AN85951 - PSoC® 4 CapSense® Design Guide

Doc. No. 001-85951 Rev. *K

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

Copyrights

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 41
5.2 Manual Tuning 45
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, C_P 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

- 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 / I^2C 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 I^2S)
- 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 1-1. PSoC 4 Device Families

Features	Device Family					
	CY8C4000	CY8C4100	CY8C4100-BL	CY8C4200	CY8C4200-BL	CYBL10X6X
CPU	16-MHz Cortex-M0 CPU	24-MHz Cortex-M0 CPU with single-cycle multiply	24-MHz Cortex-M0 CPU with single-cycle multiply	48-MHz Cortex-M0 CPU with single-cycle multiply	4- MHz Cortex-M0 CPU with single-cycle multiply	48-MHz Cortex-M0 CPU with single-cycle multiply
Flash	As much as 16 KB	As much as 128 KB	128 KB	As much as 128 KB	128 KB	128 KB
SRAM	As much as 2 KB	As much as 16 KB	16 KB	As much as 16 KB	16 KB	16 KB
GPIOs	As many as 20	As many as 55	36	As many as 55	36	36
CapSense	As many as 16 sensors	As many as 54 sensors	As many as 35 sensors	As many as 54sensors	As many as 35 sensors	As many as 35 sensors
BLE	None	None	BLE 4.0 with multiple optional features of BLE 4.1	None	BLE 4.0 with multiple optional features of BLE 4.1	BLE 4.0 with multiple optional features of BLE 4.1
ADC	None	12-bit, 806-ksps SAR ADC with sequencer	12-bit, 806-ksps SAR ADC with sequencer	12-bit, 1-Msps SAR ADC with sequencer	12-bit, 1-Msps SAR ADC with sequencer	12-bit, 1-Msps SAR ADC with sequencer
Opamps	None	4 programmable opamps	2 programmable opamps	4 programmable opamps	4 programmable opamps	None
Comparators	1 CSD comparator with fixed threshold (1.2 V)	2 low-power comparators with wakeup feature	2 low-power comparators with wakeup feature	2 low-power comparators with wakeup feature	2 low-power comparators with wakeup feature	None
IDACs	One 7-bit and one 8-bit	Two 7-bit and two 8-bit	One 7-bit and one 8-bit	Two 7-bit and two 8-bit	One 7-bit and one 8-bit	None
Programmable logic blocks (UDBs)	None	None	None	Four, each with 8 macrocells and one datapath	Four, each with 8 macrocells and one datapath	None
Power supply range	1.71 to 5.5 V	1.71 to 5.5 V	1.71 to 5.5 V	1.71 to 5.5 V	1.71 to 5.5 V	1.71 to 5.5 V
Low power modes	Deep-Sleep mode at 2.5 µA	Deep-Sleep mode at 1.3 µA Hibernate mode at 150 nA Stop mode at 20 nA	Deep-Sleep mode at 1.3 µA Hibernate mode at 150 nA Stop mode at 60 nA	Deep-Sleep mode at 1.3 µA Hibernate mode at 150 nA Stop mode at 20 nA	Deep-Sleep mode at 1.3 µA Hibernate mode at 150 nA Stop mode at 60 nA	Deep-Sleep mode at 1.3 µA Hibernate mode at 150 nA Stop mode at 60 nA
Segment LCD drive	None	4 COM segment LCD drive	4 COM segment LCD drive	4 COM segment LCD drive	4 COM segment LCD drive	4 COM segment LCD drive
Serial communication	1 I^2C	2 independent Serial Communication Blocks (SCBs) with programmable I^2C, SPI, or UART. CAN, and LIN for automotive	2 independent Serial Communication Blocks (SCBs) with programmable I^2C, SPI, or UART	2 independent Serial Communication Blocks (SCBs) with programmable I^2C, SPI, or UART, CAN, and LIN for automotive	2 independent Serial Communication Blocks (SCBs) with programmable I^2C, SPI, or UART	1 or 2 independent Serial Communication Blocks (SCBs) with programmable I^2C, SPI, or UART
Timer / Counter / Pulse-Width Modulator (TCPWM)	1	8	4	8	4	As many as 4
Other Fixed Function Peripherals (I^2S, PWM)	None	None	None	None	None	I^2S, PWM

Features	Device Family					
	CY8C4000	CY8C4100	CY8C4100-BL	CY8C4200	CY8C4200-BL	CYBL10X6X
Clocks	24- / 32-MHz internal main oscillator (IMO) 32-kHz internal low-speed oscillator (ILO) External Clock (0–16 MHz)	3- to 24-MHz internal main oscillator (IMO) 32-kHz internal low-speed oscillator (ILO) External Clock (0–24 MHz)	3- to 24-MHz internal main oscillator (IMO) 32-kHz internal low-speed oscillator (ILO) External Clock (0-24 MHz) 24-MHz External Crystal Oscillator 32 KHz Watch Crystal Oscillator	3- to 48-MHz internal main oscillator (IMO) 32-kHz internal low-speed oscillator (ILO) External Clock (0-48 MHz)	3-to-48 MHz internal main oscillator (IMO) 32-kHz internal low-speed oscillator (ILO) External Clock (0-48 MHz) 24-MHz External Crystal Oscillator 32 KHz Watch Crystal Oscillator	3- to 48-MHz internal main oscillator (IMO) 32-kHz internal low-speed oscillator (ILO) External Clock (0-48 MHz) 24-MHz External Crystal Oscillator 32-kHz Watch Crystal Oscillator
Power supply monitoring	Power-on reset (POR) Brown-out detection (BOD)	Power-on reset (POR) Brown-out detection (BOD) Low-voltage detection (LVD)	Power-on reset (POR) Brown-out detection (BOD) Low-voltage detection (LVD)	Power-on reset (POR) Brown-out detection (BOD) Low-voltage detection (LVD)	Power-on reset (POR) Brown-out detection (BOD) Low-voltage detection (LVD)	Power-on reset (POR) Brown-out detection (BOD) Low-voltage detection (LVD)

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

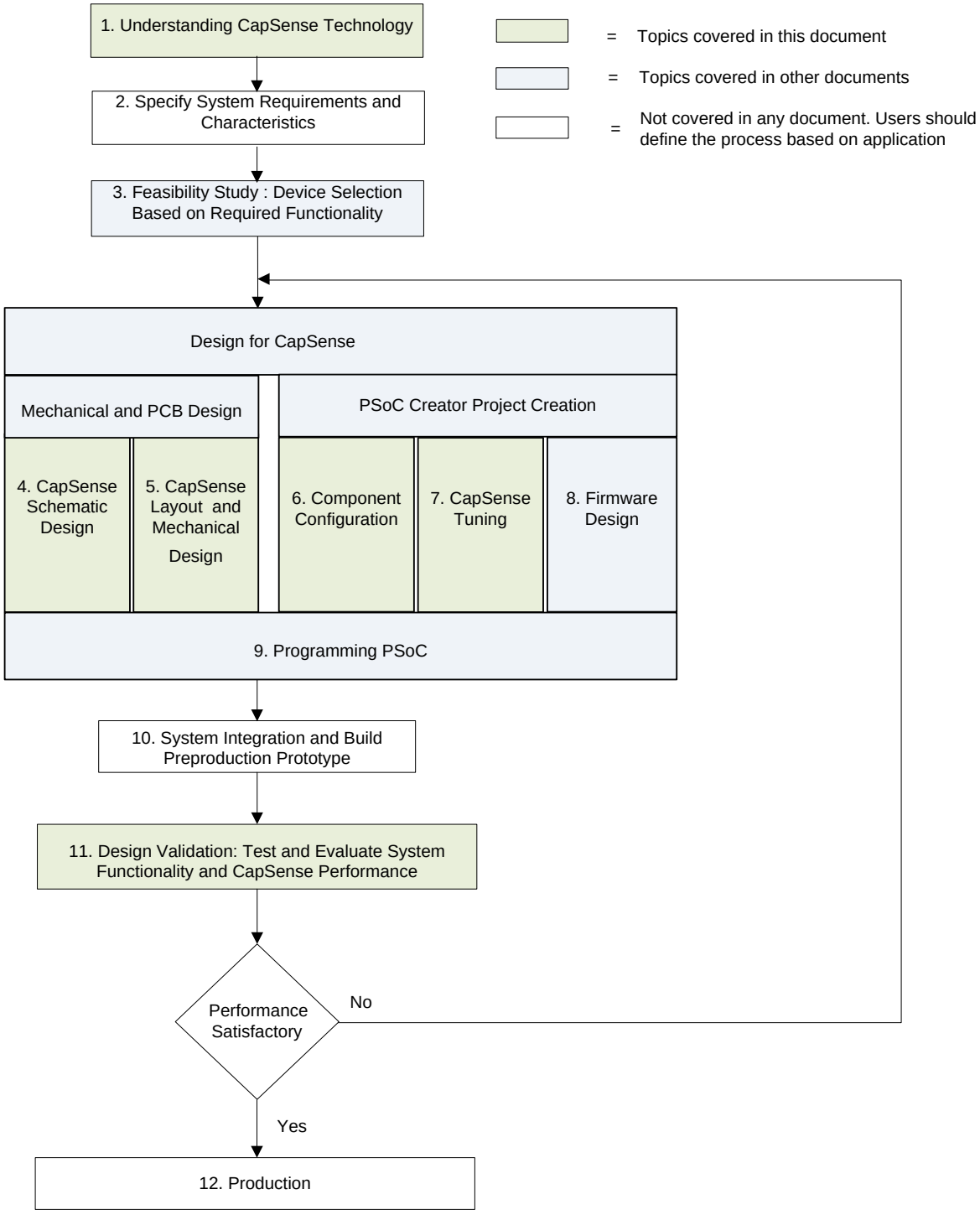


Table 1-2. Supporting Documentation

Steps in Flowchart	Supporting Cypress Documentation	
	Name	Chapter
1. Understanding CapSense	PSoC 4 CapSense Design Guide (This document)	Chapter 2 and Chapter 3
2. Specify requirements	–	–
3. Feasibility study	PSoC 4 Datasheet PSoC 4 BLE Datasheet PRoC BLE Datasheet	–
	AN79953, Getting Started with PSoC 4 AN91267, Getting Started with PSoC 4 BLE AN94020, Getting Started with PRoC BLE	–
4. Schematic design	PSoC 4 CapSense Design Guide (This document)	Chapter 6
5. Layout design	PSoC 4 CapSense Design Guide (This document)	Chapter 6
6. Component configuration	CapSense Component datasheet	–
	PSoC 4 CapSense Design Guide (This document)	Chapter 5
7. Performance tuning	PSoC 4 CapSense Design Guide (This document)	Chapter 5
8. Firmware design	PSoC Creator CapSense Component datasheet	–
	PSoC Creator example projects	–
9. Programming PSoC	PSoC Creator home page	–
10. Prototype	–	–
11. Design validation	PSoC 4 CapSense Design Guide (This document)	Chapter 5
12. Production	–	–

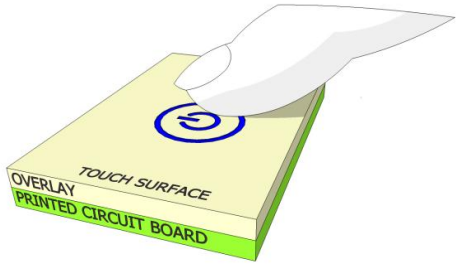
2. CapSense Technology

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.

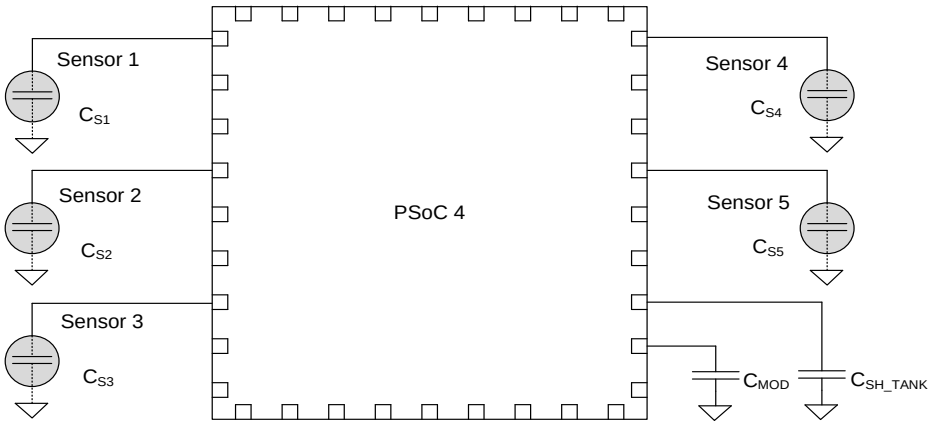
Figure 2-1. Capacitive Touch Sensor



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.

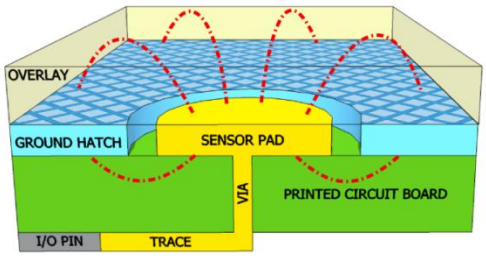
Figure 2-2. PSoC Device, Sensors, and External Capacitors



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.

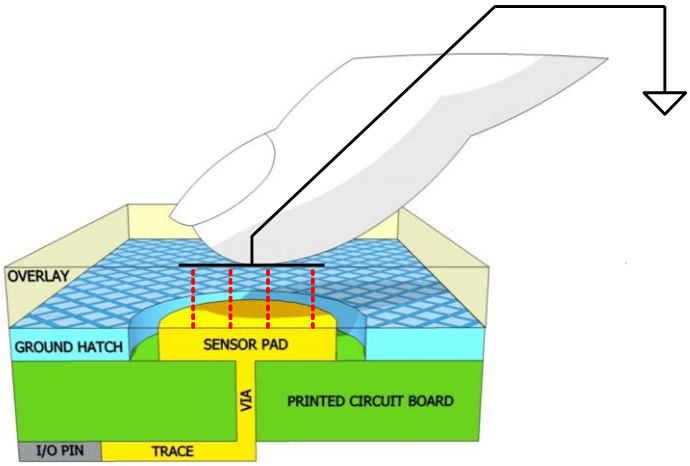
Figure 2-3 shows how a sensor GPIO pin is connected to a sensor pad by traces and vias. Typically, a ground hatch surrounds the sensor pad to isolate it from other sensors and traces. Although Figure 2-3 shows some field lines around the sensor pad, the actual electric field distribution is very complex.

Figure 2-3. Parasitic Capacitance



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



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

$$C_F = \frac{\varepsilon_0 \varepsilon_r A}{d} \qquad (2-1)$$

Where:

ε_0 = Free space permittivity

ε_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.

$$C_S = C_P + C_F \qquad (2-2)$$

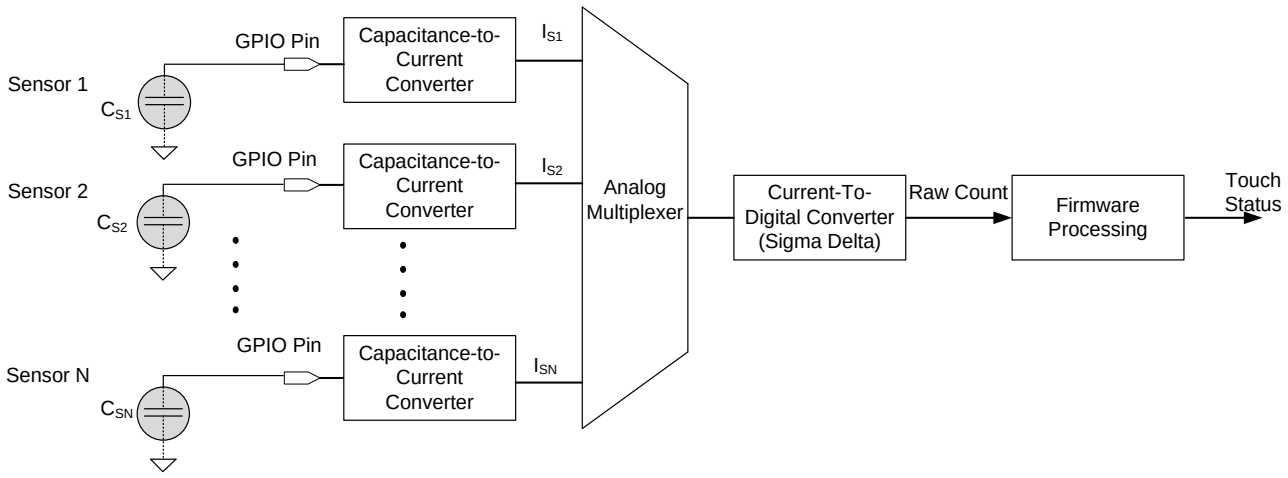
In the absence of touch, C_S is equal to C_P.

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.

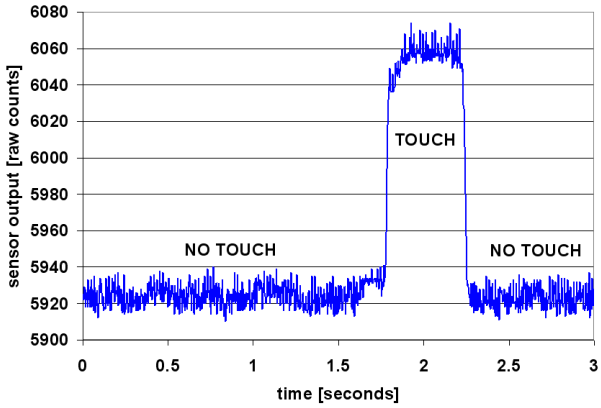
The output count of the current to digital converter, known as **raw count**, is a digital value that is proportional to the sensor capacitance:

$$\text{raw count} = G_C C_S \qquad (2-3)$$

Where G_C is the capacitance to digital conversion gain of CapSense.

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.

Figure 2-6. Raw Count Versus Time



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.

Figure 2-7. Several types of Widgets

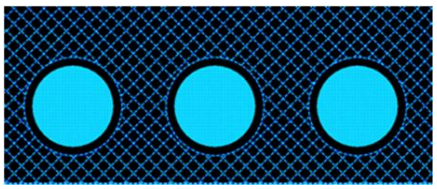
Button Sensor | Slider Sensor | Proximity Sensor

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.

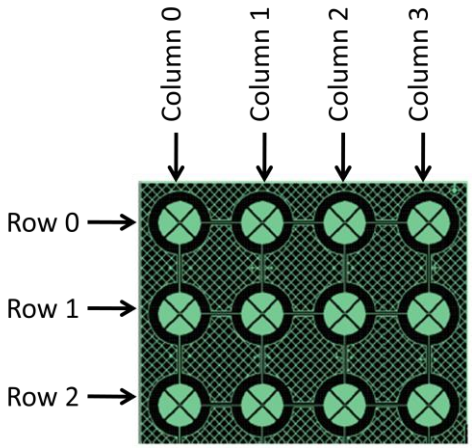
A simple CapSense button consists of a circular copper pad connected to a PSoC GPIO using PCB traces, as Figure 2-8 shows. The button is surrounded by grounded copper hatch to isolate it from other buttons and traces. A circular gap separates the button pad and the ground hatch. Each button requires one PSoC GPIO.

Figure 2-8. Simple CapSense Buttons



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.

Figure 2-9. Matrix Buttons



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.

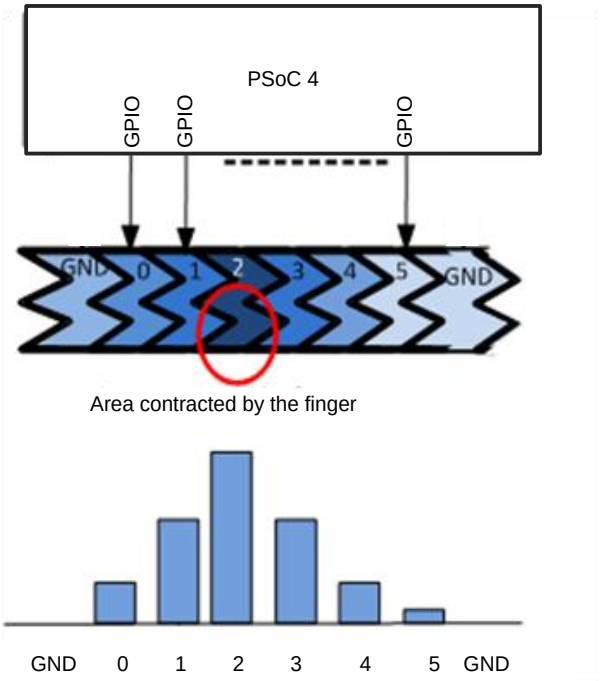
2.3.2 Sliders (One-Dimensional)

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.

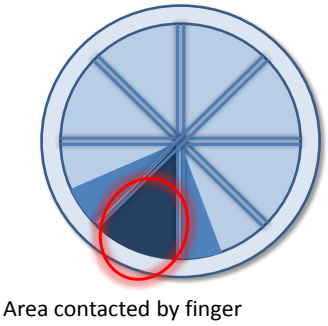
In a linear slider, the segments are arranged inline, as Figure 2-10 shows. Each slider segment connects to a PSoC GPIO. A zigzag pattern (double chevron) is recommended for slider segments. This layout ensures that when a segment is touched, the adjacent segments are also partially touched, which aids estimation of the centroid position.

Figure 2-10. Linear Slider



Radial sliders are similar to linear sliders except that radial sliders are continuous. Figure 2-11 shows a typical radial slider.

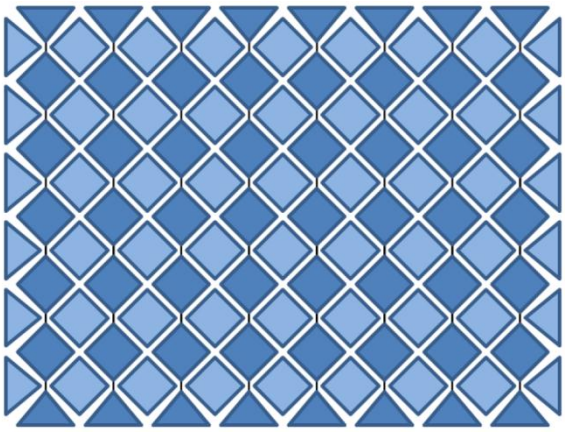
Figure 2-11. Radial Slider



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.

Figure 2-12. Trackpad Sensor Arrangement

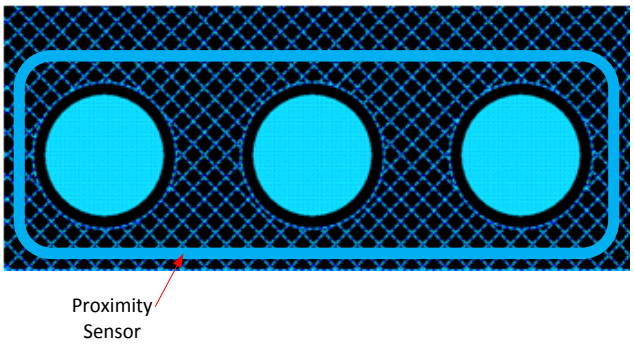


2.3.4 Proximity (Three-Dimensional)

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.

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.

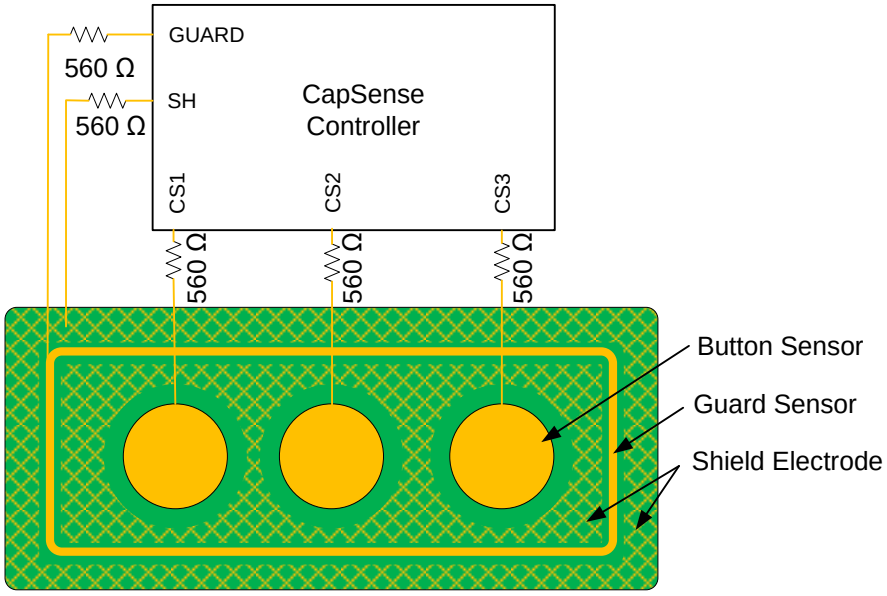
2.4 Liquid Tolerance

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.

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

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.

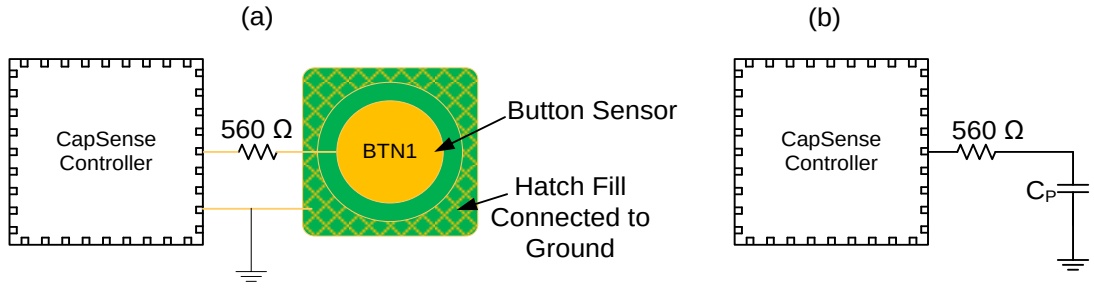
Figure 2-15. Shield Electrode (SH) and Guard Sensor (GUARD) Connected to CapSense Controller



2.4.1 Effect of Liquid Droplets and Liquid Stream on a CapSense Sensor

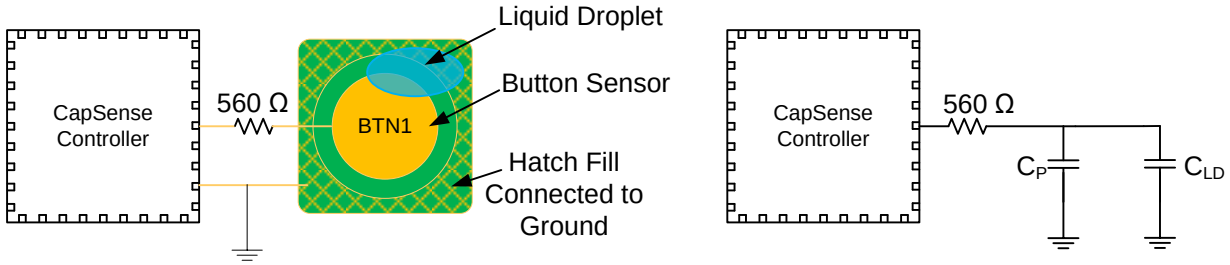
To understand the effect of liquids on a CapSense sensor, consider a CapSense system in which the hatch fill around the sensor is connected to ground, as Figure 2-16(a) shows. The hatch fill when connected to a ground improves the noise immunity of the sensor. Parasitic capacitance of the sensor is denoted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

Figure 2-16. Typical CapSense System Layout



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.

