TouchKit driver user guide for Linux

The TouchKit driver archive contains a pre-compiled X module and utility for X window system. This driver supports RS232, PS/2 and USB TouchKit controllers. All of versions support RS232 and USB TouchKit controllers. The X module version later than 1.06 supports PS/2 TouchKit controller.

1. Installation of the X module:

To install X module, user has to copy the X module to the X input modules directory and then configure the X configuration file.

a.) copy the X module

User can use the following command to give you a clue as to where the X modules are located on your system:

find / -name mouse_drv.o

Output:

/usr/X11R6/lib/modules/input/mouse_drv.o

If the using X window version is 4.x, copy the X module **"egalax_drv.o"** to the correct X input modules directory.

For example:

cp egalax_drv.o /usr/X11R6/lib/modules/input

User can run **"X –version"** command to check your running version of X window.

Output:

XFree86 Version 4.3.0 (Red Hat Linux release: 4.3.0-2)

Release Date: 27 February 2003

X Protocol Version 11, Revision 0, Release 6.6

…

b.) configure the Xorg configuration file

Edit the X configuration file (e.g. /etc/X11/XF86Config) and add the configuration used by the driver to connect to the device installed on your system.

(1) Add an Input device declaration in "ServerLayout" section.
For example:

```
Section "ServerLayout"
    …
    …
    InputDevice    "EETI"    "SendCoreEvents"
EndSection
```

Note: *If more than one TouchKit controllers (2 or more TouchKit touchscreens) are used for the system, the user must add multiple InputDevice declarations in the "ServerLayout" section with different names.*

For example:

```
InputDevice    "EETI1"    "SendCoreEvents"
InputDevice    "EETI2"    "SendCoreEvents"
```

(2) Configure the X module configuration for TouchKit device.
For each InputDevice section declared in the "ServerLayout" section, the user will need to create additional separate configuration in the XF86Config file.

For only one USB device in the system:

```
Section "InputDevice"
    Identifier  "EETI"
    Driver      "egalax"
    Option      "Device"        "usbauto"
    Option      "Parameters"    "/var/lib/eeti.param"
    Option      "ScreenNo"      "0"
EndSection
```

Note: *The Identifier line must be the same as the name declared it in the section "ServerLayout".*

For multiple devices (RS232 and USB) in the multiple monitors system:

```
Section “InputDevice”          ( For RS232 device )
    Identifier  “EETI1”
    Driver      “egalax”
    Option      “Device”       “/dev/ttyS0”
    Option      “Parameters”   “/var/lib/eeti1.param”
    Option      “ScreenNo”     “0”
EndSection

Section “InputDevice”          ( For USB device )
    Identifier  “EETI2”
    Driver      “egalax”
    Option      “Device”       “usbauto”
    Option      “Parameters”   “/var/lib/eeti2.param”
    Option      “ScreenNo”     “1”
EndSection
```

Note: *If more than one TouchKit controllers (2 or more TouchKit touchscreens) are used for the system, the user must edit each option “Parameters” with different file names.*

Driver

The Driver line must be set to correct X module name which is copied to the X input modules directory. For example:

```
Driver      ”egalax”
```

Option “Device”

The “Device” option must be assigned so that the driver can read the data from the device node. The device node is a char device usually found in /dev. If the “Device” is set to a pipe, the driver will not work correctly. The user must determine where the controller was connected to set the “Device” option.

Note: *The driver supports three interfaces, serial RS232, PS/2 and USB so that all users need to indentify which interface controller was connected in the system before set the option “Device”.*

1.) For serial RS232 interface:

The “Device” should be set to correct name of serial device node, e. g. **/dev/ttyS0** or **/dev/ttyS1**. Besides, the user must ensure the I/O address and the IRQ number are the same as the BIOS setting. For details about checking these settings, take a reference to the following command.

setserial /dev/ttyS0 –a

Some users connected the serial RS232 touch device on /dev/ttyS4 or /dev/ttyS5 and the IRQ setting for serial touch device is NOT available. In this case the user can use **IRQ 0** for serial touch device by the following command so that the system will use polling instead of interrupt to access the serial touch device.

setserial /dev/ttyS4 irq 0

2.) For PS/2 interface:

The “Device” should be set to correct name of PS/2 auxiliary device node, e. g. **/dev/psaux**. The user must make sure that the using X module version is later than 1.06, otherwise the driver will NOT wok properly.
Note that the PS/2 device should be connected correctly before power on.

