Getting Started with CapSense®

Document No. 001-64846 Rev. *O

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): +1.800.858.1810
Phone (Intnl):+1. 408.943.2600
http://www.cypress.com

Copyrights

Trademarks

PSoC Designer™ and SmartSense™ are trademarks and PSoC®, CapSense®, and TrueTouch® are registered trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Source Code

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

Disclaimer

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

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

Contents

1. Introduction

1.1 How to Use This Guide

This guide is an ideal starting point for those who are new to capacitive touch sensing (CapSense®) as well as for learning key design considerations and layout best practices to ensure design success. In addition, you can use this guide to:

- Become familiar with the technology underlying CapSense solutions
- Understand important design considerations, such as layout, schematic, and EMI (Electro Magnetic Interference)
- Become familiar with the CapSense product portfolio
- Select the right device for your application
- Migrate between CapSense devices
- Become familiar with the many resources available to support your entire design cycle

When you are ready to design your application, consult the Design Guide specific to the CapSense device family you have selected.

1.2 Cypress's CapSense Documentation Ecosystem

Figure 1-1 and Table 1-1 summarize the Cypress CapSense documentation ecosystem. These resources allow you to quickly access the information needed to complete a CapSense product design successfully. 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 provides links to the supporting documents for each of the numbered tasks in Figure 1-1.

Figure 1-1. Typical CapSense Product Design Flow

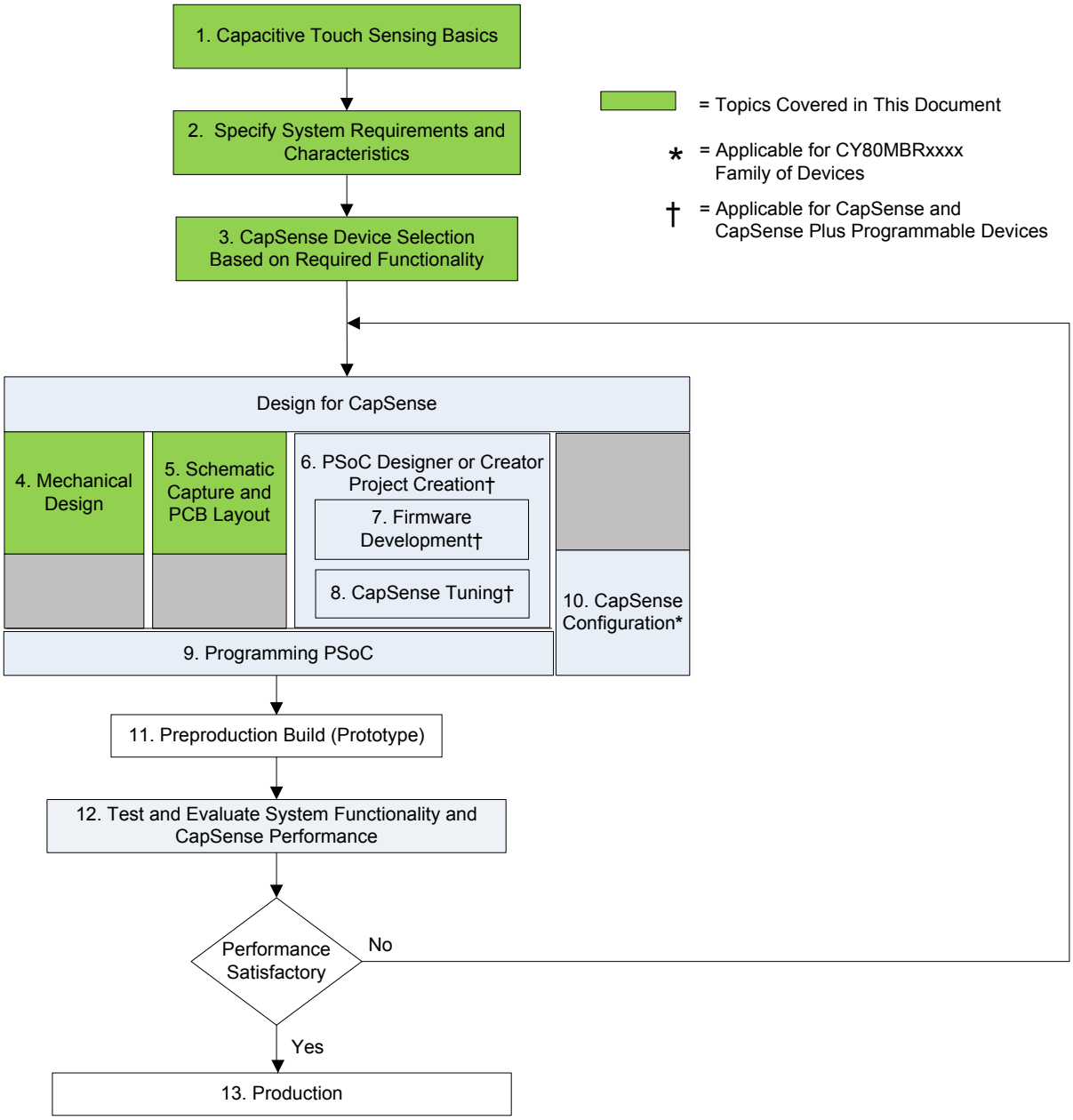


Table 1-1. Cypress Documents Supporting Numbered Design Tasks of Figure 1-1

Task No.	Supporting Cypress CapSense Documentation
1	Getting Started With CapSense
2	Getting Started With CapSense Device Family Specific CapSense Device Datasheets Device Family Specific CapSense Design Guides
3	Getting Started With CapSense
4	Getting Started With CapSense
5	Getting Started With CapSense
6	PSoC Designer User Guides PSoC Creator Quick Start Guide
7	Assembly Language User Guide C Language Compiler User Guide CapSense Controller Code Examples Device Family Specific Technical Reference Manuals PSoC® 3 8051 Code and Memory Optimization- AN60630 PSoC® 4 and PSoC 5LP ARM Cortex Code Optimization –AN89610
8	Device Family Specific CapSense Design Guides Device Family Specific CapSense User Module Datasheets (CSA_EMC) CapSense Data Viewing Tools -AN2397 CapSense Controller Code Examples Design Guide
9	Programmer User Guide MiniProg3 User Guide

1.3 Cypress CapSense Products

Cypress CapSense solutions bring elegant, reliable, and easy-to-use capacitive touch sensing functionality to your design. Our capacitive touch sensing solutions have replaced more than four billion mechanical buttons. Capacitive touch sensing has changed the face of industrial design in products such as cellphones, PCs, consumer electronics, automotive features, and white goods. Cypress's robust CapSense solutions leverage our flexible Programmable System-on-Chip (PSoC®) architecture, which accelerates time-to-market, integrates critical system functions, and reduces BOM costs.

1.3.1 Cypress CapSense Differentiation

- Robust sensing technology
- High noise immunity
- High-performance sensing across a variety of overlay materials and thicknesses
- SmartSense™ Auto-Tuning technology
- Proximity sensing
- Liquid-tolerant operation
- Complete user interface solution including audio, visual, and haptic feedback
- Low power consumption
- Wide operating voltage range (1.71—5.5 V)

- Small form-factor packaging
- Reduced BOM cost with integrated CapSense Plus features (ADC, DAC, timer, counter, and PWM)

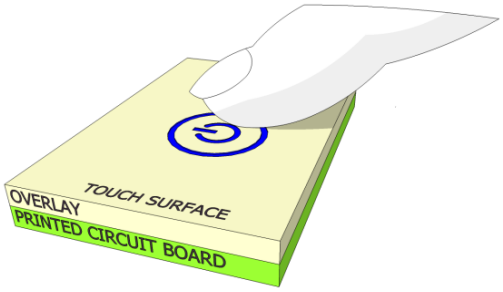
1.4 Document Conventions

Convention	Usage
`Courier New`	Displays file locations, user entered text, and source code: `C:\ ...cd\icc\`
Italics	Displays file names and reference documentation: Read about the *sourcefile.hex* file in the *PSoC Designer User Guide*.
File > Open	Represents menu paths: File > Open > New Project
Bold	Displays commands, menu paths, and icon names in procedures: Click the **File** icon and then click **Open**.
Times New Roman	Displays an equation: 2 + 2 = 4
Text in gray boxes	Describes Cautions or unique functionality of the product.

2. CapSense Technology

Cypress's CapSense controllers use changes in capacitance to detect the presence of a finger on or near a touch surface, as shown in Figure 2-1. This touch button example illustrates a capacitive sensor replacing a mechanical button. The sensing function is achieved using a combination of hardware and firmware. The following section provides an overview of capacitive sensing technology and CapSense solutions.

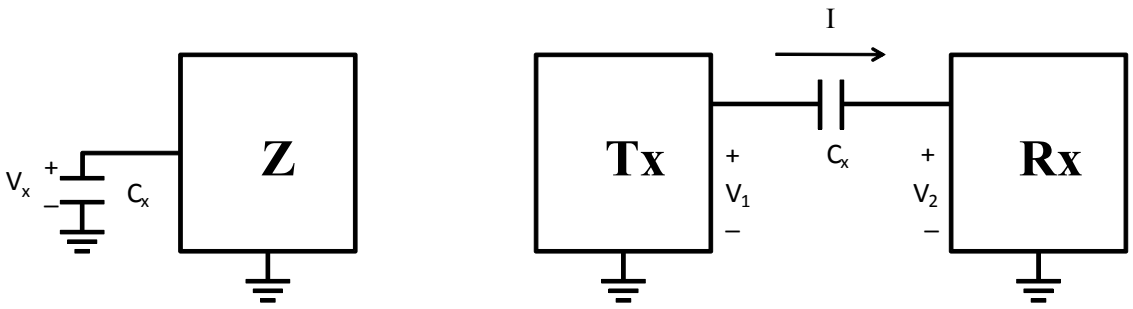
Figure 2-1. Illustration of a Capacitance Sensor Application



2.1 Capacitive Sensing Methods

Capacitance can be measured between two points using either self capacitance or mutual capacitance.

Figure 2-2. Self-Capacitance and Mutual-Capacitance Methods



Self Capacitance Mutual Capacitance

2.1.1 Self Capacitance

Self capacitance uses a single pin and measures the capacitance between that pin and ground. A self-capacitance sensing system operates by driving current on a pin connected to a sensor and measuring the voltage. When a finger is placed on the sensor, it increases the measured capacitance. Self-capacitance sensing is best suited for single-touch sensors, such as buttons and sliders.

Cypress's CapSense solutions use self-capacitance sensing because it enables efficient use of pins for single-touch sensors and sliders.

2.1.2 Mutual-Capacitance

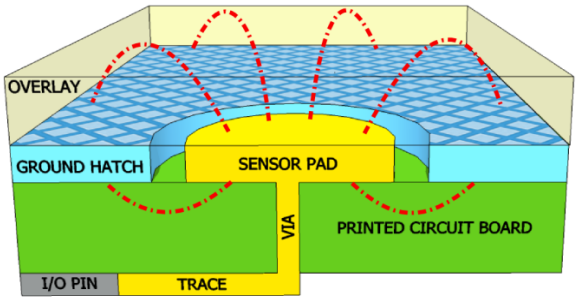
Mutual-capacitance uses a pair of pins and measures the capacitance between those pins. A mutual-capacitance system operates by driving a current on a transmit pin and measuring the charge on a receive pin. When a finger is placed between the transmit and receive pins, it decreases the measured capacitance. The mutual-capacitance effect is best suited to multi-touch systems, such as touchscreens and trackpads.

Cypress's TrueTouch® touchscreen solutions use mutual-capacitance sensing. See TrueTouch Touchscreen Controllers to learn about these products. Cypress also offers trackpad solutions. Contact your local Cypress sales office directly for more information. To find your local sales office, click here.

2.2 Self-Capacitance Equivalent Model

In a CapSense self-capacitance system, the sensor capacitance measured by the controller is called C_X. When a finger is not on the sensor, C_X equals the parasitic capacitance (C_P) of the system. This parasitic capacitance is a simplification of the distributed capacitance that includes the effects of the sensor pad, the overlay, the trace between the CapSense controller pin and the sensor pad, the vias through the circuit board, and the pin capacitance of the CapSense controller. C_P is related to the electric field around the sensor pad. Although Figure 2-3 shows field lines only around the sensor pad, the actual electric field is more complicated.

Figure 2-3. C_P and Electric Field



When a finger touches the sensor surface, it forms a simple parallel plate capacitor with the sensor pad through the overlay. The result is called finger capacitance, C_F, and is defined by Equation 1. C_F is a simplification of a distributed capacitance that includes the effects of the human body and the return path to the circuit board ground.

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

Where:

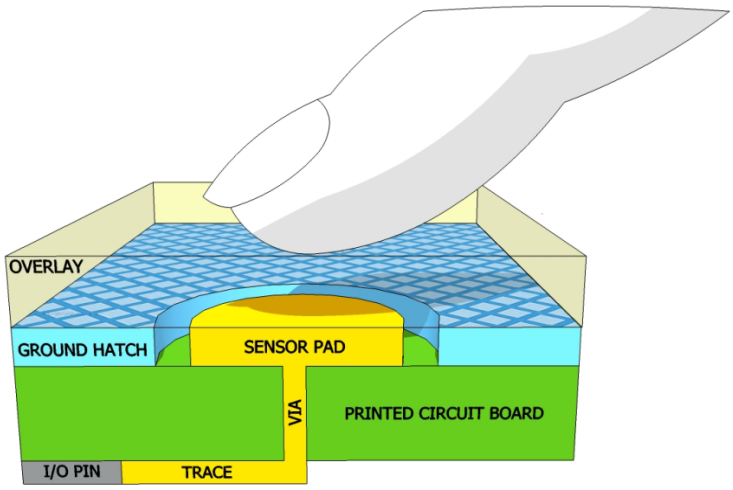
ε_0 = Free space permittivity

ε_r = Dielectric constant of overlay

A = Area of finger and sensor pad overlap

D = Overlay thickness

Figure 2-4. CapSense System Equivalent Model



With a finger on the sensor surface, C_X equals the sum of C_P and C_F.

$$C_X = C_P + C_F$$ Equation 2

2.3 CapSense Sensing Technology

2.3.1 Sensing Methods

There are a number of capacitive sensing methods currently in use across the electronics industry. Some of them include:

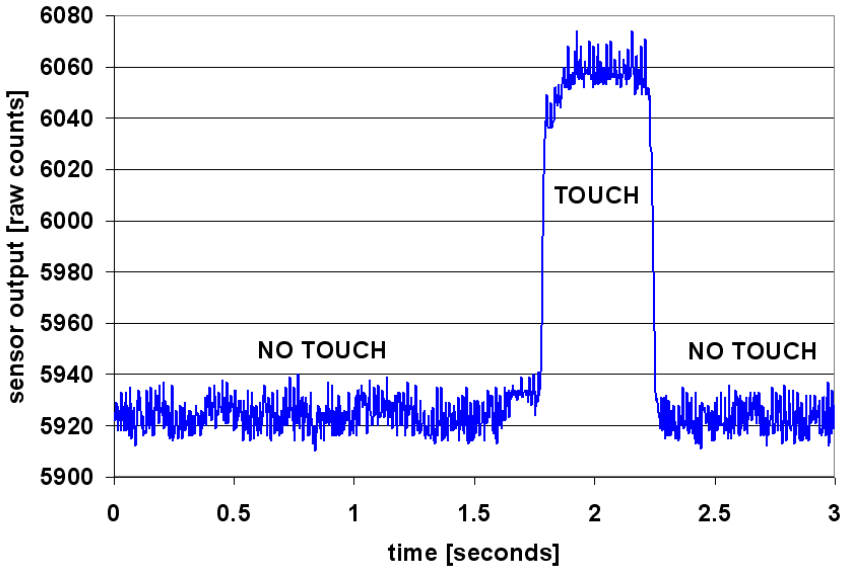
- *Charge Transfer*: The change in sensor capacitance, introduced by a finger touch, modifies the charge transfer between the sensor capacitor and the reference capacitor. These incremental charge packets are transferred until a reference voltage is achieved on the reference capacitor, which indicates the presence of a touch.
- *Relaxation Oscillator*: A sensor capacitor is used to set its frequency directly. The capacitance introduced by the finger touch is detected on the sensor by tracking the change in the frequency of the oscillator. When there is a finger touch, the sensor capacitor increases and frequency of the oscillator decreases.
- *TX-RX:* A source waveform is driven on the TX end of a mutual-capacitance system and senses the response on the RX end. The received signal reflects changes in sensor capacitance.
- *ADC:* A current source generates a linear voltage ramp on a capacitor. This voltage is input to an analog comparator circuit. The comparator's output is monitored and a counter increments whenever it transitions from high to low.

Cypress's CapSense devices measure sensor capacitance using either CapSense with Sigma Delta modulator (CSD) or CapSense Successive Approximation (CSA_EMC). Both methods are variants of the ADC method.

2.3.2 Capacitance Conversion

The CapSense algorithm converts the sensor capacitance into a digital count, called raw count. The raw count is interpreted as either a TOUCH or NO TOUCH state for the sensor. The numerical value of the raw count is the digital representation of the sensor capacitance, and increases as the capacitance increases. Sensitivity is a measure of how much the output will change for a given change on the input. The sensitivity of the CapSense sensor has units of counts-per-pF.

Figure 2-5. Sensing Algorithm Output



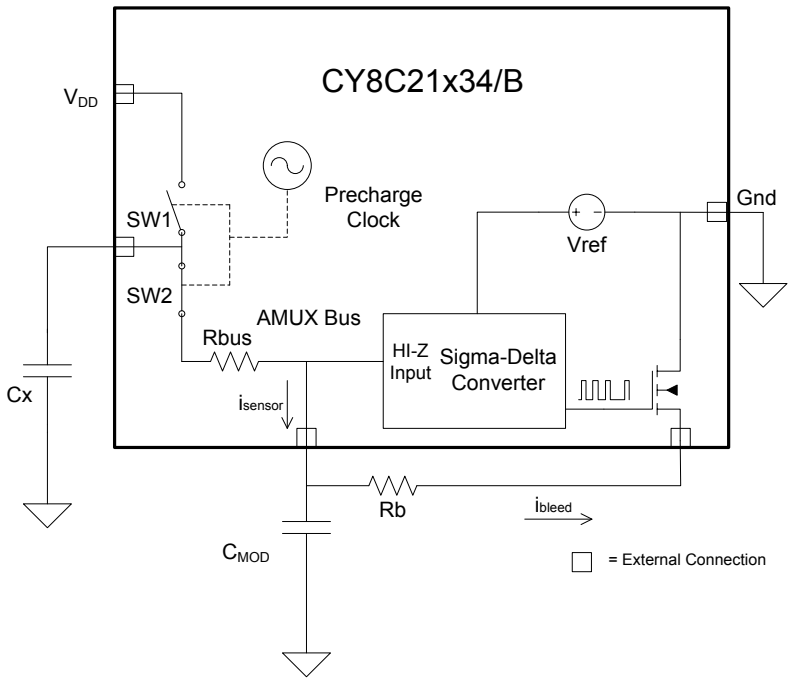
2.3.3 CapSense with Sigma Delta Modulator (CSD)

Cypress's CSD method uses a switched capacitor circuit on the front end of the system to convert the sensor capacitance to an equivalent resistor. A Sigma-Delta modulator converts the current measured through the equivalent resistor into a digital count. When a finger is on the sensor, the capacitance increases and the equivalent resistance +decreases. This causes an increase in current through the resistor, resulting in an increase in the digital count.

The CSD method requires a single dedicated pin and a single external component, C_{MOD}, or two dedicated pins and two external components, C_{MOD} and R_B, depending on what CapSense controller family is selected.

Figure 2-6 shows the CSD configuration for the CY8C21X34 CapSense controller family, and the family requires two external components and two dedicated pins.

Figure 2-6. CY8C21X34/B CSD Block Diagram



The CY8C20XX6A/AS/H family of CapSense controllers uses a single external component C_{MOD} and a single dedicated pin.

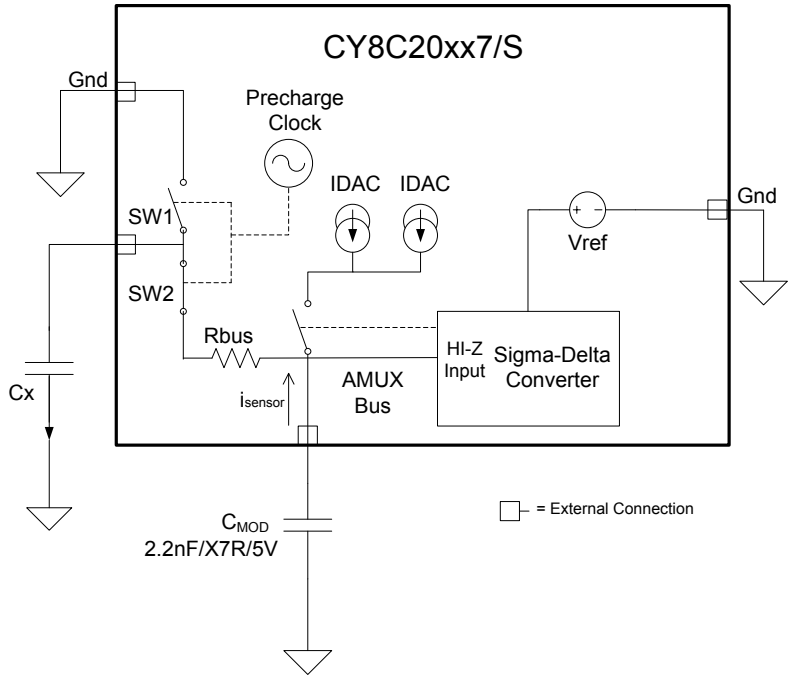
2.3.4 CapSense Plus with Sigma Delta Modulator (CSDPLUS)

Cypress's CSDPLUS method uses a switched capacitor circuit on the front end of the system to convert the sensor capacitance to an equivalent resistor. A Sigma-Delta modulator converts the current measured through the equivalent resistor into a digital count. When a finger is on the sensor, the capacitance increases and the equivalent resistance decreases. This causes an increase in current through the resistor, resulting in an increase in the digital count.

The CSDPLUS method has a change in the overall architecture that enables higher sensing sensitivity than CSD.

The CY8C20XX7/S family of CapSense controllers uses a single external component CMOD and a single dedicated pin.

Figure 2-7. CSD PLUS Block Diagram



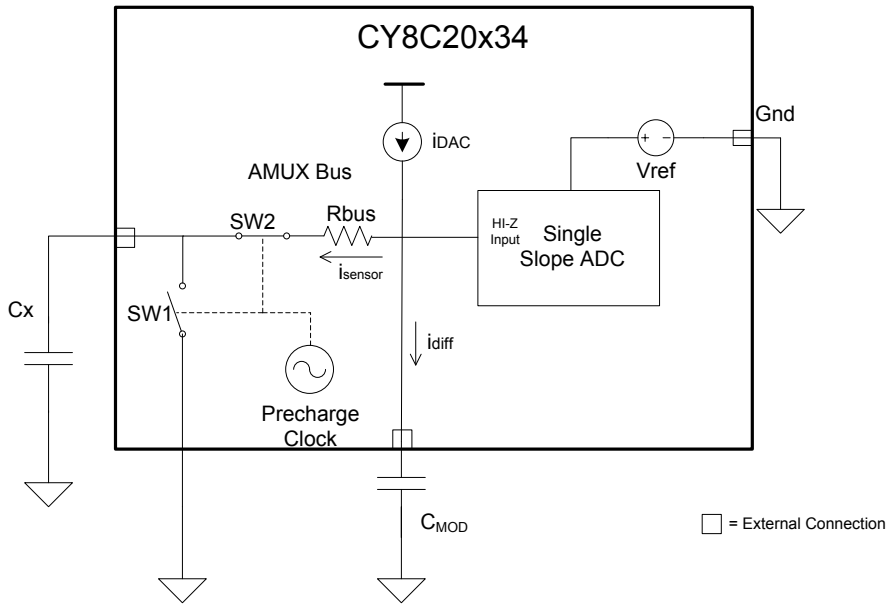
2.3.5 CapSense Successive Approximation Electromagnetic Compatible (CSA_EMC)

Cypress's CSA_EMC method also uses a switched-capacitor circuit on the front end of the system to convert the sensor capacitance to an equivalent resistor. An internal constant current source called the iDAC is calibrated with a successive approximation procedure until an equilibrium voltage develops across the integration capacitor C_{MOD}. This equilibrium voltage is measured using a single slope ADC. When you place a finger on the sensor, the capacitance increases. Further, this causes the equilibrium voltage on C_{MOD} to decrease and the ADC output increases that result in an increase in the digital count.

The CSA_EMC method requires a single dedicated pin and a single external component, C_{MOD}, which is an integration capacitor that is used by the single-slope ADC.

The CSA_EMC CapSense algorithm is developed to work well in the presence of RF interference. CSA_EMC is used in applications where CapSense is exposed to conducted interference, AC noise, and other noise sources such as inverters, transformers, and power supplies. For more detail on CapSense EMC feature, see Electromagnetic Compatibility (EMC) Considerations.

Figure 2-8. CSA_EMC Block Diagram



For a detailed discussion of Cypress's CSD and CSA_EMC sensing methods, see the respective device design guides. Table 5-1 shows the CapSense controller offerings and the sensing method supported for each. Table 5-2 compares these two CapSense sensing technologies in detail.

2.3.6 SmartSense_EMCplus (SmartSense Electromagnetic Compatible with CY8C20XX7/S family)

In addition to the SmartSense_EMC Auto-Tuning algorithm discussed previously, the SmartSense_EMCplus User Module includes an improved sensing architecture for greater sensitivity. With increased sensor signal, the SmartSense_EMCplus User Module improves the robustness of the capacitive sensing algorithm/circuit against high-frequency conducted and radiated noise.

Every electronic device must comply with specific limits for radiated and conducted external noise. These limits are specified by several regulatory bodies (for example, FCC, CE, U/L, and so on). An excellent PCB layout design, power supply design, and system design is mandatory for a product to pass the conducted and radiated noise tests. In some instances, ideal design practices cannot be followed because of the cost and form factor limitations of the product. SmartSense_EMCplus with superior noise immunity is well suited and useful for such applications that require liquid tolerance or proximity to pass radiated and conducted noise tests.

2.4 CapSense Tuning

Optimal CapSense system performance depends on board layout, button dimensions, overlay material, and application requirements. These factors are discussed in Design Considerations. In addition to these factors, switching frequency and threshold levels must be carefully selected for robust and reliable performance. Tuning is the process of determining the optimum values for these parameters. Tuning is required to maintain high sensitivity to touch and to compensate for process variations in the sensor board, overlay material, and environmental conditions.

Before we move further, a number of terms need to be defined to understand the tuning process.

2.4.1 Definitions

- **Raw Count:** As seen in Figure 2-9, sensor capacitance is converted into a count value by the CapSense algorithm. The unprocessed count value is referred to as raw count. Processing of the raw count results in ON/OFF states for the sensor.
- **Baseline:** The baseline is an estimate of the average sensor count level when the sensor is in the OFF state. The baseline provides a reference level for the ON/OFF comparison.
- **Difference Count:** Subtracting the baseline level from the raw count produces the difference count that is used in the ON/OFF decision process. The actual baseline is dynamically adjusted by the user module to compensate for environmental changes through a process called baseline update.

The thresholds are offset by a constant amount from the baseline level. The thresholds have the following functions:

- **Noise Threshold:** If the difference count is below the noise threshold, then the baseline is updated.
- **ON Threshold (Finger Threshold + Hysteresis):** If the difference count is increasing and exceeds the level of (Finger Threshold + Hysteresis), then the sensor state changes from OFF to ON.
- **OFF Threshold (Finger Threshold - Hysteresis):** If the difference count is decreasing and drops below the level of (Finger Threshold - Hysteresis), then the sensor state changes from ON to OFF.

2.4.2 Signal-to-Noise Ratio (SNR)

Signal is a generic engineering term that can have many meanings. For the capacitive sensor application under consideration for CapSense applications, signal is defined as the change in the average sensor output between the OFF and ON states when the rising edge of the difference count starts below the noise threshold.

Noise is another term that has many meanings. The following discussion presents a definition of CapSense noise that uses a simple mathematical model of the sensor output over time.

When the sensor is in the OFF state, the counts, X(t), can be modeled by an average count and a noise component.

$$X(t) = X0 + N0(t)$$ Equation 3

- X0 is the average of X(t)
- N0(t) is the noise component for t during the OFF state

The same model applies when the sensor is in the ON state.

$$X(t) = X1 + N1(t)$$ Equation 4

- X1 is the average of X(t)
- N1(t) is the noise component for t during the ON state

X0 is called the baseline level of the raw counts. The difference between X0 and X1 is called the signal, S.

$$S = X1 - X0$$ Equation 5

The noise components N0(t) and N1(t) are similar but not identical. For example, N1(t) usually contains a higher level of AC line noise in finger sensing applications compared to N0(t). This occurs because the human body acts as an antenna to 50 Hz and 60 Hz line noise, and the finger contact with the sensor overlay couples the noise into the CapSense system.

We define the noise level N as the worst case measured peak noise in the OFF state.

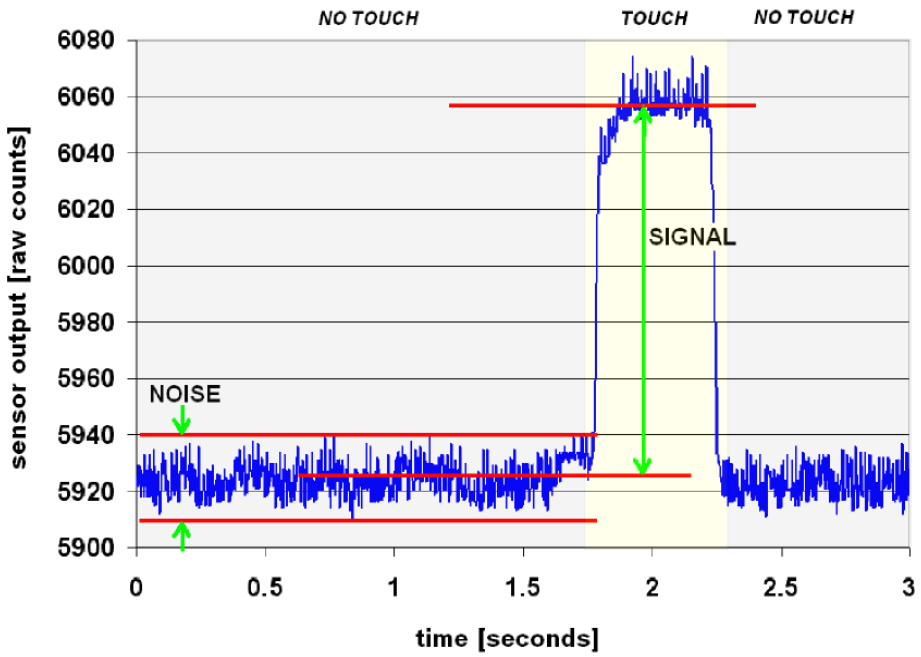
$$\mathrm{N} = \max(\mathrm{N0(t)}) = \max(\mathrm{X(t)}) - \min(\mathrm{X(t)})$$ Equation 6

Thus, CapSense Signal-to-Noise Ratio, SNR, is defined as the ratio of signal (S) to noise (N).

$$SNR = S{:}N$$ Equation 7

For robust operation of CapSense, a minimum SNR of 5:1 is recommended.

Figure 2-9. Signal and Noise



2.4.3 Measuring SNR

SNR should be measured in the noise environment where CapSense is intended to be used. In other words, measure the system SNR under worst-case noise conditions.

The first step in measuring SNR is to monitor the raw count for each sensor. This can be done using data logging to a text file and plotting in a spreadsheet, or using the Cypress MultiChart GUI tool and I^{2}C-USB Bridge (see Chapter 7 - Resources for more details). Whatever the method, the raw count should be observed for SNR measurement. The difference count should not be used in the measurement of SNR since it is a function of the baseline update process, which involves filtering (filling the "bucket") and nonlinear threshold events.

Another factor to consider is how the signal is produced. The worst-case ON and OFF scenario should be used when measuring SNR. If the system is designed to sense the presence of a finger, then measure SNR with a light touch of the sensor area, and position the contact point slightly off-center. For automated testing, a worst-case finger touch (0.1 pF) can often be simulated by an equivalent metal disc that is the size and shape of a small coin.

As an example of measuring SNR, consider the raw count waveform in Figure 2-9.

X0 = 5925 counts

X1 = 6055 counts

S = 130 counts

N = 5940 - 5925 = 15 counts

SNR = 130:15 = 8.6:1

2.4.4 SmartSense Auto-Tuning

2.4.4.1 What Is SmartSense?

Tuning the touch sensing user interface is a critical step in ensuring proper system operation and a pleasant user experience. The typical design flow involves tuning the sensor interface in the initial design phase, during system integration, and finally production fine-tuning before the production ramp. Because tuning is an iterative process, it can be time-consuming. SmartSense Auto-Tuning helps to simplify the user interface development cycle. In addition, the method is easy to use and reduces the design cycle time by eliminating the tuning process throughout the product development cycle, from prototype to mass production.

2.4.4.2 What Does SmartSense Do?

SmartSense tunes each CapSense sensor automatically at power up and then monitors and maintains optimum sensor performance during runtime. The number of parameters to be tuned is reduced from 17 in CSD to four with SmartSense.

- **Power-up tuning:** SmartSense tunes the parameters of each sensor based on the individual sensor parasitic capacitance to get the desired sensitivity for the sensor.
- **Runtime tuning:** Noise in the system is measured dynamically. The thresholds are adjusted accordingly for each sensor to overcome false triggering due to dynamic variations in noise in the CapSense system.

2.4.4.3 How and Where is SmartSense Helpful?

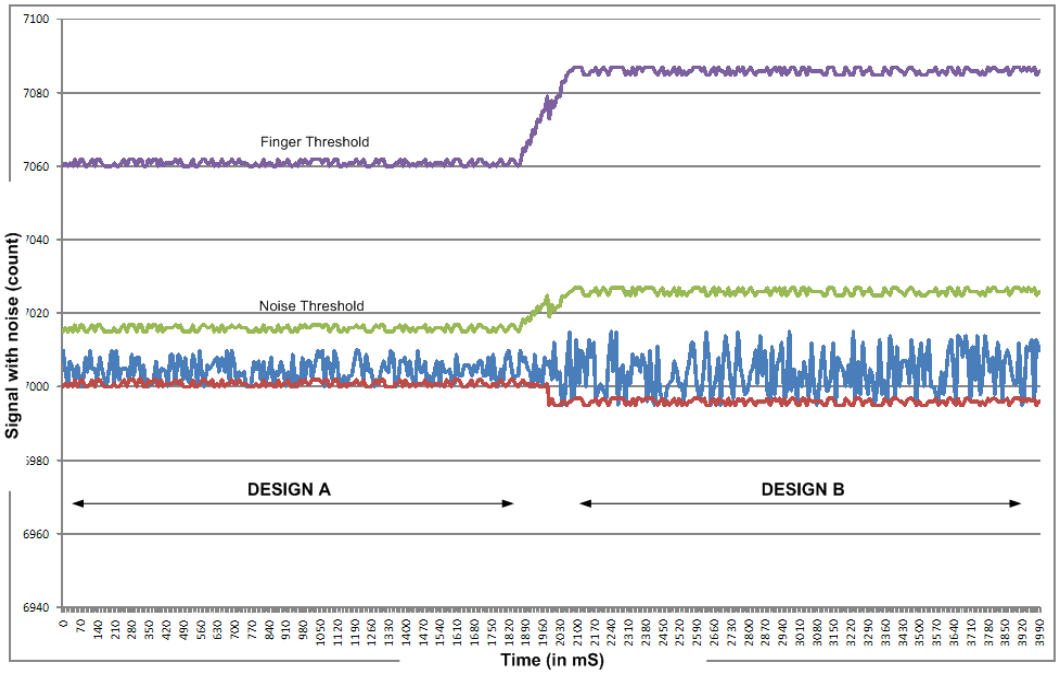
SmartSense technology adapts for manufacturing variations in PCBs, overlays, and noise generators, such as LCD inverters, AC line noise, and switch-mode power supplies and automatically tunes them out. SmartSense handles changes in system environment, such as temperature, humidity, and noise sources such as RF, SMPS, LCD Inverter, and AC line noise. In systems with special requirements or very high C_P (greater than 45 pF), Auto-Tuning may not be the ideal solution.

The following sections describe scenarios in which SmartSense is instrumental in adapting to the external noise. By maintaining a robust signal-to-noise ratio, the false triggering of buttons is prevented.

2.4.4.3.1 Different Noise Levels in Different Designs

SmartSense technology dynamically tunes itself (adjusts noise and finger thresholds) for different noise environments. In Figure 2-10, Design A and Design B have different noise levels. To maintain a minimum SNR of 5:1, dynamic threshold adjustment is required. SmartSense does this automatically, allowing seamless transition from one model to another with minimal or no tuning required.

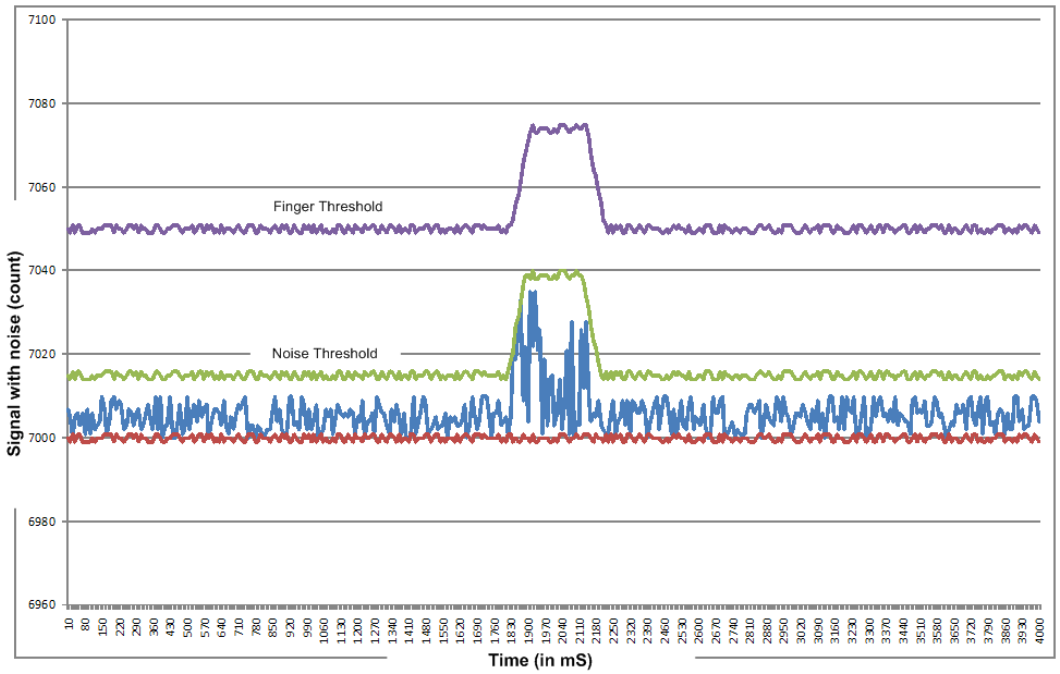
Figure 2-10. Different Noise Levels in Design A and B Being Compensated Automatically



2.4.4.3.2 *Noise Spikes During Production*

SmartSense technology also automatically tunes out the noise spikes (in production) that may not be seen during the design stage, as indicated in Figure 2-11. Noise spikes is a powerful feature prevents false button presses in the end system, which prevents a failure analysis for a mass production design.

Figure 2-11. Finger Threshold Dynamically Adjusted to Prevent False Button Touches



2.4.5 SmartSense_EMC (SmartSense Electromagnetic Compatible)

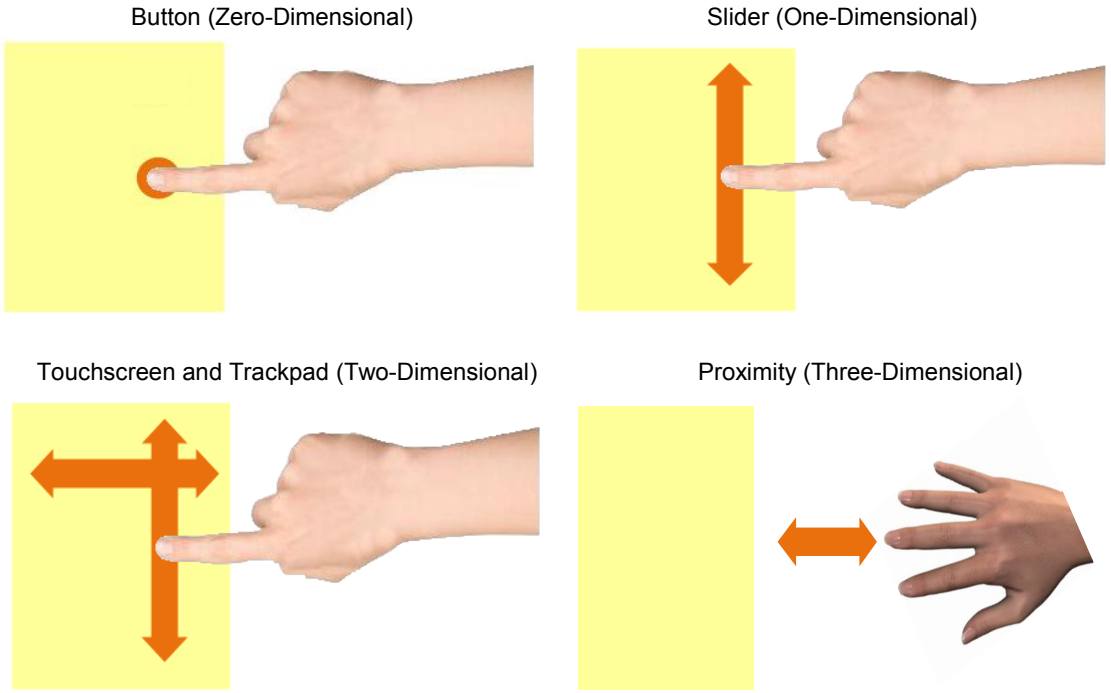
In addition to the SmartSense Auto-Tuning algorithm discussed previously, the SmartSense_EMC user module includes a unique algorithm to improve robustness of capacitive sensing algorithm/circuit against high-frequency conducted and radiated noise.

Every electronic device must comply with specific limits for radiated and conducted external noise. These limits are specified by several regulatory bodies (for example, FCC, CE, U/L, and so on). An excellent PCB layout design, power supply design, and system design is mandatory for a product to pass the conducted and radiated noise tests. In some instances, ideal design practices cannot be followed because of the cost and form factor limitations of the product. SmartSense_EMC with superior noise immunity is well suited and useful for such applications to pass radiated and conducted noise tests.

2.5 Sensor Types

Capacitive sensors can be broadly classified into four categories: buttons, sliders, touchscreens/trackpad, and proximity sensors. Different sensor types cater to different market segments.

Figure 2-12. Types of Capacitive Sensors



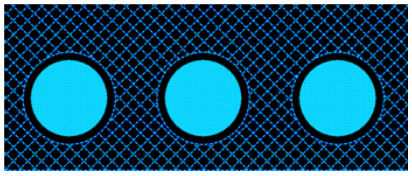
2.5.1 Buttons (Zero-Dimensional Sensors)

CapSense buttons are used in a wide variety of applications including: home appliances, medical devices, televisions, monitors, audio systems, photo frames, notebooks, home security systems, white goods, industrial products, and lighting controls. To get higher reliability, lower cost, and appealing industrial design, use CapSense buttons instead of mechanical buttons.

2.5.1.1 Simple Buttons

The simplest capacitive sensor consists of a copper pad connected to a CapSense controller pin with a trace. A button is defined as the combination of the copper sensor pad and the nonconductive overlay material. The button is surrounded by grounded copper hatch separated by an annular gap. Each button requires one I/O pin of the CapSense controller.

