TSI 软件库介绍

复旦微电子提供的 TSI 模块配套软件库提供一套完整的基于中断的按键扫描应用库，结合复旦微的 TSITuner©上位机软件，为用户提供了快速实现触摸按键的方式。目前软件库支持的控件包括按键控件、接近感应控件（可以用来制作非接触按键）和滑条控件，可以覆盖大多数常用应用。

TSI 软件库文件结构

TSI 软件库为单目录结构，库中的文件罗列如下:

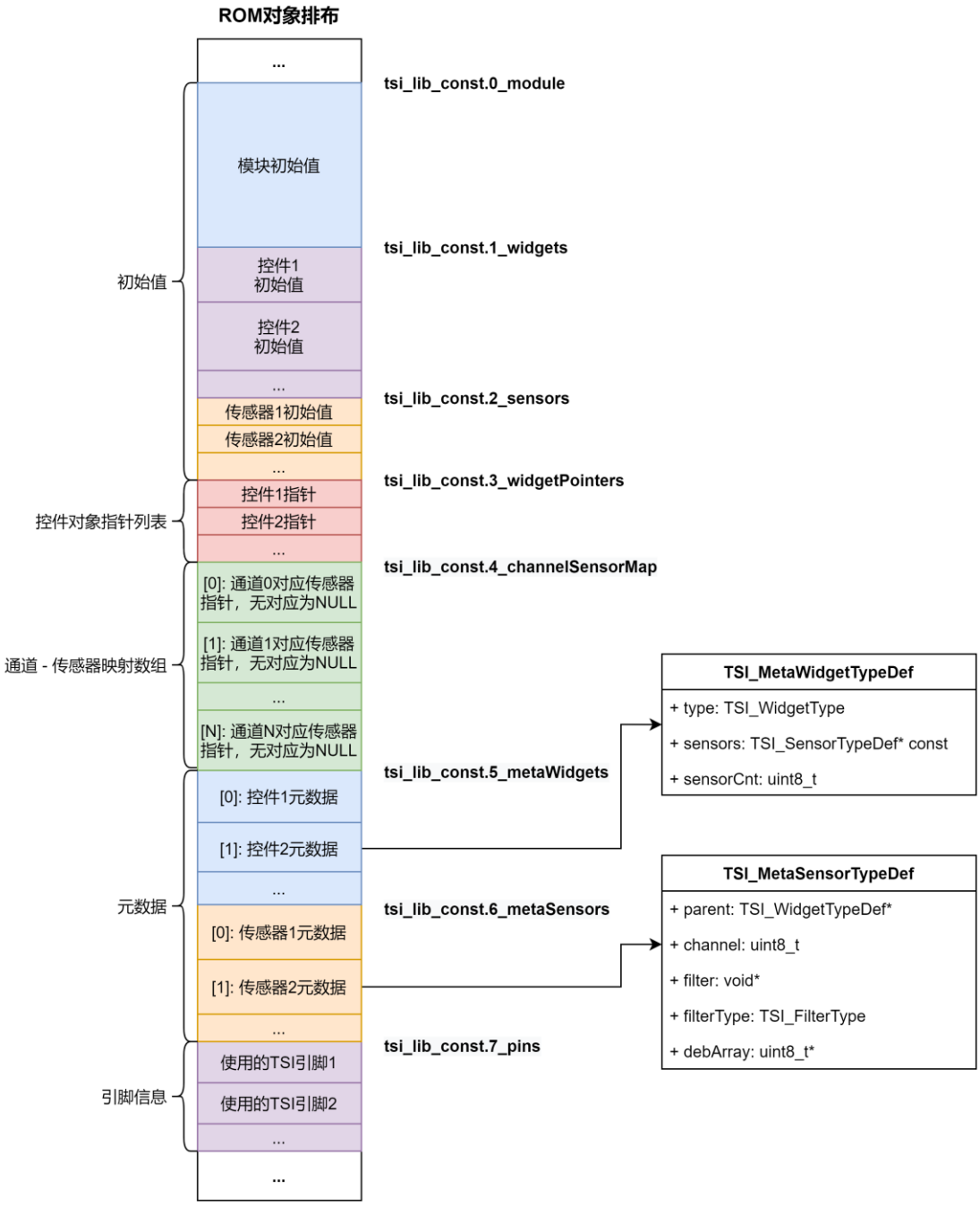
1. **tsi.c，tsi.h**：包含库的运行控制函数（启动、停止等）；
2. **tsi_baseline.c，tsi_baseline.h**：包含基准线算法相关实现；
3. **tsi_calibration.c，tsi_calibration.h**：包含触摸硬件参数自动校准算法的相关实现；
4. **tsi_conf.h**：用户配置文件，TSI 软件库的一些可配参数都罗列于此。我们仅提供对应该文件的模板 **tsi_conf_template.h**，用户修改后重新命名为 **tsi_conf.h** 即可使用；
5. **tsi_def.h**：包含库所需的一些定义，包括一些配置可选参数值的定义；
6. **tsi_filter.c，tsi_filter.h**：包含 RawCount 滤波器的实现；
7. **tsi_interrupt.c**：包含 TSI 模块的中断服务程序；
8. **tsi_objects.c，tsi_objects.h**：tsi_objects.c 包含全部的 TSI 模块控件定义和参数，tsi_objects.h 包含全部 TSI 软件库数据结构定义；**tsi_objects.c** 我们仅提供对应的模板 **tsi_objects_template.c**，用户修改后重新命名为 **tsi_objects.h** 即可使用；
9. **tsi_sensor.c，tsi_sensor.h**：包含传感器使用的配置、初始化、更新状态函数；
10. **tsi_widget.c，tsi_widget.h**：包含控件使用的配置、初始化、更新状态函数；

