Copyrights

© Cypress Semiconductor Corporation, 2013-2015. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Trademarks

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

Source Code

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

Disclaimer

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

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

1. Introduction .................................................................................................................................................. 5
   1.1 Abstract .................................................................................................................................................. 5
   1.2 Introduction ........................................................................................................................................... 5
   1.3 PSoC 4 and PRoC BLE CapSense Features ....................................................................................... 5
   1.4 PSoC 4 and PRoC BLE CapSense Plus Features ............................................................................. 6
   1.5 CapSense Design Flow ....................................................................................................................... 9
2. CapSense Technology ..................................................................................................................................... 11
   2.1 CapSense Fundamentals .................................................................................................................... 11
   2.2 Capacitive Touch Sensing Method ...................................................................................................... 13
   2.3 CapSense Widgets ............................................................................................................................. 14
      2.3.1 Buttons (Zero-Dimensional) ......................................................................................................... 14
      2.3.2 Sliders (One-Dimensional) .......................................................................................................... 15
      2.3.3 Touchpads / Trackpads (Two-Dimensional) ............................................................................... 17
      2.3.4 Proximity (Three-Dimensional) .................................................................................................. 17
   2.4 Liquid Tolerance ................................................................................................................................... 18
      2.4.1 Effect of Liquid Droplets and Liquid Stream on a CapSense Sensor ........................................... 19
      2.4.2 Driven-Shield Signal and Shield Electrode .................................................................................. 21
      2.4.3 Guard Sensor .................................................................................................................................. 21
3. PSoC 4 and PRoC BLE CapSense ................................................................................................................ 24
   3.1 CapSense CSD Sensing ....................................................................................................................... 24
      3.1.1 GPIO Cell Capacitance to Current Converter .............................................................................. 25
      3.1.2 CapSense Clock Generator .......................................................................................................... 27
      3.1.3 Sigma Delta Converter ................................................................................................................. 28
      3.1.4 Analog Multiplexer ....................................................................................................................... 29
   3.2 CapSense CSD Shielding ..................................................................................................................... 29
   3.3 CapSense in PSoC 4 M-Series ............................................................................................................ 30
4. CapSense Design and Development Tools ............................................................................................... 32
   4.1 PSoC Creator ........................................................................................................................................ 32
      4.1.1 CapSense_CSD Component ......................................................................................................... 32
      4.1.2 Example Projects ......................................................................................................................... 33
   4.2 Hardware Kits ...................................................................................................................................... 33
5. CapSense Performance Tuning .................................................................................................................. 35
   5.1 SmartSense .......................................................................................................................................... 35
      5.1.1 Component Configuration for SmartSense .................................................................................. 36
      5.1.2 Widget Configuration .................................................................................................................... 38
## 5.1.3 Scan Order

---

## 5.2 Manual Tuning

- **5.2.1 Fundamentals of Manual Tuning** ................................................................. 45
- **5.2.2 Manual Tuning Process** .................................................................................. 54

## 6. Design Considerations .......................................................................................... 65

### 6.1 Firmware .............................................................................................................. 65
- **6.1.1 Low-Power Design** ......................................................................................... 66

### 6.2 Sensor Construction ............................................................................................. 67
- **6.2.1 Overlay Material** ............................................................................................ 69
- **6.2.2 Overlay Thickness** .......................................................................................... 69
- **6.2.3 Overlay Adhesives** .......................................................................................... 70

### 6.3 PCB Layout Guidelines ......................................................................................... 70
- **6.3.1 Parasitic Capacitance, Cpar** ............................................................................ 70
- **6.3.2 Board Layers** .................................................................................................. 70
- **6.3.3 Button Design** ................................................................................................ 70
- **6.3.4 Slider Design** .................................................................................................. 71
- **6.3.5 Sensor and Device Placement** ........................................................................ 78
- **6.3.6 Trace Length and Width** .................................................................................. 78
- **6.3.7 Trace Routing** ................................................................................................ 78
- **6.3.8 Crosstalk Solutions** ......................................................................................... 79
- **6.3.9 Vias** .................................................................................................................. 80
- **6.3.10 Ground Plane** ................................................................................................ 81
- **6.3.11 Power Supply Layout Recommendations** .................................................... 84
- **6.3.12 Layout Guidelines for Liquid Tolerance** ......................................................... 85
- **6.3.13 Schematic Rule Checklist** ............................................................................. 88
- **6.3.14 Layout Rule Checklist** .................................................................................... 89

### 6.4 ESD Protection ...................................................................................................... 92
- **6.4.1 Preventing ESD Discharge** .............................................................................. 92
- **6.4.2 Redirect** ......................................................................................................... 93
- **6.4.3 ESD Protection Devices** ................................................................................ 93

### 6.5 Electromagnetic Compatibility (EMC) Considerations ........................................ 94
- **6.5.1 Radiated Interference and Emissions** ............................................................... 94
- **6.5.2 Conducted RF Noise** ..................................................................................... 102

## 7. CapSense Plus ......................................................................................................... 103

## 8. Resources ................................................................................................................ 106
- **8.1 Website** .............................................................................................................. 106
- **8.2 Datasheet** .......................................................................................................... 106
- **8.3 Technical Reference Manual** ........................................................................... 106
- **8.4 Development Kits** ............................................................................................ 106
- **8.5 PSoC Creator** .................................................................................................... 107
- **8.6 Application Notes** ............................................................................................. 107
- **8.7 Design Support** .................................................................................................. 107

## Revision History ....................................................................................................... 108
1. Introduction

1.1 Abstract

The PSoC® 4 CapSense® Design Guide shows how to design capacitive touch sensing applications with the PSoC 4 and PRoC BLE families of devices. The CapSense feature in PSoC 4 and PRoC BLE offers unprecedented signal-to-noise ratio, best-in-class liquid tolerance, and a wide variety of sensors such as buttons, sliders, track pads and proximity sensors. This guide explains the CapSense operation, CapSense design tools, the PSoC Creator™ CapSense CSD Component, performance tuning, and design considerations.

1.2 Introduction

Capacitive touch sensors are user interface devices that use human body capacitance to detect the presence of a finger on or near a sensor. Capacitive sensors are aesthetically superior, easy-to-use, and have long lifetimes. Cypress CapSense solutions bring elegant, reliable, and easy-to-use capacitive touch sensing functionality to your product. Cypress CapSense solutions have replaced more than four billion mechanical buttons.

This design guide focuses on the CapSense feature in the PSoC 4 and PRoC BLE families of devices. These are true programmable embedded system-on-chip integrating configurable analog and digital peripheral functions, memory, radio, and a microcontroller on a single chip. These devices are highly flexible and can implement many functions in addition to CapSense, which accelerates time-to-market, integrates critical system functions, and reduces overall system cost.

This guide assumes that you are familiar with developing applications for PSoC 4 or PRoC BLE using the Cypress PSoC Creator IDE. If you are new to PSoC 4, an introduction can be found in AN79953, Getting Started with PSoC 4 or AN92167, Getting Started with PSoC 4 BLE. If you are new to PRoC BLE, take a look at AN94020, Getting Started with PRoC BLE. If you are new to PSoC Creator, see the PSoC Creator home page.

This design guide helps you understand:

- Fundamentals of CapSense technology
- CapSense in PSoC 4 and PRoC BLE
- Design and development tools available for CapSense
- Performance tuning
- CapSense Plus other applications

1.3 PSoC 4 and PRoC BLE CapSense Features

CapSense in PSoC 4 and PRoC BLE has the following features:

- Robust sensing technology
- CapSense Sigma Delta (CSD) operation, which provides best-in-class signal-to-noise ratio (SNR)
- High-performance sensing across a variety of overlay materials and thicknesses
- SmartSense™ Auto-tuning technology
- High range proximity sensing
- Water-tolerant operation
Introduction

- Low power consumption
- Two IDAC operation to increase scan speed and SNR
- Pseudo random sequence (PRS) clock source for lower electromagnetic interference (EMI)
- Supports both positive and negative charge transfer methods
- GPIO precharge (supported on two dedicated pins) quickly initializes external tank capacitors
- Wide operating voltage range (1.71 – 5.5 V)
- Reduced BOM cost with integrated CapSense Plus features (ADC, DAC, timer, counter, PWM, BLE)

1.4 PSoC 4 and PRoC BLE CapSense Plus Features

You can create PSoC 4 or PRoC BLE “CapSense Plus applications” that feature capacitive touch sensing and additional system functionality. The main features of these chips are:

- ARM® Cortex™-M0 CPU with single cycle multiply delivering up to 43 DMIPS at 48 MHz
- 1.71 – 5.5-V operation over -40 to 85 °C ambient
- Up to 128 KB of flash (Cortex-M0 has >2X code density over 8-bit solutions)
- Up to 16 KB of SRAM
- Up to 55 programmable GPIOs
- Independent center-aligned PWMs with complementary dead-band programmable outputs, synchronized ADC operation (ability to trigger the ADC at a customer-specifiable time in the PWM cycle) and synchronous refresh (ability to synchronize PWM duty cycle changes across all PWMs to avoid anomalous waveforms)
- Comparator-based triggering of PWM Kill signals (to terminate motor-driving when an over-current condition is detected)
- 12-bit 1 MspS ADC including sample-and-hold (S&H) capability with zero-overhead sequencing allowing the entire ADC bandwidth to be used for signal conversion and none used for sequencer overhead
- Opamps with comparator mode and SAR input buffering capability
- Segment LCD direct drive that supports up to 4-Commons
- SPI / UART / I²C serial communication channels
- Bluetooth Low Energy (BLE) communication compliant with version 4.0 and multiple features of version 4.1
- Programmable logic blocks, each having 8 macrocells and a cascadable data path, called universal digital blocks (UDBs) for very efficient implementation of programmable peripherals (such as I²S)
- Controller Area Network (CAN)
- Fully supported PSoC Creator design entry, development, and debug environment providing:
  - Design entry and build (comprehending analog routing)
  - Components for all fixed-function peripherals and common programmable peripherals
  - Documentation and training modules
  - Support for porting builds to ARM MDK environment (previously known as RealView) and others

Table 1-1 shows the features available in CY8C4000, CY8C4100, CY8C4100-BL, CY8C4200, CY8C4200-BL and CYBL10X6X device families.
<table>
<thead>
<tr>
<th>Features</th>
<th>CY8C4000</th>
<th>CY8C4100</th>
<th>CY8C4100-BL</th>
<th>CY8C4200</th>
<th>CY8C4200-BL</th>
<th>CYBL10X6X</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>16-MHz Cortex-M0 CPU</td>
<td>24-MHz Cortex-M0 CPU with single-cycle multiply</td>
<td>24-MHz Cortex-M0 CPU with single-cycle multiply</td>
<td>48-MHz Cortex-M0 CPU with single-cycle multiply</td>
<td>48-MHz Cortex-M0 CPU with single-cycle multiply</td>
<td>48-MHz Cortex-M0 CPU with single-cycle multiply</td>
</tr>
<tr>
<td>Flash</td>
<td>As much as 16 KB</td>
<td>As much as 128 KB</td>
<td>128 KB</td>
<td>As much as 128 KB</td>
<td>128 KB</td>
<td>128 KB</td>
</tr>
<tr>
<td>SRAM</td>
<td>As much as 2 KB</td>
<td>16 KB</td>
<td>As much as 16 KB</td>
<td>16 KB</td>
<td>16 KB</td>
<td>16 KB</td>
</tr>
<tr>
<td>GPIOs</td>
<td>As many as 20</td>
<td>As many 55</td>
<td>36</td>
<td>As many as 55</td>
<td>36</td>
<td>36</td>
</tr>
<tr>
<td>CapSense</td>
<td>As many as 16 sensors</td>
<td>As many as 54 sensors</td>
<td>As many as 35 sensors</td>
<td>As many as 35 sensors</td>
<td>As many as 35 sensors</td>
<td>As many as 35 sensors</td>
</tr>
<tr>
<td>BLE</td>
<td>None</td>
<td>None</td>
<td>BLE 4.0 with multiple optional features of BLE 4.1</td>
<td>None</td>
<td>BLE 4.0 with multiple optional features of BLE 4.1</td>
<td>BLE 4.0 with multiple optional features of BLE 4.1</td>
</tr>
<tr>
<td>ADC</td>
<td>None</td>
<td>12-bit, 806-ksp SAR ADC with sequencer</td>
<td>12-bit, 806-ksp SAR ADC with sequencer</td>
<td>12-bit, 1-Msp SAR ADC with sequencer</td>
<td>12-bit, 1-Msp SAR ADC with sequencer</td>
<td>12-bit, 1-Msp SAR ADC with sequencer</td>
</tr>
<tr>
<td>Opamps</td>
<td>None</td>
<td>4 programmable opamps</td>
<td>4 programmable opamps</td>
<td>4 programmable opamps</td>
<td>4 programmable opamps</td>
<td>None</td>
</tr>
<tr>
<td>Comparators</td>
<td>1 CSD comparator with fixed threshold (1.2 V)</td>
<td>2 low-power comparators with wakeup feature</td>
<td>2 low-power comparators with wakeup feature</td>
<td>2 low-power comparators with wakeup feature</td>
<td>2 low-power comparators with wakeup feature</td>
<td>None</td>
</tr>
<tr>
<td>IDACs</td>
<td>One 7-bit and one 8-bit</td>
<td>Two 7-bit and two 8-bit</td>
<td>One 7-bit and two 8-bit</td>
<td>Two 7-bit and two 8-bit</td>
<td>One 7-bit and one 8-bit</td>
<td>None</td>
</tr>
<tr>
<td>Programmable logic blocks (UDBs)</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>Four, each with 8 macrocells and one datapath</td>
<td>Four, each with 8 macrocells and one datapath</td>
<td>None</td>
</tr>
<tr>
<td>Power supply range</td>
<td>1.71 to 5.5 V</td>
<td>1.71 to 5.5 V</td>
<td>1.71 to 5.5 V</td>
<td>1.71 to 5.5 V</td>
<td>1.71 to 5.5 V</td>
<td>1.71 to 5.5 V</td>
</tr>
<tr>
<td>Low power modes</td>
<td>Deep-Sleep mode at 2.5 µA</td>
<td>Deep-Sleep mode at 1.3 µA Hibernation mode at 150 nA Stop mode at 20 nA</td>
<td>Deep-Sleep mode at 1.3 µA Hibernation mode at 150 nA Stop mode at 60 nA</td>
<td>Deep-Sleep mode at 1.3 µA Hibernation mode at 150 nA Stop mode at 20 nA</td>
<td>Deep-Sleep mode at 1.3 µA Hibernation mode at 150 nA Stop mode at 60 nA</td>
<td>None</td>
</tr>
<tr>
<td>Segment LCD drive</td>
<td>None</td>
<td>4 COM segment LCD drive</td>
<td>4 COM segment LCD drive</td>
<td>4 COM segment LCD drive</td>
<td>4 COM segment LCD drive</td>
<td>4 COM segment LCD drive</td>
</tr>
<tr>
<td>Serial communication</td>
<td>1 I²C</td>
<td>2 independent Serial Communication Blocks (SCBs) with programmable I²C, SPI, or UART, CAN, and LIN for automotive</td>
<td>2 independent Serial Communication Blocks (SCBs) with programmable I²C, SPI, or UART</td>
<td>2 independent Serial Communication Blocks (SCBs) with programmable I²C, SPI, or UART</td>
<td>2 independent Serial Communication Blocks (SCBs) with programmable I²C, SPI, or UART</td>
<td>1 or 2 independent Serial Communication Blocks (SCBs) with programmable I²C, SPI, or UART</td>
</tr>
<tr>
<td>Timer / Counter / Pulse-Width Modulator (TCPWM)</td>
<td>1</td>
<td>8</td>
<td>4</td>
<td>8</td>
<td>4</td>
<td>As many as 4</td>
</tr>
<tr>
<td>Other Fixed Function Peripherals (I²S, PWM)</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>I²S, PWM</td>
</tr>
<tr>
<td>Features</td>
<td>CY8C4000</td>
<td>CY8C4100</td>
<td>CY8C4100-BL</td>
<td>CY8C4200</td>
<td>CY8C4200-BL</td>
<td>CYBL10X6X</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Clocks</td>
<td>24-/ 32-MHz internal main oscillator (IMO)</td>
<td>3- to 24-MHz internal main oscillator (IMO)</td>
<td>3- to 24-MHz internal main oscillator (IMO)</td>
<td>3- to 48-MHz internal main oscillator (IMO)</td>
<td>3- to 48-MHz internal main oscillator (IMO)</td>
<td>3- to 48-MHz internal main oscillator (IMO)</td>
</tr>
<tr>
<td></td>
<td>32-kHz internal low-speed oscillator (ILO)</td>
<td>32-kHz internal low-speed oscillator (ILO)</td>
<td>32-kHz internal low-speed oscillator (ILO)</td>
<td>32-kHz internal low-speed oscillator (ILO)</td>
<td>32-kHz internal low-speed oscillator (ILO)</td>
<td>32-kHz internal low-speed oscillator (ILO)</td>
</tr>
<tr>
<td></td>
<td>External Clock (0–16 MHz)</td>
<td>External Clock (0–24 MHz)</td>
<td>External Clock (0–24 MHz)</td>
<td>External Clock (0–48 MHz)</td>
<td>External Clock (0–48 MHz)</td>
<td>External Clock (0–48 MHz)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Low-voltage detection (LVD)</td>
<td>Low-voltage detection (LVD)</td>
<td>Low-voltage detection (LVD)</td>
<td>Low-voltage detection (LVD)</td>
<td>Low-voltage detection (LVD)</td>
<td>Low-voltage detection (LVD)</td>
</tr>
</tbody>
</table>
1.5 CapSense Design Flow

Figure 1-1 shows the typical flow of a product design cycle with capacitive sensing; the information in this guide is highlighted in green. Table 1-2 on page 10 provides links to the supporting documents for each of the numbered tasks in Figure 1-1.

Figure 1-1. CapSense Design Flow

1. Understanding CapSense Technology
2. Specify System Requirements and Characteristics
3. Feasibility Study: Device Selection Based on Required Functionality

Design for CapSense

Mechanical and PCB Design

4. CapSense Schematic Design
5. CapSense Layout and Mechanical Design

PSoC Creator Project Creation

6. Component Configuration
7. CapSense Tuning
8. Firmware Design

9. Programming PSoC

10. System Integration and Build Preproduction Prototype

11. Design Validation: Test and Evaluate System Functionality and CapSense Performance

Performance Satisfactory

No

Yes

12. Production

Topics covered in this document =
Topics covered in other documents =
Not covered in any document. Users should define the process based on application =
<table>
<thead>
<tr>
<th>Steps in Flowchart</th>
<th>Supporting Cypress Documentation</th>
<th>Name</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Understanding CapSense</td>
<td>PSoC 4 CapSense Design Guide (This document)</td>
<td></td>
<td>Chapter 2 and Chapter 3</td>
</tr>
<tr>
<td>2. Specify requirements</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. Feasibility study</td>
<td>PSoC 4 Datasheet</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PSoC 4 BLE Datasheet</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PRoC BLE Datasheet</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AN79953, Getting Started with PSoC 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AN91267, Getting Started with PSoC 4 BLE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>AN94020, Getting Started with PRoC BLE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Schematic design</td>
<td>PSoC 4 CapSense Design Guide (This document)</td>
<td></td>
<td>Chapter 6</td>
</tr>
<tr>
<td>5. Layout design</td>
<td>PSoC 4 CapSense Design Guide (This document)</td>
<td></td>
<td>Chapter 6</td>
</tr>
<tr>
<td>6. Component configuration</td>
<td>CapSense Component datasheet</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PSoC 4 CapSense Design Guide (This document)</td>
<td></td>
<td>Chapter 5</td>
</tr>
<tr>
<td>7. Performance tuning</td>
<td>PSoC 4 CapSense Design Guide (This document)</td>
<td></td>
<td>Chapter 5</td>
</tr>
<tr>
<td>8. Firmware design</td>
<td>PSoC Creator CapSense Component datasheet</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PSoC Creator example projects</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9. Programming PSoC</td>
<td>PSoC Creator home page</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10. Prototype</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11. Design validation</td>
<td>PSoC 4 CapSense Design Guide (This document)</td>
<td></td>
<td>Chapter 5</td>
</tr>
<tr>
<td>12. Production</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2. CapSense Technology

Capacitive touch sensing technology measures changes in capacitance between a plate (the sensor) and its environment to detect the presence of a finger on or near a touch surface.

