文档编号：AN099

上海东软载波微电子有限公司

用户手册

触控按键芯片

修订历史

版本	修订日期	修改概要
V2.0	2017-8-22	V2.0 发布
V2.1	2017-9-18	添加 TK 通道选择注意事项
V2.2	2017-12-1	修订软件的使用说明。
V2.3	2018-1-12	1）启动界面版本选择介绍 2）添加“通道设置”功能说明 3）添加“坐标系”功能说明 4）添加 Lite 版固件说明 5）添加低功耗版固件说明
V2.4	2018-5-23	1）修改原理图集 PCB 注意事项部分表述，增加原理图及 PCB Checklist 2）修改 Lite 工程使用说明表述 3）修改低功耗工程使用说明表述 4）增加可调参数工程使用说明 5）修改 TK 通道选择注意事项描述
V2.5	2018-6-13	1）修改描述中的错误
V2.6	2018-10-25	1）更新 Lite 版本新功能描述
V2.7	2019-4-25	变更 Logo。

地　址：中国上海市龙漕路 299 号天华信息科技园 2A 楼 5 层
邮　编：200235
E-mail：support@essemi.com
电　话：+86-21-60910333
传　真：+86-21-60914991
网　址：http://www.essemi.com/

目 录

内容目录

第1章 芯片简介

1.1 芯片特点

HR7P201/ES7P202 系列芯片采用了“电容电荷转移”工作原理，内部集成触摸感应按键模块（TKM），最大可支持 14 个触摸按键(ES7P202 芯片最大支持 24 个触控通道)。

该芯片在多种应用环境下能有效识别手指触摸并读取按键状态。

HR7P201 芯片特性：

- HR7P RISC CPU 内核
- 16K Words FLASH 程序存储器
- 1K Bytes SRAM 数据存储器
- 最大 14 个按键通道
- 支持 UART 串行通讯
- 工作电压：3.0V ~ 5.5V
- 工作功耗：小于 2mA@5V
- IDLE 功耗：小于 16uA
- 工作温度：-40 ~ 85℃

1.2 应用领域

HR7P201 系列芯片既可以作为通用 MCU 使用，又可以利用内部集成的 TKM 实现触控按键功能，简化系统设计，降低 BOM 成本，因此可以应用于触摸按键、小家电领域。本公司提供 TKM 驱动库函数，有助于用户对 TKM 的设计。用户根据实际应用的需要，通过调用驱动库函数，完成对按键个数、触摸门限和灵敏度的配置。TKM 驱动库函数能支持多个独立触摸按键。将来还会不断升级完善以满足更多用户系统的应用场景，本固件库同样适用于 ES7P202，具体请查看 ES7P202 数据手册，本手册说明例程以 HR7P201 为主，所有触控功能对于 ES7P202 同样适用。

针对 7P201 的 TK 通道选择需要注意以下事项：

1）在某些恶劣环境下，为保证 TK 各按键的触摸灵敏度，建议不要将复用管脚同时用作 TK 触摸按键输入与 ADC 模拟输入功能。

2）TK0 和 TK1 的信噪比相对于其它通道会低。对于某些恶劣环境，不建议将这两个通道作为 TK 通道。

针对 7P202 的 TK 通道选择需要注意以下事项：

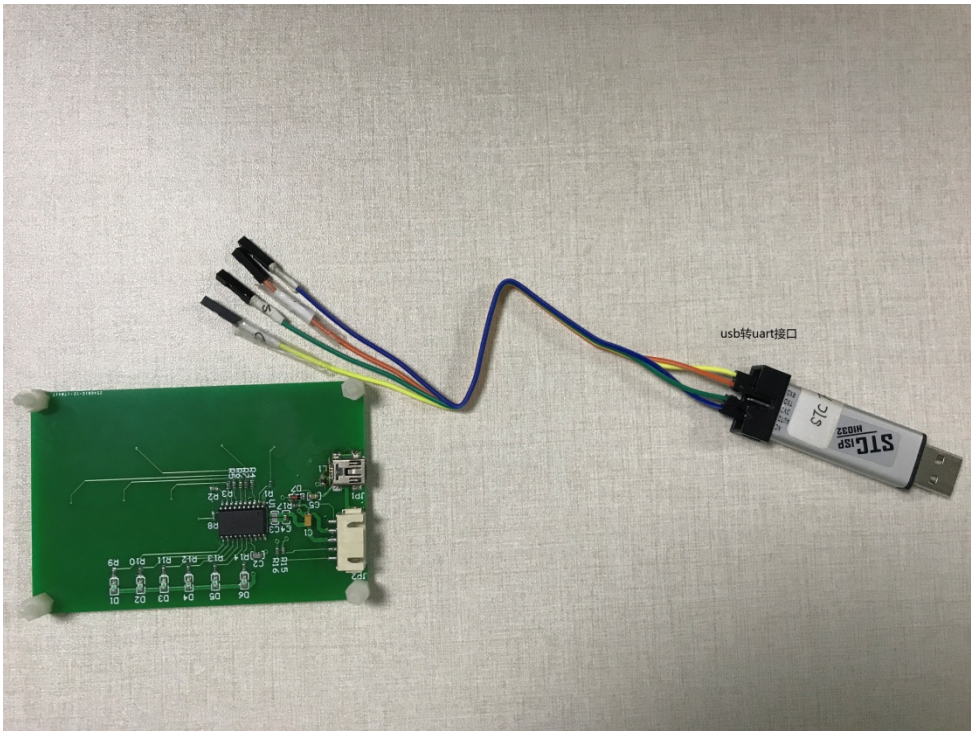
1）在某些恶劣环境下，为保证 TK 各按键的触摸灵敏度，建议不要将复用管脚同时用作 TK 触摸按键输入与 ADC 模拟输入功能。

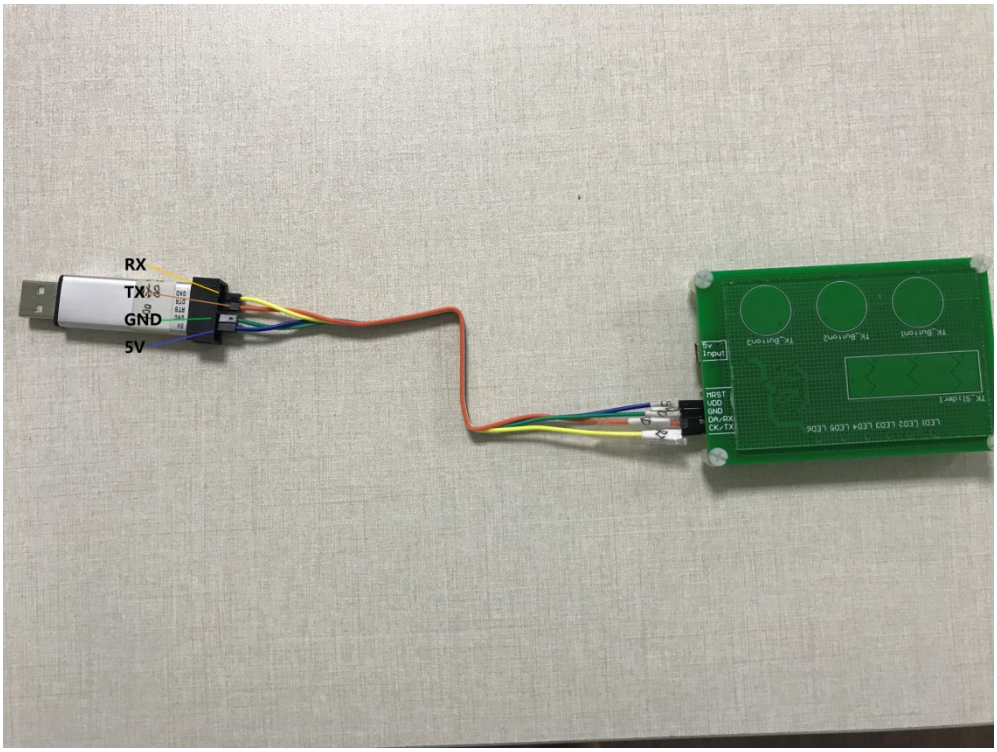
2）TK4 和 TK5 的信噪比相对于其它通道会低。对于某些恶劣环境，不建议将这两个通道作为 TK 通道。

第2章　系统软件

2. 1　硬件连接

采用 USB-UART 转接通信线将 DEMO 板连接至电脑，通过 PC 机 TK-GUI 界面软件对按键性能进行测试，硬件连接方式如下图所示。注意，板载 TX 口连接工具 RX 口，板载 RX 口连接工具 TX 口。





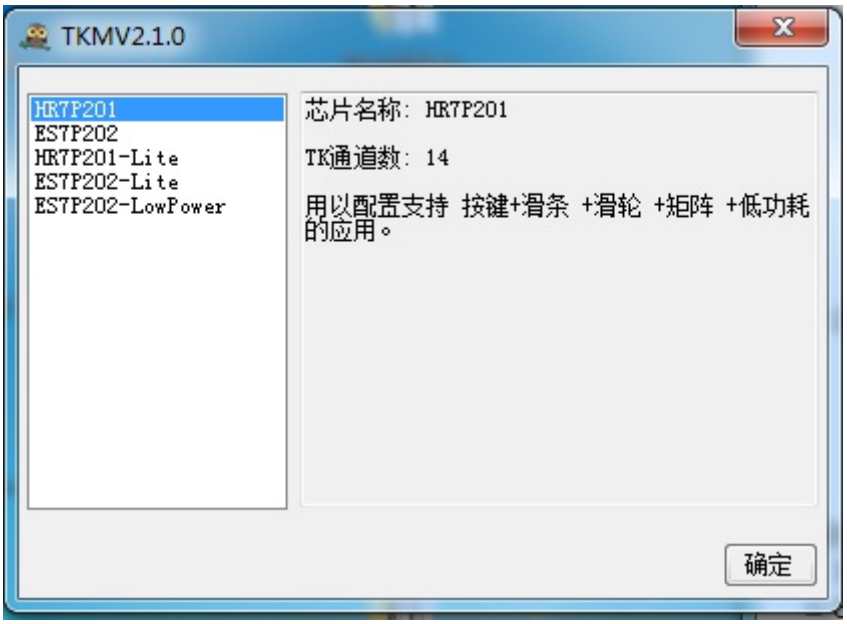
2.2 软件安装

- **支持操作系统**：Win XP 及以上版本；
- **CPU 主频**：1.6GHz，推荐 2.2GHz 以上；
- **其它软件环境**：Net Framework4.0。

本软件无需安装，直接点击 TKMV2.exe 运行应用程序启动软件。

2.3 软件启动

双击应用程序，启动软件，进入芯片选择界面。



目前支持：

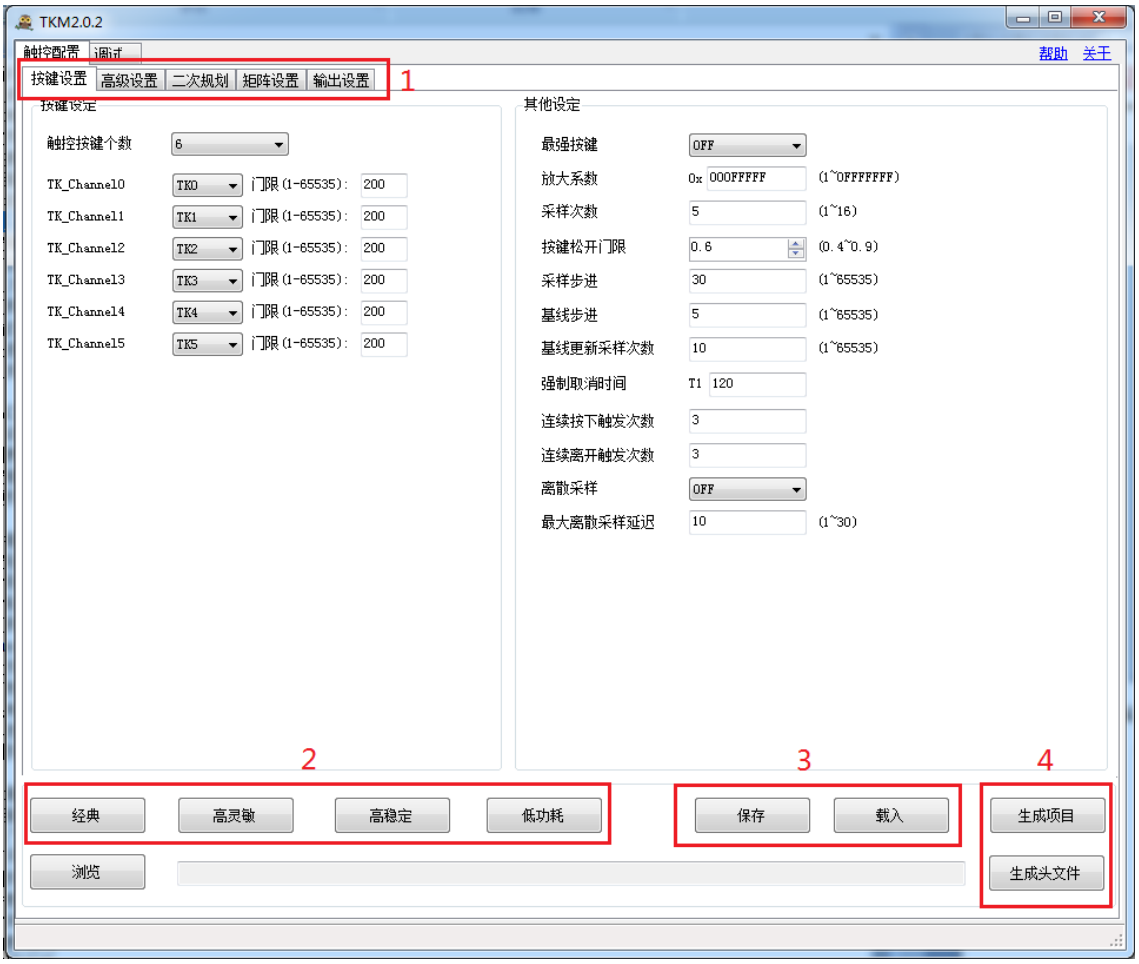
- HR7P201：最多 14 个按键通道，可支持按键、滑条、滑轮、矩阵、低功耗等应用；
- ES7P202：最多 24 个按键通道，可支持按键、滑条、滑轮、矩阵、低功耗等应用；
- HR7P201-Lite：生成支持纯按键功能的固件，更小的 RAM（256 bytes 左右）与 ROM 占用（2K words 左右）；
- ES7P202-Lite：生成支持纯按键功能的固件，更小的 RAM（256 bytes 左右）与 ROM 占用（2K words 左右）；
- ES7P202-LowPower：支持可配置任意数量通道唤醒，功耗更低，任意四个按键唤醒最低功耗小于 10uA；

