AN1334

堅牢なタッチセンシングの設計手法

Author: *Burke Davison*
Microchip Technology Inc.

はじめに

このアプリケーション ノートでは、ノイズの多い環境で使う静電容量式タッチ アプリケーションの最良の設計手法を説明します。最初に、ノイズが引き起こす問題を定義し、そのノイズが一般的にシステムにどのような影響を与えるかについて説明します。次に、アプリケーション本来の S/N 比 (SNR) を最大化するのに役立つハードウェア ガイドラインを提供します。続いて、ソフトウェアの手法を説明します。ここでは、センサの信号にフィルタをかけて SNR を高めた後、静電容量式センサの挙動に基づいてデコード値を求めるのに良く使われる手法をいくつか説明します。

本書で扱うハードウェア設計のトピックは以下の通りです。

1. ボタンおよびスライダパッドの設計と間隔
2. カバーの材質と厚さ
3. 推奨接着層
4. センサパターンと直列抵抗
5. 静電気放電(ESD)保護に対応するレイアウト手法
6. 電源と接地経路
7. VDD とバイパス コンデンサの選択

mTouch™ および RightTouch™ センシング ソリューション システムは、伝導および放射ノイズ感受性と放射妨害波に関する業界標準の試験に合格しました。このアプリケーション ノートでは、静電容量式タッチ設計に関する重要な事項を説明します。これらの事項を適切なプリント基板 (PCB) 手法と連携して活用すると、システムは過酷環境でも性能を維持できます。

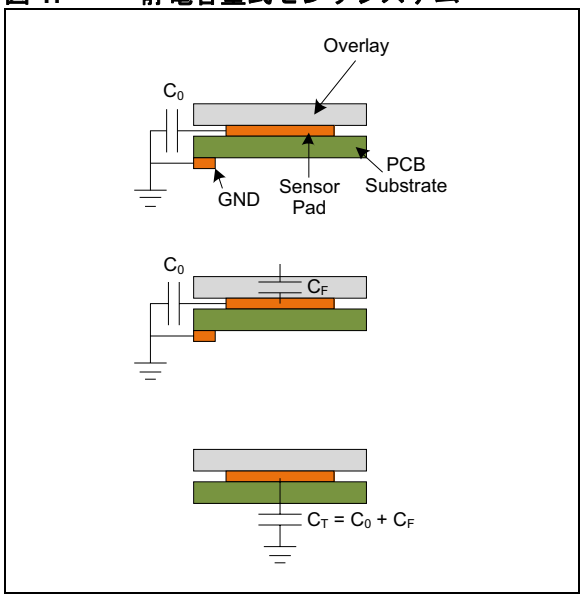
静電容量式タッチセンシングの基礎と応用については、Microchip 社のウェブページ「Touch and Input Sensing Solutions」(http://www.microchip.com/mTouch) を参照してください。

静電容量式タッチの基本事項の確認

静電容量式センサは銅箔で埋められたPCB上の領域で、トレースパターンを使って PIC® マイコンに接続されます。PIC マイコンは、通常ユーザの指がセンサに近づく事で生じる静電容量のわずかな変化を検出して計測します。ソフトウェアは常時静電容量を読み取り、システムは静電容量に変化があった時にセンサが押されたものとして判断します。

図 1 で、C_{BASE} はパッド上に物体がない場合の静電容量値です。この値をセンサの基準静電容量と呼びます。C_F は指のタッチで生じる静電容量の変化です。C_T はセンサの総静電容量です。

図 1: 静電容量式センサシステム



このシステムの静電容量は式 1 に示す平行平板静電容量で求める事ができます。実際のアプリケーションでは静電容量式センサシステムは式 1 よりもはるかに複雑である事に注意が必要です。一般的に、システムは PCB、カバー、人体、環境の結果であるコンデンサ、抵抗、インダクタの回路網と見なす事ができます。従って、静電容量式センサシステムの正確な特性を求める事は非常に困難です。

本書の推奨内容は実験に基づいたものであり、絶対的な条件ではありません。静電容量の式とアプリケーションの要求に基づいて、ユーザが独自に調整すべきものです。

式 1:　　平行平板静電容量

$$C = \varepsilon_r \varepsilon_0 \frac{A}{d}$$

ε_r = 誘電体の比誘電率

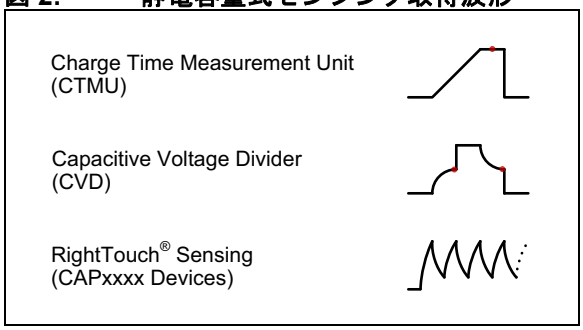
ε_0 = 真空の誘電率 (8.854 x 10^{-12} F/m)

A = 電極の面積 (m^2)

d = 電極間の距離 (m)

マイクロコントローラを使って静電容量の変化を検出する手法は 2 つあります。1 つ目は電圧を計測する方法です。センサピンの静電容量値に基づいて電圧をセンサに印加し、電圧の変化を検出します。Microchip 社の充電時間計測ユニット (CTMU) や静電容量式分圧器 (CVD) 等がこれに該当します。2 つ目は周波数を使って計測する方法です。擬似的に無作為化した周波数で静電容量の変化を検出するRightTouchスキャンがこれに該当します。図 2 に、上記 3 つのスキャン手法で得られる波形を示します。

図 2:　　静電容量式センシング取得波形



取得波形

本書はシステムのハードウェア設計と、ファームウェアの中でも信号の取得に関連しないセクションに焦点を当てています。CVD または CTMU 手法を使う場合、Microchip アプリケーション ライブラリのソースコードを使って実装できます。RightTouch ターンキー製品を使う場合、これらの手法はあらかじめソリューションに組み込まれています。

「ノイズ耐性」対「低消費電力」

静電容量式タッチシステムを開発する場合、製品開発の初期段階から、主目標を明確にしておく事が重要です。大部分のアプリケーションでは、システムへの電源供給方法が、この答えです。ライン電源駆動システムの場合、伝導ノイズ耐性が主な課題です。バッテリ駆動システムの場合、低消費電力化が主な課題です。

また、ライン電源駆動、バッテリ駆動を重複して使うシステムもあります。その例として、USB ケーブルで電源を供給できる携帯電話が挙げられます。このような携帯電話では通常使用時は低消費電力化が主な課題ですが、主電源ラインから電源供給する時の伝導ノイズも考慮する必要があります。この理由から、これらのシステムには電圧ベースの取得手法のみを使う必要があります。

ノイズ耐性と低消費電力は必ずしも両立しないとは言えませんが、どちらか一方に焦点を当てると、もう一方に関しては妥協が必要です。例えば、伝導ノイズに対する感受性を低減するためにスルーレート リミッタフィルタを実装するとサンプリング レートを上げる必要があり、結果として総消費電力が増加します。アプリケーションの低消費電力化においてV_{DD}を下げる事は非常に有効ですが、そうするとノイズ耐性が低下してしまいます (セクション、「電源に関する注意事項」参照)。本書ではノイズ感受性に焦点を当て、低消費電力化は二次目標として扱います。アプリケーションの主目標が低消費電力化の場合、http://www.microchip.com/XLP にアクセスして詳細な技術解説を参照してください。

静電容量式タッチセンサにおけるノイズの影響

「プッシュボタン」対「静電容量式センサ」

堅牢な静電容量式タッチ アプリケーションの開発方法を検討する前に、ノイズが問題となる基本的な理由を理解しておく事が重要です。機械式ボタンを使う場合、マイクロコントローラのポート回路が、スイッチのピンが High/Low のどちらにあるかを判断し、1 ビットのバイナリ結果を出力します。その後、この結果はリンギングを調整するためにデバウンスされます。ボタンの状態は、デバウンス変数の状態によって決まります。

しかし、静電容量式タッチセンサ アプリケーションはアナログです。一番大きな違いは、データを手動で読み取る必要がある点です。機械式スイッチを使う場合、マイクロコントローラは内部のハードウェア ロジックを使ってピンを読み出す事ができます。静電容量式タッチ アプリケーションの場合、別のハードウェア モジュールを使ってセンサトレースを操作する必要があります。電圧ベースと周波数ベースのどちらの計測でも、アナログ結果は整数値で提供されます。通常は、この後各種デジタル信号処理手法を使ってこの値にフィルタをかける事で信号を増幅、ノイズを減衰させます。続いて、フィルタ後の値にデバウンス アルゴリズムとより複雑なデコーディング プロセスを適用します。システムを閉ループで動作させ、センサの現在の状態に基づいて挙動を調節するように設計している場合、より複雑な処理が必要です。

静電容量式タッチ ソフトウェアの処理は、3 つの段階に分けてとらえる事ができます。

1. 取得

   電圧ベースまたは周波数ベースの計測手法を使って、静電容量式タッチセンサからサンプルを取得します。

2. フィルタ処理

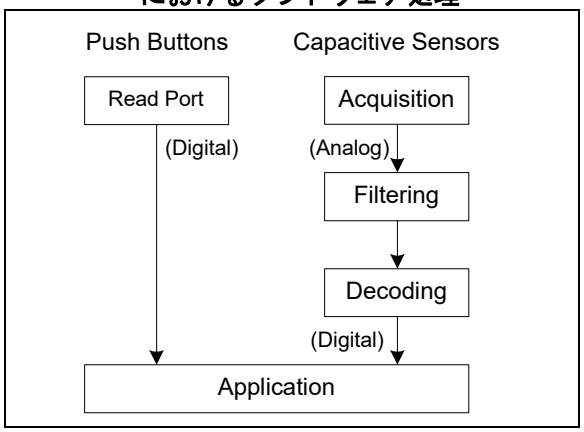
   センサで取得したサンプルを処理し、ノイズを減衰させてシステムの実効 SNR を高めます。

3. デコード

   サンプルの現在値と直前のセンサ挙動に基づいて、センサが押されたのか解放されたのかを判断します。

図 3 に、プッシュボタンと静電容量式タッチセンサの違いを示します。また、静電容量式タッチシステム ソフトウェアの 3 つの主な段階も示します。

図 3: プッシュボタンと静電容量式センサにおけるソフトウェア処理



伝導ノイズと放射ノイズ

静電容量式タッチシステムを不安定にするノイズには、大きく分けて「伝導」と「放射」の 2 種類があります。伝導ノイズは、外部のデバイスから電源を供給するシステムで発生します。これには、主電源ラインから電源を取っているシステム、デスクトップから電源を供給する USB デバイス、ユーザとアプリケーションがグランドを共有しないようなその他のあらゆる状況が含まれます。

放射ノイズは、全ての静電容量式タッチシステムに共通の課題です。特に静電容量式タッチセンサはスキャン中に高インピーダンス入力となるため、高周波アンテナとして作用してしまいます。そのため、静電容量式タッチシステムの近くに電磁場を放射する電子デバイスがあると、読み出しに影響を与えます。例えば、携帯電話、高出力の通信線、蛍光灯がこれに該当します。

これらの 2 種類のノイズが生じる主な理由は、以下の 2 つです。

1. ユーザが静電容量式タッチセンサを押下すると、ユーザはシステムの一部となります。そのため、ユーザとシステムの参照グランドが異なる場合、システムはユーザをセンサに注入された AC 信号と解釈します。
2. アナログの読み出し値は外部の影響を受けて変動しがちです。一方、機械式スイッチは、High または Low どちらかのデジタル値を出力します。

本書では、これら 2 タイプのノイズを回避するための推奨設計手法を紹介します。これらのガイドラインに加えてアプリケーションの将来的な動作環境を考慮し、システムに干渉する恐れのあるノイズの多い電子機器が近くに存在しないようにする必要があります。

静電容量式センサノイズの挙動

静電容量式タッチセンサにノイズが注入されると、システムが不安定になります。CTMU、CVD 等の電圧ベースの mTouch センシング ソリューションにおけるノイズの影響は、RightTouch® スキャン手法等の周波数ベースの読み出しの場合とは異なります。電圧ベースのシステムでは、特定の時点におけるセンサの電圧で読み出す整数値が決まります。周波数ベースのシステムでは、注入されたノイズの周波数とセンサの発振周波数により、読み値に対する影響が異なります。

ノイズの挙動 : 周波数ベースの取得

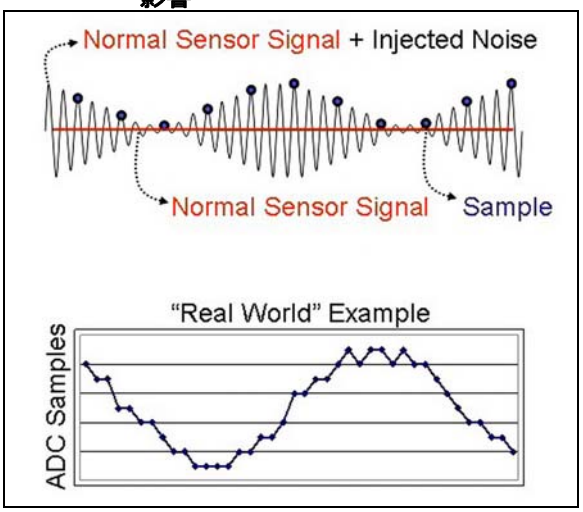
周波数ベースの取得方法では、周波数ホッピングで高調波ノイズを除去する必要があります。RightTouch ターンキー製品では、周波数ホッピングは自動的に行われ、常に有効です。さらに複数の独自技術を使ってデバイス内のノイズを検出し、それに応じてシステムを調整します。

ノイズの挙動 : 電圧ベースの取得

電圧ベースのシステムでは、注入されたノイズによって元のサンプル値に対して正または負のオフセットが発生する場合があります。サンプリング レートと注入されたノイズの高調波が重畳すると、共振が起こる場合があります。共振が発生した場合、注入されたノイズのピークまたは谷の値がサンプリングされます。

