変調器 (PWM)、パルス幅識別器 (PWD)、ハードウェア 7 セグメント駆動
  - マスター、スレーブ、マルチマスター コンフィギュレーションの $I^2C$ コミュニケーション
  - SPI、UART、IRDA、1-Wire の通信プロトコル
- 動作条件
  - 動作電圧: 2.4V～5.25V、オンチップ スイッチ モード ポンプ (SMP) を使用して動作電圧を 1.0V まで低減可能
  - 温度範囲: –40°C～+85°C
- パッケージ
  - 16 ピン SOIC (150-MIL) ～32 ピン QFN (5 x 5mm)
  - 車載電子部品評議会 (AEC) 認定の車載用グレードパーツ – CY8C21334 と CY8C21534

## 1.4 本書の表記法

| 表記法 | 使用法 |
|---|---|
| `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 コードを示す<br>`C: \ ...cd\icc\` |
| イタリック フォント | ファイル名および参考ドキュメントを示す<br>*PSoC Designer User Guide* にある *sourcefile.hex* 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<br>[**Enter**] または [**Ctrl**] [**C**] |
| File > Open | メニュー パスを示す<br>File > Open > New Project |
| 太字 | 次の手順でコマンド、メニュー パス、アイコン名を示す<br>**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 | 数式を表示する<br>2 + 2 = 4 |
| 灰色のボックス内のテキスト | 製品の注意点や製品固有の機能を示す |

# 2. CapSense の技術

## 2.1 CapSense の原理

CapSense は、センサーとして設計された CapSense コントローラー上の各入力／出力静電容量を計測することで実現するタッチ検知技術です。図 2-1 に示される通り、n 個のセンサーを持つデザインにおいて、各センサー ピンの総静電容量が、$C_{x,1}$～$C_{x,n}$ の値を持つ等価集中コンデンサとしてモデリングされます。CY8C21x34/B デバイスへの内部回路が、各 $C_x$ の規模をデジタル コードに変換し、このデジタル コードは後処理用に保存されます。$R_b$ と $C_{MOD}$ に代表されるその他のコンポーネントは、CapSense コントローラーの内部回路によって使用されます。

図 2-1. CY8C21x34/B PSoC デバイス内の CapSense の実行

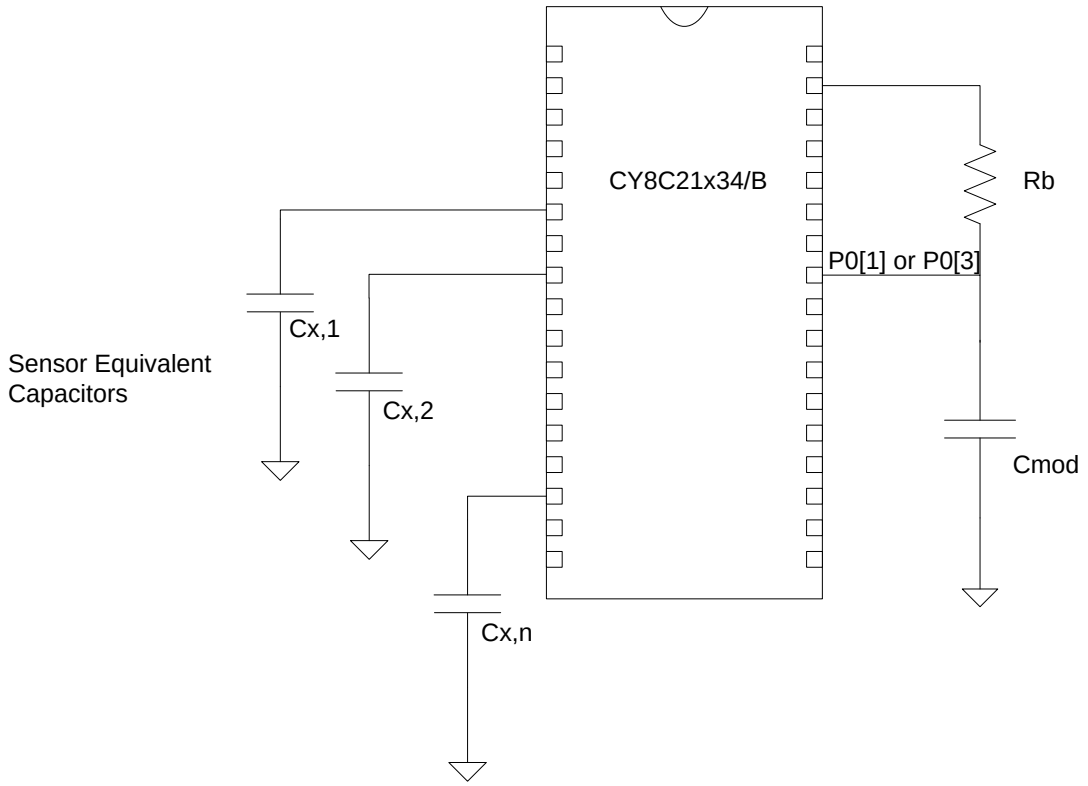
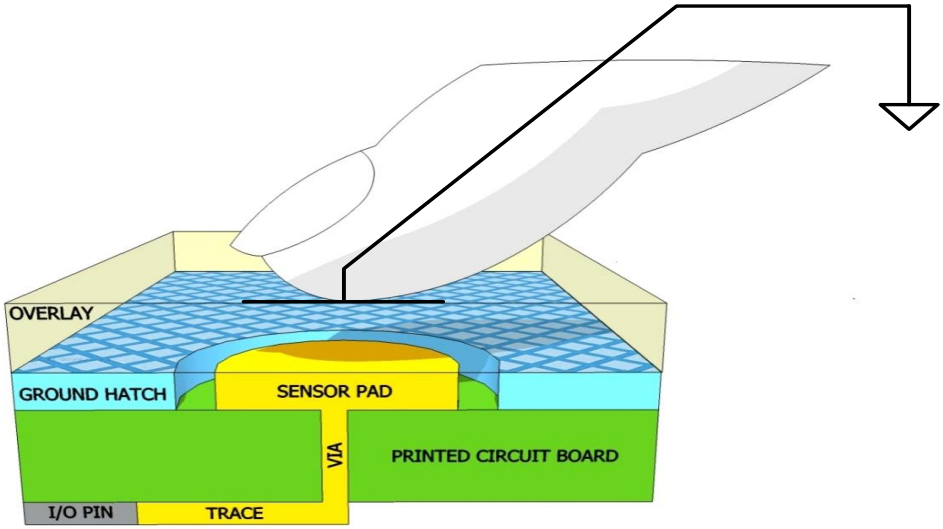



図 2-2 に示すように、各センサーの I/O ピンは、必要に応じて配線、ビアまたはその両方によりセンサー パッドに接続されます。オーバーレイはセンサー パッドを覆う非導電のカバーで、製品のタッチ インターフェースをコンフィギュレーションします。指がオーバーレイに接触すると、人体の伝導性と大きさにより、グランドに接続された導体板がセンサー パッドと平行に置かれるのと同じ状況になります。これを図 2-2 に示します。この配置は平行板コンデンサをコンフィギュレーションし、その静電容量は式 1 で与えられます。

図 2-2. 指によってアクティブにされたセンサーを持つ CapSense プリント基板の断面図




$$C_F = \frac{\varepsilon_0 \varepsilon_r A}{D}$$ 式 1

その中、

$C_F$ = 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

$\varepsilon_0$ = 真空の誘電率

$\varepsilon_r$ = オーバレイの比誘電率

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

平行板コンデンサに加えてオーバーレイに接触している指は、それ自身とすぐ近くにある他の導体との間に静電結合を引き起こします。このようにフリンジング磁界の影響は平行板コンデンサと比較して通常は小さく、無視することができます。

指がオーバーレイに接触しなくても、センサー I/O ピンは寄生容量 ($C_P$) 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。これらの導体は、グランド面、他の配線、製品のシャーシまたはエンクロージャ内の金属のいずれか等です。CapSense コントローラーは、センサー ピンに接続しているすべての静電容量 ($C_X$) を計測します。

指がセンサーに触れていない場合:

$$C_X = C_P$$ 式 2

指がセンサー パッドにある場合、

$$C_X = C_P + C_F$$ 式 3

一般的に、$C_P$ は $C_F$ より桁違いに大きい値です。通常、$C_P$ は 6pF～15pF の範囲ですが、極端な場合は、50pF になることもあります。$C_F$ は通常 0.1pF から 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はきわめて重要であり、これについては「ユーザー モジュールによる CapSense の性能のチューニング」で討論します。

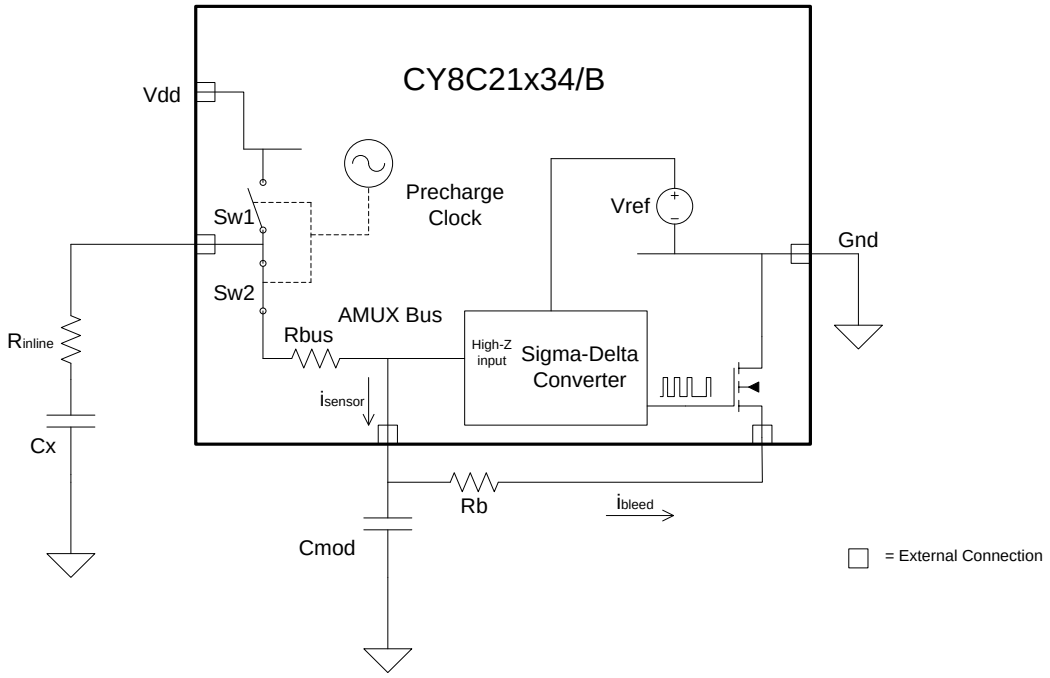
## 2.2 CY8C21x34/B の CapSense モデル

CY8C21x34/B デバイスは、センサー静電容量 ($C_x$) をデジタル カウントに変換するにあたり、数通りの CapSense 方式に対応しています。対応方式は、CapSense Sigma Delta (CSD)、ADC を用いた CSD (CSDADC)、SmartSense です。これらの方式は PSoC Designer ユーザー モジュールから実行され、詳細は本設計ガイドの節 3 で紹介されています。

### 2.2.1 CapSense シグマ デルタ (CSD)

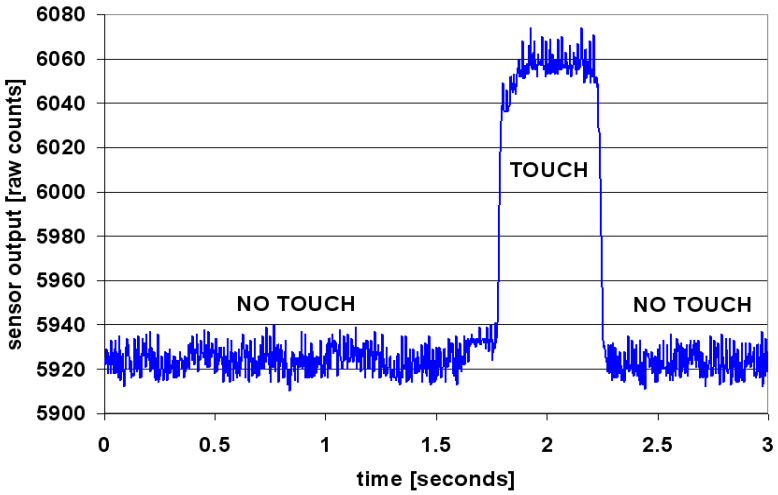
CY8C21x34/B デバイスの CSD 方式では、図 2-3 に示すように、$C_x$ がスイッチト キャパシタ回路に接続されます。センサー ($C_x$) は $V_{DD}$ とアナログ MUX (AMUX) バスに重ならない 2 つのスイッチ (Sw1 と Sw2) それぞれによって交互に接続されます。Sw1 と Sw2 はプリチャージ クロックによって駆動され、電流 ($I_{SENSOR}$) を AMUX バスに提供します。$I_{SENSOR}$ の大きさは、$C_x$ に正比例します。シグマ-デルタ 変調器は AMUX バス電圧をサンプリングし、定電流源 ($I_{DAC}$) を制御する変調ビット ストリームを生成します。これにより、AMUX が充電され、平均 AMUX バス電圧が $V_{REF}$ で維持されます。センサーは、変調コンデンサ ($C_{MOD}$) から $I_{SENSOR}$ を充電します。$C_{MOD}$ は R バスと共にローパスフィルターを形成し、シグマ-デルタ 変調器入力においてプリチャージ スイッチング トランジェントを減衰させます。

図 2-3 CSD ブロック図




AMUX 平均電圧を定常値 ($V_{REF}$) に保つ上で、Sigma-Delta コンバータは、ビットストリームのデューティサイクルを制御することで、平均ブリード電流 ($I_{BLEED}$) を、$I_{SENSOR}$ に一致させます。シグマ デルタ コンバーターはセンサー スキャンの間ビット ストリームを保存します。その積算値は $C_x$ に正比例した、Raw カウントと呼ばれたデジタル出力値となります。この 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。図 2-4 は、指でセンサーに触れてから離すまでの連続スキャン回数により CSD の Raw カウントをプロットしたものです。「CapSense の原理」で説明されているように、指の接触が $C_F$ の分だけ $C_x$ を増加させ、結果として Raw カウントが比例して増加します。定常状態の Raw カウント レベルの変化を事前に設定された閾値と比較することで、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. 指で接触中の CSD raw カウント




## 2.2.2 ADC 機能を備えた CSD (CSDADC)

CSDADC CapSense 方式は、静電容量の測定に加えて ADC 機能を含むことを除けば、CSD 方式と同一です。

### *2.2.2.1 ADC* の特長

- 絶対入力 ADC とレシオメトリック入力 ADC をランタイム モードで切り替えでき、異なるタイプのセンサーのスキャンを簡単に実行
- 絶対電圧入力モードのシングルスロープ ADC
- シングル スロープ ADC 用の較正メカニズム内蔵
- レシオメトリック入力に対応した積分型インクリメンタル ADC

## 2.2.3 SmartSense 自動チューニング

タッチ検知式のユーザーインタフェースのチューニングは、適切なシステムの動作と快適なユーザー操作のために不可欠なステップです。標準的な設計フローには、初期設計段階、システム統合中、および生産立上げ前の最終製造の微細調整でのセンサー インターフェースのチューニングが含まれています。チューニングは繰り返しプロセスなので時間がかかることがあります。SmartSense 自動チューニングは、ユーザー インターフェースの開発サイクルを簡易化するために開発されます。また、使用方法は簡単であり、プロトタイプから大量生産までの製品開発サイクル全体からチューニング プロセスを除外して、設計サイクル時間を大幅に短縮できます。SmartSense は、電源投入時に各 CapSense センサーを自動的に調整してから、実行中に最適なセンサー性能を監視し、維持します。SmartSense 自動チューニング技術では、プリント基板やオーバーレイによる製造のばらつきに適応し、LCD インバータ、AC ライン、スイッチング電源などのノイズ発生源を自動的に調整してノイズを取り除きます。SmartSense 自動チューニングにはコンフィグレーションによって 2.1KB ～ 3.7KB のフラッシュ、センサーごとに 29B の RAM が必要です。

### 2.2.3.1 プロセスのばらつき

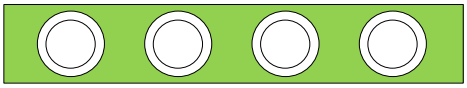
CY8C21x34B のための SmartSense ユーザー モジュール (UM) は、5pF～45pF のセンサー静電容量と共に使用できるように設計されています (一般的に $C_P$ の値は 10pF から 20pF の間です)。各センサーの感度パラメーターは、特定のセンサーの特性に基づいて自動的に設定されます。これにより、大量生産における歩留まりが上がります。これは、指定した 5pF～41pF の範囲内でセンサー間の $C_P$ 変化に関係なくすべてのセンサーから一貫性のある応答を維持できるためです。プリント基板 レイアウト、プリント基板 製造プロセスの変化、またはマルチソースのサプライチェーン内でのベンダー間の プリント基板 の違いにより、個々のセンサーの寄生容量が異なる可能性があります。センサーの感度はその寄生容量に応じて決まります。$C_P$ 値が高くなるとセンサー感度が低下し、フィンガー タッチ信号振幅が低下します。場合によっては、$C_P$ 値の変化がシステムの性能化を下げ、最適なセンサー性能を得られない (過敏か感度不足になる) 結果となり、最悪の場合センサーが機能しないことがあります。いずれの場合でも、システムを再調整する必要があります。場合によっては UI サブシステムを最適化する必要があります。SmartSense 自動チューニングがこのような問題を解決します。

SmartSense 自動チューニングはプラットフォーム設計を実現します。ノートパソコンの静電容量タッチ センサーのマルチメディアキーを想像してください；ボタンの間隔はノート PC のサイズとキーボードのレイアウトに左右されます。この例では、ワイド画面のマシンでは標準画面のモデルに比べてボタン間のスペースが大きくなります。ボタンの間隔がより広くなるということは、セ

ンサーと CapSense コントローラー間の配線を増やすということを意味し、センサーの寄生容量を高めることにつながります。つまり、CapSense ボタンの寄生容量は同じプラットフォーム デザインであってもモデルが違えば、異なることを意味します。これらのボタンの機能はすべてのノート PC で同じですが、センサーは各モデル毎にチューニングする必要があります。SmartSense では、チューニングが効果的に自動で行われるため、「CapSense 入門」内のプリント基板レイアウトで推奨されるベスト プラクティスを活用してプラットフォーム設計ができます。

