PSoC® 4

PSoC® 4 CapSense® Design Guide

Doc. No. 001-85951 Rev. *A

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

Copyrights

Trademarks

Source Code

Disclaimer

Contents

1. Introduction

1.1 Abstract

The PSoC® 4 CapSense® Design Guide shows how to design capacitive touch sensing applications with the PSoC 4 family of devices. The CapSense feature in PSoC 4 offers unprecedented signal to noise ratio, best in class waterproofing, and a wide variety of sensors such as buttons, sliders, track pads and proximity sensors. This guide explains PSoC 4 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 family of devices. PSoC 4 is a true **programmable embedded system-on-chip** integrating configurable analog and digital peripheral functions, memory, and a microcontroller on a single chip. This device is 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 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. 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
- Design and development tools available for CapSense
- Performance tuning
- CapSense plus other applications

1.3 PSoC 4 CapSense Features

PSoC 4 CapSense 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
- Supports as many as 35 sensors
- High range proximity sensing
- Water tolerant operation

- Low power consumption
- Two IDAC operation to increase scan speed and SNR
- Any GPIO pin can be used for sensing or shielding
- 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)

1.4 PSoC 4 CapSense Plus Features

You can create PSoC 4 "CapSense plus applications" that feature capacitive touch sensing and additional system functionality. The main features of PSoC 4 are:

- Cortex-M0 CPU with single cycle multiply delivering 43 DMIPS at 48 MHz
- 1.71 – 5.5-V operation over –40 to 85 °C ambient
- 32 KB of flash (Cortex-M0 has >2X code density over 8-bit solutions)
- 4 KB of SRAM
- 36 GPIO pins in the larger packages
- 4 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
- 2 opamps with comparator mode and SAR input buffering capability
- 4-common segment LCD direct drive
- Low leakage retention (Hibernate) and non-retention (Stop) power modes with wakeup ability, using only 150 nA and 20 nA respectively
- 2 SPI / UART / I2C serial communication channels
- 4 programmable logic blocks, each having 8 macrocells and a cascadable data path, called universal digital blocks (UDBs) for very efficient implementation of programmable peripherals (such as I2S)
- 28-pin SSOP, 40-pin QFN, and 44-pin TQFP packages
- 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

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 1-1 on page 8 provides links to the supporting documents for each of the numbered tasks in Figure 1-1.

Figure 1-1. CapSense Design Flow

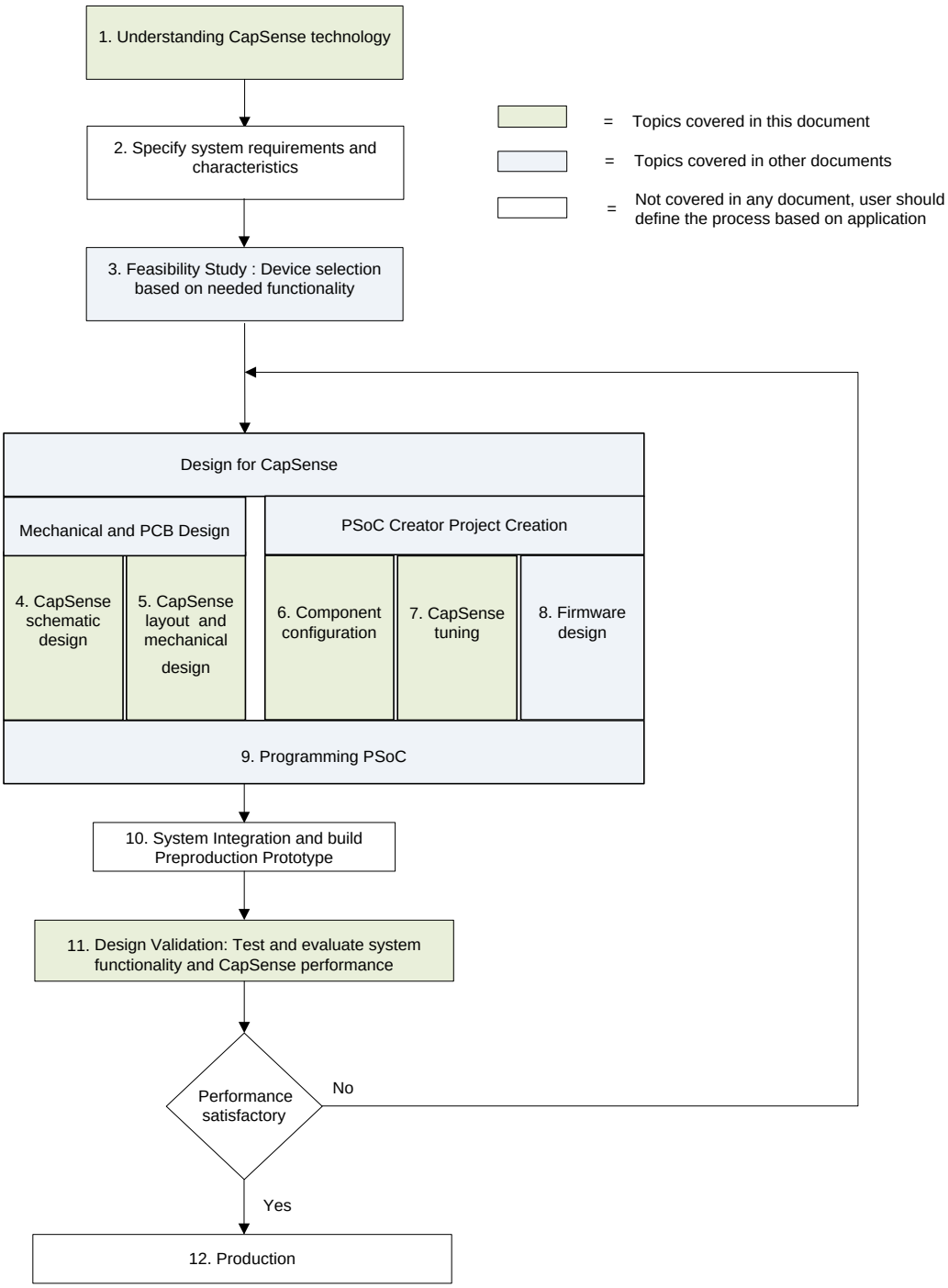


Table 1-1. Supporting Documentation

Steps in flowchart	Supporting Cypress Documentation	
	Name	Chapter
1. Understanding CapSense	PSoC 4 CapSense Design Guide	Chapter 2 and Chapter 3
2. Specify requirements	–	–
3. Feasibility study	PSoC 4 Datasheet	–
	AN79953, Getting Started with PSoC 4	–
4. Schematic design	PSoC 4 CapSense Design Guide	Chapter 6
5. Layout design	PSoC 4 CapSense Design Guide	Chapter 6
6. Component configuration	CapSense Component datasheet	–
	PSoC 4 CapSense Design Guide	Chapter 5
7. Performance tuning	PSoC 4 CapSense Design Guide	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	Chapter 5
12. Production	–	–

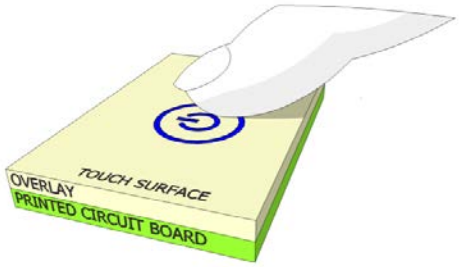
2. CapSense Technology

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.

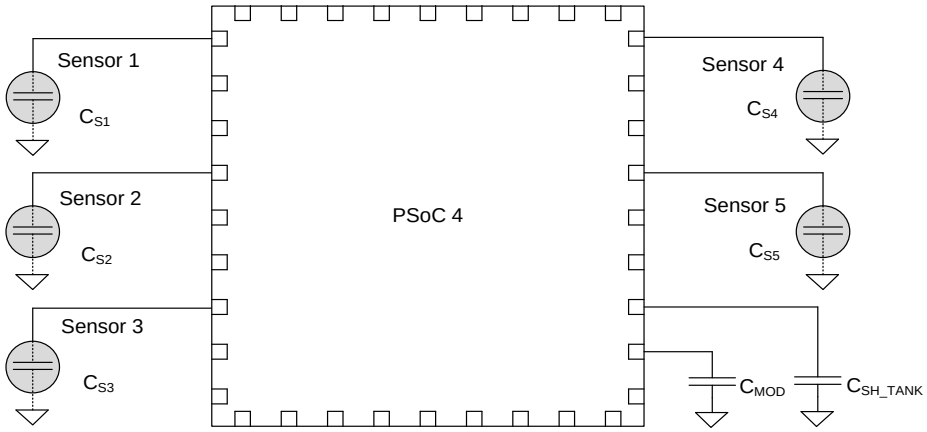
Figure 2-1. Capacitive Touch Sensor



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 waterproofing or proximity sensing is used, an additional C_{SH_TANK} capacitor may be required.

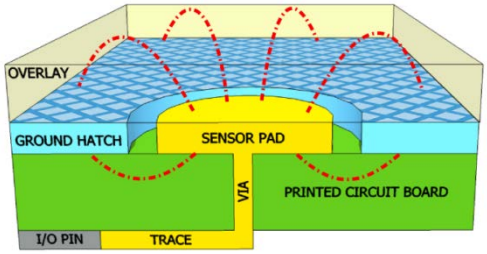
Figure 2-2. PSoC Device, Sensors, and External Capacitors



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.

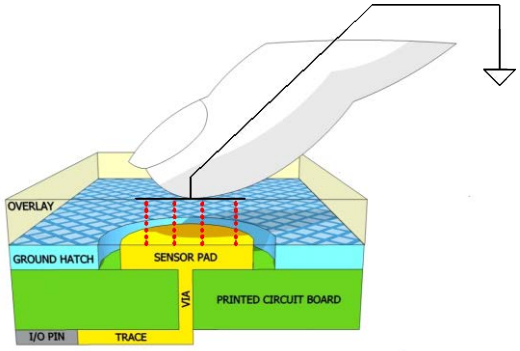
Figure 2-3 shows how a sensor GPIO pin is connected to a sensor pad by traces and vias. Typically, a ground hatch surrounds the sensor pad to isolate it from other sensors and traces. Although Figure 2-3 shows some field lines around the sensor pad, the actual electric field distribution is very complex.

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.

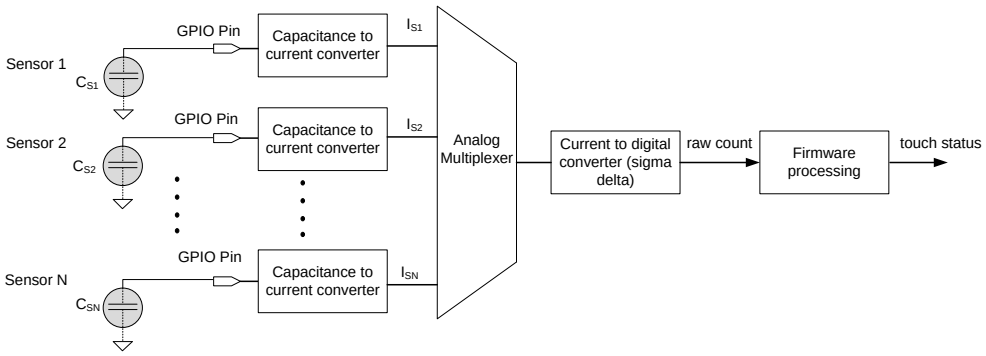
PSoC converts the capacitance C_S into equivalent digital counts called raw counts. Since a finger touch increases the total capacitance of the sensor pin, an increase in the raw counts indicates a finger touch.

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.

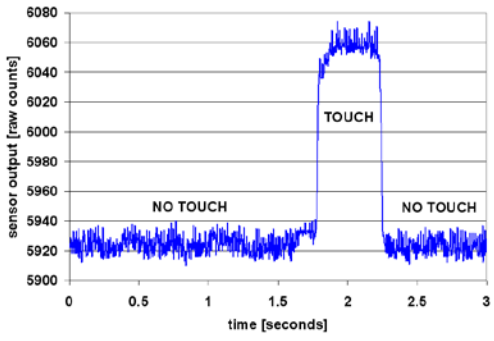
The output count of the current to digital converter, known as **raw count**, is a digital value that is proportional to the sensor capacitance:

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

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.

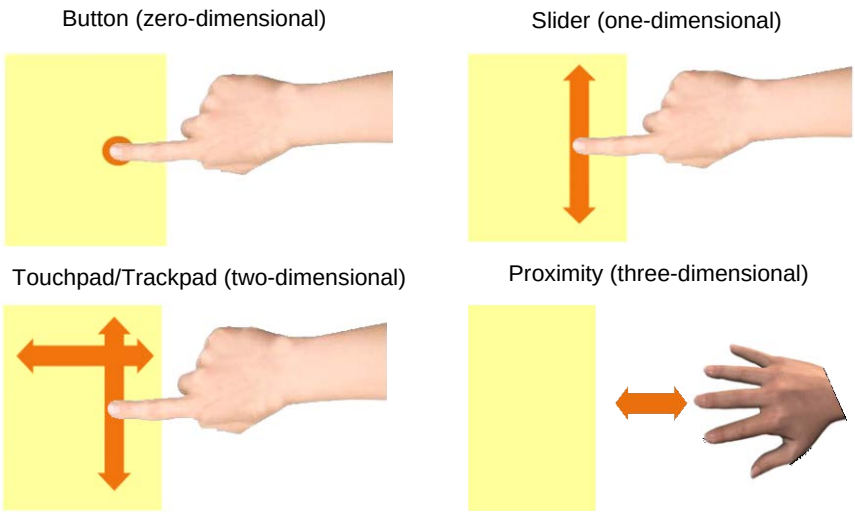
Figure 2-6. Raw Count Versus Time



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. Types of Widgets

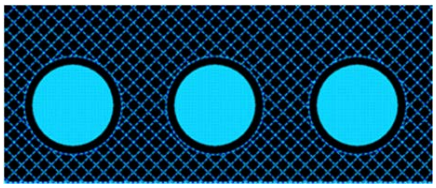


2.3.1 Buttons (Zero-Dimensional)

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.

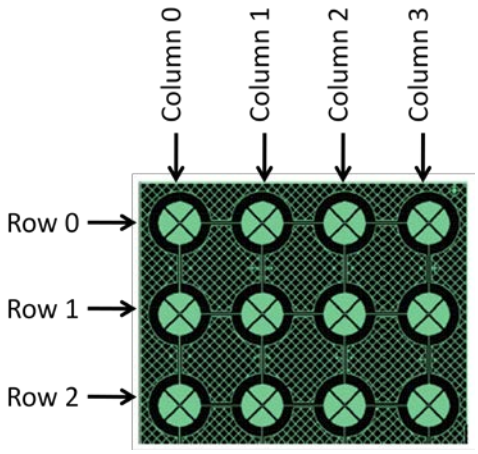
A simple CapSense button consists of a circular copper pad connected to a PSoC GPIO using PCB traces, as Figure 2-8 shows. The button is surrounded by grounded copper hatch to isolate it from other buttons and traces. A circular gap separates the button pad and the ground hatch. Each button requires 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 7 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 to be 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.

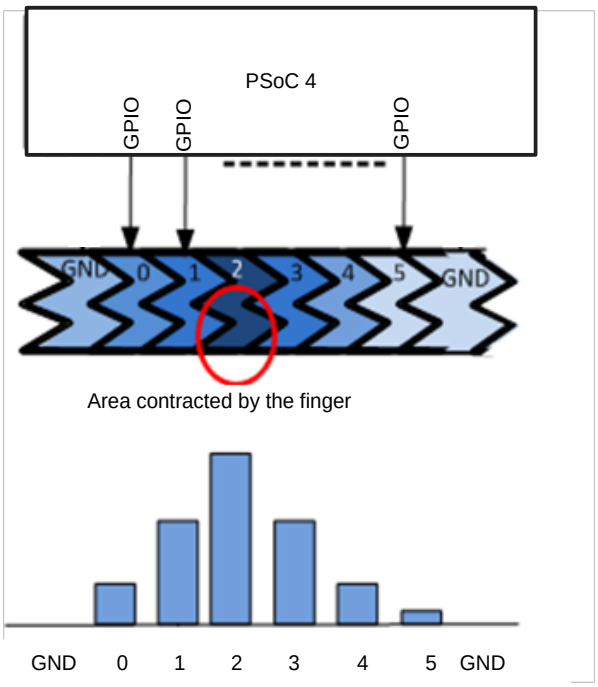
2.3.2 Sliders (One-Dimensional)

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.

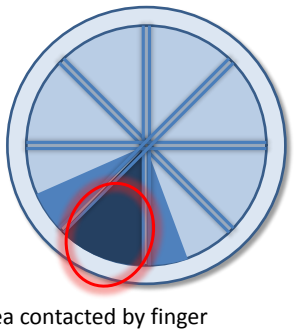
In a linear slider, the segments are arranged inline, as Figure 2-10 shows. Each slider segment connects to a PSoC GPIO. A zigzag pattern (double chevron) is recommended for slider segments. This layout ensures that when a segment is touched, the adjacent segments are also partially touched, which aids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

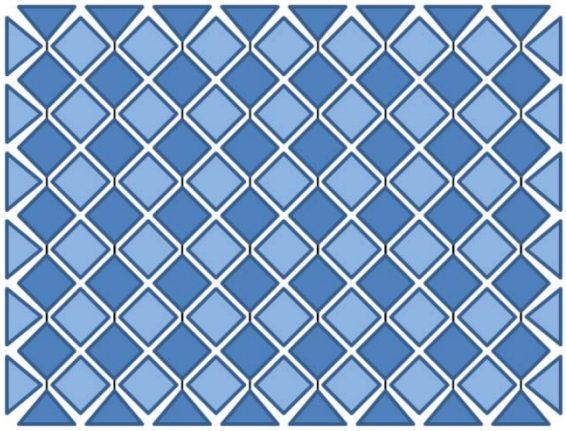
Figure 2-11. Radial Slider



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

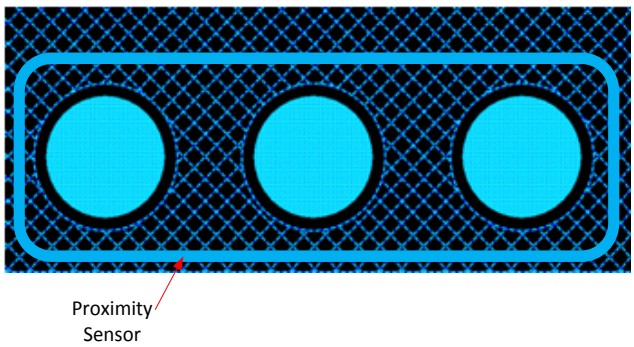


2.3.4 Proximity (Three-dimensional)

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.

