

# Linux I2C Touch Device Driver

Version: V0\_0\_1\_4  
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

1. 總體描述.....	4
2. 標頭檔中相關宏的說明.....	6
3. 部分代碼說明.....	9
4. 相關功能說明.....	10
5. 讀取 FW Key 按鍵資訊流程.....	13
6. 常見問題.....	15

## Revision History

Version No.	Date	Page	Description
0.0.1	2011.3.7	All	Firstly release
0.0.2	2011.5.12	3	Modified driver file name.
0.0.3	2011.9.30	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	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	1. 拿掉 ILI2120 功能 2. 新增 glove mode 開關
0_0_0_8	2019.11.29	14	1. 新增 sensor test 卡控讀 INI 檔方式
0_0_0_9	2020.5.8	14	1. 新增 chrome_os define 2. 新增支援 Lego 系列 IC 3. 新增 V6 開機升級
0_0_1_0	2020.7.31	15	1. 新增 Low power 設定 2. 新增 Raw data 長度設定 3. 新增 FW 更新長度設定
0_0_1_2	2022.5.6	All	1. 修改文件模板 2. 更新開機更新 FW 說明 3. 更新手勢喚醒使用說明
0_0_1_3	2022.11.23	All	1. 更新手勢喚醒使用說明 2. 更新 Firmware 模式使用說明 3. 更新 ESD 檢測使用說明 4. 更新 OTA 開機更新使用說明 5. 更新 Low Power 模式使用說明
0_0_1_4	2023.6.14	5,6,7	1. 移除使用 ILITEK_KERNEL_DRIVER flag 2. 增加 config 說明

## 1. 總體描述

### A. 簡介

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這個宏。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	<b>Protocol V3 ICs:</b> 2312、2315、2510、2511、2712 <b>Protocol V6 ICs:</b> 2520、2521、2322、2323、2316、2326、2130、2131、2132、2900、2901、2910、2911、2531、2532
I2C 設備位址（7位元）	0x41
開機自動升級	支援
支援平臺	Qcom (default)、Rockchip、MTK、Allwinner、Amlogic

### B. 各文件說明如下：

- ilitek\_ts.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、註冊輸入裝置資訊，報點，休眠喚醒處理等
- ilitek\_update.c:  
FW固件更新的具體實現
- ilitek\_tool.c:  
用於支持Linux\_Daemon\_ITS tool 以及創建用命令升級固件、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:  
ILITEK protocol 相關流程
- ilitek\_report.c:  
處理收報點資料與解碼相關流程
- ilitek\_crypto.c:  
處理資料加解密相關流程

### C. 驅動移植步驟說明：

- 將 <Linux\_I2CDriver\_ITS> 資料夾複製到 /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) 下面
- 在 kernel/drivers/input/touchscreen/Makefile裡面添加一行如下  
obj-y += <Linux\_I2CDriver\_ITS 資料夾名稱>/
- 添加 I2C 設備：

#### ✓ 使用 board file 的方式：

找到 kernel中初始化 I2C 匯流排的板級檔，範例路徑如：

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
```

#### ✓ 使用 device tree 註冊的方式：

在 dts 檔對應 I2C bus 節點下加入如下參考內容，粗體紅字為使用者須修改至相對應的 GPIO 參數。

```
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

#define DRIVER_VERSION_0	5
#define DRIVER_VERSION_1	9
#define DRIVER_VERSION_2	2
#define DRIVER_VERSION_3	0
#define CUSTOMER_H_ID	0
#define CUSTOMER_L_ID	0
#define TEST_VERSION	0

### ✓ 平臺設定:

選擇對應的平臺賦給 ILITEK\_PLAT 這個宏，亦可選擇 QCOM 為預設，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 <b>ILITEK_PLAT</b>	<b>ILITEK_PLAT_QCOM</b>

### ✓ #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\_CHECK\_ENABLE 0

ESD 保護開關，**預設關閉**，設定為 0，表示關閉，設定為 1，表示啟動  
(預設不啟用，如欲啟用，請與 ILITEK/代理商確認)

