

# 3

## 3.1

Linux Linux

## 3.2

Linux input core( ) drivers( ) event handlers( ) Lin

InputSystem\_Image002  
InputSystem\_Image002

--> --> --> --->

## 3.3

```
cat /proc/bus/input/devices
```

event ubuntu

InputSystem\_Image003  
InputSystem\_Image003

I N P S U H B

!id of the device( ID)

struct input\_id

```
41 struct input_id {  
42     //  
43     __u16 bustype;  
44     //      ID  
45     __u16 vendor;  
46     //      ID  
47     __u16 product;  
48     //      ID  
49     __u16 version;  
50 };
```

**N:name of the device**

**P:physical path to the device in the system hierarchy**

**S:sysfs path**

sys

**U:unique identification code for the device(if device has it)**

**H:list of input handles associated with the device.**

**B:bitmaps( )**

PROP:device properties and quirks.

EV:types of events supported by the device.

KEY:keys/buttons this device has.

MSC:miscellaneous events supported by the device.

LED:leds present on the device.

PROP:

EV:

KEY:        /

MSC:

LED:

```
cat /proc/bus/input/devices
```

event1

event1

**cat**

InputSystem\_Image004

Image not found or type unknown

```
cat /dev/input/event1
```

**hexdump**

InputSystem\_Image005

Image not found or type unknown

input\_event

/usr/include/linux/input.h

input\_event

```
24 struct input_event {
25     /*
26     * struct timeval time;
27     */
28     __u16 type;
29     /*
30     * __u16 code;
31     */
32     __s32 value;
33 };
```

input\_event      time

```
1 struct timeval
2 {
3     __time_t tv_sec;          /* Seconds. */
4     __suseconds_t tv_usec;    /*Microseconds. */
5 };
```

tv\_sec Epoch      struct timeval      tv\_usec      Epoch      1970

input\_event      type

type      code      X Y      value

(type)

```
/usr/include/linux/input-event-codes.h
```

Linux

```
/usr/include/linux/input.h
```

```
34 /*
35  * Event types
36  */
37
38 #define EV_SYN 0x00//
39 #define EV_KEY 0x01//
40 #define EV_REL 0x02//
41 #define EV_ABS 0x03//
42 #define EV_MSC 0x04
43 #define EV_SW 0x05
44 #define EV_LED 0x11
45 #define EV_SND 0x12
46 #define EV_REP 0x14
47 #define EV_FF 0x15
48 #define EV_PWR 0x16
49 #define EV_FF_STATUS 0x17
50 #define EV_MAX 0x1f
51 #define EV_CNT (EV_MAX+1)
```

**(code)**

:

```
/usr/include/linux/input-event-codes.h
```

Linux

```
/usr/include/linux/input.h
64 /*
65  * Keys and buttons
66  *
67  * Most of the keys/buttons are modeled after USB HUT 1.12
68  * (see http://www.usb.org/developers/hidpage).
69  * Abbreviations in the comments:
70  * AC - Application Control
71  * AL - Application Launch Button
72  * SC - System Control
73  */
74
75 #define KEY_RESERVED 0
76 #define KEY_ESC 1
77 #define KEY_1 2
78 #define KEY_2 3
79 #define KEY_3 4
80 #define KEY_4 5
81 #define KEY_5 6
82 #define KEY_6 7
83 #define KEY_7 8
84 #define KEY_8 9
85 #define KEY_9 10
86 #define KEY_0 11
87 #define KEY_MINUS 12
88 #define KEY_EQUAL 13
89 #define KEY_BACKSPACE 14
90 #define KEY_TAB 15
91 #define KEY_Q 16
92 #define KEY_W 17
...