3.) For USB interface:

There are two kernel modules support USB TouchKit device.
(1) Inbuilt hid module for **HID touch device**.
(2) TouchKit tkusb module for **both HID and non-HID touch devices**.

If the system has only one USB TouchKit device and the version of X module is 1.08 or later, the “Device” option can be set to **“usbauto”** so that the X module will attempt to determine the communication device node automatically. It might be better to manually configure the “Device” option declaration by user.
For example:

Option “Device” “usbauto”

The user can run "**cat /proc/bus/usb/devices"** to identify the USB touch device class and check which kernel module is loaded for touch device.

Part of output:

P: Vendor=0eef ProdID=0001 Rev=1.00 **(HID compliant device)**

S: Product=USB TouchController

I: If#= 0 Alt=0 #EPs=1 Cls=03(**HID**) Sub=00 Prot=00 Driver=**hid**

P: Vendor=0eef ProdID=0001 Rev=1.00 **(non-HID compliant device)**

S: Product=USB TouchController

I: If#= 0 Alt=0 #EPs=1 Cls=ff(**vend**.) Sub=ff Prot=ff Driver=**TouchKit**

Note: *If the inbuilt **hid** kernel module is loaded for HID touch device, there is always a default mouse driver which will run simultaneously with TouchKit driver if the user does NOT prevent the mouse driver from reading. It is extremely important that set the "Device" option for mouse to a real mouse device node like "/dev/input/mouseX" instead of default device node "/dev/input/mice". Otherwise, the user will get double click events and all kind of strange things. The user can run the command "**cat /proc/bus/input/devices"** to check which real mouse device node is used for mouse.*

Part of output:

N: Name="ImPS/2 Generic Wheel Mouse"

P: Phys=isa0060/serio1/input0

S: Sysfs=/class/input/input2

*H: Handlers=**mouse1** event2*

*Part of X configuration file: e.g. **/etc/X11/XF86Config***

```
Section "InputDevice"
        Identifier    "Mouse0"
        Driver        "mouse"
        Option        "Device" "/dev/input/mouse1"
        …
EndSection
```

If the above section does NOT exist in the X configuration file, the user has to append it manually.

3-1) **Use inbuilt HID kernel module:**

The Linux kernel 2.4 supports HID compliant TouchKit device with inbuilt **hid** kernel module. If the user is working with HID compliant TouchKit device and **hid** kernel module, there should be a device node for the HID compliant TouchKit device in /dev or /dev/usb. The user needs to identify which hiddevX device node represents the HID compliant TouchKit device and set the proper "Device" option.

For example:

Option "Device" "/dev/hiddev0"

Note*: If the system has only one HID compliant TouchKit device, the "Device" option for Xorg module version 1.06 or later can be set to*

Option "Device" "hiddev*"

or ***Option "Device" "hiddevs"***

so that the Xorg module will determine HID device node for HID compliant TouchKit device automatically.

3-2) **Use TouchKit USB kernel module:**

If vendor provided USB kernel module **tkusb.o** was loaded and the device node **/dev/tkpanel0** were created for USB TouchKit device, this "Device" option should be set to **/dev/tkpanel0** as follow.

Option "Device" "/dev/tkpanel0"

For details about building the TouchKit USB kernel module **tkusb.o**, see another document **"How to build module"**.

Option "Parameters"

The user can assign a writeable file path for the driver to save the parameters. All of the control parameters will be saved in this file. A separate file will be needed for a TouchKit device. It is recommended the files are saved in the variable library directory. For example:

Option "Parameters" "/var/lib/eeti.param"

Option "ScreenNo"

The user can define which screen number the TouchKit touchscreen will work with. If the system has only one TouchKit touchscreen, this value should be set to "0". For example:

Option "ScreenNo" "0"

Option "SkipClick"

When working touch controller is **HID compliant device** and the using option "Device" for mouse is **"/dev/input/mice"**, this option could be appended as example below in the Xorg configuration file (e.g. /etc/X11/XF86Config) to avoid conflict of mouse click. But, this will cause the **Click On Release** function abnormally. A way to solve this issue, please see page 5 for details. **Note that this option is used for HID compliant touch controller only.**

For example:

Option "SkipClick" "1"

c.) Restart X window

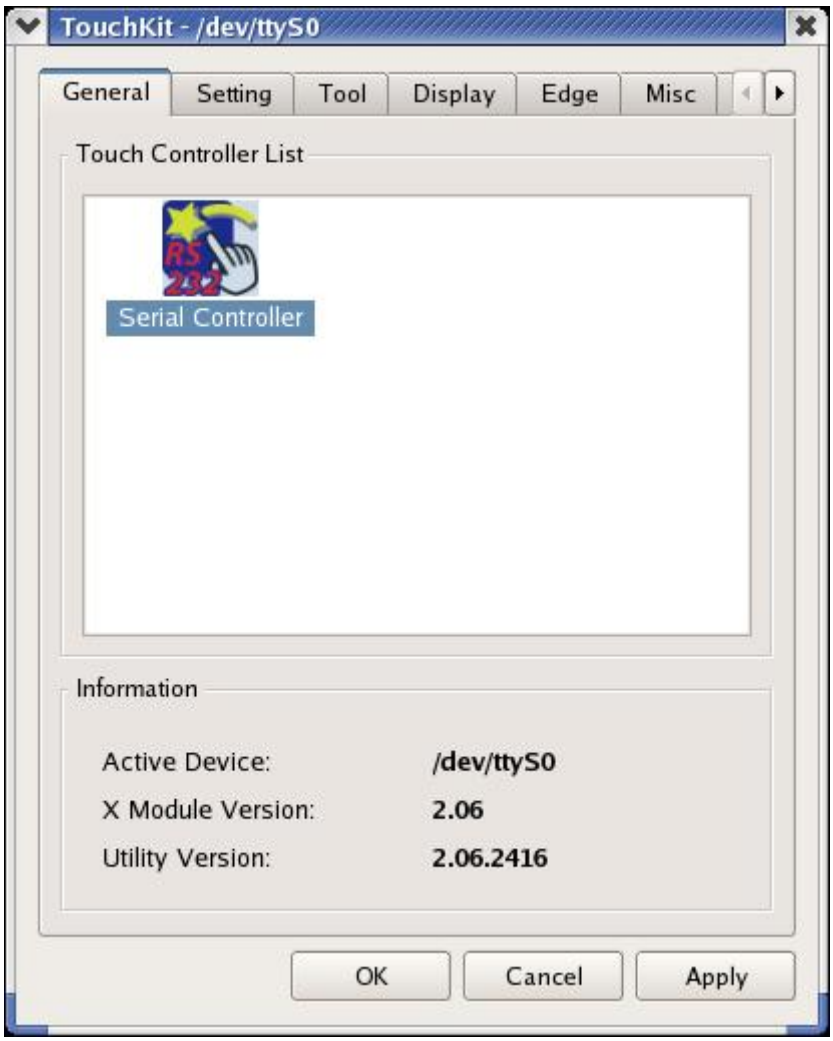
Restart X window to make sure the X module is loaded for TouchKit controller.

2. Utility

TouchKit driver archive for X window provides all users with a configuration tool utility for TouchKit touchscreen. The utility contains property pages **General, Setting, Tool, Display, Edge, Misc** and **About**.

Note: *Please make sure the X module is installed successfully. Otherwise, the utility does not work properly.*

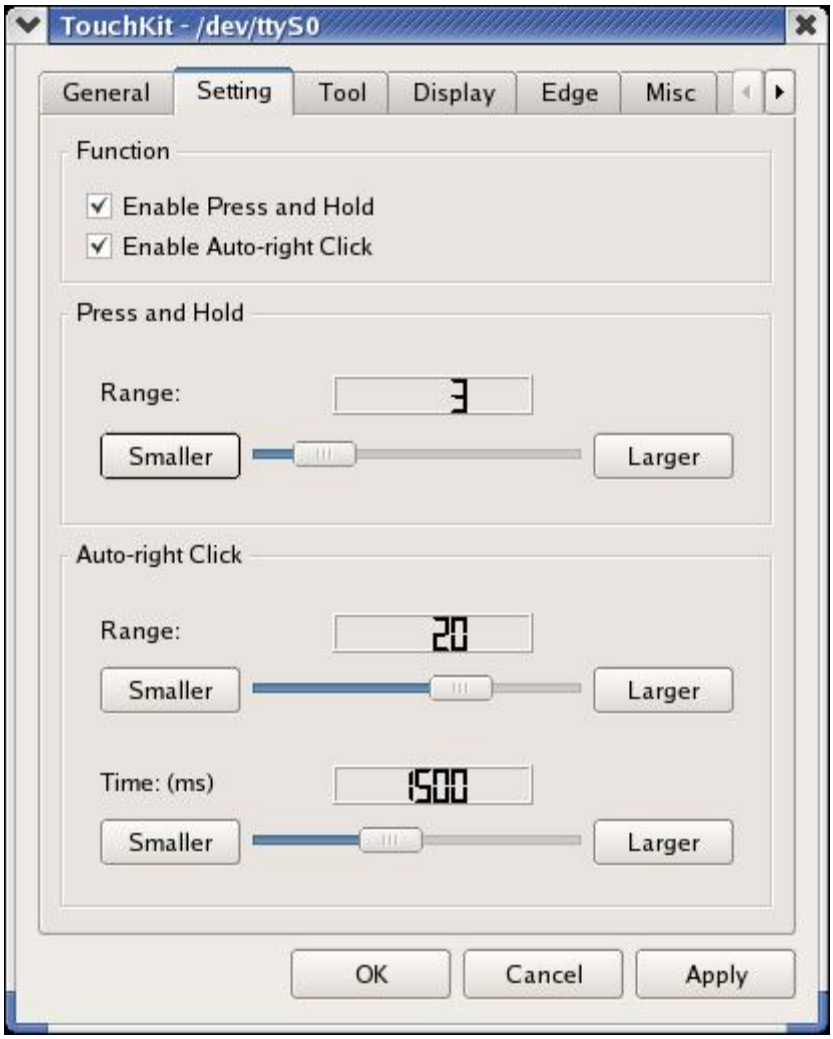
2.1) General Property:



The utility enumerates TouchKit touchscreen controller installed in this system. All of the enumerated TouchKit controllers will be listed in the “Touch Controller List” Window.

It also shows device name which active device node the device is connected. In addition, the X module and utility versions will be shown in the Information window as well.

3.2) Setting Property:



Some options can be configured for mouse emulation.

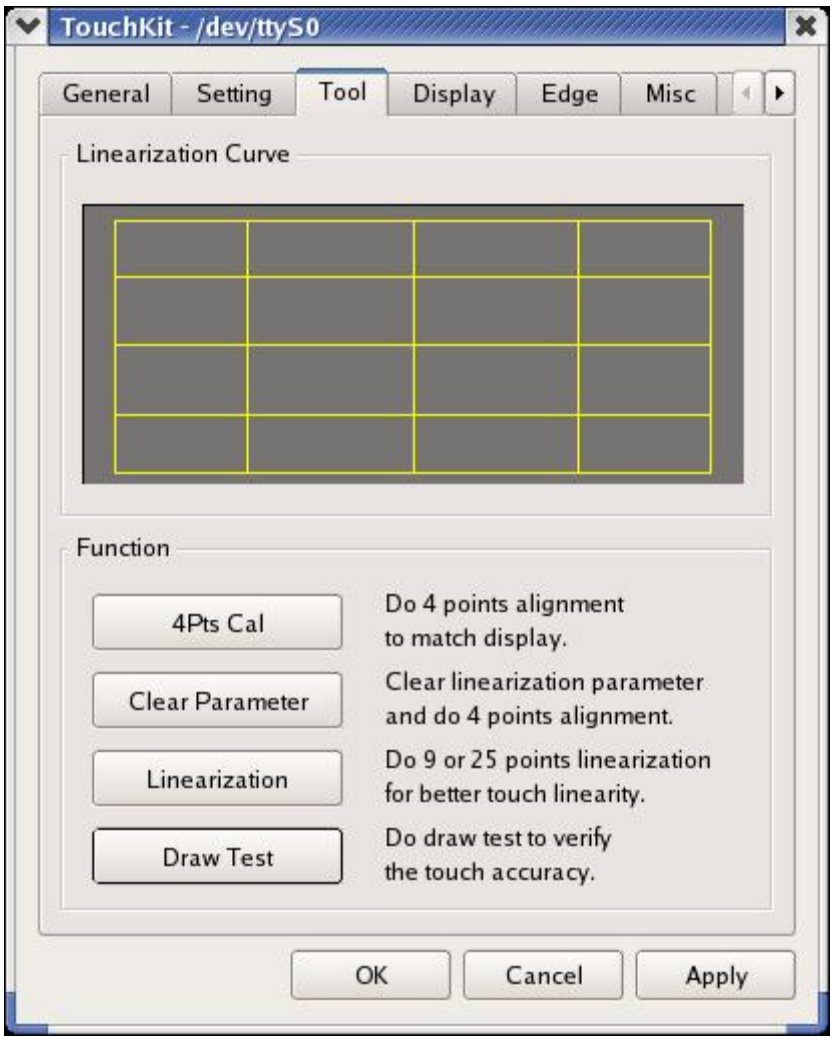
Press and Hold:

Press and hold at the same point. In some applications, the application program does not want to receive too many touch points for the touch held at same position, the user can check the checkbox to enable constant touch function so that the driver would not report other points unless the position difference between current position and last position is greater than the **Range** value or lift up. The range of the point difference can be configured with the **Range** slider.

