Getting started with CAPSENSE™

About this document

Scope and purpose

This guide is an ideal starting point for those who are new to capacitive touch sensing (CAPSENSE™). You can use this guide to:

- Become familiar with the technology underlying CAPSENSE™ solutions
- Understand important design considerations, such as schematic, layout, and electromagnetic interference (EMI)
- Select the right device for the application
- Find a CAPSENSE™ resource to help with the design

When you are ready to design the application, consult the Design Guide specific to the CAPSENSE™ device family you have selected. See the Glossary for the definitions of CAPSENSE™ terms.

Intended audience

This application note is intended for users using (or interested in using) CAPSENSE™ devices.

Table of contents

Table of contents

Table of contents

Table of contents

1 Introduction

1.1 Infineon CAPSENSE™ differentiation

Capacitive touch sensing has changed the face of industrial design in products such as cellphones, PCs, consumer electronics, automotive, and white goods. Infineon 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. A CAPSENSE™-based user interface design has the following advantages over a mechanical-buttons based design:

- Mechanical buttons are less reliable and wear out over time due to the physical movement. CAPSENSE™ designs do not involve moving parts
- Mechanical buttons pose problems when moisture seeps through the gaps in the assembly. CAPSENSE™-based front panels can be completely sealed under the overlay
- Mechanical buttons require a small force to operate compared to the touch buttons and this force can increase over time due to the accumulation of dirt in the gaps
- Mechanical buttons require multiple parts and increase the BOM cost whereas many CAPSENSE™ designs consist of only a PCB and an overlay with adhesive
- Mechanical buttons include the cost of tools required to make cutouts in the front panel. CAPSENSE™ designs do not require such cutouts
- Mechanical buttons yield poor aesthetics compared to the sleek and elegant touch buttons. CAPSENSE™ designs also offer more flexibility in designing the user interface in terms of button shape and graphical representation

Infineon'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 cost. Infineon offers a wide range of configurable and programmable CAPSENSE™ controllers. Configurable CAPSENSE™ controllers are hardware or I2C configurable. Programmable devices provide complete flexibility to meet your exact design requirements, including reducing BOM cost by integrating further system functionality. Following are some of the unique features offered by CAPSENSE™ products.

- 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 V – 5.5 V)
- Small form-factor packaging
- Reduced BOM cost with integrated features like ADC, DAC, timer, counter, and PWM

1.2 CAPSENSE™ design flow

Figure 1 depicts the typical flow of a CAPSENSE™ product design. This flow is similar to any other electronic system design flow except that CAPSENSE™ designs involve an additional step called Tuning. This is the process of finding the optimum values for various hardware and software parameters required for CAPSENSE™ operation. These parameters vary depending on the board layout, sensor dimensions, overlay properties, and application requirements such as power consumption and response time. Therefore, this step is usually performed when the pre-production builds are available. Many of the CAPSENSE™ devices support Infineon's Auto-tuning algorithm called SmartSense that automatically sets parameters for optimal performance after the design phase and continuously compensates for system, manufacturing, and environmental changes.

The enclosure or casing design is an integral part of a CAPSENSE™ product design as the aesthetic feel and the performance of the end product depend on the casing material and its design. Since the casing acts as an overlay for the sensors, the touch-sensing performance depends on the overlay properties such as thickness and material type. Therefore, it is important to test and evaluate the performance along with the overlay material, which is similar to the one used in the end-product right from the prototype stage.

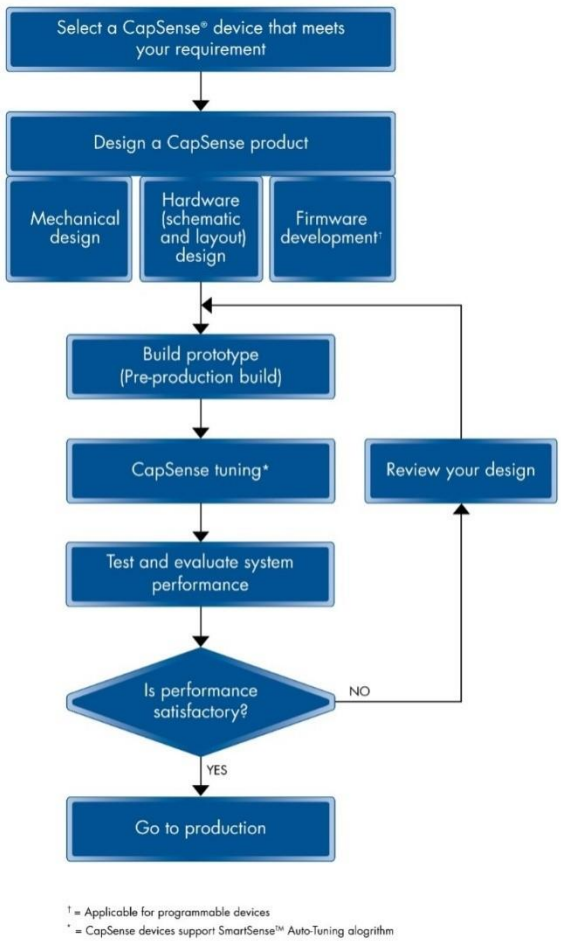

Figure 1 Typical CAPSENSE™ product design flow

2 CAPSENSE™ technology

Infineon'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. This touch button example illustrates a capacitive sensor replacing a mechanical button. The sensing function is achieved using a combination of hardware and firmware. See the Glossary for the definitions of CAPSENSE™ terms.

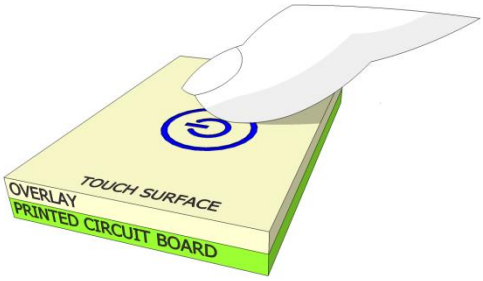

Figure 2 Illustration of a capacitive sensor application

2.1 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

2.1.1 Hardware component

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

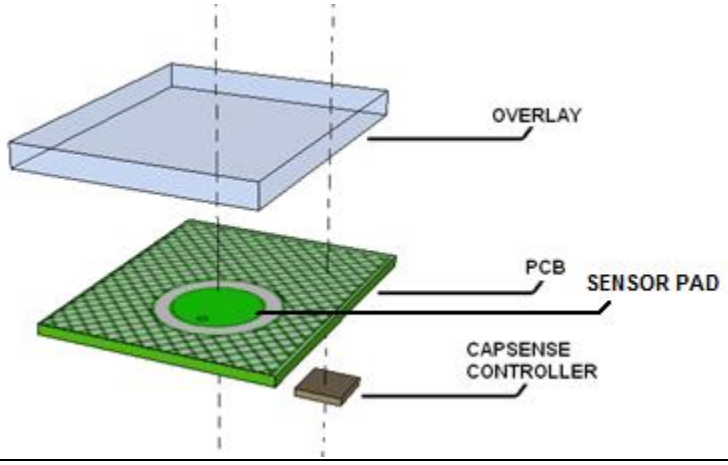

Figure 3 Exploded view of the CAPSENSE™ hardware

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 (see Ground plane) on the top, and the electrical components on the bottom. The ground plane is also provided surrounding the electrical components. The electrical components include the CAPSENSE™ controller and associated parts that convert the sensor capacitance into digital raw counts. Figure 4 shows a cross-sectional view of a two-layer board stack-up. The

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 the device-specific Design Guide.

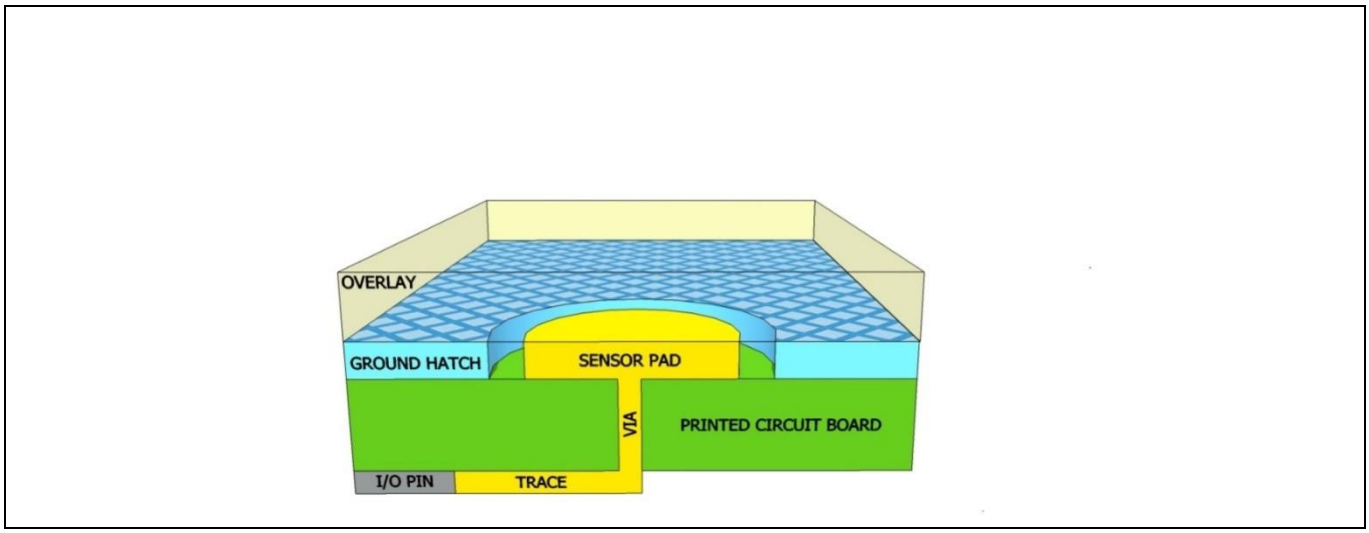

Figure 4 Two-layer stack-up of a CAPSENSE™ board

2.1.1.1 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. The increase in parasitic capacitance is unwanted as it reduces the sensitivity. 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 Figure 5 shows. Typical hatching for the ground fill is 7-mil line, 45 mil spacing on the top layer, and 7-mil line, and 70-mil spacing on the bottom layer. The same hatch-fill on the top layer is driven with shield signal when liquid tolerance is required. See Liquid tolerance to learn more.

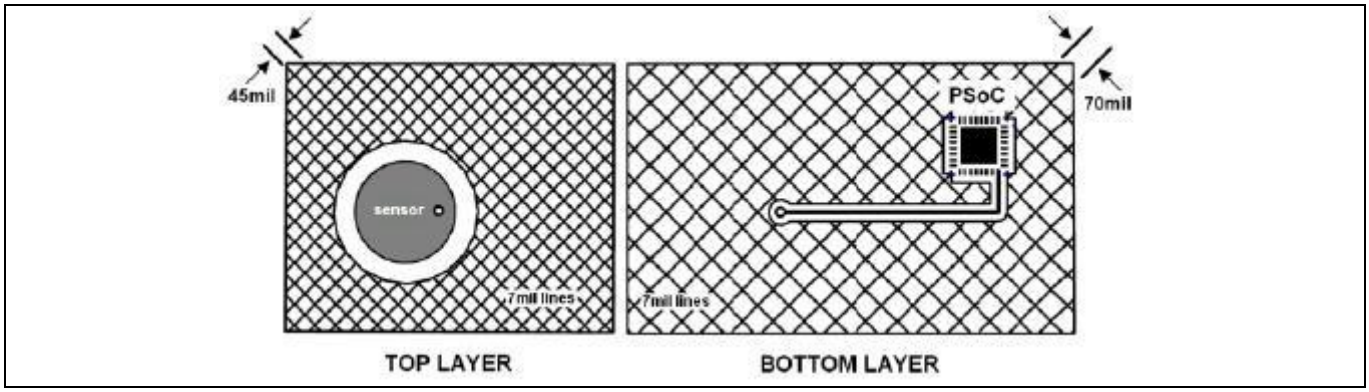

Figure 5 Ground fill on a PCB

2.1.2 Firmware component

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

Devices from the CAPSENSE™ Express family are fully configurable either through hardware or through I2C and do not require any firmware development on the CAPSENSE™ controller. The finger touch data is sent to a host for higher level processing; see Figure 6. These devices are appropriate for systems where simplicity of design and short time-to-market are the key requirements.

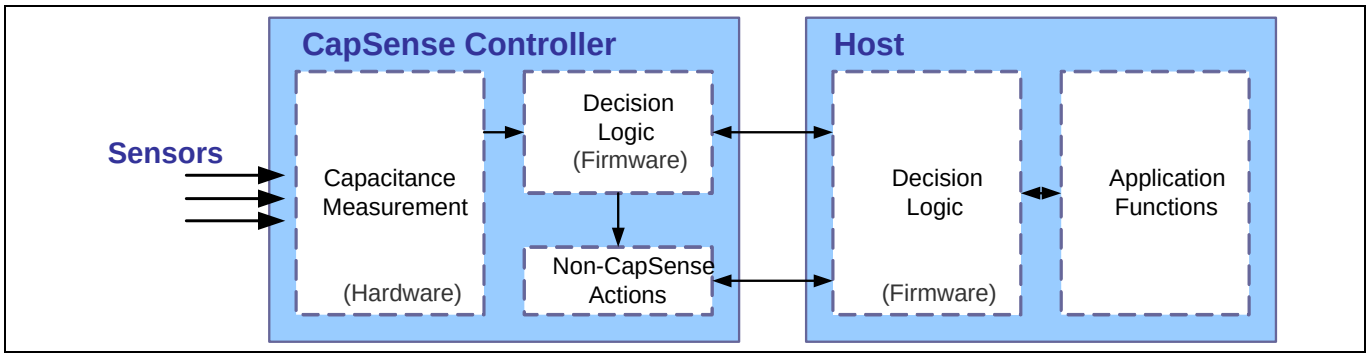

Figure 6 Example CAPSENSE™ express system implementation

The programmable devices allow complex system-level integration. These controllers can process the raw count data as well as perform other system functions.

See Capsense™ selector guide for additional details. Infineon's PSoC™ Creator, ModusToolbox™ software accommodate firmware development in C and assembly languages. See Software tools for more information on this and other tools.

2.2 CAPSENSE™ fundamentals

Capacitance can be measured between two points using either self-capacitance or mutual capacitance. The left side of Figure 7 shows the self-capacitance method and the right side shows the mutual-capacitance method.

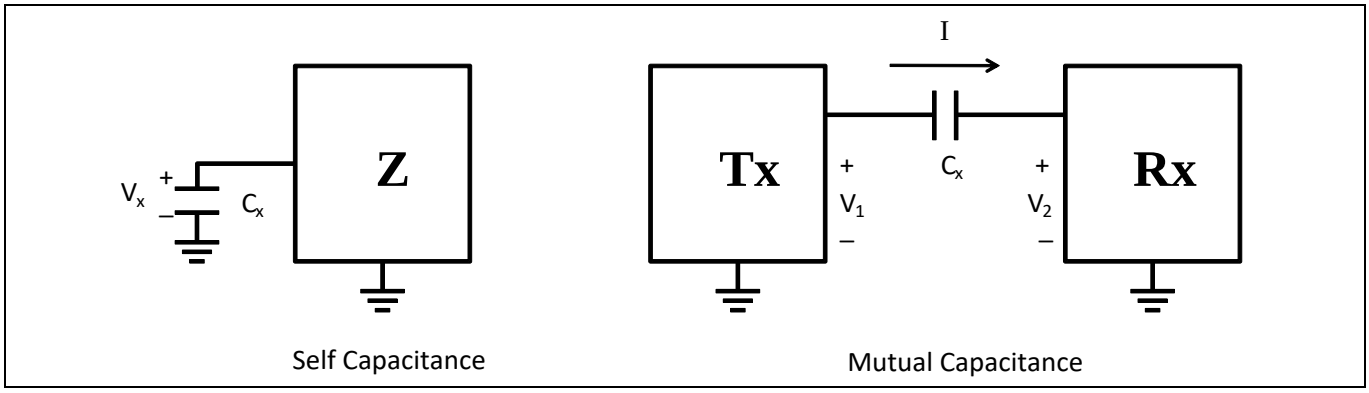

Figure 7 Self-capacitance and mutual-capacitance methods

2.2.1 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.

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

In a CAPSENSE™ self-capacitance system, the sensor capacitance measured by the controller is called C_S. When a finger is not on the sensor, C_S 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 the following diagram shows field lines only around the sensor pad, the actual electric field is more complicated.

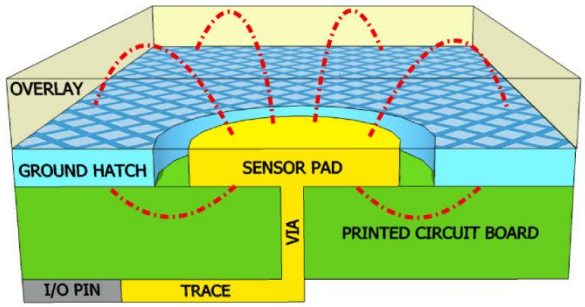

Figure 8 C_P and electric field

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

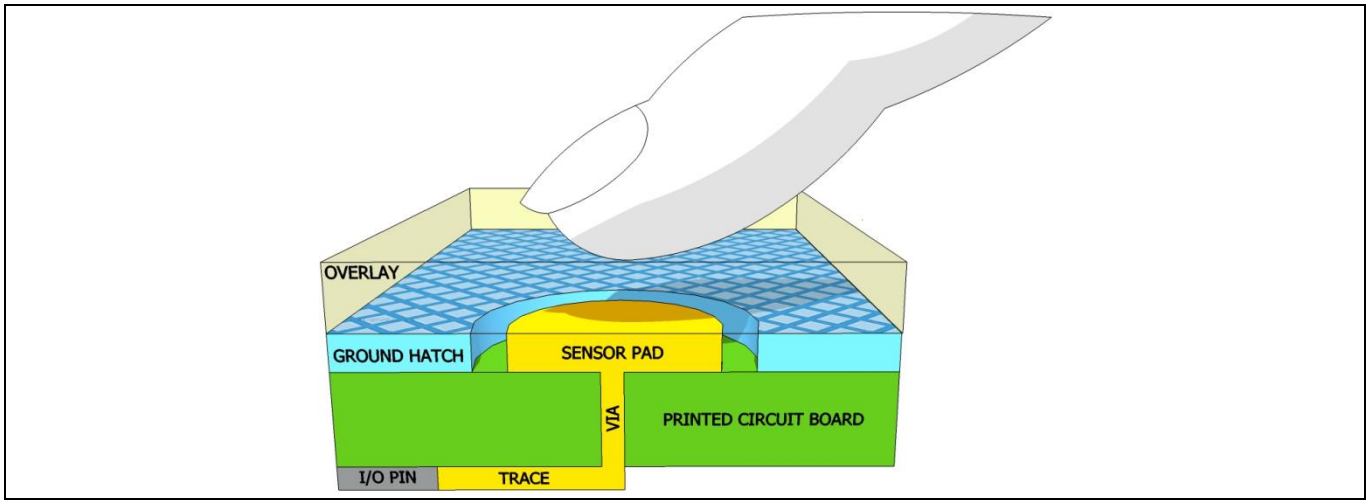

Figure 9 CAPSENSE™ system equivalent model

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

$$C_S = C_P + C_F$$

Equation 2

2.2.2 Mutual capacitance

Figure 10 shows the button sensor layout for mutual-capacitance sensing. Mutual-capacitance sensing measures the capacitance between two electrodes. One of the electrodes is called the transmit (TX) electrode and the other electrode is called the receive (RX) electrode.

In a mutual-capacitance measurement system, a digital voltage (signal switching between V_{DDD} and GND) is applied to the TX pin and the amount of charge received on the RX pin is measured. The amount of charge received on the RX electrode is directly proportional to the mutual capacitance (C_M) between the two electrodes.

When a finger is placed between the TX and RX electrodes, the mutual-capacitance decreases to C_M^1 as shown in Figure 11. Because of the reduction in mutual-capacitance, the charge received on the RX electrode also decreases. The CAPSENSE™ system measures the amount of charge received on the RX electrode to detect the touch/no touch condition.

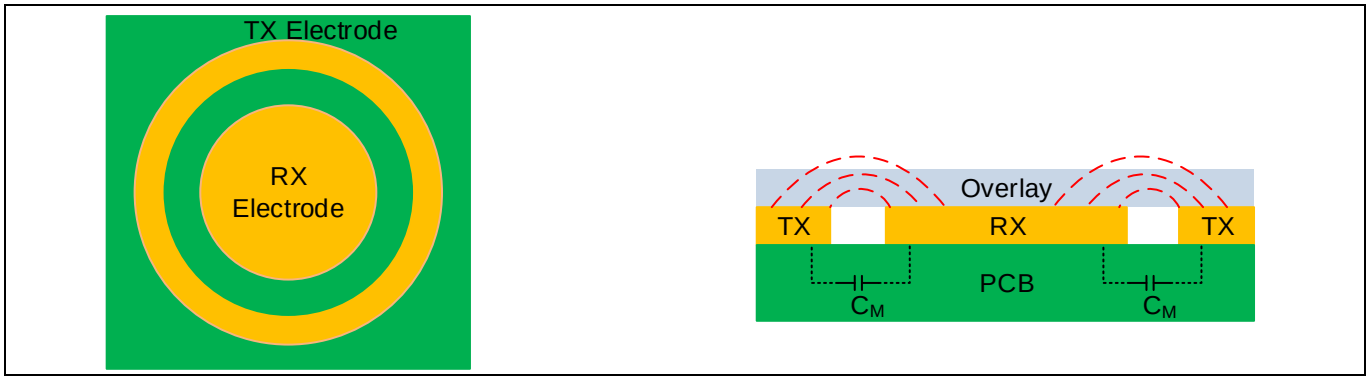

Figure 10 Mutual capacitance sensing working

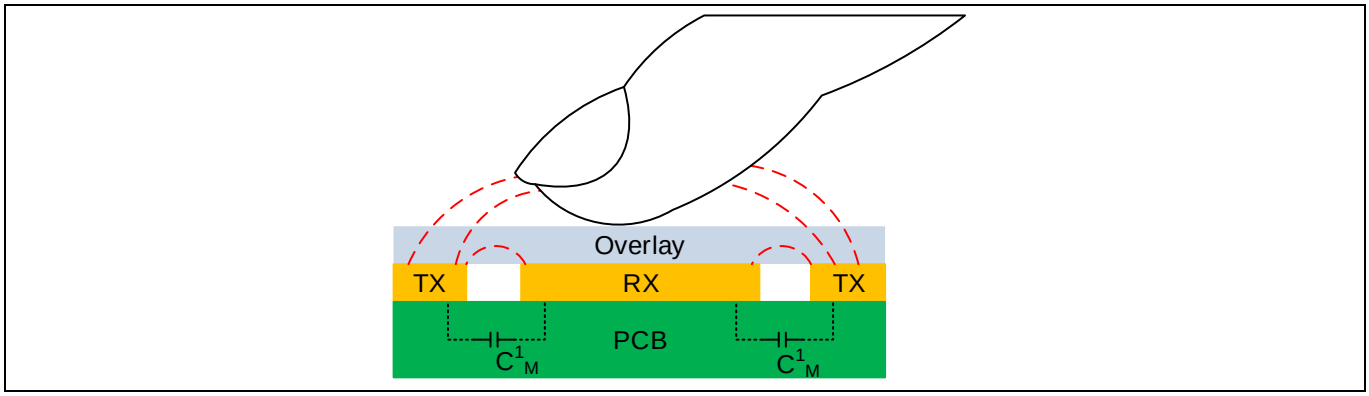

Figure 11 Mutual capacitance with finger touch

The mutual-capacitance effect is best suited to multi-touch systems such as touchscreens and trackpads. Infineon offers mutual-capacitance-based trackpad solutions for consumer applications and PSoC™ Automotive Multitouch touchscreen solutions for automotive and home appliance applications. Contact your local Infineon sales office directly for more information. To find your local sales office, click here.

2.3 Capacitive touch sensing method

PSoC™ uses Infineon patented capacitive touch sensing methods known as CAPSENSE™ Sigma Delta (CSD) for self-capacitance sensing and CAPSENSE™ Crosspoint (CSX) for mutual-capacitance scanning. The CSD and CSX touch sensing methods provide the industry's best-in-class Signal-to-noise ratio (SNR). These sensing methods are a combination of hardware and firmware techniques.

2.3.1 CAPSENSE™ sigma delta modulator (CSD) sensing method

Figure 12 shows a simplified block diagram of the CSD method.

In CSD, each GPIO has a switched-capacitance circuit that converts the sensor capacitance into an equivalent current. An analog multiplexer then selects one of the currents and feeds it into the current to digital converter. The current to digital converter is similar to a sigma delta ADC. The output count of the current to digital converter, known as **raw count**, is a digital value that is proportional to the self-capacitance between the electrodes.

Raw count and sensor capacitance relationship in CSD

$$\text{raw count} = G_C C_S$$

Equation 3

Where G_C is the capacitance to digital conversion gain of CSD, and C_S is the self-capacitance of the electrode.

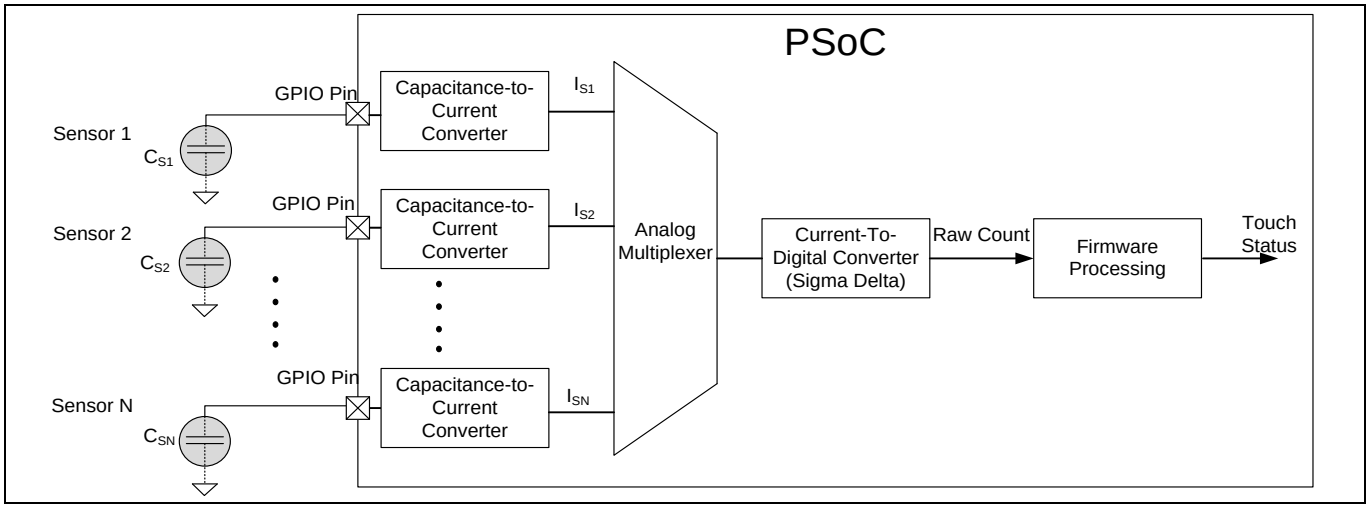

Figure 12 Simplified diagram of CAPSENSE™ sigma delta method

Figure 13 shows a plot of raw count over time. When a finger touches the sensor, the C_S increases from C_P to C_P + C_F, and the raw count increases. By comparing the change in raw count to a predetermined threshold, logic in firmware decides whether the sensor is active (finger is present).

2.3.2 CAPSENSE™ crosspoint (CSX) sensing method

Figure 13 shows the simplified block diagram of the CSX method.

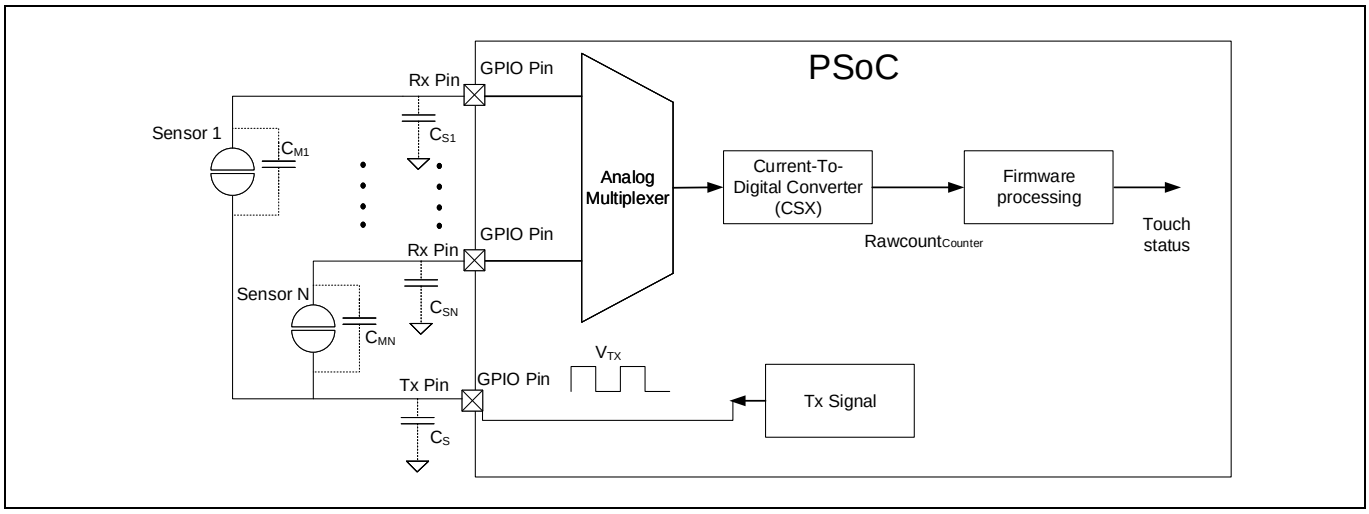

Figure 13 Simplified diagram of CSX method

With CSX, a voltage on the Tx pin (or Tx electrode) couples charge on to the RX pin. This charge is proportional to the mutual capacitance between the Tx and Rx electrodes. An analog multiplexer then selects one of the Rx channels and feeds it into the current to digital converter.

The output count of the current to digital converter, known as $Rawcount_{Counter}$, is a digital value that is proportional to the mutual-capacitance between the Rx and Tx electrodes as shown by Equation 4.

Raw count and sensor capacitance relationship in CSX

$$\text{Rawcount}_{\text{Counter}} = G_{\text{CM}}\, C_{\text{M}}$$

Equation 4

Where G_{CM} is the capacitance to digital conversion gain of Mutual Capacitance method, and C_M is the mutual-capacitance between two electrodes.

Figure 14 shows a plot of raw count over time. When a finger touches the sensor, C_M decreases from C_M to C_M^1 (see Figure 10) hence the counter output decreases. The firmware normalizes the raw count such that the raw counts go high when C_M decreases. This is to maintain the same visual representation of raw count between CSD and CSX methods. By comparing the change in raw count to a predetermined threshold, logic in firmware decides whether the sensor is active (finger is present).

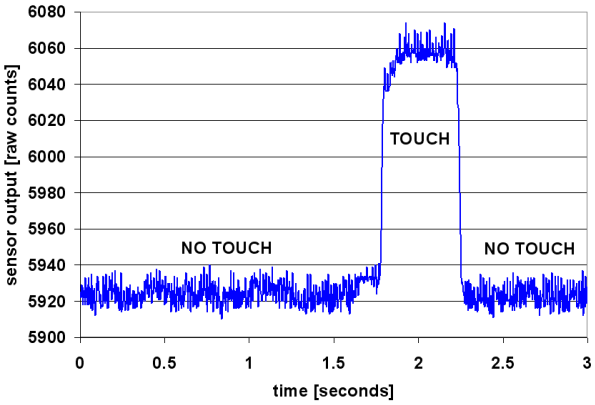

Figure 14 Raw count versus time

For a detailed derivation of raw count as a function of Sensor Capacitance and Finger Capacitance, see the AN85951 - PSoC™ 4 and PSoC™ 6 MCU CAPSENSE™ design guide.

2.4 Comparison of CAPSENSE™ architecture in different devices

The fourth-generation CAPSENSE™ functionality in PSoC™ 4 S-Series, PSoC™ 4100S Plus, PSoC™ 4100PS, and PSoC™ 6 devices is an improved version of the previous generation. The main differences between the two generations of CAPSENSE™ architecture are listed in Table 1.

Table 1 Comparison of CAPSENSE™ architecture

Feature	Third-generation CAPSENSE™	Fourth-generation CAPSENSE™	Advantages of fourth-generation over third generation CAPSENSE™	
			For CSD	For CSX
Sensor parasitic capacitance (C_P) range	5 pF – 60 pF	5 pF – 200 pF	Supports high-CP design applications	
Sensing modes	Self-Cap and Mutual-Cap modes[1]	Self-Cap, Mutual-Cap, and ADC modes	The CAPSENSE™ hardware block can be used as a 10-bit ADC when CAPSENSE™ sensor scanning is not in progress Refer PSoC™ 4 Capacitive Sensing (CAPSENSE™) ADC/CAPSENSE™ middleware library 4.0 datasheet for detailed ADC specifications.	
V_{REF}	1.2 V	0.6 V to VDDA-0.6 V[2]	Higher Vref allows improved SNR	NA
IDAC LSB size	1.2 µA, 2.4 µA	37.5 nA, 300 nA, 2.4 µA	Improved sensitivity, small IDAC for improved tuning.	
Split IDAC capability	Requires two IDACs	Requires one IDAC[3]	Requires fewer resources to achieve the same performance and frees up one IDAC for general-purpose use.	NA
EMI reduction – digital	Supports only PRS method	Supports additional spread spectrum clock (SSC) method	More options to control the sense/Tx clock frequency spread for EMI reduction.	
Modulator clock frequency range	Lower	Higher	Higher modulator-clock frequency implies faster scans.	Higher modulator-clock frequency implies increased sensitivity and accuracy.

[1] PSoC™ 4100 family does not support CSX sensing method since it does not have UDB resources which is required to implement CSX for this family.

[2] The CAPSENSE™ component automatically selects the VREF voltage depending on the VDDA voltage specified in the cydwr window.

[3] Require one IDAC if compensation and modulation IDAC split is 50-50; if it is not 50-50, it requires two IDACs.

CAPSENSE™ technology

Feature	Third-generation CAPSENSE™	Fourth-generation CAPSENSE™	Advantages of fourth-generation over third generation CAPSENSE™	
			For CSD	For CSX
Hardware state machine[4]	No	Yes	Initiation of sensor scanning is less dependent on CPU; there are fewer critical sections during scan initialization	
Tx clock frequency	Supports up-to 300 kHz	Supports much higher clock frequencies (up-to 3 MHz)	NA	Higher Tx clock results in shorter scan time

The CAPSENSE™ hardware in PSoC™ 4 S-Series, PSoC™ 4100S Plus, PSoC™ 4100PS, and PSoC™ 6 support both self-capacitance- and mutual capacitance-based capacitive sensing. The hardware also supports input voltage measurement when CAPSENSE™ scanning is not in progress. See the **CAPSENSE™** chapter in the respective device datasheet for a detailed explanation of the CAPSENSE™ hardware in PSoC™ 4 S-Series, PSoC™ 4100S Plus, PSoC™ 4100PS, and PSoC™ 6 devices. See the device-specific Design Guide for the basic knowledge of fourth-generation CAPSENSE™ architecture.

Table 2 PSoC™ device family and CAPSENSE™ architecture

PSoC™ device family	CAPSENSE™ architecture
PSoC™ 4	Third-generation CAPSENSE™
PSoC™ 4-M	
PSoC™ 4100-BLE	
PSoC™ 4-L	
PSoC™ 4 S-Series	Fourth-generation CAPSENSE™
PSoC™ 4100S Plus	
PSoC™ 4100PS	
PSoC™ 6 MCU	

2.5 CAPSENSE™ tuning

Optimal CAPSENSE™ system performance depends on the board layout, button dimensions, overlay material, and application requirements. 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.

Many of the CAPSENSE™ devices support SmartSense, Infineon's Auto-tuning algorithm, which automatically sets parameters for optimal performance and continuously compensates for system, manufacturing and environmental changes. See SmartSense auto-tuning for more information.

This section gives an introduction about the tuning process. For details on all the parameters involved in CAPSENSE™ operation and the step-by-step tuning procedure, see the device-specific Design Guide. Infineon provides many tools to make tuning and data monitoring easy. See Data monitoring tools to learn more about these tools.

[4] The hardware state machine is a logic which controls the CAPSENSE™ block and sensor scanning.

The following section defines some terms that will help you understand the tuning process.

2.5.1 Definitions

- **Raw count:** As seen in Figure 19, 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 raw count value of a sensor may vary gradually due to changes in the environment such as temperature and humidity. Therefore, the raw count is low-pass filtered to create a new count value known as baseline that keeps track of and compensates for the gradual changes in raw count. The baseline is less sensitive to sudden changes in the raw count caused by a touch. Therefore, the baseline value provides the reference level for computing the signals (explained below). Figure 15 shows the concept of raw count, baseline and signal.

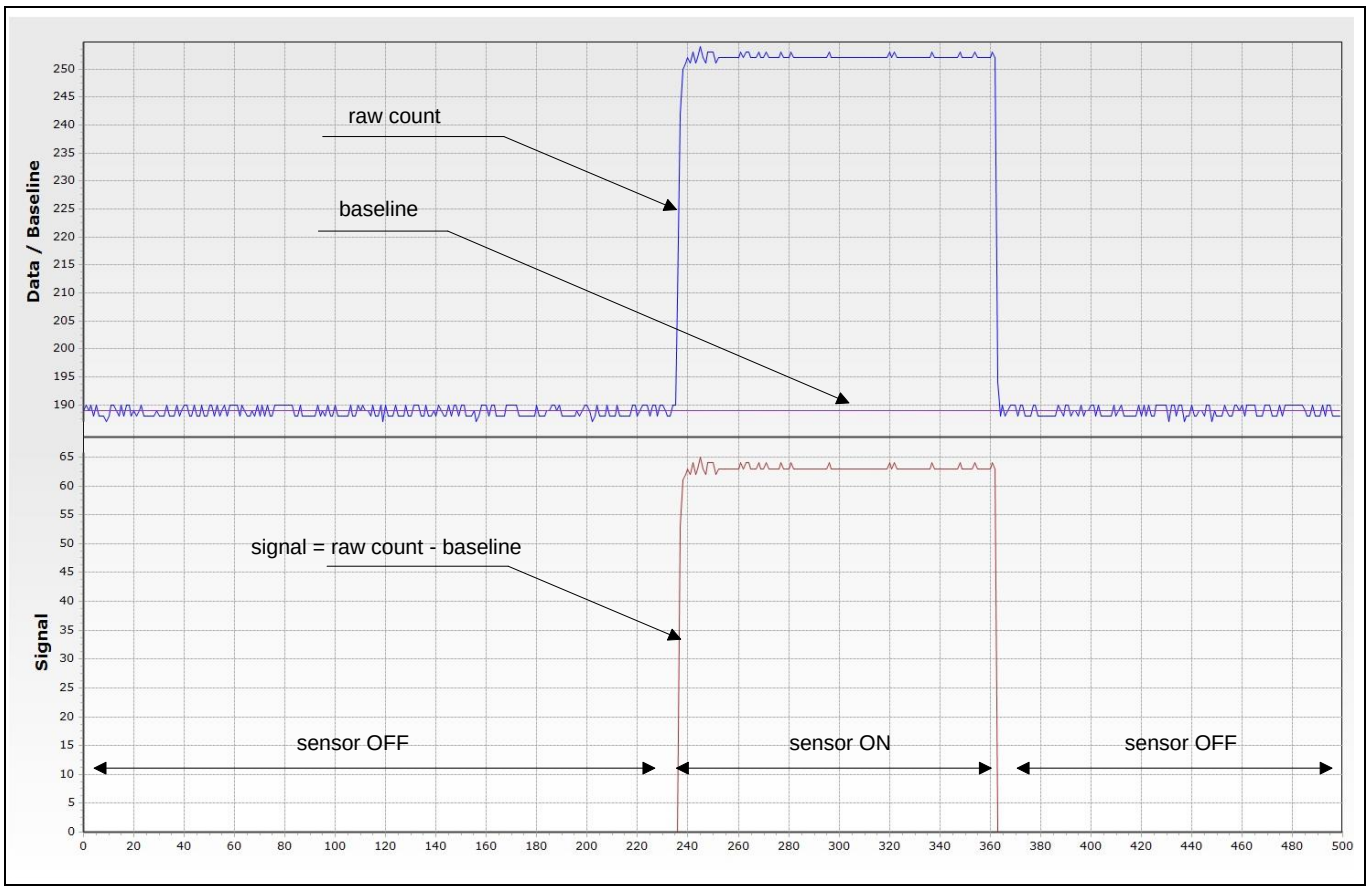

Figure 15 FRaw count and baseline

- **Difference count or signal:** Subtracting the baseline level from the raw count produces the difference count that is used in the ON/OFF decision process. The thresholds are offset by a constant amount from the baseline level. The thresholds have the following functions.
- **Noise threshold:** A parameter used to differentiate signal from noise. If the raw count is above noise threshold, then the baseline is not updated and the difference count indicates the difference between raw count and baseline. If the raw count is below the noise threshold, then the baseline is updated and the difference count is zero. See Figure 16 for details.
- **Finger threshold:** A parameter used with Hysteresis to determine the state of the sensor, as Equation 5 and Figure 16 show.

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

Equation 5

Hysteresis: A parameter used with Finger Threshold to determine the state of the sensor, as Equation 5 and Figure 16 shows. Hysteresis provides immunity against noisy transitions of sensor state.

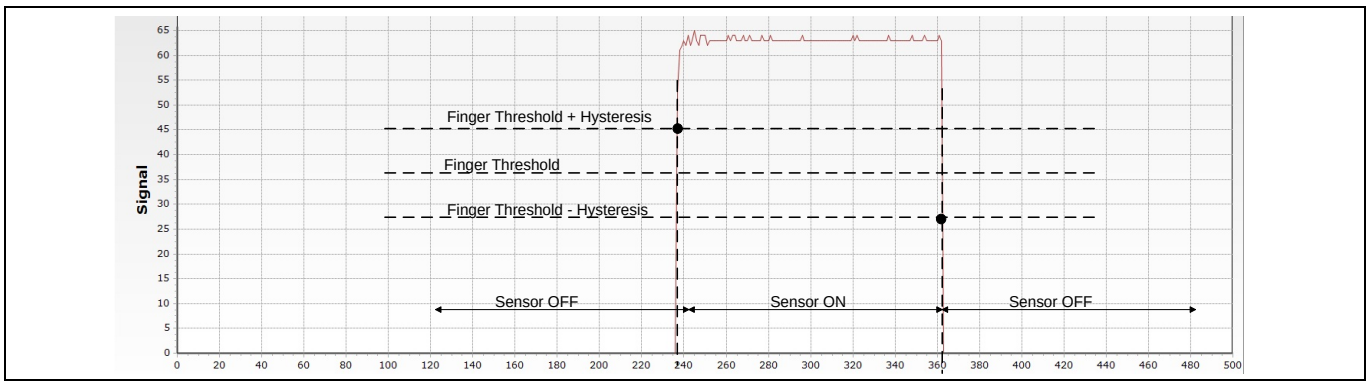

Figure 16 Hysteresis

2.5.2 SmartSense auto-tuning

2.5.2.1 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.

2.5.2.2 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 4 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.

2.5.2.3 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.

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.

Different noise levels in different designs

SmartSense technology dynamically tunes itself (adjusts noise and finger thresholds) for different noise environments. In Figure 17, Design A and Design B have different noise levels. To maintain a minimum SNR of 5:1, you must adjust the dynamic threshold. SmartSense does this automatically, allowing seamless transition from one model to another with minimal or no tuning required.

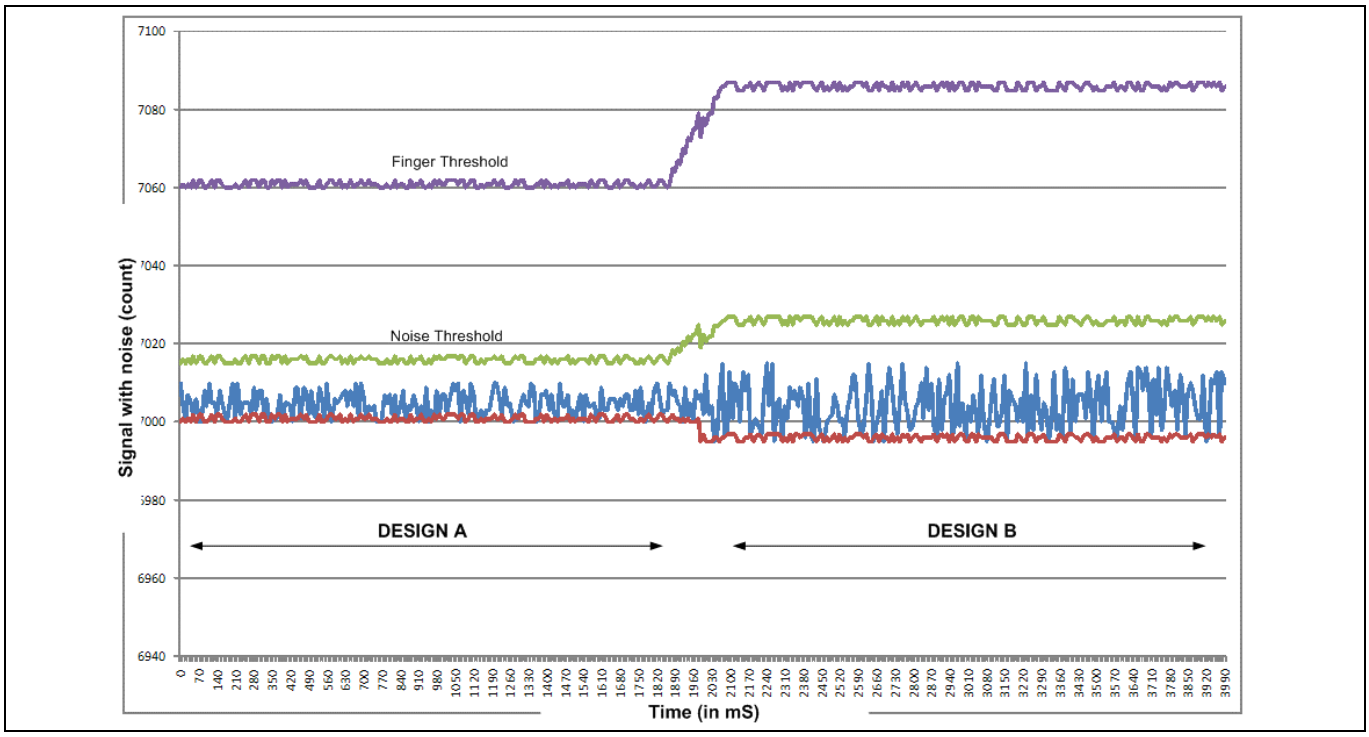

Figure 17 Different noise levels in design A and B being compensated automatically

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 18. This is a powerful SmartSense feature that prevents false button presses in the end system, which prevents a failure analysis for a mass production design.

