AN75400 - PSoC® 3 和 PSoC® 5LP

CapSense®设计指南

文档编号：001-80488 版本*C

赛普拉斯半导体
198 Champion Court
San Jose, CA 95134-1709
http://www.cypress.com

版权所有

赛普拉斯半导体公司，2012-2019 年。本文件是赛普拉斯半导体公司及其子公司，包括 Spansion LLC（“赛普拉斯”）的财产。本文件，包括其包含或引用的任何软件或固件（“软件”），根据全球范围内的知识产权法律以及美国与其他国家签署条约由赛普拉斯所有。除非在本款中另有明确规定，赛普拉斯保留在该等法律和条约下的所有权利，且未就其专利、版权、商标或其他知识产权授予任何许可。如果软件并不附随有一份许可协议且贵方未以其他方式与赛普拉斯签署关于使用软件的书面协议，赛普拉斯特此授予贵方属人性质的、非独家且不可转让的如下许可（无再许可权）（1）在赛普拉斯特软件著作权项下的下列许可权（一）对以源代码形式提供的软件，仅出于在赛普拉斯硬件产品上使用之目的且仅在贵方集团内部修改和复制软件，和（二）仅限于在有关赛普拉斯硬件产品上使用之目的将软件以二进制代码形式的向外部最终用户提供（无论直接提供或通过经销商和分销商间接提供），和（2）在被软件（由赛普拉斯公司提供，且未经修改）侵犯的赛普拉斯专利的权利主张项下，仅出于在赛普拉斯硬件产品上使用之目的制造、使用、提供和进口软件的许可。禁止对软件的任何其他使用、复制、修改、翻译或汇编。

在适用法律允许的限度内，赛普拉斯未对本文件或任何软件作出任何明示或暗示的担保，包括但不限于关于适销性和特定用途的默示保证。没有任何电子设备是绝对安全的。因此，尽管赛普拉斯在其硬件和软件产品中采取了必要的安全措施，但是赛普拉斯并不承担任何由于使用赛普拉斯产品而引起的安全问题及安全漏洞的责任，例如未经授权的访问或使用赛普拉斯产品。此外，本材料中所介绍的赛普拉斯产品有可能存在设计缺陷或设计错误，从而导致产品的性能与公布的规 格不一致。（如果发现此类问题，赛普拉斯会提供勘误表）赛普拉斯保留更改本文件的权利，届时将不另行通知。在适用法律允许的限度内，赛普拉斯不对因应用或使用本文件所述任何产品或电路引起的任何后果负责。本文件，包括任何样本设计信息或程序代码信息，仅为供参考之目的提供。文件使用人应负责正确设计、计划和测试信息应用和由此生产的任何产品的功能和安全性。赛普拉斯产品不应被设计为、设定为或授权用作武器操作、武器系统、核设施、生命支持设备或系统、其他医疗设备或系统（包括急救设备和手术植入物）、污染控制或有害物质管理系统中的关键部件，或产品植入之设备或系统故障可能导致人身伤害、死亡或财产损失其他用途（“非预期用途”）。关键部件指，若该部件发生故障，经合理预期会导致设备或系统故障或会影响设备或系统安全性和有效性的部件。针对由赛普拉斯产品非预期用途产生或相关的任何主张、费用、损失和其他责任，赛普拉斯不承担全部或部分责任且贵方不应追究赛普拉斯之责任。贵方应赔偿赛普拉斯因赛普拉斯产品任何非预期用途产生或相关的所有索赔、费用、损失和其他责任，包括因人身伤害或死亡引起的主张，并使之免受损失。

赛普拉斯、赛普拉斯徽标、Spansion、Spansion 徽标，及上述项目的组合，WICED，及 PSoC、CapSense、EZ-USB、F-RAM 和 Traveo 应视为赛普拉斯在美国和其他国家的商标或注册商标。请访问 cypress.com 获取赛普拉斯商标的完整列表。其他名称和品牌可能由其各自所有者主张为该方财产。

目录

1. 简介 5

1.1 摘要 5
1.2 赛普拉斯 CapSense 文档体系 5
1.3 PSoC 3 和 PSoC 5LP 器件特性 8
1.4 文档规范 9

2. CapSense 技术 10

2.1 CapSense 基本原理 10
2.2 电容转换 11
2.3 SmartSense 自动调校 13

3. PSoC 3 和 PSoC 5LP 中的 CapSense 14

3.1 CSD 实现 14
3.2 独特的 CapSense 功能 15
3.3 CapSense PLUS 22

4. CapSense 设计工具 23

4.1 PSoC Creator 23
4.2 硬件套件 24

5. CapSense_CSD 组件 25

5.1 参数汇总 26
5.2 全局数组 27
5.3 高层参数 28
5.4 低层参数 34
5.5 Widget 配置 51
5.6 调校方法 54
5.7 通道数量 55
5.8 调谐器 GUI 58

6. CapSense 功能调校 63

6.1 基本原理 63
6.2 手动调校 64
6.3 SmartSense 自动调校 66

7. 设计的注意事项 68

7.1 软件滤波 68
7.2 功耗 69
7.3 响应时间 73
7.4 引脚分配 76
7.5 PCB 布局指南 76

8. 资源 77

8.1 网站 77
8.2 数据手册 77
8.3 技术参考手册 77
8.4 开发套件 77
8.5 PSoC Programmer 78
8.6 Multi-Chart（多图工具） 78
8.7 PSoC Creator 78
8.8 代码示例 78
8.9 设计支持 79

术语表 80

修订记录 85

1. 简介

1.1 摘要

本文档为在 PSoC® 3 和 PSoC® 5LP 系列器件中设计 CapSense®应用程序提供设计指导。它面向熟悉电容式感应技术并且为他们的应用选择了 PSoC 3 和 PSoC 5LP 系列器件的设计工程师。为了全面了解 CapSense 技术，请参见 CapSense 入门。

1.2 赛普拉斯 CapSense 文档体系

图 1-1 和表 1-1 总结了赛普拉斯 CapSense 文档体系。利用这些资源，可快速获得所需的信息来完成 CapSense 产品设计。图 1-1 显示的是电容式感应系统产品设计周期的典型流程。本指南介绍了绿色显示的各个主题。表 1-1 提供的文档链接与图 1-1 中列出的每个已编号的任务相对应。

图 1-1. 典型的 CapSense 产品设计流程

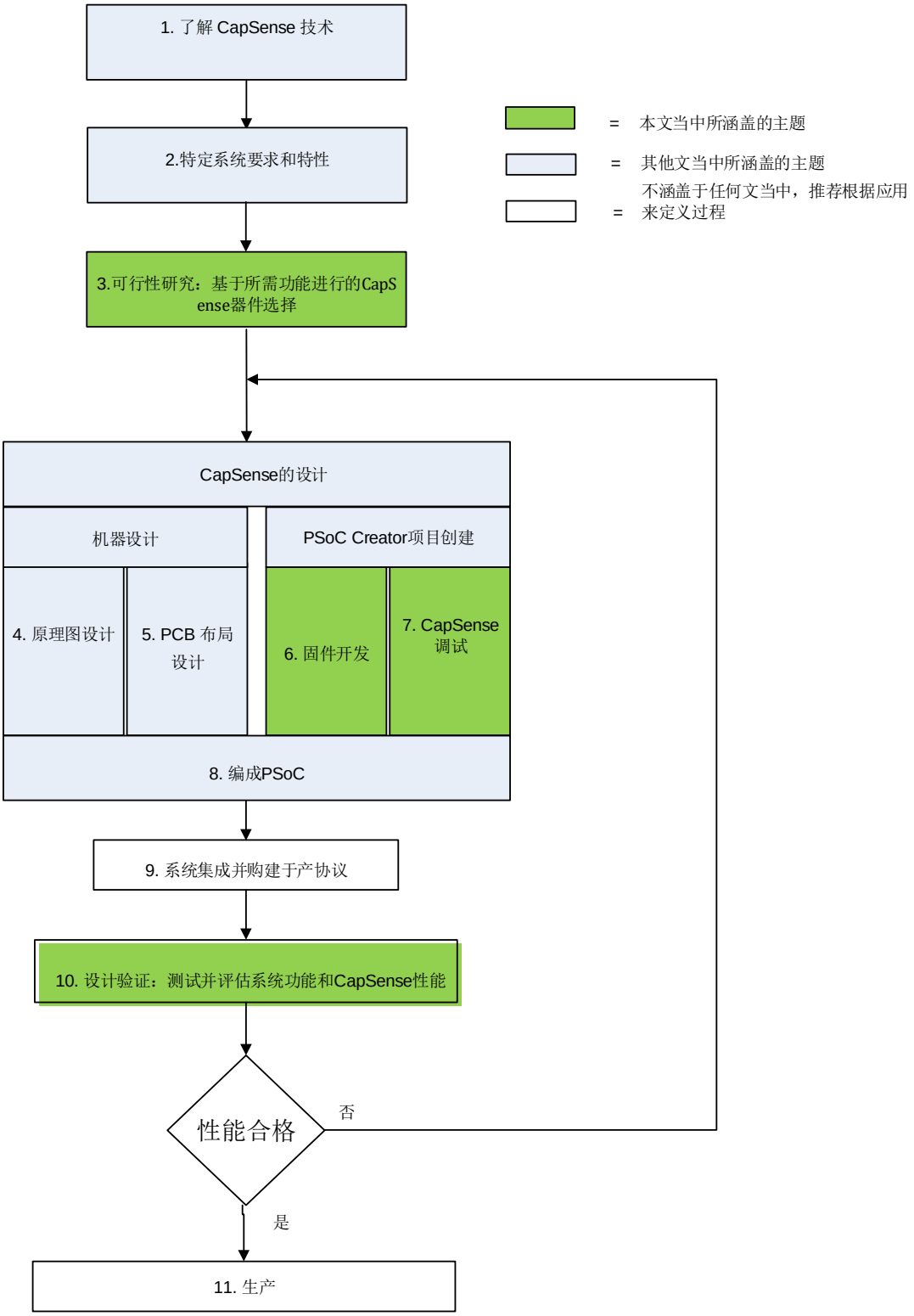


表 1-1.图 1-1 中每个已编号的任务相对应的赛普拉斯文档

已编号的设计任务	赛普拉斯的 CapSense 文档		
	名称	节	说明
1	Capsense 入门	2	CapSense 工作原理的深入讨论
2	Capsense 入门	4 和 5	介绍器件特性，并有助于用户选择器件
	PSoC 3 器件数据手册 PSoC 5LP 器件数据手册	-	
3	Capsense 入门	4 和 5	介绍器件特性，并有助于用户选择器件
	本文档	3	
4	Capsense 入门	3	提供引脚分配和 C_{MOD} 值等信息
	本文档	7	
5	Capsense 入门	3	提供 PCB 布局指南
6	PSoC Creator 帮助主题（位于 Help（帮助）选项卡下的 PSoC Creator IDE 中）	-	PSoC Creator 帮助主题提供使用 PSoC Creator IDE 的指南
	本文档	4、5、6 和 7	本文档和 CapSense_CSD 组件数据手册为开发 CapSense 应用提供有关固件方面的指导
	CapSense_CSD 组件数据手册	应用编程接口（API）	
7	本文档	6	介绍了如何调校 CapSense 系统
8	MiniProg3 用户指南	-	提供 PSoC 3 和 PSoC 5LP 编程的详细信息
	AN61290 — PSoC 3/PSoC 5LP 硬件设计注意事项	编程与调试	
9	N/A	-	-
10	Capsense 入门	3	介绍了重要的设计注意事项
	本文档	6 和 7	
11	N/A	-	-

1.3 PSoC 3 和 PSoC 5LP 器件特性

PSoC 3 和 PSoC 5LP 是真正的可编程嵌入式片上系统，集成了可配置的模拟和数字外设功能、存储器和微控制器。这些器件非常灵活，除 CapSense 功能外，它们还能实现其他多种功能。它们的主要特性包括：

器件特性

- PSoC 3 拥有 67 MHz 的 8051 CPU
- PSoC 5LP 拥有 67 MHz 的 ARM Cortex-M3 CPU
- PSoC 3 拥有高达 64 KB 的闪存、8 KB 的 SRAM 以及 2 KB 的 EEPROM
- PSoC 5LP 拥有高达 256 KB 的闪存、64 KB 的 SRAM 以及 2 KB 的 EEPROM
- 24 通道 DMA
- 多达 72 个 I/O 引脚
- 低功耗模式
  - PSoC 3 睡眠模式电流为 1 µA
  - PSoC 5LP 睡眠模式电流为 2 µA
  - PSoC 3 休眠模式电流为 200 nA
  - PSoC 5LP 休眠模式电流为 300 nA
- 模拟特性
  - 支持 8 至 20 位分辨率的可配置 Delta-Sigma 模数转换器（ADC）
  - 多达四个电压比较器、运算放大器、DAC 和多功能模拟模块
  - 1.024 V ±0.1%内部电压参考
- 数字特性
  - 多达四路 16 位可配置定时器、计数器和 PWM 模块
  - 多达 24 个基于可编程逻辑器件（PLD）的通用数字模块（UDB）
- 多种封装：QFN、SSOP 和 TQFP
- 所有 GPIO 均支持段式 LCD 驱动，最多支持 46 COM x 16 SEG
- I^2C、UART、全速 USB、SPI 和 CAN 通信接口

CapSense 功能

- 支持 CapSense 按键、滑条、矩阵按键和接近传感器的组合
- 支持多达 61 个电容式传感器，全部 GPIO 引脚均支持 CapSense 功能
- 通过集成式 API 开发固件
- 支持防水设计
- 双通道设计：资源充足，可同时扫描两个传感器
- SmartSense™自动调校
  - 在加电或运行期间自动设置和监控调校参数
  - 灵活高效的用户配置界面设计，方便设计的可移植性。
  - 补偿运行期间产生的环境变化
  - 检测低至 0.1 pF 的触摸

1.4 文档规范

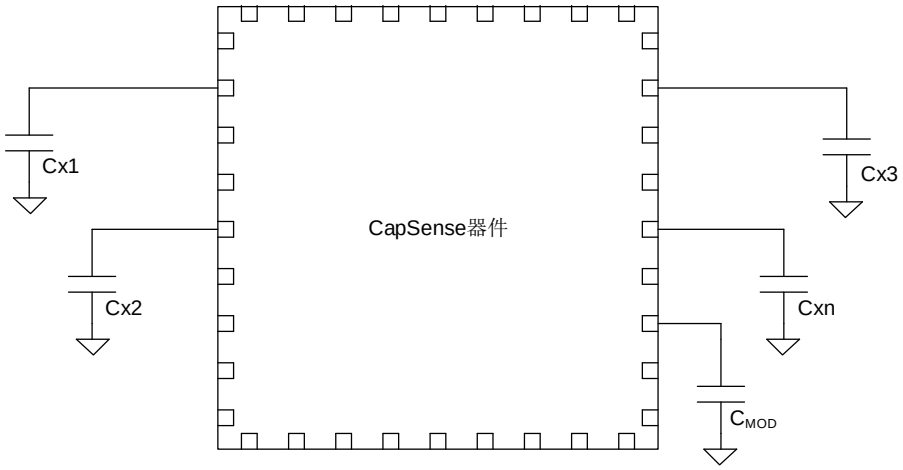
规范	使用说明
Courier New 字体	显示文件位置、用户输入的文本和源代码： C:\ ...cd\icc\
斜体字	用于显示文件名称和参考文档： 请阅读 *PSoC Designer 用户指南*中的 *sourcefile.hex* 文件。
[方括号、粗体]	显示程序中的键盘指令： **[Enter]**或**[Ctrl] [C]**
File（文件）> Open（打开）	表示菜单路径： File > Open > New Project
粗体字	显示操作过程中的各条指令、菜单路径和图标名称： 请点击 **File** 图标，然后点击 **Open**。
Times New Roman 字体	用于显示公式： 2 + 2 = 4
灰色框中的文本	用于说明警告或产品的独特功能。

2. CapSense 技术

2.1 CapSense 基本原理

CapSense 是一种触摸感应技术，用于测量被指定为传感器的各个 I/O 引脚的电容大小。每个传感器引脚的总电容值可以模拟为等效的一个总容值，从 Cx1 到 Cxn，如图 2-1 所示。CapSense 技术需要一个外部调节电容器（C_{MOD}）。电容转换一节中将会更加详细地介绍 C_{MOD}。

图 2-1. CapSense 器件扫描传感器 Cx1 至 Cxn



根据要求使用走线、过孔或同时使用二者将每个传感器 I/O 引脚连接至触摸传感器。每个触摸传感器需要覆盖非导电性覆盖层来构成系统的触摸界面。当手指与覆盖层接触时，人体组织的导电性会产生一个与传感器板并行的接地导电层。如图 2-2 所示。此操作构成一个平行板电容器，其容值可通过下面公式计算得到：

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

其中：

C_F = 手指与传感器覆盖层接触时所产生的电容值

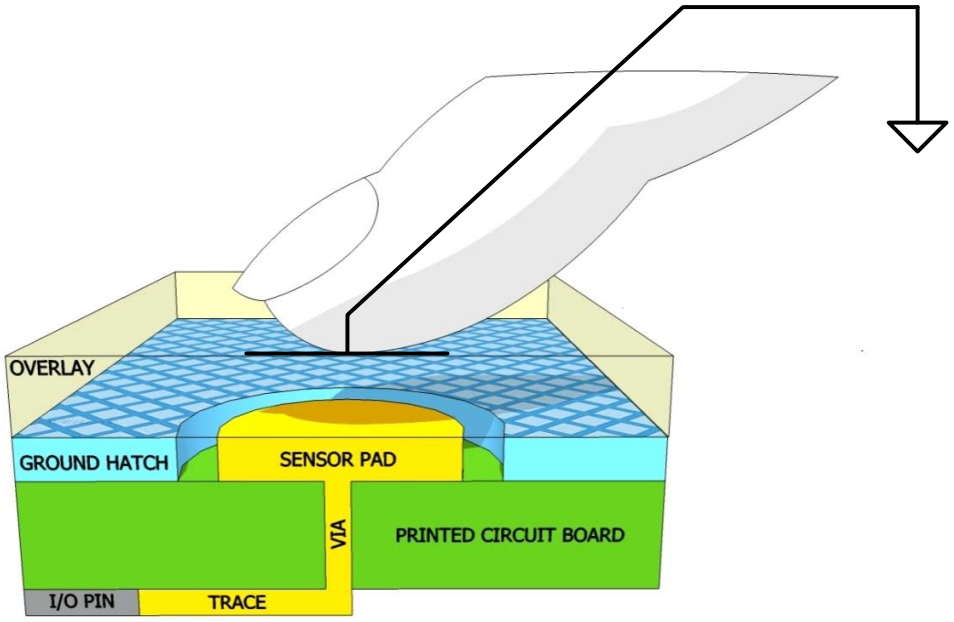
ε_0 = 空气介电常数

ε_r = 覆盖层的介电常数

A = 手指与传感器板覆盖层的接触面积

D = 覆盖层的厚度

图 2-2. CapSense 系统等效模型



即使手指未触摸覆盖层，传感器输入引脚也会有一些寄生电容（C_P）。C_P 是由 CapSense 控制器内部寄生电容与电场共同产生的，其中电场是在传感器焊盘、走线和过孔以及系统中其他导体（如接地层、其他走线、产品底架或外壳中的所有金属）之间耦合产生的。由于 C_P 和 C_F 均用于连接传感器引脚与地，因此两者彼此平行。

当手指未触摸传感器时：

$$C_X = C_P$$ 公式 2

当手指触摸传感器时：

$$C_X = C_P + C_F$$ 公式 3

通常，C_P 比 C_F 大几个数量级。C_P 的范围通常为 6 pF 至 15 pF，但在极端情况下可以高达 45 pF。C_F 的取值范围通常为 0.1 至 0.4 pF。调校 CapSense 系统时，C_P 的大小至关重要。要获得最佳性能，C_P 应尽量保持较低数值。

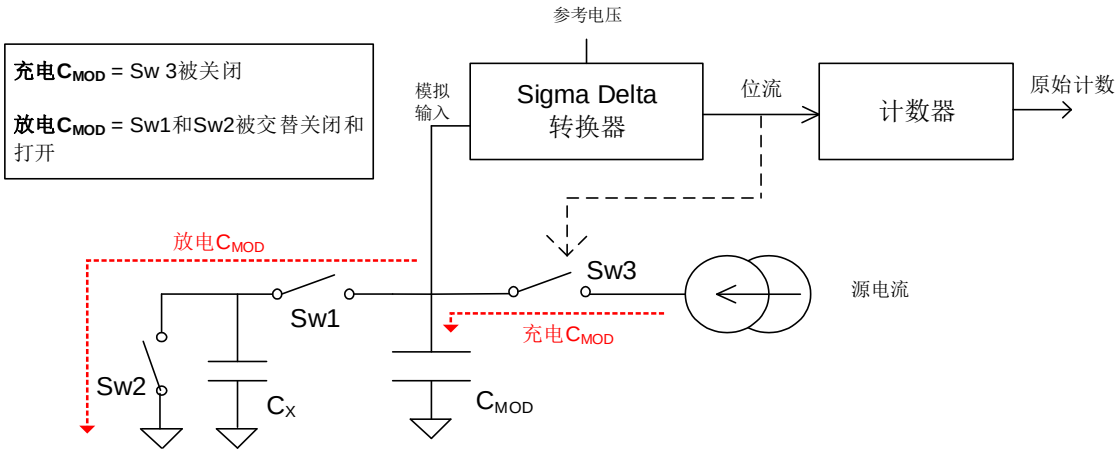
2.2 电容转换

CapSense 器件将不同大小的 Cx 转换为数字计数进行保存和处理，以便检测传感器焊盘上或附近是否存在手指。

2.2.1 CapSense Sigma Delta（CSD）

CapSense Sigma Delta（CSD）是一种将传感器电容转换为数字计数的方法。CSD 方法优于其他方法。图 2-3 显示的是 CSD 方法的框图。

图 2-3. CapSense CSD 框图



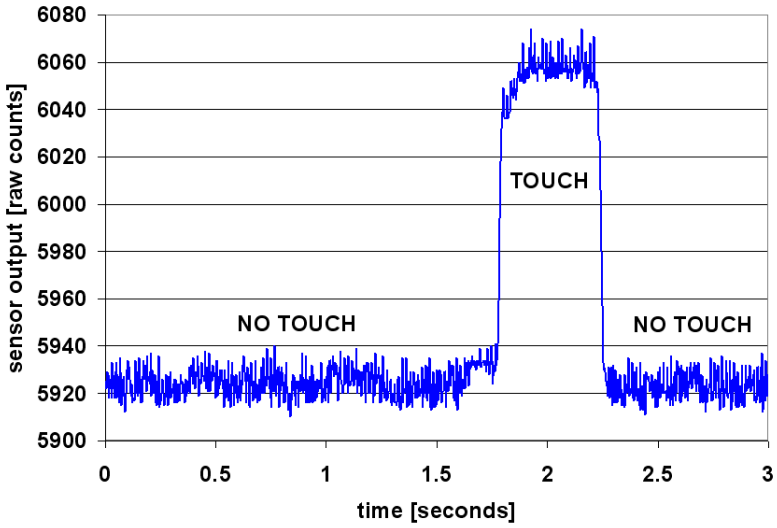
CSD 系统需要一种名为调制电容（C_{MOD}）的积分大电容来配合工作。传感器电容 C_X 快速交替连接到开关 Sw1 和 Sw2，从而形成开关电容电路。Sw1 用于将 C_X 连接到 C_{MOD}，而 Sw2 用于将 C_X 接地。Sw1 和 Sw2 交替开关，并不会产生重叠，因此为 C_{MOD} 提供了一条放电路径。

Sigma delta 转换器用于将 C_X 的电容转换为数字计数。恒流源通过开关 Sw3 连接至 C_{MOD}。当关闭 Sw3 时，电流源为 C_{MOD} 充电。C_{MOD} 作为输入信号连接到 sigma delta 转换器。根据其电压输入，Sigma delta 转换器控制 Sw3 开关以便将 C_{MOD} 的平均电压维持在参考电压的水平。Sigma delta 控制器的输出为一组比特流，该值表示 Sw3 的占空比。

Sw3 的占空比与 C_X 电容值成正比。例如，较大的 C_X 值将产生较大的 C_{MOD} 放电电流。要将 C_{MOD} 电压维持在参考电压，Sigma delta 转换器应增大 Sw3 的占空比。

转换为数字值的比特流称为原始计数。原始计数由高级算法进行解析，来确定传感器的状态。当手指接触传感器时，C_X 会加大 C_F，原始计数将按比例增大。通过对比稳定状态下原始计数的变化量和预定阈值，高级算法能够确定传感器是在 ON（触摸）还是 OFF（未触摸）状态。图 2-4 显示了手指触摸传感器并维持若干扫描周期之后释放得到的 CSD 计数波形。

图 2-4. 原始计数与时间



有关赛普拉斯 CSD 感应技术的深入探讨，请参见 PSoC 3、PSoC 5LP Architecture TRM。

2.2.2 CSD 实现

有多种方法可实现 CSD 方法。

IDAC 源电流方法：IDAC 对 C_{MOD} 充电，而传感器对 C_{MOD} 放电，如图 2-3 所示。

IDAC 灌电流方法：IDAC 对 C_{MOD} 放电，而传感器对 C_{MOD} 充电。

外部电阻方法：外部电阻对 C_{MOD} 放电，而传感器对 C_{MOD} 充电。

这三种实施形式将在电流源方法一节中详细讨论。

2.3 SmartSense 自动调校

组成 CapSense 系统的硬件和固件拥有若干参数，可用于确定系统的运行方式。调校这些参数对于确保系统正常运行和用户的良好体验非常重要。但不幸的是，调校非常费时，因为它是一个重复的过程。在典型的开发周期中，会在初始设计阶段时调校界面（即在系统集成过程中和在产品量产前）。

SmartSense 自动调校是赛普拉斯的先进技术。SmartSense 是一种复杂算法，可自动优化一系列应用中的系统性能。它易于使用，并且在原型和制造阶段通过消除手动调校缩短了设计周期。SmartSense 自动调校在加电时对每个 CapSense 按键进行自动调校，并维护运行时的最佳按键性能。该技术适合解决 PCB、覆盖层的制造误差，并且能够自动消除噪声源（如 LCD 反相器、交流线路和切换模式电源）产生的噪声。

SmartSense 自动调校会在自动调校一节中详细介绍。

3. PSoC 3 和 PSoC 5LP 中的 CapSense

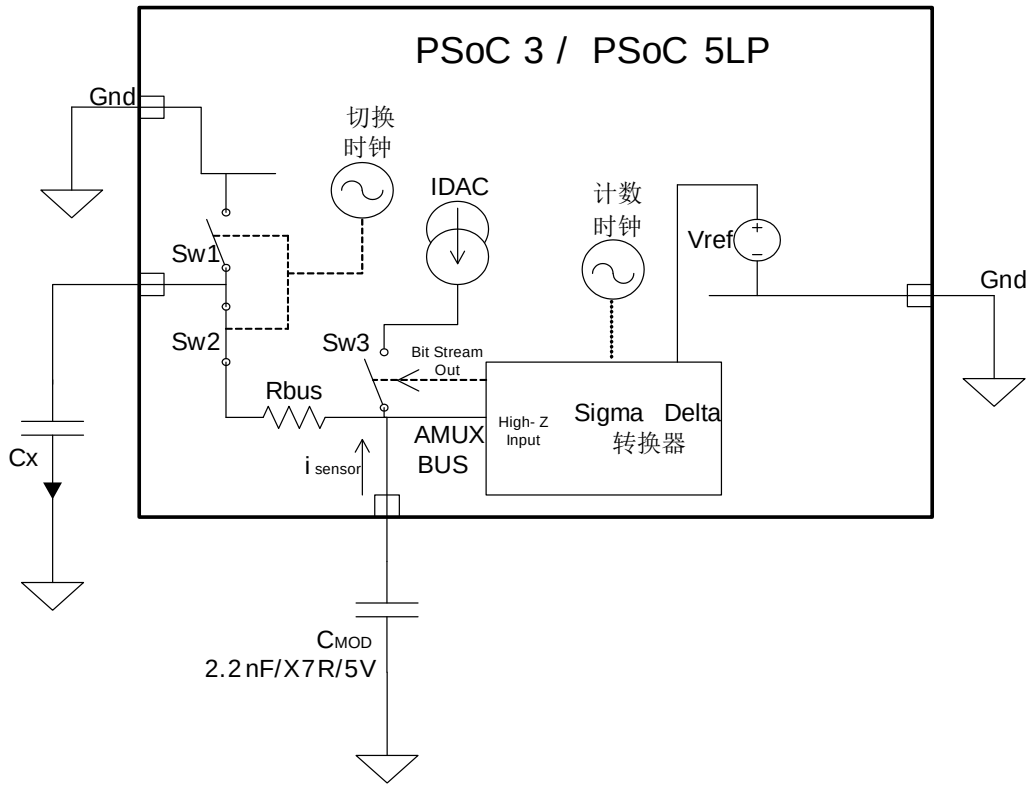
PSoC 3 和 PSoC 5LP 器件是可编程器件，具备一系列丰富的模拟和数字资源。这些器件通过多种独有的方式实现 CapSense CSD 方法。由于这些器件非常灵活且资源丰富，因此除 CapSense 之外，它们还可执行多种其他系统功能。

