

简介

本应用笔记主要讲述如何基于HolyChip的家电产品线（Home Appliance Product Line）提供的触摸库函数例程同时结合实际的应用环境来进行项目开发，从而缩短用户的项目开发周期。本笔记将以HC89F30xC为例介绍触摸库的使用。

- 触摸库函数例程版本：HC89F30xC_Touch_Library_Code V3.0.0.0。
- 本应用笔记适用芯片：HC89F30xC 系列芯片。
- 相关数据手册、工具及技术文档下载网址：<http://www.holychip.cn/>。

目录

1	资料获取	3
2	开发环境	3
3	触摸库	4
3.1	函数介绍	4
3.2	触摸相关参数配置介绍	6
3.3	触摸调试	14
3.4	注意事项	18
4	版本说明	19

1 资料获取

前往芯圣官网 <http://www.holychip.cn/> →产品中心→家电 MCU，点击 HC89F30xC，在“应用指南”类中有“HC89F30xC 系列触摸库使用说明”选项，点击下载。

下载完成后，进行压缩文件解压。

2 开发环境

编译环境：Keil C51，uVision4.00 及以上版本。

仿真工具：HC-LINK，可进行程序的下载和仿真。

烧录工具：HC-PM51，量产烧录工具。

直接在浏览器地址栏中输入地址：www.holychip.cn 进行软件下载。

3 触摸库

HolyChip 提供的 HC89F30xC _Touch_Library_Code 触摸例程库已建立了一个完整的项目工程。程序框架已搭建完成,用户先根据自己的需要在指定的文件中进行相应的配置,然后调用对应 API 函数即可。用户配置文件: HC89F30xC _GeneralKey_Touch_Lib_CFG.h、HC89F30xC _GeneralKey_Touch_Auxiliary.h。

3.1 函数介绍

3.1.1 CTK_Init

描述: CTK 初始化函数,调用此程序前必须先打开中断总开关 (EA=1),因为 CTK 初始化时,需要 CTK 中断配合

C 语言原型: void CTK_Init(void)

输入参数: 无

返回值: 无

3.1.2 CTK_ISR_Handle

描述: CTK 中断处理函数,此函数必须在 CTK 中断入口中调用,CTK 中断优先级默认为最低

C 语言原型: CTK_ISR_Handle(void)

输入参数: 无

返回值: 无

3.1.3 TouchRawDataFilter

描述: RawData 滤波函数,如需使能 RawData 数据滤波,需在 HC89F30xC _GeneralKey_Touch_Auxiliary.h 中设置滤波深度,此函数不可以删除,且必须在 Touch_Signal_Handle 之前调用

C 语言原型: TouchRawDataFilter (void)

输入参数: 无

返回值: 无

3.1.4 Touch_Signal_Handle

描述: 扫描完成一轮后, 调用一次, 对各触摸通道基线进行动态更新

C 语言原型: Touch_Signal_Handle(void)

输入参数: 无

返回值: 无

3.1.5 TouchMultibuttonHandle

描述: 按键模式选择函数, 支持多按键模式、单按键模式以及双按键模式

C 语言原型: TouchMultibuttonHandle (void)

输入参数: 无

返回值: 无

3.1.6 Scan_Pause

描述: CTK 扫描暂停, 调用本函数后, 只有当前通道扫描结束后才会退出次函数, 即本函数占用最长时间为一通道扫描完成时间

C 语言原型: Scan_Pause(void)

输入参数: 无

返回值: 无

3.1.7 Scan_Continue

描述: CTK 扫描继续

C 语言原型: Scan_Continue(void)

输入参数: 无

返回值: 无

3.1.8 BaseLineRest

描述：复位所有通道基线值

C 语言原型：BaseLineReset(void)

输入参数：无

返回值：无

3.2 触摸相关参数配置介绍

注：设置长/短按键生效最长时间、设置噪声阈值更新时间、设置低基准线复位时间这四个值均可以转换为对应的时长，详细的计算公式请参考本文 3.3.2.1 章节 [《基准线更新原理》](#)。

3.2.1 设置需要检测的通道

参数修改文件：HC89F30xC _GeneralKey_Touch_Lib_CFG.h

```
#define HCTouchLib_TK_CH_MASK      (x,x,x,x,x,x,x,x,x,x,x,x,x,x,x,x)
```

其中 0：不检测；1：允许检测。

注意：需要打开某个通道，只需将对应 TKx 配置为 1（如图：打开 TK1，对应 TK1 配置为 1）

通道对应标号如下图所示：

```
//      设置需要检测的通道标志 ----->[0: 不检测; 1: 允许检测]
//      HC89F3xx1----->      TK0 | TK1 | TK2 | TK3 | TK4 | TK5 | TK6 | TK7 | TK8 | TK9 | TK10 | TK11 | TK12 | TK13 | TK14 | TK15 |
#define HCTouchLib_TK_CH_MASK      { 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 }
```

