

Linux I²C Touch Device Driver

Version: V0_0_0_7
Document: ILITEK_LINUX_I2C_DRIVER.pdf

ILI TECHNOLOGY CORP.

8F, No.1, Taiyuan 2nd-St., Jhubei City, Hsinchu County 302, Taiwan, R.O.C.
Tel.886-3-5600099; Fax.886-3-5600055
<http://www.ilitek.com>

目錄

章節	頁碼
1. 總體描述	3
2. 標頭檔中相關宏的說明	4
3. 部分代碼說明	7
4. 相關功能說明	8
5. 常見問題	13

1. 總體描述

- A. 這份檔對ILITEK_LINUX_I2C_DRIVER進行說明。此版驅動將不同平臺整合在一起，其實就是將不同平臺能夠共用的就共用，不能共用的會進行區分，因此寫Makefile選擇編譯哪些檔時要注意，若MTK平臺不是使用dts的方式需要開啟MTK_UNDTS這個巨集。同時在ilitek_ts.h標頭檔中設定選擇的平臺（沒有的可以直接設定ILITEK_PLAT_QCOM），以對應各平臺的差異！設定方式如下：

```
00064:
00065: #define ILITEK_PLAT_QCOM
00066: #define ILITEK_PLAT_MTK
00067: #define ILITEK_PLAT_ROCKCHIP
00068: #define ILITEK_PLAT_ALLWIN
00069: #define ILITEK_PLAT_AMLOGIC
00070:
00071: #define ILITEK_PLAT
00072:
00073:
```

1
2
3
4
5

ILITEK_PLAT_QCOM

支援晶片型號	ILI230X、ILI231X、ILI251X
I2C 設備位址（7位元）	0x41
開機自動升級	標頭檔包含ili檔，或者使用bin檔
支援平臺	Qcom、Rockchip、MTK（dts方式）、Allwinner、Amlogic，沒有對應平臺就以Qcom為base調試

- B. 各文件說明如下：

ilitek_ts.h：提供客戶修改，驅動標頭檔，包含驅動中要用到的一些標頭檔以及宏和函數的聲明等。

ilitek_common.h：驅動標頭檔，包含驅動中要用到的一些標頭檔以及宏和函數的聲明等。

ilitek_platform_init.c：平臺載入初始化需要的檔。

ilitek_main.c：驅動主文件，完成probe的具體實現，GPIO註冊、讀取TP資訊、註冊輸入裝置資訊，報點，休眠喚醒處理等

ilitek_update.c：驅動升級IC固件功能的具體實現

ilitek_tool.c：用於支持TouchUtility apk 以及創建用命令升級固件、sensor test、查看固件版本的設備節點以及一些調試命令的使用等等

ilitek_protocol.c：共用的函式與ilitek protocol等。

- C. 驅動移植說明：

1. 將 ilitek_lim 資料夾複製到 kernel/drivers/input/touchscreen/（一般是放在這裡，如果平臺有指定則放到指定路徑下）下麵，
2. 在 kernel/drivers/input/touchscreen/Makefile裡面添加一行obj-y += ilitek_lim/，如果需要Kconfig,可自行寫Kconfig檔
3. 根據具體平臺選擇所需編譯的文件如下：

```
obj-y += ilitek_main.o \
ilitek_platform_init.o \
ilitek_update.o \
ilitek_tool.o \
ilitek_protocol.o
```

4. 添加 I2C 設備：

對於使用 board file 的方式，找到 kernel 中初始化 I2C 匯流排的板級檔，若本驅動測試過的 Tiny 4412 此檔為 linux-3.5/arch/arm/mach-exynos/mach-tiny4412.c，添加內容如下：

```
00985:
00986: static struct i2c_board_info initdata i2c_tpd={
00987:     I2C_BOARD_INFO("ilitek i2c", 0x41),
00988:     //platform_data = &ilitek_pdata,
00989:
00990: };
                                名称      地址

02243:     i2c_register_board_info(2, &i2c_tpd, 1);
02244:
                                总线号
```