前两个选项选择后，可以进入配置界面，Lite 版及 LowPower 只支持调试界面，点击“创建项目”，进入 Demo 工程文件夹。

2.4 触控配置

触控配置页可以对触控库的各项参数进行配置，并生成 iDesigner 项目，触控配置页主要分以下几个部分：

- **触控配置**：用户自定义设置按键相关所有参数，支持“按键设置”、“高级设置”、“二次规划”、“矩阵设置”、“输出设置”五个部分。
- **快速设置**：有四种预设模式可供选择，一键设置所有参数。
- **模版保存和载入**：配置完的参数可以保存为模版，下次打开 TKM，即可直接载入，无需调试时反复配置参数。
- **生成项目**：根据所有参数配置自动生成可供运行、烧录的 iDesigner 项目。
- **生成头文件**：可以直接生成参数文件。

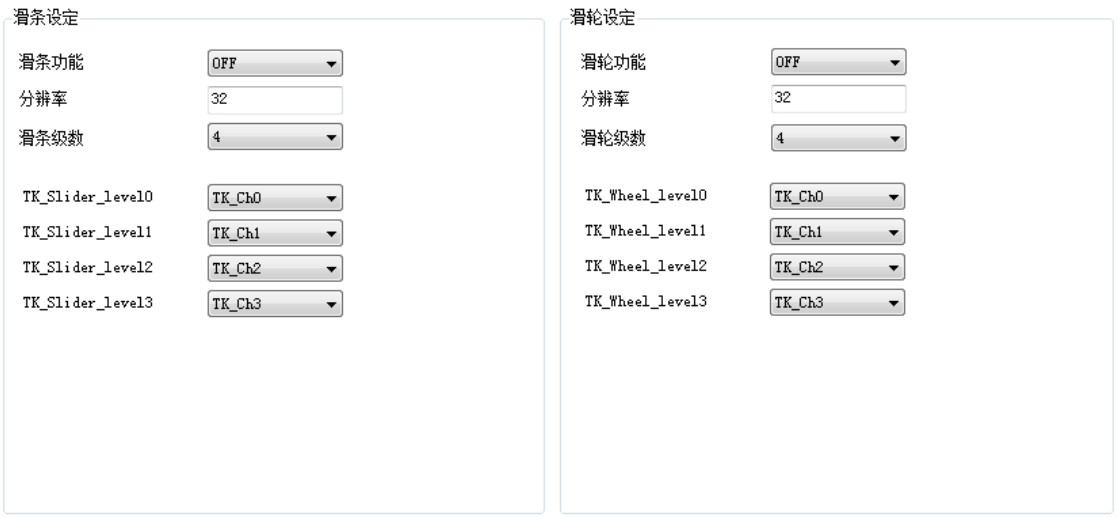


2.4.1 按键设置

按键设定用来配置实际使用按键个数、将实际按键映射到按键通道上，同时设定按键门限越小越灵敏。

- **最强按键**：使能该功能，则只识别多个触摸中信号量最大的按键。禁用时，识别所有超过门限的按键。
- **放大系数**：对应放大系数寄存器，对按键信号硬件放大。
- **按键松开门限**：按键松开时的系数设定，当差值（滤波-基线）低于设定的门限乘以此系数时按键才算离开。越大越灵敏。
- **采样步进**：滤波值更新允许变化的最大值。
- **基线步进**：基线更新允许变化的最大值。
- **基线更新采样次数**：当扫描完成此采样设定次数后，对基线进行更新，越大越慢。
- **强制取消时间**：按键长时间判定为按下时，超过该设定时间，则失效。时间换算公式：T = T1*（T2 * 0.008）s，其中 T1*T2 的值不能大于 0xFFFF。若 T2=125，则 T1 数值单位为秒。
- **连续按下触发次数**：确认按下的防抖判定。
- **连续离开触发次数**：确认离开的防抖判定。
- **离散采样**：使能离散采样，采样参数的时间间隔不固定。
- **最大离散采样延迟**：设定时间间隔偏移的最大范围。

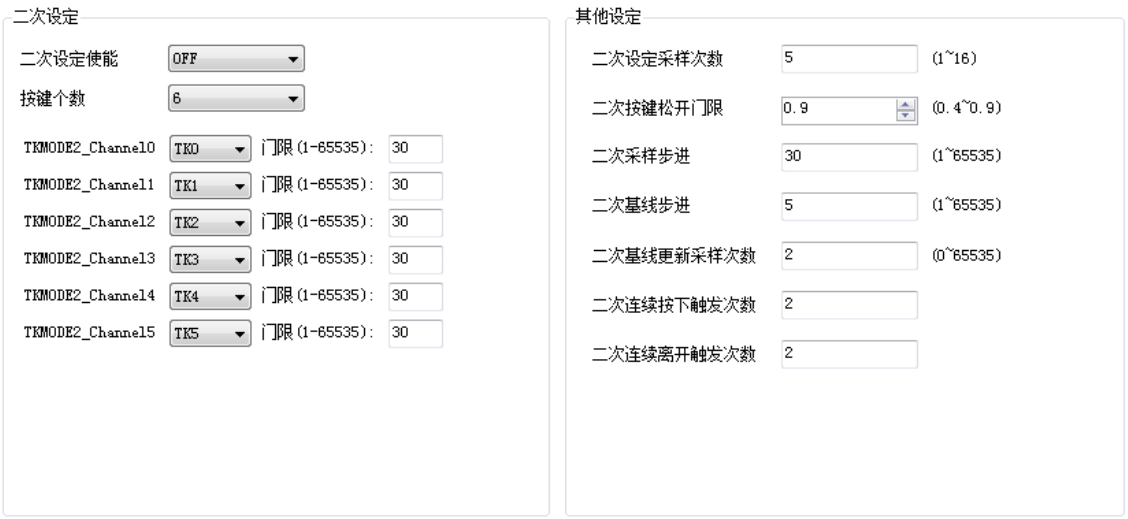
2.4.2 高级设置



滑条设定与滑轮设定配置方法相同。

- **分辨率**：按键间能够实现的差值。
- **滑条级数**：实际用于滑条的按键数。
- **TK_Slider_levelx**：将虚拟按键通道映射到滑条的某一按键上，根据实际硬件设置。滑条的输出结果为 0 ---（滑条级数-1）*分辨率
- **TK_Wheel_levelx**：将虚拟按键通道映射到滑轮的某一按键上，根据实际硬件设置。滑轮的输出结果为 0--- 滑条级数*分辨率

2.4.3 二次规划



二次规划使能后，允许存在两套不同的按键设定配置，典型应用为低功耗扫描按键。

设置参数与按键设置相同。

2.4.4 矩阵设置

矩阵功能实现了矩阵按键的功能，用较少的管脚实现较多按键的功能。矩阵按键输出序号遵循先行后列的规则，例如上图所示 4*4 的按键矩阵。按键序号依次为：1 行 1 列、1 行 2 列、1 行 3 列、1 行 4 列、2 行 1 列、2 行 2 列……以此类推，数值从 1 开始。

2.4.5 输出设置

输出设置实现快速设定管脚点亮发光二极管的功能。

- **LED 模式**：设置按下按键时，LED 的响应为触发模式，或是指示模式。
- **LED_ON**：设置 LED 亮时电平高低。
- **UART 输出使能**：使能后使用默认 uart 管脚通信（201 和 202 均配置在与仿真管脚复用的管脚上），实现和上位机调试界面的通信。
- **低功耗模式使能**：使能低功耗模式，使能的同时需要使能二次规划功能。

- **看门狗休眠时间**：低功耗休眠由看门狗唤醒，设置看门狗休眠时间即确认了低功耗睡眠时间。
- **自动进入低功耗模式时间**：时间换算公式：T = T1*（T2 * 0.008）s，其中 T1*T2 的值不能大于 0xFFFF，此时间决定模式一在无按键操作设定的时间后，进入低功耗扫描模式二。

2.4.6 快速设置

TKM 预设了四种配置，点击快速设置按钮，自动将预设配置值载入配置界面。

为了方便对参数不熟悉用户的使用，设置了四种模式，自动完成所有参数配置，为此我们提供了四种模式可供选择：经典、高灵敏、高稳定、低功耗。

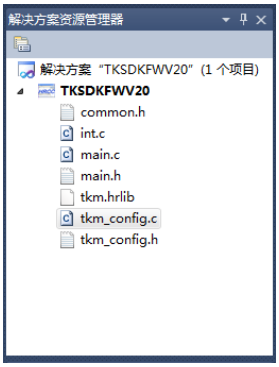
2.4.7 模版保存及载入

当用户通过调试确认一套按键配置参数后，可以将该参数保存为一个.xml 文件，下次打开软件调试时，可以载入该文件，从而避免反复配置相同的参数。

2.4.8 生成项目

点击生成按钮，将根据当前配置自动生成相应 iDesigner 项目。用户可在 iDesigner 中打开该项目，编译并下载程序。

也可以选择生成头文件直接更新参数，可以指向工程文件中的 tkm_config.h 对已存在的工程进行配置参数直接更新。



2.5 调试

调试页面有串口设置及调试控制几个按钮。

2.5.1 串口设置

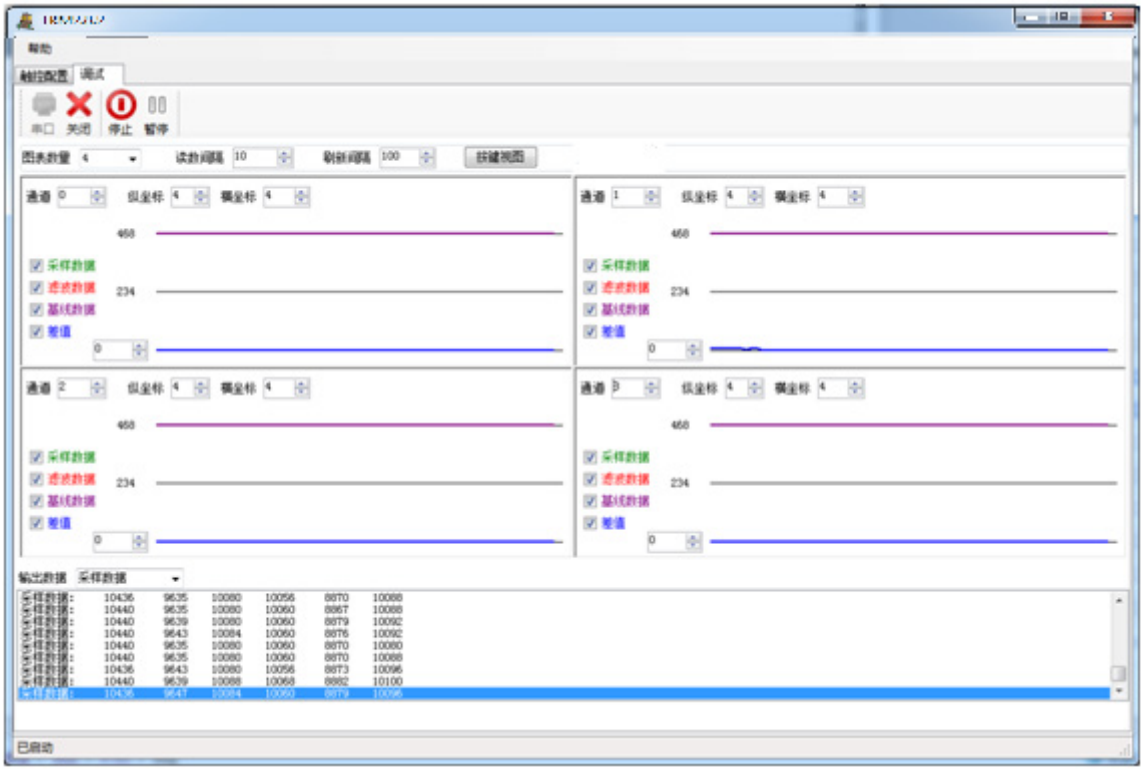
进行调试前先要配置串口，串口连线与烧录口复用。点击“串口”按钮打开串口设置窗口，依次设置串口名、波特率、数据位、校验位和停止位，点击确定。再点击“打开”按钮以打开串口。串口打开后无法再配置串口，必须先关闭已打开的串口。

