电容触摸

程序使用说明

（第三版）

上海芯旺微电子有限公司

2015.05

目录

1 电容触摸简介

目前 ChipON 提供的触摸算法程序有通用库和防水库两种。主要应用于 KF8S 系列芯片和 KF8TS2X[1]系列芯片的触摸应用。本文档对触摸库在不同系列芯片上的使用进行说明。以 KF8S1011 触摸芯片为例，该款触摸芯片性能优越，使用方便，具有以下优点：

a) 电容触摸通道多，最多可达 16 个，其中包含了一个内部参考通道；

b) 灵敏度高；

c) 抗干扰能力强，具有防电磁等功能；

d) 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；

2 条件说明

ChipON 触摸芯片需添加外挂电容，外挂电容的大小影响对采样值有一定影响，建议选择 2.2nf 到 44nf 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，建议系统采样值调试至 700--3000 之间。触摸设计请参考另一文档《Chipon 电容触摸设计指南》。

以 KF8S1011 为例，示例中所使用条件如下：

a) 外挂电容为 4.7nf；

b) 亚克力板厚度为 3mm；

3 介绍

3.1 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

使用不同的芯片进行触摸开发时，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：

[1]注：KF8TS2X 包括 KF8TS23、KF8TS24、KF8TS25、KF8TS27，下同

S 系列通用库头文件 kf8s_touch_g.h

S 系列防水库头文件 kf8s_touch_w.h

TS 系列通用库头文件 kf8ts24_touch_g.h

TS 系列防水库：创建头文件 kf8ts24_touch_w.h

② 定义触摸库变量

在项目中创建触摸库的.c 参数文件，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：

S 系列通用库创建文件 kf8s_touch_g.c

S 系列防水库创建文件 kf8s_touch_w.c

TS 系列通用库创建文件 kf8ts24_touch_g.c

TS 系列防水库创建文件 kf8ts24_touch_w.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，具体见 3.4 节。

④ 配置 CMCTL1(KF8S 系列芯片)/CTCTL1（KF8TS 系列芯片）

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。

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，具体见 3.6 节。

S 系列通用库触摸初始化函数：_KF8S_Init_Touch_General_()

S 系列防水库触摸初始化函数：_KF8S_Init_Touch_Waterproof_()

TS 系列通用库触摸初始化函数：_KF8TS24_Init_Touch_General_()

TS 系列防水库触摸初始化函数：_KF8TS24_Init_Touch_Waterproof_()

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，具体见 3.7 节。

S 系列通用库函数名称为_KF8S_Touch_Process_General_()

S 系列防水库函数名称为_KF8S_Touch_Process_Waterproof_()

TS 系列通用库函数名称为_KF8TS24_Touch_Process_General_()

TS 系列防水库函数名称为_KF8TS24_Touch_Process_Waterproof_()

3.2 创建触摸库头文件说明

S 系列通用库 kf8s_touch_g.h 文件如下:

```
#ifndef    KF8S_TOUCH_G_H_
#define    KF8S_TOUCH_G_H_
#include   “main.h”

#define    MX_CH 7
extern unsigned char const     CHS_AMOUNT;
extern unsigned char const     TCS_AMOUNT;
extern unsigned char const     _KF8_Disturb_Protect_Circle_;
extern unsigned int const      _KF8_DoubleKey_Compages_Limitable_Circle_;
extern unsigned int const      _KF8_Key_Max_Pressed_Limitable_Circle_;
extern signed char const       _KF8_BaseLine_Update_Circle_Count_For_UP_;
extern signed char const       _KF8_BaseLine_Update_Circle_Count_For_Down_;
extern unsigned char const     _KF8_Define_Disturb_Channels_Amount_;
extern signed char const       _KF8_BasicLine_UP_Noise_Threshold_;
extern signed char const       _KF8_BasicLine_Down_Noise_Threshold_;
extern signed char const       _KF8_Data_Steady_Threshold_Set_;
extern signed char const       _KF8_Inside_Reference_Channel_Disturb_Threshold_Set_;
extern unsigned int            _KF8_LIBi_channel_flag_;
extern unsigned char           _KF8_LIBc_channel_;
extern unsigned char           _KF8_LIBc_channel_Buf;
extern unsigned char const     _KF8_Inside_Reference_Channel_Declare_;
extern signed char const       _KF8_Config_Finger_Threshold[MX_CH];
```

```
extern unsigned char const    _KF8_TOUCH_CH_EN[MX_CH];
extern signed char            _KF8_LIBi_Date_Change_[MX_CH];
extern signed char            _KF8_LIBc_baseline_count_[MX_CH];
extern unsigned int           _KF8_LIBi_buff_hit_[MX_CH];
extern unsigned int           _KF8_LIBi_buff_baseline_[MX_CH];
extern unsigned char          _KF8_LIBc_touch_count_[MX_CH];

unsigned char       _KF8S_Get_Ver_General_();//获取当前库版本号
void        _KF8S_Init_Touch_General_();//初始化电容触摸内部参数函数
void        _KF8S_Touch_Process_General_();//电容触摸通道处理
void        _KF8S_Touch_TEST_General_();//简化的电容触摸通道处理函数，不识别按键
#endif
```

TS 系列通用库 kf8ts2X_touch_g.h 文件仅与 S 系列通用库 kf8s_touch_g.h 存在头文件定义和函数声明的不同，参数定义一致。TS 通用库 kf8ts2X_touch_g.h 的头文件定义和函数声明如下：

```
#ifndef    KF8TS24_TOUCH_G_H_
#define    KF8TS24_TOUCH_G_H_

unsigned char       _KF8TS24[2]_Get_Ver_General_();//获取当前库版本号
void        _KF8TS24_Init_Touch_General_();//初始化电容触摸内部参数函数
void        _KF8TS24_Touch_Process_General_();//电容触摸通道处理
void        _KF8TS24_Touch_TEST_General_();//简化的电容触摸通道处理函数，不识别按键
```