對於使用 dts 註冊的方式，可在 dts 檔對應 I2C 匯流排節點下加入如下參考內容：

```
ilitek@41 {
    compatible = "tchip,ilitek";
    reg = <0x41>;
    interrupt-parent = <&msm_gpio>;
    interrupts = <13 0x0>;
    vdd-supply = <&pm8916_l17>;
    vcc_i2c-supply = <&pm8916_l6>;
    ilitek,irq-gpio = <&msm_gpio 13 0x0>;
    ilitek,reset-gpio = <&msm_gpio 12 0x0>;
    ilitek,vbus = "vcc_i2c";
    ilitek,vdd = "vdd";
    ilitek,name = "ilitek_i2c";
};
```

2. 標頭檔中相關宏的說明

驅動版本資訊，不要修改第一碼（DERVER_VERSION_MAJOR）

//driver information

```
#define DERVER_VERSION_MAJOR      5
#define DERVER_VERSION_MINOR      0
#define CUSTOMER_ID               0
#define MODULE_ID                 0
#define PLATFORM_ID               0
#define PLATFORM_MODULE           0
#define ENGINEER_ID               0
```

平臺設定，選擇對應的平臺賦給 ILITEK_PLAT 這個宏，沒有可選擇 QCOM，對應編譯 ilitek_platform_init.c 這個檔，如果使用平臺有特定的設定，則代碼內相關部分按照使用平臺的使用方式修改

```
#define ILITEK_PLAT_QCOM          1
#define ILITEK_PLAT_MTK          2
```

```
#define ILITEK_PLAT_ROCKCHIP 3
#define ILITEK_PLAT_ALLWIN 4
#define ILITEK_PLAT_AMLOGIC 5
```

```
#define ILITEK_PLAT ILITEK_PLAT_QCOM
```

```
#define ILITEK_TOOL
供調試工具所用，默認開啟
```

```
#define ILITEK_TUNING_MESSAGE
供 FW 調試時丟出 debug 資訊所用，預設開啟
```

```
//#define ILITEK_ESD_PROTECTION
ESD 保護開關，預設關閉
```

```
#define ILITEK_TOUCH_PROTOCOL_B
報點協定使用 B 類報點，默認開啟
```

```
//#define ILITEK_USE_LCM_RESOLUTION
使用 LCM 的解析度，默認關閉
```

```
//#define MTK_UNDTS //no use dts and for mtk old version
MTK 平臺使用非 dts 方式需要開啟此巨集
```

```
#define ILITEK_ROTATE_FLAG 0
報點將 X、Y 調換，默認設為 0，啟用時設為非 0 即可
```

```
#define ILITEK_REVERT_X 0
報點將 X 做鏡像，即最大變最小，最小變最大，默認設 0，啟用時設為非 0 即可
```

```
#define ILITEK_REVERT_Y 0
報點將 Y 做鏡像，即最大變最小，最小變最大，默認設 0，啟用時設為非 0 即可
```

```
#define ILITEK_ENABLE_REGULATOR_POWER_ON
使用 regulator 方式上電，當需要我們驅動來控制上電且是這種方式時可開啟此巨集，預設打開
```

```
#define ILITEK_GET_GPIO_NUM
當我們驅動能夠解析獲取 reset、irq 對應 pin 腳時，開啟此宏，目前只有寫用 dts 方式解析，當不開此巨集時需要設定 ILITEK_RESET_GPIO 和 ILITEK_IRQ_GPIO 為對應的值
```

```
#define ILITEK_CLICK_WAKEUP                                0
單機喚醒（driver 實現），主要用於大尺寸

#define ILITEK_DOUBLE_CLICK_WAKEUP                        1
按兩下喚醒（driver 實現），主要用於大尺寸

#define ILITEK_GESTURE_WAKEUP                              2
手勢喚醒（FW 實現），主要用於小尺寸

//#define ILITEK_GESTURE                                ILITEK_CLICK_WAKEUP
設定手勢喚醒方式，預設關閉此功能

//#define ILITEK_UPDATE_FW
開機升級功能，前期調試時建議關閉，調試好後再開啟測試，默認關閉

#define ILI_UPDATE_BY_CHECK_INT
升級時通過檢測 INT 的狀態來看 IC 是否 ready 好繼續寫下一筆資料，開此巨集升級速度較快，針對大尺寸 ILI2302 和 ILI2312，默認開啟

#define ILITEK_UPGRADE_WITH_BIN                            0
升級檔案用 bin 檔，且是放在系統對應 firmware 資料夾下（一般為/system/etc/firmware/，或自己定義的路徑），

#define ILITEK_FW_FILENAME                                "ilitek_i2c.bin"
升級檔案用 bin 檔的檔案名設定

#define TOUCH_SCREEN_X_MAX                                (1080) //LCD_WIDTH
#define TOUCH_SCREEN_Y_MAX                                (1920) //LCD_HEIGHT
LCD 的解析度設定，當 MTK 平臺且我們代碼有開啟 ILITEK_USE_MTK_INPUT_DEV 或有開啟 ILITEK_USE_LCM_RESOLUTION 這個宏時要設定正確

#define ILITEK_USE_MTK_INPUT_DEV
針對 MTK 平臺使用平臺內的 tpd->dev，默認開啟，當報點有異常時可以關掉此宏看看，預設開啟

#if defined ILITEK_GET_GPIO_NUM
#undef ILITEK_GET_GPIO_NUM
#endif
針對 MTK 平臺不需要解析 reset、irq 對應 pin 腳時關掉 ILITEK_GET_GPIO_NUM 這個宏，注意設定 ILITEK_RESET_GPIO 和 ILITEK_IRQ_GPIO 為對應的值

#define ILITEK_ERR_LOG_LEVEL                               (1)
#define ILITEK_INFO_LOG_LEVEL                              (3)
#define ILITEK_DEBUG_LOG_LEVEL                             (4)
#define ILITEK_DEFAULT_LOG_LEVEL                           (3)

#define debug_level(level, fmt, arg...) do {\
```

```

if (level <= ilitek_log_level_value) {\
    if (level == ILITEK_ERR_LOG_LEVEL) {\
        printk(" %s ERR   line = %d %s : "fmt, "ILITEK", __LINE__, __func__, ##arg);\
    }\
    else if (level == ILITEK_INFO_LOG_LEVEL) {\
        printk(" %s INFO line = %d %s : "fmt, "ILITEK", __LINE__, __func__, ##arg);\
    }\
    else if (level == ILITEK_DEBUG_LOG_LEVEL) {\
        printk(" %s DEBUG line = %d %s : "fmt, "ILITEK", __LINE__, __func__, ##arg);\
    }\
}\
} while (0)