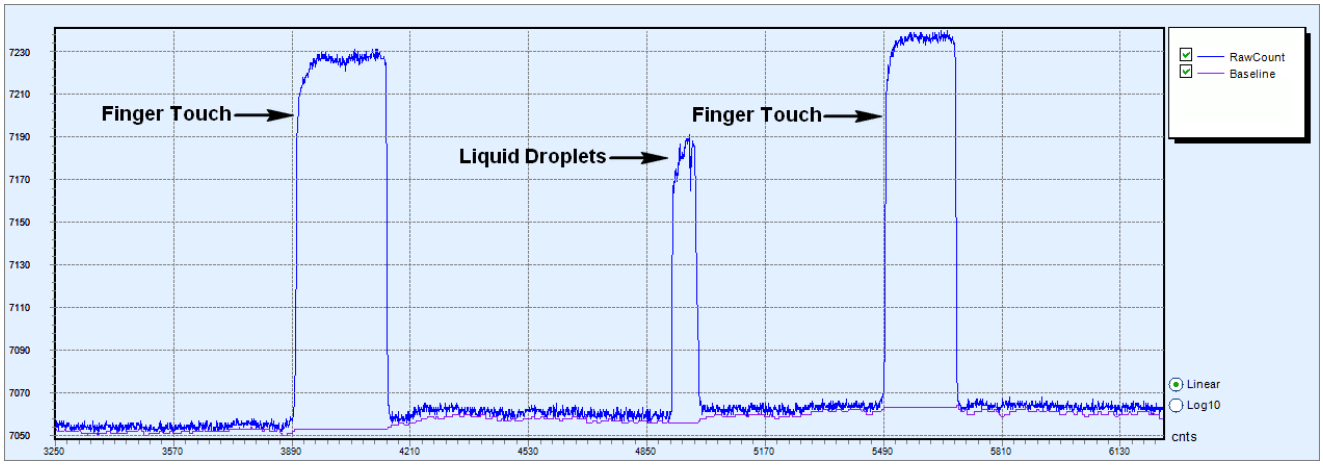
Figure 2-17. Capacitance Added by Liquid Droplet when the Hatch Fill is Connected to Ground



C_P – Sensor Parasitic Capacitance

C_{LD} – Capacitance Added by the Liquid Droplet

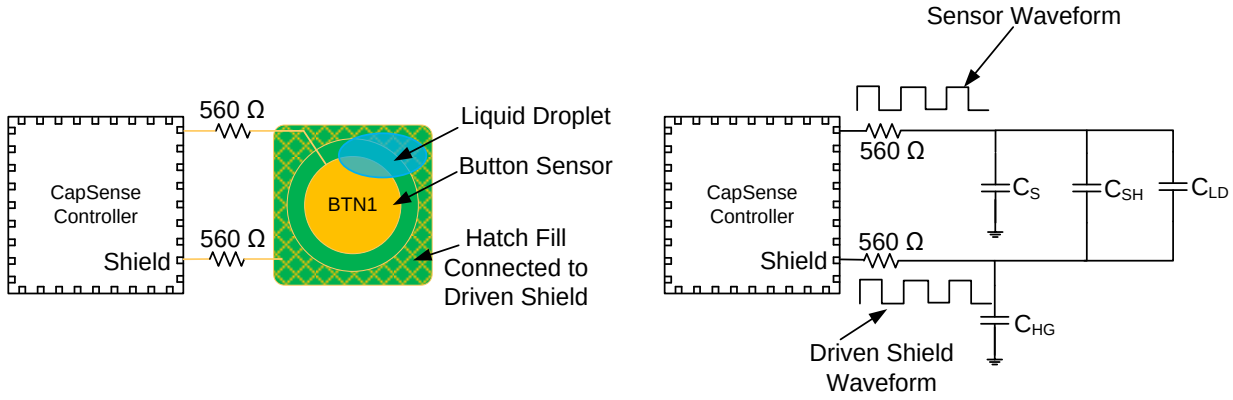
Figure 2-18. Effect of Liquid Droplet when the Hatch Fill Around the Sensor is Connected to Ground



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



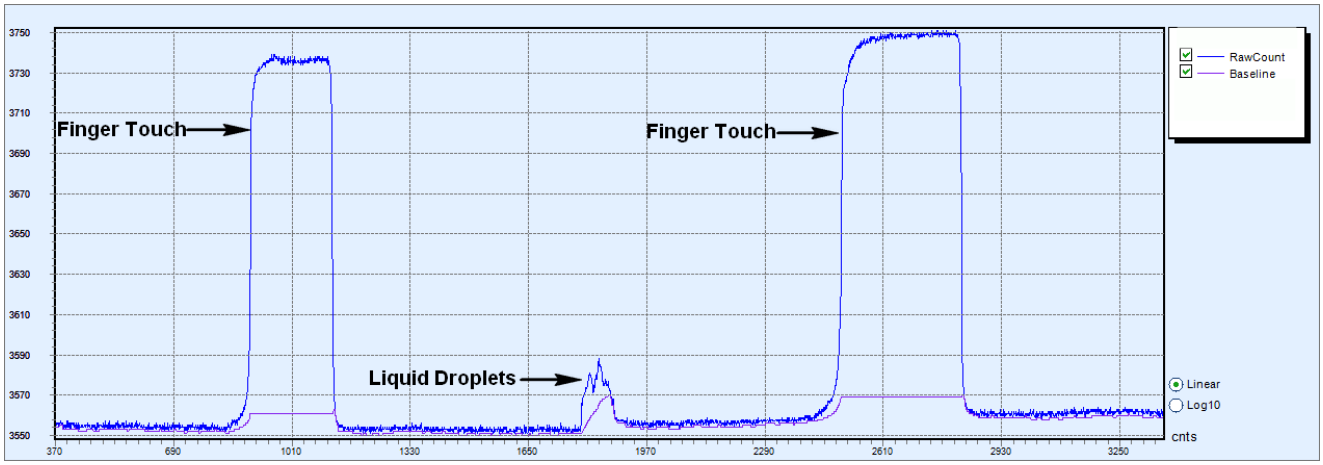
C_S – Sensor Parasitic Capacitance

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

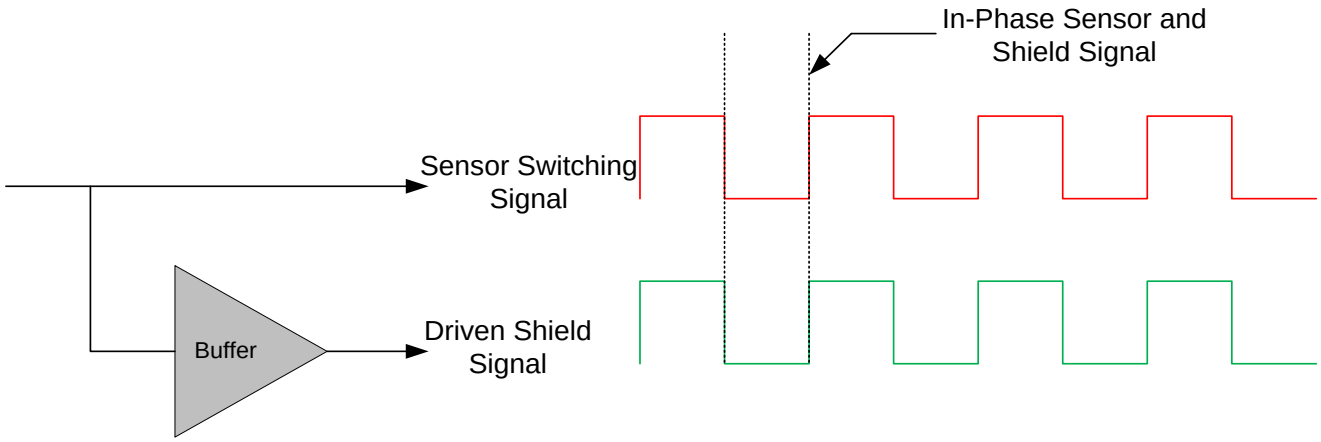
Figure 2-20. Effect of Liquid Droplet when the Hatch Fill Around the Sensor is Connected to the Driven-Shield



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



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.

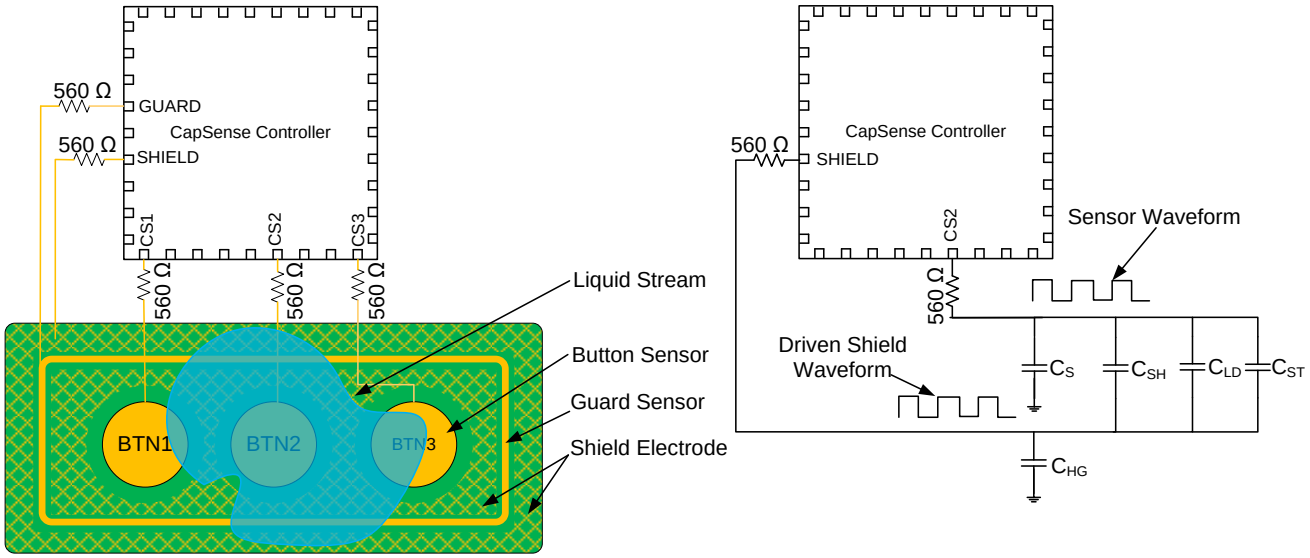
2.4.3 Guard Sensor

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

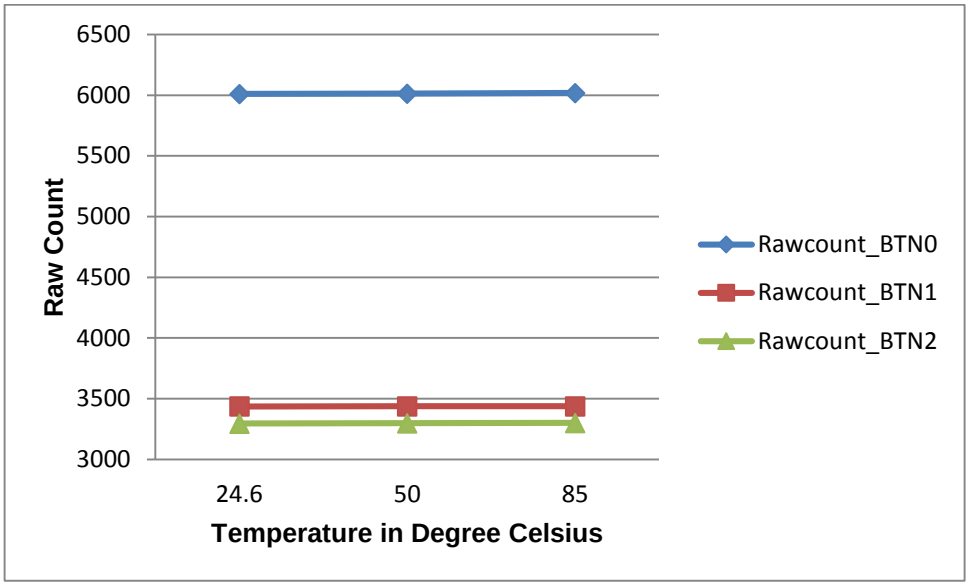


2.4.3.1 Effect of Liquid Properties on the Liquid Tolerance Performance

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.

3. PSoC 4 and PRoC BLE CapSense

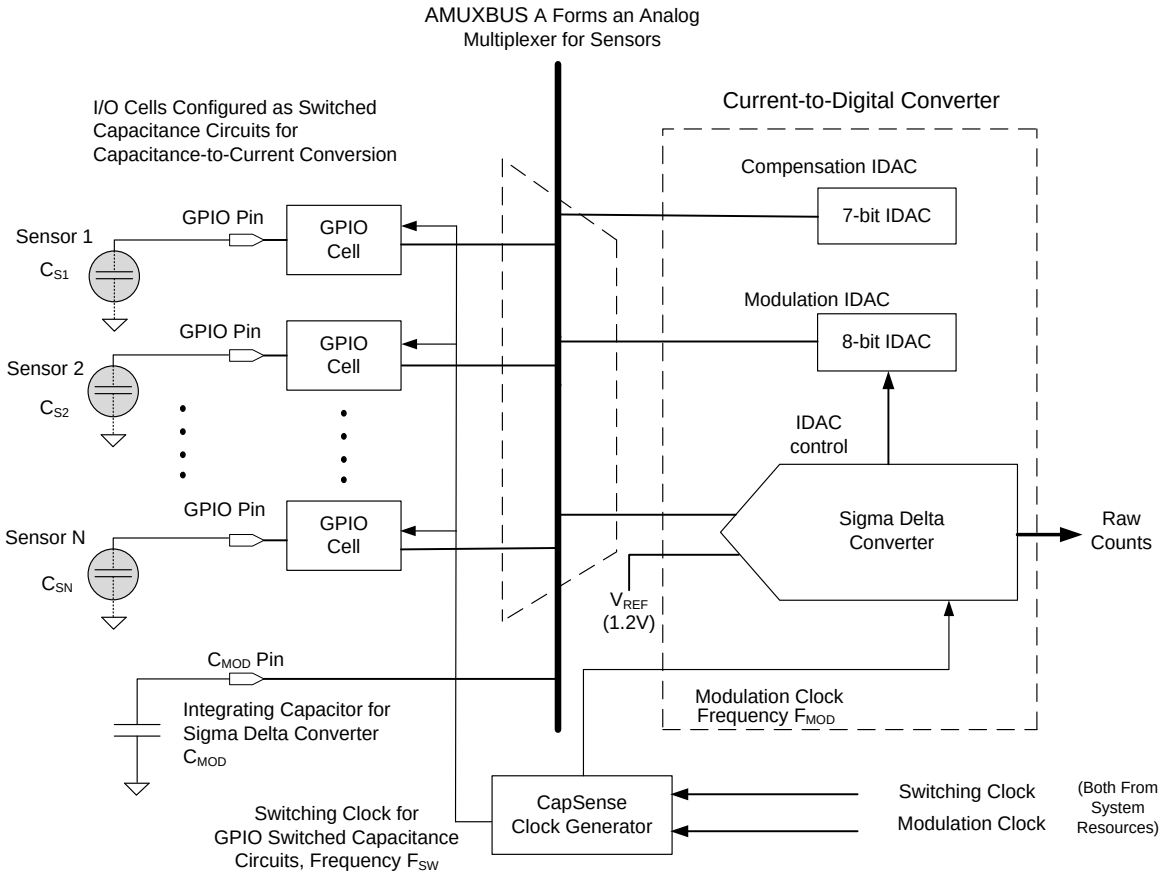
This chapter explains in detail how CapSense CSD is implemented in the PSoC 4 and PRoC BLE devices. See Capacitive Touch Sensing Method to understand the basic principles of CapSense CSD. A basic knowledge of the PSoC 4 or PRoC BLE device architecture is a prerequisite for this chapter. If you are new 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.

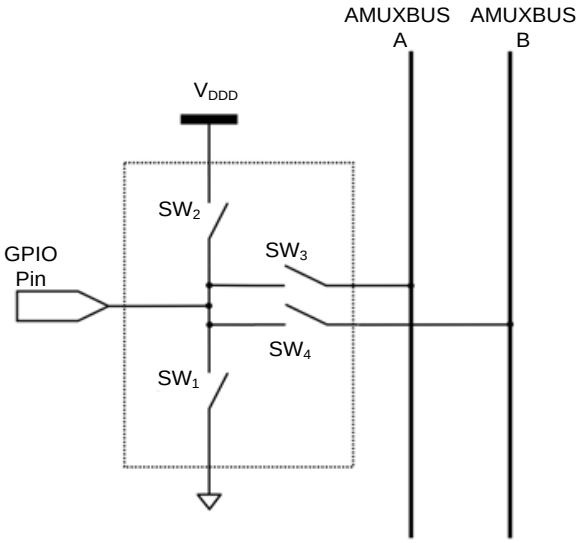
Figure 3-1. PSoC 4 / PRoC BLE CapSense CSD Sensing



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

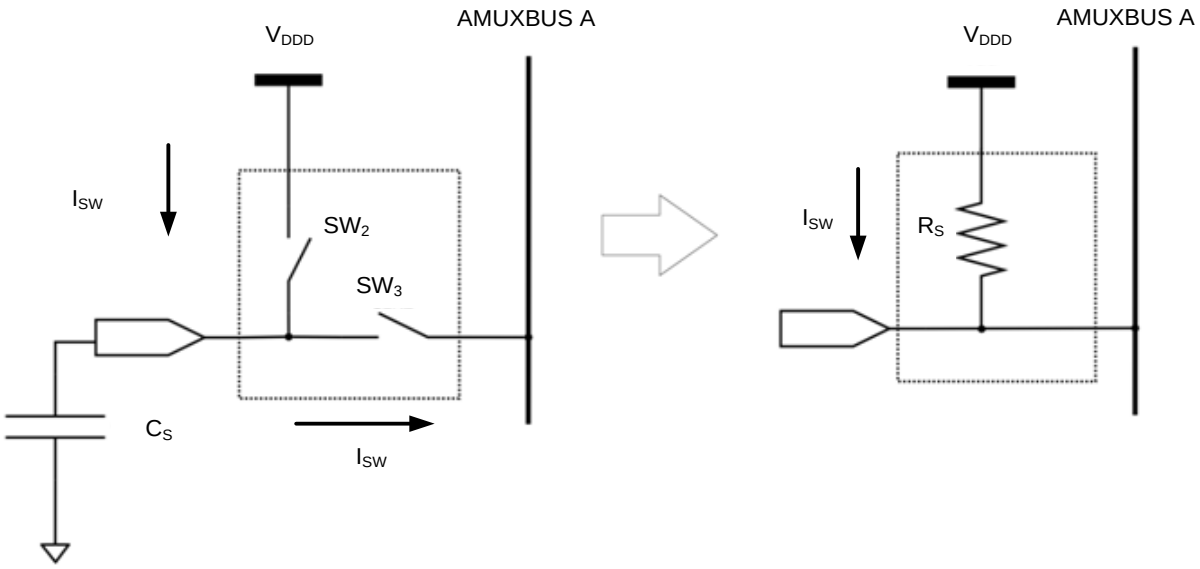


PSoC 4 and PRoC BLE devices have two analog multiplexer buses: AMUXBUS A is used for CSD sensing and AMUXBUS B is used for CSD shielding. The GPIO switched-capacitance circuit has two possible configurations: source current to AMUXBUS A or sink current from AMUXBUS A.

3.1.1.1 Current Sourcing

Figure 3-3 shows the switched-capacitance configuration for sourcing current to AMUXBUS A.

Figure 3-3. 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:

$$R_S = \frac{1}{C_S F_{SW}} \qquad (3-1)$$

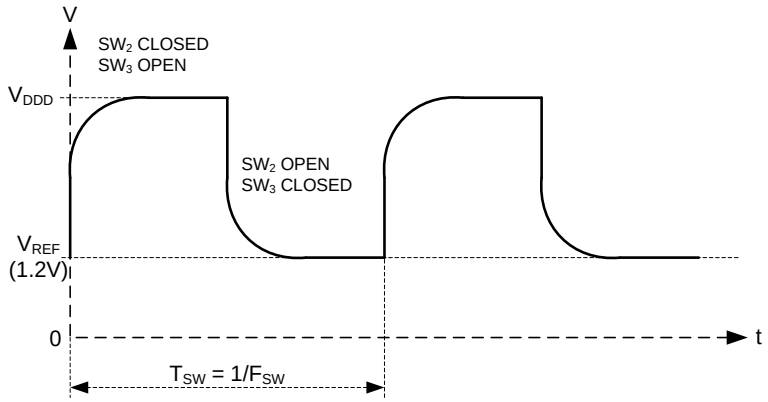
Where:

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.

Figure 3-4. Voltage Across Sensor Capacitance



Equation 3-2 gives the value of average current supplied to AMUXBUS A.

$$I_{CS} = C_S F_{SW} (V_{DDD} - V_{REF}) \qquad (3-2)$$

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 C_S.

Figure 3-5. Sinking Current from AMUXBUS A

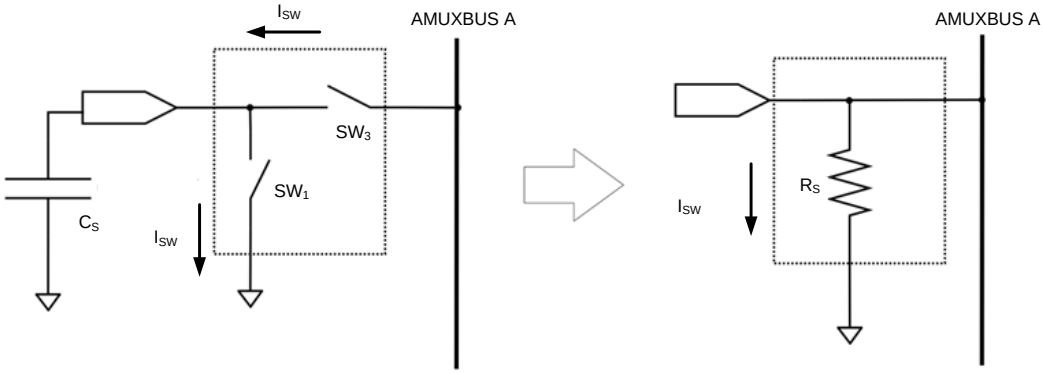
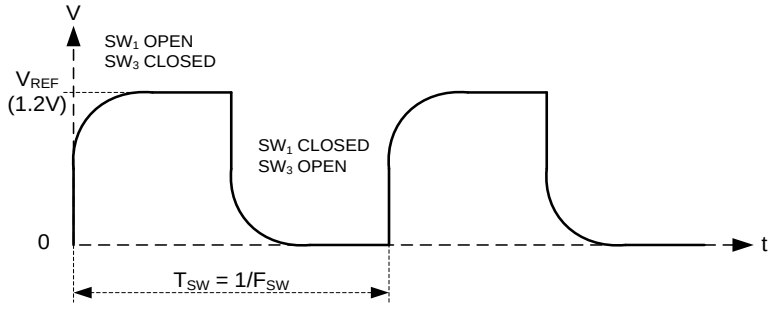


Figure 3-6. Voltage Across Sensor Capacitance



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

$$I_{CS} = C_S F_{SW} V_{REF} \qquad (3-3)$$

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.

The modulation clock is used by the Sigma Delta converter. For more details on CapSense clocks, refer to the 'CapSense' and 'Clocking' chapters in the PSoC 4 Technical Reference Manual, PSoC 4 BLE Technical Reference Manual, or PRoC BLE Technical Reference Manual.

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 A 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.

$$\text{raw count} = (2^N - 1)\frac{V_{REF}\, F_{SW}}{I_{MOD}}\, C_S \qquad (3-4)$$

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

$$\text{raw count} = (2^N - 1)\frac{(V_{DD} - V_{REF})\, F_{SW}}{I_{MOD}}\, C_S \qquad (3-5)$$

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:

$$\text{raw count} = (2^N - 1)\frac{V_{REF}\, F_{SW}}{I_{MOD}}\, C_S - (2^N - 1)\frac{I_{COMP}}{I_{MOD}} \qquad (3-6)$$

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

$$\text{raw count} = (2^N - 1)\frac{(V_{DD} - V_{REF})\, F_{SW}}{I_{MOD}}\, C_S - (2^N - 1)\frac{I_{COMP}}{I_{MOD}} \qquad (3-7)$$

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

Sl.No	Parameter	CapSense Component Parameter	Comments
1	N	Scan Resolution	-
2	V_{REF}	N/A	The VREF value is 1.2 V
3	F_{SW}	Sense Clock Divider	Refer to CapSense Clocking Section in component datasheet to understand the relation between Sense Clock Divider and FSW
4	I_{MOD}	Modulation IDAC	-
5	I_{COMP}	Compensation IDAC	-
6	V_{DD}	N/A	This parameter is the device supply voltage
7	C_S	N/A	This parameter is the sensor parasitic capacitance

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 SW_3 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_{MOD} to the Sigma Delta converter circuit.

3.2 CapSense CSD Shielding

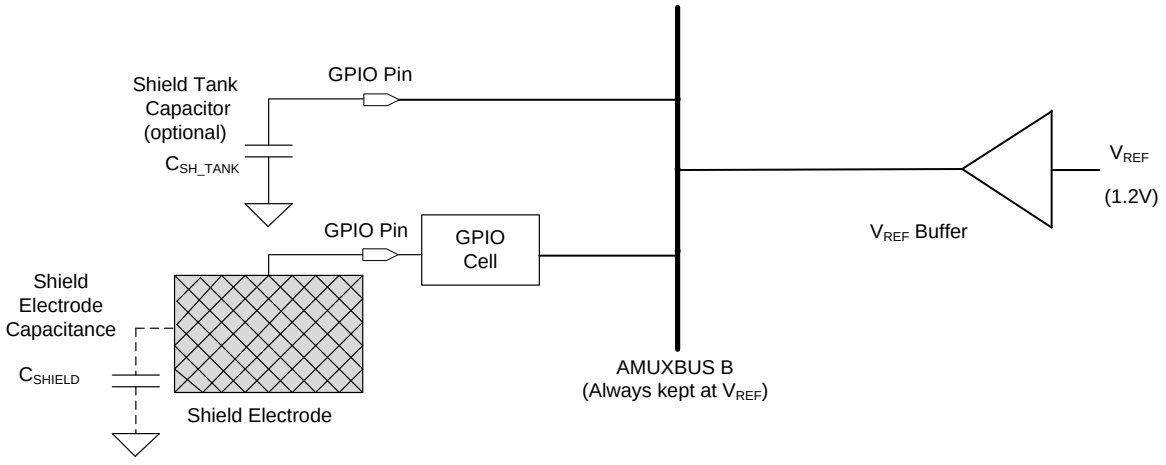
PSoC 4 / PRoC BLE CapSense supports shield electrodes for liquid tolerance and proximity sensing. See the Liquid Tolerance section for details. The shield electrode is always kept at the same potential as the sensors. CapSense has a shielding circuit that drives the shield electrode with a replica of 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_{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_{DD} or ground, depending on the configuration). The shielding circuit works in a similar way; AMUXBUS B is always kept at V_{REF}. The GPIO cell switches the shield between AMUXBUS B and a supply rail (either V_{DDD} 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_{REF}, two different configurations are possible:

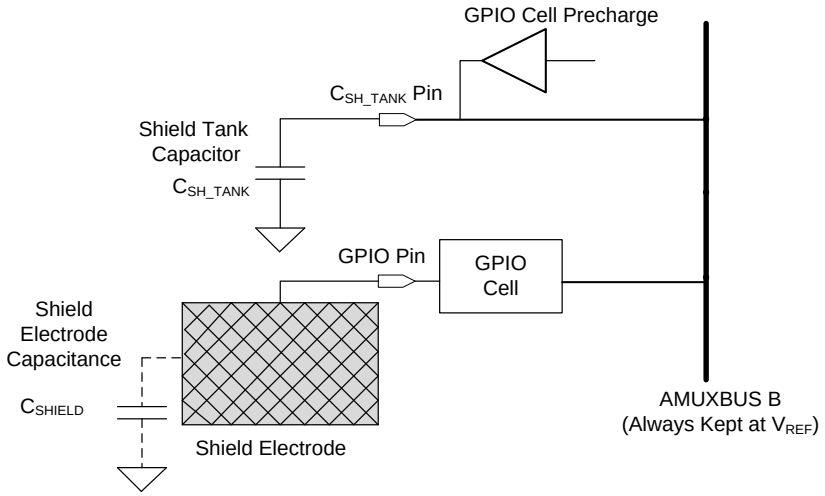
- Shield driving using V_{REF} buffer: In this configuration, a voltage buffer is used to drive AMUXBUS B to V_{REF}, as Figure 3-7 shows. An external 10-nF C_{SH_TANK} capacitor is recommended to reduce switching transients.

