ML56-TK Linux 模組驅動

NuMicro® 64/32位系列微控制器範例代碼介紹

文件資訊

應用簡述	本範例代碼說明如何在 Linux 核心下導入 ML56-TK 驅動
BSP 版本	Linux-5.4.x
開發平台	NuMaker-Base-MA35D1B1 V2.0

1. 概述

本範例代碼示範如何將 ML56-TK 核心模組引入 Linux，並透過 “input_test” 測試程式解碼輸入事件成人類可讀取的格式。

為了讓使用者能簡單的橋接 ML56-TK 至應用程式，ML56-TK 驅動將 ML56-TK 作為輸入裝置引入 Linux，且將觸控鍵視為是一鍵盤並回報鍵值(KEY_A, KEY_B…等)以及按鍵狀態(按下/放開) 至系統，使用者可透過設備樹定義欲回報的鍵值。而觸控條/觸控滑輪則視為是一觸控螢幕並轉換原始數據為 X-Y 的座標回報至系統，X 值代表百分比、Y 值則用來識別此原始數據是來自觸控條或觸控滑輪。

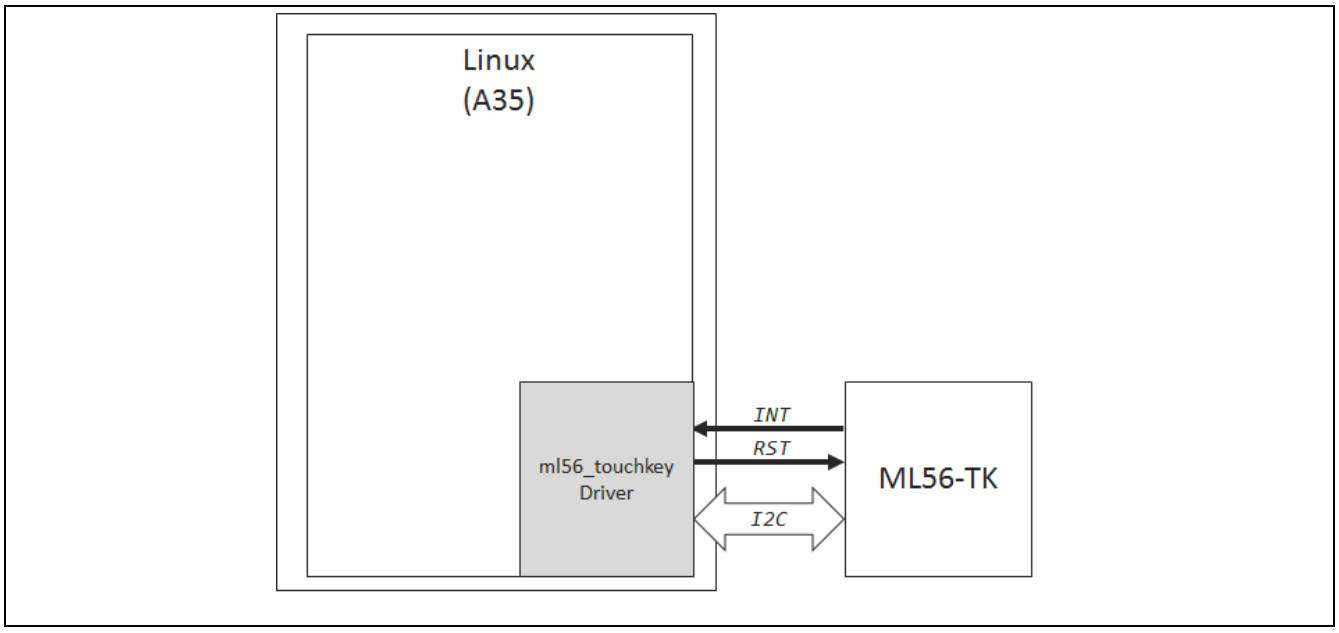

圖 1-1 ML56-TK Linux 驅動架構

1.1 驅動設定

ML56-TK 核心模組 I2C 與多功能腳位 Device Tree 的設定參考如下。

1.1.1 I2C 設定

ML56-TK 核心模組由 gpio 驅動 i2c protocol，device tree 設定如下。

```
i2c_gpio0: i2c-gpio-0 {
    reg = <0 0 0 0>;
    compatible = "i2c-gpio";
    gpios = <&gpiom 14 0 /* sda */
      &gpiom 15 0 /* scl */
     >;
    i2c-gpio,delay-us = <20>;        /* ~100 kHz */
    #address-cells = <1>;
    #size-cells = <0>;
    status = "disabled";
};
```