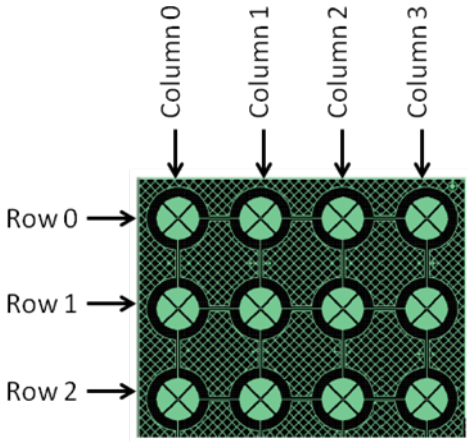
Figure 2-13. Typical Simple Buttons



2.5.1.2 *Matrix Buttons*

In applications requiring a high number of buttons such as a calculator keypad or a QWERTY keyboard, capacitive sensors can be arranged in a matrix. This allows a design to have more buttons than there are I/O pins on the CapSense controller.

Figure 2-14. Typical Matrix Buttons



A matrix button design consists of two groups of capacitive sensors: Row sensors and Column sensors. When a button is touched, it can be resolved by identifying the row and column sensors that are both in the TOUCH state. The number of buttons supported by the matrix is equal to the product of the number of rows and the number of columns.

$$Number\ of\ Matrix\ buttons\ = Number\ of\ Row\ Sensors\ \times Number\ of\ Column\ Sensors \quad \text{Equation 8}$$

Using a matrix button design can significantly reduce the number of I/O pins required. For example, the matrix in Figure 2-14 implements 12 buttons, but requires only seven I/O pins for sensors. Additional dedicated pins need to be assigned to external components, depending on the sensing method selected.

Matrix buttons can only be sensed one at a time. When more than one row or column sensor is in the TOUCH state, then the finger location cannot be resolved, and the situation is considered an invalid condition. Some applications require multiple buttons to be sensed simultaneously, such as a keyboard with a Shift, Ctrl, and Alt key. In this case, the Shift, Ctrl, and Alt keys should be designed as individual buttons, or should be changed to a mutual-capacitance sensor design.

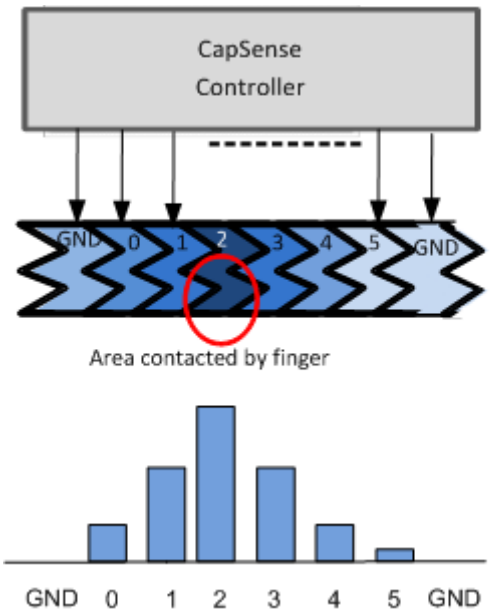
2.5.2 Sliders (One-Dimensional Sensors)

Sliders are used for controls requiring gradual adjustments. Examples include a lighting control (dimmer), volume control, graphic equalizer, and speed control. A slider is built using an array of capacitive sensors called segments that are placed adjacent to one another. Actuation of one segment results in partial actuation of physically adjacent segments. By using an interpolation method called a centroid, you can achieve a higher resolution than the number of slider segments. In a typical application, a slider with five segments can resolve at least 100 physical finger positions on the slider. High resolution makes for smooth transitions in light or sound as a finger glides across a slider.

2.5.2.1 Linear Sliders

In a linear slider, each CapSense controller I/O pin is connected to one slider segment. 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. Sensor data from multiple sensors improves the estimation of finger position. The maximum number of slider segments is a function of the number of available CapSense controller pins and the required response time.

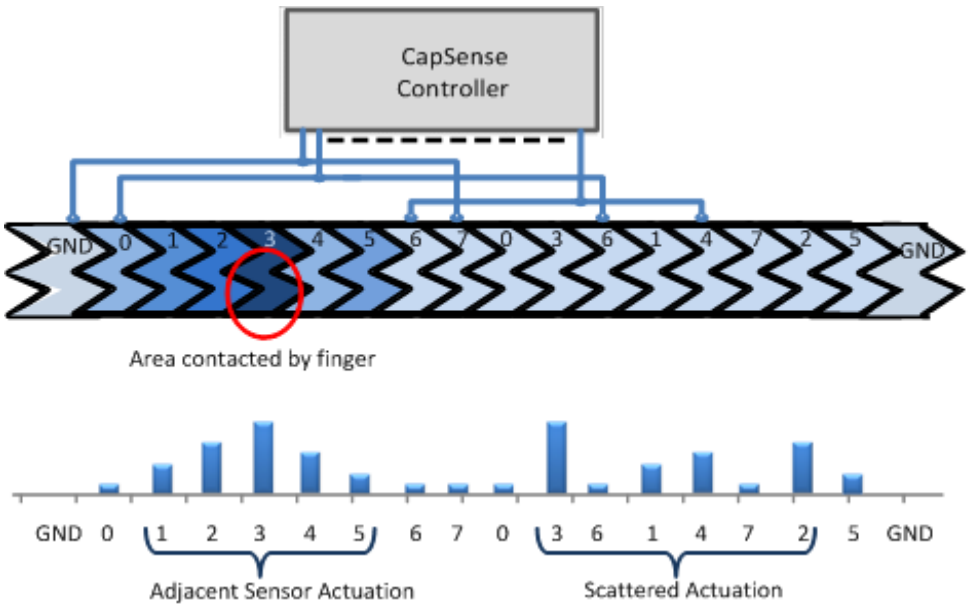
Figure 2-15. Linear Slider



2.5.2.2 Diplexed Sliders

In a diplexed slider, each CapSense controller I/O pin is connected to two different slider segments. This allows a design to have twice as many slider segments as there are I/O pins. For example, a diplexed 16-segment slider requires only eight CapSense controller I/O pins.

Figure 2-16. 16-Segment Diplexed Slider

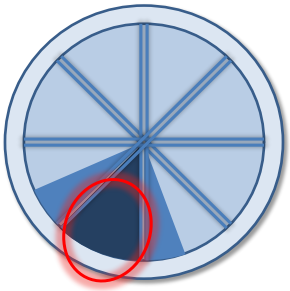


For a diplexed slider to work properly, the slider segments must be connected to the CapSense controller I/O pins in a pre-determined order. The first half of the segments are connected to the CapSense controller I/O pins sequentially (0, 1, 2 …7) and operate similar to a linear slider. The second half of the segments are connected to the same CapSense controller I/O pins in a non-sequential order. This order exploits the fact that activation of one segment results in partial actuation of neighboring segments. While slider actuation of one half of the slider results in aliasing on to the other half, the levels will be scattered in the untouched half. Sensing algorithms search for strong adjacent segment actuation and ignore scattered actuation to determine finger position on the slider accurately.

2.5.2.3 Radial Sliders

Radial sliders are similar to linear sliders in that finger position is estimated using data from adjacent sensors; however, radial sliders are continuous (does not have a beginning or end).

Figure 2-17. Radial Slider



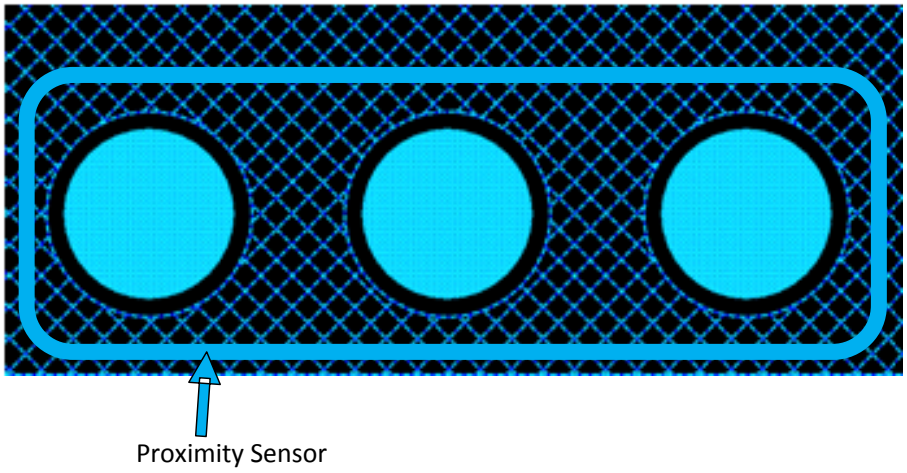
2.5.3 Touchscreens and Trackpads (Two-Dimensional Sensors)

Cypress's TrueTouch touchscreen solutions use mutual capacitance sensing. For more information on these products, see TrueTouch Touchscreen Controllers. Cypress also offers trackpad solutions. Contact your local Cypress sales office directly for more information. To find your local sales office, click here.

2.5.4 Proximity (Three-Dimensional Sensors)

Proximity sensors detect the presence of a hand or other conductive object before it makes contact with the touch surface. Imagine a hand stretched out to operate a car audio system in the dark. The proximity sensor causes the buttons of the audio system to glow through backlight LEDs when the user's hand is near. One implementation of a proximity sensor consists of a long trace on the perimeter of the user interface, as shown in Figure 2-18.

Figure 2-18. Proximity Sensor

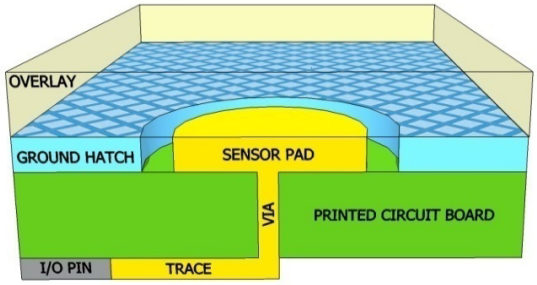


Another way to implement a proximity sensor is by ganging sensors together. This is accomplished by combining multiple sensor pads into one large sensor using firmware (see Firmware Component) to connect the sensors from the internal analog multiplexer bus. Make sure you do not exceed the C_P limit for the sensing method when ganging sensors together.

2.6 Sensor Construction

2.6.1 Field-Coupled Via Copper Trace (PCB)

Figure 2-19. Field Coupled Using PCB



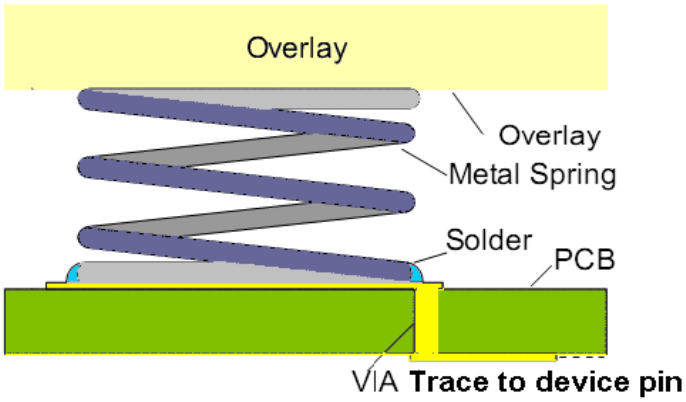
Features of a PCB-based design:

- Most common implementation
- Copper pads etched on the surface of the PCB act as sensor pads
- Electric field emanates from the copper sensor pad to ground plane
- No mechanical moving parts
- A nonconductive overlay serves as the touch surface for the button
- Ideal topology for simple flat panel designs
- Low BOM cost

For more information on springs, see Appendix A: Springs.

2.6.2 Field Coupled Via Spring/Gasket/Foam

Figure 2-20. Field Coupled via Spring



Features of a design based on springs/gaskets/foam:

- Electrical field coupled from PCB to overlay using a compressed spring, or conductive gasket or foam
- Conductive material itself acts as capacitive sensor pad
- No mechanical moving parts. Springs and foam do not move
- Coupled to touch sensor surface through nonconductive overlay
- Any conductive overlay serves as the button touch surface

- Ideal topology for curved, sloping, or otherwise irregular front panels
- Ideal for designs where touch sensor surface is physically separated from silicon or mother board
- Ideal for designs where CapSense and mechanical button combination is desired

For more information on springs, see Appendix A: Springs.

2.6.3 Field Coupled Via Printed Ink

Features of a design based on printed ink:

- Electric field coupled with printed patterns on a flexible substrate using conductive ink
- High series resistance due to higher ohms-per-square of printed ink compared to copper
- High parasitic capacitance due to thin PCB
- No mechanical moving parts, but substrate is flexible
- Coupled to touch sensor surface with a nonconductive overlay
- Ideal topology for flexible front panels
- Flexible PCB can be one-layer or two-layer film

2.6.4 Field Coupled Via ITO Film on Glass

Features of a design based on ITO film:

- Electric field coupled with printed or deposited patterns on glass
- Higher series resistance of ITO films compared to copper and printed ink
- No mechanical moving parts
- Ideal topology for graphical front panels

2.7 User Interface Feedback

Effective user interface designs include some feedbacks to the user when they are using the capacitive touch sense buttons. There are various forms of feedback, including visual, audio, and haptic (tactile). Depending on the user interface design, multiple types of feedback can be used in combination.

2.7.1 Visual Feedback

LEDs and LCDs provide visual feedback.

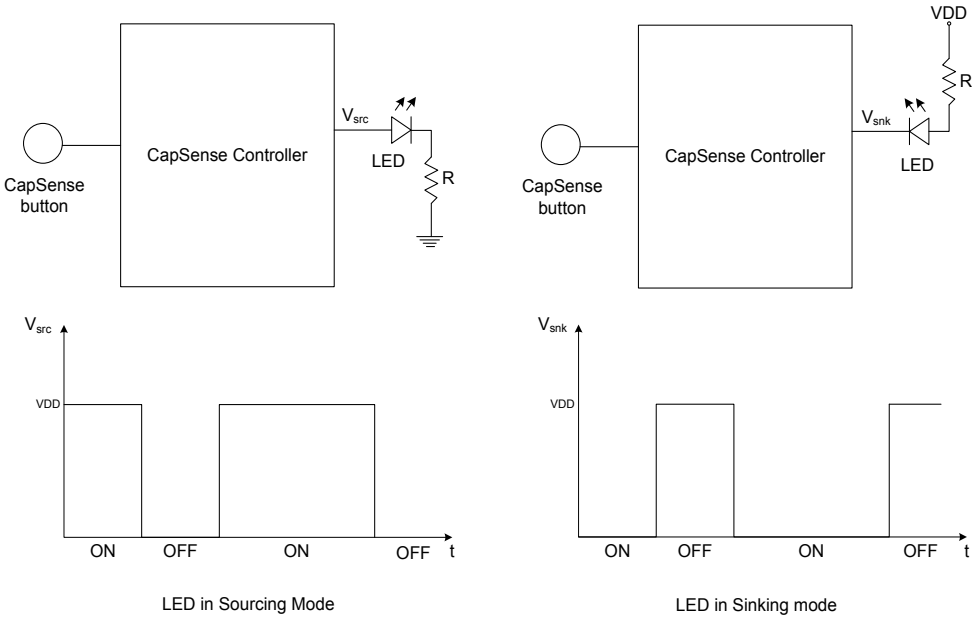
2.7.1.1 LED-Based Visual Feedback

Visual feedback is widely used in user interfaces. LEDs are used to indicate the status of buttons, sliders, and proximity sensors. LEDs can implement different effects when the sensor status changes.

2.7.1.1.1 LED ON/OFF

In visual feedback's simplest form, LEDs are turned ON or OFF in response to a finger touch. General-purpose I/Os are used to drive LEDs in either a sourcing or sinking configuration, as shown in Figure 2-21.

Figure 2-21. LED Sourcing and Sinking Configuration



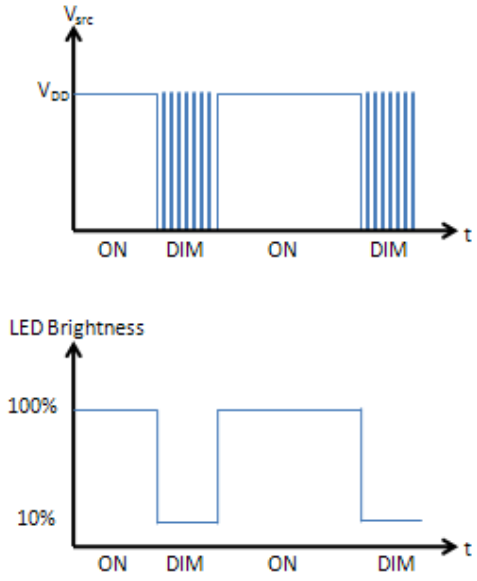
2.7.1.1.2 *Advanced LED Effects*

For user interfaces that require sophisticated visual effects, a single hardware PWM or timer can be used to drive the LEDs. By varying the duty cycle of the PWM output, you can achieve advanced effects, such as variable LED brightness, fading, and breathing. A single hardware PWM or timer can be used to drive multiple LEDs.

2.7.1.1.3 *LED Brightness*

By varying the duty cycle of the PWM output, you can adjust the LED brightness as shown in Figure 2-22. This enables adjusting your user interface brightness in response to ambient lighting conditions.

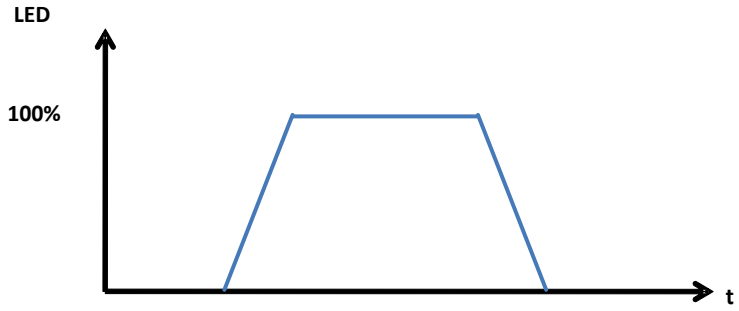
Figure 2-22. LED Brightness Control



2.7.1.1.4 LED Fading

By gradually changing the duty cycle between LED states, you can achieve a fading effect (see Figure 2-23). For example, the LED appears to “fade in” (from OFF to ON) when the duty cycle is increased in a series of small steps.

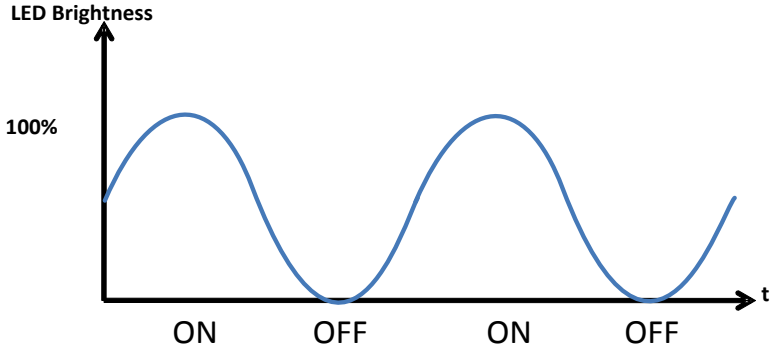
Figure 2-23. LED Fading



2.7.1.1.5 LED Breathing

Gradually increasing and decreasing the duty cycle between two levels on a continuous basis makes the LED appear to “breathe”, as shown in Figure 2-24. LED breathing is useful when a system is in idle or stand-by mode. For example, a power button can appear to breathe to alert the user that it is active and can be operated.

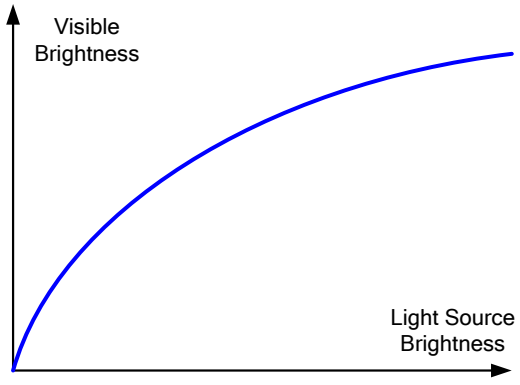
Figure 2-24. LED Breathing



Human Eye Sensitivity

The human eye’s sensitivity to the brightness of a light source looks similar to a logarithmic function (Figure 2-25). To provide a visible linear brightness change, change the PWM duty cycle in an exponential manner.

Figure 2-25. Human Eye Brightness Perception versus LED Luminous Flux



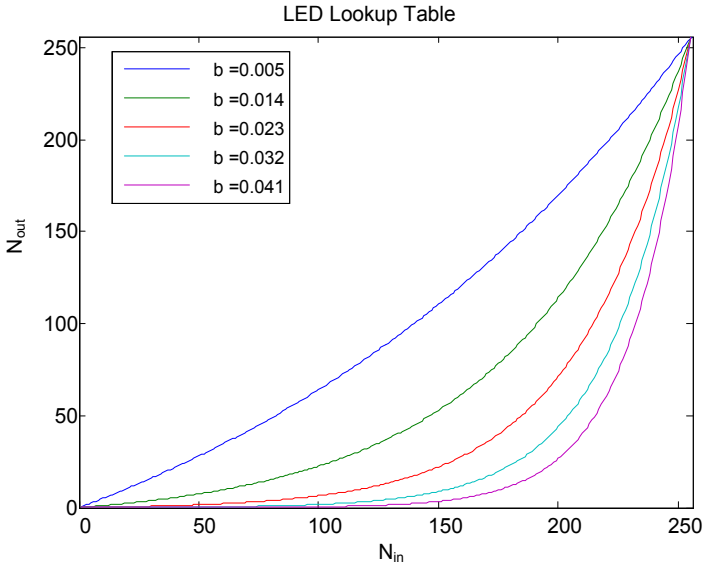
The linear brightness levels are transferred to exponential duty cycle values using the lookup table. The following expression is used for the conversion:

$$N_{out} = A(exp(N_{in} . b) - 1)$$ Equation 9

$$A = \frac{N_{max}}{exp(N_{max} . b) - 1}$$ Equation 10

Figure 2-26 illustrates the table graphs at different values of parameter *b*. Note that N_{MAX} is set to 255. This expression converts an 8-bit unsigned byte value to the same range. The figure shows that the transfer characteristic becomes more exponential as the *b* parameter increases.

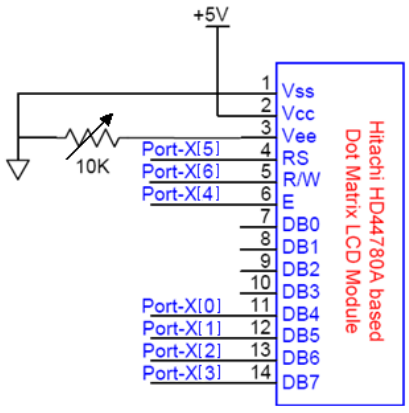
Figure 2-26. LED Duty Cycle Conversion Table



2.7.1.2 LCD-Based Visual Feedback

LCDs provide visual feedback for CapSense buttons and sliders. The main advantage of using an LCD is that it can provide more information along with the feedback for each button press event. PSoC has built-in user modules for driving Hitachi HD44780A LCD module. This user module supports high-level and low-level APIs that can directly display data on the screen with ease. Figure 2-27 shows the typical connection for using the Hitachi HD44780A LCD module.

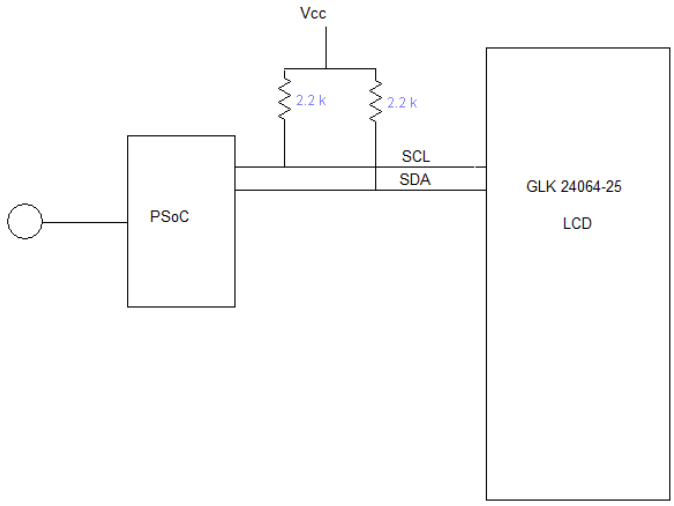
Figure 2-27. Hitachi Dot Matrix LCD Pin Connections



PSoC can also control LCDs through I^2C. The I2CHW user module available in PSoC controls the GLK 24064-25 WB graphics LCD. Sending the commands through the I^2C bus is simplified. The CSD user module is configured to scan a set of buttons and any button press event initiates an I^2C transfer from the PSoC to the LCD as a feedback.

The typical circuit diagram for driving the GLK 24064-25 WB graphics LCD follows.

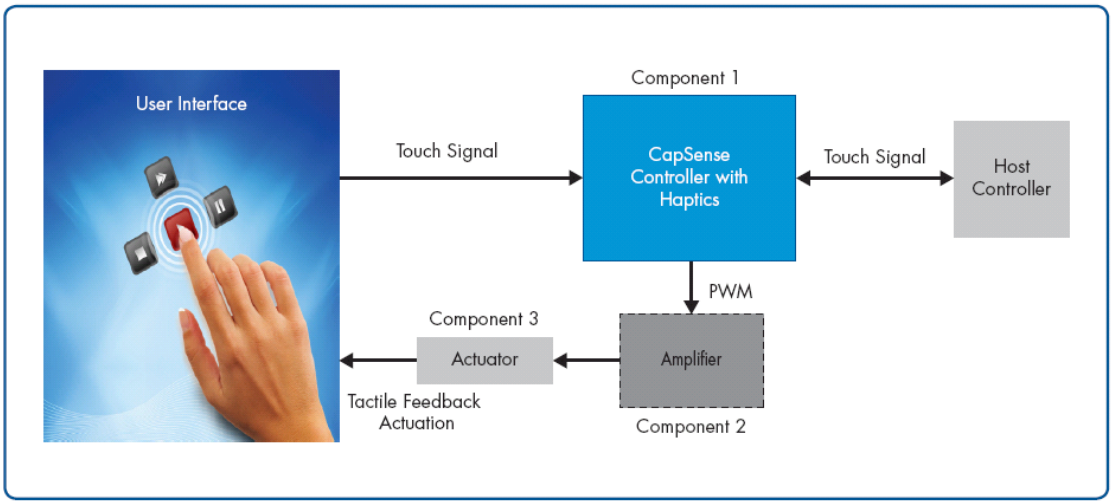
Figure 2-28. Implementing LCD Feedback with CapSense on CapSense Plus



2.7.2 Haptic Feedback

Haptic, or tactile, feedback uses vibration to let you know that the system has detected a finger touch.

Figure 2-29. Cypress Haptics Ecosystem



Vibrations are created by an actuator (DC motor) with eccentric rotating mass (ERM). When the end user touches a CapSense button, a touch signal is sent to the CapSense controller. The CapSense controller processes the data and drives the actuator to get a particular haptic effect. The CY8C20XX6H CapSense controller can generate 14 predefined haptics effects. PWM and Timer user modules are used to generate a haptics signal to drive the actuator. The PWM signal is updated periodically. The Timer user module is used to generate a 5-ms periodic (for haptic-effects generation) interrupt to update the effect that is currently being played. The Amplifier supplies the required current to the actuator. The firmware can also generate the 5-ms timer to save a digital block, but this increases the CPU load. The PWM user module is configured to generate output signals of frequency greater than 22 kHz to drive the DC motor. This frequency is outside the audible range of the human ear.

The Haptics User Module datasheet has more information about the 14 predefined haptics effects and also provides a code example. This document is available at http://www.cypress.com/?rID=55635.

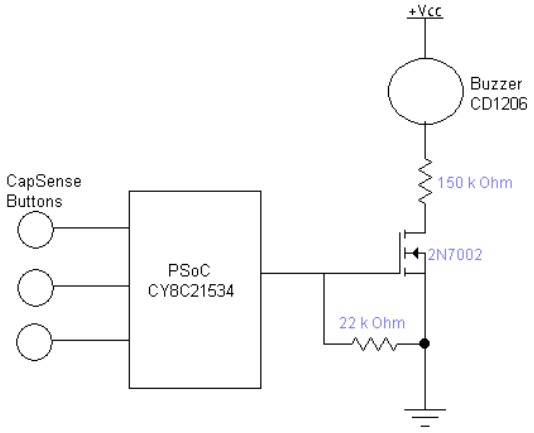
To know more about CY8C20XX6H, refer to the device datasheet at http://www.cypress.com/?rID=50279.

If you need a level 1 kit or any further assistance, contact capsense@cypress.com.

2.7.3 Audible Feedback

Audible feedback for CapSense buttons is implemented using a buzzer. The pulse-width modulator (PWM) is used to output the PWM signal required for driving the buzzer as specified in the buzzer datasheet. The PWM user module available in PSoC is used for this purpose. PSoC can implement CapSense through its CSA and CSD algorithms. The CSD user module is configured to scan a set of buttons and sliders. When a button press event occurs, the feedback is given by driving the buzzer at a particular intensity level. The circuit diagram for implementing the buzzer feedback follows.

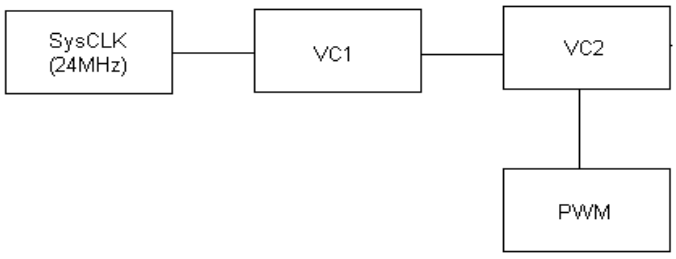
Figure 2-30. Implementing Audible Feedback for CapSense in CapSense Plus/CapSense Express Devices



2.7.3.1 CapSense with Audible Feedback Configuration

Select CSD and PWM8 user module from the user module list. Set the parameters for the CSD user module as shown in Table 2-1. Calculate the PWM user module parameters based on the buzzer's resonant frequency. For example, consider the buzzer CD1206 with a resonant frequency of 2.4 kHz. A 2.4-kHz PWM signal with a 50-percent duty cycle is required to drive the buzzer to produce proper audio feedback.

Figure 2-31. PWM Clock Divider Calculation



To calculate the clock dividers to obtain a 2.4-kHz PWM output, see Figure 2-31.

The system clock is set to 24 MHz. The required PWM output frequency is 2.4 kHz. For this reason:

$$SysCLK\ /\ \left(N1\ \times N2\ \times\ (PeriodValue\ +1)\right)\ =2.4\ kHz$$ Equation 11

Where, N1 and N2 are the VC1 and VC2 clock divider values, respectively. Period Value is the value of period register input to the PWM.

That means:

$$24\ MHz\ /\ \left(N1\ \times N2\ \times\ (PeriodValue\ +1)\right)\ =2.4\ kHz$$ Equation 12

Rearranging the equation gives the following result:

$$N1 \times N2 \times (PeriodValue + 1) = 10000$$ Equation 13

The previous equation has various integral solutions. For simplicity, this example uses N1 = 4 and N2 = 10. Substituting these values in the previous equation generates the values:

$$PeriodValue + 1 = 10000(4 \times 10) \equiv 250$$ Equation 14

Thus, Period Value is 249. To have a 50-percent duty cycle, the Compare value for the PWM is set as:

$$(PeriodValue + 1)/2 = (249 + 1)/2 \equiv 125$$ Equation 15

User module parameters are matched as shown in the following table.

Table 2-1. PWM8 User Module Parameters

Parameter	Value
Name	PWM
Configuration	8 bit
Clock	VC2
Period	249
Pulse Width	125
Compare Type	Less than
Interrupt Type	Compare True
Clock Sync	Sync to SysClk

Note that the CSD user module automatically varies the clock dividers based on scan speed and resolution settings of the CSD user module. Therefore, re-enter the clock dividers every time the PWM module is invoked by writing the values directly to the configuration register OSC_CR1. For details about the Configuration register OSC_CR1, see the Technical Reference Manual.

The clock dividers VC1, VC2, and VC3 vary with the CSD scan speed and resolution, as shown in Table 2-2 and Table 2-3.

Table 2-2. Scan Speed Versus VC1 Divider

Scanning Speed	VC1
Ultra fast	1
Fast	2
Normal	4
Slow	8

Table 2-3. Resolution Versus VC2 and VC3 Clock Dividers

Resolution Bits	VC2	VC3
9	8	16
10	8	32
11	8	64
12	8	128
13	8	256
14	8	256
15	8	256
16	8	256

2.8 Liquid Tolerance

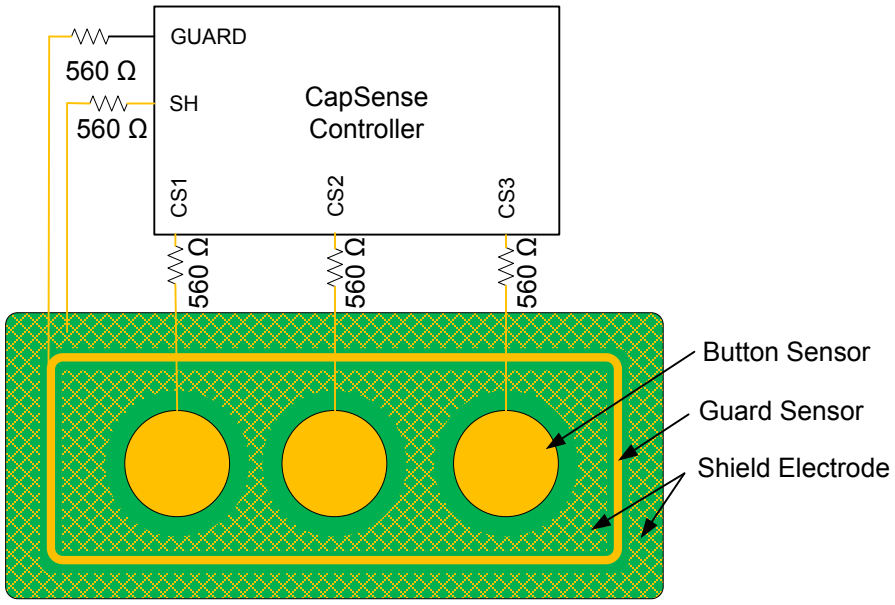
CapSense is used in a variety of applications such as Home Appliances, automotive, and industrial applications. These applications require robust CapSense operation even in the presence of mist, moisture, water, ice, and humidity changes. In a CapSense design, false sensing of touch may happen due to the presence of a film of liquid or liquid droplets on the touch surface. Cypress's CapSense sensing method can compensate for variation in raw count due to mist, moisture, water, ice, and humidity changes and provide a robust, reliable, CapSense operation.

Figure 2-32. CapSense-based Touch User Interface in a Washing Machine with Liquid Tolerance

To compensate for changes in raw count due to mist, moisture, and humidity changes, the CapSense sensing method continuously adjusts the baseline of the sensor to prevent sensor false triggers. To compensate for changes in raw count due to a liquid droplet or liquid flow, you should implement a shield electrode and a Guard Sensor to provide robust touch sensing, as Figure 2-33 shows.

When liquid droplets are present on the touch surface and if shield electrode is implemented, the CapSense system can reliably work even in the presence of liquid droplets and report sensor ON/OFF status. When there is a liquid flow or a liquid pool on the touch surface, the CapSense system detects the liquid by using a guard sensor and disables the scanning for all other sensors in the system to prevent false triggers. Therefore, when there is a liquid flow or liquid pool on the touch surface, the CapSense system will not detect a finger touch as long as the liquid is present on the touch surface.

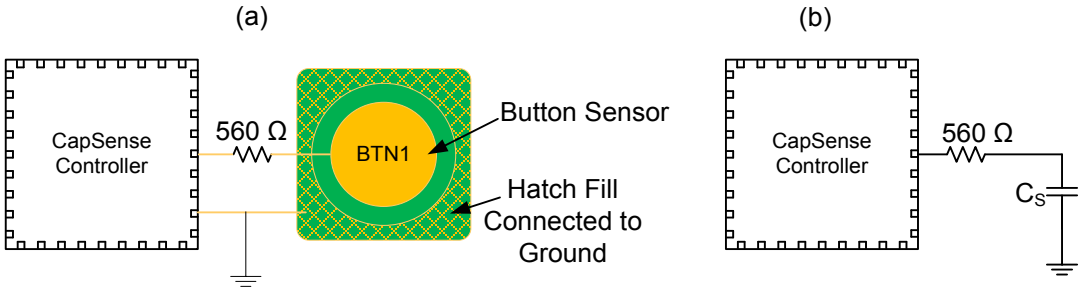
Figure 2-33. Shield Electrode (SH) and Guard Sensor (GUARD) Connected to CapSense Controller



2.8.1 Effect of Liquid Droplets and Liquid Stream on CapSense

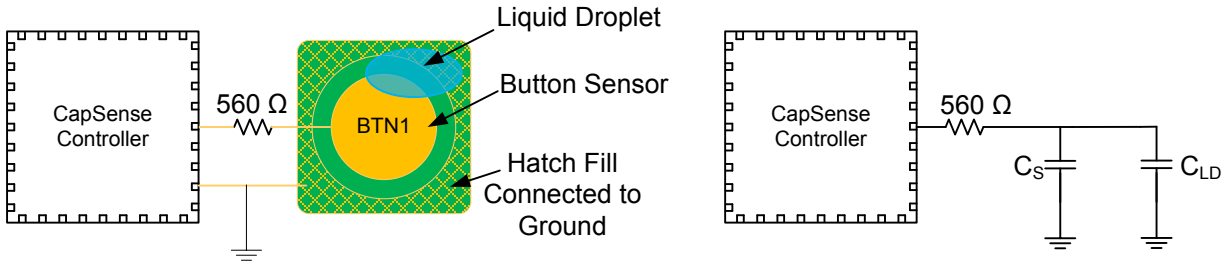
To understand the effect of a liquid droplet or a liquid stream on a CapSense sensor, consider a CapSense system in which the hatch fill around the sensor is connected to ground, as Figure 2-34 (a) shows. Surrounding the sensor with a hatch fill connected to ground improves the noise immunity of the sensor. The parasitic capacitance of sensor is denoted as C_S in Figure 2-34 (b).

Figure 2-34. Typical CapSense System Layout



As shown in Figure 2-35, when a liquid droplet falls on the touch surface, due to its conductive nature, it provides a strong coupling path for the electric field lines to return to ground and therefore adds a capacitance C_{LD} in parallel to C_P. When the sensor is charged and discharged, the capacitance C_{LD} draws some amount of charge from the AMUX bus because of the nonzero voltage difference across the capacitance C_{LD}. This increases the overall capacitance seen by the CapSense circuitry and results in an increase in the sensor raw count. In some cases, where the liquid is highly conductive (salty water or water with high mineral content), the increase in raw count when a liquid droplet falls on the touch surface might be equal to the increase in raw count due to a finger touch and thus causes false triggers, as Figure 2-36 shows.

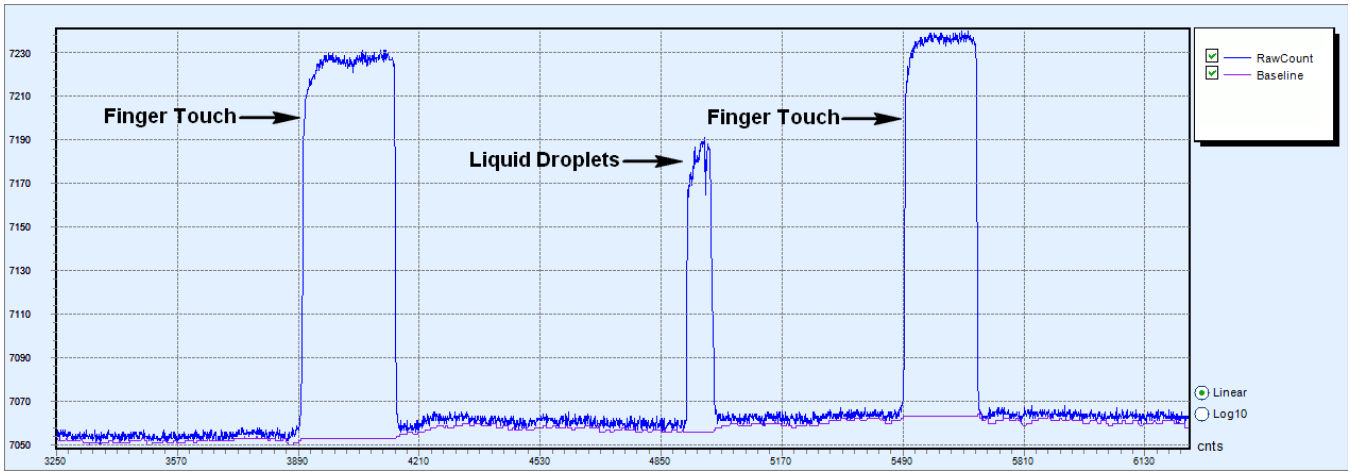
Figure 2-35. Capacitance Added by Liquid Droplet when the Hatch Fill Is Connected to Ground



C_S – Sensor Parasitic Capacitance

C_{LD} – Capacitance added by Liquid Droplet

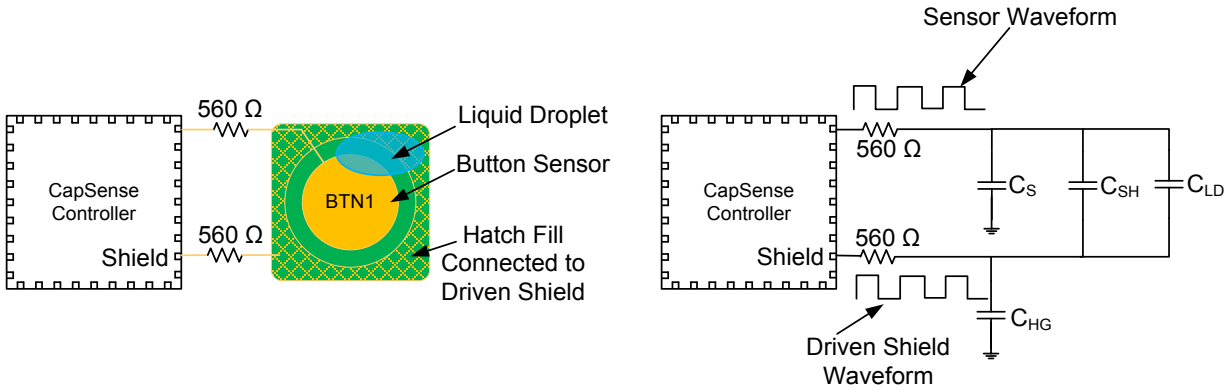
Figure 2-36. Effect of Liquid Droplet when the Hatch Fill Around the Sensor Is Connected to Ground



To compensate the capacitance added by the liquid droplet to the CapSense circuitry, you should drive the hatch fill surrounding the sensor with the driven-shield signal.

As shown in Figure 2-37, when the hatch fill surrounding the sensor is connected to the driven-shield signal and when a liquid droplet falls on the touch surface, because the voltage on both the sides of liquid droplet is kept at the same potential, the capacitance C_{LD} added by the liquid droplet is nullified. Because the voltage difference across the capacitance C_{LD} is zero, it will not draw any charge from the AMUX bus and therefore the increase in the raw count when a liquid droplet falls on the sensor will be very small, as Figure 2-38 shows.