Figure 3-7. Shield Driving Using V_{REF} Buffer



- 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



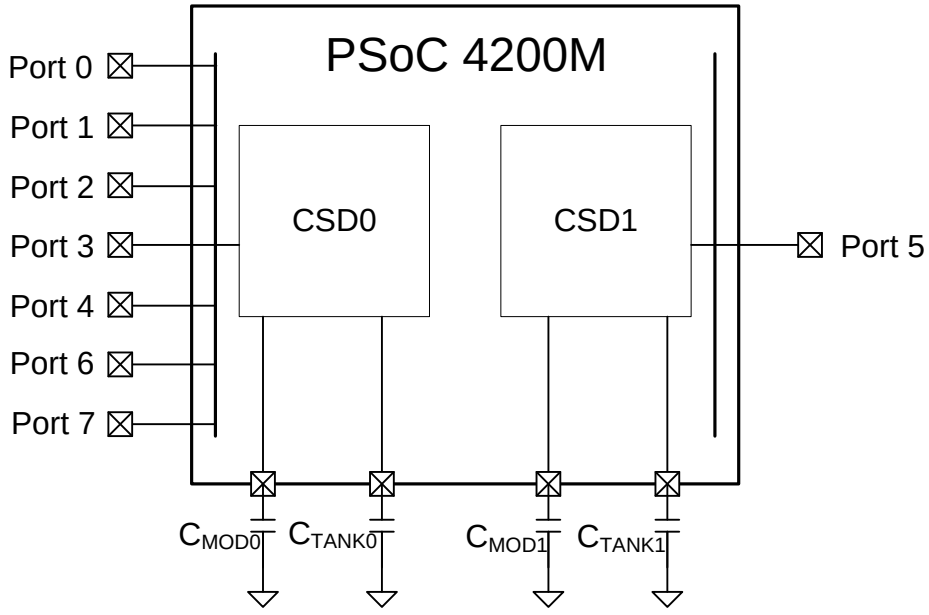
This GPIO cell precharge capability is available only on a fixed C_{SH_TANK} pin. See the device pinout in the PSoC 4 datasheet, PSoC 4 BLE Datasheet or PRoC BLE Datasheet for details.

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



Each CSD block requires separate C_{MOD} and C_{TANK} 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 3-2. Difference between CSD0 and CSD1 block in PSoC 4 M-Series

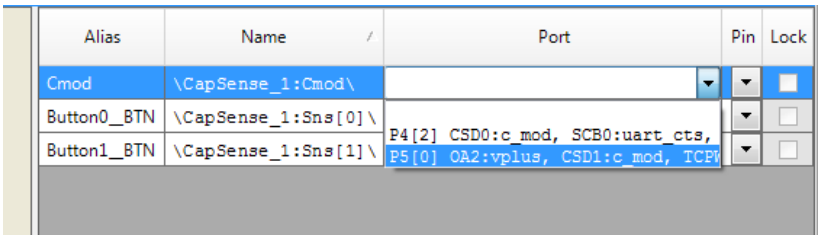
	CSD0	CSD1
C_{MOD}	P4.2	P5.0
C_{TANK}	P4.3	P5.1
CapSense Pin	Any pin except PORT5 pins	Any pin in PORT 5
Shield Pin	Any pin except PORT5 pins	Any pin in PORT 5
Max Number of CapSense Pins	47*	4*

* Max number of pins is specified assuming two pins are used for C_{MOD} and C_{TANK} 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



4. CapSense Design and Development Tools

Cypress provides a complete set of hardware and software tools to develop your CapSense application.

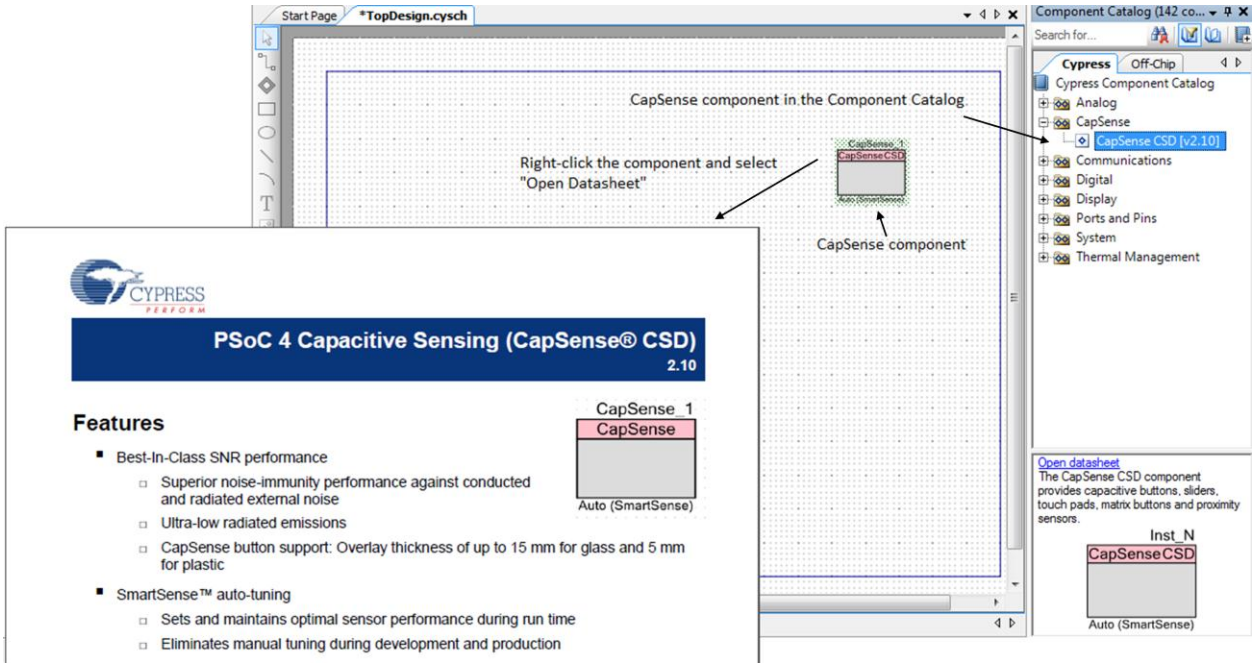
4.1 PSoC Creator

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, I^2C, 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



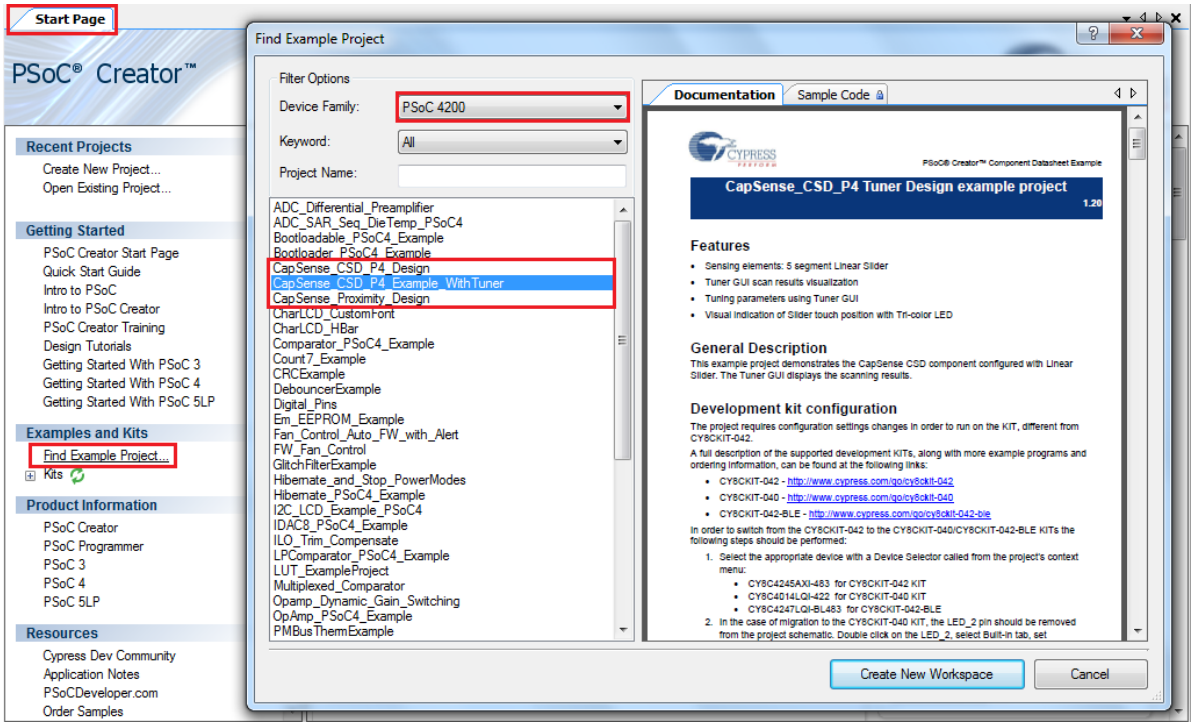
Each Component has an associated datasheet that explains details about the Component. To open the Component datasheet, right-click on the Component and select “Open Datasheet”.

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

Development Kit	Supported CapSense Features
PSoC 4 Pioneer Kit (CY8CKIT-042)	A 5-segment linear slider
PSoC 4000 Pioneer Kit (CY8CKIT-040)	A 6 × 5 touchpad and a wire proximity sensor
PSoC 4 BLE Bluetooth Low Energy Pioneer Kit (CY8CKIT-042-BLE)	A 5-segment linear slider and a wire proximity sensor
PSoC 4200-M Pioneer Kit (CY8CKIT-044)	A 5-element gesture detection and two-wire proximity sensors
PSoC 4200 Processor Module (CY8CKIT-038), with PSoC Development Kit (CY8CKIT-001)	A 5-segment linear slider and two buttons
CapSense Expansion Board Kit (CY8CKIT-031), to be used with CY8CKIT-038 and CY8CKIT-001	A 10-segment slider, 5 buttons and a 4 × 4 matrix button with LED indication.
MiniProg3 Program and Debug Kit (CY8CKIT-002)	CapSense performance tuning in CY8CKIT-038

Table 4-2 lists the Reference Design Kits that support evaluation of PRoC BLE CapSense.

Table 4-2. PRoC BLE CapSense Reference Design Kits

Reference Design Kit	Supported CapSense features
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

5. CapSense Performance Tuning

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 I^2C or UART communication with a host PC.

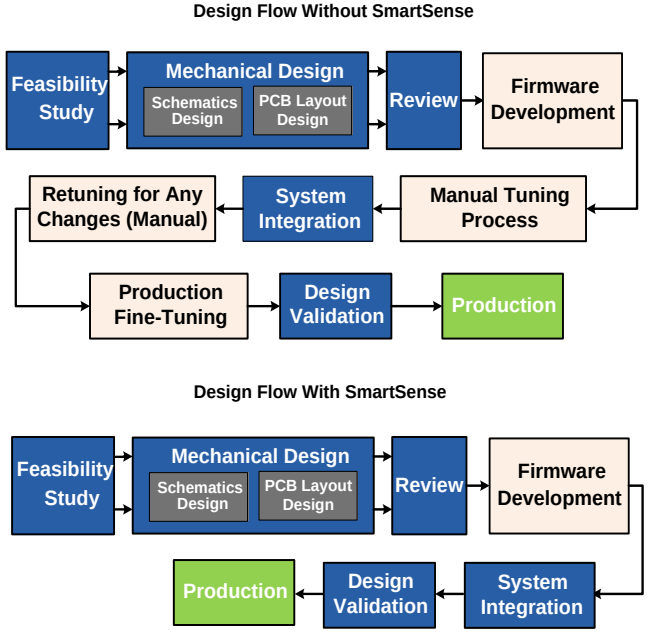
After tuning and finalizing all CapSense parameters, you can turn off the tuning. In this mode, all tuning parameters are stored in flash.

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



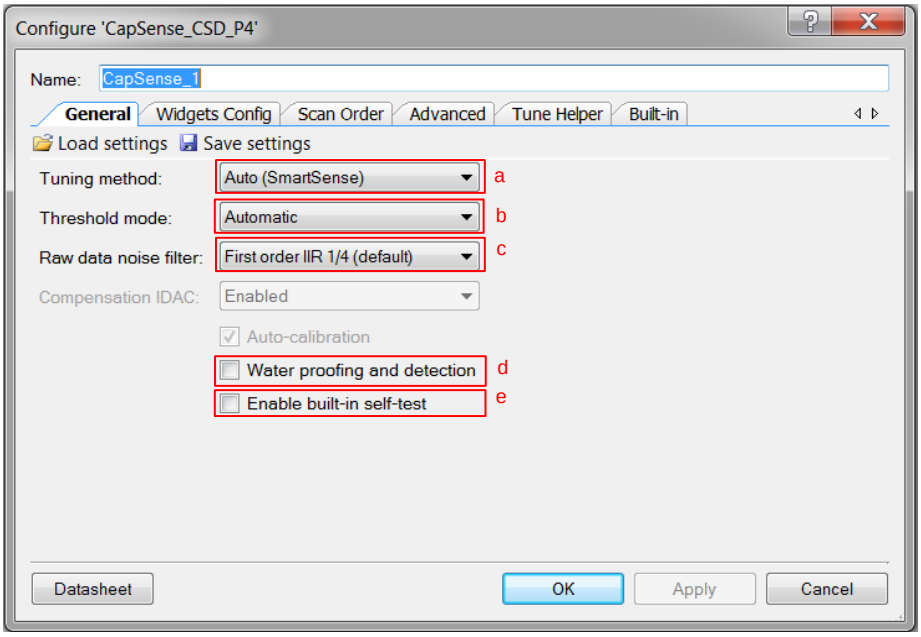
- Performance is independent of PCB variations: The parasitic capacitance of individual sensors can vary due to process variations in PCB manufacturing, or vendor-to-vendor variation in a multi sourced supply chain. If there is significant variation in parasitic capacitance C_P across product batches, 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”.

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.

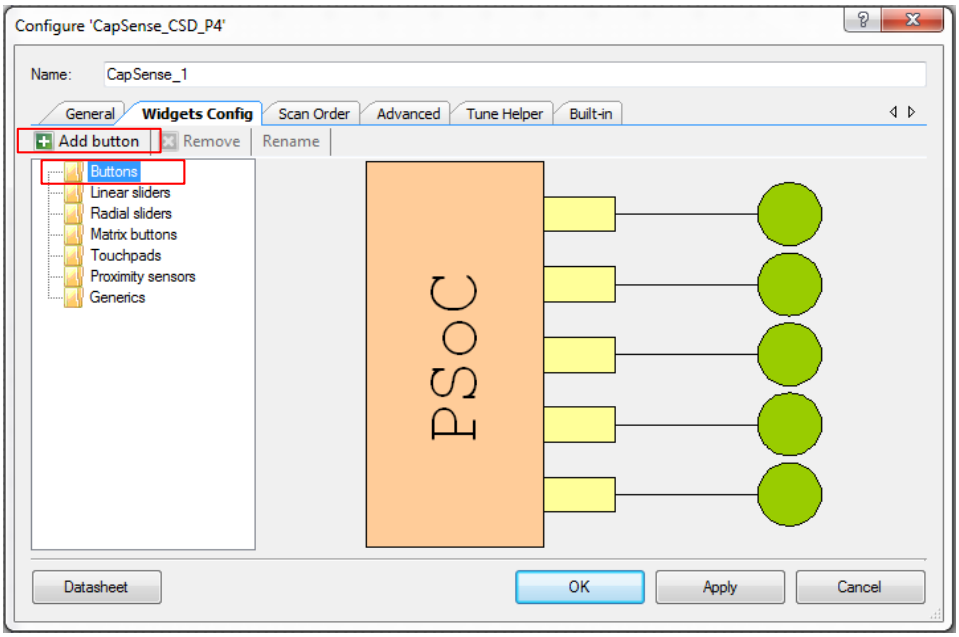
Table 5-1. Raw Data Noise Filters

Filter	Description	Mathematical Description	Application
Median	Non linear filter that takes the three most recent samples and computes the median value.	$y[i] = median\ (x[i],\ \ x[i-1],\ \ x[i-2])$	Eliminates noise spikes from motors and switching power supplies
Average	Finite impulse response filter (no feedback) with equally weighted coefficients. It takes the three most recent samples and computes their average.	$y[i] = \frac{1}{3} * (x[i] + x[i-1] + \ x[i-2])$	Eliminates periodic noise (e.g., from power supplies)
First order IIR 1/k	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	$y[i] = \frac{1}{k} * \{x[i] + \ (k-1)y[i-1]\}$	Eliminates high frequency noise.
Jitter	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 ±1, then the output is changed by ±1.	$y[i] = \begin{cases} x[i] - \ 1, & x[i] - \ y[i-1] > 1 \\ x[i] + \ 1, & x[i] - \ y[i-1] < 1 \\ y[i-1], & otherwise \end{cases}$	Noise due to thick overlay. Especially useful for slider centroid data.

5.1.2 Widget Configuration

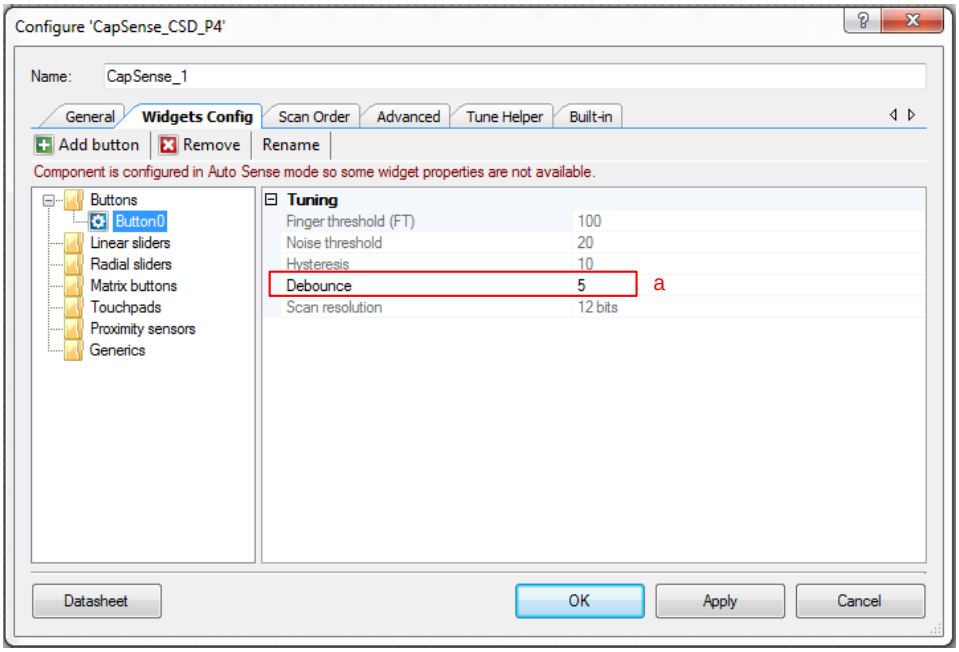
The “Widgets Config” tab allows you to add and configure CapSense widgets, as Figure 5-3 shows. See CapSense Widgets for details on each widget type. To add a widget, select the type of widget and click “**Add**”.

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.

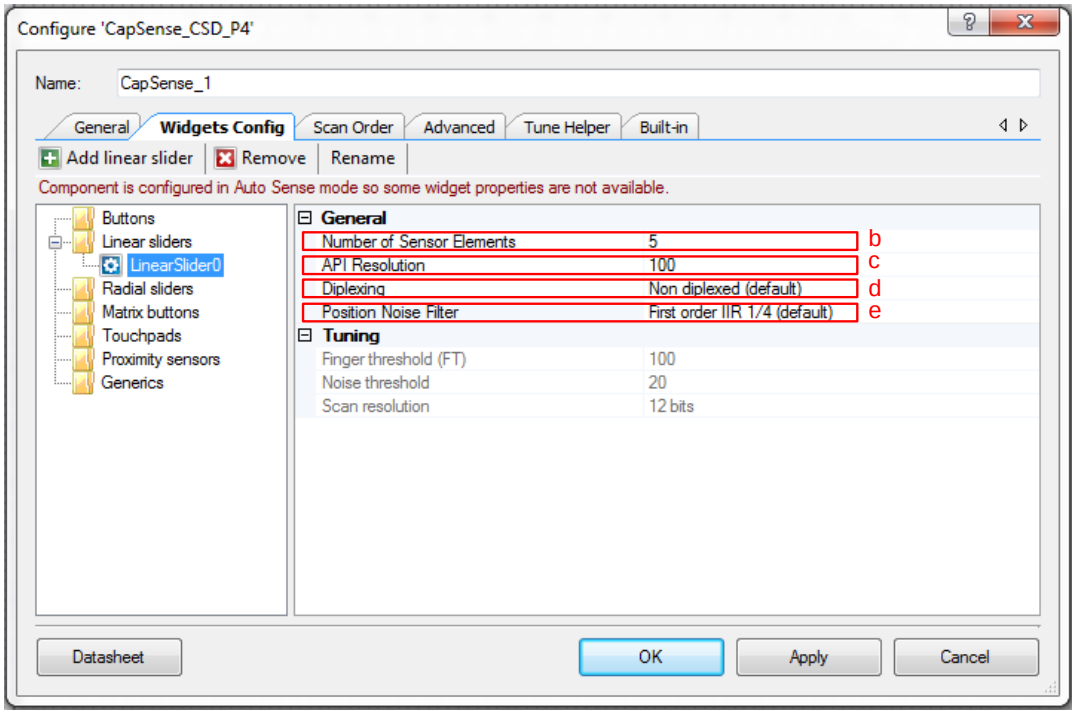
c. API resolution: (see Figure 5-5; available for Linear Sliders, Radial Sliders and Touchpads). This parameter defines the number of discrete finger positions that a slider or a touchpad must resolve. This parameter is different from the resolution of the Sigma Delta converter. A good ratio of API resolution 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.

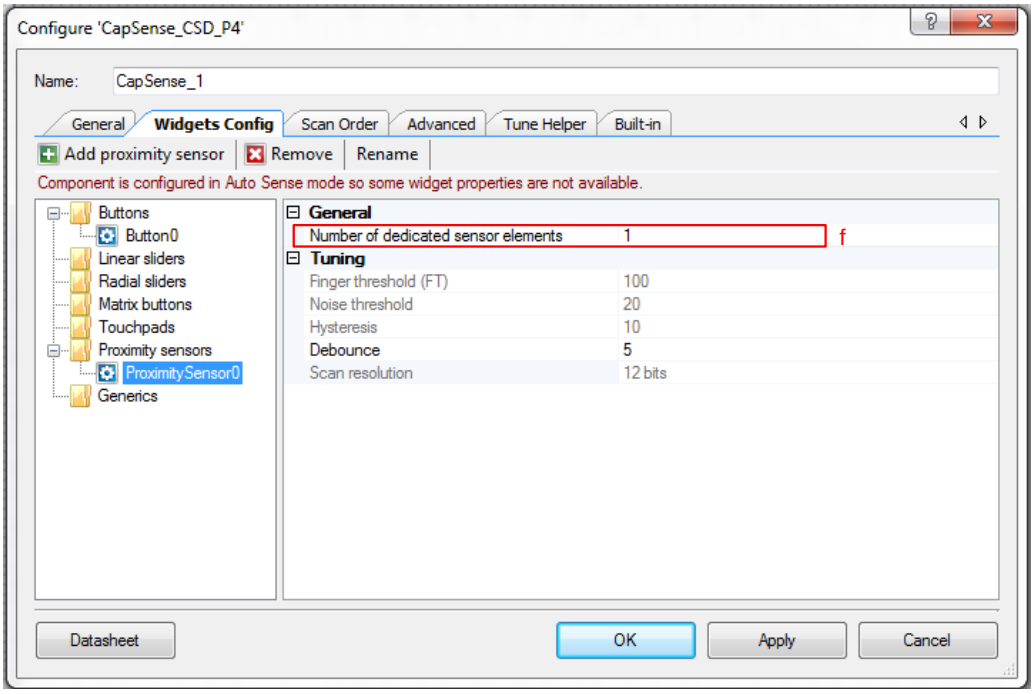
e. Position Noise filter: (see Figure 5-5; available for Linear Sliders, Radial Sliders and Touchpads). This parameter selects the type of firmware noise filter used to remove noise from the calculated finger position. The available filters and their applications are explained in Table 5-1 on page 37.

Figure 5-5. Configuration of a Slider



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.

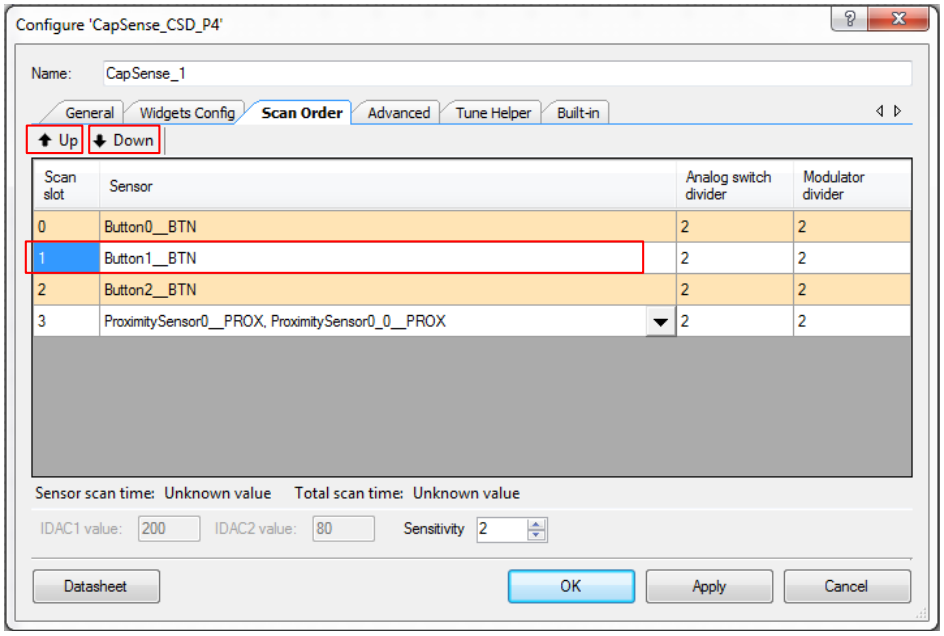
Figure 5-6. Configuration of a Proximity Sensor



5.1.3 Scan Order

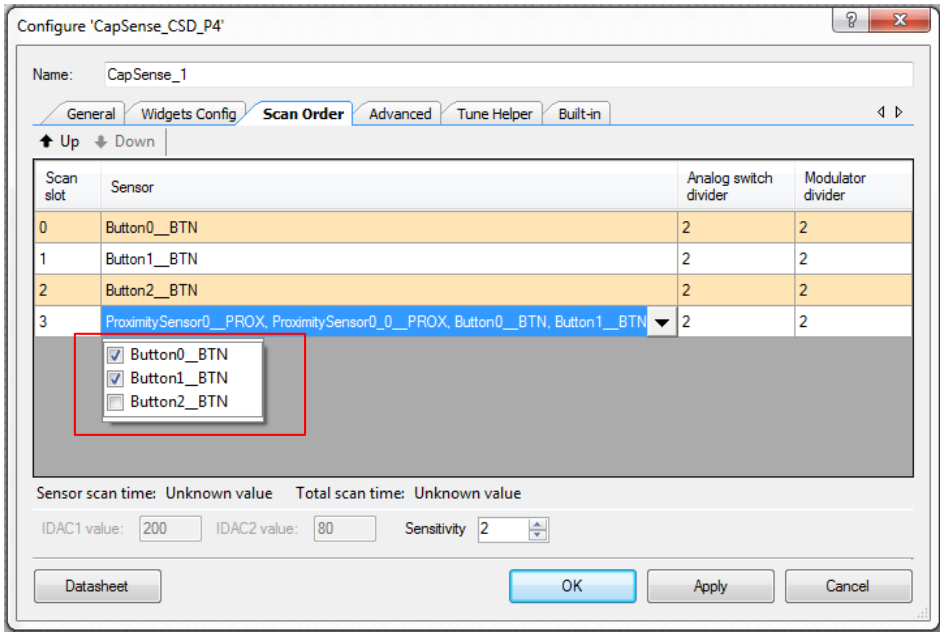
Generally, CapSense performance is not impacted by the order in which the sensors are scanned. However, if you want to change scan order, you can use this tab, as Figure 5-7 shows. To change the scan order, select a sensor and click “Up” or “Down”.

