

---

# 快速入门指南

欢迎您体验 TSI Tuner 软件，本软件是搭配复旦微芯片的触摸模块（TSI）推出的，旨在方便用户能够快速建立触摸应用原型设计并进行参数调试。让我们以一个简单的需求来带领您快速理解本软件的使用方法吧！

## 确定需求

为了能够全方面的展现 TSI Tuner 软件的各项功能，我们的 DEMO 工程计划设计实现一个可以使用触摸滑条和按键进行调色调亮度的小彩灯。该设计需要如下控件：

- 2 个按键，一个用作开关，一个用作模式选择；
- 1 个触摸滑条，用于调节亮度或者颜色；

主控芯片选择复旦微触摸 DEMO 板使用的 FM33FR0510。最终我们制作的 DEMO PCB 如图所示：

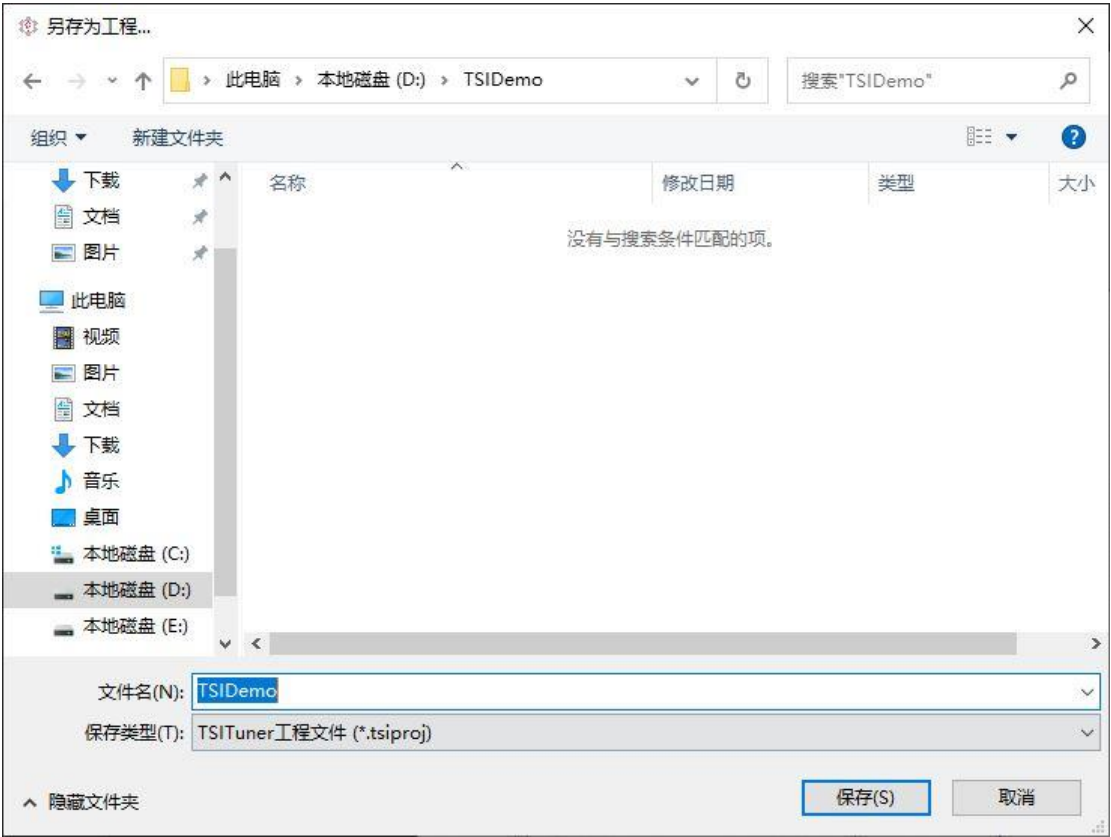


# 新建工程

打开软件后，在首页点击 New 按钮，将会打开新建工程对话框。我们需要填写工程名称，选择芯片系列和芯片型号。确认无误后，点击创建按钮，完成新建工程操作。

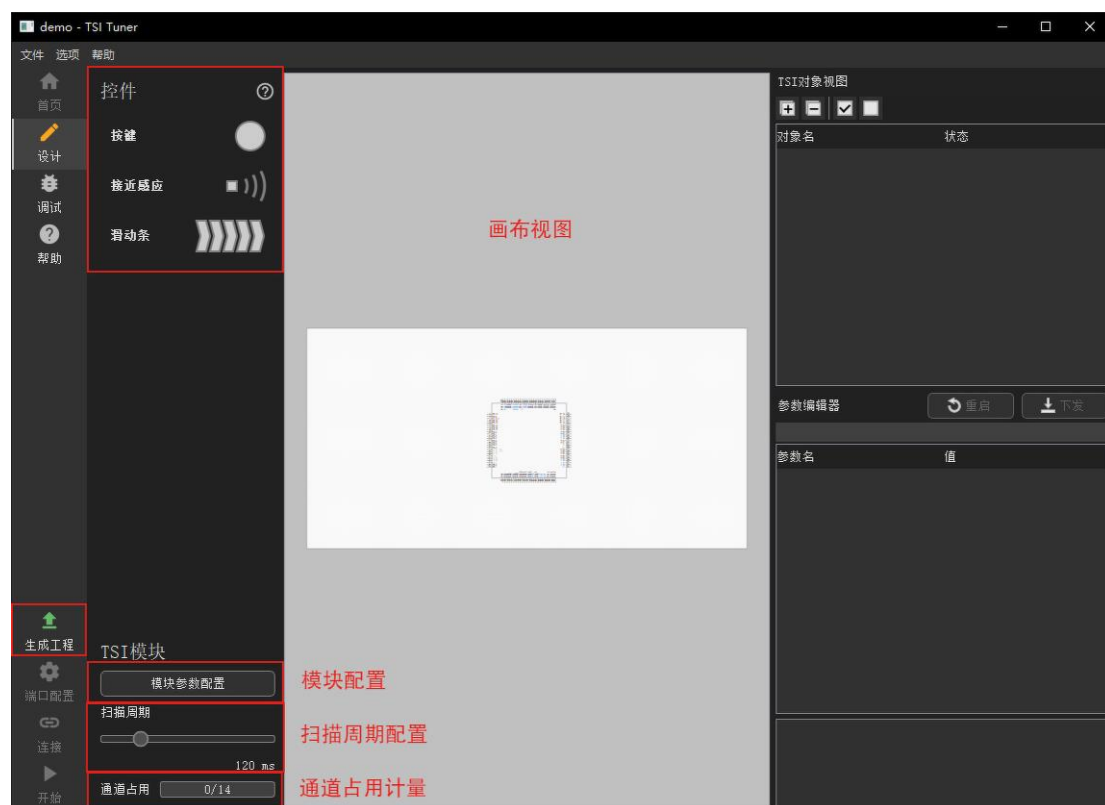


创建工程后，可以通过文件⇒保存工程对工程文件进行保存，或者可以使用快捷键 Ctrl+S。这里我们将工程文件保存至 D:\TSIDemo 文件夹中。



## 设计

在创建工程完成后，进入到设计界面。在该界面，我们可以通过图形化的方式，规划布置触摸控件（按键、接近感应、滑条等）。下面对设计页面做一下简单介绍。