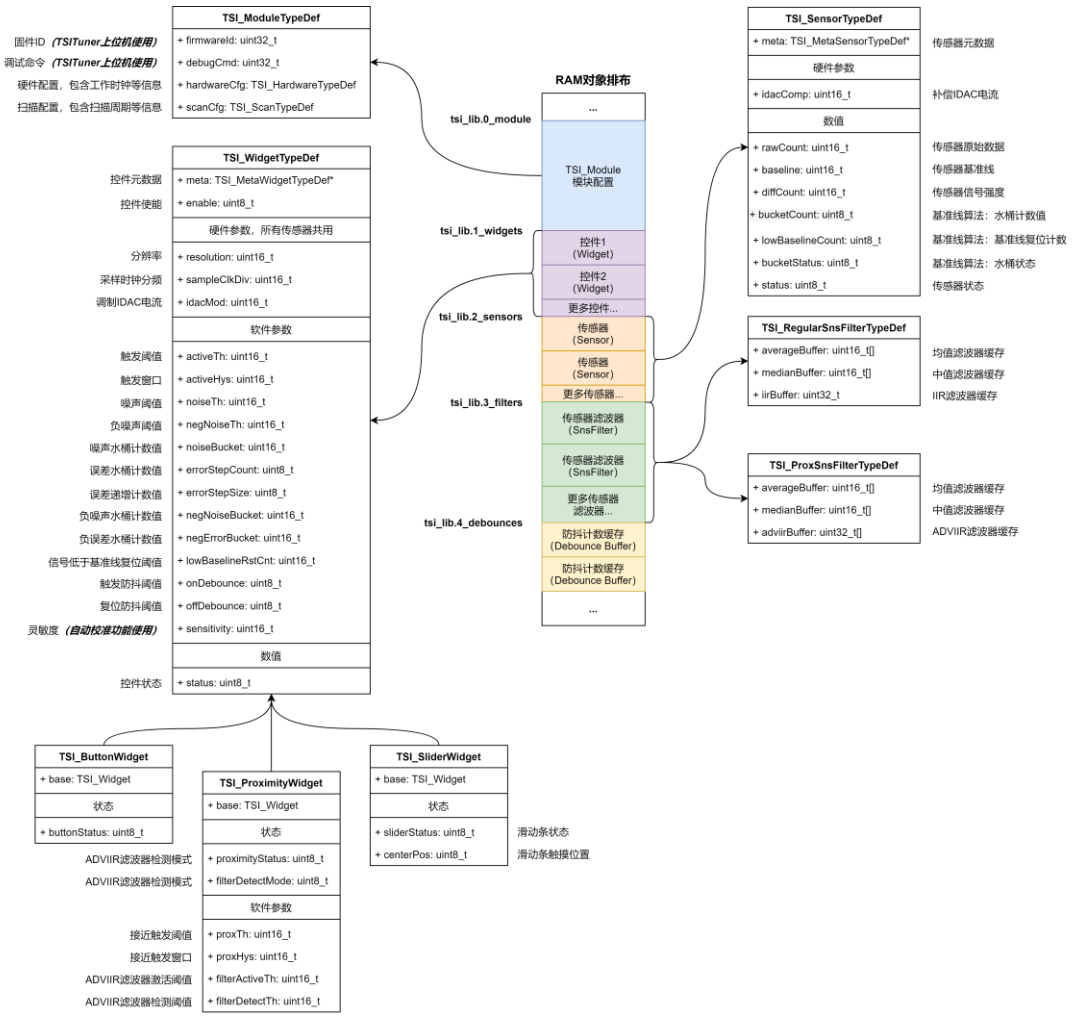
TSI 软件库使用方法

用户需要首先完成软件库基本配置以及控件配置，才能正常使用软件库。需要配置的文件为 tsi_objects.c 和 tsi_conf.h。接下来我们结合相关文件来梳理一下 TSI 软件库的基本数据结构和运行逻辑，方便您快速上手。特别建议您使用我们推出的 TSITuner©上位机软件，可以使用图形化方式自动生成这配置文件和对应的调参专用工程，可以极大地方便您的控件配置和调参工作，加快产品设计的速度。