2.5.2 启动

连接硬件后，点击“启动”按钮进入通讯状态。软件将持续与固件通讯，读取当前触控板状态，并以图表方式展现。

调试主界面用户有以下地方可以配置：

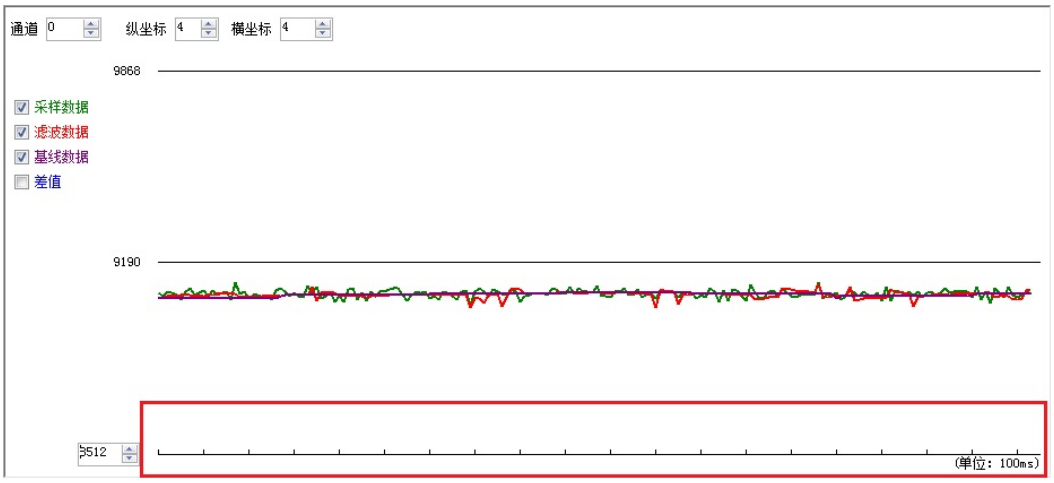
- **图表数量**：主界面显示的图表数量，支持 1、4、6、9 个图表。设置后，相应数量的图表将排列在主界面中，每个图表可以分别显示不同的数据。
- **读数间隔**：与固件通讯的频率。
- **刷新间隔**：图表重绘的频率。
- **输出数据**：输出数据的类型。



启动后用户可以选择暂停或停止。暂停后再恢复，之前的数据保留；停止后再启动，之前的数

据将清除。

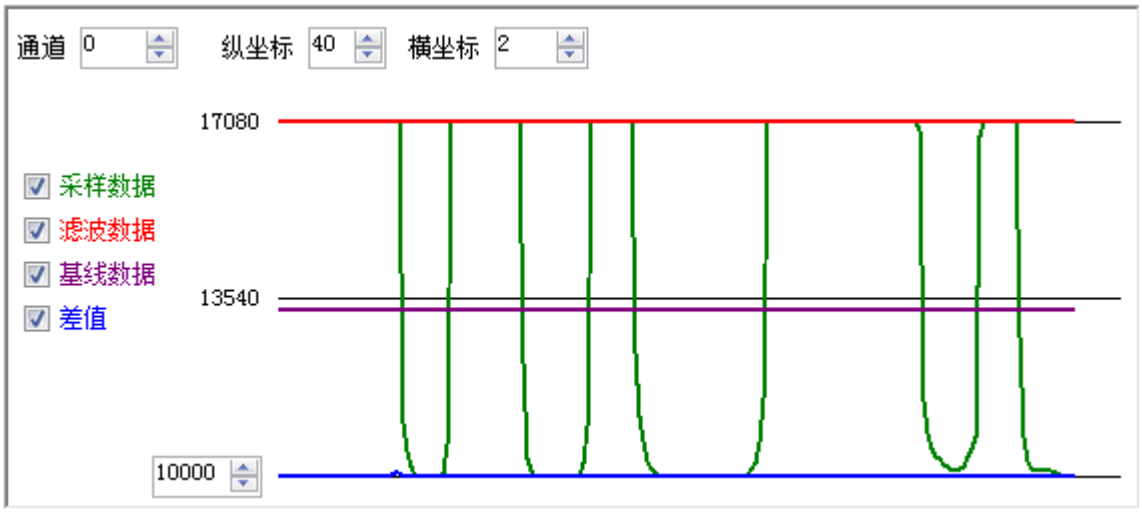
另外，这个版本中，软件新增量尺功能，供用户计量某个事件持续的时间。配合“读数间隔”的设置，可以放大或者缩小该量尺。



2.5.3 图表

图表之间相互独立，互不影响。在图表视图中可以设置以下参数：

- **通道**：选择需要显示的通道，可选通道由启动调试时软件向固件查询而来。
- **纵坐标**：每个像素表示的数值。
- **横坐标**：每组数据之间的 X 轴像素。
- **下限数值**：显示下限的数值。
- **显示数据**：勾选后相应数据将在图表中绘制成对应颜色的线条。



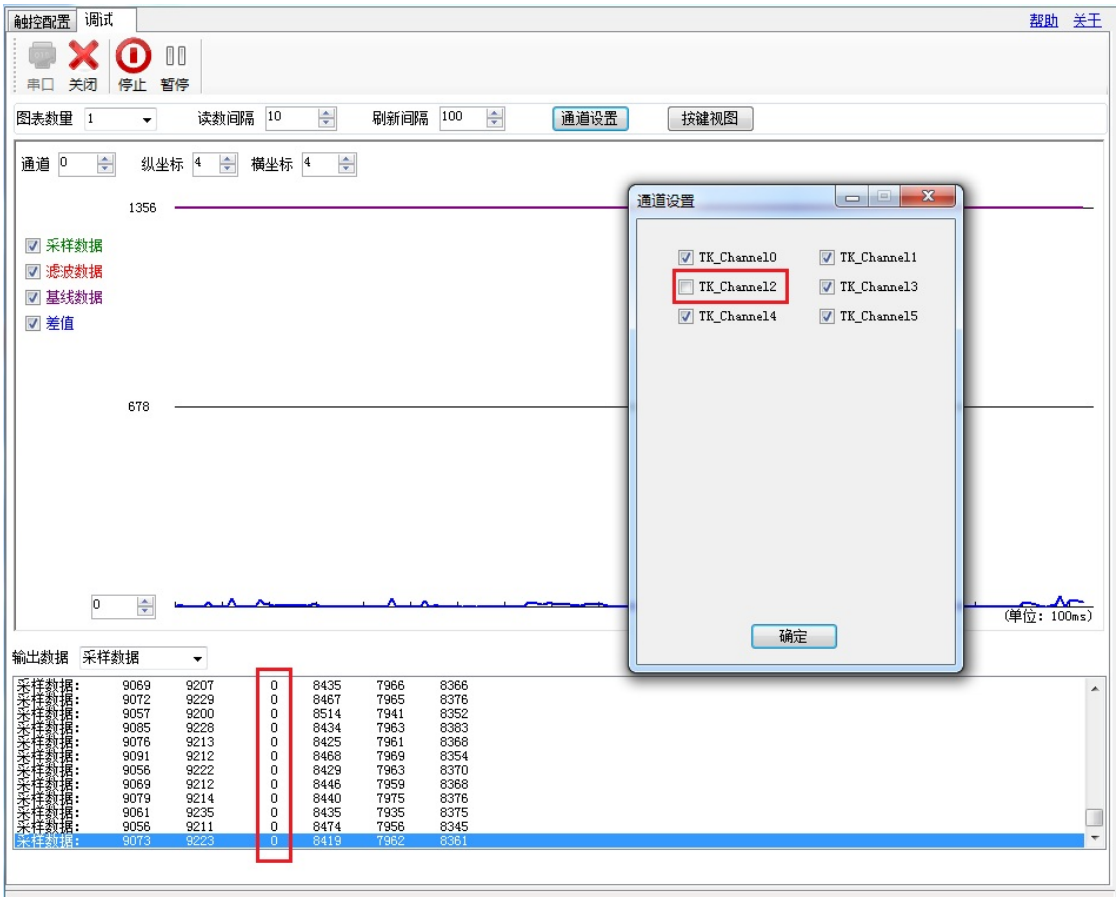
2.5.4 通道设置

点击“通道设置”按钮，打开通道设置视图面板，该面板中可以对需要监视的通道进行选择。通过关闭不需要监视的通道，可以减少串口通信开销，配合“读数间隔”设置，可以降低待观察通道的数据漏报率，从而获取更丰富的信息。

如下图所示，点击去掉勾选，并按下“确认”按钮后，相应通道数据采集显示为 0，表示该通

道数据不再上传，再次勾选该通道，并按下“确认”按钮后，该通道数据恢复显示。

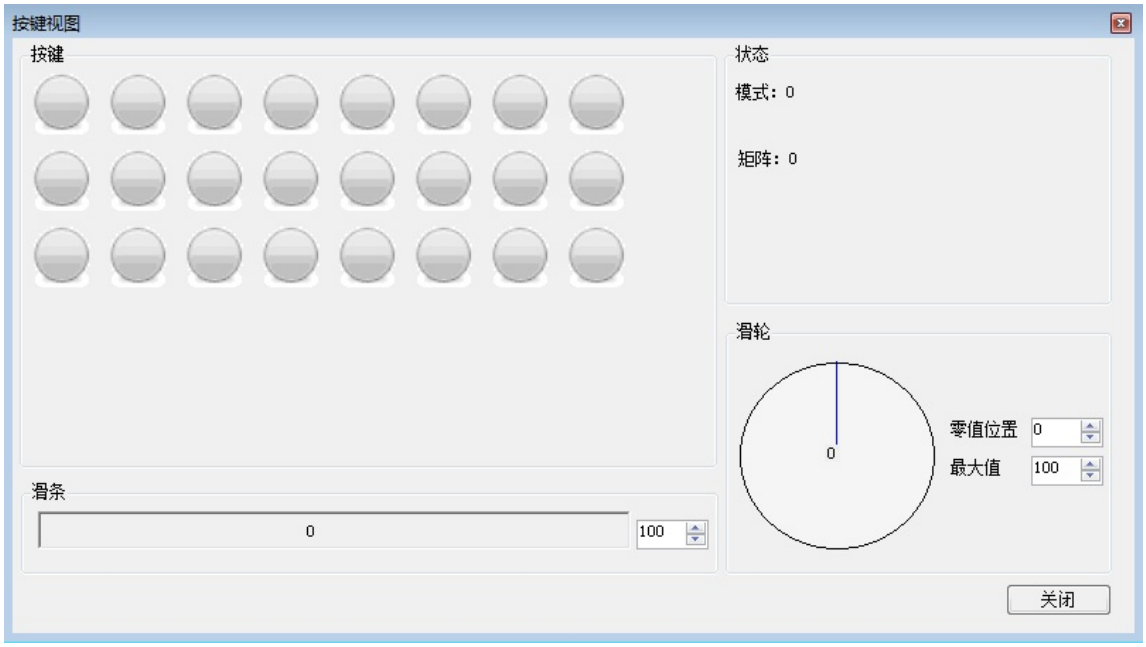
默认情况下，界面显示所有固件配置打开的通道。



2.5.5 按键视图

点击“按键视图”按钮，打开按键视图面板，该面板中可以看到按键、滑轮、滑条的状态，以及当前所处的模式。

需要注意的是，滑轮及滑条的最大值需要根据实际分辨率进行调整，公式见 2.4.2

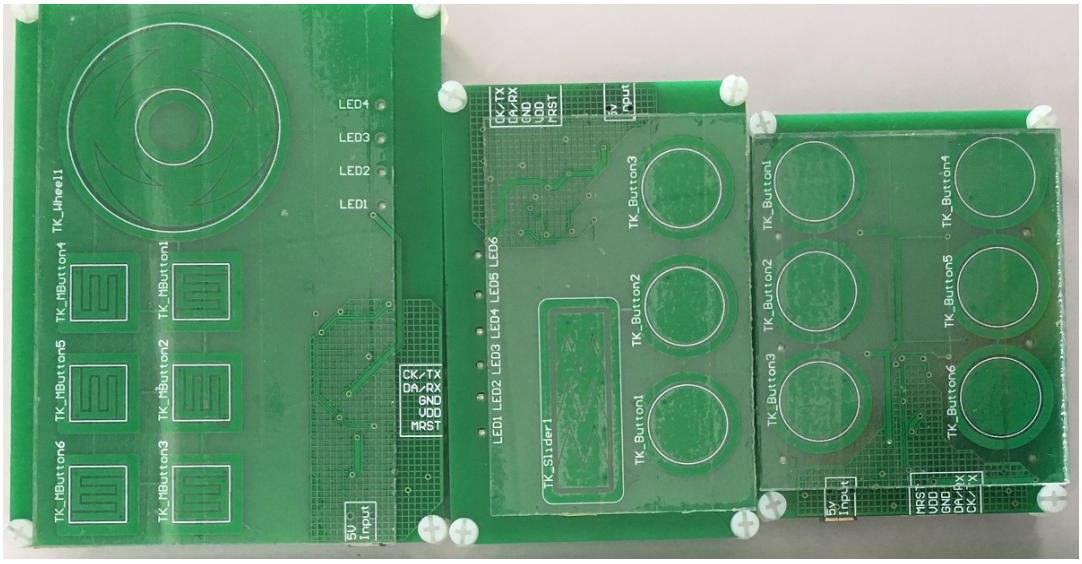


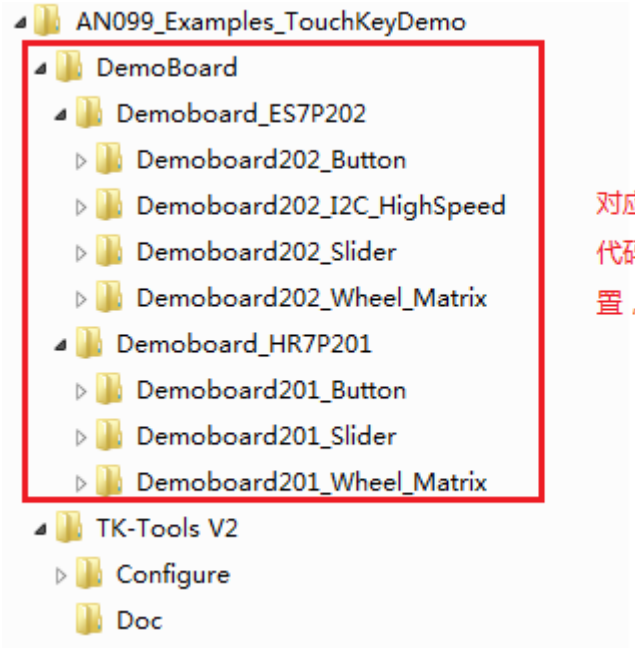
第3章 系统硬件

3.1 概述

为了方便用户评估和开发，提供 3 种触摸按键演示板设计方案供参考(HR7P201 芯片与 ES7P202 芯片各三块)。

设计了 3 块独立的开发板，分别为纯按键、按键与滑条、滑轮与矩阵三种组合。MCU 一共具有 14 个 GPIO 和 TK 通道复用的引脚，这些引脚除连接 TK 按钮外，其余全部用于 LED 控制。HR7P201 的串口与调试接口复用，通过端子引出，用于与调试和通信。实物如下图：





