CapTIvate™电容式触摸入门手册

版本历史

版本	时间	作者	说明
1.0	4/6/2020	Eason Zhou/Xiaodong Li	第一版正式版本
1.1	5/26/2021	Eason Zhou/Xiaodong Li	更新参考文档
1.2	6/20/2021	Eason Zhou/Xiaodong Li	增加原理描述，更改错误

术语和缩写

缩写/术语	Meaning / ExplanatI/On	含义/解释
FRAM	Ferroelectric RAM (FeRAM, F-RAM or FRAM)	铁电存储器（FeRAM、F-RAM 或 FRAM）
GUI	Graphical user interface	图形用户界面
IDE	Integrated development environment	综合开发环境
BSL	Bootloader	引导加载程序
JTAG	JTAG(named after the Joint Test ActI/On Group) is an industry standard for verifying designs, testing printed circuit boards programming and debugging after manufacture	JTAG（联合测试工作组）是一种国际标准测试协议（IEEE 1149.1 兼容），主要用于芯片内部测试,下载程序与调试
SBW	2-wire Spy-Bi-Wire interface, a typical JTAG interface for MSP430	两线 Spy-Bi-Wire 接口, 专门用于 MSP430 的串行 JTAG 通讯接口
MSP	Mixed Signal Processor	混合信号处理器
NVM	Nonvolatile memory	非易失性存储器

参考文档

- MSP430 开发手册：
  - MSP430™ MCUs 开发手册★
- CapTIvate™ 用户手册：
  - CapTIvate™ Technology Guide（最新版）
  - CapTIvate™ 技术指南 – 简化版★
- CapTIvate™ 设计流程指导手册：
  - Capacitive Touch Design Flow for MSP430™ MCUs With CapTIvate™ Technology(最新版)
  - 采用 CapTIvate™ 技术的 MSP430™ MCU 的电容式触控设计流程★
- CapTIvate™ 其他相关应用文档：
  - Sensitivity, SNR, and design margin in capacitive touch applications（最新版）
  - 电容式触控应用中的灵敏度、SNR 和设计裕度★
  - Enabling noise tolerant capacitive touch HMIs with MSP CapTIvate™ technology
  - Capacitive Touch Gesture Software and Tuning
  - Automating Capacitive Touch Sensor PCB Design Using OpenSCAD Scripts

目录

1 引言

1.1 概述

电容式触摸感应，是一种通过电容的变化来检测手指接近或触及触摸表面的技术。通过电容式感应， 机械开关和旋钮可替换为外观雅致的按钮、 滑条和滚轮,以解决：

1. 长时间使用后磨损和可靠性降低
2. 前面板与按键之间存在缝隙，容易被水分渗透，而引起不良
3. 需施加力度才能触发
4. 前面板开孔会一定程度上增加成本
5. 按键形状较为固定

TI CapTIvate™电容触摸技术支持按键，接近感应，滚轮，滑条，触摸面板 5 种传感器类型，支持多种覆盖材料。具有功耗低，感应技术强大稳定，抗噪能力强，支持防水功能的特点。

整个文档按实际的电容触摸开发流程编写，也是为了帮助读者快速了解 TI CapTIvate™电容触摸技术的全貌。对于前期功能评估，建议阅读章节 1 章节 2 和附录。对于硬件与机械结构开发，建议阅读章节 1 和章节 2 和章节 3。对于软件开发，建议阅读章节 2，章节 4 和附录。

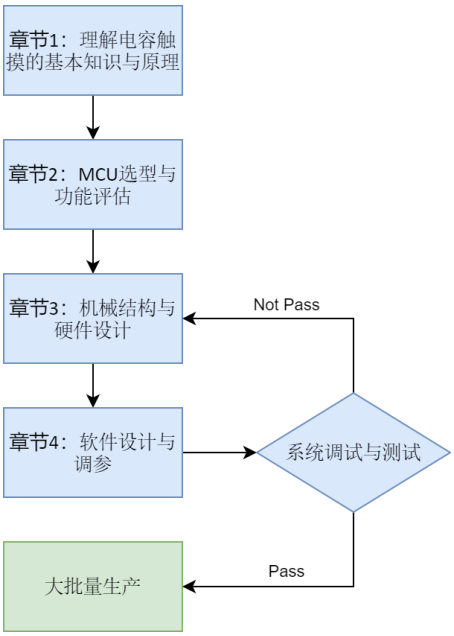


图 1-1 电容触摸开发流程

1.2 相关词汇介绍

CapTIvate™：TI 的电容触摸设计系统

Base 电容：手指触摸前的传感器寄生电容

CAP I/O：MSP430 上专门用于实现电容触摸功能的引脚

LPM0/3/4：MSP430 不同的低功耗模式，具体功耗可参考相关器件的 datasheet

Rx：互电容或自电容模式下，负责从寄生电容往 MCU 内部参考电容充电的引脚/电极

Tx：互电容下，负责给寄生电容充电的引脚/电极

2 电容触摸的基本知识与原理

常见的电容触摸传感器如图 2-1 所示，一般以 PCB 上的覆铜作为电极。结构上，顶层会覆盖非导电性的防护层，如玻璃或塑料，利用胶水和 PCB 粘连。另外传感器周围会覆有网格地。

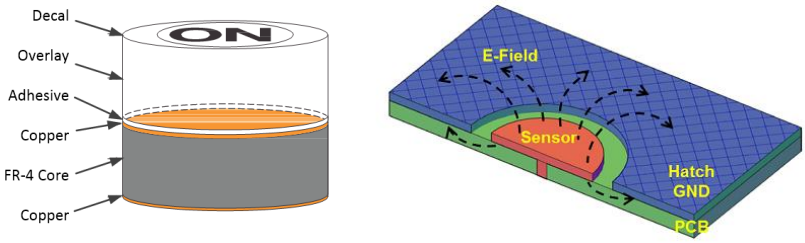


图 2-1 电容触摸结构示意图

基于所检测电容的类型，电容触摸可分为自感型电容检测（检测单电极和地之间的电容值），互感型电容检测（检测双电极之间的电容值）。

2.1 自感型电容检测

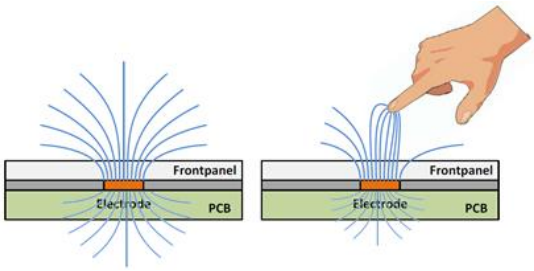


图 2-2 自感型电容检测示意图

以最简单的单按键为例，自感型电容的检测示意图如图 2-2 所示，检测模型如图 2-3 所示。自感型电容利用覆铜形成的单电极（接收电极 Rx），来检测电极对地的电容变化。按键对地的初始电容为 C_p. 当人手触摸时，会给整个环路引入 C_t，C_h 与 C_g，从而使按键的对地电容增大。

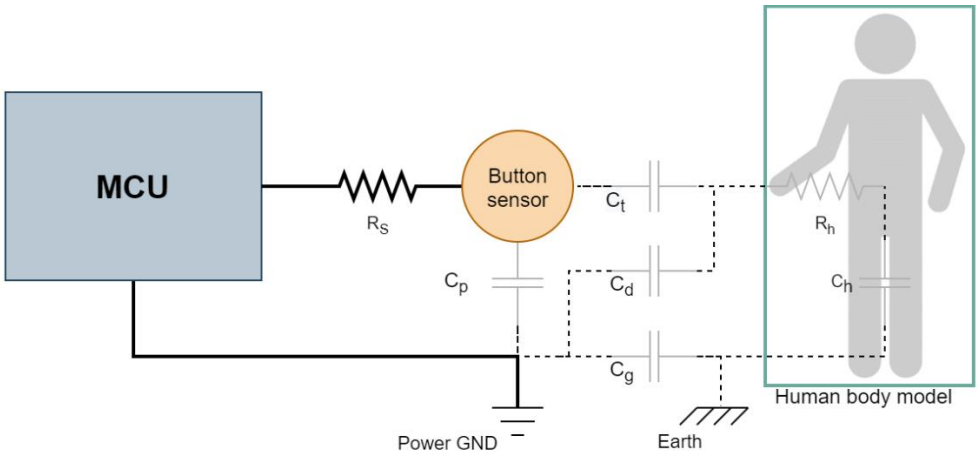


图 2-3 自感型电容检测模型

说明：实线表示实际走线，虚线表示非实际走线。灰色元器件表示等效电容或电阻。

R_h: 人体电阻。

R_s: 串联电阻，推荐值为 470Ω。

C_p: 按键与所连导线的对电源地寄生电容。

C_g: 电源地与大地之间的电容。对于电池应用，大约为 1pF。对于接地应用为短路。
C_h: 人体与大地之间的串联电容。
C_t: 电级与人指尖形成的电容，类似于平板电容器结构。
C_d: 人手与电源地形成的电容。
为便于分析，忽略 R_h，R_s 的影响。按键对地的等效电容如公式 1-1 所示。灵敏度可以表征为触摸产生的电容变化与基础电容之间的比，如公式 1-1 所示。其中由于 C_h 较于 C_g 和 C_t 较大，因此可忽略。在地平面较稀时，C_d较小，因此$C_g + C_d$可约等于C_g。

$$C_{equal} = C_{touch} + C_{base} = C_t||(C_h||C_g + C_d) + C_p \approx C_t||C_g + C_p \tag{1-1}$$

$$Sensitivity = \frac{C_{equal} - C_{base}}{C_{base}} \approx \frac{C_t||C_g}{C_p} \tag{1-2}$$

而平行板电容的计算公式为：

$$C = \varepsilon_r \varepsilon_0 \frac{A}{d} \tag{1-3}$$

A: 手指与传感器垫片覆盖层的接触面积。
d: 覆盖层的厚度。
ε_0: 空气介电常数。
ε_r: 覆盖层的介电常数。
由公式 1-2 和 1-3 可知，提高灵敏度的方法有：1）减小盖板的厚度，提高盖板的ε_r，从而提高C_t；2）减小网格地的密度，或增加 PCB 的厚度，从而降低C_p；3）由于C_t与C_g数量级相同，合理的将电源地与大地相连从而增加C_g；4）合理的增大电极的面积，通过提高手指与传感器垫片覆盖层的接触面积 A 来提高 C_t。要注意，无法通过无限增大电极的方式来增加灵敏度。主要因为平行板电容 C_t 的最大有效面积与手指触摸面积相同，另外过大的电极面积无法增加触摸信号强度，反而会增加C_p，导致灵敏度降低。

2.2 互感型电容检测

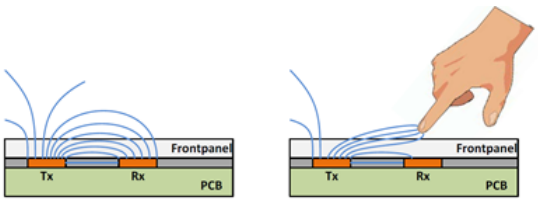


图 2-4 互感型电容检测示意图

如图 2-4 所示，互感型电容利用覆铜形成的双电极（接收电极 Rx，发送电极 Tx）来检测两电极之间电容的变化。互感型电容检测的最大特点是可以忽略按键对电源地的寄生电容 C_p 的影响。以最简单的单按键为例，互感型电容的检测模型如图 2-5 所示。当人手触摸时，C_{RT} 变成两个 $2C_{RT}$，同时引入 C_{RTt}，C_t，C_h 与 C_g。最终使双电极间的电容减小。

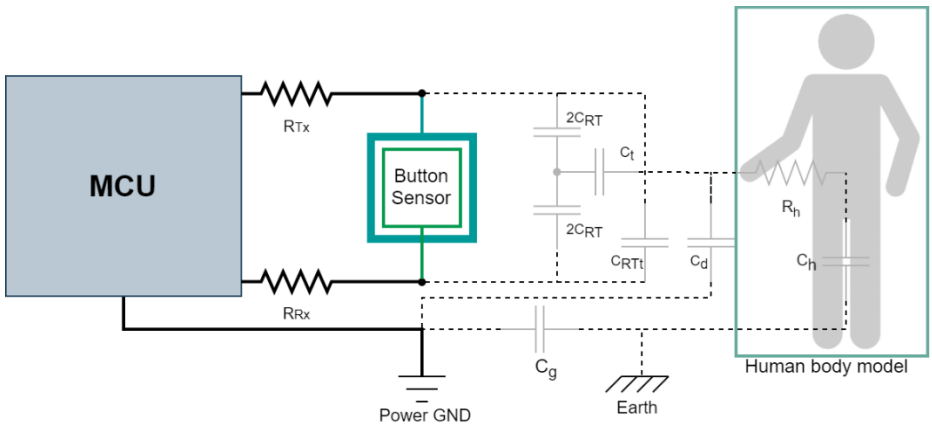


图 2-5 互感型电容检测模型

说明：实线表示实际走线，虚线表示非实际走线。灰色元器件表示等效电容或电阻。

C_{RTt}: 手指触摸引入的 Rx 和 Tx 电极之间的并联电容。

C_{RT}: Rx 和 Tx 电极之间的电容，当人手触摸时，等效分为两个容值 2 C_{RT} 的电容。

Tx 与 Rx 之间的等效电容如公式 1-4 所示。灵敏度可以表征为触摸产生的电容变化与基础电容之间的比，如公式 1-5 所示。

$$C_{equal} = \frac{4{C_{RT}}^2}{4C_{RT} + C_t||C_h||C_g} + C_{RTt} \approx \frac{4{C_{RT}}^2}{4C_{RT} + C_t||C_g} + C_{RTt} \quad (1\text{-}4)$$

$$Sensitivity = \frac{C_{base} - C_{equal}}{C_{base}} \approx \frac{\frac{C_{RT} * C_t||C_g}{4C_{RT} + C_t||C_g} - C_{RTt}}{C_{RT}} \quad (1\text{-}5)$$

对于互感触摸，提高灵敏度的主要方式为：1）降低覆层的厚度；2）增大 Tx 和 Rx 之间的间距。要注意虽然增大 Tx 和 Rx 之间的间距能够减小C_{RT}，提高检测距离，变相地提高灵敏度，但如果手指无法同时覆盖 Tx 与 Rx，灵敏度反而会减小。

一般来说，对于自感与互感型电容检测，手指触摸产生的电容变化均在 1pF 左右。但自感的 base 电容（触摸前的电容值）一般会高于互感的 base 电容。因此相对来说互感的灵敏度更高，但也更易受噪声的影响。从应用的角度来看，自感型方案由于结构简单使用的更广，而互感型方案更多的用于矩阵按键，以使支持的按键数远超过电容触摸的 IO 口数（自感型按键数）。两种方案之间的比较如表 2-1 所示。

表 2-1 互感型与自感型方案比较

点	自感型电容检测	互感型电容检测
设计与布线难度	简单	复杂
是否受接地金属外壳影响	是	否
是否受浮地金属外壳影响	否	是
基于按键所能通过的 CNI 测试（EMC）	高	低
支持高密度按键	否	是
防水防雾性能	低	高
支持金属触摸	是	否
滚轮或滑条的性能	高分辨率	低分辨率
所能实现的接近感应距离	>10cm	<3-4cm
支持触摸屏功能	否	是

2.3 TI 的电容触摸技术

TI 的电容触摸感应技术 CapTIvate™以电荷转移采集为基础。该原理包括：1）将传感器电容C_{equal}充电；2）将累积电荷转移至内部采样电容C_{sample}，这两部分。此过程将不断重复，直至 C_{sample} 两侧电压达到内部比较器的触发电压V_{trip}。达到阈值所需的电荷转移次数直接表征C_{equal}的大小。当电容传感器被人手触摸时，C_{equal}发生改变，这意味着 C_{sample} 电压达到 V_{trip}所需的电荷转移次数发生了变化，MCU 通过比较前后电荷转移次数的差异，来感知触摸事件的发生。MSP430 内部采用电流镜来控制C_{sample}的输入电流和C_{equal}的放电电流之间的比例关系，以此来等效放大C_{sample}，从而拥有较大的量程。

对于自感检测，传感器电容C_{equal}等于 Tx I/O 口的对地电容，通过对地的充放电，将C_{equal}中的电荷转移到内部的C_{sample}中，如图 2-6 所示。互感检测的传感器电容C_{equal}等于 Tx 与 Rx 两 I/O 之间的互电容。互感检测的电路结构相比自感检测更为复杂，但实际上也是对地充放电。通过保持充放电前后对地电容两端电压不变的方式，来实现仅对互电容的电荷转移。

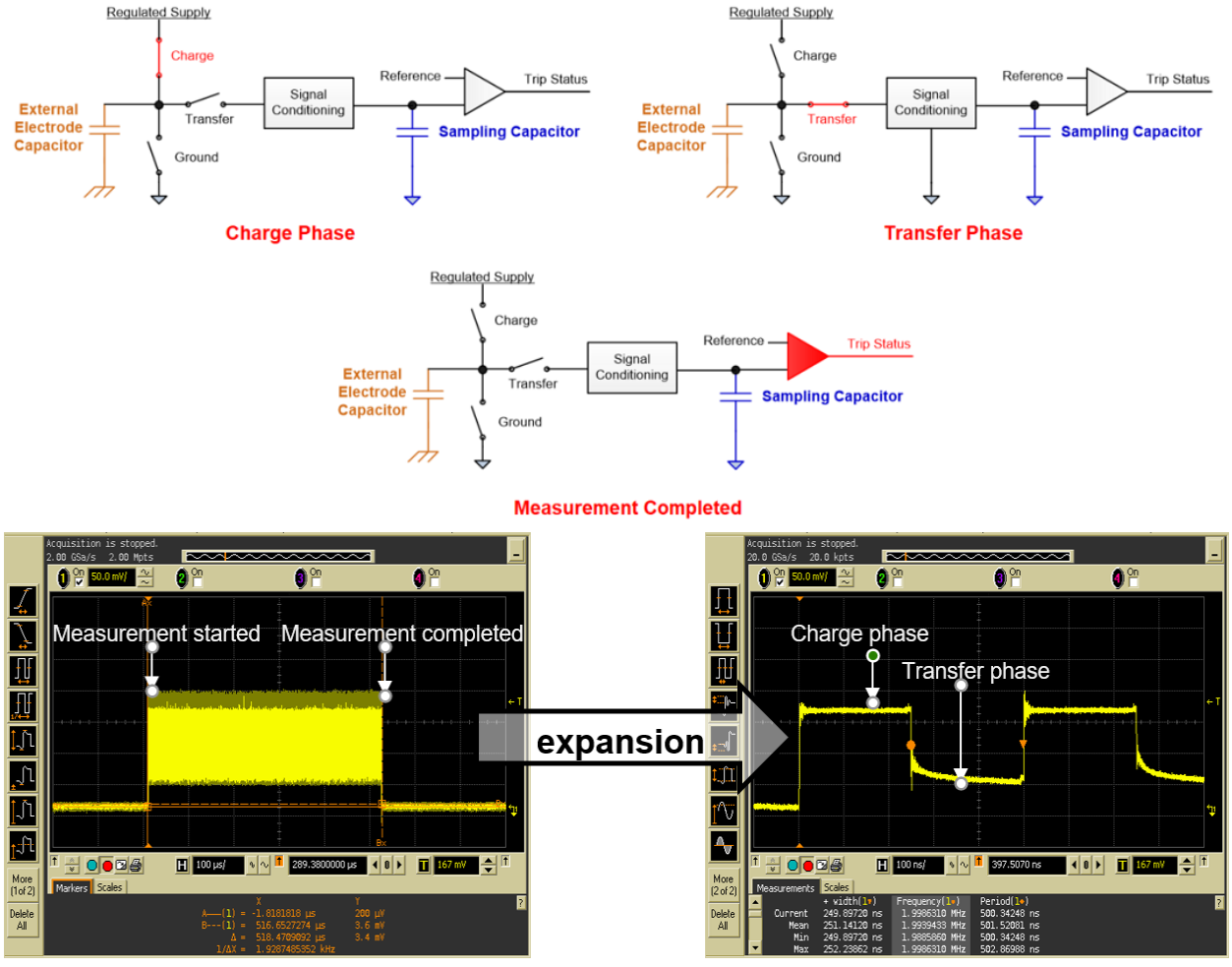


图 2-6 自感型电容检测原理

3 MCU 选型与功能评估

3.1 确定传感器需求

在开始以 MSP430 为开发对象的电容触摸项目之前，首先要了解选用的**传感器类型**，**传感器数量**，以及所占用的 **CAP I/O 口**。最后根据相应的需求选用合适的 MSP430。目前，电容触摸传感器的形态主要分为按键/接近感应，滑条/滚轮和触控屏。