位于设计页面中央的是画布视图，用于浏览操作设计图纸。图纸设计操作按键如下：

- **浏览：**
  - 右键按住移动鼠标可以平移画布视图；
  - 鼠标滚轮上下滚动可以缩放画布视图；
- **选择和编辑：**
  - 鼠标左键单击：用于选择控件和连线。控件和连线选中状态下，按 Delete 键可删除；
  - 鼠标左键双击：对控件有效。双击控件会弹出一个浮动窗口，可以配置控件的属性（名称，滑条控件可配置段数等等）；
- **绘制连线：**画布中芯片封装的引脚，以及控件的引出脚都是可以进行连接的端口。将鼠标移动到需要连线的一个端口上，如果出现红色“X”光标，此时单击鼠标左键可以开始连线。连线时可以使用空格键控制拐弯方向，鼠标左键按下可以创建新的路点。当走线到达目标端口时，鼠标放在目标端口上，同样如果出现红色“X”光标，则单击鼠标左键

---

可以结束连线绘制。如果在连线过程中需要放弃连线重新连接，可单击鼠标右键，这将退出连线模式，并删除当前绘制的连线。

**注意：**如果鼠标放在端口上后，鼠标变成了禁止的图标，说明该端口不可连线。

页面左侧的面板上有两个栏目：控件列表和模块配置栏目。控件列表栏目内容为当前可用的控件，我们可以在想要加入设计的控件项目上左键按下并拖动来将其加入设计；模块配置栏目内容包括模块参数配置，扫描周期配置和通道占用计量条。