### ✓ #define ILITEK\_TOUCH\_PROTOCOL\_B

報點協定使用 B 類報點，**默認開啟**

### ✓ #define ILITEK\_REPORT\_PRESSURE

註冊 input device pressure 開關，**默認關閉**

- ✓ **#define MTK\_UNDTS**  
針對 MTK 平臺使用非 device tree 方式需要開啟此巨集
- ✓ **#define ILITEK\_USE\_MTK\_INPUT\_DEV**  
針對 MTK 平臺使用平臺內的 tpd->dev，**預設開啟**
- ✓ **#define ILITEK\_USE\_LCM\_RESOLUTION**  
使用 LCM 的解析度，**默認關閉**，當 MTK 平臺有開啟 ILITEK\_USE\_MTK\_INPUT\_DEV 也會啟動這個宏，LCM 解析度由下列宏調整  
**(預設不啟用，如欲啟用，請與 ILITEK/代理商確認)**

✓	<b>#define TOUCH_SCREEN_X_MAX</b>	<b>(1080)</b>	<b>//LCD_WIDTH</b>
✓	<b>#define TOUCH_SCREEN_Y_MAX</b>	<b>(1920)</b>	<b>//LCD_HEIGHT</b>
- ✓ **#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  
**(預設不啟用，如欲啟用，請與 ILITEK/代理商確認)**
- ✓ **#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**，非 0 值則啟用  
**(預設不啟用，如欲啟用，請與 ILITEK/代理商確認)**
- ✓ **#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**，非 0 值則啟用  
**(預設不啟用，如欲啟用，請與 ILITEK/代理商確認)**
- ✓ **#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\_GESTURE\_DEFAULT                      Gesture\_Disable

手勢喚醒系統預設值設定，請將粗體字改為如下選項

```
#define ILITEK_GESTURE_TYPES      \
    X(Disable,          0, "disable")  \    關閉手勢喚醒功能
    X(Single_Click,    1, "single-click") \    單擊喚醒 (driver 實現)
    X(Double_Click,   2, "double-click") \    雙擊喚醒 (driver 實現)
```

雙擊喚醒相關參數如下

```
#define DOUBLE_CLICK_DISTANCE      1000
    雙擊兩點之間的距離設定
#define DOUBLE_CLICK_ONE_CLICK_USED_TIME  800
    第一次點擊所用最長時間，單位為 ms
#define DOUBLE_CLICK_NO_TOUCH_TIME  1000
    兩次點擊時間間隔，單位為 ms
```

✓ #define ILITEK\_LOW\_POWER\_DEFAULT                      Low\_Power\_Sleep

系統休眠模式設定，請將粗體字改為如下選項，

**(Idle 模式須搭配手勢喚醒功能並與代理商/ILITEK 確認該 firmware 支援 IDLE)**

```
#define ILITEK_LOW_POWER_TYPES    \
    X(Sleep,          0, "sleep")    \    使用 ILITEK 休眠模式
    X(Idle,           1, "idle")     \    使用 ILITEK 省電模式，需搭配手勢喚醒功能
    X(PowerOff,      2, "poweroff") \    系統休眠使用斷電方式
```

### 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，若是有其他方式請按具體方式修改
- ✓ `void ilitek_reset(int delay_ms)`  
IC HW reset 函數，`delay_ms` 為 reset 後的延遲時間，單位為 ms，需要大於 IC 的初始化時間。
- ✓ `int api_update_ts_info(void *handle)`  
讀取 IC 相關資訊函數，如固件版本、報點解析度、IC 模組資訊、固件模式等等。
- ✓ `int ilitek_request_irq(void)`  
註冊 ISR 函數入口，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
- ✓ `ISR_FUNC(ilitek_i2c_isr)`  
中斷執行函式入口，報點處理與系統輸入裝置事件回報。
- ✓ `void ilitek_suspend(void) / void ilitek_resume(void)`  
裝置休眠與喚醒處理函式入口，處理系統休眠與喚醒時的裝置設定行為。
- ✓ `int ilitek_upgrade_firmware(char *filename)`  
firmware 固件更新函數總入口，支援.bin/.hex 檔案。