S 系列防水库 kf8s_touch_w.h 文件的内容为

```
#ifndef    KF8S_TOUCH_W_H_
#define    KF8S_TOUCH_W_H_

#include  "main.h"
```

[2]注：函数名称中的 KF8TS24 仅为 KF8TS24XX 系列芯片使用，KF8TS2X 其余系列需更改为对应的型号名称，后文与此相同

```
#define    MX_CH 7
extern unsigned char const    CHS_AMOUNT;
extern unsigned char const    TCS_AMOUNT;
extern unsigned int const     _KF8_Disturb_Protect_Circle_;
extern unsigned int const     _KF8_DoubleKey_Compages_Limitable_Circle_;
extern unsigned int const     _KF8_Key_Max_Pressed_Limitable_Circle_;
extern signed char const      _KF8_BaseLine_Update_Circle_Count_For_UP_;
extern signed char const      _KF8_BaseLine_Update_Circle_Count_For_Down_;
extern signed char const      _KF8_Define_Disturb_Channels_Amount_;
extern signed char const      _KF8_BasicLine_UP_Noise_Threshold_;
extern signed char const      _KF8_BasicLine_Down_Noise_Threshold_;
extern signed char const      _KF8_Data_Steady_Threshold_Set_;
extern signed char const      _KF8_Inside_Reference_Channel_Disturb_Threshold_Set_;
extern unsigned int           _KF8_LIBi_channel_flag_;
extern unsigned char          _KF8_LIBc_channel_;
extern unsigned char          _KF8_LIBc_channel_Buf;
extern unsigned char const    _KF8_Inside_Reference_Channel_Declare_;
extern signed int const       _KF8_Config_Finger_Threshold[MX_CH];
extern signed int const       _KF8_CONFIG_FINGER_THRESHOLD_IN_WATER_[MX_CH];
extern unsigned char const    Double_Key_SET_EN_1;
extern unsigned char const    Double_Key_SET_EN_2;
extern unsigned char const    Double_Key_SET_EN_3;
extern unsigned char const    _KF8_TOUCH_CH_EN[MX_CH];
extern signed int             _KF8_LIBi_Date_Change_[MX_CH];
extern signed char            _KF8_LIBc_baseline_count_[MX_CH];
extern unsigned int           _KF8_LIBi_buff_hit_[MX_CH];
extern unsigned int           _KF8_LIBi_buff_baseline_[MX_CH];
```

```
extern unsigned char            _KF8_LIBc_touch_count_[MX_CH];

unsigned char       _KF8S_Get_Ver_Waterproof_()；//获取当前库版本号
void        _KF8S_Init_Touch_Waterproof_();//初始化电容触摸内部参数函数
void        _KF8S_Touch_Process_Waterproof_();//电容触摸通道处理
void        _KF8S_Touch_TEST_Waterproof_();//简化的电容触摸通道处理函数，不识别按键
#endif
```

TS 系列防水库 kf8ts24_touch_w.h 仅与 S 系列防水库 kf8s_touch_w.h 存在头文件定义和函数声明的不同，参数定义一致。TS 通用库 kf8ts24_touch_w.h 的头文件定义和函数声明如下：

```
#ifndef KF8TS24_TOUCH_W_H_
#define KF8TS24_TOUCH_W_H_

unsigned char       _KF8TS24_Get_Ver_Waterproof_();//获取当前库版本号
void        _KF8TS24_Init_Touch_Waterproof_();//初始化电容触摸内部参数函数
void        _KF8TS24_Touch_Process_Waterproof_();//电容触摸通道处理
void        _KF8TS24_Touch_TEST_Waterproof_();//简化的电容触摸通道处理函数，不识别按键
```

3.3 定义触摸库变量

根据触摸库类型和芯片类型定义如下变量，变量含义及参数设定原则详见第四章说明。

S 系列通用库 kf8s_touch_g.c 文件和 **TS 系列通用库** kf8ts24_touch_g.c 文件的参数一致，如下所示：

```
#include "main.h"
signed char             _KF8_LIBc_baseline_count_[MX_CH];
unsigned int            _KF8_LIBi_buff_hit_[MX_CH];
unsigned int            _KF8_LIBi_buff_baseline_[MX_CH];
unsigned char           _KF8_LIBc_touch_count_[MX_CH];
signed char             _KF8_LIBi_Date_Change_[MX_CH];
```

```
unsigned char          _KF8_LIBc_channel_;
unsigned char const    CHS_AMOUNT = MX_CH;
unsigned char const    TCS_AMOUNT = 4;
unsigned char const    _KF8_Disturb_Protect_Circle_=100;
unsigned int const     _KF8_DoubleKey_Compages_Limitable_Circle_=300;
unsigned int const     _KF8_Key_Max_Pressed_Limitable_Circle_=8000;
signed char const      _KF8_BaseLine_Update_Circle_Count_For_UP_=32;
signed char const      _KF8_BaseLine_Update_Circle_Count_For_Down_=-32;
unsigned char const    _KF8_Define_Disturb_Channels_Amount_=3;

signed char const      _KF8_BasicLine_UP_Noise_Threshold_=-35;
signed char const      _KF8_BasicLine_Down_Noise_Threshold_=35;
signed char const      _KF8_Data_Steady_Threshold_Set_=10;
signed char const      _KF8_Inside_Reference_Channel_Disturb_Threshold_Set_=15;
unsigned char const    _KF8_TOUCH_CH_EN[MX_CH]={                    };
unsigned char const    _KF8_Inside_Reference_Channel_Declare_=10;
signed char const      _KF8_Config_Finger_Threshold[MX_CH]={        };
unsigned int           _KF8_LIBi_channel_flag_;
extern unsigned int    _KF8_LIBi_channel_flag_;
```