页面右侧面板有两个窗口，分别是 TSI 对象视图和参数配置视图。这两个窗口在设计模式和调试模式都存在，但用途会有一些差别。设计模式中，TSI 对象视图用于展示当前画布中存在的控件信息，以及他们的状态。参数编辑器视图则用于配置对象的参数。通过在 TSI 对象视图单击想要编辑参数的对象，来在参数编辑器视图中编辑参数。

## 添加控件到设计

依据先前制定的需求，我们将按键和滑条加入到设计画布中。在左侧控件列表中分别按住按键和滑条项目，拖动到画布中，即可完成添加。双击控件，会弹出控件属性列表，我们可以为它重新起一个便于记忆的名字。这里我们分别命名按键为 Power 和 Mode，命名滑条为 Slider。



通过在画布上连线，可以为控件分配管脚。我们将 Power 按键分配到通道 10，将 Mode 按键分配到通道 4，将 Slider 滑条的第 1~5 段依次连接到通道 5~9。最终配置如下图所示：





---

议设置为 32 以上;

- 接近传感通道 (接近感应控件): 推荐您开启**中值滤波器**和**高级二段式 IIR 滤波器**。对于干扰较大的情况, 可以选择开启**均值滤波器**;

**允许 Baseline 始终更新**选项会使得基线算法在触摸信号大于触发阈值后继续更新。大多数情况下, 您并不需要勾选此项。

**使用 IIR Baseline** 选项能够配置固件使用 IIR 滤波器作为 Baseline 算法。在此模式下需要调校的参数较少, 实现设计较为省心, **建议勾选该项**。

**参数调校方式**选项提供了 3 种方式: **手动调校**、**硬件参数自动调校**和**补偿 IDAC 自动调校** (仅在使能补偿 IDAC 时可选)。**我们推荐您仅在原型设计的时候使用硬件参数自动调校**, 这将使得触摸库自动把硬件参数 (采样时钟分频、分辨率、IDAC 电流) 调整到一个相对适宜的范围内; 而当您开始细调配置的时候, 强烈建议您修改为使用**手动调校**或者**补偿 IDAC 自动调校**:

- 手动调校会使触摸库完全按照您指定的数值来配置硬件参数, 从而保证调参时和实际测试生产环境参数的一致性;
- 补偿 IDAC 自动调校在手动调校之上加入了对补偿 IDAC 的自动校准, **在灵敏度不变前提下增强对环境的适应性**;

综上所述, 对于我们的示例应用, 我们配置了如下项目:

- 时钟和功能配置页面:
  - TSI 时钟源: APB
  - TSI 时钟源频率: 24MHz
  - TSI 采样时钟源: PRS
  - 勾选**使用补偿 IDAC** 选项
- **触摸检测配置**页面:
  - 普通传感通道信号滤波设置中, 勾选**均值滤波器**和**中值滤波器**;
  - 勾选**使用 IIR Baseline**;
  - 参数调校方式选择**硬件参数自动调校**;