## 4. 相關功能說明

### A. 開機升級 firmware 固件功能 (OTA)

#### ✓ 事前準備:

需先在 Makefile 加入 ILITEK\_BOOT\_UPDATE flag，如下，請依以下不同檔案類型進行環境準備。

**ccflags-y += -DILITEK\_BOOT\_UPDATE**

#### ● .ili 檔案

需將 ilitek\_fw.h 與 .ili 檔案置於 driver 同路徑下，檔案由代理商或是 ILITEK 提供。

#### ● .bin or .hex 檔案

需確認開啟 kconfig “CONFIG\_FW\_LOADER=y”，並且事先將 .bin or .hex 檔案置於 kernel 設定之路徑下，請參考 /drivers/base/firmware\_class.c 之 fw\_path 參數設定

※ kernel v4.17 版本以後請參考 /drivers/base/firmware\_loader/main.c

#### ✓ 判定是否進行 firmware 固件更新條件:

依不同檔案類型如下，若判斷 IC 為 bootloader 模式時，則強制進行 firmware 固件更新。

#### ● .ili 檔案

判斷 8 byte firmware 固件版本，由高位判斷至低位，若檔案之固件版本高於目前固件版本，則進行更新，詳情請參考 “decode\_ili(…)” API。

#### ● .bin or .hex 檔案

不比較 firmware 固件版本，直接進行固件更新。

### B. 手勢喚醒功能 (Gesture WakeUp)

#### ✓ 此功能開啟後，預設為系統休眠時不斷電，請確保系統休眠時 TP 不斷電

#### ✓ 事前準備:

設定預設手勢喚醒模式 ILITEK\_GESTURE\_DEFAULT 宏為單擊或是雙擊模式，預設關閉，相關設定參數請見宏定義。

#### ✓ 使用者亦可在系統運行期間透過指令更改喚醒模式。

#### ● 確認目前模式: `cat /sys/bus/i2c/devices/<device name>/gesture`

```
rpi4:/data/local/tmp # cat /sys/bus/i2c/devices/1-0041-1/gesture
[disable] single-click double-click
```

#### ● 關閉手勢喚醒: `echo disable > /sys/bus/i2c/devices/<device name>/gesture`

#### ● 開啟單擊喚醒: `echo single-click > /sys/bus/i2c/devices/<device name>/gesture`

#### ● 開啟雙擊喚醒: `echo double-click > /sys/bus/i2c/devices/<device name>/gesture`

### C. 切 firmware 模式功能:

- 請先跟代理商/ILITEK確認該版firmware固件是否支援此功能
- 列出支援模式數目與確認目前模式: `cat /proc/ilitek/func_mode`

```
rpi4:/data/local/tmp # cat /proc/ilitek/func_mode
function mode: [0] 1 2
```

- 切入 firmware mode 0: `echo 0 > /proc/ilitek/func_mode`
- 切入 firmware mode 1: `echo 1 > /proc/ilitek/func_mode`
- 切入 firmware mode 2: `echo 2 > /proc/ilitek/func_mode`

### D. ESD 檢測

#### ✓ 功能說明:

功能啟動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（或者自行加入上下電），可修改 "ts->esd\_delay" 來改變間隔時間，預設檢測方式為下命令讀取 IC Protocol 版本。

#### ✓ 使用方式:

- 若需要預設啟動，請將 ILITEK\_ESD\_CHECK\_ENABLE 宏設定為 1，預設為 0，亦可在 driver 掛載後手動啟動/關閉如下。
- 確認目前啟動/關閉狀態: `cat /sys/bus/i2c/devices/<device name>/esd_check`

```
rpi4:/data/local/tmp # cat /sys/bus/i2c/devices/1-0041-1/esd_check
enable [disable]
```

