凌思微电子（厦门）有限公司

LINKEDSEMI

凌思微 TK 模块应用笔记

修订记录

版本	修订日期	修订说明	作者
V1.0	2022-1-21	初始版本	
V1.1	2022-3-23	修改初始化的灵敏度	

目录

第1章 概述

电容式触摸按键满足用户在复杂应用中对稳定性、灵敏度、功耗、响应速度、防水、带水操作、抗震动、抗电磁干扰等方面的高体验要求。开发过程非常简单，最大限度的降低了方案成本。适用于遥控器、灯具调光、各类开关以及车载、小家电和家用电器控制界面等应用中。

TK 模块具备电容式触摸按键检测功能，当按键有触摸时，可以被 TK 模块检测到。同时，TK 模块还可以选择不同的工作模式，降低在空闲时的运行功耗。

本应用笔记将介绍触摸按键模块（TK 模块）的初始化配置，可被调用的功能以及调试方法。

第2章 初始化结构体参数介绍

2. 1 初始化结构体

该结构体可用于初始化 TK 模块设置，具体参数介绍在后续章节中介绍。

```
typedef struct
{
  volatile uint8_t NPRR;
  volatile uint8_t DNPR;
  volatile uint8_t MultiOn;
  volatile uint8_t Debounce;
  volatile uint16_t ChannelEN;
  volatile uint8_t GbSen;
  volatile uint8_t LP;
  volatile uint8_t IComGS;
  volatile uint8_t CdcFswDiv;
  volatile uint8_t LowPowerScanPeriod;
  volatile uint8_t KeyDebounceCount;
  volatile uint8_t LpDelay;
  volatile uint8_t ScanPeriod;
  volatile uint16_t GsMask;
  volatile uint8_t LSenseGS;
} TK_InitTypeDef;

typedef struct __TK_HandleTypeDef
{
      TK_InitTypeDef Init; /*!< TK required parameters */
      HAL_LockTypeDef Lock; /*!< TK locking object */
      volatile uint32_t ErrorCode; /*!< TK Error code */
} TK_HandleTypeDef;
```

2. 2 基础参数介绍

1. 噪声门限比例倍数设置参数（NPRR）
   该参数可以设置范围从 0 到 7，对应 4 倍，6 倍，8 倍，10 倍，12 倍，14 倍，16 倍，18 倍；
   当噪声保护使能时，如果检测信号超出触摸信号阈值的 NPRR 倍， 则忽略此次检测结果。
   默认配置示例： tkHandle.Init.NPRR = 0;

2. 噪声保护使能参数（DNPR）
   该参数设置为 TK_DNPR_ENABLE 时，开启噪声保护。如果检测信号超出触摸信号阈值的 NPRR 倍， 则忽略此次检测结果。
   该参数设置为 TK_DNPR_DISABLE 时，关闭噪声保护。任何超出触摸信号阈值的检测 结果都会被采用。
   默认配置示例：tkHandle.Init.DNPR = TK_DNPR_DISABLE;

3. 多感应通道同时感应使能参数（MultiOn）
   该参数配置为 TK_MULTION_ENABLE 时，开启多个感应通道同时输出触摸结果。
   该参数配置为 TK_MULTION_DISABLE 时，关闭多个感应通道同时输出触摸结果。
   默认配置示例：tkHandle.Init.MultiOn = TK_MULTION_DISABLE;

4. 防抖动参数（Debounce）
   该参数可配置范围从 0 到 255。实际应用中，可能出现由于误触摸或者随机干扰造成的按键误触发情况，软件增加按键防抖动参数可以过滤掉这些干扰，保证按键的稳定性。如果按键灵敏度过低可适当减小防抖动参数，如果按键灵敏度过高可适当增大防抖动参数，默认配置为 3。
   默认配置示例：tkHandle.Init.Debounce = 3;

5. CDC 模块扫描频率参数（CdcFswDiv）
   该参数可配置范围从 0 到 255。配置该参数可改变按键寄生电容的充放电次数，默认配置为 0x07。
   默认配置示例： tkHandle.Init.CdcFswDiv = 0x07;

6. 消抖拍数（KeyDebounceCount）
   该参数设置范围从 1 到 3，定义按键按下和抬起时的消抖拍数。配置该参数为 1 时，连续两拍按下 才会识别为按键按下，连续两拍没有触摸才会识别为按键抬起。
   默认配置示例：tkHandle.Init.KeyDebounceCount = 1;

7. 扫描周期参数（ScanPeriod）
   该参数设置范围从 1 到 3，定义扫描周期（ScanPeriod*16ms）。配置该参数可改变 TK 模块正常工作时的扫描周期，配置该参数为 1 时，TK 模块在正常工作的扫描周期为 16ms。
   默认配置示例：tkHandle.Init.ScanPeriod = 1;

2. 3 常用功能参数介绍

1. 电容按键通道使能参数（ChannelEN）

   该参数可设置范围从 0x0000 到 0x3FFF，用于控制电容通道使能，共 13 个通道可设置，对应该参数的 13 位，如图 1，配置该参数为 0x3FFF 时，将使能 13 个按键通道。

   默认配置示例：tkHandle.Init.ChannelEN = 0x3fff;

电容按键通道	12	11	10	9	8	7	6	5	4	3	2	1	0
电容按键参数设置为0x3FFF时	1	1	1	1	1	1	1	1	1	1	1	1	1

图 1 按键通道使能

2. 全局灵敏度参数（GbSen）

   该参数可设置范围从 0 到 255，最高为 0，可实现一次性设置 13 个通道的灵敏度。配置该参数为 0xA0 时，13 个通道的灵敏度都为 0xA0。但过高或过低的灵敏度都可能导致触摸检测失败。灵敏度过低时用户触摸无响应或需大力 增大触摸面积才有响应，灵敏度过高时用户轻触或接近时有响应，但正常触摸时反而没响应，这是因为误触发了噪声保护功能。触摸灵敏度的更改必须先关闭扫描后再设置，新的灵敏度数据将在重新使能扫描 100ms 后生效。

   默认配置示例：tkHandle.Init.GbSen = 0XA0;