模块参数配置 - TSI Tuner

时钟和功能配置

触摸检测配置

普通传感通道信号滤波设置

☒ 均值滤波器

☒ 中值滤波器

☐ IIR滤波器

IIR滤波器系数

接近传感通道信号滤波设置

☐ 均值滤波器

☐ 中值滤波器

☐ 高级二段式IIR滤波器

检测算法配置

☐ 允许Baseline始终更新

☒ 使用IIR Baseline

参数调校配置

参数调校方式 

硬件参数自动调校

OK

Cancel

模块参数配置 - TSI Tuner

时钟和功能配置

触摸检测配置

时钟配置

TSI时钟源 

APB

TSI时钟源频率  MHz

TSI时钟分频 

无

TSI工作时钟频率为 24.00 MHz

TSI采样时钟源 

PRS

功能配置

☒ 使用补偿IDAC

☐ 使用屏蔽电极

不活动通道连接方式 

接地

OK

Cancel

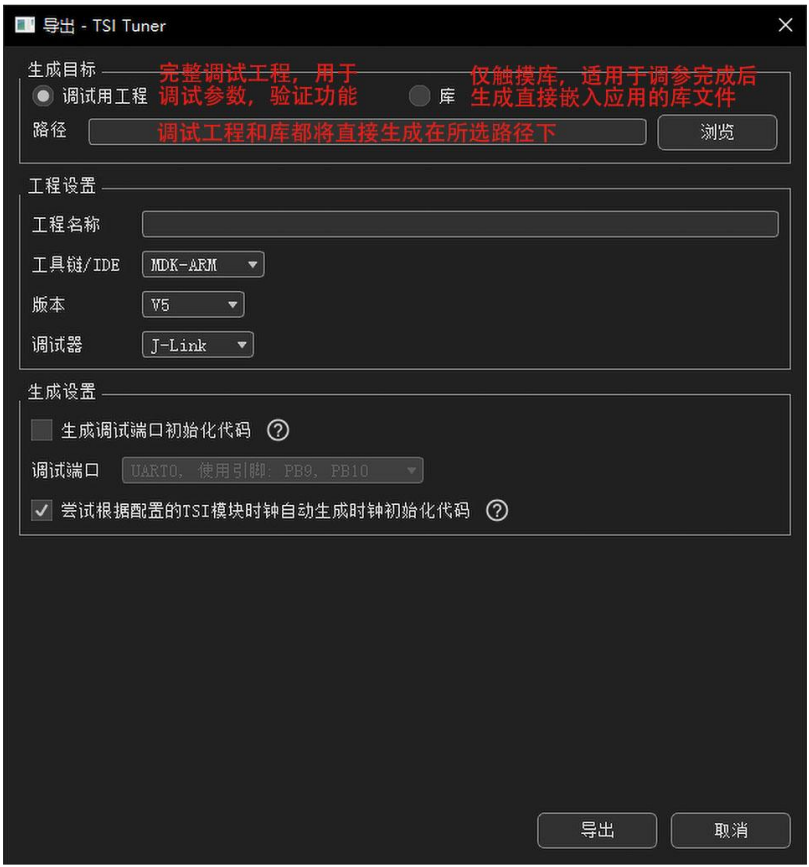
## 配置扫描周期

扫描周期代表了每一轮触摸按键扫描的周期，也即每次数据更新的最小时间单位。示例工程使用的 FM33FR0 系列芯片，其扫描周期最小为 20ms，并以 20ms 为单位步进。在示例工程使用的场景下，由于不需要低功耗，为了使按键响应尽可能快，我们将扫描周期配置为最短的 20ms。



## 生成调试工程

在完成设计后，我们就可以生成调试工程并下载到芯片中，进行调试工作了。单击左侧工具栏中的生成工程按钮，将弹出导出对话框。下面介绍一下导出对话框内各个输入框和选项：





- 
- **生成目标：**
    - 可以选择生成完整调试工程还是仅生成库；
    - 路径：指定生成路径；
  - **工程设置：**
    - 工程名称：生成工程的名称；
    - 工程路径：生成工程所在的路径，工程会直接生成在该路径中；该输入框无法直接编辑，您可以通过右侧的**浏览**按钮选择目标文件夹；
    - 工具链/IDE、版本、调试器：选择您打开工程使用的工具链及其版本，以及仿真和下载使用的调试器类型；
  - **生成设置：**
    - **生成调试端口初始化代码模版**选项：TSITuner 软件提供了一些调试端口配置模版，方便您快速配置工程，**一般推荐您首次导出的时候打开该选项**；生成调试端口配置后，可以根据实际情况再修改端口使用的 IO 口等来匹配您 PCB 的实际情况；
    - **尝试根据配置的 TSI 模块时钟自动生成时钟初始化代码**选项：推荐勾选，TSITuner 软件会根据您的模块参数配置，尝试自动生成符合要求的时钟初始化代码，免去手动配置的烦恼（大多数情况都能正确配置，少数特殊情况无法自动生成，需要手动在代码中添加）。