```

(value)

# 3.4 USB

USB cat /proc/bus/input/devices USB ev

InputSystem\_Image006

Image not found or type unknown

hexdump :

InputSystem\_Image007

Image not found or type unknown

3.3 USB

1 (type)

3.3

```
/usr/include/linux/input-event-codes.h
```

Linux

```
/usr/include/linux/input.h
```

```
34 /*
35  * Event types
36  */
37
38 #define EV_SYN 0x00//
39 #define EV_KEY 0x01//
40 #define EV_REL 0x02//
41 #define EV_ABS 0x03//
```

```

42 #define EV_MSC 0x04
43 #define EV_SW 0x05
44 #define EV_LED 0x11
45 #define EV_SND 0x12
46 #define EV_REP 0x14
47 #define EV_FF 0x15
48 #define EV_PWR 0x16
49 #define EV_FF_STATUS 0x17
50 #define EV_MAX 0x1f
51 #define EV_CNT (EV_MAX+1)

```

## 2 (code)

USB code,

```
/usr/include/linux/input-event-codes.h
```

Linux

```
/usr/include/linux/input.h
```

```

696 /*
697  * Relative axes
698  */
699
700 #define REL_X 0x00 // X
701 #define REL_Y 0x01 // Y
702 #define REL_Z 0x02
703 #define REL_RX 0x03
704 #define REL_RY 0x04
705 #define REL_RZ 0x05
706 #define REL_HWHEEL 0x06
707 #define REL_DIAL 0x07
708 #define REL_WHEEL 0x08
709 #define REL_MISC 0x09
710 #define REL_MAX 0x0f
711 #define REL_CNT (REL_MAX+1)

```

REL\_X REL\_Y

| value, | (type) | (code) | X | Y |
|--------|--------|--------|---|---|
|--------|--------|--------|---|---|

input

```
#include <linux/input.h>
```

1           input\_event   input

```
struct input_event event_mouse ;
```

2   input                    USB   event2

```
open("/dev/input/event2", O_RDONLY);
```

3

```
read( fd , &event_mouse , sizeof(event_mouse));
```

4

```
//
if(EV_ABS == event_mouse.type || EV_REL == event_mouse.type)
{
    //code    X Y    X   X    value
    //    Y   Y    value
    if(event_mouse.code == REL_X)
    {
        printf("event_mouse.code_X: %d\n", event_mouse.code);
        printf("event_mouse.value_X: %d\n", event_mouse.value);
    }
    else if(event_mouse.code == REL_Y)
    {
        printf("event_mouse.code_Y: %d\n", event_mouse.code);
        printf("event_mouse.value_Y: %d\n", event_mouse.value);
    }
}
```

5

```
close( fd );
```



## Linux

```
01 #include <stdio.h>
02 #include <unistd.h>
03 #include <stdlib.h>
04 #include <fcntl.h>
05 #include <linux/input.h>
06
07 int main(void)
08 {
09     //1         input
10     struct input_event event_mouse ;
11     //2  input          USB          event2
12     int fd      = -1 ;
13     fd = open("/dev/input/event2", O_RDONLY);
14     if(-1 == fd)
15     {
16         printf("open mouse event fair! \n");
17         return -1 ;
18     }
19     while(1)
20     {
21         //3
22         read(fd, &event_mouse, sizeof(event_mouse));
23         if(EV_ABS == event_mouse.type || EV_REL == event_mouse.type)
24         {
25             //code      X  Y      X      X      value
26             //      Y      Y      value
27             if(event_mouse.code == REL_X)
28             {
29                 printf("event_mouse.code_X: %d\n", event_mouse.code);
30                 printf("event_mouse.value_X: %d\n", event_mouse.value);
31             }
32             else if(event_mouse.code == REL_Y)
33             {
34                 printf("event_mouse.code_Y: %d\n", event_mouse.code);
35                 printf("event_mouse.value_Y: %d\n", event_mouse.value);
36             }
```

```
37  }  
38  }  
39  close(fd);  
40  return 0 ;  
41 }
```

```
gcc test_mouse.c -o test_mouse
```