Figure 2-37. Capacitance Added by Liquid Droplet when the Hatch Fill Around the Sensor Is Connected to Shield



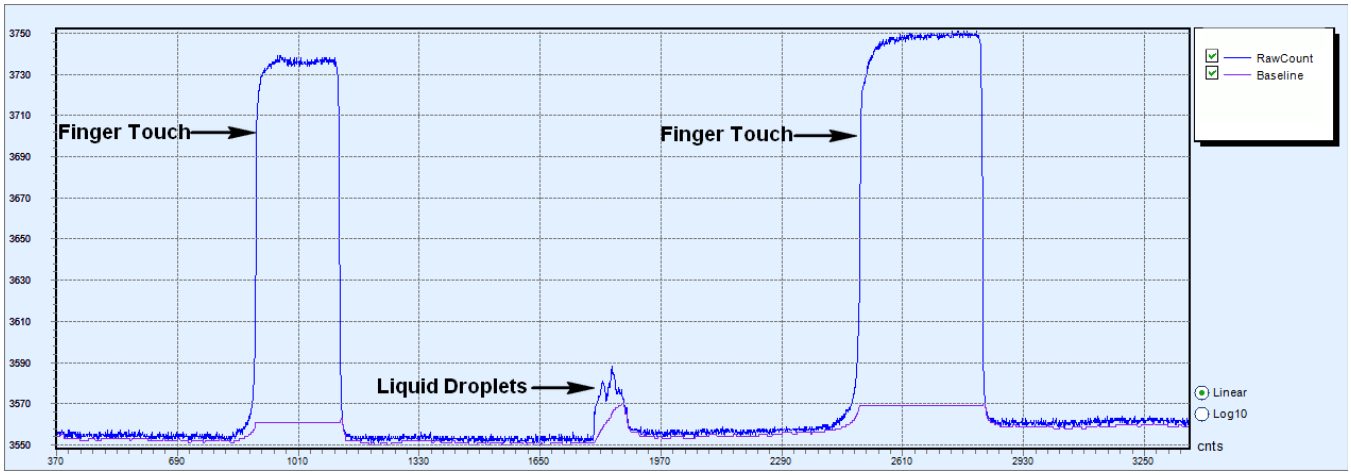
C_S – Sensor Parasitic Capacitance

C_{SH} – Capacitance Between Sensor and Hatch Fill

C_{HG} – Capacitance Between Hatch Fill and Ground

C_{LD} – Capacitance Added by Liquid Droplet

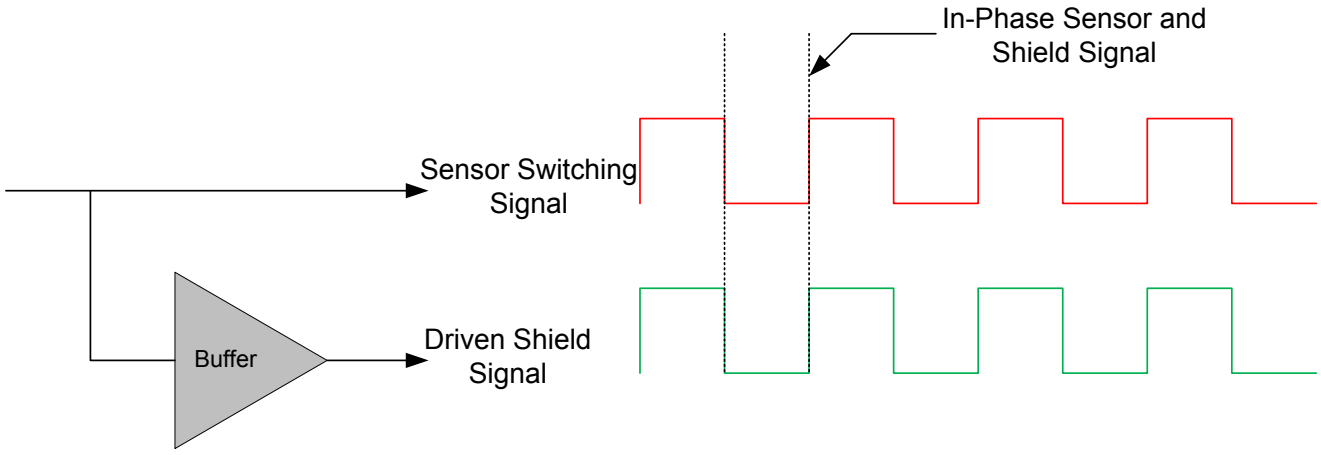
Figure 2-38. Effect of Liquid Droplet when the Hatch Fill Around the Sensor Is Connected to the Driven Shield



2.8.2 Driven-Shield Signal and Shield Electrode

The driven-shield signal is a buffered version of the sensor-switching signal, as Figure 2-39 shows. The driven-shield signal has the sample amplitude, frequency, and phase as that of the sensor-switching signal. The buffer provides sufficient current for the driven-shield signal to drive the high parasitic capacitance of the hatch fill on the PCB. When the hatch fill surrounding the sensor is connected to the driven-shield signal, it is referred as shield electrode. As explained in the Effect of Liquid Droplets and Liquid Stream on CapSense section, because the shield electrode is driven with a voltage which is same as the sensor-switching signal, the capacitance added by a liquid droplet when it falls on the touch surface will be nullified. To achieve the best liquid-tolerance performance, it is required that the driven shield signal has the same voltage and phase as that of the sensor-switching signal.

Figure 2-39. Driven-Shield Signal

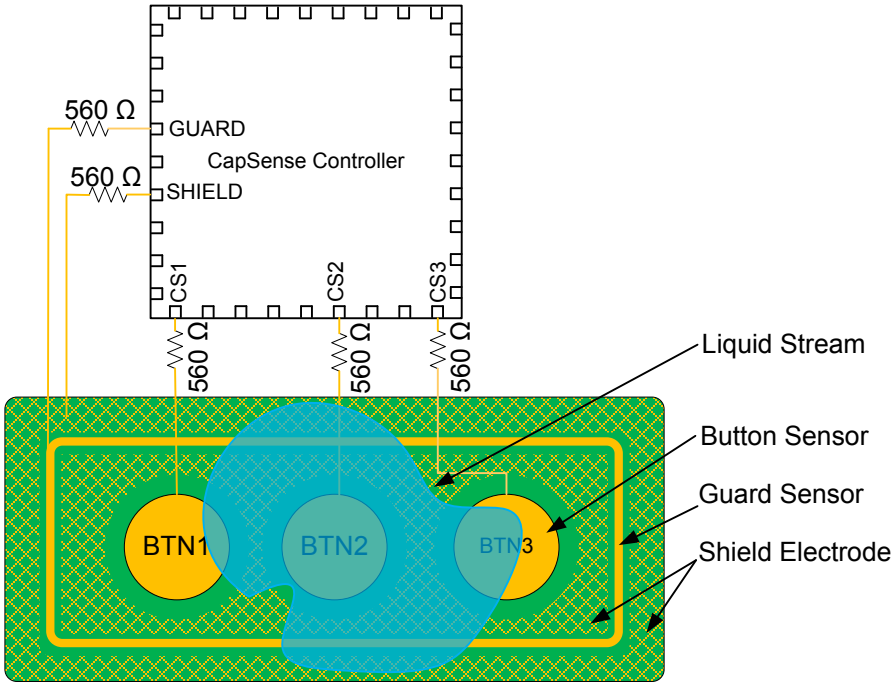


2.8.3 Guard Sensor

When a continuous liquid stream is applied to the touch interface, the liquid stream adds a large capacitance (C_{ST}) to the CapSense circuitry. This capacitance might be several times larger than C_{LD}. Because of this, the effect of the shield electrode is completely masked, resulting in the increase in sensor raw count potentially higher than that due to a finger touch. In such situations the guard sensor is useful to prevent a false touch-sensing.

A guard sensor is a copper trace that surrounds all the sensors on the PCB, as Figure 2-41 shows. A guard sensor is similar to a button sensor and is used to detect the presence of liquid stream. When a guard sensor is triggered, the firmware can disable the scanning of all other sensors in the system to prevent a false touch-sensing. Because the sensors are not scanned when the guard sensor is triggered, the touch cannot be detected when there is a liquid stream.

Figure 2-40. Capacitance Measurement with a Liquid Stream

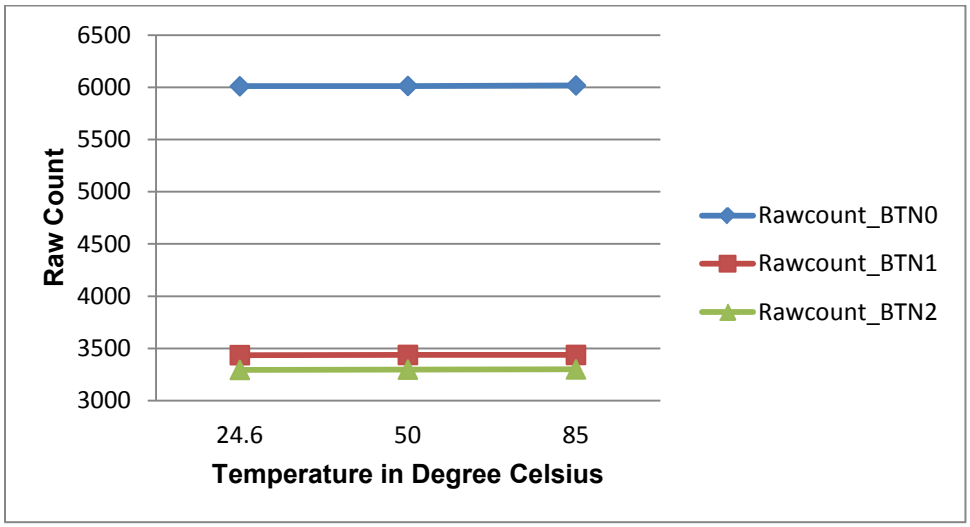


2.8.4 Effect of Liquid Properties on the Liquid-Tolerance Performance

In certain applications, the CapSense system has to work reliably in the presence of a variety of liquids such as soap water, sea water, and water with high mineral content. In such applications, it is recommended that you tune the CapSense parameters for the sensors by considering the worst-case shift in the raw count due to liquids on the touch surface. To simulate the worst-case condition, you can prepare a salty water solution by dissolving 40 gm of cooking salt (NaCl) in 1 liter of water and measure the shift in raw counts when water droplets falls on the sensor.

In applications such as an induction cook-top, there might be chances of hot water spilling on the CapSense touch surface. To determine the impact of temperature of a liquid droplet on the liquid-tolerance performance, tests were done with liquid droplets at different temperatures. Experiment results show that the effect of hot liquid droplets is the same as that of the liquid droplets at room temperature. This is because the hot liquid droplet cools down immediately to room temperature when it falls on the touch surface.

Figure 2-41. Raw Count Variation Versus Water Temperature



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

1. Choose a CapSense controller that supports the driven-shield signal. Refer to the CapSense Selector Guide to select the CapSense controller that supports the shield feature.
2. Follow the schematic and layout guidelines explained in the Shield Electrode and Guard Sensor section to construct the shield electrode and guard sensor.
3. Tune the guard sensor (if implemented) such that it is triggered only when there is a liquid stream. In the firmware, ensure that the sensors are not scanned when the guard sensor is triggered.

Refer to the individual CapSense design guides for detailed procedures of how to tune the CapSense parameters to achieve liquid tolerance. The application note AN92239 – Proximity Sensing with CapSense shows how to implement a proximity-sensing system with liquid tolerance for PSoC 4 device.

2.9 CapSense System Overview

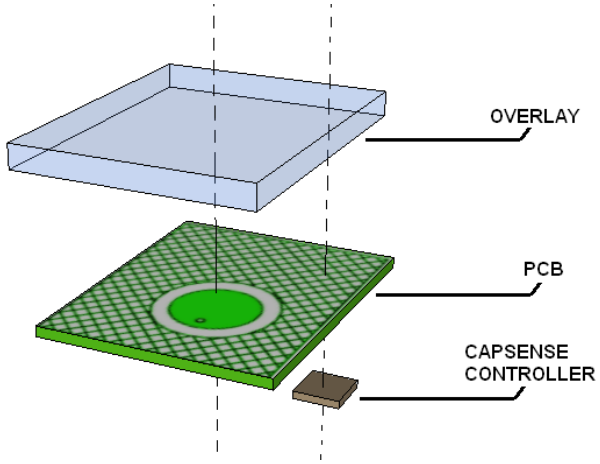
CapSense touch sensing solutions include the entire system environment in which they operate. This includes:

- Hardware components such as PCB and guard sensor
- Firmware component to process the sensor data

2.9.1 Hardware Component

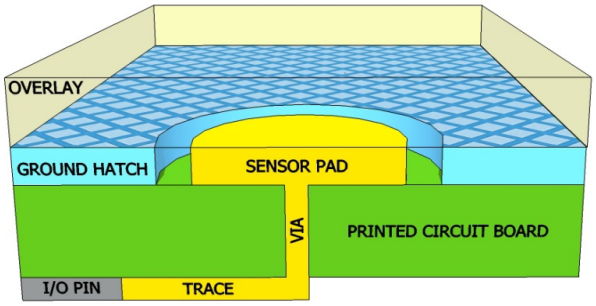
The CapSense controller resides within a larger system composed of a specially printed circuit board (PCB), and a touch-surface called the overlay that protects the PCB.

Figure 2-42. Exploded View of the CapSense Hardware



The capacitive sensor pads of a sensor board are formed by the PCB traces. The most common PCB format is a two-layer board with sensor pads and a hatched ground plane on the top, and the electrical components on the bottom. The electrical components include the CapSense controller and associated parts that convert the sensor capacitance into digital counts. Figure 2-43 shows a cross-sectional view of a two-layer board stack-up.

Figure 2-43. Two-Layer Stack-Up of a CapSense Board



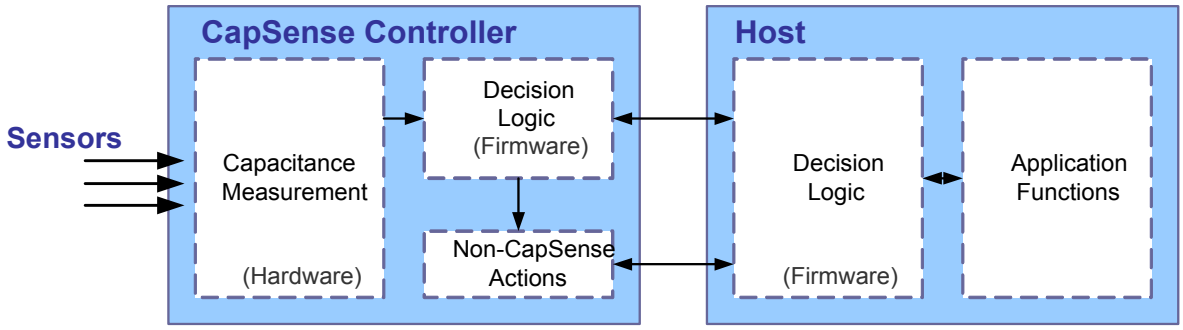
Four-layer design is an option when the board area must be minimized. PCB layout plays a very important role in CapSense system performance. Best practices are discussed in Design Considerations.

2.9.2 Firmware Component

Firmware is a vital component of the CapSense system that processes the raw count data and makes logical decisions. The amount of firmware development required for your application depends on which CapSense controller family that you select.

Devices from the CapSense Express family are fully configurable either through hardware or through I^2C and do not require any firmware development on the CapSense controller itself. These devices are appropriate for systems where the finger touch data is sent to a host for higher level processing; see Figure 2-44.

Figure 2-44. Example CapSense Express System Implementation



Devices from the CapSense and CapSense Plus families are fully programmable. These devices allow complex system-level integration. These controllers can process the raw count data as well as perform other system functions.

See the CapSense Product Portfolio and CapSense Selector Guide for additional details. Cypress's PSoC Designer accommodates firmware development in C and assembly language. See Resources for more information on this and other tools.

3. Design Considerations

When designing capacitive touch sense technology into your application, it is important to remember that the CapSense device exists within a larger framework. Careful attention to every level of detail from PCB layout to user interface to end-use operating environment will enable robust and reliable system performance.

3.1 Overlay Selection

In a CapSense design, overlay material is placed over the sensor pad to protect it from the environment and prevent direct finger contact.

3.1.1 Overlay Material

In the Self-Capacitance Equivalent Model section, Equation 1 shows the finger capacitance.

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

Where:

ε_0 = Free space permittivity

ε_r = Dielectric constant of overlay

A = Area of finger and sensor pad overlap

D = Overlay thickness

The geometry of a CapSense system is more complex than a parallel plate capacitor. The conductors in the sensor include the finger and PCB copper. However, like a parallel plate capacitor, C_F is directly proportional to ε_r. High dielectric constants lead to high sensitivity. Because air has the lowest dielectric constant, any air gaps between the sensor pad and overlay must be eliminated.

Dielectric constants of some common overlay materials are listed in Table 3-1. Materials with dielectrics between 2.0 and 8.0 are well suited to capacitive sensing applications.

Table 3-1. Dielectric Constants of Common Materials

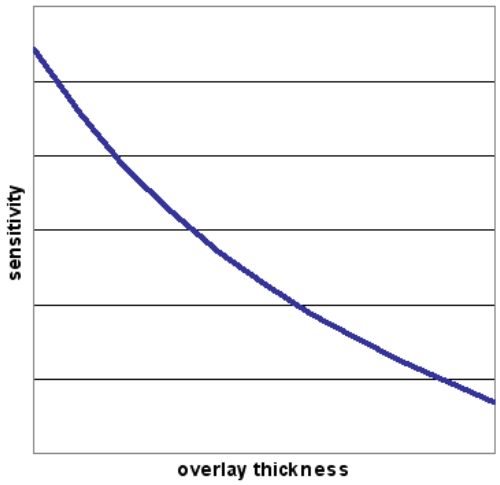
Material	ε_r
Air	1.0
Formica®	4.6–4.9
Glass (Standard)	7.6–8.0
Glass (Ceramic)	6.0
PET Film (Mylar®)	3.2
Polycarbonate (Lexan®)	2.9–3.0
Acrylic (Plexiglass®)	2.8
ABS	2.4–4.1
Wood Table and Desktop	1.2–2.5
Gypsum (Drywall)	2.5–6.0

Conductive material cannot be used as an overlay because it interferes with the electric field pattern. For this reason, do not use paints that contain metal particles in the overlay.

3.1.2 Overlay Thickness

Sensitivity is inversely proportional to overlay thickness, as illustrated in Figure 3-1.

Figure 3-1. Sensitivity Versus Overlay Thickness



Both signal and noise are affected by the overlay properties. Table 3-2 lists the recommended maximum overlay thicknesses for PSoC CapSense applications with an acrylic overlay material.

Table 3-2. Maximum Overlay Thickness With an Acrylic Overlay Material

Design Element	Max. Overlay Thickness (mm)
Button	5
Slider	5
Touchpad	0.5

3.1.3 Overlay Adhesives

Overlay materials must have good mechanical contact with the sensing PCB. This is achieved using a nonconductive adhesive film. This film increases the sensitivity of the system by eliminating any air gaps between overlay and the sensor pads. 3M™ makes a high performance acrylic adhesive called 200MP that is widely used in CapSense applications in the form of adhesive transfer tape (product numbers 467MP and 468MP).

3.2 ESD Protection

Robust ESD tolerance is a natural byproduct of thoughtful system design. By considering how contact discharge will occur in your end-product, particularly in your user interface, it is possible to withstand an 18 kV discharge event without incurring any damage to the CapSense controller.

Table 3-3. Overlay Material Dielectric Strength

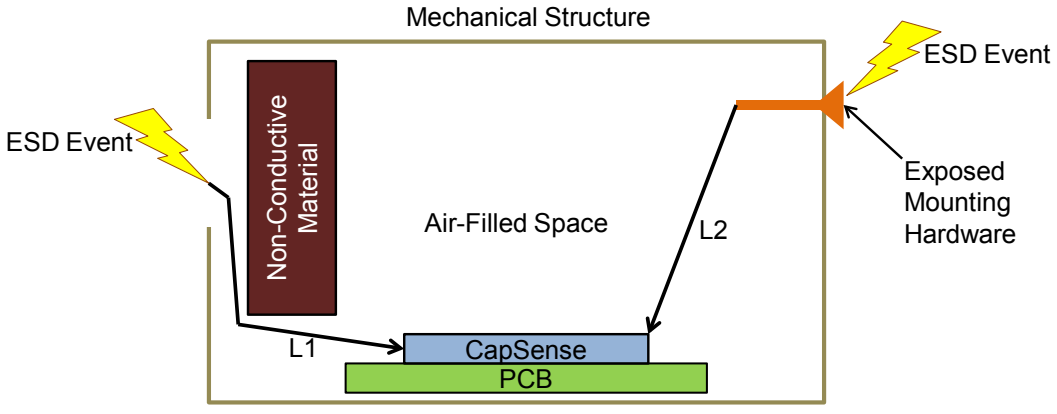
Material	Breakdown Voltage (V/mm)	Min. Overlay Thickness at 12 kV (mm)
Air	1200–2800	10
Wood – dry	3900	3
Glass – common	7900	1.5
Glass – Borosilicate (Pyrex®)	13,000	0.9
PMMA Plastic (Plexiglass)	13,000	0.9
ABS	16,000	0.8
Polycarbonate (Lexan)	16,000	0.8
Formica	18,000	0.7
FR-4	28,000	0.4
PET Film (Mylar)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

CapSense controller pins can withstand a direct 2-kV event. In most cases, the overlay material provides sufficient ESD protection for the controller pins. Table 3-3 lists the thickness of various overlay materials required to protect the CapSense sensors from a 12-kV discharge as specified in IEC 61000-4-2. If the overlay material does not provide sufficient protection, ESD countermeasures should be applied in the following order: Prevent, Redirect, Clamp.

3.2.1 Preventing ESD Discharge

Preventing the ESD discharge from reaching the CapSense controller is the best countermeasure you can take. Make certain that all paths on the touch surface have a breakdown voltage greater than any voltage to which the surface may be exposed. Also, design your system to maintain an appropriate distance between the CapSense controller and possible sources of ESD. In the example illustrated in Figure 3-2, if L1 and L2 are greater than 10 mm, the system will withstand 12 kV.

Figure 3-2. ESD Path

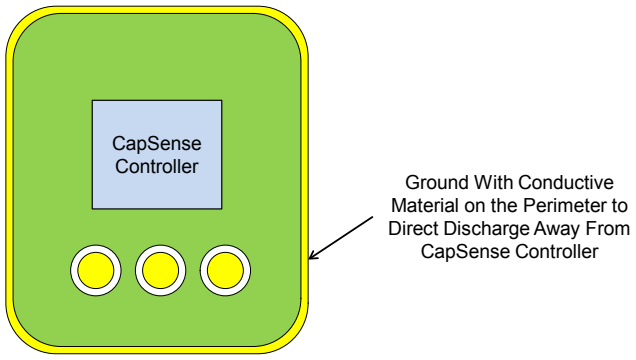


If it is not possible to maintain adequate distance, place a protective layer of a high breakdown voltage material between the ESD source and CapSense controller. One layer of 5 mil-thick Kapton tape will withstand 18 kV. See Table 3-3 for other material dielectric strengths.

3.2.2 Redirect

If your product is densely packed, it may not be possible to prevent the discharge event. In this case, you can protect the CapSense controller by controlling where the discharge occurs. This can be achieved through a combination of PCB layout, mechanical layout of the system, and conductive tape or other shielding material. A standard practice is to place a guard ring on the perimeter of the circuit board. The guard ring should connect to chassis ground.

Figure 3-3. Guard Ring

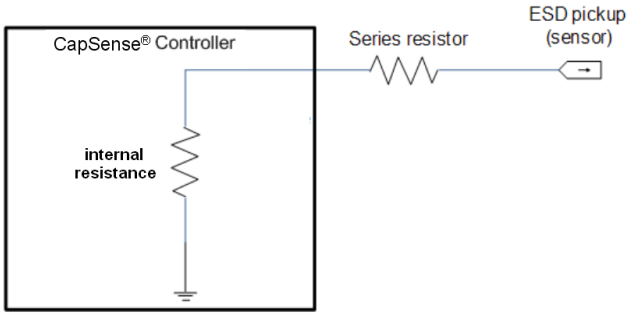


As recommended in PCB Layout Guidelines, providing a hatched ground plane around the button or slider sensor can also redirect the ESD event away from the sensor and CapSense controller.

3.2.3 Clamp

Because CapSense sensors are placed in close proximity to the touch surface, it may not be practical to redirect the discharge path. Including series resistors or special purpose ESD protection devices may be appropriate. Adding a series resistor on the vulnerable traces is a cost-effective protection method. This technique works by splitting the dissipation between the resistor and the controller. The recommended series resistance added to the CapSense inputs is 560 ohms. More details are available in the Series Resistor section.

Figure 3-4. ESD Protection Using a Series Resistor



A more effective method is to provide special purpose ESD protection devices on the vulnerable traces. ESD protection devices for CapSense need to be low capacitance. Table 3-4 lists devices recommended for use with CapSense controllers.

Table 3-4. ESD Protection Devices

ESD Protection Device		Input Capacitance	Leakage Current	Contact Discharge Maximum Limit	Air Discharge Maximum 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

3.3 Electromagnetic Compatibility (EMC) Considerations

EMC is related to the generation, transmission, and reception of electromagnetic energy that can upset the working of an electronic system. The source (emitter) produces the emission and a transfer or coupling path transfers the emission energy to a receptor, where it is processed, resulting in either desired or undesired behavior. 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.

3.3.1 Radiated Interference and Emissions

Radiated electrical energy can influence system measurements and potentially influence the operation of the CapSense processor core. The interference enters the CapSense chip at the PCB level through the sensor traces and other digital and analog inputs. While CapSense offers an intuitive and robust interface that increases product reliability by eliminating mechanical parts, it can also contribute to electromagnetic compatibility (EMC) issues in the form of radiated emissions (RE).

Use the following techniques to minimize radiated interference and emissions.

3.3.1.1 General EMI/EMC Guidelines

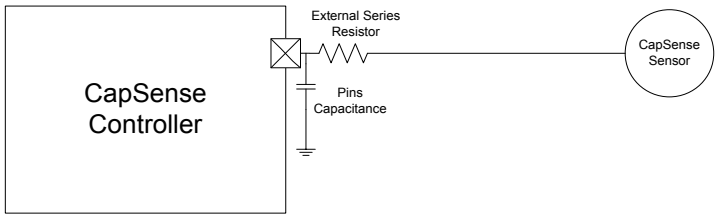
3.3.1.1.1 Ground Plane

In general, a proper ground plane on the PCB reduces both RF emissions and interference. However, solid grounds near CapSense sensors or traces connecting these sensors to the PSoC pins increase the parasitic capacitance of the sensors. It is thus recommended that you use hatched ground planes surrounding the sensor and on the bottom layer of the PCB,, below the sensors, as explained in the Ground Plane section. A solid ground may be used below the device and other circuitry on the PCB, which is far from CapSense sensors and traces. A solid ground flood is not recommended within 10 mm of CapSense sensors or traces. Multiple-layer boards should be the preferred choice. If you are using a board with four layers or more, you can provide a complete layer for ground that will further help to reduce emissions as it reduces the ground bounce significantly.

3.3.1.1.2 Series Resistor

Every CapSense controller pin has some parasitic capacitance (C_P) associated with it. Adding an external resistor forms a low-pass RC filter that can dampen the RF noise amplitude coupled with the pin. This resistance also forms a low-pass filter with the parasitic capacitance of the trace connected to the pin (for example, sensor trace and sensor pad as shown in Figure 3-5) that can significantly reduce RF emissions. Thus, series resistors help in eliminating higher-order harmonics and attenuating the RF interference and emission.

Figure 3-5. RC Filter



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

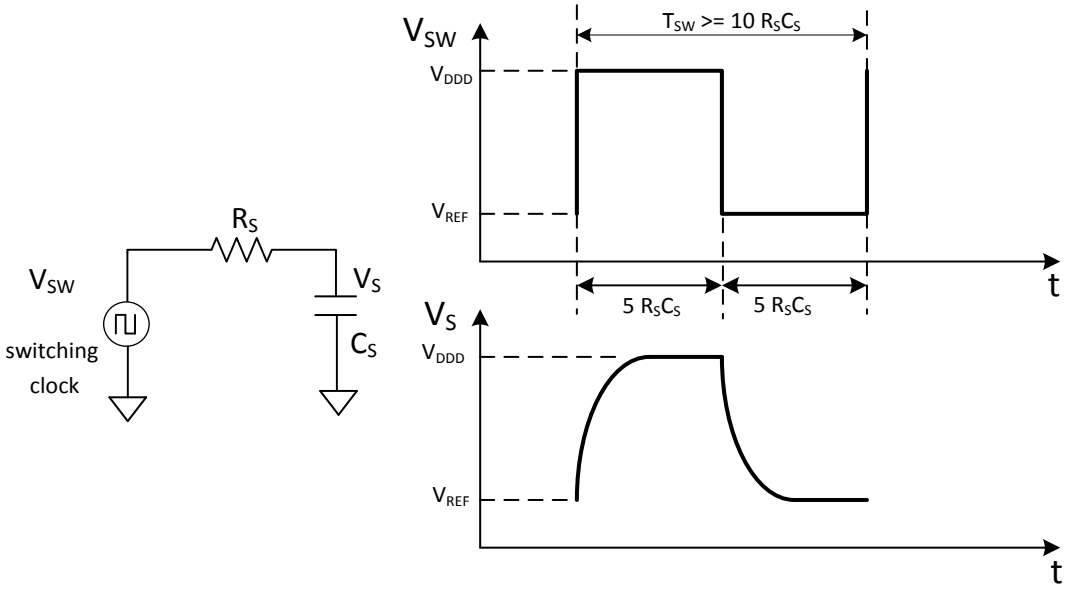
3.3.1.1.2.1 CapSense Input Lines

The sensor must be fully charged and discharged during each switching cycle for proper operation of CapSense. The charge and discharge paths of the sensor capacitor include series resistances that slow down the charging / discharging process. Figure 3-6 shows an equivalent circuit and resulting waveforms. Adding a resistance changes the time constant of the switched-capacitor circuit that converts C_P into an equivalent resistor. If the series resistance value is set high, the slower time constant of the switching circuit suppresses the emission but limits the amount of charge that it can transfer. Thus the sensors may not get charged and discharged completely. This lowers the signal

level, which in turn lowers the SNR. Smaller values are better, but are less effective at blocking RF emissions and interference.

The recommended series resistance for CapSense input lines on general copper PCBs is 560 Ω. An ITO panel already provides a high resistance; one may not have too much flexibility in the value selection (range 100 Ω – 1 kΩ).Series resistors are generally used in the range of 560 Ω – 4.7 kΩ for EMC purpose. The actual maximum value of the series resistor that can be used varies from device to device. This depends on multiple factors such as the resistance of the GPIO used as sensor, the switching frequency used to scan sensors, and the SNR required.

Figure 3-6: 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. For a given switching frequency, you must select the series-resistor value such that the sensor capacitor is charged and discharged fully. On the other hand, for a given series resistor, you must choose the switching frequency value such that the sensor capacitor is charged and discharged fully. Lowering the switching frequency lowers the SNR if you are unable to modify other CapSense parameters. Therefore, it is a tradeoff between the series resistor value and the switching frequency to achieve the desired performance.

The rule of thumb is to allow a period of $5R_SC_S$ for charging and discharging cycles. Equations for the minimum time period and maximum frequency are:

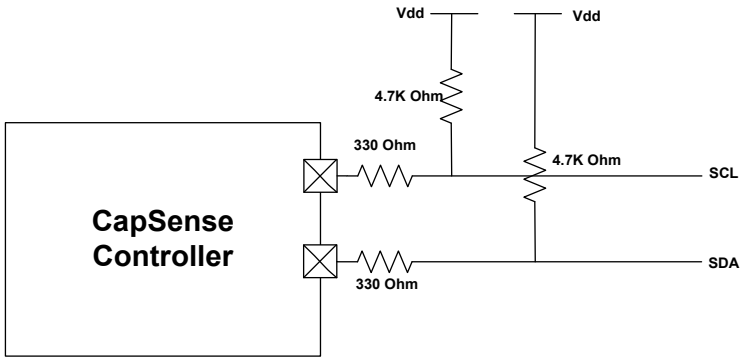
$$T_{SW}(\text{minimum}) = 10R_SC_S$$

$$F_{SW}(\text{maximum}) = \frac{1}{10R_SC_S}$$

3.3.1.1.2.2 Digital Communication Lines

Communication lines, such as I^2C and SPI, also benefit from series resistance and 330 Ω is recommended for communication lines. Communication lines have long traces that act as antennae similar to CapSense traces. The recommended pull-up resistor values for communication lines is 4.7 kΩ . So, if more than 330 Ω is placed in series on these lines, the voltage levels(V_{IL} and V_{IH}) fall out of the specifications with the worst-case combination of supply voltages between systems and the input impedance of the receiver. 330 Ω will not affect the I^2C operation because the V_{IL} level still remains within the I^2C specification limit of 0.3 V_{DD} when the PSoC device outputs a LOW.

Figure 3-7: Series Resistors on Communication Lines



3.3.1.1.3 Trace Length

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

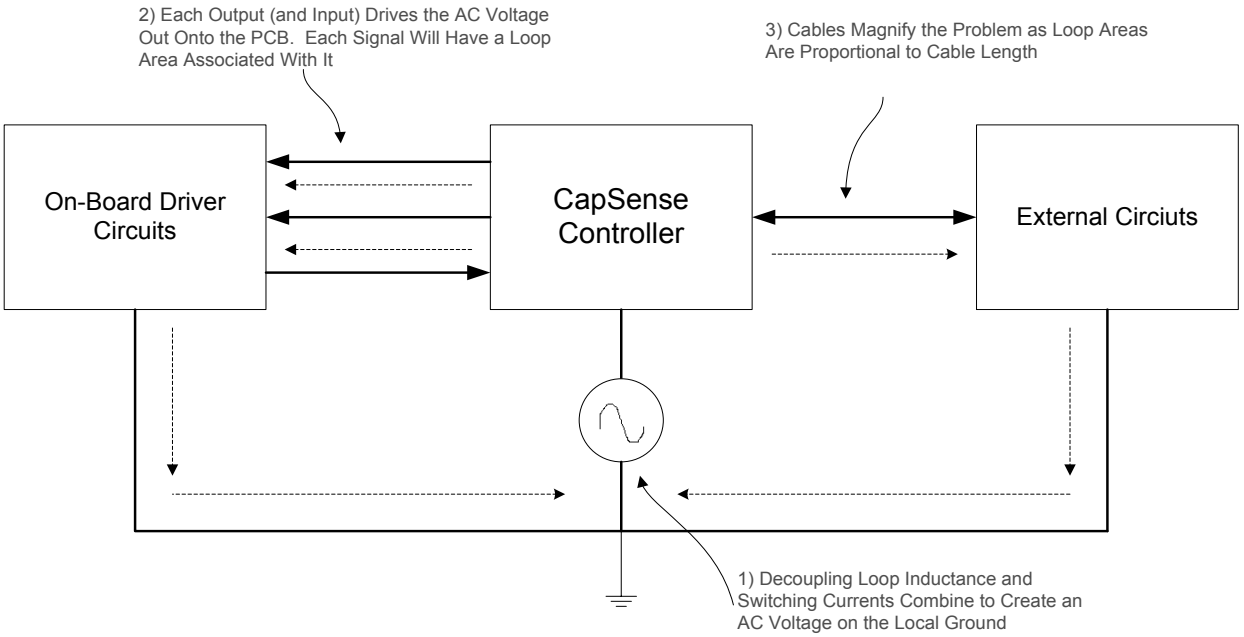
3.3.1.1.4 Current Loop Area

Another important layout consideration is to minimize the return path for current. A ground plane can lower the overall ground impedance, thus reducing the high-frequency ground bounce. You should ensure good GND return paths for each sensor line. This is important as the current flows in loops. Unless there is a proper return path for high-frequency signals, the return current will flow through a longer return path forming a larger loop; this can cause signal issues due to the mutual inductance caused. Thus, it leads to increased emissions and interference.

When a device package contains high-frequency current loops, energy can also be coupled out of the device through a magnetic field. It is possible for the magnetic flux to form a current loop in the device to link to the circuit loops outside the device. This mutual inductance can produce an unwanted voltage in the external loop. Likewise, an external magnetic flux can induce an unwanted voltage across an interior circuit loop. Magnetic field coupling can be minimized by keeping the power and signal loop areas as small as possible. Stitch all the grounds with as many vias as possible. This will reduce the overall ground impedance. High-frequency traces, such as those used for clock and oscillator circuits, should be contained by two ground lines. This will ensure that there is no coupling, which results in crosstalk. Use separate ground plane and power planes wherever possible.

Figure 3-8 shows an example of an improper grounding scheme. The layout greatly improves by reducing the loop area.

Figure 3-8. Improper Ground Scheme and Ground Loop



In Figure 3-9, two sensors are surrounded by a ground plane that is connected to CapSense controller ground, while a third sensor is surrounded by ground. The third sensor is connected to the other ground plane through the long traces of other circuitry, which creates a large current loop. With this layout, the third sensor may be more susceptible to radiated noise and have increased emissions. These two sections of ground are the same location on the schematic, so they can potentially be one connected area with a better layout.

Figure 3-9. Improper Current Loop Layout

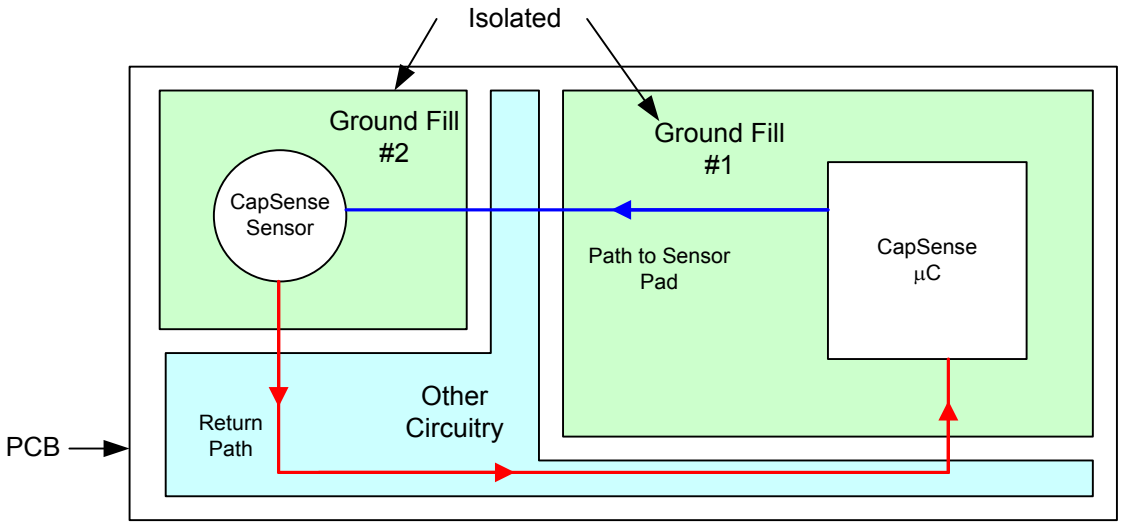
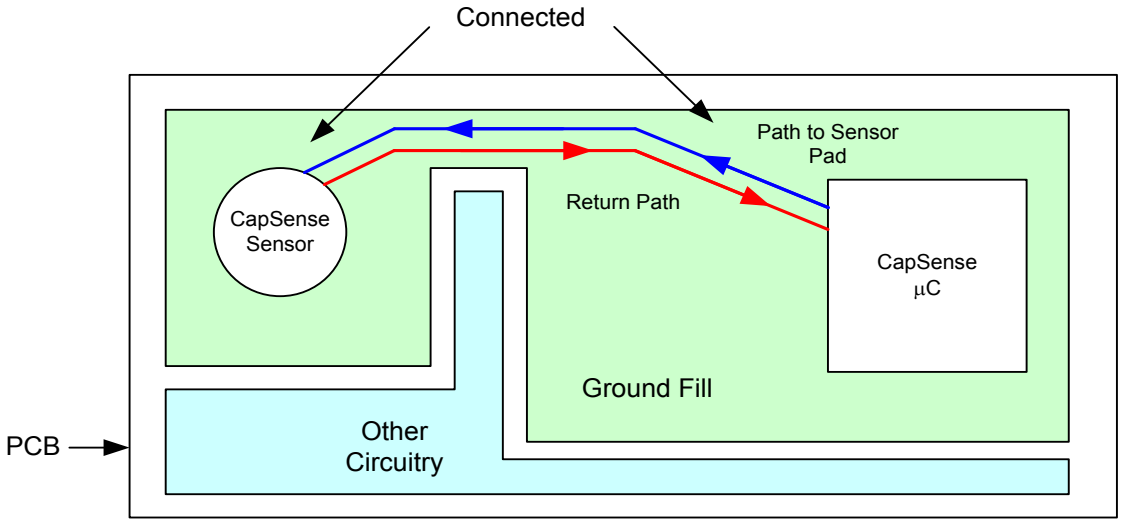


Figure 3-10 illustrates the proper layout for the previous example. The loop area is reduced by connecting the two grounded areas.

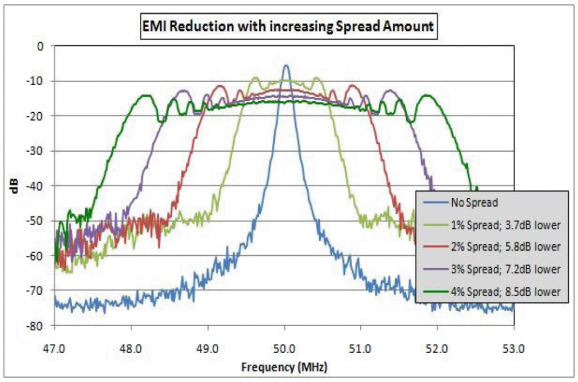
Figure 3-10. Proper Current Loop Layout



3.3.1.1.5 Frequency Hopping

Frequency hopping is a method of spreading the input or operating frequency over a narrow band of frequencies. This method will help to reduce the peaks and spread out the emissions as well as the interference over a range of frequencies as shown in Figure 3-11. The following are the methods of frequency hopping in PSoC:

Figure 3-11.Frequency Hopping



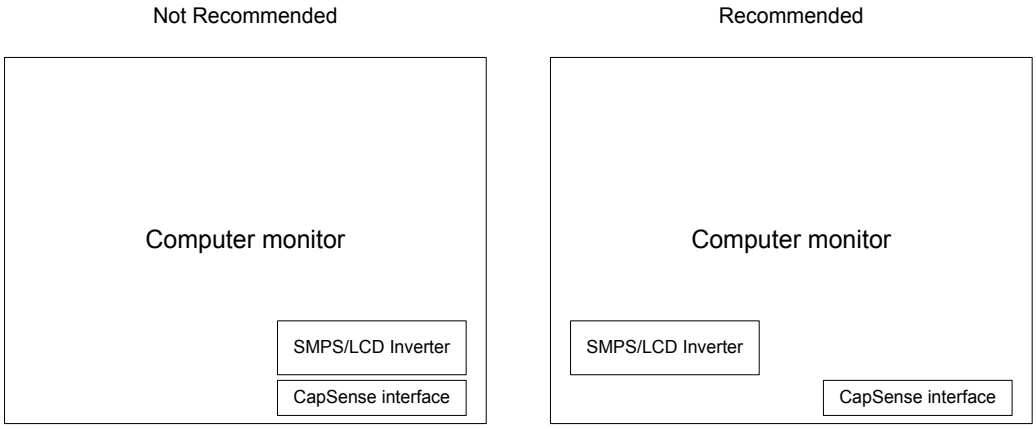
- **IMO dithering across sensor scans:** IMO dithering or trimming can be done across different sensors. For example, the IMO frequency is swept over a range from 24 MHz to 22 MHz when the base IMO frequency is 24 MHz. A sensor is scanned at one frequency always. Different sensors are scanned at different frequencies. This reduces the peaks by spreading the emissions.
- **IMO dithering within each scan:** IMO dithering can be done within each scan as well. When a sensor is being scanned, the IMO frequency is swept over a range from 24 MHz to 22 MHz. Thus, this method avoids a sensor being scanned at one frequency. This reduces the peaks by spreading the emissions. This also improves the immunity to RF interference.
- **Spread Spectrum Clock:** PSoC can also work using an external clock. Using a spread-spectrum clock will help in spreading out the emissions over a wider range of frequencies, similar to IMO dithering. PSoC1 allows only port P1[4] to be used to supply the external clock. In this case, pin P1[4] drive mode must be set to HI-Z digital. This increases the immunity to RF interference and spreads the emissions.
- **Pseudo random sequencer (PRS):** A PRS is used instead of a fixed clock-source to attenuate the emitted noise on CapSense pins by reducing the amount of EMI created by a fixed frequency-source and to increase the EMI immunity from other sources and their harmonics. This increases the immunity to RF interference and spreads the emissions.

