CHSC6XXX Watch Driver Description
V1.0.1--2023-07-05

目录

1.基合 IC 报点数据格式

1.1 基合自容触控 IC **IIC 地址固定为 0x2E(7bit)，加上读写位，写 0x5C，读 0x5D。**

1.2 读坐标可以直接读或写任意地址读。

读 3 个 byte 数据格式：

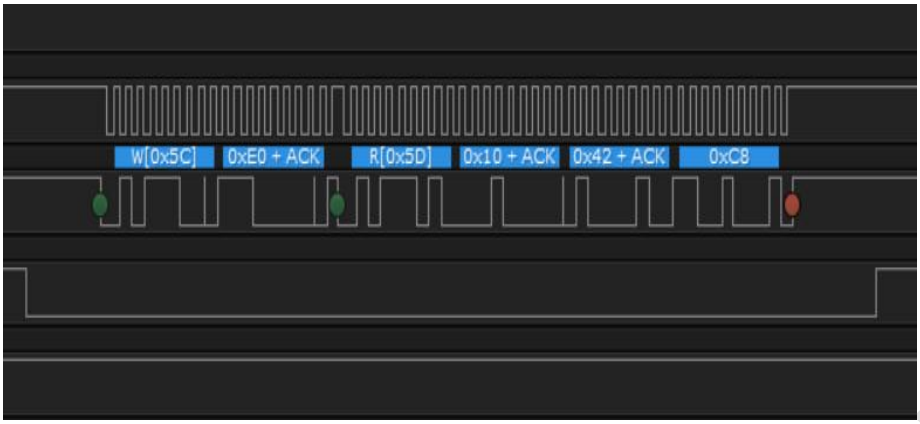
Byte0	[7]:y 坐标大于 255 时的高位 [6]:x 坐标大于 255 时的高位 [5:4]: 当前点的触摸状态: 00：按下；10：保持；01：松开； [3:2]: RSVD：00； [1:0]: 触摸点个数：00：无触摸；01：单指触摸 [0:7]:FC-手掌覆盖 [0:7]:FD-双击事件
Byte1	X 坐标低位
Byte2	Y 坐标低位

读 5 个 byte 数据格式：

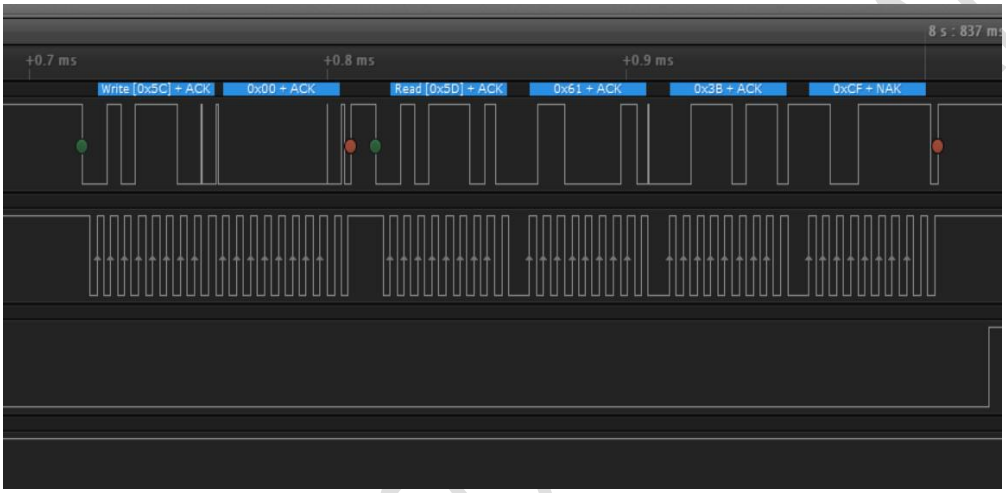
Byte0	触摸点个数：00：无触摸；01：单指触摸
Byte1	[7:4]:当前点的触摸状态: 0：按下；8：保持；4：松开 [3:0]:X 坐标高 4 BIT 位
Byte2	[7:0]:X 坐标低 8 BIT 位
Byte3	[7:4]:手指 ID，单指是 0 [3:0]:Y 坐标高 4 BIT 位
Byte4	[7:0]:Y 坐标低 8 BIT 位