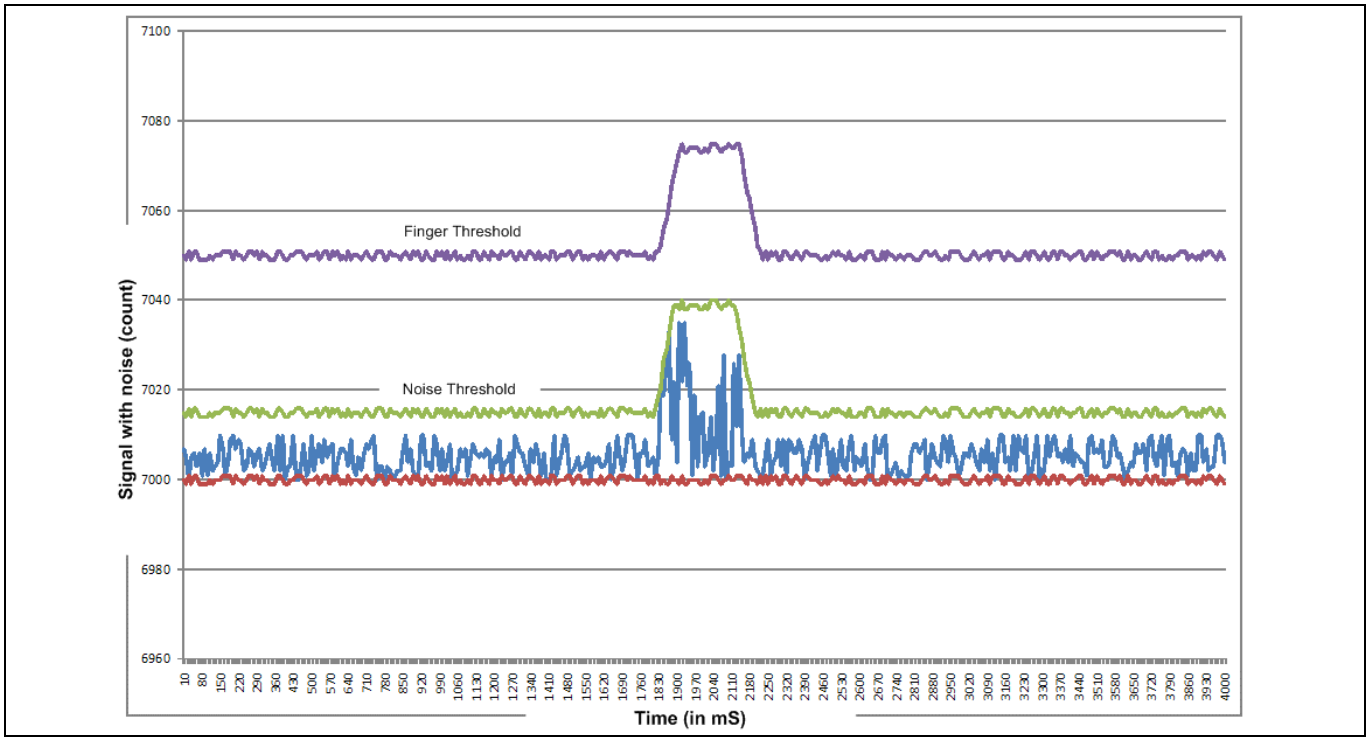

Figure 18 Finger threshold dynamically adjusted to prevent false button touches

2.5.2.4 When is manual-tuning advantageous?

SmartSense allows a device to calibrate itself for optimal performance and complete the entire tuning process automatically. This technology will meet the needs of most designs, but, in the case where SmartSense will not work or there are specific SNR or power requirements, the CAPSENSE™ CSD parameters can be manually adjusted to meet system requirements. This is called manual tuning. Some advantages of manual tuning, as opposed to SmartSense Auto-tuning are:

- Strict control over parameter settings: SmartSense sets all of the parameters automatically. However, there may be situations where you need strict control over the parameters. For example, use manual tuning if you need to strictly control the time CSD takes to scan a group of sensors. This can be done to reduce EMI in systems.
- Supports higher parasitic capacitances: SmartSense supports parasitic capacitances as high as 45 pF for a 0.2-pF finger capacitance, and as high as 35 pF for a 0.1-pF finger capacitance. If the parasitic capacitance is higher than the value supported by SmartSense, use manual tuning.

See the device-specific Design Guide for the step-by-step procedure on manual tuning.

2.6 Signal-to-noise ratio (SNR)

Signal is a generic engineering term that can have many meanings. For CAPSENSE™ applications, signal is defined as the change in the raw count between the OFF and ON states. Signal is also called Difference Count.

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 6

- *X*0 is the average of *X(t)*
- *N*0(*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 7

- *X*1 is the average of *X*(*t*)
- *N*1(*t*) is the noise component for t during the ON state

*X*0 is called the baseline level of the raw counts. The difference between *X*1 and *X*0 is called the signal, S.

$$S = X1 - X0$$

Equation 8

The noise components *N*0(*t*) and *N*1(*t*) are similar but not identical. For example, *N*1(*t*) usually contains a higher level of AC line noise in finger sensing applications compared to *N*0(*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.

$$N = \max\big(N0(t)\big) = \max\big(X(t)\big) - \min\,(X(t))$$

Equation 9

Thus, CAPSENSE™ SNR, is defined as the ratio of signal (*S*) to noise (*N*).

$$SNR = S:N$$

Equation 10

Based on the experiments and knowledge from many CAPSENSE™ applications, Infineon recommends a minimum SNR of 5:1 to ensure sufficient margin between noise and signal for robust ON/OFF operation.

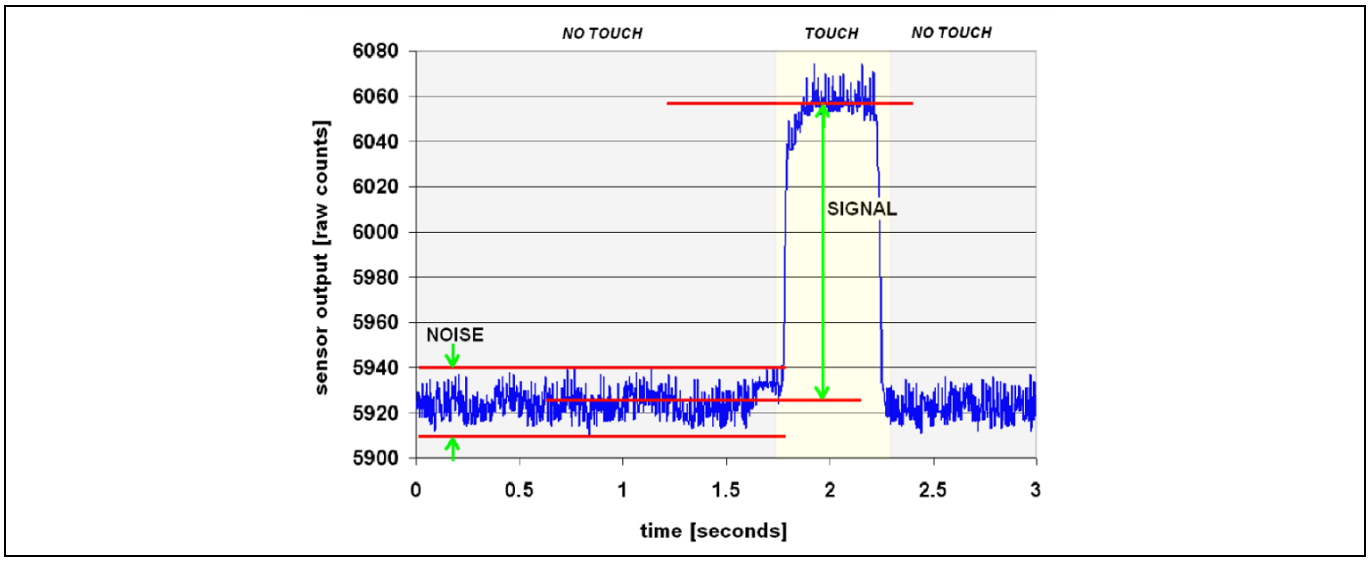

Figure 19 Signal and noise

2.6.1 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 Infineon Bridge Control Panel and Miniprog3 or MiniProg4[5] or by using the Tuner tool, which directly displays the SNR, available with the CAPSENSE™ component in PSoC™ Creator (see Data monitoring tools for more details). See AN2397 – PSoC™ 1 and CAPSENSE™ controllers - CAPSENSE™ data monitoring tools that teaches how to monitor raw count using these tools. 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.

[5] To know more about the Programming and Debugging kits, see Kits for programming and debugging.

As an example of measuring SNR, consider the raw count waveform in Figure 19.

X0 = 5925 counts

X1 = 6055 counts

S = 130 counts

N = 5940 – 5910 = 30 counts

SNR = 130:30 = 4.3:1

2.7 CAPSENSE™ widgets

CAPSENSE™ widgets consist of one or more CAPSENSE™ sensors, which as a unit represent a certain type of user interface. CAPSENSE™ widgets are broadly classified into four categories – Buttons (Zero-Dimensional), Sliders (One-Dimensional), Touchpads/Trackpads (Two-Dimensional), and Proximity sensors (Three-Dimensional). Figure 20 shows button, slider, and proximity sensor widgets. This section explains the basic concepts of different CAPSENSE™ widgets. For a detailed explanation of sensor construction, see Sensor construction.

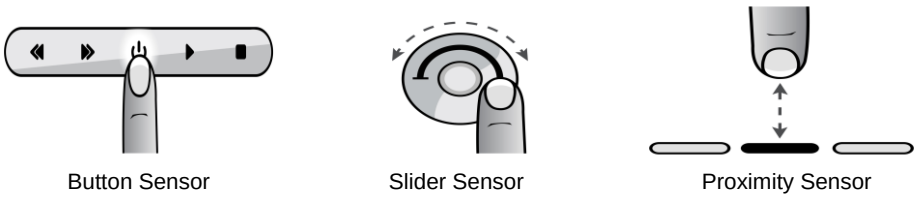

Figure 20 Several types of widgets

2.7.1 Buttons (zero-dimensional)

CAPSENSE™ buttons replace mechanical buttons in a wide variety of applications such as home appliances, medical devices, white goods, lighting controls, and many other products. It is the simplest type of CAPSENSE™ widget, consisting of a single sensor. A CAPSENSE™ button gives one of two possible output states: active (finger is present) or inactive (finger is not present). These two states are also called ON and OFF states, respectively.

For the self-capacitance based i.e. CSD sensing method, a simple CAPSENSE™ button consists of a circular copper pad connected to a PSoC™ GPIO with a PCB trace. The button is surrounded by grounded copper hatch to isolate it from other buttons and traces. A circular gap separates the button pad and the ground hatch. Each button requires one PSoC™ GPIO. These buttons can be constructed using any conductive material on a non-conductive substrate; for example, indium Tin Oxide on a glass substrate, or silver ink on a non-conductive film. Even metallic springs can be used as button sensors; see Sensor construction for more details.

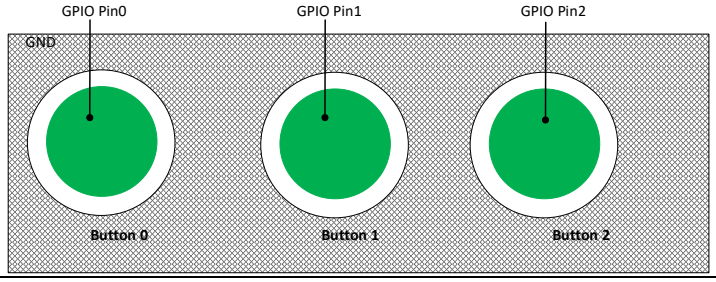

Figure 21 Simple CAPSENSE™ buttons for self-capacitance sensing method

For the mutual-capacitance based i.e. CSX sensing method, each button requires one GPIO pin configured as Tx electrode and one GPIO pin configured as Rx electrode. The Tx pin can be shared across multiple buttons, as shown in Figure 22.

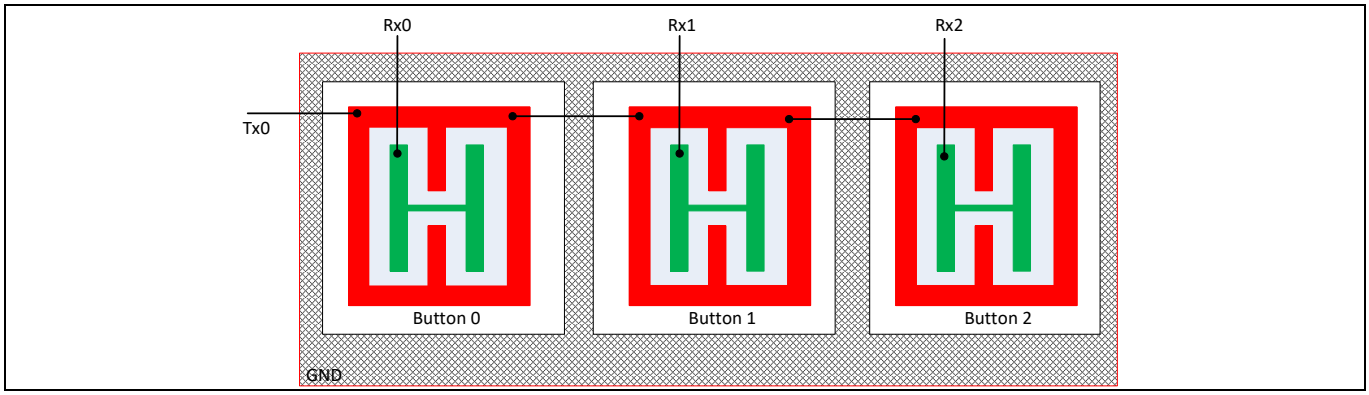

Figure 22 Simple CAPSENSE™ buttons for mutual-capacitance sensing method

If the application requires many buttons, such as in a calculator keypad or a QWERTY keyboard, you can arrange the CAPSENSE™ buttons in a matrix, as Figure 23 shows. This allows a design to have multiple buttons per GPIO. For example, the 12-button design in Figure 23 requires only eight GPIOs.

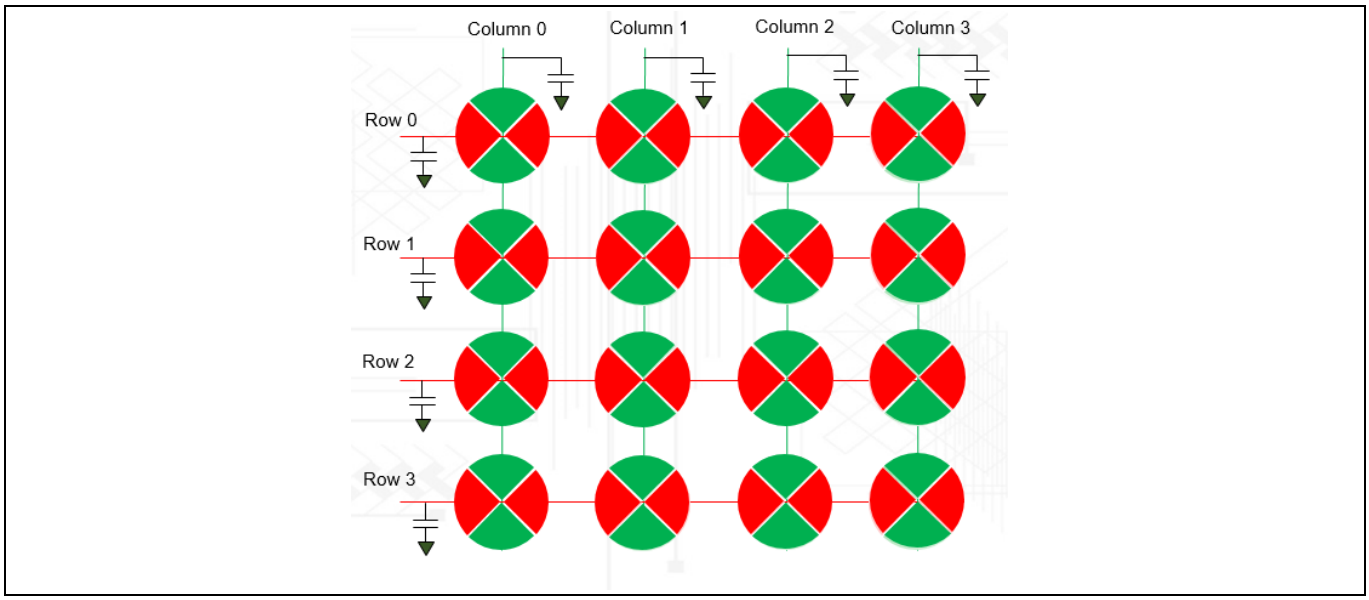

Figure 23 Matrix buttons based on CSD

A matrix button design has two groups of capacitive sensors: row sensors and column sensors. The matrix button architecture can be used for both self-capacitance (CSD) and mutual-capacitance (CSX) methods.

In Self-capacitance mode, each button consists of a row sensor and a column sensor, as Figure 23 shows. When a button is touched, both row and column sensors of that button become active. The CSD-based matrix button should be used only if the user is expected to touch one button at a time. If the user touches more than one diagonally opposite buttons, the finger location cannot be resolved as Figure 24 shows. This effect is called as ghost effect, which is considered an invalid condition.

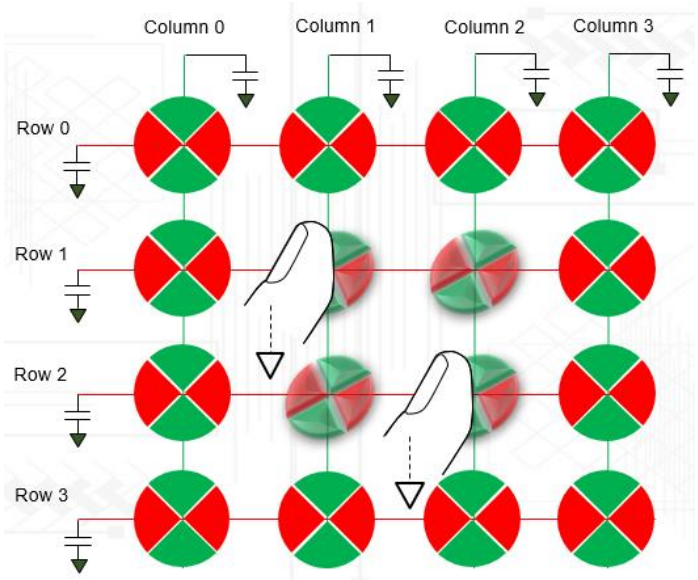

Figure 24 Ghost effect in matrix button based on CSD

Mutual capacitance is the recommended sensing method for matrix buttons because it doesn't suffer from ghost touch and provides better SNR for high Cp sensors. This is because it senses mutual capacitance formed at each intersection rather than sensing rows and columns as shown in Figure 25. Applications that require simultaneous sensing of multiple buttons, such as a keyboard with Shift, Ctrl, and Alt keys can use mutual-capacitance sensing method or you should design the Shift, Ctrl, and Alt keys as individual CSD buttons.

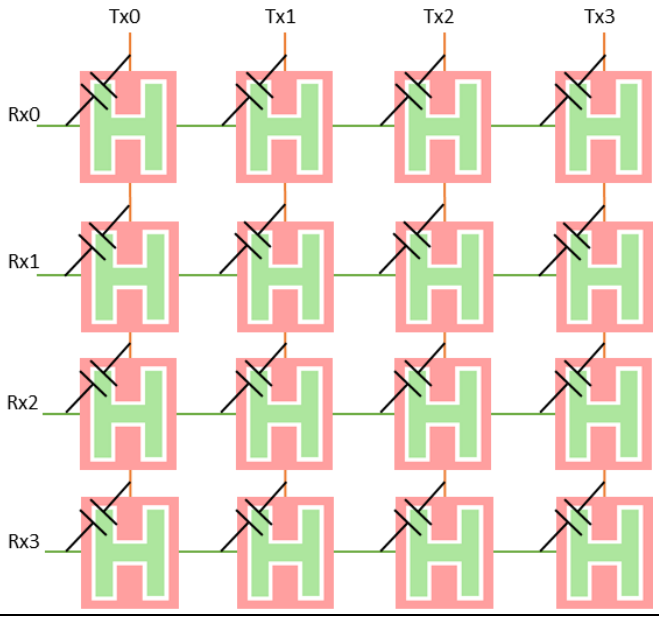

Figure 25 Matrix button based on CSX

Note however that scanning a matrix keypad using CSX sensing method may require a longer overall scan time than the CSD sensing method. This is because the CSD sensing method scans rows and columns as sensors, while the CSX sensing method scans each intersection as a sensor.

2.7.2 Sliders (one-dimensional)

Sliders are used when the required input is in the form of a gradual increment or decrement. Examples include lighting control (dimmer), volume control, graphic equalizer, and speed control. Currently, the CAPSENSE™ Component in PSoC™ Creator and ModusToolbox™ supports only self-capacitance-based sliders. Mutual capacitance-based sliders will be supported in future version of component.

A slider consists of a one-dimensional array of capacitive sensors called segments, which are placed adjacent to one another. Touching one segment also results in partial activation of adjacent segments. The firmware processes the raw counts from the touched segment and the nearby segments to calculate the position of the geometric center of the finger touch, which is known as the **centroid position**.

The actual resolution of the calculated centroid position is much higher than the number of segments in a slider. For example, a slider with five segments can resolve at least 100 physical finger positions. This high resolution gives smooth transitions of the centroid position as the finger glides across a slider.

In a linear slider, the segments are arranged inline, as Figure 26 shows. Each slider segment connects to a PSoC™ GPIO. A zigzag pattern (double chevron) is recommended for slider segments. This layout ensures that when a segment is touched, the adjacent segments are also partially touched, which aids estimation of the centroid position.

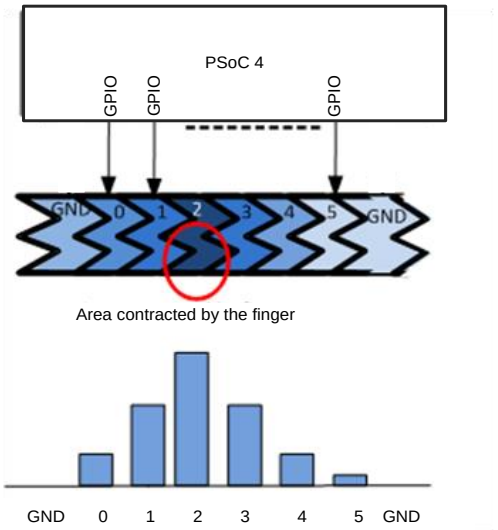

Figure 26 Linear slider

Radial sliders are similar to linear sliders except that radial sliders are continuous. Figure 27 shows a typical radial slider.

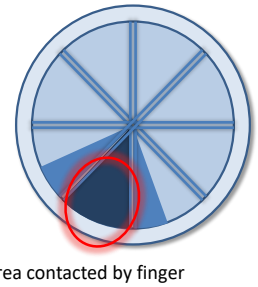

Figure 27 Radial slider

2.7.3 Touchscreens and trackpads (two-dimensional sensors)

A trackpad (also known as touch pad) has two linear sliders arranged in an X and Y pattern, enabling it to locate a finger's position in both X and Y dimensions. Figure 28 shows a typical arrangement of a track pad sensor.

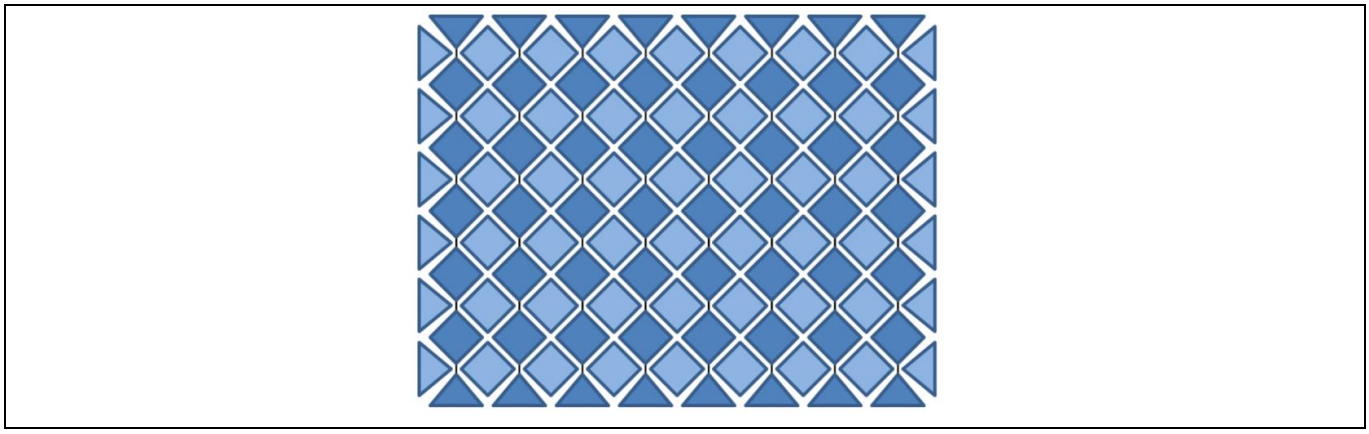

Figure 28 Trackpad sensor arrangement

2.7.4 Proximity (three-dimensional sensors)

Proximity sensors detect the presence of a hand or other conductive object before it makes contacts 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 29. Another way to implement a proximity sensor is by ganging sensors together. See Proximity sensing to learn more.

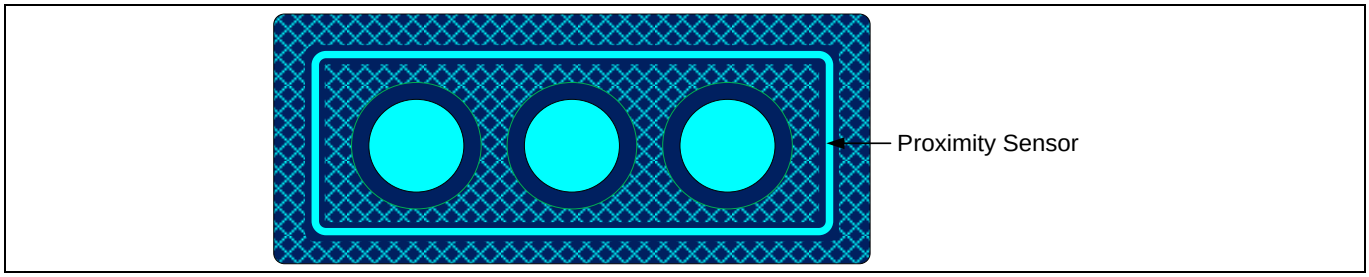

Figure 29 Proximity sensor

2.8 Sensor construction

A capacitive sensor can be constructed using different materials depending on the application requirement. In a typical sensor construction, a conductive pad or surface that senses the user touches is connected to the pin of the capacitive controller using a conductive trace or link. This whole arrangement is placed below a non-conductive overlay material and the user interacts on top of the overlay. A very common method of sensor construction is to etch copper pads and traces on a FR4 PCB. However, in touchscreen applications, Indium Tin Oxide (ITO) is used to construct transparent sensors. This section presents various methods of constructing a sensor and the features of each method so that you can choose one that fits your requirements.

2.8.1 Field-coupled via copper trace (PCB)

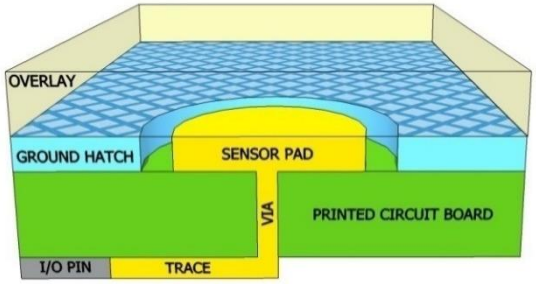

Figure 30 Field coupled using PCB

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

2.8.2 Field coupled via spring/gasket/foam

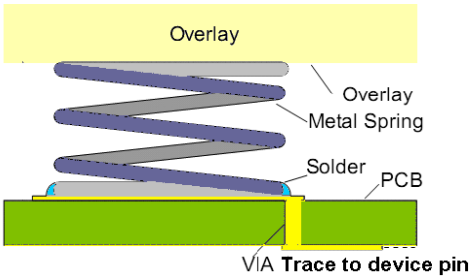

Figure 31 Field coupled via spring

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 (Closed-cell conductive foam should be used. Materials that absorb moisture should be avoided.)
- Conductive material itself acts as capacitive sensor pad
- No mechanical moving parts. Springs and foam do not move
- Any non-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

2.8.3 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 sheet resistance (ohms-per-square) of printed ink compared to copper. However, the series resistance of the materials such as silver loaded inks, ITO, and PEDOT depend on the thickness variation as their sheet resistance is relatively low
- High parasitic capacitance due to thin PCB substrate
- No mechanical moving parts, but substrate is flexible
- Coupled to the touch sensor surface with a nonconductive overlay
- Ideal topology for flexible front panels
- Flexible PCB can be one-layer or two-layer film

2.8.4 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
- No mechanical moving parts
- Ideal topology for graphical front panels

2.9 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. Infineon'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 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 33 shows. All sensor pins can optionally have a 560-Ω series resistor for improved noise immunity.

When liquid droplets are present on the touch surface and if the 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.

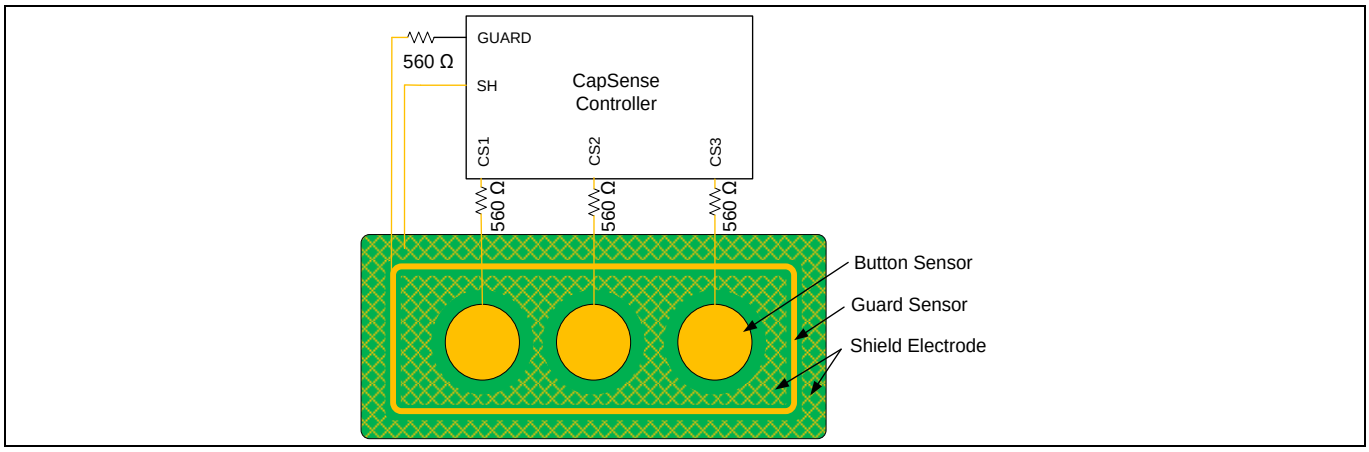

Figure 33 Shield electrode (SH) and guard sensor (GUARD) connected to CAPSENSE™ controller

2.9.1 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 34 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 34.

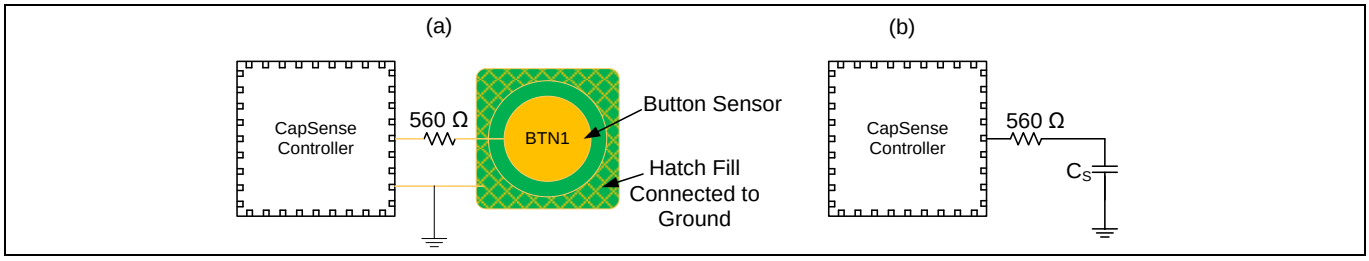

Figure 34 Typical CAPSENSE™ system layout

As shown in Figure 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 CLD in parallel to CP. When the sensor is charged and discharged, the capacitance CLD draws some amount of charge from the AMUX bus because of the nonzero voltage difference across the capacitance CLD. 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 36 shows.

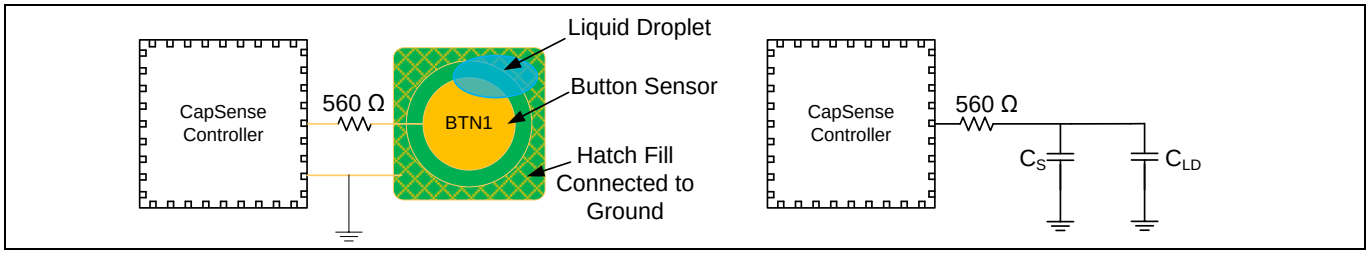

Figure 35 Capacitance added by liquid droplet when the hatch fill is connected to ground

C_S – Sensor parasitic capacitance

C_{LD} – Capacitance added by liquid droplet

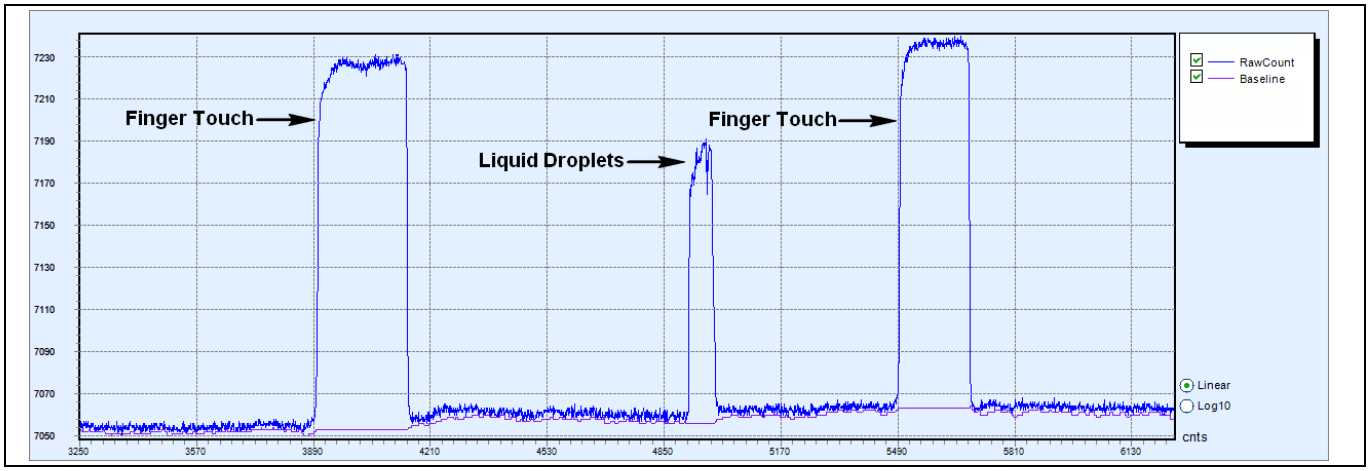

Figure 36 Effect of liquid droplet when the hatch fill around the sensor is connected to ground

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 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 38 shows.

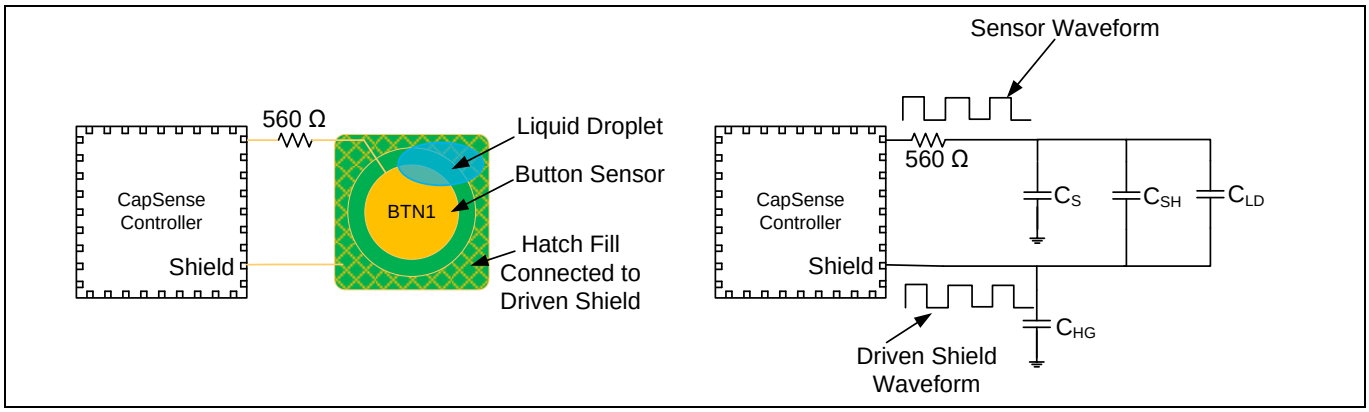

Figure 37 Capacitance added by liquid droplet when the hatch fill around the sensor is connected to shield

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

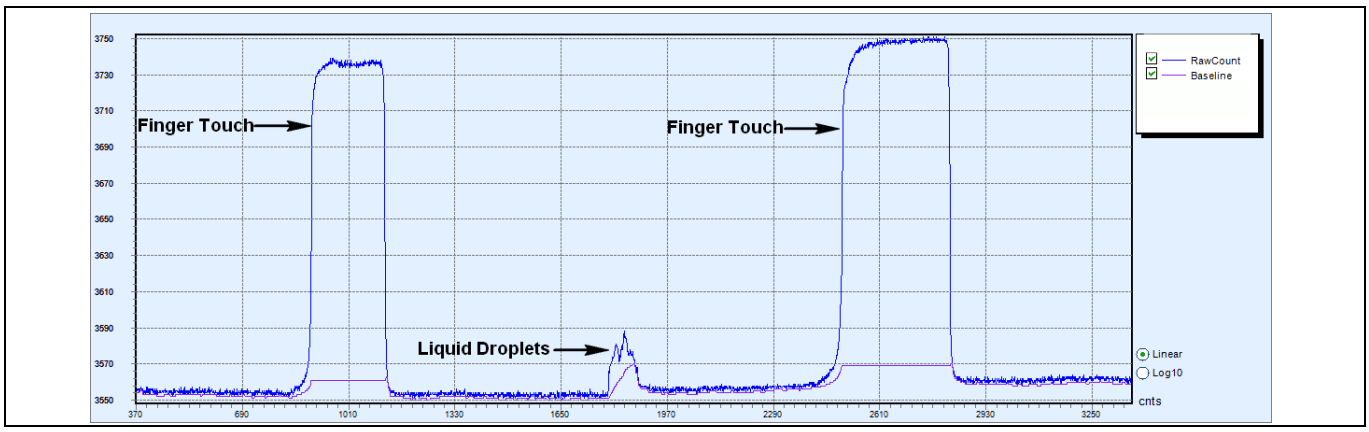

Figure 38 Effect of liquid droplet when the hatch fill around the sensor is connected to the driven shield

2.9.2 Driven-shield signal and shield electrode

The driven-shield signal is a buffered version of the sensor-switching signal, as Figure 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 Effect of liquid droplets and liquid stream on CAPSENSE™, because the shield electrode is driven with a voltage which is the 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.

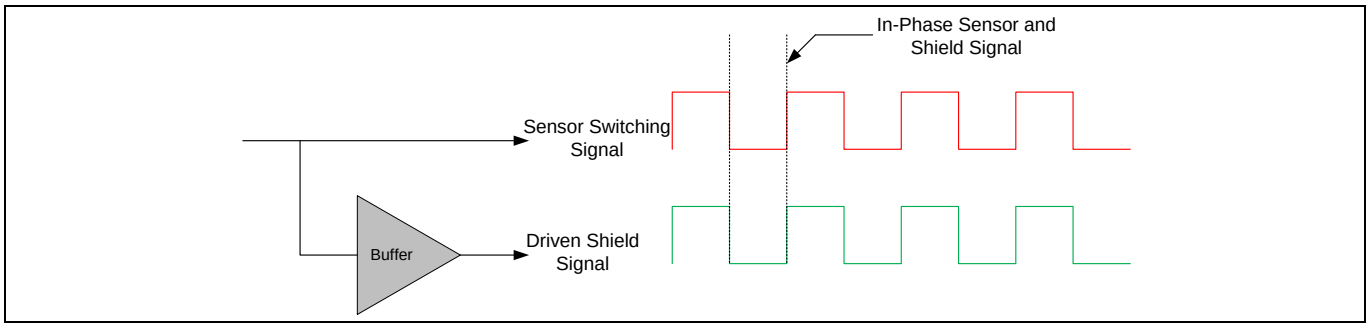

Figure 39 Driven-shield signal

2.9.3 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 40 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.

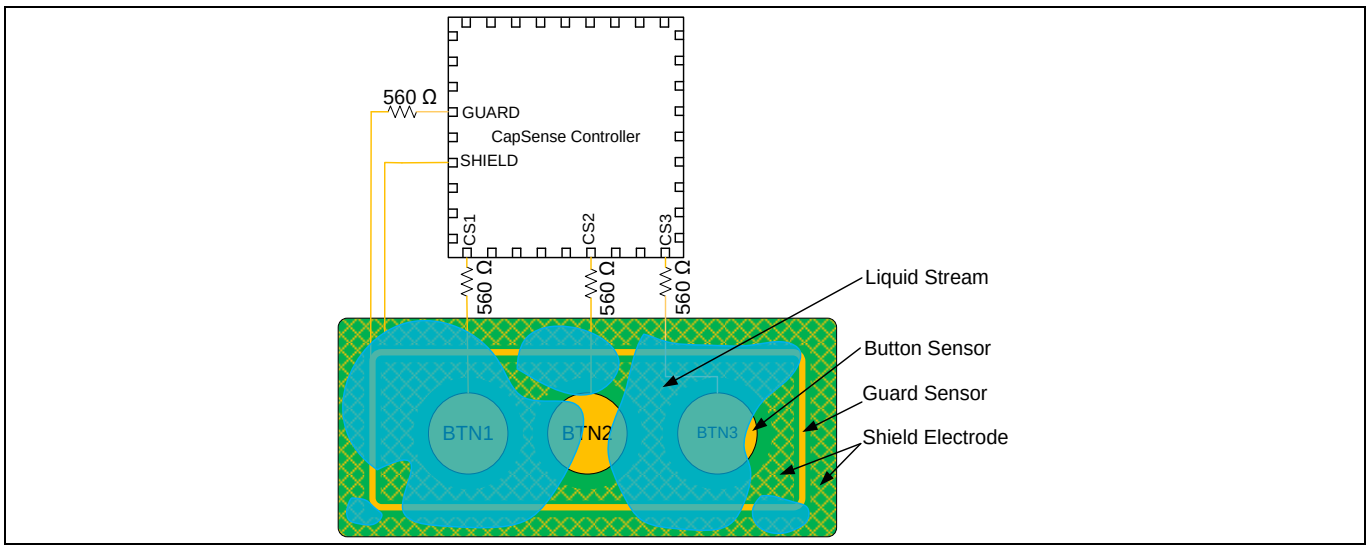

Figure 40 Capacitance measurement with a liquid stream

2.9.4 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 fall 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 a hot liquid droplet cools down immediately to room temperature when it falls on the touch surface.

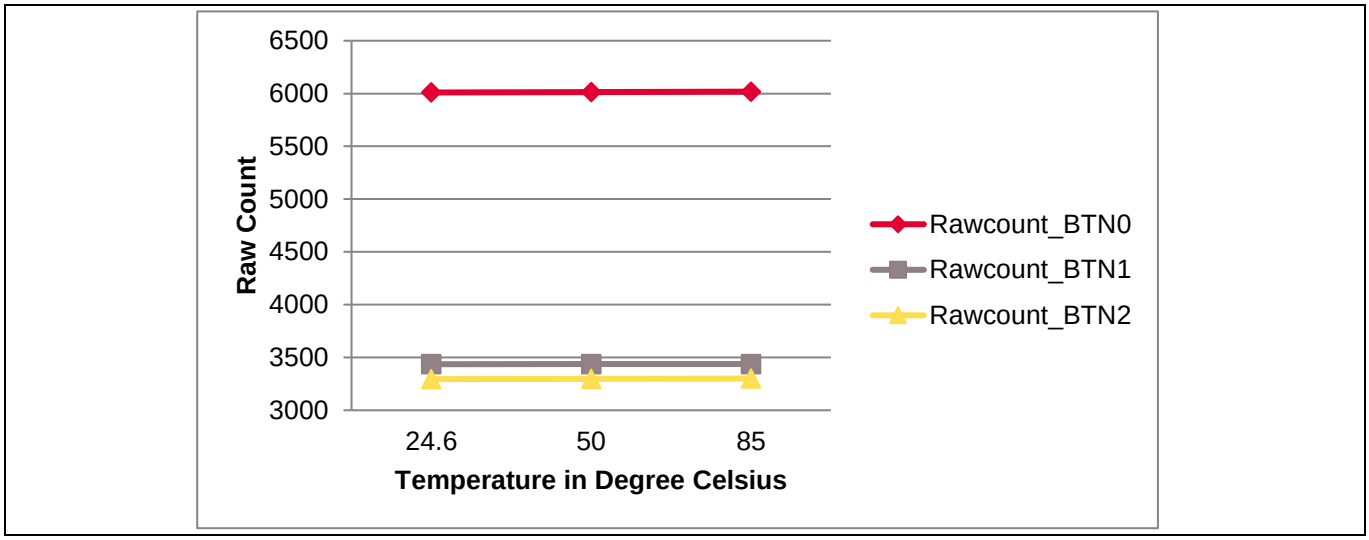

Figure 41 Raw count variation versus water temperature

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

1. Choose a CAPSENSE™ controller that supports the liquid tolerance feature. See the Capsense™ selector guide to select the CAPSENSE™ controller that supports the liquid tolerance feature.
2. Follow the schematic and layout guidelines explained in the device-specific Design Guide 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.

See 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 devices.

2.10 Proximity sensing

Proximity sensing is the process of detecting a nearby object without any physical contact. Proximity sensors use an 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, and 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. This section presents an introduction to CAPSENSE™-based proximity sensing. See AN92239 – Proximity sensing with CAPSENSE™ for the complete proximity design guidelines.

2.10.1 Proximity-sensing applications based on CAPSENSE™

Proximity sensing based on CAPSENSE™ is used in a variety of applications as explained below.

- **Face detection in mobile phones and tablets: Face** detection is a feature in mobile phones that disables the phone's touchscreen and dims the brightness of the display when a user answers a phone call, as Figure 42 shows. Face detection prevents false touches when the phone is placed on the ear and optimizes the device's power consumption. Using proximity sensing based on CAPSENSE™ in this application offers advantages over IR proximity sensing because it does not require cutouts in the overlay material, which reduces the tooling cost and improves the aesthetics of the end product.

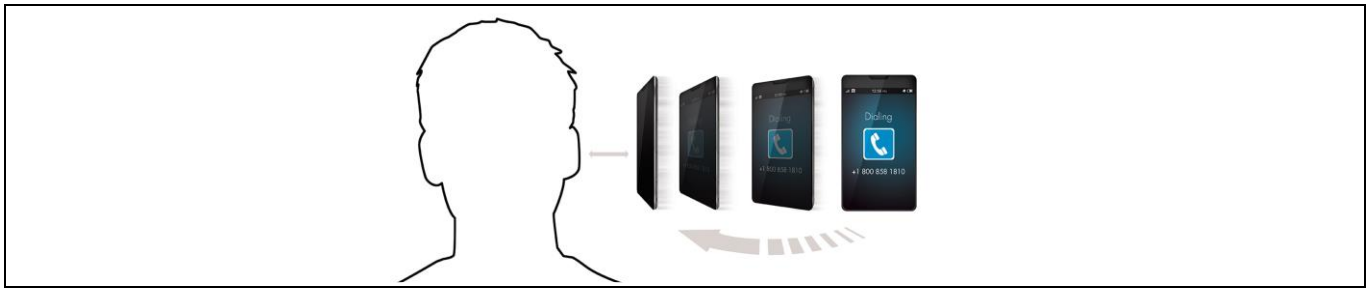

Figure 42 Face detection in mobile phones

- **SAR regulation in tablets and mobile phones:** SAR is a measure of the rate at which energy is absorbed by the human body when exposed to an RF electromagnetic field. Regulatory bodies, such as the Federal Communications Commission (FCC), require devices to limit the absorption of RF energy by reducing the device's RF transmit power when the device is in close proximity to the human body, as Figure 43 shows. Proximity sensors based on CAPSENSE™ can be used to detect the proximity of the human body and reduce RF power.

Figure 43 SAR regulation in tablets

- **Wake-on-approach:** This feature activates the system from the Sleep or Standby mode when an object approaches the system, as Figure 44 shows. Wake-on-approach is also used to control the backlight LEDs when the user approaches the system. This feature reduces system wakeup time, improves device responsiveness, reduces device power consumption, and improves aesthetics. It is useful in battery-operated applications such as mice and keyboards.