2. 4 低功耗模式设置参数介绍

当需要使用低功耗扫描模式时，可按照要求配置以下参数，其他时候这些参数配置为默认值。

1. 低功耗扫描使能参数（LP）
   该参数控制是否自动进入低功耗扫描模式。
   设置为 TK_LP_ENABLE 时，开启自动进入低功耗扫描功能，当一段时间（这个时间由 lp_delay 设置）没有触摸按键将自动进入低功耗扫描。
   设置为 TK_LP_DISABLE 时，关闭自动进入低功耗扫描功能。
   默认关闭。
   默认配置示例：tkHandle.Init.LP = TK_LP_DISABLE;
2. 群扫描补偿电流参数（IComGS）
   该参数可设置范围从 0 到 127，该参数直接作用于内部 CSD 模块群扫描补偿电流参数，该值配置越大，群扫描信号越大，越容易低功耗唤醒；但要避免 因为过大而无法进入低功耗。
   默认设置为 40。
   默认配置示例：tkHandle.Init.IComGS = 0x28;
3. 低功耗扫描下的扫描周期参数（LowPowerScanPeriod）
   该参数控制进入低功耗扫描模式时的扫描周期。
   该参数设置为 TK_LPSP_500 时，低功耗扫描模式下的扫描周期为 500ms；
   该参数设置为 TK_LPSP_250 时，低功耗扫描模式下的扫描周期为 250ms；
   该参数设置为 TK_LPSP_100 时，低功耗扫描模式下的扫描周期为 100ms。
   默认设置为 TK_LPSP_500。
   默认配置示例：tkHandle.Init.LowPowerScanPeriod = TK_LPSP_500;
4. 群扫描灵敏度参数（LSenseGS）
   该参数设置可范围从 0 到 255，默认为 0（不配置），直接使用全局灵敏度参数所配置值；当配置为非 0 时，则使用所配置值。具体含义参照附录 TK_GSR 含义对照表。通过配置该参数，可优化低功耗唤醒。
   普通扫描是指一次开启一个通道，通过内部 CSD 模块扫描得到这个通道的信号值。群扫描主要相对于普通扫描而言，是指一次同时开启所有配置通道，通过内部 CSD 模块扫描得到这些通道的整体信号值，触摸其中任一通道，整体信号值均会有所体现，但是，整体信号值并非所有配置通道普通扫描信号的直接简单累加。群扫描是软件低功耗的实现方式。
   默认配置示例：tkHandle.Init.LSenseGS = 0x68;
5. 按键通道参与低功耗扫描使能参数（GsMask）
   该参数可设置范围从 0x0000 到 0x3FFF，用于控制电容通道参与低功耗扫描使能，共 13 个通道可设置，对应该参数的 13 位，如图 2，配置该参数为 0x3FFF 时，将使能 13 个按键通道参与低功耗扫描。
   默认设置为 0x00，不参与低功耗扫描。
   默认配置示例：tkHandle.Init.GsMask = 0x00;

电容按键通道	12	11	10	9	8	7	6	5	4	3	2	1	0
GsMask设置为0x3FFF时	1	1	1	1	1	1	1	1	1	1	1	1	1

图 2，按键通道是否参与低功耗扫描

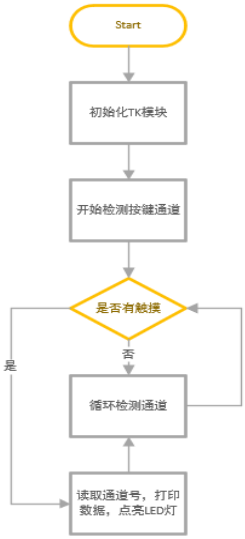
注：当 TK 模块进入低功耗扫描模式时，TK 模块将无法被修改配置，如：无法开启扫描，无法开启睡眠模式等。直按键有被触摸，TK 模块才会被唤醒，推出低功耗扫描模式。

2. 5 初始化结构体调用示例（默认正常工作）

```
//TK 结构体声明
static TK_HandleTypeDef tkHandle;
//设置结构体参数（设置参数为默认值）
tkHandle.Init.NPRR = 0;                  // 设置噪声阈值比例倍数为 4
tkHandle.Init.DNPR = TK_DNPR_DISABLE;// 关闭噪声保护
tkHandle.Init.MultiOn = TK_MULTION_DISABLE;//关闭多通道同时感应
tkHandle.Init.Debounce = 3;              //设置防抖参数为 3
tkHandle.Init.ChannelEN = 0x3FFF;     //使能按键通道
tkHandle.Init.GbSen = 0xA0; //设置全局灵敏度为合适的值（可通过调试得到，调试方法在后续章节介绍）
tkHandle.Init.LP = TK_LP_ DISABLE;    //关闭自动进入低功耗扫描模式
tkHandle.Init.IComGS = 0x28;           //设置群扫描补偿电流为 40
tkHandle.Init.CdcFswDiv = 0x07;        //设置 CDC 模块扫描频率为 0x07
tkHandle.Init.LowPowerScanPeriod = TK_LPSP_500;//设置低功耗扫描周期为 500ms
tkHandle.Init.KeyDebounceCount = 1; //设置消抖拍数为 1
tkHandle.Init.LpDelay = 4;             //默认值，4S 无按键变化进入低功耗扫描
tkHandle.Init.ScanPeriod = 1;          //设置扫描周期为默认值 1，既 1*16ms
tkHandle.Init.GsMask = 0x00;         //禁止所有通道参与低功耗扫描
tkHandle.Init.LSenseGS = 0x00;       //设置群扫描灵敏度为 0
//调用 TK 模块初始化函数
if (HAL_TK_Init(&tkHandle) != HAL_OK)
{
      Error_TKHandle();
}
```

初始化结果：

初始化配置完成后，TK 模块进入正常扫描模式，开始对触摸按键扫描，检测是否由触摸发生。正常工作流程图：



2. 6 初始化结构体调用示例（开启低功耗扫描工作）

```
//TK 结构体声明
static TK_HandleTypeDef tkHandle;
//设置结构体参数（根据 1.4 章节设置相应参数，其他保持默认值）
tkHandle.Init.NPRR = 0;                  // 设置噪声阈值比例倍数为 4
tkHandle.Init.DNPR = TK_DNPR_DISABLE;// 关闭噪声保护
tkHandle.Init.MultiOn = TK_MULTION_DISABLE;//关闭多通道同时感应
tkHandle.Init.Debounce = 3;              //设置防抖参数为 3
tkHandle.Init.ChannelEN = 0x3FFF;    //使能按键通道
tkHandle.Init.GbSen = 0xA0;              //设置全局灵敏度为合适的值（可通过调试得到，调试
方法在后续章节介绍）
tkHandle.Init.LP = TK_LP_ ENABLE;    //使能自动进入低功耗扫描模式
tkHandle.Init.IComGS = 0x28;             //设置群扫描补偿电流为 40
tkHandle.Init.CdcFswDiv = 0x07;          //设置 CDC 模块扫描频率为 0x07
tkHandle.Init.LowPowerScanPeriod = TK_LPSP_500;//设置低功耗扫描周期为 500ms
kHandle.Init.KeyDebounceCount = 1; //设置消抖拍数为 1
tkHandle.Init.LpDelay = 4;               //设置默认无触摸 4s 进入低功耗扫描模式
tkHandle.Init.ScanPeriod = 1;            //设置扫描周期为默认值 1，既 1*16ms
tkHandle.Init.GsMask = 0x3FFF;           //开启所有通道参与低功耗扫描模式
tkHandle.Init.LSenseGS = 0x68;           //设置群扫描灵敏度为合适的值（可通过调试得到，调
试方法在后续章节介绍）
//调用 TK 模块初始化函数
if (HAL_TK_Init(&tkHandle) != HAL_OK)
{
      Error_TKHandle();
}
```

