# EETI eGTouch Android Guide v2.5M

## TABLE OF CONTENTS

TABLE OF CONTENTS....0
Sec 1: Introduction....1
Sec 2: Before install....1
2-1 Patch kernel module....1
2-1.a make menuconfig....2
2-1.b manually modify config file....3
2-2 Patch kernel source code....3
2-2.a kernel 2.6.33 downwards....3
2-2.b kernel 2.6.34 upwards....6
2-2.c kernel 3.8 to 3.12....8
2-3 check device....9
Sec 3: Install Process....10
Sec 4: Calibration Tool....12
4-1 USB interface....12
4-2 RS232 interface....12
Sec 5: eGTouchA.ini Parameter Explanations....12
Sec 6: FAQ....16
6-1 Touch not working....16
6-2 Touch screen can (NOT) wake up display....18
6-3 eGTouchD can NOT find UART interface device....18
6-4 My UART device receive unexpected data from eGTouchD....18
Sec 7: Support....19
7-1 Need Support From EETI....19
7-2 Environment Information....19
7-3 Register input devices....19
7-4 Driver debug log....19

# Sec 1: Introduction

EETI provides all kinds of touch solution. EETI eGTouch is a touch daemon driver for EETI touch controller. Support USB & RS232 interface. Available for Android 2.1 upward. This document would assist you to install eGTouch and describe eGTouch feature in detail.

**If you encounter any problem as running eGTouchD driver, please help us follow the steps described in SEC 8 to collect debug information.** Send the information to us and tell us your problem. With useful information we could help you solve the problem faster.
Technical support and any question: **touch_fae@eeti.com**

# Sec 2: Before install

## 2-1 Patch kernel module

Before installing the driver, we need below kernel modules support:

**1. CONFIG_INPUT_EVDEV**

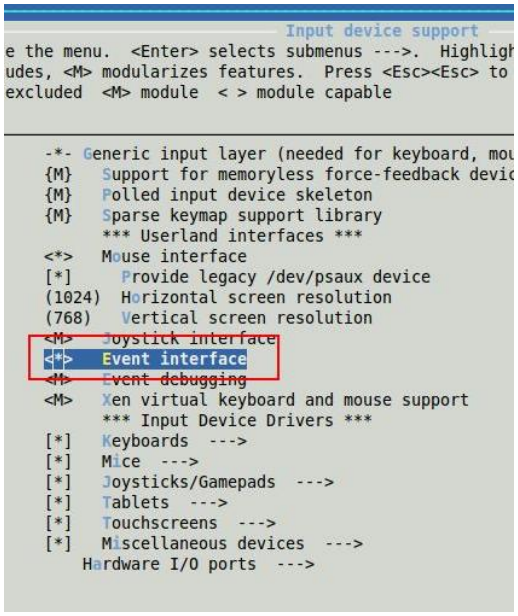
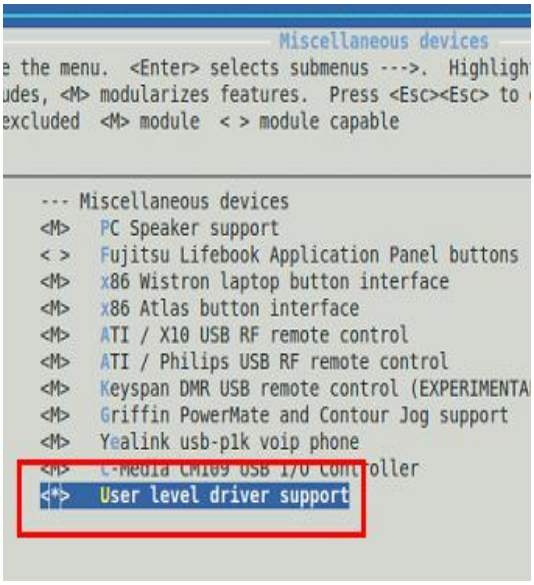
**2. CONFIG_INPUT_UINPUT**