S 系列防水库 kf8s_touch_w.c 文件和 **TS 系列防水库** kf8ts24_touch_w.c 文件的参数一致，如下所示：

```
#include "main.h"
signed char            _KF8_LIBc_baseline_count_[MX_CH];
unsigned int           _KF8_LIBi_buff_hit_[MX_CH];
unsigned int           _KF8_LIBi_buff_baseline_[MX_CH];
unsigned char          _KF8_LIBc_touch_count_[MX_CH];
signed int             _KF8_LIBi_Date_Change_[MX_CH];
```

```
unsigned char           _KF8_LIBc_channel_;
unsigned char const     CHS_AMOUNT = MX_CH;
unsigned char const     TCS_AMOUNT = 4;
unsigned int const      _KF8_Disturb_Protect_Circle_=200;
unsigned int const      _KF8_DoubleKey_Compages_Limitable_Circle_=800;
unsigned int const      _KF8_Key_Max_Pressed_Limitable_Circle_=8000;
signed char const       _KF8_BaseLine_Update_Circle_Count_For_UP_=32;
signed char const       _KF8_BaseLine_Update_Circle_Count_For_Down_=-32;
signed char const       _KF8_Define_Disturb_Channels_Amount_=3;
signed char const       _KF8_BasicLine_UP_Noise_Threshold_=-35;
signed char const       _KF8_BasicLine_Down_Noise_Threshold_=35;
signed char const       _KF8_Data_Steady_Threshold_Set_=10;
signed char const       _KF8_Inside_Reference_Channel_Disturb_Threshold_Set_=15;
unsigned char const     Double_Key_SET_EN_1=0x01;
unsigned char const     Double_Key_SET_EN_2=0x23;
unsigned char const     Double_Key_SET_EN_3=0x45;
unsigned char const     _KF8_TOUCH_CH_EN[MX_CH]={                    };
unsigned char const     _KF8_Inside_Reference_Channel_Declare_=10;
signed int const        _KF8_Config_Finger_Threshold[MX_CH]={        };
signed int const        _KF8_CONFIG_FINGER_THRESHOLD_IN_WATER_[MX_CH]={};
unsigned int            _KF8_LIBi_channel_flag_;
extern unsigned int     _KF8_LIBi_channel_flag_;
```

3.4 配置芯片寄存器说明

1、 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，如 KF8S1011 芯片的内部参考通道是 P3.2 口，则 P3.2 口必须设置为输入态。

◎ 外接电容端口需设置成模拟口，如当选择 P1.2 口为外接电容引脚时，需 ANSEL |= 1<<6;

◎ 触摸用到的寄存器以及 bit 位有 T1H(定时计数器高八位，TS 系列为 T3H)、T1L（定时计数器低八位，TS 系列为 T3H）、T1ON(T1 启动控制位)、T1IE(T1 中断使能位)、T1IF(T1 溢出标志位，TS 系列为 CTIF)、PUIE(在初始化触摸内部参数函数会打开，不需要用户额外设置)、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，无需用户额外开启)。

KF8S1011 芯片范例如下：

```
void init_mcu()
{
    OSCCTL = 0x60;              //8M
    TR0 = 0XFF;                 //设置IO口为输入
    TR1 = 0XFF;
    TR2 = 0X7F;
    TR3 = 0X0F;                 //注意点:TR3.2这一位必须设置为1
    TR4 = 0XFF;
    P0 = 0X00;
    P1 = 0X00;
    P2 = 0x80;
    P3 = 0XF9;
    P4 = 0X00;

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

    //定时器1初始化
    T1CTL = 0X00;               //1：1分频，不开启计数器T1  即T1ON = 0;
```

```
    T1L = 0;
    T1H = 0;
    T1IF = 0;                    // 清中断溢出标志
    AIE = 0;                     // 关总中断
}
```

3.5 配置 CMCTL1/CTCTL1 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阀值。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 寄存器说明如下：

寄存器8.1 CMCTL1：控制寄存器(地址：1AH)

复位值	bit7							bit0
0000 ——	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 8 分频，基准电压为 0.5VDD。

KF8TS 系列芯片的 CTCTL1 寄存器说明如下：

9.2.1 CTCTL1：控制寄存器(地址: 17H)

如寄存器 9.1 所示，CTCLKSEL<1:0>为电容触摸时钟预分频比选择位，可以选择 4 种分频比，分别是 Fosc、Fosc/2、Fosc/4 和 Fosc/8。CTVREFSEL<1:0>为电容触摸基准电压选择位，可选择 2 种，分别为 0.5VDD、0.7VDD。

寄存器9.1 CTCTL1：控制寄存器(地址：17H)

	bit7							bit0
复位值	CTCLKSEL1	CTCLKSEL0	CTVREFSEL1	CTVREFSEL0	-	-	-	-
0000 ----	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

01 = Fosc/2

10 = Fosc/4

11 = Fosc/8

CTVREFSEL<1:0>电容触摸基准电压选择位

01 = 0.5VDD

10 = 0.7VDD

如 CTCTL1 = 0x50; 则 设置触摸时钟为 2 分频，基准电压为 0.5VDD。

3.6 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，建议 500ms。如 S 系列通用库样例如下：

```
delay_ms(500);
_KF8S_Init_Touch_General_();
```