2.1 CapSense Fundamentals

A typical CapSense sensor consists of a copper pad of proper shape and size etched on the surface of a PCB. A nonconductive overlay serves as the touch surface for the button, as Figure 2-1 shows.

PCB traces and vias connect the sensor pads to PSoC GPIOs that are configured as CapSense sensor pins. As Figure 2-2 shows, the total amount of capacitance on each of the sensor pins is modeled as equivalent lumped capacitors with values of $C_{S1}$, $C_{S2}$, through $C_{S5}$. CapSense circuitry internal to the PSoC converts these capacitance values into equivalent digital counts (see Chapter 3 for details). These digital counts are then processed by the CPU to detect touches.

CapSense also requires an external capacitor $C_{MOD}$, which is connected between one of the GPIOs and ground. If liquid tolerance or proximity sensing is used, an additional $C_{SH\_TANK}$ capacitor may be required.
The capacitance of the sensor in the absence of a touch is called the parasitic capacitance, $C_P$. Parasitic capacitance results from the electric field between the sensor (including the sensor pad, traces and vias) and other conductors in the system such as the ground planes, traces, any metal in the product’s chassis or enclosure, etc. The GPIO and internal capacitances of PSoC also contribute to the parasitic capacitance. However, these internal capacitances are typically very small compared to the sensor capacitance.

Figure 2-3 shows how a sensor GPIO pin is connected to a sensor pad by traces and vias. Typically, a ground hatch surrounds the sensor pad to isolate it from other sensors and traces. Although Figure 2-3 shows some field lines around the sensor pad, the actual electric field distribution is very complex.

![Figure 2-3. Parasitic Capacitance](image)

When a finger is present on the overlay, the conductive nature and large mass of the human body forms a grounded, conductive plane parallel to the sensor pad, as Figure 2-4 shows.

![Figure 2-4. Finger Capacitance](image)

This arrangement forms a parallel plate capacitor. The capacitance between the sensor pad and the finger is:

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

(2-1)

Where:

- $\varepsilon_0$ = Free space permittivity
- $\varepsilon_r$ = Relative permittivity of overlay
- $A$ = Area of finger and sensor pad overlap
- $d$ = Thickness of the overlay

$C_F$ is known as the finger capacitance. The parasitic capacitance $C_P$ and finger capacitance $C_F$ are parallel to each other because both represent the capacitances between the sensor pin and ground. Therefore, the total capacitance $C_S$ of the sensor, when the finger is present on the sensor, is the sum of $C_P$ and $C_F$. 

In the absence of touch, \( C_S \) is equal to \( C_P \).

PSoC converts the capacitance \( C_S \) into equivalent digital counts called raw counts. Because a finger touch increases the total capacitance of the sensor pin, an increase in the raw counts indicates a finger touch. As the parasitic capacitance \( C_P \) increases, the ratio of \( C_F \) to \( C_P \) decreases – the per unit change in capacitance corresponding to a finger touch decreases. Therefore, as \( C_P \) increases, touch detection becomes more difficult. PSoC 4 CapSense supports parasitic capacitance values as high as 65 pF for 0.3-pF finger capacitance, and as high as 35 pF for 0.1-pF finger capacitance.

### 2.2 Capacitive Touch Sensing Method

PSoC 4 uses a capacitive touch sensing method known as CapSense Sigma Delta (CSD). The CapSense Sigma Delta touch sensing method provides the industry’s best-in-class signal-to-noise ratio. CSD is a combination of hardware and firmware techniques. Figure 2-5 shows a highly simplified block diagram of the CSD method.

Figure 2-5. Simplified Diagram of CapSense Sigma Delta Method

With CSD, each GPIO has a switched-capacitance circuit that converts the sensor capacitance into an equivalent current. An analog multiplexer then selects one of the currents and feeds it into the current to digital converter. The current to digital converter is similar to a Delta Sigma ADC. For an in-depth discussion of the PSoC 4 CapSense CSD block, see PSoC 4 CapSense.

The output count of the current to digital converter, known as raw count, is a digital value that is proportional to the sensor capacitance:

\[
\text{raw count} = G_c C_S \quad (2-3)
\]

Where \( G_c \) is the capacitance to digital conversion gain of CapSense.

\[ C_S = C_P + C_F \quad (2-2) \]
Figure 2-6 shows a plot of raw count over time. When a finger touches the sensor, the \(C_S\) increases from \(C_P\) to \(C_P + C_F\), and the raw count increases proportionally. By comparing the change in raw count to a predetermined threshold, logic in firmware decides whether the sensor is active (finger is present) or not.

\[
\begin{array}{c}
\text{Figure 2-6. Raw Count Versus Time} \\
\end{array}
\]

2.3 CapSense Widgets

CapSense widgets consist of one or more CapSense sensors, which as a unit represent a certain type of user interface. CapSense widgets are broadly classified into four categories, as Figure 2-7 shows: buttons, sliders, touchpad / trackpad, and proximity sensors. This section explains the basic concepts of different CapSense widgets. For a detailed explanation of sensor construction, see Sensor Construction.

\[
\begin{array}{c}
\text{Figure 2-7. Several types of Widgets} \\
\end{array}
\]

2.3.1 Buttons (Zero-Dimensional)

CapSense buttons replace mechanical buttons in a wide variety of applications such as home appliances, medical devices, white goods, lighting controls and many other products. It is the simplest type of CapSense widget, consisting of a single sensor. A CapSense button gives one of two possible output states: active (finger is present) or inactive (finger is not present). These two states are also called ON and OFF states, respectively.
A simple CapSense button consists of a circular copper pad connected to a PSoC GPIO using PCB traces, as Figure 2-8 shows. The button is surrounded by grounded copper hatch to isolate it from other buttons and traces. A circular gap separates the button pad and the ground hatch. Each button requires one PSoC GPIO.

If the application requires a large number of buttons, such as in a calculator keypad or a QWERTY keyboard, you can arrange the CapSense buttons in a matrix, as Figure 2-9 shows. This allows a design to have multiple buttons per GPIO. For example, the 12-button design in Figure 2-9 requires only seven GPIOs.

A matrix button design has two groups of capacitive sensors: row sensors and column sensors. Each button consists of a row sensor and a column sensor, as Figure 2-9 shows. When a button is touched, both row and column sensors of that button become active. The number of buttons supported by the matrix is equal to the product of the number of rows and the number of columns.

Matrix buttons can only be sensed one at a time. If more than one row or column sensor is in the active state, the finger location cannot be resolved, which is considered an invalid condition. Some applications require simultaneous sensing of multiple buttons, such as a keyboard with Shift, Ctrl, and Alt keys. In this case, you should design the Shift, Ctrl, and Alt keys as individual buttons.

2.3.2 Sliders (One-Dimensional)

Sliders are used when the required input is in the form of a gradual increment or decrement. Examples include lighting control (dimmer), volume control, graphic equalizer, and speed control. A slider consists of a one-dimensional array of capacitive sensors called segments, which are placed adjacent to one another. Touching one segment also results in partial activation of adjacent segments. The firmware processes the raw counts from the touched segment and the nearby segments to calculate the position of the geometric center of the finger touch, which is known as the centroid position.

The actual resolution of the calculated centroid position is much higher than the number of segments in a slider. For example, a slider with five segments can resolve at least 100 physical finger positions. This high resolution gives smooth transitions of the centroid position as the finger glides across a slider.
In a linear slider, the segments are arranged inline, as Figure 2-10 shows. Each slider segment connects to a PSoC GPIO. A zigzag pattern (double chevron) is recommended for slider segments. This layout ensures that when a segment is touched, the adjacent segments are also partially touched, which aids estimation of the centroid position.

Radial sliders are similar to linear sliders except that radial sliders are continuous. Figure 2-11 shows a typical radial slider.
2.3.3 Touchpads / Trackpads (Two-Dimensional)

A trackpad (also known as touch pad) has two linear sliders arranged in an X and Y pattern, enabling it to locate a finger’s position in both X and Y dimensions. Figure 2-12 shows a typical arrangement of a track pad sensor.

![Trackpad Sensor Arrangement](image)

Figure 2-12. Trackpad Sensor Arrangement

2.3.4 Proximity (Three-Dimensional)

Proximity sensors detect the presence of a hand in the three-dimensional space around the sensor. However, the actual output of the proximity sensor is an ON/OFF state similar to a CapSense button. Proximity sensing can detect a hand at a distance of several centimeters to tens of centimeters depending on the sensor construction.

Proximity sensing requires electric fields that are projected to much larger distances than buttons and sliders. This demands a large sensor area. However, a large sensor area also results in a large parasitic capacitance $C_p$, and detection becomes more difficult (see CapSense Fundamentals). This requires a sensor with high electric field strength at large distances while also having a small area. Use a trace with a thickness of 2-3 mm surrounding the other sensors, as Figure 2-13 shows.

![Proximity Sensor](image)

Figure 2-13. Proximity Sensor

You can also implement a proximity sensor by ganging other sensors together. This is accomplished by combining multiple sensor pads into one large sensor using firmware. The disadvantage of this method is high parasitic capacitance. See the CapSense CSD Component datasheet for details.

Refer to Getting Started with CapSense design guide for details on proximity sensor layout guidelines.
2.4 Liquid Tolerance

Capacitive sensing is used in a variety of applications such as home appliances, automotive, and industrial applications. These applications require robust capacitive-sensing operation even in the presence of mist, moisture, water, ice, and humidity changes. In a capacitive-sensing application design, false sensing of touch or proximity detection may happen due to the presence of a film of liquid or liquid droplets on the sensor surface, due to the conductive nature of some liquids. Cypress’s CapSense sensing method can compensate for variation in raw count due to these causes and provide a robust, reliable, capacitive sensing application operation.

To compensate for changes in raw count due to mist, moisture and humidity changes, the CapSense sensing method continuously adjusts the baseline of the sensor to prevent false triggers. To compensate for changes in raw count due to a liquid droplet or liquid flow, you should implement a Shield Electrode and Guard Sensor to provide robust touch sensing, as Figure 2-15 shows. CapSense reliably works and reports the sensor ON/OFF status when a shield electrode is implemented and liquid droplets are present on the sensor surface. When there is a liquid flow, the sensors will not be scanned and therefore the sensor ON/OFF status will not be reported.

Figure 2-14. Liquid-Tolerant CapSense-Based Touch User Interface in a Washing Machine

Figure 2-15. Shield Electrode (SH) and Guard Sensor (GUARD) Connected to CapSense Controller
2.4.1 Effect of Liquid Droplets and Liquid Stream on a CapSense Sensor

To understand the effect of liquids on a CapSense sensor, consider a CapSense system in which the hatch fill around the sensor is connected to ground, as Figure 2-16(a) shows. The hatch fill when connected to a ground improves the noise immunity of the sensor. Parasitic capacitance of the sensor is denoted as \( C_P \) in Figure 2-16(b).

As shown in Figure 2-17, when a liquid droplet falls on the sensor surface, due to its conductive nature it provides a strong coupling path for the electric field lines to return to ground; this adds a capacitance \( C_{LD} \) in parallel to \( C_P \). This added capacitance draws an additional charge from the AMUX bus as explained in GPIO Cell Capacitance to Current Converter, resulting in an increase in the sensor raw count. In some cases (when the liquid is salty or contains minerals), the increase in raw count when a liquid droplet falls on the sensor surface might be equal to the increase in raw count due to a finger touch, as Figure 2-18 shows. In such a situation, sensor false triggers might occur.

\[ \text{Figure 2-16. Typical CapSense System Layout} \]

\[ \text{Figure 2-17. Capacitance Added by Liquid Droplet when the Hatch Fill is Connected to Ground} \]

\[ \text{Figure 2-18. Effect of Liquid Droplet when the Hatch Fill Around the Sensor is Connected to Ground} \]
To nullify the effect of capacitance added by the liquid droplet to the CapSense circuitry, you should drive the hatch fill around the sensor with the driven-shield signal.

As Figure 2-19 shows, when the hatch fill around the sensor is connected to the driven-shield signal and when a liquid droplet falls on the touch interface, the voltage on both the sides of the liquid droplet remains at the same potential. Because of this the capacitance, $C_{LD}$, added by the liquid droplet does not draw any additional charge from the AMUX bus and hence the effect of capacitance $C_{LD}$ is nullified. Therefore, the increase in raw count when a water droplet falls on the sensor will be very small, as Figure 2-20 shows.

Figure 2-19. Capacitance Added by Liquid Droplet when the Hatch Fill Around the Sensor Is Connected to Shield

Figure 2-20. Effect of Liquid Droplet when the Hatch Fill Around the Sensor is Connected to the Driven-Shield

- $C_S$ – Sensor Parasitic Capacitance
- $C_{SH}$ – Capacitance Between the Sensor and the Hatch Fill
- $C_{HG}$ – Capacitance Between the Hatch Fill and Ground
- $C_{LD}$ – Capacitance Added by the Liquid Droplet
2.4.2 Driven-Shield Signal and Shield Electrode

The driven-shield signal is a buffered version of the sensor-switching signal, as Figure 2-21 shows. The driven-shield signal has the same amplitude, frequency, and phase as that of sensor switching signal. The buffer provides sufficient current for the driven-shield signal to drive the high parasitic capacitance of the hatch fill. When the hatch fill around the sensor is connected to the driven shield signal, it is referred as shield electrode.

![Figure 2-21. Driven Shield Signal](image)

Shield electrode is used for the following purposes:

- To implement liquid-tolerant CapSense designs: Shield electrode helps in making CapSense designs liquid-tolerant as explained above.

- To improve the proximity sensing distance in the presence of floating or grounded conductive objects: A shield electrode when placed between the proximity sensor and a floating or a grounded conductive object reduces the effect of these objects on the proximity-sensing distance and helps in achieving large proximity-sensing distance. See Proximity Sensing section in the Getting Started with CapSense® Design Guide for more details.

- To reduce the parasitic capacitance of the sensor: When a CapSense sensor has a long trace, the $C_p$ of the sensor will be very high because of the increased coupling of sensor electric field lines from the sensor trace to the surrounding ground. By implementing a shield electrode, the coupling of electric field lines to ground is reduced, which results in reducing the $C_p$ of the sensor.

See Layout Guidelines for Liquid Tolerance for layout guidelines of shield electrode.

2.4.3 Guard Sensor

When a continuous liquid stream is applied to the sensor surface, the liquid stream adds a large capacitance ($C_{ST}$) to the CapSense circuitry. This capacitance may be several times larger than $C_{LD}$. Because of this, the effect of the shield electrode is completely masked and the sensor raw counts will be same as or even higher than a finger touch. In such situations, a guard sensor is useful to prevent sensor false triggers.
A guard sensor is a copper trace that surrounds all the sensors on the PCB, as Figure 2-22 shows. A guard sensor is similar to a button sensor and is used to detect the presence of streaming liquids. When a guard sensor is triggered, the firmware disables the scanning of all other sensors except the guard sensor to prevent sensor false triggers.

**Note** Because the sensors are not scanned when the guard sensor is triggered, touch cannot be detected when there is a liquid stream on the touch surface.

**Figure 2-22. Capacitance Measurement with a Liquid Stream**

2.4.3.1 **Effect of Liquid Properties on the Liquid Tolerance Performance**

In certain applications, the CapSense system has to work in the presence of a variety of liquids such as soap water, sea water, and mineral water. In such applications, it is always recommended to tune the CapSense parameters for sensors by considering the worst-case signal due to liquid droplets. To simulate the worst-case conditions, it is recommended that you test the liquid-tolerance performance of the sensors with salty water by dissolving 40 g of cooking salt (NaCl) in one liter of water. Tests were done using soapy water; the results show that the effect of soapy water is similar to the effect of salty water. Therefore, if the tuning is done to reject salty water, the CapSense system will work even in the presence of soapy water.
In applications such as induction cook-tops, there are chances of hot water spilling on to the CapSense touch surface. To determine the impact of the temperature of a liquid droplet on CapSense performance, droplets of water at different temperatures were poured on a sensor, and the corresponding change in raw counts was monitored. Experiment shows that the effect of hot liquid droplets is same as that of the liquid at room temperature as Figure 2-23 shows. This is because the hot liquid droplet cools down immediately to room temperature when it falls on the touch surface. If hot water continuously falls on the sensor and the temperature of the overlay rises because of the hot water, the increase in raw count due to the increase in temperature is compensated by the baseline algorithm, thereby preventing any false triggering of the sensors.

Figure 2-23. Raw Count Variation Versus Water Temperature

To make your design liquid-tolerant, follow these steps:

1. If your application requires tolerance to liquid droplets, implement a shield electrode. If your application requires tolerance to streaming liquids along with liquid droplets, implement a shield electrode and a guard sensor. Follow the schematic and layout guidelines explained in the Layout Guidelines for Liquid Tolerance section to construct the shield electrode and guard sensor respectively.

2. In the CapSense CSD Component, enable the driven-shield signal and specify the “Inactive sensor connection” option as “Shield”. See the Advanced Settings section.

3. If the SmartSense algorithm is used, set the sensitivity of the guard sensor (if implemented) such that it will be triggered only when there is a liquid stream on the touch surface. If manual tuning is used, set the resolution of the guard sensor such that it will be triggered only when there is a liquid stream on the touch surface.
This chapter explains in detail how CapSense CSD is implemented in the PSoC 4 and PRoC BLE devices. See Capacitive Touch Sensing Method to understand the basic principles of CapSense CSD. A basic knowledge of the PSoC 4 or PRoC BLE device architecture is a prerequisite for this chapter. If you are new to PSoC 4, refer to AN79953, Getting Started with PSoC 4, or AN91267, Getting Started with PSoC 4 BLE. If you are new to PRoC BLE, refer to AN94020, Getting Started with PRoC BLE.

You can skip this chapter if you are using the automatic tuning feature (SmartSense™) of the Component. See the CapSense Performance Tuning chapter for details.

### 3.1 CapSense CSD Sensing

Figure 3-1 shows the block diagram of the CapSense block, which scans the CapSense sensors.
3.1.1 GPIO Cell Capacitance to Current Converter

In the CapSense CSD system, the GPIO cells are configured as switched-capacitance circuits that convert sensor capacitances into equivalent currents. Figure 3-2 shows a simplified diagram of the GPIO cell structure.

Figure 3-2. GPIO Cell Structure

PSoC 4 and PRoC BLE devices have two analog multiplexer buses: AMUXBUS A is used for CSD sensing and AMUXBUS B is used for CSD shielding. The GPIO switched-capacitance circuit has two possible configurations: source current to AMUXBUS A or sink current from AMUXBUS A.

3.1.1.1 Current Sourcing

Figure 3-3 shows the switched-capacitance configuration for sourcing current to AMUXBUS A.
Two non-overlapping, out of phase clocks of frequency \( F_{SW} \) (see Figure 3-1 on page 24) control the switches SW\(_2\) and SW\(_3\). The continuous switching of SW\(_2\) and SW\(_3\) forms an equivalent resistance \( R_S \), as Figure 3-3 shows. The value of the equivalent resistance \( R_S \) is:

\[
R_S = \frac{1}{C_S F_{SW}} \quad \text{(3 - 1)}
\]

Where:

\( C_S \) = Sensor capacitance  
\( F_{SW} \) = Frequency of the switching clock

The Sigma Delta converter maintains the voltage of AMUXBUS A at a constant \( V_{REF} \) (this process is explained in Sigma Delta Converter). Figure 3-4 shows the voltage waveform across the sensor capacitance.