1.1.2 多功能腳位設定

ML56-TK 核心模組多功能腳位，device tree 設定如下。

```
&i2c_gpio0 {
    status = "okay";
   ...
    ml56-tk@52 {
        status = "okay";
        pinctrl-names = "default";
        pinctrl-0 = <&pinctrl_i2c_gpio0>;
        compatible = "nuvoton,ml56-tk";
        reg = <0x52>;
        reset-gpios = <&gpiom 1 GPIO_ACTIVE_LOW>;
        interrupt-parent = <&gpiok>;
        interrupts = <8 IRQ_TYPE_LEVEL_LOW>;
        irq-gpios = <&gpiok 8 0x00>;
        ml56-tk,keymap = <10 KEY_A>, <11 KEY_B>;
    };
};

/* Add ml56 module pin configuration to device tree's pinctl as below */

&pinctrl {
....
    pcfg_ml56_interrupt: pcfg-pcfg_ml56_interrupt {
     slew-rate = <0>; /* 0:normal, 1:high */
     input-schmitt-disable;  /* input-schmitt-disable, input-schmitt-enable */
     bias-pull-up; /* bias-disable, bias-pull-up, bias-pull-down  */
     power-source = <3300>; /* 1800:1.8v , 3300:3.3v */
     drive-strength = <3>; /* 0 ~ 7 */
     };
```

```
    pcfg_ml56_scl: pcfg-pcfg_ml56_sck {
     slew-rate = <0>;         /* 0:normal, 1:high */
     input-schmitt-disable;   /* input-schmitt-disable, input-schmitt-enable */
     bias-disable;            /* bias-disable, bias-pull-up, bias-pull-down  */
     power-source = <3300>;   /* 1800:1.8v , 3300:3.3v */
     drive-strength = <3>;    /* 0 ~ 7 */
    };

    pcfg_ml56_sda: pcfg-pcfg_ml56_sda {
     slew-rate = <0>;         /* 0:normal, 1:high */
     input-schmitt-disable;   /* input-schmitt-disable, input-schmitt-enable */
     bias-disable;            /* bias-disable, bias-pull-up, bias-pull-down  */
     power-source = <3300>;   /* 1800:1.8v , 3300:3.3v */
     drive-strength = <3>;    /* 0 ~ 7 */
    };

    pcfg_ml56_reset: pcfg-pcfg_ml56_reset {
     slew-rate = <0>;         /* 0:normal, 1:high */
     input-schmitt-disable;   /* input-schmitt-disable, input-schmitt-enable */
     bias-pull-up;            /* bias-disable, bias-pull-up, bias-pull-down  */
     power-source = <3300>;   /* 1800:1.8v , 3300:3.3v */
     drive-strength = <3>;    /* 0 ~ 7 */
    };

    i2c_gpio0 {
       pinctrl_i2c_gpio0: i2c_gpio0grp {
      nuvoton,pins =
       <SYS_GPK_MFPH_PK8MFP_GPIO       &pcfg_ml56_interrupt>,
       <SYS_GPM_MFPL_PM1MFP_GPIO       &pcfg_ml56_reset>,
       <SYS_GPM_MFPH_PM15MFP_GPIO      &pcfg_ml56_scl>,
       <SYS_GPM_MFPH_PM14MFP_GPIO      &pcfg_ml56_sda>;
       };
    };
};
```