3.7 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，TS 系列为 CTIF。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。

以 S 系列通用库为例如下：

```
void INT_FUN() __interrupt    （0）
{
    if(T1IF) // TS系列标志位CTCTL1
    {
        T1IF=0;
        _KF8S_Touch_Process_General_ ();        // 在中断中执行触摸函数
    }
}
```

也可以在中断中设置标志位，然后在主函数中进行调用，范例如下：

```
void INT_FUN() __interrupt    （0）
{
    if(T1IF) //
    {
        T1IF=0;
        touch_process_flag =1;              // 在中断中置标志位
    }
}
```

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
    _KF8S_Touch_Process_General_ (); // 在主函数中执行触摸函数
}
```

3.8 获取触摸库程序版本

触摸库中提供了专门的函数输出程序版本号，版本号为八位数据，高四位为系列号，低

四位为版本号。具体为：

1X 通用库S系列芯片，如KF8S1011

2X 防水库S系列芯片，如KF8S1011

3X 通用库TS系列芯片，如24TS2414

4X 防水库TS系列芯片，如24TS2414

X1 为首版本

S 系列通用库函数名称为_KF8S_Get_Ver_General_()。

S 系列防水库函数名称为_KF8S_Get_Ver_Waterproof_ ()。

TS 系列通用库函数名称为_KF8TS2X_Get_Ver_General_()。

TS 系列防水库函数名称为_KF8TS2X_Get_Ver_Waterproof_()。

S系列通用库，在主函数中作如下定义

