



ChipON 水位监测

开发指南

(第二版)

上海芯旺微电子有限公司

2016.01

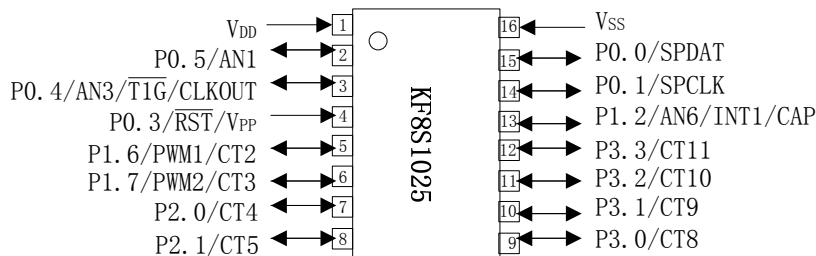
目录

1	触摸芯片选型	3
2	条件说明	3
3	产品开发设计约束	4
4	软件库使用说明	5
4.1	使用触摸步骤.....	5
4.2	创建触摸库头文件说明.....	6
4.3	定义触摸库变量.....	8
4.4	配置芯片寄存器说明.....	9
4.5	配置 CMCTL1 说明.....	10
4.6	调用触摸初始化内部参数函数.....	11
4.7	调用电容触摸通道处理函数.....	11
4.8	在线校准的支持等.....	12
4.9	专用水位监测程序实现.....	12
5	触摸库参数及设置说明	12
5.1	MX_CH/CHS_AMOUNT	13
5.2	TCS_AMOUNT:	13
5.3	_KF8_DISTURB_PROTECT_CIRCLE_ _KF8_DISTURB_PROTECT_CIRCLE_DEFINE .	13
5.4	_KF8_UP_BASELINE_CIRCLE_ _KF8_UP_BASELINE_CIRCLE_DEFINE	13
5.5	_KF8_TOUCH_CH_EN[MX_CH]	14
5.6	_KF8_INSIDE_REFERENCE_CHANNEL_ORDER (参考通道声明)	14
5.7	_KF8_INSIDE_REFERENCE_CHANNEL_DISTURB_THRESHOLD_SET_ (参考通道异常波动阈值设定)	14
5.8	_KF8_CONFIG_FINGER_THRESHOLD [MX_CH] 有水识别阈值.....	14
5.9	CONFIG_NO_FINGER_THRESHOLD [MX_CH] 无水识别阈值.....	15
6	附录一 芯片引脚、_KF8_LIBI_CHANNEL_FLAG_和 _KF8_TOUCH_CH_EN 对应表.....	15
7	附录二 通道变化率的采集和计算	16

1 触摸芯片选型

目前 ChipON 推荐的水位监测芯片使用 KF8S1025。该款触摸芯片具有 4kFlase 存贮，256 的 BEE 存贮，具有以下优点：

- ✚ 电容触摸通道多，最多可达 8 个，其中 1 个作用参考通道；
- ✚ 灵敏度高；
- ✚ 抗干扰能力强，具有防电磁等功能；
- ✚ 电容触摸通道可调范围大，用户可根据需要更改外挂电容、分频比和参考电压；
- ✚ 资源丰富，还可实现控制，比如开关信号，温度采样，PWM 控制实现等。

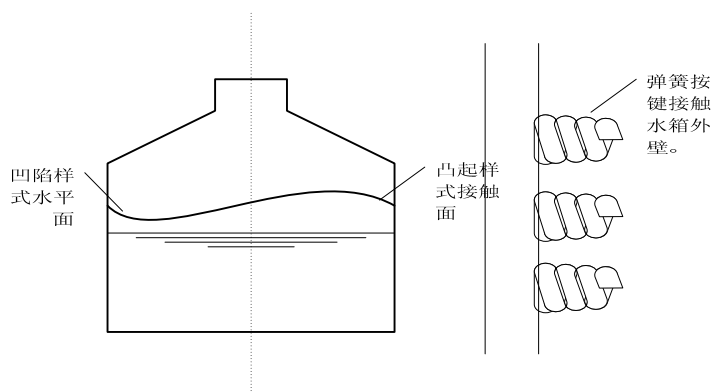


2 条件说明

ChipON 触摸芯片需添加外挂电容，使用 P12 脚，外挂电容的大小影响对采样值有一定影响，可选择 2.2nf 到 44nf 的电容，一般可使用标称 472/103 的电容，具体参数根据产品的实际情况进行更改，可以借助 TS TOOL 软件采集采样值和变化率数据，建议系统采样值调整在 1000-8000 的范围内。细节触摸设计请参考另一文档《Chipon 电容触摸设计指南》。

水位监测不同于普通按键的识别，普通按键一般存在最初按键按下时间，但不同产品的水位监测，可能水到来后会保持很长一段时间，处于这个时刻的通道自身缺少更随环境能力。因此通用的触摸按键识别算法设计并不是适合水位监测，即使水快速消耗，也不行，因为无水的停留时间不确定，一点水和完全无水的区分界限不明确。另外还存在带水开机的情况，普通的算法更无法适用。我公司的算法实现可完全解决这种问题。