对于示例工程，我们最终配置如下（仅供参考，您可以根据自己的实际需要进行配置）：



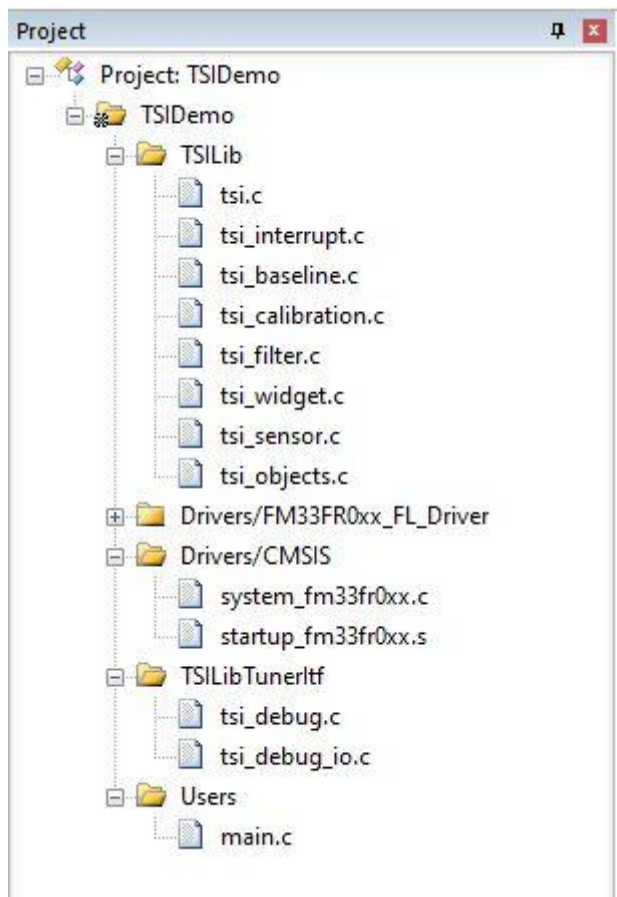
单击**导出**按钮，可以将工程导出到指定文件夹。

**注意：**如果在之前您没有保存过工程文件，单击导出按钮后会先弹出窗口让您选择工程文件的保存位置，再执行导出操作。

在弹出导出成功提示后，我们点击**打开**按钮打开调试工程：



调试工程目录结构如下图所示：



**TSILib** 目录下内容为 TSI 软件库；**TSILibTunerItf** 为 TSI 软件库调试接口实现，仅用作调试使用，实际应用中不需要添加这些文件；**Drivers/FM33FR0xx\_FL\_Driver** 和 **Drivers/CMSIS** 为您选择的芯片对应的 FL 驱动库；**Users** 文件夹中的 **main.c** 包含主程序入口。

## 补充代码并下载

要让生成的调试工程能够正常运行，我们还需要补充两处代码：

- **调试端口配置代码：** **TSILibTunerItf/tsi\_debug\_io.c**，需要实现相应接口函数；如果在上一步没有勾选**生成调试端口初始化模版**的选项，这里需要手动补充；建议首次配置的时候先勾选生成端口配置代码，再根据模板修改成需要的配置；

***注意：**如果您使用生成的配置代码作为模板进行了修改来适配设计，请注意之后如果还需要导出代码到该工程的时候，不要再勾选生成调试端口初始化代码选项，否则会将您编写的代码覆盖；*

- **时钟初始化配置代码：**位于 **Users/main.c** 中的 **SystemClockInit** 函数；如果您在上一步没有手动勾选**尝试根据配置的 TSI 模块时钟自动生成时钟初始化代码**选项，需要手动配置时钟初始化使其符合您设定的模块工作时钟；