图 3-1

3.2.2 设置需要检测的通道个数

参数修改文件：HC89F30xC _GeneralKey_Touch_Lib_CFG.h

```
#define HCTouchLib_OPENED_TK_AMOUNT      x
```

注意：此处个数设置必须与被允许检测通道个数对应，否则程序执行会出错。如：通道标志一共设置了六个，则此处必须填 6。

3.2.3 设置短按按键及长按按键

参数修改文件：HC89F30xC _GeneralKey_Touch_Lib_CFG.h

```
#define HCTouchLib_TK_LONGKEY      (x,x,x,x,x,x,x,x,x,x,x,x,x,x,x,x)
```

其中 0：短按按键；1：长按按键。

注意：需设置长短按键，只需将对应通道置 1，如：打开 TK1、TK2，将 TK1 设为长按键，只需将对应通道设为 1。

通道对应标号如下图所示：

```
// 设置需要检测的通道标志 ----->[0: 不检测; 1: 允许检测]
// HC89F3xx1-----> | TK0 | TK1 | TK2 | TK3 | TK4 | TK5 | TK6 | TK7 | TK8 | TK9 | TK10 | TK11 | TK12 | TK13 | TK14 | TK15 |
#define HCTouchLib_TK_CH_MASK { 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 }

// 设置短按按键及长按按键 ----->[0: 短按按键; 1: 长按按键]
// 使用注意：本配置必须配合长/短按键生效最长时使用
// HC89F3xx1-----> | TK0 | TK1 | TK2 | TK3 | TK4 | TK5 | TK6 | TK7 | TK8 | TK9 | TK10 | TK11 | TK12 | TK13 | TK14 | TK15 |
#define HCTouchLib_TK_LONGKEY { 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 }
```