```

```

#define tp_log_err(fmt, arg...) debug_level(ILITEK_ERR_LOG_LEVEL, fmt, ##arg)
#define tp_log_info(fmt, arg...) debug_level(ILITEK_INFO_LOG_LEVEL, fmt, ##arg)
#define tp_log_debug(fmt, arg...) debug_level(ILITEK_DEBUG_LOG_LEVEL, fmt, ##arg)

```

log 列印相關，等級設定與 tp_log_err、tp_log_info、tp_log_debug 對應，ILITEK_DEFAULT_LOG_LEVEL 預設列印等級，低於或等於此等級的都會列印，log 關鍵字 “ILITEK”

3. 部分代碼說明

int ilitek_power_on(bool status)

當有開啟 **ILITEK_ENABLE_REGULATOR_POWER_ON** 這個宏時才有具體實現，對應的 **ilitek_data->vdd** 或 **ilitek_data->vdd_i2c** 在對應的平臺初始化代碼內有實現，若是其他方式按具體方式修改

int ilitek_get_gpio_num(void)

會去獲取對應的 **reset**、**irq gpio**，若不需要獲取則注意設定 **ILITEK_RESET_GPIO** 和 **ILITEK_IRQ_GPIO** 為對應的值

int ilitek_request_gpio(void)