**3. CONFIG_HIDRAW ( for USB Interface )**

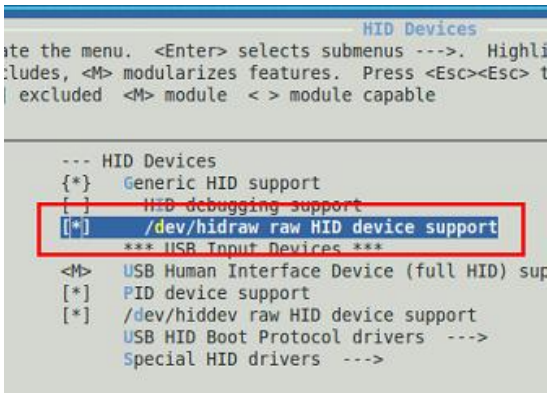
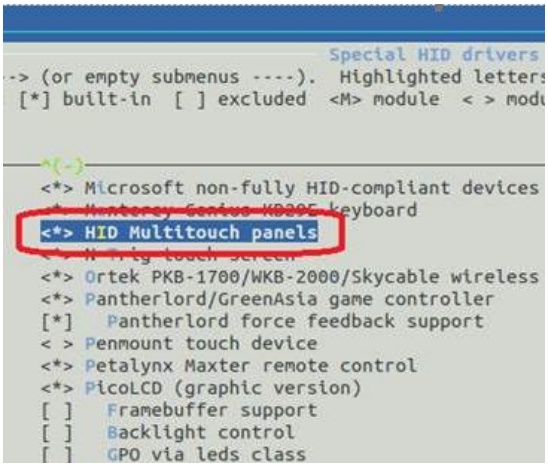
**4. HID_MULTITOUCH ( USB Interface & Kernel 3.0 upwards )**

Please make sure to these modules are enabled. Users could check this by command `make menuconfig` or modify config file manually.

## 2-1.a make menuconfig

| [Device Drivers] / [Input device support] / [Event interface] | [Device Drivers] / [Input device support] / [Miscellaneous devices] / [User level driver support] |
|---|---|
|  |  |



| [Device Drivers] / [HID Devices] / [/dev/hidraw raw HID device support] **( for USB Interface )** | [Device Drivers] / [HID Devices] / Special HID drivers / HID Multitouch panels **( If Kernel Version 3.0 upwards & for USB Interface )** |
|---|---|
|  |  |

### 2-1.b manually modify config file

In some specific Android bsp, `make menuconfig` is invalid for setting kernel config. In this case, we'll advise to set those kernel config by directly modifying the config file. You could see many kernel config files under this path:

**/Your_Android_bsp/kernel/arch/arm/configs/**

Please select the file relevant to your platform model. For example, if you're using "tegra harmony" platform, the file name would be "**tegra_harmony_android_defconfig**".

Find out the config file based on that rule and modify the file manually. Make sure necessary items are set correctly as below:

**CONFIG_INPUT_EVDEV=y**

**CONFIG_INPUT_UINPUT=y**

**CONFIG_HIDRAW=y**

**CONFIG_HID_MULTITOUCH=y**

## 2-2 Patch kernel source code

If you're **NOT** using **USB** interface, you could **SKIP** this part.

Please append following **RED** section into your source code.

| If your kernel is **2.6.33 downwards**, please follow section **2.2.a.** |
|---|
| If your kernel is **2.6.34 upwards**, please follow section **2.2.b.** |

### 2-2.a kernel 2.6.33 downwards

1. /SourceCode/drivers/input/**evdev.c**

```
static struct input_device_id evdev_blacklist[] =
{ /* Added by EETI */
        {
        .flags  = INPUT_DEVICE_ID_MATCH_BUS | INPUT_DEVICE_ID_MATCH_VENDOR,
        .bustype = BUS_USB,
        .vendor = 0x0EEF,
        },
        {},     /* Terminating entry */
};
```

```
static struct input_handler evdev_handler = {
        .event = evdev_event,
        .connect = evdev_connect,
        .disconnect = evdev_disconnect,
        .fops = &evdev_fops,
        .minor = EVDEV_MINOR_BASE,
        .name = "evdev",
        .id_table = evdev_ids,
        .blacklist = evdev_blacklist, /* Added by EETI */
};
```

2. /SourceCode/drivers/input/**mousedev.c**

```
static struct input_device_id mousedev_blacklist[] =
{       /* Added by EETI */
        {
        .flags = INPUT_DEVICE_ID_MATCH_BUS | INPUT_DEVICE_ID_MATCH_VENDOR,
        .bustype = BUS_USB,
        .vendor = 0x0EEF,
        },
        {
        .flags  = INPUT_DEVICE_ID_MATCH_BUS | INPUT_DEVICE_ID_MATCH_VENDOR,
        .bustype = BUS_VIRTUAL,
        .vendor = 0x0EEF,
        },
        {},     /* Terminating entry */
};
static struct input_handler mousedev_handler = {
        .event = mousedev_event,
        .connect = mousedev_connect,
        .disconnect = mousedev_disconnect,
        .fops = &mousedev_fops,
        .minor = MOUSEDEV_MINOR_BASE,
        .name = "mousedev",
        .id_table = mousedev_ids,
        .blacklist = mousedev_blacklist, /* Added by EETI */
};
```