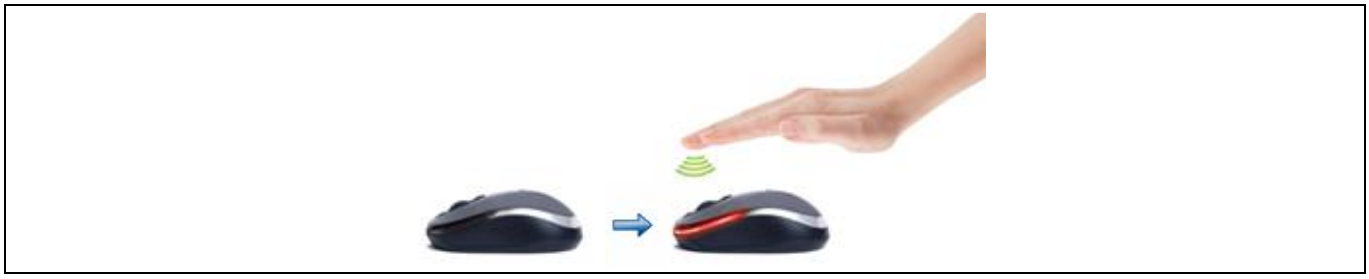

Figure 44 Wake-on-approach implemented in a mouse

- **Gesture detection:** Gesture detection is the technique of interpreting human body movements and providing gesture-type information to the device. Gesture-based user interfaces provide an intuitive way for the user to interact with the system, improving the user experience. Gesture detection is used in applications such as laptops, tablets, and mobile phones for controlling the user interface.

Proximity sensors based on CAPSENSE™ can be used in these applications to detect gestures without any physical contact between the user and the device. Figure 45 shows an example of an implementation of gesture detection in a laptop where proximity sensors placed near the trackpad are used for the pan movement of the onscreen map.

Figure 45 Gesture detection implementation in a laptop

- **IR replacement:** Proximity sensors based on CAPSENSE™ can replace IR proximity sensors in applications such as faucets and soap dispensers, as Figure 46 shows. Proximity sensing based on CAPSENSE™ offers the following advantages over IR proximity sensing:
  - It is a low-cost solution compared to IR proximity sensing. Proximity sensors based on CAPSENSE™ can be constructed using a copper trace on the PCB, whereas IR proximity sensing requires extra IR sensors
  - Proximity sensors based on CAPSENSE™ do not require any cutout in the overlay material to detect proximity sensing, unlike IR proximity sensors. Therefore, proximity sensing based on CAPSENSE™ reduces the tooling cost and improves the aesthetics of the end product
  - They consume less power than IR proximity sensors
  - They are immune to ambient light, whereas IR-based proximity sensors can have performance issues with varying ambient light

Figure 46 Proximity sensing based on CAPSENSE™ in a soap dispenser

2.10.2 Proximity sensing with CAPSENSE™

The proximity-sensing technique based on 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 can be constructed using a conductive (usually copper or indium tin oxide) pad or trace laid on a nonconductive material like PCB or glass. In essence, a proximity sensor is like any other sensor but designed with very minimum ground near the sensor and tuned for maximum sensitivity.

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. See Proximity sensing design to learn the various ways of constructing a proximity sensor and to learn the various parameters that affect the proximity distance.

2.11 User interface feedback

Effective user interface designs include feedback 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.

2.11.1 Visual feedback

LEDs and LCDs provide visual feedback.

2.11.1.1 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 as listed below.

- LED ON/OFF – GPOs are used to drive LEDs in either a sourcing (GPO supplies current to the LED) or sinking (GPO sinks current from the LED) configuration
- LED Brightness Control – 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 adjust the LED brightness. This enables adjusting your user interface brightness in response to ambient lighting conditions
- LED Fading – By gradually changing the duty cycle between LED states, you can achieve a fading effect. For example, the LED appears to “fade in” (from OFF to ON) when the duty cycle is increased in a series of small steps
- LED Breathing – Gradually increasing and decreasing the duty cycle between two levels on a continuous basis makes the LED appear to “breathe”. 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

2.11.1.2 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. Depending on the device family, programmable devices support different types of LCD technologies with pre-built components and user modules which provide APIs for displaying data with ease.

- **Character LCD with parallel interface:** PSoC™ devices support interface with the Hitachi HD44780A LCD module. Figure 47 shows the typical connection for using the Hitachi HD44780A LCD module. See the Character LCD component datasheet to learn more

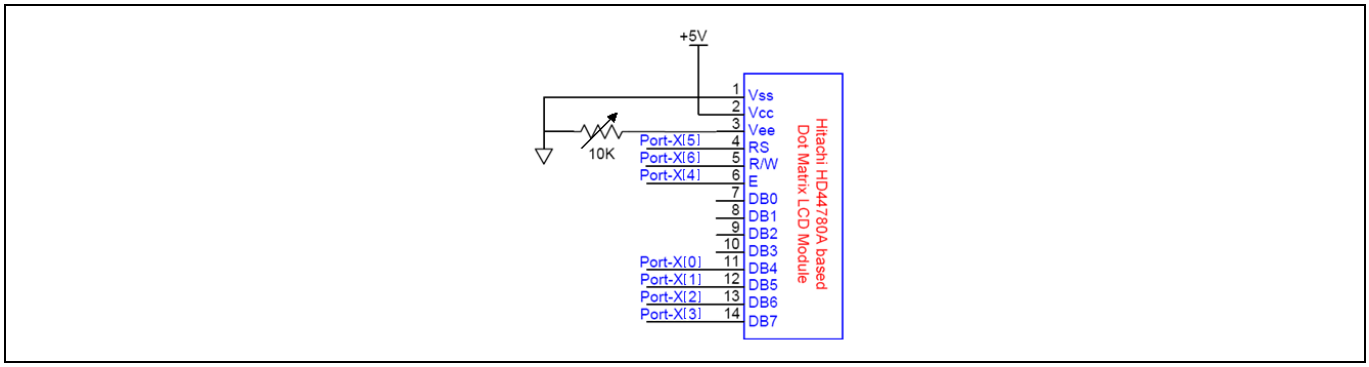

Figure 47 Hitachi dot matrix LCD pin connections

- **Character LCD with I²C interface:** PSoC™ can also control LCDs through I2C with support for the NXP PCF2119x command format. Figure 48 shows the typical circuit diagram for driving an LCD with I2C interface. See the Character LCD with I2C interface (I2C LCD) component datasheet to learn more

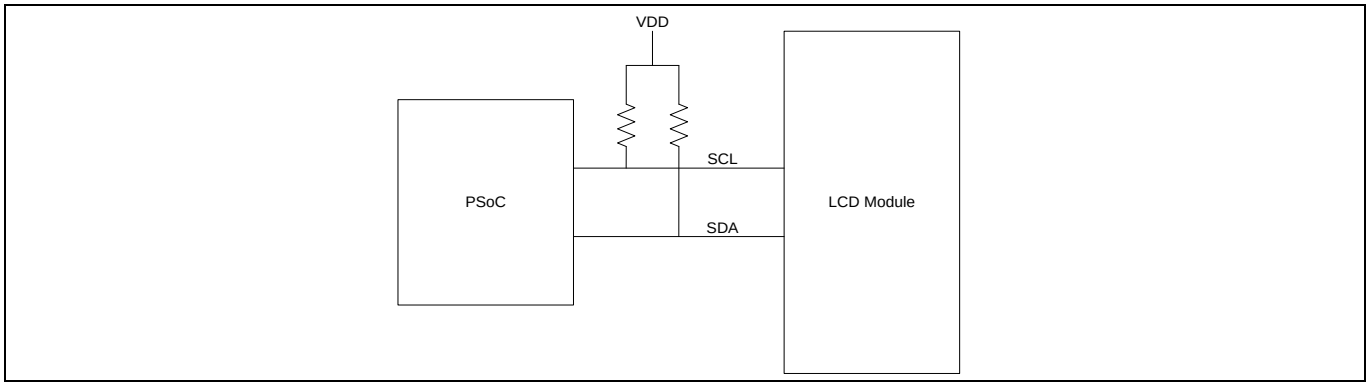

Figure 48 Interfacing an LCD with I2C interface

Segment LCD glass: PSoC™ devices integrate the LCD driver to directly drive a segment LCD glass. See the Capsense™ selector guide to know the LCD drive capability of different PSoC™ devices

2.11.2 Haptic feedback

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

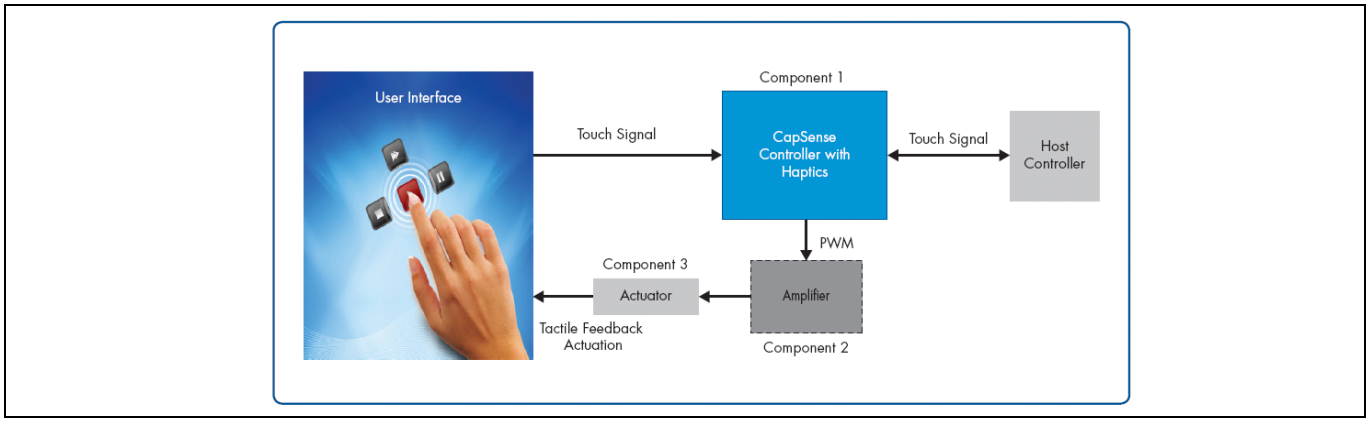

Figure 49 Infineon haptics ecosystem

Different haptic effects are created by varying the duration, frequency, and shape of the vibration profile. Vibrations are created by an actuator. The key requirements for an actuator are – Response time, Power consumption, Size, Form factor, Durability, and Vibration strength. Multiple actuator options are available with varying drive requirement. The four types of actuators are:

- Eccentric Rotation Mass (ERM) actuators – These actuators are most cost-effective, have a current requirement of 130 to 160 mA, and require an external power amplifier for drive. These are not suitable for applications requiring fast response such as typing
- Linear Resonant Actuators (LRAs) – LRAs are used in many smartphones, require AC input, draw current at around 65 to 70 mA, are relatively faster, but cost more than ERM actuators. LRAs require an external power amplifier for drive
- Piezo Modules – Piezo modules respond fast, making them suitable for typing applications, but they cost more than ERM and LRA actuators. Though their instantaneous current draw is more at around 300 mA at 3-V supply, their average current consumption is not more than the ERM and LRA actuators

- Electro-Active Polymer Actuators (EAPs) – EAPs offer a performance similar to that of the piezo modules but integrating them into tight spaces is a challenge. They require 800 V for a drive signal

The CY8C20XX6H CAPSENSE™ controller integrates Immersion's TS2000 Haptic effects library for Eccentric Rotating Motor (ERM) drive control and can generate 14 predefined haptic effects. The Haptics user module datasheet has more information about these effects and also provides a code example. To know more about CY8C20XX6H, see the device CAPSENSE™ applications datasheet.

2.11.3 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. 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.

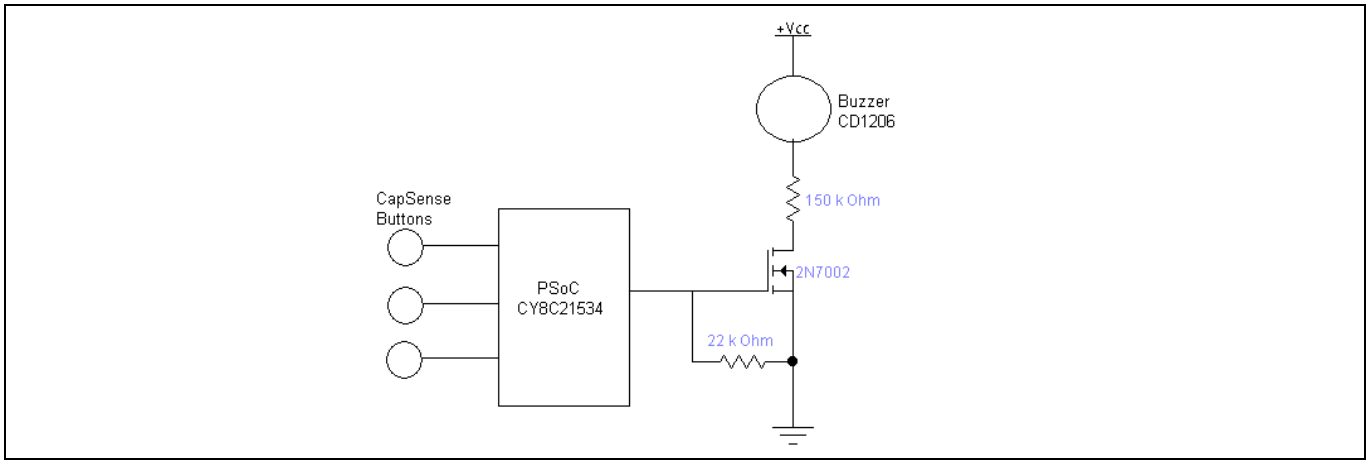

Figure 50 Implementing audible feedback for CAPSENSE™

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 Self-capacitance, shows the finger capacitance.

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

Equation 11

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. Materials with dielectrics between 2.0 and 8.0 are well suited to capacitive sensing applications.

Table 3 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 51.

Figure 51 Sensitivity versus overlay thickness

Both signal and noise are affected by the overlay properties. Table 4 lists the recommended maximum overlay thicknesses for PSoC™ CAPSENSE™ applications with an acrylic overlay material.

Table 4 Maximum overlay thickness with an acrylic overlay material

Design element	Maximum 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 a 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 any damage to the CAPSENSE™ controller.

Table 5 Overlay material dielectric strength

Material	Breakdown voltage (V/mm)	Minimum 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 5 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 52, if L1 and L2 are greater than 10 mm, the system will withstand 12 kV.

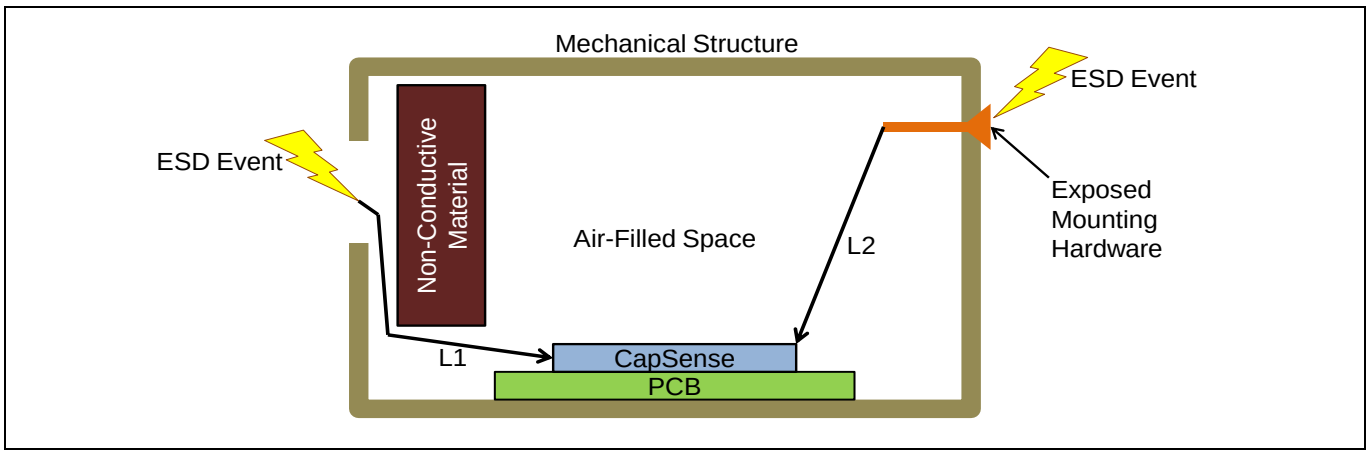

Figure 52 ESD path

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 5 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.

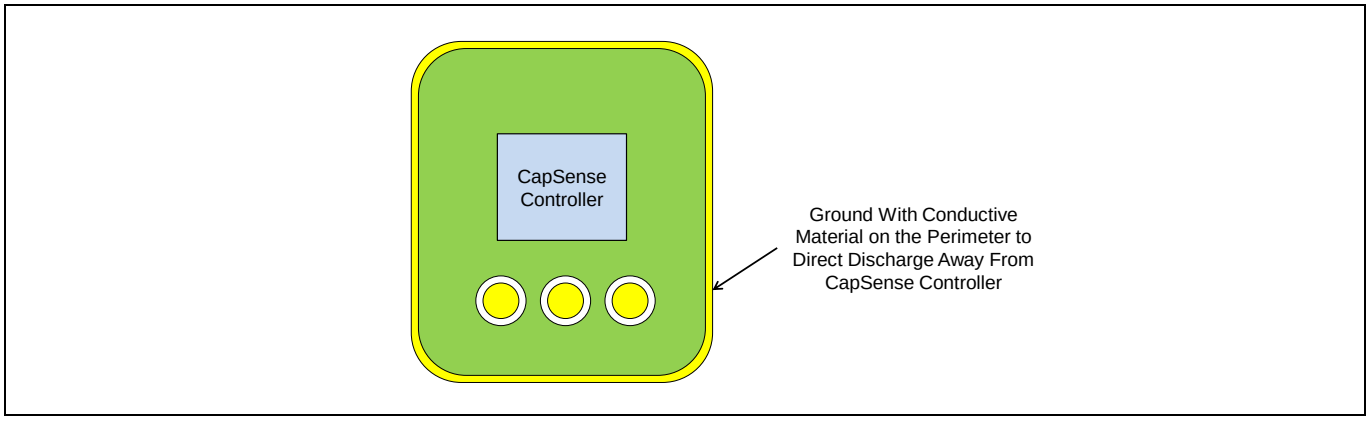

Figure 53 Guard ring

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 Series resistor.

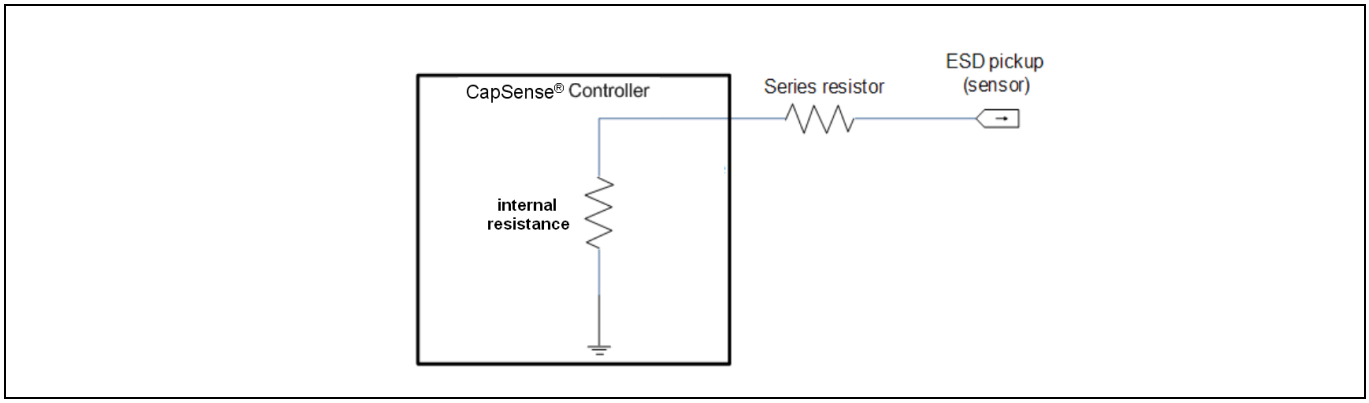

Figure 54 ESD protection using a series resistor

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 6 lists devices recommended for use with CAPSENSE™ controllers.

Table 6 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

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.

Design considerations

Series resistor

Every CAPSENSE™ controller pin has some parasitic capacitance (CP) 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 55) that can significantly reduce RF emissions. Thus, series resistors help in eliminating higher-order harmonics and attenuating the RF interference and emission.

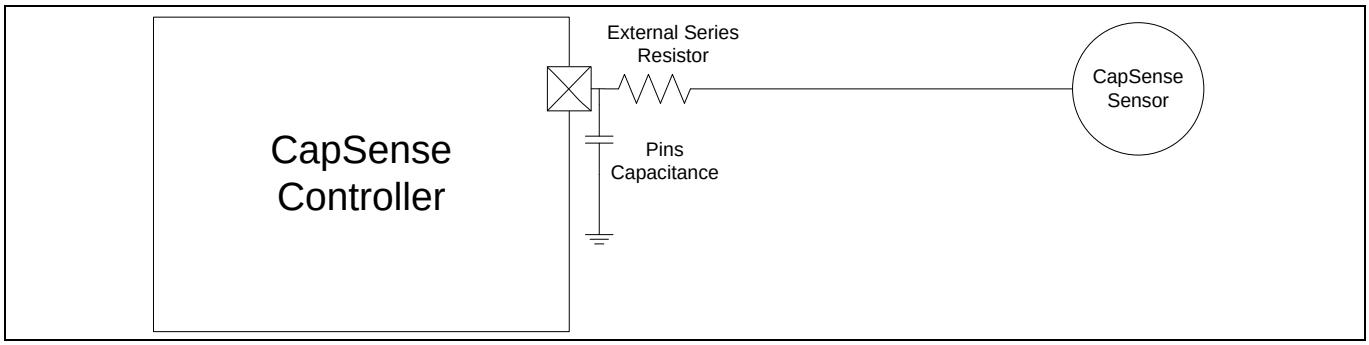

Figure 55 RC filter

Series resistors should be placed close to 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.

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 56 shows an equivalent circuit and resulting waveforms. Adding a resistance changes the time constant of the switched-capacitor circuit that converts CP 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.

Design considerations

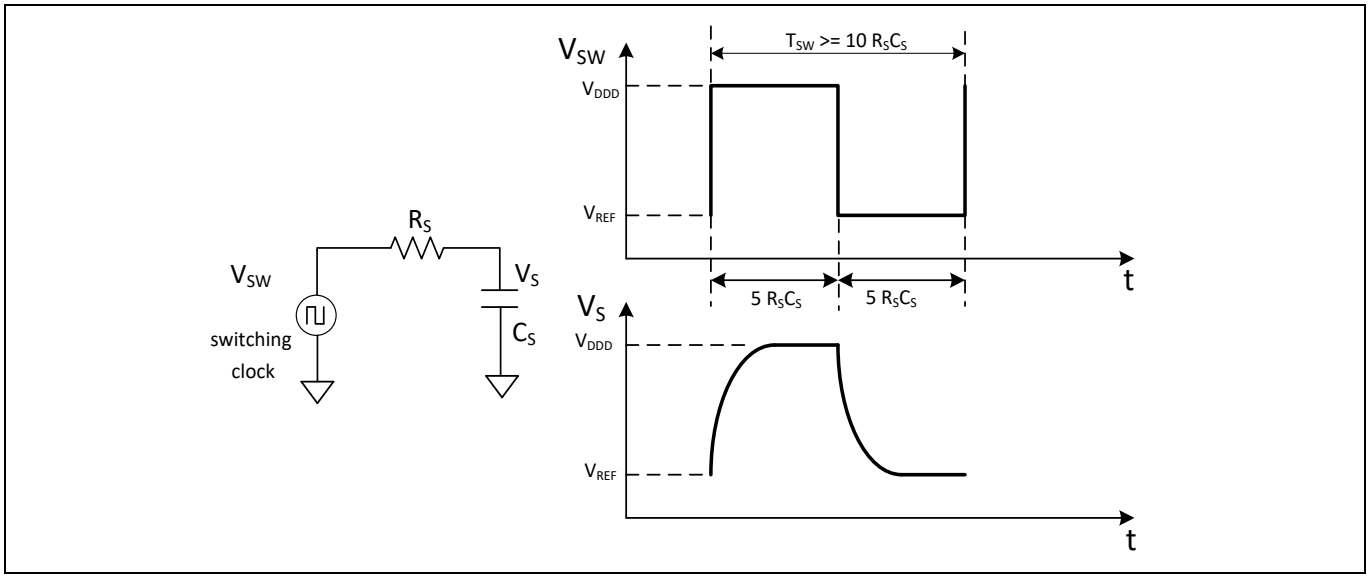

Figure 56 Equivalent circuit and waveforms

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:

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

Equation 12

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

Equation 13

Digital communication lines

Communication lines, such as I2C 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 value 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 I²C operation because the V_{IL} level still remains within the I²C specification limit of 0.3 V_{DD} when the PSoC™ device outputs a LOW.

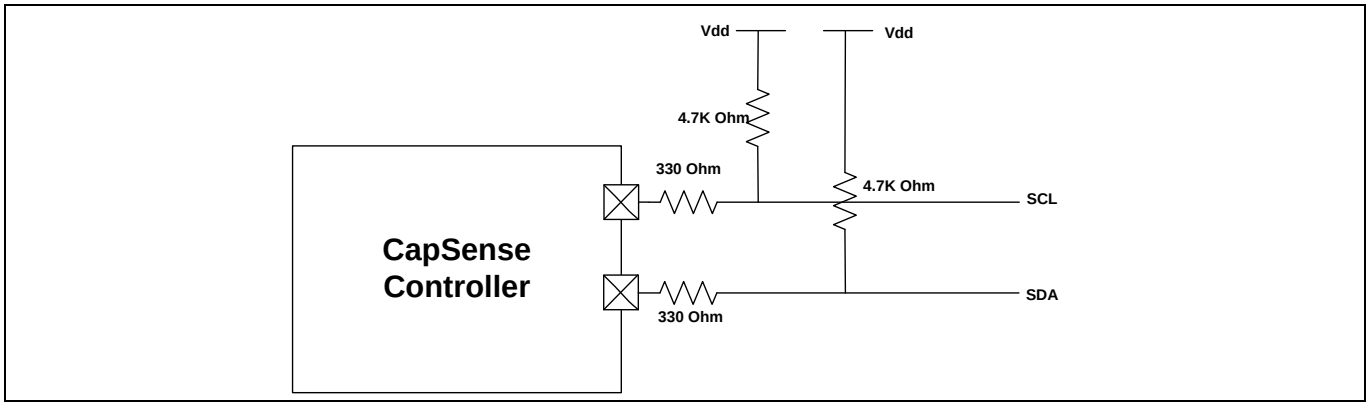

Figure 57 Series resistors on communication lines

Trace length

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

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. 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 mutual inductance. 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 58 shows an example of an improper grounding scheme. The layout greatly improves by reducing the loop area.

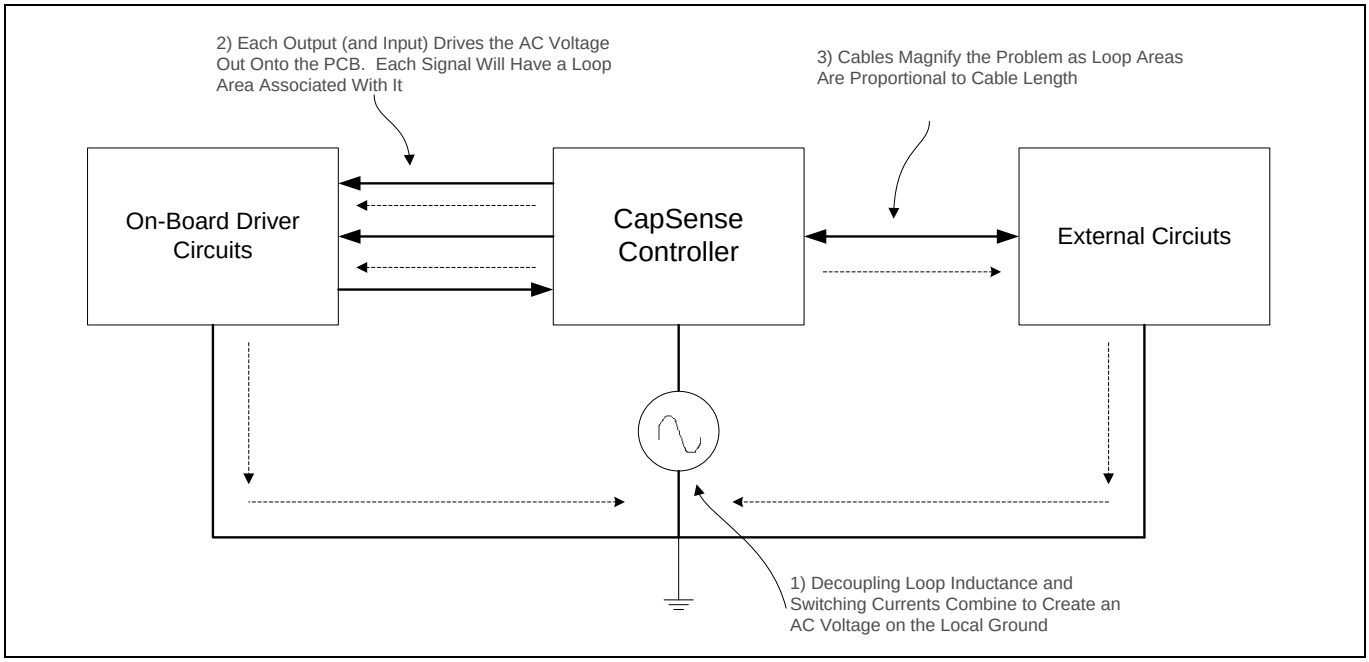

Figure 58 Improper ground scheme and ground loop

In Figure 59, two sensors are surrounded by a ground plane that is connected to a 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 another 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 in the same location on the schematic, so they can potentially be one connected area with a better layout.

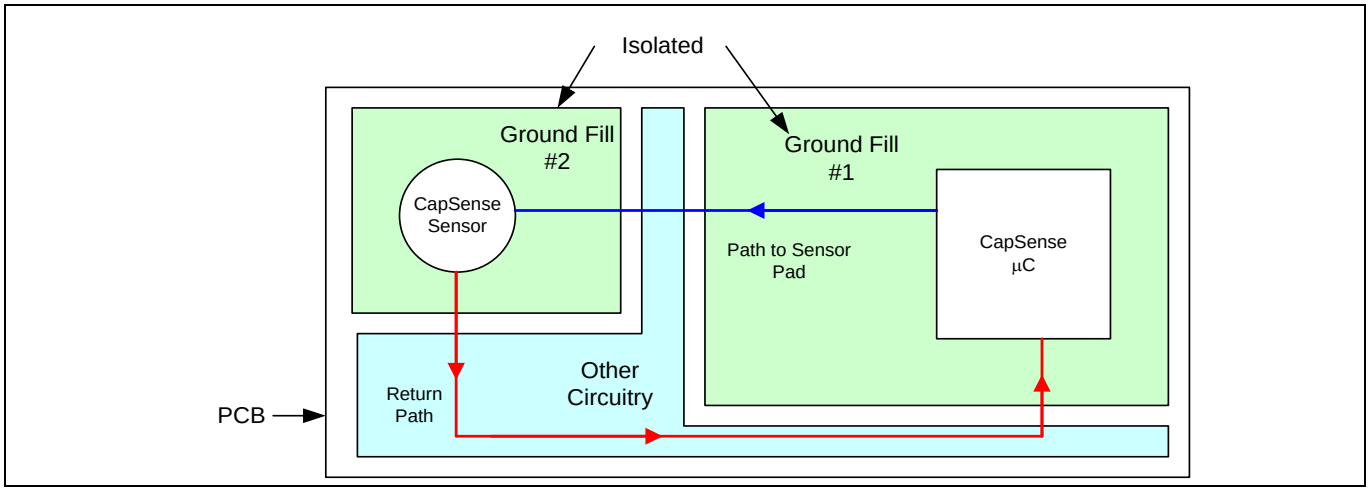

Figure 59 Improper current loop layout

Figure 60 illustrates the proper layout for the previous example. The loop area is reduced by connecting the two grounded areas.

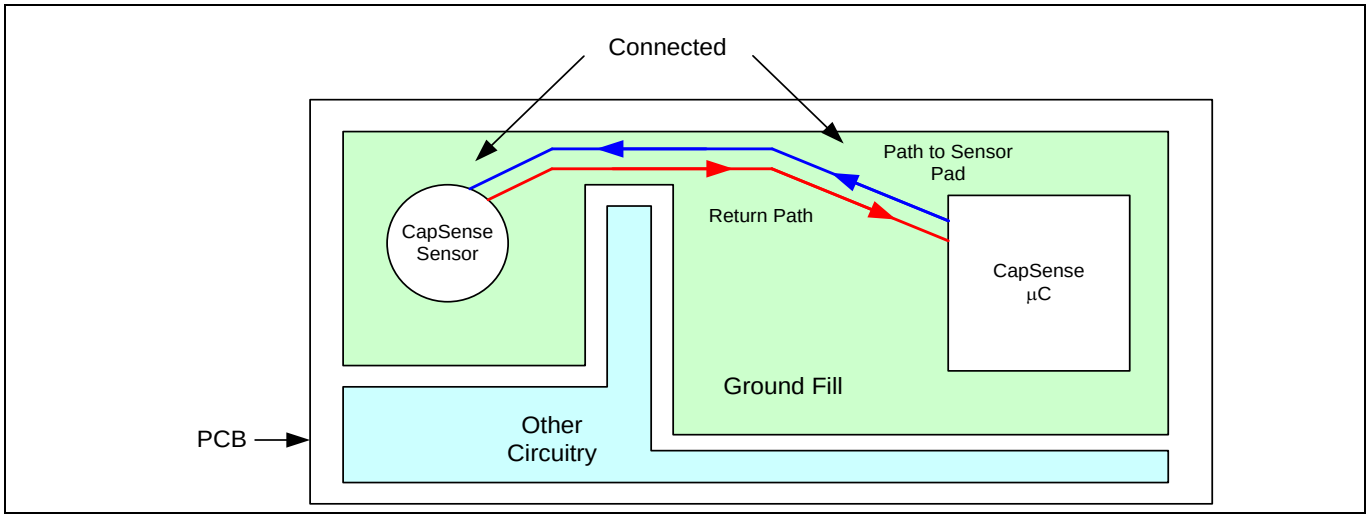

Figure 60 Proper current loop layout

Frequency hopping

Frequency hopping is a method of spreading the input or operating frequency over a narrow band of frequencies. This method helps to reduce the peaks and spread out the emissions as well as the interference over a range of frequencies as shown in Figure 61. The following are the methods of frequency hopping in PSoC™:

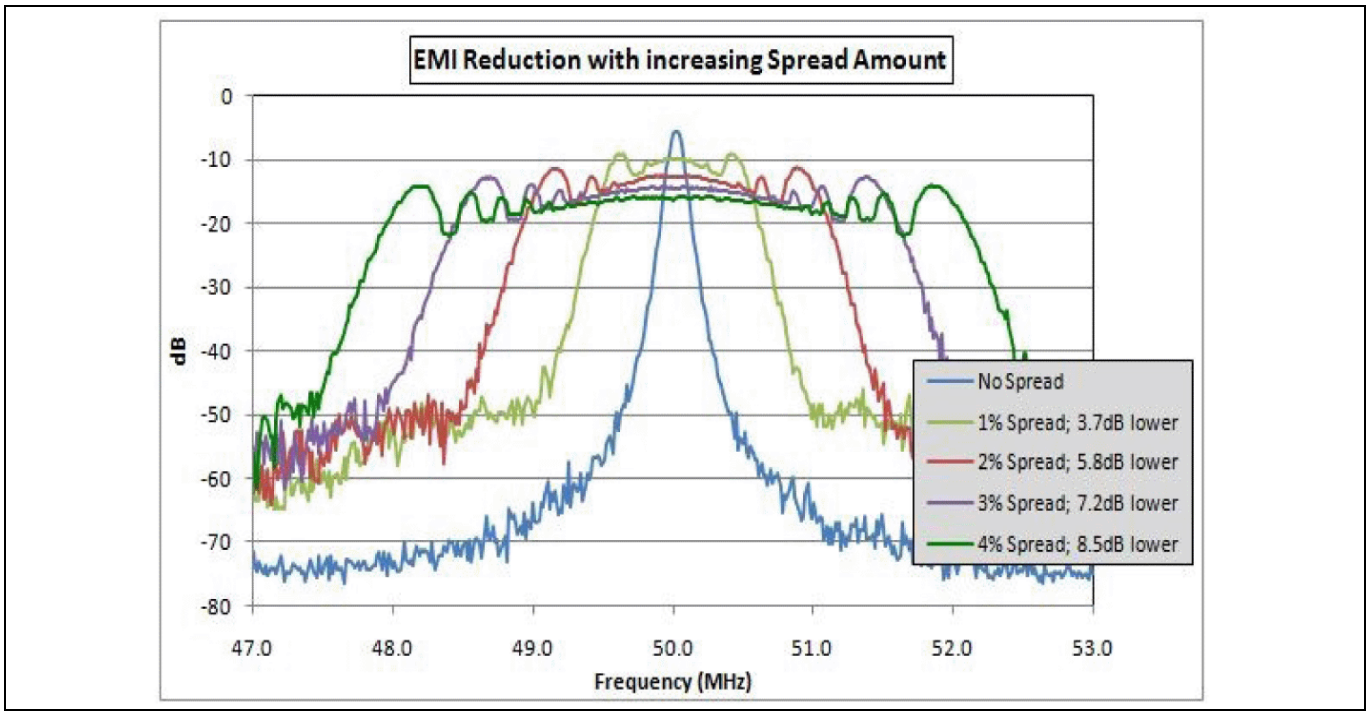

Figure 61 Frequency hopping

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 (SSC): 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. PSoC™ 1 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

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 62. 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™.

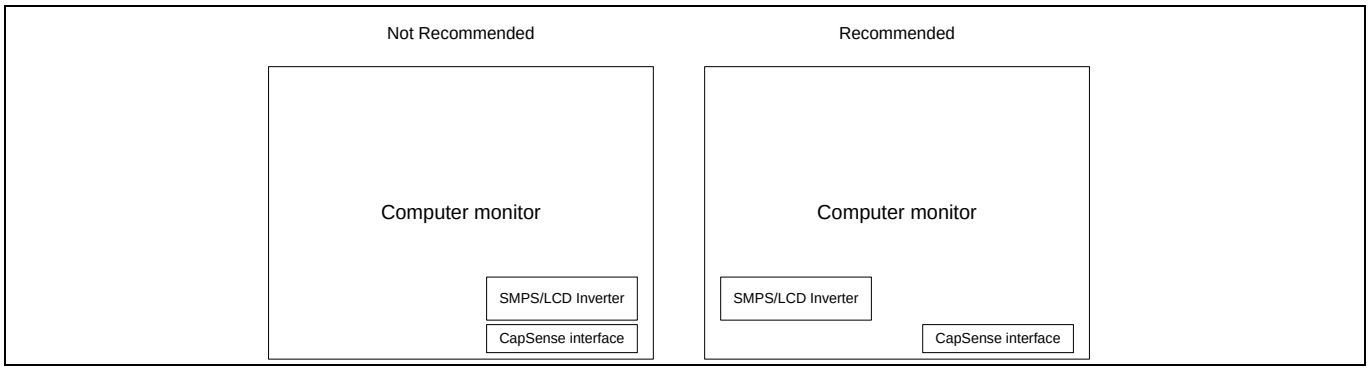

Figure 62 Separating noise sources

EMC feature

CAPSENSE™ User Modules and components with the EMC feature implement 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.

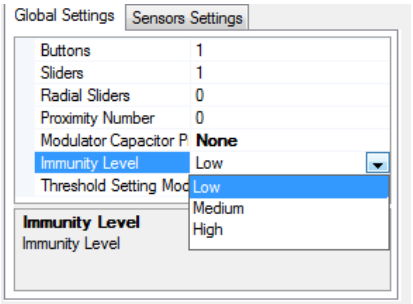

Figure 63 Immunity level selection

3.3.1.3 Radiated emissions

Operating voltage

For devices in which sensors switch between an operating voltage and a reference voltage, such as CY8C21x34, reducing the operating voltage helps to reduce emissions to a great extent. This is because the amplitude of the switching signal at any pin is dependent on the device's operating voltage and the emissions are directly proportional to voltage levels at which the switching happens. Figure 64 shows the impact of operating voltage on radiated emissions.

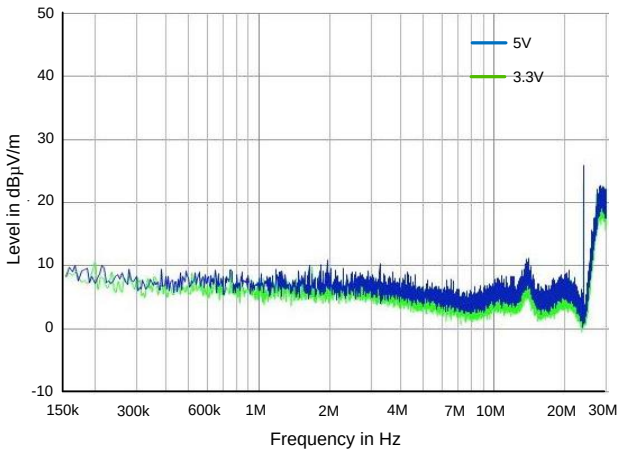

Figure 64 Impact of operating voltage on emissions

System oscillator frequency (IMO)

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

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 65 shows the impact of the sensor-switching frequency.

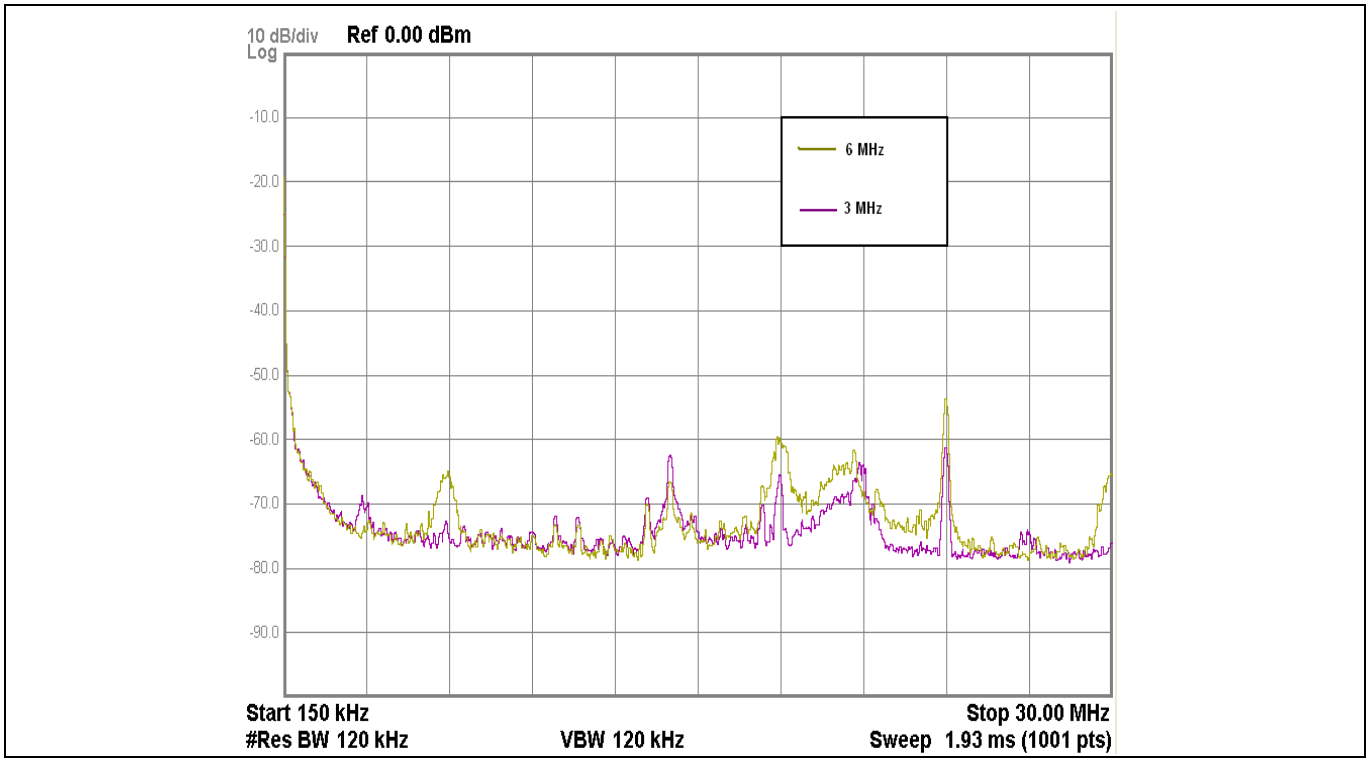

Figure 65 Impact of switching frequency

Slew rate control

Figure 66 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.

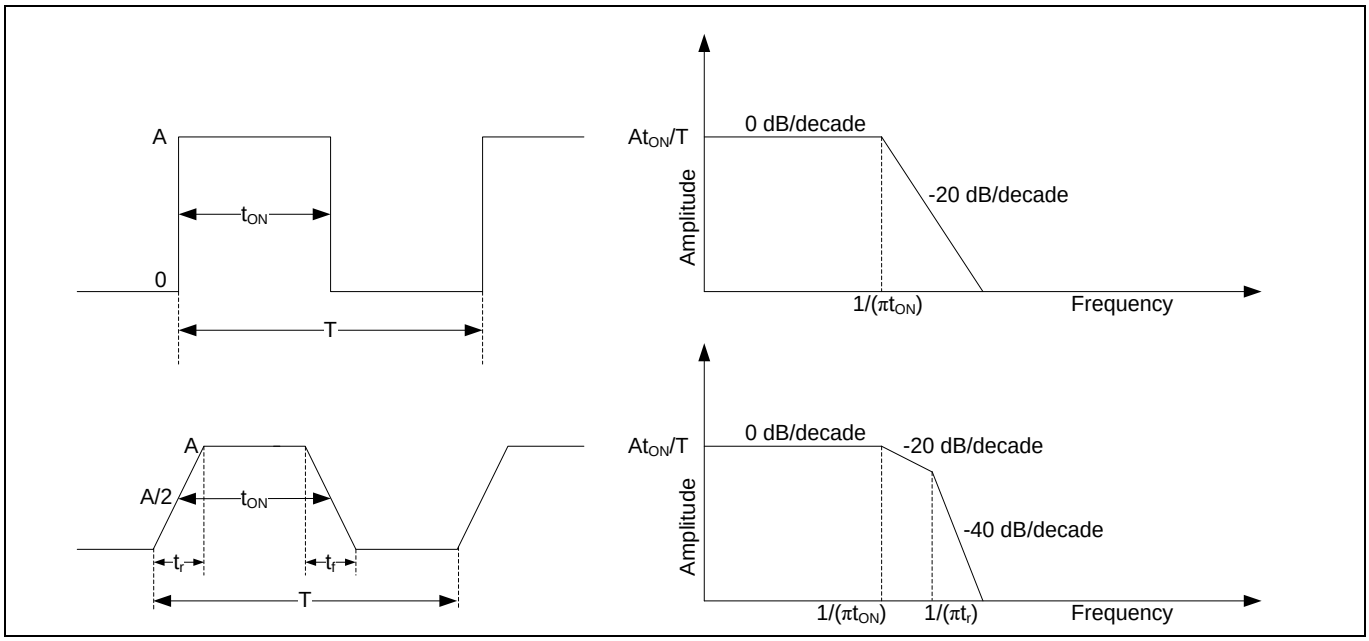

Figure 66 Impact of slew rate on emissions

Sensor Scan time

The scan time of sensors impacts radiated emissions. Figure 67 shows the impact of the sensor-scan time on radiated emissions. An increase in the sensor-scan time results in increased emissions. Table 7 shows the parameter settings and associated sensor scan times.