申請 **gpio**，當申請失敗時會先 **free** 然後再 **try** 一次，申請成功後會設定 **reset** 輸出高，**irq** 為輸入，關於 **GPIO** 的操作請以具體平臺操作方式為準

reset 函數，**delay** 為從拉低到拉高後的延時時間，需要大於 **IC** 的初始化時間，之前有發現 **MTK** 平臺用 **tpd_gpio_output** 這個介面不能正常拉高拉低，可以改為 **gpio_direction_output** 測試看看

```

void ilitek_reset(int delay) {
    tp_log_info("delay = %d\n", delay);
    if (ilitek_data->reset_gpio > 0) {
        #if ILITEK_PLAT != ILITEK_PLAT_MTK
            gpio_direction_output(ilitek_data->reset_gpio,1);
            mdelay(10);
            gpio_direction_output(ilitek_data->reset_gpio,0);

```

```

        mdelay(10);
        gpio_direction_output(ilitek_data->reset_gpio, 1);
        mdelay(delay);
    #else
        tpd_gpio_output(ilitek_data->reset_gpio, 1);
        mdelay(10);
        tpd_gpio_output(ilitek_data->reset_gpio, 0);
        mdelay(10);
        tpd_gpio_output(ilitek_data->reset_gpio, 1);
        mdelay(delay);
    #endif
}
else {
    tp_log_err("reset pin is invalid\n");
}
return;
}

```

int ilitek_read_tp_info(void)

1. 當判斷到 IC 為 ILI2511 時會將 ilitek_repeat_start 置為 false
2. 對於大尺寸 IC 當 0xC0 命令讀到資料為 0x55（即 BL 模式）是會將強制升級的 flag 置起來，強制升級 flag : ilitek_data->force_update
3. 存放按鍵資訊的 keyinfo 這個陣列設定大小為 10，當按鍵數大於 10 時需在 struct ilitek_ts_data 結構體內修改成員 keyinfo 的大小

static int ilitek_request_irq(void)

```

#if ILITEK_PLAT != ILITEK_PLAT_MTK
    ilitek_data->client->irq = gpio_to_irq(ilitek_data->irq_gpio);
#else
    node = of_find_matching_node(NULL, touch_of_match);
    if (node) {
        ilitek_data->client->irq = irq_of_parse_and_map(node, 0);
    }
#endif

```

注意 irq 號碼的獲取，若使用平臺對 irq 號有特殊的設定注意修改此處得到正確的中斷號

4. 相關功能說明

A. 開機升級功能

1. 開啟 ILITEK_UPDATE_FW 這個巨集開關

2. 若是使用 ili 檔，則需提供 ili 檔才能使用此功能
3. 是否升級判定方式

```
tp_log_info("ilitek df_startaddr=0x%X, df_endaddr=0x%X, df_checksum=0x%X, df_len = 0x%X\n", df_startaddr, df_endaddr, df_checksum, df_len);
if (!ilitek_data->force_update) {
    for (i = 0; i < 8; i++) {
        tp_log_info("ilitek_data.firmware_ver[%d] = %d, firmware_ver[%d] = %d\n", i, ilitek_data->firmware_ver[i], i, firmware_ver[i]);
        if (firmware_ver[i] < ilitek_data->firmware_ver[i]) {
            i = 8;
            break;
        }
        if (firmware_ver[i] > ilitek_data->firmware_ver[i]) {
            break;
        }
    }
}

if (i >= 8) {
    if (!ilitek_data->ic_2120) {
        tp_log_info("firmware version is older so not upgrade\n");
    }
    else {
        tp_log_info("firmware version is same so not upgrade\n");
        if (ILITEK_UPGRADE_WITH_BIN) {
            if (CTPM_FW_BIN) {
                vfree(CTPM_FW_BIN);
                CTPM_FW_BIN = NULL;
            }
        }
    }
    return 1;
}
} ? end if !(ilitek_data->force_... ?
```