对应每块开发板的源代码，无需修改配置，烧录即可使用。

3.2 触摸按键形状设计

触摸按键分为按钮、滑条、滑轮和矩阵几种样式。

3. 2. 1 按钮

按钮一般被用于检测一次单独的按键操作，按钮的形状有多重，可以被设计为圆形、方形、三角形等，具体形状如图 3-1 所示。图中第二个形状可以用于在触摸按键中间安装指示灯。

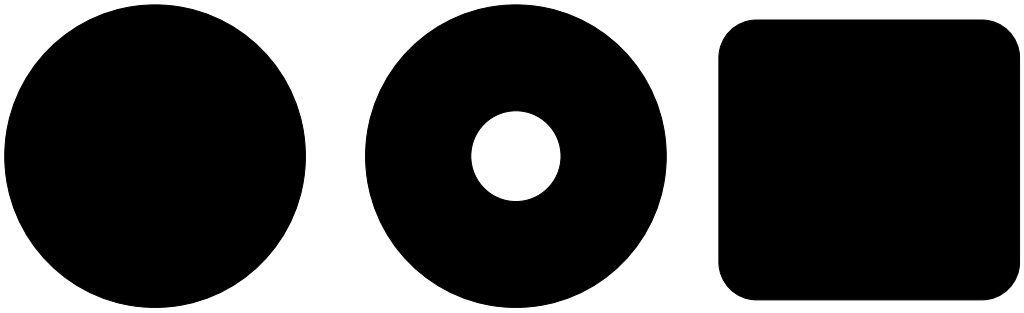


图 3-1 触摸按键形状

按键的尺寸根据正常人手指大小确定，以圆形按键为例，其直径一般设计为 10mm-15mm。图 3-2 中为三种直径的圆形按钮对地距离改变时，按钮对地寄生电容值的变化曲线。

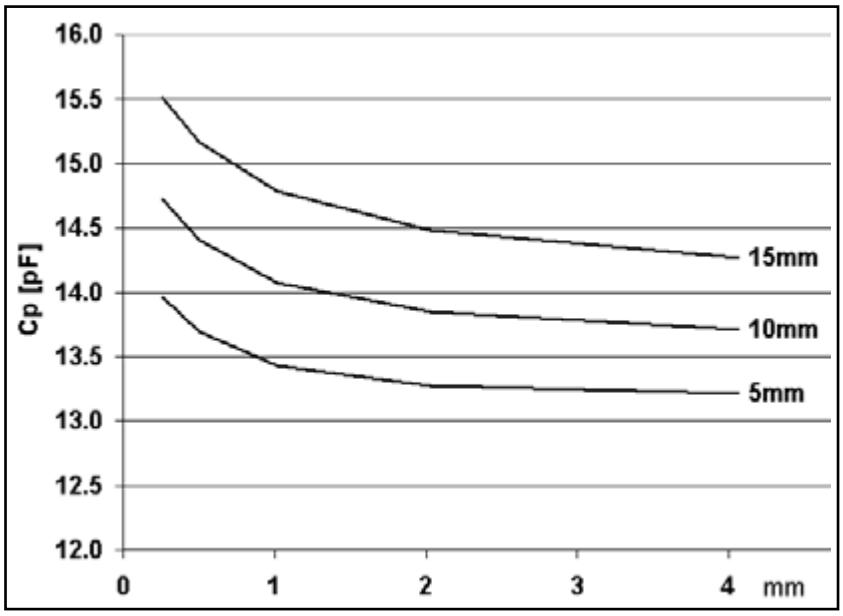


图 3-2 5mm、10mm、15mm 直径按钮对地电容值

3. 2. 2 滑条

滑条可以用于检测手指持续滑动和当前所在位置。理想状态下，当手指在滑条上滑动时，滑条各级电容的变化成线性变化，通过检测相邻两个滑条之间电容的变化值，就可以判断手指滑动距离和方向。按键可以分为多级，级数增加，滑条的长度也随之增加。图 3-3 为一个四级滑条的示意图。

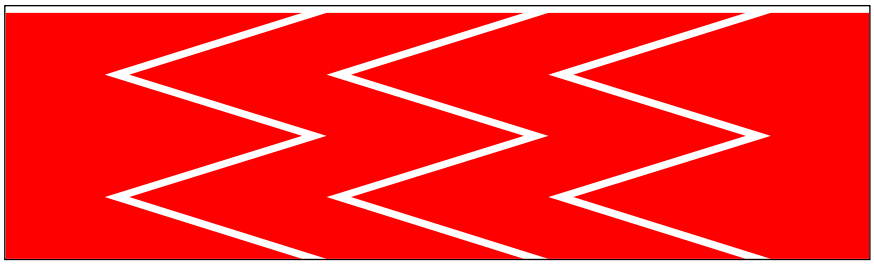


图 3-3　四级滑条示意图

滑条的外形设计需要尽量符合理想状态要求，即手指滑动造成的电容变化是线性的以及同一个时刻，**至少影响覆盖两级滑条的电极**，边缘两个电极未咬合处的宽度建议为半个手指宽度以内，如图 3-4 所示，边缘电极未咬合处长度为 4mm。滑条的具体形状可以按图 3-4 所示尺寸设计。

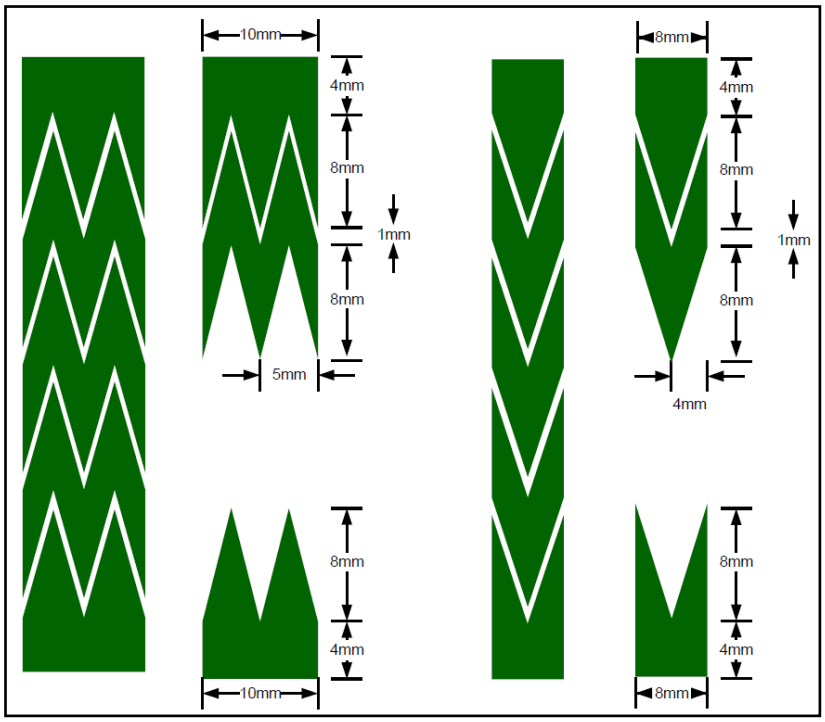


图 3-4　滑条设计尺寸

3. 2. 3　滑轮

滑轮可以理解为一种变形的滑条，用于检测手指的旋转操作，图 3-5 为一个四级滑轮的示意图。

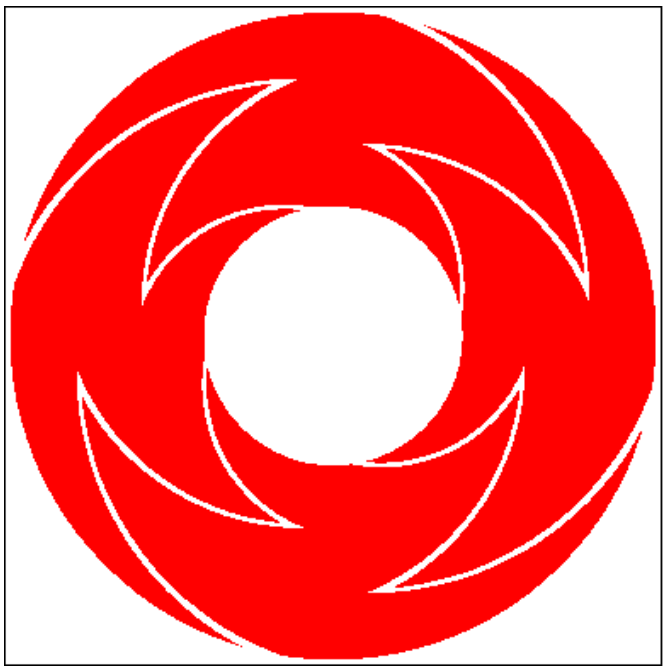


图 3-5 转轮形状示意图

滑轮在设计时，与滑条要求类似，要求手指沿滑轮转动时，造成的滑轮各级电容呈线性变化，并且某一时刻，手指能够对至少两级电极造成影响。在图 3-5 中，滑轮两级之间深度咬合，当手指沿滑轮转动时，如果手指离开第 n-1 级电极的边缘，则立即开始接触 n+1 级电极边缘。在小尺寸滑轮上支持饼状图(菊花状)的滑轮图案设计。

3. 2. 4 矩阵

矩阵按钮可以在芯片 TK 通道有限的情况下，扩展出更多的触摸按键按钮。矩阵按键以行列的方式布置，布置如图 3-6 所示。可以看出，图中一共有 3x4=12 个按钮，但是实际只使用了 Row0~2 与 Column0~3 共 7 个 TK 通道。

在使用时，如果检测到 Row a 和 Column b 两个通道同时被触发，从图中即可判断出，a 行 b 列的按钮被按下。

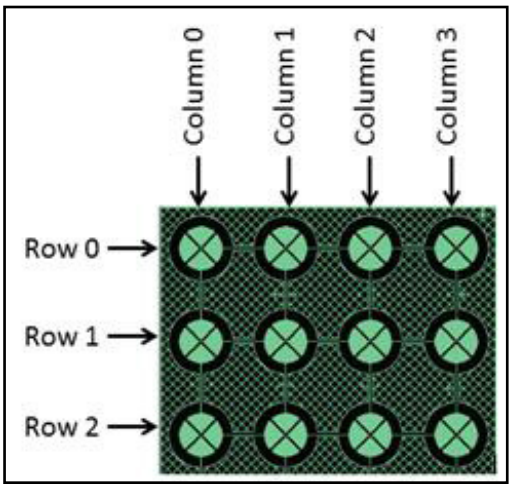


图 3-6 矩阵放置示意图

从矩阵的使用方式可以看出，每一个矩阵按钮都连接有两个 TK 通道，所以矩阵按钮必须被设

计为两个咬合在一起但又互相不连接的电极。一个电极用于连接所在行的 TK 通道，另一个电极用于连接所在列的通道。矩阵按钮可以按如图 3-7 所示设计。

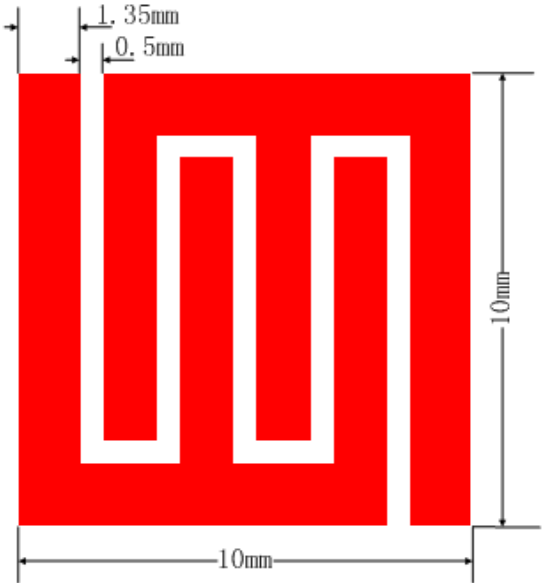


图 3-7 矩阵尺寸示意图

矩阵可以扩展出更多的按钮，但是如果同时按下两个矩阵按钮，有可能被误识别。如图 3-6 中同时按下 row0+column0 和 row1+column1，则 row0、1 和 column0、1 四个通道同时被触发，则有可能被识别为 row0+column1 和 row1+column0。