3.3.1.2 Radiated Immunity

3.3.1.2.1 RF Source Location

When systems, such as computer monitors or digital photo frames, are designed with CapSense devices, make sure you prevent noise from LCD inverters and switched-mode power supplies (SMPS) from upsetting the CapSense system. A simple technique to minimize this kind of interaction is to partition the system with noise sources from CapSense inputs, as demonstrated in Figure 3-12. Due to the practical limitations of product size, the noise source and the CapSense circuitry may only be separated by a few inches. This small separation can provide the extra margin required for good sensor performance compared to the case with close proximity between noise source and CapSense.

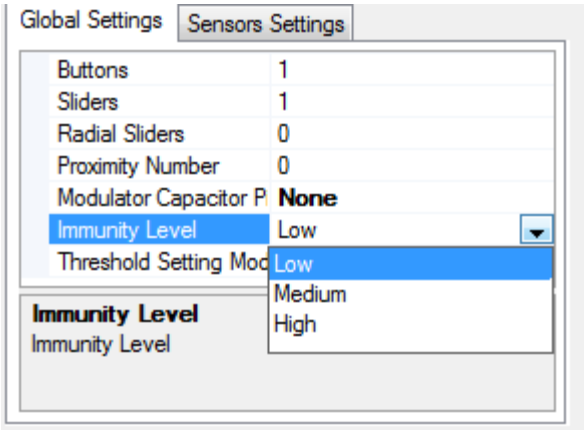
Figure 3-12. Separating Noise Sources



3.3.1.2.2 EMC Feature

CapSense user modules/components with the EMC feature implements IMO dithering to scan each sensor. Each sensor is scanned at two or three different frequencies depending on the immunity level chosen for each sample for raw count. Use this option to improve the immunity against RF interference.

Figure 3-13: Immunity Level Selection

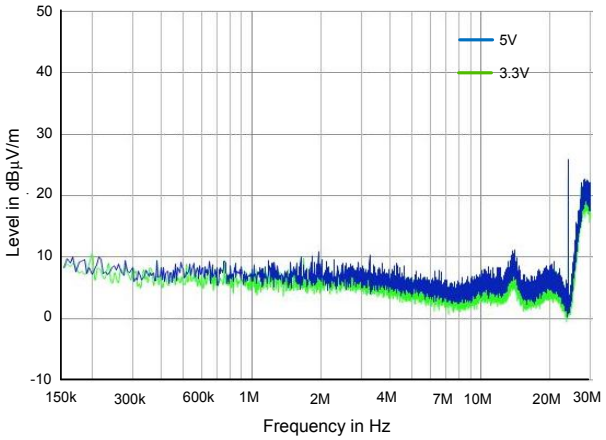


3.3.1.3 Radiated Emissions

3.3.1.3.1 Operating Voltage

For devices in which sensors switch between an operating voltage and a reference voltage such as CY8C21x34, reducing the operating voltage will help to reduce emissions to a great extent as the amplitude of the switching signal at any pin is dependent on the operating voltage of the device and the emissions is directly proportional to voltage levels at which the switching happens. Figure 3-14 shows the impact of operating voltage on radiated emissions.

Figure 3-14 Impact of Operating Voltage on Emissions



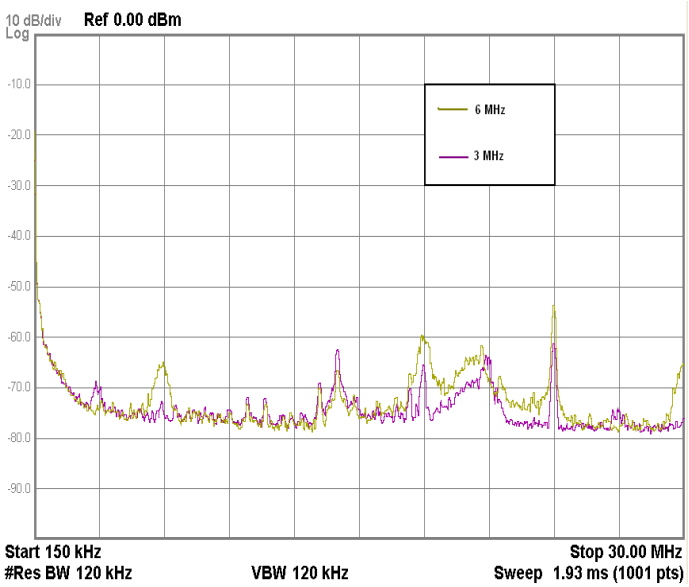
3.3.1.3.2 System Oscillator Frequency (IMO)

Lowering the system clock will lower radiated emissions by a good amount. However, lowering the system clock will impact the performance of your system because with a low IMO, it can take more time to scan the sensors and perform the processing. Therefore, lower the system frequency depending on your application.

3.3.1.3.3 Sensor Switching Frequency

The CapSense sensing methods use a switched-capacitor front end to interact with sensors. Selecting a low frequency for the switched-capacitor clock helps you to reduce the radiated emissions from the CapSense sensors. Figure 3-15 shows the impact of the sensor-switching frequency.

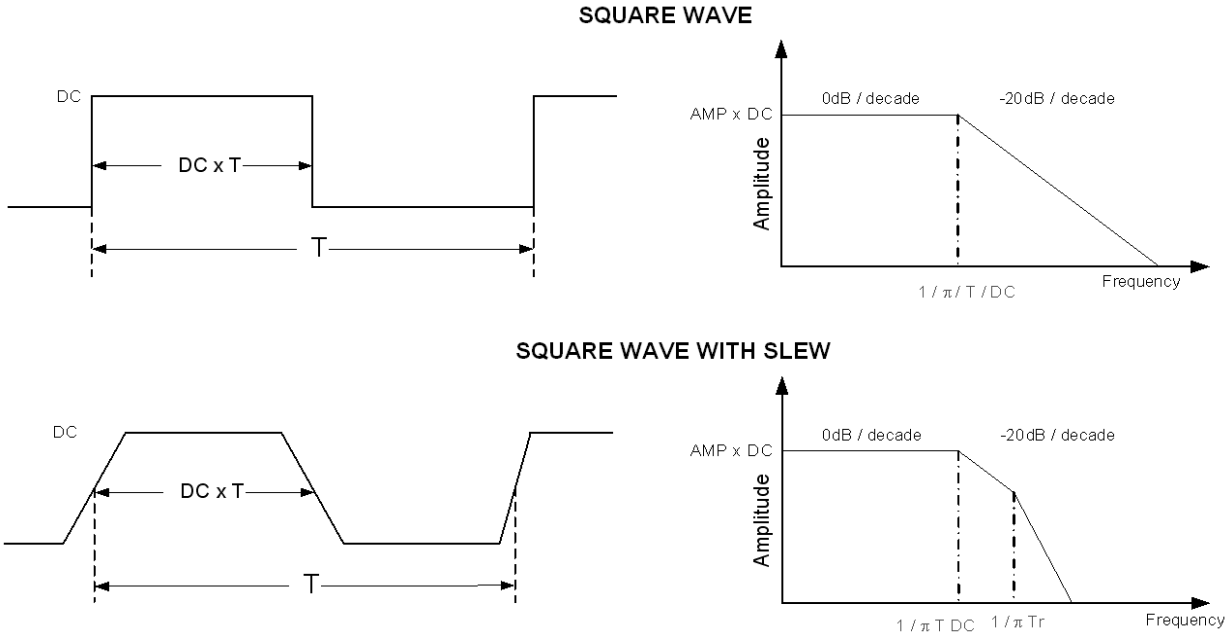
Figure 3-15 Impact of Switching Frequency



3.3.1.3.4 Slew Rate Control

Figure 3-16 shows the impact of rise/fall time of a square wave on radiated emissions. Note that slowing the transitions introduces the cutoff point and damps the radiated-energy level. Internal clock signals of the CapSense controller are slew-controlled to reduce the radiated emission.

Figure 3-16. Impact of Slew Rate on Emissions



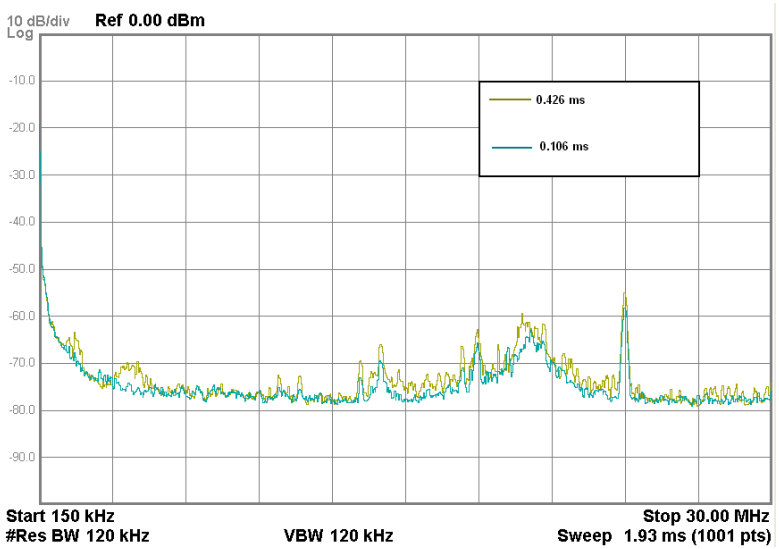
3.3.1.3.5 Sensor Scan time

The scan time of sensors impact radiated emissions. Figure 3-17 shows the impact of the sensor-scan time on radiated emissions. An increase in the sensor-scan time results in increased emissions. Table 3-5 shows the parameter settings and associated sensor scan times.

Table 3-5: Sensor Scan time

Parameter	Total Scan Time for Five Buttons	
	0.426 ms	0.106 ms
Scan resolution	10 bits	8 bits
Individual sensor scan time	0.085 ms	0.21 ms

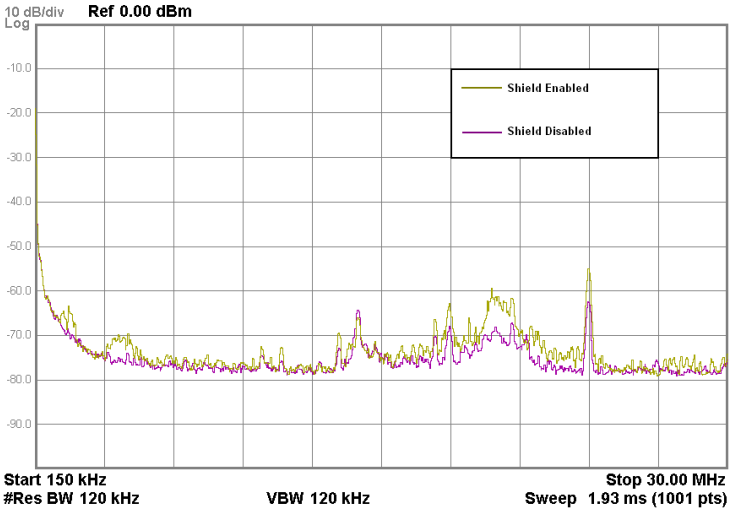
Figure 3-17– Effect of Scan Time



3.3.1.3.6 Shield Signal

The shield signal is driven on the hatch fill to reduce the parasitic capacitance of sensors for liquid tolerance and for proximity sensing. See the Shield Electrode and Guard Sensor section for more details. The shield signal is a replica of the sensor signal. The shield signal further increases radiated emissions as it is a high-frequency switching signal and is driven on a spread-out hatch fill. Figure 3-18 shows the emissions with and without the driven-shied signal:

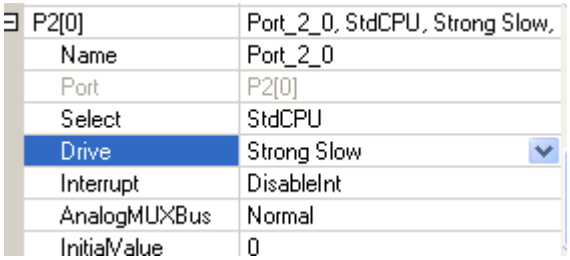
Figure 3-18 Impact of Shield on Emissions



Emissions can be reduced by the following techniques:

1. Reduce the size of the shield hatch fill to have a maximum of 10 mm from the sensors. Refer to the Shield Electrode and Guard Sensor section for more details.
2. Drive the shield signal only when required. The shield must be driven only when sensors are being scanned and only when those sensors are being scanned which need shield protection.
3. Limit the placement of the shield to selected sensors only. Do not spread the shield around sensors which do not need shield protection.
4. Slow the edges of shield waveform by one of the following means:
    a. Add a capacitor filter between the shield electrode port pin and ground
    b. In most of the PSoC 1 CapSense devices, slew rate of shield signal can also be controlled by changing the drive mode of the shield pin from Strong to Strong Slow as shown in the figure below:

Figure 3-19: Drive Mode Selection for Shield



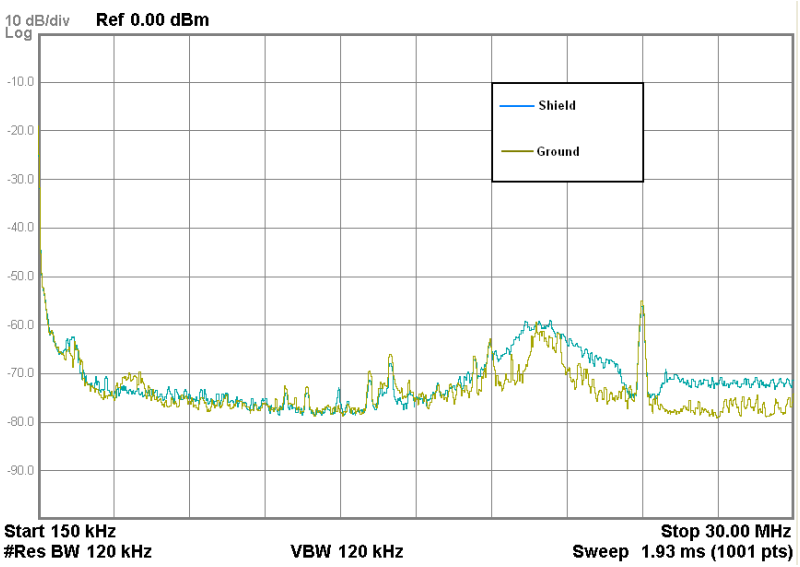
By adding a passive low-pass filter (LPF) to the shield signal by putting a series resistor to the shield pin. An inherent LPF is formed by the resistance of the electrode material and the parasitic capacitance of the electrode. Therefore, adding a series resistor increases the RC of the filter and helps to improve emissions to a great extent because the RC filter formed will eliminate higher-order harmonics of these frequencies.

Note: Filtering the shield too much can cause a phase difference between the driven-shield signal and the sensor-switching signal. You must ensure that the shield electrode gets charged and discharged completely when you add filters. Refer to the CapSense Input Lines section for charge and discharge waveforms, and to see how to choose the right value for series resistors.

3.3.1.3.7 Inactive Sensor Termination

Connect inactive sensors to ground to reduce emissions instead of the shield unless it is a stringent requirement to connect them to the shield. Figure 3-20 shows the impact of different inactive-sensor terminations on radiated emissions.

Figure 3-20: Effect of Inactive Sensor Termination



For CapSense applications, it is very important to have a clean power supply for CapSense devices to reduce problems related to radiated interference and emissions. Guidelines to filter the noise at the power supply of CapSense devices are given in the following section. It is recommended that you incorporate these guidelines to handle any EMC/EMI issues.

3.3.2 Conducted Immunity and Emissions

The noise current generated by high-frequency switching circuits entering the system through the power and communication lines is called conducted noise.

3.3.2.1 Board-Level Solutions

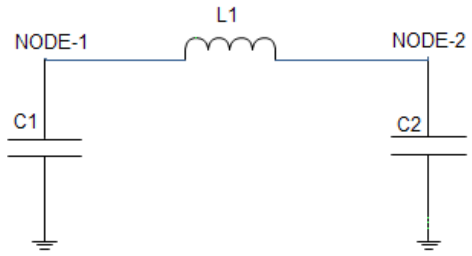
Proper use of decoupling capacitors as recommended by the datasheet can limit the problem with conducted emissions. For detailed information on general decoupling capacitors, refer to the section 3.8.12. For unregulated power supplies, use a large electrolytic capacitor (typically 10 µF –100 µF) no more than one inch away from the

chip. This capacitor acts as a reservoir of charge to supply the instantaneous charge requirements of the circuits locally so that the charge need not come through the inductance of the power trace.

For further protection, a passive filter can be used. Passive filter effectively limits not just the conducted noise emitted but also the noise entering the system. Thus, it improves the conducted noise immunity of the system.

A pi-filter is a simple bidirectional low-pass filter. The two main types of pi-filters are the series inductor and the series resistor. The series inductor pi-filter has two shunt capacitors and one series inductor configured similar to the Greek letter π, as shown in Figure 3-21. The noise is filtered by all three elements (L1, C1, and C2) in both directions. The bidirectional nature of the filter is important. Not only does it prevent the supply noise from affecting sensitive parts, it can also prevent the switching noise of the part from coupling back onto the power planes.

Figure 3-21: Series Inductor Pi-Filter



The values of the components are selected based on the frequency that needs to be attenuated.

3.3.2.2 Power Supply Solutions

The following guidelines help you to prevent conducted noise from entering your CapSense design:

- Provide GND and V_{DD} planes that reduce current loops.
- If the CapSense controller PCB is connected to the power supply by 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.
    - Localizes the noise in the system.
    - Keeps external high frequency noise away from the IC.
    - Keeps internally generated noise from propagating to the rest of the system

For more information on design considerations for EMC, refer to the following documents:

- Top 10 EMC Design Considerations
- PSoC®1 /PSoC3 / PSoC 5 EMI Design Considerations
- PSoC®3 /PSoC4 and PSoC 5LP EMC Best Practices and Recommendations

3.4 Software Filtering

Software filters are one of the techniques of dealing with high levels of system noise. Table 3-6 lists the types of filters that are useful for CapSense.

Table 3-6. CapSense Filter Types

Type	Description	Application
Average	Finite impulse response filter (no feedback) with equally weighted coefficients	Periodic noise from power supplies
IIR	Infinite impulse response filter (feedback) with a step response similar to an RC filter	High frequency white noise (1/f noise)
Median	Nonlinear filter that computes median input value from a buffer of size N	Noise spikes from motors and switching power supplies
Jitter	Nonlinear filter that limits current input based on	Noise from thick overlay (SNR < 5:1), especially useful for

Type	Description	Application
	previous input	slider centroid data
Event-Based	Nonlinear filter that causes a predefined response to a pattern observed in the sensor data	Commonly used to block generation or posting of nonexistent events
Rule-Based	Nonlinear filter that causes a predefined response to a pattern observed in the sensor data	Commonly used during normal operation of the touch surface to respond to special scenarios such as accidental multi-button selection

3.4.1 Average Filter

An average filter is a finite impulse response (FIR) filter with equal-weighted coefficients. The average filters work well with periodic noise that is attenuated by spacing the samples out over one noise cycle. Sample spacing is not critical. For example, power line noise can be anywhere from 50 Hz to 60 Hz. Without adjusting the sampling rate, the average filter works well for 50-Hz and 60-Hz noise. Figure 3-22 shows a sample rate that is synchronized with a simple periodic waveform. There is no feedback path in this filter.

Figure 3-22. Synchronized Sample Rate

The general equation for an average filter is:

$$y[i] = \frac{1}{N}\left(x[i] + x[i-1] + \cdots + x[i-N+1]\right) \quad \text{Equation 16}$$

Figure 3-23 and Figure 3-24 illustrate the results of using an average filter on real CapSense data using the 16-sample filter equation:

$$y[i] = \frac{1}{16}\left(x[i] + x[i-1] + \cdots + x[i-15]\right) \quad \text{Equation 17}$$

Figure 3-23. Average Filter Noise (16 Samples)

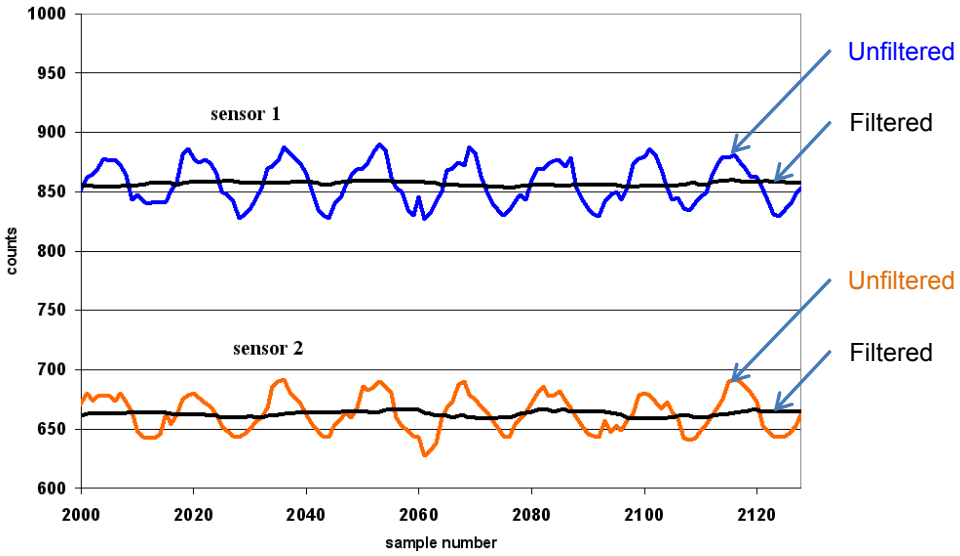
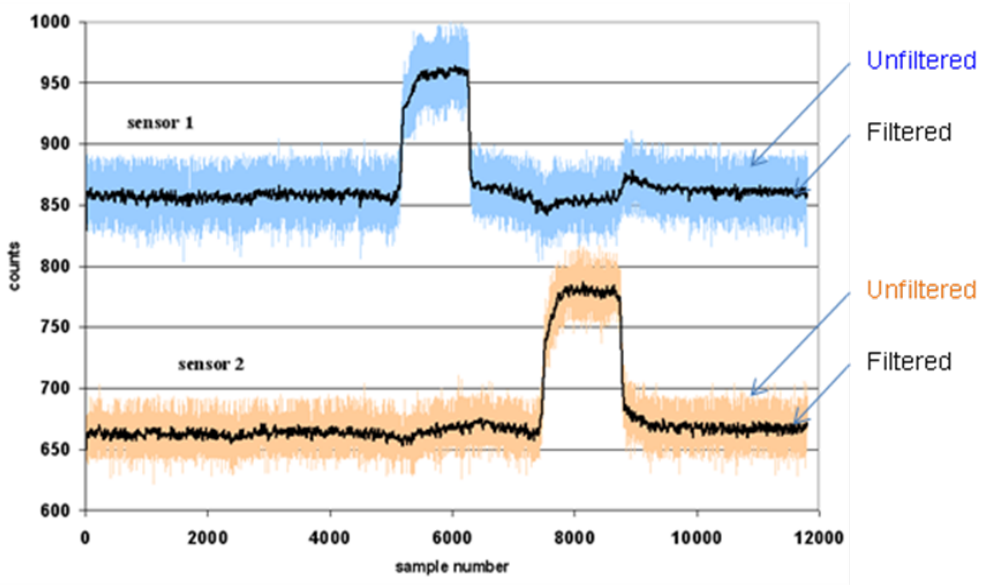


Figure 3-24. Average Filter Finger Touch (16 Samples)

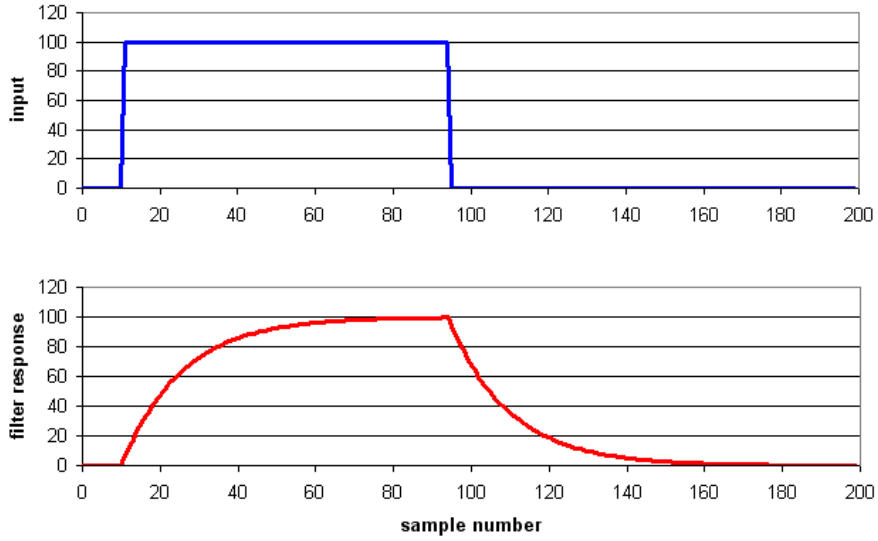


The previous examples are representative of power supply noise. The filter works well in this example because the period of the noise is close to the length of the filter (N = 16). For more information about how to implement an average filter, see the code example CSA Software Filters with EzI2Cs Slave on CY8C20XX6.

3.4.2 IIR Filter

Infinite impulse response filters (IIR) produce a step response similar to RC filters. IIR filters attenuate high-frequency noise components and pass lower frequency signals, such as finger touch–response waveforms.

Figure 3-25. IIR Filter Step Response



The general equation for a first-order IIR filter is:

$$y[i] = \frac{1}{k}\left(x[i] + \left((k-1) \times y[i-1]\right)\right)$$ Equation 18

Figure 3-26 and Figure 3-27 illustrate the results of a first-order IIR filter on real CapSense data using the filter equation with k = 16:

$$y[i] = \frac{1}{16}\left(x[i] + (15 \times y[i-1])\right)$$ Equation 19

Figure 3-26. IIR Filter Noise

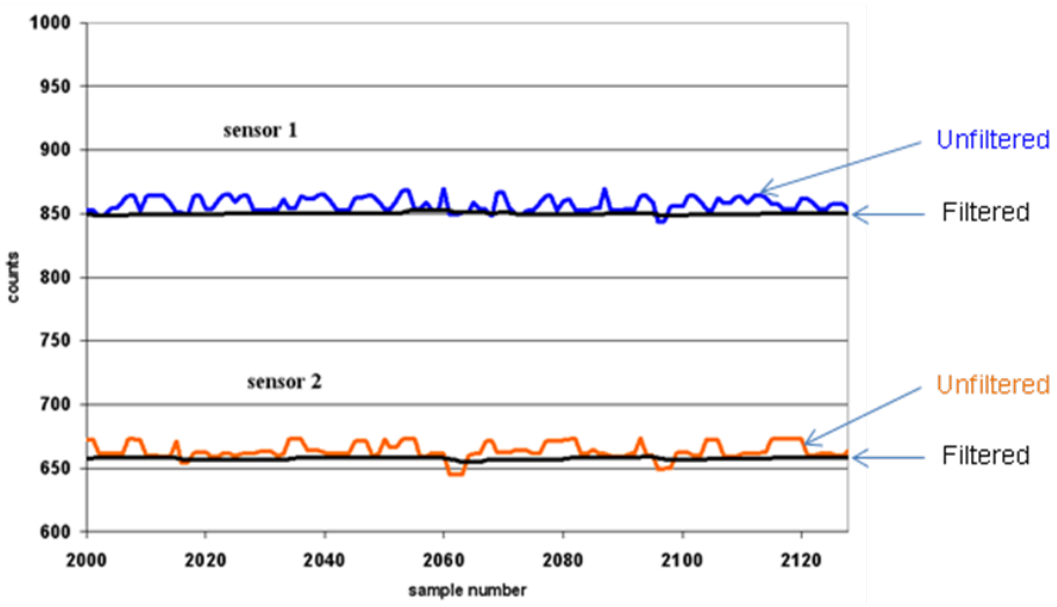
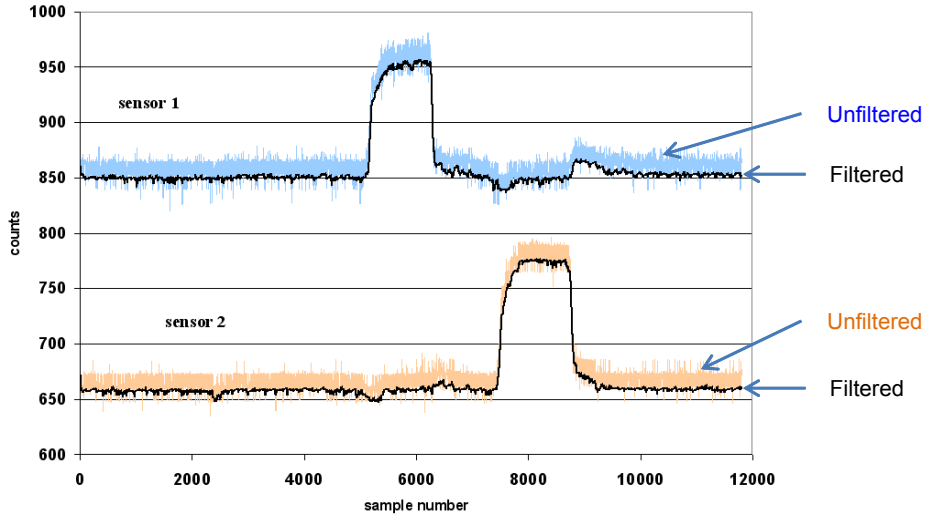


Figure 3-27. IIR Filter Finger Touch



For more information about how to implement an IIR filter, see the code example CSA Software Filters with EzI2Cs Slave on CY8C20XX6.

3.4.3 Median Filter

Median filters eliminate noise spikes most commonly associated with motors and switching power supplies. In a median filter, a buffer of size N stores the N most recent samples of the input. The median is then computed using a two-step process. First, the buffer values are sorted from smallest to largest; then, the middle value is selected from the ordered list. The buffer is scanned for the median with each update of the buffer. This is a nonlinear filter. The general equation for a median filter is:

$$y[i] = median(x[i],\ x[i-1], \ldots, x[i-N+1])$$ Equation 20

Figure 3-28 and Figure 3-29 show the results of a median filter on real CapSense data using the general filter equation with N = 16.

$$y[i] = median(x[i],\ x[i-1], \ldots, x[i-15]) \quad \text{Equation 21}$$

Figure 3-28. Median Filter Noise Spike

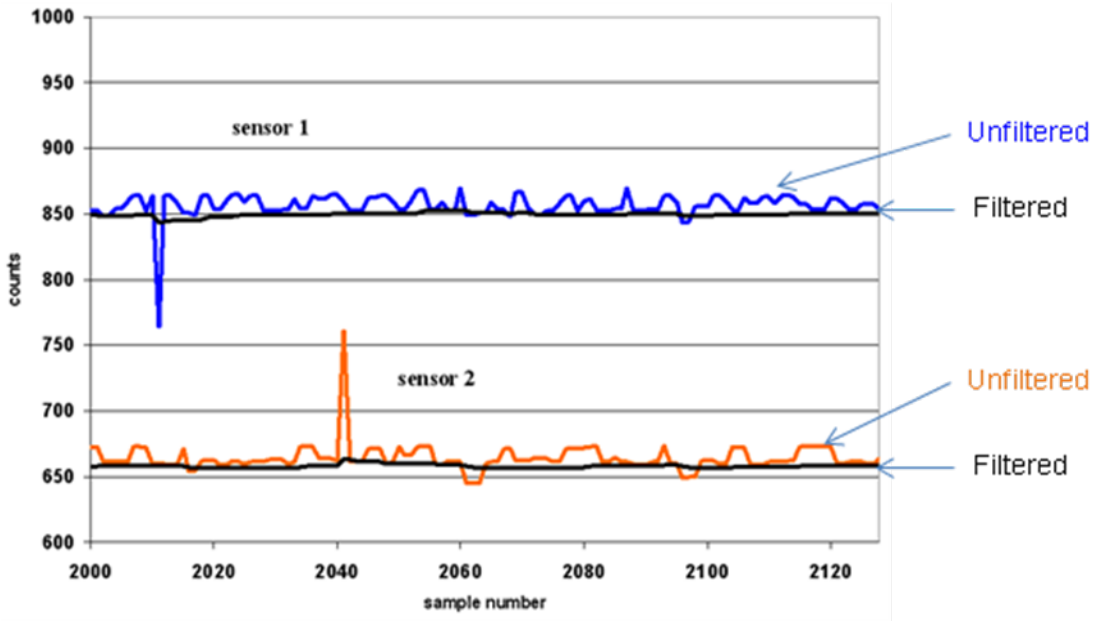
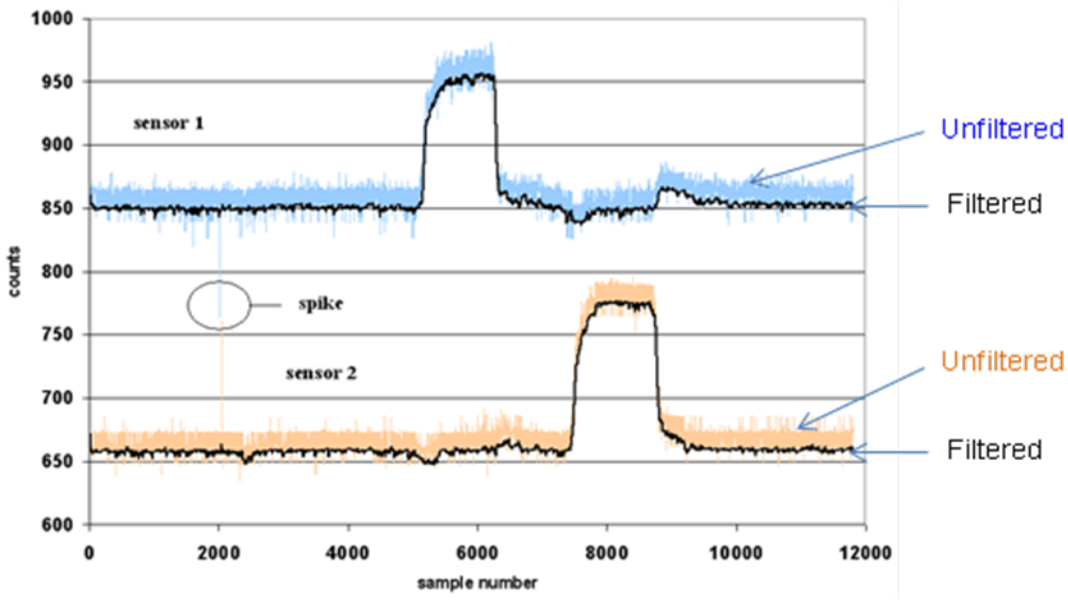


Figure 3-29. Median Filter (16-Sample) Finger Touch



For more information about how to implement a median filter, see the code example CSA Software Filters with EzI2Cs Slave on CY8C20XX6.

3.4.4 Jitter Filter

3.4.4.1 Jitter Filter for Noisy Slider Data

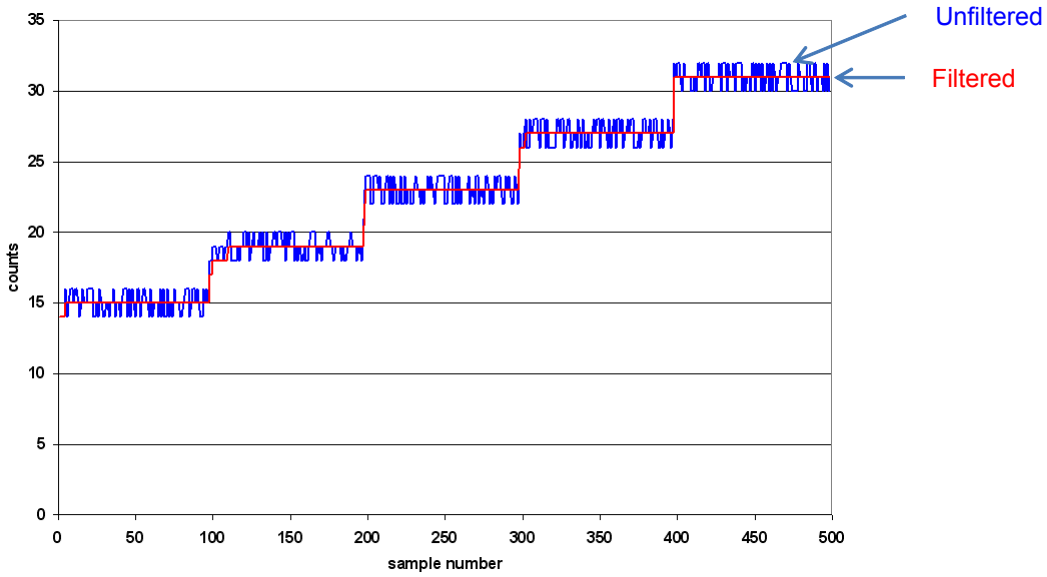
The centroid function is used to estimate finger position on a slider. When the signal level is low, usually because of thick overlay on the slider, the estimate of finger position will appear to shake and jitter even when the finger is held at a fixed position. This jitter noise can be removed using a jitter filter. To do this, the previous input is stored in a buffer. The current input is compared to the previous output. If the difference is greater than ±1, the output is changed by ±1 (matching sign), as shown in Equation 22. This is a nonlinear filter.

$$y[i] = x[i] - 1, \quad if\ x[i] > y[i-1] + 1$$
$$y[i] = x[i] + 1, \quad if\ x[i] < y[i-1] - 1$$
$$y[i] = y[i-1], \quad otherwise$$

Equation 22

Figure 3-30 shows the results of applying a jitter filter applied to noisy centroid data.

Figure 3-30. Jitter Filter Applied to Noisy Centroid Data



3.4.4.2 Jitter Filter for Raw Counts

Although the jitter filter is intended for use with noisy slider data, it is also used with noisy buttons. If the change in the current input exceeds a set threshold level, the output is changed to the previous input plus or minus the threshold amount. The output is not changed if the current input changes by less than the threshold amount. The general equation for a jitter filter applied to buttons is:

$$y[i] = x[i] - threshold, \quad if\ x[i] > y[i-1] + threshold$$
$$y[i] = x[i] + threshold, \quad if\ x[i] < y[i-1] - threshold$$
$$y[i] = y[i-1], \quad otherwise$$

Equation 23

Figure 3-31 and Figure 3-32 show the result of using a jitter filter on real button data with a large component of periodic noise.

Figure 3-31. Jitter Filter for Button Noise

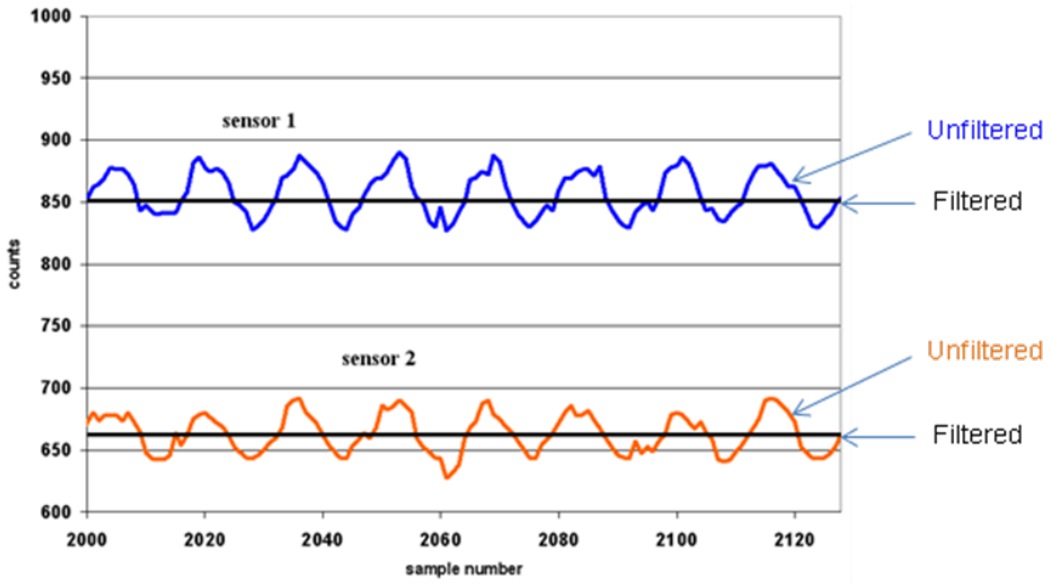
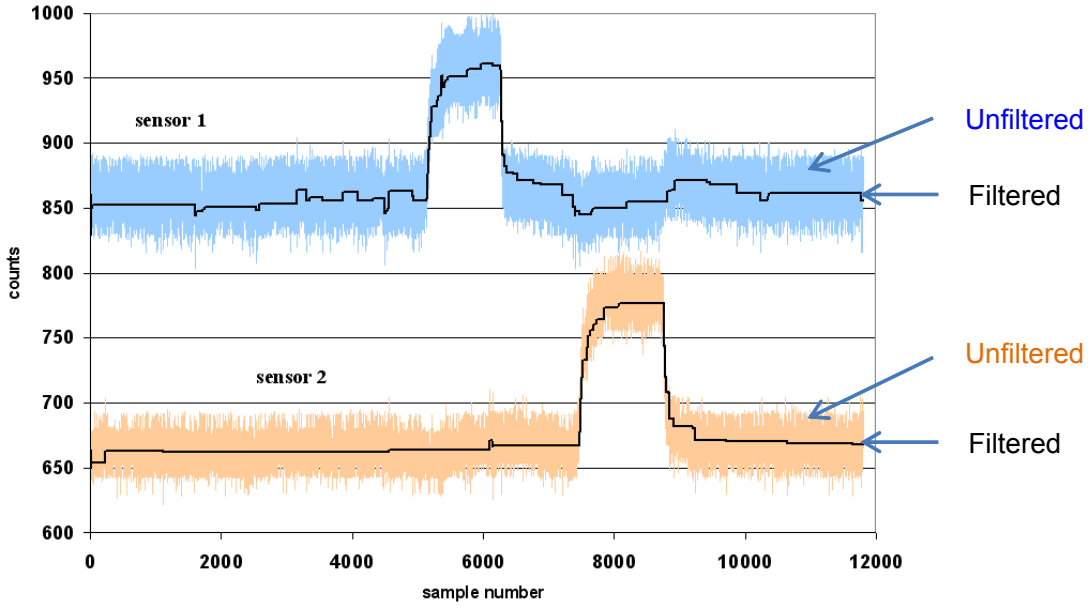


Figure 3-32. Jitter Filter for Button Finger Touch



For more information about how to implement a jitter filter, see the code example CSA Software Filters with EzI2Cs Slave on CY8C20XX6.

3.4.5 Event-Based Filters

Event-based filters involve a special filtering method, where a pattern observed in the sensor data causes a predefined response in the CapSense system. The pattern in the data is triggered by an event, such as a handheld product being placed into a pocket, or the power supply voltage (V_{DD}) dropping suddenly in a camera phone when the camera flash circuit is being charged. Common responses used with event-based filter are the following:

a. To block the CapSense data transmission until the pattern returns to normal
b. To reset the level of the baseline reference defined in Signal-to-Noise Ratio (SNR)
c. To drop or ignore the sample of data when the event occurred

An example for an event-based filter is dropping the sample or resampling of the data when the interrupt occurred. I^2C is one of the common communication protocols used in CapSense applications. Because I^2C interrupts are

asynchronous in nature, they may occur when the sensors are being scanned. interrupts which occur during the scan may increase the noise and therefore decrease the SNR. In such cases, you may implement an event-based filter such as ignoring the raw count sample corresponding to that scan when interrupts occurred and rescanning.

3.4.6 Rule-Based Filters

Rule-based filters are another special filtering method, where a pattern observed in the sensor data causes a rule-based response in the CapSense system. Unlike the event-based filter, the rule-based filter acts on patterns in the sensor data that are encountered during normal operation of the touch surface. The rule-based filter takes into account special scenarios on how sensors are used.

For example, with a set of radio channel selection buttons, two buttons can be touched accidentally, but only one should be selected. The rule-based filter sorts out this kind of situations in a predefined way. Another example is having a virtual sensor in CapSense applications. The virtual sensor is not expected to be triggered during normal operation of the sensors, but in unexpected scenarios such as presence of water (for example, the guard sensor) or when the system is subjected to RF noise. Therefore, when the virtual sensor is triggered, all the actual sensors are turned off so that there is no unintended triggering of the actual sensors.