図 2-5. 21 インチモデル用のノート PC マルチメディア キーのデザイン

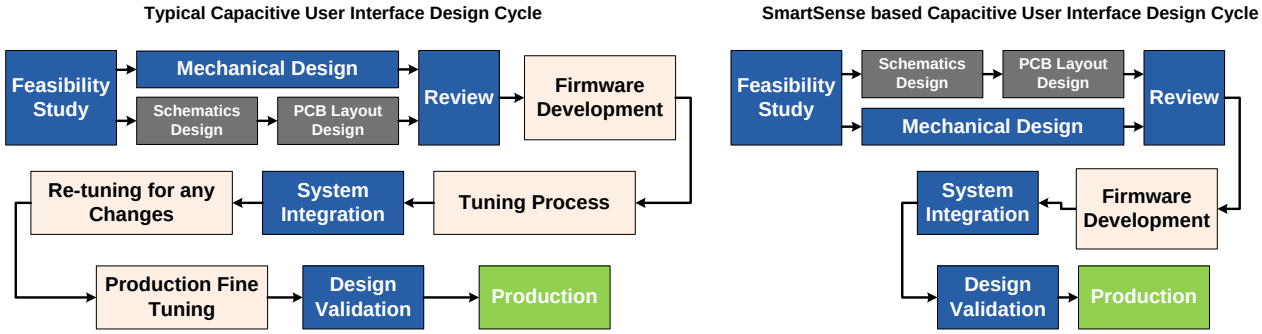
図 2-6. 機能とボタン サイズが同一の 15 インチ モデル用のノート PC マルチメディア キーの設計



### 2.2.3.2 デザイン サイクル タイムの短縮

通常、静電容量センサーのインターフェイス設計において最も時間のかかる作業は、ファームウェアの開発とセンサーのチューニングです。通常のタッチ検知コントローラーでは、同じ設計を異なるモデルに移植したり、プリント基板 やセンサー プリント基板レイアウトの機械的寸法に変更があった場合、センサーを返品しなければなりません。SmartSense 搭載の設計では、ファームウェアの開発が最小限に抑えられ、チューニングや再チューニングが必要ないため、こういった変更も問題になりません。これによって典型的なデザインサイクルがより速くなります。図 2-7 は、一般的なタッチ センサー コントローラーの設計サイクルと、SmartSense に基づいた設計を比較しています。

図 2-7. 一般的な静電容量インターフェース設計サイクルの比較



# 3. CapSense 設計ツール

## 3.1 概要

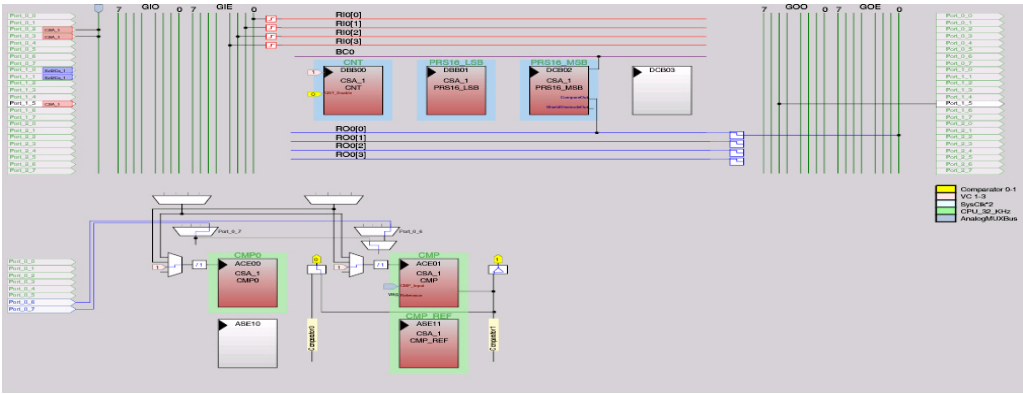
サイプレスは、CapSense 容量タッチ検知アプリケーション開発用のフルラインのハードウェアおよびソフトウェアを提供しています。CY8C21x34/B ファミリの基本開発システムには、本章で扱うコンポーネントが含まれます。注文情報については、「リソース」を参照してください。

### 3.1.1 PSoC Designer およびユーザー モジュール

サイプレスの専用統合設計環境である PSoC Designer では、アナログとデジタル ブロックのコンフィギュレーション、ファームウェアの開発、設計のチューニングとデバッグが可能です。アプリケーションは、ユーザー モジュールの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。ユーザー モジュールは、デバイス エディタ GUI 、あるいはファームウェアで特定のレジスタに書込むことで設定されます。PSoC Designer は、専用 C コンパイラおよび組み込みのプログラマと一体化しています。複雑なデザイン用には pro コンパイラがあります。

CSD、CSDADC、SmartSense ユーザー モジュールは、スイッチト キャパシタ回路、アナログ マルチプレクサ、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 センサーを動作させます。その他アナログとデジタル周辺のためのユーザー モジュールは、 $I^2C$、SPI、TX8、タイマー、PWM、などといった追加機能を実行するために用意されています。

図 3-1. PSoC Designer デバイス エディタ




#### *3.1.1.1 CapSense* ユーザー モジュールのスタート

新しい CY8C21x34/B プロジェクトを PSoC Designer で作成:

1. 新しい CY8C21x34/B プロジェクトを PSoC Designer で作成する
2. $I^2C$ や LCD など、専用ピンを必要とするユーザー モジュールを選択、配置し、ポートとピンを割り当てる
3. CSD、CSDADC、SmartSense ユーザー モジュールを選択し、配置する
4. ユーザー モジュールを右クリックしてユーザー モジュール ウィザードにアクセスする
5. ボタンのセンサー カウント、スライダーのコンフィグレーション、ピンの割り当てと接続を設定する
6. ピンとユーザー モジュールのグローバル パラメーターを設定する

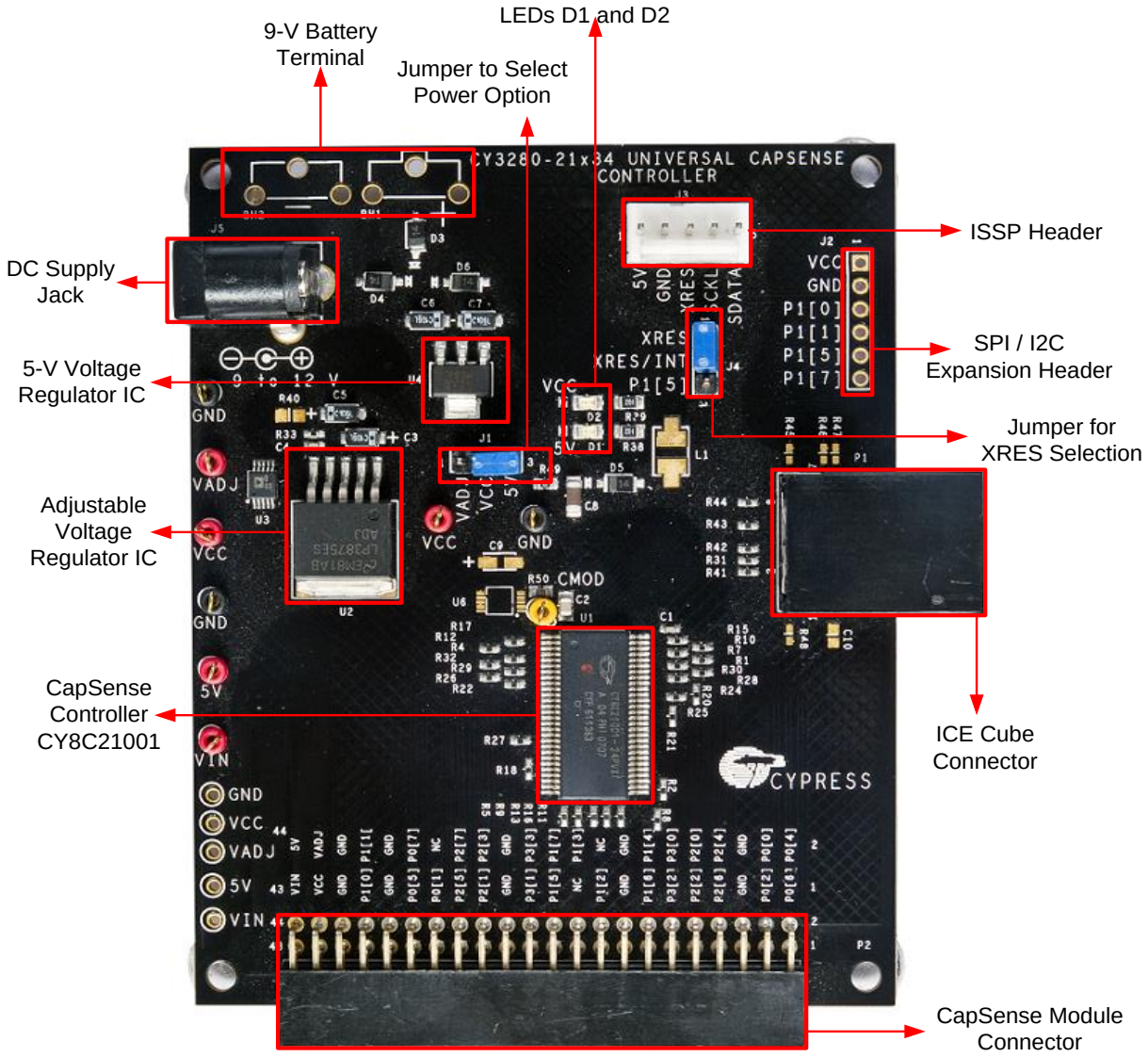
7. アプリケーションを生成し、アプリケーション エディターを開く

8. ユーザー モジュール データシート: (CY8C21x34) からのサンプル コードを適用し、ボタンやスライダーを実装する

### 3.1.2 汎用 CapSense コントローラー キット

汎用 CY3280-BK1 CapSense コントローラー キットは、あらかじめ設定された制御回路、プラグイン ハードウェアを備えており、試作やデバッグを簡単に行えます。チューニングとデータ取得用にプログラミングと I²C-USB ブリッジ ハードウェアが含まれています。

図 3-2. CY3280-21x34 CapSense コントローラー キット



### 3.1.3 汎用 CapSense コントローラー モジュール基板

サイプレスのモジュール基板には、お客様の用途に応じたさまざまなセンサー、LED、およびインタフェースが用意されています。

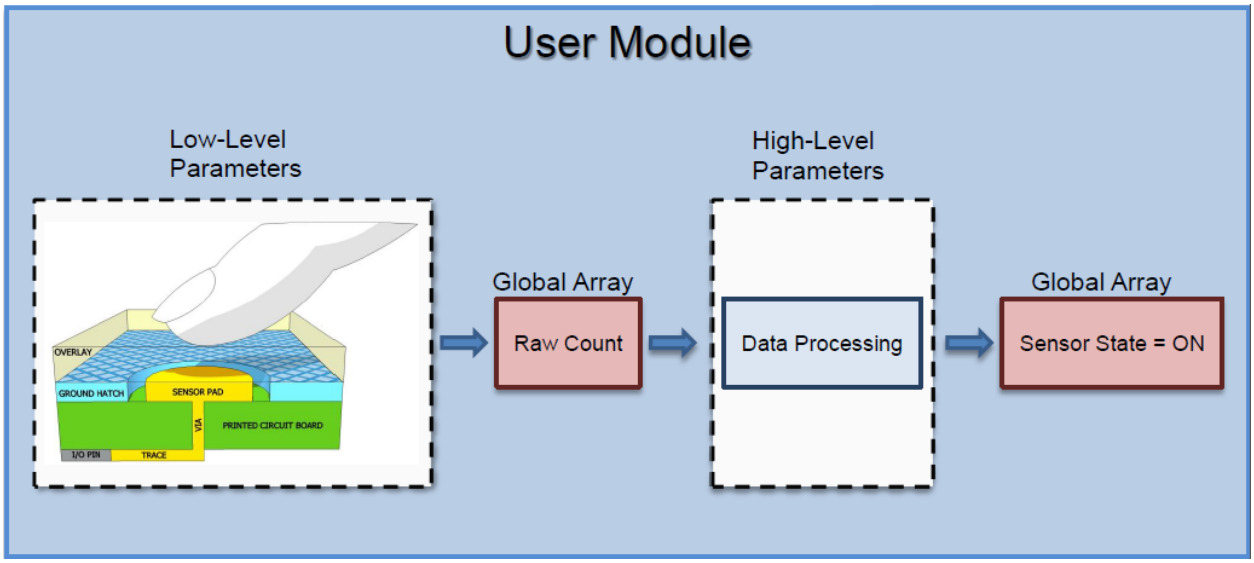
- CY3280-BSM シンプル ボタン モジュール
- CY3280-BMM マトリクス ボタン モジュール
- CY3280-SLM リニア スライダー モジュール
- CY3280-SRM ラジアル スライダー モジュール
- CY3280-BBM 汎用 CapSense プロトタイプ作成モジュール

### 3.1.4 CapSense データ表示ツール

CapSense 設計プロセスの多くの場面で、チューニングやデバッグの目的で CapSense データ (Raw カウント、ベースライン、差分カウント、など) を見たいと思うでしょう。これは、マルチチャートとブリッジ制御パネルという 2 つの CapSense データ表示ツールを使用して実現できます。ツールの詳細は、アプリケーション ノート「AN2397 – CapSense Data Viewing Tools 」で討論します。

## 3.2 ユーザー モジュール概要

図 3-3. ユーザー モジュール ブロック図




ユーザー モジュールには、物理的な検知からデータ処理までの CapSense システム全体が含まれます。ユーザー モジュールの動作は、いくつかのパラメーターを使用して設定できます。パラメーターはセンサーシステムの異なるパーツに影響し、低レベルおよび高レベルのパラメーターに分けられ、グローバル アレイを使用して互いに通信します。

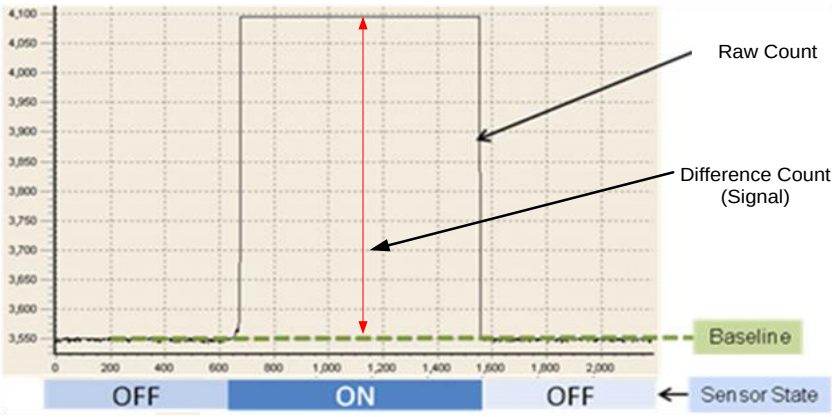
スキャン センサーの速度と分解能などの低レベルパラメーターは、物理層でのセンサー動作を定義し、これらは静電容量から Raw カウントへの変換に連係します。低レベル パラメーターは検知方式の各種類によって異なり、それぞれ「CSDADC ユーザー モジュールのコンフィギュレーション」、「CSDADC ユーザー モジュール パラメーター」と「SmartSense ユーザー モジュール パラメーター」で説明されています。

デバウンス カウントやノイズ閾値といった高レベル パラメーターでは、Raw カウントを処理する方法を定義して、 センサーのオン/オフ状態やスライダー上の指の予想位置などの情報を生成します。これらのパラメーターはすべての検知方式に同様であり、「CSD ユーザー モジュール高レベルのパラメーター」に説明されています。

## 3.3 CapSense ユーザー モジュール グローバル アレイ

CapSense ユーザー モジュール パラメーターについて習得する前に、CapSense システムで使用する特定のグローバル アレイに精通しておく必要があります。これらのアレイは手動で変更することはできませんが、デバッグ作業用に検査できます。

図 3-4. Raw カウント、ベースライン値、差分カウント、およびセンサー状態




### 3.3.1 Raw カウント

CapSense コントローラーのハードウェア回路はセンサー容量を測定します。測定値は、ユーザー モジュール API *UMname*_ScanSensor() 呼び出し時に、Raw カウントと呼ばれるデジタル フォームに保存されます。*UMname* は CSD、CSDADC、SmartSense となります。

センサーの Raw カウントは、そのセンサー容量に比例しています。センサー静電容量の値が増加すると、Raw カウントが増加します。センサーの Raw カウント値は、*UMname*_waSnsResult[]整数アレイに保存されます。このアレイは *UMname.h* のヘッダ ファイルで定義されています。

### 3.3.2 ベースライン

温度や湿度などの段階的な環境変化は、センサーの Raw カウントに影響を及ぼし、その結果カウントが変動します。

ユーザー モジュールは複雑なベースライン作成アルゴリズムを使用し、これらの変化を補正します。アルゴリズムでは、ベースライン値変数を使用してこの補正を行います。ベースライン値変数は、Raw カウント値の段階的な変動を追跡します。本質的に、ベースライン変数はデジタル ローパス フィルターの出力は、どの入力 Raw カウント値が供給されるために保持する。ベースライン アルゴリズムはユーザー モジュール API *UMname*_UpdateSensorBaseline (*UMname* は CSD、CSDADC、または SmartSense) によって実行されます。

センサーのベースライン値は、*UMname*_waSnsBaseline[]整数アレイに保存されます。このアレイは *UMname.h* のヘッダ ファイルで定義されています。

### 3.3.3 差分カウント (信号)

差分カウントはセンサーの信号としても知られ、センサーの Raw カウントとベースライン値の間でのカウント差として定義されます。センサーがアクティブでないときに差分カウントはゼロです。センサーのアクティブ化 (触ることにより) によって、差カウントが正の値となります。

センサーの差分カウント値は *UMname*_waSnsDiff[] 整数アレイに保存され、*UMname* は CSD、SmartSense、または CSA_EMC とすることができます。このアレイは *UMname*.h のヘッダ ファイルで定義されています。差分カウント変数はユーザー モジュール API *UMname*_UpdateSensorBaseline()により更新されます。

### 3.3.4 センサー状態

センサー状態は物理的センサーのアクティブ／非アクティブなステータスを示します。センサー状態は、指が接触すると 0 から 1 に変わり、指が離れると 0 に戻ります。

センサー状態は *UMname*_baSnsOnMask[]array という名のバイト アレイに保存されます。*UMname* は CSD、CSDADC、または SmartSense となるでしょう。各アレイエレメントは 8 つの連続センサーのセンサー状態を保存します。センサー状態はユーザー モジュール *API* UMname_bIsAnySensorActive()により更新されます。

### 3.3.5 信号

このアレイは SmartSense ユーザー モジュールのみに適用されます。信号を取得するために差分カウントが正規化されます。通常、センサーが非アクティブの時、信号が 0 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、プラスの信号値が発生します。このアレイは閾値モード パラメーターが手動モードにセットされた場合に SmartSense ユーザー モジュールをチューニングするために使用されます。

信号値は SmartSense_baSnsSignal[]という整数アレイに格納されます。

## 3.4 CSD ユーザー モジュールのコンフィギュレーション

CSD ユーザー モジュールには 2 つの選択可能なクロックコンフィギュレーションがあります。これらのコンフィギュレーションは、図 2-3 (ページ 13) に示すように、CSD フロントエンドのスイッチト キャパシタ回路に異なる信号ソースを使用します。

CSD ユーザー モジュールを PSoC Designer プロジェクトに初めて導入するときに、クロックコンフィギュレーションを選択してください。この選択は、CSD ユーザー モジュールを右クリックし、「**User Module Selection Options**」(ユーザー モジュール選択オプション) を選択すると変更できます。