3.1.1 按键/接近感应

按键与接近感应均属于零维传感器。两者实现的原理相同。不同的是接近感应往往具有更高的灵敏度，更大的传感器围成面积。

自感型按键：

从图 3-1 可知，自感型按键主要由连接到 CAP I/O 引脚上的覆铜，以及绝缘覆盖材料构成，而按键周围是用环形间隙隔开的接地覆铜。每个按键需要占用 **1** 个 MCU 的 CAP I/O 引脚。

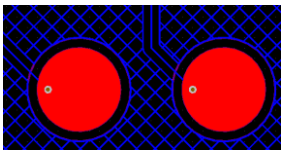


图 3-1 自感型按键 PCB 图

互感型按键：

在需要大量按键的应用中，可以将电容传感器排列成一个矩阵，每个按键将连接着两个 CAP I/O，一个 CAP I/O 将连着一行或一列的按键。如图 3-2 所示，内外两个方框型覆铜为 Tx 和 Rx。该方案支持多按键触摸。X 行*Y 列个按键需要占用 **X+Y** 个 MCU 的 CAP I/O 引脚。

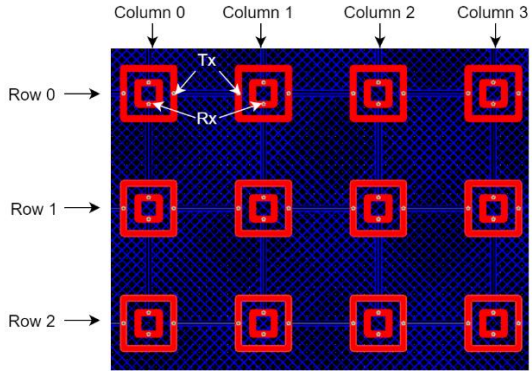


图 3-2 互感型按键矩阵 PCB 图

自感型接近感应：

通常，接近感应多选择自感方案。主要原因是自感比互感拥有更长的检测距离，更灵活的布线与结构，更高的信噪比。图 3-3 的 PCB 外圈展示的就是接近感应传感器。除利用 PCB 覆铜构成接近感应传感器外，也可以选择导线，甚至导电金属结构体。一个自感型接近感应传感器需要占用 **1** 个 CAP I/O 引脚。

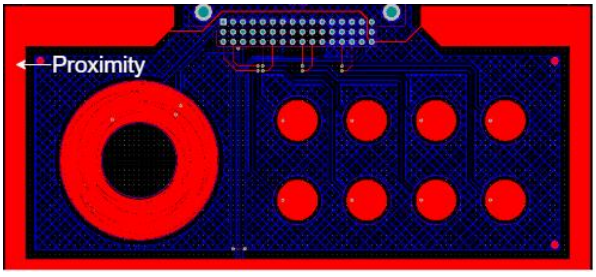


图 3-3 自感型接近感应 PCB 图

3.1.2 滑条/滚轮

滑条与滚轮均属于一维线性传感器，两者实现的原理与结构类似。由于互感的 base 电容较低，信号波动较大，导致分辨率要低于自感方案，采用较少，因此只论述自感类滑条/滚轮，如图 3-4 所示。滑条/滚轮的基础结构为按键，通过按键互相交错的方式，实现人手移动时，因为在相邻两个按键的接触面积不同，而能检测到渐变信号。此外，滑条首尾两个覆铜应通过走线连接到一起，属于同一个按键。
对于自感滑条和滚轮，至少占用 **3** 个 CAP I/O，软件所支持的滑条或滚轮的分辨率范围为 **3** 到 **65535**。实际占用的 CAP I/O 数量与所选分辨率，得根据实际需求进行调整。

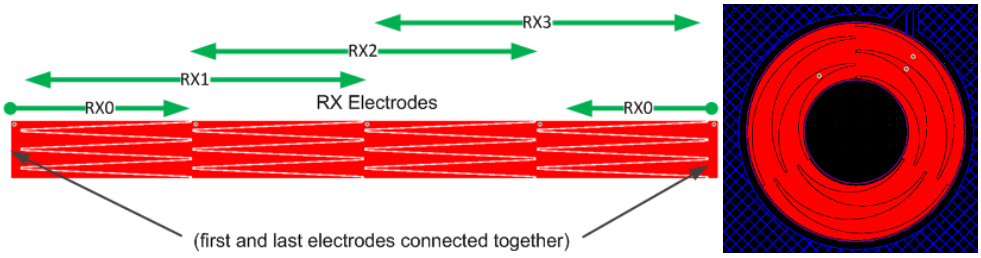


图 3-4 自感型滑条/滚轮示意图与 PCB 图

3.1.3 触摸面板

CapTIvate™触摸板是基于互电容的二维传感器，由菱形图案的行和列的交点形成检测点。其基础结构类似于矩阵按键，通过按键互相交错的方式，实现人手移动时能在两个相邻按键上检测到渐变信号。在下图 3-5 中，4 个 RX 4 个 TX 的交叉就形成了 16 个测量节点。单个触摸面板往往要占用 **8** 个以上的 CAP I/O。

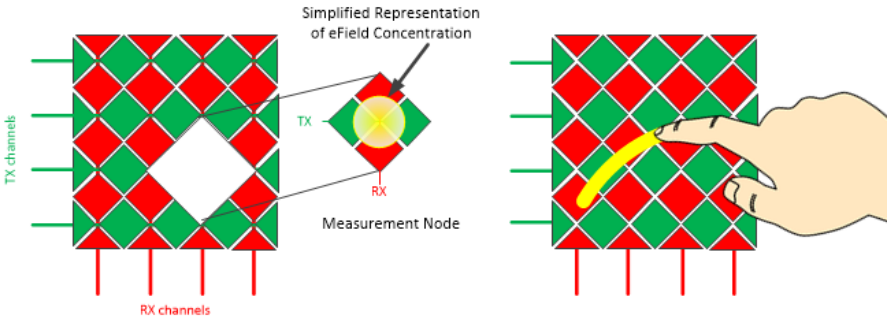


图 3-5 触摸面板示意图

3.2 MCU 选型

表 3-1 汇总了所有支持电容触摸的 MSP430，根据所需的 CAP I/O 与相应的外设，即可选出合适的 MSP430。要注意的是电容触摸的代码根据不同配置需要占据 **3-6k** 的空间，另外 I/O 口 3V 供电下的最大输出电流仅为 5mA。

表 3-1 MSP430 CapTIvate™ 系列器件选型表

Part number	Generation	CAP I/O	FRAM (kB)	RAM (KB)	ADC	Communication	Package Group
MSP430FR2512	Gen1	4	8	2	10 bit	1 UART; 1 I2C; 2 SPI	TSSOP|16, VQFN|20
MSP430FR2522	Gen1	8	8	2	10 bit	1 UART; 1 I2C; 2 SPI	TSSOP|16, VQFN|20
MSP430FR2532	Gen1	8	8	1	10 bit	2 UART; 1 I2C; 3 SPI	VQFN|24
MSP430FR2533	Gen1	16	16	2	10 bit	2 UART; 1 I2C; 3 SPI	TSSOP|32, VQFN|32
MSP430FR2632	Gen1	8	8	2	10 bit	2 UART; 1 I2C; 3 SPI	DSBGA|24, VQFN|24

MSP430FR2633	Gen1	16	16	4	10 bit	2 UART; 1 I2C; 3 SPI	DSBGA|24, TSSOP|32, VQFN|32
MSP430FR2672	Gen2	16	8	2	12 bit	2 UART; 2 I2C; 4 SPI	VQFN|32
MSP430FR2673	Gen2	16	16	4	12 bit	2 UART; 2 I2C; 4 SPI	VQFN|32
MSP430FR2675	Gen2	16	32	6	12 bit	2 UART; 2 I2C; 4 SPI	LQFP|48, VQFN|32, VQFN|40
MSP430FR2676	Gen2	16	64	8	12 bit	2 UART; 2 I2C; 4 SPI	LQFP|48, VQFN|32, VQFN|40

需要补充的是 MSP430 的 CapTIvate™系列主要分为两个系列，第一代产品 Gen1 主打的是性价比，而第二代产品 Gen2 拥有更丰富的外设，更强的 CapTIvate™模块的性能。具体区别如表 3-2 所示。

表 3-2 Gen1 与 Gen2 的对比表

特性	Gen1 device	Gen2 device	优势
电极充电电压	VREG 供电：1.5V	VREG 供电：1.5V DVCC 供电：2.7-3.6V	会影响充放电的电压，新增的 DVCC 模式能提高信噪比与抗干扰能力
输入偏置电流	无	有	提高了抗干扰能力
转换与抗噪处理	软件	硬件	提高响应速度，节省代码空间
信噪比典型值(-40°C- 25°C)	19:1–36:1	28:1–42:1	更高的信噪比与更低的温漂

3.3 EVM 开发板选择与评估

在前期电容触摸功能评估的过程中，所涉及的软硬件如表 3-3，与图 3-6 所示。要搭起整个开发链需要 GUI， IDE， SDK，Programmer，MCU，sensor board，总共 6 个组件，部分 MCU board 和 sensor board 是集成的。

表 3-3 CapTIvate™ 开发链

GUI	IDE	SDK	Programmer	MCU board	Sensor board	Supported Sensor type
CapTIvate™ Design Center	CCS /IAR	MSPWare	CAPTIVATE™-PGMR	CAPTIVATE™-FR2676	CAPTIVATE™-BSWP	Self-mode button/Wheel/slider/Proximity
				CAPTIVATE™-FR2633	CAPTIVATE™-PHONE	Mutual-mode button/Wheel/slider/Proximity
				BOOSTXL-CAPKEYPAD(FR2522)		12 mutual mode buttons and 1 proximity
				EVM430-CAPMINI (FR2512)		4 self-mode buttons

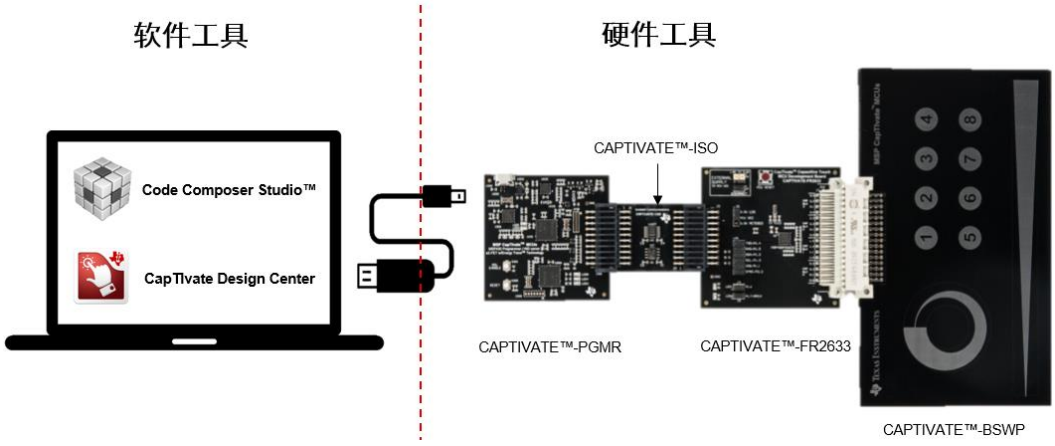


图 3-3 CapTIvate™ 开发链

开发链的软件部分为：

- GUI：用于生成代码。在线调试参数。
- IDE：用于烧录代码，添加额外功能。
- SDK（可选）：软件开发包，用于提供开发自定义功能所需的 driverlib，非必须安装。

开发链的硬件部分为：

- Programmer：支持 JTAG 与 USB HID to UART/I2C 通讯，用于烧录代码与电容触摸在线调参。
- MCU board： MSP430 单芯片的评估板。
    - CapTIvate™-FR2676 用于评估 Gen2 的解决方案，或单独评估 MSP430FR267x 系列。
    - CapTIvate™-FR2633 用于评估 Gen1 的解决方案，或单独评估 MSP430FR263x/FR265x 系列。
    - BOOSTXL-CAPKEYPAD 只用于评估特定的 12 个按键面板方案，或评估 MSP430FR2522。
    - EVM430-CAPMINI 只用于评估特定的 4 个自感型按键方案，或评估 MSP430FR2512。
- Sensor board：包含各种 sensor 的评估板。
    - CAPTIVATE™-BSWP 用于评估 TI 的自感型解决方案
    - CAPTIVATE™-PHONE 用于评估 TI 的互感型解决方案

对于其他 CapTIvate™相关资料，包括说明文档，教学视频，开发板，应用手册请参考 MSP430™ MCUs 开发手册的 5.2.1.1 章节。

TI 为所有的 EVM 板都配置了相应的 GUI 程，CCS 工程。默认安装目录为：C:\Users\UserName\CapTIvate™DesignCenter_x_xx_xx_xx\CapTIvate™DesignCenterWorkspace\TI_Examples。用户可参考附录的步骤完成 EVM 的评估，其中也包括具体的软件安装，以及 EVM 的使用调试。

4 机械结构与硬件设计

作为电容触摸设计的第一环，机械结构与硬件设计极大影响了电容触摸方案的灵敏度与抗噪能力。因此在进入本章前，建议先阅读第一章节，了解电容触摸的基本原理，以更好的理解所给出的建议。总的来说机械结构与硬件设计主要在结构限制，灵敏度，抗噪性能之间权衡。也就意味着往往需要根据测试反馈，反复进行修改和妥协。

4.1 机械结构设计

4.1.1 覆盖层设计

在电容触摸的设计中，电容型传感器均放置在覆盖层下，以减少环境影响，并防止手指直接接触产生的 ESD 问题。由公式 1-3 可知，覆盖材料的介电常数与厚度均会影响电容触摸的灵敏度。不同材料的介电常数如表 4-1 所示。

表 4-1 覆盖材料的介电常数

材料	ε_r
空气	1.00059
玻璃	4 - 10
蓝宝石玻璃	9 - 11
云母	4 - 8
尼龙	3
树脂玻璃	3.4
聚乙烯	2.2
聚苯乙烯	2.56
聚对苯二甲酸乙二酯(PET)	3.7
FR4 (玻璃纤维 + 环氧树脂)	4.2
PMMA (聚甲基丙烯酸甲酯)	2.6 - 4

对于由不同材料堆叠而成的面板，其触摸电容可以按以下公式等效计算：

$$C = \varepsilon_r \varepsilon_0 \frac{A}{d} = A\varepsilon_0 \sum \frac{\varepsilon_{ri}}{d_i} \tag{4-1}$$

A: 手指与传感器垫片覆盖层的接触面积。

d_i: 不同覆盖层的厚度。

ε_0: 空气介电常数。

ε_{ri}: 不同覆盖层的介电常数。

由于空气介电常数仅有 1 左右，由公式 4-1 可知，其存在变相增厚了面板，所以要尽量消除传感器和覆盖层间的空气。另外很多时候按键的灵敏度较低，主要因为面板过厚，一般推荐厚度为 3mm。因为导体会与电容充放电的电场模式相干扰，所以传导材料不能用做覆盖层，也不能使用含有金属微粒的油漆或粘合剂。推荐的典型合剂为 3M™ 200MP、 467MP 和 468MP。

4.1.2 传感器结构选择

根据实际应用的要求，可以使用各种材料来构建电容式传感器。下面将简单介绍常见的传感器结构。

4.1.2.1 覆铜型传感器（PCB）

该方案将蚀刻在 PCB 表面上的铜焊盘作为传感器，是实现电容触摸的最常见的实现方法。硬质 PCB 使用的最为广泛，适合于平面型前面板，而柔性 PCB 适合曲面型前面板。要注意的是柔性 PCB 较薄，这意味按键与地平面的距离较近，因此与硬质 PCB 相比要适当的减少网格地平面的密度。

4.1.2.2 导电垫圈/弹簧型传感器

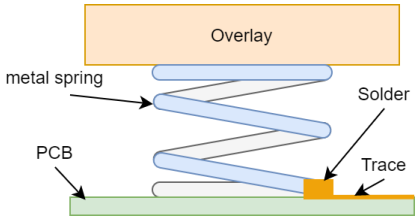


图 4-1 弹簧结构示意图

该方案通过导电材料垫圈/弹簧实现了对触摸按键空间上的伸展，主要适用于触摸前面板无法与 PCB 贴合的情况，也适用于曲面、斜面或不规则前面板方案。

4.1.2.3 电子印墨型传感器

电子印墨型传感器利用导电墨水构成电容型传感器。一般通过喷涂或印刷的方式实现与前面板的紧密结合，主要的特点为图案灵活，设计周期更短，适用于不规则前面板。另一种应用场景为 ITO 液晶显示屏，主要利用了 ITO 薄膜的透明性特点。由于电子印墨的薄膜电阻比铜大，所以等效串联电阻值较高，需要适当减少串联电阻。

4.1.3 机械设计检查清单

机械结构主要有三个方面会影响触摸灵敏度：覆层，外壳，周围器件。不合理的覆层设计主要影响的是手指触摸产生的电容变化量。对于自感型方案，外壳接地会提高产品的抗噪能力，但会增加对地电容，降低灵敏度，触摸面板周围存在大量铺地的 PCB 也会起到一样的效果。对于互感型方案，周围环境接地对灵敏度影响较小，但外壳不接地容易导致误触，尤其是设计中含有互感型接近感应传感器。电容触摸属于敏感器件同时也是干扰源，一般需采用网格地平面进行保护隔离。表 4-2 为机械结构设计的检查清单。