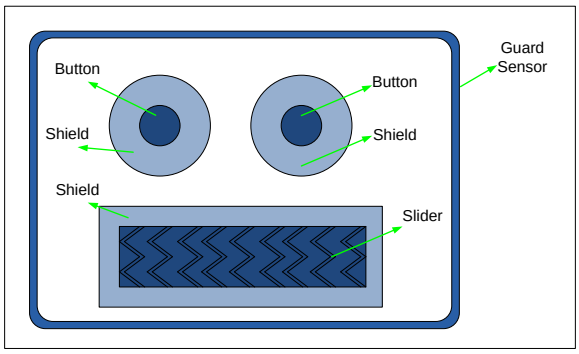
2.4 Shield Electrode and Guard Sensor

In a CapSense design, false touch sensing may happen due to the presence of water film or droplets on the overlay. If your application requires tolerance to water droplets and moisture, you should use a shield electrode in your design. If your application also requires tolerance to water flow on the touch surface, you should use a guard sensor together with a shield electrode.

A shield electrode is a copper pad or hatch in the PCB that surrounds the sensors, as Figure 2-14 shows. Each shield electrode requires one GPIO.

Since each sensor is connected to a switched capacitance circuit (see Capacitive Touch Sensing Method), the resulting switching signal across the sensor is similar to a square wave. The shield GPIO drives the shield electrode with a replica of this sensor switching signal. Since the shield electrode and sensors are driven with the same signals, the potential difference between them is zero. Hence any capacitance between the sensors and the shield electrode cannot cause a charge transfer. Therefore any water film or droplets that are present partially on both the sensor and shield area do not change the sensor capacitance, allowing CapSense to work in the presence of water film or droplets.

Figure 2-14. Shield Electrode and Guard Sensor

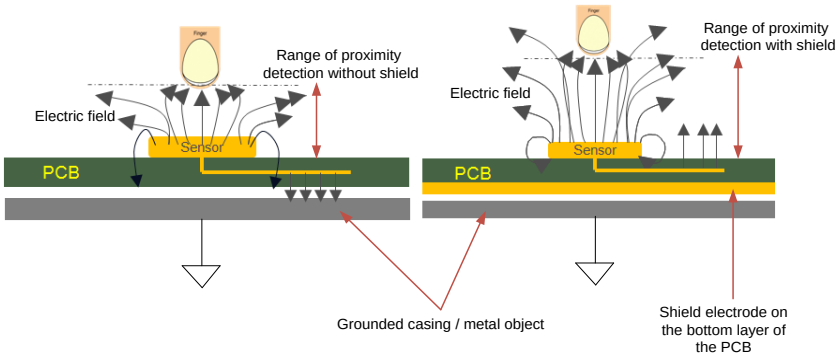


A guard sensor is used to detect the presence of water on the entire surface. CapSense scans the guard sensor in the same way as the other sensors. However, the guard sensor surrounds the entire touch sensing area, as Figure 2-14 shows. When water is present on the guard sensor, it becomes active and it disables the other sensors in firmware to avoid false touch detection.

The section Component Configuration explains how to enable the shield electrode and guard sensors. For a detailed explanation of shield electrode and guard sensor construction, see PCB Layout Guidelines. In addition to a dedicated shield electrode, you can use sensors that are not being scanned as part of the shield.

Since the shield electrode is kept at the same potential as the sensors, the electric field coupling between the sensor and its environment is reduced. Hence the shield electrode also reduces the effective parasitic capacitance of the sensors. In proximity sensing, use a shield electrode on the bottom side of the PCB to increase detection range, as Figure 2-15 shows.

Figure 2-15. Using a Shield to Increase Proximity Detection Range



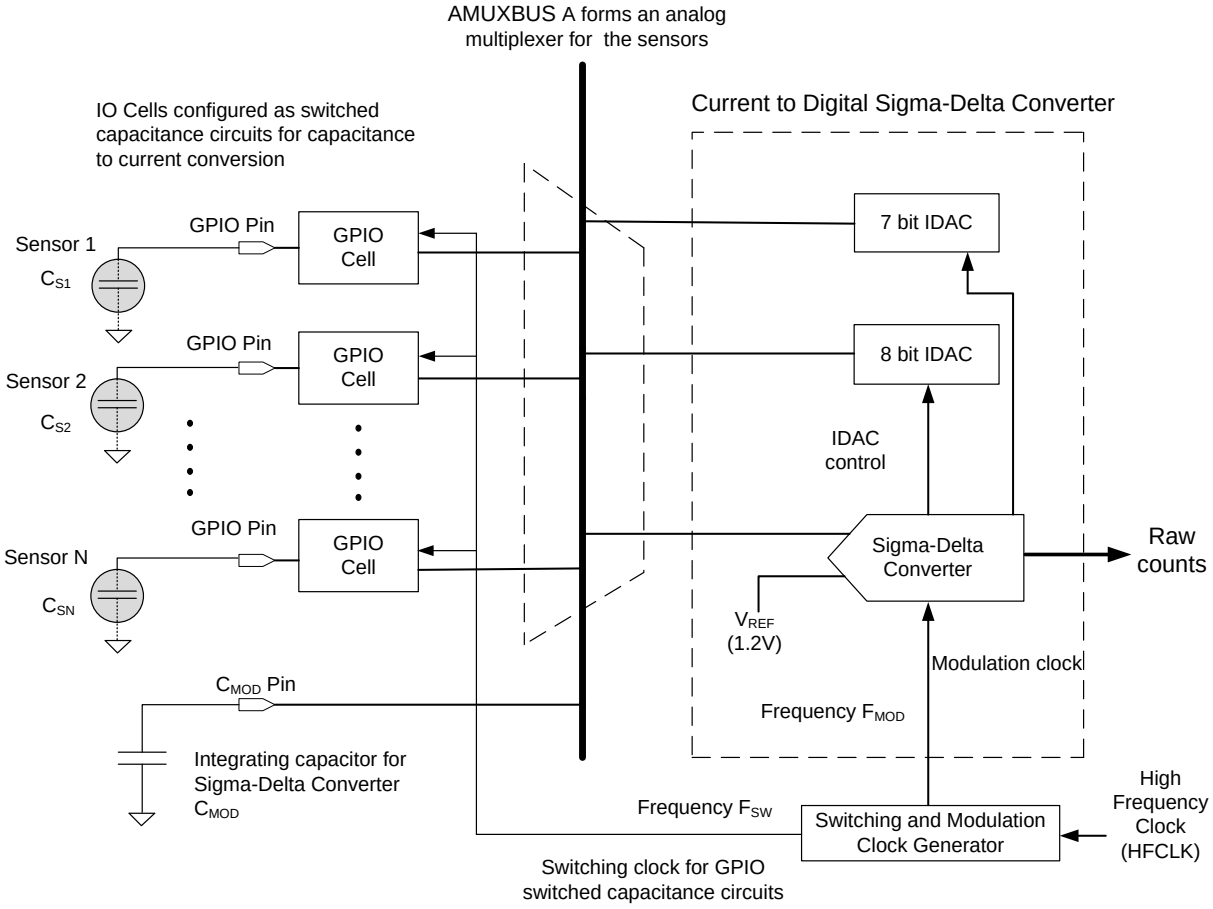
3. PSoC 4 CapSense

This chapter explains in detail how CapSense CSD is implemented in the PSoC 4 device. See Capacitive Touch Sensing Method to understand the basic principles of CapSense CSD. A basic knowledge of the PSoC 4 device architecture is a prerequisite for this chapter. If you are new to PSoC 4, refer to AN79953, Getting Started with PSoC 4.

3.1 CapSense CSD Sensing

Figure 3-1 shows the block diagram of the PSoC 4 CapSense block, which scans the CapSense sensors.

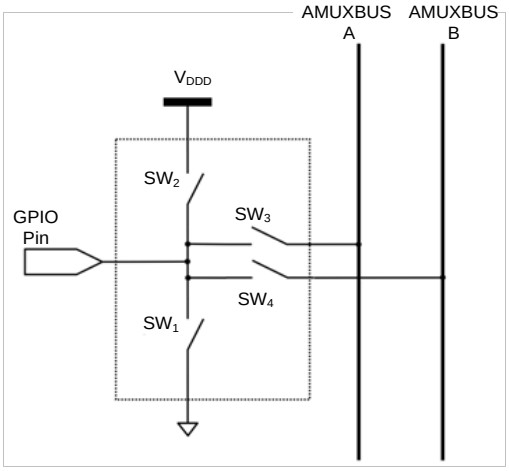
Figure 3-1. PSoC 4 CapSense CSD Sensing



3.1.1 GPIO Cell Capacitance to Current Converter

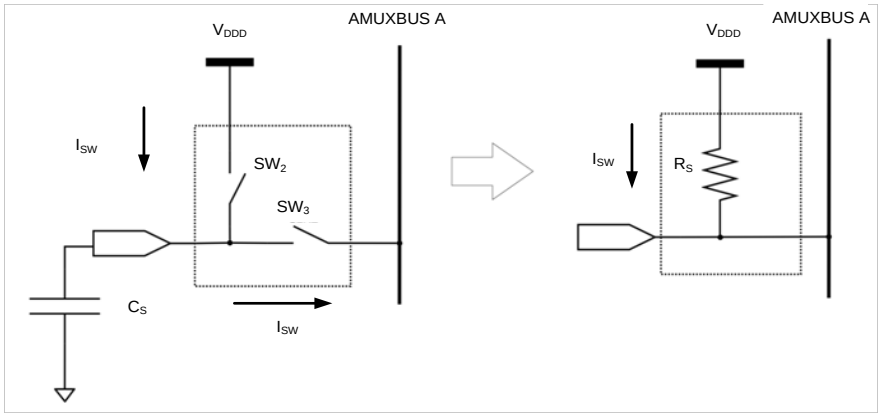
In the CapSense CSD system, the GPIO cells are configured as switched capacitance circuits that convert the sensor capacitances to equivalent currents. Figure 3-2 shows a simplified diagram of the PSoC 4 GPIO cell structure.

Figure 3-2. PSoC 4 GPIO Cell



PSoC 4 has 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. 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 17) 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:

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

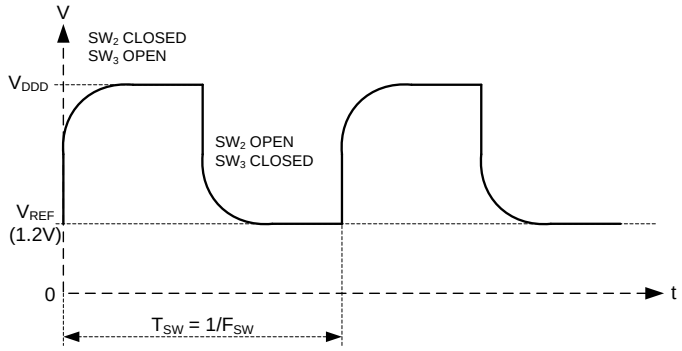
Where:

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.

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

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

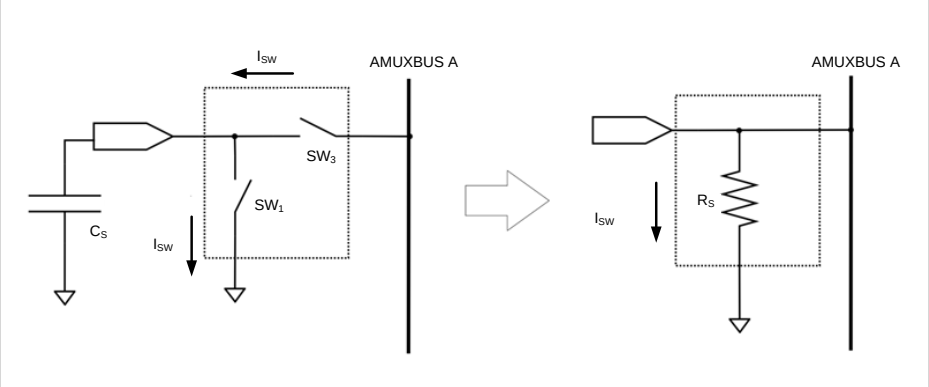
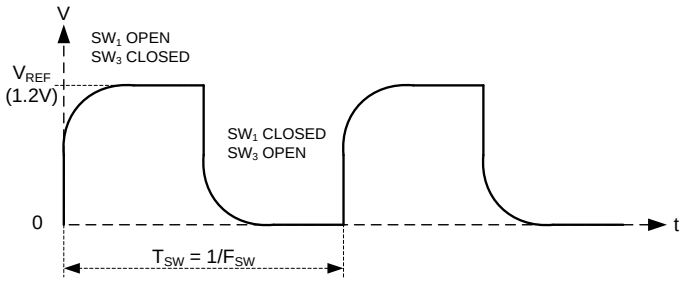


Figure 3-6. Voltage Across Sensor Capacitance



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 Switching Clock Generator

This block generates the switching clock F_{SW} from the high frequency clock (HFCLK), as Figure 3-1 on page 17 shows. The switching clock is required for the GPIO cell switched capacitance circuits. The switching clock generator output has 3 options: direct, 8-bit pseudo random sequence (PRS), and 12-bit PRS. You can set the desired switching frequency by selecting a clock divider parameter of the switching clock generator. This clock divider parameter is known as the analog switch divider. If the “direct” output is selected, the value of the generated switching clock frequency F_{SW} is

$$F_{SW} = \frac{\text{HFCLK}}{2\ \text{Analog Switch Divider}} \qquad (3-4)$$

You can also select one of the PRS outputs to lower the Electro Magnetic Interference (EMI) effect – it averages the switching frequency over a wide range. If PRS output is selected, Equation 3-4 gives the average value of F_{SW}. Equations 3-5 and 3-6 give the maximum and minimum frequencies.

$$F_{SW}\ (\text{maximum}) = \frac{\text{HFCLK}}{\text{Analog Switch Divider}} \qquad (3-5)$$

$$F_{SW}\ (\text{minimum}) = \frac{\text{HFCLK}}{2\ \text{m Analog Switch Divider}} \qquad (3-6)$$

Where m is the resolution of the PRS (8 or 12 bits).

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 modulation switch divider and two current sourcing / sinking digital to analog converters (IDACs), as Figure 3-1 on page 17 shows.

The Sigma Delta modulator controls the current of one of the IDACs in an on/off manner. This IDAC is known as the compensation IDAC. The other IDAC, known as the baselining 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 8 bit IDAC is the compensation IDAC; the baselining IDAC (7 bit IDAC) is always OFF.
- In the dual IDAC mode, the 7 bit IDAC is the compensation IDAC; the baselining IDAC (8 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 17 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: If the switched capacitor circuit sinks current from AMUXBUS A, 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 compensation 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_{COMP} is the value of the compensation IDAC current, the approximate value of raw count in IDAC sourcing mode is given by Equation 3-7.

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

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

$$\text{raw count} = 2^N \frac{(V_{DD} - V_{REF})\ F_{SW}}{I_{COMP}}\ C_S \qquad (3-8)$$

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_{COMP}, I_{BASE}, 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 baselining IDAC is always ON. If I_{BASE} is the baselining IDAC current, the equation for the raw count in IDAC sourcing mode is:

$$\text{raw count} = 2^N \frac{V_{REF}\ F_{SW}}{I_{COMP}}\ C_S - 2^N \frac{I_{BASE}}{I_{COMP}} \qquad (3-9)$$

Raw count in IDAC sinking mode is given by equation 3-10.

$$\text{raw count} = 2^N \frac{(V_{DD} - V_{REF})\ F_{SW}}{I_{COMP}}\ C_S - 2^N \frac{I_{BASE}}{I_{COMP}} \qquad (3-10)$$

Note that raw count values are always positive.

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 17 shows. The AMUXBUS A and the GPIO cell switches (see SW_3 in Figure 3-3 on page 18) form this analog multiplexer. AMUXBUS A connects to all GPIOs, so you can use any PSoC 4 GPIO for CSD sensing. AMUXBUS A also connects the integrating capacitor C_{MOD} to the Sigma Delta converter circuit.

3.2 CapSense CSD Shielding

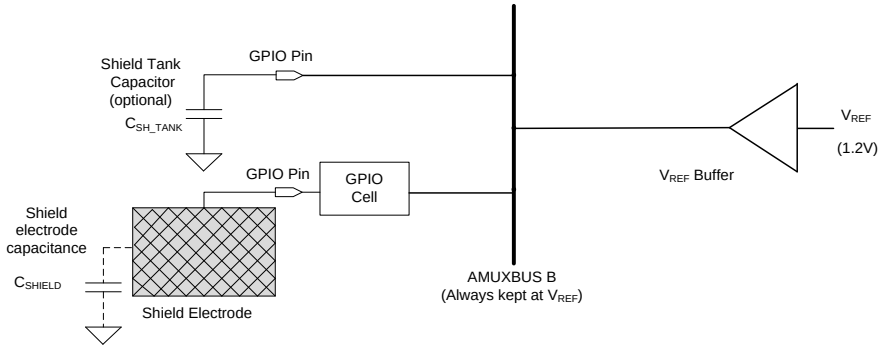
PSoC 4 CapSense supports shield electrodes for waterproofing and proximity sensing. See Shield Electrode and Guard Sensor for details. The shield electrode is always kept at the same potential as the sensors. PSoC 4 CapSense has a shielding circuit that drives the shield electrode with a replica of the 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.