### 3.4.1 クロックプリスケーラなしの CSD

このコンフィグレーションでは 16 ビットの疑似ランダム シーケンス(PRS) が CSD フロントエンドのスイッチト キャパシタ回路の信号ソースとして使用されます。システム クロック (IMO) が PRS のクロック源です。PRS はクロックの周波数スペクトラムを拡散し、良好なノイズ耐性を提供します。このコンフィグレーションは 3 つのデジタル ブロックを必要とします。

### 3.4.2 クロックプリスケーラありの CSD

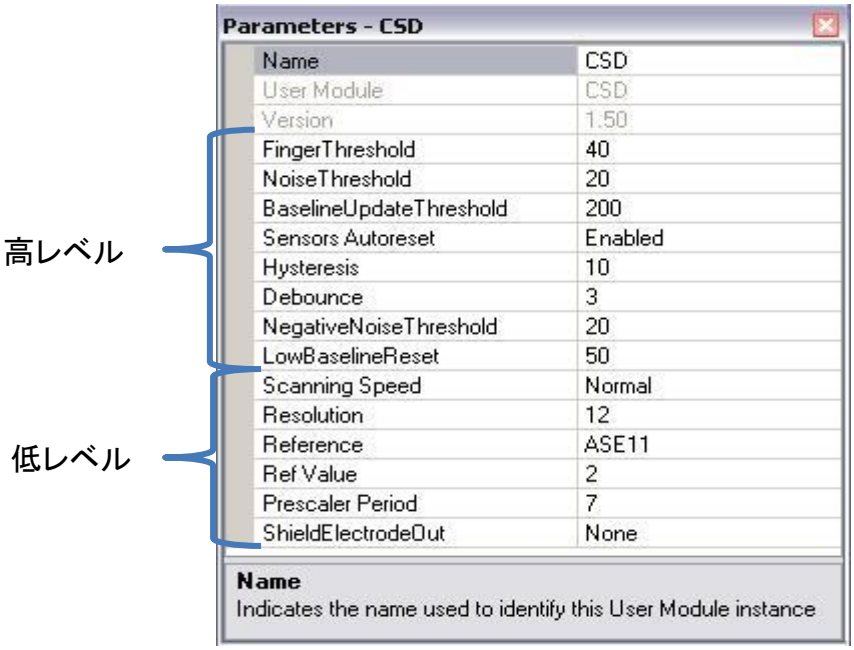
このコンフィグレーションでは 8 ビットの PRS が CSD フロントエンドのスイッチト キャパシタ回路の信号ソースとして使用されます。IMO は調整可能なプリスケーラによって分周され、PRS のクロック源として使用されます。PRS はクロックの周波数スペクトラムを拡散し、ノイズ耐性を提供します。クロック プリスケーラ コンフィギュレーションなしの CSD 内で使用される 16 ビット PRS は、より良好なノイズ耐性を提供します。このコンフィグレーションは 3 つのデジタル ブロックを必要とします。

これはより低いスイッチング周波数を必要とするため、高い $C_P$ 値の環境で動作する場合は、このコンフィグレーションを使用します。$C_P$ とスイッチング周波数の関係は、「プリスケーラの選択」で説明されています。

## 3.5 CSD ユーザー モジュール パラメーター

CSD ユーザー モジュール パラメーターは高レベルと低レベルのパラメーターに分類されます。CSD ユーザー モジュールのパラメーター一覧、およびその分類については、図 3-5 を参照してください。

図 3-5. PSoC Designer - CSD パラメーター ウィンドウ



## 3.6 CSD ユーザー モジュール高レベルのパラメーター

### 3.6.1 指閾値

指閾値パラメーターはユーザー モジュールで使用し、センサーのアクティブ／非アクティブ状態を判定します。センサーの差分カウントが指閾値より大きい場合、センサーはアクティブとして判定されます。この定義では、ヒステリシス レベルが 0 に設定され、デバウンスが 1 に設定されていることを前提としています。

可能な値は 5 から 255 です。

### 3.6.2 ノイズ閾値

ユーザー モジュールはノイズ閾値を使用して Raw カウントのノイズ定数の上限を解釈します。個々のセンサーについては、Raw カウントがベースラインを上回り、これらの差がこの閾値を上回る場合、ベースライン更新アルゴリズムは一時停止します。

スライダー センサーの場合、差分カウントがノイズ閾値を超えるとセントロイド計算は停止されます。

可能な値は 3～255 です。正常なユーザー モジュールの動作のためには、ノイズ閾値が、指の閾値からヒステリシス設定を差し引いた値よりも高くなってはいけません。

### 3.6.3 ベースライン更新閾値

Raw カウント値が現在のベースライン値を上回っており、その差がノイズ閾値を下回る場合は、現在のベースライン値と Raw カウントの差は「バケツ」内データに加算されます。バケツが満杯になると、ベースライン値が増分されバケツは空になります。このベースライン値更新閾値パラメーターは、ベースライン値が増分するためにバケツが到達しなければならない閾値を設定します。

可能な値は 0 から 255 です。

### 3.6.4 センサーの自動リセット

センサー自動リセットパラメーターにより、ベースライン値は常に更新されるのか、それとも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るかが決定されます。

センサーの自動リセットが有効にされていると、ベースライン値は常に更新されます。この設定により、センサーの最大時間は制限されますが、センサーに何も触れないのに Raw カウントが誤って上昇した場合に、センサーが恒久的にオンになることを妨ぐことができます。この突然の上昇の原因には、大幅な電源の電圧変化、高エネルギー RF ノイズ源、非常に速い温度変化があります。

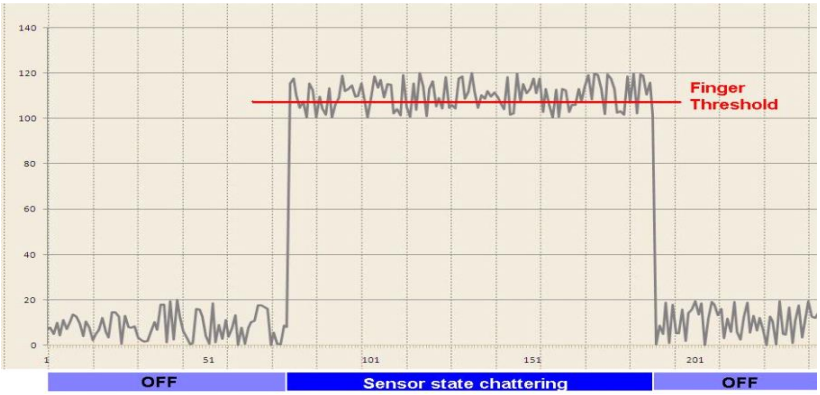
センサーの自動リセットが無効化されていると、ベースライン値は差分カウントがノイズ閾値パラメーターを下回った場合にのみ更新されます。

可能な値は、Enabled と Disabled です。

### 3.6.5 ヒステリシス

ヒステリシス設定は、システム ノイズによるチャタリングがセンサーを「ON」状態にすることを防ぎます。ヒステリシスの関数は式 4 で与えられます。式は、デバウンスが 1 にセットされることを前提にします。

図 3-6. ヒステリシスがゼロに設定されている場合のセンサー状態対差分カウント




$$if DifferenceCount \geq FingerThreshold + Hysteresis,\ SensorState = ON$$
$$if DifferenceCount \leq FingerThreshold - Hysteresis,\ SensorState = OFF \qquad \text{式 4}$$

### 3.6.6 デバウンス

デバウンス パラメーターは、Raw カウントのスパイクによりセンサー状態がオフからオンに変わることを防ぎます。センサー状態 OFF から ON に移行するには、差カウントの値が、指定したサンプル数の指の閾値とヒステリシスレベルの合計より大きくあり続ける必要があります。

可能な値は 1～255 です。1 をセットするとデバウンスは起こりません。

### 3.6.7 負のノイズ閾値

ネガティブノイズ しきい値のパラメーターは、マイナスの差分カウントしきい値として機能します。Raw カウントが、Low Baseline Reset パラメーターが指定するサンプル数のマイナスノイズ閾値を差し引いたベースラインよりも少ない場合、ベースラインは新しい Raw カウント値に設定されます。

可能な値は 0 から 255 です。

### 3.6.8 低ベースライン値リセット

低ベースライン値リセット パラメーターは、マイナスのノイズ閾値パラメーターと連動します。これは、ベースライン値のリセットが必要となる異常に低いサンプルの数を数えるもので、起動時に指が置かれている状態の補正をするために使用されます。

可能な値は 0～255 です。

### 3.6.9 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ON で差分カウント 75 パーセントに設定
- **ノイズ閾値**: センサーOFF で差分カウント 40 パーセントに設定
- **ベースライン更新閾値**: ノイズ閾値の 2 倍に設定
- **センサー 自動リセット**: デザインの必要条件に基づく
- **ヒステレシス**: センサーON で差分カウント 15 パーセントに設定
- **デバウンス**: デザインの用件に基づく
- **負のノイズ閾値**: ノイズ閾値
- **低ベースライン リセット**: 10 に設定

## 3.7 CSD ユーザー モジュール低レベル パラメーター

CSD ユーザー モジュールには、高レベル パラメーターに加えて、複数の低レベル パラメーターがあります。これらのパラメーターは CSD 検知方式によって異なり、どのように Raw カウントデータをセンサーから取得するのかが決まります。

### 3.7.1 スキャン速度

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。すばやいスキャン速度は早い反応時間を得られますが、遅いスキャン速度には次の利点があります。

- 優れた S/N 比
- 電源と温度変化に対するより良いイミュニティ
- システム割り込みレイテンシの必要性が減少し、より長い割り込みを処理可能

スキャン速度と分解能性能については表 3-1 と表 3-2 を参照してください。可能な値は、超高速、高速、通常、および低速

### 3.7.2 分解能

このパラメーターはスキャン分解能をビット数で判断します。ビット数が N の場合、スキャン分解能の最大 Raw カウントは $2^N$–1 です。分解能を上げると感度と S/N 比が向上しますが、スキャン時間が減少します。スキャン速度と分解能性能については表 3-1 と表 3-2 を参照してください。可能な値は 9～16 です。

表 3-1. 24MHz IMO 処理のスキャン速度および分解能、クロックプリスケーラ (PRS16 コンフィグレーション)なし

| 分解能 (ビット) | スキャン速度 (µs) | | | |
|---|---|---|---|---|
| | 超高速 | 高速 | 通常 | 低速 |
| 9 | 75 | 110 | 170 | 300 |
| 10 | 110 | 170 | 300 | 510 |
| 11 | 170 | 300 | 510 | 1010 |
| 12 | 300 | 510 | 1010 | 2030 |
| 13 | 510 | 1010 | 2030 | 4060 |
| 14 | 850 | 1690 | 3380 | 6760 |
| 15 | 1520 | 3040 | 6080 | 12200 |
| 16 | 2880 | 5720 | 11500 | 23200 |

表 3-2. 24MHz IMO 処理のスキャン速度および分解能、クロック プリスケーラ (PRS8 コンフィグレーション) あり

| 分解能 (ビット) | スキャン速度 (µs) | | | |
|---|---|---|---|---|
| | 超高速 | 高速 | 通常 | 低速 |
| 9 | 60 | 85 | 150 | 255 |
| 10 | 85 | 150 | 255 | 510 |
| 11 | 150 | 255 | 510 | 1020 |
| 12 | 255 | 510 | 1020 | 2040 |
| 13 | 510 | 1020 | 2010 | 4080 |
| 14 | 845 | 1700 | 3380 | 6760 |
| 15 | 1530 | 3060 | 6120 | 12100 |
| 16 | 2880 | 5800 | 11500 | 23000 |

### 3.7.3 リファレンス

コンパレータ入力への電圧リファレンスが CSD の適切な動作のために必要です。

可能な値は以下の通りです。

- VBG: 固定バンドギャップ リファレンス電圧から生成した 1.3V の内部電圧リファレンスです。
- ASE11: PWM から生成した内部可変電圧リファレンスです。
- AnalogColumn_InputSelect_1: 抵抗分周器ネットワークまたは外部フィルターPWM／PRSPWM 信号などの外部電圧リファレンス。このリファレンスは CapSense コントローラー ピンに接続でき、コンパレータ リファレンス元として使用できます。

### 3.7.4 リファレンス値

このパラメーターの作用は、リファレンスパラメーターの選択によります。VBG または AnalogColumn_InputSelect_1 からのリファレンスを得る場合、この値はまったく影響しません。ASE11 からリファレンスを得る場合、このパラメーターがリファレンス値を設定します。

可能な値は 0 (最小値 $V_{DD}$ の¼)～8 (最大値 $V_{DD}$ の¾)。

### 3.7.5 プリスケーラ期間

センサー静電容量は、スキャン時に 2 つの電位間で継続的に切り替わります。このパラメーターは、プリスケーラ ピリオド レジスタを設定し、プリチャージ スイッチ出力周波数を決定します。このパラメーターは、クロックプリスケーラコンフィギュレーションをもつ CSD だけに使用できます。

可能な値は 1～255 です。

### 3.7.6 シールド電極出力

シールド電極出力は、耐水性を必要とするアプリケーションを実現します。シールド電極信号源は、空いているデジタル Row バス (Row_0_Output_1～Row_0_Output_3) のいずれかから選択できます。各 Row 出力は、3 つのピンのいずれかにルーティングできます。Row LUT 関数を A に設定してください。

可能な値は、Enabled と Disabled です。

## 3.8 CSDADC ユーザー モジュールのコンフィギュレーション

CSDADC は静電容量検知を ADC 機能と組み合わせています。CSD と ADC で共通のモジュールを再利用することで、コード容量を節約します。静電容量式検知と ADC 機能の両方を必要する場合は CSDADC を使用します。このうち片方だけが必要なアプリケーションの場合は、CSD ユーザー モジュール、ADC8 または ADC10 ユーザー モジュールを使用します。

CSDADC ユーザー モジュールには、4 つの選択可能なクロック コンフィギュレーションがあります。これらのコンフィギュレーションは CSD フロントエンドのスイッチト キャパシタ回路に異なる信号ソースを使用します。

CSDADC ユーザー モジュールを PSoC Designer プロジェクトに初めて導入するときに、クロック コンフィギュレーションを選択する必要があります。この選択は、CSDADC ユーザー モジュールを右クリックし、「**User Module Selection Options**」(ユーザー モジュール選択オプション) を選ぶと変更できます。

### 3.8.1 PRS16 クロック源が備わった CSDADC

このコンフィグレーションでは 16 ビットの疑似ランダム シーケンス(PRS) が CSD フロントエンドのスイッチト キャパシタ回路の信号ソースとして使用されます。システム クロック (IMO) が PRS のクロック源です。PRS はクロックの周波数スペクトラムを拡散し、良好なノイズ耐性を提供します。このコンフィグレーションは 3 つのデジタル ブロックを必要とします。

### 3.8.2 PRS8 クロックソースが備わった CSDADC

このコンフィグレーションでは、8 ビットの PRS が CSD フロントエンドのスイッチト キャパシタ回路の信号ソースとして使用されます。システム クロック (IMO) が PRS のクロック源です。PRS はクロックの周波数スペクトラムを拡散してノイズ耐性を提供します。クロック プリスケーラ コンフィギュレーションなしの CSD 内で使用される 16 ビット PRS は、より良好なノイズ耐性を提供します。このコンフィグレーションは 3 つのデジタル ブロックを必要とします。

これはより低いスイッチング周波数を必要とするため、高い $C_P$ 値の環境で動作させる場合は、このコンフィグレーションを使用します。$C_P$ とスイッチング周波数の関係は「ユーザー モジュールによる CapSense の性能のチューニング」で説明されています。

### 3.8.3 PWM8 クロックソースが備わった CSDADC

このコンフィギュレーションでは、IMO が調節可能なプリスケーラによって分周され、CSD フロントエンドのスイッチト キャパシタ回路に使用されます。このコンフィギュレーションでは、動作周波数でのノイズおよびその高調波の感度がより高くなります。このコンフィギュレーションは 2 つのデジタルブロックを必要とします。

高抵抗素材 (ITO など) を通しての検知、または放出問題のために低い動作周波数が必要な時にこのコンフィギュレーションが適しています。

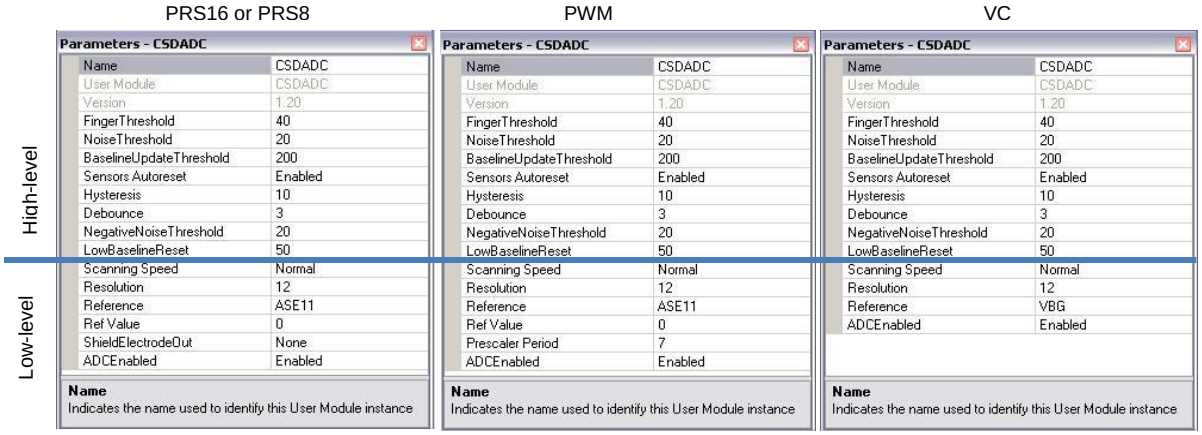
### 3.8.4 VC2 クロックソースが備わった CSDADC

このコンフィギュレーションではシステムクロックが調節可能な分周器 VC2 によって分周され、CSD フロントエンドのスイッチトキャパシタ回路に使用されます。このコンフィギュレーションでは、動作周波数でのノイズとその高調波の感度がより高くなります。このコンフィギュレーションは 1 つのデジタル ブロックを必要とします。

高抵抗素材 (ITO など) を通しての検知、または放出問題のために低い動作周波数が必要な時にこのコンフィギュレーションが適しています。

## 3.9 CSDADC ユーザー モジュール パラメーター

図 3-7. PSoC Designer - CSDADC パラメーター ウィンドウ




## 3.10 低レベル パラメーター

### 3.10.1 スキャン速度

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。より速いスキャン速度は早い反応時間を実現しますが、遅いスキャンス速度は次の利点が得られます。

- 優れた S/N 比
- 電源と温度変化に対するより良いイミュニティ
- システム割り込みレイテンシの必要性が減少し、より長い割り込みを処理可能

スキャン速度と分解能性能については表 3-1 と表 3-2 を参照してください。可能な値は、超高速、高速、通常、および低速です。

### 3.10.2 分解能

このパラメーターはスキャン分解能をビット数で判断します。ビット数が N の場合、スキャン分解能の最大 Raw カウントは $2^N$–1 です。分解能を上げると感度と S/N 比が向上しますが、スキャン時間が減少します。スキャン速度と分解能性能については表 3-1 と表 3-2 を参照してください。