3 产品开发设计约束



- 1、触摸通道的走线应该设计的尽可能细，如 0.254mm。
- 2、触摸走线最好同其他线隔离，尤其信号线，应避免平行走势。
- 3、触摸芯片的工作电源应避免频繁波动，瞬态波动算法实现保护。
- 4、为了更好的实现抗干扰能力，芯片电源供应端应设低通滤波器，至少 1 级。
- 5、参考通道应避免通过板子和水形成分布电容。一般措施为弹簧接触面隔离、芯片焊接的对面参考通道附近实现敷地，进而隔离水带入分布电容。也可以将检测板设计到水位以上或以下不接触的结构位置。
- 6、针对水箱存在一定高度的情况下，可以采用端口输出接口，触盘单独版通过引线接入触摸芯片。
- 7、要求触摸通道上的 1k 电阻靠近芯片引脚。
- 8、产品的通道参数只有整定后才最适合自身产品。
- 9、下载完程序的首次上电是参考数据的存入阶段，因此要求下载完程序后，检测板要同产品一起按照设计完成组装，在没有水的情况下上电，上电环境要求无干扰。
- 10、基于第 9 点可以采用芯片空闲引脚（编程脚除外，方便程序维护，参数整定）。接外部按键或做协议支持在线存入校准参数。
- 11、检测板芯片端滤波电容应尽可能靠近单片机。
- 12、板子预留或焊接下载端口或下载端点，便于调试维护和在线升级功能。
- 13、P12 脚的外接电容建议采用容值偏差小，温度波动范围小的电容。电容

值容值范围可选择 $1\text{nF} \sim 10\text{nF}$ 之间，要求使用 10%或以上精度的涤纶电容、X7R 材质电容或 NPO 材质贴片电容。

14、整定阈值参数使还要还要考虑如上图所示水膜在水箱壁上附着的情况，一般应该将水箱壁打湿，水膜符合左侧的凹陷平面。

15、如果使用弹簧，建议采用接触面大的弹簧按键，如果使用 PCB 上焊盘直接紧贴水箱壁要求最好用粘着性材料使可靠接触，检测通道接触间不能能有空气间隙，否则直接验证影响水带来的变化率，造成无法识别，除非变化率还很大。即仅允许很小的空气间隙，产品一致性考虑建议避免空隙。针对结构限制的只能增加触盘的接触面积，来提供变化率，对存在间隙的，应该考虑间隙下的参数整定。一般触盘大小因水位测试一般设计较窄，建议高度不小于 3mm ，长度根据产品实际尽量加宽。建议接触面积不小于 90mm^2 ，实际衡量看水能带来的变量率，越大越好，建议不低于 40。

16、要检测的液体应该单一，主要原因在于电容触摸考虑的是电介常数，相对电容量。液体不确定下容值变化波动不一定适合整定过的参数，无法保证产品的可靠运行。

17、产品出厂应该完成常规和老化实验，针对老化过程中出现的偏差需要重新存入校准值并重新老化观察。

4 软件库使用说明

4.1 使用触摸步骤

开发触摸按键系统需要通过以下 6 个步骤完成软件平台的搭建（或者直接使用 ChipON 提供的范例程序进行修改）：

① 创建触摸库头文件

需要在 `main.h` 中声明相应芯片的头文件和触摸库的头文件，触摸库头文件如下：

头文件 `kf8spsw_lib_touch.h`

② 定义触摸库变量

在项目中创建触摸库参数源文件，文件中的参数值需要用户根据产品的特性进行修改，

具体见 3.3 节。文件如下：

源文件 `kf8spsw_lib_touch.c`

③ 配置芯片寄存器

在 `main.c` 中需要对 MCU 进行初始化，初始化 IO 口和芯片功能。

④ 配置 CMCTL1 寄存器

可在 `main.c` 的初始化部分进行配置，根据芯片手册此寄存器用于设置电容触摸时钟分频比和电容触摸基准电压，时钟分频比越大，通道采样值越大，基准电压越大，通道采样值越大。根据不同产品，用户需对此做不同配置，具体参数含义可阅读相应芯片手册，具体见 3.5 节，一般参数设定为 0x50 即可。

⑤ 调用触摸模块初始化函数

调用触摸初始化函数对触摸参数进行初始化，用户只需在主函数中直接调用下面的函数即可，无需修改其内容参数。

水位库触摸初始化函数：`_KF8spsw_LIBf_init_touch_()`

除此之外，你可以选择函数 `_KF8spsw_lib_SET_MCU_Parameter(A)` 完成计时器的选择，默认不需要调用，但如果产品需要实现 PWM 控制时必须切换计时器。

⑥ 调用触摸库函数

在 `mian.c` 中调用触摸库处理函数。此函数用于实现触摸按键识别功能。

水位库函数名称为 `_KF8spsw_LIBf_touch_process_()`

以及触摸中断进行函数调用完成数据处理和按键识别即可。针对该类应用一般用软件查询即可。但该芯片的中断入口根据版本存在 2 种情况，即中断标志 T1IF (E00) 或 CTIF，需要更加芯片版本确定触摸中断的代码处理，demo 中已提供对应的实现代码。