1.3 读坐标长度为固定 3 个 byte，固件默认只支持 1 点坐标，具体格式定义如下：

1.4 读坐标支持的 3 种 IIC 波形：



类型 1



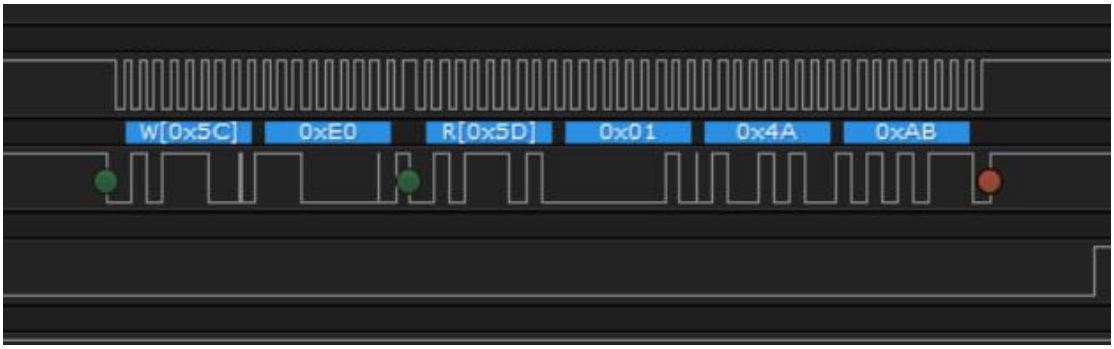
类型 2

类型 3

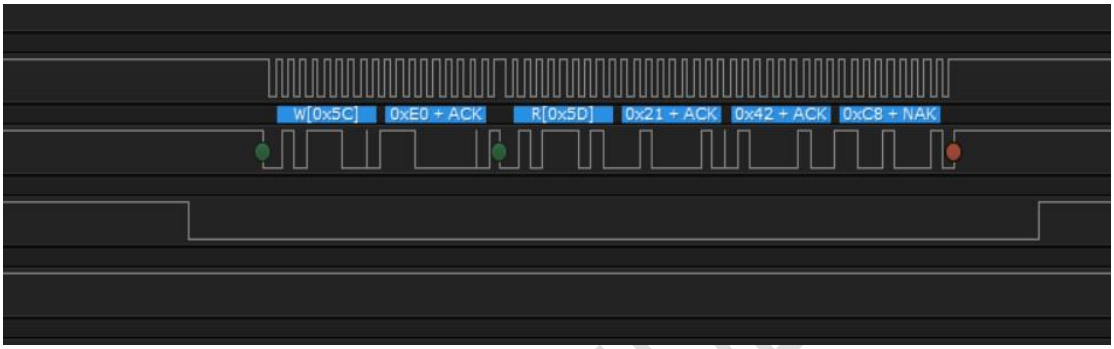
1.5 具体坐标数据示例：

下面是单点坐标数据上报的示例

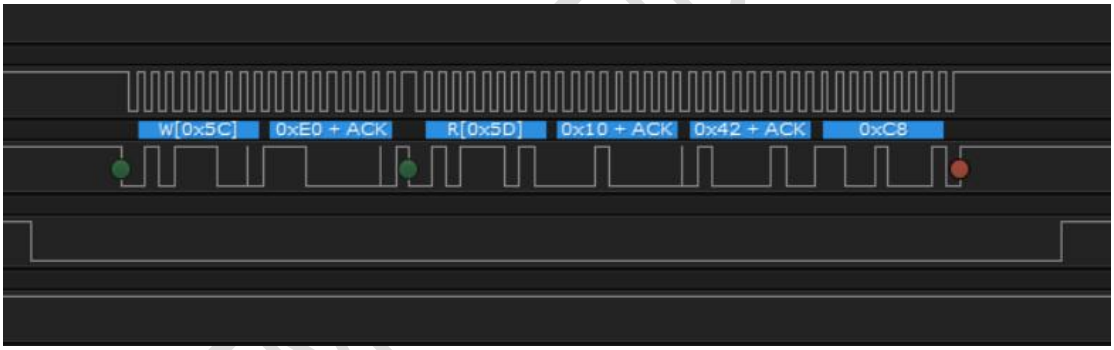
按下：



保持：



松开：



1.6 读取坐标数据，在 chsc6x_main.c -> chsc6x_read_touch_info（函数）

2. struct ts_fw_infos 结构体介绍

```
struct ts_fw_infos {
    unsigned short chsc6x_cfg_version;
    unsigned short chsc6x_boot_version;
    unsigned short chsc6x_vendor_id;
    unsigned short chsc6x_project_id;
    unsigned short chsc6x_chip_id;
    unsigned short chsc6x_chip_type;
    unsigned short chsc6x_rpt_lcd_x;
    unsigned short chsc6x_rpt_lcd_y;
    unsigned short chsc6x_max_pt_num;
};
```