可能な値は 9～16 です。

### 3.10.3 リファレンス

コンパレータ入力への電圧リファレンスが CSD の適切な動作のために必要です。

可能な値は以下の通りです。

- VBG: 固定バンドギャップ リファレンス電圧から生成した 1.3V の内部電圧リファレンスです。

- ASE11: PWM から生成した内部可変電圧リファレンスです。
- AnalogColumn_InputSelect_1: 抵抗分周器ネットワークまたは外部フィルターPWM／PRSPWM 信号などの外部電圧リファレンス。このリファレンスは CapSense コントローラー ピンに接続でき、コンパレータ リファレンス源として使用できます。

### 3.10.4 リファレンス値

リファレンス値の影響はリファレンスパラメーターによって決まります。リファレンスが ASE11 に設定されている場合、リファレンス値がリファレンス値を決定します。

可能な値は 0 (最小値 $V_{DD}$ の¼) ～8 (最大値 $V_{DD}$ の¾)。

リファレンス値が高くなると検出感度が下がりますが、シールド電極の影響が大きくなります。センサーに静電容量の顕著な差がある設計 (異なる面積のセンサーなど) では、API 関数を使い、より大きい静電容量を持つセンサーに高いリファレンス値を設定して、Raw カウントのバランスを取ることができます。

リファレンスが VBG または AnalogColumn_InputSelect_1 に設定されている場合、リファレンス値は何の影響も与えません。リファレンス値は VC2 クロック源コンフィギュレーションを用いた CSDADC では使用できません。

### 3.10.5 プリスケーラ期間

プリスケーラ期間は、プリチャージ スイッチ出力周波数を決定します。

可能な値は 1～255 です。最大の信号対ノイズ比 (1、3、7、15、31、63、127 または 255)を得るための推奨値は $2^N - 1$ です。その他の値を使うと、特に低い分解能と高いスキャン速度の場合に、ノイズが増えます。

プリスケーラ期間は、プリスケーラを使用したコンフィギュレーションにのみ使用できます。

### 3.10.6 シールド電極出力

シールド電極信号源は、空いているデジタル Row バス (Row_0_Output_1～Row_0_Output_3) のいずれかから選択できます。各 Row 出力は、3 つのピンのいずれかにルーティングできます。Row LUT 関数を A に設定してください。

可能な値は、Enabled と Disabled です。

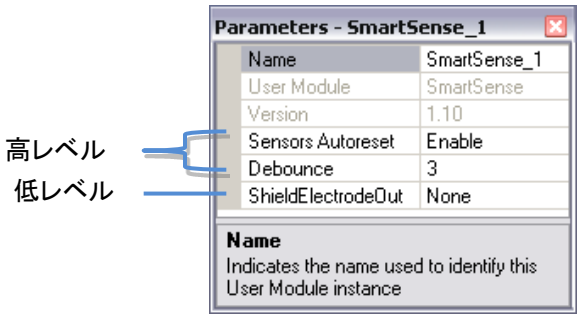
このパラメーターは、PRS8／PSR16 クロック源コンフィギュレーションを持つ CSDADC でのみ使用できます。PWM8 クロック源コンフィギュレーションを持つ CSDADC で、シールド極性信号は恒久的に Row_0_Output_0 に接続されています。

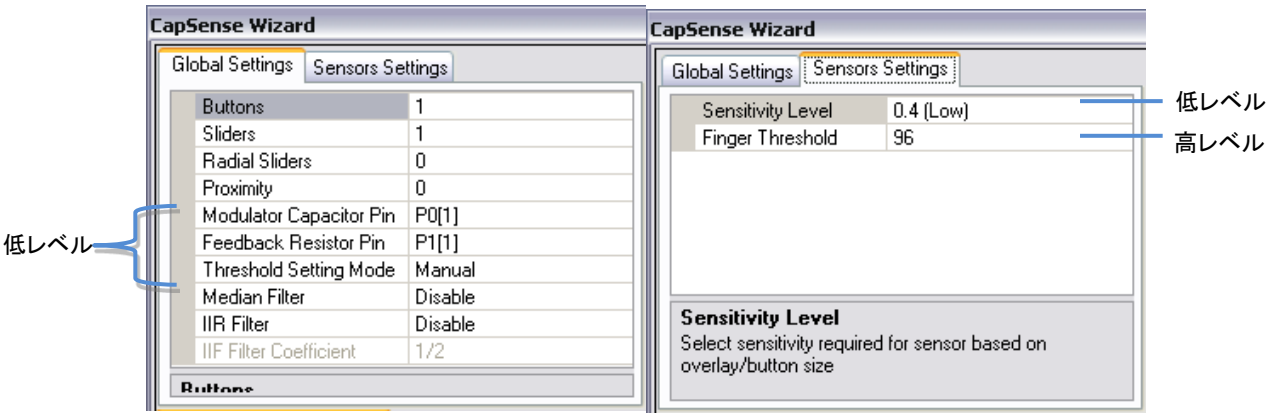
### 3.10.7 ADC イネーブル

このパラメーターを Enabled に設定すると、ADC ルーチンが互換コードに含められ、ユーザーのコードから呼び出すことができるようになります。このパラメーターが Disabled に設定されていると、ADC ルーチンは含まれません。設計で ADC が不要になったとき、このパラメーターを Disabled に設定するとフラッシュメモリに保存する上で便利なことがあります。このパラメーターが Disabled に設定されていると、CSDADC ユーザー モジュールが CSD ユーザー モジュールと同様に動作します。

## 3.11 SmartSense ユーザー モジュール パラメーター

図 3-8. PSoC デザイナ SmartSense パラメーター






## 3.12 低レベル パラメーター

### 3.12.1 シールド電極出力

シールド電極は寄生容量を削減するために使用します。シールド電極信号は、空いているデジタル Row バスのいずれかにルーティングするように選択できます (Row_0_Output_1 から Row_0_Output_3)。その後、各 Row 出力は使用可能なピンのいずれかにルーティングできます。Row LUT 関数を A に設定してください。

### 3.12.2 変調器キャパシタ ピン

このパラメーターは、外付け変調コンデンサ ($C_{MOD}$) を接続するピンを設定します。利用できるピン P0[1]、P0[3] から選択します。

### 3.12.3 フィードバック抵抗ピン

このパラメーターは、外付けフィードバック抵抗 ($R_b$) を接続するピンを設定します。次のピンから選択します: P1[1]、P1[5]、P3[1]。デバイス パッケージによっては使用できないピンもあります。プログラミングの問題を避けるために、P1[5]または P3[1] を使用してください。

### 3.12.4 閾値設定モード

このパラメーターを使用して、自動またはマニュアルいずれかの閾値設定を選択します。自動閾値モードは、システム内でみられるノイズに基づいて指の閾値を動的に調整するアルゴリズムを使用します。手動閾モードでは、固定の指閾値を与えることができる手動閾設定モードの用途と指の閾値の決定方法の詳細は「SmartSense ユーザー モジュールのコンフィギュレーション」節で説明されます。

### 3.12.5 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。低い感度 (0.1pF) により、センサー信号が強くなります。オーバーレイの厚いデザインでは、正い実行のためには強いセンサー信号が必要です。高 (0.1pF)、中 (0.2pF) と低 (0.4pF) の感度が選択可能です。

センサーから強い信号 (高感度) を作り出すには、SmartSense UM のセンサー スキャン時間を延長します。つまり、センサーを感度 0.1pF (高) に設定すると、感度レベルを 0.2 pF (中高) に設定したセンサーに比べて、スキャン時間が長くなります。

### 3.12.6 指閾値

このパラメーターは、閾値設定モードがマニュアルモードに設定されている場合のみに使用できます。SmartSense_baSnSSignal[] アレイ内に保存されたセンサー信号の 80 パーセントの値に設定することが推奨されています。このアレイは $I^2C$ または UART 通信プロトコルを使用して簡単に監視することができます。

# 4. ユーザー モジュールによる CapSense の性能のチューニング

最適ユーザー モジュール パラメーター設定は、ボードレイアウト、ボタン寸法、オーバレイ材料やアプリケーションの必要条件によって異なります。これらの要因については、「設計の注意事項」で説明します。チューニングとは、安定で信頼できるセンサー操作のための最適パラメーター設定を特定するプロセスです。

## 4.1 一般的な注意事項

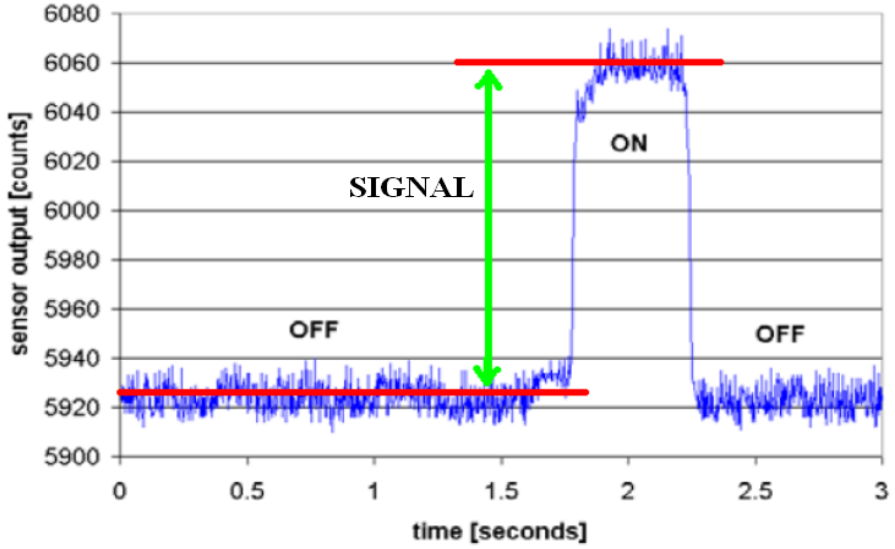
### 4.1.1 信号、ノイズ、および SNR (信号対ノイズ比)

適切にチューニングされた CapSense システムによって、オンとオフのセンサー状態が確実に識別されます。このレベルの性能を得るには、CapSense 信号は CapSense ノイズよりかなり大きくする必要があります。信号とノイズを比較する測定が、信号対ノイズ比 (S/N 比) です。CapSense の S/N 比 についての説明を行う前に、接触検知のコンテキストにおける信号とノイズの定義づけをまず行います。

#### 4.1.1.1 CapSense 信号

CapSense 信号とは、図 4-1 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、センサー容量をトラック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、指によるカウントの変化を追跡するため、信号 = 6060 – 5925 = 135 カウントとなります。

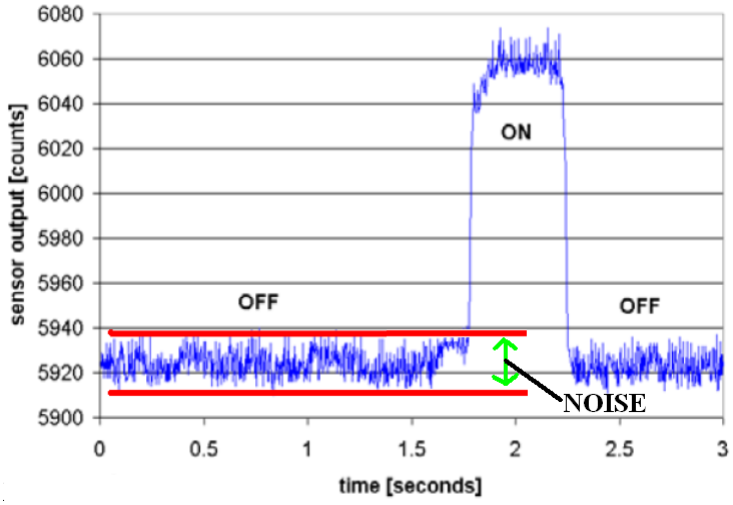
図 4-1. CapSense 信号の例



#### 4.1.1.2 *CapSense のノイズ*

CapSense ノイズは、図 4-2 に示されているように、指の接触がない時のセンサー反応におけるピーク ツー ピークの変化です。この例では、指を置かない場合の出力波形は、最小で 5912 カウント、最大で 5938 カウントに制限されています。ノイズとはこの波形の最小値と最大値の差分であるため、ノイズ = 5938 – 5912 = 26 カウントとなります。

図 4-2. CapSense ノイズの例




#### *4.1.1.3 CapSense S/N 比*

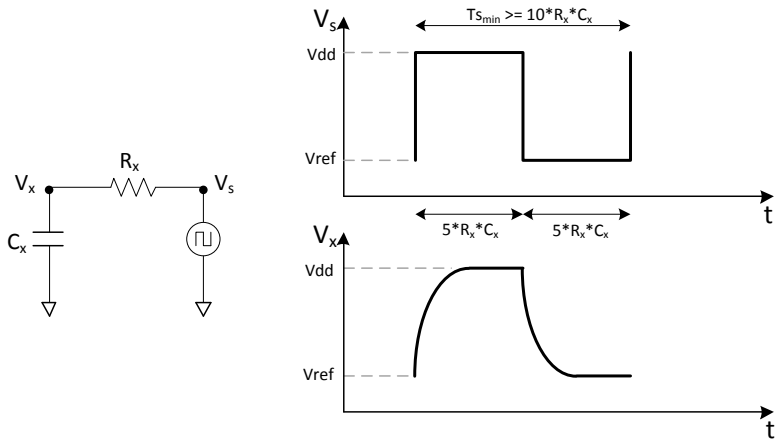
CapSense S/N 比 とは信号とノイズの単純比です。本例で説明を続けると、信号が 135 カウントでノイズが 26 カウントの場合、S/N 比 は 135:26 で、S/N 比 は 5.2:1 に約分されます。推奨の CapSense 最小 S/N 比は 5:1 (信号値がノイズ値の 5 倍大きいです) です。一般的に、フィルターはノイズを減らすためファームウェアに実装されます。詳細は「ソフトウェアのフィルター処理」を参照してください。

### 4.1.2 充電／放電速度

チューニングプロセスで最大感度を実現するには、各サイクルでセンサー静電容量が完全に充電および放電される必要があります。充電／放電パスは、ユーザー モジュールでクロック パラメーターによって設定された比率で 2 つの状態を切り替えます。

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。電荷転送の変化比率は、図 4-3 に示されるように、センサー キャパシタと直列抵抗による RC 時定数によって決まります。

図 4-3. 充電／放電波形



充電／放電比率を、RC 時定数に合うレベルに設定するように注意してください。大体の目安として、各転送には 5RC の期間を合わせて、1 期間 (充電 1 回、放電 1 回) 2 回の転送とします。最小時間と最大周波数の式を次に示します。最短期間と最大周波数の式は以下のとおりです。

$$Ts_{min} = 10 \times R_X C_X \quad \text{式 5}$$

$$fs_{max} = \frac{1}{10 \times R_X C_X} \quad \text{式 6}$$

例えば、直列抵抗に、外部抵抗 560Ω と内部抵抗 800Ω がつながれていて、センサー容量が標準的であると仮定すれば:

$R_X$ = 1.4kΩ

$C_X$= 24pF

この例における時定数の値とフロントエンド スイッチ周波数は次のようになります:

$Ts_{min}$ = 0.34µs

$fs_{max}$ = 3MHz

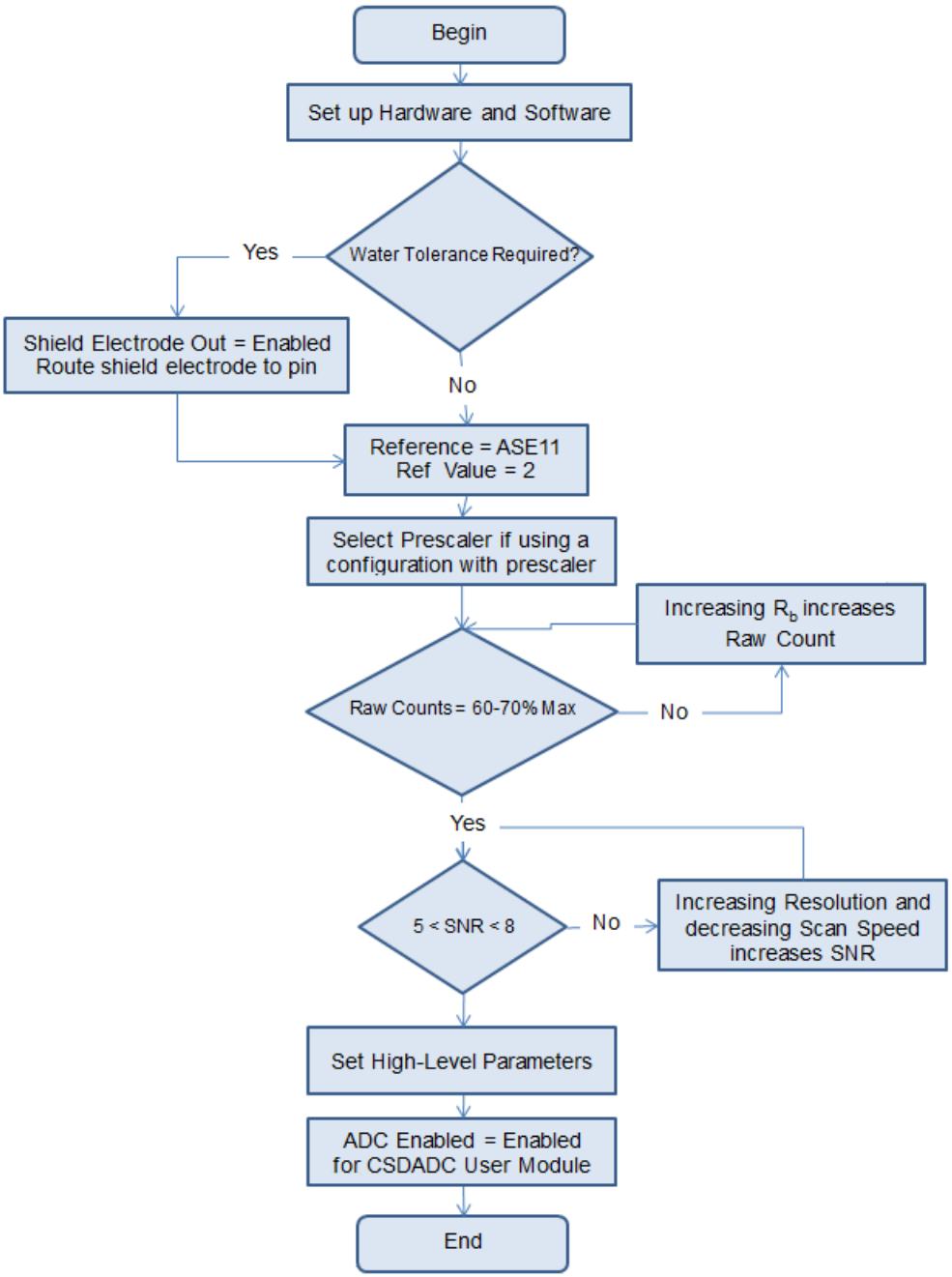
### 4.1.3 ベースライン値更新閾値の検証の重要性

温度と湿度はどちらも、平均カウント数の経時的変動の原因となります。ベースライン値は CapSense 測定の参照カウント レベルであり、環境への影響補正に重要な役割を果たします。指あり状態と指なし状態といった高レベル決定は、ベースライン値で設定される参照レベルに基づきます。各センサーにはそれに関連付けられた一意の寄生容量があるため、各容量センサーには独自のベースライン値があります。