表 4-2 机械结构设计检查清单

编号	组件		建议
1	覆层	材料	避免使用导电材料和导电涂料
2		厚度	10mm 或更薄， 推荐 2-3mm，具体取决于材料和传感器尺寸
3		层叠	覆层与按键之间避免空隙，使用非导电粘合材料
4	金属外壳		在确保灵敏度的情况下，外壳尽量接地，提高抗噪能力
5	周围器件		带按键功能的 PCB 尽量远离产品内部辐射噪声源 带按键功能的 PCB 尽量远离敏感器件 带按键功能的 PCB 尽量远离大量铺地的 PCB

4.2 硬件设计

4.2.1 原理图设计

原理图设计主要关注的是 MCU 功能是否正常，以及 EMC 抗噪性能。表 4-3 为原理图设计检查清单。对于具体的设计方案请参考相应器件 datasheet，或者参考本手册 3.3 章节中 EVM 的原理图。

表 4-3 原理图设计检查清单

编号	分类	组件	建议
1	最小系统	复位与烧录电路	Reset pin 增加 47kΩ 上拉电阻， 1nF 下拉电容
2		供电电路	VCC 与 GND 增加 10μF 电容与 0.1μF 电容， 靠近 MCU 放置
3	电容触摸	VREG 滤波电路	靠近 VREG 引脚增加 1μF 对地电容， ESR ≤ 200mΩ
4		CAP I/O 上的串联电阻	靠近 MCU 引脚处增加 470Ω – 10k Ω 电阻，用于 ESD 保护与抗噪滤波
5		CAP I/O 引脚分配（如果可能）	建议使用 CapTIvate™ 设计中心“Auto-Assign”功能分配 CAP I/O 引脚。**注意：**除按键与接近感应外，滚轮、滑条、触摸面板对扫描 CAP I/O 有顺序要求，强烈建议在配

			置完按键后，使用 Auto-Assign”功能分配滚轮、滑条、触摸面板的 CAP I/O 引脚。	
6	EMC 抗噪（可选）	EMC 滤波电容	对于互感应用，在 RX 引脚上的串联电阻与传感器电极之间增加 68pF 对地电容	
7		TVS 二极管	在 CAP I/O 串联电阻与电极之间增加**低漏电，低寄生电容**的 3.3V TVS 管 在电源与外部连接线上增加通用 TVS 管	
8		共模电感/磁珠	在电源上按需增加共模电感与磁珠	
9	其他	I2C 通信线路上拉电阻器	增加 2.2kΩ 上拉电阻	
10		I2C 通讯引脚（**GUI 默认配置**）	MSP430FR25x2： P2.4: IRQ (OPEN DRAIN) P2.5: UCB0 I2C SDA P2.6: UCB0 I2C SCL	MSP430FR263x/253x/267x： P1.1: IRQ (OPEN DRAIN) P1.2: UCB0 I2C SDA P1.3: UCB0 I2C SCL
11		UART 通讯引脚（**GUI 默认配置**）	MSP430FR25x2： P2.0: UCA0 UART TXD P2.1: UCA0 UART RXD	MSP430FR263x/253x/267x： P1.4: UCA0 UART TXD P1.5: UCA0 UART RXD
12		引导加载程序 (BSL)	有关 BSL 引脚定义 请参阅相应器件 datasheet 中的 Bootloader 章节	
13		按键到 MCU 的接插件	会增加对地寄生电容，建议不使用	
14		测试点	增加 VCC, GND，通讯口的相关测试点	

对于 EMC 抗噪的设计，在前期评估时，建议按需预留好相应元件的焊接位置，按照最终测试结果选择性焊接。要重点说明的是 CAP I/O 上串联电阻的选择。因为串联电阻会同时影响 base 电容与触摸产生的电容变化，所以串联电阻本身并不会影响按键的灵敏度。由于会和寄生电容构成低通滤波器，较大的串联电阻反而能提升按键的抗噪能力。但要注意的是过大的电阻会影响电荷转移时间，CAP I/O 上的充放电波形如图 4-2 所示。这会直接导致按键的灵敏度降低，该情况在传感器 base 电容较大的自感型方案上尤其明显。此时可延长电荷转移周期，修改 GUI 中的“Frequency Divider”参数来解决。

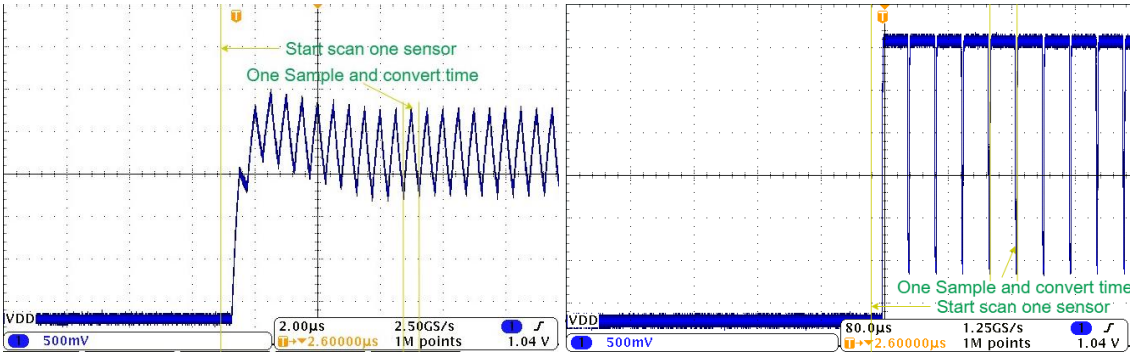


图 4-2 不完整和完整的电荷转移周期（GEN2）

4.2.2 PCB 布局

PCB 布局与机械结构设计的出发点相同，一个是提高信号强度，另一个减少 EMC 问题。表 4-4 为 PCB 布局检查清单。具体范例请参考 TI EVM 板 CAPTIVATE™-BSWP（自感型）和 CAPTIVATE™-PHONE（互感型）。

表 4-4 PCB 布局检查清单

编号	组件		建议
1	按键（自电容）	形状	实心圆形或方形
		尺寸	直径/边长为 10mm -12mm
		与接地网格覆铜的间距	按键放 PCB 顶层，接地覆铜距按键 0.5 倍覆盖层厚度
2	按键（互电容）	形状	方形或圆形，内 Rx，外 Tx
		尺寸	外直径/边长为 10mm 和 12mm RX 宽度为 0.5mm，Tx 宽度为 1mm，Rx 到 Tx 的间距为 0.5mm
		与网格覆铜的间距	按键放 PCB 顶层，接地覆铜距按键 0.5 倍覆盖层厚度

3	滑条/滚轮 （自电容）	形状	请参阅使用 OpenSCAD 脚本自动化电容式触摸滑块和滚轮 PCB 设计
		尺寸	电极宽 10-12mm，长取决于所需的触摸面积
		电极数量	大于等于 3 个电极，根据灵敏度分辨率需求增加电极
4	接近感应 （自电容）	形状尺寸	实心覆铜，空心型覆铜或导线，感应距离与围成面积成正相关
5	传感器走线	宽度	8 mil 或 PCB 制造商允许的最小厚度
		长度	尽量减小从传感器到控制器的长度
		走线注意点	与接地覆铜保持 10mil-20mil 间距 走线无锐角 减少过孔，过孔直径 10mil 远离其他触摸信号线，数字信号线与模拟信号线 4mm 以上。或选择垂直走线
7	接地覆铜		使用网格状覆铜（走线宽度 8mil,方格宽度 64mil，角度 45°） 自电容型设计，双面覆铜，按键正下方可选择性覆铜 互电容型设计，可选择只底面覆铜
8	串联电阻,TVS 管，EMC 滤波电容放置		放在 MCU 引脚 10 mm 范围内
8	防潮湿和耐液体性		自感型方案：建议减少接地覆铜，提高触摸灵敏度，以减少水产生的寄生电容变化。 互感方案：参考耐液体性电容式触摸键盘参考设计

灵敏度方面，与手指触摸宽度近似的按键与滑条滚轮，较窄较短的传感器走线能够增强触摸产生的电容变化。较少的过孔，较稀的覆铜能够减少对地电容的大小。但是接地覆铜也直接影响了 EMC 抗噪 能力，具体的铺设范围与网格密度需要根据实际的需求进行调整。由于电容触摸既是干扰源也是敏感器件，EMC 方面的考量上，除增加保护器件外，建议在布局时远离其他信号线或器件。

5 软件设计与调参

电容触摸的软件开发较为复杂，需要涉及 GUI，IDE 等软件工具，开发流程上会包括整定参数，功能开发。因此在进入本章之前，建议阅读本手册的 3.3 章节与附录，了解整个电容触摸的开发链，熟悉 GUI 和 IDE 的基本操作。对于 MSP430 器件本身的的相关技术资料，请参考 MSP430™ MCUs 开发手册。
实际电容触摸软件开发大体可分为如图 5-1 所示的五个部分。如果从开发时间的维度来看，软件调参耗时最长。本章节将会以这 5 个 phase 依次展开。

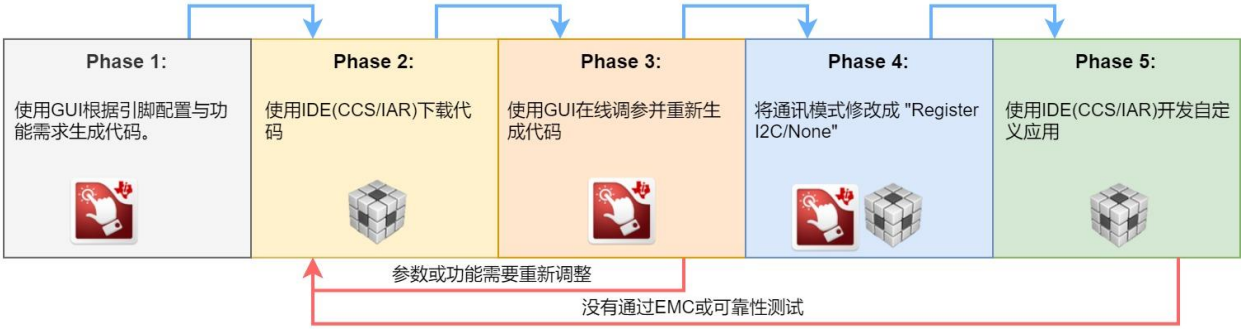


图 5-1 软件设计流程

5.1 CapTIvate™软件开发所需的概念

在进入介绍各个开发阶段之前，首先介绍一下 CapTIvate™ 的几个重要概念，帮助用户深入理解 CapTIvate™ 的工作方式，工作原理，以及参数间的关联性。更多信息请参考用户手册中的 Technology 章节。

5.1.1 CapTIvate 模块与 GUI 功能说明

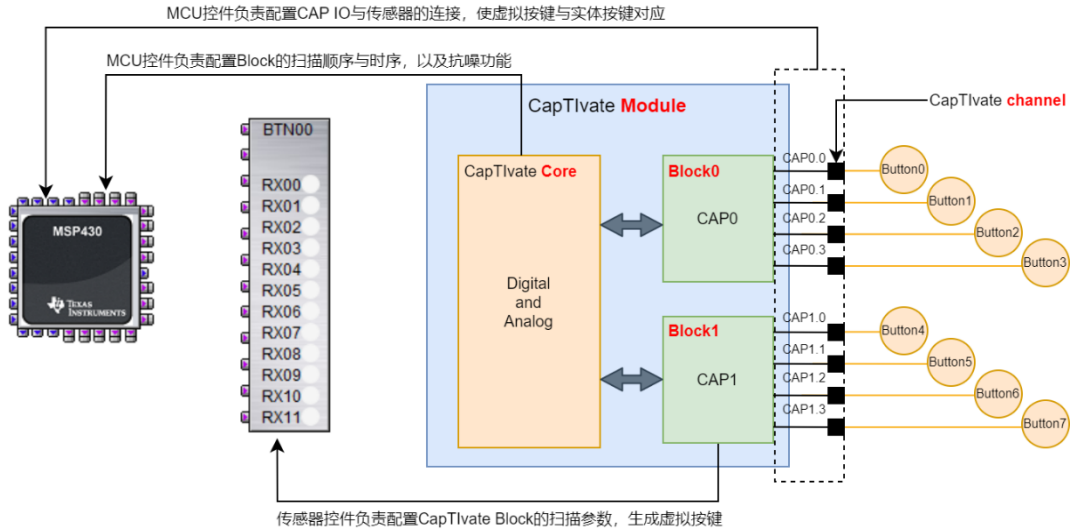


图 5-2 GUI 与 CapTIvate 模块的对应关系图

CapTIvate 模块由一个个 Block，以及一个 CapTIvate Core 组成。Block 负责对外部电容的扫描。CapTIvate Core 负责对 Block 的扫描控制，以及抗噪功能。要注意的是，物理上一个 Block 对应着四个 CAP IO。这也意味着一个 Block 下同时只能支持对一个 CAP IO 进行扫描，因此合理地配置按键的扫描顺序能够节省一定的时间。

GUI 最重要的是 MCU 控件与传感器控件。这两个控件负责对 CapTIvate Core 的配置，GUI 与 CapTIvate 模块的对应关系如图 5-2 所示。传感器控件主要负责对 Block 的控制。MCU 控制主要负责对 CapTIvate Core 的控制。

5.1.2 MCU 工作模式

CapTIvate™ MCU 总共有定义了两种工作模式，Active 模式和 Wake-on-Prox 模式，不同模式下的功耗请参考相应器件的 datasheet，这部分功能在 MCU 控件中进行配置。如图 5-3 所示，在 Active 模式下，CPU 会控制 Cap Peripheral 进行传感器扫描，然后 CPU 休眠进入 LPM 模式。等待“Acitve Mode Scan Rate ms”后 Cap peripheral 内的 Timer 会唤醒 CPU 重新扫描传感器，如此往复循环。

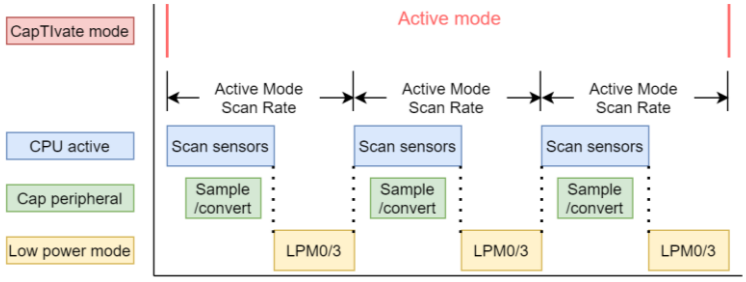


图 5-3 Active 模式下 MCU 状态示意图

Wake-on-Prox 模式主要利用了 Cap peripheral 可以脱离 CPU 工作的特性，通过长时间关闭 CPU 的方式，实现比 Active 模式更低的功耗。由于无 CPU 参与，意味着 Cap peripheral 寄存器配置无法更改，因此所扫描的接近唤醒传感器只能有一个。

其工作流程如图 5-4 所示，MCU 刚上电，会工作在 Active 模式，多个按键轮流扫描。如果没有触摸发生，等待“Inactivity Timeout”个传感器扫描周期后，由 Active 模式进入 Wake-on-Prox 模式。关闭 CPU，然后接近唤醒传感器按“Wake On Prox Mode Scan Rate”扫描间隔进行扫描。此时，MCU 可选择进入 LPM4 进一步降低功耗。

等待“Wakeup Interval”扫描时间间隔（一般几分钟）后，MCU 唤醒进入 Active 模式，对所有按键进行参数更新，以应对环境漂移问题。然后进入“Inactivity Timeout”计数，接着再次进入 Wake-on-Prox 模式。如此往复循环。

在 Wake-on-Prox 模式下，如果接近唤醒传感器检测到接近感应事件，或检测信号超过 Error Threshold，也会唤醒 MCU 进入 Active 模式。

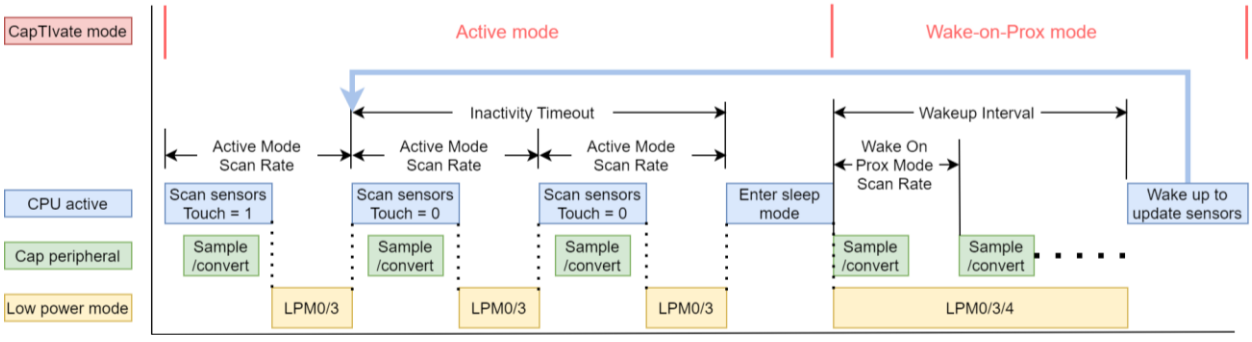


图 5-4 Wake-on-Prox 模式下 MCU 状态示意图

5.1.3 CapTIvate™中重要参数关系

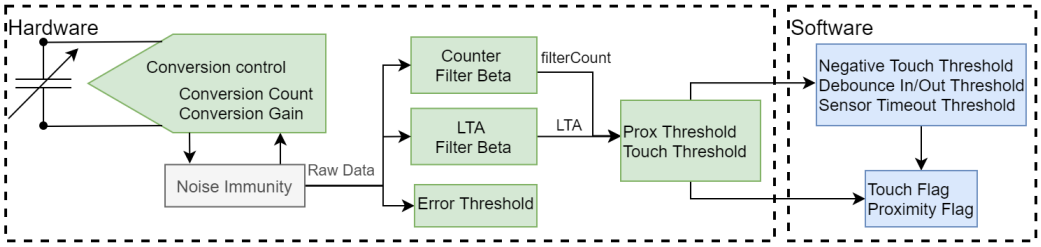


图 5-5 CapTIvate™中的参数关系示意图

整个 CapTIvate™中的参数结构如图 5-5 所示。总体分为硬件配置参数与软件配置参数，均在传感器控件中完成配置。由此也可了解当 CPU 不工作，CapTIvate™ MCU 处于 Wake-on-Prox 模式时，哪些参数能够自动更新。

首先采样模块通过检测外部的电容产生原始数据，Conversion Count 与 Conversion Gain 决定了采样模块的增益。接着数据通过可选的抗噪模块（采样频率扩频，过采样等功能）实现噪声滤除。要说明的是对于 GEN1，部分抗噪功能会由软件实现。

输出的数据首先会进行 Error Threshold 进行判断。接着通过不同强度的 IIR 滤波产生了 filterCount（默认强度 1），用于表征实时的电容变化，以及 LTA（Long time average，默认强度 7），用于表征环境的 base 电容。这里 filterCount 对应着 GUI 数据监测模块中出现的“Count”，LTA 对应着“LTA”。两者之间的差值 Delta 用于表征人手触摸产生的电容的变化。触摸产生的电容变化百分比与filterCount和LTA之间的关系如公式 5-2 所示。对于自电容检测，Delta 为正值，对于互电容检测 Delta 为负值。

$$Delta = \text{filterCount} - \text{LTA} \tag{5-1}$$

$$\Delta C_{touch} = \text{Gain}\left(\frac{1}{\text{LTA} + \text{Delta}} - \frac{1}{\text{LTA}}\right) * 100\% \tag{5-2}$$