3.1 CSD 实现

请参考电容转换一节，了解 CSD 方法的详细信息。图 3-1 显示了如何在 PSoC 3 和 PSoC 5LP 器件中实现 CSD 方法。这些器件拥有多达四个 DAC 资源。CapSense 使用其中一个 DAC 资源来实现电流源（IDAC）。电容开关由非重叠的时钟（开关时钟）进行控制。模拟复用器总线（AMUXBUS）同时连接 C_{MOD}、C_X、IDAC 和 sigma delta 转换器输入。Sigma-Delta 转换器控制着 IDAC，从而使 C_{MOD} 电压维持在参考电压（Vref）附近。Sigma delta 转换器将比特流输出到某个计数器。计数器输出原始信号。然后，CPU 会处理原始信号，并确定传感器的状态。

图 3-1 显示的是实现 IDAC 源电流 CSD 的方法。有关其他方法的信息，请参见电流源方法一节。

图 3-1. CSD 实现（IDAC 源电流方法）



3.2 独特的 CapSense 功能

3.2.1 双通道设计

PSoC 3 和 PSoC 5LP 器件可以同时扫描两个传感器。这样可减少一半的扫描时间。缩短扫描时间可延长按钮开/关检测的响应时间，同时降低平均功率损耗。

由于使用双通道所消耗的资源是使用单通道的两倍，因此建议设计中传感器数量超过 20 时才使用双通道。通道数量选项解释了如何选择双通道设计。表 3-1 对单通道和双通道设计所需资源进行了对比

表 3-1. 单通道和双通道设计的资源对比

资源类型	单通道	双通道
模拟资源[1]	1 或 2 个 AMUXBUS 1 个电压比较器 1 个 DAC	2 个 AMUXBUS 2 个电压比较器 2 个 DAC
数字资源[2]	4 个数据路径 19 个宏单元	6 个数据路径 31 个宏单元
外部组件[3]	1 个 C_{MOD} 电容器 1 个泄流电阻	2 个 C_{MOD} 电容器 2 个泄流电阻

3.2.2 防水设计

某些 CapSense 电容式触摸感应系统需要在带有水份环境进行可靠的工作。白色家电、汽车应用和工业应用均涉及水、冰和湿度变化的环境，传感器系统必须在这种环境下完成工作。对于这样的应用环境而言，屏蔽电极和保护传感器可以提供强健的触摸感应系统。

如果您的应用要求防水滴和潮气，则应使用屏蔽电极。屏蔽电极是传感器的边缘区域，如图 3-2 所示。屏蔽电极与 I/O 引脚相连，并通过与传感器引脚相同的开关信号驱动屏蔽电极。使用相同的信号来驱动屏蔽电极和传感器引脚会使两者之间的电容相互抵消。这意味着传感器和屏蔽区域中的部分水膜或水滴引起的电容变化将被有效地消除。屏蔽电极还可降低传感器的 C_P。所需的 I/O 引脚数量取决于电路板面积和屏蔽区域。如果屏蔽电极区域较大，则必须使用多个 I/O 引脚才可驱动它。

如果您的应用要求接触面能够防水流，则应同时使用屏蔽电极和保护传感器。保护传感器应包围整个触摸感应区。通常，保护传感器沿着触摸感应区域的外围进行布线，如图 3-2 所示。CapSense 器件采用与其他传感器相同的扫描方式扫描保护传感器。当保护传感器上存在液体时，该传感器将被激活并禁止对其他传感器的扫描，以防止检测误手指触摸。

屏蔽电极和保护传感器一节阐述了如何使能屏蔽电极和保护传感器。

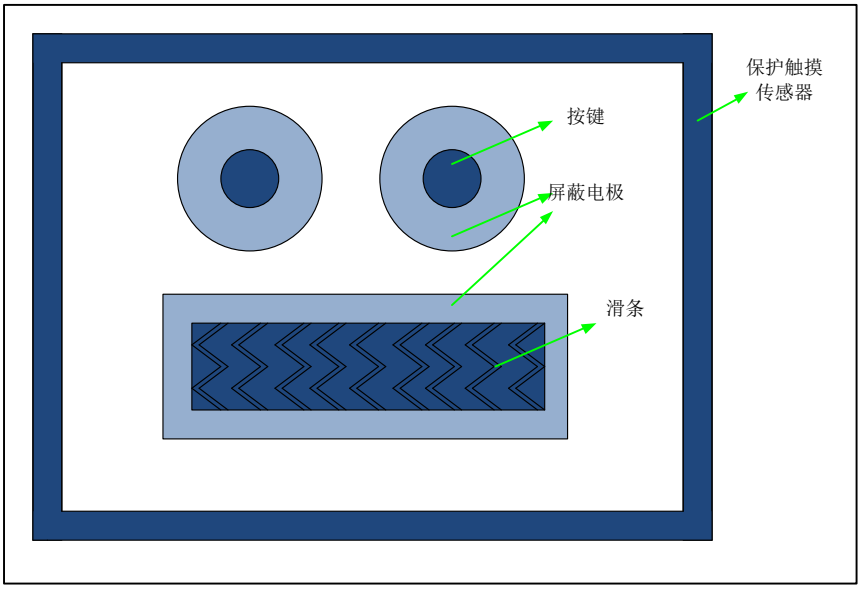
---

[1] PSoC3/5LP 拥有一个左侧 AMUXBUSL 和一个右侧 AMUXBUSR。当所有传感器都位于芯片同一侧（上侧或下侧）时，单通道设计仅需要使用一个 AMUXBUS。双通道设计需要将传感器分两组放置在芯片两侧（上侧或下侧），两组传感器分别连接到彼此独立的 AMUXBUSL 和 AMUXBUSR。
当选择 IDAC 源电流或 IDAC 灌电流方法时，最少需要选用一个 DAC。请参见电流源方法。

[2] 数据路径和宏单元是通用数字模块（UDB）的子模块。PSoC 3 和 PSoC 5LP 器件中有多达 24 个 UDB 模块。

[3] 当选择外部电阻 CSD 方法时，需要使用泄流电阻。请参见电流源方法。

图 3-2. 屏蔽电极和保护传感器



3.2.2.1 SIO 引脚

PSoC 3 和 PSoC 5LP 器件拥有特殊的输入/输出（SIO）引脚，这些引脚具有可编程“逻辑高电平”的特性。这意味着 SIO 的逻辑高电平不是固定到 V_{DD} 而是可编程为用户定义的电压。

当 SIO 作为屏蔽电极使用时，这一特性尤其重要。当屏蔽电极中的信号与 CapSense 传感器中的信号一致时，屏蔽电极将发挥最大作用。使用 SIO 引脚时，您可以选择与 CapSense 传感器相匹配的逻辑高电平。

3.2.3 电流源方法

通过 PSoC 3 和 PSoC 5LP 器件，您可以选择三种不同 CSD 实现方法中的一种。影响设计中最佳执行方法的因素包括：资源要求、噪声敏感度以及屏蔽电极要求。对于所有方法，Vref 的默认值等于带隙电压（1.024 V）。

表 3-2 对三种实现方法进行了对比。请注意，基于不同的芯片编号，PSoC 3 和 PSoC 5LP 器件中最多包含四个 DAC。如果设计中的其他功能使用了全部 DAC，则 CapSense 可以使用外部电阻方法。电流源一节介绍了如何选择电流源的方法。

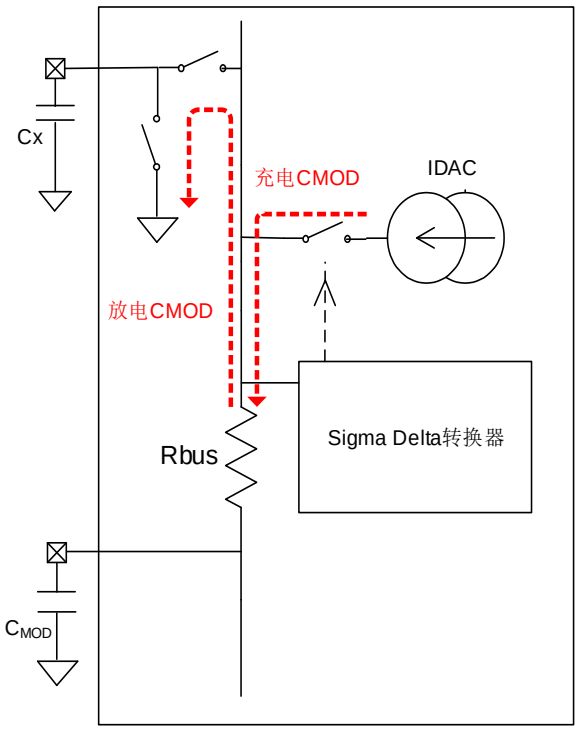
3.2.3.1 IDAC 源电流方法

在此配置中，CapSense 传感器在 C_{MOD} 和 GND 之间连续切换，以对 C_{MOD} 进行放电。IDAC 配置为源电流模式且在 C_{MOD} 电压低于 Vref 时为打开状态。图 3-3 显示了 IDAC 源电流的框图。

由于传感器上电压摆幅较小（电压摆幅为从 Vref 到 GND），所以很容易受到手指传导噪声的影响。增大 Vref 和 VDAC 会提高抗噪声能力，但需要一个额外的 DAC 资源。

当使用 IDAC 源电流方法时，应将 SIO 引脚作为屏蔽电极。这样您才能切换 Vref 和 GND 之间的屏蔽电极，从而确保它获得与传感器相同的信号。

图 3-3. IDAC 源电流方法

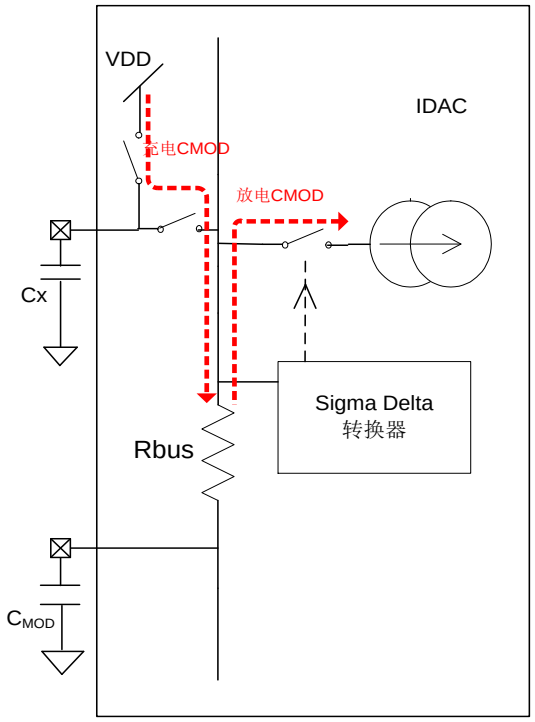


3.2.3.2 IDAC 灌电流方法

在此配置中，CapSense 传感器在 V_{DD} 和 C_{MOD} 之间连续切换，以对 C_{MOD} 进行充电。IDAC 配置为灌电流模式且在 C_{MOD} 电压超过 Vref 时为打开状态。图 3-4 显示了 IDAC 灌电流方法的框图。

因为传感器在 V_{DD} 和 Vref 之间来回切换，所以这些方法易受电源噪声的影响。

图 3-4. IDAC 灌电流方法



3.2.3.3 外部电阻方法

除使用了一个外部电阻（泄流电阻）来代替 IDAC 对 C_{MOD} 放电外，该方法与 IDAC 灌电流方法相同。CapSense 传感器在 V_{DD} 和 C_{MOD} 之间连续切换，以对 C_{MOD} 进行充电。当 C_{MOD} 电压超过 Vref 时，泄流电阻接地。图 3-5 显示了外部电阻方法的框图。

因为传感器在 V_{DD} 和 Vref 之间来回切换，所以这些方法很容易受到电源噪声的影响。

图 3-5. 外部电阻方法

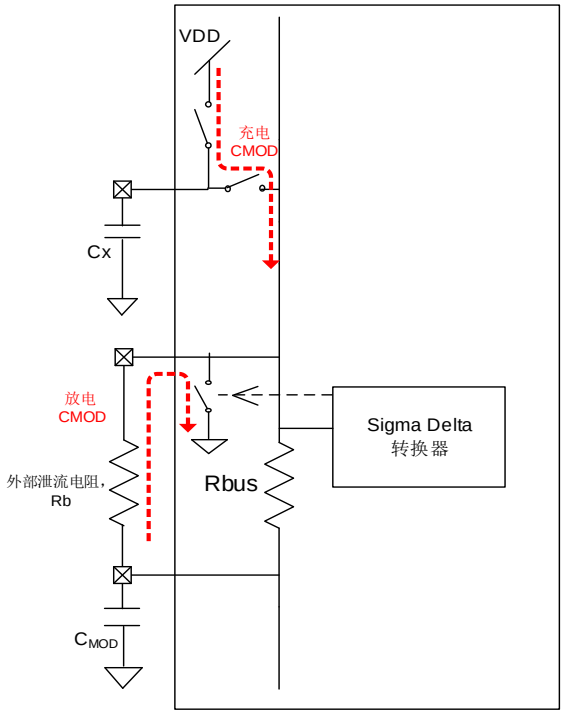


表 3-2. 电流源方法的比较

	IDAC 源电流	IDAC 灌电流	外部电阻
DAC 资源	需要	需要	无需
外部泄流电阻	无需	无需	需要
电源噪声	不受影响	受影响	受影响
手指传导噪声	受影响 应使用 VDAC 增大 Vref	不易受影响	不易受影响
屏蔽电极	SIO 引脚	任何 GPIO 引脚	任何 GPIO 引脚

3.2.4 调校

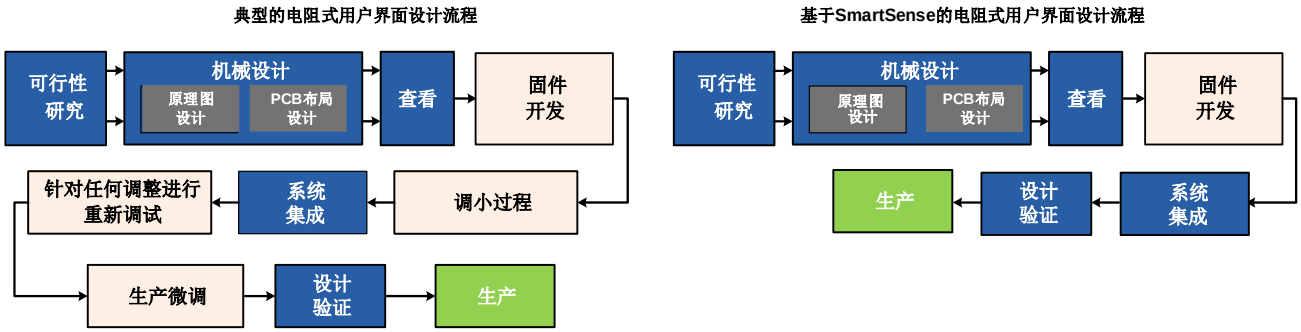
SmartSense 自动调校是一个高级技术，在加电时对每个 CapSense 按键进行自动调校，并维护运行时的最佳按键性能。它降低了设计周期时间并严格控制工艺变化。如果您需要更多的控制参数或 C_P 为高，那么可以手动调校 CapSense 参数。调校方法一节中会介绍如何选择调校方法。CapSense 功能调校一节介绍了自动调校和手动调校的流程。

3.2.4.1 自动调校功能

缩短了设计周期时间

图 3-6 介绍的是 SmartSense 自动调校明显降低设计周期时间的方式。

图 3-6. 设计周期比较

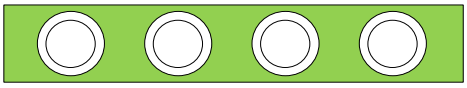


下述示例显示的是 SmartSense 自动调校大大节省时间并实现平台设计的方式。图 3-7 和图 3-8 显示的是分别针对 21 英寸和 15 英寸笔记本机型设计的多媒体按键。这些按键均是 CapSense 按键，具有相同的功能和尺寸。但是，对于 21 英寸的笔记本，按键具有较宽的空间，并且按键和 CapSense 控制器之间的走线较长，所以在这两种机型之间不能直接移植设计。所以对设计进行重新调校。借助 SmartSense 自动调校，开发人员可以将相同的设计移植到其他机型中，从而大大缩短了设计时间。

图 3-7. 21 英寸笔记本电脑的多媒体按键设计



图 3-8. 15 英寸笔记本电脑的多媒体按键设计（与 21 英寸机型有相同的功能和按键大小）



消除工艺变化

C_P 可能由于 PCB 布局和走线长度、不同的 PCB 制造工艺或多源供应链中不同的 PCB 供应商，而有所差异。按键的灵敏度取决于 C_P 的大小；C_P 的值越高，灵敏度就越低，进而导致手指触摸信号振幅降低。C_P 值的变化，会使某个按键过于灵敏、灵敏度不够或者不能正常工作。当发生这种情况时，您必须重新调校系统，有时候需要重新认证用户界面子系统。SmartSense 自动调校可以解决这些问题。

易用性

通过 SmartSense 自动调校，您不需要深入了解全部参数也能够快速轻松地设计 CapSense 应用。

3.2.4.2 手动调校功能

深度控制

通过手动调校，您可以选择每个 CapSense 参数值。对于工作条件特殊的系统（如具有高噪声的系统），该内容非常重要。在噪杂的系统中使用自动调校可能会引起按键误触问题。手动增大手指阈值可提高系统的抗噪能力。

高寄生电容

设计自动调校主要用于控制 C_P，使其取值范围为 5 ~ 45 pF。如果由于走线较长或按键尺寸较大使 C_P 值高于 45 pF，则应使用手动调校。

错误检测算法

自动调校在启动时和运行时均可更改参数。这样会难以写入错误检测算法。

比如：通过错误检测算法检查基准线计数是否位于出场闪存中所存储的数值范围内。假设在扫描分辨率为 12 位时进行测量，则存储值为 3000 次计数。基准线计数是 C_P 的平均计数，并且扫描分辨率为电容测量分辨率（请参考原始计数和 Scan Resolution（扫描分辨率），深入了解这些术语的详细解释）。使用自动调校时，根据物理或环境条件的变化可以将扫描分辨率参数设置为 13 位。将扫描分辨率设置为 13 位时，基准线计数为 6000，并且系统会失败。

更低的 RAM 和闪存使用

自动调校算法要求更多的闪存和 RAM。表 3-3 显示了含有四个传感器的 CapSense 项目所需的存储器。

表 3-3. 手动调校和自动调校的存储器需求比较

调校方法	PSoC 3		PSoC 5LP	
	RAM（字节）	闪存（字节）	RAM（字节）	闪存（字节）
手动调校	207	6345	384	5104
自动调校	292	8282	488	6152

3.2.5 无阻塞架构

CapSense 扫描具有以下三个阶段：

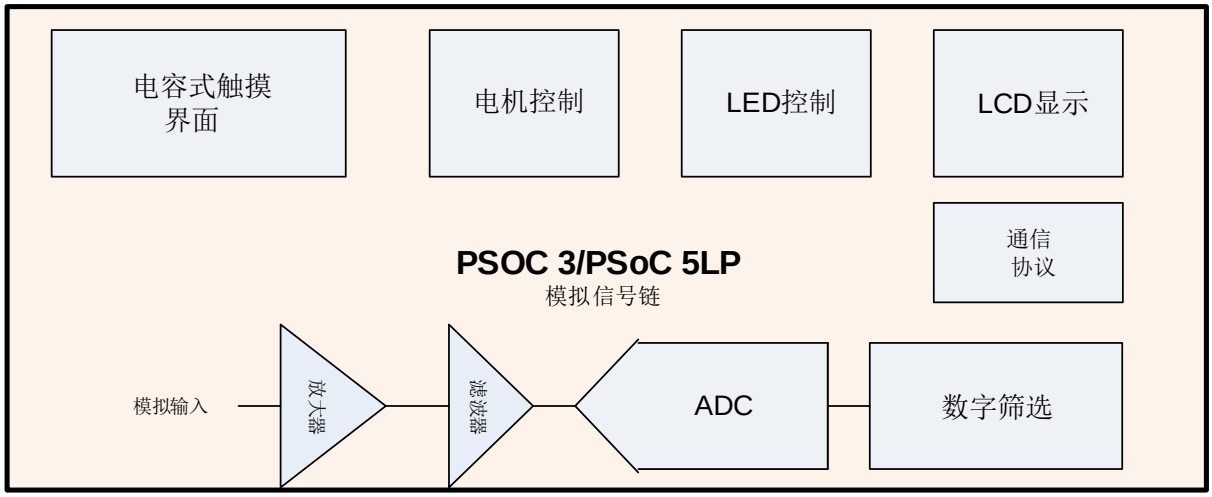
1. **预扫描**：硬件准备
2. **硬件扫描**：实际硬件扫描
3. **后扫描**: 处理和存储结果

只有第一阶段和第三阶段要求 CPU 执行代码。代码架构无需等待第二阶段完成。相反，CPU 将执行下一行代码，并且硬件在完成扫描时会生成一个中断，CPU 将在 ISR 中执行扫描后的代码。该无阻塞架构对 CPU 服务多种应用的设计非常有用。

3.3 CapSense PLUS

CapSense PLUS 是指执行 CapSense 附加的各项功能的 PSoC 器件。PSoC 3 和 PSoC 5LP 器件均属于这种类型。这些器件提供的各种功能允许您将不同的系统功能集成到单芯片内，如图 3-9 所示。这样可降低电路板面积、BOM 成本以及功耗。

图 3-9. CapSense PLUS 在系统中的其他功能



请访问以下链接，了解更多信息：

- 模拟功能
- 通信调制解调器
- LCD 驱动
- 低功耗
- USB 连接

请访问以下链接，查看使用 PSoC 3 和 PSoC 5LP 器件的成功设计：

- 血糖仪
- 血压计
- 排卵检测器
- 输液泵
- iPod、iPhone 和 iPad 的配件
- LED 投影机
- 磁卡读卡器
- 脉搏血氧计
- 主动快门式 3D 眼镜

4. CapSense 设计工具

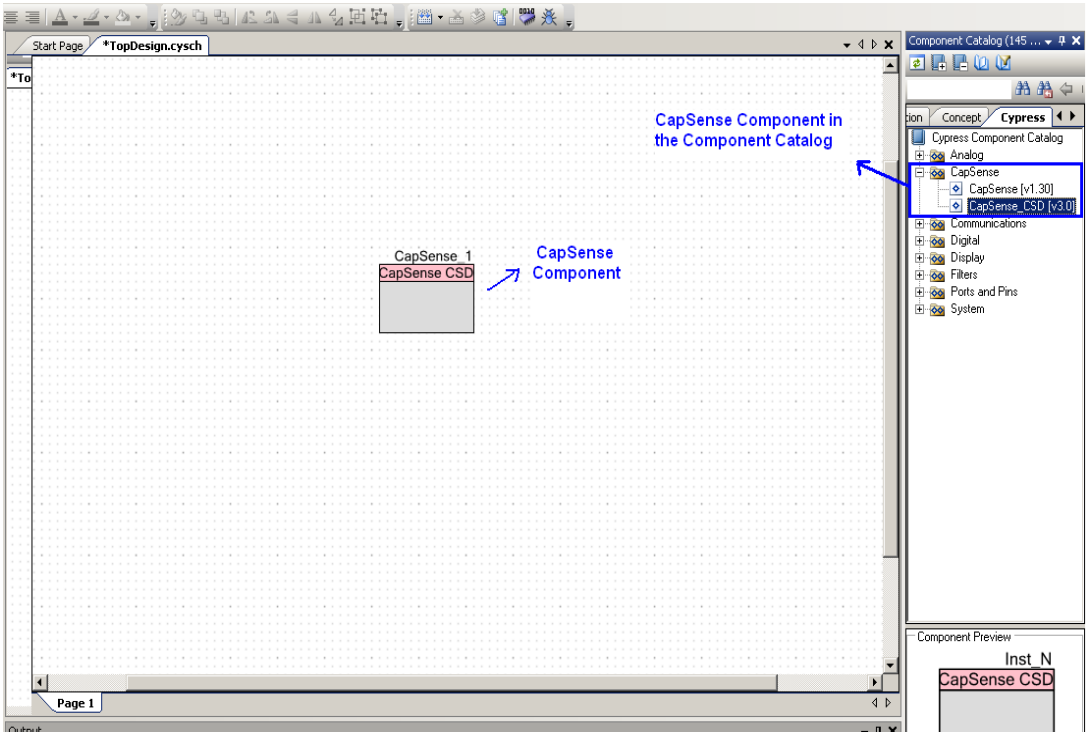
赛普拉斯提供用来开发 CapSense 电容式触摸感应应用的全套硬件和软件工具。有关订购信息，请参见资源。

4.1 PSoC Creator

赛普拉斯的 PSoC Creator 提供一个集成的设计环境。PSoC Creator 可在单个统一工具中提供硬件配置和软件开发的独特组合。在拖放式设计环境中，使用组件库开发应用。

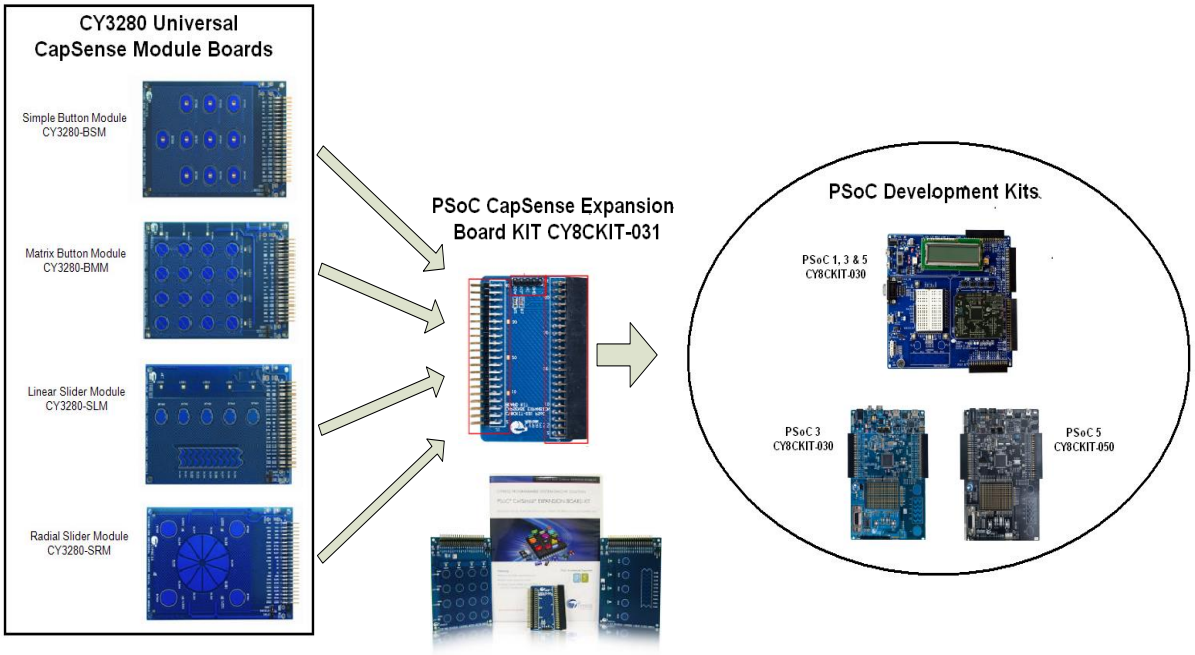
PSoC Creator 可提供 CapSense_CSD 组件。该组件使用开关式电容电路、一个模拟总线、一个电压比较器、数字计数功能和高层软件子程序（API）来实现一个个电容式触摸感应系统。另外，器件还提供了其他模拟和数字组件，用于实现 I²C、SPI、UART、定时器、PWM、放大器、ADC 和 LCD 等其他功能。图 4-1 显示的是从组件目录中拖放并放置在 TopDesign 上的 CapSense_CSD 组件。