ベースライン値は、ベースライン値更新閾値パラメーターで設定された速度でカウントの変化を追跡します。更新速度が目的に合致していることを確認します。更新速度が速すぎると、ベースライン値は指による全ての変化を補正するため、動いている指は検出されません。更新速度が遅すぎると、比較的ゆっくりの環境変化が指と間違えられる可能性があります。開発時に、ベースライン更新閾値設定を確認する必要があります。

## 4.2 CSD および CSDADC ユーザー モジュールのチューニング

図 4-4. CSD および CSDADC ユーザー モジュールのチューニング



### 4.2.1 チューニングを行うためのハードウェアおよびソフトウェア セットアップ

チューニングを始める前に、「概要」に記載されるように、いくつかのハードウェアおよびソフトウェア ツールが必要となります。チューニングにはセンサーの Raw カウント、ベースライン、差分カウントを通信インターフェイスを通じてモニタリングすることが必要です。PC と通信する $I^2C$ 通信プロトコルのコードの例はこちらからダウンロードできます。例をダウンロードした後:

1. PSoC Designer を使ってコードの例を開きます。Workspace Explorer 内で「**CSD User Module**」(CSD ユーザー モジュール) を右クリックし、「**CSD Wizard**」(CSD ウィザード) を選択します。ピンに割り当てられた 4 つの CapSense ボタンが表示されます。アプリケーション要件に従ってボタンの数字およびピンの割り当てを変更します。
2. 必要に応じて、変調コンデンサ ピンおよびフィードバック 抵抗ピンを変更します。
3. プロジェクトのジェネレーションとビルドを行います。
4. hex ファイルで CapSense コントローラーのプログラムを行います。
5. Raw カウント、ベースライン、差分カウントのモニタリング方法についての説明は、コードの例に配置される「CapSense データ モニタリングのための USB-I2C ツール使用に関する説明」を参照してください。

### 4.2.2 プリスケーラの選択

次の手順に従ってプリスケーラの値を選択します。

1. LCR メータまたは CapSense コントローラーを用いてセンサー寄生容量の最大値を測定します。LCR メータを使用する場合、ステップ 5 に進んでください。
2. クロックプリスケーラと以下のパラメーターが備わった CSD ユーザー モジュール:
   a. スキャン速度 = 通常
   b. 分解能 = 16
   c. レファレンス = VBG
   d. プリスケーラピリオド= 15 (SysClk = 24 MHz)、7 (SysClk = 12 MHz)、または 3 (SysClk = 6 MHz)
   e. ShieldElectrodeOut = アプリケーションにより、有効または無効
3. プロジェクトをビルドし、CapSense コントローラーにダウンロードします。
4. すべてのセンサー (スライダー セグメントを含む) から Raw カウントのみをモニタリングし、最も高い Raw カウント値を書き留めます。
5. 式 7 を使用して最大寄生静電容量を計算します。

$$C_X = \frac{\text{Highest Raw Coun}}{88.85\text{e}12\times(V_{DD}-1.3)} \qquad \text{式 7}$$

6. 式 8 を使用して $fs_{max}$ を計算します。

$$fs_{max} = \frac{1}{10\times R_X C_X} \qquad \text{式 8}$$

7. クロックコンフィギュレーションに基づいてプリスケーラ値を計算します。
8. クロックプリスケーラコンフィギュレーションを持つ CSD:

$$\text{Prescaler} \geq \frac{\text{SysClk}}{fs_{max}\times 2} \qquad \text{式 9}$$

9. PWM8 クロック源コンフィギュレーションを持つ CSDADC:

$$\text{Prescaler} \geq \frac{\text{SysClk}}{fs_{max}} \qquad \text{式 10}$$

10. より良好な S/N 比性能を得るには、計算したプリスケーラを $2^n - 1$ の値 (1、3、7、15、31 など) に切り上げます。

### 4.2.3 $R_b$ に応じて Raw カウント範囲を設定

ブリード抵抗の値 ($R_b$) は CapSense センサーの感度を決定します。感度とは、接触と非接触状態の違いを測定した差分カウントです。$R_b$ が上がると、感度と非接触状態での Raw カウント値が上げます。$R_b$ はまた、CapSense センサーによって測定できる最大静電容量値を下げます。

図 4-5. CSD 測定範囲

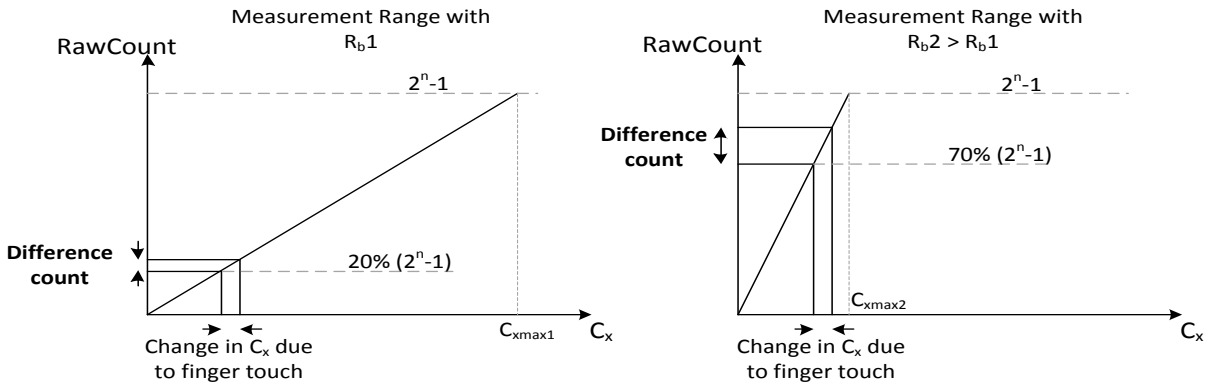



図 4-5 の 2 つの Raw カウント測定範囲のグラフを考察します。左の図 ($R_b$ = $R_b1$) では、$C_{xmax1}$ までシステムが静電容量を測定できます。Raw カウントは最大値の 20 パーセントとなっています。指の接触により導き出される差分カウントはとても小さく、感度は低くなります。右の図ではブリード抵抗が大きくなっています ($R_b2 > R_b1$)。Raw カウントは最大値の 70 パーセントとなります。この場合の差分カウントは先のケースに比べて大幅に大きいため感度が上がります。

$R_b$ の一般的な値は、500Ω と 10kΩ です。最適な抵抗値を見つける方法:

1. まず $R_b$ に 2kΩ 抵抗を使用します。
2. すべてのセンサーについて Raw カウントをモニタリングします。
3. 選択したスキャン分解能で Raw カウントがフルスケール範囲の 70 パーセントになるまで $R_b$ を調整します。たとえば 9 ビットの分解能が選択されている場合、フルスケール範囲は 511 カウントとなります。Raw データが 358 カウントになるまで $R_b$ を調整します。

複数センサーを使用する設計では、すべての Raw カウントを 70 パーセントきっちりに合わせることは難しい場合があります。このような場合、すべての Raw カウントを最大値の 60～70 パーセントの間に合わせます。

### 4.2.4 高レベル パラメーターを設定する

「高レベル パラメーターの推奨事項」で紹介される推奨に従ってください。

## 4.3 SmartSense ユーザー モジュールのコンフィギュレーション

ここでは、一般的な CapSense アプリケーションにおいて、SmartSense ユーザー モジュールをコンフィギュレーションする基本的なガイドラインを説明します。

1. ターゲット ボードを準備します。ターゲット アプリケーションの プリント基板 を組み立て、オーバーレイをそれに固定します。オーバーレイに付着する非導電の接着剤または特殊接着テープを使用します。感度が著しく下がり、接触時にエアギャップのシフトが発生して誤ったボタン トリガが発生するため、プリント基板 とオーバーレイ間に空気が入らないように注意してください。

2. リアルタイムで 1 つ以上のデータシリーズを監視するために、マルチチャート (項「CapSense データ表示ツール」) などのリアルタイム データチャート作成ツールをセットアップします。このプロセスを行う間、センサーの Raw カウント (waSnsResult)、ベースライン (waSnsBaseline)、差分カウント (waSnsDiff)、シグナル (baSnsSignal)、ボタン指閾値 (baBtnFThreshold) を監視する必要があります。UART-USB ブリッジまたは I2C-USB ブリッジ ハードウェアを使用すると、USB ポートを介した CapSense コントローラーと PC の間のインターフェイスを確立できます。LCD やその他の数値ディスプレイは表示が遅く、動的なデータの変化を視覚化できないため、データのモニタには使用しないでください。

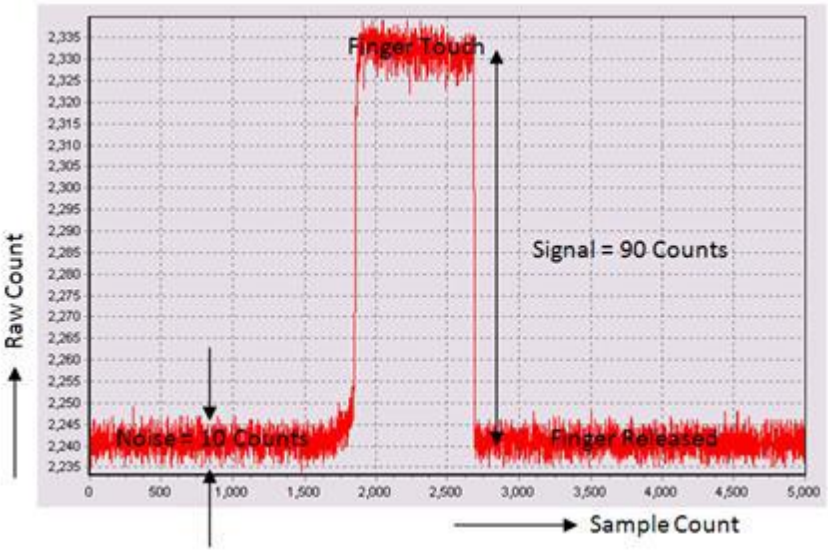
   **注 :** SmartSense と EzI2C を使用している間、P1[1]が SCL に使用されるため、P1[1]は $R_b$ に対応しません。

3. 最初に行うデフォルトコンフィギュレーションは、次の通りです。

- IMO = 24 MHz、CPU_Clock = SysClk/1 に設定します。
- IMO = 24 MHz に SmartSense ユーザー モジュールコンフィギュレーションを設定します。
- 「**Sensor Autoreset** (センサーの自動リセット)」を無効にします。「**Debounce** (デバウンス)」を 3 に、「**Shield Electrode Out** (シールド電極出力)」を None に設定します。
- CapSense ウィザードの「Global Settings (グローバル設定)」で、「Button (ボタン)」カテゴリ内に 1 つのセンサーを割り当てます。変調コンデンサおよびフィードバック抵抗 ピンも同様に選択します。「**Threshold Setting Mode**」 (閾値設定モード) を Manual (手動) に、メジアン フィルターと IIR フィルターを無効に設定します。基板上で $R_b$ 値が 15K、$C_{MOD}$ が 10nF になっていることを確認します。
- CapSense ウィザードの Sensor Settings (センサーの設定) タブから、「**Sensitivity Level**」(感度レベル) を Low (低) に、「**Finger Threshold**」 (指の閾値) を 96 に設定します。

4. ユーザー モジュール データシート内のサンプル ファームウェア コードを使って、プロジェクトの生成／ビルドを行います。
5. センサーの Raw カウントをモニタリングし、S/N 比 を計算します。図 4-6 は監視されたセンサーの Raw カウントを示しています。CapSense ベスト プラクティスによると、デザインを堅牢にするように、SNR を 5 以上にしなければなりません。この場合、監視される SNR は 9 です。SNR が 5 未満未満の場合、ステップ 7 での SNR 向上技術を参照してください。

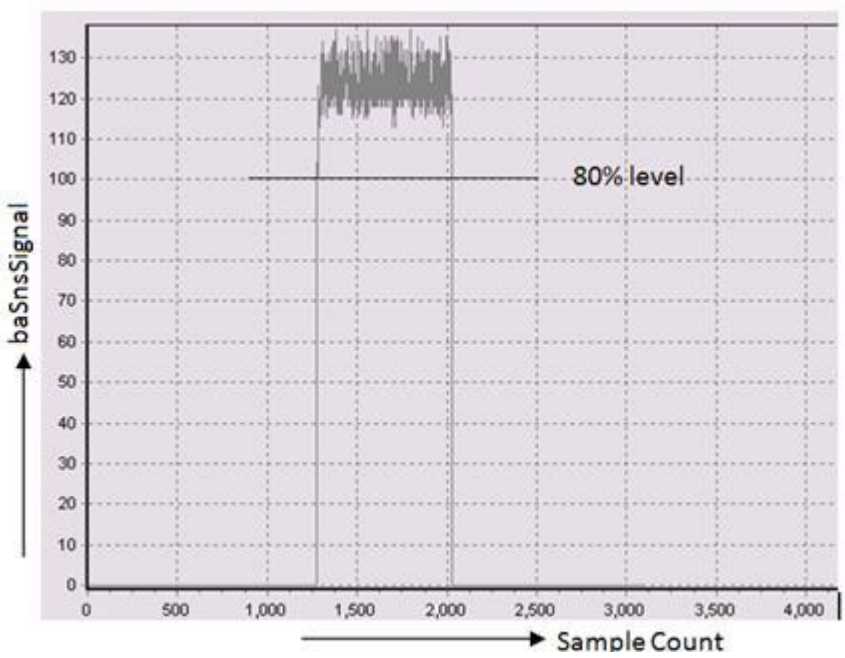
図 4-6. モニタリングされたセンサーの Raw カウント




6. アプリケーションが指の閾値の精密な制御を必要とされない場合、あるいは自動閾値設定モードが選択されている場合、この手順を飛ばすことができます。自動閾値モードは、システム内でみられるノイズに基づいて指の閾値を動的に調整するアルゴリズムを使用します。次の状況の場合は、自動閾値モードではなくマニュアルモードを選択してください。
   - RAM／ROM 実装面積が、サポートするセンサーの数と比べて小さい場合。
   - センサーの指の閾値を精密に調整することが必要な設計。
   - ノイズの多い状況下で動作する必要のある設計。

手動設定モードが使用される場合、チューニング プロセスを完了させるには指の閾値を設定する必要があります。センサーの信号 (baSnsSignal) をモニタリングします。最大値の 80 パーセントを決定し、これが指の閾値となります。図 4-7 に示される例では、ほぼ 100 カウントとなっています。

図 4-7. センサーの信号 (baSnsSignal) のモニタリング




次に CapSense ウィザードを開き、「Sensor Settings」 (センサーの設定) タブより、指の閾値を計算値に設定します (図 4-7 の例では 100 です)。

7. これで SmartSense センサーのチューニングプロセスが完了しました。しかしながら、厚いオーバーレイを使用している時など、ステップ 5 で得た S/N 比が 5: 1 より小さい場合があります。このような場合は、次の推奨事項に従います。
   - 感度レベルを「Low (低)」から「Medium (中)」に変更します。これでも S/N 比が 5: 1 に達しない時は、レベルを「High (高)」に変えます。
   - 時によっては、システムでみられる過剰なノイズにより 5: 1 の要件を満たせないことがあります。こういった状況については、ソフトウェア フィルターを使用します。係数 1/2 の IIR フィルターで始めます。これでもノイズが減らない場合は、係数 1/4 の IIR に変更します。

# 5. 設計の注意事項

静電容量タッチ センス技術をアプリケーションで設計する際に、CapSense デバイスがより大きなフレームワーク内に存在していることを 覚えておくことが必須となります。プリント基板レイアウトからエンドアプリ環境でのユーザー インターフェースにいたるまで、すべての設計レベルにおいて詳細に注意を払うことで、堅牢で信頼性の高いシステム性能を実現させることが可能です。さらに詳しい情報については、「CapSense 入門」を参照してください。

## 5.1 オーバーレイの選択

「CapSense の原理」では、指の静電容量について次のとおり式 1 が紹介されています。

$$C_F = \frac{\varepsilon_0 \varepsilon_r A}{D}$$

ここで、

$\varepsilon_0$ = 真空の誘電率

$\varepsilon_r$ = 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

CapSense 信号の強度を上げるには、誘電定数の高いオーバーレイ素材を選び、オーバーレイの厚みを抑え、ボタンの直径を大きくします。

表 5-1. オーバーレイ素材の絶縁耐力

| 素材 | 絶縁破壊電圧 (V/mm) | 最小値 12 kV でのオーバーレイの厚さ (mm) |
|---|---|---|
| 空気 | 1200–2800 | 10 |
| 木材 (乾燥) | 3900 | 3 |
| ガラス (一般的) | 7900 | 1.5 |
| ガラス – ホウケイ酸ガラス (Pyrex®) | 13,000 | 0.9 |
| PMMA プラスチック (プレキシグラス®) | 13,000 | 0.9 |
| ABS | 16,000 | 0.8 |
| ポリカーボネート (Lexan®) | 16,000 | 0.8 |
| フォーマイカ (高圧メラミン化粧板) | 18,000 | 0.7 |
| FR-4 | 28,000 | 0.4 |
| PET フィルム (Mylar®) | 280,000 | 0.04 |
| ポリイミド フィルム (Kapton®) | 290,000 | 0.04 |

導電性の素材は、電場パターンとの干渉を起こしてしまうため、オーバーレイとしては使用できません。そのため、オーバーレイには金属粒子を含む塗料は使用できません。

オーバーレイを CapSense プリント基板に接合するために、接着剤を使用します。3M™の 200MP と呼ばれる透明なアクリル系接着剤は CapSense アプリケーションでの使用に適しています。この特殊な接着剤は裏に紙が付いたテープロール形状で販売されています (3M™商品番号 467MP および 468MP)。

## 5.2 ESD 保護

安定した ESD トレランスは、慎重なシステム設計から生まれた当然の結果です。製品で静電気がどれだけ発生するかを考慮することで、CapSense コントローラーにダメージを与えることなく、18kV までの静電気に耐えることが可能です。

CapSense コントローラー ピンは直流 2kV 放電現象に耐えることが可能です。ほとんどの場合、オーバーレイの素材がコントローラー ピンへの十分な ESD 保護を提供します。表 5-1 に、CapSense センサー＝を 12kV 放電から保護するために必要なさまざまなオーバーレイの IEC 61000-4-2 で指定された素材の厚さ一覧を示します。オーバーレイの素材が十分な ESD 保護を提供しない場合は、対応策は次の順番で適用します: 予防、リダイレクト、クランプ。

### 5.2.1 予防

接触面のすべてのパスが、潜在的な高電圧接触よりも高い絶縁破壊電圧を備えていることを確認してください。またシステム内の、CapSense コントローラーと ESD 発生源となり得る部分間の距離がいつでも適切であるように設計します。適切な距離を維持できない場合は、絶縁破壊電圧の高い素材による保護レイヤを ESD ソースと CapSense コントローラーの間に設けてください。5mm 厚さの Kapton®テープによる層は 18 kV に耐久します。

### 5.2.2 転向

基板にはコンポーネントの密度が高い場合は、放電現象を避けることは難しいかもしれません。この場合、放電の発生源を制御することにより、CapSense コントローラーを保護することができます。標準的な手順としては、シャーシの接地に接続された回路ボードの外周部に保護リングを配置します。「プリント基板レイアウトのガイドライン」で推奨するように、ボタンまたはスライダー センサーの周囲にハッチチング グランド面を施すことによってセンサーと CapSense コントローラーへの ESD の影響を回避することができます。