4. 升級流程選擇

```
if ((ilitek_data->mcu_ver[0] == 0x11 || ilitek_data->mcu_ver[0] == 0x10) && ilitek_data->mcu_ver[1] == 0x25) {
    df_startaddr = 0xF000;
    ret = ilitek_upgrade_2511(df_startaddr, df_endaddr, ap_startaddr, ap_endaddr, CTPM_FW);
    if (ret < 0) {
        tp_log_err("ilitek_upgrade_2511 err ret = %d\n", ret);
        //goto Retry;
    }
} else {
    df_startaddr = 0x1F000;
    if (df_startaddr < df_endaddr) {
        ilitek_data->has_df = true;
    } else {
        ilitek_data->has_df = false;
    }
    ret = ilitek_upgrade_2302or2312(df_startaddr, df_endaddr, df_checksum, ap_startaddr, ap_endaddr, ap_checksum, CTPM_FW);
    if (ret < 0) {
        tp_log_err("ilitek_upgrade_2302or2312 err ret = %d\n", ret);
        //goto Retry;
    }
}
```

B. 手勢、單按兩下喚醒功能

#define ILITEK_CLICK_WAKEUP 0 //按一下喚醒

#define ILITEK_DOUBLE_CLICK_WAKEUP 1 //按兩下喚醒

#define ILITEK_GESTURE_WAKEUP 2 //手勢喚醒，針對 FW 內部做手勢判斷的方式

開啟此功能要確保休眠時 TP 沒有下電

大尺寸有些客戶需要點擊喚醒或按兩下喚醒功能時需要如下設置

#define ILITEK_GESTURE ILITEK_CLICK_WAKEUP //點擊喚醒

#define ILITEK_GESTURE ILITEK_DOUBLE_CLICK_WAKEUP //按兩下喚醒

點擊喚醒功能，必須在點擊抬起後執行喚醒動作

按兩下喚醒參數說明：

#define DOUBLE_CLICK_DISTANCE 1000 //兩次點擊座標的最大距離

#define DOUBLE_CLICK_ONE_CLICK_USED_TIME 800 //一次點擊所用最長時間，單位 ms

#define DOUBLE_CLICK_NO_TOUCH_TIME 1000 //兩次點擊中間間隔時間，單位 ms

```
#define DOUBLE_CLICK_TOTAL_USED_TIME (DOUBLE_CLICK_NO_TOUCH_TIME + (DOUBLE_CLICK_ONE_CLICK_USED_TIME * 2)) //按兩下總時間
```

對於系統上層進行設置手勢功能是否啟用的介面在/sys/touchscreen/gesture，具體實現如下：

```
00191: #ifndef ILITEK_GESTURE
00192: static ssize_t ilitek_gesture_show(struct device *dev,
00193:     struct device_attribute *attr, char *buf) {
00194:     if (ilitek_data->enable_gesture) {
00195:         return sprintf(buf, "gesture: on\n");
00196:     } //若要获取手势状态则可读/sys/touchscreen/gesture这个节点
00197:     else {
00198:         return sprintf(buf, "gesture: off\n");
00199:     }
00200: }
00201: static ssize_t ilitek_gesture_store(struct device *dev,
00202:     struct device_attribute *attr, const char *buf, size_t size) {
00203:     if (buf[0]) {
00204:         ilitek_data->enable_gesture = true;
00205:     } //往/sys/touchscreen/gesture 写入第一byte数据不为0即认为启用手势功能,
00206:     else { //否则关闭手势功能
00207:         ilitek_data->enable_gesture = false;
00208:     }
00209:     return size;
00210: }
00211: static DEVICE_ATTR(gesture, S_IRWXUGO, ilitek_gesture_show, ilitek_gesture_store);
00212: #endif
00213:
```

若與客戶使用介面不對應，可修改此處以及節點路徑

C. 通過命令控制手套功能

1. cat /proc/ilitek/ enable_glove_mode
開啟手套模式
2. cat /proc/ilitek/ disable_glove_mode
關閉手套模式

D. ESD 檢測

巨集開關

```
#define ILITEK_ESD_PROTECTION //默認關閉
```

開啟此宏後會創建一個工作隊列，每間隔按照設定的時間來檢測 IC 是否有異常，若有異常則進行 reset（或者加入上下電），可修改 ilitek_data->esd_delay 來改變間隔時間