图 4-1. PSoC Creator TopDesign



4.2 硬件套件

图 4-2. CapSense 硬件



4.2.1 PSoC 3 和 PSoC 5LP 开发套件

PSoC 3 和 PSoC 5LP 开发套件支持 CapSense 功能。

- CY8CKIT-001 PSoC®开发套件
- CY8CKIT-030 PSoC® 3 开发套件
- CY8CKIT-050B PSoC® 5LP 开发套件

4.2.2 通用 CapSense 模块板

赛普拉斯提供通用的 CapSense 模块板，该模块板具有各种 CapSense 传感器、LED 和接口来满足您的需求。

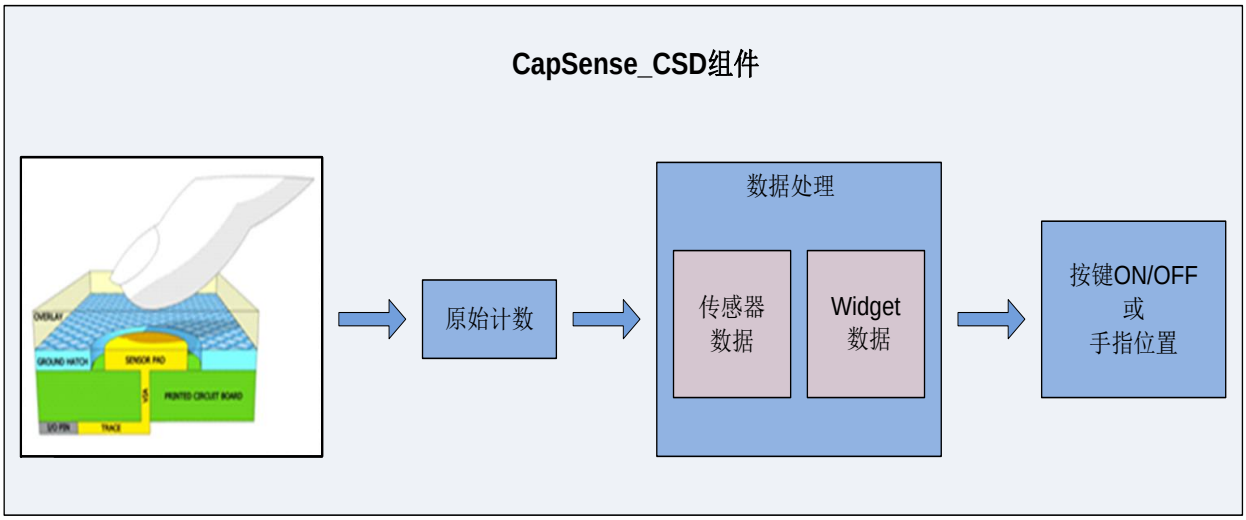
- CY3280-BSM 简单按键模块
- CY3280-BMM 矩阵按键模块
- CY3280-SLM 线性滑条模块
- CY3280-SRM 辐射滑条模块
- CY3280-BBM 通用 CapSense 原型设计模块

4.2.3 PSoC CapSense 扩展板套件

CY8CKIT-031 PSoC CapSense 扩展板套件能够将所有 PSoC 3 和 PSoC 5LP 开发套件连接到任意一个通用的 CapSense 模块板上。CY8CKIT-031 可提供接口电路板和两个模块电路板 CY3280-SLM 和 CY3280-BMM。

5. CapSense_CSD 组件

图 5-1. CapSense_CSD 组件框图



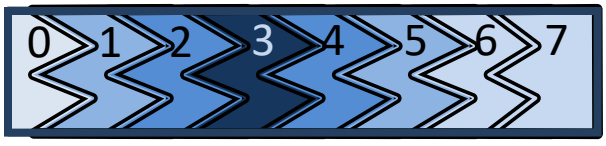
通过 CapSense_CSD 组件，可实现完整的 CapSense 系统。该组件拥有的一些参数被分类为高层和低层。这些参数在固件中使用全局数组进行彼此通信。

高层参数控制 CapSense 系统将原始计数转换为传感器 ON/OFF（触摸/未触摸）状态等有用信息的操作方式。高层参数包括手指阈值、噪声阈值和去抖计数等。请参考高层参数。

低层参数控制 CapSense 系统在物理层的运行方式。在物理层上，电容可转换为原始计数。低层参数包括 IDAC 范围、IDAC 值和扫描时钟等。请参见低层参数。

CapSense_CSD 组件提供一些不同类型的触摸接口（被称为 Widget）。Widget 包含一个或多个传感器。它表示一个接口对象，如一个滑条或一个触控板。Widget 类型包括：按键、滑条、辐射滑条、矩阵按键、触控板和接近感应传感器。图 5-2 显示了滑条的示例。要构成滑条 widget，应沿着一条直线来放置七个传感器，并在传感器之间保留较小的间隔。

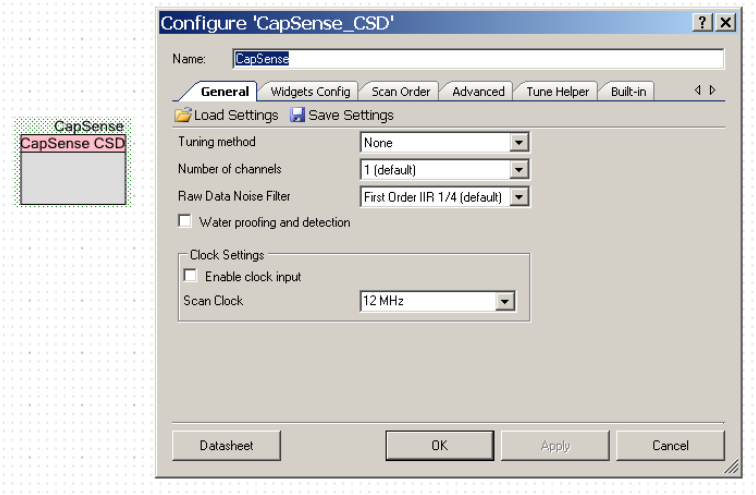
图 5-2. 一个七段滑条



5.1 参数汇总

图 5-3 显示的是 PSoC Creator 中的 CapSense_CSD 组件截屏以及配置窗口。可通过双击组件或者右键单击并选择“Configure”（配置）来打开配置窗口。

图 5-3. PSoC Creator CapSense 组件



配置窗口的不同选项卡下排列着多个参数。表 5-1 总结了该窗口中各可视参数，并提供了相关链接。

表 5-1. CapSense_CSD 组件配置窗口参数

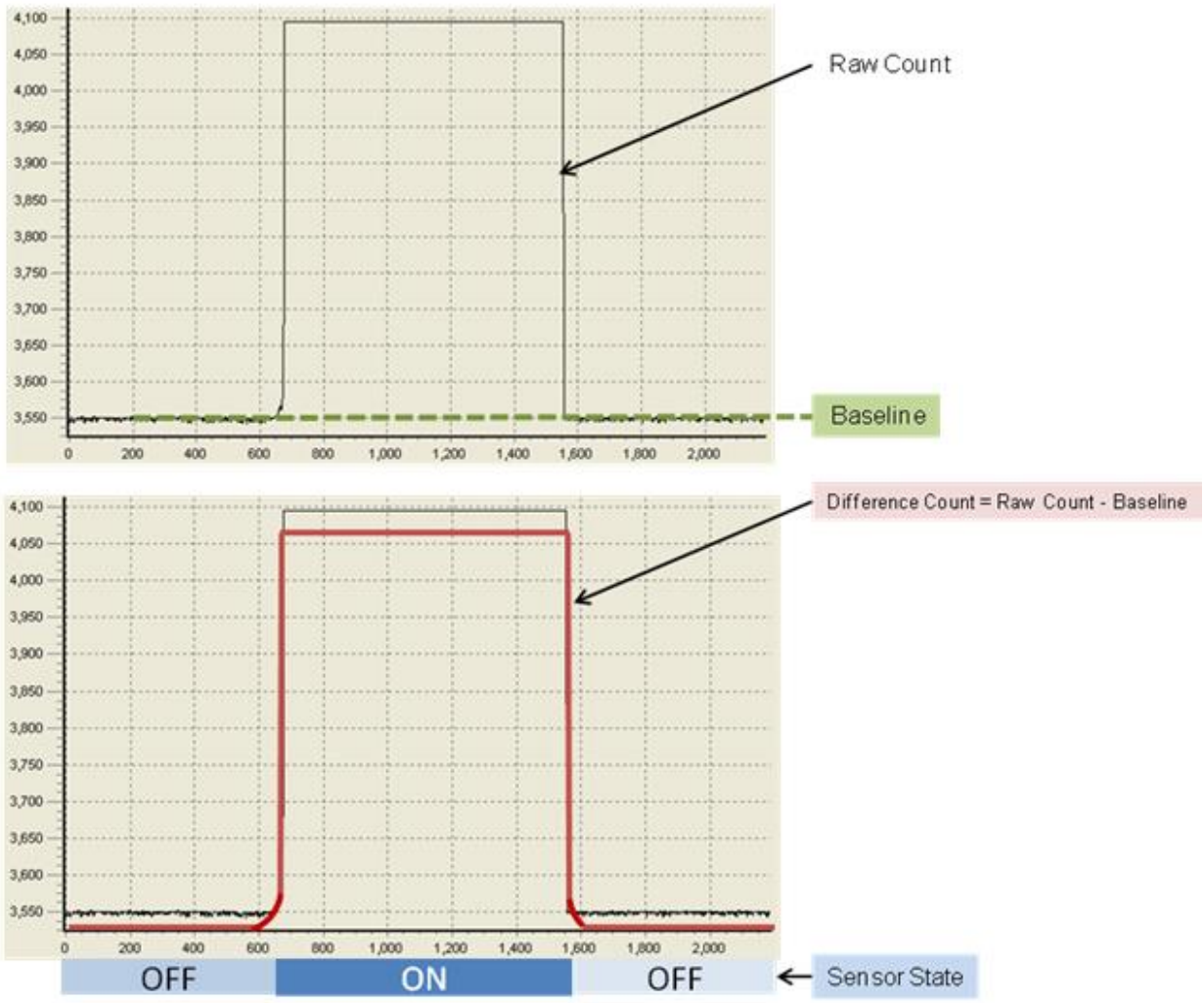
普通	Widget 配置	扫描顺序	高层		调谐器助手
调校方法	添加 Widget	模拟开关分频器	模拟开关驱动源	屏蔽	使能调谐器助手
通道数量	手指阈值	IDAC 值	多模拟开关分频器	非活动状态传感器连接	EZI2C 组件的实例名称
原始数据噪声滤波器	噪声阈值	移到通道 1/移到通道 0	模拟开关分频器	保护传感器	
防水和检测	迟滞	灵敏度	扫描速度	电流源	
使能时钟输入	去抖动		PRS 减少电磁干扰（EMI）	IDAC 范围	
扫描时钟	扫描分辨率		传感器自动复位	泄流电阻的数量，通道 0/通道 1	
	传感器元素的数量		Widget 分辨率	数字资源实现，通道 0/通道 1	
	API 分辨率		负噪声阈值	电压参考源	
	位置噪声滤波器		低基准线复位		
	专用传感器元素数量				

低层参数
高层参数
Widget 参数
其他系统参数

5.2 全局数组

CapSense 系统使用了某些全局数组，用于确保在环境变化时能正确进行检测和操作。不应该对这些数组手动更改，但如需调校，您可对其进行检查。

图 5-4. 全局参数



5.2.1 原始计数

CapSense 控制器硬件测量电容并提供以数字形式表示的结果，该结果称为原始计数。原始计数的值随着传感器电容的增大而增大。

5.2.2 基准线

传感器的原始计数值随着温度和湿度等环境因素的变化而逐渐变化。这种渐变通过基准线值补偿。基准线使用软件算法跟踪原始计数中的渐变。它是一个低通滤波器，对原始计数突变的灵敏度较低。基准线值为计算计数差值（信号值）提供了参考值。

5.2.3 计数差值（信号值）

计数差值（信号值）指传感器的原始计数与基准线之间的差值。通常，当传感器为非活动状态时，计数差值（信号值）为零。触摸传感器会导致原始计数增大，从而产生正计数差值（信号值）。

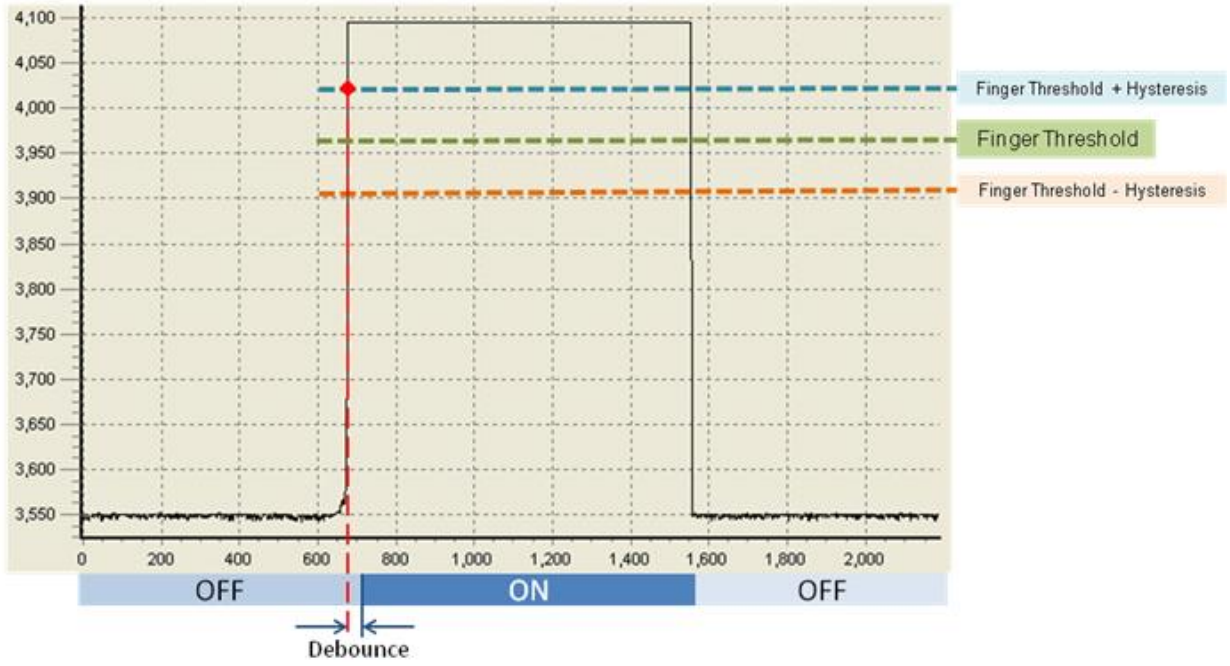
5.2.4 传感器状态

如果按键状态为 ON，各个传感器的状态将表示为 1；如果按键状态为 OFF，则各个传感器状态表示为 0。所有传感器的 ON/OFF 状态都将存储在字节数组中。每个数组元素可容纳八个传感器的 ON/OFF 状态。

5.3 高层参数

高层参数定义如何处理原始计数来产生相关信息，如：传感器 ON/OFF 状态和手指在滑条上的大致位置。图 5-5 和公式 4 对高层参数进行了概述。

图 5-5.高层参数



公式 4

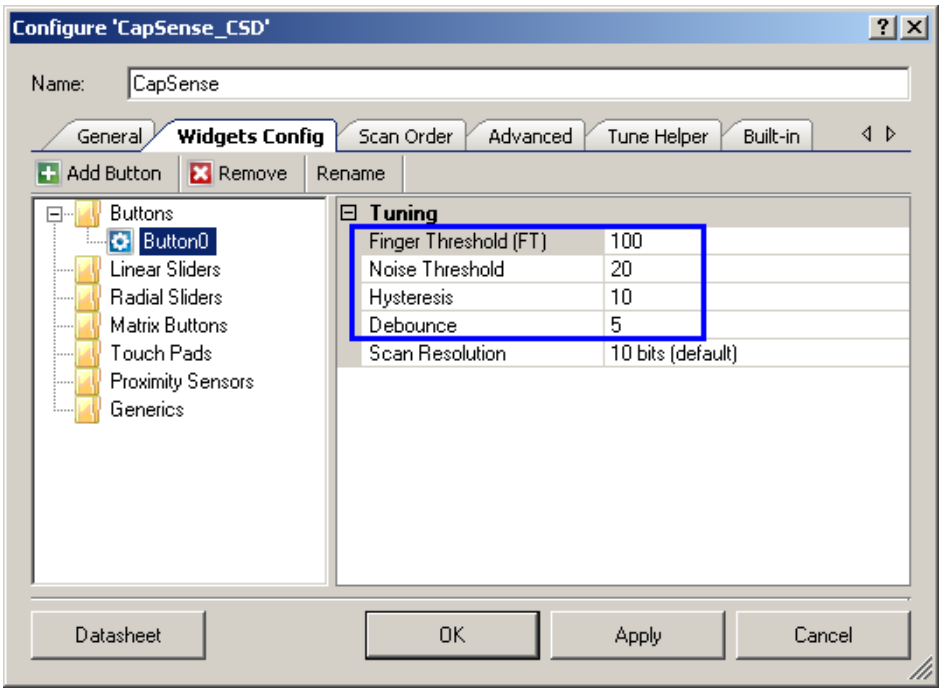
$$if\ (Difference\ Count \geq Finger\ Threshold\ + Hysteresis)\ \&\ (Sample\ Count\ \geq Debounce),\quad Sensor\ State = ON$$

$$if\ (Difference\ Count\ \leq Finger\ Threshold - Hysteresis),\quad Sensor\ State = OFF$$

其中：

采样计数 = 大于手指阈值的采样数 + 迟滞

图 5-6. 高层参数



5.3.1 手指阈值

手指阈值参数定义传感器对手指触摸的灵敏度。它与迟滞参数一起使用，可确定传感器的状态，如公式 4 所定义。取值范围为 0 到 255。

5.3.2 迟滞

迟滞参数结合手指阈值一起使用，可确定传感器的状态，如公式 4 所定义。迟滞可提高抗噪声跃变的能力。这就是按键的去抖动特性。在计数差值（信号值）略高于手指阈值之前，触摸状态将一直保持为 OFF 状态。在计数差值（信号值）略低于手指阈值之前，触摸状态将一直保持为 ON 状态。如果计数差值（信号值）杂乱无章，而且是噪声和手指阈值之间的中间值，则这可防止触摸/无触摸状态机触发。

取值范围为 0 到 255。“迟滞”参数的设置必须低于“手指阈值”参数设置。

5.3.3 去抖动

“去抖动”参数为瞬间改变传感器状态（从 OFF 切换为 ON）添加了计数器。为了使传感器从 OFF 跳变为 ON 状态，对于指定数量的采样，计数差值必须大于手指阈值与迟滞之和。

取值范围为 1 到 255。如果将该参数设置为 1，则不提供去抖动功能。

5.3.4 噪声阈值

对于单个传感器，“噪声阈值”参数设置了原始计数的上限值，以更新基准线值。对于滑条传感器，它设置原始计数的下限值，以通过质心计算计算结果。

取值范围为 3 到 255。噪声阈值设置应始终低于或等于“手指阈值”与“迟滞”之差，才可正常使用用户模块。

5.3.5 “负噪声阈值”和“低基准线复位”

图 5-7. “噪声阈值”和“基准线更新”参数

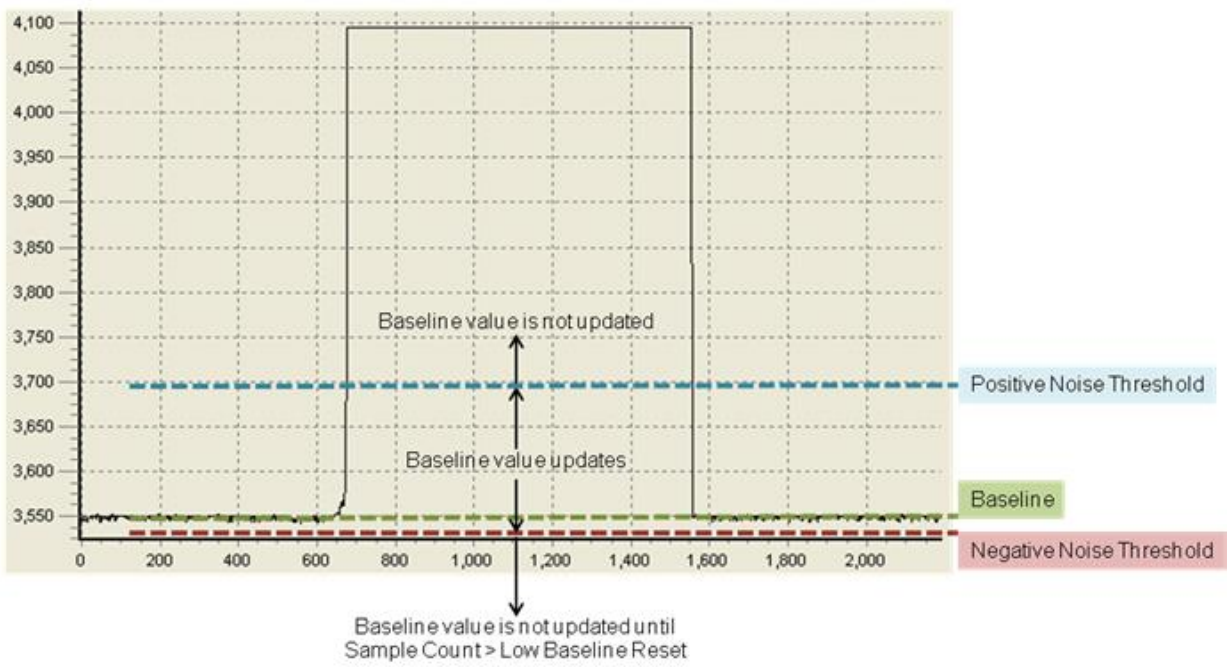
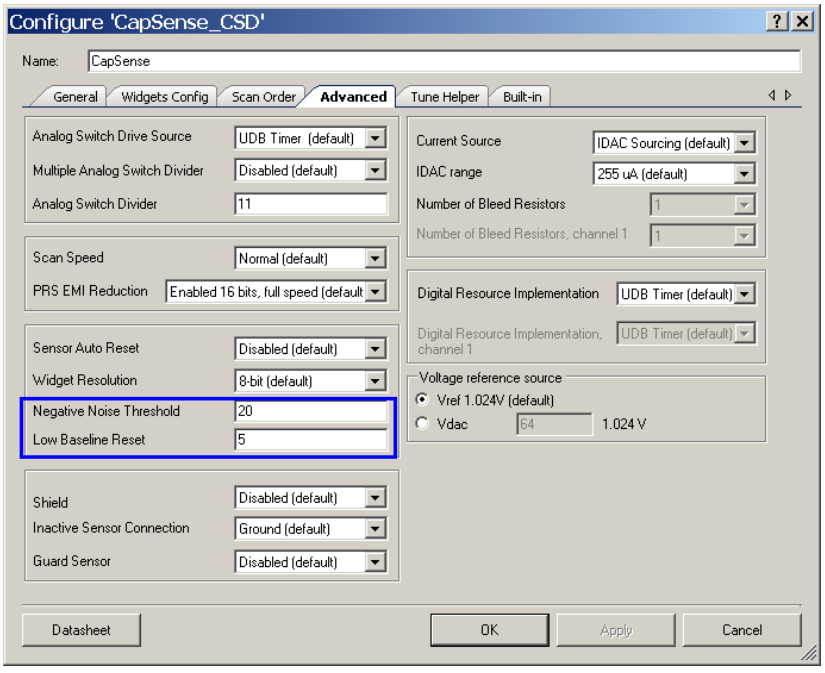


图 5-8. “负噪声阈值”和“低基准线复位”参数

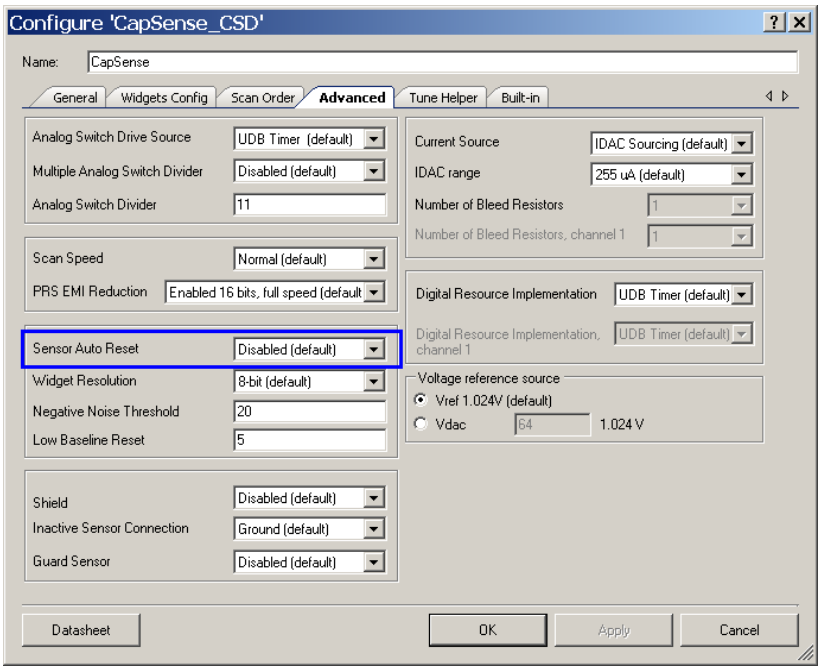


“负噪声阈值”参数可作为负计数差值（信号值）阈值使用。对于由“低基准线复位”参数指定数量的采样，如果原始计数低于基准线与负噪声阈值之差，则应将基准线设置为新的原始计数值。取值范围为 0 到 255。

低基准线复位参数与负噪声阈值参数一同被使用。它计算复位基准线时所需要的异常的低采样数量。它用来修正启动时手指在传感器上触摸的情况。取值范围为 0 到 255。

5.3.6 传感器自动复位

图 5-9. 传感器自动复位参数



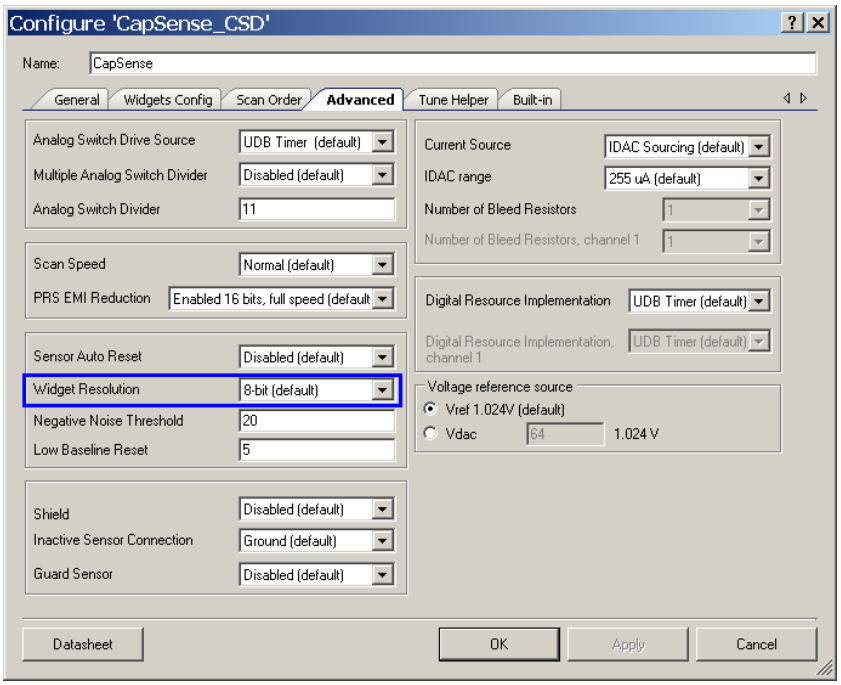
该参数用于确定是否随时更新基准线，或者仅确定计数差值低于噪声阈值的时间。

Enabled（使能） — 基准线随时更新。该设置限制传感器的最大持续时间（典型值为 5 至 10 秒），但可在无任何物体碰触传感器而原始计数突然上升的情况下，阻止传感器始终打开。较大的电源电压波动、高能射频噪声源或极强烈的温度变化都会导致计数突然上升。