整个信号变化过程为：上电时，校准函数会将环境 base 电容的等效值校准成 Conversion Count。也就是 LTA = Conversion Count。因此 Conversion Count 也可理解为决定系统分辨率的参数。此时由于无触摸 $\text{filterCount} = \text{LTA}$。当人手触摸时，由于 LTA Filter 和 Counter Filter 的滤波器强度不同，filterCount 会迅速产生变化，Delta 由 0 开始增加，当触发 Prox Threshold 时，LTA Filter 关闭，LTA 值保持不变。FilterCount 继续变化，Delta 继续增大，从而触发 Touch Threshold。当人手离开时，如果信号弱于 Prox Threshold，LTA Filter 打开，LTA 值开始刷新。因此 Prox Threshold 必须比 Touch Threshold 小，以确保 LTA 的工作机制正常。另外如果 Prox Threshold 设置的过大，人手触摸会导致 LTA 的变化也较大，会出现连续多次触摸后 LTA 漂移，系统无法正确响应后续触摸的现象。

严格来说 Prox Threshold 和 Touch Threshold 都是针对 Delta 所设的阈值。当触发时，系统相应的全局变量就被置位，该信号将送往后续的程序逻辑。要注意的是 Prox Threshold 是绝对阈值，而 Touch Threshold 是相对阈值，两者与 LTA 直接的关系如公式 5-4，5-5 所示。另外，由公式 5-2 可知，Delta 与ΔC_{touch}并无线性关系。这也是 Prox/ Touch Threshold 和 Prox/ Touch Threshold Percentage 不同的原因，因为后者直接表征触摸电容的变化。

$$|Delta| = \text{Prox Threshold} \tag{5-4}$$

$$|Delta| = \text{LTA} * \frac{\text{Touch Threshold}}{128} \tag{5-5}$$

其他 Threshold 将会决定整个 Sensor 的 Touch flag 与 Proximity flag 的置位事件。比如 MCU 在人手触摸时上电的问题，此时人手的触摸信号就会被计为环境的 base 电容，此时移开手指，就会产生一个负向信号，当负向信号达到 Negative Threshold 时，就会触发系统校准。 Debounce In/Out Threshold 会引入信号防抖动机

制来对抗噪声，一定程度上会延缓按键的响应。而 Sensor Timeout Threshold，用于防止环境变化触发 Prox Threshold 情况，该阈值触发后会系统会重新校准环境 base 电容。

5.1.4 CapTIvate™中的对象结构

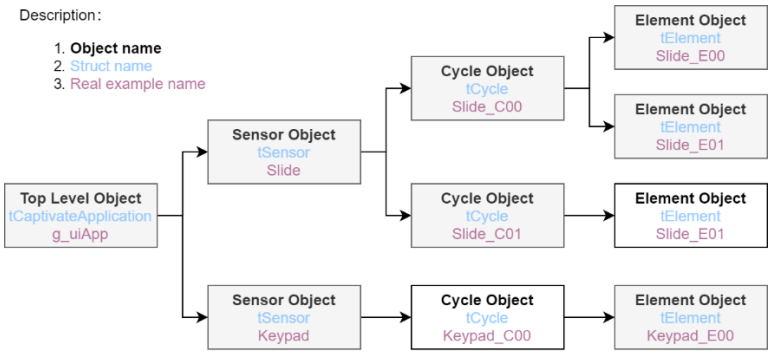


图 5-6 CapTIvate™中的对象结构态示意图

CapTIvate™ 中共有四种对象，Top Level，Sensor，Cycle，Element。图 5-6 展示了不同结构体之间的父子关系，同时给出了对象名称，结构体名称，实际的命名。

Top Level 用于描述 MCU 的工作模式，属于 MCU 的抽象概念，它对应着 GUI 中的 MCU 控件，而 MCU 控件所配置的参数就挂着 Top Level 结构体下，这部分内容在本章的 5.1.1 小节进行了论述。Sensor，Cycle，Element 对应着 GUI 中的传感器控件，传感器控件所配置的参数就挂下这三个结构体下，这部分内容也在本章的 5.1.2 小节进行了论述。

Sensor 对应按键面板，接近感应，滚轮，滑条，触摸面板，接近感应这 5 种传感器形态，属于实体传感器的抽象概念，因此一个系统里会有多个 Sensor 对象。

由于 CapTIvate™ MCU 的一个电容检测模块对应着 4 个 CAP IO，因此需要通过片选的方式依次选择所扫描的 IO 口，因此 Cycle 就对应着每一轮扫描。

Element 是一个个实体按键的抽象概念。所有的 Sensor 模式，按键面板，接近感应，滚轮，滑条，触摸面板，其本质都是基于一个个按键所测得的信号，然后软件上进行处理，得到相应的零维，一维，二维信息。因此按键是传感器中最基础的结构，Element 也是对象中最基本的对象。

5.1.5 CapTIvate™ MCU 通讯模式

CapTIvate™支持的通讯模式共有“UART”，“BUCK_I2C”，“Register_I2C”，“None”四种模式。UART 默认波特率为 250Kbps，I2C 的默认地址为 7-bit 0x0A。

- NONE：不使能通讯模块。需要客户通过调用库函数自己开发通讯协议来与 Host MCU 通讯，若想进一步节省代码空间可以选择 MSPWare 中的寄存器级代码作为参考进行开发。
- UART/BUCK_I2C：用于通过 UART/I2C 与 CapTIvate™ Design Center 通讯，以实现 GUI 对电容触摸的调参。支持更丰富的通讯协议，总线会传输大量的底层信息，一般用于在线调参。UART 通讯模块占用代码约 1.5k，BUCK_I2C 通讯占用代码约 2k。
- Register_I2C：用于通过 I2C 与 Host MCU 通讯以实现 Host MCU 对电容触摸的调参。支持的调试命令较少，占用代码约 1.4k。

整个使用逻辑是，用户先选用“UART/BUCK_I2C”通讯模式配合 GUI 完成参数调试，然后将通讯模式改为“None”，自行开发上位机通讯的协议。

Phase 1：GUI 配置

请按照如下的说明对相关参数进行配置，更多的信息请参考应用手册中 Design Center GUI 章节。

5.1.6 GUI 主界面操作

GUI 的主界面如图 5-7 所示，该界面的操作主要是创建工程，生成对应的虚拟传感器。所需配置的参数如表 5-1 所示。其他参数将会在 Phase2 中涉及。

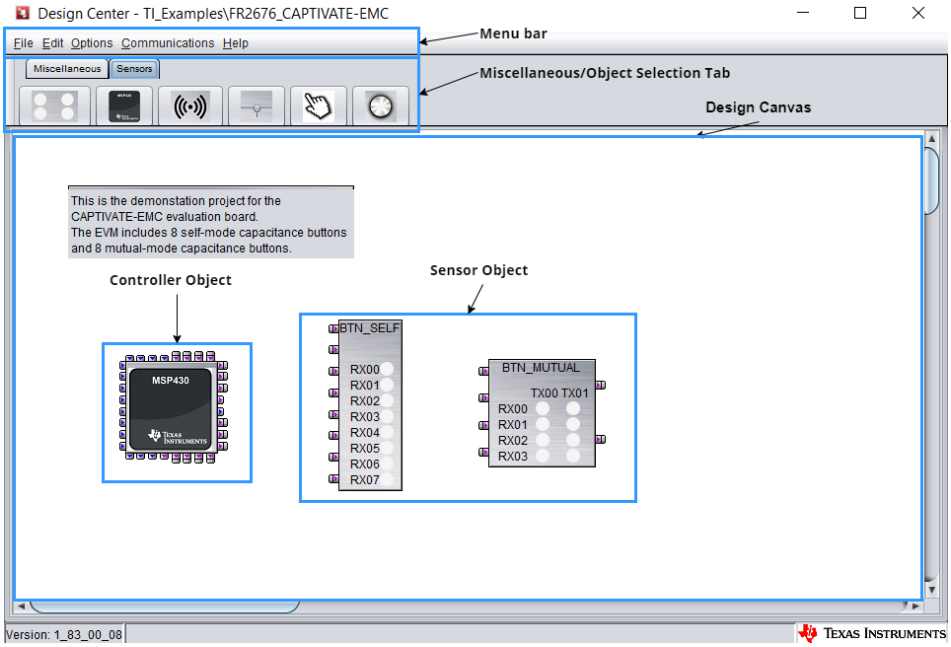


图 5-7 GUI 主界面

表 5-1 GUI 主界面中需设置参数

图中选框	操作目录	操作内容	推荐值	说明
Menu bar	File	新建 GUI 工程，或导入例程进行修改	无	
	Options	点击“Features”选择“Advanced”来进入高级调参模式	“Advanced”模式	
	Help	“Topic”按键可直接打开电容触摸的使用手册	无	
Object Selection	Sensors	将 MCU 控件与所需传感器控件，拖动至 design canvas	无	

5.1.7 传感器控件配置

MCU 控件和传感器控件是 GUI 中最重要的两个部分，涵盖了代码生成，在线调参，传感器信号监控等功能。首先我们介绍传感器控件所需的配置，如图 5-8 所示。这部分主要配置传感器组合的名称，数量与模式，将虚拟按键对应到虚拟引脚。所需配置的参数如表 5-2 所示，所属选框中的编号与图 5-8 中的对应，其他参数将会在 Phase2 中涉及。

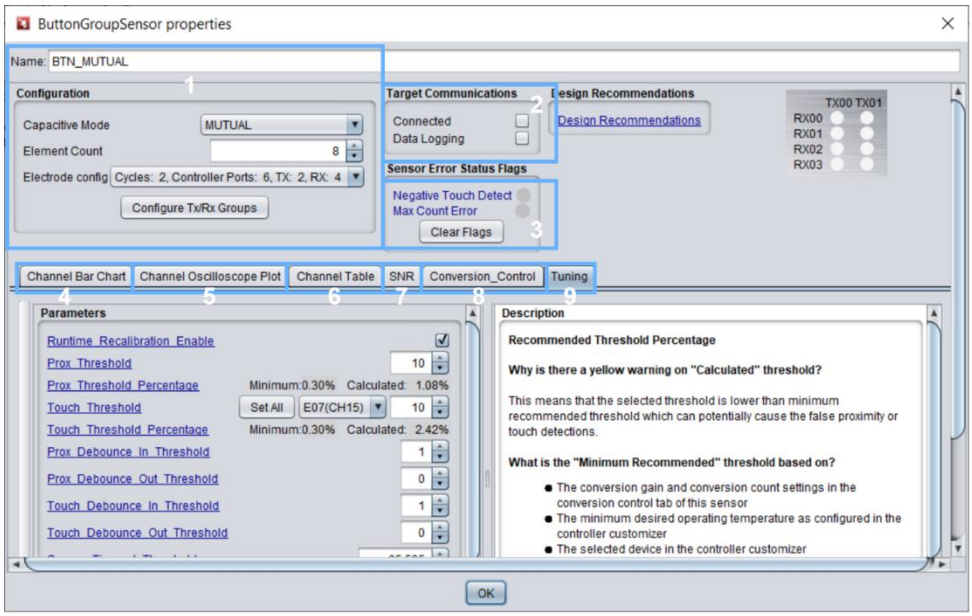


图 5-8 传感器控件界面

表 5-2 传感器控件中需设置参数

图中选框	操作目录	操作内容	推荐值	说明
编号 1	Name	配置传感器组合 Sensor 的名称，如 Keypad	可用默认值	
	Capacitive Mode	选择该 Sensor 属于互感电容模式还是自感电容模式	无	
	Element Count	用于选择该 Sensor 所属的 Element 个数，即占用多少个按键	无	
	Electrode config	在互感模式下，**根据实际设计**的行列数，选择相应的 TX 和 RX	无	
	Configure Tx/Rx Groups	用于了解各个虚拟的 Element 所占用的 TX 和 RX，以将虚拟按键对应到虚拟引脚	无	

5.1.8 MCU 控件配置

MCU 控件界面如图 5-9 所示。这部分主要配置 MCU 型号（编号 1）；MCU 工作模式与低功耗参数（编号 2，5，8），抗噪功能（编号 9）；将虚拟引脚映射到实际引脚（编号 4）。所需配置的参数如表 5-3 所示，所属选框中的编号与图 5-9 中的对应，其他参数将会在 Phase2 中涉及。

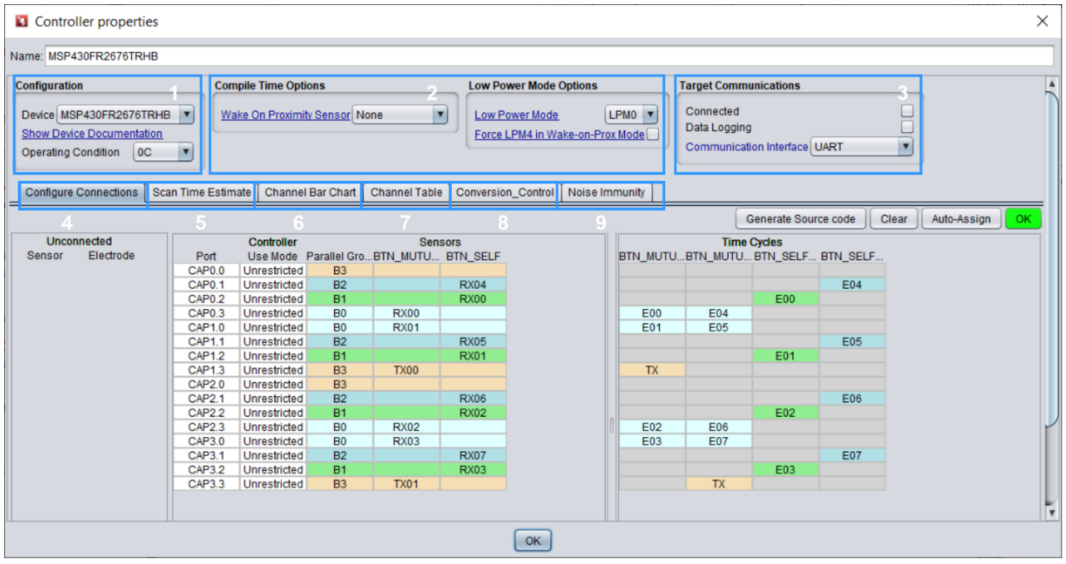


图 5-9 MCU 控件界面

表 5-3 MCU 控件中需设置参数

图中选框	操作目录	操作内容	推荐值	说明
编号 1	Device	选择对应的 device 以及封装	无	
	Operating Condition	选择实际的工作温度	25C	温度会影响器件的噪声，影响传感器控件中的 Threshold Percentage
编号 2	Wake On Proximity Sensor	若有超低功耗需求选择用作接近唤醒传感器的 Sensor	可不使能	
	Low Power Mode	根据需求选择相应的低功耗模式	可用默认值	LPM3 比 LPM0 功耗更低，但此时无法主动接收 UART 数据，只能发送数据
	Force LPM4 in Wake-on-Prox Mode	使能后 MCU 在 Wake-on-Prox 模式下功耗更低	可不使能	使能后，计时时钟源为 VLO clock，计时误差较大
编号 3	Communication Interface	根据实际设计选择 UART 或 BUCK_I2C 用于与 GUI 通讯	无	所使用的通讯引脚和 IRQ PIN 请参考本手册的 4.2.1 章节
编号 4	Configure Connections	根据实际的按键设计，将虚拟 Tx 和 Rx 引脚映射到 CAP IO 上	可直接点击 Auto-Assign	除按键与接近感应功能外，建议选择 Auto-Assign 功能
编号 5	Scan Time Estimate	自动计算 Active 模式下，按键的扫描总时长	无	用于评估最短的“Active Mode Scan Rate ms”
编号 8	Active Mode Scan Rate ms	根据需求设置 active 模式下，传感器的扫描时间间隔	可用默认值	
	Wake On Prox Mode Scan Rate ms	设置 Wake-on-Prox 模式下，接近唤醒传感器的扫描时间间隔	可用默认值	时间越短接近唤醒的扫描速度越快，但功耗越高
编号 9	Enable Noise Immunity	使能 noise immunity 功能	可不使能	用于提高抗噪能力，但代码量会更大，扫描时间会增长
	Frequency Hopping	使能多频率采样功能	可用默认值	用于提高抗噪能力，但代码量会更大，扫描时间会增长
	Self/Mutual mode Oversampling	调整过采样等级	可用默认值	能提高抗噪水平，但过采样会增加单个周期的扫描时间
	Dynamic Threshold Adjustment	使能接近感应与触摸阈值的动态调整功能	使能	使能后能保证灵敏度会随着噪声等级而变化，但遇到突发型噪声时，可能出现阈值变化落后于噪声变化的情况

要说明的是在传感器控件中的“Configure Tx/Rx Groups”可以知道虚拟按键所对应的 Tx 和 Rx，而 MCU 控件中的“Configure Connections”，可以知道实体按键所对应的 CAP x.x，由此可以将实体按键与虚拟按键对应起来。

配置完后，如图 5-10 所示，当“Configure Connections”目录中 OK 按键变绿时，且确定虚拟引脚与实际引脚映射正确，即可点击“Generate Source Code”，即可生成代码。由于生成新工程会**删除**当前文件夹下的所有文件，如不放在默认目录，请新建一个文件夹，再放入工程。

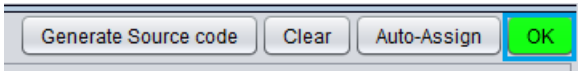


图 5-10 OK 按键示意图

5.2 Phase 2：下载代码

下载工具支持 CCS 和 IAR，具体下载工程的方法请参考使用手册中的 Loading and running generated projects 章节。如果出现无法导入的问题，请查看 CCS workspace 下是否有**相同名称**的工程。对于 IAR，请确保下载最新的版本以支持所有电容触摸相关的 MSP430。如果时第一次下载，需要等待 10~20 分钟左右时间。

5.3 Phase 3：调整参数

电容触摸最为重要的是参数整定，这一小节将具体介绍所有常用的参数。

5.3.1 参数调整逻辑关系与调参方法

调参所涉及的步骤可大致分为四个部分。数据监测，灵敏度参数，系统可靠性，响应速度与功耗调整。整体思路为首先调整灵敏度，接着调整触摸的可靠性，最后调整响应时间与功耗，整个调参的过程中利用数据监测功能评估参数调整的效果。

参数调整的逻辑框图 5-11 所示。每个参数调整的步骤往往需要反复进行。要注意的是，整个参数整定的难度与初期的机械与硬件设计有着极大的关系，良好的机械与硬件设计意味着触摸信号相比噪声越强，系统可靠性参数调整也越简单，同时也意味着可以大大提高响应时间。

整个开发过程中两个关键节点是：触摸产生的信号变化是否高于最小阈值；系统能否通过可靠性测试。前一个表征着触摸产生的信号量，如果信号量太小，系统可能会受噪声的影响而出现误触。后者表征着系统能否在特定条件下正常工作。以上任何一个条件满足不了都意味着需要修改机械与硬件。