3. /SourceCode/drivers/input/**joydev.c**

```
static const struct input_device_id joydev_blacklist[] =
{
        {
        .flags = INPUT_DEVICE_ID_MATCH_EVBIT | INPUT_DEVICE_ID_MATCH_KEYBIT,
        .evbit = { BIT_MASK(EV_KEY) },
        .keybit = { [BIT_WORD(BTN_TOUCH)] = BIT_MASK(BTN_TOUCH) },
        },      /* Avoid itouchpads and touchscreens */
        {
        .flags = INPUT_DEVICE_ID_MATCH_EVBIT | INPUT_DEVICE_ID_MATCH_KEYBIT,
        .evbit = { BIT_MASK(EV_KEY) },
        .keybit = { [BIT_WORD(BTN_DIGI)] = BIT_MASK(BTN_DIGI) },
        },      /* Avoid tablets, digitisers and similar devices */
        {
        .flags = INPUT_DEVICE_ID_MATCH_BUS | INPUT_DEVICE_ID_MATCH_VENDOR,
        .bustype = BUS_VIRTUAL,
        .vendor = 0x0EEF,
        },      /* Added by EETI */
        { }     /* Terminating entry */
};

static struct input_handler joydev_handler = {
        .event = joydev_event,
        .connect = joydev_connect,
        .disconnect = joydev_disconnect,
        .fops = &joydev_fops,
        .minor = JOYDEV_MINOR_BASE,
        .name = "joydev",
        .id_table = joydev_ids,
        .blacklist = joydev_blacklist,
};
```

### 2-2.b kernel 2.6.34 upwards

1. /SourceCode/drivers/input/**evdev.c**

```
static bool evdev_match(struct input_handler *handler, struct input_dev *dev)
{
        /* Avoid EETI USB touchscreens */
        #define VID_EETI 0x0EEF
        if ((BUS_USB == dev->id.bustype) && (VID_EETI == dev->id.vendor))
            return false;
        return true;
}

static struct input_handler evdev_handler = {
        .event = evdev_event,
        .match = evdev_match,  /* Added by EETI*/
        .connect = evdev_connect,
        .disconnect = evdev_disconnect,
        .fops = &evdev_fops,
        .minor = EVDEV_MINOR_BASE,
        .name = "evdev",
        .id_table = evdev_ids,
};
```

2. /SourceCode/drivers/input/**mousedev.c**

```
static bool mousedev_match(struct input_handler *handler, struct input_dev *dev)
{
        /* Avoid EETI USB touchscreens */
        #define VID_EETI 0x0EEF
        if ((BUS_USB == dev->id.bustype) && (VID_EETI == dev->id.vendor))
            return false;
        /* Avoid EETI virtual devices */
        if ((BUS_VIRTUAL == dev->id.bustype) && (VID_EETI == dev->id.vendor))
            return false;
        return true;
}
static struct input_handler mousedev_handler = {
        .event = mousedev_event,
        .match = mousedev_match,    /* Added by EETI */
```

```
        .connect = mousedev_connect,
        .disconnect = ousedev_disconnect,
        .fops = &mousedev_fops,
        .minor = MOUSEDEV_MINOR_BASE,
        .name = "mousedev",
        .id_table = mousedev_ids,
};
```

3. /SourceCode/drivers/input/**joydev.c**

```
static bool joydev_match(struct input_handler *handler, struct input_dev *dev)
{
        /* Avoid touchpads and touchscreens */
        if (test_bit(EV_KEY, dev->evbit) && test_bit(BTN_TOUCH, dev->keybit))
            return false;
        /* Avoid tablets, digitisers and similar devices */
        if (test_bit(EV_KEY, dev->evbit) && test_bit(BTN_DIGI, dev->keybit))
            return false;
        /* Avoid EETI virtual devices */
        #define VID_EETI 0x0EEF
        if (( BUS_VIRTUAL == dev->id.bustype) && (VID_EETI == dev->id.vendor))
            return false;
        return true;
}

static struct input_handler joydev_handler = {
        .event = joydev_event,
        .match = joydev_match,
        .connect = joydev_connect,
        .disconnect = joydev_disconnect,
        .fops = &joydev_fops,
        .minor = JOYDEV_MINOR_BASE,
        .name = "joydev",
        .id_table = joydev_ids,
};
```

### 2-2.c kernel 3.8 to 3.12

**Important! Before install driver, if your system fulfill below two conditions, please patch kernel hid-core.c first, or driver would NOT be functional.**

| | |
|---|---|
| **1.** | **Interface is USB** |
| **2.** | **Kernel Version is 3.8.x to 3.12.x** |
| **3** | **Resistive or SCAP controller** |

If your Linux kernel version is **3.8** to **3.12**, using **resistive** or **SCAP** touch controller, please comment the following **RED** section in your source code.