Equation 3-2 gives the value of average current supplied to AMUXBUS A.
\[ I_{CS} = C_F S_W (V_{DDD} - V_{REF}) \]  

(3-2)

### 3.1.1.2 Current Sinking

Figure 3-5 shows the switched-capacitance configuration for sinking current from AMUXBUS A. Figure 3-6 shows the resulting voltage waveform across Cs.

**Figure 3-5. Sinking Current from AMUXBUS A**

![Image of sinking current configuration](image)

**Figure 3-6. Voltage Across Sensor Capacitance**

![Image of voltage waveform](image)

Equation 3-3 gives the value of average current taken from AMUXBUS A.

\[ I_{CS} = C_F S_W V_{REF} \]  

(3-3)

### 3.1.2 CapSense Clock Generator

This block generates the switching clock, \( F_{SW} \), and the modulation clock, \( F_{MOD} \), from the clock inputs from system resources, as Figure 3-1 on page 24 shows. The switching clock is required for the GPIO cell switched-capacitance circuits. The switching clock output has three options: direct, 8-bit pseudo random sequence (PRS), and 12-bit PRS.

You can use one of the PRS outputs to lower the Electro Magnetic Interference (EMI) effect – it averages the switching frequency over a wide range.

3.1.3 Sigma Delta Converter

The Sigma Delta converter converts the input current to a corresponding digital count. It consists of a Sigma Delta converter, a clock generator known as a modulator clock divider and two current sourcing / sinking digital to analog converters (IDACs), as Figure 3-1 on page 24 shows.

The Sigma Delta modulator controls the current of the 8-bit IDAC in an on/off manner. This IDAC is known as the modulation IDAC. The 7-bit IDAC, known as the compensation IDAC, is either always ON or always OFF.

The Sigma Delta converter can operate in either single IDAC mode or dual IDAC mode:

- In the single IDAC mode, the modulation IDAC (8-bit IDAC) is controlled by the Sigma Delta modulator; the compensation IDAC (7-bit IDAC) is always OFF.
- In the dual IDAC mode, the modulation IDAC (8-bit IDAC) is controlled by the Sigma Delta modulator; the compensation IDAC (7-bit IDAC) is always ON.

See CapSense Performance Tuning for details.

The Sigma Delta converter also requires an external integrating capacitor $C_{MOD}$, as Figure 3-1 on page 24 shows. The recommended value of $C_{MOD}$ is 2.2 nF.

The Sigma Delta modulator maintains the voltage across $C_{MOD}$ at $V_{REF}$. It works in one of the following modes:

- IDAC sourcing mode: In this mode, the switched-capacitor circuit sinks current from AMUXBUS A, and the IDACs then source current to AMUXBUS B to balance its voltage.
- IDAC sinking mode: In this mode, the IDACs sink current from $C_{MOD}$, and the switched-capacitor circuit sources current to $C_{MOD}$.

In both cases, the modulation IDAC current is switched ON and OFF corresponding to the small voltage variations across $C_{MOD}$ to maintain the $C_{MOD}$ voltage at $V_{REF}$.

The Sigma Delta converter can operate from 8-bit to 16-bit resolutions. In the single IDAC mode, the raw count is proportional to the sensor capacitance. If 'N' is the resolution of the Sigma Delta converter and $I_{MOD}$ is the value of the modulation IDAC current, the approximate value of raw count in the IDAC sourcing mode is given by Equation 3-4.

$$\text{raw count} = (2^N - 1) \frac{V_{REF} F_{SW}}{I_{MOD}} C_S$$

(3 - 4)

Similarly, the approximate value of raw count in IDAC sinking mode is:

$$\text{raw count} = (2^N - 1) \frac{(V_{DD} - V_{REF}) F_{SW}}{I_{MOD}} C_S$$

(3 - 5)

In both cases, the raw count is proportional to sensor capacitance $C_S$. The raw count is then processed by the CapSense CSD Component firmware to detect touches. The hardware parameters such as $I_{MOD}$, $I_{COMP}$, and $F_{SW}$, and the firmware parameters, should be tuned to optimum values for reliable touch detection. For an in-depth discussion of the tuning, see CapSense Performance Tuning.

In dual IDAC mode, the compensation IDAC is always ON. If $I_{COMP}$ is the compensation IDAC current, the equation for the raw count in IDAC sourcing mode is:

$$\text{raw count} = (2^N - 1) \frac{V_{REF} F_{SW}}{I_{MOD}} C_S - (2^N - 1) \frac{I_{COMP}}{I_{MOD}}$$

(3 - 6)

Raw count in IDAC sinking mode is given by equation 3-7.

$$\text{raw count} = (2^N - 1) \frac{(V_{DD} - V_{REF}) F_{SW}}{I_{MOD}} C_S - (2^N - 1) \frac{I_{COMP}}{I_{MOD}}$$

(3 - 7)

Note that raw count values are always positive.

The relation between the parameters shown in the above equation to the CapSense Component parameters is listed in Table 3-1.
Table 3-1. Relation Between CapSense Raw Count Equation and CapSense CSD Parameters

<table>
<thead>
<tr>
<th>Sl.No</th>
<th>Parameter</th>
<th>CapSense Component Parameter</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>N</td>
<td>Scan Resolution</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>V\text{REF}</td>
<td>N/A</td>
<td>The V\text{REF} value is 1.2 V</td>
</tr>
<tr>
<td>3</td>
<td>F\text{SW}</td>
<td>Sense Clock Divider</td>
<td>Refer to CapSense Clocking Section in component datasheet to understand the relation between Sense Clock Divider and F\text{SW}</td>
</tr>
<tr>
<td>4</td>
<td>I\text{MOD}</td>
<td>Modulation IDAC</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>I\text{COMP}</td>
<td>Compensation IDAC</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>V\text{DD}</td>
<td>N/A</td>
<td>This parameter is the device supply voltage</td>
</tr>
<tr>
<td>7</td>
<td>C\text{S}</td>
<td>N/A</td>
<td>This parameter is the sensor parasitic capacitance</td>
</tr>
</tbody>
</table>

3.1.4 Analog Multiplexer

The Sigma Delta converter scans one sensor at a time. An analog multiplexer selects one of the GPIO cells and connects it to the input of the Sigma Delta converter, as Figure 3-1 on page 24 shows. The AMUXBUS A and the GPIO cell switches (see SWs in Figure 3-3 on page 26) form this analog multiplexer. AMUXBUS A connects to all GPIOs that support CapSense. See the device datasheet for a list of ports that support CapSense. AMUXBUS A also connects the integrating capacitor C\text{MOD} to the Sigma Delta converter circuit.

3.2 CapSense CSD Shielding

PSoC 4 / PRoC BLE CapSense supports shield electrodes for liquid tolerance and proximity sensing. See the Liquid Tolerance section for details. The shield electrode is always kept at the same potential as the sensors. CapSense has a shielding circuit that drives the shield electrode with a replica of the sensor switching signal (see GPIO Cell Capacitance to Current Converter) to nullify the potential difference between sensors and shield electrode.

In the sensing circuit, the Sigma Delta converter keeps the AMUXBUS A at V\text{REF} (see Sigma Delta Converter). The GPIO cells generate the sensor waveforms by switching the sensor between AMUXBUS A and a supply rail (either V\text{DD} or ground, depending on the configuration). The shielding circuit works in a similar way; AMUXBUS B is always kept at V\text{REF}. The GPIO cell switches the shield between AMUXBUS B and a supply rail (either V\text{DD} or ground, same configuration as the sensor). This process generates a replica of the sensor switching waveform on the shield electrode.

Depending on how AMUXBUS B is kept at V\text{REF}, two different configurations are possible:

- Shield driving using V\text{REF} buffer: In this configuration, a voltage buffer is used to drive AMUXBUS B to V\text{REF}, as Figure 3-7 shows. An external 10-nF C\text{SH\_TANK} capacitor is recommended to reduce switching transients.

![Figure 3-7. Shield Driving Using V\text{REF} Buffer](image)
- Shield driving using GPIO cell precharge: This configuration requires an external 10 nF $C_{SH, TANK}$ capacitor, as Figure 3-8 shows. A special GPIO cell charges the $C_{SH, TANK}$ capacitor and hence the AMUXBUS B to V$_{REF}$. This is the recommended method for precharging of shield drive.

Figure 3-8. Shield Driving Using GPIO Precharge

[Diagram showing shield driving using GPIO cell precharge]

This GPIO cell precharge capability is available only on a fixed $C_{SH, TANK}$ pin. See the device pinout in the PSoC 4 datasheet, PSoC 4 BLE Datasheet or PRoC BLE Datasheet for details.

Shield drive can be configured through the “$C_{SH, TANK}$ precharge” option in the Advanced Settings tab of CapSense Component.

### 3.3 CapSense in PSoC 4 M-Series

The PSoC 4 M-Series devices support two CapSense CSD blocks - CSD0 and CSD1. Each CSD block has the same functionality and performance as explained in section CapSense CSD Sensing. The main difference between CSD0 and CSD1 block is CSD0 block can scan CapSense sensors on all the GPIOs except Port 5 pins and CSD1 block can scan CapSense sensors on only Port 5 pins as shown in Figure 3-9.

Figure 3-9. CapSense in PSoC 4 M-Series

[Diagram showing CapSense in PSoC 4 M-Series]
Each CSD block requires separate CMOD and CTANK capacitor. The summary of differences between CSD0 and CSD1 block is listed in Table 3-2.

Currently, the PSoC creator software allows using only one CSD block in a project i.e. you can either use CSD0 or CSD1 in a project. Therefore, depending on the number of sensor pins required you should choose the appropriate CSD block for your project. The maximum number of CapSense pins supported by each block is listed in the Table 3-2.

<table>
<thead>
<tr>
<th>CSD0</th>
<th>CSD1</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMOD</td>
<td>P4.2</td>
</tr>
<tr>
<td>CTANK</td>
<td>P4.3</td>
</tr>
<tr>
<td>CapSense Pin</td>
<td>Any pin except PORT5 pins</td>
</tr>
<tr>
<td>Shield Pin</td>
<td>Any pin except PORT5 pins</td>
</tr>
<tr>
<td>Max Number of CapSense Pins</td>
<td>47*</td>
</tr>
</tbody>
</table>

* Max number of pins is specified assuming two pins are used for CMOD and CTANK in the design.

To select a specific CSD block, follow the below procedure:
1. Place the CapSense CSD component in the PSoC Creator schematic.
2. In the PSoC Creator cydwr pins tab, assign the CMOD pin depending on the required CSD block, as shown in Figure 3-10. For example, if you want to use CSD0 block, select CMOD pin as P4.2. PSoC Creator will use CSD0 block for all CapSense operations.

Figure 3-10. Selecting CSD0 or CSD1 Block in PSoC 4 M-Series
Cypress provides a complete set of hardware and software tools to develop your CapSense application.

4.1 **PSoC Creator**

PSoC Creator is a state-of-the-art, easy-to-use integrated development environment. PSoC Creator offers a unique combination of hardware configuration and software development based on classical schematic entry. You can develop applications in a drag-and-drop design environment using a library of Components. For details, see the [PSoC Creator home page](#).

4.1.1 **CapSense_CSD Component**

PSoC Creator provides a CapSense_CSD Component. You can create a capacitive touch system in PSoC or PRoC by simply configuring this Component. The Component also provides an application program interface (API) to simplify firmware development. There are other analog and digital Components available in PSoC Creator to implement additional functionalities such as BLE, I²C, SPI, UART, timers, PWMs, amplifiers, ADCs, and LCDs. Figure 4-1 shows an example of PSoC Creator schematic entry with a CapSense Component dragged from the Component Catalog and placed on the schematic page.

![Figure 4-1. PSoC Creator Component Placement](image-url)
Each Component has an associated datasheet that explains details about the Component. To open the Component datasheet, right-click on the Component and select “Open Datasheet”.

The CapSense Component also has a Tuner GUI to help with the tuning process. See CapSense Performance Tuning for details.

4.1.2 Example Projects

You can use the CapSense example projects provided in PSoC Creator to learn schematic entry and firmware development. To find a CapSense example project, go to the PSoC Creator Start Page, click “Find Example Project”, and select the appropriate architecture, as Figure 4-2 shows.

Figure 4-2. PSoC Creator Example Project

4.2 Hardware Kits

Table 4-1 lists the development kits that support evaluation of PSoC 4 CapSense.

Table 4-1. PSoC 4 CapSense Development Kits

<table>
<thead>
<tr>
<th>Development Kit</th>
<th>Supported CapSense Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSoC 4 Pioneer Kit (CY8CKIT-042)</td>
<td>A 5-segment linear slider</td>
</tr>
<tr>
<td>PSoC 4000 Pioneer Kit (CY8CKIT-040)</td>
<td>A 6 × 5 touchpad and a wire proximity sensor</td>
</tr>
<tr>
<td>PSoC 4 BLE Bluetooth Low Energy Pioneer Kit (CY8CKIT-042-BLE)</td>
<td>A 5-segment linear slider and a wire proximity sensor</td>
</tr>
<tr>
<td>PSoC 4200-M Pioneer Kit (CY8CKIT-044)</td>
<td>A 5-element gesture detection and two-wire proximity sensors</td>
</tr>
<tr>
<td>PSoC 4200 Processor Module (CY8CKIT-038), with PSoC Development Kit (CY8CKIT-001)</td>
<td>A 5-segment linear slider and two buttons</td>
</tr>
<tr>
<td>CapSense Expansion Board Kit (CY8CKIT-031), to be used with CY8CKIT-038 and CY8CKIT-001</td>
<td>A 10-segment slider, 5 buttons and a 4 × 4 matrix button with LED indication.</td>
</tr>
<tr>
<td>MiniProg3 Program and Debug Kit (CY8CKIT-002)</td>
<td>CapSense performance tuning in CY8CKIT-038</td>
</tr>
</tbody>
</table>
Table 4-2 lists the Reference Design Kits that support evaluation of PRoC BLE CapSense.

### Table 4-2. PRoC BLE CapSense Reference Design Kits

<table>
<thead>
<tr>
<th>Reference Design Kit</th>
<th>Supported CapSense features</th>
</tr>
</thead>
</table>
| PRoC BLE Remote Control Reference Design Kit *(CY5672)* | An 8 x 8 trackpad with these gestures:  
  - Top-edge swipe gesture  
  - One-finger click gesture  
  - Two-finger click gesture  
  - Pinch-zoom gesture  
  - One-finger horizontal scroll gesture  
  - One-finger vertical scroll gesture |
| PRoC BLE Touch Mouse Reference Design Kit *(CY5682)* | A 9 x 3 trackpad with these gestures:  
  - One-finger horizontal scroll gesture  
  - One-finger vertical scroll gesture |
The CapSense CSD method is a combination of hardware and firmware techniques. Therefore, it has several hardware and firmware parameters required for proper operation. These parameters should be tuned to optimum values for reliable touch detection and fast response. Most of the capacitive touch solutions in the market must be manually tuned. Cypress provides a unique feature called SmartSense (also known as Auto-tuning) for PSoC 4 and PRoC BLE CapSense. SmartSense is a firmware algorithm that automatically sets all parameters to optimum values. It reduces design cycle time and provides stable performance across PCB variations. SmartSense requires additional RAM and CPU resources.

If you need strict control over the parameters or if the sensor parasitic capacitance $C_p$ is very high, for example higher than 35 pF for 0.1-pF finger capacitance, you can manually tune the CapSense parameters. Manual tuning requires I²C or UART communication with a host PC.

After tuning and finalizing all CapSense parameters, you can turn off the tuning. In this mode, all tuning parameters are stored in flash.

5.1 SmartSense

Some advantages of SmartSense, as opposed to Manual Tuning are:

- **Reduced Design Cycle Time:** The design flow for capacitive touch applications involves tuning all of the sensors. This step can be very time consuming if there are many sensors in your design. In addition, you must repeat the tuning when there is a change in the design, PCB layout, or mechanical design. Auto-tuning solves these problems by setting all of the parameters automatically. Figure 5-1 shows the design flow for a typical CapSense application with and without SmartSense.

![Figure 5-1. Design Flow Without and With SmartSense](image-url)
Performance is independent of PCB variations: The parasitic capacitance of individual sensors can vary due to process variations in PCB manufacturing, or vendor-to-vendor variation in a multi sourced supply chain. If there is significant variation in parasitic capacitance $C_P$ across product batches, the CapSense parameters must be re-tuned for each batch. SmartSense sets parameters for each device automatically, hence taking care of variations in $C_P$.

Ease of use: SmartSense is faster and easier to use because only a basic knowledge of CapSense is needed.

5.1.1 Component Configuration for SmartSense

This section explains the Component configuration for the SmartSense mode. For details on manual tuning, see Manual Tuning.

To open the CapSense Component configuration window (Figure 5-2), either double-click the Component or right-click the Component and select “Configure”.

![CapSense Component General Tab](image)

Figure 5-2. CapSense Component General Tab

5.1.1.1 General Settings

Set the General tab configurations according to Figure 5-2 and as follows:

a. **Tuning method**: Select the Auto (SmartSense) tuning method.

b. **Threshold mode**: Selecting the “Automatic” option is recommended. If you need strict control over the threshold parameters, you can select the “flexible option”. Note that you need to know the manual tuning process to properly set the threshold parameters. See Manual Tuning for details.

c. **Raw data noise filter**: This parameter allows you to select a firmware filter to reduce the noise in raw counts. Table 5-1 on page 37 explains the available filters and their applications.

d. **Waterproofing and detection**: Enable this option if you are using a shield electrode or guard sensor for liquid tolerance. Disable otherwise. See the Liquid Tolerance section for details.

e. **Enable built-in self-test**: Enable this option if you are using the Component APIs to perform a built-in self-test of CapSense. See the Component datasheet for more details on the available built-in self-test APIs. Disabling this option reduces the flash memory usage.
Table 5-1. Raw Data Noise Filters

<table>
<thead>
<tr>
<th>Filter</th>
<th>Description</th>
<th>Mathematical Description</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median</td>
<td>Non linear filter that takes the three most recent samples and computes the median value.</td>
<td>(y[i] = \text{median}(x[i], x[i-1], x[i-2]))</td>
<td>Eliminates noise spikes from motors and switching power supplies</td>
</tr>
<tr>
<td>Average</td>
<td>Finite impulse response filter (no feedback) with equally weighted coefficients. It takes the three most recent samples and computes their average.</td>
<td>(y[i] = \frac{1}{3} \ast (x[i] + x[i-1] + x[i-2]))</td>
<td>Eliminates periodic noise (e.g., from power supplies)</td>
</tr>
<tr>
<td>First order IIR 1/k</td>
<td>Infinite impulse response filter (feedback) with a step response similar to an RC low pass filter, thereby passing the low frequency signals (finger touch responses). A higher (k)-value results in lower noise, but slows down the response.</td>
<td>(y[i] = \frac{1}{k} \ast (x[i] + (k-1)y[i-1]))</td>
<td>Eliminates high frequency noise.</td>
</tr>
<tr>
<td>Jitter</td>
<td>A thick overlay results in a low signal level, and the finger position appears to be shaky even when the finger is stationary. Jitter filter eliminates this noise by comparing the present input value with its previous output value. If the difference is greater than (\pm 1), then the output is changed by (\pm 1).</td>
<td>(y[i] = \begin{cases} x[i] - 1, &amp; x[i] - y[i-1] &gt; 1 \ x[i] + 1, &amp; x[i] - y[i-1] &lt; 1 \ y[i-1], &amp; \text{otherwise} \end{cases})</td>
<td>Noise due to thick overlay. Especially useful for slider centroid data.</td>
</tr>
</tbody>
</table>
5.1.2 Widget Configuration

The “Widgets Config” tab allows you to add and configure CapSense widgets, as Figure 5-3 shows. See CapSense Widgets for details on each widget type. To add a widget, select the type of widget and click “Add”.

Figure 5-3. Adding Widgets

Click on a widget to configure its parameters. Smart Sense automatically sets some of the widget parameters. These parameters are grayed out in the configuration window, as Figure 5-4 shows. See also Figure 5-5 on page 39 and Figure 5-6 on page 40.