Auto Right Click:

The driver generates a mouse right click event automatically whenever the driver detects the touch was press and hold for a while if the checkbox checked. The duration and the range for auto-right click emulation can be configured with the **Range** slider and the **Time** slider.

3.3) Tool Property:



TouchKit utility provides all users with tools for calibration and testing.

Linearization Curve:

After linearization finished, the linearity of the touchscreen will be shown in this linearization curve.

4 Pts Calibration:

TouchKit utility provides 4 points calibration for touchscreen alignment. The touchscreen can work correctly only after calibration. When the user presses this **"4Pts Cal"** button to do 4 points calibration, a calibration window will pop up to guide user to complete the calibration.

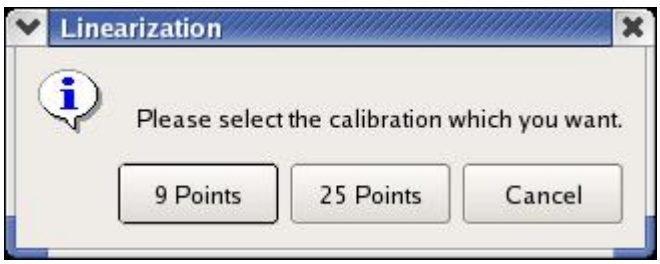
The user should **press the calibration symbol until it goes to next point or disappears**. The user can abort this calibration by pressing **<ESC>** key.

Clear Parameter:

Press this button to clear the linearization parameters and do 4 points calibration. All of the linearization parameters will be cleared if the button pressed.

Linearization:

Press this button to do linearization, a message box as below will pop up to hint user to select 9 points or 25 points calibration.

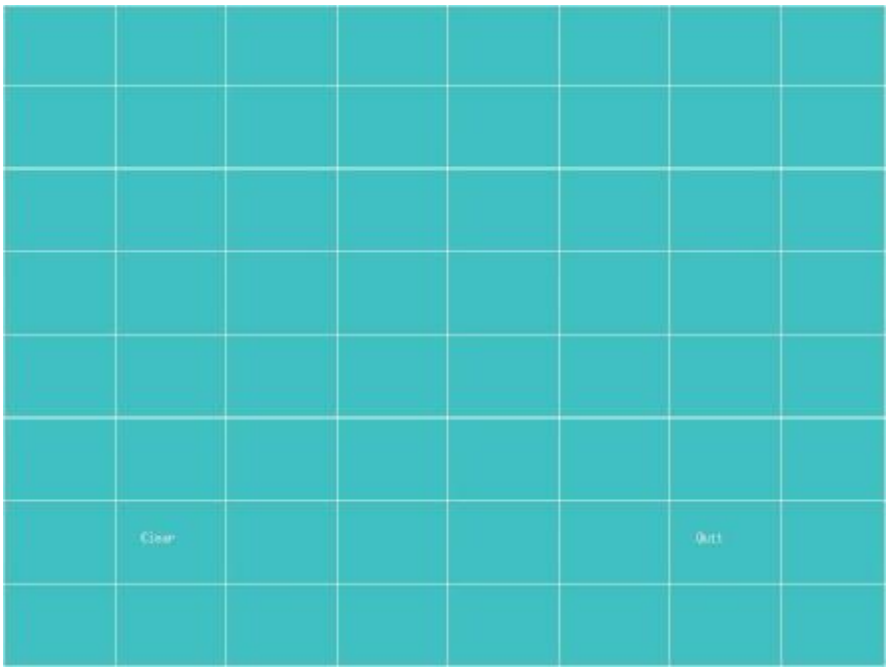


9 Pts / 25 Pts Linearization:

After 9 Pts or 25 Pts linearization, the previous linearization parameters will be overwritten by the new parameters.