Figure 5-7. Scan Order



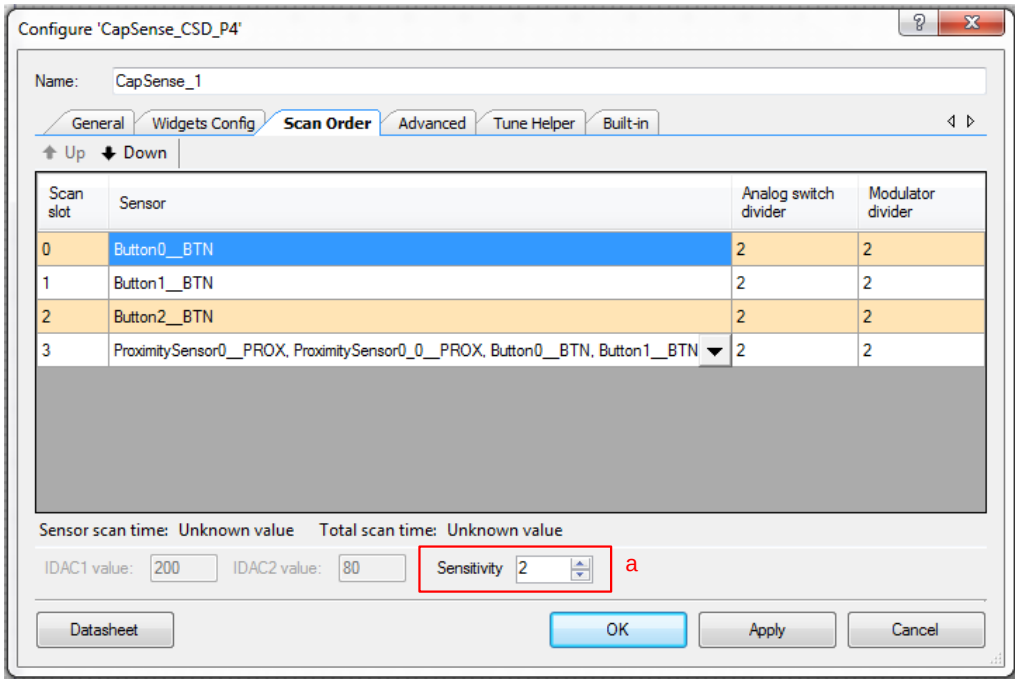
You can combine other sensors into a proximity sensor by using the drop-down menu on the Scan Order tab, as Figure 5-8 shows. Combining other sensors with the proximity sensor does not affect their operation, however, the resulting proximity sensor may have a high C_P and hence a high scan time. Therefore, 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



Click on a sensor to change its sensitivity, as Figure 5-9 shows.

Figure 5-9. Sensitivity

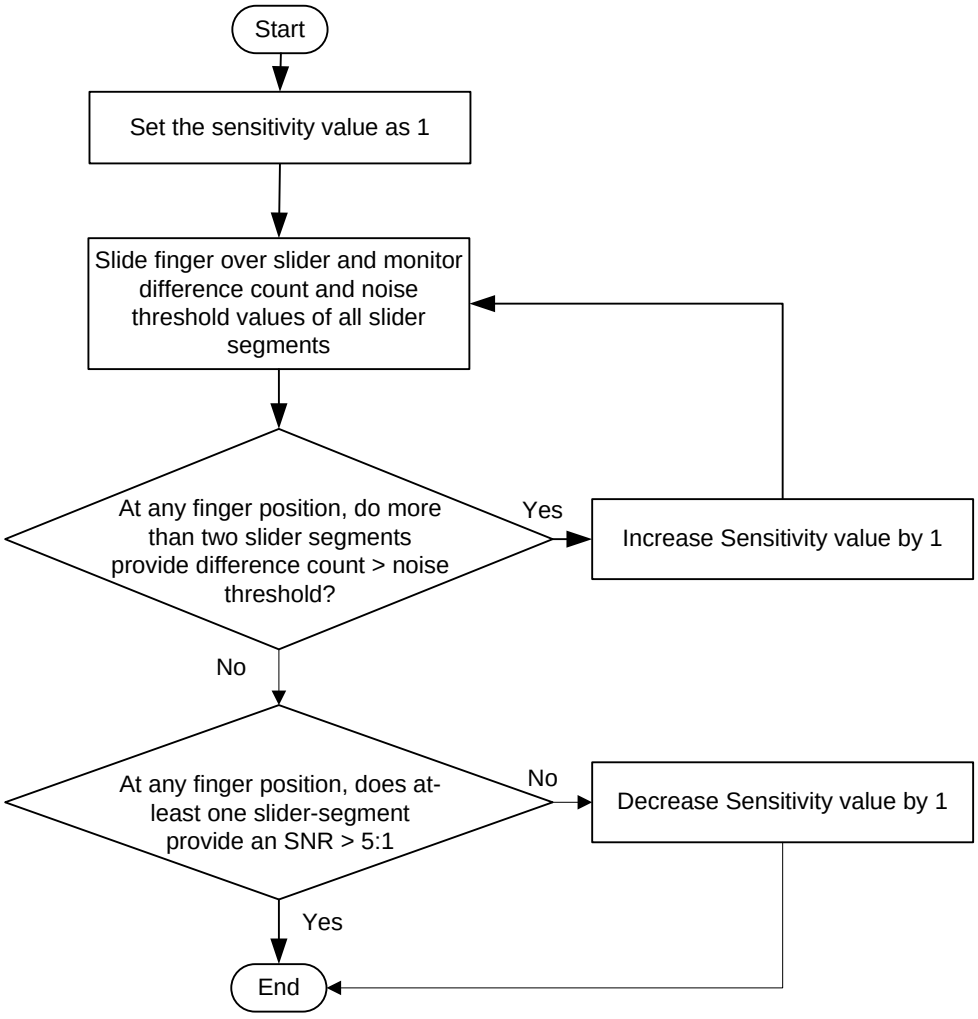


a. Sensitivity: This parameter sets the expected value of value of finger capacitance C_F. See CapSense Fundamentals for details on C_F. SmartSense multiplies the sensitivity setting by 0.1 pF to get C_F. For example, a sensitivity value of 1 represents C_F = 0.1 pF and a setting of 4 represents C_F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

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

- 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.

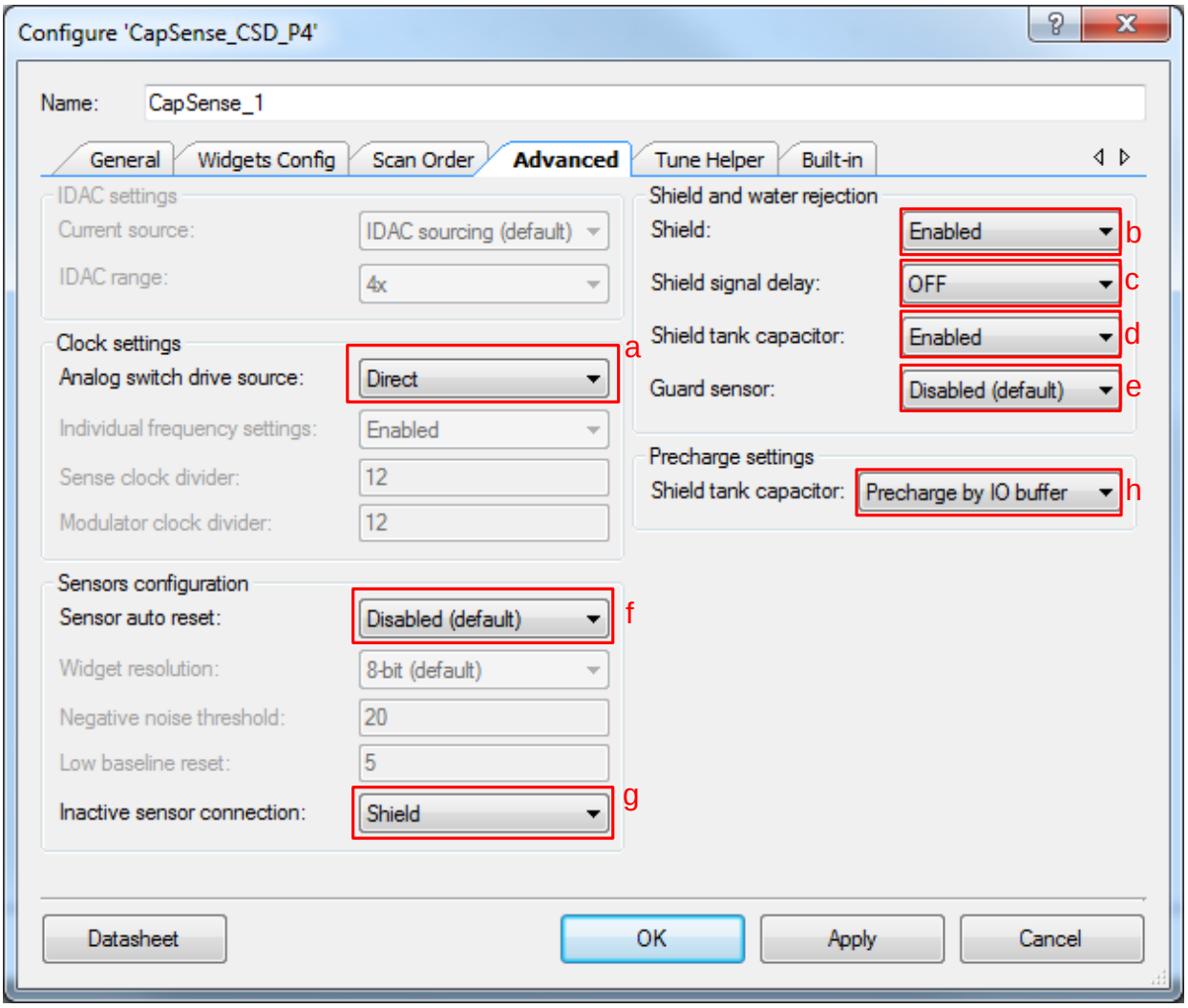
Figure 5-10. Setting sensitivity value for Sliders



5.1.3.1 Advanced Settings

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

Figure 5-11. Advanced Settings



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.

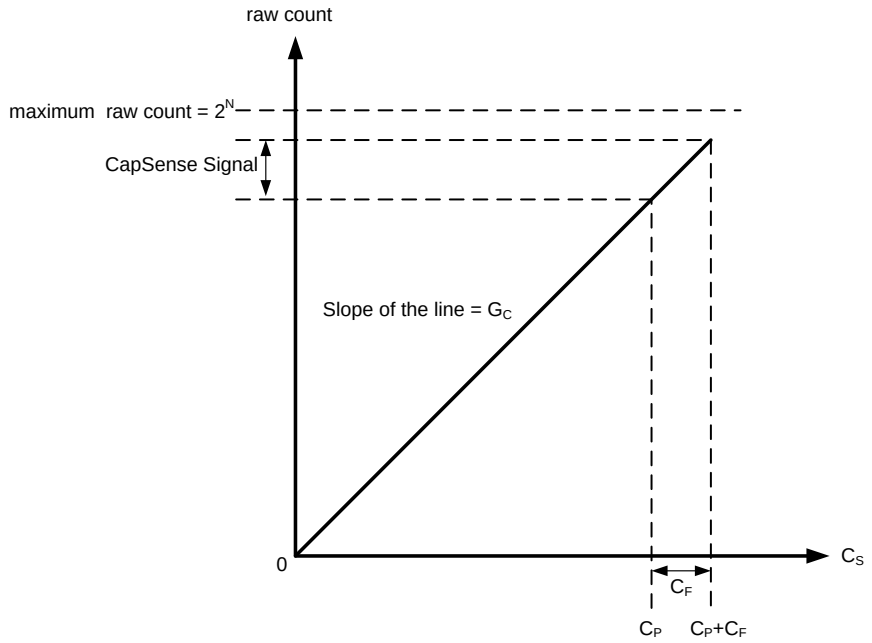
5.2.1.1 Conversion Gain and CapSense Signal

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

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

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.

Figure 5-12. 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. Signal Values for Different Conversion Gains

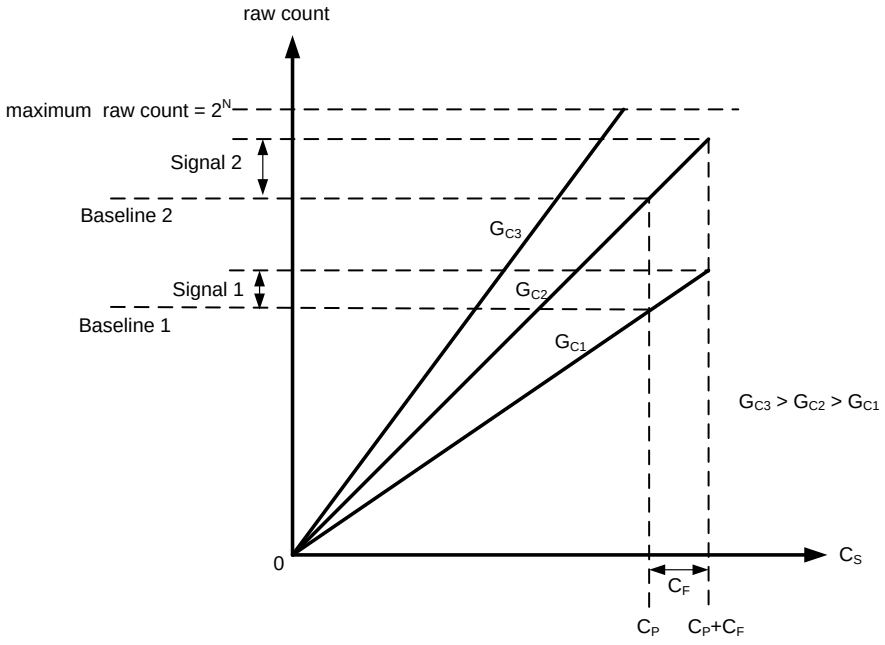
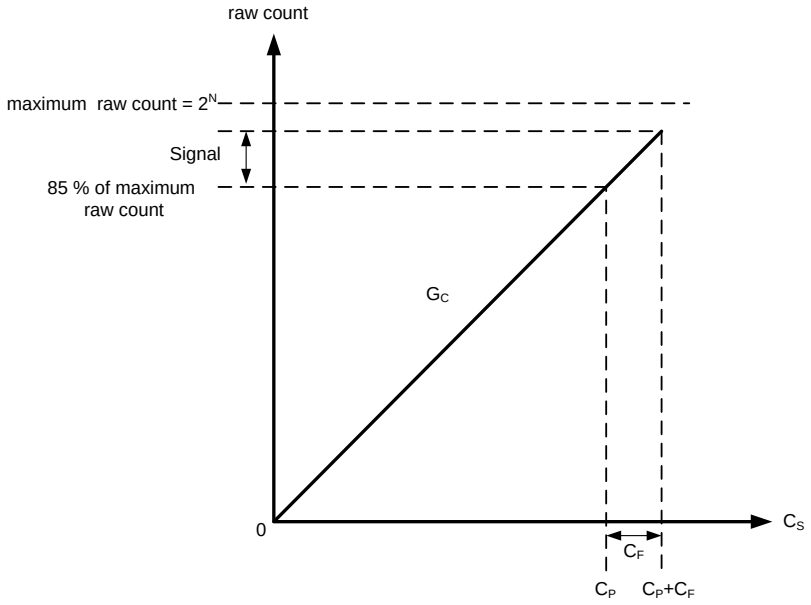


Figure 5-13 shows three plots corresponding to three conversion gain values G_{C3}, G_{C2}, and G_{C1}. An increase in the conversion gain results in higher signal value. However, this increase in the conversion gain also moves the raw count corresponding to C_P towards the maximum value of raw count (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.

Figure 5-14. Recommended Tuning



The equation for raw count, in the dual IDAC mode is

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

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. Dual IDAC Mode

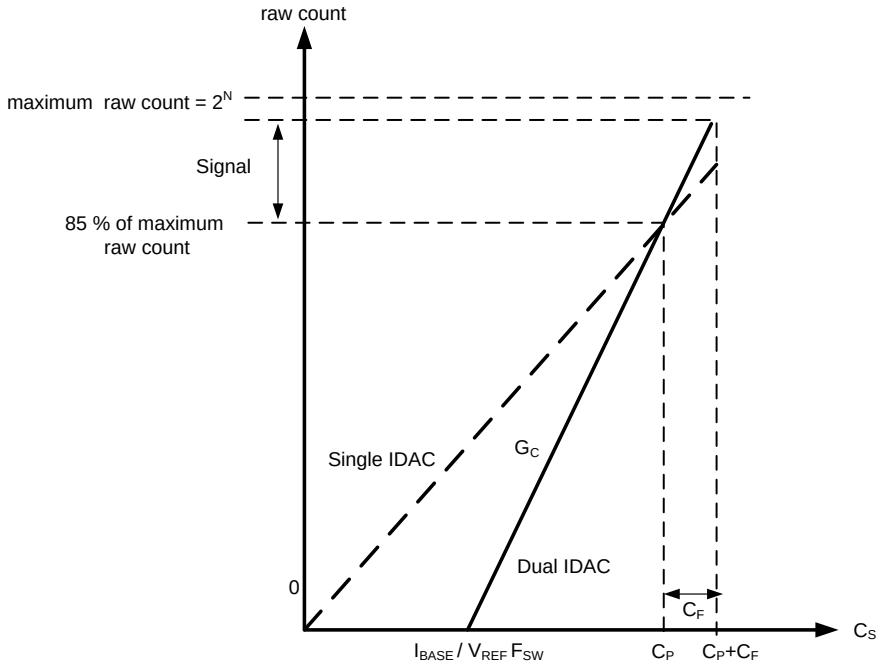


Figure 5-15 shows that dual IDAC mode yields higher signal than single IDAC mode.

Increasing the value of the compensation IDAC increases the C_S axis intercept:

$$\left(\frac{I_{COMP}}{V_{REF} \quad F_{SW}}\right)$$

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.

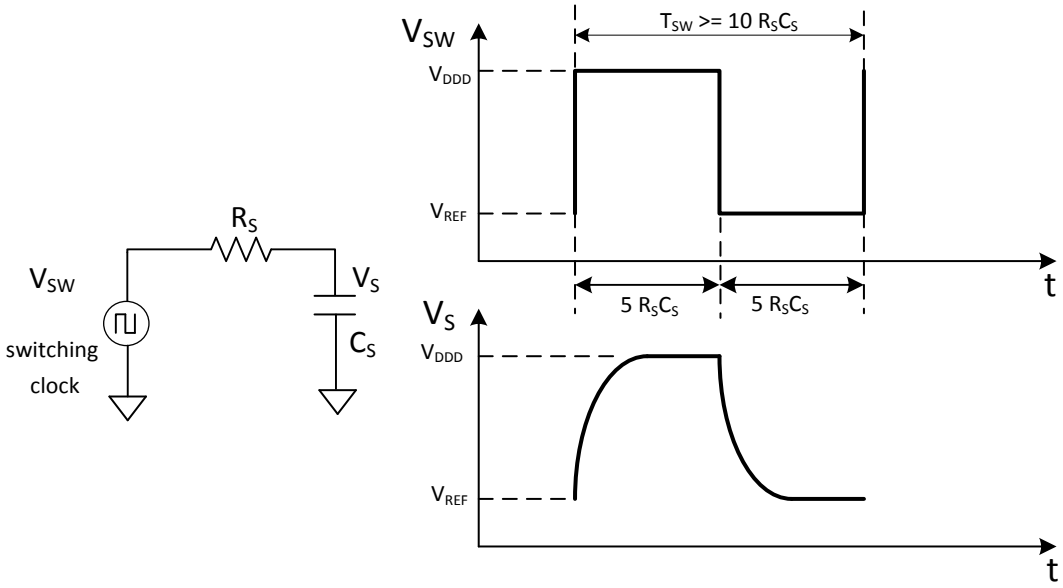
Note that this increase in signal may not proportionally increase the SNR because of the variations in noise counts when the compensation IDAC is used. Thus, the recommended method to tune a sensor for best SNR using dual IDAC is as explained in Manual Tuning Process on page 61.

5.2.1.2 Switching Clock Selection

For a given resolution N, you can vary both F_{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_{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.

Figure 5-16. Equivalent Circuit and Waveforms



R_S is the sum of the GPIO resistance and the external series resistance. C_S is the maximum capacitance of the sensor. You should select a switching frequency that is low enough to allow the sensor capacitance to fully charge and discharge. The rule of thumb is to allow a period of $5R_SC_S$ for charging and discharging cycles. The equations for minimum time period and maximum frequency are:

$$T_{SW}(\text{minimum}) = 10R_SC_S \qquad (6-4)$$

$$F_{SW}(\text{maximum}) = \frac{1}{10R_SC_S} \qquad (6-5)$$

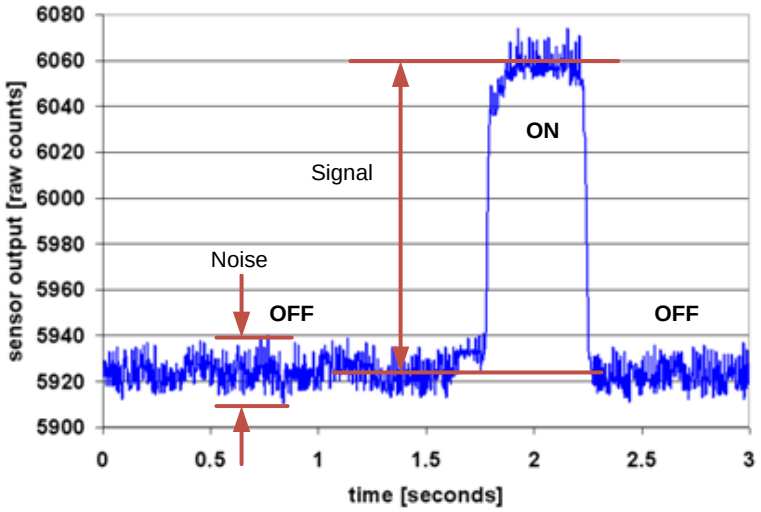
The typical value of the GPIO resistance is 500 Ω and the recommended external resistance is 560 Ω (see Series Resistors on CapSense Pins for details). Therefore, take the value of R_S as 1.06 kΩ when calculating the maximum switching frequency.

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



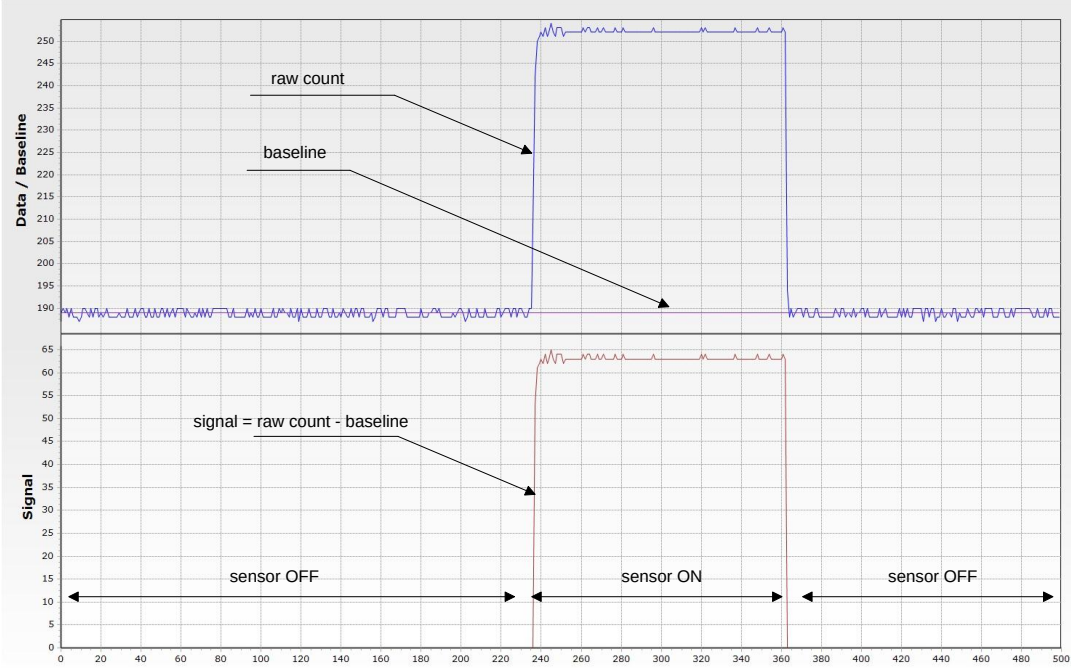
In this example, the average level of raw count in the absence of a touch is 5925 counts. When a finger is placed on the sensor, the average raw count increases to 6060 counts, therefore the signal is 6060 – 5925 = 135 counts. The minimum value of raw count in the OFF state is 5912 and the maximum value 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 for 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

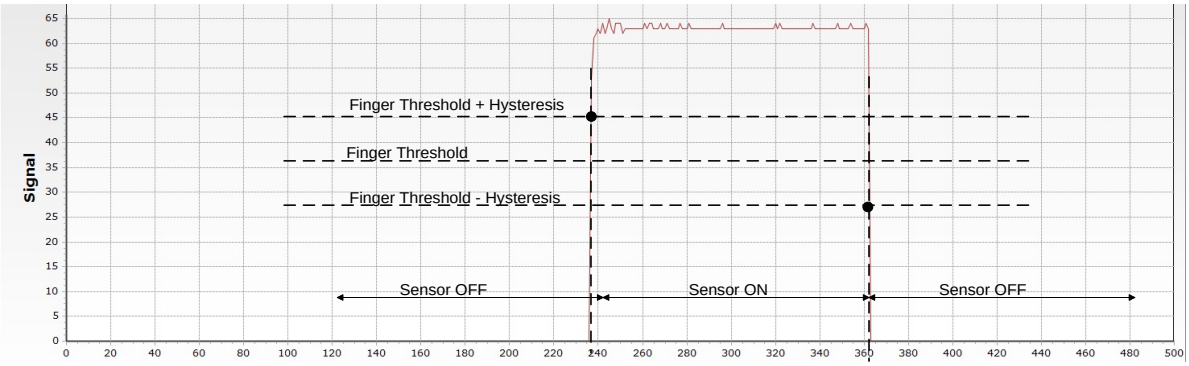


5.2.1.5 Tuning Parameters

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.

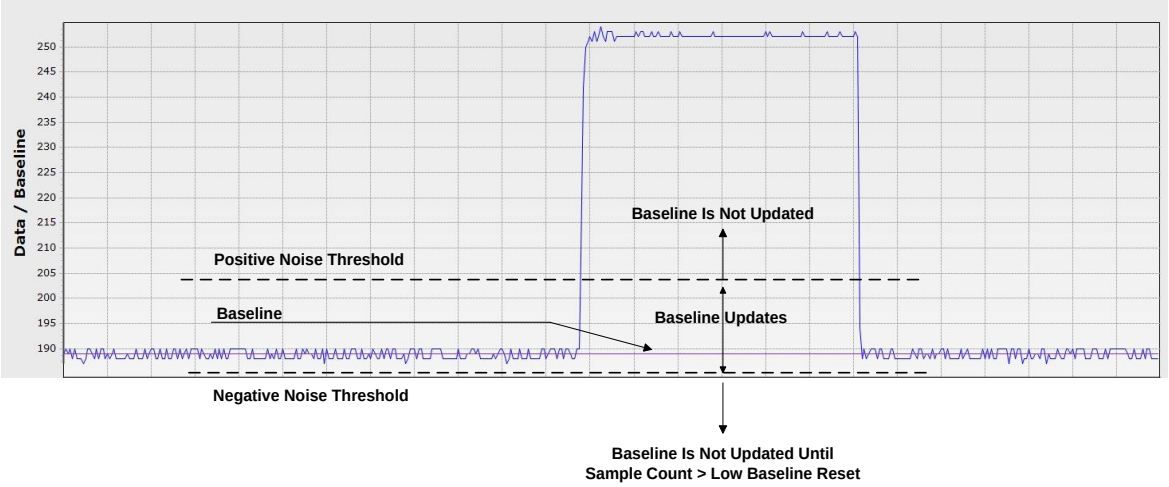
- Finger Threshold: The finger threshold parameter controls the sensitivity of a sensor to finger touches. It is used along with the hysteresis parameter to determine the sensor state, as Equation 6-6 shows.
  - $$\text{Sensor State} = \begin{cases} \text{ON} & \text{if (Signal} \geq \text{Finger Threshold} + \text{Hysteresis)} \\ \text{OFF} & \text{if (Signal} \leq \text{Finger Threshold} - \text{Hysteresis)} \end{cases} \quad (6-6)$$
- 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



- 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



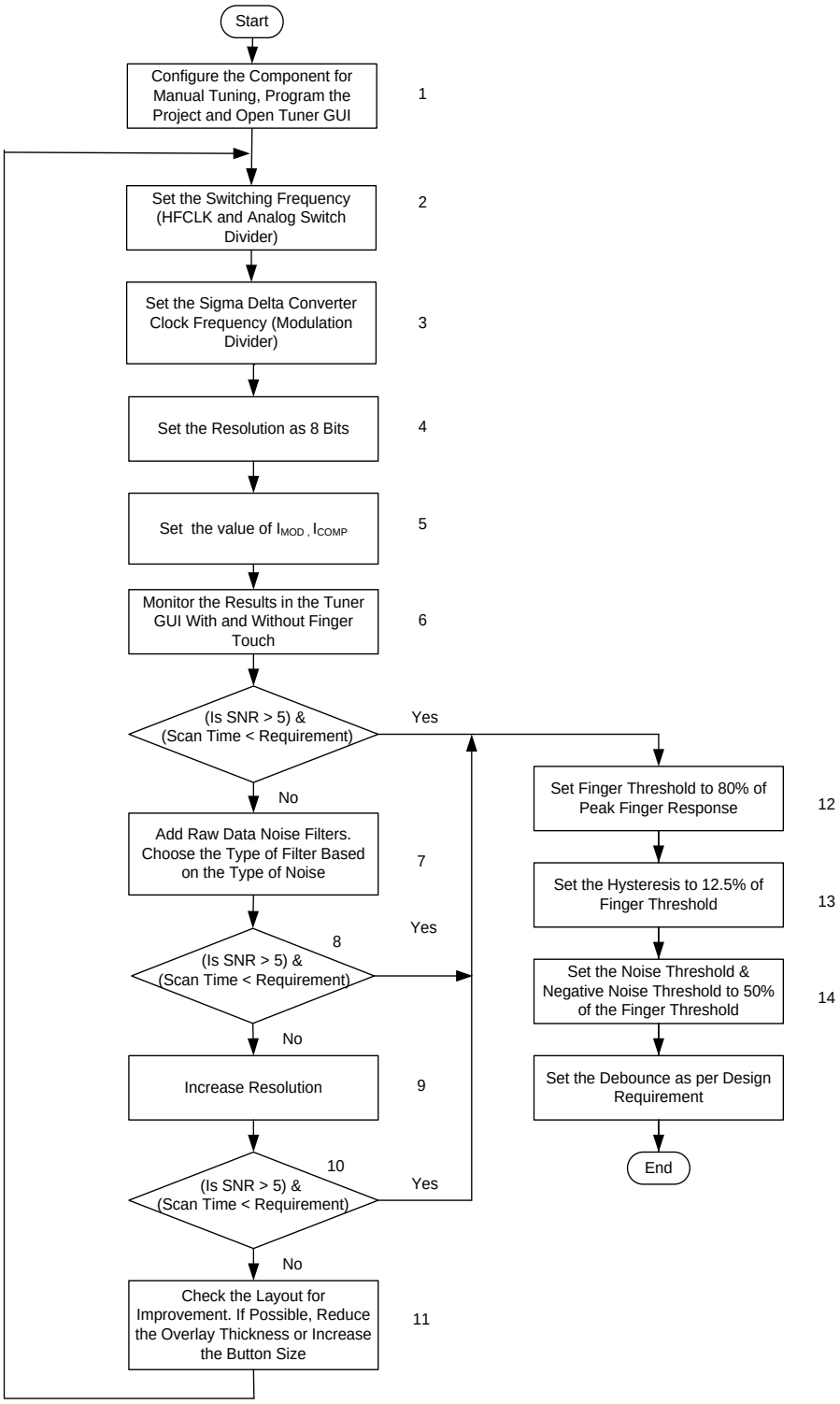
- Low Baseline Reset parameter: This parameter is used together with the negative noise threshold parameter. It counts the number of abnormally low raw counts required to reset the baseline. It is used to reset the baseline if the finger is placed on the sensor during device startup and later removed.
- 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.

$$\text{Sensor State} = \begin{cases} \text{ON} & \text{if (Signal } \geq \text{Finger Threshold} + \text{Hysteresis) for scans} \geq \text{debounce} \\ \text{OFF} & \text{if (Signal} \leq \text{Finger Threshold} - \text{Hysteresis)} \\ \text{OFF} & \text{if (Signal} \geq \text{Finger Threshold} + \text{Hysteresis) for scans} < \textit{debounce} \end{cases} \quad (6-7)$$

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.