4.2 创建触摸库头文件说明

水位通用库 `kf8spsw_lib_touch.h` 文件如下：

```
/* *****  
* 文件名: kf8spsw_lib_touch.h  
* 版本: V2.0  
* 日期: 2015-6-28  
* 作者: 上海芯旺微电子有限公司  
* 说明: 电容触摸库函数头文件  
***** */
```

```

/*****/

#ifndef KF8_LIB_TOUCH_H_
#define KF8_LIB_TOUCH_H_
#include "main.h"

#define MX_CH 4 //开启的触摸通道数量
/*****/

//声明给触摸库使用
/*****/

extern unsigned char const CHS_AMOUNT; //最大通道数量
extern unsigned int const TCS_AMOUNT; //识别延时滤波计数
//量化参考通道变化率多于x属于异常波动
extern signed int const
_KF8_Inside_Reference_Channel_Disturb_Threshold_Set_; //仅用于参考通道
extern unsigned int _KF8_Disturb_Protect_Circle_; //抖动结束
延迟计数
extern unsigned int const _KF8_Disturb_Protect_Circle_Define; //抖动
保护延时上限设定
extern unsigned int _KF8_Up_BaseLine_Circle_; //参考稳定补偿基
准线计数
extern unsigned int const _KF8_Up_BaseLine_Circle_Define; //基准线非
异常抖动n循环后更新基准线
extern volatile unsigned int _KF8_LIBi_channel_flag_; //通道识别结果
extern unsigned char const _KF8_Inside_Reference_Channel_Order; //基
准通道对应的数组下标
extern signed int const _KF8_Config_Finger_Threshold[MX_CH]; //水位阀
extern signed int const CONFIG_NO_FINGER_THRESHOLD[MX_CH]; //无水阀
extern unsigned char const _KF8_TOUCH_CH_EN[MX_CH]; //逻辑化顺序通道
extern signed int _KF8_LIBi_Date_Change_[MX_CH]; //通道变化量
extern volatile unsigned char _KF8_LIBc_channel_; //当前处理通道数组下标
extern unsigned int _KF8_LIBi_buff_hit_[MX_CH]; //保存采样值
extern unsigned int _KF8_LIBi_buff_baseline_[MX_CH]; //保存基准线
extern unsigned int _KF8_LIBi_buff_refline_[MX_CH]; //出厂参考值
extern unsigned int _KF8_LIBc_touch_count_[MX_CH]; //按键滤波计数
extern volatile unsigned char Arr_erom_SW[24]; //BEE 缓存, 在线校准用
/*****/

//声明给触摸库使用结束
/*****/

void _KF8spsw_LIBf_init_touch(); //初始化电容触摸
void _KF8spsw_LIBf_touch_process(); //电容触摸通道处理
// 参数更换 0: 选用 T1H T1L PWM不可用; 1: 选用T1U, T3L但仅适合E03版本芯片;
2: 选用T1U, T3L, 满足E05及以后版本芯片。

```



```
void _KF8spsw_Lib_SET_MCU_Parameter(unsigned char parameterin);  
void read_data_eep_tsw();// 读通道出厂参考值到 Arr_erom_SW_  
void write_eep_tsw();    // 将Arr_erom_SW_作为出厂参考值写入，在线重校。  
#endif /* KF8_LIB_TOUCH_H_ */
```

4.3 定义触摸库变量

定义如下变量，变量含义及参数设定原则详见第五节说明。

水位库 `kf8spsw_lib_touch.c`，文件的参数需与头文件一致，如下所示：

```
/**/*****  
// * 文件名: kf8spsw_lib_touch.c  
// * 版 本:   v2.0  
// * 日 期:   2015-6-28  
// * 作 者:   上海芯旺微电子有限公司  
// * 说明:     电容触摸库函数文件  
*****/  
#include "kf8spsw_lib_touch.h"    //引入触摸库函数头文件  
/*****/  
//触摸库使用  
/*****/  
unsigned char const CHS_AMOUNT = MX_CH; //传递系统最大通道数量，固定  
//满足阈值的次数滤波识别一个状态的转换，速度与模型相关和需求相关。  
unsigned int const TCS_AMOUNT = 200;  
  
//量化参考通道变化率多于x属于异常波动  
signed int const  
    _KF8_Inside_Reference_Channel_Disturb_Threshold_Set_=15; //参考通  
道抖动识  阈  
  
unsigned int      _KF8_Disturb_Protect_Circle_;           // 抖动保护计时  
unsigned int const _KF8_Disturb_Protect_Circle_Define=20; // 抖动保护  
时长设定  
  
unsigned int      _KF8_Up_BaseLine_Circle_;// 参考更新计时  
unsigned int const _KF8_Up_BaseLine_Circle_Define=200; // 参考更新时长  
//使用的通道数据及通道号，这里需和芯片对应，如CT3写3  
unsigned char const _KF8_TOUCH_CH_EN[MX_CH]={  
    10,  
    9,  
    8,  
    11,
```



```
2,
4,
5, // 需满足设定的通道个数，多余的会优化掉
};

// 算法中比引入参考通道数组位置
unsigned char const _KF8_Inside_Reference_Channel_Order=3; //即CT11参考
// 有水阈值 阈值设定 有水大于无水阈值，中间为识别死区
signed int const _KF8_Config_Finger_Threshold[MX_CH]={
    38,
    30,
    20,
    1000, // 参考通道不参与判断
    // . . . 须满足设定的通道个数
};

// 无水阈值
signed int const CONFIG_NO_FINGER_THRESHOLD[MX_CH]={
    20,
    20,
    17,
    800, // 参考通道不参与判断
    // . . . 须满足设定的通道个数
};

volatile unsigned char _KF8_LIBc_channel_; //当前处理通道位置
volatile unsigned int _KF8_LIBi_channel_flag_; //对外提供按键信息
/*以下为触摸算法所用到的通用变量数据，数组的元素个数必须与所开通的通道数一致****/
unsigned int _KF8_LIBi_buff_hit_[MX_CH]; //记录当前通道的采样值
unsigned int _KF8_LIBi_buff_baseline_[MX_CH]; //当前通道的基准值
unsigned int _KF8_LIBi_buff_refline_[MX_CH]; //当前设备的出厂参考值
unsigned int _KF8_LIBc_touch_count_[MX_CH]; //按键识别次数滤波
signed int _KF8_LIBi_Date_Change_[MX_CH]; //通道变化量
/*****/
//触摸库使用资源结束
/*****/
```