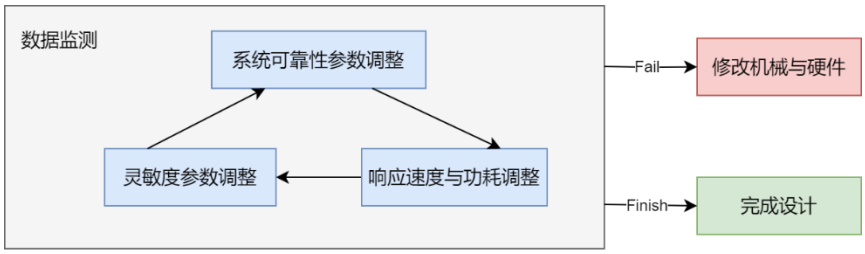


图 5-11 参数调整的逻辑框图

在调整参数之前，首先在 GUI 主界面，点击 Communications->Connect 将 MCU 连接到 GUI，接着按照下面几小节调整相关参数。而在线调参过程中主会用到传感器控件中的以下几个按键来控制参数的更新与复位，如图 5-12 所示。相应的功能见表 5-4。如果没有 GUI 没有响应，请重新插拔电源，同时确认一下 HID Bridge UART/I2C 是否连接正确。

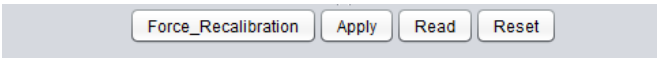


图 5-12 在线调参按键

表 5-4 在线调参按键说明

序列	按键	功能说明
1	Force_Recalibration	强制系统校准，当按键功能不正常时使用
2	Apply	将 GUI 上的参数传输到 MCU 中
3	Read	读取 MCU 中当前的相关参数
4	Reset	将 GUI 与 MCU 中的触摸参数复位至 GUI 的初始配置值

5.3.2 数据监测

传感器的数据监测功能同时分布在 MCU 控件与传感器控件中。两者的区别为 MCU 控件中可观测所有 Sensor 和 Element 的数据，而传感器控件中只展示自身所包含的 Element 数据。由于实际调参时主要针对传感器控件进行调参，因此本小节只论述传感器控件中的相关功能。所涉及的功能如表 5-5 所示，所属选框中的编号与图 5-13 中的对应。

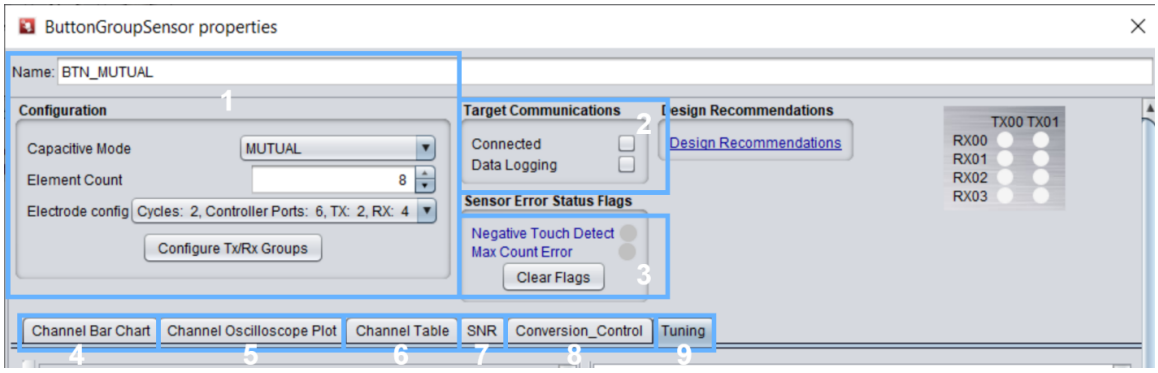


图 5-13 传感器控件界面

表 5-5 传感器数据监测功能表

图中选框	操作目录	操作内容	说明
编号 2	Data Logging	使能后将按键实时采集的数据存在 GUI 工程的创建目录下	一般用于记录一段数据后，评估系统可靠性
编号 4	Channel Bar Chart	通过柱状图可观查该 Sensor 所属的各个 Element 通道的信号变化情况	可直观地观测信号与触摸阈值，接近感应阈值的关系，主要应用于灵敏度调整
编号 5	Channel Oscilloscope Plot	展示该 Sensor 所属的各个 Element 通道数据与时间关系图	可观测一段时间内的信号变化，主要用于系统可靠性调整
编号 6	Channel Table	通过表格的方式观查各 Element 通道的实时信号变化	可直接观测到数据，一般用于评估阈值设置的合理性，是否留有足够裕量
编号 7	SNR	多次采样评估整个系统设计与阈值设置的合理性	

5.3.3 灵敏度参数调整

灵敏度参数的调整主要是根据实际检测需求选择合适的系统增益与合适的阈值。无限增加 CapTIvate™系统的增益，放大触摸信号的同时也会放大噪声，最终是只是增加了系统的分辨率，但信噪比本身并无变化。由于较大的增益会导致转换时间增长，因此不建议将整个系统的增益调的过高。这部分所涉及的功能如表 5-6 所示，所属选框中的编号与图 5-13 中的对应。

表 5-6 灵敏度参数调整表

图中选框	操作目录	操作内容	推荐值	说明
编号 8	Conversion Count	设置对应 Sensor 的电容等效设置值	<1000	决定系统分辨率，数值越大意味着系统增益越大
	Conversion Gain	设置对应 Sensor 的系统增益	100	数值最小为 100，越大系统增益越低
	Frequency Divider	设置电荷采样与转换的分频系数	可用默认值	如果寄生电容过大，需要抓取波形检查分频系数是否需要调整
编号 9	Prox Threshold	设置接近感应阈值	使 Prox Threshold Percentage 大于 Minimum 值	触摸信号绝对变化的阈值，所有 Element 共享一个接近感应阈值
	Touch Threshold	设置触摸阈值	保证优先触发接近感应阈值	触摸信号百分比变化的阈值，**同时调整要选择“select all”**，所有 Element 拥有独自的触摸阈值
	Desired Resolution	滑条，滚轮或触摸面板的分辨率	<100	过大的分辨率容易受噪声影响

5.3.4 系统可靠性参数调整

系统可靠性参数调整主要从软件的角度降低噪声影响，处理异常模式。这部分所涉及的功能如表 5-7 所示，所属选框中的编号与图 5-13 中的对应。

表 5-7 系统可靠性参数调整表

图中选框	操作目录	操作内容	推荐值	说明
编号 8	Modulation Enable	设置抖频功能	可不使能	对于自感模式使能抖频功能，互感不使能
编号 9	Pro/Touch Debounce In Threshold	设置系统判断发生接近感应或触摸事件的延迟时间	1-3	响应延迟以 Active Mode Scan Rate ms 为单位
	Pro/Touch Debounce out Threshold	设置系统判断无接近感应或触摸事件的延迟时间	1-3	响应延迟以 Active Mode Scan Rate ms 为单位
	Sensor Timeout Threshold	设置连续触发接近感应时系统复位校准的时间。	建议根据时间需求设置	时间以 Active Mode Scan Rate ms 为单位，可防止由环境变化引起的连续性正向信号漂移问题
	Negative Touch Threshold	设置电容等效“Count”值负向变化的阈值	默认值	触摸信号绝对变化的阈值，可防止由触摸上电引起的负向信号变化问题
	Counter Filter Beta	设置对电容等效“Count”值进行 IIR 滤波的强度	1-3	过大会影响响应速度
	LTA Filter Beta	设置对 LTA 进行 IIR 滤波的强度	7	不建议改小，会导致多次触摸后出现无响应问题
	Error Threshold	设置电荷转换输出的最大值	conversion count x 1.25	用户需根据标志位 bMaxCountError 来自行增加处理逻辑

一般来说，噪声源会来自于 CapTIvate™模块本身的噪声，MCU 内部的数字信号，电源，电容触摸引脚处的输入信号等。合理的参数能够减少噪声的影响，但往往需要反复调整，反复调整的对象主要是“Pro/Touch Debounce In Threshold”，“Counter Filter Beta”功能。同时确保使能了 MCU 控件中 noise Immunity 功能。

在进行系统可靠性参数调整时，建议使能 datalogging 功能，或其他方式将数据导出，通过长时间的数据比较来观察参数的调整效果。另外可以通过函数 CAPT_getGlobalFilteredNoiseLevel()来观察 MCU 所监测到的系统噪声强度。

5.3.5 响应速度与功耗调整

影响响应速度的一个是扫描时间，另一个是触摸信号的迟滞效应。这部分所涉及的功能如表 5-8 所示。

表 5-8 响应速度调整参数表

控件名	子目录	功能	说明
MCU 控件	Conversion_Control	Active Mode Scan Rate ms	传感器的扫描周期设置
	Noise Immunity	Enable	影响实际的扫描时间长度
传感器控件	Conversion_Control	Conversion Count	参数过大意味着电荷转移的次数增加，影响实际的扫描时间长度
	Tuning	Debounce In/Out Threshold	影响信号变化速度
		Counter Filter Beta	影响信号变化速度

另外不建议主机采用 I2C 固定周期轮询的方式来获取 MSP430 的按键状态信息，因为这要保证 MSP430 的传感器扫描速度要两倍于主机的轮询速度，间接降低了整个系统的按键响应速度的上限。

扫描时间越长，扫描频率越快意味着功耗越高。建议使用 CAPTIVATE™-PGMR 提供的 Energytrace 功能，通过观测时间与功耗的关系进行系统调优，注意所使用的电源引脚为 **3V3 Metered**。如果对功耗有较高要求，建议使能 LPM3 模式与 Wake-on-Prox 模式。

EnergyTrace： 用于优化功耗的代码分析工具，可用于测量和显示应用程序的功耗情况，包括软件和硬件部分。软件部分集成在 CCS 和 IAR 中。

- 产品页面：EnergyTrace Technology

- 用户手册：ULP Advisor™ Software and EnergyTrace™ Technology

5.4 Phase 4：修改通讯模式

在调试完电容触摸参数后，下一步是使用其他上位机尝试与 MSP430 创建连接。Communication influence 建议选择 “None”，然后利用 GUI 重新生成代码，或直接修改 CapTIvate_config->CAPT_UserConfig.h-> CAPT_INTERFACE 的宏定义。

5.5 Phase 5：开发自定义应用

开发自定义的应用意味着需要了解 CapTIvate™ 的代码构成，这一部分将重点涉及常见的自定义需求。演示说明

5.5.1 程序架构

首先将展示程序的组成，如图 5-14 所示。更多相关说明，可以参考 Starting from Scratch with the Starter Project。

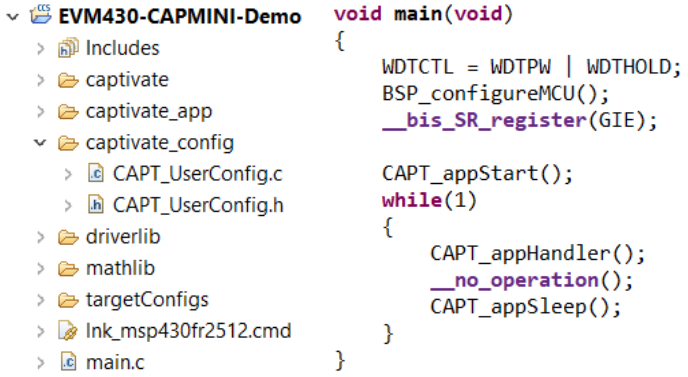


图 5-14 程序架构与主函数示意图

软件工程的目录结构如图 5-14 的左图所示，这部分所涉及的文件夹或文件如表 5-9 所示，其中对用户来说最核心的就是 CapTIvate_config 文件夹。如图 5-15 所示，GUI 与 CapTIvate_config 中的参数对应关系可以点击 GUI 中相应参数，阅读“Affected Software Parameters”了解。

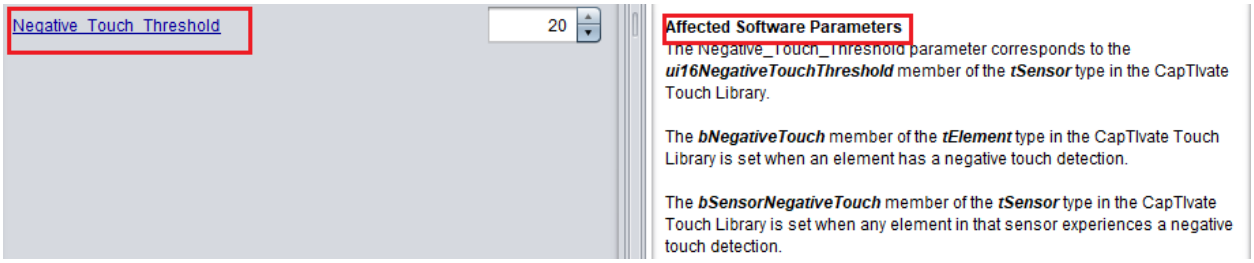


图 5-15 GUI 与代码参数对应关系说明

表 5-9 软件工程的组成表

序列	目录	说明
1	CapTIvate™	电容触摸相关外设的底层驱动库，包含电容检测与通讯配置
2	CapTIvate™_app	电容触摸的顶层逻辑代码，包含系统初始化，扫描与校准逻辑
3	CapTIvate™_config	GUI 生成的配置参数
4	driverlib	MSP430 外设驱动库
5	mathlib	定点计算库
6	targetConfigs	MCU 型号选择与烧录配置（无需改动）
7	lnk_msp430fRxxxx.cmd	链接器命令文件，给出了程序空间和数据空间的设置

8	main.c	主函数

主函数的程序结构如图 5-14 的右图所示，这部分所涉及的功能函数如表 5-10 所示。更多函数的说明，请参考 API Guide（C:\ti\msp\CapTIvateDesignCenter_x.xx.xx.xx\CapTIvateDesignCenter\docs\api_guide）。

表 5-10 重要程序构成表

序列	目录	说明
1	WDTCTL = WDTPW | WDTHOLD	禁止 watchdog
2	BSP_configureMCU()	配置通讯 IO 引脚，配置时钟
3	__bis_SR_register(GIE)	使能全局中断
4	CAPT_appStart()	电容触摸模块初始化与校准，同时使能通讯外设
5	CAPT_appHandler()	电容触摸扫描与模式切换
6	__no_operation()	CPU 等待一个周期
7	CAPT_appSleep()	CPU 进入休眠，等待周期性中断唤醒
8	CAPT_initUI()	电容触摸模块初始化，与使能通讯外设
9	CAPT_calibrateUI()	电容触摸模块校准
10	CAPT_updateUI()	更新所有传感器的参数
11	I2CSlave_setRequestFlag()	用于触摸时产生高电平脉冲，提醒上位机出现触摸事件

5.5.2 电容触摸状态与参数的读取与处理

正如 5.1.3 章节所描述的， Sensor 结构体是实际传感器模块的抽象，Sensor 结构体下挂着 Cycle 结构体，Cycle 结构体下挂着 Element 结构体。用户可以直接对 CapTIvate_config->CAPT_UserConfig.c 中声明的结构体进行访问以得知各个模块的状态，同时也能实时根据需求修改结构体中的参数。最常用的结构体是 Sensor 和 Element。Sensor 可直接全局访问，而 Element 结构体需将其声明为全局变量再进行访问。

Sensor 结构体最常用的变量如表 5-11 所示。

表 5-11 Sensor 结构体中常用变量表

序列	变量名称	类型	说明
1	. bSensorProx	Bool	Sensor 下的 Element 有无检测到接近感应事件
2	. bSensorTouch	Bool	Sensor 下的 Element 有无检测到触摸事件
3	. bSensorPrevTouch	Bool	Sensor 下的 Element 在上一个扫描周期有无检测到触摸事件
4	.pSensorParams-> SliderPosition.ui16Natural	uint16_t	获取当前滚轮或滑条的位置
5	.pvCallback	void (* pvCallback)(struct tSensor *)	可以将回调函数链接到传感器上，用来处理 Sensor 在扫描时发生的状态变化，每次按键扫描均会调用

Element 结构体最常用的变量如表 5-12 所示。

表 5-12 Element 结构体中常用变量表

序列	变量名称	类型	说明
1	. bProx	Bool	该 Element 有检测到接近感应
2	. bTouch	Bool	该 Element 有无检测到触摸

处理按键的状态变化可通过 Sensor 所属的回调函数实现，具体的使用说明可参考 Register a callback function★。而自定义工程的例子，可参考 EVM430-CAPMINI 对应的 CCS/IAR 例程 EVM430-CAPMINI-Demo★。

5.5.3 通讯功能的实现与自定义

与主机 MCU 通讯建议利用提供的 UART 与 I2C 库函数进行开发。首先 GUI 生成的默认代码，均会在 BSP_configureMCU()函数中完成对 UART（eUSCI_A0）与 I2C（eUSCI_B0）的 IO 配置与时钟配置。因此只需对通讯模块进行配置，与编写通讯协议即可。如果需修改通讯接口，注意查看 datasheet 是否要额外修改 SYSCFG3 寄存器，进行通讯口的重定向。

基于 UART 的通讯开发：

- Communication Interface 配置：NONE
- 时钟配置：BSP_configureMCU()
- IO 配置：BSP_configureMCU()
- UART 外设配置与中断使用：Driverlib UART loopback 例程
    - 说明：
        - 波特率参数配置工具：MSP430 USCI/EUSCI UART Baud Rate Calculation
        - 不同系统时钟源的频率见 CapTIvate_app->CAPT_BSP.h

基于 I2C 的通讯开发共有两个方案可供参考，方案占用更少的代码，而方案二更节省开发时间。

方案一：

- Communication Interface 配置：NONE
- 时钟配置：BSP_configureMCU()
- IO 配置：BSP_configureMCU()
- I2C 外设配置与中断使用：Driverlib I2C salveTxMultiple 例程；Driverlib I2C salveRxMultiple 例程

方案二★：

- User Guide：I2C Slave driver user guide
- Communication Interface 配置：REGISTER_I2C
- 时钟配置：BSP_configureMCU()
- IO 配置：BSP_configureMCU()
- I2C 外设配置与中断使用：REGISTER_I2C 配置下的代码包含 I2C 外设配置与中断使用。用户只需对 32 byte 的收发 Buffer 进行解析即可。操作步骤如下：
    - 步骤一：修改 CapTIvate->COMM->CAPT_Interface.c 中 CAPT_I2CRegisterReceiveHandler() 函数，该函数会在 I2C 写操作完成后产生 stop 信号或 restart 信号时被调用。如图 5-16 所示，删除其他代码，新增自定义帧处理函数。

```
static uint8_t g_ui8I2CDataBuffer[CAPT_I2C_REGISTER_RW_BUFFER_SIZE];
static bool CAPT_I2CRegisterReceiveHandler(uint16_t ui16Cnt)
{
    Cus_CAPT_CommunicationPackageAnalysis(g_ui8I2CDataBuffer);
    //
    // Return false to exit with the CPU in its previous state.
    // Since the response packet generation was handled immediately here,
    // there is no need to exit active.
    //
    return false;
}
```