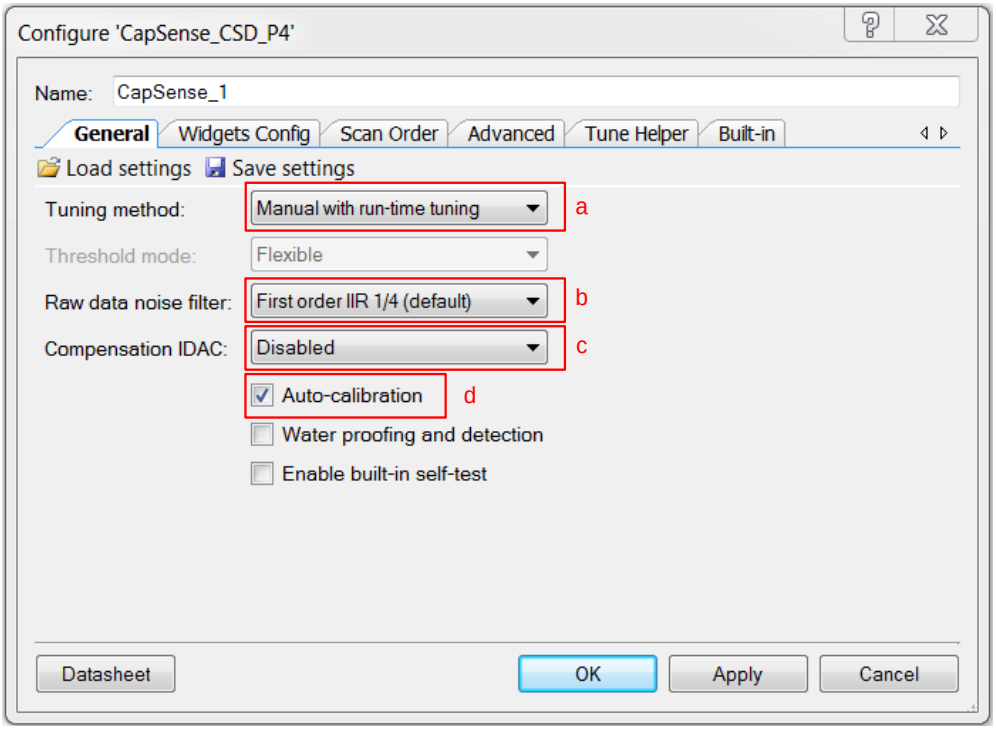
Figure 5-21. Manual Tuning Process



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



   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.

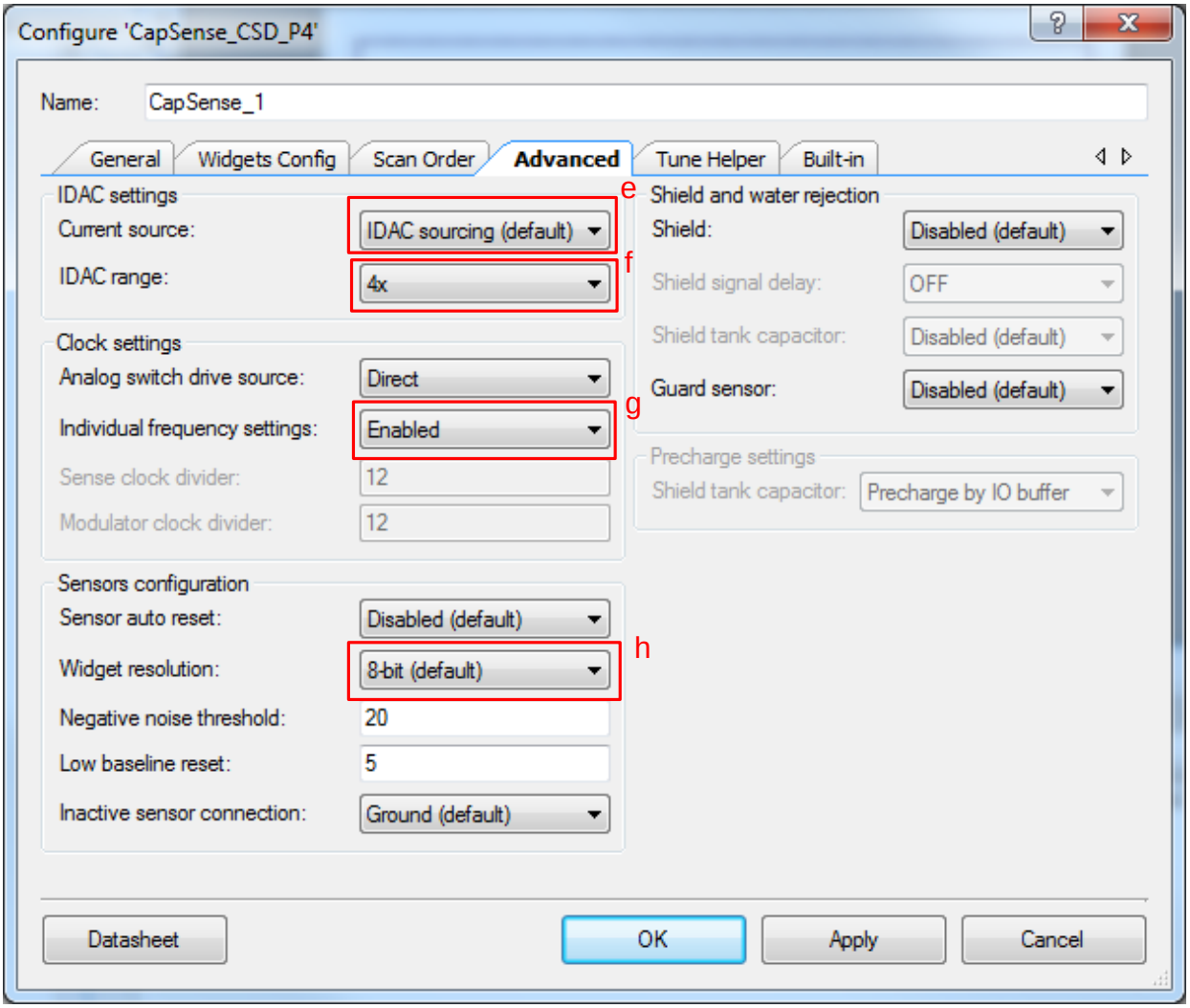
      Manual tuning in the single IDAC mode is easier than the dual IDAC mode. However, the dual IDAC mode gives higher signal values. See Fundamentals of Manual Tuning for more details.

   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



e. 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.

f. 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.

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

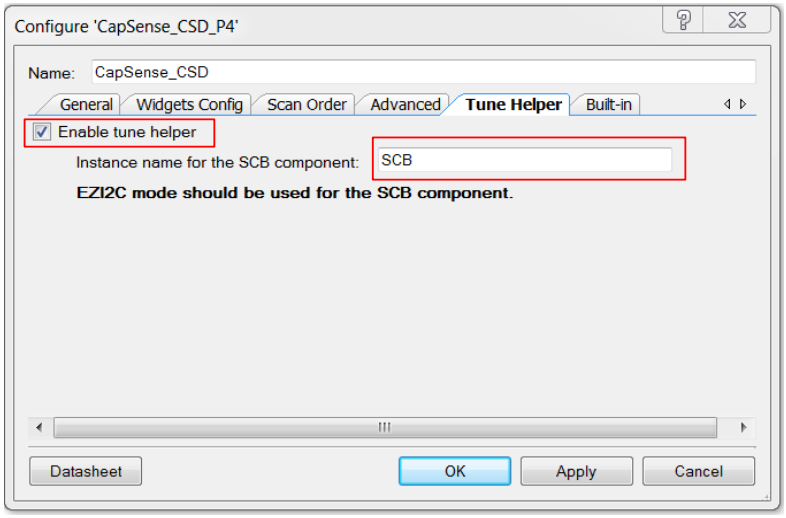
h. Widget resolution: This parameter selects between 8-bit or 16-bit variables for signal count. In most cases, you should use the 8-bit widget resolution. If you have more signal than an 8-bit variable can handle, it means that you are using a higher resolution than required, and the resolution can 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.

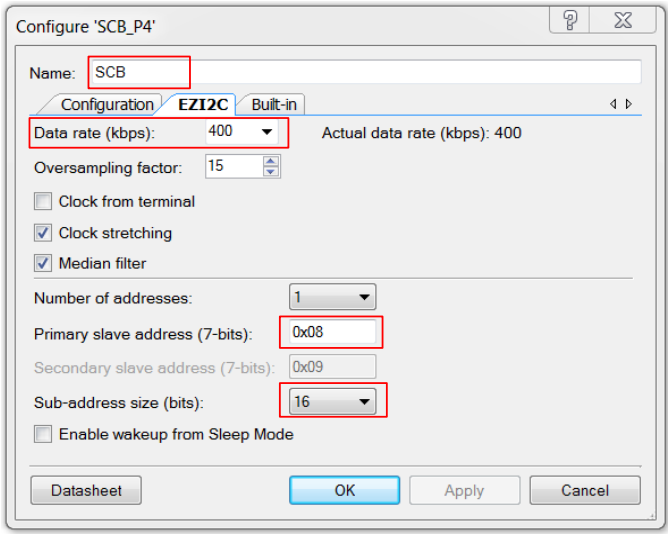
i. Go to the “Tune Helper” tab, enable the tune helper, and give an instance name for the SCB Component, as Figure 5-24 shows.

Figure 5-24. Enabling Tune Helper



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



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();
      }
}
```

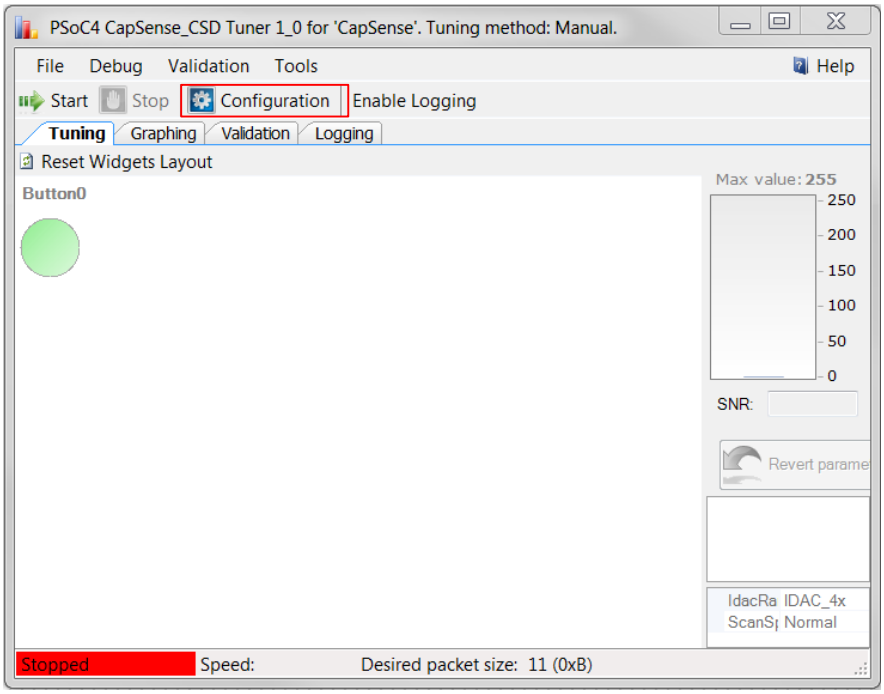
After finishing the manual tuning process, you can delete the function calls from your code. Note that they are not needed for SmartSense.

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”.

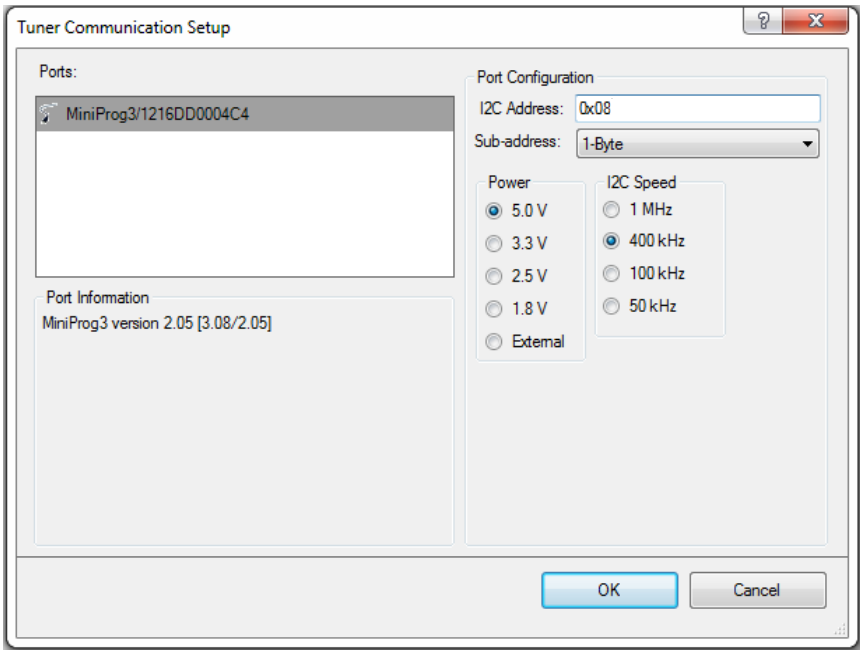
vi. Open the Tuner Communication Setup window by clicking on the configuration button in the tuner window, as Figure 5-26 shows.

Figure 5-26. Tuner GUI



vii. 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

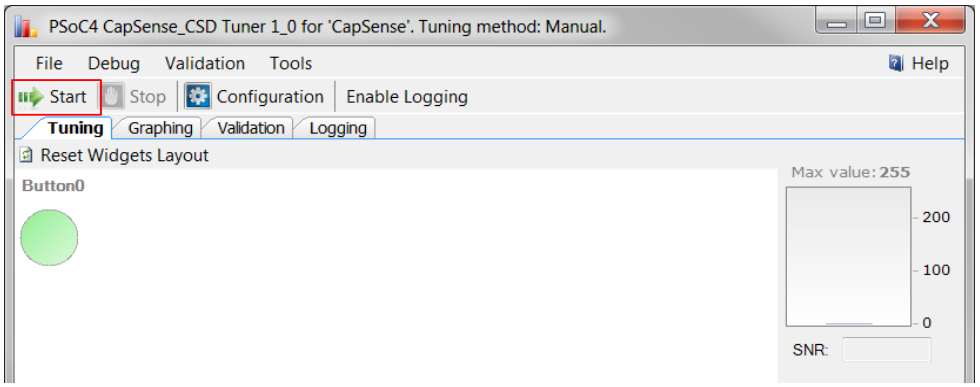


Note The voltage settings in this window must match the voltage settings on the board.

viii. Click OK.

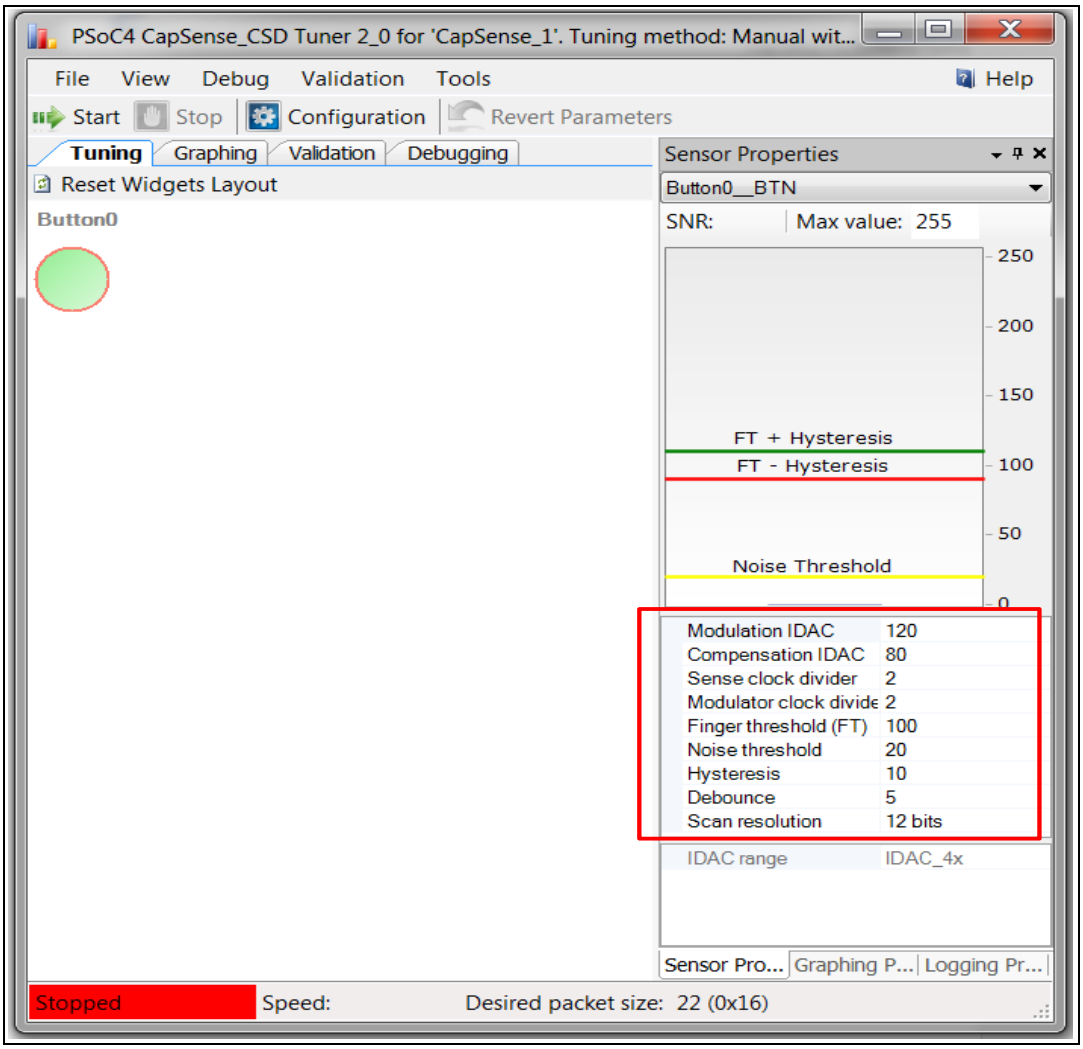
ix. Click the “Start” button in the tuner window to initiate I²C communication, as Figure 5-28 shows.

Figure 5-28. Starting Tuner Communication



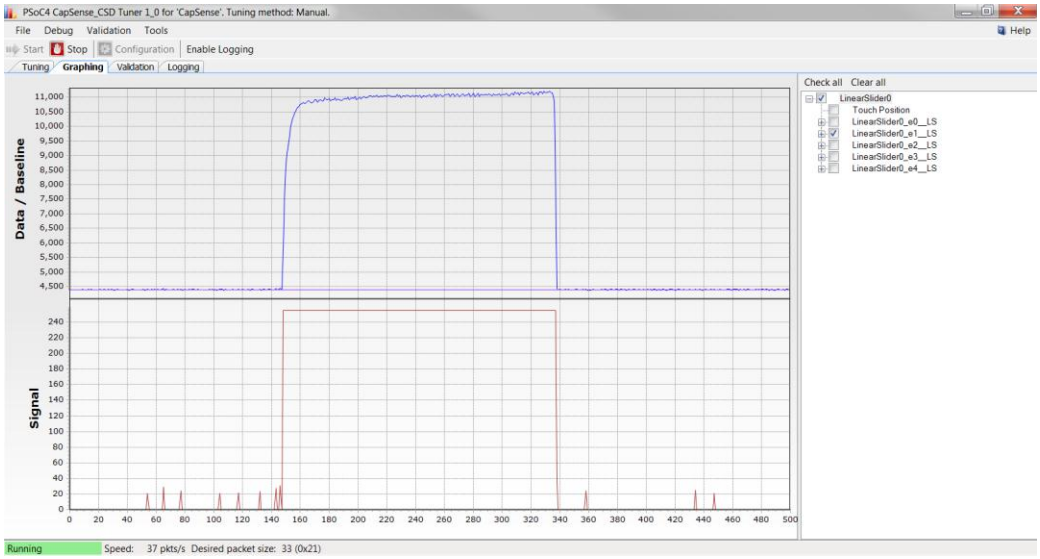
x. You can set the parameters for each sensor at the lower right corner of this window, as Figure 5-29 shows. Clicking OK copies the parameters from the tuner GUI to CapSense Component.

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 their 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.

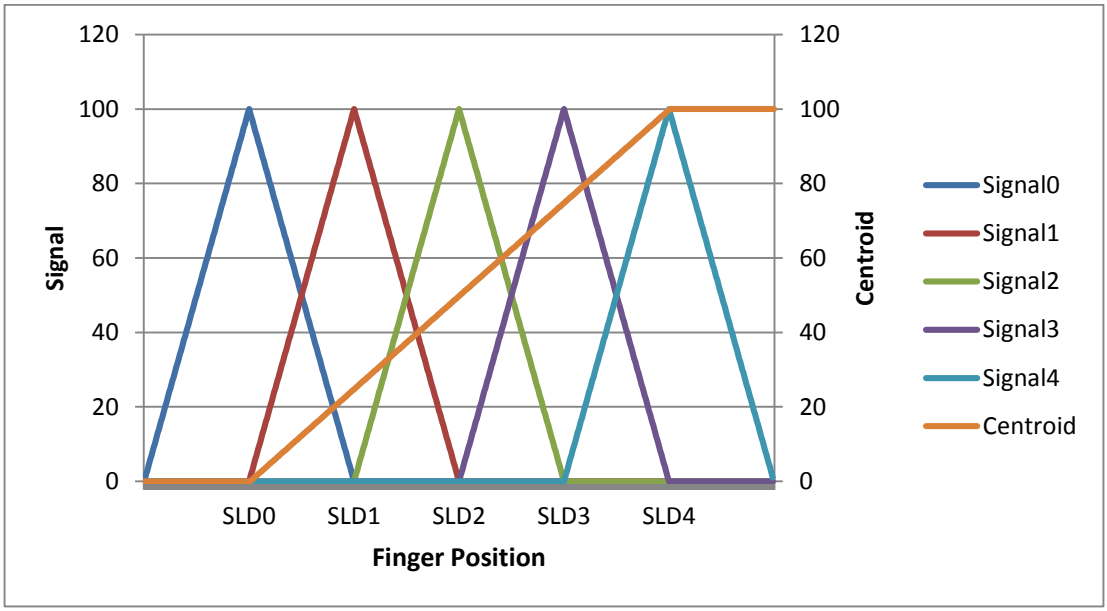
To validate your design, click the “Validation” tab in the tuner GUI and follow the onscreen instructions. If you see any false touch detections, increase the debounce value of the sensors (see Debounce). Note that increasing the debounce also increases scan time. If the resulting scan time is not within 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 C_P of all the segments in a slider remains same. If all the slider segments have the same C_P, 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.

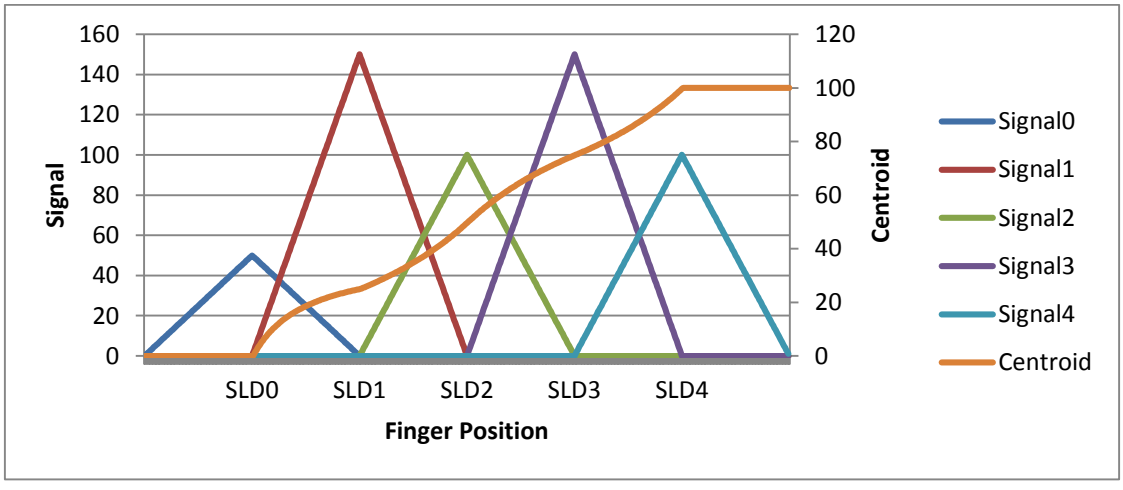
Figure 5-31. Response of Centroid Versus Finger Location when Signals of All Slider Elements Are Equal



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



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 Cp 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

Threshold Parameter	Recommendation
Finger Threshold	80% of the Signal
Noise Threshold	40% of the Signal
Negative Noise Threshold	40% of the Signal
Low-Baseline Reset	Set to 30

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:

```
/* 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.