图 3-2

3.2.4 设置触摸相关寄存器

参数修改文件：HC89F30xC _GeneralKey_Touch_Lib_CFG.h

- 设置数据采样时钟个数：

```
#define HCTouchLib_DSCR x
```

设置范围（0~255）。

采样时钟个数越大，计数值越大。

单个通道扫描时间 $\approx ((\text{HCTouchLib_DSCR}+1)*256 - 1)/(\text{Fosc}/2)$

- 设置触摸充电电源电压：

```
#define HCTouchLib_VD x
```

设置范围（0~255）：0->2v,1->2.5v,2->3v,3->4v。

充电电源电压越大，计数值越大。

使用注意：必须保证芯片 VDD 电压 > 用户设置充电电源电压+0.5V

- 设置触摸放电电阻：

```
#define HCTouchLib_RBS x
```

设置范围（0~7）：0->放电电阻最小，7->放电电阻最大，方向 0(小)-->7(大)。

放电电阻越小，触摸计数值越小。

3.2.5 设置手指触摸相关阈值参数

主要围绕下图，设置手指触摸相关的一些阈值参数，如迟滞值、手指触摸值、噪声、手指触摸或离开确认次数等。

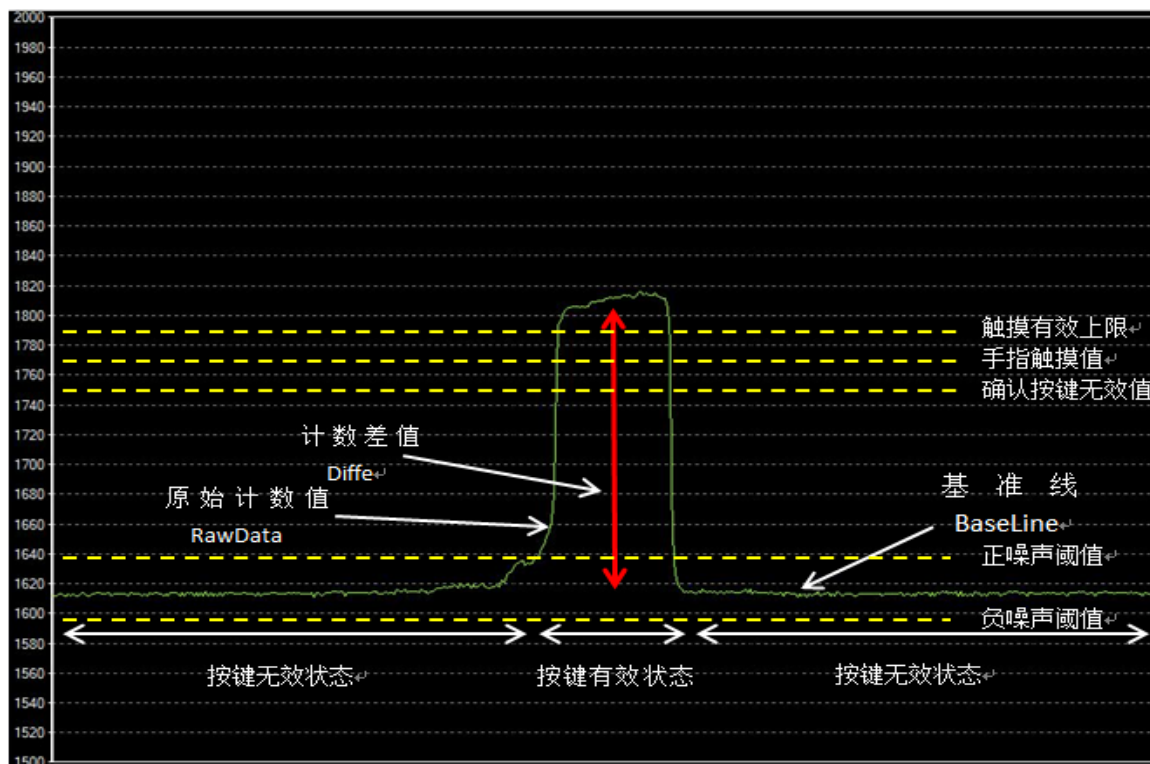


图 3-3 手指触摸参数

参数修改文件：HC89F30xC _GeneralKey_Touch_Lib_CFG.h

➤ 手指触摸值：

```
#define HCTouchLib_NOISE_THRESHOLD (x,x,x,x,x,x,x,x,x,x,x,x,x,x,x,x)
```

设置范围（25~65535）。

其中 x 取值需具体应用环境及 CTK SFR 配置共同决定。

注意：设置手指触摸值时，需要注意设置的位与使用的 CTK 通道对应。

➤ 设置确认按键有效值比例：

```
#define HCTouchLib_CONFIRM_TOUCH_RATIO x
```

设置范围（40~1000）。

该参数为触摸按键的状态由无效转换为有效时最小值，当 $Differ > \text{手指触摸值} * \text{确认按键有效值比例} / 100$ 后按键生效。

- 设置确认按键无效值比例:

```
#define HCTouchLib_CONFIRM_NOTOUCH_RATIO    x
```

设置范围（20~1000）。

该参数为触摸按键的状态由有效转换为无效时最大值，当 $Differ < \text{手指触摸值} \times \text{确认按键无效值比例} / 100$ 后按键失效。

- 设置触摸有效上限值:

```
#define HCTouchLib_LIMIT_FACTOR    x
```

设置范围（0~255）。

该参数为触摸有效上限值，当手指按在触摸按键上， $Differ$ 值 $>$ 触摸有效上限值时，按键生效。

上限值 = $HCTouchLib_LIMIT_FACTOR \times \text{对应通道的手指触摸值}$ ，0 表示不设置上限

- 设置按键确认消抖次数，即触摸有效状态消抖:

```
#define HCTouchLib_CONFIRM_TOUCH_TIME    x
```

设置范围（1~255）。

该参数为触摸按键的状态由无效转换为有效时进行去抖计数，手指触摸有效信号应连续大于等于该参数时才会认为触摸按键有效。该参数对所有使能的触摸按键有效。

按下消抖时间 = 单个按键扫描时间 \times 开启的检测通道个数 \times 按键确认消抖次数

- 设置按键离开消抖次数，即触摸无效状态消抖:

```
#define HCTouchLib_CONFIRM_NOTOUCH_TIME    x
```

设置范围（1~255）。

该参数为触摸按键的状态由有效转换为无效时进行去抖计数，手指触摸无效信号应连续大于等于该参数时才会认为触摸按键无效。该参数对所有使能的触摸按键有效。

松开消抖时间 = 单个按键扫描时间 \times 开启的检测通道个数 \times 按键离开消抖次数

3.2.6 设置基线更新参数

以下参数的功能及使用原理见本文 3.3.2 章节 [《基准线更新》](#)。

参数修改文件：HC89F30xC _GeneralKey_Touch_Lib_CFG.h

- 设置短按键生效最长时间计数值，用于触摸短按键或适应外界条件变化:

```
#define HCTouchLib_CONFIRM_SHORTTOUCH_TIME x
```

设置范围（0~2000）

- 设置长按键生效最长时间计数值，用于触摸长按键或适应外界条件变化：

```
#define HCTouchLib_CONFIRM_LONGTOUCH_TIME x
```

设置范围（SHORTTOUCH_TIME ~65535）

当 Differ 大于手指触摸值（HCTouchLib_FINGER_THRESHOLD）的时长（时间由下面的公式可推出）超过了所设的该值后，系统将自动更新当前 Baseline。该设置虽限制了触摸按键有效的最长时间（时间由下面的公式可推出），但可以防止以下情况发生：由于某种干扰或故障，虽然手指未触摸，但触摸计数差值 Differ 却突然大于所设的手指触摸值，导致触摸处于异常工作状态。

注意：此值可以在程序运行时修改，对应的变量为 XDATA_CONFIRM_LONGTOUCH_TIME、XDATA_CONFIRM_SHORTTOUCH_TIME。