預設檢測方式為：下命令讀取資料（預設為 0x42 獲取 protocol），會 retry 3 次，若 3 次都失敗則 reset
具體實現如下：

```
00375: static void ilitek_esd_check(struct work_struct *work) {
00376:     int i = 0;
00377:     unsigned char buf[4]={0};
00378:     tp_log_info("enter.....\n");
00379:     if(ilitek_data->operation_protection){
00380:         tp_log_info("ilitek esd ilitek_data->operation_protection is true SO not check\n");
00381:         goto ilitek_esd_check_out;
00382:     }
00383:     ilitek_data->operation_protection = true;
00384:     buf[0] = ILITEK_TP_CMD_GET_PROTOCOL_VERSION;
00385:     if(ilitek_data->esd_check){
00386:         for (i = 0; i < 3; i++) {
00387:             if(ilitek_i2c_write_and_read(buf, 1, 0, buf, 2) < 0){
00388:                 tp_log_err("ilitek esd i2c communication error \n");
00389:                 if (i == 2) {
00390:                     tp_log_err("esd i2c communication failed three times reset now\n");
00391:                     break;
00392:                 }
00393:             }
00394:             else {
00395:                 if (buf[0] == 0x03) {
00396:                     tp_log_info("esd ilitek_ts_send_cmd successful, response ok\n");
00397:                     goto ilitek_esd_check_out;
00398:                 }
00399:                 else {
00400:                     tp_log_err("esd ilitek_ts_send_cmd successful, response failed\n");
00401:                     if (i == 2) {
00402:                         tp_log_err("esd ilitek_ts_send_cmd successful, response failed three times reset now\n");
00403:                         break;
00404:                     }
00405:                 }
00406:             }
00407:         } // end for i=0;i<3;i++ ?
00408:     } // end if ilitek_data->esd_check ?
00409: }
```

E. 通過命令升級固件

- `"/data/local/tmp/2511dftest.hex"`為對應路徑和檔案名，推薦路徑為`/data/local/tmp`這裡一般都會有許可權操作，這裡是使用 `hex` 檔來升級

- upgrade successfull ilitek firmware version is 6.0.0.0.1.2.255.255

[illegible]

```

if (!ilitek_data->ic_2120)) {
    ret = sscanf(buf, "%d,%d,%d,%d,%d,%d,%d,%d,%s",&ilitek_short_threshold,&ilitek_open_threshold,&ilitek_open_txdelatathreshold,&ilitek_open_rxdelatathreshold,&ilitek_allnode_max_threshold,&ilitek_allnode_min_threshold,&ilitek_allnode_testw1,&ilitek_allnode_testw2,&ilitek_allnodemodule,&ilitek_allnodetx,&ilitek_printsensorsorteddata,sensor_test_data_path);
    tp_log_info("ilitek_short_threshold = %d\n", ilitek_short_threshold);
    tp_log_info("ilitek_open_threshold = %d\n", ilitek_open_threshold);
    tp_log_info("ilitek_open_txdelatathreshold = %d\n", ilitek_open_txdelatathreshold);
    tp_log_info("ilitek_open_rxdelatathreshold = %d\n", ilitek_open_rxdelatathreshold);
    tp_log_info("ilitek_allnode_max_threshold = %d\n", ilitek_allnode_max_threshold);
    tp_log_info("ilitek_allnode_min_threshold = %d\n", ilitek_allnode_min_threshold);
    tp_log_info("ilitek_allnode_testw1 = %d\n", ilitek_allnode_testw1);
    tp_log_info("ilitek_allnode_testw2 = %d\n", ilitek_allnode_testw2);
    tp_log_info("ilitek_allnodemodule = %d\n", ilitek_allnodemodule);
    tp_log_info("ilitek_allnodetx = %d\n", ilitek_allnodetx);
}
else {
    ret = sscanf(buf, "%d,%d,%d,%d,%s",&ilitek_short_threshold,&ilitek_open_threshold,&ilitek_allnode_max_threshold,&ilitek_allnode_min_threshold,&ilitek_printsensorsorteddata,sensor_test_data_path);
    tp_log_info("ilitek_short_threshold = %d\n", ilitek_short_threshold);
    tp_log_info("ilitek_open_threshold = %d\n", ilitek_open_threshold);
    tp_log_info("ilitek_allnode_max_threshold = %d\n", ilitek_allnode_max_threshold);
    tp_log_info("ilitek_allnode_min_threshold = %d\n", ilitek_allnode_min_threshold);
}
tp_log_info("ilitek_printsensorsorteddata = %d\n", ilitek_printsensorsorteddata);
tp_log_info("sensor_test_data.path = %s\n",sensor_test_data_path);
}

