3

3.1

Linux Linux

3.2

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

InputSystem_Image002 Image not found or type unknown

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

3.3

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

event ubuntu

InputSystem_Image003 Image not found or type unknown

I N P S U H B

I:id of the device(ID)

struct input_id

```
struct input_id {
	//
	__u16 bustype;
	//    ID
	__u16 vendor;
	//    ID
	__u16 product;
	//  ID
	__u16 version;
};
```

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

```
struct input_event {
	//
	struct timeval time;
	//
	__u16 type;
	//
	__u16 code;
	//
	__s32 value;
};
```

input_event time

```
struct timeval
{
    __time_t tv_sec;         /* Seconds.  */
    __suseconds_t tv_usec;     /*Microseconds.  */
};
```

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
```

```
/*
 * Event types
 */

#define EV_SYN          0x00    //
#define EV_KEY          0x01    //
#define EV_REL          0x02    //
#define EV_ABS          0x03    //
#define EV_MSC          0x04
#define EV_SW           0x05
#define EV_LED          0x11
#define EV_SND          0x12
#define EV_REP          0x14
#define EV_FF           0x15
#define EV_PWR          0x16
#define EV_FF_STATUS    0x17
#define EV_MAX          0x1f
#define EV_CNT          (EV_MAX+1)
```

(code)

:

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

Linux

```
/usr/include/linux/input.h
/*
 * Keys and buttons
 *
 * Most of the keys/buttons are modeled after USB HUT 1.12
 * (see http://www.usb.org/developers/hidpage).
 * Abbreviations in the comments:
 * AC - Application Control
 * AL - Application Launch Button
 * SC - System Control
 */

#define KEY_RESERVED□□0
#define KEY_ESC□□□1
#define KEY_1□□□2
#define KEY_2□□□3
#define KEY_3□□□4
#define KEY_4□□□5
#define KEY_5□□□6
#define KEY_6□□□7
#define KEY_7□□□8
#define KEY_8□□□9
#define KEY_9□□□10
#define KEY_0□□□11
#define KEY_MINUS□□12
#define KEY_EQUAL□□13
#define KEY_BACKSPACE□□14
#define KEY_TAB□□□15
#define KEY_Q□□□16
#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
```

```
/*
 * Event types
 */

#define EV_SYN□□□0x00□//
#define EV_KEY□□□0x01□//
#define EV_REL□□□0x02□//
#define EV_ABS□□□0x03□//
```

```
#define EV_MSC      0x04
#define EV_SW       0x05
#define EV_LED      0x11
#define EV_SND      0x12
#define EV_REP      0x14
#define EV_FF       0x15
#define EV_PWR      0x16
#define EV_FF_STATUS    0x17
#define EV_MAX      0x1f
#define EV_CNT      (EV_MAX+1)
```

2 (code)

USB code,

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

Linux

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

```
/*
 * Relative axes
 */

#define REL_X       0x00 //  X
#define REL_Y       0x01 //  Y
#define REL_Z       0x02
#define REL_RX      0x03
#define REL_RY      0x04
#define REL_RZ      0x05
#define REL_HWHEEL  0x06
#define REL_DIAL    0x07
#define REL_WHEEL   0x08
#define REL_MISC    0x09
#define REL_MAX     0x0f
#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

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <fcntl.h>
#include <linux/input.h>

int main(void)
{
    //1           input
    struct input_event event_mouse ;
    //2   input           USB       event2
    int fd    = -1 ;
    fd = open("/dev/input/event2", O_RDONLY);
    if(-1 == fd)
    {
        printf("open mouse event fair!\n");
        return -1 ;
    }
    while(1)
    {
        //3
        read(fd, &event_mouse, sizeof(event_mouse));
        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);
            }
□□}
    }
    close(fd);
    return 0 ;
}
```

```
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>
```

1 input_event input

```
struct input_event event_keyboard ;
```

2 input event1

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

3

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

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);
}
```

5

```
close(fd);
```

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <fcntl.h>
#include <linux/input.h>

int main(void)
{
    //1           input
    struct input_event event_keyboard ;
    //2  input                 event1
    int fd    = -1 ;
□   fd = open("/dev/input/event1", O_RDONLY);
    if(-1 == fd)
    {
        printf("open mouse event fair!\n");
        return -1 ;
    }
    while(1)
    {
        //3
        read(fd, &event_keyboard, sizeof(event_keyboard));
□□   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);
□□}
    }
    close(fd);
    return 0 ;
}
```

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

```
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdlib.h>
#include <linux/input.h>

int main(int argc, char **argv)
{
    int tp_fd  = -1 ;
    int tp_ret = -1 ;
    int touch_x,touch_y ;
    struct input_event imx6ull_ts ;
    //1
    tp_fd = open("/dev/input/event1",O_RDONLY);
    if(tp_fd < 0)
    {
       printf("open /dev/input/event1 fail!\n");
       return -1 ;
    }
    while(1)
    {
        //2
         read(tp_fd ,&imx6ull_ts ,sizeof(imx6ull_ts));
         switch(imx6ull_ts.type)
        {
          case EV_ABS:
            if(imx6ull_ts.code == ABS_MT_POSITION_X)
                touch_x = imx6ull_ts.value ;
            if(imx6ull_ts.code == ABS_MT_POSITION_Y)
                touch_y = imx6ull_ts.value ;
            break ;
            defalut:
            break ;
        }
        printf("touch_x: %d touch_y: %d\n",touch_x,touch_y);
        usleep(100);
    }
    close(tp_fd);
    return 0;
}
```

```
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