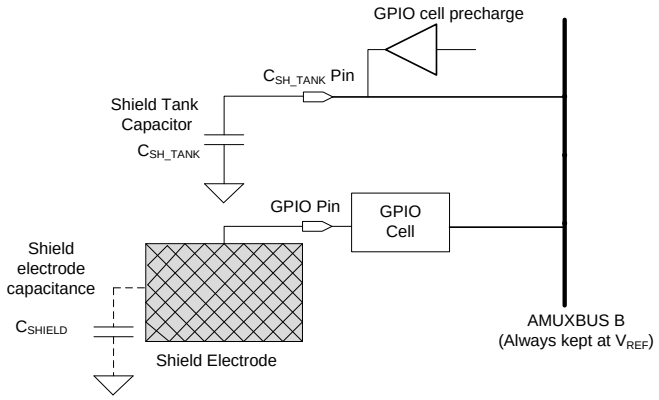
- Shield driving using V_{REF} buffer: In this configuration, a voltage buffer is used to drive AMUXBUS B to V_{REF}, as Figure 3-7 shows. Select this configuration if the shield electrode capacitance is less than 200 pF. An external 10 nF C_{SH_TANK} capacitor is recommended to reduce switching transients.

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}.

Figure 3-8. Shield Driving Using GPIO Precharge



You should select this configuration if the shield electrode capacitance is greater than 200 pF. This GPIO cell precharge capability is available only on a fixed C_{SH_TANK} pin. See the device pinout in the PSoC 4 datasheet for details.

3.2.1 C_{MOD} Precharge

When the CapSense hardware is enabled for the first time, the voltage across C_{MOD} starts at zero. Then the Sigma Delta converter slowly charges the C_{MOD} to V_{REF}. The charging current is supplied by the IDACs in the IDAC sourcing mode and it is supplied by the sensor switched capacitance circuit in IDAC sinking mode. However this is a slow process since C_{MOD} is a relatively large capacitor.

Precharging of C_{MOD} is the process of quickly initializing the voltage across C_{MOD} to V_{REF}. Precharging is used to reduce the time required for the Sigma Delta converter to start its operation. There are two options for precharging C_{MOD}.

- Precharge using V_{REF} buffer: When the shield is enabled, the output of the V_{REF} buffer is always connected to AMUXBUS B, as Figure 3-8 shows. To precharge using the V_{REF} buffer, C_{MOD} is initially connected to AMUXBUS B. After the precharging process, C_{MOD} is connected to AMUXBUS A for normal Sigma Delta operation.

  When the shield is disabled, the output of the V_{REF} buffer is always connected to AMUXBUS A for precharging, and disconnected afterwards.

- Precharge using GPIO cell: In this configuration, a special GPIO cell charges the C_{MOD} capacitor to V_{REF}. This GPIO cell precharge capability is available only on a fixed C_{MOD} pin. See the device pinout in the PSoC 4 datasheet for details.

  Precharge using a GPIO cell is faster than the precharge using the V_{REF} buffer. Therefore GPIO precharge is the recommended precharge configuration. However, if you don't need a fast initialization of the CapSense, you can use V_{REF} buffer precharge. In this mode, you can connect C_{MOD} to any GPIO.

4. CapSense Design and Development Tools

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

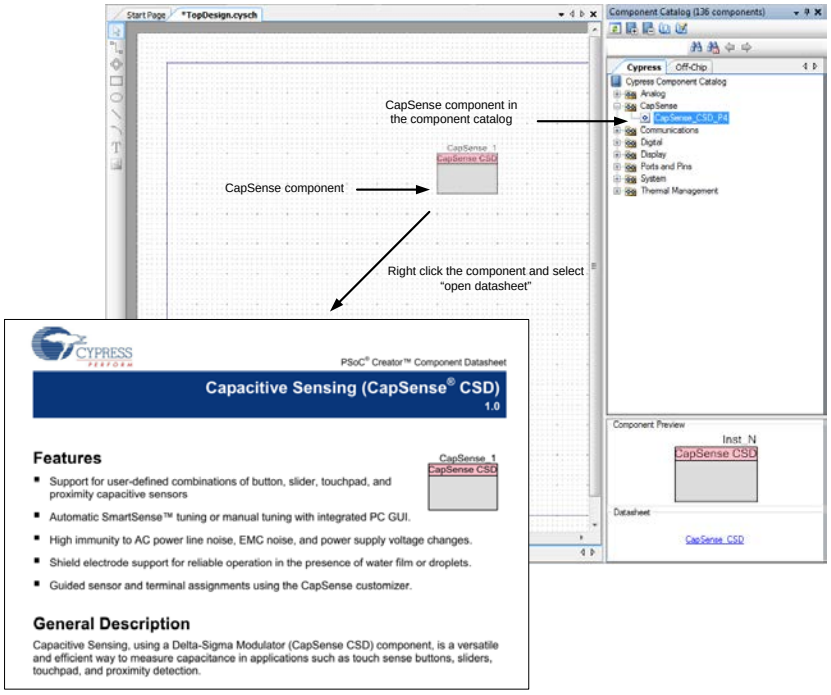
4.1 PSoC Creator

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 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 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



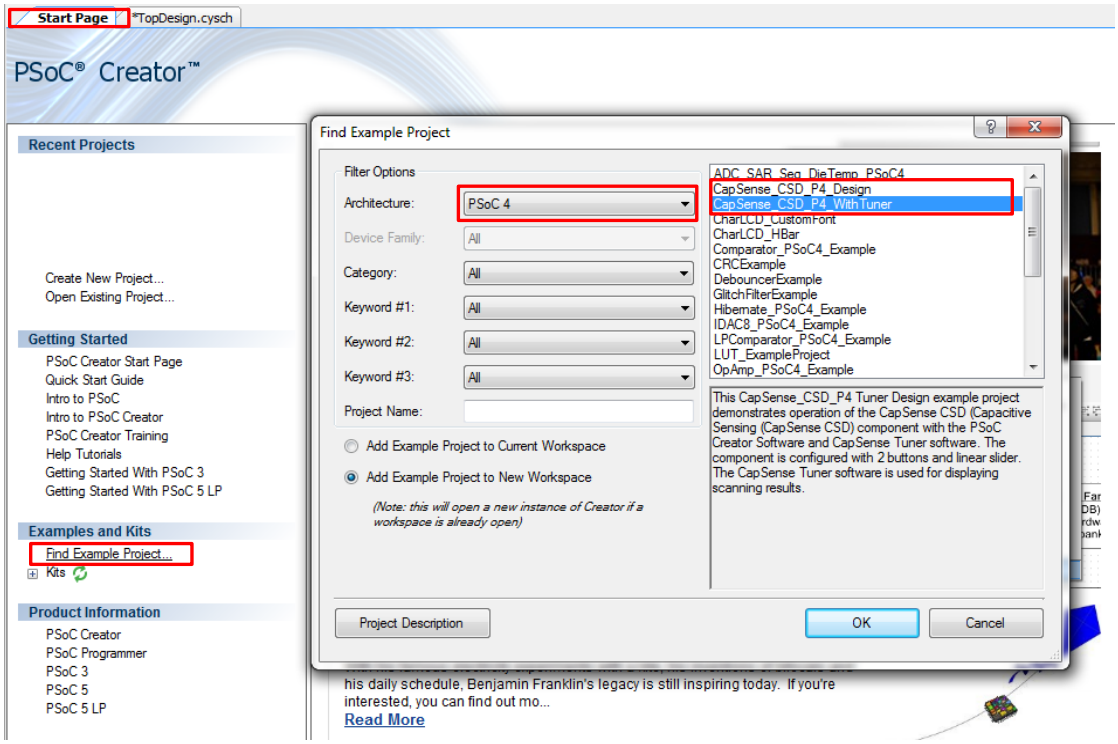
Each Component has an associated datasheet that explains details about the Component. To open the Component datasheet, right-click on the Component and select “Open Datasheet”.

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 PSoC 4 CapSense example project, go to the PSoC Creator Start Page, click “Find Example Project”, and select PSoC 4 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 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

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 CapSense. SmartSense is a firmware algorithm that automatically sets all parameters to optimum values. It reduces design cycle time and provides stable performance across PCB variations. SmartSense requires additional RAM and CPU resources.

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

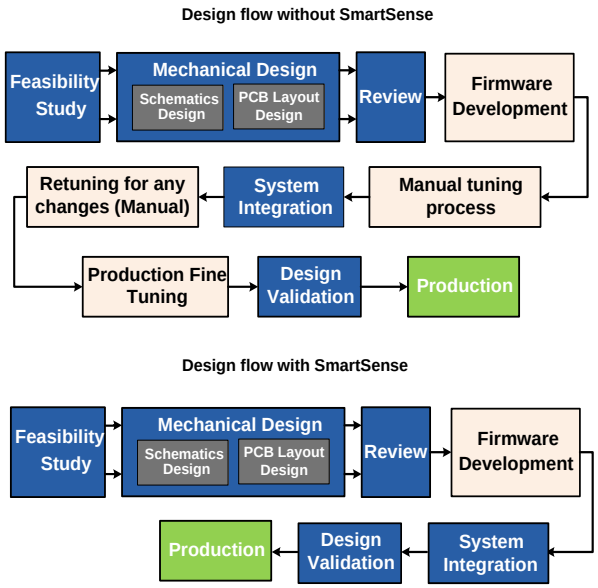
After tuning and finalizing all CapSense parameters, you can turn off the tuning. In this mode, all tuning parameters are stored in flash.

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. Also, 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 With and Without SmartSense

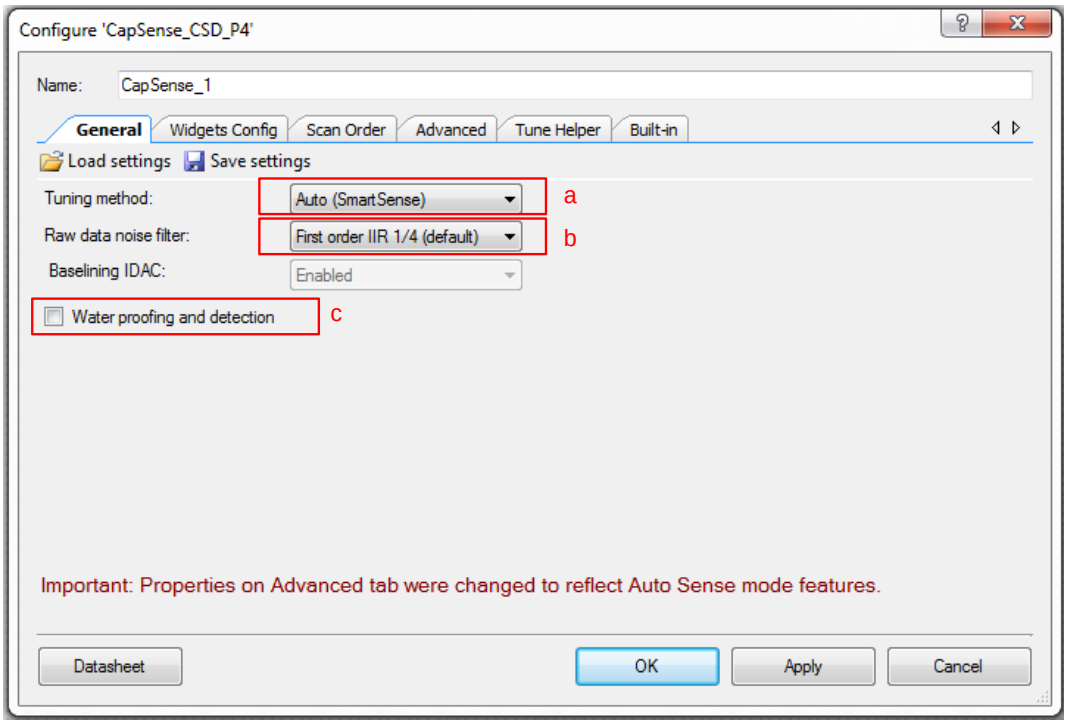


- Performance is Independent of PCB Variations: The parasitic capacitance of individual sensors can vary due to process variations in PCB manufacturing, or vendor-to-vendor variation in a multi sourced supply chain. If there is significant variation in parasitic capacitance C_P across product batches,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

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. **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 27 explains the available filters and their applications.

c. **Waterproofing and detection:** Enable this option if you are using a shield electrode or guard sensor for water proofing. Disable otherwise. See Shield Electrode and Guard Sensor for details.

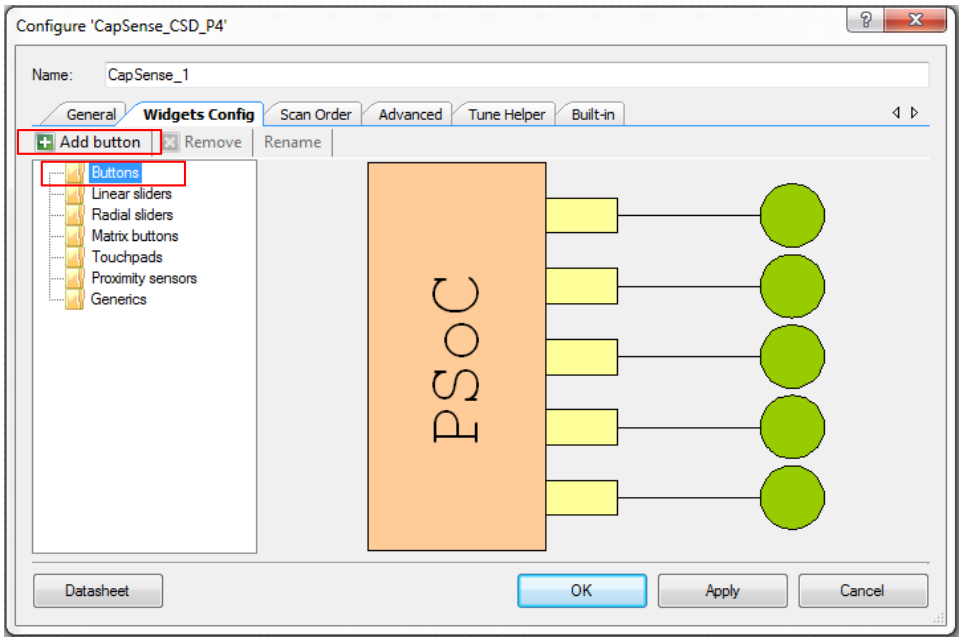
Table 5-1. Raw Data Noise Filters

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

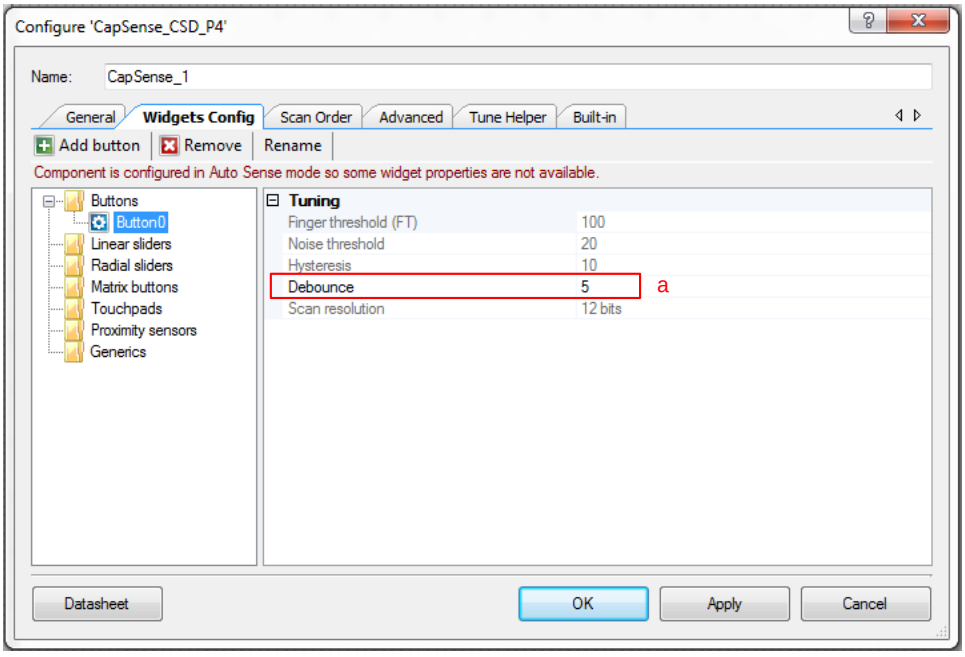
The “Widgets Config” tab allows you to add and configure CapSense widgets, as Figure 5-3 shows. See CapSense Widgets for details on each widget type. To add a widget, select the type of widget and click “Add”.

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 29 and Figure 5-6 on page 30.

Figure 5-4. Configuration of a Button



a. **Debounce:** (see Figure 5-4 on page 28; 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.

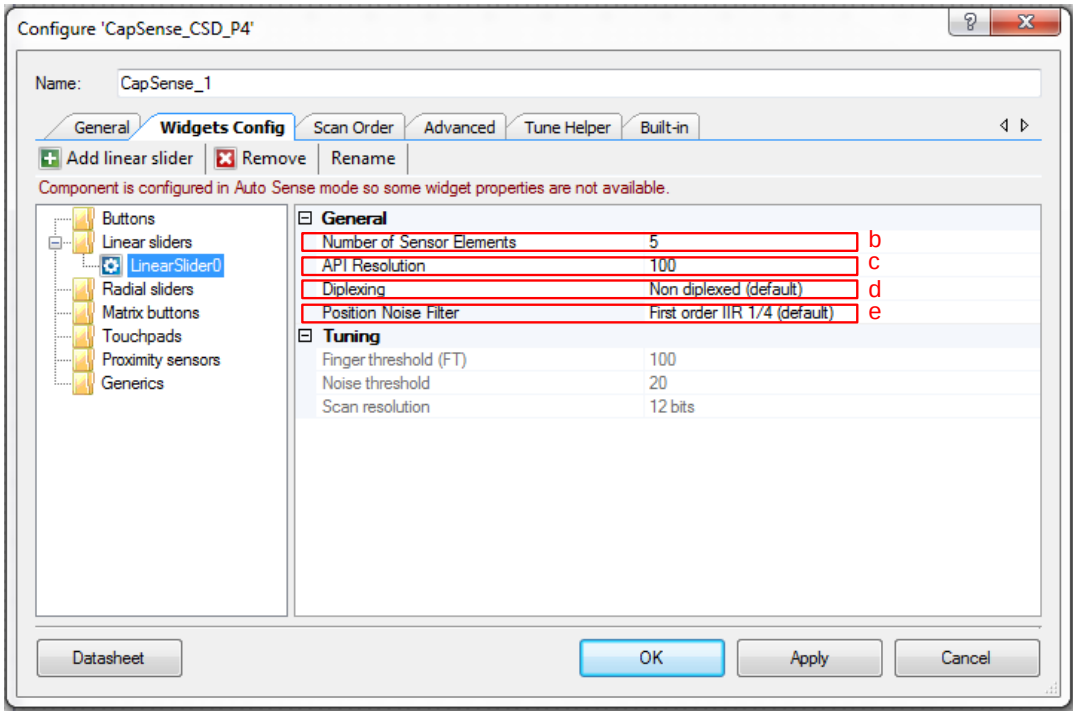
c. **API resolution:** (see Figure 5-5; available for Linear Sliders, Radial Sliders and Touchpads). This parameter defines the number of discrete finger positions that a slider or touchpad must resolve. This parameter is different from the resolution of the Sigma Delta converter.