tsi_objects.c 文件中包含所有 TSI 软件库中用到对象的定义。TSI 软件库在 RAM 和 ROM 中

分别占据一块连续的空间，下图详细的展现了 TSI 软件库在 RAM 和 ROM 中的内存分布以及各数据结构的定义：





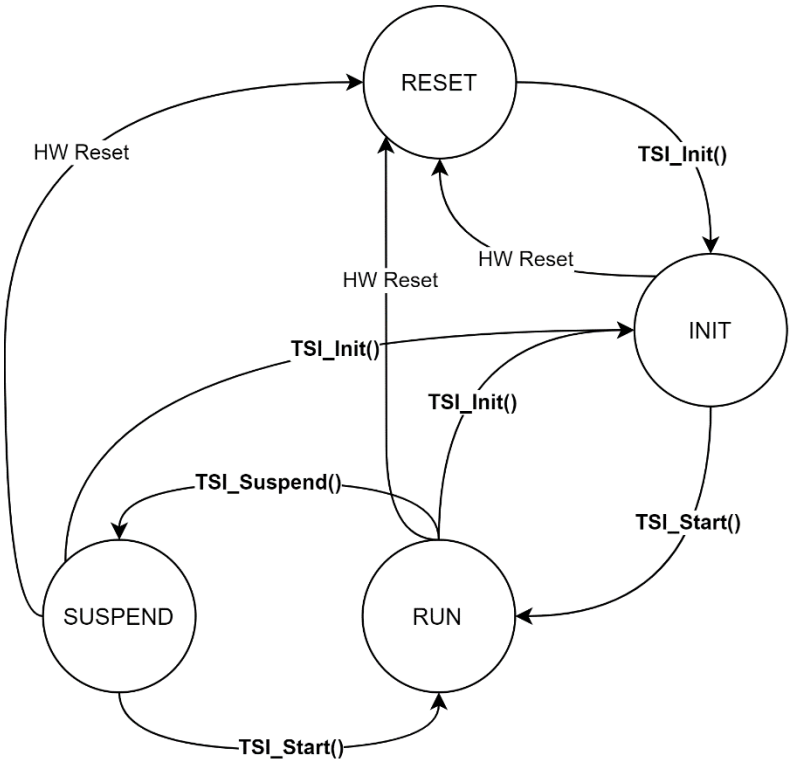
上图中所有的数据结构定义可以在 tsi_objects.h 中找到，而 tsi_objects.c 中则定义了所有的 RAM/ROM 对象。