```
unsigned char temp;
temp=_KF8S_Get_Ver_General_();
```

执行完后temp的值为0x11，说明当前库为S系列通用库首次发布版本。

3.9 简化的触摸库处理函数说明

此函数是触摸库函数的简化版本，当用户需要使用上位机查看采样值时，编译时如果发现程序存储空间不足，则可以把原始的触摸库处理函数改为简化版的。

S 系列通用库函数名称为_KF8S_Touch_TEST_General_ ()。

S 系列防水库函数名称为_KF8S_Touch_TEST_Waterproof_ ()。

TS 系列通用库函数名称为_KF8TS24_Touch_TEST_General_()。

TS 系列防水库函数名称为_KF8TS24_Touch_TEST_Waterproof_ ()。

以 S 系列为例，当需要使用简化版触摸库函数时，则把_KF8S_Touch_Process_General_()函数更改为_KF8S_Touch_TEST_General_()即可，_KF8S_Touch_TEST_General_()不具有按键识别功能，只能输出采样值，供调试参考。

4 触摸库参数及设置说明

通过上一章节的六个步骤，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。

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。

4.1 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kf8s_touch_w.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。

4.2 TCS_AMOUNT：

按键识别周期参数，即当通道变化率大于手指阀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。

4.3 _KF8_Disturb_Protect_Circle_

此参数为异常状态的保护周期，可以设定异常状态的保护时间，在保护期间内，不识别新的按键。值设置为 200 时，即为 200 个扫描周期，该参数有效范围为 0-65535，更具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。

4.4 _KF8_DoubleKey_Compages_Limitable_Circle_

此参数设置允许双键组合的识别时间。如该参数设定为100。当组合键第一个按键被识

别后。必须在100个周期的时间范围内第二按键被按下识别才有效，。识别第一个按键后，超过该双键识别时间，第二个按键即使被按下也不识别。仅保留第一个被按下的识别。当该参数指定值为0时，关闭时间限定。即不论任何时长后只要第一按键不松开，第二按键按下均有效识别。该参数范围为0-65535。

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。

4.5 _KF8_Key_Max_Pressed_Limitable_Circle_

此参数设定按键按下被识别的最长周期，超过该周期按键自动释放，所以该值不能设置过小。如果 _KF8_Key_Max_Pressed_Limitable_Circle_ > 0, 则当用户按下当前键超出 _KF8_Key_Max_Pressed_Limitable_Circle_ 规定的扫描周期数且没有释放，则当前键会被判定为自动释放，避免了因某个按键被按下时间过长而导致其它键无法识别的情况。 当设定 _KF8_Key_Max_Pressed_Limitable_Circle_ = 0，则时间限定概念失效，即用户按下按键时间为无限长且不会被判定为自动释放。

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。

4.6 _KF8_ BaseLine_Update_Circle_Count_For_UP_

该参数决定基准线向上更新的快慢，当数据稳定且满足当前设定计数次数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16/32/64。

4.7 _KF8_ BaseLine_Update_Circle_Count_For_Down_