   Higher API resolution results in a smoother slider. Set this value 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 8 GPIO pins. See the CapSense Component datasheet for details.

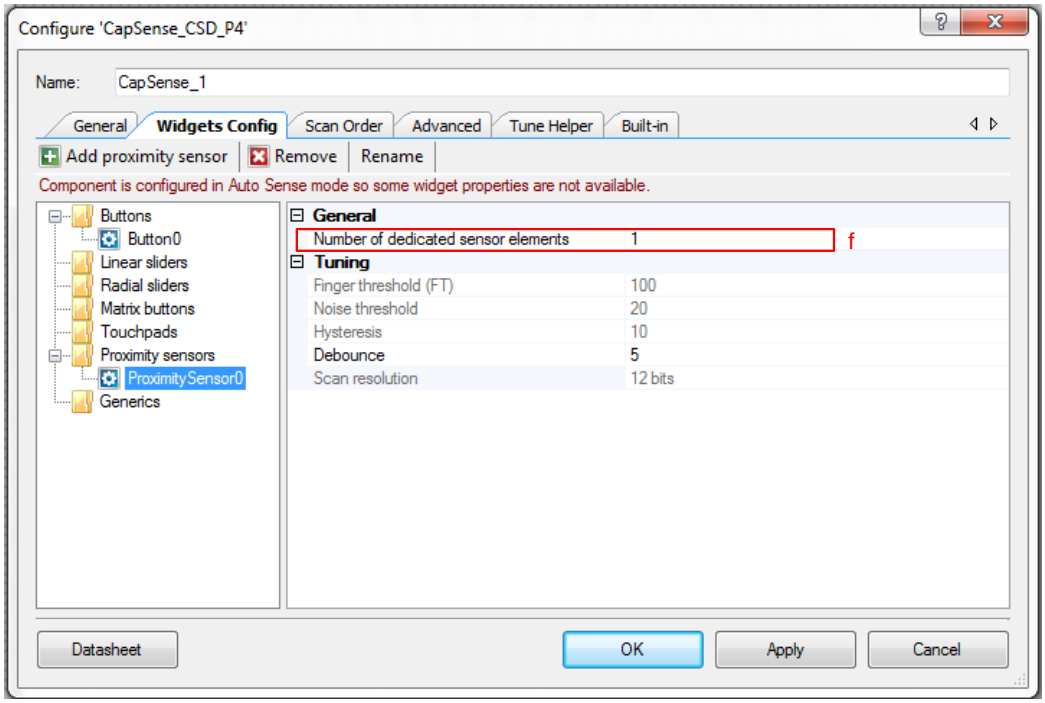
e. **Position Noise filter:** (see Figure 5-5; available for Linear Sliders, Radial Sliders and Touchpads). This parameter selects the type of firmware noise filter used to remove noise from the calculated finger position. The available filters and their applications are explained in Table 5-1 on page 27.

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.

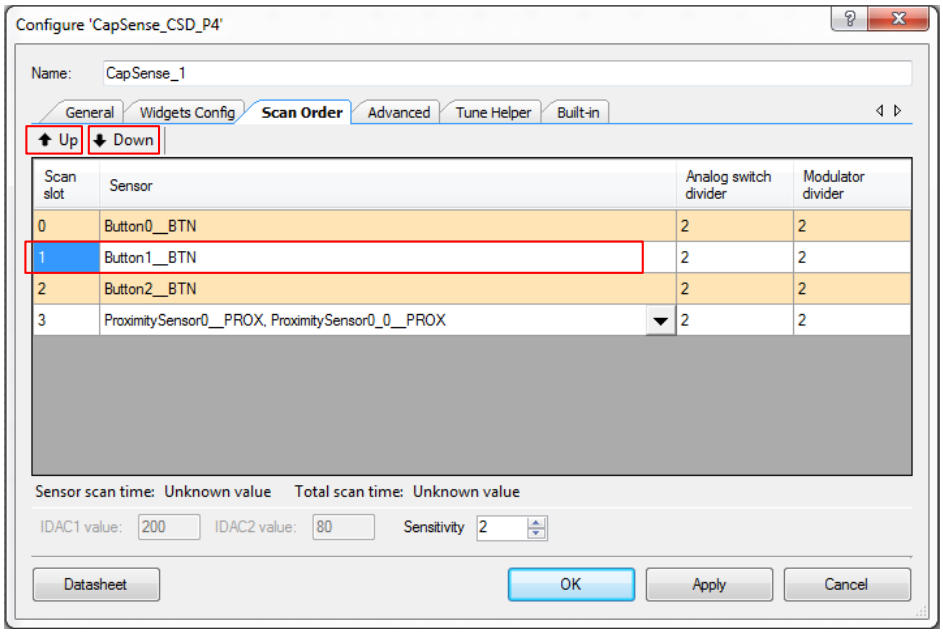
Figure 5-6. Configuration of a Proximity Sensor



5.1.3 Scan Order

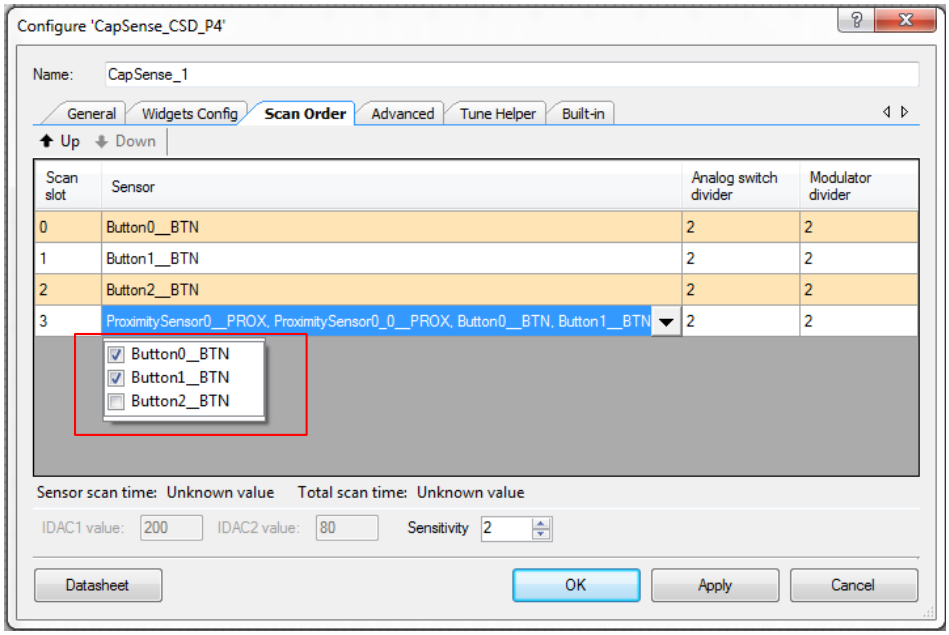
Generally, CapSense performance is not impacted by the order in which the sensors are scanned. However, if you want to change scan order, you can use this tab, as Figure 5-7 shows. To change the scan order, select a sensor and click “Up” or “Down”.

Figure 5-7. Scan Order



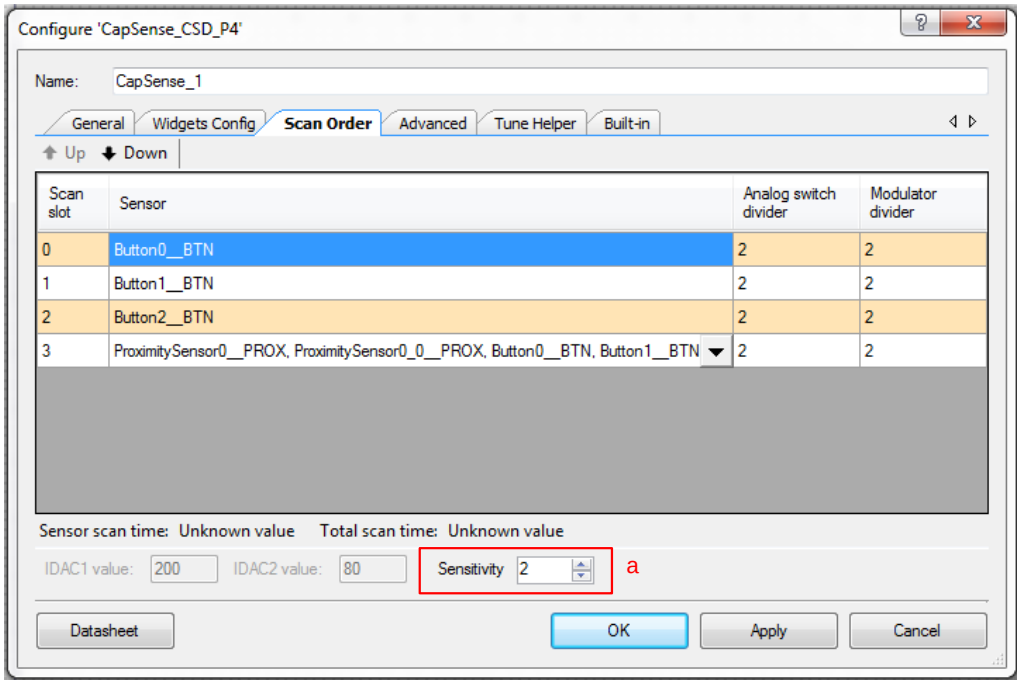
You can combine other sensors into a proximity sensor by using the drop-down menu on the Scan Order tab, as Figure 5-8 shows. Combining other sensors with the proximity sensor does not affect their operation, however, the resulting proximity sensor may have a high C_P and hence a high scan time. Therefore,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

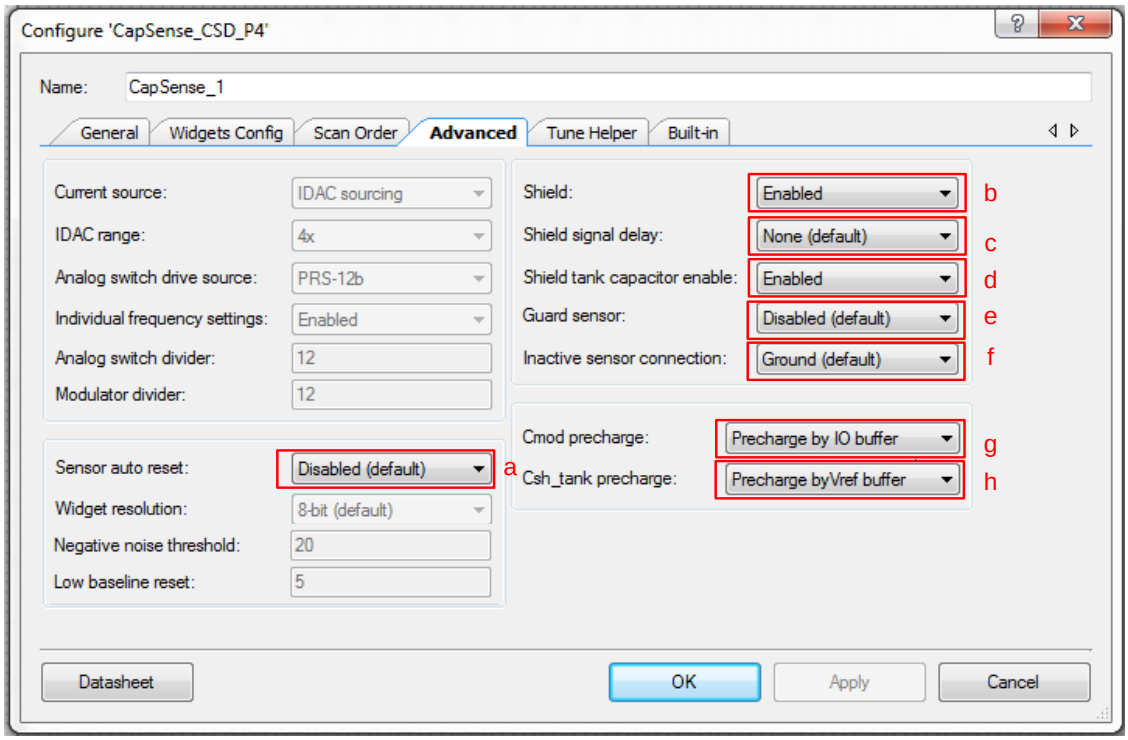
$$C_F = 0.1\ \text{pF} \times \text{sensitivity} \qquad (6-1)$$

If you don't know the value of C_F, set the sensitivity at 1 and check the sensor performance. If the sensor becomes active even before the finger touches it, increase the sensitivity value.

5.1.3.1 Advanced Settings

Advanced settings are available on the Advanced tab, as Figure 5-10 shows.

Figure 5-10. Advanced Settings



a. 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 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.

b. Shield: You should enable the shield electrode if waterproofing is required. See Shield Electrode and Guard Sensor for details.

c. Shield 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. If they are not aligned, use this option to add delay to the shield signal to align the two signals.

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 waterproofing. See Shield Electrode and Guard Sensor for details. Enable this option if you are using a guard sensor in your design.

f. 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. If you have a shield electrode in your design, you can connect the inactive sensor to shield for reduced parasitic capacitance and increased waterproofing ability.

g. C_{MOD} precharge: This setting selects the precharge configuration of C_{MOD}. “Precharge by IO buffer” is recommended because it is faster than ”Precharge by V_{REF} buffer”, see CMOD Precharge for details.

h. C_{SH_TANK} precharge: This option selects the precharge configuration of C_{SH_TANK}. You should select “Precharge by V_{REF} buffer” if the shield electrode capacitance is less than 200 pF. If the shield electrode capacitance is higher than 200 pF, 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 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 CapSense architecture is a prerequisite for this section. See Capacitive Touch Sensing Method and CapSense CSD Sensing to become familiar with PSoC 4 CapSense architecture. You can skip this section if you are not planning to use manual tuning in your design.

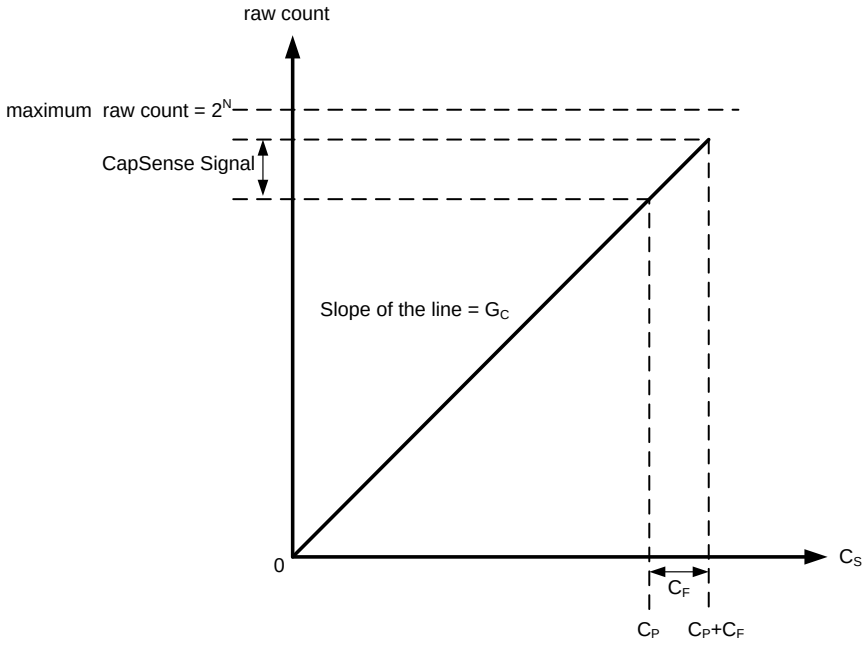
5.2.1.1 Conversion Gain and CapSense Signal

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 $2^N \frac{V_{REF} F_{SW}}{I_{COMP}}$ in IDAC sourcing mode, and $2^N \frac{(V_{DD} - V_{REF}) F_{SW}}{I_{COMP}}$ 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_{COMP}. Figure 5-11 shows a plot of raw count versus sensor capacitance.

Figure 5-11. Raw Count Versus Sensor Capacitance



The change in raw counts when a finger is placed on the sensor is called a CapSense **signal**. Figure 5-12 shows how the value of the signal changes with respect to the conversion gain.