4.4 配置芯片寄存器说明

1、参考晶振初始化，如 OSCCTL = 0x60; 根据芯片手册可知为 8M 晶振，晶振频率不能太低，否则会影响触摸反应速度。

2、端口初始化

© 将电容触摸通道对应的 I/O 口设置为输入，即给 TRx 方向寄存器的对应位置 1。

- ◎ 芯片内部参考通道必须设置为输入态，这里使用外部通道作为参考，必须设置为输入态。
- ◎ 外接电容端口需设置成模拟口，P1.2 口为外接电容引脚，需 $ANSEL |= 1 << 6$;
- ◎ 触摸用到的寄存器以及 bit 位有 T1H(T1U)(定时计数器高八位)、T1L(T3L) (定时计数器低八位,)、T1ON(T1 启动控制位)、T1IE (CTIE)(触摸中断使能位)、T1IF (CTIF) (触摸中断标志位)、PUIE(在初始化触摸内部参数函数会打开，不需要额外设置)、AIE(全局中断使能位，调用初始化触摸内部参数函数时也会打开总中断，无需额外开启)。

芯片程序范例如下：

```
void init_mcu()
{
    OSCCTL = 0x70;           //MAX16M   0x70   0x60   0x50   ...
    //端口初始化
    TR0 = 0x08; // P03 只能输入
    TR1 = 0x04; // P12 外部电容
    TR2 = 0x00; //
    TR3 = 0x0F; // P30 P31 P32 P33 前3个按键通道以及参考通道

    P0 = 0x00; P1 = 0x00;
    P2 = 0x00; P3 = 0x00;

    ANSEL = 0x40;           //设置AN6为模拟口, 外挂电容脚设置为模拟口
    (ANSEL |= 1 << 6;       //设置AN6为模拟口, 外挂电容脚设置为模拟口)

    T1CTL = 0x00;           //1: x分频 如果使用T1H, T1L时可以设置计数分频
}
```

4.5 配置 CMCTL1 说明

寄存器“CMCTL1”是 KF8S 系列设置触摸分频比和基准电压的寄存器，用户通过设置 CMCTL1 的值调节对外挂电容充电的频率和电压阈值。分频比越大，则频率越低，通道采样值则越大。电压阈值越大，通道采样值也越大。

KF8S 系列芯片的 CMCTL1 寄存器引用数据手册说明如下：

寄存器8.1 CMCTL1: 控制寄存器(地址: 1AH)

复位值 0000	bit7				bit0			
	CTCLKSEL1	CTCLKSEL0	CTVREFSEL1	CTVREFSEL0	-	-	-	-
	R/W	R/W	R/W	R/W	U	U	U	U

CTCLKSEL<1:0> 电容触摸时钟预分频比选择位

00 = Fosc/4

01 = Fosc/8

10 = Fosc/16

11 = Fosc/32

CTVREFSEL<1:0> 电容触摸基准电压选择位

01 = 0.5VDD

10 = 0.7VDD

11 = 0.9VDD

如 CMCTL1 = 0x50; 则 设置触摸时钟为 2 分频, 基准电压为 0.5VDD, 这里分频源为系统时钟, 如选择 16M 时, 触摸时钟为 8M, 一般频率需设定为 1M 或 2M 时最佳。

4.6 调用触摸初始化内部参数函数

触摸初始化函数主要是对触摸算法的内部参数以及部分寄存器进行初始化, 如配置触摸使能、定时器工作设定等。因开机时采样并计算参数数据。在进行初始化前建议作适当延时, 避免上电波动使基准线数据不准确而带来异常, 参考 200ms。示例如下:

```
init_mcu();
delay_ms(200);    //随后开始功能的运行
```

4.7 调用电容触摸通道处理函数

电容触摸按键处理函数能够对触摸按键进行手指按下与否的判断, 可以在中断内、外中执行, 返回值为 `_KF8_LIBi_channel_flag_`。它的每个 bit 位对应相应的通道, 对应关系见附录 1。当有通道被触摸时, `_KF8_LIBi_channel_flag_` 的对应 bit 位被置 1, 无触摸时对应 bit 位置 0。电容触摸中断标志位 S 系列芯片为 T1IF, 或 CTIF, 具体和版本有关。每次进中断时判断, 如果 T1IF/CTIF 为 1 时调用电容触摸通道处理函数, 还可以在不影响中断内部处理时间的条件下, 设置一个标志位, 然后再在主程序中调用电容触摸通道处理函数。

```
void INT_FUN() __interrupt (0)
{
    if(T1IF||CTIF)
```

```
{
    T1IF=0; CTIF=0; // E00 板芯片使用T1IF, 其他版本使用CTIF
    _KF8spsw_LIBf_touch_process_ ();      // 在中断中执行触摸函数
}
/*如果使用PWM, E00版本不可以, 触摸中断标志为CTIF, T1IF为PWM使用*/
}
也可以在中断中做一个标志后退出, 然后在 main.c 中如下处理
if(touch_process_flag)                // 判断标志位
{
    touch_process_flag = 0;           // 清标志位
    _KF8spsw_LIBf_touch_process_ (); // 在主函数中执行触摸函数
}
```

4.8 在线校准的支持等

在线校准的方法可通过端口高低电平或协议指令下进行, 进行方法为在稳定的环境中, 无水的情况下, 按照逻辑顺序将开启的通道的采样值_KF8_LIBi_buff_hit_送入到数组_KF8_LIBi_buff_baseline_和_KF8_LIBi_buff_refline_以及 Arr_erom_SW_中, 然后调用写函数 write_eep_tsw_() 执行写入。操作函数自动校验功能, 校验失败会重试, 直到成功为止。一般不需要使用读函数 read_data_eep_tsw_(), 但程序加密下, 代码增加命令和协议可以实现将参数读取并送出实现产品分析。