3.3 原理图及PCB注意事项

3.3.1 走线及封装注意

通过第一章的触摸按键原理可知，触摸按键检测的关键在于检测按下前后按键对地寄生电容C_P的变化，假设触摸操作导致的电容值改变为C_F，则电容变化率可以用以下公式表示：

$$变化率=C_F/C_P*100\%$$

人体造成的C_F值变化是很小的，如果C_P过大，则变化率将变小，过小的变化率对按键检测不利，所以 PCB Layout 的主要目标是保持C_P在合理的水平。

C_P是电极及电极与芯片引脚间走线对地的寄生电容。在电极形状和尺寸固定的情况下，限制走线宽度和走线长度可以降低C_P，一般要求使用宽度为 6mil 的线，线长度尽量短。如果走线串联有电阻，电阻封装应当使用 0603 或 0402。

3.3.2 铺地

合适的铺地可以减少射频干扰，但是靠近按键及其相关走线的铺地会增加传感器的寄生电容，因此，一般情况下需要遵循以下原则：

1. 电极背面禁止铺地。
2. 如果需要铺地，铺地距离电极至少 2mm 以上的距离。
3. 铺地需要使用网格铺铜，网格线宽度为 10mil，间隙为 40mil。

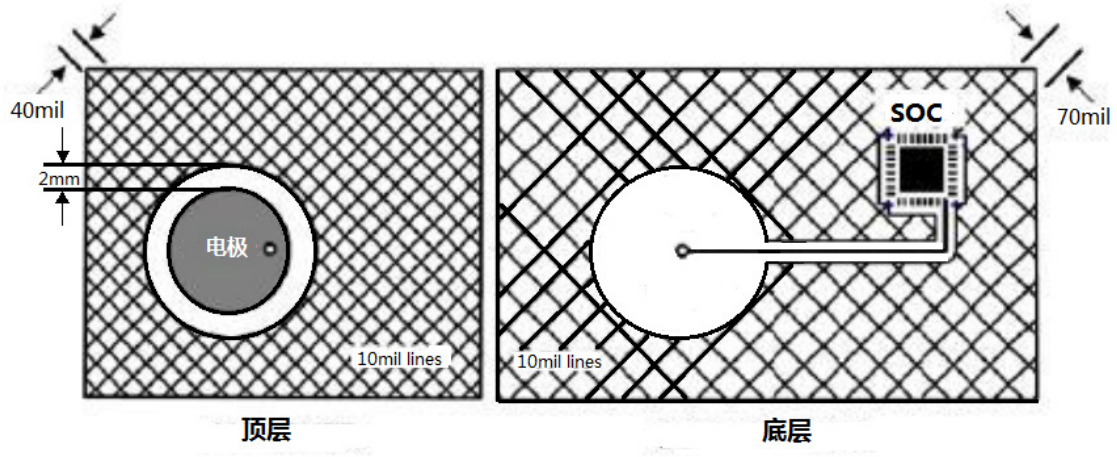


图 3-8　铺地网格示意图

3.3.3　供电

1. 电源纹波过大，可能会影响触控按键的稳定性，应尽量减小电源纹波。
2. 电源线可通过串接磁珠增强 EFT 性能，磁珠应尽量靠近接插件接口位置。
3. 电源线宽不能低于 1mm。
4. 电源线上的去耦电容应尽量靠近芯片的电源和地管脚。
5. 如图 3-9 所示，连到触控芯片上的电源线不要再引出去驱动其它负载。

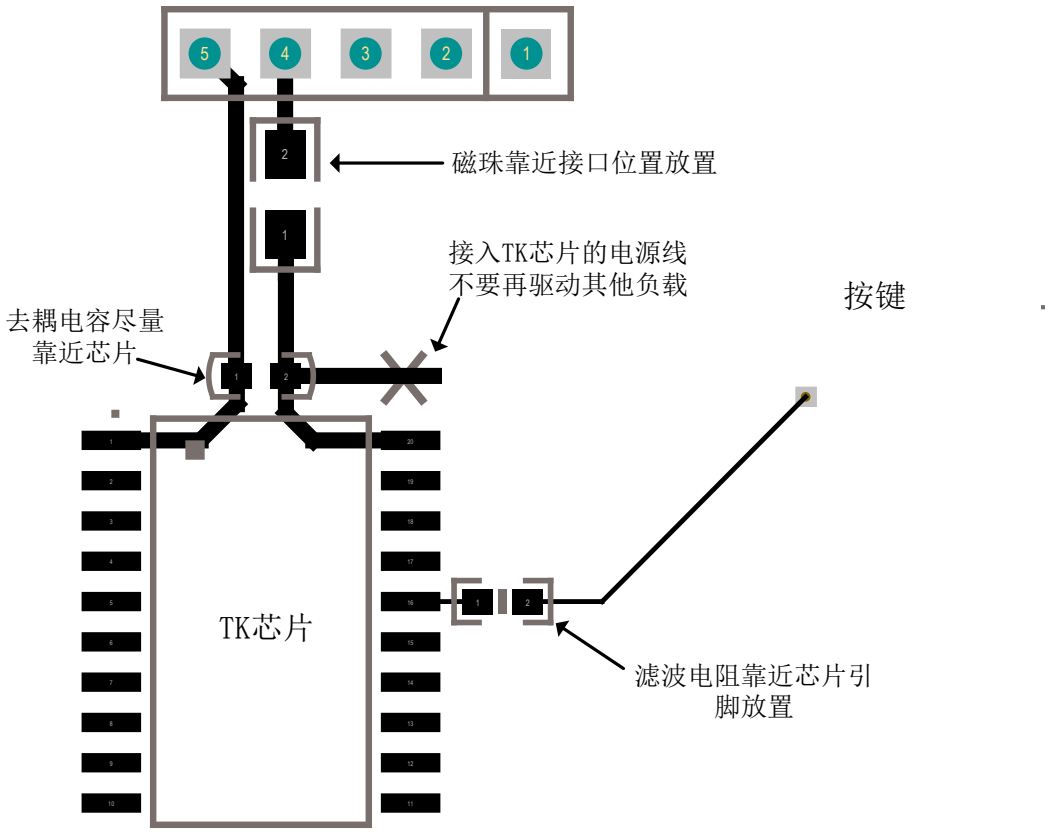


图 3-9　电源线布局示意图

3.3.4　覆盖物

大部分的触控应用都涉及到覆盖物的选择。在计算手指按键产生电容C_F时，使用以下公式：

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

其中：ε_r表示覆盖物的介电常数，可以看出介电常数越大越便于手指接近的检测。表 3-1 列出常见材质的节点常数供用户参考。

材质	介电常数
空气	1.0
玻璃（标准）	7.6-8.0
玻璃（陶瓷）	6.0
树脂玻璃	2.8
ABS	2.4-4.1
木质	1.2-2.5
石膏	2.5-6.0
Mylar®聚酯薄膜	3.2

3.3.5　布局建议

1.　电容选择

参考电容选用温度稳定性高的 X7R 或 NP0 材质贴片电容，靠近芯片放置，其对地走线要尽量短，建议地线从芯片 pin1（VSS）引出。

2.　按键间距

两个触摸按键间的间距尽量放置在 5mm 以上。

3.　EMC 布局规则

- 触控按键引线上的串联电阻尽量靠近芯片管脚；
- 作为专用触控芯片与主控芯片的通信线，每根都要加阻容滤波，一般串接 500 欧姆电阻，并联 1uF 的电容。
- 在 PCB 布线时，电阻电容应尽量靠近接插件。

3.3.6　原理图Checklist

编号	项目	推荐参数或方法
1	电源去耦电容	0.1uF
2	电源大电容	1uF
3	Cx 电容	1-10nF
4	管脚排布	不要让触控传感和通信 IO 混排
5	触控管脚电阻	100Ω-1kΩ
6	通信线串联电阻	500 Ω

7	通信线上拉电阻	10K Ω
8	尽量避免 OSC 管脚用作 TK 功能	

3. 3. 7 PCB Checklist

编号	项目	子项目	最小值	最大值	推荐参数或方法
1	去耦电容（小）			1uF	0.1uF，靠近芯片管脚
2	去耦电容（大）			100uF	10uF，靠近芯片管脚，瓷片电容或电解电容
3	Cx 电容	容值	3nF	10nF	6.8nF
		位置			靠近 Cx 管脚
		材质			X7R、NP0
4	铺地		1mm	2mm	2mm
5	按键	形状	NA	NA	实心圆/圆角长方形
		尺寸	5mm	15mm	10mm
		按键与网格铺地间距	0.5mm	2mm	一般间距与覆盖物厚度成正比，覆盖物越厚，间距越大
6	滑条	宽度	4mm		8mm 至少影响覆盖两级滑条的电极。 边缘电极未咬合处宽度为 4mm
		高度	7mm	15mm	10mm
		按键与网格铺地间距	0.5mm	2mm	一般间距与覆盖物厚度成正比，覆盖物越厚，间距越大
7	滑轮	宽度	4mm		8mm 手指至少需要覆盖两级滑轮的电极。
		高度	7mm	15mm	10mm
		按键与网格铺地间距	0.5mm	2mm	一般间距与覆盖物厚度成正比，覆盖物越厚，间距越大
8	传感器走线	宽度		7mil	6mil
		走线与网格铺地间距	1mm	2mm	2mm
		转角			没有尖锐转角
		走线规则			如果有交叉走线，走垂直方向
9	网格铺地				电极背面禁止铺地。 如果需要铺地，铺地距离电极至少 2mm 以上的距离。 铺地需要使用网格铺铜，网格线宽度为 10mil，间隙为 40mil。

第4章 系统固件

4.1 概述

一方面用户可根据芯片的数据手册自主开发触控按键底层驱动软件，另一方面为了能满足快速应用的需求，用户也可以选用本公司提供的 TKM 驱动库函数，其封装文件为 tkm.hrlib。