该参数决定基准线向下更新的快慢，当数据稳定且满足当前设定计数次数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-16/-32/-64。

4.8 _KF8_Define_Disturb_Channels_ Amount_

此参数设置超过噪声阈值的通道数量多于多少个属于异常。默认为3个通道的数据超过噪声阀值系统进入保护模式，在_KF8_Disturb_Protect_Circle_定义的周期时间内不进行按

键识别，如同时按下三个按键或者电源抖动等引起的异常变化都会进入该模式。因为双键的功能，该参数不建议设置为2.

4.9 _KF8_ BasicLine_UP_Noise_Threshold_ (上噪声门限)

此参数设定采样值在基准线上边时的噪声阀值，变化率在噪声门限内的基准线正常更新。当一些突变因素（如移除盖板）导致采样值大幅上升，变化率超出噪声门限时，稳定计数满足100个周期立即提升基准线至当前的采样值附近。

该参数的设定可以采用最大变化量的40%作为噪声门限，即当前产品的按键按下时产生的通道变化率为千分之80，则可设定为：80*40% = -32，也可以通过实验法，根据测试物比如盖板移除带入的实际变化量设定。这里规定数据向上变化率计值为负数。有效参数为-125到-5。

4.10 _KF8_ BasicLine_Down_Noise_Threshold_ (下噪声门限)

此参数设定采样值在基准线下边的噪声阀值，噪声门限内的基准线正常更新，不识别按键。噪声门限外按键未识别的通道基准线采用慢速更新。同时该门限为通道异常时的判定阀值，某一通道的变化率大于该门限时计数加 1 并且不重复计数，3 个及其以上通道变化率大于该门限时，判定触摸系统波动异常，异常后程序进行_KF8_Disturb_Pro tect_Circle_的周期数的按键识别保护。

此阈值参考设定为按键按下时通道变化率的 40%。如当前产品的按键按下时产生的通道变化率为千分之 80，可设定值为：80*40% = 32。也可以根据作用适当调整该值的大小。该参数的有效范围是 5-125。

4.11 _KF8_Data_ Steady_Threshold_Set_ （数据稳定设定门限）

上一次采样值偏离基准线变化率与当前变化率的差值大于该数值时认定系统波动，波动情况下不更新基准线，基准线只在稳定数据稳定情况下更新。该值的大小设定和产品的实际波动范围相关。阈值的大小应该要满足系统自身采样值的波动，建议设定值为4到15。

4.12 _KF8_Config_Finger_Threshold[MX_CH] 无水按下手指阀值

手指阈值设定，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手指阈值并且累计到TCS_AMOUNT 规定的周期时间，则被判定为按下。手指阈值的大小可设置为

按键按下时通道总变化率的80%。如果调试时发现按键按下未被识别，则应该减小相应通道的手指阈值，如按键有误触发现象，则应该增大相应通道的手指阈值。在附录二的步骤5中，当通道0按下时我们计算出其通道变化率为114，则对应的手指阈值可以设置为 114*80% = 91，该值对按键灵敏度有直接影响。

通用库该数为有符号char型数据，可以设定的范围20到126，防水库为有符号int型数据。

4.13 KF8_TOUCH_CH_EN[MX_CH]

定义当前产品所需开通的通道号。如需开通KF8S1011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。

4.14 _KF8_Inside_Reference_Channel_Declare_ (内部参考通道声明)

声明芯片的内部参考通道号。内部参考通道的变化率可用于识别电源波动，超过设定的电源波动时系统会进入保护模式，保护模式下不识别按键，不占用外部引脚资源。不同型号的芯片其内部基准参考通道号不同，部分芯片没有内部参考通道。如KF8S1011芯片的内部参考通道为CT10，可以定义_KF8_Inside_Reference_Channel_Declare_=10即可。相对于没有内部参考通道的芯片可以使用_KF8_Inside_Reference_Channel_Declare_=40。