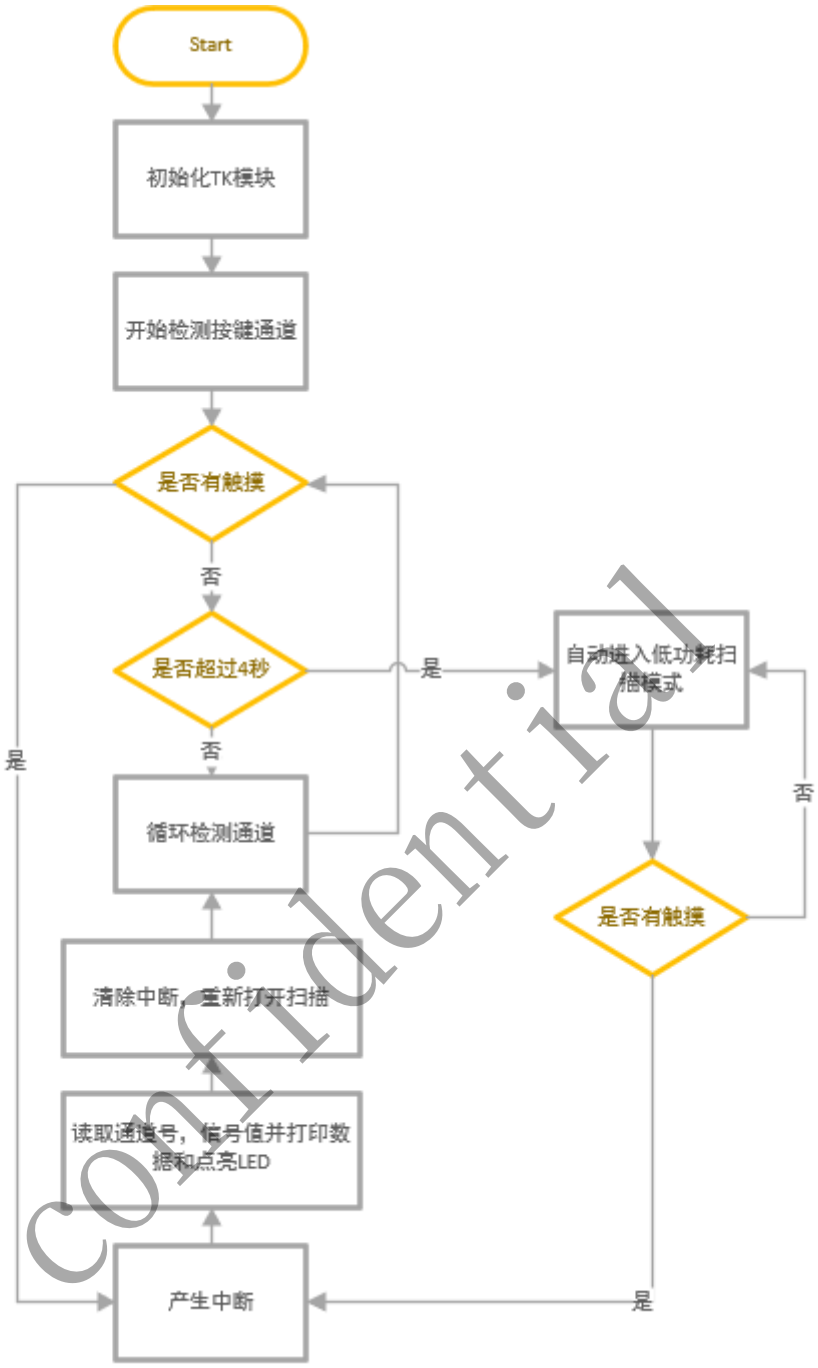
初始化结果：

初始化完成后，TK 模块进入正常扫描模式，开始对触摸按键扫描，每 16ms（初始化设置）扫描一次按键，检测是否有触摸。

由于开启了自动进入低功耗模式功能，当触摸按键 4S（初始化设置）没有被触摸时，TK 模块进入低功耗扫描模式，降低运行功耗。此时，TK 模块每 500ms（初始化设置）扫描一次触摸按键，如果检测到触摸，TK 模块会被唤醒，重新进入正常扫描模式。

注：当 TK 模块进入低功耗扫描模式时，TK 模块的参数将无法通过寄存器参数设置函数更改，只有 TK 模块正常扫描时才可以使用寄存器参数设置函数。

流程图：



第3章 TK 模块的功能函数介绍

3. 1 初始化函数

1. 函数名称：HAL_TK_ Init();
2. 参数：
   TK_HandleTypeDef *htk：htk 指向包含 TK_InitTypeDef 结构体的指针
3. 返回值：HAL status
4. 函数使用方法：配置好初始化结构体参数后，调用该函数初始化 TK 模块，如下：

```
if (HAL_TK_Init(htk) != HAL_OK)
{
    Error_TKHandle();
}
```

3. 2 反初始化函数

3. 函数名称：HAL_TK_ DeInit();
4. 参数：
   TK_HandleTypeDef *htk：htk 指向包含 TK_InitTypeDef 结构体的指针
3. 返回值：HAL status
4. 函数使用方法：调用该函数实现反初始化 TK 模块，如下：

```
if (HAL_TK_ DeInit(htk) != HAL_OK)
{
    Error_TKHandle();
}
```

3. 3 关闭写保护函数

5. 函数名称：HAL_TK_ Disable_WriteProtection();
6. 参数：
   TK_HandleTypeDef *htk：htk 指向包含 TK_InitTypeDef 结构体的指针
3. 返回值：HAL status
4. 函数使用方法：调用该函数关闭 TK 模块写保护功能，只有关闭写保护功能，才能设置 TK 模块参数，如任何两次写操作间隔超过 0.15s，写保护自动生效，如下：

```
while (HAL_TK_ Disable_ WriteProtection(htk) != HAL_OK)
{
    return HAL_ERROR;
}
```