Table 7 Sensor scan time

Parameter	Value	
Scan resolution	8 bits	10 bits
Individual sensor scan time	0.021 ms	0.085 ms
Total scan time for five buttons	0.105 ms	0.425 ms

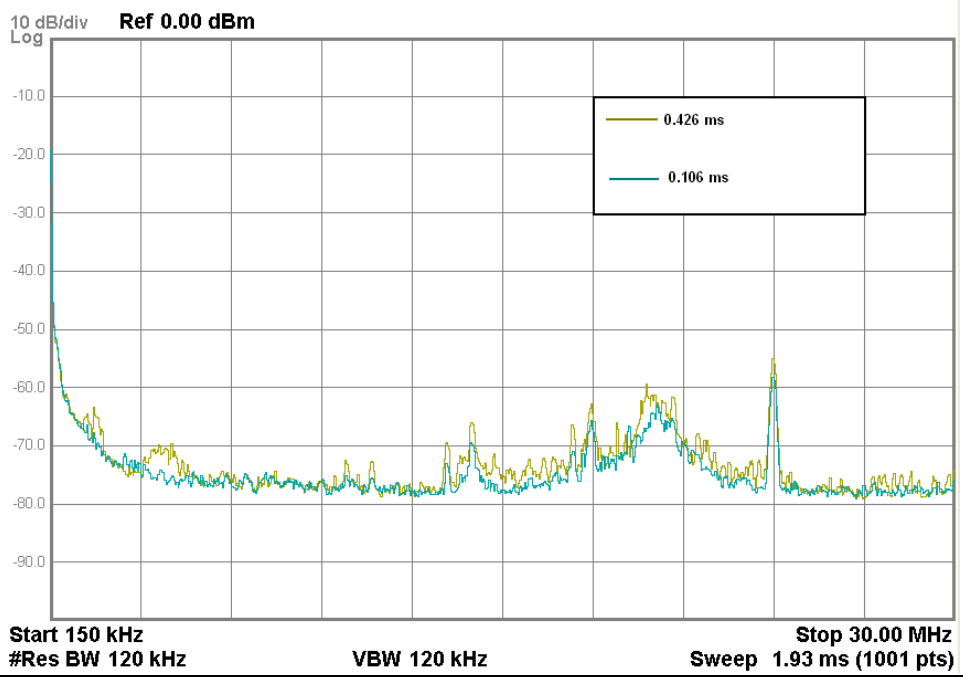

Figure 67 Effect of scan time

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 68 shows the emissions with and without the driven-shield signal:

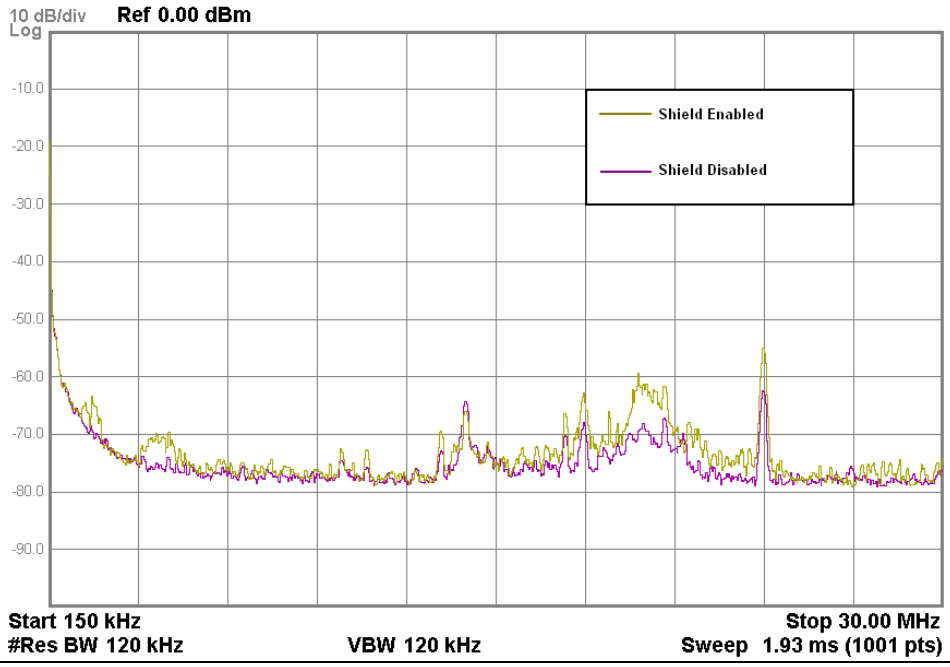

Figure 68 Impact of shield on emissions

You can reduce emissions by the following techniques:

1. Reduce the size of the shield hatch fill to have a maximum of 10 mm from the sensors. See 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 scanned, and only when the sensors that need shield protection are scanned.
3. Limit the placement of the shield to selected sensors only. Do not spread the shield around sensors that do not need shield protection.
4. Slow the edges of the shield waveform by one of the following means:
   - Add a capacitor filter between the shield electrode port pin and ground
   - In most of the PSoC™ 1 CAPSENSE™ devices, the slew rate of the shield signal can also be controlled by changing the Drive mode of the shield pin from Strong to Strong Slow as shown in Figure 69.

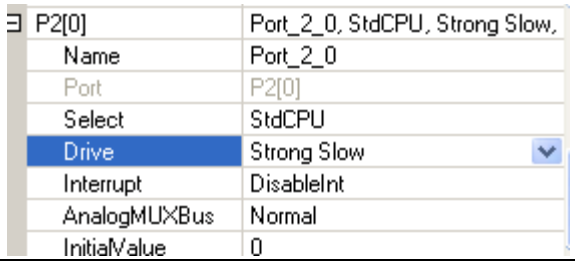

Figure 69 Drive mode selection for shield

Add 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. Ensure that the shield electrode gets charged and discharged completely when you add filters. See the CAPSENSE™ input lines section for charge and discharge waveforms, and to see how to choose the right value for series resistors.

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 70 shows the impact of different inactive-sensor terminations on radiated emissions.

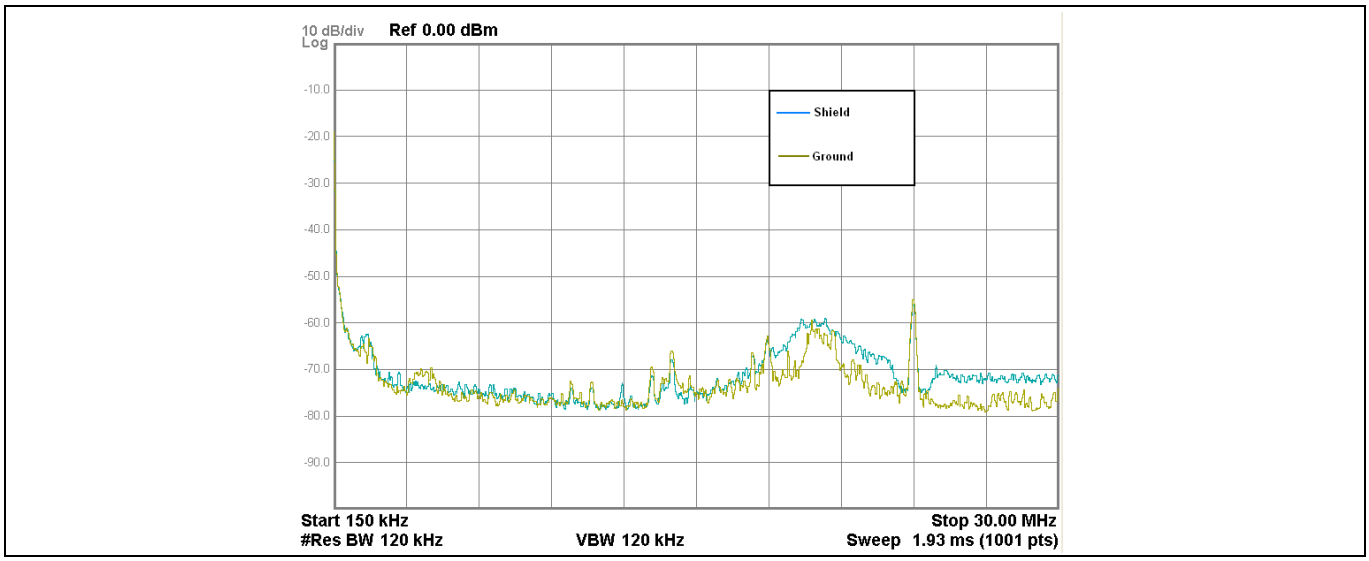

Figure 70 Effect of inactive sensor termination

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, see the Power supply layout recommendations section. 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 71. 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 to the power planes.

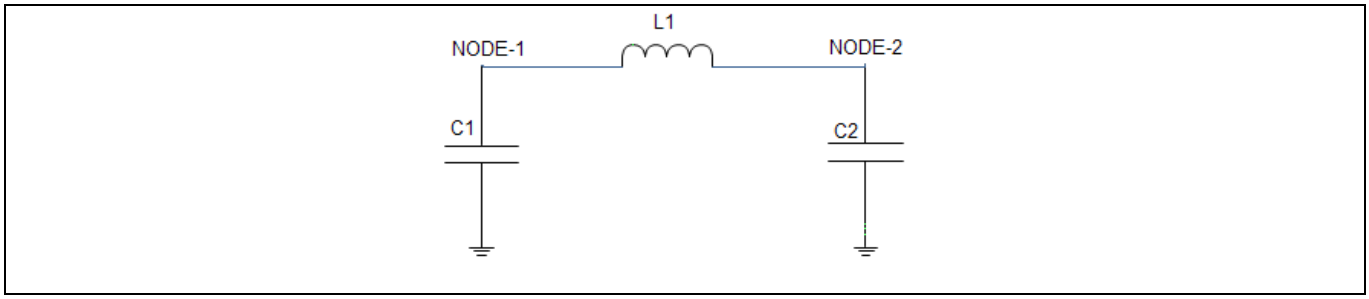

Figure 71 Series inductor Pi-filter

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 the 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, see the following documents:

- Top 10 EMC design considerations
- AN2155 - PSoC™ EMI design considerations
- AN80994 - Design considerations for Electrical Fast Transient (EFT) Immunity

3.4 Software filtering

Software filters are one of the techniques of dealing with high levels of system noise. Table 8 lists the types of filters that are useful for CAPSENSE™.

Table 8 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 previous input	Noise from thick overlay (SNR < 5:1), especially useful for 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 72 shows a sample rate that is synchronized with a simple periodic waveform. There is no feedback path in this filter.

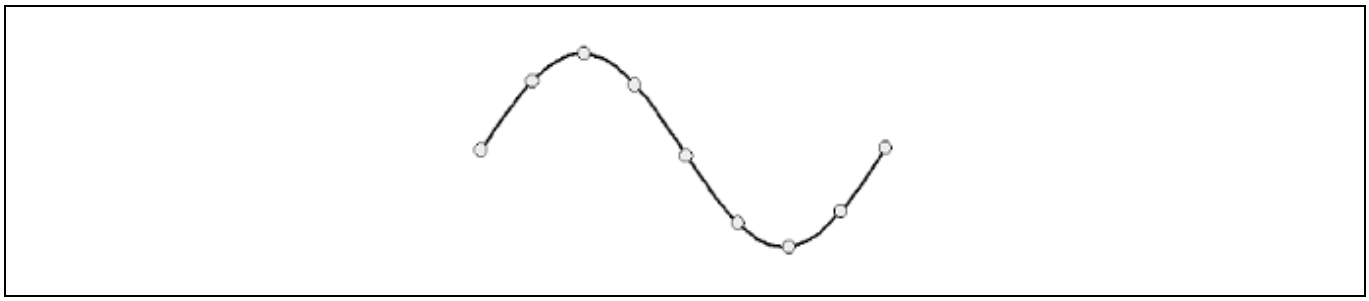

Figure 72 Synchronized sample rate

The general equation for an average filter is:

$$y[i] = \frac{1}{N}(x[i] + x[i-1] + \cdots + x[i-N+1])$$

Equation 14

Design considerations

Figure 73 and Figure 74 illustrate the results of using an average filter on real CAPSENSE™ data using the 16-sample filter equation:

$$y[i] = \frac{1}{16}(x[i] + x[i-1] + \cdots + x[i-15])$$

Equation 15

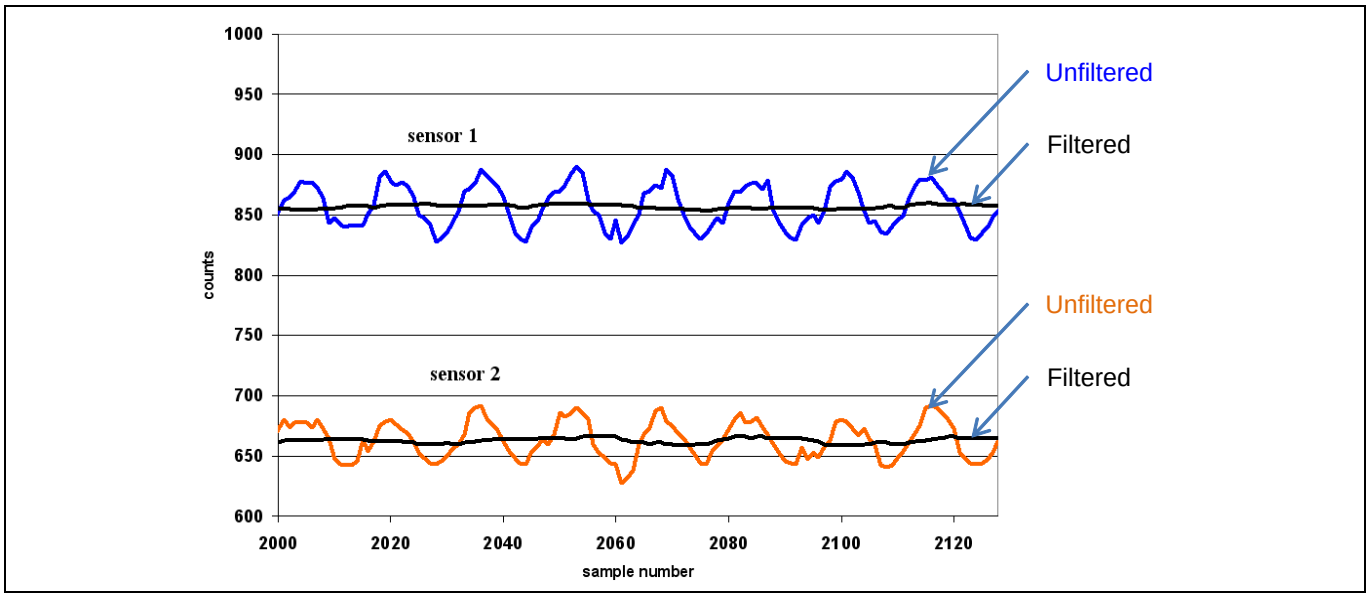

Figure 73 Average filter noise (16 samples)

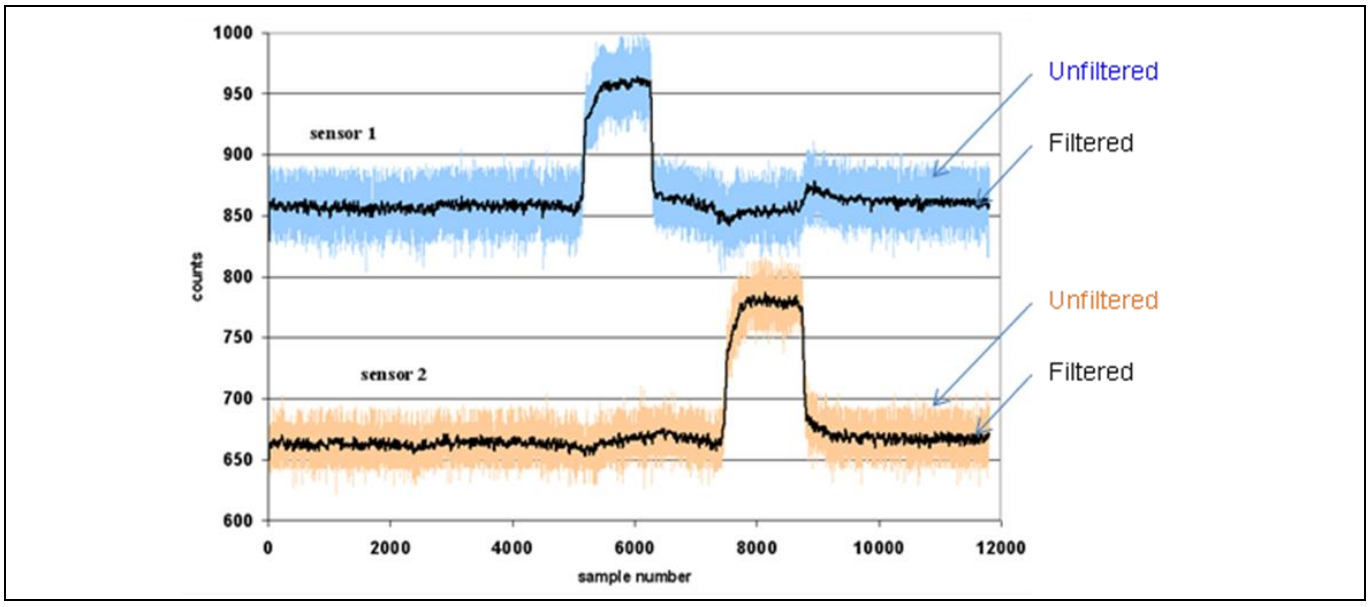

Figure 74 Average filter finger touch (16 samples)

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).

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.

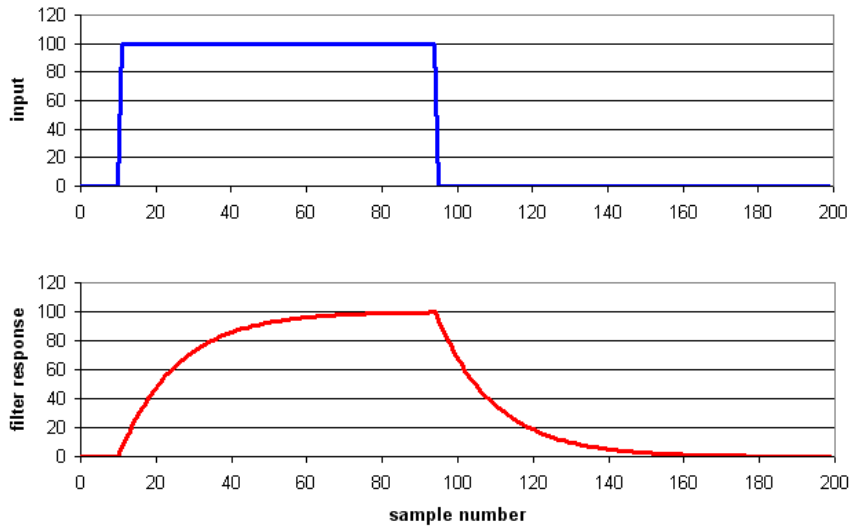

Figure 75 IIR filter step response

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 16

Figure 76 and Figure 77 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 17

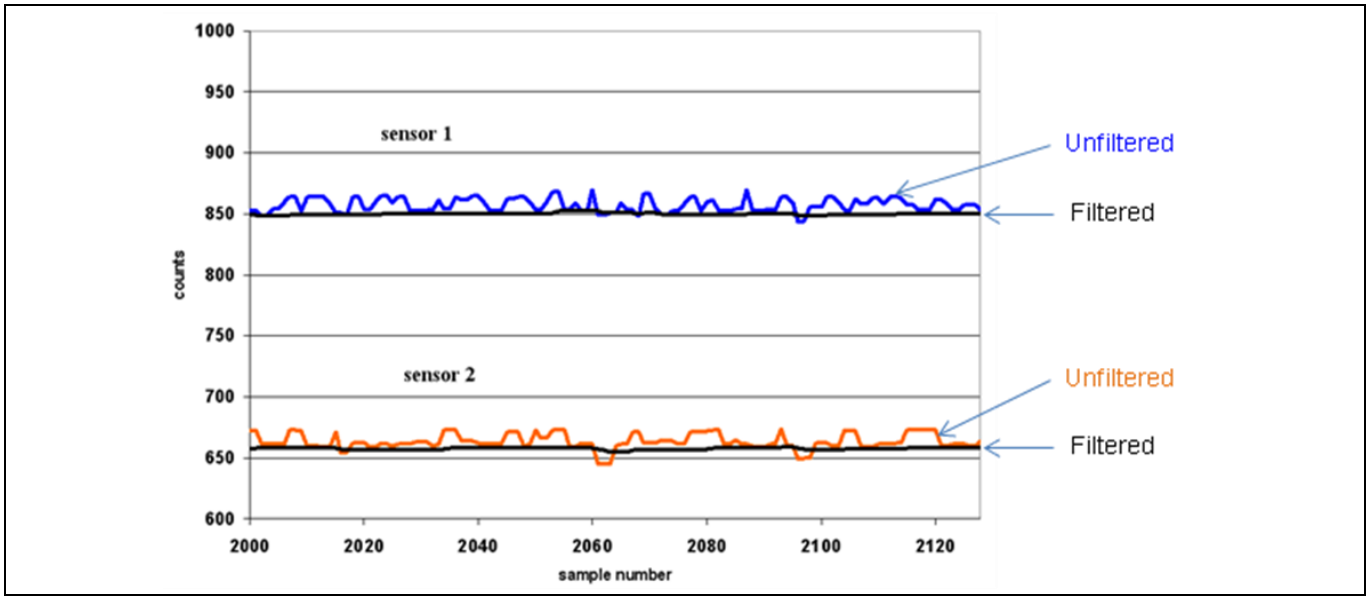

Figure 76 IIR filter noise

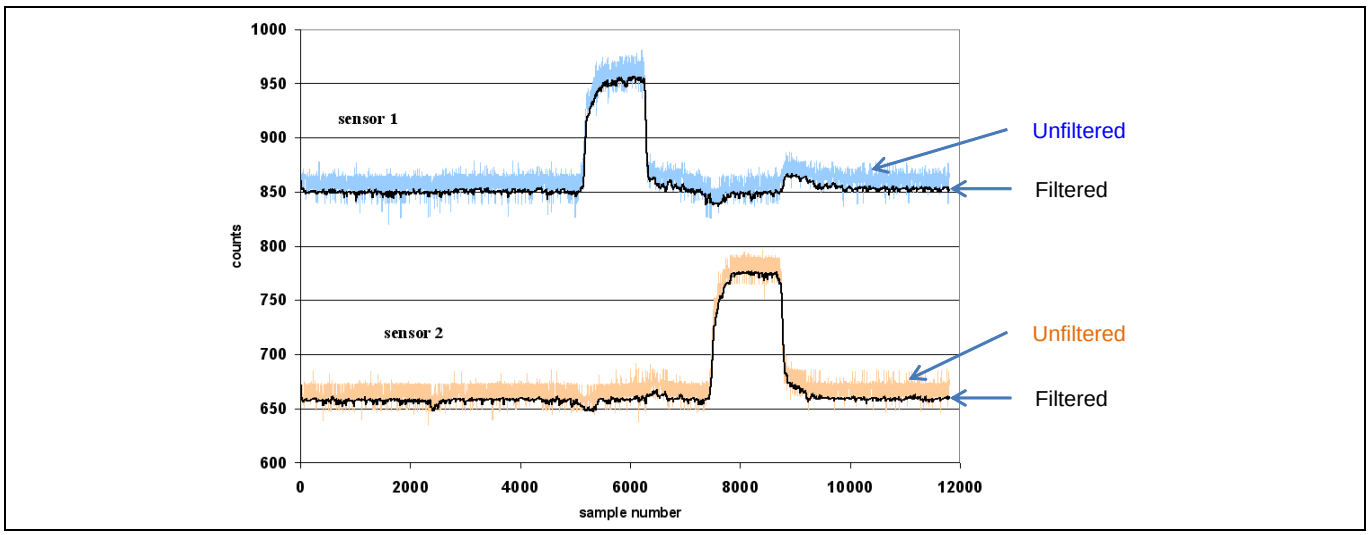

Figure 77 IIR filter finger touch

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:

$$y[i] = \text{median}(x[i], x[i-1], \ldots, x[i-N+1])$$

Equation 18

Figure 78 and Figure 79 show the results of a median filter on real CAPSENSE™ data using the general filter equation with N = 16.

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

Equation 19

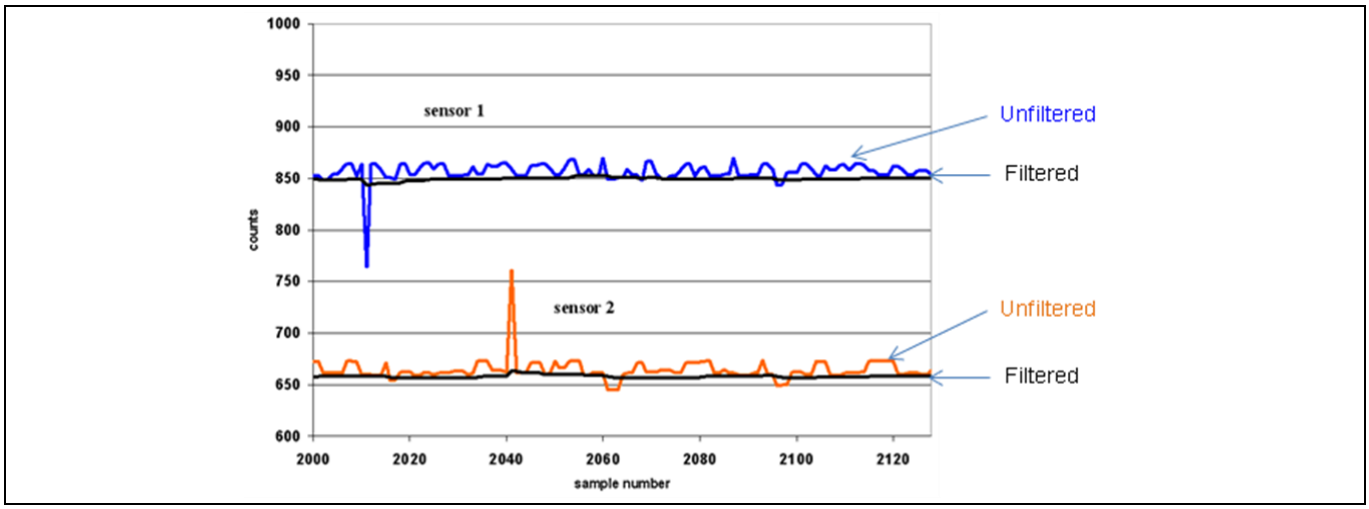

Figure 78 Median filter noise spike

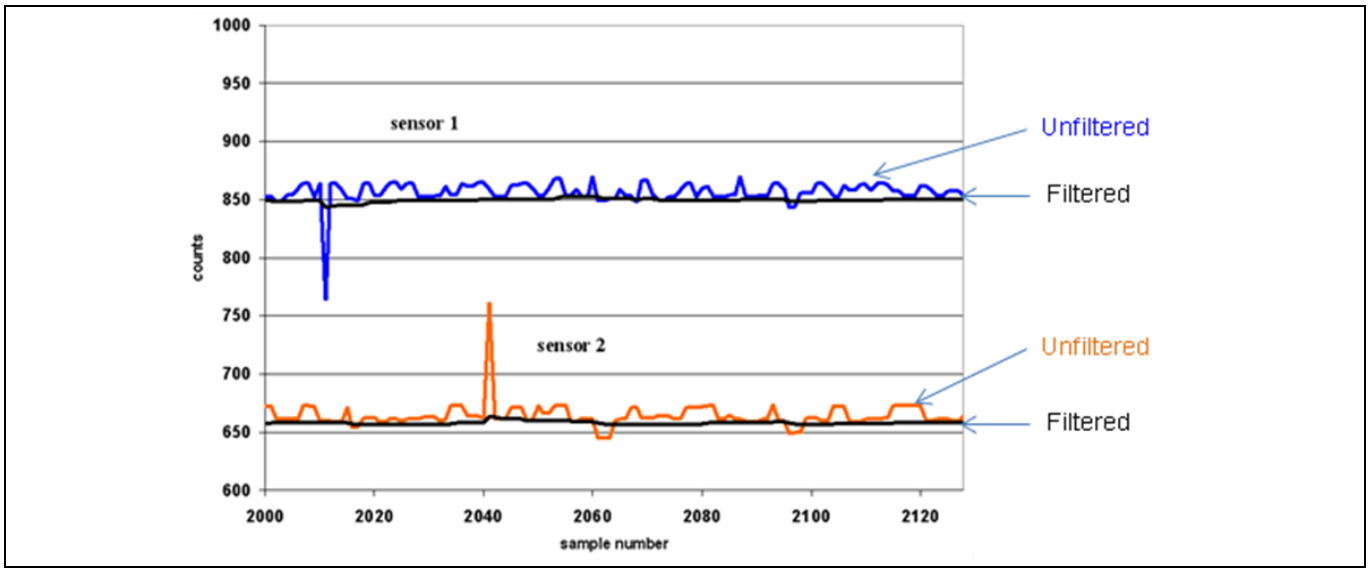

Figure 79 Median filter (16-sample) finger touch

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 20. This is a nonlinear filter.

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

Equation 20

Figure 80 shows the results of applying a jitter filter applied to noisy centroid data.

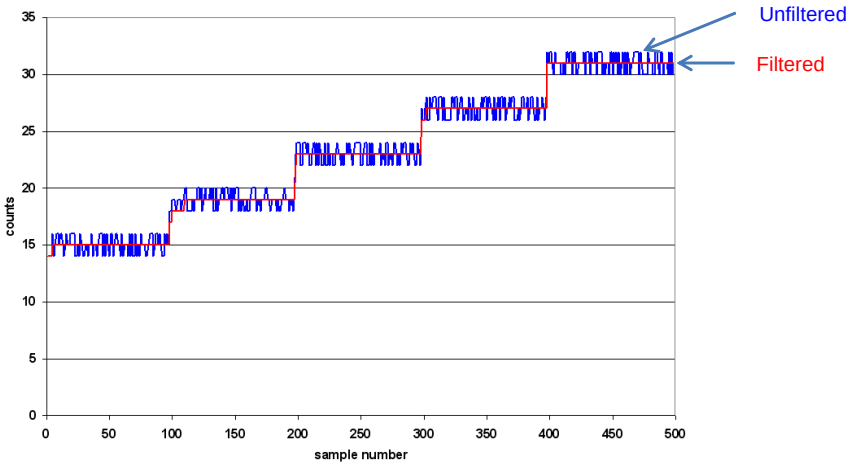

Figure 80 Jitter filter applied to noisy centroid data

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 reverted to the previous input plus or minus the threshold amount. The output does not change if the current input changes by less than the threshold amount. The general equation for a jitter filter applied to buttons is:

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

Equation 21

Figure 81 and Figure 82 show the result of using a jitter filter on real button data with a large component of periodic noise.

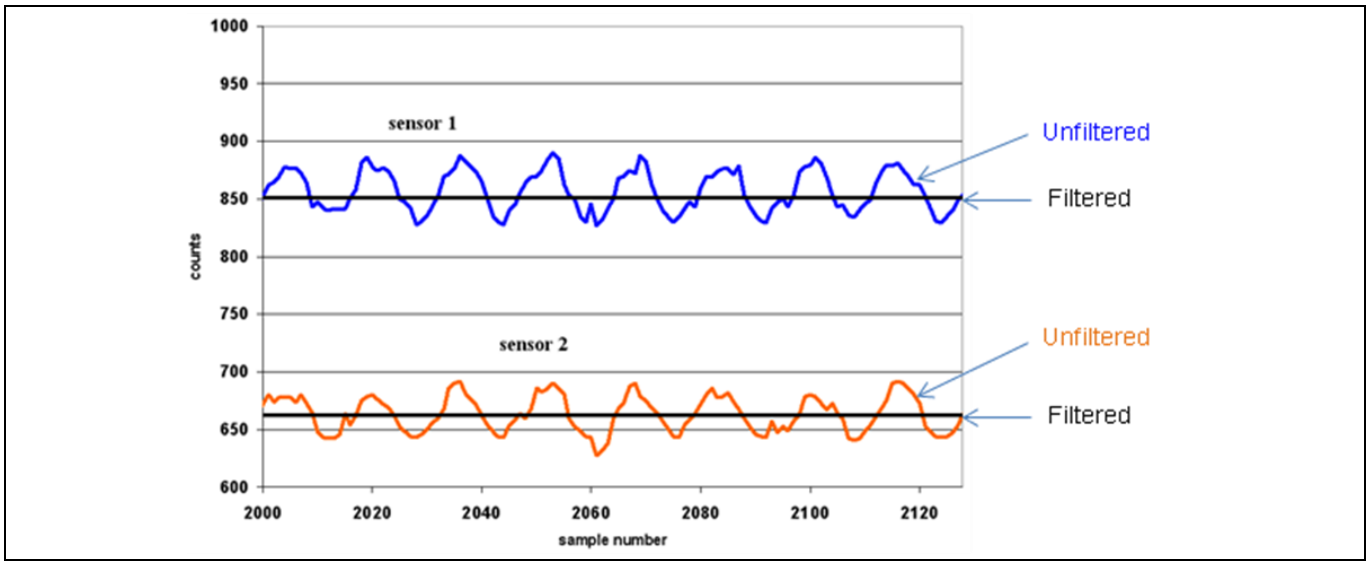

Figure 81 Jitter filter for button noise

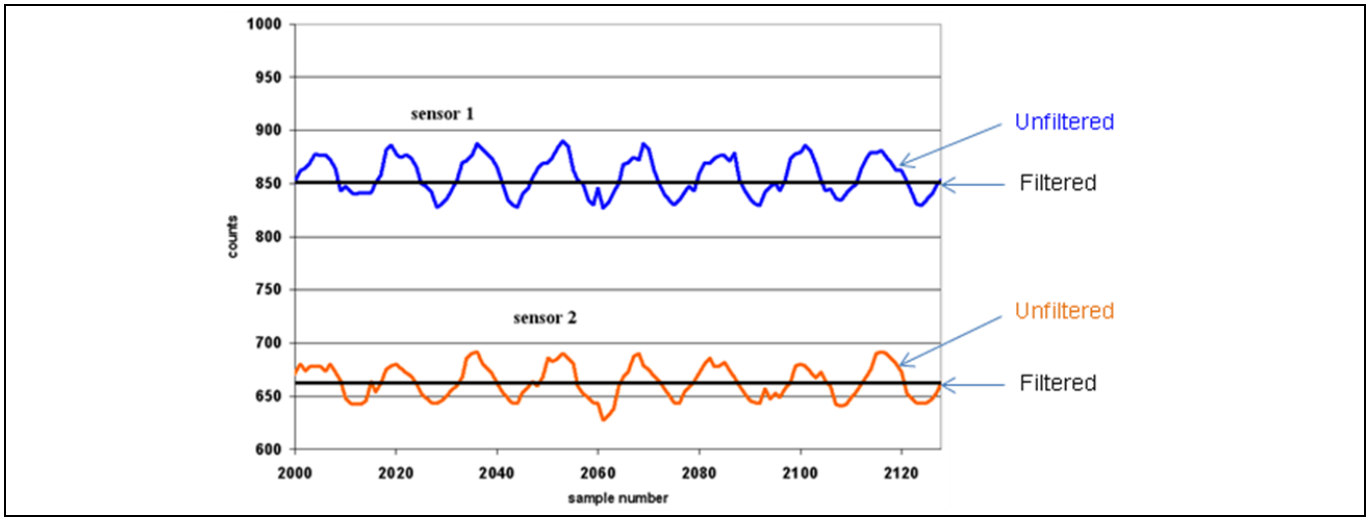

Figure 82 Jitter filter for button finger touch

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 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. Following are the common responses used with an event-based filter:

- To block the CAPSENSE™ data transmission until the pattern returns to normal
- To reset the level of the baseline reference defined in SNR
- 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. I²C is one of the common communication protocols used in CAPSENSE™ applications. Because I²C interrupts are asynchronous in nature, they may occur when the sensors are being scanned. Interrupts that 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 considers 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 these kinds 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 it may occur 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 83.

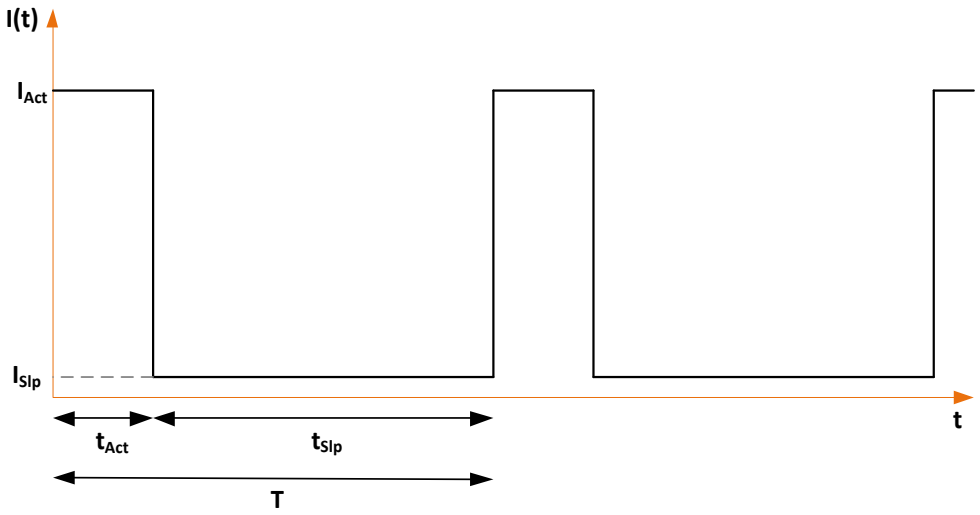

Figure 83 Instantaneous current

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 22

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

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

Equation 23

3.5.3 Response time versus power consumption

As illustrated in Equation 23, 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 design

This section presents the steps involved in implementing a proximity sensor and the various factors that affect the proximity distance. For the detailed design guidelines, see AN92239 – Proximity sensing with CAPSENSE™.

3.6.1 Implementing proximity sensing with CAPSENSE™

Figure 84 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 Infineon'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, see 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 see 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, see the Proximity sensor design and Factors affecting proximity distance sections.
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.

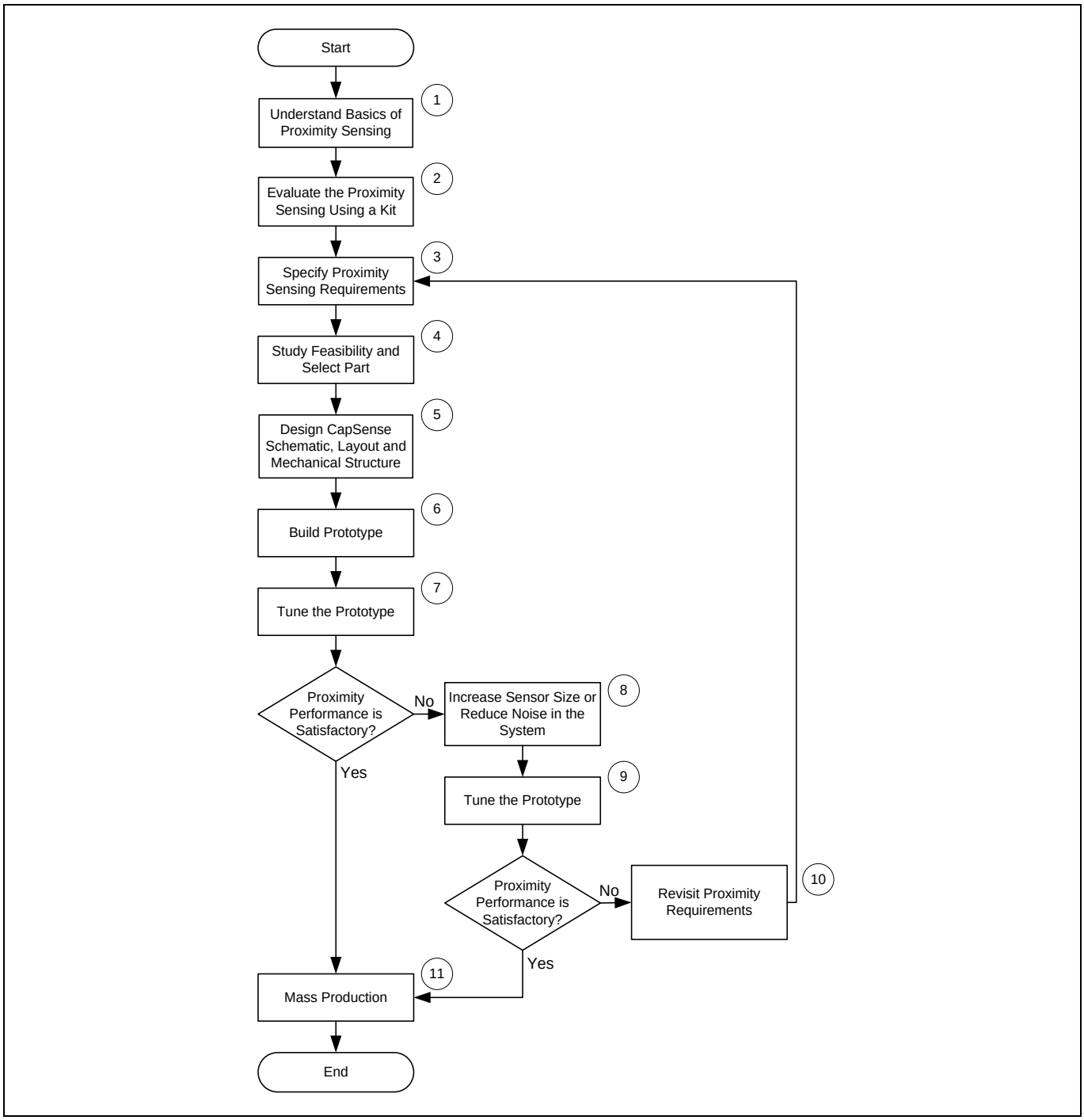

Figure 84 CAPSENSE™-based proximity sensing design

3.6.2 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 85. 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.

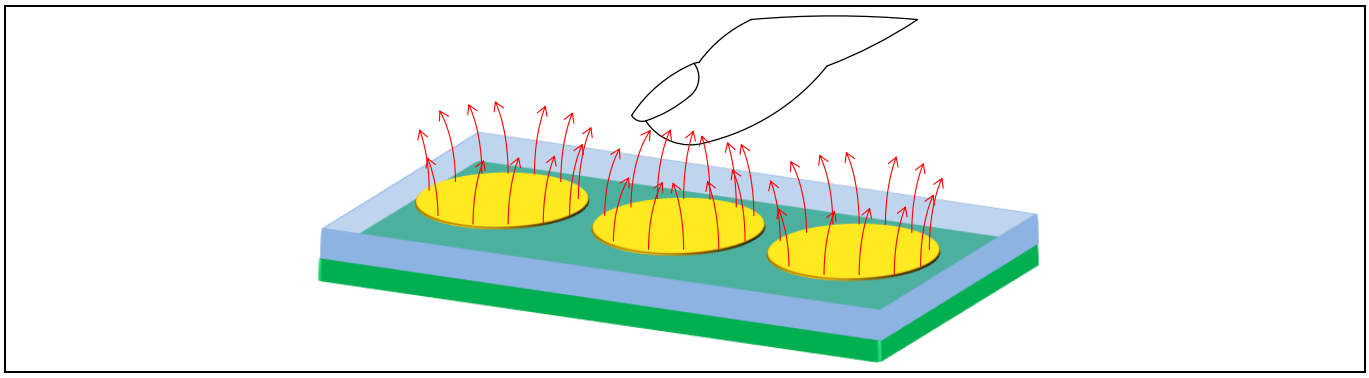

Figure 85 CAPSENSE™-based proximity sensing with button sensor

- **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 86. Ganging multiple sensors increases the effective sensor area and results in higher proximity-sensing distance, but ensure that the C_P of the ganged sensor does not cross the maximum C_P limit of 45 pF. See AN92239 – Proximity sensing with CAPSENSE™ for details on how to implement proximity sensing using sensor ganging.

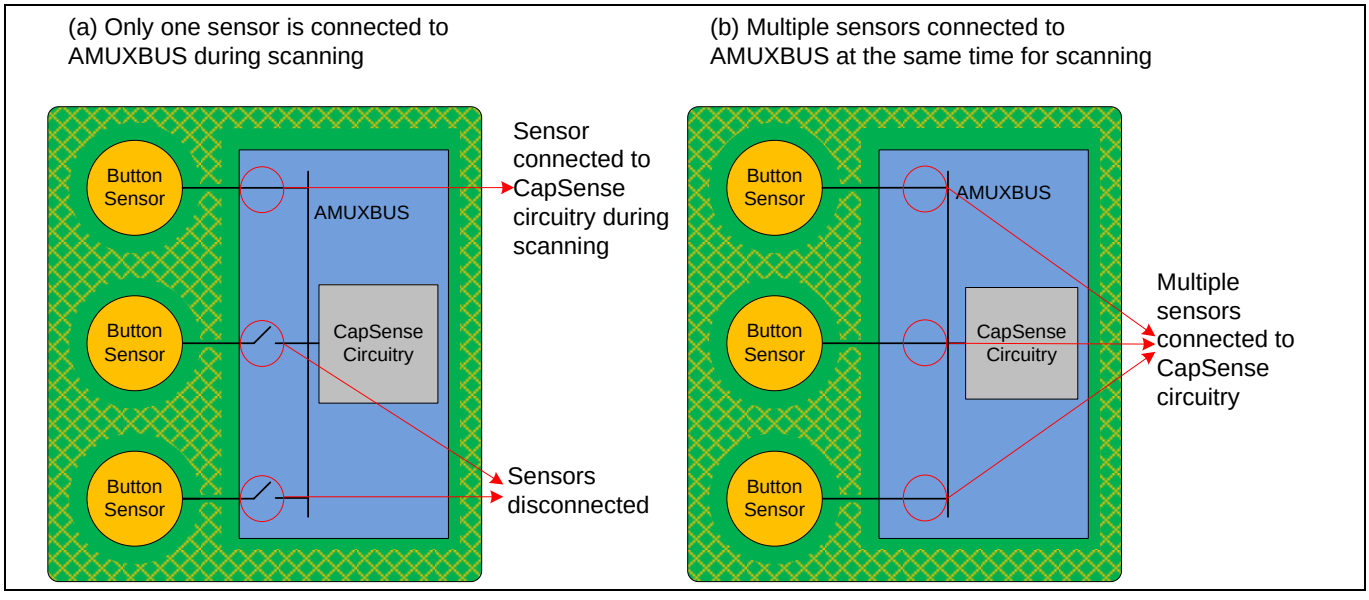

Figure 86 CAPSENSE™-based proximity sensing with sensor ganging

- **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 87), or it can surround the perimeter of a system's user interface, as shown in Figure 87. Implementing a proximity sensor with a PCB trace has the following advantages when compared to other sensor implementation methods:
  - Proximity-sensor CP is less

- Proximity-sensing distance is higher because more electric field lines couples to the hand
- More appropriate for mass production

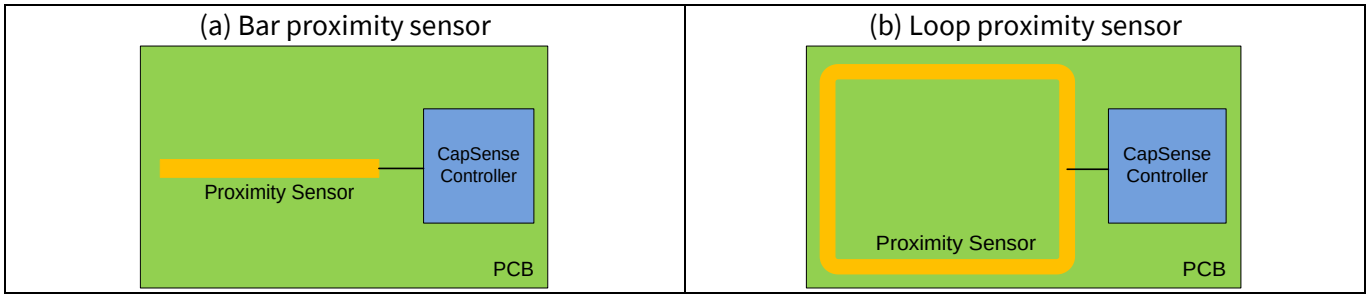

Figure 87 CAPSENSE™-based proximity sensing with PCB trace

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

3.6.3 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

3.6.3.1 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 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 the proximity-sensor type. Table 9 shows when to use a specific proximity-sensor implementation method.

Table 9 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 88 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 87) 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.