Figure 5-12. Signal Values for Different Conversion Gains

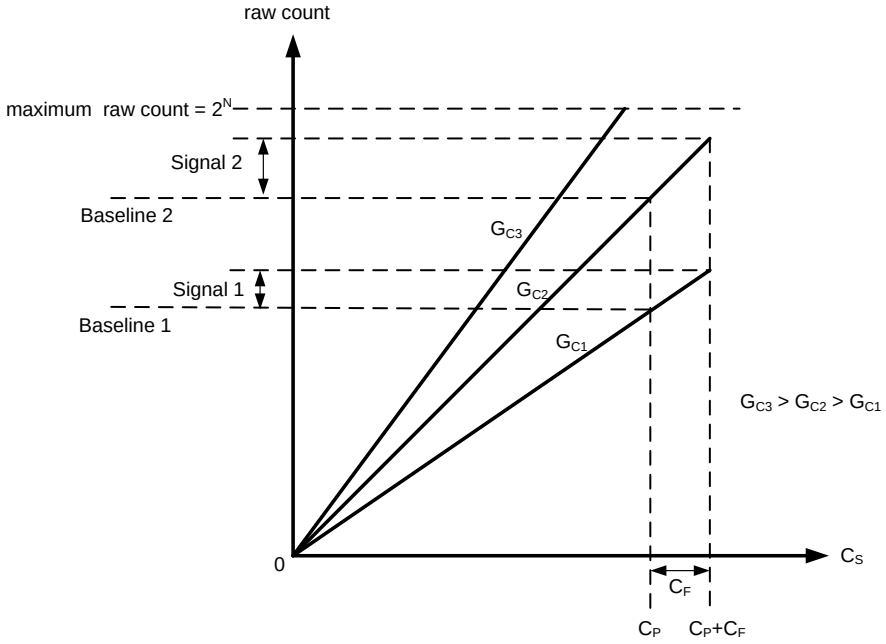
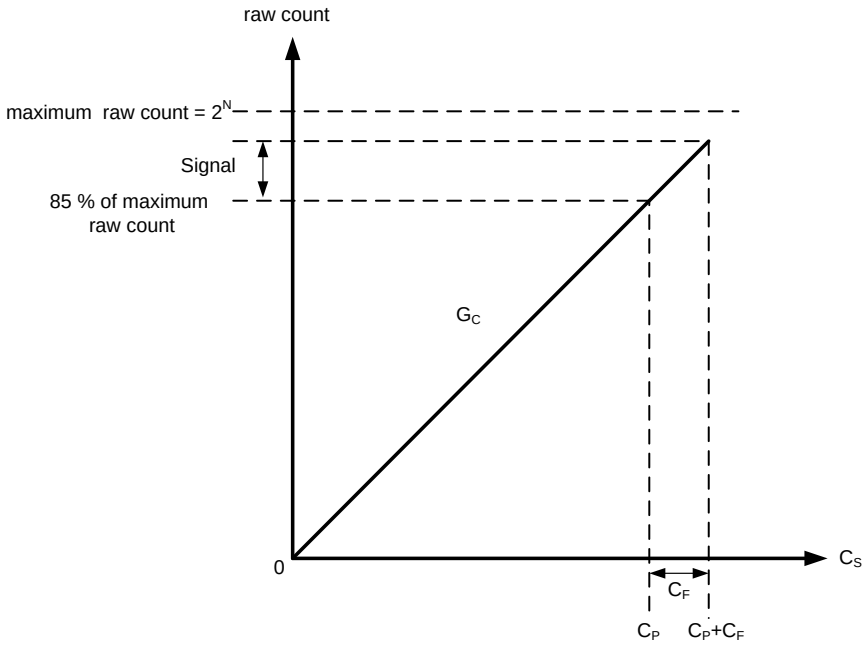


Figure 5-12 shows three plots corresponding to three conversion gain values G_{C3}, G_{C2}, and G_{C1}. An increase in the conversion gain results in higher signal value. But this increase in the conversion gain also moves the raw count corresponding to C_P towards the maximum value of raw count (2^N). 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 5-13 shows.

Figure 5-13. Recommended Tuning



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

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

See Equations 3-9 and 3-10 for details. Figure 5-14 shows a plot of raw count versus sensor capacitance in dual IDAC mode.

Figure 5-14. Dual IDAC mode

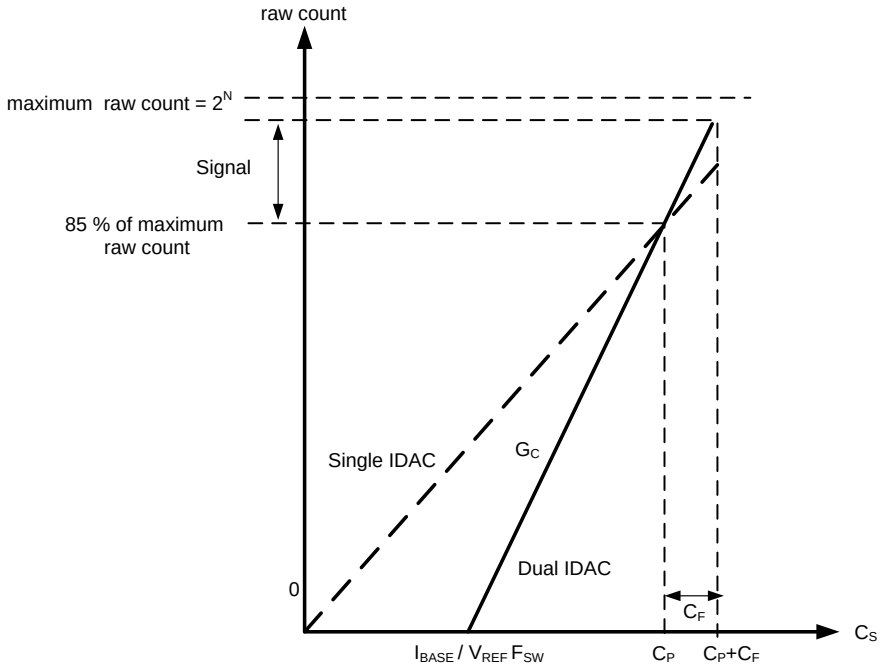


Figure 5-14 shows that dual IDAC mode yields higher signal than single IDAC mode. Increasing the value of the baselining IDAC increases the C_S axis intercept:

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

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 baselining IDAC value increases.

Note that this increase in signal may not proportionally increase the SNR. This because of the variations in noise counts when the baselining IDAC is used.

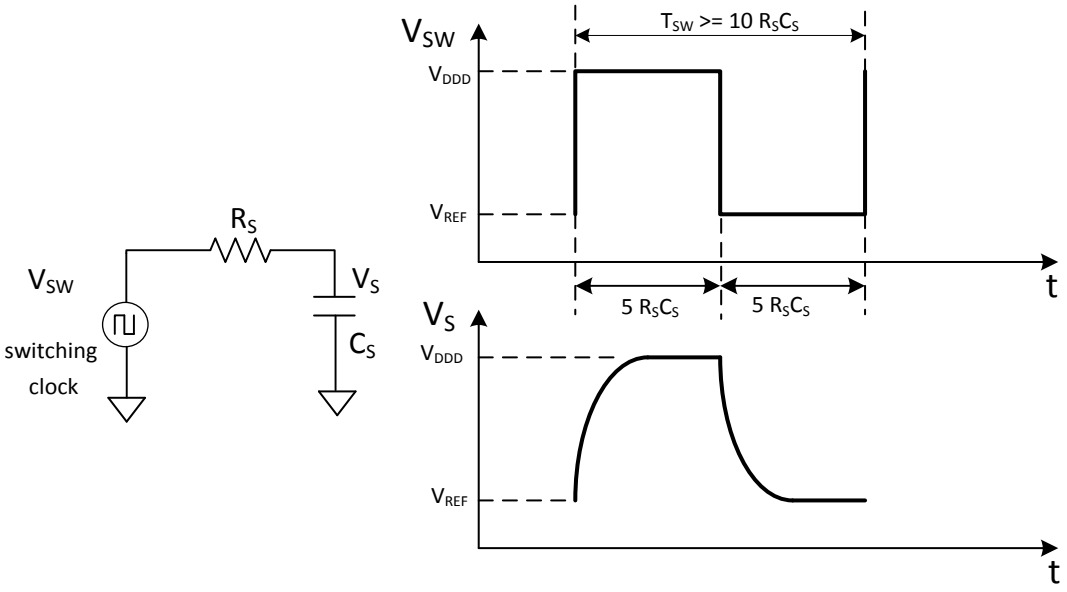
The dual IDAC mode complicates the tuning process because of the additional variable (I_{BASE}) involved. However, you can use this mode to increase the signal. The details are explained in Manual Tuning Process.

5.2.1.2 Switching Clock Selection

For a given resolution N, you can vary both F_{SW} and I_{COMP} to make the raw count corresponding to C_P equal to 85% of the maximum raw count value. But, 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 5-15 shows an equivalent circuit and resulting waveforms.

Figure 5-15. 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:

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

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

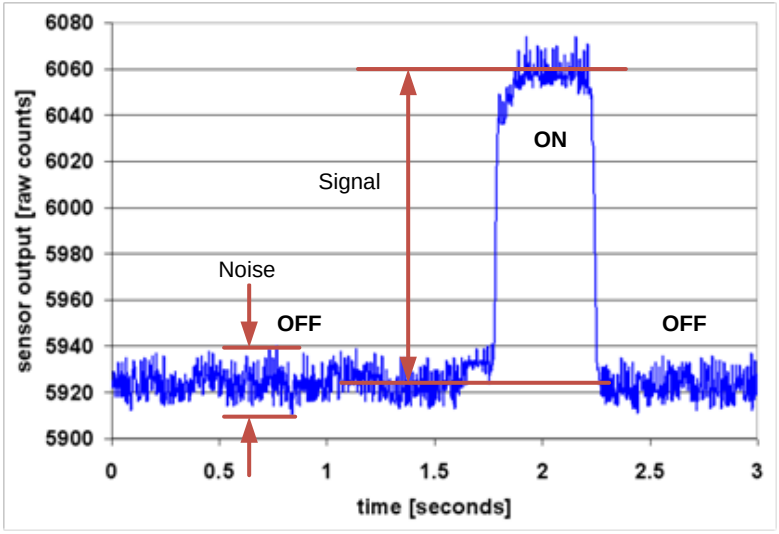
The typical value of the GPIO resistance is 500 Ω and the recommended external resistance is 560 Ω (see Series Resistance on CapSense Input Lines for details). Therefore, take the value of R_S as 1.06 kΩ when calculating the maximum switching frequency.

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 5-16 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 5-16. SNR



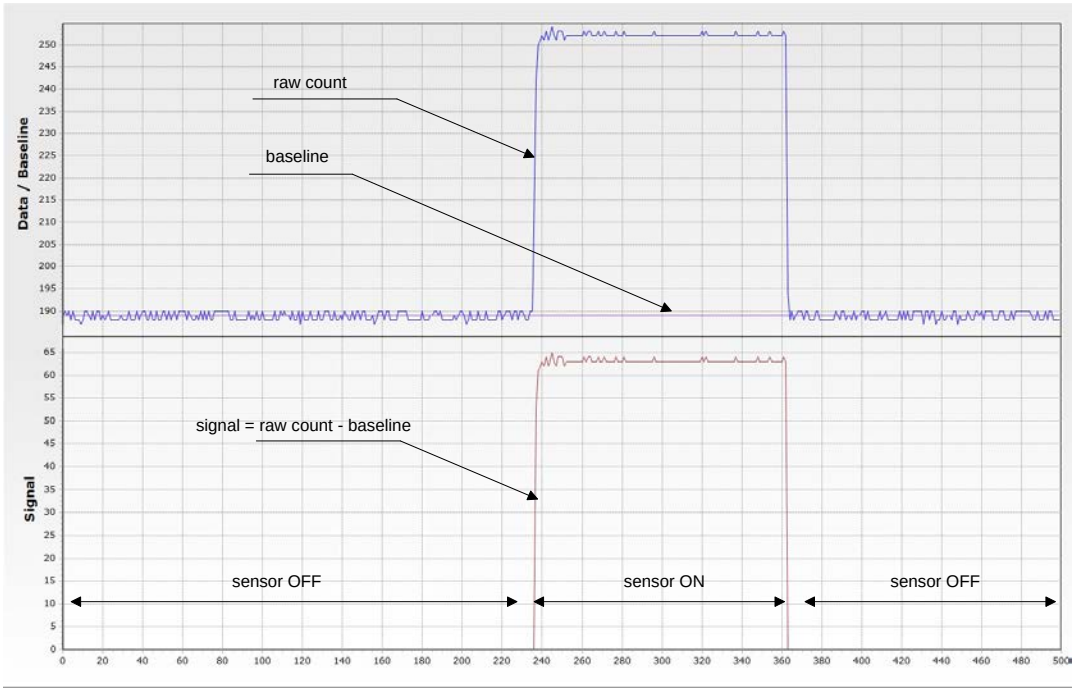
In this example, the average level of raw count in the absence of a touch is 5925 counts. When a finger is placed on the sensor, the average raw count increases to 6060 counts, therefore the signal is 6060 – 5925 = 135 counts. The minimum value of raw count in the OFF state is 5912 and the maximum value 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 5-17 shows the concept of raw count, baseline and signal.

Figure 5-17. Raw Count and Baseline



5.2.1.5 Tuning Parameters

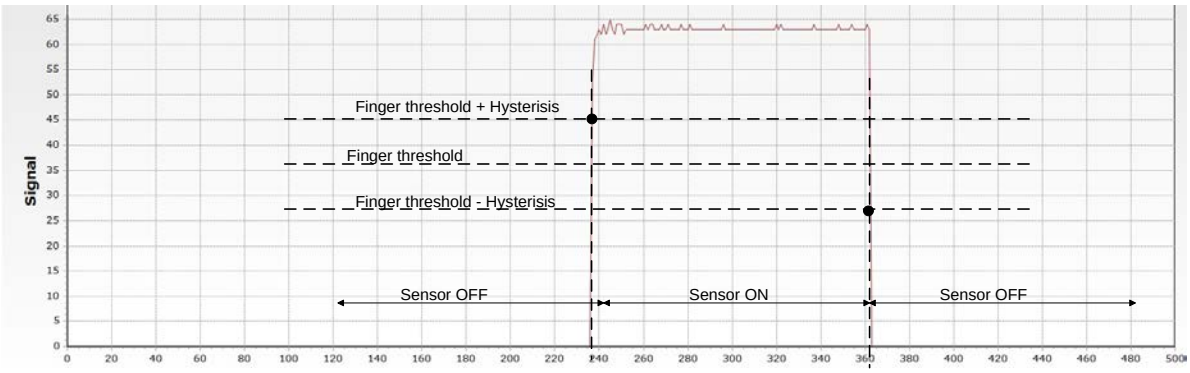
The CapSense Component uses several tuning parameters for proper operation of CapSense Hardware tuning parameters include CSD resolution N, IDAC values, and analog switch frequency F_{SW} – see Sigma Delta Converter for details. These are the software parameters.

- 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.

$$\text{Sensor State} = \begin{cases} \text{ON} & \text{if (Signal} \geq \text{Finger Threshold} + \text{Hysteris)} \\ \text{OFF} & \text{if (Signal} \leq \text{Finger Threshold} - \text{Hysteris)} \end{cases} \qquad (6-6)$$

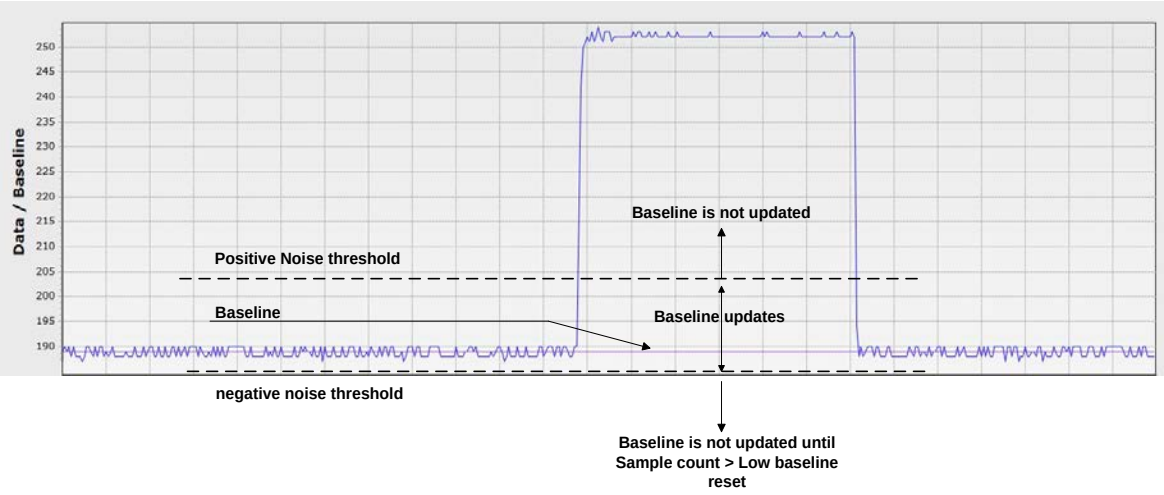
- Hysteresis: The hysteresis parameter is used along with the finger threshold parameter to determine the sensor state, as Equation 6-6 and Figure 5-18 show. Hysteresis provides immunity against noisy transitions of sensor state. The hysteresis parameter setting must be lower than the Finger Threshold parameter setting.