```

非 ILI2120 卡控范围设定方式

ILI2120卡控范围设定方式

打印及显示flag

保存sensor test 数据的路径, 推荐/data/local/tmp/

- 代碼會判斷測試是否 pass 來丟出 pass 或 fail

```
if (short_test_result == 0 && open_test_result == 0 && allnode_test_result ==0) {
    seq_printf(m, "pass\n");
}
else {
    seq_printf(m, "fail\n");
}
```

G. NoiseFre 功能

1. 直接 `cat /proc/ilitek/noisefre_data` 這個節點即可（使用默認的參數），丟出資料如下

```
shell@msm8916_64:/ # cd /proc/ilitek
cd /proc/ilitek
shell@msm8916_64:/proc/ilitek # cat no*
cat no*
0300, 0305, 0310, 0315, 0320, 0325, 0330, 0335, 0340, 0345, 0350, 0355, 0360, 0365, 0370, 0375, 0380, 0385, 0390, 0395,
0005, 0005, 0005, 0005, 0005, 0004, 0005, 0005, 0005, 0005, 0005, 0004, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005,
0400, 0405, 0410, 0415, 0420, 0425, 0430, 0435, 0440, 0445, 0450, 0455, 0460, 0465, 0470, 0475, 0480, 0485, 0490, 0495,
0005, 0005, 0005, 0004, 0004, 0005, 0004, 0005, 0005, 0005, 0005, 0005, 0004, 0005, 0004, 0005, 0005, 0005, 0005, 0005,
0500, 0505, 0510, 0515, 0520, 0525, 0530, 0535, 0540, 0545, 0550, 0555, 0560, 0565, 0570, 0575, 0580, 0585, 0590, 0595,
0005, 0005, 0005, 0005, 0004, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0004, 0005, 0005, 0005, 0004, 0004, 0004, 0005,
0600, 0605, 0610, 0615, 0620, 0625, 0630, 0635, 0640, 0645, 0650, 0655, 0660, 0665, 0670, 0675, 0680, 0685, 0690, 0695,
0005, 0004, 0005, 0004, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0004, 0005, 0005, 0004, 0005, 0004, 0005,
0700, 0705, 0710, 0715, 0720, 0725, 0730, 0735, 0740, 0745, 0750, 0755, 0760, 0765, 0770, 0775, 0780, 0785, 0790, 0795,
0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0004, 0005, 0005, 0005, 0005, 0005, 0005, 0005,
0800, 0805, 0810, 0815, 0820, 0825, 0830, 0835, 0840, 0845, 0850, 0855, 0860, 0865, 0870, 0875, 0880, 0885, 0890, 0895,
0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0004, 0004, 0004, 0005, 0005, 0005, 0005, 0005, 0005, 0004, 0005, 0005,
0900, 0905, 0910, 0915, 0920, 0925, 0930, 0935, 0940, 0945, 0950, 0955, 0960, 0965, 0970, 0975, 0980, 0985, 0990, 0995,
0005, 0004, 0005, 0005, 0004, 0005, 0005, 0005, 0005, 0005, 0004, 0004, 0005, 0004, 0005, 0005, 0004, 0004, 0005, 0005,
1000, 1005, 1010, 1015, 1020, 1025, 1030, 1035, 1040, 1045, 1050, 1055, 1060, 1065, 1070, 1075, 1080, 1085, 1090, 1095,
0004, 0005, 0005, 0005, 0004, 0005, 0004, 0005, 0005, 0004, 0005, 0005, 0004, 0005, 0004, 0005, 0005, 0005, 0005, 0005, 0004,
1100, 1105, 1110, 1115, 1120, 1125, 1130, 1135, 1140, 1145, 1150, 1155, 1160, 1165, 1170, 1175, 1180, 1185, 1190, 1195,
0004, 0004, 0005, 0005, 0004, 0004, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005, 0005,
1200,
0005,
```