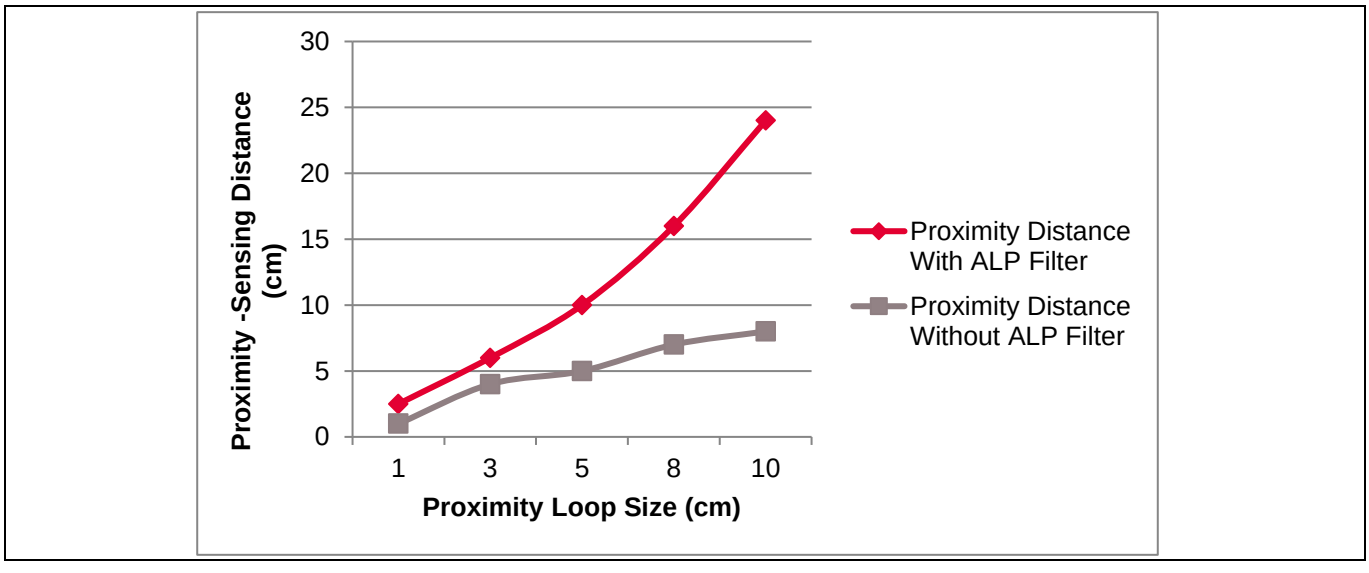

Figure 88 Proximity loop size versus proximity distance

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 needed to achieve a required proximity-sensing distance by making sensor prototypes. You can use a copper foil, as Figure 89 shows, to make a quick sensor prototype to determine the sensor size needed to achieve the required proximity-sensing distance.

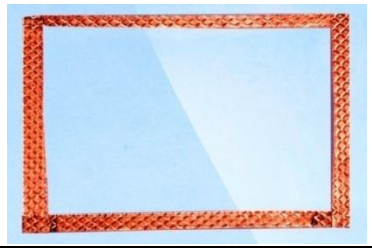

Figure 89 Proximity sensor prototype using copper tape

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 cannot 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 10 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. See AN92239 – Proximity sensing with CAPSENSE™ for details on ALP filter.

Table 10 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 floating or grounded conductive object nearby. The following factors cause the proximity-sensing distance to reduce drastically when conductive objects are placed close to the proximity sensor:
  - The CP 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 91.

  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 92. For layout recommendations on shield electrode, see the Shield electrode and guard sensor section.

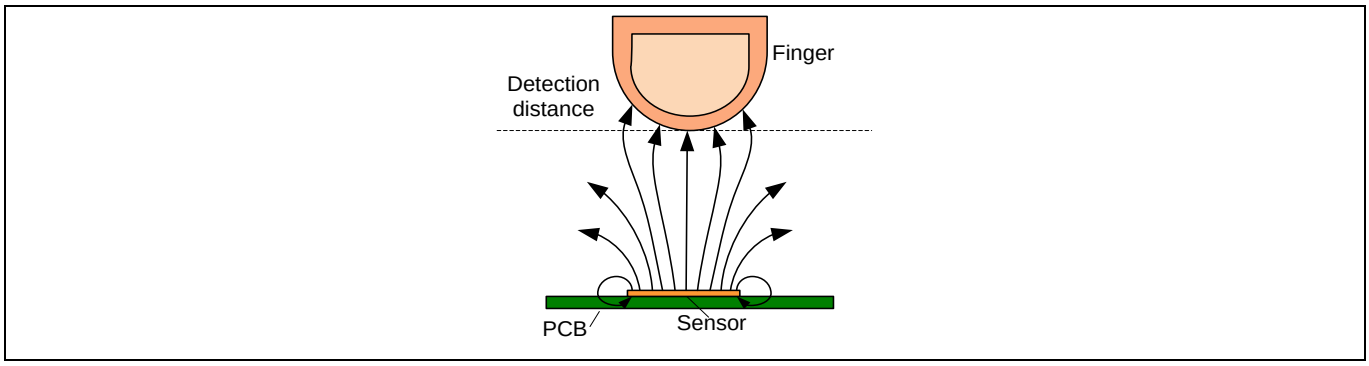

Figure 90 Electrical field propagation for a single sensor configuration without a metal object

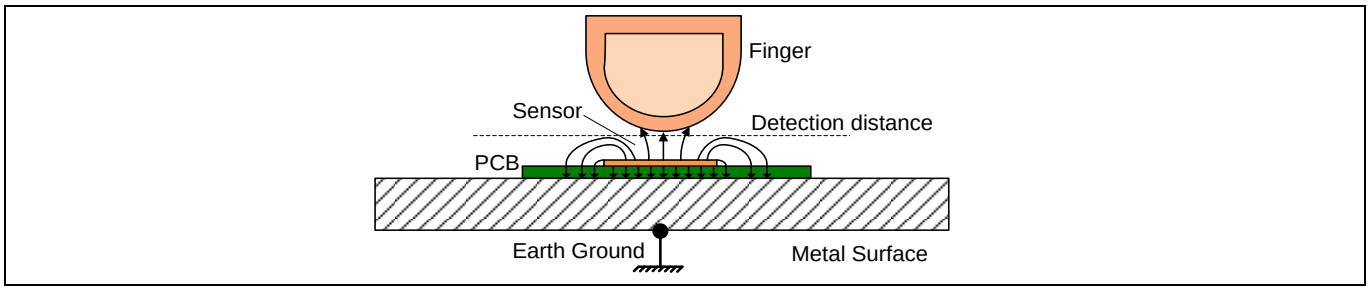

Figure 91 Electrical field propagation for a single sensor configuration with a solid metal object

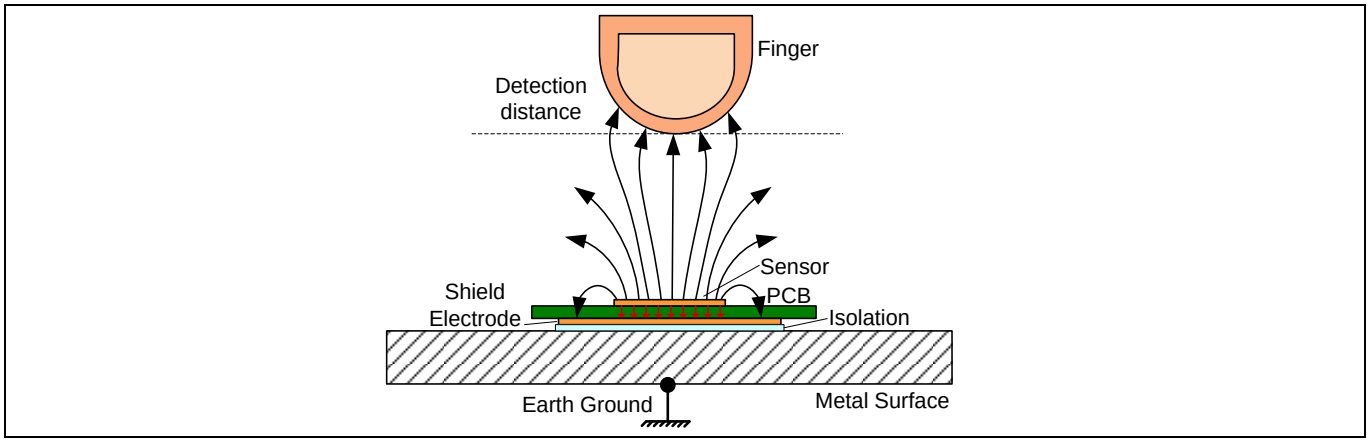

Figure 92 Using a shield electrode to decrease the metal object's influence

3.6.3.2 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, you can detect small changes in C_F 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 help in reducing the noise, thereby increasing the SNR and the proximity-sensing distance. You can use IIR, median, average or ALP filters to reduce the noise. See the Software filtering section for details on IIR, median, and average filters. See the application note AN92239 – Proximity sensing with CAPSENSE™ for details on the ALP filter.

3.6.3.3 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 93.

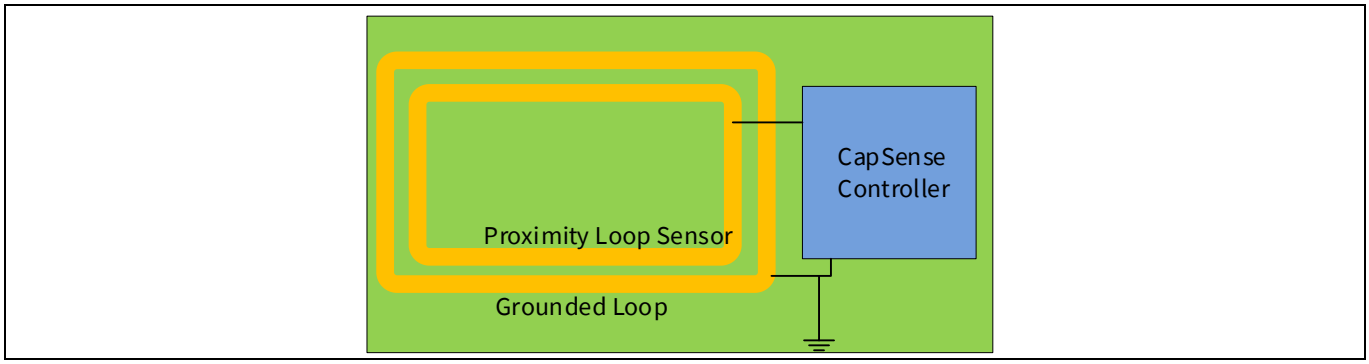

Figure 93 Ground loop surrounding the sensor for improved ESD performance

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 94 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.

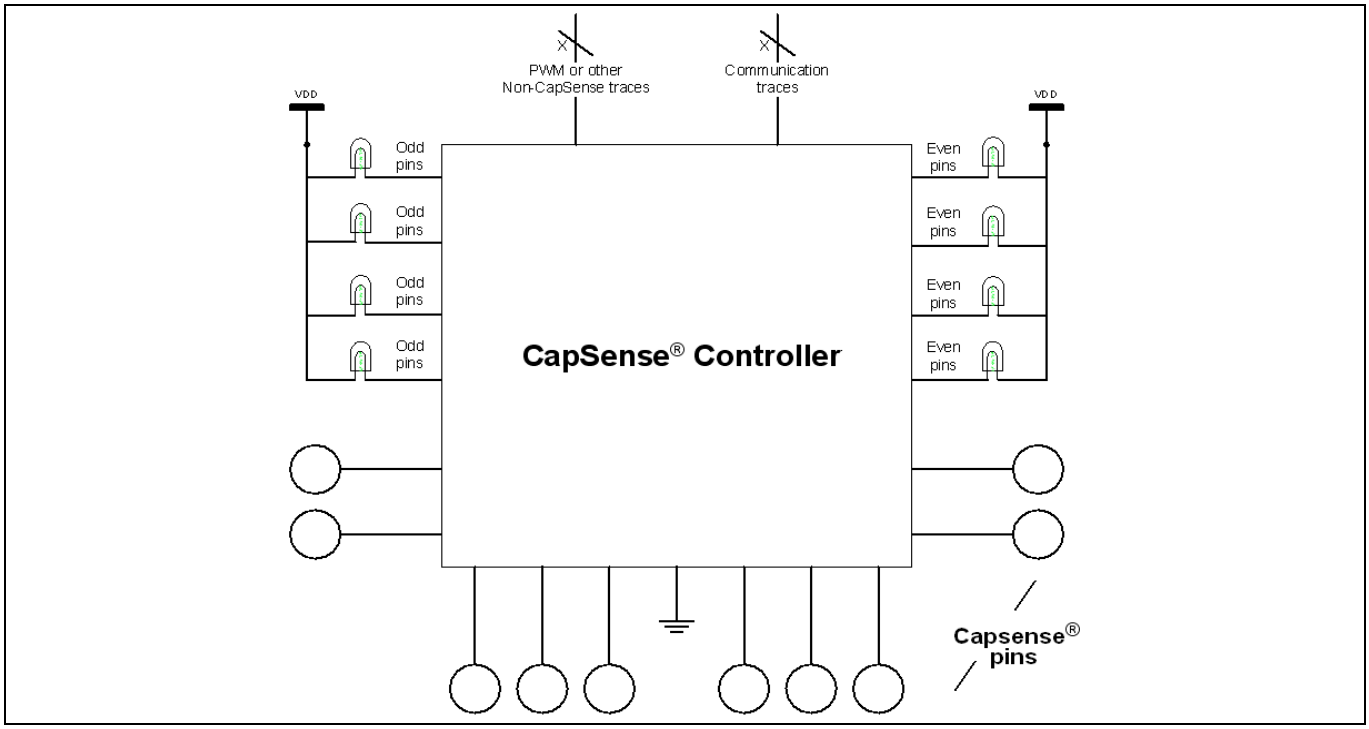

Figure 94 Recommended: Port isolation for communication, CAPSENSE™, and LEDs

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 95, CAPSENSE™ and non-CAPSENSE™ traces are not isolated, and CAPSENSE™ pins are far from ground. This is an example of a bad pin assignment.

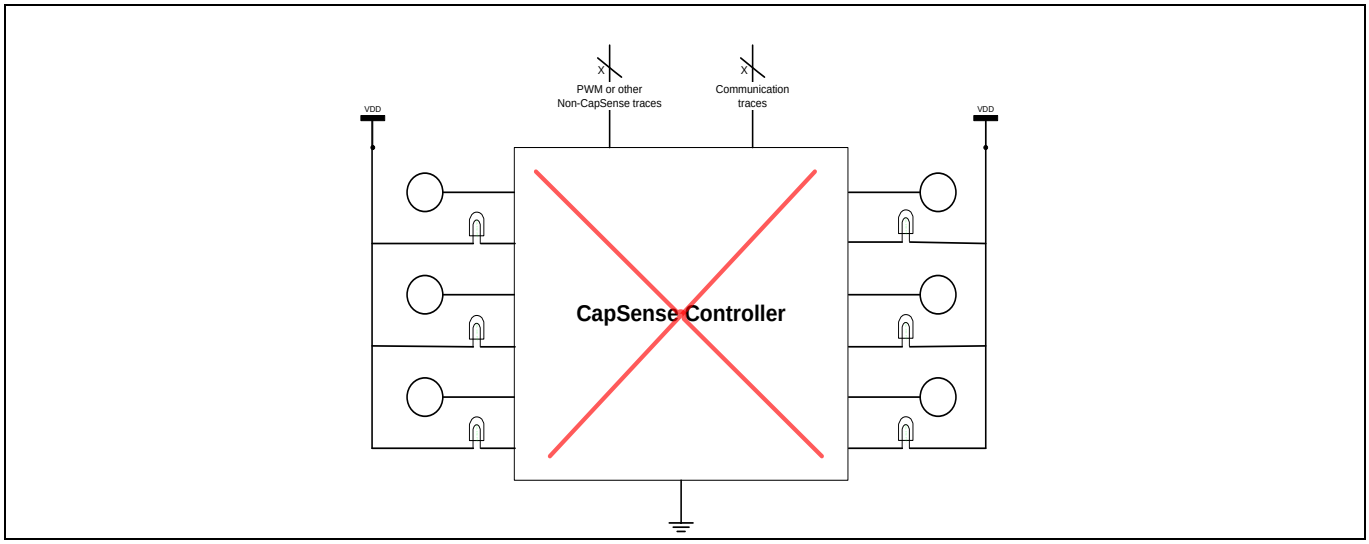

Figure 95 Not recommended – CAPSENSE™ and non-CAPSENSE™ pins in proximity

The example in Figure 96 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.

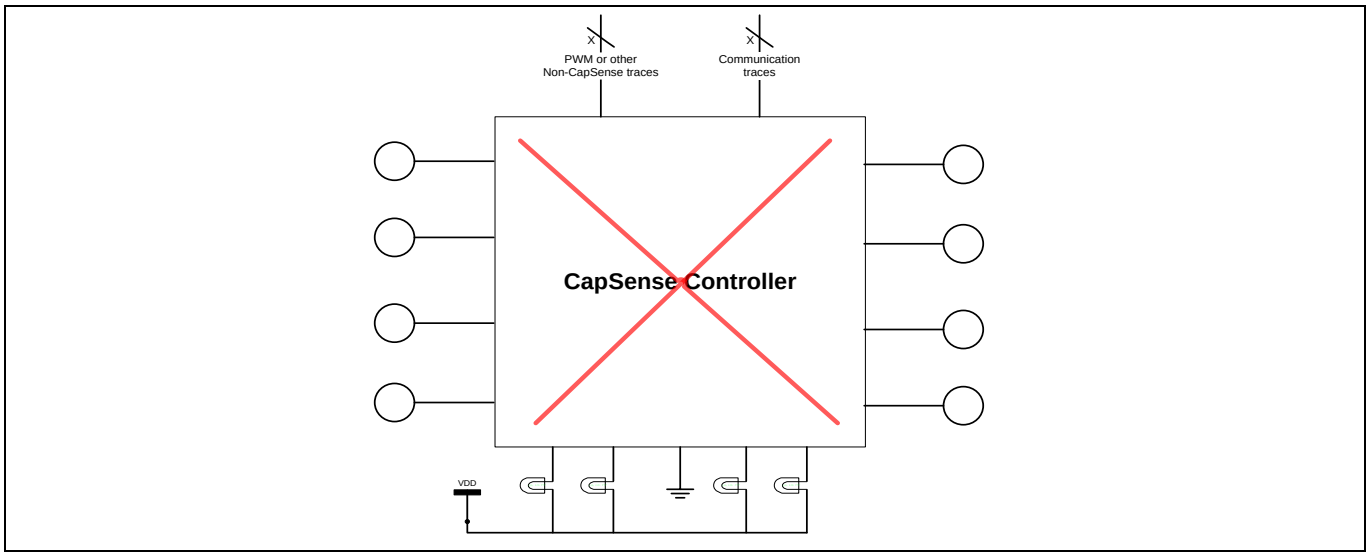

Figure 96 Not recommended – LEDs and ground pins in proximity

Further, LEDs should not be placed close to C_{MOD}/R_B pin to avoid crosstalk as illustrated in Figure 97.

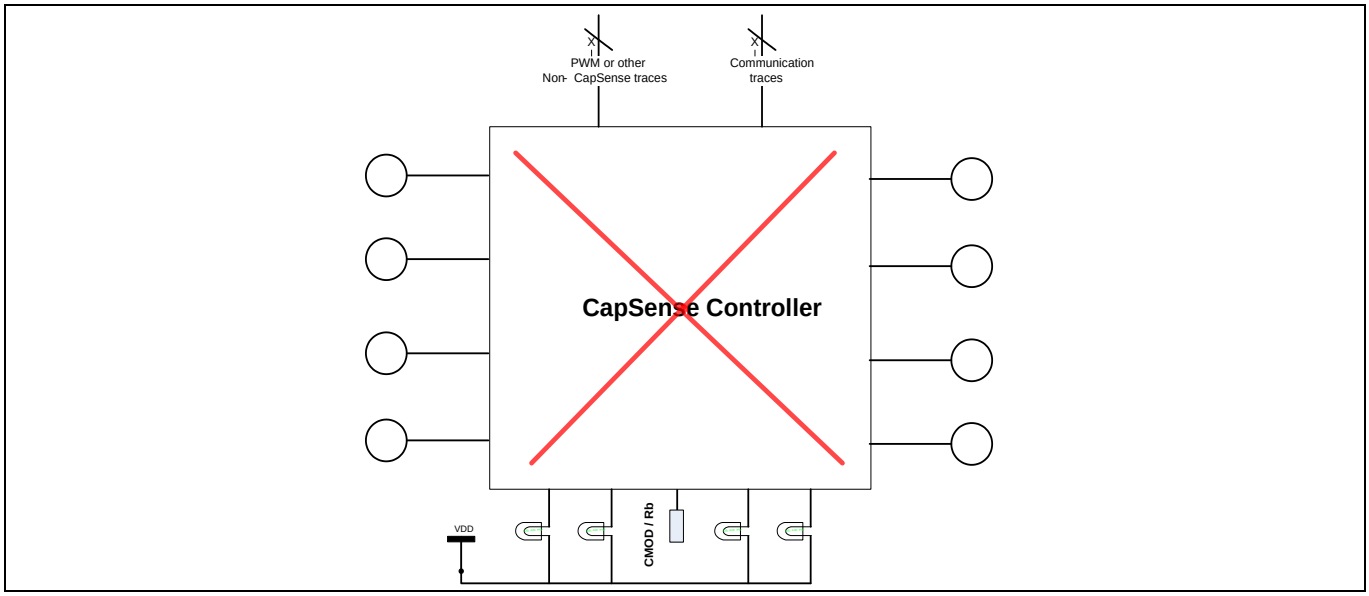

Figure 97 Not recommended: C_{MOD}/R_B and LED pins in proximity

Note that in PSoC™ 1, 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 see the individual device design guides webpage having the sample schematics of all the CAPSENSE™ devices:

- AN66271 - CY8C21X34/B - CAPSENSE™ desing guide
- AN66269 - CY8C20x34 - CAPSENSE™ design guide
- AN65973 - CY8C20xx6A/H/AS - CAPSENSE™ design guide
- AN78329 - CY8C20xx7/S - CAPSENSE™ design guide

For similar guidelines on PSoC™ 3, PSoC™ 4 and PSoC™ 5LP, see 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 98. 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.

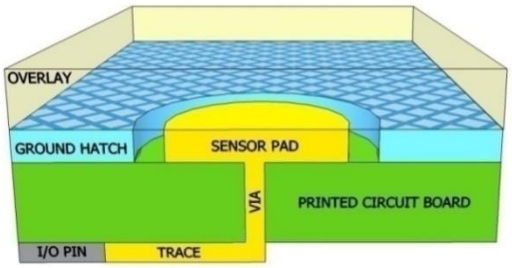

Figure 98 Two-layer stack-up for CAPSENSE™ boards

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

This section explains the structure for self-cap and mutual cap buttons.

3.8.4.1 Self-cap button structure

The best shape for self-cap 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.

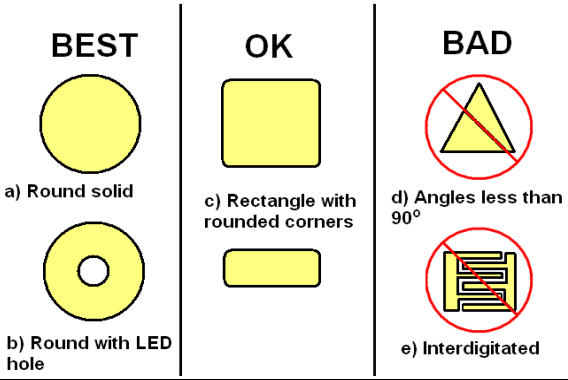

Figure 99 Recommended button shapes

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.

3.8.4.2 Mutual cap buttons of fishbone structure

Mutual capacitance sensing measures the change in capacitive coupling between two electrodes. The sensor pattern should be designed in such a way that the finger disturbs the electric field between the electrodes to a maximum extent.

Prongs or fishbone are standard shapes for mutual-capacitance buttons. The Tx forms a box or ring around the outside of the key to help shielding Rx from noise. There are interlaced Tx and Rx prongs inside the border to form the electric field between the two electrodes. Figure 100 shows an example of a three-prong fishbone sensor structure. The gap between the outer wall of the Tx electrode and the coplanar hatch ground should be greater than the air-gap of Tx and Rx electrodes. The reference plane (PCB bottom layer) of the fishbone structure should have void region as shown in Figure 100.

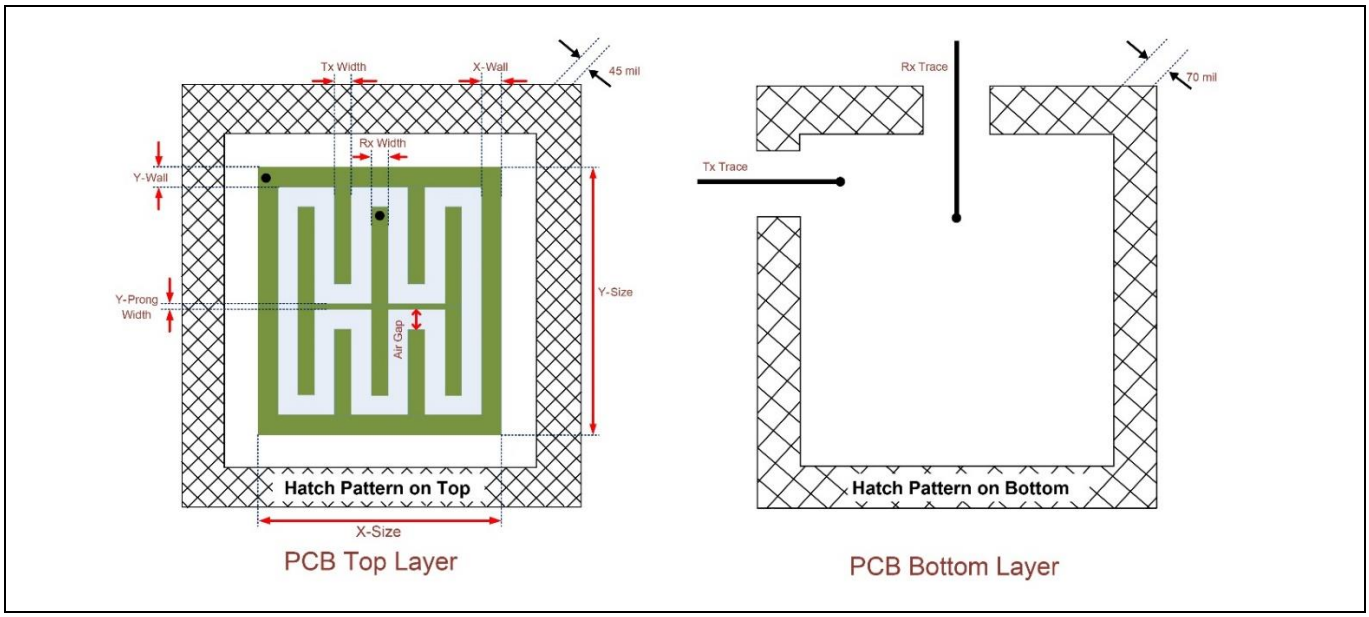

Figure 100 Fishbone pattern of mutual-capacitance button design

Table 11 lists some of the common fishbone structure dimensions.

Table 11 Dimension of fishbone buttons (all units in mm)

Key size (X-size, Y-size)	PCB thickness	Number of Rx-prongs	Gap between Tx and Rx	Tx width	Rx width	X-wall width	Y-wall width
13, 10	1.5	2	1.4	1.8	1.8	1	1
13, 10	0.1	3	1.2	0.7	1.2	0.4	0.3
20, 13	0.1	2	1.2	2.7	2.7	3	3.55
10, 13	1.5	2	1.2	0.6	0.6	1.7	1.5
10, 10	1.5	2	1.4	1	1	0.7	0.7

The key dimensions listed in Table 11 are examples of some proven button structures that show good SNR performance for different types of PCBs (FR4, flex PCB, ITO based touch film), 4 mm thick overlay, and of overlay permittivity value 3.5. You can contact Infineon technical support team if you need to create custom key pattern in your application.

3.8.5 Slider design

Figure 101 shows the recommended slider pattern for a linear slider and Table 12 shows the recommended values for each of the linear slider dimensions. Detailed explanation on the recommended layout guidelines are provided in the following sections.

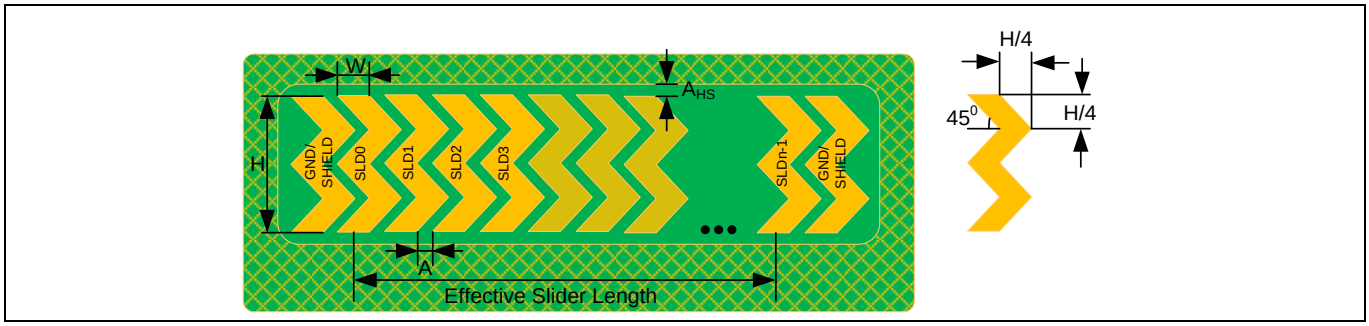

Figure 101 Typical linear slider pattern

Table 12 Linear slider dimensions

Parameter	Acrylic overlay thickness	Minimum	Maximum	Recommended
Width of the segment (*W*)	1 mm	2 mm	-	8 mm[6]
	3 mm	4 mm	-	
	4 mm	6 mm	-	
Height of the segment (*H*)	-	7 mm[7]	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

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[8]. 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 101 shows. This shape helps in achieving a centroid response close to the ideal response, as Figure 102 and Figure 103 show.

[6] The recommended slider-segment-width is based on an average human finger diameter of 9 mm. See Slider-segment shape, width, and air gap section for more details.

[7] The minimum slider segment height of 7 mm is recommended based on a minimum human finger diameter of 7 mm. Slider height may be kept lower than 7 mm, provided the overlay thickness and CAPSENSE™ tuning is such that an SNR ≥ 5:1 is achieved when the finger is placed in the middle of any segment.

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

For the same reason, the slider-segment width and air-gap (i.e. dimensions “W” and “A” respectively, as marked in Figure 101) should follow the relation mentioned in Equation 24.

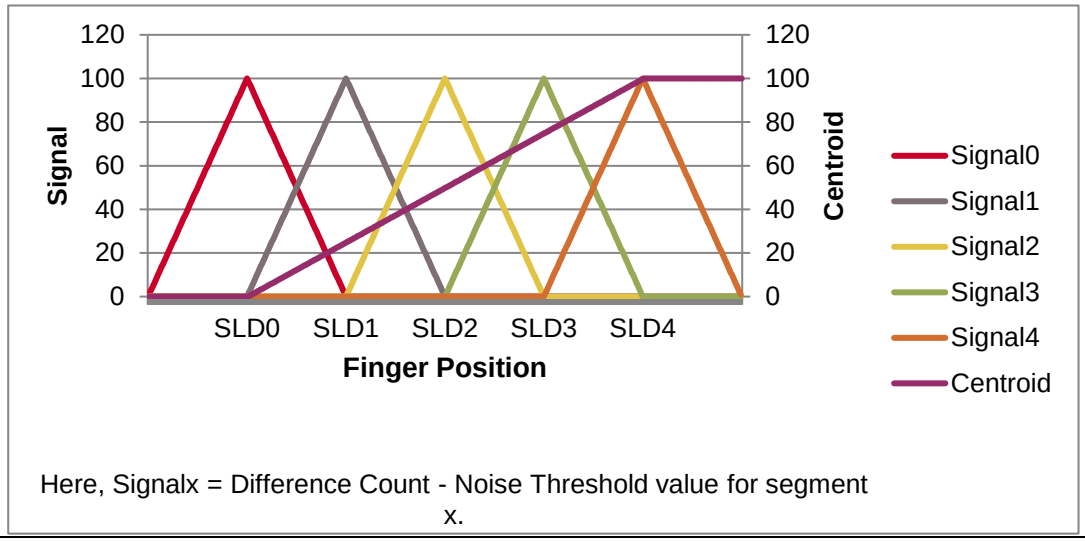

Figure 102 Ideal slider segment signals and centroid response

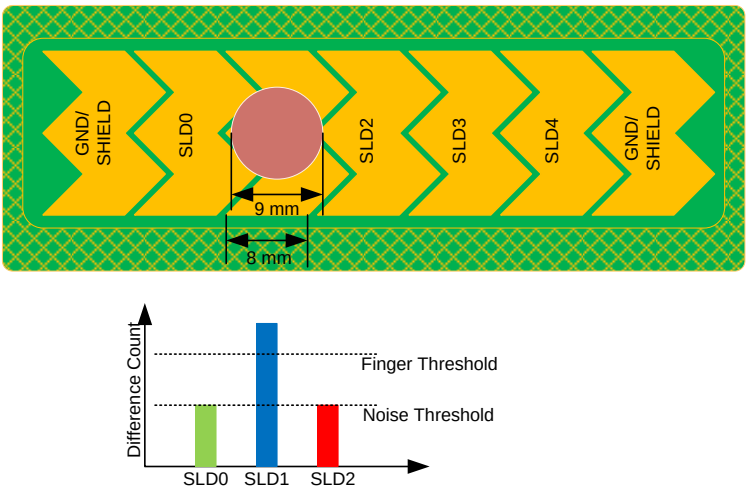

Figure 103 Ideal slider signals

Segment width and air-gap relation with finger diameter.

$$W + 2A = \text{finger diameter}$$

Equation 24

Typically, an average human finger diameter is approximately 9 mm. Based on this average finger diameter and Equation 24, the recommended slider-segment-width and air-gap is 8 mm and 0.5 mm respectively.

If the *slider-segment-width + 2 * air-gap* is lesser than *finger diameter*, as required per Equation 24, 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 104 shows. Thus, calculated centroid position per Equation 25 will be non-linear, as Figure 104 shows.

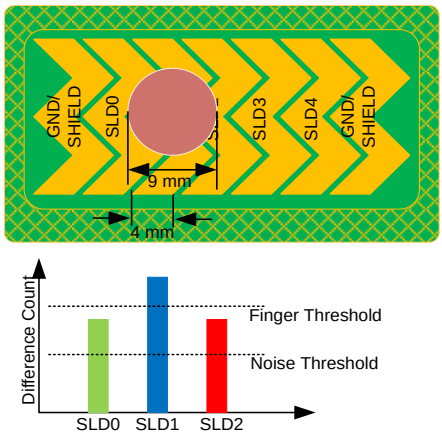

Figure 104 Finger causes valid signal on more than two segments when slider segment width is lower than recommended

Centroid algorithm used by CAPSENSE™

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

Equation 25

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

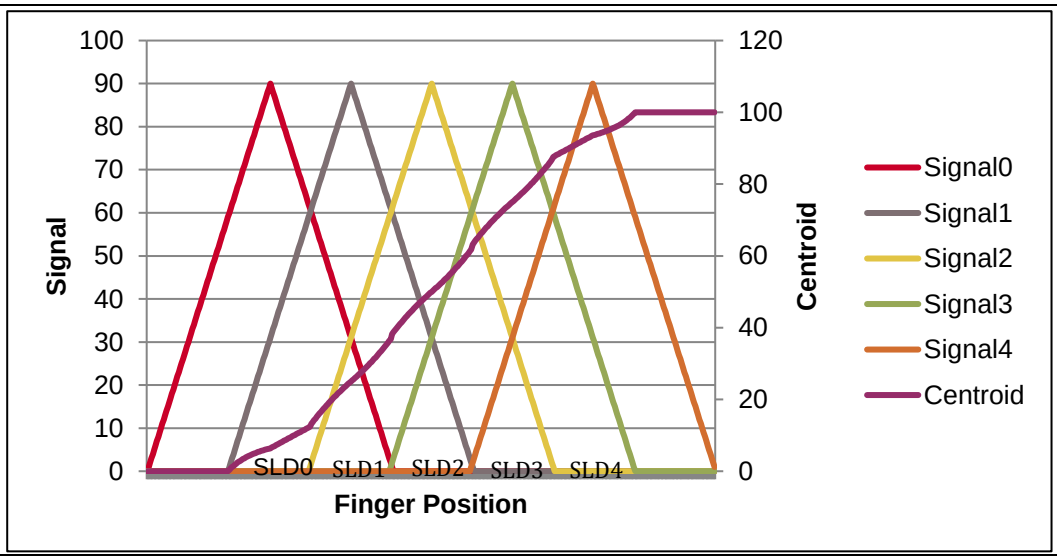

Figure 105 Nonlinear centroid response when slider segment width is lower than recommended

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 105 shows; it may still be used in an end application where the linearity of reported centroid versus 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[9] , as Figure 106 shows.

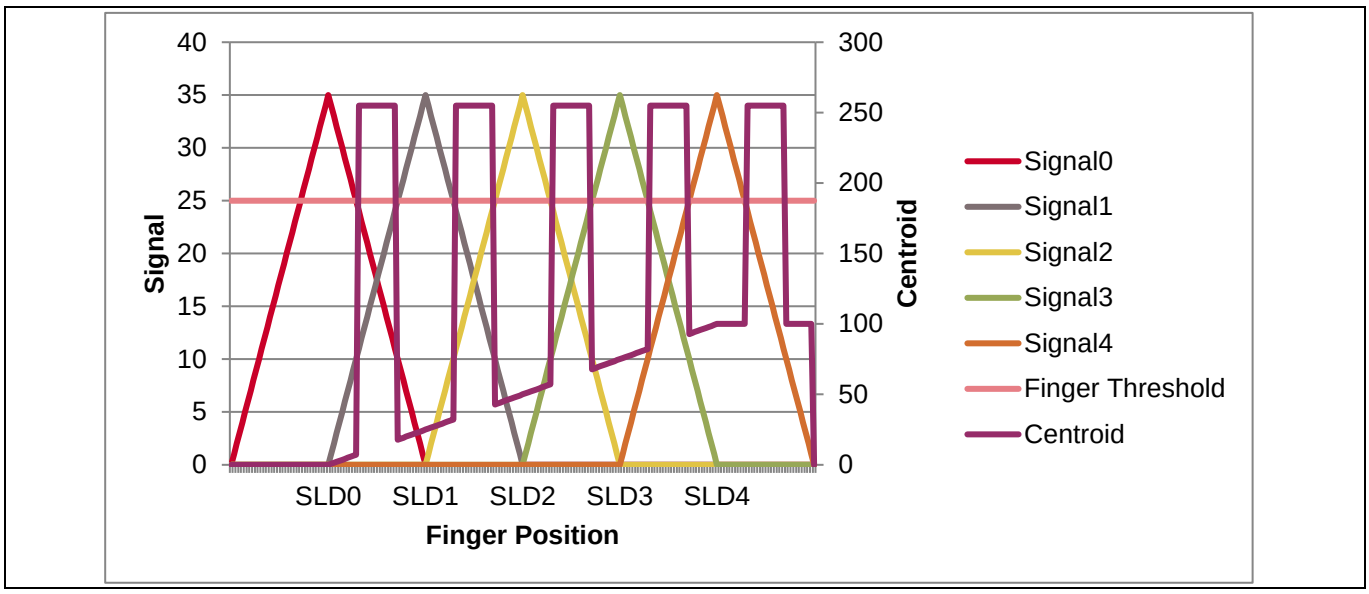

Figure 106 Incorrect centroid reported when slider segment width is too low

The minimum value of slider-segment-width for certain specific overlay thickness values, for an acrylic overlay, are provided in Table 12. For acrylic overlays of thickness values, which are not specified in Table 12 and Figure 107 may be used to estimate the minimum slider-segment-width.

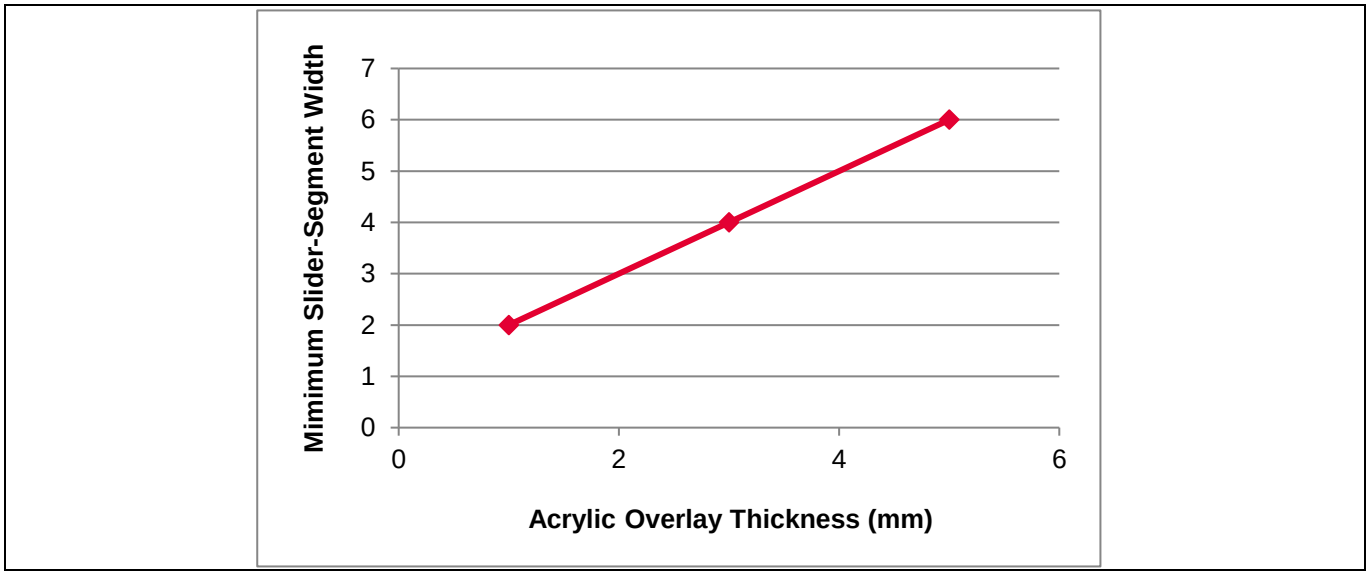

Figure 107 Minimum slider-segment-width with respect to overlay thickness for an acrylic overlay

If the *slider-segment-width + 2 * air-gap* is higher than *finger diameter*, as required per Equation 24, the centroid response will have flat spots i.e., if the finger is moved a little near the middle of any segment, the reported

[9] 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.

centroid position will remain constant as Figure 108 shows. This is because, as Figure 109 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.

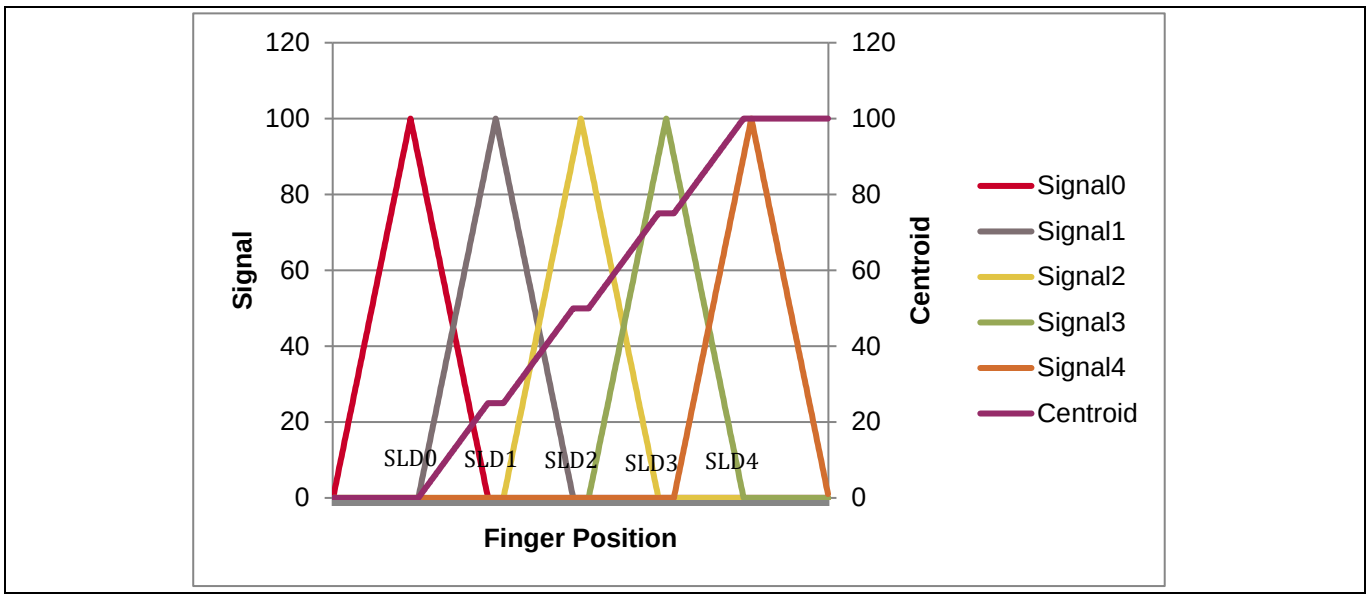

Figure 108 Flat spots (non-responsive centroid) when slider segment width is higher than recommended

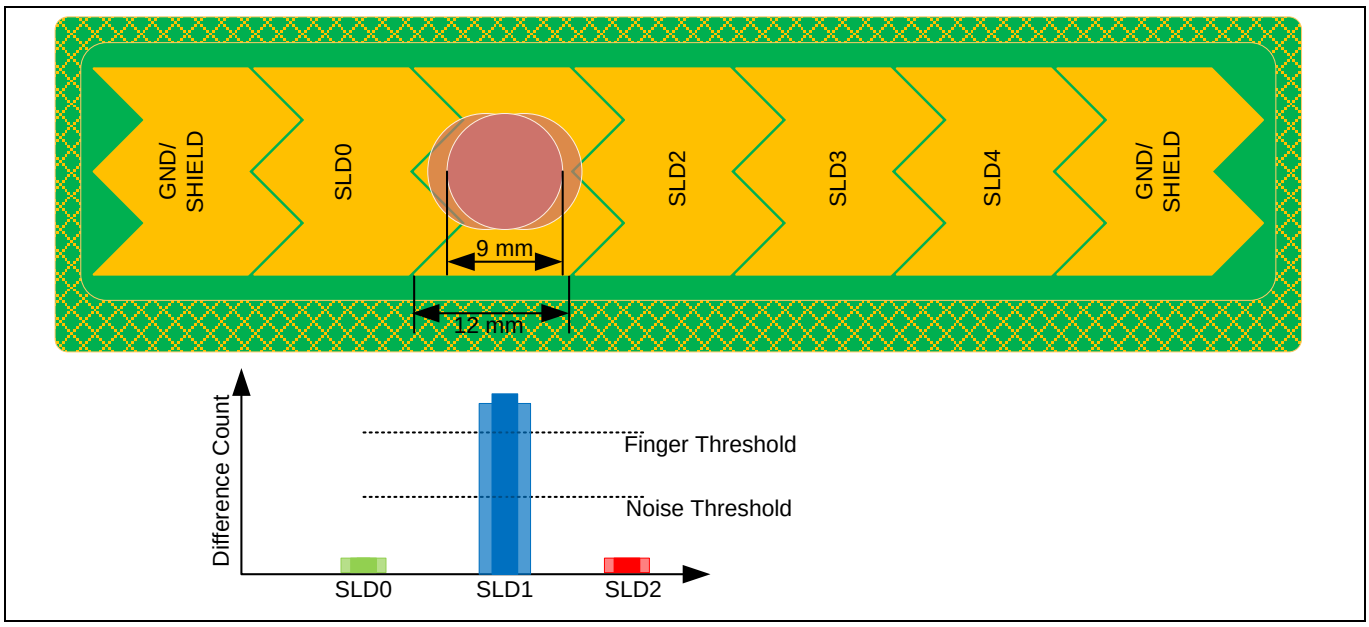

Figure 109 Signal on slider segments when slider segment width is higher than recommended

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 102); 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 101 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 110 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.

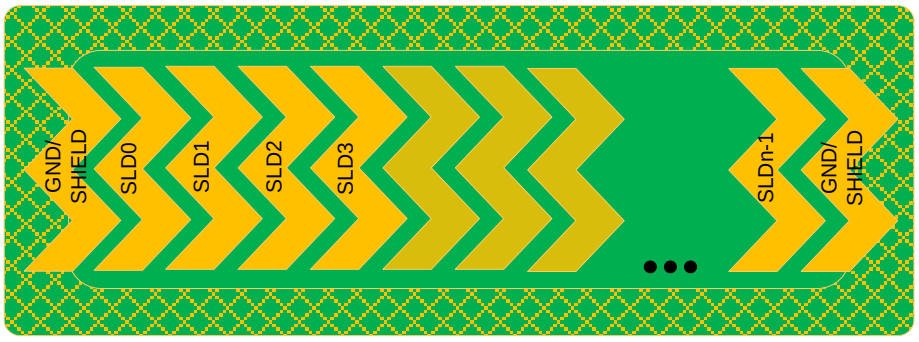

Figure 110 Linear slider pattern when first and last segments are connected to top hatch fill

3.8.5.3 Deciding slider dimensions

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

1. Decide the required length of slider (*L*), based on application requirements. This is same as the “effective slider length” as Figure 101 shows.
2. 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 12.
3. The slider-segment-width and the air-gap between slider segments should be as recommended in Table 12.
4. The recommended slider-segment-width and air-gap for an average finger diameter of 9 mm is 8 mm and 0.5mm respectively.
5. For a given slider length, *L*, calculate the number of segments required using the following formula:

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

Equation 26

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 108 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 101 shows. PCB length can be related to slider length as follows:

Relationship between minimum PCB length and slider length.

$$\text{PCB length} = \text{slider length} + 3 \text{ x slider segment width} + 2 \text{ x air gap}$$

Equation 27

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:

$$\text{PCB length} = \text{slider length} + \text{slider segment width}$$

Equation 28

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 111 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. See Table 12 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.