Disabled（禁用） — 仅当计数差值低于噪声阈值参数时基准线才进行更新。

5.3.7 Widget 分辨率

图 5-10. Widget 分辨率参数



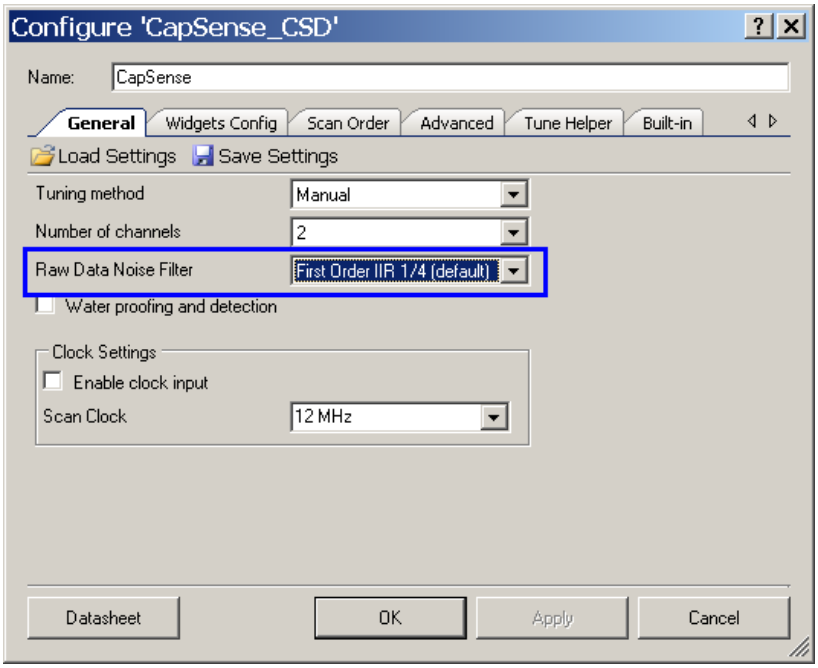
该参数决定对信号计数使用 8 位变量还是 16 位变量。信号计数是原始计数与基准线之差，如图 5-5 所示。该参数被设置为 widget 等级。Widget 中的所有传感器的分辨率应相同。

在大多数情况下，8 位的默认值比较合适。但是，在高电平信号设计中，由于覆盖层较薄或传感器区域较大，因此应该选择 16 位分辨率。因为滑条使用相邻传感器的相对信号来计算手指位置，所以饱和时会导致非平稳运行状态，并且需要 16 位分辨率。

5.3.8 选择滤波器

5.3.8.1 原始数据噪声滤波器

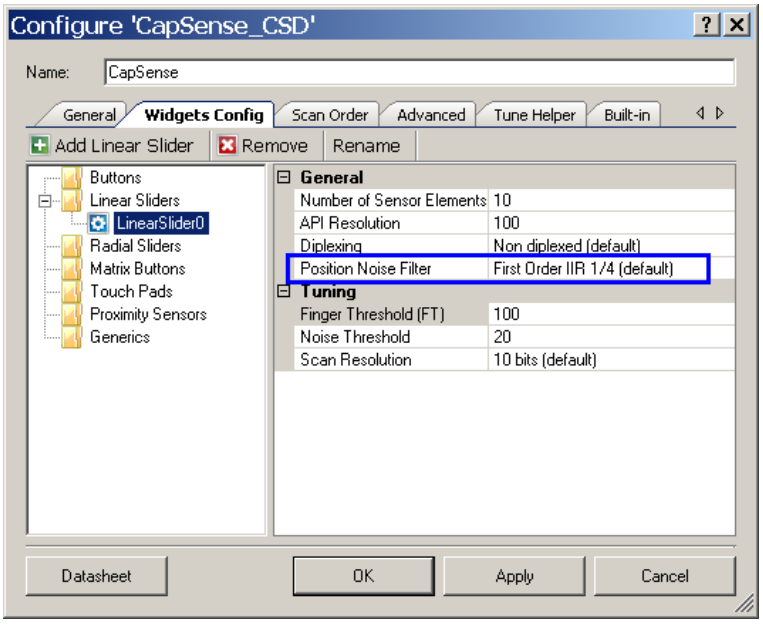
图 5-11. “原始数据噪声滤波器”参数



原始计数是 CapSense 系统的原始输出，并具有固有噪声。原始计数中的噪声可通过软件滤波器最小化。通过“原始数据噪声滤波器”参数您可以选择一些软件滤波器来过滤原始计数和手指位置数据。有关各个滤波器类型的信息，请参见软件滤波。

5.3.8.2 位置噪声滤波器

图 5-12. “位置噪声滤波器”参数



线性滑条和辐射滑条等 Widget 可输出手指位置。通过“位置噪声滤波器”参数您可以通过选择一些软件滤波器来过滤掉手指位置输出。

5.3.9 高层参数建议

为获得最佳参数设置，建议采用以下初始值

- **手指阈值**：传感器为 ON 时设置为 75%的原始计数
- **噪声阈值**：传感器为 OFF 时设置为 40%的原始计数
- **负噪声阈值**：设置为噪声阈值
- **迟滞**：传感器为 ON 时设置为 15%的原始计数
- **低基准线复位**：设置为 10
- **传感器自动复位**：根据设计要求
- **去抖动**：根据设计要求
- **Widget 分辨率**：如果正在使用滑条并且信号因薄覆盖层或较大的传感器区域而饱和，这时要将所有设计的 Widget 分辨率设为 8 位，否则选择 16 位。
- **滤波器选择**：请参见软件滤波，以选择合适的滤波器

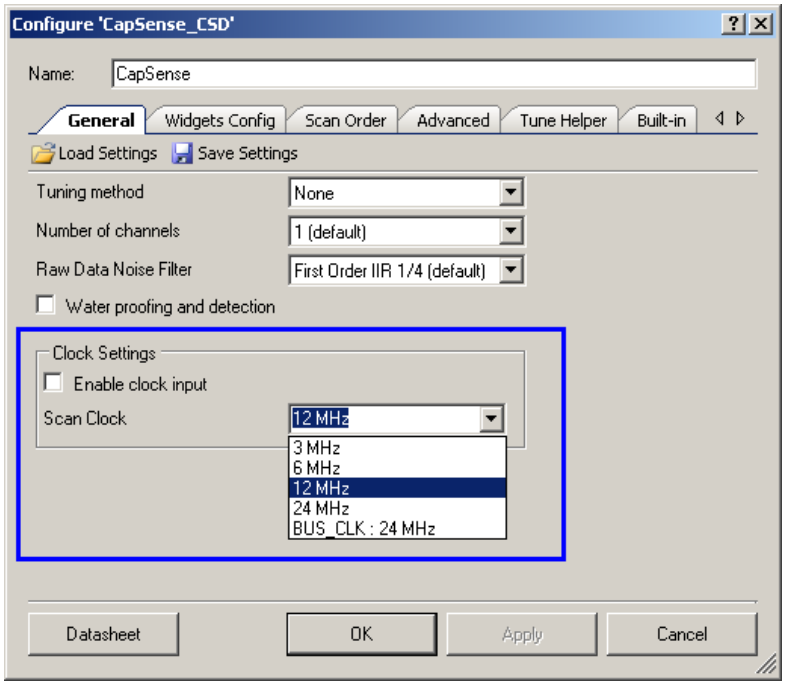
5.4 低层参数

低层参数在物理层定义 CapSense 系统的行为，并涉及从电容到原始计数的转换。

5.4.1 时钟设置

5.4.1.1 扫描时钟

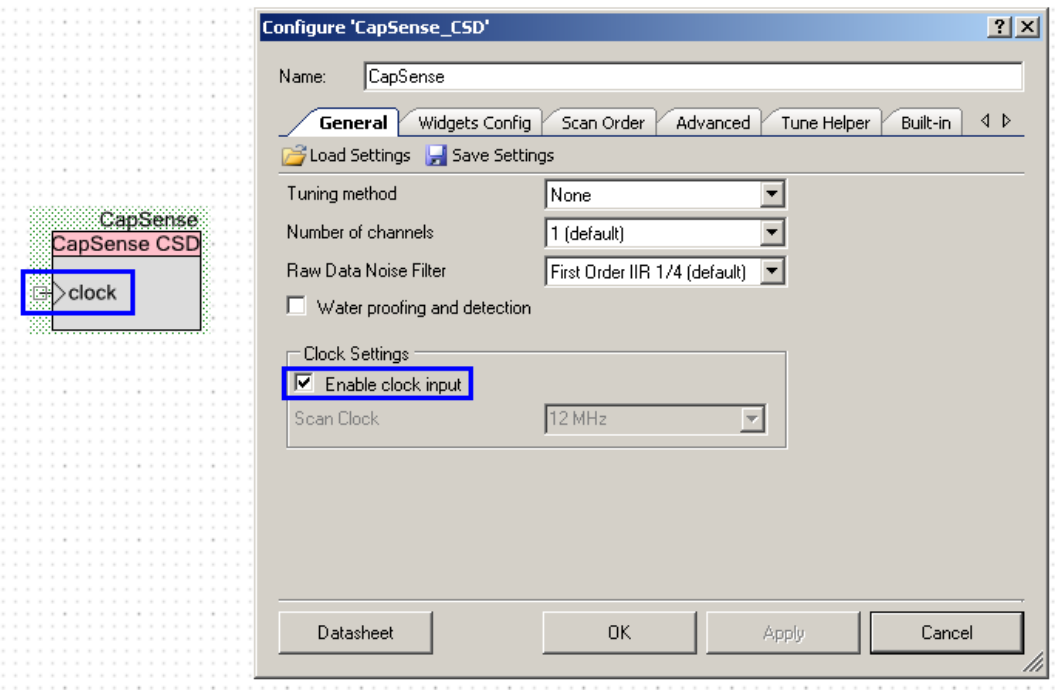
图 5-13. “扫描时钟”参数



扫描时钟参数用于为 CapSense_CSD 模块选择时钟源。“扫描时钟”与 CPU 频率无关。它由主时钟派生而来，因此主时钟频率应大于或等于“扫描时钟”参数设置的。

5.4.1.2 使能时钟输入

图 5-14. “外部时钟输入”参数

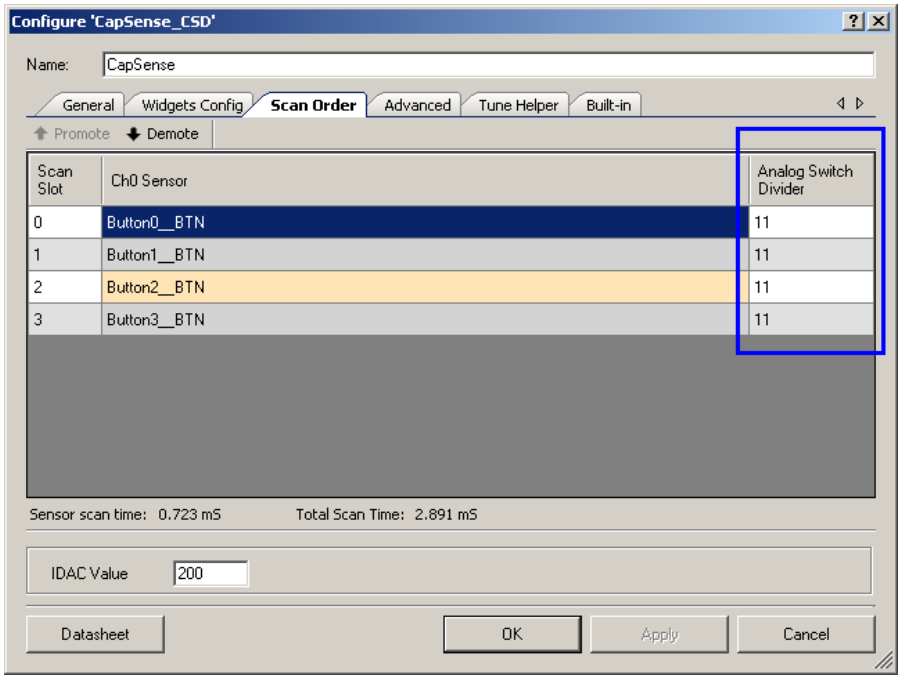


选择“Enable Clock Input”（使能时钟输入）参数表示时钟源来自 CapSense_CSD 模块的外部。使能该选项时，“Scan Clock”的频率被禁用，并在 CapSense_CSD 组件上出现一个终端，如图 5-14 所示。系统中的数字信号和通过 GPIO 路由的外部信号可被连接到该终端。

5.4.2 模拟开关分频器

5.4.2.1 模拟开关分频器（预分频器）

图 5-15. “模拟开关分频器”参数



“模拟开关分频器”参数可为 CapSense 传感器设置开关频率。当采用 IDAC 源电流模式时，CapSense 传感器将在 C_{MOD} 和 GND 之间连续切换。当采用 IDAC 灌电流和外部电阻模式时，它们将在 C_{MOD} 和 V_{DD} 之间连续切换。开关时钟频率被定义为：

$$Switching\ Clock = \frac{Scan\ Clock}{Analog\ Switch\ Divider}$$ 公式 5

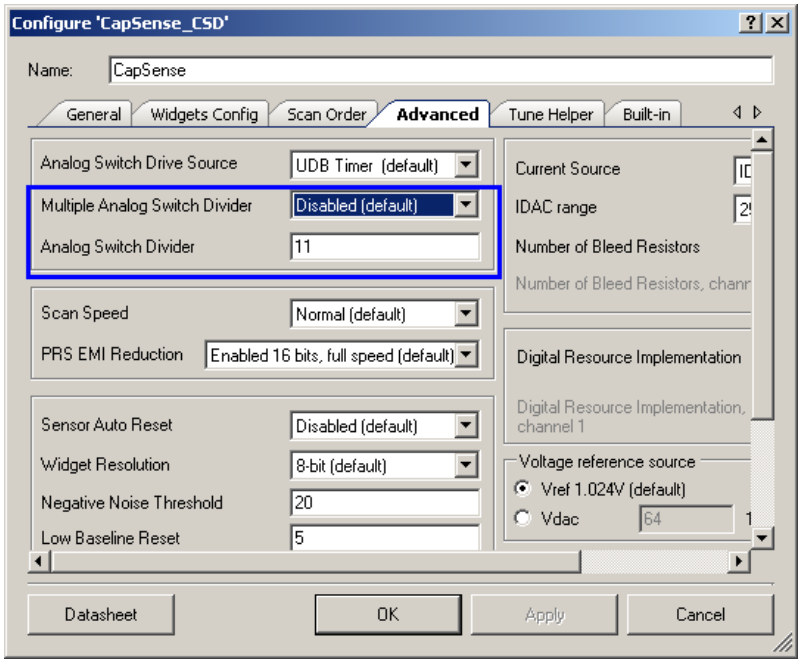
其中：

扫描时钟 = CapSense 模块的时钟源（如时钟设置所阐释）

开关时钟 = CapSense 传感器开关的频率

5.4.2.2 多模拟开关分频器

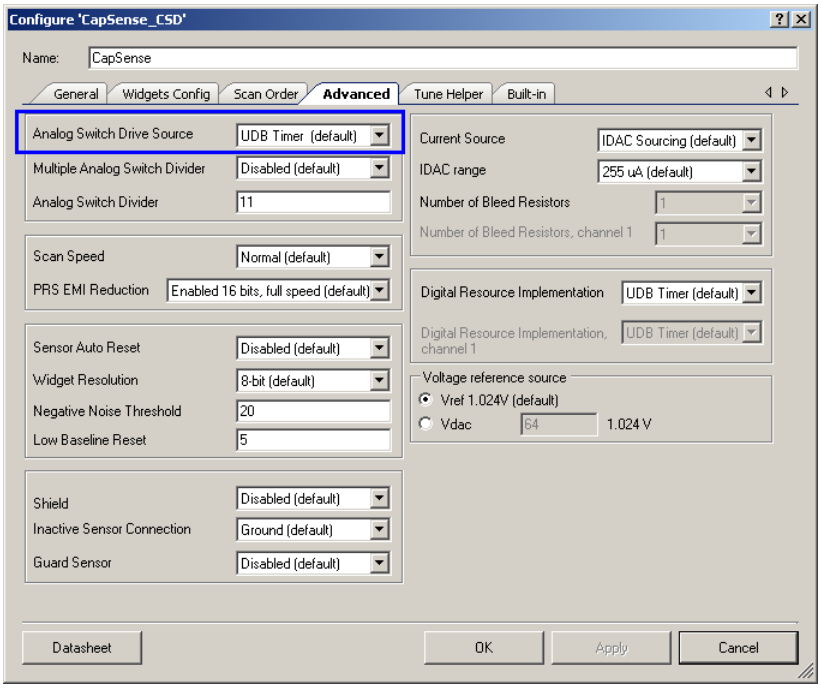
图 5-16. “多模拟开关分频器”参数



通过 CapSense_CSD 您可以为各个 CapSense 传感器设置不同的“模拟开关分频器”或为所有传感器设置一个相同的“模拟开关分频器”的选项。当所有传感器的 C_P 值相差不大时，应该使能一个相同的“模拟开关分频器”参数。如果传感器之间的 C_P 值差异较大，那么应该使能“多模拟开关分频器”参数。使能“多模拟开关分频器”参数后，可在 scan order（扫描顺序）选项卡中设置这些值，如图 5-14 所示。

5.4.2.3 模拟开关驱动源

图 5-17. “模拟开关驱动源”参数



该参数用于选择实现模拟开关分频器的方式。“模拟开关分频器”是一个基于定时器的分频器。可以选择下面三个选项中的一个作为该“定时器”所利用的资源：

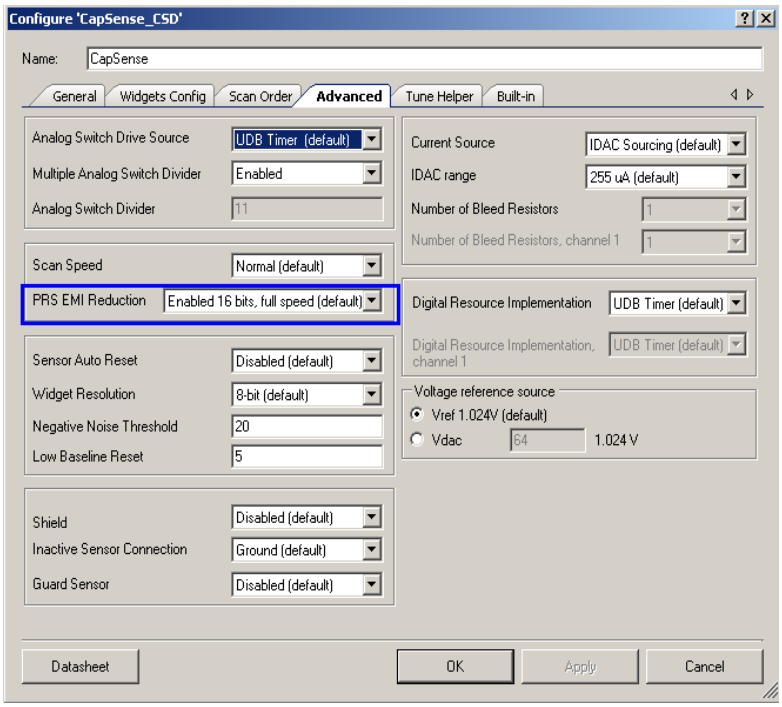
Direct — 选中该选项时，不使用任何“模拟开关分频器”，扫描时钟会自动成为开关时钟。对资源使用存在特殊要求时，需要使用该选项，但必须确保当前设计正确工作。

UDB Timer（UDB 定时器） — 选择该选项时，通过使用一个 UDB 实现模拟开关分频器的定时器。这是默认选项，因为具有多个 UDB 模块。

FF Timer（FF 定时器） — 选择该选项时，通过使用一个固定功能的数字模块实现模拟开关分频器的定时器。

5.4.3 伪随机序列（PRS）

图 5-18. PRS EMI Reduction（PRS 减少电磁干扰）参数



通过该参数您可以在开关时钟上使用伪随机序列（PRS）。PRS 按照其中心频率来改变开关时钟频率，从而扩展频谱，并降低 EMI。您可以选择以下各选项：

Disabled（禁用） — 该选项提供了最佳 SNR，因此如果应用不需要降低电磁干扰，那么建议选择该选项。

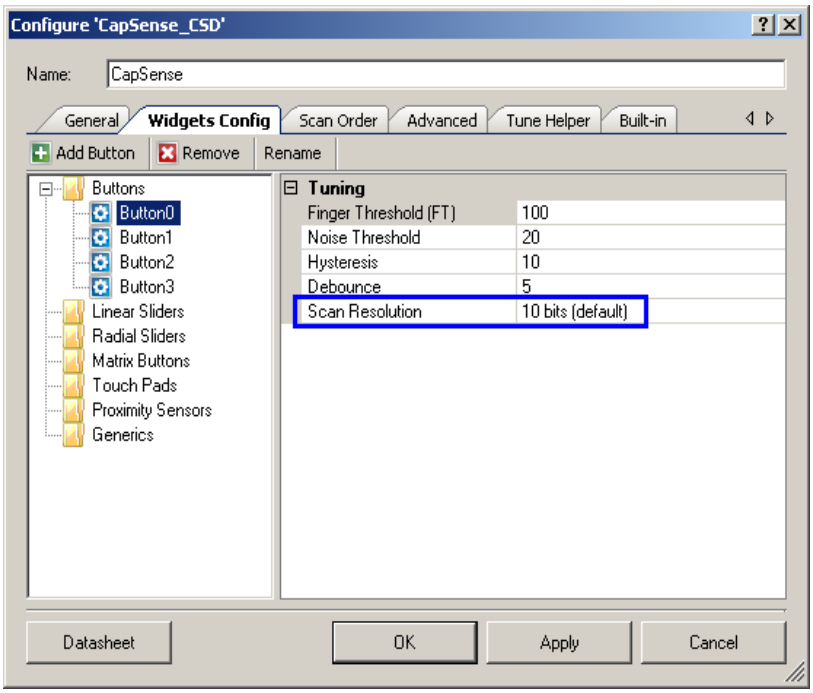
Enabled 8 bits（使能 8 位） — 与 16 位相比，8 位可以提供更好的信噪比，但由于它的重复周期比较短，因此会增加 EMI。

Enabled 16 bits, full speed (default)（使能 16 位，全速（默认）） — 16 位可以提供更低的信噪比，并且可以使减少电磁干扰的效果更佳。想要减少 EMI，建议使用该选项。

Enabled 16 bits, 1/4 speed（使能 16 位，1/4 速度） — 与使能 16 位，全速相比，要求一个更快的 4 倍时钟频率，这样才能获得相同的 PRS 时钟输出。

5.4.4 Scan Resolution（扫描分辨率）

图 5-19. Scan Resolution（扫描分辨率）参数

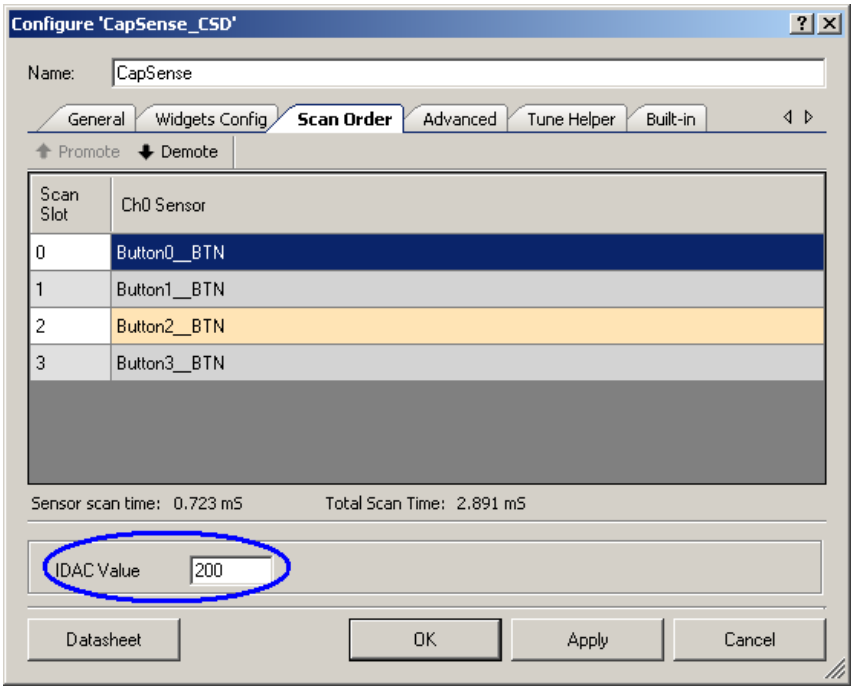


该参数确定扫描分辨率（单位为位）。N 位的扫描分辨率的最大原始计数值为 $2^N–1$。增大分辨率可提高灵敏度，但是也可以增大扫描传感器所需的时间。取值范围为 8 - 16 位。

5.4.5 IDAC 电流

5.4.5.1 IDAC 值

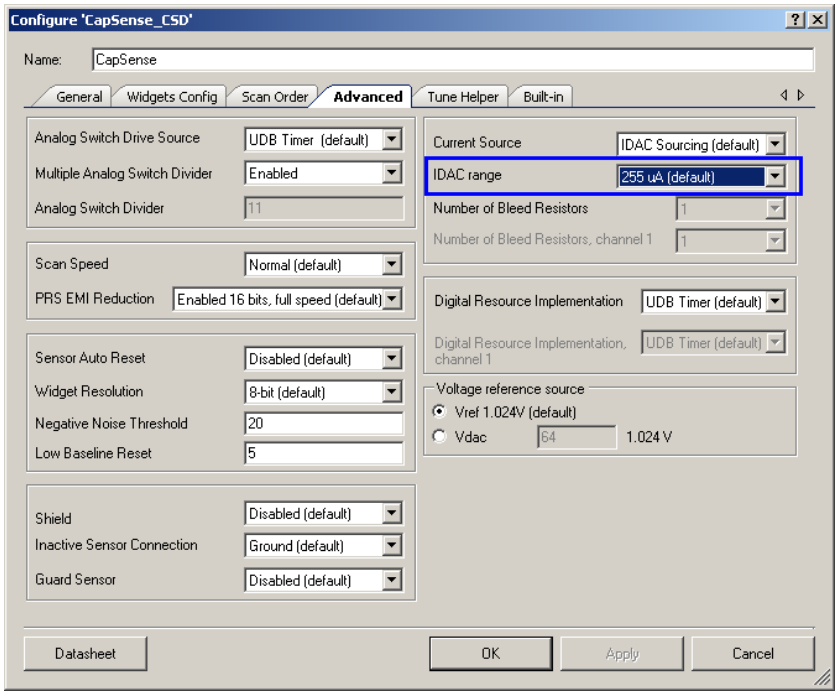
图 5-20. IDAC Value（IDAC 值）参数



通过该参数设置 iDAC 电流。当 IDAC 电流减小时，原始计数会随着灵敏度逐渐增大。N 位的扫描分辨率的最大原始计数值为 2^N–1。将 IDAC 电流应设置为：原始计数介于最大值的 50%和 80%之间。

5.4.5.2 IDAC 范围

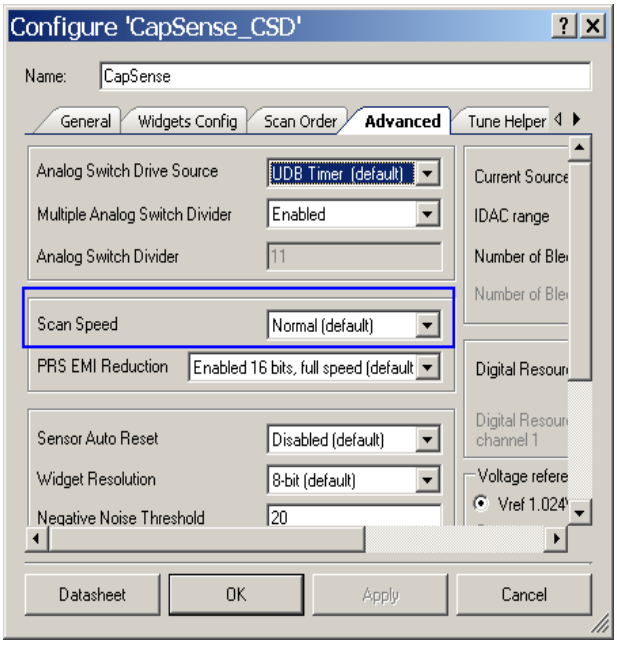
图 5-21. IDAC Range（IDAC 范围）参数



该参数用于选择 IDAC 电流的范围。所需 IDAC 电流会根据 C_P 直接变化。对于较大的 C_P 值，应选择较大的范围，反之亦然。可选值为 32 µA、255 µA 以及 2048 µA。

5.4.6 扫描速度

图 5-22. Scan Speed（扫描速度）参数



该参数用来设置传感器扫描速度。更快的扫描速度可以提供更好的响应时间。扫描速度越低，信噪比越高，抗电源和温度变化能力也越好。可能值为 Very Fast（非常快）、Fast（快）、Normal（正常）和 Slow（慢）。