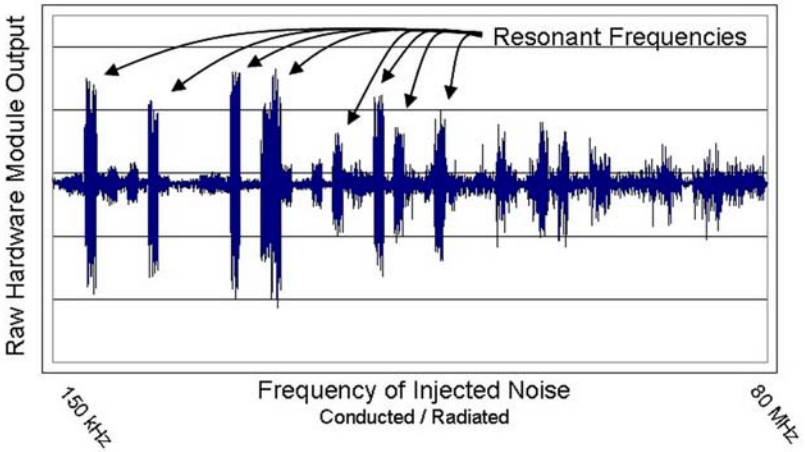
図 4 に、この挙動の例を示します。これらの高調波のいずれかと同じ周波数でサンプリングを実行すると、取得の開始時間によって読み値は全てノイズのピークに重なるか、または中間になります。このため、特定の周波数の複数の読み値が、多くのノイズを示します。図 5 に、この状態を示します。一部のノイズ周波数はサンプリング レートの高調波ですが、その他は高調波ではありません。

図 4: 電圧ベースの取得における高調波の影響



http://www.microchip.com/mla から入手できる Microchip アプリケーション ライブラリは、取得およびフィルタ処理手法を実装しており、センサ出力からこの挙動を除去できます。

図 5: 電圧ベースの取得におけるノイズの周波数応答



信号 / ノイズ比

ハードウェアとソフトウェアの変更がシステムにどのような影響を与えるかを理解するために、信号の現在の性能を計測する必要があります。感度 (システムの変化量) だけでは、システムの安定度を判断する基準として不十分です。例えば、センサの出力平均値が 20,000 で変化量が 2,000 に達するシステムでは、単に読み値から 18,000 を引き、100% (2,000 カウントの変化) の実現を主張できるでしょう。しかし現実には、変化量または「信号」はノイズの量と比較する必要があります。ノイズによりセンサが 1,000 カウント分ドリフトしてしまえば、動作に問題が生じます。

システムの安定度 (システムがどの程度ノイズの影響を受けているか) を見極める最も簡単な方法の 1 つに、システムの S/N 比 (SNR) を見るという方法があります。これは、外乱ノイズと比較した信号強度を計測する方法です。

本書では、式 2 で定義する SNR 計算式を使います。

式 2: 信号 / ノイズ比

$$SNR = \frac{|\mu_U - \mu_P|}{\sigma}$$

μ_U = 押下しない場合の信号強度の平均値

μ_P= 押下した場合の信号強度の平均値

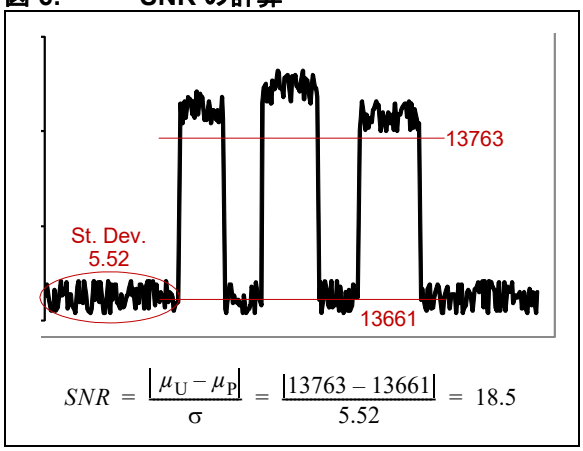
σ= 信号強度の標準偏差

この式の分子は、押下時の変化量または「信号」です。分母は、ノイズが読み値に与える影響の度合いです。これらの比から求めた値でセンサ信号の質を表す事ができ、除去すべきノイズ量に対する信号変化の必要量を判断できます。

伝導ノイズが存在する場合、センサを押下した時としない時で SNR は変わります。これは、注入されるノイズの周波数でも変わります。

以下に SNR の計算例を示します (図 6)。

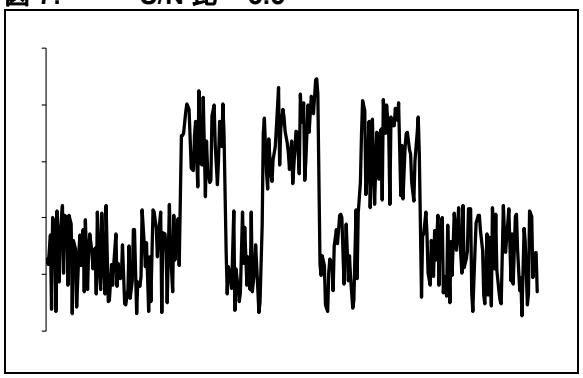
図 6: SNR の計算



システムの SNR を求める式は、この他にもあります。重要なのは、どのハードウェアとソフトウェアの変更が良く、どの変更が悪いか正当に判断できるように、計測間で条件を揃える事です。

図 7 に、式 2 で求める S/N 比が 3.5 の場合にどのような表示になるかを示します。ピークツーピーク値ではなくノイズの標準偏差を使うため、SNR が 3.5 だと確実にタッチを認識させるためのしきい値を設定する余裕がほとんどありません。押下状態＋ノイズと、押下されていない時のセクション＋ノイズを完全に識別できる固定のしきい値を設定するには、SNR が 7 以上のシステムが必要です。実際のアプリケーションでは、高い信頼性を得るためにシステムの SNR は少なくとも 15 とすべきです。

図 7: S/N 比 = 3.5



ハードウェア設計

システム全体の成功には、静電容量式センサ アプリケーションのハードウェア設計が非常に重要です。この段階での決定内容によって、正常に機能する堅牢なアプリケーションを容易に作成できるかどうかが決まります。本ハードウェア設計ガイドラインに従うと、センサの感度を向上させ、業界標準のノイズ規格に合格させる事も非常に簡単です。一方、ガイドラインに従わなければ、合格するのが非常に難しいか不可能となります。どのガイドラインに従って設計するかを決定する際は、この点を忘れないようにしてください。

平行平板静電容量の式

ハードウェア設計で最も重要な点は、「ハードウェア設計時の仕様とその結果得られるシステムの感度との関係は、基本的な静電容量の計算式 (式 1) で決まる」という事です。

例えば、指とセンサの間隔が半分になると感度は 2 倍になります。また、センサの面積が 2 倍になると (指の押下面積よりも小さい限り)、感度も 2 倍になります。

静電容量式タッチセンサのもう 1 つの重要な特徴は、センサのベース静電容量 (C_{BASE}) を決める寄生容量の存在です。式 3 に、C_{BASE} がシステムの感度に与える影響を示します。計測できるのはセンサの総静電容量 (C_T) のみのため、C_{BASE} の影響が大きいほど、指のために生じる静電容量の変化 (C_F) は分かりにくくなります。

このシステムの概略図は図 1 に示した通りです。

式 3: センサの総静電容量

$$C_T = C_{BASE} + C_F$$

C_T= 総静電容量 (計測値)

C_{BASE}= センサのベース静電容量

C_F= 指の静電容量

式 1 と式 3 は、静電容量式タッチのハードウェア設計ガイドラインの基礎です。これらの式は単純に物理的現象に基づくものです。これらのガイドラインはシステムの SNR を最大にするための推奨事項であり、できるだけ従う必要があります。アプリケーションによっては、全てのガイドラインに従う事ができない場合もあります。例えば、システムにサイズの制約がある場合や、損傷を防ぐために厚いカバーを使う必要がある場合です。そのような場合、特に注意を払い、その他の推奨事項に従う事で良好な S/N 比を確保する必要があります。

設計目標

mTouch または RightTouch ソリューションを使った静電容量式センサシステムの性能を最適化するため、以下に留意して設計する必要があります。

- ノイズに対する静電容量 (C_F) の変化を大きくする。
- センサのベース静電容量 (C_{BASE}) を最小化する。
- Metal Over Cap システム以外では導電性のカバー材を使わない。
- カバーをできるだけ薄くする。

PCB 設計における注意事項

基板材料 – 特殊な要件なし

層の厚さ – 特殊な要件なし

推奨する 2 層 PCB 積層構造を以下に示します。

- **第 1 層** (上): 静電容量式センサパッドと、第 2 層でトレースしきれないセンサトレース
- **第 2 層** : 全ての部品、LED 信号トレース、電源トレース、通信トレース

推奨する 4 層 PCB 積層構造を以下に示します。

- **第 1 層** (上): 静電容量式センサパッド
- **第 2 層** : 静電容量式センサトレース
- **第 3 層** : グランドプレーン (静電容量式センサパッド直下を除く)。グランドプレーンはできる限り切れ目なく連続させる。静電容量式センシングコントローラ直下の領域は特にこれを守る
- **第 4 層** (下): 全ての部品、LED 信号トレース、電源トレース、通信トレース

5 層以上の PCB では、静電容量式センサトレースは常に静電容量式センサパッド層に近い層にトレースし、GND またはガード層を静電容量式センサトレースとその他の信号層の間に配置します。

ボタンパッド設計における注意事項

形状 – 特殊な要件なし

寸法 – 15x15 mm (0.6”x0.6”) を推奨

パッド間距離 – 10mm (0.4”) またはカバー厚さの 2 ～ 3 倍を推奨

ボタンパッド形状

mTouch および RightTouch 静電容量式タッチセンサのボタンは、最も良く使われる形状である正方形、長方形、円、楕円を含め任意の形状で正常に動作します。長方形または楕円のセンサパッドを設計する場合、縦横比は 4:1 未満にする事を推奨します。

ボタンパッド寸法

式 1 では、「A」は指とセンサが重なる面積と定義しています。これは、静電容量式タッチ アプリケーションの場合、最も小さい静電容量式プレートが制約事項となる事を意味します。センサが指の押下面積よりも小さいと、センサの面積が制約要因となります。センサが指の押下面積よりも大きいと、指が制約要因となります。

ユーザの指のサイズを変える事はできませんが、センサのサイズを調節して感度を最大に高める事はできます。センサが大きいほど、センサのベース静電容量値は大きくなります。これは感度を低下させ、ユーザがボタンを押下した際、より多くの伝導ノイズがシステムに注入される原因となります。また、センサが小さいほど、ユーザの指のサイズではなく、センサのサイズが感度の制約要因になる事が多くなります。このため、最適なセンササイズは指の押下面積と概ね同じと言えます。

最適な方法 : センササイズを、平均的なユーザの指による押下面積 (15×15 mm) と同等にします。

方法 2: センサを、最適なサイズよりも小さく設計する場合

- 指とセンサが重なる面積 (式 1 の A) が小さくなるため、最大感度が下がります。
- センサのクロストークを抑えるため、センサ間を離す事が重要です。
- 感度を高めるために薄いカバーを使います。

方法 3: センサを、最適なサイズよりも大きく設計する場合

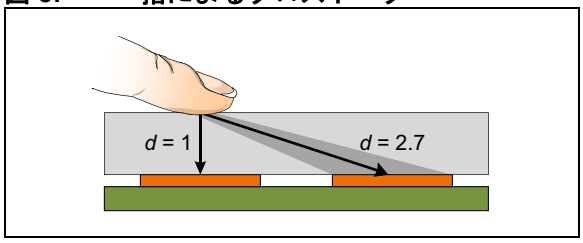
- センサがグランドに接近するとベース静電容量 (式 3 の C_{BASE}) が大きくなる場合があります。
- この場合感度が下がり、伝導ノイズによる外乱が増大します。
- 大きい指より小さい指の方が、指とセンサが重なる面積 (式 1 の A) が小さいため、信号の変化量も小さくなります。そのため、押下による変化量のばらつきが大きくなります。
- 近接センシング能力が向上します。

パッド間距離

静電容量式タッチ アプリケーションでカバーが厚過ぎる場合またはセンサ同士の間隔が近過ぎる場合、クロストークが大きな課題となる場合があります。クロストークとは、押下するセンサとは別のセンサで生じる、望ましくない信号変化の事です。アプリケーションでクロストークが問題になっている場合、ソフトウェアで 2 つのセンサからの信号値を比較し、どちらが実際に押下されているのかを判断する必要があります。しかしこの作業はデコード処理のステップを増やし、エラーが生じる可能性を高め、(実装方法によっては) マルチタッチの実装を不可能にします。本ガイドラインに従えば、センサの信号値をその他全てのセンサ信号値と比較する必要のあるシステムを実装せずに済みます。

図 8 に、センサの押下が周囲のセンサにどのような影響を与えるかを示します。センサ同士の間隔をカバーの厚さの 2 ～ 3 倍にする事で、指とセンサ間のカップリングの強度を小さくできます。距離の変数 (式 1 の d) に焦点を当てる方法もあります。図 8 に示すように、センサ同士を離す事でクロストーク押下は非常に厚いカバーを通しての押下のような状態となります。こうする事でシステムからのクロストーク応答が低減し、有効信号が大きくなります。

図 8: 指によるクロストーク



設計に悪影響を与え得るクロストークのもう1つの形態として、センサ間のカップリングがあります。図 9 に、静電容量式センサからの電気力線の放射を示します。電気力線が隣接するセンサに与える影響は、伝わる距離と伝わる媒体で決まります。

図 9: センサの理想的な電界

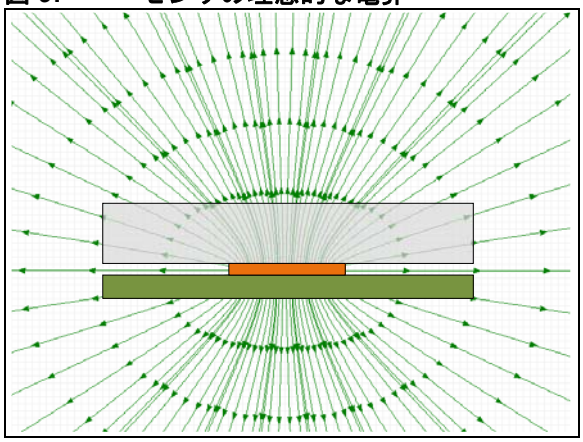
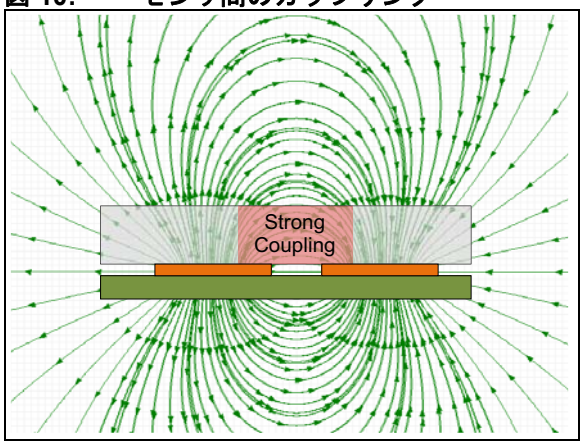