### 5.2.3 クランプ

CapSense センサーは意図的にタッチ方面に近接して配置されているため、放電パスのリダイレクトは実践的でないかもしれません。この場合、直列 抵抗を含む特殊用途の ESD 保護デバイスが適切かもしえません。

推奨する直列抵抗は 560Ω です。

より効果的な方法は、専用の ESD 保護デバイスを脆弱な配線上に置くことです。CapSense 用の ESD 保護デバイスは低静電容量である必要があります。表 5-2 に、CapSense コントローラー用として使用目的に推奨するデバイスの一覧を示します。

表 5-2. CapSense 用の推奨低静電容量 ESD 保護デバイス

| ESD 保護デバイス | | 入力静電容量 | リーク電流 | 接触放電の最高限度 | 空中放電の最高限度 |
|---|---|---|---|---|---|
| メーカー | 型番 | | | | |
| Littelfuse | SP723 | 5pF | 2nA | 8kV | 15kV |
| Vishay | VBUS05L1-DD1 | 0.3pF | 0.1µA< | ±15kV | ±16kV |
| NXP | NUP1301 | 0.75pF | 30nA | 8kV | 15kV |

## 5.3 EMC (電磁環境適合性) の注意点

### 5.3.1 放射性干渉

放射性電気エネルギーはシステム測定に影響を与え、さらにプロセッサ コアの動作に影響を与える可能性もあります。干渉は、CapSense センサー配線、およびその他のデジタル入力またはアナログ入力を通して、プリント基板レベルで CapSense コントローラーに入る可能性があります。RF 干渉はの影響を最小限にするためのレイアウト ガイドラインは以下を含みます:

- **グランド面**: プリント基板にグランド面を設けます。
- **直列抵抗**: CapSense コントローラー ピンから 10mm 以内に直列抵抗を配置します。
  - CapSense 入力ライン向けの推奨直列抵抗は、560Ω です。
  - $I^2C$、および SPI などの通信ライン向けの推奨直列抵抗は、330Ω です。

- **配線の長さ**: 可能な限り配線を短くします。
- **電流ループの面積**: 電流の帰路を短くします。固体を充填する代わりに、斜めの接地をセンサーと配線の 1 cm 以内に取り付け、寄生容量の影響を低減させます。
- **RF 源の位置**: LCD インバータおよびスイッチング電源 (SMPS) のようなノイズ源を CapSense 入力から隔てるために隔離手段をとります。電源のシールドは干渉を防ぐもう一つの良くある技術です。

### 5.3.2 放射妨害波

CapSense センサーからの放射エミッションを低減させるには、スイッチト キャパシタ クロックで低周波数を選択してください。このクロックは、プリスケーラ オプションを使用するファームウェアで制御しています。プリスケーラ値を増加させることにより、スイッチクロックの周波数が減少します。

### 5.3.3 伝導イミュニティおよびエミッション

他のシステムとの電線による接続を通じてシステムに入ったノイズは伝導ノイズと呼ばれます。相互接続には電源や通信ラインを含みます。CapSense コントローラーは低電力デバイスで、伝導性電磁波を回避できます。以下のガイドラインは導電性エミッションおよびイミュニティを減らす助けになります。

- データシートで推奨しているようにデカップリングコンデンサを使用する。
- システム電源への入力上に双方向性のフィルターを追加する。導電性エミッションおよびイミュニティの両方に効果的です。パイ型フィルターは、電源ノイズが高感度のパーツに影響を与えるのを防ぐことができ、また、パーツ自体のスイッチングノイズがパワープレーンにカップリング バックするのを防ぎます。
- CapSense コントローラーのプリント基板がケーブルで電源に接続されている場合は、ケーブル長を最短にして、シールドケーブルの使用を検討することが必要
- 電源または通信ラインの周辺にフェライト ビーズを置き、高周波ノイズをフィルターできます。

## 5.4 ソフトウェアのフィルター処理

ソフトウェア フィルターは、高レベルのシステム ノイズに対処する 1 つの方法です。表 5-3 は CapSense に役立つフィルターの種類の一覧です。

表 5-3. CapSense フィルターの表

| タイプ | 説明 | 応用 |
|---|---|---|
| アベレージ | 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ー (フィードバックなし) | 電源からの周期的なノイズ |
| IIR | RC フィルターに類似したステップ レスポンスを備えた、有限インパルス応答フィルター (フィードバック付き) | 高周波ホワイト ノイズ(1/f ノイズ) |
| Median (メジアン) | サイズ N 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ー | モーターおよび電源切り替えに伴って出るスパイク ノイズ |
| Jitter (ジッタ) | 前の入力に基づいて、電流の入力を制限する非線形フィルター | 厚いオーバーレイからのノイズ (SNR < 5: 1); 特にスライダーのセントロイド データに役立つ |
| イベント ベース | センサー データで観察されたパターンへの事前定義された応答を発生させる非線形フィルター | 一般に、CapSense データ送信をブロックするための非接触イベント中に使用される |
| ルール ベース | センサー データで観察されたパターンへの事前定義された応答を発生させる非線形フィルター | タッチ表面の通常操作中によく使用され、偶発的なマルチボタン選択などの特殊なシナリオに対応します |

上記のフィルターをすべて使用するコード例は、こちらからダウンロードできます。

表 5-4 に、さまざまなソフトウェア フィルターの RAM とフラッシュ要件を示します。各フィルター種類に必要なフラッシュ量は、コンピュータの性能に依存しています。こちらに一覧表示されている要件は、ImageCraft コンパイラおよび ImageCraft Pro コンパイラの両方を対象としています。

表 5-4. RAM およびフラッシュの要件

| フィルターの種類 | フィルターの順序 | RAM (センサー毎のバイト) | Flash (バイト) ImageCraft コンパイラ | Flash (バイト) ImageCraft Pro コンパイラ |
|---|---|---|---|---|
| アベレージ | 2-8 | 6 | 675 | 665 |
| IIR | 1 | 2 | 429 | 412 |
| | 2 | 6 | 767 | 622 |
| Median (メジアン) | 3 | 6 | 516 | 450 |
| | 5 | 10 | 516 | 450 |
| Raw カウントのジッタ フィルター | 該当なし | 2 | 277 | 250 |
| スライダー セントロイドのジッタ フィルター | 該当なし | 2 | 131 | 109 |

上記のフィルターをすべて使用するコード例は、こちらからダウンロードできます。

## 5.5 消費電力

### 5.5.1 システム デザインの推奨事項

多くのデザインで、消費電力を最小限に抑えることは重要な目標です。ご使用の CapSense 静電容量タッチスクリーン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。

- 低電力用に GPIO 駆動モードに設定する
- 高出力ブロックを停止させる
- 低電力用に CPU の速度を最適化する
- 低い $V_{DD}$ で動作させる

これらの方法に加えて、スリープ スキャン方法を適用することが大変有効です。

### 5.5.2 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があります。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小さくなります。

長期にわたってデバイスに消費された平均電流は、次の式で計算することができます。

$$I_{AVE} = \frac{(I_{Act} \times t_{Act}) + (I_{Slp} \times t_{Slp})}{T} \qquad \text{式 11}$$

ここで、

$I_{ACT}$ = アクティブ電流

$T_{ACT}$ = アクティブ時間

$I_{Slp}$ = スリープ電流

$t_{Slp}$ = スリープ時間

T = 合計期間

デバイスが消費した平均電力は、次のように計算することができます。

$$P_{AVE} = V_{DD} \times I_{AVE}$$ 式 12

ここで、

$V_{DD}$ = 電源電圧

$I_{AVE}$ = 平均電流

Excel ベースの平均消費電力カリキュレータをご利用いただけます。カリキュレータをダウンロードするには、こちらをクリックしてください。

### 5.5.3 応答時間と消費電力の比較

式 12 で図解されているように、平均消費電力は $I_{AVE}$ または $V_{DD}$ を減少させることにより削減することができます。$I_{AVE}$ はスリープ時間を増やして、削減することができます。スリープ時間を極めて高い値に増加させると、CapSense ボタンの応答時間が遅くなります。結果として、スリープ時間はシステム要件に基づいている必要があります。

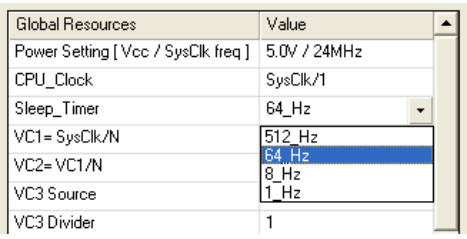
アプリケーションで、消費電力と応答時間は考慮すべき重要なパラメーターである場合、連続スキャンとスリープ スキャン モードの双方を組み合わせて最適化する方法を使用できます。この方法では、デバイスが大部分の時間はスリープスキャンモード(前のセクションで説明したように、定期的にセンサーをスキャンしてスリープ状態になる)であるため、消費電力が削減されます。ユーザーがセンサーに触れてシステムを操作すると、デバイスは連続スキャン モードに移行します。連続スキャン モードでは、スリープを起動することなくセンサーを定期的にスキャンし、極めて優れた応答時間を提供します。デバイスは、指定されたタイムアウト期間の間、連続スキャン モードのままとなります。ユーザーがこのタイムアウト期間内にセンサーを操作しない場合、デバイスはスリープ スキャン モードに戻ります。

### 5.5.4 平均消費電力の測定

以下の指示では、スリープ スキャン方法を使用するときの平均消費電力を判定する方法について説明します。

1. スリープ状態に入らずに全センサーをスキャンするプロジェクトを作成します(連続スキャン モード)。センサーをスキャンする前に、コードにピン トグル機能を含めます。出力ピン状態を切り替えれば、タイムマーカーとなり、オシロスコープで確認することができます。
2. プロジェクトを CapSense デバイスにダウンロードし、消費電流を測定します。測定された電流を $I_{ACT}$ に割り当てます。
3. データシートからスリープ電流情報を入手し、それを $I_{Slp}$ に割り当てます。
4. 出力ピンの切り替えをオシロスコープで切り替えを監視し、2 回の切り替えの間の期間を測定します。これにより有効時間が分かります。この値を $t_{Act}$ に割り当てます。
5. スリープ スキャンをプロジェクトに適用します。スリープ スキャン サイクルである時間 T は、図 5-1 に示すように、グローバル リソース ウィンドウでスリープ タイマー周波数を選択して設定します。
6. スリープ スキャン サイクル時間からアクティブな時間を差し引き、スリープ時間を求めます。$t_{Slp} = T - t_{Act}$。
7. 式 11 を使用して平均電流を計算します。
8. 式 12 を使用して平均消費電力を計算します。

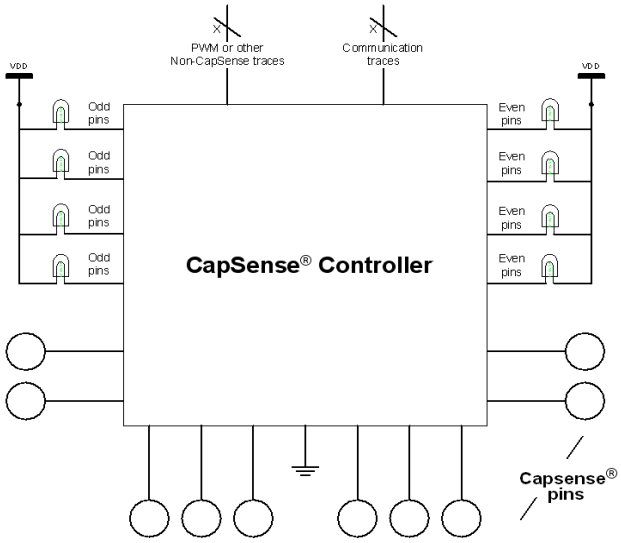
図 5-1. グローバル リソース ウィンドウ



## 5.6 ピンの割り当て

CapSense センサー配線と通信配線と非 CapSense 配線間の相互影響を制限する効果的な方法としては、それぞれをポートの割り当てで隔離することです。図 5-2 に、32 ピン QFN パッケージ向けのこの分離の基本バージョンを示します。各機能が隔離されているため、CapSense コントローラーは通信配線、LED 配線と検知配線の交差がないように配置されます。

図 5-2. 推奨: 通信、CapSense と LED の配線時ポートによる隔離




CapSense コントローラー アーキテクチャは、偶数および奇数のポート ピン番号について電流量を必要とします。奇数のポート ピン番号の電流量が 100 mA である場合、すべての奇数のポート ピンから引き出される合計電流は、100 mA を超えてはいけません。電流量の制限に加えて、CapSense コントローラーデータシートで定義された各ポート ピンの最大電流限度があります。

すべての CapSense コントローラーは高電流吸い込みまたは吐き出しが可能なポート ピンを提供します。高電流シンクまたはポート ピンからのソースを使用する際は、ノイズを最小限に抑えるために、デバイス グランド ピンに最も近いポートを使用してください。

## 5.7 プリント基板レイアウトのガイドライン

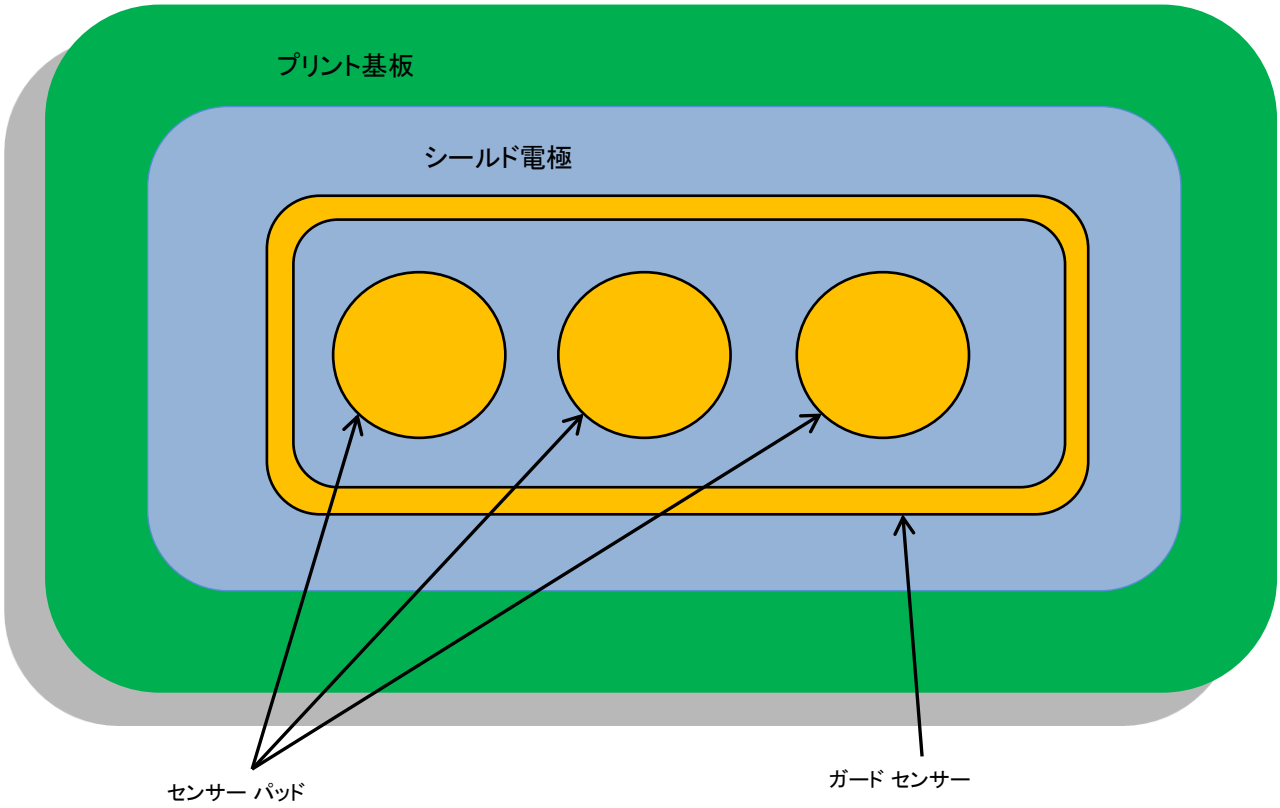
詳細なプリント基板のレイアウト ガイドラインについては、「CapSense 入門」を参照してください。

# 6. 耐水性

一部の CapSense 静電容量タッチスクリーンアプリケーションは、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、車載アプリケーション、および産業用アプリケーションは、水、氷および湿度の変化を伴う環境で機能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 センサーが堅牢なタッチスクリーンを提供します。

## 6.1 シールド電極とガード センサー

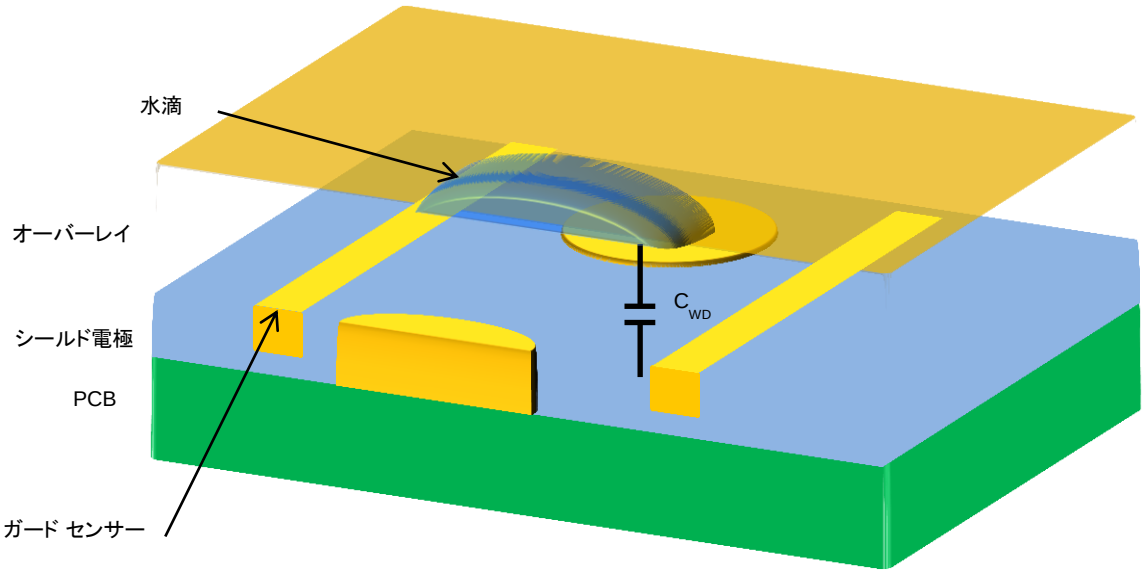
図 6-1. シールド電極とガード センサーを実装したプリント基板のレイアウト




### 6.1.1 シールド基板

シールド電極は、CapSense ボタンセンサーが水滴による誤ったタッチを検出しないように保護します。オーバーレイの表面に水滴が存在する場合、図 6-2 に示すように、シールド電極とセンサー パッド間のカップリングが $C_{WD}$ の分だけ増加します。