长/短按键最长有效时间 \approx HCTouchLib_CONFIRM_LONGTOUCH_TIME（SHORTTOUCH_TIME） \times 单个按键扫描时间 \times 开启的检测通道个数

- 设置判断触摸有效上限值（毛刺的判断）条件，用于过滤由于特殊情况造成的剧烈触摸值抖动：

```
#define HCTouchLib_LIMIT_FACTOR x
```

设置范围（0~255）

0: 不消除毛刺

非 0: 对于设定的触摸有效上限值而言，当 Differ > HCTouchLib_LIMIT_FACTOR \times 对应通道的手指触摸值时判断为毛刺

举例：手指触摸值为 100，该值为 3，则认为 Differ 值大于 300 时为毛刺，此时按键无效

- 设置触摸有效上限值（毛刺消除时间），配合长短按键使用。用于当由于外界条件造成触摸计数值长时间过高时的基线更新，如有硬物置于触摸按键上等：

设置范围：由长短按键生效最长时间决定

当 Differ 连续大于毛刺（HCTouchLib_LIMIT_FACTOR \times 对应通道的手指触摸值）的时长超过了所设的该值后，系统将自动更新当前 Baseline。

毛刺消除更新时长 \approx HCTouchLib_CONFIRM_LONGTOUCH_TIME（SHORTTOUCH_TIME） \times 单个按键扫描时间 \times 开启的检测通道个数

- 设置低基准线复位计数值:

```
#define HCTouchLib_OVERLOW_TIME    x
```

设置范围 (0~255)

0: 禁止用该功能

非 0: 当 Differ 连续低于负噪声阈值 (-HCTouchLib_SETx_N_THRESHOLD) 的时长超过了所设的该值后, 系统自动更新当前 Baseline。

低基准线复位时长 \approx HCTouchLib_OVERLOW_TIME \times 单个按键扫描时间 \times 开启的检测通道个数

- 设置正基准线更新时间:

```
#define HCTouchLib_OVERZERO_TIME    x
```

设置范围 (0~65535)

系统会对在±噪声 (HCTouchLib_SETx_N_THRESHOLD) 之间的 Differ 的时间进行累加, 当累加的和大于正基准线更新时间时, 将对 Baseline 进行微调 (+2 或-2)

3.2.7 软件可调参数配置介绍

本版本中一共新增了几个数据变量, 这几个变量均可在程序运行时进行修改, 适用于一些特殊情况, 用户可以根据自己的实际情况进行调整。这几个变量均在 HC89F30xC _GeneralKey_Touch_Lib.h 文件中进行了申明, 只需要包含了这个头文件即可在程序中修改这几个参数, 分别为长/短按键最长有效时间 (XDATA_CONFIRM_LONGTOUCH_TIME、XDATA_CONFIRM_SHORTTOUCH_TIME)、确认按键有/无效值比例 (XDATA_CONFIRM_TOUCH_RATIO、XDATA_CONFIRM_NOTOUCH_RATIO)、按键确认/离开消抖次数 (XDATA_CONFIRM_TOUCH_TIME、XDATA_CONFIRM_NOTOUCH_TIME)。

- 触摸长按键最长有效时间计数值:

```
extern ux16    XDATA_CONFIRM_LONGTOUCH_TIME;
```

设置范围 (SHORTTOUCH_TIME ~65535)

- 触摸短按键最长有效时间计数值:

```
extern ux16    XDATA_CONFIRM_SHORTTOUCH_TIME;
```

设置范围 (0~2000)

本参数原理与 3.2.6 章节 [《设置基线更新参数》](#) 中的设置触摸长短按键最长有效时间 (HCTouchLib_

CONFIRM_LONGTOUCH_TIME、HCTouchLib_CONFIRM_SHORTTOUCH_TIME)原理相同，HCTouchLib_CONFIRM_LONGTOUCH_TIME 相当于 XDATA_CONFIRM_LONGTOUCH_TIME 这个变量的初始化值。