图 5-16 CAPT_I2CRegisterReceiveHandler()函数修改示意图

    - 步骤二：编写自定义帧处理函数。如图 5-17 所示，MSP430 会读取主机所发送的一个字节，然后修改 Buffer 中数据段的第一个字节，然后等待主机主动读取该字节。要说明的是 I2C 的发送与接收共用一个 32 byte 的 buffer，且整个过程并不会对其复位。另外该 buffer 中接收/发送数据段会有 3 个 Byte 的偏移量。因此 Buffer 的第四个 byte 对应着接收/发送数据的第一个 byte。

```
void Cus_CAPT_CommunicationPackageAnalysis(uint8_t * pBuffer)
{
    switch (pBuffer[3])
    {
        case CUSTOM_KEY_STATE_CMD:
            pBuffer[3] = keyValue;
            break;
        case CUSTOM_DEVICE_NUMBER_CMD:
            pBuffer[3] = DEVICE_NUMBER;
            break;
        default:
            break;
    }
}
```

图 5-17 帧处理函数示意图

- 步骤三：按如图 5-18 所示的方式搭建整体代码。额外包含的功能有：
  - Panel_Init()：系统自定义初始化，注册 sensor 的回调函数。
  - BTNEventHandler()：定义 sensor 的状态的处理函数。

```
#include <msp430.h>
#include "driverlib.h"
#include "captivate.h"
#include "CAPT_App.h"
#include "CAPT_BSP.h"
#include "applications.h"
#include "I2CSlave.h"

#define CUSTOM_KEY_STATE_CMD  0x0B       //I2C CMD Byte

void BTNEventHandler(tSensor *pButton)
{
    if((pButton->bSensorTouch == 1) &&
            (pButton->bSensorPrevTouch == 0))
    {
        I2CSlave_setRequestFlag();
    }
}

void Cus_CAPT_CommunicationPackageAnalysis(uint8_t * pBuffer)
{
    switch (pBuffer[3])
    {
        case CUSTOM_KEY_STATE_CMD:
            pBuffer[3] = BTN00.bSensorTouch;
            break;
        default:
            break;
    }
}

void Panel_Init()
{
    //Customized initialization
    BTN00.pvCallback = (pvCallbackAddress)&BTNEventHandler;

}

void main(void)
{
    WDTCTL = WDTPW | WDTHOLD;
    BSP_configureMCU();
    Panel_Init();
    CAPT_appStart();

    while(1)
    {
        CAPT_appHandler();
        CAPT_appSleep();
    }
}
```

图 5-18 整体代码示意图

表 5-13 展示的是与自定义通讯功能相关的信息。

表 5-13 自定义通讯功能表

序列	修改对象	修改方式	说明
1	eUSCI_A0/eUSCI_B0 的端口	修改 CapTIvate_app->CAPT_BSP.C->BSP_configureMCU()中的引脚定义	选择其他通讯外设，需要额外修改外设初始化
2	I2C 的 IRQ 信号	修改 CapTIvate->COMM->CAPT_CommConfig.h 中对 IRQ 引脚的宏定义 修改 CapTIvate->COMM->I2CSlave.h 中对 IRQ 引脚动作的宏定义	该引脚用于提醒上位机出现触摸事件
3	CapTIvate 模块是否启用 Timer 外设	修改 CapTIvate->COMM->FunctionTimer.h 中对 FUNCTIONTIMER__ENABLE 的赋值	用于 I2C 通讯中的 timeout 功能
5	I2C 通讯的 Timeout 功能	修改 CapTIvate->COMM->I2CSlave_Definitions.h 中对 I2CSLAVE__TIMEOUT_ENABLE 的赋值 修改 CapTIvate->COMM->I2CSlave_Definitions.h 中 I2CSLAVE__TXFR_TIMEOUT_CYCLES 的赋值	默认情况下主机单帧传输超过 8ms 会复位 I2C
5	修改 I2C 的地址	修改 CapTIvate->COMM->I2CSlave_Definitions.h 中的 I2CSLAVE__ADDRES	

5.5.4 Bootloader

Bootloader 主要用于在线升级。硬件上需要占用 4 个 Pin 脚。两个引脚（RST， TEST ）用于选择进入 Bootloader 模式，剩余 2 个引脚用于通讯。由于电容触摸功能需要占据大量的代码空间，推荐使用 MSP430 内部自带的 ROM Bootloader。

说明文档：

MSP430 FRAM Devices Bootloader (BSL) User's Guide

例程:

MSP430 Bootloader With SimpleLink MCUs

MSP430 Bootloader With Sitara Embedded Linux Host

5.5.5 测试，生产与烧录

设计成功的电容式触摸感应系统需要使用原型设备进行系统验证。 TI 建议构建 20 到 50 个设备进行现场测试。如果系统满足性能要求， 设计可以转向大规模生产。 如果系统不符合性能要求， 需要调整性能预期或重新进行软硬件设计，甚至机械结构设计。

由于人体对地电容的不确定性，与不同人手的粗细不同，不建议使用人手来进行相关测试。建议使用不同粗细的接地铜柱来模拟人手的触摸。用粗铜柱表征灵敏度下限，此时每个测试按键均能正确响应触摸触摸信号。用细铜柱表征灵敏度下限，此时每个测试按键应不响应触摸触摸信号。具体铜柱的粗细需根据实际的灵敏度需求进行设计。

小批量生产时，推荐的烧录软件为 Uniflash，烧录工具为 eZ-FET，或 MSP-FET：

UniFlash★： UniFlash 是 TI 公司提供的支持 JTAG 和 BSL 的 GUI 编程工具。只支持程序下载，不支持调试。另外，可按照该链接中的步骤生成 UniFlash 支持的下载文件。如无法识别 MSP430，请安装 MSP430 FET Drivers。

- 产品页面：UniFlash
- 用户手册：
  - UniFlash Quick Start Guide
  - Programming the Bootloader of MSP430™ Using UniFlash

MSP-FET★： 功能最强大的 MSP430 调试器。输出电压可配，最大供电电流为 100mA。

- 产品页面：MSP-FET MSP MCU Programmer and Debugger

eZ-FET★： 一种低成本的 MSP430 调试器，与 LaunchPad 配套出售。此外，输出电压固定。

- 产品页面：请参阅相关的 Launchpad 产品页面，如 MSP430FR2311 LaunchPad kit。

大批量生产时，推荐使用 MSP-GANG 与其配套的上位机软件：

MSP-GANG★： MSP Gang 下载器不支持调试代码，主要用于产品生产烧录。它可以在没有 PC 介入的情况下单独运行，支持同时对 8 个 MSP430 进行编程。

- 产品页面：MSP-GANG Production Programmer

6 附录

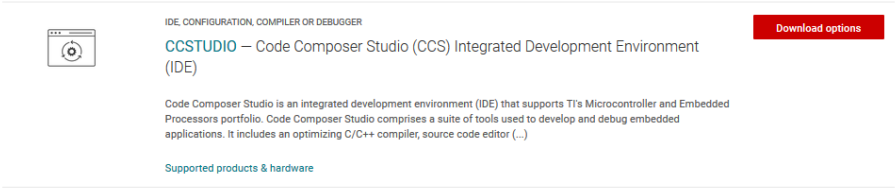
该附录将展示如何一步一步安装 CCS 与 CapTIvate 开发中心（CapTIvate 开发中心用于生成代码与在线调参，CCS 用于下载 GUI 生成的代码）；实际硬件连接介绍；如何快速评估 TI 提供的 EVM 板，以及如何快速开发自己的应用。

6.1 CCS 安装

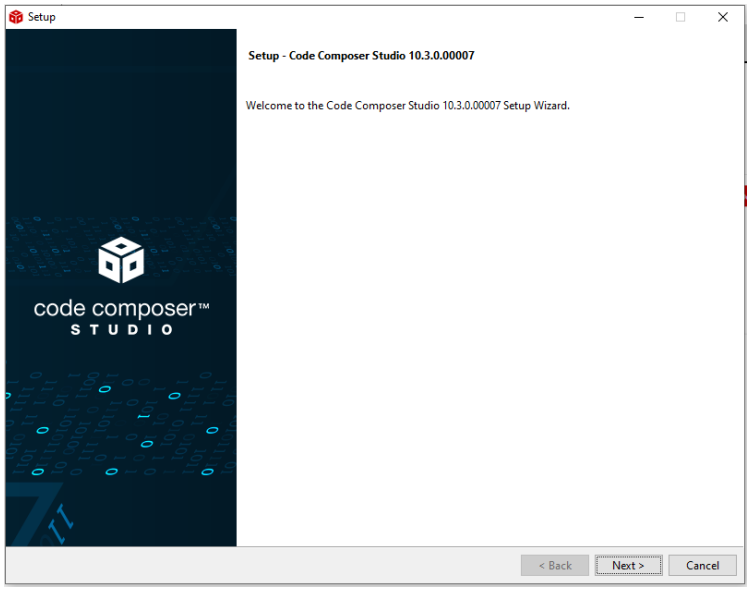
1. 了解和下载集成开发环境 CCS

产品页面：CCS IDE for MSP430

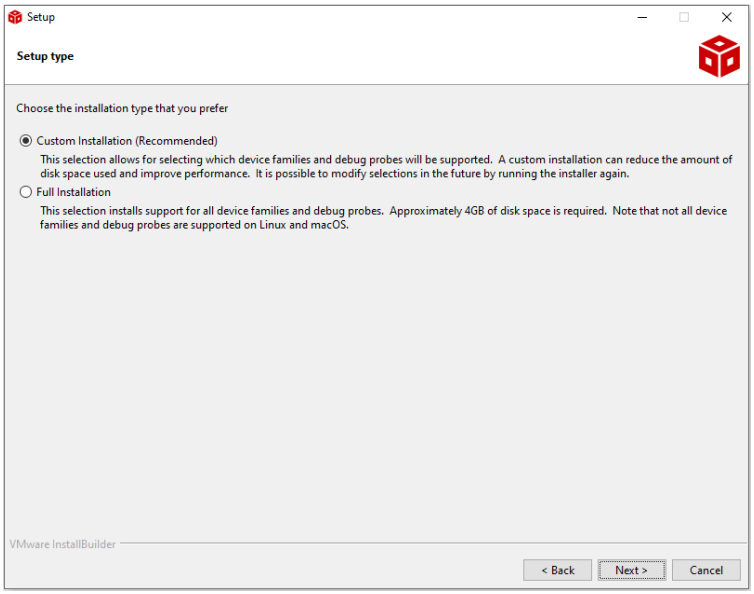
2. 根据安装电脑的操作系统，选择下载离线版或者在线安装版本



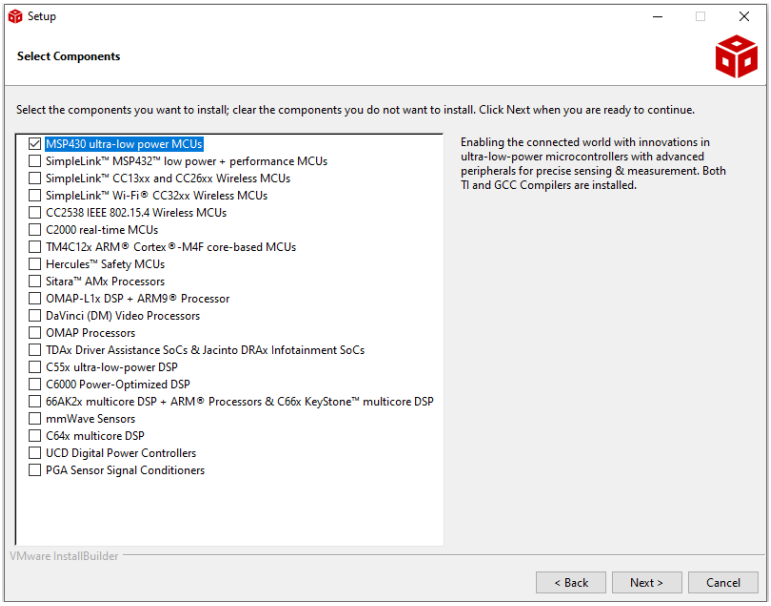
3. 开始安装



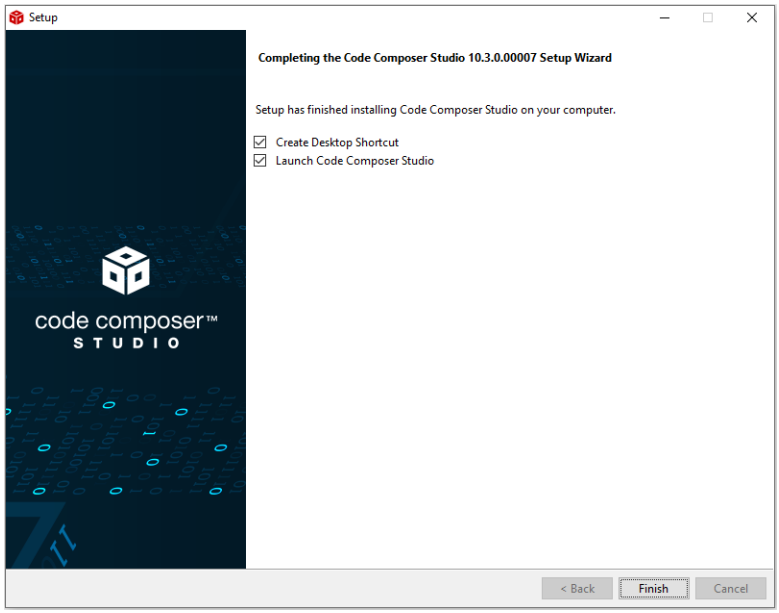
4. 推荐选择自定义安装



5. 选择 MSP430 MCU



6. 安装完成

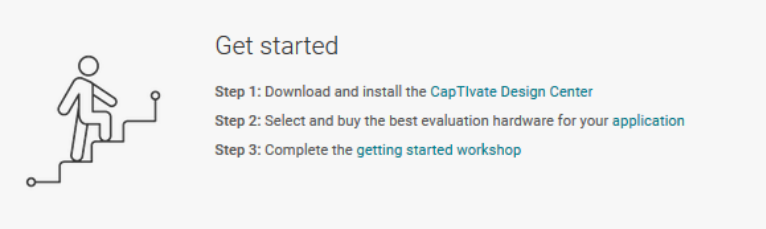


6.2 CapTIvate 开发中心安装

注意安装 CapTIvate 开发中心还需要安装对应的 Java JDK。

1. 了解和下载 CapTIvate 开发中心

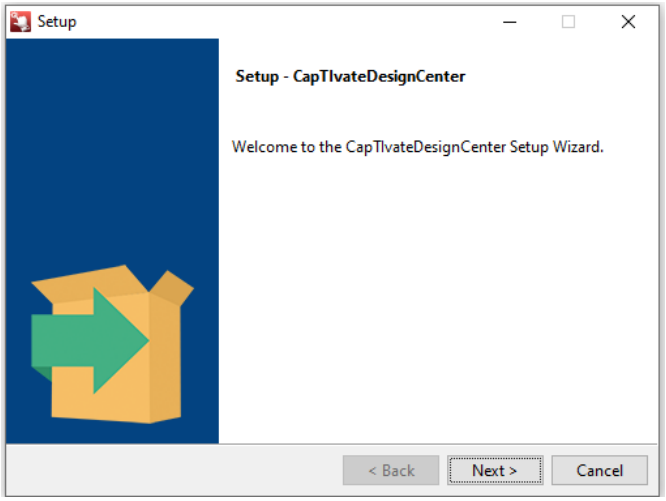
产品页面：CapTIvate™ Design Center



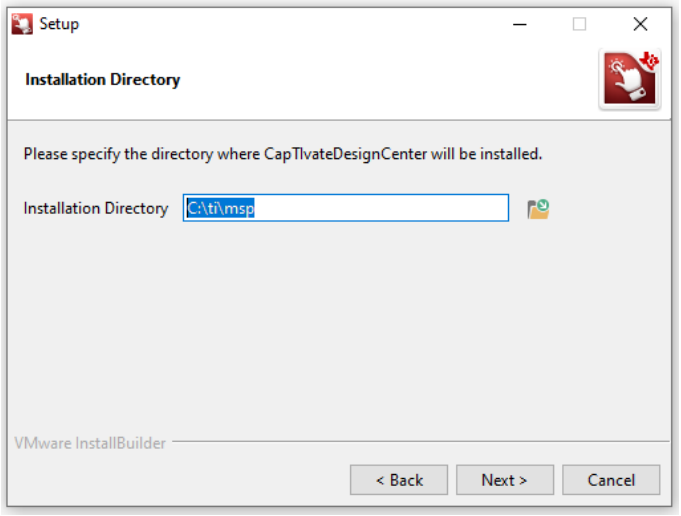
CapTIvate_Design_Center Product downloads

Title	Description	Size
CapTIvateDesignCenter-1.83.00.08-windows-installer.exe	Captivate Design Center for Windows	56784K
CapTIvateDesignCenter-1.83.00.08-osx-installer.app.zip	Captivate Design Center for Mac OS X	53424K
CapTIvateDesignCenter-1.83.00.08-linux-x64-installer.run	Captivate Design Center for Linux	56232K
md5sum.txt	MD5 Sum	4K
Captivate Design Documentation		
Release_Notes_CapTIVateDesignCenter_1_83_00_08.html	Release Notes	80K
API Guide	API Guide	8K
Users Guide	Users Guide	4K

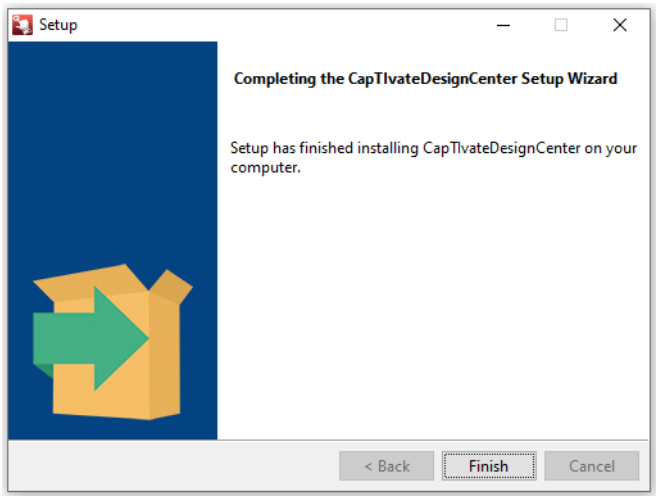
2. 开始安装



3. 推荐默认安装目录



4. 安装完成

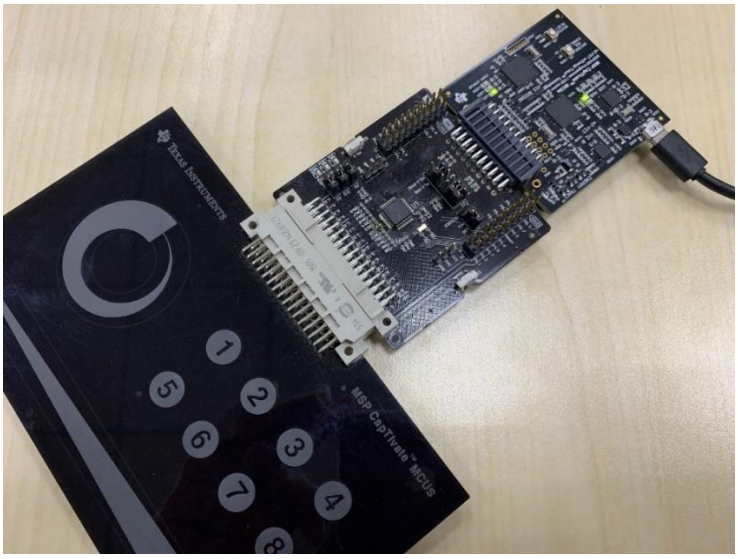


6.3 硬件连接

1. 使用编程器，处理器和电容触摸演示板快速组成验证硬件

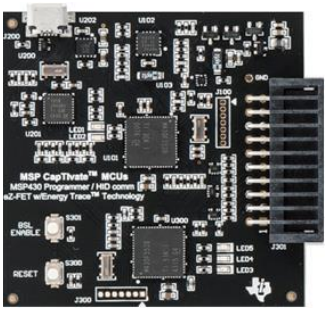
硬件连接，快速评估，快速开发的演示将基于下表中所标注的开发链进行。对于其他的 EVM 板，与该评估方式类似。