扫描速度时钟通过下面的公式来设置计数时钟：

$$Counting\ Clock = \frac{Scan\ Clock}{Scan\ Speed\ Divider}$$ 公式 6

其中：

扫描时钟 = CapSense_CSD 模块的时钟源

扫描速度分频器 = 1（非常快）、2（快）、3（正常）、4（慢）

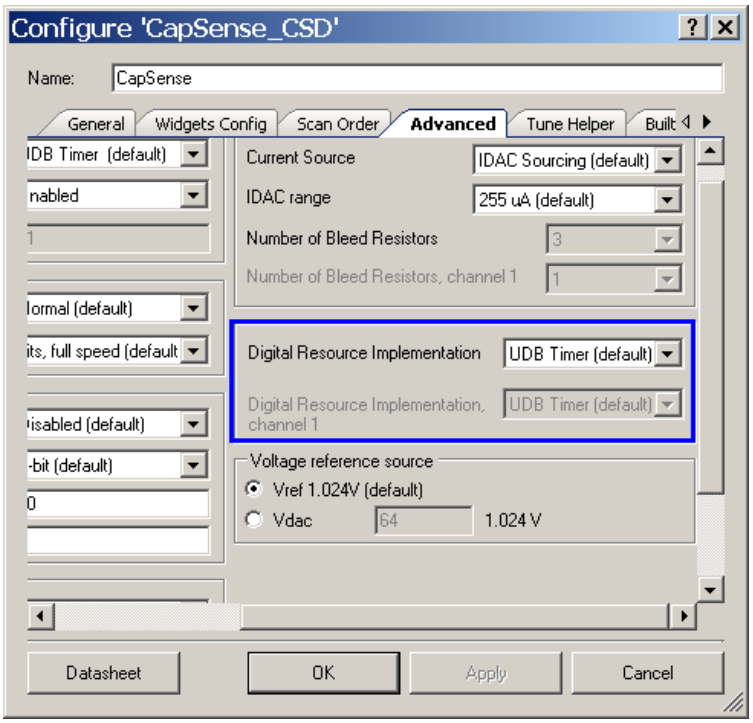
扫描时间取决于扫描速度、扫描分辨率、扫描时钟和 CPU 时钟。表 5-2 显示的是在扫描时钟频率为 24 MHz、CPU 时钟频率为 48 MHz 时单个传感器的扫描时间。

表 5-2. 单传感器的扫描时间（单位为 µs）、扫描速度和扫描分辨率。

分辨率（位）	扫描速度			
	非常快	快速	正常速度	慢速
8	58	80	122	208
9	80	122	208	377
10	122	208	377	718
11	208	377	718	1400
12	377	718	1400	2770
13	718	1400	2770	5500
14	1400	2770	5500	10950
15	2770	5500	10950	21880
16	5500	10950	21880	43720

5.4.7 数字资源实现

图 5-23. Digital Resource Implementation（数字资源实现）参数



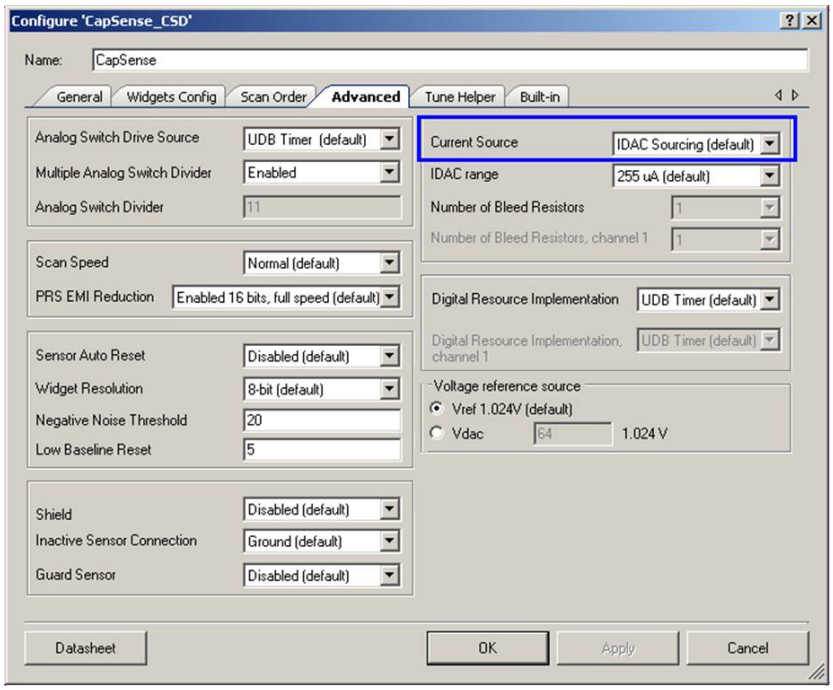
“扫描速度”设置使用一个基于定时器的分频器来创建计数时钟。该参数允许您选择实现该分频器所需的资源。针对各个通道，可以单独选择“数字资源实现”参数。其选项为：

UDB Timer（UDB 定时器） — 选择该选项时，定时器通过一个 UDB 实现。这是默认选项，因为存在多个 UDB 模块。

FF Timer（FF 定时器） — 选中该选项后，可使用固定功能数字模块来实现定时器。

5.4.8 电流源

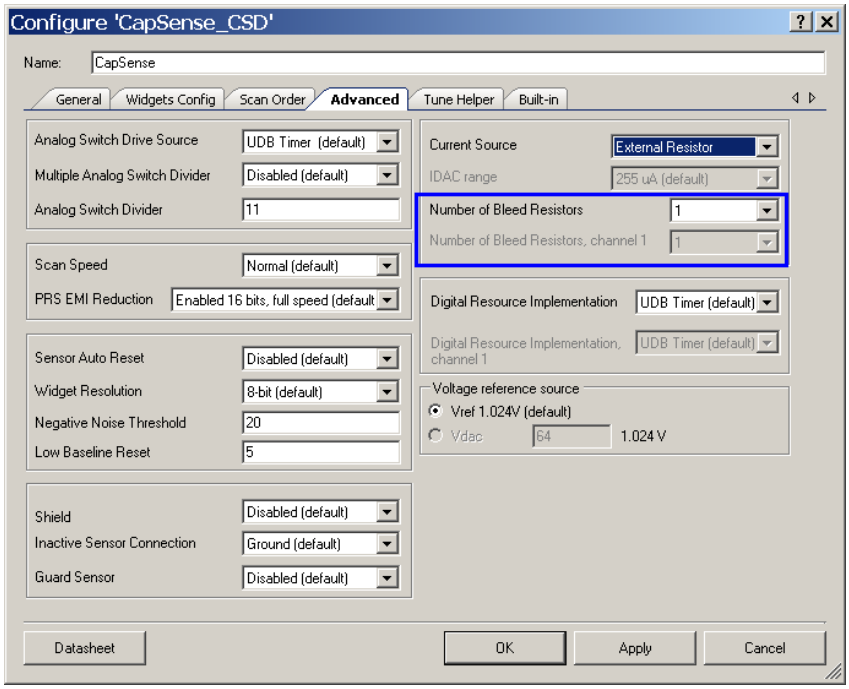
图 5-24. Current Source（电流源）参数



该参数用于选择电流源方法。可能的值分别为 IDAC Sourcing（IDAC 源电流）、IDAC Sinking（IDAC 灌电流）以及 External Resistor（外部电阻）。有关这些电流源方法的详细信息，请参见电流源方法一节。

5.4.8.1 泄流电阻的数量

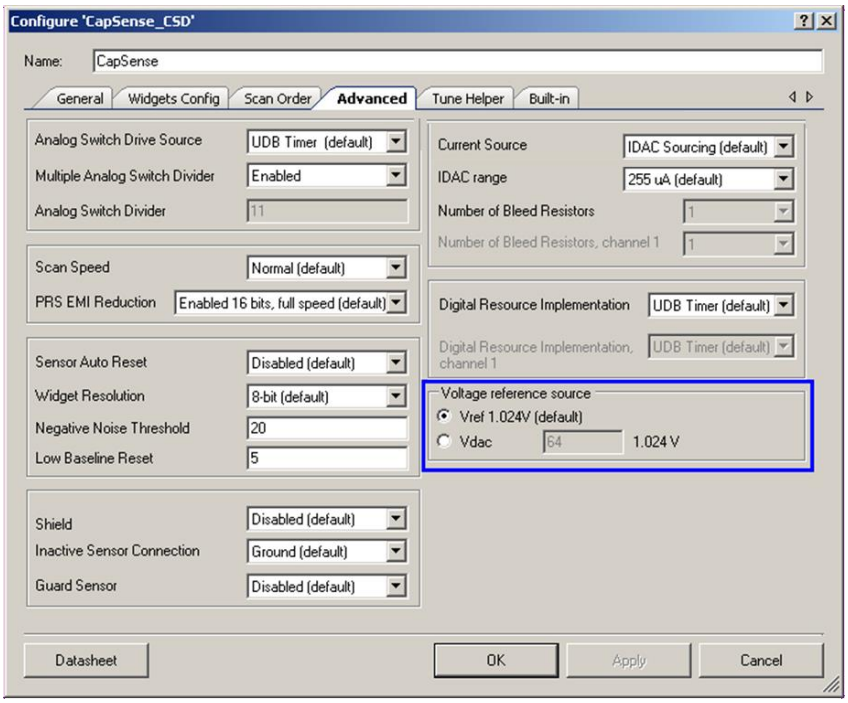
图 5-25. Number of Bleed Resistors（泄流电阻的数量）参数



外部电阻方法要求使用 PSoC 器件外的泄流电阻。该参数允许您指定泄流电阻的数量。可连接多达三个泄流电阻。对于双通道电容式感应，每个通道需要一套单独的泄流电阻。

5.4.9 电压参考源

图 5-26. Voltage Reference Source（电压参考源）参数



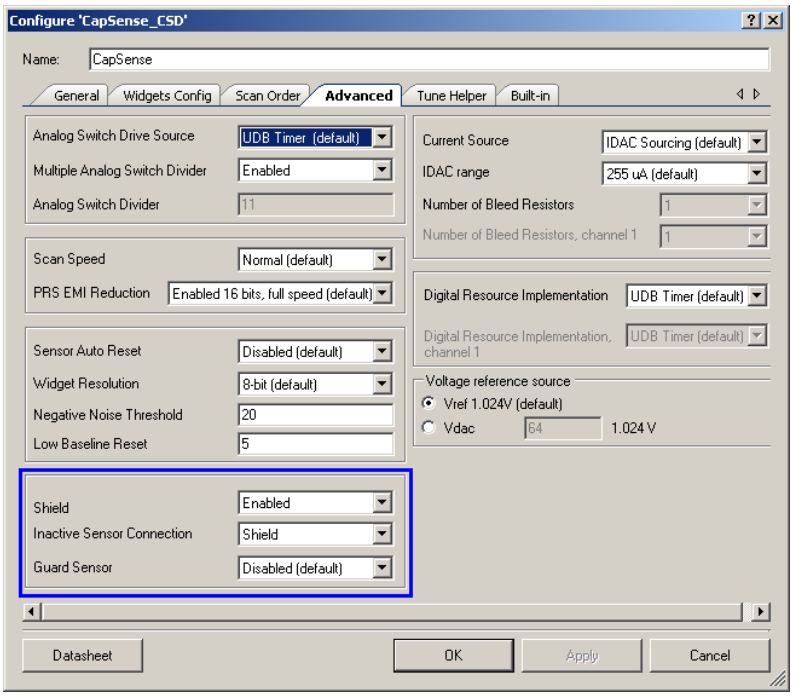
通过该参数，您可以指定电压参考源。其选项为：

Vref 1.024 V — 它是带隙电压

Vdac — 通过该选项，您能够选择 0 到 4 V 的电压。当使用 IDAC 源电流方法时，该选项很有用。增大参考电压，您可以增大传感器上摆动的电压。这样可提供更佳的抗手指传导噪声能力。这种设置会占用一个 DAC 资源。

5.4.10 屏蔽电极和保护传感器

图 5-27. “屏蔽电极和保护传感器”参数



5.4.10.1 屏蔽电极

如果您的设计使用了屏蔽电极，那么应使能该参数。请参考防水设计一节。

5.4.10.2 Inactive Sensor Connection（非活动状态传感器连接）

传感器未被扫描时，通过该参数可指定传感器的连接情况。可能的值为：

GND — 此处为推荐的设置内容，因为它会产生较小的噪声。

Shield — 如果您的设计使用了屏蔽电极，并包括具有相邻传感器的 Widget（如滑条），那么应该选择该设置。将非活动传感器连接到屏蔽电极可防止存在水时所引起的误触发。

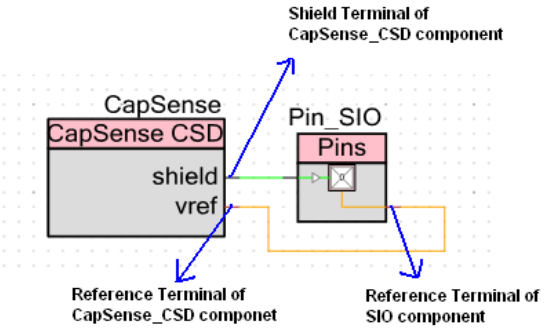
HI-Z — 使非活动状态传感器处于高阻抗模式。使用这种设置来降低传感器接地时所造成的总寄生电容。

5.4.10.3 保护传感器

如果您的设计使用保护传感器，应使能该参数。请参考防水设计一节。

5.4.10.4 作为屏蔽电极使用的 SIO 引脚

图 5-28. 作为屏蔽电极使用的 SIO 引脚

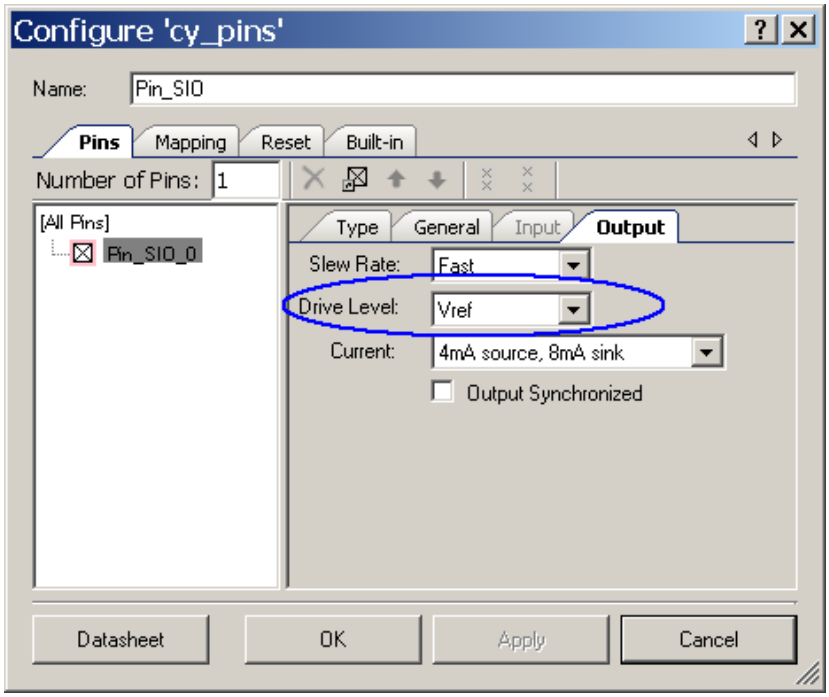


启用屏蔽电极后，组件上将出现一个屏蔽终端，如图 5-28 所示。

应将一个 I/O 引脚连接至该终端。将 IDAC 灌电流或使用外部电阻方法作为电流源时，应该使用一个普通的 GPIO 引脚。但是，如果正在使用 IDAC 源电流方法，应该将屏蔽电极连接到 SIO 引脚上。通过使用 SIO 引脚，您可以使屏蔽信号同传感器信号相互匹配。

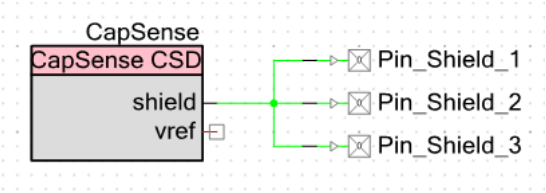
要将 SIO 引脚用作屏蔽电极，可从组件目录中拖入一个数字输出引脚。打开配置窗口，然后将 Drive Level（驱动电平）选项设为“Vref”，如下所示。

图 5-29. 配置“Drive Level”（驱动电平）以将组件引脚设为 SIO



5.4.10.5 多屏蔽电极引脚

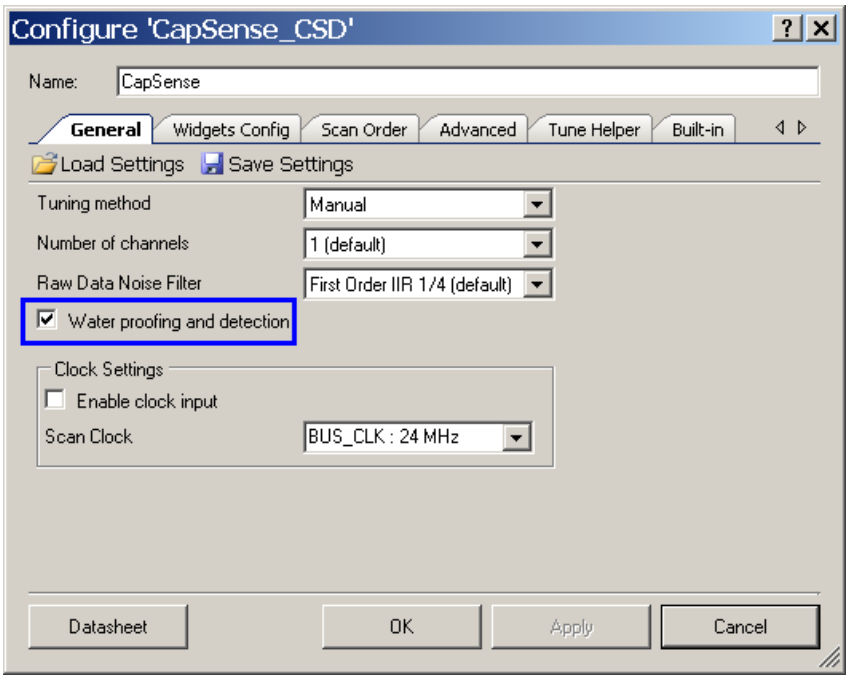
图 5-30. 多屏蔽电极引脚



如果屏蔽面积较大，则单个屏蔽电极引脚可能无法驱动屏蔽电极。您需要验证屏蔽电极是否满冲满放。为检查这一点，可通过探针探查屏蔽引脚并在示波器对其进行监控。由于普通探针会显著增大引脚电容，因此，应将探针放置于 10x 模式下。推荐使用 FET 探针。如果屏蔽电极并非满充满放，则请使用多个 I/O 引脚来驱动屏蔽电极。I/O 引脚（包括 GPIO 和 SIO）的驱动电流为 4 mA。

5.4.10.6 防水和检测

图 5-31. 防水和检测参数



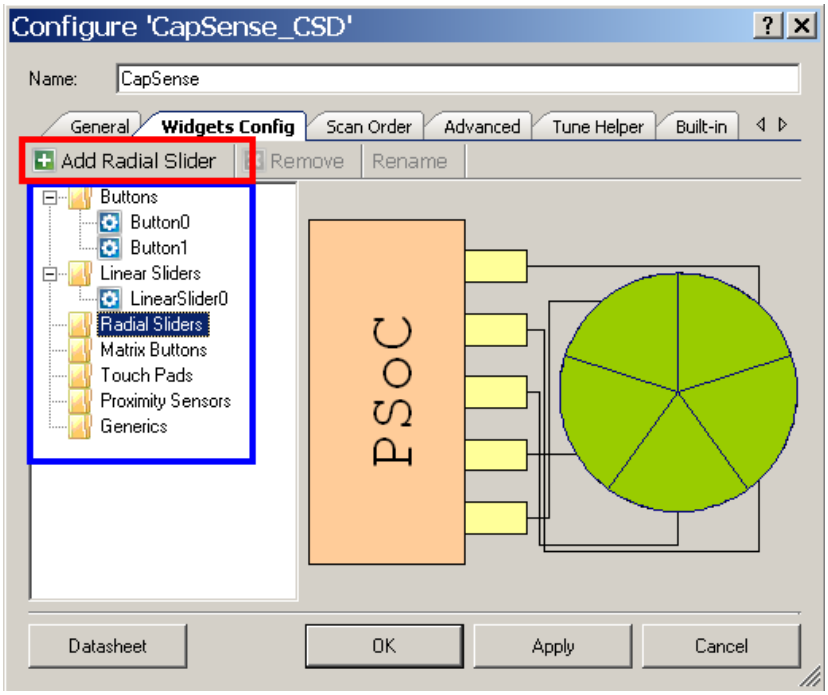
驱动屏蔽电极和保护传感器的另一方法是：在 General（通用）选项卡上选择“Water proofing and Detection”（防水和检测）参数。勾选该框可保证在 Advanced（高级）选项卡上都启用了屏蔽电极和保护传感器。

5.5 Widget 配置

通过“Widgets Config”选项卡，您可以添加并配置 widget。

5.5.1 添加 Widget

图 5-32. 添加 Widget



要添加一个 Widget，请选择 Widget 类型，然后点击“Add”（添加）项。

5.5.2 传感器元素的数量

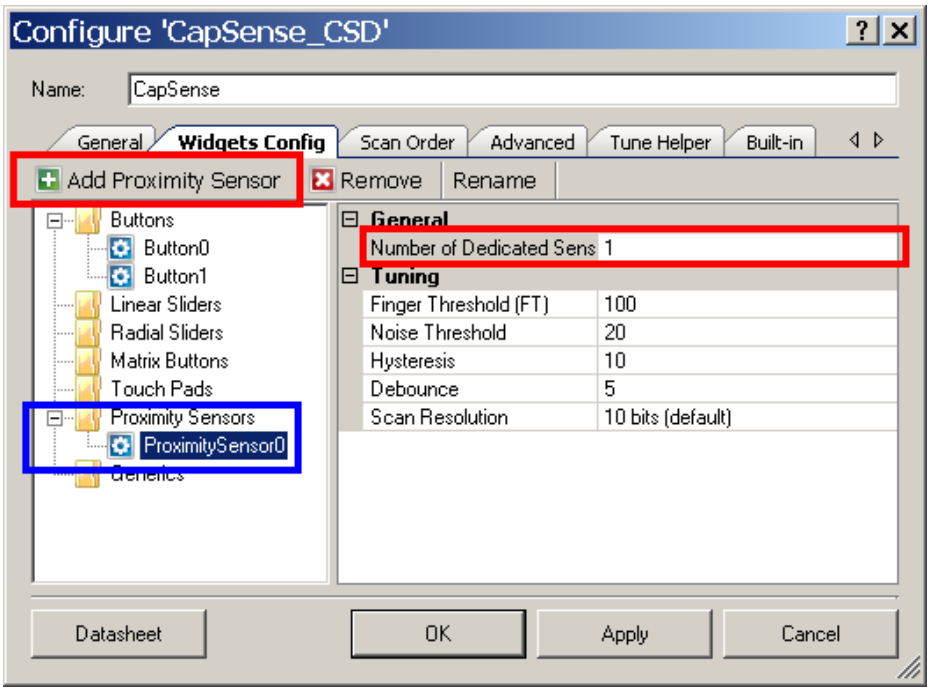
在该选项卡中，设置每个 widget 的传感器数量。

5.5.3 API 分辨率

某些 widget 类型（如滑条和触控板）使用的是手指位置数据而不是传感器的 ON/OFF（开/关）状态。手指位置数据的分辨率被称为 API 分辨率。

5.5.4 接近感应传感器

图 5-33. 接近感应传感器参数



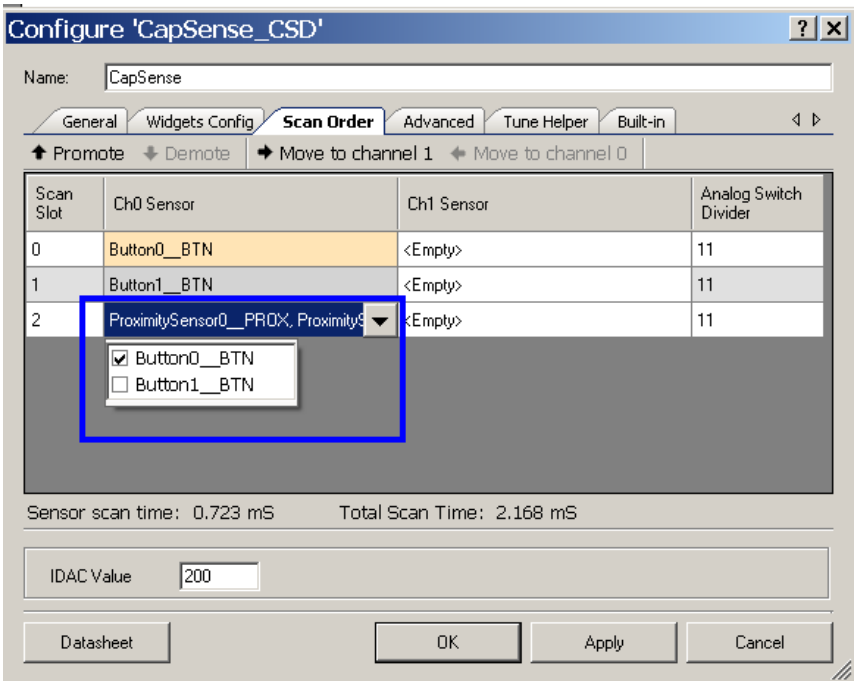
接近感应传感器用于检测传感器附近是否存在人手，无需实际接触它。接近感应传感器可以由一条 PCB 上用户接口绕着的长走线构成。另外，可以短路各个传感器，然后将它们作为单个传感器进行扫描。您也可以将这两种方法结合起来，并将用户界面周围的接近感应传感器与其他传感器短接，然后将其作为单个传感器进行扫描。

注意：在 PSoC Creator 中，接近感应传感器默认为禁用，而其他所有传感器都默认为启用。进行扫描前，必须明确启用接近感应传感器，如以下代码段所示。

```
17
18      CyGlobalIntEnable;  /* Uncomment this line to enable global interrupts. */
19      CapSense_Start();
20      CapSense_EnableWidget(CapSense_PROXIMITYSENSOR0__PROX); /*Enabling proximity sensor explicitly*/
21      CapSense_InitializeAllBaselines();
22
23      for(;;)
24      {
25          if ( CapSense_IsBusy()==0 )
26          {
27              CapSense_UpdateEnabledBaselines();
28              CapSense_CheckIsAnyWidgetActive();
29              CapSense_ScanEnabledWidgets();
30          }
31      }
32  }
```

5.5.4.1 专用传感器元素的数量

图 5-34. 现有传感器和接近传感器的组合

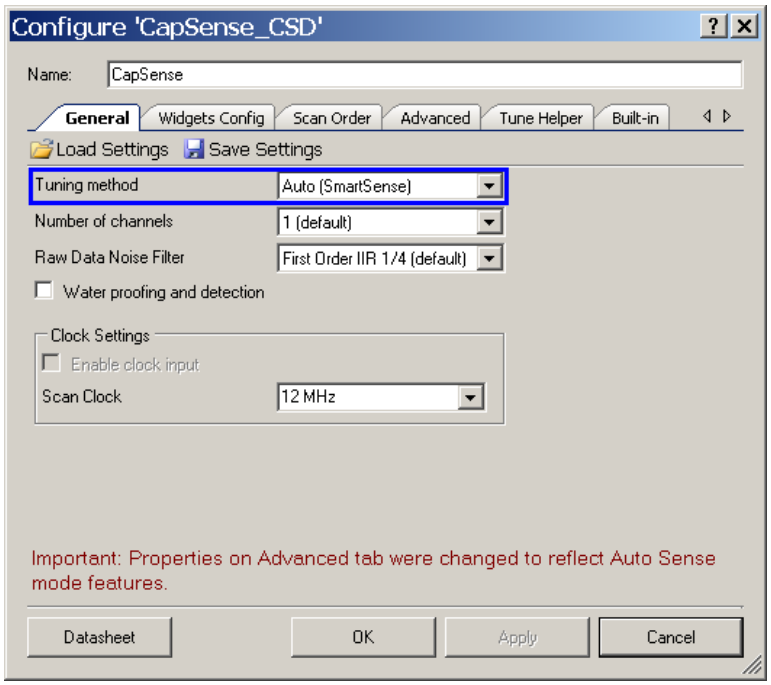


要将现有的传感器与一个专用的接近感应传感器组合起来，请设置 Widgets Config 选项卡上的‘Number of Dedicated Sens’（专用传感器的数量）项，如图 5-33 所示。然后通过使用 Scan Order（扫描顺序）选项卡上的下拉菜单选择需要与接近感应传感器组合的现有传感器。

将它们结合起来，不会影响它们的运行情况，但是接近传感器的 C_P 值会变大，并会使扫描时间变长。因此，应以低于其他传感器的扫描速率扫描接近感应传感器，以避免出现较长的扫描间隔。