➤ 确认按键有效值比例：

```
extern ux8    XDATA_CONFIRM_TOUCH_RATIO;
```

设置范围（40~1000）

Differ>手指触摸值*确认按键有效值比例/100，后按键生效。

举例：手指触摸值设置为 100，此值设置为 70，那么当 Differ 大于 70 时按键生效

该值常规情况下无需调整，在遇到一些特定情况下，可以动态调整手指触摸值。

➤ 确认按键无效值比例：

```
extern ux8    XDATA_CONFIRM_NOTOUCH_RATIO;
```

设置范围（20~1000）

Differ<手指触摸值*确认按键无效值比例/100，后按键失效。

举例：手指触摸值设置为 100，此值设置为 30，那么当 Differ 小于 30 时按键无效

该值常规情况下无需调整，在遇到一些特定情况下，可以动态调整手指触摸值。

➤ 按键确认消抖次数：

```
extern ux8    XDATA_CONFIRM_TOUCH_TIME;
```

设置范围（1~255）

按下消抖时间 = 单个按键扫描时间 × 开启的检测通道个数 × 按键确认消抖次数

举例：按键确认消抖次数为 2，手指触摸有效信号应连续时间大于等于 2 时才会认为触摸按键有效。

按键离开消抖次数

```
extern ux8    XDATA_CONFIRM_TOUCH_TIME;
```

设置范围（1~255）

松开消抖时间 = 单个按键扫描时间 × 开启的检测通道个数 × 按键离开消抖次数

举例：按键离开消抖次数为 2，手指触摸无效信号应连续时间大于等于 2 时才会认为触摸按键无效。

3.2.8 RawData 滤波参数配置介绍

参数修改文件: HC89F30xC _GeneralKey_Touch_Auxiliary_H

- 设置 RawData 滤波深度，打开后系统会对 RawData 采用递推中位值平均滤波，即对递推数组进行排序后去除最高值和最低值，然后对剩余数据计算平均值：

```
#define HCTouchLib_FILTER_DEPTH          x
```

0: 关闭滤波功能

3~15: 设置滤波数组长度，如设置 5，则对 RawData 数据进行排序，然后去除最高值和最低值，剩余的 3 个数据计算平均值得到最终值

3.2.9 按键模式参数配置介绍

参数修改文件: HC89F30xC _GeneralKey_Touch_Filter.h

- 设置按键模式选择

```
#define HCTouchLib_MULTIBUTTON          x
```

0: 多按键模式，支持多个按键同时有效

1: 单按键模式（最强按键），仅信号最强（differ 最大）的按键有效

2: 双按键模式（组合按键），同一时间内最多只有两个按键有效

- 设置程序使能的通道，使能最强按键后必须配置本值：

```
#define HCTouchLib_OPENTOUCHCHANNEL    {x,x,x,x,x,x,x,x}
```

举例：如果打开了 TK1，TK2，TK5，TK8，则必须在 {} 中填入 1，2，5，8

注意：1、此处必须严格按照使能的通道号填入，如填入错误会导致 TouchKeyFlag 输出错误
如未使能最强按键，此值无需填写

- 设置双按键组成元素，最多提供两组组合按键，必须配合双按键模式使用：

```
#define HCTouchLib_COMBINATIONKEYA     {x,x}
```

```
#define HCTouchLib_COMBINATIONKEYB     {x,x}
```

举例：如果 TK0、TK1，TK8、TK9 作为两组组合按键，则分别填入 {0，1}，{8、9}

3.3 触摸调试

HCTouch VLAB 是作为芯圣触摸产品的开发调试辅助工具。用户可通过该调试工具，观察触摸实际工作环境，如下图

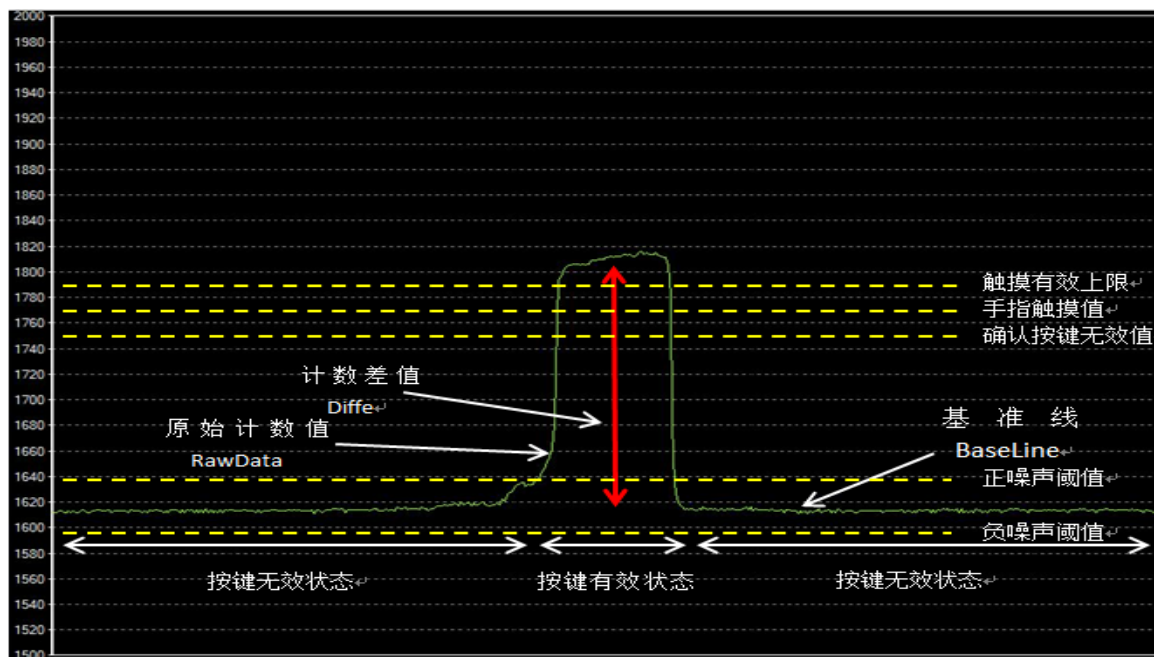


图 3-4 触摸调试窗口

3.3.1 手指触摸值设置

为获得最佳触摸效果，建议采用以下设置阈值方式：

- 手指触摸值：为手指触摸后 Differ（计数差值）的 75%
- 按键确认消抖次数要小于短按键生效最长时间
- 按键离开消抖次数要小于低基准线复位时间
- 确认按键有效值比例要等于确认按键无效值比例
- 其他值：根据设计需要