/SourceCode/drivers/hid/hid-core.c

```
bool hid_ignore(struct hid_device *hdev)
{
   …
   switch (hdev->vendor) {
        …
        /*case USB_VENDOR_ID_DWAV:*/
            /* These are handled by usbtouchscreen. hdev->type is probably
             * HID_TYPE_USBNONE, but we say !HID_TYPE_USBMOUSE to match
             * usbtouchscreen. */
             /*if ((hdev->product == USB_DEVICE_ID_EGALAX_TOUCHCONTROLLER ||
                  hdev->product == USB_DEVICE_ID_DWAV_TOUCHCONTROLLER) &&
                  hdev->type != HID_TYPE_USBMOUSE)
                      return true;
              break;*/
        …
   }
   …
}
```

## 2-3 check device

Please make sure these check items are passed, if not so, the driver would not be executed successfully.

1.) After patching kernel, build new kernel and reboot for taking effect on changes.

2.) Check uninput functions enable or not

| UINPUT device node |
| --- |
| You should see uinput under **/dev/input/uinput** or **/dev/uinput.**<br>For example:<br>File Edit View Terminal Help<br>root@william-desktop:/dev/input# pwd<br>/dev/input<br>root@william-desktop:/dev/input# ls uinput -al<br>crw-r----- 1 root root 10, 223 2010-01-05 15:43 uinput<br>root@william-desktop:/dev/input# |

3.) If interface is **USB**. After plug in an USB device, check below functions.

| HIDRAW device node |
| --- |
| As the usb device is plug-in, there would be a **hidraw** node generated under **/dev**<br>File Edit View Terminal Help<br>root@william-desktop:/dev# pwd<br>/dev<br>root@william-desktop:/dev# ls hidraw* -al<br>crw-rw---- 1 root root 251, 0 2010-01-05 17:02 hidraw0<br>root@william-desktop:/dev# |
| **USB touch device handlers** |
| Type command "**cat /proc/bus/input/devices**" and see the result.<br>If you need and have done the kernel source code patch at section 2.2, you would see a **blank content** behind the **Handlers** item.<br>I: Bus=0003 Vendor=0eef Product=720c Version=0100<br>N: Name="eGalax Inc. USB TouchController"<br>P: Phys=usb-0000:00:1d.0-2/input0<br>S: Sysfs=/devices/pci0000:00/0000:00:1d.0/usb2/2-2/2-2:1.0/input/input7<br>U: Uniq=<br>H: Handlers=<br>B: EV=1b<br>B: KEY=421 0 30001 0 0 0 0 0 0 0<br>B: ABS=100 3f<br>B: MSC=10 |

# Sec 3: Install Process

This section describes how to install eGTouch into Android. Before following this, please make sure referring to "**Section 2 Patch Kernel**" to rebuild kernel for supporting necessary features.

1. EETI eGTouch package contains:
   a) eGTouchD: a daemon service driver for EETI touch controller.
   b) eGTouchA.ini : a parameter list could be loaded by driver
   c) eGalaxTouch_VirtualDevice.idc: a file necessary for Android 3.0 upwards
   d) eGalaxCalibrator: a tool provides calibration and line drawing.

2. Place "eGTouchA.ini" into Android system directory "/data/eGalax/eGTouchA.ini" where driver would load it. We can change driver behavior by modifying this file. **The detail descriptions of parameters are described in Section 5.** ( You can see brief definitions in eGTouchA.ini )
   You can also use parameter "-i" to set ini file path.
   Ex: "eGTouchD –i /data/eGalax/eGTouchA.ini"
   or add below string to file: "Android/system/core/rootdir/init.rc"

```
mkdir /data/eGalax
```

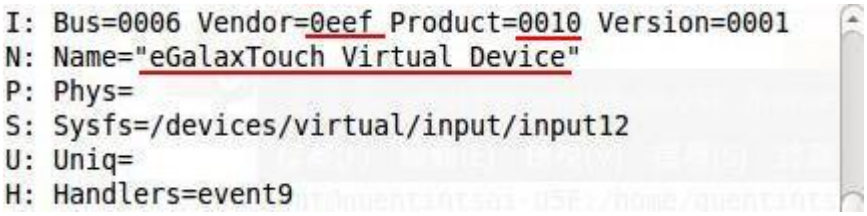
   eGTouchD will create default eGTouchA.ini at /data/eGalax/ automatically.

3. To make eGTouchD driver run from the beginning of system operation. Please put eGTouchD driver into Android system directory "/system/bin". Modify the file "**init.rc**" in Android system and add service:

| | |
|---|---|
| **Below** Android **4.0** | **service eGTouchD /system/bin/eGTouchD**<br>**user root**<br>**group root**<br>**oneshot** |
| **Above** Android **4.0** | **service eGTouchD /system/bin/eGTouchD**<br>**class main**<br>**user root**<br>**group root** |

4. [Android 3.0 upwards] Please put file “eGalaxTouch_VirtualDevice.idc” into Android system directory "/system/usr/idc/".