Figure 5-18. 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 5-19 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 5-19. Finger Threshold



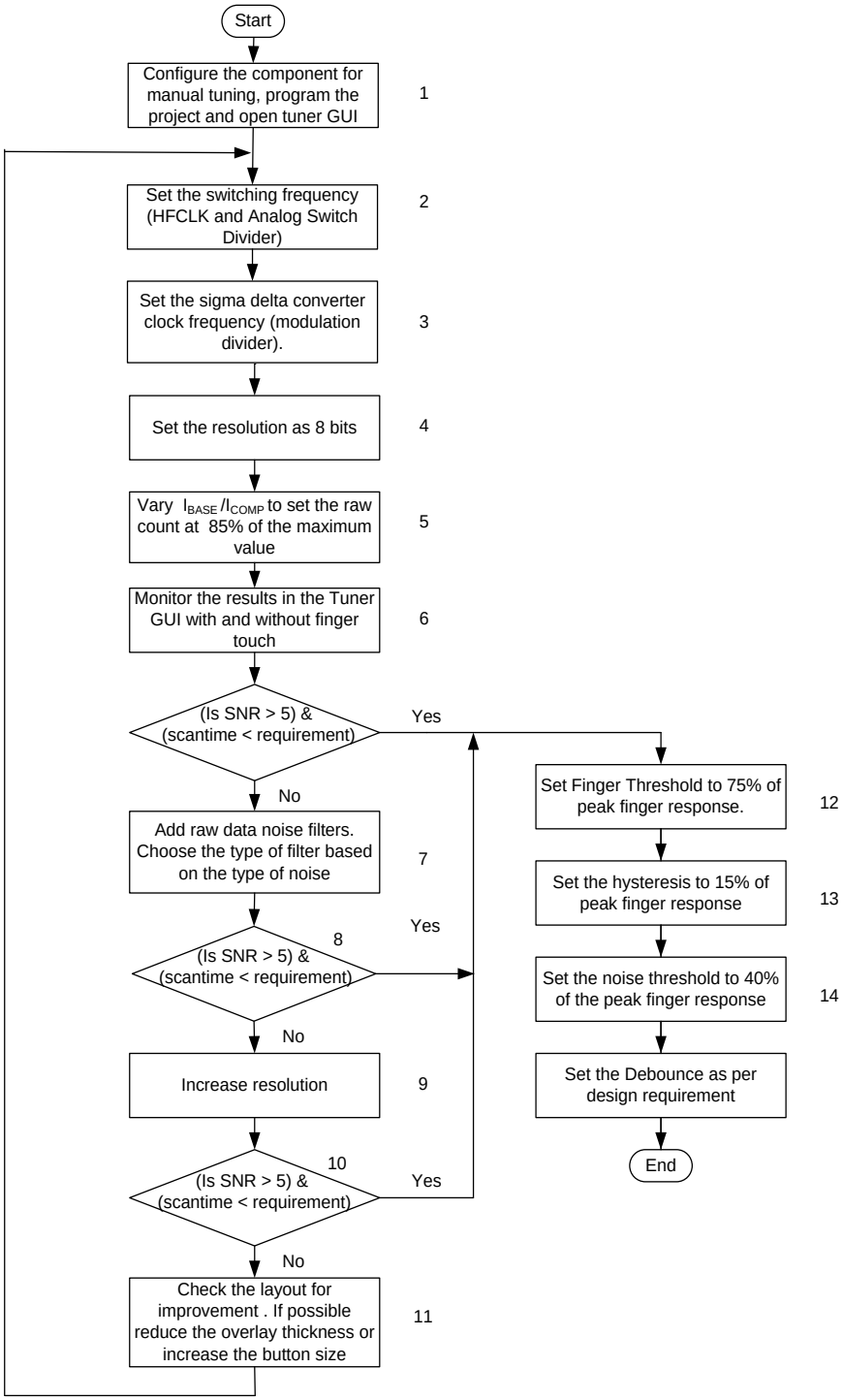
- Low Baseline Reset parameter: This parameter is used together with the negative noise threshold parameter. It counts the number of abnormally low raw counts required to reset the baseline. It is used to reset the baseline if the finger is placed on the sensor during device startup and later removed.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

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

5.2.2 Manual Tuning Process

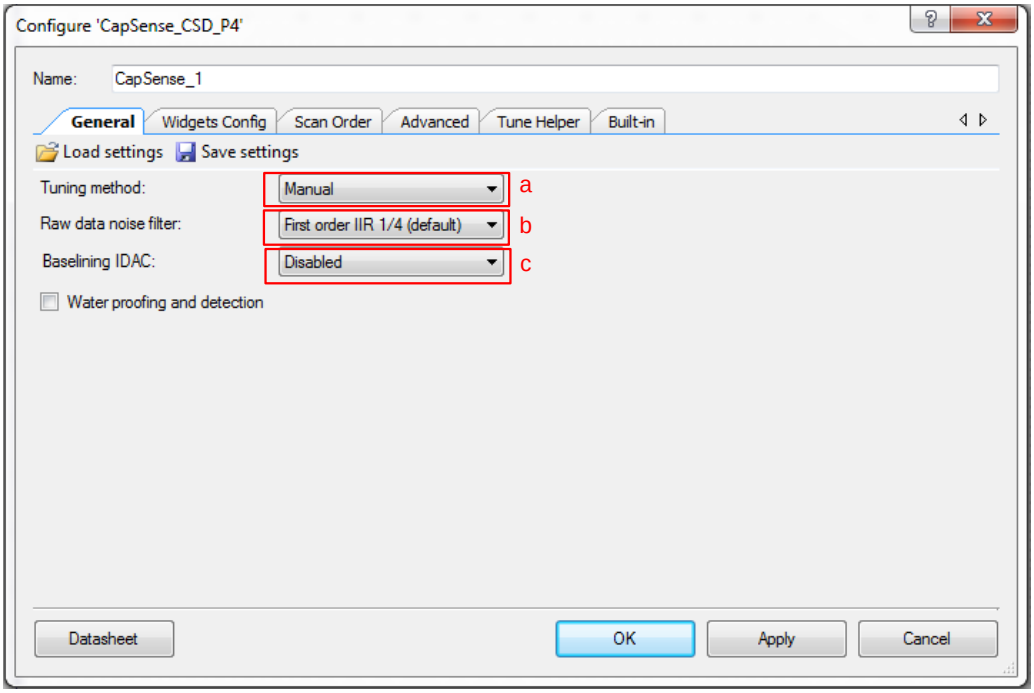
Figure 5-20 illustrates the manual tuning process. Follow the flow chart to manually set all the tuning parameters. See the following pages for details on each step.

Figure 5-20. Manual Tuning Process



1. Configure the Component for manual tuning as Figure 5-21 shows. This parameter allows you to select a firmware filter to reduce the noise in raw counts. Table 5-1 on page 27 explains the available filters and their applications.

Figure 5-21. General Settings



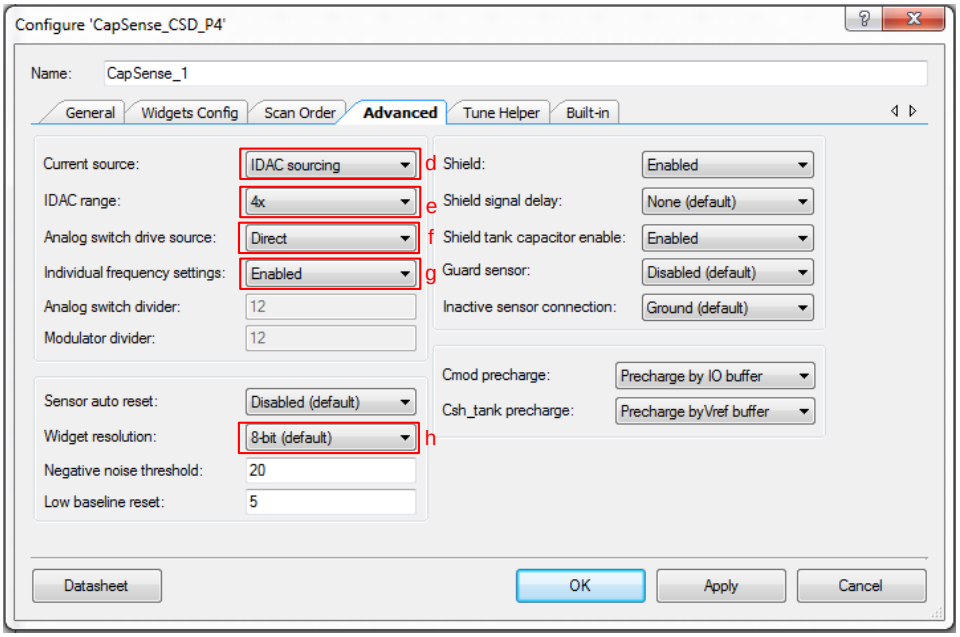
   a. **Tuning method:** Select “Manual”.
   b. **Raw data noise filter:** This parameter allows you to select a firmware filter to reduce the noise in raw counts. Table 5-1 on page 27 explains the available filters and their applications.
   c. **Baselining IDAC:** Enabling the baselining IDAC selects 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 single IDAC mode is easier than dual IDAC mode. However, the dual IDAC mode gives higher signal values. See Fundamentals of Manual Tuning for more details.

Configuration of the widgets is explained in CapSense Widgets and Scan Order. However, the selecting the manual tuning exposes some additional parameters that are explained in Tuning Parameters. 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 5-22 shows.

Figure 5-22. Advanced Settings



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

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

f. Analog switch drive source: The options available are Direct, 8-bit pseudo random sequence (PRS), and 12-bit PRS. You should use 8-bit or 12-bit PRS if your design has strict electromagnetic compatibility requirements, as it reduces the electromagnetic emission. Select “Direct” otherwise since it provides higher sensitivity compared to PRS. See Switching Clock Generator for details.

g. Individual frequency settings: Enable this option if your sensors do not have similar C_P values. Disable otherwise.

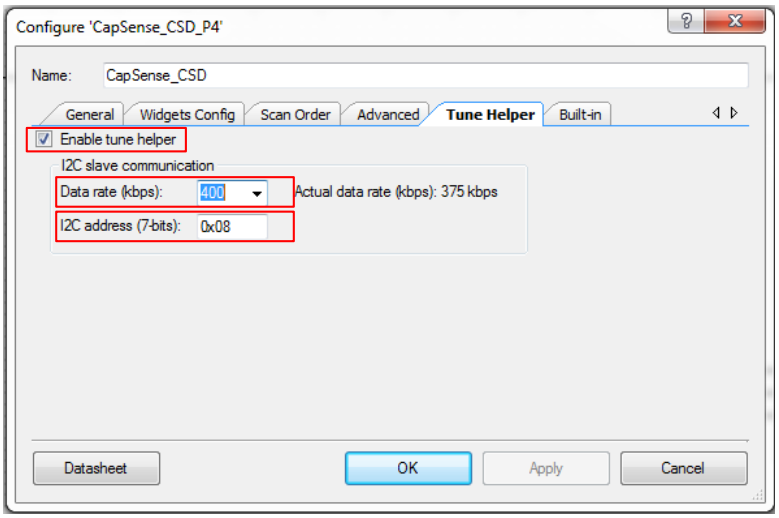
h. Widget resolution: This parameter selects between 8-bit or 16-bit variables for signal count. In most cases, you should use 8-bit widget resolution. If you have more signal than an 8-bit variable can handle, it means that you are using a higher resolution than required, and the resolution can be reduced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

i. Go to the “Tune Helper” tab, enable the tune helper, and set the data rate and I2C address, as Figure 5-23 shows.