3.3.2 基准线更新

触摸产品实际工作环境中，由于温度、湿度或外界干扰都会引起触摸计数值的变化。基准线是触摸测量值的参考计数，这对于环境影响补偿而言起着重要作用。手指触摸按键的状态（例如，手指触摸和手指离开状态）是基于基准线建立的参考判断。由于每路触摸通道具有相应的寄生电容，因此所有触摸通道均有各自的基准线。

基准线以基准线更新的对应阈值参数来跟踪计数变化，确保更新速率与预期应用相互匹配。如果更新速率过快，那么，基准线将补偿手指产生的所有变量，并不予检测手指移动的情况。如果更新速率过慢，那么，相对较慢的环境变化可能导致手指出错。开发期间，应验证基准线更新阈值的设置情况。

系统在初始化过程中（CTK_Init()函数）已对开启的各触摸通道设定了对应的触摸基线值。触摸初始化完成后，要求用户在每次扫描完成后，都要在主程序中调用一次函数 Touch_Signal_Handle()，该函数会对触摸基线值进行更新。

3.3.2.1 基准线更新原理

触摸基准线更新方式有五种，如下图流程图说明了基准线更新方式。

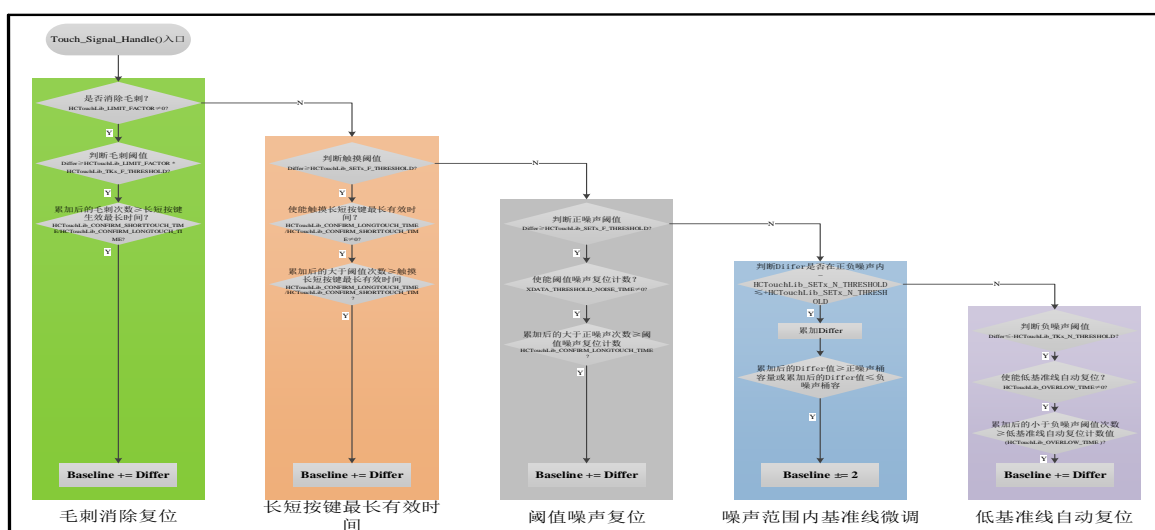


图 3-5 基线更新方式

- **触摸有效上限值复位：**使能该功能（即设置触摸上限值判断条件：HCTouchLib_LIMIT_FACTOR 为非 0；根据长短按键最长有效时间计数：HCTouchLib_CONFIRM_SHORTTOUCH_TIME/ LONGTOUCH_TIME）时，对于已设定的采样数量（HCTouchLib_OVERHIGH_TIME）而言，当触摸 Differ 值大于触摸有效上限值判断条件后，基准线会被更新。