3. 4 寄存器数据读取函数

1. 函数名称： HAL_TK_Get_Reg
2. 参数：
   - uint8_t regAddr：寄存器地址
   - uint8_t *data： 存放数据缓存区
   - uint8_t size： 数据大小
3. 返回值：HAL status
4. 函数使用方法：通过填写要读取的寄存器，数据存放缓存区，数据大小进行读取，如下：

```
HAL_TK_Get_Reg(CNTLR, (uint8_t *)RxBUFF, 2);//读取通道检测信息
```

3. 5 寄存器参数设置函数

1. 函数名称：HAL_TK_Set_Reg
2. 参数：
   - uint8_t regAddr：寄存器地址
   - uint8_t *param： 存放数据缓存区
   - uint8_t size： 数据大小
3. 返回值：HAL status
4. 函数使用方法：通过填写要写的寄存器，存放数据的缓存区，数据大小进行写入，如下：

```
TK_SetReg(WPR, TK_DISABLE_WP, 1);//关闭写保护功能
```

（注：使用寄存器参数设置函数之前，必须关闭 TK 模块写保护功能）

3. 6 开启扫描函数

7. 函数名称：HAL_TK_ScanEnable();
8. 参数：
   TK_HandleTypeDef *htk：htk 指向包含 TK_InitTypeDef 结构体的指针
3. 返回值：HAL status
4. 函数使用方法：调用该函数使能 TK 模块扫描功能，如下：

```
while (HAL_TK_ScanEnable(htk) != HAL_OK)
 {
      return HAL_ERROR;
 }
```

3. 7 关闭扫描函数

1. 函数名称： HAL_TK_ScanDisable
2. 参数：
   TK_HandleTypeDef *htk：htk 指向包含 TK_InitTypeDef 结构体的指针
3. 返回值：HAL status
9. 函数使用方法：调用该函数关闭 TK 模块扫描功能，如下：

```
while (HAL_TK_ScanDisable(htk) != HAL_OK)
 {
      return HAL_ERROR;
 }
```

3. 8 单通道灵敏度设置函数

1. 函数名称： HAL_TK_Set_Lsense
2. 参数：
   TK_HandleTypeDef *htk： htk 指向包含 TK_InitTypeDef 结构体的指针
   uint8_t ChannelNum： 电容按键通道号
   uint8_t Lsense： 灵敏度大小
3. 返回值：HAL status
4. 函数使用方法：将要配置的通道填入 ChannelNum，灵敏度填入 Lsense，如下：
   HAL_TK_Set_Lsense(&tkHandle,0,0xDA);//将通道 0 的灵敏度设置为 0x68

3. 9 开启睡眠函数

1. 函数名称：
   HAL_StatusTypeDef HAL_TK_Enable_DP(TK_HandleTypeDef *htk, , uint8_t _IComGS);
2. 参数：
   TK_HandleTypeDef *htk： htk 指向包含 TK_InitTypeDef 结构体的指针
   uint8_t _WTR： 唤醒阈值
   uint8_t Lsense： 群扫描补偿电流
3. 返回值：HAL status
4. 函数使用方法：将唤醒阈值填入_WTR，群扫描补偿电流填入_IComGS，如下：
   HAL_TK_Enable_DP(&tkHandle, 0x2d,0x28);//使能后立即进入睡眠模式，模块所有功能禁用，唤醒需要调用唤醒函数。
   注：唤醒阈值可设置范围从 0 到 255，默认为 0xFF，越小越不灵敏；
   TK 模块有两种唤醒机制：
   （1） 超低功耗唤醒，当 wake_th 不为 0xff 时进入此模式，此时功耗在 10uA 以下，阈值范围从 30 到 60，值越小越灵敏。（即，睡眠模式）
   （2） 软件轮询唤醒，wake_th 为 0xff 时进入此模式，即低功耗扫描模式。

当 LPSP=0 时，10min 平均功耗约 6uA。

当 LPSP=1 时，10min 平均功耗约 8uA。 注：具体功耗大小与硬件设计、参数配置等因素相关。

注：当 TK 模块进入睡眠模式时，TK 模块将无法进行除了唤醒以外的任何操作，如：无法读数据，无法修改配置，直到调用睡眠唤醒函数将 TK 模块唤醒，并且唤醒后需要重新配置 TK 模块。

3. 10 唤醒睡眠函数

1. 函数名称：HAL_TK_Wake_DP
2. 参数：无
3. 返回值：HAL status
4. 函数使用方法：调用该函数即可唤醒 TK 模块，如下：

```
HAL_TK_Wake_DP();//唤醒是 TK 模块
lsTK_Init();  //唤醒后需要重新配置 TK 模块
```

3. 11 中断使能函数

1. 函数名称：HAL_TK_Enable_IT
2. 参数：

   TK_HandleTypeDef *htk：htk 指向包含 TK_InitTypeDef 结构体的指针
3. 返回值：HAL status
4. 函数使用方法：调用该函数即可开启按键响应中断，当检测到按下或松开时，中断产生响应，如下：

```
HAL_TK_Enable_IT(&tkHandle);
```

3. 12 中断禁止函数

1. 函数名称：HAL_TK_ DisEnable_ IT
2. 参数：

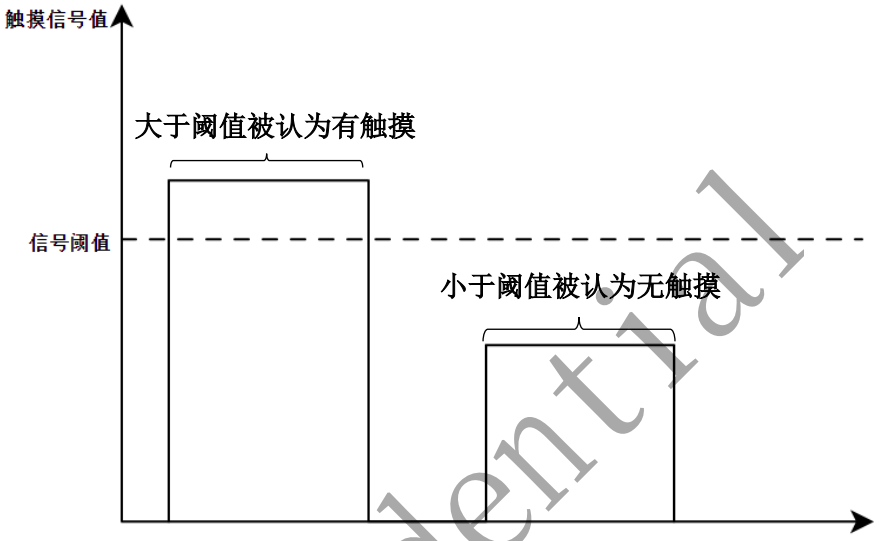
   TK_HandleTypeDef *htk：htk 指向包含 TK_InitTypeDef 结构体的指针
3. 返回值：HAL status
4. 函数使用方法：调用该函数即可关闭按键响应中断，当检测到按下或松开时，中断不会响应，如下：

```
HAL_TK_DisEnable_IT(&tkHandle);
```