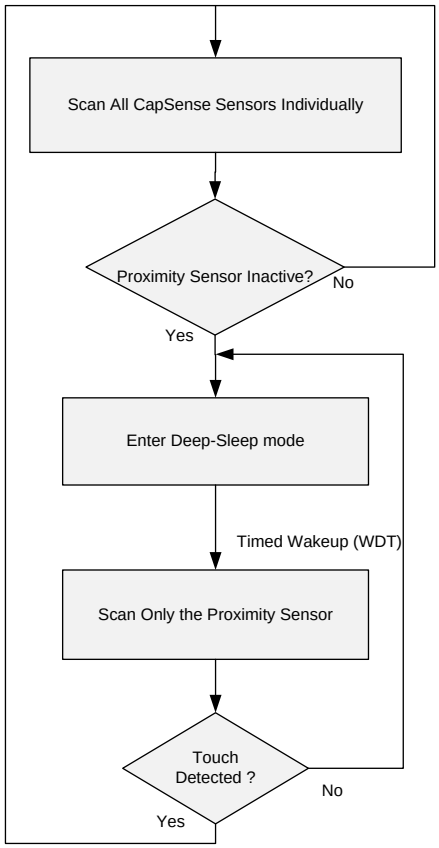
The CPU intervention is not required between the start and the end of a CapSense scan. If the firmware does not have any additional task than waiting for the scan to finish, you can put the device to Sleep mode after initiating a scan to save power. When the CSD hardware completes the scan, it generates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

Figure 6-1. Low Power CapSense Design



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 I^2C are required.

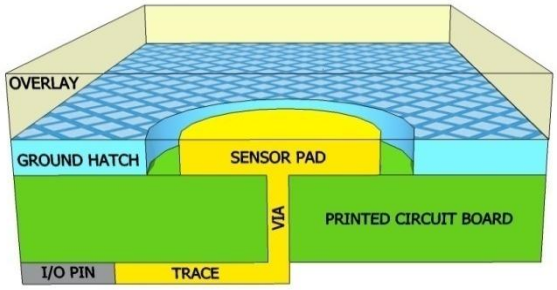
Note: In PSoC 4000 devices, it is not recommended to enter Sleep mode if a CapSense scan is in progress.

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.

Figure 6-2. CapSense Sensor Construction

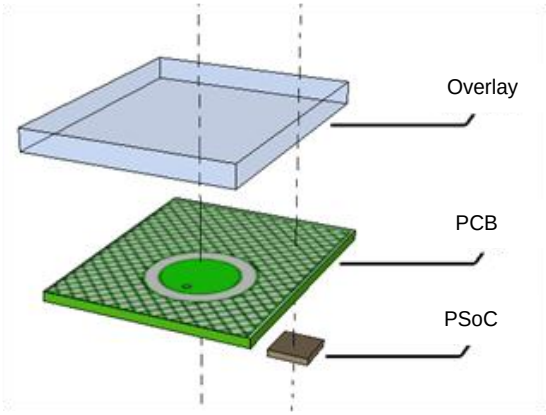


The copper pads etched on the surface of the PCB act as CapSense sensors. A nonconductive overlay serves as the touch surface. The overlay also protects the sensor from the environment and prevents direct finger contact. A ground hatch surrounding the sensor pad isolates the sensor from other sensors 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.

Figure 6-3. 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

Material	ε_r
Air	1.0
Formica	4.6 – 4.9
Glass (Standard)	7.6 – 8.0
Glass (Ceramic)	6.0
PET Film (Mylar®)	3.2
Polycarbonate (Lexan®)	2.9 – 3.0
Acrylic (Plexiglass®)	2.8
ABS	2.4 – 4.1
Wood Table and Desktop	1.2 – 2.5
Gypsum (Drywall)	2.5 – 6.0

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

Widget	Maximum thickness (mm)
Button	5
Slider	5[1]
Touchpad	0.5

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, C_P

The main components of C_P are trace capacitance and sensor capacitance. See CapSense Fundamentals for details. The relationship between C_P and the PCB layout features is not simple. C_P 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 C_P. 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 C_P is to increase the gap between the sensor pad and ground hatch. However, widening the gap also decreases noise immunity. You can also reduce C_P by driving the hatch with a shield signal.

See for details.

6.3.2 Board Layers

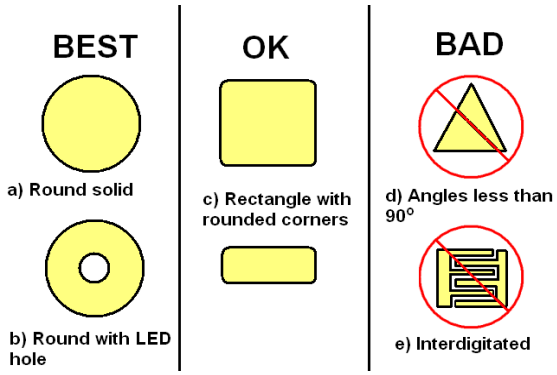
Most applications use a two-layer board with the sensor pads and the hatched ground planes on the top side and all other components on the bottom side. PCBs that are more complex use four layers. FR4-based PCB designs perform well with board thickness ranging from 0.020 inches (0.5 mm) to 0.063 inches (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 90º) since they concentrate electric fields. Figure 6-4 shows recommended button shapes.

Figure 6-4. Recommended Button Shapes

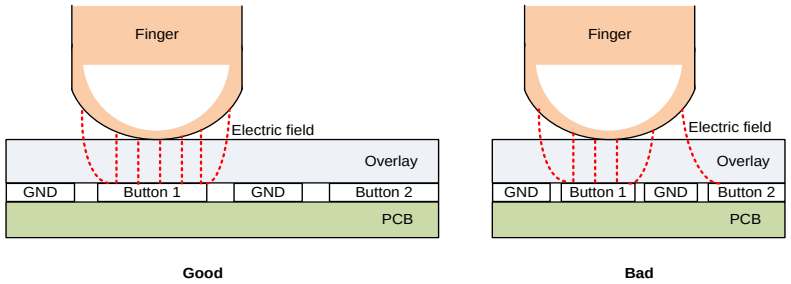


Button diameter should be 5 mm to 15 mm, with 10 mm suitable for most applications. A larger diameter is appropriate for thicker overlays.

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.

Figure 6-5. Spacing Between Buttons



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.

Figure 6-6. Typical Linear Slider Pattern

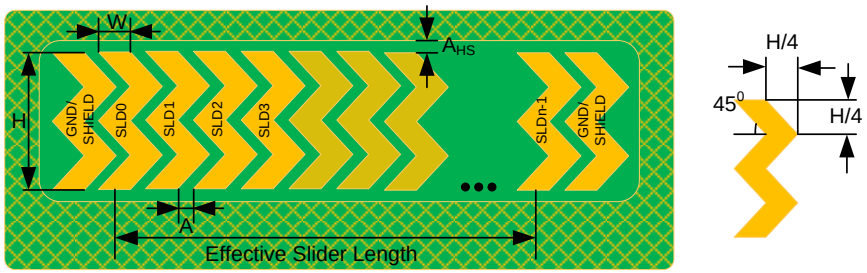


Table 6-3. Linear Slider Dimensions

Parameter	Acrylic Overlay Thickness	Minimum	Maximum	Recommended
Width of the segment (W)	1 mm	2 mm	-	8 mm[1]
	3 mm	4 mm	-	
	4 mm	6 mm	-	
Height of the segment (H)	-	7 mm[2]	15 mm	12 mm
Air gap between segments (A)	-	0.5 mm	2 mm	0.5 mm
Air gap between the hatch and the slider (A_{HS})	-	0.5 mm	2 mm	Equal to overlay thickness

[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.

Figure 6-7. Ideal Slider Segment Signals and Centroid Response

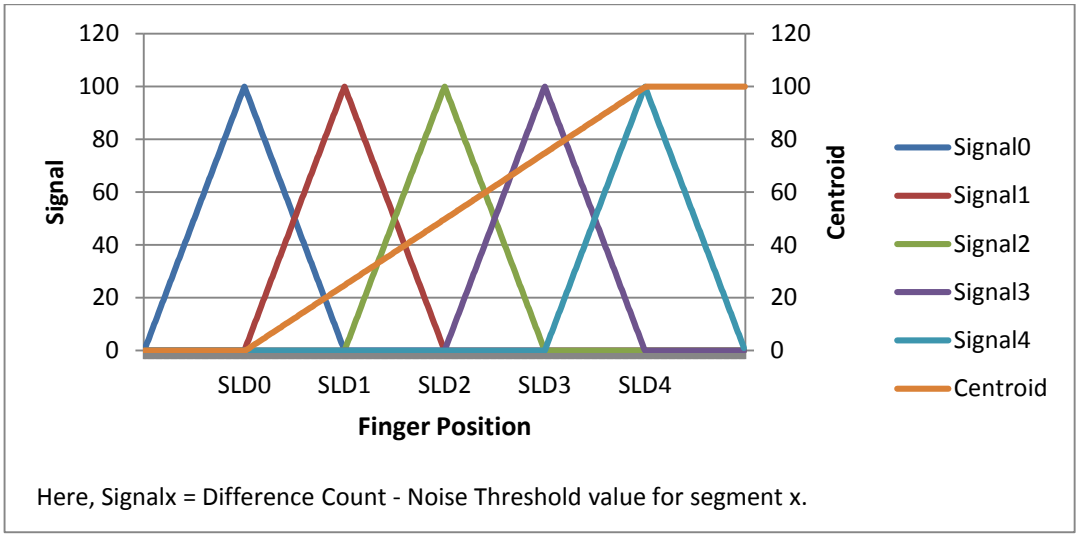
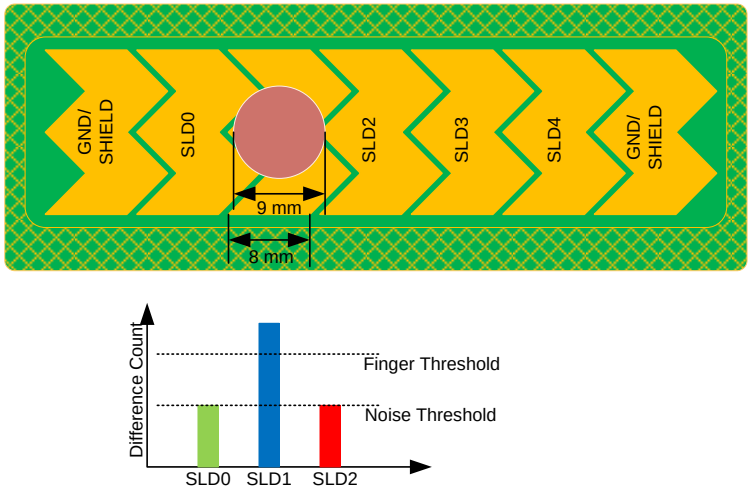


Figure 6-8. Ideal Slider Signals



Equation 6-1 Segment width and air gap relation with the finger diameter

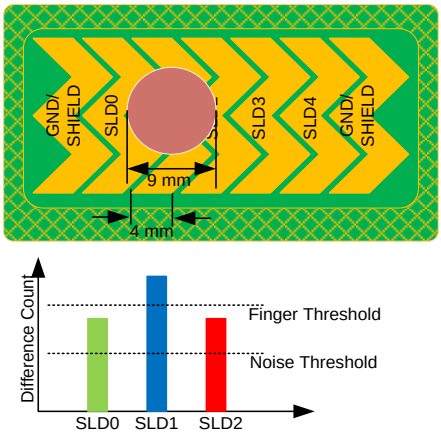
$$W + 2A = finger\ diameter$$

[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 *slider-segment-width + 2 * air-gap* is lesser than *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



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

$$Centroid\ position = \left(\frac{S_{x+1} - S_{x-1}}{S_{x+1} + S_{x0} + S_{x-1}} + maximum\right) * \frac{Resolution}{(n-1)}$$

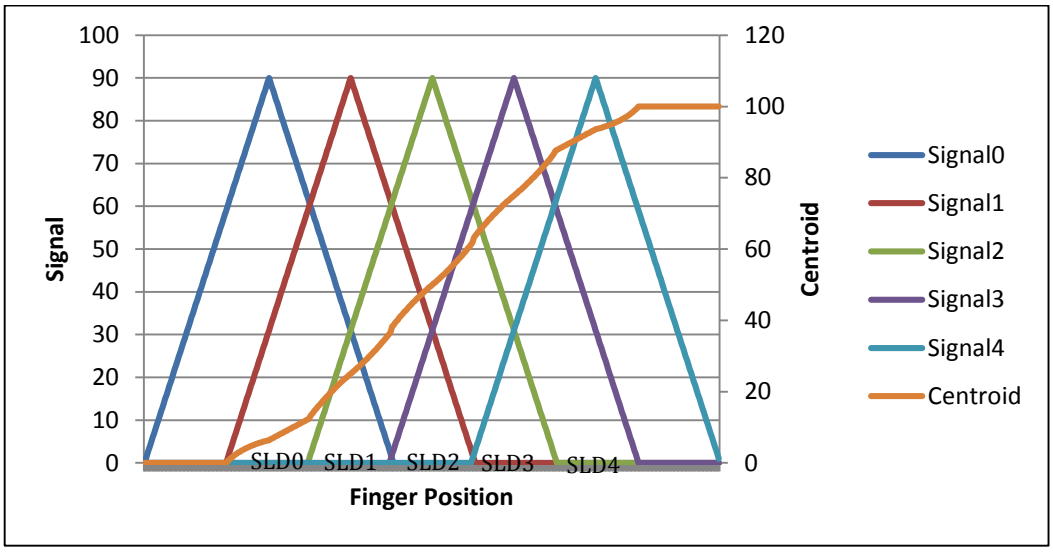
Resolution – API Resolution set in the CapSense Component Customizer

n – Number of sensor elements in the CapSense Component Customizer

maximum: Index of element which gives maximum signal

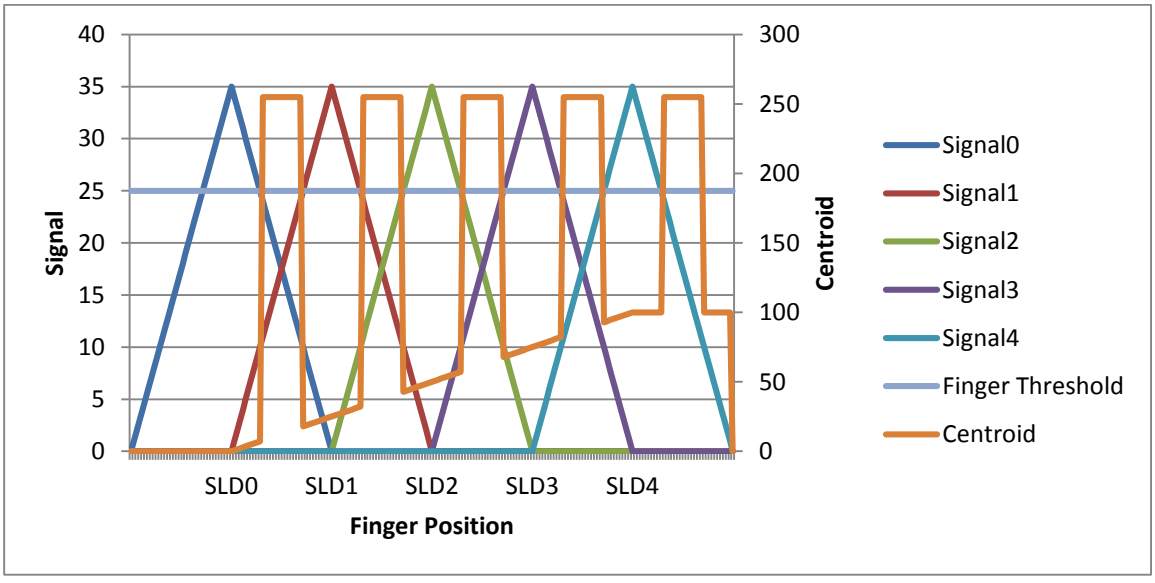
Si – different counts (with subtracted Noise Threshold value) near by the maximum position

Figure 6-10. Nonlinear Centroid Response when Slider Segment Width Is Lower Than Recommended



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 0xFF[1], as Figure 6-11 shows.

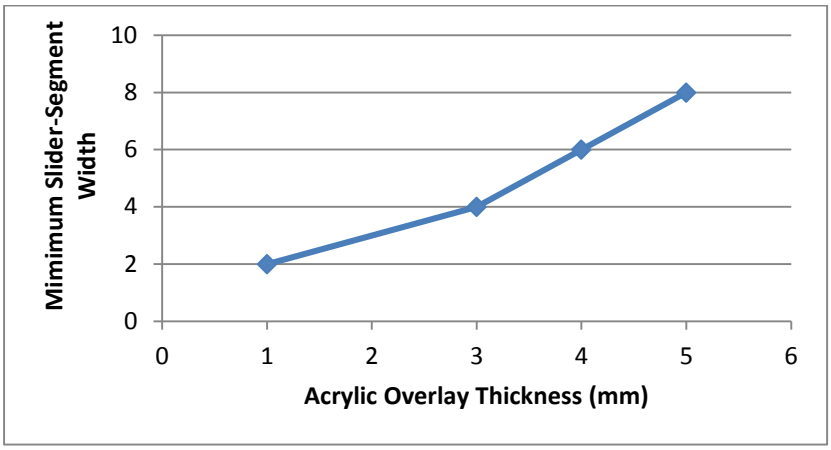
Figure 6-11. Incorrect Centroid Reported when Slider-Segment-Width Is Too Low



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.

Figure 6-12. Minimum Slider-Segment Width w.r.t. Overlay Thickness for an Acrylic Overlay



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.

Figure 6-13. Flat Spots (Nonresponsive Centroid) when Slider-Segment Width Is Higher than Recommended

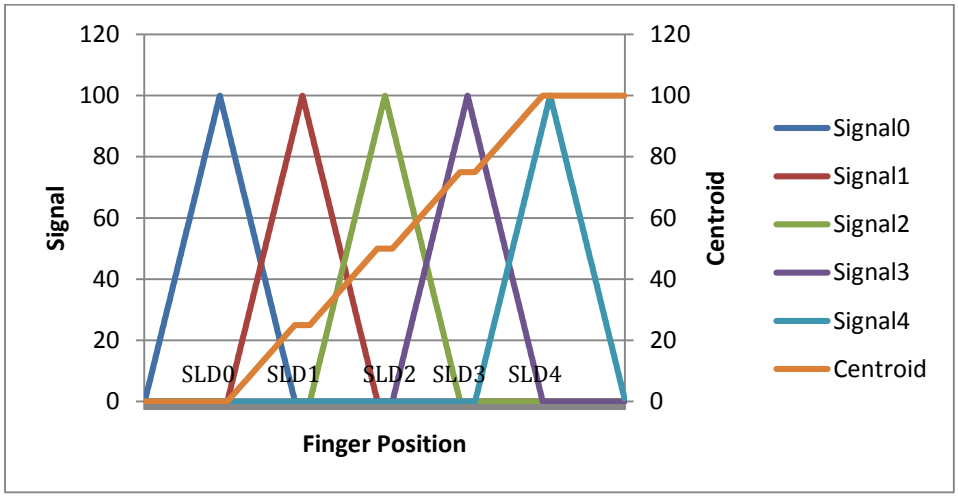
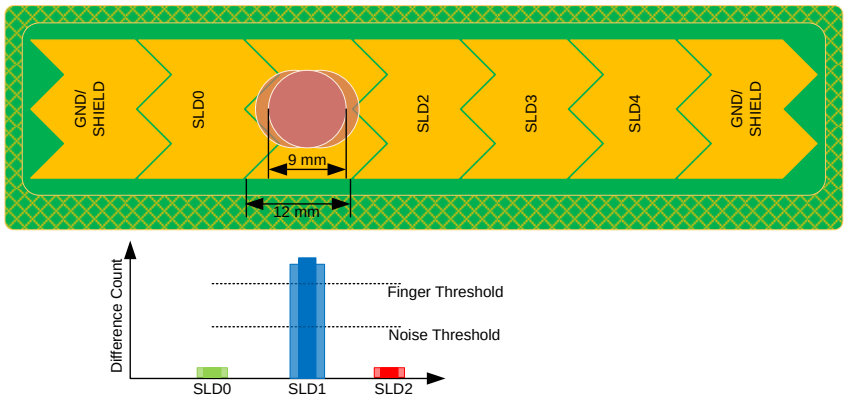


Figure 6-14. Signal on Slider Segments when Slider-Segment Width is Higher Than Recommended



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.

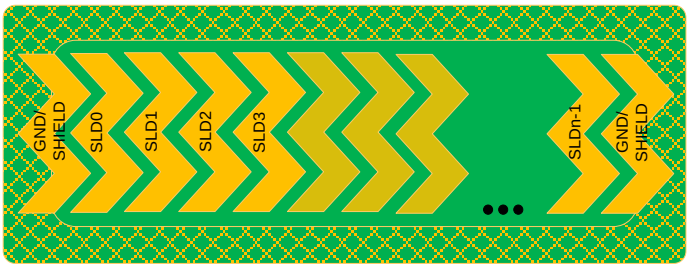
6.3.4.2 Dummy Segments at the Ends of a Slider

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

$$PCB\ length = Slider\ Length + 3 * slider\ segment\ width + 2 * air\ gap$$

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:

$$PCB\ length = Slider\ Length + slider\ segment\ width$$

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 Cp of other segments should be greater than 22.5 pF.

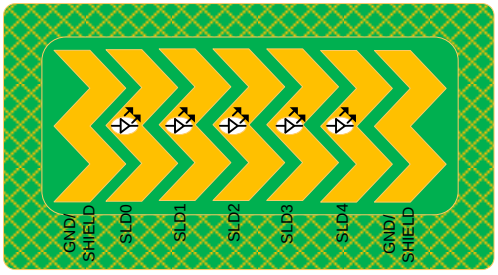
Implement the following layout design rules to meet this condition:

- 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, I^2C 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 I^2C 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

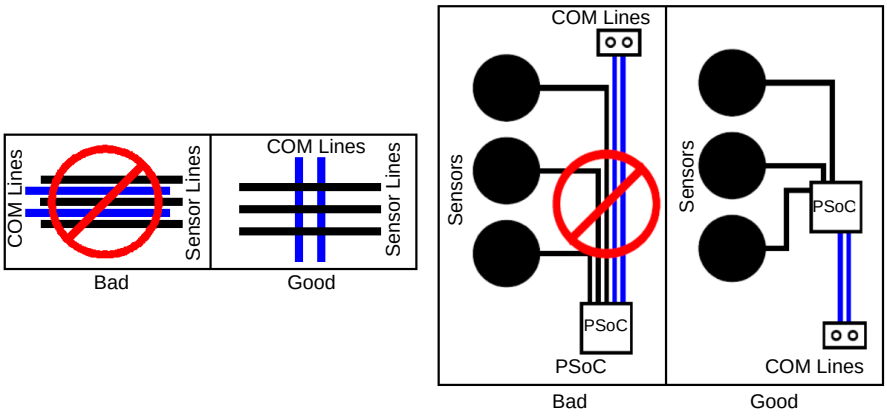
Use short and narrow PCB traces to minimize the parasitic capacitance of the sensor. The maximum recommended trace length is 12 inches (300 mm) for a standard PCB and 2 inches (50 mm) for flex circuits. The maximum recommended trace width is 7 mil (0.18 mm). You should surround the CapSense traces with 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.

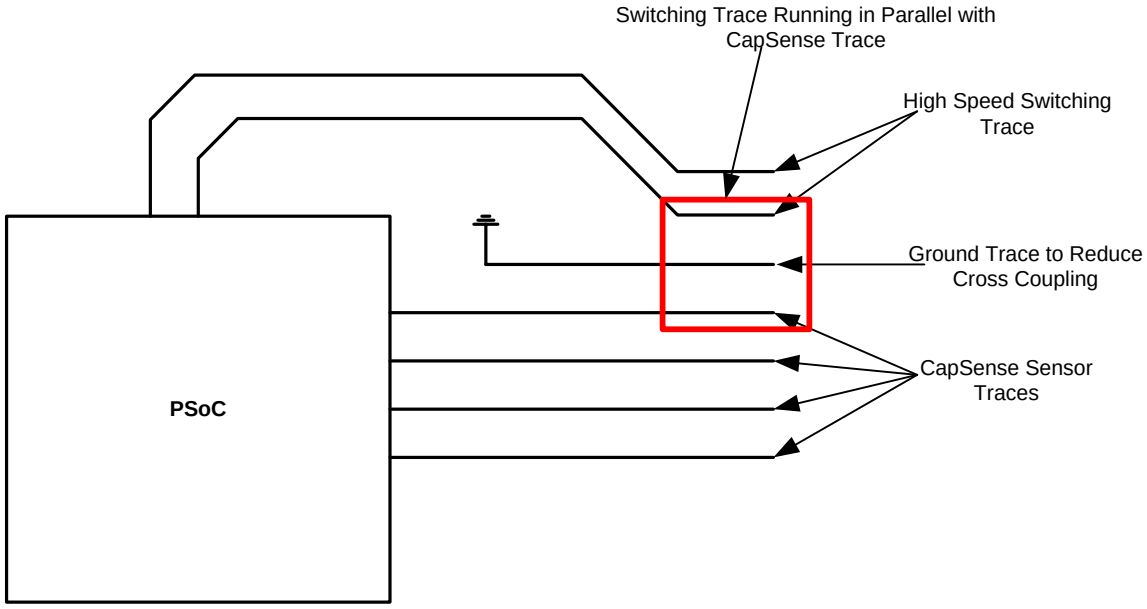
Figure 6-17. Routing of Sensor and Communication Lines



If, due to spacing constraints, sensor traces run in parallel with high-speed traces such as I^2C 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

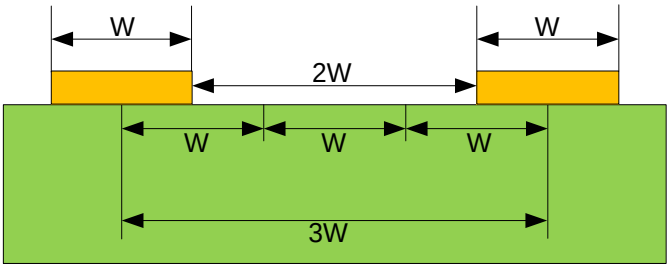
as a traces from temperature sensor to the PSoC device. The thickness of the ground trace can be 7 mils and the spacing from sensor trace to ground trace should be equal to minimum of 10 mils to reduce the C_P of the CapSense sensor.

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

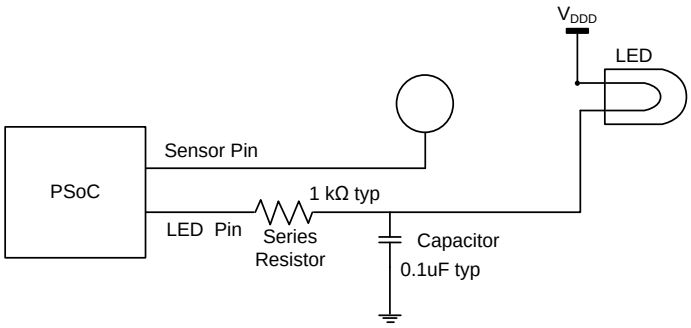


6.3.8 Crosstalk Solutions

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.