- chsc6x_cfg_version
  cfg bin 文件（正常调试主要改此文件）的版本号，也就是我们俗称的总的固件版本号。

- chsc6x_boot_version
  boot bin 文件（主要是固件算法部分，改动较少）的版本号。
- chsc6x_vendor_id
  供应商编号。
- chsc6x_project_id
  供应商对当前 TP 的项目编号。
- chsc6x_chip_id
  触控芯片的 IIC 模块的 Slaver Id，基合芯片此 id 位 8bit 0x5c。
- chsc6x_chip_type
  基合触控芯片的具体型号，目前我司自容类触控 IC 型号如下：
  1——TLSC6206A　2——TLSC6306　3——TLSC6206
  4——TLSC6324　5——TLSC6332　6——CHSC6440
  7——CHSC6448　8——CHSC6432;　9——CHSC6424
  10——CHSC6306BF　11——CHSC6413　12——CHSC6417
  13——CHSC6540
- chsc6x_rpt_lcd_x
  TP 上报的 X 坐标最大值。
- chsc6x_rpt_lcd_y
  TP 上报的 Y 坐标最大值。
- chsc6x_max_pt_num
  当前固件支持的最多报点个数，自容类 IC 最多支持两点触控，穿戴类项目因为可视区较小，默认只支持单点触控。

3.全局接口函数介绍

- int chsc6x_tp_dect (struct ts_fw_infos *pfw_infos,
  unsigned char *update_ret_flag);

1) 此接口函数须在系统初始化过程中调用，来检测 TP IC 是否为基合触控 IC，检测到是基合的触控 IC 则返回 1，检测不到则返回 0。
2) 此接口函数在检测到基合触控 IC 后，会读取此项目相关的 TP 信息并填进 pfw_infos 指向的结构体中，只填前面 6 个元素的信息。
3) 在打开 CHSC6X_AUTO_UPGRADE 宏开关情况下，调用此接口函数就会在开机过程中尝试升级固件，最后一个参数 update_ret_flag 的值就是升级是否成功的返回值，成功返回 1，失败返回 0。
4) 驱动开机升级的内容即为 chsc6x_flash_boot.h 文件中的数组，需要 FAE 提供。
5) 注意此接口函数升级固件时，必须确保驱动编译的固件数组的版本号大于 TP 中的当前固件版本号（版本号只能增大），才会执行升级流程，否则不会升级；或者在升级过程中 IIC 出错且多次尝试失败后，重启设备再次调用此接口函数时，会强制更新固件，以此来防止升级变砖。

- void chsc6x_ota_upgrade_tp_fw (struct ts_fw_infos *pfw_infos,
  unsigned char* p_fw_upd, unsigned int fw_len);

1) 此接口函数是为 OTA 升级封装的固件升级接口，在合适场景下调用即可。
2) 此接口函数调用成功后，会读取此项目相关的 TP 信息并填进 pfw_infos 指向的结构体中，只填前面 6 个元素的信息。
3) p_fw_upd 为升级数组的指针，fw_len 为升级数组的字节长度。
4) 注意此接口函数升级固件时，只要接口传入的升级数组的固件版本号与 TP 当前固件版

本号**不一样**（版本号可加可减），就会执行升级流程，否则不会升级；或者在升级过程中 IIC 出错且多次尝试失败后，再次调用此接口函数时，会强制更新固件，以此来防止升级变砖。

- void chsc6x_get_chip_info(struct ts_fw_infos * **pfw_infos**);

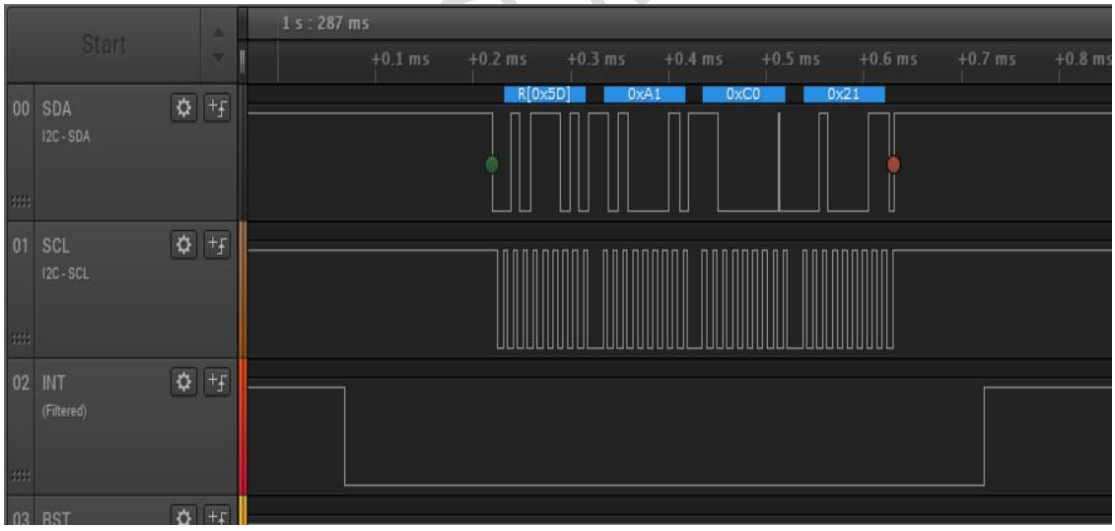
1) 此接口函数用来获取 ts_fw_infos 结构体中的所有元素信息，可在任意时刻 调用。
2) 此接口函数调用成功后，会将此 TP 项目相关的信息填进 **pfw_infos** 指向的结构体中。