## InputSystem\_Image008

Image not found or type unknown

test\_mouse      test\_mouse

## InputSystem\_Image009

Image not found or type unknown

X                  value X

## InputSystem\_Image010

Image not found or type unknown

Y                  value Y

# 3.5

3.3

3.4

event1,    3.3    3.4

input

```
#include <linux/input.h>
```

```
struct input_event event_keyboard ;
```

```
open("/dev/input/event1", O_RDONLY);
```

```
4
//
if(EV_KEY == event_keyboard.type)
{
    if(1 == event_keyboard.value)
        printf("      :%d      :%d  \n", event_keyboard.type, event_keyboard.code);
    else if(0 == event_keyboard.value)
        printf("      :%d      :%d  \n", event_keyboard.type, event_keyboard.code);
}
```

```
01 #include <stdio.h>
02 #include <unistd.h>
03 #include <stdlib.h>
04 #include <fcntl.h>
05 #include <linux/input.h>
06
07 int main( void)
08 {
09     //1          input
10     struct input_event event_keyboard ;
11     //2  input          event1
12     int fd      = -1 ;
```

```

13  fd = open("/dev/input/event1", O_RDONLY);
14  if(-1 == fd)
15  {
16      printf("open mouse event fail! \n");
17      return -1 ;
18  }
19  while(1)
20  {
21      //3
22      read( fd, &event_keyboard, sizeof(event_keyboard));
23  if(EV_KEY == event_keyboard.type)
24  {
25      if(1 == event_keyboard.value)
26      printf("      :%d      :%d  \n", event_keyboard.type, event_keyboard.code);
27      else if(0 == event_keyboard.value)
28      printf("      :%d      :%d  \n", event_keyboard.type, event_keyboard.code);
29  }
30  }
31  close(fd);
32  return 0 ;
33 }

```

USB

USB

value

```
gcc test_keyboard.c -o test_keyboard
```

InputSystem\_Image011

Image not found or type unknown

test\_keyboard

test\_keyboard

InputSystem\_Image012

Image not found or type unknown

# 3.6

# imx6ul

input

input X Y

EV\_ABS

X Y

ABS\_MT\_POSITION\_X ABS\_MT\_POSITION\_Y

```
01 #include <stdio.h>
02 #include <unistd.h>
03 #include <fcntl.h>
04 #include <stdlib.h>
05 #include <linux/input.h>
06
07 int main(int argc, char **argv)
08 {
09     int tp_fd = -1 ;
10     int tp_ret = -1 ;
11     int touch_x, touch_y ;
12     struct input_event imx6ull_ts ;
13     //1
14     tp_fd = open("/dev/input/event1", O_RDONLY);
15     if(tp_fd < 0)
16     {
17         printf("open /dev/input/event1 fail! \n");
18         return -1 ;
19     }
20     while(1)
21     {
22         //2
23         read(tp_fd , &imx6ull_ts , sizeof(imx6ull_ts));
24         switch(imx6ull_ts.type)
25         {
26             case EV_ABS:
```

```

27 if(imx6ull_ts.code == ABS_MT_POSITION_X)
28     touch_x = imx6ull_ts.value ;
29 if(imx6ull_ts.code == ABS_MT_POSITION_Y)
30     touch_y = imx6ull_ts.value ;
31 break ;
32 default:
33 break ;
34 }
35 printf("touch_x: %d touch_y: %d\n", touch_x, touch_y);
36 usleep(100);
37 }
38 close(tp_fd);
39 return 0;
40 }

```

```
gcc test_touchscreen.c -o test_touchscreen
```

( PC )

InputSystem\_Image013

Image not found or type unknown

rz PC test\_touchscreen

InputSystem\_Image014

Image not found or type unknown

11 PC

test\_touchscreen :

InputSystem\_Image015

Image not found or type unknown

test\_touchscreen

InputSystem\_Image016

Image not found or type unknown

Revision #1

Created 2 March 2022 10:23:54 by

Updated 2 March 2022 10:24:09 by