1.2 執行結果

本範例代碼執行的結果如圖 1-2 所示

```
Wed Mar  2 03:50:39 2022.437039 type 0x0001; code 0x0030; value 0x00000001; Key 48 (0x30) press
Wed Mar  2 03:50:39 2022.437039 type 0x0000; code 0x0000; value 0x00000000;
Wed Mar  2 03:50:39 2022.657376 type 0x0001; code 0x0030; value 0x00000000; Key 48 (0x30) release
Wed Mar  2 03:50:39 2022.657376 type 0x0000; code 0x0000; value 0x00000000;
Wed Mar  2 03:50:41 2022.888186 type 0x0001; code 0x001e; value 0x00000001; Key 30 (0x1e) press
Wed Mar  2 03:50:41 2022.888186 type 0x0000; code 0x0000; value 0x00000000;
Wed Mar  2 03:50:42 2022.107135 type 0x0001; code 0x001e; value 0x00000000; Key 30 (0x1e) release
Wed Mar  2 03:50:42 2022.107135 type 0x0000; code 0x0000; value 0x00000000;

Wed Mar  2 03:51:20 2022.610418 type 0x0001; code 0x014a; value 0x00000001; Button 74 press
Wed Mar  2 03:51:20 2022.610418 type 0x0003; code 0x0000; value 0x00000004; Absolute X 4
Wed Mar  2 03:51:20 2022.610418 type 0x0003; code 0x0001; value 0x00000000; Absolute Y 0
```

```
Wed Mar  2 03:51:20 2022.807639 type 0x0003; code 0x0000; value 0x00000061; Absolute X 97
Wed Mar  2 03:51:20 2022.898640 type 0x0001; code 0x014a; value 0x00000000; Button 74 release

Wed Mar  2 03:53:18 2022.847879 type 0x0001; code 0x014a; value 0x00000001; Button 74 press
Wed Mar  2 03:53:18 2022.847879 type 0x0003; code 0x0000; value 0x00000000; Absolute X 0
Wed Mar  2 03:53:18 2022.847879 type 0x0003; code 0x0001; value 0x00000002; Absolute Y 2
Wed Mar  2 03:53:19 2022.544309 type 0x0003; code 0x0000; value 0x00000050; Absolute X 80
Wed Mar  2 03:53:19 2022.644725 type 0x0001; code 0x014a; value 0x00000000; Button 74 release
```