5. After launching eGTouchD, check **/proc/bus/input/devices** file and we will find a virtual devices called “eGalaxTouch Virtual Device”. Information like below figure:




We could check event node which was assigned to the virtual device and read/get input event through this device node, e.g. /dev/input/eventX.

6. If in item 5, the virtual device is not generated, please check whether there is hidraw generated under /dev/ directory as plug-in the device. If not, this may cause by invalid kernel patch, please do kernel patch based on Sec 2.1.b.

# Sec 4: Calibration Tool

If you're using the "**PCAP controller**", please **SKIP this section** and **DO NOT install** this calibration tool. The PCAP controller doesn't need to do any calibration.

## 4-1 USB interface

Please install the tool **eGalaxCalibrator.apk** under the **/CalibrationTool/USB/** directory.

## 4-2 RS232 interface

Please install the tool **eGalaxAutoCalib.apk** under the **/CalibrationTool/UART/** directory.

As the calibration couldn't work properly, please contact EETI touch_fae@eeti.com for technical support.

# Sec 5: eGTouchA.ini Parameter Explanations

The file **eGTouchA.ini** has a parameter list which would be loaded by driver. Driver's behavior could be changed by these parameters. Please **DON'T** modify the front title as setting up eGTouchA.ini.

This table describe the detailed usage of all parameters. There is also a simple description in eGTouchA.ini.

| ◆ DebugEnableBits | Debug message you want to show. |
|---|---|
| 0 | Close all Debug |
| 1 | Print initialization debug message **[Default]** |
| FFFFF | Open all Debug |
| ◆ ShowDebugPosition | Position you want to show/store Debug message |
| 0 | Print in file located at /data/eGalax/ |
| 1 | Print in logcat **[Default]** |
| 2 | Print in above both |
| ◆ RemoveDriverLog | Remove old Driver Log. |
| 0 | Keep all old log. |
| 1 | Remove all old log. **[Default]** |
| 2 | Keep last 4 old log + new log. |
| ◆ DeviceNums | How many devices you want to plug-in to the system. If you want more than one device, please modify this value. |
| 1 | Only one device **[Default]** |
| 2~10 | More than one device. **[Max = 10]** |

| | |
|---|---|
| ◆ Baudrate | Choose the BaudRate |
| 0 | Auto detect Baudrate **[Default]** |
| X | Set Baudrate to X bps. ( PCAP: 57600 , Resis: 9600 ) |
| ◆ ScanInterface | Choose scan interface |
| 0 | Scan all interface **[Default]** ( USB / RS232 / PS/2 ) |
| 1 | Scan USB interface only. |
| 2 | Scan UART interface only. |
| 3 | Scan PS/2 interface only. |
| ◆ UseDriverCalib | Use Driver Calibration |
| 0 | No Driver Calibration. [Default] |
| 1 | Use Driver Calibration. |
| ◆ SkipFirstByte | Skip First Byte with some old Linux system. |
| 0 | No Skip. [Default] |
| 1 | Skip First Byte. |
| ◆ **ScanDevStartDelayTime** | Driver booting delay time |
| 0 | No delay [Default] |
| 1 | Delay X millisecond to start driver. |
| ◆ ScanDevInterval | Scan Device Interval |
| 50 | 50ms [Default] |
| X | Delay X millisecond of Scan Device Interval. |
| ◆ RS232HWFlowCtl | RS232 HW Flow Ctrl |
| 1 | HW Flow Ctrl.[Default] |
| 0 | No HW Flow Ctrl. |
| ◆ SerialPath | RS232 Serial Path |
| default | Default path /dev/ttySX ( X could be equals to 0-10 ) **[Default]** |
| /dev/serial/ttyS0 | Customized path. Please type in your specific serial path accordinating to the form. |
| ◆ eGalaxToolPath | Pipe device node & Debug Log Path. |
| /data/eGalax | Default Pipe device node & Debug Log Path:/data/eGalax/ **[Default]** |
| ◆ DeviceNums | How many devices you want to plug-in to the system. If you want more than one device, please modify this value. |
| 1 | Only one device **[Default]** |
| 2-10 | More than one device. **[Max = 10]** |
| ◆ SupportPoints | The amount of points you want to report (This is also confined by Controller) |
| 0 | No point |
| 1 | Single-touch |