Figure 5-4. Configuration of a Button
a. **Debounce**: (see Figure 5-4 on page 38; available for buttons, matrix buttons and proximity sensors). This parameter selects the number of consecutive CapSense scans during which a sensor must be active to generate an ON state from the Component. Debounce ensures that high-frequency, high-amplitude noise does not cause false detection. You can increase the debounce value if the CapSense detects false touches. If not, decrease the debounce value to improve the response time.

b. **Number of Sensor Elements**: (see Figure 5-5; available for Linear Sliders, Radial Sliders and Touchpads). This parameter defines the number of segments within the slider (See Sliders for details). The number of sensor elements depends on the required API resolution of the slider. A good ratio of API resolution to sensor elements is 20:1. Increasing the ratio of API resolution to sensor elements beyond 20:1 may result in noisy finger position.

c. **API resolution**: (see Figure 5-5; available for Linear Sliders, Radial Sliders and Touchpads). This parameter defines the number of discrete finger positions that a slider or a touchpad must resolve. This parameter is different from the resolution of the Sigma Delta converter. A good ratio of API resolution to the number of slider segments is 20:1. Increasing the ratio of API resolution to sensor elements too much can result in increased noise on the calculated finger position.

For example, if your design has five slider segments, set the API resolution as any value between 1 to 100 according to your application requirement.

d. **Diplexing**: (see Figure 5-5; available for Linear Sliders and Radial Sliders). Use diplexing to reduce the number of GPIOs required. Diplexing allows a design to have two slider segments per GPIO pin. For example, a diplexed 16-segment slider requires only eight GPIO pins. See the CapSense Component datasheet for details.

e. **Position Noise filter**: (see Figure 5-5; available for Linear Sliders, Radial Sliders and Touchpads). This parameter selects the type of firmware noise filter used to remove noise from the calculated finger position. The available filters and their applications are explained in Table 5-1 on page 37.
f. **Number of dedicated sensor elements:** (see Figure 5-6; available for proximity sensors.) Select 1 if you are using a dedicated proximity sensor. Set to 0 if you are ganging other sensors together to form a proximity sensor. See **Proximity Sensor** for details.

Figure 5-6. Configuration of a Proximity Sensor
5.1.3 Scan Order

Generally, CapSense performance is not impacted by the order in which the sensors are scanned. However, if you want to change scan order, you can use this tab, as Figure 5-7 shows. To change the scan order, select a sensor and click “Up” or “Down”.

![Figure 5-7. Scan Order](image1)

You can combine other sensors into a proximity sensor by using the drop-down menu on the Scan Order tab, as Figure 5-8 shows. Combining other sensors with the proximity sensor does not affect their operation, however, the resulting proximity sensor may have a high CP and hence a high scan time. Therefore, a proximity sensor should be scanned at a lower rate than the other sensors to avoid long scanning intervals. Proximity sensors are disabled by default. You must enable them using firmware before scanning them. Refer to the Component datasheet for details.

![Figure 5-8. Combining Sensors to Form a Proximity Sensor](image2)
Click on a sensor to change its sensitivity, as Figure 5-9 shows.

Figure 5-9. Sensitivity
a. **Sensitivity:** This parameter sets the expected value of value of finger capacitance $C_F$. See *CapSense Fundamentals* for details on $C_F$. SmartSense multiplies the sensitivity setting by 0.1 pF to get $C_F$. For example, a sensitivity value of 1 represents $C_F = 0.1$ pF and a setting of 4 represents $C_F = 0.4$ pF.

$$C_F = 0.1 \text{ pF} \times \text{ sensitivity} \quad (6 - 1)$$

- If you do not know the value of $C_F$, set the sensitivity at 1 and measure the SNR. If the SNR is less than 5:1, revise your PCB per *PCB Layout Guidelines* or use the *Manual Tuning Process*. If the SNR for all sensors is greater than 5:1, check the sensor performance and set the sensitivity to an optimal value as follows:
  - For button sensors, if the sensor becomes active even before the finger touches it, increase the sensitivity value.
  - For sliders, slide your finger on the slider. If at any slider position, more than two slider segments report a difference count value greater than the noise threshold value, as *Figure 5-9* and *Figure 5-10* show, increase the sensitivity value. If the current sensitivity value results in an SNR of 5:1 on at least one sensor irrespective of where the finger is placed on slider, use this sensitivity value. Else, use the last sensitivity value, which provides an SNR of 5:1 on at least one sensor irrespective of where the finger is placed on slider. The following flowchart explains this process.

*Figure 5-10. Setting sensitivity value for Sliders*
5.1.3.1 Advanced Settings

Advanced settings are available on the Advanced tab, as Figure 5-11 shows.

![Figure 5-11. Advanced Settings](image)

a. **Analog switch drive source**: The options available are Direct, 8-bit pseudo random sequence (PRS), 12-bit PRS, and PRS Auto. The PRS Auto option automatically selects the length of the sequence. You should use PRS if your design has strict electromagnetic compatibility requirements, as it reduces the electromagnetic emission. The other option is to select “Direct” because it provides higher sensitivity compared to PRS. See CapSense Clock Generator for details.

b. **Shield**: You should enable the shield electrode if proximity sensing or liquid tolerance is required, or if your board has a shield electrode to reduce $C_P$ of sensors. See Driven-Shield Signal and Shield Electrode for details.

c. **Shield signal delay**: For proper operation of the shield electrode, the shield signal should exactly match the sensor signal in phase. You can use an oscilloscope to view both sensor and shield signals to verify this condition. If they are not aligned, use this option to add delay to the shield signal to align the two signals. The available delays are 50 nS and 10 nS.

d. **Shield tank capacitor enable**: Enable this option if you are using a $C_{SH\_TANK}$ capacitor; see CapSense CSD Shielding for details.

e. **Guard Sensor**: A guard sensor is used for liquid tolerance. See Guard Sensor for details. Enable this option if you are using a guard sensor in your design.
f. **Sensor auto reset:** Use this setting to avoid latch-up of sensors in high-noise conditions. If enabled, CapSense limits the maximum time duration for which the sensor stays ON (typically 5 to 10 seconds). This setting prevents the sensors from permanently turning on when the raw count accidentally rises because of a large power supply voltage fluctuation, or a sudden change in noise conditions.

g. **Inactive sensor connection:** CapSense scans one sensor at a time. This option determines the connection of sensors when they are not being scanned. The “Ground” option is recommended since it reduces noise on the scanned sensors. For liquid-tolerant designs, this option should be specified as “Shield”.

   **Note:** You should enable the “Shield” option to specify the “Inactive sensor connection” option to “Shield”.

h. **Shield tank capacitor:** This option selects the precharge configuration of $C_{SH,TANK}$. It is recommended to select “Precharge by IO buffer”. See CapSense CSD Shielding for details.

## 5.2 Manual Tuning

Some advantages of manual tuning, as opposed to SmartSense, are:

- **Strict Control over Parameter Settings:** SmartSense sets all of the parameters automatically. However, there may be situations where you need to have strict control over the parameters. For example, use manual tuning if you need to strictly control the time PSOC 4 / PRoC BLE takes to scan a group of sensors.

- **Supports Higher Parasitic capacitances:** SmartSense supports parasitic capacitances as high as 55 pF for 0.2-pF finger capacitance, and as high as 35 pF for 0.1-pF finger capacitance. If the parasitic capacitance is higher than the value supported by SmartSense, you should use manual tuning.

## 5.2.1 Fundamentals of Manual Tuning

This section explains manual tuning in detail. Knowledge of the PSOC 4 / PRoC BLE CapSense architecture is a prerequisite for this section. See Capacitive Touch Sensing Method and CapSense CSD Sensing to become familiar with PSOC 4 / PRoC BLE CapSense architecture. You can skip this section if you are not planning to use manual tuning in your design.

You should use SmartSense if you want a faster and easier tuning method. It requires only a basic knowledge of CapSense.

### 5.2.1.1 Conversion Gain and CapSense Signal

In the single IDAC mode, the raw count is directly proportional to the sensor capacitance.

$$\text{raw count} = G_C C_S$$ (6-2)

Where $G_C$ is the capacitance to digital conversion gain of CapSense CSD. The approximate value of this conversion gain is $2^N \frac{V_{REF} F_{SW}}{I_{MOD}}$ in IDAC sourcing mode, and $2^N \frac{(V_{DD} - V_{REF}) F_{SW}}{I_{MOD}}$ in IDAC sinking mode respectively (See Equations 3-7 and 3-8). The value of $V_{REF}$ is 1.2 volts. For a given resolution, the tunable parameters of the conversion gain are $F_{SW}$ and $I_{MOD}$. Figure 5-12 shows a plot of raw count versus sensor capacitance.
The change in raw counts when a finger is placed on the sensor is called a CapSense signal. Figure 5-13 shows how the value of the signal changes with respect to the conversion gain.

Figure 5-13 shows three plots corresponding to three conversion gain values $G_{C3}$, $G_{C2}$, and $G_{C1}$. An increase in the conversion gain results in higher signal value. However, this increase in the conversion gain also moves the raw count corresponding to $C_P$ towards the maximum value of raw count ($2^N$). For very high gain values, the raw count saturates as the plot of $G_{C3}$ shows. Therefore, you should tune the conversion gain to get a good signal value while avoiding saturation of raw count. Tuning the gain in such a way that the raw count corresponding to $C_P$ is 85% of the maximum raw count is recommended, as Figure 5-14 shows.
The equation for raw count, in the dual IDAC mode is

\[ \text{raw count} = G_C C_S - 2^N \frac{I_{\text{COMP}}}{I_{\text{MOD}}} (6-3) \]

See Equations 3-9 and 3-10 for details. Figure 5-15 shows a plot of raw count versus sensor capacitance in dual IDAC mode.

Figure 5-15 shows that dual IDAC mode yields higher signal than single IDAC mode. Increasing the value of the compensation IDAC increases the \( C_S \) axis intercept:
\[ \frac{I_{\text{COMP}}}{V_{\text{REF}} F_{\text{SW}}} \]

This also increases the slope of the line when the raw count is again tuned to 85% of the maximum value. Therefore, the signal increases when the compensation IDAC value increases.

Note that this increase in signal may not proportionally increase the SNR because of the variations in noise counts when the compensation IDAC is used. Thus, the recommended method to tune a sensor for best SNR using dual IDAC is as explained in Manual Tuning Process on page 61.

5.2.1.2 Switching Clock Selection

For a given resolution N, you can vary both \( F_{\text{SW}} \) and IDACs to make the raw count corresponding to \( C_p \) equal to 85% of the maximum raw count value. However, selecting an improper switching clock \( F_{\text{SW}} \) can affect the operation of CapSense CSD.
The sensor capacitor must be fully charged and discharged during each switching cycle for proper operation of CapSense CSD (see GPIO Cell Capacitance to Current Converter). The charge and discharge paths of the sensor capacitor include series resistances that slow down the charging / discharging process. Figure 5-16 shows an equivalent circuit and resulting waveforms.

The typical value of the GPIO resistance is 500 $\Omega$ and the recommended external resistance is 560 $\Omega$ (see Series Resistors on CapSense Pins for details). Therefore, take the value of $R_S$ as 1.06 k$\Omega$ when calculating the maximum switching frequency.
5.2.1.3 Signal-to-Noise Ratio

In practice, the raw counts vary due to noise in the system. CapSense noise is the peak-to-peak variation in raw counts in the absence of a touch, as Figure 5-17 shows.

A well-tuned CapSense system reliably discriminates between the ON and OFF states of the sensors. To achieve good performance, the CapSense signal must be significantly larger than the CapSense noise. Signal-to-noise Ratio (SNR), which is defined as the ratio of CapSense signal to CapSense noise is the most important performance parameter of a CapSense sensor.

![Figure 5-17. SNR](image)

In this example, the average level of raw count in the absence of a touch is 5925 counts. When a finger is placed on the sensor, the average raw count increases to 6060 counts, therefore the signal is 6060 – 5925 = 135 counts. The minimum value of raw count in the OFF state is 5912 and the maximum value is 5938 counts. Therefore the CapSense noise is 5938 – 5912 = 26 counts. This results in an SNR of 135 / 26 = 5.2.

The minimum SNR recommended for a CapSense sensor is 5. In other words, the signal should be at least five times larger than the noise.
5.2.1.4 Baseline

The raw count value of a sensor may vary due to changes in the environment such as temperature and humidity. Therefore, the raw count is low pass filtered to create a new count value known as baseline that keeps track of gradual changes in raw count. The baseline is less sensitive to sudden changes in the raw count caused by a touch. Therefore, the baseline value provides the reference level for computing the signals. Figure 5-18 shows the concept of raw count, baseline and signal.

Figure 5-18. Raw Count and Baseline
5.2.1.5 Tuning Parameters

The CapSense Component uses several tuning parameters for proper operation of CapSense Hardware tuning parameters include CSD resolution N, IDAC values, and analog switch frequency $F_{SW}$ – see Sigma Delta Converter for details. These are the software parameters.

- **Finger Threshold:** The finger threshold parameter controls the sensitivity of a sensor to finger touches. It is used along with the hysteresis parameter to determine the sensor state, as Equation 6-6 shows.

  \[
  \text{Sensor State} = \begin{cases} 
  \text{ON} & \text{if } (\text{Signal} \geq \text{Finger Threshold} + \text{Hysteresis}) \\
  \text{OFF} & \text{if } (\text{Signal} \leq \text{Finger Threshold} - \text{Hysteresis}) 
  \end{cases} \tag{6-6}
  \]

- **Hysteresis:** The hysteresis parameter is used along with the finger threshold parameter to determine the sensor state, as Equation 6-6 and Figure 5-19 show. Hysteresis provides immunity against noisy transitions of sensor state. The hysteresis parameter setting must be lower than the Finger Threshold parameter setting.

![Figure 5-19. Hysteresis](image1)

- **Noise Threshold:** For single sensor widgets such as buttons and proximity sensors, the noise threshold parameter sets the raw count limit above which the baseline is not updated, as Figure 5-20 shows. In other words, the baseline remains constant as long as the raw count is above `baseline + noise threshold`. This keeps the baseline from becoming too high due to a finger touch. You should set the Noise Threshold value below `finger threshold - hysteresis`.

- **Negative Noise Threshold:** If the raw count is $<$ (baseline - negative noise threshold) for the number of samples specified by the low baseline reset parameter, the baseline is reset to the new raw count value. This change in baseline resets any sensor that is stuck in the active state during the device power ON.

![Figure 5-20. Finger Threshold](image2)
- Low Baseline Reset parameter: This parameter is used together with the negative noise threshold parameter. It counts the number of abnormally low raw counts required to reset the baseline. It is used to reset the baseline if the finger is placed on the sensor during device startup and later removed.

- Debounce: This parameter selects the number of consecutive CapSense scans during which a sensor must be active to generate an ON state from the Component. Debounce ensures that high-frequency, high-amplitude noise does not cause false detection.

\[
\text{Sensor State} = \begin{cases} 
\text{ON} & \text{if } (\text{Signal} \geq \text{Finger Threshold} + \text{Hysteresis}) \text{ for } \text{scans} \geq \text{debounce} \\
\text{OFF} & \text{if } (\text{Signal} \leq \text{Finger Threshold} - \text{Hysteresis}) \\
\text{OFF} & \text{if } (\text{Signal} \geq \text{Finger Threshold} + \text{Hysteresis}) \text{ for } \text{scans} < \text{debounce}
\end{cases}
\]
5.2.2 Manual Tuning Process

Figure 5-21 illustrates the manual tuning process. Follow the flow chart to manually set all the tuning parameters. See the following pages for details on each step.

Figure 5-21. Manual Tuning Process

Start

1. Configure the Component for Manual Tuning, Program the Project and Open Tuner GUI

2. Set the Switching Frequency (HFCLK and Analog Switch Divider)

3. Set the Sigma Delta Converter Clock Frequency (Modulation Divider)

4. Set the Resolution as 8 Bits

5. Set the value of I_{MOD}, I_{COMP}

6. Monitor the Results in the Tuner GUI With and Without Finger Touch

7. Add Raw Data Noise Filters. Choose the Type of Filter Based on the Type of Noise

8. If (SNR > 5) & (Scan Time < Requirement)

9. Increase Resolution

10. If (SNR > 5) & (Scan Time < Requirement)

11. Check the Layout for Improvement. If Possible, Reduce the Overlay Thickness or Increase the Button Size

12. If (SNR > 5) & (Scan Time < Requirement)

13. Set Finger Threshold to 80% of Peak Finger Response

14. Set the Noise Threshold & Negative Noise Threshold to 50% of the Finger Threshold

End
1. The first step in manual tuning is configuring the Component and Tuner GUI. If you are familiar with the Component and Tuner GUI settings, you can skip to the next step.

Configure the Component for manual tuning, as Figure 5-22 shows.

**Figure 5-22. General Settings**

![Configure CapSense_CSD_P4](image_url)

**a. Tuning method**: Select “Manual with run-time tuning”. This option allows you to change the tuning parameters during run-time.

**b. Raw data noise filter**: This parameter allows you to select a firmware filter to reduce the noise in raw counts. Table 5-1 on page 37 explains the available filters and their applications.

**c. Compensation IDAC**: Enabling the compensation IDAC selects the dual IDAC mode operation of the sigma delta converter. Disabling this IDAC selects single IDAC mode. See Sigma Delta Converter for more details.

Manual tuning in the single IDAC mode is easier than the dual IDAC mode. However, the dual IDAC mode gives higher signal values. See Fundamentals of Manual Tuning for more details.

**d. Auto-calibration**: The parasitic capacitance of sensors may change from one PCB to another. This in-turn results in a change in the sensor tuning. Enabling this option maintains the tuning by automatically calibrating the IDACs. Note that this option does not affect other parameters such as scan resolution, clocks and thresholds. Therefore, this option is not a replacement for SmartSense.

The configuration of the widgets is explained in the CapSense Widgets and Scan Order sections. However, selecting manual tuning exposes some additional parameters that are explained in the Tuning Parameters section. Leave these parameters at their default values. You must configure these parameters using the Tuner GUI during the tuning process.
The Advanced tab also shows some additional parameters, as Figure 5-23 shows.

**Figure 5-23. Advanced Settings**

- **Current source**: This option selects the operating mode of the sigma delta converter. See *Sigma Delta Converter* for details. The IDAC sourcing mode is recommended for most applications, as it is free from power supply noise compared to the IDAC sinking mode. However, if you have a low-noise power supply, you can use the IDAC sinking mode to reduce finger-conducted noise.

- **IDAC range**: The options available are 4X and 8X. The 4X range is sufficient for most applications. The 8-bit IDAC provides a current of 0–306 µA in this range. You can use the 8X range if the C_P is very high. The 8-bit IDAC provides a current of 0–612 µA in this range.

- **Individual frequency settings**: Enable this option if your sensors do not have similar C_P values; disable this option if they have similar values.

- **Widget resolution**: This parameter selects between 8-bit or 16-bit variables for signal count. In most cases, you should use the 8-bit widget resolution. If you have more signal than an 8-bit variable can handle, it means that you are using a higher resolution than required, and the resolution can be reduced to save scan time.
However, 16 bits of widget resolution is useful in cases where the signal is too high because of thin overlays. In such cases, if widget resolution is just 8 bits, the signal saturates.

The remaining settings are similar to SmartSense settings. See Advanced Settings for how to configure these settings. After configuring the “Advanced” tab, follow the steps below to start tuning. These CapSense parameters can also be tuned by monitoring the sensor data such as raw count, baseline and difference count using UART communication. For a detailed explanation on how to use UART communication for sensor tuning refer to application note AN2397 - PSoC 1 and CapSense Controllers – CapSense Data Monitoring Tools.

i. Go to the “Tune Helper” tab, enable the tune helper, and give an instance name for the SCB Component, as Figure 5-24 shows.

![Figure 5-24. Enabling Tune Helper](image)

ii. Drag and drop an EZI2C Slave (SCB mode) Component to the TopDesign. Rename the Component to match the instance name given in the “Tune Helper” tab of the CapSense Component (see the previous step). Configure the EZI2C parameters as Figure 5-25 shows.