生成工程中，main.c 和 tsi\_debug\_io.c 中包含形如下方的用户代码块：

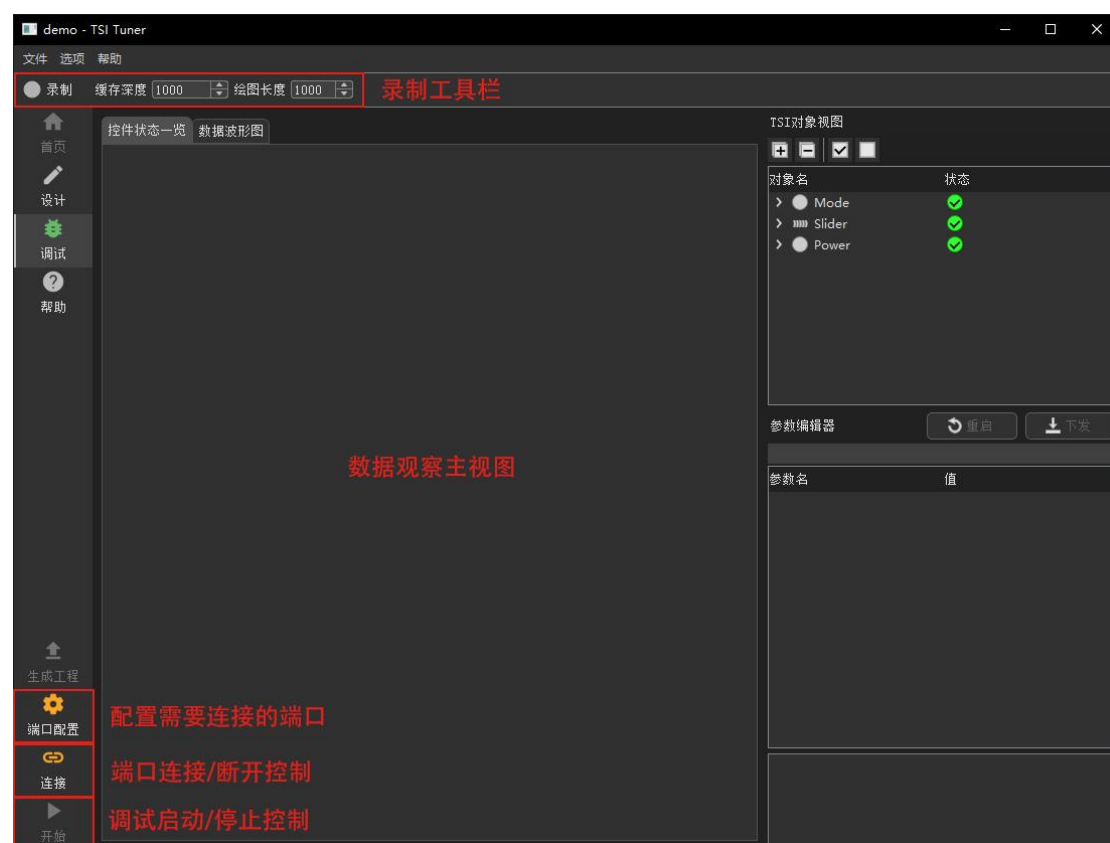
```
/* USER ... BEGIN */  
  
...  
  
/* USER ... END */
```

这些代码块在重新生成工程的时候内容会被保留，可以用于添加一些用户想要额外做的工作。请您务必把自己额外编写的代码放置在这些代码块中，否则可能丢失。

完成上述代码补充后，编译并下载工程到芯片中，我们就可以进行参数调试了。

## 调试

单击窗口左侧的**调试**按钮，进入到调试界面。在该界面，我们可以观察目标板各个触摸按键的信号强度和当前状态，并根据情况来调节参数。下面对调试页面做一下简单介绍。



位于调试界面中央的是**控件状态一览**界面和**数据波形图**界面，可以通过上方的标签切换。控件状态以比较直观的方式展现所有观察控件的状态，数据波形图则用于详细观察每个传感器的信号波形以及控件的状态。数据波形图界面里的每个波形的右上角都有两个按钮：小尺寸/全尺寸按钮用于使波形图在大图和小图之间切换，方便同时观察多个波形；隐藏/显示按钮