3.5 Power Consumption

Minimizing power consumption is an important design goal. For many CapSense systems, extending battery life is critical to the success of the product. In systems that do not use batteries, power consumption still plays a role in optimizing power supply designs to reduce costs and PCB area.

3.5.1 Active and Sleep Current

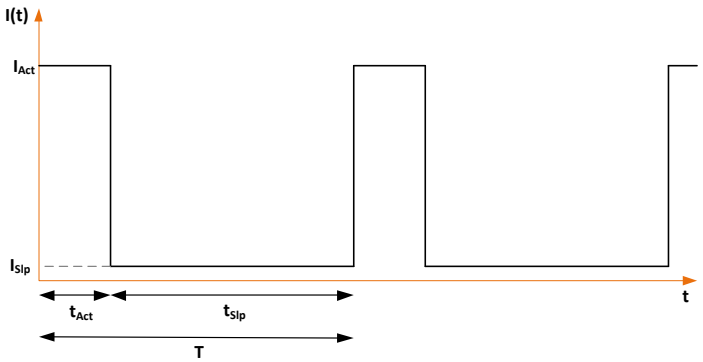
Active current is the current consumed by the device when all the selected analog and digital blocks are enabled and the CPU is running. In typical applications, the CapSense controller does not need to be in the active state all the time.

The device can be put into the sleep state to stop the CPU and the major blocks of the device. Current consumed by the device in sleep state is called sleep current. Sleep current is much lower than the active current.

3.5.2 Average Current

In typical applications, sleep state can be invoked periodically to reduce power consumption. This means that during a preset time period, the CapSense controller wakes up from sleep state, performs all necessary operations in the active state (scan all sensors, update all baselines, check if any sensor is in the TOUCH state, and so on), and then returns to sleep state. The resulting instantaneous current graph is shown in Figure 3-33.

Figure 3-33. Instantaneous Current



Where:

I(t) = Instantaneous current

I_{Act} = Active current

I_{Slp} = Sleep current

t_{Act} = Active time

t_{Slp} = Sleep time

T = Time period of a cycle

The average current consumed by the device over a long period can be calculated by using the following equation.

$$I_{AVE} = \frac{(I_{Act} \times t_{Act}) + (I_{Slp} \times t_{Slp})}{T}$$ Equation 24

The average power consumed by the device can be calculated as follows:

$$P_{AVE} = V_{DD} \times I_{AVE}$$ Equation 25

3.5.3 Response Time Versus Power Consumption

As illustrated in Equation 25, the average power consumption can be reduced by decreasing I_{AVE} or V_{DD}. I_{AVE} may be decreased by increasing sleep time. Increasing sleep time to a very high value leads to poor response time of the CapSense button. Because of this tradeoff between response time and power consumption, the application developer must carefully select the sleep time based on system requirements.

In any application, if both power consumption and response time are important parameters to be considered, then, an optimized method can be used that incorporates both continuous-scan and sleep-scan modes. In this method, the device spends most of its time in sleep-scan mode where it scans the sensors and goes to sleep periodically as explained in the previous section and thereby consuming less power. When you touch a sensor to operate the system, the device jumps to continuous-scan mode where the sensors are scanned continuously without invoking sleep and thereby giving very good response time. The device remains in continuous-scan mode for a specified time-out period. If you do not operate any sensor within this time-out period, the device returns to the sleep-scan mode.

3.6 Proximity Sensing

Proximity sensing is the process of detecting a nearby object without any physical contact. Proximity sensors use electromagnetic field, beam of electromagnetic radiation, or changes in ambient conditions to detect the proximity of a nearby object. There are various types of proximity sensors such as capacitive, inductive, magnetic, Hall effect, optical, and ultrasonic sensors; each has its own advantages and disadvantages. Capacitive proximity sensing has gained huge popularity because of its low cost, high reliability, low power, sleek aesthetics, and seamless integration with existing user interfaces.

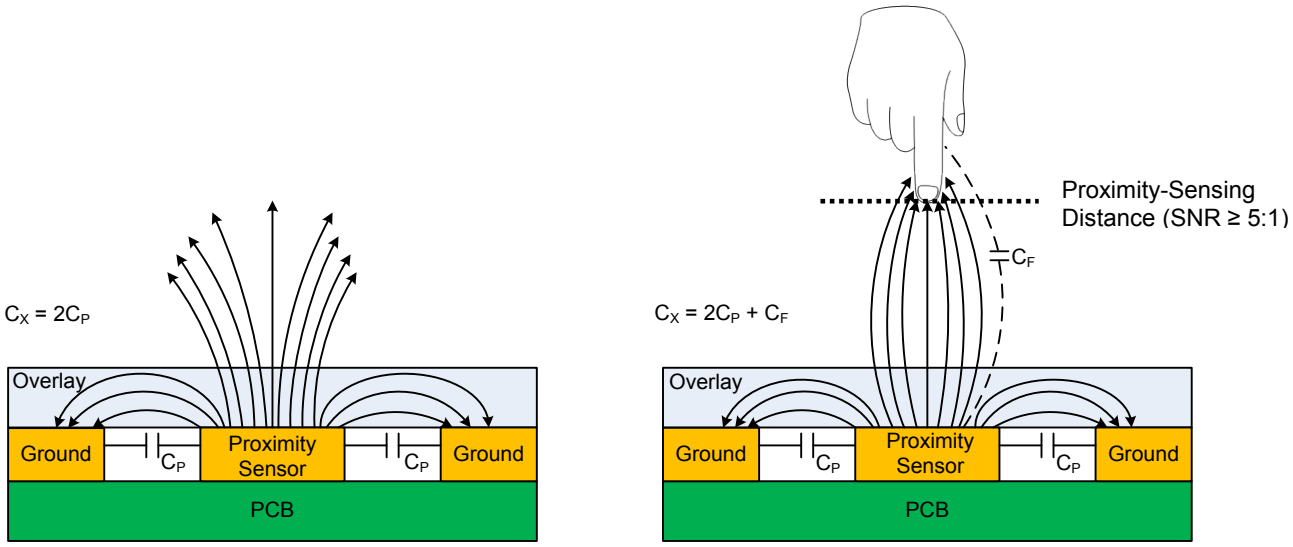
3.6.1 Proximity Sensing with CapSense

The proximity-sensing technique based in CapSense involves measuring the change in capacitance of a proximity sensor when a target object approaches the sensor. The target object can be a human finger, hand, or any conductive object. Proximity sensors based on CapSense can be constructed using a conductive (usually copper or indium tin oxide) pad or trace laid on a nonconductive material like PCB or glass. The intrinsic capacitance of the PCB trace or other connections to a capacitive sensor results in a sensor parasitic capacitance (C_P).

When a proximity sensor based on CapSense is excited by a voltage source, an electric field is created around the sensor. A small number of electric field lines couple to the nearby ground, while most of the electric field lines are projected into the nearby space, as shown in Figure 3-34 (a).

Figure 3-34. CapSense-Based Proximity Sensing

(a) Proximity Sensor Without a Finger Nearby (b) Proximity Sensor with a Finger Nearby



C_X = Total capacitance measured by the proximity-sensing system based on CapSense

C_P = Sensor parasitic capacitance

C_F = Capacitance added by a nearby target object

When a target object approaches the sensor, some of the electric field lines couple to the target object and add a small amount of finger capacitance (C_F) to the existing C_P, as shown in Figure 3-34 (b). This change in capacitance is measured by the CapSense circuitry to detect proximity of the target object.

For detecting the target object, the Signal-to-Noise Ratio (SNR) should be greater than or equal to 5:1. Therefore, you can detect the proximity of the target object without error up to a certain distance from the sensor. This distance is called the proximity-sensing distance, as shown in Figure 3-34 (b).

3.6.2 Implementing Proximity Sensing with CapSense

Figure 3-35 shows the steps for designing a proximity-sensing system based on CapSense.

1. **Understand proximity sensing:** The Proximity Sensing section in this design guide explains how proximity sensing based on CapSense works and describes the parameters that affect proximity-sensing distance.
2. **Evaluate how proximity sensing works:** Use Cypress's CY8CKIT-024 – CapSense Proximity Shield.
3. **Specify the proximity-sensing requirements:** After evaluating the proximity sensor performance, specify the proximity-sensing requirements such as the required proximity-sensing distance, area available on the PCB for sensor construction, system power consumption requirements, and EMI/ESD performance. These requirements help you to select the right CapSense device and design the sensor layout.
4. **Select the right CapSense device:** After the requirements are finalized, refer to the CapSense Selector Guide chapter to select the right CapSense device based on the required proximity-sensing distance.
5. **Design the schematic and layout:** After selecting the CapSense device, design the schematic and layout. Follow the guidelines mentioned in the Pin Assignments section to design the schematic. You should also refer to the device datasheet and device-specific CapSense Design Guides for details on the schematic design. For proximity-sensor layout guidelines such as proximity-sensor type and size, refer to the section Proximity Sensor Design and Factors Affecting Proximity Distance.
6. **Build the prototype:** After the schematic and layout design is completed, build the prototype of the design to check if the design meets the performance requirements.
7. **Tune:** Tune the prototype board to achieve the required performance. See the AN92239 – Proximity Sensing with CapSense and device-specific CapSense Design Guides.

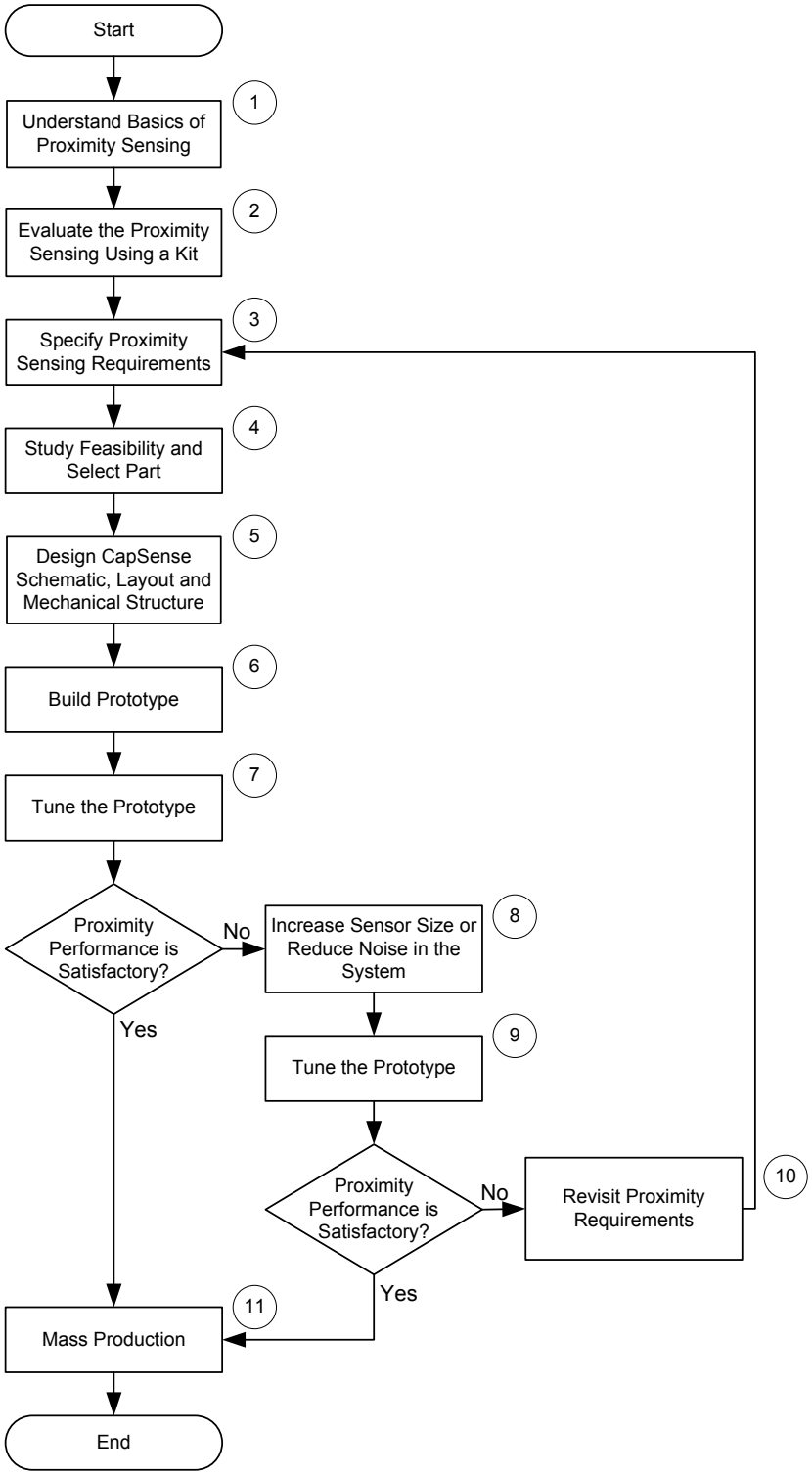
   After tuning the sensor, check if the proximity sensor performance meets the requirements. If the requirements are met, proceed to step 11; otherwise continue with Step 8.

8. **Redesign if necessary:** If the proximity sensor does not provide the required performance after you have set the optimum parameters, increase the sensor size or reduce the noise in the system by shielding the sensor from noise sources and continue with Step 9.
9. **Retune:** After redesigning the proximity sensor, retune the sensor and check if the sensor performance meets the requirements. If the requirements are met, proceed to Step 11; otherwise continue with Step 10.
10. **Revisit the design or requirements:** If the proximity sensor does not meet the required performance even after you have changed the sensor dimensions to the maximum possible value and tuned it with the optimum parameters, revisit the requirements.

    If you are not able to achieve the required proximity-sensing distance, select a device that has a better proximity performance than the current device.

    If you are not able to achieve the required proximity-sensing distance even with the best device, you need to change the proximity sensor requirements, such as the area available for the sensor or the required proximity-sensing distance, and repeat the procedure from Step 1.
11. **Proceed to mass production:** If the proximity sensor meets the required performance, you can proceed to mass production.

Figure 3-35. CapSense-based Proximity Sensing Design

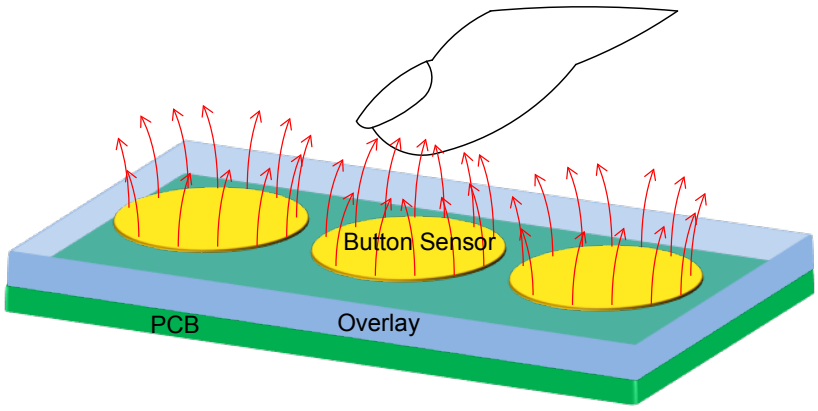


3.6.3 Proximity Sensor Design

A capacitive proximity sensor can be constructed using one of the following methods:

- **Button:** A button sensor when tuned for high sensitivity can be used as a proximity-sensor, as shown in Figure 3-36. The proximity-sensing distance is directly proportional to the sensor area. Because the diameter of a button sensor typically ranges from 5 mm to 15 mm, the proximity-sensing distance achieved with a button sensor is very less when compared to other sensor implementation methods.

Figure 3-36. CapSense-based Proximity Sensing with Button Sensor

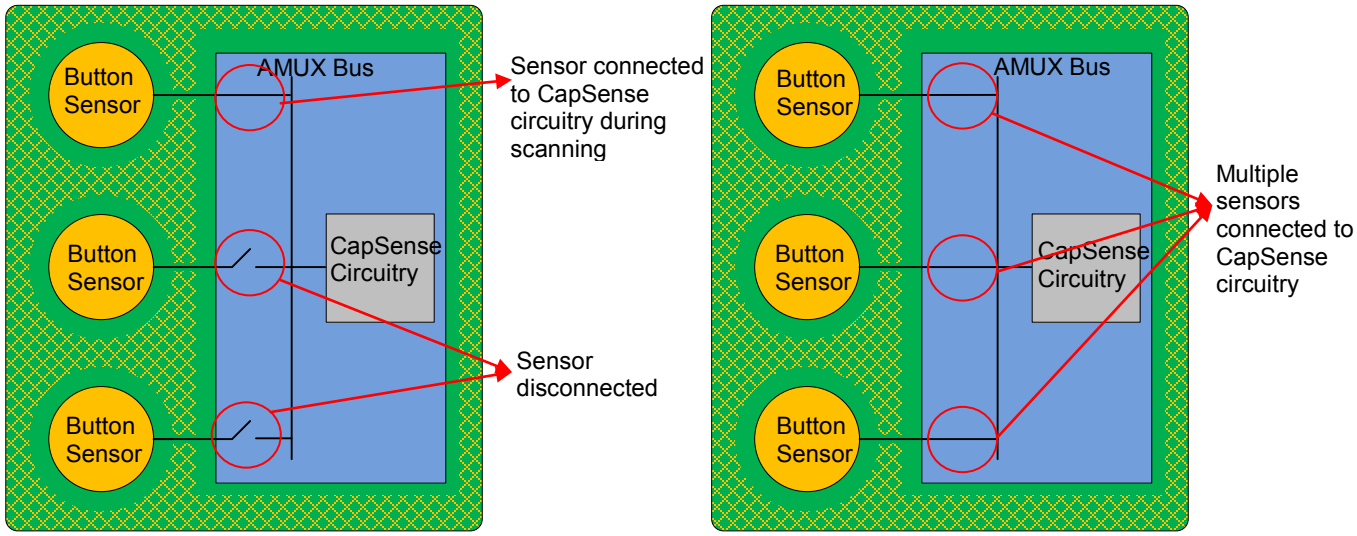


- **Sensor Ganging:** Sensor ganging refers to connecting multiple sensors (Buttons, Proximity trace, Proximity loop) to the CapSense circuitry and scanning them as a single sensor as shown in Figure 2-35 (b). Ganging multiple sensors increases the effective sensor area and results in higher proximity-sensing distance, but care should be taken to ensure that the C_P of the ganged sensor does not cross the maximum C_P limits supported by the device. Refer to AN92239 for details on how to implement proximity sensing using sensor ganging.

Figure 3-37. CapSense-based Proximity Sensing with Sensor Ganging

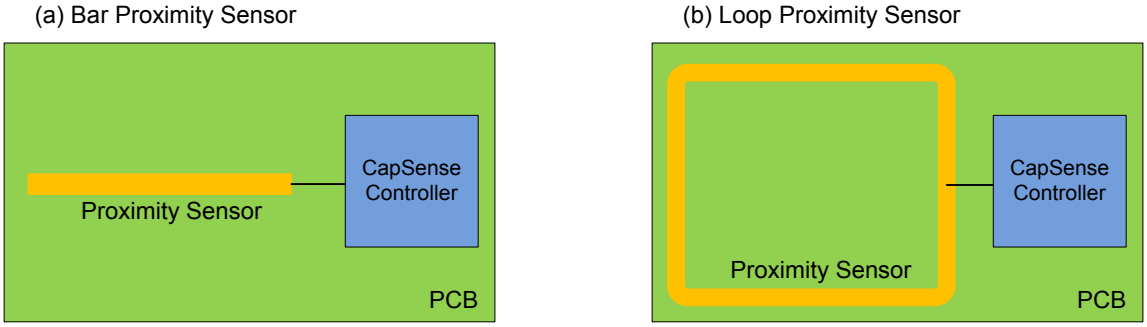
(a) Only one sensor is connected to AMUX bus during scanning

(b) Multiple sensors connected to AMUX bus at the same time for scanning



- **PCB Trace:** A long PCB trace on a FR4 or a Flexible Printed Circuit (FPC) board can form a proximity-sensor. The trace can be a straight line (Figure 3-38(a)), or it can surround the perimeter of a system's user interface, as shown in Figure 3-38 (b). Implementing a proximity-sensor with a PCB trace has the following advantages when compared to other sensor implementation methods:
  - Proximity-sensor C_P will be very less
  - Proximity–sensing distance will be higher because more electric field lines couples to the hand
  - More appropriate for mass production

Figure 3-38. CapSense-Based Proximity Sensing with PCB Trace



- **Wire:** A single length of wire works well as a proximity sensor. The proximity distance achieved with a wire loop sensor is much higher when compared to a PCB trace. But, using a wire sensor is not an optimal solution for mass production because of manufacturing cost and complexity.

3.6.4 Factors Affecting Proximity Distance

Proximity-sensing distance depends on the following hardware, software, and system parameters:

- Hardware parameters
  - Type of the sensor
  - Size of the sensor
  - Parasitic Capacitance (Cp) of the sensor
  - Overlay material and thickness
  - Nearby floating or grounded conductive objects
- Software parameters
  - Resolution of the sensor
  - Firmware filters
- System parameters
  - Power consumption
  - Response time
  - EMI/EMC/ESD performance

3.6.4.1 Hardware Parameters

- **Type of the sensor** – Proximity-sensing distance is directly proportional to the area of the sensor. A proximity-sensor implemented with the button sensor (typical diameter of 5-15 mm) has a small proximity sensing distance when compared to a proximity-sensor implemented using a PCB trace or wire with a diameter of 1-30 cm. Therefore, the required proximity-sensing distance decides the selection of proximity-sensor type. Table 3-7 shows when to use a specific proximity-sensor implementation method.

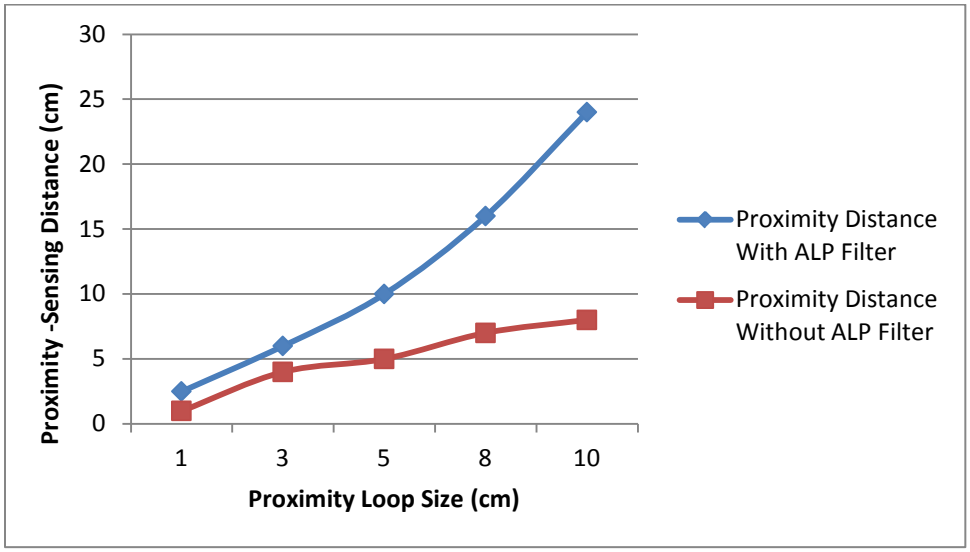
Table 3-7. Selecting Proximity Sensor Implementation Method

Proximity Sensor Type	When to Use
Button sensor	Use this method when the required proximity-sensing distance or the area available for the sensor is very small.
Ganged sensor	Use this method when there is no sensor pin or area available on the PCB for implementing a proximity sensor. Ganging sensors can achieve a larger proximity-sensing distance compared to using button sensors.
PCB trace	Use this method when the required proximity-sensing distance is very large. This method is preferred in most cases.
Wire loop	Use this method when the required proximity-sensing distance is very large. This method has the disadvantage of higher manufacturing cost compared to the implementation with PCB trace.

- **Size of the sensor:** The proximity sensor size depends on various factors such as the required proximity-sensing distance, presence of noise sources, and floating or grounded conductive objects. Noise sources and floating or grounded conductive objects reduce the SNR and the proximity-sensing distance. Therefore, large proximity sensors are needed to achieve the proximity-sensing distance required in your design.

  Figure 3-37 shows the relationship between the proximity-sensor loop size and the proximity-sensing distance for a given system. A larger sensor area results in more electric field lines coupling with the target object, resulting in increase in the sensor signal. However, a large sensor area results in a high sensor C_P and high noise, and thus reduces the proximity-sensing distance. Using a loop sensor (Figure 3-38 (b)) instead of a solid-fill sensor results in a low sensor C_P, low noise, and thus a large proximity-sensing distance. Also, loop sensors require less sensor area, leaving more space to place components on the PCB.

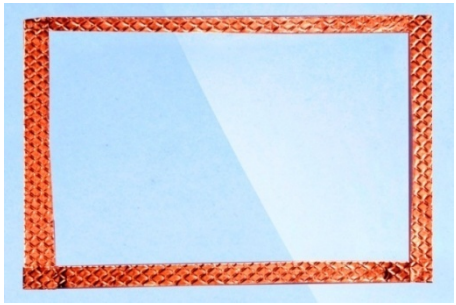
Figure 3-39. Proximity Loop Size Versus Proximity Distance



Note In the above graph, proximity-sensing distance for different loop sizes was measured under lab conditions. The actual proximity-sensing distance varies depending on the end-system environment.

It is very difficult to derive a relationship between the sensor size and the proximity-sensing distance. Depending on the end-system environment, the proximity-sensing distance may vary for a specific sensor size. You can find the sensor size required to achieve a required proximity-sensing distance by making sensor prototypes. You can use a copper foil, as Figure 3-40 shows, to make a quick sensor prototype to determine the sensor size required to achieve the required proximity-sensing distance.

Figure 3-40. Proximity Sensor Prototype Using Copper Tape



As a rule of thumb, it is recommended that you start with a minimum loop diameter (in the case of a circular loop) or diagonal (in the case of a square loop) equal to the required proximity-sensing distance. If you are not able to achieve the required proximity-sensing distance with a loop diameter or diagonal equal to the required proximity-sensing distance, you can increase the sensor loop diameter or diagonal until the required proximity-sensing distance is achieved.

Table 3-8 summarizes the proximity-sensor layout guidelines. If the area available for the proximity sensor is less than the area required to achieve the required proximity-sensing distance, you can implement firmware filters such as the Advanced Low Pass (ALP) Filter. The ALP filter attenuates the noise in the sensor raw count and increases the SNR. An increase in SNR results in a large proximity-sensing distance. Refer to AN92239 – Proximity Sensing with CapSense for details on ALP filter.

Table 3-8. Proximity Sensor Layout Recommendations

Details	Minimum	Recommendation
Proximity sensor loop diameter or diagonal	The sensor loop diameter or diagonal should be equal to or greater than the required proximity-sensing distance if the ALP filter is disabled. If the ALP filter is enabled, the sensor loop diameter or diagonal should be equal to or greater than half of the required proximity-sensing distance.	Start with a sensor loop diameter or diagonal equal to the required proximity-sensing distance and increase the diameter or diagonal until the required proximity-sensing distance is achieved.
Proximity sensor trace width	1.5 mm	1.5 mm

- **Parasitic capacitance of the sensor:** The proximity-sensing distance depends on the ratio of the C_F to the C_P. The proximity-sensing distance increases with an increase in the C_F/C_P ratio. For a given sensor size, the value of C_F depends on the distance between the sensor and the target object. To maximize this ratio, you need to increase C_F and decrease C_P. The C_P of the sensor can be minimized by selecting an optimum sensor area, reducing the sensor trace length, and minimizing the coupling of sensor electric field lines to the ground.

  To reduce the coupling of sensor electric field lines to the ground, drive the hatch fill in the top and bottom layer of the PCB (if there is any) with the driven-shield signal. A hatch fill that is connected to the driven-shield signal is called a “shield electrode.” The driven shield signal is a replica of the sensor signal.

  For shield electrode layout guidelines, see Shield Electrode and Guard Sensor.

  To minimize the sensor trace length and thereby the sensor C_P, place the CapSense device as close as possible to the sensor.

- **Nearby floating or grounded conductive objects:** The proximity-sensing distance reduces drastically if there is any nearby floating or grounded conductive object. The following factors cause the proximity-sensing distance to reduce drastically when conductive objects are placed close to the proximity sensor:

- The C_P of the sensor increases. Larger sensor C_P often requires reducing the sensor switching frequency, causing the proximity-sensing distance to decrease.
- A grounded conductive object catches a part of the sensor electric field and reduces the capacitance added by the target as shown in Figure 3-42.

You should either remove the nearby conductive object or use a shield electrode to isolate the proximity sensor from the conductive object. The influence of a nearby metal surface on the proximity sensor is reduced by placing a shield electrode between the proximity sensor and the metal object, as shown in Figure 3-43. For layout recommendations on shield electrode refer to Shield Electrode and Guard Sensor section.

Figure 3-41. Electrical Field Propagation for a Single Sensor Configuration Without a Metal Object

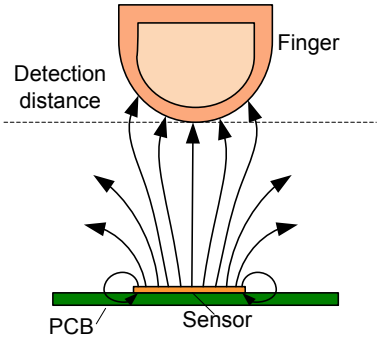


Figure 3-42. Electrical Field Propagation for a Single Sensor Configuration with a Solid Metal Object

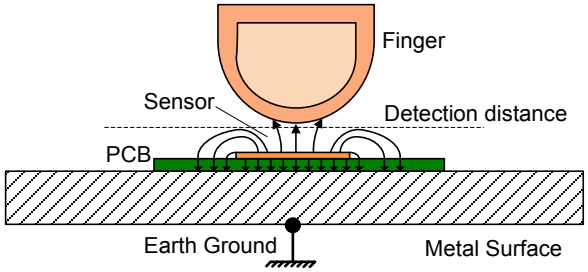
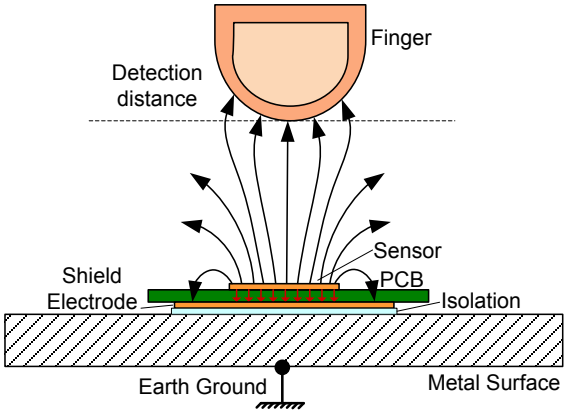


Figure 3-43. Using a Shield Electrode to Decrease the Metal Object's Influence



3.6.4.2 Software Parameters

- **Resolution of CSD:** The proximity-sensing distance is directly proportional to the resolution parameter of the CapSense sensing method. With a high resolution value, small changes in C_F can be detected with an SNR > 5:1. Detecting small changes in C_F essentially means a large proximity distance.

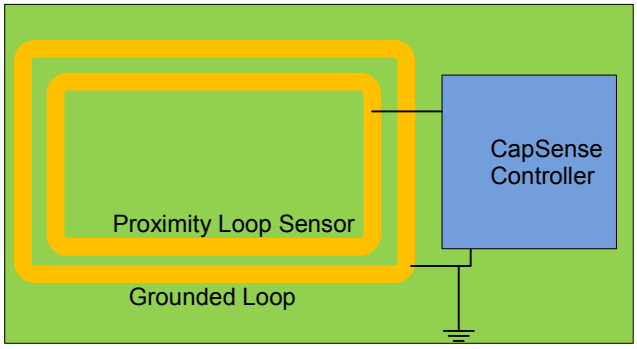
- **Firmware Filters:** Proximity-sensors are more susceptible to noise because of their large sensor area and high-sensitivity settings. High noise decreases SNR and hence reduces the proximity-sensing distance. Firmware filters helps in reducing the noise, thereby increasing the SNR and the proximity-sensing distance. You can use IIR, median, average or Advanced Low Pass (ALP) filters to reduce the noise. Refer to Software Filtering section for details on IIR, median, and average filters. Refer to the application note AN92239 for details on the ALP filter.

3.6.4.3 System Parameters

- **Power consumption:** Proximity sensors require scanning the sensor at a high resolution (15 or 16 bits) to achieve a large proximity-sensing distance. A higher resolution results in a longer scan time and increases the device active time, which leads to a higher power consumption. Therefore, larger proximity distance requires higher power consumption.
- **EMI/EMC/ESD performance:** To achieve a large proximity-sensing distance, the proximity sensor must be tuned for high sensitivity. A high-sensitivity setting results in reduced EMI/EMC performance. Therefore, there is a tradeoff between the proximity-sensing distance and the EMI/EMC performance.

  To improve the ESD performance of the proximity sensor, you can surround the sensor with a ground loop as shown in Figure 3-44.

Figure 3-44. Ground Loop Surrounding the Sensor for Improved ESD Performance



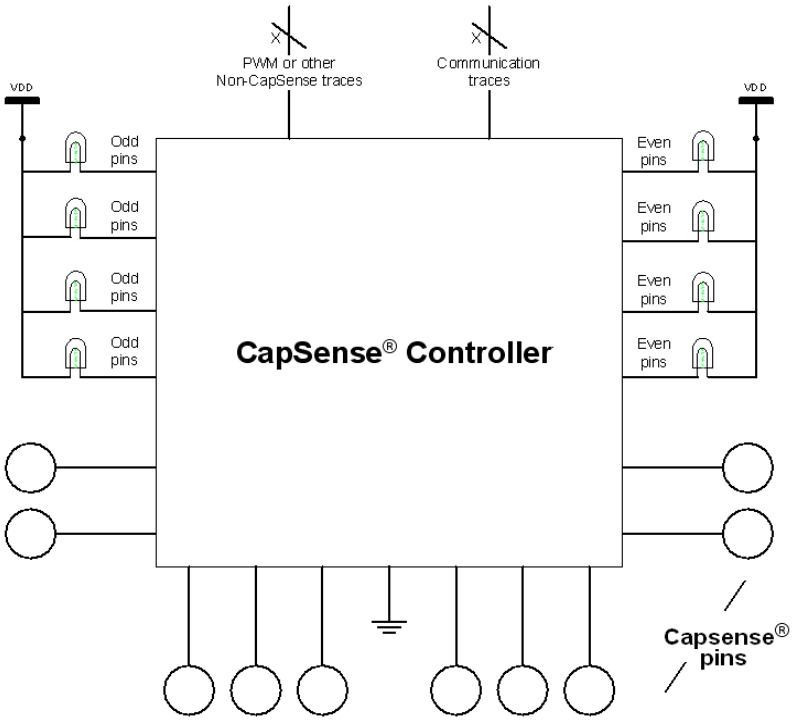
Having a ground loop around the sensor reduces noise in the proximity-sensor and provides a discharge path to the ground during ESD events but reduces the proximity-sensing distance. Therefore, there is a tradeoff between noise immunity and the proximity-sensing distance. The minimum recommended width for ground loop is 1.5 mm and the minimum air gap between the proximity loop and ground loop is 1 mm.

Note Having a ground loop with a small trace width (1.5 mm) around the sensor will not reduce the proximity distance by a drastic amount unlike a large grounded/floating conductive object.

3.7 Pin Assignments

An effective method to reduce interaction between CapSense sensor traces and communication and non-CapSense traces is to isolate each by port assignment. Figure 3-45 shows a basic version of this isolation for a 32-pin QFN package. Because each function is isolated, the CapSense controller is oriented such that there is no crossing of communication, LED, and sensing traces.

Figure 3-45. Recommended: Port Isolation for Communication, CapSense, and LEDs

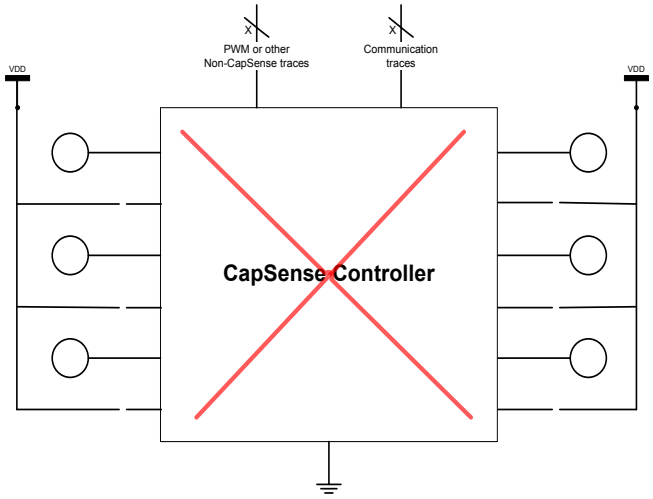


The CapSense controller architecture imposes a restriction on current budget for even and odd port pin numbers. For a CapSense controller, if the current budget of an odd port pin is 100 mA, the total current drawn though all odd port pins should not exceed 100 mA. In addition to the total current budget limitation, there is also a maximum current limitation for each port pin. See the datasheet of the CapSense controller used in the application to know the specification of that particular CapSense controller.

All CapSense controllers provide high current sink and source capable port pins. When using high current sink or source from port pins, select the ports that are closest to the device ground pin to minimize the noise.

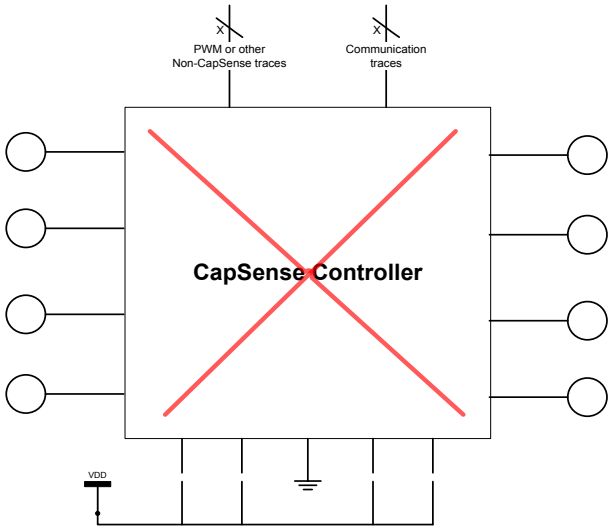
The following three examples demonstrate common pin assignment mistakes. In Figure 3-46, CapSense and non-CapSense traces are not isolated, and CapSense pins are far from ground. This is an example of a bad pin assignment.

Figure 3-46. Not Recommended - CapSense and Non-CapSense Pins in Proximity



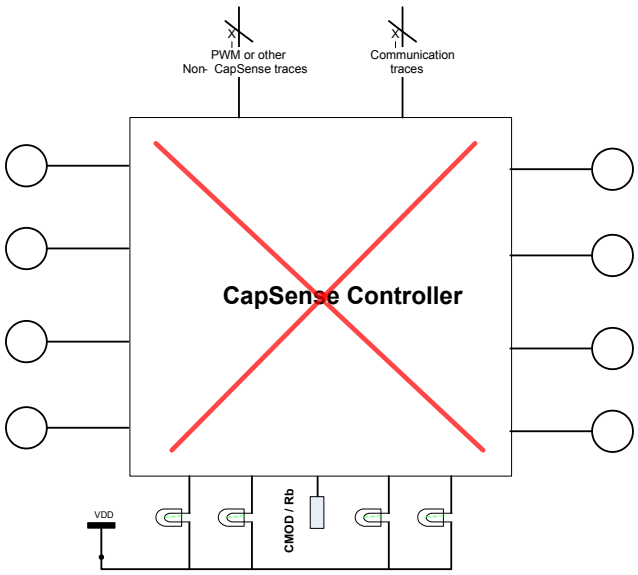
The example in Figure 3-47 achieves good isolation, but it has a bad pin assignment because the LEDs are placed next to the ground pin. The CapSense sensors are assigned to the side of the chip that does not include ground. If the CapSense pins are away from the ground pin, the impedance of the ground path increases, which in turn causes the drive circuit's reference voltage to shift. This shift may lead to false triggering of sensors. For this reason, it is recommended to have CapSense pins near the ground pin.

Figure 3-47. Not Recommended - LEDs and Ground Pins in Proximity



Further, LEDs should not be placed close to C_{MOD}/R_B pin to avoid crosstalk as illustrated in Figure 3-47.

Figure 3-48. Not Recommended: C_{MOD}/R_B and LED Pins in Proximity



Note that in PSoC1, using the P1.0 and P1.1 pins for LEDs or for communication purposes is not recommended. This is because, P1.0 and P1.1 pins are programming lines and upon power up, there will be a low pulse on the P1.0 and P1.1 pins. For further clarity you can also refer to the individual device design guides webpage having the sample schematics of all the CapSense devices:

- CY8C21X34 Design Guide
- CY8C20X34 Design Guide
- CY8C20XX6A Design Guide
- CY8C20XX7/S Design Guide

For similar guidelines on PSoC3, PSoC4 and PSoC5LP, refer to the respective datasheets and design guides.

3.8 PCB Layout Guidelines

In the typical CapSense application, the capacitive sensors are formed by the traces of a printed circuit board (PCB) or flex circuit. Following CapSense layout best practices will help your design achieve higher noise immunity, lower C_P, and higher signal-to-noise ratio (SNR). The CapSense signal drops off at high C_P levels due to drive limits of the internal current sources that are part of the CapSense circuitry. The long time constants associated with high C_P are another reason to avoid high C_P.

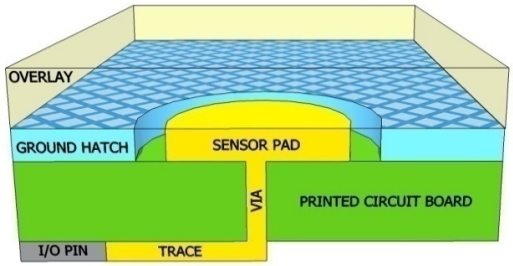
3.8.1 Parasitic Capacitance, C_P

The main components of C_P are trace capacitance and sensor capacitance. C_P is a nonlinear function of sensor diameter, trace length, trace width, and the annular gap. There is no simple relation between C_P and PCB layout features, but here are the general trends. An increase in sensor size, an increase in trace length and width, and a decrease in the annular gap all cause an increase in C_P. One way to reduce C_P is to increase the air gap between the sensor and ground. Unfortunately, widening the gap between sensor and ground will decrease noise immunity.

3.8.2 Board Layers

Most applications use a two-layer board with sensor pads and a hatched ground plane on the top side and all other components on the bottom side. The two-layer stack-up is shown in Figure 3-49. In applications where board space is limited or the CapSense circuit is part of a PCB design containing complex circuitry, four-layer PCBs are used.

Figure 3-49. Two-Layer Stack-Up for CapSense Boards



3.8.3 Board Thickness

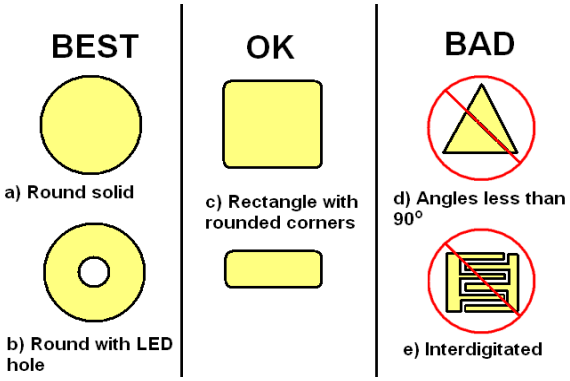
FR4-based PCB designs perform well with board thicknesses ranging from 0.020 inches (0.5 mm) to 0.063 inches (1.6 mm).

Flex circuits work well with CapSense, and are recommended for curved surfaces. All guidelines presented for PCBs also apply to flex. Ideally, flex circuits should be no thinner than 0.01 inches (0.25 mm). The high breakdown voltage of the Kapton® material (290 kV/mm) used for flex circuits provides built in ESD protection for the CapSense sensors.

3.8.4 Button Design

The best shape for buttons is round. Rectangular shapes with rounded corners are also acceptable. Because sharp points concentrate fields, avoid sharp corners (less than 90°) when designing your sensor pad.