| | |
|---|---|
| >=2 | Multi-touch **[Default = 10]** |
| ◆ VirtualDevName | Set Input Virtual Device Name |
| null<br>X | null **[Default]**<br>X |
| ◆ Direction | Change the X and Y direction |
| 0<br>1<br>2<br>3<br>4 | Don't make any invert **[Default]**<br>Invert X<br>Invert Y<br>Invert both X and Y<br>Swap X and Y |
| ◆ Orientation | Change the orientation |
| 0<br>1<br>2<br>3 | 0 degree **[Default]**<br>90 degree<br>180 degree<br>270 degree |
| ◆ EdgeCompensate | Do edge compensate |
| 0<br>1 | Disable **[Default]**<br>Enable |
| EdgeLeft, EdgeRight<br>EdgeTop, EdgeBottom | Edge compensate value |
| X | If equals to 100, it means no change.<br>If you set Left=50, you'll see the left-edge points are shrinks inward. And vice versa. **[Min 50 - 150 Max] [Default = 100]** |
| ◆ HoldFilterEnable | Filter out constant touch or not |
| 0<br>1 | Disable **[Default]**<br>Enable |
| HoldRange | Constant touch valid area |
| X | ±X range of the point which would lead to constant touch<br>**[Min 0 - 50 Max] [Default = 10]** |
| ◆ SplitRectMode | Split the display into Specific Rect. Touch would just show on the specific Rect. |
| 0<br>1-8<br><br><br><br>9 | No change (Full Display) **[Default]**<br>Driver in-built split Rect<br>2 \| 1 / 3 \| 4 ; 5 / 6 ; 8 \| 7<br>Customized Rect. |
| CustomRectLeft<br>CustomRectRight | Theses parameters are valid as SplitRectMode=9. You can customize the Rect by these parameters. |

| | |
|---|---|
| CustomRectTop<br>CustomRectBottom | |
| 0-4095 | Four sides of the customized Rect |
| ◆ VKEYEnable | Enable this option if there's virtual key on your touch sensor. |
| 0<br>1 | Disable **[Default]**<br>Enable |
| VKEYReportMod | Virtual Key Sensitivity. |
| X | Smaller value means more sensitive. On the other hand, larger is less sensitive. |
| VKEY_**X** **Y** | 1. The value of **X** refer to the vkey package **reported by controller** as touching the specified virtual key on your sensor.<br>2. The value of **Y** refer to the **keyevent code in input.h** which you want to report to system. You can choose the keyevent you want and fill in the code. |
| Example A:<br>**VKEY_0 139** | 1. As controller report vkey package [0], we'll send keyevent code [139] to system.<br>2. The event code [139] in input.h refers to MENU_KEY.<br>Note: 139 is a decimal number. |
| Example B(Combo Key):<br>**VKEY_1 29+56+111** | 1. You can fill in Combo Key number in **Y**, separated by symble "+".<br>2. As controller report vkey package [1], eGTouchD will send keyevent code 29+56+111 to system(CTRL+ALT+Del).<br>3. Maximum support 5 Keys Combo.<br>The event code 29 in input.h refer to KEY_LEFTCTRL.<br>The event code 56 in input.h refer to KEY_LEFTALT.<br>The event code 111 in input.h refer to KEY_DELETE. |
| Note: If you're not sure controller's vkey package value, please contact EETI vendor. | |

# Sec 6: FAQ

## 6-1 Touch not working

1. **Check connection of controller.**

   For USB interface: **$ dmesg | grep eGalax**

   [PCAP example]

```
william@ubuntu:~$ dmesg | grep eGalax
[    2.535665] usb 3-2: Product: eGalaxTouch P80H60 -0738-01.00.00.00
[    2.535666] usb 3-2: Manufacturer: eGalax Inc.
[    3.135173] input: eGalax Inc. eGalaxTouch P80H60 -0738-01.00.00.00 Touchscreen as /devices/pci0000:00/
[    3.135240] input: eGalax Inc. eGalaxTouch P80H60 -0738-01.00.00.00 Mouse as /devices/pci0000:00/0000:0
[    3.135301] hid-generic 0003:0EEF:C002.0002: input,hiddev0,hidraw1: USB HID v1.11 Mouse [eGalax Inc. eG
[    5.793160] input: eGalax Inc. eGalaxTouch P80H60 -0738-01.00.00.00 as /devices/pci0000:00/0000:00:15.0
[    5.793233] input: eGalax Inc. eGalaxTouch P80H60 -0738-01.00.00.00 Mouse as /devices/pci0000:00/0000:0
[    5.793334] hid-multitouch 0003:0EEF:C002.0002: input,hiddev0,hidraw1: USB HID v1.11 Mouse [eGalax Inc.
```

   [Resistant touch example]

```
william@ubuntu:~$ dmesg | grep eGalax
[  271.583758] usb 3-2: Manufacturer: eGalax Inc.
[  271.598261] input: eGalax Inc. USB TouchController Mouse as /devices/pci0000:00/0000:00:15.0/0
[  271.659540] input: eGalax Inc. USB TouchController as /devices/pci0000:00/0000:00:15.0/0000:03
[  271.659597] input: eGalax Inc. USB TouchController as /devices/pci0000:00/0000:00:15.0/0000:03
[  271.659673] hid-multitouch 0003:0EEF:0001.0003: input,hiddev0,hidraw1: USB HID v1.00 Mouse [eG
```