図 10 に示すように、力線がカバー内のみを伝播する場合、影響は大きくなります。力線が隣接するセンサに影響する際に、カバーを通して自由空間に出てさらにカバーに戻る場合、クロストーク量は大幅に低減されます。これを、強いカップリング力線と弱いカップリング力線の違いとして図中で示します。最初のハードウェア設計ガイドラインに従うと、力線が隣接するセンサに到達するためには一度外の空間を通過する必要があり、センサ間のカップリングによるクロストークはわずかになります。

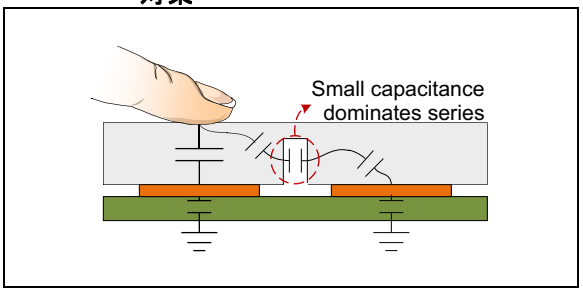
図 10: センサ間のカップリング



または、カバーまたは PCB にエアギャップを設けて、力線が空間を通過する必要があるようにする事もできます。図 11 に方法を示します。ここでは、非常に小さなコンデンサをクロストーク経路に通常の寄生容量と直列に接続したのと等しい事に注目します。この小さなコンデンサは回路内で支配的となり、結果としてクロストークによる変化量は非常に小さくなります。エアギャップが大きいほど静電容量は小さくなり、この方法はより良好に機能します。

最後にもう 1 つ、近くのグランドトレースを使って力線を遮蔽し、センサ感度を制限する方法があります。この手法を使う前に、セクション、「レイアウト設計における注意事項」でセンサ近くのグランドの推奨用法を確認してください。システムの感度を下げるという決定は、他の方法では問題を解決できない場合に最後の手段として使うものです。

図 11: エアギャップによるクロストーク対策



このハードウェア設計ガイドラインに従うと、指とセンサ間のカップリング、センサ間のカップリングを低減できます。その結果、システムで発生するクロストークが非常に小さくなり、処理オーバーヘッドが低下するため応答時間が速くなります。また、センサの信号が悪影響を受けにくくなるため、システムの信頼性が高まります。

最適な方法 : できるだけセンサを離します。

理想的な距離は最低でもカバーの厚さの2～3倍です。

- 指とセンサとの間隔と比べて、センサ同士の間隔 (式 1 の d) が大きくなるため、センサのクロストークが低減します。
- 寄生容量 (式 3 の C_{BASE}) が指の静電容量 (C_F) と比較して小さくなるため感度が高まります。

方法 2: カバー内にエアギャップを設けます。

- センサ間の比誘電率 (式 1 の ε_r) が「1」に下がるため、センサ間のカップリングが低下し、センサのクロストークが低減します。

方法 3: センサ間にガードトレースを設けます。

- ガードは 2 つのセンサ間の低インピーダンスのシールドとして機能します。ガードはスキャンするセンサと同電位であるため、センサの力線はガードと隣接センサから遠ざかります。そのためクロストークの影響が減少します。

アクティブガードの駆動

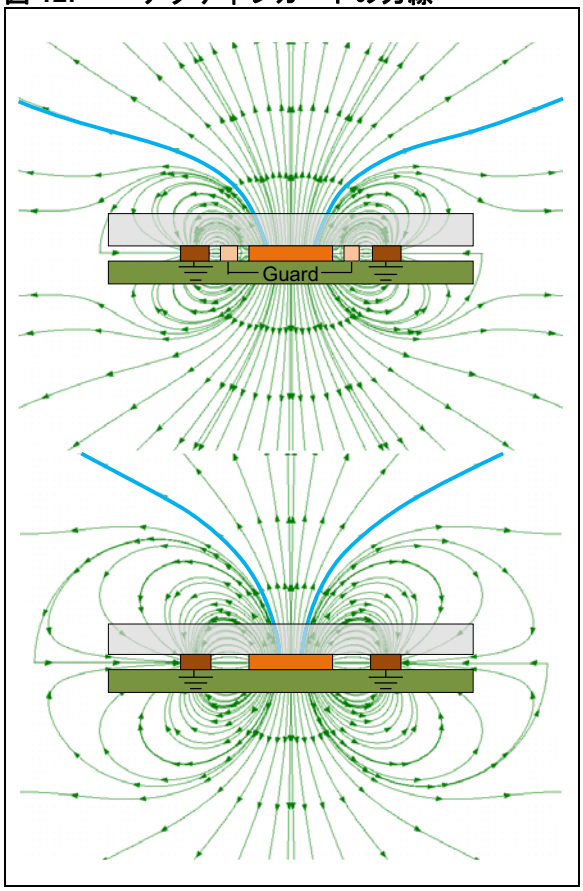
センサの感度はベース静電容量で決まります。この静電容量値は、センサをスキャンする際にセンサ周囲の電位をセンサ電位と揃える事で小さくできます。

ガードをどの程度正確に駆動する必要があるかはmTouch 手法または使う RightTouch デバイスで決まります。しかしこの電圧は波形と完全に一致させる必要はありません。単純にセンサと同期して I/O を駆動するだけでも 60 ～ 70% の効力が得られます。

- 1 つのガードトレースを全てのセンサに使えます。センサは 1 つずつスキャンされるため、他のセンサには影響を与えずに、現在スキャン中のセンサに対するガードをアクティブに駆動する事ができます。
- 電源プレーンまたは低インピーダンスのトレースは全てセンサからガードします。
- センサパッド周囲のガードトレースは厚さを1 mm とし、センサから 2 ～ 3 mm 離します。
- PIC マイコンへのセンサトレースの周囲のガードトレースは、センサトレースと同じ厚み (0.1 ～ 0.3 mm) でかまいません。センサトレースとガードトレースの間隔は 0.5 mm で十分です。

図 12 に、アクティブガードがセンサの力線の形状を変化させ、センサ感度を向上させる様子を示します。

図 12:　アクティブガードの力線



相互駆動

相互駆動とは、相互駆動 (Tx) とセンサ (Rx) 間の比誘電率の変化を計測する意図で mTouch 波形と同期して駆動する各 I/O ピンの事です。

相互駆動センサを実装する場合、ベースとなるカップリング容量とカップリング経路に新規材料を配置した際のカップリング容量の変化分が問題になります。最大感度を確保するには、増加した静電容量による電圧変化が誘電率 (相互カップリング) の変化による電圧変化と同方向になるように、センサ波形と同期して相互波形を駆動する必要があります。

相互駆動信号が設計上役に立つ状況には主に以下の2つがあります。

- 金属片がセンサに接触する可能性がある場合にその金属を相関信号で駆動すると、短絡発生時に生じるセンサの読み値のグリッチを抑止できます

(例えば Metal Over Cap システムの金属層です。これは必須ではありませんが、金属層がセンサと短絡する可能性がある場合には有効です)。

- 検出対象がセンサのグランド基準から絶縁されている場合、センサの近くに相互駆動を配置すると、センサと相互駆動の間の誘電率変化を検出できます。

静電容量式センシングのタイプに応じて、相互駆動の方法は異なります。差動 CVD 波形は、波形と同期して相互駆動する必要があります。しかし CTMU のシングルエンド波形を使うと基板の全てのグランドは相互駆動と同様に機能します。

スライダセンサ設計

スライダは、Microchip アプリケーション ライブラリで入手できる mTouch のフレームワークとライブラリを使って実装できます。またCAP1114 等の RightTouch 静電容量式センシング デバイス上でも実装できます。

図 13 に、代表的なスライダ形状を示します。静電容量式センシングパッドを個別に設計する場合と同様、パッド間の距離は 1.3 mm(約 50 mil) より大きくする必要があります。

理論的には、ボタン用のパッド形状はスライダ用のパッドとして使えます。しかし、図 13 に示す V 字形状にすると、パッド上の指が隣のパッドに移動した際に線形的 (滑らか) な応答が得られます。また、このような形状は回路設計、PCB レイアウト、組み立ての各工程で方向を明確に示す役割も果たします。

図 13: 推奨スライダ設計

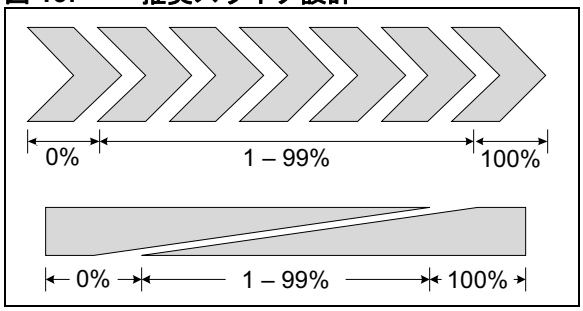


図 13 に示す 7 パッドスライダを使うと、ほとんどのアプリケーションに対して良好な感度と十分な精度が得られます。しかし、アプリケーションによっては 7 パッドより少ないスライダも使えます。パッド寸法が最小要件を満たす事ができず、かつ精度を下げても良い場合、スライダのパッド数を 2 つにまで減らせます。

各パッドの幅とパッド間距離は、通常スライダの全長で制限されます。スライダの高さは、通常機器の寸法で制限されます。

カバー材に関する注意事項

厚さ – できるだけ薄くする事 (理想的には 3 mm 未満)

材質 – ガラス、プラスチック等比誘電率 2.0 ～ 8.0 を推奨

接着剤 – 高誘電率で薄く気泡がない事

ほとんどのアプリケーションでは、静電容量式センサパッドは図 1 に示すようにカバーで覆って保護します。カバーの材質と厚さ、接着剤の種類はシステムの性能に影響します。

カバーの厚さ

カバーの厚さは、静電容量式タッチシステムの感度に非常に大きく影響します。製品の設計者は、最終製品の耐久性を高めるためにカバーをできるだけ厚くしようとしますが、こうするとシステムの感度は低下します。式 1 は、静電容量式タッチ アプリケーションでカバーの厚さが重要である理由を理解する上で役立ちます。PCB と指が離れるほど、予測される静電容量の変化量は小さくなります。

図 14 に、カバーの厚さと感度の関係を示します。この線図で注目すべき点は、カバー材料の誘電率が感度に与える影響です。厚さが同じでも、誘電率の低い材料よりも高い材料の方が感度の変化が大きい事が分かります。ただし誘電率が高いと、クロストーク量が増加するという悪影響があります。カバー材料の導電性が高いと、静電容量式システムは動作しない事があります。

図 8に、指が隣接するセンサに与える影響を示します。カバーの誘電率が高くなるほど、このカップリングも増大します。

厚いカバーで覆う必要がある場合、センサとユーザの指の距離が近くなるように、センサ位置のカバーに窪みを設けます。この窪みに PCB 全体を収める事ができない場合、導電性のスポンジでカバーとセンサ間のエアギャップを埋めます。

最適な方法 : できるだけカバーを薄くします。理想的には、感度を最大にするためにカバーの厚さを 3 mm 以下にする必要があります。

図 15 (a) を参照してください。

方法 2: 最適な厚さよりもカバーが厚い場合、センサの面積を大きくして感度を高めます。

図 15 (b) を参照してください。

方法 3: 最適な厚さよりもカバーが厚い場合、センサが表面に近づくようにカバーの材料を加工します。

図 15 (c) を参照してください。

方法 4: 最適な厚さよりもカバーが厚い場合、EMI ガスケットまたはばねで PCB と指との距離を縮めるようにカバーの材料を加工します。

図 15 (d) を参照してください。

カバーが最適な厚さよりも厚い場合、以下のようになります。

- センサとユーザの指の間隔 (式 1 の d) が大きくなるため、指とセンサ間の静電容量 (式 3 の C_F) が低下し、感度が低下します。
- 図 10に示すように、空気より誘電率の高いカバー内を伝わる力線が増えるため、センサ間のクロストークが増大します。

図 14: カバーが感度に及ぼす影響

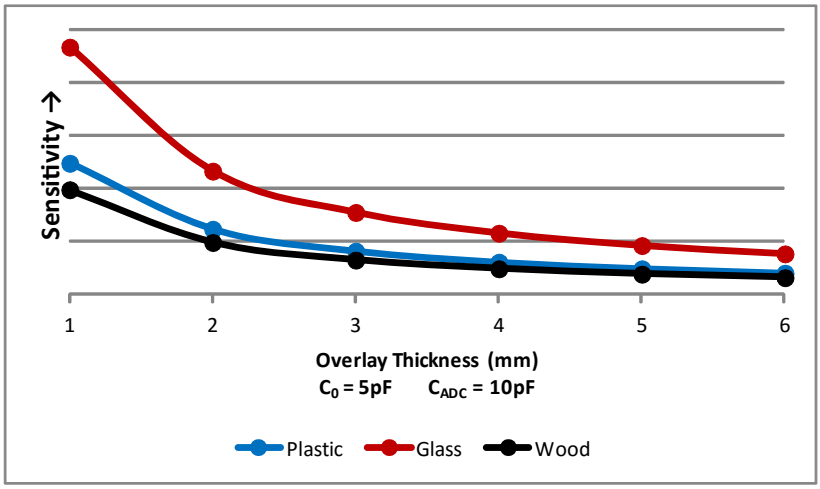
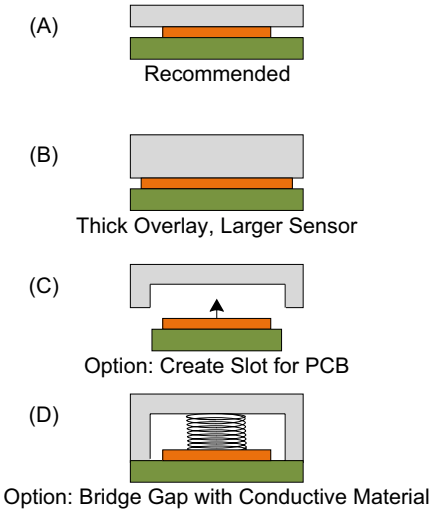