5.6 调校方法

图 5-35. 调校方法参数



通过该参数，您可以选择调校 CapSense 系统的方式。对于所有调校方法中，调谐器 GUI 可用于监控 CapSense 信号。可能值为：

Auto（自动） — SmartSense 自动调校算法将所有参数设为最优值。选择‘Auto’（自动调校）时，下面各参数不发生改变：

表 5-3. 选择‘Auto’时各固定参数

参数	设置
电流源	IDAC 源电流/IDAC 灌电流 不支持外部电阻
IDAC 范围	255 µA
多模拟开关分频器	使能
扫描速度	正常
PRS	启用 16 位，全速

Manual（手动调校）— 您必须手动调校所有参数。

None（无）— 各参数均为固定值并存储在闪存中。必须已经使用手动调校或自动调校方法完成调校各个参数。

5.6.1 灵敏度

灵敏度参数用于设置 SmartSense 自动调校算法的 C_F 值。将灵敏度值乘以 0.1 pF 便能得到 C_F。灵敏度的值为 1 表示 C_F = 0.1 pF，灵敏度的值为 4 表示 C_F = 0.4 pF。

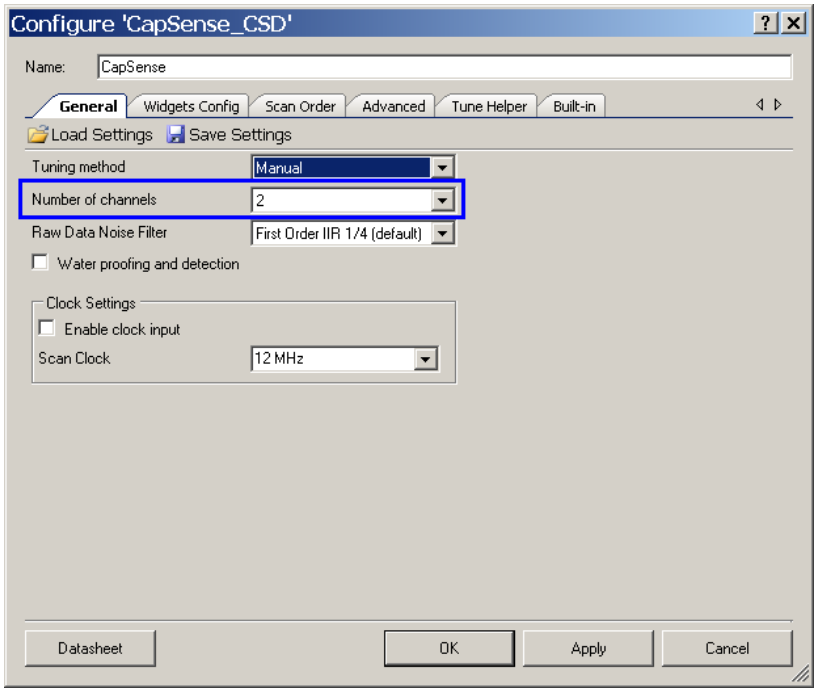
SmartSense 自动调校方法通过使用灵敏度参数来计算表 5-4.中列出的各个参数。

表 5-4. 自动调校方法使用灵敏度设置的参数

参数	何时计算
模拟开关分频器	在 CapSense_CSD 启动时计算一次
IDAC 值	在 CapSense_CSD 启动时计算一次
扫描分辨率	在 CapSense_CSD 启动时计算一次
手指阈值	在传感器扫描期间连续计算
噪声阈值	在传感器扫描期间连续计算
迟滞	在传感器扫描期间连续计算

5.7 通道数量

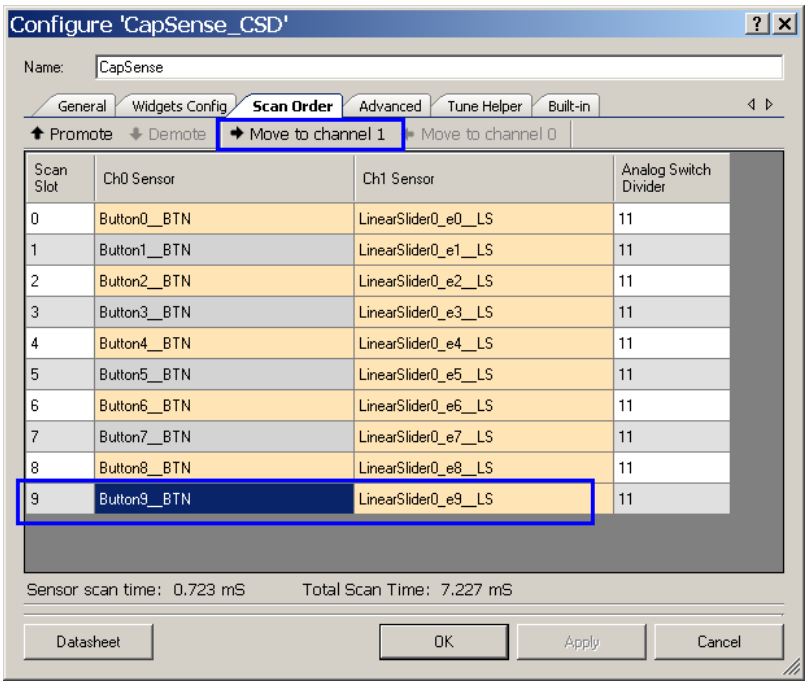
图 5-36. 通道数量



通过该参数，可以实现一个双通道设计。

5.7.1 移到通道 1/通道 0

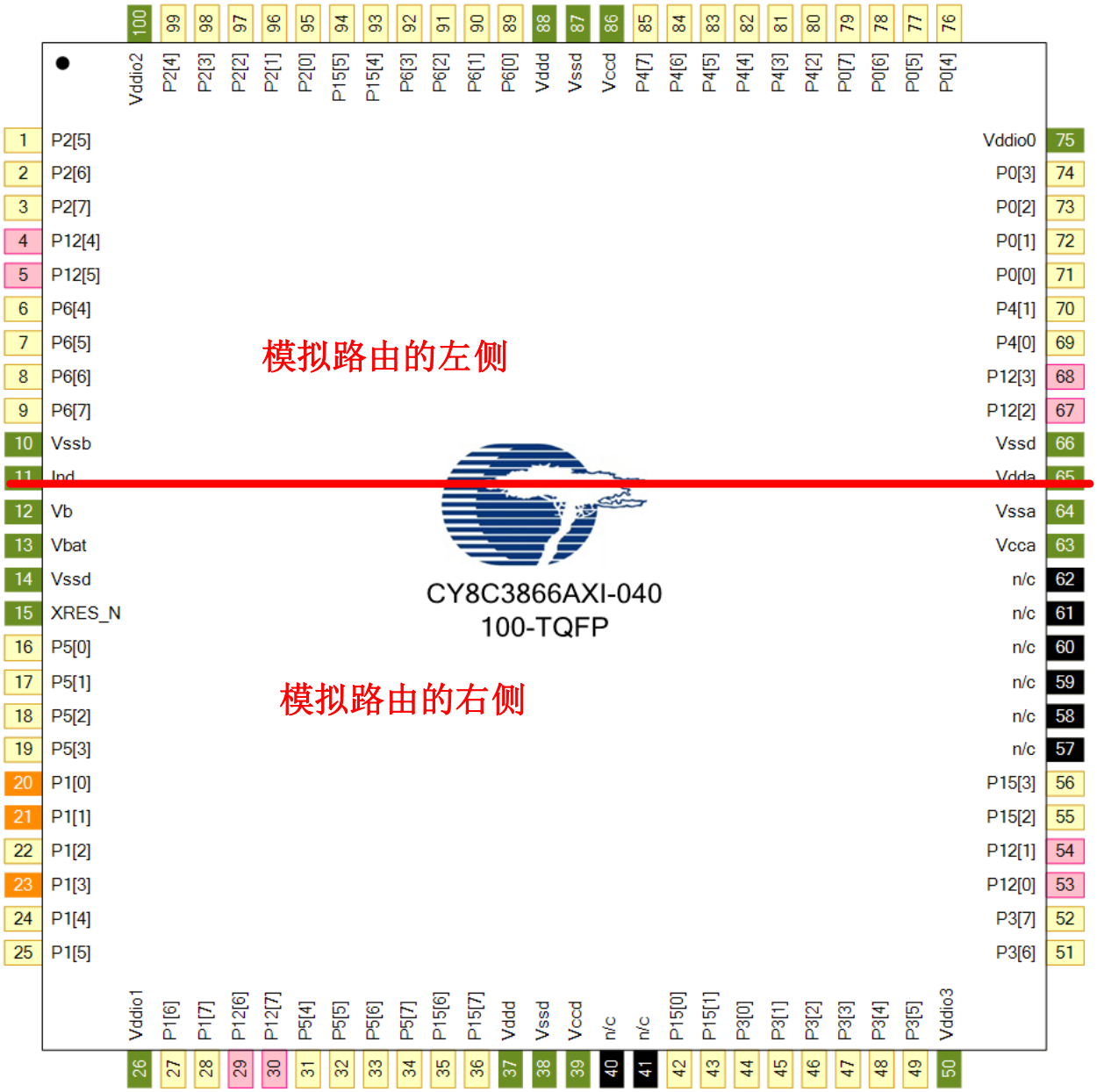
图 5-37. 设置传感器的通道分配



通过该参数，您可以指定每个传感器的通道分配情况。在 Scan Order（扫描顺序）选项卡上，请选择所需传感器，并点击 Move to Channel 1（移到通道 1）或 Move to Channel 0（移到通道 0）。

5.7.2 双通道设计的引脚分配

图 5-38. 双通道设计的引脚分配



PSoC 的每个 GPIO 引脚都是模拟走线左侧或右侧的一部分，具体情况取决于它与内部路由总线的连接情况。请参考器件数据手册中“模拟走线”一节，了解详细信息。图 5-38 是 PSoC 的引脚框图，它显示了 GPIO 和模拟走线中相应的侧面。

在双通道设计中，分配给同一通道的所有传感器都置于走线的同一侧，并且分配给其他通道的所有传感器置于走线的另一侧。如果在您的设计中仅使用了一个通道，那么 C_{MOD} 和泄流电阻引脚与所有传感器都要位于走线的同一侧。

5.8 调谐器 GUI

嵌入式 GUI 是 CapSense_CSD 的一项有用功能，可用于监控 CapSense 信号。调谐器 GUI 适用于自动调校和手动调校。CapSense 功能调校一节中阐释了调校过程。调谐器 GUI 通过 MiniProg3 与 PC 的 USB 相连接，并通过使用 I²C 连接到器件上。

图 5-39. MiniProg3 作为 I²C 至 USB 桥接器

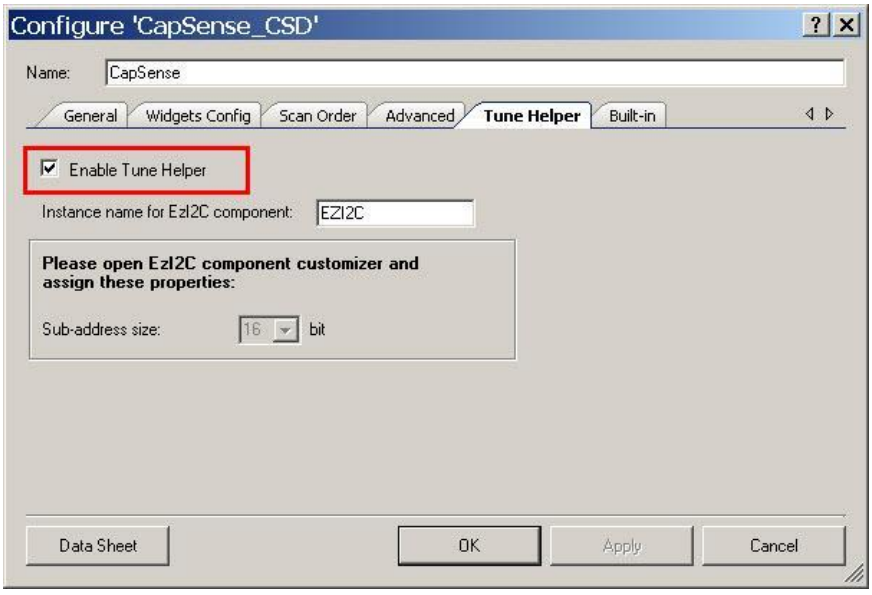


5.8.1 设置

注意： 请先完成第 1 到第 5 步来构建 PSoC Creator 项目。

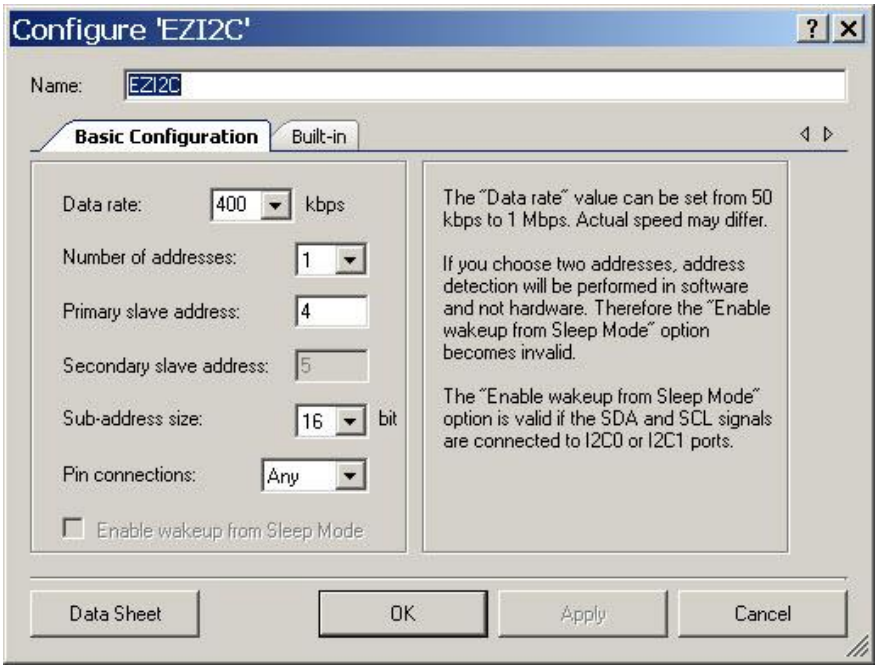
1. 将调校方法（调校方法）项设置为 Auto（自动调校）或 Manual（手动调校）。如果该项被设置为‘None’（无），那么调谐器 GUI 不会工作。
2. 添加 Widget。
3. 请勾选 Tune Helper（调校助手）选项卡的‘Enable Tuner Helper’（使能调校助手）项。

图 5-40. 启用调谐器



4. 放置并配置 EZI2C 组件

图 5-41. 重新命名 EZI2C 组件



5. 将下列代码段添加到您的项目中。

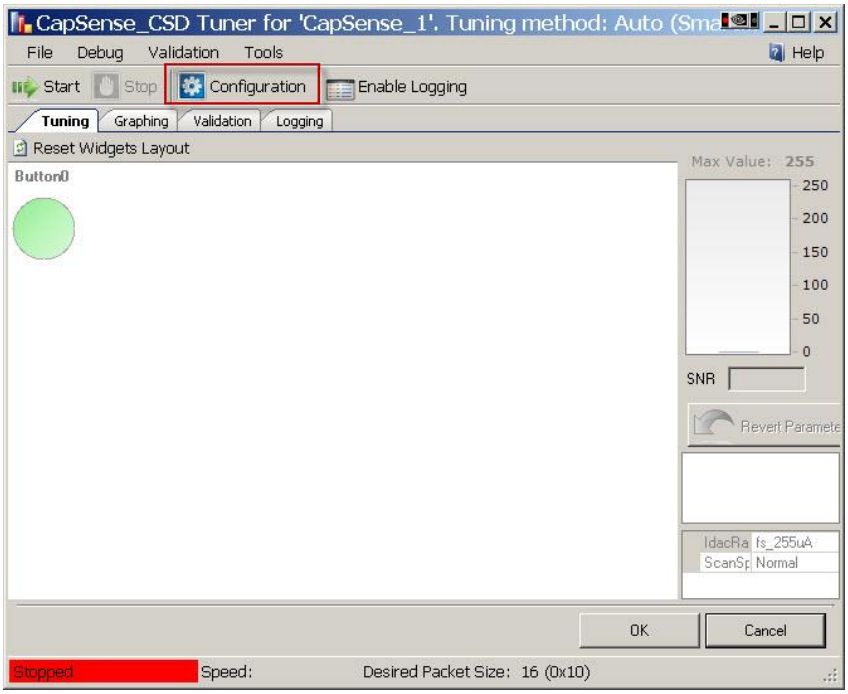
```
    #include <device.h>
void main()
{
CyGlobalIntEnable; /* Global Interrupt enable */

CapSense_TunerStart(); /* CapSense Block and Tuner Communication power up and
Intialization */

While(1)
{
CapSense_TunerComm(); /*Scan all the sensors and send the result to PC */
}
}
```

注意：在编程好器件并正确连接到 MiniProg3 后，请完成下列步骤。

1. 右键单击 CapSense_CSD 组件，然后点击 Launch Tuner（启动调谐器）。

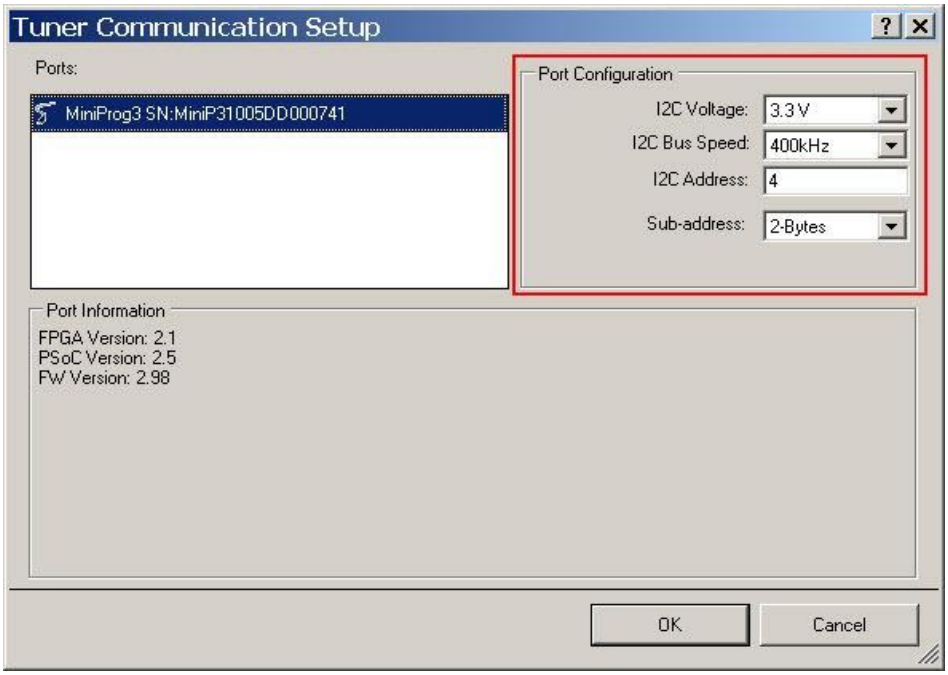
2. 单击 Tuner（调谐器）窗口中的“Configuration”（配置）选项卡。

图 5-42. 调谐器 GUI



3. 在 Tuner Communication Setup（调谐器通信设置）窗口内点击 MiniProg3，然后设置各个参数，如下所示：

图 5-43. 调谐器通信设置

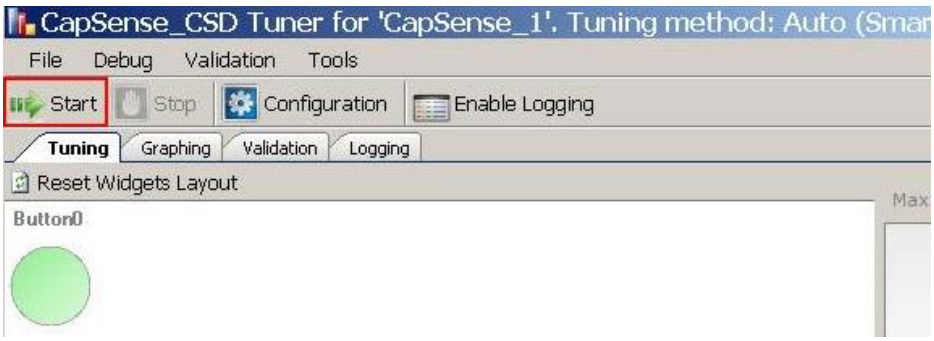


注意：MiniProg3 的各项电压设置必须同电路板上的电压设置相匹配。

4. 点击 OK 后，将关闭调谐器通信设置窗口。

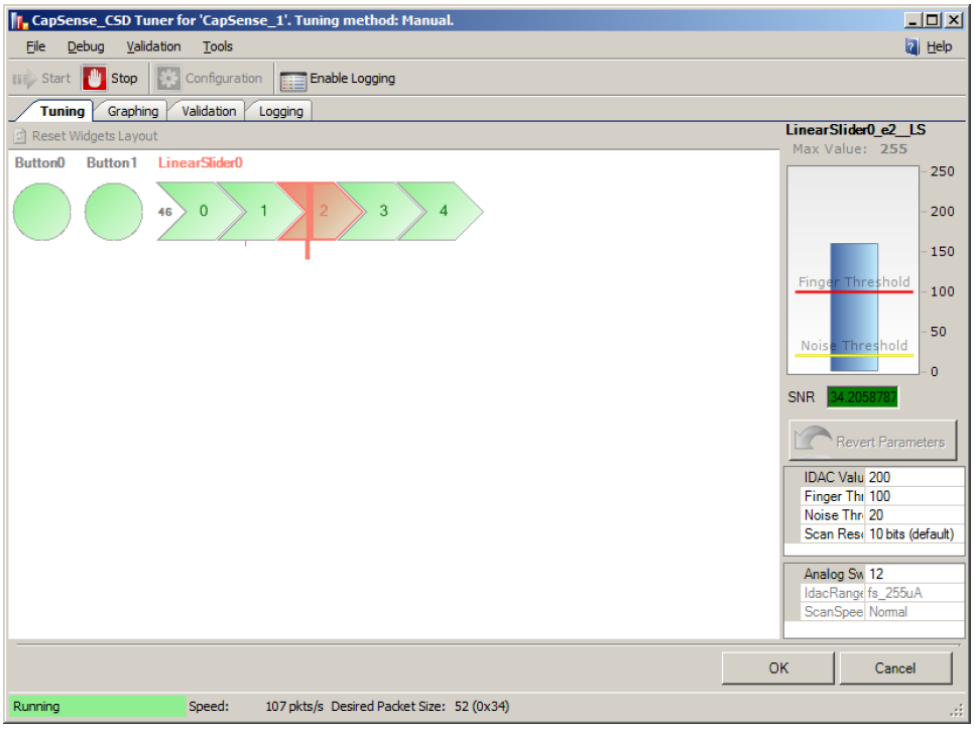
5. 在 Tuner（调谐器）窗口中，请点击 Start 项，以启动 I²C 通信并开始监控 CapSense 的各参数。

图 5-44. 启动调谐器通信



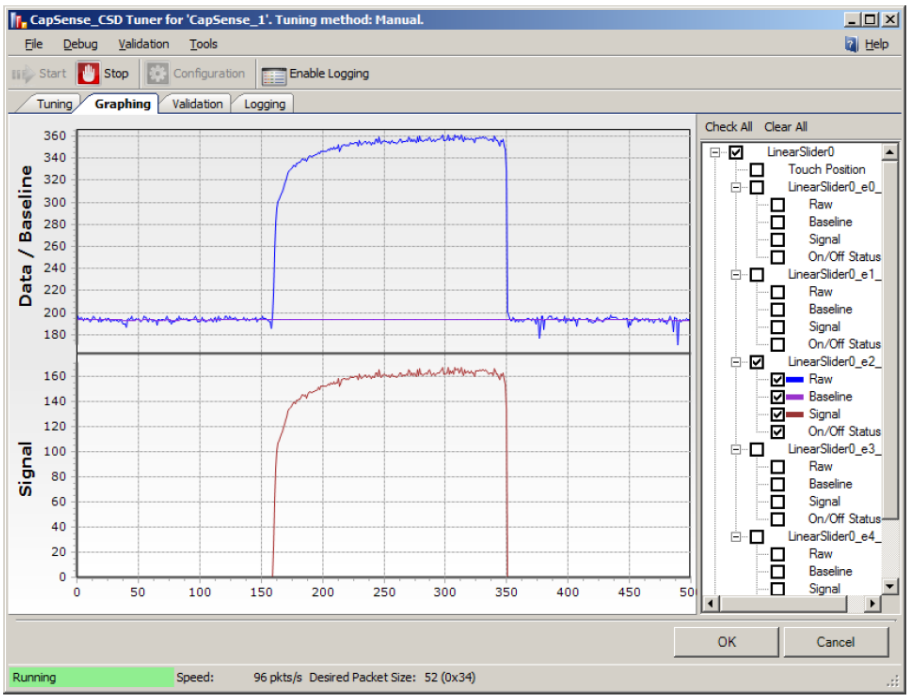
6. 设置每个传感器的参数调校完成后，请点击调谐器窗口右下角的“OK”即可设置组件中的参数。

图 5-45. 设置传感器的参数



7. 通过使用 Graphing（图形）选项卡，您可以监视各信号。您可以使用“Logging”（记录）选项卡记录各数据。

图 5-46. 监控 CapSense 结果



注意：Graphing 选项卡中显示的原始计数信号是未过滤的原始计数值。如果使用滤波器来过滤原始计数值，那么请通过使用已过滤的原始计数值来计算信噪比 SNR。请参考原始数据噪声滤波器一节。

6. CapSense 功能调校

调校是一个将 CapSense 参数达到最优值的过程。调校用于维持较高的触摸灵敏度，以及弥补这个过程中传感器板、覆盖层材料和环境条件引起的不确定性。

6.1 基本原理

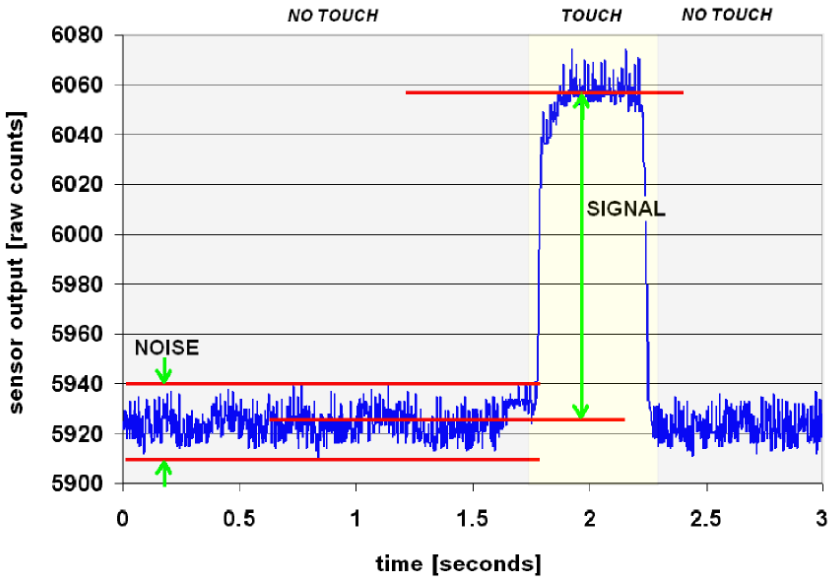
6.1.1 信号、噪声和信噪比

调校好的 CapSense 系统能够准确地识别传感器的 ON 和 OFF 状态。要实现该性能的级别，CapSense 信号必须远远大于 CapSense 噪声。测量时对信号和噪声进行比较，由此得出信噪比（SNR）。讨论 SNR 对 CapSense 的意义之前，需要了解触摸感应环境中信号和噪声这两个概念。

6.1.1.1 CapSense 信号

CapSense 信号是指手指放在传感器上时传感器响应的变化，如图 6-1 所示。传感器的输出是一个数字计数器，该计数器带有表示传感器电容的值。在该示例中，在手指没有触摸传感器时，平均输出为 5925 次计数。当手指放在传感器上时，平均输出增至 6060 次计数。CapSense 信号跟踪手指接触所产生的计数变化，因此，信号 = 6060 – 5925 = 135 次计数。

图 6-1. CapSense 信号和噪声的示例



6.1.1.2 CapSense 噪声

CapSense 噪声是传感器响应不存在手指触摸时的峰至峰变化，如图 6-1 所示。在该示例中，手指未触摸时输出波形跳跃限幅为：最小为 5912 次计数，最大为 5938 次计数。噪声是此波形的最大值与最小值之间的差值，因此，噪声 = 5938 – 5912 = 26 次计数。

注意：如果您的系统中使用了原始数据噪声滤波器，那么请使用过滤后的原始计数来计算噪声。调谐器 GUI 在 Graphing 选项卡上显示的是未过滤的原始计数值，因此您应记录数据从而计算噪声。