如果时间要求不是特别, 可以触摸处理函数前保留 TOUCH_DEBUG_TRS_DEAL ()。该函数在 debug_touch.c 中实现, 可随时在线观察产品的运行情况, 采用该函数会拉长扫描一周的时间, 但可以通过所需参数实现同一现实时间内的识别工作, 一般来说不开启主机情况下消耗时间较少。

4.9 专用水位监测程序实现

可以采用该型号芯片的固化程序进行产品的开发, 仅需要调试参数写到对应的 hex 文件中进行批量编程即可, 支持 BCD 和 I2C 方式输出, 能够加快产品开发周期, 减少开发工作量。也可以采用在线模式实现通信完成数据的交互, 包括在线校准, 手动设定参数, 功能配置等。

5 触摸库参数及设置说明

通过上一章节的前七个步骤, 触摸按键产品的整体代码完成。但在当前产品上能否运行

正确还需要对相应的参数进行调试。

名词解释：

1、 扫描周期时间：即程序对所有按键扫描并处理一遍所使用的时间。文档的关于时间的概念设定值均为周期数。如设置为100，可以想象如果每个按键占用1ms时间，7个按键就是7ms，100个周期则代表着700ms，具体每个通道通道的处理时间与电压选择、电容大小相关。

2、 通道变化率：通道变化率是有符号数，正数表示采样值在基准线下方，负数表示采样值在基准线上方。关于此参数的说明和计算方式详见参考附录二。

5.1 MX_CH/ CHS_AMOUNT

MX_CH参数设定产品触摸通道数，在程序上采用宏定义的方法，可在kf8spsw_lib_touch.h中修改。CHS_AMOUNT参数同样为使用的通道数据，该数值应用于通道循环扫描判断，赋值为MX_CH。

5.2 TCS_AMOUNT

按键识别周期参数，即当通道变化率大于手指阈值时每一周期计数加一，当计数值与该参数相等时，在无保护发生的情况下，认为按键有效，电容触摸处理函数返回有效的按键键值，数值越大，判断一个结果用的时间约长。

5.3 _KF8_Disturb_Protect_Circle_ _KF8_Disturb_Protect_Circle_Define

此参数实现异常状态的保护周期，可以设定异常状态的保护时长，在保护期间内，不识别按键判断。值设置为 200 时，即为 200 个扫描周期，该参数有效范围为 0-65535，根据实际情况设置保护周期的大小。

时间与扫描周期关系：首先假定单个按键的执行时间为 1ms。根据按键（包括基准通道）的数量计数出一个扫描周期使用的时间。

5.4 _KF8_Up_BaseLine_Circle_ _KF8_Up_BaseLine_Circle_Define

该参数决定基准线更新的快慢，当参考通道数据稳定且满足当前设定计数时长时调用基准线更新算法。该数据同样以扫描周期为单位。可以根据扫描周期的长短灵活设定该值的大

小。参考数据为64/128/200/300。

5.5 _KF8_TOUCH_CH_EN[MX_CH]

定义当前产品所需开通的通道号。如需开通KF8S1025芯片P2.0和P2.1引脚的触摸通道，根据附录一的表格可知，用户只需给_KF8_TOUCH_CH_EN 数组元素赋上4和5两个值即可。

5.6 _KF8_Inside_Reference_Channel_Order (参考通道声明)

声明芯片的参考通道号。参考通道的变化率可用于识别环境、和瞬间波动，超过设定的波动时系统会进入保护模式，保护模式下不识别按键。这里要求值是通道开启数组的对应通道所在数组小标。

5.7 _KF8_Inside_Reference_Channel_Disturb_Threshold_Set_ (参考通道异常波动阈值设定)

此参数设定参考通道的波动阈值，当考通道变化率的绝对值大于该变量设定的值时，判定触摸系统波动异常，异常进入保护模式。如

_KF8_Inside_Reference_Channel_Disturb_Threshold_Set_ = 15。当参考通道采样数据同上一次数据的变化量差15个千分单位时执行按键保护。具体可以通过实际产品的正常波动变化率调整。因参考属于稳定通道，不存在水影响和触碰到，该值一般设置较小，原则是该变化率值叠加到普通通道上不能因此无识别发声。

5.8 _KF8_Config_Finger_Threshold [MX_CH] 有水识别阈值

有水阈值设定，数组的每一个元素对应一个按键通道。如果通道变化率大于对应的阈值并且累计到TCS_AMOUNT 规定的扫描周期数量，则被判定为有水。阈值的大小可设置为通道需求识别水量时通道总变化率的95%。但随着水位的升高，通道变化率还应有变大的可观空间，否则该数值还要再减小。该值对水位识别高度有直接影响。

5.9 CONFIG_NO_FINGER_THRESHOLD [MX_CH]无水识别阈值

此参数数组的每一个元素对应一个按键通道,在通道变化率小于对应的手指阈值并且累加到TCS_AMOUNT规定的扫描周期数量,则被判定为有水到无水的转换。阈值的大小可为通道需求识别无水量时通道总变化率的105%,但随着水位的升高,通道变化率还应有变大的可观空间,否则该数值还要再增加。但无水识别阈值一定要小于有水识别阈值,否则会状态识别混乱,中间的差值为水位高度的失败死区。