TSI 软件库的核心对象为模块（**TSI_Module**）、控件（列表，**TSI_WidgetList**）和传感器对象（列表，**TSI_SensorList**），它们在逻辑上拥有层次关系，一个模块对象拥有一个或多个控件对象，每个控件对象又拥有一个或多个传感器对象。**tsi_objects.c** 中，对每个对象都有相应的初始值定义，这些初始值在 TSI 软件库初始化的时候加载到位于 RAM 中的对应对象中。

为了维护核心对象间的层次关系和提供可拓展性，每个控件（Widget）和传感器（Sensor）对象都有相应的元数据信息：**控件的元数据**包括**控件类型**和**所拥有的传感器对象**等；**传感器的元数据**则包括其**父对象（控件对象）**和**滤波器指针和类型**信息、**消抖缓存指针**和**通道**信息。控件通过元数据信息实现**多态**，不同控件间使用控件类型区分，具有很好的拓展性。

tsi_conf.h 文件中包含 TSI 软件库的功能配置、控件和通道使用信息以及控件和传感器列表的数据结构定义。

TSI 软件库共有 4 种执行状态：复位状态，初始状态，运行状态，暂停状态。状态机如下图所示：



- **TSI_Init()函数：初始化**
  - 首次执行初始化时，初始化时钟和使用的引脚；
  - 使用初始化参数初始化各个 TSI 对象；
  - 根据 TSI 对象的配置，将硬件参数应用到硬件；
  - 执行硬件参数自动校准（在使能校准情况下）
  - 执行初次扫描，初始化基准线和滤波器；
- **TSI_Start()函数：启动**
  - 使能中断并开始扫描
- **TSI_Suspend()函数：暂停**
  - 关闭扫描并禁止中断
- **TSI_Resume()函数：继续，功能同 TSI_Start()函数**

TSI 软件库在启动后便会持续进行扫描，TSI 软件库使能了 TSI 硬件的序列完成中断和通道完成中断，用于接收和处理数据。相关代码可以在 **tsi_interrupt.c** 中找到。在 TSI 序列完成中

断触发后，TSI 软件库会置位**扫描完成标志**，可以通过执行 **TSI_GETSTAT_SCAN_CPLT()**获取该标志位状态，并使用 **TSI_CLRSTAT_SCAN_CPLT()**来清除该标志位。当用户获取到**扫描完成标志**置位之后，需要执行 **TSI_Widget_UpdateAll()**函数，该函数会依据获取到的扫描数据对所有控件进行一轮更新。之后，用户就可以根据当前控件的状态来执行操作了。各类控件获取状态的方法如下：

1. 按钮控件（Button）：通过 **TSI_WidgetList.[控件名].buttonStatus** 获取状态，0 为无事件，1 为有触摸事件；

2. 滑条控件（Slider）：通过 **TSI_WidgetList.[控件名].sliderStatus** 获取状态，0 为无事件，1 为有触摸事件；通过 **TSI_WidgetList.[控件名].centerPos** 获取滑条触摸位置，范围为 **0-255**；

3. 接近感应控件（Proximity）：通过 **TSI_WidgetList.[控件名].proximityStatus** 获取状态，0 为无事件，1 为有接近事件，2 为有触摸事件；