   For UART interface: **$ dmesg | grep tty**

   Find system current ttyS inputs.

```
william@ubuntu:~$ dmesg | grep tty
[    0.097465] printk: console [tty0] enabled
[    0.810898] 00:05: ttyS0 at I/O 0x3f8 (irq = 4, base_baud = 115200) is a 16550A
[    0.835707] 00:06: ttyS1 at I/O 0x2f8 (irq = 3, base_baud = 115200) is a 16550A
```

   Use hexdump and touch screen, if connection is OK, it will print some information.

   **$ hexdump /dev/ttySX** , in this example, X should be 0 or 1.

```
william@ubuntu:~$ sudo hexdump /dev/ttyS1
0000000 0781 4742 0781 4741 0781 4741 0781 4840
```

   If you have trouble in this step, please check the hardware connection of your device and controller.

2. **Check eGTouch driver status.**

   **$ ps –ef | grep eGTouchD**

```
william@ubuntu:~$ ps -ef | grep eGTouchD
root      197212   79390  1 16:06 ttyS1    00:00:41 eGTouchD -f
william   197603  197331  0 17:04 pts/4    00:00:00 grep --color=auto e
```

   If eGTouchD is not running, please running eGTouchD manually. **$ eGTouchD -f**

   Check touch function, if touch is work, you may find the similar message as below in kernel log, in this case, please refer to Android Secure document, Writing SELinux Policy.

```
init: service eGTouchD does not have a SELinux domain defined
```

**3. Check input device has been created.**

**$ cat /proc/bus/input/devices | grep eGalax** To see is there any eGalaxTouch Virtual Device in list.

[PCAP example]

```
x86_64:/dev # cat /proc/bus/input/devices  | grep eGalax
N: Name="eGalax Inc. eGalaxTouch P80H60 -0738-01.00.00.00"
N: Name="eGalaxTouch_VirtualDevice"
N: Name="eGalaxTouch_VirtualPen"
```

[Resistant touch example]

```
x86_64:/dev # cat /proc/bus/input/devices  | grep eGalax
N: Name="eGalax Inc. USB TouchController Pen"
N: Name="eGalax Inc. USB TouchController"
N: Name="eGalaxTouch_VirtualDevice"
N: Name="eGalaxTouch_VirtualPen"
```

If you have trouble in this step, please check the required modules have been installed correctly. (Refer to **2.1, 2.2**)

If you are using UART interface, please also have a check to **6.3** section.

**4. Check input data flow.**

Use getevent to get the data from input device. **$ getevent –t -l**

```
x86_64:/dev/input # getevent -t -l
add device 1: /dev/input/event9
  name:     "eGalaxTouch_VirtualPen"
could not get driver version for /dev/input/mouse4, Not a typewriter
add device 2: /dev/input/event8
  name:     "eGalaxTouch_VirtualDevice"
could not get driver version for /dev/input/mouse3, Not a typewriter
add device 3: /dev/input/event6
  name:     "Android Power Button"
could not get driver version for /dev/input/mice, Not a typewriter
add device 4: /dev/input/event0
  name:     "Power Button"
add device 5: /dev/input/event1
  name:     "VMware VMware Virtual USB Mouse"
add device 6: /dev/input/event5
  name:     "PC Speaker"
add device 7: /dev/input/event2
  name:     "AT Translated Set 2 keyboard"
could not get driver version for /dev/input/mouse0, Not a typewriter
could not get driver version for /dev/input/mouse2, Not a typewriter
add device 8: /dev/input/event3
  name:     "VirtualPS/2 VMware VMMouse"
could not get driver version for /dev/input/js0, Invalid argument
add device 9: /dev/input/event4
  name:     "VirtualPS/2 VMware VMMouse"
could not get driver version for /dev/input/mouse1, Not a typewriter
[     1552.947715] /dev/input/event8: EV_ABS       ABS_MT_TRACKING_ID   00000000
[     1552.947715] /dev/input/event8: EV_ABS       ABS_MT_POSITION_X    0000066f
[     1552.947715] /dev/input/event8: EV_ABS       ABS_MT_POSITION_Y    00000495
[     1552.947715] /dev/input/event8: EV_SYN       SYN_REPORT           00000000
[     1552.998677] /dev/input/event8: EV_ABS       ABS_MT_TRACKING_ID   ffffffff
[     1552.998677] /dev/input/event8: EV_SYN       SYN_REPORT           00000000
```

Touch screen and check is there any data printing from screen.