---

则用于隐藏该波形，方便挑选感兴趣的波形进行观察。

页面顶部有一个工具栏，可以操作数据录制，以及配置数据缓存深度和波形图的绘图长度。

## 连接芯片并开始调试

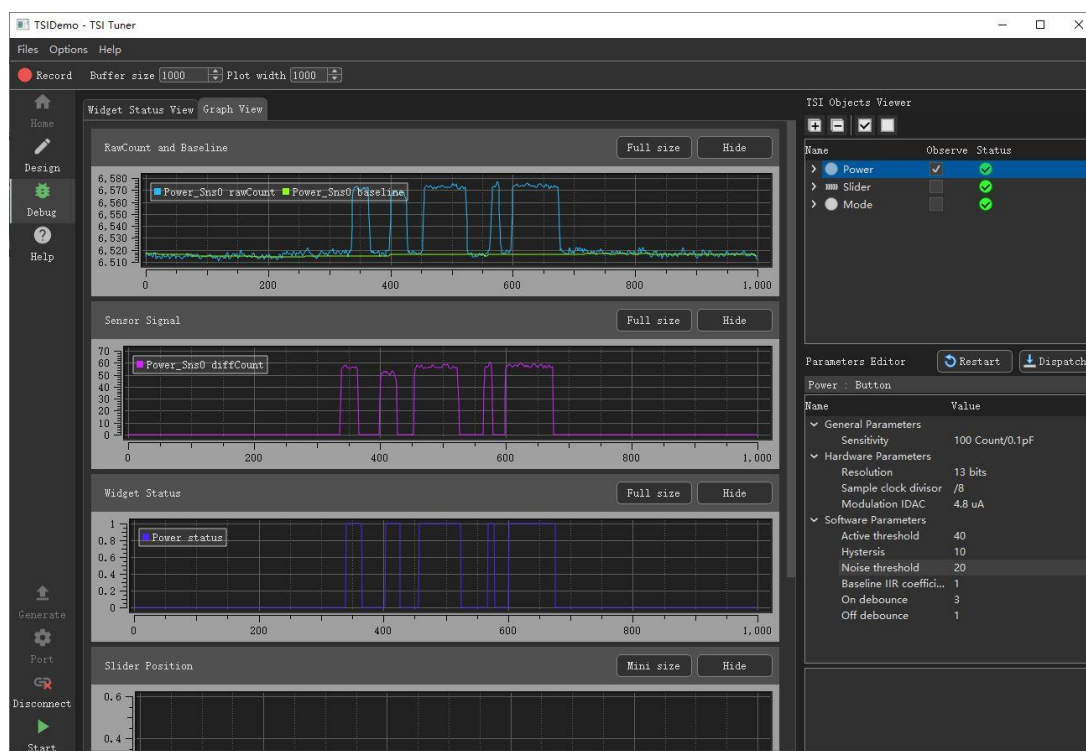
要连接到芯片调试接口，可以单击左侧工具栏的**端口配置**按钮，会弹出一个对话框。在选择好需要连接的端口和配置之后，单击确定离开对话框。之后单击**连接**按钮进行连接，如果芯片程序正常运行，**开始**按钮将会变为活动状态，同时**连接**按钮变为**断开连接**按钮，这代表连接芯片的操作已经完成。

接下来，我们单击**开始**按钮，如果运行成功，**开始**按钮会变为**暂停**按钮，这代表着已经开始调试流程。

## 选择观察对象

在连接到芯片后，可以注意到右侧的**TSI 对象视图**面板的列表会多出一栏：**观察**。这个栏目是用于选择需要观察哪些控件和传感器对象的，只有我们勾选了的对象，它的波形和状态才会显示在视图中。

在示例应用中，我们来观察一下 Power 按键的波形和状态：在**TSI 对象视图**窗口勾选 Power 按键的**观察**选项，它所包含的传感器 **Power\_Sns0** 的观察选项也会被自动勾选。我们切换到**数据波形图**界面，就可以看到相应的波形了。可以尝试按下按键，对应波形图会发生相应变化。

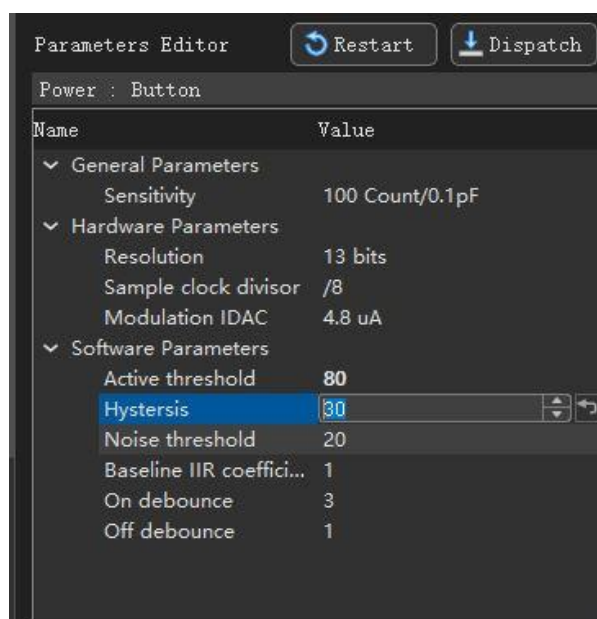


## 调节参数

在调试过程中，如果观察到的波形和状态不符合预期，我们可以对参数进行调节。在窗口右侧的 **TSI 对象视图** 面板中，选择需要调节参数的对象，该对象的参数会展示在下方的 **参数编辑器** 面板中。单击想要修改的参数值，即可进行修改。在所有参数修改完成后，单击右上角的下发按钮，可以将参数下发到芯片。参数编辑的展现效果如下：

- 修改后的参数值如果和原来的值相比有变化，会加粗显示；
- 参数编辑框右侧的撤销按键可以用于撤销当前作出的更改（可以撤销时，箭头会变为红色）；





关于每个参数具体的含义，详见 [TSI 触摸检测原理和算法](#) 文档。

在调节到满意参数后，我们可以返回设计工程来保存参数。点击左下角断开连接按钮断开调试连接，然后点击**设计**按钮返回设计视图。这里如果您在调试过程中修改了参数，会弹出一个参数比较对话框让您确认参数变动。如果您不想修改某项参数，可以取消勾选这项参数的更新选择框，这样参数就不会更新到工程中。