Draw Test:

After linearization or alignment, the user can press this button to check the touch accuracy, linearization, response, etc…



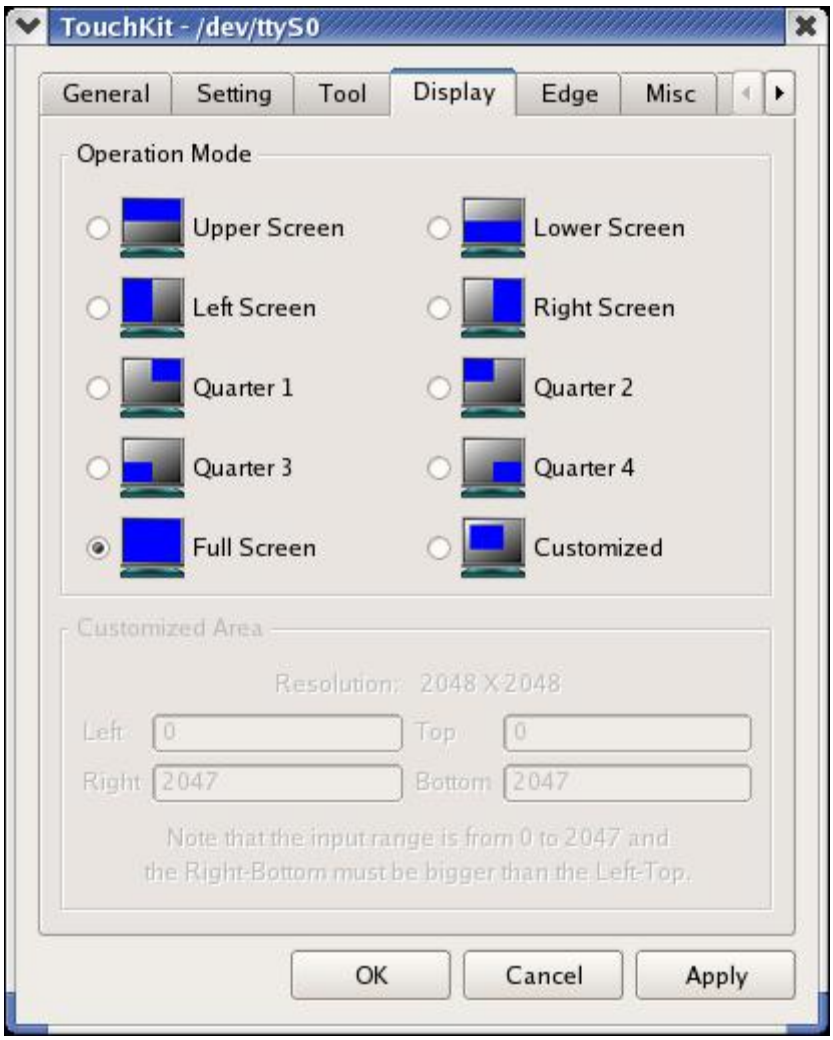
[Clear]:

The user can touch within the rectangle of **[Clear]** to clear the window of draw test.

[Quit]:

The user can quit this window by touching within the rectangle of **[Quit]**, or pressing **[ESC]** key.

3.4) Display Property:

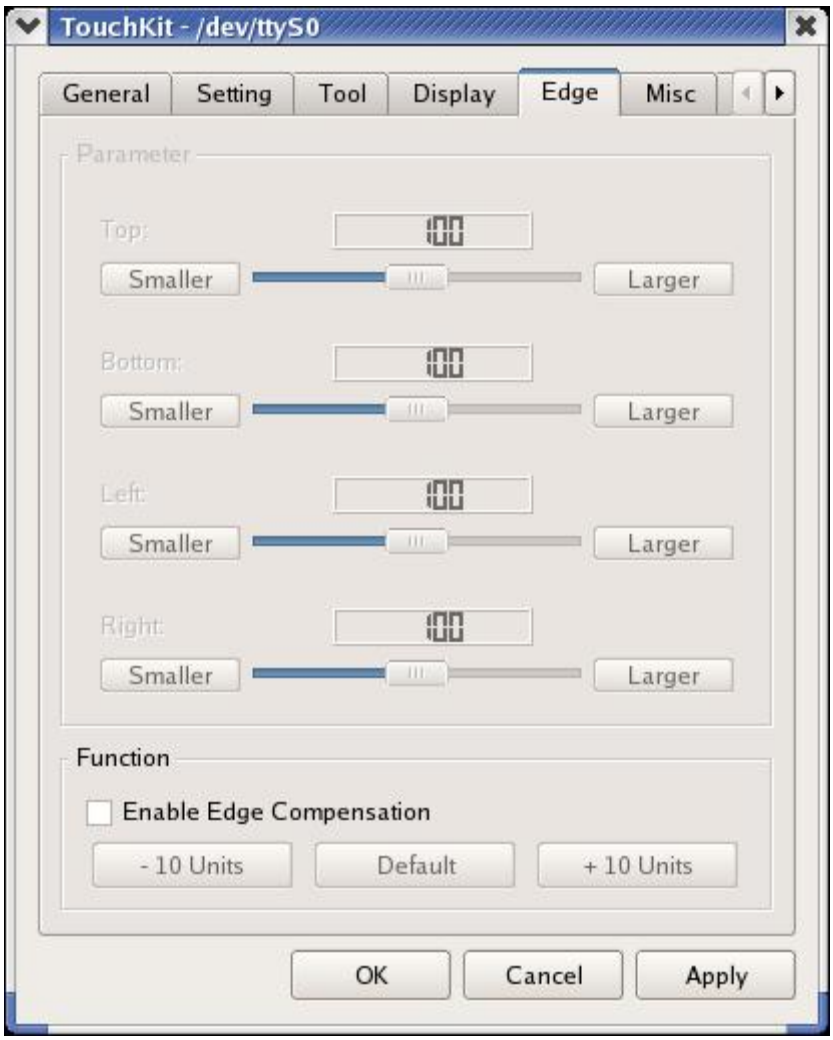


TouchKit utility supports split display feature.

The working area of the touchscreen can be mapped to anywhere on the video display.

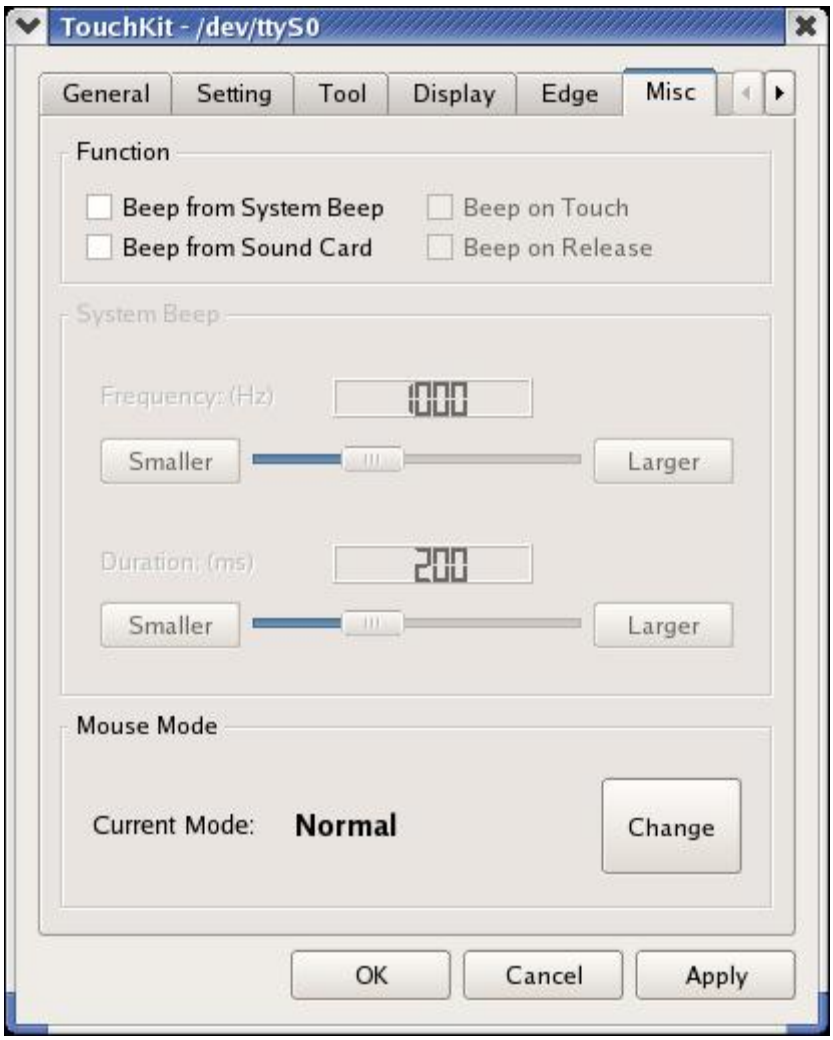
The user can choose any options to define where the touchscreen will be mapped. However, if the **Customized** is selected, it needs to enter the area to map to. The TouchKit utility always **assumes the resolution is 2048 X 2048**. **If the video resolution is not 2048 X 2048, the user has to calculate the area manually**.