図 15: 厚いカバーを使う場合



接着剤の選択

接着剤はカバーを PCB に固定するために使いますが、これは静電容量式タッチシステムの信頼性にとって重要な要素です。良好な接着の重要性は、式 1 を見ると分かります。空気の比誘電率は約 1 です。プラスチックは通常 2 ～ 3 です。ガラスは約 4 です。カバーと PCB の間に空気があると、実際の比誘電率 ε_r は大きく低下します。例えば 1 mm のエアギャップがあると、ない場合に比べて感度は半分から 1/4 に下がります。コンデンサを直列に配置すると、一番容量の小さいものが支配的となる事を思い出しましょう。

Metal Over Cap 方式では特に、アプリケーションに最適な接着剤を使う事が重要です。これらの設計では、数十μm(10 μm ≒ 0.4 mil)の距離が大きな差を生みます。開発中のアプリケーションにとって最適な接着剤を確実に選択するため、3M 社またはその他の接着剤メーカーの販売担当者に相談する事を推奨します。

接着剤の選択または使用に関する注意点は他にもあります。

1. 感度を高く保つために、接着剤は薄く塗ります。ほとんどの一般的な静電容量式タッチシステムでは、2 mil (50 μm) が適当な厚さです。
2. 接着剤の説明書は必ず読んでください。説明書によっては、確実に接着するために必要な圧力、温度、時間を指定しているものもあります。

3. 接着剤の温度制限を確認してください。環境条件によっては接着剤が剥離し、動作不良となる事もあり得ます。
4. 接着剤を適用する時は、気泡に注意します。接着剤に気泡が混入すると、カバーと PCB の間にエアギャップがある状態と同様に、感度が低下します。
5. 接着剤のタイプがカバーの材料と適合する事を確認します。表面エネルギの高いプラスチックと低いプラスチックそれぞれに対応した各種接着剤が製造されています。ほとんどの接着剤は、ガラスと PCB に問題なく接着します。

適合する接着剤の例を以下に示します。

表面エネルギの高いプラスチックの場合：

例 : ABS またはポリカーボネート

3M 社製 Adhesive Transfer Tape 467MP

表面エネルギの低いプラスチックの場合：

例 : ポリプロピレン

3M 社製 Adhesive Transfer Tape 9626

3M 社製 Adhesive Transfer Tape F-9752PC

3M 社製 Adhesive Transfer Tape 9122

3M 社製 Optically Clear Adhesive (OCA):

8211, 8212, 8213, 8214, 8215

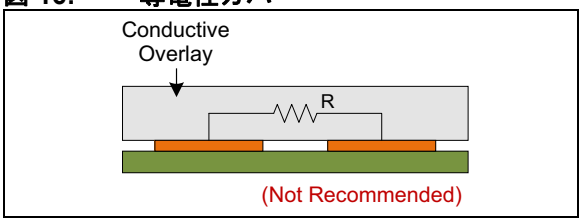
これらは全て PCB とガラスに接着します。

導電性カバー

Metal Over Cap ソリューションを設計する場合を除き、導電性が高いカバーは推奨しません。しかし、静電容量式センサ アプリケーションの中には、非導電性プラスチック上の導電性コーティングまたはカバーの色を黒くするためにカーボンを使ったために、カバーが導電性を持つものがあります。

静電容量式センサが導電性材料で覆われている場合、隣り合うセンサ間の等価抵抗と材料の抵抗が加わる事になります。指がセンサにタッチした場合、そのパッドの静電容量値の変化が、タッチしていない別のセンサに影響して静電容量値を変化させます。カバーの抵抗が小さくなるに従い、タッチしていないセンサピンの静電容量値の変化量は大きくなります。抵抗が小さすぎると、信号に差がなくなり、あるセンサの押下と別のセンサの押下の区別が付かなくなります。その様子を図 16 に示します。

図 16: 導電性カバー



導電性カバーを使う事は以下の理由から推奨しません。

- プラスチック中のカーボンの含有量が経時的に変化し抵抗が変動する場合があります。
- 場所によって抵抗が異なる場合があります。
- カバーの製造ロット (日付コード) で抵抗が異なる場合あります。

以上はいずれも静電容量式センサの入力値に影響を与え、デバイスの設定 (例 : しきい値) を誤らせる原因になります。

設計で導電性材料を使う必要がある場合、常にカバーサンプルを試験して導電率の許容レンジを決定する必要があります。

レイアウト設計における注意事項

以下のガイドラインに従う事で、静電容量式センサの PCB を適切に設計できます。しかし、これらは推奨事項であり要件ではありません。

- LED 出力トレースは、GND またはガードプレーンを間に挟んで静電容量式センサパッドとは別の層に配置する必要があります。

Note:	PCB 上の他のスイッチング信号も全て同様に、静電容量式センサパッド / トレースから離す必要があります。mTouch または RightTouch デバイス以外の信号源で生成した信号も同様です。

- センサから見ると、隣接するセンサトレースは GND として見えます。2 つのセンサのトレースを並べて配置する事は、両トレースを GND に沿って配置する事と等価です。
- 同層または隣接層で、センサトレースと LED 出力トレースを並べてはいけません。センサトレースが隣接層の出力信号を横切る場合、必ず直交するようにします。
- センサのトレース幅 : 0.1 ～ 0.2 mm
- センサのトレース長 : できるだけ短くするかガードする事
- センサの直列抵抗 :
  - CVD : 4.7 ～ 10 kΩ
  - CTMU : 1 ～ 2.5 kΩ
  - RightTouch : 不要
- センサトレースの最小間隔 : 0.1 mm
- センサトレースでは、ビアはできるだけ使わないようにします。ベース静電容量が増大するためです。
- 未使用の RightTouch センサと LED ピンは、プルダウン抵抗で終端するか GND に直接接続する必要があります。

Note:	GND に接続した未使用の LED/GPIO ピンは、制御ファームウェアが駆動する事がないようにします。

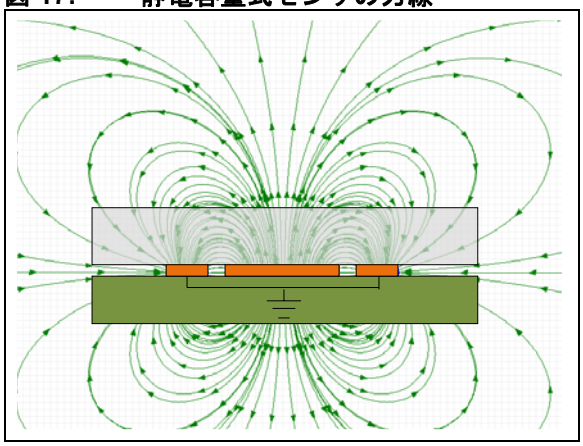
- 静電容量式センシング コントローラの VDD ピンの近くにバイパス コンデンサを配置する必要があります。

静電容量式センサのカップリング

静電容量式センサの力線は、付近のコンポーネントまたは低インピーダンスのトレースとカップリングする事なくパッドから放射するのが理想的です。人体がセンサに近づいた場合、力線が遠くまで達しているほど素早く変化を検出できるからです。実際には常にカップリングが存在し、理想的な力線パターンに何らかのひずみを生じさせます。この挙動を図 17 に示します。

設計では常に、パッドと周囲環境との間に見られるカップリングの程度をできるだけ小さくする事を目標とし、センサの力線ができるだけ理想的になるようにします。

図 17:　静電容量式センサの力線



静電容量式センサトレース

最適な方法 : センサトレースは細く短くします。

これには主に 2 つの理由があります。第一に、トレースが短いと CBACE を最小限に抑える事ができ、システムの感度を向上できます。トレースが長いとアンテナのような挙動を示し、アプリケーションのノイズフロアが上昇します。

通信線はできるだけセンサトレースから離しておく必要があります。それができない場合、通信線とセンサトレースを直交させ、外乱を最小限に抑えます。第二に通信線をグランドトレースでガードし、より敏感な静電容量式センサのトレースではなく、グランドとカップリングするようにする事ができます。静電容量式センサトレースはノイズを発生するトレースと平行に配置しないようにします。また、グランドまたは別の静電容量式センサトレースとは離し、寄生容量が最小となるようにします。

高感度を維持しながらノイズを低減するには、センサトレースを短くします。

静電容量式センサ直列抵抗

CVDおよびCTMUセンサピンに直列抵抗を追加すると、高周波ノイズ環境でのセンサ読み値を安定化できます。抵抗と内部のピンの静電容量がローパスフィルタを形成するためです。抵抗値が増大するに従って、フィルタのカットオフ周波数が下がります。しかし抵抗値が大きくなりすぎると、波形のセトリングタイムが増大し、低周波ノイズが信号に混入しやすくなります。

CTMU を使う場合、モジュールの最大入力インピーダンスである 2.5 kΩ を超えないようにします。これを超えるとスキャンレートが低下します。直列抵抗の推奨最小値は 1 kΩ です。

CVD アクイジション手法を使う場合、代表的な抵抗値は 4.7 kΩ です。しかし、アプリケーション要件に応じて 1 ～ 10 kΩ の間で変わります。ADC 変換開始前にコンデンサを完全に充電できるように、センサ波形のセトリング遅延調整が必要な事もあります。

RightTouch 製品を使う場合、直列抵抗は不要です。

静電気放電保護

Microchip 社の静電容量式タッチセンサは、高レベルの静電気放電 (ESD) にも物理的に損傷する事なく耐える事ができます。さらに、動作時の電磁干渉 (EMI) はハードウェアおよびソフトウェア フィルタ処理を通じて最小化されます。しかし過度な環境条件が原因で誤トリガの生成、内蔵 ESD 保護クランプの導通、VDD とグランドの変動のいずれかが起こり、デバイスがリセットする事もあります。そのため、システム設計プロセスのできるだけ早い段階から電磁環境適合性 (EMC) を考慮する事が重要です。

ESD にはシステムへの侵入経路で 2 種類あります。

1. 基板間の接続を通して侵入する過渡的電荷 (回路設計による対策が必要)
2. カップリングで基板に誘導される過渡的電荷 (レイアウトとシステム上の対策が必要)

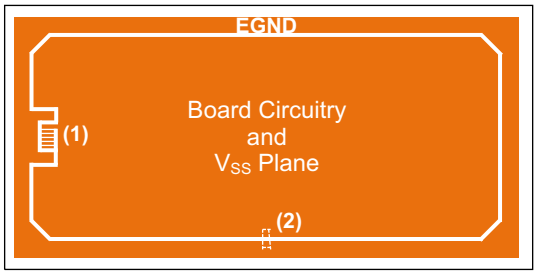
前者に対しては、以下のアプローチが問題解決に役立ちます。

- VDD およびグランドライン上の ESD 保護素子 (例 : 直列抵抗、フェライトビーズ、コモンモード チョーク) を挿入して高周波でのインピーダンスを増大させます。
- これらの ESD 保護素子を通信線に挿入します。
- VDD とグランド間に、ESD 電流を逃がす過渡電圧抑制ダイオード (TVS、アバランシェ ダイオードとも呼ぶ) を追加します。

後者に対しては、以下のアプローチが問題解決に役立ちます。

- ESD 電荷の侵入経路を特定する必要があります。カバー材料におけるエアギャップ、2 つのカバー材料の合わせ面、カバー材料のエッジ部分等が可能性として考えられます。
- ESD 電荷を逃がすために、PCB 上に ESD グランド (EGND) として専用のメタル層を設ける事も有効です。ESD 電流をシャシーに逃がすため、EGND としてリング状のメタルパターンを PCB の最外周に沿って設けます。EGND は、できれば導電性スポンジを使ってシャシーに直接接続します。EGND リングは PCB の全層に設ける必要があります。
- EGND とその他の全ての PCB トレースの間に、これらのトレースから 0.5 ～ 1 mm (20 ～ 40 mil) 離して信号グランドを配置します。

図 18: GND トレース例



(1) Example connector:
- Shield connected to EGND
- V_{SS} pin connected to board's V_{SS} plane

(2) 0.1µF coupling capacitor

EGND は、約 0.5 mm (20 mil) の間隔で基板のグランドから絶縁された帯状のメタルトレースであり、0.1 µF のコンデンサでシステムのグランドと AC 結合しています。

以下の点に注意が必要です。

- シャシーグランド等システムの安全な領域にグランド ストラップまたは同様の方法を通じてエネルギを導く事で、センサデバイス ピンへの ESD の直接カップリングを最小限にする必要があります。
- プラスチックのみで作られたものより、金属製のカバー材料と表面は ESD に対する保護が困難です。また、プラスチックは「気中放電」ESD 準拠 (IEC 61000-4-2) のみを必要とするのに対し、金属性カバーおよびボタンは「直接接触放電」に準拠する必要があります。通常、金属部品が露出した筐体内よりプラスチック製筐体内の方が、容易に安全な ESD 環境を作れます。
- 別の基板、特に同一カバー上に設置され静電容量式センシング基板に直接接続された基板は、ESD の発生源になり得る事に注意が必要です。

  例 : 機械式電源ボタン基板

  通常、これらの基板は静電容量式センサ PCB と同じ V_{DD} を使います。2 つの基板の電源を値の小さい抵抗 (約 50 Ω) で分離する事を推奨します。
- 付近の金属製シャシー表面からの ESD エネルギの放射は、カップリング要因として考慮する必要があります。このタイプのカップリングを緩和する方法は、薄い絶縁層の上を導電性の層で覆ったテープで、シャシー表面とボタン基板層のどちらかをシールドする事です。
- ESD エネルギを機械的方法で逃がせない場合、TVS ダイオードを V_{DD}、MCLR、長い LED トレース (特に基板から出ていくトレース) に接続する方法も有効です。これらの TVS デバイスの保護電圧は V_{DD} と一致させる必要があり、基板上の配置は ESD 発生源と静電容量式センサ コントローラの間とするのが理想的です。
- 全層の基板接地とプラスチック製筐体の導電性被膜は、EMI の影響を最小化する最良の方法です。
- コントローラピンのできるだけ近くのセンサトレース上に小容量 (5 ～ 15 pF) のバイパス コンデンサを配置すると、過剰なエネルギをデバイス内に入れずにグランドに逃がすのに役立ちます。

Note: 他の手法では十分な ESD 保護が得られない場合を除いて、センサのバイパス コンデンサは推奨しません。この手法はセンサの感度を制限する場合があり、近接センサでは使えません。