6.1.1.3 CapSense 信噪比

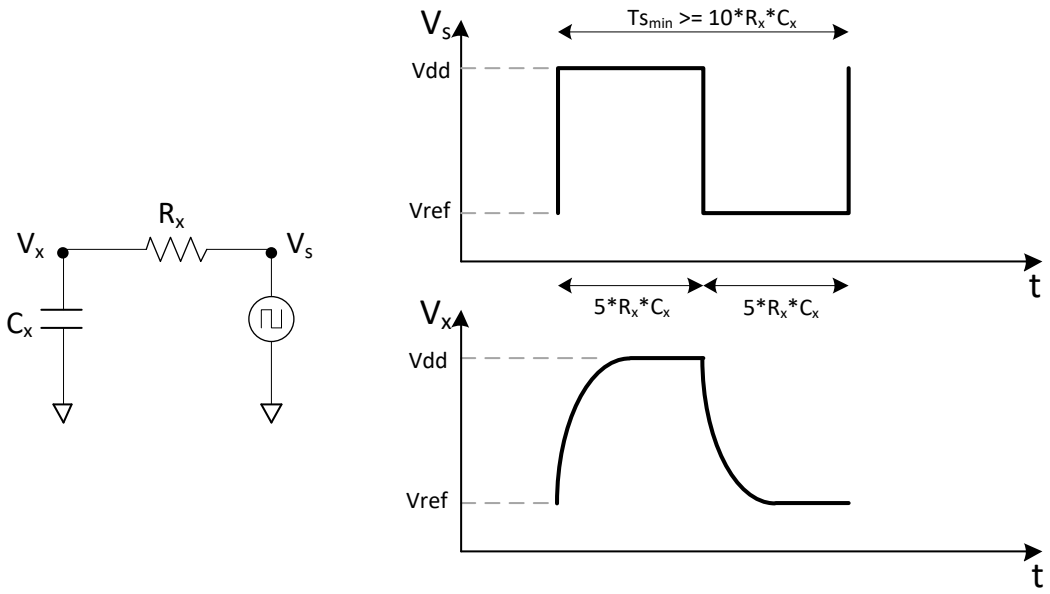
CapSense 信噪比是信号和噪声的简单比例。继续该示例，如果信号为 135 次计数，噪声为 26 次计数，那么 SNR 将为 135:26，这就将 SNR 降低到 5.2:1。推荐 CapSense SNR 最小为 5:1，这表示信号比噪声大 5 倍。滤波器通常实施在固件中以降低噪声。更多有关信息，请参见软件滤波。

6.1.2 充电/放电率

在调校过程中，要获得最大灵敏度，传感器电容在每个周期内必须满充满放。充电/放电路径以与公式 5 所定义的开关时钟相等的速率在两个状态之间进行切换。

充电/放电路径包括串联电阻，这降低了电荷转移的速率。该电荷转移的变化率由 RC 时间常数描绘，该时间常数与传感器电容和串联电阻相关，如图 6-2 所示。

图 6-2. 充电/放电波形



将充电/放电率设置为与 RC 时间常数兼容的级别。基本原则容许每次跃迁周期为 5RC，每个周期有两次跃迁（一次充电、一次放电）。最小时间周期和最大频率的计算公式如下：

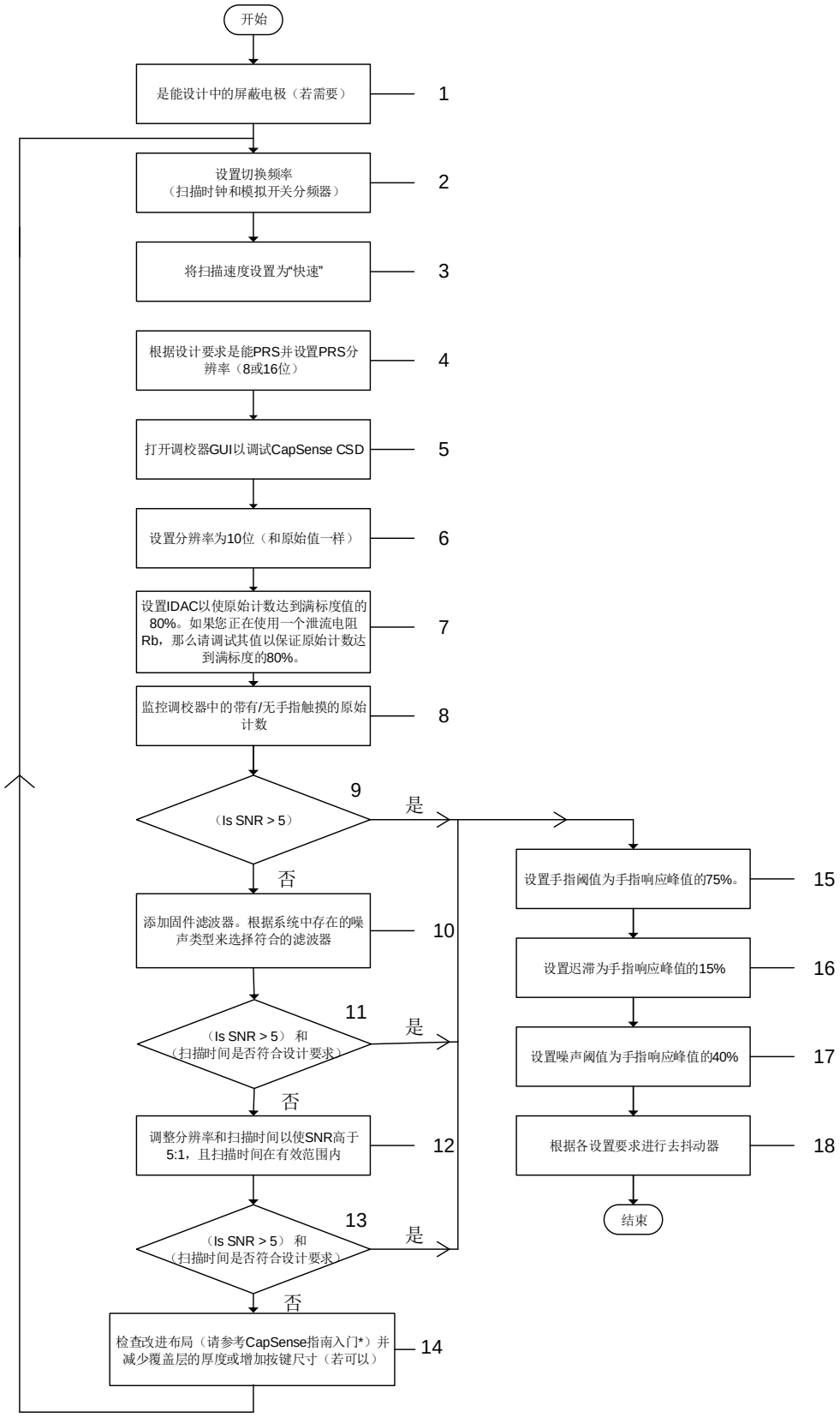
$$Ts_{min} = 10 \times R_X C_X$$ 公式 7

$$fs_{max} = \frac{1}{10 \times R_X C_X}$$ 公式 8

6.2 手动调校

图 6-3 中的流程图提供了手动调校 CapSense_CSD 组建的详细说明。

图 6-3. 调校 CapSense_CSD 组件



1. 请参考防水设计一节中的内容，了解需要使用屏蔽电极的时间。
2. 各个电容式传感器的开关频率设置应能使传感器完全充放电，如模拟开关分频器一节中所述。否则，请加大 CapSense_CSD 配置窗口中的模拟开关分频器的值。一旦更改该值后，您必须重新构建项目并重新编程器件。
3. 如果 SNR 或扫描时间是不可接受的，稍后可以再次对扫描速度进行扫描。
4. 请参考伪随机序列（PRS）一节，了解使用该特性的方式和时间。
5. 请参考调谐器 GUI 一节，了解打开 GUI 的详细说明。
6. 如果设计的覆盖层厚度小于 1 mm，可以将较低的分辨率（8 或 9）作为初始值使用。
7. 在调谐器 GUI 中更改 IDAC 值（IDAC 值），直至原始计数值达到满幅值的 80%。如果使用调谐器 GUI 时未能达到该值的 80%，请调校 CapSense_CSD 组件配置窗口中的 IDAC 范围。一旦更改该值后，您必须重新构建项目并重新编程器件。

   如果正在使用泄流电阻 Rb 而不是 IDAC 灌电流/IDAC 源电流方法，请改变电阻值，使原始计数值等于满幅值的 80%。
8. 请记录峰-峰噪声和峰值手指触摸反应。
9. 信噪比简单表示信号与噪声的比例。为了实现较好的 CapSense 设计，应设置 SNR > 5:1。如果已满足该要求，请跳过第 15 步。
10. 请参考选择滤波器一节，适当选择滤波器类型，以避免系统存在噪声。First Order IIR 1/4（一阶 IIR 1/4）滤波器是一个很好的初始选择，因为它要求的 SRAM 最小但响应速度很快。
11. 请检查当前的 SNR 是否大于 5:1。如果满足要求，请确保扫描时间满足您设计的要求。要想检查扫描时间，需要执行以下操作：
    - 点击调谐器 GUI 上的 OK，以更新 CapSense_CSD 组建的参数
    - 选择扫描顺序选项卡可以知道扫描时间

    如果 SNR 和扫描速度都满足要求，请跳过第 15 步。
12. 您可以通过增大分辨率并降低扫描速度来进一步提高信噪比。但是修改这两个数值会延长扫描时间。
13. 请参考第 11 步。
14. 如果 SNR 仍低于 5:1，可能需要改进 PCB 布局或覆盖层的设计。降低覆盖层厚度和增加按键直径都会提高灵敏度。有关 PCB 设计指南，请参考 CapSense 入门指南的第 3 章。
15. 使用检查 SNR 时所测量到的手指峰值响应的 75%。
16. 使用检查 SNR 时所测量到的手指峰值响应的 15%。
17. 使用检查 SNR 时所测量到的手指峰值响应的 40%。
18. 请参考去抖动一节。通常，应将去抖动值设为 2。在快速扫描设计中，该值应设置为更高的值，比如 5。

注意： 即使调校方式被设置为自动调校，您仍能手动调校各参数。如果通过使用调谐器 GUI 来修改各参数，然后点击 OK，这时手动输入的参数会应用到组件上。

6.3 SmartSense 自动调校

将调校方法设为自动调校时，SmartSense 算法会选择 CapSense_CSD 组件参数。SmartSense 将每个传感器的信噪比维持在 5:1 到 11:1 之间，这样可确保在使 Capsense 获得最佳性能的同时，它仍能稳定运行。您只需要设置参数设置一节所介绍的四个低级参数即可。SmartSense 仅适用于 IDAC 灌电流/IDAC 源电流方法。如果您使用泄流电阻方法，则不能使用这种调校方法。

6.3.1 指南

要使用 SmartSense 自动调校，需要满足下面条件：

- 所有传感器的 C_P 值都要介于 5 - 45 pF 之间
- C_F 值至少为 0.1 pF
- C_{MOD} 电容 X7R、2.2 nF、额定电压大于 5 V
- SmartSense 在自动调校时支持相同的 API。除非已经了解使用 API 修改固件中的 CSD 参数对设计产生的影响，否则请勿使用它。

6.3.2 参数设置

6.3.2.1 传感器自动复位

该参数用于确定更新基准线的时间：随时更新还是仅在信号差值低于噪声阈值时才更新它。当设置为“Enabled”（启用）时，基准线随时更新。虽然该设置限制传感器运行的最长时间（通常是 5 至 10 秒），但是防止了下述情况的发生：由于某种系统故障，即使未触碰传感器，原始计数却突然上升，导致传感器一直运行。请参考传感器自动复位一节。

6.3.2.2 去抖动

去抖动参数为传感器的有效转换添加去抖动计数器。对于从无效状态转换到有效状态的传感器，应在传感器上显示手指触摸信号，作为连续扫描的去抖动数量。该参数影响所有类似传感器。请参考去抖动一节。

6.3.2.3 调制器电容引脚

该参数用于选择 C_{MOD} 电容所连接的引脚。可选的引脚为 P0[1]和 P0[3]。

注意：必须使用一个 2.2 nF 的外部电容，这样 SmartSense 才能正常工作。

6.3.2.4 灵敏度级别

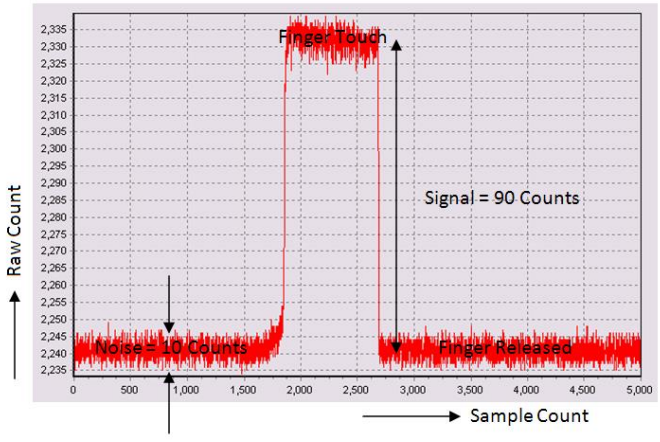
灵敏度用于增加或减少传感器发出的信号强度。可将它设为高（0.1 pF）、偏高（0.2 pF）、偏低（0.3 pF）或低（0.4 pF）。

为满足要求，所设计的覆盖层越厚，则所需传感器信号也越强。为产生更强的传感器信号，SmartSense 必须消耗更长的传感器扫描时间。这便意味着：与灵敏度偏高的传感器相比，灵敏度高的传感器需要更长的扫描时间。

设置该参数时，请尽量使用最低的灵敏度。为确保性能稳定，请执行下列操作：

1. 打开调谐器 GUI。
2. 将灵敏度级别设为低（0.4 pF），并计算信噪比。图 6-4 显示了手指触摸式的典型原始计数图。调整灵敏度级别，直到满足 5:1<SNR<10:1 为止。

图 6-4. 手指触摸时典型传感器的原始计数图



7. 设计的注意事项

当您的应用设计中采用电容式触摸感应技术时，必须将 CapSense 元件置入较大的框架中。请认真考虑从 PCB 布局、用户界面到最终使用操作环境等各项细节，以实现既稳定又可靠的系统性能。有关覆盖层选择、ESD 保护和电磁兼容性（EMC）注意事项等详尽信息，请参考 CapSense 入门文档。本节将对 PSoC 3 和 PSoC 5LP 特定的设计注意事项进行阐述。

7.1 软件滤波

CapSense_CSD 组件可提供一些现成的固件滤波器，能有效消除 CapSense 扫描结果中的噪声。下表对这些滤波器以及何时使用它们进行了介绍。

表 7-1. 组件提供的 CapSense 滤波器列表

滤波器	说明	应用
中值	非线性滤波器用于使用大小为 3 的缓冲区计算中值输入值	来源于电机和开关电源的噪声毛刺
平均值	具有同样的加权系数的有限脉冲响应滤波器（无反馈回路）	来源于电源的周期性噪声
IIR	具有与 RC 低通滤波器类似的阶跃响应的无限脉冲响应滤波器（有反馈回路）	高频噪声
抖动	根据之前输入来限制现在输入的非线性滤波器	来自厚覆盖层的噪声（SNR < 5:1），对滑条中心数据非常有用。

注意：

1. 组件提供的中值滤波器使用大小为 2 的缓冲区。该缓冲区中包含两个之前的样本。当前样本和之前两个样本在缓冲区中进行排序，中间值将取为中值。
2. 组件中的均值滤波器使用大小为 2 的缓冲区。该缓冲区中包含两个之前的样本。滤波器输出是之前两个样本和当前样本的平均值。

图 7-1. 原始计数过滤
黑色 = 未过滤的原始计数
绿色 = 已过滤的原始计数

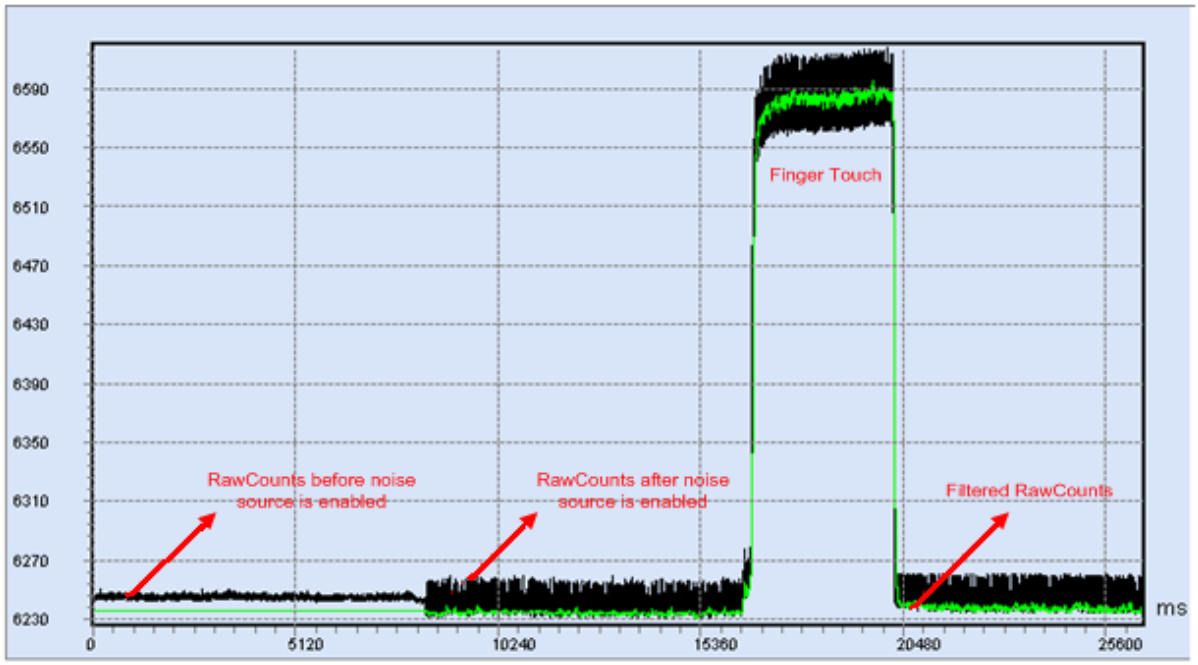


图 7-1 显示的是噪声和过滤对原始计数的影响。通过切换临近的传感器从而生成 500 kHz 的噪声，用于模拟由传感器附近的高电平开关线引起的噪声。在该示例中：

- 无噪声源的噪声 = N = 8 次计数（峰至峰）
- 无噪声源的噪声 = N1 = 40 次计数（峰至峰）
- 滤波之后的噪声 = N2 = 12 次计数
- 信号（手指峰值响应）= S = 320 次计数
- 滤波之后的 SNR = SNR1 = S / N1 = 320 / 40 = 8
- 滤波之后的 SNR = SNR2 = S / N2 = 320 / 12 = 26.7

7.2 功耗

在电池供电应用中，电流电平越低，电池寿命越长。考虑到这一点，系统设计人员应最小化系统的平均电流消耗。

表 7-2. 用于扫描一个传感器的 PSoC3 电流消耗

CPU 速度（MHz）	mA（V_{DD} = 3.3 V）	mA（V_{DD} = 5 V）
3	3.7	4.7
6	4.4	5.5
12	5.9	6.9
24	8.4	9.5
48	14.5	16.3

有多种方法可降低 CapSense 电容式触摸感应系统的功耗。

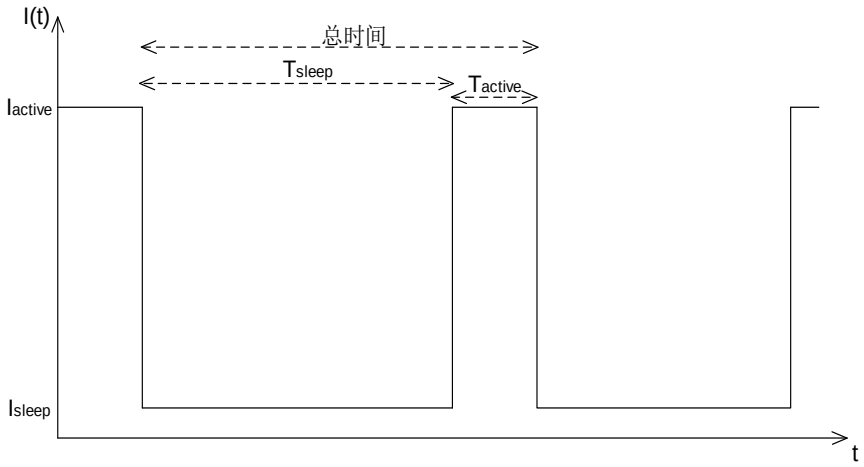
- **将 GPIO 驱动模式设置为 HI-Z（高阻模式）：** 默认情况下所有 GPIO 引脚均被配置为 HI-Z。不过如果项目调整了某些引脚驱动模式的配置，那么不使用时需要将其恢复为 HI-Z。
- **优化 CPU 速度以降低功耗：**由于使用了专用的硬件扫描传感器，因此对于 CapSense 操作不需要 CPU 的速度过快。但是，在硬件开始扫描之前，CPU 需要对硬件进行配置，并在硬件完成扫描之后，将由 CPU 来处理结果。因此，降低 CPU 速度可以延长扫描时间。请参见无阻塞架构，了解如何将扫描总时间分为扫描前处理、硬件扫描以及扫描后处理三个阶段。
- **在较低的 V_{DD} 下运行。**

除上述建议之外，还可应用睡眠扫描方法。

7.2.1 睡眠-扫描方法

在各典型应用中，CapSense 控制器不需要经常保持活动状态。可以使器件进入睡眠状态，从而使器件的 CPU 和主要模块停止运行。在睡眠状态下，该器件所消耗的电流远低于有效电流。

图 7-2. 睡眠-扫描方法



通过以下公式，可以计算出器件在较长的运行时间中所消耗的平均电流。

$$Iaverage = \frac{(Iactive \times Tactive) + (Isleep \times Tsleep)}{Total\ Time}$$ 公式 9

其中：

$I_{average}$ = 器件的平均电流

I_{active} = 器件有效电流

T_{active} = 器件有效时间

I_{sleep} = 器件睡眠电流

T_{sleep} = 器件睡眠时间

总时间 = T_{active} + T_{sleep}

使用下面公式可计算器件的平均功耗：

$$Paverage = V_{DD} \times Iaverage$$ 公式 10

其中：

$P_{average}$ = 器件平均功耗

V_{DD} = 器件的供电电压

$I_{average}$ = 器件平均电流

7.2.2 测量平均功耗

7.2.2.1 有效电流

为了测量有效电流，需要配置器件并测量所有应用引脚上的电压，例如：V_{DDD}、V_{DDA} 和 V_{DDIO}。最佳方法是，短路供电引脚并测量公共供电结点的电流。

7.2.2.2 有效时间

可以通过在扫描环路前后切换一个 GPIO 引脚来测量有效时间。以下代码段为典型的 CapSense 扫描环路，其引脚在开始扫描之前进行设置并在扫描完成之后重新设置。

```
22      for(;;)
23      {
24          if (CapSense_IsBusy() ==0 )
25          {
26              Pin_Write(0);
27              CapSense_UpdateEnabledBaselines();
28              if (CapSense_CheckIsWidgetActive())
29              {
30              /* Do required */
31              }
32
33              /*Check other CapSense sensros active and do the required */
34
35              Pin_Write(1);
36              CapSense_ScanEnabledWidgets();
37          }
38          /* Other application code than CapSense */
39      }
40
41
42
43  }
```

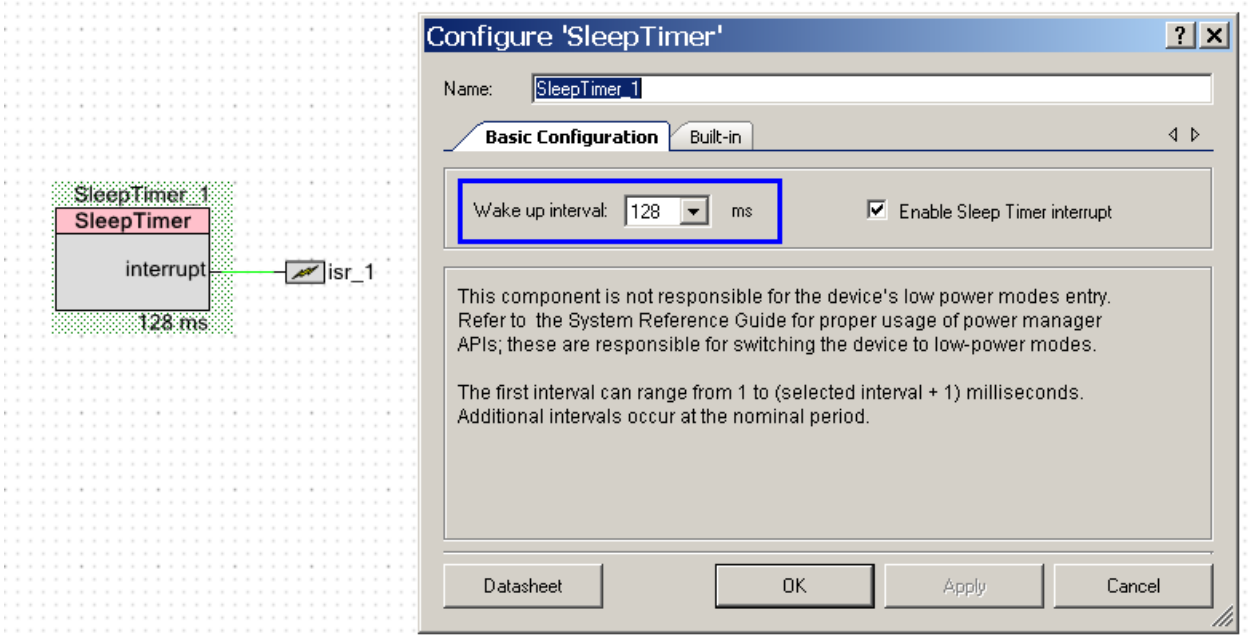
7.2.2.3 睡眠电流

PSoC 3 和 PSoC 5LP 的睡眠电流值分别为 1 µA 和 2 µA。为了测量睡眠电流，需要创建一个能够使器件永久保持睡眠模式的项目，并使用有效电流部分介绍的方法。

7.2.2.4 睡眠时间

通过使用 API“CyPmSleep()” 可以将器件设置为睡眠模式。睡眠定时器将在特定间隔后生成中断，以唤醒器件，如图 7-3 所示。

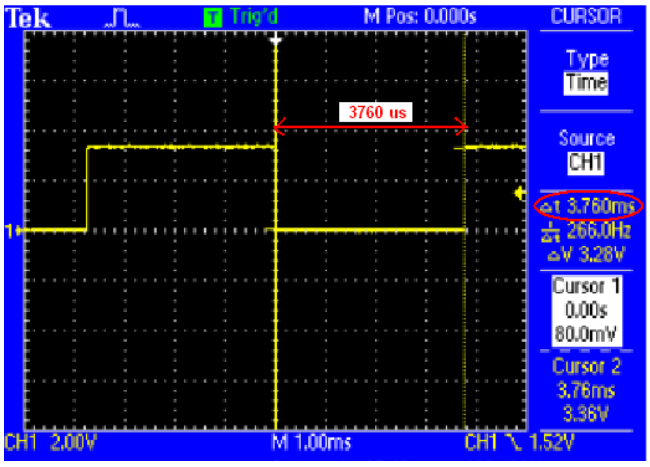
图 7-3. 睡眠定时器



7.2.2.5 计算平均功耗

表 7-1 中的**有效电流**，CPU 速度 = 24 MHz，运行电压为 3 V：
I_{active} = 8.4 mA。

图 7-4. 有效时间测量



PSoC 3 的**有效时间**，CPU 速度 = 24 MHz，分辨率 = 12 位和扫描速度 = 正常：
T_{active} = 3760 µs。

数据手册中的**睡眠电流**：
I_{sleep} = 1 µA

睡眠时间，假设睡眠定时器中将睡眠周期设置为 128 ms：
睡眠时间 = 总时间 – 有效时间
总时间 = 扫描间隔 = 128 ms
T_{sleep} = 128,000 µs – 3760 µs= 124240 µs

平均电流，使用公式 9：
$I_{average}$ = 247.7 µA

平均功耗
$P_{average} = V_{DD} \times I_{average}$
$P_{average}$ = 3.3 V x 247.7 µA = 817.4 µW