4.2 系统环境

iDesignerV4.1.2.138 和 HRCCV1.2.0.71 编译器及以上版本。

较低版本只能针对 HR7P201 芯片进行开发。

4.3 软件安装

从官网下载 V2.0 开发支持包，上位机部分无需安装。

找到 iDesigner 工程直接打开，或是在上位机中生成 iDesigner 工程后打开。

4.4 快速入门

4.4.1 固件开发环境

打开工程文件首先确认芯片型号与配置字。

以 HR7P201FHD&S 为例。

图 4-1 芯片选择界面

配置字设定配置见下图：

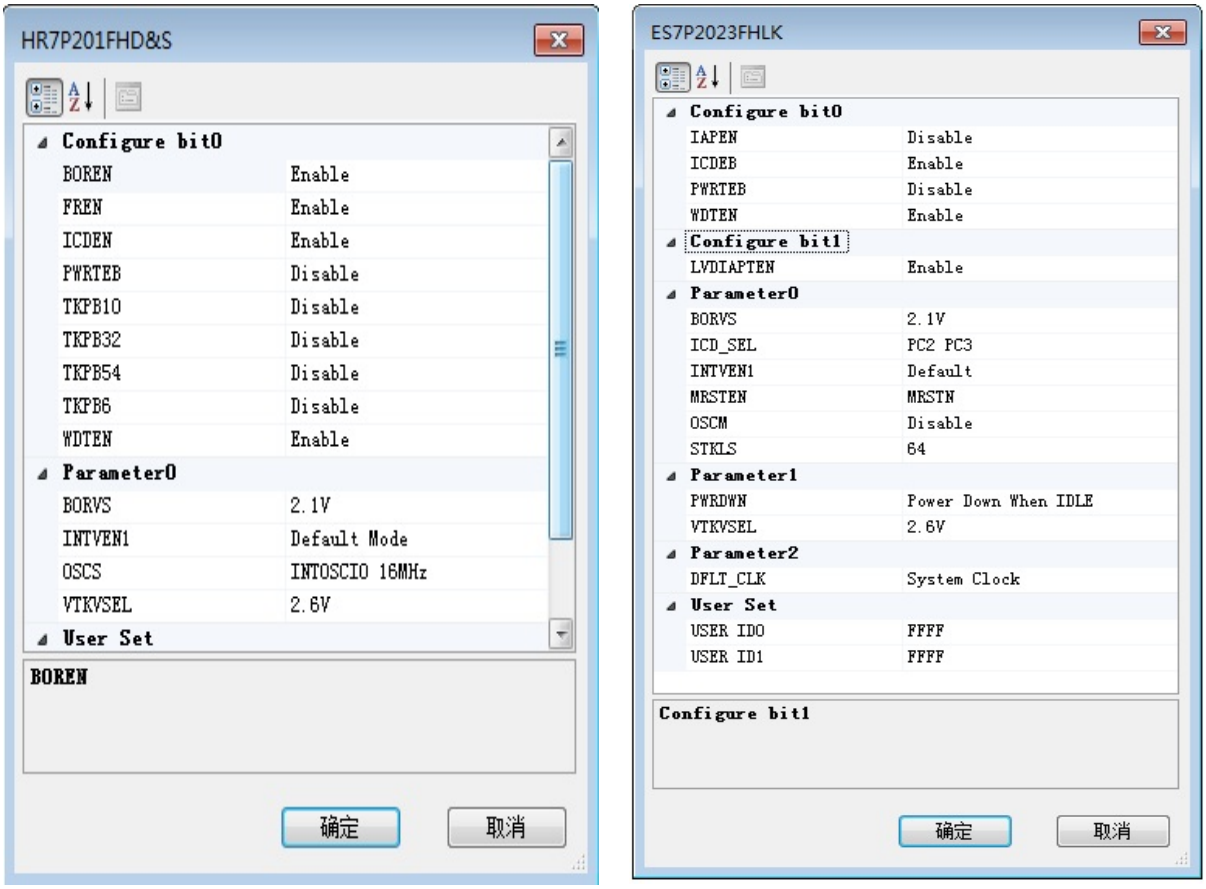


图 4-2 配置字界面

编译选项需要打开 stdlib 选项。

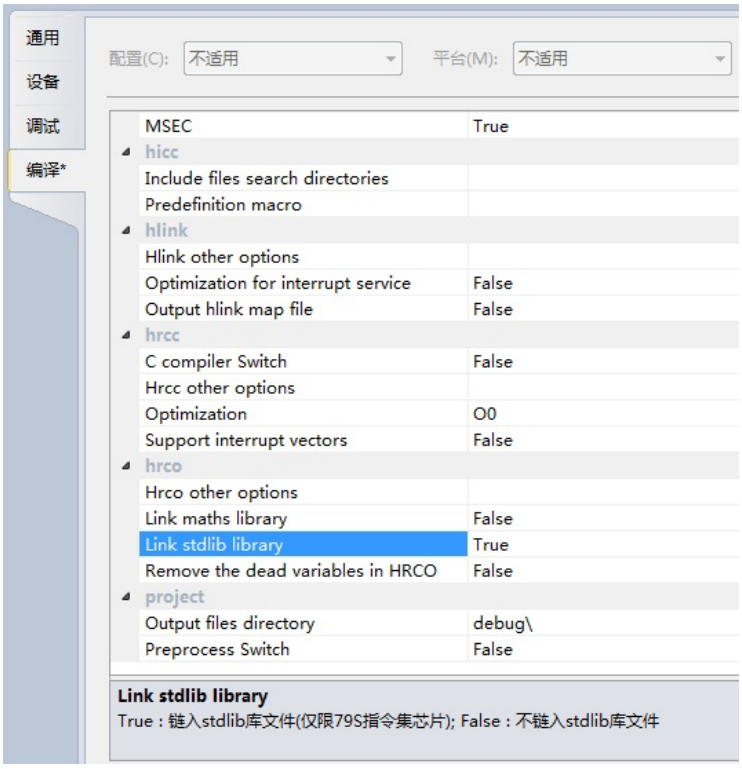


图 4-3 工程属性配置界面

本固件库目前支持：

1. HR7P201，最多 14 个按键通道；
2. ES7P202，最多 24 个按键通道。

打开工程后 iDesigner 页面概览如下：

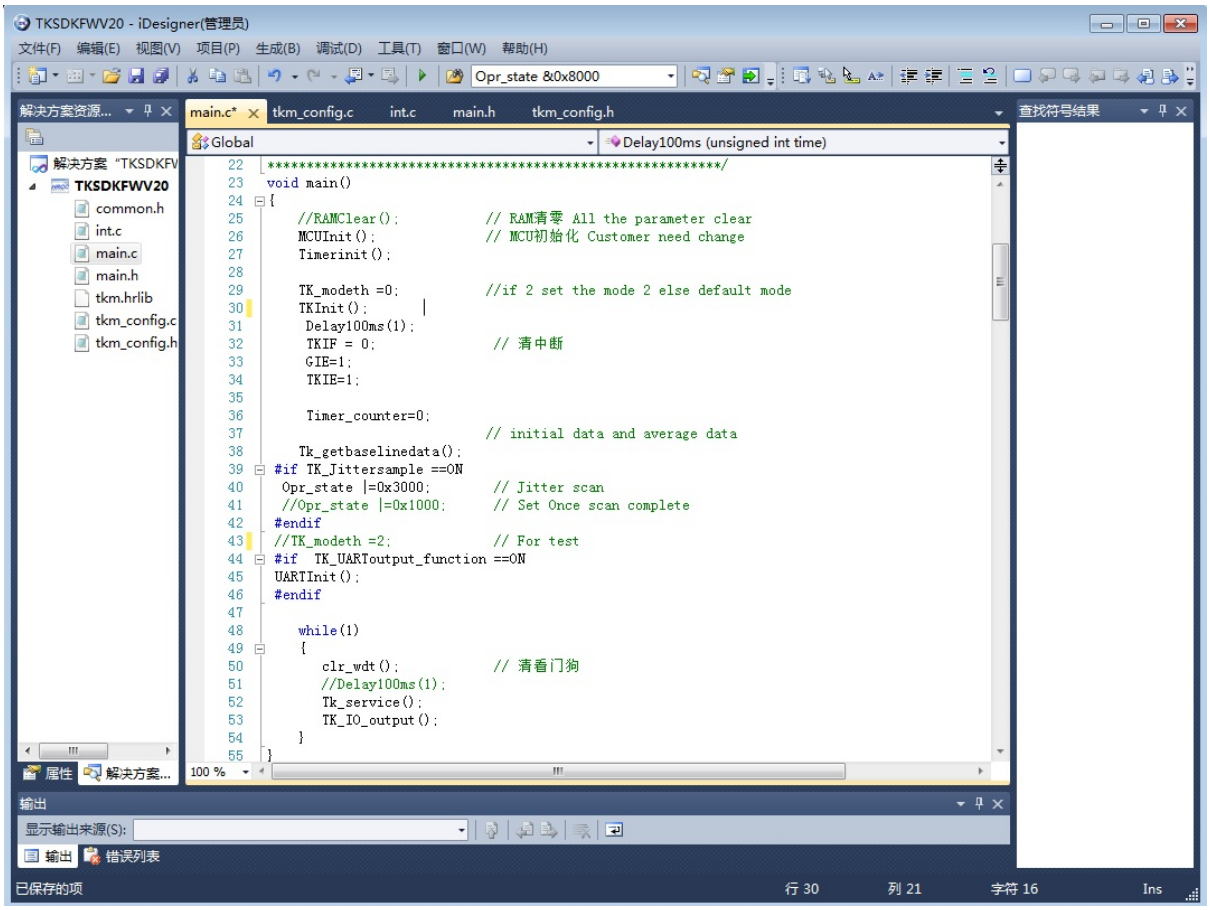


图 4-4 IDE 开发环境界面

4. 4. 2 工程文档结构

V2.1 固件工程包含如下文件：

1. common.h

工程支持文件，工程中用到的宏定义，及常用指令宏定义。

2. main.h

主要用来存放全局变量的声明文件。

3. tkm_config.h

主要用来存放 TK 各项参数的配置，与条件编译开关，与 tkm_config.c 中用到的函数与变量声明，此为 V2.0 核心文件，上位机参数更改的目标文件。

4. int.c

中断服务程序，存放 TK 中断，定时器 T8N (HR7P201) 或是 T10(ES7P202)8ms 中断，UART 串口中断处理函数。

5. tkm.hrlib

针对 TK 的各种数据处理算法函数库，包含按键，矩阵，滑轮，滑条及数据更新算法函数库。

6. tkm_config.c

TK 外围函数源文件，包含变量定义，TK 初始化函数，及可选数据输出函数，及 TK 服务函数，接口函数。

4.4.3 配置参数使用说明

为方便工程师使用触控库，本库所有常用的触控配置都集成在 tkm_config.h 文件中，请尽量在此文件中修改，其它文件请慎动。

下面以 6 个按键为例修改 tkm_config.h 文件。

◆ 按键通道设定

```
#define   TK_NUM      6
```

修改此定义数目为 6，包含所有用到的触控通道，包括滑条滑轮矩阵，然后根据目标板实际原理图在如下 6 个通道后对应上所连接的 TK 通道号。

```
#define   TK_Channel0   TK6
#define   TK_Channel1   TK3
#define   TK_Channel2   TK1
#define   TK_Channel3   TK8
#define   TK_Channel4   TK10
#define   TK_Channel5   TK12
```

定义各通道的门限，门限设定原则为最终产品手触摸变化量（差值）的一半。

```
#define   TK_Threshold_Channel0     200
#define   TK_Threshold_Channel1     200
。。。
#define   TK_Threshold_Channel5     200
```

通道总数配置完成后，可以大致对扫描周期进行估算，方法如下：

HR7P201 (OSC@16MHz)：T = TK_NUM * TK_Samples_perscan*155uS

ES7P202 (OSC@16MHz)：T = TK_NUM * TK_Samples_perscan*55uS

◆ 触控参数设定

```
#define   TK_Singlepress              OFF
//ON/OFF，  ON 同一刻仅允许有一个最强按键按下，实际按压不超过 3 个，OFF  允许所有按键同时按下
#define   TK_Amplifi_setting          0x000FFFFF
//TK  放大系数设定，1--0x0FFFFFFF，注意参数设定过大会导致采样数据溢出，需根据实际调整
#define   TK_Samples_perscan          5
// 采样次数累加设定，1-16，累加设定后 TK 采样结果不超过 16 位数据，否则请减小此数据
#define   TK_Threshold_release      7
//按键松开迟滞设定，4--9，对应为实际数值 0.4-0.9，若定为 5，则如果按下是 200，松开就是 100，
//其中 5 表示 0.5（50%）
#define   TK_Samplevalue_step       30
// 采样步进设定，1—65535，不超过门限三分之一为宜
#define   TK_basevalue_step           5
// 基线步进设定，1—65535，比采样步进要小
#define   TK_BaseSamples_perscan    64
```

```
// 基线更新采样次数设定，1—65535，基线在没有按键按下时自动修正以适应环境
#define   TK_Press_timeout          (120*125)
// 120s，超过此时间后强制取消此按下状态，并且平均基线建议值为 125--65535
#define   TK_Debounce_press         3
// 连续按下的触发次数设定
#define   TK_Debounce_release       3
// 连续离开的触发次数设定
```

◆ 离散采样功能开关与相关参数

```
#define   TK_Jittersample           OFF                 //  ON / OFF，使能开关
#define   TK_Jitter_maxdelay        5                   //  XX*32uS，1--30 this will cost scan time
```

若离散采样打开则采样延时最大值为定义值*32 微秒。

◆ 二次规划设定

```
#define   TK_MODE2          OFF                         //ON /OFF，使能开关
```

按键二次规划设定打开后，扫描按键时可由用户在两种模式下软件切换 使用变量 TK_modeth =1 或 TK_modeth =2 直接在两模式间切换，切换时间为一次 TK 通道扫描时间。

以下 MODE2 各通道参数设定，同上述触控参数设定

```
#define   TKMODE2_NUM       3                           //触控按键个数 1--TK_NUM，模式二；0-- Disable
//可以在低功耗的应用时启用，在休眠时启用模式二，仅扫描一个通道以节省电量
#define   TKMODE2_Channel0  TK6                         //TK0--TK13，可重复定义不同门限
#define   TKMODE2_Channel1  TK8
#define   TKMODE2_Channel2  TK12
#define   TKMODE2_Threshold_Channel0    160             //视具体情况而定，1--65535
#define   TKMODE2_Threshold_Channel1    160
#define   TKMODE2_Threshold_Channel2    160
#define   TK_Md2_Samples_perscan        3               // 采样次数累加设定 1--16，不超过 16 位数据
#define   TK_Md2_Threshold_release      9
//按键松开迟滞设定，4--9，对应为实际数值 0.4-0.9，若定为 5，则如果按下是 200，松开就是 100，
//其中 5 表示 0.5（50%）
#define   TK_Md2_Samplevalue_step       30              // 采样步进设定，1--65535，不超过门限三分之一
#define   TK_Md2_basevalue_step         30              // 基线步进设定，1—65535，比采样步进要小
#define   TK_Md2_BaseSamples_perscan 0                  // 基线更新采样次数设定，0--65535
#define   TK_Md2_Debounce_press         1               // 连续按下的触发次数设定
#define   TK_Md2_Debounce_release       1               // 连续离开的触发次数设定
```

◆ 矩阵按键设定

```
#define   TK_Matrix_Function    OFF                     // ON/ OFF，使能开关
#define   TK_Matrix_Row_NUM     2                       // 2--14，矩阵行数设置
#define   TK_Matrix_Row0        TK_Ch2
#define   TK_Matrix_Row1        TK_Ch4
。。。
#define   TK_Matrix_Row13       TK_Ch13
#define   TK_Matrix_Column_NUM  3                       // 2--14，矩阵列数设置
```

```
#define   TK_Matrix_Column0     TK_Ch3
#define   TK_Matrix_Column1     TK_Ch1
。。。
#define   TK_Matrix_Column13    TK_Ch13
```

输出结果计算方法： OutputValue = (Row th *Column NUM) + Column th +1 Only strong signal valued