電源に関する注意事項

安定してノイズが少ない電源を使うと、静電容量式センサの読み値も安定して雑音が少なくなります。設計に用いる電源を評価する場合、以下の項目を考慮する必要があります。

接地経路

人体が静電容量式センサに接近する事で影響を受けるカップリング経路として、以下の 3 つがあります。1 つ目は、指が近づく事で静電容量式センサと基板のグランドとの間のカップリングが変化するものです。これは局所的な現象で、電源または基板レイアウトによる影響をそれほど多く受けません。

2 つ目のカップリング経路は、センサとアース接地を人体を通して接続するものです。これはセンサの静電容量を増加させ、また伝導ノイズ注入の主な経路の 1 つです。人体が基板とグランドを共有していない開回路の場合、この経路で増加した静電容量は無視できます。

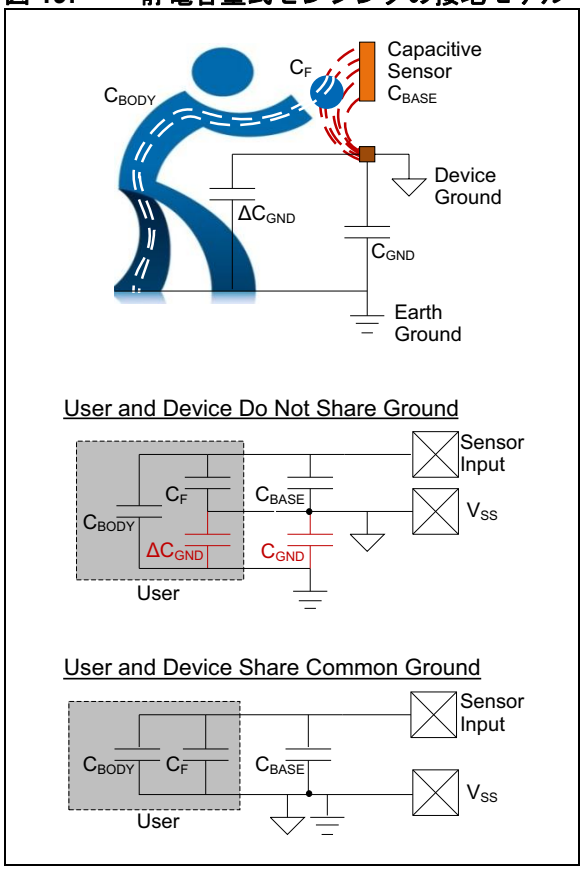
3 つ目のカップリング経路は、人体が基板のグランドに追加する、アース接地に対する静電容量の増加分です。基板とアース接地間の静電容量が増加すると、センサとアース接地間のカップリングがセンサに及ぼす影響は大きく見えます。

図 19 に示すこのモデルは、高品質の静電容量式システムを最適設計するための非常に重要な結論を導きます。

1. 人体とセンサの間に共通グランドを設ける事で最も高い感度を得る事ができます。
2. 共通グランドを設ける事ができない場合、人体とアース接地間の経路の静電容量を最大にする必要があります。

共有グランドシステムでは、人体とグランドを共有しないシステムと比較して 2 倍程度の感度が得られます。

図 19: 静電容量式センシングの接地モデル



ノイズ耐性を最大化するための VDD の選択

最適な方法：ノイズ耐性を最大とするため、VDD はできるだけ高くします。

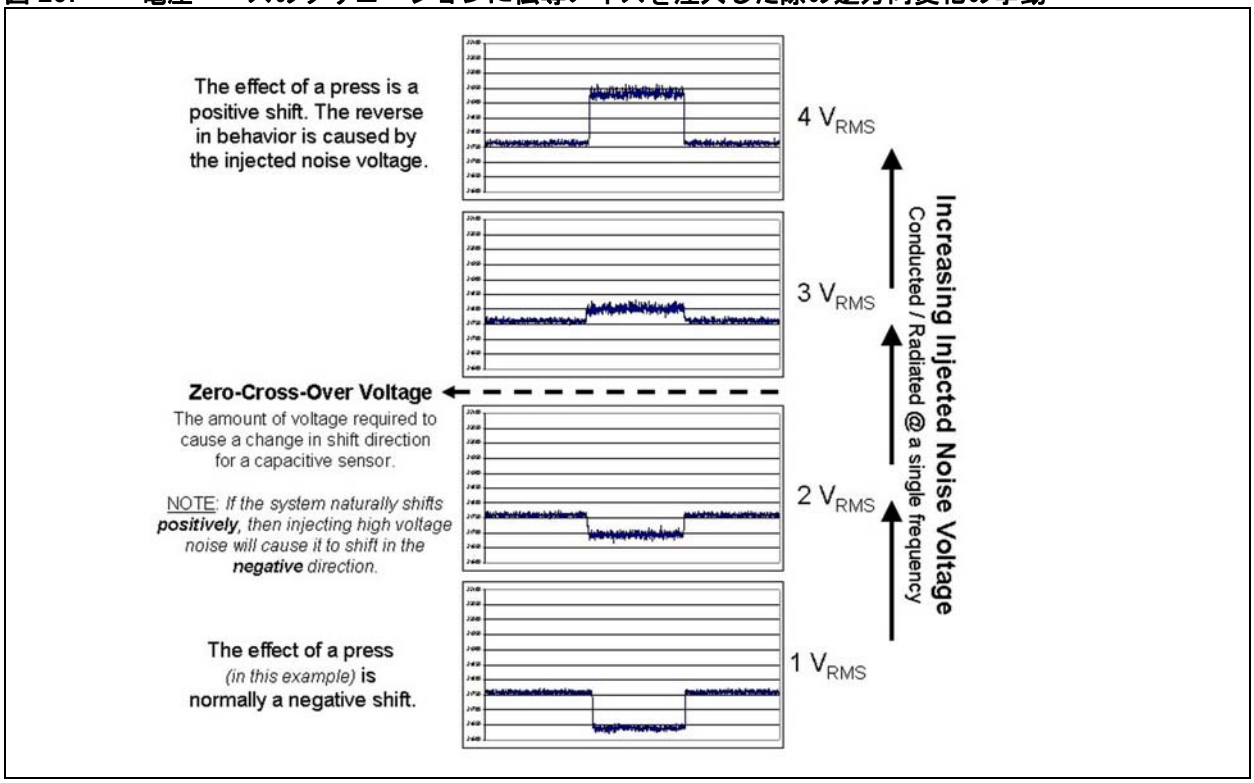
VDD を高くすると消費電力が増えますが、伝導ノイズが多い環境ではノイズ耐性が高くなります。これは、ノイズの電圧レベルの上昇に伴い、最終的には全ての静電容量式タッチシステムが注入ノイズで飽和するためです。VDD をより高い値にすると、この状態を引き起こしてしまう注入ノイズの電圧レベルを高くする事ができます。

電圧ベースのソリューションでは、この挙動はリバースプレス現象として出現します。図 20 に、各種電圧レベルのノイズを注入した標準的なセンサの挙動を示します。ノイズは読み値に対して電圧を追加し、最終的に通常の静電容量式センシング挙動を圧倒します。このため、センサ押下時の変化が逆方向となります。

バイパス コンデンサ

グランドへのバイパス コンデンサ (約 0.1 µF) を、コントローラの各 VDD ピンの近くに少なくとも 1 個接続する必要があります。小容量の小型パッケージのコンデンサを別途並列に配置する事で、追加のフィルタ処理も可能です。

図 20:　　電圧ベースのソリューションに伝導ノイズを注入した際の逆方向変化の挙動



ソフトウェアの手法

本セクションで説明するソフトウェア手法は、mTouch フレームワーク、mTouch ライブラリ、RightTouch 静電容量式センシング コントローラで実装済みの手法の一部です。Microchip 社の静電容量式センシング ソリューションを使っている場合、これらを新たに実装する必要はありません。

システム要件の検討

システムには大きく分けて 2 つの制限があります。制限を決定する 1 つ目の要因は、ハードウェア設計上の決定事項により、アプリケーションの基本 SNR が決まる事です。基本 SNR が高い場合、ソフトウェアはそれほど信号にフィルタをかける必要はなく、検出処理も迅速で簡単です。

基本 SNR が低い場合、ソフトウェアによる信号フィルタ処理が複雑となります。また、誤検出を避けるために検出処理により多くのステップが必要です。制限を決定する 2 つ目の要因は、アプリケーションの性能要件です。応答時間とメモリ容量が限られている場合、適用できないソフトウェア手法があります。

例えば、ゲーム機器で最も重要な要件は速度です。ソフトウェア フィルタ処理の手法を選択する際は、応答時間に大きく影響しないように注意する必要があります。実行速度を上げるためにコードサイズを小さくする必要があり、場合によってはシステムのサンプリング レートを上げる必要があります。

本セクションで説明するソフトウェア手法を検討する際は、どれを選択しても処理時間、消費電力、使用メモリ容量が増える事に注意します。常にメリットとデメリットを検討する必要があります。

サンプリング レート

静電容量式タッチ ファームウェアを作成する場合はまず、メインループから新規スキャンをトリガするか、または割り込みサービスルーチン (ISR) からトリガするかを決めます。ノイズが懸念されるアプリケーションでは、割り込みサービスルーチンを使ったトリガを推奨します。

フィルタが正しく動作するためには時間固定のサンプリング レートが重要であり、新しい挙動が実時間でどのくらいの間計測され続けているかに基づいて押下を検出する必要があります。システムのサンプリング レートがメインループからの関数コール等、固定的でない時間間隔に基づく場合、システム内のその他のアプリケーションがサンプリング レートに影響を与えてしまう場合があります。例えば、電源の出力電圧をアクティブに制御しているシステムはタイミング要件が特別であり、mTouch センシングよりも優先度が高いはずです。電源制御アプリケーションが mTouch センシングに定期的なスキャン実行を許可しない場合、システムが押下/解放の検出に失敗する可能性があります。

必要なもう 1 つの判断は、固定サンプリング レートをいくつにするかという事です。これは、システム固有の要件だけではなく、選択した取得手法も大きな決定要素となります。

電圧ベースの取得では、頻繁かつ非常に高速にスキャンを実行しますが、周波数ベースの取得では、比較的長い時間をかけて低速レートでスキャンを実行します。非常に高速な応答を必要とするゲーム機器では、1 秒間に 100 回を超えるスキャンを実行する場合がありますが、バッテリ駆動の近接センサでは、近くにユーザがいる事を検出するまでは 1 秒間に 3 回しかスキャンを実行しない場合があります。

ノイズが懸念される多くのシステムでは、サンプリング レートとデコードレートが異なる場合があります。例えば、フィルタを常時更新しながら 50 µs に 1 回ずつセンサをスキャンできるシステムで、デコード シーケンスを 10 ms に 1 回しか実行しない場合があります。こうする事で、システムが変化する環境に常に適合できるようにしたまま、処理オーバーヘッドを抑えられます。

ソフトウェア フィルタとデコード アルゴリズムは、センサのサンプリング レートを考慮しながら設計する必要があります。そうしないと、フィルタが高速になり過ぎてごくわずかなノイズ低減に苦しめられたり、低速になり過ぎて信号を減衰させたり遅延させたりてしまう場合があります。システムのサンプリング レートを考慮していないデコード アルゴリズムは、応答時間に問題があったり、状態変化 (ON/OFF) が早過ぎたりする場合があります。

サンプリング レートのジッタリング

電圧ベースの取得手法のみ

割り込みサービスルーチンを使って新規スキャンをトリガすると、スキャンがサンプリング レートの高調波で注入されるノイズに弱くなるという問題があります。ジッタリングを使うと、放射 / 伝導ノイズのどちらかの形でシステムに注入される高周波ノイズを減衰させる事ができます。図 13 に、この例を示します。

最も簡単なソリューションは、ジッタリングを使って、サンプルを取得するたびにごくわずかにサンプリング レートを変更する事です。例えば、400 µs ごとにセンサをスキャンしている場合、読み出しを 0 ～ 10 µs ずつ遅らせて (この遅延量は毎回変化させる)、高調波に重ならないようにします。厳密に言うとこの方法ではサンプリング レートが変わってしまいますが、平均サンプリング レートは変わらず、またサンプリング間隔全体と比較するとその変化もごくわずかです。

次に示すジッタリング実装 (例 1) では、直近の ADC サンプル最下位ビットを使って、小さな遅延をランダムに生成しています。値を `0x0F` でマスクして最大遅延を制限しています。

例 1:　　サンプリング レートのジッタリング

```
void interrupt ISR()
{
    if (T0IE & T0IF)
    {
        // Short Delay
        jitter = ADRESL & 0x0F;
        while(jitter--);

        mTouch_Service();
    }
}
```

周波数ホッピング

周波数ベースの取得手法のみ

これは周波数ベースの取得手法であり、電圧ベースの取得手法におけるサンプリング レートのジッタリングと同等です。高調波ノイズの問題を回避するために波形のサンプリング周波数を変える事は重要です。信号のオフセット値を一定に保つために、ホッピング手順は各サンプリング周期で等しい事が必要です。

RightTouch 静電容量式センシング コントローラは、システムのノイズレベルに基づいて自動的にこの手順を実行します。

オーバー サンプリング

オーバー サンプリングは、1 回の「センサ読み出し」に複数のサンプルを使う処理です。例えば、ほとんどのシステムでは割り込みサービスルーチンにより新規読み出しの実行タイミングが決まります。この時システムは値を取得し、その値を新規の読み値として保存します。読み値の安定度を高めるには、同じセンサを 2 回スキャンして 2 つのサンプルを加算 / 平均化して、1 つのセンサ読み値を生成する、という方法が考えられます。

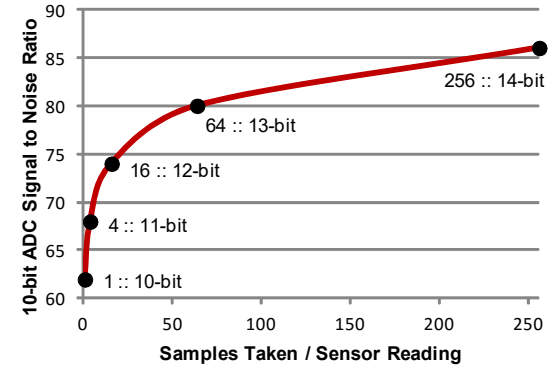
Note:	「サンプル」とは、ハードウェア モジュール (例 : ADC) を使ったセンサのスキャンを指します。「読み値」とは、加算 / 平均化されフィルタ処理 / デコード処理ルーチンに送信される一連のサンプルを指します。