- 啟動 ESD 檢測功能: `echo enable > /sys/bus/i2c/devices/<device name>/esd_check`
- 關閉 ESD 檢測功能: `echo disable > /sys/bus/i2c/devices/<device name>/esd_check`

### E. 手動 Firmware 固件升級

#### ✓ 事前準備:

- 請先確認 OS image 有開啟 kconfig "CONFIG\_FW\_LOADER=y"。
- 將 .hex 或 .bin 固件置於 kernel 特定路徑，請查 /driver/base/firmware\_class.c 之 fw\_path 參數設定。  
(kernel v4.17 版本以後請參考 /drivers/base/firmware\_loader/main.c)
- 亦可透過如下指令開通特定路徑 (ex. 開通 /data/local/tmp 路徑下之固件)。  
`echo -n "/data/local/tmp" > /sys/module/firmware_class/parameters/path`
- 須將固件命名為 ilitek.hex 或 ilitek.bin，亦可透過軟連結建立，如下。

```
rpi4:/data/local/tmp # ln -s /data/local/tmp/ILI2901_7.80.0.0_0.0.0.2.hex ilitek.hex
```

#### ✓ 執行 firmware 固件更新方式:

- 透過 procfs: `cat /proc/ilitek/update_fw`
- 透過 sysfs: `cat /sys/bus/i2c/devices/<device name>/update_fw`
- 成功後會顯示升級成功以及更新後的固件版本，如下:

```
rpi4:/data/local/tmp # cat /proc/ilitek/update_fw
upgrade success, fw version: 07-00-00-00-00-00-00-00
```

## F. 系統休眠模式設定

- ✓ 如欲調整預設模式，可以設定宏 `ILITEK_LOW_POWER_DEFAULT`，相關設定參數請見宏定義，亦可在 driver 掛載後手動調整，如下。
- ✓ 查看支援模式: `cat /sys/bus/i2c/devices/<device name>/low_power`  

```
rpi4:/data/local/tmp # cat /sys/bus/i2c/devices/1-0041-1/low_power
[sleep] idle poweroff
```
- ✓ 設定為 Sleep 模式: `echo sleep > /sys/bus/i2c/devices/<device name>/low_power`
- ✓ 設定為 Idle 模式: `echo idle > /sys/bus/i2c/devices/<device name>/low_power`  
**(Idle 模式須搭配手勢喚醒功能，並與代理商/ILITEK 確認該版 firmware 支援該模式)**

## G. 查看 Firmware 固件版本

- ✓ 透過 procs: `cat /proc/ilitek/firmware_version`
- ✓ 透過 sysfs: `cat /sys/bus/i2c/devices/<device name>/firmware_version`

```
rpi4:/data/local/tmp # cat /proc/ilitek/firmware_version
fw-version-tag: [07-00-00-00-00-00-00]
```

## H. 查看 Module Name

- ✓ 透過 sysfs: `cat /sys/bus/i2c/devices/<device name>/product_id`

```
rpi4:/data/local/tmp # cat /sys/bus/i2c/devices/1-0041-1/product_id
module-name-tag: [ILI2510000000000]
```

## I. Console 功能

- ✓ 指令格式: `<write bytes length> <read bytes length> <cmd %x-%x-%x>`
- ✓ 讀取與寫入等資訊顯示在 kernel log 中，如下
  - 只有寫入指令: `echo 3 0 f0-0-0 > /proc/ilitek/console`  

```
[13927.228790] [ILITEK][MSG][ilitek_console_write:500] [write]: f0-00-00, len: 3
```
  - 寫入並讀取指令: `echo 1 3 42 > /proc/ilitek/console`

```
[13938.241406] [ILITEK][MSG][ilitek_console_write:500] [write]: 42, len: 1
[13938.242916] [ILITEK][MSG][ilitek_console_write:506] [read]: 06-00-04, len: 3
```