6 附录一 芯片引脚、_KF8_LIBi_channel_flag_ 和 _KF8_TOUCH_CH_EN 对应表

触摸通道 使能配置	按下时 _KF8_LIBi_channel_flag_ 对应值	对应位置	芯片 引脚
CT2	0000 0000 0000 0100	Bit2	P16
CT3	0000 0000 0000 1000	bit 3	P17
CT4	0000 0000 0001 0000	bit 4	P20
CT5	0000 0000 0010 0000	bit 5	P21
CT8	0000 0001 0000 0000	bit 8	P30
CT9	0000 0010 0000 0000	bit 9	P31
CT10	0000 0100 0000 0000	bit 10	P32
CT11	0000 1000 0000 0000	bit 11	P33

如果用户选择使用通道 P30, P31。对应到表中就是 CT8 和 CT9,则需给 _KF8_TOUCH_CH_EN 数组元素赋上 8 和 9,顺序可调换,即当调换顺序时对输出值 _KF8_LIBi_channel_flag 不会造成影响的。

变量_KF8_LIBi_channel_flag的某位为1说明相应通道识别水,如果为0则说明没有识别,通道8和通道9分别识别时,变量_KF8_LIBi_channel_flag的第8位和第9位分别为1(从第0位开始,最低位代表CT0的识别结果)。

7 附录二 通道变化率的采集和计算

通道变化率的采集，我们使用到一个 ChipON TSTool 上位机软件，通过此软件，用户可以直观地看到各个通道的采样值、基准线，使用之前必须处理调试相关函数的使能，debug_touch.c 提供了 3 个函数接口，其中 TOUCH_DEBUG_TRS_DEAL 负责完成曲线输出的使能，TOUCH_DEBUG_TRS_BAIHUALV 选项输出结果为变化率，TOUCH_DEBUG_SEND_OUT 可输出 4 个 byte 数据，通过串口实现产品代码调试功能。但之前还要在 debug_touch.h 文件中设定调试用的 2 个 IO 端口，要求为双向口，默认使用编程接口。