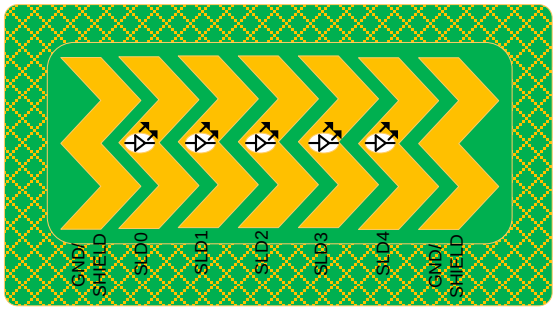

Figure 111 Slider design with LED backlighting

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.

3.8.6.1 2-Layer PCB

- Place the sensors on the top layer of the PCB, as Figure 112 shows.
- Place the components and route the sensor traces on the bottom layer of the PCB.

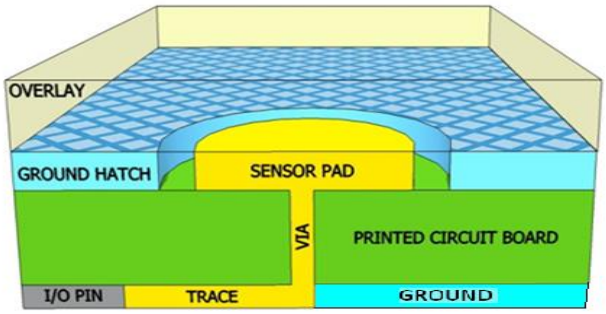

Figure 112 CAPSENSE™ design on 2-layer PCB

3.8.6.2 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 113 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.

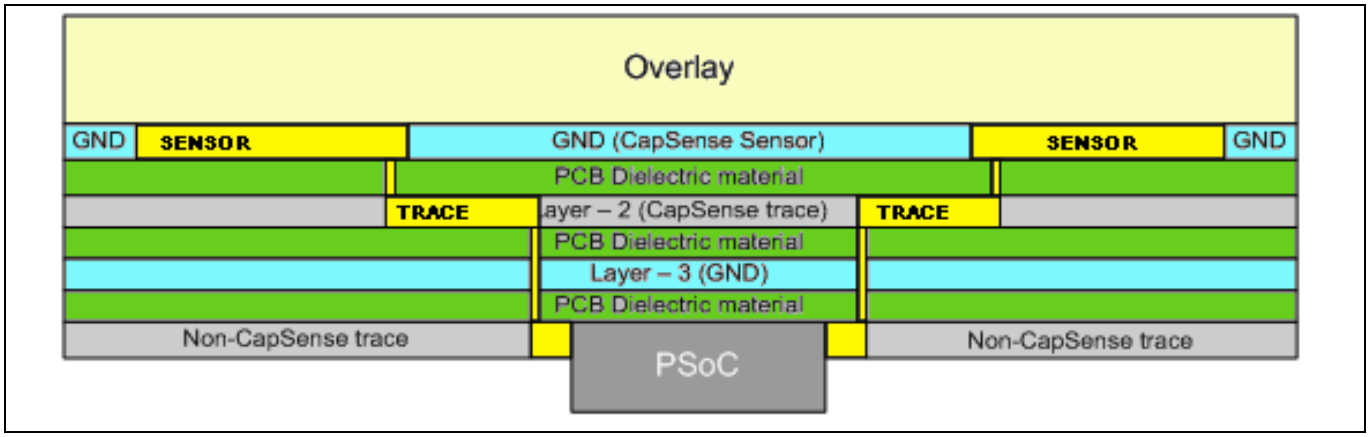

Figure 113 CAPSENSE™ design on 4-layer PCB

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 CP 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 114.

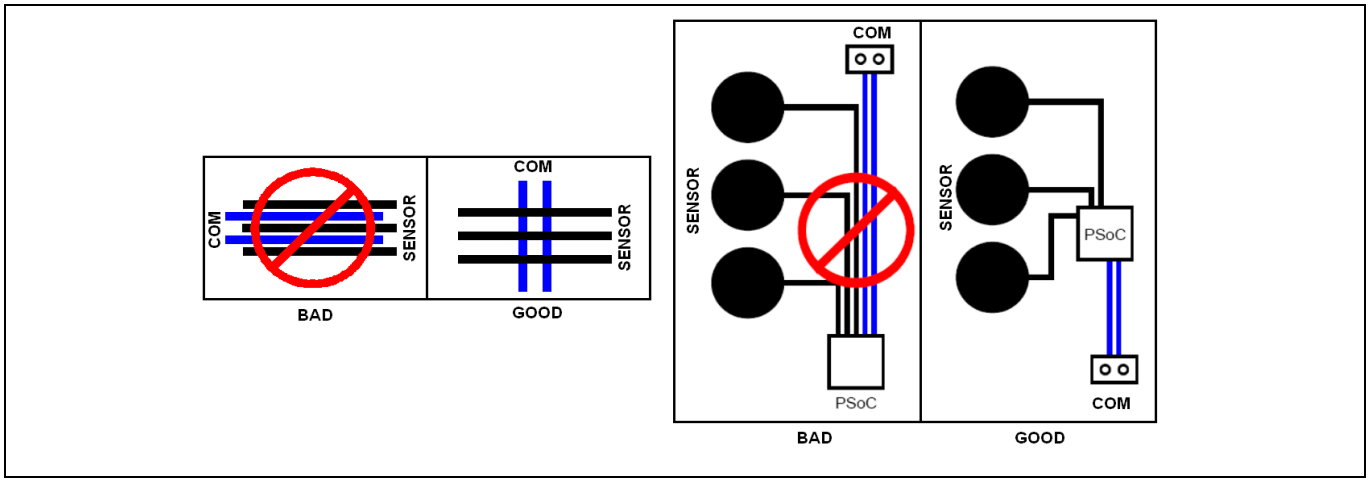

Figure 114 Routing of sensing and communication lines

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.

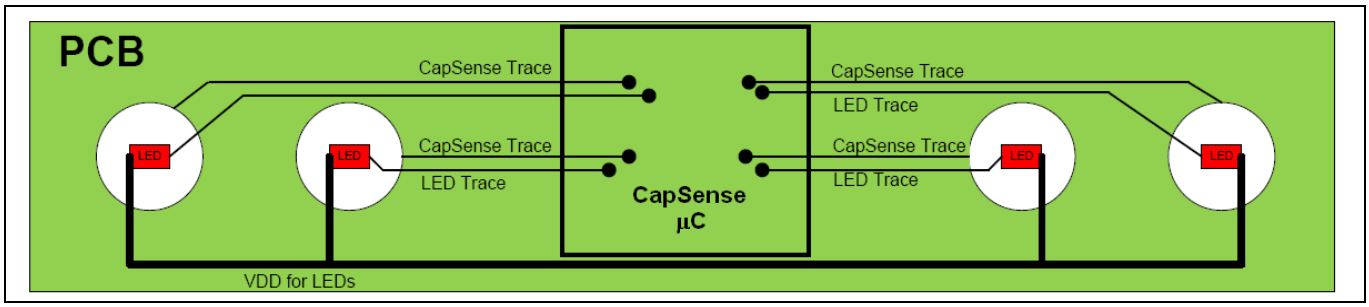

Figure 115 Not recommended – LED and CAPSENSE™ in close proximity

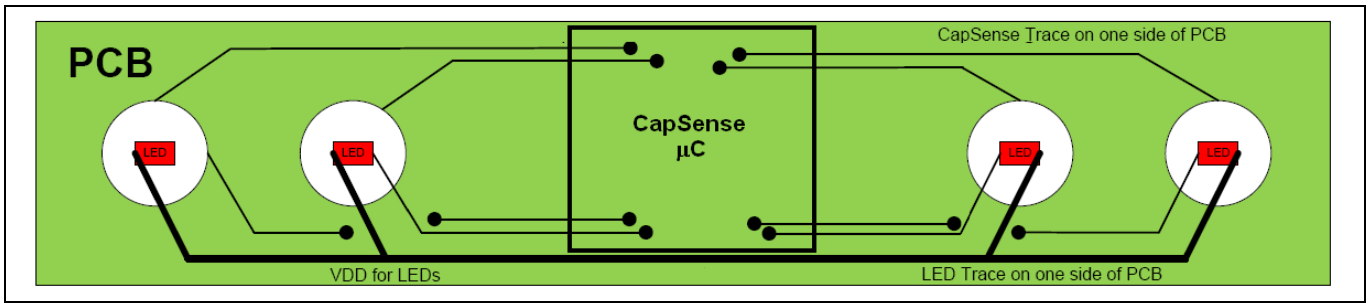

Figure 116 Recommended – LED and CAPSENSE™ with wide separation

Another approach to reducing crosstalk is to slow down the rising and falling edges of the LED drive voltage using a filter capacitor. Figure 117 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.

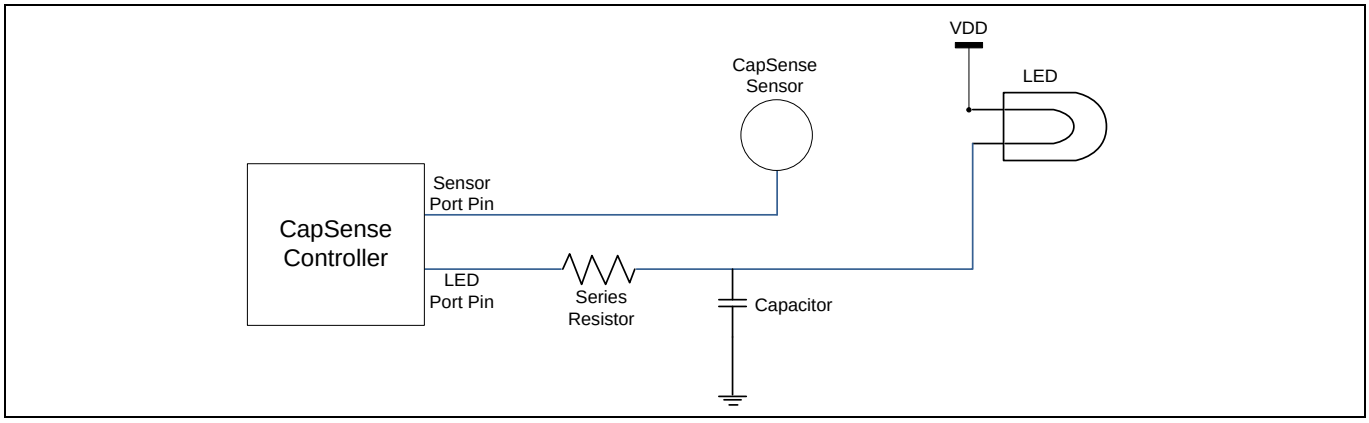

Figure 117 Filter capacitor solution for crosstalk

3.8.10 LEDs close to CAPSENSE™ sensors

If LEDs are placed close to the CAPSENSE™ sensors (within 4-mm distance), and if either end of the LED changes to a non-low impedance state at any point in time, the capacitance of the sensors changes between the On and Off states of the LEDs. The change in output impedance of the LED driver circuit can cause the sensors to false-trigger or sensors to go off unintentionally when the LEDs change state.

To avoid the effect of LEDs that are placed close to the sensors, the LEDs must be bypassed with a capacitor with a typical value of 1 nF. This is important in scenarios where LEDs are pulled down or pulled up to switch on and are left floating when switched off.

The value of the bypass capacitor must be such that it provides a constant low-impedance path as less as 1 kΩ at 100 KHz, as seen by the sensor on both the ends of the LEDs.

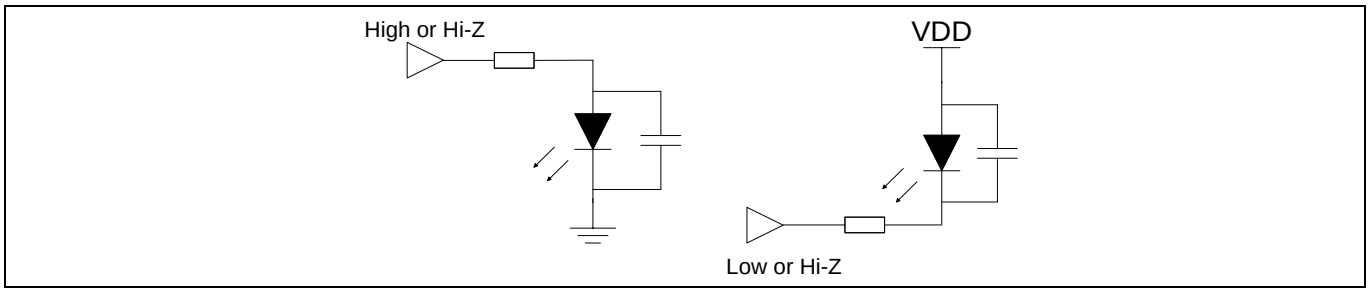

Figure 118 LED circuits

3.8.11 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 119.

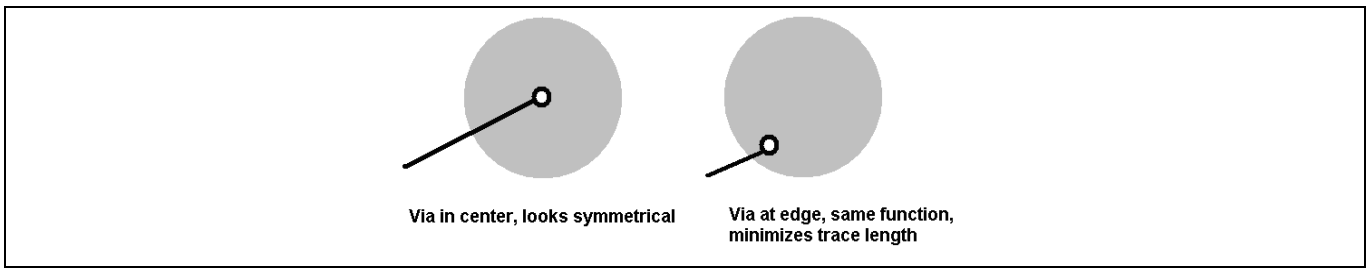

Figure 119 Vias placement on sensor pad

3.8.12 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).

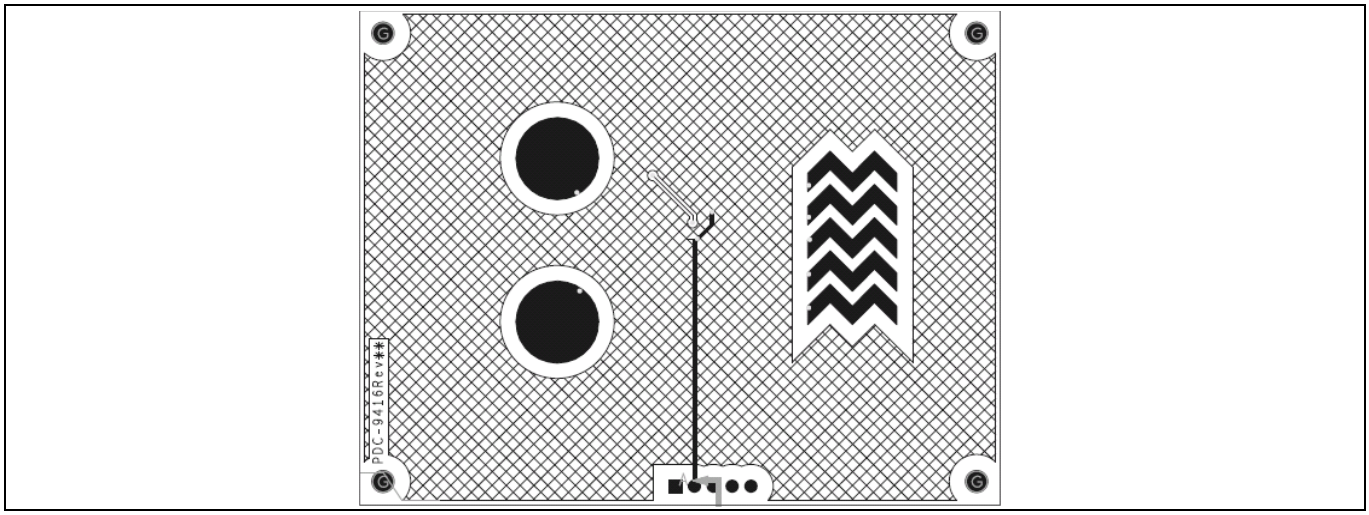

Figure 120 Recommended button and slider layout top layer

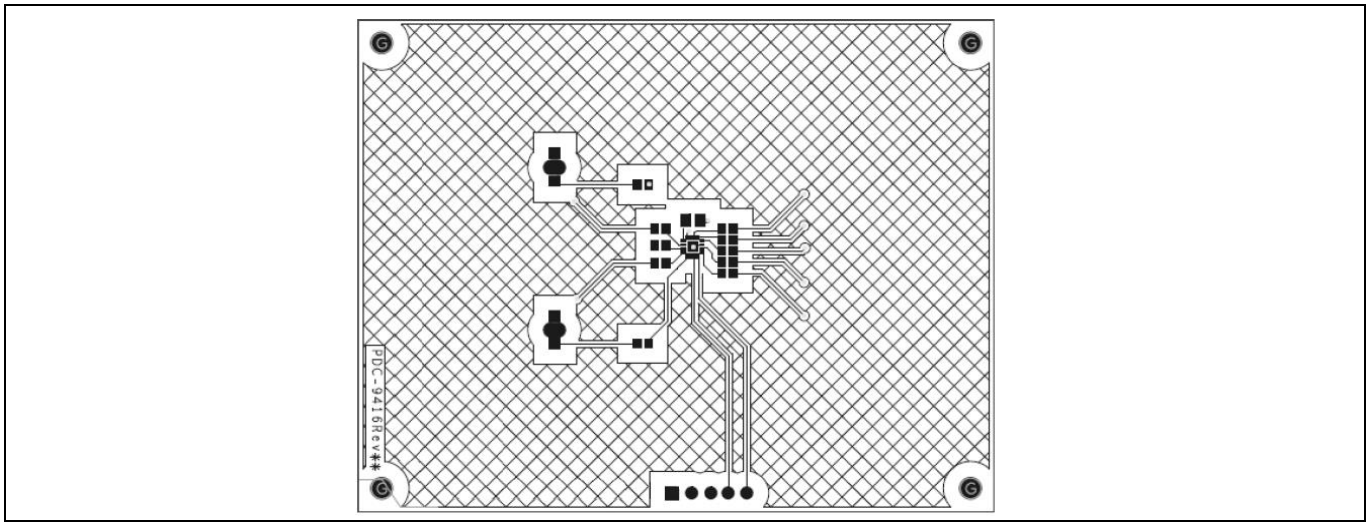

Figure 121 Recommended button and slider layout bottom layer

3.8.13 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 122 illustrates the schematic diagram with these recommendations. C1, C2, and C4 are decoupling capacitors and C3 is the CMOD capacitor.

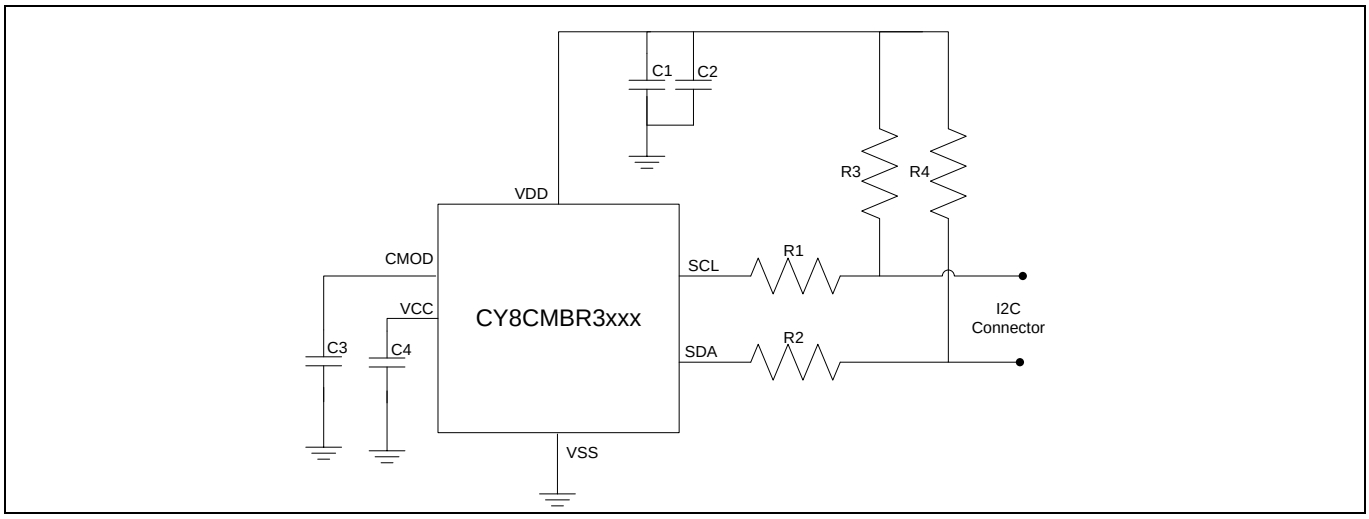

Figure 122 Example schematics for improved SNR

Figure 123 shows an example board layout diagram with placements of decoupling and CMOD capacitors and routing of ground and supply. (Note that the I2C pull-ups present in Figure 122 are not shown in the layout in this figure).

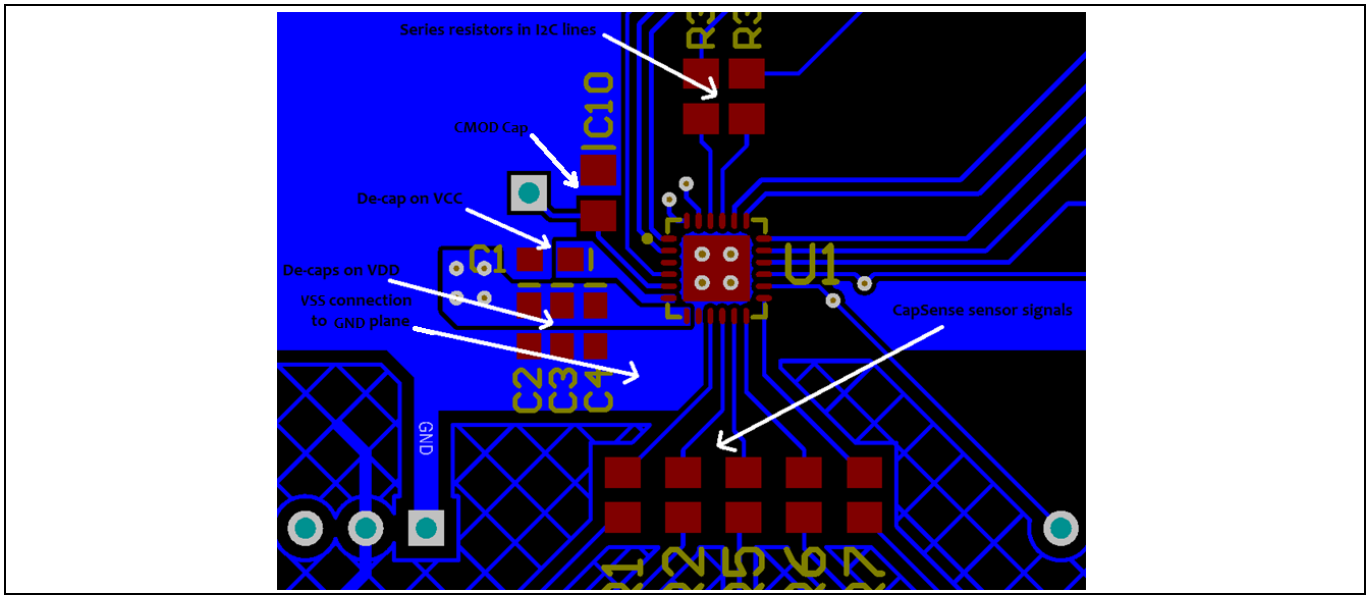

Figure 123 Example board layout for improved SNR

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 124.

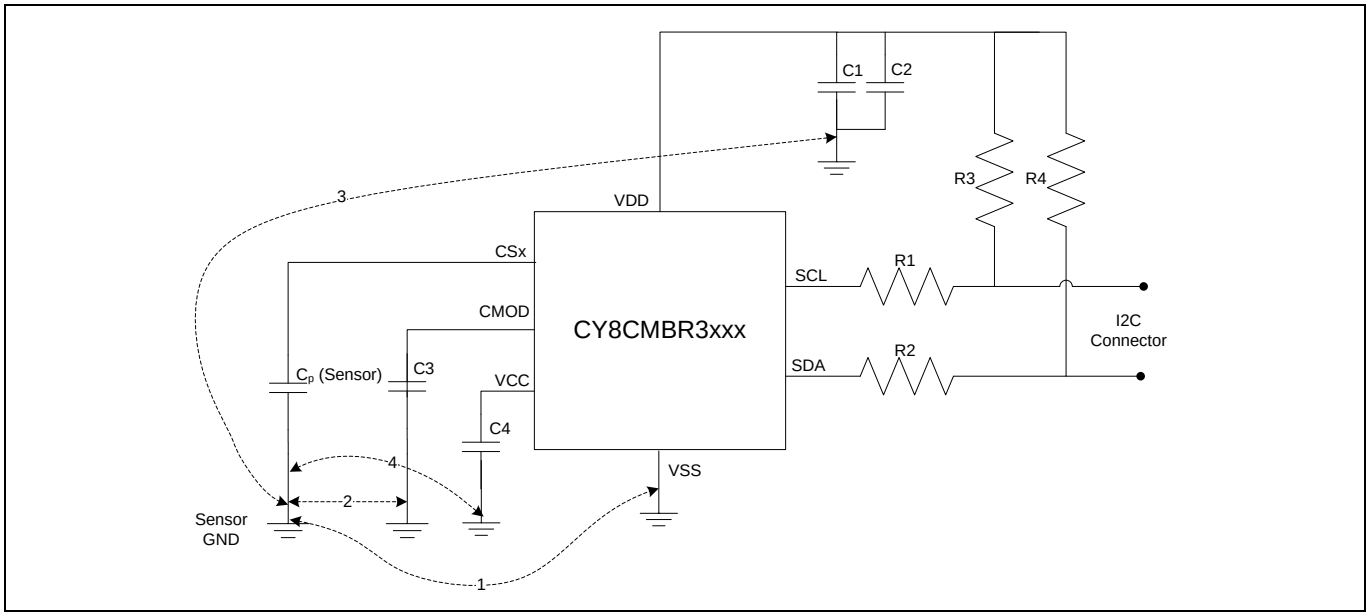

Figure 124 Important GND nodes in CAPSENSE™ design

3.8.14 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 92.
- **Provide directionality to proximity sensing:** The electric field of a proximity sensor is omnidirectional 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 Liquid tolerance, shield electrodes prevent false triggers due to the presence of liquid droplets on the CAPSENSE™ sensor.

3.8.14.1 Shield electrode for proximity sensing

If you want to use shield electrode for reducing sensor CP 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 unidirectional, 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.

3.8.14.2 Shield electrode construction for liquid tolerance

As explained in Liquid tolerance, 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 125 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.

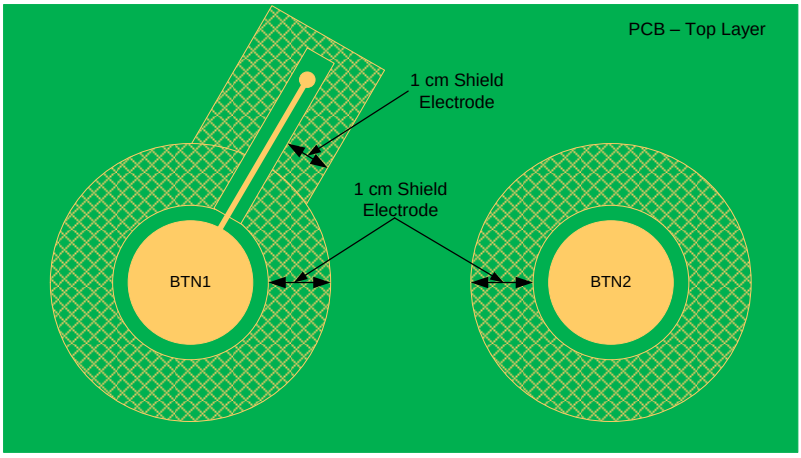

Figure 125 Shield electrode placement when sensor trace is routed in top and bottom layer

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: VDD Plane
- Bottom layer: Hatch fill with 7-mil trace and 70-mil grid (17 percent fill). Hatch fill should be connected to ground.
- The recommended air-gap between the sensor and the shield electrode is 1 mm.

3.8.14.3 Guard sensor

As explained in Liquid tolerance, the guard sensor is a copper trace that surrounds all of the sensors, as Figure 126 shows.

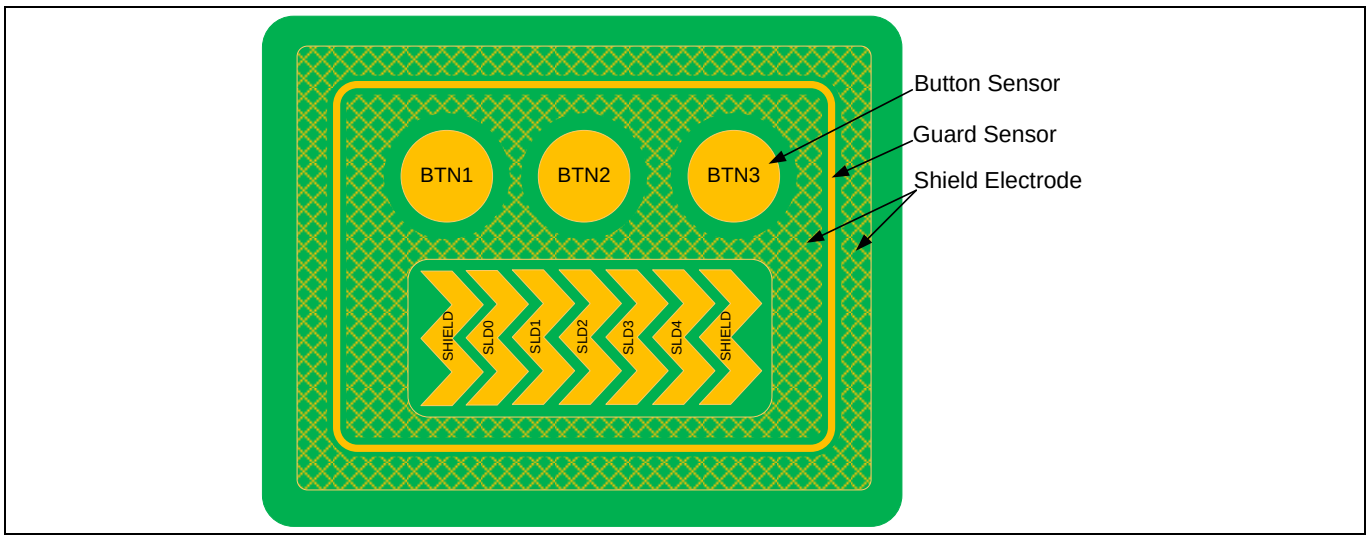

Figure 126 PCB layout with shield electrode and guard sensor

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 126 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.

3.8.15 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. Infineon'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.

See Appendix B: Schematic and layout checklist before you start your design to ensure that the best practices for a CAPSENSE™ design are followed.

3.8.16 CAPSENSE™ system design with ITO

For applications where the CAPSENSE™ sensors need to be transparent (sensors are positioned over the display), CAPSENSE™ sensors can be implemented with ITO. However, if transparent sensors are not required, it is recommended to use copper pads as it provides a higher yield, lower cost, and better performance when compared to ITO sensors.

The ITO sensors can be implemented on a glass substrate or a plastic film (polyethylene terephthalate) substrate. The sensor shape recommended in section Button design for button sensors and section Slider design for slider sensors applies to the sensor design on ITO. Make sure that the sensor length or width (where length is the always the longest dimension) aspect ratio does not exceed 5/3.

Trace length for the sensors should be kept at minimum to reduce the overall sensor resistance. The formula for trace resistance is provided in the following equation:

$$\text{Resistance} = \frac{\text{Trace sheet resistance} \times \text{Trace length}}{\text{Trace width}}$$

Equation 29

Trace resistance is evaluated relative to the sensor resistance. High trace resistance, compared to the sensor resistance, degrades the touch performance. Therefore, it is recommended to keep the sensor trace length as short as possible. The layout guidelines for ITO sensors are summarized in Table 13.

Table 13 Layout guidelines for ITO sensors

Category	Parameter	Min	Typ	Max	Units	Remarks
ITO	Sheet resistance (glass substrate)	-	-	120	Ω/sq	-
	Sheet resistance (film substrate)	-	-	270	Ω/sq	-
	Sensor maximum resistance	-	1	30	kΩ	End-to-end
	Spacing between traces	30	50	100	µm	-
	Routing channel trace width	10	30	50	µm	-

3.9 Example schematic and layout

See the design files of the Development kits corresponding to your chosen device for reference schematics. See any development kit for a reference layout and sensor design. Layout files of the kits are the examples of different PCB stack-up, placement of devices, routing procedure, and implementation of different CAPSENSE™ buttons and sliders. For example:

- CY8CKIT-149 – PSoC™ 4100S Plus prototyping kit has the implementation of recommended fishbone structure of mutual capacitance buttons of different dimensions
- CY8CPROTO-062S3-4343W - PSoC™ 62S3 Wi-Fi Bluetooth® prototyping kit has fishbone button structure optimized for both mutual capacitance and self-capacitance-based sensing.

Visit the corresponding kit webpages for accessing the design files.

You can also see the individual device design guides webpage for sample layouts of all the CAPSENSE™ devices:

- AN66271 - CY8C21x34/B – CAPSENSE™ design guide
- AN66269 - CY8C20x34 – CAPSENSE™ design guide
- AN65973 - CY8C20xx6A/H/AS – CAPSENSE™ design guide
- AN78329 - CY8C20xx7/S – CAPSENSE™ design guide
- AN90071 - CY8CMBR3xxx – CAPSENSE™ desing guide
- AN85951 - PSoC™ 4 and PSoC™ 6 MCU CAPSENSE™ design guide

3.10 PCB assembly and soldering

It is important to follow PCB assembly and soldering standards and guidelines for any CAPSENSE™ design. Although CAPSENSE™ PCB assembly and soldering do not have any specific guidelines, the following specifications and the application notes give the standard and guidelines respectively.

- IPC A-610 - Acceptability of Electronic Assemblies
- AN72845 - Design guidelines for Infineon quad flat no lead (QFN) packaged devices
- AN69061- Design, manufacturing, and handling guidelines for Infineon wafer level chip scale packages

4 Capsense™ selector guide

Infineon is the world leader in capacitive sensing technologies. Our broad range of solutions provide robust noise immunity, enable quick time to market and system scalability, and have replaced more than 5 billion mechanical buttons over the past several years. The CAPSENSE™ portfolio ranges from simple buttons and sliders to more sophisticated solutions integrating other system components to reduce total BOM cost and form factor. Infineon CAPSENSE™ controllers feature best-in-class liquid tolerance and capacitive proximity sensing, which is easy to implement with SmartSense Auto-Tuning, an algorithm that constantly monitors and compensates for environmental conditions.

Salient features include:

- Advanced sensing techniques for easy finger detection through 15 mm of glass or 5 mm of plastic
- Revolutionary and unique SmartSense Auto-Tuning algorithm
- Industry's best solutions for advanced features such as proximity and water tolerance
- Ultra-low power with industry's widest voltage range
- Industry's leading small form factor packaging such as WLCSP (2 mm x 2 mm)

4.1 Defining CAPSENSE™ requirements

You must consider several key system requirements when selecting the best CAPSENSE™ device for your application.

- Configurable/programmable

  Choose configurable controllers if you are looking for an easy and quick-to-market solution without firmware development. These devices can work with or without a host controller. Choose programmable controllers if you need more integration in one chip and more flexibility in designing your application. Infineon offers both hardware-configurable and register-configurable controllers.
- Configuration interface

  Configurable controllers have two configuration interfaces – hardware-based and register-based. Hardware-configurable controllers require external resistors for configuring different features and eliminate the configuration step required during production or during system operation from a host controller. However, register-configurable devices support register-based configuration and comprehensive status reporting through I2C. The same I2C interface can serve as both configuration and host communication interfaces. These controllers support retaining the configuration in EEPROM, enabling standalone operation without a host chip.
- Programming interface

  The flexible design of PSoC™ allows the programming pins to be reused as GPIOs thus reducing the I/O count. However, you should ensure in the design that the external components or long trace-length, required for the GPIO function, do not interfere with programming and vice versa.

  Updating the device firmware in-system from a host controller is an important requirement, particularly for mobile applications. PSoC™ controllers offer two solutions.
- Host-sourced serial programming (HSSP)

  This method uses the dedicated PSoC™ programming interface and does not require a change in the PSoC™ firmware. However, it requires the reset pin of the PSoC™ to be controlled by the host. In this method, the host requires a special firmware that programs the PSoC™ through bit-banging and may require the host-side I/Os to be dedicated for programming.
- Programming through bootloader

This method uses the standard communication interfaces such as I2C, UART, and USB for programming and requires bootloader to be implemented as part of the PSoC™ firmware.

Infineon provides many HSSP and bootloader application notes. See Table 21 to pick one and learn more.

- Number and type of capacitive sensors

  PSoC™'s unique design allows any I/O pin[10] to be used as a CAPSENSE™ sensor of any type, which includes a button, a slider segment, or a proximity sensor. This offers full flexibility to implement a wide variety of capacitive touch sensing applications. In addition to the I/O requirement for the sensors, most of the latest CAPSENSE™ devices require only one external capacitor (CMOD). The following table takes a typical CAPSENSE™ application and calculates the number of IOs required.

Table 14 I/O requirement calculation example

Requirement	I/O Count
10 Button sensors	10
1 Five segment slider	5 (1 pin for each segment)
1 Proximity sensor	1
10 LEDs	10
Liquid tolerance	1 pin for shield electrode[11]
I2C/Programming through bootloader	2
CMOD[12] capacitor	1
Total I/Os	30

- Number of CAPSENSE™ blocks

  Some CAPSENSE™ controllers provide two CAPSENSE™ blocks, which can be used to scan two sensors simultaneously. You should choose these controllers if your application includes many sensors and has a very strict response time requirement.

- Communication interface

  Communication interface is an important requirement if your CAPSENSE™ design involves a host controller. This interface will allow the host controller to configure the device and get the user interface data in the system. I^2C is a popular interface in capacitive touch sensing applications. Infineon controllers support I2C, SPI, and UART depending on the device.

- CPU, Flash, and RAM requirements

  Arm® is the popular CPU choice for embedded applications as it allows portability across different vendors and scalability for complex application requirements. Infineon offers the Arm® Cortex®-M0 based PSoC™ 4 family, the Arm® Cortex®-M3 based PSoC™ 5LP and dual core (Arm® Cortex®-M4 and Cortex®-M0+) based PSoC™ 6 families with different Flash and RAM combinations to suit your needs.

- Liquid tolerance

  Liquid tolerance is a critical requirement for home appliance and other high-reliability applications for preventing false touches in the presence of liquid. CAPSENSE™ controllers support liquid tolerance through the driven-shield[2] technique.

- Operating voltage range – CAPSENSE™ controllers support a wide operating voltage range of 1.71 V to 5.5 V

[10] Some devices do not support sensing on all the I/O pins. See the pinouts section in the corresponding device datasheet to understand the capability of I/O pins.

[11] See Liquid tolerance to learn how CAPSENSE™ controllers enable liquid tolerance using shield electrode.

[12] CSD requires one external capacitor called CMOD for operation. See CAPSENSE™ sigma delta modulator (CSD) sensing method for details.

- Feedback – CAPSENSE™ solutions support standard I/O as well as PWM outputs for buzzers, LEDs, and Haptics
- Package size and pin count – CAPSENSE™ controllers support a wide variety of packages suitable for mobile, home appliances, and industrial markets. The packages available, depending on the device, include QFN, SSOP, SOIC, TQFP, and LQFP. CAPSENSE™ controllers also include chip-scale packages (CSP) than can be as small as 2 mm x 2 mm. After you select a device, See the datasheet for more information on the packages.
- Additional features such as ADC, PWM, Timers, LCD driver, and analog peripherals such as opamps, and comparators.

4.2 CAPSENSE™ portfolio

Infineon offers a wide range of configurable and programmable CAPSENSE™ controllers. Figure 127 presents the overview of the CAPSENSE™ portfolio. The following sections explain the different CAPSENSE™ families.

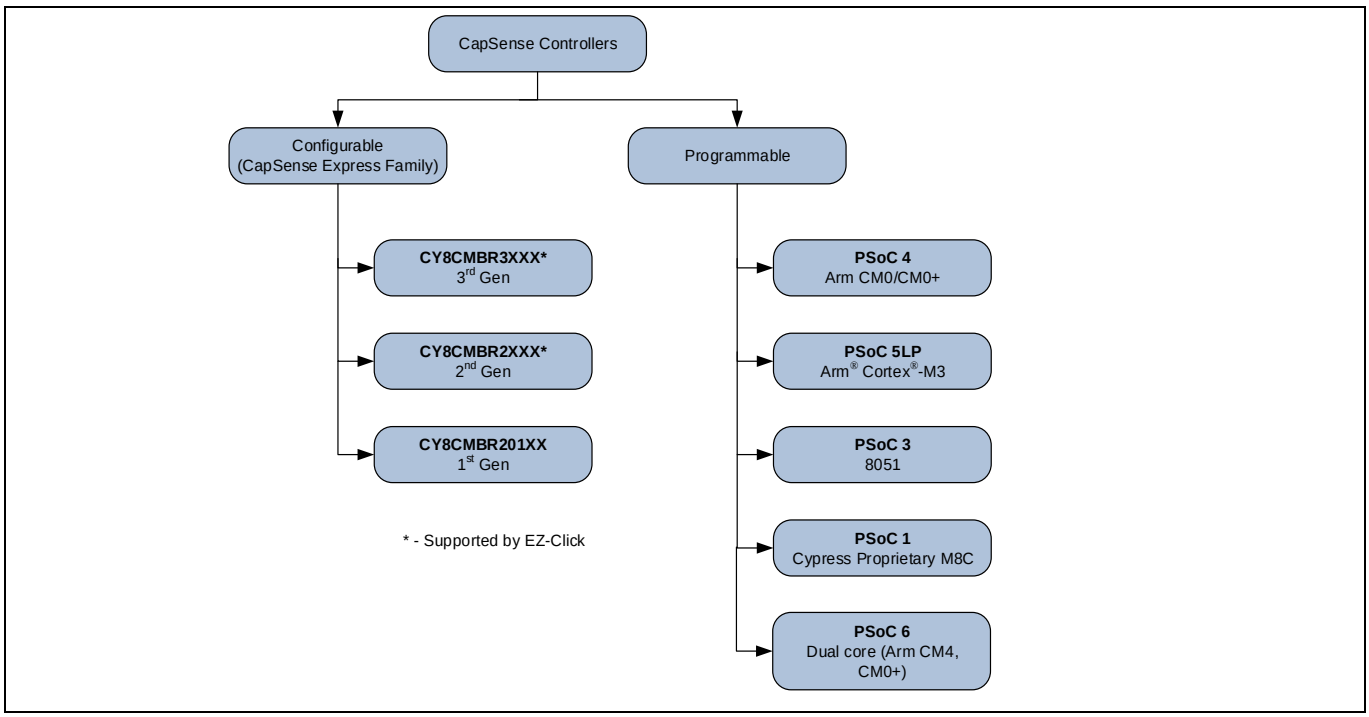

Figure 127 CAPSENSE™ portfolio overview

4.2.1 Configurable CAPSENSE™ controllers (CAPSENSE™ express family)

Configurable CAPSENSE™ controllers, also called CAPSENSE™ Express controllers, eliminate the need for firmware development and make capacitive touch designs very easy and quick. These controllers are configurable either through I²C or through hardware straps (resistors) and feature SmartSense Auto-Tuning (except CY8C201xx family), which eliminates the tuning process and compensates for environmental changes during runtime. The parasitic capacitance (C_P) supported by these devices is limited to 45 pF.

- CY8CMBR3XXX (CAPSENSE™ MBR3) controllers are the third and latest generation of configurable controllers that can be configured over I²C and are supported by the GUI-based configuration tool CAPSENSE™ controllers configuration tools EZ-Click. These devices support up to 16 buttons, up to eight LEDs, up to two proximity sensors, up to two 5-segment sliders, and an operating voltage range of 1.71 V to 5.5 V. See CAPSENSE™ MBR3 solution product overview to understand the difference between the devices available in this family. These controllers provide improved noise immunity over the previous generation devices and Infineon recommends these controllers for new designs.

Capsense™ selector guide

- CY8CMBR2XXX (CAPSENSE™ Express) controllers are second generation configurable devices, which include both hardware-configurable (CY8CMBR20XX) and I²C-configurable (CY8CMBR21XX) devices and the I²C-configurable devices are supported by the CAPSENSE™ controllers configuration tools EZ-Click tool. Choose these devices if you need a hardware-based configuration or advanced LED effects such as dimming and fading in your application. These devices support up to 16 buttons and operating voltage range of 1.71 V to 5.5 V.
- CY8C201xx controllers are first generation configurable devices supporting I2C interface for configuration. These devices support up to 10 I/Os for buttons, LEDs, sliders, and operating voltage range of 2.4 V to 5.5 V. Infineon does not recommend these controllers for new designs.

Table 15 shows the comparison between the different parts in the configurable category.

Table 15 CAPSENSE™ express family features comparison

Features/ device family	CY8CMBR3XXX						CY8CMBR2XXX				
	CY8CMBR 3116	CY8CMBR 3106S	CY8CMBR 3110	CY8CMBR 3108	CY8CMBR 3102	CY8CMBR 3002	CY8CMBR 2016	CY8CMBR 2110	CY8CMBR 2010	CY8CMBR 2044	CY8C201xx
Datasheet	CY8CMBR3XXX						CY8CMBR2016	CY8CMBR2110	CY8CMBR2010	CY8CMBR2044	CY8C201xx
Maximum number of sensors	16	11	10	8	2	2	16	10	10	4	10
Buttons	Up to 16	Up to 11	Up to 10	Up to 8	Up to 2	2	Up to 16	Up to 10	Up to 10	Up to 4	Up to 10
Sliders	No	Up to 2	No	No	No	No	No	No	No	No	1
Proximity	Up to 2	Up to 2	Up to 2	Up to 2	Up to 2	No	No	No	No	No	No
Liquid tolerance	Yes					No					
LED/GPO	Up to 8	0	Up to 5	Up to 4	Up to 1	2	Up to 8	Up to 10	Up to 10	Up to 4	Up to 10
Buzzer	Yes	Yes	Yes	Yes	No	No	Yes	Yes	No	No	No
Configuration interface	I2C	I2C	I2C	I2C	I2C	No (Fixed Function)	Hardware	I2C	Hardware	Hardware	I2C
Communication interface	I2C/GPO	I2C	I2C/GPO	I2C/GPO	I2C/GPO	GPO	GPO	I2C/GPO	GPO	GPO	I2C/GPO
Power-on self-test	Yes										No
Operating voltage	1.71-5.5V										2.4 V-5.25 V
Automotive qualified (AEC-Q100)[13]	No										
Package	24-pin QFN	24-pin QFN	16-pin SOIC	16-pin QFN	8-pin SOIC	8-pin SOIC	48-pin QFN	32-pin QFN	32-pin QFN	16-pin QFN	8/16-pin SOIC, 16-pin QFN

[13] Contact Infineon to know the latest status of the products.

4.2.2 Programmable CAPSENSE™ controllers

Programmable CAPSENSE™ controllers are based on the PSoC™ platform and offer a rich set of analog and digital peripherals along with CAPSENSE™. Infineon provides a great deal of pre-built, production-ready, and GUI-configurable firmware components to speed-up your PSoC™ system design.