4.平台相关接口函数介绍

- **要实现基合 TP 的所有功能，须将以下 chsc6x_platform.c 文件中的 5 个与平台相关的函数接口完全实现，同时须配置 chsc6x_platform.h 中的宏定义。**

➢ int chsc6x_i2c_read(unsigned char **id**, unsigned char ***p_data**,
unsigned short **lenth**);

1) 此函数接口专门用来读取 IIC 传输的报点信息。
2) 参数 **id** 是 IIC 模块的 slaver id，参数 **p_data** 为读到的数据存放的内存地址，参数 **lenth** 为要读取的数据的字节长度。
3) 此函数调用**成功**无出错，必须**返回**成功读到的**数据的字节长度**；其它返回值则表示调用出错。
4) INT 引脚下降沿时，调用此函数读取坐标数据，读取时可以直接读，不用写任何寄存器地址，如果写了地址也没事，读到的是一样的结果，建议按不写寄存器地址的方式来读，这样 TP 功耗会更低；INT 下降沿时尽量及时读走数据，不能及时读走数据会引起功耗会高一些，严重时甚至会引起与下一帧坐标数据读写冲突导致坐标错乱问题。
5) 如下示例为典型的读坐标 IIC 波形：

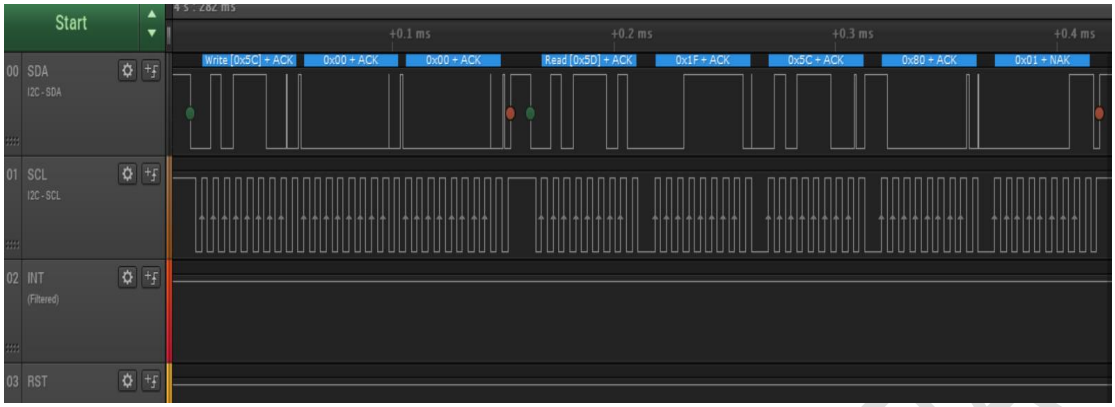


➢ int chsc6x_read_bytes_u16addr_sub (unsigned char **id**,
unsigned short **adr**, unsigned char ***rxbuf**, unsigned short **lenth**);

1) 此函数接口专门用来读取基合 IC 内部寄存器，在固件升级，读取 TP 信息等操作时会用到此接口。
2) 参数 **id** 是 IIC 模块的 slaver id，参数 **adr** 为要访问的基合 IC 内部寄存器的 16 位地址，2 个字节；参数 **rxbuf** 为读到的数据存放的内存地址，参数 **lenth** 为要读取的数据的字节长度。
3) 此函数接口调用成功无出错，必须返回 0；非 0 则表示调用出错。

4) 调用此接口须根据平台对单帧**读字节数的限制**，来配置 MAX_IIC_RD_LEN 宏定义的值，以免引起**系统死机类问题**，通常配置的越大烧录会越快。

5) 如下示例波形为从基合 IC 内部 0x0000 寄存器读取连续的 4 个字节。



➢ int **chsc6x_write_bytes_u16addr_sub** (unsigned char **id**,
unsigned short **adr**, unsigned char ***txbuf**, unsigned short **lenth**);