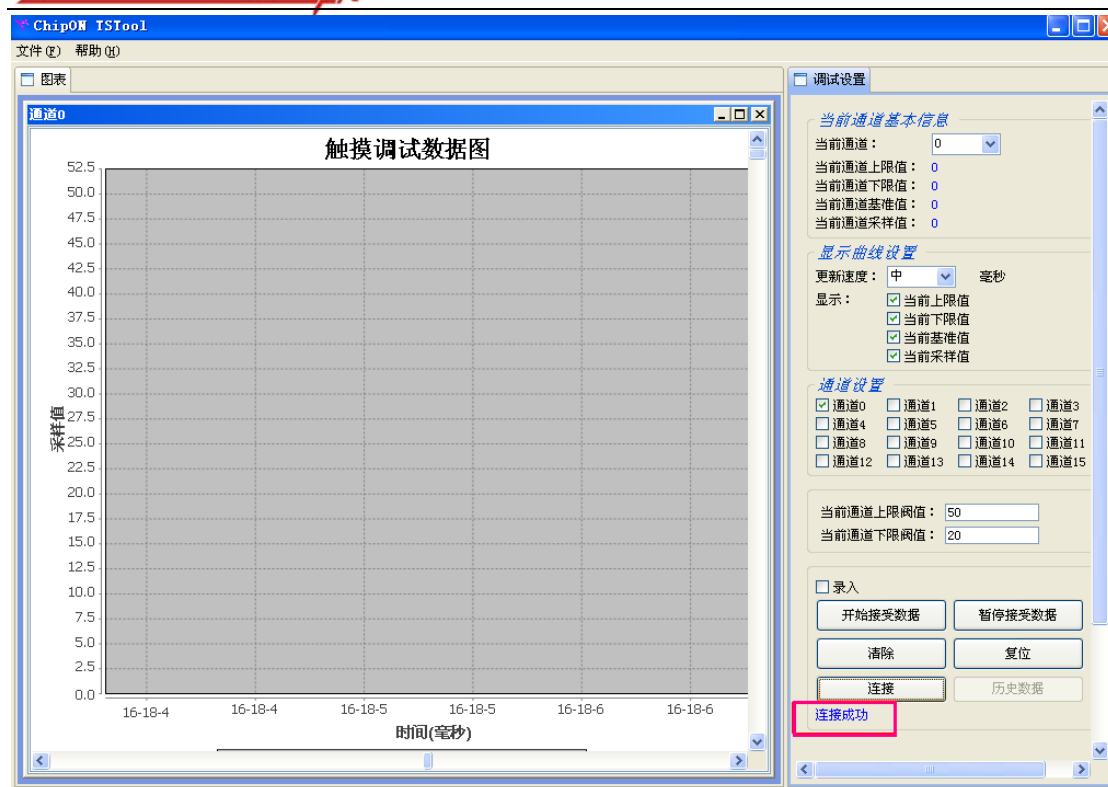
下面介绍如何使用此工具软件。

步骤 1、连接好编程器和产品板，下载触摸库程序完毕。

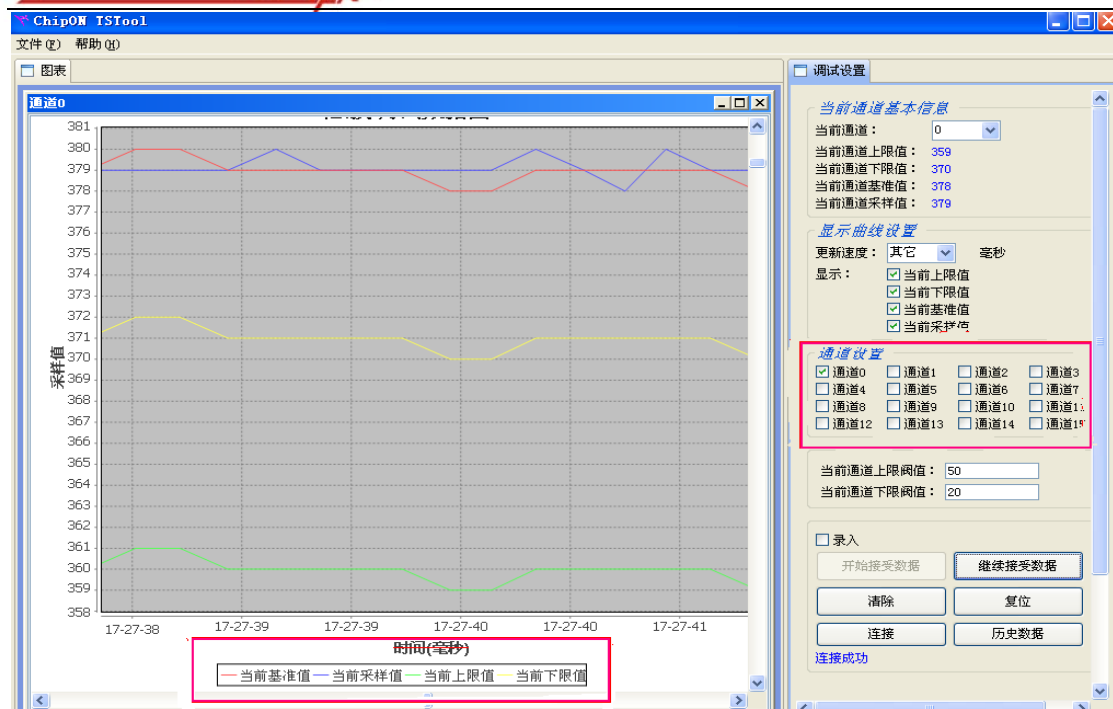
步骤 2、保持编程器和产品板之间的连接，打开 ChipON TSTool 上位机软件，并点击红色区域的【连接】按钮，打开后如下：



步骤 3、连接成功后如下，红色区域提示连接成功。



步骤 4、通过点击【通道设置】中相应的通道并点击【开始接受数据】按钮，在数据图区域则可以看到选定通道的采样值和基准线，选择未开通的通道软件不能正常输出曲线，会报数据接收异常，如果只开启了 CT8, CT9, 选择 8, 9 以外的通道均报异常。点击【暂停接受数据】按钮则暂停接受数据，点击【清除】按钮则清除当前数据图的数据，并根据数据自动适应坐标，将显示区分最大化。点击【复位】按钮则上位机复位，点击【复位】按钮后若要重新接受数据，则需回到步骤 1 重新开始操作。

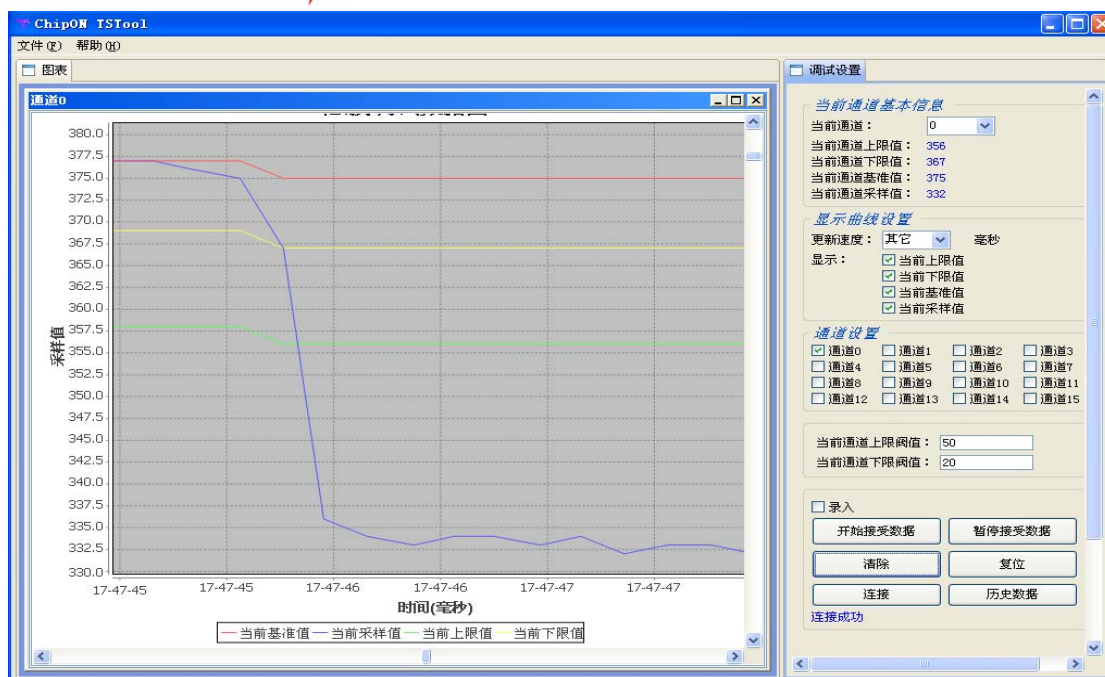


如上图是采集了通道 0 的采样值（蓝色）和基准线（红色），，上位机的通道号和_KF8_TOUCH_CH_EN 数组元素是对应的，即上位机的通道 0 对应数组_KF8_TOUCH_CH_EN 元素值为 0 的通道，芯片的 CT0，上位机的通道 1 对应数组_KF8_TOUCH_CH_EN 元素值为 1 的通道，芯片的 CT1，以此类推。