Figure 5-23. Enabling Tune Helper



ii. To do manual tuning, you must include in your PSoC Creator project code 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();
  }
}
```

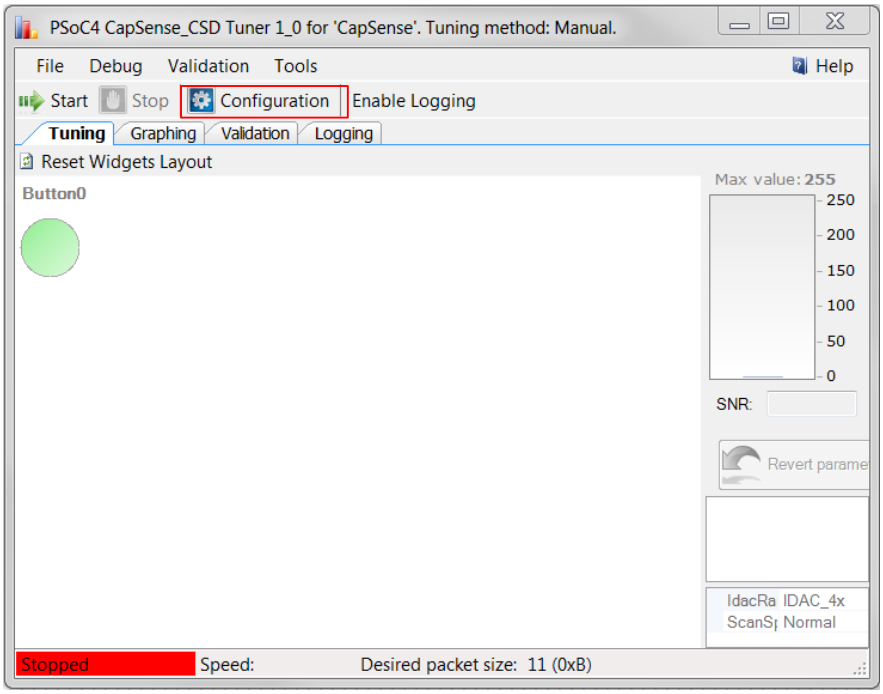
After finishing the manual tuning process, you can delete the function calls from your code. Note that they are not needed for SmartSense.

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

iv. Right click on the CapSense Component and click “launch tuner”.

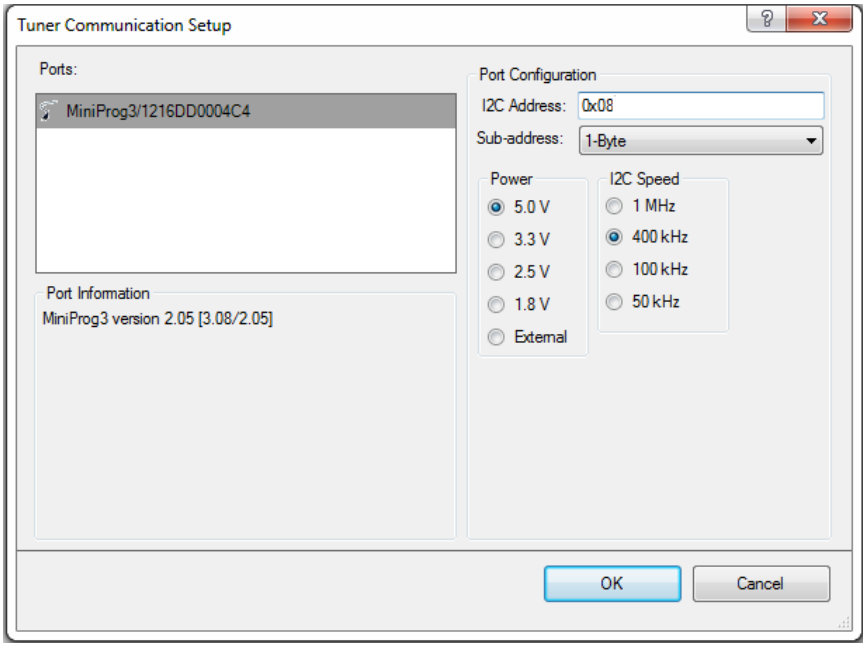
v. Open the Tuner Communication Setup window by clicking on the configuration button in the tuner window, as Figure 5-24 shows.

Figure 5-24. Tuner GUI



vi. Click on the MiniProg3 / I2C-USB bridge in the Tuner Communication Setup window and set the parameters, as Figure 5-25 shows:

Figure 5-25. Tuner Communication Setup

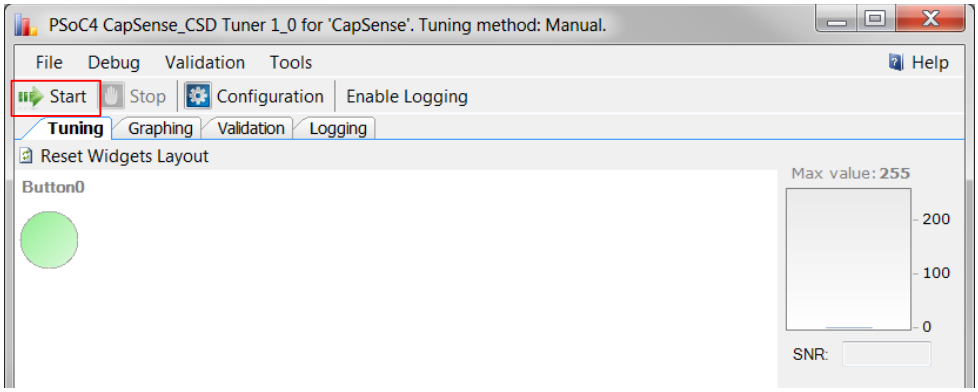


Note: The voltage settings in this window must match the voltage settings on the board.

vii. Click OK.

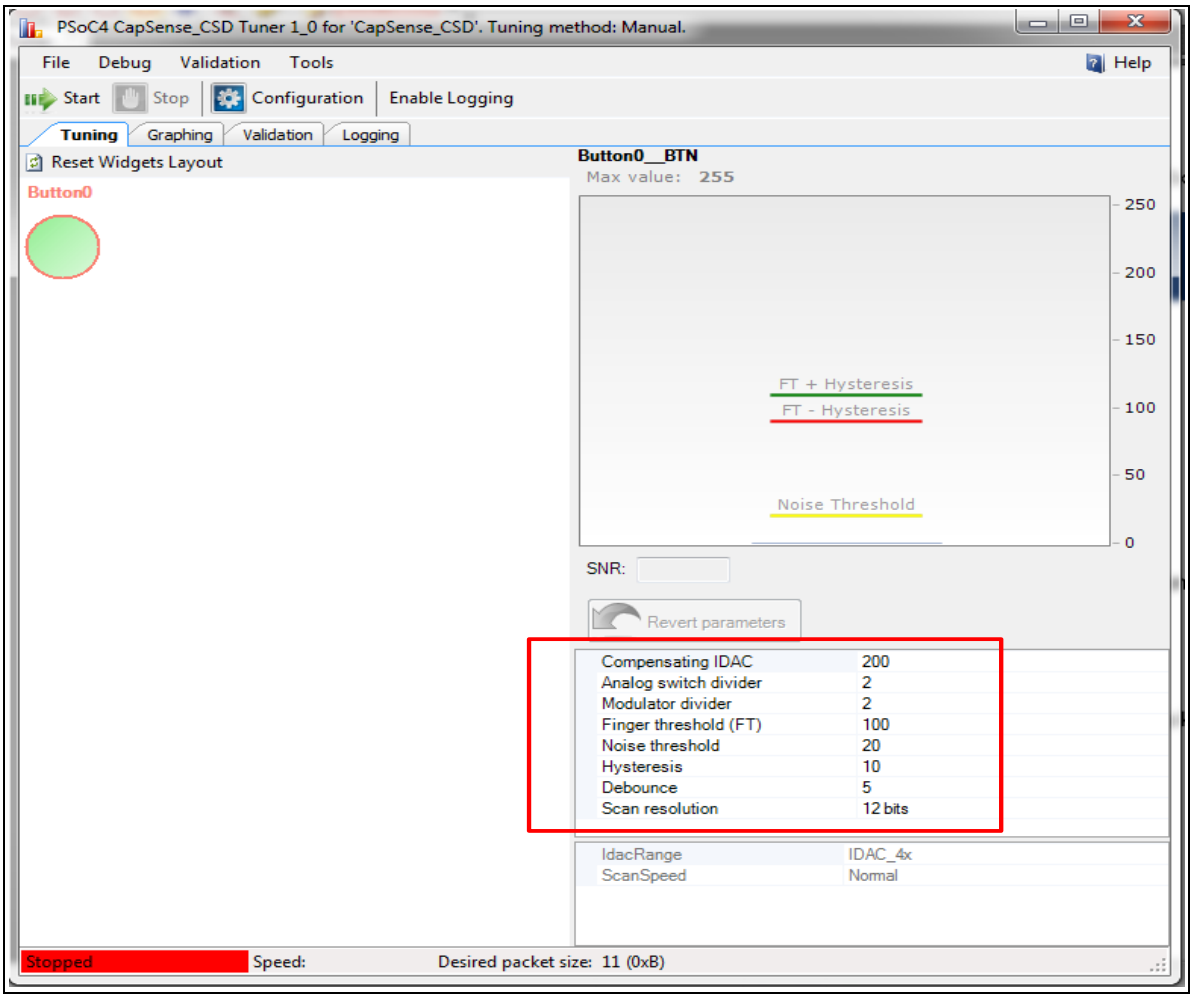
viii. Click the “Start” button in the tuner window to initiate I²C communication, as Figure 5-26 shows.

Figure 5-26. Starting Tuner Communication



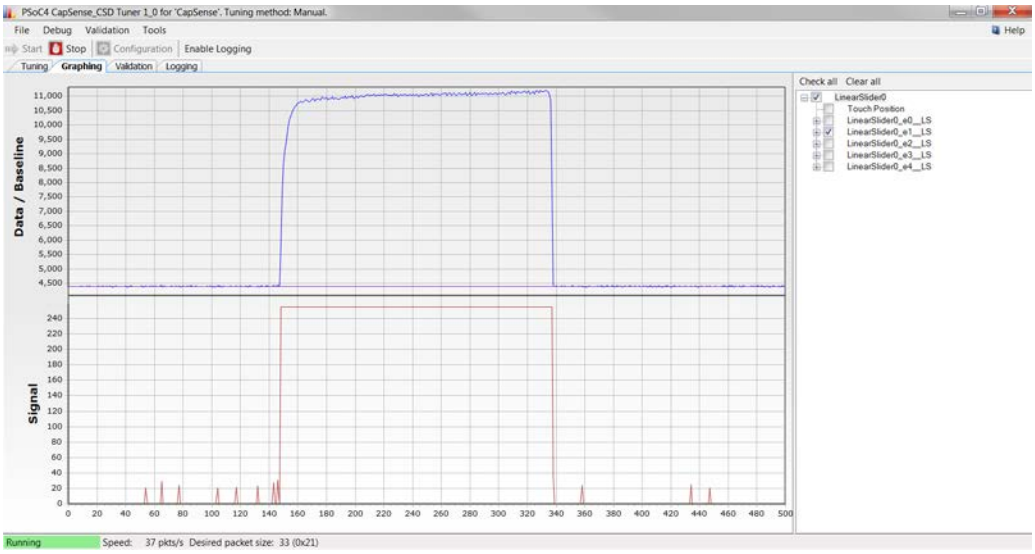
ix. You can set the parameters for each sensor at the lower right corner of this window, as Figure 5-27 shows. Clicking “OK” copies the parameters from the tuner GUI to CapSense Component.

Figure 5-27. Setting the Parameters For a Sensor



x. The Graphing tab allows you to monitor signals, as Figure 5-28 shows. You can log data using the Logging tab.

Figure 5-28. Monitoring the CapSense Results



2. Set the analog switch 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 don't 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 10x mode reduces their capacitance. Use a FET input probe if available. Increase the analog switch divider, if the sensor is not charging and discharging properly.

3. Set the modulation switch divider for each sensor. The modulation switch divider should be less than the analog switch divider. You should also make sure that the analog switch divider is an integral multiple of the modulation switch divider. Selecting a higher modulation switch divider gives improved SNR. However, the total scan time increases as the modulation switch divider increases.

   You should initially set the modulation switch divider at $1/4^{th}$ of the analog switch divider. After tuning is complete, if the SNR is good, reduce the modulation divider to reduce the scan time, else increase it.

4. Set the resolution N to 8 bits. Increasing the 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 compensation IDAC value in the GUI until the raw counts reach 85% of the maximum raw counts (2^N). Note that decreasing the compensation 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 dual IDAC mode, you must tune both the compensation IDAC and the baselining IDAC. This is a multiple iteration process.

   Initially, tune the raw counts to 85% of the maximum raw counts by only using the compensation IDAC (keep the baselining IDAC at zero). Then increase the baselining IDAC to 10% of the compensation IDAC value, and re-tune the compensation IDAC to get 85% of the maximum raw counts. Note down the SNR. Repeat this procedure with a higher value of baselining IDAC. Note that a higher baselining IDAC value results in higher signal (see Fundamentals of Manual Tuning for more details).

   Repeat the process until maximum SNR is achieved.

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 modulation switch divider. You have to make sure that SNR is not going below 5 when reducing the modulation switch 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 modulation switch 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 75 percent of the signal value.
13. Set the hysteresis to 15 percent of the signal value.
14. Set the noise threshold to 40 percent of the signal value.

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” to “None”.

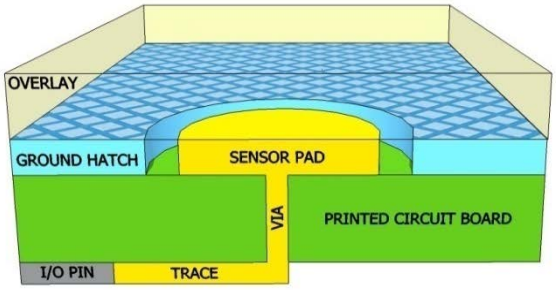
6. Design Considerations

This chapter explains hardware and software design considerations for CapSense.

6.1 Sensor Construction

Figure 6-1 shows the most common CapSense sensor construction.

Figure 6-1. CapSense sensor construction

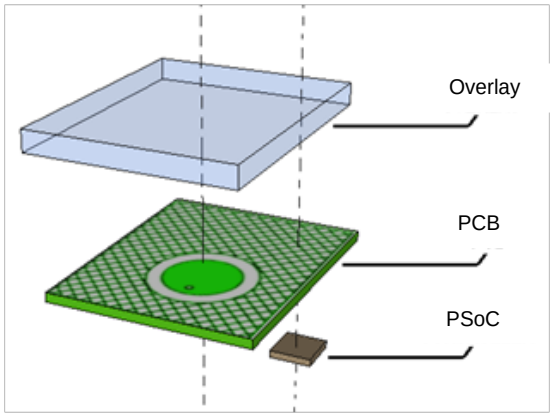


The copper pads etched on the surface of the PCB act as CapSense sensors. A nonconductive overlay serves as the touch surface. The overlay also protects the sensor from the environment and prevents direct finger contact. A ground hatch surrounding the sensor pad isolates the sensor from other sensors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

If waterproofing 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 Shield Electrode and Guard Sensor 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 6-2 shows an exploded view of the CapSense hardware.

Figure 6-2. CapSense hardware



6.1.1 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.

6.1.2 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	2
Touchpad	0.5

6.1.3 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.

6.2 PCB Layout Guidelines

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

6.2.1 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. But widening the gap also decreases noise immunity. You can also reduce C_P by driving the hatch with a shield signal.

See Layout Rule Checklist for details.

6.2.2 Board Layers

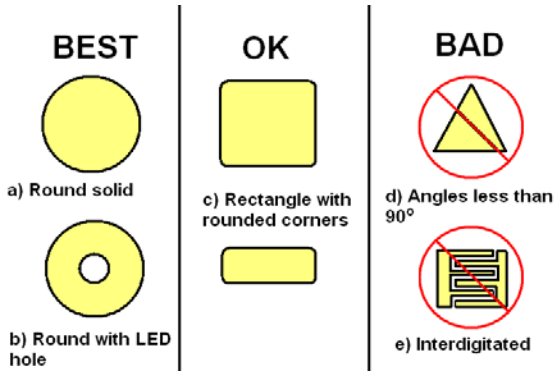
Most applications use a two-layer board with the sensor pads and the hatched ground planes on the top side and all other components on the bottom side. More complex PCBs use four layers. FR4-based PCB designs perform well with board thickness ranging from 0.020 inches (0.5 mm) to 0.063 inches (1.6 mm).

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.

6.2.3 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 6-3 shows recommended button shapes.

Figure 6-3. Recommended Button Shapes

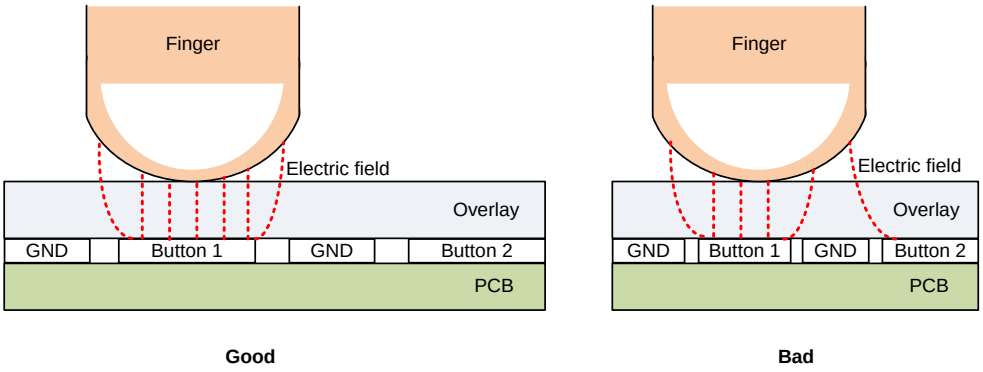


Button diameter should be 5 mm to 15 mm. 10 mm is suitable for most applications. A larger diameter is appropriate for thicker overlays.

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

You should select the spacing between the 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 6-4 shows.

Figure 6-4. Spacing Between Buttons



6.2.4 Slider Design

Figure 6-5 shows the recommended slider pattern, and Table 6-3 shows the recommended dimensions.

Figure 6-5. Typical Slider Pattern

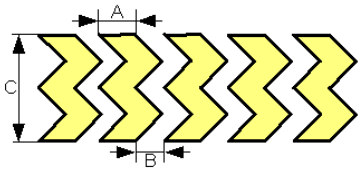


Table 6-3. Slider Dimensions

Parameter	Min	Max	Recommended
Width of the Segment (A)	1.5 mm	4 mm	Equal to overlay thickness, but within the min/max limits.
Clearance between Segments (B)	0.5 mm	2 mm	Equal to sensor to ground clearance.
Height of the segment (C)	7 mm	15 mm	12 mm

When one segment is scanned, the adjacent segments are grounded. To maintain a uniform signal level from the sensor, you should ground the two segments at the both ends of a slider Therefore, if your application requires an ‘n’ segment slider, you should create n + 2 physical segments. If the hatch around the slider is shielded, you should shield the last two segments.

6.2.5 Sensor and Device Placement

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

- Minimize the trace length from the PSoC pins to the sensor pad.
- Mount series resistors within 10 mm of the PSoC pins to reduce RF interference and provide ESD protection. See Series Resistance on CapSense Input Lines for details.
- Mount the PSoC 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.
- Avoid connectors between the sensor and the PSoC pins because connectors increase C_P and noise pickup.

6.2.6 Trace Length and Width

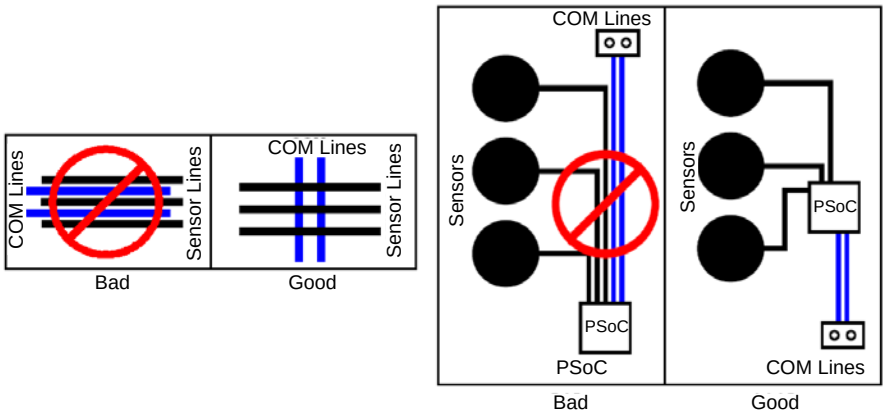
Use short and narrow PCB traces to minimize the parasitic capacitance of the sensor. The maximum recommended trace length is 12 inches (300 mm) for a standard PCB and 2 inches (50 mm) for flex circuits. The maximum recommended trace width is 7 mil (0.18 mm). You should surround the CapSense traces with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

6.2.7 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 6-6 shows.

Figure 6-6. Routing of Sensor and Communication Lines

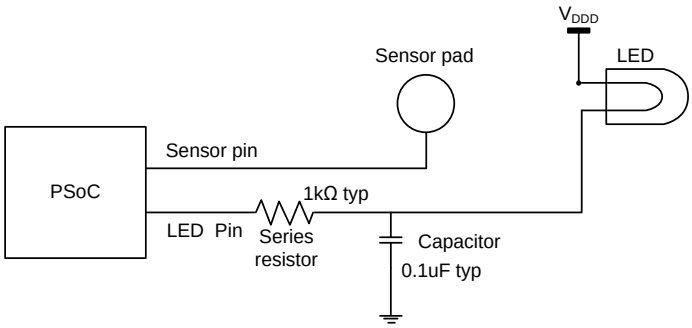


6.2.8 Crosstalk Solutions

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 6.2.7 explains

You can also reduce crosstalk by removing the rapid transitions in the LED drive voltage, by using a filter as Figure 6-7 shows. Design the filter based on the required LED response speed.

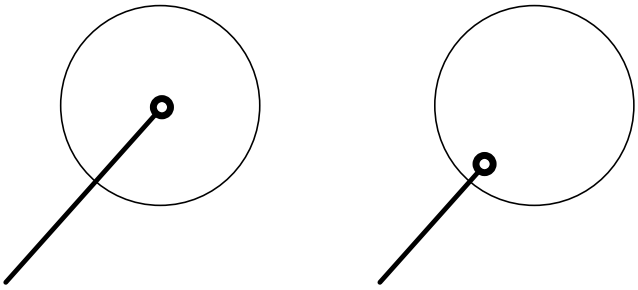
Figure 6-7. Reducing Crosstalk



6.2.9 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 6-8 shows.

Figure 6-8. Via Placement on the Sensor Pad



Via at the center of the sensor (long trace)

Via near the edge of the sensor (short trace)

6.2.10 Ground Plane

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), as Figure 6-9 shows, and 17 percent on the bottom layer (7 mil line, 70 mil spacing), as Figure 6-10 shows. Use the same hatching if you are using shield instead of ground.

Figure 6-9. Example of a Button and Slider Layout, Top Layer

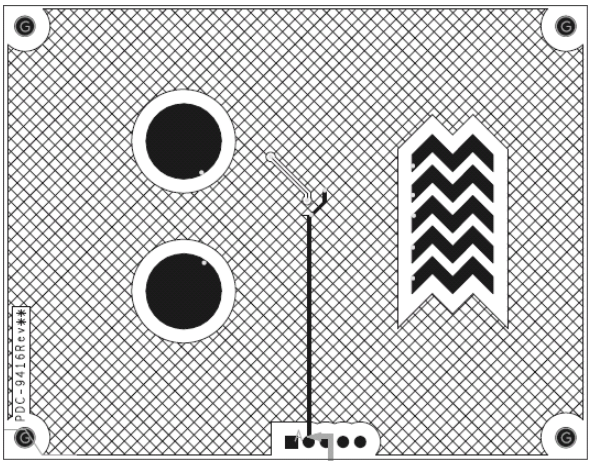
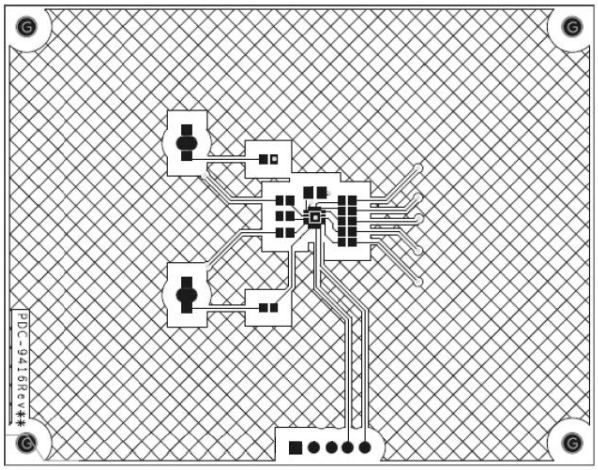


Figure 6-10: Recommended Button and Slider Layout, Bottom Layer



6.2.11 Shield Electrode and Guard Sensor

See Shield Electrode and Guard Sensor for the basic principle behind waterproofing.

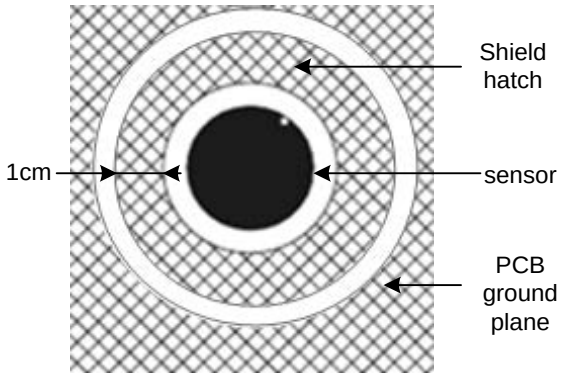
6.2.11.1 Shield