1) 此函数接口专门用来写数据到基合 IC 内部寄存器，在固件升级，读取 TP 信息等操作时会用到此接口。

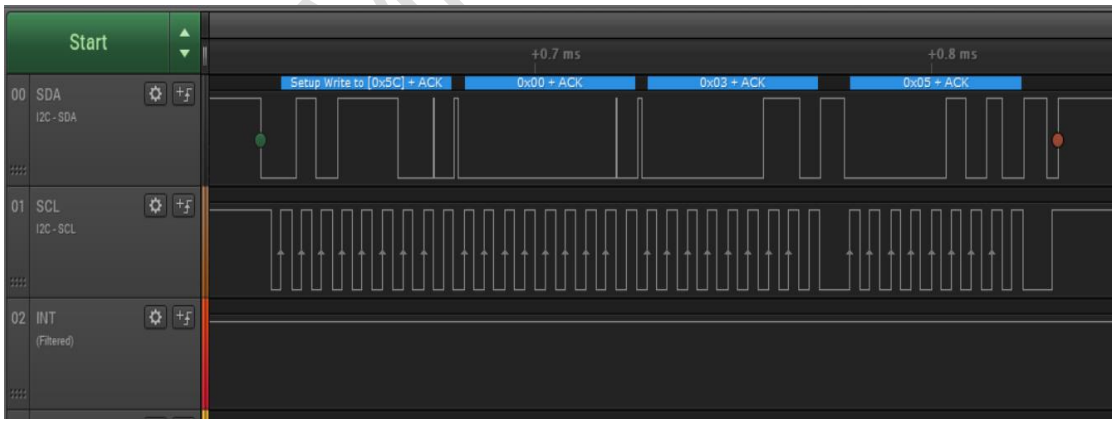
2) 参数 **id** 是 IIC 模块的 slaver id，参数 **adr** 为要访问的基合 IC 内部寄存器的 16 位地址，2 个字节；参数 **txbuf** 为准备的写数据存放的内存地址，参数 **lenth** 为要写入的数据的字节长度。

3) 此函数接口调用成功无出错，必须返回 0；非 0 则表示调用出错。

4) 调用此接口须根据平台对单帧**写字节数的限制**，来配置 MAX_IIC_WR_LEN 宏定义的值，以免引起**系统死机类问题**，通常配置的越大烧录会越快。

5) **休眠指令**，即写 **0xA5 0x03**，也是通过调用此接口实现的，只需要 adr=0xa503，lenth=0 即可实现。

6) 如下示例波形为往基合 IC 内部 0x0003 寄存器写入 1 个字节。



➢ void **chsc6x_msleep** (int **ms**);

此函数接口实现，等待 **ms** 毫秒时间，用于固件升级时等待烧录过程完成。

➢ void **chsc6x_tp_reset**(void);

此函数接口复位 TP，复位脚拉低和拉高各 30ms，误差不要太大，否则会有异常。

5. IIC 主控访问触控 IC 内存的方式

- **基合触控 IC 内部集成了两种 IIC 工作模式，具体介绍如下：**

➢ **直接地址访问模式（DMA 模式）**

直接地址访问模式，表示 IIC 的主控可以访问 TP IC 的任意地址，也就是说 HOST 访问的时候要提供 16 位的起始地址，固件升级，获取 TP 信息等必须工作在此模式下；

➢ **映射地址访问模式（Mapping 模式）**

1) HOST 的读或写访问都会分别映射到固定的内存空间，FW 可以配置这里提到的映射的内存空间（定义为 MTK_TXRX_BUF，设定的地址是 0x809000）。对于 chsc64xx 读和写操作映射的空间大小上限 128 字节。在这种模式下，IIC 上写进来的数据除了 SLAVE 地址外都被当作普通的数据写到配置好了的 RAM 空间，在读操作的时候，IIC 上的数据即为配置好的 RAM 空间里面的数据，读坐标就是在这种模式下工作的。
2) 映射地址访问模式的设定是为了防止主控无意的将错误的数据写入到 IC 内部的寄存器，从而可能导致 IC 运行固件失败或者不可意料的未知错误；

➢ **两种地址访问模式的切换**

1) TP IC 默认工作在“直接地址访问模式”下，固件运行起来后会通过修改寄存器切换到“映射地址访问模式”，即常规工作模式为“映射地址访问模式”；驱动中将直接地址模式切换到映射地址只需要调用 `chsc6x_tp_reset()` 函数即可。
2) 驱动中，需要获取 TP IC 的某些信息，例如 TP 的固件版本号，供应商编号以及项目编号等等信息时，需要将映射地址访问模式切换到直接地址访问模式，然后直接读取相应的内存区域并解析即可；