## J. 開啟偵錯 debug log

- ✓ 開啟 debug log: `echo dbg_debug > /proc/ilitek_ctrl`
- ✓ 關閉 debug log: `echo dbg_info > /proc/ilitek_ctrl`
- ✓ 此功能主要可看到處理報點時的 debug log，可在 Kernel Log 中找關鍵字 "[ILITEK]"。

## 5. 讀取 FW Key 按鍵資訊流程

- ✓ 針對 V3 與 V6 系列 IC，如下，讀取 key info 的方式有所不同，詳細流程將透過 console 功能描述如下。

V3	For IC: ILI2312/ 2315/ 2510/ 2511/ 2712
V6	For IC: ILI2316/ 2326/ 2130/ 2131/ 2132/ 2520/ 2521/ 2322/ 2323/ 2900/ 2901/ 2910/ 2911/ 2531/ 2532

### ✓ V3 系列 IC:

- 先透過 panel Info. cmd: 0x20 讀取 FW key 數目，寫入 1 byte 0x20，並讀取 15 bytes 資料，FW key 數目位於讀取的**第 9 byte**

- `echo 1 15 20 > /proc/ilitek/console`

```
[write]: 20, len: 1
[read]: 00-20-00-20-1a-0d-0a-0a-0a-ff-ff-ff-ff-ff-01, len: 15
```

- 透過 key info. cmd: 0x22，讀取 key 資訊，寫入 1 bytes 0x22，並讀取 29 bytes 資料，若 key 數目大於 5，需要再次讀取 25 bytes，如下。

- `echo 1 29 22 > /proc/ilitek/console && echo 0 25 0 > /proc/ilitek/console`

```
[write]: 22, len: 1
[read]: 01-00-01-00-72-02-bc-20-64-73-05-78-20-64-66-08-34-20-64-74-0a-f0-20-64-02-0d-ac-20-64, len: 29
wlen: 0, rlen: 25, command: 0
[read]: 02-10-68-20-64-02-13-24-20-64-02-15-e0-20-64-02-18-9c-20-64-02-1b-58-20-64, len: 25
```

- 前 29 bytes 與後 25 bytes 格式如下。

前 29 bytes	0	Key 範圍 X 長度	Key 範圍 X 長度(15:8)
	1		Key 範圍 X 長度(7:0)
	2	Key 範圍 Y 長度	Key 範圍 Y 長度(15:8)
	3		Key 範圍 Y 長度(7:0)
	4	Key 0	Key ID
	5		X 起始位置 (15:8)
	6		X 起始位置 (7:0)
	7		Y 起始位置(15:8)
	8		Y 起始位置(7:0)
	9~13	Key 1	格式同 Key 0
	14~18	Key 2	格式同 Key 0
	19~23	Key 3	格式同 Key 0
	24~28	Key 4	格式同 Key 0
後 25 bytes	0~4	Key 5	格式同 Key 0
	5~9	Key 6	格式同 Key 0
	10~14	Key 7	格式同 Key 0
	15~19	Key 8	格式同 Key 0
	20~24	Key 9	格式同 Key 0

✓ V6 系列 IC:

1. 先透過 panel Info. cmd: 0x20 讀取 FW key 數目，寫 1 bytes 0x20，並讀取 15 bytes 資料，FW key 數目位於讀取的第 10 byte

- 指令: `echo 1 15 20 > /proc/ilitek/console`

```
wlen: 1, rlen: 15, command: 20
[write]: 20, len: 1
[read]: 00-40-00-40-1a-00-0d-00-0a-04-01-01-00-01-05, len: 15
```

2. 透過 key info. cmd: 0x22，讀取 key 資訊，寫 1 byte 0x22，並讀取 (5 + 5 \* key 數目) bytes 資料 (ex. 若 FW key 數量為 4，則讀取 5 + 5 \* 4 = 25 bytes 資料)。

- 指令: `echo 1 25 22 > /proc/ilitek/console`

```
wlen: 1, rlen: 25, command: 22
[write]: 22, len: 1
[read]: 02-00-01-00-01-74-b0-04-64-40-72-60-09-64-40-73-10-0e-64-40-66-c0-12-64-40, len: 25
```