Figure 3-50. Recommended Button Shapes



Button diameter can range from 5 mm to 15 mm, with 10 mm being suitable for the majority of applications. A larger diameter helps with thicker overlays.

Annular gap size should be equal to the overlay thickness, but no smaller than 0.5 mm, and no larger than 2 mm. For example, a PCB layout for a system with a 1-mm overlay should have a 1-mm annular gap, while a 3-mm overlay design should have a 2-mm annular gap. The spacing between the two adjacent buttons should be large enough that if one button is pressed, a finger should not reach the annular gap of the other button.

3.8.5 Slider Design

Figure 3-51 shows the recommended slider pattern for a linear slider and Table 3-9 shows the recommended values for each of the linear slider dimensions. Detailed explanation on the recommended layout guidelines are provided in the following sections.

Figure 3-51. Typical Linear Slider Pattern

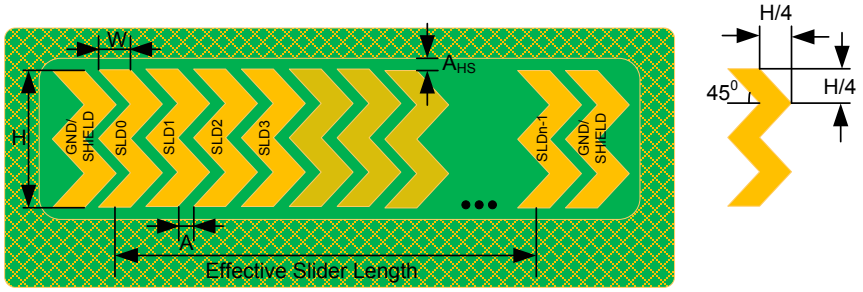


Table 3-9. Linear Slider Dimensions

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

[a] The recommended slider-segment-width is based on an average human finger diameter of 9 mm. Refer to Section 3.8.5.1, “Slider-Segment Shape, Width, and Air Gap” for more details.

3.8.5.1 Slider-Segment Shape, Width, and Air Gap

A linear response of reported finger position (i.e. Centroid) vs. actual finger position on a slider requires that a slider design be such that whenever a finger is placed anywhere between middle of segment SLD0 and middle of segment SLDn-1, other than the exact middle of slider segments, exactly two sensors report a valid signal[a]. If finger is placed at the exact middle of any slider segment, the adjacent sensors should report difference count = noise threshold. Therefore, it is recommended to use a double chevron shape, as Figure 3-51 shows. This shape helps in achieving a centroid response close to the ideal response, as Figure 3-52 and Figure 3-53 show. For the same reason, the slider-segment width and air-gap (i.e. dimensions "W" and "A" respectively, as marked in Figure 3-51) should follow the relation mentioned in Equation 3-1.

Figure 3-52. Ideal Slider Segment Signals and Centroid Response

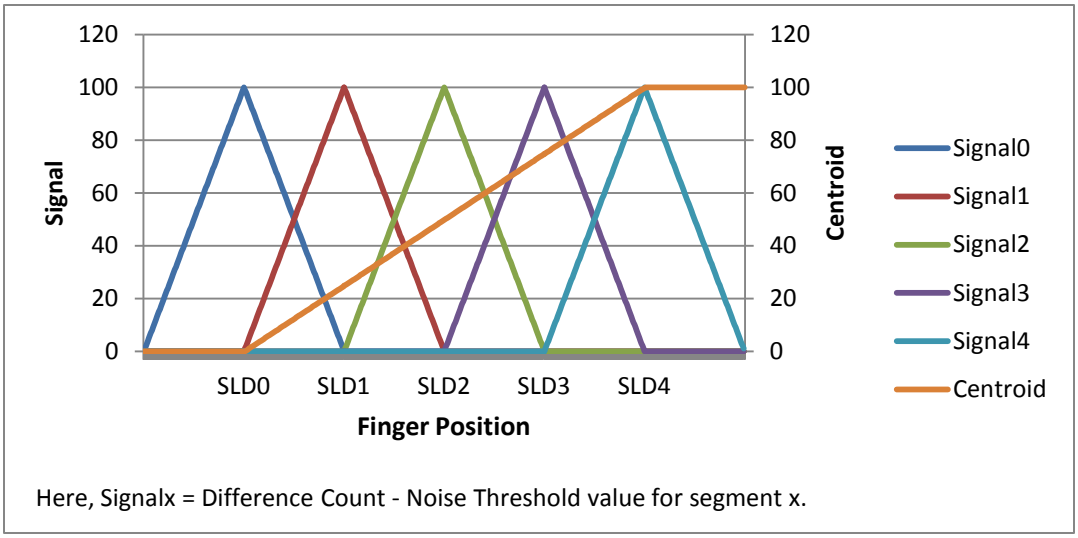
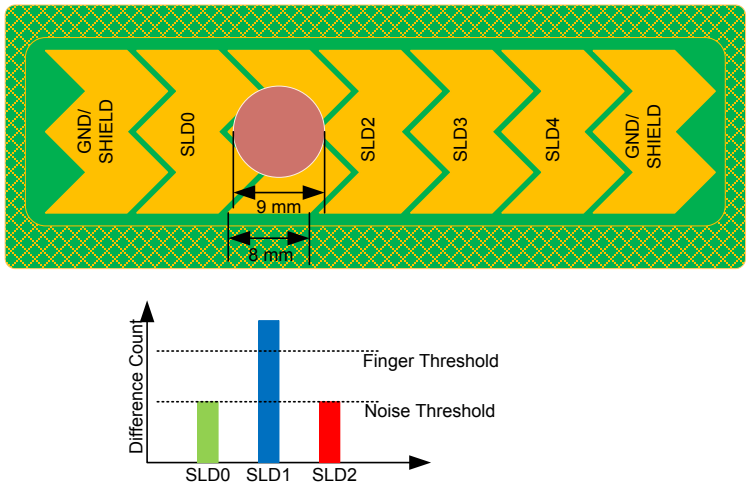


Figure 3-53. Ideal Slider Signals



Equation 3-1 Segment width and air-gap relation with finger diameter

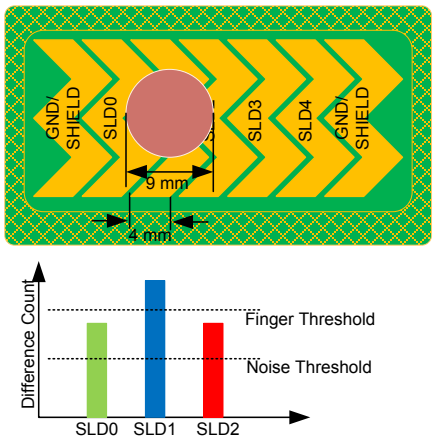
$$W + 2A = finger\ diameter$$

Typically, an average human finger diameter is approximately 9 mm. Based on this average finger diameter and Equation 3-1, the recommended slider-segment-width and air-gap is 8 mm and 0.5 mm respectively.

[a] Here, a valid signal means that the difference count of the given slider-segment is greater than or equal to the noise threshold value

If the *slider-segment-width + 2 * air-gap* is lesser than *finger diameter*, as required per Equation 3-1, the centroid response will be non-linear. This is because, in this case, a finger placed on the slider will add capacitance, and hence valid signal to more than two slider-segments at some given position, as Figure 3-54 shows. Thus, calculated centroid position per Equation 3-1 will be non-linear, as Figure 3-54 shows.

Figure 3-54. Finger Causes Valid Signal on More Than Two Segments when Slider Segment Width Is Lower Than Recommended



Equation 3-2. Centroid algorithm used by CapSense

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

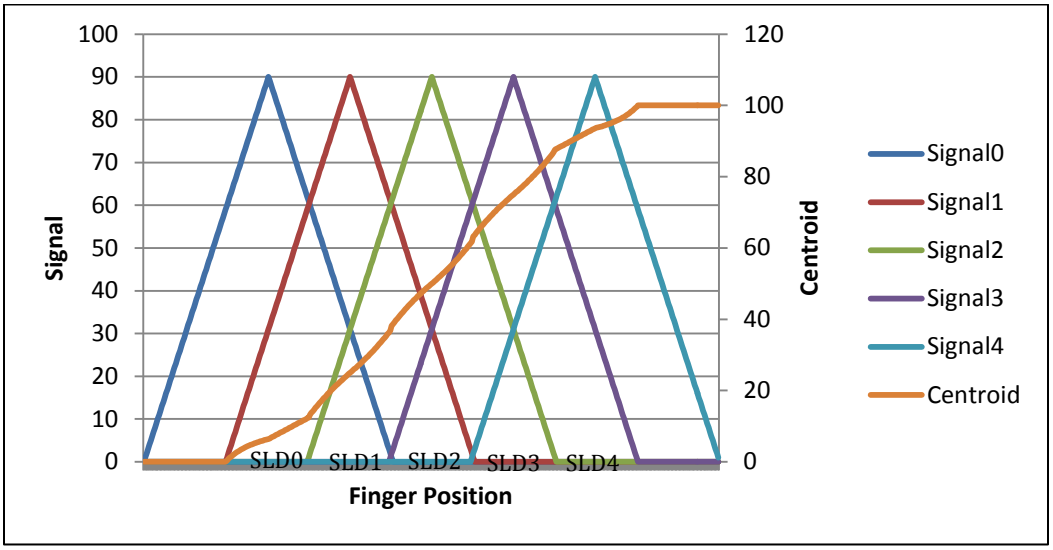
Resolution - API Resolution set in the Customizer,

n - Number of sensor elements in the Customizer.

maximum: Index of element which gives maximum signal.

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

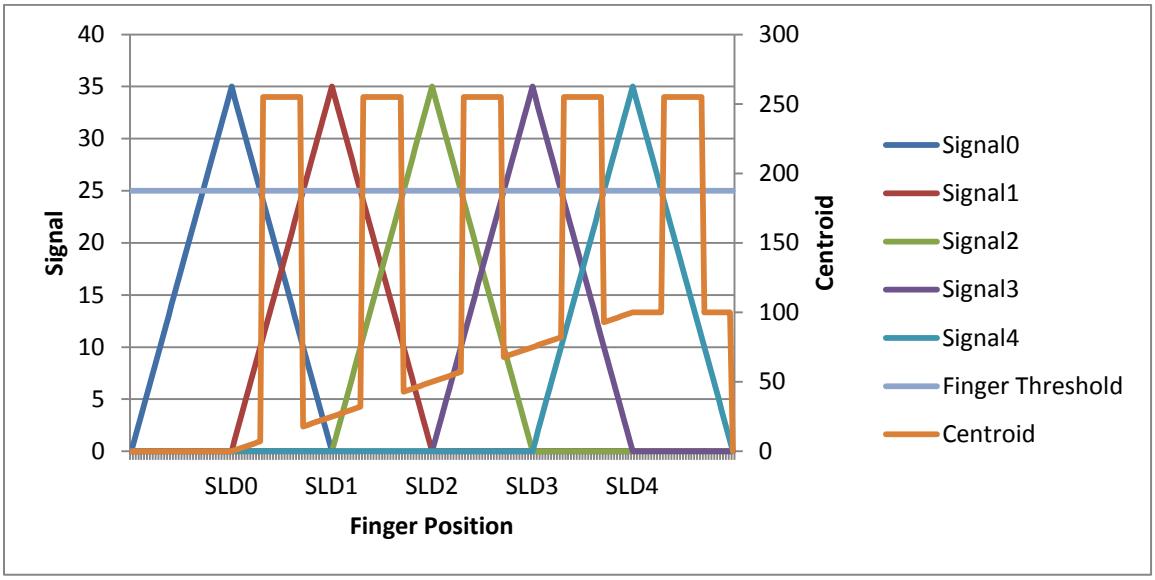
Figure 3-55. Nonlinear Centroid Response when Slider Segment Width Is Lower Than Recommended



Note that even though a *slider-segment-width* value of less than *finger diameter – 2 * air-gap* provides a non-linear centroid response, as Figure 3-54 shows; it may still be used in an end application where the linearity of reported centroid vs actual finger position does not play a significant role. However, a minimum value of slider-segment-width

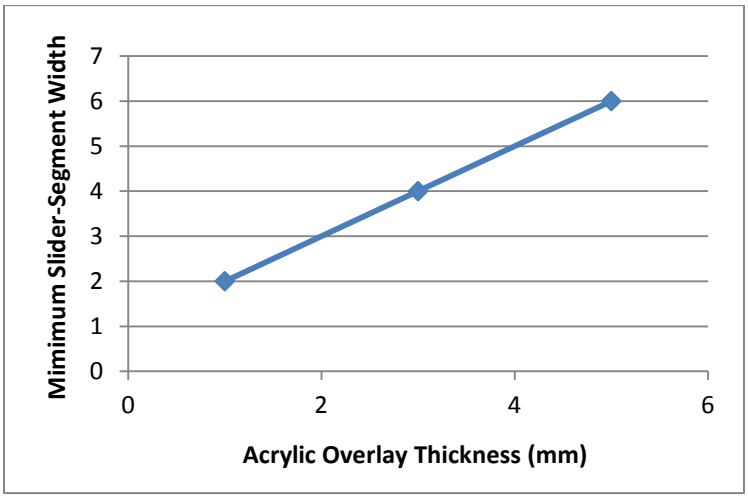
must be maintained, based on overlay thickness, such that, at any position on the effective slider length, at-least one slider-segment provides an SNR of >=5:1 (i.e. signal >= Finger Threshold parameter) at that position. If the slider-segment-width is too low, a finger may not be able to couple enough capacitance, and hence, none of the slider-segments will have a 5:1 SNR, resulting in a reported centroid value of 0xFF[a], as Figure 6-11 shows.

Figure 3-56. Incorrect Centroid Reported when Slider Segment Width Is Too Low



The minimum value of slider-segment-width for certain specific overlay thickness values, for an acrylic overlay, are provided in Table 3-9. For acrylic overlays of thickness values, which are not specified in Table 3-9, Figure 3-57 may be used to estimate the minimum slider-segment-width.

Figure 3-57. Minimum Slider-Segment-Width w.r.t. Overlay Thickness for an Acrylic Overlay



[a] The CapSense component in PSoC Creator reports a centroid of 0xFF when there is no finger detected on the slider, or when none of the slider-segments reports a difference count value greater than Finger Threshold parameter.

If the *slider-segment-width + 2 * air-gap* is higher than *finger diameter*, as required per Equation 3-1, the centroid response will have flat spots i.e., if the finger is moved a little near the middle of any segment, the reported centroid position will remain constant as Figure 3-58 shows. This is because, as Figure 3-59 shows, when the finger is placed in the middle of a slider-segment, it will add valid signal only to that segment even if the finger is moved a little towards the adjacent segments.

Figure 3-58. Flat Spots (Nonresponsive Centroid) when Slider Segment Width Is Higher Than Recommended

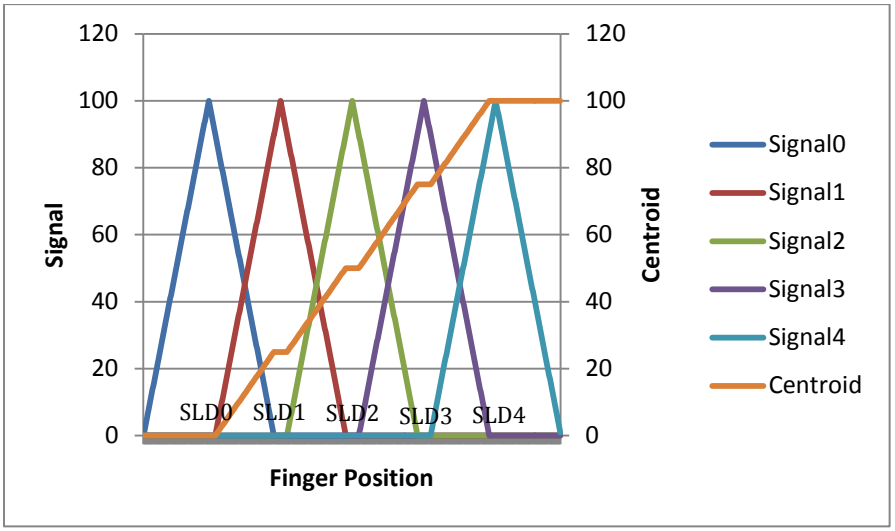
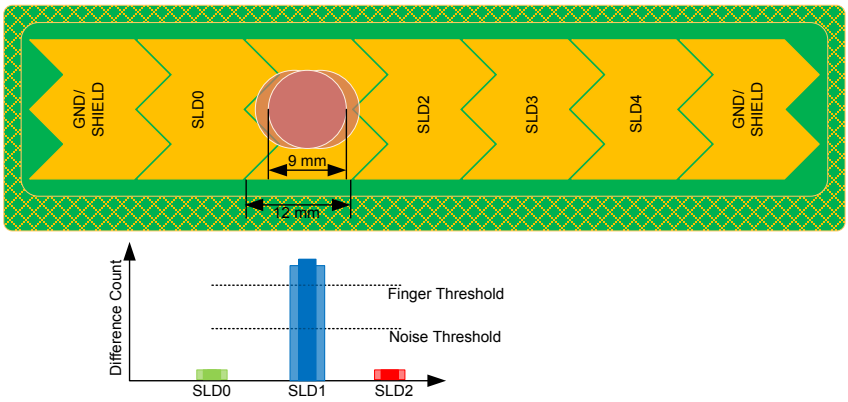


Figure 3-59. Signal on Slider Segments when Slider Segment Width Is Higher Than Recommended



Note that if the *slider-segment-width + 2 * air-gap* is higher than *finger diameter,* it may be possible to increase and adjust the sensitivity of all the slider segments such that even if the finger is placed in middle of a slider-segment, the adjacent sensors report a difference count value equal to noise threshold value (as required per Figure 3-52. Ideal Slider Segment Signals and Centroid Response); however, this will result in hover effect i.e. the slider may report a centroid position even if the finger is hovering above the slider and not touching the slider.

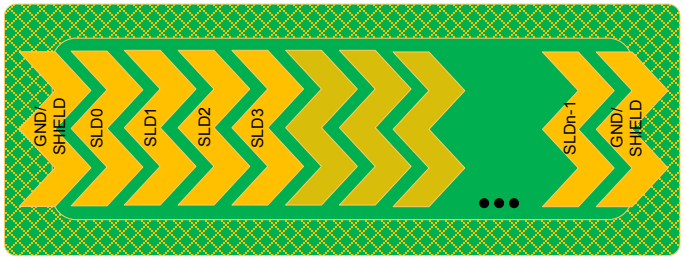
3.8.5.2 Dummy Segments at the Ends of Slider

In a CapSense design, when one segment is scanned; the adjacent segments are connected to either ground or to the driven shield signal based on the option that will be specified in the “Inactive sensor connection” parameter in the CapSense CSD component. For linear centroid response, the slider requires all the segments to have same sensitivity i.e. the increase in the raw count (signal) when a finger is placed on the slider segment should be same for all the segments. To maintain a uniform signal level from all the slider segments, it is recommended to physically connect the two segments at the both ends of a slider to either ground or driven shield signal. The connection to ground or to the driven shield signal depends on the value that will be specified in the “Inactive sensor connection” parameter. Therefore, if your application requires an ‘n’ segment slider, it is recommended to create n + 2 physical segments, as Figure 3-51 shows.

If it is not possible to have two segments at the both ends of a slider due to space constraints, you can implement these segments in the top hatch fill, as Figure 3-60 shows. Also, if the total available space is still constrained, the width of these segments may be kept lesser than the width of segments SLD0 through SLDn-1, or these dummy segments may even be removed.

If the two segments at the both ends of a slider are connected to the top hatch fill, you should connect the top hatch fill to the signal that will be specified in the “Inactive sensor connection” parameter. If liquid tolerance is required for the slider, the hatch fill around the slider, the last two segments and the inactive slider segments should be connected to driven shield signal.

Figure 3-60. Linear Slider Pattern when First and Last Segments Are Connected to Top Hatch Fill



3.8.5.3 Deciding Slider Dimensions

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

a) Decide the required length of slider (L), based on application requirements. This is same as the “effective slider length” as Figure 3-51 shows.
b) Decide the height of segment based on available space on the board. Use maximum allowed segment height (15 mm) if board space permits, else, use a lesser height but ensure that the height is greater than the minimum specified in Table 3-9.
c) The slider-segment-width and the air-gap between slider segments should be as recommended in Table 3-9.
   The recommended slider-segment-width and air-gap for an average finger diameter of 9 mm is 8 mm and 0.5mm respectively.
d) For a given slider length, L, calculate the number of segments required using the following formula:

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

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

   If the available number of CapSense pins is slightly lesser than the calculated number of segments for a certain application, you may increase the segment width to achieve the required slider length with given number of pins. For example, a 10.2 cm slider requires 13 segments. However, if only 10 pins are available, segment width may be increased to 10.6. This will either result in a non-linear response as Figure 3-58 shows, or a hover effect; however, this layout may be used if the end application does not need a high linearity.

   Note that the PCB length is higher than the required slider length as Figure 3-51 shows. PCB length can be related to slider length as follows:

   Equation 3-3 Relationship between minimum PCB length and slider length

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

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

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

Keep in mind the following layout guidelines while designing a slider:

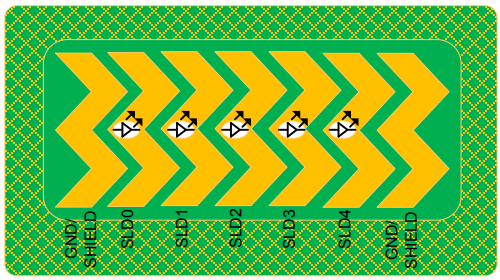
- Design the shape of all segments to be as uniform as possible.

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

3.8.5.4 Slider Design with LEDs

In some applications it might be required to display finger position by driving LEDs. You can either place the LEDs just above the slider segments or drill a hole in the middle of a slider segment for LED backlighting, as Figure 6-8 shows. When a hole is drilled for placing an LED, the effective area of the slider segment reduces. To achieve an SNR > 5:1, you need to have a slider segment with a width larger than the LED hole size. Refer to Table 3-9 for minimum slider width required to achieve an SNR > 5:1 for a given overlay thickness. Follow the guidelines provided in Crosstalk Solutions section for routing the LED traces.

Figure 3-61. Slider Design with LED Backlighting



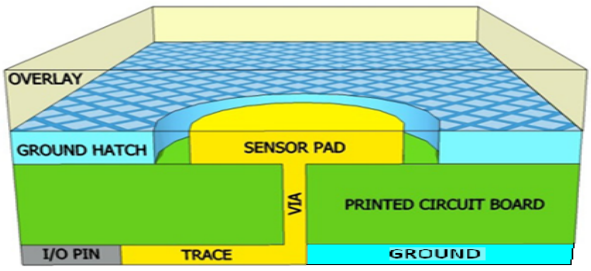
3.8.6 Sensor and Device Placement

For a CapSense design on 2-Layer and 4-Layer PCBs, follow the below guidelines for sensor and component placement. If your design requires liquid tolerance, follow the guidelines explained in the Hardware Component section.

2-Layer PCB:

- Place the sensors on the top layer of the PCB, as Figure 3-62 shows.
- Place the components and route the sensor traces on the bottom layer of the PCB.

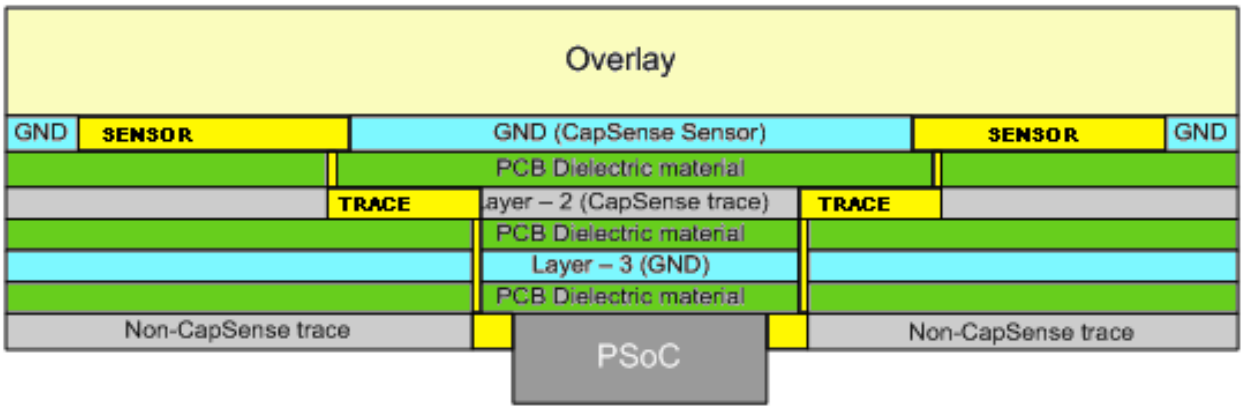
Figure 3-62. CapSense Design on 2-Layer PCB



4-Layer PCB:

- Place the sensors on the top layer of the PCB
- Route the sensor traces in the layer-2
- Place a hatch fill of 7-mil trace and 70-mil spacing and connect it to ground in layer-3
- Place components in the bottom layer, as Figure 3-63 shows. The unoccupied areas can be filled with a hatch copper fill of 7-mil trace and 70-mil spacing and should be connected to ground.

Figure 3-63. CapSense Design on 4-Layer PCB



In addition to these guidelines, follow the best practices to ensure a robust and reliable CapSense design.

- Minimize the trace length from the CapSense controller pins to the sensor pad to optimize signal strength.
- Mount series resistors within 10 mm of the controller pins to reduce RF interference and provide ESD protection.
- Mount the controller and all 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 the sensor PCB traces. Do this by placing them at least 4 mm apart and fill a hatched ground between CapSense traces and non-CapSense traces to avoid crosstalk.
- Avoid connectors between the sensor and the controller pins because connectors increase C_P and decrease noise immunity.

3.8.7 Trace Length and Width

Minimize the parasitic capacitance of the traces and sensor pad. Trace capacitance is minimized when they are short and narrow.

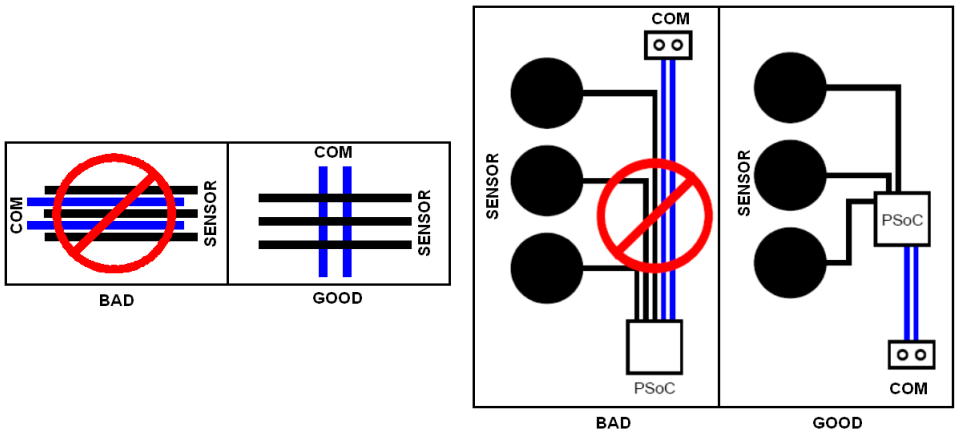
- The maximum recommended **trace length** is 12 inches (300 mm) for a standard PCB and 2 inches (50 mm) for flex circuits.
- **Trace width** should not be greater than 7 mil (0.18 mm). CapSense traces should be surrounded by hatched ground with trace-to-ground air gap of 10 mil to 20 mil (0.25 mm to 0.51 mm).

3.8.8 Trace Routing

Route sensor traces on the bottom layer of the PCB, so that the only user interaction with the CapSense sensors is with the active sensing area. Do not route traces directly under any sensor pad unless the trace is connected to that sensor.

Do not run capacitive sensing traces in close proximity to communication lines, such as I^2C or SPI masters. If it is necessary to cross communication lines with sensor pins, make sure the intersection is at right angles, as illustrated in Figure 3-64.

Figure 3-64. Routing of Sensing and Communication Lines



3.8.9 Crosstalk Solutions

A common backlighting technique for panels is to mount an LED under the sensor pad so that it shines through a hole in the middle of the sensor. When the LED is switched on or off, the voltage transitions on the trace that drives the LED can couple into the capacitive sensor input, creating noisy sensor data. This coupling is referred to as crosstalk. To prevent crosstalk, isolate CapSense and non-CapSense traces from one another. In the case of CY8C21X34/B, the crosstalk can also occur due to the coupling of the LED voltage transitions with the R_B resistor. To avoid this, isolate the R_B trace from the non-CapSense traces. A minimum separation of 4 mm is recommended. A hatched ground plane also can be placed between those traces to isolate them. LED drive traces and CapSense traces (including R_B trace) should not be routed together.

Figure 3-65. Not Recommended - LED and CapSense in Close Proximity

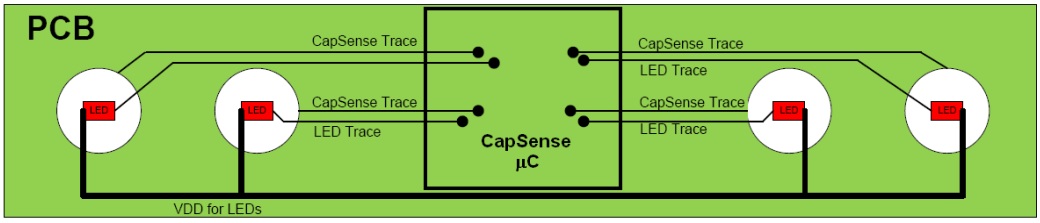
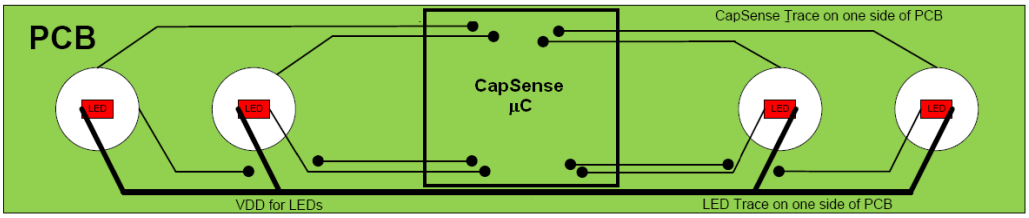
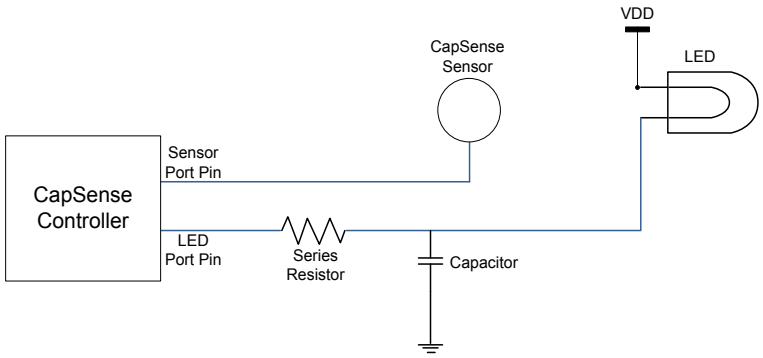


Figure 3-66. Recommended - LED and CapSense with Wide Separation



Another approach to reducing crosstalk is to slow down the rising and falling edges of the LED drive voltage using a filter capacitor. Figure 3-67 shows an example circuit of this solution. The value of the added capacitor depends on the drive current requirements of the LED; however, a value of 0.1 µF is typical.

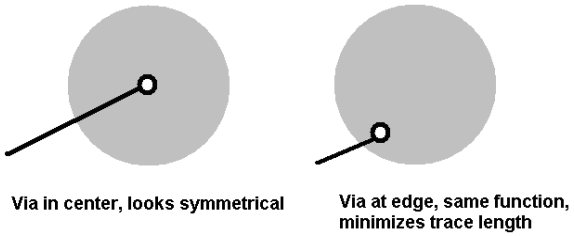
Figure 3-67. Filter Capacitor Solution for Crosstalk



3.8.10 Vias

Use the minimum number of vias to route CapSense inputs to minimize parasitic capacitance. The vias should be placed to minimize the trace length, which is usually on the edge of the sensor pad, as shown in Figure 3-68.

Figure 3-68: Vias Placement on Sensor Pad



3.8.11 Ground Plane

Ground fill is added to both the top and bottom of the sensing board. When ground fill is added near a CapSense sensor pad, there is a tradeoff between maintaining a high level of CapSense signal and increasing the noise immunity of the system. Typical hatching for the ground fill is 25 percent on the top layer (7 mil line, 45 mil spacing) and 17 percent on the bottom layer (7 mil line, 70 mil spacing).

Figure 3-69: Recommended Button and Slider Layout Top Layer

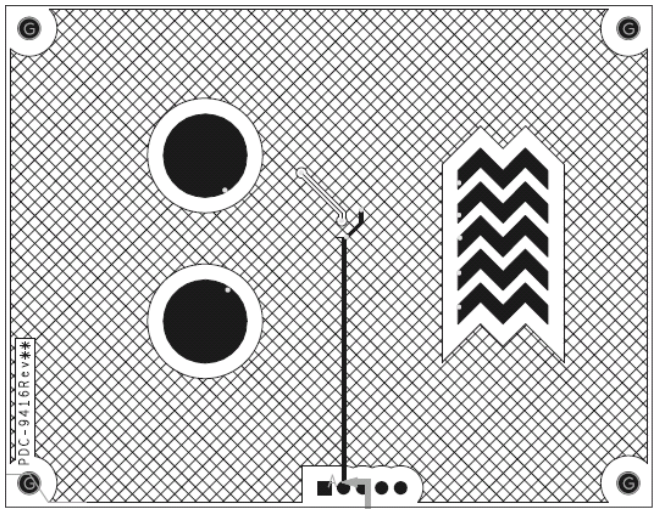
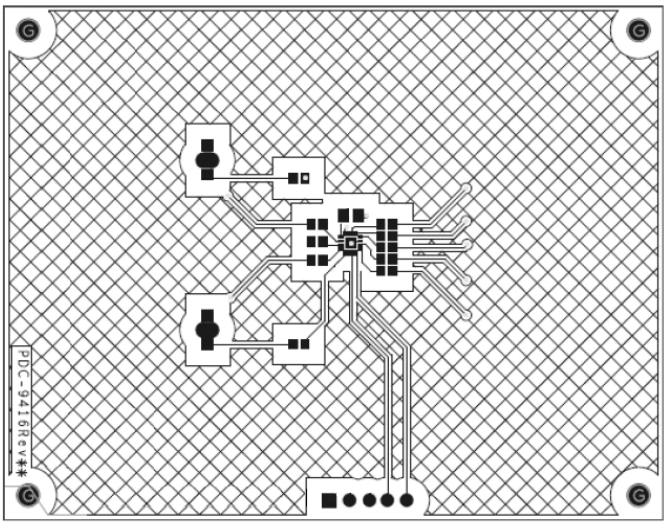


Figure 3-70. Recommended Button and Slider Layout Bottom Layer



3.8.12 Power Supply Layout Recommendations

A poor PCB layout would introduce noise in high-sensitivity sensors (e.g., proximity sensors and button sensors that use overlays thicker than 1 mm). To achieve low noise on high-sensitivity CapSense sensors in designs, it is important that the PCB layout follows the best practices on power supply trace and placement.

1. Two decoupling capacitors must be connected between the VDD and VSS pins. (Capacitor specification: 0.1 µF and 1 µF, 16 V, Ceramic, X7R).
2. One decoupling capacitor must be connected between the VCC and VSS pins for CY8CMBR3XXX devices. (Capacitor specification: 0.1 µF, 16 V, Ceramic, X7R).
3. When using packages E-pad (paddle), it must be connected to the board GND.
4. The decoupling capacitors and CMOD capacitor must be connected as close as possible to the chip to keep impedance of the ground and supply traces as low as possible.

Figure 3-71 illustrates the schematic diagram with these recommendations. C1, C2, and C4 are decoupling capacitors and C3 is the CMOD capacitor.

Figure 3-71. Example Schematics for Improved SNR

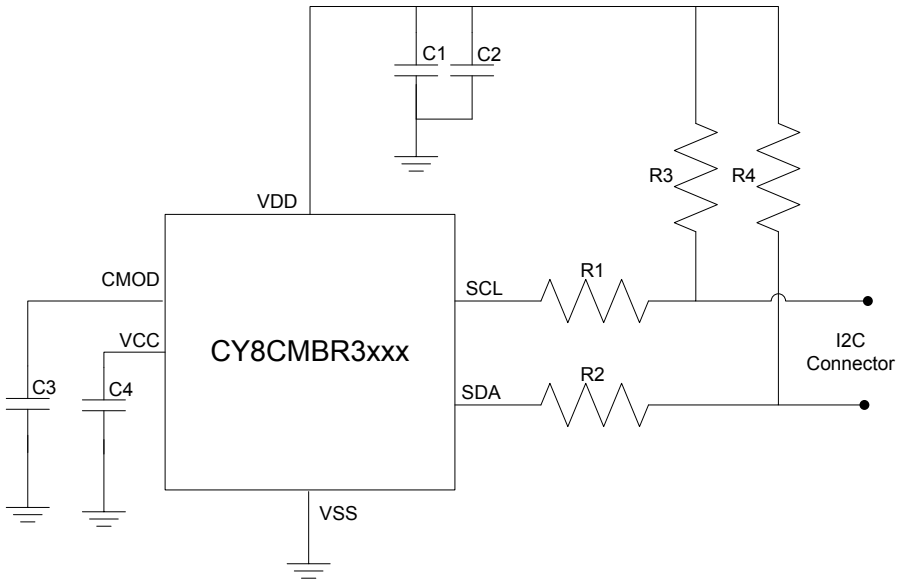
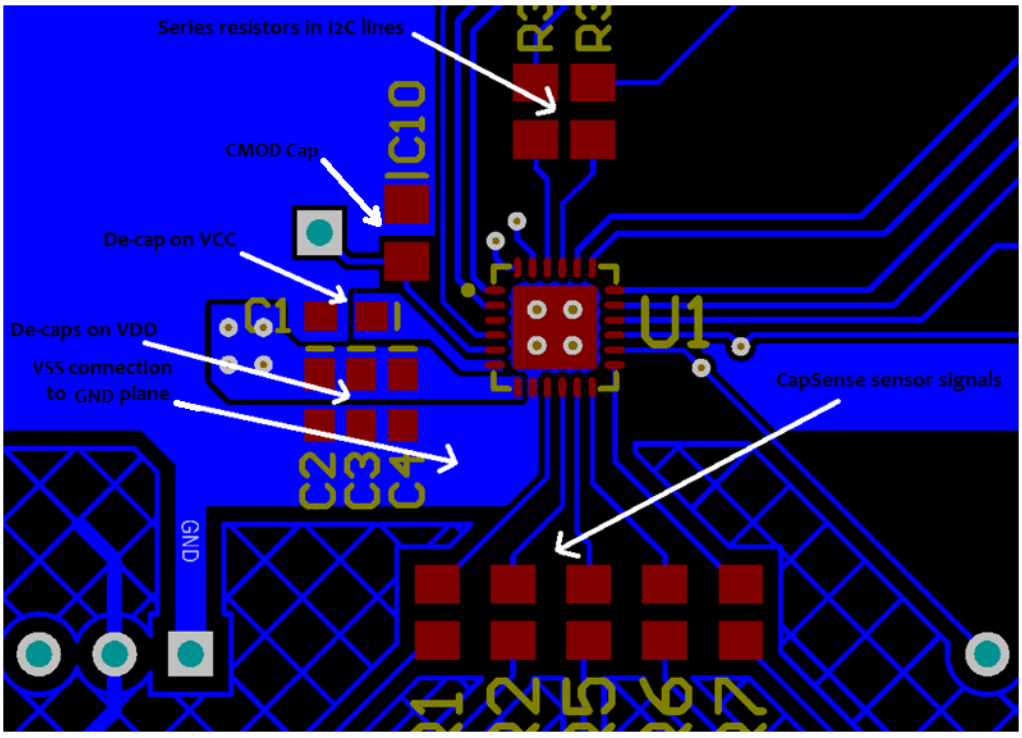


Figure 3-72 shows an example board layout diagram with placements of decoupling and CMOD capacitors and routing of ground and supply. (Note that the I^2C pull-ups present in Figure 3-71 are not shown in the layout in this figure).

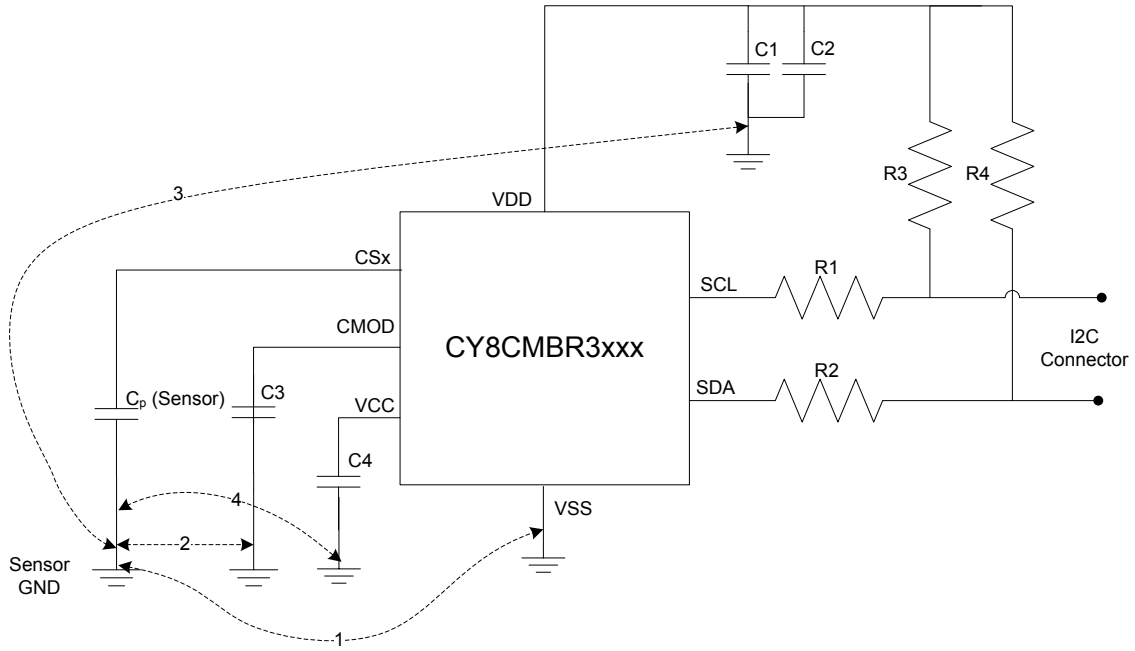
Figure 3-72. Example Board Layout for Improved SNR



A good CapSense schematics diagram must have all passive components shown in the schematics diagram.

The PCB layout shown above is for guidelines only. For a good layout, the inductance between multiple ground nodes must be below 0.2 nH. The ground nodes are indicated in the schematics diagram in Figure 3-73.

Figure 3-73. Important GND Nodes in CapSense Design



3.8.13 Shield Electrode and Guard Sensor

A shield electrode is a hatched fill that is driven with a signal which is the replica of the sensor signal. A shield electrode is used for the following purposes:

- **Reduce sensor parasitic capacitance (C_P):** In most of the CapSense applications it is recommended to have a hatch fill in the top and bottom layer of the PCB surrounding the sensor and its traces. This hatch fill is connected to ground for improved noise immunity. When a sensor has a large trace length it results in high sensor C_P. High sensor C_P results is low sensor sensitivity and increases power consumption. To reduce the sensor C_P you should drive the hatch fill in the top and bottom layer with a driven shield signal.
- **Reduce the effect of nearby floating/grounded conductive objects:** As explained in the Factors Affecting Proximity Distance section, shield electrode can be used to reduce the effect of floating/grounded conductive objects on proximity distance. In this case, the shield electrode should be placed in between the conductive object and the proximity sensor, as shown in Figure 3-43.
- **Provide directionality to proximity sensing:** The electric field of a proximity sensor is omni directional and can detect proximity in all directions. In most of the applications, it is required to detect proximity from only one direction. In such cases, you can use a shield electrode to make the proximity sensor sense the target object in a single direction.
- **Provide liquid tolerance:** As explained in the Liquid Tolerance section, shield electrodes prevent false triggers due to the presence of liquid droplets on the CapSense sensor.