4.15 _KF8_Inside_Reference_Channel_Disturb_Threshold_Set_ (参考通道异常波动阀值设定)

此参数设定内部基准参考通道的阀值，当内部基准参考通道变化率的绝对值大于_KF8_Inside_Reference_Channel_Disturb_Threshold_Set_设定的值时，判定触摸系统电源异常，异常进入保护模式。如_KF8_Inside_Reference_Channel_Disturb_Threshold_Set_ = 25。当内部参考通道采样数据同上一次数据的变化量差25个千分单位时执行按键保护。可以设定的范围是5-125。具体可以通过实际产品的正常波动变化率调整。参考设定为25，有效数据范围0~127。

4.16 _KF8_CONFIG_FINGER_THRESHOLD_IN_WATER_[MX_CH]有水按下的手指阈值

此参数仅在防水库中包含，用于设定有水情况下判定按键按下的手指阈值。数组的每一个元素对应一个按键通道，在按键有水覆盖且通道变化率大于对应的手指阈值并且累加到

TCS_AMOUNT规定的次数时，则被判定为按下。由于有水泼下时采样值下降，但基准线不作下调处理，所以有水按下的手指阈值较无水按下的手指阈值大。

注：触摸按键识别在有水和无水情况下的最大区别在于有水时当前键按下会对邻键造成较大串扰，这对触摸库的调试造成一定困难。在此列举一种常见的情况进行分析。

如产品中通道1和通道2的触摸按键在物理位置上相邻。当有水泼下时通道1和通道2同时被水覆盖。当分别按下两个通道按键时采集到的通道变化率如表所示：

	通道1变化率	通道2变化率
通道1按键按下	100	110
通道2按键按下	90	115

在触摸库中，按键被识别为按下必须要满足的两个条件是

1、 当前通道变化率必须大于对应的手指阈值

2、 当前通道变化率必须是所有通道变化率最大的2个之一。

如表格所示，当通道1按下时，如果通道2的手指阈值设置为108，则通道2会满足如上两个条件（假设其他通道变化率都小于100），所以会被判定为按下，造成误触发。如果通道2的手指阈值设定为112，则可以避免误触发。

4.17 Double_Key_SET_EN_双键定义

此参数仅在防水库中包含，用于对双键组合的定义。在防水库中，双键按下最多能有三种组合，分别通过Double_Key_SET_EN_1、Double_Key_SET_EN_2、Double_Key_SET_EN_3三个变量进行设置。

样例：

TOUCH_CH_EN数组如下定义

unsigned char const TOUCH_CH_EN[MX_CH]={2,4,3,0,5,1,10};

如果我们希望设定通道0和通道3为一个双键组合，即可将通道0和通道3在数组TOUCH_CH_EN中的下标号（必须小于16）设定为组合码，即数组TOUCH_CH_EN[2]为3，TOUCH_CH_EN[3]为0，所以此组合码为0x23或者0x32。当所需组合数小于3时，可以将多余组合码设定为0xFF。即当前样例下，如果我们只需设定通道0和通道3为双键组合，则组合变量的值为

Double_Key_SET_EN_1=0x23;

Double_Key_SET_EN_2=0xFF;

Double_Key_SET_EN_3=0xFF;

5 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	_KF8_TOUCH_CH_EN 数组元素大小	芯片 引脚
CT0	0000 0000 0000 0001	0	P0.7
CT1	0000 0000 0000 0010	1	P0.6
CT2	0000 0000 0000 0100	2	P1.6
CT3	0000 0000 0000 1000	3	P1.7
CT4	0000 0000 0001 0000	4	P2.0
CT5	0000 0000 0010 0000	5	P2.1
CT6	0000 0000 0100 0000	6	P2.2
CT7	0000 0000 1000 0000	7	P2.3
CT8	0000 0001 0000 0000	8	P3.0
CT9	0000 0010 0000 0000	9	P3.1
基准通道		10	-
CT11	0000 1000 0000 0000	11	P3.3
CT12	0001 0000 0000 0000	12	P3.4