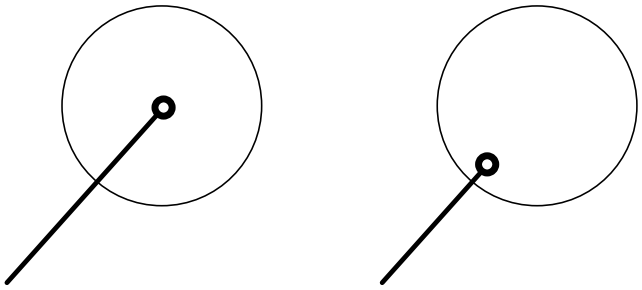
Figure 6-20. Reducing Crosstalk



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

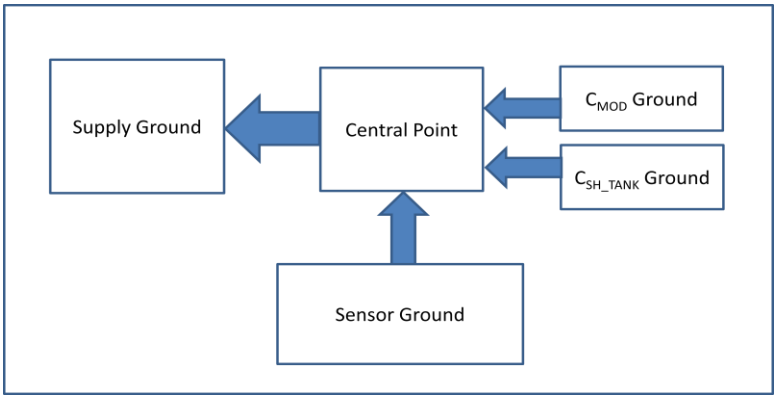


Table 6-4. Central Point for Star Connection

Family	Central point
PSoC 4000	VSS pin
PSoC 4100/ 4100 M	VSS pin
PSoC 4200/ 4200 M	VSS pin
PSoC 4100-BL	E-pad
PSoC 4200-BL	E-pad
PRoC BLE	E-pad

- 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

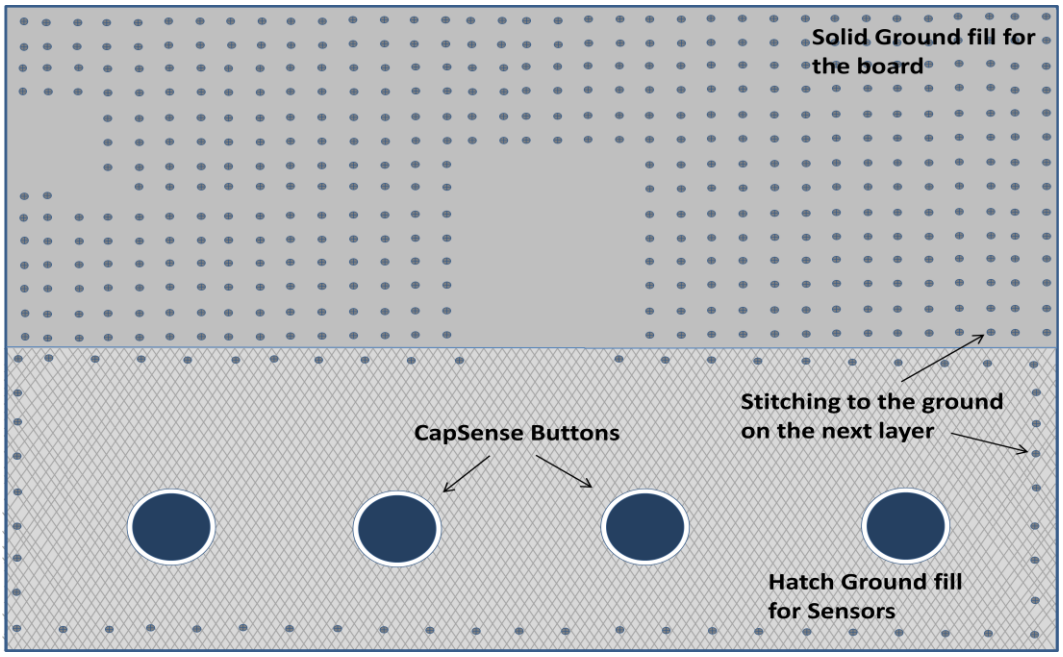
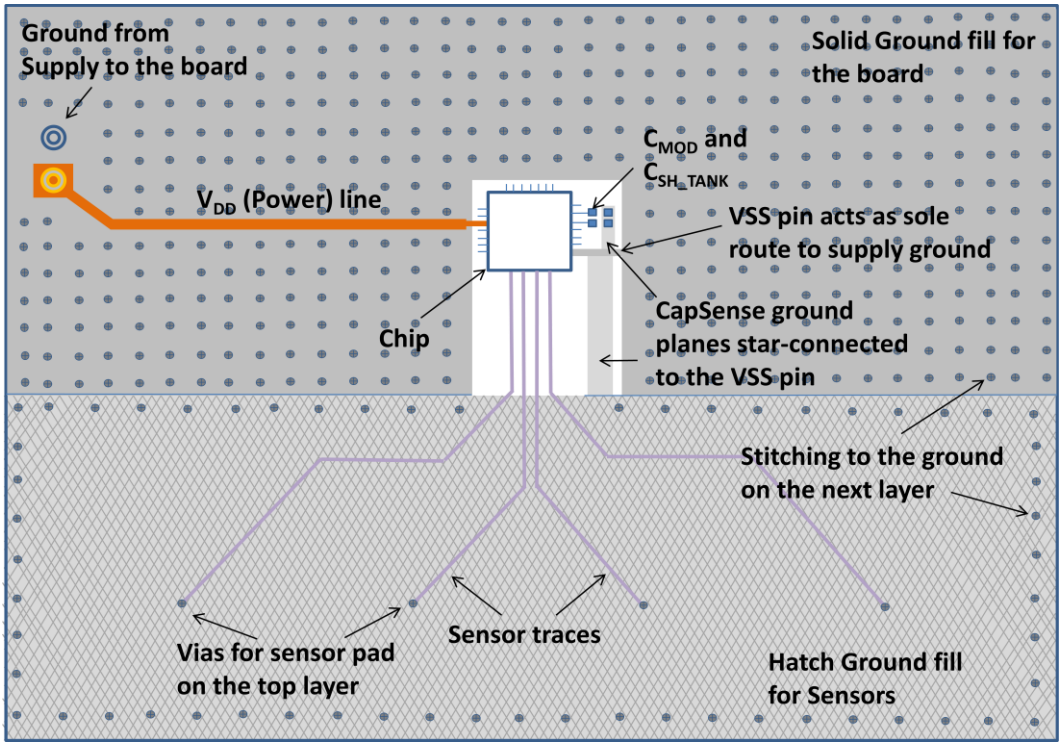


Figure 6-24. PCB Bottom Layer Layout Using a Chip Without E-pad



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)

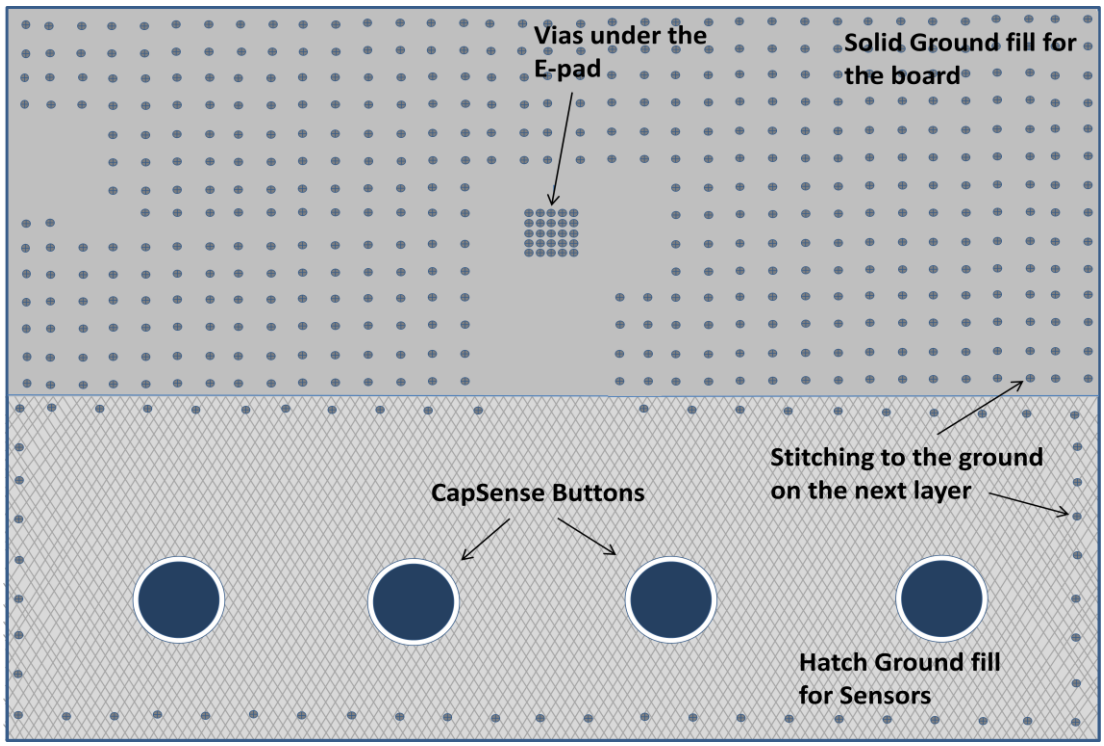
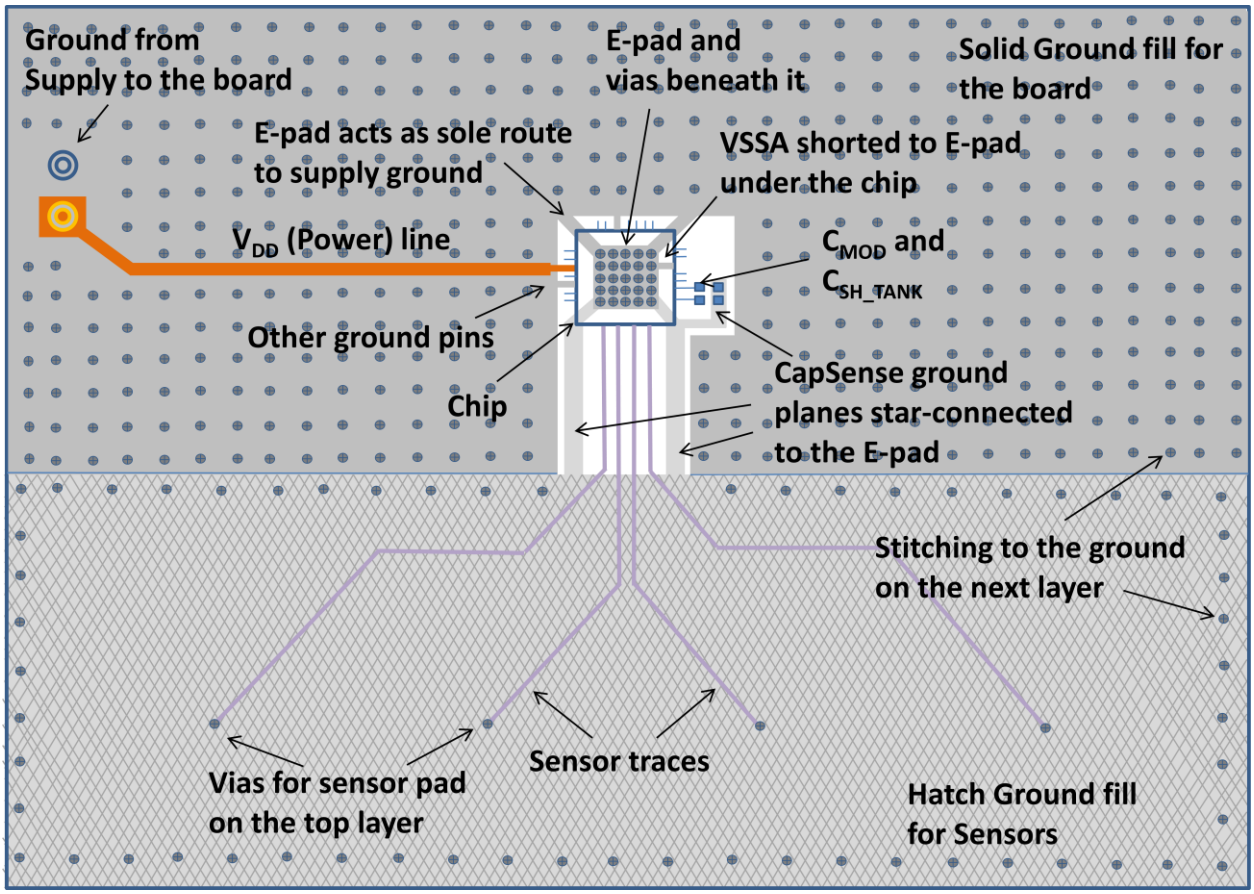


Figure 6-26. PCB Bottom Layer Layout with PSoC 4 BLE / PRoC BLE (with E-pad)



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 6-5. Decoupling Capacitors on Power Lines

Power Line	Decoupling Capacitors	Corresponding Ground Terminal	Applicable Device Family
VDD	0.1 µF and 1 µF	VSS	PSoC 4000
VDDIO	0.1 µF	VSS	PSoC 4000
VDDD	0.1 µF and 1 µF	VSS	PSoC 4100, PSoC 4200
	0.1 µF and 1 µF	VSSD	PSoC 4100-BL, PSoC 4200-BL, PRoC BLE
VDDA	0.1 µF and 1 µF	VSSA	PSoC 4100, PSoC 4200, PSoC 4100-BL, PSoC 4200-BL, PRoC BLE
VDDR	0.1 µF and 1 µF	VSSD	PSoC 4100-BL, PSoC 4200-BL, PRoC BLE
VCCD	Refer device datasheet	VSS (PSoC 4000) or VSSD (all others)	All device families

The decoupling capacitors and C_{MOD} 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.

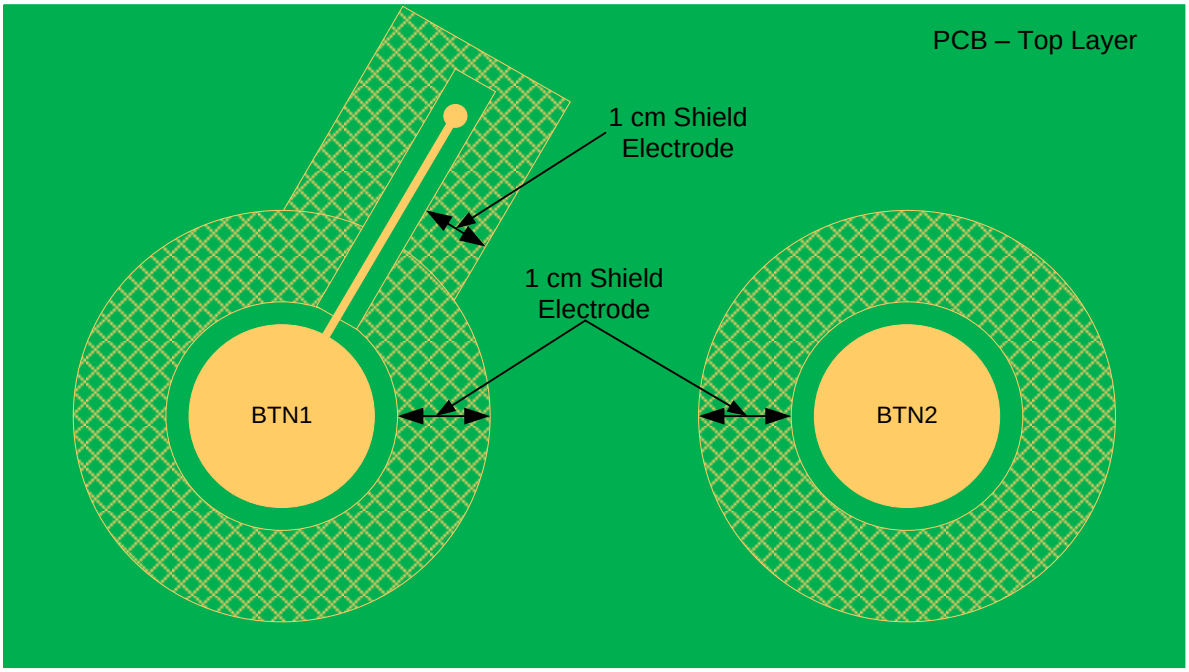
The area of the shield electrode depends on the size of the liquid droplet and the area available on the board for implementing the shield electrode. The shield electrode should surround the sensor pads and traces, and spread no further than 1 cm from these features. Spreading the shield electrode beyond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

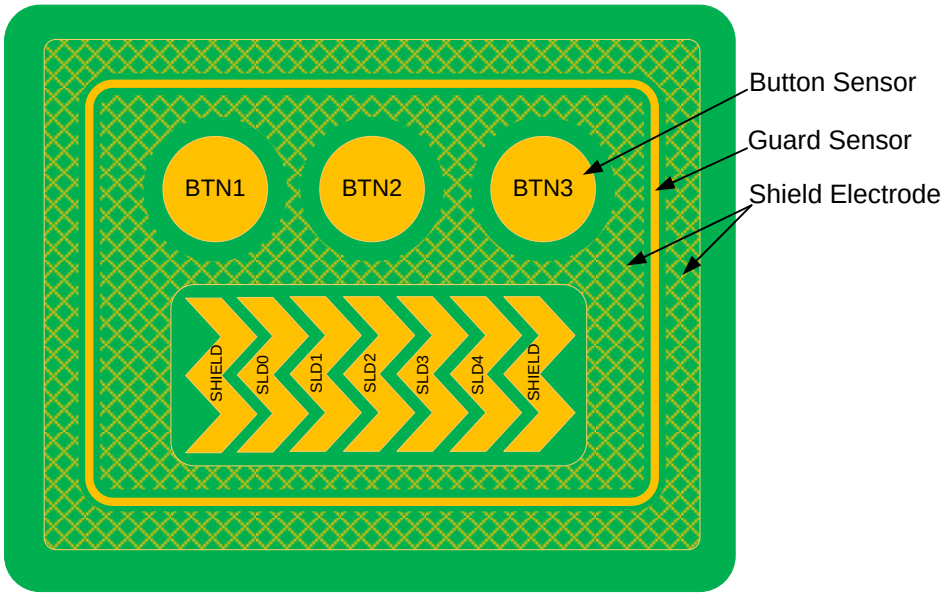
- Top layer: Hatch fill with 7-mil trace and 45-mil grid (25 percent fill). Hatch fill should be connected to the driven-shield signal.
- Layer-2: Hatch fill with 7-mil trace and 70-mil grid (17 percent fill). Hatch fill should be connected to the driven-shield signal.
- Layer-3: V_{DD} Plane
- Bottom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

Figure 6-28. PCB Layout with Shield Electrode and Guard Sensor



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.

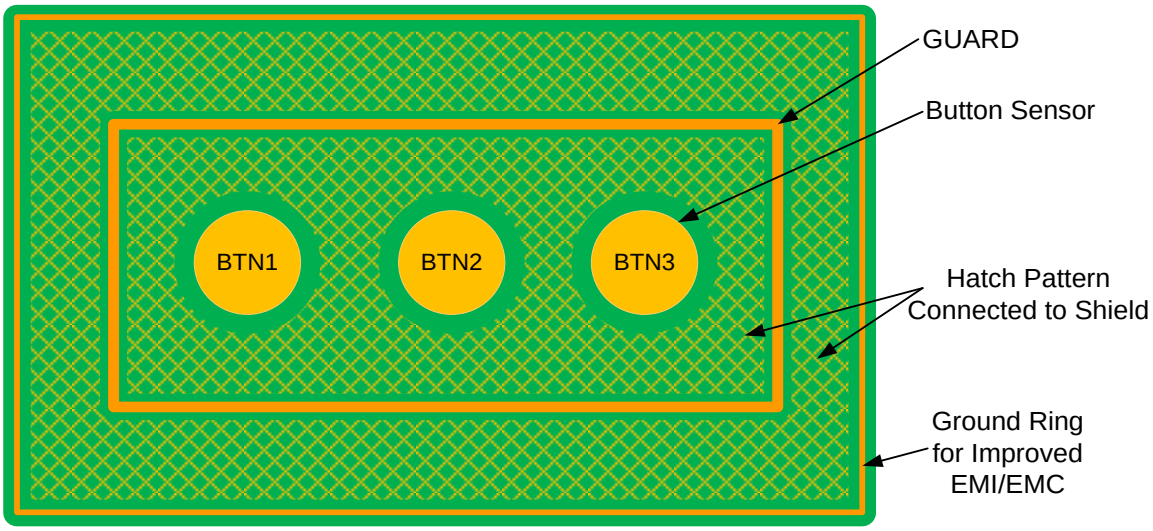
If there is no space on the PCB for implementing a guard sensor, the guard sensor functionality can be implemented in the firmware. For example, you can use the ON/OFF status of different sensors to detect a liquid stream. The following conditions can be used to detect a liquid stream on the touch surface:

- 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

No.	Category	Recommendations / Remarks
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 Ω 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^2C 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}

No.	Type	Recommended Pin				
		PSoC 4000	PSoC 4100, PSoC 4200	PSoC 4200M[1]	PSoC 4100-BL, PSoC 4200-BL	CYBL10X6X
1	C_{MOD}	P0[4]	P4[2]	P4[2] – CSD0 P5[0] – CSD1	P4[0]	P4[0]
2	C_{SH_TANK}	P0[2]	P4[3]	P4[3] – CSD0 P5[1] – CSD1	P4[1]	P4[1]

[1] See CapSense in PSoC 4 M-Series section for details on CSD0 and CSD1 block.

6.3.14 Layout Rule Checklist

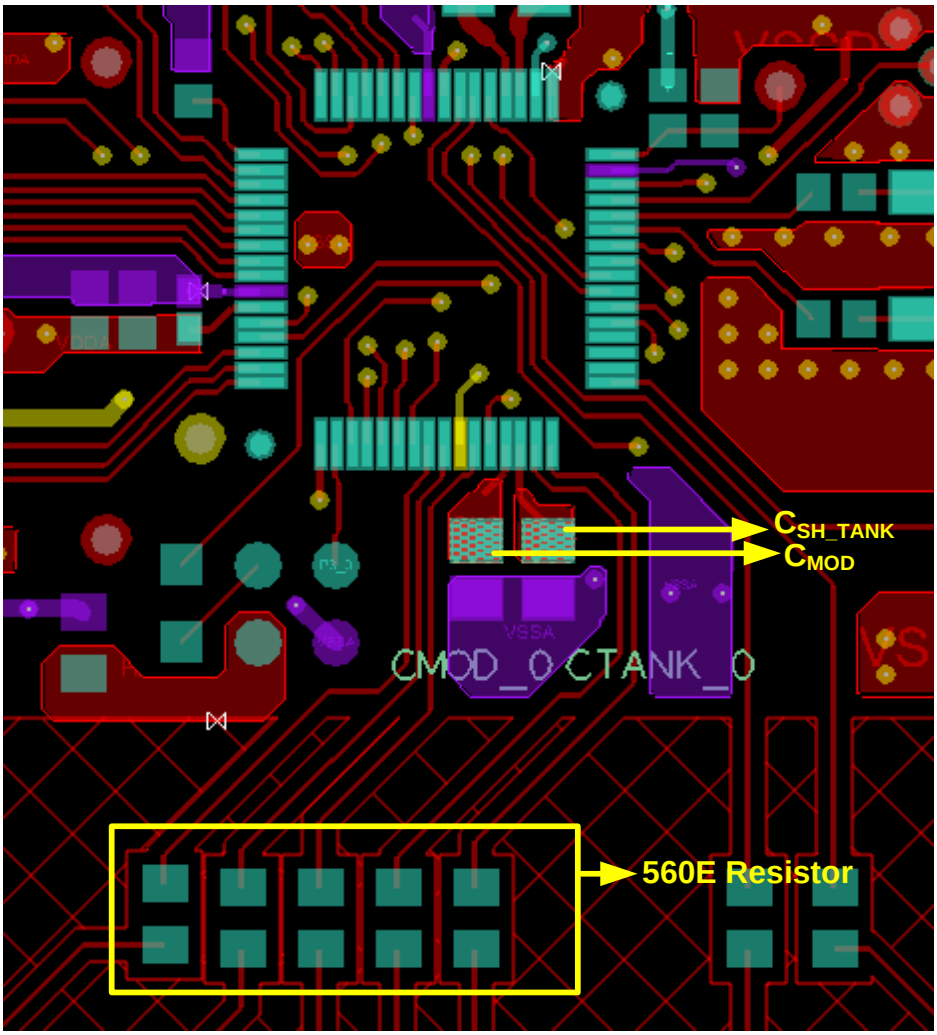
You can use the checklist provided in Table 6-8 to help verify your layout design.

Table 6-8. Layout Rule Checklist

No.	Category		Minimum Value	Maximum Value	Recommendations / Remarks
1	Button	Shape	N/A	N/A	Circle or rectangular with curved edges
		Size	5 mm	15 mm	10 mm
		Clearance to ground hatch	0.5 mm	2 mm	Should be equal to overlay thickness
2	Slider	Width of segment	1.5 mm	8 mm	8 mm
		Clearance between segments	0.5 mm	2 mm	0.5 mm
		Height of segment	7 mm	15 mm	12 mm
3	Overlay	Type	N/A	N/A	Material with high relative permittivity (except conductors) Remove any air gap between sensor board and overlay / front panel of the casing.
		Thickness for Buttons	N/A	5 mm	
		Thickness for Sliders	N/A	5 mm	
		Thickness for Touchpads	N/A	0.5 mm	
4	Sensor Traces	Width	N/A	7 mil	Use the minimum width possible with the PCB technology that you use.
		Length	N/A	300 mm for a standard (FR4) PCB 50 mm for flex PCB	Keep as low as possible
		Clearance to ground and other traces	0.25 mm	N/A	Use maximum clearance while keeping the trace length as low as possible
		Routing	N/A	N/A	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.
5	Via	Number of vias	1	2	At least 1 via is required to route the traces on the opposite side of the sensor layer
		Hole size	N/A	N/A	10 mil

No.	Category		Minimum Value	Maximum Value	Recommendations / Remarks
6	Ground		N/A	N/A	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)
7	Series resistor placement		N/A	N/A	Place resistor within 10 mm of PSoC pin. Refer to Figure 6-30 for an example placement of series resistance on board.
8	Shield electrode	Spread	N/A	1 cm	If you have PCB space, use 1-cm spread.
9	Guard sensor (for water tolerance)	Shape	N/A	N/A	Rectangle with curved edges
		Thickness	N/A	N/A	Recommended thickness of guard trace is 2 mm and distance of guard trace to shield electrode is 1 mm.
10	C_{MOD}		N/A	N/A	Place close to the PSoC pin. Refer to Figure 6-30 for an example placement of C_{MOD} on printed circuit board.
11	C_{SH_TANK}		N/A	N/A	Place close to the PSoC pin. Refer to Figure 6-30 for an example placement of C_{SH_TANK} on board.

Figure 6-30. Example Placement for C_{MOD}, C_{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 6-9. Overlay Thickness for ESD Protection

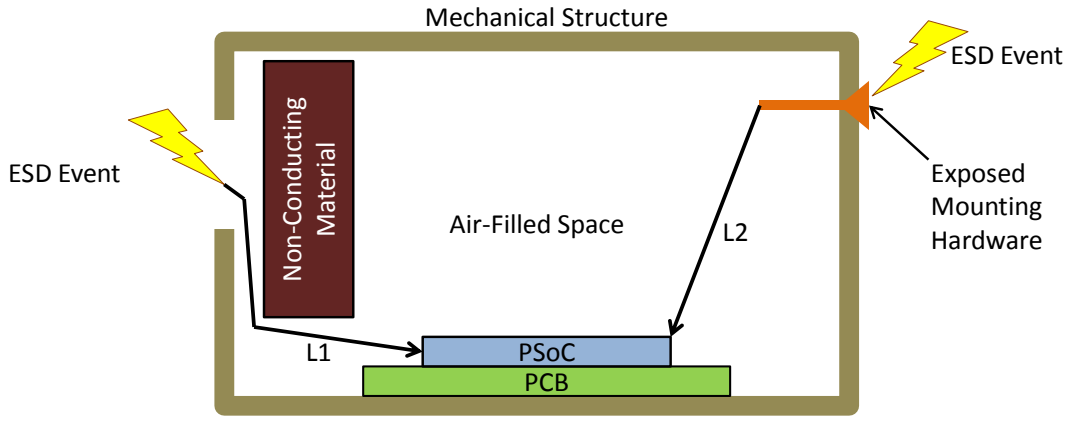
Material	Breakdown Voltage (V/mm)	Minimum Overlay Thickness for Protection Against 12 kV ESD (mm)
Air	1200 – 2800	10
Wood – dry	3900	3
Glass – common	7900	1.5
Glass – Borosilicate (Pyrex®)	13,000	0.9
PMMA Plastic (Plexiglas®)	13,000	0.9
ABS	16,000	0.8
Polycarbonate (Lexan®)	16,000	0.8
Formica	18,000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

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.

Figure 6-31. ESD Paths

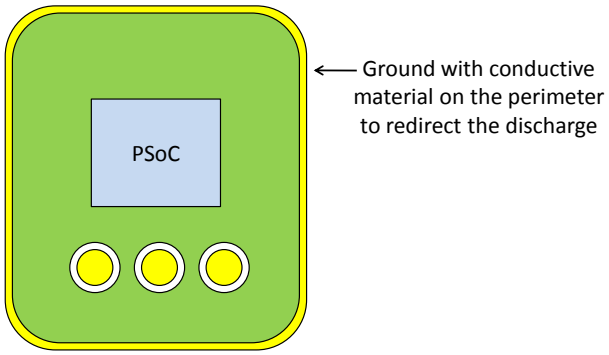


If it is not possible to maintain adequate distance, place a protective layer of nonconductive material with a high breakdown voltage between the possible ESD source and PSoC. One layer of 5-mil thick Kapton® tape can withstand 18 kV. See Table 6-9 for other material dielectric strengths.