図 6-2. 水滴を伴う静電容量測定




- $C_{WD}$ – 水滴およびシールド電極間の静電容量

シールド電極の目的は、タッチ センサーの周囲に水の影響の軽減に役立つ電場を設定することです。シールド電極は、タッチ センサーの電圧をミラーリングすることにより動作します。

これらのガイドラインに従って、適切なシールド動作を確保してください。

- 回路図
- レイアウト
- ファームウェア開発

### 6.1.1.1 回路図

シールド電極アウト信号を駆動するための適切なピンを選択してください。次のピンは、シールド電極アウト信号を駆動するために使用しないでください。

- 変調器コンデンサ ピン ($C_{MOD}$): P0[1]または P0[3]
- フィードバック抵抗ピン ($R_B$): P1[1]または P1[5]
- その他のポート ピン: P0[0]、P0[4]、P1[0]、P1[4]、P2[0]、P2[4]、および P3[0]

### 6.1.1.2 レイアウト

「CapSense 入門」に記載されているレイアウト ガイドラインに従ってください。

### 6.1.1.3 ファームウェア開発

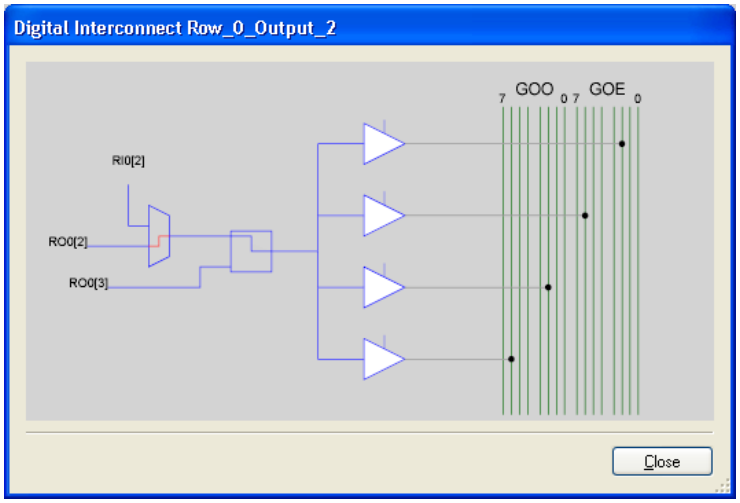
**ステップ 1:** シールド電極信号源を、空いているデジタル Row バスのいずれかから選択してください (Row_0_Output_1～Row_0_Output_3):

- シールドピンが P0[5]、P1[5]、P2[1]、P2[5]の場合は、Row_0_Output_1 を選択します。
- シールドピンが P0[2]、P0[6]、P1[2]、P1[6]、P2[2]、P2[6]、P3[2]の場合は、Row_0_Output_2 を選択します。
- シールドピンが P0[7]、P1[3]、P1[7]、P2[3]、P2[7]の場合は、Row_0_Output_3 を選択します。

**ステップ 2:** シールド電極をシールドピンの外に配線します。以下の手順に従ってください。

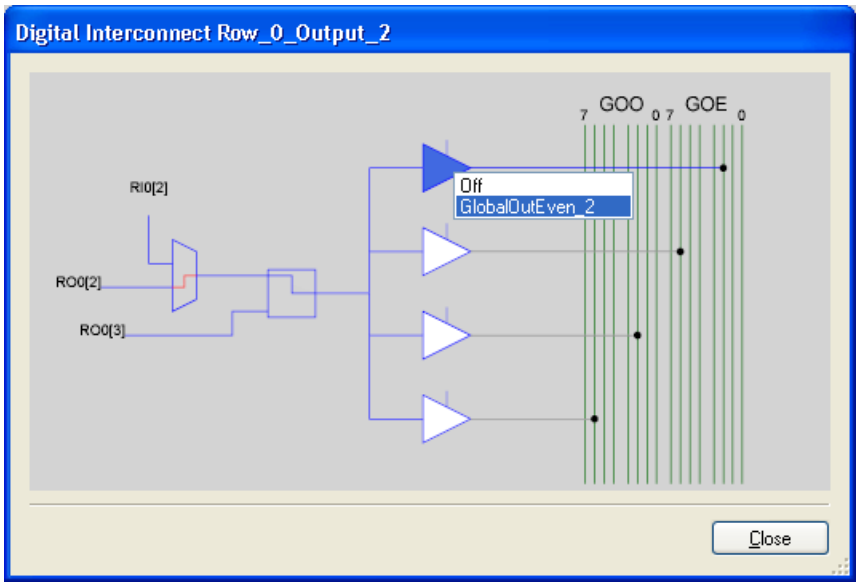
1. シールド電極を選択します。
2. Row 出力信号をグローバル アウト奇数／偶数に配線します。
3. グローバル アウト奇数／偶数をポート ピンに接続します。例えば、P0[2]がシールド ピンとして選択された場合、ユーザー モジュール プロパティから、Row_0_Output_2 の選択でシールド電極を選択します。
4. **Row_0_Output_2** をクリックし、図 6-3 に示すように、デジタル相互接続ビューを開きます。

図 6-3. デジタル インターコネクトの回路図表示



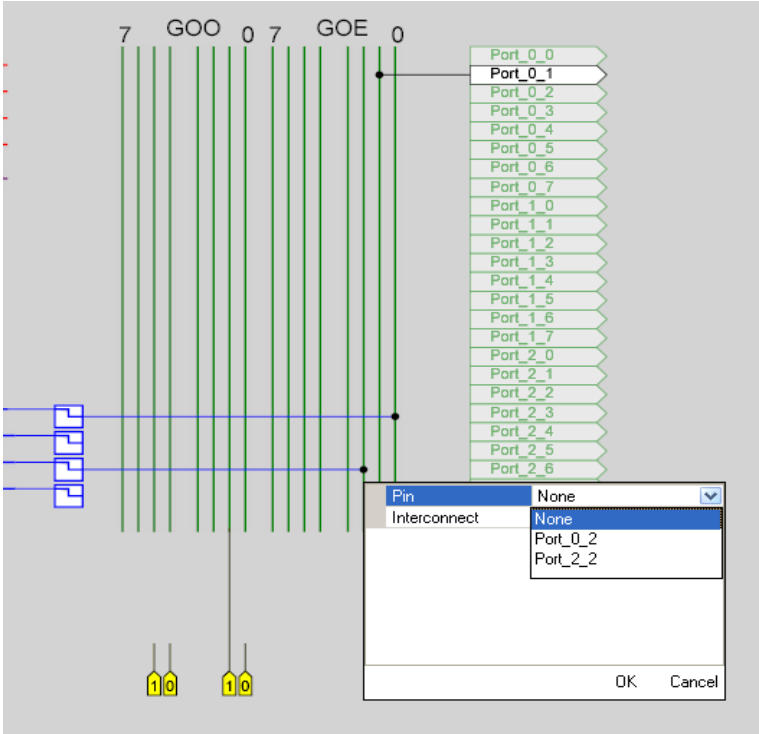

5. 図 6-4 に示すように、**Row_0_Output_2_Drive_0** を選択し、次に **GlobalOutEven_2** を選択します。

図 6-4. 出力選択の回路図表示




6. 次の図に示すように **GlobalOutEven_2** をクリックして、ピン オプションで **P0[2]**を選択します。

図 6-5. シールドとガード センサーを搭載したプリント基板




## 6.1.2 ガード センサー

図 6-6. シールドとガード センサーを搭載したプリント基板

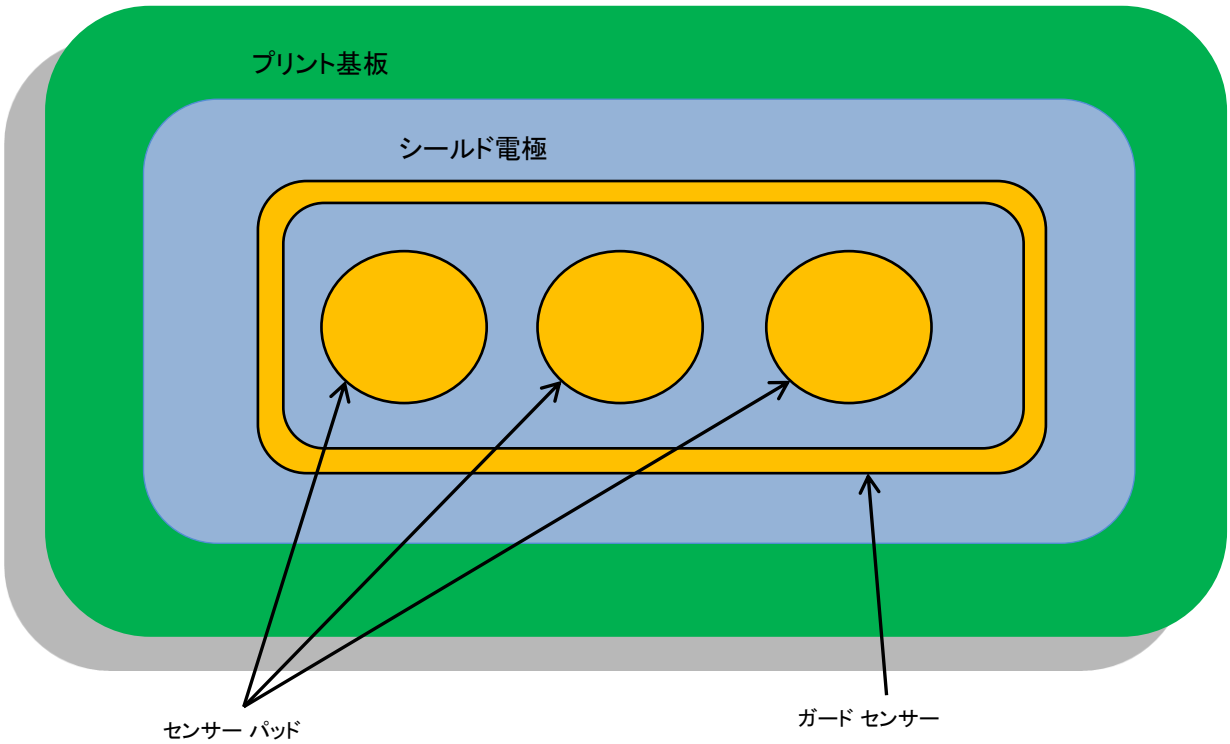
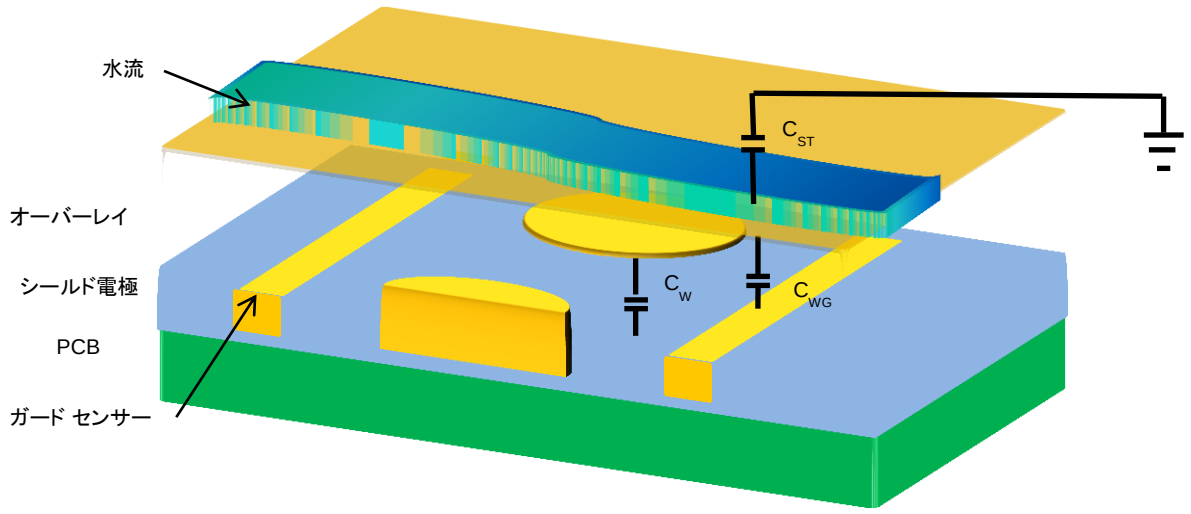


図 6-6 に示されているように、ガード センサーは銅配線で、プリント基板上のすべてのセンサーを囲み、連続的な水流の存在の検知に使用されます。感知表面に水流が存在する場合、図 6-7 に示されているように、大静電容量 $C_{ST}$ がシステムに追加されます。この静電容量は、$C_{WD}$ の数倍大きくなる可能性があります。このため、シールド電極の効果は完全にマスクされており、センサーにより測定される Raw カウントは、指のタッチと同じかそれより高くなります。この場合、ガード センサーが役立ちます。水流を検知すると、その他のセンサーがトリガするのをブロックします。

図 6-7. 水流を伴う静電容量測定



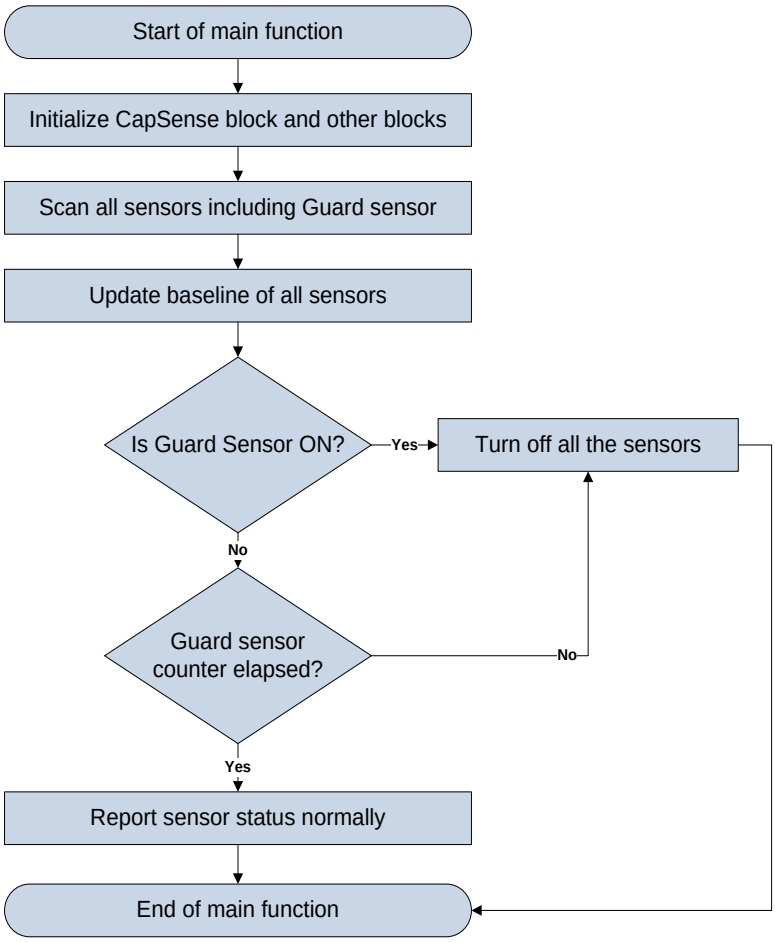

- $C_{WD}$ - 水流およびシールド電極間の静電容量
- $C_{ST}$ - 水流およびシステムグランド間の静電容量
- $C_{WG}$ - 水流およびガード センサー間の静電容量

ガード センサーはファームウェアに実装してください。ガード センサーを実装するには、1 つの CapSense ピンおよびカウンター (ハードウェア／ソフトウェア) が必要です。

水流がボードに適用される際に、ガード センサーがこのイベントを検知して、タッチ センサーの処理ロジックを無効化します。追加のガード センサー「デッド」タイムにより、早過ぎるセンサーの解除を防止します。水流が無くなると、短い間ガード カウンターがタッチ処理を抑制します (ガード センサー カウンター)。これにより、基板上に残っている水による誤ったタッチの検出をなくします。

図 6-8 は、ガード センサーをファームウェアに実装する方法を表したフローチャートです。

図 6-8. ガード センサーを実装するためのフローチャート




## 6.2 設計に関する推奨事項

次のシステム レベルとプリント基板レイアウトに関する推奨事項は、水にさらされる CapSense システムに適用されます。

- シールド電極銅ハッチングに関する推奨事項:
  - 最上層 – 7-mil の配線と 45-mil のグリッド(15-パーセント充填)
  - 最下層 – 7-mil の配線と 70-mil のグリッド(10-パーセント充填)
  - ボタンの間のシールド電極の幅は最低でも 10mm
  - センサー パッドと CapSense コントローラーを取り囲むエリアだけをグランドに接続
- 水滴が自然にセンサーから消えて大きな水滴が蓄積しないように、センサーの表面を、垂直、または水平に対してある角度をもたせて設置します。
- 撥水性で非吸収性のオーバーレイ素材を使用してください。これにより、デバイス パネル上の水の筋および水膜を最小限に抑えます。海水のように導電性の高い水の場合、これは特に重要です。
- アプリケーションが連続的な水流にさらされる可能性がある状況では、ガード センサーを使用してください。デバイスがさらされるのが雨のみである場合、ガード センサーは必要ありません。

# 7. 近接検知

近接センサーは、静電容量式タッチ面に接触する前に、手またはその他の導電性の物質の存在を検知します。暗い状態で車載用オーディオ システムを操作する場合を想像してください。近接検知は、指を近づけることによりシステムをイネーブルにしライトアップします。たとえば、オーディオシステムのボタンは、ユーザーの手が近くにあるとバックライト LED を光らせます。

## 7.1 近接センサーの種類

### 7.1.1 ボタン

大きな $C_P$ および小さな差分カウントを備えたボタンは、近接センサーとして機能することができます。ボタンとして実装された近接センサーの感度は、通常の静電容量式タッチスクリーンのボタンよりもはるかに高くなります。

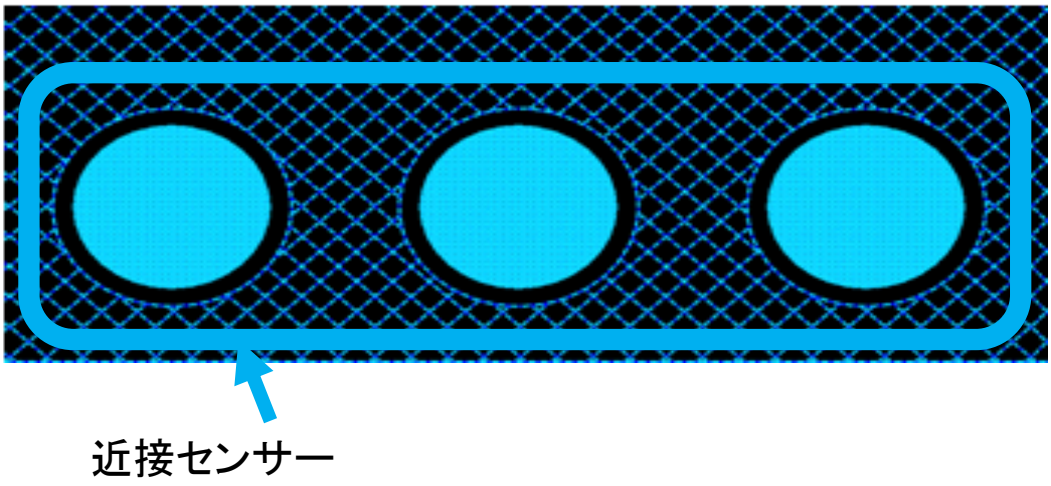
### 7.1.2 ワイヤ

シングル長のワイヤは、近接センサーとして有効に機能します。手の検出は電界の変化による静電容量の変化に依存しているため、ワイヤの周囲の電界に影響を与える浮遊静電容量または物体は近接センサーの範囲に影響を与えます。ワイヤ センサーの使用は、製造コストや複雑さから、大量生産に関する最適なソリューションではありません。