GUI	IDE	SDK	Programmer	MCU board	Sensor board	Supported Sensor type
CapTIvate™ Design Center	CCS /IAR	MSPWare	CAPTIVATE™-PGMR	CAPTIVATE™-FR2676	CAPTIVATE™-BSWP	Self-mode button/Wheel/slider/Proximity
				CAPTIVATE™-FR2633	CAPTIVATE™-PHONE	Mutual-mode button/Wheel/slider/Proximity
				BOOSTXL-CAPKEYPAD(FR2522)		12 mutual mode buttons and 1 proximity
				EVM430-CAPMINI (FR2512)		4 self-mode buttons



2. CAPTIVATE-PGMR 编程器

编程器产品主页



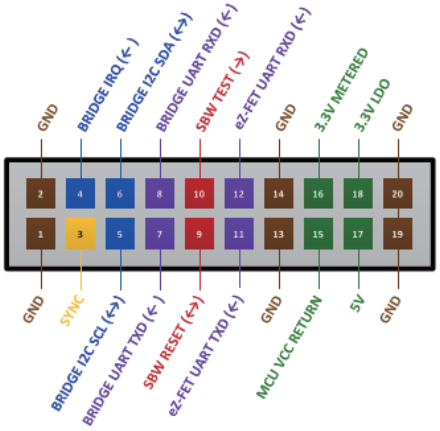
CAPTIVATE-PGMR 编程器定义

供电引脚为：3.3V METERED，3.3V LDO。注意 METERED 用于 Energytrace 功能，电源纹波噪声较大，而 LDO 适用于纹波较小的供电需求。

烧录引脚为：SBW TEST 与 SBW RESET。

调试串口的功能引脚为：eZ-FET UART RXD；eZ-FET UART TXD。注意该功能为 USB 转串口，与在线调参的功能引脚进行区分。

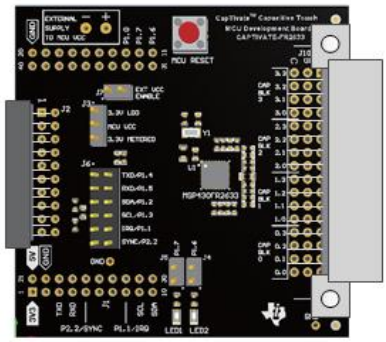
电容触摸在线调参的功能引脚为：BRIDGE UART RXD；BRIDGE UART TXD。BRIDGE I2C SDA；BRIDGE I2C SCL。



PCB 20-PIN Male Connector

3. CAPTIVATE-FR2633 处理器电路板

处理器电路板产品主页



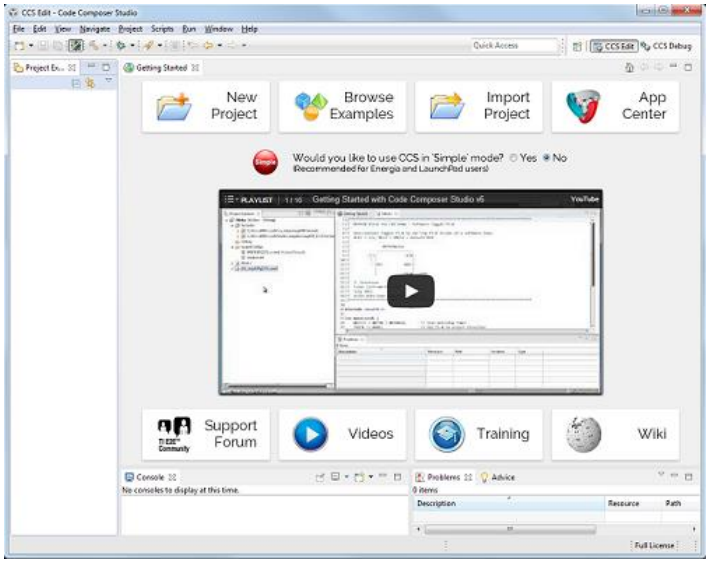
4. CAPTIVATE-BSWP 自电容演示板

触摸演示板产品主页

6.4 快速评估

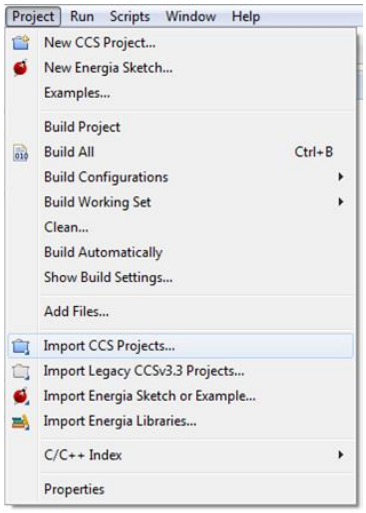
当 CapTIvate 开发中心已经安装同时硬件验证平台已经准备好，我们可以考虑在 CCS 中打开 CapTIvate 开发中心自带的样例工程，快速体验电容触控的魅力。

1. 打开 CCS



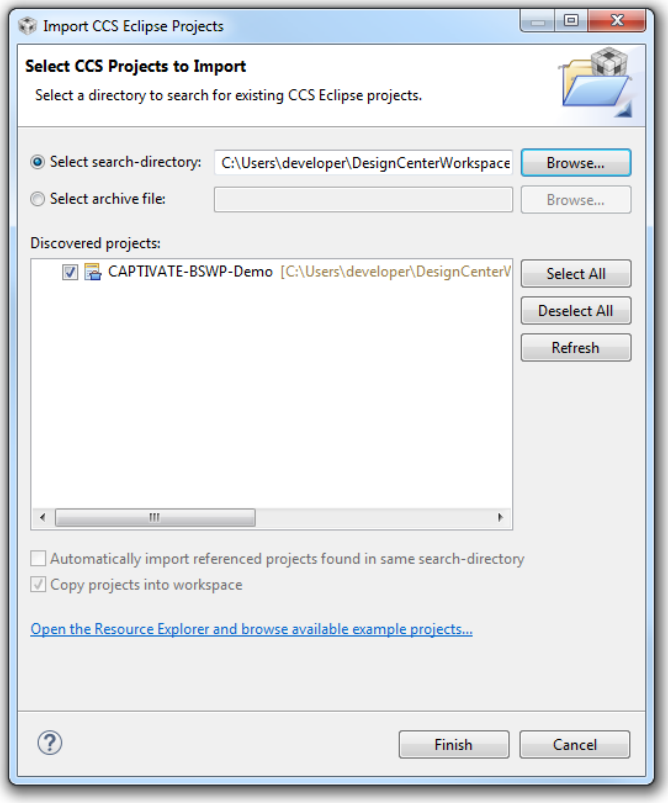
2. 导入 CCS 工程

选择 “Project->Import CCS Projects”



3. 在 design center 的默认路径中选择相应的样例工程

示例路径为：
C:\Users\Username\CapTIvateDesignCenter_x_xx_xx_xx\CapTIvateDesignCenterWorkspace\TI_Examples\FR2676_CAPTIVATE-BSWP



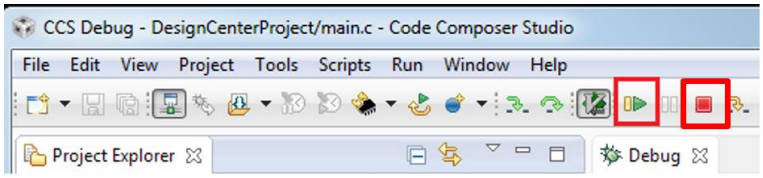
4. 运行和体验电容触摸的样例工程

点击调试按钮

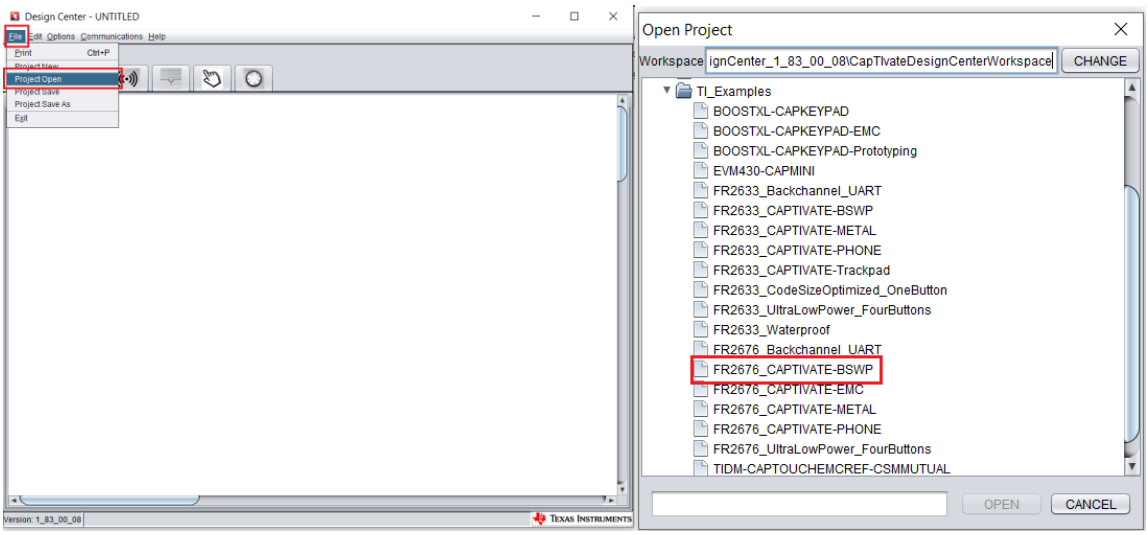
第一次编译可能花费较长时间，请耐心等待。

```
warning #10366-D: automatic library build: using library
"C:\ti\ccs1030\ccs\tools\compiler\ti-cgt-msp430_20.2.4.LTS\lib\rts430x_lc_ld_eabi.lib" for the first time, so it must be built.  This may
take a few minutes.
```

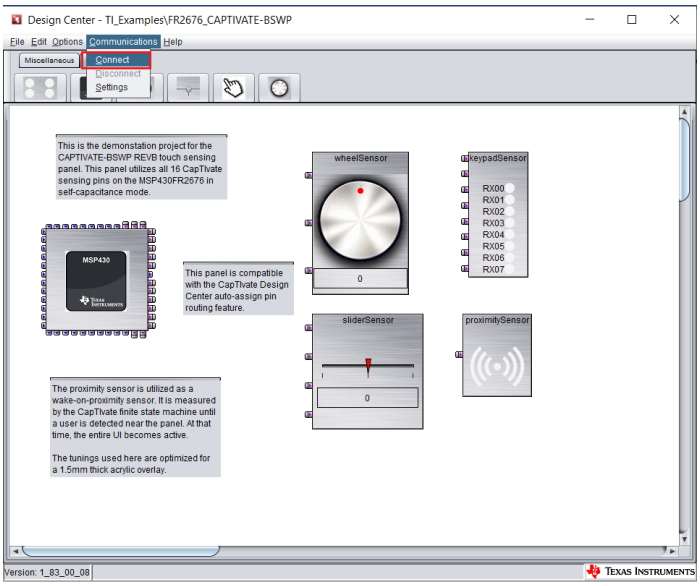
点击运行按钮和结束按钮。然后可能需要**插拔**一下电源。



5. 导入对应的 CapTIvate 开发中心工程

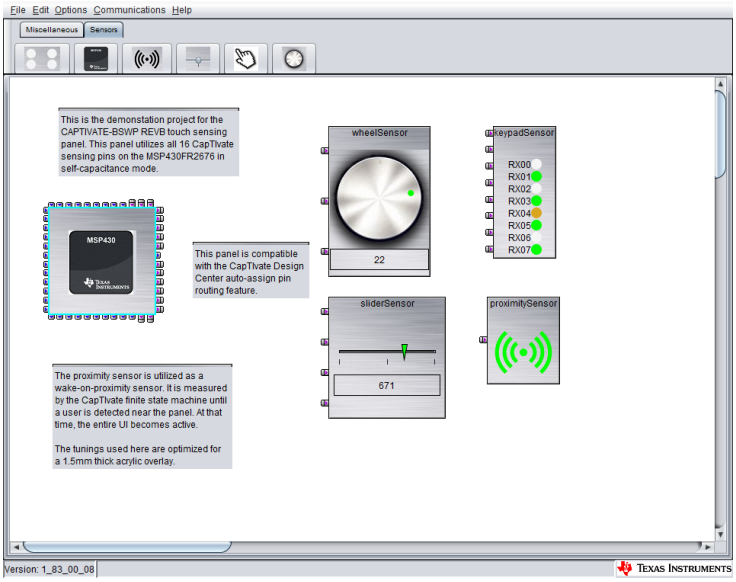


6. 将 CapTIvate 开发中心连接到 MSP430



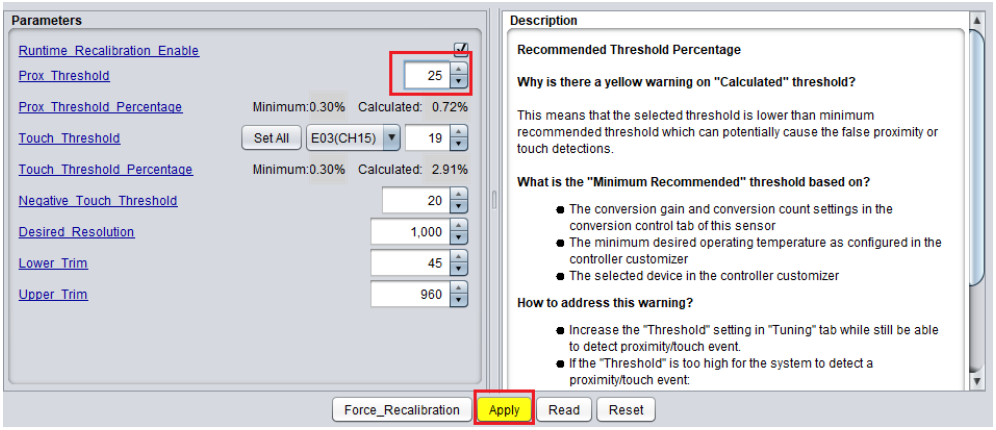
7. 体验电容触摸的相关功能

说明：白色表示无信号触发，黄色表示接近感应信号触发，绿色表示触摸信号触发。



8. 在线调参

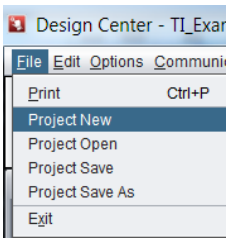
修改参数，点击 Apply 按键即可完成参数的在线更新，更多说明与介绍请参考本手册的 5.3 章节。