圖 1-2 代碼執行結果

2. 代碼介紹

“input_test” 讀取 /dev/input/event* 並且轉換事件為人類可讀的格式。

```
if (argc > 1) {
        sprintf(name, "/dev/input/event%d", atoi(argv[1]));
        if ((fd = open(name, O_RDWR, 0)) >= 0) {
            printf("%s: open, fd = %d\n", name, fd);
            for (i = 0; i < LED_MAX; i++) {
                event.time.tv_sec  = time(0);
                event.time.tv_usec = 0;
                event.type         = EV_LED;
                event.code         = i;
                event.value        = 0;
                write(fd, &event, sizeof(event));
            }

            while ((rc = read(fd, &event, sizeof(event))) > 0) {
                printf("%-24.24s.%06lu type 0x%04x; code 0x%04x;"
                       " value 0x%08x; ",
                       ctime(&event.time.tv_sec),
                       event.time.tv_usec,
                       event.type, event.code, event.value);
                switch (event.type) {
                case EV_KEY:
                    if (event.code > BTN_MISC) {
                        printf("Button %d %s",
                               event.code & 0xff,
                               event.value ? "press" : "release");
                    } else {
                        printf("Key %d (0x%x) %s",
                               event.code & 0xff,
                               event.code & 0xff,
                               event.value ? "press" : "release");
                    }
                    break;
                case EV_REL:
                    switch (event.code) {
                    case REL_X:      tmp = "X";       break;
                    case REL_Y:      tmp = "Y";       break;
                    case REL_HWHEEL: tmp = "HWHEEL";  break;
                    case REL_DIAL:   tmp = "DIAL";    break;
                    case REL_WHEEL:  tmp = "WHEEL";   break;
                    case REL_MISC:   tmp = "MISC";    break;
                    default:         tmp = "UNKNOWN"; break;
                    }
                    printf("Relative %s %d", tmp, event.value);
                    break;
                case EV_ABS:
                    switch (event.code) {
                    case ABS_X:        tmp = "X";        break;
                    case ABS_Y:        tmp = "Y";        break;
                    case ABS_Z:        tmp = "Z";        break;
```

```
                    case ABS_RX:       tmp = "RX";       break;
                    case ABS_RY:       tmp = "RY";       break;
                    case ABS_RZ:       tmp = "RZ";       break;
                    case ABS_THROTTLE: tmp = "THROTTLE"; break;
                    case ABS_RUDDER:   tmp = "RUDDER";   break;
                    case ABS_WHEEL:    tmp = "WHEEL";    break;
                    case ABS_GAS:      tmp = "GAS";      break;
                    case ABS_BRAKE:    tmp = "BRAKE";    break;
                    case ABS_HAT0X:    tmp = "HAT0X";    break;
                    case ABS_HAT0Y:    tmp = "HAT0Y";    break;
                    case ABS_HAT1X:    tmp = "HAT1X";    break;
                    case ABS_HAT1Y:    tmp = "HAT1Y";    break;
                    case ABS_HAT2X:    tmp = "HAT2X";    break;
                    case ABS_HAT2Y:    tmp = "HAT2Y";    break;
                    case ABS_HAT3X:    tmp = "HAT3X";    break;
                    case ABS_HAT3Y:    tmp = "HAT3Y";    break;
                    case ABS_PRESSURE: tmp = "PRESSURE"; break;
                    case ABS_DISTANCE: tmp = "DISTANCE"; break;
                    case ABS_TILT_X:   tmp = "TILT_X";   break;
                    case ABS_TILT_Y:   tmp = "TILT_Y";   break;
                    case ABS_MISC:     tmp = "MISC";     break;
                    default:           tmp = "UNKNOWN";  break;
                    }
                    printf("Absolute %s %d", tmp, event.value);
                    break;
                case EV_MSC: printf("Misc"); break;
                case EV_LED: printf("Led");  break;
                case EV_SND: printf("Snd");  break;
                case EV_REP: printf("Rep");  break;
                case EV_FF:  printf("FF");   break;
                    break;
                }
                printf("\n");
            }
            printf("rc = %d, (%s)\n", rc, strerror(errno));
            close(fd);
        }
    }
```