◆ 滑条设定

```
#define   TK_Slider_Function     OFF          // ON / OFF，使能开关
#define   TK_Slider_Levels       4            // 2--14，Sensor num，由几个 Sensor 组成的 Slider
#define   TK_Slider_poles_setting    32       // Value <= TK_Slider_poles_setting* (TK_Slider_Levels-1)
```

此为单极分辨率设定

```
#define   TK_Slider_level0       TK_Ch1
#define   TK_Slider_level1       TK_Ch2
。。。
#define   TK_Slider_level13       TK_Ch13
```

◆ 滑轮设定

```
#define   TK_Wheel_Function     OFF              // ON / OFF，滑轮功能开关
#define   TK_Wheel_Levels       4                // 2--14，Sensor num
#define   TK_Wheel_poles_setting    32           //  结果<= TK_Wheel_Levels * TK_Slider_Levels

#define   TK_Wheel_level0       TK_Ch1
#define   TK_Wheel_level1       TK_Ch2
。。。
#define   TK_Wheel_level13       TK_Ch13
```

◆ IO 输出设定

```
#define   TK_IOoutput_function      ON          // ON/ OFF，IO 输出功能开关 最大支持 8 路输出
#define   TK_IOoutput_port0     PA6             // PA PB PC，Any bit
#define   TK_IOoutput_port1     PA3
#define   TK_IOoutput_port2     PA0
#define   TK_IOoutput_port3     PB2
#define   TK_IOoutput_port4     PB4
#define   TK_IOoutput_port5     PB6

#define   TK_IOoutput_port6     PB6
#define   TK_IOoutput_port7     PA6
```

定义输出管脚通道

```
#define   TK_IOoutput_trise0    PAT6
#define   TK_IOoutput_trise1    PAT3
。。。
#define   TK_IOoutput_trise7    PAT4
```

```
#define   TK_IOLED_Function   Indicate                  //Indicate/Toggle
Indicate，是按下亮松开灭，Toggle 是开关触发模式，按下亮再按下灭
#define   LED_On      0                                 //   0/1  定义 IO 口接 LED 时亮灭控制电平
#define   LED_Off     1                                 //   0/1
```

◆ UART 输出设定

```
#define   TK_UARToutput_function      ON                // ON/ OFF，功能开关

#define   TK_UARToutput_Baudrate      115200
//如果使用该波特率通信不成功，用户可尝试将波特率调整为 9600，再作测试。

#define   TK_UARToutput_Databit       8                 // 4,5,6,7,8.9
#define   TK_UARToutput_Stopbit       1                 // 1,2,3(1.5)
#define   TK_UARToutput_Verifybit     No                //No
```

◆ 寄存器设定

以下设定请详查芯片数据手册定义，慎改！

```
#define   TK_reg_VRC1           0x82
//For VRC1 register，使能 ADCVREF 2.6V，VREF1=1.4V
#define   TK_reg_ACPC4          0x11
//For ACPC4 register，使能比较器 C4，内部 VREF1
#define   TK_reg_TKTUN          0x30
#define   TK_reg_TKSEL          0x40
//设定感应按键通道，充放电占空比 1:2，扫描频率 fosc/4
```

◆ 保护环输出

```
#define   TK_Guardsensor_output     OFF               //保护环输出控制开关
#define   TK_Guardsensor_setting    PB0
//保护环扫描时提高被扫通道信噪比，在休眠模式可设计为唤醒通道。
#define   TK_GuardIOtrise_setting   PBT0              //方向寄存器
```

4.4.4 常用变量使用说明

本节提到的常用变量都是库函数中某一特定算法的输出值，这些变量存在于 TK_Service 中，在调试时，可以通过这些值来观察按键效果。

程序运行状态控制字：Opr_state

Bit15	扫描使能 Scan Enable Bit	1-On	0-Off
Bit14	扫描数据满 Tempdata is Full	1-Full	0-Receive
Bit13	去抖采样使能 Jillter Sample Enable Bit	1-Enable	0-Disable
Bit12	一次扫描完成标志 Once Scan Complete Bit	1-Complete	0-Busy
Bit11	滑条处理标志 Slider Bit	1-Touched	0- Not Touched

Bit10	滑轮处理标志 Wheel Bit	1-Touched	0- Not Touched
Bit9	矩阵行状态标志 Matrix Row Statue	1-Touched	0- Not Touched
Bit8	矩阵列状态标志 Matrix Column Status	1-Touched	0- Not Touched
Bit7	模式状态标志 Mode Status Bit	1-Mode 2	0-Mode 1
Bit6	模式切换标志 Mode Switch Start Switching	1-Mode need switching 0- switching done	
Bit5	低功耗模式使能标志 Power save Flag Setting	1-Enable Power save 0-Disable Power save function	
Bit4	保留		
Bit3	保留		
Bit2	保留		
Bit1	保留		
Bit0	保留		

按键通道数： Tkchnum;
系统使用 TK 功能通道总数。

当前扫描通道：Tkscan_numth;
TK 芯片采用的是轮询方式，该值表示当前正在被激活扫描的通道，与 TKSEL 寄存器相关。

当前扫描通道扫描次数计数：Tkscan_sampcounter;
与宏定义 TK_Samples_perscan 相关，计数超过该定义值时，进行下一步处理。

采样数据：TK_value_origin[TK_NUM] ={0};
采集 TK 通道上未经处理的数据。

滤波数据：TK_value_filter[TK_NUM] ={0};
对采样数据进行软件滤波算法后的数据。

基线数据：TK_value_average[TK_NUM] ={0};
在没有按键事件时，对采样数据进行处理后的值，作为是否产生按键事件的基准值。

差值数据：TK_D_value[TK_NUM]={0};
滤波数据与基线数据之差，当该差值超过门限时，判为有按键产生。

按键超时计数：TK_timeout_counter[TK_NUM]={0};
与 TK_Press_timeout 相关，所有通道按下连续时间不能超过该宏定义时长，如果超过，则自动更新基线。

备份基线数据：TK_average_backup[TK_NUM] ={0};
用于模式 1、2 切换时备份。

模式 2 基线数据：TK_md2average_backup[TK_NUM] ={0};
用于模式 1、2 切换时备份。

按键按下累积次数计数：TK_press_table[TK_NUM]　={0};
超过该计数判为有按下事件。

按键离开累积次数计数：TK_release_table[TK_NUM]={0};
超过该计数判为有按键离开事件。

矩阵行按键差值：TK_D_Rvalue[TK_Matrix_Row_NUM]={0};
矩阵列按键差值：TK_D_Cvalue[TK_Matrix_Column_NUM]={0};

矩阵行按键 ID：TK_Matrix_RID;
每一位对应一个按键通道。1 代表有键按下。

矩阵列按键 ID ：TK_Matrix_CID;
每一位对应一个按键通道。1 代表有键按下。

矩阵按键值：TK_Matrix_value;
矩阵键值清零计数器，无操作计到 12，大概 100ms 后清矩阵按键值：Matrix_timer;

滑条差值数组：TK_D_slider[TK_Slider_Levels]={0};
取自 TK_D_value，用于滑条算法的处理。

滑条当前按键 ID：TK_slider_ID;
每一位对应一个按键通道。

滑条结果值：TK_slider_value;
滑条键值清零计数器： Slider_timer;
无操作约 100ms 后清矩阵按键值。

滑轮差值数组：TK_D_wheel[TK_Wheel_Levels]={0};
取自 TK_D_value，用于滑轮算法的处理。

滑轮当前按键 ID：TK_wheel_ID;
每一位对应一个按键通道。1 代表有键按下。

滑轮结果值：TK_wheel_value;
滑轮键值清零计数器，无操作约 100ms 后清矩阵按键值：Wheel_timer;

按键模式选择： TK_modeth;
当前运行模式控制位值可为 1 或 2，使 TK 运行在 Mode1 或 Mode2 模式。

当前按键值：TK_state;
每一位对应一个按键通道。1 代表有键按下。

4.4.5 函数说明

函 数 名	Tk_service()
源文件名	tkm_config.c
函数说明	TK SDK 服务函数主函数
输入参数	tkm_config.h 中各参数
返 回 值	无影响全局变量按键状态值，或是滑轮滑条值结果输出
调用方法	初始化完成后可直接调用

函 数 名	TK ISR
源文件名	int.c
函数说明	TK 扫描中断服务函数
输入参数	TKIE TKIF
返 回 值	当前通道扫描结果与下一通道扫描开始
调用方法	中断

函 数 名	Singlekeyprocess()
源文件名	tkm.hrlib
函数说明	单按键模式判断处理函数，选择信号最强的按键输出
输入参数	全局变量中的TK_D_value[]，各按键的Dlt量
返 回 值	TK_state_single最强按键状态
调用方法	数据处理函数后直接调用

函 数 名	Matrix_process()
源文件名	tkm.hrlib
函数说明	矩阵服务处理函数
输入参数	全局变量中的TK_D_Rvalue[]，TK_D_Rvalue[]，各按键的Dlt量
返 回 值	TK_Matrix_value
调用方法	数据处理函数后直接调用

函 数 名	Slider_process()
源文件名	tkm.hrlib
函数说明	滑条服务处理函数
输入参数	全局变量中的TK_value_filter[]，TK_D_slider[]
返 回 值	TK_slider_value
调用方法	数据处理函数后直接调用

函 数 名	Wheel_process
源文件名	tkm.hrlib
函数说明	滑轮服务处理函数
输入参数	全局变量中的TK_value_filter[]，TK_D_wheel[]
返 回 值	TK_wheel_value
调用方法	数据处理函数后直接调用

函 数 名	Tk_getbaselinedata()
源文件名	tkm.hrlib
函数说明	读取各通道 TK 值，更新所有滤波值与基线值
输入参数	全局变量中的Channel_table[]
返 回 值	TK_value_origin[]，TK_value_filter[]，TK_value_average[]
调用方法	在进入主函数循环前，或是模式切换前

函 数 名	TKScan()
源文件名	tkm.hrlib
函数说明	根据当前扫描序号进行 TK 通道的配置与扫描
输入参数	全局变量中的Channel_table[]与Tkscan_numth序号
返 回 值	无
调用方法	TK 中断后，进入下一通道扫描时

在 main.c 中已添加部分常用的客户版空函数，并有详细的使用时间节点说明，客户只需要添加上合适的动作指令，就能在空函数中完成主流的触控动作应用，详情请直接查看生成的 main.c 代码。

4.4.6 编译运行

在上述参数设定下，在工程名上右键重新生成，编译通过，点开始调试按钮，开始调试，见如下界面。

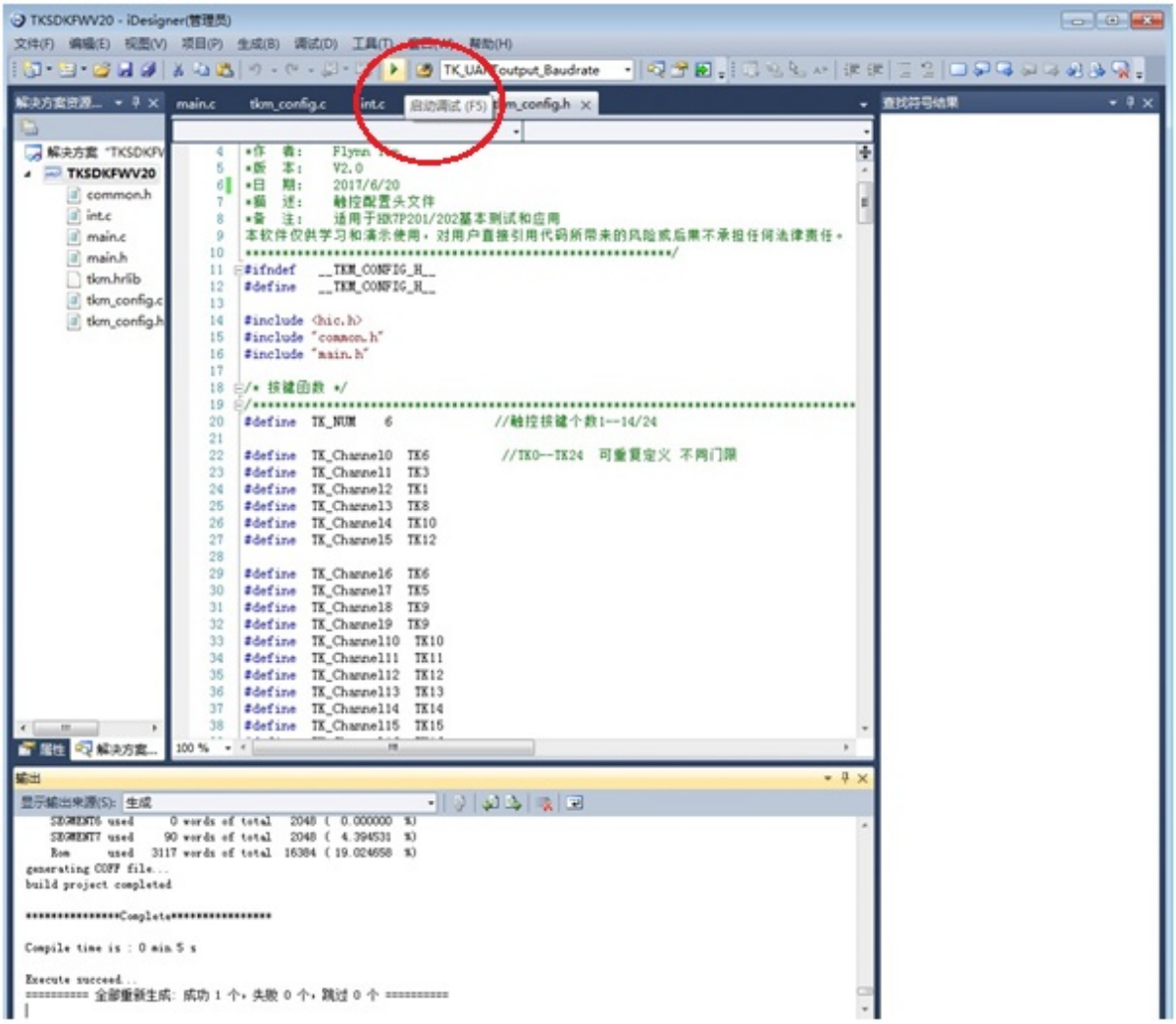


图 4-5 IDE 开发环境界面

4.4.7 Lite工程使用说明

为了保证库函数对芯片资源的最小占用率，在 2.1.0 版本之后，增加了 Lite 版本库函数及工程。

对于上位机软件，Lite版不提供配置界面选项，只提供调试及示例工程。

存储资源占用：

在 Max_Minvalueoff_filter及TK_UARToutput_function宏定义关闭的情况下，ROM占用2144 words，RAM占用170 bytes。