- PSoC™ 6 family features 32-bit dual core CPU subsystem (Arm® Cortex®-M4 and CM0+) clocked up to 150 MHz and integrates many advanced peripherals 12-bit SAR ADC, up to two opamps, up to two comparators, PWM, USB-FS host and device peripherals, I2S and PDM channels for audio subsystem. Table 16 shows the comparison between the different sub-families of PSoC™ 6.
- PSoC™ 4 family features 32-bit Arm® CM0 CPU clocked up to 48 MHz and integrates many advanced peripherals – 12-bit SAR ADC, opamps, comparators, PWM, Bluetooth Low Energy (BLE), CAN, and a segment LCD driver. PSoC™ 4 supports up to 54 sensors. PSoC™ 4 is the best low-power mixed-signal architecture and the most cost-effective device family. Table 17 shows the comparison between the different sub-families of PSoC™ 4.
- PSoC™ 5LP family features 32-bit Arm® Cortex®-M3 CPU clocked up to 80 MHz and integrates many advanced peripherals, such as two CAPSENSE™ blocks, 20-bit delta-sigma ADC, 12-bit SAR ADC, 8-bit DAC, opamps, comparators, PWM, USB 2.0 Full-Speed peripheral, CAN, and a segment LCD driver. PSoC™ 5LP supports up to 62 sensors and is suitable for large and complex systems that need higher levels of integration in a single chip. Table 18 shows the comparison between the different PSoC™ 5LP sub-families.
- PSoC™ 3 family features an 8-bit, single-cycle 8051 CPU clocked up to 67 MHz and integrates many advanced peripherals, such as a 20-bit delta-sigma ADC, 8-bit DAC, opamps, comparators, PWM, USB 2.0 Full-Speed peripheral, CAN, and a segment LCD driver. PSoC™ 3 supports up to 62 sensors. Choose PSoC™ 3 if you require the same large-scale integration offered by PSoC™ 5LP but do not require a high-performance CM3 CPU. Table 19 shows the comparison between the different PSoC™ 3 sub-families.

PSoC™ 1 family is the Infineon's first programmable system-on-chip solution featuring the proprietary 8-bit M8C CPU clocked up to 24 MHz. This family supports many CAPSENSE™ features such as buttons, sliders, proximity sensing, and liquid tolerance at an economical price. PSoC™ 1 supports up to 44 sensors and includes many automotive qualified (AEC-Q100) parts compared to other device families. Table 20 shows the comparison between the different PSoC™ 1 sub-families.

Table 16 PSoC™ 6 family features comparison

Features/device family	PSoC™ 61	PSoC™ 62	PSoC™ 63	CY8C62xA, CY8C62x8[14]	CY8C62x5
Datasheet	PSoC™ 61 datasheet	PSoC™ 62: CY8C62x6, CY8C62x7 datasheet	PSoC™ 63 with Bluetooth® LE datasheet	PSoC™ 6: CY8C62x8, CY8C62xA datasheet	CY8C62x5 datasheet
CPU	150-MHz Arm® Cortex®-M4	150-MHz Arm® Cortex®-M4 and 100-MHz Cortex® M0+	150-MHz Arm® Cortex®-M4 and 100-MHz Cortex® M0+	150-MHz Arm® Cortex®-M4, 100-MHz Cortex®-M0+	150-MHz Arm® Cortex®-M4, 100-MHz Cortex®-M0+
Flash	Up to 1 MB Application Flash with 32 KB EEPROM area and 32 KB Supervisory Flash	1 MB Application Flash with 32-KB EEPROM area and 32-KB Supervisory Flash	Up to 1-MB Application Flash, 32-KB emulated EEPROM area, and 32-KB Supervisory Flash	2 MB Application Flash, 32-KB emulated EEPROM area, and 32-KB Supervisory Flash	512 KB Application Flash, 32-KB emulated EEPROM area, and 32-KB Supervisory Flash
SRAM	Up to 288 KB	288 KB	Up to 288 KB	1024-KB	256 KB
CAPSENSE™ architecture	Fourth Generation (supports CSD and CSX)	Fourth Generation (supports CSD and CSX)	Fourth Generation (supports CSD and CSX)	Fourth Generation (supports CSD and CSX)	Fourth Generation (supports CSD and CSX)
CAPSENSE™ average current consumption per sensor	Programmable from 37.5 nA to 2400 µA	Programmable from 37.5 nA to 2400 µA	Programmable from 37.5 nA to 2400 µA	Programmable from 37.5 nA to 609 µA	Programmable from 37.5 nA to 609 µA
SSC/PRS feature	Yes	Yes	Yes	Yes	Yes
ADC	12-bit 1-Msps SAR ADC	12-bit 1-Msps SAR ADC	12-bit 1-Msps SAR ADC	12-bit 1-Msps SAR ADC	12-bit, 1-Msps SAR ADC
Total I/Os	Up to 100[15]	Up to 100[15]	Up to 100[15]	Up to 100[15]	Up to 64[15]
Number of CAPSENSE™ blocks	1	1	1	1	1
Sensor parasitic Capacitance (CP) range	5 pF – 200 pF	5 pF – 200 pF	5 pF – 200 pF	5 pF – 200 pF	5 pF – 200 pF
DAC	1x 12-bit	1x 12-bit	1x 12-bit	0	0
Comparators	2	2	2	2	2
Op-Amps	2	2	2	0	0
Serial communication Block (SCB[16])	9	9	9	13	7
Universal digital block (UDB[17])	12	12	12	0	0
Secure digital host controller (SDHC) block	0	0	0	2	1

[14] These devices support 2 MB Flash memory and have MPNs that belong to both PSoC™ 61 (single CPU) and PSoC™ 62 (dual CPU) series.

[15] In PSoC™ 6 family devices, CAPSENSE™ use is restricted to few ports with switching restrictions on other ports. Follow the recommendations stated in PSoC® 4 and PSoC® 6 MCU CAPSENSE™® Design Guide to achieve the best CAPSENSE™ sensitivity and accuracy.

[16] Each Serial Communication Block (SCB) can function as an I²C, an SPI, or a UART communication block. In addition, PSoC™ Creator provides a Software Transmit UART (TX8) Component for all devices.

[17] Universal Digital Block (UDB) allows implementing custom digital logic. PSoC™ Creator provides a wide array of UDB-based Components such as I²C, I²S, UART, SPI, PWM, and Counter/Timer.

Capsense™ selector guide

Table 17 PSoC™ 4 family features comparison

Features/device family	PSoC™ 4000-Series	PSoC™ 4100-Series	PSoC™ 4200-Series	PSoC™ 4100 M-Series	PSoC™ 4200 M-Series	PSoC™ 4100 BLE-Series	PSoC™ 4200 BLE-Series	PSoC™ 4200 L-Series	PSoC™ 4000 S- Series	PSoC™ 4100 S-Series/ PSoC™ 4100PS	PSoC™ 4100S Plus
Datasheet	CY8C401X	CY8C412X	CY8C424X	CY8C412X-M	CY8C424X-M	PSoC™ 4xx7_BLE – 128 KB Flash, BLE 4.1 PSoC™ 4xx8_BLE – 256 KB Flash, BLE 4.1 PSoC™ 4xx8_BLE 4.2 – 256 KB Flash, BLE 4.2		CY8C424X-L	CY8C40XX-S	CY8C41XX-S CY8C41XX-PS	CY8C41XX-S
CPU	16-MHz Arm® Cortex®-M0	24 MHz Arm® Cortex®-M0	48 MHz Arm® Cortex®-M0	24 MHz Arm® Cortex®-M0	48 MHz Arm® Cortex®-M0	24 MHz Arm® Cortex®-M0	48 MHz Arm® Cortex®-M0	48 MHz Arm® Cortex®-M0	48-MHz Arm® Cortex®-M0+	48-MHz Arm® Cortex®-M0+	48-MHz Arm® Cortex®-M0+
Flash/SRAM	Up to 16 KB/ Up to 2 KB	Up to 32 KB/ Up to 4 KB	Up to 32 KB/ Up to 4 KB	Up to 128 KB/ Up to 16 KB	Up to 128 KB/Up to 16 KB	Up to 256 KB/Up to 32 KB		Up to 256 KB/ Up to 32 KB	Up to 32 KB/ Up to 4 KB	Up to 64 KB/ Up to 8 KB (for PSoC™ 4100PS) Up to 32 KB/Up to 4 KB	Up to 128 KB/ Up to 16 KB
Total I/Os	Up to 20	Up to 36	Up to 36	Up to 55	Up to 55	Up to 36	Up to 36	Up to 98	Up to 36	Up to 36 Up to 38 (For PSoC™ 4100PS)	Up to 54
CAPSENSE™ I/Os (supports button, slider, proximity, shield)	Up to 16	Up to 35	Up to 35	Up to 54	Up to 54	Up to 35	Up to 35	Up to 97	Up to 35	Up to 35 Up to 33 (for PSoC™ 4100PS)	Up to 53
CAPSENSE™ architecture	Third Generation	Third Generation	Third Generation	Third Generation	Third Generation	Third Generation	Third Generation	Third Generation	Fourth Generation	Fourth Generation	Fourth Generation
Sensor parasitic Capacitance (CP) range	5pF – 60pF	5pF – 60pF	5pF – 60pF	5pF – 60pF	5pF – 60pF	5pF – 60pF	5pF – 60pF	5pF – 60pF	5pF – 200pF	5pF – 200pF	5pF – 200pF
Number of CAPSENSE™ blocks	1	1	1	2	2	1	1	2	1	1	1
Self-capacitance and mutual-capacitance support	Both	Self	Both	Both	Both	Both	Both	Both	Both	Both	Both
CAPSENSE™ average current consumption per sensor	6 µA	6 µA	6 µA	6 µA	6 µA	6 µA	6 µA	6 µA	3 µA	3 µA	3 µA
SSC / PRS feature	PRS	PRS	PRS	PRS	PRS	PRS	PRS	PRS	SSC and PRS	SSC and PRS	SSC and PRS

Capsense™ selector guide

Features/device family	PSoC™ 4000-Series	PSoC™ 4100-Series	PSoC™ 4200-Series	PSoC™ 4100 M-Series	PSoC™ 4200 M-Series	PSoC™ 4100 BLE-Series	PSoC™ 4200 BLE-Series	PSoC™ 4200 L-Series	PSoC™ 4000 S- Series	PSoC™ 4100 S-Series/ PSoC™ 4100PS	PSoC™ 4100S Plus
Liquid tolerance	Yes										
SmartSense auto-tuning	Yes								Yes		
ADC	CSD ADC[18]	12-bit SAR at 806 ksps	12-bit SAR at 1 Msps	12-bit SAR at 806 ksps	12-bit SAR at 1 Msps	12-bit SAR at 806 ksps	12-bit SAR at 1 Msps	12-bit SAR at 1 Msps	10-bit Slope ADC at 46 ksps	10-bit Slope ADC at 46 ksps and 12-bit SAR ADC at 1 Msps	10-bit Slope ADC at 46 ksps and 12-bit SAR ADC at 1 Msps
DAC[19]	1x 8-bit 1x 7-bit	1x 8-bit 1x 7-bit	1x 8-bit 1x 7-bit	2x 8-bit 2x 7-bit	2x 8-bit 2x 7-bit	1x 8-bit 1x 7-bit	1x 8-bit 1x 7-bit	2x 8-bit 2x 7-bit	2x 7-bit	2x 7-bit	2x 7-bit
Comparators	1 with fixed 1.2-V threshold	Up to 4	Up to 4	Up to 6	Up to 6	Up to 2	Up to 4	Up to 6	3	Up to 5 Up to 7 (For PSoC™ 4100PS)	2
Op-Amps	0	Up to 2	Up to 2	Up to 4	Up to 4	2	4	Up to 4	0	Up to 2 Up to 4 Op-Amps/PGAs (for PSoC™ 4100PS)	Up to 2
Programmable voltage reference (PVref)	No	No	No	No	No	No	No	No	No	No Up to 4 Channels (for PSoC™ 4100PS only)	No
Voltage DAC (VDAC)	No	No	No	No	No	No	No	No	No	No 13-bit VDAC (for PSoC™ 4100PS only)	No
Serial communication block (SCB[20])	1 (I2C only[21])	2	2	4	4	2	2	4	2	3	5

[18] CAPSENSE™ block is repurposed as an ADC to measure the input voltage of 0-5 V with the result in mV. When this ADC is used, CAPSENSE™ is not available for capacitive sensing.

[19] DAC with current output, except PSoC™ 4000S PSoC™ 4100S, and PSoC™ 4100PS. Each CSD block uses 1x 8-bit DAC in single IDAC mode and 1x 8-bit and 1x 7-bit IDACs in dual-IDAC mode. See AN85951 – PSoC™ 4 and PSoC™ 6 MCU CAPSENSE™ design guide to learn more about these modes. In PSoC™ 4000S, PSoC™ 4100S, and PSoC™ 4100PS, the CSD block uses 1x 7-bit DAC in single IDAC mode and two 1x 7-bit IDACs in dual-IDAC mode.

[20] Each Serial Communication Block (SCB) can function as an I²C, an SPI, or a UART communication block. In addition, PSoC™ Creator provides a Software Transmit UART (TX8) Component for all devices.

[21] The 16-pin QFN device in this family supports the I²C bus voltage to be different from the device operating voltage.

Capsense™ selector guide

Features/device family	PSoC™ 4000-Series	PSoC™ 4100-Series	PSoC™ 4200-Series	PSoC™ 4100 M-Series	PSoC™ 4200 M-Series	PSoC™ 4100 BLE-Series	PSoC™ 4200 BLE-Series	PSoC™ 4200 L-Series	PSoC™ 4000 S- Series	PSoC™ 4100 S-Series/ PSoC™ 4100PS	PSoC™ 4100S Plus
Universal digital block (UDB[22])	0	0	Up to 4	0	4	0	4	8	0	0	0
TCPWM[23]	1	4	4	8	8	4	4	8	5	5 8 (for PSoC™ 4100PS)	8
Segment LCD drive	No	Up to 4 commons, 32 segments		Up to 4 commons, 51 segments		Up to 4 commons, 32 segments		Up to 8 commons, 64 segments	Up to 8 commons, 36 segments		Up to 4 commons, 49 segments
Bluetooth low energy (BLE)	No					Yes		No	No	No	No
CAN 2.0[24]	No	No	No	No	Yes	No	No	Yes	No	No	Yes
USB	No	No	No	No	No	No	No	Yes	No	No	No
Operating voltage	1.71-5.5 V										
Automotive qualified (AEC-Q100)[25]	Yes			No							
Package	8/16-pin SOIC 16/24-pin QFN 28-pin SSOP 16-pin WLCSP[26]	28-pin SSOP 40-pin QFN 44-pin TQFP 48-pin LQFP 35-pin WLCSP[26]		48-pin TQFP 64-pin TQFP 68-pin QFN		56-pin QFN 68-pin WLCSP[26] 76-pin WLCSP[26]		124-BGA 68-QFN 64-pin TQFP 48-pin TQFP 48-pin TQFP-USB	48-pin TQFP 24-pin QFN 32-pin QFN 25-pin WLCSP	For PSoC™ 4100S, 48-pin TQFP 40-pin QFN 32-pin QFN 35-pin WLCSP For PSoC™ 4100PS, 28-pin SSOP 45-pin WLCSP 48-pin TQFP 48-pin QFN	44-pin TQFP 64-pin TQFP

[22] Universal Digital Block (UDB) allows implementing custom digital logic. PSoC™ Creator provides a wide array of UDB-based Components such as I^2C, I^2S, UART, SPI, PWM, and Counter/Timer.

[23] Timer, Counter, PWM block.

[24] Controller Area Network.

[25] Contact Infineon to know the latest status of these products

[26] Wafer-Level Chip-Scale Package.

Capsense™ selector guide

Table 18 PSoC™ 5LP family features comparison

Features/device family	PSoC™ 5200	PSoC™ 5400	PSoC™ 5600	PSoC™ 5800
Datasheet	CY8C52XX	CY8C54XX	CY8C56XX	CY8C58XX
CPU and speed	Arm® Cortex®-M3 clocked up to 80 MHz			
Flash, SRAM, EEPROM	Up to 256 KB, Up to 64 KB, 2 KB			
Total I/Os	Up to 72			
CAPSENSE™ I/Os (supports button, slider, proximity, shield)	Up to 62			
Number of CAPSENSE™ blocks	2			
Liquid tolerance	Yes			
SmartSense auto-tuning	Yes			
ADC	1x 12-bit SAR	1x 12-bit SAR or 1x 12-bit Del-Sig	2x 12-bit SAR or 1x 12-bit Del-Sig and 1x 12-bit SAR	2x 12-bit SAR and 1x 20-bit Del-Sig
DAC	1x 8-bit	2x 8-bit	4x 8-bit	4x 8-bit
Comparators	2	4	4	4
Op-Amps	0	2	4	4
SC/ST analog block[27]	0	2	4	4
Universal digital block (UDB[28])	Up to 24			
16-Bit timer/PWM	4			
Digital filter block (DFB[29])	No	No	Yes	Yes
Segment LCD drive	Up to 46x16 segments			
USB 2.0 full-speed peripheral	Yes			
CAN 2.0[30]	No	No	Yes	Yes
Package	68-pin QFN 100-pin TQFP 99-pin WLCSP[31]			
Automotive qualified (AEC-Q100)	No			
Operating voltage	1.71 – 5.5 V			

[27] Switched Capacitor/Continuous Time blocks, which are programmable to work as Opamp, Unity Gain Buffer, Programmable Gain Amplifier, Transimpedance Amplifier (TIA), etc.

[28] Universal Digital Block (UDB) allows implementing custom digital logic. PSoC™ Creator provides wide variety of UDB based components such as I2C, I2S, UART, SPI, LIN Slave, PWM, Counter/Timer etc.

[29] Digital Filter Block (DFB) is programmable to perform IIR and FIR digital filters and several custom functions. It is capable of implementing filters with up to 64 taps and of 48-bit single-cycle multiply-accumulate (MAC) operation.

[30] Controller Area Network.

[31] Wafer-Level Chip-scale Package.

Capsense™ selector guide

Table 19 PSoC™ 3 family features comparison

Features/device family	PSoC™ 3200	PSoC™ 3400	PSoC™ 3600	PSoC™ 3800
Datasheet	CY8C324X	CY8C346X	CY8C366X	CY8C386X
CPU and speed	Single-Cycle 8051 clocked up to 50 MHz		Single-Cycle 8051 clocked up to 67 MHz	
Flash, SRAM, EEPROM	Up to 64 KB, Up to 8 KB, Up to 2 KB			
Total I/Os	Up to 72			
CAPSENSE™ I/Os (supports button, slider, proximity, shield)	Up to 62			
Number of CAPSENSE™ blocks	2			
Liquid tolerance	Yes			
SmartSense auto-tuning	Yes			
ADC	1x 12-bit Del-Sig	1x 12-bit Del-Sig	1x 12-bit Del-Sig	1x 20-bit Del-Sig
DAC	1x 8-bit	2x 8-bit	Up to 4x 8-bit	Up to 4x 8-bit
Comparators	2	4	Up to 4	Up to 4
Op-Amps	0	2	Up to 4	Up to 4
SC/ST analog block[32]	0	2	Up to 4	Up to 4
Universal digital block (UDB[33])	Up to 24			
16-Bit timer/PWM	4	4	Up to 4	4
Digital filter block (DFB[34])	No	No	Yes	Yes
Segment LCD drive	Up to 46x16 segments			
USB 2.0 full-speed peripheral	Yes			
CAN 2.0[35]	No	Yes	Yes	Yes
Operating voltage	1.71-5.5 V			
Automotive qualified (AEC-Q100)	No			
Package	48-pin SSOP 48-pin QFN 68-pin QFN 100-pin TQFP 72-pin WLCSP[36]	48-pin SSOP 48-pin QFN 68-pin QFN 100-pin TQFP	48-pin SSOP 48-pin QFN 68-pin QFN 100-pin TQFP 72-pin WLCSP[36]	48-pin SSOP 48-pin QFN 68-pin QFN 100-pin TQFP 72-pin WLCSP[36]

Table 20 PSoC™ 1 family features comparison

Features/device family	CY8C20xx7/S	CY8C21x34/B	CY8C20xx6A/S	CY8C20xx6H	CY8C24x94	CY8C22x45	CY8C28xxx	CY8C20x34[37]
Datasheet	CY8C20xx7/S	CY8C21x34/B	CY8C20xx6A/S	CY8C20xx6H	CY8C24x94	CY8C22x45	CY8C28xxx	CY8C20x34
CPU and speed	24 MHz M8C							12 MHz M8C
Flash/SRAM	Up to 32 KB/ Up to 2 KB	8 KB/512 Bytes	Up to 32 KB/ Up to 2 KB	Up to 16 KB/ Up to 2 KB	16 KB/1 KB	16 KB/1 KB	16 KB/1 KB	8 KB/ 512 Bytes

[32] Switched Capacitor/Continuous Time blocks, which are programmable to work as Op-Amp, Unity Gain Buffer, Programmable Gain Amplifier, and Transimpedance Amplifier (TIA), etc.

[33] Universal Digital Block (UDB) allows implementing custom digital logic. PSoC™ Creator provides a wide array of UDB based components such as I2C, I2S, UART, SPI, LIN Slave, PWM, Counter/Timer etc.

[34] Digital Filter Block (DFB) is programmable to perform IIR and FIR digital filters and several custom functions. It is capable of implementing filters with up to 64 taps and of 48-bit single-cycle multiply-accumulate (MAC) operation.

[35] Controller Area Network.

[36] Wafer-Level Chip-scale package.

[37] Not recommended for new designs.

Capsense™ selector guide

Features/device family	CY8C20xx7/S	CY8C21x34/B	CY8C20xx6A/S	CY8C20xx6H	CY8C24x94	CY8C22x45	CY8C28xxx	CY8C20x34[37]
CAPSENSE™ I/Os (supports buttons, sliders, proximity)	31	24	33	33	44	37	41	25
Number of CAPSENSE™ blocks	1	1	1	1	1	2	Up to 2	1
Liquid tolerance	Yes	Yes	No	No	Yes	Yes	Yes	No
SmartSense auto-tuning	Yes (only on -S parts)	Yes (only on –B parts)	Yes	Yes	No	Yes	Yes	No
Comparators	Yes	Yes	Yes	Yes	-	Yes	Yes	Yes
ADC	-	8- and 10-bit	8- and 10-bit	8- and 10-bit	7- to 13-bit	10-bit SAR	Up to 14-bit	-
DAC	-	-	-	-	6,8 and 9-bit	8-bit	Up to 9-bit	-
Amplifiers	-	-	-	-	Yes	-	Yes	-
I2C	Master and slave interface							
SPI								
UART	Transmitter – Software	UART	Transmitter – Software	Transmitter – Software	UART	UART	UART	Transmitter – Software
USB	-	-	-	Full Speed USB	Full Speed USB	-	-	-
Timer	16-bit timer	8- to 32-bit timer/ counter	16-bit timer	16-bit timer	8- to 32-bit timer/ counter	8- to 32-bit timer/ counter	8- to 32-bit timer/counter	13-bit timer
PWM	-	8- to 32-bit deadband option	-	-	8- to 32-bit	8- to 32-bit	8- and 16-bit	-
LCD	16x2 character LCD interface and Up to 4 commons segment LCD drive							
EEPROM	Emulation							
Haptics	-	-	-	Yes	-	-	-	-
Operating voltage	1.71 V–5.5 V	2.4 V–5.25 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	3.0 V-5.25 V	2.4 V-5.25 V
Automotive qualified (AEC-Q100)	No	Yes	Yes	No	Yes	Yes	No	Yes
Package	16/24/32/48-pin QFN 16-pin SOIC 30-pin WLCSP[38]	16-pin SOIC 20/28/56-pin SSOP 32-pin QFN	16/24/32/48-pin QFN 48-pin SSOP 30-pin WLCSP[38]	24-pin QFN 32-pin QFN	56-pin QFN 68-pin QFN	28-pin SOIC 44-pin TQFP	20/28-pin SSOP 44-pin TQFP 48-pin QFN	8/16-pin SOIC 28-pin SSOP 16/24/32-pin QFN 30-pin WLCSP[38]

[38] Wafer-Level Chip-scale Package.

5 CAPSENSE™ resources

Infineon provides many resources to simplify the CAPSENSE™ design process. To make it easier for you to find what you need from this rich selection, this section organizes the available documentation based on a typical workflow for a CAPSENSE™ design. Table 21 is a quick reference for finding the correct documentation.

Table 21 CAPSENSE™ resources navigator

I want to	Where to go?
Evaluate CAPSENSE™	1. In this document, you can learn about – Capacitive touch sensing method – CAPSENSE™ tuning – CAPSENSE™ widgets – Liquid tolerance – Proximity sensing 2. Buy a Development kits. Use the CY3280-MBR3 – CAPSENSE™ MBR3 evaluation kit if you want to quickly develop a CAPSENSE™ solution without firmware development. Use CY8CKIT-040 – PSoC™ 4000 pioneer develipment kit to evaluate the complete programmability of the CAPSENSE™ family of devices. 3. Go through the code examples provided with any of our Development kits. The user guide has step-by-step procedures for running the code examples.
Select a CAPSENSE™ part	1. See Capsense™ selector guide in this guide. 2. Once you have selected a part, see the device-specific Design Guide to help with your design.
Design CAPSENSE™ hardware	1. Watch the CAPSENSE™ layout best practices videos. 2. See the device-specific Design Guide for schematic, layout, and overlay design guidelines, sample schematics and layouts, and other design considerations. 3. Find all schematic symbols and footprints for Infineon controllers at this page. 4. Visit the webpages of different development kits listed in Development Kit for the example schematic and layout files. 5. You can use Altium Designer sensor patterns such as a button, or slider, to make your layout design easier. See this page to learn more. 6. Submit your design for review through Infineon Tech Support. You need to register at Infineon website to be able to contact tech support. Infineon recommends PDF prints for the schematic and Gerber files with layer information for the layout.
Configure a CAPSENSE™ MBR[39] device	1. Download the CAPSENSE™ controllers configuration tools EZ-Click software tool. 2. See the device-specific Design Guide for configuration instructions. Design guides also present other methods of configuration.
Develop firmware	This step is applicable only to the programmable[40] CAPSENSE™ devices. 1. Download a development environment.

[39] MBR stands for Mechanical Button Replacement. CAPSENSE™ MBR devices are configurable but require no firmware development. See the Capsense™ selector guide for details.

[40] Programmable devices support various programmable digital and analog peripherals in addition to CAPSENSE™ and require firmware development. See the Capsense™ selector guide for details.

CAPSENSE™ resources

I want to	Where to go?
	Use PSoC™ Creator for PSoC™ 3, PSoC™ 4, PSoC™ 5LP systems. Use PSoC™ Designer Archive for PSoC™ 1 systems. For PSoC™ 6 systems: – Use PSoC™ Creator or ModusToolbox™ for CY8C62x6, CY8C62x7 devices – Use PSoC™ Creator or ModusToolbox™ for CY8C63xx devices – Use ModusToolbox™ software for CY8C62x8, CY8C62xA devices – Use ModusToolbox™ software for CY8C62x5 devices See Software tools to learn more. 2. Study the example code. Use the code examples provided with the Kits and the IDEs. See the CAPSENSE™ Controller Code Examples document. Implement dynamic reconfiguration with based CAPSENSE™ devices, AN49079 – PSoC™ 1 CAPSENSE™ Plus: Dynamically Configuring CAPSENSE™ 3. See PSoC™ 3/5 CAPSENSE™ component datasheet/PSoC™ 4 capacitive sensing (CAPSENSE™) component datasheet/PSoC™ 6 capacitive sensing (CAPSENSE™) component datasheet /CAPSENSE middlewar library 4.0 API guide. 4. Optimize the code. See these application notes to learn how: AN89610 - PSoC™ 4 Arm® Cortex® code optimization AN60630 - PSoC™ 3 8051 code and memory optimization AN60486 - PSoC™ 1 M8C ImageCraft C code optimization
Tune a CAPSENSE™ design	1. Enable SmartSense[41] to reduce or eliminate manual tuning. 2. See the CAPSENSE™ Manual Tuning section in the device-specific Design Guide. 3. See AN2397 – PSoC™ 1 and CAPSENSE™ controllers – CAPSENSE™ data monitoring tools to learn how to monitor and log real-time sensor data over I2C or UART.
Design CAPSENSE™ proximity sensing	1. In this document, you can learn about Proximity sensing 2. Evaluate proximity sensing using CY8CKIT-024 - CAPSENSE™ Proximity Shield along with a CY8CKIT-042 - PSoC™ 4 pioneer kit. 3. See Design CAPSENSE™ hardware in this table above to help with your schematic and layout design. 4. Explore additional information on proximity design guidelines, AN92239 - Proximity sensing with CAPSENSE™. This application note also provides the code examples.
Design CAPSENSE™ liquid level sensing	1. Evaluate liquid level sensing using CY8CKIT-022 - CAPSENSE™ Liquid Level Sensing Shield along with a CY8CKIT-042 - PSoC™ 4 pioneer kit. 2. Explore information on theory, sensor layout, and other system design guidelines, AN202478 - PSoC™ 4 - Capacitive Liquid-Level Sensing. 3. Go through the code example CE202479.

[41] Infineon's SmartSense Auto-Tuning algorithm automatically sets sensing parameters for optimal performance and continuously compensates for system, manufacturing and environmental changes.

CAPSENSE™ resources

I want to	Where to go?
Design for low power	AN210998 - PSoC™ 4 Low-Power CAPSENSE™ design CE95288 - CAPSENSE™ Low Power with PSoC™ 4 AN86233 - PSoC™ 4 MCU Low-Power modes and power reduction techniques AN77900 - PSoC™ 3 and PSoC™ 5LP Low-Power modes and power reduction techniques AN47310 - PSoC™ 1 power savings using Sleep mode See the device-specific Design Guide for more low power techniques.
Solve ESD, EMC, and EFT issues	AN96475 - Design considerations for Electrical Fast Transient Immunity of a CAPSENSE™ system AN80994 - Design considerations for Electrical Fast Transient (EFT) Immunity AN72362 - Reducing radiated emissions in Automotive CAPSENSE™ applications See the device-specific Design Guide f-or ESD and EMC design considerations.
Implement class-B safety functions	AN89056 - PSoC™ 4 – IEC 60730 Class B and IEC 61508 SIL Safety Software library AN78175 - PSoC™ 3 and PSoC™ 5LP - IEC 60730 Class B Safety Software library AN79973 - PSoC™ 3 and PSoC™ 5LP CAPSENSE™ CSD - IEC 60730 Class B Safety Software library AN81828 - PSoC™ 1 – IEC 60730 Class B Safety Software library
Program the CAPSENSE™ controller in-system from an external host	For programming through a bootloader: AN73854 - PSoC™ Creator - Introduction to bootloaders AN97060 - PSoC™ 4 Bluetooth® LE and PRoC Bluetooth® LE - Over-The-Air (OTA) Device Firmware Upgrade (DFU) Guide AN2100 - Bootloader: PSoC™ 1 For programming through a dedicated programming interface: AN84858 - PSoC™ 4 programming using an external microcontroller (HSSP) AN73054 - PSoC™ 3 and PSoC™ 5LP programming using an external microcontroller (HSSP) AN44168 - PSoC™ 1 device programming using external microcontroller (HSSP) AN59389 - Host Sourced Serial Programming for CY8C20xx6A, CY8C20xx6AS, CY8C20xx6L and CY8C20xx7/S
Learn CAPSENSE™ design tips and tricks	See the extensive Knowledge Base Articles on the Infineon website. Use the **Product Category** filters to narrow the results.

5.1 CAPSENSE™ design guides and application notes

Our technical documentation includes special CAPSENSE™ Design Guides. Design guides contain all the information such as schematic and layout guidelines, tuning steps, ESD and EMC design considerations, and other information required to complete a CAPSENSE™ design with a particular part family. Also listed below are the Getting Started Applications Notes for the various device families.

Design guides for PSoC™ 3, PSoC™ 4, and PSoC™ 5LP devices

- AN85915 - PSoC™ 4 and PSoC™ 6 MCU CAPSENSE™ design guide
- AN75400 - PSoC™ 3 and PSoC™ 5LP CAPSENSE™ design guide

Design guides for the CAPSENSE™ express family

- AN90071 - CY8CMBR3XXX CAPSENSE™ design guide
- AN76000 - CY8CMBR2110 CAPSENSE™ design guide
- AN73034 - CY8CMBR2016 CAPSENSE™ design guide
- AN75999 - CY8CMBR2010 CAPSENSE™ design guide
- AN66308 - CY8CMBR2044 CAPSENSE™ design guide

Getting started application notes for PSoC™ 1, PSoC™ 3, PSoC™ 4, and PSoC™ 5LP

- AN75320 - Getting started with PSoC™ 1
- AN54181 - Getting started with PSoC™ 3
- AN79953 - Getting started with PSoC™ 4 MCU
- AN77759 - Getting started with PSoC™ 5LP

5.2 Additional CAPSENSE™ resources

5.2.1 Website

At the Infineon CAPSENSE™ Controllers website, you can access all the reference materials such as datasheets, reference manuals, application notes, and code examples.

5.3 Software tools

5.3.1 Integrated development environments

5.3.1.1 ModusToolbox™

Infineon introduces the ModusToolbox™ software suite for the development of PSoC™ 6 based CAPSENSE™ applications. You can download ModusToolbox™ from here. Before you start working with this software, Infineon recommends that you go through the ModusToolbox™ tools package quick start guide and ModusToolbox™ tools package user guide. If you have ModusToolbox™ IDE installed in your system, you can create a CAPSENSE™ application. The ModusToolbox™ CAPSENSE™ Configurator user guide explains steps for simple CAPSENSE™ Buttons and a Linear Slider example to get started. You can also see the ModusToolbox™ CAPSENSE™ Tuner user guide for tuning your CAPSENSE™ design.

The related documents are as follows:

- ModusToolbox™ tools package release notes
- ModusToolbox™ tools package installation guide
- ModusToolbox™ tools package user guide
- ModusToolbox™ tools package quick start guide
- ModusToolbox™ CAPSENSE™ Configurator user guide
- ModusToolbox™ CAPSENSE™ Tuner user guide
- ModusToolbox™ Device Configurator user guide
- ModusToolbox™ Smart I/O Configurator user guide

Note that PSoC™ Creator does not support all the PSoC™ 6 devices. The CY8C62x8 and CY8C62xA device families are supported in ModusToolbox™ only. The CY8C6xx7 device family is supported in ModusToolbox™ and PSoC™ Creator 4.2.

5.3.1.2 PSoC™ Creator

PSoC™ Creator is a free Windows-based Integrated Development Environment (IDE). It enables concurrent hardware and firmware design of PSoC™ 3, PSoC™ 4, and PSoC™ 5LP systems. See Figure 128. With PSoC™ Creator, you can:

1. Drag and drop Components to build your hardware system design in the main design workspace.
2. Co-design your application firmware with the PSoC™ hardware.
3. Configure Components using configuration tools.
4. Explore the library of 100+ Components.
5. Review Component datasheet.

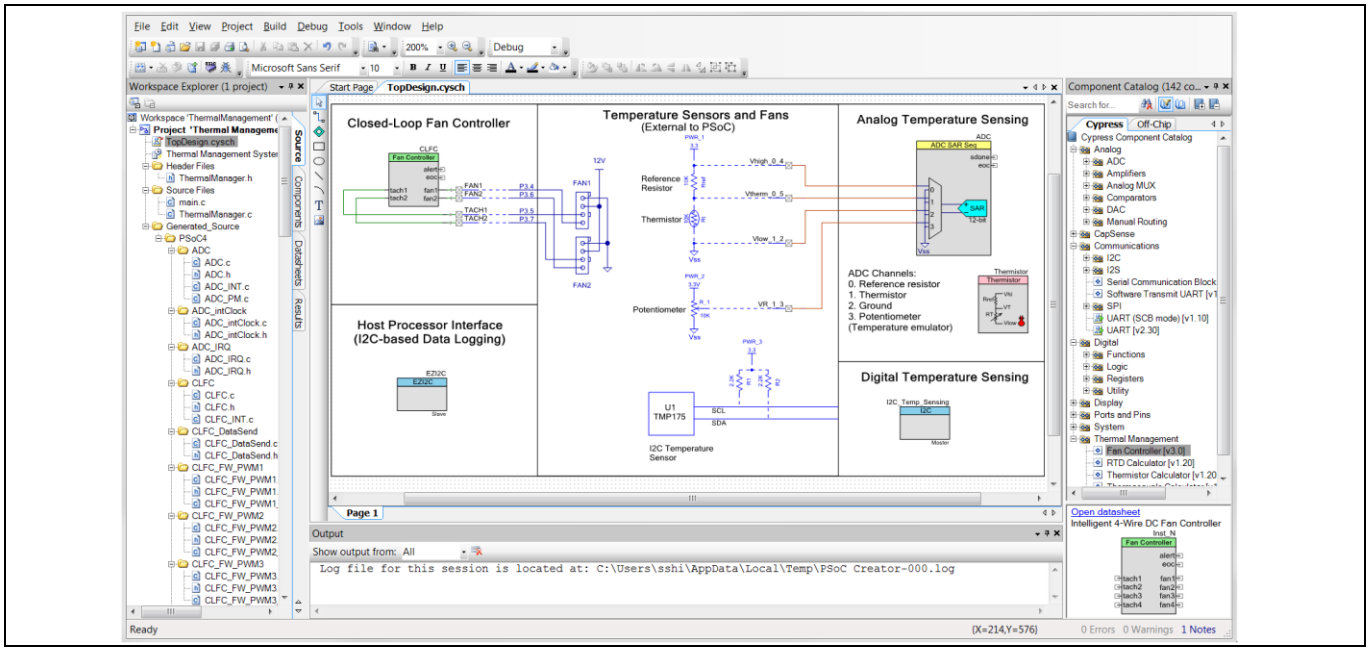

Figure 128 PSoC™ creator features

5.3.1.3 PSoC™ Designer

PSoC™ Designer Archive is a free Windows-based Integrated Design Environment (IDE) offered by Infineon for developing PSoC™ 1 systems. Develop your applications using a library of pre-characterized analog and digital peripherals in a drag-and-drop design environment. Then, customize your design leveraging the dynamically generated API libraries of code. Figure 129 shows the PSoC™ Designer windows.

Note: This is not the default view.

1. **Global Resources** – all device hardware settings.
2. **Parameters** – the parameters of the currently selected User Modules
3. **Pinout** – information related to device pins
4. **Chip-Level Editor** – a diagram of the resources available on the selected chip
5. **Datasheet** – the datasheet for the currently selected UM
6. **User Modules** – all available User Modules for the selected device
7. **Device Resource Meter** – device resource usage for the current project configuration
8. **Workspace** – a tree-level diagram of files associated with the project
9. **Output** – output from project build and debug operations

Note: For detailed information on PSoC™ Designer, go to PSoC™ Designer > Help > Documentation > Designer Specific Documents > IDE User Guide.

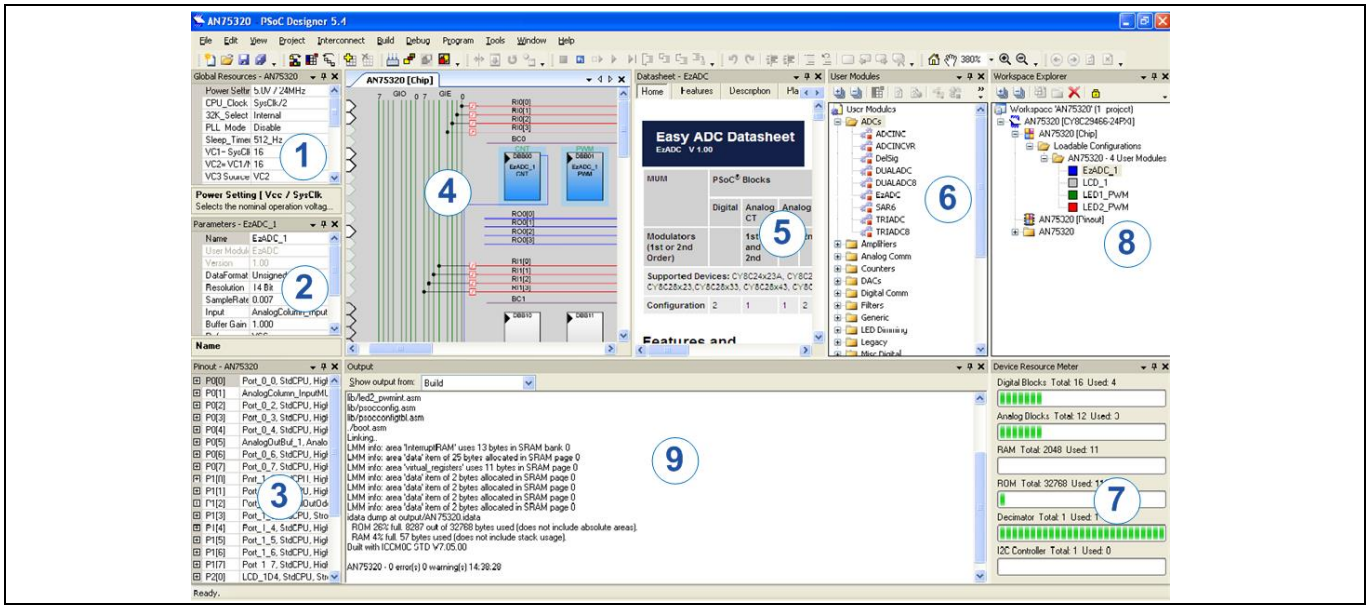

Figure 129 PSoC™ designer layout

5.3.1.4 Programmer

PSoC™ Programmer is a Windows-based, stand-alone application used to program PSoC™ devices using the hardware tools MiniProg3 (CY8KIT-002), MiniProg4 (CY8CKIT-005-A), or KitProg, the on-board programmer integrated with the development kits. PSoC™ Programmer also provides APIs to develop your own application utilizing Infineon's programmers and bridge devices in C, C#, Perl, and Python languages. See the PSoC™ Programmer Component Object Model (COM) Interface Guide available under *[install path]\Programmer\Documents* folder for more details. PSoC™ Creator and PSoC™ Designer provide in-window programming interface that in turn invokes the PSoC™ Programmer APIs.

5.3.2 Data monitoring tools

These tools help you to debug, monitor, and tune CAPSENSE™ designs. For more details on the tools, see the application note AN2397 – PSoC™ 1 and CAPSENSE™ Controllers – CAPSENSE™ data monitoring tools.

5.3.3 CAPSENSE™ tuner

CAPSENSE™ Tuner is a graphic user interface (GUI) tool available in PSoC™ Creator to debug and tune CAPSENSE™ applications over I2C. To launch the Tuner GUI, right-click the CAPSENSE™ symbol placed in the schematic editor and select **Launch Tuner**, as shown in Figure 130.

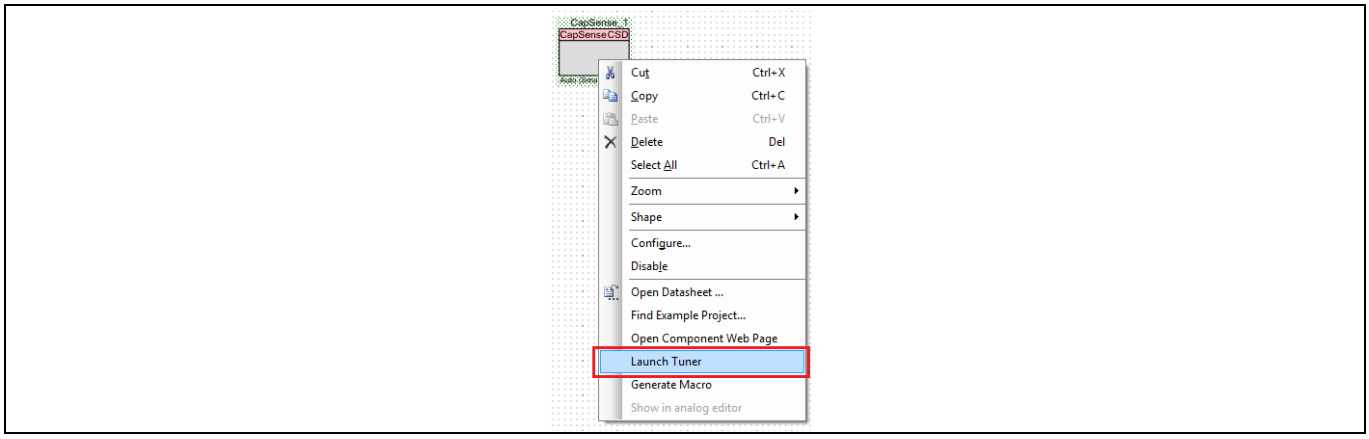

Figure 130 Launching tuner

5.3.4 EZ-Click

CAPSENSE™ controllers configuration tools EZ-Click is a simple yet powerful software tool that enables development of CAPSENSE™ MBR[42] solutions. The tool allows you to configure, debug, monitor real time sensor output, and run production-line system diagnostics of all register-configurable CAPSENSE™ MBR families.

5.3.5 Bridge control panel

Bridge Control Panel is a Windows-based software tool installed along with PSoC™ Programmer. This tool is used along with MiniProg3 (CY8CKIT-002) or MiniProg4 (CY8CKIT-005-A) to monitor and log serial communication data from I²C slave devices. This tool can also receive data from UART devices.

5.4 Development kits

5.4.1 PSoC™ 4 development kits

5.4.1.1 Pioneer kits

The pioneer kits are an easy-to-use and inexpensive development platform that support Arduino™ compatible shields.

- CY8CKIT-040 – PSoC™ 4000 pioneer development kit for PSoC™ 4000 devices
- CY8CKIT-042 – PSoC™ 4 pioneer kit for PSoC™ 4100/4200 devices
- CY8CKIT-044 – PSoC™ 4 M-Series pioneer kit for PSoC™ 4 M-Series devices
- CY8CKIT-046 – PSoC™ L-Series pioneer kit for PSoC™ 4 L-Series devices

5.4.1.2 Shield kits

Shield kits are designed to be Arduino compatible and to work with the PSoC Pioneer kits.

- CY8CKIT-022 - CAPSENSE™ Liquid Level Sensing Shield
- CY8CKIT-024 - CAPSENSE™ Proximity Shield

[42] MBR stands for Mechanical Button Replacement. CAPSENSE™ MBR devices include CY8CMBR3xxx and CY8CMBR2xxx families. See the Capsense™ selector guide for details.

5.4.1.3 Prototyping kits

Prototyping kits are low-cost platforms for prototyping products using PSoC™ 4 devices.

- CY8CKIT-043 – PSoC™ 4 M-Series prototyping kit
- CY8CKIT-145-40XX – PSoC™ 4000S CAPSENSE™ prototyping kit
- CY8CKIT-147 – PSoC™ 6 Wi-Fi Bluetooth® prototyping kit
- CY8CKIT-149 – PSoC™ 4100S Plus prototyping kit

5.4.2 PSoC™ 3 and PSoC™ 5LP development kits

- CY8CKIT-059 – PSoC™ 5LP prototyping kit with onboard programmer and debugger

5.4.3 CAPSENSE™ express development kits

- CY3280-MBR3 Evaluation Kit for CY8CMBR3xxx family

5.4.4 PSoC™ 6 development kits

- CY8CPROTO-062-4343W – PSoC™ 6 Wi-Fi Bluetooth® prototyping kit
- CY8CKIT-062-BLE – PSoC™ 6 Bluetooth® LE pioneer kit
- CY8CKIT-062-WIFI-BT – PSoC™ 6 WiFi-Bluetooth® pioneer kit
- CY8CPROTO-062S3-4343W – PSoC™ 62S3 Wi-Fi Bluetooth® prototyping kit

5.4.5 Kits for programming and debugging

5.4.5.1 Miniprog3

The CY8CKIT-002 – PSoC™ MiniProg3 Program and Debug Kit is an all-in-one programmer for PSoC™ 1, PSoC™ 3, PSoC™ 4, and PSoC™ 5LP architectures, a debug tool for PSoC™ 3, PSoC™ 4, and PSoC 5LP architectures, and a USB-I2C Bridge for communicating with PSoC devices. Other than for programming, this device is mainly used as a USB-to-I2C Bridge in data monitoring and tuning of CAPSENSE™ solutions.

5.4.5.2 Miniprog4

The CY8CKIT-005 – MiniProg4 Program and Debug Kit is an all-in-one development programmer and debugger for PSoC™ 4, PSoC™ 5LP, and PSoC™ 6 MCU devices. MiniProg4 is used as a programmer or debug probe for supported Infineon devices. and development kits.

5.5 Design support

Infineon has several support channels to ensure the success of your CAPSENSE™ design:

- Infineon Developer Community – Connect with the Infineon technical community and exchange information.
- Video Library – Get up to speed with tutorial videos.
- Infineon Design Partner Program – An expansion of our engineering capabilities providing customers with access to design services and solutions from trusted and capable partners.
- Technical Support – Excellent technical support is available online.
- Quality and Reliability – Infineon is committed to complete customer satisfaction. At our Quality website, you can find reliability and product qualification reports.