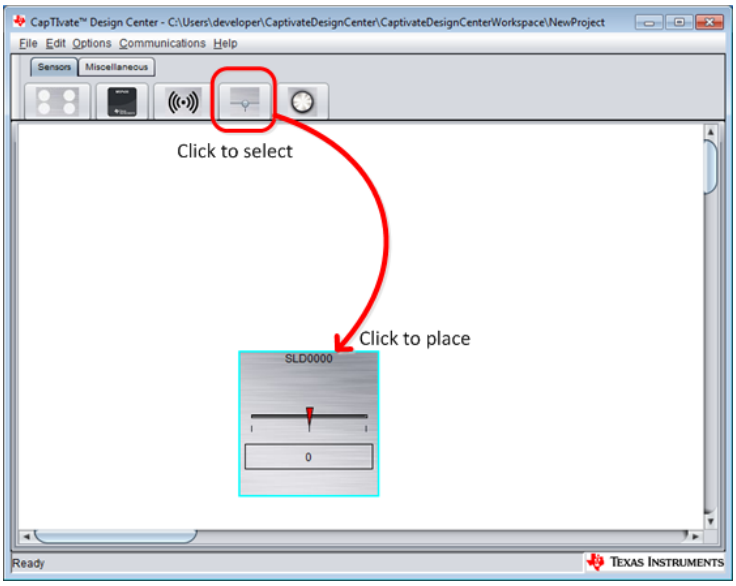
6.5 快速开发

在用户开发自己的硬件之前，我们可以基于 CapTIvate 开发中心和 MSP 官方的硬件验证平台快速创建工程，配置触控功能，快速开始验证和简单开发。

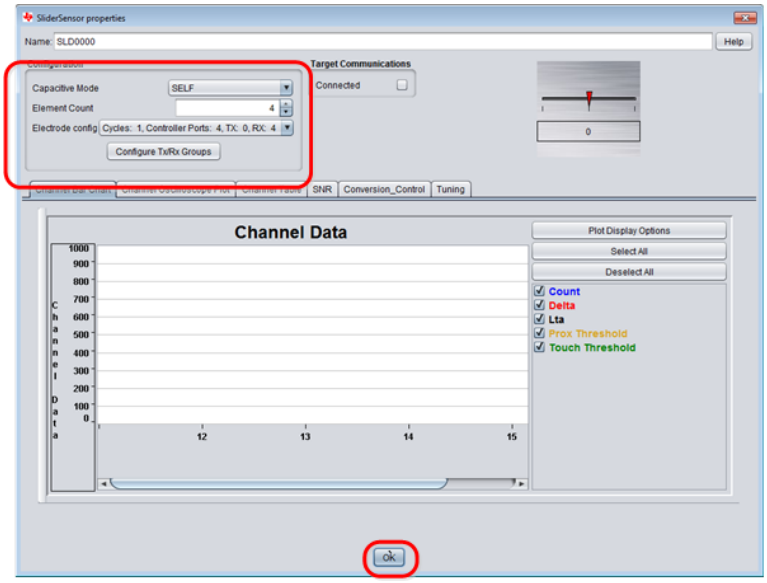
1. 在 CapTIvate 开发中心中创建一个新的工程



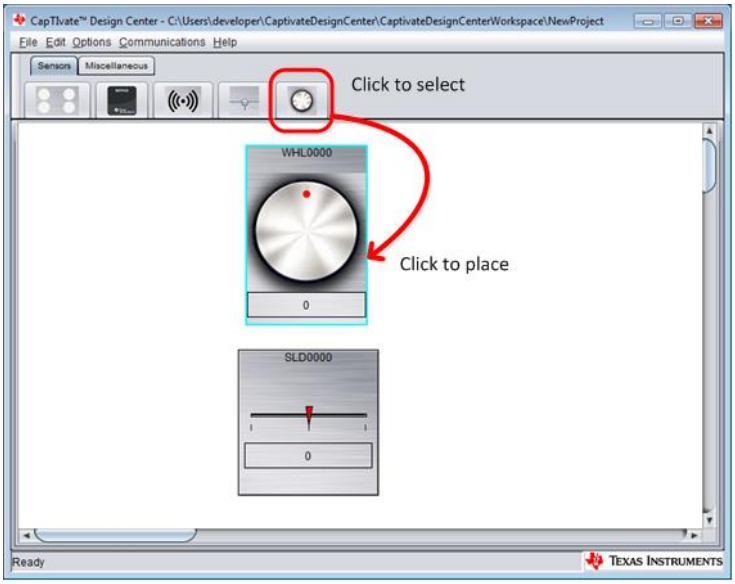
2. 根据设计需求放置滑条传感器



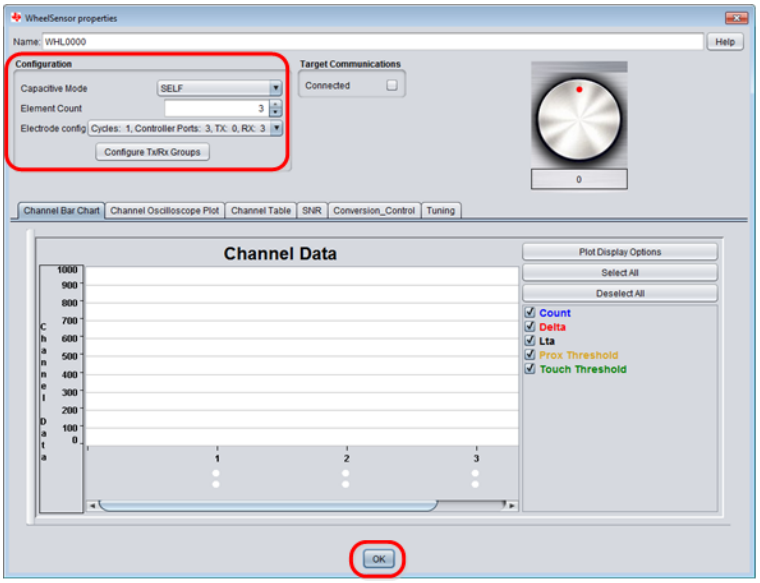
3. **根据设计需求设置滑条传感器的属性**



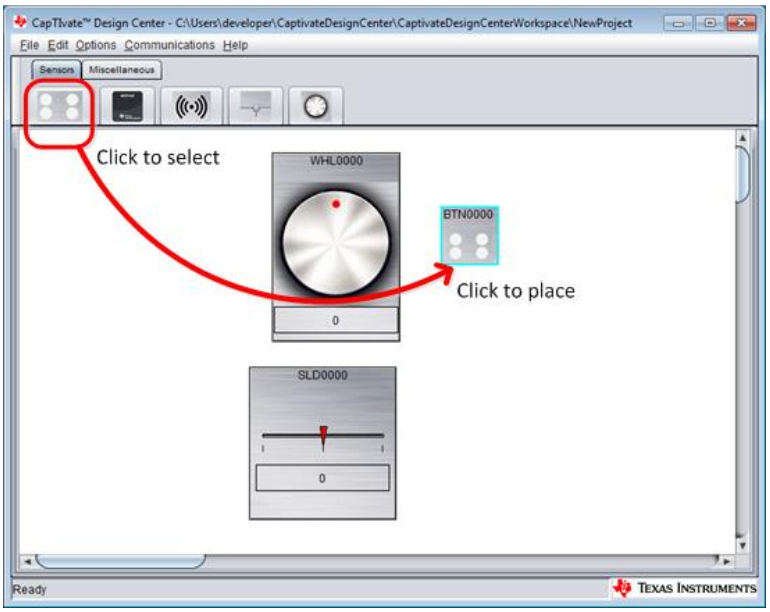
4. **根据设计需求放置滚轮传感器**



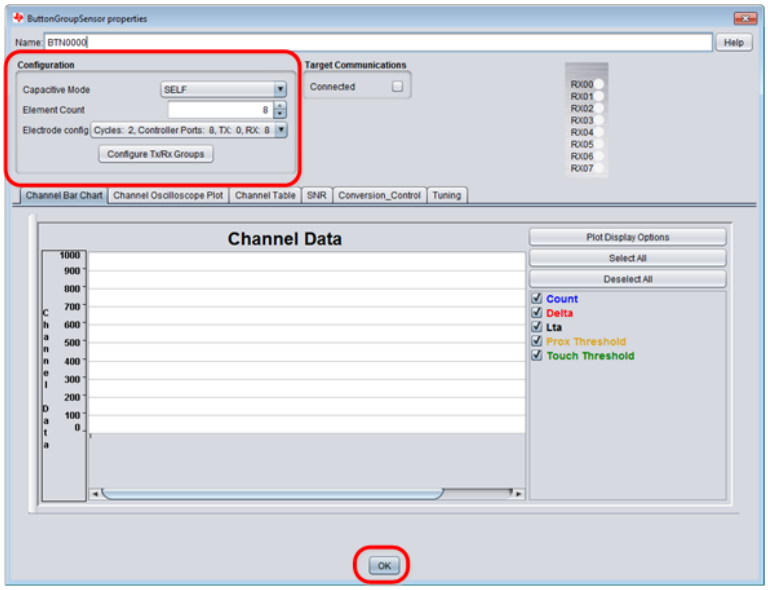
5. **根据设计需求设置滚轮传感器的属性**



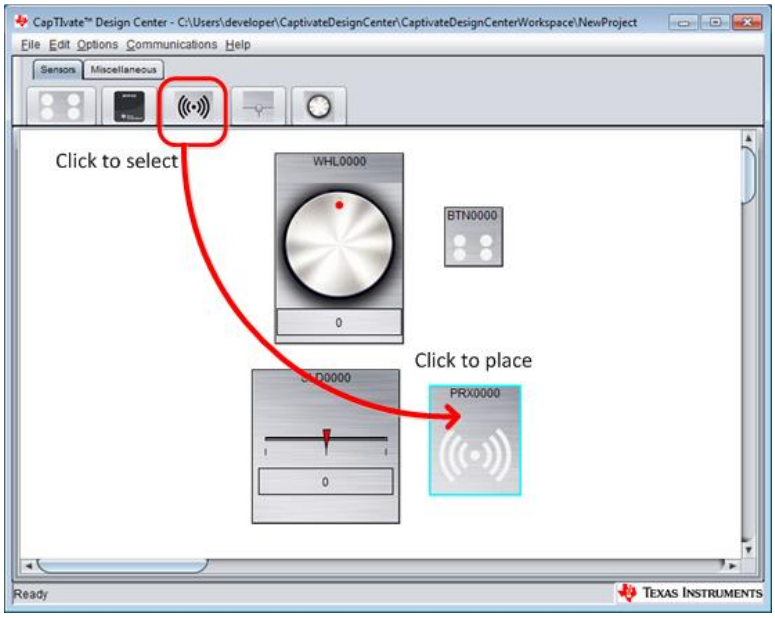
6. **根据设计需求放置按键传感器**



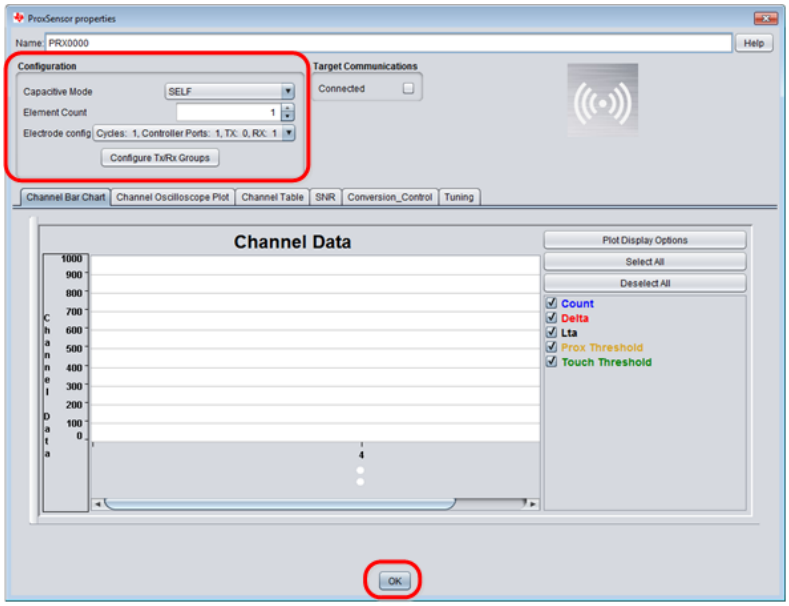
7. **根据设计需求设置按键传感器的属性**



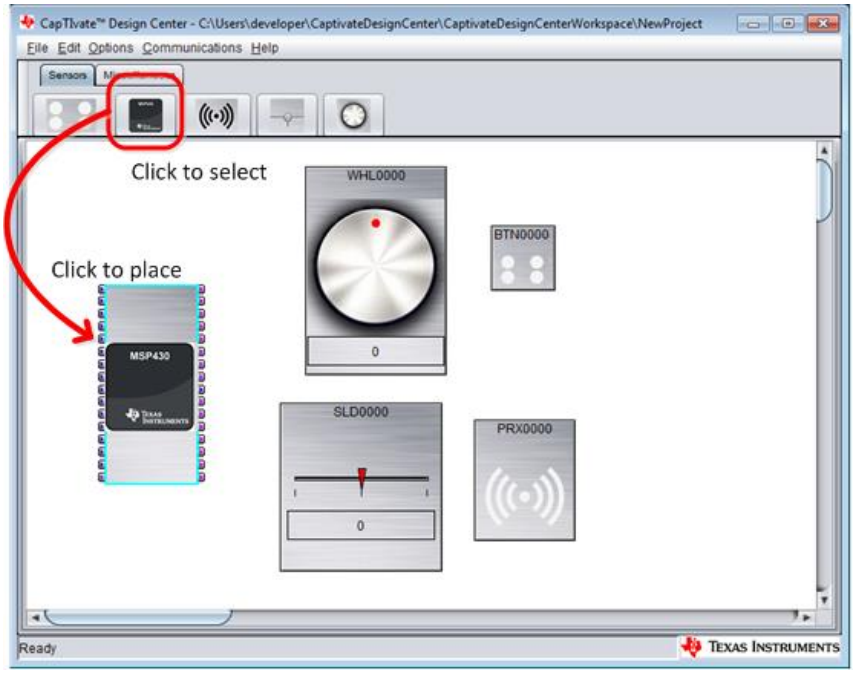
8. **根据设计需求放置接近感应传感器**



9. 根据设计需求设置接近感应传感器的属性

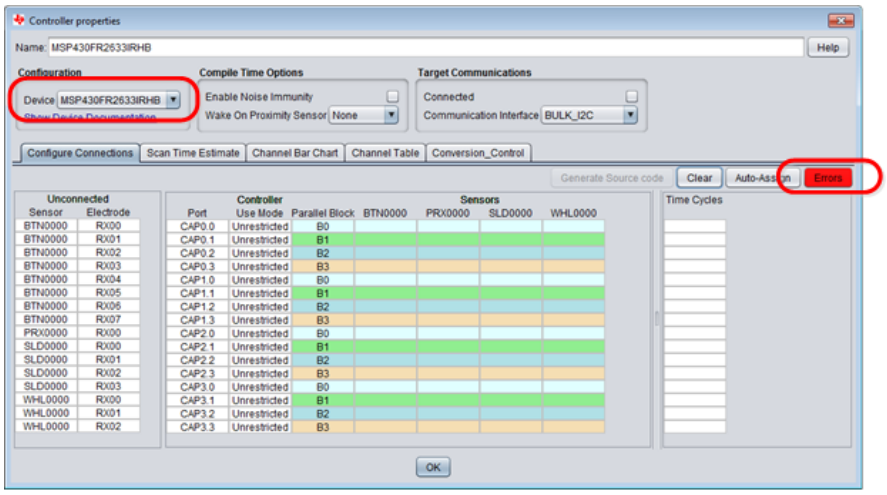


10. 选择合适的 **CapTIvate™ MCU**

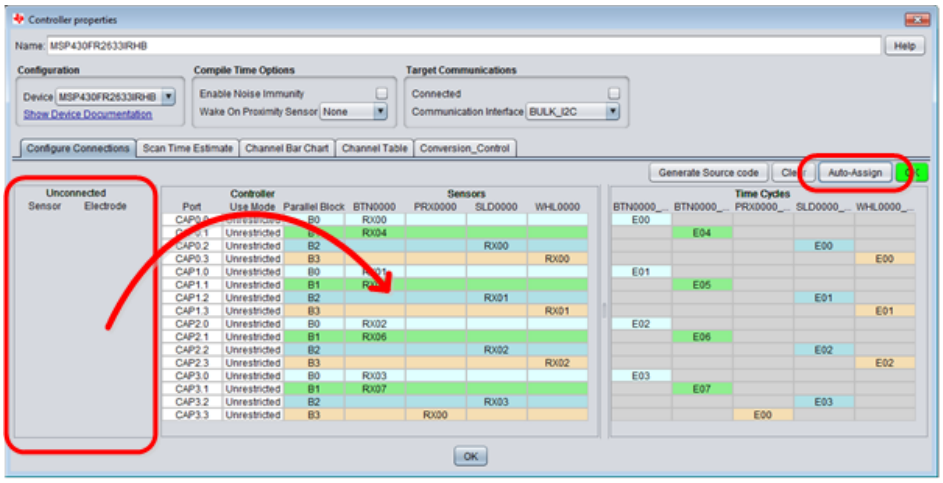


11. 将传感器连接到 CapTIvate™ MCU 电容式触摸 I/O 端口

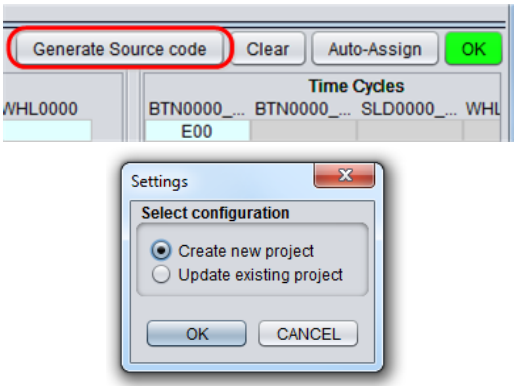
双击选择的 CapTIvate™ MCU 控件以显示属性。
将 MSP430 控件配置为 MSP430FR2633IRHB (32 引脚 QFN 封装)
请注意，红色 LED 亮起，表示此时仍有未连接的传感器端口。



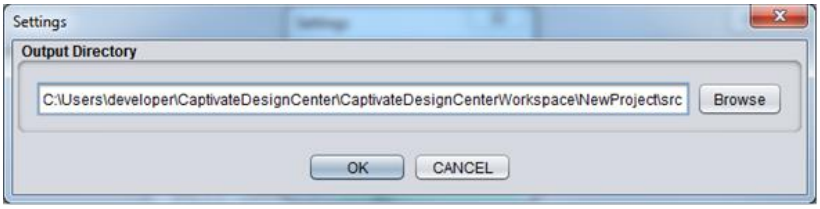
选择“Auto-Assign”按钮，自动将所有传感器端口分配给 CapTIvate™ MCU 上的相应端口。**注意**：实际项目开发时需要根据需求进行虚拟端口与实际端口的连接。当绿色 LED 指示灯亮起，表示所有传感器端口都已分配给 CapTIvate™ MCU 的端口。



12. 自动生成源代码

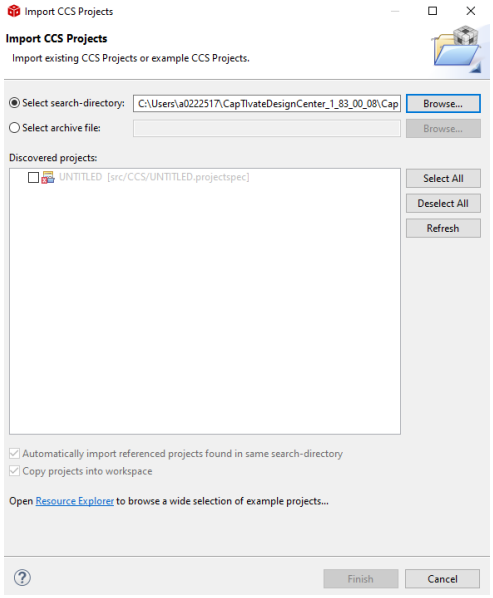


推荐保存源代码在默认 CapTIvate Design Center 目录下。

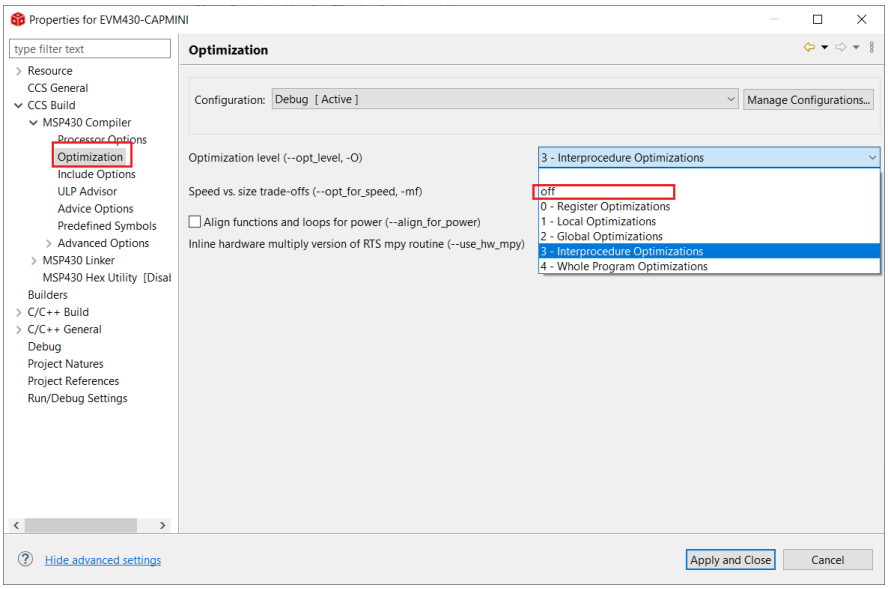


13. 在 CCS 上导入工程，开始上手开发

根据 6.4 章节完成快速评估的流程。然后在 CCS 中导入调整完参数后生成的工程，添加系统功能。注意：如果 CCS 图标中显示灰色，表示 workspace 或 CCS 目录中有相同命名的工程，如下图所示。



注意：由于为了节省代码空间，代码编译优化选择最高级别。建议实际开发时将 Optimization 关闭。



重要声明和免责声明

TI 提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、某特定用途方面的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他安全、安保或其他要求。这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的应用。严禁对这些资源进行其他复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。您应全额赔偿因在这些资源的使用中对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，TI 对此概不负责。

TI 提供的产品受 TI 的销售条款 (https:www.ti.com.cn/zh-cn/legal/termsofsale.html) 或 ti.com.cn 上其他适用条款/TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。

邮寄地址：上海市浦东新区世纪大道 1568 号中建大厦 32 楼，邮政编码：200122