![Figure 5-25. Enabling Tune Helper](image)

iii. To do manual tuning, you must include in two API function calls: CapSense_1_TunerStart(), and CapSense_1_TunerComm(), as the following example shows.
#include <device.h>
void main()
{
    CyGlobalIntEnable; /* Global Interrupt enable */
    /* CapSense Block and Tuner Communication power up and initialization */
    CapSense_1_TunerStart();

    while(1)
    {
        /*Scan all the sensors and send the result to PC */
        CapSense_1_TunerComm();
    }
}

After finishing the manual tuning process, you can delete the function calls from your code. Note that they are not needed for SmartSense.

iv. After programming the device, connect EZI2C to your PC using a MiniProg3 or the I2C-USB bridge in your development kit (see the kit user guide for details).

v. Right-click on the CapSense Component and select “Launch Tuner”.
vi. Open the Tuner Communication Setup window by clicking on the configuration button in the tuner window, as Figure 5-26 shows.

Figure 5-26. Tuner GUI
Click on the MiniProg3 / I²C-USB bridge in the Tuner Communication Setup window and set the parameters, as Figure 5-27 shows:

**Figure 5-27. Tuner Communication Setup**

![Tuner Communication Setup Window](image)

**Note** The voltage settings in this window must match the voltage settings on the board.

Click OK.

Click the “Start” button in the tuner window to initiate I²C communication, as Figure 5-28 shows.

**Figure 5-28. Starting Tuner Communication**

![Starting Tuner Communication](image)
You can set the parameters for each sensor at the lower right corner of this window, as Figure 5-29 shows. Clicking OK copies the parameters from the tuner GUI to CapSense Component.

Figure 5-29. Setting the Parameters for a Sensor
xi. The Graphing tab allows you to monitor signals, as Figure 5-30 shows. You can log data using the Logging tab.

Figure 5-30. Monitoring CapSense Results

2. Set the sense clock divider using the Tuner GUI to set the switching frequency $F_{SW}$ so that the sensor capacitor is fully charged and discharged during each switching cycle. See Switching Clock Selection for details.
   - If you know the parasitic capacitance, $C_P$, of the sensors, you can calculate the required switching frequency by using the equations provided in Switching Clock Selection.
   - If you do not know the $C_P$ of the sensors, view the switching waveform on each sensor pin to make sure that they are charging and discharging properly. Note that, while observing the sensor voltage, the probe capacitance is added to the sensor capacitance. Using probes in the 10X mode reduces the capacitance. Use a FET input probe if available. Increase the sense clock divider, if the sensor is not charging and discharging properly.

3. Set the modulator clock divider for each sensor. For PSoC 4100/4200 devices, the clock dividers are cascaded. Therefore, the sense clock divider should be an integral multiple of the modulator clock divider. The CapSense customizer does not allow other ratios. For PSoC 4000/4100M/4200M/4100-BL/4200-BL devices, the dividers are not cascaded and any ratio is allowed. However, it is recommended to set the sense clock divider to be an integral multiple of the modulator clock divider. Other ratios would result in a small increase in rawcount noise. Selecting a higher modulator clock divider gives an improved SNR. However, the total scan time increases as the modulator clock divider increases.
   - You should initially set the modulator clock divider at 1/4th of the sense clock divider. After tuning is complete, if the SNR is good, reduce the modulator clock divider to reduce the scan time; else increase it.

4. Set the scan resolution $N$ to 8 bits. Increasing the scan resolution increases the signal, but it also increases scan time. You can view the scan time required for each sensor and the total scan time in the “Scan order” tab of the Component configuration window.

5. **Single IDAC mode:** Change the modulation IDAC value in the GUI until the raw counts reach 85% of the maximum raw counts ($2^N$). Note that decreasing the modulation IDAC value increases the raw counts and vice versa. If it is not possible to achieve 85 percent with any of the IDAC values then change the IDAC range in the Component configuration. (See IDAC range.)
   - **Dual IDAC mode:** In the dual IDAC mode, you must tune both the modulation IDAC and the compensation IDAC. This is a two-iteration process.
     - Initially, tune the raw counts to 85% of the maximum raw counts by only using the modulation IDAC (keep the compensation IDAC at zero) and keep a note of this modulation IDAC value. Let this modulation IDAC value be $I_{MOD85}$. Then, set the compensation IDAC and modulation IDAC value to $I_{MOD85}/2$. 
6. Monitor the SNR in the “Tuning” tab of the Tuner GUI. You can also view the raw counts and manually calculate the SNR (see Signal-to-Noise Ratio). If the SNR is above 5, the current tuning is good.
   - You should also verify that the total scan time meets the design requirements. The “scan order” tab in the Component Configuration window shows the scan time of each sensor. If the total scan time is very high, the response of the CapSense sensors may be slow.
   - If the total scan time does not meet the design requirements, reduce the modulator clock divider. Make sure that the SNR is not going below 5 when reducing the modulator clock divider.
   - If both SNR and scan time are satisfactory, go to step 12.

7. If the SNR requirement is not met, then you should enable filters in the Component. See filter selection and select the type of filter that can eliminate the noise present in the system. After enabling the filters, build the project again, program the device, and launch the Tuner.

8. Check the SNR and scan time as Step 6 explains. If the results are satisfactory, go to step 12. If the scan time requirement is not met, then go to step 11.

9. If the SNR is below 5, then you should increase the resolution. Note that increasing the resolution also increases the scan time. If an increase in scan time does not meet the design requirements, use dual IDAC mode. See General Settings for details.

10. Check the SNR and scan time as Step 6 explains. If the results are satisfactory, go to step 12. If scan time requirement is not met, try decreasing the modulator clock divider as step 6 explains.

11. If you are unable to achieve an SNR of 5, while keeping the scan time within the requirement, then you should improve your PCB layout and sensor design. Try reducing the parasitic capacitance, overlay thickness, or increase the button size. See Design Considerations for details.

12. If you achieved good SNR and scan time, you should set the remaining tuning parameters (see Tuning Parameters). Set the finger threshold at 80 percent of the signal value.

13. Set the hysteresis to 12.5 percent of the finger threshold.

14. Set the noise threshold and negative noise threshold to 50 percent of the finger threshold.

15. Set the low baseline reset parameter to 30.

Now press “OK” in the tuner to copy all the parameters to the CapSense Component. Repeat the procedure for all the sensors in the design.

To validate your design, click the “Validation” tab in the tuner GUI and follow the onscreen instructions. If you see any false touch detections, increase the debounce value of the sensors (see Debounce). Note that increasing the debounce also increases scan time. If the resulting scan time is not within your requirements, then you must re-tune the sensor.

Note After finishing the manual tuning, you should change the tuning method in the Component configuration form “Manual with run-time tuning” to “Manual”. The “Manual” option stores the tuning parameters in flash and these cannot be changed during run-time.
5.2.2.1  Manual Tuning For Slider

A slider has many segments, each of which is connected to the CapSense input pins of the PSoC 4 / PRoC BLE device. The slider layout design should ensure that the CP of all the segments in a slider remains same. If all the slider segments have the same CP, tuning will be easier. To avoid nonlinearity in the centroid, you need to ensure that the signal from all the slider segments is equal, as Figure 5-31 shows. Here, the signal for each slider segments is the shift in raw count minus the Noise Threshold, when a finger is placed at the center of the slider segment. If the signal of the slider segments is different, then the centroid will be nonlinear, as Figure 5-32 shows.

Figure 5-31. Response of Centroid Versus Finger Location when Signals of All Slider Elements Are Equal

![Graph showing response of centroid versus finger location when signals of all slider elements are equal.](image)

**Note** Signal = Difference count – Noise Threshold

Difference count = Raw Count - Baseline

Figure 5-32. Response of Centroid Versus Finger Location when the Signal of All Slider Elements Are Different

![Graph showing response of centroid versus finger location when the signal of all slider elements are different.](image)
Use the following steps to manually tune the slider segments to achieve an equal signal for all slider segments:

1. Measure the $C_P$ of the slider segments using the $C_P$ measurement API CapSense_GetSensorCp(). You need to check the **Enable built-in self-test** checkbox in the General tab of the CapSense CSD Component configuration window to use this API.

2. Follow the manual tuning procedure shown in Figure 5-21 and tune the slider segment which has the maximum $C_P$ value among all the slider segments.

3. For the remaining segments in the slider, set the same value as that of the maximum $C_P$ slider segment for the following parameters:
   - Scan resolution
   - Sense clock divider
   - Modulator clock divider
   - Modulation IDAC
   - Compensation IDAC

   Setting the same values for these parameters for all segments will ensure that the sensitivity (change in the raw count for a given change in $C_F$) is the same for all sensors.

   **Note** If Compensation IDAC is enabled, the tuning algorithm described earlier requires that the $C_P$ of other segments in the slider should be greater than 75% of the $C_P$ of the segment with the maximum value in that slider. For example, in a slider, if 30 pF is the maximum $C_P$ of a segment, the $C_P$ of other segments should be greater than 22.5 pF.

4. Measure the **SNR** for each slider segment and ensure that it is greater than 5:1 for all the slider segments. If the SNR is not greater than 5:1 for any slider segment, increase the resolution by one bit for the slider segment with the maximum $C_P$, and repeat from Step 2 until the SNR is greater than 5:1 for all segments.

5. After confirming that the SNR for all slider segments is greater than 5:1, set the following thresholds to the value listed in **Table 5-2**.

### Table 5-2: Threshold Parameter Values

<table>
<thead>
<tr>
<th>Threshold Parameter</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finger Threshold</td>
<td>80% of the Signal</td>
</tr>
<tr>
<td>Noise Threshold</td>
<td>40% of the Signal</td>
</tr>
<tr>
<td>Negative Noise Threshold</td>
<td>40% of the Signal</td>
</tr>
<tr>
<td>Low-Baseline Reset</td>
<td>Set to 30</td>
</tr>
</tbody>
</table>
6. Design Considerations

This chapter explains firmware and hardware design considerations for CapSense.

6.1 Firmware

The PSoC 4 / ProC BLE CapSense Component provides multiple application programming interfaces to simplify firmware development. The CapSense Component datasheet provides a detailed list and explanation of the available APIs. You can use the CapSense example projects provided in PSoC Creator to learn schematic entry and firmware development. See Chapter 4 for more details.

The CapSense scan is non-blocking in nature. The CPU intervention is not required between the start and the end of a CapSense scan. Therefore, you can use CPU to perform other tasks while a CapSense scan is in progress. However, note that CapSense is a high sensitive analog system. Therefore, sudden changes in the device current may increase the noise present in the rawcounts. If you are using widgets that require high sensitivity such as proximity sensors, or buttons with thick overlay, you should use a blocking scan. Example firmware for a blocking scan is shown below:

```c
/* Start the CapSense component */
CapSense_CSD_Start();

/* Initialize baselines */
CapSense_CSD_InitializeAllBaselines();

for(;;)
{
    /* Disable and clear all the non-CapSense interrupts used in the project before staring the CapSense scan*/

    /* Update all baselines */
    CapSense_CSD_UpdateEnabledBaselines();

    /* Start scanning all enabled sensors */
    CapSense_CSD_ScanEnabledWidgets();

    /* Wait for scanning to complete. Do not run any other code while the CapSense scan is in progress*/
    while(CapSense_CSD_IsBusy() != 0);

    /* Enable all the non-CapSense interrupts after the scan */

    /* Rest of the user code*/
}
```
You should avoid interrupted code, power mode transitions, and switching ON/OFF peripherals while a high-sensitivity CapSense scan is in progress. However, if you are not using high-sensitivity widgets, you can use CPU to perform other tasks. You can also use low-power modes of PSoC 4 / PRoC BLE to reduce the average power consumption of the CapSense system, as explained in the next section. Monitoring and verifying the raw counts and SNR using the tuner GUI is recommended if you are using a non-blocking code.

6.1.1 Low-Power Design

PSoC 4 and PRoC BLE’s low-power modes allow you to reduce overall power consumption while retaining essential functionality. See AN86233, PSoC 4 Low-Power Modes and Power Reduction Techniques, for a basic knowledge of PSoC 4’s low-power modes.

The CPU intervention is not required between the start and the end of a CapSense scan. If the firmware does not have any additional task than waiting for the scan to finish, you can put the device to Sleep mode after initiating a scan to save power. When the CSD hardware completes the scan, it generates an interrupt to return the device to the Active mode.

If you use APIs that scan multiple sensors together, the device returns to Active mode after finishing the scan of a single sensor. Therefore, if you have multiple sensors in your design, you should scan each one individually.

You can use the Deep-Sleep mode of PSoC 4 or PRoC BLE to considerably reduce the power consumption of a CapSense design. However, the CapSense hardware is disabled in the Deep-Sleep mode. Therefore, the device must wake up frequently to scan for touches. You can use the Watchdog Timer (WDT) in PSoC 4 / PRoC BLE to wake up the device from the Deep-Sleep mode at frequent time intervals. Increasing the frequency of the scans improves the response of the CapSense system, but it also increases the average power consumption.

As the number of sensors in the design increases, the device has to spend more time in the Active mode to scan all sensors. This, in turn, increases the average power consumption. If a design has many sensors, you should include a separate proximity sensor loop that surrounds the sensors. When the device wakes up from the Deep-Sleep mode, only scan this proximity sensor. If the proximity sensor is active, the device must stay in the Active mode and scan other sensors. If the proximity sensor is inactive, the device can return to the Deep-Sleep mode. Figure 6-1 illustrates this process.
To further reduce the power consumption, you can make the device enter the Sleep mode when the CPU activity is not required, but other high-speed peripherals such as system timers and I²C are required.

**Note:** In PSoC 4000 devices, it is not recommended to enter Sleep mode if a CapSense scan is in progress.

You can also add a shield hatch in the design to reduce the parasitic capacitance and hence the scan time. You can achieve very low system currents while maintaining a good touch response, by properly tuning CapSense and the wakeup interval.

### 6.2 Sensor Construction

Figure 6-2 shows the most common CapSense sensor construction.
The copper pads etched on the surface of the PCB act as CapSense sensors. A nonconductive overlay serves as the touch surface. The overlay also protects the sensor from the environment and prevents direct finger contact. A ground hatch surrounding the sensor pad isolates the sensor from other sensors and PCB traces.

If liquid tolerance is required, you should use a shield hatch instead of the ground hatch. In this case, drive the hatch with a shield signal instead of connecting to ground. See Liquid Tolerance for details.

The simplest CapSense PCB design is a two layer board with sensor pads and hatched ground plane on the top, and the electrical components on the bottom. Figure 6-3 shows an exploded view of the CapSense hardware.
6.2.1 Overlay Material

The overlay is an important part of CapSense hardware as it determines the magnitude of finger capacitance. The finger capacitance is directly proportional to the relative permittivity of the overlay material. See finger capacitance for details.

Table 6-1 shows the relative permittivity of some common overlay materials. Materials with relative permittivity between 2.0 and 8.0 are well suited for CapSense overlay.

Table 6-1. Relative Permittivities of Overlay Materials

<table>
<thead>
<tr>
<th>Material</th>
<th>( \varepsilon_r )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Air</td>
<td>1.0</td>
</tr>
<tr>
<td>Formica (Standard)</td>
<td>4.6 – 4.9</td>
</tr>
<tr>
<td>Glass (Ceramic)</td>
<td>3.2</td>
</tr>
<tr>
<td>PET Film (Mylar®)</td>
<td>6.0</td>
</tr>
<tr>
<td>Polycarbonate (Lexan®)</td>
<td>2.9 – 3.0</td>
</tr>
<tr>
<td>Acrylic (Plexiglass®)</td>
<td>2.8</td>
</tr>
<tr>
<td>ABS</td>
<td>2.4 – 4.1</td>
</tr>
<tr>
<td>Wood Table and Desktop</td>
<td>1.2 – 2.5</td>
</tr>
<tr>
<td>Gypsum (Drywall)</td>
<td>2.5 – 6.0</td>
</tr>
</tbody>
</table>

Note Conductive materials interfere with the electric field pattern. Therefore, you should not use conductive materials for overlay. You should also avoid using conductive paints on the overlay.

6.2.2 Overlay Thickness

Finger capacitance is inversely proportional to the overlay thickness. Therefore, a thin overlay gives more signal than a thick overlay. See finger capacitance for details.

Table 6-2 lists the recommended maximum thickness of acrylic overlay for different CapSense widgets.

Table 6-2. Maximum Thickness of Acrylic Overlay

<table>
<thead>
<tr>
<th>Widget</th>
<th>Maximum thickness (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Button</td>
<td>5</td>
</tr>
<tr>
<td>Slider</td>
<td>5(^1)</td>
</tr>
<tr>
<td>Touchpad</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Because finger capacitance also depends on the dielectric constant of the overlay, the dielectric constant also plays a role in the guideline for the maximum thickness of the overlay. Common glass has a dielectric constant of approximately \( \varepsilon_r = 8 \), while acrylic has approximately \( \varepsilon_r = 2.5 \). The ratio of \( \varepsilon_r/2.5 \) is an estimate of the overlay thickness relative to plastic for the same level of Sensitivity. Using this rule of thumb, a common glass overlay can be about three times as thick as a plastic overlay while maintaining the same level of sensitivity.

\(^1\) For a 5-mm acrylic overlay, the SmartSense Component requires a minimum of 9-mm finger diameter for slider operation. If the finger diameter is less than 9 mm, manual tuning should be used.
6.2.3 Overlay Adhesives

The overlay must have a good mechanical contact with the PCB. You should use a nonconductive adhesive film for bonding the overlay and the PCB. This film increases the sensitivity of the system by eliminating the air gap between the overlay and the sensor pads. 3M™ makes a high-performance acrylic adhesive called 200MP that is widely used in CapSense applications. It is available in the form of adhesive transfer tapes; example product numbers are 467MP and 468MP.

6.3 PCB Layout Guidelines

PCB layout guidelines help you to design a CapSense system with good sensitivity and high SNR.

6.3.1 Parasitic Capacitance, CP

The main components of CP are trace capacitance and sensor capacitance. See CapSense Fundamentals for details. The relationship between CP and the PCB layout features is not simple. CP increases when:

- sensor pad size increases
- trace length and width increases
- gap between the sensor pad and the ground hatch decreases

You should decrease the trace length and width as much as possible to reduce CP. Reducing trace length increases noise immunity. Reducing the sensor pad size is not recommended as it also reduces the finger capacitance.

Another way to reduce CP is to increase the gap between the sensor pad and ground hatch. However, widening the gap also decreases noise immunity. You can also reduce CP by driving the hatch with a shield signal.

See for details.

6.3.2 Board Layers

Most applications use a two-layer board with the sensor pads and the hatched ground planes on the top side and all other components on the bottom side. PCBs that are more complex use four layers. FR4-based PCB designs perform well with board thickness ranging from 0.020 inches (0.5 mm) to 0.063 inches (1.6 mm).

Flex circuits work well with CapSense. You should use flex circuits for curved surfaces. All PCB guidelines in this document also apply to flex. You should use flex circuits with thickness 0.01 inches (0.25 mm) or higher for CapSense. The high breakdown voltage of the Kapton® material (290 kV/mm) used in flex circuits provides built in ESD protection for the CapSense sensors.

6.3.3 Button Design

You should use circular sensor pads for CapSense buttons. Rectangular shapes with rounded corners are also acceptable. However, you should avoid sharp corners (less than 90°) since they concentrate electric fields. Figure 6-4 shows recommended button shapes.

![Figure 6-4. Recommended Button Shapes](image-url)
Button diameter should be 5 mm to 15 mm, with 10 mm suitable for most applications. A larger diameter is appropriate for thicker overlays.

The width of the gap between the sensor pad and the ground hatch should be equal to the overlay thickness, and range from 0.5 mm to 2 mm. For example, if the overlay thickness is 1 mm, you should use a 1-mm gap. However, for a 3-mm overlay, you should use a 2-mm gap.

Select the spacing between two adjacent buttons such that when touching a button, the finger is not near the gap between the other button and the ground hatch, to prevent false touch detection on the adjacent buttons, as Figure 6-5 shows.