2. 修改起始和結束頻率以及跳動大小方式，操作如下：

```
echo 30,120,5,/data/local/tmp/ > /proc/ilitek/noisefre_data
```

30,120,5,/data/local/tmp/ ➔ 起始頻率，結束頻率，跳動大小，資料保存路徑

H. 針對中大尺寸 debug 資訊是 ASCII 的方式使用 debug 節點直接看 debug 資訊

1. `echo dbg_flag > /proc/ilitek_debug` //此命令用來打開或關閉此功能的 flag，每下一次是上一次的非動作
2. `cat /proc/ilitek_debug` 這命令執行下去後若有 debug 資訊就會印出來
3. 當不看時記得再下一次 `echo dbg_flag > /proc/ilitek_debug`

I. 獲取固件版本

直接下命令 `cat /proc/ilitek/firmware_version` 即可丟出如下資訊

```
ilitek firmware version is 6.0.0.0.1.2.255.255
```

對於客戶上層獲取版本號的需求可直接讀取此節點或 `/sys/touchscreen/firmware_version`

5. 常見問題

A. 驅動不會進入 probe 函數

對於使用 board file 方式註冊的查看 ILITEK_TS_NAME 和註冊的 I2C 設備名稱是否一致，此處必須一致

對於使用 dts 方式註冊的查看 of_match_table = ilitek_touch_match_table 中 ilitek_touch_match_table 內 compatible 是否和 dts 註冊中的 compatible 匹配，此處必須匹配

B. 通信不通

1. 軟體上的只有 I2C 匯流排號及位址會影響到通信，軟體配置確保這兩項是 OK 的
2. 硬體上首先確認 IC 電這塊是否 OK
3. 抓取波形確認是否滿足通信協議
4. 用其他器件通信是否 OK，可嘗試把此匯流排上的其他設備都卸掉測試

C. 報點問題

1. 有觸摸效果，只是座標 mapping 問題
 - i. X、Y 需要交換 → 將 ILITEK_ROTATE_FLAG 設定值由 0 改為 1 或由 1 改為 0
 - ii. X、Y 值要做鏡像變化即最大變最小 → 將 ILITEK_REVERT_X 或 ILITEK_REVERT_Y 的設定值由 0 改為 1 或由 1 改為 0
 - iii. 若需要使用顯示幕的解析度，則開啟#define ILITEK_USE_LCM_RESOLUTION 這個宏，同時將 TOUCH_SCREEN_X_MAX 和 TOUCH_SCREEN_Y_MAX 設為正確值
2. 觸摸沒反應
 - i. 確認中斷是否註冊成功，同步確認中斷號是否正確
 - ii. 通過 log 確認觸摸時是否有中斷回應，若有中斷回應，則可將 ilitek_read_data_and_report_3XX 內收到的資料列印出來看資料是否正確
 - iii. 抓取觸摸時 INT 的波形確認是否有正常拉高拉低的動作

Revision History

Version No.	Date	Page	Description
0.0.1	2011/03/07	All	Firstly release
0.0.2	2011/05/12	3	Modified driver file name.
0.0.3	2011/09/30	3	Modified version id
0.0.4	2012/11/26	3	Method of adding idc files
0.0.5	2017/07/14	15	Modified driver structure
0.0.0.6	2017/09/12	16	<ol style="list-style-type: none"> 1. MTK 平臺支援非 dts 方式 2. 針對中大尺寸添加 debug 資訊節點 3. 升級前查看 hex 檔是否匹配 4. 針對 Intel 平臺添加對應 match 方式
0.0.0.7	2019/5/7	14	<ol style="list-style-type: none"> 1. 拿掉 ILI2120 功能 2. 新增 glove mode 開關