



HCRTOS hy46xx_ts userguide

1. 目录

HCRTOS hy46xx_ts userguide

1. 目录
2. 文档履历
3. 概述
 - 3.1 编写目的
 - 3.2 读者对象
4. 模块介绍
5. 模块接口说明
6. 模块测试用例与Sample Code
 - 6.1 测试代码
7. 模块调试方法
8. 常见问题

2. 文档履历

版本号	日期	制/修订人	制/修订记录
1.0	2023.4.10	邱浩佳	新增文档说明

3. 概述

3.1 编写目的

介绍hy46xx驱动模块的使用

3.2 读者对象

软件开发工程师和技术支持工程师。

4. 模块介绍

- hy46xx_ts触摸屏最多支持五点的多点触摸。
- 使用i2c协议进行数据通信。
- 驱动对接了Linux的input子系统，应用层获取数据通过对应的input节点，就可以获取到数据。



4.1 设备树配置

```
1  i2c@0{
2      pinmux-active = <PINPAD_L28 3 PINPAD_L29 3>;
3      device_type = "hichip,hcrtos-setup-setbit";
4      reg_bit = <0xb8800094 18 1>;
5      devpath = "/dev/i2c0";
6      baudrate = <200000>;
7      mode = "master";
8      status = "okay";
9  };
10
11 hy46xx_ts{
12     i2c-devpath = "/dev/i2c0";
13     i2c-addr = <0x38>;
14     reset-gpio = <PINPAD_L24 0>;
15     irq-gpio = <PINPAD_L27 0>;
16     status = "okay";
17 };
```

i2c@0节点的配置请参考i2c使用说明文档。

hy46xx_ts节点的使用说明：

i2c-devpath：表示hy46xx触摸屏使用的是i2c0；

i2c-addr：表示hy46xx的器件地址；

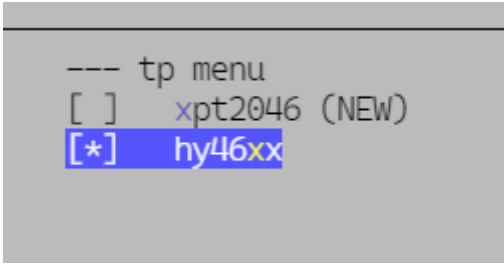
reset-gpio：hy46xx的复位引脚，0代表配置为gpio；

irq-gpio：hy46xx的中断引脚，0代表配置为gpio；

4.2 menuconfig配置

在SDK根目录输入make menuconfig，按照下面的路径进行勾选hy46xx_ts驱动。

```
1 Location:
2   -> Components
3     -> kernel (BR2_PACKAGE_KERNEL [=y])
4       -> Drivers
5         -> input event (CONFIG_DRV_INPUT [=y])
6           -> tp menu (CONFIG_TP [=y])
```



```
--- tp menu
[ ] xpt2046 (NEW)
[*] hy46xx
```

4.3 编译

配置完成后，输入make kernel-rebuild all，进行编译和烧录，在串口控制台输入ls命令就可以看到hy46xx驱动。

```
driver modules:
  avsync
  vidsink
  llav_dis
  viddec
  spo_dai
  i2s_dai
  cjc8988i_dai
  cjc8990i_dai
  cjc8988oi_dai
  pwm_dac_dai
  cjc8988_dai
  cjc8990_dai
  cs4344_dai
  auddec
  audsink
  fb
  rc_core
  ir_nec_decode
  rc_map_hcdemo
  hc_saradc_clk_init
  i2c
  wm8960_for_i2si_dai
  pwm
  mmz
  pcmo_dai
  pcmi1_dai
  pcmi2_dai
  musb_driver
  hcl6xx_driver
  hcdisk_driver
  efuse
  llav_hdmi
  spin_platform
  spo_platform
  i2si_platform
  i2so_platform
  hc_rc
  hc_hy46xx_driver
  rgb
  ge
  llav_vdec
  hcl6xx_link
```

同时，输入下面的命令就可以查看获取hy46xx数据的节点。

```
hc1512a@dbB200# nsh
hc1512a@dbB200(nsh)# ls
/:
dev/
hc1512a@dbB200(nsh)# cd dev
hc1512a@dbB200(nsh)# ls
/dev:
auddec
audsink
avsync0
avsync1
bus/
dis
fb0
ge
i2c0
input/
mmz
mtdblock0
mtdblock1
mtdblock2
mtdblock3
mtdblock4
null
persistentmem
pwm0
sf_prodetect
sndC0i2so
uart1
uart_dummy
viddec
vidsink
hc1512a@dbB200(nsh)# cd input
hc1512a@dbB200(nsh)# ls
/dev/input:
event0
event1
hc1512a@dbB200(nsh)#
```