この方法が有益である理由は複数あります。

1. 個別のサンプルは読み値の一部に過ぎないため、インパルスノイズによるサンプリング エラーはシステムに影響を与えません。すなわち平均化処理により、取得処理中に一定のエラーを効果的に排除できます。
2. サンプルには小数はありません。複数回のスキャンから 1 つの読み値を生成する事で、信号の分解能を高める事ができます。

図 21 に、システムの SNR を向上させるこの方法の利点を示します。システムの時間と消費電力の要件に対応するため、オーバーサンプリングの応答を明瞭にして数を減らす事を考慮する必要があります。

図 21:　　オーバーサンプリングのトレードオフ : 「サンプリング数」対「SNR」



グラフのラベルは、より高い分解能を持つ ADC と等価な SNR を得るのに必要なサンプル数を示しています。

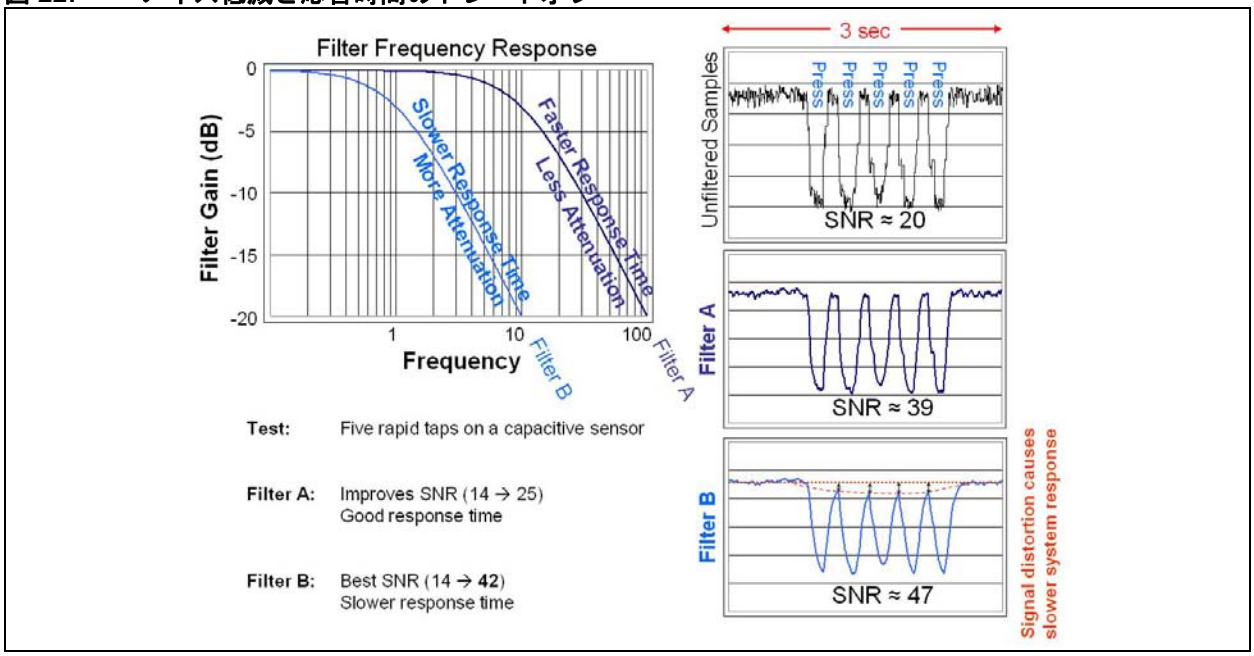
A/D 変換におけるオーバーサンプリングの効果の詳細は、Microchip 社ウェブサイト (http://www.microchip.com) で入手できるアプリケーション ノート『AN1152 - Achieving Higher ADC Resolution Using Oversampling』を参照してください。

ソフトウェア フィルタ処理

フィルタとは、入力信号を取り込んで修正後出力するアルゴリズムです。フィルタの機能はフィルタタイプを基に決まります。また、フィルタ処理性能も、フィルタの機能に大きく影響します。ファームウェアの観点からは、フィルタの機能はコード構造と実行する処理で決まります。通常、帯域幅は実装内の定数 (除数、左右へのビットシフト回数、結果に適用する乗数を設定) で決まります。

フィルタの設計では、ノイズの低減と応答時間はトレードオフの関係にあります。図 22 に、このトレードオフの関係を示します。

図 22:　　ノイズ低減と応答時間のトレードオフ



フィルタの帯域幅が狭いと、通過するノイズは少なくなりますが、フィルタが信号に追従するのに長い時間を要する事があります。一方でフィルタの帯域幅が広いと、通過するノイズは多くなりますが、フィルタはより短い時間で信号に追従します。複数のフィルタを組み合わせると、悪影響を抑えながら各フィルタの利点を生かす事ができます。

mTouch のアプリケーションで一般的に使うフィルタには 3 つのタイプがあります。この他にも使えるフィルタは数多くあり、アプリケーションによってはそれらのフィルタの方がより適している場合があります。ここに記載したフィルタは、ほとんどの設計で複数を組み合わせて使えるものです。

以下に、それら 3 つのフィルタタイプを示します。

1. **スルーレート リミッタ**

   インパルスノイズを除去して信号を平滑化するために適用する、最初の入力フィルタとして使います。

   取得ルーチンに実装します。

2. **L 点移動平均フィルタ**

   各センサに対して、デコード処理中の基準点となる低速で更新される (高時定数) ベースライン (平均値) を作成するために使います。これにより温度、湿度等の環境変化を追跡できます。

   フィルタ処理ルーチンに実装します。

3. **ローパス Butterworth フィルタ**

   高速な応答を維持しつつ (低時定数)、センサ読み値からホワイトノイズを除去するために使います。

   フィルタ処理ルーチンで、デコード アルゴリズムに送信する前の「読み値」変数に実装します。

「FIR フィルタ」対「IIR フィルタ」

有限インパルス応答 (FIR) フィルタは、一定数の過去の入力値から出力値を生成します。

有限インパルス応答フィルタには、以下の利点があります。

- 実装が簡単である。
- 比較的安定している。
- 整数の精度に関する要求が厳しくない。

無限インパルス応答 (IIR) フィルタは、入力値と過去のフィルタ出力値を組み合わせて、次の出力値を生成します。

無限インパルス応答フィルタには、以下の利点があります。

- メモリ使用量が少ない。
- 処理が軽い。

いずれにしても、どちらのフィルタタイプもアプリケーションに役立ちます。静電容量式タッチシステムでは、IIR フィルタは更新速度が低い環境条件のベースラインに使えます。このフィルタは、不安定にならずにインパルスノイズを処理できる事が必要です。FIR フィルタは、現在の読み値等の更新速度が高いセンサ変数に使えます。

フィルタ : スルーレート リミッタ

スルーレート リミッタ (SRL) フィルタの主な目的は、センサ読み値からインパルスノイズを除去する事です。デシメーション フィルタとも呼ばれる SRL フィルタの実装には特別なスキャン方法が必要であり、これによりサンプリング レートが変わってしまう場合があります。

SRL フィルタのコンセプトは単純です。PIC マイコンが各センサの「現在の読み値」変数を保持します。一般的なシステムでは、新しいセンサ読み値が生成されると、「現在の読み値」変数は新しい値で置き換わります。SRL フィルタを実装したシステムでは、新しい読み値が生成されると、最新の読み値が「現在の読み値」変数よりも高いか低いかに従って「現在の読み値」を 1 ずつインクリメント / デクリメントします。例えば、センサの現在の読み値が 200 で、次の取得結果の値が 300 の場合、システムは「現在の読み値」を 201 に更新します。システムが現在の読み値である 300 に到達するためには、続く 99 回のスキャンが現在の読み値よりも高い値である必要があります。

この挙動は、各サンプルの影響を制限するため、非常に便利です。インパルスノイズがシステムに影響を与えている場合、インパルスノイズの影響を受けた 1 つの読み値は、読み値変数にわずか 1 ビット分のノイズしか生じさせません。一方で、「現在の読み値」変数の変化がゆっくりであるため、サンプリング レートを高く設定する必要があります。ユーザがセンサを押した時、「現在の読み値」変数は、指の静電容量に従って素早く変化する必要があります。なお、読み値は一度に 1 ずつしか変化しないため、各読み出し後にシステムのデコード機能にアクセスする必要はありません。

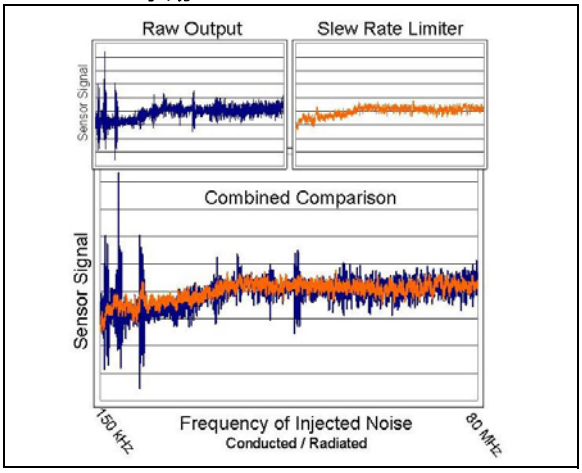
これらの特殊な要件に対処するために、SRL フィルタ実装ではいくつかの点に注意する必要があります。第一に、システムがタイマの割り込みに基づいて素早くスキャンを実行する点です。各スキャンの実行後、システムは SRL フィルタを実行して「現在の読み値」変数をインクリメントまたはデクリメントします。N 回目のサンプリングの後、フラグをセットしてデコード機能を実行します。例 2 に、このフィルタのコード実装例を示します。

例 2: スルーレート リミッタフィルタ

```
#define SCANS_PER_DECODE 100

uint16_t reading;
uint16_t counter;

void main(void)
{
    // Main Loop
    while(1)
    {
        if (counter >= SCANS_PER_DECODE)
        {
            mTouch_decode();
            counter = 0;
        }
    }
}

void interrupt ISR(void)
{
    uint16_t newReading;

    if (TMR0IE && TMR0IF)
    {
        // Take a reading, store the value
        newReading = mTouch_getReading();

        // Initialize
        if (reading == 0)
            reading = newReading;

        // Slew Rate Limiter
        if (newReading > reading) reading++;
        else reading--;

        counter++;
    }
}
```

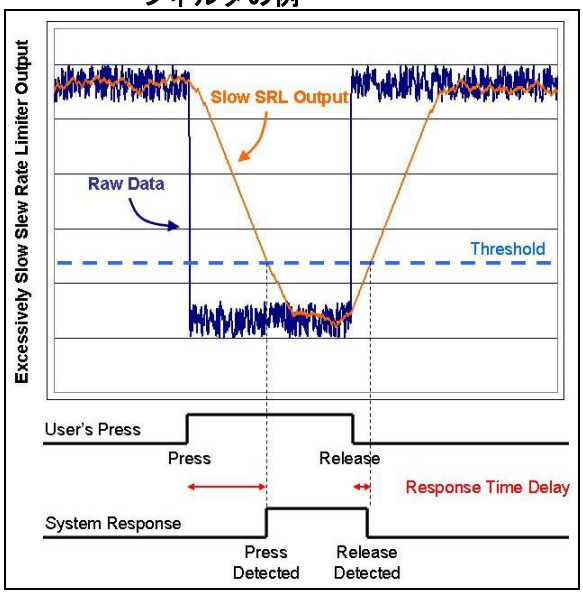
図 23 に、このフィルタの利点を示します。システムのインパルスノイズが除去され、信号のデコードがより容易となりました。

図 23: スルーレート リミッタフィルタの挙動



SRLフィルタの動きが遅過ぎないように注意する必要があります。サンプリング レートが低過ぎると読み値の変化が遅くなり、応答時間に問題が発生します。図 24 に、遅過ぎる挙動の例を示します。

図 24: 遅過ぎるスルーレート リミッタ フィルタの例



この問題を解決するには、以下のどちらかを実行します。

1. タイマ割り込みの間隔を短くします。
2. インクリメントとデクリメントの量を 1 よりも大きい値に変更します。しかしこの値を大きくするほど、フィルタが除去できるインパルスノイズは減ります。

フィルタ : L 点移動平均

このフィルタ方法は非常に多くのアプリケーションで使われており、完全に実証済みです。式 4 に、この挙動の定義を示します。現在値 (x[n]) は、直近の L-1 の値で平均化します。L の値を決定する際は、除算演算に注意します。L の値に 2 の累乗を選択すれば除算演算は一連のビットシフトとする事ができ、複雑なフィルタ処理を軽減できます。

式 4: L 点平均 (FIR)

$$y[n] = \frac{1}{L}\sum_{k=0}^{L=l} x[n-k]$$

y[n]= 時間「n」における出力
x[n]= 時間「n」における入力
L= フィルタのメモリ
k= カウンタ変数

式 4 から分かるように、これは FIR フィルタです。すなわち任意の読み値は一定数のサンプル (L) の出力にしか影響しません。その期間の後、このサンプル (読み値) がシステムの挙動に影響する事はありません。ノイズが注入される状況では、このフィルタ特性は非常に便利です。ただし、このフィルタはメモリ使用量が多いため、ウィンドウを大きくする事は困難です。

式 5 に示すように、FIR から IIR に挙動を変更すればこの制約を受けません。こうすると、フィルタのメモリ使用量を低く、しかも固定できますが、一部のフィルタ機能が低下します。L 点移動平均の IIR バージョンは、L を大きくとるほど元のフィルタから歪んで行きます。