- 資料格式如下。

0	Key 模式	1: HW key, 2: HSW key, 3: Virtual key, 其他值: Disable
1	Key 範圍 X 長度	Key 範圍 X 長度(7:0)
2		Key 範圍 X 長度(15:8)
3	Key 範圍 Y 長度	Key 範圍 Y 長度(7:0)
4		Key 範圍 Y 長度(15:8)
5	Key 0	Key ID
6		X 起始位置(7:0)
7		X 起始位置(15:8)
8		Y 起始位置(7:0)
9		Y 起始位置(15:8)
10~14	Key 1	格式同 Key 0
...	...	...
5 + 5 * N	Key N	格式同 Key 0

## 6. 常見問題

### A. 驅動無法進入 probe 函數

- 對於使用 board file 註冊方式  
請確認 ITEK\_TS\_NAME 和註冊的 I2C 設備名稱是否一致。
- 對於使用 device tree 註冊方式  
請確認 ilitek\_touch\_match\_table 內 compatible 是否和 dts 的 compatible 匹配。
- 請確認系統有建立 i2c device 裝置與相關節點  
請確認 /sys/bus/i2c/devices 下有裝置節點 (<i2c-bus-number>-0041)，如下。

```
ls /sys/bus/i2c/devices/
0-001b 0-0041 4-0022 4-0041-1 i2c-1 i2c-3 i2c-9
0-0040 1-001c 4-0041 i2c-0 i2c-10 i2c-4
```

### B. I2C 通信異常、IC 資訊異常

- 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。
- 硬體上首先確認 IC 電這塊是否 OK。  
ILI213X 單電源  
VDD、VDDIO 的部分需要平台提供 2.97V 到 3.4V  
ILI213X 雙電源  
VDD 的部分需要平台提供 2.97V 到 3.4V，VDDIO 的部分需要平台提供 1.8V  
ILI231X/ILI251X 單電源  
VDD、VDDIO 的部分需要平台提供 3.0V 到 3.6V  
ILI231X/ILI251X 雙電源  
VDD 的部分需要平台提供 3.0V 到 3.6V，VDDIO 的部分需要平台提供 1.8V  
如有使用 Regulator Voltage 來設定電壓，要做相應的修改
- 透過邏輯分析儀或是示波器抓取波形確認是否滿足通信協議。
- 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

- 有觸摸效果，只是座標 mapping 問題。
  - X、Y 需要交換。  
將 ILITEK\_ROTATE\_FLAG 設定值由 0 改為 1 或由 1 改為 0。
  - X、Y 值要做鏡像變化。  
將 ILITEK\_REVERT\_X 或 ILITEK\_REVERT\_Y 的設定值由 0 改為 1 或由 1 改為 0。
  - 若需要指定顯示幕的解析度，則開啟 ILITEK\_USE\_LCM\_RESOLUTION 宏，同時將 TOUCH\_SCREEN\_X\_MAX 和 TOUCH\_SCREEN\_Y\_MAX 設為正確值。

## 2. 觸摸沒反應

- 確認中斷是否註冊成功，同步確認中斷號是否正確。
- 請確認系統有收到 TP 中斷，檢查如下結果中斷次數是否在觸屏時有增加。

```
rk3399_firefly_mid:/data/local/tmp # cat /proc/interrupts | grep ilitek
110:          776          0          0          0          0          0      gpio1  7 Edge    ilitek_touch_irq
```

- 開啟偵錯 debug log 確認觸摸時是否有進中斷處理，若有中斷回應，請確認打印資料是否正確。
- 抓取觸摸時 INT 的波形確認是否有正確，請參考 spec。

## 3. 飛線問題

- 如果是使用 Lego series IC 請檢查中斷觸發設定是否為 Rising Trigger，在 Driver 版本 5.9.0.3 之後的版本已經將 Lego series 改成 Rising trigger