3.8.13.1 Shield Electrode for Proximity Sensing

If you want to use shield electrode for reducing sensor C_P or reduce the effect of nearby floating/grounded conductive objects or provide directionality to proximity sensing, follow the below guidelines:

- To reduce the sensor C_P, use a hatch fill of 0.17 mm (7 mil) trace and 1.143 mm (45 mil grid) in the top layer and a hatch fill of 0.17 mm (7 mil) trace and 1.778 mm (70 mil grid) in the bottom layer and drive it with driven shield signal.
- To reduce the effect of floating/grounded conductive object on the proximity-sensing distance, place a hatch fill of 0.17 mm (7 mil) trace and 1.143 mm (45 mil grid) in between the sensor and the conductive object and drive the hatch fill with the driven shield signal.

- To make the proximity sensing uni-directional, place a hatch fill of 0.17 mm (7 mil) trace and 1.143 mm (45 mil grid) in between the sensor and the direction in which proximity detection should be avoided and drive the hatch fill with the driven shield signal.

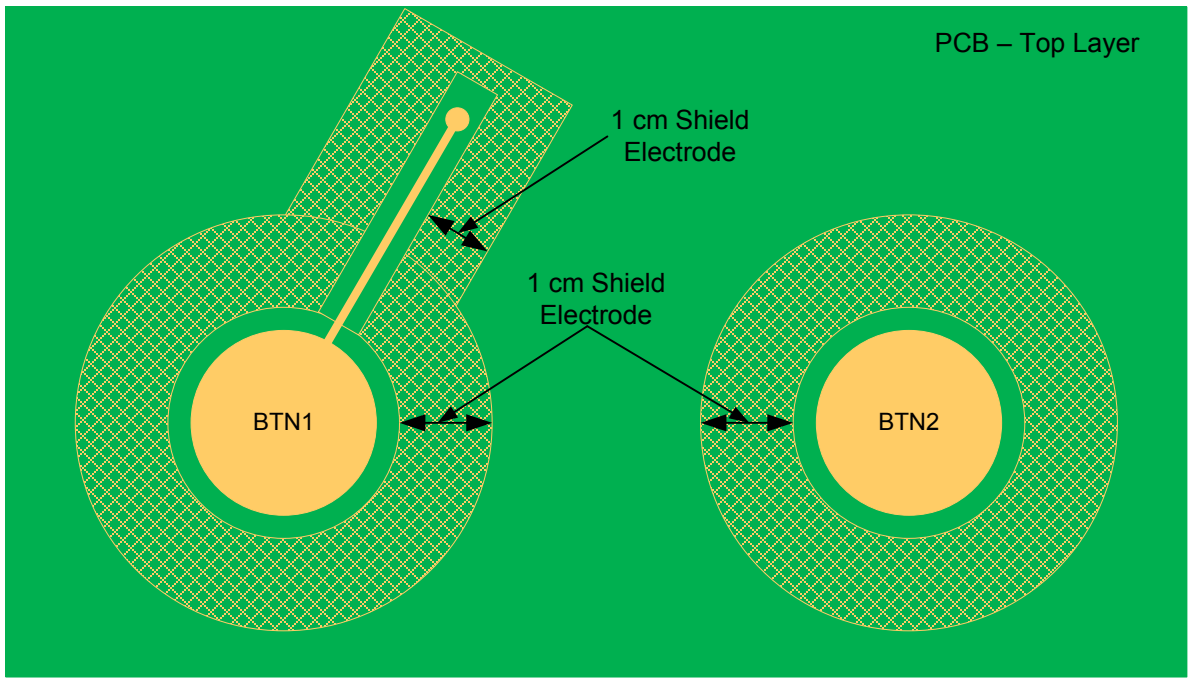
3.8.13.2 Shield Electrode Construction for Liquid Tolerance

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

The shield electrode area depends on the size of the liquid droplet and the area available on the board for implementing the shield electrode. The shield electrode should surround the sensor pads and traces, and spread no further than 1 cm from these features. Spreading the shield electrode beyond 1 cm has negligible effect on system performance. Also, having a large shield electrode might increase the radiated emissions. If the board area is very large, the area outside the 1-cm shield electrode should be left empty, as Figure 3-74 shows. For improved liquid tolerance performance there should not be any hatch fill or a trace connected to ground in the top and bottom layer of PCB. When there is a grounded hatch fill or a trace then, when a liquid droplet falls on the touch interface, it might cause sensor false triggers. Even if there is a shield electrode in between the sensor and ground, the effect of shield electrode will be totally masked out and sensors might false trigger.

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

Figure 3-74. Shield Electrode Placement when Sensor Trace is routed in Top and Bottom Layer



Follow the below guidelines implementing the shield electrode in a 2-Layer and 4-layer PCB:

2-Layer PCB:

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

4-Layer PCB:

- Top layer: Hatch fill with 7-mil trace and 45-mil grid (25 percent fill). Hatch fill should be connected to driven shield signal.

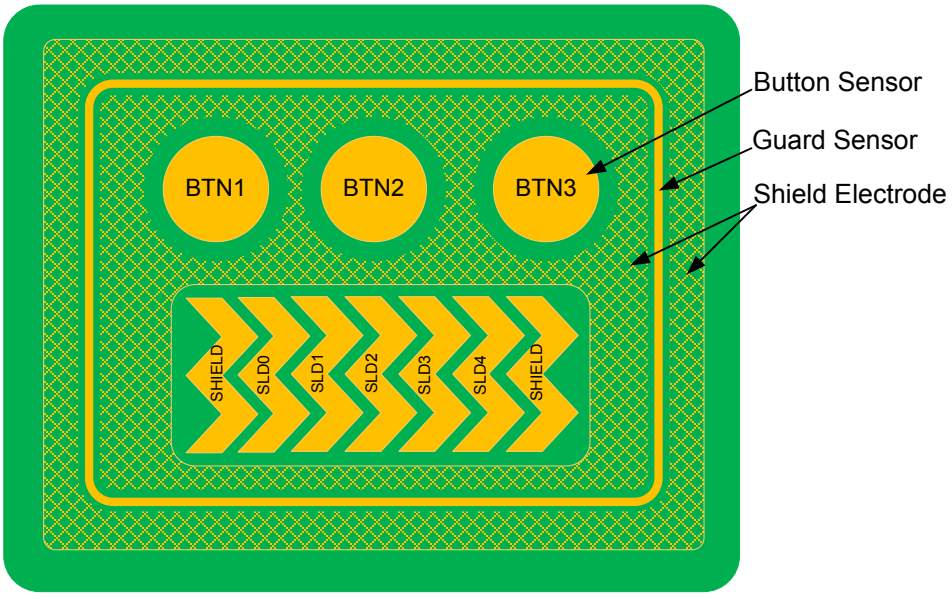
- Layer-2: Hatch fill with 7-mil trace and 70-mil grid (17 percent fill). Hatch fill should be connected to driven shield signal.
- Layer-3: V_{DD} Plane
- Bottom layer: Hatch fill with 7-mil trace and 70-mil grid (17 percent fill). Hatch fill should be connected to ground.

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

3.8.13.3 Guard Sensor

As explained in the Liquid Tolerance section, the guard sensor is a copper trace that surrounds all of the sensors, as Figure 3-75 shows.

Figure 3-75. PCB Layout with Shield Electrode and Guard Sensor



The guard sensor layout should be placed such that:

- It should be the first sensor to turn ON when there is a liquid stream on the touch surface. To accomplish this, the guard sensor surrounds all the sensors in a CapSense system as Figure 3-75 shows.
- It should not be triggered while pressing a button or slider sensor. Otherwise, the button sensors and slider sensor scanning will be disabled and the CapSense system become non operational until the guard sensor is turned OFF. To ensure the guard sensor is not accidentally triggered, place the guard sensor at a distance greater than 1 cm from the sensors.

Follow the below guidelines for implementing the guard sensor:

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

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

The following conditions can be used to detect a liquid stream on the touch surface:

- When there is a liquid stream, more than one button sensor will be active at a time. If your design does not require multi-touch sensing, you can detect this and reject the sensor status of all the button sensors to prevent false triggering.
- In a slider, if the slider segments which are turned ON are not adjacent segments, you can reset the slider segments status or reject the calculated slider centroid value.

You can also refer to the individual device design guides webpage for sample layouts of all the CapSense devices:

- CY8C21X34 Design guide
- CY8C20X34 Design guide
- CY8C20XX6A Design guide
- CY8C20XX7/S Design Guide
- CY8CMBR3XXXx CapSense Design Guide

3.8.14 CapSense System Design with Single Layer PCB

Electronic product manufacturers face constant pressure to lower system costs. Several markets, including consumer and home appliances, are switching to single layer PCBs to support their required product margins. Cypress's CapSense controllers provide robust touch sensing on single layer PCBs, and their driven shield capability enables longer trace length, proximity sensing, and liquid tolerance. CapSense delivers IEC (IEC 61000-6-1, IEC 61000-6-2) noise compliant performance for accurate touch responses even in noisy environments using sophisticated firmware algorithms. For more details on implementing CapSense touch sensing on single layer boards, contact capsense@cypress.com.

Refer to Appendix B: Schematic and Layout Checklist before you start your design to ensure that the best practices for a CapSense design are followed.

4. CapSense Product Portfolio

Cypress's CapSense controller solutions are based on the Programmable System-on-Chip (PSoC®) platform and offer a wide range of features.

4.1 Cypress's CapSense Controller Solutions

Cypress is the world leader in capacitive sensing technologies; the broad range of solutions provides robust noise immunity, enable quick time to market, and system scalability. With CapSense controllers, you can:

- Replace mechanical components with simple CapSense buttons and sliders.
- Reduce total BOM cost and form factor by integrating CapSense with other system components.
- Optimize board space with our small form factor packaging (WLCSP, SOIC, and QFN).
- Use our advanced sensing techniques for easy finger detection through 15 mm of glass or 5 mm of plastic.
- Get to market more quickly using SmartSense Auto-Tuning.
- Implement additional user interface functionality, such as LED effects, proximity and liquid tolerance.

Cypress offers a wide range of configurable and programmable CapSense controllers. Configurable CapSense controllers are hardware or I^2C configurable. Programmable devices provide complete flexibility to meet your exact design requirements, including reducing BOM cost by integrating further system functionality. The different CapSense device families are listed in the following sections.

4.1.1 CapSense Express Controllers (Configurable Solutions)

4.1.1.1 CY8CMBR3XXX

CY8CMBR3XXX devices feature configurable I/Os that can be used as capacitive sensing inputs or as GPIOs for driving LEDs, interrupt outputs, and other digital I/O functionalities. These devices support up to sixteen capacitive sensing inputs, two proximity sensors, and two sliders elements. The device configuration is register-based through an I^2C interface.

4.1.1.2 CY8CMBR20XX

The CY8CMBR20XX device features a wide operating voltage range, 1.71 V to 5.5 V operating voltage and CSD capacitive sensing with SmartSense Auto-Tuning. These devices are hardware configurable; no I^2C is required.

4.1.1.3 CY8C201XX

CY8C201XX devices feature configurable I/Os that can be used as capacitive sensing inputs or as GPIOs for driving LEDs, interrupt outputs, wake-up on interrupt inputs and other digital I/O functionalities. These devices support register-based configuration through an I^2C interface.

4.1.2 CapSense Controllers (Programmable Solutions)

4.1.2.1 CY8C20X34, CapSense

CY8C20X34 devices feature CSA_EMC capacitive sensing and can implement up to 25 CapSense buttons.

4.1.2.2 CY8C20X36A, CapSense

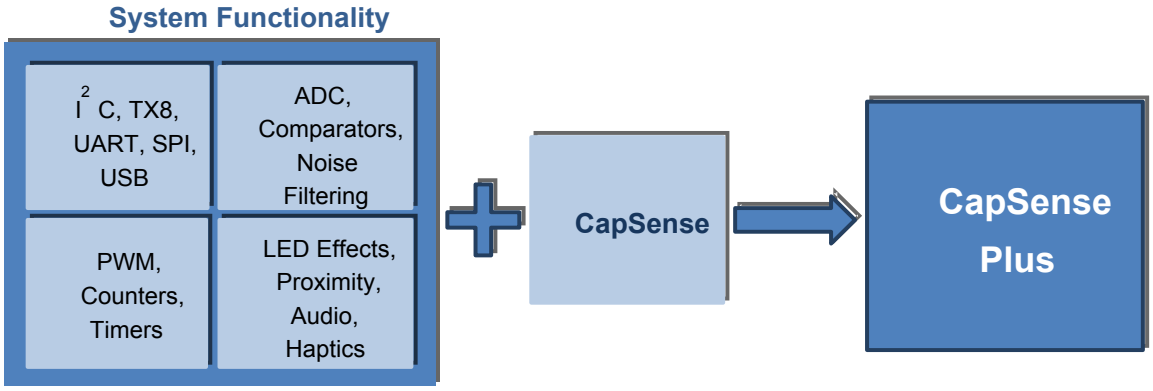
CY8C20XX36A devices can operate at voltages as low as 1.71 V to 5 V; offer both CSA_EMC and CSD capacitive sensing, SmartSense, and support up to 33 CapSense buttons.

4.1.2.3 CY8C20XX7, CapSense

CY8C20XX7 devices can operate at voltages as low as 1.71 V from 5 V; offer both CSD and CSDPLUS (enhanced sensitivity) capacitive sensing with liquid tolerance, SmartSense_EMCplus, and support up to 31 CapSense buttons.

4.1.3 CapSense Plus (Programmable Solutions)

Figure 4-1. CapSense Plus



CapSense Plus devices feature capacitive touch sensing and additional system functionality. Using CapSense Plus devices can result in significant cost savings. Additional features include:

- Feedback - LED, audio, haptics
- Communication - I^2C, TX8, UART, SPI, USB
- Digital functions - PWM, counters, timers
- Analog functions - ADC, comparator
- Bootloaders

4.1.3.1 CY8C20XX7/S, CapSense Plus

CY8C20XX7/S devices can operate at voltages as low as 1.71 V to 5 V, offer both CSD and CSDPLUS (enhanced sensitivity) capacitive sensing, liquid tolerance, proximity detection, SmartSense_EMCplus, and support up to 31 CapSense buttons/proximity sensors, I2C, SPI, and up to 32-KB Flash memory.

4.1.3.2 CY8C20XX6A/AS, CapSense Plus

CY8C20XX6A/AS devices can operate at voltages as low as 1.71 V to 5 V, offer both CSA_EMC, CSD capacitive sensing, SmartSense and SmartSense_EMC, and support up to 33 CapSense buttons, USB, I^2C, SPI, and up to 32-KB Flash memory.

4.1.3.3 CY8C21X34/B, CapSense Plus

CY8C21X34 devices feature CSD capacitive sensing and SmartSense sensing (CY8C21X34B) with advanced digital and analog peripherals, are liquid tolerant, and can support up to 24 CapSense buttons.

4.1.3.4 CY8C24X94, CapSense Plus

CY8C24X94 devices feature CSD capacitive sensing, proximity sensing, are liquid tolerant, support a wide range of interfaces (SPI, I^2C USB 2.0, and UART), and can support up to 44 CapSense buttons.

4.1.3.5 CY8C22X45, CapSense Plus

CY8C22XX45 devices feature CSD capacitive sensing, proximity sensing, are liquid tolerant, support a wide range of interfaces (SPI, I^2C, and UART), have 10-bit SAR ADC, 8 digital blocks and can support up to 37 CapSense buttons.

4.1.4 PSoC with CapSense

4.1.4.1 PSoC 3

PSoC 3 is a true programmable embedded system-on-chip, integrating custom analog and digital peripheral functions such as a 20-bit ADC, opamps, comparators, communication peripherals, programmable digital blocks, memory, and an 8051 microcontroller on a single chip. PSoC 3 supports dual-channel CapSense with as many as 62 sensors.

4.1.4.2 PSoC 4

PSoC 4 is a true programmable embedded system-on-chip, integrating custom analog and digital peripheral functions such as a 12-bit ADC, opamps, comparators, communication peripherals, programmable digital blocks, memory, and an ARM Cortex-M0 microcontroller on a single chip. PSoC 4 supports CapSense with as many as 35 sensors.

4.1.4.3 PSoC 5LP

PSoC 5LP is a true programmable embedded system-on-chip, integrating custom analog and digital peripheral functions such as a 20-bit ADC, opamps, comparators, communication peripherals, programmable digital blocks, memory, and an ARM® Cortex®-M3 microcontroller on a single chip. PSoC 5LP supports dual channel CapSense with as many as 62 sensors.

4.1.4.4 Dynamic Reconfiguration

Dynamic reconfiguration is a clever way to optimize total system cost. There are situations in which the number of digital and analog blocks required by a particular application exceeds the resources of the chip. In these situations, it may be possible to time-share the analog and digital blocks. The process of reusing analog and digital resources at different points in time is called dynamic reconfiguration. If the application requires CapSense, an ADC, and a counter, but not all at the same time, reconfiguring the hardware blocks dynamically will enable all features to be implemented. For more information about dynamic reconfiguration, see the Cypress application note AN2104 – PSoC 1 - Dynamic Reconfiguration with PSoC Designer.

5. CapSense Selector Guide

5.1 Selecting the Right CapSense Device

Several key system requirements must be considered when selecting the best CapSense device for your application.

- Flash and RAM requirements
- Operating voltage range
- Configurable/Programmable
- Number and type of capacitive sensors
- Tuning method
- Liquid tolerance
- Feedback
- Package size and pin count
- Additional features, such as ADC, communication protocol, and timers

Figure 5-1 provides a high-level guide to the CapSense devices based on design requirements. Table 5-1 identifies the key features for each CapSense product family. After completing the selection process, the next step in your development cycle is to consult the more detailed design guide for the selected device. See the Device-Specific Design Guides.

Figure 5-1. Device Selection Tree

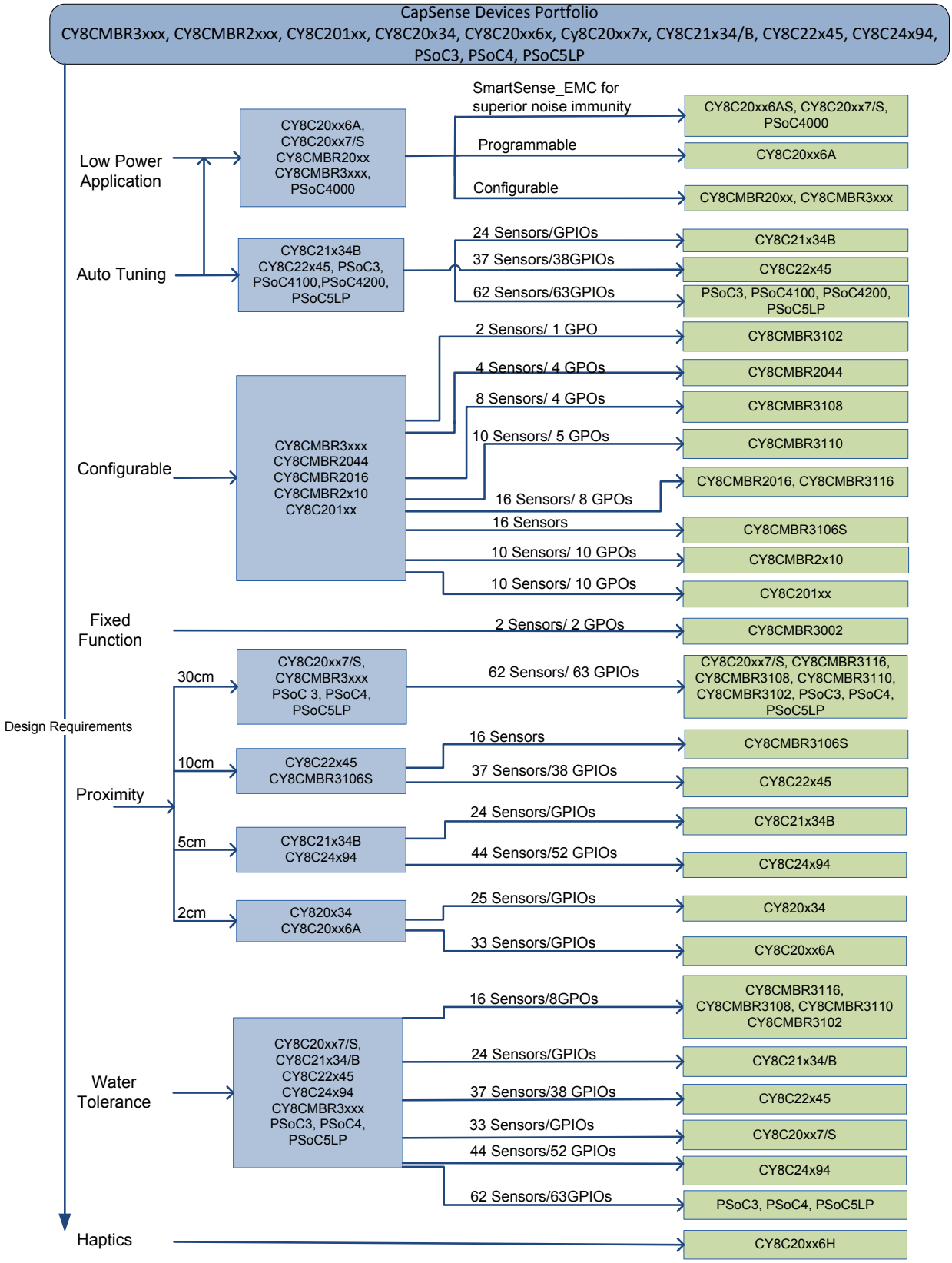


Table 5-1. CapSense Sensing Method by Product Family

		Product Family											
CapSense User Modules		CY8CMBR3XXX	CY8CMBR2X1X	CY8CMBR2044	CY8C201XX	CY8C21X34/B	CY8C20X34	CY8C20XX6A/H	CY8C20XX7	CY8C20XX6AS	CY8C20XX7/S	CY8C24X94	CY8C22X45
	CSD	-	Yes	Yes	-	Yes	-	Yes	Yes	Yes	Yes	Yes	Yes
	CSDPLUS	Yes	-	-	-	-	-	-	Yes	-	Yes	-	-
	CSD2X	-	-	-	-	-	-	-	-	-	-	-	Yes
	CSA_EMC	-	-	-	Yes	-	Yes	Yes	-	Yes	-	-	-
	SmartSense	-	-	Yes	-	Yes (B)	-	Yes	-	Yes	-	-	-
	SmartSense_EMC	-	Yes	-	-	-	-	-	-	Yes	-	-	-
	SmartSense_EMCplus	Yes	-	-	-	-	-	-	-	-	Yes	-	-
	SmartSense2X	-	-	-	-	-	-	-	-	-	-	-	Yes
	SmartSense2X_EMC	-	-	-	-	-	-	-	-	-	-	-	Yes

Table 5-2. CapSense Key Features by Product Family

		Product Family											
Feature		CY8CMBR3XXX	CY8CMBR2X1X	CY8CMBR2044	CY8C201XX	CY8C21X34/B	CY8C20X34	CY8C20XX6A/H	CY8C20XX7	CY8C20XX6AS	CY8C20XX7/S	CY8C24X94	CY8C22X45
	RAM (Bytes)	-	-	-	-	512	512	1 K/2 K	1 K/2 K	1 K/2 K	1 K/2 K	1 K	1K
	Flash (Bytes)	-	-	-	-	8 K	8 K	8 K/16 K/32 K	8 K/16 K/32 K	16 K/32 K	16 K/32 K	16 K	16K
	Operating Voltage	1.71–5.5 V	1.71–5.5 V	1.71 V–5.5 V	2.4 V-5.25 V	2.4 V–5.25 V	2.4 V-5.25 V	1.71 V–5.5 V	1.71 V–5.5 V	1.71 V–5.5 V	1.71 V–5.5 V	3.0 V-5.25 V	3.0 V-5.25 V
	Configurable	Yes	Yes	Yes	Yes	-	-	-	-	-	-	-	-
	Programmable	-	-	-	-	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Maximum Number of Sensors	16	16	4	10	24	25	33	31	33	31	44	37
	Buttons	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Sliders	Yes	-	-	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Proximity	Yes	-	-	-	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	I^2C	HW Slave	HW Slave for CY8CMBR2110 only	-	HW Slave	Master and Slave Interface	Master and Slave Interface	Master and Slave Interface	Master and Slave Interface	Master and Slave Interface	Master and Slave Interface	Master and Slave Interface	Master and Slave Interface

	Product Family											
	CY8CMBR3XXX	CY8CMBR2X1X	CY8CMBR2044	CY8C201XX	CY8C21X34/B	CY8C20X34	CY8C20XX6A/H	CY8C20XX7	CY8C20XX6AS	CY8C20XX7/S	CY8C24X94	CY8C22X45
SPI	-	-	-	-	Master and Slave Interface	Master and Slave Interface	Master and Slave Interface	Master and Slave Interface	Master and Slave Interface	Master and Slave Interface	Master and Slave Interface	Master and Slave Interface
UART	-	-	-	-	UART	Transmitter - Software	Transmitter - Software	Transmitter - Software	Transmitter - Software	Transmitter - Software	UART	UART
USB	-	-	-	-	--	-	Full speed USB	-	-	-	Full speed USB	-
Timer	-	-	-	-	8- to 32-bit timer/counter	13-bit timer	16-bit timer	16-bit timer	16-bit timer	16-bit timer	8- to 32-bit timer/counter	8- to 32-bit timer/counter
PWM	-	-	-	-	8- to 32-bit deadband option	-	-	-	-	-	8- to 32-bit	8- to 32-bit
LED	-	-	-	-	7-segment support	7-segment support	-	-	-	-	7-segment support	-
LCD	-	-	-	-	20x2 controller interface	20x2 controller interface	20x2 controller interface	20x2 controller interface	20x2 controller interface	20x2 controller interface	20x2 controller interface	20x2 controller interface
EEPROM	-	-	-	-	Emulation	Emulation	Emulation	Emulation	Emulation	Emulation	Emulation	Emulation
Liquid	Tolerant	-	-	-	Tolerant			Tolerant		Tolerant	Tolerant	Tolerant
Bootloader	-	-	-	-	I^2C	I^2C	$I^2$2C Full speed USB	I^2C	I^2C	I^2C	I^2C Full speed USB	I^2C
Feedback	-	-	-	-	-	-	Haptics	-	-	-	-	-
Comparators	-	-	-	-	Yes	Yes	Yes	Yes	Yes	Yes	-	Yes
ADC	-	-	-	-	8- and 10-bit	-	8- and 10-bit	8- and 10-bit	8- and 10-bit	-	7- to 13-bit	10-bit
DAC	-	-	-	-	-	-	-	-	-	-	6,8 and 9-bit	8-bit
Random Sequencer	-	-	-	-	-	-	-	-	-	-	8- to 32-bit pseudo	8- to 32-bit pseudo
Amplifiers	-	-	-	-	-	-	-	-	-	-	Yes	-
Filter	-	-	-	-	-	-	-	-	-	-	2-pole band and low pass	-

For details on features supported by PSoC 3, PSoC 4, and PSoC 5LP devices, refer to respective datasheets and design guides:

- Device Family Specific CapSense Device Datasheets
- Device Family Specific CapSense Design Guides

Table 5-3. Comparison Between CSD/SmartSense/SmartSense_EMC and CSA_EMC

Parameter	CSD/SmartSense/SmartSense_EMC	CSDPLUS/SmartSense_EMCplus	CSA_EMC
External Components	1-2	1	0-1
Sensitivity	+++	++++	++
Noise Immunity[1]	+++	++++	++
Spread Spectrum Precharge Clock	Yes	Yes	No
Scanning Time Range	75 µs…23 ms	75 µs…23 ms	90 µs…1 ms
Liquid Proof Operation	++	+++	-
Shield Electrode	+++	++++	-
Buttons, Sliders, Touchpad's	+++	+++	+++
Proximity	+++	++++	++
Linear Transfer Characteristic	Yes	Yes	No
Power Consumption	++	++	+++
Sleep Support	+++	++++	+++
Environment Changes Immunity	+++	+++	++

[1] CSD, SmartSense, and SmartSense_EMC have better immunity to external noise than CSA_EMC. However, with CSA_EMC, immunity against noise can be improved by selecting more number of scan frequencies.

6. CapSense Migration Paths

As a system evolves, it may require a CapSense controller with more advanced features. The following sections describe possible migrations among CapSense controllers.

6.1 CY8C20XX6/H/AS to CY8C20XX7/S

CY8C20XX7 devices implement CSD, CSDPLUS, and SmartSense_EMC plus using improved sensing hardware over CY8C20XX6A/H/AS or CY8C20XX7/S. Therefore, upgrading your design from CY8C20XX6A/H/AS or CY8C20XX7/S to CY8C20XX7/S does not require any change in external hardware, but may require some minor firmware changes. Simply clone your PSoC Designer project to CY8C20XX7/S using the **New Project > Clone** option found the **File** menu in PSoC Designer. For more information on cloning a project please read the CapSense Controller Code Examples.

CY8C20XX7/S has the following advantages over CY8C20XX6A/H/AS or CY8C20XX7/S:

- Greater touch sensitivity (that is, you will get a larger touch signal from an equivalent sensor in CY8C20XX7, thus increasing the system SNR)
- Lower system noise
- Lower power consumption
- Improved proximity detection
- Liquid-tolerant sensing capability

6.2 CY8C20X34 to CY8C20XX6A/H/AS or CY8C20XX7/S

CY8C20XX6A/H/AS or CY8C20XX7/S devices implement CSA_EMC using dedicated hardware similar to that in CY8C20X34. Therefore, upgrading your design from CY8C20X34 to CY8C20XX6A/AS does not require any external hardware or firmware changes; simply clone your PSoC Designer project to CY8C20XX6A/AS.

Advantages of CY8C20XX6A/H/AS or CY8C20XX7/S over CY8C20X34:

- More I/O pins
- Wider operating voltage range
- Lower power consumption
- Immersion TS2000 Haptics Technology for ERM control (CY8C20XX6A/H/AS only)

6.3 CY8C20XX6A/H/AS or CY8C20XX7/S to CY8C21X34/B / CY8C24X94

All four devices feature CSD capacitive sensing. CY8C20XX6A/H/AS and CY8C20XX7/S devices implement CSD using dedicated hardware as opposed to programmable digital and analog blocks as in CY8C21X34/B and CY8C24X94 devices. Furthermore, CY8C21X34/B and CY8C24X94 devices use an external bleed resistor in the current measuring circuit, whereas CY8C20XX6A/H/AS and CY8C20XX7/S devices use an on-chip iDAC. These architectural differences necessitate different external circuitry, user module parameters, and settings for similarly named parameters and APIs between the two devices. If the USB module is used in CY8C20XX6A, then migration is not possible to CY8C21X34/B.

CY8C21X34/B and CY8C24X94 provide the following advantages over CY8C20XX6A/H/AS and CY8C20XX7/S: Proximity up to 5 cm and liquid tolerance.

6.4 Pin-to-Pin Compatibility

When migrating from one device to another, PCB change is not required if you assign the same pin functionality to the new device. If this is the case, then the two devices are pin-to-pin compatible. Table 6-1 gives information about available device packages and pin-to-pin compatibility between the same packages of CY8C20X34, CY8C21X34/B, CY8C20XX6A/AS, and CY8C24X94 devices.

Table 6-1. Pin-to-Pin Compatibility

Package	Device					
	CY8C20X34	CY8C21X34/B	CY8C20XX6A/H/AS or CY8C20XX7/S	CY8C20XX7/S	CY8C24X94	CY8C22X45
16 QFN	N[a]	-	N[a]	Y (CY8C20XX6A/H/AS)	-	-
16 SOIC	N	N	-	N	-	-
24 QFN without USB	Y	-	Y	N[c]	-	-
28 SSOP	Y	Y	-	-	-	-
30 WLCSP	N	-	N	Y (CY8C20XX6A/H/AS)	-	-
32 QFN without SMP	Y	Y	Y[b]	N[d]	-	-

a. Pin 2 is different: CY8C20X34 → P2[1], CY8C21X34/B → P2[3]. If Pin 2 is used, modify the firmware for migration.
b. Without USB.
c. One pin - Pin 23 is V_{SS} in 20XX7/S while it is an I/O in 20XX6A/AS.
d. Two Pins - Pin 28 is I/O and Pin 29 is V_{SS} in 20XX7/S while in 20XX6A/AS pin 28 is V_{SS} and Pin 29 is an I/O.

Symbols:

Y pin-to-pin compatible (same package dimensions)

N packages not pin-to-pin compatible

- package not available

For more details, see the "Pin Information" section of the respective datasheets.

7. Resources

7.1 Website

At the Cypress CapSense Controllers website, you can access all the reference material discussed in this section as well as device specific datasheets and design guides.

7.2 Device-Specific Design Guides

Design guides are available for each CapSense family of devices. These documents are intended for design engineers who are familiar with capacitive sensing technology and have selected a family of devices.

- CY8C20X34 CapSense® Design Guide
- CY8C20XX6A/H CapSense® Design Guide
- CY8C20XX7/S Design Guide
- CY8C21X34/B CapSense® Design Guide
- CapSense® Express™: CY8C201xxx Application Notes
- CY8CMBR2044 CapSense® Design Guide
- CY8CMBR2010 CapSense® Design Guide
- CY8CMBR2110 CapSense® Design Guide
- CY8CMBR2016 CapSense® Design Guide
- CY8CMBR3XXX CapSense® Design Guide

Design Guides for PSoC with CapSense

- PSoC® 3 and PSoC® 5LP CapSense® Design Guide
- PSoC® 4 CapSense® Design Guide

In addition to these device specific design guides, our CapSense Code Examples Design Guide is available to help you design and evaluate CapSense applications quickly using our Development Kits.

7.3 Technical Reference Manuals

Cypress has created the following technical reference manuals to provide quick and easy access to information on CapSense controller functionality including top-level architectural diagrams, register summaries, and timing diagrams.

- CY8C201XX: Register Reference Guide
- PSoC® CY8C20X66, CY8C20X66A, CY8C20X46/96, CY8C20X46A/96A, CY8C20X36, CY8C20X36A Technical Reference Manual (TRM)
- PSoC® CY8C20XX7/S Technical Reference Manual (TRM)
- CY8CPLC20, CY8CLED16P01, CY8C29X66, CY8C27X43, CY8C24X94, CY8C24X23, CY8C24X23A, CY8C22X13, CY8C21X34, CY8C21X23, CY7C64215, CY7C603XX, CY8CNP1XX, and CYWUSB6953 PSoC® Programmable System-on-Chip TRM

- PSoC® CY8C20X24, CY8C20X34 Technical Reference Manual (TRM)
- CY8CMBR3xxx Technical Reference Manual

Technical Reference Manuals for PSoC with CapSense:

- PSoC3 Technical Reference Manual
- PSoC4 Technical Reference Manual
- PSoC5 Technical Reference Manual

7.4 Development Kits

7.4.1 Universal CapSense Controller Kits

Universal CapSense Controller Kits feature predefined control circuitry and plug-in hardware to make prototyping and debugging easy. Programming and I^2C-to-USB Bridge hardware are included for tuning and data acquisition.

- CY3280-20X34 Universal CapSense Controller
- CY3280-21X34 Universal CapSense Controller
- CY3280-20XX6A Universal CapSense Controller

Note CY3280-20X34 Universal CapSense Controller and CY3280-21X34 Universal CapSense Controller kits are available only as a part of the CY3280-BK1 Universal CapSense Controller - Basic Kit 1.

7.4.2 Universal CapSense Module Boards

7.4.2.1 Simple Button Module Board

The CY3280-BSM Simple Button Module consists of ten CapSense buttons and ten LEDs. This module connects to any CY3280 Universal CapSense Controller Board.

7.4.2.2 Matrix Button Module Board

The CY3280-BMM Matrix Button Module consists of eight LEDs as well as eight CapSense sensors organized in a 4x4 matrix format to form 16 physical buttons. This module connects to any CY3280 Universal CapSense Controller Board.

7.4.2.3 Linear Slider Module Board

The CY3280-SLM Linear Slider Module consists of five CapSense buttons, one linear slider (with ten sensors), and five LEDs. This module connects to any CY3280 Universal CapSense Controller Board.

7.4.2.4 Radial Slider Module Board

The CY3280-SRM Radial Slider Module consists of four CapSense buttons, one radial slider (with ten sensors), and four LEDs. This module connects to any CY3280 Universal CapSense Controller Board.

7.4.2.5 Universal CapSense Prototyping Module

The CY3280-BBM Universal CapSense Prototyping Module provides access to every signal routed to the 44-pin connector on the attached controller board(s). The prototyping module board is used in conjunction with a Universal CapSense Controller board to implement additional functionality that is not part of the other single-purpose Universal CapSense Module boards.

7.4.3 CapSense Express Evaluation Kits for CY8C201XX

CY8C3218-CAPEXP series of CapSense Express evaluation kits available for CY8C201XX family of devices enables designers to replace mechanical buttons/sliders by implementing touch sensing designs with the CapSense touch sensing family, CapSense Express. With Cypress's PSoC Designer visual embedded system design tool and CapSense Express configuration tool, designers configure, monitor, and tune buttons or sliders, LEDs, and other general purpose I/Os over I^2C in real time using a graphical user interface. The following are the lists of CapSense Express evaluation kits available for CY8C201XX family of devices.

- CY3218-CAPEXP1 CapSense Express Kit (Up to 10 I/O for Buttons) with CY8C20110 device

- CY3218-CAPEXP2 CapSense Express Kit (Up to 10 I/O for Sliders) with CY8C201A0 device

7.4.4 CapSense Express Evaluation Kits for CY8CMBR2044

CY3280-MBR CapSense Express kit for CY8CMBR2044 device enables designers to replace mechanical buttons with CapSense buttons. Designers can configure up to four CapSense buttons in hardware, eliminating the need for software tools, firmware development, and chip programming.

7.4.5 CapSense Evaluation Kits for CY8CMBR3XXX

CY3280-MBR3 Evaluation Kit for CY8CMBR3XXX devices is designed to showcase the abilities of the CY8CMBR3116 register-configurable CapSense® Express™ controller with an ARM® Cortex™-M0 CPU. It is also designed as an Arduino-compatible shield that supports various Arduino baseboards as well as other stackable shields available in the market. The CY3280-MBR3 kit features four CapSense buttons, one proximity sensor loop, LEDs, and an onboard USB-to-I^2C bridge to communicate to the EZ-Click™ 2.0 customizer tool that configures the CY8CMBR3XXX controller.

7.4.6 Evaluation Kit for Proximity

The CY8CKIT-024 - CapSense Proximity Shield enables customers to evaluate and develop CapSense proximity applications. The shield has been designed to work with any of the Cypress Pioneer development platforms.

The CY8CKIT-024 supports following proximity features:

- High proximity range: The kit has a proximity loop, which when interfaced with CY8CKIT - 040, provides a proximity sensing distance of approximately 10 cm
- Horizontal and vertical hover gestures: The kit includes four 5.5 cm-long-proximity sensors for detecting horizontal and vertical gestures
- Liquid tolerance: The kit includes a slide switch to select between ground and driven shield for the shield loop on the top layer. Driven shield along with the recommended CapSense configuration provides liquid tolerant proximity sensing. Use the dropper provided along with the kit to evaluate liquid tolerant performance of the kit.

7.4.7 Evaluation Pods

PSoC EvalPods are pods that connect to the ICE In-Circuit Emulator (CY3215-DK kit) to allow debugging capability. They can also function as a standalone device without debugging capability. The EvalPod has a 28-pin DIP footprint on the bottom for easy connection to development kits or other hardware. The top of the EvalPod has prototyping headers for easy connection to the devices pins. The following are the evaluation pods available.

- CY3210-CY8C20X34 PSoC Evaluation Pod (EvalPod)
- CY3210-CY8C21X34 PSoC Evaluation Pod (EvalPod)
- CY3210-CY8C20X36/46/66 PSoC Evaluation Pod (EvalPod)
- CY3210-CY8C24X94 PSoC Evaluation Pod (EvalPod)

7.4.8 In-Circuit Emulation (ICE) Kits

The ICE pod provides the interconnection between the CY3215-DK In-Circuit Emulator via a flex cable and the target PSoC device in a prototype system or PCB via package-specific pod feet. The kit guide and quick start guide for the In-Circuit Emulator (ICE) Development Kit are available here. Following are the pods available.

- CY3250-21X34QFN ICE Pod Kit to debug QFN CY8C21X34 PSoC devices
- CY3250-24X94QFN ICE Pod Kit to debug QFN CY8C24X94 PSoC devices
- CY3250-20246QFN ICE Pod Kit to debug CY8C20236/46A/46AS PSoC devices
- CY3250-20346QFN ICE Pod Kit to debug CY8C20336/346A/346AS CapSense PSoC devices
- CY3250-20666QFN ICE Pod Kit to debug CY8C20636/646/666A/646AS/666AS CapSense PSoC devices
- CY3250-20566 ICE Pod Kit to debug CY8C20536/546/566A CapSense PSoC devices
- CY3250-20466QFN ICE Pod Kit to debug CY8C20436/46/66/46AS/66AS CapSense PSoC devices
- CY3250-20334QFN ICE Pods (2) to debug QFN CY8C20334 PSoC devices

Order replacement ICE pods here.

7.4.9 PSoC 3 and PSoC 5LP Development Kits

The following PSoC 3 and PSoC 5LP development kits support CapSense functionality:

- CY8CKIT-001 PSoC® Development Kit
- CY8CKIT-030 PSoC® 3 Development Kit
- CY8CKIT-050B PSoC® 5LP Development Kit

7.4.10 PSoC CapSense Expansion Board Kit

The CY8CKIT-031 PSoC CapSense Expansion Board Kit connects any of the PSoC 3 and PSoC 5LP Development Kits to any of the Universal CapSense Module Boards. The CY8CKIT-031 kit provides and interface board and two module boards, CY3280-SLM and CY3280-BMM.