3. 軟體與硬體需求

3.1 軟體需求

- BSP 版本
  - Linux-5.4.x
- 模組名稱
  - ml56-touchkey.ko

3.2 硬體需求

- 電路元件
  - NuMaker-HMI-MA35D1 V2.0
  - ML56-TK Module
- 示意圖

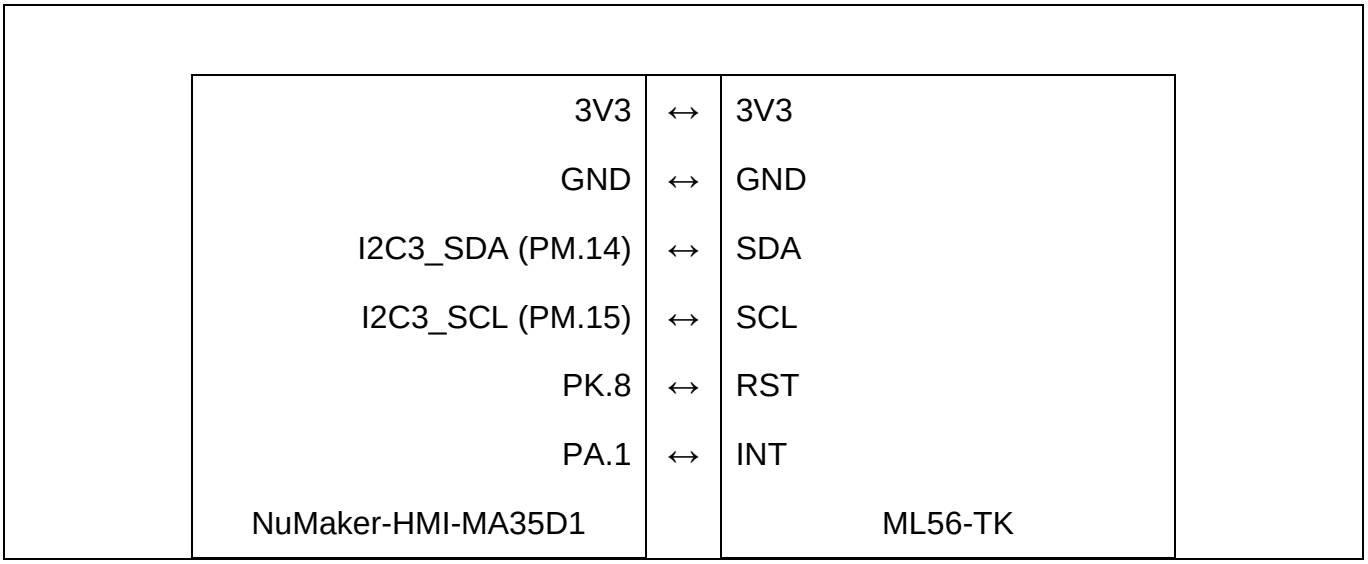

圖 3-1 連接示意圖

4. 目錄資訊

EC_MA35D1_ML56-TK_Driver_V1.00	
Src	Source files of ML56-TK device driver
Ko	Kernel object file of ML56-TK device driver
Input_test	The test program of input device
DT_Binding	Declares requirements on the contents of ML56-TK driver

圖 4-1 目錄資訊

5. 範例程式執行

1. 編修所需的腳位(請參考驅動設定)
2. 在編譯環境中重新編譯核心與模塊
3. 將 NuMaker-HMI-MA35D1 開機
4. 複製 ml56-touchkey.ko、input_test 到 NuMaker-HMI-MA35D1 上
5. 導入 ml56-tk 核心模組

```
# insmod ml56-touchkey.ko
  會出現如下畫面,紅色框部分的數字為應用程式的輸入參數
[   61.425658] input: ml56-tk as /devices/platform/0.i2c-gpio-0/i2c-0/0-0052/input/input2
```

6. 執行應用程式

```
# ./input_test 2
```

7. 觸碰 ML56-TK 觸控鍵

```
Wed Mar  2 03:50:39 2022.437039 type 0x0001; code 0x0030; value 0x00000001; Key 48 (0x30) press
Wed Mar  2 03:50:39 2022.437039 type 0x0000; code 0x0000; value 0x00000000;
Wed Mar  2 03:50:39 2022.657376 type 0x0001; code 0x0030; value 0x00000000; Key 48 (0x30) release
Wed Mar  2 03:50:39 2022.657376 type 0x0000; code 0x0000; value 0x00000000;
Wed Mar  2 03:50:41 2022.888186 type 0x0001; code 0x001e; value 0x00000001; Key 30 (0x1e) press
Wed Mar  2 03:50:41 2022.888186 type 0x0000; code 0x0000; value 0x00000000;
Wed Mar  2 03:50:42 2022.107135 type 0x0001; code 0x001e; value 0x00000000; Key 30 (0x1e) release
Wed Mar  2 03:50:42 2022.107135 type 0x0000; code 0x0000; value 0x00000000;
```

8. 觸碰 ML56-TK 觸控條及觸控滑輪

```
Wed Mar  2 03:51:20 2022.610418 type 0x0001; code 0x014a; value 0x00000001; Button 74 press
Wed Mar  2 03:51:20 2022.610418 type 0x0003; code 0x0000; value 0x00000004; Absolute X 4
Wed Mar  2 03:51:20 2022.610418 type 0x0003; code 0x0001; value 0x00000000; Absolute Y 0
Wed Mar  2 03:51:20 2022.807639 type 0x0003; code 0x0000; value 0x00000061; Absolute X 97
Wed Mar  2 03:51:20 2022.898640 type 0x0001; code 0x014a; value 0x00000000; Button 74 release
```

6. 修訂紀錄

Date	Revision	Description
2022.09.27	1.00	初始發布。

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.