中断资源占用：

在 2.2.0 版本中，对中断资源作出调整，int.c 中只保留 125uS 中断，采用查询 TKIF 方式更新 TK 数据，并使用计数方式更新原 Timer 8MS 中断内容。其中 125uS 中断中数据处理最长时间占用可能占用为 70uS 左右，请注意尽量不要在此中断中加入过多的客户处理程序

在2.2.0版本中，TK时间处理函数在Main中调用Timer_check()；调用时间建议不超过10mS。

在Lite版本中提供的Max_Minvalueoff_filter为可选滤波算法，打开该算法有助于系统提高CS测试性能。

在2.2.0版本中，调试接口增加干扰强度输出指示功能，GUI界面借用原滑条输出界面，可以用此信号评估系统噪声强度。

在2.2.0版本中，增加干扰强度门限设定 ：

#define Jitter_level1_Threshold 800 //超过此设定，基线将锁定0.5S不更新，请根据具体干扰强度信号来设定门限

#define Jitter_level2_Threshold 8000 //超过此强度，本次的按键触发无效，TK_state不会被更新

在2.3.0版本中，增加临键抑制功能：

#define TK_Mult_inhibition 2

// 0---5 临键抑制系数 0 无抑制 1： 50% 2：25% 3：12.5% 4：6.25% 用于解决手指按压临键误触问题，抑制系数 1最强 4最弱

在2.3.0版本中针对Hr7p201芯片硬件堆栈只有8级问题已做优化：

向量中断中不开中断优先级占用2级堆栈，主函数中占用2级堆栈，考虑到数学运算最大可能本库将占用5级堆栈，在添加客户程序时尽量做到平行调用函数若是在库函数中调用函数请尽量不要调用超过三级。

若使用向量中断模式 并配置中断优先级后由于中断嵌套 中断将占用4级堆栈，函数运行将可能占用7级堆栈，请尽量不要在库函数中再添加子函数，尽量在主函数平行处添加客户函数。

若客户程序较复杂需要多层函数嵌套可以 配置 #define TK_Getdata_ISR OFF 来关闭库函数的向量中断模式而采用主循环查询方式 采用主循环模式 这样主程序中只占用3级堆栈。

采用主循环模式 TK数据查询时间 请配置 #define TK_counter_checkdata 12

// 1---65535 主循环计数时基 建议调整计数使周期为 100--500uS 注意主循环周期间隔 假如主循环一次需要10us 定义为12 即为每12次120uS时间查询下TK有无数据更新。数字随主循环时间来调整。

4.4.8 低功耗工程使用说明

如果系统对功耗敏感，可以在上位机界面中选择低功耗版本工程（图 4-6），在此基础上进行开发。目前低功耗版本模版只支持在 ES7P202 上的开发，用户也可以将该应用移植到 HR7P201 上。

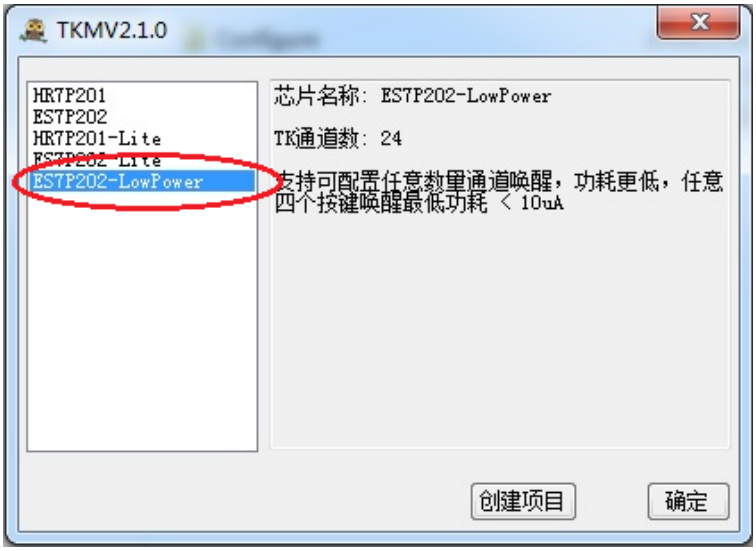


图 4-6 TKM 低功耗工程选择开机界面

该工程通过在休眠模式下，定时唤醒扫描按键的方法，降低系统整体功耗，功耗的计算公式如下：

$$I_{avg} = \frac{I_{SCAN} \times T_{SCAN} + I_{IDLE} \times T_{IDLE}}{T_{SCAN} + T_{IDLE}}$$

I_{SCAN} ： 低功耗模式下扫描按键功耗，一般按照正常工作 3mA 计算。

T_{SCAN} ： 低功耗模式下扫描按键时间。

I_{IDLE}：低功耗模式下休眠功耗，在关闭所有模块时钟，正确配置管脚电平情况下，可以达到 5uA。
T_{IDLE}：低功耗模式下休眠时间。
评估功耗时，为了方便计算，一般预设$T_{SCAN} + T_{IDLE} = 1s$。

低功耗版本不提供配置界面选项，只提供调试及示例工程。

低功耗版本提供了更低的功耗算法及更灵活的唤醒按键配置方法。

程序中通过 g_wakeup_flag 切换工作及低功耗状态，低功耗时的处理全部在 TKStandby 中完成。主要实现唤醒后的低功耗按键扫描算法。

TKStandby中通过调用SleepMode完成睡眠前底层的低功耗配置。

需要注意的是，为了实现更低的功耗，ES7P202在执行sleep指令前，需要把Cx管脚配成输出高。

```
clr_wdt();              // 清看门狗
if (g_wakeup_flag)
{
    Tk_service();
    TK_IO_output();
}
else
{
    TKStandby();    //sleep run
}
```

空间资源占用：

ROM占用4362 words，RAM占用284 bytes。

4.4.9 可调参数工程使用说明

本工程针对初次使用我司产品及需要实现主控配置 TK 从机参数的用户。目前该版本仅支持在 7P202 上的使用，用户也可以将该应用移植到 HR7P201 上。

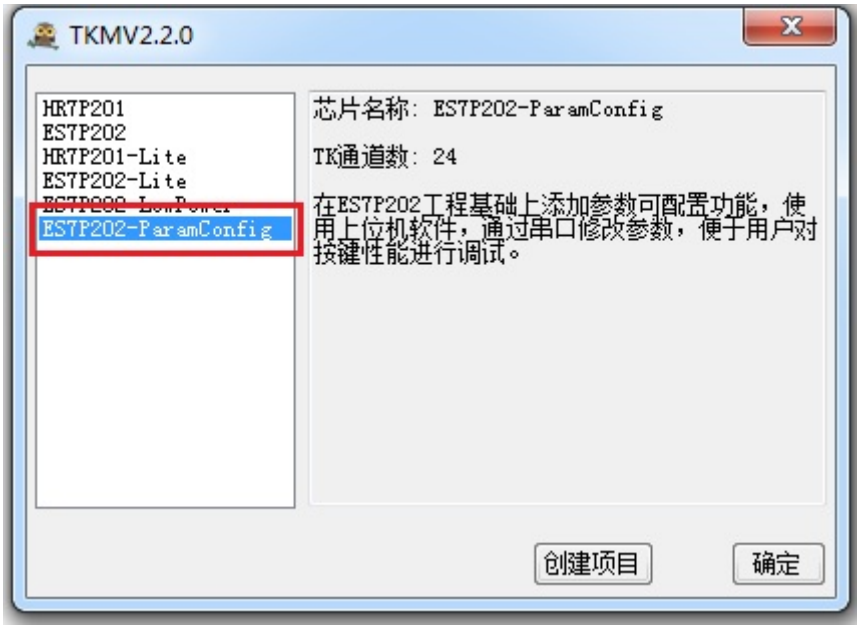


图 4-7 TKM 参数可调工程选择开机界面

创建的示例工程基于 ESD-TKT-7P202-Button V2.0 开发板，用户可以通过适当修改通道参数，在 7P202 的其他开发板进行调试。

点击“创建项目”，可以创建一个已经可以直接运行的 ES7P202 参数可调工程。

点击“确定”，打开调试界面，点击“配置界面”按键。

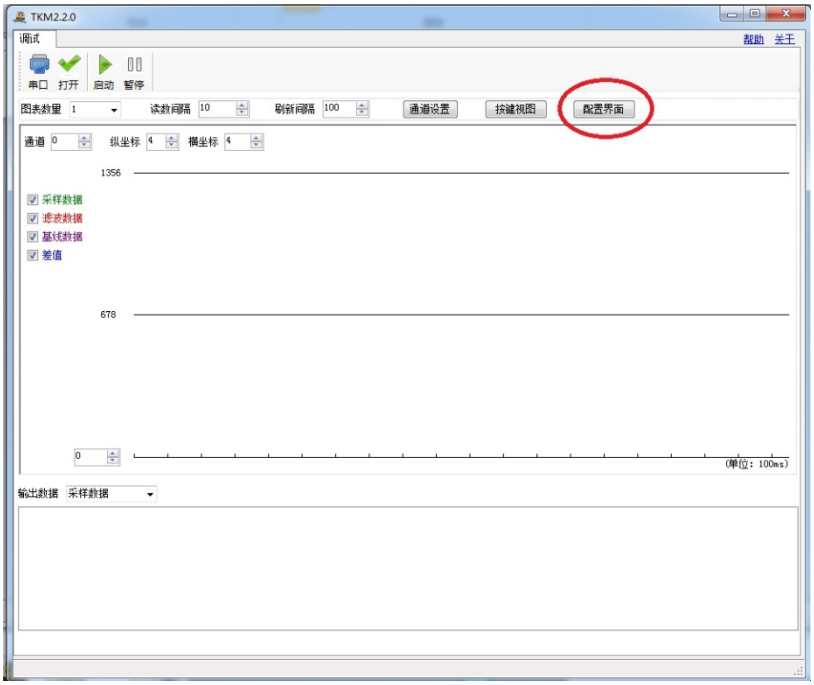


图 4-7 配置界面按钮示意图

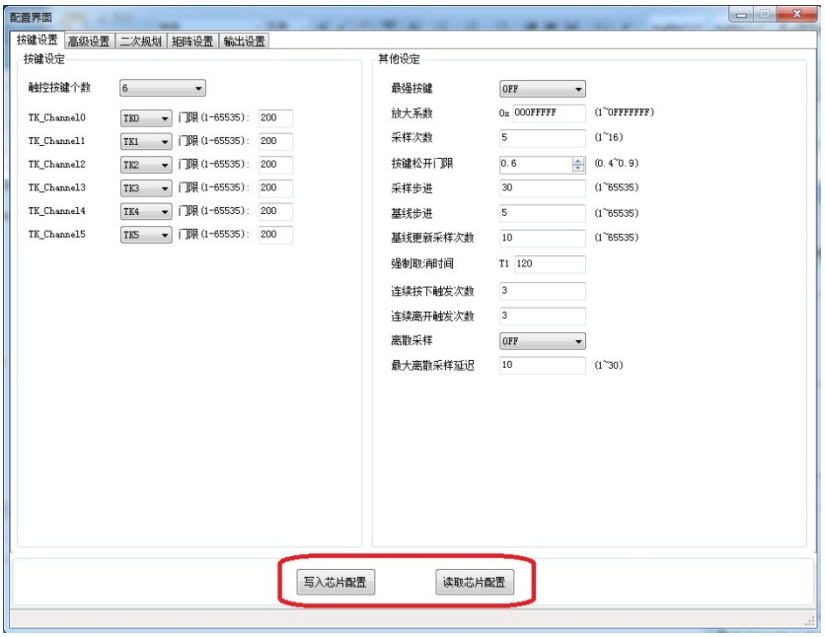


图 4-8 配置界面

图 4-8 为参数可调的上位机主界面，在保证与触控板通信正常的情况下，可以读取、写入芯片配置参数，这样的操作比反复烧录程序更方便，利于用户进行参数的调整评估。

目前开通参数配置主要针对用户调试，所以对于二次规划、矩阵及输出等设置界面的参数基本屏蔽。

目前该版本基于完整版的工程进行开发，因此暂时不包含 Lite 和低功耗等特殊版本中的功能。