How to configure touch screen for multi-monitor

This document is used to guide users to configure touch screen for multi-monitor. Users should set multi-monitor with extension mode before follow below steps to make touch screen working.

1.) Touch screen X configuration:

EETI touch driver supports one or more touch screens. If users only have one touch screen, no extra setting is requested. The configuration section could be ignored.

1. **Create X configuration file:**
   If **/etc/X11/xorg.conf** dose not exist by default, users should issue below command to create X configuration file manually.

   X –configure :1
   cp /root/xorg.conf.new /etc/X11/xorg.conf

   Note that touch driver must be removed and then install again after the file created.

2. **Assign real device node:**
   A. For serial and PS/2 interface:
      If using serial interface, **the "Device" option of touch configuration should be assigned to "/dev/ttySX"**, X means a number kernel assigned to serial device.
      If using P/S2 interface, **the "Device" option of touch configuration should be assigned to "/dev/serio_rawX"**, X means a number kernel assigned to PS/2 device.
      Users must assign correct device node, otherwise touch screen will not work properly. It is advised to issue **"cat"** command like **"cat /dev/ttyS0"** as root and check whether any output on the screen while touch screen is pressed. If there is any data on the screen, it means the device node is correct.

   B. For USB interface:
      Totally, there are two kinds of USB type. Users should issue **"lsusb –v –d 0eef:"** command to check whether USB touch screen supports HID or not. See below example.

[HID Interface Descriptor]

bLength 9

bDescriptorType 4

bInterfaceNumber 0

bAlternateSetting 0

bNumEndpoints 1

bInterfaceClass 3 Human Interface Device

bInterfaceSubClass 0 No SubClass

bInterfaceProtocol 0 None

iInterface 0

[Vendor Interface Descriptor]

bLength 9

bDescriptorType 4

bInterfaceNumber 0

bAlternateSetting 0

bNumEndpoints 1

bInterfaceClass 255 Vendor Specific Class

bInterfaceSubClass 255 Vendor Specific Subclass

bInterfaceProtocol 255 Vendor Specific Protocol

iInterface 0

If USB touch screen is HID compatibility device, it is highly recommended to use HID device node like **"/dev/usb/hiddevX"** or **"/dev/hidrawX"** instead of **"/dev/input/eventX"** in order to get better compatibility. Users can issue **"find /dev/ -name hid*"** (to search all of hid device nodes) or **"find /dev/ -name event*"** (to search all of event input device nodes) to list all of input device nodes, then issue **"cat"** command like **"cat /dev/usb/hiddev0"** as root and check whether any output on the screen while touch screen is pressed. If there is any data on the screen, it means the device node is correct. **Note that the "Device" option of touch configuration must be assigned to correct device node, otherwise touch screen will not work properly.**

After checked correct device node, users must modify **/etc/X11/xorg.conf** and update all of **"Device"** options of touch configuration with correct device node. See below configuration example for two touch screens.

```
Section "ServerLayout"
    InputDevice "EETI1" "SendCoreEvents"
    InputDevice "EETI2" "SendCoreEvents"
    Identifier    "X.org Configured"
    Screen     0  "Screen0" 0 0
    InputDevice   "Mouse0" "CorePointer"
    InputDevice   "Keyboard0" "CoreKeyboard"
EndSection

### Touch Configuration Beginning ###
Section "InputDevice"
Identifier "EETI1"
Driver "egalax"
Option "Device" "/dev/hiddev1"
Option "Parameters" "/var/lib/eeti1.param"
Option "ScreenNo" "0"
EndSection
### Touch Configuration End ###

### Touch Configuration Beginning ###
Section "InputDevice"
Identifier "EETI2"
Driver "egalax"
Option "Device" "/dev/hiddev2"
Option "Parameters" "/var/lib/eeti2.param"
Option "ScreenNo" "0"
EndSection
### Touch Configuration End ###
```

Note that the system must be restarted for some changes to take effect.

3. Create udev rule to fix USB device node:

Sometimes, the device node of USB touch screen assigned by kernel may be different after reboot. This will cause touch screens working abnormally. In this situation, users should create an udev rule in **/etc/udev/rules.d/** to fix USB working device node to solve this problem. See below example.

[69-eGalaxUSB.rules]

#----------

The file should be copied to **/etc/udev/rules.d**

KERNEL=="hidraw*", KERNELS=="3-1", SYSFS{idVendor}=="0eef", SYSFS{idProduct}=="0001"**, SYMLINK+="hidraw18"**
KERNEL=="hidraw*", KERNELS=="3-2", SYSFS{idVendor}=="0eef", SYSFS{idProduct}=="0001"**, SYMLINK+="hidraw19"**

#----------

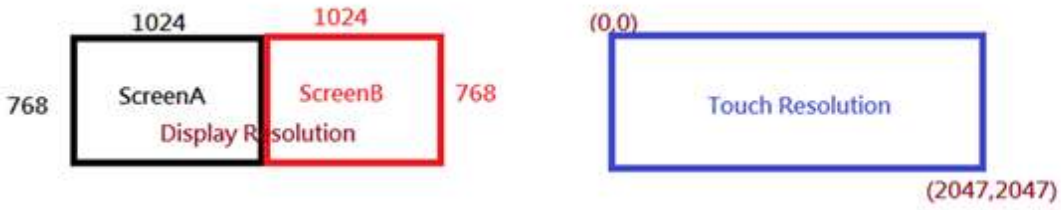
The above example assumes using HIDRAW device node and USB touch screens connected to USB port “3-1” and “3-2”.