驱动中将映射地址模式切换到直接地址访问模式的函数接口是：

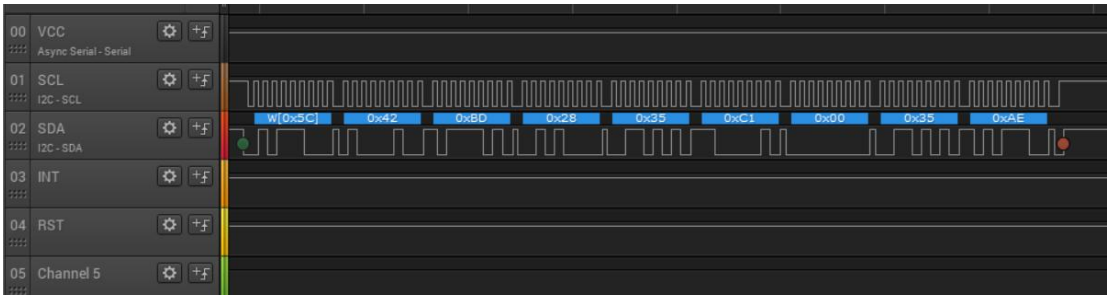
```
int chsc6x_set_dd_mode (void);
```

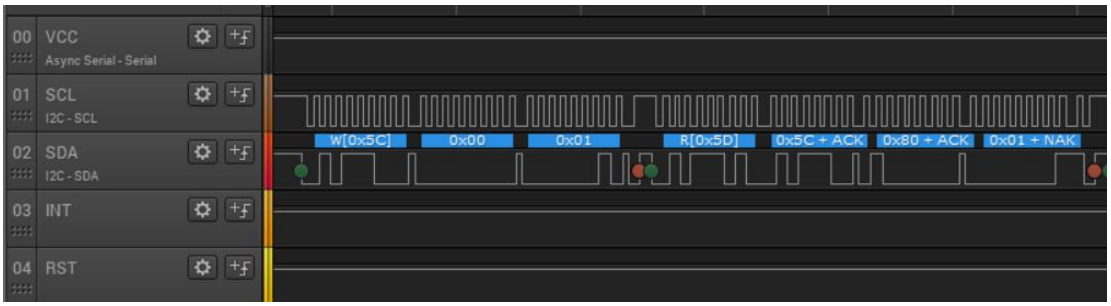
6. 类寄存器常用操作

6.1 获取 TP 相关信息

了解第 5 节内容后，我们可以知道通过 IIC 主控访问 IC 内存的方式，达到类似于读取寄存器相关的操作

6.1.1 第一步可以拉一次 RST，延时 30ms 后，主控 W 0x42 0xBD 0x28 0x35 0xC1 0x00 0x35 0xAE，切换 TP 到直接地址访问模式（DMA）。切换地址成功后，延时 20ms，能从 0x0001 地址获取到 IIC 地址，如下图：





6.1.2 W 0x9E00 R 4 ,考虑小端存储，下图读到 0x0E3E,0X100E 连续 2 个 short 类型数据。

第一个 short 数据,[0:8]项目编号 PID 低 9 位，[9：15]供应商编号 VID 低 7 位。

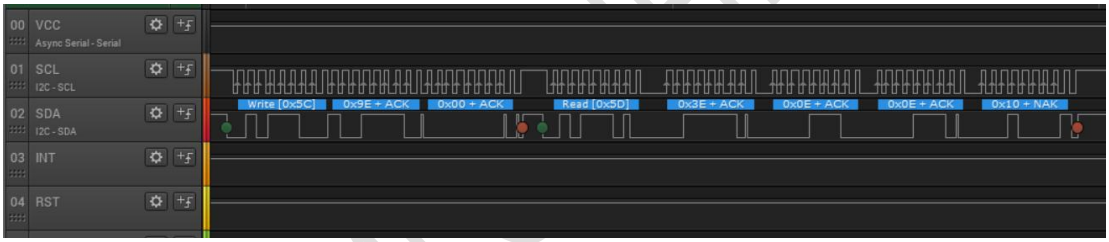
第二个 short 数据,[4:5]项目编号 PID 的高 2 位,[6:7]供应商编号 VID 高 2 位,[10:15]cfg 固件版本号。

解析后：

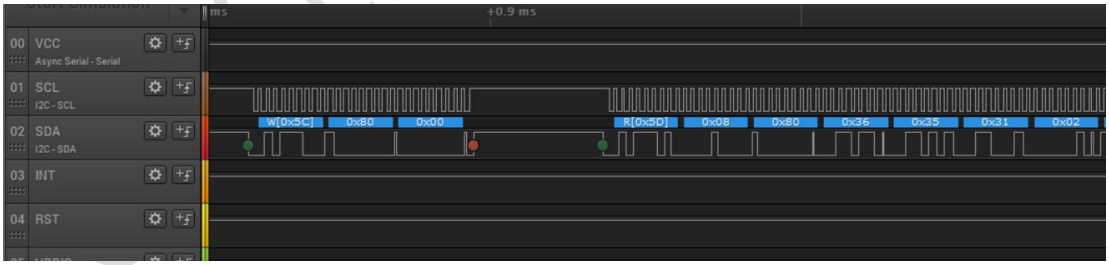
VID=0x7=7;

PID=0x3E=62;