### 7.1.3 プリント基板配線

長いプリント基板配線は、近接センサーを形成できます。配線は、直線か、または図 7-1 に示されているようにシステムのユーザー インタフェースの周囲を囲む可能性があります。この方法は大量生産には適していますが、ワイヤ センサーほど感度は高くありません。

図 7-1. プリント基板配線を使用した近接センサー



### 7.1.4 センサー連結

近接センサーを実装するもう一つの方法は、センサーを一緒に連結することです。これは、PSoC Designer における内部アナログ マルチプレクサ バスからのセンサーを接続するファームウェアを使用して、複数のセンサーを 1 つの大きなセンサーに結合することにより実現されます。この方法を利用する際は、設計の $C_P$ 限度を超えないように注意してください。

## 7.2 設計に関する推奨事項

シールド電極を効果的に使用することにより、近接センサーの検知距離を拡大します。これは、金属が存在する場で近接センサーが動作する必要がある場合に特に役に立ちます。

シールド電極から遠く離れた場所に配置可能であるため、ワイヤ センサーはシールド電極の効果がより有益になります。

近接センサー内の大きな固体グランド フィル エリアは、感度を低下させるので避けてください。

スキャン速度を遅くすると、あらゆる距離において感度が増加します。近接センサーの場合、スキャン速度は極めて遅くしてください。

# 8. 低消費電力設計の注意事項

消費電力は、マイクロコントローラー デザインの重要な側面です。CapSense コントローラーにより消費される平均電流を軽減するいくつかの技術のなかで、スリープ モードが最も広く用いられています。何の機能も実行する必要がない場合、CapSense コントローラーは、携帯電話で一定のアイドル時間の後バックライトが暗くなると同じように、スリープ モードになります。これは、すべてのバッテリー使用アプリケーションがそうであるように、デバイスの消費する平均電流を削減するためです。CapSense コントローラーは、CPU_SCR0 レジスタ (ビット 3) 内のスリープ ビットに「1」と書くとスリープ モードに入ります。これは、M8C_スリープ マクロを呼び出すことにより実行されます。スリープ モードの間、CPU は停止し、内部メイン発振器 (IMO) は無効化され、バンドギャップ電圧リファレンスの電源は切断されてフラッシュ メモリ モデルは無効化されます。供給電圧モニターと 32-kHz 内部オシレータ回路のみが動作するままにされます。スリープ モード以外の省電力技術を以下に示します。

- CapSense (PSoC) アナログブロック リファレンスを無効にする
- Continuous time (CT) およびスイッチト キャパシタ (SC) ブロックを無効化する
- CapSense (PSoC) アナログ出力バッファーを無効にする
- 駆動モードをアナログ HI-Z にセットする

スリープ モードには、デザインに対する悪影響があります。注意を怠ると、想定外の状況が発生することがあります。PSoC は、スリープを解除し、またユーザーはデバイスがスリープ モードで追加処理を行うことを認識しておく必要があります。

## 8.1 追加的な省電力技術

スリープ モードを除くすべての省電力技法は、アプリケーションに基づいており、期待しない結果をもたらすものもあります。各技法について以下に詳しく説明します。

```
ABF_CR0 &= 0xc3; // Buffer Off
```

### 8.1.1 駆動モードをアナログ HI-Z にセットする

CapSense コントローラーの駆動モードが消費電力に悪影響を及ぼすことがあります。駆動モードは、システムに悪影響を与えないピンにおいてのみ変更可能です。ライングリッチを発生させないように変更を実施します。この順序は、ピンの現在の駆動モードおよびポート データ レジスタの状態に依存します。HI-Z またはストロング 駆動モードの間で切り替える際に、CapSense コントローラー駆動モード構造のために、ピンは一時的に抵抗プルアップ、または、抵抗プルダウン駆動モードのどちらかに入る必要があります。一時駆動モードは、ピンに関わるその前の値と反対になります。そのため、ピンが HIGH で駆動された場合、一時的な駆動モードは抵抗プルダウンである必要があります。これにより、ピンの駆動モードにレジスティブがないことを保証し、起こり得るあらゆるグリッチを排除します。

駆動モードは、スリープに入る前にソフトウェア内でマニュアウでセットされます。駆動モードをコントロールする 3 つのレジスタがある、PRTxDM0、PRTxDM1 と PRTxDM2。ピンには 1 レジスタ当たり 1 ビットが割り当てられています。そのため、単一ピンの駆動モードを変更するには、3 つのレジスタ書き込みが必要になります。しかし、同一の 3 つのレジスタ書き込みにより、ポート全体が変更されるので便利です。アナログ HI-Z の適正なビットパターンは 110b です。次のコードを使用して、最初に抵抗プルダウンに進み、ポートゼロをストロングからアナログ HI-Z に設定します。

```
PRT0DM0 = 0x00; // low bits
PRT0DM1 = 0xff; // med bits
PRT0DM2 = 0xff; //high bits
```

### 8.1.2 まとめ

以下のコードは、28 ピン部分の代表的スリープ準備シーケンスの例です。この順序では、割り込みは無効化され、アナログ回路はオフになり、すべての駆動モードはアナログ HI-Z に設定され、割り込みが再度有効にされます。

```
void PSoC_Sleep(void){
   M8C_DisableGInt;
   ARF_CR &= 0xf8; // analog blocks Off
   ABF_CR0 &= 0xc3; // analog buffer off
   PRT0DM0 = 0x00; // port 0 drives
   PRT0DM1 = 0xff;
   PRT0DM2 = 0xff;
   PRT1DM0 = 0x00; // port 1 drives
   PRT1DM1 = 0xff;
   PRT1DM2 = 0xff;
   PRT2DM0 = 0x00; // port 2 drives
   PRT2DM1 = 0xff;
   PRT2DM2 = 0xff;
   M8C_EnableGInt;
   M8C_Sleep;
}
```

### 8.1.3 スリープ モードの混乱

CapSense コントローラーは、リセット、または、割り込みによってスリープから解除することができます。CapSense コントローラーはリセットが 3 種あります: 外部リセット、ウォッチドッグ リセット、およびパワーオン リセット。これらのリセットのどれでも CapSense コントローラーにスリープ モードを終了させます ; リセットがデアサートすると、CapSense コントローラーは *Boot.asm* から始まるコードの実行を開始します。CapSense コントローラーをウェイクアップするのに使用できる割り込みはスリープ タイマー、低電圧モニター、GPIO、アナログ カラム、および非同期です、割り込みにより CapSense コントローラーを有効にする場合と、スリープ中にデジタル通信をしようとする場合にスリープ モードの混乱が問題になります。これらの検討事項については、次のセクションで詳しく説明されています。

### 8.1.4 保留中の割り込み

割り込みがペンディング、有効化、そして、CPU_SCR0 レジスタのスリープ ビットへの書き込みの後にスケジュールされている場合、システムはスリープ状態にはなりません。これでも命令は実行されますが、CapSense コントローラーによりスリープ ビットが設定されありません。その代わり、割り込みが可能となり、事実上 CapSense コントローラーは、スリープ命令を無視することになります。これを回避するには、スリープ準備が行われている間に割り込みが全体的に無効化され、その後スリープ ビットを書き込む直前に割り込みを再度イネーブルにしなければなりません。

### 8.1.5 グローバル割り込みのイネーブル:

CapSense コントローラーを割り込みによってウェイクするには、グローバル インタラプト イネーブル レジスタ (CPU_F) を、有効にする必要はありません。以下の例のように、割り込みによってスリープから解除する場合の必要条件は INT_MSKx レジスタ内の適切な割り込みマスクを使用します。グローバル割り込みが無効化されている場合、CapSense コントローラーをウェイクアップする ISR は実行されませんが、CapSense コントローラーはスリープ モードを終了します。

この場合、ISR を可能にするには、保留中の割り込みを手動でクリアするか、グローバル割り込みを有効にする必要があります。割り込みは、INT_CLRx レジスタ内でクリアされます。

```
//Set Mask for GPIO Interrupts M8C_EnableIntMask(INT_MSK0, INT_MSK0_GPIO)

// Clear Pending GPIO Interrupt
INT_CLR0 &= 0x20;
```

## 8.2 ウェイクアップ後の実行シーケンス

CapSense コントローラーがリセットによりウェイクアップされる場合、ブート コードの始めに実行が開始されます。CapSense コントローラーが割り込みサービスルーティンにより有効になった場合、スリープ命令直後の命令が最初に実行されます。これは、スリープ命令直後の命令は、CapSense コントローラーが完全にスリープになる前にプリフェッチされているからです。そのため、グローバル割り込みが無効にされた場合、命令実行はスリープが開始される前に中断されたところから再開されます。

### 8.2.1 PLL モードの有効化

PLL モードが有効にされた場合、CPU 周波数をスリープに入る前に、最低の 3MHz まで下げる必要があります。これは、CapSense コントローラーが起動して、再度ウェイクアップされた後に、PLL が再度ロックしようとして、毎回オーバーシュートするからです。さらに、適正に実行するために、通常 CPU オペレーション前に 10ms 待つ必要があります。このことは、スリープ モードと PLL を使用するにはソフトウェアは 3MHz で実行できなければならないことを意味します。OSC_CR0 レジスタへの簡単な書き込みによって、CPU を減速することができます。しかし、このレジスタは SYSCLK のデバイダをセットするだけなので、異なった SYSCLK を使う部品群間で CPU 速度が異なることを意味します。通常、SYSCLK は 24MHz です。

```
OSC_CR0 &= 0xf8; // CPU = 3 IMO = 24
```

### 8.2.2 グローバル割り込み有効化の実行

スリープ ビットの書き込みの命令境界に割り込みがあることは望ましくありません。これはスリープコマンドが割り込みからの戻り (reti) 命令が実行されたときには、スリープ モード入りをバイパスされるように全てのファームウェアを予め準備しておく必要があるということです。これを防止するために、割り込みはスリープ準備前に、一時的に無効化され、スリープに入る前に再度イネーブルにします。グローバル割り込みの命令のタイミングにより、次の命令の間は割り込みを起こせません。この場合は、スリープ ビットの設定がこれに該当します。

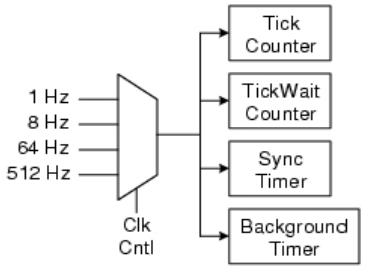
### 8.2.3 スレーブ モードを持った $I^2C$ スレーブ

$I^2C$ スレーブをスリープ モードで使用する場合、いくつかの混乱があります。スリープの間は、IMO と CPU が停止しているため、CapSense コントローラー内には何の処理もありません。問題は $I^2C$ アドレスで発生します。$I^2C$ スタート条件が特定のアドレスに送信された場合、CapSense コントローラーがアドレスを処理できないために、NAK で応答します。代表的な回避策は、$I^2C$ バスのクロック、または、データ ライン上に立ち下がりエッジ割り込みを作る方法です。こうすればマスターが CapSense コントローラーをウェイクアップするダミーのスタート条件を送信することができます。ウェイクアップと $I^2C$ アドレスが処理可能となるまでの間に、若干のタイムラグがあるため、マスターは次の送信を最長 200µs 遅らせるか、または、ACK を受信するまで送信を継続する必要があります。このソリューションには、CapSense コントローラーがあらゆる $I^2C$ 立ち下がりエッジ トラフィックでウェイクアップし、それにより合計アクティブ時間が増加してスリープ電流が高くなるという第 2 の問題があります。もう一つのソリューションでは、3 番目の GPIO ピンを使用して CapSense コントローラーをウェイクアップし、適切な遅延時間の後で初回の START 条件を送信します。

### 8.2.4 スリープ タイマー

CapSense コントローラーは、スリープ タイマーとスリープ タイマー ユーザー モジュールを提供します。これらは CapSense コントローラーがスリープになっている間に使用され、両方とも同様の機能を実行します。実際のスリープ タイマーは、常に電源が切られない内部低速オシレータをコピーします。タイマーは、1Hz、8Hz、64Hz と 512Hz の選択可能な間隔で割り込みを発生させます。CapSense コントローラーを定期的に有効にして何かの処理をする、またはアクティビティのチェックは、多くの場合効果的です。この一例として、定期的にウェイクアップし、センサーをスキャンする方法があります。スリープ タイマー ユーザー モジュールは別の追加的機能を生成させるのにスリープ タイマーを使用します。この機能としては定期的な割り込みを発生するバックグランド ティック カウンター、プログラム ループの遅延機能、設定可能なダウンカウンター、ループタイムをコントロールするループガバナーなどがあります。図 8-1 にこの機能の簡単なブロック図を示します。

図 8-1. スリープ タイマー ユーザー モジュール ブロック図



# 9. リソース

## 9.1 ウェブサイト

サイプレス CapSense コントローラー ウェブサイトでは、この節で討論した参照資料すべてを見ることができます。

CY8C21x34/B ウェブページには、CapSense CY8C21x34/B デバイス ファミリのさまざまな技術リソースが掲載されています。

## 9.2 データシート

CapSense CY8C21x34/B デバイス ファミリのデータシートは、http: //www.cypress.com から入手できます。

- CY8C21234, CY8C21334, CY8C21434, CY8C21534, CY8C21634
- CY8C21234B, CY8C21334B, CY8C21434B, CY8C21534B, CY8C21634B

## 9.3 テクニカル リファレンス マニュアル

サイプレスは、以下の技術リファレンス マニュアルを作成し、トップレベルのアーキテクチャ図、レジスタ、およびタイミング図などの CapSense コントローラーの機能に関する情報に素早く簡単にアクセスできるようにしています。

- CY8CPLC20 、CY8CLED16P01 、CY8C29x66 、CY8C27x43 、CY8C24x94 、CY8C24x23 、CY8C24x23A 、CY8C22x13 、CY8C21x34 、CY8C21x23 、CY7C64215 、CY7C603xx 、CY8CNP1xx 、および CYWUSB6953 PSoC(R) プログラマブル システム オンチップ 技術リファレンス マニュアル(TRM)

## 9.4 開発キット

### 9.4.1 汎用 CapSense コントローラー キット

汎用 CapSense コントローラー キットは設計済みのコントロール回路およびプラグイン ハードウェアを備えており、プロトタイピングとデバッグを簡単にします。チューニングとデータ取得用にプログラミングと $I^2C$-USB ブリッジ ハードウェアが含まれています。

- CY3280-BK1 汎用 CapSense コントローラー

### 9.4.2 汎用 CapSense モジュール基板

#### 9.4.2.1 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 汎用 CapSense コントローラー基板と接続します。

#### 9.4.2.2 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 モジュールには、4×4 マトリックス様式に整理され、物理的ボタン 16 個となる LED8 個と CapSense センサー8 個が含まれます。このモジュールはあらゆる CY3280 汎用 CapSense コントローラー基板と接続します。

#### 9.4.2.3 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 汎用 CapSense コントローラー基板と接続します。

#### 9.4.2.4 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 汎用 CapSense コントローラー基板と接続します。

#### 9.4.2.5 汎用 CapSense プロトタイピング モジュール

CY3280-BBM 汎用 CapSense プロトタイピング モジュールは、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にアクセスが可能です。プロトタイピング モジュール基板は、汎用 CapSense コントローラー ボードと共に使用され、その他の専用の汎用 CapSense モジュール基板にはない追加機能を実装します。

### 9.4.3 インサーキット エミュレーション (ICE) キット

ICE ポッドは、パッケージ特有のポッド フィートによりフレックス ケーブルまたはプリント基板を経由して、CY3215-DK インサーキット エミュレータとプロトタイプ システム上の対象 PSoC デバイス間の相互接続を提供します。以下のポッドを利用可能です。

- QFN CY8C21x34 PSoC デバイスのデバッグ用 CY3250-21X34QFN インサーキット エミュレーション (ICE) ポッド キット

## 9.5 PSoC Programmer

PSoC Programmer は柔軟性のある、統合されたプログラミング アプリケーションで、PSoC デバイスをプログラムします。PSoC Programmer は PSoC Designer と PSoC Creator と共に使用し、あらゆる設計を PSoC デバイスにプログラムします。

PSoC Programmer は、API をもつハードウェア レイヤを、プログラマとブリッジ デバイスを使用する設計専用アプリケーションに提供します。PSoC Programmer ハードウェア層は、COM ガイド ドキュメントだけでなく、C#、C、Perl、および Python の言語でコード例としても説明されています。

## 9.6 MultiChart

MultiChart はリアル タイムで CapSense データを表示と記録する、簡単な PC 用ツールです。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。

## 9.7 PSoC Designer

サイプレスは、第一級の統合設計環境である PSoC Designer を提供しています。PSoC Designer では、アナログおよびデジタル ブロックのコンフィギュレーション、ファームウェアの開発および設計のチューニングが可能になります。CapSense を含めた、あらかじめ特性確定済みのアナログおよびデジタル機能のライブラリを使い、ドラッグ & ドロップによる設計環境でアプリケーションを開発できます。PSoC Designer には内蔵の C コンパイラと組み込みプログラマが含まれています。複雑なデザイン用には pro コンパイラがあります。

## 9.8 サンプル コード

サイプレスには、ユーザーのデザインを速やかに完成させるための大量のコード実例があります。

- CapSense コントローラー コード例の設計ガイド
- CY8C21x34 に EzI2Cs スレーブを搭載した CSD ソフトウェア フィルター
- PSoC の SPI EEPROM へのインタフェース
- 擬似ランダム系列発生器

- CY8C21x34 に I2CHW スレーブを搭載した CSD
- CMPPRG ユーザー モジュール例
- CYRF6936 および CY8C27643 を使用した受信文字列
- CY8C21x34 に SPIS を搭載した CSD

## 9.9 設計サポート

サイプレスには多くの設計サポート チャンネルがあり、CapSense ソリューションの成功を期しています。

- 知識ベース記事： 製品ファミリごとのテクニカル記事を参照する、あるいはさまざまな CapSense トピックを検索してください。
- CapSense アプリケーション ノート： この文書で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：静電容量タッチ インターフェースの高度なトピックについて学びます。
- 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。
- CapSense 製品セレクタ ガイド: サイプレス CapSense 製品ラインの製品提供の全体を見ることができます。
- 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になることができます。
- 技術サポート：流の技術サポートをオンラインで受けられます。

## 9.10 改訂履歴

| 版 | 発行日 | 変更者 | 変更内容 |
|---|---|---|---|
| ** | 05/05/2012 | HZEN | これは英語版 001-66271 Rev. *B を翻訳し た日本語版 001-78915 Rev. **です。 |
| *A | 05/21/2015 | HZEN | これは英語版 001-66271 Rev. *F を翻訳し た日本語版 001-78915 Rev. *A です。 |