第4章 TK 模块调试

4. 1 TK 调试说明

电容触摸按键被触摸时，TK 模块会产生相应按键通道的触摸信号值，如果触摸信号值大于信号阈值，则 TK 模块将判断该触摸按键检测到触摸。我们通过设置通道的灵敏度，可以改变触摸时的触摸信号值大小，如果灵敏度设置太低，触摸信号值太小，按键可能无法检测到触摸；如果灵敏度设置太高，触摸信号值太大，按键可能会被误触。所以，需要通过调试，得到适合的灵敏度大小。本章将介绍 TK 模块不同灵敏度的调试方法。



4. 2 关于灵敏度寄存器详细说明

4. 2. 1 全局灵敏度寄存器 TK_GSR

这是一个关于灵敏度的综合寄存器，有以下多层含义（表格在附录）：

a) 全局灵敏度寄存器 TK_GSR 取值范围为：0~255。

b) TK_GSR 每 32 对应一档，即，TK_GSR 除以 32，取整换算得到扫描精度 N，余查表得到信号阈值 FingerTH。

c) 扫描精度 N = 16-(TK_GSR /32)，即，16~9。扫描精度直接作用于 TK 模块内部 CSD 模块扫描窗口参数 CSD_WIN，对应关系为：CSD_WIN=win_tab[N-9]，其中，win_tab[] = {2,4,8,16,32,64,128,252}。

d) 信号阈值 FingerTH = FingerTHTbl[TK_GSR%32]，其中，FingerTHTbl[32] = {90,94,98,102,106,110,114,118,120,124,128,132,136,140,144,148,152,156,160,164,168, 172,176,180, 184,188,192,196,200,204,208,212}。信号阈值 FingerTH 为信号是否达到有效上报键值的判断阈值；

e) 噪声阈值 NoiseTH 由扫描精度 N 与信号阈值 FingerTH 换算而来：NoiseTH = (FingerTH/2)*N/10，在低功耗扫描中，噪声阈值 NoiseTH 为信号是否达到有效触摸 唤醒的判断阈值。

注：TK_GSR 越小，N 越大，CSD_WIN 越大，扫描信号越大，灵敏度越高

4. 2. 2 单个通道灵敏度寄存器 TK_LSRx

单个通道灵敏度 TK_LSRx 只作用于单个所配置通道，默认为 0（不配置），直接使用全局灵敏度寄存器 TK_GSR 配置值；当配置为非 0 时，则使用所配置值。具体含义参照 TK_GSR 含义对照表

4. 2. 3 群扫描灵敏度寄存器 TK_LSGR

普通扫描是指一次开启一个通道，通过内部 CSD 模块扫描得到这个通道的信号值。群扫描主要相对于普通扫描而言，是指一次同时开启所有配置通道（具体由 TK_CHELR、TK_CHEHR、TK_GMR0、 TK_GMR1 寄存器配置），通过内部 CSD 模块扫描得到这些通道的整体信号值，触摸其中任一通道，整体信号值均会有所体现，但是，整体信号值并非所有配置通道普通扫描信号的直接简单累加。群扫描是软件低功耗的实现方式。 群扫描灵敏度由群扫描灵敏度寄存器 TK_LSGR 配置，默认为 0（不配置），直接使用全局灵敏度寄存器 TK_GSR 所配置值；当配置为非 0 时，则使用所配置值。具体含义参照 TK_GSR 含义对照表。通过配置该寄存器，可优化低功耗唤醒。

4. 2. 4 群扫描补偿电流寄存器 TK_ICGSR

该寄存器直接作用于内部 CSD 模块群扫描补偿电流参数 CSD_RXO_1MS，取值范围为 0~127。默认为 40。一般，该值配置越大，群扫描信号越大，越容易低功耗唤醒；但要避免 因为过大而无法进入低功耗。

4. 2. 5 CSD 模块扫描频率分频寄存器 TK_CFDR

该寄存器直接作用于内部 CSD 模块时钟参数 CSD_CKSEL，默认值为 0x07，一般不需要配置。有时，为了规避扫描干扰，可以尝试配置该寄存器。

4. 3 三种功耗下的灵敏度调试

4. 3. 1 正常扫描

正常扫描即不进入低功耗而进行正常功耗扫描，主要需要配置检测通道、全局灵敏度等参数，其它参数一般选择默认即可。 其中，重点是配置全局灵敏度寄存器 TK_GSR， 必要时配置单个通道灵敏度寄存器 TK_LSRx。用铜柱（9mm）触摸，读取各通道触摸信号值，应大于相应信号阈值，并确保每个通道触摸均有响应键值上报。（注：需要带面板触摸，或覆盖同面板厚度覆盖物）

在正常扫描下，主控可以采用轮询模式来获取键值，或接收 IRQ 的信号，再来读取键值。由于低功耗模式下无法使用 TK 模块，如果主控采用轮询模式，则不可以使能低功耗模式。

1. 初始化设置（除了以下参数之外，其他参数都设置为默认值）

```
tkHandle.Init.ChannelEN = 0x3fff;                    //使能 13 个按键通道
tkHandle.Init.GbSen = 0x68;                          //设置全局灵敏度（待调试）
```