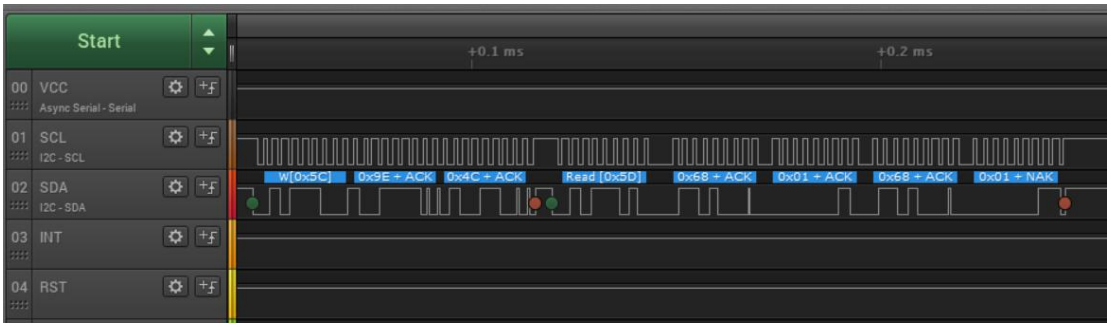
cfg=0x4=4;



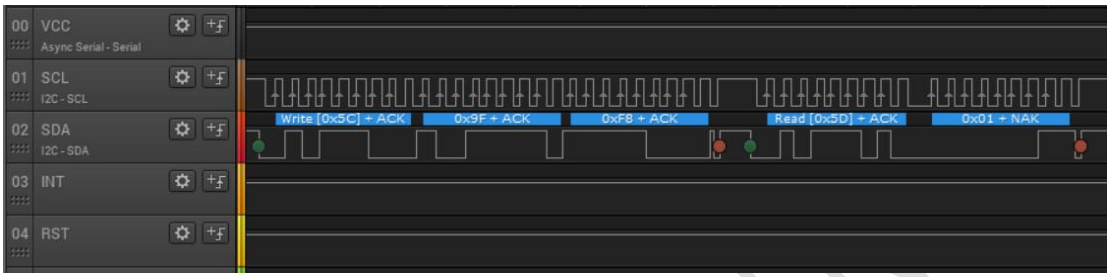
6.1.3 W 0x8004， R 2 ,读到 0x0231,即 boot 固件版本。



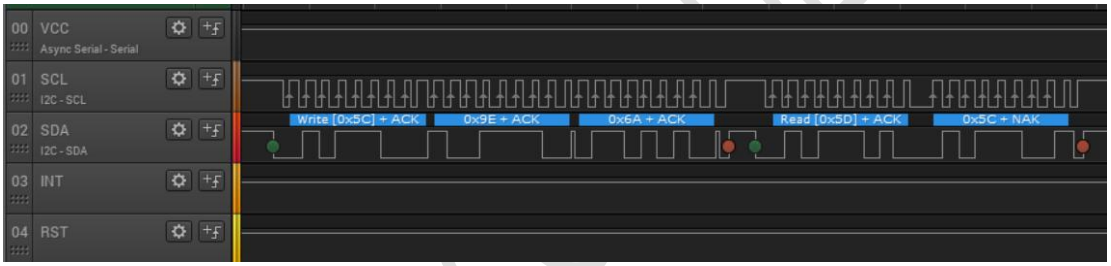
6.1.4 W 0x9E4C， R 4,读到 0x0168 和 0x0168 连续 2 个 short 类型数据，对应 x y 分辨率。