CT13	0010 0000 0000 0000	13	P3.5
CT14	0100 0000 0000 0000	14	P3.6
CT15	1000 0000 0000 0000	15	P3.7

如果用户有两个触摸按键连接在芯片的 P2.0 和 P2.1 引脚上，并且希望开通这两个通道。对应到表中就是通道 4 和通道 5,则只需给_KF8_TOUCH_CH_EN 数组元素赋上 4 和 5，并且顺序可调换，即当调换顺序时对输出值_KF8_LIBi_channel_flag 是不会造成影响的。即数组_KF8_TOUCH_CH_EN 为

unsigned char const _KF8_TOUCH_CH_EN[MX_CH]={4,5,10};10通道为内部参考通道

变量_KF8_LIBi_channel_flag的某位为1说明相应通道按下，如果为0则说明没有按下，通道4和通道5分别按下时，变量_KF8_LIBi_channel_flag的第4位和第5位分别为1（从第0位

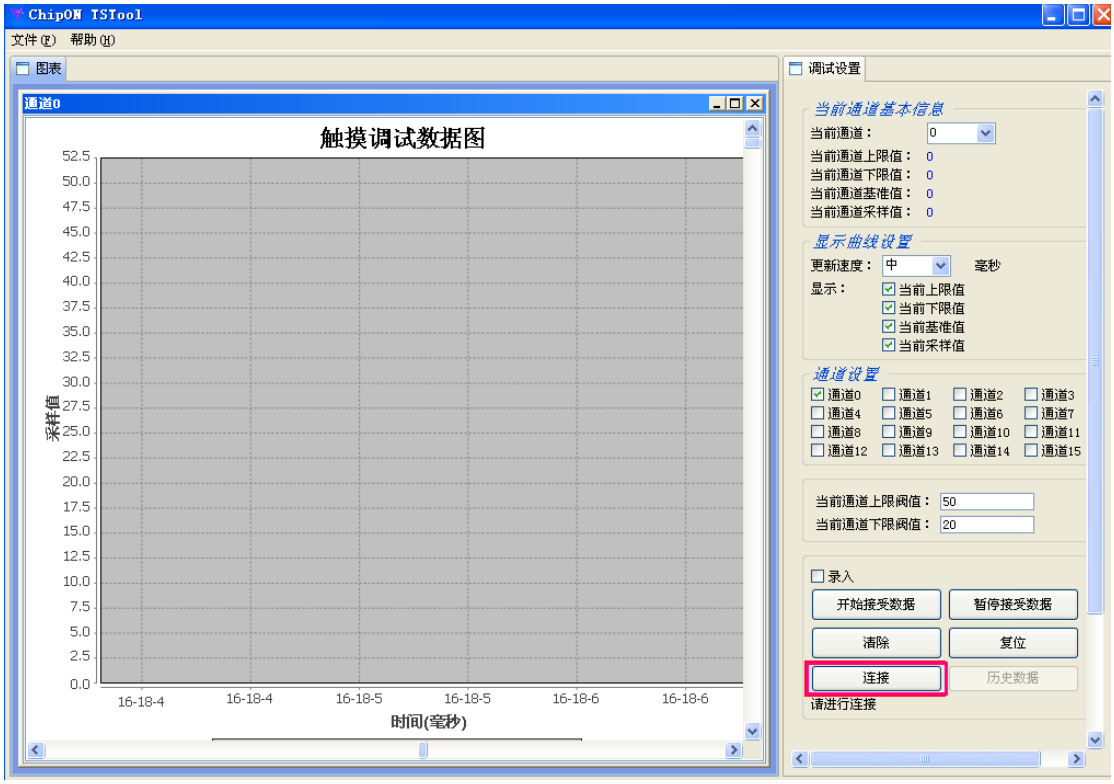
数起）。

6 附录二 通道变化率的采集和计算

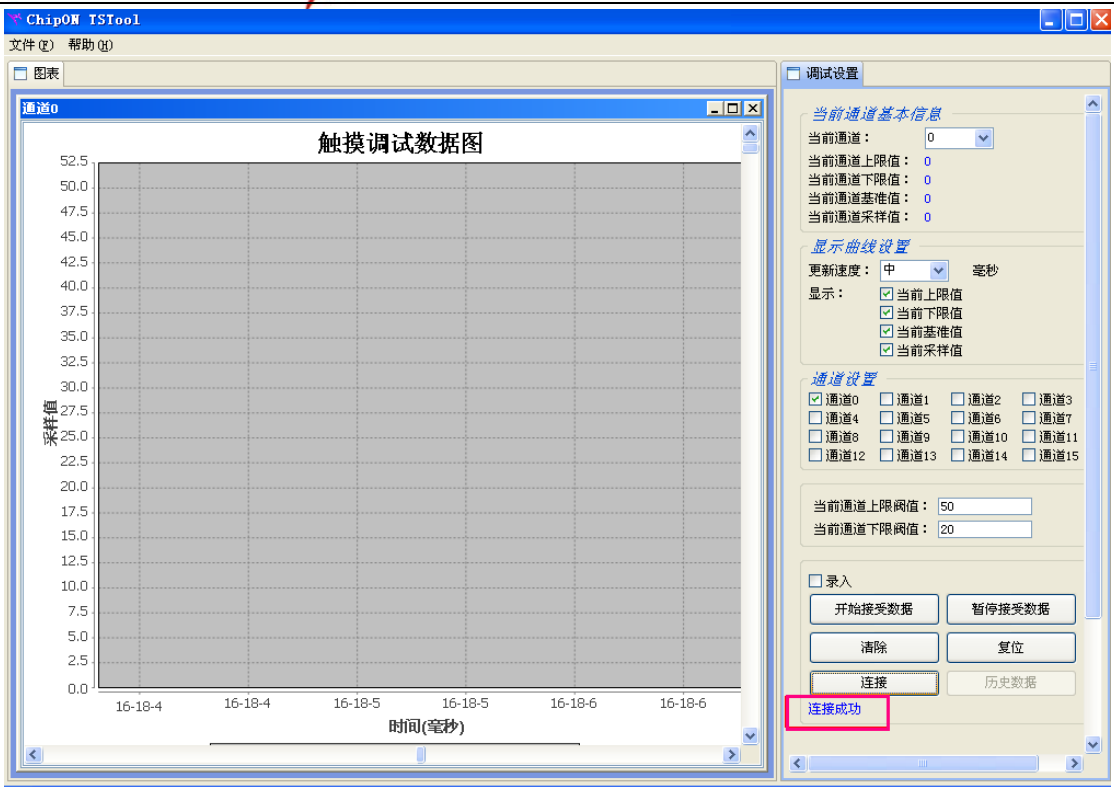
通道变化率的采集，我们使用到一个 ChipON TSTool 上位机软件，通过此软件，用户可以直观地看到各个通道的采样值、基准线。下面介绍如何使用此工具软件。

步骤 1、连接好编程器和 Demo 板，下载触摸库程序完毕。

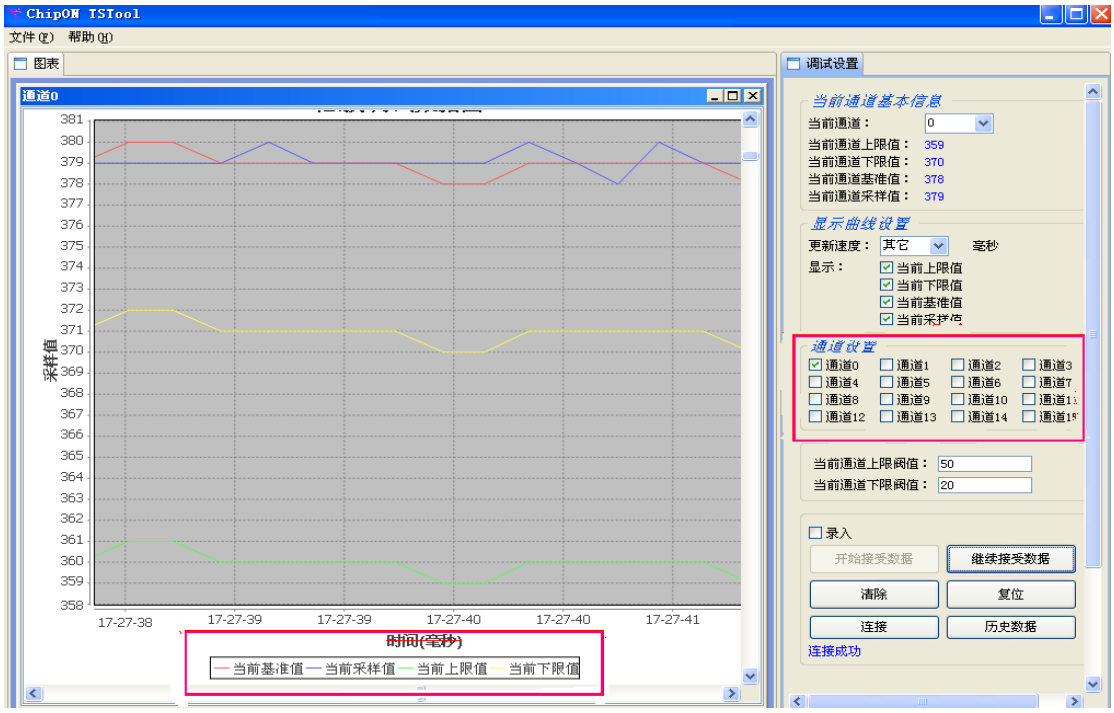
步骤 2、保持编程器和 Demo 板之间的连接，打开 ChipON TSTool 上位机软件，并点击红色区域的【**连接**】按钮，打开后如下：



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，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，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。如下图：

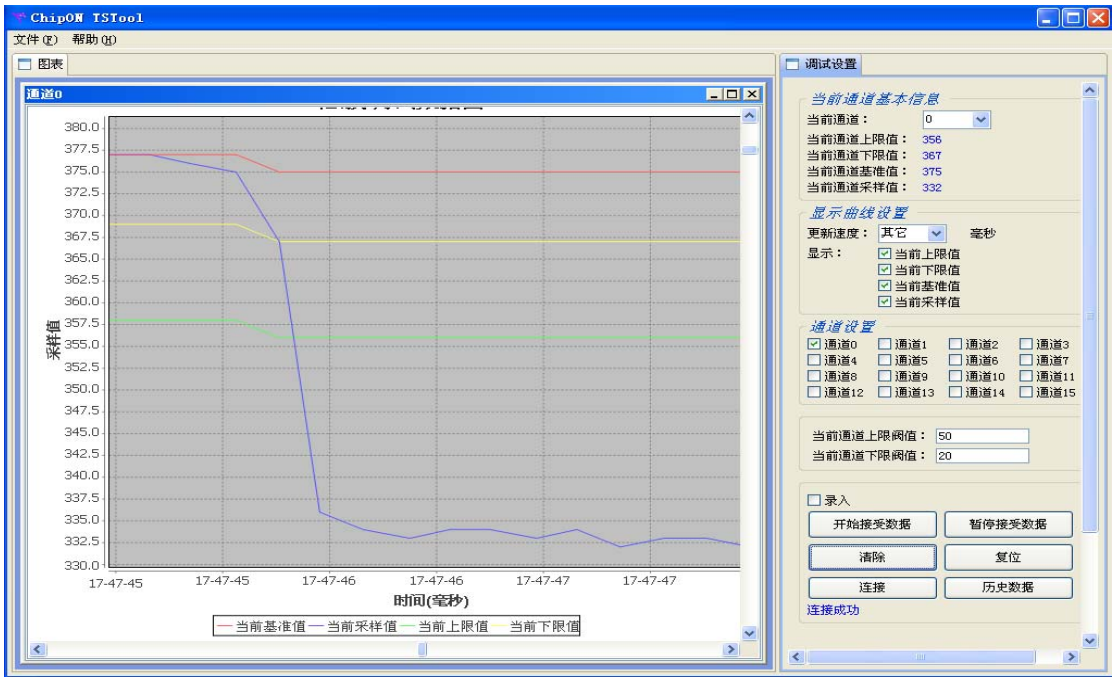


如上图是采集了通道 0 的采样值（蓝色）和基准线（红色）,，上位机的通道号和_KF8_TOUCH_CH_EN 数组元素是对应的，即上位机的通道 0 对应数组_KF8_TOUCH_CH_EN 元素值为 0 的通道，芯片的 CT0，上位机的通道 1 对应数组_KF8_TOUCH_CH_EN 元素值为 1 的通道，芯片的 CT1，以此类推。

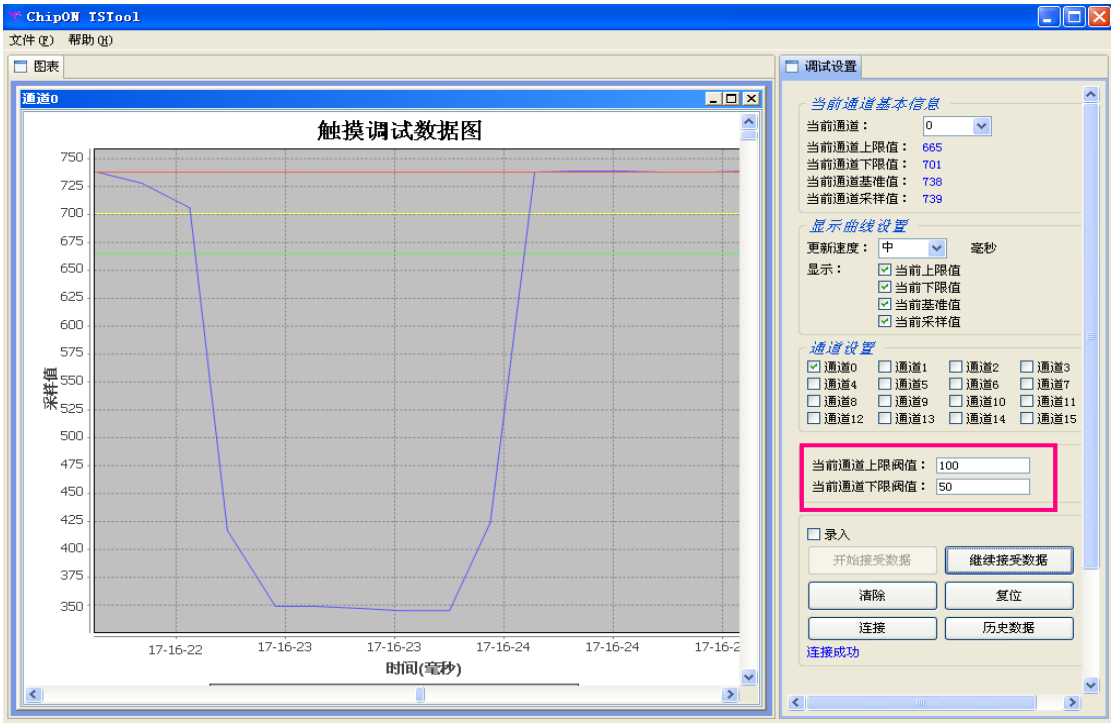
```
unsigned char const _KF8_TOUCH_CH_EN [MX_CH]={
                                    0,
                                    1,
                                    2,
                                    3,
                                    4,
                                    5,
                                    10,
                                };
```

如上的 _KF8_TOUCH_CH_EN 设定，上位机的通道 0 - 5 则分别对应数组_KF8_TOUCH_CH_EN 元素的 0 - 5。

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，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，我们判定为按下操作。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，我们判定为释放状态，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。