6.3.4 Slider Design

Figure 6-6 shows the recommended slider pattern for a linear slider and Table 6-3 shows the recommended values for each of the linear slider dimensions. A detailed explanation on the recommended layout guidelines is provided in the following sections.

### Table 6-3. Linear Slider Dimensions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Acrylic Overlay Thickness</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Recommended</th>
</tr>
</thead>
<tbody>
<tr>
<td>Width of the segment (W)</td>
<td>1 mm</td>
<td>2 mm</td>
<td>-</td>
<td>8 mm¹</td>
</tr>
<tr>
<td></td>
<td>3 mm</td>
<td>4 mm</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 mm</td>
<td>6 mm</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Height of the segment (H)</td>
<td>-</td>
<td>7 mm²</td>
<td>15 mm</td>
<td>12 mm</td>
</tr>
<tr>
<td>Air gap between segments (A)</td>
<td>-</td>
<td>0.5 mm</td>
<td>2 mm</td>
<td>0.5 mm</td>
</tr>
<tr>
<td>Air gap between the hatch and the slider (A₁⁻)</td>
<td>-</td>
<td>0.5 mm</td>
<td>2 mm</td>
<td>Equal to overlay thickness</td>
</tr>
</tbody>
</table>

¹ The recommended slider-segment width is based on an average human finger diameter of 9 mm. Refer to section 6.3.4.1, “Slider-Segment Shape, Width, and Air Gap” for more details.

² The minimum slider segment height of 7 mm is recommended based on a minimum human finger diameter of 7 mm. Slider height may be kept lower than 7 mm if the overlay thickness and CapSense tuning is such that an SNR ≥ 5:1 is achieved when the finger is placed in the middle of any segment.
6.3.4.1 Slider-Segment Shape, Width, and Air Gap

A linear response of the reported finger position (that is, the Centroid position) versus the actual finger position on a slider requires that the slider design is such that whenever a finger is placed anywhere between the middle of the segment SLD0 and middle of segment SLDn-1, other than the exact middle of slider segments, exactly two sensors report a valid signal\(^1\). If a finger is placed at the exact middle of any slider segment, the adjacent sensors should report a difference count = noise threshold. Therefore, it is recommended that you use a double-chevron shape as Figure 6-6 shows. This shape helps in achieving a centroid response close to the ideal response, as Figure 6-7 and Figure 6-8 show. For the same reason, the slider-segment width and air gap (dimensions “W” and “A” respectively, as marked in Figure 6-6) should follow the relationship mentioned in Equation 6-1.

\[
W + 2A = \text{finger diameter}
\]

\(^1\) Here, a valid signal means that the difference count of the given slider segment is greater than or equal to the noise threshold value.
Typically, an average human finger diameter is approximately 9 mm. Based on this average finger diameter and Equation 6-1, the recommended slider-segment-width and air-gap is 8 mm and 0.5 mm respectively.

If the \( \text{slider-segment-width} + 2 \times \text{air-gap} \) is lesser than \( \text{finger diameter} \), as required per Equation 6-1, the centroid response will be non-linear. This is because, in this case, a finger placed on the slider will add capacitance, and hence valid signal to more than two slider-segments at some given position, as Figure 6-9 shows. Thus, calculated centroid position per Equation 6-2 will be non-linear as Figure 6-10 shows.

**Figure 6-9. Finger Causes Valid Signal on More Than Two Segments When Slider Segment Width Is Lower Than Recommended**

![Figure 6-9 Diagram](image)

Equation 6-2. Centroid algorithm used by CapSense Component in PSoC Creator

\[
\text{Centroid position} = \frac{(S_{x+1} - S_{x-1} + \text{maximum}) \times \text{Resolution}}{(n - 1)}
\]

Resolution – API Resolution set in the CapSense Component Customizer
n – Number of sensor elements in the CapSense Component Customizer
maximum: Index of element which gives maximum signal
Si – different counts (with subtracted Noise Threshold value) near by the maximum position
Note that even though a slider-segment-width value of less than finger diameter – 2 * air-gap provides a non-linear centroid response, as Figure 6-10 shows; it may still be used in an end application where the linearity of reported centroid vs actual finger position does not play a significant role. However, a minimum value of slider-segment-width must be maintained, based on overlay thickness, such that, at any position on the effective slider length, at-least one slider-segment provides an SNR of >=5:1 (i.e. signal >= Finger Threshold parameter) at that position. If the slider-segment-width is too low, a finger may not be able to couple enough capacitance, and hence, none of the slider-segments will have a 5:1 SNR, resulting in a reported centroid value of 0xFF¹, as Figure 6-11 shows.

The minimum value of slider-segment width for certain overlay thickness values for an acrylic overlay are provided in Table 6-3. For thickness values of acrylic overlays, which are not specified in Table 6-3, Figure 6-12 may be used to estimate the minimum slider-segment width.

¹ The CapSense Component in PSoC Creator reports a centroid of 0xFF when there is no finger detected on the slider, or when none of the slider segments reports a difference count value greater than the Finger Threshold parameter.
If the slider-segment-width + 2 * air-gap is higher than the finger diameter value as required per Equation 6-1, the centroid response will have flat spots; that is, if the finger is moved a little near the middle of any segment, the reported centroid position will remain constant as Figure 6-13 shows. This is because, as Figure 6-14 shows, when the finger is placed in the middle of a slider segment, it will add a valid signal only to that segment even if the finger is moved a little towards adjacent segments.
Design Considerations

Note that if the value of slider-segment-width + 2 * air-gap is higher than the finger diameter, it may be possible to increase and adjust the sensitivity of all slider segments such that even if the finger is placed in the middle of a slider segment, adjacent sensors report a difference count value equal to the noise threshold value (as required per Figure 6-7, Ideal Slider Segment Signals and Centroid Response); however, this will result in the hover effect – the slider may report a centroid position even if the finger is hovering above the slider and not touching the slider.

6.3.4.2 Dummy Segments at the Ends of a Slider

In a CapSense design, when one segment is scanned, adjacent segments are connected to either ground or to the driven-shield signal based on the option specified in the “Inactive sensor connection” parameter in the CapSense CSD Component. For a linear centroid response, the slider requires all the segments to have the same sensitivity, that is, the increase in the raw count (signal) when a finger is placed on the slider segment should be the same for all segments. To maintain a uniform signal level from all slider segments, it is recommended that you physically connect the two segments at both ends of a slider to either ground or driven shield signal. The connection to ground or to the driven-shield signal depends on the value specified in the “Inactive sensor connection” parameter. Therefore, if your application requires an ‘n’ segment slider, it is recommended that you create n + 2 physical segments, as Figure 6-6 shows.

If it is not possible to have two segments at both ends of a slider due to space constraints, you can implement these segments in the top hatch fill, as Figure 6-15 shows. Also, if the total available space is still constrained, the width of these segments may be kept lesser than the width of segments SLD0 through SLDn-1, or these dummy segments may even be removed.

If the two segments at the both ends of a slider are connected to the top hatch fill, you should connect the top hatch fill to the signal specified in the “Inactive sensor connection” parameter. If liquid tolerance is required for the slider, the hatch fill around the slider, the last two segments, and the inactive slider segments should be connected to the driven-shield signal. See the Effect of Liquid Droplets and Liquid Stream on a CapSense Sensor section for more details.

Figure 6-15. Linear Slider Pattern when First and Last Segments are Connected to Top Hatch Fill

6.3.4.3 Deciding Slider Dimensions

Slider dimensions for a given design can be chosen based on following considerations:

a) Decide the required length of the slider (L) based on application requirements. This is same as the “effective slider length” as Figure 6-6 shows.

b) Decide the height of the segment based on the available space on the board. Use the maximum allowed segment height (15 mm) if the board space permits; if not, use a lesser height but ensure that the height is greater than the minimum specified in Table 6-3.

c) The slider-segment width and the air gap between slider segments should be as recommended in Table 6-3. The recommended slider-segment width and air gap for an average finger diameter of 9 mm is 8 mm and 0.5 mm respectively.

d) For a given slider length L, calculate the number of segments required by using the following formula:

\[
Number\ of\ segments = \frac{slider\ length}{slider\ segment\ width + air\ gap} + 1
\]

Note that a minimum of two slider segments are required to implement a slider.

If the available number of CapSense pins is slightly less than the number of segments calculated for a certain application, you may increase the segment width to achieve the required slider length with the available number of pins. For example, a 10.2-cm slider requires 13 segments. However, if only 10 pins are available, the segment
width may be increased to 10.6 cm. This will either result in a nonlinear response as Figure 6-13 shows, or a hover effect; however, this layout may be used if the end application does not need a high linearity.

Note that the PCB length is higher than the required slider length as Figure 6-6 shows. PCB length can be related to the slider length as follows:

Equation 6-3 Relationship Between Minimum PCB Length and Slider Length

\[
PCB \text{ length} = \text{Slider Length} + 3 \times \text{slider segment width} + 2 \times \text{air gap}
\]

If the available PCB area is less than that required per this equation, you can remove the dummy segments. In this case, the minimum PCB length required will be as follows:

\[
PCB \text{ length} = \text{Slider Length} + \text{slider segment width}
\]

6.3.4.4 Routing Slider Segment Trace

A slider has many segments, each of which is connected separately to the CapSense input pin of the device. Each segment is separately scanned and the centroid algorithm is applied finally on the signal values of all the segments to calculate the centroid position. The SmartSense algorithm implements a specific tuning method for sliders to avoid nonlinearity in the centroid that could occur due to the difference of \( C_p \) in the segments. However, the following layout conditions need to be met in order for the slider to work:

1. \( C_p \) of any segment should always be within the supported range of 5-45 pF.
2. \( C_p \) of other segments should be greater than 75% of the \( C_p \) of the segment with the maximum value in that slider. For example, in a slider, if 30 pF is the maximum \( C_p \) of a segment, the \( C_p \) of other segments should be greater than 22.5 pF.

Implement the following layout design rules to meet this condition:

- Design the shape of all segments to be as uniform as possible.
- Ensure that the length and the width of the traces connecting the segments to the device are same for all the segments.
- Maintain the same air gap between the sensors or traces to ground plane or hatch fill.

6.3.4.5 Slider Design with LEDs

In some applications, it might be required to display the finger position by driving LEDs. You can either place the LEDs just above the slider segments or drill a hole in the middle of a slider segment for LED backlighting, as Figure 6-16 shows. When a hole is drilled for placing an LED, the effective area of the slider segment reduces. To achieve an SNR > 5:1, you need to have a slider segment with a width larger than the LED hole size. Refer to Table 6-3 for the minimum slider width required to achieve an SNR > 5:1 for a given overlay thickness. Follow the guidelines provided in the Crosstalk Solutions section for routing the LED traces.

Figure 6-16. Slider Design with LED Backlighting
6.3.5 Sensor and Device Placement

Follow these guidelines while placing the sensor and the device in your PCB design:

- Minimize the trace length from the device pins to the sensor pad.
- Mount series resistors within 10 mm of the device pins to reduce RF interference and provide ESD protection. See Series Resistors on CapSense Pins for details.
- Mount the device and the other components on the bottom layer of the PCB.
- Isolate switching signals, such as PWM, I²C communication lines, and LEDs, from the sensor and sensor traces. You should place them at least 4 mm apart and fill a hatched ground between the CapSense traces and the switching signals to avoid crosstalk.
- DC loads such as LEDs and I²C pins should be physically separate from the CapSense pins by a full port wherever possible. For example, if there are LED pins in port P1, it is recommended to avoid having a CapSense pin in the same port. Also, it is recommended to limit the total source or sink current through GPIOs to less than 40 mA while the CapSense block is scanning the sensor. Sinking a current greater than 40 mA during the CapSense sensor scanning might result in excessive noise in the sensor raw count.
- Avoid connectors between the sensor and the device pins because connectors increase C_p and noise pickup.

6.3.6 Trace Length and Width

Use short and narrow PCB traces to minimize the parasitic capacitance of the sensor. The maximum recommended trace length is 12 inches (300 mm) for a standard PCB and 2 inches (50 mm) for flex circuits. The maximum recommended trace width is 7 mil (0.18 mm). You should surround the CapSense traces with a hatched ground or hatched shield with trace-to-hatch clearance of 10 mil to 20 mil (0.25 mm to 0.51 mm).

6.3.7 Trace Routing

You should route the sensor traces on the bottom layer of the PCB, so that the finger does not interact with the traces. Do not route traces directly under any sensor pad unless the trace is connected to that sensor.

Do not run capacitive sensing traces closer than 0.25 mm to switching signals or communication lines. Increasing the distance between the sensing traces and other signals increases the noise immunity. If it is necessary to cross communication lines with sensor pins, make sure that the intersection is at right angles, as Figure 6-17 shows.

Figure 6-17. Routing of Sensor and Communication Lines

If, due to spacing constraints, sensor traces run in parallel with high-speed traces such as I²C communication lines or BLE antenna traces, it is recommended to place a ground trace between the sensor trace and the high-speed trace as shown in Figure 6-18. This guideline also applies to the cross talk caused by CapSense sensor trace with precision analog trace such
as a trace from temperature sensor to the PSoC device. The thickness of the ground trace can be 7 mils and the spacing from sensor trace to ground trace should be equal to minimum of 10 mils to reduce the C$_{p}$ of the CapSense sensor.

Figure 6-18. Reducing Cross Talk between High Speed Switching Trace and CapSense Trace

If a ground trace cannot be placed in between the switching trace and the CapSense trace, the 3W rule can be followed to reduce the cross talk between the traces. The 3W rule states that “To reduce cross talk from adjacent traces, a minimum spacing of two trace widths should be maintained from edge to edge” as shown in Figure 6-19.

Figure 6-19. 3W Trace Spacing to Minimize Cross Talk

6.3.8 Crosstalk Solutions

A common backlighting technique for panels is an LED mounted under the sensor pad so that it is visible through a hole in the middle of the sensor pad. When the LED is switched ON or OFF, voltage transitions on the LED trace can create crosstalk in the capacitive sensor input, creating noisy sensor data. To prevent this crosstalk, isolate CapSense and the LED traces from one another as section 6.3.7 explains.

You can also reduce crosstalk by removing the rapid transitions in the LED drive voltage, by using a filter as Figure 6-20 shows. Design the filter based on the required LED response speed.
6.3.9 Vias

Use the minimum number of vias possible to route CapSense signals, to minimize parasitic capacitance. Place the vias on the edge of the sensor pad to reduce trace length, as Figure 6-21 shows.

**Figure 6-21. Via Placement on the Sensor Pad**

- Via at the Center of the Sensor (Long Trace)
- Via Near the Edge of the Sensor (Short Trace)
6.3.10 Ground Plane

When designing the ground plane, follow these guidelines:

- Ground surrounding the sensors should be in a hatch pattern. If you are using ground planes in both top and bottom layers of the PCB, you should use a 25 percent hatching on the top layer (7-mil line, 45-mil spacing), and 17 percent on the bottom layer (7-mil line, 70-mil spacing). Use the same hatching fill on both the top and bottom layers if you are using driven shield instead of ground.
- For the other parts of the board not related to CapSense, solid ground should be present as much as possible.
- The ground planes on different layers should be stitched together as much as possible, depending on the PCB manufacturing costs. Higher amount of stitching results in lower ground inductance, and brings the chip ground closer to the supply ground. This is important especially when there is high current sinking through the ground, such as when the radio is operational.
- Every ground plane used for CapSense should be star-connected to a central point, and this central point should be the sole return path to the supply ground. Specifically:
  - The hatch ground for all sensors must terminate at the central point
  - The ground plane for \( C_{MOD} \) must terminate at the central point
  - The ground plane for \( C_{SH\_TANK} \) must terminate at the central point

Figure 6-22 explains the star connection. The central point for different families is mentioned in Table 6-4.

![Figure 6-22. Star Connection for Ground](image)

Table 6-4. Central Point for Star Connection

<table>
<thead>
<tr>
<th>Family</th>
<th>Central point</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSoC 4000</td>
<td>VSS pin</td>
</tr>
<tr>
<td>PSoC 4100/ 4100 M</td>
<td>VSS pin</td>
</tr>
<tr>
<td>PSoC 4200/ 4200 M</td>
<td>VSS pin</td>
</tr>
<tr>
<td>PSoC 4100-BL</td>
<td>E-pad</td>
</tr>
<tr>
<td>PSoC 4200-BL</td>
<td>E-pad</td>
</tr>
<tr>
<td>PRoC BLE</td>
<td>E-pad</td>
</tr>
</tbody>
</table>

- All the ground planes for CapSense should have an inductance of less than 0.2 nH from the central point. To achieve this, place the \( C_{MOD} \) and \( C_{SH\_TANK} \) capacitor pads close to the chip, and keep their ground planes thick enough.
6.3.10.1 Using packages without E-pad

When not using the E-pad, the VSS pin should be the central point and the sole return path to the supply ground.

High-level layout diagrams of the top and bottom layers of a board when using a chip without the E-pad are shown in Figure 6-23 and Figure 6-24.

Figure 6-23. PCB Top Layer Layout Using a Chip Without E-pad

Figure 6-24. PCB Bottom Layer Layout Using a Chip Without E-pad
6.3.10.2 Using packages with E-pad

If you are using packages with E-pad, the following guidelines must be followed:

- The E-pad must be the central point and the sole return path to the supply ground.
- The E-pad must have vias underneath to connect to the next layers for additional grounding. Usually unfilled vias are used in a design for cost reasons, but silver-epoxy filled vias are recommended for the best performance as they result in the lowest inductance in the ground path.

6.3.10.3 Using PSoC 4 BLE or PRoC BLE chips

In the case of PSoC 4 BLE or PRoC BLE chips in QFN package (with E-pad):

- The general guidelines of Ground plane (discussed above) apply.
- The E-pad usage guidelines of Section 6.3.10.2 apply.
- The VSSA pin should be connected to the E-pad below the chip itself.
- The vias underneath the E-pad are recommended to be 5 x 5 vias of 10-mil hole size.

High-level layout diagrams of the top and bottom layers of a board when using PSoC 4 BLE or PRoC BLE chips are shown in Figure 6-25 and Figure 6-26.

Figure 6-25. PCB Top Layer Layout with PSoC 4 BLE / PRoC BLE (with E-pad)
6.3.11 Power Supply Layout Recommendations

CapSense is a high-sensitivity analog system. Therefore, a poor PCB layout introduces noise in high-sensitivity sensor configurations such as proximity sensors and buttons with thick overlays (>1 mm). To achieve low noise in a high-sensitivity CapSense design, the PCB layout should have decoupling capacitors on the power lines, as per Table 6-5.

<table>
<thead>
<tr>
<th>Power Line</th>
<th>Decoupling Capacitors</th>
<th>Corresponding Ground Terminal</th>
<th>Applicable Device Family</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDD</td>
<td>0.1 µF and 1 µF</td>
<td>VSS</td>
<td>PSoC 4000</td>
</tr>
<tr>
<td>VDDIO</td>
<td>0.1 µF</td>
<td>VSS</td>
<td>PSoC 4000</td>
</tr>
<tr>
<td>VDDD</td>
<td>0.1 µF and 1 µF</td>
<td>VSS</td>
<td>PSoC 4100, PSoC 4200</td>
</tr>
<tr>
<td></td>
<td>0.1 µF and 1 µF</td>
<td>VSSD</td>
<td>PSoC 4100-BL, PSoC 4200-BL, PRoC BLE</td>
</tr>
<tr>
<td>VDDA</td>
<td>0.1 µF and 1 µF</td>
<td>VSSA</td>
<td>PSoC 4100, PSoC 4200, PSoC 4100-BL, PSoC 4200-BL, PRoC BLE</td>
</tr>
<tr>
<td>VDDR</td>
<td>0.1 µF and 1 µF</td>
<td>VSSD</td>
<td>PSoC 4100-BL, PSoC 4200-BL, PRoC BLE</td>
</tr>
<tr>
<td>VCCD</td>
<td>Refer device datasheet</td>
<td>VSS (PSoC 4000) or VSSD (all others)</td>
<td>All device families</td>
</tr>
</tbody>
</table>
The decoupling capacitors and CMOD capacitor must be placed as close to the chip as possible to keep ground impedance and supply trace length as low as possible.