6 Appendix A: Springs

6.1 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

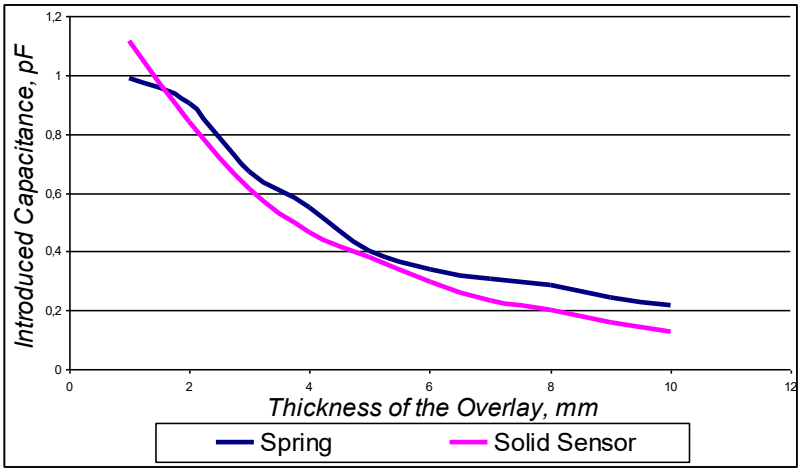

Figure 131 FTC versus overlay thickness

- Influence of height on FTC

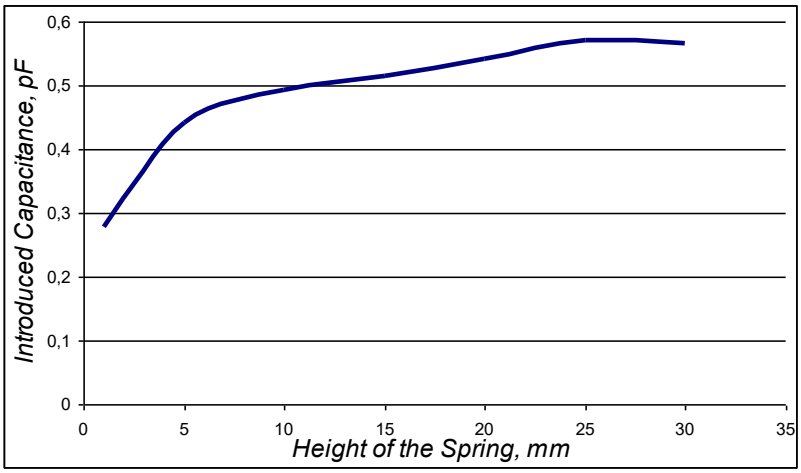

Figure 132 FTC versus spring height

- Influence of diameter on FTC

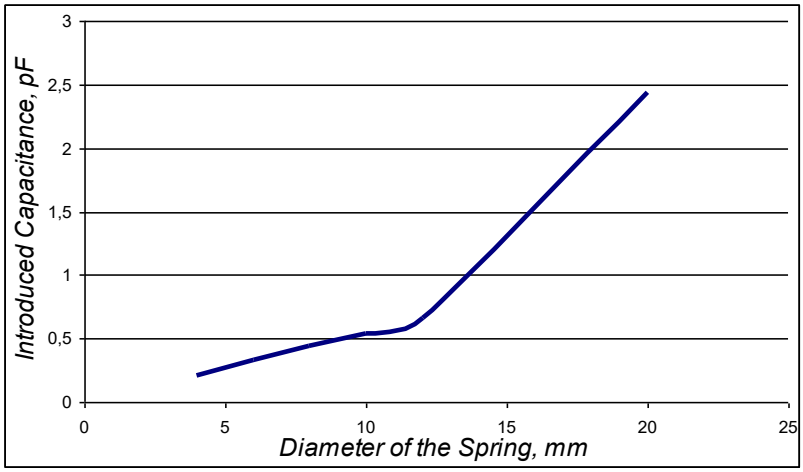

Figure 133 FTC versus spring diameter

- Influence of wire thickness of the spring on FTC

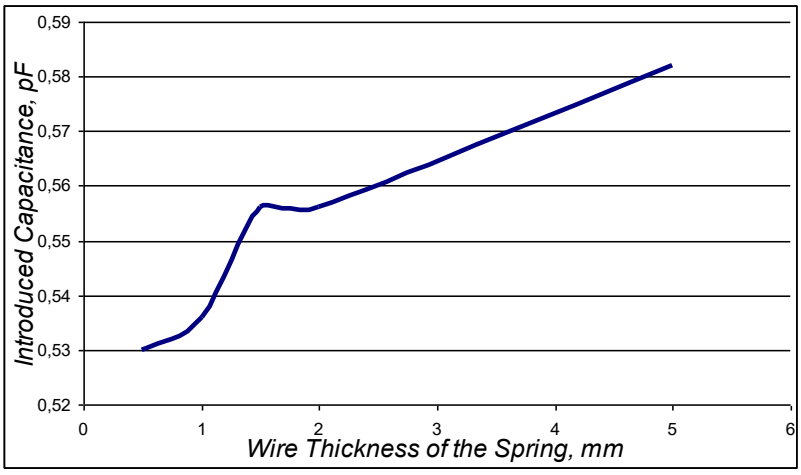

Figure 134 FTC versus wire thickness of spring

6.1.1 Mounting springs to the PCB

Figure 135 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.

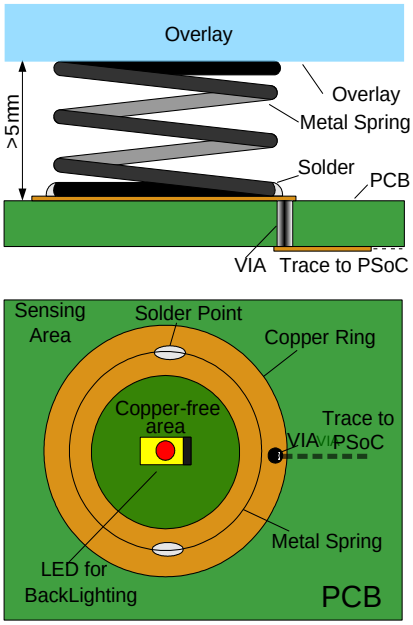

Figure 135 Spring-mounting example

Figure 136 shows examples of footprints for springs. Unlike a button sensor, which is a copper pad on the PCB surrounded by the ground hatch, it is not possible to surround a spring with ground as its surface is above the PCB. However, you can still provide the ground hatch near the footprint of the spring similar to buttons with an air gap of 1 mm between the sensor ring and the ground. See Ground plane for details on the ground hatch fill.

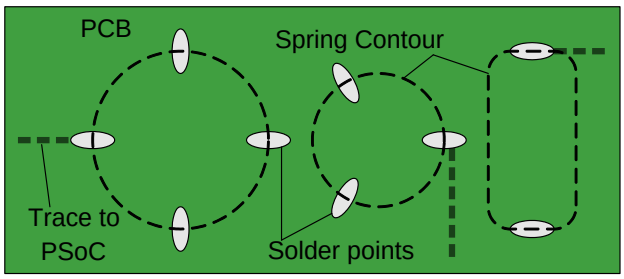

Figure 136 Proposed spring footprints

6.2 CAPSENSE™ and mechanical button combination

The hollow space inside a spring can also be used as a mechanical button, as shown in Figure 137.

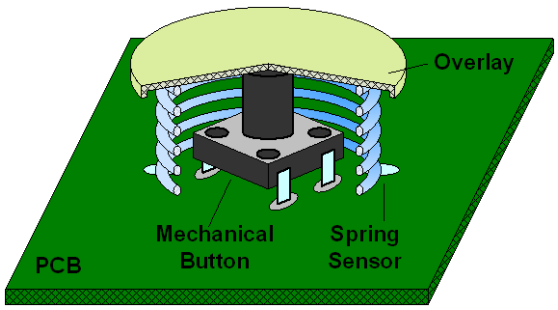

Figure 137 CAPSENSE™ and mechanical button combination

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.

6.3 Design examples

Figure 138 and Figure 139 show project demonstrator examples for white goods applications.

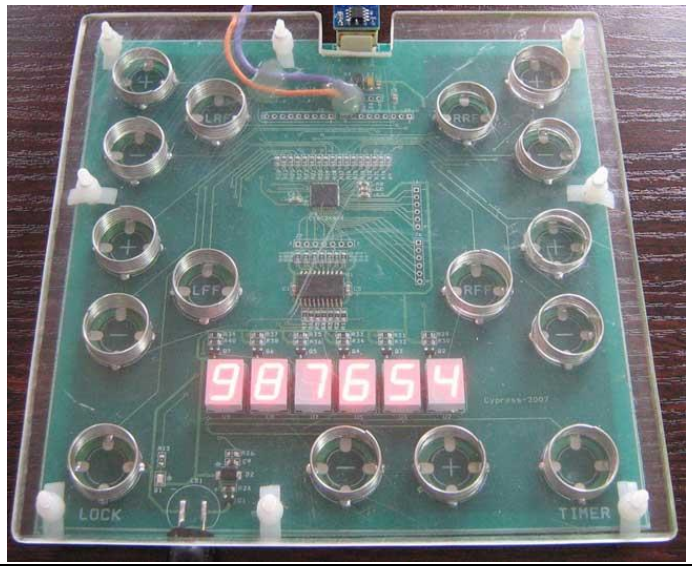

Figure 138 Demo cooktop

Figure 139 Cooktop front panel

7 Appendix B: Schematic and layout checklist

7.1 Schematic checklist

See the PSoC™ 3 datasheet for PSoC™ 3 devices, PSoC™ 4 datasheet for PSoC™ 4 devices, PSoC™ 5LP datasheet for PSoC™ 5LP devices, and PSoC™ 6 datasheet for PSoC™ 6 devices.

No.	Category	Recommendations/remarks
1	Decoupling capacitor	0.1 µF
2	Bulk capacitor	1 µF
3	Pin assignment	Sensors should be placed near to ground. No switching signal should be placed near to sensors
4	C_{MOD}	2.2 nF Ensure that CMOD is not adjacent to any switching or communication pins
5	C_{INTA} and C_{INTB}	470 pF Ensure that C_{INTA} and C_{INTB} are not adjacent to any switching or communication pins
6	Sensor pin selection	If possible, avoid pins that are close to the GPIOs carrying switching/communication signals. Physically separate DC loads such as LEDs and I2C pins from the CAPSENSE™ pins by a full port wherever possible. See the Tuning Debug FAQ, Raw counts show a level-shift or increased noise when GPIOs are toggled, for more details. **Note**: For PSoC™ 6 family devices, to achieve the best CAPSENSE™ sensitivity and accuracy, follow the recommendations stated in AN85951 – PSoC™ 4 and PSoC™ 6 MCU CAPSENSE™ design guide.
7	RB	Ensure that RB is not adjacent to any switching or communication pins
8	Series resistor on CAPSENSE™ lines	560 Ω
9	Series resistor on communication lines	330 Ω
10	Pull-up resistor on Communication lines	4.7 kΩ
11	Avoid using programming pins as I2C pins if possible	

7.1.1 Decoupling capacitor

See Power supply layout recommendations for complete details.

7.1.2 Bulk capacitor

See Power supply layout recommendations for complete details.

7.1.3 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.

See Pin assignments for more details.

7.1.4 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 the CAPSENSE™ (analog) signal rather than being surrounded by the 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	5.6 nF – 10 nF for PRS8 and PRS16 configuration. 22nF – 47nF for prescaler configuration
CY8C21x34 SmartSense,	10 nF
CY8C24x94, CY8C22x45	5.6 nF – 10 nF
CY8C20x34	1.2 nF – 5.6 nF
CY8CMBR3xxx, CY8CMBR2xxx, CY8C20xx7A/S, PSoC™ 3/4/5LP	2.2 nF

See the User Module and Component datasheets for more details. The User Module and Component datasheets get downloaded when you install PSoC™ Designer and PSoC™ Creator, respectively.

7.1.5 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

7.1.6 Series resistor on CAPSENSE™ lines

A 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, see Series resistor.

7.1.7 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 I²C operation as the VIL level still remains within the I²C specification limit of 0.3 VDD when PSoC™ outputs a LOW. For complete details, see Series resistor.

7.2 Layout checklist

No.	Category		Minimum	Maximum	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 no 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

Appendix B: Schematic and layout checklist

No.	Category		Minimum	Maximum	Recommendations
					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	Vias 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)
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	See Figure 125. 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, see Shield electrode.

No.	Category		Minimum	Maximum	Recommendations
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.

7.2.1 Buttons

The best shape for CSD 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. See Figure 99 for details.

The best geometry to implement the CSX button is the fishbone structure. See Mutual cap buttons of fishbone structure to get idea of some common fishbone patterns, or contact Infineon.

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.

7.2.2 Slider

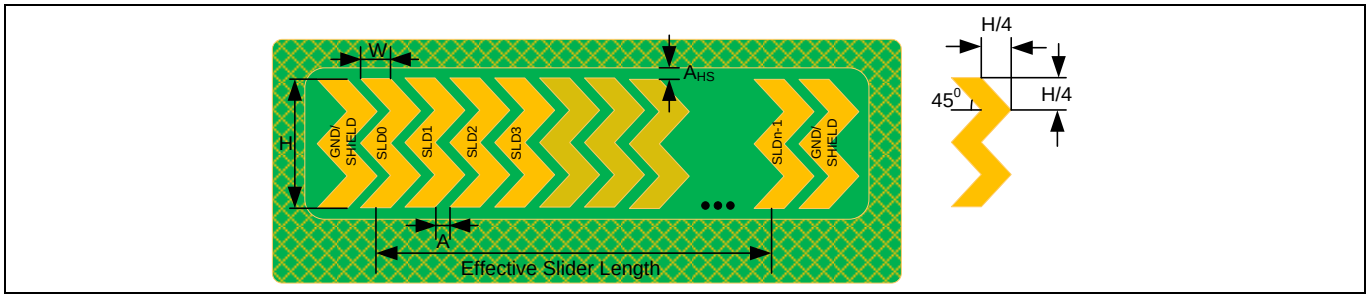

Figure 140 Typical liner slider pattern

- Keep the width of the segment (*W*) at 8 mm for an average finger width of 9 mm.
- Keep the air gap between segments (*A*) at 0.5 mm.
- Keep the height of the segment (*H*) at 12 mm.
- Keep the air gap between the hatch and the slider (A_{HS}) equal to the overlay thickness.
- 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.

See Slider design for more details.

7.2.3 Overlay

- 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.
- 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.
- For sliders, with SmartSense, the maximum acrylic overlay thickness is 4 mm and that of glass is 12 mm.

7.2.4 Sensor traces

- Keep the width less than or equal to 7 mil (0.18 mm)
- Keep the maximum trace length as 12 inches (300 mm) for a standard PCB and 2 inches (50 mm) for flex circuits.
- 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).
- Do not have sharp (90 degrees) turns as this will pick up noise; in addition, charges concentrate at sharp corners.
- Routing of traces (See 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 I2C or SPI masters. If it is necessary to cross communication lines with sensor trace, make sure that the intersection is at right angles.

7.2.5 Vias on sensors

- Placement should be at the edges of the CAPSENSE™ button to minimize increase in CP due to vias.
- Minimize the number of vias to reduce the parasitic capacitance, at up to two on a sensor (trace and pad).
- Keep the via hole diameter for sensor traces at 10 mil (See Figure 119 for details).

7.2.6 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.

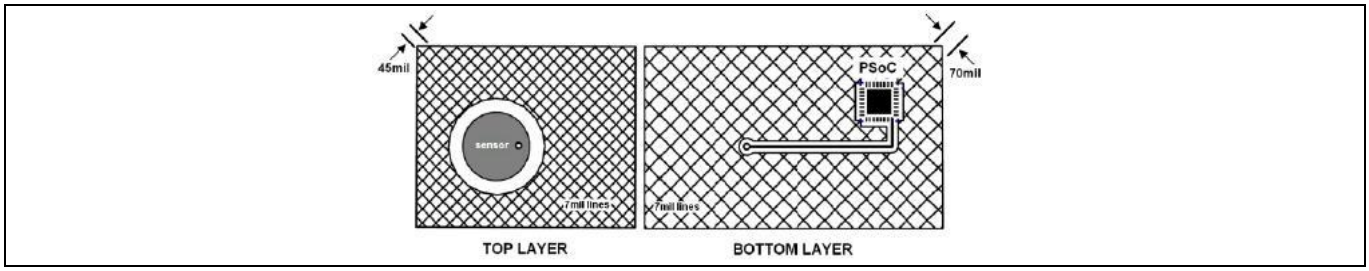

Figure 141 Typical ground fill

7.2.7 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.

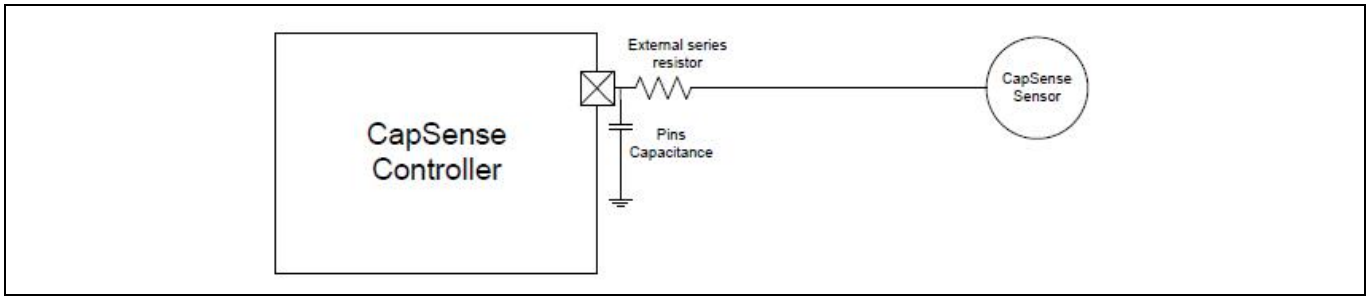

Figure 142 Series resistor placement

7.2.8 Shield electrode

1. Reduce the size of the shield fill (maximum 1 cm from the sensor).
2. Limit the placement of the shield to only the selected sensors.
3. 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, see Shield signal.

7.2.9 Guard sensor

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

8 Appendix C: Clearance between sensor and ground

The ground plane is placed on the same layer of the board as the buttons as shown in Figure 143. The clearance between the button and ground plane plays an important role in the performance of the button. Electric field lines fringing between a button and the ground plane are illustrated in Figure 144. The parasitic capacitance of the sensor, C_P, is related to this electric field.

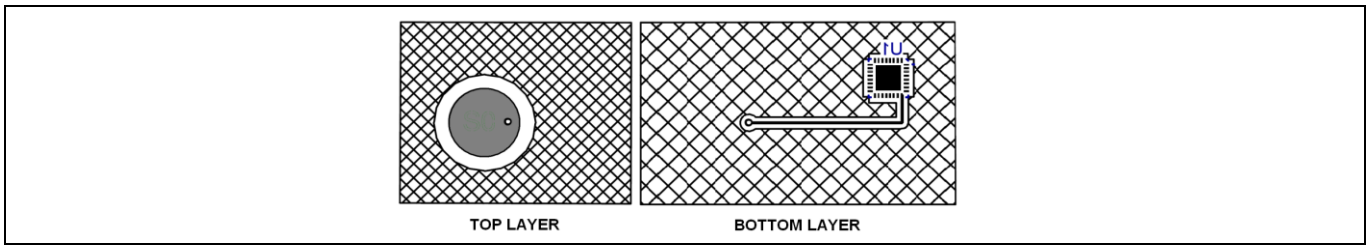

Figure 143 CAPSENSE™ board top and bottom layer

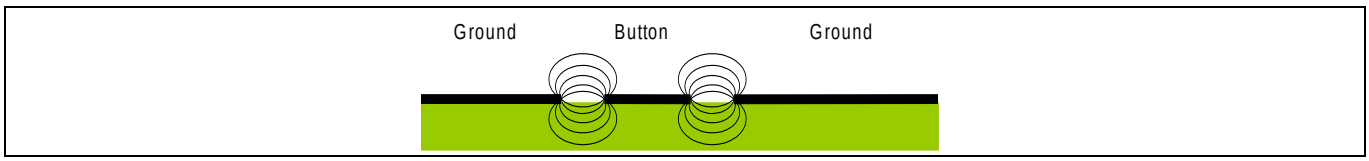

Figure 144 Button-ground plane fringing fields

The capacitance, C_P, decreases as the clearance surrounding the button increases. An example of this dependence of C_P on the gap is shown in Figure 145 through Figure 148. In these plots, the board material is FR4 with a thickness of 62 mils (1.57 mm), and the acrylic overlay has a thickness of 2 mm. Each plot contains data for three button sizes (5 mm, 10 mm, and 15 mm diameter).

The C_P in Figure 145 does not include the effect of the traces or vias. It is only the parasitic capacitance of the sensor pad.

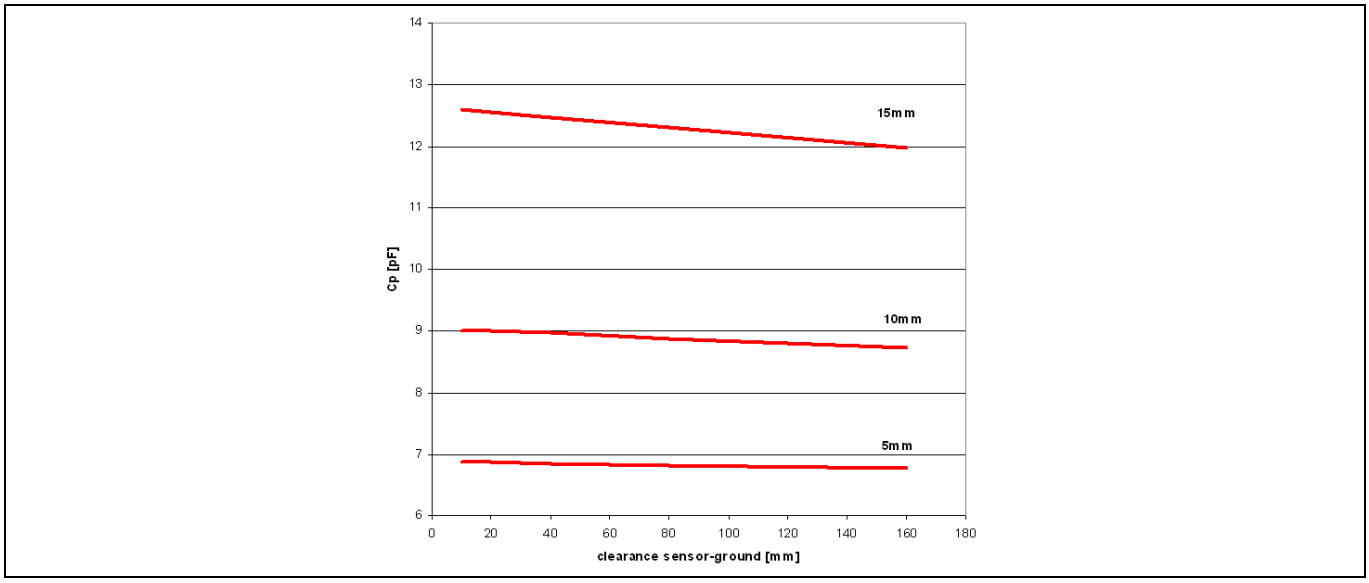

Figure 145 Parasitic capacitance, C_P, as a function of button-ground clearance and button diameter

Note: The finger is not on the sensor. Capacitance increases with sensor size, but decreases with the gap.

Appendix C: Clearance between sensor and ground

The capacitance, C_{sensor}, is the total sensor capacitance when the finger is not on the sensor. It includes the effect of the sensor pad, the traces, and vias. Figure 146 shows the sensor capacitance for a board routed with 50-mm trace length, 8-mil (0.3 mm) trace width, and 20-mil (0.8 mm) spacing from trace to the coplanar ground.

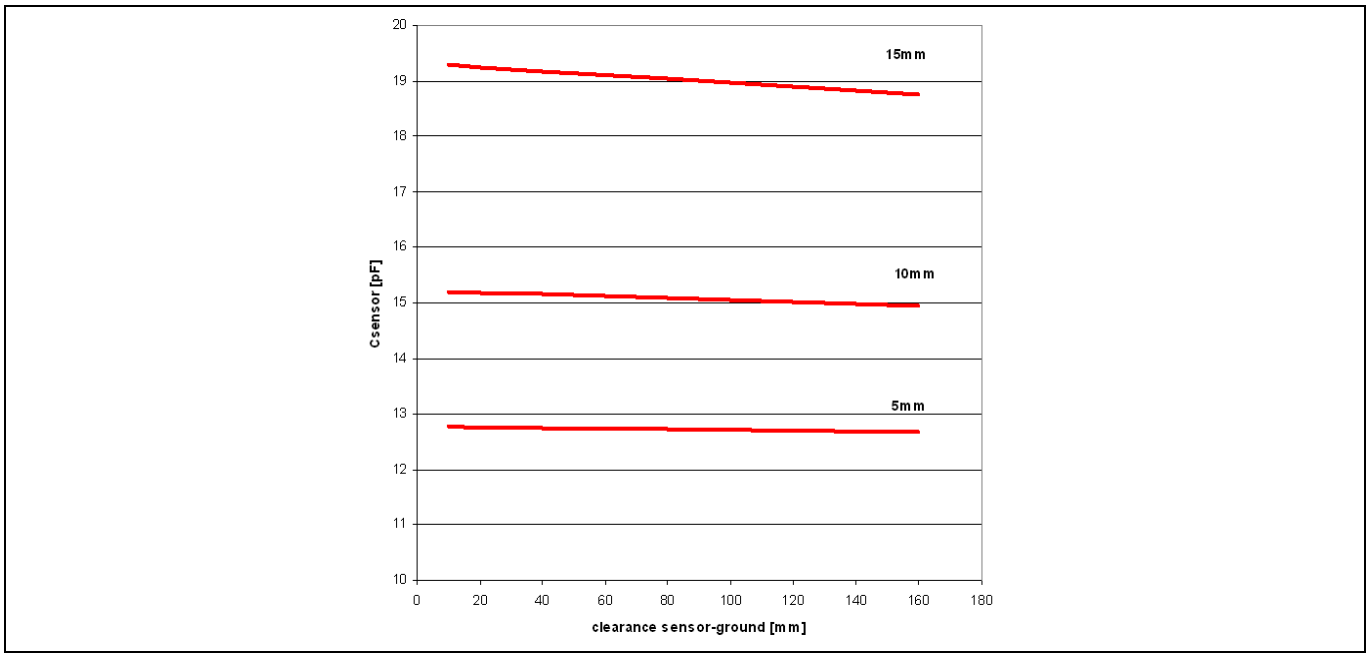

Figure 146 Sensor capacitance, C_{sensor}, as a function of button-ground clearance and button diameter

Note: The finger is not on the sensor. Sensor capacitance decreases with the size of the gap.

The capacitance, C_F, in Figure 147 is the capacitance added by the touch of the finger. Total capacitance of the sensor pad and the finger is $C_P + C_F$.

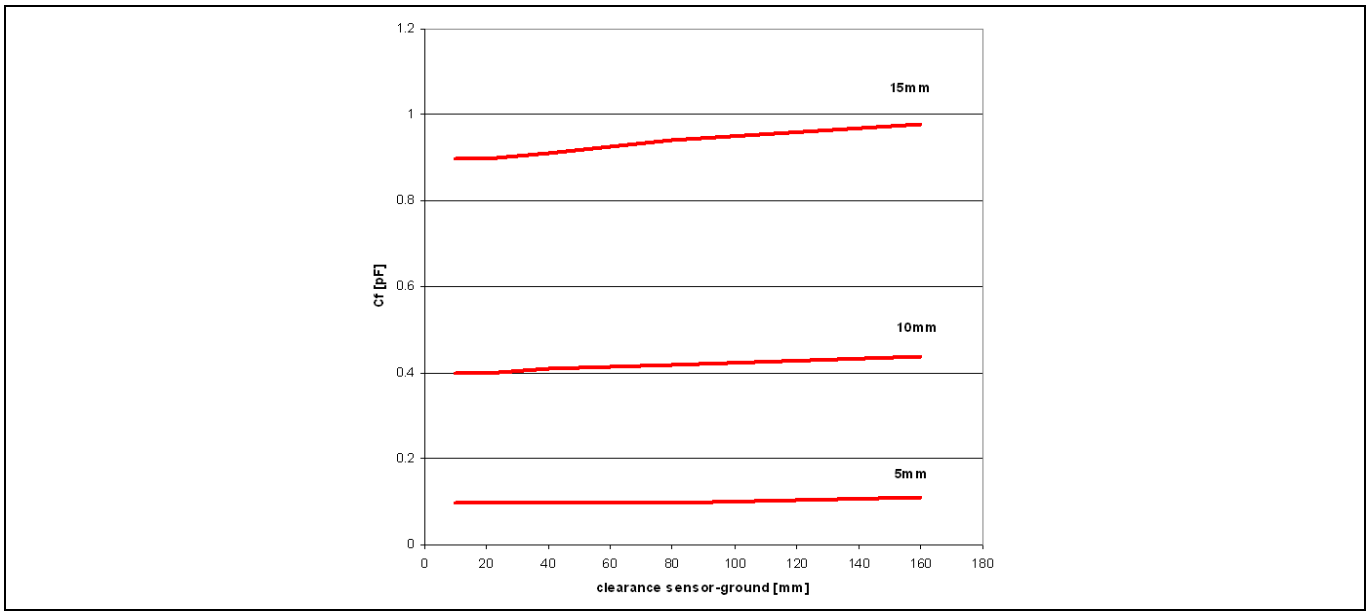

Figure 147 Finger capacitance, C_F, as a function of button-ground clearance and button diameter

Note: The finger is on the sensor. Capacitance increases with both sensor size and gap.

Appendix C: Clearance between sensor and ground

Figure 148 plots finger capacitance as a percentage of the sensor capacitance. This is the sensitivity of the sensor. The sensitivity of the system changes with the routing of the CAPSENSE™ traces. For example, increasing the trace length between the PSoC™ and the sensor pad decreases the button's sensitivity.

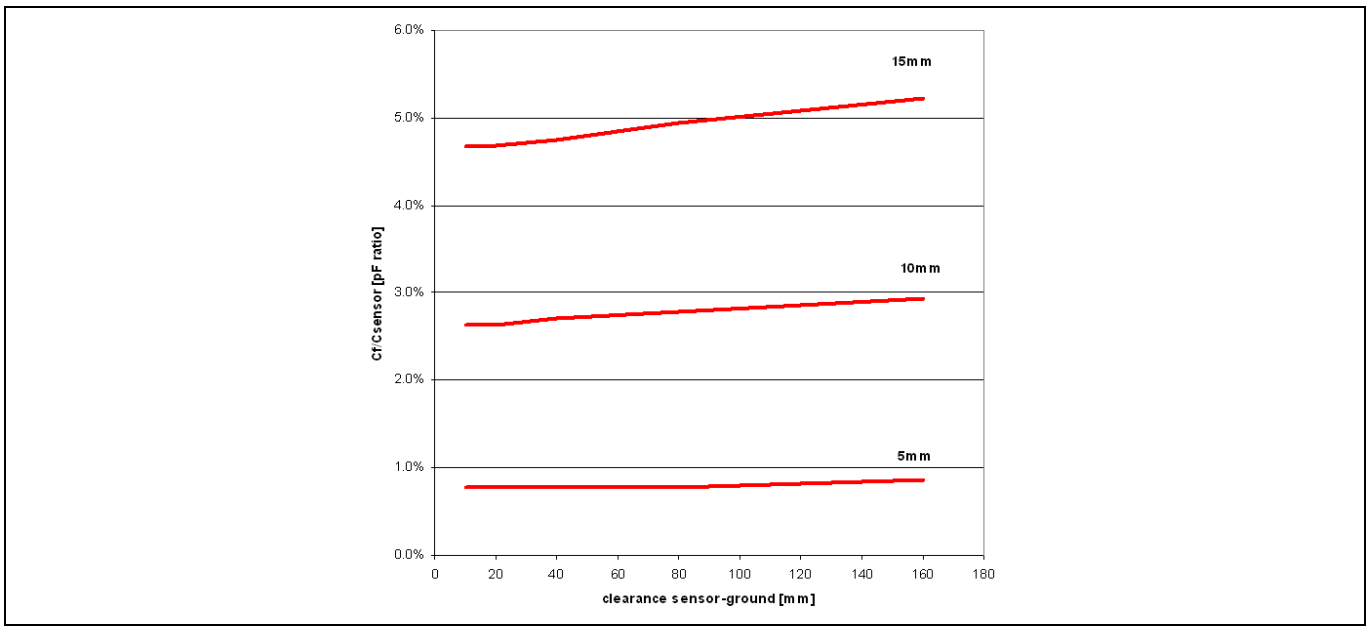

Figure 148 Sensitivity, C_F/C_{sensor} as a function of button-ground clearance and button diameter

Note: The sensitivity increases with both button size and gap.

Glossary

Term	Definition
AMUXBUS	Analog multiplexer bus available inside PSoC™ that helps to connect I/O pins with multiple internal analog signals.
SmartSense Auto-Tuning	A CAPSENSE™ algorithm that automatically sets sensing parameters for optimal performance after the design phase and continuously compensates for system, manufacturing, and environmental changes.
Baseline	A value resulting from a firmware algorithm that estimates a trend in the Raw Count when there is no human finger present on the sensor. The Baseline is less sensitive to sudden changes in the Raw Count and provides a reference point for computing the Difference Count.
Button or Button Widget	A widget with an associated sensor that can report the active or inactive state (that is, only two states) of the sensor. For example, it can detect the touch or no-touch state of a finger on the sensor.
Difference Count	The difference between Raw Count and Baseline. If the difference is negative, or if it is below Noise Threshold, the Difference Count is always set to zero.
Capacitive Sensor	A conductor and substrate, such as a copper button on a printed circuit board (PCB), which reacts to a touch or an approaching object with a change in capacitance.
CAPSENSE™	Infineon's touch-sensing user interface solution, which is the industry's leading solution in sales.
CAPSENSE™ Mechanical Button Replacement (MBR)	Infineon's configurable solution to upgrade mechanical buttons to capacitive buttons. It requires minimal engineering effort to configure the sensor parameters and does not require firmware development. These devices include the CY8CMBR3XXX and CY8CMBR2XXX families.
Centroid or Centroid Position	A number indicating the finger position on a slider within the range given by the Slider Resolution. This number is calculated by the CAPSENSE™ centroid calculation algorithm.
CINTA and CINTB Capacitors	Two external capacitors required for the operation of a CSX block in Mutual-Capacitance sensing mode.
Compensation IDAC	A programmable constant current source, which is used by CSD to compensate for excess sensor C_P. This IDAC is not controlled by the Sigma-Delta Modulator in the CSD block unlike the Modulation IDAC.
CSD	CAPSENSE™ Sigma Delta (CSD) is a Infineon-patented method of performing self-capacitance (also called self-cap) measurements for capacitive sensing applications. In the CSD mode, the sensing system measures the self-capacitance of an electrode, and a change in the self-capacitance is detected to identify the presence or absence of a finger.
CSX	CAPSENSE™ Crosspoint (CSX) is a Infineon-patented method of performing mutual-capacitance (also called mutual-cap) measurements for capacitive sensing applications. In the CSX mode, the sensing system measures the capacitance between two electrodes, and a change in the capacitance is detected to identify the presence or absence of a finger.

Appendix C: Clearance between sensor and ground

Term	Definition
Debounce	A parameter that defines the number of consecutive scan samples for which the touch should be present for it to become valid. This parameter helps to reject spurious touch signals. A finger touch is reported only if the Difference Count is greater than Finger Threshold + Hysteresis for a consecutive Debounce number of scan samples.
Driven-Shield	A technique used by CSD for enabling liquid tolerance in which the Shield Electrode is driven by a signal that is equal to the sensor switching signal in phase and amplitude.
Electrode	A conductive material such as a pad or a layer on PCB, ITO, or FPCB. The electrode is connected to a port pin on a CAPSENSE™ device and is used as a CAPSENSE™ sensor or to drive specific signals associated with CAPSENSE™ functionality.
Finger Threshold	A parameter used with Hysteresis to determine the state of the sensor. Sensor state is reported ON if the Difference Count is higher than Finger Threshold + Hysteresis, and it is reported OFF if the Difference Count is below Finger Threshold – Hysteresis.
Ganged Sensors	The method of connecting multiple sensors together and scanning them as a single sensor. Used for increasing the sensor area for proximity sensing and to reduce power consumption. To reduce power when the system is in Low-Power mode, all the sensors can be ganged together and scanned as a single sensor taking less time instead of scanning all the sensors individually. When the user touches any of the sensors, the system can transition into Active mode where it scans all the sensors individually to detect which sensor is activated. PSoC™ supports sensor-ganging in firmware, that is, multiple sensors can be connected simultaneously to AMUXBUS for scanning.
Gesture	Gesture is an action, such as swiping and pinch-zoom, performed by the user. CAPSENSE™ has a gesture detection feature that identifies the different gestures based on predefined touch patterns. In the CAPSENSE™ component, the Gesture feature is supported only by the Linear Slider, Radial Slider and Touchpad Widgets.
Guard Sensor	Copper trace that surrounds all the sensors on the PCB, similar to a button sensor and is used to detect a liquid stream. When the Guard Sensor is triggered, firmware can disable scanning of all other sensors to prevent false touches.
Hatch Fill or Hatch Ground or Hatched Ground	While designing a PCB for capacitive sensing, a grounded copper plane should be placed surrounding the sensors for good noise immunity. But a solid ground increases the parasitic capacitance of the sensor which is not desired. Therefore, the ground should be filled in a special hatch pattern. A hatch pattern has closely-placed, crisscross lines looking like a mesh and the line width and the spacing between two lines determine the fill percentage. In case of liquid tolerance, this hatch fill (referred as a shield electrode) is driven with a shield signal instead of ground.
Hysteresis	A parameter used to prevent the sensor status output from random toggling due to system noise, used in conjunction with the Finger Threshold to determine the sensor state. See **Finger Threshold**.

Appendix C: Clearance between sensor and ground

Term	Definition
IDAC (Current-Output Digital-to-Analog Converter)	Programmable constant current source available inside PSoC™, used for CAPSENSE™ and ADC operations.
Liquid Tolerance	The ability of a capacitive sensing system to work reliably in the presence of liquid droplets, streaming liquids, or mist.
Linear Slider	A widget consisting of more than one sensor arranged in a specific linear fashion to detect the physical position (in single axis) of a finger.
Low Baseline Reset	A parameter that represents the maximum number of scan samples where the Raw Count is abnormally below the Negative Noise Threshold. If the Low Baseline Reset value is exceeded, the Baseline is reset to the current Raw Count.
Manual-Tuning	The manual process of setting (or tuning) the CAPSENSE™ parameters.
Matrix Buttons	A widget consisting of more than two sensors arranged in a matrix fashion, used to detect the presence or absence of a human finger (a touch) on the intersections of vertically and horizontally arranged sensors. If *M* is the number of sensors on the horizontal axis and *N* is the number of sensors on the vertical axis, the Matrix Buttons Widget can monitor a total of $M \times N$ intersections using ONLY $M + N$ port pins. When using the CSD sensing method (self-capacitance), this widget can detect a valid touch on only one intersection position at a time.
Modulation Capacitor (CMOD)	An external capacitor required for the operation of a CSD block in Self-Capacitance sensing mode.
Modulator Clock	A clock source that is used to sample the modulator output from a CSD block during a sensor scan. This clock is also fed to the Raw Count counter. The scan time (excluding pre and post processing times) is given by $(2^N - 1)$/Modulator Clock Frequency, where N is the Scan Resolution.
Modulation IDAC	Modulation IDAC is a programmable constant current source, whose output is controlled (ON/OFF) by the sigma-delta modulator output in a CSD block to maintain the AMUXBUS voltage at V_{REF}. The average current supplied by this IDAC is equal to the average current drawn out by the sensor capacitor.
Mutual-Capacitance	Capacitance associated with an electrode (say TX) with respect to another electrode (say RX) is known as mutual capacitance.
Negative Noise Threshold	A threshold used to differentiate usual noise from the spurious signals appearing in negative direction. This parameter is used in conjunction with the Low Baseline Reset parameter. Baseline is updated to track the change in the Raw Count as long as the Raw Count stays within Negative Noise Threshold, that is, the difference between Baseline and Raw count (Baseline – Raw count) is less than Negative Noise Threshold. Scenarios that may trigger such spurious signals in a negative direction include: a finger on the sensor on power-up, removal of a metal object placed near the sensor, removing a liquid-tolerant CAPSENSE™-enabled product from the water; and other sudden environmental changes.
Noise (CAPSENSE™ Noise)	The variation in the Raw Count when a sensor is in the OFF state (no touch), measured as peak-to-peak counts.

Appendix C: Clearance between sensor and ground

Term	Definition
Noise Threshold	A parameter used to differentiate signal from noise for a sensor. If Raw Count – Baseline is greater than Noise Threshold, it indicates a likely valid signal. If the difference is less than Noise Threshold, Raw Count contains nothing but noise.
Overlay	A non-conductive material, such as plastic and glass, which covers the capacitive sensors and acts as a touch-surface. The PCB with the sensors is directly placed under the overlay or is connected through springs. The casing for a product often becomes the overlay.
Parasitic Capacitance (C_P)	Parasitic capacitance is the intrinsic capacitance of the sensor electrode contributed by PCB trace, sensor pad, vias, and air gap. It is unwanted because it reduces the sensitivity of CSD.
Proximity Sensor	A sensor that can detect the presence of nearby objects without any physical contact.
Radial Slider	A widget consisting of more than one sensor arranged in a specific circular fashion to detect the physical position of a finger.
Raw Count	The unprocessed digital count output of the CAPSENSE™ hardware block that represents the physical capacitance of the sensor.
Refresh Interval	The time between two consecutive scans of a sensor.
Scan Resolution	Resolution (in bits) of the Raw Count produced by the CSD block.
Scan Time	Time taken for completing the scan of a sensor.
Self-Capacitance	The capacitance associated with an electrode with respect to circuit ground.
Sensitivity	The change in Raw Count corresponding to the change in sensor capacitance, expressed in counts/pF. Sensitivity of a sensor is dependent on the board layout, overlay properties, sensing method, and tuning parameters.
Sense Clock	A clock source used to implement a switched-capacitor front-end for the CSD sensing method.
Sensor	See **Capacitive Sensor**.
Sensor Auto Reset	A setting to prevent a sensor from reporting false touch status indefinitely due to system failure, or when a metal object is continuously present near the sensor. When Sensor Auto Reset is enabled, the Baseline is always updated even if the Difference Count is greater than the Noise Threshold. This prevents the sensor from reporting the ON status for an indefinite period of time. When Sensor Auto Reset is disabled, the Baseline is updated only when the Difference Count is less than the Noise Threshold.
Sensor Ganging	See **Ganged Sensors**.
Shield Electrode	Copper fill around sensors to prevent false touches due to the presence of water or other liquids. Shield Electrode is driven by the shield signal output from the CSD block. See **Driven-Shield**.
Shield Tank Capacitor (CSH)	An optional external capacitor (C_{SH} Tank Capacitor) used to enhance the drive capability of the CSD shield, when there is a large shield layer with high parasitic capacitance.
Signal (CAPSENSE™ Signal)	Difference Count is also called Signal. See Difference Count.

Appendix C: Clearance between sensor and ground

Term	Definition
Signal-to-Noise Ratio (SNR)	The ratio of the sensor signal, when touched, to the noise signal of an untouched sensor.
Slider Resolution	A parameter indicating the total number of finger positions to be resolved on a slider.
Touchpad	A Widget consisting of multiple sensors arranged in a specific horizontal and vertical fashion to detect the X and Y position of a touch.
Trackpad	See **Touchpad**.
Tuning	The process of finding the optimum values for various hardware and software or threshold parameters required for CAPSENSE™ operation.
V_{REF}	Programmable reference voltage block available inside PSoC™ used for CAPSENSE™ and ADC operation.
Widget	A user-interface element in the CAPSENSE™ component that consists of one sensor or a group of similar sensors. Button, proximity sensor, linear slider, radial slider, matrix buttons, and touchpad are the supported widgets.

References

Code examples

[1] Code examples for ModusToolbox™ software

[2] PSoC™ Creator code examples

Video training library

[3] Video training library

Revision history

Document revision	Date	Description of changes
**	2010-12-17	New guide
*A	2011-03-04	Multiple chapter enhancements for content and reader clarity
*B	2011-08-16	Multiple section and table updates
*C	2011-12-07	Multiple chapter enhancements for content clarity
*D	2012-04-27	Updated slider section. Updated PCB layout guidelines. Updated Table 21. Corrected phone number in the title page
*E	2012-07-19	• Updated sample schematics and layouts Included information on layout/trace routing guidelines for C_{MOD}, R_B pin
*F	2012-08-31	• Updated references to external documents
*G	2012-10-16	• Updated Section 3.1 (Overlay selection) and moved Table 3 to Section 3.2
*H	2013-01-07	• Updated Section 3.1. Added Section 3.7.13
*I	2013-03-07	• Updated Section 2.7.2. Haptic feedback
*J	2013-06-07	• Added the CY8C20XX7/S part family. Added proximity information
*K	2013-09-04	• Added the CY8C22X45 part family information
*L	2013-10-04	• Corrected document revision on the footer of some pages
*M	2014-02-19	• Added information specific to CY8CMBR3XXX
*N	2014-03-05	• Updated the operating voltage specifications in Table 20
*O	2014-09-17	• Updated Figure 1 to have a mention of PSoC™ creator • Updated CSD block diagram to remove unnecessary bends • Updated Equation 6 to reflect the correct noise definition • Updated Liquid tolerance and Proximity (three-dimensional sensors) sections • Updated references to PSoC™ 3/4/5LP in multiple places • Removed reference of CY3235 – Proximity Detection Demonstration Kit • In Chapter 4, added PSoC™ 3/4/5LP references in multiple places
*P	2015-01-22	• Added guidelines on LEDs close to the sensor • Added a note on slider guidelines • Updated template • Changed document title
*Q	2015-06-18	• Fixed broken reference
*R	2016-01-19	• Updated Chapter 1 Introduction – added the benefits of CAPSENSE™ over mechanical buttons. Added CAPSENSE™ design flow • Updated Chapter 2 CAPSENSE™ technology • Updated the sections Capacitive touch sensing method, CAPSENSE™ tuning, CAPSENSE widgets, and User interface feedback

Revision history

Document revision	Date	Description of changes
		• Added Ground plane and Proximity sensing, re-aligned the sections CAPSENSE™ system overview and Liquid tolerance • Added chapter Capsense™ selector guide and CAPSENSE™ resources • Added a Glossary of CAPSENSE™ terms
*S	2016-02-11	• Updated Table 21 – Added reference to AN202478, AN72362, AN88890, AN86272, and AN49079 • Added Mutual cap content in Mutual capacitance section • Updated CAPSENSE™ sigma delta modulator (CSD) sensing method section with content on fourth generation CAPSENSE™ • Updated Figure 127 to add Cortex®-M0+ for PSoC™ 4000S and PSoC™ 4100S family of devices • Updated Table 17 with PSoC™ 4200 L, PSoC™ 4000S and PSoC™ 4100S family of devices details • Updated PSoC™ 4 development kits with PSoC™ 4 Pioneer Kits and Shield Kits
*T	2016-02-24	• Updated Table 21 – Added reference to AN210998, AN96475.
*U	2016-10-26	• Fixed a typo • Updated template • Fixed formatting throughout
*V	2017-04-19	• Updated logo and copyright
*W	2017-09-27	• Added references to PSoC™ 4100S Plus throughout the document • Updated Table 17 with PSoC™ 4100S Plus features • Updated section PSoC™ 4 development kits with CY8CKIT-149 PSoC™ 4100S Plus Prototyping Kit • Updated section CAPSENSE™ design guides and application notes with links to Getting Started Application Notes
*X	2018-02-28	• Added support for PSoC™ 4100PS throughout the document • Updated Introduction chapter with the code example and video link • Updated Table 17 with additional CAPSENSE™ features
*Y	2020-02-13	• Added information related to PSoC™ 6 devices • Added information on ModusToolbox™ • Added Section CAPSENSE™ sigma delta modulator (CSD) sensing method • Added Section CAPSENSE™ crosspoint (CSX) sensing method • Added Section Self-cap button structure • Added Section Mutual cap buttons of fishbone structure • Added a note on CAPSENSE™ pin selection with GPIO toggling in section Schematic checklist. Added CSX information in Glossary • Added Section Miniprog4

Revision history

Document revision	Date	Description of changes
		• Updated Design CAPSENSE™ Hardware and Develop Firmware sections of Table 21 to include references to different development kits • Added Section Example schematic and layout
*Z	2022-08-17	• Updated to Infineon template • Rebranding to PSoC™ multitouch guidelines
AA	2024-04-12	• Fixed the links • Removed the section 5.4.4 PSoC™ 1 development kits and section 9 Appendix D: PSoC™ 1 in-circuit emulation (ICE) pods

Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

The Bluetooth® word mark and logos are registered trademarks owned by Bluetooth SIG, Inc., and any use of such marks by Infineon is under license.

Edition 2024-04-12

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email: erratum@infineon.com

Document reference

001-64846 Rev. AA

Important notice

The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

Warnings

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.