5. 模块接口说明

该模块暂无提供接口。

6. 模块测试用例与Sample Code

在SDK根目录输入make menuconfig，根据下面路径选中测试命令

```
1 Location:
2   -> Components
3     -> Cmds (BR2_PACKAGE_CMDS [=y])
```

```

--- Cnds
[*] OS operations --->
[ ] pthread operations
[*] Nsh operations --->
[ ] sound test operations ----
[*] lsmod operations
[ ] adc test operations ----
[ ] fb test operations
[*] input event operations
[ ] mtd operations
[ ] spi operations
[ ] persistent memory operations
[ ] pok test operations
[ ] uart test operations
[ ] watchdog test operations
[ ] Efuse bits dump operations
[ ] hdmi rx test operations
[ ] tv decoder(cvbs in) test operations

```

选中后，输入make cmds-rebuild all，进行编译和烧录。在串口控制台输入下面命令即可以查看测试命令。

```

hc1512a@dbB200#
hc1512a@dbB200# input -h
*****
input test cmds help
    for example : input_test -il
                  'i'      1 means event1
*****
hc1512a@dbB200# input -il

```

按压触摸屏就会接收到触摸点的消息。

6.1 测试代码

```

1  #include <stdlib.h>
2
3  #include <poll.h>
4  #include <unistd.h>
5  #include <stddef.h>
6  #include <stdio.h>
7  #include <fcntl.h>
8  #include <sys/ioctl.h>
9  #include <hcuapi/input.h>
10 #include <kernel/lib/console.h>
11
12 #define BUF_SIZE 1024
13
14 static void print_help(void) {
15     printf("*****\n");
16     printf("input test cmds help\n");
17     printf("\tfor example : input_test -il\n");
18     printf("\t'i'      1 means event1\n");
19     printf("*****\n");
20 }
21
22 static int input_test(int argc, char *argv[])

```

```

22 {
23     int fd;
24     struct input_event t;
25     struct pollfd pfd;
26     char input_buf[BUF_SIZE];
27     char *s = "/dev/input/event";
28
29     long tmp;
30     int x = 0, y = 0;
31     int event_num = -1;
32     char ch;
33     opterr = 0;
34     optind = 0;
35
36     while((ch = getopt(argc, argv, "hi:")) != EOF){
37         switch (ch) {
38             case 'h':
39                 print_help();
40                 return 0;
41             case 'i':
42                 tmp = strtoll(optarg, NULL, 10);
43                 event_num = tmp;
44                 break;
45             default:
46                 printf("Invalid parameter %c\r\n", ch);
47                 print_help();
48                 return -1;
49         }
50     }
51     if(event_num == -1)
52     {
53         print_help();
54         return -1;
55     }
56
57     sprintf(input_buf, "/dev/input/event%d", event_num);
58
59     fd = open(input_buf, O_RDONLY);
60     pfd.fd = fd;
61     pfd.events = POLLIN | POLLRDNORM;
62
63     if(fd < 0){
64         printf("can't open %s\n", input_buf);
65         return -1;
66     }
67
68     while (1) {
69         if (poll(&pfd, 1, -1) <= 0)
70             continue;
71
72         if (read(fd, &t, sizeof(t)) != sizeof(t))
73             continue;
74
75         printf("type:%d, code:%d, value:%ld\n", t.type, t.code,
76 t.value);
77
78         if (t.type == EV_KEY) {
79             printf("key %d %s\n", t.code,

```

```

79         (t.value) ? "Pressed" :
"Released");
80         if (t.code == KEY_POWER && !t.value) {
81             while (read(fd, &t, sizeof(t)) ==
sizeof(t))
82                 ;
83             break;
84         }
85     }
86     else{
87         if (t.type == EV_ABS)
88         {
89             if (t.type == EV_ABS&& t.code == ABS_X) {
90                 x = t.value;
91             }
92             if (t.type == EV_ABS && t.code == ABS_Y) {
93                 y = t.value;
94             }
95         }
96         if (t.type == EV_SYN) {
97             printf("(%4d %4d)\n",x,y);
98         }
99     }
100 }
101
102 close(fd);
103
104 return 0;
105 }
106
107 CONSOLE_CMD(input, NULL, input_test, CONSOLE_CMD_MODE_SELF, "input test,
press power to exit test")

```

7. 模块调试方法

使用测试命令测试。

8. 常见问题

暂无