If so, make sure all data are from one event and event number should correspond to “eGalaxTouch_VirtualDevice”, or you may check inbox touch driver has been blocked correctly. (Refer to **2.1, 2.2** section)

If you do not see data printing from screen, please refer to **7.1** section, describe your issue and contact us.

## 6-2 Touch screen can (NOT) wake up display

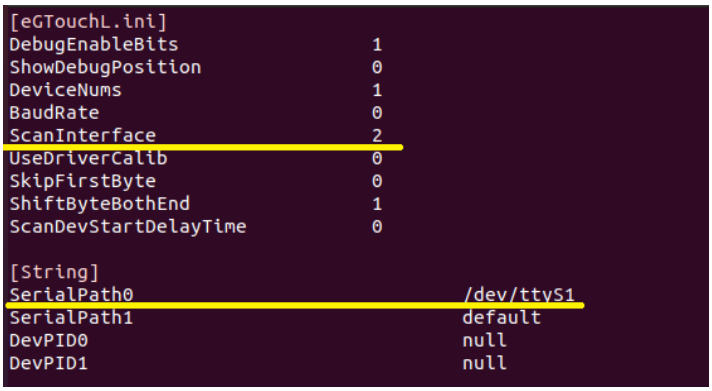
Android system will automatically detects input device capabilities based on the event types and properties, that are reported by the associated Linux kernel input device driver. Touch devices usually need IDC files to specify the property of input device, refer to the documentation definition: *device.internal = 0 | 1* specifies the device is external or internal. Internal input devices generally do not wake the display from sleep and external input devices usually wake the device more aggressively.

The default setting of input device of *eGalaxTouch_VirtualDevice* is **external**, please modified **the value of *device.internal*** depend on your requirement.

If you want to get more information, you can refer to Android Develop document, Input Device Configuration Files.

## 6-3 eGTouchD can NOT find UART interface device

Check the setting of ScanInterface and SerialPath in **eGTouchA.ini**, please try to change ScanInterface setting to **2** and assign serial path with your input path, the definition of ScanInterface refer to section 5, to figure out serial path can refer to 6.1 section.




After modifying the parameter, please reboot your system or restart driver to make it valid.

## 6-4 My UART device receive unexpected data from eGTouchD

As default, eGTouchD will scan USB and UART interface to find touch controller, eGTouchD use Vendor ID (VID) and Product ID (PID) to interpret USB interface touch device and send some data to interpret UART interface touch device, this issue can be avoided by the below setting.

If you are using USB interface touch controller, please modify the ScanInterface to **1** in **eGTouchA.ini**, than eGTouchD will skip to scan UART interface.

If you are using UART interface touch controller, please refer to 6.3 section to specify ScanInterface and SerialPath in **eGTouchA.ini**, eGTouchD will only scan specified serial path. After modifying the parameter, please reboot your system or restart driver to make it valid.

# Sec 7: Support

## 7-1 Need Support From EETI

If you have any problems when running the eGTouchD driver and the above FAQs still cannot solve your problem, please help to collect debugging information and provide a description of your issue. Collecting required information according to the following steps can help us understand your problem and provide assistance as soon as possible.

Please run **eetiGetInfoA.sh** script as root to collect debugging information, the information may include your system information, kernel log, driver setting and driver information, we only use these information to assist your question. Also there are **WindowsGetInfo.bat** and **LinuxGetInfo.sh** can help you get debugging information in one step if you are using windows or Linux system, just assure you have installed adb driver and have root authorization.

This may takes few seconds, and you will find an **eeti.tar.gz** file in your current path.

Please attach **eeti.tar.gz**, describe your issue and contact us by mail: **touch_fae@eeti.com**

Or provide us below information and the description of the problem you encountered for us.

## 7-2 Environment Information

Fill in this chart

| | |
|---|---|
| 1. CPU type | |
| 2. Kernel version | |
| 3. Android version. | |
| 4. Touch Controller Interface | |
| 5. Touch Controller Type | |

## 7-3 Register input devices

1. Execute command `cat /proc/bus/input/devices`
2. Send us the output result.

## 7-4 Driver debug log

1. Modify file eGTouchA.ini. Change the value of the parameter DebugEnableBits from 1 to FFFFF. Change the value of ShowDebugPosition from 1 to 0.

As below

```
[eGTouchA.ini]
DebugEnableBits         FFFFF
ShowDebugPosition       0
```

2. Reboot your system. After rebooting, please touch four corner of the touch panel.
3. Default log file would be printed in /data/eGalax/eGTouch_[year]_[date]_[time].log
4. You may see one more log named eGTouch_[year]_[date]_[time].log. Please send us the Newest one for analyzing. Thanks.