2. 数据读取

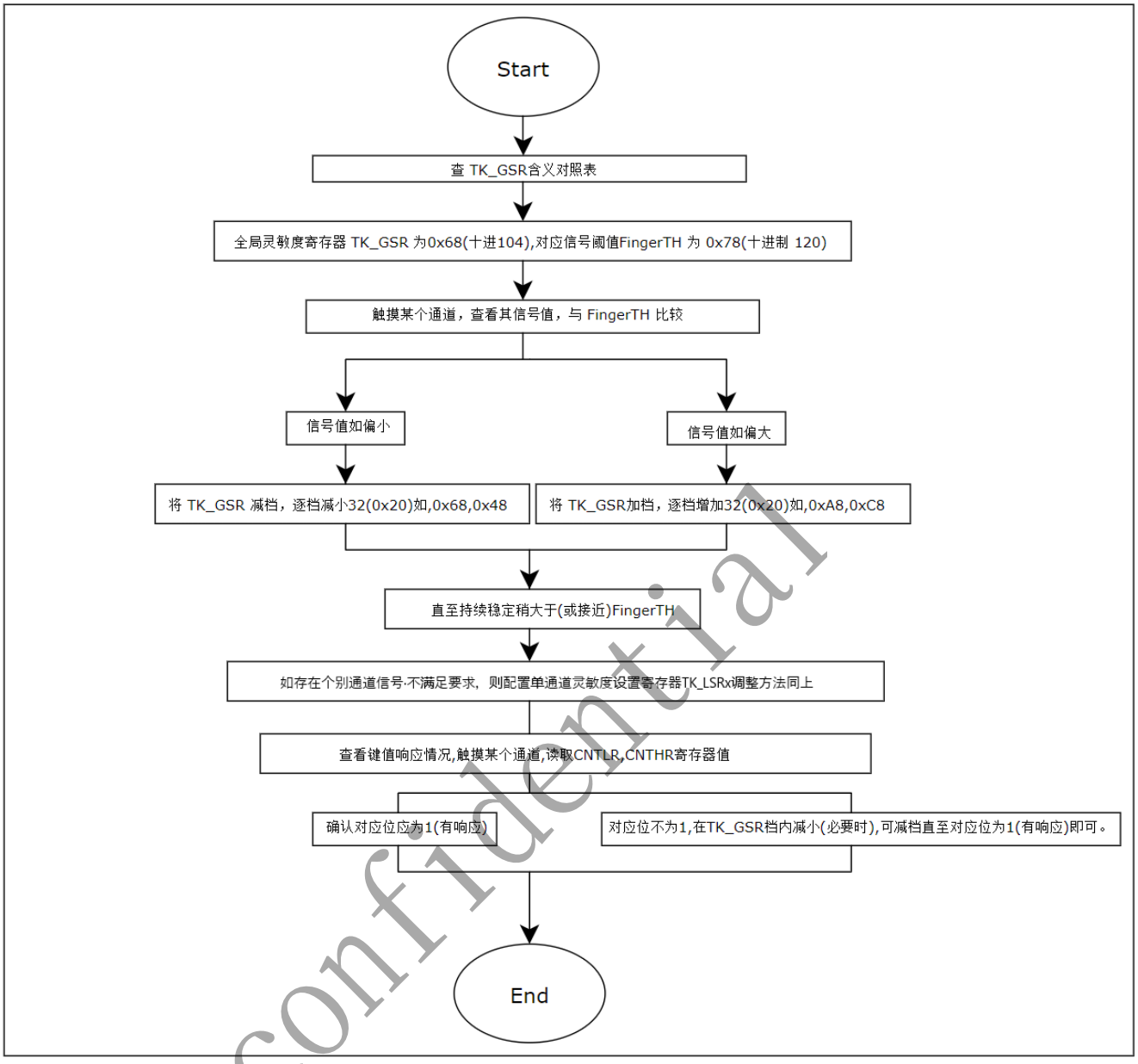
a)从寄存器 TK_CNTLR 和 TK_CNTHR 读取按键和滑条状态，如下：

```
HAL_TK_Get_Reg((uint8_t *)&tkHandle.Instance->CNTLR, RxBUFF, 2);
```

b)从寄存器 TK_SIGLRx 和 TK_SIGHRx 读取信号值，如下：

```
HAL_TK_Get_Reg((uint8_t *)&tkHandle.Instance->SIGLR1, RxBUFF, 1);
HAL_TK_Get_Reg((uint8_t *)&tkHandle.Instance->SIGHR1, RxBUFF + 1, 1);
```

3. 调试步骤



4. 3. 2 低功耗扫描

在正常扫描参数基础上，增配低功耗扫描通道，并对群扫描相关寄存器予以配置调试。软件低功耗是采用群扫描方式来实现的。其中，重点是配置群扫描补偿电流寄存器 TK_ICGSR、以及群扫描灵敏度寄存器 TK_LSGR。各通道的群扫描信号值，应大于群扫描噪声阈值，确保可以进入低功耗，也可唤醒。

1. 初始化参数设置（除了一下参数之外，其他参数都设置为默认值）

```
tkHandle.Init.ChannelEN = 0x3fff;              //使能 13 个按键通道
tkHandle.Init.GbSen = 0xA0;                    //设置全局灵敏度（根据调试确定）
tkHandle.Init.LP = TK_LP_ENABLE;               //开启自动进入低功耗扫描
tkHandle.Init.IComGS = 0x10;                   //配置群扫描补偿电流（待调试）
tkHandle.Init.LpDelay = 4;                  //设置默认无触摸 4s 进入低功耗扫描模式
tkHandle.Init.LowPowerScanPeriod = TK_LPSP_500;//低功耗扫描周期 500ms
```

```
tkHandle.Init.GsMask = 0x3FFF;                    //配置检测通道参与低功耗扫描
tkHandle.Init.LSenseGS = 0x68;                    //配置群扫描灵敏度（待调试）
```