使用睡眠扫描模式时的系统平均功耗为 817.4 µW。而在未使用睡眠模式时的系统平均功耗为 27.7 mW。

7.3 响应时间

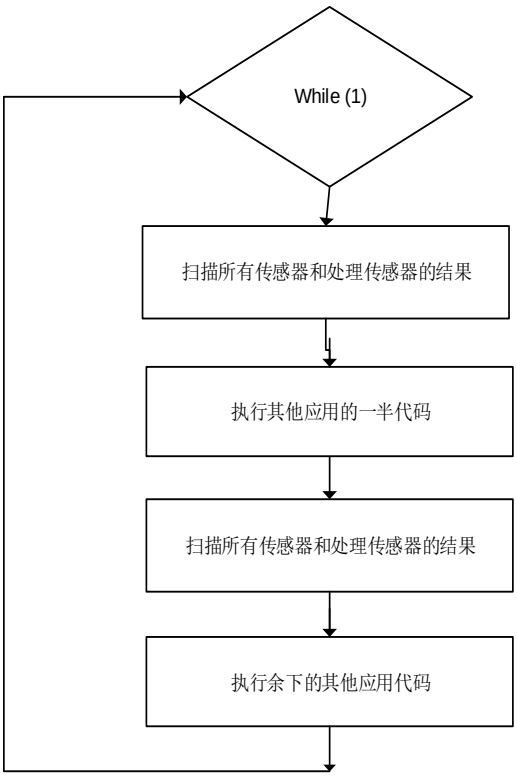
7.3.1 睡眠扫描模式

在扫描间隔内将器件设置为睡眠模式可以节省大量电能。但如果睡眠时间延长过久，会使响应时间变差。如果在您的设计中功耗和响应时间是重要的参数，那么您可以采用能够结合连续扫描和睡眠扫描模式的方法。使用这种方法时，器件大部分时间会处于睡眠扫描模式：完成扫描传感器后，器件会进入睡眠模式。用户触摸传感器时，该器件将切换到连续扫描模式，其中传感器将被连续扫描。器件将在特定时间内处于连续扫描模式。如果用户在此期间不再触摸传感器，则器件将返回睡眠扫描模式。

7.3.2 较长的后台循环

若后台循环执行各种应用代码的时间比 CapSense 越久，响应时间越差。您可以实现多个 CapSense 扫描次数/每个环路，如图 7-5 所示。

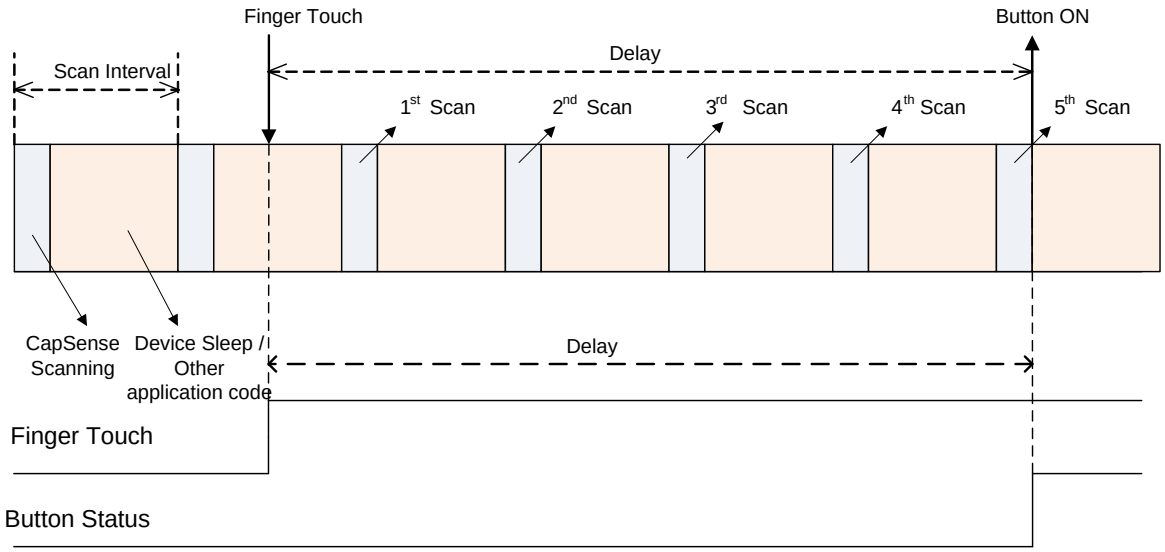
图 7-5. 在循环中多次扫描可提高响应时间



7.3.3 去抖动

去抖动功能通过检测伪手指触摸有助于保护您的系统，但它也会延长手指检测的时间，如图 7-6 所示。

图 7-6. 扫描间隔和检测延迟，去抖动计数 = 5



基于去抖动的最大延迟如下所示：

$$Max\ Delay = Debounce \times Scan\ Interval$$

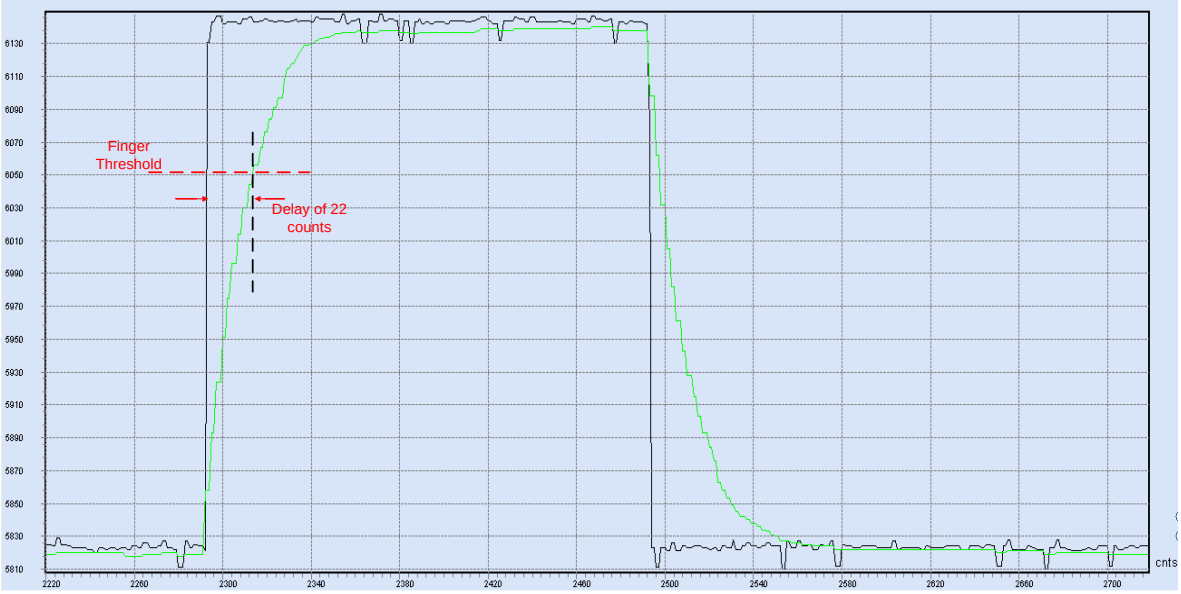
在睡眠模式下，应将去抖动值设置为 1。由于睡眠模式会延长响应时间，因此能够有效地去抖动信号。如果没有充分去抖动，在检测到手指触摸时，您可以对传感器多次进行扫描，如下面代码片段所示。

```
if (CapSense_CheckIsWidgetActive(CapSense_BUTTON0__BTN))
{
    /* Scan the sensor multiple times */
    for (i=0; i < DEBOUNCE ; i++)
    {
        if ( (CapSense_IsBusy() == 0 ) && CapSense_CheckIsWidgetActive(CapSense_BUTTON0__BTN) )
        {
            CapSense_ScanEnabledWidgets();
        }
    }
    /* After number of scans equal to debounce count is complete, check for sensor active.
    * If it is still active, then sensor can be considered as ON.
    */
    if (CapSense_CheckIsWidgetActive(CapSense_BUTTON0__BTN))
    {
        /*  Process the code which is based on sensor active */
    }
}
```

7.3.4 滤波器延迟

使用滤波器也会延迟手指触摸检测。下面介绍的是使用 IIR16 滤波器的作用：注：IIR 16 滤波器中增加了 1/16 的新数据和 15/16 的旧数据来计算新的滤波样本。

图 7-7. 手指触摸时的滤波器延迟



基于手指的步长变化 = 320 次计数

手指阈值 = 手指峰值响应的 75% = 240 次计数。

使用下面公式可计算基于滤波器的延迟：

$T_n = a \times r^{n-1}$

其中：

a = 1，步长变化或手指峰值感应

Tn = 0.25，因为步长变化的 75%是阈值。

r = 1/16，因为滤波器器是 IIR16

n = 22

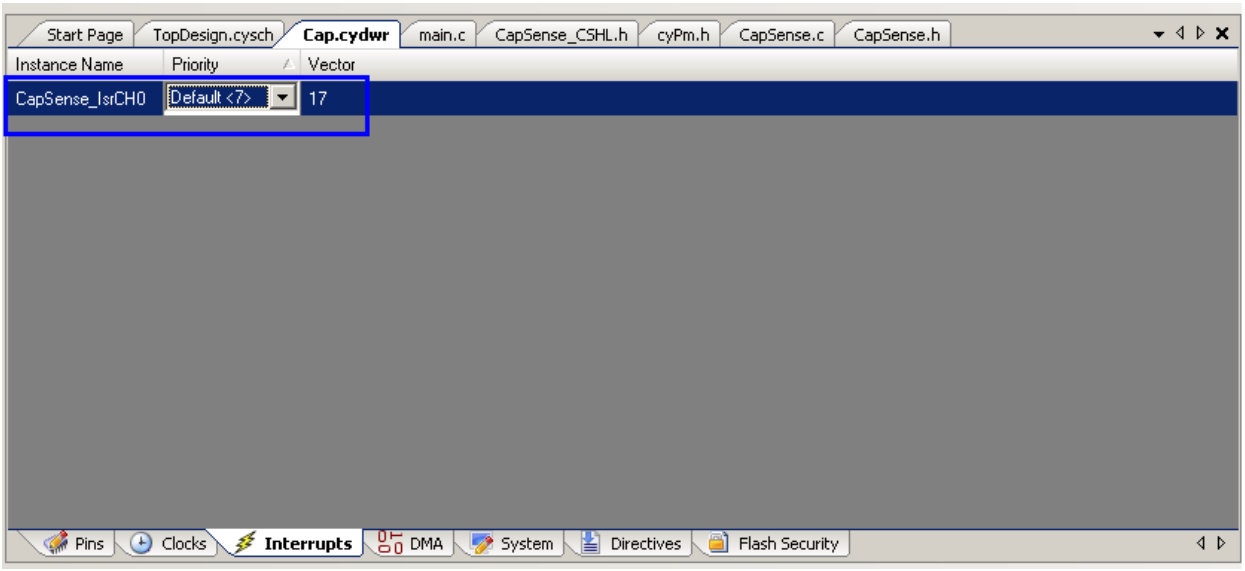
因此手指触摸仅在 22 次扫描后被检测到。要降低滤波器延迟，选择适当的滤波器是一个很好的方法。例如，使用 IIR4 滤波器时只会发生 7 次扫描延迟。

如果需要使用 IIR16 滤波器，建议您进行多次扫描，如去抖动部分所介绍的内容。不管传感器当前处于 ON 还是 OFF 状态，都应对每个扫描环路进行多次扫描。

7.3.5 中断优先级

如无阻塞架构部分所述，CapSense_CSD 组件的代码架构具有无阻塞性质，因此硬件扫描 CapSense 传感器时，CPU 可以执行其他应用代码。在完成扫描之后，硬件将对 CPU 生成中断。要想调整 CapSense 中断的优先级，请打开<project name>.*cydwr* 文件并点击 Interrupts（中断）选项卡。

图 7-8. 更改中断优先级



如果存在优先级更高的中断（如通信协议中断），那么可以降低 CapSense 中断的优先级。如果在激活 CapSense 中断时，正在处理某个优先级更高的中断，那么 CapSense 中断将延后处理。虽然延迟期间不会丢失数据，但它会使下一个传感器扫描被延迟，从而延长总扫描时间。

7.4 引脚分配

7.4.1 运算放大器输出引脚

运算放大器输出直接被连接到 P0[0]、P0[1]、P3[6]以及 P3[7]上。如果将这些引脚用于 CapSense，则无法使用运算放大器。由于这些引脚直接连接着运算放大器输出端，所以它们的 C_P 较大。所以请避免将这四个引脚使用于 CapSense。如果您必须使用这些引脚，请将它们用于按键，并避免将其用于滑条或触控板。此外，请不要短接这些引脚的走线，以便减少 C_P。

7.4.2 双通道设计的引脚分配

在使用双通道设计时，属于特定通道的引脚应放置于芯片的同一侧。有关双通道设计的更多详细信息，请参见通道数量。

7.4.3 C_{MOD} 引脚分配

为实现最高效的模拟路由，应将以下各引脚应用于 C_{MOD} 连接。

- 左侧：P2 [0]、P2 [4]、P6 [0]、P6 [4]、P15 [4]
- 右侧：P1 [0]、P1[4]、P5 [0]、P5 [4]

7.5 PCB 布局指南

《CapSense 入门手册》中介绍了详细的 PCB 布局指南。

8. 资源

8.1 网站

欲了解 PSoC 3 和 PSoC 5LP 中的 CapSense 应用的相关信息，请访问 CapSense® PLUS。

8.2 数据手册

要想获取 PSoC 3 和 PSoC 5LP 器件系列的数据手册，请访问 www.cypress.com。

- PSoC 3 数据手册
- PSoC 5LP 数据手册

8.3 技术参考手册

通过以下技术参考手册，您很容易便能快速查询到有关 PSoC 3 和 PSoC 5LP 架构的信息，如顶层架构框图、寄存器汇总以及时序框图。

- PSoC 3 技术参考手册
- PSoC 5LP 技术参考手册

8.4 开发套件

8.4.1 PSoC 3 和 PSoC 5LP 开发套件

- CY8CKIT-001 PSoC®开发套件
- CY8CKIT-030 PSoC® 3 开发套件
- CY8CKIT-050 PSoC® 5LP 开发套件

8.4.2 连接模块电路板与开发套件的接口电路板

- CY8CKIT-031 PSoC CapSense 扩展板套件

8.4.3 通用 CapSense 模块板

8.4.3.1 简单触摸按键模块板

CY3280-BSM 简单按键模块由十个 CapSense 按键和十个 LED 构成。此模块可以连接到任何 CY3280 通用 CapSense 控制器电路板

8.4.3.2 矩阵触摸按键模块板

CY3280-BMM 阵列按键模块由 8 个 LED 和 8 个 CapSense 传感器组成（以 4×4 阵列格式组织，从而构成 16 个物理按键）。该模块可连接至任何 CY3280 通用 CapSense 控制器电路板。

8.4.3.3 线性滑条模块板

CY3280-SLM 线性滑条模块由五个 CapSense 按键、一个线性滑条（带十个传感器）和五个 LED 组成。该模块可连接至任何 CY3280 通用 CapSense 控制器电路板。

8.4.3.4 辐射滑条模块板

CY3280-SRM 辐射滑条模块由四个 CapSense 按键、一个辐射滑条（带十个传感器）和四个 LED 组成。该模块可以连接至任何 CY3280 通用 CapSense 控制器电路板

8.4.3.5 通用 CapSense 原型设计模块

CY3280-BBM 通用 CapSense 原型设计模块提供了与路由至连有的控制器电路板上 44 引脚连接器的各个信号的连接。原型设计模块电路板与通用的 CapSense 控制器电路板一起使用，用于实施其他功能（该功能不是其他单用途通用 CapSense 模块电路板所拥有的）。

8.4.4 MiniProg3

MiniProg3 可用于编程和调试 PSoC 3 和 PSoC 5LP 器件。此外，它还可以作为调谐器 GUI 的 I²C-USB 通信桥接器使用。

- CY8CKIT-002 PSoC® MiniProg3 编程和调试套件 http://www.cypress.com/?rID=39045

8.5 PSoC Programmer

PSoC Programmer 是用来为 PSoC 器件编程的既灵活又具有高集成度的编程应用程序。它可与 PSoC Designer 和 PSoC Creator 搭配使用，进行 PSoC 器件的任何设计编程。

PSoC Programmer 为您提供了带有 API 的硬件层，以便通过使用编程器和桥接器来设计特定应用程序。在 COM 指南文档中介绍了 PSoC Programmer 硬件层以及跨下列语言的示例代码：C#、C、Perl 和 Python。

8.6 Multi-Chart（多图工具）

除了使用 I²C 通信的调谐器 GUI 外，用户还可使用 UART 监测 CapSense 的结果。Multi-Chart 是基于 UART 通信的简单 PC 工具，使用它可以实时查看和记录 CapSense 数据。借助该应用，您可以查看来自多达 48 个传感器的数据、保存和打印图表、保存数据以便日后在电子表格中进行分析。

8.7 PSoC Creator

赛普拉斯具备独有的集成设计环境，PSoC Creator。借助 PSoC Creator，您可以在单个工具中进行硬件配置和固件开发。

8.8 代码示例

赛普拉斯提供了多个代码示例，以便实现您的设计并使它快速运行。打开 PSoC Creator，然后点击 Start Page（起始页）中的“find example project”链接。为了熟悉 CapSense，建议您参考下面两个实例项目

- CapSense_CSD_Design
- CapSense_CSD_With Tuner

8.9 设计支持

赛普拉斯具有各种设计支持渠道，以确保 CapSense 解决方案成功。

- 知识库文章 — 按产品系列浏览技术文章或对各种 CapSense 主题执行搜索。
- CapSense 应用手册 — 包括与本文档中的信息密切相关的广泛应用笔记。
- 白皮书 — 了解电容式触摸接口的高级主题。
- 赛普拉斯开发社区 — 与赛普拉斯技术社区联系并交换信息。
- 视频库 — 使用视频教程提高学习速度。
- 质量和可靠性 — 赛普拉斯承诺满足客户的要求。在我们的质量网站上，可以找到可靠性和产品资质报告。
- 技术支持 — 在线提供一流的技术支持。

术语表

AMUXBUS

指的是 PSoC 中的模拟复用器总线，通过它可将 I/O 引脚连接至多个内部模拟信号。

SmartSense™ 自动调校

设计阶段结束后，CapSense 算法自动设置各个感应参数以得到最佳性能，然后连续补偿由于系统、生产过程和环境不同引起的变化。

基准线

指的是从固件算法得到的数值。当传感器上没有手指触摸时，该算法将估计原始计数的值。基准线对原始计数突变的灵敏度较低，另外它还为计数差值提供了参考点。

按键或按键 widget

指的是带有相关传感器的 widget，它会报告传感器的活动或非活动状态（即仅两种状态）。例如，它可以检测到传感器上是否有手指触摸。

计数差值

指的是原始计数与基准线间的差值。如果该差值为负，或如果它低于噪声阈值，则计数差值总是被设置为‘0’。

电容式传感器

导体和基板（如印刷电路板（PCB）上的铜质按键）会对触摸事件或接近电容变化物体作出反应。

CapSense®

赛普拉斯的触摸感应用户界面的解决方案这是行业排名第一的解决方案，销量是排名第二的方案的四倍。

CapSense 机械按键替换（MBR）

将机械按键升级到电容式按键的赛普拉斯可配置解决方案仅需要很少的工程功耗，并且不需要固件开发。这些器件包括 CY8CMBR3XXX 和 CY8CMBR2XXX 系列。

中心或中心位置

是指在滑条分辨率所给定的范围内，表示滑条上的手指位置的数字。该数字由 CapSense 中心计算算法计算得出。

补偿 IDAC

指的是可编程的恒流源，CSD 通过使用该恒流源补偿多余的传感器 C_P。与调制 IDAC 不同，该 IDAC 没有受 CSD 模块中 Sigma-delta 调制器的控制。

CSD

CapSense Sigma Delta（CSD）是赛普拉斯专利方法，用于测量电容式感应应用的自电容。

在 CSD 模式下，感应系统测量电极的自电容，且检测自电容的变化，从而确定是否有手指触摸。

去抖动

用于定义连续扫描样本数量的参数，只有存在手指触摸时，该参数才有效。该参数有助于抑制假的触摸信号。

对于连续扫描样本的去抖动数量，仅在计数差值大于手指阈值+迟滞时，手指触摸才被报告。

驱动屏蔽（Driven-Shield）

指的是 CSD 所使用的一种技术，用于使能防水功能，其中屏蔽电极由一个信号驱动，该信号的相位和幅度与传感器开关信号的相等。

电极

指的是导电材料，如 PCB 板、ITO 或 FPCB 板上的垫片或物理层。电极连接到 CapSense 器件的端口引脚，并作为 CapSense 传感器使用或用于驱动与 CapSense 功能相关的特定信号。

手指阈值

与 Hysteresis（迟滞）一起使用的参数，旨在确定传感器的状态。如果计数差值高于手指阈值+迟滞，传感器状态将显示‘ON’；如果计数差值低于手指阈值-迟滞，则传感器状态将显示‘OFF’。

组合传感器

这是将多个传感器连接在一起，并将它们作为单个传感器进行扫描的方法。该方法用于扩大接近感应的传感器面积，并降低功耗。

当系统处于低功耗模式时，为了降低功耗，需要将所有传感器连接在一起并将其作为单个传感器进行扫描（而不是单独扫描所有传感器），这样可以缩短扫描时间。当用户触摸任何传感器时，系统会进入活动模式，在该模式中，它会单独扫描所有传感器，以检测哪个传感器被激活。

PSoC 通过固件支持传感器组合，这意味着，可以将多个传感器同时连接到 AMUXBUS，以进行扫描。

手势

手势是一个由用户执行的动作，如滑动和线捏/缩放等等。CapSense 具有手势检测功能，即根据预定义的触摸格式来识别不同的手势。在 CapSense 组件中，只有触摸板 widget 支持手势功能。

保护传感器

指的是 PCB 板上围绕所有传感器的铜线，它类似于按键传感器并用于检测水流。触发保护传感器时，固件会禁用对所有其他传感器进行的扫描，以防止误触摸。

网格填充、网格地填充或网格铺地

当设计一个拥有电容式感应功能的 PCB 板时，应将铜制接地层放置在传感器周边，以获取良好的抗噪能力。但是实心接地层会使传感器的寄生电容增加（这种电容是不需要的）。因此，应该以特殊网格方式填充接地层。网格图案被紧密放置、纵横交错，同丝网一样，线宽度和两条线间的距离决定了填充百分比。具有防水功能时，将通过屏蔽信号（而不是接地层）驱动该网格填充（作为屏蔽电极使用）。

迟滞

用于防止由系统噪声产生随机切换造成传感器状态的参数，它与手指阈值一起使用，以确定传感器状态。请查看手指阈值。

IDAC（电流输出的数模转换器）

PSoC 中的可编程恒流源，用于 CapSense 和 ADC 操作。

防水功能

存在水滴、水流或薄雾时，电容感应系统仍能够正常工作的能力。

线性滑条

指的是至少包含一个传感器的 Widget。这些传感器以特殊的线性方式安排以检测手指的物理位置（在单轴上）。

低基准线复位

表示扫描样本最大数量的参数，其中原始计数异常低于负噪声阈值。如果超过了低基准线复位值，基准线将被复位到当前的原始计数。

手动调校

指的是手动设置（或调校）CapSense 参数的过程。

矩阵按键

指的是至少包含两个传感器（这些传感器以矩阵方式安排）的 widget。通过使用它可以在各个传感器（这些传感器以垂直方向和横向安排）的交点上检测是否有手指（触摸）。

如果 M 是横轴上的传感器数量，且 N 是纵轴上的传感器数量，那么矩阵按键 Widget 只需要使用 M + N 端口引脚就可以监控 M x N 总交叉点。

使用 CSD 感应方法（自电容）时，该 Widget 一次只能检测一个交叉点位置上的有效触摸。

调制电容（CMOD）

在自电容感应模式下 CSD 模块操作所需要的外部电容。

调制器时钟

指的是一个时钟源，在传感器扫描过程中用于采样从 CSD 模块输出的调制器。该时钟也是原始计数计数器的源。扫描时间（不包括前处理和后处理时间）的计算公式为$(2^N – 1)$/调制器的时钟频率，其中 N 是扫描分辨率。

Modulation IDAC（调制 IDAC）

调制 IDAC 是可编程的恒流源，它的输出由 CSD 模块中的 Sigma-delta 控制器输出控制（ON/OFF），以保持 AMUXBUS 电压始终为 V_{REF}。该 IDAC 提供的平均电流等于传感器电容引出的平均电流。

互电容

一个电极（假设为 TX）与另一个电极（假设为 RX）间的相对电容被称为互电容。

负噪声阈值

用于区分通常噪声与不想要的杂散信号的阈值。该参数与低基准线复位参数结合使用。

通过更新基准线，可以跟踪原始计数和负噪声阈值范围内的原始计数的变化，也就是基准线与原始计数之差（基准线 – 原始计数）小于负噪声阈值。

负方向的杂散信号可被触发的场合包括：上电时传感器上有手指触摸，除去传感器附近的金属物体，移除带有防水功能的 CapSense 产品上的水滴，以及突然发生其他的环境变化。

噪声（CapSense 噪声）

传感器处于‘OFF’状态（无触摸）时原始计数的变量，使用峰至峰计数来测量。

噪声阈值

用于区分传感器的信号和噪声的参数。如果原始计数 – 基准线的值大于噪声阈值，该参数将表示信号可能有效。如果差值小于噪声阈值，则该原始计数仅包括噪声。

覆盖层

指的是覆盖电容式传感器，并用作触摸表面的非导电材料（如塑胶和玻璃）。将带有多个传感器的 PCB 直接放置在覆盖层下面，或通过弹簧连接。产品的外壳常作为覆盖层使用。

寄生电容（C_P）

寄生电容是由 PCB 走线、传感器垫片、过孔以及气隙组成的传感器电极的内部电容。这是不想要的情况，因为它会使 CSD 的灵敏度降低。

接近感应传感器

指的是不需要物理接触却能够检测到附近的物体的传感器。

辐射滑条

指的是包含多于一个传感器的 Widget。这些传感器以特殊的圆形方式设置，以检测手指的物理位置。

原始计数

代表传感器物理电容的 CapSense 硬件模块的未处理数值输出。

刷新间隔

传感器两次连续扫描间的时间。

扫描分辨率

由 CSD 模块生产的原始计数分辨率（单位为位）。

扫描时间

完成传感器的扫描过程所需要的时间。

自电容

与电路接地和电极相关的电容。

灵敏度

指的是原始计数随传感器电容的变化，用计数/pF 来表示。传感器灵敏度取决于电路板布局、覆盖层属性、感应方法以及调校参数。

感应时钟

用来实现 CSD 感应方法的开关电容前端的时钟源。

传感器

请参见电容式传感器。

传感器自动复位

用于防止传感器无限期地报告由系统故障或金属物体连续显示在传感器附近时造成的误触摸状态的设置。

使能传感器自动复位时，即使计数差值大于噪声阈值，也可以更新基准线。这样将防止传感器无限期地报告 ‘ON’ 状态。禁用传感器自动复位时，只有计数差值小于噪声阈值时才能更新基准线。

传感器组合

请参见组合传感器。

屏蔽电极

传感器周围填充铜，以便防止水滴或其他液体引起的误触摸。屏蔽电极由 CSD 模块输出的屏蔽信号驱动。请参见驱动屏蔽（Driven-Shield）。

屏蔽槽电容（C_{SH}）

指的是（当有一个带有高的寄生电容的大屏蔽层时，）用于增强 CSD 屏蔽的驱动能力的可选外部电容（C_{SH} 槽电容）。

信号（CapSense 信号）

计数差值还被称为信号。请参见计数差值。

信噪比（SNR）

有手指触摸时的传感器信号与无手指触摸时的传感器信号间的比例。

滑条分辨率

表示滑条上需要处理的手指位置总数的参数。

触摸板

指的是包含多个传感器的 Widget（这些传感器以特殊的横向和纵向安排），用于检测一个触摸的 X 和 Y 位置。

触摸板

请参见触摸板。

调校

“调校”是使 CapSense 操作中所需的各种硬件和软件或阈值参数达到最佳值的过程。

V_{REF}

PSoC 中的可编程参考电压模块，用于 CapSense 和 ADC 操作。

Widget

指的是 CapSense 组件中包括一个传感器或一组类似传感器的用户界面元素。受支持的 widget 包括按键、接近感应传感器、线性滑条、辐射滑条，矩阵按键和触摸板。

修订记录

文档修订记录

文档标题：AN75400 - PSoC® 3 和 PSoC® 5LP CapSense® 设计指南

文档编号：001-80488

版本	提交日期	变更人	变更说明
**	01/11/2012	PVKV	新设计指南。
*A	06/24/2015	XZNG	本文档版本号为 Rev*A，译自英文版 001-75400 Rev*B。
*B	07/25/2017	AESATMP9	更新徽标和版权。
*C	01/16/2019	XITO	本文档版本号为 Rev*C，译自英文版 001-75400 Rev*E。