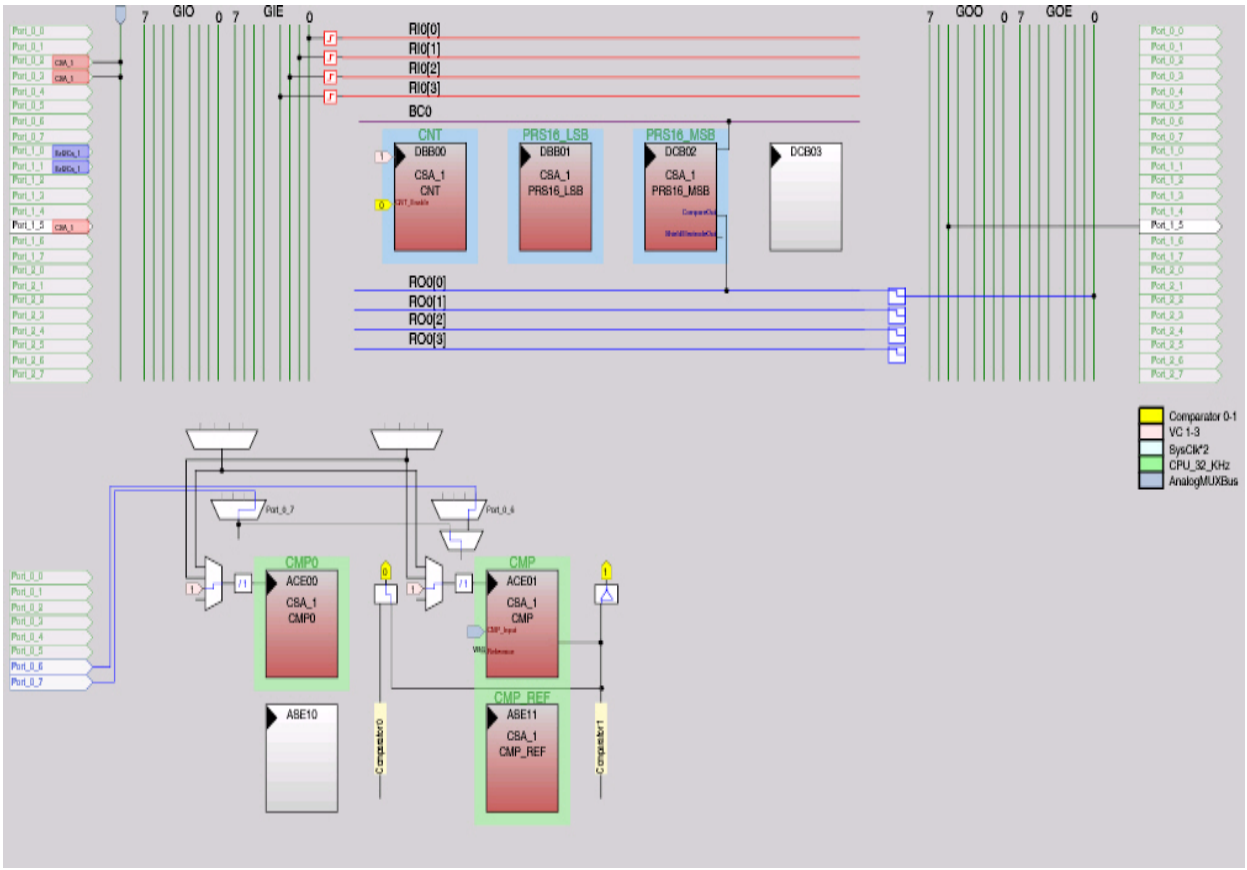
7.4.11 PSoC 4 Development Kits

- PSoC 4 Pioneer Kit (CY8CKIT-042) can be used to evaluate a 5-segment linear slider
- CY8C4000 Family Kit (CY8CKIT-40) can be used to evaluate 2-button, 5-segment linear slider. 6x5 touchpad, and wire proximity sensors.
- CapSense Expansion Board Kit (CY8CKIT-031) can be used with CY8CKIT-038 and CY8CKIT-001 and has a 10-segment slider, five button sensors and a 4x4 matrix buttons with LED indications.
- MiniProg3 Program and Debug Kit (CY8CKIT-002) can be used for CapSense performance tuning in CY8CKIT-038.

7.5 PSoC Designer

Cypress offers an exclusive Integrated Design Environment, PSoC Designer. With PSoC Designer, you can configure analog and digital blocks, develop firmware, and tune your design. Applications are developed in a drag-and-drop design environment using a library of pre-characterized analog and digital functions, including CapSense. PSoC Designer comes with a built-in C compiler and an embedded programmer. A pro compiler is available for complex designs.

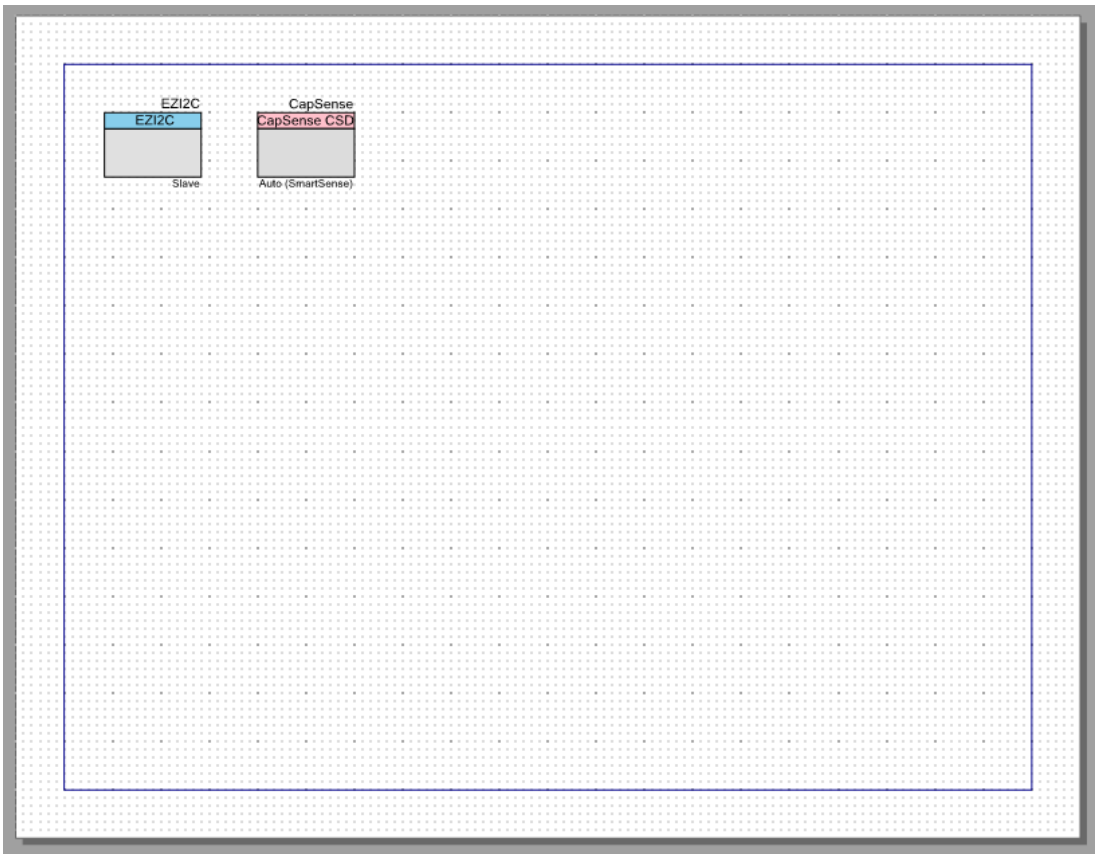
Figure 7-1. PSoC Designer Device Editor



7.6 PSoC Creator

PSoC Creator is an Integrated Design Environment (IDE) which allows concurrent hardware and application firmware design of PSoC 3, PSoC 4, and PSoC 5LP systems. PSoC systems are designed using classic, familiar schematic capture supported by over 120 pre-verified, production-ready PSoC Components™.

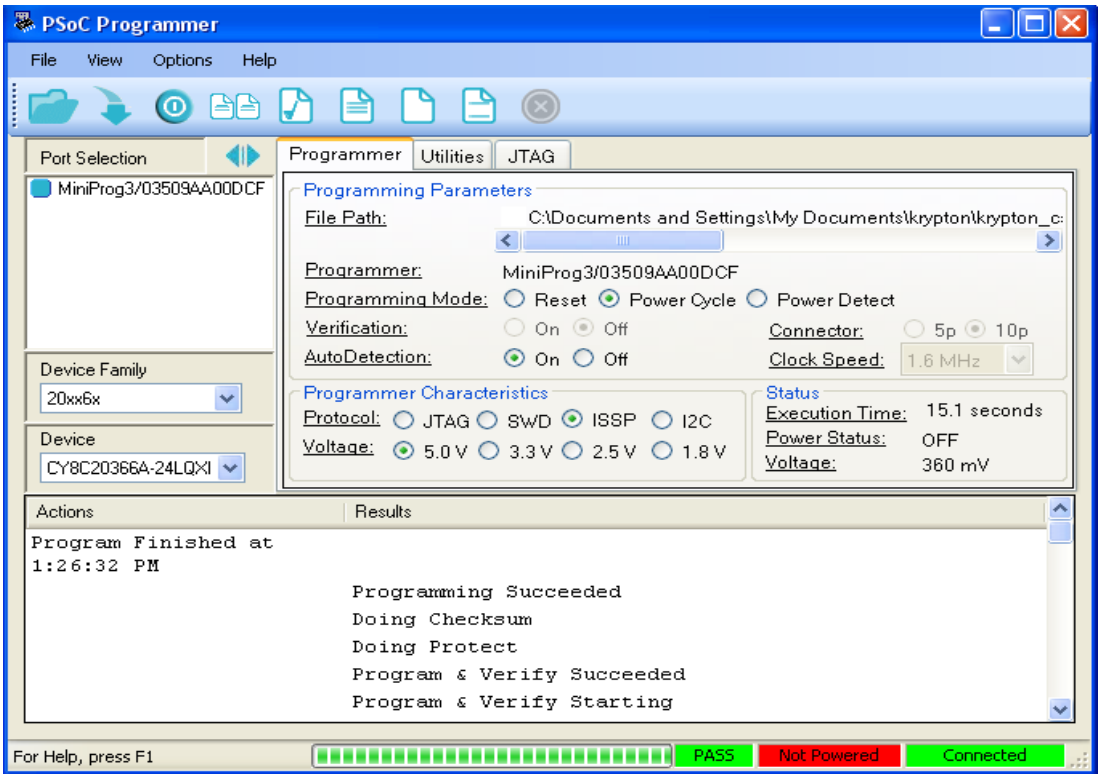
Figure 7-2: PSoC Creator Top Schematic View



7.7 PSoC Programmer

PSoC Programmer is a flexible, integrated programming application to program PSoC devices. PSoC Programmer can be used with PSoC Designer and PSoC Creator to program any design on to a PSoC device.

Figure 7-3. PSoC Programmer



PSoC Programmer provides you a hardware layer with APIs to design specific applications using the programmers and bridge devices. The PSoC Programmer hardware layer is explained in the COM guide documentation as well as example code across the following languages: C#, C, Perl, and Python.

7.8 I²C-to-USB Bridge Kit

The CY3280-I2USB kit allows you to test, tune, and debug hardware and software by bridging the USB port to I²C. Populated with the CY8C24894 PSoC device, a wide variety of devices can be connected to the PC using this bridge including:

- EEPROMs
- Real-time clocks
- ADC/DAC converters
- LCD displays
- Regulated DC/DC converters
- Port expanders
- Other devices incorporating the I²C interface

The number of devices that can be connected is constrained only by the I²C address limit and physical ability of the I²C bus.

7.9 Debugging/Data Viewing Tools

Software tools are available for data viewing, debugging, and tuning CapSense applications. These tools can help you monitor critical data, such as raw counts and CapSense parameters.

The debugging and data viewing tools are:

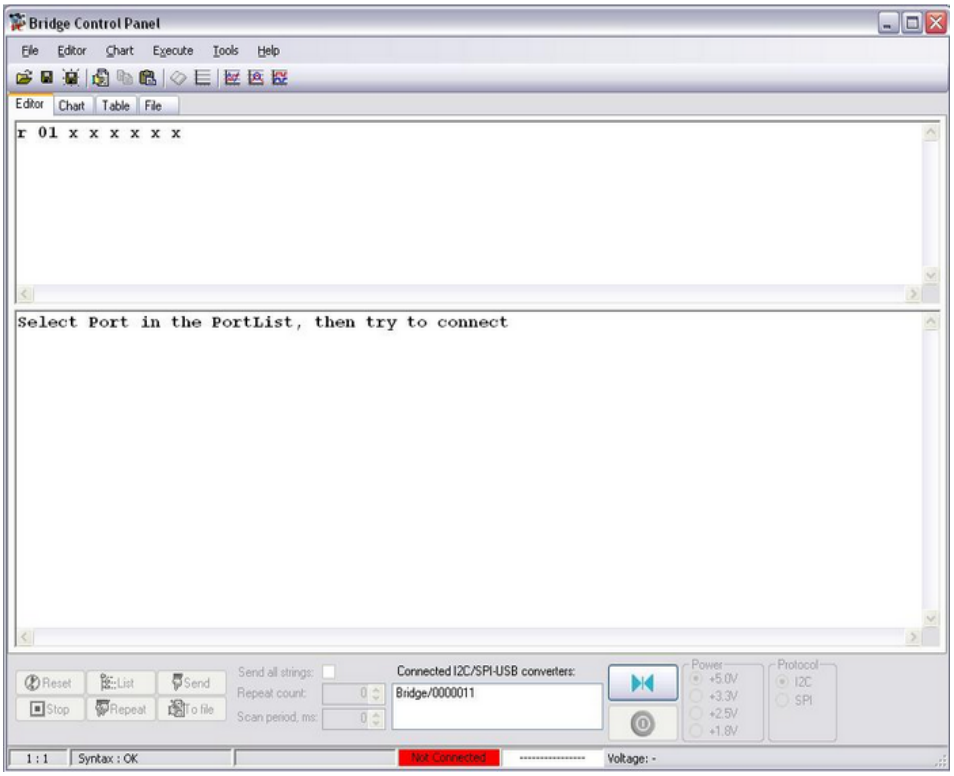
- Bridge Control Panel
- MultiChart

For more details on the tools, see the application note AN2397 – CapSense Data Viewing Tools.

7.10 Bridge Control Panel

Bridge Control Panel is a software tool used with CY3240 USB-I^{2}C Bridge to enable communication with I^{2}C slave devices. The software tool is used to configure I^{2}C devices as well as acquire and process data received from I^{2}C slave devices. The Bridge Control Panel helps in optimizing, debugging, and calibrating the target applications.

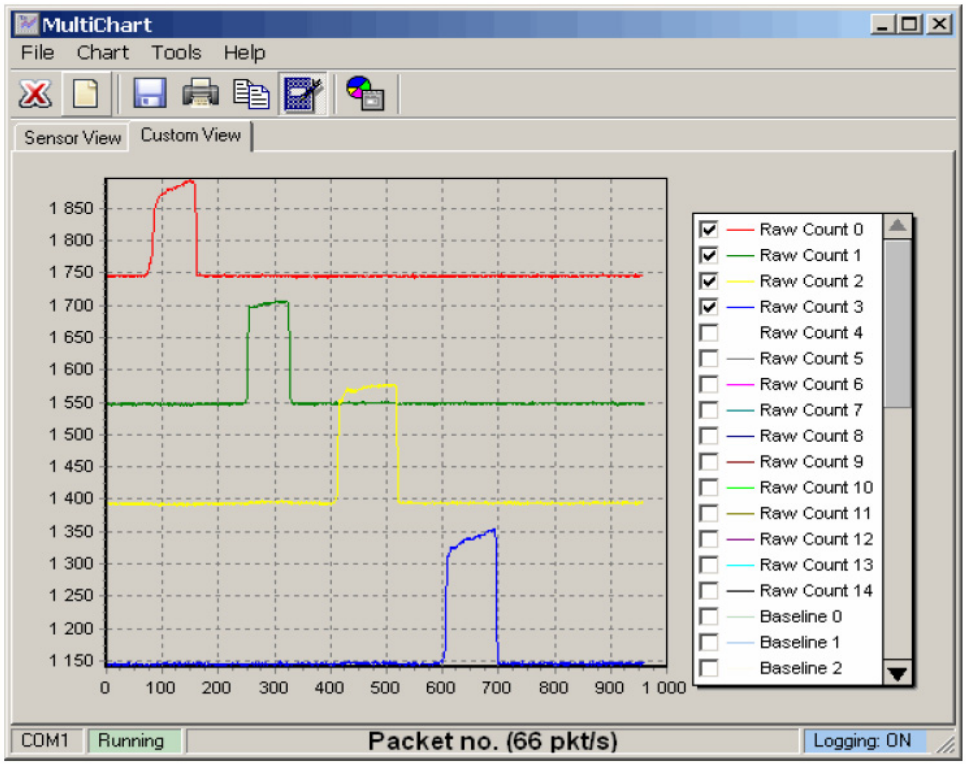
Figure 7-4. Bridge Control Panel



7.10.1 MultiChart

MultiChart is a simple PC tool for real-time CapSense data viewing and logging. The application allows you to view data from up to 48 sensors, save and print charts, and save data for later analysis in a spreadsheet.

Figure 7-5. MultiChart User Interface



7.11 Design Support

Cypress has several support channels to ensure the success of your CapSense design.

- Code Examples – Our vast collection of code examples will help get your design up and running fast.
- CapSense® Controller Code Examples Design Guide - Our 10 most frequently asked CapSense "How to" code examples.
- Knowledge Base Articles – Browse technical articles by product family or perform a search on various CapSense topics.
- White Papers and Technical Articles – Learn about advanced capacitive touch interface topics.
- Cypress Developer Community – Connect with the Cypress technical community and exchange information.
- Video Library – Get up to speed with tutorial videos.
- Quality and Reliability – Cypress is committed to complete customer satisfaction. At our Quality website, you can find reliability and product qualification reports.
- Cypress Design Partner Program – An expansion of our engineering capabilities providing customers with access to design services and solutions from trusted and capable partners.
- Cypress Developer Community – A very active online community to discuss technical issues.
- Technical Support – Excellent technical support is available online.

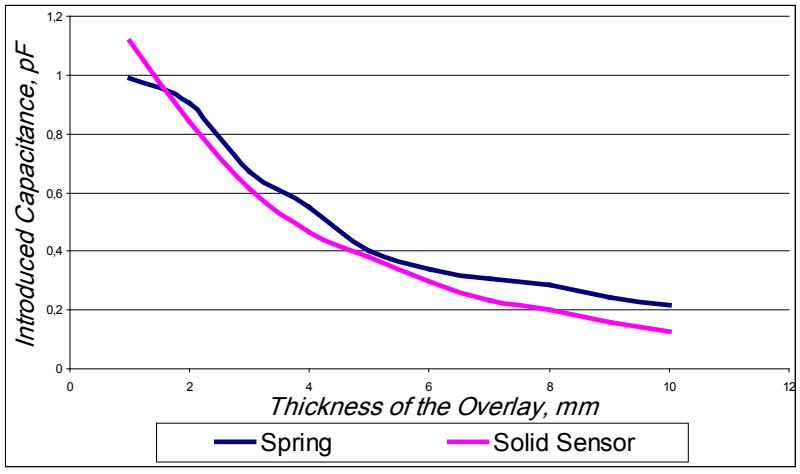
8. Appendix A: Springs

8.1 Finger-Introduced Capacitance

This section gives the influence of various physical parameters on finger-introduced capacitance in a CapSense design with springs.

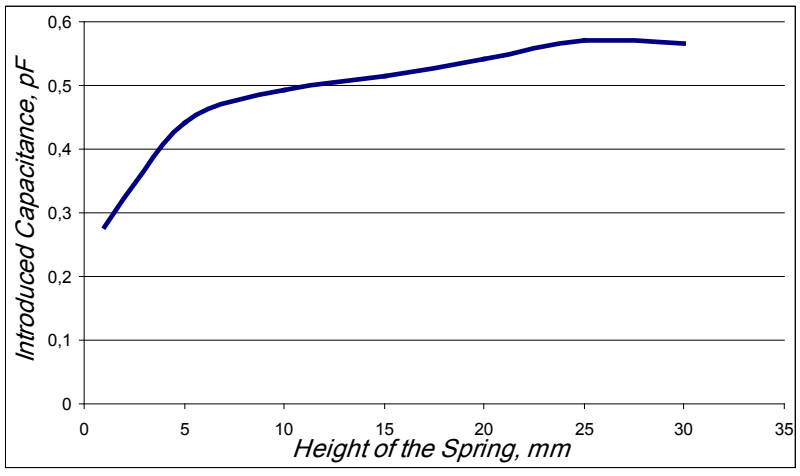
- Influence of overlay thickness on Finger Touch added Capacitance (FTC) with springs is similar to that with solid sensors

Figure 8-1. FTC Versus Overlay Thickness



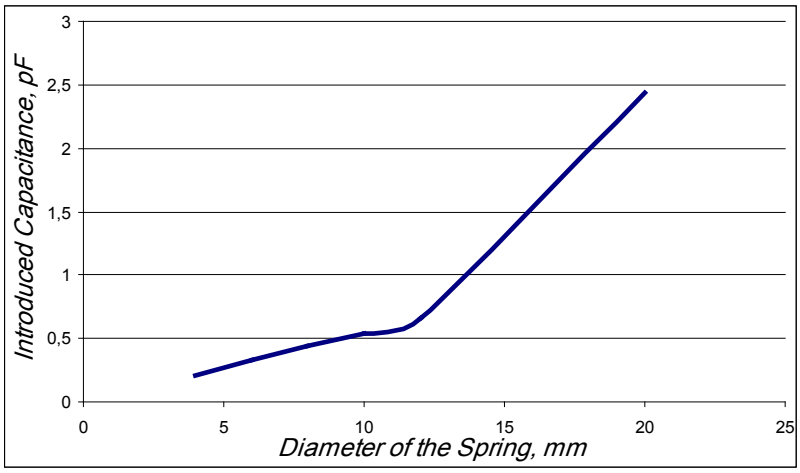
- Influence of height on FTC

Figure 8-2. FTC Versus Spring Height



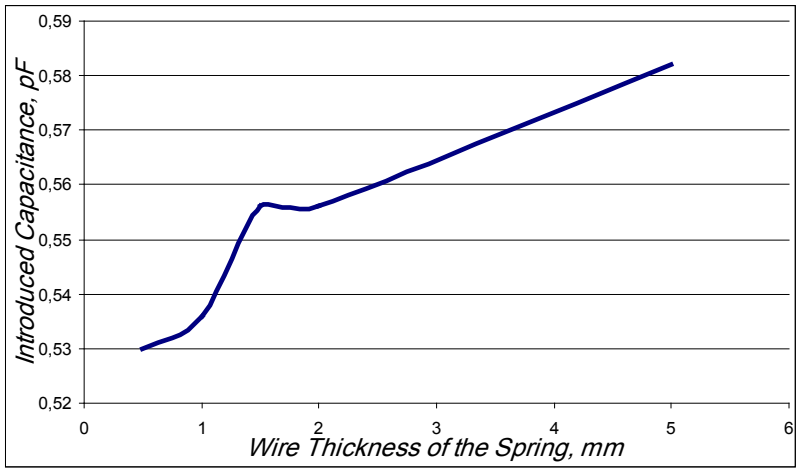
- Influence of diameter on FTC

Figure 8-3. FTC Versus Spring Diameter



- Influence of wire thickness of the spring on FTC

Figure 8-4. FTC Versus Wire Thickness of Spring



8.1.1 Mounting Springs to the PCB

Figure 8-5 shows an example of spring mounting. This section discusses how to design spring sensors. Because springs have higher side sensitivity, the neighboring spring sensors must be placed as far as possible from each other to prevent false detections. Add a comparison level if the sensor pitch is small.

The requirements for the sensitive area of a spring are the same as the requirements for solid buttons. When using thick overlays, the spring diameter must be larger than the overlay thickness by at least 2 or 3 times. The distance between the PCB and the overlay must be 5 mm or more.

Figure 8-5. Spring-Mounting Example

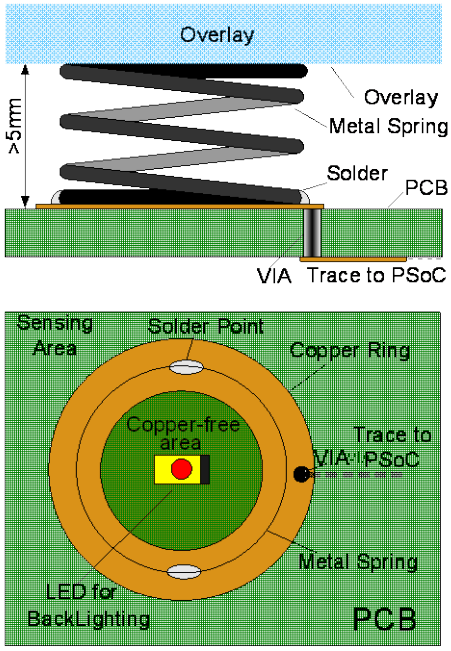
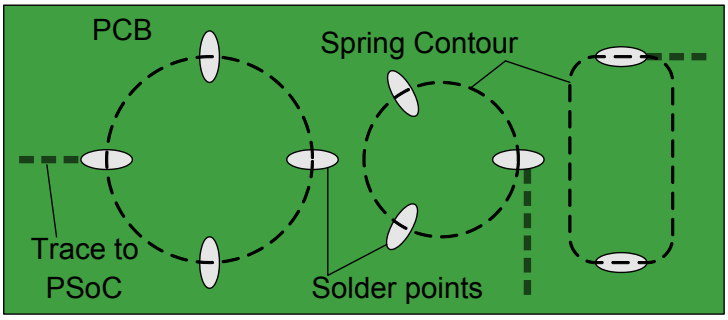


Figure 8-6 shows examples of footprints for springs. Do not place solid grounds under the springs as this complicates the spring soldering and increases the native capacitance of the sensors.

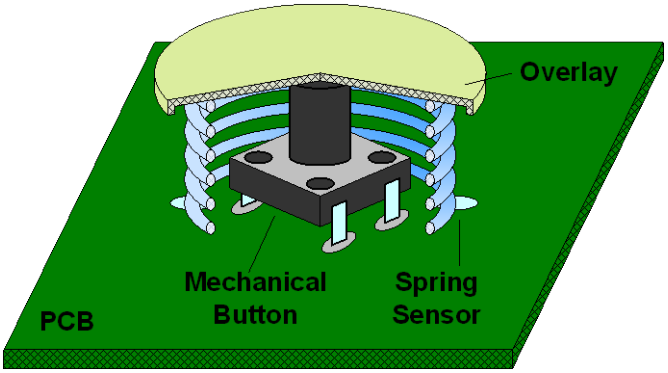
Figure 8-6. Proposed Spring Footprints



8.2 CapSense and Mechanical Button Combination

The hollow space inside a spring can also be used as a mechanical button, as shown in Figure 8-7.

Figure 8-7. CapSense and Mechanical Button Combination



Touching such a button only triggers the sensor, while pressing the button activates both the sensor and mechanical button. In this case, preparatory actions such as backlighting, prompt showing, and others are possible only if the sensor works. The final action is performed when both buttons work. For example, in a GPS navigation system, touching a button shows only a hint and pressing the button takes an action.

8.3 Design Examples

Figure 8-8 and Figure 8-9 show project demonstrator examples for white goods applications.

Figure 8-8. Demo Cooktop

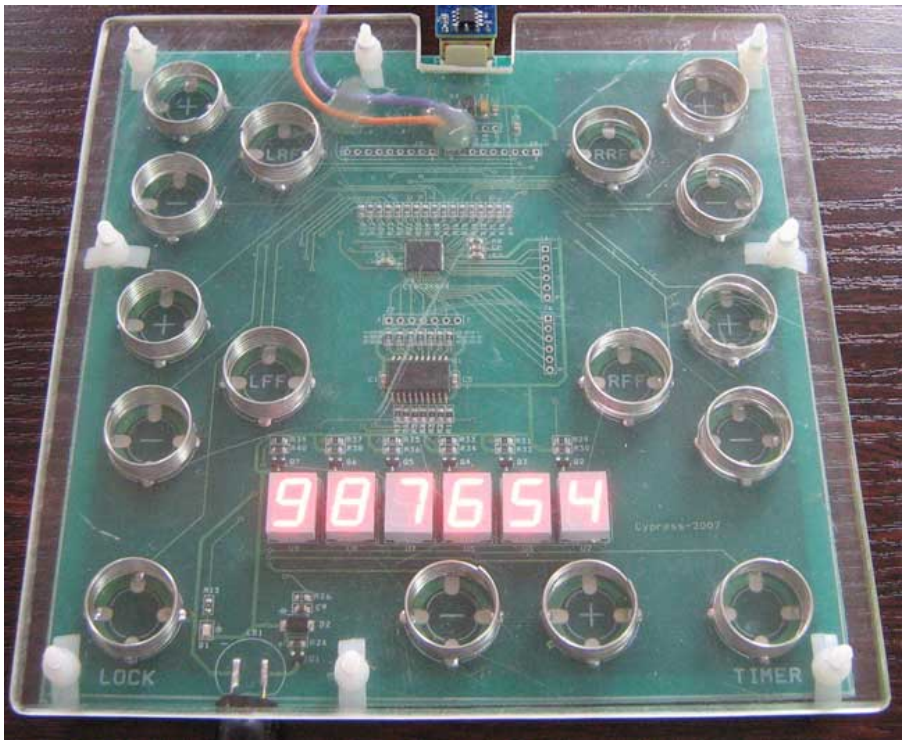


Figure 8-9. Cooktop Front Panel

9. Appendix B: Schematic and Layout Checklist

9.1 Schematic Checklist

For PSoC 3 devices refer PSoC 3 datasheet, for PSoC 4 device refer PSoC 4 datasheet and for PSoC 5LP device refer PSoC 5LP datasheet.

No.	Category	Recommendations/Remarks
1	Decoupling capacitor	0.1 µF
2	Bulk capacitor	1 uF
3	Pin Assignment	Sensors should be placed near to ground. No switching signal should be placed near to sensors
4	CMOD	Ensure that CMOD is not adjacent to any switching or communication pins
5	RB	Ensure that RB is not adjacent to any switching or communication pins
6	Series resistor on CapSense Lines	560 Ω
7	Series resistor on Communication lines	330 Ω
8	Pull-up resistor on Communication lines	4.7 kΩ
9	Avoid using programming pins as I2C pins if possible	

9.1.1 Decoupling Capacitor

Refer to Power Supply Layout Recommendations for complete details.

9.1.2 Bulk Capacitor

Refer to Power Supply Layout Recommendations for complete details.

9.1.3 Pin Assignment

- Distribute the LEDs evenly among even and odd pins such as P2[0], P2[2] and P2[1], P2[3]
- Try not to keep LEDs next to CapSense pins.
- Reserve pins for R_B (if required), C_{MOD} and Shield tank capacitors (if required).
- It is recommended that you keep CapSense pins near the ground pin. Otherwise, the increase in the impedance of the ground path will cause the drive circuit's reference voltage to shift.
- LEDs or any switching signal should not be placed close to C_{MOD}/R_B pin to avoid crosstalk.

Refer to Pin Assignments for more details.

9.1.4 C_{MOD}

Ensure that the C_{MOD} is not adjacent to any switching/communication pin. Since there is an analog signal on C_{MOD}, it should be surrounded by CapSense (analog) signal rather than being surrounded by switching/communication (digital) signal. Ground of C_{MOD} should have a least possible path to device ground.

Family	C_{MOD} Value Recommended
CY8C20xx6/A/AS	2.2 nF for CSD 1.2 nF – 5.6 nF for CSA
CY8C21x34	5.6nF-10nF for PRS8 and PRS16 cofiguration. 22nF- 47nF for Prescaler Configuration
CY8C21x34 SmartSense,	10 nF
CY8C24x94 ,CY8C22x45	5.6nF-10nF
CY8C20x34	1.2 nF – 5.6 nF
CY8CMBR3xxx, CY8CMBR2xxx, CY8C20xx7A/S, PSoC3/4/5LP	2.2 nF

Refer to User Module/Component datasheets for more details. The User Module and Component datasheets get downloaded when you install PSoC Designer and PSoC Creator respectively.

9.1.5 R_B

Ensure that the R_B is not adjacent to any switching/communication pin.

Family	R_B Value Recommended
CY8C21x34	Minimum of 2 kΩ
CY8C21x34 SmartSense	15 kΩ

9.1.6 Series resistor on CapSense lines

560 ohm resistor on CapSense signal lines. If the series resistance value is set larger than 560 ohms, the slower time constant of the switching circuit limits the amount of charge that can transfer. This lowers the signal level, which in turn lowers SNR. Smaller values are better, but are less effective at blocking RF. For complete details refer to section Series Resistor.

9.1.7 Series resistor on Communication lines

A 330-Ω resistor is recommended on communication lines. If more than 330 Ω is placed in series on these lines, the voltage levels fall out of specifications with the worst-case combination of the supply voltages between systems and the input impedance of the receiver. 330 Ω will not affect the I^2C operation as the V_{IL} level still remains within the I^2C specification limit of 0.3 Vdd when PSoC outputs a LOW. For complete details, refer to Series Resistor.

9.2 Layout Checklist

No.	Category		Min	Max	Recommendations
1	Buttons	Shape	NA	NA	Solid round or rectangle with curved edges
		Diameter/Diagonal	5 mm	15 mm	10 mm
		Air gap between button and hatch	0.5 mm	2 mm	Should be equal to overlay thickness. Hatch can be connected to ground or shield.
		Placement near any switching element	NA	NA	Isolate switching signals from sensor and the sensor PCB traces.
2	Slider	Width of segment for 1 mm acrylic overlay thickness	2 mm		8 mm
		Width of segment for 3 mm acrylic overlay thickness	4 mm		
		Width of segment for 4 mm acrylic overlay thickness	6 mm		
		Air gap between segments	0.5 mm	2 mm	0.5 mm
		Air gap between hatch and slider	0.5 mm	2 mm	Equal to overlay thickness. Hatch can be connected to ground or shield.
		Height of segment	7 mm	15 mm	12 mm
3	Overlay	Type			Use material with a high dielectric constant except conductors. There should be air gap between sensor board and overlay/Front panel of the casing.
		Thickness for acrylic overlay		5 mm	
		Thickness for glass overlay		15 mm	Overlay thickness should be low. This is applicable to both buttons and sliders. For proximity sensors, thicker overlays increase the capacitance coupled with human hand/finger and the sensor and hence increases the signal. However, the parasitic capacitance might increase by a small amount.
4	Sensor traces	Width		7 mil	
		Length		300 mm for a standard (FR4) PCB 50 mm for flex PCB.	
		Air gap between ground and sensor traces	10 mil	20 mil	
		Turns			No sharp turns
		Routing			Should be routed on non sensor side. If any non CapSense trace crosses CapSense trace, ensure that intersection is orthogonal.
5	Via on sensors	Number of vias	1	2	1(including sensor trace and sensor pad)
		Via diameter			10 mil
6	Ground				Use hatch ground which reduces 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)

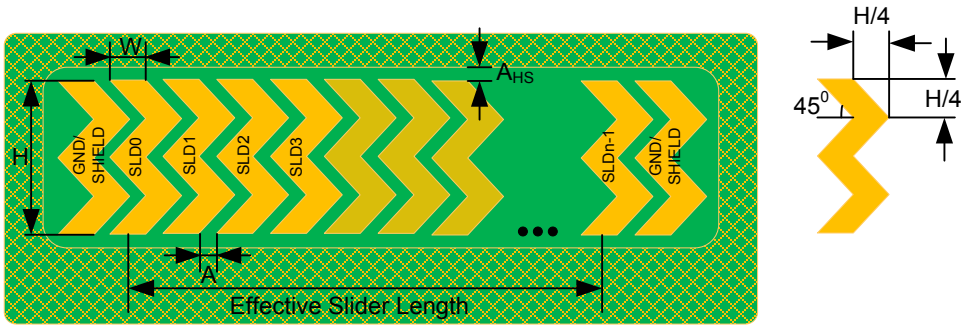
No.	Category		Min	Max	Recommendations
7	Series resistor				Place resistor within 10 mm of CapSense controller pin
8	Shield electrode	Size of shield fill		Width of 1 cm around Sensors	Refer to Figure 3-74. Shield Electrode Placement when Sensor Trace is routed in Top and Bottom Layer. Drive the shield only when required such as in applications requiring liquid tolerance and proximity sensing.
		Shield pattern			Use hatch fill instead of solid fill to reduce the parasitic capacitance of the shield electrode and to reduce the emissions. The hatching specifications are same as that of ground hatching. For detailed guidelines for reducing emissions, refer to Shield electrode.
9	Guard Sensor	Shape			Rectangular trace with curved edges
		Thickness			Recommended thickness of copper trace is 2 mm and distance of copper trace to hatch(ground/shield) is 1 mm.

9.2.1 Buttons

- The best shape for buttons is round. Rectangular shapes with rounded corners are also acceptable. Because sharp points concentrate fields, avoid sharp corners (less than 90º) when designing your sensor pad. Refer Figure 3-50 for details.
- Size can range from 5 mm to 15 mm. Go for larger diameter for thicker overlays.
- Air gap to ground and other sensors should be equal to the overlay thickness, but no smaller than 0.5 mm, and no larger than 2 mm. The spacing between the two adjacent buttons should be large enough that if one button is touched, a finger should not reach the air gap of the other button.
- Placement near any switching elements
  - Reduce the trace length from controller pins.
  - Mount series resistors within 10 mm of the controller pins.
  - Avoid connectors between sensors and other controller pins.

9.2.2 Slider

Figure 9-1. Typical Liner Slider Pattern



a. Keep the width of the segment (W) at 8 mm for an average finger width of 9 mm.
b. Keep the air gap between segments (A) at 0.5 mm.
c. Keep the height of the segment (H) at 12 mm.
d. Keep the air gap between the hatch and the slider (A_{HS}) equal to the overlay thickness.
e. For a slider with 'n' segments, employ n+2 segments. The first and last segments of the slider should be grounded or shielded depending on the application.

Refer to the Slider Design section for more details.

9.2.3 Overlay

a. Type (material): Do not use conductive materials as overlay since it interferes with the electric field pattern. Select a material with a higher dielectric constant.
b. Overlay thickness should be kept low to have high signal. Sensor signal is directly proportional to the finger capacitance. Finger capacitance is inversely proportional to the thickness of the overlay which is the distance between the two electrodes (sensor pad and finger) which together form a parallel plate capacitor. Hence reducing the overlay thickness increases the signal.
c. For sliders, with SmartSense, the maximum acrylic overlay thickness is 4 mm and that of glass is 12 mm.

9.2.4 Sensor traces

a. Keep the width less than or equal to 7 mil (0.18 mm)
b. Keep the maximum trace length as 12 inches (300 mm) for a standard PCB and 2 inches (50 mm) for flex circuits.
c. Keep the air gap between a CapSense trace and ground in the range of 10 mil to 20 mil (0.25 mm to 0.51 mm).
d. Do not have sharp (90 degrees) turns as this will pick up noise; in addition, charges concentrate at sharp corners.
e. Routing of traces (Refer Trace Routing for details)
    - Route sensor traces on the bottom layer of the PCB.
    - Do not run traces underneath a sensor unless the sensor and the trace are connected.
    - Do not route sensor traces in parallel to noisy clock and LED lines. Use ground or shield around sensor traces for shielding.
    - Do not run sensor traces in close proximity to communication lines, such as I^2C or SPI masters. If it is necessary to cross communication lines with sensor trace, make sure that the intersection is at right angles.

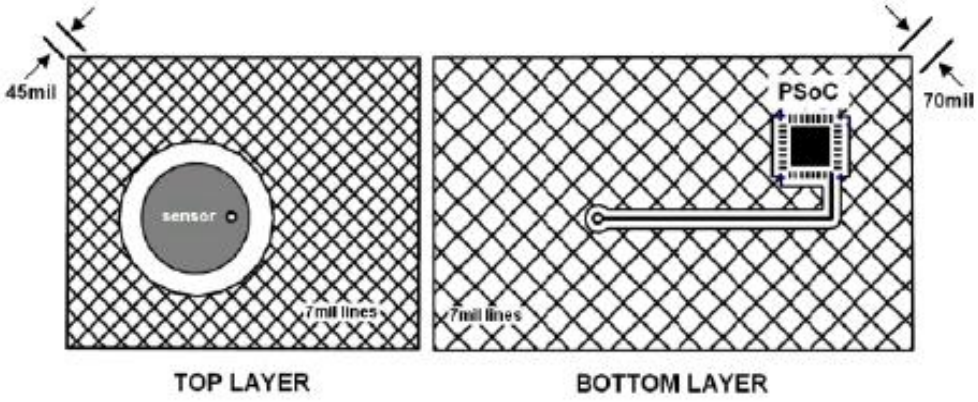
9.2.5 Vias on Sensors

a. Placement should be at the edges of the CapSense button to minimize increase in C_p due to vias.
b. Minimize the number of vias to reduce the parasitic capacitance, at up to two on a sensor (trace and pad).
c. Keep the via hole diameter for sensor traces at 10 mil (Refer Figure 3-68 for details).

9.2.6 Ground Plane/Mesh

Placing solid ground around sensors and decreasing the air gap between the sensor and the surrounding ground reduce noise, but increase the parasitic capacitance. Thus, there is a tradeoff between the CapSense signal and noise immunity. Hatching the ground decreases the ground area and therefore the parasitic capacitance.

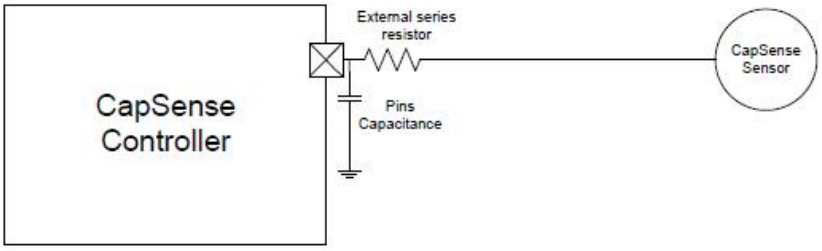
Figure 9-2. Typical Ground Fill



9.2.7 Series Resistor

Place series resistors within 10 mm of the CapSense controller pins. Adding an external resistor forms a low-pass RC filter that can dampen RF noise amplitude.

Figure 9-3. Series Resistor Placement



9.2.8 Shield Electrode

a. Reduce the size of the shield fill (maximum 1 cm from the sensor).
b. Limit the placement of the shield to only the selected sensors.
c. Slow the edges of the shield waveform to reduce emissions:
   - Adding a capacitor filter between the shield electrode port pin and ground will reduce slew rate.
   - Put a small value of series resistor

For more details on shield design considerations for emission reduction, refer to Shield .

9.2.9 Guard Sensor

a. The shield electrode should surround the guard sensor pad and exposed traces, and spread no further than 10 mm from these features.
b. The recommended shape for a guard sensor is rectangular trace with curved edges.
c. The recommended thickness of a copper trace is 2 mm and distance of the copper trace to the shield hatch is 1 mm.

Revision History

Document Revision History

Document Title: Getting Started With CapSense®

Document Number: 001-64846

Revision	Issue Date	Origin of Change	Description of Change
**	12/17/2010	SSHH	New guide
*A	03/04/2011	SSHH	Multiple chapter enhancements for content and reader clarity
*B	08/16/2011	SSHH/BVI	Multiple section and table updates
*C	12/07/2011	SSHH	Multiple chapter enhancements for content clarity
*D	04/27/2012	UDYG	Updated slider section. Updated PCB layout guidelines. Updated table 5-2. Corrected phone number in the title page.
*E	07/19/2012	ZINE	Updated sample schematics and layouts. Included information on layout/trace routing guidelines for C_{MOD}, R_B pin.
*F	08/31/2012	ZINE	Updated references to external documents
*G	10/16/2012	ZINE	Updated Section 3.1 (Overlay Selection) and moved Table 3-1 to Section 3.2.
*H	01/07/2013	SSHH	Updated Section 3.1. Added Section 3.7.13.
*I	03/07/2013	ZINE	Updated Section 2.7.2. Haptic Feedback.
*J	06/07/2013	DST	Added the CY8C20XX7/S part family. Added proximity information.
*K	09/04/2013	TEJU	Added the CY8C22X45 part family information.
*L	10/04/2013	LPG	Corrected document revision on the footer of some pages.
*M	02/19/2014	SSHH	Added information specific to CY8CMBR 3XXX
*N	03/05/2014	SHAS	Updated the Operating Voltage Specifications in Table 5-2. CapSense Key Features by Product Family

Revision	Issue Date	Origin of Change	Description of Change
*O	09/17/2014	SSHH/DCHE/SLAN	Updated Figure 5-1 Updated Figure 1-1 to have a mention of PSoC Creator Updated Table 1-1 to include references corresponding to PSoC 3/4/5LP Updated Figure 2-6 to remove unnecessary bends. Updated Figure 2-7 to have the correct block diagram Updated Figure 2-8 to remove unnecessary bends Updated Equation 6 to reflect the correct noise definition Updated Liquid Tolerance and Proximity (Three-Dimensional Sensors) Section Updated section 3.3 Electromagnetic Compatibility (EMC) Considerations with emission and immunity guidelines Updated event based filter and rule based filter explanations in sections 3.4.5 and 3.4.6 respectively. Added section 3.6 Proximity Sensing Updated references to PSoC 3/4/5LP in multiple places Added slider design in section 3.8.5 Updated Shield Electrode and Guard Sensor section Removed reference of CY3235 – Proximity Detection Demonstration Kit Added CY8CKIT-024 to the section Evaluation Kit for Proximity In Chapter 4 CapSense Product Portfolio, added PSoC 3/4/5LP references in multiple places In Chapter 5 CapSense Selector Guide, added PSoC 3/4/5LP references in multiple places Added CY8CMBR2x1x in Table 5-1 and Table 5-2 In Chapter 7 Resources, added PSoC 3/4/5LP references in multiple places Added a description on PSoC Creator in section 7.5 Added Appendix B: Schematic and Layout Checklist