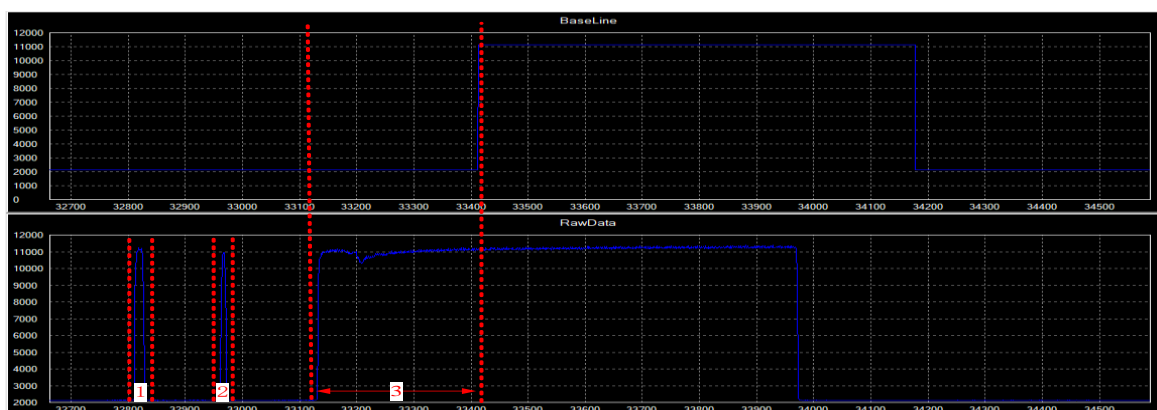


图 3-6 触摸有效上限复位

如上图所示，设置了 HCTouchLib_LIMIT_FACTOR，此时 1 和 2 均为一个偶然性的毛刺，由于这个毛刺时长并未超过长短按键最长有效设置的时间，算法将其忽略。而毛刺 3 则为一个持续时间较长的毛刺，并且毛刺时长超过了长短按键设置的时间，于是 $\text{Baseline} \pm \text{Differ}$ 。

- **触摸长/短按键最长有效时间：**使能该功能（即将 HCTouchLib_CONFIRM_LONGTOUCH_TIME/HCTouchLib_CONFIRM_SHORTTOUCH_TIME 配置为非 0 值）时，配合手指触摸值（HCTouchLib_SETx_F_THRESHOLD）使用。当 Differ 大于手指触摸值的时长超过了所设的该值后，系统将自动更新当前 Baseline。虽然该设置限制了触摸按键有效的最长时间，但可以防止以下情况发生：由于某种干扰或故障，导致触摸计数差值 Differ 突然大于所设的手指触摸值，而手指并未触摸，导致触摸处于异常工作状态。
- **低基准线自动复位时间：**使能该功能（即将 HCTouchLib_OVERLOW_TIME 配置为非 0 时），配合噪声阈值（HCTouchLib_SETx_N_THRESHOLD）使用。当触摸 Differ 值低于负噪声阈值后，基准线会被更新。可以防止以下情况发生：由于某种干扰或故障导致触摸计数差值 Differ 突然小于所设的噪声阈值，而手指并未触摸，导致触摸处于异常工作状态。
- **正基准线更新时间：**使能该功能（即将 HCTouchLib_OVERLOW_TIME 配置为非 0 时），配合噪声阈值（HCTouchLib_SETx_N_THRESHOLD）使用。当触摸 Differ 值高于正噪声阈值后，基准线会被更新。可以防止以下情况发生：由于某种干扰或故障导致触摸计数差值 Differ 突然大于所设的噪声阈值，而手指并未触摸，导致触摸处于异常工作状态。

本章节中涉及的以下几个参数转换时长计算方式：

HCTouchLib_CONFIRM_LONGTOUCH_TIME/HCTouchLib_CONFIRM_SHORTTOUCH_TIME（设置触摸自动复位计数值）

HCTouchLib_OVERLOW_TIME（设置低基准线复位计数值）

其时长的计算方式均相同，如：

触摸自动复位时长 $\approx \text{HCTouchLib_CONFIRM_LONGTOUCH_TIME} \times \text{单个按键扫描时间} \times \text{开启的检测通道数}$ 。

单通道扫描时间 $\approx ((\text{HCTouchLib_DSCR} + 1) \times 256 - 1) / (\text{Fosc} / 2)$

计算举例：

Fosc = 32Mhz

开启通道数 = 6

HCTouchLib_DSCR = 48

HCTouchLib_CONFIRM_LONGTOUCH_TIME = 1000

单通道扫描时间 $\approx ((48+1) * 256 - 1) / 16000000 \approx 0.7839\text{ms}$

触摸自动复位时长 $\approx 0.7839\text{ms} * 6 * 1000 \approx 4.7\text{s}$

注：为了简化计算，HolyChip 提供了一个自动计算的 Excle 表格，放置于各工程下，用户自行试用计算相关时间。

3.3.2.2 抗干扰

对于整个触摸系统而言，遭遇干扰的表现是当前计数值（RawData）突变，这个值可能会突然下降或者突然上升，用户需要通过配置 HCTouchLib_CONFIRM_LONGTOUCH_TIME、HCTouchLib_LIMIT_FACTOR、HCTouchLib_OVERZERO_TIME、HCTouchLib_OVERLOW_TIME 几个值来确保 Baseline 不会再遭遇 RawData 突变时随之更新，导致干扰结束后 Baseline 无法恢复到正常的值，从而使得按键触摸无效。