2. 数据读取

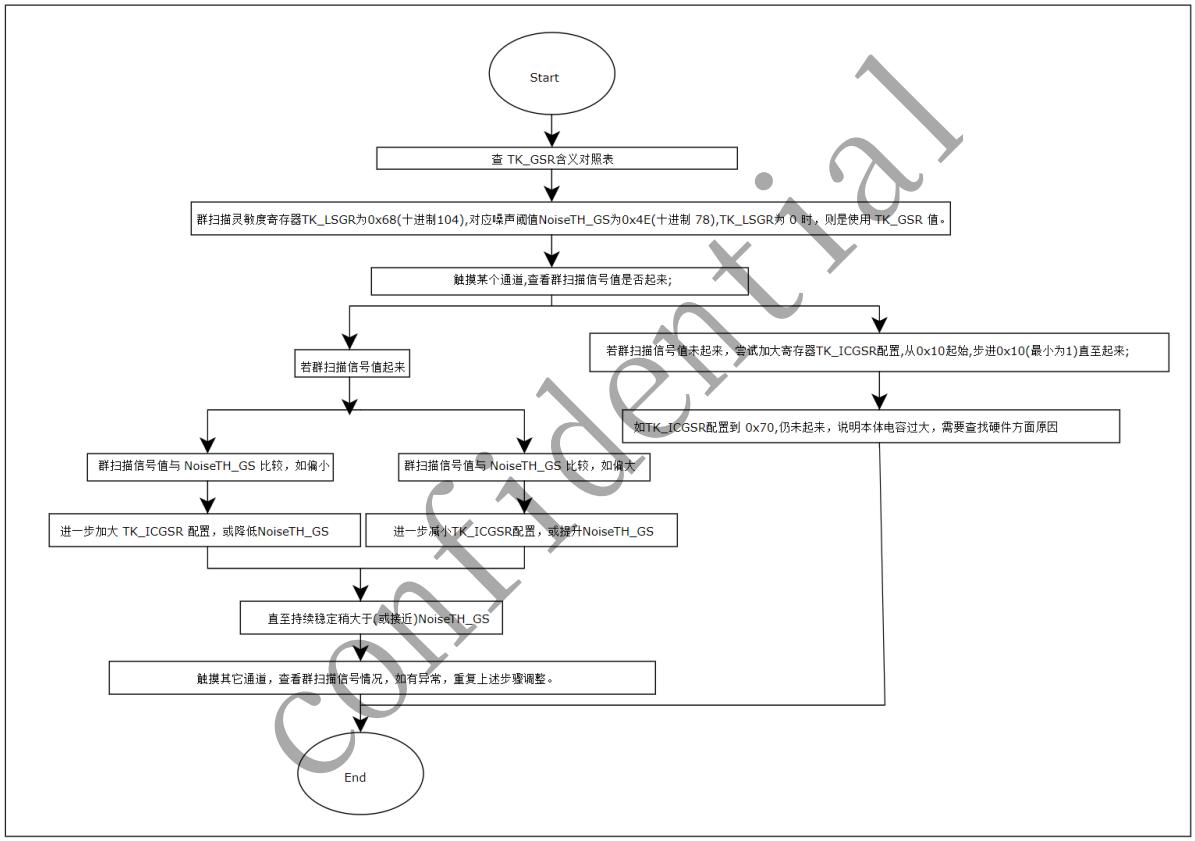
a)从寄存器 TK_CNTLR 和 TK_CNTHR 读取按键和滑条状态，如下：

```
HAL_TK_Get_Reg((uint8_t *)&tkHandle.Instance->CNTLR, RxBUFF, 2);
```

b)从寄存器 TK_SIGSLRx 和 TK_SIGSHRx 读取对应通道的群扫描信号值，如下所示为读取通道 0 的群扫描信号值：

```
HAL_TK_Get_Reg((uint8_t *)&tkHandle.Instance->SIGLSR0, RxBUFF, 1);
HAL_TK_Get_Reg((uint8_t *)&tkHandle.Instance->SIGSHR0, RxBUFF + 1, 1);
```

3. 调试模式步骤：



注：成功进入软件低功耗后的电流会在 20uA 以内跳动

4. 3. 3 睡眠模式

当低功耗扫描的调节不起作用，即群扫描信号一直无法超过对应的阈值，则可尝试使用睡眠模式降低功耗。常用的寄存器有：触摸阈值 TK_WTR，补偿电流 TK_ICGSR。

1. 初始化参数设置，与正常扫描设置一致

```
tkHandle.Init.ChannelEN = 0x3fff;                 //使能 13 个按键通道
tkHandle.Init.GbSen = 0xDA;                       //设置全局灵敏度（调试好的灵敏度）
```

2. 调用睡眠使能函数

调用该函数后，TK 模块将立即进入睡眠模式

```
HAL_StatusTypeDef HAL_TK_Enable_DP(TK_HandleTypeDef *htk, uint8_t _WTR, uint8_t
_IComGS);
```

3. 调试模式步骤：

a) TK_ICGSR（_IComGS），一般取 5~120，默认为 40，值越大越容易退出低功耗。

b) TK_WTR（_WTR），一般取 30~60，默认为 0xFF。当为 0xff 时，程序会进入软件低功耗，使用硬件低功耗时必须 不为 0xff。一般取 45,值越大越容易退出低功耗。 由于进入硬件低功耗模式后，iic 无法通信，这时可以对地短接 IRQ 脚来强制退出低功耗，进而读取一些信号数据。

注：成功进入硬件低功耗后的电流会保持在 10uA 以内。

1、 TK_WTR,值越小越不容易唤醒。建议在范围内，按照由大到小顺序尝试设置，确保能够进入硬件低功耗，也可顺利唤醒。

2、 如果按照步骤 1 调试都无法成功进入或退出硬件低功耗，则可以结合 TK_ICGSR 寄存器一起调试。该寄存器直接作用于内部 CSD 模块群扫描补偿电流参数 CSD_ICOM，取值范围为 0~127。 默认为 40，一般该值配置越大，扫描信号越大，越容易低功耗唤醒；但要避免因为过大而无法进入低功耗。

3、 硬件低功耗扫描对整体硬件电路设计要求较高，如达不到则可能会出现无法唤醒情况，导致无法使用。建议客户注意这一风险，做好系统稳定与功耗的取舍，谨慎选择硬件低功耗扫描。一般情况下，选择软件低功耗扫描即可满足应用要求。如确需选择硬件低功耗扫描，必须在充分测试基础上进行。

附录：

全局灵敏度寄存器 TK_GSR 含义对照表-1

GbSen	N	CSD_WIN	FingerTH	NoiseTH	GbSen	N	CSD_WIN	FingerTH	NoiseTH
0	16	252	90	72	**32**	15	128	90	67
1	16	252	94	75	**33**	15	128	94	70
2	16	252	98	78	**34**	15	128	98	73
3	16	252	102	81	**35**	15	128	102	76
4	16	252	106	84	**36**	15	128	106	79
5	16	252	110	88	**37**	15	128	110	82
6	16	252	114	91	**38**	15	128	114	85
7	16	252	118	94	**39**	15	128	118	88
8	16	252	120	96	**40**	15	128	120	90
9	16	252	124	99	**41**	15	128	124	93
10	16	252	128	102	**42**	15	128	128	96
11	16	252	132	105	**43**	15	128	132	99
12	16	252	136	108	**44**	15	128	136	102
13	16	252	140	112	**45**	15	128	140	105
14	16	252	144	115	**46**	15	128	144	108
15	16	252	148	118	**47**	15	128	148	111
16	16	252	152	121	**48**	15	128	152	114
17	16	252	156	124	**49**	15	128	156	117
18	16	252	160	128	**50**	15	128	160	120
19	16	252	164	131	**51**	15	128	164	123
20	16	252	168	134	**52**	15	128	168	126
21	16	252	172	137	**53**	15	128	172	129
22	16	252	176	140	**54**	15	128	176	132
23	16	252	180	144	**55**	15	128	180	135
24	16	252	184	147	**56**	15	128	184	138
25	16	252	188	150	**57**	15	128	188	141
26	16	252	192	153	**58**	15	128	192	144
27	16	252	196	156	**59**	15	128	196	147
28	16	252	200	160	**60**	15	128	200	150
29	16	252	204	163	**61**	15	128	204	153
30	16	252	208	166	**62**	15	128	208	156
31	16	252	212	169	**63**	15	128	212	159