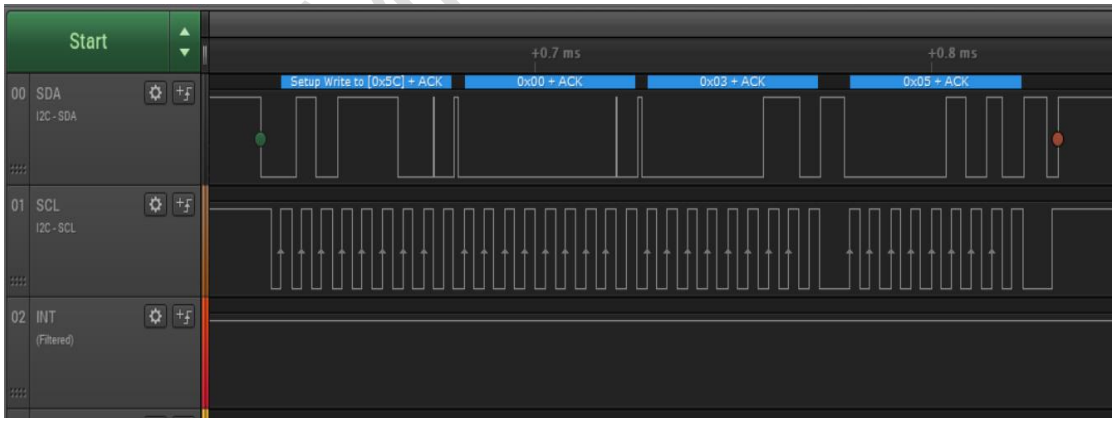
6.1.5 W 0x9FF8, R 1,当前固件支持的最大触摸点数。



6.1.6 W 0x9E6A, R 1,获取 IIC 地址，可视作读 ID 动作。



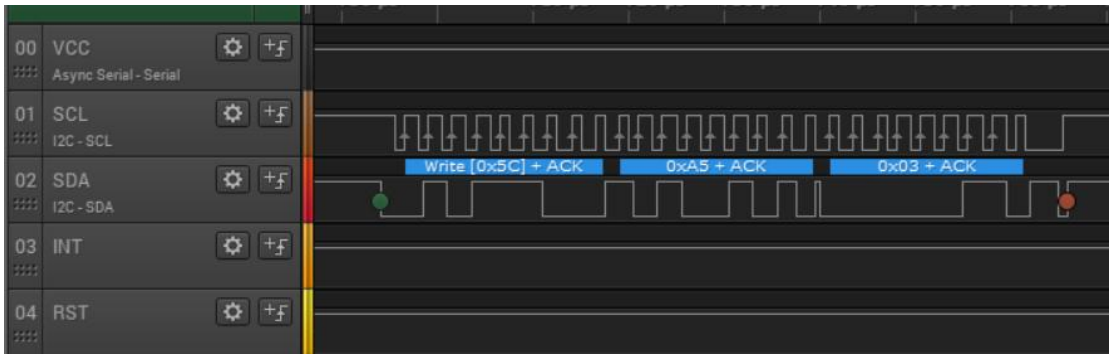
完成读取操作后，复位一下 TP，退出直接地址模式（通过寄存器方式，往 0x3 地址写 0x5，也可切回映射地址模式）。



以下操作，在映射地址模式下完成。

6.2 TP 进入休眠

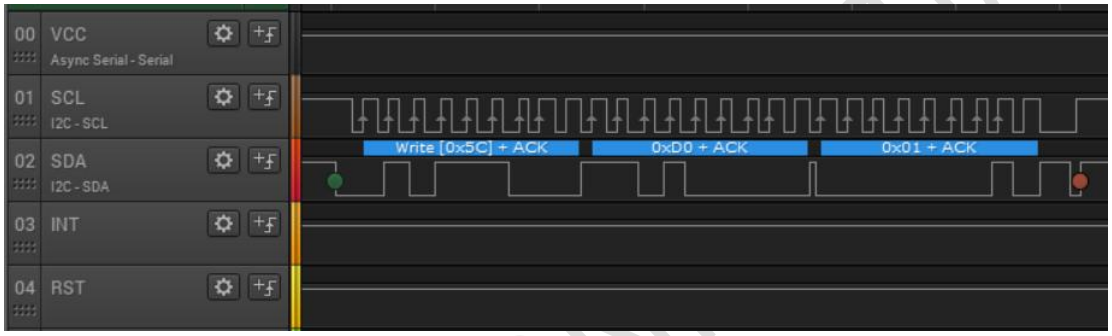
W 0xA5 0x03



复位后退出

6.3 TP 进入手势模式

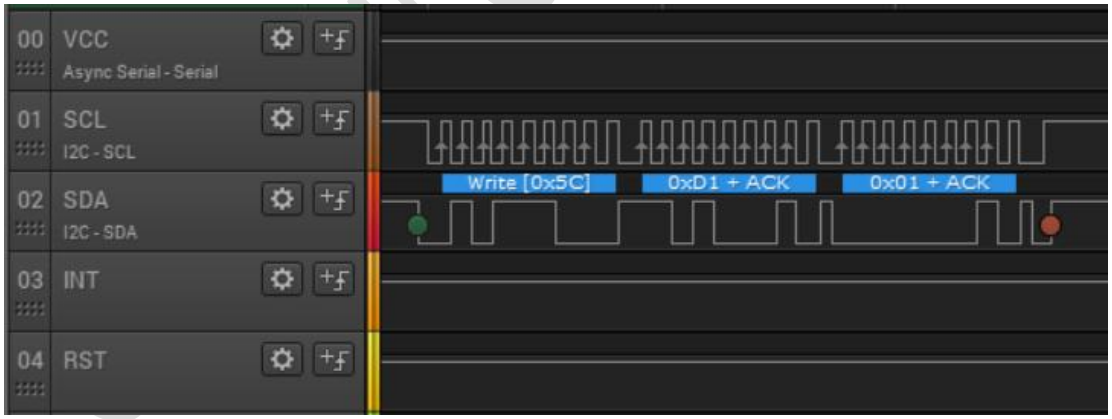
W 0xD0 0x01



复位后退出

6.4 进入 Palm check

W 0xD1 0x01



复位后退出