下图为触摸系统遭遇的外部电磁干扰的图形。

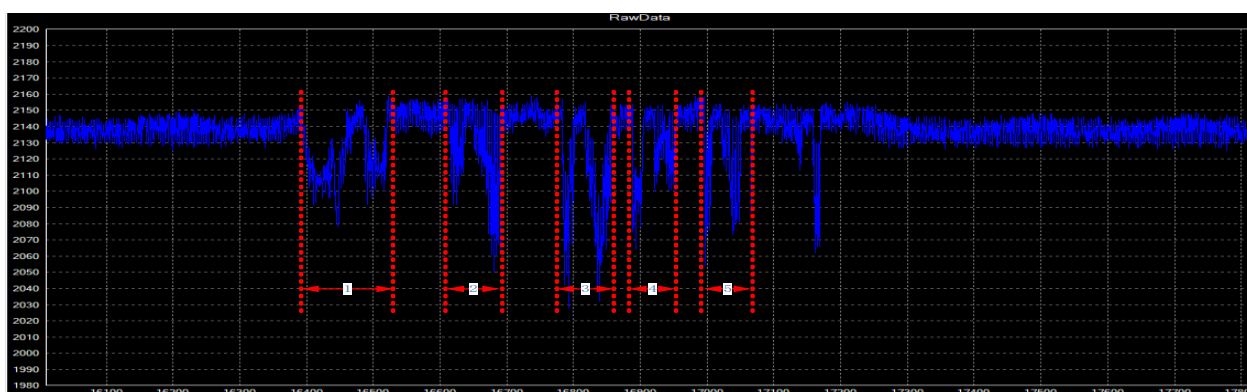


图 3-7 外部电磁干扰

如图所示，整个系统中的干扰区间被大概分成了 5 个，其中 1 号区间的干扰持续时间最长，那么设置 HCTouchLib_OVERLOW_TIME 的时常就需要超过这个区间的时长即可，这样的话就可以确保这种非持续性的干扰导致 Baseline 的降低。

但是设置 HCTouchLib_OVERLOW_TIME 的时常不可以过长，否则遇到用户手指按在按键上上电后恢复正常 Baseline 的时间过长，从而导致用户体验较差。

设置 HCTouchLib_OVERZERO_TIME 的方式则与上面相同。

3.3.3 应用开发

➤ 一轮扫描完成标志位

```
extern bit OneCycleScanDone; //扫描一轮置位一次
```

一轮扫描完成标志位，一轮扫描完成后标志位 OneCycleScanDone 置 1，用户需要在一轮扫描完成后调用 Touch_Signal_Handle()函数，并且将 OneCycleScanDone 清 0。

➤ 按键响应标志位

```
extern ux32 TouchKeyFlag; //按键响应标志，每 1BIT 对应一个通道，BIT 对应位与  
HCTouchLib_TK_CH_MASK 相对应
```

按键响应标志位，每 1BIT 对应一个通道，当对应通道确认按下后，对应的 BIT 会置 1，BIT 对应位与 HCTouchLib_TK_CH_MASK 相对应，具体对应方式可以参考本文 3.2.3 章节 [《设置需要检测的通道》](#)。

3.4 注意事项

- 1、建议在 CTK 初始化之前初始化 ADC，并且需要在 ADC 初始化内配置好需要使用的 ADC 通道，确保不会和触摸通道产生冲突。另外使能完成 ADC_EN 以后建议保持打开，不要关闭。

错误示例：ADC_Init 在 CTK_Init 之前，并且未对 ADCC1 寄存器进行配置（此时 ADC 通道默认 AN0），然后使能 CTK，并且使能了 TK0，由于 AN0 与 TK0 为同一个端口，于是 AN0 和 TK0 同时打开，导致初始化的 RawData 的数据错误。

- 2、在进行 Flash 做类 EEPROM (IAP) 操作进行扇区擦除时，建议先暂停 CTK 扫描 (Scan_Pause 函数)，等到操作完成后再继续进行 CTK 扫描 (Scan_Continue 函数)。

4 版本说明

版本	日期	描述
V1.00	2020/07/20	初版
V1.01	2021/11/05	文档规范化

HOLYCHIP公司保留对以下所有产品在可靠性、功能和设计方面的改进作进一步说明的权利。

HOLYCHIP不承担由本手册所涉及的产品或电路的运用和使用所引起的任何责任，HOLYCHIP的产品不是专门设计来应用于外科植入、生命维持和任何HOLYCHIP产品产生的故障会对个体造成伤害甚至死亡的领域。如果将HOLYCHIP的产品用于上述领域，即使这些是由HOLYCHIP在产品设计和制造上的疏忽引起的，用户应赔偿所有费用、损失、合理的人身伤害或死亡所直接或间接所产生的律师费用，并且用户保证HOLYCHIP及其雇员、子公司、分支机构和销售商与上述事宜无关。

芯圣电子

2021 年 11 月