Users should issue **“udevinfo”** or **“udevadmin”** command to see related udev information of touch screen and set above parameters accordingly. After updated the udev rule, users should also update all of **“Device”** options of touch configuration in **/etc/X11/xorg.conf**. Otherwise the problem will still persist.

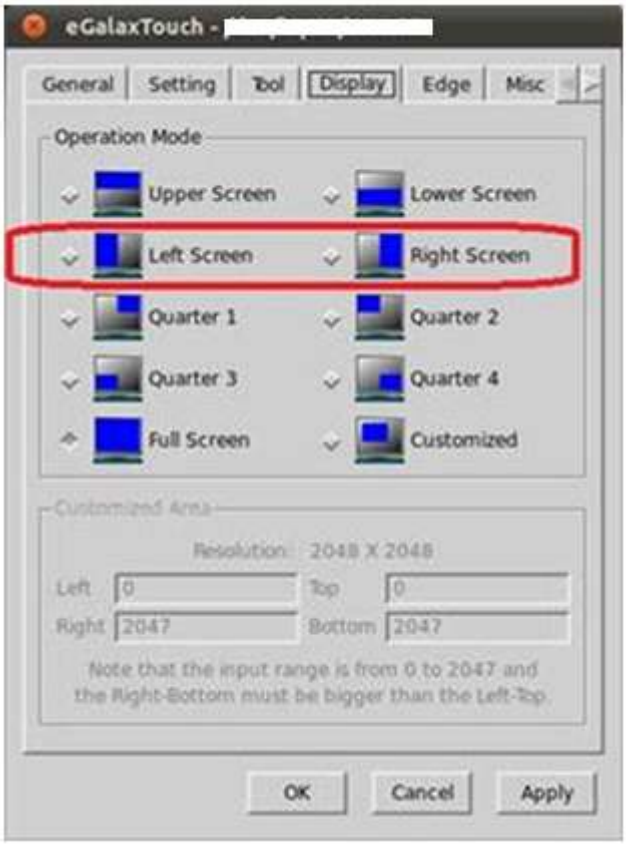
2.) Touch screen mapping algorithm

EETI touch utility offers display function used to set working area of touch screen. Users should check running display resolution and calculate correct working area. Then, update the working area to make touch screen working. See below two examples for more information.

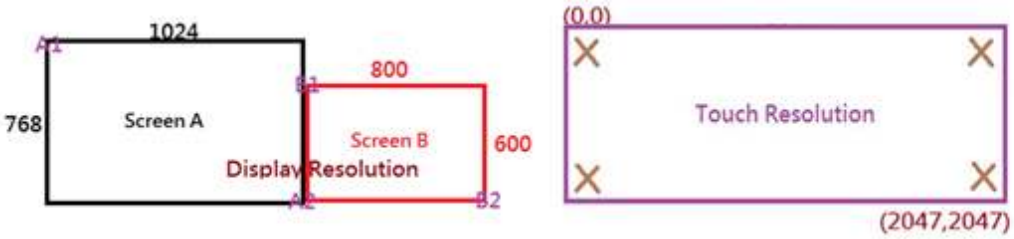
1. Two screens use the same display resolution.
   It is assumed that the running display resolutions both are 1024x768.
   See below figure.



Users can choose any options to define where the touch screen will be mapped via touch utility. For above example, users should choose Left or Right Screen. Note that users need clicking [Apply] button to validate new setting.



2. Two screens use different display resolutions.
   It is assumed that the display resolution of Left screen is 1024x768 and Right screen is 800x600. See below figure.



EETI touch driver always assumes touch resolution is 2048x2048. If running display resolution is different from touch resolution, users have to manually scale working area of touch screen to make touch screen working. See below for more information about that.

[Screen A]

A1: (0, 0) - A2: (1148, 2047)

A2:

$$X = \left(\frac{1024}{(1024+800)} * 2048\right) - 1 = 1148$$

$$Y = \left(\frac{768}{768} * 2048\right) - 1 = 2047$$

[Screen B]

B1: (1148, 447) - B2: (2047, 2047)

B1:

$$X = \left(\frac{1024}{(1024+800)} * 2048\right) - 1 = 1148$$

$$Y = \left(\frac{768-600}{768} * 2048\right) - 1 = 447$$

After calculated working area of touch screen according to the above algorithm, users should choose customized mode and update new working area accordingly via touch utility. Finally, click [Apply] button to validate new working area.