For further details on bypass capacitors, refer to the Power section in the PSoC 4 Datasheet, PSoC 4 BLE Datasheet, or PRoC BLE Datasheet.

6.3.12 Layout Guidelines for Liquid Tolerance

As explained in the Liquid Tolerance section, by implementing a shield electrode and a guard sensor, a liquid-tolerant CapSense system can be implemented. This section shows how to implement a shield electrode and a guard sensor.

The area of the shield electrode depends on the size of the liquid droplet and the area available on the board for implementing the shield electrode. The shield electrode should surround the sensor pads and traces, and spread no further than 1 cm from these features. Spreading the shield electrode beyond 1 cm has negligible effect on system performance.

Also, having a large shield electrode might increase radiated emissions. If the board area is very large, the area outside the 1-cm shield electrode should be left empty, as Figure 6-27 shows. For improved liquid-tolerance performance, there should not be any hatch fill or a trace connected to ground in the top and bottom layers of the PCB.

When there is a grounded hatch fill or a trace then, when a liquid droplet falls on the touch surface, it might cause sensor false triggers. Even if there is a shield electrode in between the sensor and ground, the effect of the shield electrode will be totally masked out and sensors might false trigger.

In some applications, there might not be sufficient area available on the PCB for shield electrode implementation. In such cases, the shield electrode can spread less than 1 cm; the minimum area for shield electrode can be the area remaining on the board after implementing the sensor.

Figure 6-27. Shield Electrode Placement when Sensor Trace is Routed in Top and Bottom Layer

Follow the guidelines below to implement the shield electrode in 2-layer and 4-layer PCBs:

Two-Layer PCB:
- Top layer: Hatch fill with 7-mil trace and 45-mil grid (25 percent fill). Hatch fill should be connected to the driven-shield signal.
- Bottom layer: Hatch fill with 7-mil trace and 70-mil grid (17 percent fill). Hatch fill should be connected to the driven-shield signal.
Four (or More)-Layer PCB:

- Top layer: Hatch fill with 7-mil trace and 45-mil grid (25 percent fill). Hatch fill should be connected to the driven-shield signal.
- Layer-2: Hatch fill with 7-mil trace and 70-mil grid (17 percent fill). Hatch fill should be connected to the driven-shield signal.
- Layer-3: VDD Plane
- Bottom layer: Hatch fill with 7-mil trace and 70-mil grid (17 percent fill). Hatch fill should be connected to ground.

The recommended air gap between the sensor and the shield electrode is 1 mm.

6.3.12.1 Guard Sensor

As explained in the Guard Sensor section, the guard sensor is a copper trace that surrounds all of the sensors, as Figure 6-28 shows.

The guard sensor should be triggered only when there is a liquid stream on the touch surface. Make sure that the shield electrode pattern surrounds the guard sensor to prevent it from turning on due to liquid droplets. The guard sensor should be placed such that it meets the following conditions:

- It should be the first sensor to turn on when there is a liquid stream on the touch surface. To accomplish this, the guard sensor is usually placed such that it surrounds all sensors.
- It should not be accidentally touched while pressing a button or slider sensor. Otherwise, the button sensors and slider sensor scanning will be disabled and the CapSense system will become nonoperational until the guard sensor is turned off. To ensure the guard sensor is not accidentally triggered, place the guard sensor at a distance greater than 1 cm from the sensors.

Follow the guidelines below for implementing the guard sensor:

- The guard sensor should be in the shape of a rectangle with curved edges and should surround all the sensors.
- The recommended thickness for a guard sensor is 2 mm.
- The recommended clearance between the guard sensor and the shield electrode is 1 mm.

If there is no space on the PCB for implementing a guard sensor, the guard sensor functionality can be implemented in the firmware. For example, you can use the ON/OFF status of different sensors to detect a liquid stream. The following conditions can be used to detect a liquid stream on the touch surface:
- When there is a liquid stream, more than one button sensor will be active at a time. If your design does not require multi-touch sensing, you can detect this and reject the sensor status of all the button sensors to prevent false triggering.

- In a slider, if the slider segments which are turned ON are not adjacent segments, you can reset the slider segments status or reject the slider centroid value that is calculated.

6.3.12.2 Liquid Tolerance with Ground Ring

In some applications, it is required to have a ground ring (solid trace or a hatch fill) around the periphery of the board for improved ESD and EMI/EMC performance, as shown in Figure 6-29. Having a ground ring around the board might result in sensor false triggers when liquid droplets fall in between the sensor and the ground sensor. Therefore it is recommended not to have any ground in the top layer. If the design must have a ground ring in the top layer, use a ground ring with the minimum thickness (8 mils).

Figure 6-29. CapSense Design with a Ground Ring for Improved ESD and EMI/EMC Performance
6.3.13 Schematic Rule Checklist

You can use the checklist provided to verify your CapSense schematic.

Table 6-6. Schematic Rule Checklist

<table>
<thead>
<tr>
<th>No.</th>
<th>Category</th>
<th>Recommendations / Remarks</th>
</tr>
</thead>
</table>
| 1   | $C_{MOD}$                             | 2.2 nF  
Refer to Table 6-7 for pin selection.                                                 |
| 2   | $C_{SH\_TANK}$                        | 10 nF  
Refer to Table 6-7 for pin selection.                                                  |
| 3   | Series resistance on input lines      | 560 $\Omega$  
See Series Resistors on CapSense Pins for details.                                        |
| 4   | Sensor pin selection                  | If possible, avoid pins that are close to the GPIOs carrying switching/communication signals. Physically separate DC loads such as LEDs and I$^2$C pins from the CapSense pins by a full port wherever possible. See Sensor and Device Placement for details. |
| 5   | GPIO Source/Sink Current              | Ensure that the total sink current through GPIOs is not greater than 40 mA when the CapSense block is scanning the sensors. |

6.3.13.1 $C_{MOD}$ and $C_{SH\_TANK}$ Pin Selection

Table 6-7 lists the recommended pins for $C_{MOD}$ and $C_{SH\_TANK}$. Note that pins other than recommended pins may be used for $C_{MOD}$ with the following constraints:

1. The recommended $C_{MOD}$ pin is always used because it is directly connected to the Sigma Delta Converter. If a pin other than the recommended pin is selected for $C_{MOD}$, the recommended $C_{MOD}$ pin would not be available for any other function. For example, if you try routing $C_{MOD}$ to P2[0] in PSoC Creator for a PSoC 4200 device, it uses both P2[0] and P4[2].

2. If a pin other than the recommended pin is used for $C_{MOD}$, the shield cannot be used. This is because the $C_{MOD}$ connection to a pin other than the recommended pin requires the use of AMUXBUS B. The shield signal requires the same AMUXBUS B as mentioned in the CapSense CSD Shielding section. If you want to use the shield, $C_{MOD}$ should be placed on a recommended pin only.

Table 6-7. Recommended Pins for $C_{MOD}$ and $C_{SH\_TANK}$

<table>
<thead>
<tr>
<th>No.</th>
<th>Type</th>
<th>PSoC 4000</th>
<th>PSoC 4100, PSoC 4200</th>
<th>PSoC 4200M$^1$</th>
<th>PSoC 4100-BL, PSoC 4200-BL</th>
<th>CYBL10X6X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P5[0] – CSD1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P5[1] – CSD1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$^1$ See CapSense in PSoC 4 M-Series section for details on CSD0 and CSD1 block.
### 6.3.14 Layout Rule Checklist

You can use the checklist provided in Table 6-8 to help verify your layout design.

Table 6-8. Layout Rule Checklist

<table>
<thead>
<tr>
<th>No.</th>
<th>Category</th>
<th>Minimum Value</th>
<th>Maximum Value</th>
<th>Recommendations / Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Button</td>
<td>Shape</td>
<td>N/A</td>
<td>Circle or rectangular with curved edges</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Size</td>
<td>5 mm</td>
<td>15 mm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clearance to ground hatch</td>
<td>0.5 mm</td>
<td>2 mm</td>
</tr>
<tr>
<td>2</td>
<td>Slider</td>
<td>Width of segment</td>
<td>1.5 mm</td>
<td>8 mm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clearance between segments</td>
<td>0.5 mm</td>
<td>2 mm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Height of segment</td>
<td>7 mm</td>
<td>15 mm</td>
</tr>
<tr>
<td>3</td>
<td>Overlay</td>
<td>Type</td>
<td>N/A</td>
<td>Material with high relative permittivity (except conductors) Remove any air gap between sensor board and overlay / front panel of the casing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Thickness for Buttons</td>
<td>N/A</td>
<td>5 mm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Thickness for Sliders</td>
<td>N/A</td>
<td>5 mm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Thickness for Touchpads</td>
<td>N/A</td>
<td>0.5 mm</td>
</tr>
<tr>
<td>4</td>
<td>Sensor Traces</td>
<td>Width</td>
<td>N/A</td>
<td>7 mil</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Length</td>
<td>N/A</td>
<td>300 mm for a standard (FR4) PCB 50 mm for flex PCB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clearance to ground and other traces</td>
<td>0.25 mm</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Routing</td>
<td>N/A</td>
<td>Route on the opposite side of the sensor layer. Isolate from other traces. If any non CapSense trace crosses CapSense trace, ensure that intersection is orthogonal. Do not use sharp turns.</td>
</tr>
<tr>
<td>5</td>
<td>Via</td>
<td>Number of vias</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hole size</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
## Design Considerations

<table>
<thead>
<tr>
<th>No.</th>
<th>Category</th>
<th>Minimum Value</th>
<th>Maximum Value</th>
<th>Recommendations / Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Ground</td>
<td>N/A</td>
<td>N/A</td>
<td>Use hatch ground to reduce parasitic capacitance. Typical hatching: 25% on the top layer (7-mil line, 45-mil spacing) 17% on the bottom layer (7-mil line, 70-mil spacing)</td>
</tr>
<tr>
<td>7</td>
<td>Series resistor placement</td>
<td>N/A</td>
<td>N/A</td>
<td>Place resistor within 10 mm of PSoC pin. Refer to Figure 6-30 for an example placement of series resistance on board.</td>
</tr>
<tr>
<td>8</td>
<td>Shield electrode</td>
<td>Spread</td>
<td>1 cm</td>
<td>If you have PCB space, use 1-cm spread.</td>
</tr>
<tr>
<td>9</td>
<td>Guard sensor (for water tolerance)</td>
<td>Shape</td>
<td>N/A</td>
<td>Rectangle with curved edges</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Thickness</td>
<td>N/A</td>
<td>Recommended thickness of guard trace is 2 mm and distance of guard trace to shield electrode is 1 mm.</td>
</tr>
<tr>
<td>10</td>
<td>C\textsubscript{MOD}</td>
<td>N/A</td>
<td>N/A</td>
<td>Place close to the PSoC pin. Refer to Figure 6-30 for an example placement of C\textsubscript{MOD} on printed circuit board.</td>
</tr>
<tr>
<td>11</td>
<td>C\textsubscript{SH_TANK}</td>
<td>N/A</td>
<td>N/A</td>
<td>Place close to the PSoC pin. Refer to Figure 6-30 for an example placement of C\textsubscript{SH_TANK} on board.</td>
</tr>
</tbody>
</table>
Figure 6-30. Example Placement for CMOD, C\textsubscript{SH,TANK} and Series Resistance on Input Lines in PSoC 4M Device
6.4 ESD Protection

The nonconductive overlay material used in CapSense provides inherent protection against ESD. Table 6-9 lists the thickness of various overlay materials, required to protect the CapSense sensors from a 12-kV discharge (according to the IEC 61000-4-2 specification).

<table>
<thead>
<tr>
<th>Material</th>
<th>Breakdown Voltage (V/mm)</th>
<th>Minimum Overlay Thickness for Protection Against 12 kV ESD (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Air</td>
<td>1200 – 2800</td>
<td>10</td>
</tr>
<tr>
<td>Wood – dry</td>
<td>3900</td>
<td>3</td>
</tr>
<tr>
<td>Glass – common</td>
<td>7900</td>
<td>1.5</td>
</tr>
<tr>
<td>Glass – Borosilicate (Pyrex®)</td>
<td>13,000</td>
<td>0.9</td>
</tr>
<tr>
<td>PMMA Plastic (Plexiglas®)</td>
<td>13,000</td>
<td>0.9</td>
</tr>
<tr>
<td>ABS</td>
<td>16,000</td>
<td>0.8</td>
</tr>
<tr>
<td>Polycarbonate (Lexan®)</td>
<td>16,000</td>
<td>0.8</td>
</tr>
<tr>
<td>Formica</td>
<td>18,000</td>
<td>0.7</td>
</tr>
<tr>
<td>FR-4</td>
<td>28,000</td>
<td>0.4</td>
</tr>
<tr>
<td>PET Film (Mylar®)</td>
<td>280,000</td>
<td>0.04</td>
</tr>
<tr>
<td>Polymide film (Kapton®)</td>
<td>290,000</td>
<td>0.04</td>
</tr>
</tbody>
</table>

If the overlay material does not provide sufficient protection (for example, ESD from other directions), you can apply other ESD countermeasures, in the following order: Prevent, Redirect, ESD protection devices.

6.4.1 Preventing ESD Discharge

Preventing the ESD discharge from reaching the PSoC is the best countermeasure you can take. Make sure that all paths to PSoC have a breakdown voltage greater than the maximum ESD voltage possible at the surface of the equipment. You should also maintain an appropriate distance between the PSoC and possible ESD sources. In the example illustrated in Figure 6-31, if L1 and L2 are greater than 10 mm, the system can withstand a 12-kV ESD.

If it is not possible to maintain adequate distance, place a protective layer of nonconductive material with a high breakdown voltage between the possible ESD source and PSoC. One layer of 5-mil thick Kapton® tape can withstand 18 kV. See Table 6-9 for other material dielectric strengths.
6.4.2 Redirect
If your product is densely packed, preventing the discharge event may not be possible. In such cases, you can protect the PSoC from ESD by redirecting the ESD. A standard practice is to place a ground ring on the perimeter of the circuit board, as Figure 6-32 shows. The ground ring should connect to the chassis ground. Using a hatched ground plane around the button or slider sensor can also redirect the ESD event away from the sensor and PSoC.

Figure 6-32. Ground Ring

6.4.3 ESD Protection Devices
You can use ESD protection devices on vulnerable traces. Select ESD protection devices with a low input capacitance to avoid reduction in CapSense sensitivity. Table 6-10 lists the recommended ESD protection devices.

Table 6-10. ESD Protection Devices

<table>
<thead>
<tr>
<th>ESD Protection Device</th>
<th>Input Capacitance</th>
<th>Leakage Current</th>
<th>Contact Maximum ESD Limit</th>
<th>Air Discharge Maximum ESD Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturer</td>
<td>Part Number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Littelfuse</td>
<td>SP723</td>
<td>5 pF</td>
<td>2 nA</td>
<td>8 kV</td>
</tr>
<tr>
<td>Vishay</td>
<td>VBUS05L1-DD1</td>
<td>0.3 pF</td>
<td>0.1 µA</td>
<td>±15 kV</td>
</tr>
<tr>
<td>NXP</td>
<td>NUP1301</td>
<td>0.75 pF</td>
<td>30 nA</td>
<td>8 kV</td>
</tr>
</tbody>
</table>
6.5 Electromagnetic Compatibility (EMC) Considerations

EMC is related to the generation, transmission, and reception of electromagnetic energy that can affect the working of an electronic system. Electronic devices are required to comply with specific limits for emitted energy and susceptibility to external events. Several regulatory bodies worldwide set regional regulations to help ensure that electronic devices do not interfere with each other.

CMOS analog and digital circuits have very high input impedance. As a result, they are sensitive to external electric fields. Therefore, you should take adequate precautions to ensure their proper operation in the presence of radiated and conducted noise.

Computing devices are regulated in the US by the FCC under Part 15, Sub-Part B for unintentional radiators. The standards for Europe and the rest of the world are adapted from CENELEC. These are covered under CISPR standards (dual labeled as ENxxxx standards) for emissions, and under IEC standards (also dual labeled as ENxxxx standards) for immunity and safety concerns.

The general emission specification is EN55022 for computing devices. This standard covers both radiated and conducted emissions. Medical devices in the US are not regulated by the FCC, but rather are regulated by FDA rules, which include requirements of EN55011, the European norm for medical devices. Devices that include motor controls are covered under EN55014 and lighting devices are covered under EN50015.

These specifications have essentially similar performance limitations for radiated and conducted emissions. Radiated and conducted immunity (susceptibility) performance requirements are specified by several sections of EN61000-4. Line voltage transients, electrostatic discharge (ESD) and some safety issues are also covered in this standard.

6.5.1 Radiated Interference and Emissions

While PSoC 4 and PRoC BLE offer a robust CapSense performance, radiated electrical energy can influence system measurements and potentially influence the operation of the CapSense processor core. Interference enters the CapSense device at the PCB level through sensor traces and through other digital and analog inputs. CapSense devices can also contribute to electromagnetic compatibility (EMC) issues in the form of radiated emissions.

Use the following techniques to minimize the radiated interference and emissions.

6.5.1.1 Hardware Considerations

6.5.1.1.1 Ground Plane

In general, proper ground plane on the PCB reduces both RF emissions and interference. However, solid grounds near CapSense sensors or traces connecting these sensors to PSoC pins increase the parasitic capacitance of the sensors. It is thus recommended to use hatched ground planes surrounding the sensor and on the bottom layer of the PCBs, below the sensors, as explained in the Ground Plane section in PCB Layout Guidelines. Solid ground may be used below the device and other circuitry on the PCB which is farther from CapSense sensors and traces. A solid ground flood is not recommended within 1 cm of CapSense sensors or traces.

6.5.1.1.2 Series Resistors on CapSense Pins

Every CapSense controller pin has some parasitic capacitance, CP, associated with it. As Figure 6-33 shows, adding an external resistor forms a low-pass RC filter that attenuates the RF noise amplitude coupled to the pin. This resistance also forms a low-pass filter with the parasitic capacitance of the CapSense Sensor that significantly reduces the RF emissions.

Series resistors should be placed close the device pins so that the radiated noise picked by the traces gets filtered at the input of the device. Thus, it is recommended to place series resistors within 10 mm of the pins.
For CapSense designs using copper on PCBs, the recommended series resistance for CapSense input lines is 560 $\Omega$. Adding the resistance increases the time constant of the switched-capacitor circuit that converts $C_P$ into an equivalent resistor; see GPIO Cell Capacitance to Current Converter. If the series resistance value is larger than 560 $\Omega$, the slower time constant of the switching circuit suppresses the emissions and interference, but limits the amount of charge that can transfer. This lowers the signal level, which in turn lowers the SNR. Smaller values are better in terms of SNR, but are less effective at blocking RF.

### 6.5.1.1.3 Series Resistors on Digital Communication Lines

Communication lines, such as I$^2$C and SPI, also benefit from series resistance; 330 $\Omega$ is the recommended value for series resistance on communication lines. Communication lines have long traces that act as antennae similar to the CapSense traces. The recommended pull-up resistor values for I$^2$C communication lines is 4.7 k$\Omega$. So, if more than 330 $\Omega$ is placed in series on these lines, the $V_{IL}$ and $V_{IH}$ voltage levels may fall out of specifications. 330 $\Omega$ will not affect I$^2$C operation as the $V_{IL}$ level still remains within the I$^2$C specification limit of 0.3 $V_{DD}$ when PSoC outputs a LOW.

![Figure 6-34. Series Resistors on Communication Lines](image)

### 6.5.1.1.4 Trace Length

Long traces can pick up more noise than short traces. Long traces also add to $C_P$. Minimize the trace length whenever possible.

### 6.5.1.1.5 Current Loop Area

Another important layout consideration is to minimize the return path for currents. This is important as the current flows in loops. Unless there is a proper return path for high-speed signals, the return current will flow through a longer return path forming a larger loop, thus leading to increased emissions and interference.