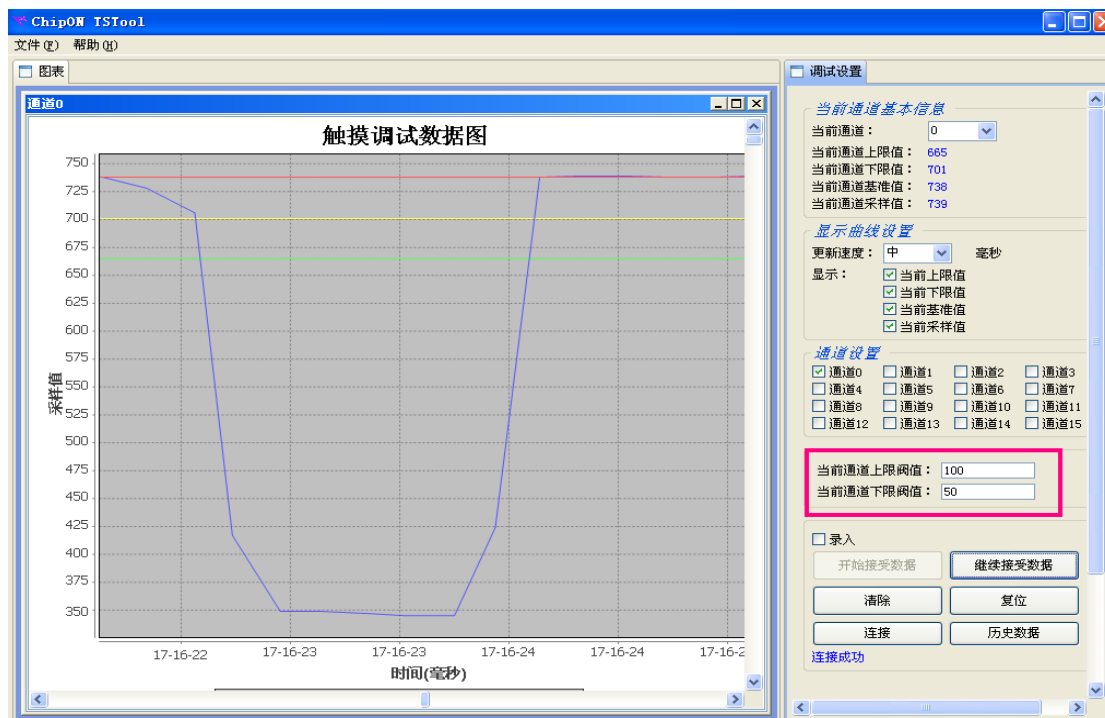
```
unsigned char const _KF8_TOUCH_CH_EN [MX_CH]={
    11,      10,
    9,       8,
    5,       4,
    3,       2,
};
```

如上的 _KF8_TOUCH_CH_EN 设定，上位机的通道 10 则分别对应数组_KF8_TOUCH_CH_EN 元素的 10，访问下标为 1。

步骤 5、计算通道变化率，当通道 0 按下时，从上位机上可看出采样值（蓝色）出现突然下降，按下后取采样值为 332，基准值为 375，则 变化率 = (375-332) / 375 = 0.114,变化率中我们使用千分比进行比较，所以通道变化率为 114，通道变化率为有符号数，正数表示采样值在基准线下方(按下/触水)，负数表示采样值在基准线上方。



步骤 6 设置当前通道上限阈值和当前通道下限阈值，这两个值是为了方便用户可以直观地看到通道按下和释放时通道变化率的范围，用户可以根据不同产品的通道变化率来进行设定。如图：



当前通道上限值：为了比较直观的看到当通道按下时采样值与基准值之间差值的千分比

的范围，我们设定了当前通道上限阈值，当采样值曲线在上限值曲线之下时，我们判定为有水。如上图，当前通道上限阈值设定为 100，我们可以看到上限值为 663（浅绿色），基准值为 737（蓝色）， $(737 - 663) / 737 = 0.1$ ，即千分之 100。

当前通道下限值：同理，为了比较直观地判别按键是否处于释放状态，我们设定了当前通道下限阈值，当采样值曲线在下限值曲线之上时，我们判定为无水状态，如上图，当前通道下限阈值设定为 50，我们可以看到下限值为 700（黄色），基准值为 737（蓝色）， $(737 - 367) / 375 = 0.05$ ，即接近千分之 50。

针对触摸应用，结果识别为手指按下和松开，针对水位识别结果为识别和释放。

实际过程中仅需要针对单个通道，查看计数满足是否满足要求，要求采样数据至少 700，建议 1000-8000 之间。然后根据水的数据变化幅度同默认的 20、50 的参数进行对比。通过估算的方式修改 20、50 的数值，使对应的数据线同蓝线处于一个水平即可，这时填入的数值即为该通道按下带来的变化量数据。该变化量单位为千分率，水位阈值整定可以采用产品上测试水位到达一定高度的变化量和未达到高度的变化量（死区）做参考，一般在该数值的基础上做调整后作为产品的参数，建议水变化量要大于 50，不能达到时通过增加接触面积实现。同时建议阈值设定考虑误差值 20，如有水变化量为 60，无水处变化存在 20，可以将识别阀设定为 40，是否阀设定为 26，具体根据死区大小调整。

参考通道主要用来恒定环境，采样值和基准值往往存在偏离。但实现算法后的普通通道无水时采样值和基准值往往在同一水平。这要求出厂参数存入时的环境真实化，或后续在线校正（需代码实现）。

一般情况下状态识别的滤波计数时长远小于基准的更新时长，但根据系统需要可以设定基准更新比识别快，防止一个更新周期的误差引入带来的差异化识别。这个误差主要还来源于参考通道的正常范围波动，因此波动阈值的设定也很重要。