全局灵敏度寄存器 TK_GSR 含义对照表-2

GbSen	N	CSD_WIN	FingerTH	NoiseTH	GbSen	N	CSD_WIN	FingerTH	NoiseTH
64	14	64	90	63	**96**	13	32	90	58
65	14	64	94	65	**97**	13	32	94	61
66	14	64	98	68	**98**	13	32	98	63
67	14	64	102	71	**99**	13	32	102	66
68	14	64	106	74	**100**	13	32	106	68
69	14	64	110	77	**101**	13	32	110	71
70	14	64	114	79	**102**	13	32	114	74
71	14	64	118	82	**103**	13	32	118	76
72	14	64	120	84	**104**	13	32	120	78
73	14	64	124	86	**105**	13	32	124	80
74	14	64	128	89	**106**	13	32	128	83
75	14	64	132	92	**107**	13	32	132	85
76	14	64	136	95	**108**	13	32	136	88
77	14	64	140	98	**109**	13	32	140	91
78	14	64	144	100	**110**	13	32	144	93
79	14	64	148	103	**111**	13	32	148	96
80	14	64	152	106	**112**	13	32	152	98
81	14	64	156	109	**113**	13	32	156	101
82	14	64	160	112	**114**	13	32	160	104
83	14	64	164	114	**115**	13	32	164	106
84	14	64	168	117	**116**	13	32	168	109
85	14	64	172	120	**117**	13	32	172	111
86	14	64	176	123	**118**	13	32	176	114
87	14	64	180	126	**119**	13	32	180	117
88	14	64	184	128	**120**	13	32	184	119
89	14	64	188	131	**121**	13	32	188	122
90	14	64	192	134	**122**	13	32	192	124
91	14	64	196	137	**123**	13	32	196	127
92	14	64	200	140	**124**	13	32	200	130
93	14	64	204	142	**125**	13	32	204	132
94	14	64	208	145	**126**	13	32	208	135
95	14	64	212	148	**127**	13	32	212	137

全局灵敏度寄存器 TK_GSR 含义对照表-3

GbSen	N	CSD_WIN	FingerTH	NoiseTH	GbSen	N	CSD_WIN	FingerTH	NoiseTH
128	12	16	90	54	**160**	11	8	90	49
129	12	16	94	56	**161**	11	8	94	51
130	12	16	98	58	**162**	11	8	98	53
131	12	16	102	61	**163**	11	8	102	56
132	12	16	106	63	**164**	11	8	106	58
133	12	16	110	66	**165**	11	8	110	60
134	12	16	114	68	**166**	11	8	114	62
135	12	16	118	70	**167**	11	8	118	64
136	12	16	120	72	**168**	11	8	120	66
137	12	16	124	74	**169**	11	8	124	68
138	12	16	128	76	**170**	11	8	128	70
139	12	16	132	79	**171**	11	8	132	72
140	12	16	136	81	**172**	11	8	136	74
141	12	16	140	84	**173**	11	8	140	77
142	12	16	144	86	**174**	11	8	144	79
143	12	16	148	88	**175**	11	8	148	81
144	12	16	152	91	**176**	11	8	152	83
145	12	16	156	93	**177**	11	8	156	85
146	12	16	160	96	**178**	11	8	160	88
147	12	16	164	98	**179**	11	8	164	90
148	12	16	168	100	**180**	11	8	168	92
149	12	16	172	103	**181**	11	8	172	94
150	12	16	176	105	**182**	11	8	176	96
151	12	16	180	108	**183**	11	8	180	99
152	12	16	184	110	**184**	11	8	184	101
153	12	16	188	112	**185**	11	8	188	103
154	12	16	192	115	**186**	11	8	192	105
155	12	16	196	117	**187**	11	8	196	107
156	12	16	200	120	**188**	11	8	200	110
157	12	16	204	122	**189**	11	8	204	112
158	12	16	208	124	**190**	11	8	208	114
159	12	16	212	127	**191**	11	8	212	116

全局灵敏度寄存器 TK_GSR 含义对照表-4

GbSen	N	CSD_WIN	FingerTH	NoiseTH	GbSen	N	CSD_WIN	FingerTH	NoiseTH
192	10	4	90	45	**224**	9	2	90	40
193	10	4	94	47	**225**	9	2	94	42
194	10	4	98	49	**226**	9	2	98	44
195	10	4	102	51	**227**	9	2	102	45
196	10	4	106	53	**228**	9	2	106	47
197	10	4	110	55	**229**	9	2	110	49
198	10	4	114	57	**230**	9	2	114	51
199	10	4	118	59	**231**	9	2	118	53
200	10	4	120	60	**232**	9	2	120	54
201	10	4	124	62	**233**	9	2	124	55
202	10	4	128	64	**234**	9	2	128	57
203	10	4	132	66	**235**	9	2	132	59
204	10	4	136	68	**236**	9	2	136	61
205	10	4	140	70	**237**	9	2	140	63
206	10	4	144	72	**238**	9	2	144	64
207	10	4	148	74	**239**	9	2	148	66
208	10	4	152	76	**240**	9	2	152	68
209	10	4	156	78	**241**	9	2	156	70
210	10	4	160	80	**242**	9	2	160	72
211	10	4	164	82	**243**	9	2	164	73
212	10	4	168	84	**244**	9	2	168	75
213	10	4	172	86	**245**	9	2	172	77
214	10	4	176	88	**246**	9	2	176	79
215	10	4	180	90	**247**	9	2	180	81
216	10	4	184	92	**248**	9	2	184	82
217	10	4	188	94	**249**	9	2	188	84
218	10	4	192	96	**250**	9	2	192	86
219	10	4	196	98	**251**	9	2	196	88
220	10	4	200	100	**252**	9	2	200	90
221	10	4	204	102	**253**	9	2	204	91
222	10	4	208	104	**254**	9	2	208	93
223	10	4	212	106	**255**	9	2	212	95