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

ESD Protection Device		Input Capacitance	Leakage Current	Contact Maximum ESD Limit	Air Discharge Maximum ESD Limit
Manufacturer	Part Number				
Littelfuse	SP723	5 pF	2 nA	8 kV	15 kV
Vishay	VBUS05L1-DD1	0.3 pF	0.1 µA	±15 kV	±16 kV
NXP	NUP1301	0.75 pF	30 nA	8 kV	15 kV

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

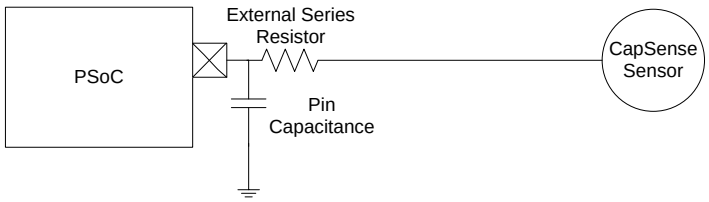
6.5.1.1.1 Ground Plane

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, C_P, 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.

Figure 6-33. RC Filter



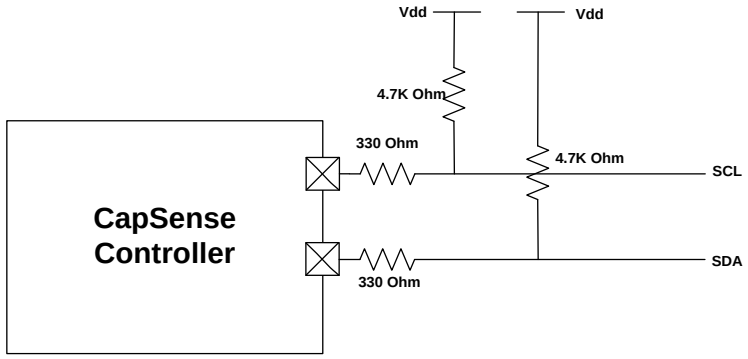
Series resistors should be placed close the device pins so that the radiated noise picked by the traces gets filtered at the input of the device. Thus, it is recommended to place series resistors within 10 mm of the pins.

For CapSense designs using copper on PCBs, the recommended series resistance for CapSense input lines is 560 Ω. 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 Ω, 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^2C and SPI, also benefit from series resistance; 330 Ω 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^2C communication lines is 4.7 kΩ. So, if more than 330 Ω is placed in series on these lines, the V_{IL} and V_{IH} voltage levels may fall out of specifications. 330 Ω will not affect I^2C operation as the V_{IL} level still remains within the I^2C specification limit of 0.3 V_{DD} when PSoC outputs a LOW.

Figure 6-34. Series Resistors on Communication Lines



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

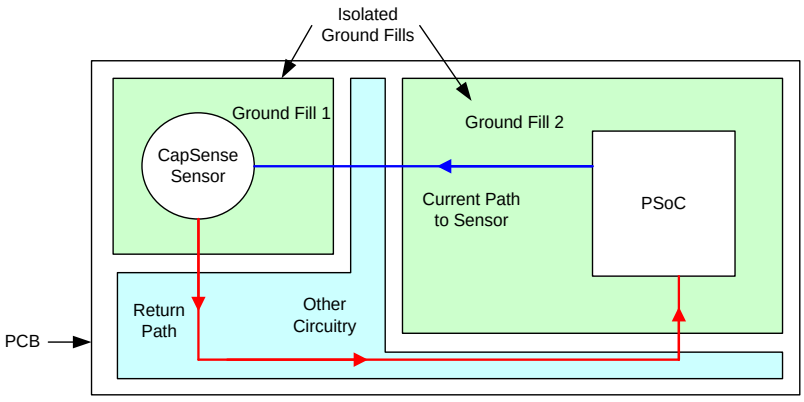
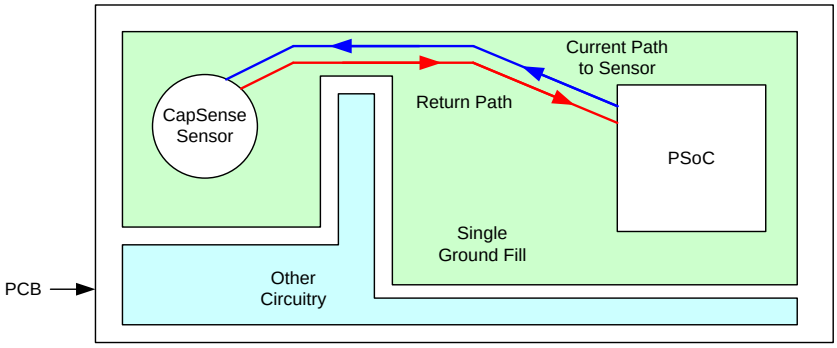


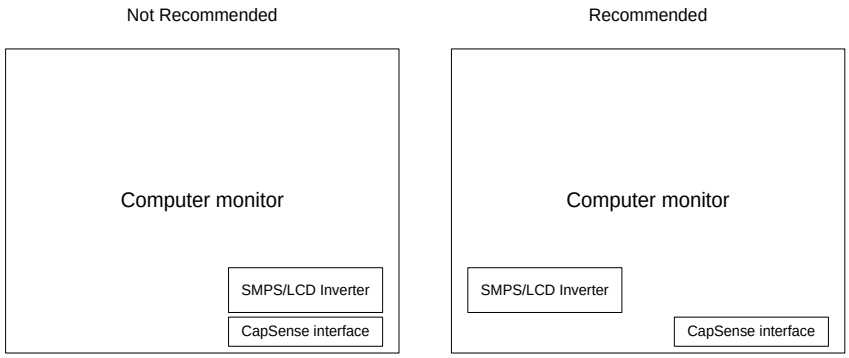
Figure 6-36. Proper Current Loop Layout



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.

Figure 6-37. Separating Noise Sources



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 V_{DD} on Radiated Emissions (150 kHz – 30 MHz)

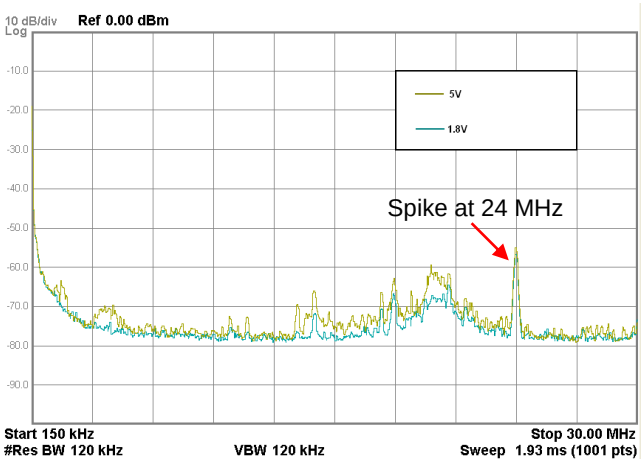
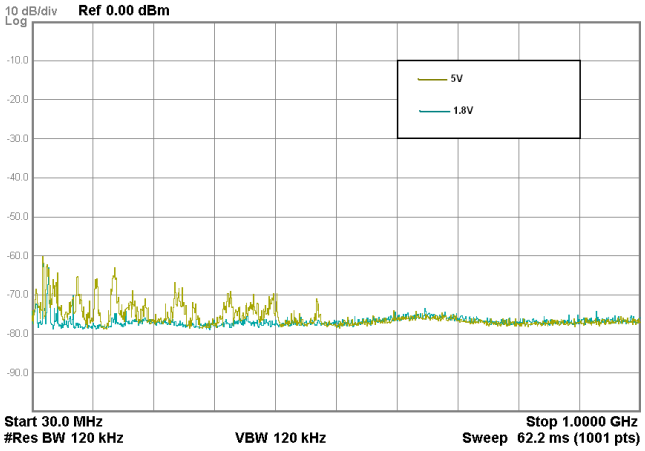


Figure 6-39. Effect of VDD on Radiated Emissions (30 MHz – 1 GHz)



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 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)

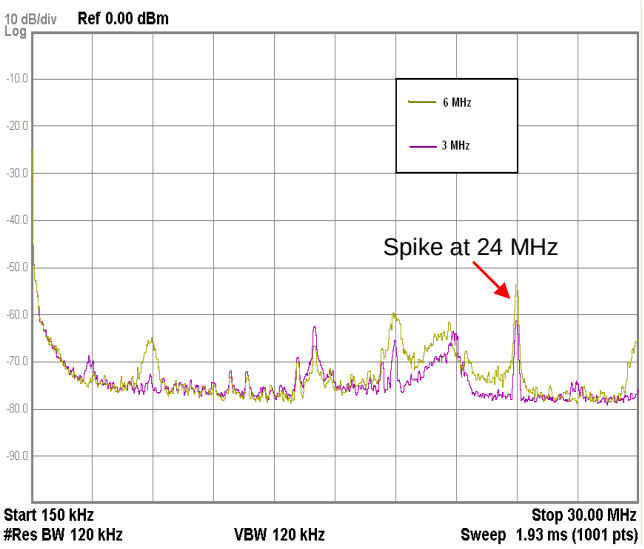
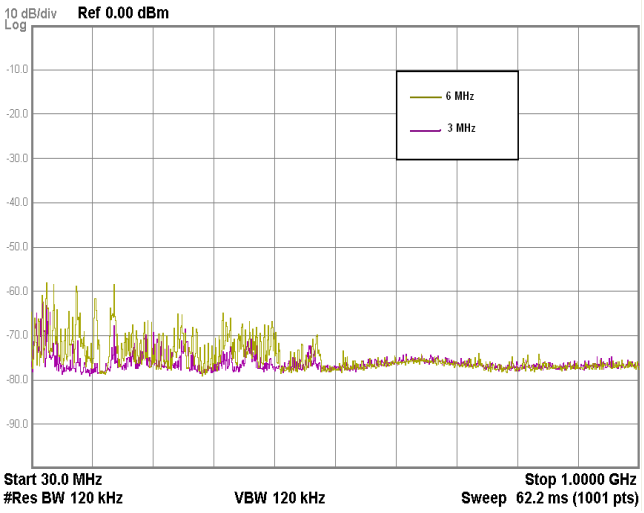


Figure 6-41. Effect of Sensor-Switching Frequency on Radiated Emissions (30 MHz – 1 GHz)



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)

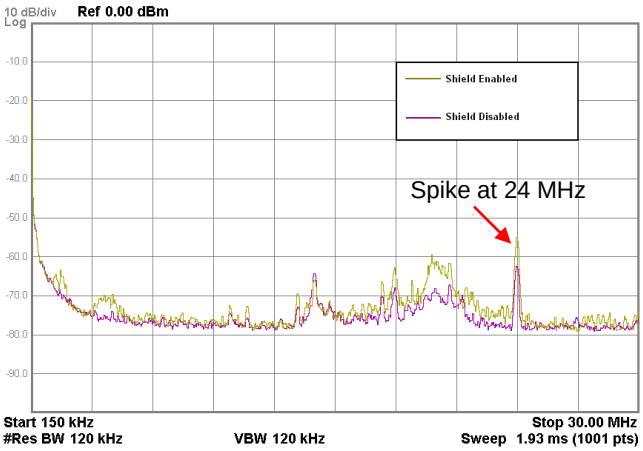
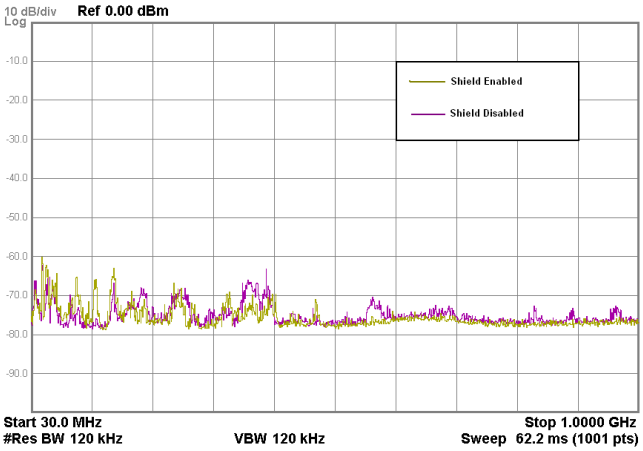


Figure 6-43. Effect of Shield Electrode on Radiated Emissions (30 MHz – 1 GHz)



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

Parameter	Total Scan time for 5 Buttons	
	0.426 ms	0.106 ms
Modulation Clock Divider	2	2
Scan Resolution	10 bits	8 bits
Individual sensor scan time	0.085 ms	0.021 ms

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

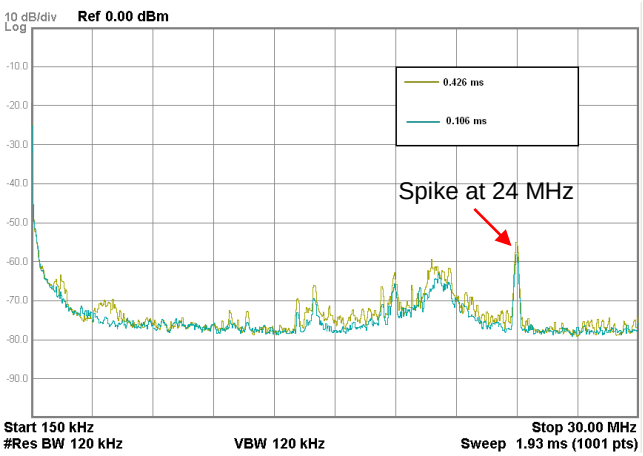
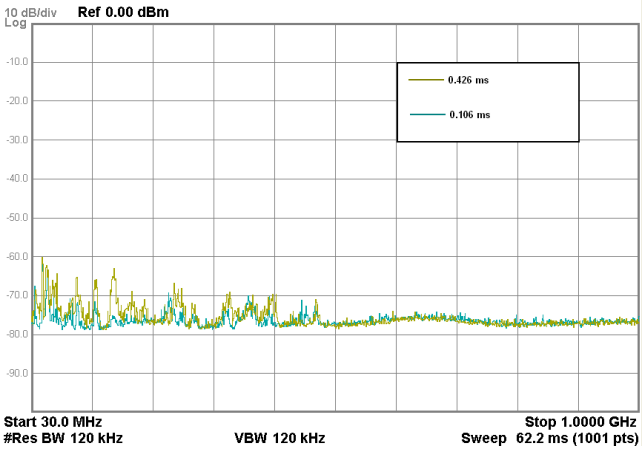


Figure 6-45. Effect of Scan Time on Radiated Emissions (30 MHz – 1 GHz)



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

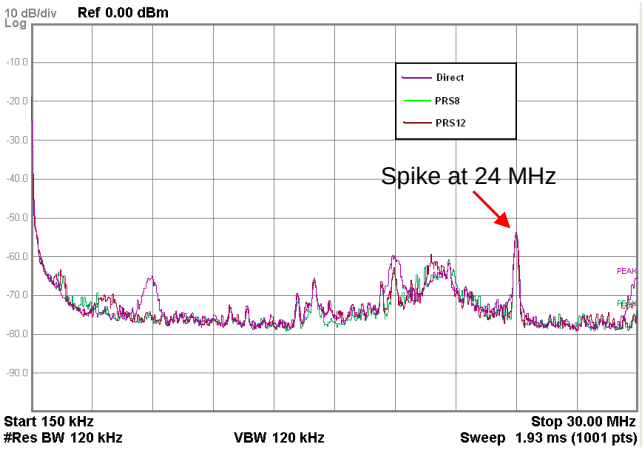
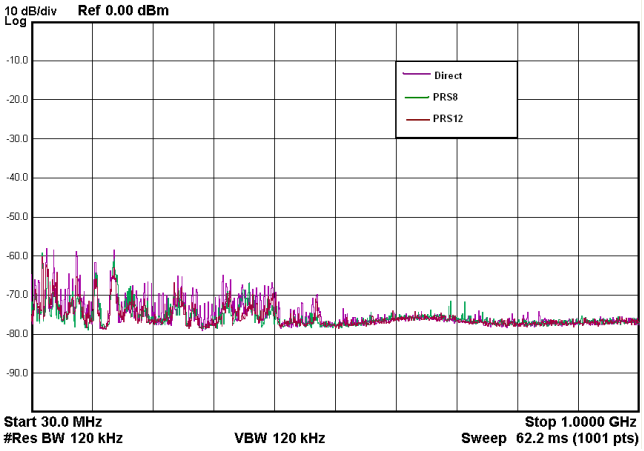


Figure 6-47. Effect of Analog Switch Drive Source on Radiated Emissions (30 MHz – 1 GHz)



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)

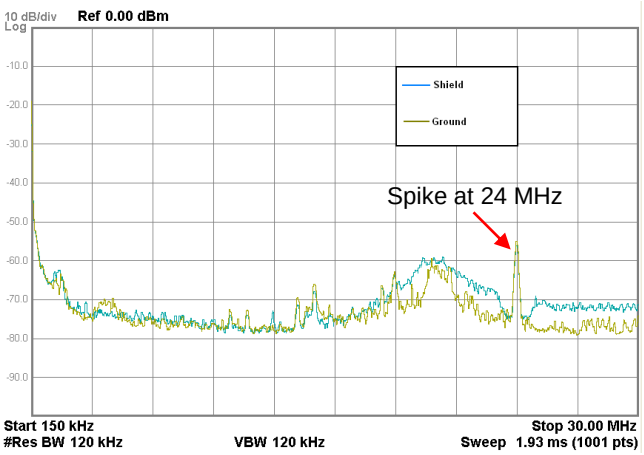
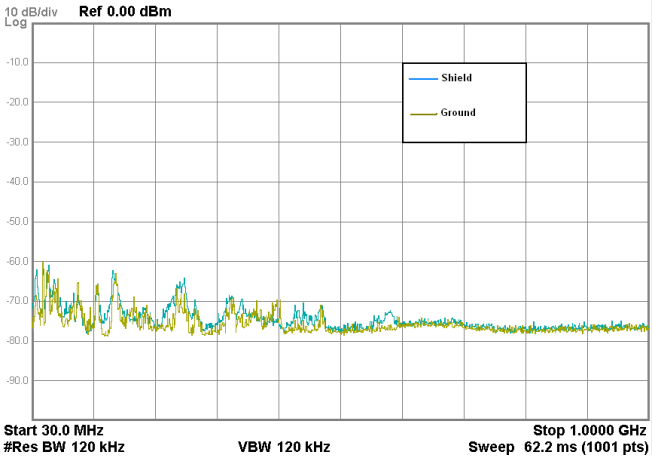


Figure 6-49. Effect of Inactive Sensor Termination on Radiated Emissions (30 MHz – 1 GHz)



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.

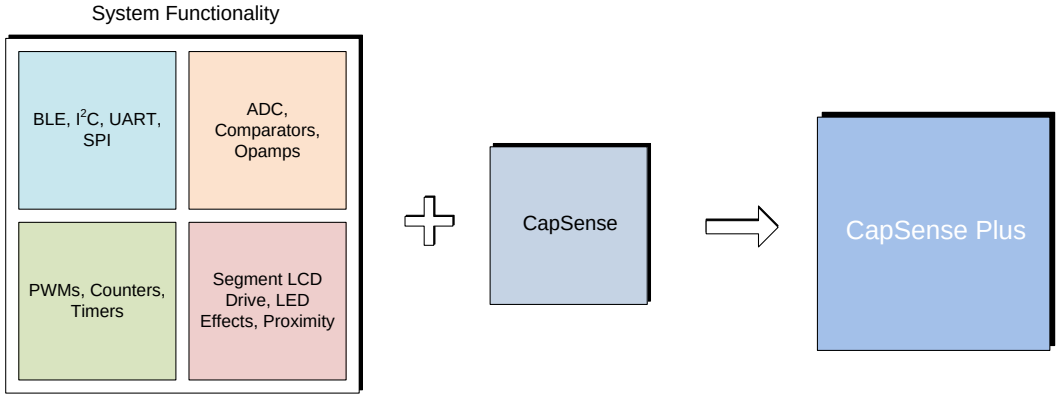
- Use decoupling capacitors on the power supply pins to reduce the conducted noise from the power supply. See section 6.3.11 and the PSoC 4 device datasheet for details.
- Provide GND and V_{DD} planes on the PCB to reduce current loops.
- If the PSoC PCB is connected to the power supply using a cable,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

7. CapSense Plus

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.

Figure 7-1. CapSense Plus



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

- Communication: BLE, I^2C, 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 7-1. Examples of CapSense Plus

Application	CapSense	OpAmp	ADC	Comp	PWM, Counter, Timer, UDBs	Comm (BLE, I²C, SPI, UART)	LCD drive	GPIOs
Heart Rate Monitor (Wrist Band)	User interface: buttons, linear sliders	TIA, Buffer	Heart Rate Measurement, Battery voltage measurement		LED Driving	BLE	Segment LCD	LED indication
LED Bulb	User interface: buttons, radial sliders	Amplifier	LED current measurement	Short Circuit Protection	LED color control (PrISM*)	BLE		LED indication
Washing machine	User interface: buttons, radial sliders		Temperature sensor	Water level monitor	Buzzer, FOC** motor control	I²C LCD display, UART network interface	Segment LCD	LED indication
Water heater	User interface: buttons, linear sliders		Temperature sensor, water flux sensor	Water level monitor	Buzzer	I²C LCD display, UART Network Interface	Segment LCD	LED indication
IR remote controllers	User interface: buttons, linear and radial sliders, touch pads				Manchester encoder			LED indication
Induction cookers	User interface: buttons, linear sliders		Temperature sensor				Segment LCD	LED indication
Motor control systems	User interface: buttons, linear sliders				BLDC*** and FOC motor control			LED indication
Gaming / simulation controllers	User interface: buttons, touchpads		Reading analog joysticks			I²C/SPI/UART communication interface	Segment LCD	LED indication

Application	CapSense	OpAmp	ADC	Comp	PWM, Counter, Timer, UDBs	Comm (BLE, I²C, SPI, UART)	LCD drive	GPIOs
Thermal printers	User interface: buttons		Overheat protection, paper sensor		Stepper motor control	SPI communication interface		LED indication

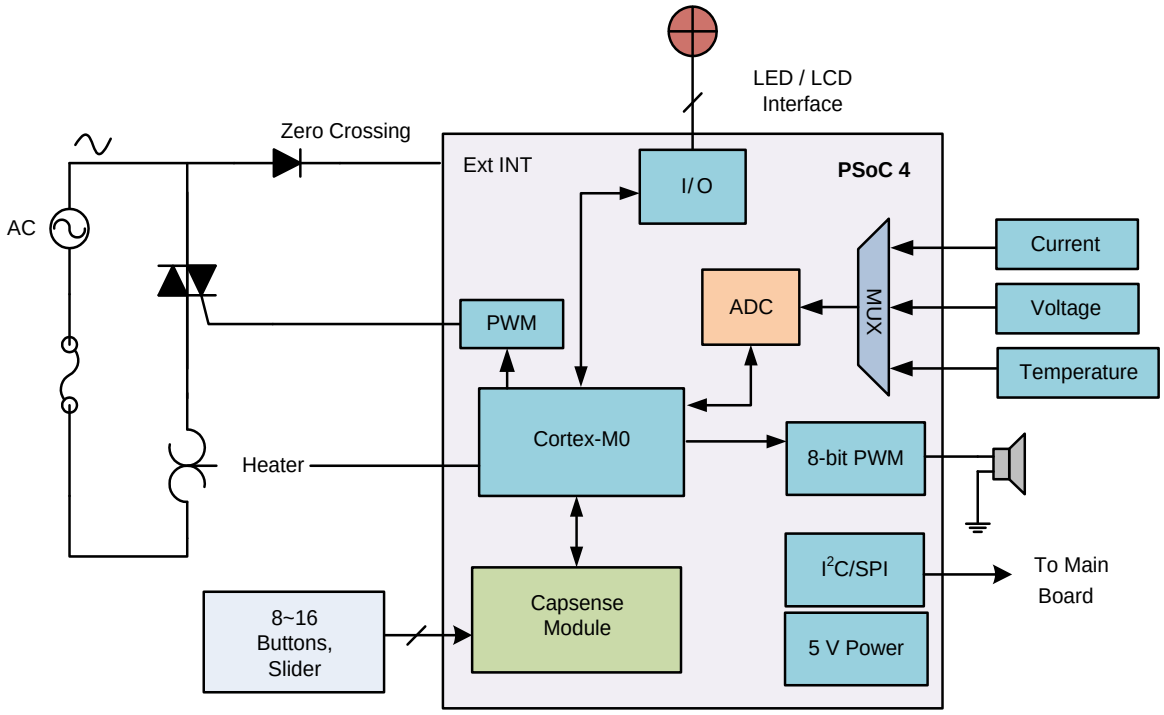
* PrISM stands for Precision Illumination Signal Modulation

** FOC stands for Field Oriented Control

*** BLDC stands for Brushless DC Motor

Figure 7-2 shows a general block diagram of a CapSense Plus application, such as an induction cooker or a microwave oven.

Figure 7-2. A CapSense Plus System With PSoC 4



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 8-1. PSoC 4 CapSense Development Kits

Development Kit	Supported CapSense Features
PSoC 4 Pioneer Kit (CY8CKIT-042)	A 5-segment linear slider
PSoC 4 BLE Bluetooth Low Energy Pioneer Kit (CY8CKIT-042-BLE)	A 5-segment linear slider and a wire proximity sensor
PSoC 4200-M Pioneer Kit(CY8CKIT-044)	A 5-element gesture detection, two proximity wire sensors
PSoC 4 Processor Module (CY8CKIT-038), with PSoC Development Kit (CY8CKIT-001)	A 5-segment linear slider and two buttons
CapSense Expansion Board Kit (CY8CKIT-031), to be used with CY8CKIT-038 and CY8CKIT-001	A 10-segment slider, 5 buttons and a 4x4 matrix button with LED indication.
MiniProg3 Program and Debug Kit (CY8CKIT-002)	CapSense performance tuning in CY8CKIT-038
PSoC 4000 Pioneer Kit (CY8CKIT-040)	A 5x6 CapSense Touchpad and a wire proximity sensor

Table 8-2 lists Cypress Reference Design kits that support PRoC BLE CapSense.

Table 8-2. PRoC BLE CapSense Reference Design Kits

Reference Design Kit	Supported CapSense Features
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

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				
Revision	**ECN#**	**Issue Date**	**Origin of Change**	**Description of Change**
**	3973432	04/19/2013	NIDH	New Design Guide
*A	4059171	07/29/2013	NIDH	Added dual IDAC support. Updated some shematics in chapter 6. Other minor changes to chapters 3, 5, and 6.
*B	4189700	11/13/2013	NIDH	Added support of CY8C4000 devices. Minor fixes throughout the document.
*C	4289925	02/24/2014	NIDH	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.
*D	4293476	02/27/2014	NIDH	Updated Table 1-1 per PSoC 4000 datasheet.
*E	4314223	03/20/2014	NIDH	Added firmware design considerations to Chapter 6. Added power supply layout and schematic considerations to Chapter 6. Updated the IMO range for PSoC 4000
*F	4339713	04/15/2014	NIDH	Updated to support PSoC 4000 and PSoC Creator 3.0 SP1.
*G	4494249	08/29/2014	DCHE	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 Sction 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

Document Title: AN85951 – PSoC® 4 CapSense® Design Guide Document Number: 001-85951				
Revision	**ECN#**	**Issue Date**	**Origin of Change**	**Description of Change**
				Added recommendations on DC loads in Section 6.3.5 Sensor Device and Placement Renamed Section 6.3.12 to Layout Guidelines for Liquid Tolerance and re-wrote this section Added Section Updated slider related recommendations in Table 6-8. Layout Rule Checklist Updated Section 6.5 Electromagnetic Compatibility (EMC) Considerations, added extensive data on hardware and firmware considerations.
*H	4602375	12/19/2014	UDYG	Added information for the PSoC 4 BLE family of devices. Added information for the PRoC BLE family of devices. Updated ground and power layout guidelines in Section 6.3.10 and Section 6.3.11 .
*I	4624027	01/21/2015	NIDH	Added information for PSoC 4200-M family of devices Added footnote in section 6.3.4 Slider Design Added GPIO source/sink current limit in Table 6-6. Changed document title to PSoC® 4 CapSense® Design Guide – AN85951
*J	4771699	06/02/2015	DCHE	Changed Document Title to read as “AN85951 – PSoC® 4 CapSense® Design Guide”. Updated Design Considerations: Updated ESD Protection: Updated Preventing ESD Discharge: Updated Figure 6-31. Updated Redirect: Replaced "Guard Ring" with "Ground Ring". Updated Figure 6-32.
*K	4891423	08/20/2015	DCHE	Added Table 3-1. Removed section 3.2.1 CMOD Precharge. Added section CapSense in PSoC 4 M-Series. Updated section Trace Routing. Added reference of AN2397. Added recommendation for modulator clock divider in section Manual Tuning Process. Added Figure 6-30.