式 5: L 点平均 (IIR)

$$y[n] = y[n-1] + \frac{x[n]-y[n-1]}{L}$$

y[n] = 時間「n」における出力
x[n] = 時間「n」における入力
L= フィルタのメモリ

例 3: FIR L 点平均フィルタ

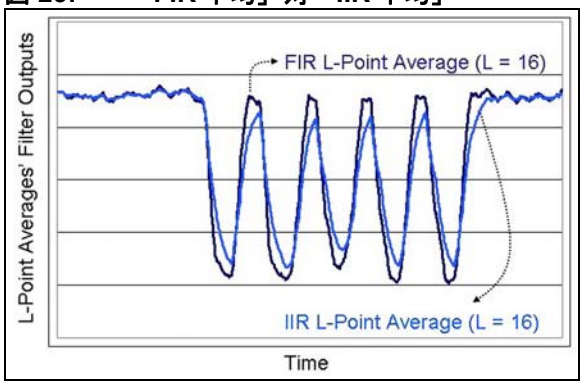
```
#define     HISTORY     8   // 'L'

uint16_t    reading[HISTORY];
uint8_t     index;

uint16_t FIR_Average(uint16_t new)
{
    uint16_t    average = 0;

    // Replace oldest reading
    reading[index] = new;

    // Sum all reading values
    for (uint8_t i = 0; i < HISTORY; i++)
    {
        average += reading[i];
    }

    // Divide by the history window size
    // NOTE:This operation is efficient
    // as long as HISTORY is a power of 2.
    average = (uint16_t)(average/HISTORY);

    // Update index for next function call
    index++;
    if (index >= HISTORY) index = 0;

    return average;
}
```

例 4:　　IIR L 点平均フィルタ

```
#define     HISTORY     8   // 'L'

uint16_t    average;

uint16_t IIR_Average(uint16_t new)
{
    // Update Average
    average -= (uint16_t)(average/HISTORY);
    average += new;

    // NOTE:
    // This filter implementation has a gain
    // of 'HISTORY'.For that reason, we
    // can divide the average by HISTORY
    // or multiply the reading by HISTORY
    // before we compare the two.
    return (uint16_t)(average/HISTORY);
}
```

図 25 に、FIR L 点移動平均と IIR L 点移動平均予測の違いを示します。平均予測ではシステムの応答時間に遅延が生じます。このため、読み出した信号に直接このフィルタを使う事は推奨しません。ただし、このフィルタの遅延時間はベースラインの計算には影響しません。ベースラインの変化は緩慢であるため、遅延時間が挙動に悪影響を与える事はありません。

図 25:　　「FIR 平均」対「IIR 平均」



フィルタ：ローパス Butterworth フィルタ

このフィルタは L 点移動平均と置き換え可能です。両者ともローパスフィルタですが、本セクションで説明するデジタル ローパスフィルタは、Butterworth フィルタのデジタル実装に基づくものです。式 6 に、この定義を示します。このフィルタにおける複雑な演算は、K と直近のフィルタ出力値との乗算のみです。K をうまく選択すれば、このフィルタはビットシフトを使って簡単に実装できます。

式 6:　　デジタルローパス Butterworth フィルタ

$$y[n] = x[n] + x[n-1] + Ay[n-1]$$

y[n] = 時間「n」における出力

x[n] = 時間「n」における入力

A = フィルタの係数 (0 ≤ A < 1)

A の値について、いくつか良い例を挙げると、0.8125、0.8750、0.9375、0.9688 等があります。これらの係数は、y[n-1] の値にビットシフト演算を使うだけで、簡単に乗算できます。以下に、この例を示します。

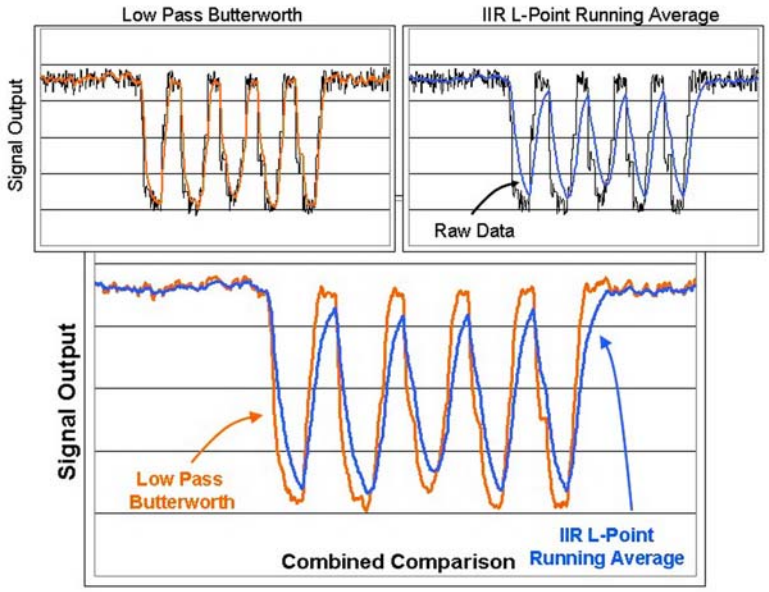
「A」の値が 1 に近づくにつれ、ローパス Butterworth フィルタのカットオフ周波数は 0 に近づきます。こうする事で、フィルタの効果が高まりますが、フィルタのセトリングタイムも若干長くなってしまいます。また、「A」の値が 1 に近づくにつれ、y[n] の整数値も急激に増加します。各センサの信号を標準の 16 ビット整数値で表現し続ける場合、これが「A」に上限を設ける事になります。

図 26 に、L 点移動平均と比較したこのタイプのフィルタの利点を示します。移動平均を実装するよりも Butterworth を実装する方がセンサの実効 SNR は大きく増加します。ただし、移動平均予測の実装は非常に安価であり、センサのベースラインに対するフィルタとして良好に機能します。

例 5: デジタルローパス Butterworth フィルタ

```
#define     FILTER_GAIN     3

typedef struct
{
    uint16_t y;
    uint16_t x;
} filter_t;

filter_t    filter;

uint16_t LP_Butterworth(uint16_t reading)
{
    // Pointer to the correct filter
    // variables for our sensor...
    filter_t* h = &filter;

    // Temporary variables
    uint16_t x1, y1, ay1;
    uint16_t temp0, temp1;

    // Populate temporary variables
    x1 = (h -> x);
    y1 = (h -> y);

    // Calculate:( a * y[1] )
    // Where, a = 0.8125.
    temp0 = y1 >> 2;
    temp1 = y1 >> 4;
    ay1 = y1 - temp0 + temp1;

    // 1st Order Filter Equation:
    // y1 = x[0] + x[1] + (a * y[1])
    y1 = reading + x1 + ay1;

    // Store values for next time
    (h -> y) = y1;
    (h -> x) = reading;

    // Return the filter's new result
    return y1 >> FILTER_GAIN;
}
```

図 26:　　ローパスフィルタの比較



センサしきい値の設定

センサ信号の読み出しとフィルタ処理の完了後、デコード処理を開始します。この段階では、センサの信号と事前に定義されたしきい値を比較し、押下イベントが発生したかどうかを判断します。

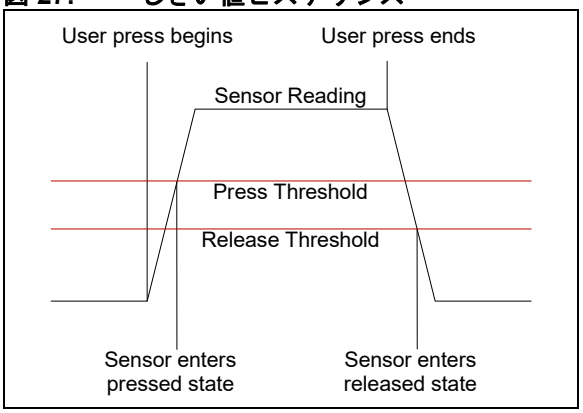
しきい値を設ける場合、固定のしきい値を設定するか、実行時に計算するか、という 2 つの選択肢があります。固定のしきい値は最も簡単で処理時間もメモリも節約できますが、うまく機能しない状況が存在します。量産システムの場合、ボード間でセンサの既定値が若干ばらつく事があります。アプリケーションで固定のしきい値を使った場合、ボードによってしきい値がうまく機能しない場合があります。もう 1 つの選択肢は、実行時にしきい値を計算する方法です。この方法では、システムは起動時に複数の読み値を取得し、これを基にしきい値を設定します。ほとんどのアプリケーションでは、計算したしきい値がシステムの挙動を大きく変える事はありません。また、柔軟性が高いため使用が好まれます。

2 つのしきい値をヒステリシスのように使う事もできます。その場合、一方を押下状態への遷移、もう一方を解放状態への遷移に使います。図 27 に、この挙動を示します。

システムのしきい値の設定は、堅牢なソリューションを設計する上で最も重要かつ最も難しい作業の 1 つです。押下時にセンサ信号が降下するシステムの場合、しきい値が高過ぎると押下の誤検出動作が生じます。反対にしきい値が低過ぎると、押下が検出されない状況が出てきます。

タッチセンサは様々な人が使うという事を覚えておく事が重要です。手の大きい人の場合、手の小さい人よりも押下時の信号変化は大きくなります。どんな人でもセンサ押下イベントを発生させられるようにしきい値を設定し、様々な人でテストします。

図 27:　　しきい値ヒステリシス



包絡線検波

包絡線検波は、追加の変数を使って、センサのベースライン(または平均)からの平均偏差を追跡するデコード手法です。この手法は、大量のノイズが注入されるシステムで特に効果的です。図 27 に、このデコード手法を使うべきシステムの例を示します。

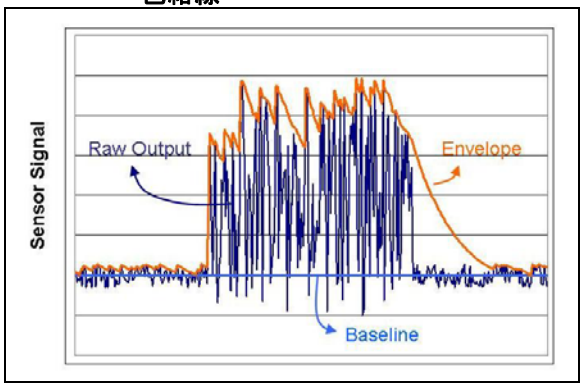
静電容量式タッチセンサに高レベルの注入ノイズが存在する場合、サンプル値の変化を検出するだけのデコード アルゴリズムでは押下の検出が難しい場合があります。複数の周波数のノイズによって図 28 に示すように、押下の挙動が発生する事があります。この現象が発生した場合、エンベロープを生成してシステムのノイズレベルを追跡する事で、しきい値と併用して押下を判断できます。例 6 に、このデコード手法の実装例を示します。

この手法を実装する際は、デルタ値が現在のセンサ読み値と平均値の差の絶対値である点に注意してください。これは、正負どちら向きの変化でも包絡線が増加する事を意味します。この挙動が好ましくない場合、サンプルコードから絶対値のセクションを削除します。包絡線検波デコード アルゴリズムの詳細は、アプリケーション ノート『AN1152 - mTouch™ Conducted Noise Immunity Techniques for the CTMU』を参照してください。

例 6: 包絡線検波

```
// Speeds from Fastest-Slowest:
//          2, 4, 8, 16, 32
#define     SPEED   4

uint16_t envelope;
uint16_t baseline;

uint16_t UpdateEnvelope(uint16_t reading)
{
    int16_t delta = baseline - reading;

    // Absolute Value
    if (delta < 0)
        delta = -delta;

    // Update envelope
    envelope -= (uint16_t)(envelope/SPEED);
    envelope += (uint16_t)(delta);

    return envelope;
}
```

図 28: 高レベルの注入ノイズが存在する時の包絡線



共通の課題

静電容量式タッチシステムの挙動には、共通して出現するものがいくつかあります。ここでは、これらのアプリケーションにおける主な問題点と適用可能な各種ソリューションについて説明します。

ここで扱う課題とソリューションは、以下の通りです。

- クロストーク
- インパルスノイズ
- 無反応なボタン
- フリッカが起こるボタン
- リバース動作

共通の課題 : クロストーク

クロストークとは、隣接するセンサが押された時に発生する、望ましくない信号変化の事です。この現象は主にセンサ同士が近過ぎる場合の弊害ですが、カバーの厚さと比誘電率にも影響されます。

最適な方法 : センサ同士の間隔を広くします。

可能であればソフトウェアによる調整ではなく、ハードウェアの設計手法でクロストークを減らす事を推奨します。厚いカバーを使うシステムではクロストークはさらに大きな問題となるため、カバーを薄くする事が必要かもしれません。

クロストークの原因と、クロストークを低減するためのハードウェア変更の詳細は、本書の**セクション「パッド間距離」**を参照してください。

方法 2: しきい値を調整します。

上述のハードウェア変更による解決策が十分でないか、対象アプリケーションで使えない場合、ソフトウェアの調整がそれに代わる唯一の選択肢です。システム感度が高過ぎる場合、押下の検出に必要な信号変化量を大きくとってしきい値を変更する事が、最適な方法である場合があります。

押下検出に必要なセンサ読み値の変化量をシステムのクロストークによる変化量の最大値よりも大きくする事で、クロストークで押下が誤検出される事を回避します。ただしこの場合、システムノイズがクロストークによる変化量の最大値を増加させ、将来的に誤検出を招く可能性がある事に注意してください。

方法 3:「最も押下されている」センサを検出するアルゴリズムを実装します。

もう 1 つの方法として、各センサの信号変化量をその他のセンサと比較し、どのセンサが「最も押下されている」かを判断するようにシステムのデコード アルゴリズムを変更する事もできます。

このアルゴリズムを実装すると、システムの機能が制限されます。第一の制限事項は、システムは常に一番変化量の大きいセンサのみを押下されたと認識するため、一度に 1 つの押下しかサポートできない事です。この制限による影響を最小限に抑える方法は、隣接するセンサのみと比較を行う事です。第二の制限事項は応答時間です。プロセッサが必要とする追加の実行時間と、デコード処理の前に全センサをスキャンしなければならないという要件により、システムの処理速度が下がります。アプリケーションによって、この点は設計上のトレードオフとして受け入れる事が可能な場合と不可能な場合があります。

共通の課題 : インパルスノイズ

インパルスノイズは、指による押下ではなくノイズを原因とする、前後の読み値とは全く異なる挙動を見せる個別のまたは一連の読み値です。これらのスパイクを適切に処理しないと、システムはすぐに不安定になります。さらに、ノイズスパイクがシステムで頻繁に起こる問題である場合、ノイズスパイクにより実効 SNR が低下する場合もあります。

この問題を解決する方法としては、通常ソフトウェア フィルタが最も簡単な手法です。重要なのは、システムの応答時間に影響を与えずに、スパイクの影響を低減するようにフィルタを調整する事です。例えば、1 つの読み値の影響を最小化するために、平均を取って速度を下げるのみとした場合、システムにおけるノイズスパイクを低減できます。ただしこの場合、非常に低速な平均化に依存する事になるため、応答時間も遅くなります。