3.5) Edge Property



TouchKit utility supports edge compensation to make sure that the touchscreen can achieve the display edge area.

3.6) Misc Property



Beep from System Beep:

When this function enabled, the driver will generate a beep sound from **system beep** if **Beep On Touch** or **Beep On Release** checkbox is checked.

Beep from Sound Card:

When this function enabled, the driver will generate the beep sound from **sound card** if **Beep On Touch** or **Beep On Release** checkbox is checked.

Beep On Touch:

When this checkbox is checked, the driver will generate a beep sound whenever it detects the touch state changed from untouched state to touched state.

Beep On Release:

When this checkbox is checked, the driver will generate a beep sound whenever it detects the touch state changed from touched state to untouched state.

Frequency:

Change this **Frequency** value to change the frequency of the system beep.

Duration:

Change this **Duration** value to change the duration of the system beep.

Note that this feature is NOT supported in some Linux distributions.

Mouse Emulation Mode:

TouchKit driver supports three mouse emulation modes.

1.) **Normal Mode:**

The touch driver reports **a left button down event** when it detects a pen down and **a left button up event** when it receives a lift off.

2.) **Click On Touch:**

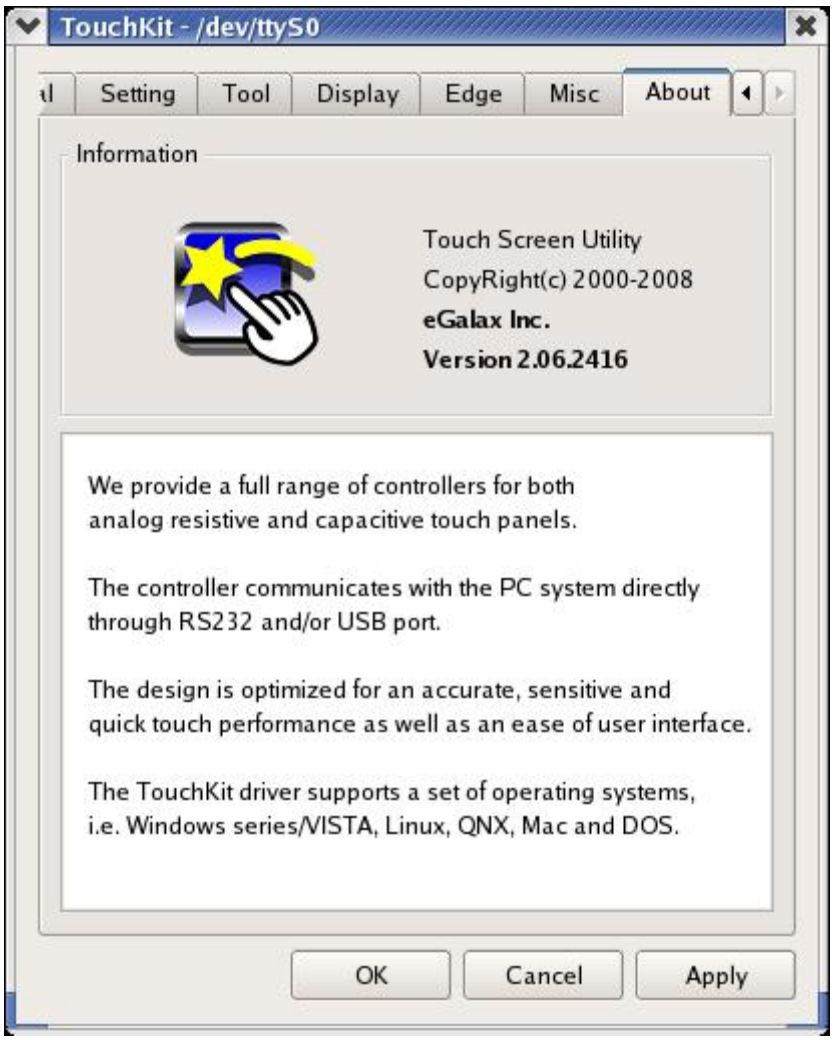
The touch driver reports **a left button click event** when it detects a pen down. Then, it does not report other events until it detects next a pen down.

3.) **Click On Release:**

The touch driver does not report any event until it detects a lift off. It reports **a left button click event** when it detects a lift off.

Note: *If the mouse emulation mode is changed to **Click On Touch** or **Click On Release**, the features **Press and Hold** and **Auto-right Click** will be **disabled** automatically.*

3.7) About Property



Information about TouchKit.