If you isolate the CapSense ground hatch and the ground fill around the device, the sensor-switching current may take a longer return path, as Figure 6-35 shows. As the CapSense sensors are switched at a high frequency, the return current may cause serious EMC issues. Therefore, you should use a single ground hatch, as Figure 6-36 shows.

![Figure 6-35. Improper Current Loop Layout](image)
6.5.1.1.6  RF Source Location
If your system has a circuit that generates RF noise, such as a switched-mode power supply (SMPS) or an inverter, you should place these circuits away from the CapSense interface. You should also shield such circuits to reduce the emitted RF. Figure 6-37 shows an example of separating the RF noise source from the CapSense interface.

6.5.1.2  Firmware Considerations
The following parameters affect Radiated Emissions (RE) in a CapSense system:

- Device operating voltage
- Device operation frequency
- Sensor switching frequency
- Shield signal
- Sensor scan time
- Analog switch drive source
- Inactive sensor termination

The following sections explain the effect of each parameter.
6.5.1.2.1  Device Operating Voltage

The emission is directly proportional to the voltage levels at which switching happens. Reducing the operating voltage helps to reduce the emissions as the amplitude of the switching signal at any output pin directly depends on the operating voltage of the device.

PSoC allows you to operate at lower operating voltages, thereby reducing the emissions. Figure 6-38 and Figure 6-39 show the impact of operating voltage on radiated emissions. Because IMO = 24 MHz, there is a spike at 24 MHz and the other spikes are caused by different hardware and firmware operations of the device.

Figure 6-38. Effect of VDD on Radiated Emissions (150 kHz – 30 MHz)

Note: Frequency axis is in log scale
6.5.1.2.2 Device Operating Frequency
Reducing the system clock frequency (IMO frequency) reduces radiated emissions. However, reducing the IMO frequency may not be feasible in all applications since the IMO frequency impacts the CPU clock and all other system timings. Chose a suitable IMO frequency based on your application.

6.5.1.2.3 Sensor-Switching Frequency
Reducing the sensor-switching frequency (see Switching Clock Selection) also helps to reduce radiated emissions. See Figure 6-40 and Figure 6-41. Because IMO = 24 MHz, there is a spike at 24 MHz and the other spikes are caused by different hardware and firmware operations of the device.

Figure 6-40. Effect of Sensor-Switching Frequency on Radiated Emissions (150 kHz – 30 MHz)

Figure 6-41. Effect of Sensor-Switching Frequency on Radiated Emissions (30 MHz – 1 GHz)

Note: Frequency axis is in log scale
6.5.1.2.4 Shield Signal

Enabling the shield signal (see Driven-Shield Signal and Shield Electrode and “Shield” explanation in Advanced Settings) on the hatch pattern increases the radiated emissions. Enable the driven-shield signal only for liquid-tolerant, proximity-sensing, or high-parasitic-capacitance designs. Also, if the shield has to be used, ensure that the shield electrode area is limited to a width of 1 cm from the sensors, as Figure 6-27 shows.

Figure 6-42 and Figure 6-43 show the impact of enabling the driven-shield signal on the hatch pattern surrounding the sensors on radiated emissions. Note that in these figures the hatch pattern is grounded when the driven-shield signal is disabled. Because IMO = 24 MHz, there is a spike at 24 MHz and the other spikes are caused by different hardware and firmware operations of the device.

Figure 6-42. Effect of Shield Electrode on Radiated Emissions (150 kHz – 30 MHz)

Note: Frequency axis is in log scale
6.5.1.2.5 Sensor Scan Time
Reducing the sensor scan time reduces the average radiated emissions. The sensor-scan time depends on the scan resolution and modulator clock divider (See Figure 5-4 and Figure 5-7). Increasing the scan resolution or modulator clock divider increases the scan time. Figure 6-44 and Figure 6-45 show the impact of sensor scan time on radiated emissions. Note that, here, the sensor-scan time was varied by changing the scan resolution. Because IMO = 24 MHz, there is a spike at 24 MHz and the other spikes are caused by different hardware and firmware operations of the device.

Table 6-11. Sensor Scan Time

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Total Scan time for 5 Buttons</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.426 ms</td>
</tr>
<tr>
<td>Modulation Clock Divider</td>
<td>2</td>
</tr>
<tr>
<td>Scan Resolution</td>
<td>10 bits</td>
</tr>
<tr>
<td>Individual sensor scan time</td>
<td>0.085 ms</td>
</tr>
</tbody>
</table>

Figure 6-44. Effect of Scan Time on Radiated Emissions (150 kHz – 30 MHz)

Figure 6-45. Effect of Scan Time on Radiated Emissions (30 MHz – 1 GHz)

Note: Frequency axis is in log scale
6.5.1.2.6 Analog Switch Drive Source

Using PRS instead of direct clock drive as analog switch drive source spreads the radiated spectrum and hence reduces the average radiated emissions. See Figure 6-46 and Figure 6-47. Because \( \text{IMO} = 24 \text{ MHz} \), there is a spike at 24 MHz and the other spikes are caused by different hardware and firmware operations of the device.

Figure 6-46. Effect of Analog Switch Drive Source on Radiated Emissions (150 kHz – 30 MHz)

Note: Frequency axis is in log scale
6.5.1.2.7 Inactive Sensor Termination
Connecting inactive sensors to ground reduces the radiated emission by a greater degree than connecting them to the shield. Figure 6-48 and Figure 6-49 show the impact of different inactive sensor terminations on radiated emission. Because IMO = 24 MHz, there is a spike at 24 MHz and the other spikes are caused by different hardware and firmware operations of the device.

Figure 6-48. Effect of Inactive Sensor Termination on Radiated Emissions (150 kHz – 30 MHz)

Figure 6-49. Effect of Inactive Sensor Termination on Radiated Emissions (30 MHz – 1 GHz)

Note: Frequency axis is in log scale

6.5.2 Conducted RF Noise
The noise current that enters the CapSense system through the power and communication lines is called conducted noise. You can use the following techniques to reduce the conducted RF noise.

- Use decoupling capacitors on the power supply pins to reduce the conducted noise from the power supply. See section 6.3.11 and the PSoC 4 device datasheet for details.
- Provide GND and VDD planes on the PCB to reduce current loops.
- If the PSoC PCB is connected to the power supply using a cable, minimize the cable length and consider using a shielded cable.

To reduce high-frequency noise, place a ferrite bead around power supply or communication lines.
PSoC 4 and PRoC BLE can perform many additional functions along with CapSense. The wide variety of features offered by this device allows you to integrate various system functions in a single chip, as Figure 7-1 shows. Such applications are known as CapSense Plus applications.

The additional features available in a PSoC 4 or PRoC BLE device include:

- Communication: BLE, I²C, UART, SPI, CAN, and LIN
- Analog functions: ADC, comparators, and opamps
- Digital functions: PWMs, counters, timers, and UDBs
- Segment LCD drive
- Bootloaders
- Different power modes: Active, Sleep, Deep Sleep, Hibernate, and Stop

For more information on PSoc 4, refer to AN79953, Getting Started with PSoc 4, or AN91267, Getting Started with PSoc 4 BLE. For more information on PRoC BLE, refer to AN94020, Getting Started with PRoC BLE.
The flexibility of the PSoC 4 / PRoC BLE and the unique PSoC Creator IDE allow you to quickly make changes to your design, which accelerates time-to-market. Integrating other system functions significantly reduces overall system cost. Table 7-1 shows a list of example applications, where using CapSense Plus can result in significant cost savings.

<table>
<thead>
<tr>
<th>Application</th>
<th>CapSense</th>
<th>OpAmp</th>
<th>ADC</th>
<th>Comp</th>
<th>PWM, Counter, Timer, UDBs</th>
<th>Comm (BLE, I²C, SPI, UART)</th>
<th>LCD drive</th>
<th>GPIOs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heart Rate Monitor (Wrist Band)</td>
<td>User interface: buttons, linear sliders</td>
<td>TIA, Buffer</td>
<td>Heart Rate Measurement, Battery voltage measurement</td>
<td>LED Driving</td>
<td>BLE</td>
<td>Segment LCD</td>
<td>LED indication</td>
<td></td>
</tr>
<tr>
<td>LED Bulb</td>
<td>User interface: buttons, radial sliders</td>
<td>Amplifier</td>
<td>LED current measurement</td>
<td>Short Circuit Protection</td>
<td>LED color control (PrISM*)</td>
<td>BLE</td>
<td></td>
<td>LED indication</td>
</tr>
<tr>
<td>Washing machine</td>
<td>User interface: buttons, radial sliders</td>
<td>Temperature sensor</td>
<td>Water level monitor</td>
<td>Buzzer, FOC** motor control</td>
<td>I²C LCD display, UART network interface</td>
<td>Segment LCD</td>
<td>LED indication</td>
<td></td>
</tr>
<tr>
<td>Water heater</td>
<td>User interface: buttons, linear sliders</td>
<td>Temperature sensor, water flux sensor</td>
<td>Water level monitor</td>
<td>Buzzer</td>
<td>I²C LCD display, UART Network Interface</td>
<td>Segment LCD</td>
<td>LED indication</td>
<td></td>
</tr>
<tr>
<td>IR remote controllers</td>
<td>User interface: buttons, linear and radial sliders, touch pads</td>
<td></td>
<td></td>
<td>Manchester encoder</td>
<td></td>
<td></td>
<td>LED indication</td>
<td></td>
</tr>
<tr>
<td>Induction cookers</td>
<td>User interface: buttons, linear sliders</td>
<td>Temperature sensor</td>
<td></td>
<td></td>
<td>Segment LCD</td>
<td>LED indication</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Motor control systems</td>
<td>User interface: buttons, linear sliders</td>
<td></td>
<td></td>
<td>BLDC*** and FOC motor control</td>
<td></td>
<td></td>
<td>LED indication</td>
<td></td>
</tr>
<tr>
<td>Gaming / simulation controllers</td>
<td>User interface: buttons, touchpads</td>
<td>Reading analog joysticks</td>
<td></td>
<td></td>
<td>I²C/SPI/UART communication interface</td>
<td>Segment LCD</td>
<td>LED indication</td>
<td></td>
</tr>
</tbody>
</table>
In this application, the 12-bit 1-MspS SAR ADC in the PSoC 4 detects over current, over voltage, and high temperature conditions. The PWM output drives the speaker for status and alarm tones. Another PWM controls the heating element in the system. The CapSense buttons and slider constitute the user interface. PSoC 4 can also drive a segment LCD for visual outputs. PSoC 4 has a serial communication block that can connect to the main board of the system.

CapSense Plus systems, such as this example, allow you to reduce your board size, BOM cost, and power consumption.
8. Resources

8.1 Website
Visit the Getting Started with PSoC 4, Getting Started with PSoC 4 BLE, or Getting Started with PRoC BLE website to understand the PSoC 4 and PRoC BLE devices.

8.2 Datasheet
PSoC 4 Datasheet
PSoC 4 BLE Datasheet
PRoC BLE Datasheet

8.3 Technical Reference Manual
The PSoC 4 Technical Reference Manual (TRM) provides quick and easy access to information on PSoC 4 architecture including top-level architectural diagrams, register summaries, and timing diagrams.

8.4 Development Kits
Table 8-1 lists Cypress development kits that support PSoC 4 CapSense.

<table>
<thead>
<tr>
<th>Development Kit</th>
<th>Supported CapSense Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSoC 4 Pioneer Kit (CY8CKIT-042)</td>
<td>A 5-segment linear slider</td>
</tr>
<tr>
<td>PSoC 4 BLE Bluetooth Low Energy Pioneer Kit (CY8CKIT-042-BLE)</td>
<td>A 5-segment linear slider and a wire proximity sensor</td>
</tr>
<tr>
<td>PSoC 4200-M Pioneer Kit(CY8CKIT-044)</td>
<td>A 5-element gesture detection, two proximity wire sensors</td>
</tr>
<tr>
<td>PSoC 4 Processor Module (CY8CKIT-038), with PSoC Development Kit (CY8CKIT-001)</td>
<td>A 5-segment linear slider and two buttons</td>
</tr>
<tr>
<td>CapSense Expansion Board Kit (CY8CKIT-031), to be used with CY8CKIT-038 and CY8CKIT-001</td>
<td>A 10-segment slider, 5 buttons and a 4x4 matrix button with LED indication.</td>
</tr>
<tr>
<td>MiniProg3 Program and Debug Kit (CY8CKIT-002)</td>
<td>CapSense performance tuning in CY8CKIT-038</td>
</tr>
<tr>
<td>PSoC 4000 Pioneer Kit (CY8CKIT-040)</td>
<td>A 5x6 CapSense Touchpad and a wire proximity sensor</td>
</tr>
</tbody>
</table>
Table 8-2 lists Cypress Reference Design kits that support PRoC BLE CapSense.

<table>
<thead>
<tr>
<th>Reference Design Kit</th>
<th>Supported CapSense Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRoC BLE Remote Control Reference Design Kit (CY5672)</td>
<td>An 8 x 8 trackpad with these gestures:</td>
</tr>
<tr>
<td></td>
<td>• Top edge swipe gesture</td>
</tr>
<tr>
<td></td>
<td>• One finger click gesture</td>
</tr>
<tr>
<td></td>
<td>• Two finger click gesture</td>
</tr>
<tr>
<td></td>
<td>• Pinch zoom gesture</td>
</tr>
<tr>
<td></td>
<td>• One finger horizontal scroll gesture</td>
</tr>
<tr>
<td></td>
<td>• One finger vertical scroll gesture</td>
</tr>
<tr>
<td>PRoC BLE Touch Mouse Reference Design Kit (CY5682)</td>
<td>A 9 x 3 trackpad with these gestures:</td>
</tr>
<tr>
<td></td>
<td>• One finger horizontal scroll gesture</td>
</tr>
<tr>
<td></td>
<td>• One finger vertical scroll gesture</td>
</tr>
</tbody>
</table>

8.5 PSoC Creator

PSoC Creator is a state-of-the-art, easy-to-use integrated development environment. For details, see the PSoC Creator home page.

8.6 Application Notes

Cypress offers a large collection of application notes to get your design up and running fast. See PSoC 4 Application Notes, PSoC 4 BLE Application Notes or PRoC BLE Application Notes, CapSense Design Guides.

8.7 Design Support

Cypress has a variety of design support channels to ensure the success of your CapSense solutions.

- Knowledge Base Articles – Browse technical articles by product family or perform a search on CapSense topics.
- White Papers – Learn about advanced capacitive-touch interface topics.
- Cypress Developer Community – Connect with the Cypress technical community and exchange information.
- Video Library – Quickly get up to speed with tutorial videos.
- Quality & Reliability – Cypress is committed to complete customer satisfaction. At our Quality website, you can find reliability and product qualification reports.
- Technical Support – World-class technical support is available online.
## Revision History

**Document Revision History**

**Document Title:** AN85951 – PSoC® 4 CapSense® Design Guide

**Document Number:** 001-85951

<table>
<thead>
<tr>
<th>Revision</th>
<th>ECN#</th>
<th>Issue Date</th>
<th>Origin of Change</th>
<th>Description of Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>**</td>
<td>3973432</td>
<td>04/19/2013</td>
<td>NIDH</td>
<td>New Design Guide</td>
</tr>
<tr>
<td>*A</td>
<td>4059171</td>
<td>07/29/2013</td>
<td>NIDH</td>
<td>Added dual IDAC support. Updated some schematics in chapter 6. Other minor changes to chapters 3, 5, and 6.</td>
</tr>
<tr>
<td>*B</td>
<td>4189700</td>
<td>11/13/2013</td>
<td>NIDH</td>
<td>Added support of CY8C4000 devices. Minor fixes throughout the document.</td>
</tr>
<tr>
<td>*C</td>
<td>4289925</td>
<td>02/24/2014</td>
<td>NIDH</td>
<td>Updated the table of device features. Changed IDAC names to sync with new PSoC Creator Component terms. Added a schematic checklist. Changed screenshots to match the new Component version.</td>
</tr>
<tr>
<td>*D</td>
<td>4293476</td>
<td>02/27/2014</td>
<td>NIDH</td>
<td>Updated Table 1-1 per PSoC 4000 datasheet.</td>
</tr>
<tr>
<td>*E</td>
<td>4314223</td>
<td>03/20/2014</td>
<td>NIDH</td>
<td>Added firmware design considerations to Chapter 6. Added power supply layout and schematic considerations to Chapter 6. Updated the IMO range for PSoC 4000</td>
</tr>
<tr>
<td>*F</td>
<td>4339713</td>
<td>04/15/2014</td>
<td>NIDH</td>
<td>Updated to support PSoC 4000 and PSoC Creator 3.0 SP1.</td>
</tr>
<tr>
<td>*G</td>
<td>4494249</td>
<td>08/29/2014</td>
<td>DCHE</td>
<td>Added Reference to Getting Started with CapSense in Section 2.3.4 Proximity (Three-Dimensional) Renamed Section 2.4 to Liquid Tolerance and re-wrote this section. Updated the recommendations for Shield drive i.e Csh_tank precharge and Cmod precharge in Section 3.2 CapSense CSD Shielding and Section 5.1.3.1 Advanced Settings respectively Added recommendation for setting “API resolution” in Section 5.1.2 Widget Configuration Added guidelines on how to select value of “Sensitivity” parameter in Section 5.1.3 Scan Order Updated recommended values of threshold and hysteresis parameters in Section 5.2.2 Manual Tuning Process Added Section 5.2.2.1 Manual Tuning For Slider Updated maximum overlay thickness value for sliders in Table 6-2. Maximum Thickness of Acrylic Overlay Added guideline on maximum thickness for overlays of materials other than acrylic in Section 6.2.2 Overlay Thickness Re-wrote Section 6.3.4 Slider Design</td>
</tr>
</tbody>
</table>
## Revision History

**Document Title: AN85951 – PSoC® 4 CapSense® Design Guide**  
**Document Number: 001-85951**

<table>
<thead>
<tr>
<th>Revision</th>
<th>ECN#</th>
<th>Issue Date</th>
<th>Origin of Change</th>
<th>Description of Change</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Added recommendations on DC loads in Section 6.3.5 Sensor Device and Placement</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Renamed Section 6.3.12 to Layout Guidelines for Liquid Tolerance and re-wrote this section</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Added Section</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Updated slider related recommendations in Table 6-8. Layout Rule Checklist</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Updated Section 6.5 Electromagnetic Compatibility (EMC) Considerations, added extensive data on hardware and firmware considerations.</td>
</tr>
<tr>
<td>*H</td>
<td>4602375</td>
<td>12/19/2014</td>
<td>UDYG</td>
<td>Added information for the PSoC 4 BLE family of devices.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Added information for the PRoC BLE family of devices.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Updated ground and power layout guidelines in Section 6.3.10 and Section 6.3.11.</td>
</tr>
<tr>
<td>*I</td>
<td>4624027</td>
<td>01/21/2015</td>
<td>NIDH</td>
<td>Added information for PSoC 4200-M family of devices</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Added footnote in section 6.3.4 Slider Design</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Added GPIO source/sink current limit in Table 6-6.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Changed document title to PSoC® 4 CapSense® Design Guide – AN85951</td>
</tr>
<tr>
<td>*J</td>
<td>4771699</td>
<td>06/02/2015</td>
<td>DCHE</td>
<td>Changed Document Title to read as “AN85951 – PSoC® 4 CapSense® Design Guide”.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Updated Design Considerations:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Updated ESD Protection:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Updated Preventing ESD Discharge:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Updated Figure 6-31.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Updated Redirect:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Replaced “Guard Ring” with “Ground Ring”.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Updated Figure 6-32.</td>
</tr>
<tr>
<td>*K</td>
<td>4891423</td>
<td>08/20/2015</td>
<td>DCHE</td>
<td>Added Table 3-1.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Removed section 3.2.1 CMOD Precharge.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Added section CapSense in PSoC 4 M-Series.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Updated section Trace Routing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Added reference of AN2397.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Added recommendation for modulator clock divider in section Manual Tuning Process.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Added Figure 6-30.</td>
</tr>
</tbody>
</table>