在每次调参之后，可能当前配置不能满足您的需求。在这种情况下，您可以在设计模式中重新配置模块参数。需要注意的是，在重新配置完成后，您也需要重新生成调试工程，编译并下载到芯片中，这样新的参数才能够生效。

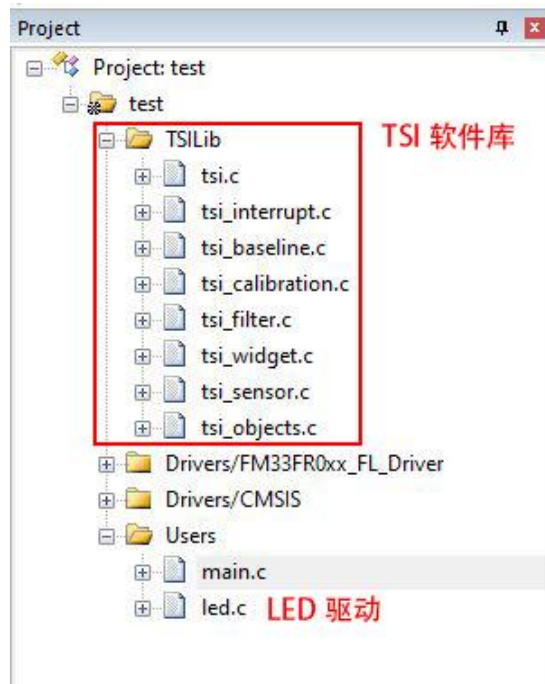
## 导出软件库

在调节完成参数后，我们可以导出包含参数的，可直接用于实际应用工程的软件库。和导出调试工程类似，我们还是单击左侧工具栏中的**生成工程**按钮，打开导出对话框。之前在[生成调试工程](#)中已经介绍了导出对话框，我们这里直接选择导出目标为**库**，并配置导出路径，然后直接单击**导出**按钮即可将包含您当前参数的软件库导出到指定文件夹中。您可以将库嵌入到您的应用工程中，加入对应函数调用即可使用。

我们将调好的示例工程参数导出到一个独立的 TSiDemoCode 文件夹，如下图所示：



打开导出文件夹，可以看到 TSI 软件库已经导出完成（位于 TSILib 文件夹内）。接下来我们只需要为我们的工程添加 TSI 软件库和应用代码逻辑，就能实现所需的功能了。我们建立一个 FM33FR0510 的工程，编写一下按键背光 LED 和状态 LED 的驱动。然后，将导出的 TSI 软件库添加到工程。最终工程目录结构如下：



接下来，我们编写 main.c 文件，调用 TSI 软件库接口，完成 TSI 软件库的初始化/启动，以及按键状态读取、背光控制和三色彩灯调色调亮度的功能。我们将部分代码片段展示如下，完整工程请见 [TSI 彩灯 DEMO 工程](#)。

- **TSI 软件库初始化和启动：**

```
/* Init TSI */
TSI_Init();

/* Enable all widgets */
TSI_Widget_EnableAll();

/* Start TSI */
TSI_Start();
```

- **读取按键状态：**

```
if(TSI_WidgetList.mode.buttonStatus)
{
    /* 按键按下 */
}
```

- 读取滑条状态：

```
if (TSI_WidgetList.slider.base.status)
{
    /* 滑条被按下，读取当前位置数据 */

    uint8_t center = TSI_WidgetList.slider.centerPos;

    /* 根据滑条位置数据 center 进行处理 */
}
```

## 最终效果

最终实现的效果为：通过按下 Power 按键，可以控制彩灯电源开启/关闭；在彩灯电源开启后，通过按下 Mode 按键来切换要调节彩灯色彩（HSV 模式）还是亮度；选择模式后，可以通过 Slider 滑条进行调节。

- 按下电源（Power）按键，彩灯（DEMO 指示灯）点亮，默认为红色：



- 初始状态时，默认调节的是彩灯的色彩，通过滑条可以进行调节：



- 单击模式 (Mode) 按键，可以切换到调节亮度模式 (Mode 灯会保持常亮，提示我们处于调节亮度模式)，通过滑条调节亮度：