最適な解決方法 : スルーレート リミッタ フィルタを実装します。

この場合、無限インパルス応答 (IIR) フィルタは、有限インパルス応答 (FIR) フィルタほど効果的ではありません。IIR フィルタでは 1 つの読み値が (理論上) 永久に信号に影響を与えるため、1 回のノイズスパイクがフィルタを不安定にしかねません。この場合の最適な選択肢は、このアプリケーション ノートのソフトウェアに関するセクションで説明したスルーレート リミッタ フィルタ (デシメーション フィルタとも呼ぶ) を使う事です。このフィルタは、サンプリング レートを高めて応答時間の減少を調整できる限り、挙動全体への影響は最小限に抑えながら全てのインパルスノイズをシステムから除去します。

共通の課題 : 無反応なボタン

無反応なボタンとは、指で押しても状態が変化しないセンサを指します。センサを無反応にする原因と考えられる主なソフトウェア要因は、しきい値とデバウンス値の 2 つです。システムにノイズが注入されると、センサ感度が変化します。これより感度が低下し、押下による信号変化がしきい値を超える事ができなくなる場合があります。システムノイズが増加すると、センサ読み値が状態変化を認識するレンジ外となる事が多くなります。最大デバウンス値が高くノイズレベルも高い場合、センサが全く反応しなくなる場合があります。

この問題を解決する方法は複数あります。

最適な解決方法 : しきい値を調整します。

まずソフトウェアの設定を変更し、押下による信号変化がしきい値を簡単に超えるようにします。システム感度がノイズで低下している場合、この調整でボタンが無反応となる可能性を低くできます。しかし、このようなしきい値の調整が押下の誤検出を生じさせる場合、この方法をとる事はできません。この状況に対応できるように高感度を確保する最善の方法は、このアプリケーション ノートで説明したハードウェア設計ガイドラインを守る事です。

方法 2: 両方向の変化をチェックします。

もう 1 つの可能性として、システムに一定量のノイズが注入された結果、押下による挙動が反転した事が考えられます。これが起こると、センサ押下時に想定とは逆方向に信号が変化します。この問題に対する解決方法は、読み値の両側にしきい値を設ける事です。ただし、この方法が適用できない場合もあります。システムによっては、変化の方向が特定の意味を持っています。例えば一部の耐水システムでは、ある方向への変化は押下を意味し、逆方向への変化は水が存在している事を意味します。両方向の変化を押下として検出すると、アプリケーションの耐水性を失う事になります。

方法 3: ハードウェアの設計変更により感度を高めます。

しきい値の調整では問題が解決しない場合、システムの基本感度を上げるためにハードウェアの設計変更が必要になる場合があります。まず、アプリケーションが全ての設計ガイドラインに従っている事を確認します。次にセンサ面積を大きくし、センサ同士の間隔を広げてクロストークを低減し、カバーを薄くするか素材を変更します。これらの提案は全て、静電容量とハードウェア設計との関係を定義した式 2 が基になっています。

共通の課題：フリッカが起こるボタン

「フリッカが起こるボタン」とは、押下中にせわしなく状態が切り換わるセンサを指します。これは押下中でないのにセンサの状態に変化が起こる、誤検出とは異なります。フリッカが起こるボタンは、無反応センサと同様の理由で発生します。センサにノイズが注入されると、センサ感度が変化します。しきい値付近まで感度が下がると、ノイズにより読み値がしきい値を超えたり、しきい値より下がったりします。こうする事で、センサの状態がせわしなく変化します。

最適な解決方法： しきい値にヒステリシスを実装します。

これに対する最適かつ最も効果的な解決方法は、しきい値に大きなヒステリシスを実装する事です。押下と解放のしきい値を別々にする事で、より大きなシステムノイズも許容できるようになります。感度が高ければ、しきい値をさらに離して設定できます。こうする事でシステムのノイズ耐性が向上します。同様に、しきい値を近づけ過ぎるとシステムの挙動はしきい値が 1 つの場合と同様となり、再びボタンのフリッカが起こる可能性があります。このような理由から、この問題に対する一般的な解決方法はヒステリシスとデバウンス処理を組み合わせる事です。

方法 2: デバウンス値を大きくします。

この問題を軽減する最も簡単な方法は最大デバウンス値を大きくする事ですが、この方法には必ずしもうまく行きません。デバウンス値を大きくすると、システムの応答時間は長くなり、フリッカは可能性が下がるのみに留まります。フリッカの可能性が下がるという事はフリッカの速度が遅くなるという事ですが、ほとんどの場合フリッカを完全に取り除く事はできません。

方法 3: しきい値を調整します。

3 つ目の方法は、簡単に超える事ができる程度にしきい値を設定するか、ハードウェアを変更してシステムの感度を高める事です。ただしノイズレベルが高い場合、これだけでは問題を解決できない場合もあります。

共通の課題：リバース動作

リバース動作は、ノイズが大量に注入されるシステムで発生します。これは、ノイズにより通常とは逆方向に信号が変化する現象です。押下時に信号が下方向に変化するシステムでは、この場合上方向に変化します。システムが一方向の変化のみ検出するものである場合、この現象が問題となります。図 24 に、「下方向」の変化のみを検出するシステムが「上方向」の変化が起こった場合にどのようにリバース動作の状態に移行してしまうかを示します。この問題を解決するには、大きく分けて 2 つの方法があります。ベースラインの速度を下げ、ベースラインの両側にしきい値を設けます。

最適な解決方法： 両方向の変化をチェックします。

この問題に対するさらに確実な解決方法は、平均値の両側にしきい値を設け、どちらの方向に変化した場合も押下として検出できるようにする事です。ただし、全てのシステムでこの方法がうまく機能するとは限りません。例えば、濡れた状態でも動作するように設計されているシステムでは、ある方向への変化は指による押下を意味し、逆方向への変化は水の存在を意味します。このような場合、逆方向への変化は無視すべきです。結局、アプリケーションごとの要件を基に解決策を決定すべきです。

方法 2: フィルタの時定数を調整します。

ベースラインの速度を下げると、逆方向への信号変化によりシステムがリバース動作するまでの時間をより長くする事ができます。これにより、特定のタイプのノイズが注入された時にボタンが無反応になる可能性がありますが、システムによっては最良の方法となります。しかし、平均の更新を遅くするだけではこの問題を完全になくす事はできません。ユーザが非常に長い時間センサを押し続けた場合、平均の更新を遅くしていても最後には押下状態に追従してしまい、指を離した時にリバース動作が起こります。この点は、どのようなベースラインを使う場合でも考慮する必要がある性能上のトレードオフです。環境条件による読み値の変化と、押下による人為的な読み値の変化も、識別が困難な場合があります。

方法 3: システムの V_{DD} を高くします。

この方法ではリバース動作を完全に防ぐ事はできませんが、V_{DD} を高くする事でより高いノイズレベルに耐える事ができます。しかし、V_{DD} を高くするとシステムの消費電流も増えるため、この方法は最後の方法です。

結論

堅牢な静電容量式タッチ設計の基礎は、適切なハードウェア設計を選択する事です。本書で説明した設計ガイドラインに従えば、高いベース SNR を持つ事ができます。これにより要求されるソフトウェア オーバーヘッドが小さくなり、システム全体の応答時間が短くなり、ノイズの多い環境でもシステムが良好に動作します。

対象アプリケーションのニーズを満たすように設計を調整する方法は多くありますが、堅牢なシステムを設計するための最適な選択は表 1 の通りです。

サンプリング レートのジッタリング、スルーレート リミッタの実装等の信号取得方法は、読み値に混入するノイズの量を低減します。L 点平均、ローパス Butterworth 等のデジタルフィルタを使うとシステムは環境変化に追従する事ができ、SNR がさらに上がります。最後に、しきい値ヒステリシス、デバウンス、「最も押下されている」センサを検出するアルゴリズム等を実装すると、システムは非常にまれな外乱ノイズにさえ対応できるようになります。

静電容量式タッチシステムは、終始一貫してできるだけセンサ信号の SNR を高くする事に重点を置いて開発する必要があります。本書で説明したハードウェア ガイドラインに沿って設計し、同じく本書で説明したソフトウェア アルゴリズムを実装すれば、費用対効果の高い、堅牢なシステムを設計できます。

mTouch センシングの基礎と応用については、Microchip 社のウェブサイト (http://www.microchip.com/mTouch) をご覧ください。

表 1: 推奨する理想的なハードウェア設計

ボタンパッド		
	形状	特に制約なし
	寸法	15x15 mm
	パッド間距離	10 mm
カバー		
	厚さ	< 3 mm
	材質	$2.0 \le \varepsilon_r \le 8.0$
	接着剤	薄い、ε_r が大きい、気泡がない
レイアウト		
	センサトレース	薄い、短い
	直列抵抗	CVD 4.7 ～ 10 kΩ CTMU 1 ～ 2.5 kΩ
	LED/ 共通トレース	センサトレースから離す
電源		
	人体と GND を共有する	感度が倍増する
	ノイズに対して有利な V_{DD}	高いほど有利
	バイパス コンデンサ	全 V_{DD} ピンに 0.1 µF

Microchip 社製デバイスのコード保護機能に関して以下の点にご注意ください。

- Microchip 社製品は、該当する Microchip 社データシートに記載の仕様を満たしています。
- Microchip 社では、通常の条件ならびに仕様に従って使用した場合、Microchip 社製品のセキュリティ レベルは、現在市場に流通している同種製品の中でも最も高度であると考えています。
- しかし、コード保護機能を解除するための不正かつ違法な方法が存在する事もまた事実です。弊社の理解では、こうした手法は Microchip 社データシートにある動作仕様書以外の方法で Microchip 社製品を使用する事になります。このような行為は知的所有権の侵害に該当する可能性が非常に高いと言えます。
- Microchip 社は、コードの保全性に懸念を抱いているお客様と連携し、対応策に取り組んでいきます。
- Microchip 社を含む全ての半導体メーカーで、自社のコードのセキュリティを完全に保証できる企業はありません。コード保護機能とは、Microchip 社が製品を「解読不能」として保証するものではありません。

コード保護機能は常に進歩しています。Microchip 社では、常に製品のコード保護機能の改善に取り組んでいます。Microchip 社のコード保護機能の侵害は、デジタル ミレニアム著作権法に違反します。そのような行為によってソフトウェアまたはその他の著作物に不正なアクセスを受けた場合、デジタル ミレニアム著作権法の定めるところにより損害賠償訴訟を起こす権利があります。

商標

ISBN: 978-1-63277-502-3

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
═ ISO/TS 16949 ═

Microchip 社では、Chandler および Tempe (アリゾナ州)、Gresham (オレゴン州) の本部、設計部およびウェハー製造工場そしてカリフォルニア州とインドのデザインセンターが ISO/TS-16949: 2009 認証を取得しています。Microchip 社の品質システム プロセスおよび手順は、PIC® MCU および dsPIC® DSC、KEELOQ® コード ホッピング デバイス、シリアル EEPROM、マイクロペリフェラル、不揮発性メモリ、アナログ製品に採用されています。さらに、開発システムの設計と製造に関する Microchip 社の品質システムは ISO 9001:2000 認証を取得しています。

各国の営業所とサービス

北米

本社
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
技術サポート :
http://www.microchip.com/support
URL:
www.microchip.com

アトランタ
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

オースティン、TX
Tel: 512-257-3370

ボストン
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

シカゴ
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

クリーブランド
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

ダラス
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

デトロイト
Novi, MI
Tel: 248-848-4000

ヒューストン、TX
Tel: 281-894-5983

インディアナポリス
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

ロサンゼルス
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

ニューヨーク、NY
Tel: 631-435-6000

サンノゼ、CA
Tel: 408-735-9110

カナダ - トロント
Tel: 905-673-0699
Fax: 905-673-6509

アジア / 太平洋

アジア太平洋支社
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2943-5100
Fax: 852-2401-3431

オーストラリア - シドニー
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

中国 - 北京
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

中国 - 成都
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

中国 - 重慶
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

中国 - 杭州
Tel: 86-571-8792-8115
Fax: 86-571-8792-8116

中国 - 香港 SAR
Tel: 852-2943-5100
Fax: 852-2401-3431

中国 - 南京
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

中国 - 青島
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

中国 - 上海
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

中国 - 瀋陽
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

中国 - 深圳
Tel: 86-755-8864-2200
Fax: 86-755-8203-1760

中国 - 武漢
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

中国 - 西安
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

中国 - 厦門
Tel: 86-592-2388138
Fax: 86-592-2388130

中国 - 珠海
Tel: 86-756-3210040
Fax: 86-756-3210049

アジア / 太平洋

インド - バンガロール
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

インド - ニューデリー
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

インド - プネ
Tel: 91-20-3019-1500

日本 - 大阪
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

日本 - 東京
Tel: 81-3-6880-3770
Fax: 81-3-6880-3771

韓国 - 大邱
Tel: 82-53-744-4301
Fax: 82-53-744-4302

韓国 - ソウル
Tel: 82-2-554-7200
Fax: 82-2-558-5932 または
82-2-558-5934

マレーシア - クアラルンプール
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

マレーシア - ペナン
Tel: 60-4-227-8870
Fax: 60-4-227-4068

フィリピン - マニラ
Tel: 63-2-634-9065
Fax: 63-2-634-9069

シンガポール
Tel: 65-6334-8870
Fax: 65-6334-8850

台湾 - 新竹
Tel: 886-3-5778-366
Fax: 886-3-5770-955

台湾 - 高雄
Tel: 886-7-213-7830

台湾 - 台北
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

タイ - バンコク
Tel: 66-2-694-1351
Fax: 66-2-694-1350

ヨーロッパ

オーストリア - ヴェルス
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

デンマーク - コペンハーゲン
Tel: 45-4450-2828
Fax: 45-4485-2829

フランス - パリ
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

ドイツ - デュッセルドルフ
Tel: 49-2129-3766400

ドイツ - ミュンヘン
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

ドイツ - プフォルツハイム
Tel: 49-7231-424750

イタリア - ミラノ
Tel: 39-0331-742611
Fax: 39-0331-466781

イタリア - ヴェニス
Tel: 39-049-7625286

オランダ - ドリューネン
Tel: 31-416-690399
Fax: 31-416-690340

ポーランド - ワルシャワ
Tel: 48-22-3325737

スペイン - マドリッド
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

スウェーデン - ストックホルム
Tel: 46-8-5090-4654

イギリス - ウォーキンガム
Tel: 44-118-921-5800
Fax: 44-118-921-5820

03/25/14