The recommendations for the shield electrode are:

- Top layer hatch: 7 mil trace and 45 mil grid (15 percent fill)
- Bottom layer hatch: 7 mil trace and 70 mil grid (10 percent fill)
- Ground only the areas surrounding the sensor pads and the PSoC

The shield electrode pattern surrounds the sensor pads and exposed traces, and spreads no further than 1 cm from these features. Spreading the shield electrode beyond 1 cm has negligible effect on system performance. If board space is limited, the shield can spread less than 1 cm. Figure 6-11 shows an example layout.

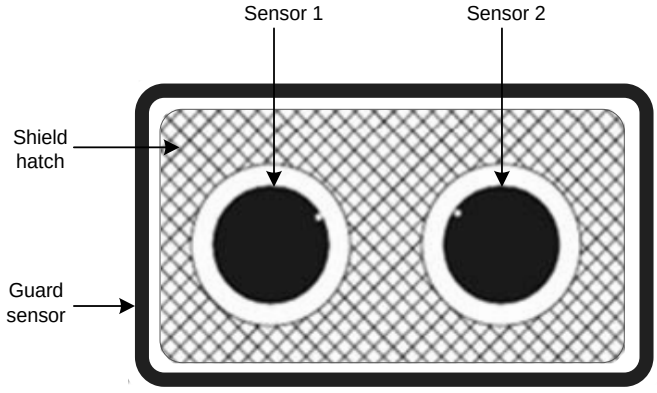
Figure 6-11. Shield Electrode Pattern



6.2.11.2 Guard Sensor

The guard sensor is a copper trace that surrounds all of the sensors, as Figure 6-12 shows.

Figure 6-12. PCB Layout with Shield Electrode and Guard Sensor



Make sure that the shield electrode pattern surrounds the guard sensor, exposed traces, and spreads no further than 1 cm from these features.

The guard sensor should be in the shape of a rectangle with curved edges. The recommended thickness of a guard sensor is 2 mm, and the maximum distance between the guard trace and the shield hatch is 1 mm.

6.2.12 Layout Rule Checklist

You can use the checklist provided in Table 6-4 to help verify your layout design.

Table 6-4. Layout Rule Checklist

Serial Number	Category		Minimum value	Maximum value	Recommendations / Remarks
1	Button	Shape			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	4 mm	Equal to overlay thickness, but within min / max limits
		Clearance between segments	0.5 mm	2 mm	Equal to sensor to ground hatch clearance, but within min / max limits.
		Height of segment	7 mm	15 mm	12 mm
3	Overlay	Type			Material with high relative permittivity (except conductors) Remove any air gap between sensor board and overlay / front panel of the casing.
4	Sensor Traces	Width		7 mil	Use the minimum width possible with the PCB technology you are using
		Length		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		Use maximum clearance while keeping the trace length as low as possible

Serial Number	Category		Minimum value	Maximum value	Recommendations / Remarks
		Routing			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			10 mil
6	Ground				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				Place resistor within 10 mm of PSoC pin
8	Shield electrode	Spread		1 cm	If you have PCB space, use 1 cm spread.
9	Guard sensor (for water tolerance)	Shape			Rectangle with curved edges
		Thickness			Recommended thickness of shield trace is 2 mm and distance of trace to shield hatch is 1 mm.

6.3 Low Power Design

See AN86233, PSoC 4 Low-Power Modes and Power Reduction Techniques, for information on how to reduce the power consumption of your design.

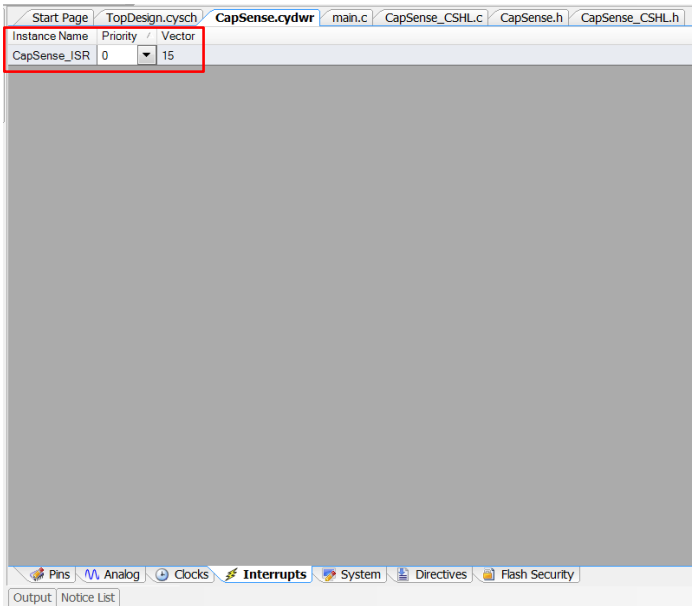
6.4 Response Time

As CapSense is a user interface solution, the response time of CapSense to a touch is a very important consideration. This section explains various factors that affect the response time.

6.4.1 Interrupt Priority

The CapSense Component in PSoC 4 has a non blocking architecture. As a result, the CPU can execute another application while the CapSense hardware is scanning the sensors. The hardware generates an interrupt when the scan is complete. The priority of this interrupt can be changed in the *.cydwr* file of the PSoC Creator project, as Figure 6-13 shows.

Figure 6-13. Changing Interrupt of the CapSense_ISR



You can increase this priority to improve the CapSense response. Priority “0” is the highest interrupt priority. Increasing this number lowers the interrupt priority. However if you have another critical function in your project, you should set the priority of the CapSense interrupt to be below that of the other interrupt.

6.5 ESD Protection

The nonconductive overlay material used in CapSense provides inherent protection against ESD. Table 6-5 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 6-5. Overlay Thickness for ESD Protection

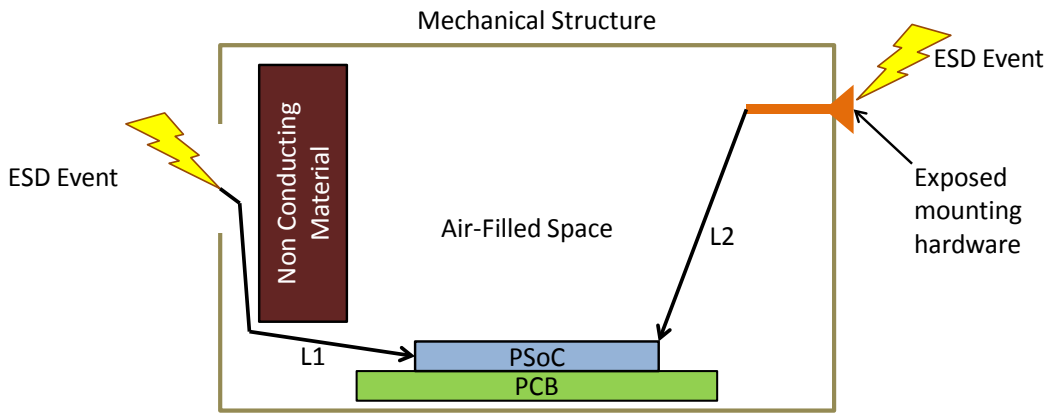
Material	Breakdown Voltage (V/mm)	Minimum overlay thickness for protection against 12 kV ESD (mm)
Air	1200 – 2800	10
Wood – dry	3900	3
Glass – common	7900	1.5
Glass – Borosilicate (Pyrex®)	13,000	0.9
PMMA Plastic (Plexiglas®)	13,000	0.9
ABS	16,000	0.8
Polycarbonate (Lexan®)	16,000	0.8
Formica	18,000	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.

6.5.1 Preventing ESD Discharge

Preventing the ESD discharge from reaching the PSoC is the best countermeasure you can take. You should 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 6-14, if L1 and L2 are greater than 10 mm, the system can withstand a 12 kV ESD.

Figure 6-14. ESD Paths

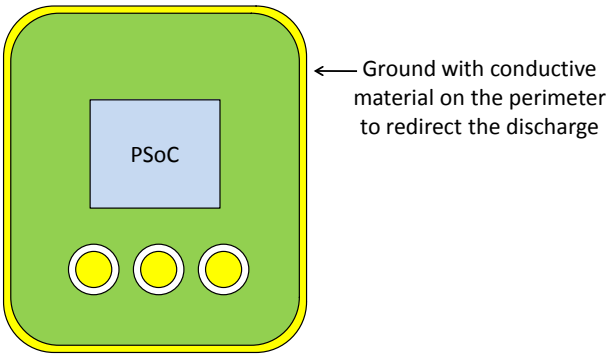


If it is not possible to maintain adequate distance, place a protective layer of nonconductive material with a high breakdown voltage between the possible ESD source and PSoC. One layer of 5 mil thick Kapton® tape can withstand 18 kV. See Table 6-5 on page 61 for other material dielectric strengths.

6.5.2 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 guard ring on the perimeter of the circuit board, as Figure 6-15 shows. The guard 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 6-15. Guard Ring



6.5.3 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 6-6 lists the recommended ESD protection devices.

Table 6-6. 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

6.6 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.

6.6.1 Radiated Interference

Radiated electrical energy can influence system measurements and potentially affect the operation of PSoC. Use the following techniques to prevent the radiated interference.

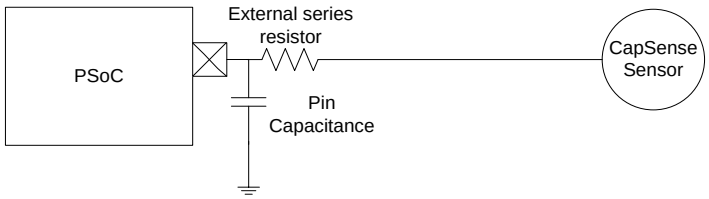
6.6.1.1 Ground Plane

In general, a ground plane on the PCB reduces generated and received RF noise.

6.6.1.2 Series Resistance on CapSense Input Lines

Each PSoC GPIO has a small pin capacitance, as Figure 6-16 shows. You can connect an external resistance between the sensor and the GPIO to form an RC circuit that filters RF noise.

Figure 6-16. RC Filter



The recommended series resistance for CapSense input lines is 560 Ω. Adding this resistance changes the time constant of the switched-capacitor circuit that converts the sensor capacitance into an equivalent current (see GPIO Cell Capacitance to Current Converter). If you use a series resistance larger than 560 Ω, the sensor capacitance may not get charged and discharged properly. This affects the operation of CapSense. Resistance values smaller than 560 Ω are less effective at blocking RF.

Series resistors should be placed within 10 mm of the PSoC pins.

6.6.1.3 Digital Communication Lines

Communication lines such as I^2C and SPI also benefit from a series resistance. You should use 330 Ω resistors for communication lines.

6.6.1.4 Current Loop Area

If you isolate the CapSense ground hatch and the ground fill around the PSoC, the sensor switching current may take a different return path, as Figure 6-17 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 6-18 shows.

Figure 6-17. Improper Current Loop Layout

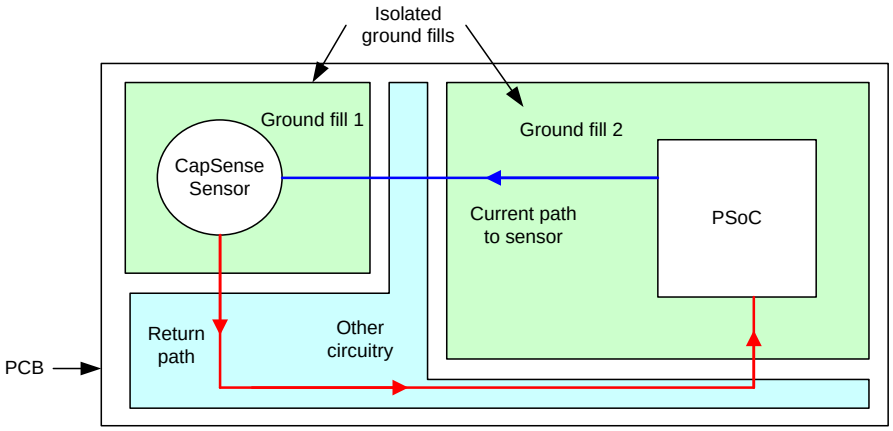
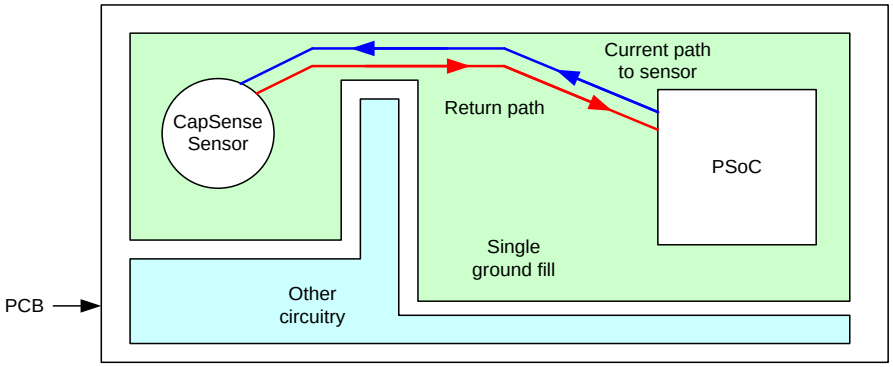


Figure 6-18. Proper Current Loop Layout



6.6.1.5 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.

6.6.2 Radiated Emissions

PSoC 4 CapSense provides an option to use pseudo random sequence (PRS) to reduce emissions from CapSense. See Switching Clock Generator for details.

6.6.3 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.

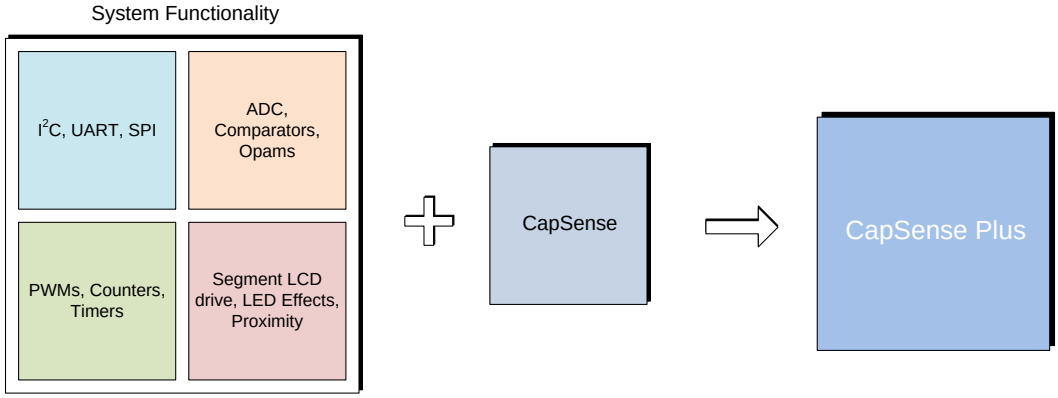
- Use decoupling capacitors on PSoC power supply pins to reduce the conducted noise from the power supply. See the PSoC 4 device datasheet for details.
- Provide GND and V_{DD} planes on the PCB to reduce current loops.
- If the PSoC PCB is connected to the power supply using a cable, minimize the cable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

- Communication: I2C, UART, SPI
- Analog functions: ADC, comparators, opamps
- Digital functions: PWMs, counters, timers, UDBs
- Segment LCD drive
- Bootloaders
- Different power modes: active, sleep, deep sleep, hibernate, stop

  For more information on PSoC 4, refer to AN79953, Getting Started with PSoC 4.

The flexibility of the PSoC 4 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 PSoC 4 CapSense Plus can result in significant cost savings.

Table 7-1. Examples of CapSense Plus

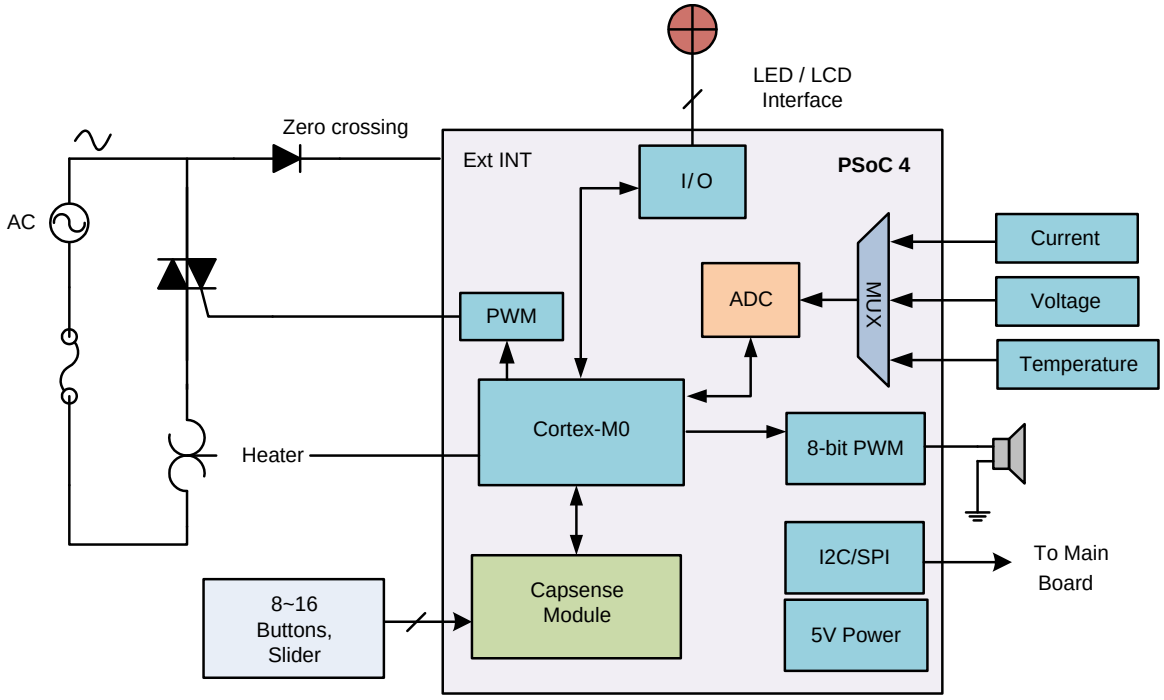
Application	CapSense	ADC	Comparator	PWM, Counter, Timer, UDBs	SCB	LCD drive	GPIOs
Washing machine	User interface: buttons, radial sliders	Temperature sensor	Water level monitor	Buzzer, FOC* motor control	I2C LCD display, UART network interface	Segment LCD	LED indication
Water heater	User interface: buttons, 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
Thermal printers	User interface: buttons	Overheat protection, paper sensor		Stepper motor control	SPI communication interface		LED indication

* 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.website to understand the PSoC 4 device.

8.2 Datasheet

PSoC 4 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 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

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.

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: PSoC® 4 CapSense® Design Guide Document Number: 001-85951			
Revision	**Issue Date**	**Origin of Change**	**Description of Change**
**	3973432	NIDH	New Design Guide
*A	4059171	NIDH	Added dual IDAC support. Updated some shematics in chapter 6. Other minor changes to chapters 3, 5, and 6.